

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

| | |
|--|--|
| <ul style="list-style-type: none"> 1. DESCRIPTION 1 <ul style="list-style-type: none"> 1.1 Features 1 1.2 Applications 1 1.3 Pin Configuration 3 1.4 Block Diagram 4 1.5 Performance Outline 5 2. OPERATION OF FUNCTIONAL BLOCKS 9 <ul style="list-style-type: none"> 2.1 Memory 9 2.2 Central Processing Unit (CPU) 15 2.3 Reset 18 2.4 Processor Mode 23 2.5 Clock Generating Circuit 36 2.6 Protection 46 2.7 Interrupts 47 2.8 Watchdog Timer 67 2.9 DMAC 69 2.10 Timer 79 2.11 Serial I/O 99 2.12 A-D Converter 149 2.13 D-A Converter 164 2.14 Data Slicer 166 2.15 HSYNC Counter 176 2.16 OSD Function 177 <ul style="list-style-type: none"> 2.16.1 Triple Layer OSD 183 2.16.2 Display Position 185 2.16.3 Dot Size 189 2.16.4 Clock for OSD 190 2.16.5 Field Determination Display 191 2.16.6 Memory for OSD 193 2.16.7 Character Color 206 2.16.8 Character Background Color 206 2.16.9 OUT1, OUT2 Signals 211 2.16.10 Attribute 212 2.16.11 Automatic Solid Space Function 217 2.16.12 Particular OSD Mode Block 218 2.16.13 Multiline Display 220 2.16.14 SPRITE OSD Function 221 2.16.15 Window Function 224 2.16.16 Blank Function 225 2.16.17 Raster Coloring Function 228 2.16.18 Scan Mode 230 2.16.19 R, G, B Signal Output Control ... 230 2.16.20 OSD Reserved Register 231 2.17 Programmable I/O Ports 232 | <ul style="list-style-type: none"> 3. USAGE PRECAUTION 245 <ul style="list-style-type: none"> 3.1 Timer A (timer mode) 245 3.2 Timer A (event counter mode) 245 3.3 Timer A (one-shot timer mode) 245 3.4 Timer A (pulse width modulation mode) 245 3.5 Timer B (timer mode, event counter mode) 246 3.6 Timer B (pulse period/pulse width measurement mode) 246 3.7 A-D Converter 246 3.8 Stop Mode and Wait Mode 246 3.9 Interrupts 247 3.10 Built-in PROM version 248 4. ITEM TO BE SUBMITTED WHEN ORDERING MASKED ROM VERSION 249 5. ELECTRICAL CHARACTERISTICS 250 <ul style="list-style-type: none"> 5.1 Absolute Maximum Ratings 250 5.2 Recommended Operating Conditions .. 251 5.3 Electrical Characteristics 252 5.4 A-D Conversion Characteristics 253 5.5 D-A Conversion Characteristics 253 5.6 Analog R, G, B Output Characteristics 253 5.7 Timing Requirements 254 5.8 Switching Characteristics 255 5.9 Measurement Circuit 259 5.10 Timing Diagram 260 6. MASK ROM CONFIRMATION FORM 265 7. MARK SPECIFICATION FORM 269 8. ONE TIME PROM VERSION M306V2EEFP MARKING 270 9. PACKAGE OUTLINE 271 |
|--|--|

1.3 Pin Configuration

Figure 1.3.1 shows the pin configuration (top view).

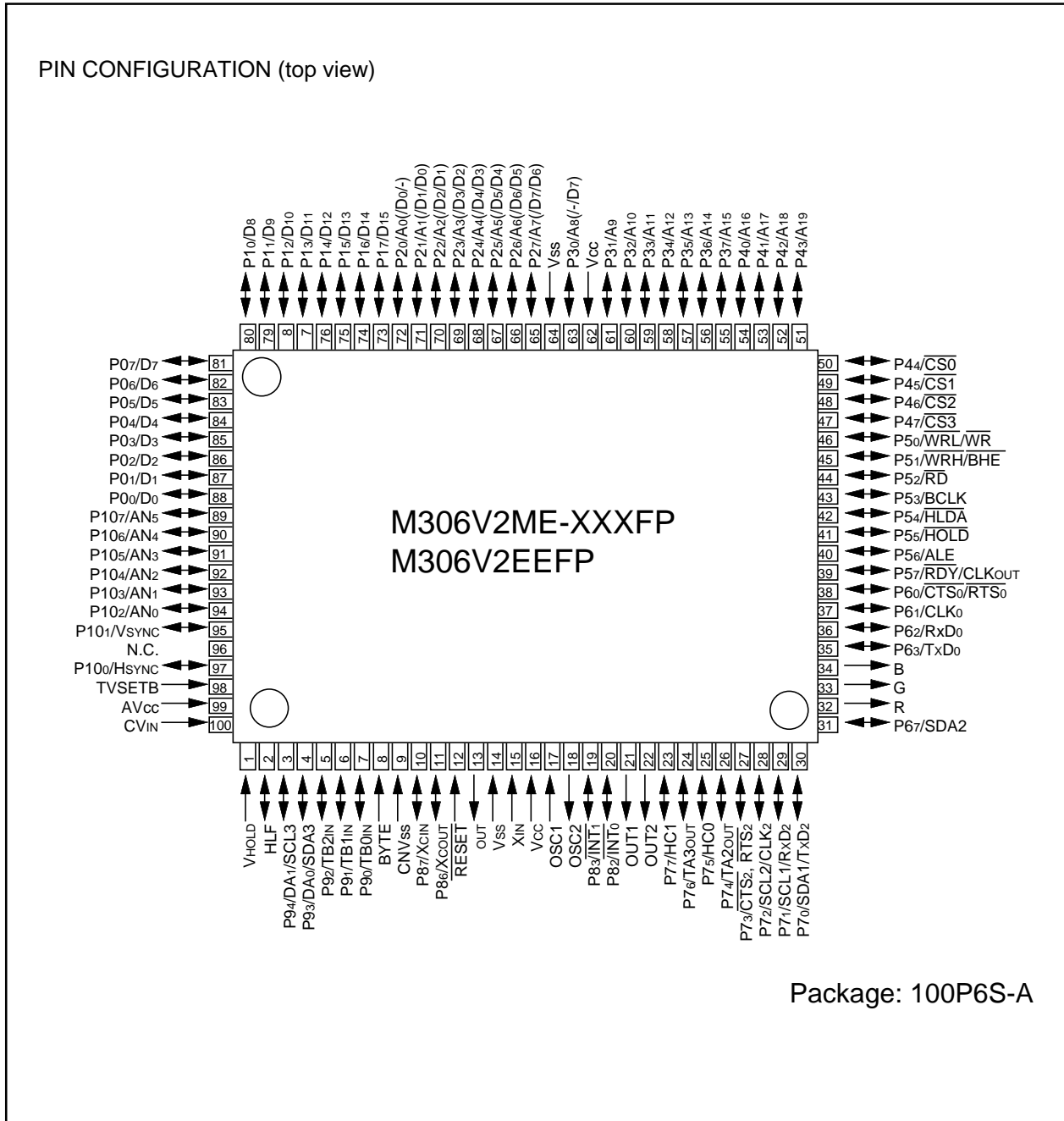


Figure 1.3.1 Pin configuration (top view)

1.4 Block Diagram

Figure 1.4.1 is a block diagram.

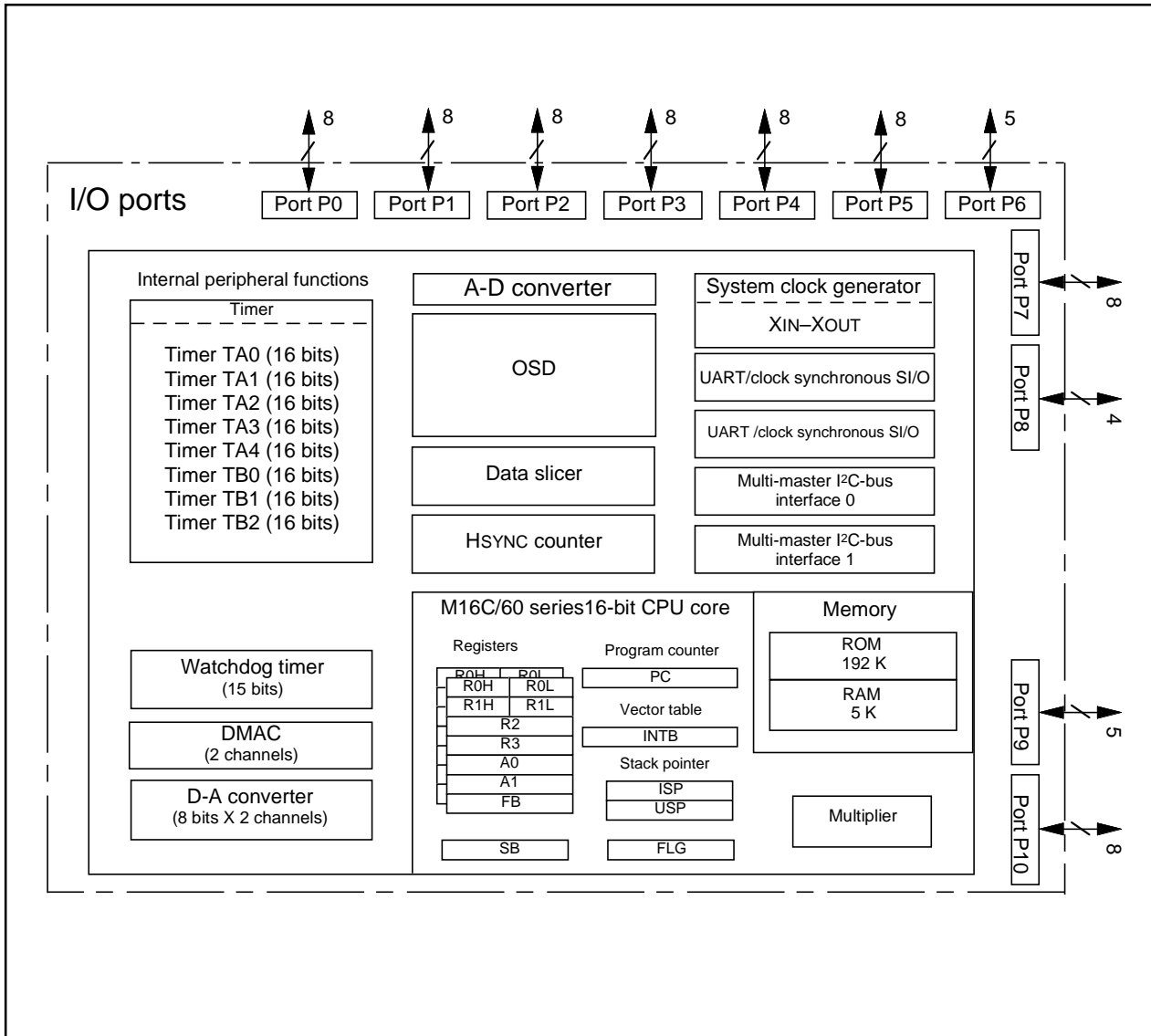


Figure 1.4.1 Block diagram

1.5 Performance Outline

Table 1.5.1 is a performance outline.

Table 1.5.1 Performance outline

| Item | | Performance |
|-------------------------------------|---|--|
| Number of basic instructions | | 91 instructions |
| Shortest instruction execution time | | 100 ns($f(X_{IN})=10$ MHz) |
| Memory size | ROM | 192K bytes |
| | RAM | 5K bytes |
| | OSD ROM | 61K bytes |
| | OSD RAM | 2.2K bytes |
| I/O port | P0 to P10 | 8 bits × 8, 5 bits × 2, 4 bits × 1 |
| Multifunction timer | TA0, TA1, TA2, TA3, TA4 | 16 bits × 5 |
| | TB0, TB1, TB2 | 16 bits × 3 |
| Serial I/O | UART0 | 1 unit: UART or clock synchronous |
| | UART2 | 1 unit: UART or clock synchronous |
| | Multi-master I ² C-BUS interface 0 | 1 unit (2 channels) |
| | Multi-master I ² C-BUS interface 1 | 1 unit (1 channel) |
| A-D converter | | 8 bits × 6 channels |
| D-A converter | | 8 bits × 2 channels |
| DMAC | | 2 channels (trigger: 23 sources) |
| OSD function | | Triple layer, 890 kinds of fonts, 42 character × 16 lines |
| Data slicer | | 32-bit buffer |
| HSYNC counter | | 8 bits × 2 channels |
| Watchdog timer | | 15 bits × 1 (with prescaler) |
| Interrupt | | 21 internal and 3 external sources, 4 software sources, 7 levels |
| Clock generating circuit | | 3 built-in clock generation circuits |
| Power source voltage | | 4.5 V to 5.5V ($f(X_{IN}) = 10$ MHz) |
| Power consumption | | 250 mW |
| I/O characteristics | I/O withstand voltage | 5 V |
| | Output current | 5 mA |
| Memory expansion | | Available |
| Operating ambient temperature | | -10 °C to 70 °C |
| Device configuration | | CMOS high performance silicon gate |
| Package | | 100-pin plastic molded QFP |

Currently supported products are listed below.

Table 1.5.2 List of supported products

| Type No | ROM capacity | RAM capacity | Package type | Remarks |
|----------------|--------------|--------------|--------------|-----------------------|
| M306V2ME-XXXFP | 192K bytes | 5K bytes | 100P6S-A | Mask ROM version |
| M306V2EEFP | 192K bytes | 5K bytes | 100P6S-A | One Time PROM version |
| M306V2EEFS | 192K bytes | 5K bytes | 100D0 | EPROM version |

Note: Since EPROM version is for development support tool (for evaluation), do not use for mass production.

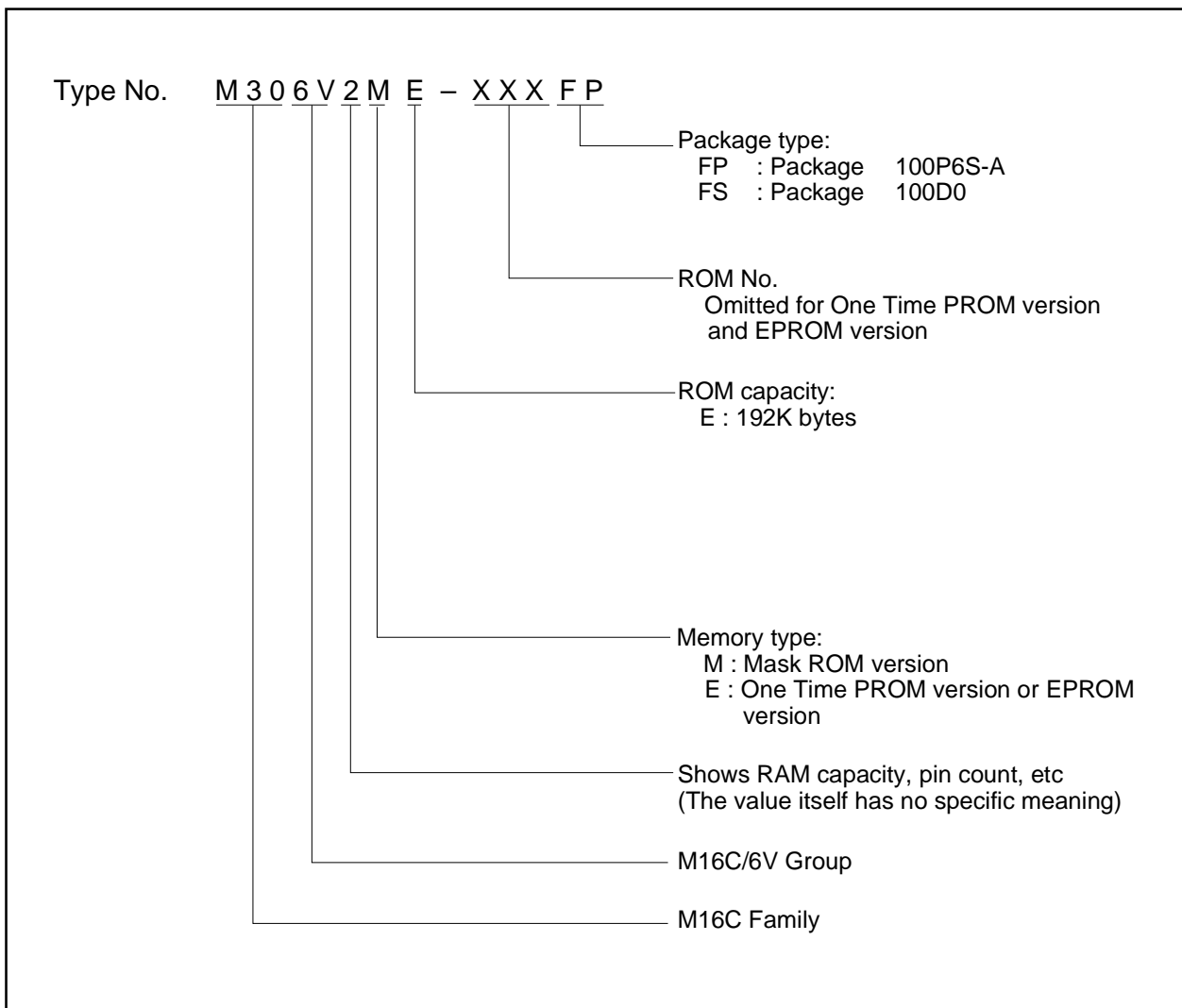


Figure 1.5.1 Type No., memory size, and package

Table 1.5.3 Pin description (1)

| Pin name | Signal name | I/O type | Function |
|--|--------------------------------------|------------------------|---|
| VCC, VSS | Power supply input | | Supply 4.5 V to 5.5 V to the VCC pin. Supply 0 V to the VSS pin. |
| CNVSS | CNVSS | Input | This pin switches between processor modes. Connect it to the VSS pin when operating in single-chip or memory expansion mode. Connect it to the VCC pin when in microprocessor mode. |
| $\overline{\text{RESET}}$ | Reset input | Input | A "L" on this input resets the microcomputer. |
| XIN XOUT | Clock input Clock output | Input Output | These pins are provided for the main clock generating circuit. Connect a ceramic resonator or crystal between the XIN and the XOUT pins. To use an externally derived clock, input it to the XIN pin and leave the XOUT pin open. |
| BYTE | External data bus width select input | Input | This pin selects the width of an external data bus. A 16-bit width is selected when this input is "L"; an 8-bit width is selected when this input is "H". This input must be fixed to either "H" or "L." When operating in single-chip mode, connect this pin to VSS. |
| AVCC | Analog power supply input | | This pin is a power supply input for the A-D converter. Connect this pin to VCC. |
| 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 in single-chip mode, the user can specify in units of four bits via software whether or not they are tied to a pull-up resistor. In memory expansion and microprocessor modes, the user cannot specify that. |
| D0 to D7 | | Input/output | When set as a separate bus, these pins input and output data (D0–D7). |
| P1 ₀ to P1 ₇ | I/O port P1 | Input/output | This is an 8-bit I/O port equivalent to P0. |
| D8 to D15 | | Input/output | When set as a separate bus, these pins input and output data (D8–D15). |
| P2 ₀ to P2 ₇ | I/O port P2 | Input/output | This is an 8-bit I/O port equivalent to P0. |
| A0 to A7 | | Output | These pins output 8 low-order address bits (A0–A7). |
| A0/D0 to A7/D7 | | Input/output | If the external bus is set as an 8-bit wide multiplexed bus, these pins input and output data (D0–D7) and output 8 low-order address bits (A0–A7) separated in time by multiplexing. |
| A0 A1/D0 to A7/D6 | | Output Input/output | If the external bus is set as a 16-bit wide multiplexed bus, these pins input and output data (D0–D6) and output address (A1–A7) separated in time by multiplexing. They also output address (A0). |
| P3 ₀ to P3 ₇ | I/O port P3 | Input/output | This is an 8-bit I/O port equivalent to P0. |
| A8 to A15 | | Output | These pins output 8 middle-order address bits (A8–A15). |
| A8/D7, A9 to A15 | | Input/output Output | If the external bus is set as a 16-bit wide multiplexed bus, these pins input and output data (D7) and output address (A8) separated in time by multiplexing. They also output address (A9–A15). |
| P4 ₀ to P4 ₇ | I/O port P4 | Input/output | This is an 8-bit I/O port equivalent to P0. |
| $\overline{\text{CS}}_0$ to $\overline{\text{CS}}_3$, A16 to A19 | | Output Output | These pins output $\overline{\text{CS}}_0$ – $\overline{\text{CS}}_3$ signals and A16–A19. $\overline{\text{CS}}_0$ – $\overline{\text{CS}}_3$ are chip select signals used to specify an access space. A16–A19 are 4 high-order address bits. |

Table 1.5.4 Pin description (continued) (2)

| Pin name | Signal name | I/O type | Function |
|--|----------------------|--|---|
| P50 to P57 | I/O port P5 | Input/output | This is an 8-bit I/O port equivalent to P0. In single-chip mode, P57 in this port outputs a divide-by-8 or divide-by-32 clock of XIN or a clock of the same frequency as XCIN as selected by software. |
| \overline{WRL} / \overline{WR} , \overline{WRH} / \overline{BHE} , \overline{RD} , BCLK, HLDA, HOLD, ALE, RDY | | Output Output Output Output Input Output Input | Output \overline{WRL} , \overline{WRH} (\overline{WR} and \overline{BHE}), \overline{RD} , BCLK, HLDA, and ALE signals. \overline{WRL} and \overline{WRH} , and \overline{BHE} and \overline{WR} can be switched using software control. <ul style="list-style-type: none"> ■ \overline{WRL}, \overline{WRH}, and \overline{RD} selected 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". ■ \overline{WR}, \overline{BHE}, and \overline{RD} selected 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. While the input level at the HOLD pin is "L", the microcomputer is placed in the hold state. While in the hold state, HLDA outputs a "L" level. ALE is used to latch the address. While the input level of the RDY pin is "L", the microcomputer is in the ready state. |
| P60 to P63, P67 | I/O port P6 | Input/output | This is an 5-bit I/O port equivalent to P0. When set for input in single-chip, microprocessor and memory expansion modes, the user can specify in units of four bits via software whether or not they are tied to a pull-up resistor. Pins in this port also function as UART0, UART2 and multi-master I ² C-BUS interface 0 I/O pins as selected by software. |
| P70 to P77 | I/O port P7 | Input/output | This is an 8-bit I/O port equivalent to P6 (P70 and P71 are N-channel open-drain output). Pins in this port also function as timers A2 and A3, UART2, multi-master I ² C-BUS interface 0, or HSYNC counter I/O pins as selected by software. |
| P82, P83, P86, P87 | I/O port P8 | Input/output | P82, P83, P86 and P87 are I/O ports with the same functions as P6. Using software, P82 and P83 can be made to function as the I/O pins for the input pins for external interrupts. P86 and P87 can be set using software to function as the I/O pins for a sub-clock generation circuit. In this case, connect a quartz oscillator between P86 (XCOUT pin) and P87 (XCIN pin). |
| P90 to P94 | I/O port P9 | Input/output | This is an 5-bit I/O port equivalent to P6. Pins in this port also function as Timer B0 to B2 input pins, D-A converter output pins, or multi-master I ² C-BUS interface 1 I/O pins. |
| P100 to P107 | I/O port P10 | Input/output | This is an 8-bit I/O port equivalent to P6. Pins in this port also function as A-D converter input pins. Furthermore, P100 and P101 also function as input pins for OSD function. |
| R, G, B | OSD output | Output | These are OSD output pins (analog output). |
| OUT1, OUT2 | OSD output | Output | These are OSD output pins (digital output). |
| OSC1 | Clock input for OSD | Input | This is an OSD clock input pin. |
| OSC2 | Clock output for OSD | Output | This is an OSD clock output pin. |
| CVIN | I/O for data slicer | Input | Input composite video signal through a capacitor. |
| VHOLD | | Input | Connect a capacitor between VHOLD and Vss. |
| HLF | | Input/output | Connect a filter using of a capacitor and a resistor between HLF and Vss. |
| TVSETB | Test input | Input | This is a test input pin. Fix it to "L." |

2. OPERATION OF FUNCTIONAL BLOKS

This microcomputer 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, OSD circuit, data slicer, A-D converter, and I/O ports.

The following explains each unit.

2.1 Memory

Figure 2.1.1 is a memory map. The address space extends the 1M bytes from address 00000_{16} to $FFFFFF_{16}$. From $FFFFFF_{16}$ down is ROM. There is 192K bytes of internal ROM from $D0000_{16}$ to $FFFFFF_{16}$. The vector table for fixed interrupts such as the reset mapped to $FFFDC_{16}$ to $FFFFFF_{16}$. The starting address of the interrupt routine is stored here. The address of the vector table for timer interrupts, etc., can be set as desired using the internal register (INTB). See the section on interrupts for details.

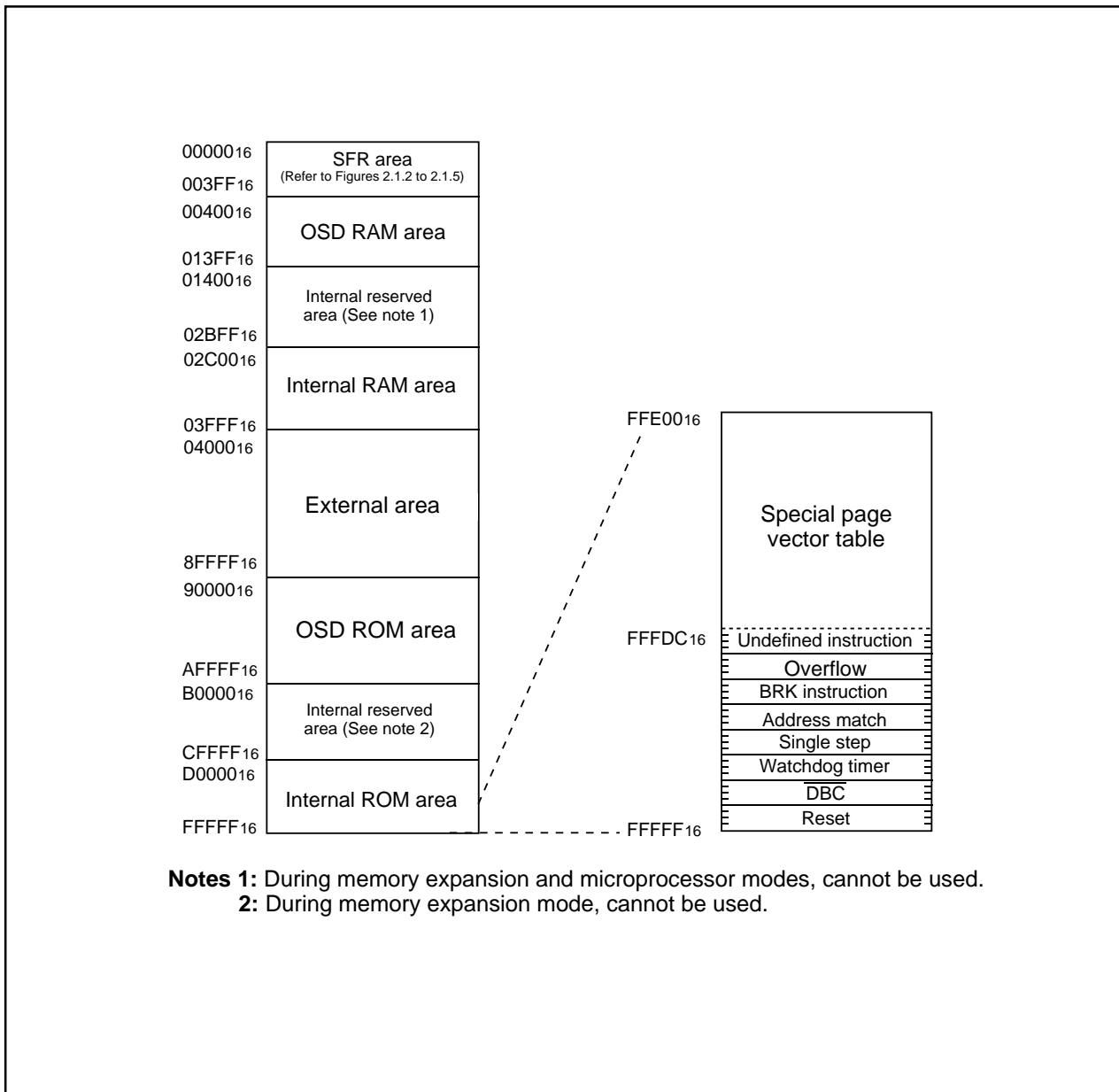
5K bytes of internal RAM is mapped to the space from $02C00_{16}$ to $03FFF_{16}$. In addition to storing data, the RAM also stores the stack used when calling subroutines and when interrupts are generated.

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 2.1.2 to 2.1.5 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. The following spaces cannot be used.

- The space between 01000_{16} and $02BFF_{16}$ (in memory expansion and microprocessor modes)
- The space between $B0000_{16}$ and $CFFFF_{16}$ (in memory expansion mode)



Notes 1: During memory expansion and microprocessor modes, cannot be used.
2: During memory expansion mode, cannot be used.

Figure 2.1.1 Memory map

| | | | |
|--------------------|--|--------------------|---|
| 0000 ₁₆ | | 0040 ₁₆ | |
| 0001 ₁₆ | | 0041 ₁₆ | |
| 0002 ₁₆ | | 0042 ₁₆ | |
| 0003 ₁₆ | | 0043 ₁₆ | |
| 0004 ₁₆ | Processor mode register 0 (PM0) | 0044 ₁₆ | OSD1 interrupt control register (OSD1IC) |
| 0005 ₁₆ | Processor mode register 1 (PM1) | 0045 ₁₆ | Interrupt control reserved register 0 (RE0IC) |
| 0006 ₁₆ | System clock control register 0 (CM0) | 0046 ₁₆ | Interrupt control reserved register 1 (RE1IC) |
| 0007 ₁₆ | System clock control register 1 (CM1) | 0047 ₁₆ | Interrupt control reserved register 2 (RE2IC) |
| 0008 ₁₆ | Chip select control register (CSR) | 0048 ₁₆ | OSD2 interrupt control register (OSD2IC) |
| 0009 ₁₆ | Address match interrupt enable register (AIER) | 0049 ₁₆ | Multi-master I ² C-BUS interface 1 interrupt control register (IIC01C) |
| 000A ₁₆ | Protect register (PRCR) | 004A ₁₆ | Bus collision detection interrupt control register (BCNIC) |
| 000B ₁₆ | | 004B ₁₆ | DMA0 interrupt control register (DM0IC) |
| 000C ₁₆ | | 004C ₁₆ | DMA1 interrupt control register (DM1IC) |
| 000D ₁₆ | | 004D ₁₆ | Multi-master I ² C-BUS interface 0 interrupt control register (IIC00C) |
| 000E ₁₆ | Watchdog timer start register (WDTS) | 004E ₁₆ | A-D conversion interrupt control register (ADIC) |
| 000F ₁₆ | Watchdog timer control register (WDC) | 004F ₁₆ | UART2 transmit interrupt control register (S2TIC) |
| 0010 ₁₆ | | 0050 ₁₆ | UART2 receive interrupt control register (S2RIC) |
| 0011 ₁₆ | Address match interrupt register 0 (RMAD0) | 0051 ₁₆ | UART0 transmit interrupt control register (S0TIC) |
| 0012 ₁₆ | | 0052 ₁₆ | UART0 receive interrupt control register (S0RIC) |
| 0013 ₁₆ | | 0053 ₁₆ | Data slicer interrupt control register (DSIC) |
| 0014 ₁₆ | | 0054 ₁₆ | VSYNC interrupt control register (VSYNCIC) |
| 0015 ₁₆ | Address match interrupt register 1 (RMAD1) | 0055 ₁₆ | Timer A0 interrupt control register (TA0IC) |
| 0016 ₁₆ | | 0056 ₁₆ | Timer A1 interrupt control register (TA1IC) |
| 0017 ₁₆ | | 0057 ₁₆ | Timer A2 interrupt control register (TA2IC) |
| 0018 ₁₆ | | 0058 ₁₆ | Timer A3 interrupt control register (TA3IC) |
| 0019 ₁₆ | | 0059 ₁₆ | Timer A4 interrupt control register (TA4IC) |
| 001A ₁₆ | | 005A ₁₆ | Timer B0 interrupt control register (TB0IC) |
| 001B ₁₆ | | 005B ₁₆ | Timer B1 interrupt control register (TB1IC) |
| 001C ₁₆ | | 005C ₁₆ | Timer B2 interrupt control register (TB2IC) |
| 001D ₁₆ | | 005D ₁₆ | INT0 interrupt control register (INT0IC) |
| 001E ₁₆ | | 005E ₁₆ | INT1 interrupt control register (INT1IC) |
| 001F ₁₆ | | 005F ₁₆ | Interrupt control reserved register 3 (RE3IC) |
| 0020 ₁₆ | | 0060 ₁₆ | |
| 0021 ₁₆ | DMA0 source pointer (SAR0) | | |
| 0022 ₁₆ | | | |
| 0023 ₁₆ | | | |
| 0024 ₁₆ | | | |
| 0025 ₁₆ | DMA0 destination pointer (DAR0) | | |
| 0026 ₁₆ | | | |
| 0027 ₁₆ | | | |
| 0028 ₁₆ | | | |
| 0029 ₁₆ | DMA0 transfer counter (TCR0) | | |
| 002A ₁₆ | | | |
| 002B ₁₆ | | | |
| 002C ₁₆ | DMA0 control register (DM0CON) | | |
| 002D ₁₆ | | | |
| 002E ₁₆ | | | |
| 002F ₁₆ | | | |
| 0030 ₁₆ | | | |
| 0031 ₁₆ | DMA1 source pointer (SAR1) | | |
| 0032 ₁₆ | | | |
| 0033 ₁₆ | | | |
| 0034 ₁₆ | | | |
| 0035 ₁₆ | DMA1 destination pointer (DAR1) | | |
| 0036 ₁₆ | | | |
| 0037 ₁₆ | | | |
| 0038 ₁₆ | DMA1 transfer counter (TCR1) | | |
| 0039 ₁₆ | | | |
| 003A ₁₆ | | | |
| 003B ₁₆ | | | |
| 003C ₁₆ | DMA1 control register (DM1CON) | | |
| 003D ₁₆ | | | |
| 003E ₁₆ | | | |
| 003F ₁₆ | | | |
| | | 01FF ₁₆ | |

Figure 2.1.2 Location of peripheral unit control registers (1)

| | | | |
|--------------------|--------------------------------------|--------------------|---|
| 0200 ₁₆ | | 0240 ₁₆ | Color palette register 1 (CR1) |
| 0201 ₁₆ | SPRITE OSD control register (SC) | 0241 ₁₆ | |
| 0202 ₁₆ | OSD control register 1 (OC1) | 0242 ₁₆ | Color palette register 2 (CR2) |
| 0203 ₁₆ | OSD control register 2 (OC2) | 0243 ₁₆ | |
| 0204 ₁₆ | Horizontal position register (HP) | 0244 ₁₆ | Color palette register 3 (CR3) |
| 0205 ₁₆ | Clock control register (CS) | 0245 ₁₆ | |
| 0206 ₁₆ | I/O polarity control register (PC) | 0246 ₁₆ | Color palette register 4 (CR4) |
| 0207 ₁₆ | OSD control register 3 (OC3) | 0247 ₁₆ | |
| 0208 ₁₆ | | 0248 ₁₆ | Color palette register 5 (CR5) |
| 0209 ₁₆ | Raster color register (RSC) | 0249 ₁₆ | |
| 020A ₁₆ | | 024A ₁₆ | Color palette register 6 (CR6) |
| 020B ₁₆ | | 024B ₁₆ | |
| 020C ₁₆ | | 024C ₁₆ | Color palette register 7 (CR7) |
| 020D ₁₆ | Top border control register (TBR) | 024D ₁₆ | |
| 020E ₁₆ | | 024E ₁₆ | Color palette register 9 (CR9) |
| 020F ₁₆ | Bottom border control register (BBR) | 024F ₁₆ | |
| 0210 ₁₆ | Block control register 1 (BC1) | 0250 ₁₆ | Color palette register 10 (CR10) |
| 0211 ₁₆ | Block control register 2 (BC2) | 0251 ₁₆ | |
| 0212 ₁₆ | Block control register 3 (BC3) | 0252 ₁₆ | Color palette register 11 (CR11) |
| 0213 ₁₆ | Block control register 4 (BC4) | 0253 ₁₆ | |
| 0214 ₁₆ | Block control register 5 (BC5) | 0254 ₁₆ | Color palette register 12 (CR12) |
| 0215 ₁₆ | Block control register 6 (BC6) | 0255 ₁₆ | |
| 0216 ₁₆ | Block control register 7 (BC7) | 0256 ₁₆ | Color palette register 13 (CR13) |
| 0217 ₁₆ | Block control register 8 (BC8) | 0257 ₁₆ | |
| 0218 ₁₆ | Block control register 9 (BC9) | 0258 ₁₆ | Color palette register 14 (CR14) |
| 0219 ₁₆ | Block control register 10 (BC10) | 0259 ₁₆ | |
| 021A ₁₆ | Block control register 11(BC11) | 025A ₁₆ | Color palette register 15 (CR15) |
| 021B ₁₆ | Block control register 12 (BC12) | 025B ₁₆ | |
| 021C ₁₆ | Block control register 13 (BC13) | 025C ₁₆ | |
| 021D ₁₆ | Block control register 14 (BC14) | 025D ₁₆ | OSD reserved register 1 (OR1) |
| 021E ₁₆ | Block control register 15 (BC15) | 025E ₁₆ | |
| 021F ₁₆ | Block control register 16 (BC16) | 025F ₁₆ | OSD control register 4 (OC4) |
| 0220 ₁₆ | Vertical position register 1 (VP1) | 0260 ₁₆ | Data slicer control register 1 (DSC1) |
| 0221 ₁₆ | | 0261 ₁₆ | Data slicer control register 2 (DSC2) |
| 0222 ₁₆ | Vertical position register 2 (VP2) | 0262 ₁₆ | Caption data register 1 (CD1) |
| 0223 ₁₆ | | 0263 ₁₆ | |
| 0224 ₁₆ | Vertical position register 3 (VP3) | 0264 ₁₆ | Caption data register 2 (CD2) |
| 0225 ₁₆ | | 0265 ₁₆ | |
| 0226 ₁₆ | Vertical position register 4 (VP4) | 0266 ₁₆ | Caption position register (CPS) |
| 0227 ₁₆ | | 0267 ₁₆ | Data slicer reserved register 2 (DR2) |
| 0228 ₁₆ | Vertical position register 5 (VP5) | 0268 ₁₆ | Data slicer reserved register 1 (DR1) |
| 0229 ₁₆ | | 0269 ₁₆ | Clock run-in detect register (CRD) |
| 022A ₁₆ | Vertical position register 6 (VP6) | 026A ₁₆ | Data clock position register (DPS) |
| 022B ₁₆ | | 026B ₁₆ | |
| 022C ₁₆ | Vertical position register 7 (VP7) | 026F ₁₆ | ≈≈ |
| 022D ₁₆ | | 0270 ₁₆ | Left border control register (LBR) |
| 022E ₁₆ | Vertical position register 8 (VP8) | 0271 ₁₆ | |
| 022F ₁₆ | | 0272 ₁₆ | Right border control register (RBR) |
| 0230 ₁₆ | Vertical position register 9 (VP9) | 0273 ₁₆ | |
| 0231 ₁₆ | | 0274 ₁₆ | SPRITE vertical position register 1 (VS1) |
| 0232 ₁₆ | Vertical position register 10 (VP10) | 0275 ₁₆ | |
| 0233 ₁₆ | | 0276 ₁₆ | SPRITE vertical position register 2 (VS2) |
| 0234 ₁₆ | Vertical position register 11 (VP11) | 0277 ₁₆ | |
| 0235 ₁₆ | | 0278 ₁₆ | SPRITE horizontal position register (HS) |
| 0236 ₁₆ | Vertical position register 12 (VP12) | 0279 ₁₆ | |
| 0237 ₁₆ | | 027A ₁₆ | OSD reserved register 4 (OR4) |
| 0238 ₁₆ | Vertical position register 13 (VP13) | 027B ₁₆ | OSD reserved register 3 (OR3) |
| 0239 ₁₆ | | 027C ₁₆ | OSD reserved register 2 (OR2) |
| 023A ₁₆ | Vertical position register 14 (VP14) | 027D ₁₆ | Peripheral mode register (PM) |
| 023B ₁₆ | | 027E ₁₆ | HSYNC counter register (HC) |
| 023C ₁₆ | Vertical position register 15 (VP15) | 027F ₁₆ | HSYNC counter latch |
| 023D ₁₆ | | 0280 ₁₆ | ≈≈ |
| 023E ₁₆ | Vertical position register 16 (VP16) | 02DF ₁₆ | ≈≈ |
| 023F ₁₆ | | | |

Figure 2.1.3 Location of peripheral unit control registers (2)

| | | | |
|--------------------|--|--------------------|--|
| 02E0 ₁₆ | I ² C0 data shift register (IIC0S0) | 0380 ₁₆ | Count start flag (TABSR) |
| 02E1 ₁₆ | I ² C0 address register (IIC0S0D) | 0381 ₁₆ | Clock prescaler reset flag (CPSRF) |
| 02E2 ₁₆ | I ² C0 status register (IIC0S1) | 0382 ₁₆ | One-shot start flag (ONSF) |
| 02E3 ₁₆ | I ² C0 control register (IIC0S1D) | 0383 ₁₆ | Trigger select register (TRGSR) |
| 02E4 ₁₆ | I ² C0 clock control register (IIC0S2) | 0384 ₁₆ | Up-down flag (UDF) |
| 02E5 ₁₆ | I ² C0 port selection register (IIC0S2D) | 0385 ₁₆ | |
| 02E6 ₁₆ | I ² C0 transmit buffer register (IIC0S0S) | 0386 ₁₆ | |
| 02E7 ₁₆ | | 0387 ₁₆ | Timer A0 register (TA0) |
| 02E8 ₁₆ | I ² C1 data shift register (IIC1S0) | 0388 ₁₆ | |
| 02E9 ₁₆ | I ² C1 address register (IIC1S0D) | 0389 ₁₆ | Timer A1 register (TA1) |
| 02EA ₁₆ | I ² C1 status register (IIC1S1) | 038A ₁₆ | |
| 02EB ₁₆ | I ² C1 control register (IIC1S1D) | 038B ₁₆ | Timer A2 register (TA2) |
| 02EC ₁₆ | I ² C1 clock control register (IIC1S2) | 038C ₁₆ | |
| 02ED ₁₆ | I ² C1 port selection register (IIC1S2D) | 038D ₁₆ | Timer A3 register (TA3) |
| 02EE ₁₆ | I ² C1 transmit buffer register (IIC1S0S) | 038E ₁₆ | |
| 02EF ₁₆ | | 038F ₁₆ | Timer A4 register (TA4) |
| | ≈ | 0390 ₁₆ | |
| | | 0391 ₁₆ | Timer B0 register (TB0) |
| 0339 ₁₆ | | 0392 ₁₆ | |
| 0340 ₁₆ | Reserved register 1 (INVC1) | 0393 ₁₆ | Timer B1 register (TB1) |
| 0341 ₁₆ | | 0394 ₁₆ | |
| 0342 ₁₆ | | 0395 ₁₆ | Timer B2 register (TB2) |
| 0343 ₁₆ | | 0396 ₁₆ | Timer A0 mode register (TA0MR) |
| 0344 ₁₆ | | 0397 ₁₆ | Timer A1 mode register (TA1MR) |
| 0345 ₁₆ | | 0398 ₁₆ | Timer A2 mode register (TA2MR) |
| 0346 ₁₆ | | 0399 ₁₆ | Timer A3 mode register (TA3MR) |
| 0347 ₁₆ | | 039A ₁₆ | Timer A4 mode register (TA4MR) |
| 0348 ₁₆ | Reserved register 0 (INVC0) | 039B ₁₆ | Timer B0 mode register (TB0MR) |
| 0349 ₁₆ | | 039C ₁₆ | Timer B1 mode register (TB1MR) |
| | ≈ | 039D ₁₆ | Timer B2 mode register (TB2MR) |
| | | 039E ₁₆ | |
| 035E ₁₆ | | 039F ₁₆ | |
| 035F ₁₆ | Interrupt request cause select register (IFSR) | 03A0 ₁₆ | UART0 transmit/receive mode register (U0MR) |
| 0360 ₁₆ | | 03A1 ₁₆ | UART0 bit rate generator (U0BRG) |
| 0361 ₁₆ | | 03A2 ₁₆ | |
| 0362 ₁₆ | Reserved register 3 (INVC3) | 03A3 ₁₆ | UART0 transmit buffer register (U0TB) |
| 0363 ₁₆ | | 03A4 ₁₆ | UART0 transmit/receive control register 0 (U0C0) |
| 0364 ₁₆ | | 03A5 ₁₆ | UART0 transmit/receive control register 1 (U0C1) |
| 0365 ₁₆ | | 03A6 ₁₆ | |
| 0366 ₁₆ | Reserved register 4 (INVC4) | 03A7 ₁₆ | UART0 receive buffer register (U0RB) |
| 0367 ₁₆ | | 03A8 ₁₆ | Reserved register 2 (INVC2) |
| 0368 ₁₆ | | 03A9 ₁₆ | |
| 0369 ₁₆ | | 03AA ₁₆ | |
| 036A ₁₆ | | 03AB ₁₆ | |
| 036B ₁₆ | | 03AC ₁₆ | |
| 036C ₁₆ | | 03AD ₁₆ | |
| 036D ₁₆ | | 03AE ₁₆ | |
| 036E ₁₆ | | 03AF ₁₆ | |
| 036F ₁₆ | | 03B0 ₁₆ | UART transmit/receive control register 2 (U0CON) |
| 0370 ₁₆ | | 03B1 ₁₆ | |
| 0371 ₁₆ | | 03B2 ₁₆ | |
| 0372 ₁₆ | | 03B3 ₁₆ | |
| 0373 ₁₆ | | 03B4 ₁₆ | |
| 0374 ₁₆ | | 03B5 ₁₆ | |
| 0375 ₁₆ | | 03B6 ₁₆ | |
| 0376 ₁₆ | Reserved register 5 (INVC5) | 03B7 ₁₆ | |
| 0377 ₁₆ | UART2 special mode register (U2SMR) | 03B8 ₁₆ | DMA0 request cause select register (DM0SL) |
| 0378 ₁₆ | UART2 transmit/receive mode register (U2MR) | 03B9 ₁₆ | |
| 0379 ₁₆ | UART2 bit rate generator (U2BRG) | 03BA ₁₆ | DMA1 request cause select register (DM1SL) |
| 037A ₁₆ | | 03BB ₁₆ | |
| 037B ₁₆ | UART2 transmit buffer register (U2TB) | 03BC ₁₆ | |
| 037C ₁₆ | UART2 transmit/receive control register 0 (U2C0) | 03BD ₁₆ | |
| 037D ₁₆ | UART2 transmit/receive control register 1 (U2C1) | 03BE ₁₆ | |
| 037E ₁₆ | | 03BF ₁₆ | |
| 037F ₁₆ | UART2 receive buffer register (U2RB) | | |

Figure 2.1.4 Location of peripheral unit control registers (3)

| | |
|--------------------|------------------------------------|
| 03C0 ¹⁶ | |
| 03C1 ¹⁶ | |
| 03C2 ¹⁶ | |
| 03C3 ¹⁶ | |
| 03C4 ¹⁶ | A-D register 0 (AD0) |
| 03C5 ¹⁶ | |
| 03C6 ¹⁶ | A-D register 1 (AD1) |
| 03C7 ¹⁶ | |
| 03C8 ¹⁶ | A-D register 2 (AD2) |
| 03C9 ¹⁶ | |
| 03CA ¹⁶ | A-D register 3 (AD3) |
| 03CB ¹⁶ | |
| 03CC ¹⁶ | A-D register 4 (AD4) |
| 03CD ¹⁶ | |
| 03CE ¹⁶ | A-D register 5 (AD5) |
| 03CF ¹⁶ | |
| 03D0 ¹⁶ | |
| 03D1 ¹⁶ | |
| 03D2 ¹⁶ | |
| 03D3 ¹⁶ | |
| 03D4 ¹⁶ | A-D control register 2 (ADCON2) |
| 03D5 ¹⁶ | |
| 03D6 ¹⁶ | A-D control register 0 (ADCON0) |
| 03D7 ¹⁶ | A-D control register 1 (ADCON1) |
| 03D8 ¹⁶ | D-A register 0 (DA0) |
| 03D9 ¹⁶ | |
| 03DA ¹⁶ | D-A register 1 (DA1) |
| 03DB ¹⁶ | |
| 03DC ¹⁶ | D-A control register (DACON) |
| 03DD ¹⁶ | |
| 03DE ¹⁶ | |
| 03DF ¹⁶ | |
| 03E0 ¹⁶ | Port P0 register (P0) |
| 03E1 ¹⁶ | Port P1 register (P1) |
| 03E2 ¹⁶ | Port P0 direction register (PD0) |
| 03E3 ¹⁶ | Port P1 direction register (PD1) |
| 03E4 ¹⁶ | Port P2 register (P2) |
| 03E5 ¹⁶ | Port P3 register (P3) |
| 03E6 ¹⁶ | Port P2 direction register (PD2) |
| 03E7 ¹⁶ | Port P3 direction register (PD3) |
| 03E8 ¹⁶ | Port P4 register (P4) |
| 03E9 ¹⁶ | Port P5 register (P5) |
| 03EA ¹⁶ | Port P4 direction register (PD4) |
| 03EB ¹⁶ | Port P5 direction register (PD5) |
| 03EC ¹⁶ | Port P6 register (P6) |
| 03ED ¹⁶ | Port P7 register (P7) |
| 03EE ¹⁶ | Port P6 direction register (PD6) |
| 03EF ¹⁶ | Port P7 direction register (PD7) |
| 03F0 ¹⁶ | Port P8 register (P8) |
| 03F1 ¹⁶ | Port P9 register (P9) |
| 03F2 ¹⁶ | Port P8 direction register (PD8) |
| 03F3 ¹⁶ | Port P9 direction register (PD9) |
| 03F4 ¹⁶ | Port P10 register (P10) |
| 03F5 ¹⁶ | |
| 03F6 ¹⁶ | Port P10 direction register (PD10) |
| 03F7 ¹⁶ | |
| 03F8 ¹⁶ | |
| 03F9 ¹⁶ | |
| 03FA ¹⁶ | |
| 03FB ¹⁶ | |
| 03FC ¹⁶ | Pull-up control register 0 (PUR0) |
| 03FD ¹⁶ | Pull-up control register 1 (PUR1) |
| 03FE ¹⁶ | Pull-up control register 2 (PUR2) |
| 03FF ¹⁶ | Port control register (PCR) |

Figure 2.1.5 Location of peripheral unit control registers (4)

2.2 Central Processing Unit (CPU)

The CPU has a total of 13 registers shown in Figure 2.2.1. Seven of these registers (R0, R1, R2, R3, A0, A1, and FB) come in two sets; therefore, these have two register banks.

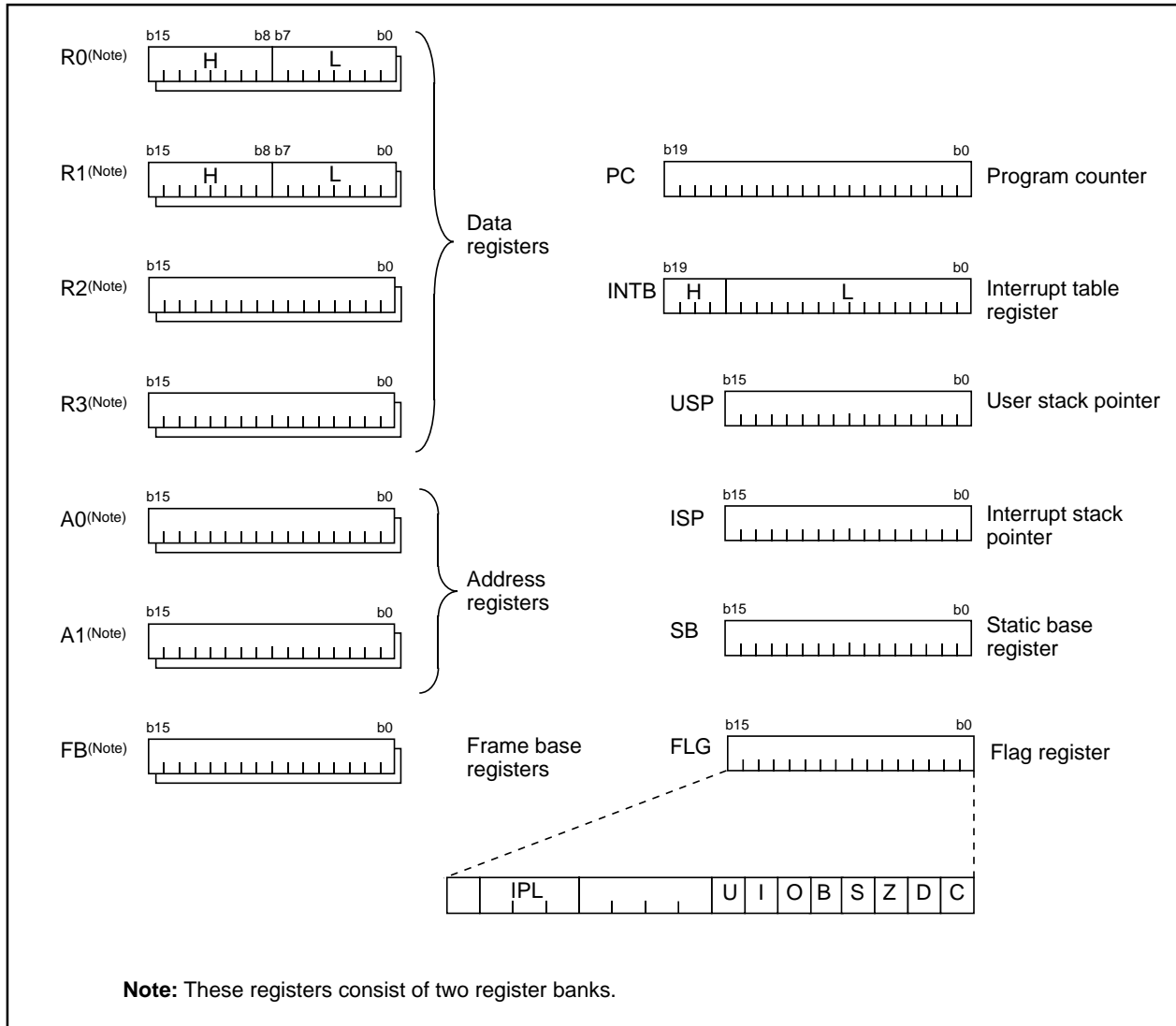


Figure 2.2.1 Central processing unit register

2.2.1 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).

2.2.2 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).

2.2.3 Frame Base Register (FB)

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

2.2.4 Program Counter (PC)

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

2.2.5 Interrupt Table Register (INTB)

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

2.2.6 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).

2.2.7 Static Base Register (SB)

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

2.2.8 Flag Register (FLG)

Flag register (FLG) is configured with 11 bits, each bit is used as a flag. Figure 2.2.2 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.

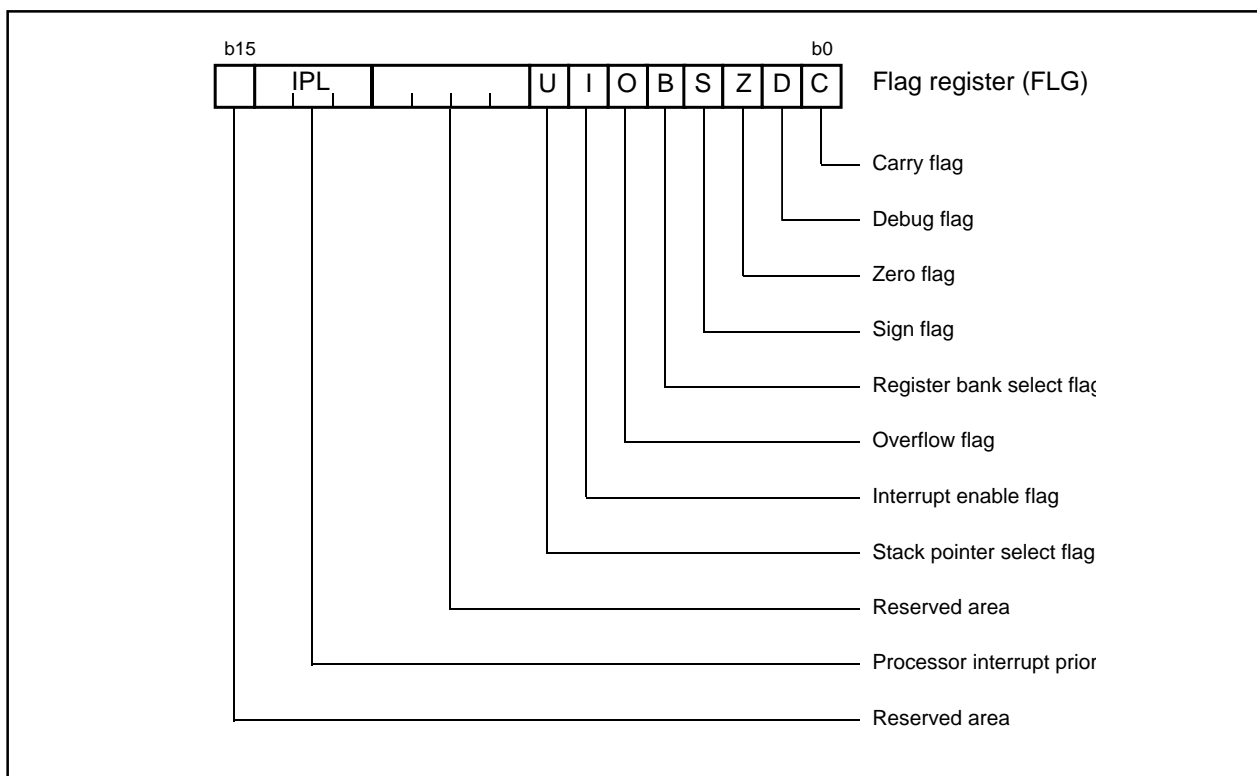


Figure 2.2.2 Flag register (FLG)

2.3 Reset

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

When the supply voltage is in the range where operation is guaranteed, a reset is effected by holding the reset pin level “L” (0.2V_{CC} 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 2.3.1 shows the example reset circuit. Figure 2.3.2 shows the reset sequence.

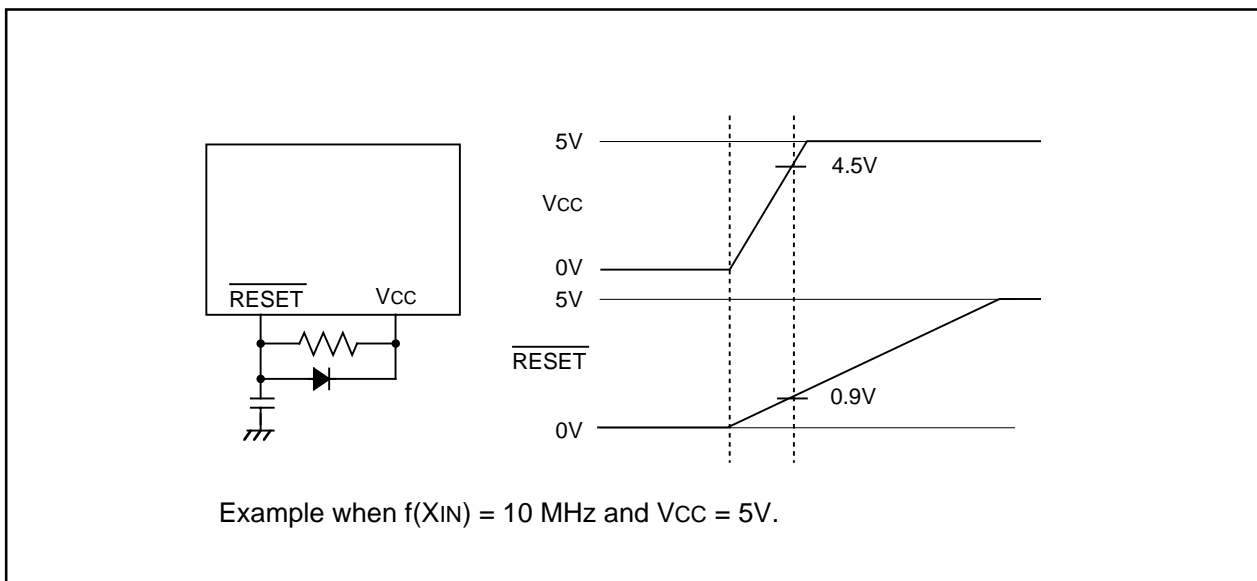


Figure 2.3.1 Example reset circuit

2.3.1 Software Reset

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

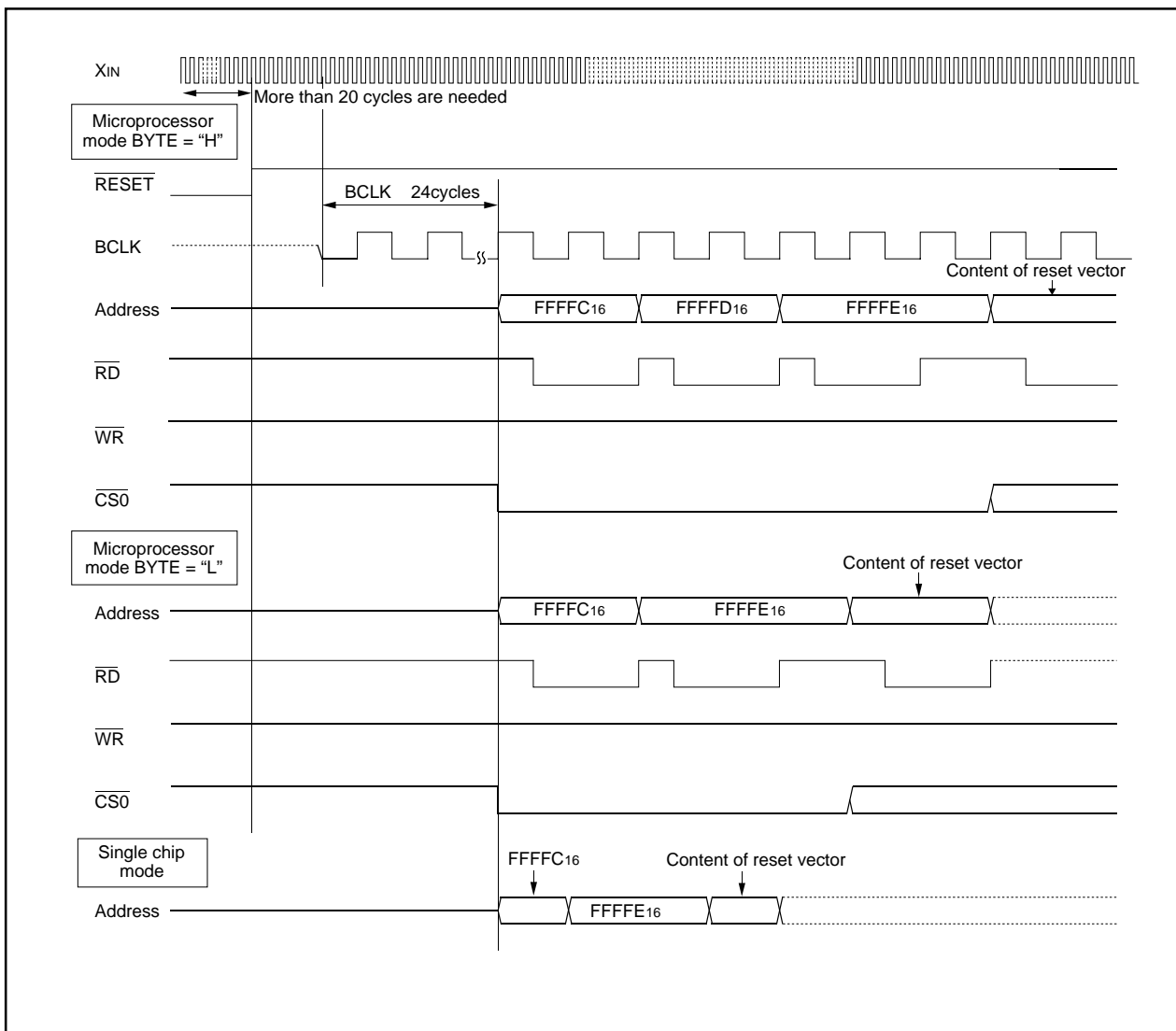


Figure 2.3.2 Reset sequence

2.3.2 Pin Status When $\overline{\text{RESET}}$ Pin Level is “L”

Table 2.3.1 shows the statuses of the other pins while the $\overline{\text{RESET}}$ pin level is “L”. Figures 2.3.3 and 2.3.4 show the internal status of the microcomputer immediately after the reset is cancelled.

Table 2.3.1 Pin status when $\overline{\text{RESET}}$ pin level is “L”

| Pin name | Status | | |
|--|-----------------------|---|---|
| | CNVss = Vss | CNVss = Vcc | |
| | | BYTE = Vss | BYTE = Vcc |
| P0 | Input port (floating) | Data input (floating) | Data input (floating) |
| P1 | Input port (floating) | Data input (floating) | Input port (floating) |
| P2, P3, P40 to P43 | Input port (floating) | Address output (undefined) | Address output (undefined) |
| P44 | Input port (floating) | $\overline{\text{CS0}}$ output (“H” level is output) | $\overline{\text{CS0}}$ output (“H” level is output) |
| P45 to P47 | Input port (floating) | Input port (floating) (pull-up resistor is on) | Input port (floating) (pull-up resistor is on) |
| P50 | Input port (floating) | $\overline{\text{WR}}$ output (“H” level is output) | $\overline{\text{WR}}$ output (“H” level is output) |
| P51 | Input port (floating) | $\overline{\text{BHE}}$ output (undefined) | $\overline{\text{BHE}}$ output (undefined) |
| P52 | Input port (floating) | $\overline{\text{RD}}$ output (“H” level is output) | $\overline{\text{RD}}$ output (“H” level is output) |
| P53 | Input port (floating) | BCLK output | BCLK output |
| P54 | Input port (floating) | $\overline{\text{HLDA}}$ output (The output value depends on the input to the HOLD pin) | $\overline{\text{HLDA}}$ output (The output value depends on the input to the HOLD pin) |
| P55 | Input port (floating) | $\overline{\text{HOLD}}$ input (floating) | $\overline{\text{HOLD}}$ input (floating) |
| P56 | Input port (floating) | ALE output (“L” level is output) | ALE output (“L” level is output) |
| P57 | Input port (floating) | $\overline{\text{RDY}}$ input (floating) | $\overline{\text{RDY}}$ input (floating) |
| P60 to P63, P67, P7, P82, P83, P86, P87, P9, P10 | Input port (floating) | Input port (floating) | Input port (floating) |
| R, G, B, OUT1, OUT2 | Output port | | |
| CVIN, VHOLD, HLF | Input/output port | | |
| OSC1 | Input port | | |
| OSC2 | Output port | | |

| | | | | | |
|--|--------------------------|------------------|--|--------------------------|------------------|
| Processor mode register 0 (Note) | (0004 ₁₆)... | 00 ₁₆ | Timer B0 interrupt control register | (005A ₁₆)... | XXXXX?000 |
| Processor mode register 1 | (0005 ₁₆)... | 00000X00 | Timer B1 interrupt control register | (005B ₁₆)... | XXXXX?000 |
| System clock control register 0 | (0006 ₁₆)... | 48 ₁₆ | Timer B2 interrupt control register | (005C ₁₆)... | XXXXX?000 |
| System clock control register 1 | (0007 ₁₆)... | 20 ₁₆ | INT0 interrupt control register | (005D ₁₆)... | XX00?000 |
| Chip select control register | (0008 ₁₆)... | 01 ₁₆ | INT1 interrupt control register | (005E ₁₆)... | XX00?000 |
| Address match interrupt enable register | (0009 ₁₆)... | XXXXXXXX00 | SPRITE OSD control register | (0201 ₁₆)... | XXXX000000 |
| Protect register | (000A ₁₆)... | XXXXXXXX000 | OSD control register 1 | (0202 ₁₆)... | 00 ₁₆ |
| Watchdog timer control register | (000F ₁₆)... | 0000???? | OSD control register 2 | (0203 ₁₆)... | 00 ₁₆ |
| Address match interrupt register 0 | (0010 ₁₆)... | 00 ₁₆ | Horizontal position register | (0204 ₁₆)... | 00 ₁₆ |
| | (0011 ₁₆)... | 00 ₁₆ | Clock control register | (0205 ₁₆)... | 00 ₁₆ |
| | (0012 ₁₆)... | XXXXXXXX0000 | I/O polarity control register | (0206 ₁₆)... | 1000000000 |
| Address match interrupt register 1 | (0014 ₁₆)... | 00 ₁₆ | OSD control register 3 | (0207 ₁₆)... | 00 ₁₆ |
| | (0015 ₁₆)... | 00 ₁₆ | Raster color register | (0208 ₁₆)... | 00 ₁₆ |
| | (0016 ₁₆)... | XXXXXXXX0000 | | (0209 ₁₆)... | 00 ₁₆ |
| DMA0 control register | (002C ₁₆)... | 000000?000 | OSD reserved register 1 | (025D ₁₆)... | 00 ₁₆ |
| DMA1 control register | (003C ₁₆)... | 000000?000 | OSD control register 4 | (025F ₁₆)... | XXXXXXXXX000 |
| OSD1 interrupt control register | (0044 ₁₆)... | XXXXX?0000 | Data slicer control register 1 | (0260 ₁₆)... | 00 ₁₆ |
| OSD2 interrupt control register | (0048 ₁₆)... | XXXXX?0000 | Data slicer control register 2 | (0261 ₁₆)... | ?0?0?0?0? |
| Multi-master I ² C-BUS interface 1 interrupt control register | (0049 ₁₆)... | XX000?0000 | Caption position register | (0266 ₁₆)... | 00?0000000 |
| Bus collision detection interrupt control register | (004A ₁₆)... | XXXXX?0000 | Data slicer reserved register 2 | (0267 ₁₆)... | 00 ₁₆ |
| DMA0 interrupt control register | (004B ₁₆)... | XXXXX?0000 | Data slicer reserved register 1 | (0268 ₁₆)... | 00 ₁₆ |
| DMA1 interrupt control register | (004C ₁₆)... | XXXXX?0000 | Clock run-in detect register | (0269 ₁₆)... | 00 ₁₆ |
| Multi-master I ² C-BUS interface 0 interrupt control register | (004D ₁₆)... | XXXXX?0000 | Data clock position register | (026A ₁₆)... | XXXX000001 |
| A-D conversion interrupt control register | (004E ₁₆)... | XXXXX?0000 | Left border control register | (0270 ₁₆)... | 01 ₁₆ |
| UART2 transmit interrupt control register | (004F ₁₆)... | XXXXX?0000 | | (0271 ₁₆)... | XXXXXXXX0000 |
| UART2 receive interrupt control register | (0050 ₁₆)... | XXXXX?0000 | Right border control register | (0272 ₁₆)... | 00 ₁₆ |
| UART0 transmit interrupt control register | (0051 ₁₆)... | XXXXX?0000 | | (0273 ₁₆)... | XXXXXXXX0000 |
| UART0 receive interrupt control register | (0052 ₁₆)... | XXXXX?0000 | SPRITE horizontal position register (high-order) | (0279 ₁₆)... | XXXXX00000 |
| Data slicer interrupt control register | (0053 ₁₆)... | XXXXX?0000 | OSD reserved register 4 | (027A ₁₆)... | X000000000 |
| Vsync interrupt control register | (0054 ₁₆)... | XXXXX?0000 | OSD reserved register 3 | (027B ₁₆)... | 00 ₁₆ |
| Timer A0 interrupt control register | (0055 ₁₆)... | XXXXX?0000 | OSD reserved register 2 | (027C ₁₆)... | 00 ₁₆ |
| Timer A1 interrupt control register | (0056 ₁₆)... | XXXXX?0000 | Peripheral mode register | (027D ₁₆)... | 0XXXX00000 |
| Timer A2 interrupt control register | (0057 ₁₆)... | XXXXX?0000 | Hsync counter register | (027E ₁₆)... | XXXX00XX00 |
| Timer A3 interrupt control register | (0058 ₁₆)... | XXXXX?0000 | | | |
| Timer A4 interrupt control register | (0059 ₁₆)... | XXXXX?0000 | | | |

X : Nothing is mapped to this bit
? : Undefined

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

Note: When the VCC level is applied to the CNVSS pin, it is 0316 at a reset.

Figure 2.3.3 Device's internal status after a reset is cleared (1)

| | | | | | |
|---|--------------------------|-------------------|--|--------------------------|--------------------|
| I ² C0 address register | (02E1 ₁₆)... | 00 ₁₆ | UART transmit/receive control register 2 | (03B0 ₁₆)... | ⊗ 0 0 0 0 0 0 0 0 |
| I ² C0 status register | (02E2 ₁₆)... | 0 0 0 1 0 0 0 ? | DMA0 request cause select register | (03B8 ₁₆)... | 00 ₁₆ |
| I ² C0 control register | (02E3 ₁₆)... | 00 ₁₆ | DMA1 request cause select register | (03BA ₁₆)... | 00 ₁₆ |
| I ² C0 clock control register | (02E4 ₁₆)... | 00 ₁₆ | A-D control register 2 | (03D4 ₁₆)... | 0 0 0 0 ? ? ? 0 |
| I ² C0 port selection register | (02E5 ₁₆)... | 0 0 ? ? 0 0 0 0 | A-D control register 0 | (03D6 ₁₆)... | 0 0 0 0 0 ? ? ? |
| I ² C1 address register | (02E9 ₁₆)... | 00 ₁₆ | A-D control register 1 | (03D7 ₁₆)... | 00 ₁₆ |
| I ² C1 status register | (02EA ₁₆)... | 0 0 0 1 0 0 0 ? | D-A control register | (03DC ₁₆)... | 00 ₁₆ |
| I ² C1 control register | (02EB ₁₆)... | 00 ₁₆ | Port P0 direction register | (03E2 ₁₆)... | 00 ₁₆ |
| I ² C1 clock control register | (02EC ₁₆)... | 00 ₁₆ | Port P1 direction register | (03E3 ₁₆)... | 00 ₁₆ |
| I ² C1 port selection register | (02ED ₁₆)... | 0 0 ? ? 0 0 0 0 | Port P2 direction register | (03E6 ₁₆)... | 00 ₁₆ |
| Reserved register 1 | (0340 ₁₆)... | 0 0 0 ? ? ? ? ? | Port P3 direction register | (03E7 ₁₆)... | 00 ₁₆ |
| Reserved register 0 | (0348 ₁₆)... | 00 ₁₆ | Port P4 direction register | (03EA ₁₆)... | 00 ₁₆ |
| Interrupt request cause select register | (035F ₁₆)... | 00 ₁₆ | Port P5 direction register | (03EB ₁₆)... | 00 ₁₆ |
| Reserved register 3 | (0362 ₁₆)... | 40 ₁₆ | Port P6 direction register | (03EE ₁₆)... | 00 ₁₆ |
| Reserved register 4 | (0366 ₁₆)... | 40 ₁₆ | Port P7 direction register | (03EF ₁₆)... | 00 ₁₆ |
| Reserved register 5 | (0376 ₁₆)... | 00 ₁₆ | Port P8 direction register | (03F2 ₁₆)... | 0 0 ⊗ 0 0 0 0 0 0 |
| UART2 special mode register | (0377 ₁₆)... | 00 ₁₆ | Port P9 direction register | (03F3 ₁₆)... | 00 ₁₆ |
| UART2 transmit/receive mode register | (0378 ₁₆)... | 00 ₁₆ | Port P10 direction register | (03F6 ₁₆)... | 00 ₁₆ |
| UART2 transmit/receive control register 0 | (037C ₁₆)... | 08 ₁₆ | Pull-up control register 0 | (03FC ₁₆)... | 00 ₁₆ |
| UART2 transmit/receive control register 1 | (037D ₁₆)... | 02 ₁₆ | Pull-up control register 1(Note) | (03FD ₁₆)... | 00 ₁₆ |
| Count start flag | (0380 ₁₆)... | 00 ₁₆ | Pull-up control register 2 | (03FE ₁₆)... | 00 ₁₆ |
| Clock prescaler reset flag | (0381 ₁₆)... | 0 ⊗ ⊗ ⊗ ⊗ ⊗ ⊗ ⊗ | Port control register | (03FF ₁₆)... | 00 ₁₆ |
| One-shot start flag | (0382 ₁₆)... | 0 0 ⊗ 0 0 0 0 0 0 | Data registers (R0/R1/R2/R3) | | 0000 ₁₆ |
| Trigger select register | (0383 ₁₆)... | 00 ₁₆ | Address registers (A0/A1) | | 0000 ₁₆ |
| Up-down flag | (0384 ₁₆)... | 00 ₁₆ | Frame base register (FB) | | 0000 ₁₆ |
| Timer A0 mode register | (0396 ₁₆)... | 00 ₁₆ | Interrupt table register (INTB) | | 0000 ₁₆ |
| Timer A1 mode register | (0397 ₁₆)... | 00 ₁₆ | User stack pointer (USP) | | 0000 ₁₆ |
| Timer A2 mode register | (0398 ₁₆)... | 00 ₁₆ | Interrupt stack pointer (ISP) | | 0000 ₁₆ |
| Timer A3 mode register | (0399 ₁₆)... | 00 ₁₆ | Static base register (SB) | | 0000 ₁₆ |
| Timer A4 mode register | (039A ₁₆)... | 00 ₁₆ | Flag register (FLG) | | 0000 ₁₆ |
| Timer B0 mode register | (039B ₁₆)... | 0 0 ? ⊗ 0 0 0 0 0 | | | |
| Timer B1 mode register | (039C ₁₆)... | 0 0 ? ⊗ 0 0 0 0 0 | | | |
| Timer B2 mode register | (039D ₁₆)... | 0 0 ? ⊗ 0 0 0 0 0 | | | |
| UART0 transmit/receive mode register | (03A0 ₁₆)... | 00 ₁₆ | | | |
| UART0 transmit/receive control register 0 | (03A4 ₁₆)... | 08 ₁₆ | | | |
| UART0 transmit/receive control register 1 | (03A5 ₁₆)... | 02 ₁₆ | | | |
| Reserved register 2 | (03A8 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 Vcc level is applied to the CNVSS pin, it is 02₁₆ at a reset.

Figure 2.3.4 Device's internal status after a reset is cleared (2)

2.4 Processor Mode

2.4.1 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.

(1) Single-chip mode

In single-chip mode, only internal memory space (SFR, OSD RAM, 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.

(2) Memory expansion mode

In memory expansion mode, external memory can be accessed in addition to the internal memory space (SFR, OSD RAM, 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 "2.4.3 Bus Settings" for details.)

(3) Microprocessor mode

In microprocessor mode, the SFR, OSD RAM, 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 "2.4.3 Bus Settings" for details.)

2.4.2 Setting Processor Modes

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

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

(1) Applying Vss to CNVss pin

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

(2) Applying Vcc to CNVss pin

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

Figures 2.4.1 and 2.4.2 show the processor mode register 0 and 1.

Figure 2.4.3 shows the memory maps applicable for each of the modes.

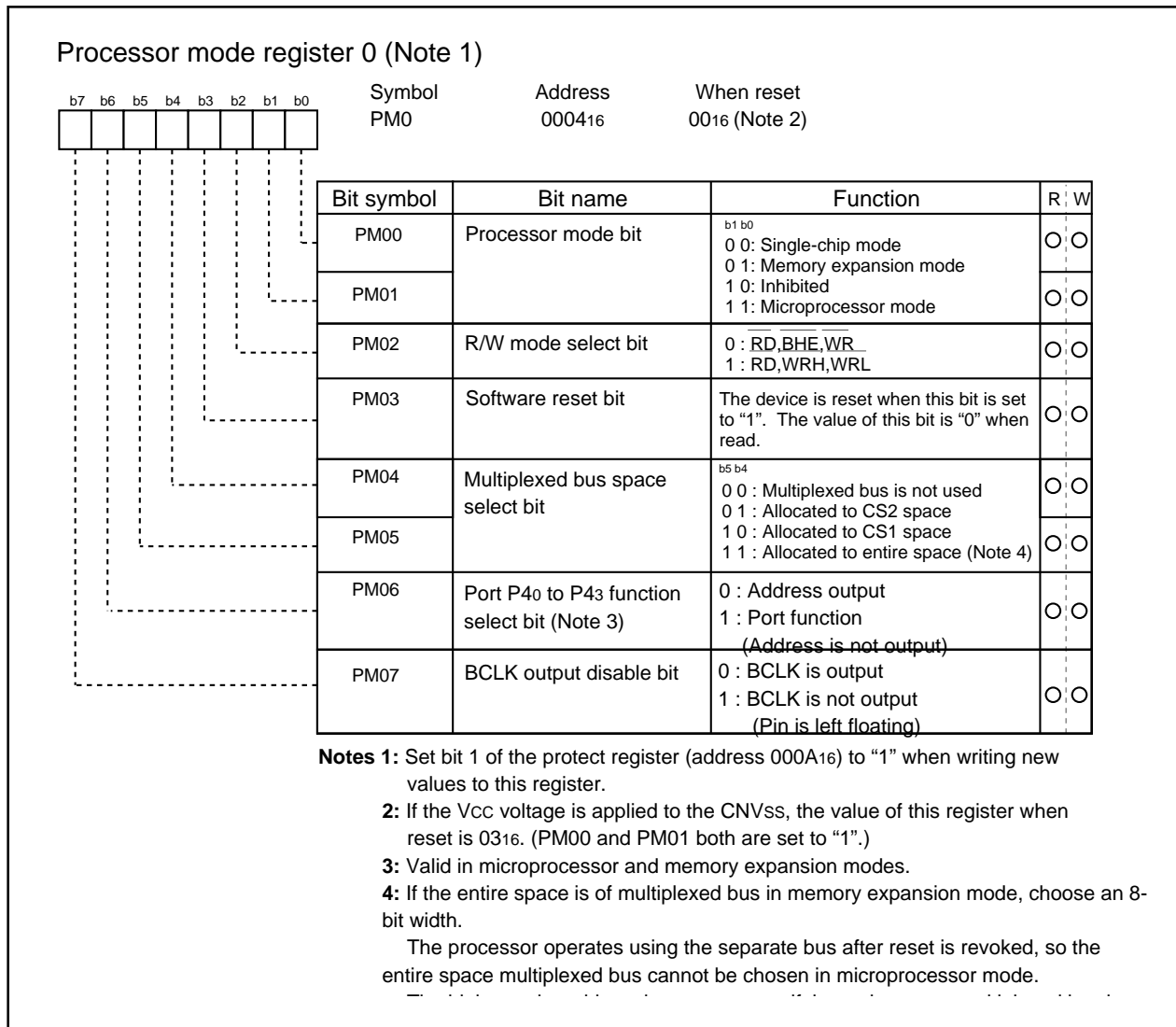


Figure 2.4.1 Processor mode register 0

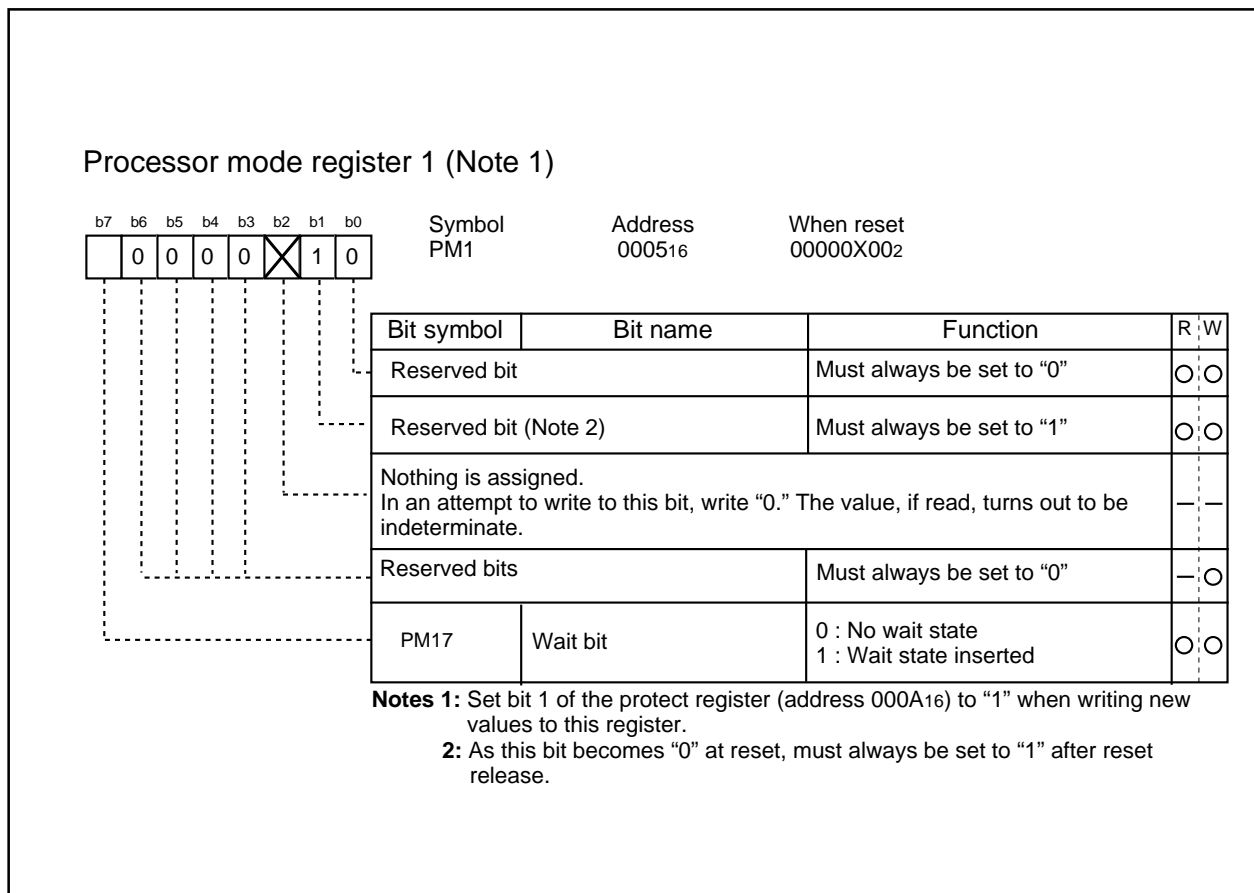


Figure 2.4.2 Processor mode register 1

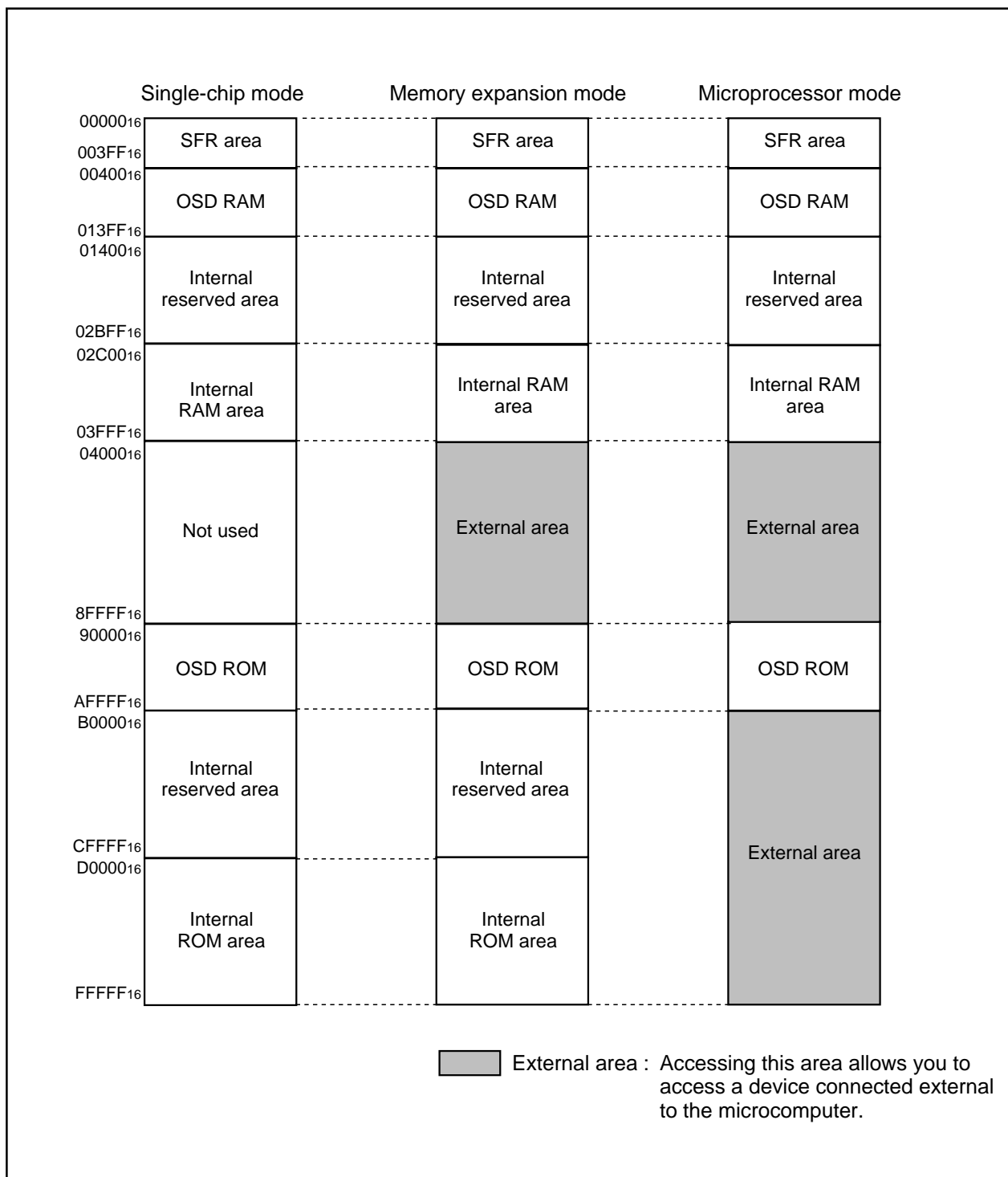


Figure 2.4.3 Memory maps in each processor mode

2.4.3 Bus Settings

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

Table 2.4.1 shows the factors used to change the bus settings.

Table 2.4.1 Factors for switching bus settings

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

(1) Selecting external address bus width

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

(2) Selecting external data bus width

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

While operating, fix the BYTE pin either to "H" or to "L."

(3) Selecting separate/multiplex bus

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

• Separate bus

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

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

• Multiplex bus

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

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

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

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

In memory expansion mode, select a 8-bit multiplex bus.

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 2.4.2 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 bits = "H" |
| P00 to P07 | I/O port | Data bus | Data bus | Data bus | Data bus | I/O port |
| P10 to P17 | I/O port | I/O port | Data bus | I/O port | Data bus | I/O port |
| P20 | I/O port | Address bus /data bus(Note 3) | Address bus | Address bus | Address bus | Address bus /data bus |
| P21 to P27 | I/O port | Address bus | Address bus /data bus(Note 3) | Address bus /data bus(Note 3) | Address bus | Address bus /data bus |
| P30 | I/O port | Address bus /data bus(Note 3) | Address bus | Address bus | Address bus | A8/D7 |
| P31 to P37 | I/O port | Address bus | Address bus | Address bus | Address bus | I/O port |
| P40 to P43 Port P40 to P43 function select bit = 1 | I/O port | I/O port | I/O port | I/O port | I/O port | I/O port |
| P40 to P43 Port P40 to P43 function select bit = 0 | I/O port | Address bus | Address bus | Address bus | Address bus | I/O port |
| P44 to P47 | I/O port | \overline{CS} (chip select) or programmable I/O port (For details, refer to "2.4.4 Bus control") | | | | |
| P50 to P53 | I/O port | Outputs \overline{RD} , \overline{WRL} , \overline{WRH} , and \overline{BCLK} or \overline{RD} , \overline{BHE} , \overline{WR} , and \overline{BCLK} (For details, refer to "2.4.4 Bus control") | | | | |
| P54 | I/O port | \overline{HLDA} | \overline{HLDA} | \overline{HLDA} | \overline{HLDA} | \overline{HLDA} |
| P55 | I/O port | \overline{HOLD} | \overline{HOLD} | \overline{HOLD} | \overline{HOLD} | \overline{HOLD} |
| P56 | I/O port | ALE | ALE | ALE | ALE | ALE |
| P57 | I/O port | \overline{RDY} | \overline{RDY} | \overline{RDY} | \overline{RDY} | \overline{RDY} |

Notes 1: In memory expansion mode, select a 8-bit multiplex bus.

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.

2: Address bus when in separate bus mode.

2.4.4 Bus Control

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

(1) Address bus/data bus

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

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

When a change is made from single-chip mode to memory expansion mode, the value of the address bus is undefined until external memory is accessed.

(2) Chip select signal

The chip select signal is output using the same pins as P44 to P47. Bits 0 to 3 of the chip select control register (address 000816) set each pin to function as a port or to output the chip select signal. The chip select control register is valid in memory expansion mode and microprocessor mode. In single-chip mode, P44 to P47 function as programmable I/O ports regardless of the value in the chip select control register.

In microprocessor mode, only $\overline{CS0}$ outputs the chip select signal after the reset state has been cancelled. $\overline{CS1}$ to $\overline{CS3}$ function as input ports. Figure 2.4.4 shows the chip select control register.

The chip select signal can be used to split the external area into as many as four blocks. Table 2.4.4 shows the external memory areas specified using the chip select signal.

Table 2.4.3 External areas specified by the chip select signals

| Chip select | Specified address range | |
|------------------|---|--|
| | Memory expansion mode | Microprocessor mode |
| $\overline{CS0}$ | 30000 ₁₆ to 8FFFF ₁₆ (384K) | 30000 ₁₆ to 8FFFF ₁₆ (384K), B0000 ₁₆ to FFFFF ₁₆ (320K) |
| $\overline{CS1}$ | 28000 ₁₆ to 2FFFF ₁₆ (32K) | 28000 ₁₆ to 2FFFF ₁₆ (32K) |
| $\overline{CS2}$ | 08000 ₁₆ to 27FFF ₁₆ (128K) | 08000 ₁₆ to 27FFF ₁₆ (128K) |
| $\overline{CS3}$ | 04000 ₁₆ to 07FFF ₁₆ (16K) | 04000 ₁₆ to 07FFF ₁₆ (16K) |

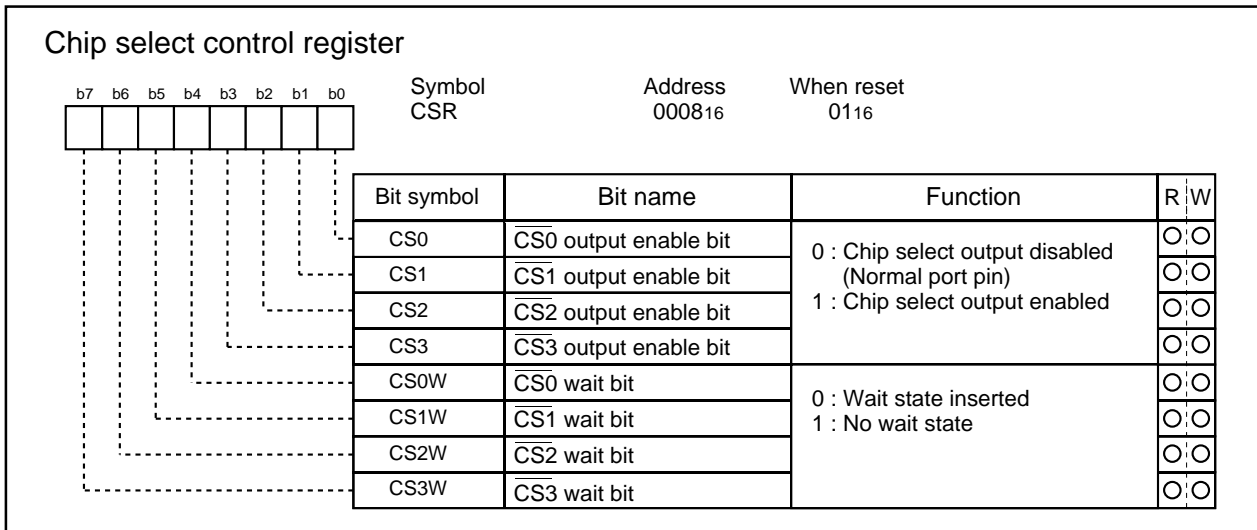


Figure 2.4.4 Chip select control register

(3) Read/write signals

With a 16-bit data bus (BYTE pin = "L"), bit 2 of the processor mode register 0 (address 000416) select the combinations of \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 000416) to "0".) Tables 2.4.4 and 2.4.5 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 000416) has been set (Note).

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

Table 2.4.4 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 2.4.5 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.

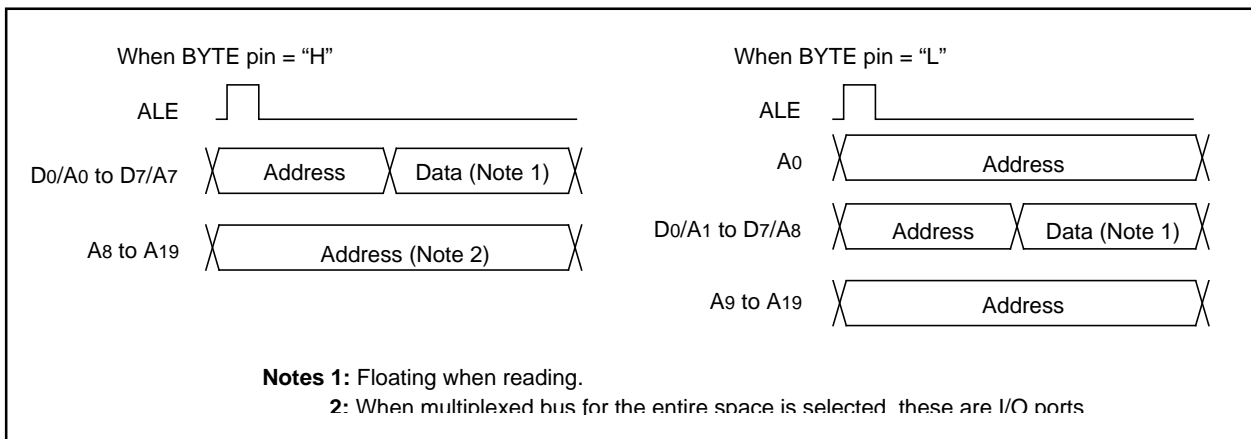


Figure 2.4.5 ALE signal and address/data bus

(5) \overline{RDY} signal

\overline{RDY} signal facilitates access of external devices that require a long time for access. As shown in Figure 2.4.6, if an “L” is being input to the \overline{RDY} pin 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 2.4.6 shows the microcomputer state in the wait state. Figure 2.4.6 shows the example of the \overline{RD} signal being extended using the \overline{RDY} signal.

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

Table 2.4.6 Microcomputer status in ready state (Note)

| Item | Status |
|--|---|
| Oscillation | On |
| R/W signal, address bus, data bus, \overline{CS} 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.

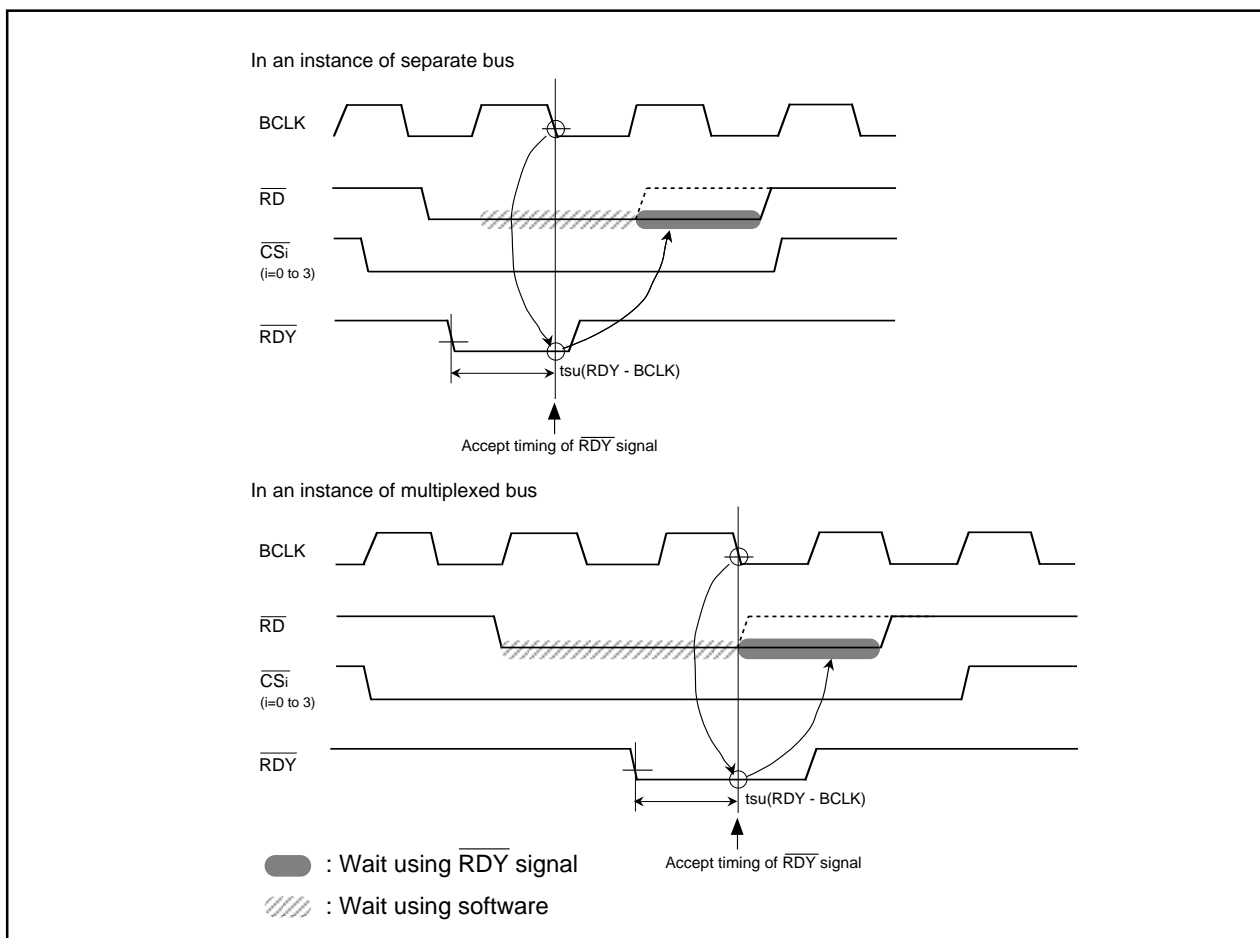


Figure 2.4.6 Example of \overline{RD} signal extended by \overline{RDY} signal

(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 2.4.7 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.

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

Figure 2.4.7 Bus-using priorities**Table 2.4.7 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 internal area is accessed

Table 2.4.8 shows the external bus status when the internal area is accessed.

Table 2.4.8 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 output of the internal clock ϕ can be selected using bit 7 of the processor mode register 0 (address 000416) (Note). The output is floating when bit 7 is set to “1”.

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

(9) Software wait

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

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

When the wait bit of the processor mode register 1 is "0", software waits can be set independently for each of the 4 areas selected using the chip select signal. Bits 4 to 7 of the chip select control register correspond to chip selects $\overline{\text{CS0}}$ to $\overline{\text{CS3}}$. 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. These bits default to "0" after the microcomputer has been reset.

The SFR area and the OSD RAM area are 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 2.4.9 shows the software wait and bus cycles. Figure 2.4.8 shows example bus timing when using software waits.

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

Table 2.4.9 Software waits and bus cycles

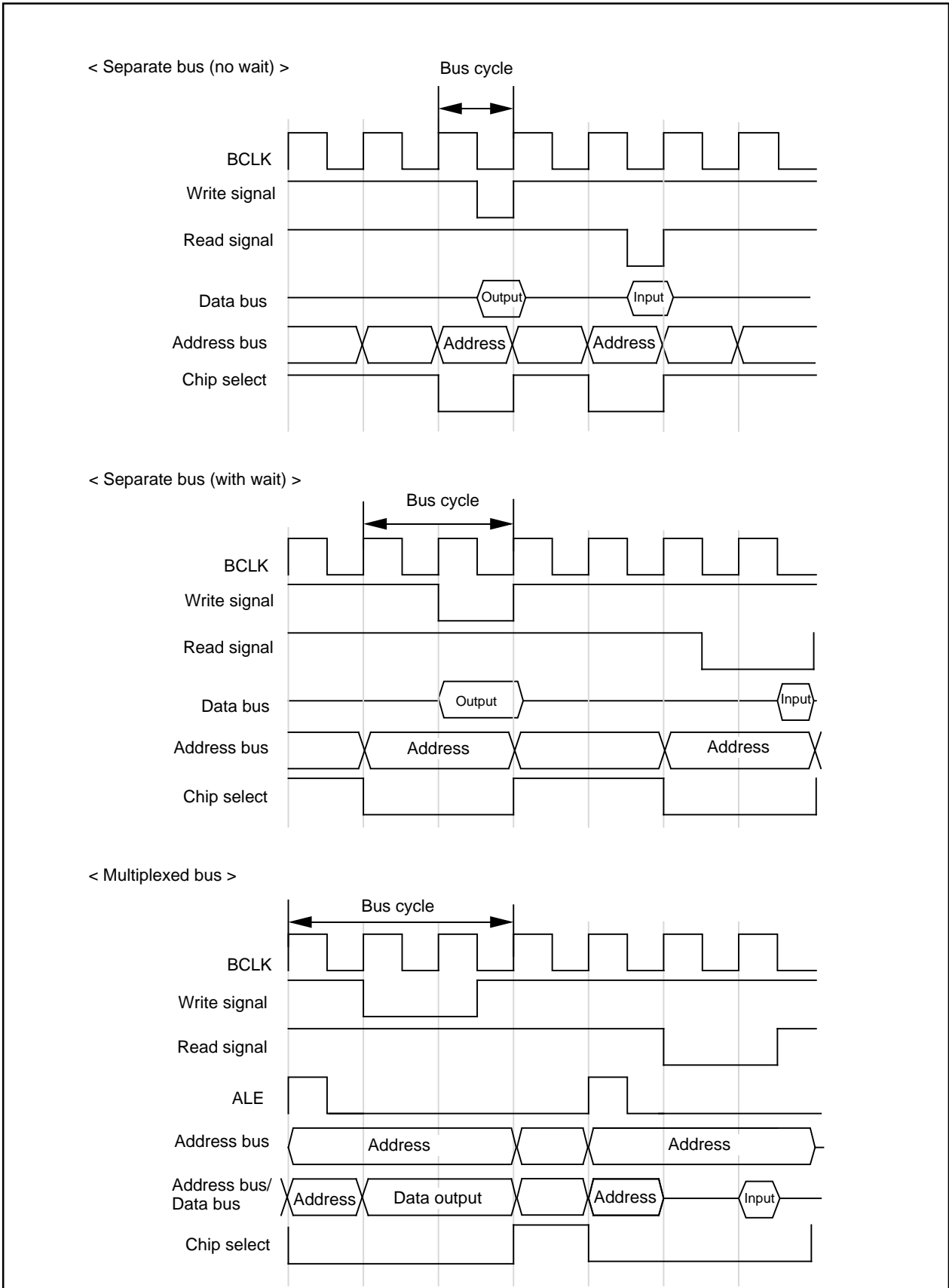


Figure 2.4.8 Typical bus timings using software wait

2.5 Clock Generating Circuit

The clock generating circuit contains 2 oscillator circuits that supply the operating clock sources to the CPU and internal peripheral units and 1 oscillator circuit that supplies the operating clock source to OSD.

Table 2.5.1. Clock oscillation circuits

| | Main clock oscillation circuit | Sub-clock oscillation circuit | OSD oscillation circuit |
|---|---|--|---|
| Use of clock | <ul style="list-style-type: none"> • CPU's operating clock source • Internal peripheral units' operating clock source | <ul style="list-style-type: none"> • CPU's operating clock source • Timer A/B's count clock source | <ul style="list-style-type: none"> • OSD's operating clock source |
| Usable oscillator | <ul style="list-style-type: none"> • Ceramic resonator (or quartz-crystal oscillator) | <ul style="list-style-type: none"> • Quartz-crystal oscillator | <ul style="list-style-type: none"> • Ceramic resonator (or quartz-crystal oscillator) • LC oscillator |
| Pins to connect oscillator | XIN, XOUT | XIN, XOUT | OSC1, OSC2 |
| Oscillation stop/restart function | Available | Available | / |
| Oscillator status immediately after reset | Oscillating | Stopped | |
| Other | Externally derived clock can be input | | |

2.5.1 Example of Oscillator Circuit

Figure 2.5.1 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 2.5.2 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 2.5.1 and 2.5.2 vary with each oscillator used. Use the values recommended by the manufacturer of your oscillator.

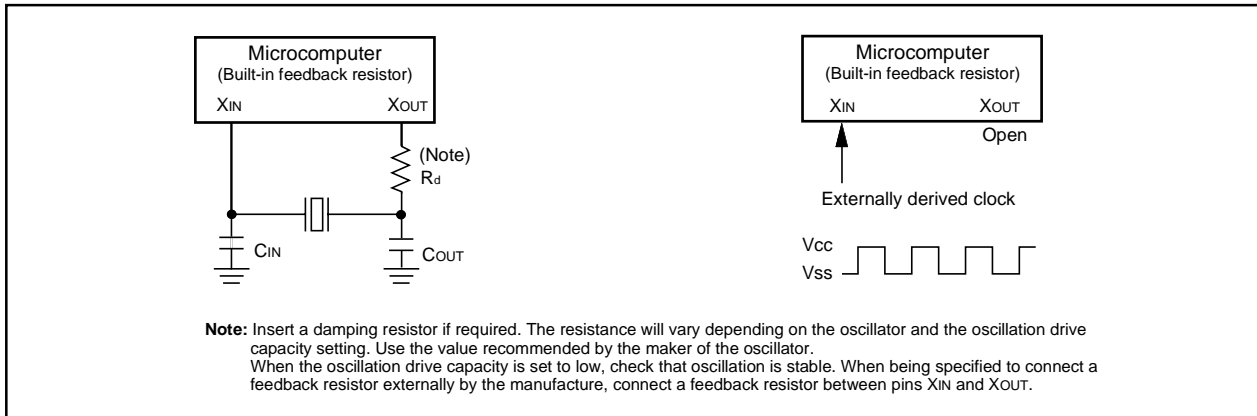


Figure 2.5.1 Examples of main clock

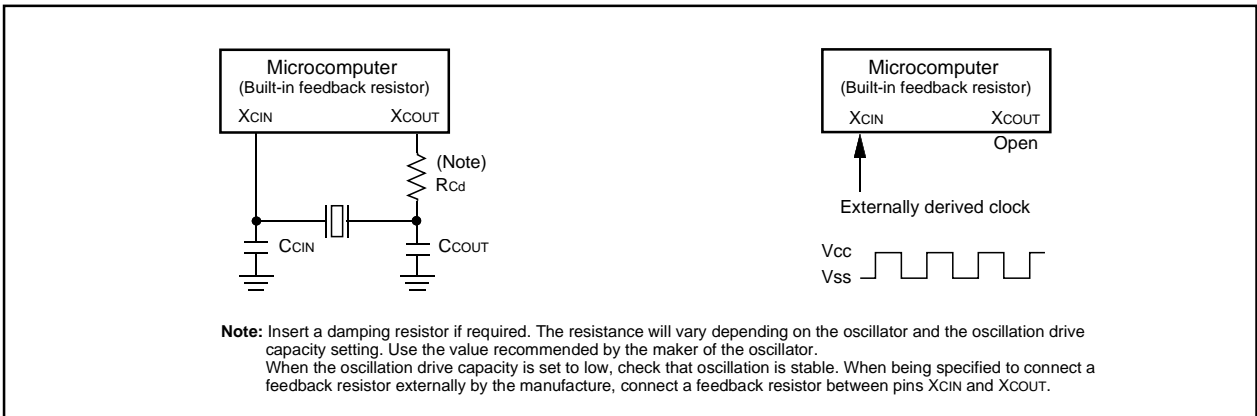


Figure 2.5.2 Examples of sub-clock

2.5.2 OSD Oscillation Circuit

The OSD clock oscillation circuit can obtain simply a clock for OSD by connecting an LC oscillator or a ceramic resonator (or a quartz-crystal oscillator) across the pins OSC1 and OSC2. Which of LC oscillator or a ceramic resonator (or a quartz-crystal oscillator) is selected by setting bits 1 and 2 of the clock control register (address 020516).

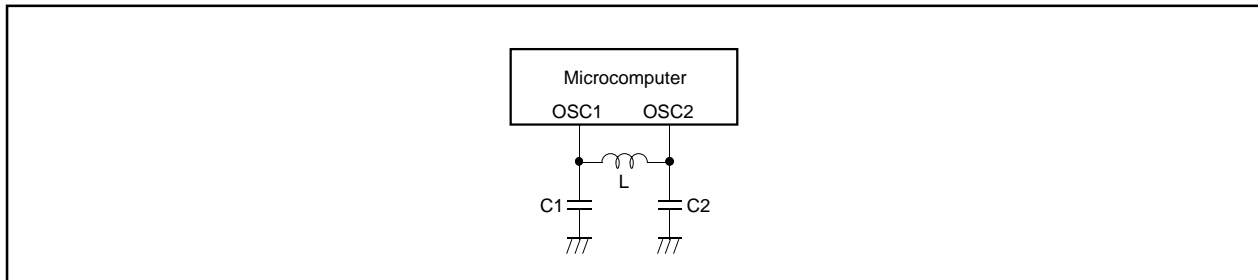


Figure 2.5.3 OSD clock connection example

2.5.3 Clock Control

Figure 2.5.4 shows the block diagram of the clock generating circuit.

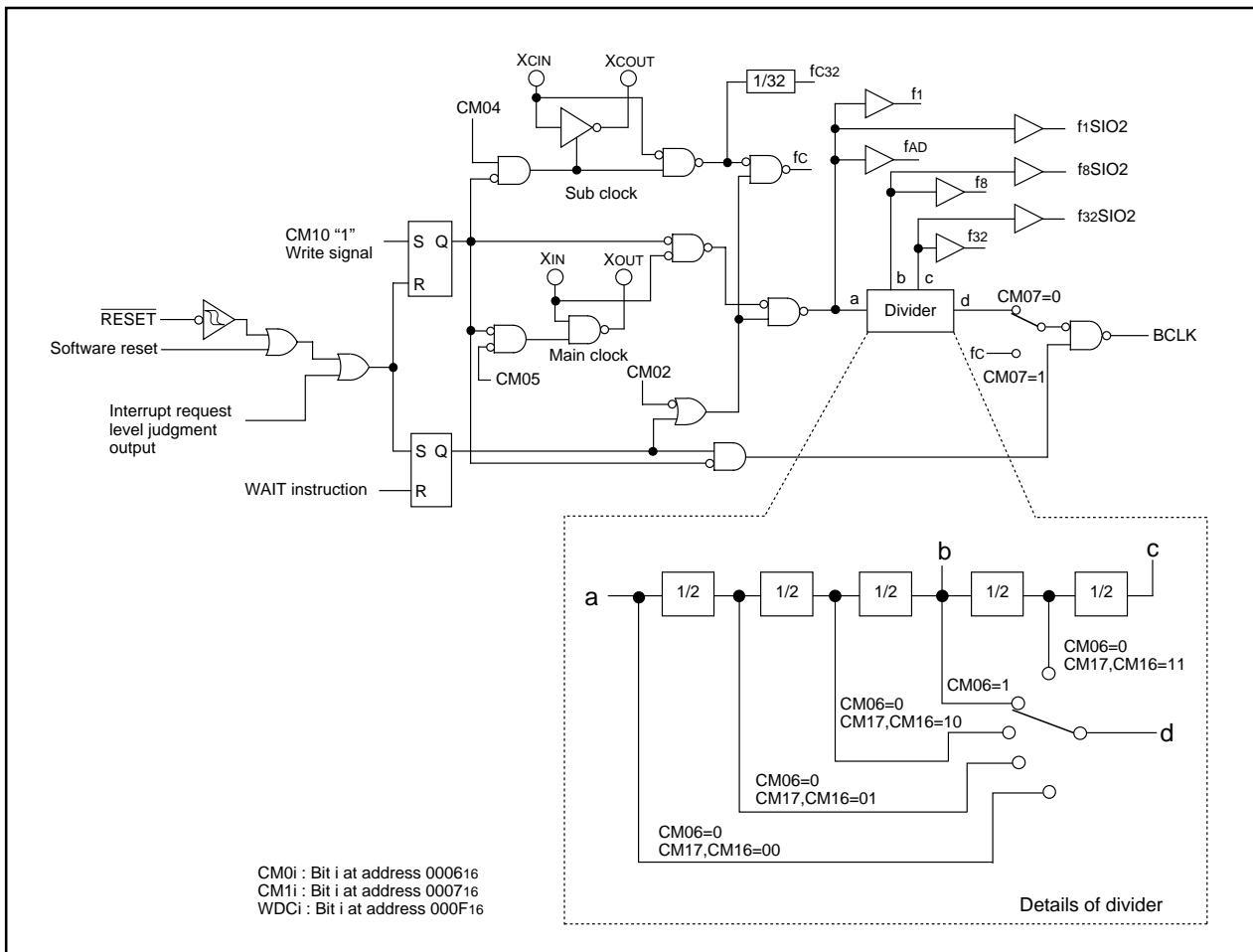


Figure 2.5.4 Clock generating circuit

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 0006₁₆). Stopping the clock, after switching the operating clock source of CPU to the sub-clock, reduces the power dissipation.

After the oscillation of the main clock oscillation circuit has stabilized, the drive capacity of the main clock oscillation circuit can be reduced using the X_{IN}-X_{OUT} drive capacity select bit (bit 5 at address 0007₁₆). Reducing the drive capacity of the main clock oscillation circuit reduces the power dissipation. This bit changes to "1" when shifting from high-speed/medium-speed mode to stop mode and at a reset. When shifting from low-speed/low power dissipation mode to stop mode, the value before stop mode is retained.

(2) Sub-clock

The sub-clock is generated by the sub clock oscillation circuit. No sub clock is generated after a reset. After oscillation is started using the port X_c select bit (bit 4 at address 0006₁₆), the sub-clock can be selected as the BCLK by using the system clock select bit (bit 7 at address 0006₁₆). However, be sure that the sub-clock oscillation has fully stabilized before switching.

After the oscillation of the sub-clock oscillation circuit has stabilized, the drive capacity of the sub-clock oscillation circuit can be reduced using the X_{CIN}-X_{COU}T drive capacity select bit (bit 3 at address 0006₁₆). Reducing the drive capacity of the sub-clock oscillation circuit reduces the power dissipation. This bit changes to "1" when shifting to stop mode and at a reset.

(3) BCLK

The internal clock ϕ is the clock that drives the CPU, and is f_c or the clock derived by dividing the main clock by 1, 2, 4, 8, or 16. The BCLK is derived by dividing the main clock by 8 after a reset. The BCLK signal can be output from pin BCLK by the BCLK output disable bit (bit 7 at address 0004₁₆) in the memory expansion and the microprocessor modes.

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

(4) Peripheral function clock (f₁, f₈, f₃₂, f_{1SIO2}, f_{8SIO2}, f_{32SIO2}, f_{AD})

The clock for the peripheral devices is derived by dividing the main clock by 1, 8 or 32. The peripheral function clock is stopped by stopping the main clock or by setting the WAIT peripheral function clock stop bit (bit 2 at 0006₁₆) to "1" and then executing a WAIT instruction.

(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.

2.5.4 Clock Output

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

2.5.5 Stop Mode

Writing "1" to the all-clock stop control bit (bit 0 at address 0007₁₆) stops all oscillation and the microcomputer enters stop mode. In stop mode, the content of the internal RAM is retained provided that V_{CC} remains above 4.5V.

Because the oscillation, BCLK, f₁ to f₃₂, f₁SIO₂ to f₃₂SIO₂, 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 B operates provided that the event counter mode is set to an external pulse, and UART_i (i = 0, 2) functions provided an external clock is selected. Table 2.5.2 shows the status of the ports in stop mode.

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

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

Table 2.5.2 Port status during stop mode

| Pin | | Memory expansion mode 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 |
| CLKOUT | When f _c 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 |

2.5.6 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 2.5.3 shows the status of the ports in wait mode.

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

Table 2.5.3 Port status during wait mode

| Pin | | Memory expansion mode Microprocessor mode | Single-chip mode |
|---|--|--|--|
| Address bus, data bus, $\overline{CS0}$ to $\overline{CS3}$ | | Retains status before wait mode | |
| RD, WR, BHE, \overline{WRL} , WRH | | "H" | |
| HLDA, BCLK | | "H" | |
| ALE | | "H" | |
| Port | | Retains status before wait mode | Retains status before wait mode |
| CLKOUT | 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". When the WAIT peripheral function clock stop bit is "1", the status immediately prior to entering wait mode is maintained. |

2.5.7 Status Transition of BCLK

Power dissipation can be reduced and low-voltage operation achieved by changing the count source for BCLK. Table 2.5.4 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 0006₁₆) is set to "1". The following shows the operational modes of internal clock ϕ .

(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

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

(7) Low power dissipation mode

fc is the BCLK and the main clock is stopped.

Note: When switching the count source for BCLK between X_{IN} and X_{CIN}, it needs that the oscillation of the switched count source is sufficiently stable. Shift after taking the oscillation stabilizing time by software.

Table 2.5.4 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 |

2.5.8 Power Control

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

Modes

Power control is available in three modes.

(1) 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.

(2) Wait mode

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

(3) 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 2.5.7 is the state transition diagram of the above modes.

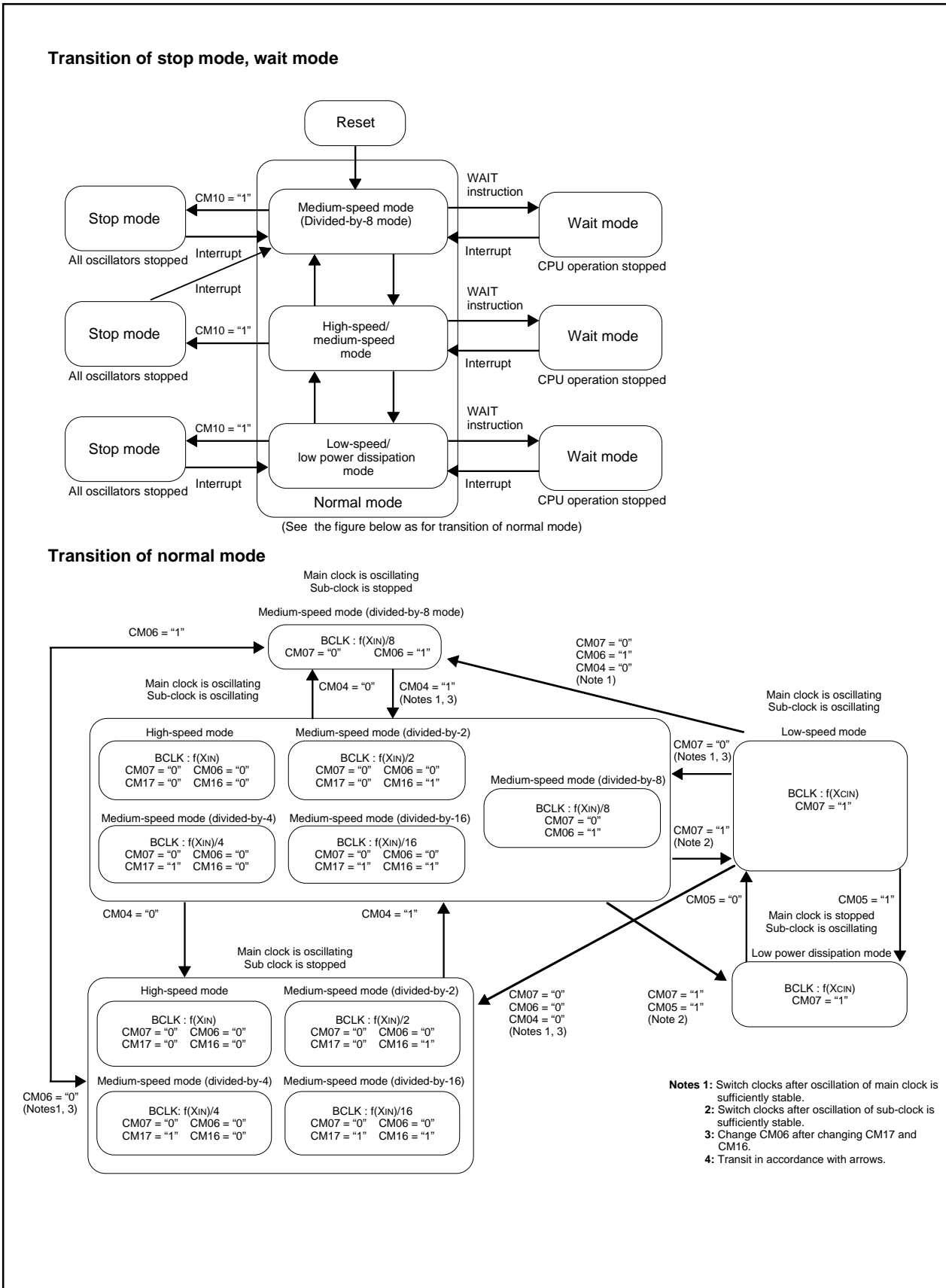


Figure 2.5.7 State transition diagram of Power control mode

2.6 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 2.6.1 shows the protect register. The values in the processor mode register 0 (address 0004₁₆), processor mode register 1 (address 0005₁₆), system clock control register 0 (address 0006₁₆), system clock control register 1 (address 0007₁₆) and port P9 direction register (address 03F3₁₆) can only be changed when the respective bit in the protect register is set to "1". Therefore, important outputs can be allocated to port P9.

If, after "1" (write-enabled) has been written to the port P9 direction register write-enable bit (bit 2 at address 000A₁₆), 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 000A₁₆) and processor mode register 0 and 1 write-enable bit (bit 1 at 000A₁₆) do not automatically return to "0" after a value has been written to an address. The program must therefore be written to return these bits to "0".

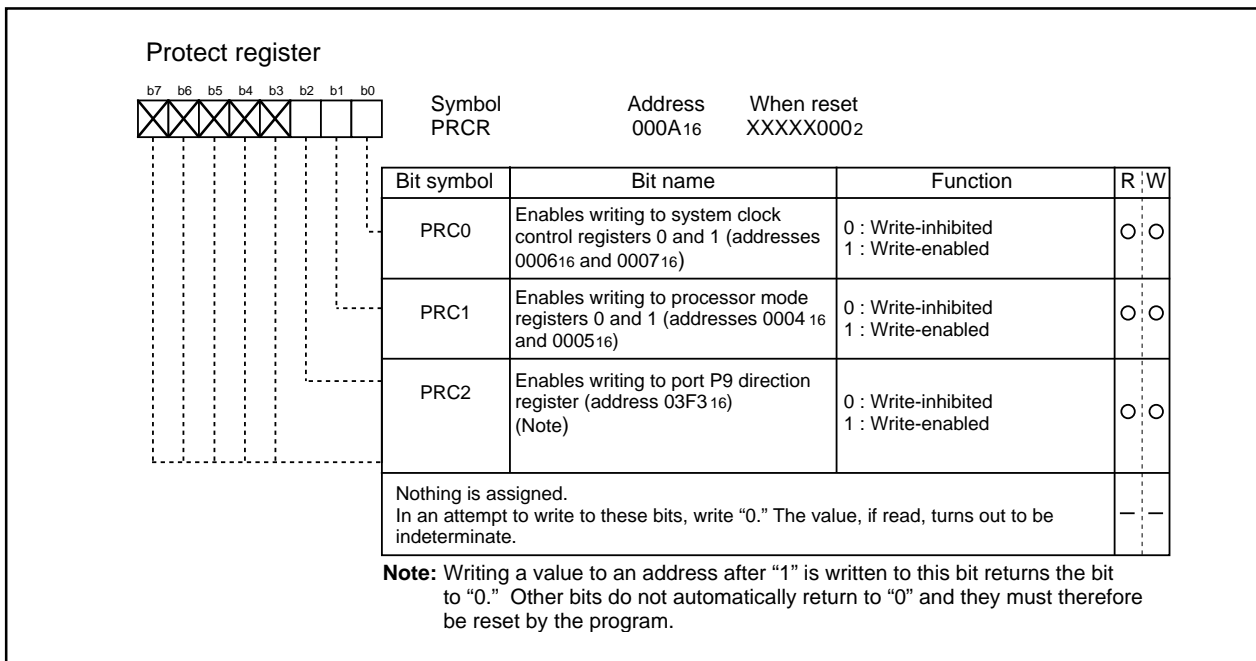


Figure 2.6.1 Protect register

2.7 Interrupts

2.7.1 Type of Interrupts

Figure 2.7.1 lists the types of interrupts.

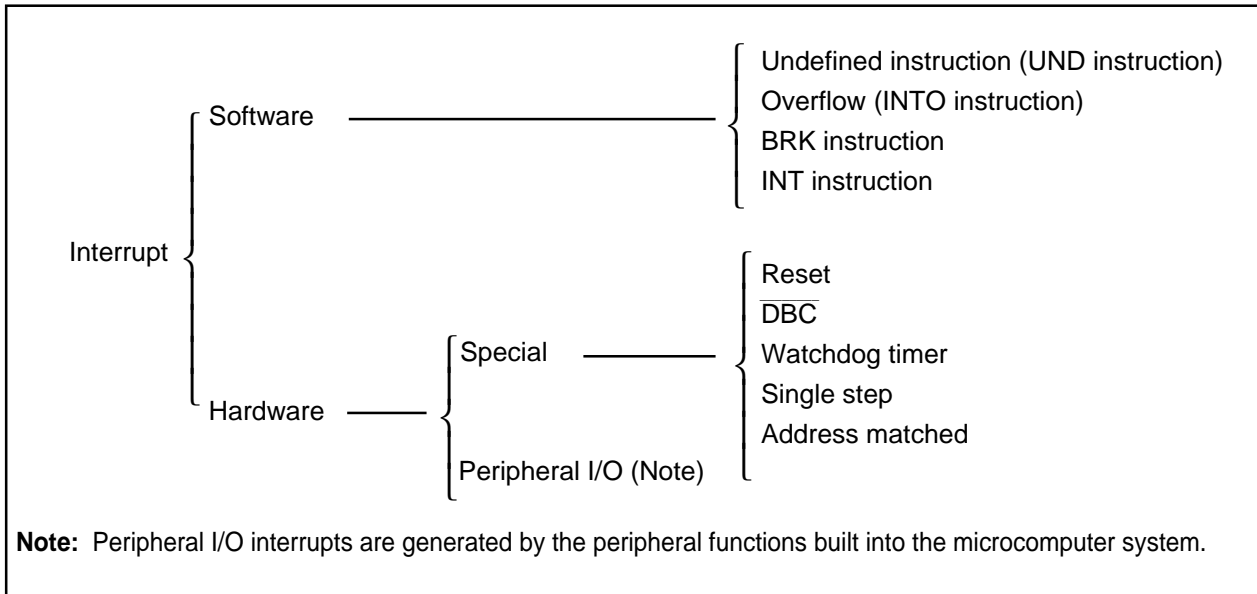


Figure 2.7.1 Classification of interrupts

- 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.

2.7.2 Software Interrupts

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

- **Undefined instruction interrupt**

An undefined instruction interrupt occurs when executing the UND instruction.

- **Overflow interrupt**

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

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

- **BRK interrupt**

A BRK interrupt occurs when executing the BRK instruction.

- **INT interrupt**

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

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

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

2.7.3 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.

- **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 INI 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 DMA generates.

- **V_{SYNC} interrupt**

V_{SYNC} interrupt occurs if a V_{SYNC} edge is input.

- **A-D conversion interrupt**

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

- **UART0 transmission, UART2 transmission interrupts**

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

- **UART0 reception, UART2 reception interrupts**

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

- **Multi-master I²C-BUS interface 0 and multi-master I²C-BUS interface 1 interrupts**

This is an interrupt that the serial I/O transmission/reception is completed, or a STOP condition is detected.

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

These are interrupts that timer A generates

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

These are interrupts that timer B generates.

- **$\overline{\text{INT0}}$ interrupt and $\overline{\text{INT1}}$ 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.

- **OSD1 interrupt and OSD2 interrupt**

These are interrupts that OSD display is completed.

- **Data slicer interrupt**

This is an interrupt that data slicer circuit requests.

2.7.4 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 2.7.2 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.

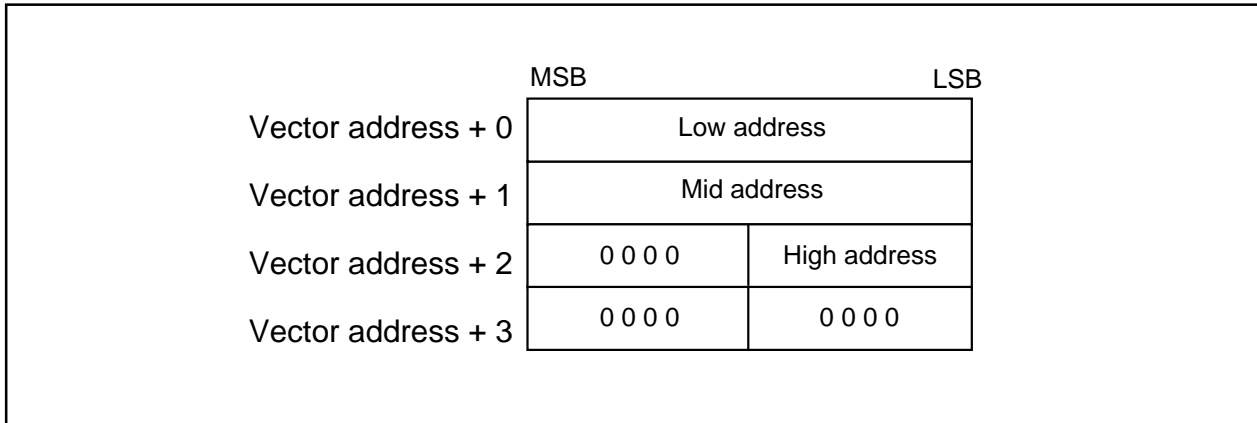


Figure 2.7.2 Format for specifying interrupt vector addresses

(1) 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 2.7.1 shows the interrupts assigned to the fixed vector tables and addresses of vector tables.

Table 2.7.1 Interrupts assigned to the fixed vector tables and addresses of vector tables

| Interrupt source | Vector table addresses Address (L) to address (H) | Remarks |
|-----------------------|--|--|
| Undefined instruction | FFFD ₁₆ to FFFD ₁₆ | Interrupt on UND instruction |
| Overflow | FFFE ₀ ₁₆ to FFFE ₃ ₁₆ | Interrupt on INTO instruction |
| BRK instruction | FFFE ₄ ₁₆ to FFFE ₇ ₁₆ | If the vector is filled with FF ₁₆ , program execution starts from the address shown by the vector in the variable vector table |
| Address match | FFFE ₈ ₁₆ to FFFE _B ₁₆ | There is an address-matching interrupt enable bit |
| Single step (Note) | FFFE _C ₁₆ to FFFE _F ₁₆ | Do not use |
| Watchdog timer | FFFF ₀ ₁₆ to FFFF ₃ ₁₆ | |
| DBC (Note) | FFFF ₄ ₁₆ to FFFF ₇ ₁₆ | Do not use |
| Reserved source | FFFE ₈ ₁₆ to FFFE _B ₁₆ | Do not use |
| Reset | FFFF _C ₁₆ to FFFF _F ₁₆ | |

Note: Interrupts used for debugging purposes only.

(2) Variable 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 2.7.2 shows the interrupts assigned to the fixed vector tables and addresses of vector tables.

Table 2.7.2 Interrupts assigned to the variable vector tables and addresses of vector tables

| Software interrupt number | Vector table address Address (L) to address (H) | Interrupt source | Remarks |
|--|--|---|-------------------------|
| Software interrupt number 0 | +0 to +3 (Note) | BRK instruction | Cannot be masked I flag |
| Software interrupt number 4 | +16 to +19 (Note) | OSD1 | |
| Software interrupt number 5 | +20 to +23 (Note) | Reserved source | |
| Software interrupt number 6 | +24 to +27 (Note) | Reserved source | |
| Software interrupt number 7 | +28 to +31 (Note) | Reserved source | |
| Software interrupt number 8 | +32 to +35 (Note) | OSD2 | |
| Software interrupt number 9 | +36 to +39 (Note) | Multi-master I ² C-BUS interface 1 | |
| Software interrupt number 10 | +40 to +43 (Note) | Bus collision detection | |
| Software interrupt number 11 | +44 to +47 (Note) | DMA0 | |
| Software interrupt number 12 | +48 to +51 (Note) | DMA1 | |
| Software interrupt number 13 | +52 to +55 (Note) | Multi-master I ² C-BUS interface 0 | |
| Software interrupt number 14 | +56 to +59 (Note) | A-D conversion | |
| Software interrupt number 15 | +60 to +63 (Note) | UART2 transmit | |
| Software interrupt number 16 | +64 to +67 (Note) | UART2 receive | |
| Software interrupt number 17 | +68 to +71 (Note) | UART0 transmit | |
| Software interrupt number 18 | +72 to +75 (Note) | UART0 receive | |
| Software interrupt number 19 | +76 to +79 (Note) | Data slicer | |
| Software interrupt number 20 | +80 to +83 (Note) | VSYNC | |
| Software interrupt number 21 | +84 to +87 (Note) | Timer A0 | |
| Software interrupt number 22 | +88 to +91 (Note) | Timer A1 | |
| Software interrupt number 23 | +92 to +95 (Note) | Timer A2 | |
| Software interrupt number 24 | +96 to +99 (Note) | Timer A3 | |
| Software interrupt number 25 | +100 to +103 (Note) | Timer A4 | |
| Software interrupt number 26 | +104 to +107 (Note) | Timer B0 | |
| Software interrupt number 27 | +108 to +111 (Note) | Timer B1 | |
| Software interrupt number 28 | +112 to +115 (Note) | Timer B2 | |
| Software interrupt number 29 | +116 to +119 (Note) | $\overline{\text{INT}}_0$ | |
| Software interrupt number 30 | +120 to +123 (Note) | $\overline{\text{INT}}_1$ | |
| Software interrupt number 31 | +124 to +127 (Note) | Reserved source | |
| Software interrupt number 32 to Software interrupt number 63 | +128 to +131 (Note) to +252 to +255 (Note) | Software interrupt | Cannot be masked I flag |

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

2.7.5 Interrupt Control

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

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

Figure 2.7.3 shows the interrupt control registers.

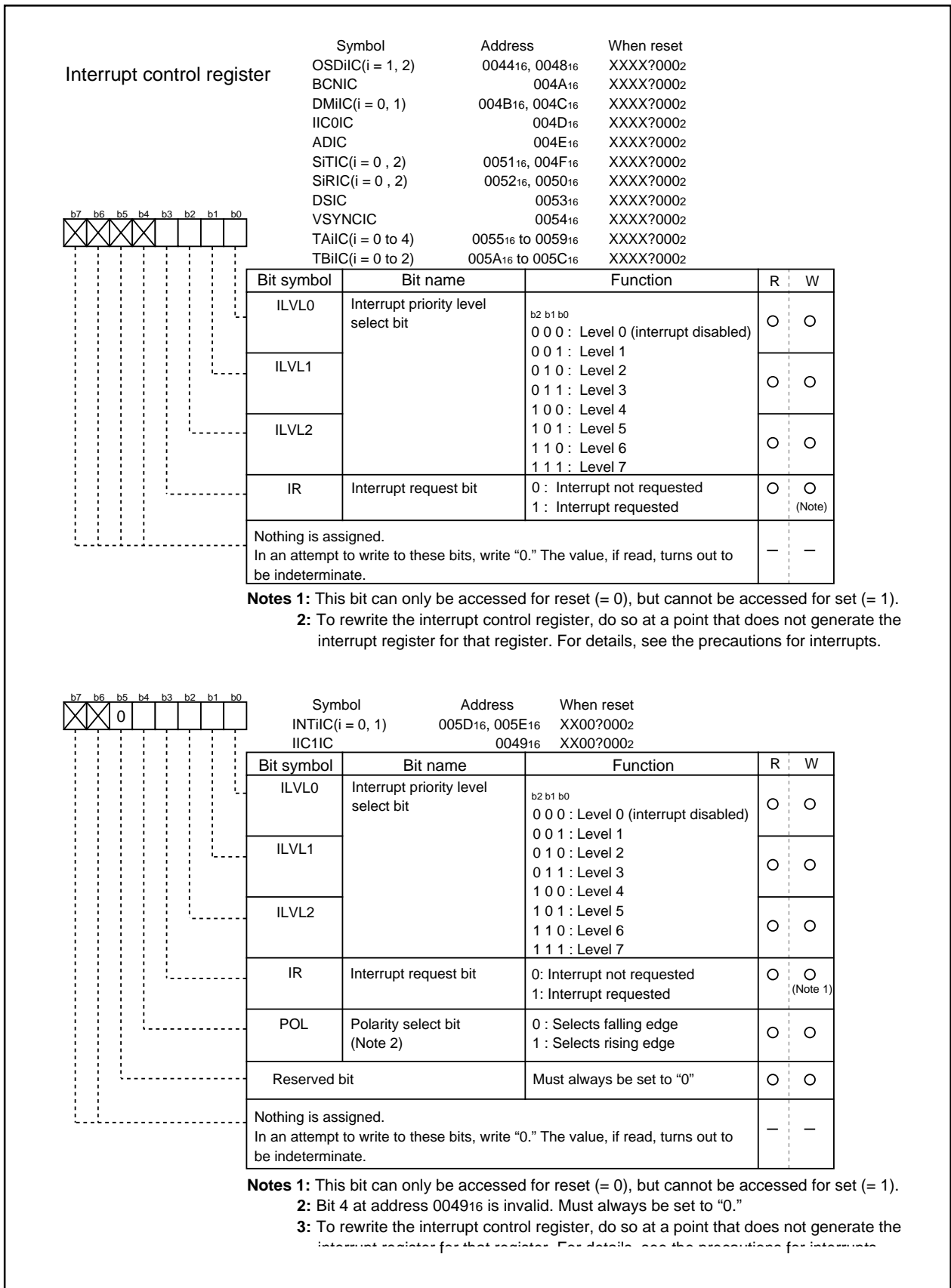


Figure 2.7.3 Interrupt control registers

2.7.6 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.

2.7.7 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").

2.7.8 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 2.7.3 shows the settings of interrupt priority levels and Table 2.7.4 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 2.7.3 Settings of interrupt priority levels

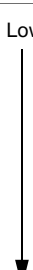
| Interrupt priority level select bit | Interrupt priority level | Priority order |
|-------------------------------------|------------------------------|--|
| b2 b1 b0 0 0 0 | Level 0 (interrupt disabled) | ———— |
| 0 0 1 | Level 1 | Low  High |
| 0 1 0 | Level 2 | |
| 0 1 1 | Level 3 | |
| 1 0 0 |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

Table 2.7.4 Interrupt levels enabled according to the contents of the IPL

| IPL | Enabled interrupt priority levels |
|---|--|
| IPL ₂ IPL ₁ IPL ₀ 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 |

2.7.9 Rewrite 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

2.7.10 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 00000₁₆.
- (2) Saves the content of the flag register (FLG) as it was immediately before the start of interrupt sequence in the temporary register (Note) within the CPU.
- (3) Sets the interrupt enable flag (I flag), the debug flag (D flag), and the stack pointer select flag (U flag) to "0" (the U flag, however does not change if the INT instruction, in software interrupt numbers 32 through 63, is executed)
- (4) Saves the content of the temporary register (Note 1) within the CPU in the stack area.
- (5) Saves the content of the program counter (PC) in the stack area.
- (6) Sets the interrupt priority level of the accepted instruction in the IPL.

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

Note: This register cannot be utilized by the user.

2.7.11 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 2.7.4 shows the interrupt response time.

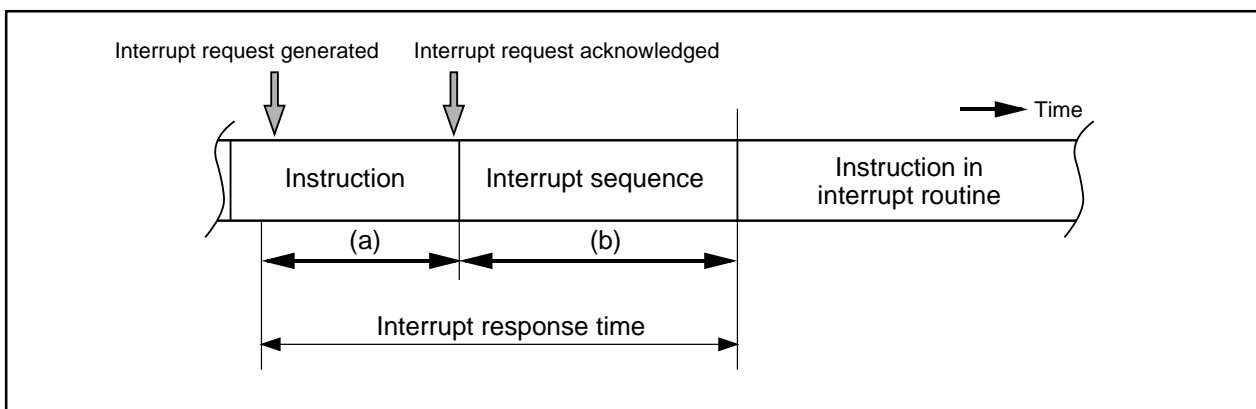


Figure 2.7.4 Interrupt response time

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 2.7.5.

Table 2.7.5 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) |

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

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

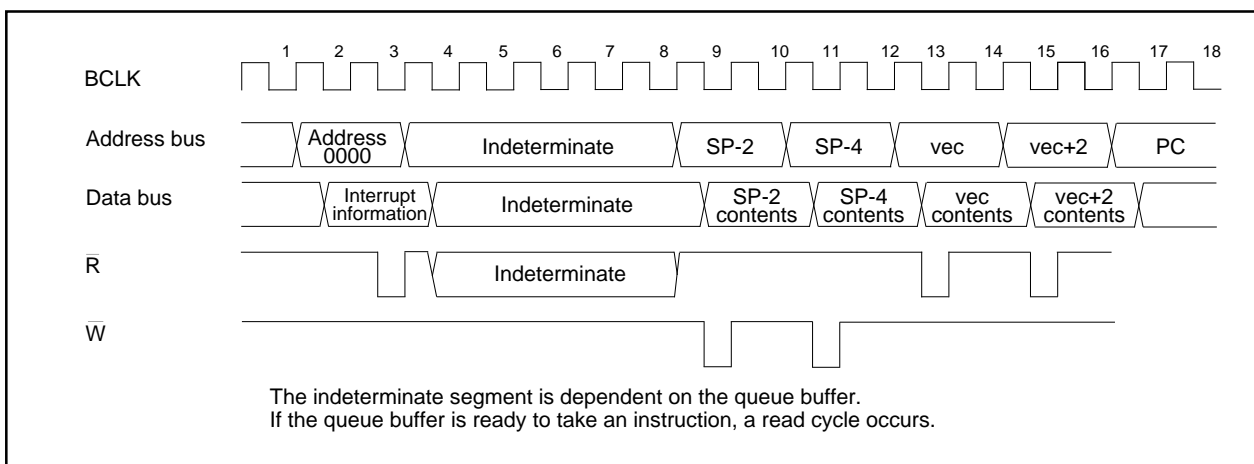


Figure 2.7.5 Time required for executing the interrupt sequence

2.7.12 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 2.7.6 is set in the IPL.

Table 2.7.6 Relationship between interrupts without interrupt priority levels and IPL

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

2.7.13 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 2.7.6 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).

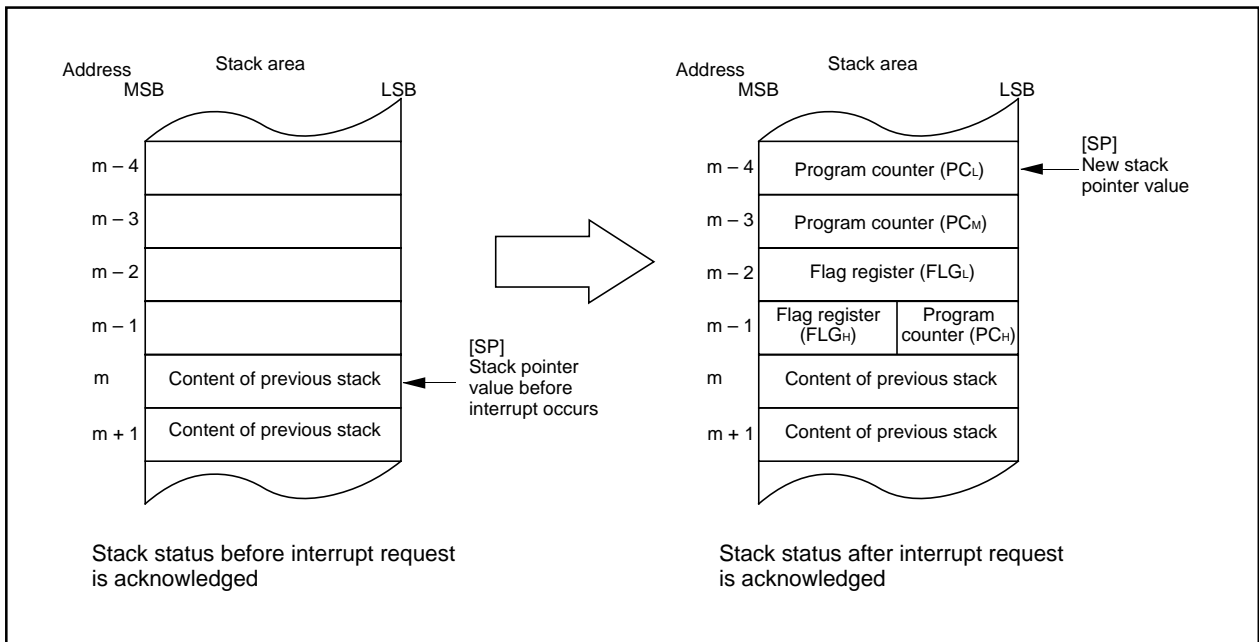


Figure 2.7.6 State of stack before and after acceptance of interrupt request

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 2.7.7 shows the operation of the saving registers.

Note: Stack pointer indicated by U flag.

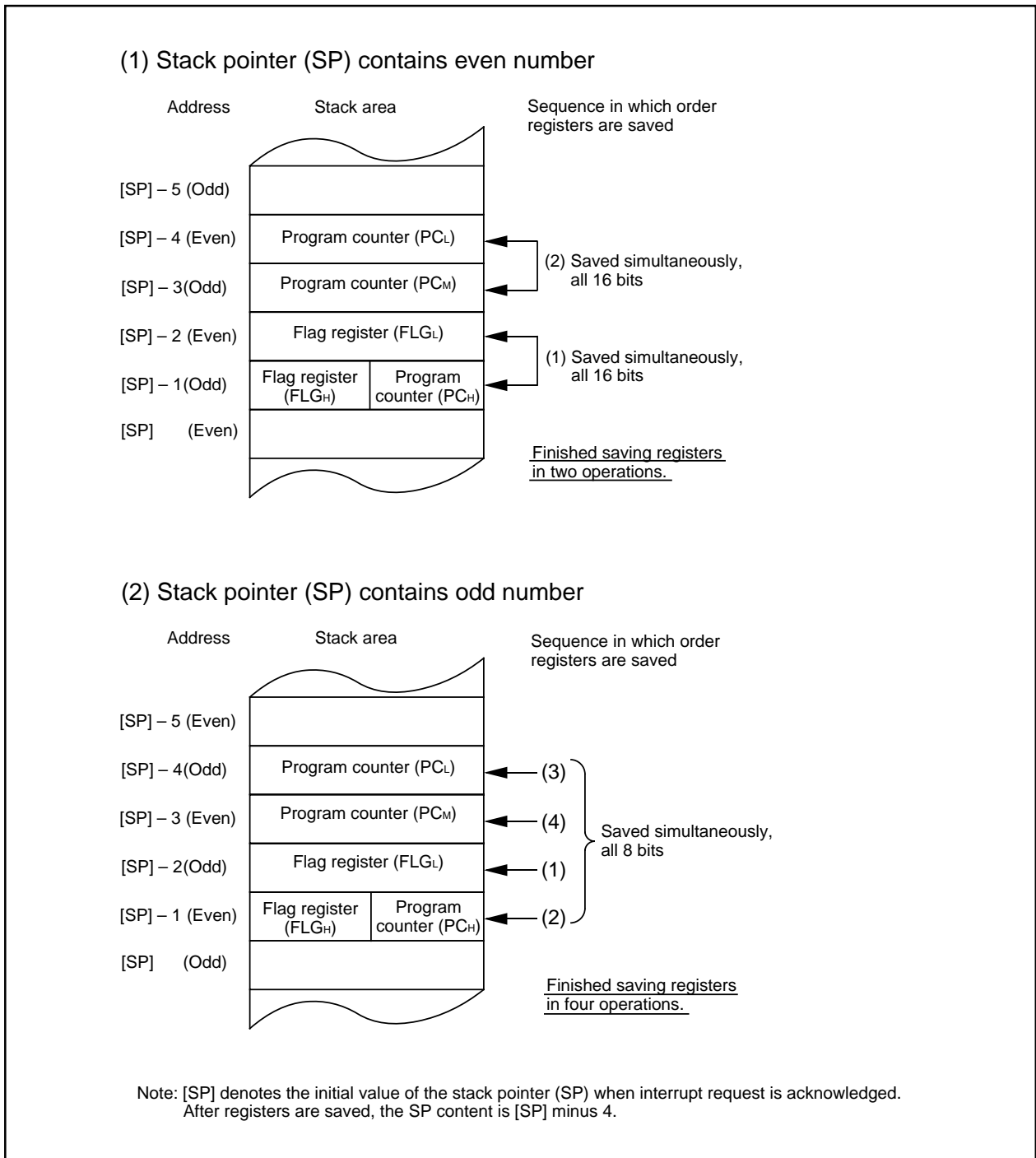


Figure 2.7.7 Operation of saving registers

2.7.14 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.

2.7.15 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 2.7.8 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.

2.7.16 Interrupt Priority Level Resolution Circuit

When two or more interrupts are generated simultaneously, this circuit selects the interrupt with the highest priority level.

Figure 2.7.9 shows the circuit that judges the interrupt priority level.

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

Figure 2.7.8 Hardware interrupts priorities

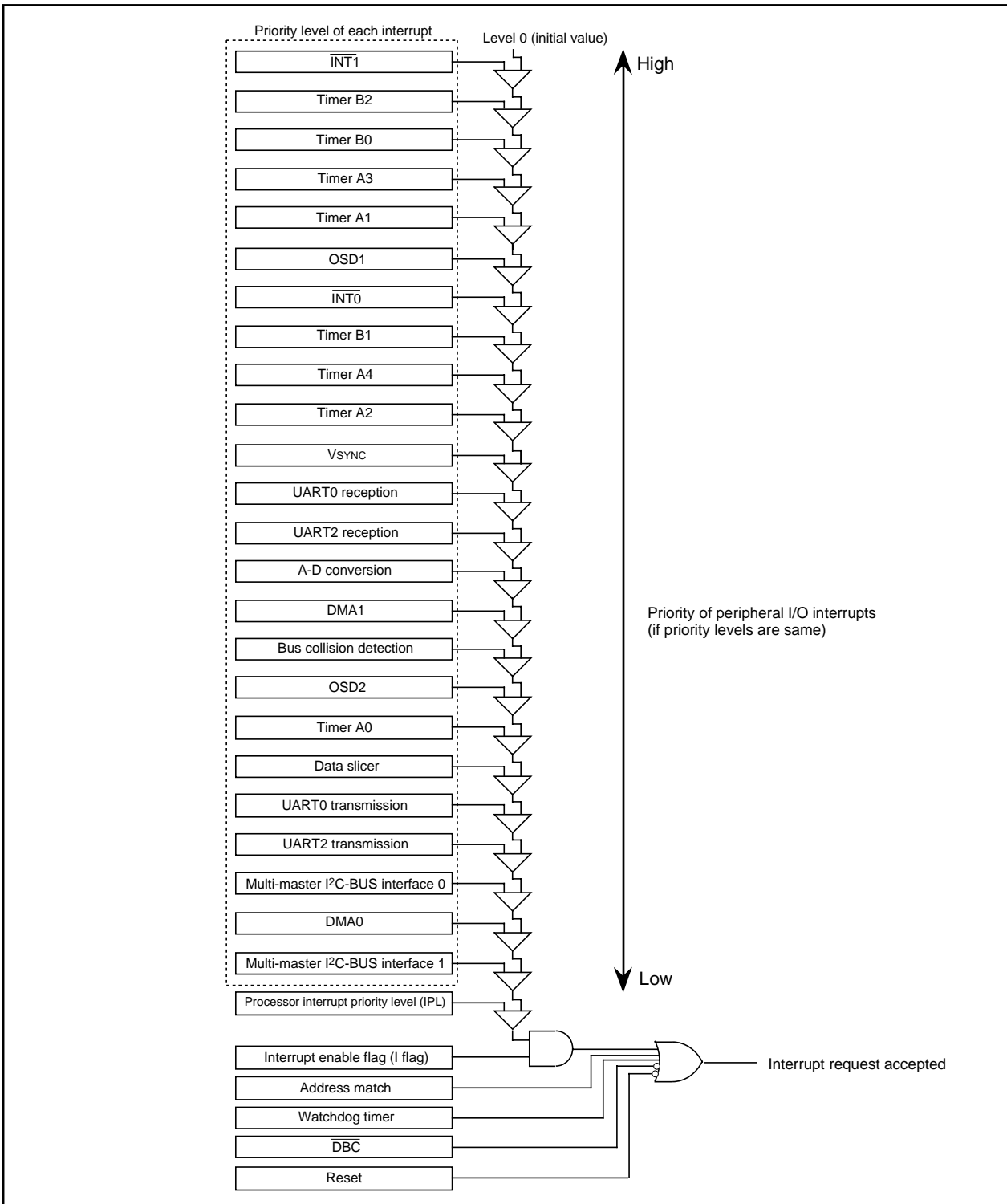


Figure 2.7.9 Maskable interrupts priorities (peripheral I/O interrupts)

2.7.17 INT Interrupt

$\overline{INT0}$ and $\overline{INT1}$ are triggered by the edges of external inputs. The edge polarity is selected using the polarity select bit.

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 (035F₁₆). To select both edges, set the polarity switching bit of the corresponding interrupt control register to ‘falling edge’ (“0”).

Figure 2.7.10 shows the Interrupt control reserved register, Figure 2.7.11 shows the Interrupt request cause select register.

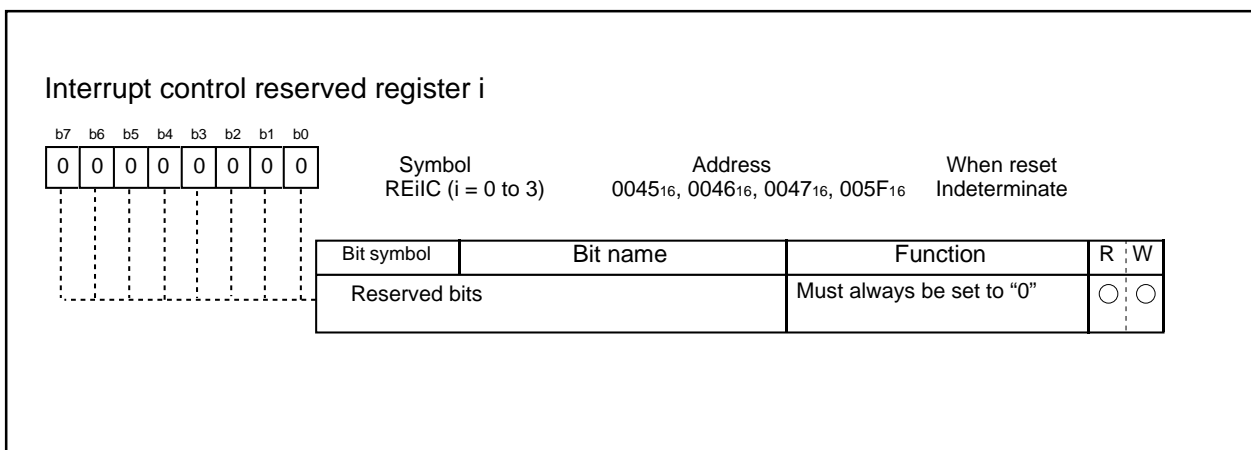


Figure 2.7.10 Interrupt control reserved register i (i = 0 to 3)

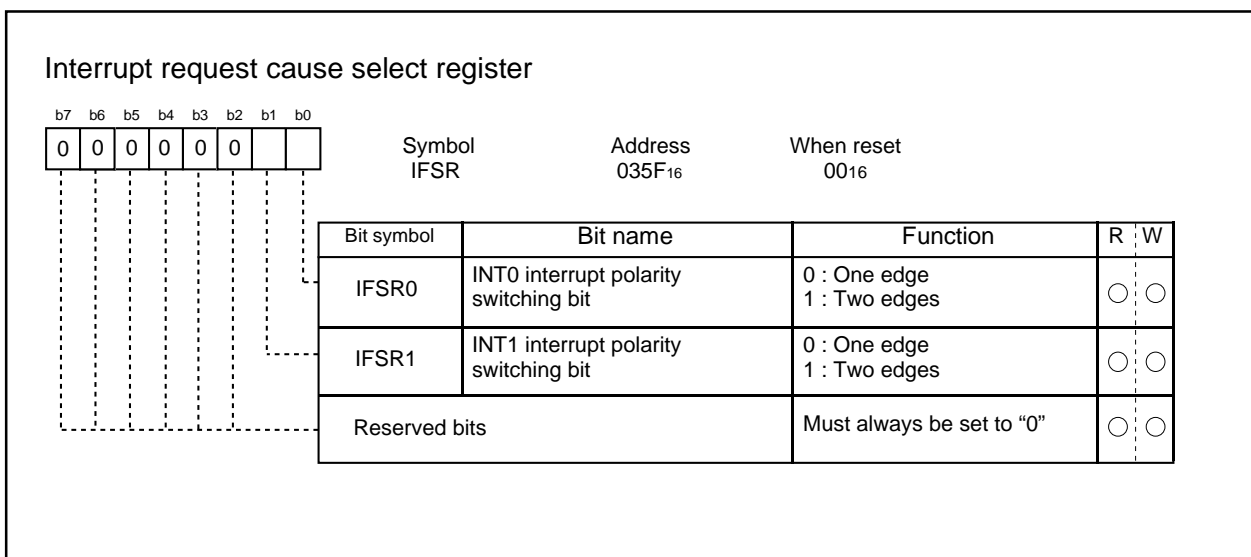


Figure 2.7.11 Interrupt request cause select register

2.7.18 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. Figures 2.7.12 and 2.7.13 show the address match interrupt-related registers.

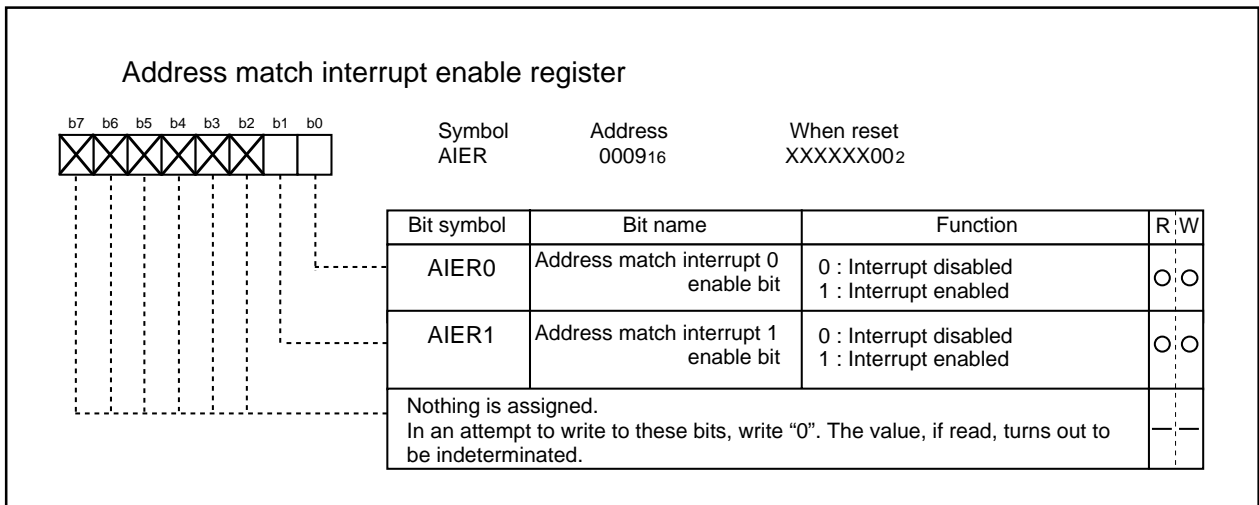


Figure 2.7.12 Address match interrupt enable register

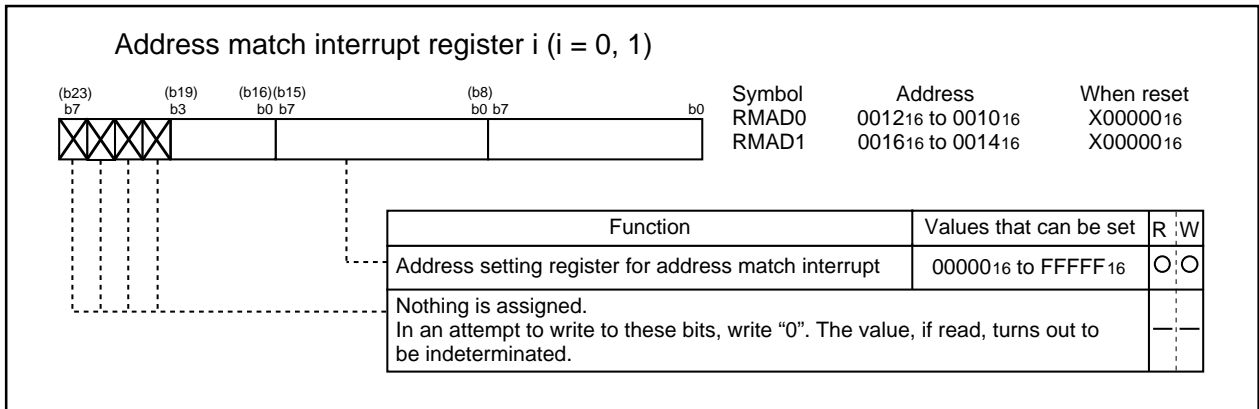


Figure 2.7.13 Address match interrupt register i (i = 0, 1)

2.7.19 Precautions for Interrupts

(1) Reading address 00000₁₆

- 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 00000₁₆ will then be set to "0".

Reading address 00000₁₆ 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 00000₁₆ by software.

(2) Setting the stack pointer

- The value of the stack pointer immediately after reset is initialized to 0000₁₆. 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.

(3) External interrupt

- Either an "L" level or an "H" level of at least 250 ns width is necessary for the signal input to pins $\overline{\text{INT}}_0$ and $\overline{\text{INT}}_1$ regardless of the CPU operation clock.
- When the polarity of the $\overline{\text{INT}}_0$ and $\overline{\text{INT}}_1$ pins is changed, the interrupt request bit is sometimes set to "1". After changing the polarity, set the interrupt request bit to "0". Figure 2.7.14 shows the procedure for changing the $\overline{\text{INT}}$ interrupt generate factor.

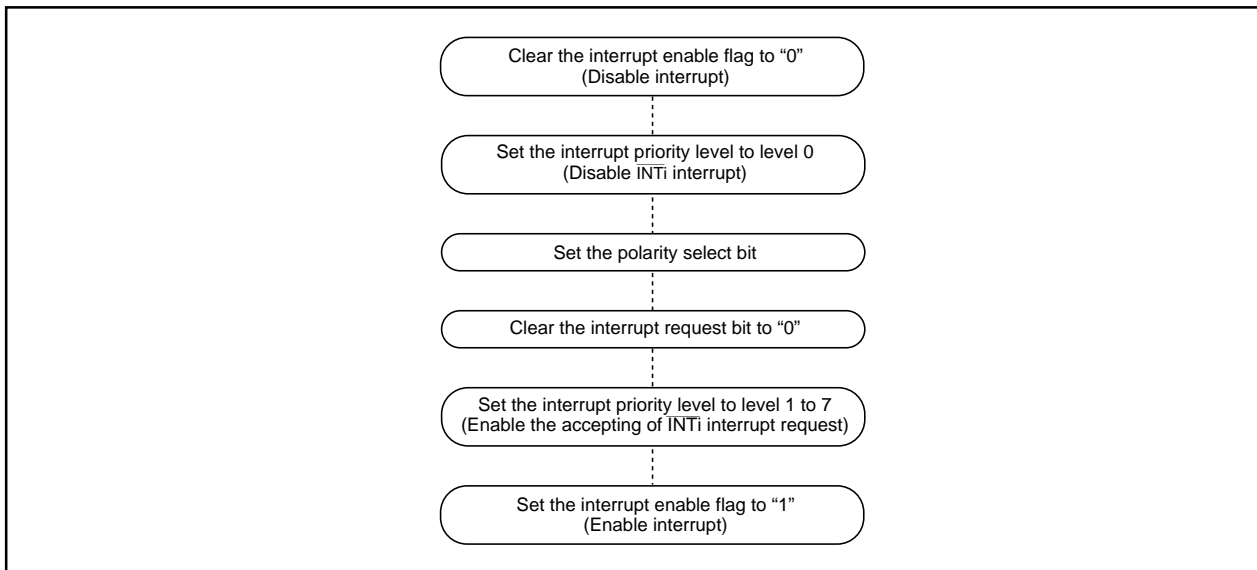


Figure 2.7.14 Switching condition of INT interrupt request

(4) Rewrite 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

2.8 Watchdog Timer

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

With XIN chosen for BCLK

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

With XCIN chosen for BCLK

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

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

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

Figure 2.8.1 shows the block diagram of the watchdog timer. Figure 2.8.2 shows the watchdog timer control register and Figure 2.8.3 shows the watchdog timer start register.

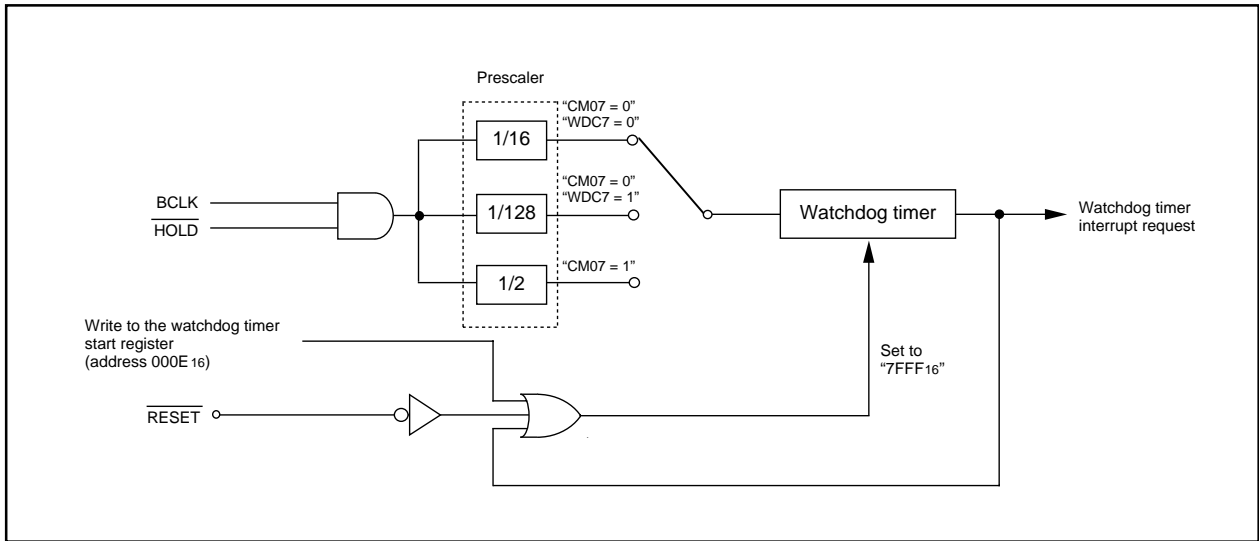


Figure 2.8.1 Block diagram of watchdog timer

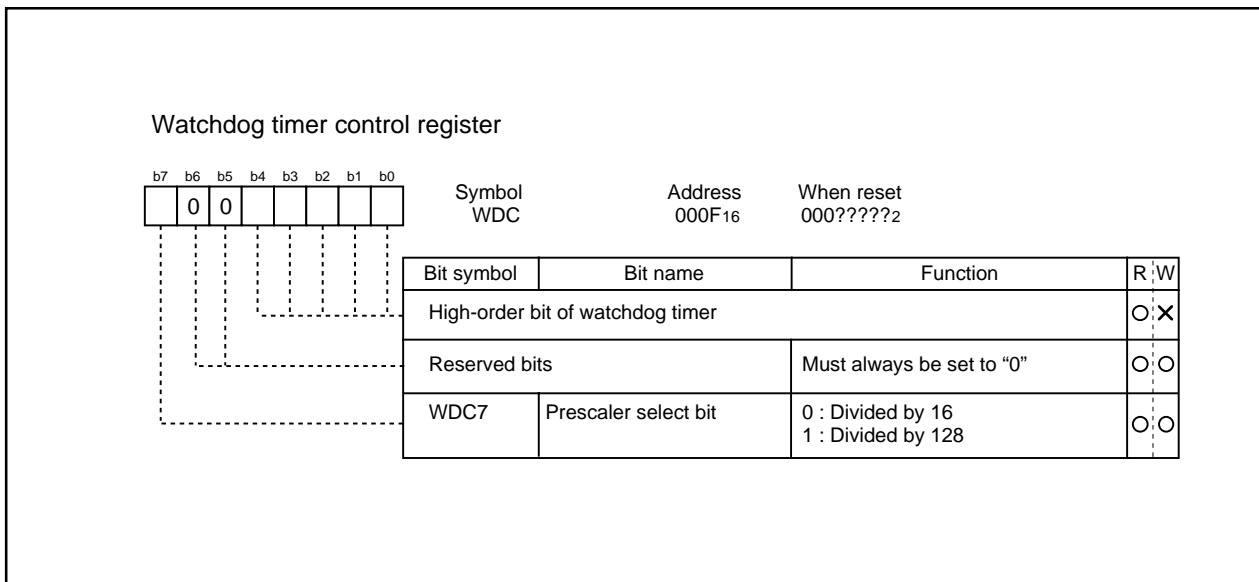


Figure 2.8.2 Watchdog timer control register

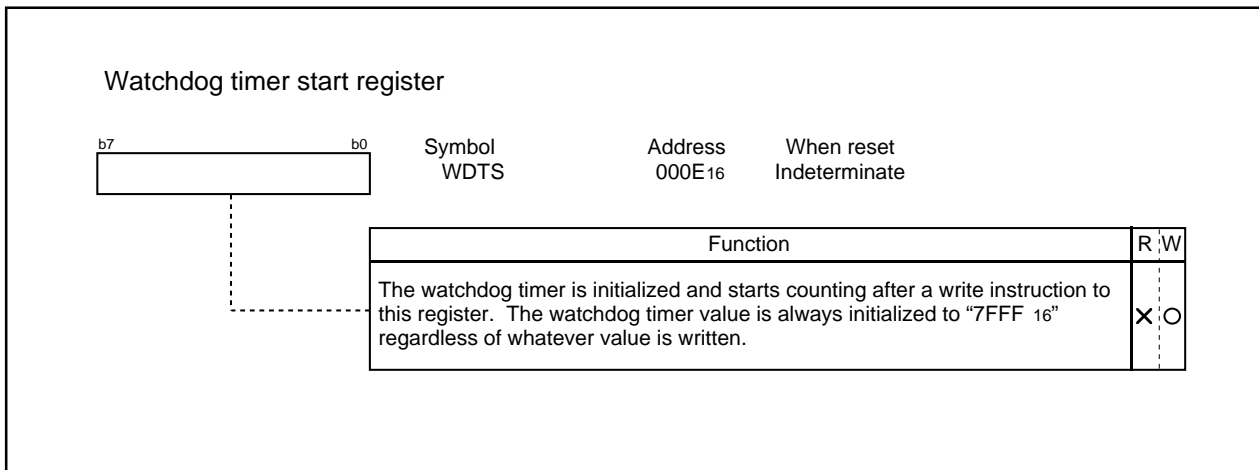


Figure 2.8.3 Watchdog timer start register

2.9 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 2.9.1 shows the block diagram of the DMAC. Table 2.9.1 shows the DMAC specifications. Figures 2.9.2 to 2.9.7 show the registers used by the DMAC.

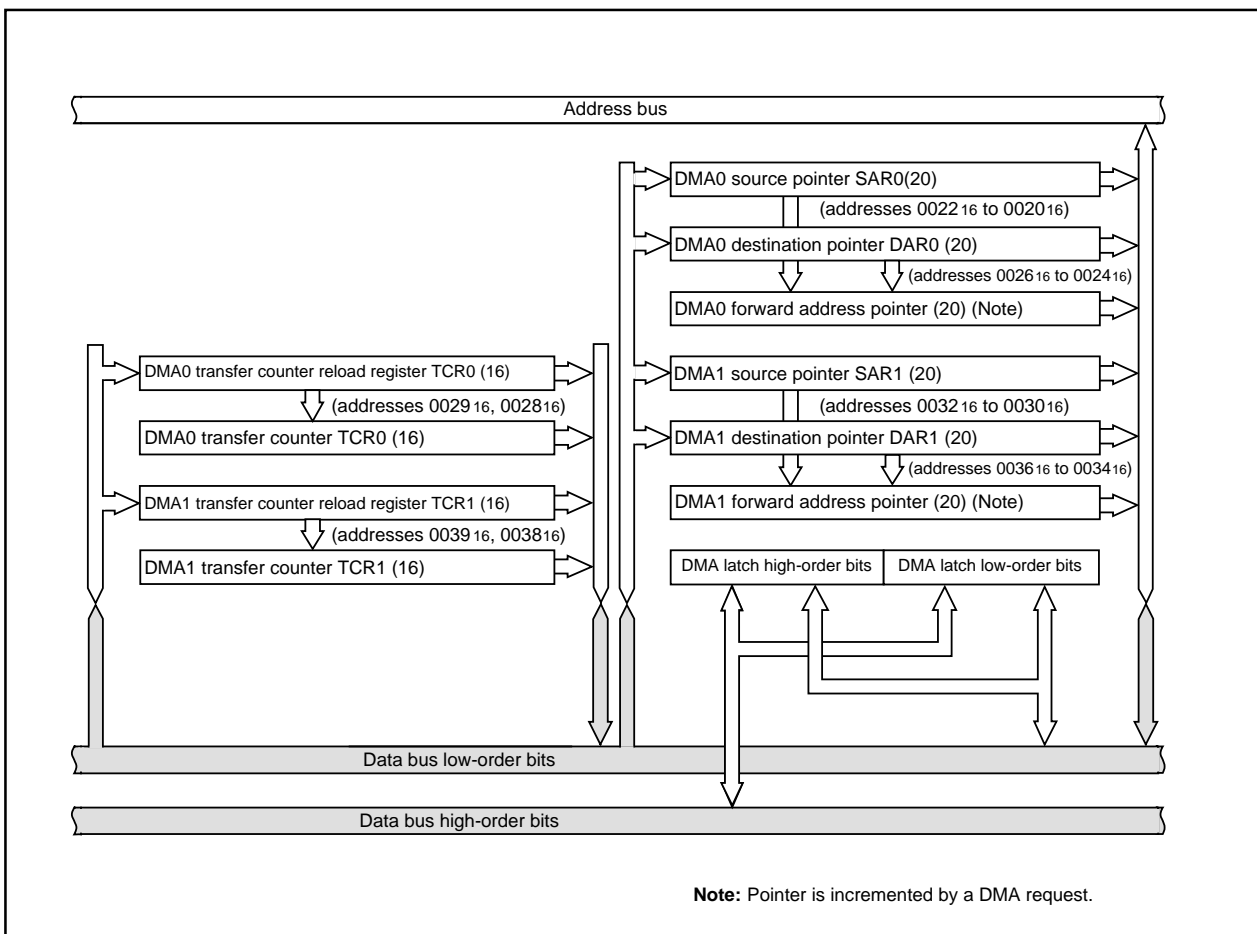


Figure 2.9.1 Block diagram of DMAC

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 2.9.1 DMAC specifications

| Item | Specification |
|--|--|
| No. of channels | 2 (cycle steal method) |
| Transfer memory space | <ul style="list-style-type: none"> From any address in the 1M bytes space to a fixed address From a fixed address to any address in the 1M bytes space From a fixed address to a fixed address (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 or both edge of pin \overline{INT}_0 Falling edge of pin \overline{INT}_1 Timer A0 to timer A4 interrupt requests Timer B0 to timer B2 interrupt requests UART0 transmission and reception interrupt requests UART2 transmission and reception interrupt requests Multi-master I ² C-BUS interface 0 interrupt request Multi-master I ² C-BUS interface 1 interrupt request A-D conversion interrupt request OSD1 and OSD2 interrupt requests Data slicer interrupt request V _{SYNC} interrupt request Software triggers |
| Channel priority | DMA0 takes precedence if DMA0 and DMA1 requests are generated simultaneously |
| Transfer unit | 8 bits or 16 bits |
| Transfer address direction | forward/fixed (forward direction cannot be specified for both source and destination simultaneously) |
| Transfer mode | <ul style="list-style-type: none"> Single transfer mode After the transfer counter underflows, the DMA enable bit turns to "0", and the DMAC turns inactive Repeat transfer mode After the transfer counter underflows, the value of the transfer counter reload register is reloaded to the transfer counter. The DMAC remains active unless a "0" is written to the DMA enable bit. |
| DMA interrupt request generation timing | When an underflow occurs in the transfer counter |
| Active | When the DMA enable bit is set to "1", the DMAC is active. When the DMAC is active, data transfer starts every time a DMA transfer request signal occurs. |
| Inactive | <ul style="list-style-type: none"> When the DMA enable bit is set to "0", the DMAC is inactive. 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. Registers specified for fixed address transfer are write-enabled when the DMA enable bit is "0". |
| Reading the register | Can be read at any time. However, when the DMA enable bit is "1", reading the register set up as the forward register is the same as reading the value of the forward address pointer. |

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

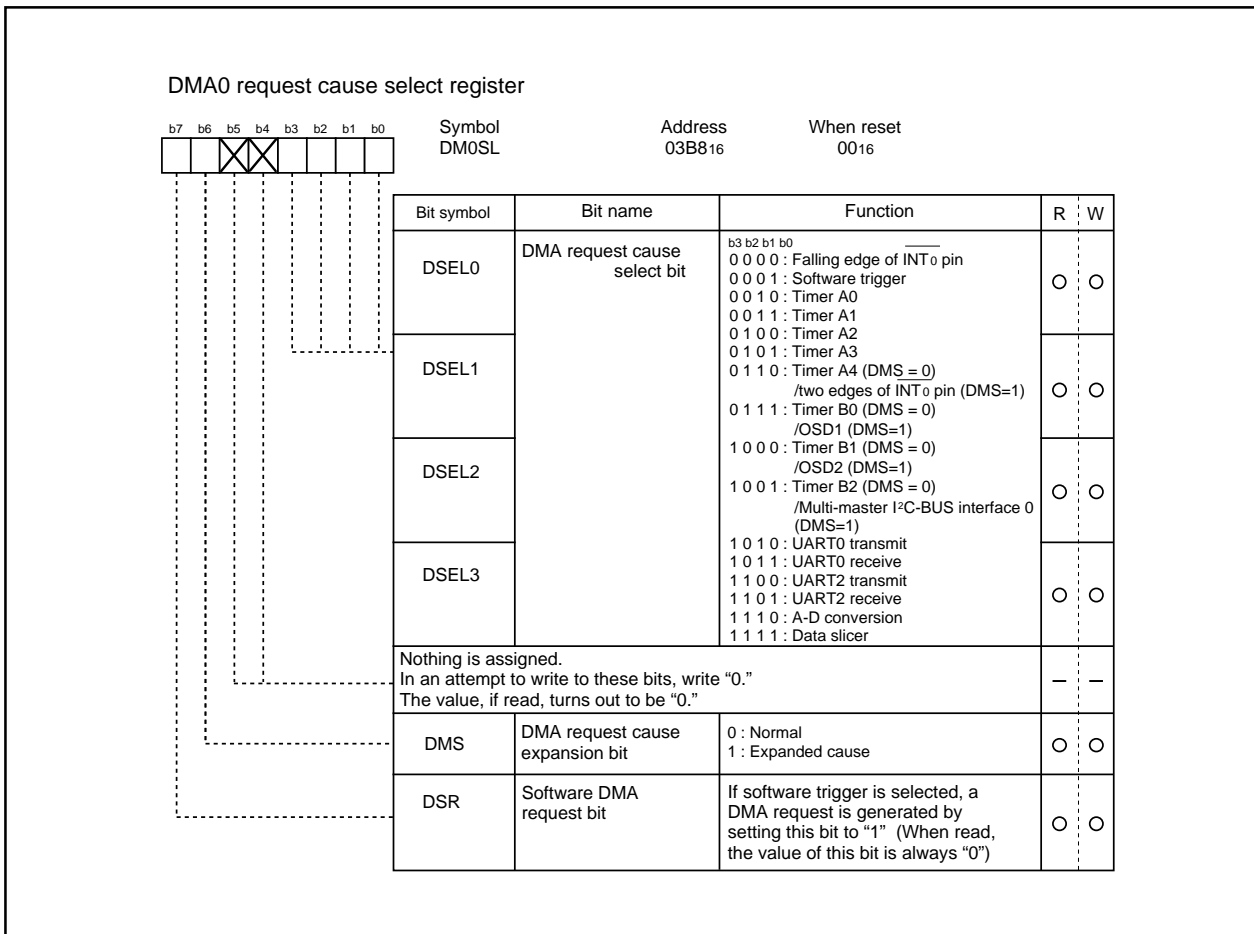


Figure 2.9.2 DMA0 request cause select register

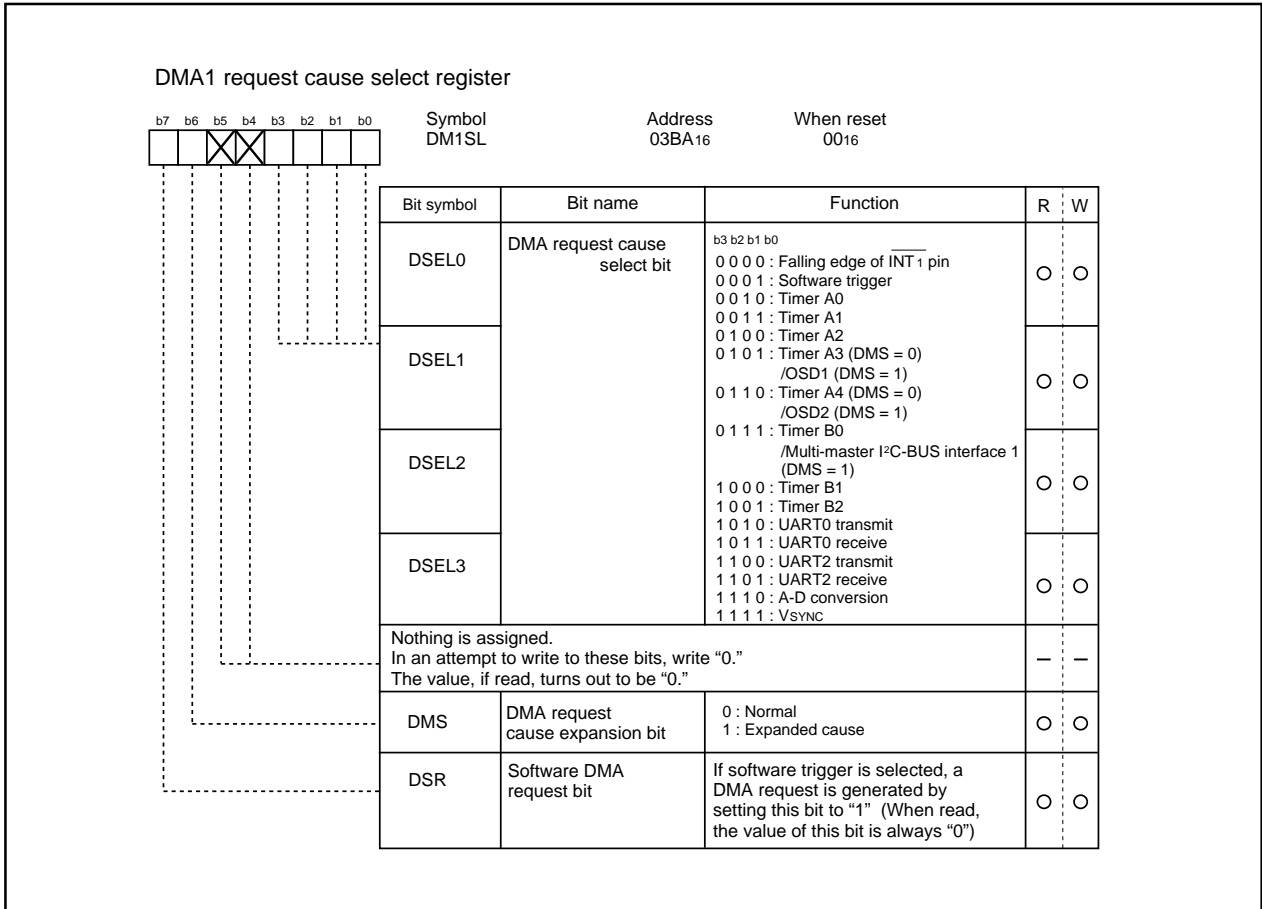


Figure 2.9.3 DMA1 request cause select register

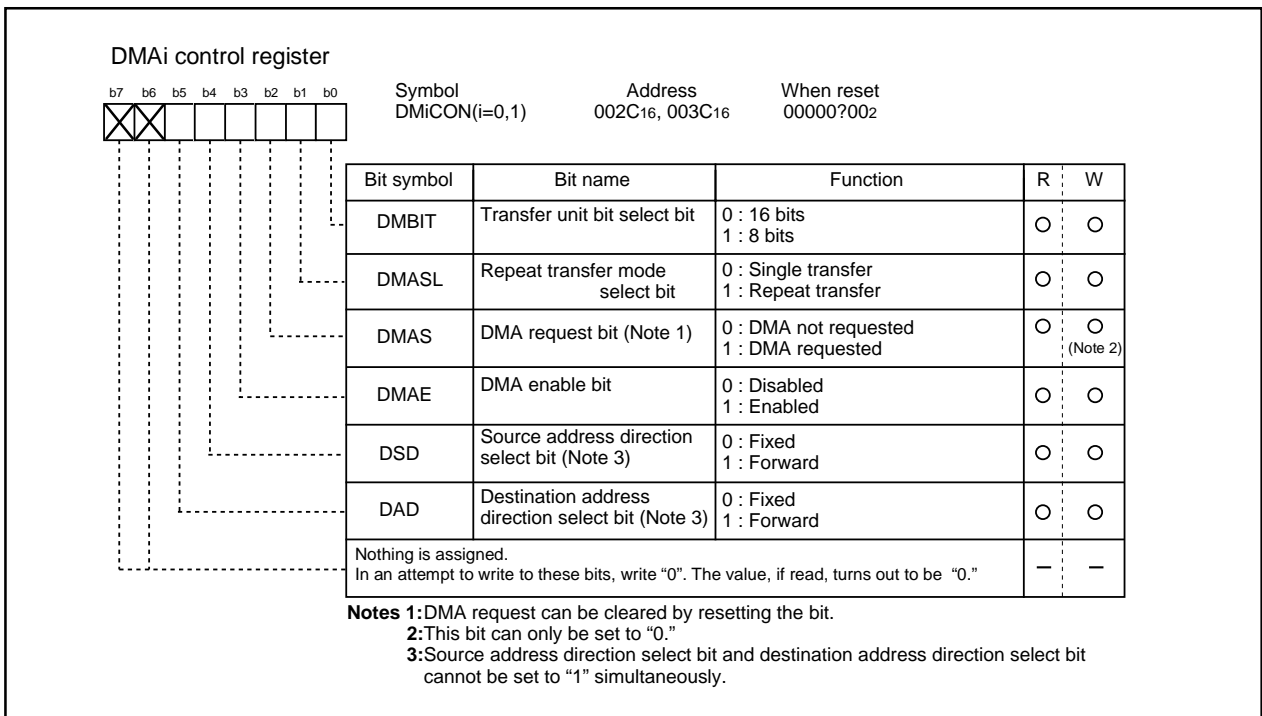


Figure 2.9.4 DMAi control register (i = 0, 1)

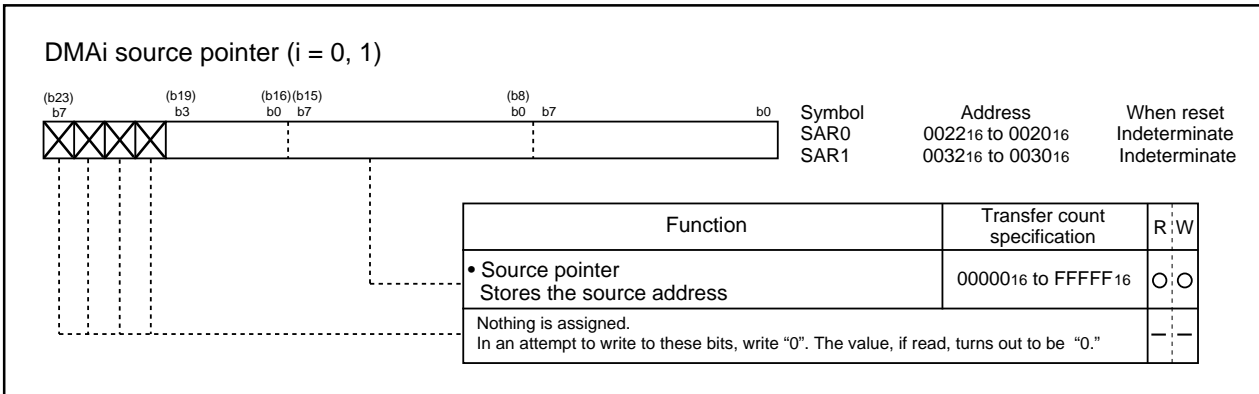


Figure 2.9.5 DMA_i source pointer (i = 0, 1)

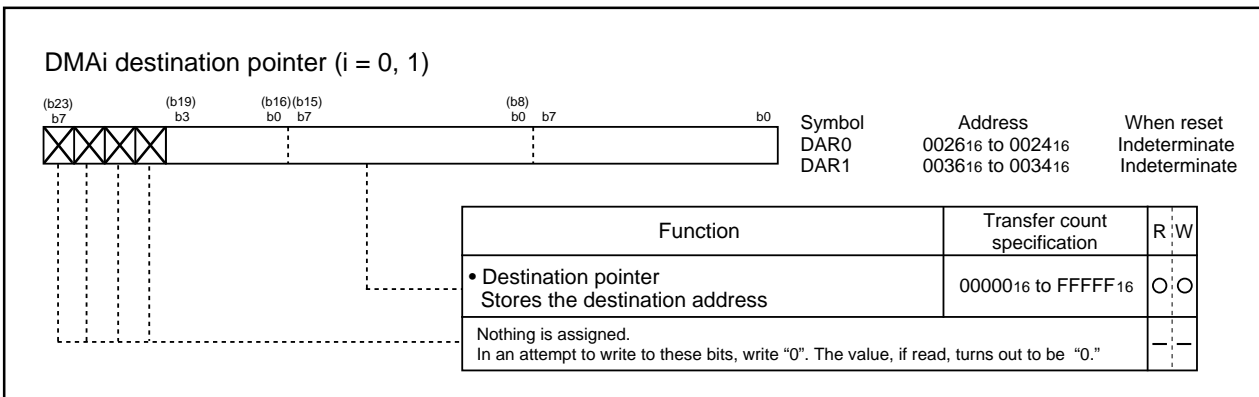


Figure 2.9.6 DMA_i destination pointer (i = 0, 1)

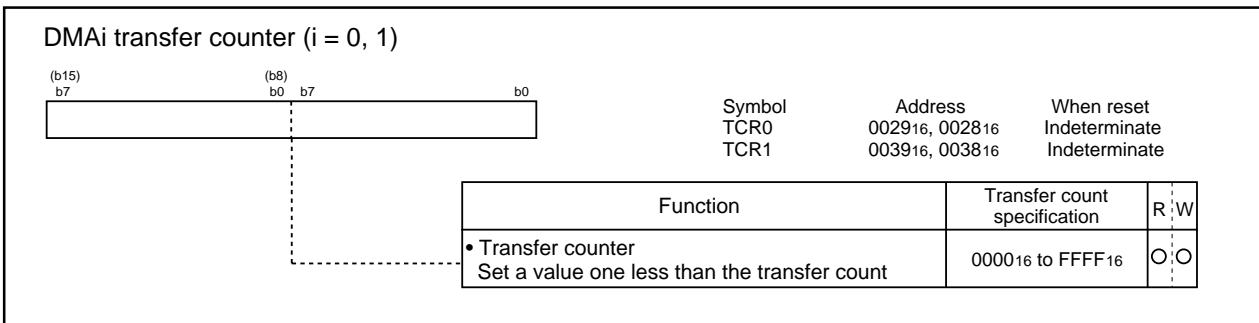


Figure 2.9.7 DMA_i transfer counter (i = 0, 1)

2.9.1 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.

(1) 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.

(2) 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.

(3) Effect of software wait

When the SFR area, the OSD RAM 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 2.9.8 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 47, if data is being transferred in 16-bit units on an 8-bit bus, two bus cycles are required for both the source read cycle and the destination write cycle.

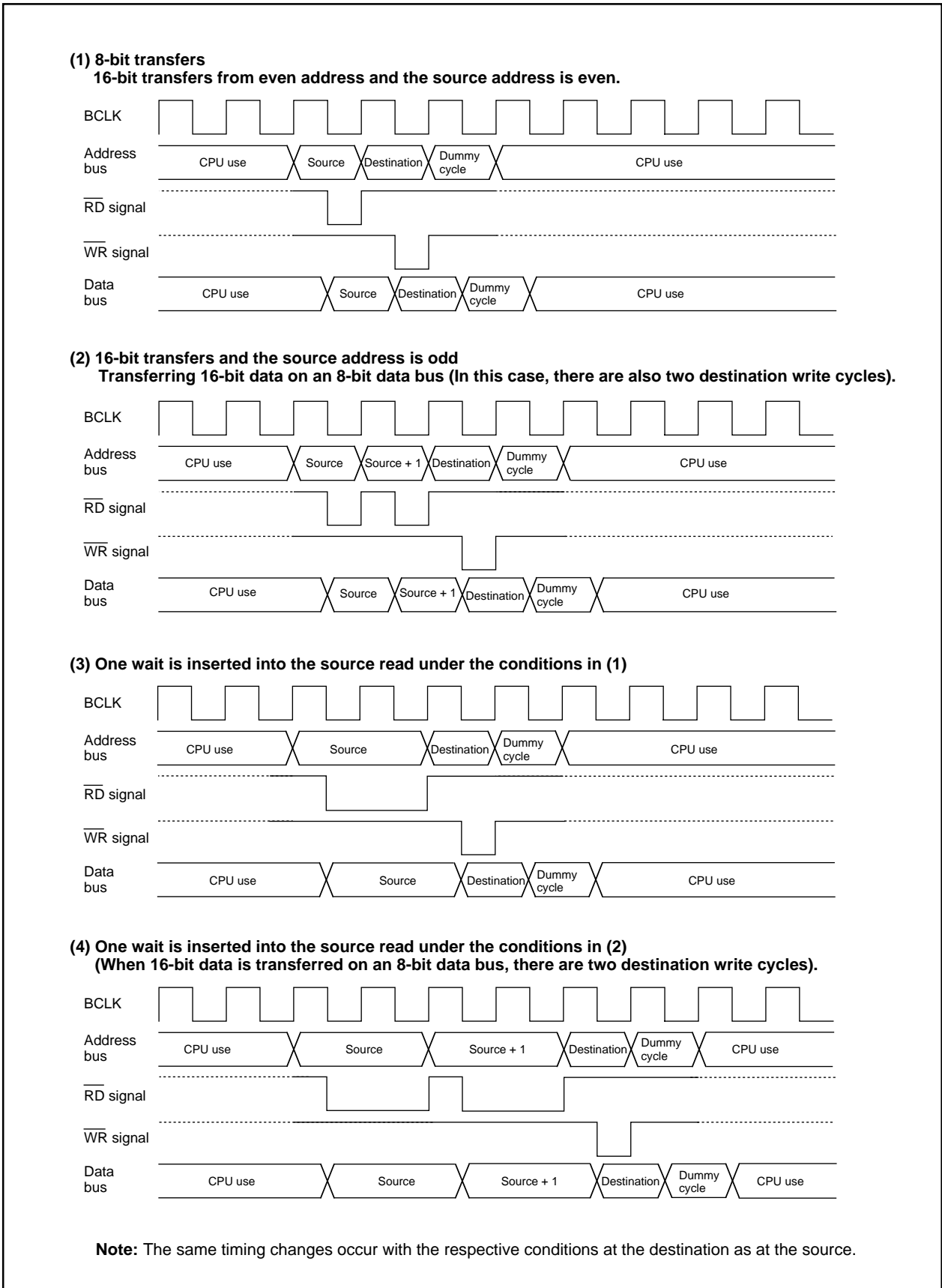


Figure 2.9.8 Example of the transfer cycles for a source read

2.9.2 DMAC Transfer Cycles

Any combination of even or odd transfer read and write addresses is possible. Table 2.9.2 shows the number of DMAC transfer cycles.

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

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

Table 2.9.2 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 | Internal ROM/RAM | SFR area /OSD RAM | Separate bus | Separate bus | Multiplex bus |
| No wait | With wait | | No wait | With wait | |
| 1 | 2 | 2 | 1 | 2 | 3 |

2.9.3 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.

2.9.4 DMA Request Bit

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

DMA request factors include the following.

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

* External factors effected by utilizing the input from external interrupt signals.

For the selection of DMA request factors, see the descriptions of the DMA_i 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 $\overline{\text{INT}}_i$ pin (i depends on which DMAC channel is used).

Selecting the $\overline{\text{INT}}_i$ pins as external factors using the DMA request factor selection bit causes input from these pins to become the DMA transfer request sig=ls.

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 $\overline{\text{INT}}_i$ 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. Figure 2.9.9 illustrates these operations.

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.

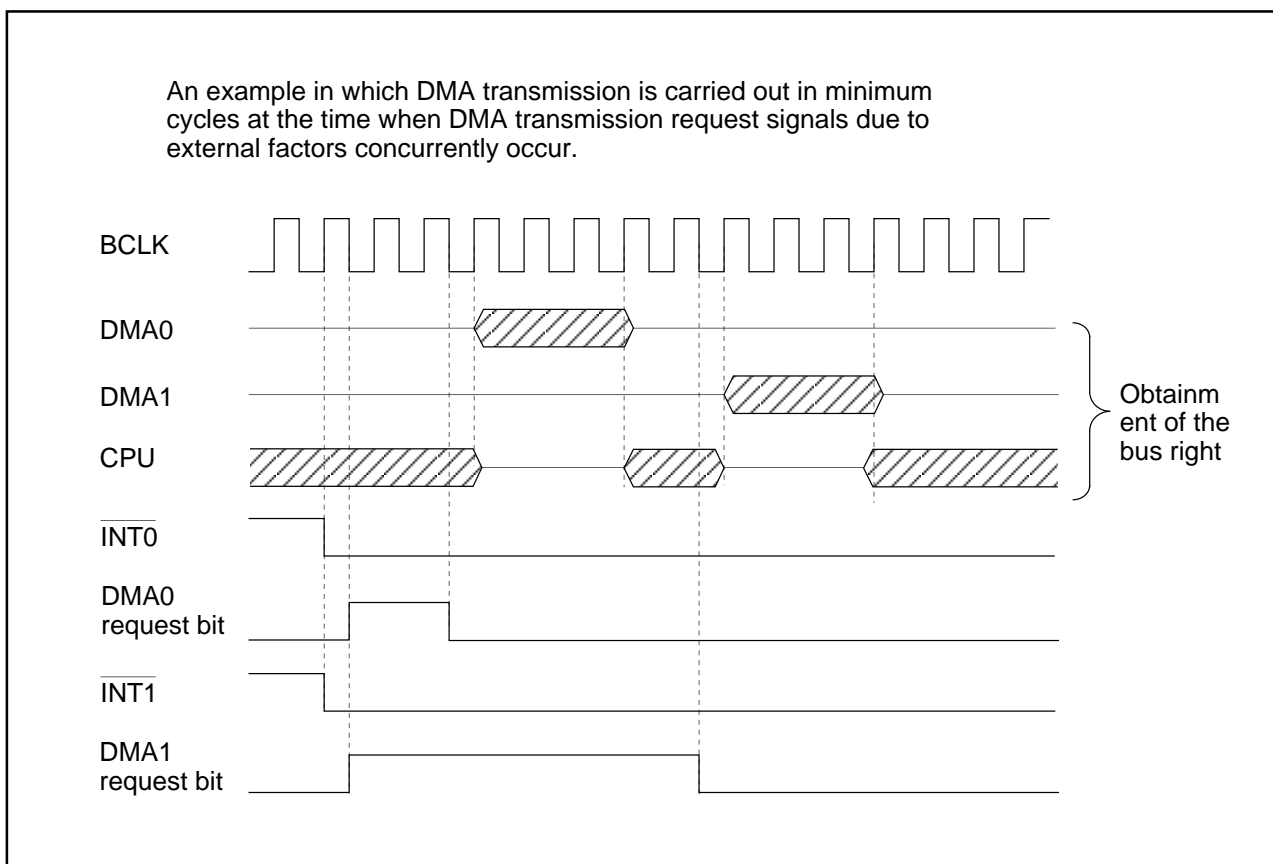


Figure 2.9.9 An example of DMA transfer effected by external factors

2.10 Timer

There are eight 16-bit timers. These timers can be classified by function into timers A (five) and timers B (three). All these timers function independently. Figures 2.10.1 and 2.10.2 show the block diagram of timers.

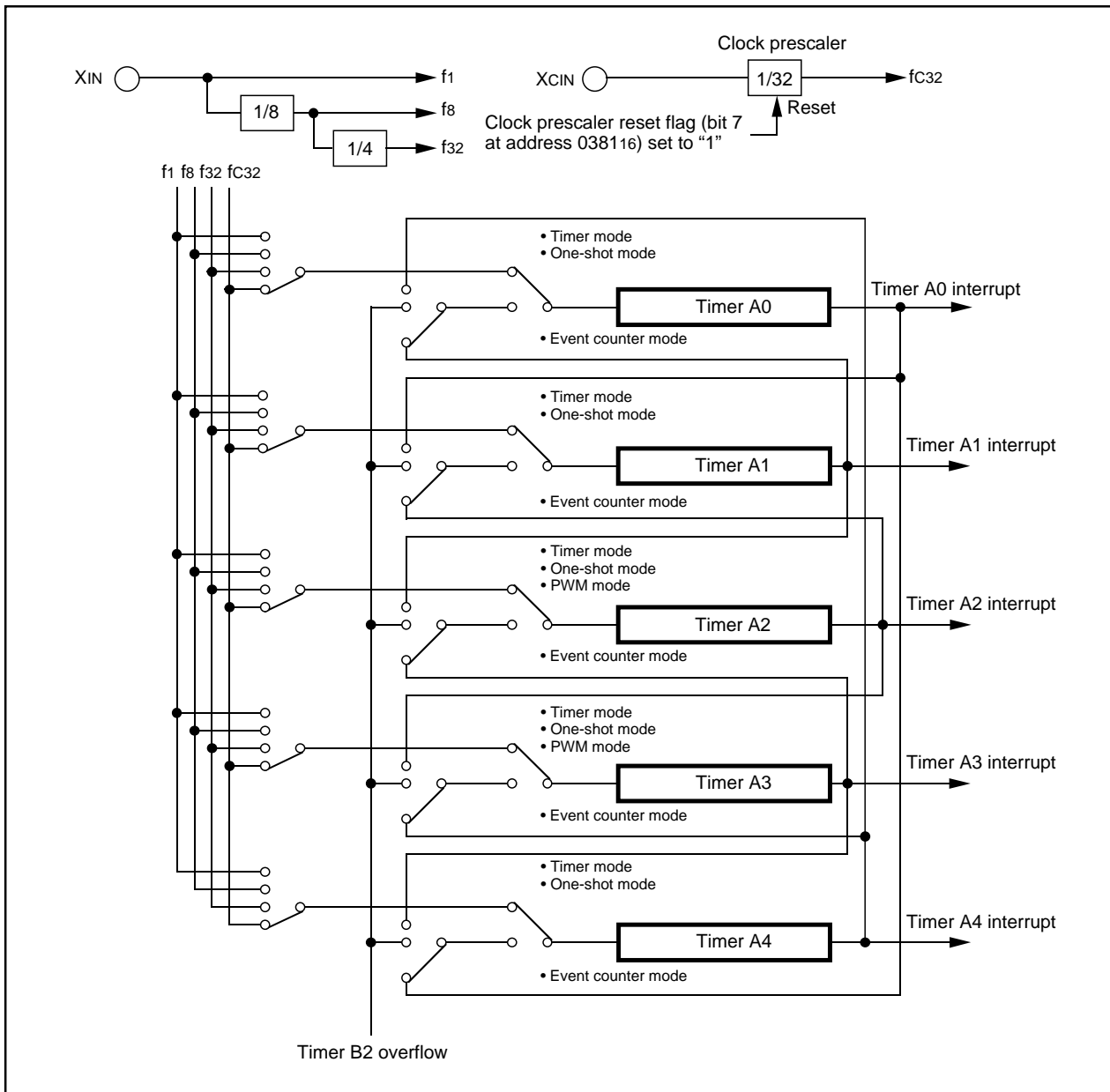


Figure 2.10.1 Timer A block diagram

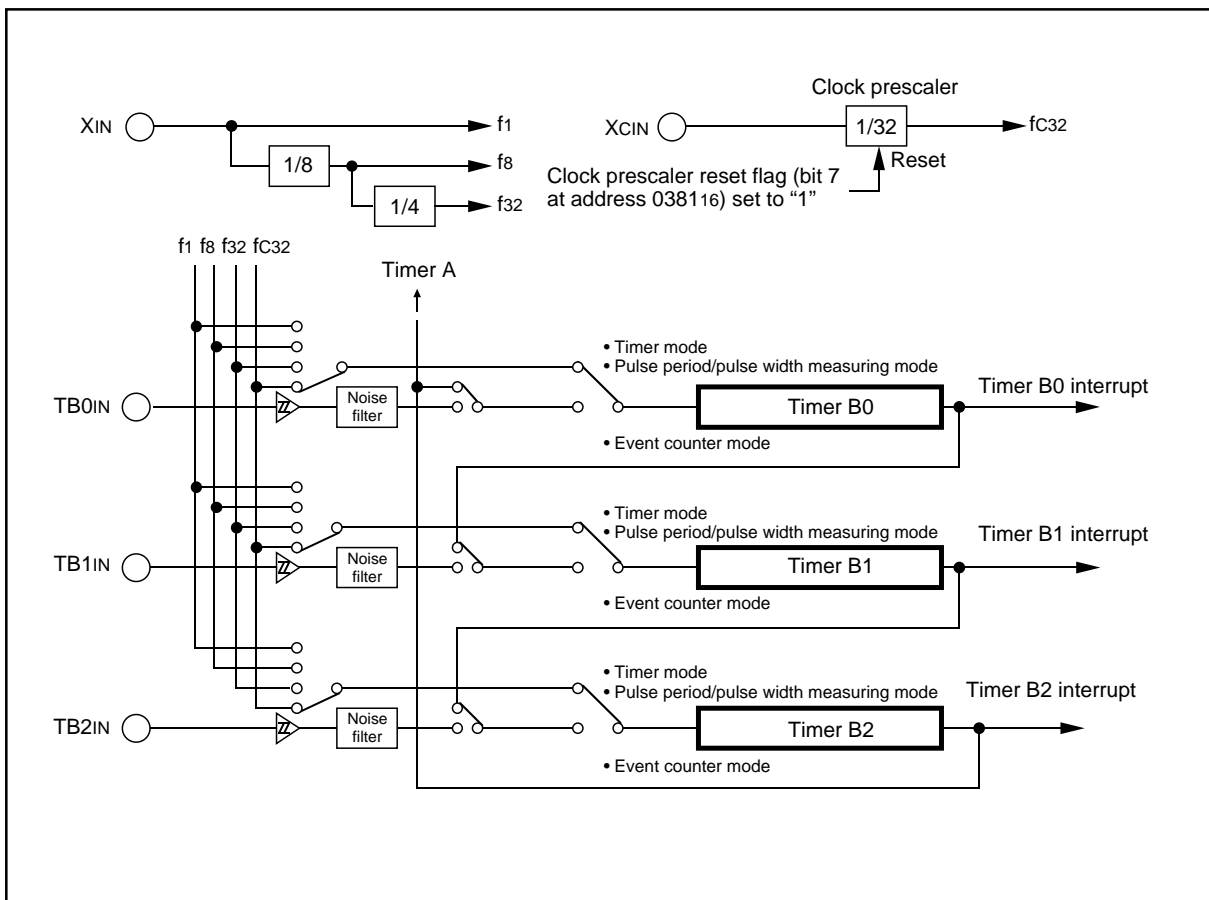


Figure 2.10.2 Timer B block diagram

2.10.1 Timer A

Figure 2.10.3 shows the block diagram of timer A. Figures 2.10.4 to 2.10.10 show the timer A-related registers.

Except the pulse output function, 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 a timer over flow.
- One-shot timer mode: The timer stops counting when the count reaches “0000₁₆”.
- Pulse width modulation (PWM) mode: The timer outputs pulses of a given width.

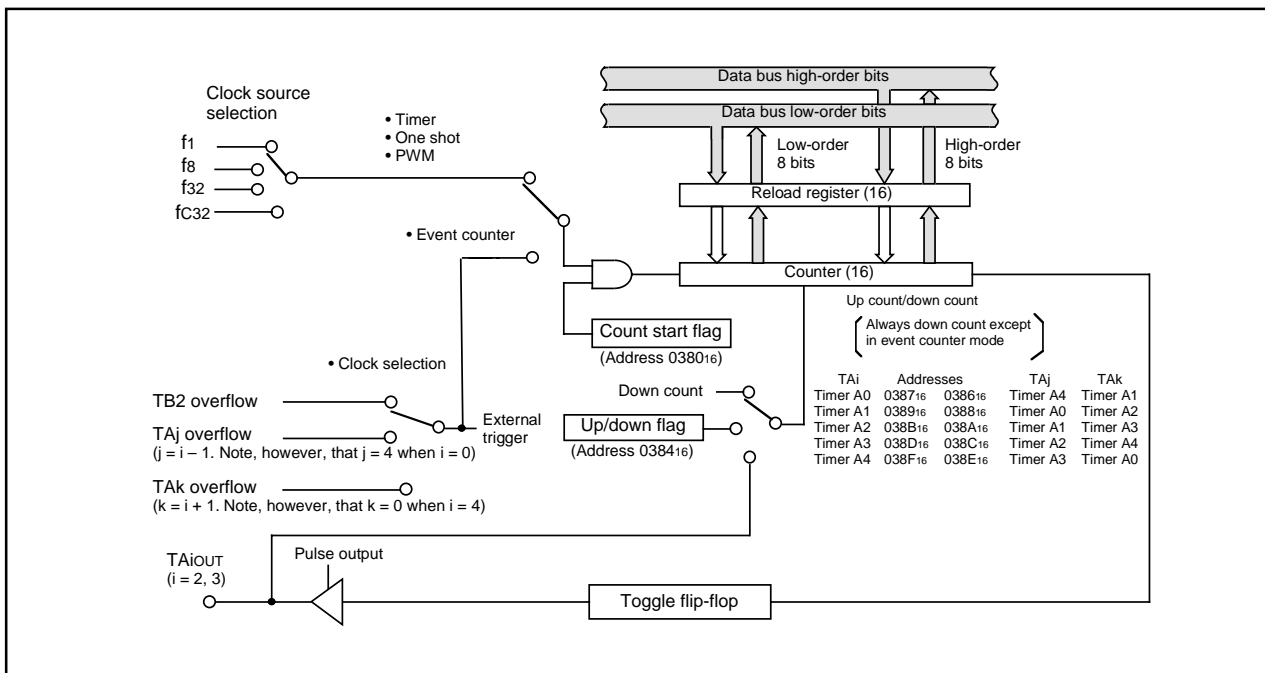


Figure 2.10.3 Block diagram of timer A

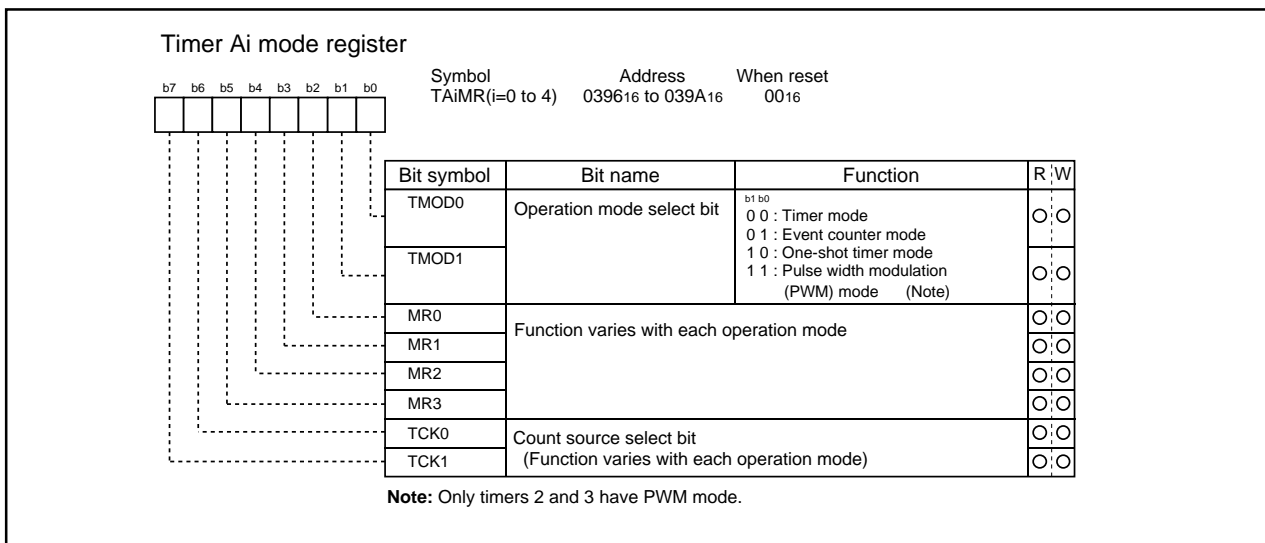


Figure 2.10.4 Timer Ai mode register (i = 0 to 4)

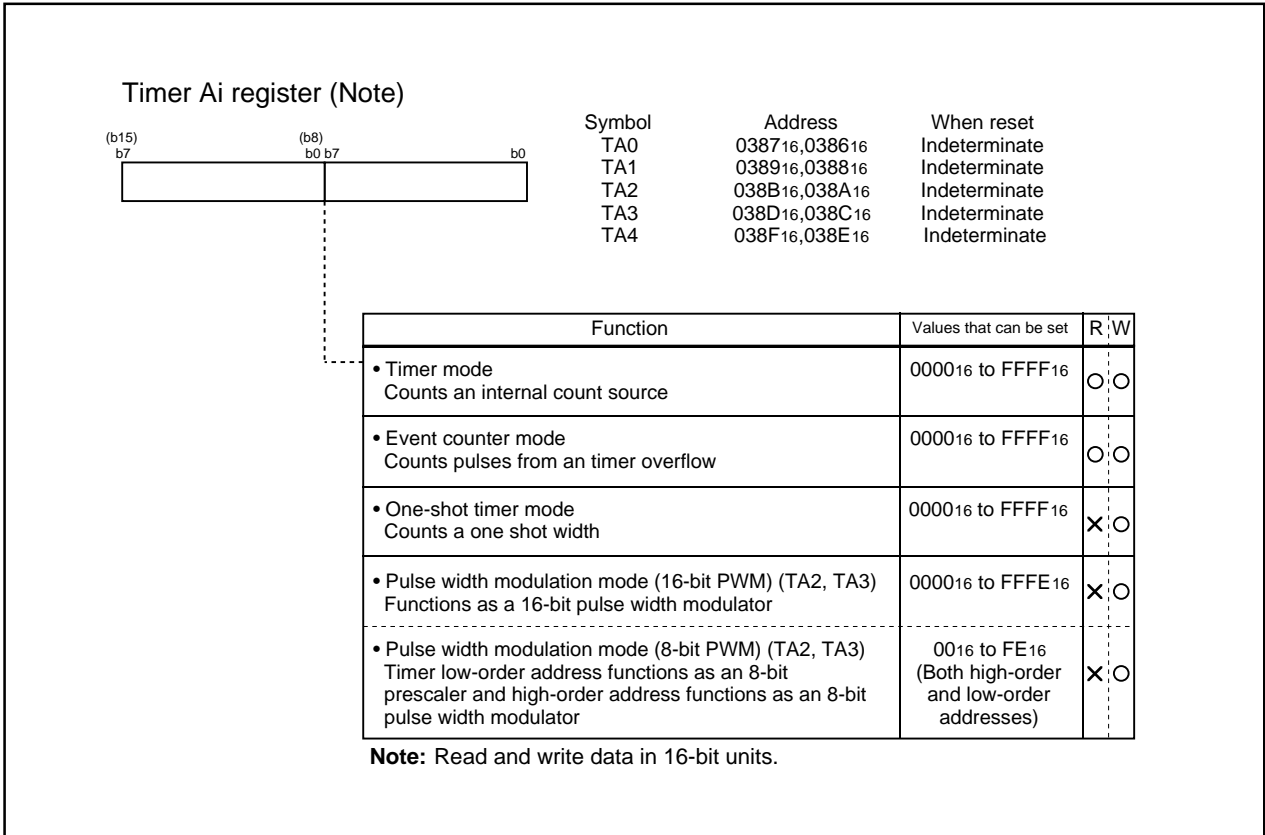


Figure 2.10.5 Timer Ai register (i = 0 to 4)

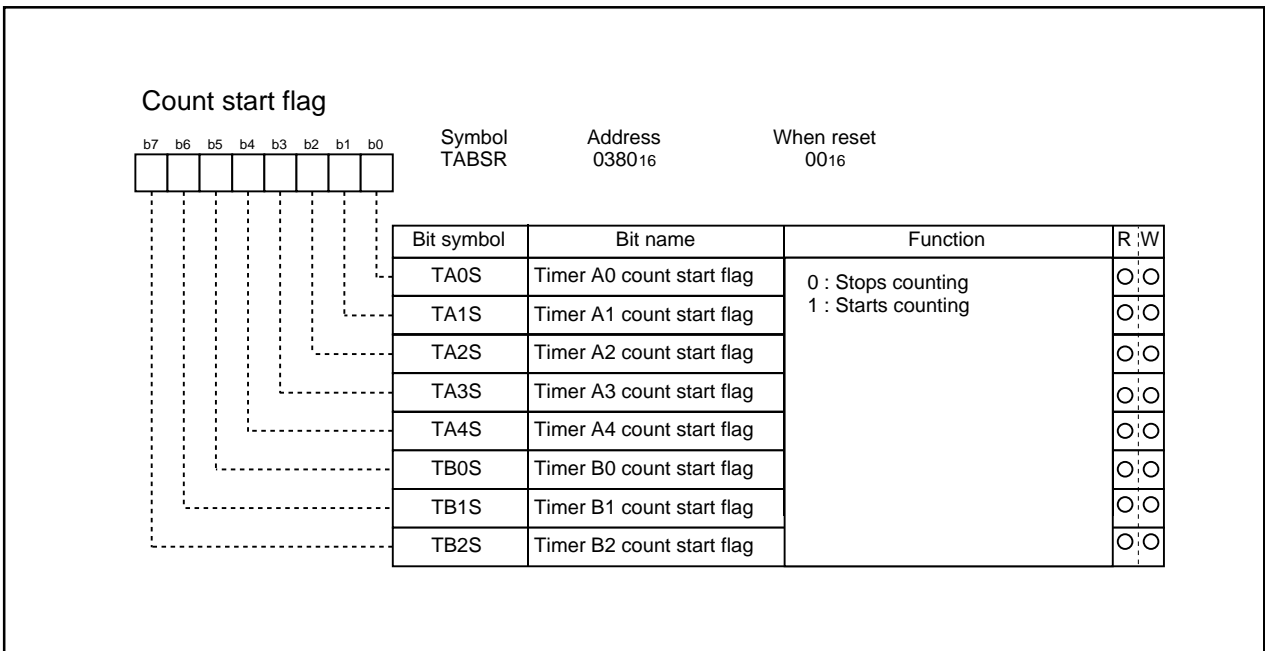


Figure 2.10.6 Count start flag

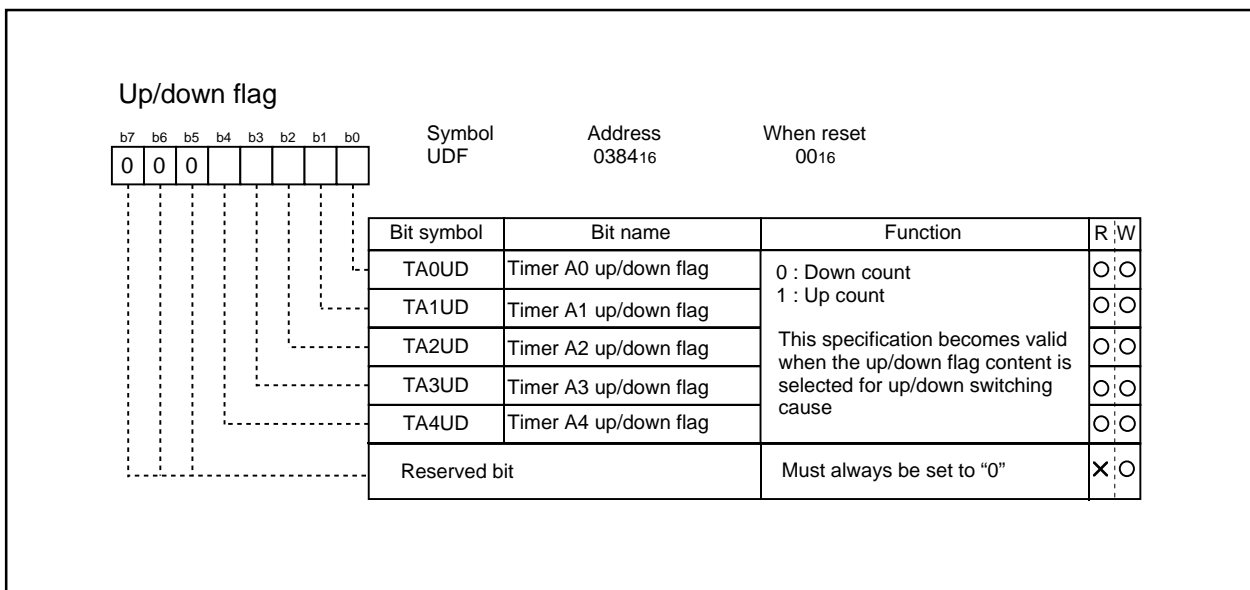


Figure 2.10.7 Up/down flag

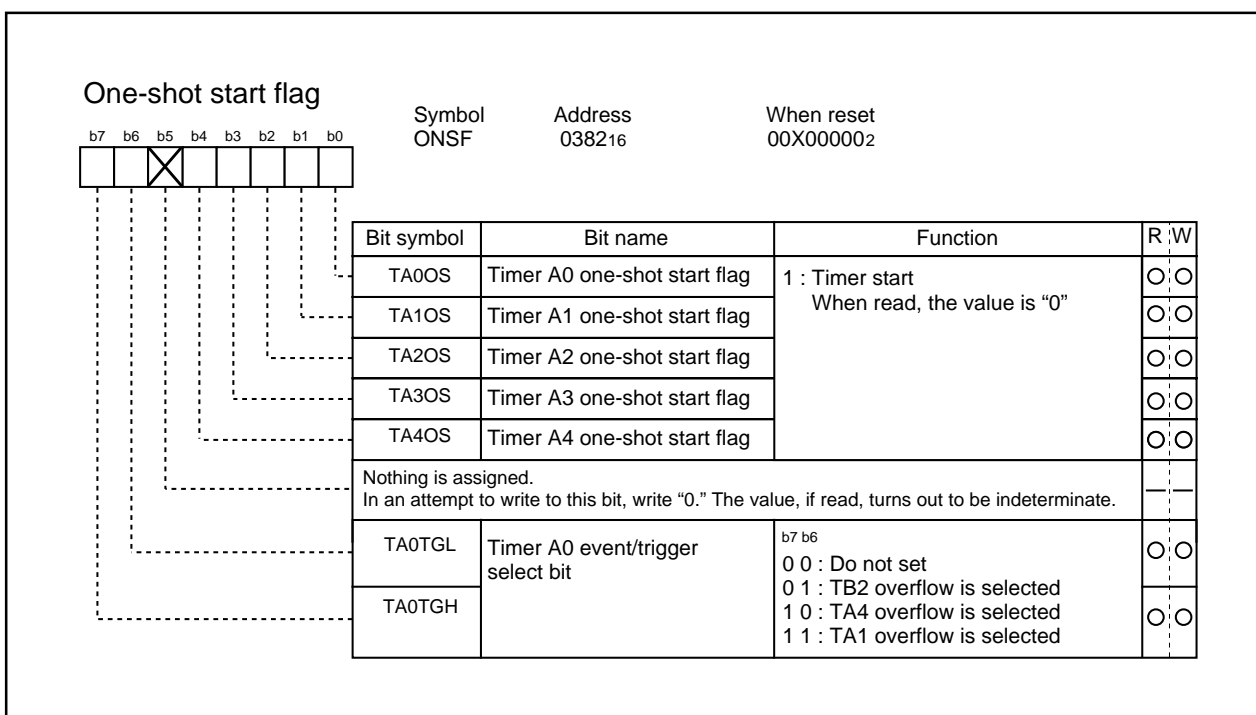


Figure 2.10.8 One-shot start flag

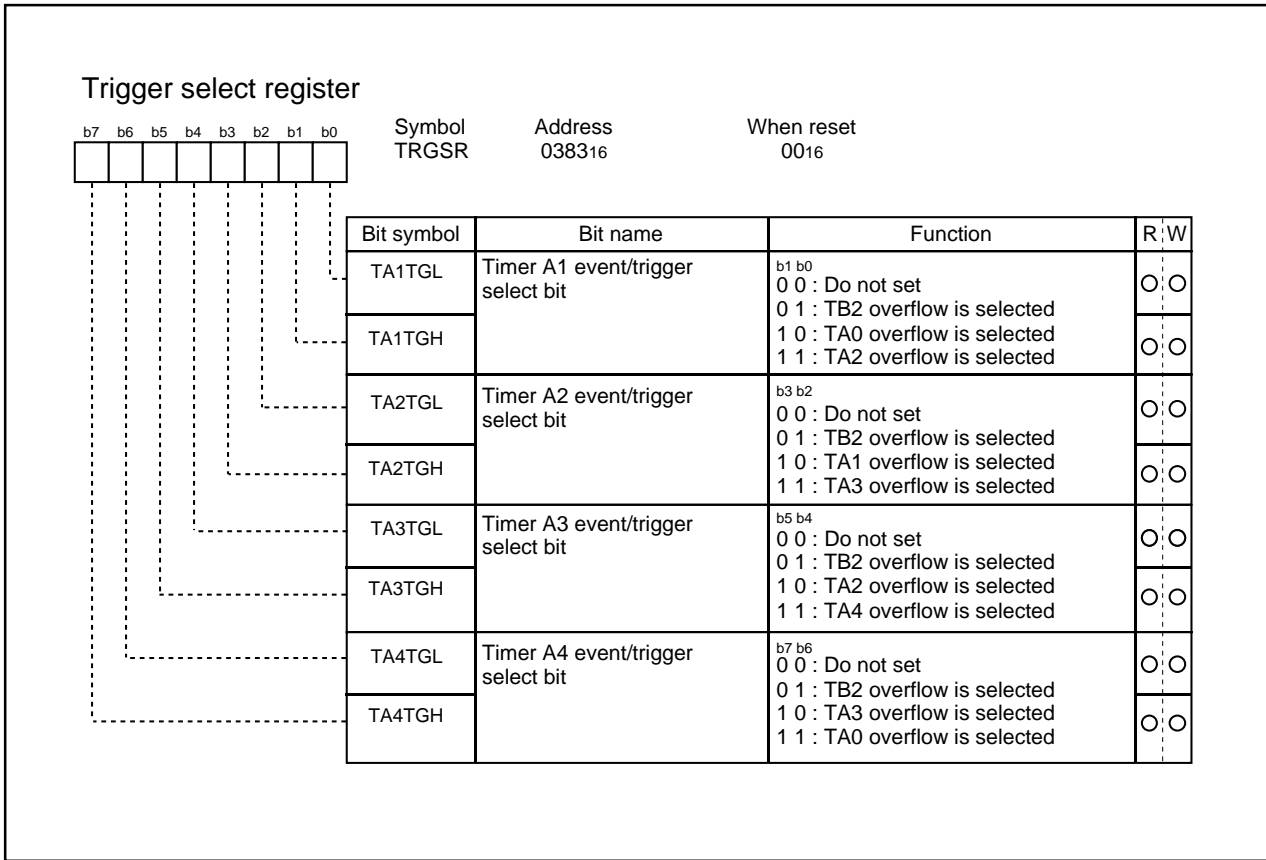


Figure 2.10.9 Trigger select register

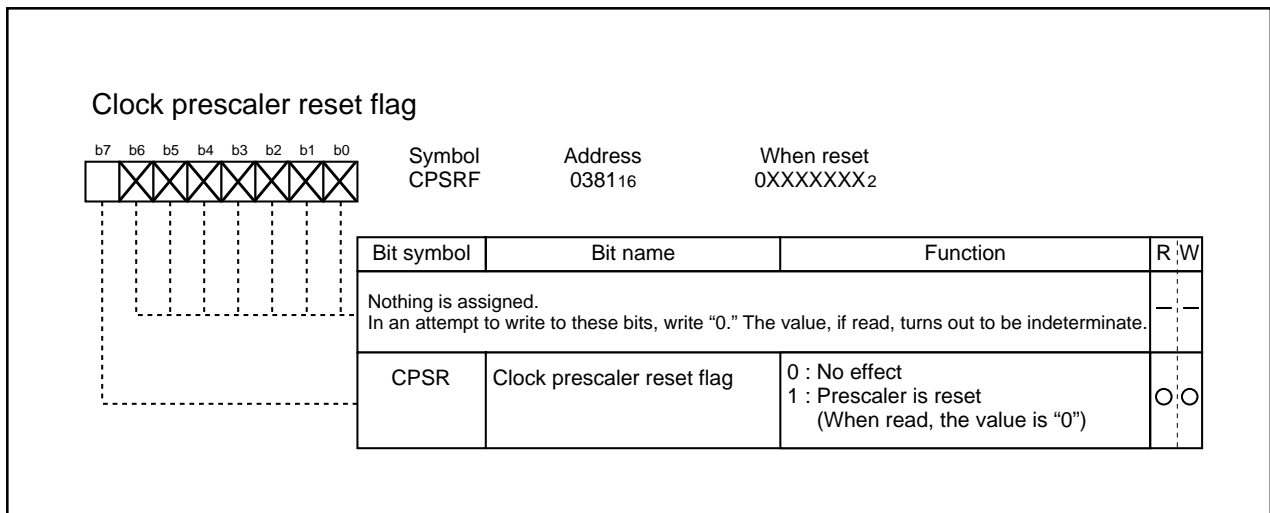


Figure 2.10.10 Clock prescaler reset flag

(1) Timer mode

In this mode, the timer counts an internally generated count source. (See Table 2.10.1.) Figure 2.10.11 shows the timer Ai mode register in timer mode.

Table 2.10.1 Specifications of timer mode

| Item | Specification |
|---|---|
| Count source | f ₁ , f ₈ , f ₃₂ , f _{c32} |
| Count operation | <ul style="list-style-type: none"> Down count 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 |
| TA2 _{out} /TA3 _{out} pin function | Programmable I/O port or pulse output |
| Read from timer | Count value can be read out by reading timer Ai register |
| Write to timer | <ul style="list-style-type: none"> When counting stopped When a value is written to timer Ai register, it is written to both reload register and counter When counting in progress When a value is written to timer Ai register, it is written to only reload register (Transferred to counter at next reload time) |
| Select function | <ul style="list-style-type: none"> Pulse output function Each time the timer underflows, the TA_{iout} pin's polarity is reversed |

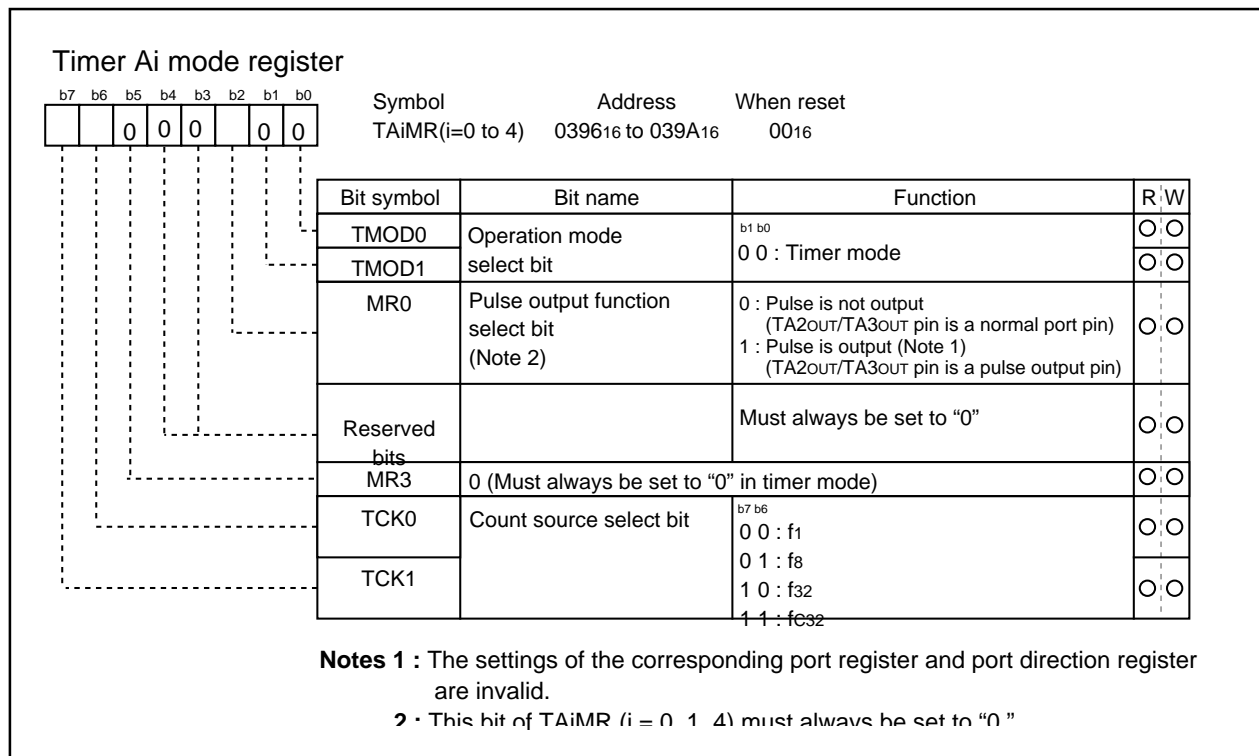


Figure 2.10.11 Timer Ai mode register in timer mode (i = 0 to 4)

(2) Event counter mode

In this mode, the timer counts an internal timer's overflow.

Table 2.10.2 Timer specifications in event counter mode

| Item | Specification |
|-------------------------------------|---|
| Count source | <ul style="list-style-type: none"> • TB2 overflow, TAj overflow, TAk overflow |
| Count operation | <ul style="list-style-type: none"> • Up count or down count can be selected by external signal or software • When the timer overflows or underflows, it reloads the reload register contents before continuing counting (Note) |
| Divide ratio | $1 / (FFFF_{16} - n + 1)$ for up count $1 / (n + 1)$ for down count n : Set value |
| Count start condition | Count start flag is set (= 1) |
| Count stop condition | Count start flag is reset (= 0) |
| Interrupt request generation timing | The timer overflows or underflows |
| TA2OUT/TA3OUT pin function | Programmable I/O port, pulse output, or up/down count select input |
| Read from timer | Count value can be read out by reading timer Ai register |
| Write to timer | <ul style="list-style-type: none"> • When counting stopped When a value is written to timer Ai register, it is written to both reload register and counter • When counting in progress When a value is written to timer Ai register, it is written to only reload register (Transferred to counter at next reload time) |
| Select function | <ul style="list-style-type: none"> • Free-run count function Even when the timer overflows or underflows, the reload register content is not reloaded to it • Pulse output function 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.

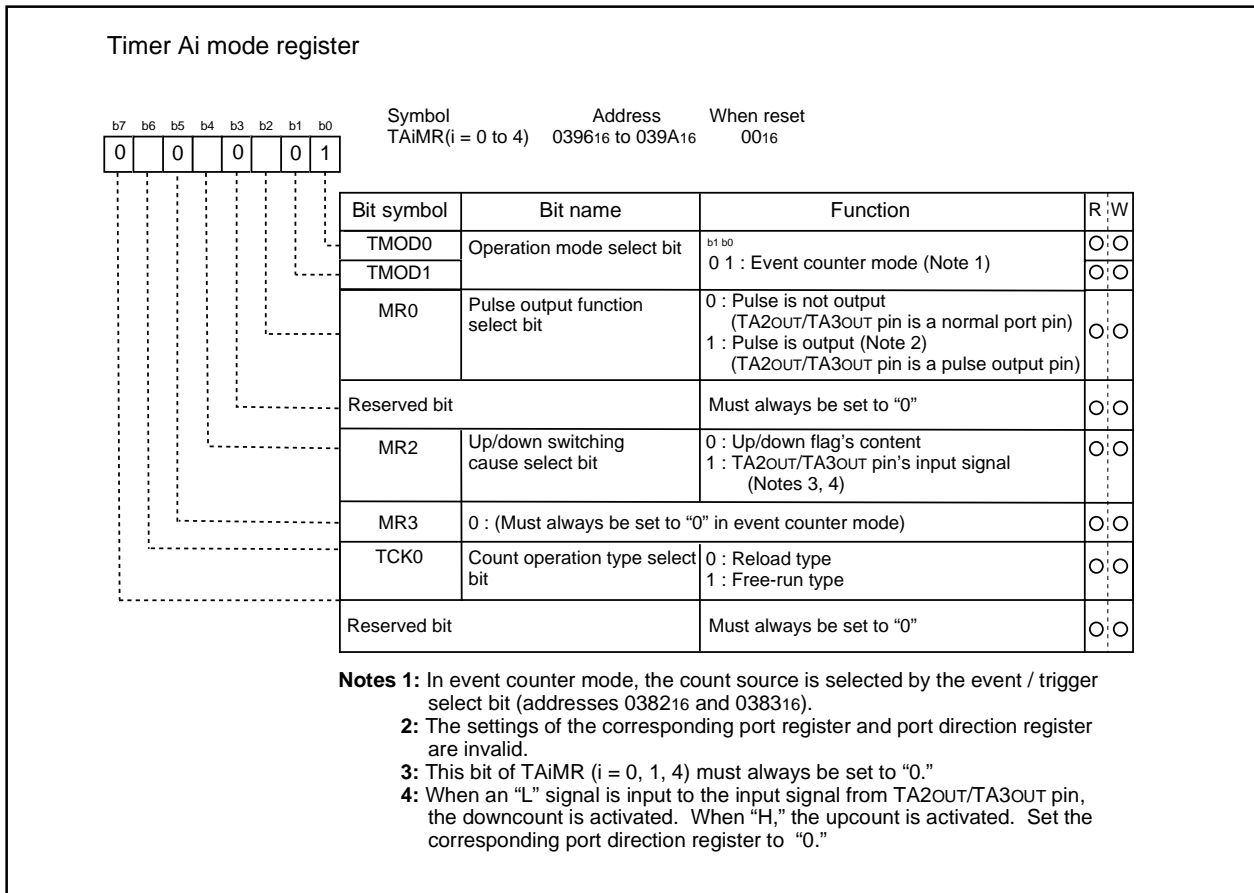


Figure 2.10.12 Timer Ai mode register in event counter mode (i = 0 to 4)

(3) One-shot timer mode

In this mode, the timer operates only once. (See Table 2.10.3.) When a trigger occurs, the timer starts up and continues operating for a given period. Figure 2.10.13 shows the timer Ai mode register in one-shot timer mode.

Table 2.10.3 Timer specifications in one-shot timer mode

| Item | Specification |
|-------------------------------------|---|
| Count source | f1, f8, f32, fc32 |
| Count operation | <ul style="list-style-type: none"> The timer counts down When the count reaches 0000₁₆, the timer stops counting after reloading a new count 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 | <ul style="list-style-type: none"> The timer overflows The one-shot start flag is set (= 1) |
| Count stop condition | <ul style="list-style-type: none"> A new count is reloaded after the count has reached 0000₁₆ The count start flag is reset (= 0) |
| Interrupt request generation timing | The count reaches 0000 ₁₆ |
| TA2OUT/TA3OUT pin function | Programmable I/O port or pulse output |
| Read from timer | When timer Ai register is read, it indicates an indeterminate value |
| Write to timer | <ul style="list-style-type: none"> When counting stopped When a value is written to timer Ai register, it is written to both reload register and counter When counting in progress When a value is written to timer Ai register, it is written to only reload register (Transferred to counter at next reload time) |

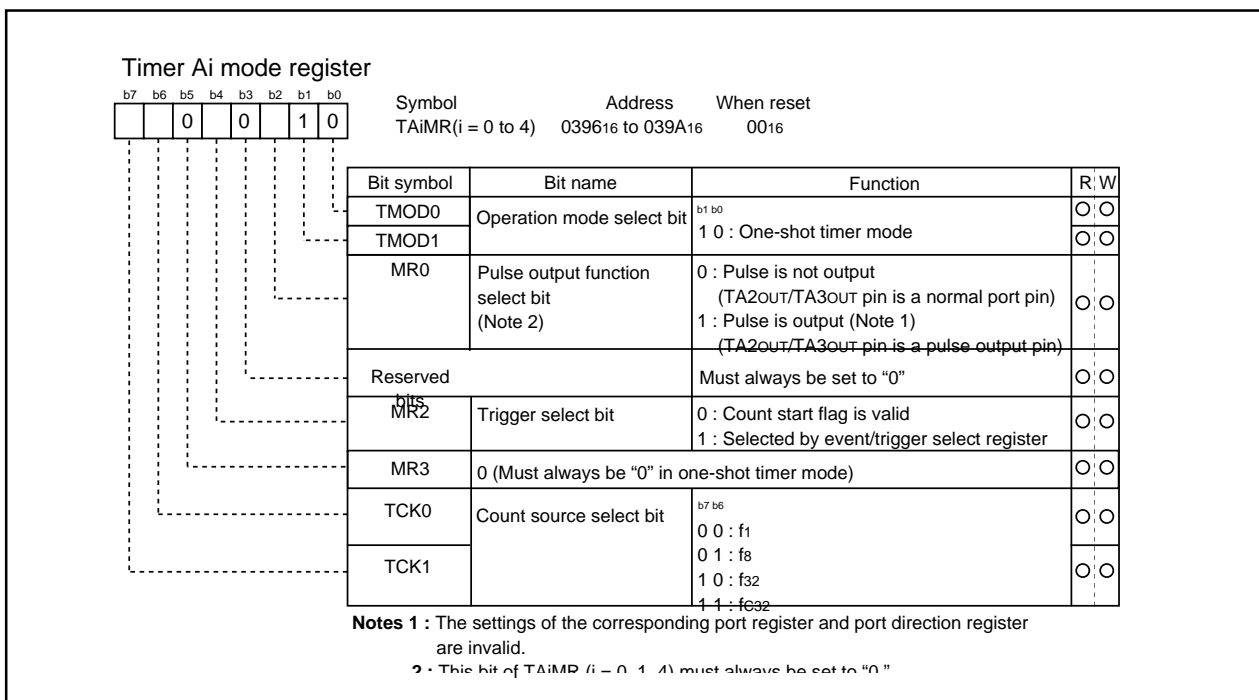


Figure 2.10.13 Timer Ai mode register in one-shot timer mode (i = 0 to 4)

(4) Pulse width modulation (PWM) mode

In this mode, the timer outputs pulses of a given width in succession. (See Table 2.10.4.) In this mode, the counter functions as either a 16-bit pulse width modulator or an 8-bit pulse width modulator. Figure 2.10.14 shows the timer Ai mode register in pulse width modulation mode. Figure 2.10.15 shows the example of how an 8-bit pulse width modulator operates.

Table 2.10.4 Timer specifications in pulse width modulation mode

| Item | Specification |
|-------------------------------------|---|
| Count source | f1, f8, f32, fC32 |
| Count operation | <ul style="list-style-type: none"> The timer counts down (operating as an 8-bit or a 16-bit pulse width modulator) The timer reloads a new count at a rising edge of PWM pulse and continues counting The timer is not affected by a trigger that occurs when counting |
| 16-bit PWM | <ul style="list-style-type: none"> High level width n / f_i n : Set value Cycle time $(2^{16}-1) / f_i$ fixed |
| 8-bit PWM | <ul style="list-style-type: none"> High level width $n \times (m+1) / f_i$ n : values set to timer Ai register's high-order address Cycle time $(2^8-1) \times (m+1) / f_i$ m : values set to timer Ai register's low-order address |
| Count start condition | <ul style="list-style-type: none"> The timer overflows The count start flag is set (= 1) |
| Count stop condition | <ul style="list-style-type: none"> The count start flag is reset (= 0) |
| Interrupt request generation timing | PWM pulse goes "L" |
| TA2OUT/TA3OUT pin function | Pulse output |
| Read from timer | When timer Ai register is read, it indicates an indeterminate value |
| Write to timer | <ul style="list-style-type: none"> When counting stopped When a value is written to timer Ai register, it is written to both reload register and counter When counting in progress When a value is written to timer Ai register, it is written to only reload register (Transferred to counter at next reload time) |

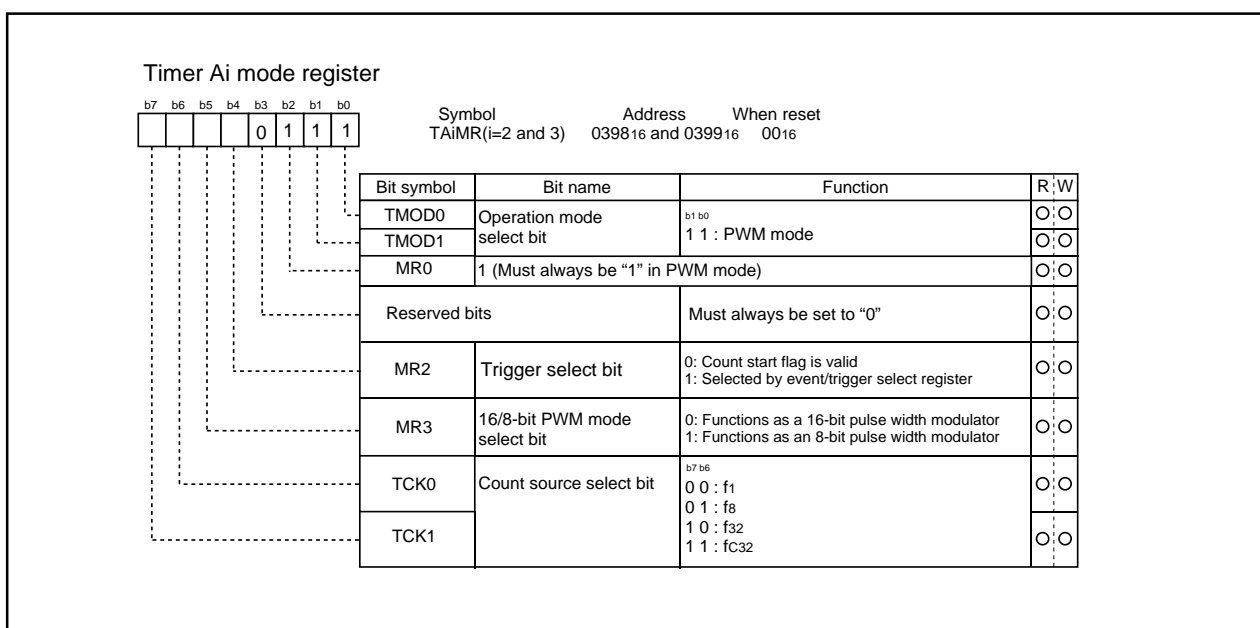


Figure 2.10.14 Timer Ai mode register in pulse width modulation mode (i = 2 and 3)

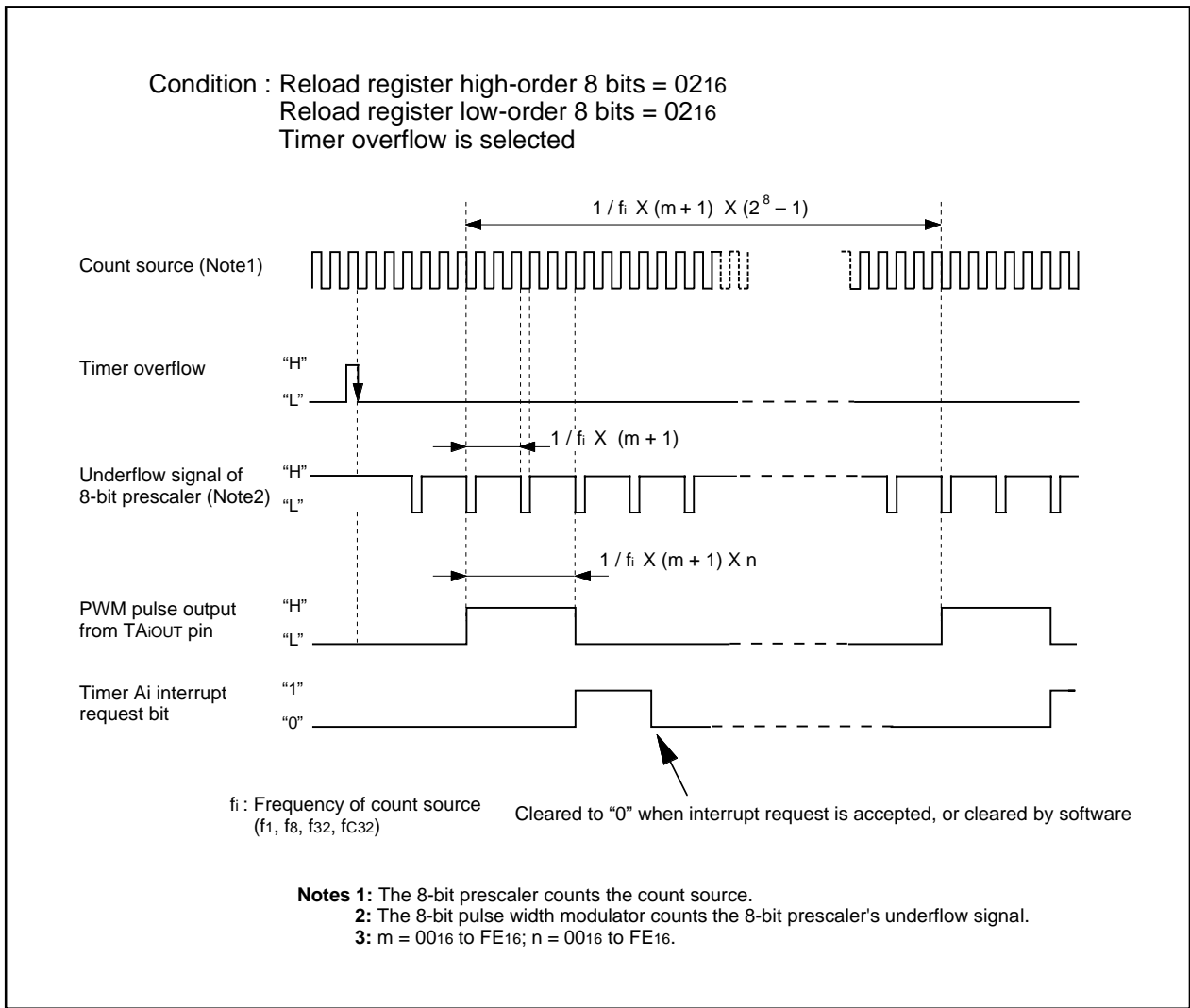


Figure 2.10.15 Example of how an 8-bit pulse width modulator operates

2.10.2 Timer B

Figure 2.10.17 shows the block diagram of timer B. Figures 2.10.17 and 2.10.20 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.

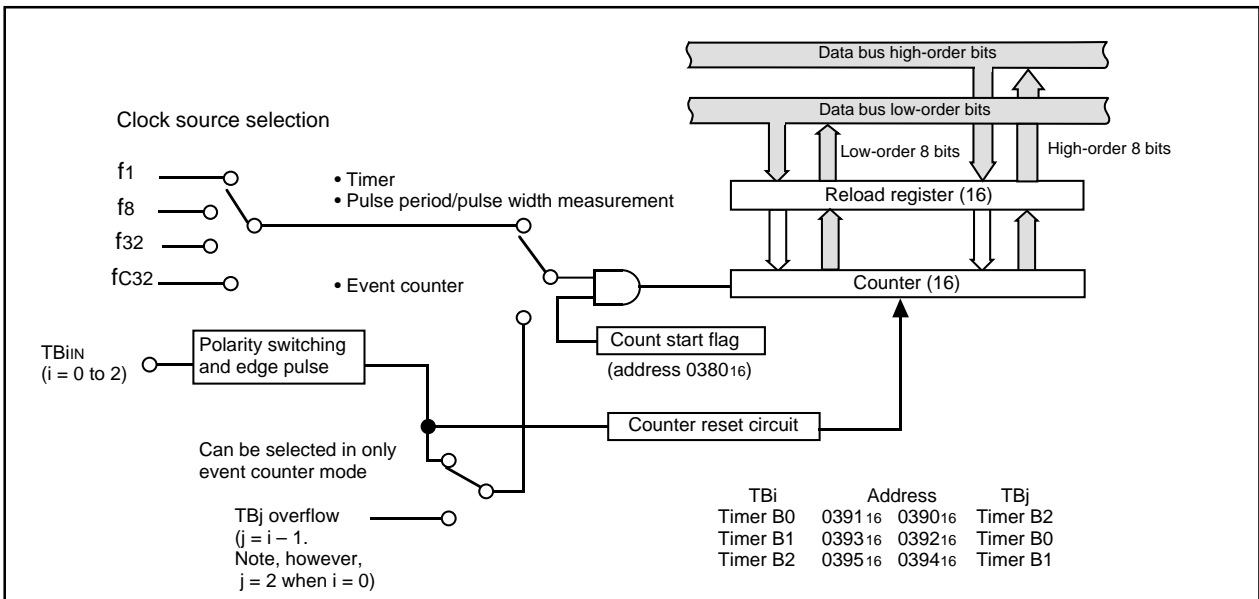


Figure 2.10.16 Block diagram of timer B

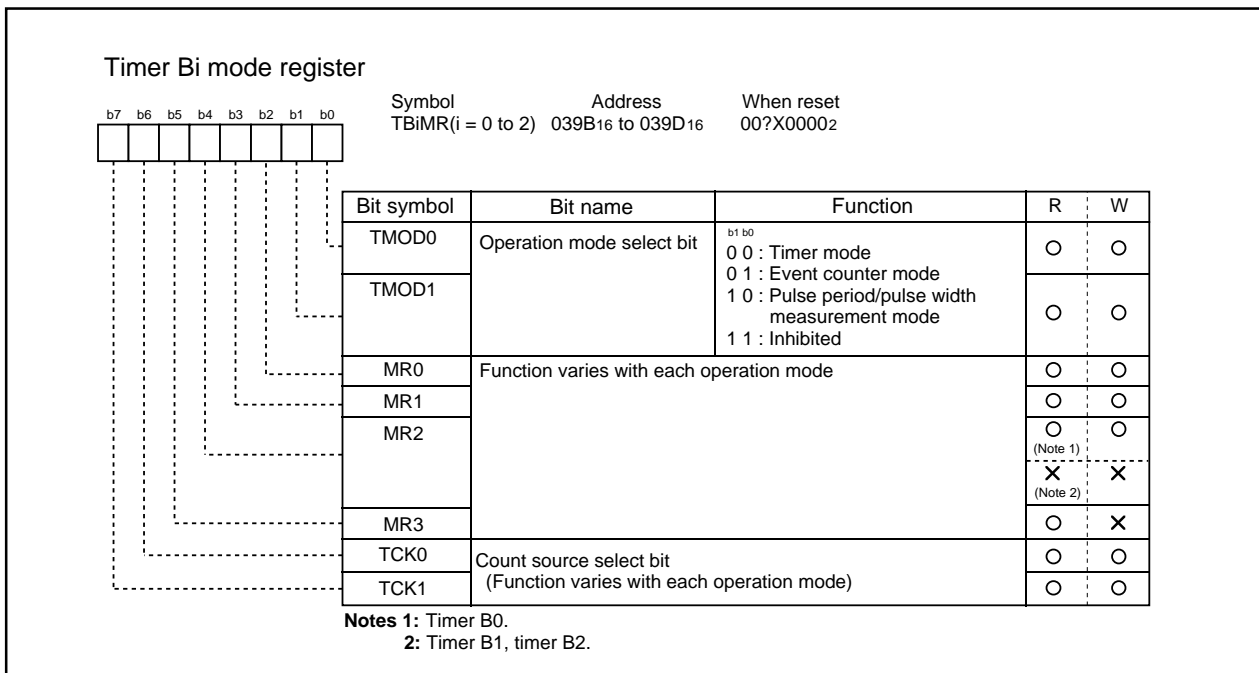


Figure 2.10.17 Timer Bi mode register (i = 0 to 2)

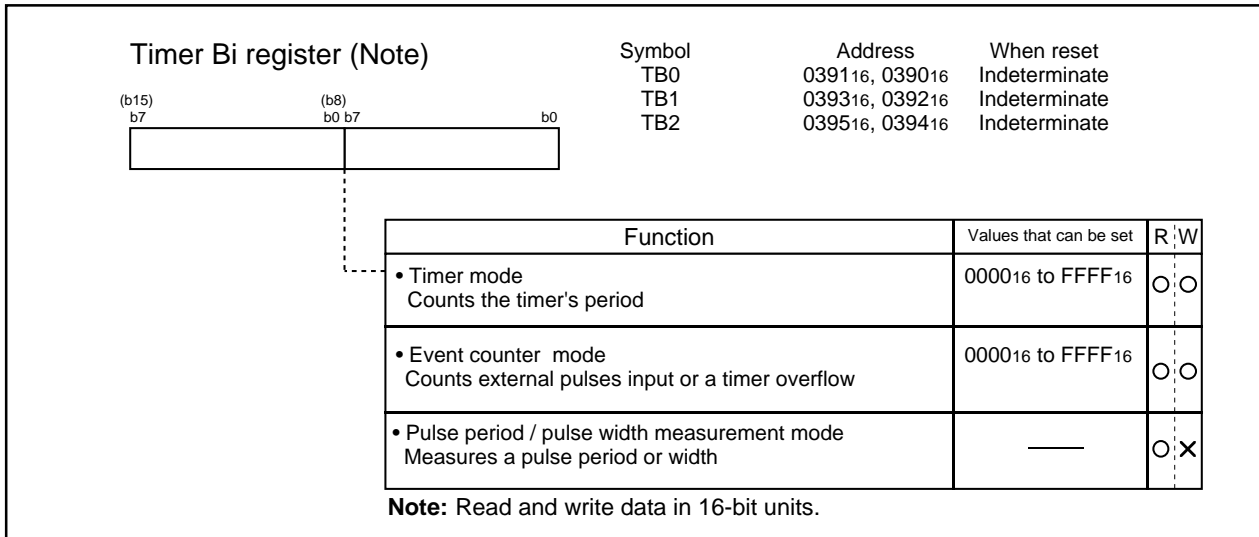


Figure 2.10.18 Timer Bi register (i = 0 to 2)

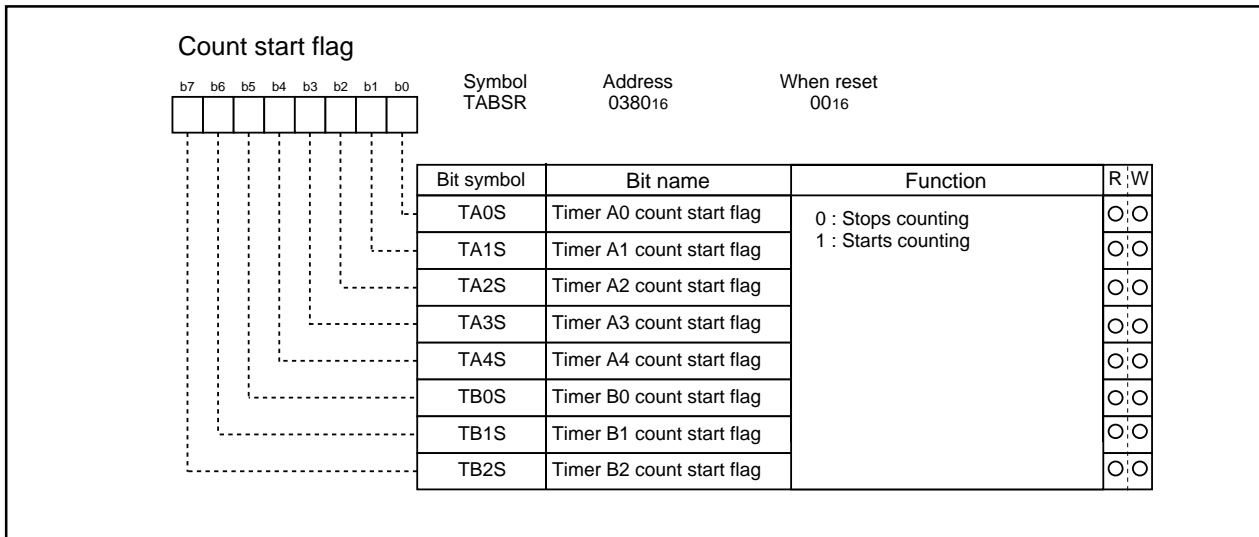


Figure 2.10.19 Count start flag

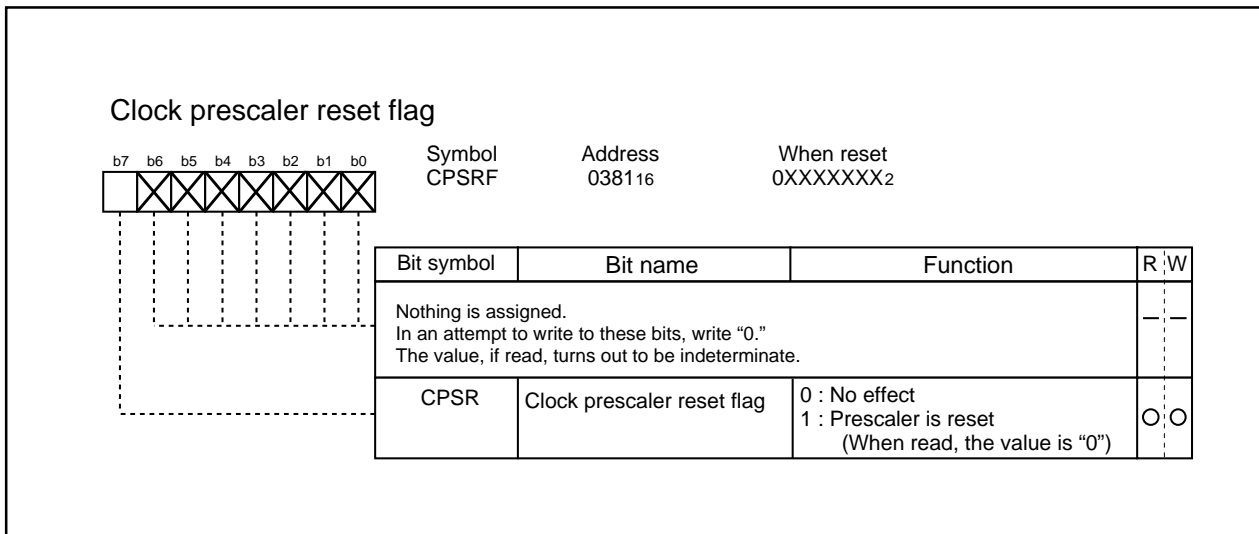


Figure 2.10.20 Clock prescaler reset flag

(1) Timer mode

In this mode, the timer counts an internally generated count source. (See Table 2.10.5) Figure 2.10.21 shows the timer Bi mode register in timer mode.

Table 2.10.5 Timer specifications in timer mode

| Item | Specification |
|-------------------------------------|---|
| Count source | f1, f8, f32, fc32 |
| Count operation | <ul style="list-style-type: none"> Counts down 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 | Programmable I/O port |
| Read from timer | Count value is read out by reading timer Bi register |
| Write to timer | <ul style="list-style-type: none"> When counting stopped When a value is written to timer Bi register, it is written to both reload register and counter When counting in progress When a value is written to timer Bi register, it is written to only reload register (Transferred to counter at next reload time) |

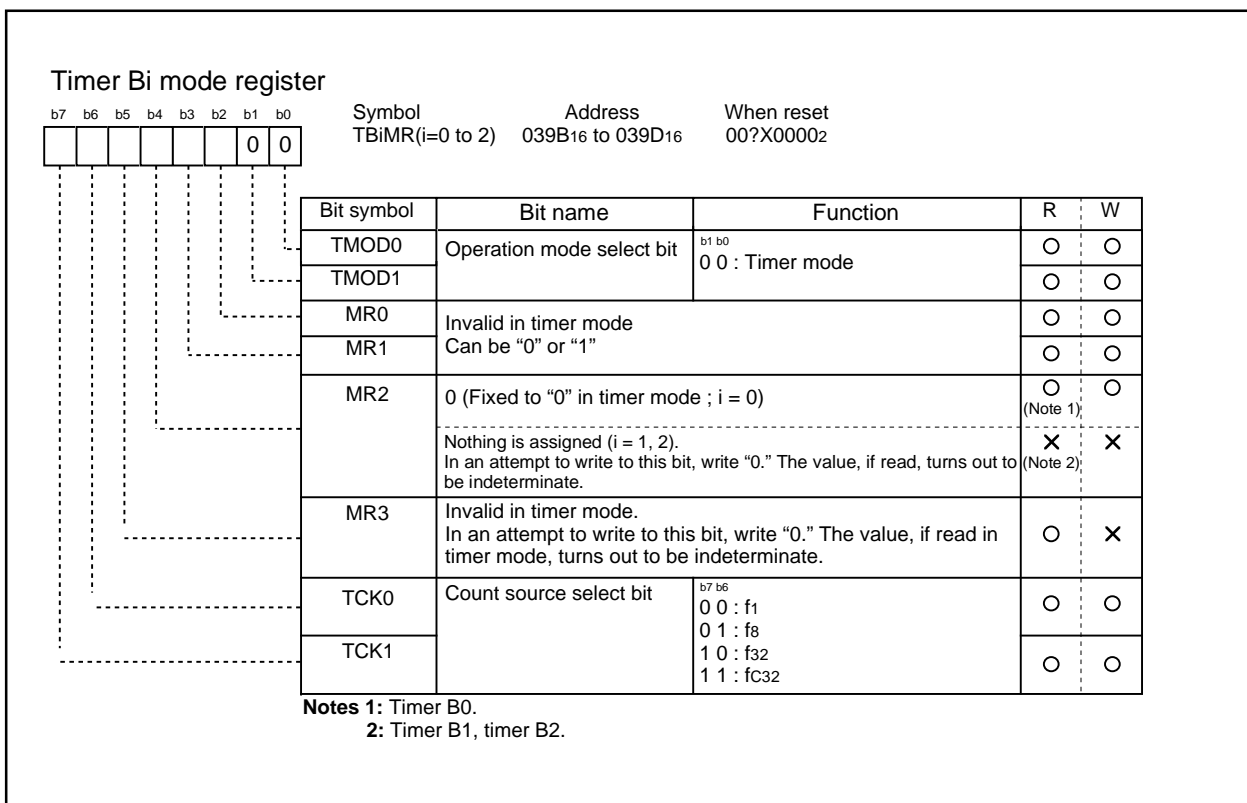


Figure 2.10.21 Timer Bi mode register in timer mode (i = 0 to 2)

(2) Event counter mode

In this mode, the timer counts an external signal or an internal timer's overflow. (See Table 2.10.6) Figure 2.10.22 shows the timer Bi mode register in event counter mode.

Table 2.10.6 Timer specifications in event counter mode

| Item | Specification |
|-------------------------------------|---|
| Count source | <ul style="list-style-type: none"> External signals input to TBiIn pin Effective edge of count source can be a rising edge, a falling edge, or falling and rising edges as selected by software |
| Count operation | <ul style="list-style-type: none"> Counts down 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 | <ul style="list-style-type: none"> When counting stopped When a value is written to timer Bi register, it is written to both reload register and counter When counting in progress When a value is written to timer Bi register, it is written to only reload register (Transferred to counter at next reload time) |

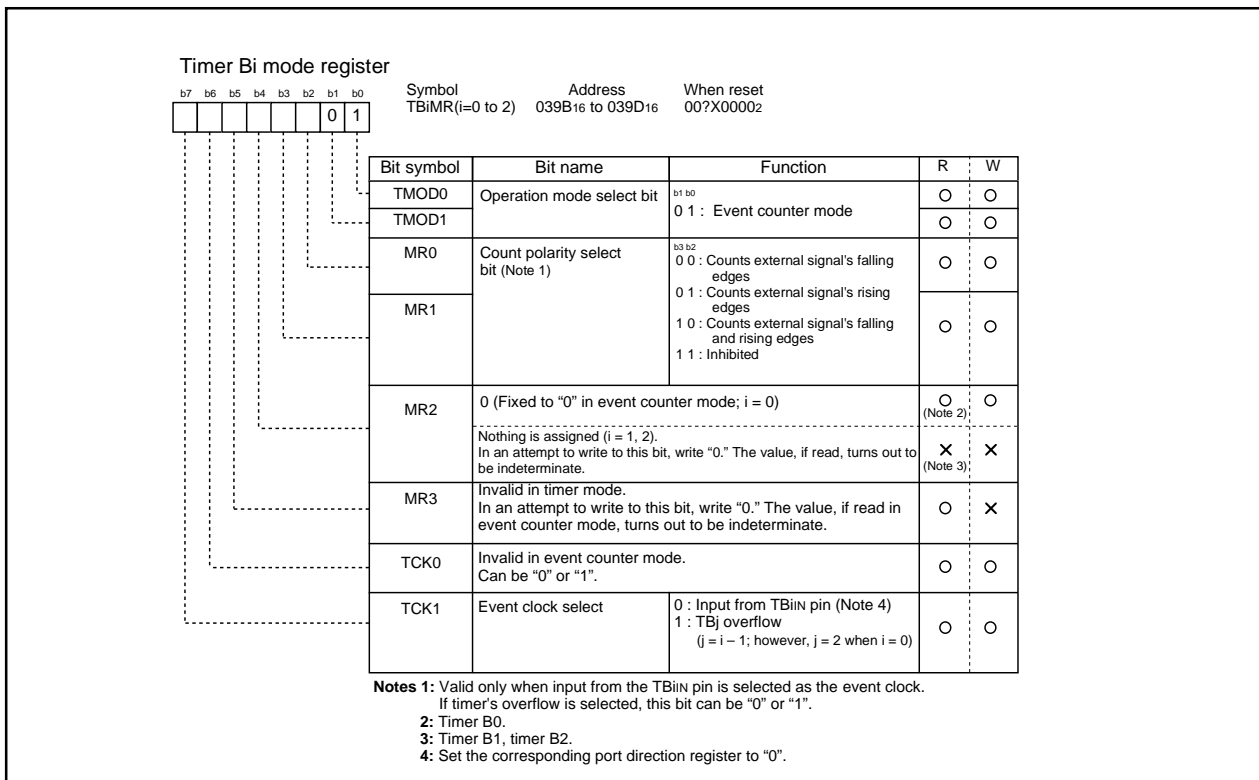


Figure 2.10.22 Timer Bi mode register in event counter mode (i = 0 to 2)

(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 2.10.7) Figure 2.10.23 shows the timer Bi mode register in pulse period/pulse width measurement mode. Figure 2.10.24 shows the operation timing when measuring a pulse period. Figure 2.10.25 shows the operation timing when measuring a pulse width.

Table 2.10.7 Timer specifications in pulse period/pulse width measurement mode

| Item | Specification |
|-------------------------------------|---|
| Count source | f1, f8, f32, fc32 |
| Count operation | <ul style="list-style-type: none"> Up count Counter value "0000₁₆" 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 | <ul style="list-style-type: none"> When measurement pulse's effective edge is input (Note 1) 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.) |
| TBiIN pin function | Measurement pulse input |
| Read from timer | When timer Bi register is read, it indicates the reload register's content (measurement result) (Note 2) |
| Write to timer | Cannot be written to |

Notes 1: An interrupt request is not generated when the first effective edge is input after the timer has started counting.
2: The value read out from the timer Bi register is indeterminate until the second effective edge is input after the timer.

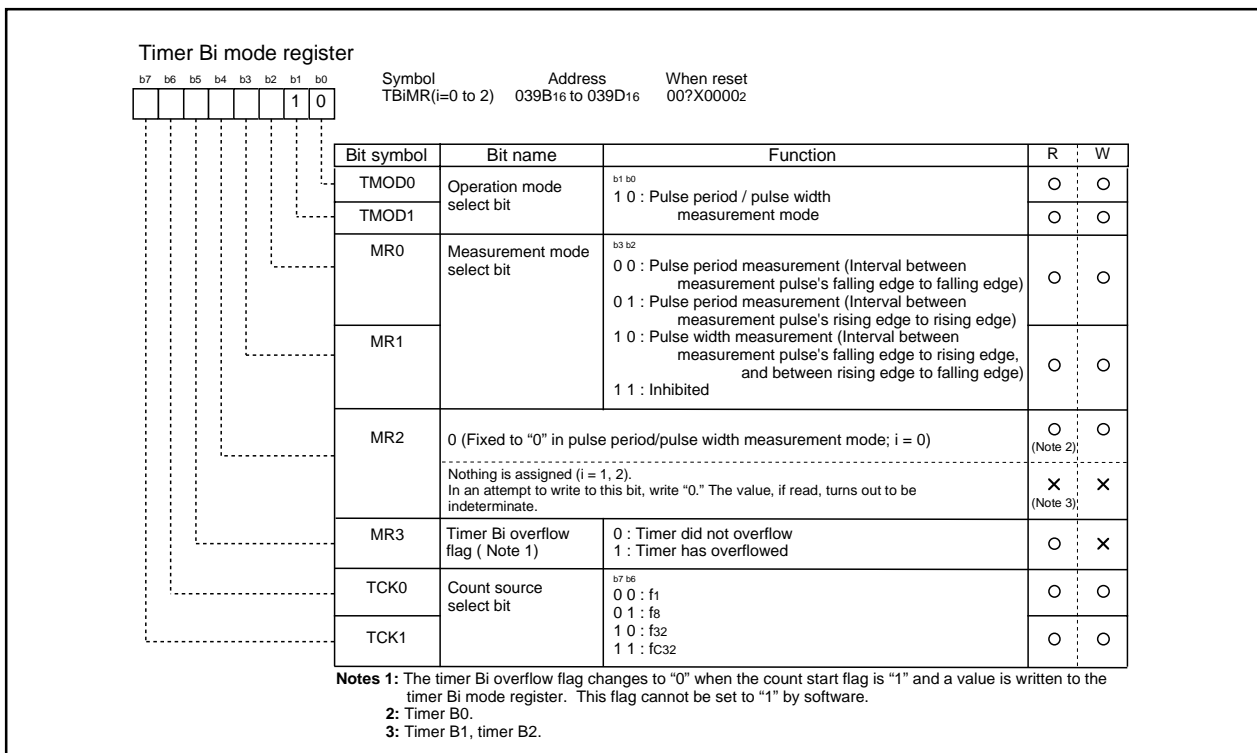


Figure 2.10.23 Timer Bi mode register in pulse period/pulse width measurement mode (i = 0 to 2)

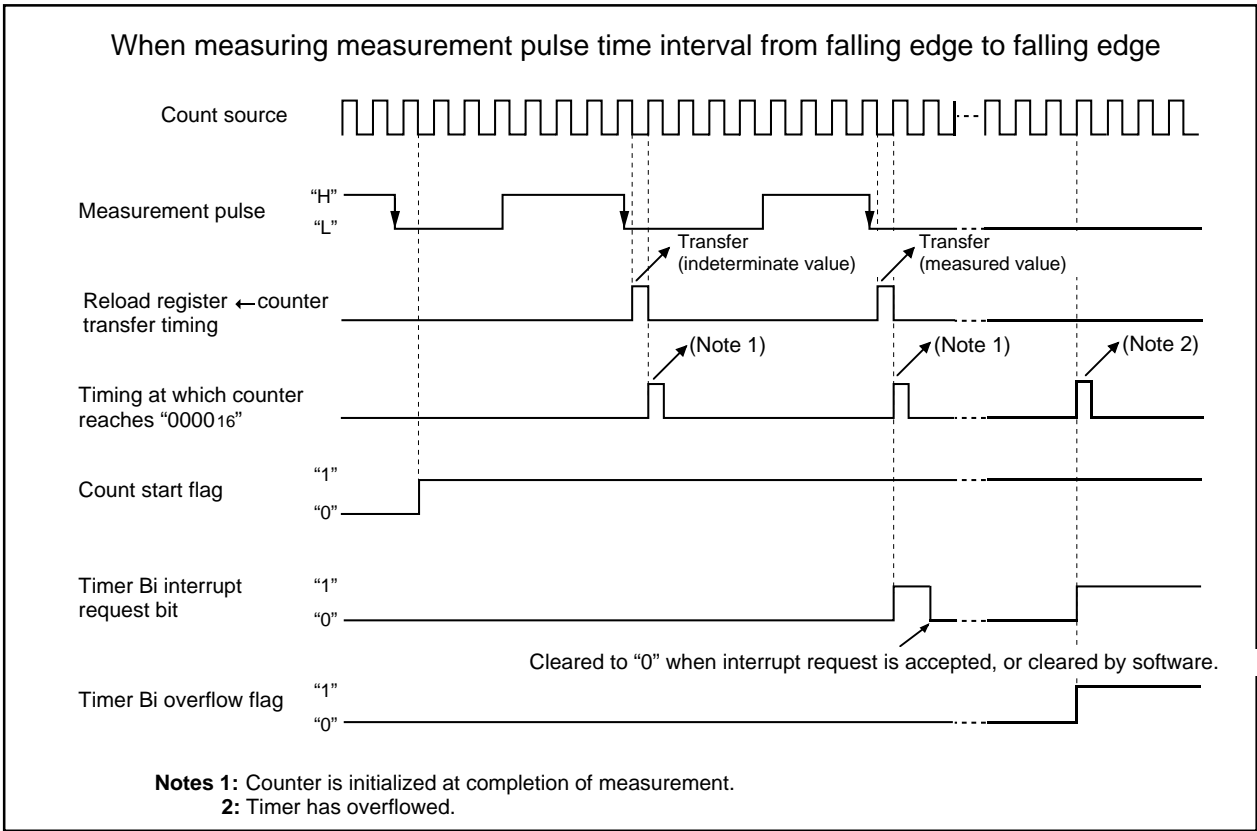


Figure 2.10.24 Operation timing when measuring a pulse period

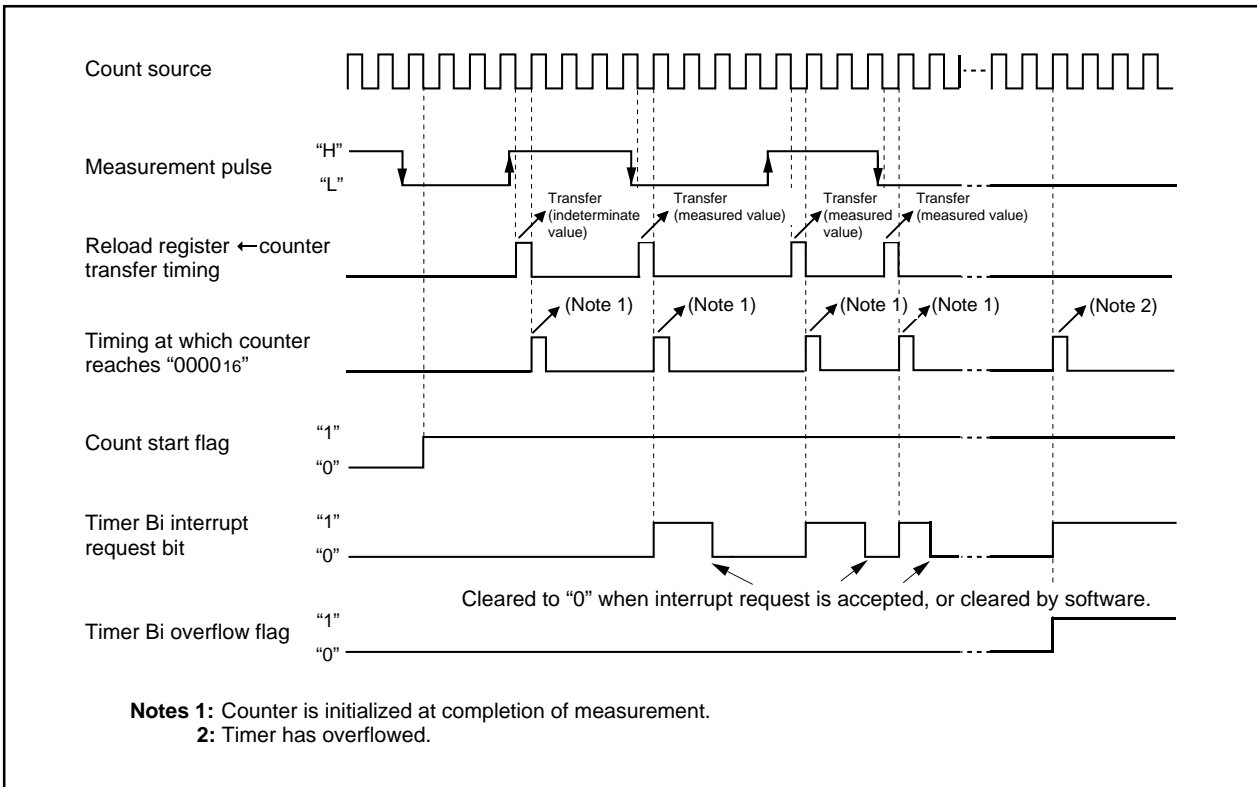


Figure 2.10.25 Operation timing when measuring a pulse width

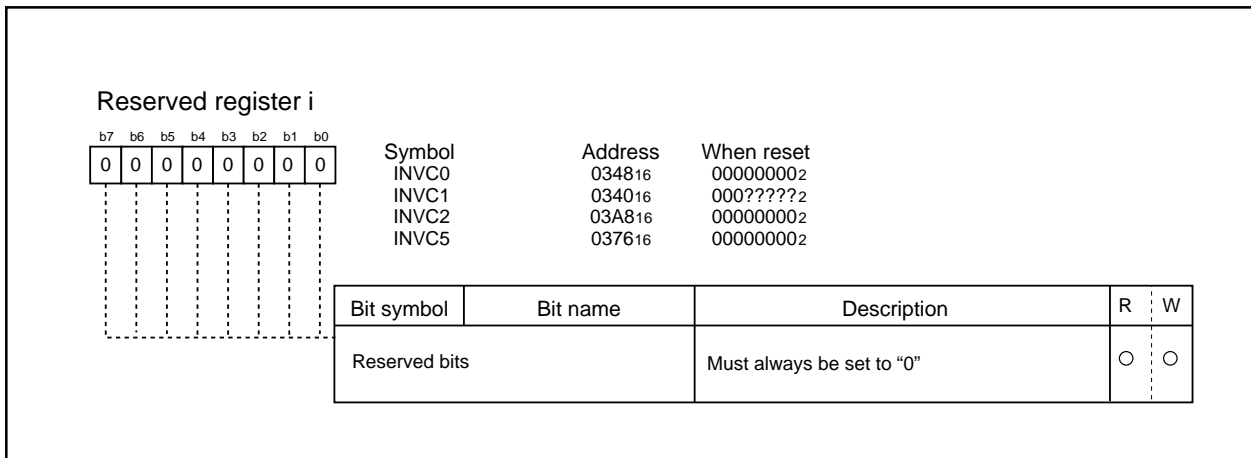


Figure 2.10.26 Reserved register i (i = 0 to 2, 5)

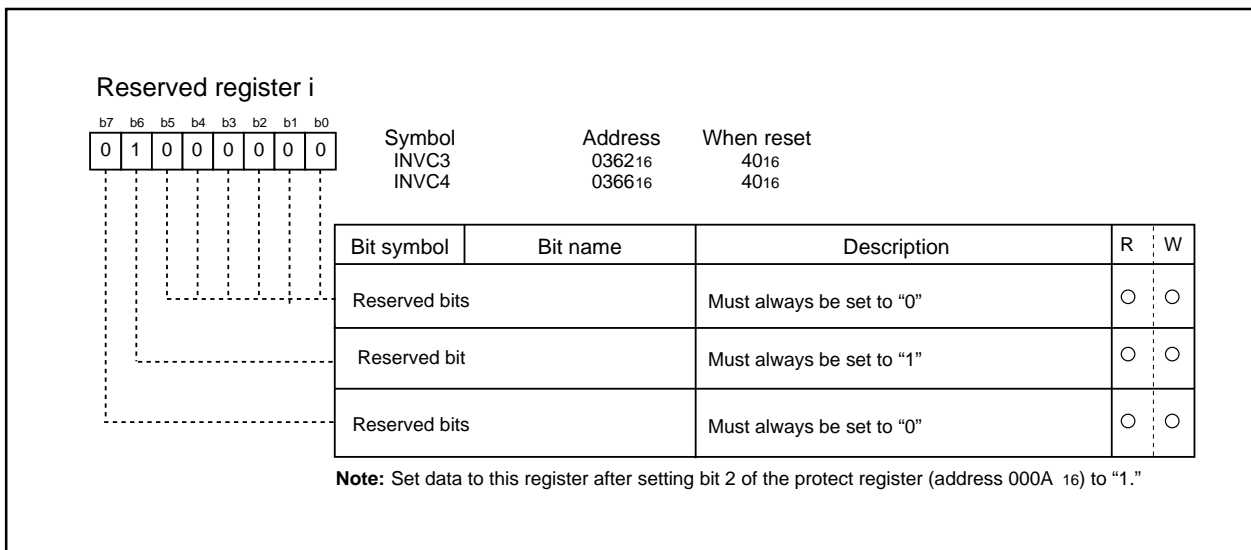


Figure 2.10.27 Reserved register i (i = 3 and 4)

(4) TB0IN noise filter

The input signal of pin TB0IN has the noise filter. The ON/OFF of noise filter and selection of filter clock are set by bits 2 to 4 of the peripheral mode register.

Note: When using the noise filter, set bit 7 of the peripheral mode register according to the main clock frequency.

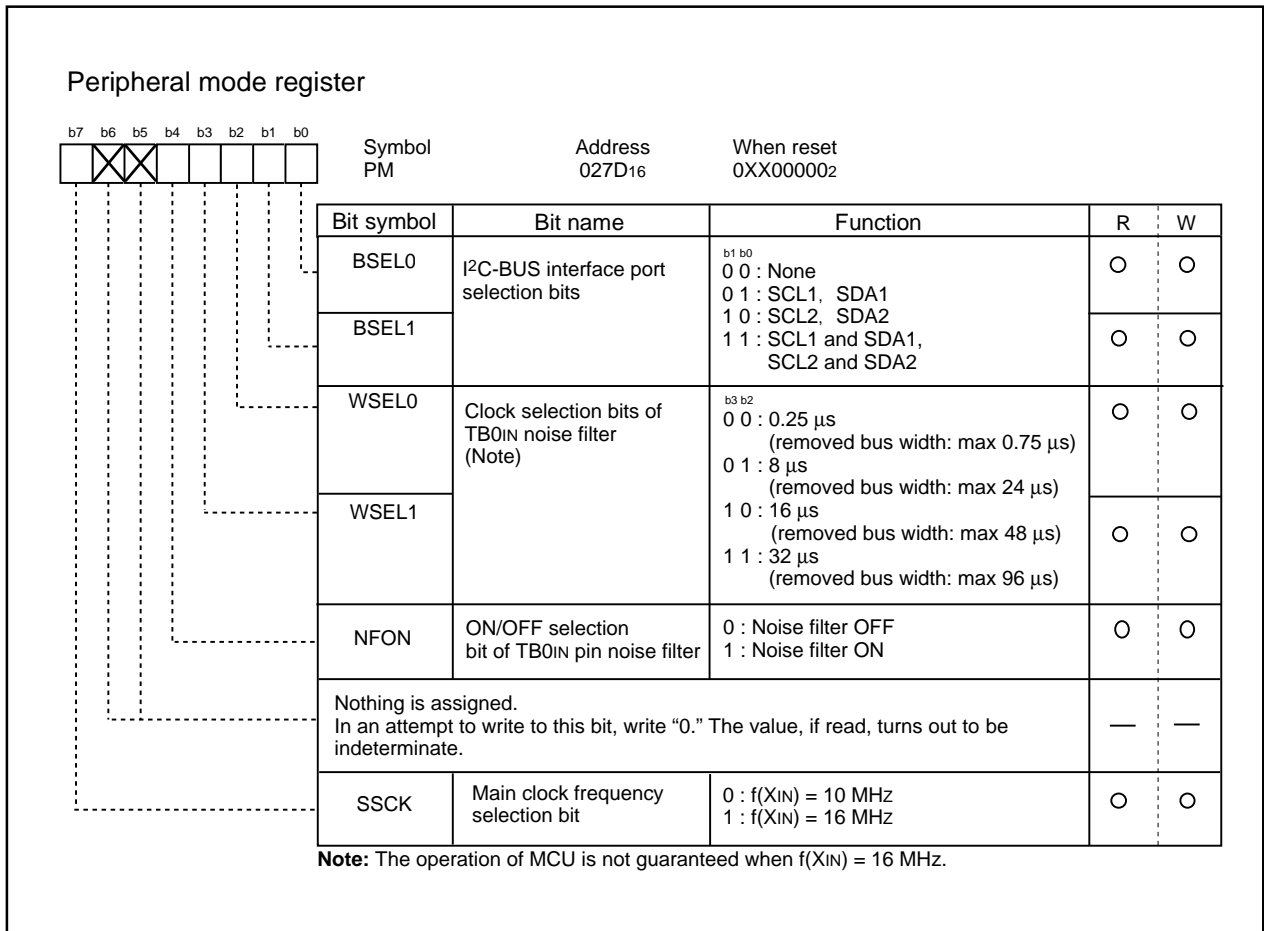


Figure 2.10.28 Peripheral mode register

2.11 Serial I/O

Serial I/O is configured as 4 unites: UART0, UART2, multi-master I²C-BUS interface 0, and multi-master I²C-BUS interface 1.

2.11.1 UART0 and UART2

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

Figure 2.11.1 shows the block diagram of UART0 and UART2. Figures 2.11.2 and 2.11.3 show the block diagram of the transmit/receive unit.

UARTi (i = 0 and 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 03A0₁₆ and 0378₁₆) determine whether UARTi is used as a clock synchronous serial I/O or as a UART. Although a few functions are different, UART0 and UART2 have almost the same functions.

UART0 and UART2 are almost equal in their functions with minor exceptions. UART2, in particular, is compliant with the SIM interface. 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 2.11.1 shows the comparison of functions of UART0 and UART2, and Figures 2.11.4 to 2.11.14 show the registers related to UARTi.

Table 2.11.1 Comparison of functions of UART0 and UART2

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

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

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

3: Only when UART mode.

4: Using for SIM interface.

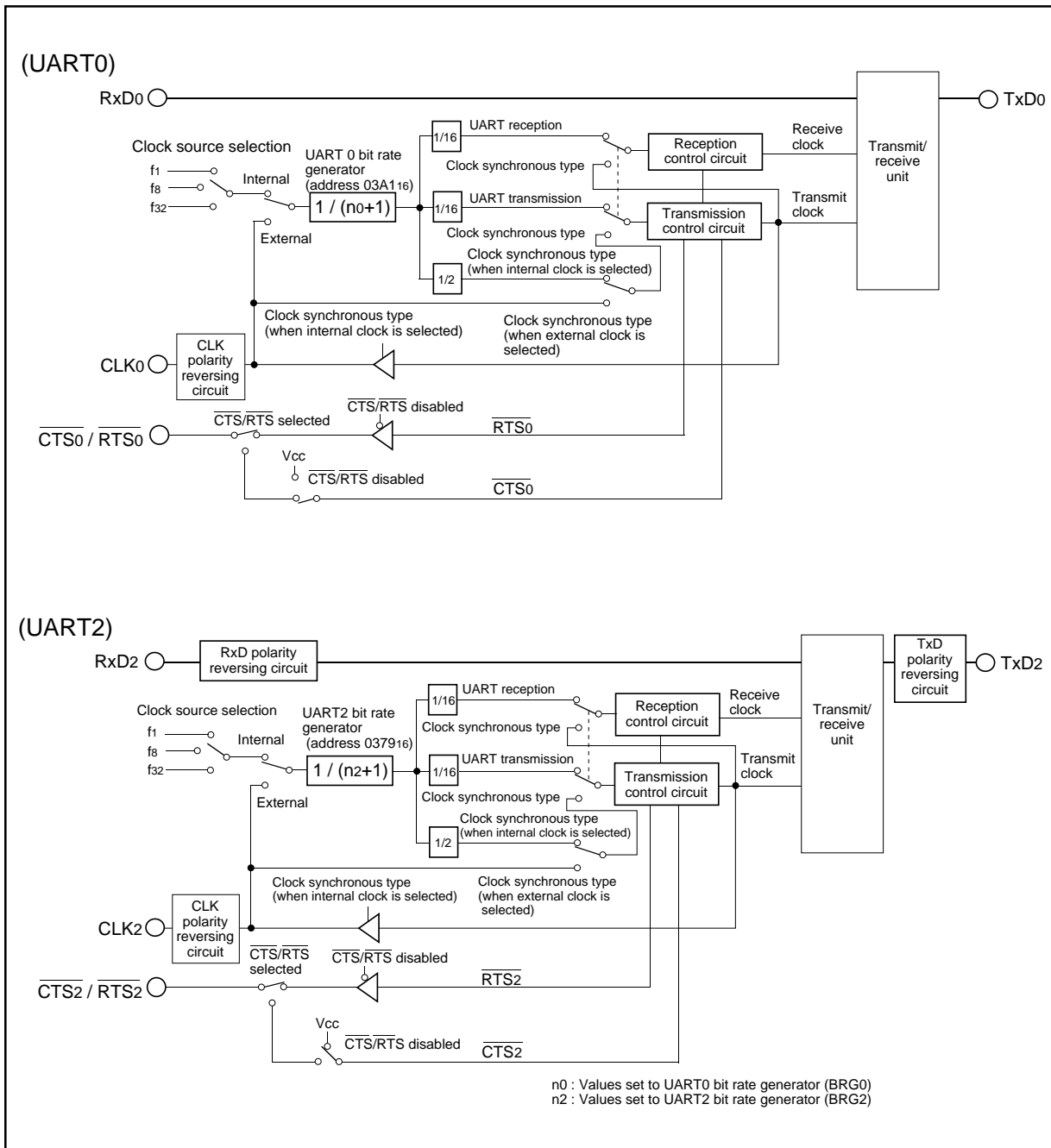


Figure 2.11.1 Block diagram of UARTi (i = 0 and 2)

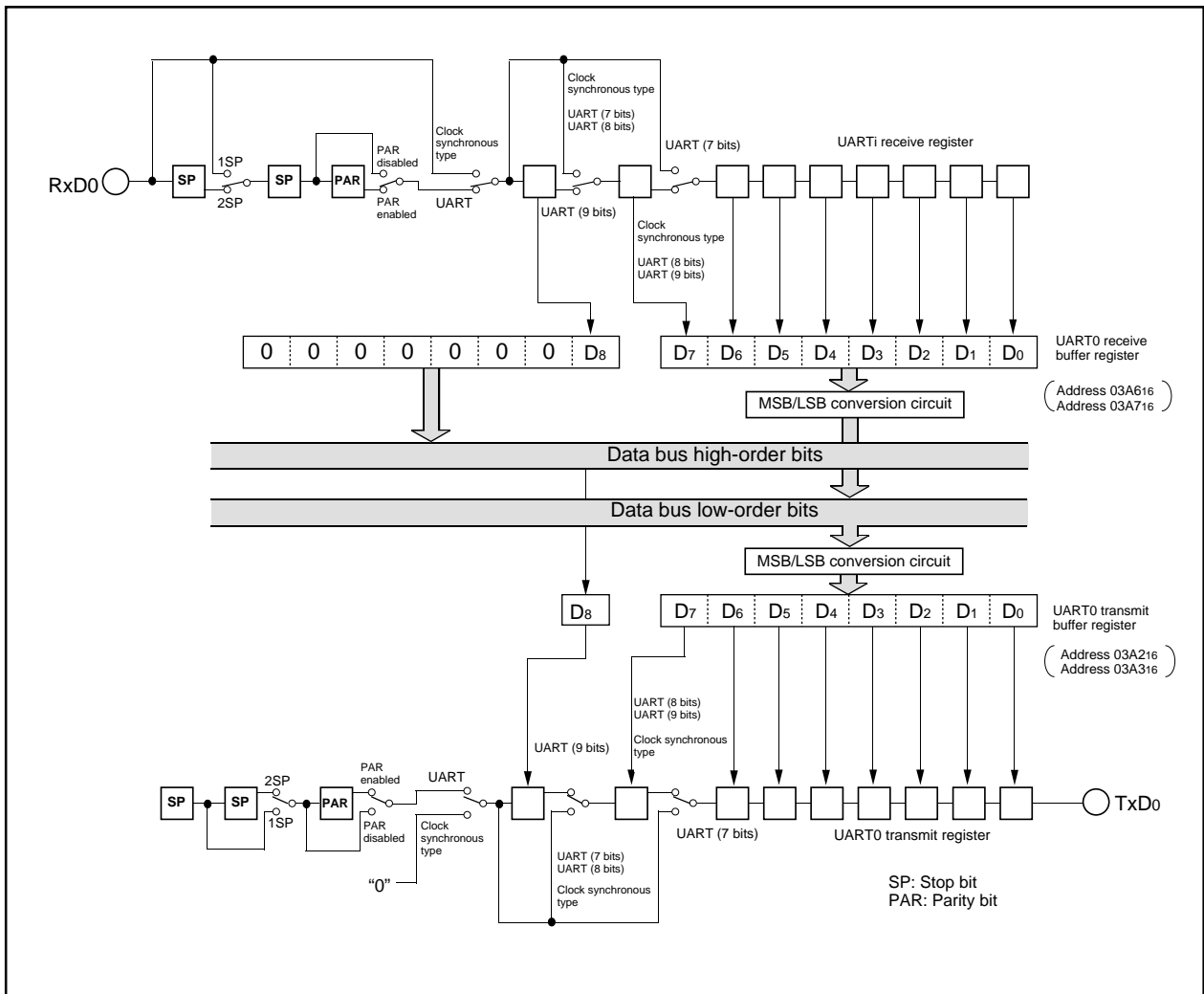


Figure 2.11.2 Block diagram of UART0 transmit/receive unit

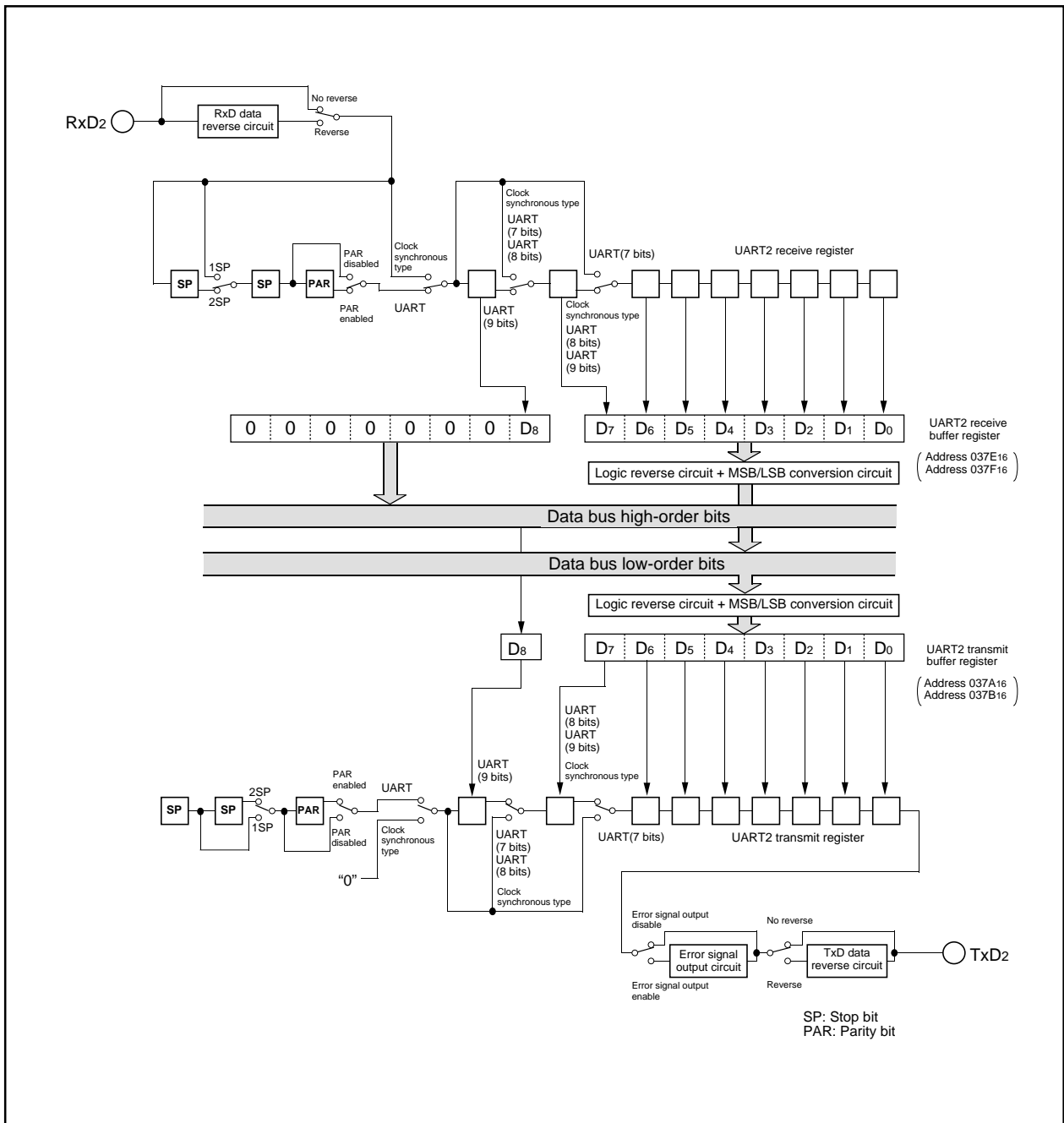


Figure 2.11.3 Block diagram of UART2 transmit/receive unit

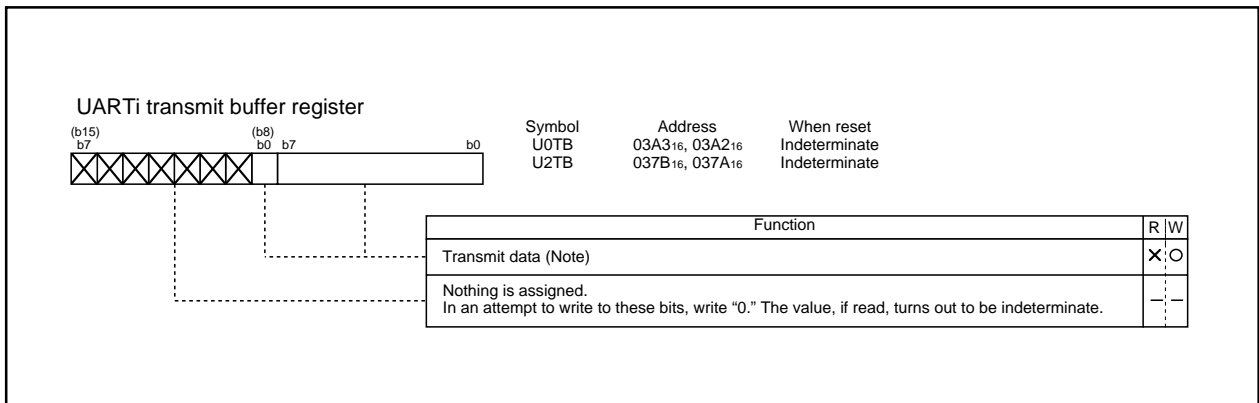


Figure 2.11.4 UARTi transmit buffer register (i = 0 and 2)

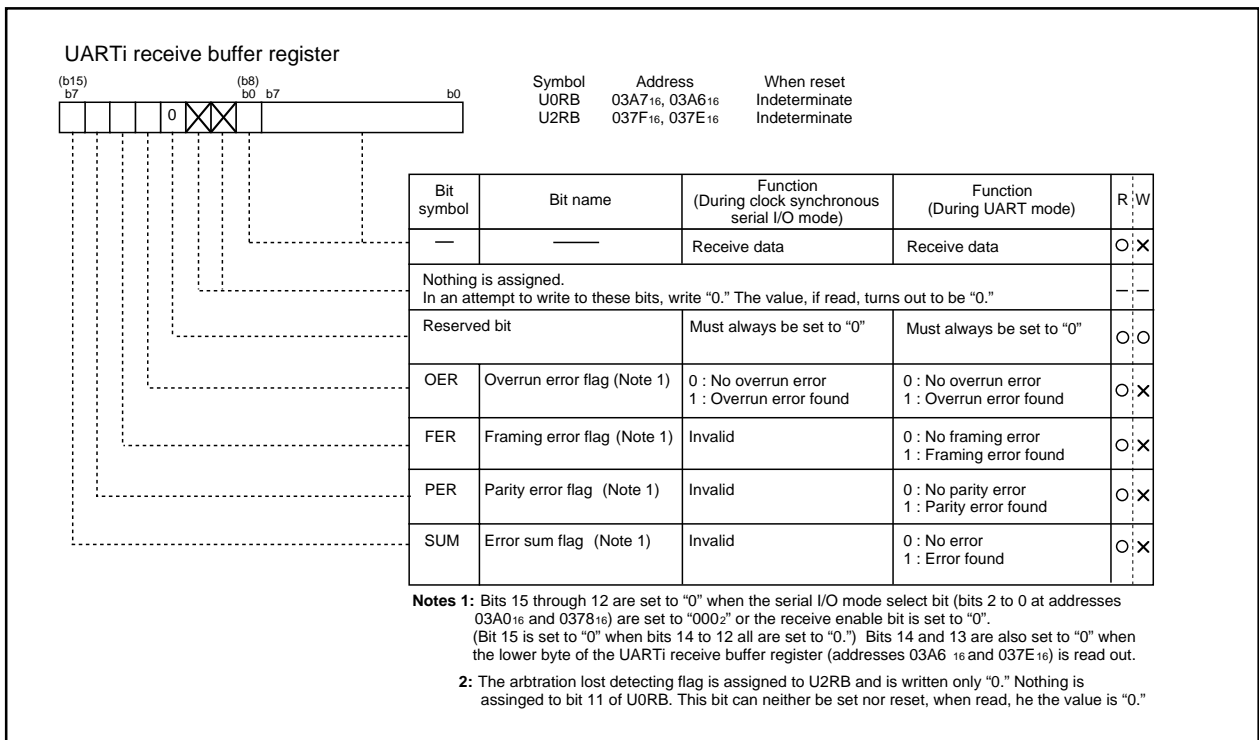


Figure 2.11.5 UARTi receive buffer register (i = 0 and 2)

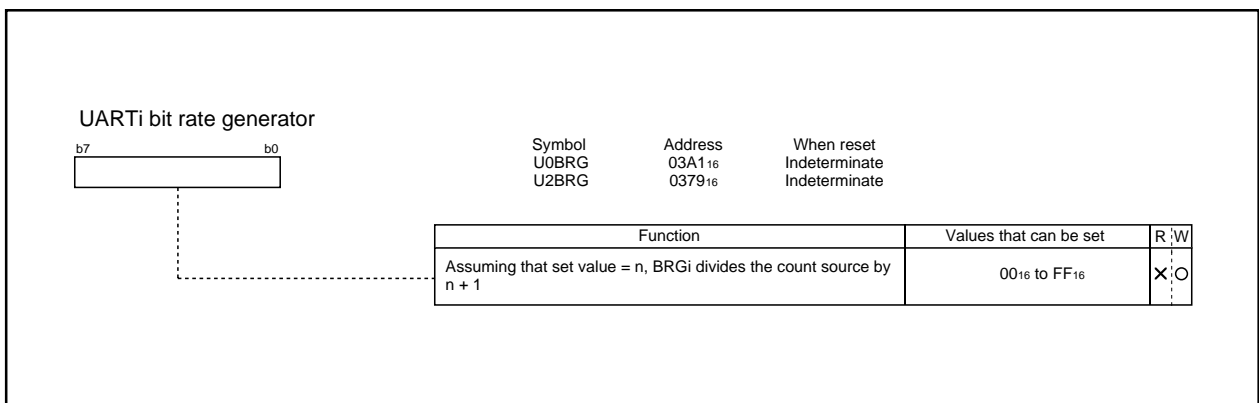


Figure 2.11.6 UARTi bit rate generator (i = 0 and 2)

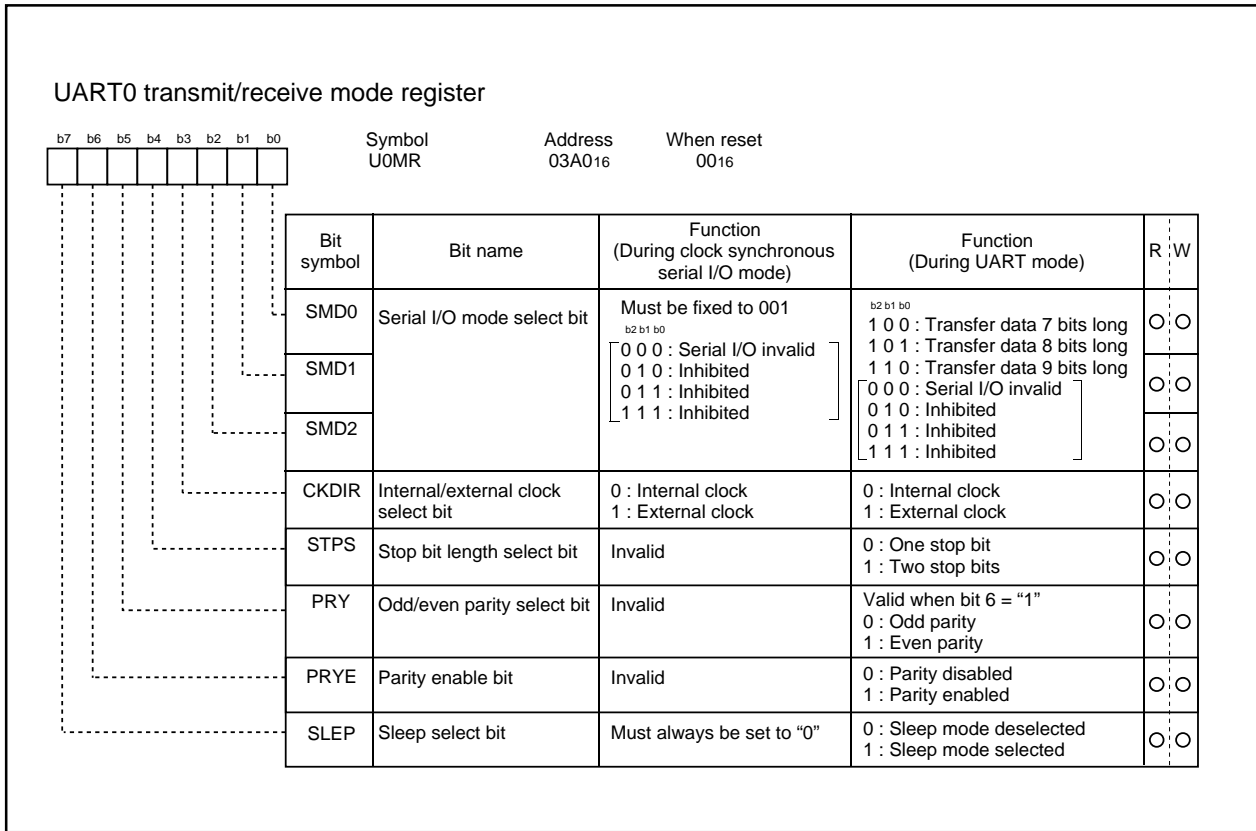


Figure 2.11.7 UART0 transmit/receive mode register

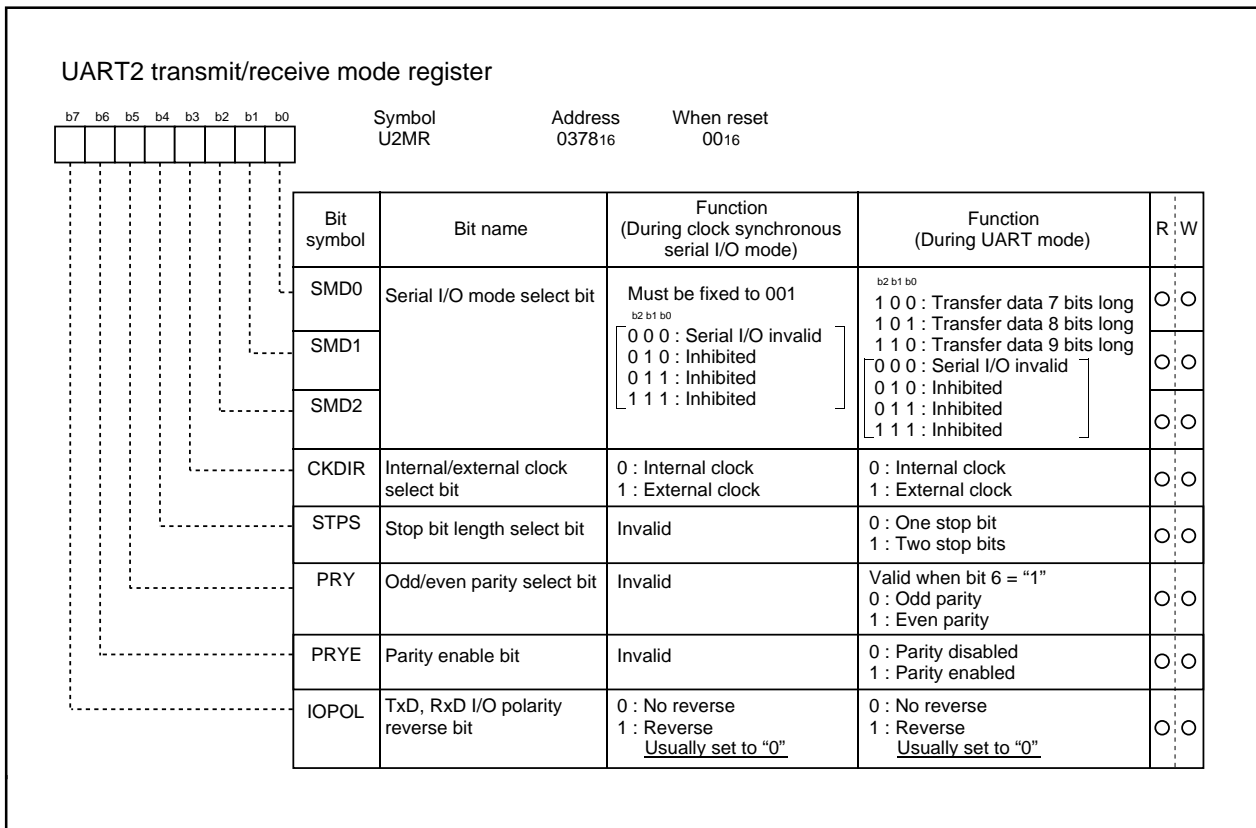


Figure 2.11.8 UART2 transmit/receive mode register

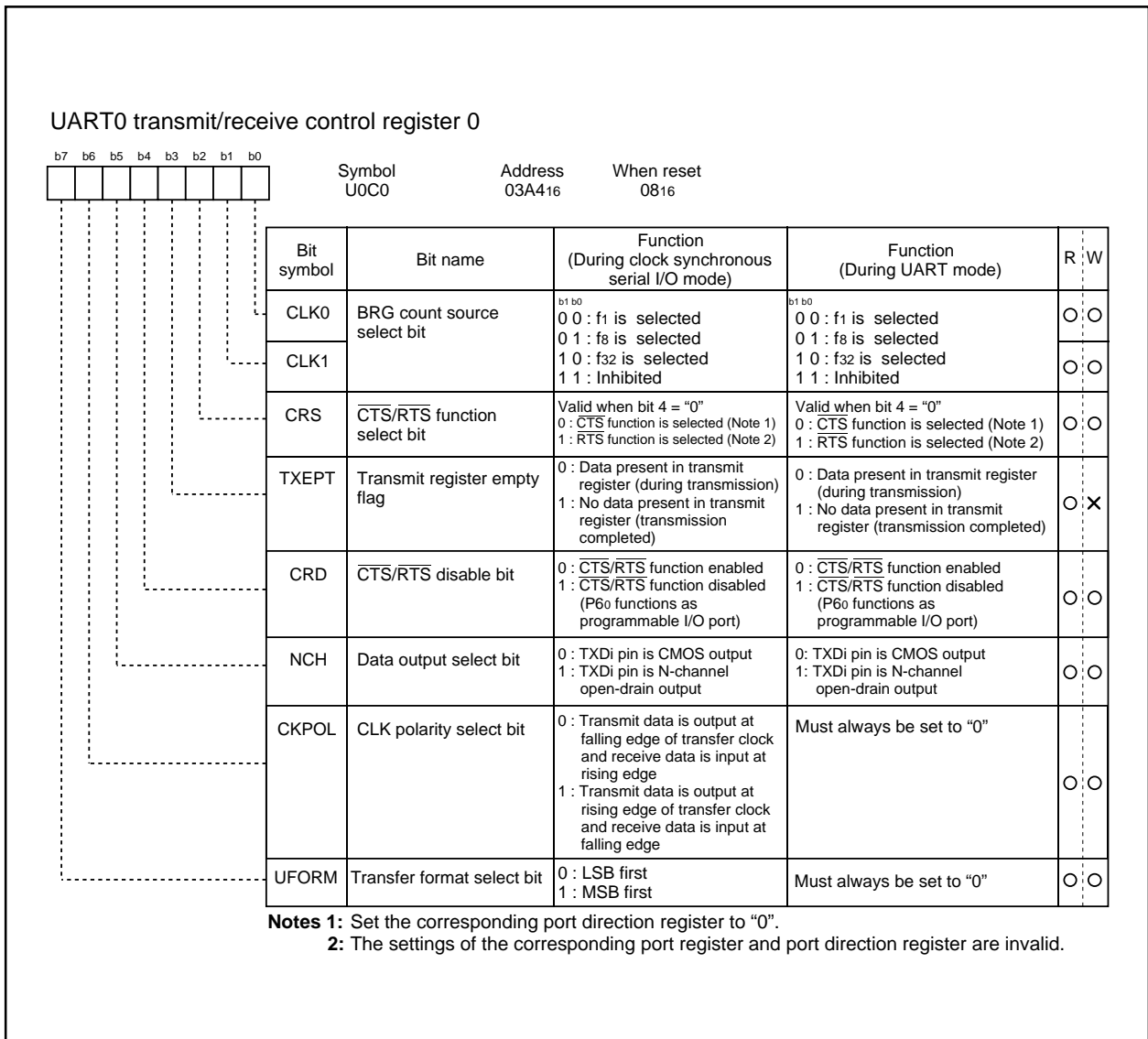


Figure 2.11.9 UART0 transmit/receive control register 0

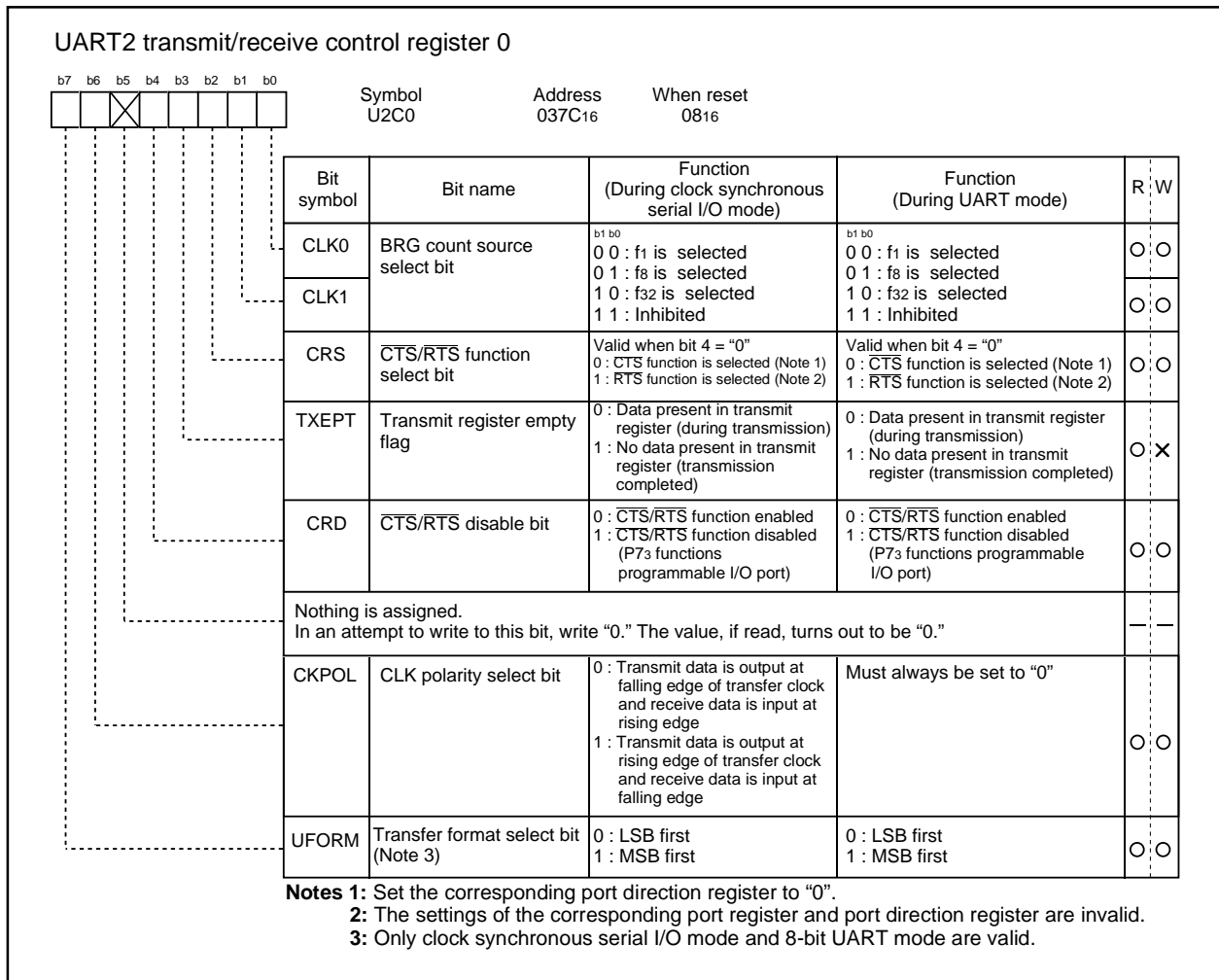


Figure 2.11.10 UART2 transmit/receive control register 0

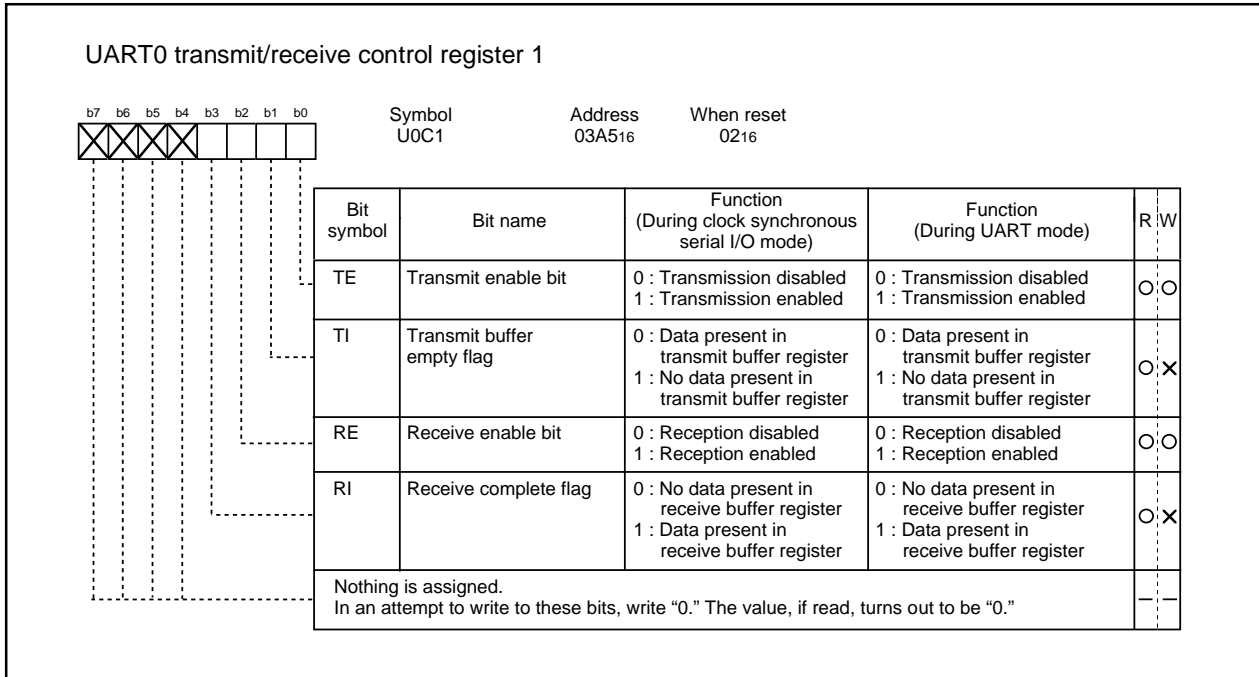


Figure 2.11.11 UART0 transmit/receive control register 1

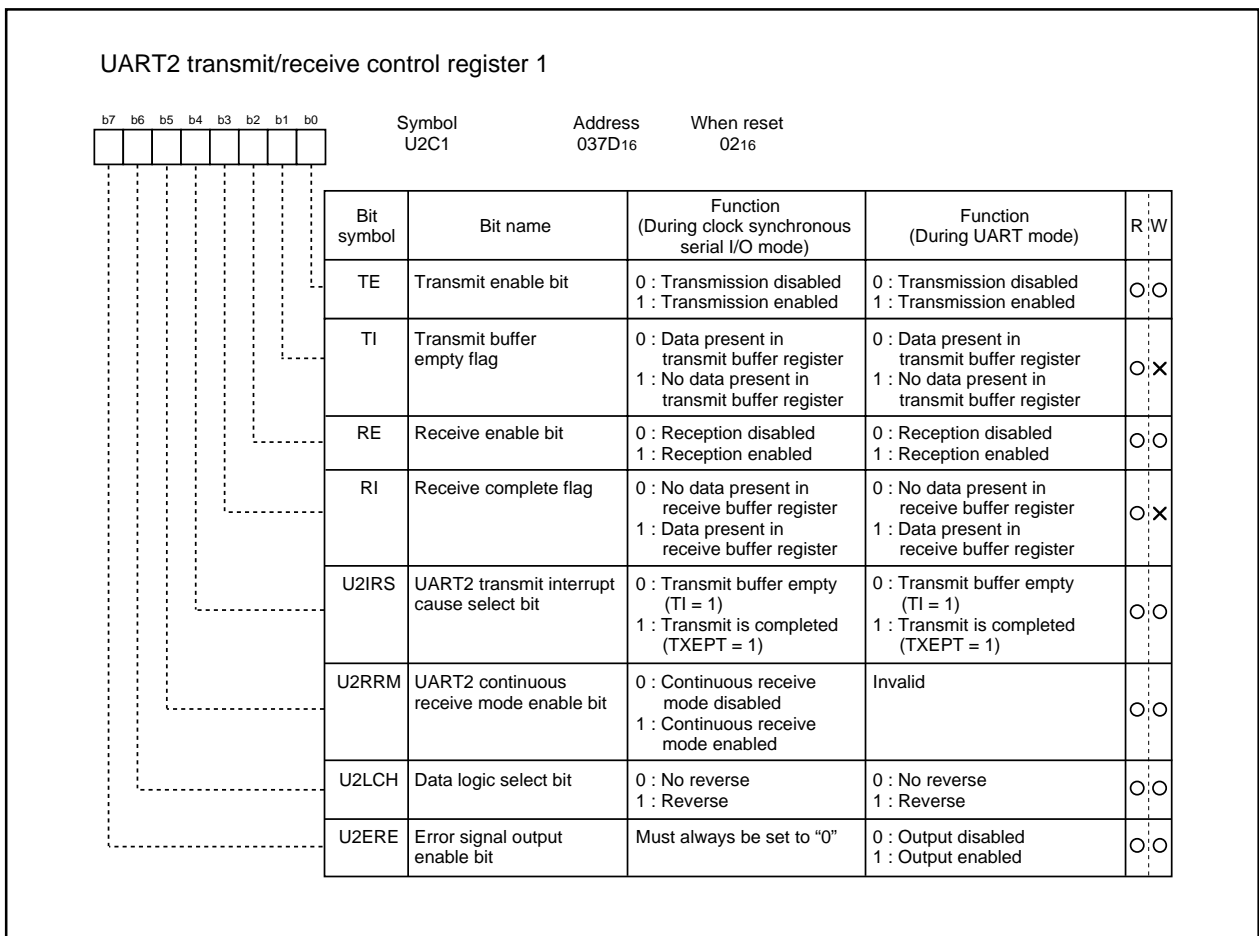


Figure 2.11.12 UART2 transmit/receive control register 1

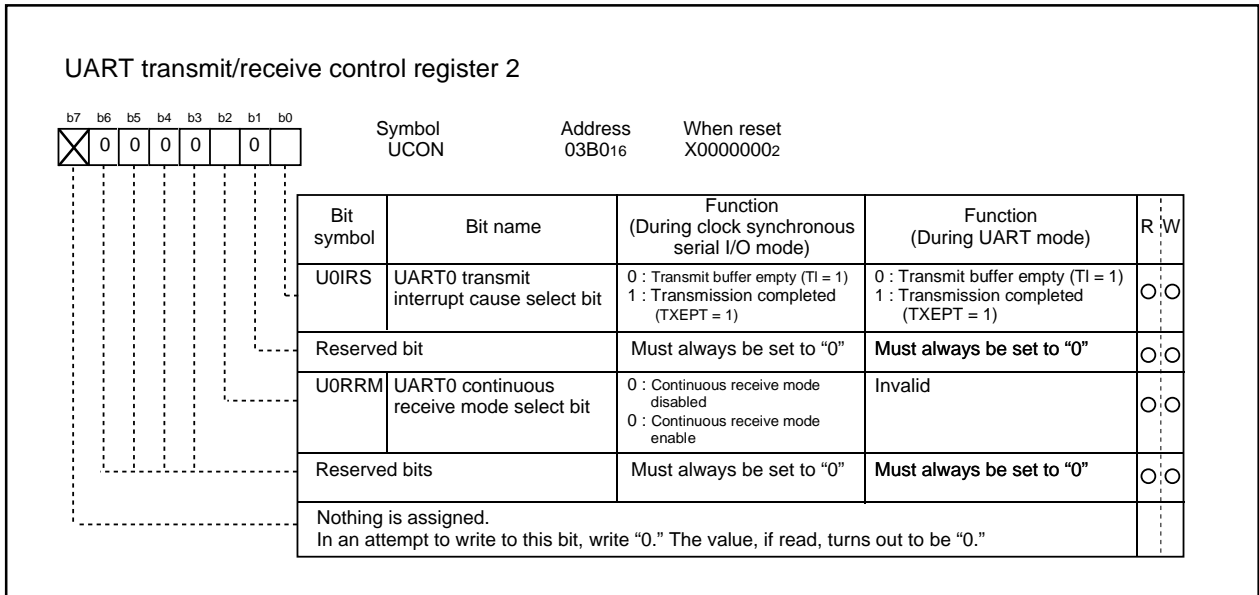


Figure 2.11.13 UART transmit/receive control register 2

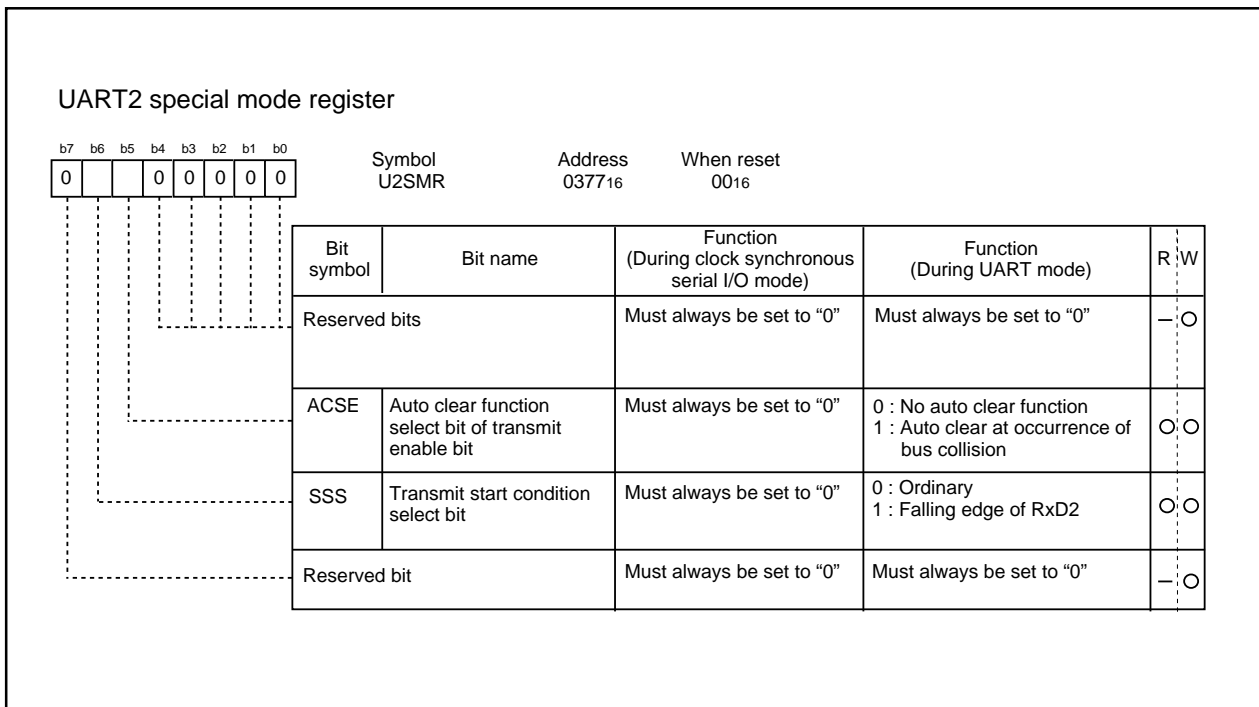


Figure 2.11.14 UART2 special mode register

2.11.2 Clock Synchronous Serial I/O Mode

The clock synchronous serial I/O mode uses a transfer clock to transmit and receive data. Tables 2.11.2 and 2.11.3 list the specifications of the clock synchronous serial I/O mode. Figures 2.11.15 and 2.11.16 show the UART_i transmit/receive mode register in clock synchronous serial I/O mode.

Table 2.11.2 Specifications of clock synchronous serial I/O mode (1)

| Item | Specification |
|--|--|
| Transfer data format Transfer clock | <ul style="list-style-type: none"> • Transfer data length: 8 bits • When internal clock is selected (bit 3 at addresses 03A0₁₆, 0378₁₆ = "0") : $f_i / 2^{(n+1)}$ (Note 1) $f_i = f_1, f_8, f_{32}$ • When external clock is selected (bit 3 at addresses 03A0₁₆, 0378₁₆ = "1") : Input from CLK_i pin |
| Transmission/reception control | <ul style="list-style-type: none"> • CTS function/RTS function/$\overline{\text{CTS}}$, $\overline{\text{RTS}}$ function chosen to be invalid |
| Transmission start condition | <ul style="list-style-type: none"> • To start transmission, the following requirements must be met: <ul style="list-style-type: none"> – Transmit enable bit (bit 0 at addresses 03A5₁₆, 037D₁₆) = "1" – Transmit buffer empty flag (bit 1 at addresses 03A5₁₆, 037D₁₆) = "0" – When $\overline{\text{CTS}}$ function selected, $\overline{\text{CTS}}$ input level = "L" • Furthermore, if external clock is selected, the following requirements must also be met: <ul style="list-style-type: none"> – CLK_i polarity select bit (bit 6 at addresses 03A4₁₆, 037C₁₆) = "0": CLK_i input level = "H" – CLK_i polarity select bit (bit 6 at addresses 03A4₁₆, 037C₁₆) = "1": CLK_i input level = "L" |
| Reception start condition | <ul style="list-style-type: none"> • To start reception, the following requirements must be met: <ul style="list-style-type: none"> – Receive enable bit (bit 2 at addresses 03A5₁₆, 037D₁₆) = "1" – Transmit enable bit (bit 0 at addresses 03A5₁₆, 037D₁₆) = "1" – Transmit buffer empty flag (bit 1 at addresses 03A5₁₆, 037D₁₆) = "0" • Furthermore, if external clock is selected, the following requirements must also be met: <ul style="list-style-type: none"> – CLK_i polarity select bit (bit 6 at addresses 03A4₁₆, 037C₁₆) = "0": CLK_i input level = "H" – CLK_i polarity select bit (bit 6 at addresses 03A4₁₆, 037C₁₆) = "1": CLK_i input level = "L" |
| Interrupt request generation timing | <ul style="list-style-type: none"> • When transmitting <ul style="list-style-type: none"> – Transmit interrupt cause select bit (bit 0 at address 03B0₁₆, bit 4 at address 037D₁₆) = "0": Interrupts requested when data transfer from UART_i transfer buffer register to UART_i transmit register is completed – Transmit interrupt cause select bit (bit 0 at address 03B0₁₆, bit 4 at address 037D₁₆) = "1": Interrupts requested when data transmission from UART_i transfer register is completed • When receiving <ul style="list-style-type: none"> – Interrupts requested when data transfer from UART_i receive register to UART_i receive buffer register is completed |
| Error detection | <ul style="list-style-type: none"> • Overrun error (Note 2) This error occurs when the next data is ready before contents of UART_i receive buffer register are read out |

Table 2.11.3 Specifications of clock synchronous serial I/O mode (2)

| Item | Specification |
|-----------------|--|
| Select function | <ul style="list-style-type: none"> • CLK polarity selection Whether transmit data is output/input at the rising edge or falling edge of the transfer clock can be selected • LSB first/MSB first selection Whether transmission/reception begins with bit 0 or bit 7 can be selected • Continuous receive mode selection Reception is enabled simultaneously by a read from the receive buffer register • Switching serial data logic (UART2) Whether to reverse data in writing to the transmission buffer register or reading the reception buffer register can be selected. • TxD, RxD I/O polarity reverse (UART2) This function is reversing TxD port output and RxD port input. All I/O data level is reversed. |

Notes 1: “n” denotes the value 00₁₆ to FF₁₆ that is set to the UART bit rate generator.

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

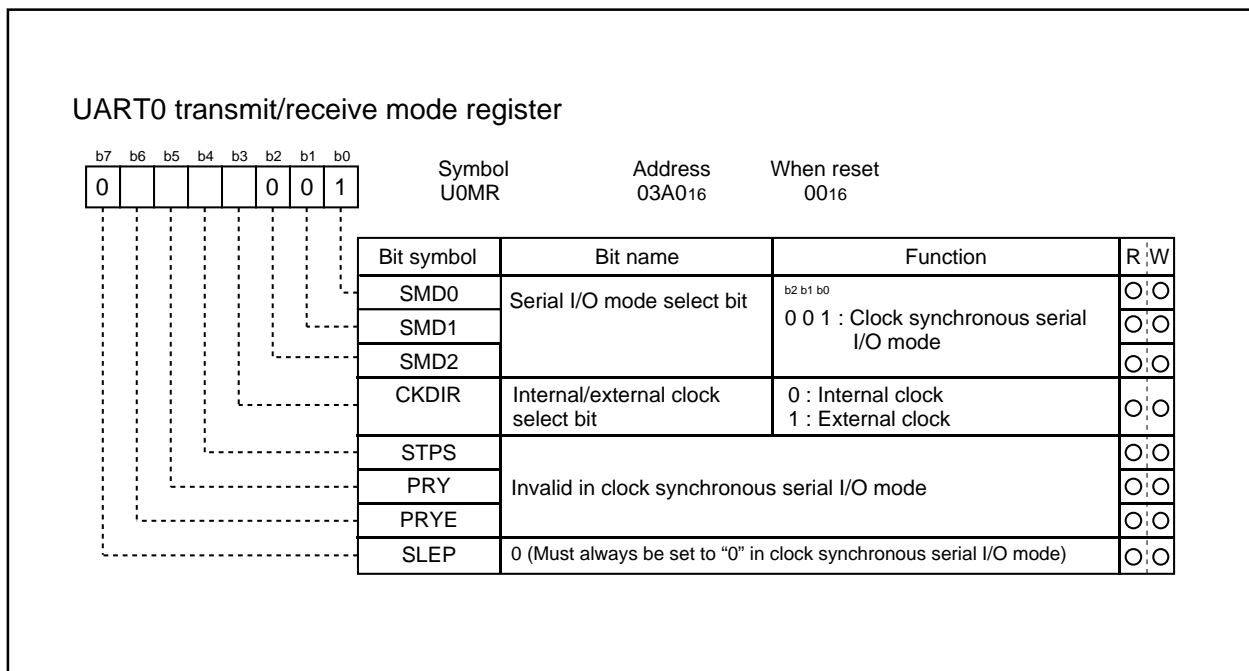


Figure 2.11.15 UART0 transmit/receive mode registers in clock synchronous serial I/O mode

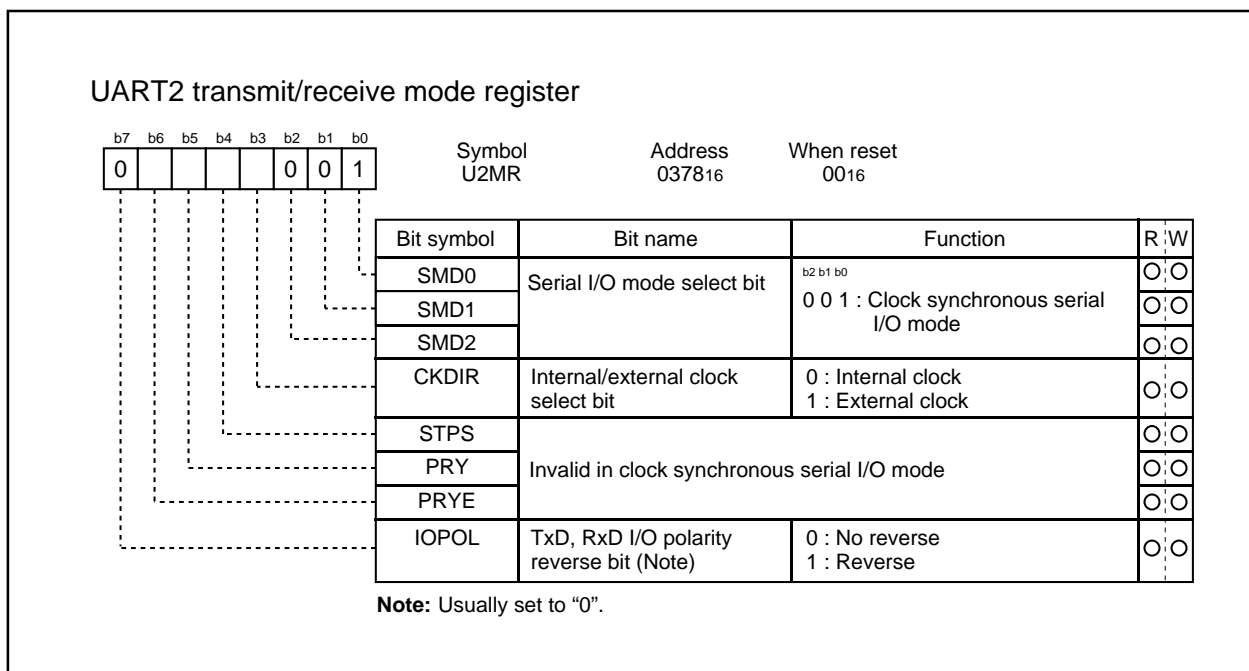


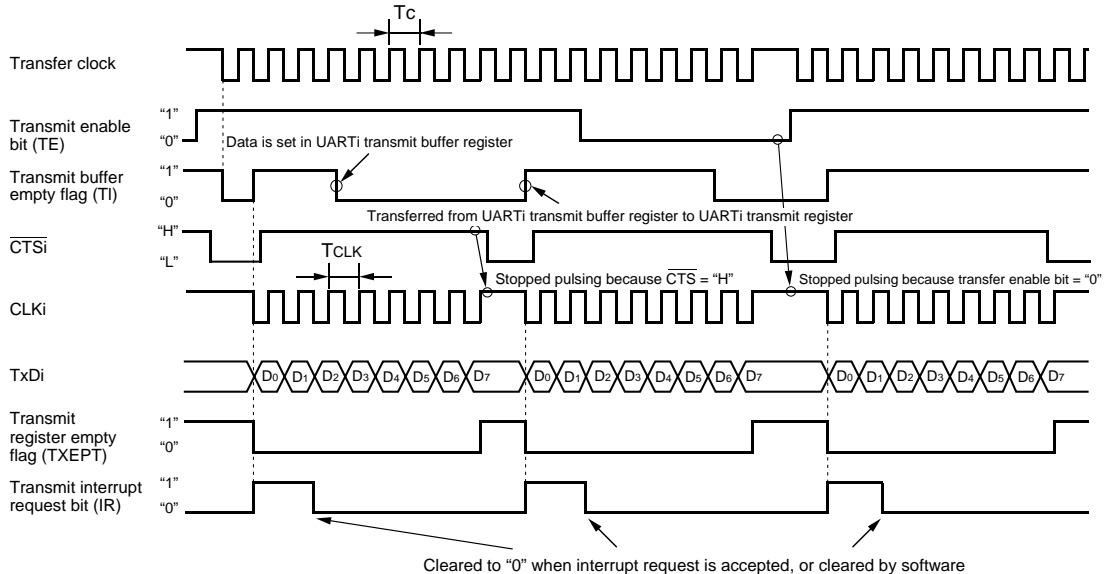
Figure 2.11.16 UART2 transmit/receive mode register in clock synchronous serial I/O mode

Table 2.11.4 lists the functions of the input/output pins during clock synchronous serial I/O mode. 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 2.11.4 Input/output pin functions in clock synchronous serial I/O mode

| Pin name | Function | Method of selection |
|---|--------------------------------|--|
| TxDi (P63, P70) | Serial data output | (Outputs dummy data when performing reception only) |
| RxDi (P62, P71) | Serial data input | Port P62 and P71 direction register (bits 2 at address 03EE 16, bit 1 at address 03EF 16) = “0” (Can be used as an input port when performing transmission only) |
| CLKi (P61, P72) | Transfer clock output | Internal/external clock select bit (bit 3 at address 03A0 16, 0378 16) = “0” |
| | Transfer clock input | Internal/external clock select bit (bit 3 at address 03A0 16, 0378 16) = “1” Port P61 and P72 direction register (bits 1 at address 03EE 16, bit 2 at address 03EF 16) = “0” |
| $\overline{\text{CTS}}/\overline{\text{RTS}}$ (P60, P73) | $\overline{\text{CTS}}$ input | $\overline{\text{CTS}}/\overline{\text{RTS}}$ disable bit (bit 4 at address 03A4 16, 037C 16) = “0” $\overline{\text{CTS}}/\overline{\text{RTS}}$ function select bit (bit 2 at addresses 03A4 16, 037C 16) = “0” Port P60 and P73 direction register (bits 0 at address 03EE 16, bit 3 at address 03EF 16) = “0” |
| | $\overline{\text{RTS}}$ output | $\overline{\text{CTS}}/\overline{\text{RTS}}$ disable bit (bit 4 at address 03A4 16, 037C 16) = “0” $\overline{\text{CTS}}/\overline{\text{RTS}}$ function select bit (bit 2 at address 03A4 16, 037C 16) = “1” |
| | Programmable I/O port | $\overline{\text{CTS}}/\overline{\text{RTS}}$ disable bit (bit 4 at address 03A4 16, 037C 16) = “1” |

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



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

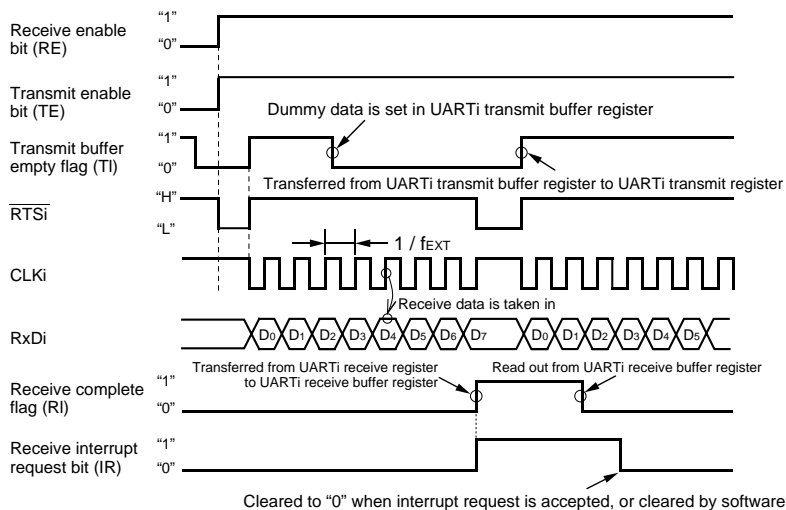


Figure 2.11.17 Typical transmit/receive timings in clock synchronous serial I/O mode

(1) Polarity select function

As shown in Figure 2.11.18, the CLK polarity select bit (bit 6 at addresses 03A416, 037C16) allows selection of the polarity of the transfer clock.

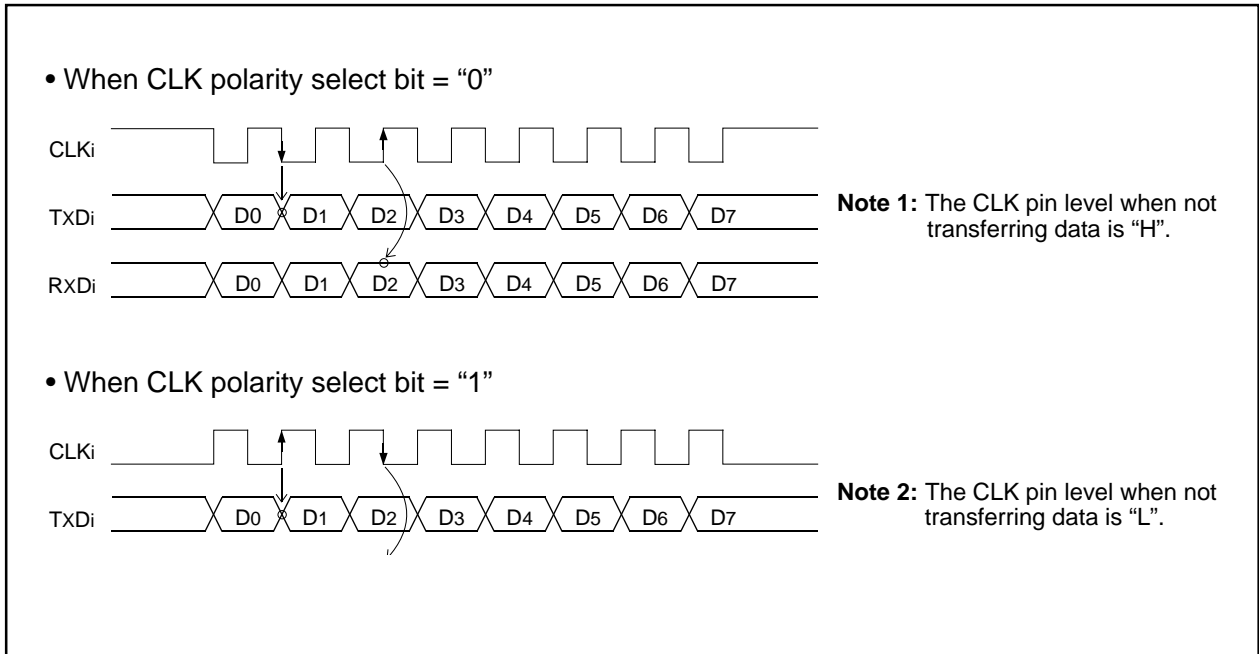


Figure 2.11.18 Polarity of transfer clock

(2) LSB first/MSB first select function

As shown in Figure 2.11.19, when the transfer format select bit (bit 7 at addresses 03A416, 037C16) = "0", the transfer format is "LSB first"; when the bit = "1", the transfer format is "MSB first".

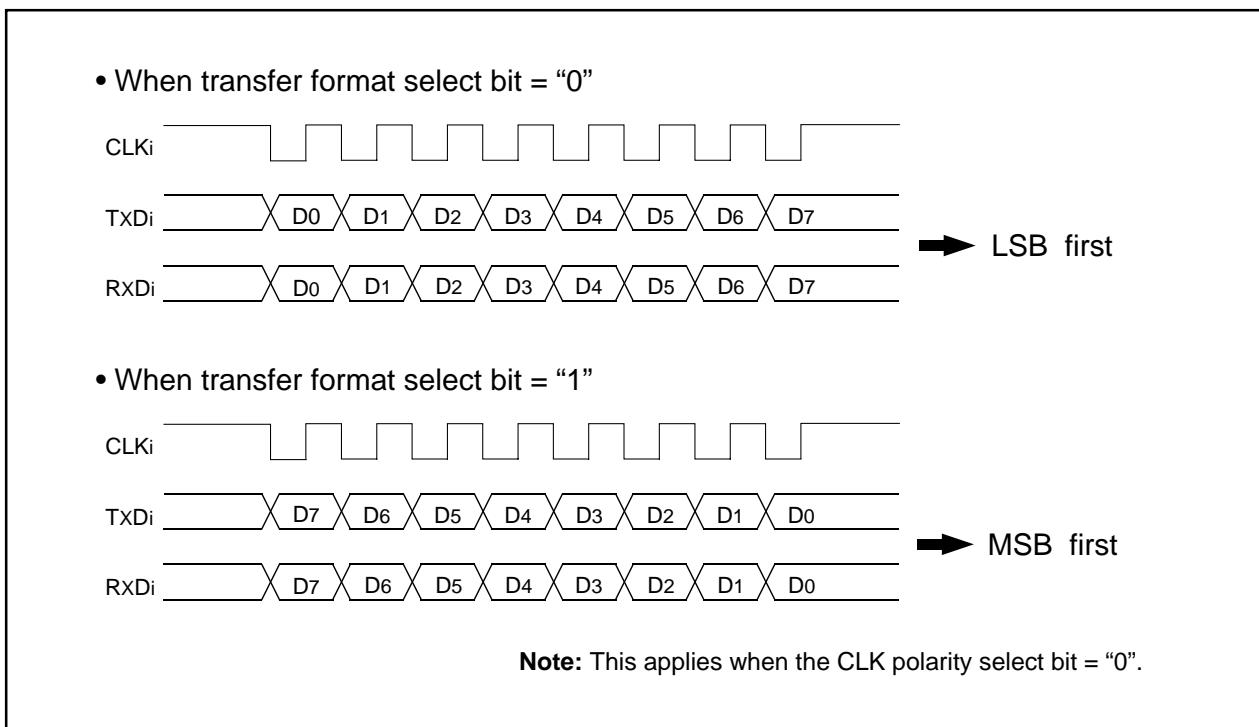


Figure 2.11.19 Transfer format

(3) Continuous receive mode

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

(4) Serial data logic switch function (UART2)

When the data logic select bit (bit6 at address 037D16) = "1", and writing to transmit buffer register or reading from receive buffer register, data is reversed. Figure 2.11.20 shows the example of serial data logic switch timing.

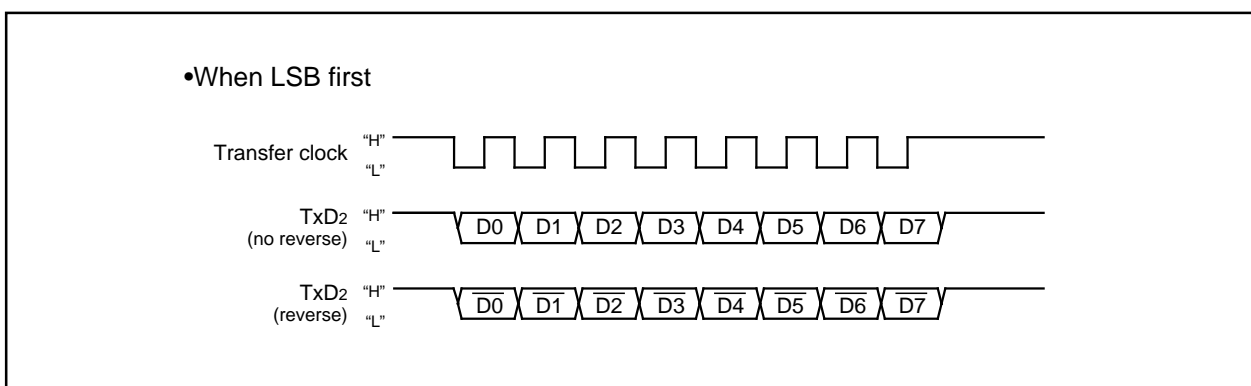


Figure 2.11.20 Serial data logic switch timing

2.11.3 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 2.11.5 and 2.11.6 list the specifications of the UART mode. Figure 2.11.21 and 2.11.22 show the UARTi transmit/receive mode register in UART mode.

Table 2.11.5 Specifications of UART Mode (1)

| Item | Specification |
|-------------------------------------|--|
| Transfer data format | <ul style="list-style-type: none"> • Character bit (transfer data): 7 bits, 8 bits, or 9 bits as selected • Start bit: 1 bit • Parity bit: Odd, even, or nothing as selected • Stop bit: 1 bit or 2 bits as selected |
| Transfer clock | <ul style="list-style-type: none"> • When internal clock is selected (bit 3 at addresses 03A0₁₆, 0378₁₆ = "0") : $f_i/16(n+1)$ (Note 1) $f_i = f_1, f_8, f_{32}$ • When external clock is selected (bit 3 at addresses 03A0₁₆, 0378₁₆ = "1") : $f_{EXT}/16(n+1)$(Note 1) (Note 2) |
| Transmission/reception control | <ul style="list-style-type: none"> • CTS function/RTS function/CTS, RTS function chosen to be invalid |
| Transmission start condition | <ul style="list-style-type: none"> • To start transmission, the following requirements must be met: <ul style="list-style-type: none"> - Transmit enable bit (bit 0 at addresses 03A5₁₆, 037D₁₆) = "1" - Transmit buffer empty flag (bit 1 at addresses 03A5₁₆, 037D₁₆) = "0" - When CTS function selected, \overline{CTS} input level = "L" |
| Reception start condition | <ul style="list-style-type: none"> • To start reception, the following requirements must be met: <ul style="list-style-type: none"> - Receive enable bit (bit 2 at addresses 03A5₁₆, 037D₁₆) = "1" - Start bit detection |
| Interrupt request generation timing | <ul style="list-style-type: none"> • When transmitting <ul style="list-style-type: none"> - Transmit interrupt cause select bits (bits 0 at address 03B0₁₆, bit4 at address 037D₁₆) = "0": Interrupts requested when data transfer from UARTi transfer buffer register to UARTi transmit register is completed - Transmit interrupt cause select bits (bits 0 at address 03B0₁₆, bit4 at address 037D₁₆) = "1": Interrupts requested when data transmission from UARTi transfer register is completed • When receiving <ul style="list-style-type: none"> - Interrupts requested when data transfer from UARTi receive register to UARTi receive buffer register is completed |
| Error detection | <ul style="list-style-type: none"> • Overrun error (Note 3) This error occurs when the next data is ready before contents of UARTi receive buffer register are read out • Framing error This error occurs when the number of stop bits set is not detected • Parity error 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 • Error sum flag This flag is set (= 1) when any of the overrun, framing, and parity errors is encountered |

Table 2.11.6 Specifications of UART Mode (2)

| Item | Specification |
|-----------------|---|
| Select function | <ul style="list-style-type: none"><li data-bbox="507 398 1426 510">• Sleep mode selection (UART0) This mode is used to transfer data to and from one of multiple slave micro-computers<li data-bbox="507 517 1426 629">• Serial data logic switch (UART2) This function is reversing logic value of transferring data. Start bit, parity bit and stop bit are not reversed.<li data-bbox="507 636 1426 748">• TxD, RxD I/O polarity switch This function is reversing TxD port output and RxD port input. All I/O data level is reversed. |

Notes 1: 'n' denotes the value 00₁₆ to FF₁₆ that is set to the UART_i bit rate generator.

2: f_{EXT} is input from the CLK_i pin.

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

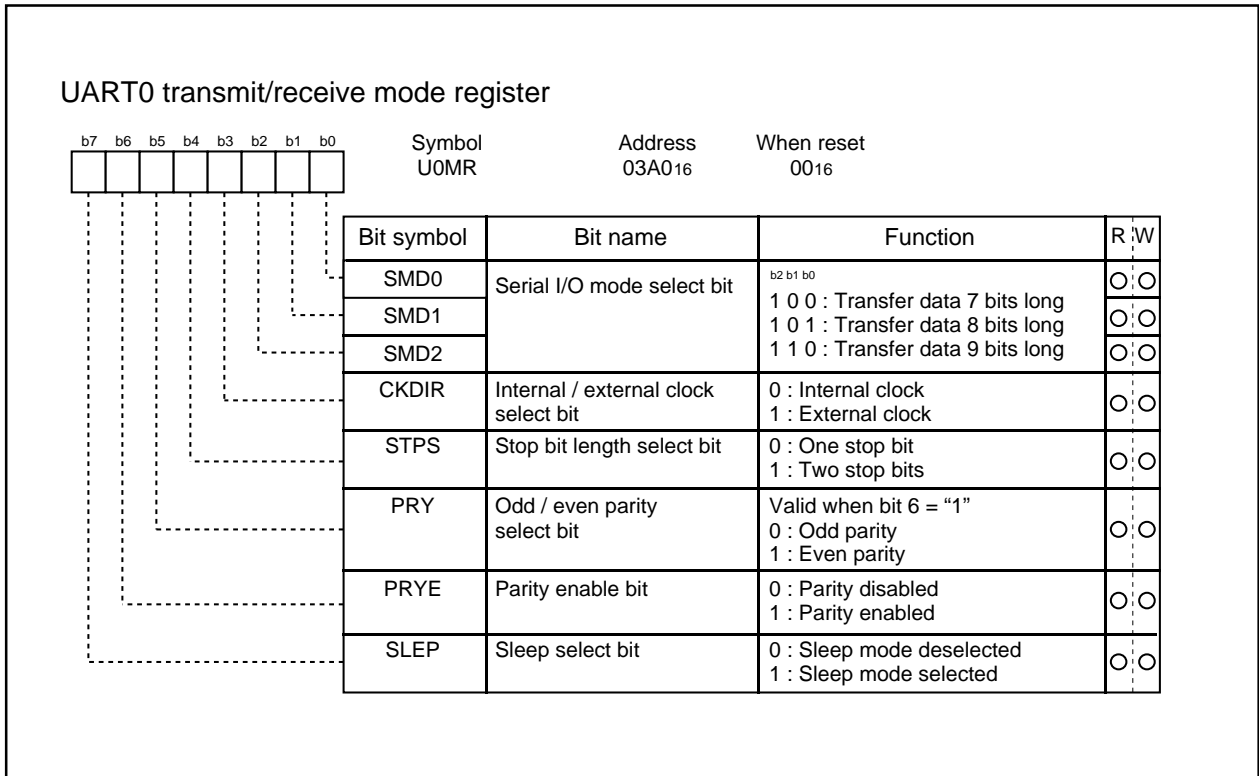


Figure 2.11.21 UART0 transmit/receive mode register in UART mode

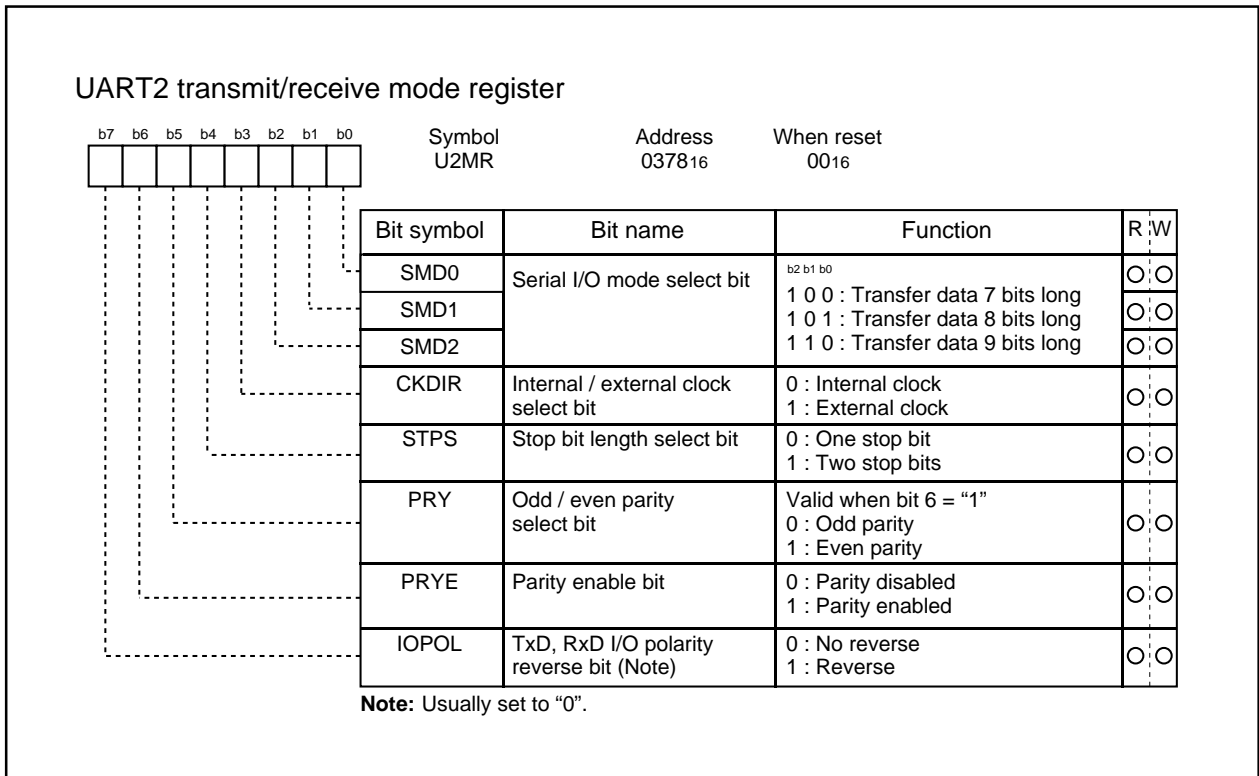


Figure 2.11.22 UART2 transmit/receive mode register in UART mode

Table 2.11.7 lists the functions of the input/output pins during UART mode. 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 2.11.7 Input/output pin functions in UART mode

| Pin name | Function | Method of selection |
|---|-------------------------------|--|
| TxDi (P63, P70) | Serial data output | |
| RxDi (P62, P71) | Serial data input | Port P62 and P71 direction register (bit 2 at address 03EE ₁₆ , bit 1 at address 03EF ₁₆) = "0" (Can be used as an input port when performing transmission only) |
| CLKi (P61, P72) | Programmable I/O port | Internal/external clock select bit (bit 3 at address 03A0 ₁₆ , 0378 ₁₆) = "0" |
| | Transfer clock input | Internal/external clock select bit (bit 3 at address 03A0 ₁₆ , 0378 ₁₆) = "1" Port P61 and P72 direction register (bit 1 at address 03EE ₁₆ , bit 2 at address 03EF ₁₆) = "0" |
| $\overline{\text{CTS}}/\overline{\text{RTS}}$ (P60, P73) | $\overline{\text{CTS}}$ input | $\overline{\text{CTS}}/\overline{\text{RTS}}$ disable bit (bit 4 at address 03A4 ₁₆ , 037C ₁₆) = "0" $\overline{\text{CTS}}/\overline{\text{RTS}}$ function select bit (bit 2 at address 03A4 ₁₆ , 037C ₁₆) = "0" Port P60 and P73 direction register (bit 0 at address 03EE ₁₆ , bit 3 at address 03EF ₁₆) = "0" |
| | RTS output | $\overline{\text{CTS}}/\overline{\text{RTS}}$ disable bit (bit 4 at address 03A4 ₁₆ , 037C ₁₆) = "0" $\overline{\text{CTS}}/\overline{\text{RTS}}$ function select bit (bit 2 at address 03A4 ₁₆ , 037C ₁₆) = "1" |
| | Programmable I/O port | $\overline{\text{CTS}}/\overline{\text{RTS}}$ disable bit (bit 4 at address 03A4 ₁₆ , 037C ₁₆) = "1" |

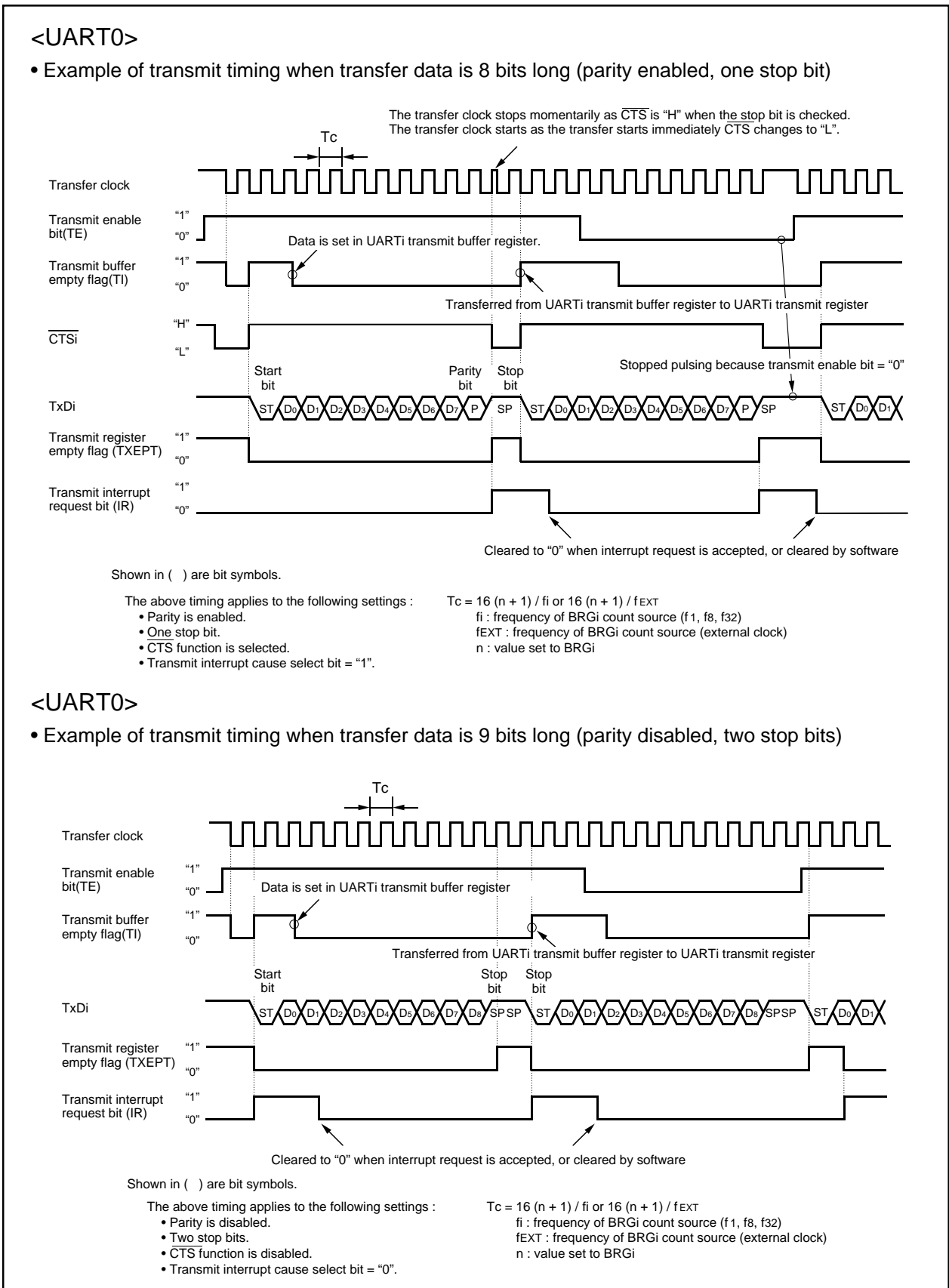


Figure 2.11.23 Typical transmit/receive timings in UART mode

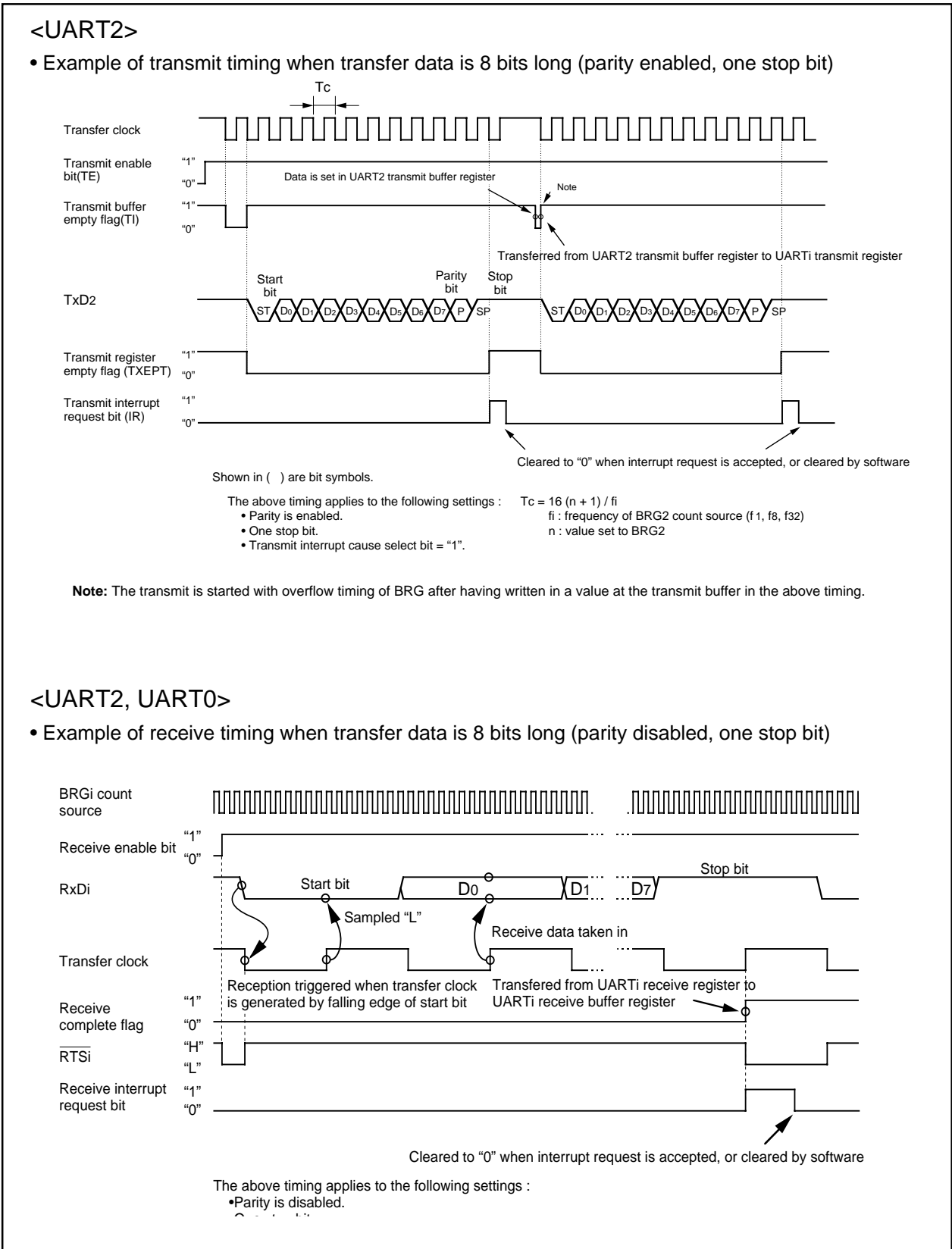


Figure 2.11.23 Typical transmit/receive timings in UART mode

(1) Sleep mode (UART0)

This mode is used to transfer data between specific microcomputers among multiple microcomputers connected using UART0. The sleep mode is selected when the sleep select bit (bit 7 at address 03A016) 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".

(2) 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 2.11.24 shows the example of timing for switching serial data logic.

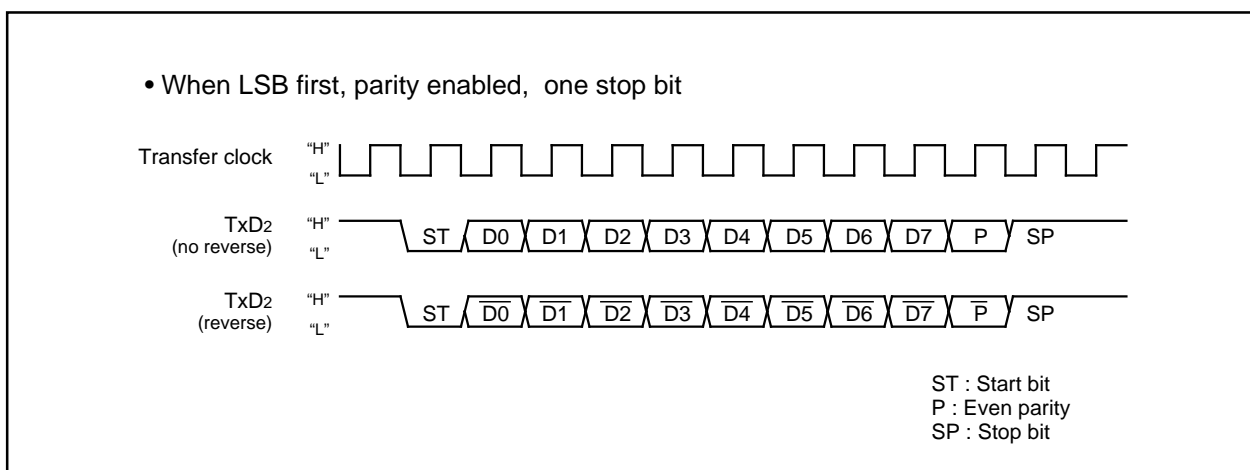


Figure 2.11.24 Timing for switching serial data logic

(3) 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.

(4) Bus collision detection function and other functions (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 2.11.26 shows the example of detection timing of a buss collision (in UART mode).

And also, bit 5 of the special UART2 mode register is used as the selection bit for auto clear function select bit of enable bit. Setting this bit to "1" automatically resets the transmit enable bit to "0" when "1" is set in the bus collision detection interrupt request bit (nonconformity) (refer to Figure 2.11.25).

Bit 6 of the special UART2 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 (refer to Figure 2.11.26).

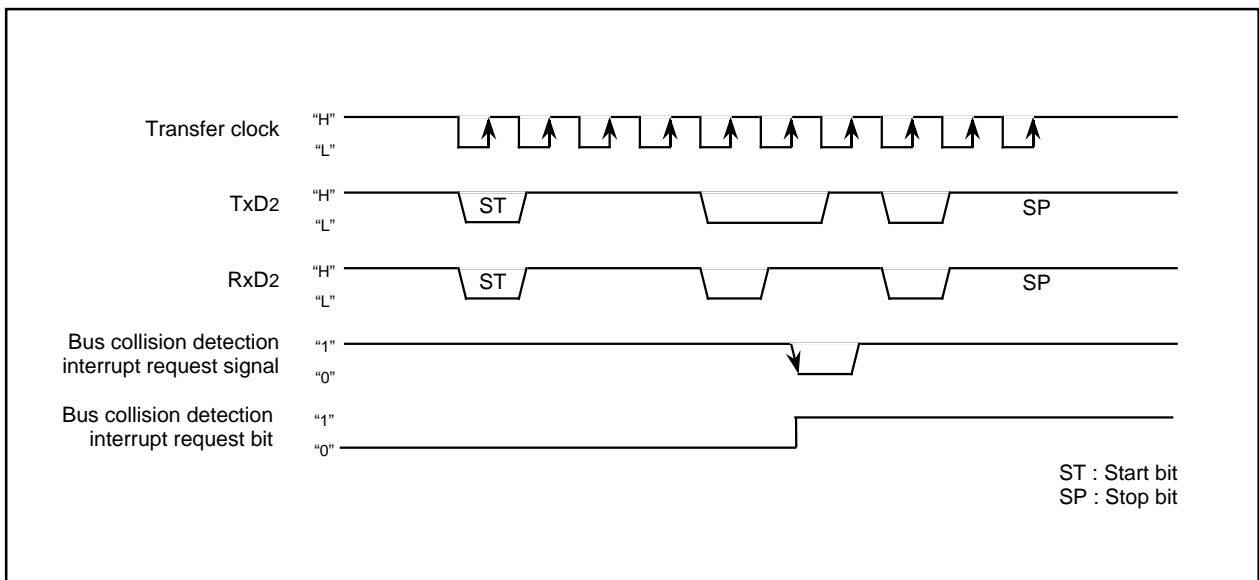


Figure 2.11.25 Detection timing of a bus collision (in UART mode)

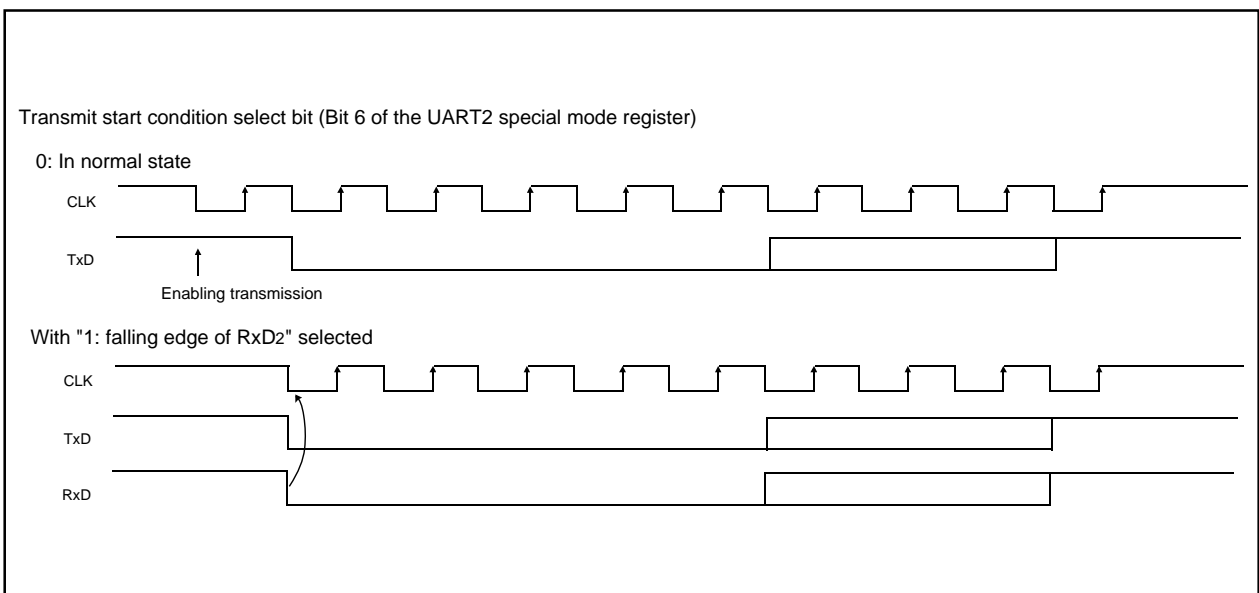


Figure 2.11.26 Some other functions

2.11.4 Clock-asynchronous Serial I/O Mode (Compliant with the SIM Interface)

The SIM interface is used for connecting the microcomputer with a memory card I/C or the like; adding some extra settings in UART2 clock-asynchronous serial I/O mode allows the user to effect this function. Tables 2.11.8 and 2.11.9 show the specifications of clock-asynchronous serial I/O mode (compliant with the SIM interface).

Table 2.11.8 Specifications of clock-asynchronous serial I/O mode (compliant with the SIM interface) (1)

| Item | Specification |
|-------------------------------------|---|
| Transfer data format | <ul style="list-style-type: none"> • Transfer data 8-bit UART mode (bit 2 through bit 0 of address 0378₁₆ = "1012") • One stop bit (bit 4 of address 0378₁₆ = "0") • With the direct format chosen <ul style="list-style-type: none"> Set parity to "even" (bit 5 and bit 6 of address 0378₁₆ = "1" and "1" respectively) Set data logic to "direct" (bit 6 of address 037D₁₆ = "0"). Set transfer format to LSB (bit 7 of address 037C₁₆ = "0"). • With the inverse format chosen <ul style="list-style-type: none"> Set parity to "odd" (bit 5 and bit 6 of address 0378₁₆ = "0" and "1" respectively) Set data logic to "inverse" (bit 6 of address 037D₁₆ = "1") Set transfer format to MSB (bit 7 of address 037C₁₆ = "1") |
| Transfer clock | <ul style="list-style-type: none"> • With the internal clock chosen (bit 3 of address 0378₁₆ = "0") : $f_i / 16 (n + 1)$ (Note 1) : $f_i = f_1, f_8, f_{32}$ • With an external clock chosen (bit 3 of address 0378₁₆ = "1") : $f_{EXT} / 16 (n+1)$ (Note 1) (Note 2) |
| Transmission / reception control | <ul style="list-style-type: none"> • Disable the \overline{CTS} and \overline{RTS} function (bit 4 of address 037C₁₆ = "1") |
| Other settings | <ul style="list-style-type: none"> • The sleep mode select function is not available for UART2 • Set transmission interrupt factor to "transmission completed" (bit 4 of address 037D₁₆ = "1") |
| Transmission start condition | <ul style="list-style-type: none"> • To start transmission, the following requirements must be met: <ul style="list-style-type: none"> - Transmit enable bit (bit 0 of address 037D₁₆) = "1" - Transmit buffer empty flag (bit 1 of address 037D₁₆) = "0" |
| Reception start condition | <ul style="list-style-type: none"> • To start reception, the following requirements must be met: <ul style="list-style-type: none"> - Reception enable bit (bit 2 of address 037D₁₆) = "1" - Detection of a start bit |
| Interrupt request generation timing | <ul style="list-style-type: none"> • When transmitting <ul style="list-style-type: none"> When data transmission from the UART2 transfer register is completed (bit 4 of address 037D₁₆ = "1") • When receiving <ul style="list-style-type: none"> When data transfer from the UART2 receive register to the UART2 receive buffer register is completed |

Table 2.11.9 Specifications of clock-asynchronous serial I/O mode (compliant with the SIM interface) (2)

| Item | Specification |
|-----------------|---|
| Error detection | <ul style="list-style-type: none"> • Overrun error (see the specifications of clock-asynchronous serial I/O) (Note 3) • Framing error (see the specifications of clock-asynchronous serial I/O) • Parity error (see the specifications of clock-asynchronous serial I/O) <ul style="list-style-type: none"> - On the reception side, an "L" level is output from the TxD2 pin by use of the parity error signal output function (bit 7 of address 037D16 = "1") when a parity error is detected - On the transmission side, a parity error is detected by the level of input to the RxD2 pin when a transmission interrupt occurs • The error sum flag (see the specifications of clock-asynchronous serial I/O) |

Notes 1: 'n' denotes the value 00₁₆ to FF₁₆ that is set to the UARTi bit rate generator.

2: f_{EXT} is input from the CLK2 pin.

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".

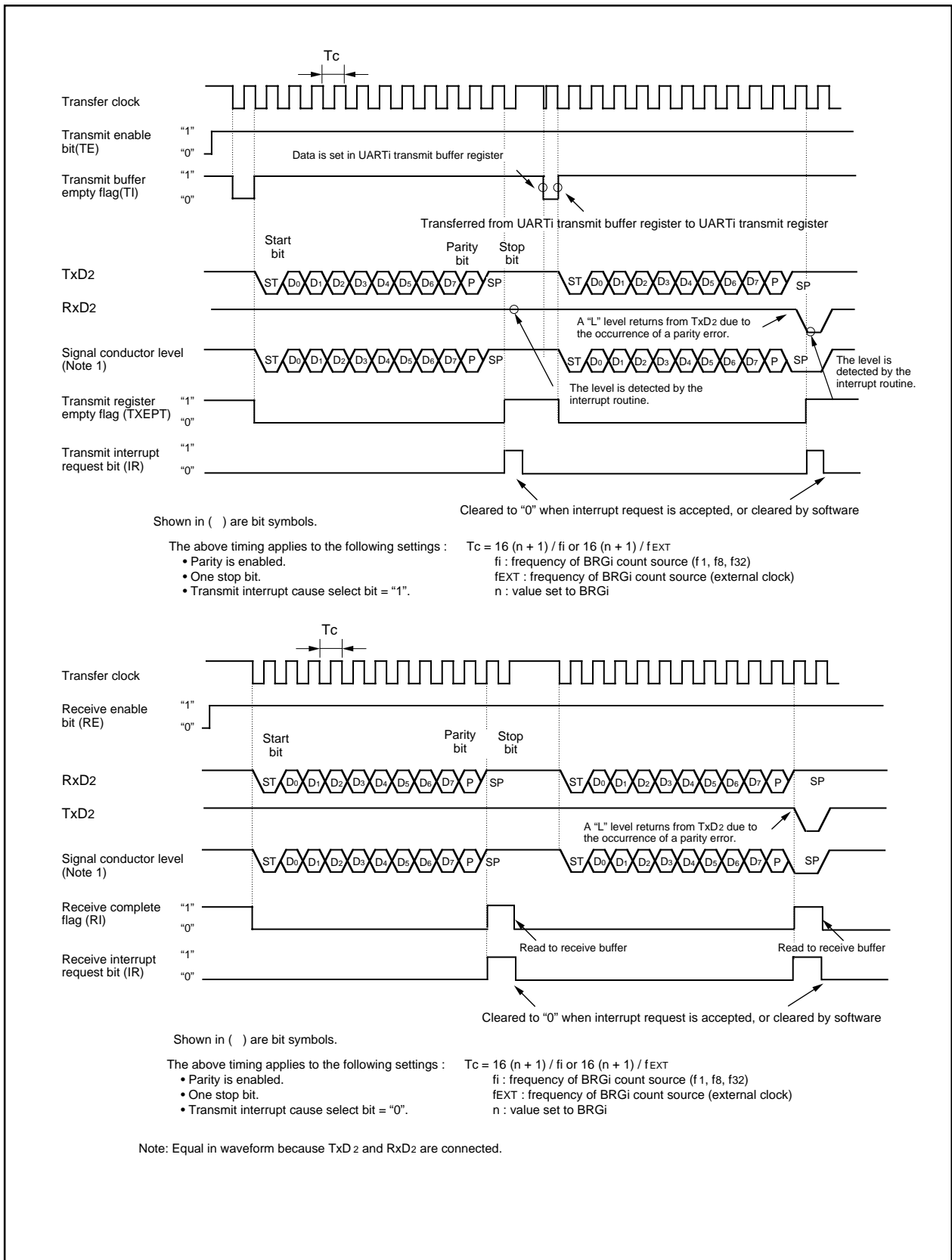


Figure 2.11.27 Typical transmit/receive timing in UART mode (compliant with the SIM interface)

(1) 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 2.11.28 shows the output timing of the parity error signal.

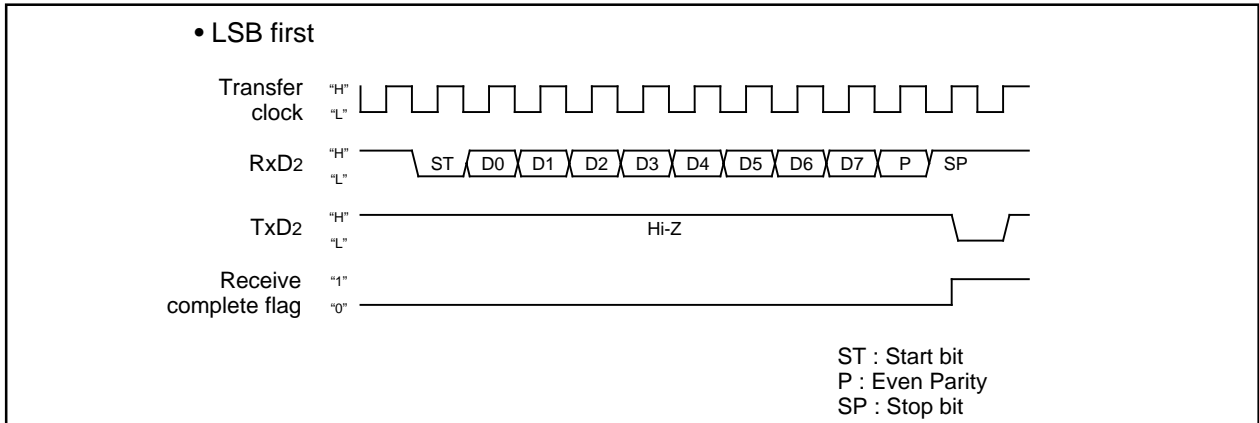


Figure 2.11.28 Output timing of the parity error signal

(2) 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 2.11.29 shows the SIM interface format.

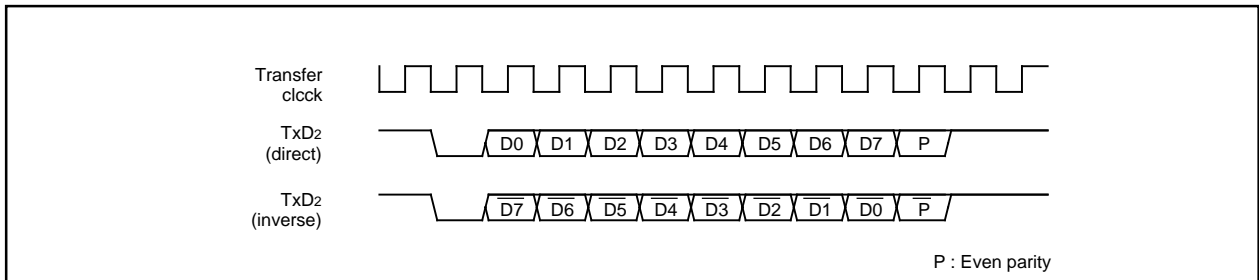


Figure 2.11.29 SIM interface format

Figure 2.11.30 shows the example of connecting the SIM interface. Connect TxD2 and RxD2 and apply pull-up.

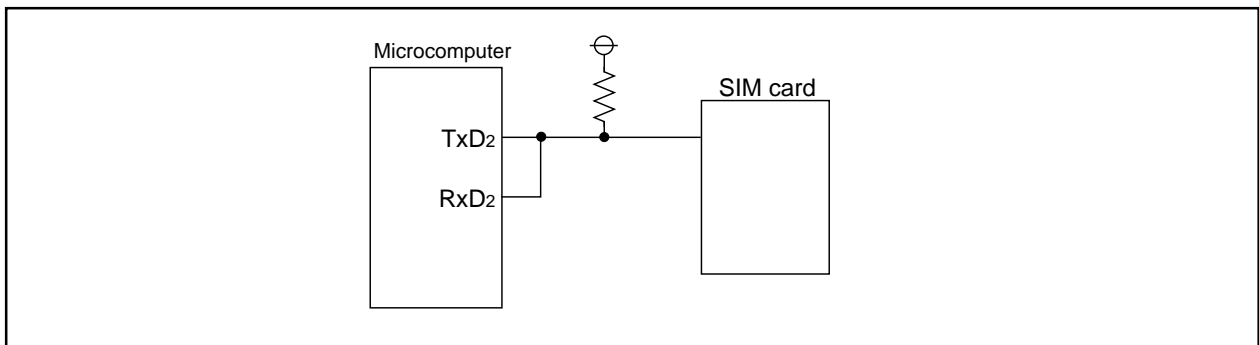


Figure 2.11.30 Connecting the SIM interface

2.11.5 Serial Interface Ports

The I/O ports (P67, P70 to P72) function as I/O ports of UART2 and multi-master I²C-BUS interface 0 (refer to "2.11.6 Multi-master I²C-BUS interface i"). Set the connection between both serial interfaces and each port by bits 0 and 1 (BSEL0 and BSEL1) of the peripheral mode register (address 027D16) and bit 2 (FIICON) of the I²C0 port selection register (address 02E516).

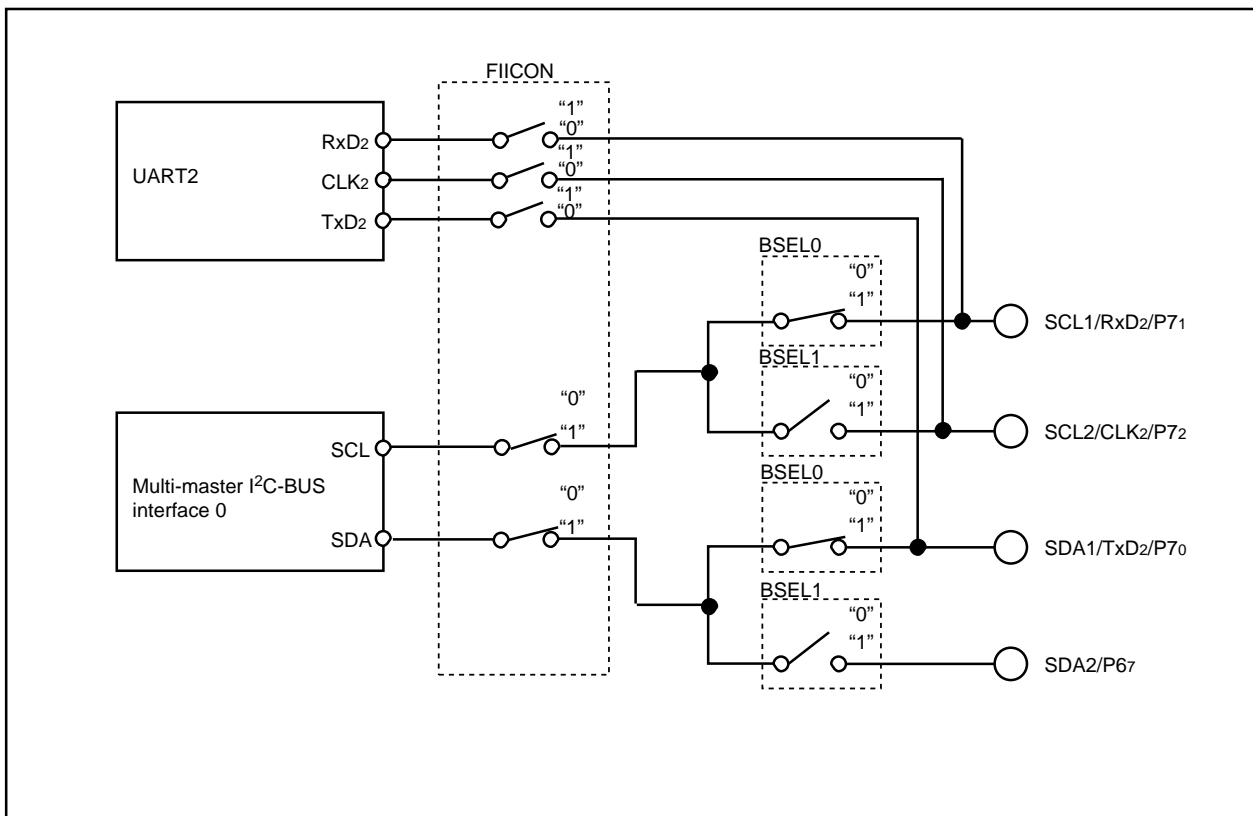


Figure 2.11.31 Serial interface port control

2.11.6 Multi-master I²C-BUS Interface 0 and Multi-master I²C-BUS Interface 1

The multi-master I²C-BUS interface 0 and 1 have each dedicated circuit and operate independently.

The multi-master I²C-BUS interface *i* is a serial communications circuit, conforming to the Philips I²C-BUS data transfer format. This interface *i*, offering both arbitration lost detection and a synchronous functions, is useful for the multi-master serial communications.

Figures 2.11.32 and 2.11.33 show a block diagram of the multi-master I²C-BUS interface *i* and Table 2.11.13 shows multi-master I²C-BUS interface *i* functions.

This multi-master I²C-BUS interface *i* consists of the I²C_i address register, the I²C_i data shift register, the I²C_i clock control register, the I²C_i control register, the I²C_i status register, the I²C_i port selection register and other control circuits.

Table 2.11.13 Multi-master I²C-BUS Interface Functions

| Item | Function |
|----------------------|--|
| Format | In conformity with Philips I ² C-BUS standard: 10-bit addressing format 7-bit addressing format High-speed clock mode Standard clock mode |
| Communication mode | In conformity with Philips I ² C-BUS standard: Master transmission Master reception Slave transmission Slave reception |
| SCL clock frequencyn | 16.1 kHz to 400 kHz (at BCLK = 10 MHz) |

Note : We are not responsible for any third party's infringement of patent rights or other rights attributable to the use of the control function (bits 6 and 7 of the I²C control register at address 027D16) for connections between the I²C-BUS interface 0 and ports (SCL1, SCL2, SDA1, SDA2).

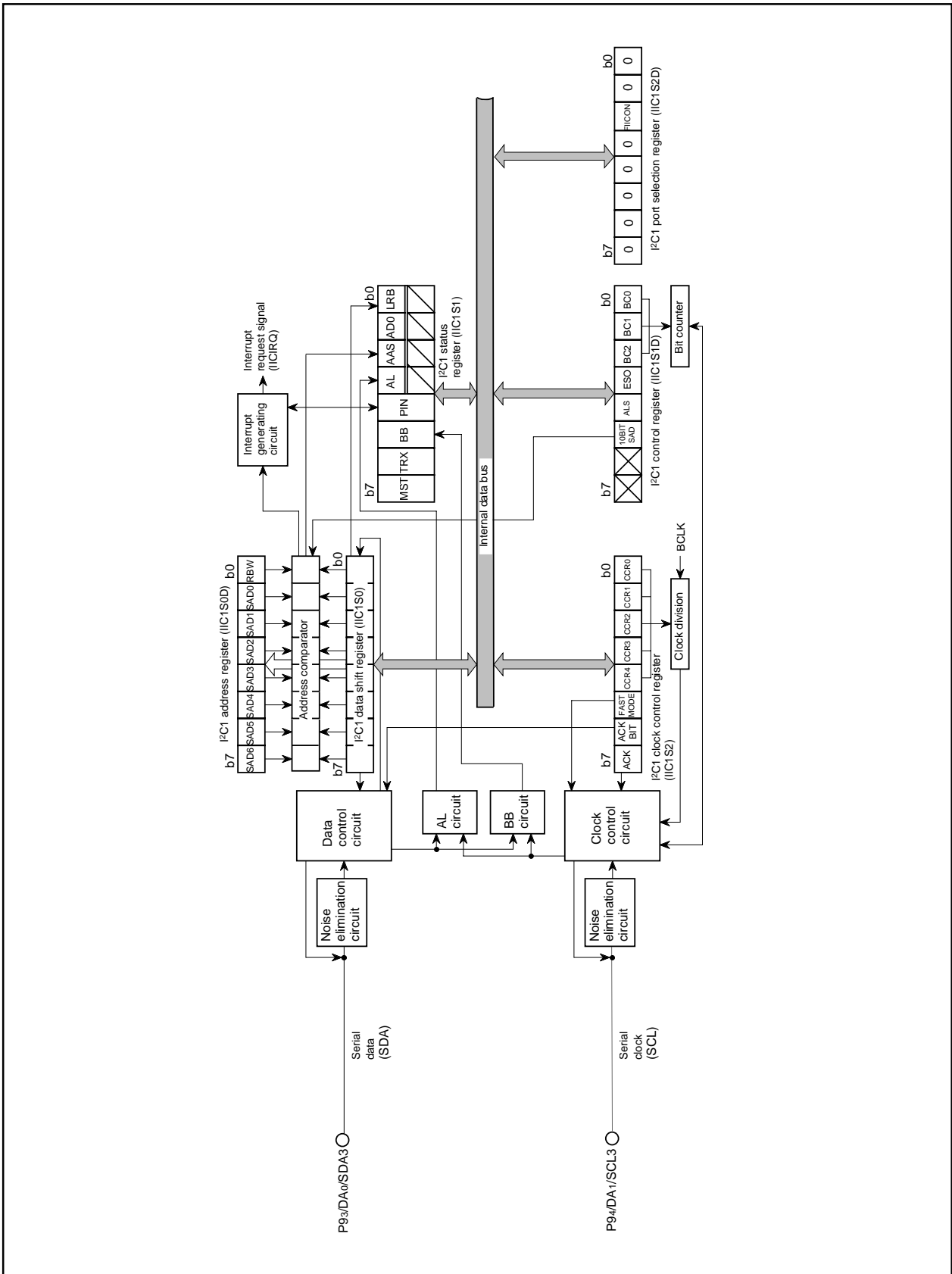


Fig. 2.11.33 Block Diagram of Multi-master I²C-BUS Interface 1

(1) I²C_i port selection register (i = 0, 1)

The I²C_i port selection register consists of bit to validate the multi-master I²C-BUS interface i function.

■ Bit 2: Multi-master I²C-BUS interface valid bit (FIICON)

When this bit is “0,” the multi-master I²C-BUS interface i is nonactive; when “1,” it is active. When selecting active, multi-master I²C-BUS interface 0 is connected with the ports selected by bits 0 and 1 of the peripheral mode register (address 027D₁₆) and multi-master I²C-BUS interface 1 is connected with the ports P9₃ and P9₄.

Note: It needs 10-BCLK cycles from setting this bit to “1” to being active of multi-master I²C-BUS interface i. Accordingly, do not access multi-master I²C-BUS interface i-related registers in this period.

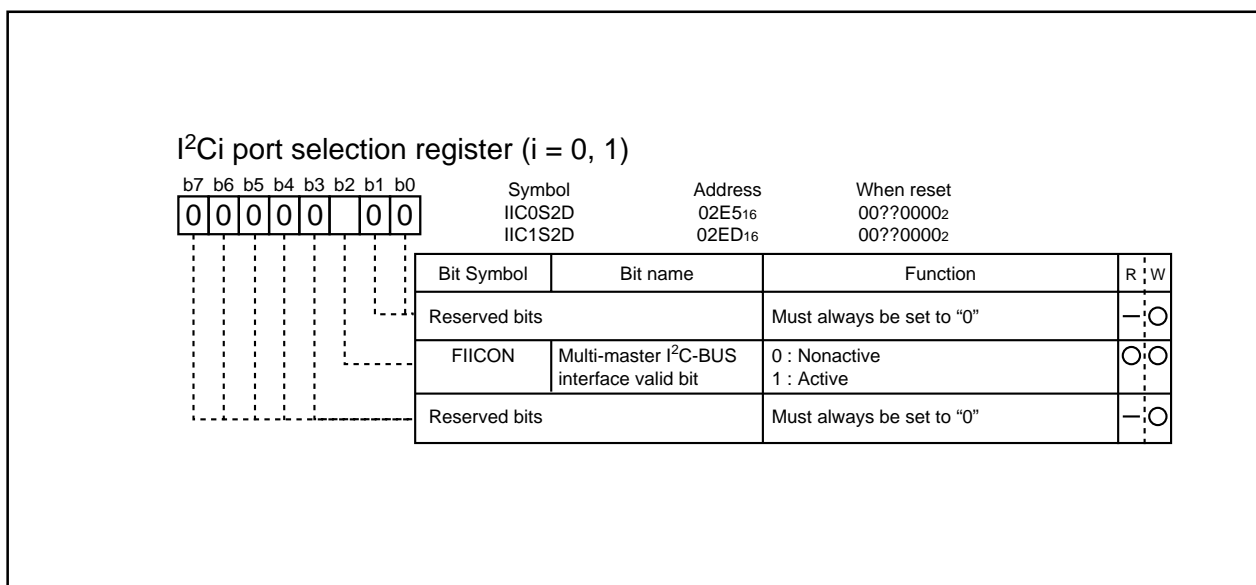


Fig. 2.11.34 I²C_i port selection register (i = 0, 1)

(2) I²Ci data shift register, I²Ci transmit buffer register (i = 0, 1)

The I²Ci data shift register is an 8-bit shift register to store receive data and write transmit data.

When transmit data is written into this register, it is transferred to the outside from bit 7 in synchronization with the SCL clock, and each time one-bit data is output, the data of this register are shifted one bit to the left. When data is received, it is input to this register from bit 0 in synchronization with the SCL clock, and each time one-bit data is input, the data of this register are shifted one bit to the left.

The I²Ci data shift register is in a write enable status only when the ESO bit of the I²Ci control register is "1." The bit counter is reset by a write instruction to the I²Ci data shift register. When both the ESO bit and the MST bit of the I²Ci status register are "1," the SCL is output by a write instruction to the I²Ci data shift register. Reading data from the I²Ci data shift register is always enabled regardless of the ESO bit value.

The I²Ci transmit buffer register is a register to store transmit data (slave address) to the I²Ci data shift register before RESTART condition generation. That is, in master, transmit data written to the I²Ci transmit buffer register is written to the I²Ci data shift register simultaneously. However, the SCL is not output. The I²Ci transmit buffer register can be written only when the ESO bit is "1," reading data from the I²Ci transmit buffer register is disabled regardless of the ESO bit value.

Notes 1: To write data into the I²Ci data shift register or the I²Ci transmit buffer register after the MST bit value changes from "1" to "0" (slave mode), keep an interval of 20 BCLK or more.

2: To generate START/RESTART condition after the I²Ci data shift register or the I²Ci transmit buffer register is written, keep an interval of 2 BCLK or more.

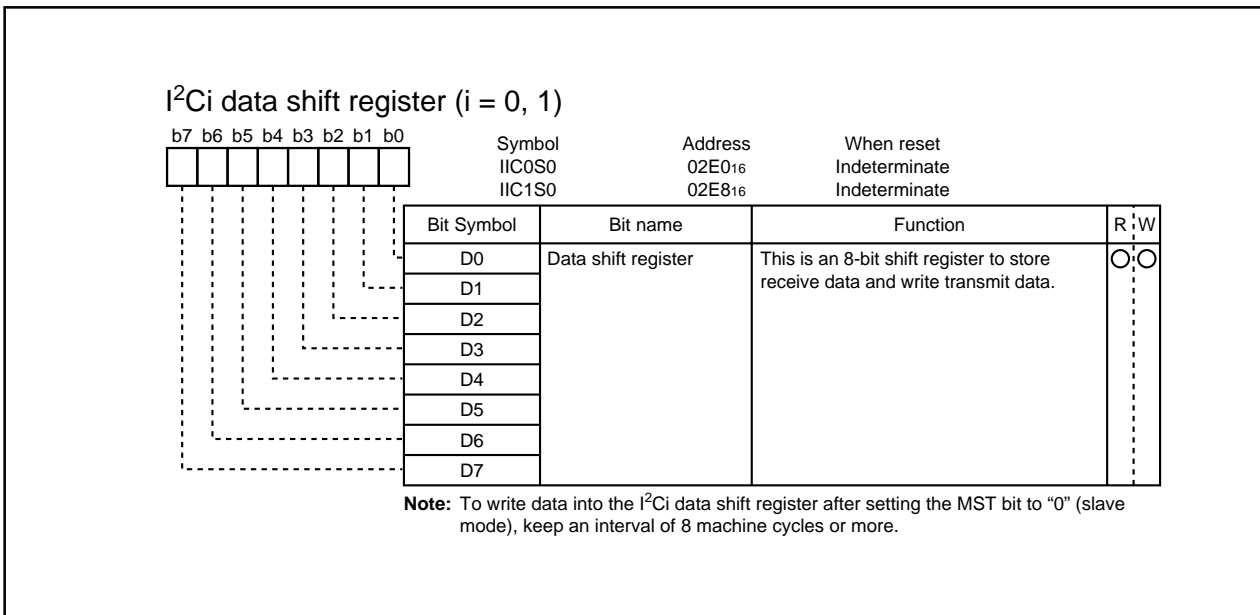


Fig. 2.11.35 I²Ci data shift register (i = 0, 1)

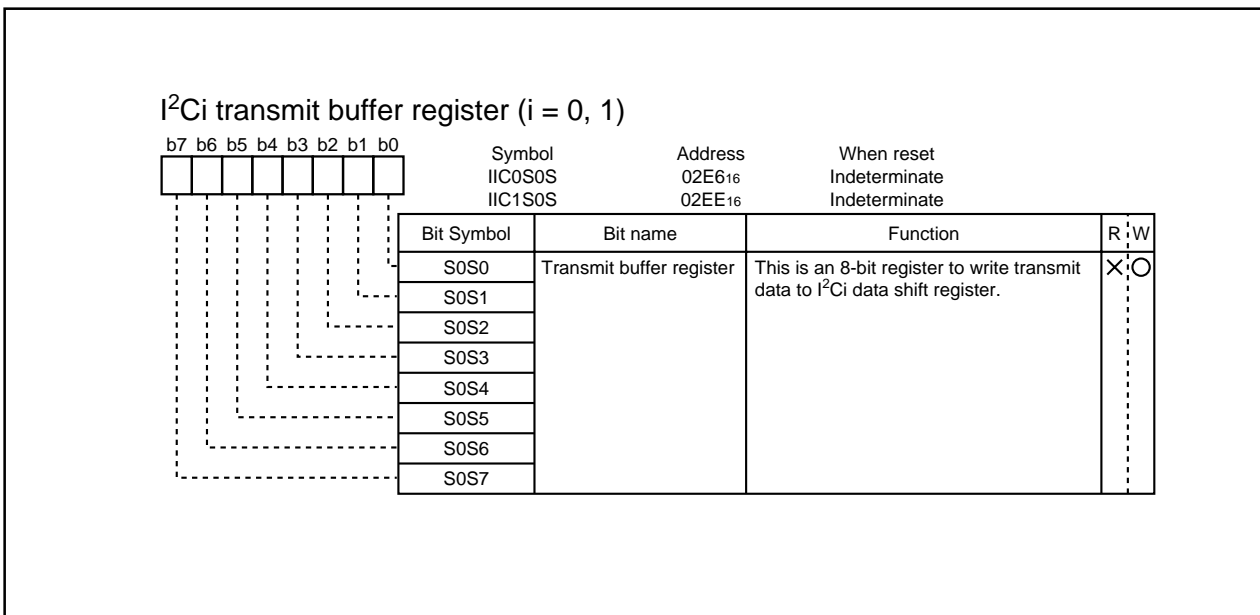


Fig. 2.11.36 I²Ci transmit buffer register (i = 0, 1)

(3) I²Ci address register (i = 0, 1)

The I²Ci address register consists of a 7-bit slave address and a $\overline{\text{read/write}}$ bit. In the addressing mode, the slave address written in this register is compared with the address data to be received immediately after the START condition are detected.

■ Bit 0: $\overline{\text{read/write}}$ bit (RBW)

Not used when comparing addresses, in the 7-bit addressing mode. In the 10-bit addressing mode, the first address data to be received is compared with the contents (SAD6 to SAD0 + RBW) of the I²Ci address register.

The RBW bit is cleared to “0” automatically when the stop condition is detected.

■ Bits 1 to 7: slave address (SAD0–SAD6)

These bits store slave addresses. Regardless of the 7-bit addressing mode and the 10-bit addressing mode, the address data transmitted from the master is compared with the contents of these bits.

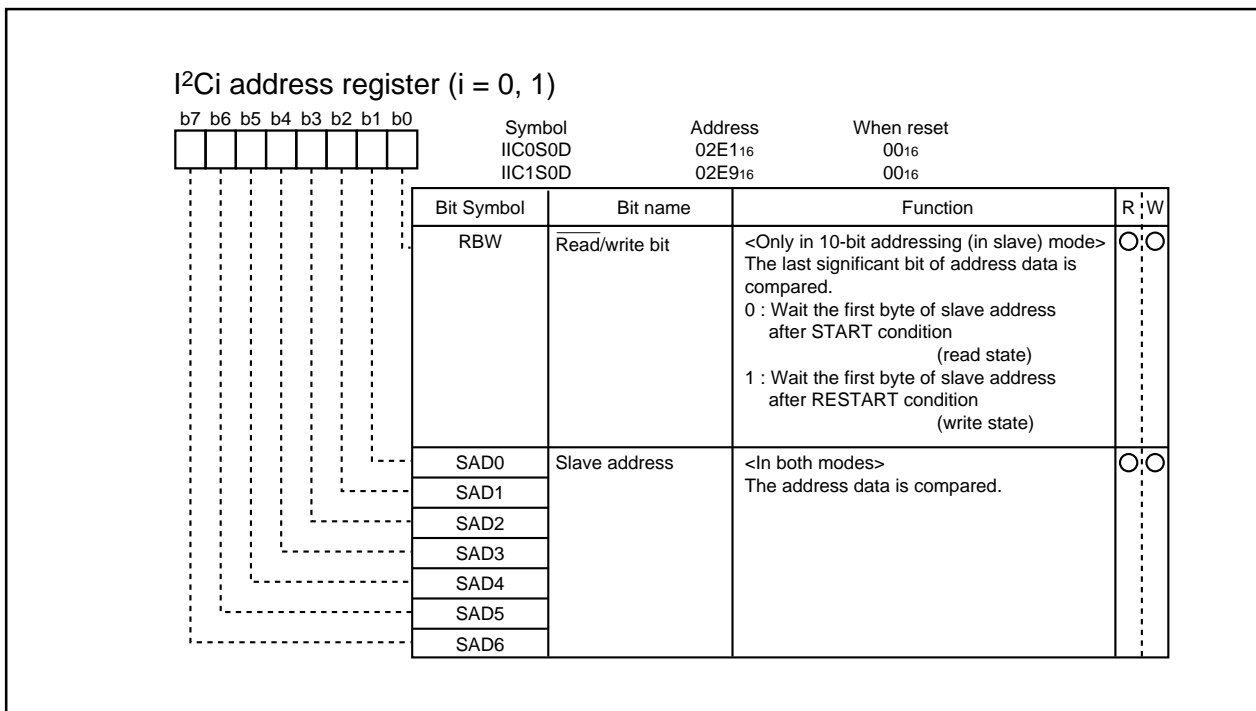


Fig. 2.11.37 I²Ci address register (i = 0, 1)

(4) I²Ci clock control register (i = 0, 1)

The I²Ci clock control register is used to set ACK control, SCL mode and SCL frequency.

■ Bits 0 to 4: SCL frequency control bits (CCR0–CCR4)

These bits control the SCL frequency.

■ Bit 5: SCL mode specification bit (FAST MODE)

This bit specifies the SCL mode. When this bit is set to “0,” the standard clock mode is set. When the bit is set to “1,” the high-speed clock mode is set.

■ Bit 6: ACK bit (ACK BIT)

This bit sets the SDA status when an ACK clock* is generated. When this bit is set to “0,” the ACK return mode is set and SDA goes to LOW at the occurrence of an ACK clock. When the bit is set to “1,” the ACK non-return mode is set. The SDA is held in the HIGH status at the occurrence of an ACK clock.

However, when the slave address matches the address data in the reception of address data at ACK BIT = “0,” the SDA is automatically made LOW (ACK is returned). If there is a mismatch between the slave address and the address data, the SDA is automatically made HIGH (ACK is not returned).

*ACK clock: Clock for acknowledgement

■ Bit 7: ACK clock bit (ACK)

This bit specifies a mode of acknowledgment which is an acknowledgment response of data transmission. When this bit is set to “0,” the no ACK clock mode is set. In this case, no ACK clock occurs after data transmission. When the bit is set to “1,” the ACK clock mode is set and the master generates an ACK clock upon completion of each 1-byte data transmission. The device for transmitting address data and control data releases the SDA at the occurrence of an ACK clock (make SDA HIGH) and receives the ACK bit generated by the data receiving device.

Note: Do not write data into the I²Ci clock control register during transmission. If data is written during transmission, the I²Ci clock generator is reset, so that data cannot be transmitted normally.

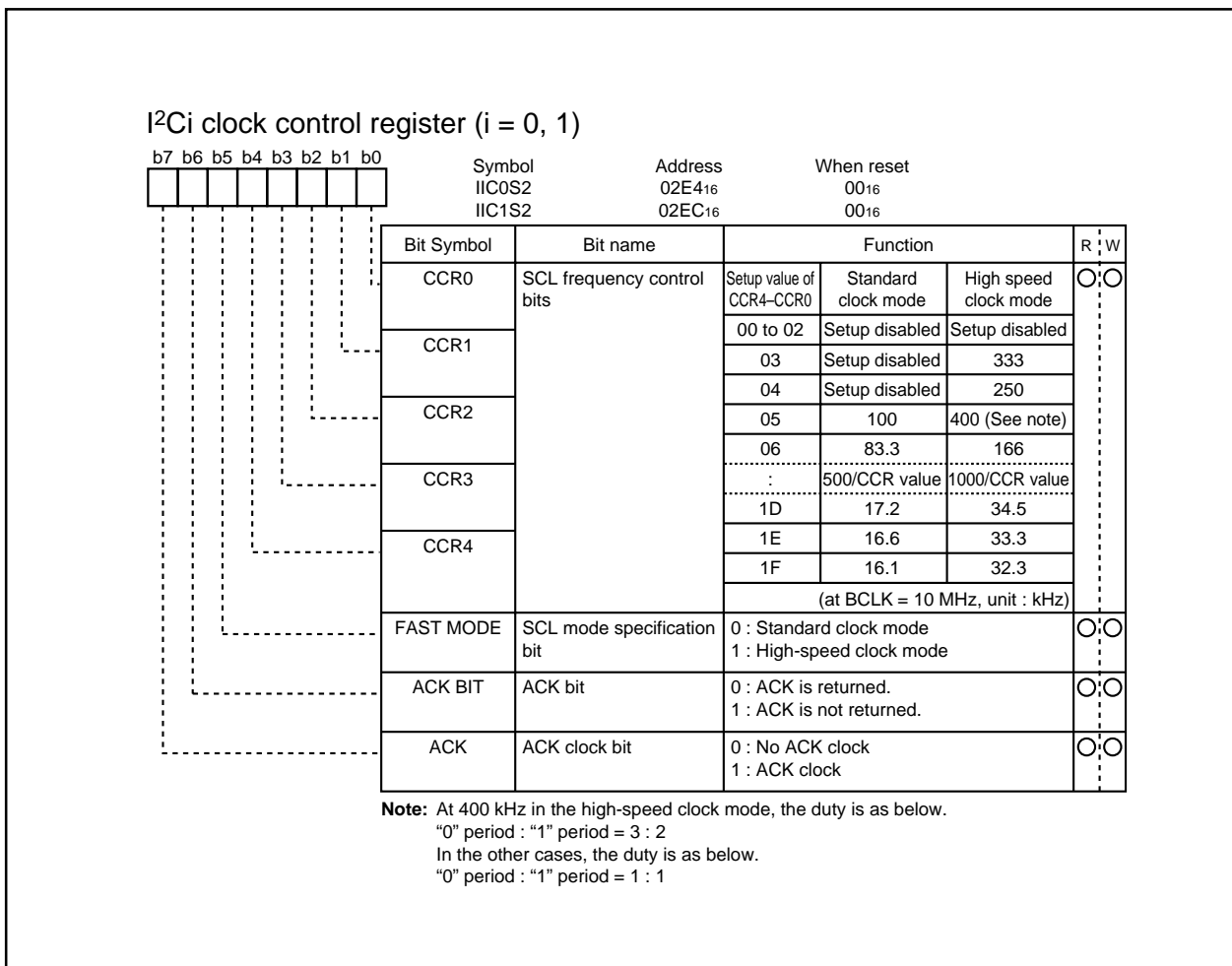


Fig. 2.11.38 I²Ci clock control register (i = 0, 1)

(5) I²Ci control register (i = 0, 1)

The I²Ci control register controls the data communication format.

■ Bits 0 to 2: bit counter (BC0–BC2)

These bits decide the number of bits for the next 1-byte data to be transmitted. An interrupt request signal occurs immediately after the number of bits specified with these bits are transmitted.

When a START condition is received, these bits become “0002” and the address data is always transmitted and received in 8 bits.

Note: When the bit counter value = “1112,” a STOP condition and START condition cannot be waited.

■ Bit 3: I²C-BUS interface i use enable bit (ESO)

This bit enables usage of the multimaster I²C-BUS interface i. When this bit is set to “0,” the use disable status is provided, so the SDA and the SCL become high-impedance. When the bit is set to “1,” use of the interface is enabled.

When ESO = “0,” the following is performed.

- PIN = “1,” BB = “0” and AL = “0” are set (they are bits of the I²Ci status register).
- Writing data to the I²Ci data shift register and the I²Ci transmit buffer register is disabled.

■ Bit 4: data format selection bit (ALS)

This bit decides whether or not to recognize slave addresses. When this bit is set to “0,” the addressing format is selected, so that address data is recognized. When a match is found between a slave address and address data as a result of comparison or when a general call (refer to “(6) I²Ci status register,” bit 1) is received, transmission processing can be performed. When this bit is set to “1,” the free data format is selected, so that slave addresses are not recognized.

■ Bit 5: addressing format selection bit (10BIT SAD)

This bit selects a slave address specification format. When this bit is set to “0,” the 7-bit addressing format is selected. In this case, only the high-order 7 bits (slave address) of the I²Ci address register are compared with address data. When this bit is set to “1,” the 10-bit addressing format is selected, all the bits of the I²Ci address register are compared with address data.

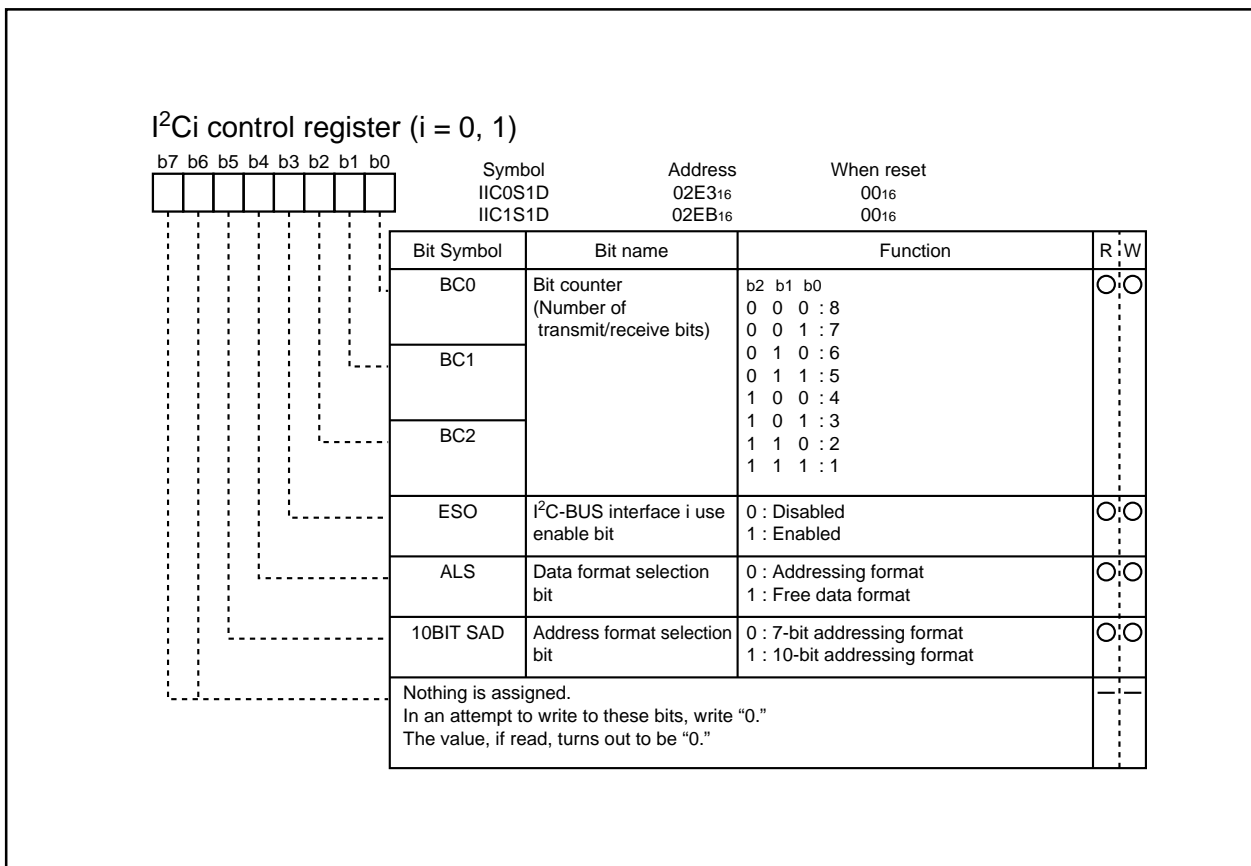


Fig. 2.11.39 I²Ci control register (i = 0, 1)

(6) I²Ci status register (i = 0, 1)

The I²Ci status register controls the I²C-BUS interface i status. Bits 0 to 3, 5 are read-only bits and bits 4, 6, 7 can be read out and written to.

■ Bit 0: last receive bit (LRB)

This bit stores the last bit value of received data and can also be used for ACK receive confirmation. If ACK is returned when an ACK clock occurs, the LRB bit is set to "0." If ACK is not returned, this bit is set to "1." Except in the ACK mode, the last bit value of received data is input. The state of this bit is changed from "1" to "0" by executing a write instruction to the I²Ci data shift register or the I²Ci transmit buffer register.

■ Bit 1: general call detecting flag (AD0)

This bit is set to "1" when a general call* whose address data is all "0" is received in the slave mode. By a general call of the master device, every slave device receives control data after the general call. The AD0 bit is set to "0" by detecting the STOP condition or START condition.

*General call: The master transmits the general call address "0016" to all slaves.

■ Bit 2: slave address comparison flag (AAS)

This flag indicates a comparison result of address data.

<<In the slave receive mode, when the 7-bit addressing format is selected, this bit is set to "1" in one of the following conditions.>>

- The address data immediately after occurrence of a START condition matches the slave address stored in the high-order 7 bits of the I²Ci address register.
- A general call is received.

<<In the slave reception mode, when the 10-bit addressing format is selected, this bit is set to "1" with the following condition.>>

- When the address data is compared with the I²Ci address register (8 bits consists of slave address and RBW), the first bytes match.

<<The state of this bit is changed from "1" to "0" by executing a write instruction to the I²Ci data shift register or the I²Ci transmit buffer register.>>

■ Bit 3: arbitration lost* detecting flag (AL)

In the master transmission mode, when a device other than the microcomputer sets the SDA to "L", arbitration is judged to have been lost, so that this bit is set to "1." At the same time, the TRX bit is set to "0," so that immediately after transmission of the byte whose arbitration was lost is completed, the MST bit is set to "0." When arbitration is lost during slave address transmission, the TRX bit is set to "0" and the reception mode is set. Consequently, it becomes possible to receive and recognize its own slave address transmitted by another master device.

*Arbitration lost: The status in which communication as a master is disabled.

■ Bit 4: I²C-BUS interface i interrupt request bit (PIN)

This bit generates an interrupt request signal. Each time 1-byte data is transmitted, the state of the PIN bit changes from “1” to “0.” At the same time, an interrupt request signal is sent to the CPU. The PIN bit is set to “0” in synchronization with a falling edge of the last clock (including the ACK clock) of an internal clock and an interrupt request signal occurs in synchronization with a falling edge of the PIN bit. When detecting the STOP condition in slave, the multi-master I²C-BUS interface interrupt request bit (IR) is set to “1” (interrupt requested) regardless of falling of PIN bit. When the PIN bit is “0,” the SCL is kept in the “0” state and clock generation is disabled. Figure 2.11.41 shows an interrupt request signal generating timing chart.

The PIN bit is set to “1” in any one of the following conditions.

- Writing “1” to the PIN bit
- Executing a write instruction to the I²Ci data shift register or the I²Ci transmit buffer register (See note).
- When the ESO bit is “0”
- At reset

Note: It takes 8 BCLK cycles or more until PIN bit becomes “1” after write instructions are executed to these registers.

The conditions in which the PIN bit is set to “0” are shown below:

- Immediately after completion of 1-byte data transmission (including when arbitration lost is detected)
- Immediately after completion of 1-byte data reception
- In the slave reception mode, with ALS = “0” and immediately after completion of slave address or general call address reception
- In the slave reception mode, with ALS = “1” and immediately after completion of address data reception

■ Bit 5: bus busy flag (BB)

This bit indicates the status of use of the bus system. When this bit is set to “0,” this bus system is not busy and a START condition can be generated. When this bit is set to “1,” this bus system is busy and the occurrence of a START condition is disabled by the START condition duplication prevention function (See note).

This flag can be written by software only in the master transmission mode. In the other modes, this bit is set to “1” by detecting a START condition and set to “0” by detecting a STOP condition. When the ESO bit of the I²Ci control register is “0” and at reset, the BB flag is kept in the “0” state.

■ Bit 6: communication mode specification bit (transfer direction specification bit: TRX)

This bit decides the direction of transfer for data communication. When this bit is “0,” the reception mode is selected and the data of a transmitting device is received. When the bit is “1,” the transmission mode is selected and address data and control data are output into the SDA in synchronization with the clock generated on the SCL.

When the ALS bit of the I²Ci control register is “0” in the slave reception mode is selected, the TRX bit is set to “1” (transmit) if the least significant bit (R/ \bar{W} bit) of the address data transmitted by the master is “1.” When the ALS bit is “0” and the R/ \bar{W} bit is “0,” the TRX bit is cleared to “0” (receive).

The TRX bit is cleared to “0” in one of the following conditions.

- When arbitration lost is detected.
- When a STOP condition is detected.
- When occurrence of a START condition is disabled by the START condition duplication prevention function (Note).
- With MST = “0” and when a START condition is detected.
- With MST = “0” and when ACK non-return is detected.
- At reset

■ **Bit 7: Communication mode specification bit (master/slave specification bit: MST)**

This bit is used for master/slave specification for data communication. When this bit is “0,” the slave is specified, so that a START condition and a STOP condition generated by the master are received, and data communication is performed in synchronization with the clock generated by the master. When this bit is “1,” the master is specified and a START condition and a STOP condition are generated, and also the clocks required for data communication are generated on the SCL.

The MST bit is cleared to “0” in one of the following conditions.

- Immediately after completion of 1-byte data transmission when arbitration lost is detected
- When a STOP condition is detected.
- When occurrence of a START condition is disabled by the START condition duplication preventing function (See note).
- At reset

Note: The START condition duplication prevention function disables the following: the START condition generation; bit counter reset, and SCL output with the generation. This bit is valid from setting of BB flag to the completion of 1-byte transmission/reception (occurrence of transmission/reception interrupt request) <IICIRQ>.

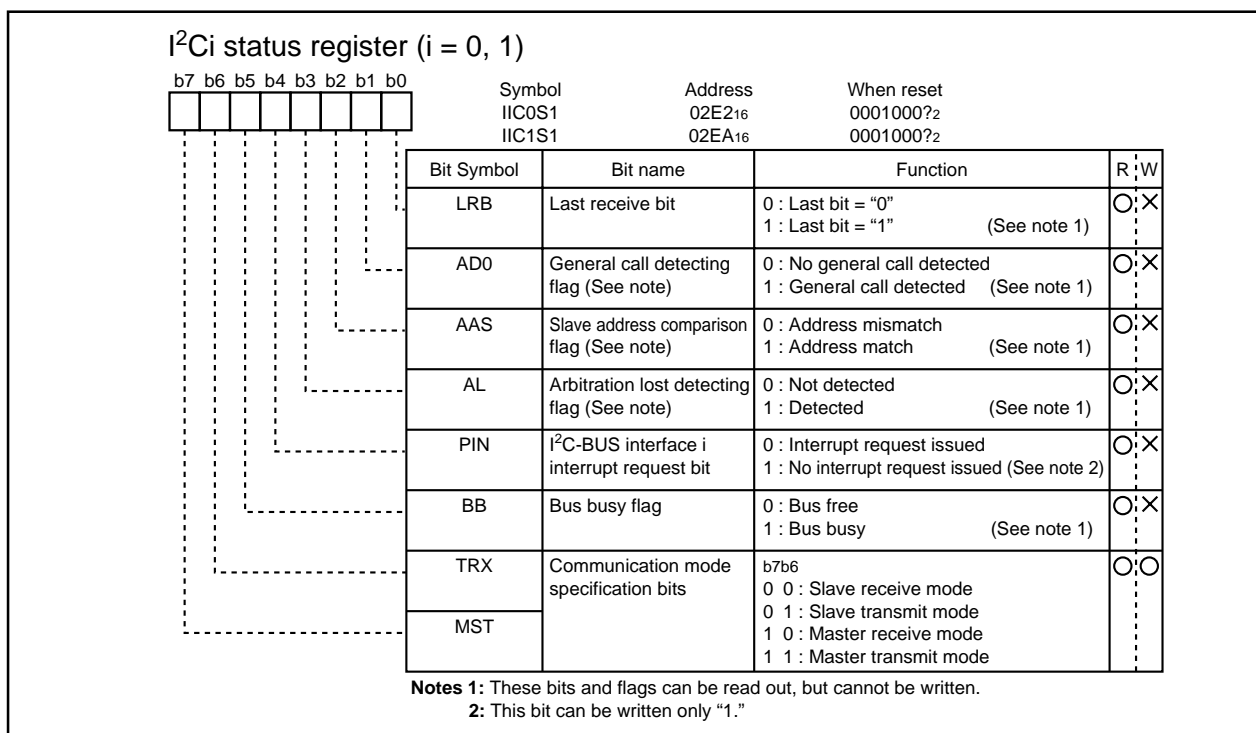


Fig. 2.11.40 I²Ci status register (i = 0, 1)

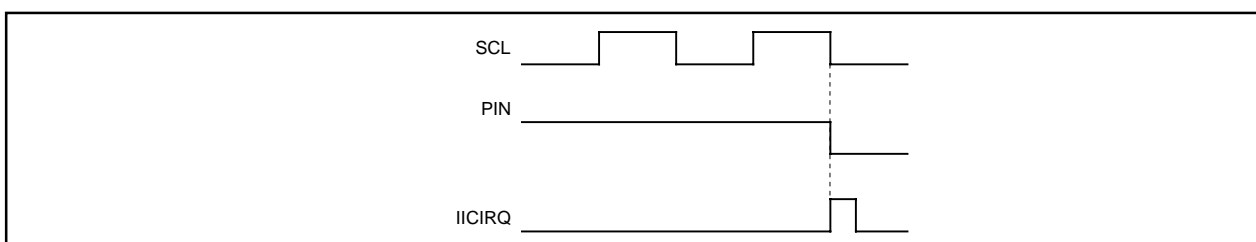


Fig. 2.11.41 Interrupt request signal generation timing

(7) START condition generation method

When the ESO bit of the I²Ci control register is “1,” execute a write instruction to the I²Ci status register to set the MST, TRX and BB bits to “1.” A START condition will then be generated. After that, the bit counter becomes “0002” and an SCL for 1 byte is output. The START condition generation timing and BB bit set timing are different in the standard clock mode and the high-speed clock mode. Refer to Figure 2.11.42 for the START condition generation timing diagram, and Table 2.11.13 for the START condition/STOP condition generation timing table.

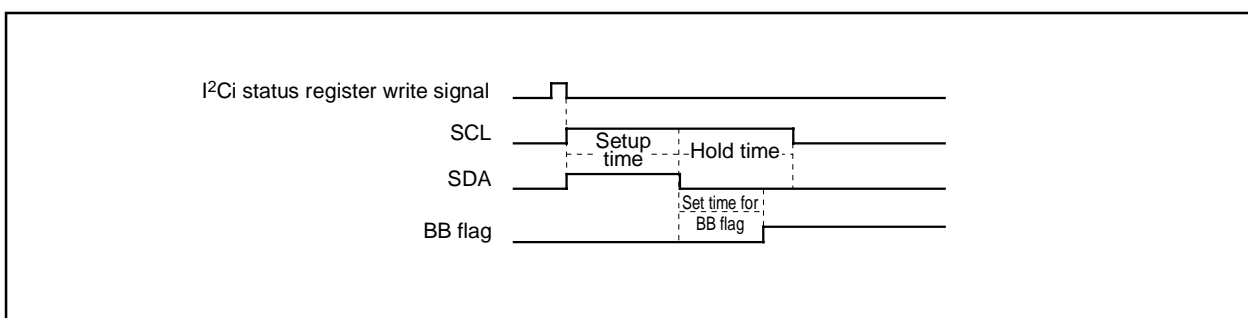


Fig. 2.11.42 START condition generation timing diagram

(8) STOP condition generation method

When the ESO bit of the I²Ci control register is “1,” execute a write instruction to the I²Ci status register for setting the MST bit and the TRX bit to “1” and the BB bit to “0”. A STOP condition will then be generated. The STOP condition generation timing and the BB flag reset timing are different in the standard clock mode and the high-speed clock mode. Refer to Figure 2.11.43 for the STOP condition generation timing diagram, and Table 2.11.13 for the START condition/STOP condition generation timing table.

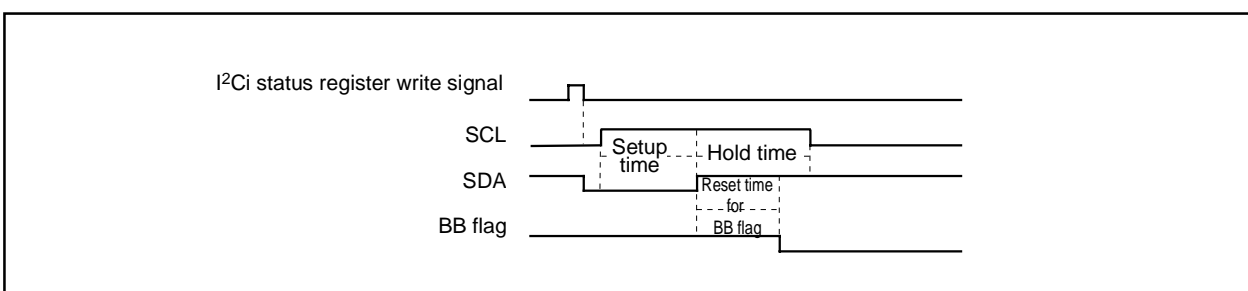


Fig. 2.11.43 STOP condition generation timing diagram

Table 2.11.13 START condition/STOP condition generation timing table

| Item | Standard Clock Mode | High-speed Clock Mode |
|----------------------------|-----------------------|-----------------------|
| Setup time | 5.35 μs (53.5 cycles) | 1.85 μs (18.5 cycles) |
| Hold time | 4.9 μs (49 cycles) | 2.4 μs (24 cycles) |
| Set/reset time for BB flag | 3.75 μs (37.5 cycles) | 0.85 μs (8.5 cycles) |

Note: Absolute time at BCLK = 10 MHz. The value in parentheses denotes the number of BCLK cycles.

(9) START/STOP condition detect conditions

The START/STOP condition detect conditions are shown in Figure 2.11.44 and Table 2.11.14. Only when the 3 conditions of Table 2.11.14 are satisfied, a START/STOP condition can be detected.

Note: When a STOP condition is detected in the slave mode (MST = 0), an interrupt request signal <IICIRQ> is generated to the CPU.

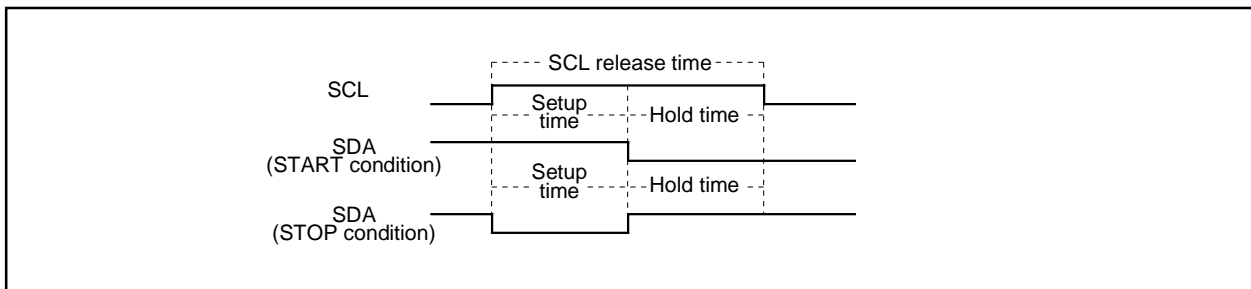


Fig. 2.11.44 START condition/STOP condition detect timing diagram

Table 2.11.14 START condition/STOP condition detect conditions

| Standard Clock Mode | High-speed Clock Mode |
|---------------------------------------|---------------------------------------|
| 6.5 μs (65 cycles) < SCL release time | 1.0 μs (10 cycles) < SCL release time |
| 3.25 μs (32.5 cycles) < Setup time | 0.5 μs (5 cycles) < Setup time |
| 3.25 μs (32.5 cycles) < Hold time | 0.5 μs (5 cycles) < Hold time |

Note: Absolute time at BCLK = 10 MHz. The value in parentheses denotes the number of BCLK cycles.

(10) Address data communication

There are two address data communication formats, namely, 7-bit addressing format and 10-bit addressing format. The respective address communication formats is described below.

■ 7-bit addressing format

To meet the 7-bit addressing format, set the 10BIT SAD bit of the I²Ci control register to “0.” The first 7-bit address data transmitted from the master is compared with the high-order 7-bit slave address stored in the I²Ci address register. At the time of this comparison, address comparison of the RBW bit of the I²Ci address register is not made. For the data transmission format when the 7-bit addressing format is selected, refer to Figure 2.11.45, (1) and (2).

■ 10-bit addressing format

To meet the 10-bit addressing format, set the 10BIT SAD bit of the I²Ci control register to “1.” An address comparison is made between the first-byte address data transmitted from the master and the 7-bit slave address stored in the I²Ci address register. At the time of this comparison, an address comparison between the RBW bit of the I²Ci address register and the R \bar{W} bit which is the last bit of the address data transmitted from the master is made. In the 10-bit addressing mode, the R \bar{W} bit which is the last bit of the address data not only specifies the direction of communication for control data but also is processed as an address data bit.

When the first-byte address data matches the slave address, the AAS bit of the I²Ci status register is set to “1.” After the second-byte address data is stored into the I²Ci data shift register, make an address comparison between the second-byte data and the slave address by software. When the address data of the 2nd bytes matches the slave address, set the RBW bit of the I²Ci address register to “1” by software. This processing can match the 7-bit slave address and R \bar{W} data, which are received after a RESTART condition is detected, with the value of the I²Ci address register. For the data transmission format when the 10-bit addressing format is selected, refer to Figure 2.11.45, (3) and (4).

(11) Example of Master Transmission

An example of master transmission in the standard clock mode, at the SCL frequency of 100 kHz and in the ACK return mode is shown below.

- ① Set a slave address in the high-order 7 bits of the I²Ci address register and “0” in the RBW bit.
- ② Set the ACK return mode and SCL = 100 kHz by setting “85₁₆” in the I²Ci clock control register.
- ③ Set “10₁₆” in the I²Ci status register and hold the SCL at the HIGH.
- ④ Set a communication enable status by setting “08₁₆” in the I²Ci control register.
- ⑤ Set the address data of the destination of transmission in the high-order 7 bits of the I²Ci data shift register and set “0” in the least significant bit.
- ⑥ Set “F0₁₆” in the I²Ci status register to generate a START condition. At this time, an SCL for 1 byte and an ACK clock automatically occurs.
- ⑦ Set transmit data in the I²Ci data shift register. At this time, an SCL and an ACK clock automatically occurs.
- ⑧ When transmitting control data of more than 1 byte, repeat step ⑦.
- ⑨ Set “D0₁₆” in the I²Ci status register. After this, if ACK is not returned or transmission ends, a STOP condition will be generated.

(12) Example of Slave Reception

An example of slave reception in the high-speed clock mode, at the SCL frequency of 400 kHz, in the ACK non-return mode, using the addressing format, is shown below.

- ① Set a slave address in the high-order 7 bits of the I²Ci address register and “0” in the RBW bit.
- ② Set the no ACK clock mode and SCL = 400 kHz by setting “25₁₆” in the I²Ci clock control register.
- ③ Set “10₁₆” in the I²Ci status register and hold the SCL at the HIGH.
- ④ Set a communication enable status by setting “08₁₆” in the I²Ci control register.
- ⑤ When a START condition is received, an address comparison is made.
- ⑥
 - When all transmitted address are “0” (general call):
AD0 of the I²Ci status register is set to “1” and an interrupt request signal occurs.
 - When the transmitted addresses match the address set in ①:
ASS of the I²Ci status register is set to “1” and an interrupt request signal occurs.
 - In the cases other than the above:
AD0 and AAS of the I²Ci status register are set to “0” and no interrupt request signal occurs.
- ⑦ Set dummy data in the I²Ci data shift register.
- ⑧ When receiving control data of more than 1 byte, repeat step ⑦.
- ⑨ When a STOP condition is detected, the communication ends.

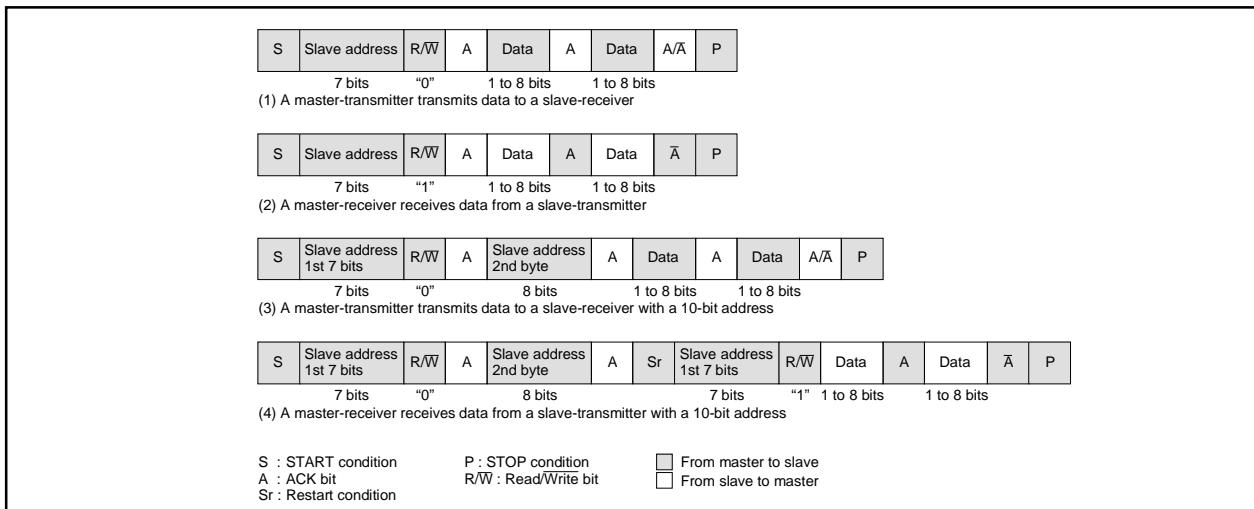


Fig. 2.11.45 Address data communication format

(13) Precautions when using multi-master I²C-BUS interface i

■ BCLK operation mode

Select the no-division mode and set the main clock frequency to $f(XIN) = 10 \text{ MHz}$.

■ Used instructions

Specify byte (.B) as data size to access multi-master I²C-BUS interface i-related registers.

■ Read-modify-write instruction

The precautions when the read-modify-write instruction such as BSET, BCLR etc. is executed for each register of the multi-master I²C-BUS interface are described below.

• I²Ci data shift register (IICiS0)

When executing the read-modify-write instruction for this register during transfer, data may become a value not intended.

• I²Ci address register (IICiS0D)

When the read-modify-write instruction is executed for this register at detecting the STOP condition, data may become a value not intended. It is because hardware changes the read/write bit (RBW) at the above timing.

• I²Ci status register (IICiS1)

Do not execute the read-modify-write instruction for this register because all bits of this register are changed by hardware.

• I²Ci control register (IICiS1D)

When the read-modify-write instruction is executed for this register at detecting the START condition or at completing the byte transfer, data may become a value not intended. Because hardware changes the bit counter (BC0–BC2) at the above timing.

• I²Ci clock control register (IICiS2)

The read-modify-write instruction can be executed for this register.

• I²Ci port selection register (IICiS2D)

Since the read value of high-order 4 bits is indeterminate, the read-modify-write instruction cannot be used.

• I²Ci transmit buffer register (IICiS0S)

Since the value of all bits is indeterminate, the read-modify-write instruction cannot be used.

■ START condition generating procedure using multi-master

```

:
FCLR      I                      (Interrupt disabled)
BTST     5, IICiS1              (BB flag confirming and branch process)
JC       BUSBUSY
BUSFREE:
MOV.B    SA, IICiS0             (Writing of slave address value <SA>)
NOP
NOP
MOV.B    #F0H, IICiS1          (Trigger of START condition generating)
FSET     I                      (Interrupt enabled)
:

BUSBUSY:
FSETI    (Interrupt enabled)
:

```

① Be sure to add NOP instruction X 2 between writing the slave address value and setting trigger of START condition generating shown the above procedure example.

② When using multi-master system, disable interrupts during the following three process steps:

- BB flag confirming
- Writing of slave address value
- Trigger of START condition generating

When the condition of the BB flag is bus busy, enable interrupts immediately.

When using single-master system, it is not necessary to disable interrupts above.

■ RESTART condition generating procedure

```

:
MOV.B    SA, IICiS0S           (Writing of slave address value <SA>) ——— ①
NOP
NOP
MOV.B    #F0H, IICiS1          (Trigger of RESTART condition generating)
:

```

① Use the I²Ci transmit buffer register to write the slave address value to the I²Ci data shift register.

And also, be sure to add NOP instruction X 2.

■ Writing to I²Ci status register

Do not execute an instruction to set the PIN bit to “1” from “0” and an instruction to set the MST and TRX bits to “0” from “1” simultaneously. It is because it may enter the state that the SCL pin is released and the SDA pin is released after about one machine cycle. Do not execute an instruction to set the MST and TRX bits to “0” from “1” simultaneously when the PIN bit is “1.” It is because it may become the same as above.

■ Process of after STOP condition generating

Do not write data in the I²Ci data shift register (IICiS0) and the I²Ci status register (IICiS1) until the bus busy flag BB becomes “0” after generating the STOP condition in the master mode. It is because the STOP condition waveform might not be normally generated. Reading to the above registers do not have the problem.

2.12 A-D Converter

The A-D converter consists of one 8-bit successive approximation A-D converter circuit with a capacitive coupling amplifier. Pins P102 to P107 also function as the analog signal input pins. The direction registers of these pins for A-D conversion must therefore be set to input. The Vref connect bit (bit 5 at address 03D716) can be used to isolate the resistance ladder of the A-D converter from the reference voltage (VREF) when the A-D converter is not used. Doing so stops any current flowing into the resistance ladder from VREF, reducing the power dissipation. When using the A-D converter, start A-D conversion only after setting bit 5 of 03D716 to connect VREF.

The result of A-D conversion is stored in the A-D registers of the selected pins.

Table 2.12.1 shows the performance of the A-D converter. Figure 2.12.1 shows the block diagram of the A-D converter, and Figures 2.12.2 to 2.12.5 show the A-D converter-related registers.

Table 2.12.1 Performance of A-D converter

| Item | Performance |
|--------------------------------------|---|
| Method of A-D conversion | Successive approximation (capacitive coupling amplifier) |
| Analog input voltage (Note 1) | 0V to AVCC (VCC) |
| Operating clock ϕ_{AD} (Note 2) | $f_{AD}/\text{divide-by-2}$ of $f_{AD}/\text{divide-by-4}$ of f_{AD} , $f_{AD}=f(XIN)$ |
| Resolution | 8-bit |
| Absolute precision | VCC = 5V <ul style="list-style-type: none"> • Without sample and hold function: ± 5 LSB • With sample and hold function: ± 5 LSB |
| Operating modes | One-shot mode, repeat mode, single sweep mode, repeat sweep mode 0, and repeat sweep mode 1 |
| Analog input pins | 6 pins (AN0 to AN5) |
| A-D conversion start condition | <ul style="list-style-type: none"> • Software trigger A-D conversion starts when the A-D conversion start flag changes to "1" |
| Conversion speed per pin | <ul style="list-style-type: none"> • Without sample and hold function 49 ϕ_{AD} cycles • With sample and hold function 28 ϕ_{AD} cycles |

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

2: Divide the frequency if $f(XIN)$ exceeds 10 MHz, and make ϕ_{AD} frequency equal to 10 MHz. Without sample and hold function, set the ϕ_{AD} frequency to 250kHz min.

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

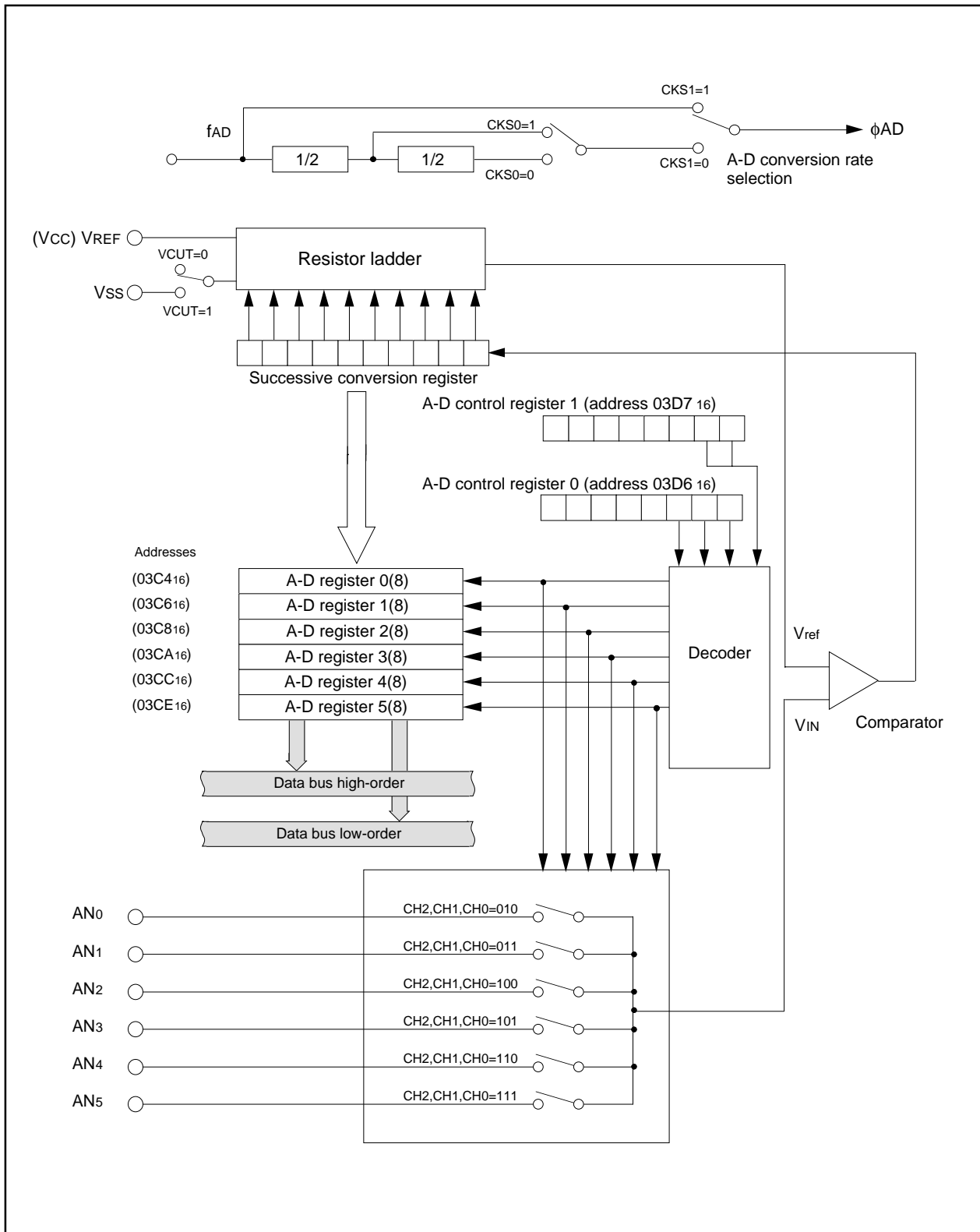


Figure 2.12.1 Block diagram of A-D converter

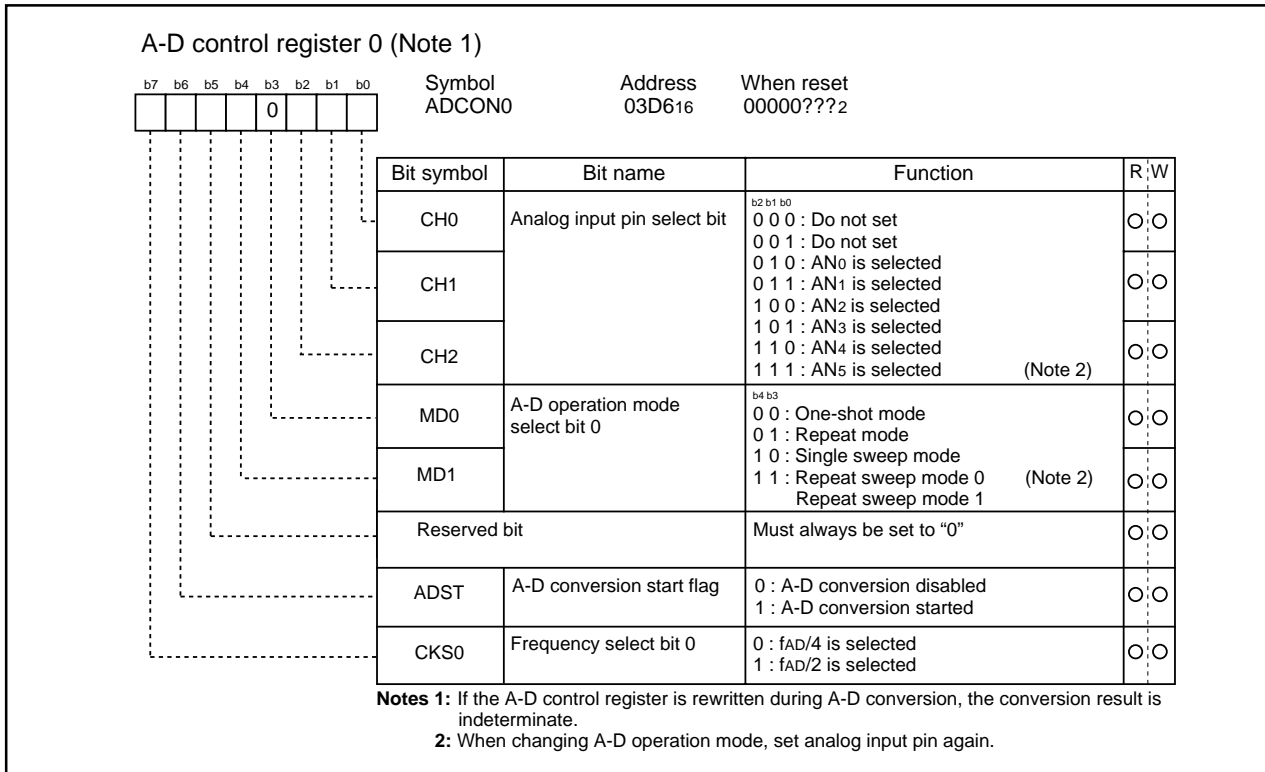


Figure 2.12.2 A-D control register 0

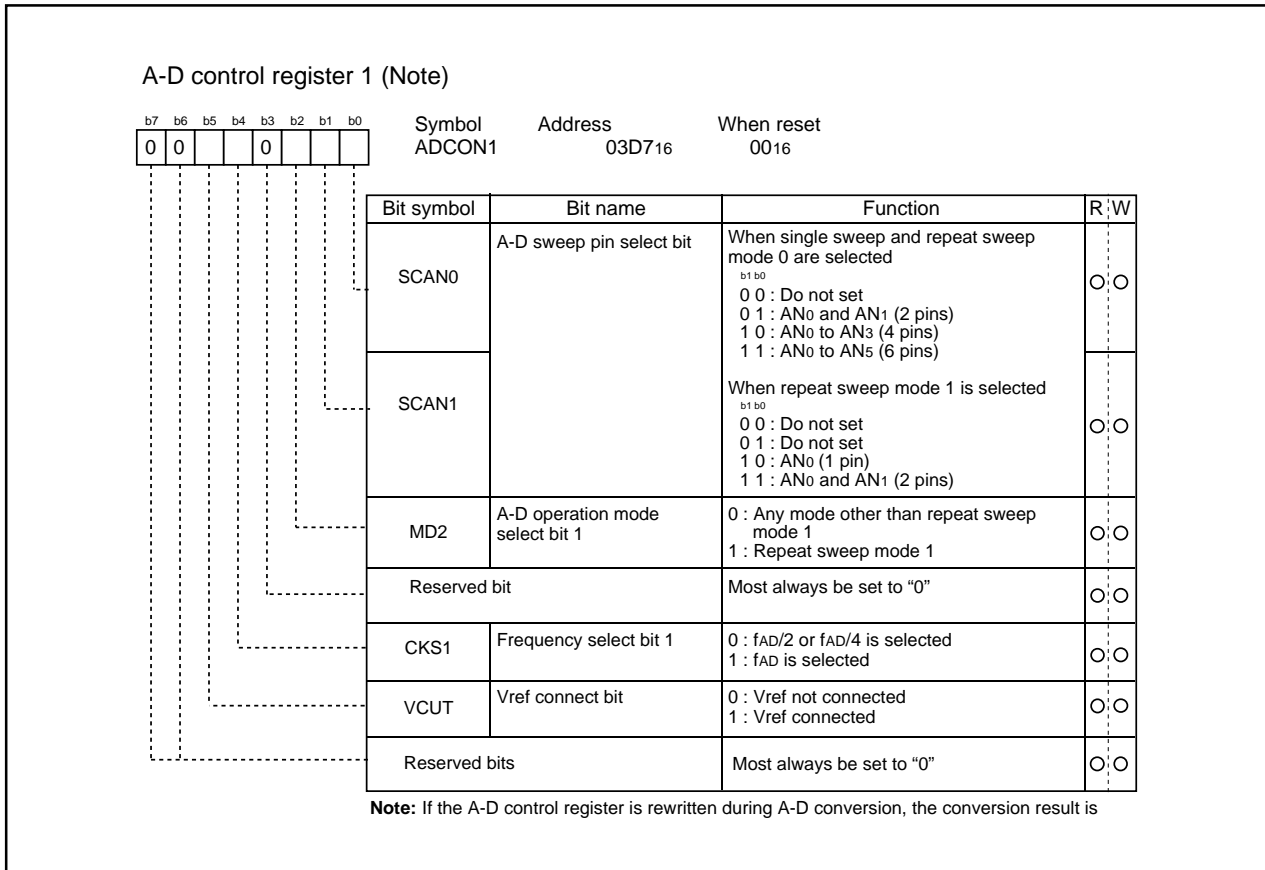


Figure 2.12.3 A-D control register 1

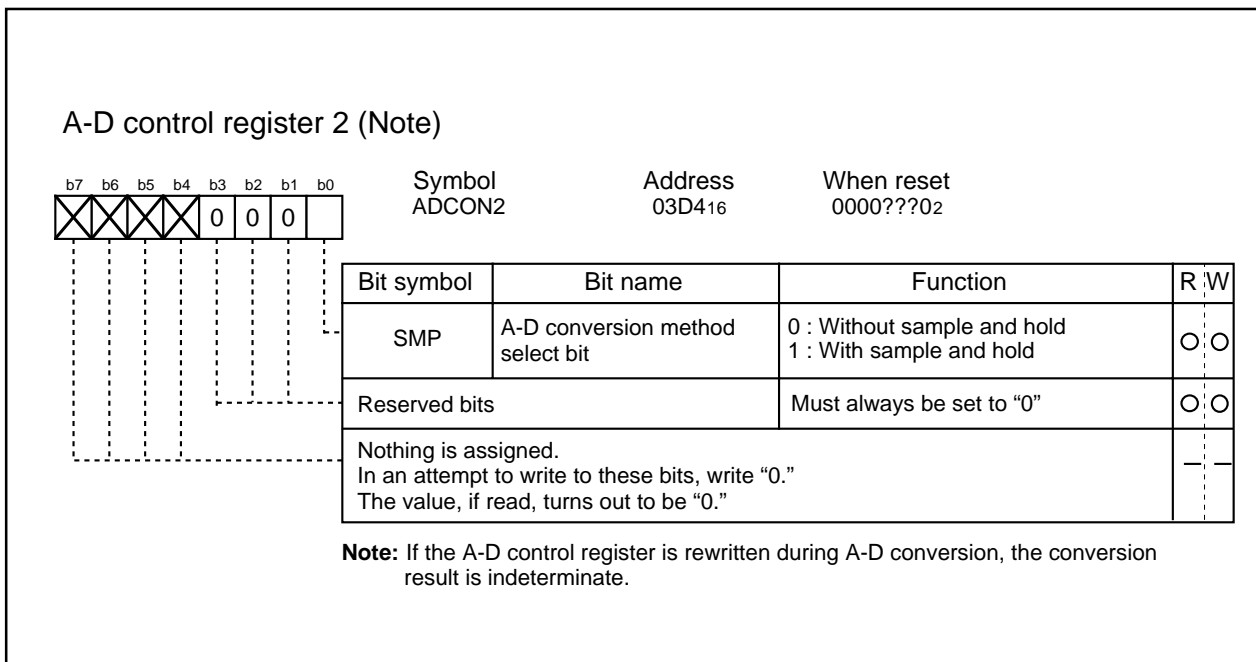


Figure 2.12.4 A-D control register 2

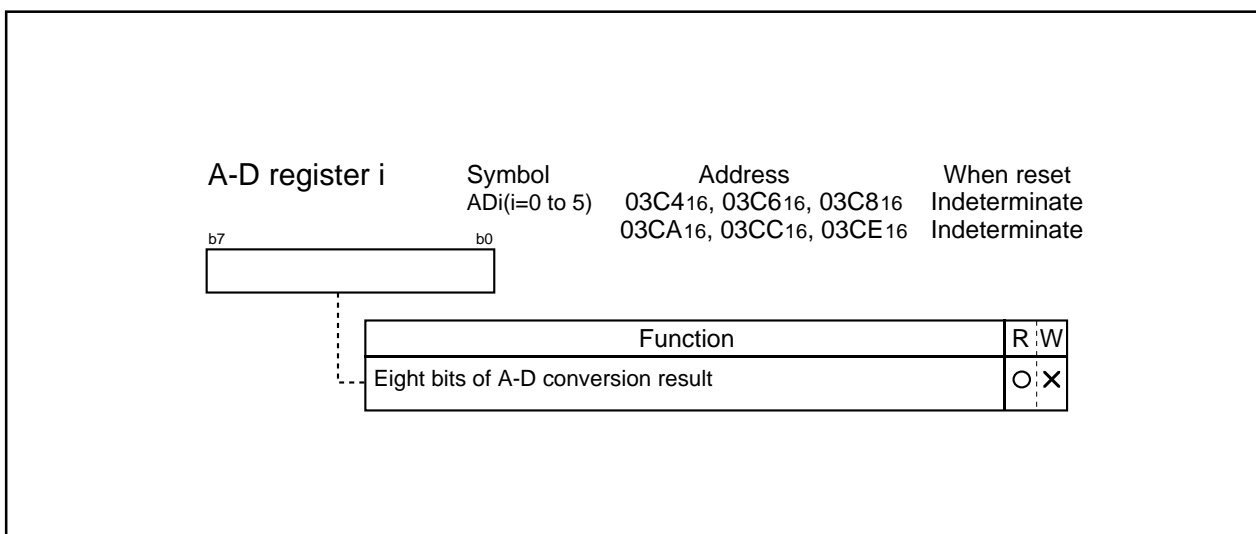


Figure 2.12.5 A-D register i (i = 0 to 5)

2.12.1 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 2.12.2 shows the specifications of one-shot mode. Figures 2.12.6 and 2.12.7 show the A-D control register in one-shot mode.

Table 2.12.2 One-shot mode specifications

| Item | Specification |
|-------------------------------------|---|
| Function | The pin selected by the analog input pin select bit is used for one A-D conversion |
| Start condition | Writing "1" to A-D conversion start flag |
| Stop condition | <ul style="list-style-type: none"> • End of A-D conversion • Writing "0" to A-D conversion start flag |
| Interrupt request generation timing | End of A-D conversion |
| Input pin | One of AN0 to AN5, as selected |
| Reading of result of A-D converter | Read A-D register corresponding to selected pin |

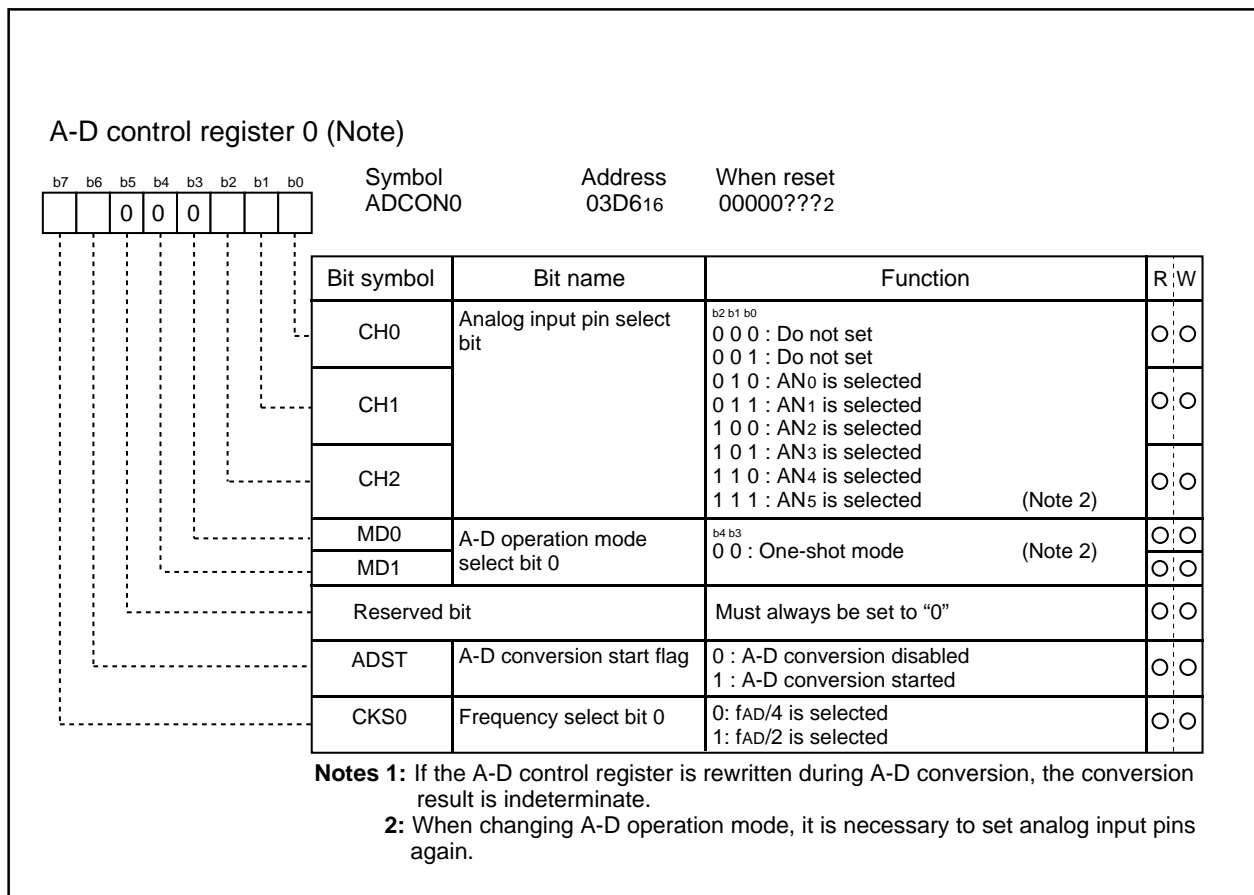


Figure 2.12.6 A-D control register 0 in one-shot mode

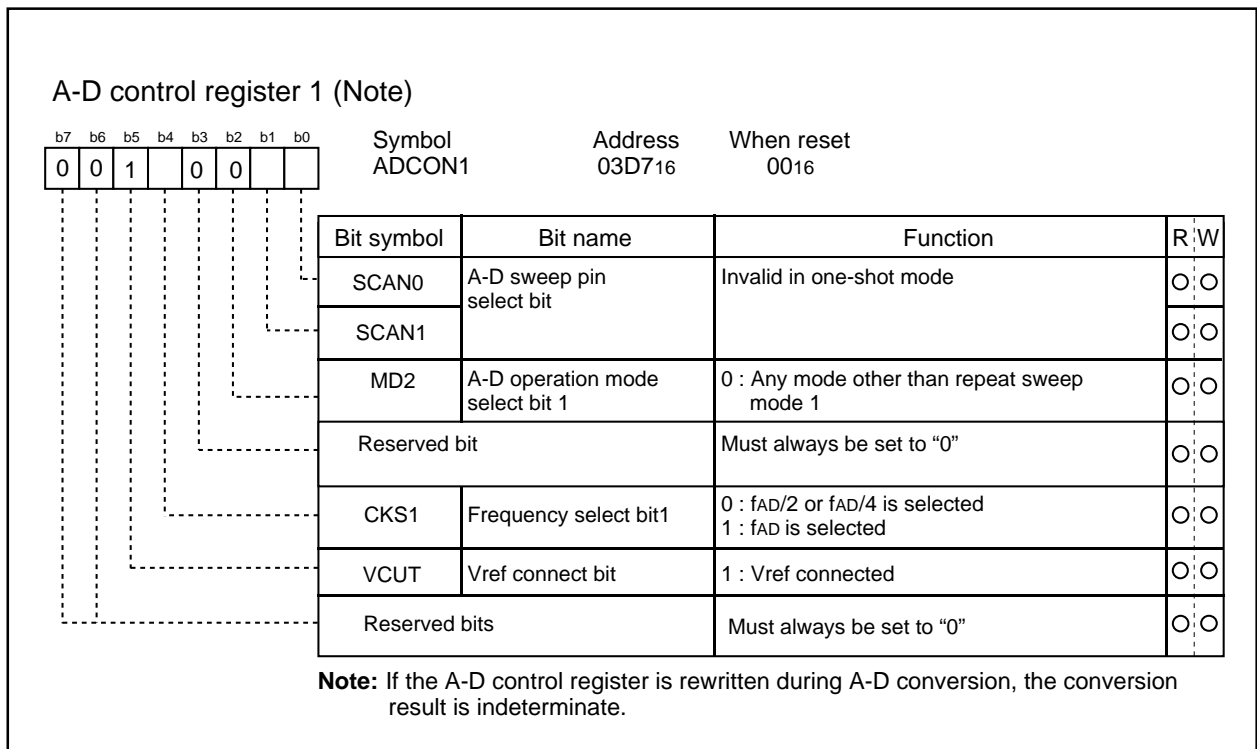


Figure 2.12.7 A-D control register 1 in one-shot mode

2.12.2 Repeat Mode

In repeat mode, the pin selected using the analog input pin select bit is used for repeated A-D conversion. Table 2.12.3 shows the specifications of repeat mode. Figures 2.12.8 and 2.12.9 show the A-D control register in repeat mode.

Table 2.12.3 Repeat mode specifications

| Item | Specification |
|-------------------------------------|---|
| Function | The pin selected by the analog input pin select bit is used for repeated A-D conversion |
| Start condition | Writing "1" to A-D conversion start flag |
| Stop condition | Writing "0" to A-D conversion start flag |
| Interrupt request generation timing | None generated |
| Input pin | One of AN0 to AN5, as selected |
| Reading of result of A-D converter | Read A-D register corresponding to selected pin |

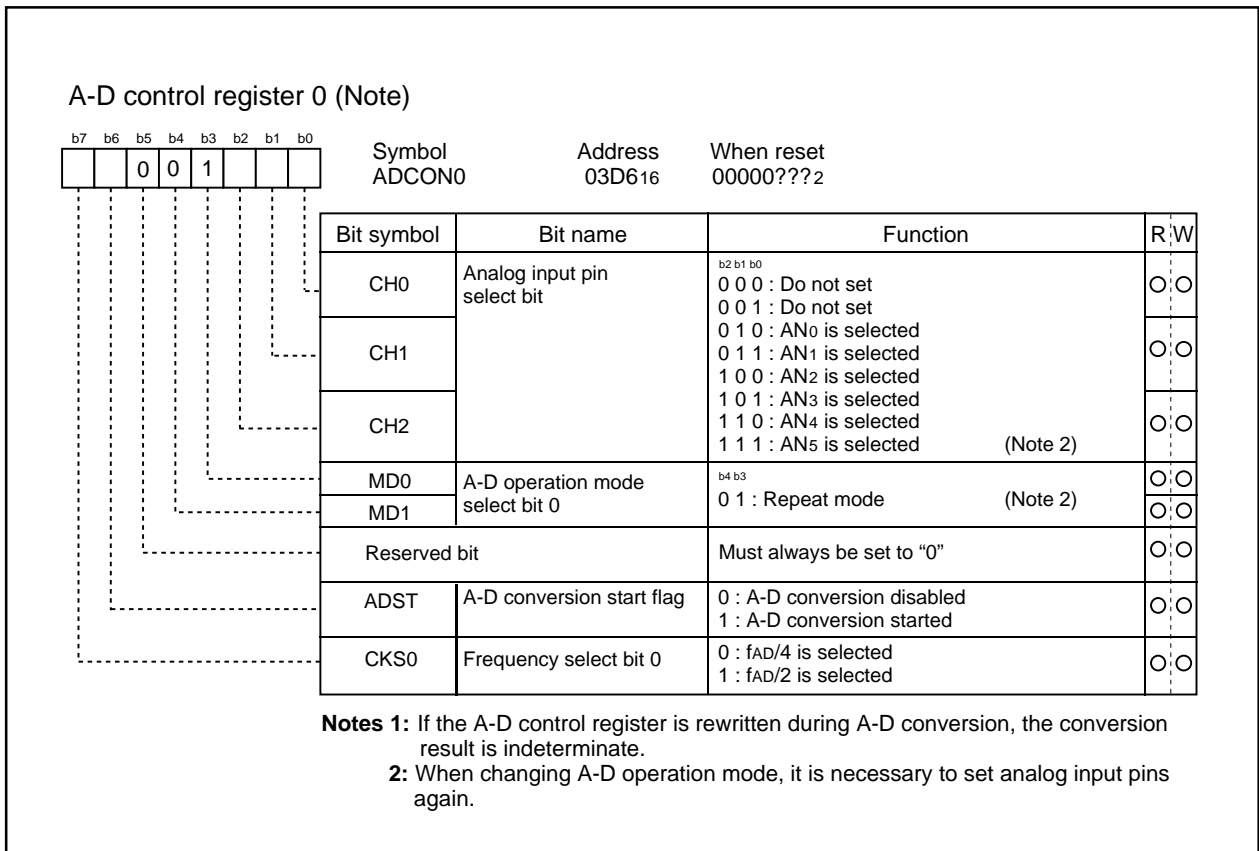


Figure 2.12.8 A-D conversion register 0 in repeat mode

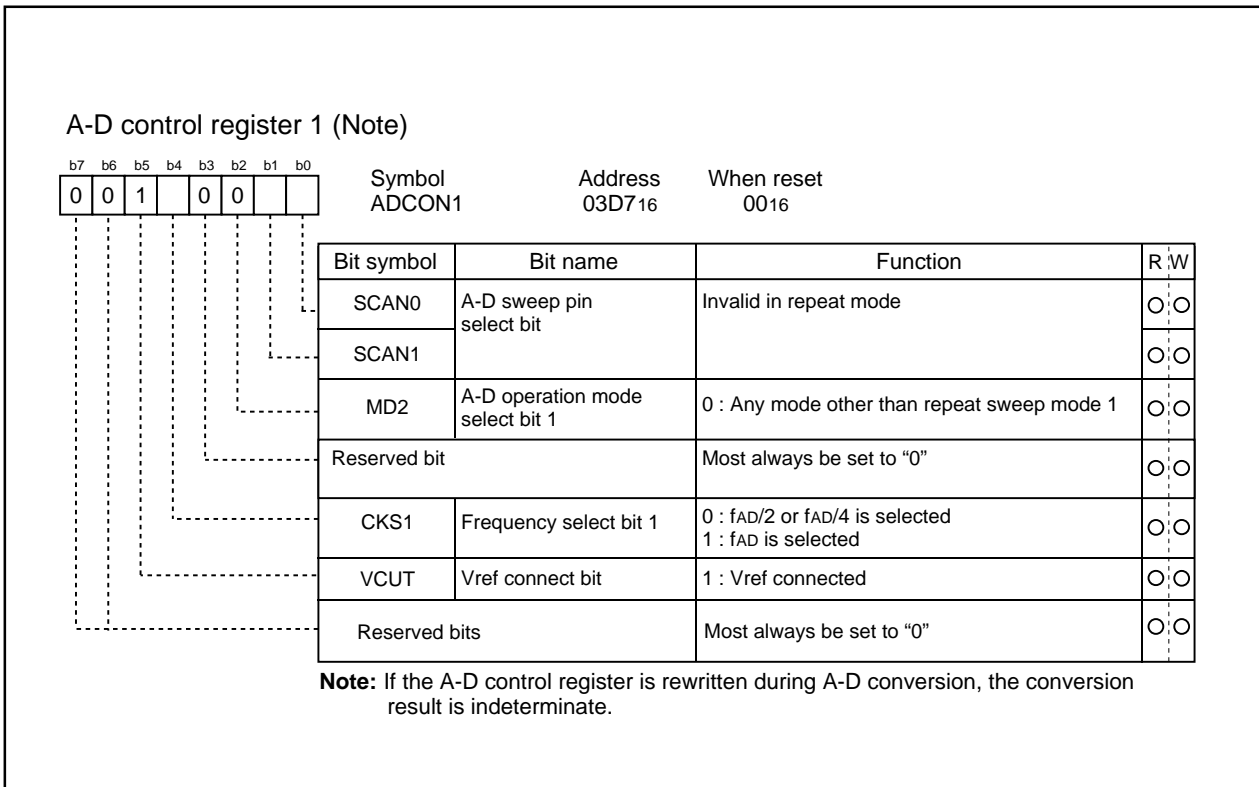


Figure 2.12.9 A-D conversion register 1 in repeat mode

2.12.3 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 2.12.4 shows the specifications of single sweep mode. Figures 2.12.10 and 2.12.11 show the A-D control register in single sweep mode.

Table 2.12.4 Single sweep mode specifications

| Item | Specification |
|-------------------------------------|--|
| Function | The pins selected by the A-D sweep pin select bit are used for one-by-one A-D conversion |
| Start condition | Writing "1" to A-D converter start flag |
| Stop condition | <ul style="list-style-type: none">• End of A-D conversion• 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) |
| Reading of result of A-D converter | Read A-D register corresponding to selected pin |

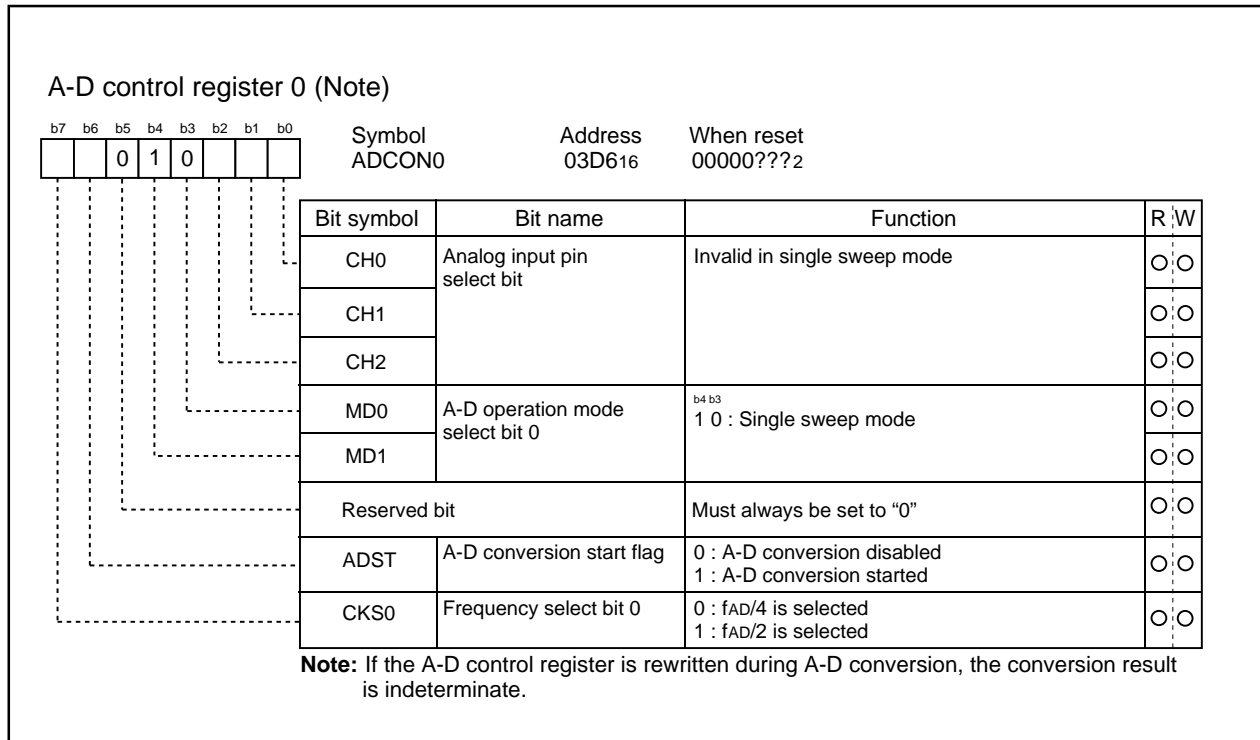


Figure 2.12.10 A-D control register 0 in single sweep mode

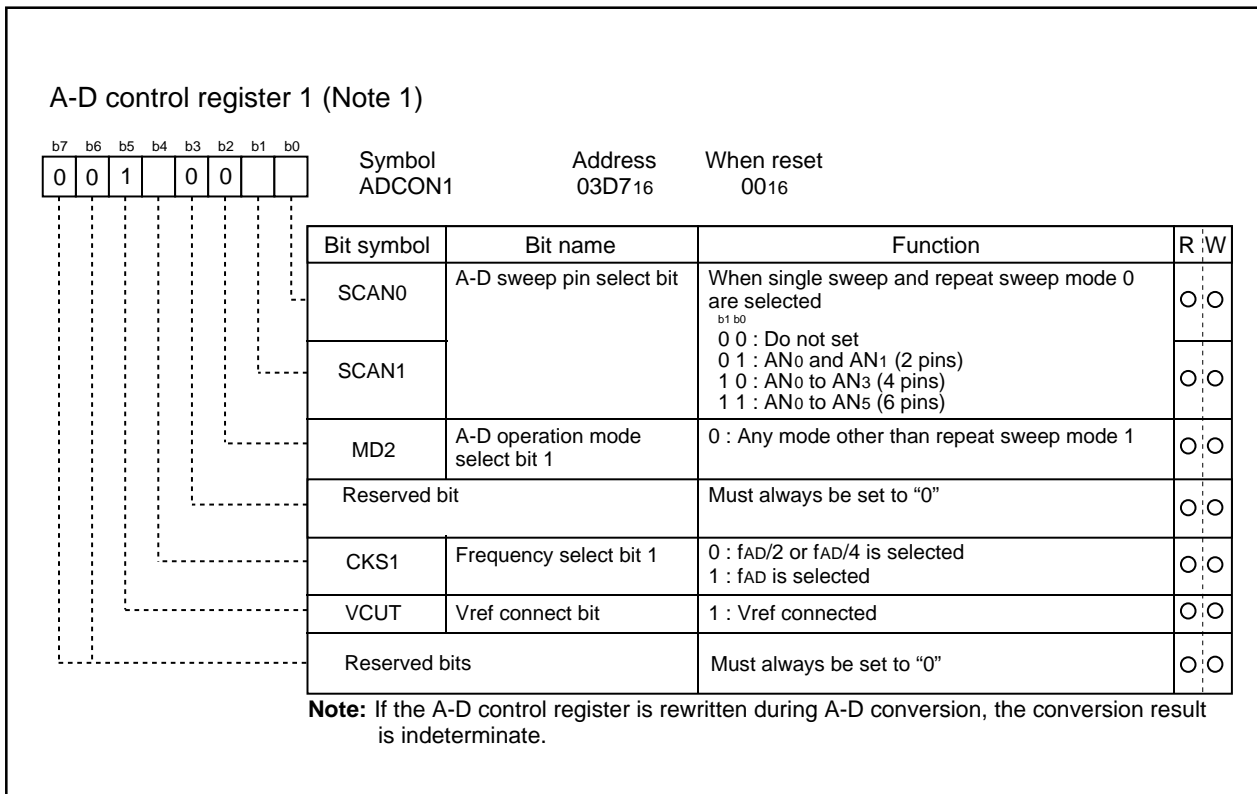


Figure 2.12.11 A-D control register 1 in single sweep mode

2.12.4 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 2.12.5 shows the specifications of repeat sweep mode 0. Figures 2.12.12 and 2.12.13 show the A-D control register in repeat sweep mode 0.

Table 2.12.5 Repeat sweep mode 0 specifications

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

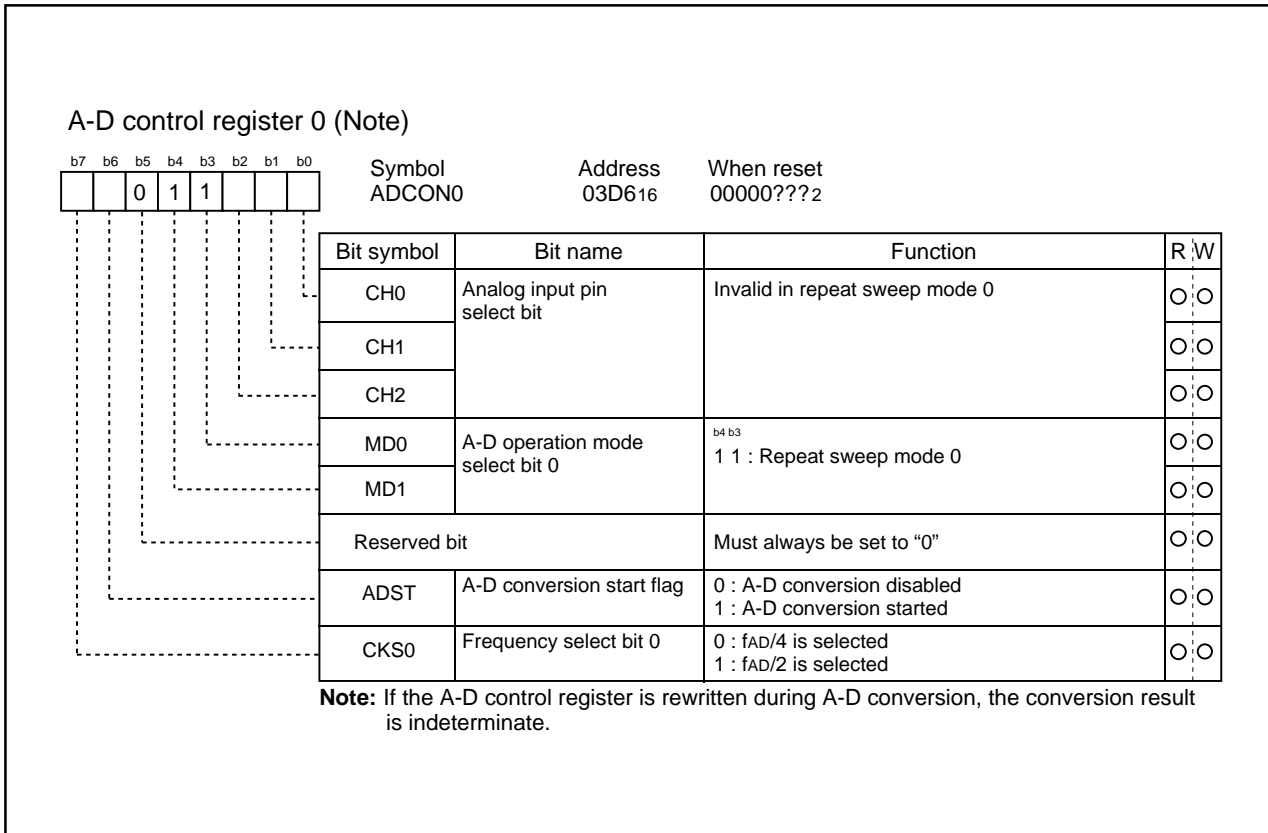


Figure 2.12.12 A-D control register 0 in repeat sweep mode 0

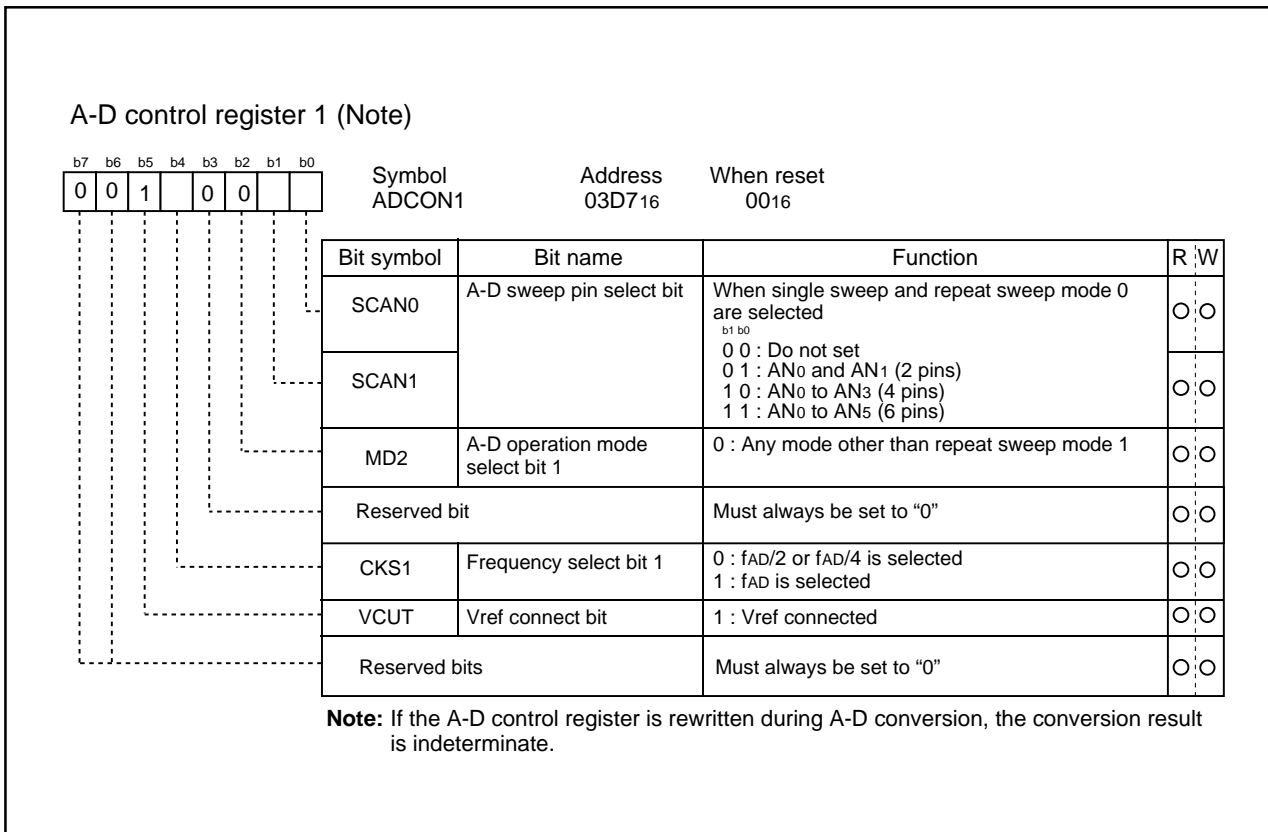


Figure 2.12.13 A-D control register 1 in repeat sweep mode 0

2.12.5 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 2.12.6 shows the specifications of repeat sweep mode 1. Figures 2.12.14 and 2.12.15 show the A-D control register in repeat sweep mode 1.

Table 2.12.6 Repeat sweep mode 1 specifications

| Item | Specification |
|-------------------------------------|---|
| Function | All pins perform repeat sweep A-D conversion, with emphasis on the pin or pins selected by the A-D sweep pin select bit Example : AN ₀ selected AN ₀ → AN ₁ → AN ₀ → AN ₂ → AN ₀ → 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) |
| Reading of result of A-D converter | Read A-D register corresponding to selected pin (at any time) |

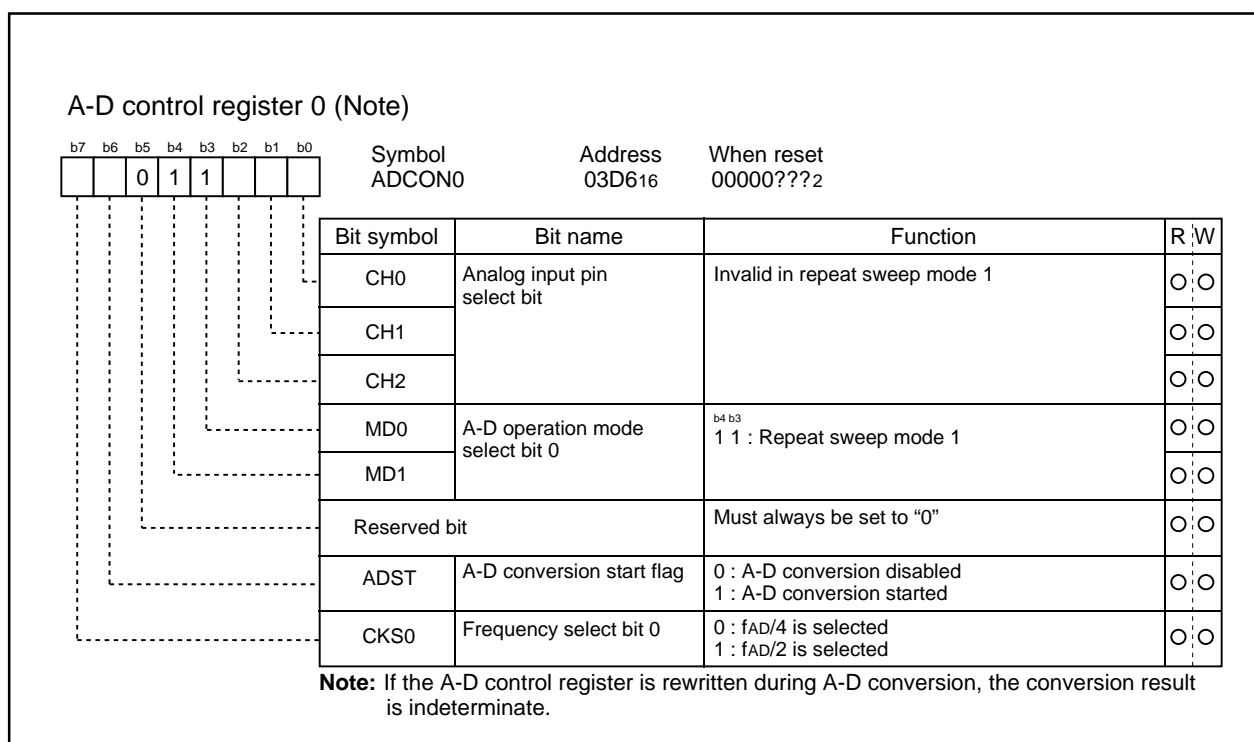


Figure 2.12.14 A-D control register 0 in repeat sweep mode 1

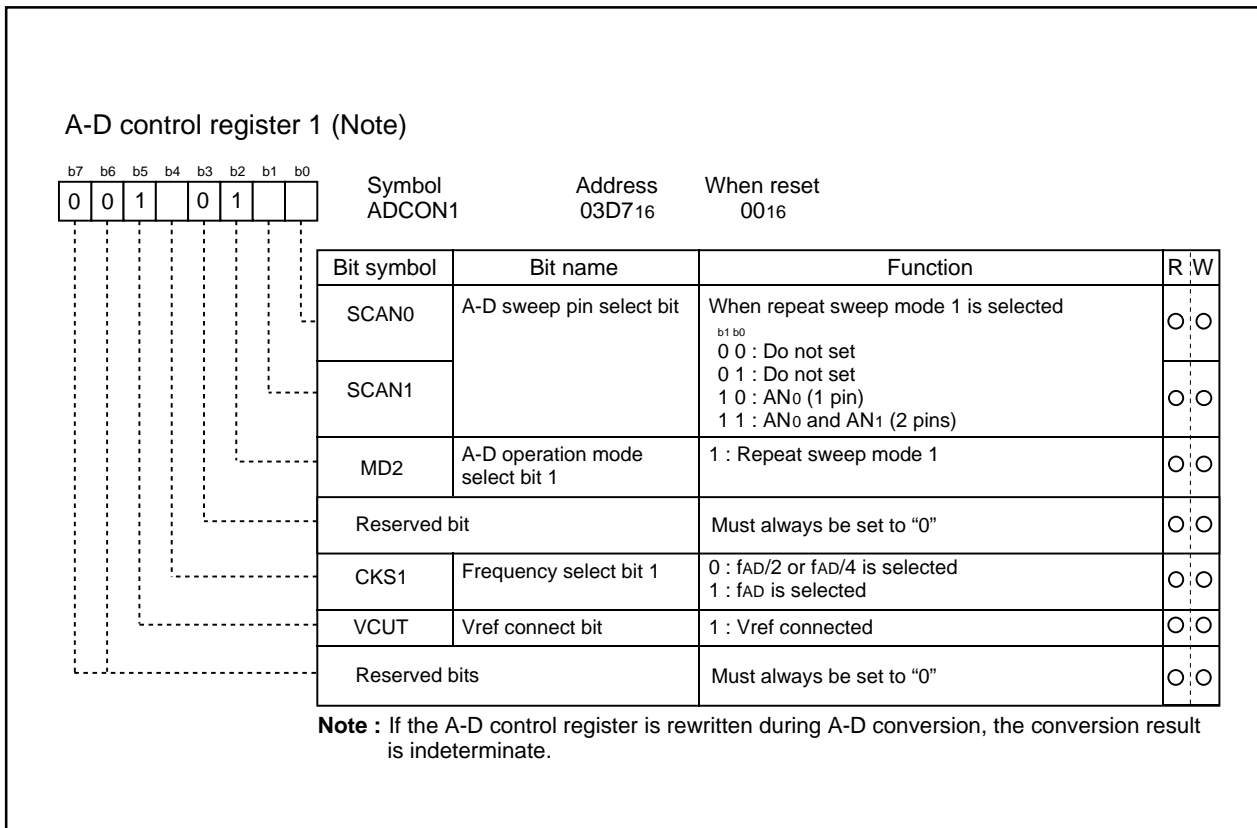


Figure 2.12.15 A-D control register 1 in repeat sweep mode 1

2.12.6 Sample and Hold

Sample and hold is selected by setting bit 0 of the A-D control register 2 (address 03D416) to "1". When sample and hold is selected, the rate of conversion of each pin increases. As a result, a 28 ϕ_{AD} cycle is achieved. 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.

2.13 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 2.13.1 lists the performance of the D-A converter. Figure 2.13.1 shows the block diagram of the D-A converter. Figure 2.13.2 shows the A-D control register, Figure 2.13.3 shows the D-A register and Figure 2.13.4 shows the D-A converter equivalent circuit.

Table 2.13.1 Performance of D-A converter

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

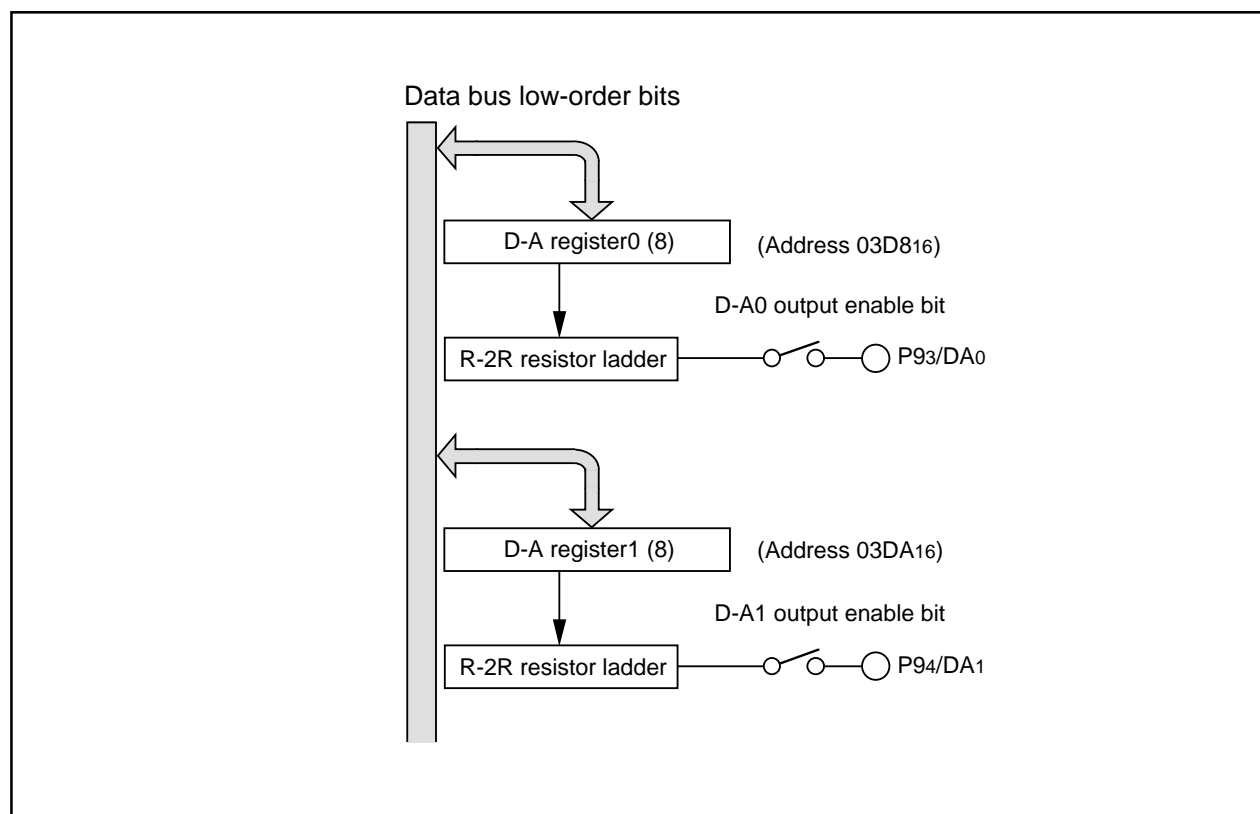


Figure 2.13.1 Block diagram of D-A converter

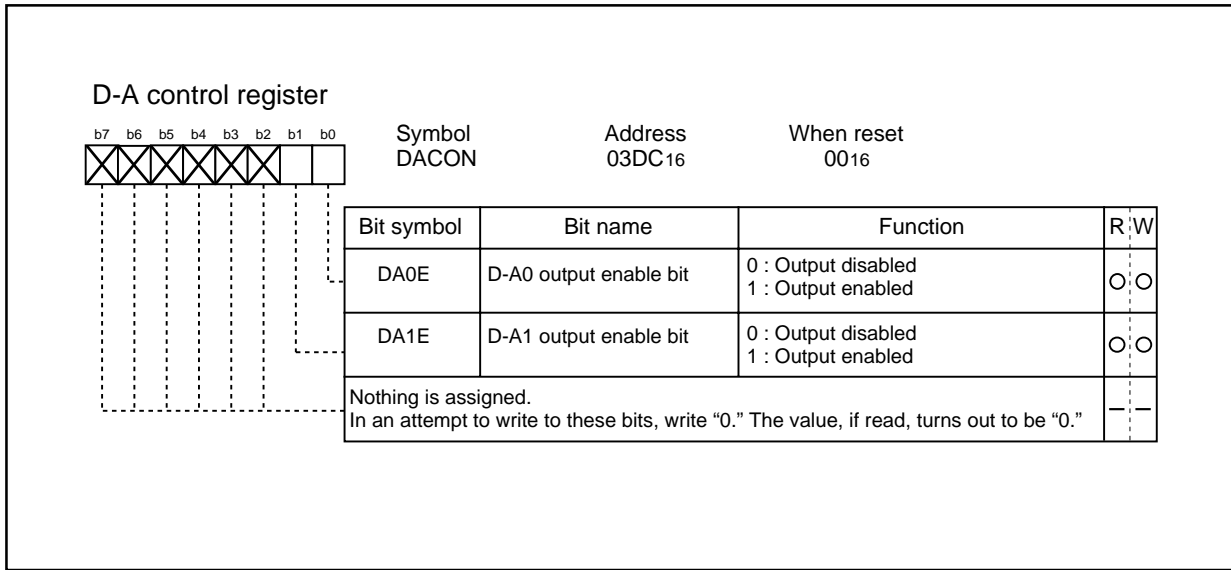


Figure 2.13.2 D-A control register

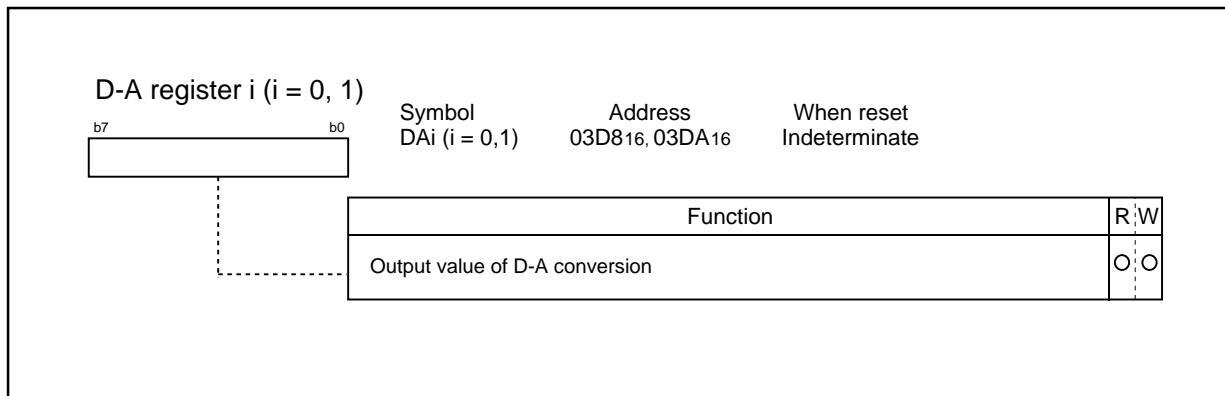


Figure 2.13.3 D-A register i (i = 0 and 1)

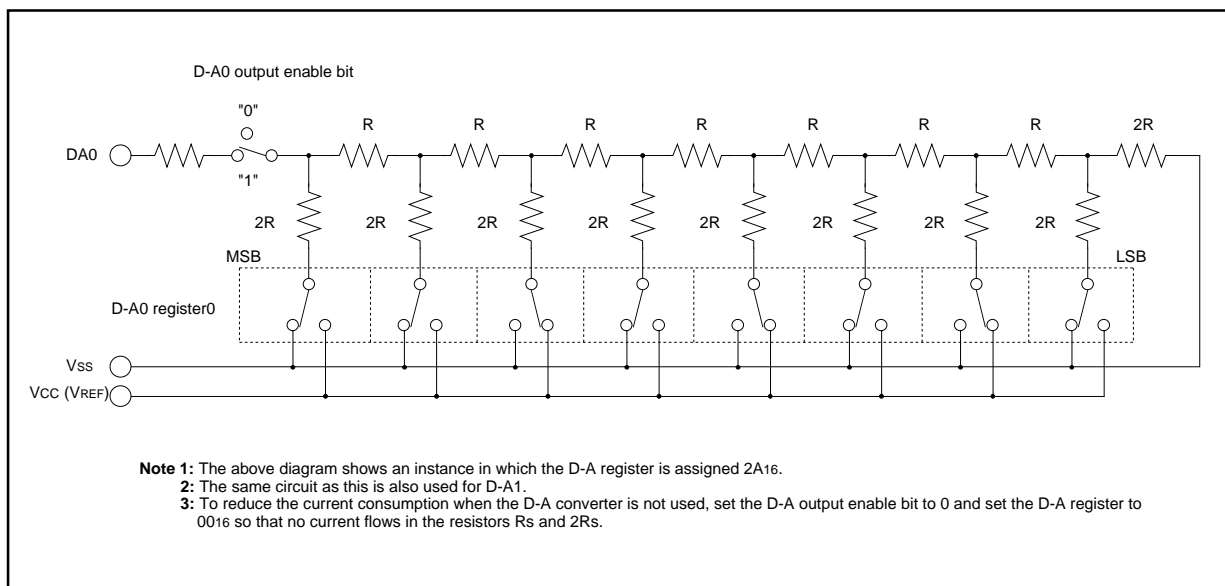


Figure 2.13.4 D-A converter equivalent circuit

2.14 Data Slicer

This microcomputer includes the data slicer function for the closed caption decoder (referred to as the CCD). This function takes out the caption data superimposed in the vertical blanking interval of a composite video signal. A composite video signal which makes the sync tip's polarity negative is input to the CVIN pin. When the data slicer function is not used, the data slicer circuit and the timing signal generating circuit can be cut off by setting bit 0 of the data slicer control register 1 (address 0260₁₆) to "0." These settings can realize the low-power dissipation.

Note: When using the data slicer, set bit 7 of the peripheral mode register (address 027D₁₆) according to the main clock frequency.

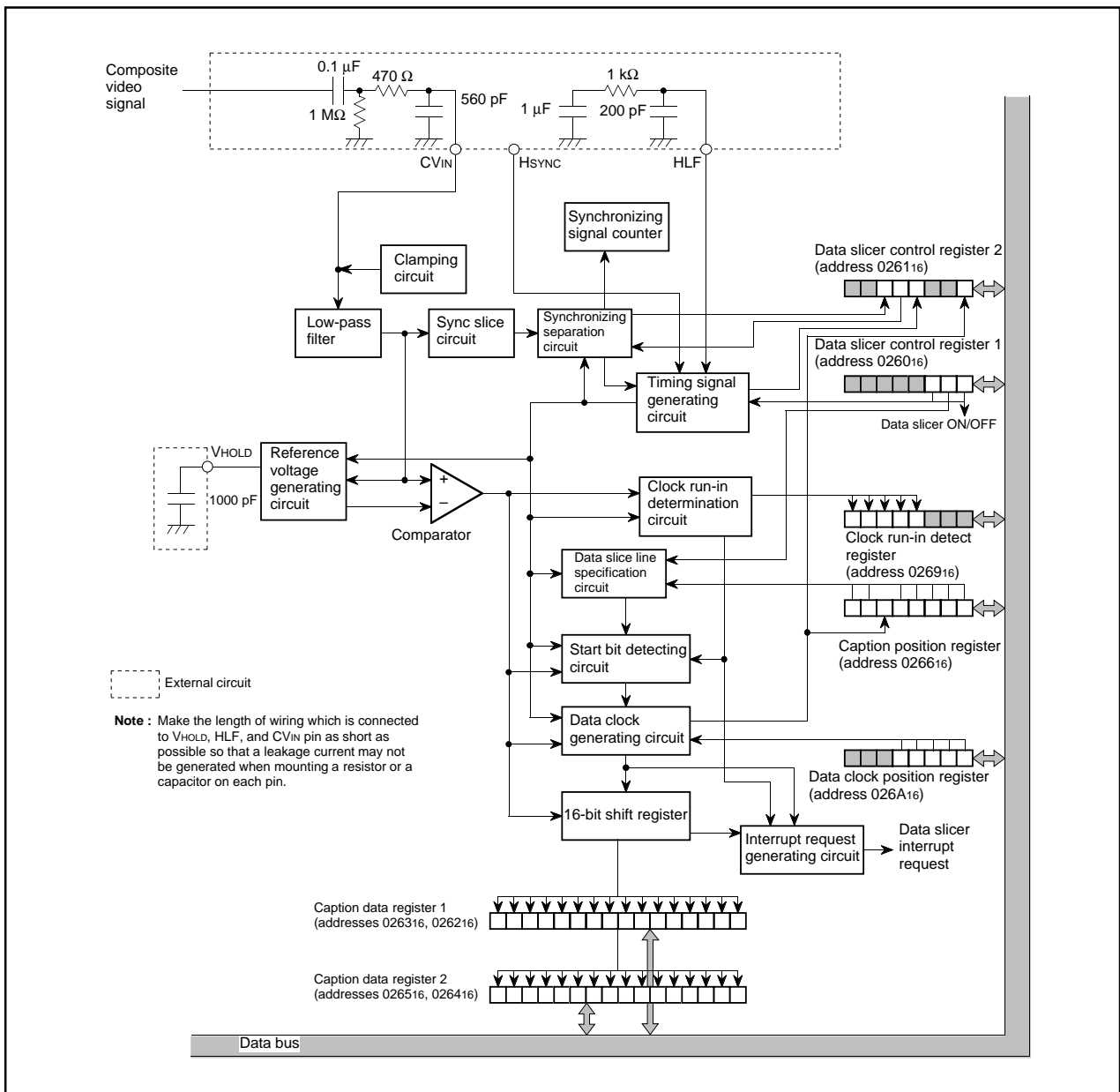


Figure 2.14.1 Data slicer block diagram

2.14.1 Notes when not Using Data Slicer

When bit 0 of data slicer control register 1 (address 026016) is "0," terminate the pins as shown in Figure 2.14.2

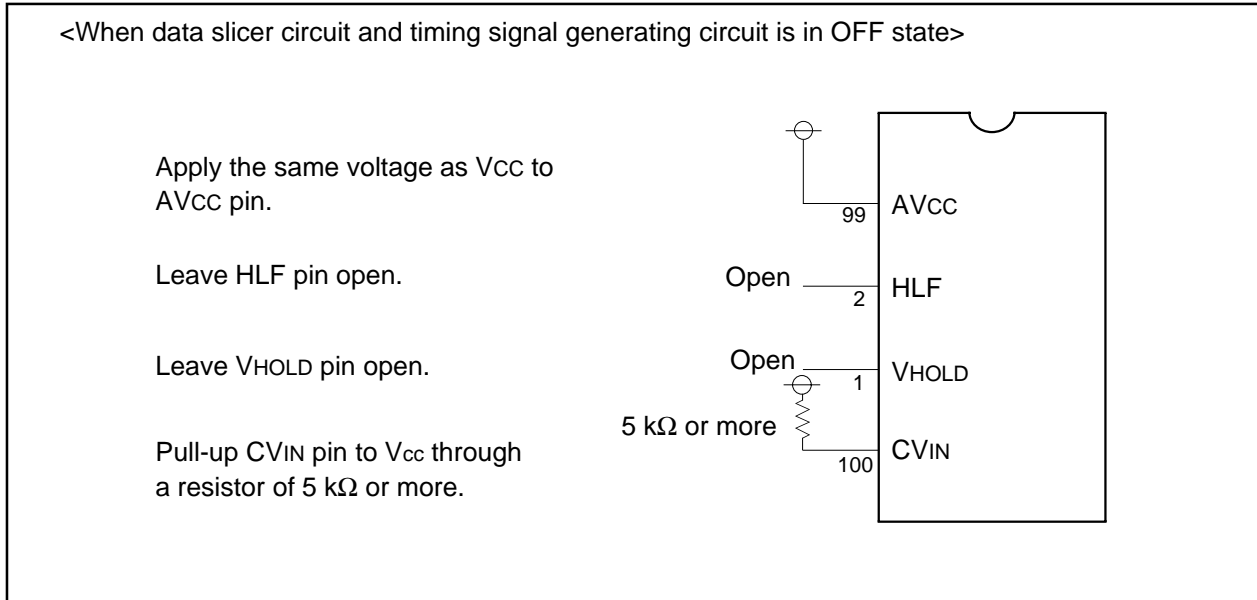


Figure 2.14.2 Termination of data slicer input/output pins when data slicer circuit and timing generating circuit is in OFF state

When both bits 0 and 2 of data slicer control register 1 (address 026016) are "1," terminate the pins as shown in Figure 2.14.3.

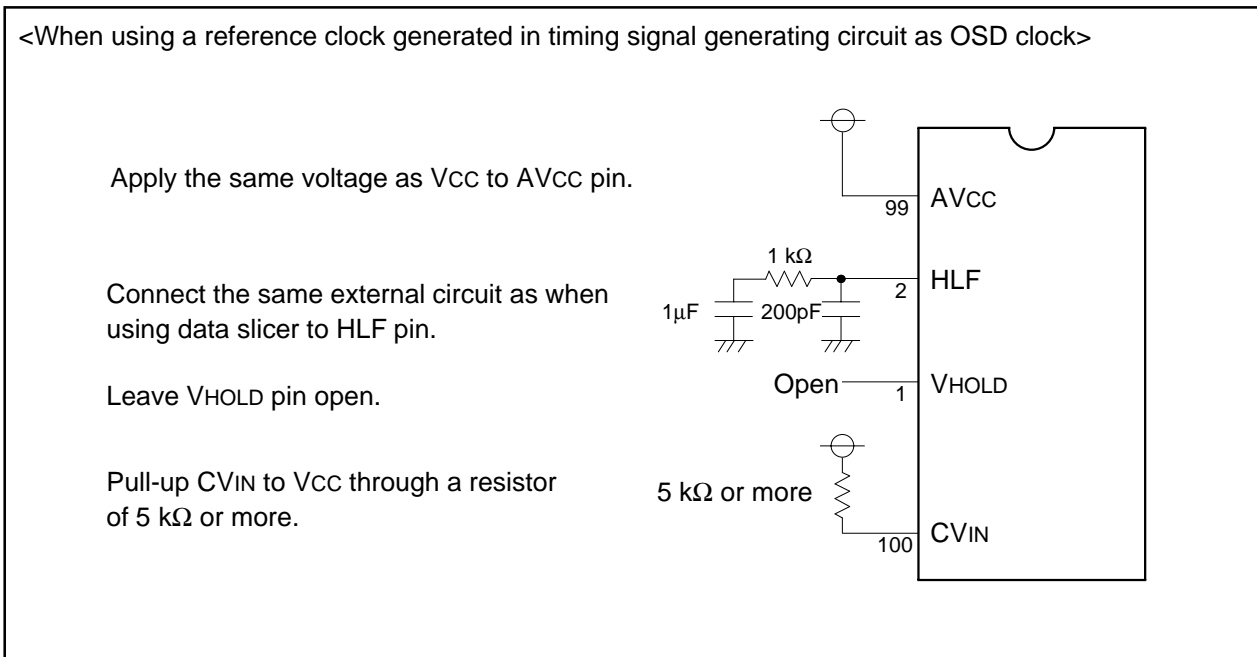


Figure 2.14.3 Termination of data slicer input/output pins when timing signal generating circuit is in ON state

Figures 2.14.4 and 2.14.5 the data slicer control registers.

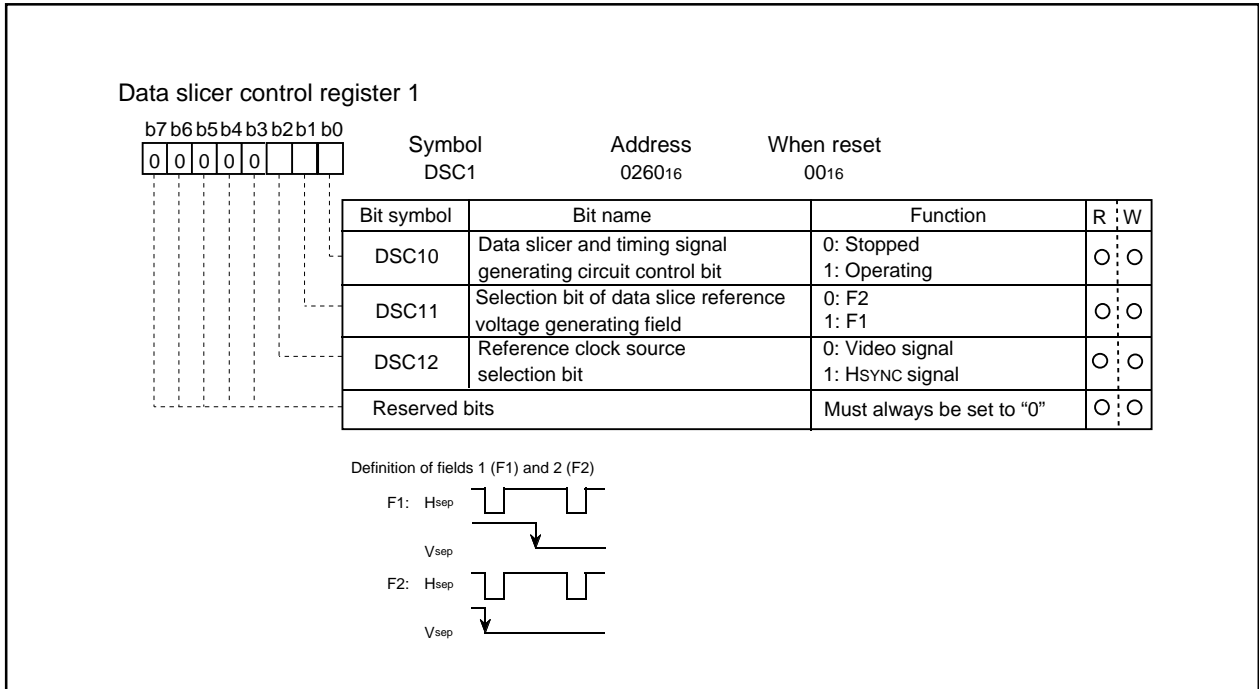


Figure 2.14.4 Data slicer control register 1

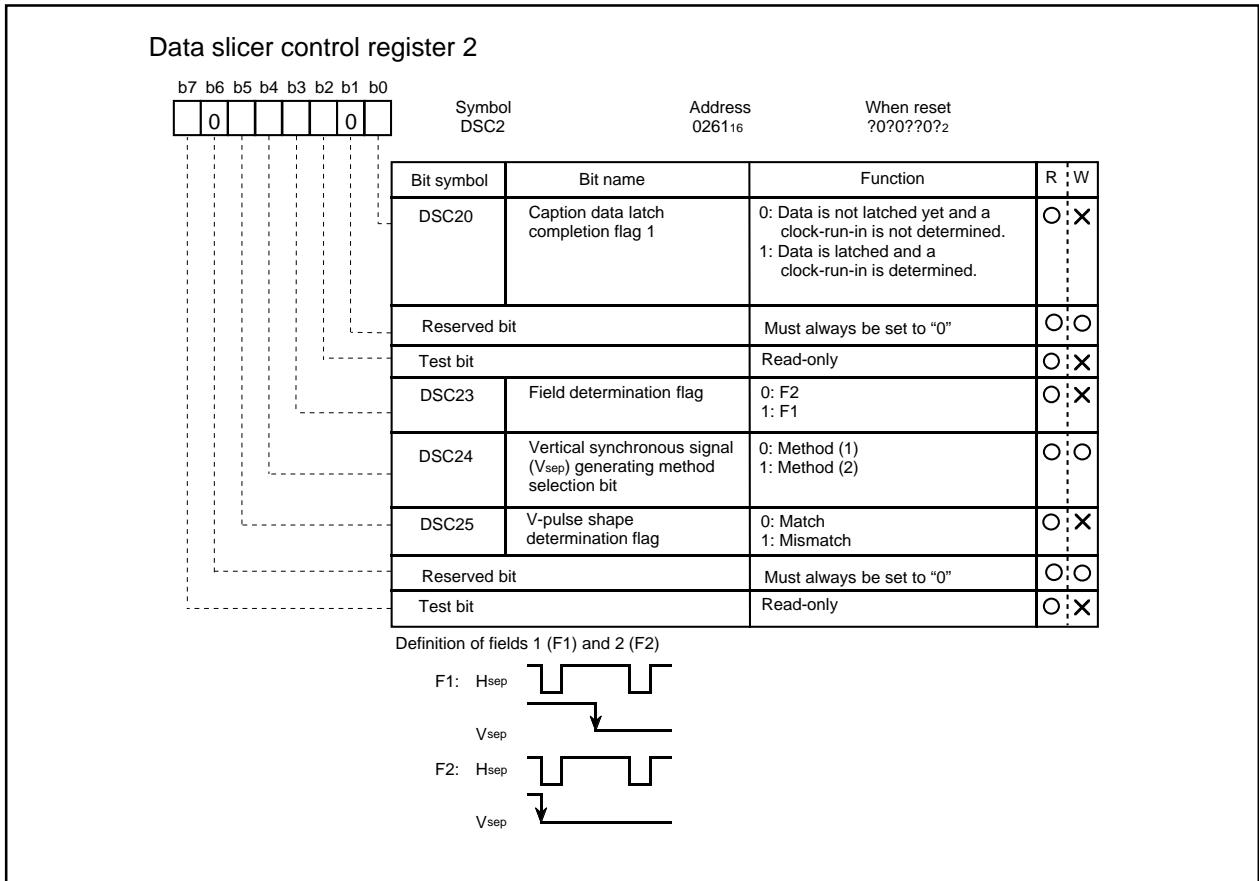


Figure 2.14.5 Data slicer control register 2

2.14.2 Clamping Circuit and Low-pass Filter

The clamp circuit clamps the sync.tip part of the composite video signal input from the CVIN pin. The low-pass filter attenuates the noise of clamped composite video signal. The CVIN pin to which composite video signal is input requires a capacitor (0.1 μ F) coupling outside. Pull down the CVIN pin with a resistor of hundreds of kilohms to 1 M Ω . In addition, we recommend to install externally a simple low-pass filter using a resistor and a capacitor at the CVIN pin (refer to Figure 2.14.1).

2.14.3 Sync Slice Circuit

This circuit takes out a composite sync signal from the output signal of the low-pass filter.

2.14.4 Synchronous Signal Separation Circuit

This circuit separates a horizontal synchronous signal and a vertical synchronous signal from the composite sync signal taken out in the sync slice circuit.

(1) Horizontal synchronous signal (Hsep)

A one-shot horizontal synchronizing signal Hsep is generated at the falling edge of the composite sync signal.

(2) Vertical synchronous signal (Vsep)

As a Vsep signal generating method, it is possible to select one of the following 2 methods by using bit 4 of the data slicer control register 2 (address 026116).

- Method 1 The "L" level width of the composite sync signal is measured. If this width exceeds a certain time, a Vsep signal is generated in synchronization with the rising of the timing signal immediately after this "L" level.
- Method 2 The "L" level width of the composite sync signal is measured. If this width exceeds a certain time, it is detected whether a falling of the composite sync signal exists or not in the "L" level period of the timing signal immediately after this "L" level. If a falling exists, a Vsep signal is generated in synchronization with the rising of the timing signal (refer to Figure 2.14.6).

Figure 2.14.6 shows a Vsep generating timing. The timing signal shown in the figure is generated from the reference clock which the timing generating circuit outputs.

Reading bit 5 of data slicer control register 2 permits determining the shape of the V-pulse portion of the composite sync signal. As shown in Figure 2.14.7, when the A level matches the B level, this bit is "0." In the case of a mismatch, the bit is "1."

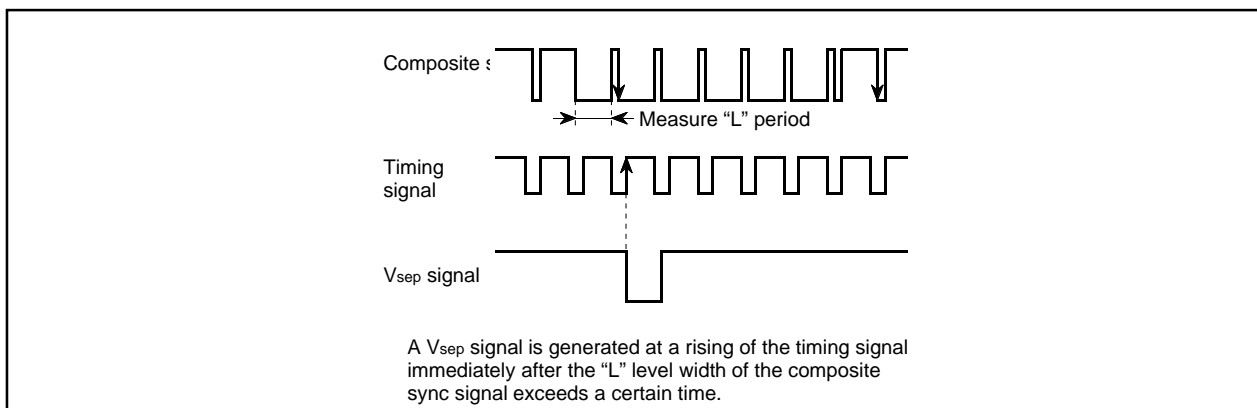


Figure 2.14.6 Vsep generating timing (method 2)

2.14.5 Timing Signal Generating Circuit

This circuit generates a reference clock which is 832 times as large as the horizontal synchronous signal frequency. It also generates various timing signals on the basis of the reference clock, horizontal synchronous signal and vertical synchronizing signal. The circuit operates by setting bit 0 of data slicer control register 1 (address 0260₁₆) to "1."

The reference clock can be used as a display clock for OSD function in addition to the data slicer. The HSYNC signal can be used as a count source instead of the composite sync signal. However, when the HSYNC signal is selected, the data slicer cannot be used. A count source of the reference clock can be selected by bit 2 of data slicer control register 1 (address 0260₁₆).

For the pins HLF, connect a resistor and a capacitor as shown in Figure 2.14.1 Make the length of wiring which is connected to these pins as short as possible so that a leakage current may not be generated.

Note: It takes a few tens of milliseconds until the reference clock becomes stable after the data slicer and the timing signal generating circuit are started. In this period, various timing signals, Hsep signals and Vsep signals become unstable. For this reason, take stabilization time into consideration when programming.

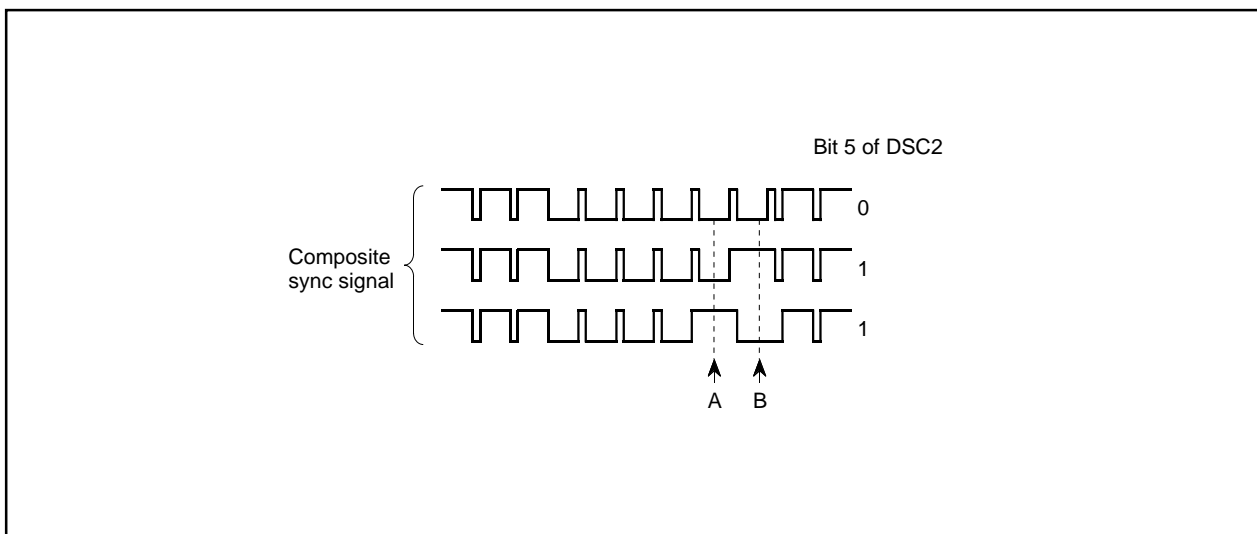


Figure 2.14.7 Determination of v-pulse waveform

2.14.6 Data Slice Line Specification Circuit

(1) Specification of data slice line

This circuit decides a line on which caption data is superimposed. The line 21 (fixed), 1 appropriate line for a period of 1 field (total 2 line for a period of 1 field), and both fields (F1 and F2) are sliced their data. The caption position register (address 0266₁₆) is used for each setting (refer to Table 2.14.1). The counter is reset at the falling edge of V_{sep} and is incremented by 1 every H_{sep} pulse. When the counter value matched the value specified by bits 4 to 0 of the caption position register, this H_{sep} is sliced.

The values of "00₁₆" to "1F₁₆" can be set in the caption position register (at setting only 1 appropriate line). Figure 2.14.8 shows the signals in the vertical blanking interval. Figure 2.14.9 shows the caption position register.

(2) Specification of line to set slice voltage

The reference voltage for slicing (slice voltage) is generated for the clock run-in pulse in the particular line (refer to Table 2.14.1). The field to generate slice voltage is specified by bit 1 of data slicer control register 1. The line to generate slice voltage 1 field is specified by bits 6, 7 of the caption position register (refer to Table 2.14.1).

(3) Field determination

The field determination flag can be read out by bit 3 of data slicer control register 2. This flag change at the falling edge of V_{sep} .

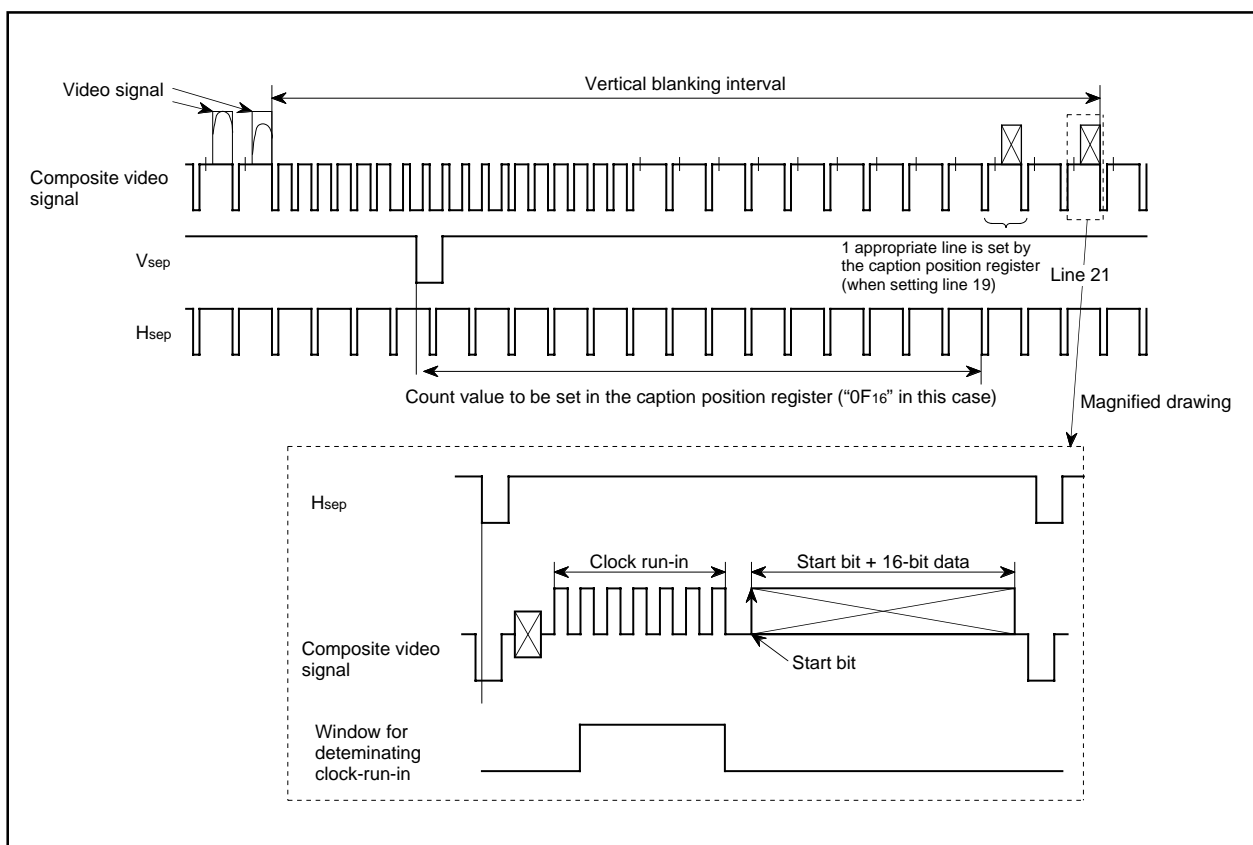


Figure 2.14.8 Signals in vertical blanking interval

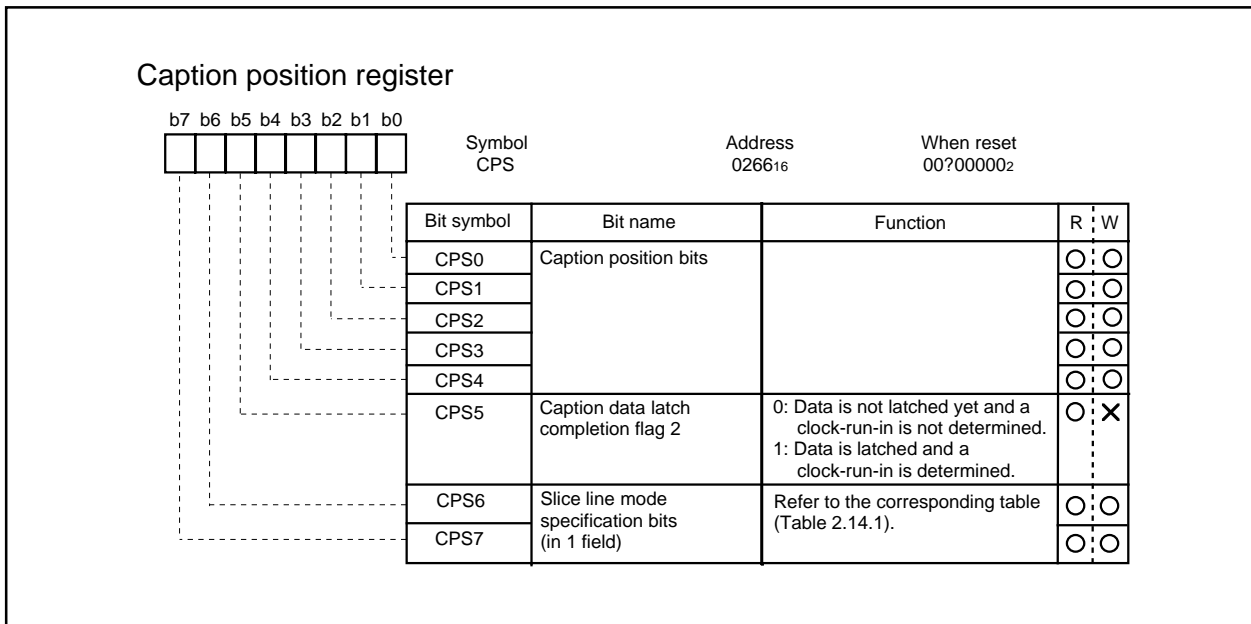


Figure 2.14.9 Caption position register

Table 2.14.1 Specification of data slice line

| CPS | | Field and Line to Be Sliced Data | Field and Line to Generate Slice Voltage |
|-----|----|---|---|
| b7 | b6 | | |
| 0 | 0 | <ul style="list-style-type: none"> Both fields of F1 and F2 Line 21 and a line specified by bits 4 to 0 of CPS (total 2 lines) (See note 2) | <ul style="list-style-type: none"> Field specified by bit 1 of DSC1 Line 21 (total 1 line) |
| 0 | 1 | <ul style="list-style-type: none"> Both fields of F1 and F2 A line specified by bits 4 to 0 of CPS (total 1 line) (See note 3) | <ul style="list-style-type: none"> Field specified by bit 1 of DSC1 A line specified by bits 4 to 0 of CPS (total 1 line) (See note 3) |
| 1 | 0 | <ul style="list-style-type: none"> Both fields of F1 and F2 Line 21 (total 1 line) | <ul style="list-style-type: none"> Field specified by bit 1 of DSC1 Line 21 (total 1 line) |
| 1 | 1 | <ul style="list-style-type: none"> Both fields of F1 and F2 Line 21 and a line specified by bits 4 to 0 of CPS (total 2 lines) (See note 2) | <ul style="list-style-type: none"> Field specified by bit 1 of DSC1 Line 21 and a line specified by bits 4 to 0 of CPS (total 2 lines) (See note 2) |

- Notes 1:** DSC is data slicer control register 1.
 CPS is caption position register.
2: Set "00₁₆" to "10₁₆" to bits 4 to 0 of CPS.
3: Set "00₁₆" to "1F₁₆" to bits 4 to 0 of CPS.

2.14.7 Reference Voltage Generating Circuit and Comparator

The composite video signal clamped by the clamping circuit is input to the reference voltage generating circuit and the comparator.

(1) Reference voltage generating circuit

This circuit generates a reference voltage (slice voltage) by using the amplitude of the clock run-in pulse in line specified by the data slice line specification circuit. Connect a capacitor between the VHOLD pin and the VSS pin, and make the length of wiring as short as possible so that a leakage current may not be generated.

(2) Comparator

The comparator compares the voltage of the composite video signal with the voltage (reference voltage) generated in the reference voltage generating circuit, and converts the composite video signal into a digital value.

2.14.8 Start Bit Detecting Circuit

This circuit detects a start bit at line decided in the data slice line specification circuit.

The detection of a start bit is described below.

- ① A sampling clock is generated by dividing the reference clock output by the timing signal.
- ② A clock run-in pulse is detected by the sampling clock.
- ③ After detection of the pulse, a start bit pattern is detected from the comparator output.

2.14.9 Clock Run-in Determination Circuit

This circuit determines clock run-in by counting the number of pulses in a window of the composite video signal.

The reference clock count value in one pulse cycle is stored in bits 3 to 7 of the clock run-in detect register (address 0269₁₆). Read out these bits after the occurrence of a data slicer interrupt (refer to 2.14.12 Interrupt request generating circuit).

Figure 2.14.10 shows the structure of clock run-in detect register.

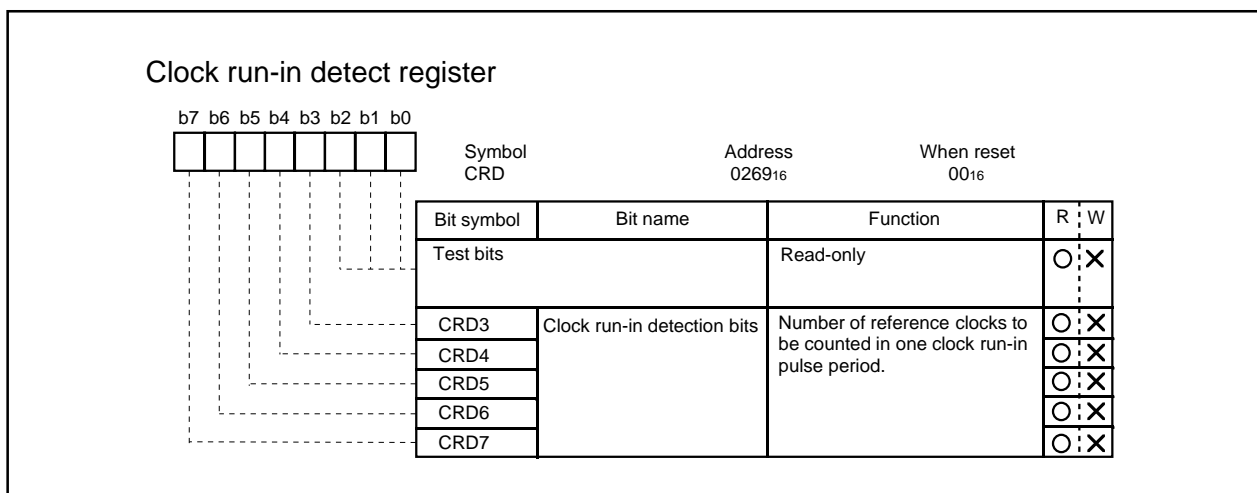


Figure 2.14.10 Clock run-in detect register

2.14.10 Data Clock Generating Circuit

This circuit generates a data clock synchronized with the start bit detected in the start bit detecting circuit. The data clock stores caption data to the 16-bit shift register. When the 16-bit data has been stored and the clock run-in determination circuit determines clock run-in, the caption data latch completion flag is set. This flag is reset at a falling of the vertical synchronous signal (V_{sep}).

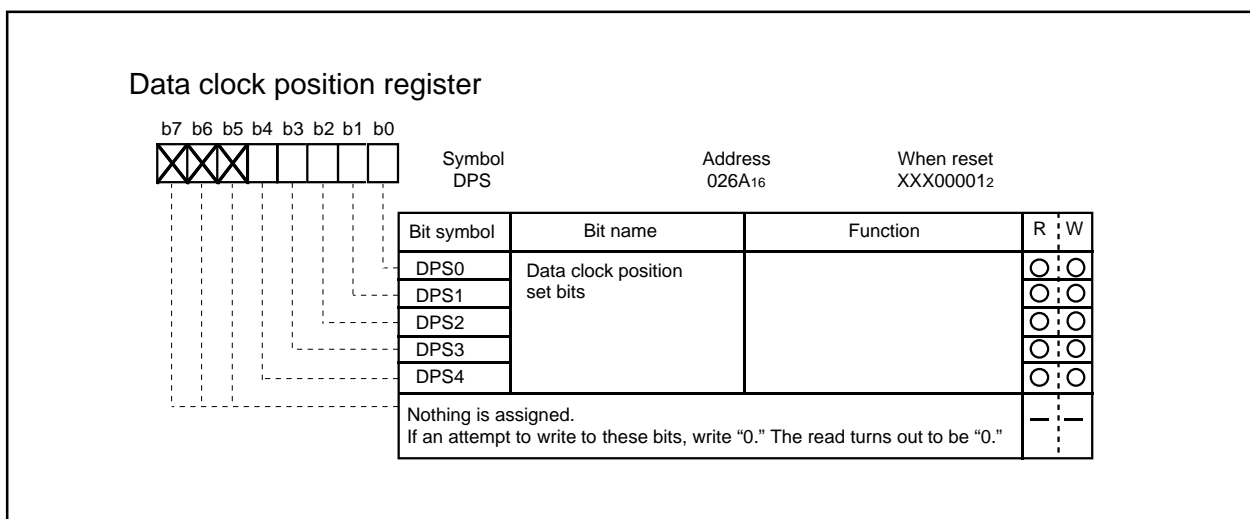


Figure 2.14.11 Data clock position register

2.14.11 16-bit Shift Register

The caption data converted into a digital value by the comparator is stored into the 16-bit shift register in synchronization with the data clock. The contents of the stored caption data can be obtained by reading out the caption data register 1 (addresses 0263₁₆, 0262₁₆) and caption data register 2 (addresses 0265₁₆, 0264₁₆). These registers are reset to "0" at a falling of V_{sep} . Read out data registers 1 and 2 after the occurrence of a data slicer interrupt (refer to "2.14.12 Interrupt request generating circuit").

2.14.12 Interrupt Request Generating Circuit

The interrupt requests as shown in Table 2.14.3 are generated by combination of the following bits; bits 6 and 7 of the caption position register (address 026616). Read out the contents of data registers 1, 2 and the contents of bits 3 to 7 of the clock run-in detect register after the occurrence of a data slicer interrupt request.

Table 2.14.2 Contents of caption data latch completion flag and 16-bit shift register

| Slice Line Specification Mode | | Contents of Caption Data Latch Completion Flag | | Contents of 16-bit Shift Register | |
|-------------------------------|-------|--|--|---|---|
| CPS | | Completion Flag 1 (bit 0 of DSC2) | Completion Flag 2 (bit 5 of CPS) | Caption Data Register 1 | Caption Data Register 2 |
| bit 7 | bit 6 | | | | |
| 0 | 0 | Line 21 | A line specified by bits 4 to 0 of CPS | 16-bit data of line 21 | 16-bit data of a line specified by bits 4 to 0 of CPS |
| 0 | 1 | A line specified by bits 4 to 0 of CPS | Invalid | 16-bit data of a line specified by bits 4 to 0 of CPS | Invalid |
| 1 | 0 | Line 21 | Invalid | 16-bit data of line 21 | Invalid |
| 1 | 1 | Line 21 | A line specified by bits 4 to 0 of CPS | 16-bit data of line 21 | 16-bit data of a line specified by bits 4 to 0 of CPS |

CPS: Caption position register
 DSC2: Data slicer control register 2

Table 2.14.3 Occurrence sources of Interrupt request

| CPS | | Occurrence Sources of Interrupt Request at End of Data Slice Line |
|-----|----|---|
| b7 | b6 | |
| 0 | 0 | After slicing line 21 |
| | 1 | After a line specified by bits 4 to 0 of CPS |
| 1 | 0 | After slicing line 21 |
| | 1 | After slicing line 21 |

CPS: Caption position register

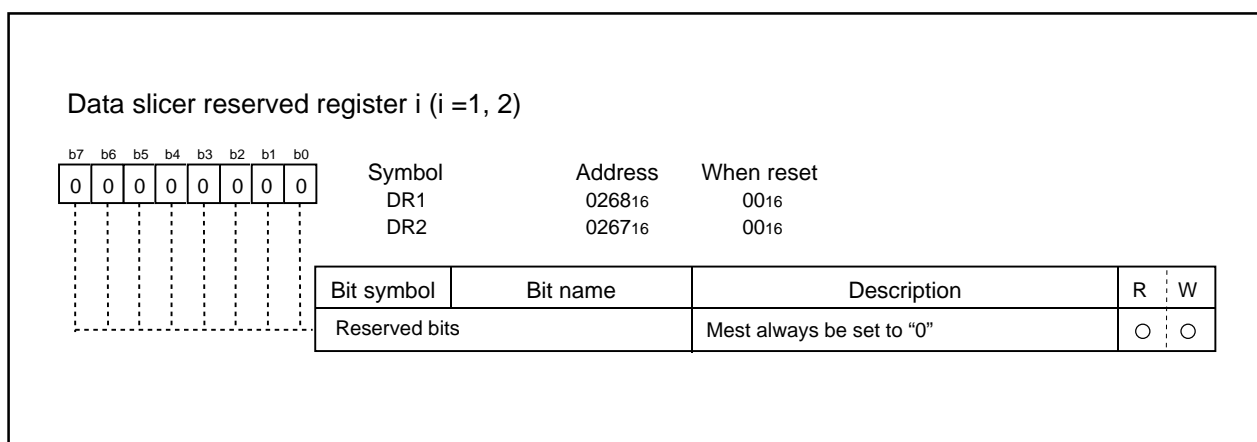


Figure 2.14.12 Data slicer reserved register i (i = 1, 2)

2.15 HSYNC Counter

The synchronous signal counter counts HSYNC from HSYNC count input pins (HC0/P7₅, HC1/P7₇) as a count source.

The count value in a certain time (T time; 1024 μs, 2048 μs, 4096 μs and 8192 μs) divided system clock f₃₂ is stored into the 8-bit latch.

Accordingly, the latch value changes in the cycle of T time. When the count value exceeds “FF₁₆,” “FF₁₆” is stored into the latch.

The latch value can be obtained by reading out the HSYNC counter latch (address 027F₁₆). A count source and count update cycle (T time) are selected by bits 0, 3 and 4 of the HSYNC counter register.

Figure 2.15.1 shows the HSYNC counter and Figure 2.15.2 shows the synchronous signal counter block diagram.

Note: When using the HSYNC counter, set bit 7 of the peripheral mode register (address 027D₁₆) according to the main clock frequency.

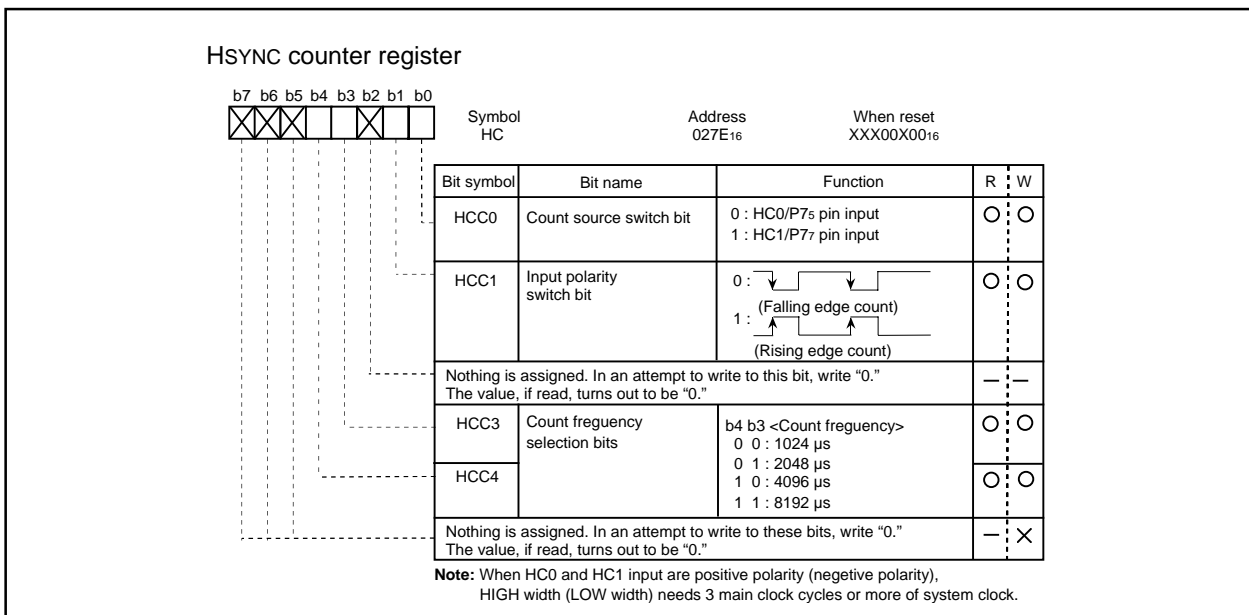


Figure 2.15.1 HSYNC counter register

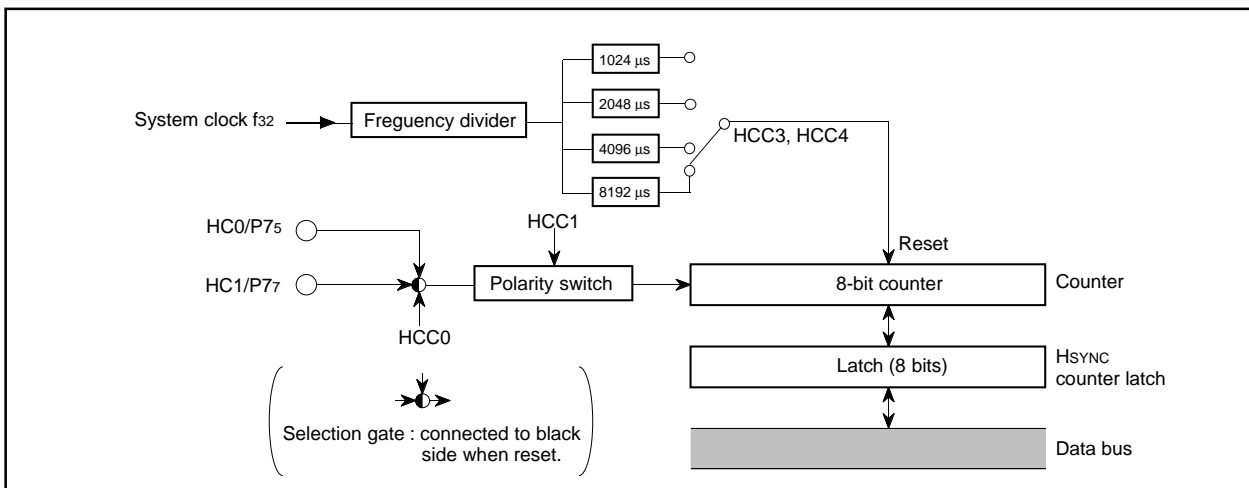


Figure 2.15.2 HSYNC counter block diagram

2.16 OSD Functions

Table 2.16.1 outlines the OSD functions of this microcomputer. This OSD function can display the following: the block display (32 characters X 16 lines or 42 characters X 16 lines) and the SPRITE display, and can display the both display at the same time. There are 3 display modes and they are selected by a block unit. The display modes are selected by block control register i (i = 1 to 16). The features of each display are described below.

Note: When using OSD function, select “No-division mode” as BCLK operating mode and set the main clock frequency to $f(XIN) = 10 \text{ MHz}$.

Table 2.16.1 Features of each display style

| Display style Parameter | | Block display | | | | SPRITE display | |
|---------------------------------------|-------------------|---|---|---|---|---|--|
| | | CC mode (Closed caption mode) | OSD mode (On-screen display mode) | | | | CDOSD mode (Color dot on-screen display mode) |
| | | | OSDS mode | OSDP mode | OSDL mode | | |
| Number of display characters | | 32 characters X 16 lines/42 characters X 16 lines | | | | 1 character X 2 lines | |
| Dot structure | | 16 X 20 dots (Character display area: 16 X 26 dots) | 16 X 20 dots 12 X 20 dots 8 X 20 dots 4 X 20 dots | 24 X 32 dots | 16 X 26 dots | 32 X 20 dots | |
| Kinds of character ROM | OSDL enable mode | 254 kinds | | 254 kinds | 126 kinds | 2 kinds of RAM font | |
| | OSDL disable mode | 508 kinds | | 254 kinds | 126 kinds | | |
| Kinds of character sizes (See note 1) | | 4 kinds | 14 kinds | 12 kinds | 14 kinds | 8 kinds | |
| Pre-divide ratio (Note) | | X 1, X 2 | X 1, X 2, X 3 | | | X 1, X 2 | |
| Dot size | | 1Tc X 1/2H, 1Tc X 1H | 1Tc X 1/2H, 1Tc X 1H, 1.5Tc X 1/2H, 1.5Tc X 1H, 2Tc X 2H, 3Tc X 3H | 1Tc X 1/2H, 1Tc X 1H, 2Tc X 2H, 3Tc X 3H | 1Tc X 1/2H, 1Tc X 1H, 1.5Tc X 1/2H, 1.5Tc X 1H, 2Tc X 2H, 3Tc X 3H | 1Tc X 1/2H, 1Tc X 1H, 2Tc X 2H, 3Tc X 3H | |
| Attribute | | Smooth italic, under line, flash | Border | | | | |
| Character font coloring | | 1 screen: 8 kinds (a character unit) Max. 512 kinds | 1 screen: 16 kinds (a character unit) Max. 512 kinds | 1 screen: 16 kinds (a dot unit) (only specified dots are colored by a character unit) Max. 512 kinds | | 1 screen: 16 kinds (a dot unit) Max. 512 kinds | |
| Character background coloring | | Possible (a character unit, 1 screen: 4 kinds, Max. 512 kinds) | Possible (a character unit, 1 screen: 16 kinds, Max. 512 kinds) | | | | |
| Display layer | | Layer 1 | Layers 1, 2 | Layer 1 | Layers 1, 2 | Layer 3 (with highest priority) | |
| OSD output (See note 2) | | Analog R, G, B output (each 8 adjustment levels: 512 colors), Digital OUT1, OUT2 output | | | | | |
| Raster coloring | | Possible (a screen unit, max 512 kinds) | | | | | |
| Other function (See note 3) | | Auto solid space function | Triple layer OSD function, window function, blank function | | | | |
| Display expansion (multiline display) | | Possible | | | | | |

- Notes**
1: The character size is specified with dot size and pre-divide ratio (refer to "2.16.3 Dot Size").
2: As for SPRITE display, OUT2 is not output.
3: As for SPRITE display, the window function does not operate.
4: The divide ratio of the frequency divider (the pre-divide circuit) is referred as "pre-divide ratio" hereafter.

The OSD circuit has an extended display mode. This mode allows multiple lines (16 lines or more) to be displayed on the screen by interrupting the display each time one line is displayed and rewriting data in the block for which display is terminated by software.

Figure 2.16.1 shows the display-enable fonts for each display style. Figure 2.16.2 shows the block diagram of the OSD circuit. Figure 2.16.3 shows the OSD control register 1. Figure 2.16.4 shows the block control register i.

| Display Styles | Display-enable Fonts |
|----------------|---|
| CC Mode | |
| OSDS Mode | |
| OSDP Mode | <p>* : Only character codes ** : Blank font</p> |
| OSDL Mode | |
| CDOSD Mode | |
| SPRITE | |

Figure 2.16.1 Display-enable fonts for each display style

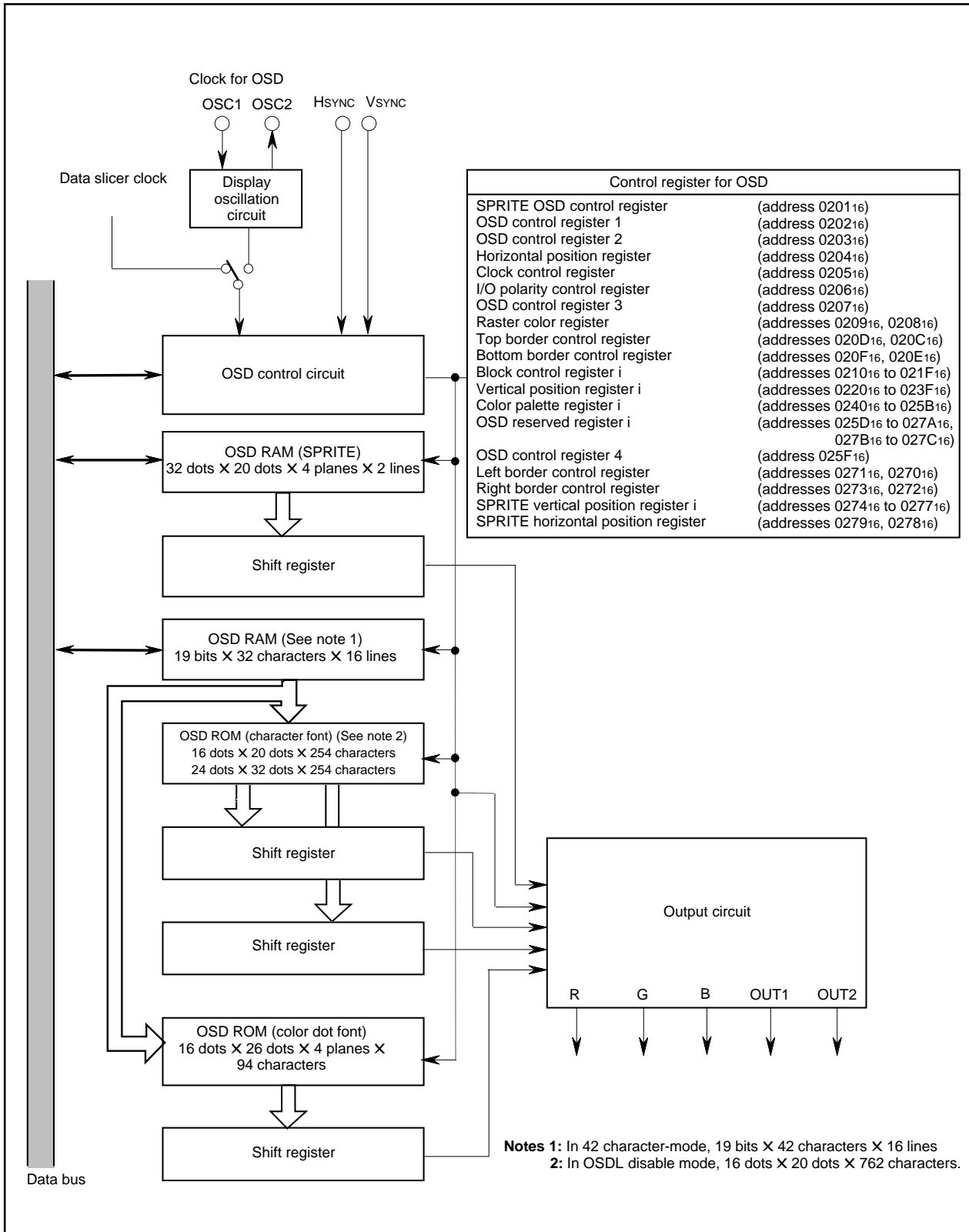


Figure 2.16.2 Block diagram of OSD circuit

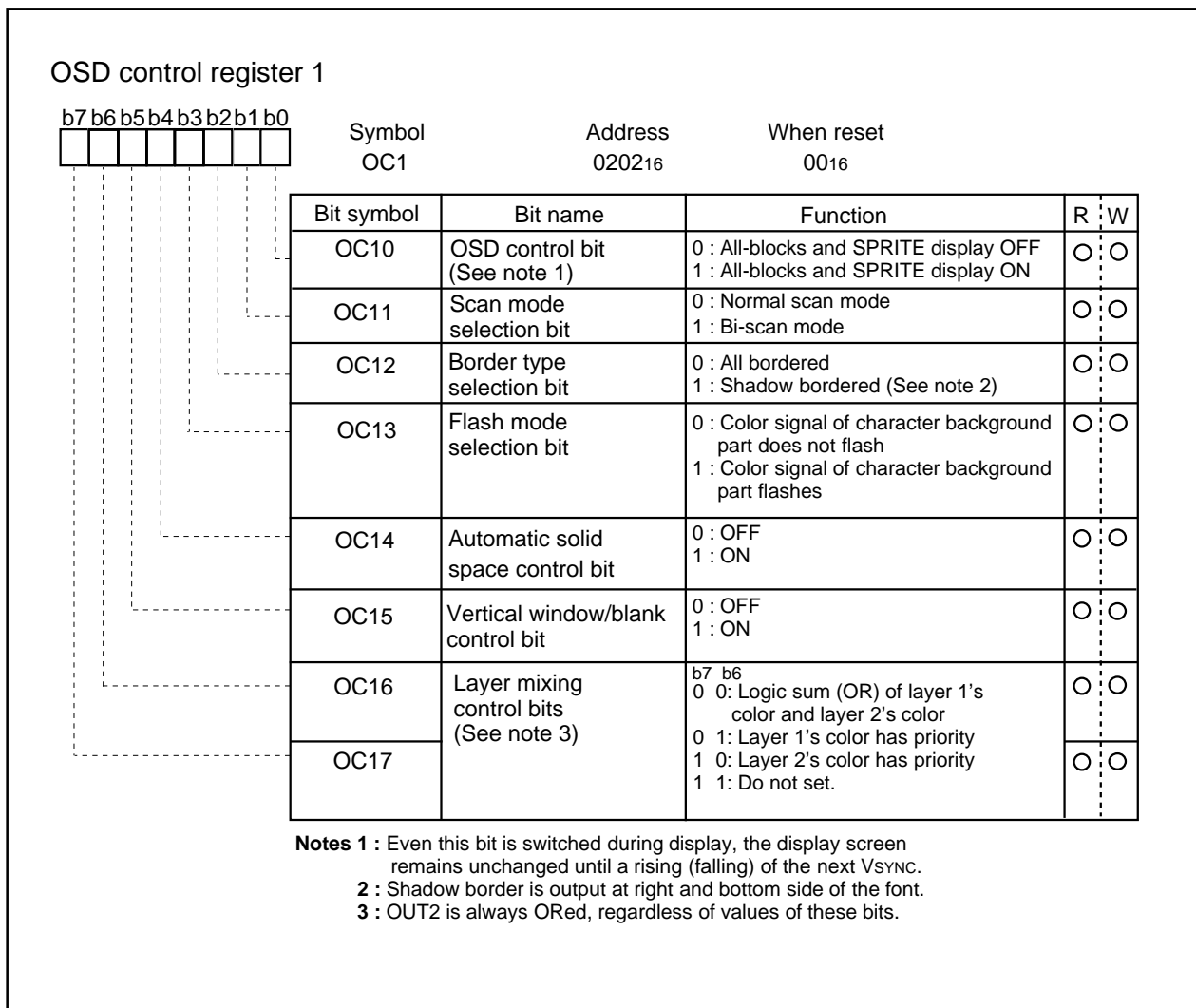


Figure 2.16.3 OSD control register 1

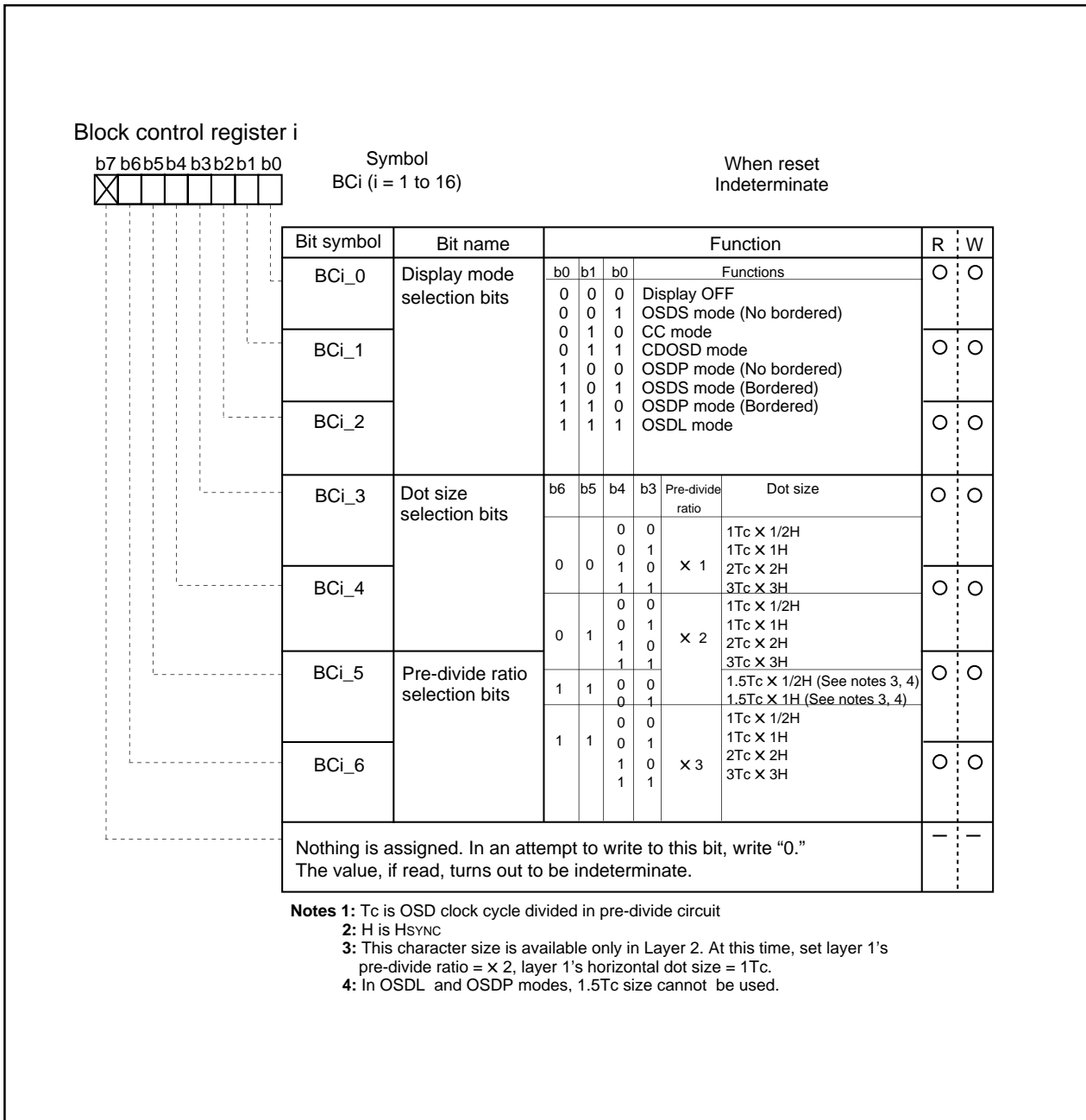


Figure 2.16.4 Block control register i (i = 0 to 16)

2.16.1 Triple Layer OSD

Three built-in layers of display screens accommodate triple display of channels, volume, etc., closed caption, and sprite displays within layers 1 to 3.

The layer to be displayed in each block is selected by bit 0 or 1 of the OSD control register 2 for each display mode (refer to Figure 2.16.7). Layer 3 always displays the sprite display.

When the layer 1 block and the layer 2 block overlay, the screen is composed with layer mixing by bit 6 or 7 of the OSD control register 1, as shown in Figure 2.16.5. Layer 3 always takes display priority of layers 1 and 2.

Notes 1: When mixing layer 1 and layer 2, note Table 2.16.2.

2: OSDP mode is always displayed on layer 1. And also, it cannot be overlapped with layer 2's block.

3: OUT2 is always ORed, regardless of values of bits 6, 7 of the OSD control register 1. And besides, even when OUT2 (layer 1 and layer 2) overlaps with SPRITE display (layer 3), OUT2 is output without masking.

Table 2.16.2 Mixing layer 1 and layer 2

| Parameter \ Block | Block in Layer 1 | Block in Layer 2 | |
|-----------------------------------|---|--|---|
| Display mode | CC, OSDS/L, CDOSD mode | OSDS/L, CDOSD mode | |
| Pre-divide ratio | X 1, X 2 (CC mode) X 1 to X 3 (OSD, CDOSD mode) | Same as layer 1 (See note) | |
| Dot size | 1Tc X 1/2H, 1Tc X 1H (CC mode) | Pre-divide ratio = X 1 | Pre-divide ratio = X 2 |
| | | 1Tc X 1/2H 1Tc X 1H | 1TcX1/2H, 1.5TcX1/2H 1TcX1H, 1.5TcX1H(Seenote) |
| | 1Tc X 1H, 1Tc X 1/2H, 2Tc X 2H, 3Tc X 3H (OSDS/L, CDOSD mode) | <ul style="list-style-type: none"> • Same size as layer 1 • 1.5Tc can be selected only when: layer 1's pre-divide ratio = X 2 AND layer 1's horizontal dot size = 1Tc. As this time, vertical dot size is the same as layer 1. | |
| Horizontal display start position | Arbitrary | Same position as layer 1 | |
| Vertical display start position | Arbitrary However, when dot size is 2Tc X 2H or 2Tc X 3H, set difference between vertical display position of layer 1 and that of layer 2 as follows. <ul style="list-style-type: none"> •2Tc X 2H: 2H units •3Tc X 3H: 3H units | | |

Note: In the OSDL mode, 1.5Tc size cannot be used.

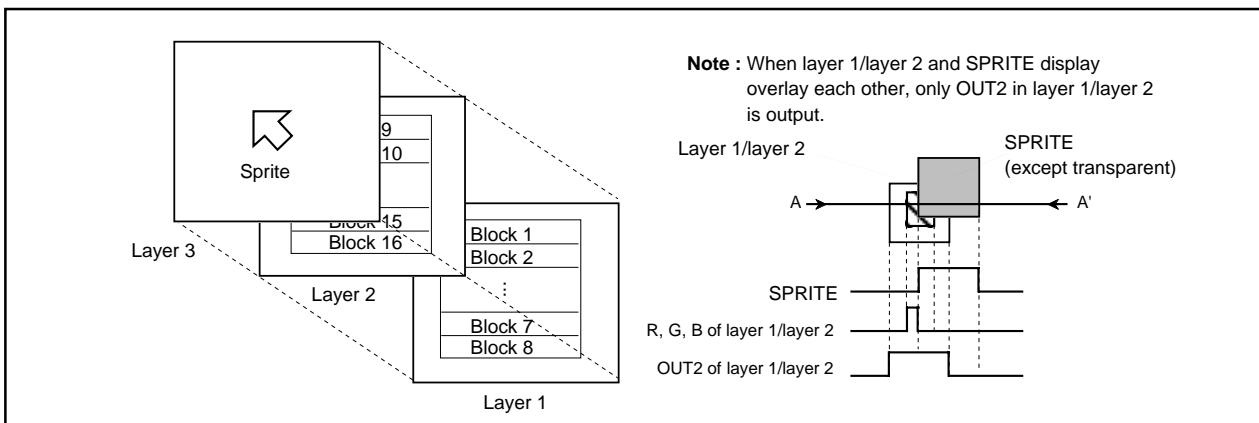


Fig 2.16.5 Triple layer OSD

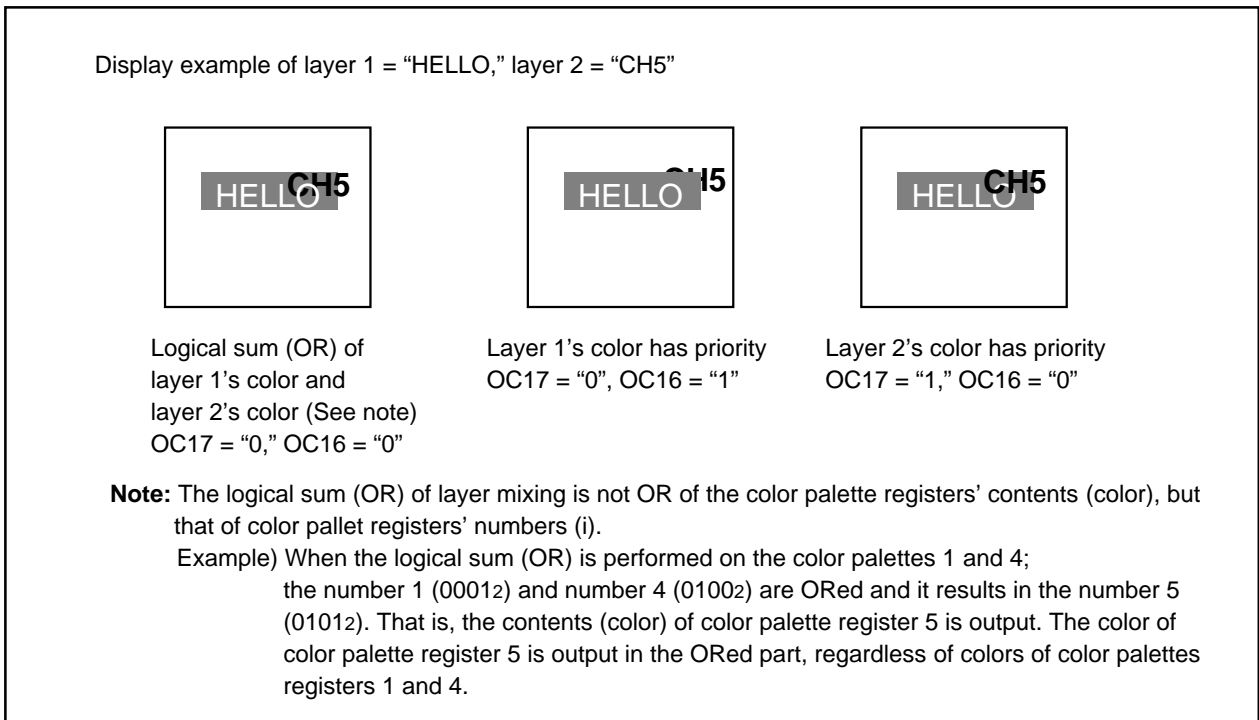


Figure 2.16.6 Display example of triple layer OSD

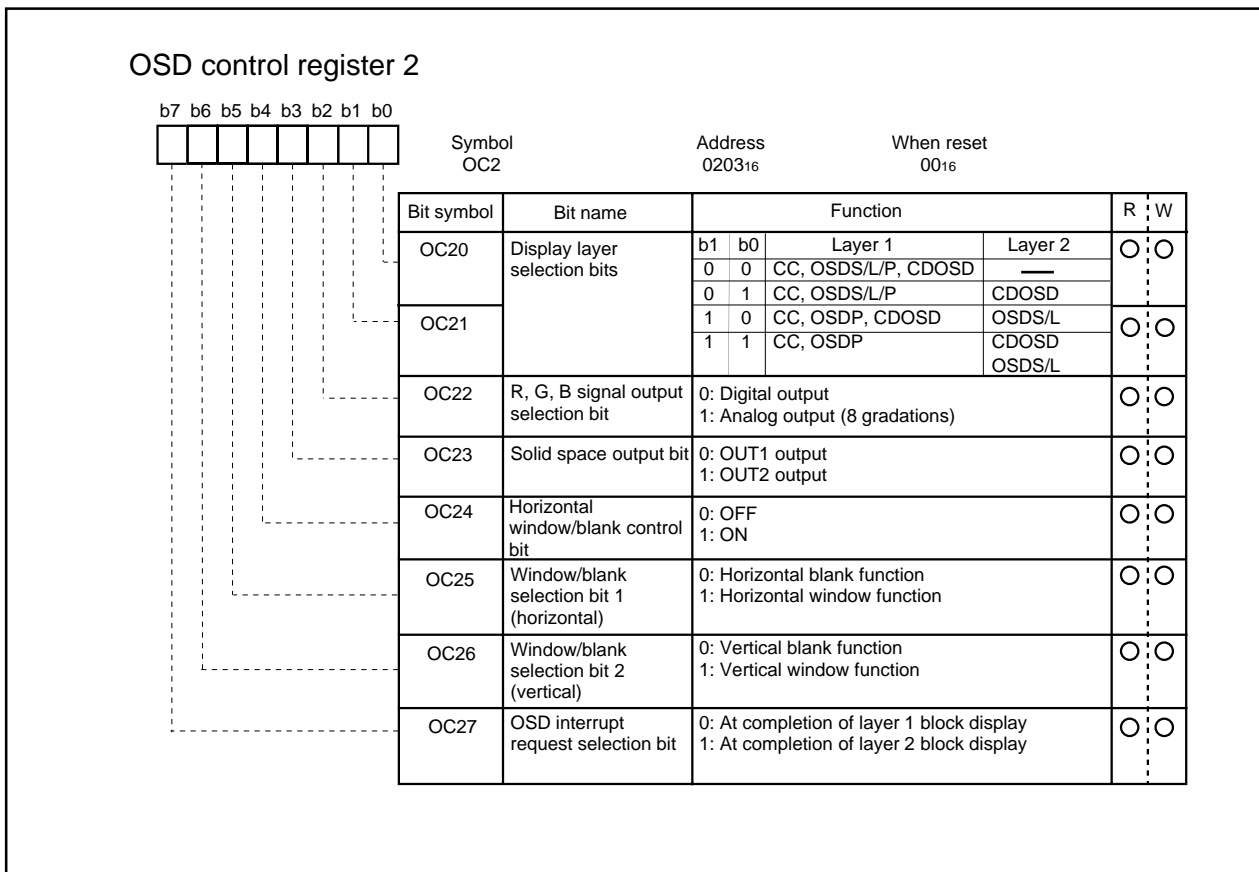


Figure 2.16.7 OSD control register 2

2.16.2 Display Position

The display positions of characters are specified by a block. There are 16 blocks, blocks 1 to 16. Up to 32 characters (32-character mode)/42 characters (42-character mode)/ can be displayed in each block (refer to 2.16.6 Memory for OSD).

The display position of each block can be set in both horizontal and vertical directions by software.

The display position in the horizontal direction can be selected for all blocks in common from 256-step display positions in units of 4 TOSC (TOSC = OSD oscillation cycle).

The display position in the vertical direction for each block can be selected from 1024-step display positions in units of 1 TH (TH = HSYNC cycle).

Blocks are displayed in conformance with the following rules:

- When the display position is overlapped with another block in the same layer (Figure 2.16.8 (b)), a low block number (1 to 16) is displayed on the front.
- When another block display position appears while one block is displayed in the same layer (Figure 2.16.8 (c)), the block with a larger set value as the vertical display start position is displayed. However, do not display block with the dot size of $2Tc \times 2H$ or $3Tc \times 3H$ during display period (*) of another block.
 - * In the case of OSDS/P mode block: 20 dots in vertical from the vertical display start position.
 - * In the case of OSDL mode block: 32 dots in vertical from the vertical display start position.
 - * In the case of CC or CDOSD mode block: 26 dots in vertical from the vertical display start position.

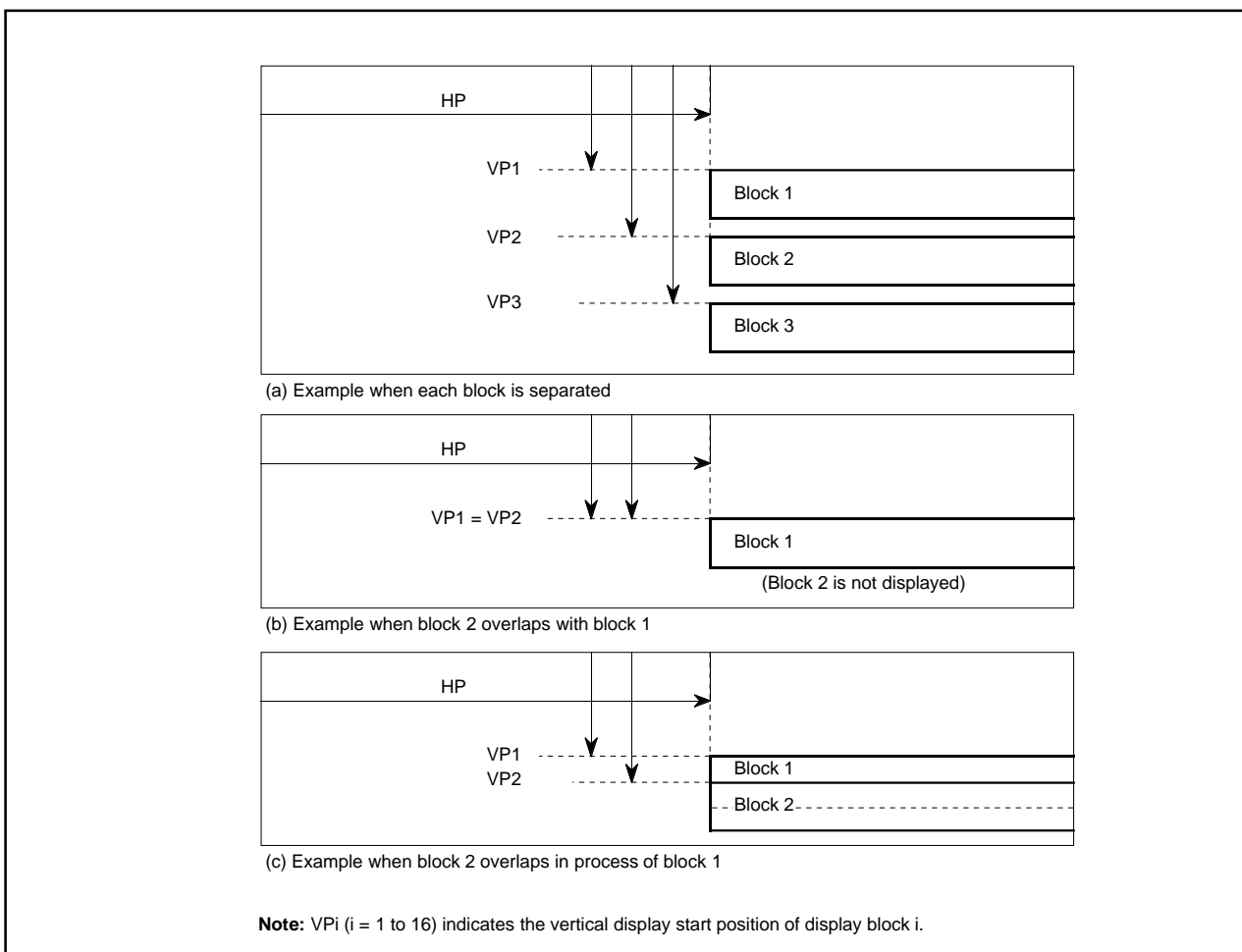


Figure 2.16.8 Display position

The display position in the vertical direction is determined by counting the horizontal sync signal (HSYNC). At this time, when VSYNC and HSYNC are positive polarity (negative polarity), it starts to count the rising edge (falling edge) of HSYNC signal from after fixed cycle of rising edge (falling edge) of VSYNC signal. So interval from rising edge (falling edge) of VSYNC signal to rising edge (falling edge) of HSYNC signal needs enough time (2 X BCLK cycles or more) for avoiding jitter. The polarity of HSYNC and VSYNC signals can select with the I/O polarity control register (address 020616).

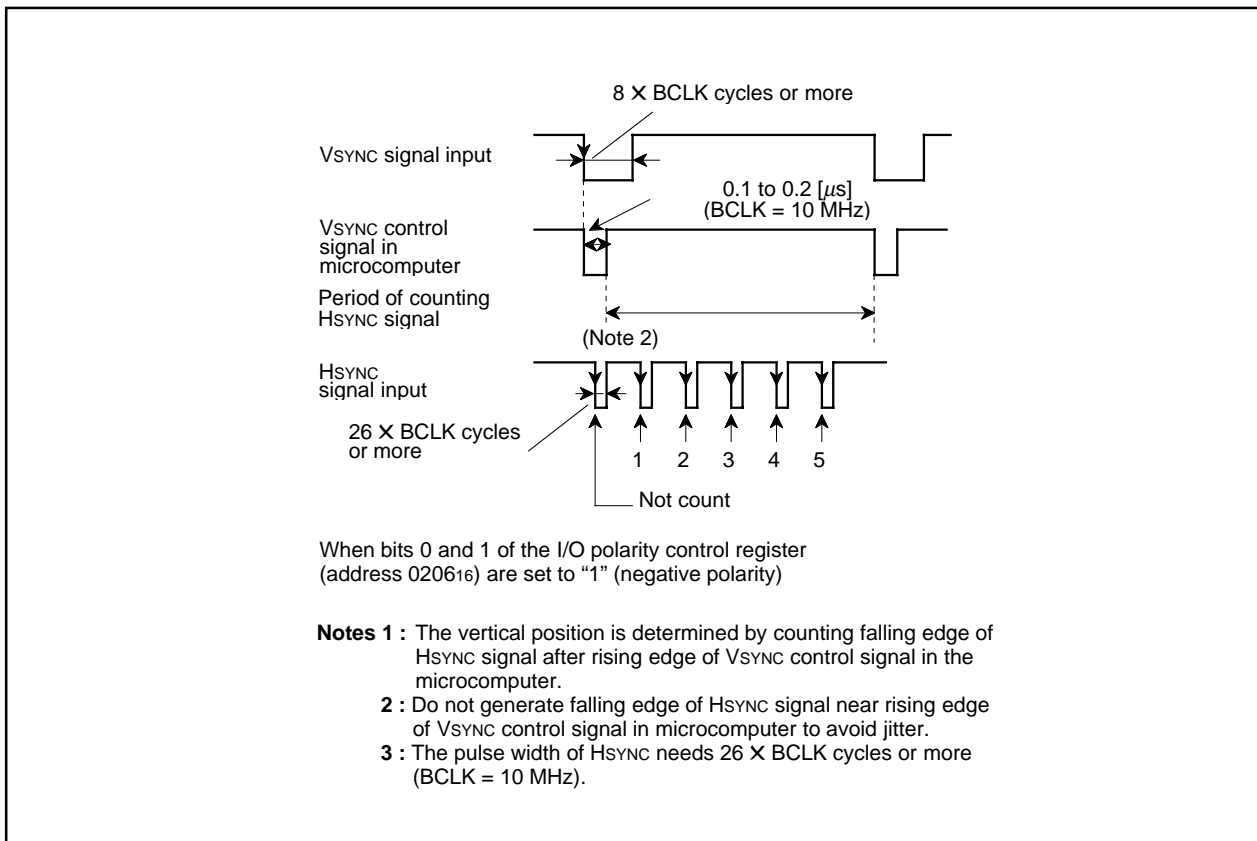


Figure 2.16.9 Supplement explanation for display position

The vertical position for each block can be set in 1024 steps (where each step is 1_{TH} (TH: HSYNC cycle)) as values “002₁₆” to “3FF₁₆” in vertical position register i (i = 1 to 16) (addresses 0220₁₆ to 023F₁₆). The vertical position register i is shown in Figure 2.16.10.

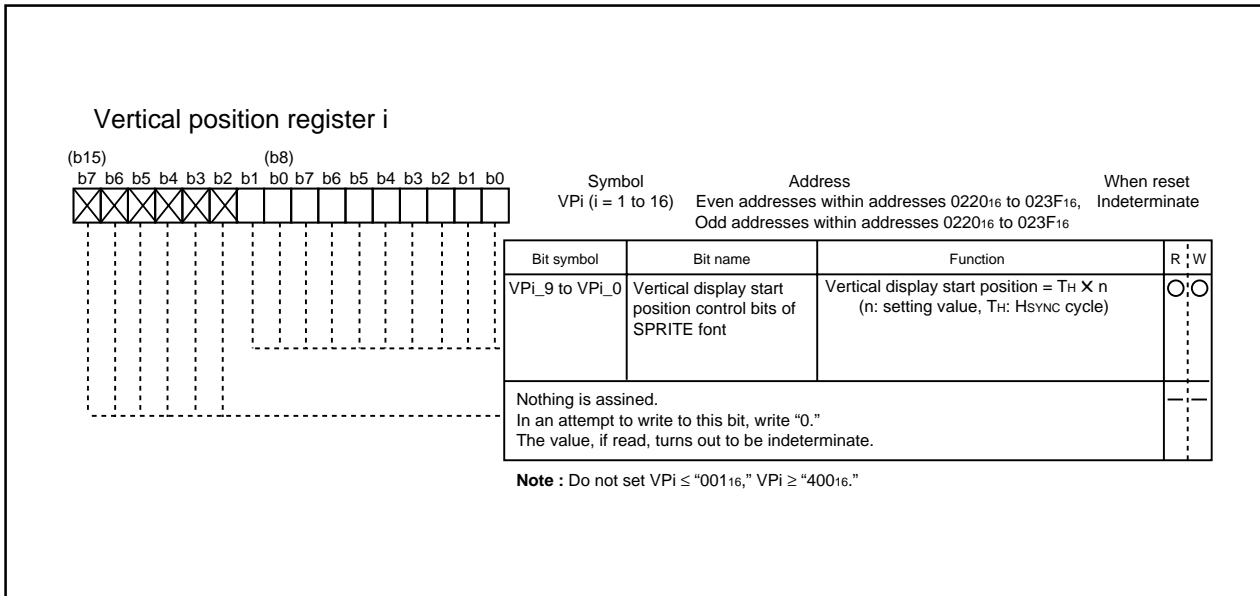


Figure 2.16.10 Vertical position register i (i = 1 to 16)

The horizontal position is common to all blocks, and can be set in 256 steps (where 1 step is 4_{Tosc}, T_{osc} being OSD oscillation cycle) as values “00₁₆” to “FF₁₆” in bits 0 to 7 of the horizontal position register (address 0204₁₆). The horizontal position register is shown in Figure 2.16.11.

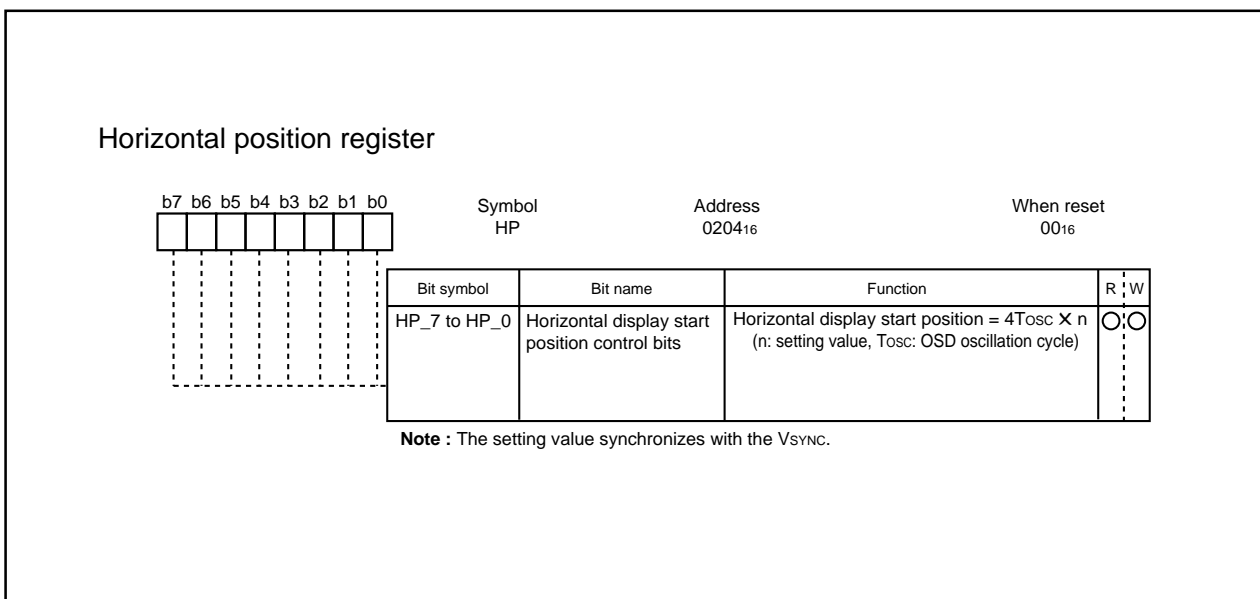


Figure 2.16.11 Horizontal position register

Note : $1T_c$ (T_c : OSD clock cycle divided in pre-divide circuit) gap occurs between the horizontal display start position set by the horizontal position register and the most left dot of the 1st block. Accordingly, when 2 blocks have different pre-divide ratios, their horizontal display start position will not match.

Ordinary, this gap is $1T_c$ regardless of character sizes, however, the gap is $1.5T_c$ only when the character size is $1.5T_c$.

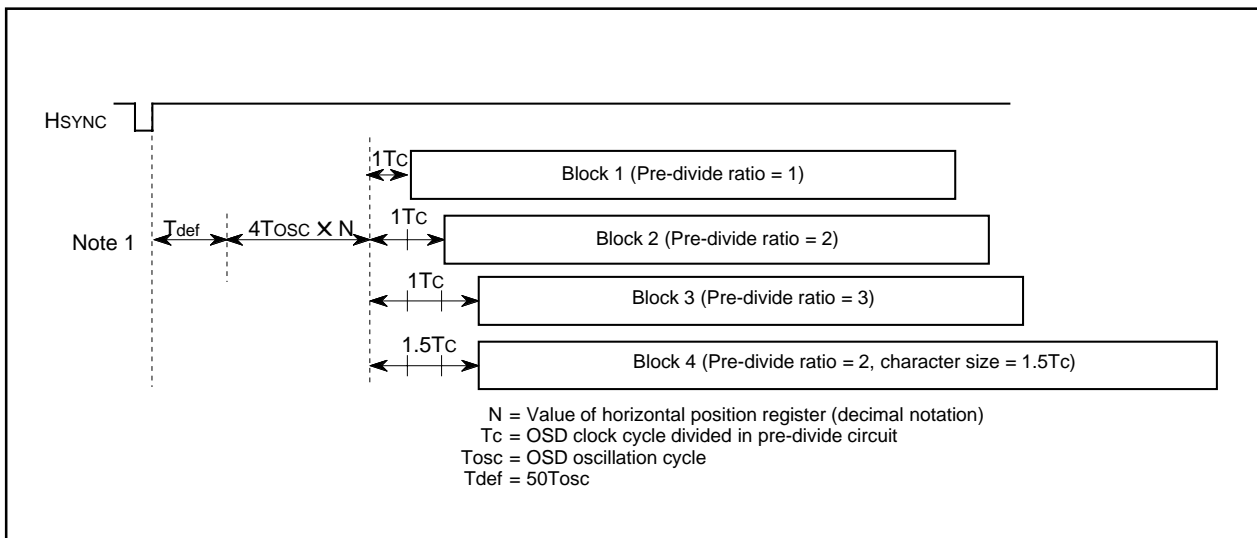


Figure 2.16.12 Notes on horizontal display start position

2.16.3 Dot Size

The dot size can be selected by a block unit. The dot size in vertical direction is determined by dividing HSYNC in the vertical dot size control circuit. The dot size in horizontal is determined by dividing the following clock in the horizontal dot size control circuit : the clock gained by dividing the OSD clock source (data slicer clock, OSC1, main clock) in the pre-divide circuit. The clock cycle divided in the pre-divide circuit is defined as 1Tc.

The dot size is specified by bits 3 to 6 of the block control register.

Refer to Figure 2.16.4 (the block control register i), refer to Figure 2.16.15 (the clock control register).

The block diagram of dot size control circuit is shown in Figure 2.16.13.

Notes 1 : The pre-divide ratio = 3 cannot be used in the CC mode.

2 : The pre-divide ratio of the layer 2 must be same as that of the layer 1 by the block control register i.

3 : In the bi-scan mode, the dot size in the vertical direction is 2 times as compared with the normal mode. Refer to “2.16.18 Scan Mode” about the scan mode.

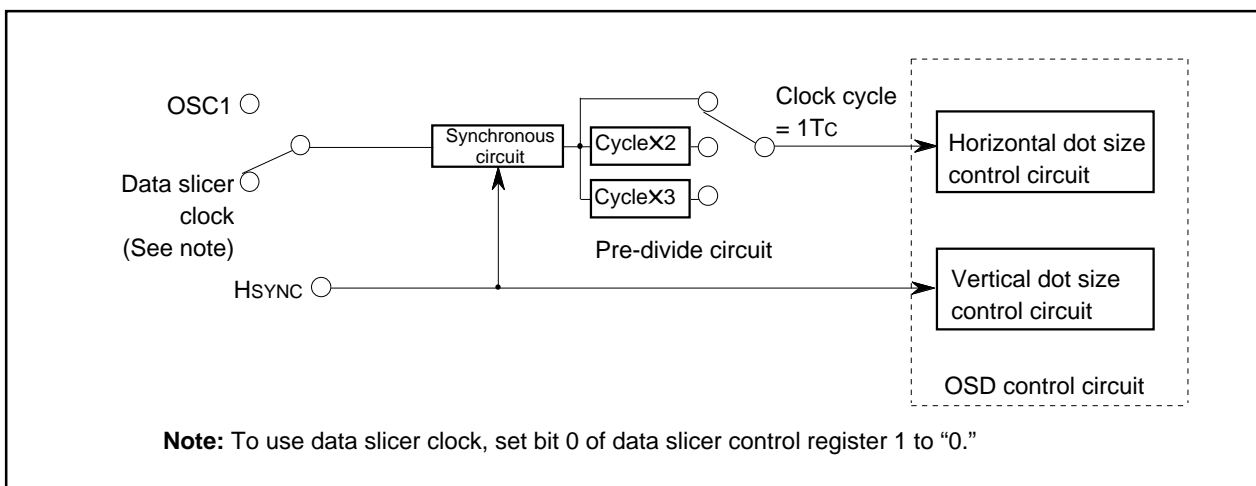


Figure 2.16.13 Block diagram of dot size control circuit

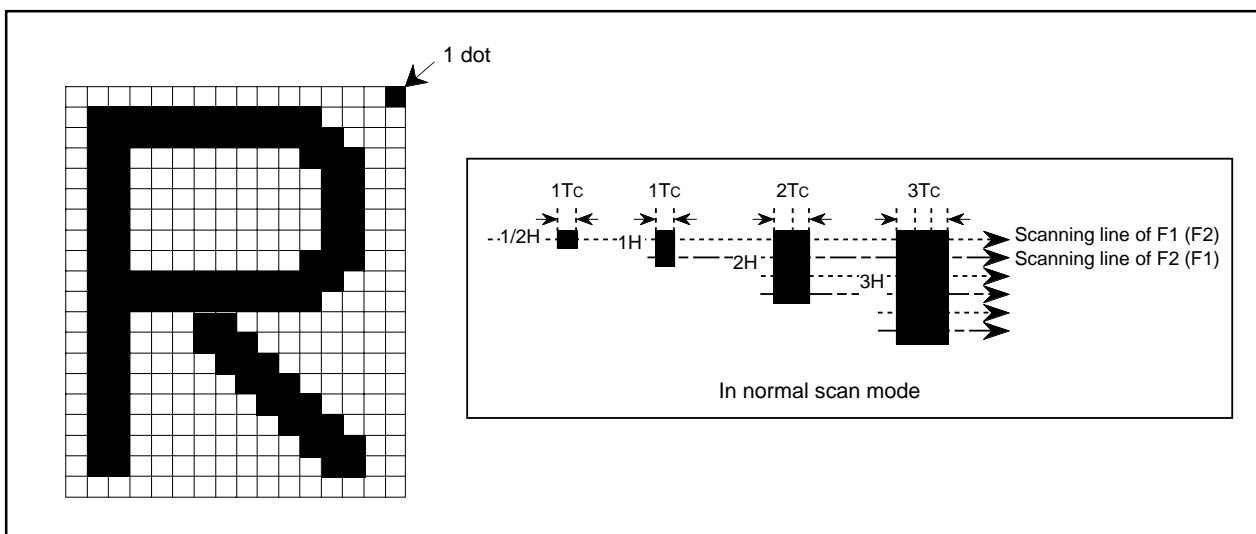


Figure 2.16.14 Definition of dot sizes

2.16.4 Clock for OSD

As a clock for display to be used for OSD, it is possible to select one of the following 3 types.

- Data slicer clock output from the data slicer (approximately 26 MHz)
- Clock from the LC oscillator supplied from the pins OSC1 and OSC2
- Clock from the ceramic resonator (or the quartz-crystal oscillator) from the pins OSC1 and OSC2

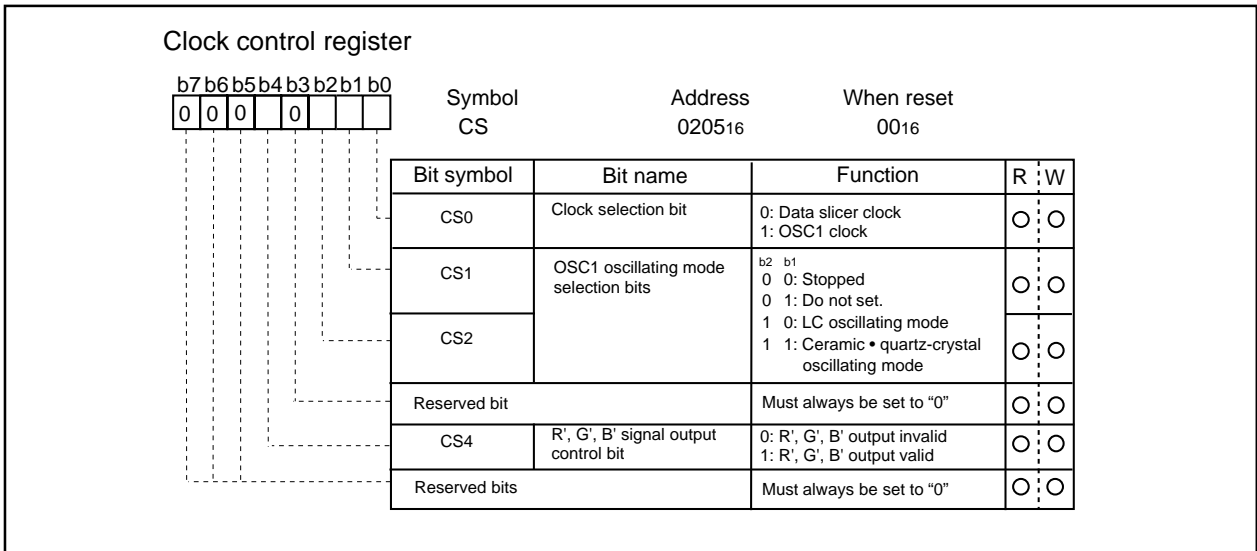


Figure 2.16.15 Clock control register

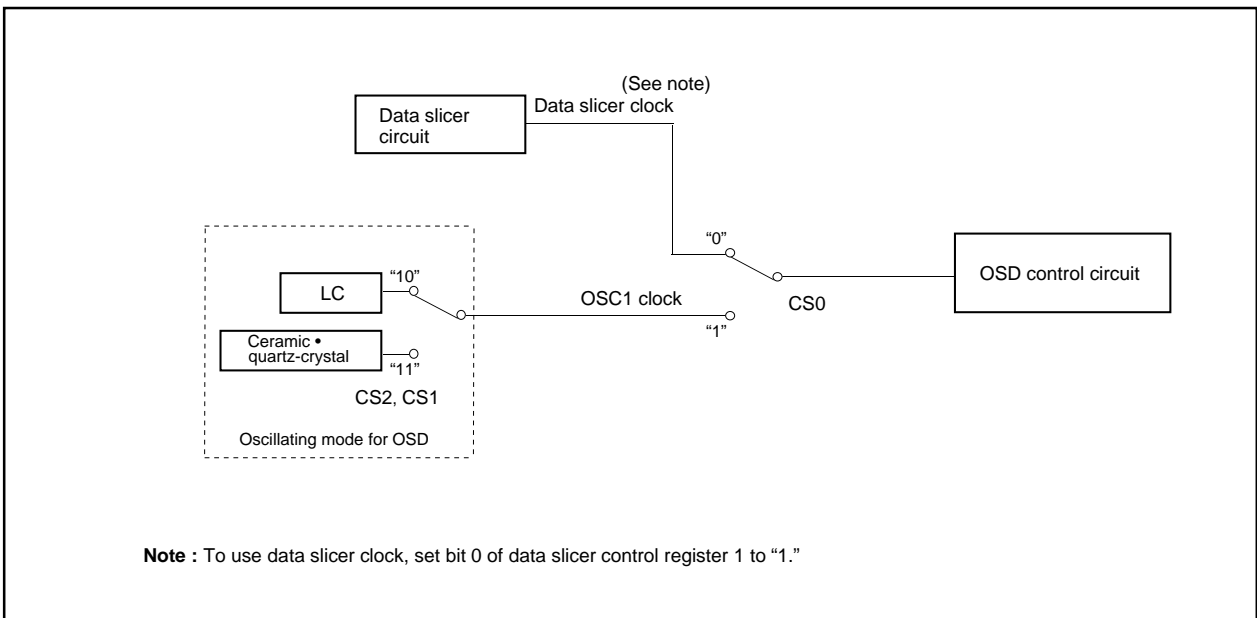


Figure 2.16.16 Block Diagram of OSD selection circuit

2.16.5 Field Determination Display

To display the block with vertical dot size of 1/2H, whether an even field or an odd field is determined through differences in a synchronizing signal waveform of interlacing system. The dot line 0 or 1 (refer to Figure 2.16.18) corresponding to the field is displayed alternately.

In the following, the field determination standard for the case where both the horizontal sync signal and the vertical sync signal are negative-polarity inputs will be explained. A field determination is determined by detecting the time from a falling edge of the horizontal sync signal until a falling edge of the VSYNC control signal (refer to Figure 2.16.9) in the microcomputer and then comparing this time with the time of the previous field. When the time is longer than the comparing time, it is regarded as even field. When the time is shorter, it is regarded as odd field.

The field determination flag changes at a rising edge of VSYNC control signal in the microcomputer .

The contents of this field can be read out by the field determination flag (bit 7 of the I/O polarity control register at address 0206₁₆). A dot line is specified by bit 6 of the I/O polarity control register (refer to Figure 2.16.18).

However, the field determination flag read out from the CPU is fixed to "0" at even field or "1" at odd field, regardless of bit 6.

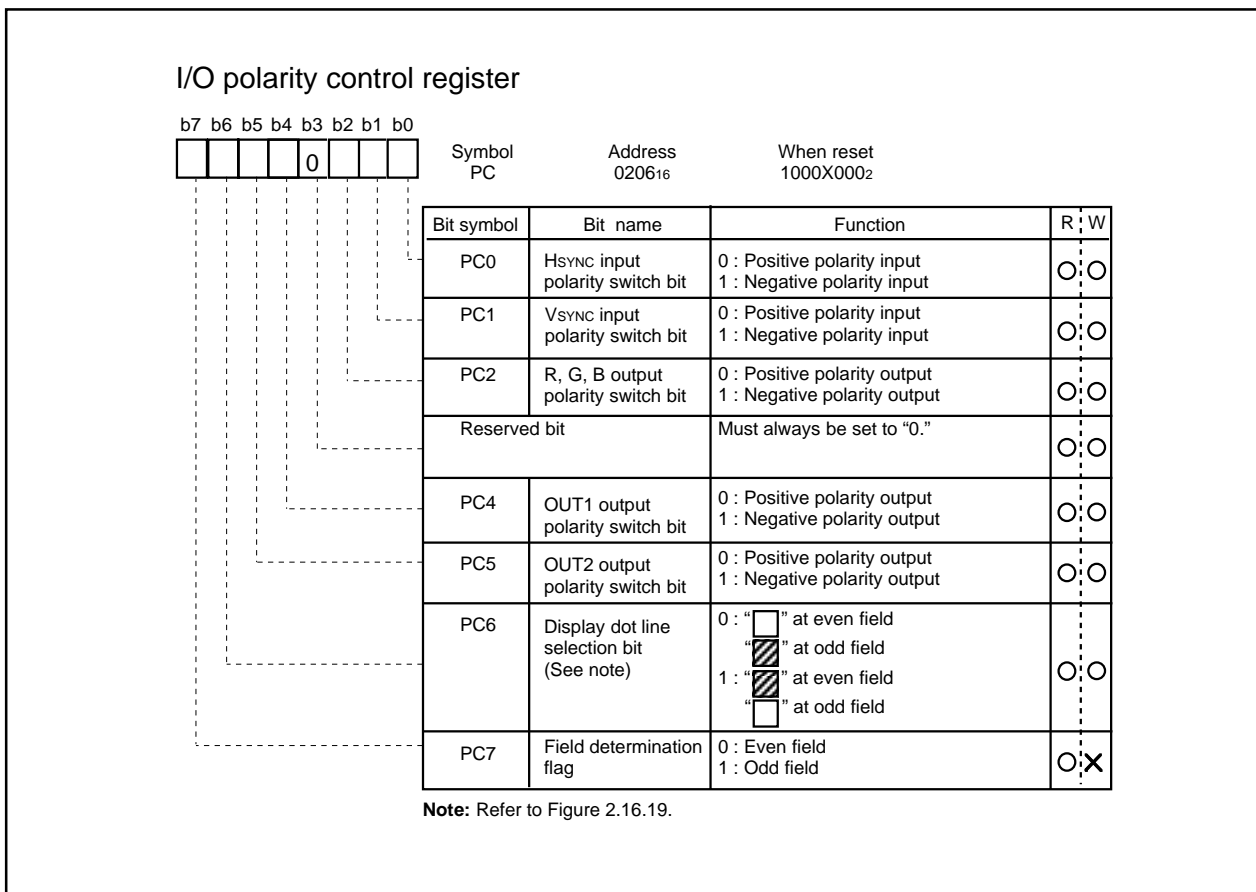
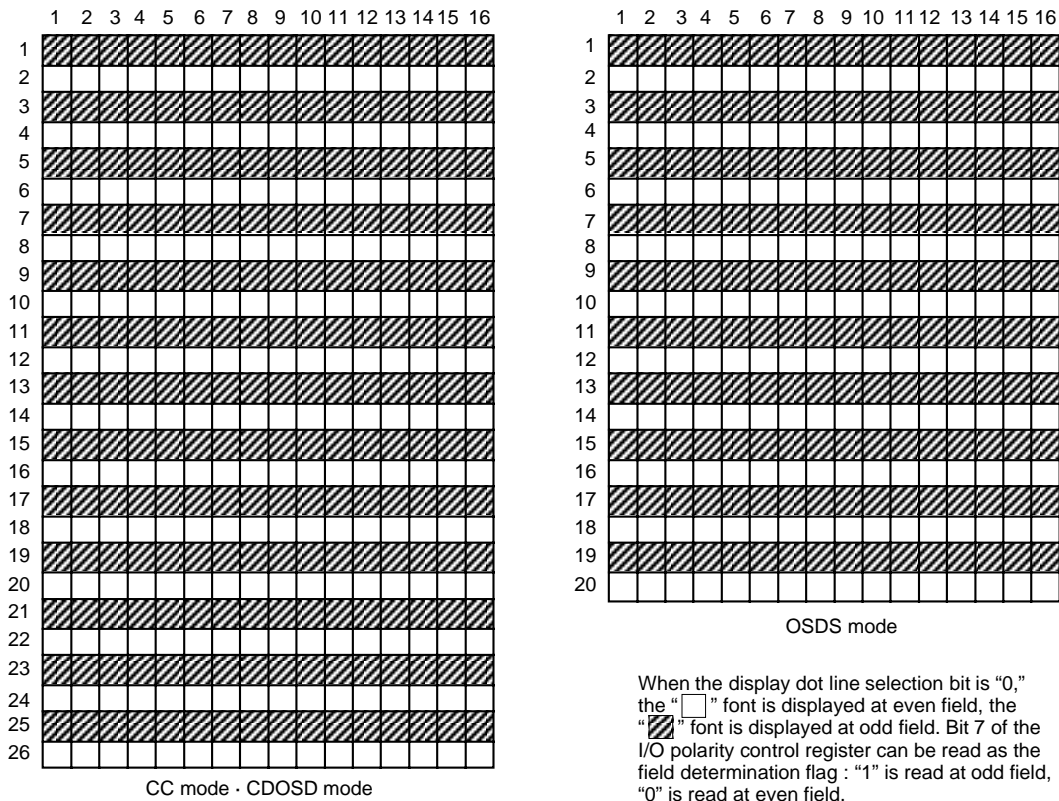


Figure 2.16.17 I/O polarity control register

Both Hsync signal and Vsync signal are negative-polarity input

| Hsync | | Field | Field determination flag(Note) | Display dot line selection bit | Display dot line |
|---|--|-------|--------------------------------|--------------------------------|--|
| | | | | | |
| Vsync and Vsync control signal in microcomputer | (n - 1) field (Odd-numbered) | Odd | / | / | / |
| | Upper : Vsync signal | Even | 0 (T2 > T1) | 0 | Dot line 1 <input type="checkbox"/> |
| | Lower : Vsync control signal in micro-computer | Odd | 1 (T3 < T2) | 1 | Dot line 0 <input checked="" type="checkbox"/> |

When using the field determination flag, set bit 7 of the peripheral mode register (address 027D16) according to the main clock frequency.



OSD ROM font configuration diagram

Note : The field determination flag changes at a rising edge of the Vsync control signal (negative-polarity input) in the microcomputer.

Figure 2.16.18 Relation between field determination flag and display font

2.16.6 Memory for OSD

There are 2 types of memory for OSD : OSD ROM (addresses 90000₁₆ to AFFFF₁₆) used to store character dot data and OSD RAM (addresses 0400₁₆ to 13FF₁₆) used to specify the kinds of display characters, display colors, and SPRITE display. The following describes each type of memory.

(1) ROM for OSD (addresses 90000₁₆ to AFFFF₁₆)

The dot pattern data for OSD characters is stored in the character font area in the OSD ROM and the CD font data for OSD characters is stored in the color dot font area in the OSD ROM. To specify the kinds of the character font and the CD font, it is necessary to write the character code into the OSD RAM.

For character font, there are the following 2 mode.

- OSDL enable mode
16 X 20-dot font and 24 X 32-dot font
- OSDL disable mode
16 X 20-dot font

The modes are selected by bit 3 of the OSD control register 3 for each screen.

The character font data storing address for OSDL enable/OSDL disable mode are shown in Figures 2.16.20 and 2.16.21. The conditions for each OSDL enable/disable mode are shown in Figure 2.16.22. The CD font data storing address is shown in Figure 2.16.23.

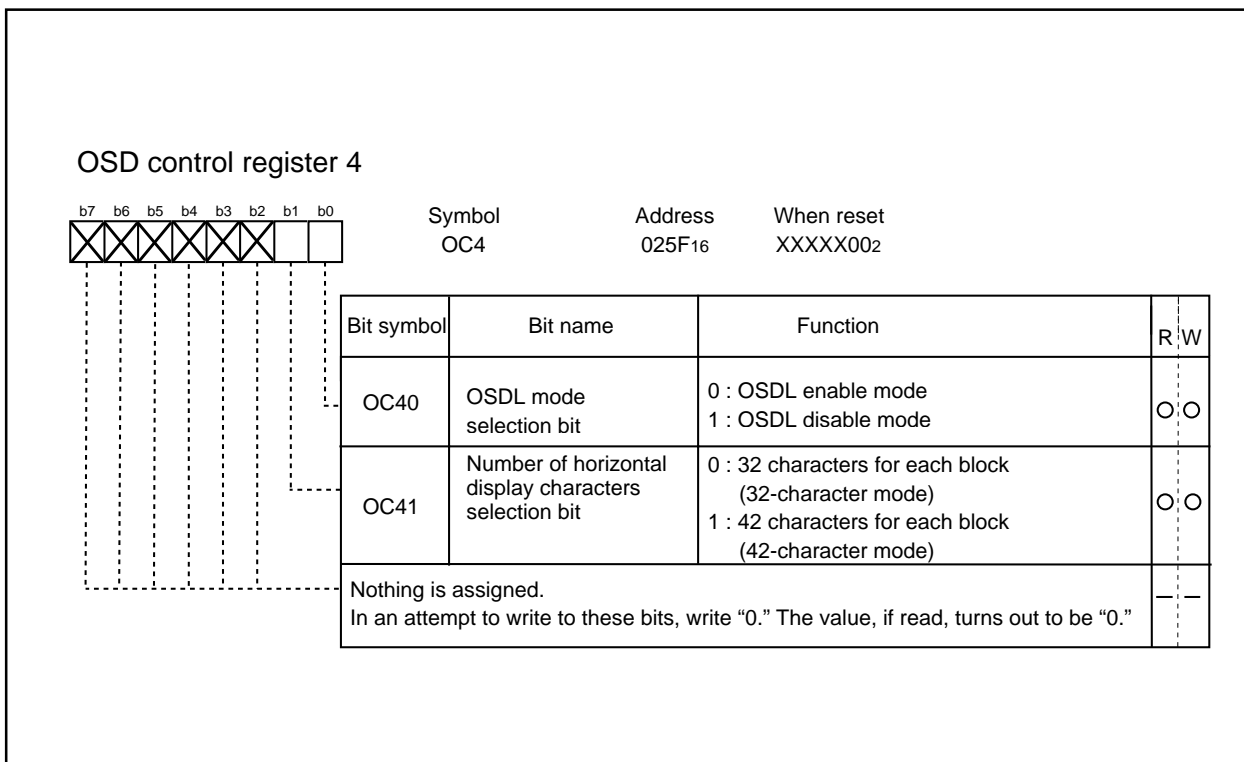
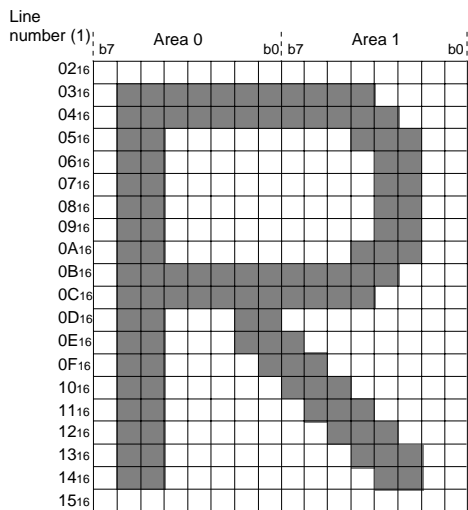


Figure 2.16.19 OSD control register 4

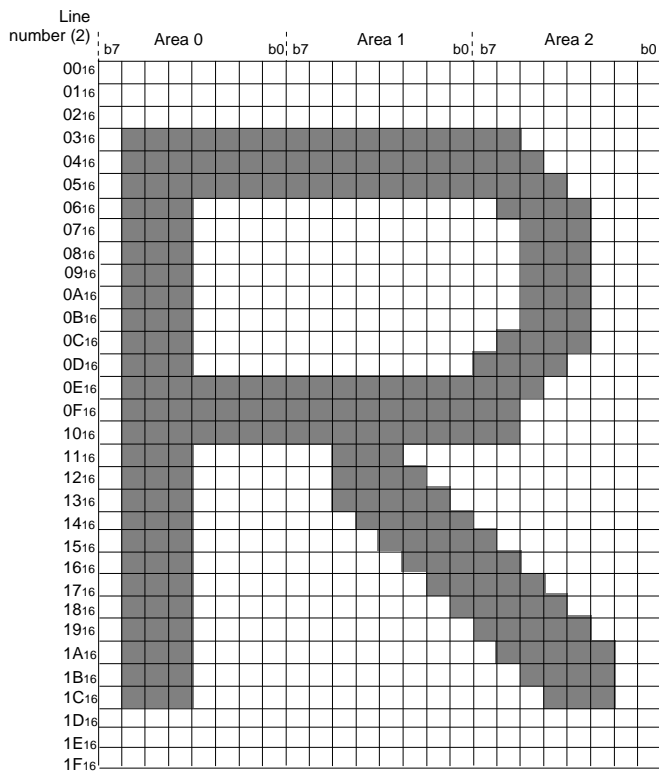
OSD ROM address of character font data (OSDL enable mode)

| OSD ROM address bit | | AD16 | AD15 | AD14 | AD13 | AD12 | AD11 | AD10 | AD9 | AD8 | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 |
|---|------------|------------------------------|------------------------------|------|------|------|------|-----------------------|---------------------------|---------------------------|-----|-----|-----|-----|-----|-----|----------|---------------------|
| Kinds of font | | Structure of address pointer | | | | | | | | | | | | | | | | |
| Font (1) Character codes 000 ₁₆ to 0FF ₁₆ | | 0 | Line number (1) (MSB to LSB) | | | | | Character code (C8)=0 | Character code (C7 to C0) | | | | | | | 0 | Area bit | |
| Font (2) Character codes 100 ₁₆ to 1FF ₁₆ | Areas 0, 1 | 0 | Line number (2) (MSB to LSB) | | | | | Character code (C8)=1 | Character code (C7 to C0) | | | | | | | 0 | Area bit | |
| | Area 2 | 1 | Line number (2) (MSB to LSB) | | | | | 0 | 0 | Character code (C6 to C0) | | | | | | | 0 | Character code (C7) |

Line number (1) = "02₁₆" to "15₁₆"
 Line number (2) = "00₁₆" to "1F₁₆"
 Character code = "000₁₆" to "1FF₁₆" ("0FE₁₆," "0FF₁₆," "100₁₆" and "180₁₆" cannot be used. Write "FF₁₆" to corresponding addresses.)
 Area bit = 0: Area 0
 1: Area 1



Font (1)
(Character codes 000₁₆ to 0FF₁₆)



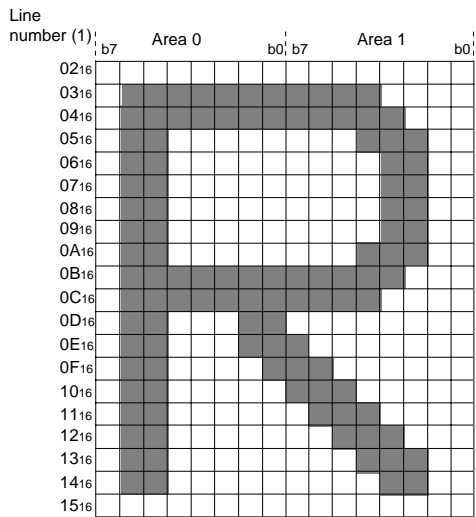
Font (2)
(Character codes 100₁₆ to 1FF₁₆)

Figure 2.16.20 Character font data storing address (OSDL enable mode)

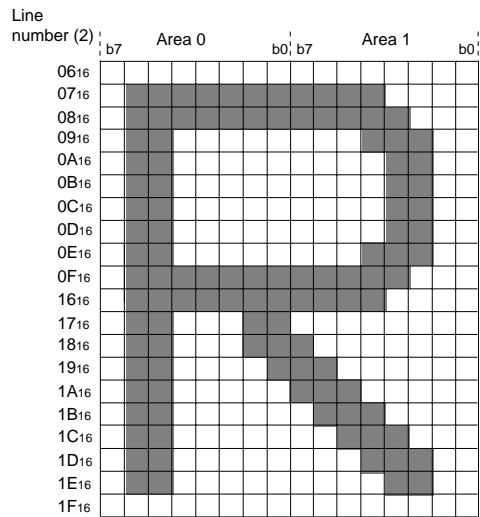
OSD ROM address of character font data (OSDL disable mode)

| OSD ROM address bit | AD16 | AD15 | AD14 | AD13 | AD12 | AD11 | AD10 | AD9 | AD8 | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 |
|---|-----------------------|------------------------------|------------------------------|------|------|------|---------------------------|-----------------------|---------------------------|-----|-----|-----|-----|-----|----------|-----|----------|
| | Kinds of font | Structure of address pointer | | | | | | | | | | | | | | | |
| Font (1) Character codes 000 ₁₆ to 1FF ₁₆ | Character code (C9)=0 | | Line number (1) (MSB to LSB) | | | | Character code (C8 to C0) | | | | | | 0 | | Area bit | | |
| Font (2) Character codes 200 ₁₆ to 27F ₁₆ | Character code (C9)=1 | | Line number (1) (MSB to LSB) | | | | Character code (C8 to C0) | | | | | | 0 | | Area bit | | |
| Font (3) Character codes 280 ₁₆ to 2FF ₁₆ | 0 | 1 | Line number (3) (NL3 to NL0) | | | | 1 | Line number (3) (NL4) | Character code (C6 to C0) | | | | | | 0 | | Area bit |

Line number (1) = "02₁₆" to "15₁₆"
 Line number (3) = "06₁₆" to "0F₁₆" and "16₁₆" to "1F₁₆"
 Character code = "000₁₆" to "2FF₁₆" ("0FE₁₆," "0FF₁₆," "100₁₆," "180₁₆," "200₁₆" and "280₁₆" cannot be used. Write "FF₁₆" to corresponding addresses.)
 Area bit = 0: Area 0
 1: Area 1



Font (1)
Font (2)
(Character codes 000₁₆ to 27F₁₆)

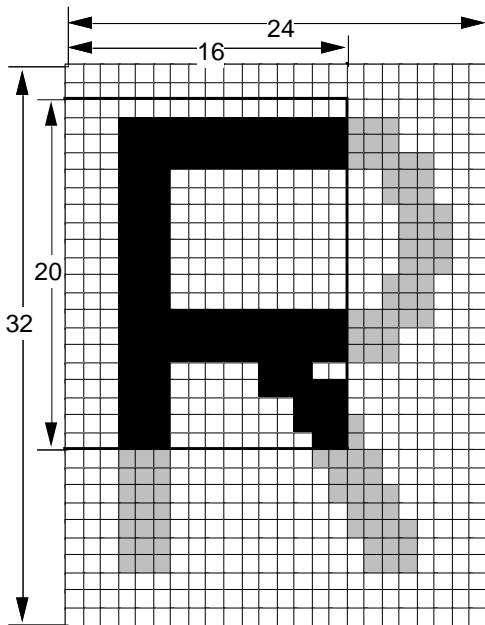


Font (3)
(Character codes 280₁₆ to 2FF₁₆)

Figure 2.16.21 Character font data storing address (OSDL disable mode)

Depending on the relationship of OSDL enable/disable mode, display mode and character code, note the conditions below.

| OSDL enable/ disable mode Display mode & character code | | OSDL enable mode (Bit 0 of OSD control register 4 = "0") | | | | OSDL disable mode (Bit 0 of OSD control register 4 = "1") | | | |
|--|--|---|--------------------------|----------------------|--------------------------|--|--------------------------------------|-------------|-------------|
| | | Character size | CC | OSDS/P | OSDL | Character size | CC | OSDS/P | OSDL |
| Specified character code | 000 ₁₆ to 0FF ₁₆ | S | Used | Used | Not used (See note 3) | S | Used | Used | Display OFF |
| | 100 ₁₆ to 1FF ₁₆ | L | Used (See note 1) | Used (See note 1) | Used | S | Used | Used | Display OFF |
| | 200 ₁₆ to 27F ₁₆ | | Not used (See note 3) | | | S | | Used | Display OFF |
| | 280 ₁₆ to 2FF ₁₆ | | | | | Not used (See note 3) | Used (No border) (See note 2) | Display OFF | |
| | 300 ₁₆ to 3FF ₁₆ | | | | | Not used | Display OFF | | |



- Notes 1:** Part of 24 X 32 font is displayed.
- 2:** In OSDL disable mode, character codes "280₁₆" to "2FF₁₆" are used in OSDS/P mode (no border).
- 3:** As setting this make output of font data indeterminate, do not use. However, "3FE₁₆" and "3FF₁₆" can be used as character codes of blank font output in OSDP mode.

Figure 2.16.22 Conditions for each OSDL enable/disable mode

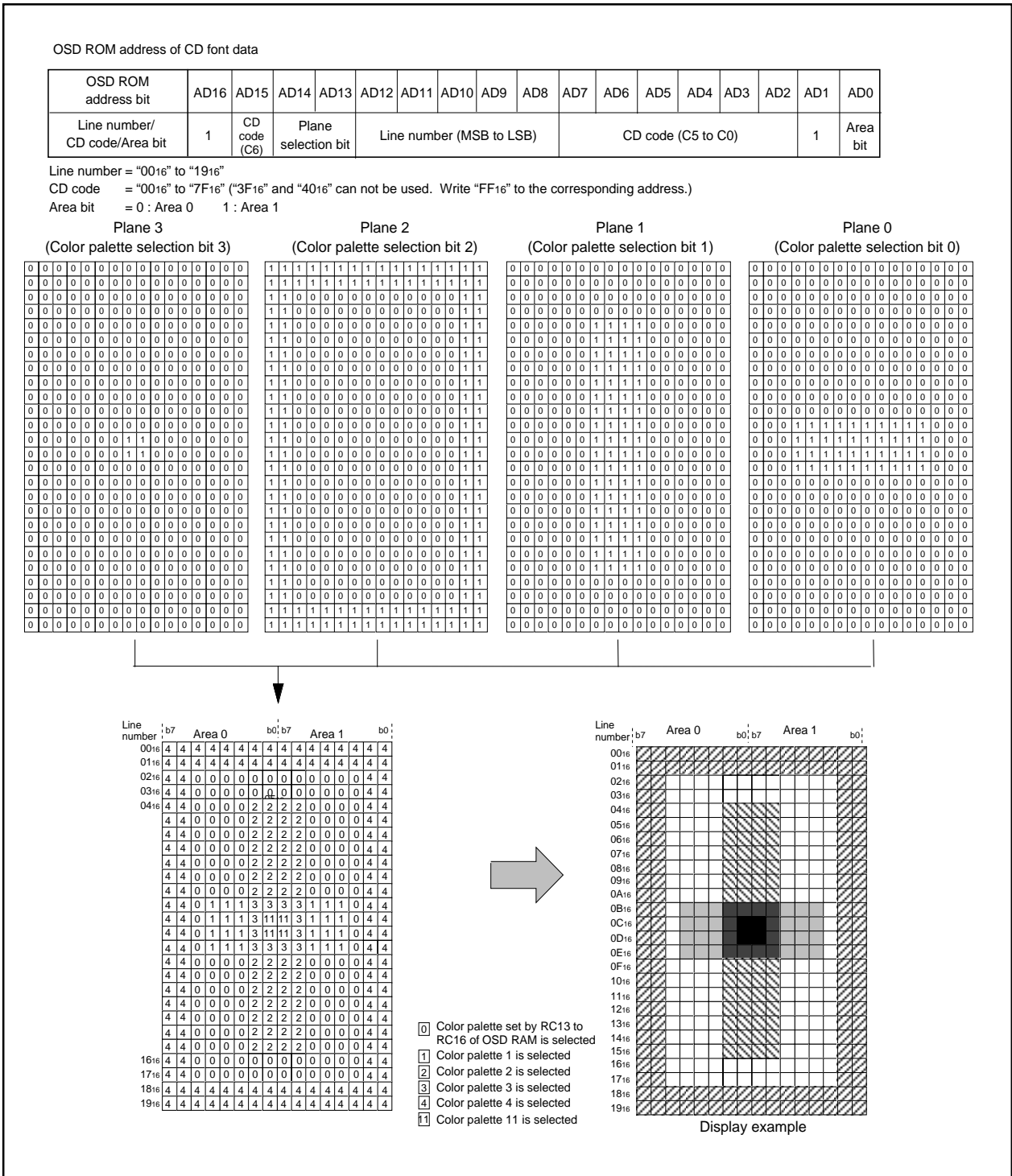


Figure 2.16.23 Color dot font data storing address

(2) OSD RAM (OSD RAM for character, addresses 0400₁₆ to 0EFF₁₆)

The OSD RAM for character is allocated at addresses 0400₁₆ to 0EFF₁₆, and is divided into a display character code specification part, color code 1 specification part, and color code 2 specification part for each block. The number of characters for 1 block (32- or 42-character mode) is selected by bit 1 of the OSD control register 4. Tables 2.16.3 to 2.16.7 show the address map.

For example, to display 1 character position (the left edge) in block 1, write the character code in address 0400₁₆, write color code 1 at 0401₁₆, and write color code 2 at 0480₁₆. The structure of the OSD RAM is shown in Figure 2.16.25.

Note : For blocks of the following dot sizes, the 3rd (n = 1 to 14) character is skipped as compared with ordinary block.

■ In OSDL mode: all dot size.

■ In OSDS and CDOSD modes of layer 2: 1.5Tc X 1/2H or 1.5Tc X 1H

Accordingly, maximum 22 characters (32-character mode)/28 characters (42-character mode) are only displayed in 1 block. Blocks with dot size of 1Tc X 1/2H and 1Tc X 1H, or blocks on the layer 1. The RAM data for the 3rd character does not effect the display. Any character data can be stored here. And also, note the following only in 32-character mode. As the character is displayed in the 28th's character area in 42-character mode, set ordinarily.

- In OSDS mode
The character is not displayed, and only the left 1/3 part of the 22nd character background is displayed in the 22nd's character area. When not displaying this background, set transparent for character background color.
- In OSDL mode
Set a blank character or a character of transparent color to the 22nd character.
- In CDOSD mode
The character is not displayed, and color palette color specified by bits 3 to 6 of color code 1 can be output in the 22nd's character area (left 1/3 part).

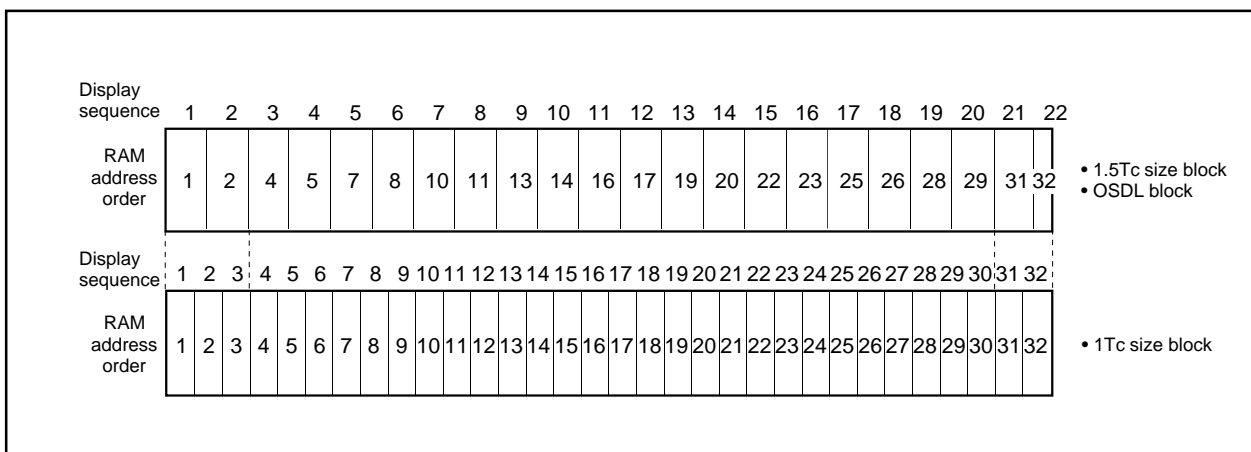


Figure 2.16.24 RAM data for 3rd character (in 32-character mode)

Table 2.16.3 Contents of OSD RAM (1st to 32nd character)

| Block | Display Position (from left) | Character Code Specification | Color Code 1 Specification | Color Code 2 Specification |
|----------|------------------------------|------------------------------|----------------------------|----------------------------|
| Block 1 | 1st character | 0400 ₁₆ | 0401 ₁₆ | 0480 ₁₆ |
| | 2nd character | 0402 ₁₆ | 0403 ₁₆ | 0482 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| | 32nd character | 043E ₁₆ | 043F ₁₆ | 04BE ₁₆ |
| Block 2 | 1st character | 0440 ₁₆ | 0441 ₁₆ | 04C0 ₁₆ |
| | 2nd character | 0442 ₁₆ | 0443 ₁₆ | 04C2 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| | 32nd character | 047E ₁₆ | 047F ₁₆ | 04FE ₁₆ |
| Block 3 | 1st character | 0500 ₁₆ | 0501 ₁₆ | 0580 ₁₆ |
| | 2nd character | 0502 ₁₆ | 0503 ₁₆ | 0582 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| | 32nd character | 053E ₁₆ | 053F ₁₆ | 05BE ₁₆ |
| Block 4 | 1st character | 0540 ₁₆ | 0541 ₁₆ | 05C0 ₁₆ |
| | 2nd character | 0542 ₁₆ | 0543 ₁₆ | 05C2 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| | 32nd character | 057E ₁₆ | 057F ₁₆ | 05FE ₁₆ |
| Block 5 | 1st character | 0600 ₁₆ | 0601 ₁₆ | 0680 ₁₆ |
| | 2nd character | 0602 ₁₆ | 0603 ₁₆ | 0682 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| | 32nd character | 063E ₁₆ | 063F ₁₆ | 06BE ₁₆ |
| Block 6 | 1st character | 0640 ₁₆ | 0641 ₁₆ | 06C0 ₁₆ |
| | 2nd character | 0642 ₁₆ | 0643 ₁₆ | 06C2 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| | 32nd character | 067E ₁₆ | 067F ₁₆ | 06FE ₁₆ |
| Block 7 | 1st character | 0700 ₁₆ | 0701 ₁₆ | 0780 ₁₆ |
| | 2nd character | 0702 ₁₆ | 0703 ₁₆ | 0782 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| | 32nd character | 073E ₁₆ | 073F ₁₆ | 07BE ₁₆ |
| Block 8 | 1st character | 0740 ₁₆ | 0741 ₁₆ | 07C0 ₁₆ |
| | 2nd character | 0742 ₁₆ | 0743 ₁₆ | 07C2 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| | 32nd character | 077E ₁₆ | 077F ₁₆ | 07FE ₁₆ |
| Block 9 | 1st character | 0800 ₁₆ | 0801 ₁₆ | 0880 ₁₆ |
| | 2nd character | 0802 ₁₆ | 0803 ₁₆ | 0882 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| | 32nd character | 083E ₁₆ | 083F ₁₆ | 08BE ₁₆ |
| Block 10 | 1st character | 0840 ₁₆ | 0841 ₁₆ | 08C0 ₁₆ |
| | 2nd character | 0842 ₁₆ | 0843 ₁₆ | 08C2 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| | 32nd character | 087E ₁₆ | 087F ₁₆ | 08FE ₁₆ |

Table 2.16.4 Contents of OSD RAM (1st to 32nd character) (continued)

| Block | Display Position (from left) | Character Code Specification | Color Code 1 Specification | Color Code 2 Specification |
|----------|------------------------------|------------------------------|----------------------------|----------------------------|
| Block 11 | 1st character | 0900 ₁₆ | 0901 ₁₆ | 0980 ₁₆ |
| | 2nd character | 0902 ₁₆ | 0903 ₁₆ | 0982 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| | 31st character | 093C ₁₆ | 093D ₁₆ | 09BC ₁₆ |
| | 32nd character | 093E ₁₆ | 093F ₁₆ | 09BE ₁₆ |
| Block 12 | 1st character | 0940 ₁₆ | 0941 ₁₆ | 09C0 ₁₆ |
| | 2nd character | 0942 ₁₆ | 0943 ₁₆ | 09C2 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| | 31st character | 097C ₁₆ | 097D ₁₆ | 09FC ₁₆ |
| | 32nd character | 097E ₁₆ | 097F ₁₆ | 09FE ₁₆ |
| Block 13 | 1st character | 0A00 ₁₆ | 0A01 ₁₆ | 0A80 ₁₆ |
| | 2nd character | 0A02 ₁₆ | 0A03 ₁₆ | 0A82 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| | 31st character | 0A3C ₁₆ | 0A3D ₁₆ | 0ABC ₁₆ |
| | 32nd character | 0A3E ₁₆ | 0A3F ₁₆ | 0ABE ₁₆ |
| Block 14 | 1st character | 0A40 ₁₆ | 0A41 ₁₆ | 0AC0 ₁₆ |
| | 2nd character | 0A42 ₁₆ | 0A43 ₁₆ | 0AC2 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| | 31st character | 0A7C ₁₆ | 0A7D ₁₆ | 0AFC ₁₆ |
| | 32nd character | 0A7E ₁₆ | 0A7F ₁₆ | 0AFE ₁₆ |
| Block 15 | 1st character | 0B00 ₁₆ | 0B01 ₁₆ | 0B80 ₁₆ |
| | 2nd character | 0B02 ₁₆ | 0B03 ₁₆ | 0B82 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| | 31st character | 0B3C ₁₆ | 0B3D ₁₆ | 0BBC ₁₆ |
| | 32nd character | 0B3E ₁₆ | 0B3F ₁₆ | 0BBE ₁₆ |
| Block 16 | 1st character | 0B40 ₁₆ | 0B41 ₁₆ | 0BC0 ₁₆ |
| | 2nd character | 0B42 ₁₆ | 0B43 ₁₆ | 0BC2 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| | 31st character | 0B7C ₁₆ | 0B7D ₁₆ | 0BF0 ₁₆ |
| | 32nd character | 0B7E ₁₆ | 0B7F ₁₆ | 0BFE ₁₆ |

Table 2.16.5 Contents of OSD RAM (33rd to 42nd character)

| Block | Display Position (from left) | Character Code Specification | Color Code 1 Specification | Color Code 2 Specification |
|---------|------------------------------|------------------------------|----------------------------|----------------------------|
| Block 1 | 33rd character | 0C00 ₁₆ | 0C01 ₁₆ | 0C80 ₁₆ |
| | 34th character | 0C02 ₁₆ | 0C03 ₁₆ | 0C82 ₁₆ |
| | : | : | : | : |
| | 39th character | 0C0C ₁₆ | 0C0D ₁₆ | 0C8C ₁₆ |
| | 40th character | 0C0E ₁₆ | 0C0F ₁₆ | 0C8E ₁₆ |
| | 41st character | 0E00 ₁₆ | 0E01 ₁₆ | 0E80 ₁₆ |
| Block 2 | 42nd character | 0E02 ₁₆ | 0E03 ₁₆ | 0E82 ₁₆ |
| | 33rd character | 0C10 ₁₆ | 0C11 ₁₆ | 0C90 ₁₆ |
| | 34th character | 0C12 ₁₆ | 0C13 ₁₆ | 0C92 ₁₆ |
| | : | : | : | : |
| | 39th character | 0C1C ₁₆ | 0C1D ₁₆ | 0C9C ₁₆ |
| | 40th character | 0C1E ₁₆ | 0C1F ₁₆ | 0C9E ₁₆ |
| Block 3 | 41st character | 0E08 ₁₆ | 0E09 ₁₆ | 0E88 ₁₆ |
| | 42nd character | 0E0A ₁₆ | 0E0B ₁₆ | 0E8A ₁₆ |
| | 33rd character | 0C20 ₁₆ | 0C21 ₁₆ | 0CA0 ₁₆ |
| | 34th character | 0C22 ₁₆ | 0C23 ₁₆ | 0CA2 ₁₆ |
| | : | : | : | : |
| | 39th character | 0C2C ₁₆ | 0C2D ₁₆ | 0CAC ₁₆ |
| Block 4 | 40th character | 0C2E ₁₆ | 0C2F ₁₆ | 0CAE ₁₆ |
| | 41st character | 0E10 ₁₆ | 0E11 ₁₆ | 0E90 ₁₆ |
| | 42nd character | 0E12 ₁₆ | 0E13 ₁₆ | 0E92 ₁₆ |
| | 33rd character | 0C30 ₁₆ | 0C31 ₁₆ | 0CB0 ₁₆ |
| | 34th character | 0C32 ₁₆ | 0C33 ₁₆ | 0CB2 ₁₆ |
| | : | : | : | : |
| Block 5 | 39th character | 0C3C ₁₆ | 0C3D ₁₆ | 0CBC ₁₆ |
| | 40th character | 0C3E ₁₆ | 0C3F ₁₆ | 0CBE ₁₆ |
| | 41st character | 0E18 ₁₆ | 0E19 ₁₆ | 0E98 ₁₆ |
| | 42nd character | 0E1A ₁₆ | 0E1B ₁₆ | 0E9A ₁₆ |
| | 33rd character | 0C40 ₁₆ | 0C41 ₁₆ | 0CC0 ₁₆ |
| | 34th character | 0C42 ₁₆ | 0C43 ₁₆ | 0CC2 ₁₆ |
| Block 6 | : | : | : | : |
| | 39th character | 0C4C ₁₆ | 0C4D ₁₆ | 0CCC ₁₆ |
| | 40th character | 0C4E ₁₆ | 0C4F ₁₆ | 0CCE ₁₆ |
| | 41st character | 0E20 ₁₆ | 0E21 ₁₆ | 0EA0 ₁₆ |
| | 42nd character | 0E22 ₁₆ | 0E23 ₁₆ | 0EA2 ₁₆ |
| | 33rd character | 0C50 ₁₆ | 0C51 ₁₆ | 0CD0 ₁₆ |
| Block 7 | 34th character | 0C52 ₁₆ | 0C53 ₁₆ | 0CD2 ₁₆ |
| | : | : | : | : |
| | 39th character | 0C5C ₁₆ | 0C5D ₁₆ | 0CDC ₁₆ |
| | 40th character | 0C5E ₁₆ | 0C5F ₁₆ | 0CDE ₁₆ |
| | 41st character | 0E28 ₁₆ | 0E29 ₁₆ | 0EA8 ₁₆ |
| | 42nd character | 0E2A ₁₆ | 0E2B ₁₆ | 0EAA ₁₆ |
| Block 8 | 33rd character | 0C60 ₁₆ | 0C61 ₁₆ | 0CE0 ₁₆ |
| | 34th character | 0C62 ₁₆ | 0C63 ₁₆ | 0CE2 ₁₆ |
| | : | : | : | : |
| | 39th character | 0C6C ₁₆ | 0C6D ₁₆ | 0CEC ₁₆ |
| | 40th character | 0C6E ₁₆ | 0C6F ₁₆ | 0CEE ₁₆ |
| | 41st character | 0E30 ₁₆ | 0E31 ₁₆ | 0EB0 ₁₆ |
| Block 9 | 42nd character | 0E32 ₁₆ | 0E33 ₁₆ | 0EB2 ₁₆ |

Table 2.16.6 Contents of OSD RAM (33rd to 42nd character) (continued)

| Block | Display Position (from left) | Character Code Specification | Color Code 1 Specification | Color Code 2 Specification |
|----------------|------------------------------|------------------------------|----------------------------|----------------------------|
| Block 8 | 33rd character | 0C70 ₁₆ | 0C71 ₁₆ | 0CF0 ₁₆ |
| | 34th character | 0C72 ₁₆ | 0C73 ₁₆ | 0CF2 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| | 39th character | 0C7C ₁₆ | 0C7D ₁₆ | 0CFC ₁₆ |
| | 40th character | 0C7E ₁₆ | 0C7F ₁₆ | 0CFE ₁₆ |
| | 41st character | 0E38 ₁₆ | 0E39 ₁₆ | 0EB8 ₁₆ |
| Block 9 | 42nd character | 0E3A ₁₆ | 0E3B ₁₆ | 0EBA ₁₆ |
| | 33rd character | 0D00 ₁₆ | 0D01 ₁₆ | 0D80 ₁₆ |
| | 34th character | 0D02 ₁₆ | 0D03 ₁₆ | 0D82 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| | 39th character | 0D0C ₁₆ | 0D0D ₁₆ | 0D8C ₁₆ |
| | 40th character | 0D0E ₁₆ | 0D0F ₁₆ | 0D8E ₁₆ |
| Block 10 | 41st character | 0E40 ₁₆ | 0E41 ₁₆ | 0EC0 ₁₆ |
| | 42nd character | 0E42 ₁₆ | 0E43 ₁₆ | 0EC2 ₁₆ |
| | 33rd character | 0D10 ₁₆ | 0D11 ₁₆ | 0D90 ₁₆ |
| | 34th character | 0D12 ₁₆ | 0D13 ₁₆ | 0D92 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| | 39th character | 0D1C ₁₆ | 0D1D ₁₆ | 0D9C ₁₆ |
| Block 11 | 40th character | 0D1E ₁₆ | 0D1F ₁₆ | 0D9E ₁₆ |
| | 41st character | 0E48 ₁₆ | 0E49 ₁₆ | 0EC8 ₁₆ |
| | 42nd character | 0E4A ₁₆ | 0E4B ₁₆ | 0ECA ₁₆ |
| | 33rd character | 0D20 ₁₆ | 0D21 ₁₆ | 0DA0 ₁₆ |
| | 34th character | 0D22 ₁₆ | 0D23 ₁₆ | 0DA2 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| Block 12 | 39th character | 0D2C ₁₆ | 0D2D ₁₆ | 0DAC ₁₆ |
| | 40th character | 0D2E ₁₆ | 0D2F ₁₆ | 0DAE ₁₆ |
| | 41st character | 0E50 ₁₆ | 0E51 ₁₆ | 0ED0 ₁₆ |
| | 42nd character | 0E52 ₁₆ | 0E53 ₁₆ | 0ED2 ₁₆ |
| | 33rd character | 0D30 ₁₆ | 0D31 ₁₆ | 0DB0 ₁₆ |
| | 34th character | 0D32 ₁₆ | 0D33 ₁₆ | 0DB2 ₁₆ |
| Block 13 | ⋮ | ⋮ | ⋮ | ⋮ |
| | 39th character | 0D3C ₁₆ | 0D3D ₁₆ | 0DBC ₁₆ |
| | 40th character | 0D3E ₁₆ | 0D3F ₁₆ | 0DBE ₁₆ |
| | 41st character | 0E58 ₁₆ | 0E59 ₁₆ | 0ED8 ₁₆ |
| | 42nd character | 0E5A ₁₆ | 0E5B ₁₆ | 0EDA ₁₆ |
| | 33rd character | 0D40 ₁₆ | 0D41 ₁₆ | 0DC0 ₁₆ |
| Block 14 | 34th character | 0D42 ₁₆ | 0D43 ₁₆ | 0DC2 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| | 39th character | 0D4C ₁₆ | 0D4D ₁₆ | 0DCC ₁₆ |
| | 40th character | 0D4E ₁₆ | 0D4F ₁₆ | 0DCE ₁₆ |
| | 41st character | 0E60 ₁₆ | 0E61 ₁₆ | 0EE0 ₁₆ |
| | 42nd character | 0E62 ₁₆ | 0E63 ₁₆ | 0EE2 ₁₆ |
| Block 14 | 33rd character | 0D50 ₁₆ | 0D51 ₁₆ | 0DD0 ₁₆ |
| | 34th character | 0D52 ₁₆ | 0D53 ₁₆ | 0DD2 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| | 39th character | 0D5C ₁₆ | 0D5D ₁₆ | 0DDC ₁₆ |
| | 40th character | 0D5E ₁₆ | 0D5F ₁₆ | 0DDE ₁₆ |
| | 41st character | 0E68 ₁₆ | 0E69 ₁₆ | 0EE8 ₁₆ |
| 42nd character | 0E6A ₁₆ | 0E6B ₁₆ | 0EEA ₁₆ | |

Table 2.16.7 Contents of OSD RAM (33rd to 42nd character) (continued)

| Block | Display Position (from left) | Character Code Specification | Color Code 1 Specification | Color Code 2 Specification |
|----------|------------------------------|------------------------------|----------------------------|----------------------------|
| Block 15 | 33rd character | 0D60 ₁₆ | 0D61 ₁₆ | 0DE0 ₁₆ |
| | 34th character | 0D62 ₁₆ | 0D63 ₁₆ | 0DE2 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| | 39th character | 0D6C ₁₆ | 0D6D ₁₆ | 0DEC ₁₆ |
| | 40th character | 0D6E ₁₆ | 0D6F ₁₆ | 0DEE ₁₆ |
| | 41st character | 0E70 ₁₆ | 0E71 ₁₆ | 0EF0 ₁₆ |
| Block 16 | 42nd character | 0E72 ₁₆ | 0E73 ₁₆ | 0EF2 ₁₆ |
| | 33rd character | 0D70 ₁₆ | 0D71 ₁₆ | 0DF0 ₁₆ |
| | 34th character | 0D72 ₁₆ | 0D73 ₁₆ | 0DF2 ₁₆ |
| | ⋮ | ⋮ | ⋮ | ⋮ |
| | 39th character | 0D7C ₁₆ | 0D7D ₁₆ | 0DFC ₁₆ |
| | 40th character | 0D7E ₁₆ | 0D7F ₁₆ | 0DFE ₁₆ |
| Block 16 | 41st character | 0E78 ₁₆ | 0E79 ₁₆ | 0EF8 ₁₆ |
| | 42nd character | 0E7A ₁₆ | 0E7B ₁₆ | 0EFA ₁₆ |

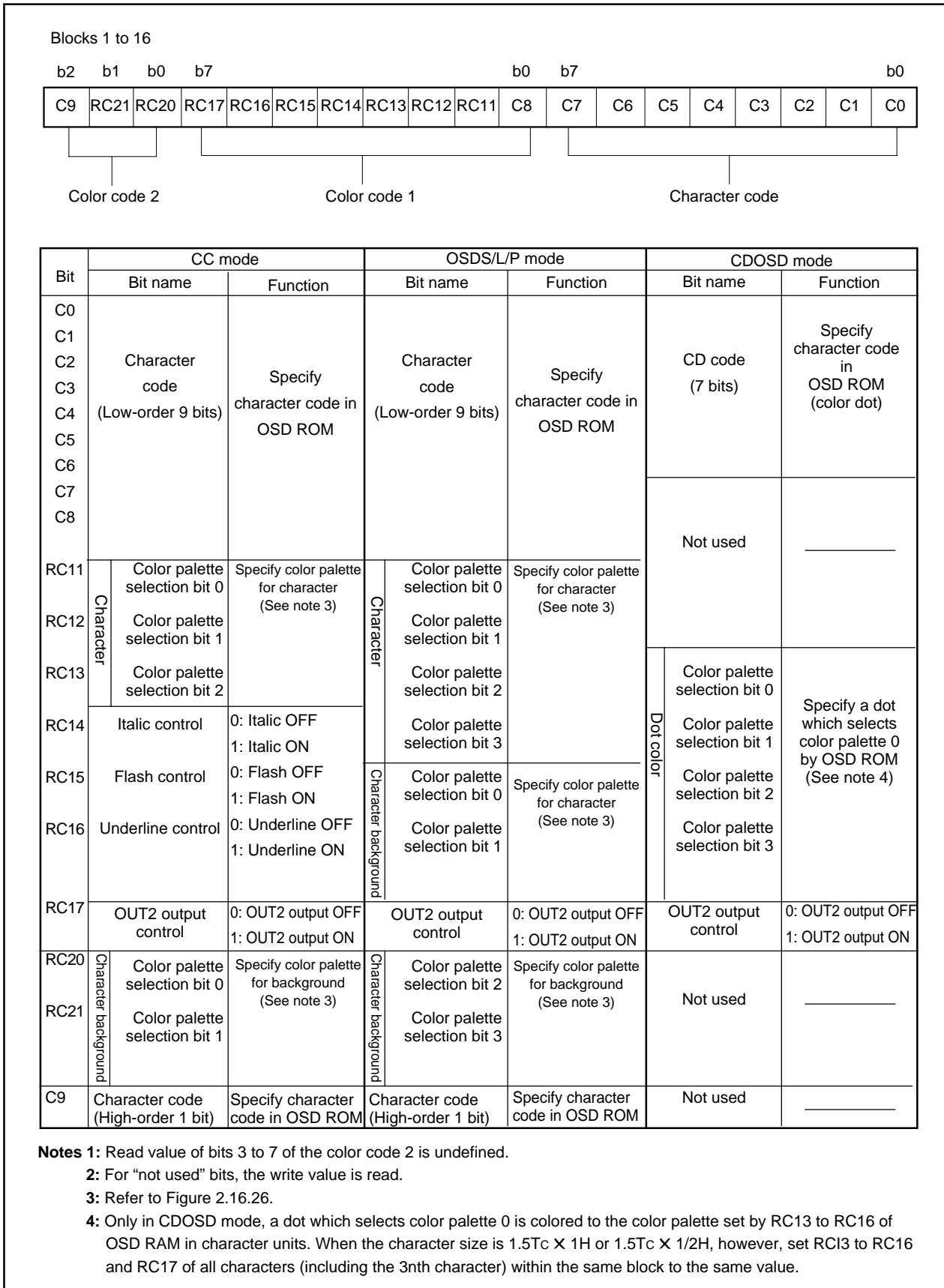


Figure 2.16.25 Structure of OSD RAM

(3) OSD RAM (OSD RAM for SPRITE, addresses 1000₁₆ to 13E7₁₆)

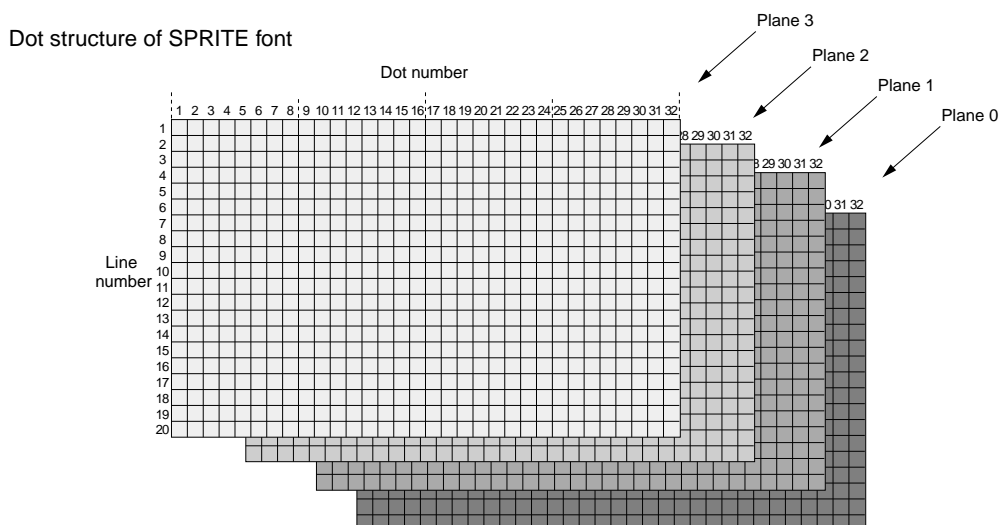
The OSD RAM for SPRITE fonts 1 and 2, consisting of 4 planes for each font, is assigned to addresses 1000₁₆ to 13E7₁₆. Each plane corresponds to each color palette selection bit and the color palette of each dot is determined from among 16 kinds.

Table 2.16.8 OSD RAM address (SPRITE font 1)

| Planes | Plane 3 (Color palette selection bit 3) | | | | Plane 2 (Color palette selection bit 2) | | | | Plane 1 (Color palette selection bit 1) | | | | Plane 0 (Color palette selection bit 0) | | | |
|---------|--|--------------------|--------------------|--------------------|--|--------------------|--------------------|--------------------|--|--------------------|--------------------|--------------------|--|--------------------|--------------------|--------------------|
| Dots | 1 to 8 | 9 to 16 | 17 to 24 | 25 to 32 | 1 to 8 | 9 to 16 | 17 to 24 | 25 to 32 | 1 to 8 | 9 to 16 | 17 to 24 | 25 to 32 | 1 to 8 | 9 to 16 | 17 to 24 | 25 to 32 |
| Bits | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 |
| Line 1 | 10C0 ₁₆ | 10C1 ₁₆ | 11C0 ₁₆ | 11C1 ₁₆ | 1080 ₁₆ | 1081 ₁₆ | 1180 ₁₆ | 1181 ₁₆ | 1040 ₁₆ | 1041 ₁₆ | 1140 ₁₆ | 1141 ₁₆ | 1000 ₁₆ | 1001 ₁₆ | 1100 ₁₆ | 1101 ₁₆ |
| Line 2 | 10C2 ₁₆ | 10C3 ₁₆ | 11C2 ₁₆ | 11C3 ₁₆ | 1082 ₁₆ | 1083 ₁₆ | 1182 ₁₆ | 1183 ₁₆ | 1042 ₁₆ | 1043 ₁₆ | 1142 ₁₆ | 1143 ₁₆ | 1002 ₁₆ | 1003 ₁₆ | 1102 ₁₆ | 1103 ₁₆ |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| Line 19 | 10E4 ₁₆ | 10E5 ₁₆ | 11E4 ₁₆ | 11E5 ₁₆ | 10A4 ₁₆ | 10A5 ₁₆ | 11A4 ₁₆ | 11A5 ₁₆ | 1064 ₁₆ | 1065 ₁₆ | 1164 ₁₆ | 1165 ₁₆ | 1024 ₁₆ | 1025 ₁₆ | 1124 ₁₆ | 1125 ₁₆ |
| Line 20 | 10E6 ₁₆ | 10E7 ₁₆ | 11E6 ₁₆ | 11E7 ₁₆ | 10A6 ₁₆ | 10A7 ₁₆ | 11A6 ₁₆ | 11A7 ₁₆ | 1066 ₁₆ | 1067 ₁₆ | 1166 ₁₆ | 1167 ₁₆ | 1026 ₁₆ | 1027 ₁₆ | 1126 ₁₆ | 1127 ₁₆ |

Table 2.16.9 OSD RAM address (SPRITE font 2)

| Planes | Plane 3 (Color palette selection bit 3) | | | | Plane 2 (Color palette selection bit 2) | | | | Plane 1 (Color palette selection bit 1) | | | | Plane 0 (Color palette selection bit 0) | | | |
|---------|--|--------------------|--------------------|--------------------|--|--------------------|--------------------|--------------------|--|--------------------|--------------------|--------------------|--|--------------------|--------------------|--------------------|
| Dots | 1 to 8 | 9 to 16 | 17 to 24 | 25 to 32 | 1 to 8 | 9 to 16 | 17 to 24 | 25 to 32 | 1 to 8 | 9 to 16 | 17 to 24 | 25 to 32 | 1 to 8 | 9 to 16 | 17 to 24 | 25 to 32 |
| Bits | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 | b7 to b0 |
| Line 1 | 12C0 ₁₆ | 12C1 ₁₆ | 13C0 ₁₆ | 13C1 ₁₆ | 1280 ₁₆ | 1281 ₁₆ | 1380 ₁₆ | 1381 ₁₆ | 1240 ₁₆ | 1241 ₁₆ | 1340 ₁₆ | 1341 ₁₆ | 1200 ₁₆ | 1201 ₁₆ | 1300 ₁₆ | 1301 ₁₆ |
| Line 2 | 12C2 ₁₆ | 12C3 ₁₆ | 13C2 ₁₆ | 13C3 ₁₆ | 1282 ₁₆ | 1283 ₁₆ | 1382 ₁₆ | 1383 ₁₆ | 1242 ₁₆ | 1243 ₁₆ | 1342 ₁₆ | 1343 ₁₆ | 1202 ₁₆ | 1203 ₁₆ | 1302 ₁₆ | 1303 ₁₆ |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| Line 19 | 12E4 ₁₆ | 12E5 ₁₆ | 13E4 ₁₆ | 13E5 ₁₆ | 12A4 ₁₆ | 12A5 ₁₆ | 13A4 ₁₆ | 13A5 ₁₆ | 1264 ₁₆ | 1265 ₁₆ | 1364 ₁₆ | 1365 ₁₆ | 1224 ₁₆ | 1225 ₁₆ | 1324 ₁₆ | 1325 ₁₆ |
| Line 20 | 12E6 ₁₆ | 12E7 ₁₆ | 13E6 ₁₆ | 13E7 ₁₆ | 12A6 ₁₆ | 12A7 ₁₆ | 13A6 ₁₆ | 13A7 ₁₆ | 1266 ₁₆ | 1267 ₁₆ | 1366 ₁₆ | 1367 ₁₆ | 1226 ₁₆ | 1227 ₁₆ | 1326 ₁₆ | 1327 ₁₆ |



2.16.7 Character Color

As shown in Figure 2.16.26, there are 16 built-in color codes. Color palette 0 is fixed at transparent, and color palette 8 is fixed at black. The remaining 14 colors can be set to any of the 512 colors available. The setting procedure for character colors is as follows:

- CC mode 8 kinds
Color palette selection range (color palettes 0 to 7 or 8 to 15) can be selected by bit 0 of the OSD control register 3 (address 0207₁₆). Color palettes are set by bits RC11 to RC13 of the OSD RAM from among the selection range.
- OSDS/L/P mode 16 kinds
Color palettes are set by bits RC11 to RC14 of the OSD RAM.
- CDOSD mode 16 kinds
Color palettes are set in dot units according to CD font data.
Only in CDOSD mode, a dot which selects color palette 0 or 8 is colored to the color palette set by RC13 to RC16 of OSD RAM in character units (refer to Figure 2.16.28).
- SPRITE display 16 kinds
Color palettes are set in dot units according to the CD font data.

Notes 1: Color palette 8 is always selected for bordering and solid space output (OUT 1 output) regardless of the set value in the register.

2: Color palette 0 (transparent) and the transparent setting of other color palettes will differ. When there are multiple layers overlapping (on top of each other, piled up), and the priority layer is color palette 0 (transparent), the bottom layer is displayed, but if the priority layer is the transparent setting of any other color palette, the background is displayed without displaying the bottom layer (refer to Figure 2.16.28).

2.16.8 Character Background Color

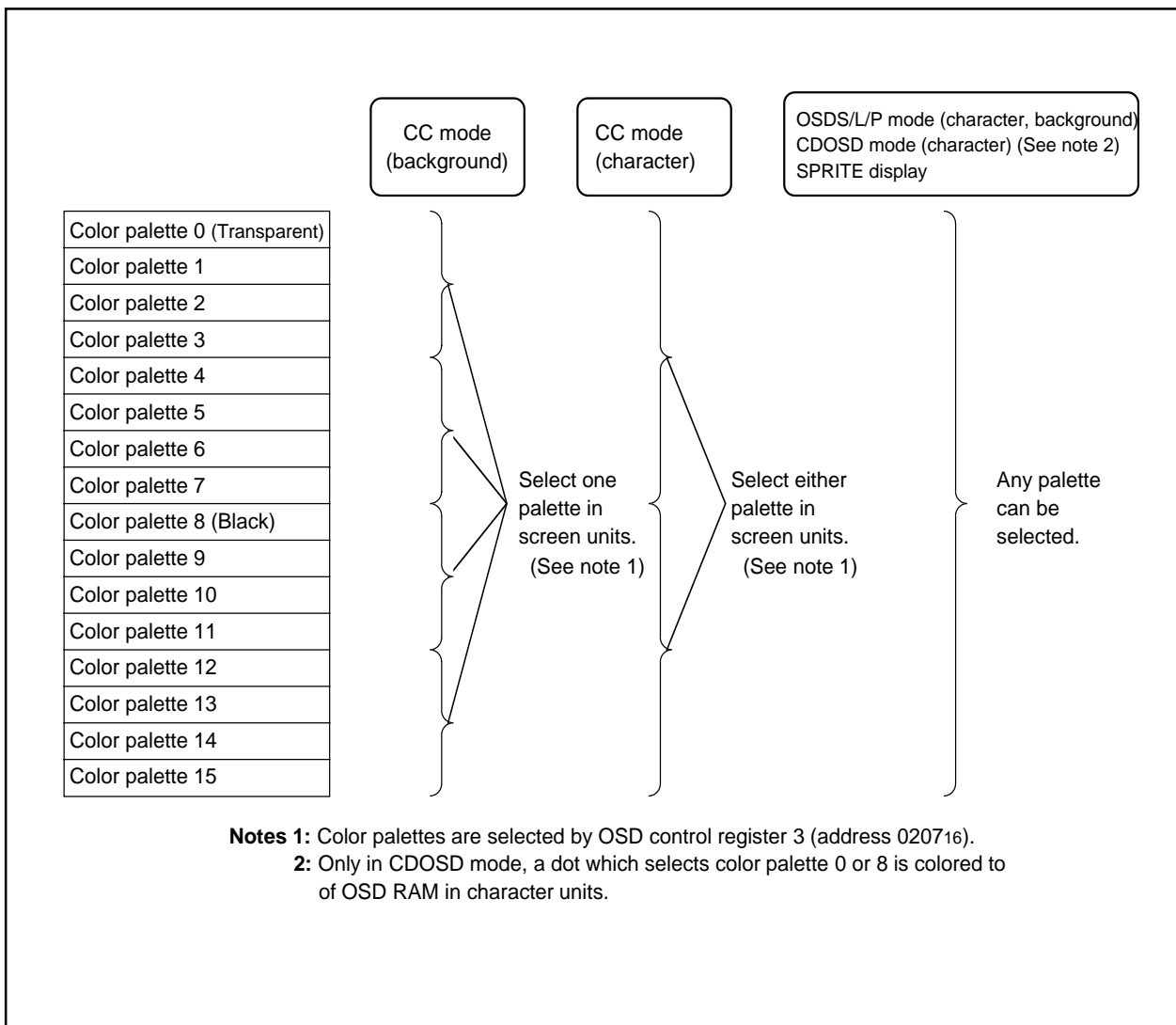
The display area around the characters can be colored in with a character background color. Character background colors are set in character units.

- CC mode 4 kinds
Color palette selection range (color codes 0 to 3, 4 to 7, 8 to 11, or 12 to 15) can be selected by bits 1 and 2 of the OSD control register 3 (address 0207₁₆). Color palettes are set by bits RC20 and RC21 of the OSD RAM from among the selection range.
- OSDS/L/P mode 16 kinds
Color palettes are set by bits RC15, RC16, RC20, and RC21 of the OSD RAM.

Note: The character background is displayed in the following part:

(character display area) – (character font) – (border).

Accordingly, the character background color and the color signal for these two sections cannot be mixed.



Notes 1: Color palettes are selected by OSD control register 3 (address 020716).
2: Only in CDOSD mode, a dot which selects color palette 0 or 8 is colored to of OSD RAM in character units.

Figure 2.16.26 Color palette selection

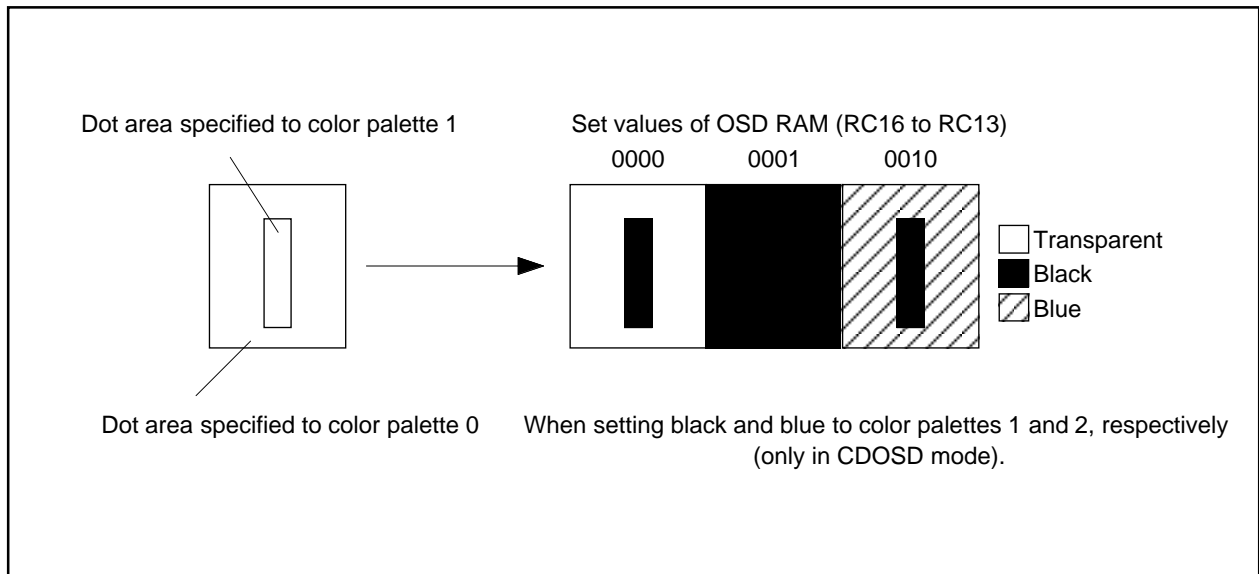


Figure 2.16.27 Set of color palette 0 or 8 in CDOSD mode

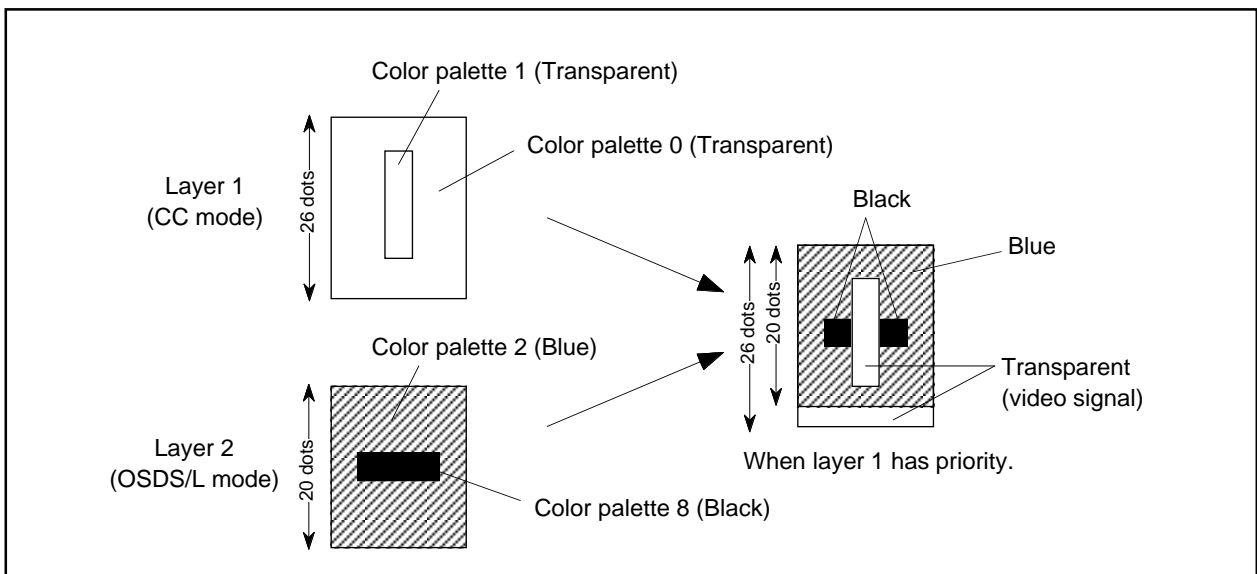


Figure 2.16.28 Difference between color palette 0 (transparent) and transparent setting of other color palettes

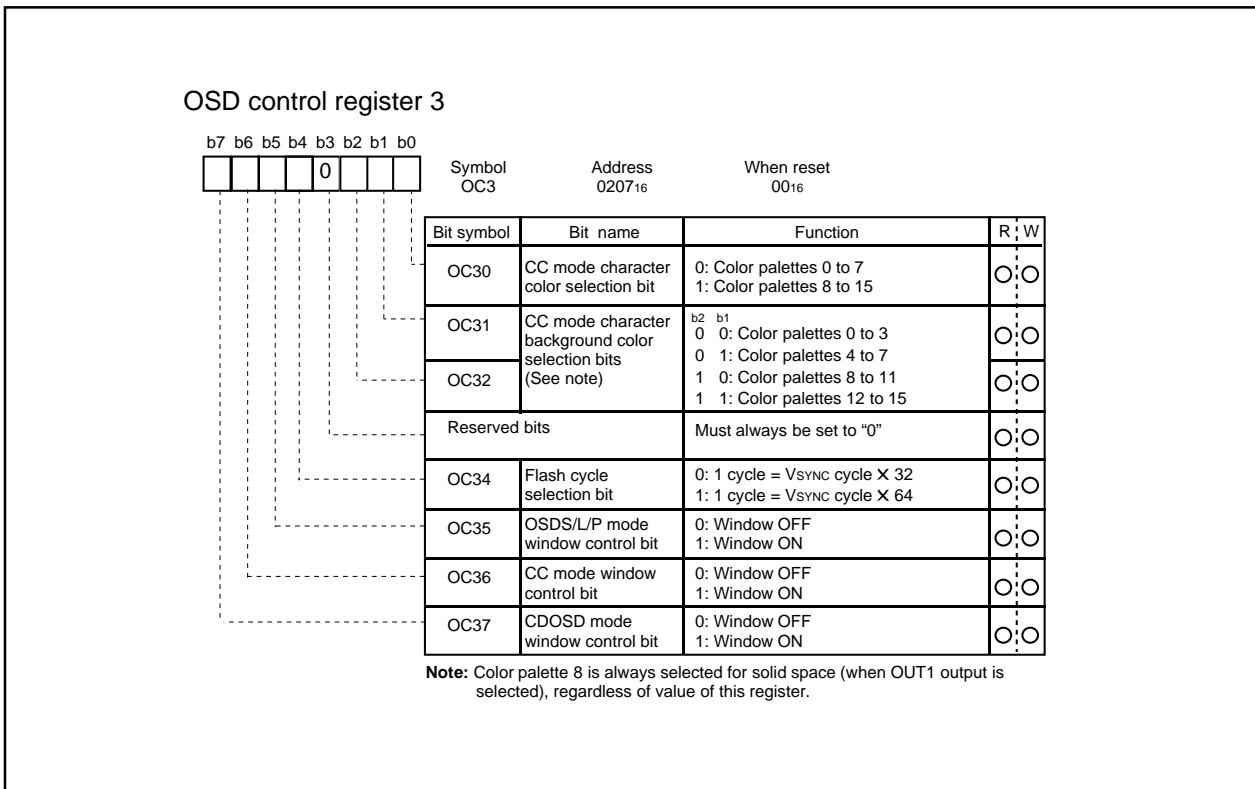


Figure 2.16.29 OSD control register 3

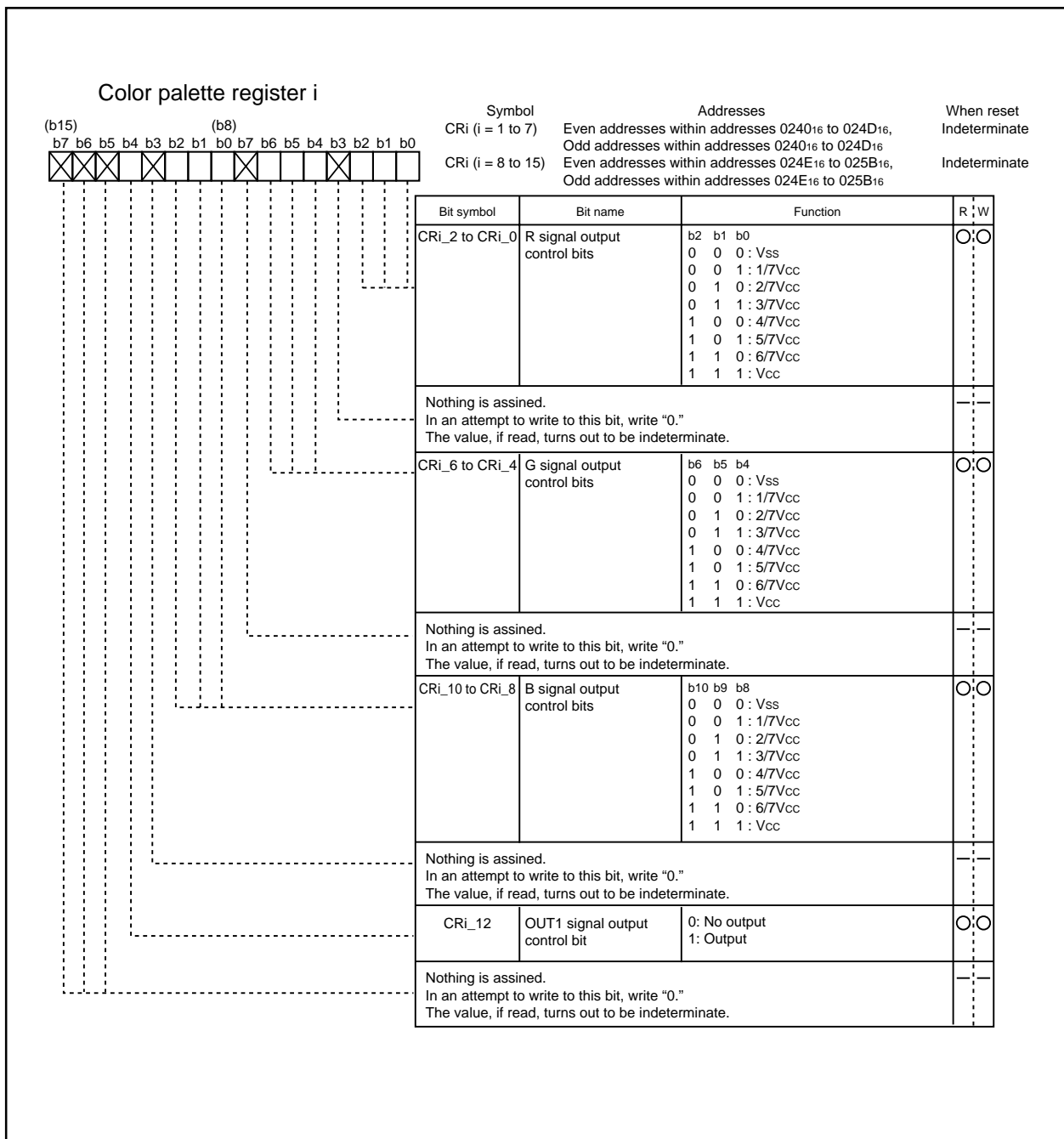


Figure 2.16.30 Color palette register i (i = 1 to 7, 9 to 15)

2.16.9 OUT1, OUT2 Signals

The OUT1, OUT2 signals are used to control the luminance of the video signal. The output waveform of the OUT1, OUT2 signals is controlled by bit 6 of the color palette register *i* (refer to Figure 2.16.30), bits 0 to 2 of the block control register *i* (refer to Figure 2.16.4) and RC17 of OSD RAM. The setting values for controlling OUT1, OUT2 and the corresponding output waveform is shown in Figure 2.16.31.

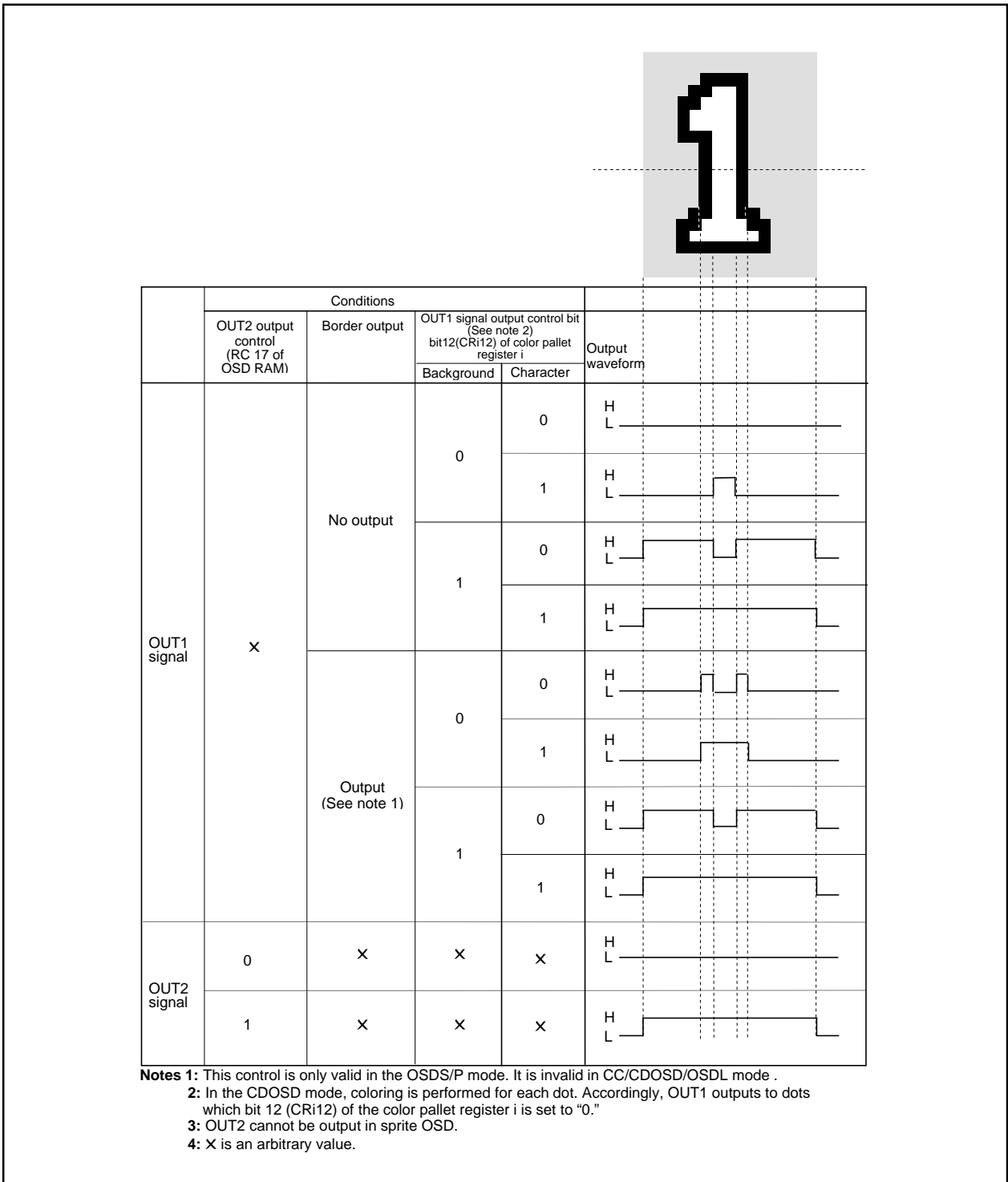


Figure 2.16.31 Setting value for controlling OUT1, OUT2 and corresponding output waveform

2.16.10 Attribute

The attributes (flash, underline, italic fonts) are controlled to the character font. The attributes for each character are specified by RC14 to RC16 of OSD RAM (refer to Figure 2.16.26). The attributes to be controlled are different depending on each mode.

CC mode Flash, underline, italic for each character

OSDS/P mode Border (all bordered, shadow bordered can be selected) for each block

(1) Under line

The underline is output at the 23rd and 24th lines in vertical direction only in the CC mode. The underline is controlled by RC16 of OSD RAM. The color of underline is the same color as that of the character font.

(2) Flash

The parts of the character font, the underline, and the character background are flashed only in the CC mode. The flash for each character is controlled by RC15 of OSD RAM. The ON/OFF for flash is controlled by bit 3 of the OSD control register 1 (refer to Figure 2.16.3). When this bit is "0," only character font and underline flash. When "1," for a character without solid space output, R, G, B and OUT1 (all display area) flash, for a character with solid space output, only R, G, and B (all display area) flash. The flash cycle bases on the VSYNC count and is selected by bit 4 of OSD control register 3.

<NTSC method>

- When bit 4 = "0"
 - VSYNC cycle X 24 ≈ 400 ms (at flash ON)
 - VSYNC cycle X 8 ≈ 133 ms (at flash OFF)
- When bit 4 = "1"
 - VSYNC cycle X 48 ≈ 800 ms (at flash ON)
 - VSYNC cycle X 8 ≈ 133 ms (at flash OFF)

(3) Italic

The italic is made by slanting the font stored in OSD ROM to the right only in the CC mode. The italic is controlled by RC14 of OSD RAM.

The display example attribute is shown in Figure 2.16.33. In this case, "R" is displayed.

Notes 1: When setting both the italic and the flash, the italic character flashes.

2: When a flash character (with flash character background) adjoin on the right side of a non-flash italic character, parts out of the non-flash italic character is also flashed.

3: OUT2 is not flashed.

4: When the pre-divide ratio = 1, the italic character with slant of 1 dot X 5 steps is displayed ; when the pre-divide ratio = 2, the italic character with slant of 1/2 dot X 10 steps is displayed (refer to Figure 2.16.32 (c), (d)). However, when displaying the italic character with the pre-divide ratio = 1, set the OSD clock frequency to 11 MHz to 14 MHz.

5: The boundary of character color is displayed in italic. However, the boundary of character background color is not affected by the italic (refer to Figure 2.16.33).

6: The adjacent character (one side or both side) to an italic character is displayed in italic even when the character is not specified to display in italic (refer to Figure 2.16.33).

7: When displaying the 32nd character (in 32-character mode)/42nd character (in 42-character mode) in the italic and when solid space is off (OC14 = "0"), parts out of character area is not displayed (refer to Figure 2.16.33).

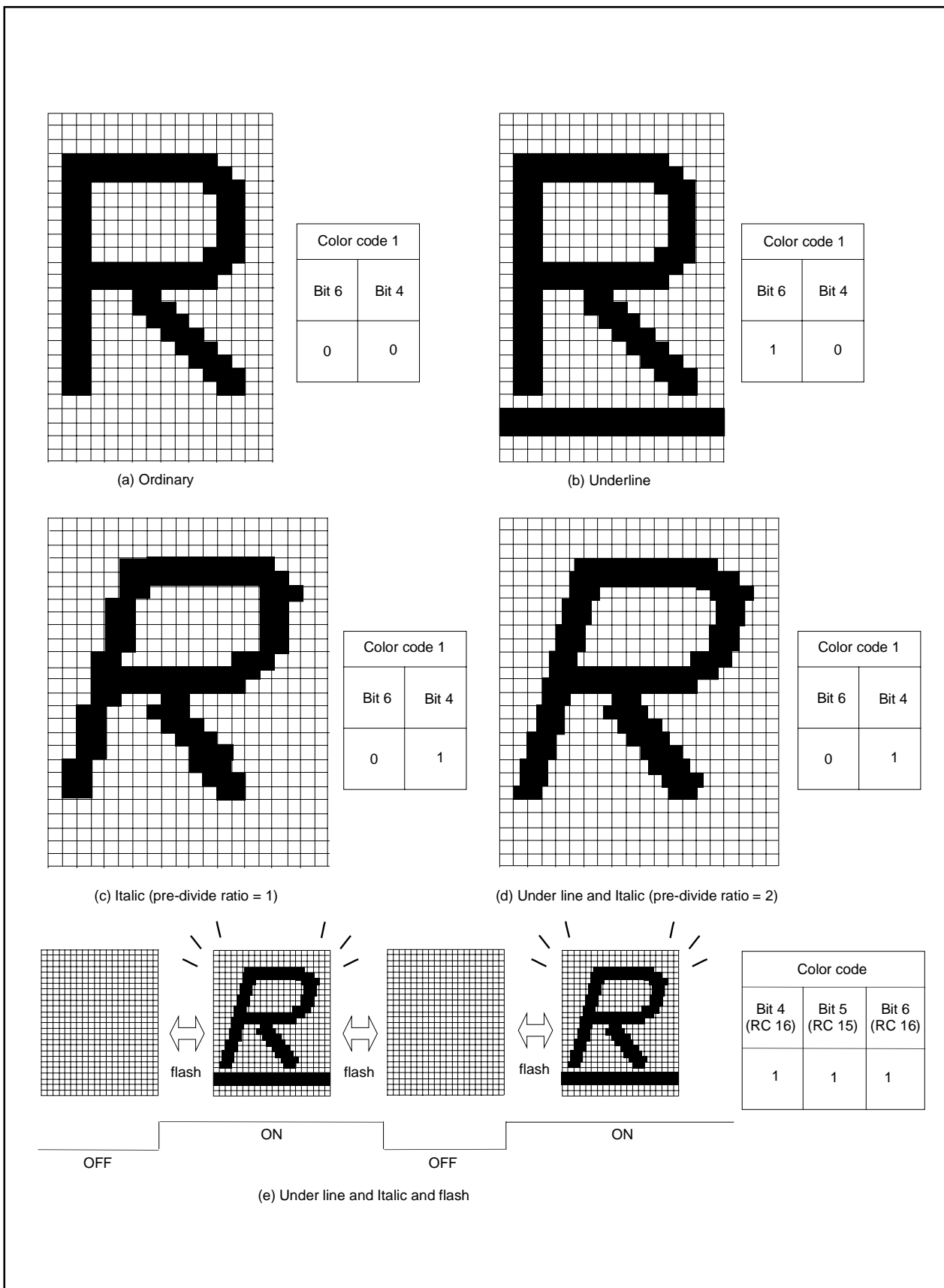


Figure 2.16.32 Example of attribute display (in CC mode)

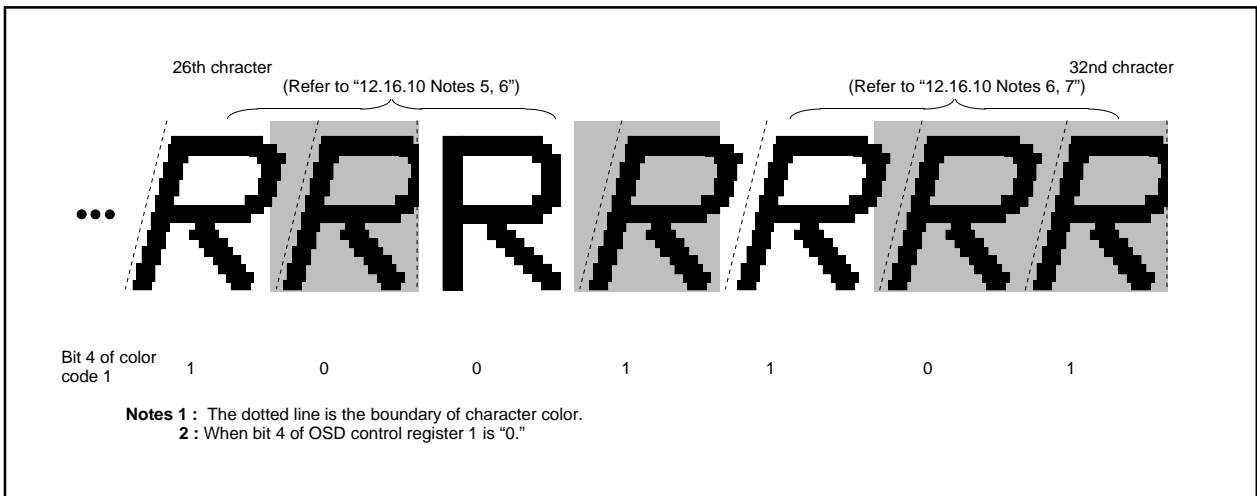


Figure 2.16.33 Example of italic display

(4) Border

The border is output in the OSD/S/P mode. The all bordered (bordering around of character font) and the shadow bordered (bordering right and bottom sides of character font) are selected (refer to Figure 2.16.34) by bit 2 of the OSD control register 1 (refer to Figure 2.16.3). The ON/OFF switch for borders can be controlled in block units by bits 0 to 2 of the block control register i (refer to Figure 2.16.4).

The OUT1 signal is used for border output. The border color is fixed at color palette 8 (black). The border color for each screen is specified by the border color register i.

The horizontal size (x) of border is 1Tc (OSD clock cycle divided in the pre-divide circuit) regardless of the character font dot size. However, only when the pre-divide ratio = 2 and character size = 1.5Tc, the horizontal size is 1.5Tc. The vertical size (y) different depending on the screen scan mode and the vertical dot size of character font.

Notes 1 : The border dot area is the shaded area as shown in Figure 2.16.36.

2 : When the border dot overlaps on the next character font, the character font has priority (refer to Figure 2.16.37 A). When the border dot overlaps on the next character back ground, the border has priority (refer to Figure 2.16.37 B).

3 : The border in vertical out of character area is not displayed (refer to Figure 2.16.38).

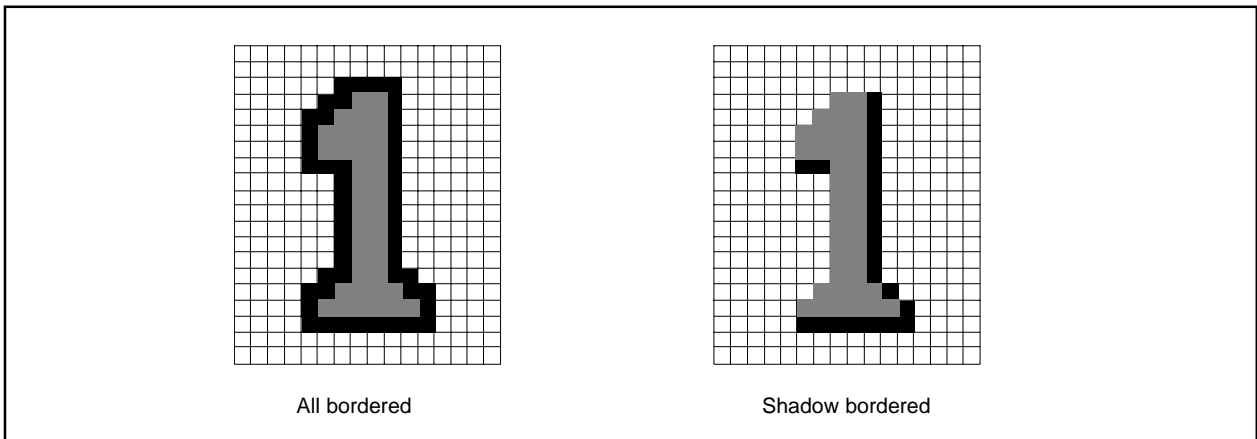


Figure 2.16.34 Example of border display

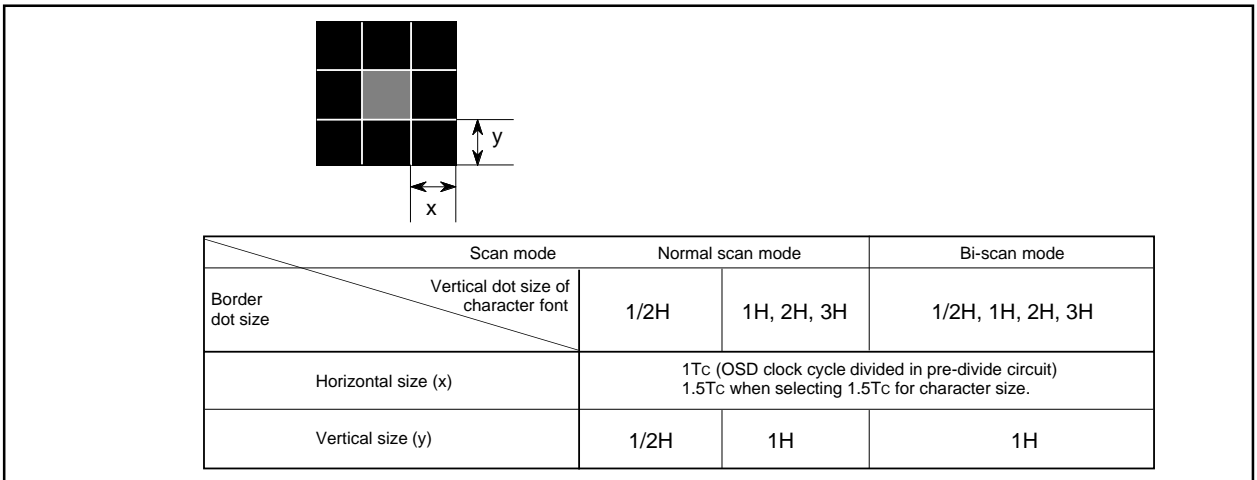


Figure 2.16.35 Horizontal and vertical size of border

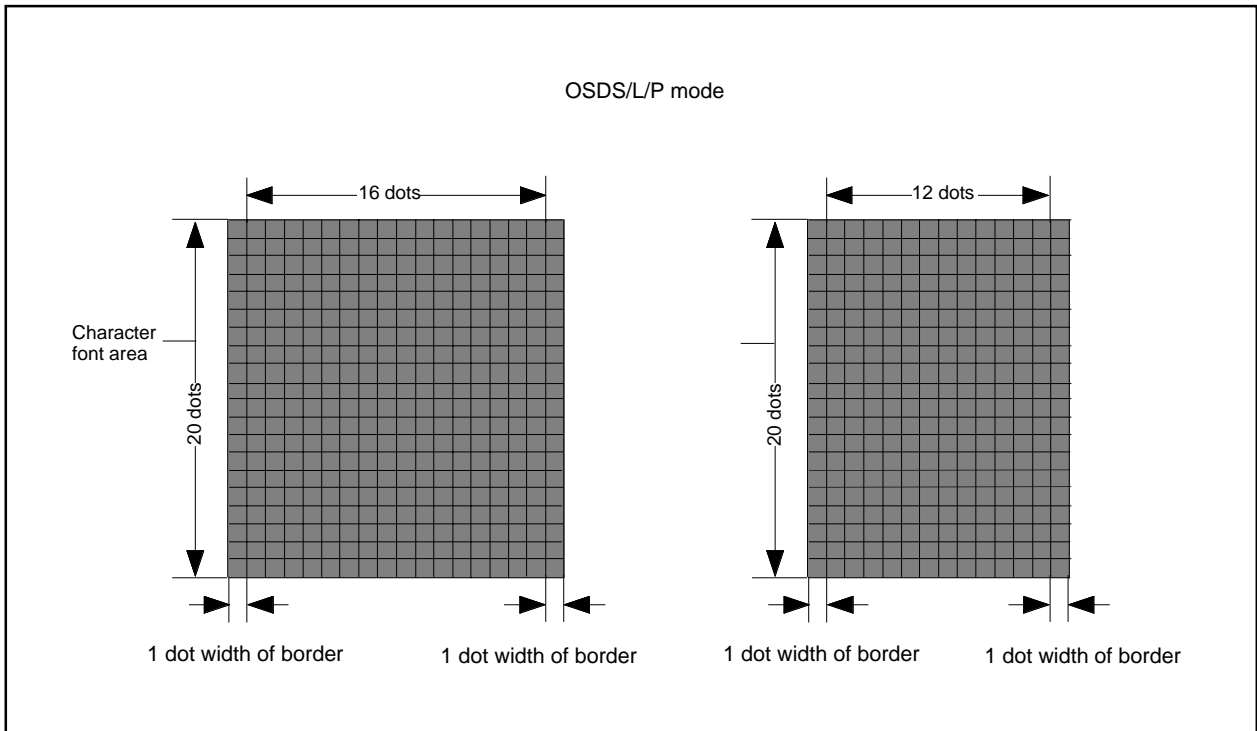


Figure 2.16.36 Border area

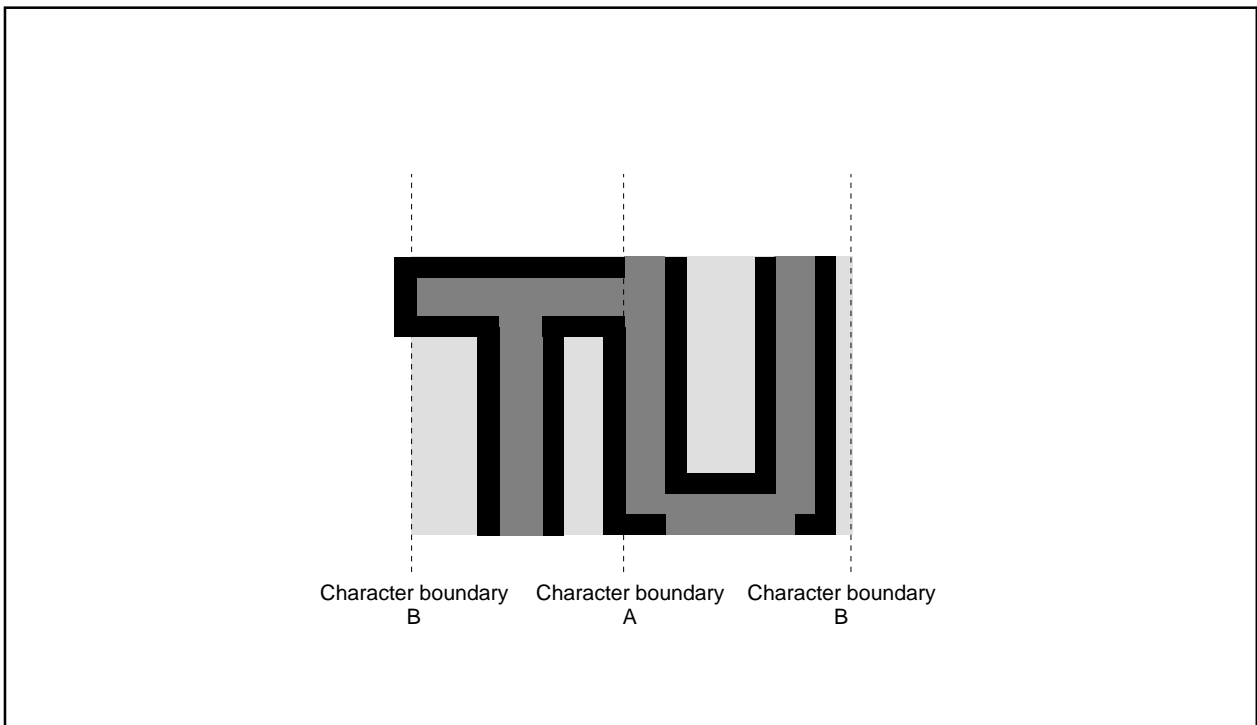


Figure 2.16.37 Border priority

2.16.11 Automatic Solid Space Function

This function generates automatically the solid space (OUT1 or OUT2 blank output) of the character area in the CC mode.

The solid space is output in the following area :

- the character area except character code “00916 ”
- the character area on the left and right sides

This function is turned on and off by bit 4 of the OSD control register 1 (refer to Figure 2.16.3).

OUT1 or OUT2 output is selected by bit 3 of the OSD control register 2.

Notes 1: When selecting OUT1 as solid space output, character background color with solid space output is fixed to color palette 8 (black) regardless of setting.

2: When selecting any font except blank font as the character code “00916,” the set font is output.

Table 2.16.10 Setting for automatic solid space

| Bit 4 of OSD control register 1 | 0 | | | | 1 | | | |
|---------------------------------|--|-------------------------|--|-------------------------|-------------------|-------------------------|--|---|
| Bit 3 of OSD control register 2 | 0 | | 1 | | 0 | | 1 | |
| RC17 of OSD RAM | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| OUT1 output signal | •Character font area •Character background area | | •Character font area •Character background area | | •Solid space area | | •Character font area •Character background area | |
| OUT2 output signal | OFF | •Character display area | OFF | •Character display area | OFF | •Character display area | •Solid space | •Solid space •Character display area |

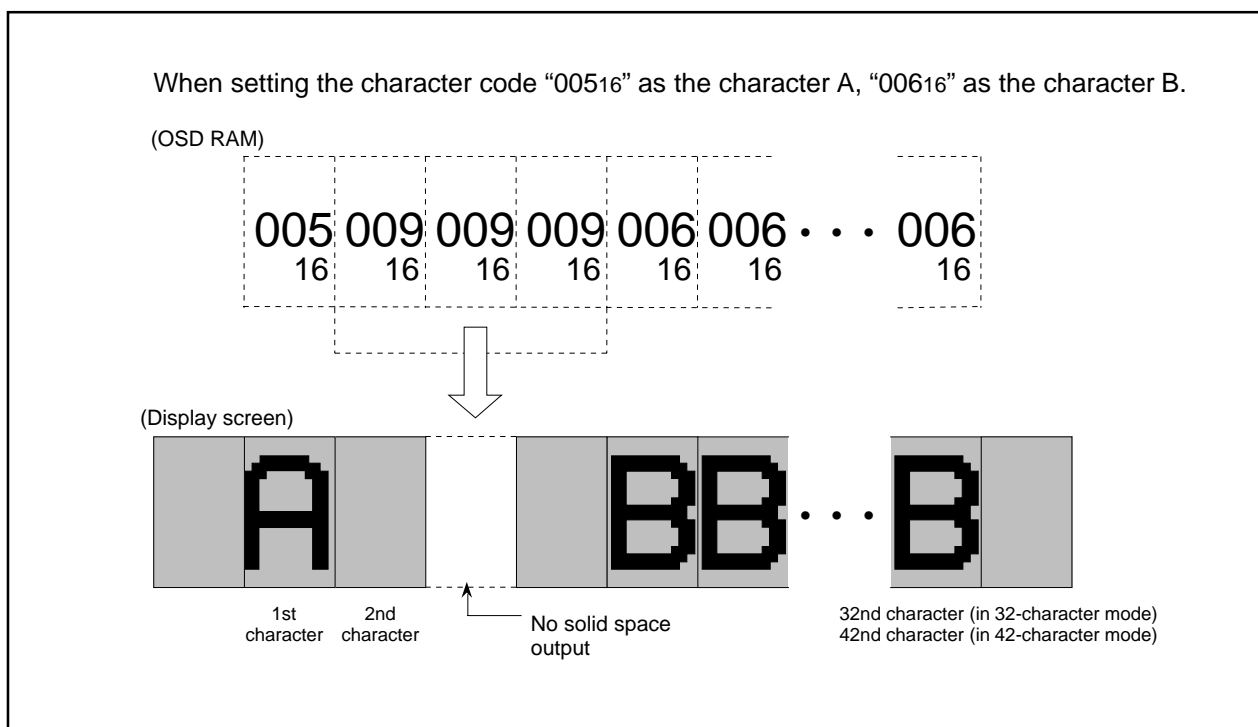


Figure 2.16.38 Display screen example of automatic solid space

2.16.12 Particular OSD Mode Block

This function can display with mixing the fonts below within the OSD mode block.

<horizontal dot structure with vertical dot structure of 20 dots>

- 16 dots
- 12 dots
- 8 dots
- 4 dots

Each font is selected by a character code. Figure 2.16.39 shows the display example of particular OSD mode block and Table 2.16.11 shows the corresponding between character codes and display fonts.

Note: As for 8 X 20-dot and 4 X 20-dot fonts, only these character background color can be displayed. And also, any character is not displayed on the right side area nor any following areas of these fonts.

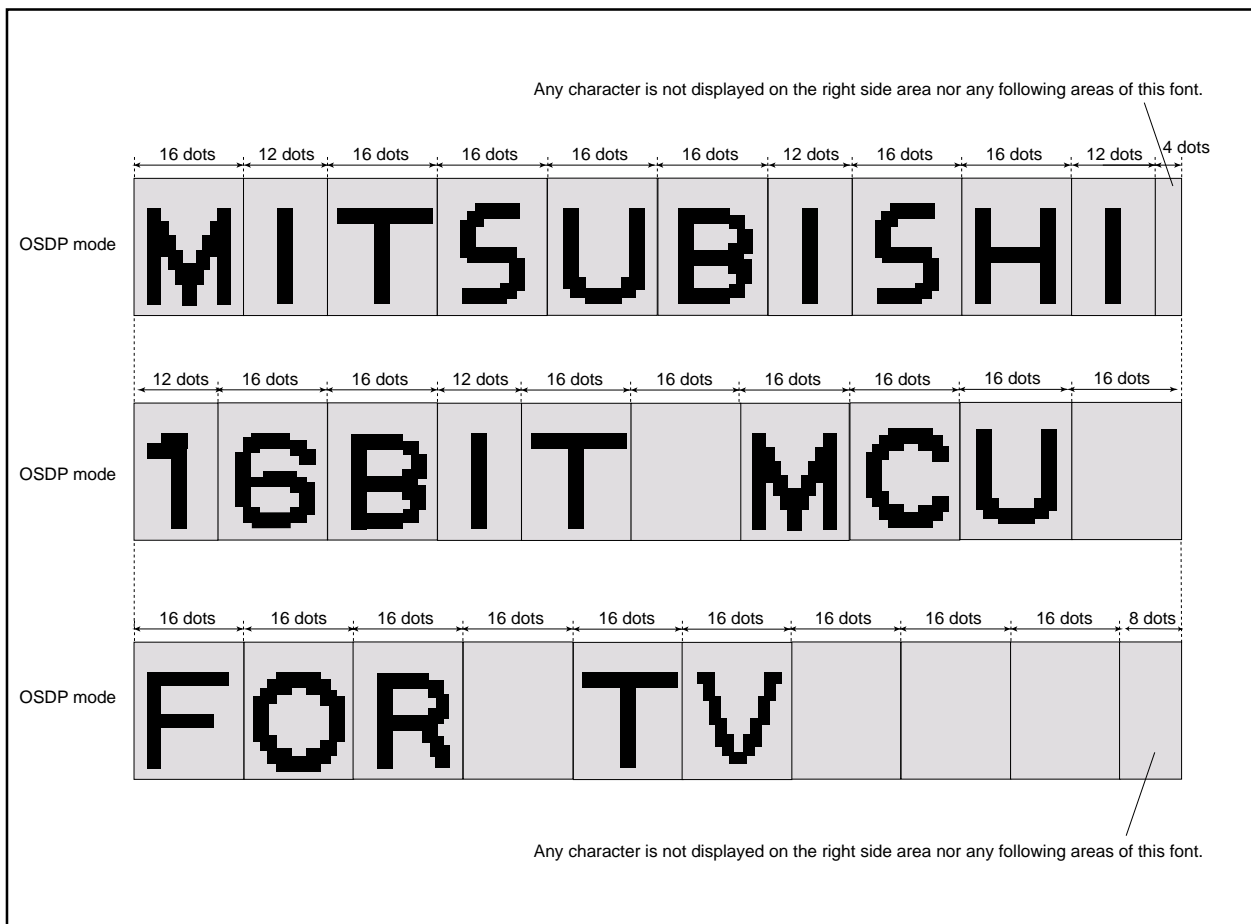
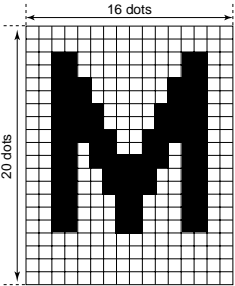
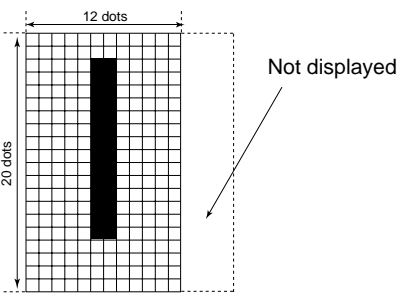
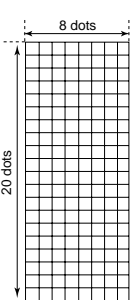
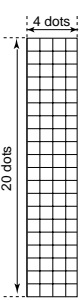


Figure 2.16.39 Display example of OSD mode block

Table 2.16.11 Corresponding between character codes and display fonts

| Character code | Display fonts | Notes |
|---|---|---|
| 00016 to 0EF16, 10016 to 2FF16 (except 10016, 18016, 20016, 28016) |  | |
| 0F016 to 0FD16 |  | <ul style="list-style-type: none"> • The left 12-dot part (16 X 12 dots) of set font is displayed. • In CC and OSDS modes, entire part (16 X 20 dots) of set font is displayed. |
| 3FE16 |  | <ul style="list-style-type: none"> • The blank font (only character background) is displayed. • Any character is not displayed on the right side area nor any following areas of this font. • Do not set this font for the 1st character (left edge) of a block. |
| 3FF16 |  | <ul style="list-style-type: none"> • The blank font (only character background) is displayed. • Any character is not displayed on the right side area nor any following areas of this font. • Do not set this font for the 1st character (left edge) of a block. |

2.16.13 Multiline Display

This microcomputer can ordinarily display 16 lines on the CRT screen by displaying 16 blocks at different vertical positions. In addition, it can display up to 16 lines by using OSD1 interrupts.

An OSD1 interrupt request occurs at the point at which display of each block has been completed. In other words, when a scanning line reaches the point of the display position (specified by the vertical position registers) of a certain block, the character display of that block starts, and an interrupt occurs at the point at which the scanning line exceeds the block. The mode in which an OSD1 interrupt occurs is different depending on the setting of the OSD control register 2 (refer to Figure 2.16.7).

- When bit 7 of the OSD control register 2 is "0"
An OSD1 interrupt request occurs at the completion of layer 1 block display.
- When bit 7 of the OSD control register 2 is "1"
An OSD1 interrupt request occurs at the completion of layer 2 block display.

Notes 1: An OSD1 interrupt does not occur at the end of display when the block is not displayed. In other words, if a block is set to off display by the display control bit of the block control register *i* (addresses 021016 to 021F16), an OSD1 interrupt request does not occur (refer to Figure 2.16.41 (A)).

2: When another block display appears while one block is displayed, an OSD1 interrupt request occurs only once at the end of the another block display (refer to Figure 2.16.40 (B)).

3: On the screen setting window, an OSD1 interrupt occurs even at the end of the CC mode block (off display) out of window (refer to Figure 2.16.40 (C)).

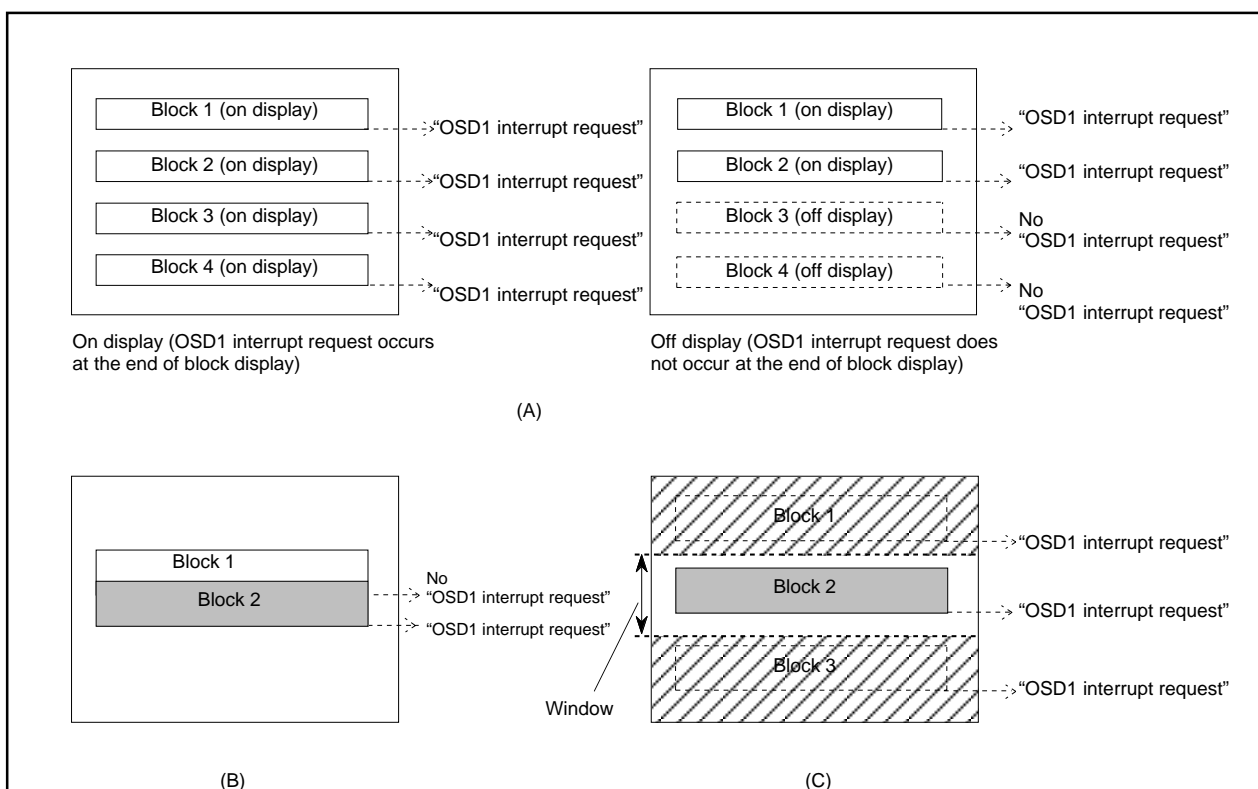


Figure 2.16.40 Note on occurrence of OSD1 interrupt

2.16.14 SPRITE OSD Function

This is especially suitable for cursor and other displays as its function allows for display in any position, regardless of the validity of block OSD displays or display positions. SPRITE font consists of 2 characters: SPRITE fonts 1 and 2. Each SPRITE font is a RAM font consisting of 32 horizontal dots X 20 vertical dots, 4 planes, and 4 bits of data per dot. Each plane has corresponding color palette selection bit, and 16 kinds of color palettes can be selected by the plane bit combination (three bits) for each dot. The color palette is set in dot units according to the OSD RAM (SPRITE) contents from among the selection range. It is possible to add arbitrary font data by software as the SPRITE fonts consist of RAM font.

The SPRITE OSD control register can control SPRITE display and dot size. The display position can also be set independently of the block display by the SPRITE horizontal position registers and the sprite horizontal vertical position registers. The vertical fonts 1 and 2 can be set independently. OSD2 interrupt request occurs at each completion of font display. The horizontal position is set in 2048 steps in 2Tosc units, and the vertical position is set in 1024 steps in 1TH units.

When SPRITE display overlaps with other OSD displays, SPRITE display is always given priority. However, the SPRITE display overlaps with the display which includes OUT2 output, OUT2 in the OSD is output without masking.

Notes 1: The SPRITE OSD function cannot output OUT2.

2: When using SPRITE OSD, do not set $HS \leq "00316"$, $HS \geq "80016"$.

3: When using SPRITE OSD, do not set $VSi = "00016"$, $VSi \geq "40016"$.

4: When displaying with SPRITE fonts 1 and 2 overlapped, the SPRITE font with a larger set value as the vertical display start position is displayed. When the set values of the vertical display start position are the same, the SPRITE font 1 is displayed.

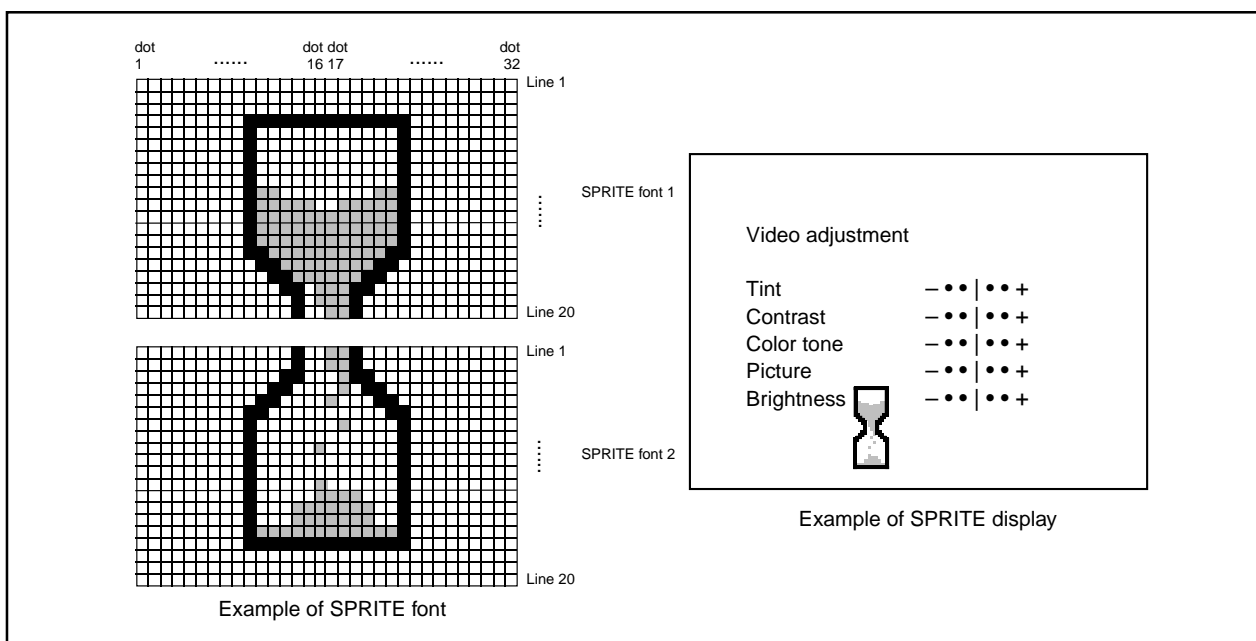


Figure 2.16.41 SPRITE OSD display example

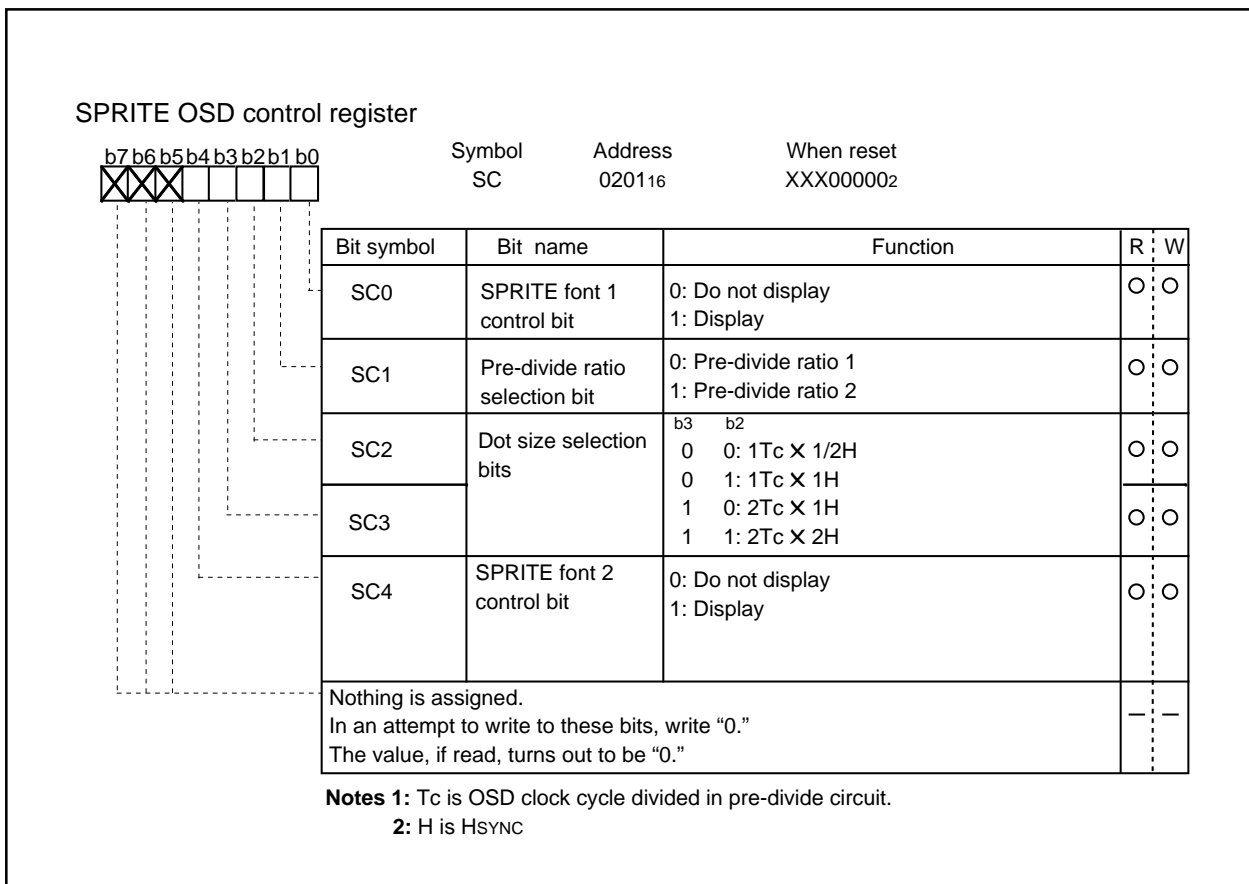


Figure 2.16.42 SPRITE OSD control register

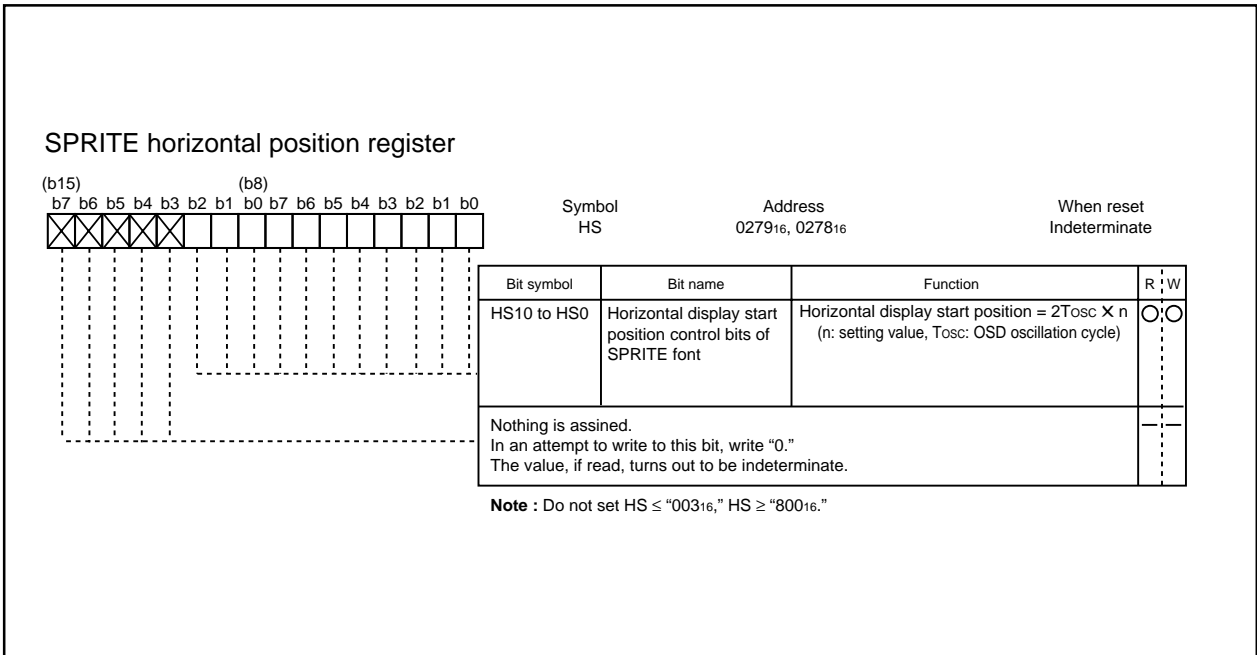


Figure 2.16.43 SPRITE horizontal position register

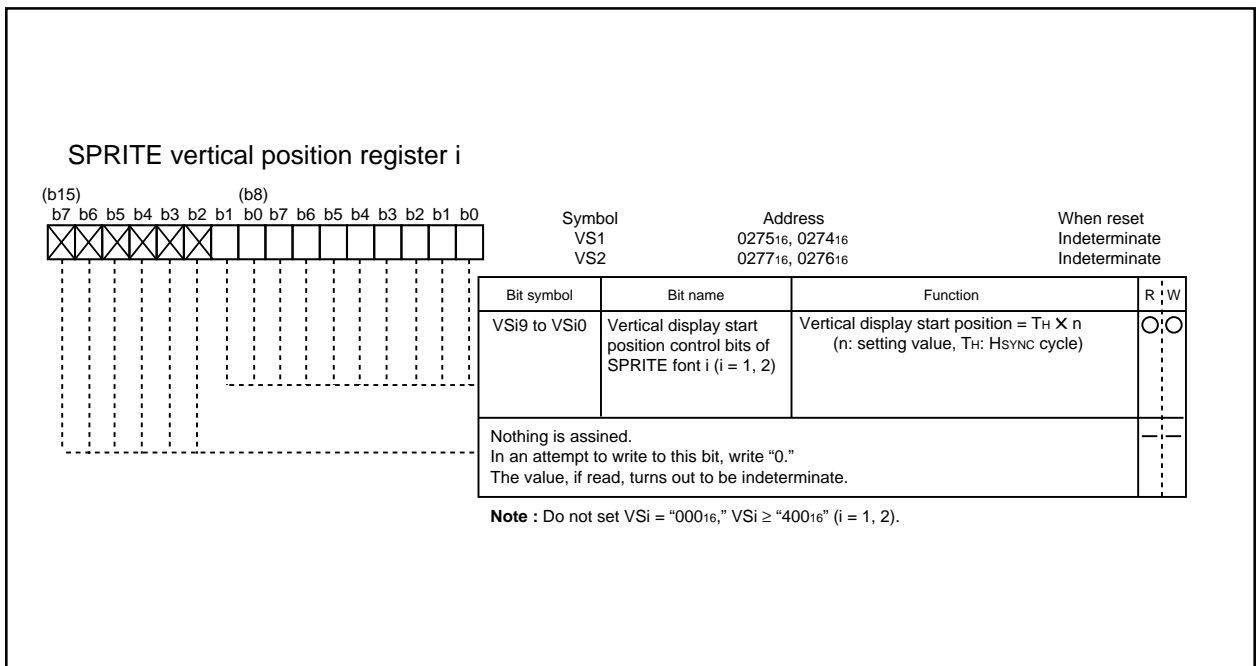


Figure 2.16.44 SPRITE vertical position register i (i = 1, 2)

2.16.15 Window Function

The window function can be set windows on-screen and output OSD within only the area where the window is set.

The ON/OFF for vertical window function is performed by bit 5 of the OSD control register 1 and is used to select vertical window function or vertical blank function by bit 6 of the OSD control register 2. Accordingly, the vertical window function cannot be used simultaneously with the vertical blank function. The display mode to validate the window function is selected by bits 5 to 7 of the OSD control register 3. The top border is set by the top border control register (TBR) and the bottom border is set by the bottom border control register (BBR).

The ON/OFF for horizontal window function is performed by bit 4 of the OSD control register 2 and is used interchangeably for the horizontal blank function with bit 5 of the OSD control register 2. Accordingly, the horizontal blank function cannot be used simultaneously with the horizontal window function. The display mode to validate the window function is selected by bits 5 to 7 of the OSD control register 3. The left border is set by the left border control register (LBR), and the right border is set by the right border control register (RBR).

Notes 1: Horizontal blank and horizontal window, as well as vertical blank and vertical window can not be used simultaneously.

2: When the window function is ON by OSD control registers 1 and 2, the window function of OUT2 is valid in all display mode regardless of setting value of the OSD control register 3 (bits 5 to 7). For example, even when make the window function valid in only CC mode, the function of OUT2 is valid in OSDS/L/P and CDOSD modes.

3: As for SPRITE display, the window function does not operate.

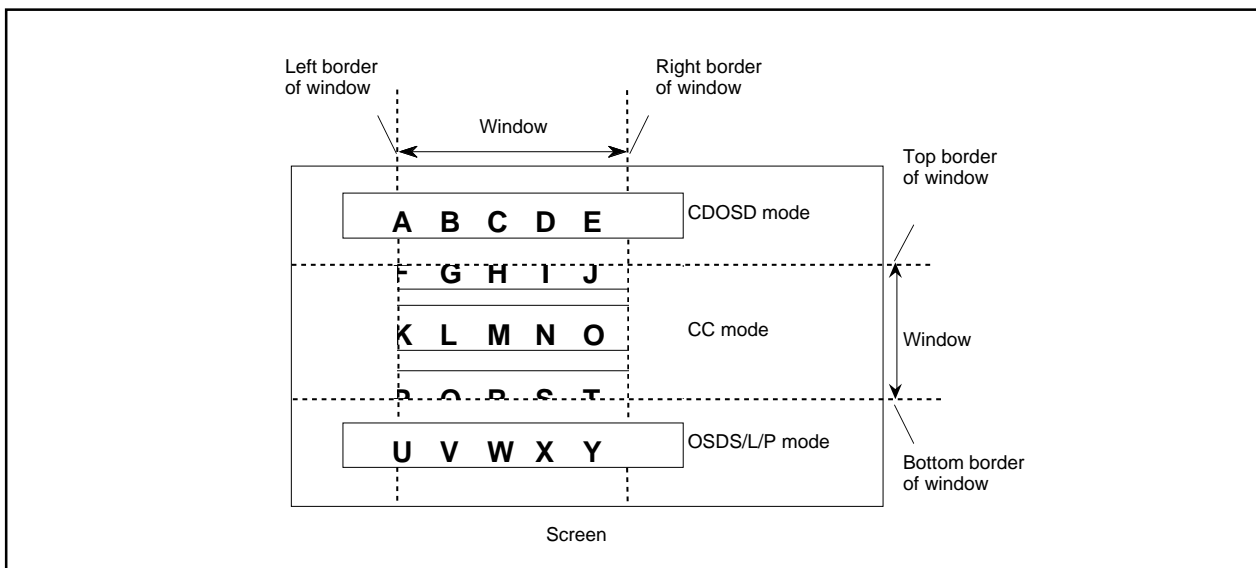


Figure 2.16.45 Example of window function (When CC mode is valid)

2.16.16 Blank Function

The blank function can output blank (OUT1) area on all sides (vertical and horizontal) of the screen. This provides the blank signal, wipe function, etc., when outputting a 3 : 4 image on a wide screen.

The ON/OFF for vertical blank function is performed by bit 5 of the OSD control register 1 and is used to select vertical window function or vertical blank function by bit 6 of the OSD control register 2. Accordingly, the vertical blank function cannot be used simultaneously with the vertical window function. The top border is set by the top border control register (TBR), and the bottom border is set by the bottom border control register (BBR), in 1H units.

The ON/OFF for horizontal blank function is performed by bit 4 of the OSD control register 2 and is used interchangeably for the horizontal window function with bit 5 of the OSD control register 2. Accordingly, the horizontal blank function cannot be used simultaneously with the horizontal window function. The left border is set by the left border control register (LBR) and the right border is set by the right border control register (RBR), in 4TOSC units.

The OSD output (except raster) in area with blank output is not deleted.

These blank signals are not output in the horizontal/vertical blanking interval.

Notes 1. Horizontal blank and horizontal window, as well as vertical blank and vertical window can not be used simultaneously.

2. When using the window function, be sure to set "1" to bit 0 of OSD control register 1.

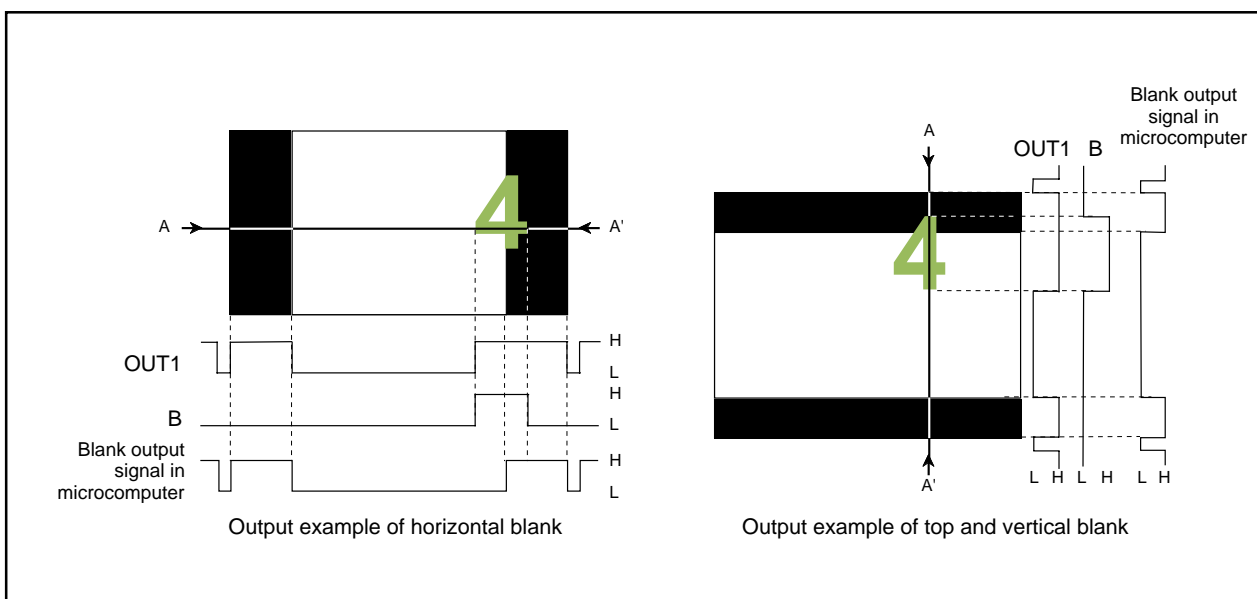


Figure 2.16.46 Blank output example (when OSD output is B + OUT1)

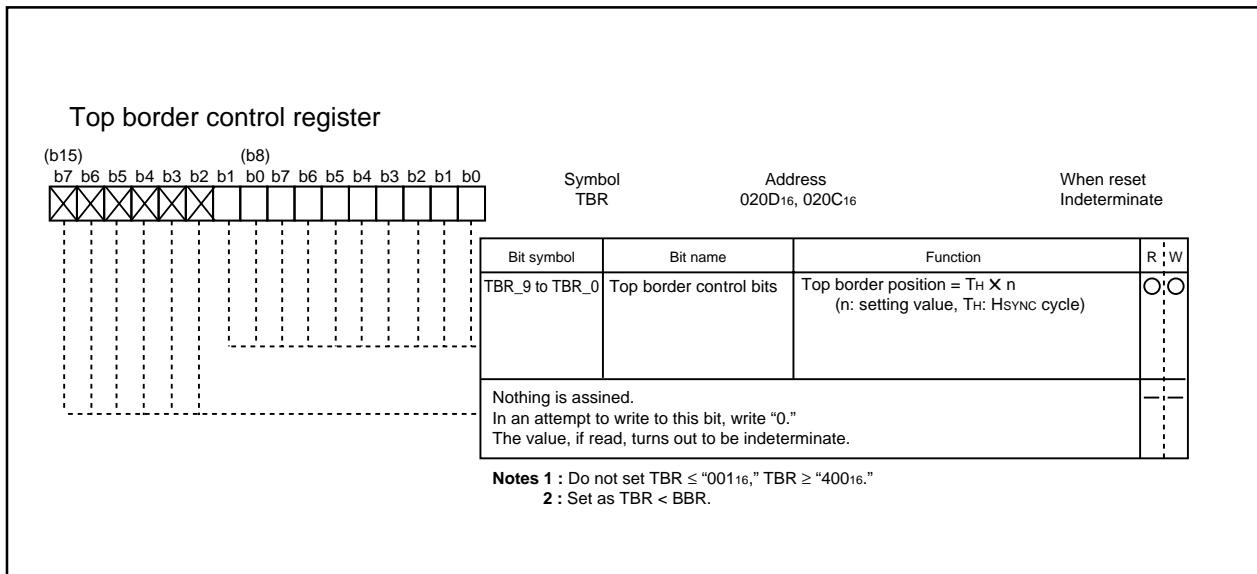


Figure 2.16.47 Top border control register

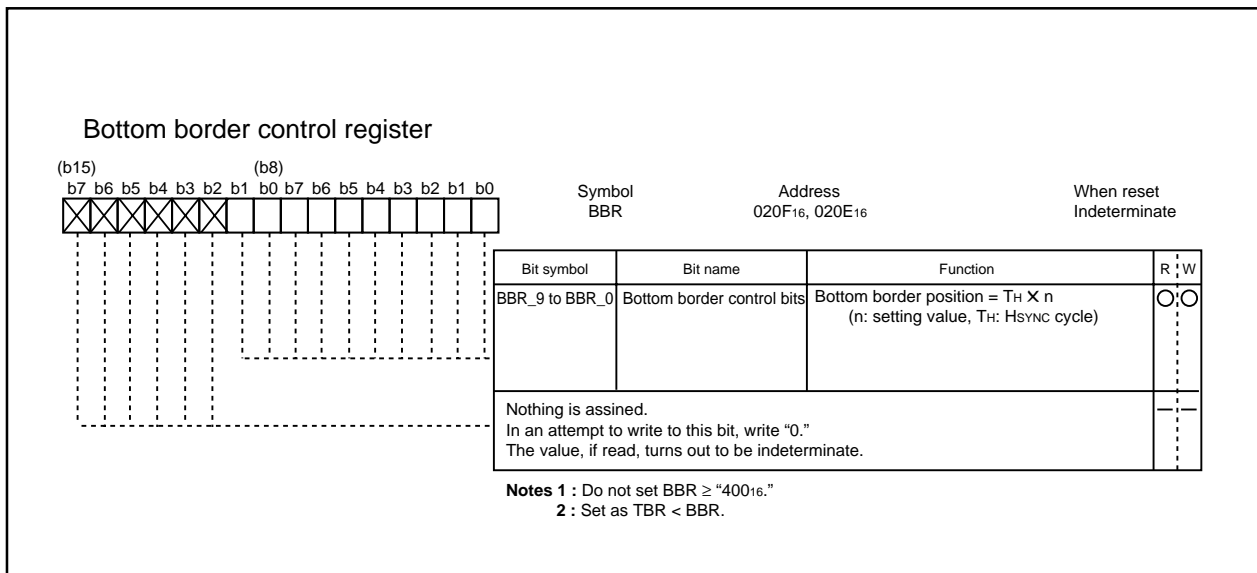


Figure 2.16.48 Bottom border control register

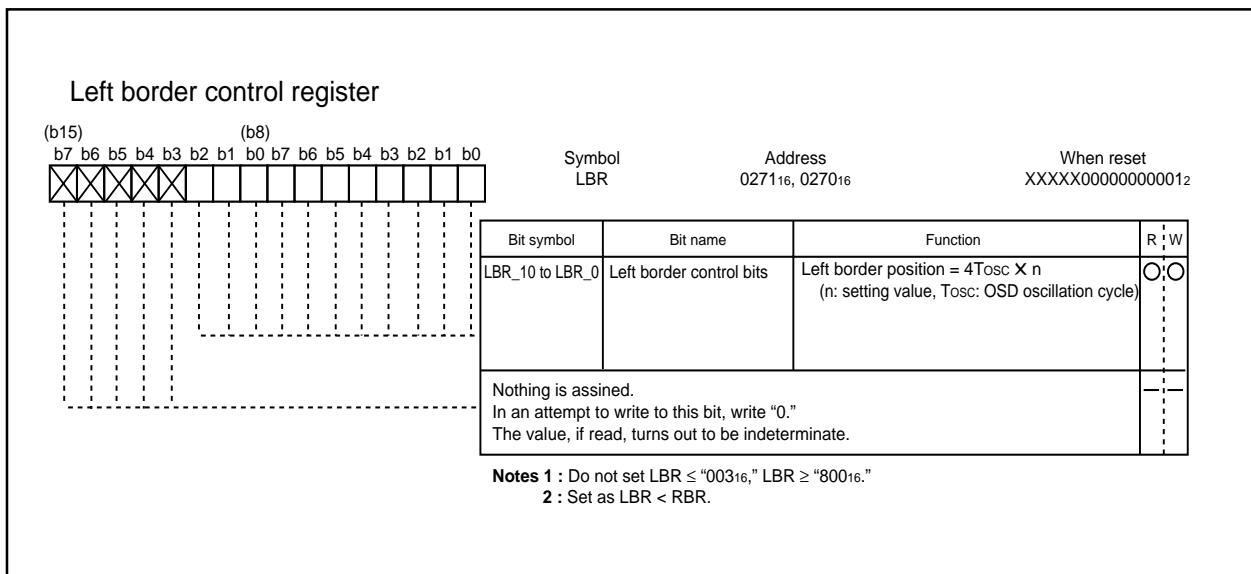


Figure 2.16.49 Left border control register

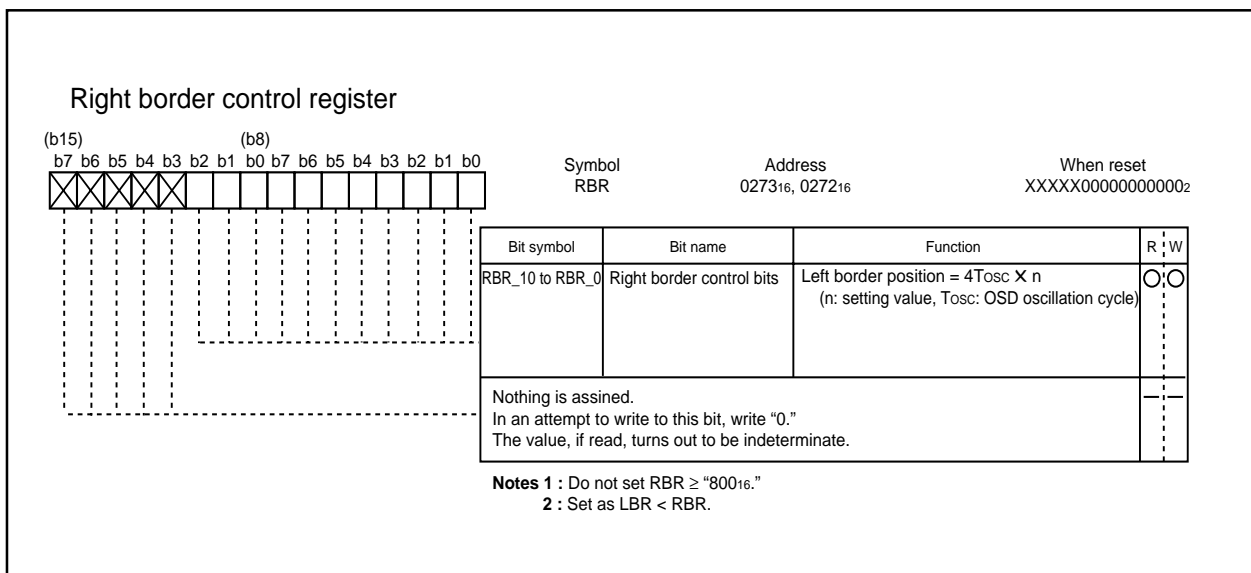


Figure 2.16.50 Right border control register

2.16.17 Raster Coloring Function

An entire screen (raster) can be colored by setting the bits 6 to 0 of the raster color register. Since each of the R, G, B, OUT1, and OUT2 pins can be switched to raster coloring output, 512 raster colors can be obtained.

When the character color/the character background color overlaps with the raster color, the color (R, G, B, OUT1, OUT2), specified for the character color/the character background color, takes priority of the raster color. This ensures that the character color/the character background color is not mixed with the raster color.

The raster color register is shown in Figure 2.16.51, the example of raster coloring is shown in Figure 2.16.52.

Note: Raster is not output to the area which includes blank area.

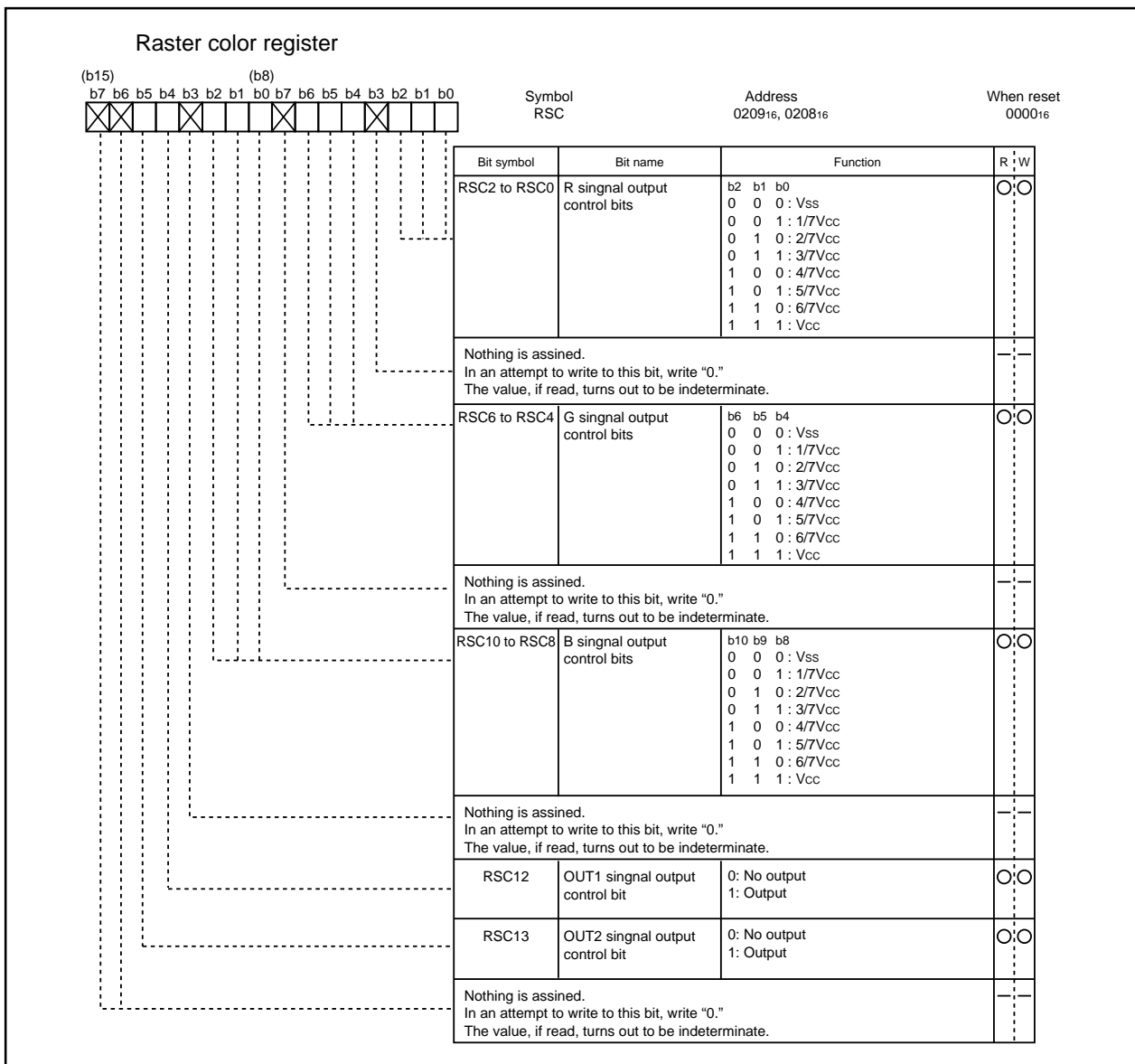


Figure 2.16.51 Raster color register

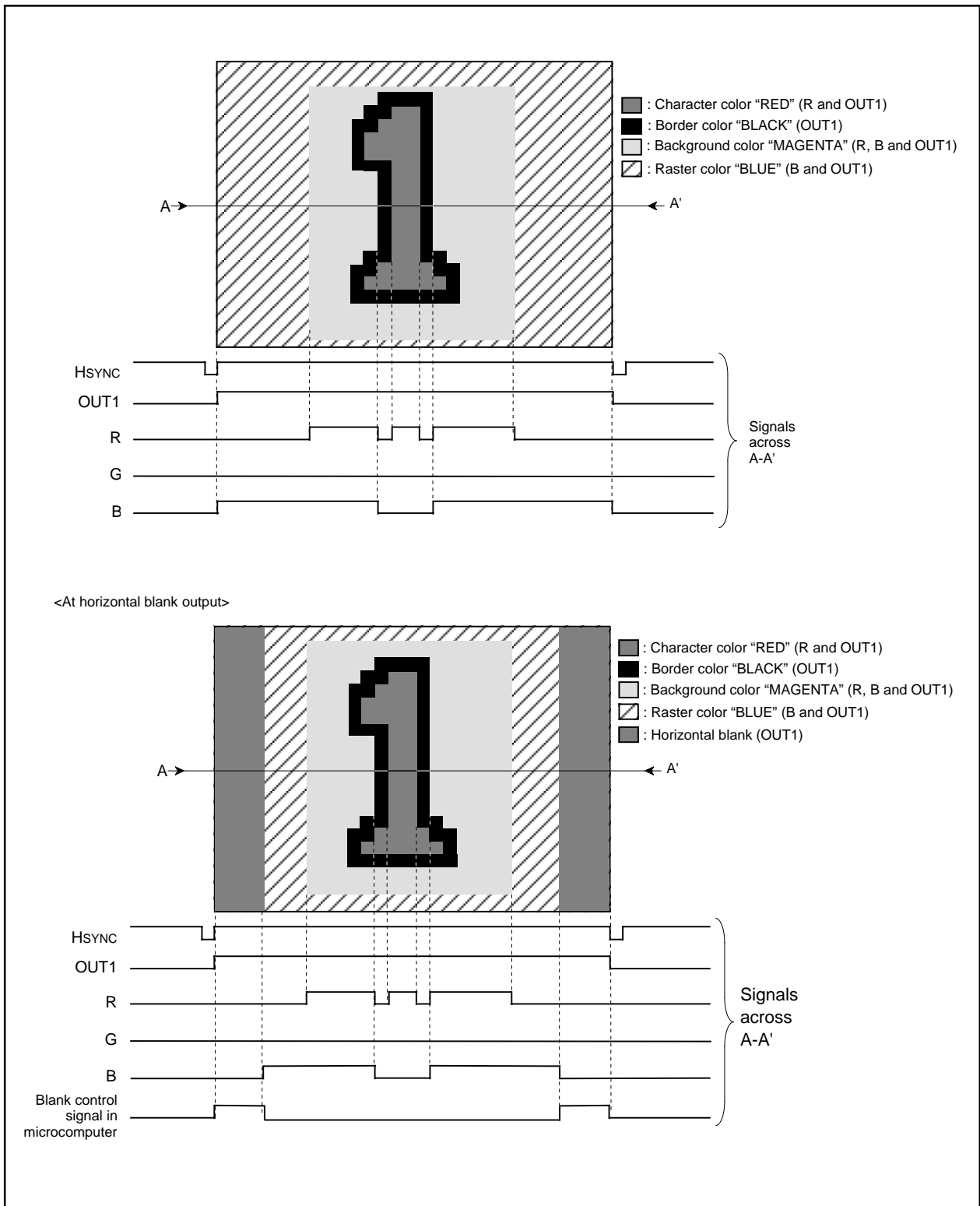


Figure 2.16.52 Example of raster coloring

2.16.18 Scan Mode

This microcomputer has the bi-scan mode for corresponding to HSYNC of double speed frequency. In the bi-scan mode, the vertical start display position and the vertical size is two times as compared with the normal scan mode. The scan mode is selected by bit 1 of the OSD control register 1 (refer to Figure 2.16.3).

Table 2.16.12 Setting for scan mode

| Parameter | Scan Mode | Normal Scan | Bi-Scan |
|---------------------------------|-----------|--|--|
| Bit 1 of OSD control register 1 | | 0 | 1 |
| Vertical display start position | | Value of vertical position register X 1H | Value of vertical position register X 2H |
| Vertical dot size | | 1Tc X 1/2H | 1Tc X 1H |
| | | 1Tc X 1H | 1Tc X 2H |
| | | 2Tc X 2H | 2Tc X 4H |
| | | 3Tc X 3H | 3Tc X 6H |

2.16.19 R, G, B Signal Output Control

The form of R, G, B signal output is controlled by bit 4 of the clock register and bit 2 of the OSD control register 2 as the table below.

Table 2.16.13 R, G, B signal output control

| Bit 4 of clock control register | Bit 2 of OSD control register 2 | Form of R, G, B signal output |
|---------------------------------|---------------------------------|---|
| 0 | 0 | Each R, G, B pin outputs 2 values (digital output). |
| | 1 | Each R, G, B pin outputs 8 values (analog output). |
| 1 | | Each R, R' (P73), G, G' (P60), B, B' (P57) pin outputs 2 values. (Corresponding to each signal output control bits of color palette register i) R, G, B: CRi_1, CRi_5, CRi_9, respectively R', G', B': CRi_0, CRi_4, CRi_8, respectively |

Note: When bit 4 of the clock control register is "1," ports P57, P60, P73 function as OSD function pins R', G', B', respectively. When emulating, however, set bit 4 to "0."

2.16.20 OSD Reserved Register

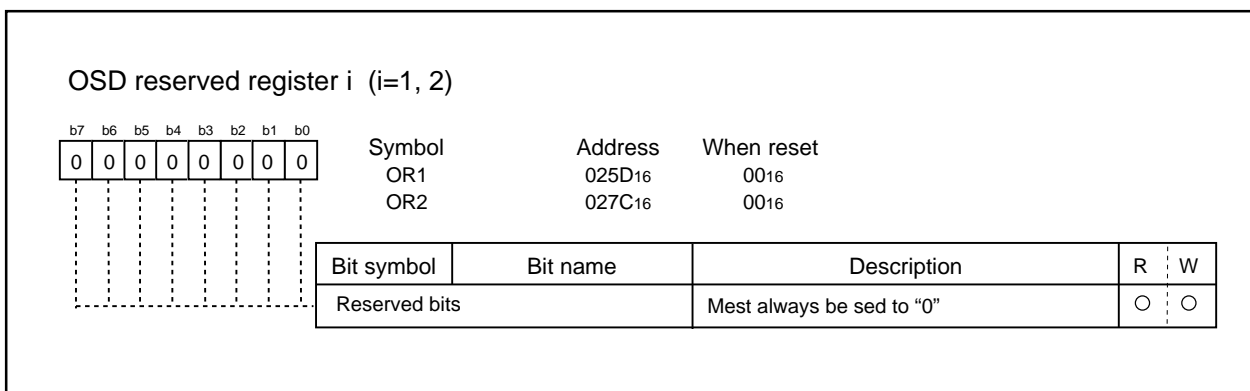


Figure 2.16.53 OSD reserved register i (i=1, 2)

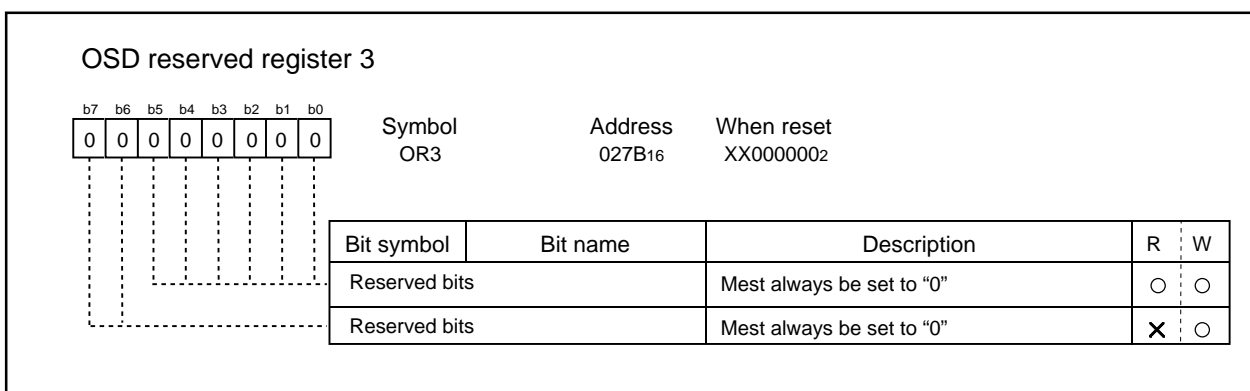


Figure 2.16.54 OSD reserved register 3

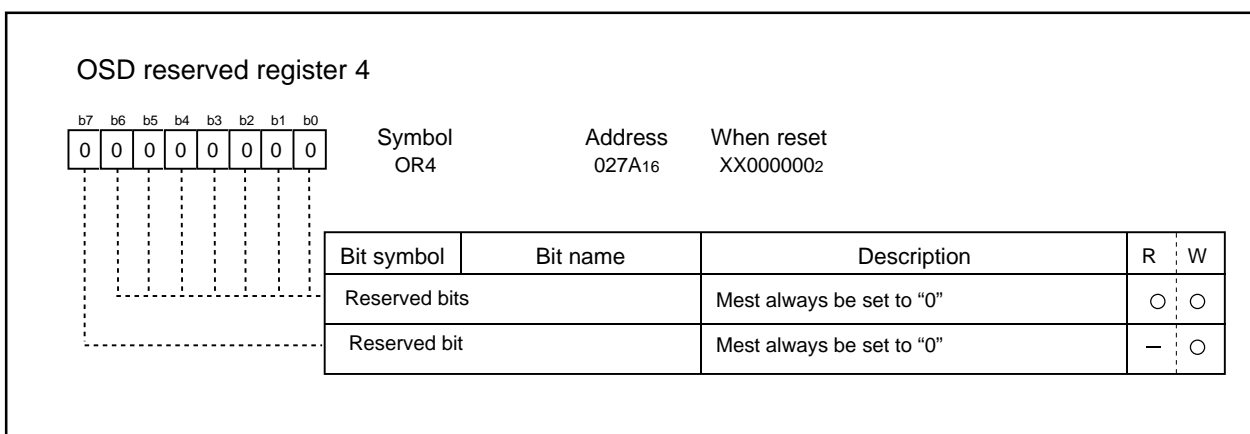


Figure 2.16.55 OSD reserved register 4

2.17 Programmable I/O Ports

There are 78 programmable I/O ports: P0–P5, P60–P63, P67, P7, P82, P83, P86, P87, P90–P94 and P10. 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.

Figures 2.17.1 to 2.17.4 show the programmable I/O ports.

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.

2.17.1 Direction Registers

Figures 2.17.6 to 2.17.9 show 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.

(1) Effect of the protection register

Data written to the direction register of P9 is affected by the protection register. The direction register of P9 cannot be easily written.

2.17.2 Port Registers

Figures 2.17.10 to 2.17.13 show 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.

(1) Reading a port register

With the direction register set to output, reading a port register takes out the content of the port register, not the content of the pin. With the direction register set to input, reading the port register takes out the content of the pin.

(2) Writing to a port register

With the direction register set to output, the level of the written values from each relevant pin is output by writing to a port register. Writing to the port register, with the direction register set to input, inputs a value to the port register, but nothing is output to the relevant pins. The output level remains floating.

2.17.3 Pull-up Control Registers

Figures 2.17.15 to 2.17.17 show 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, pull-up control register of P0 to P5 is invalid.

2.17.4 Port Control Register

Figure 2.17.14 shows the port control register.

The bit 0 of port control register 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 through port P1 is input/output port.

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.

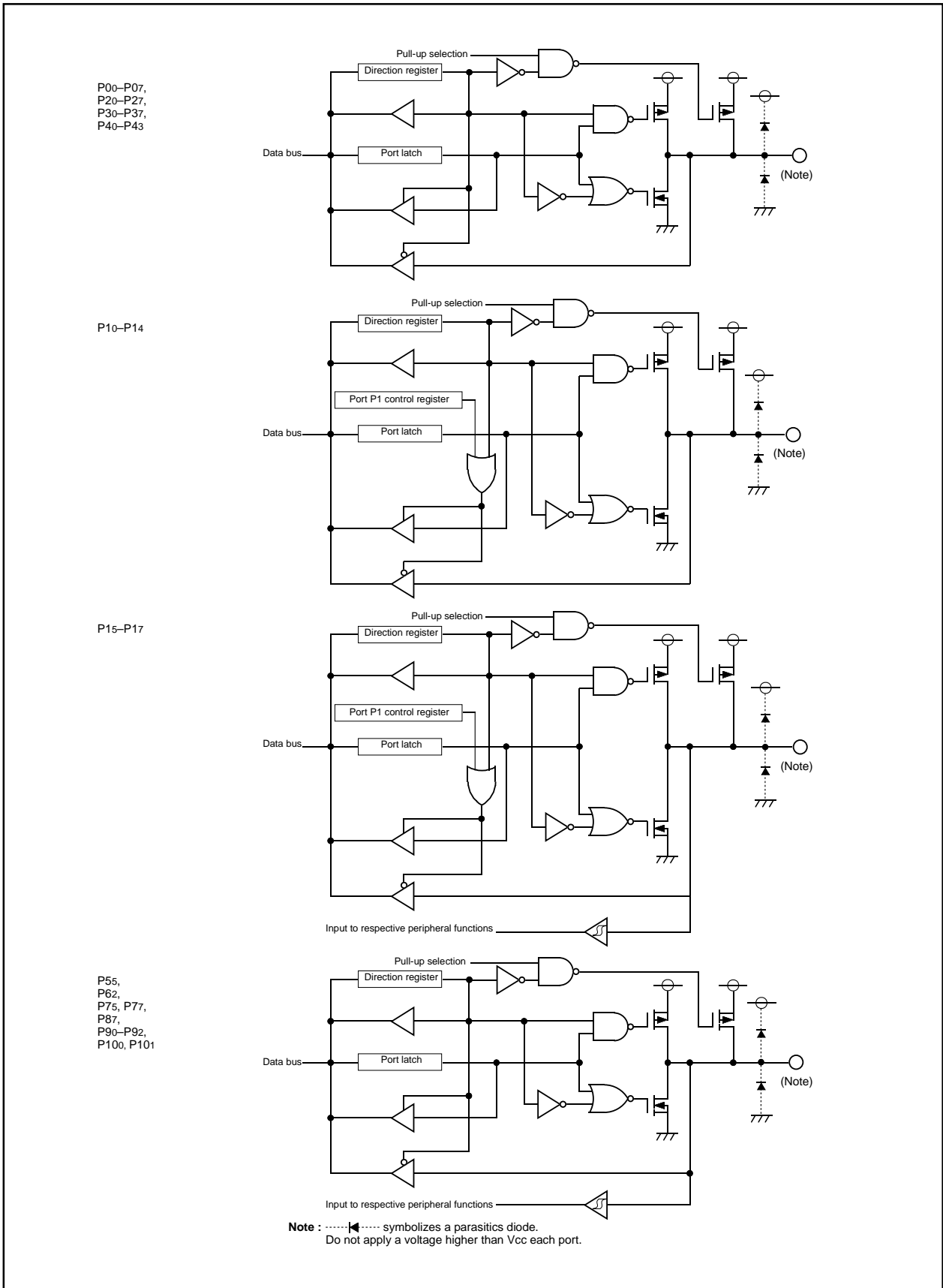


Figure 2.17.1 Programmable I/O ports (1)

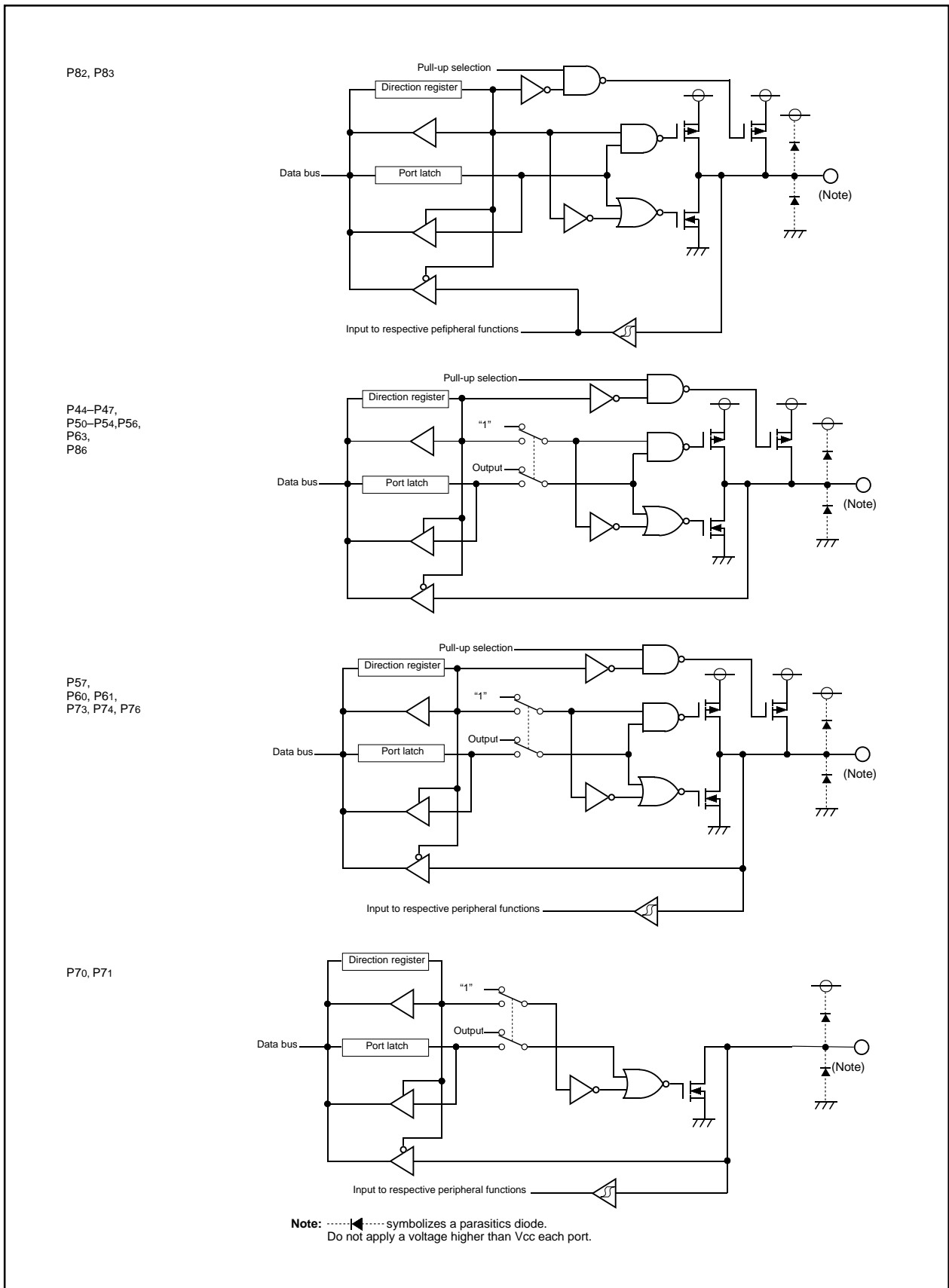


Figure 2.17.2 Programmable I/O ports (3)

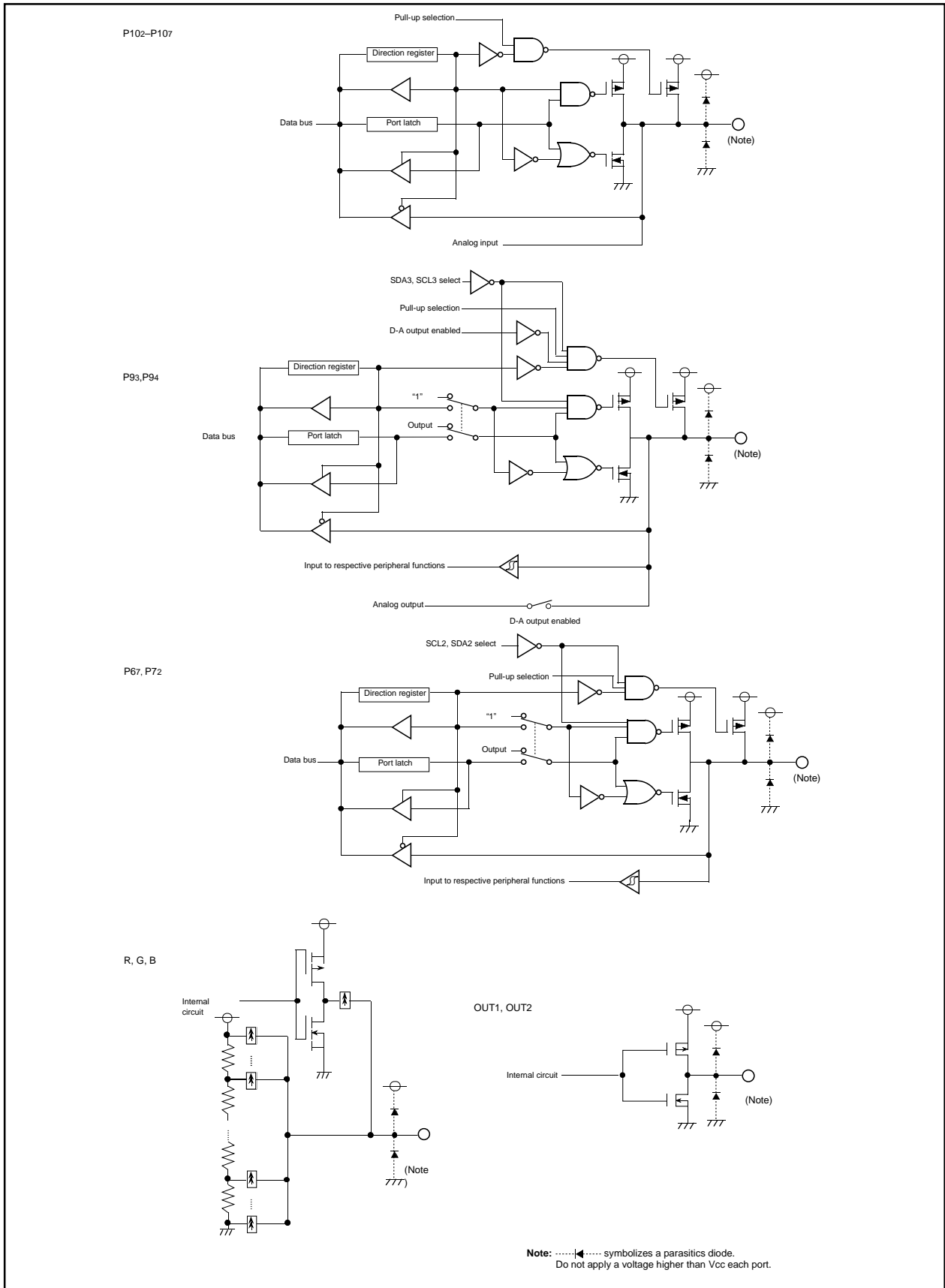


Figure 2.17.3 Programmable I/O ports (2)

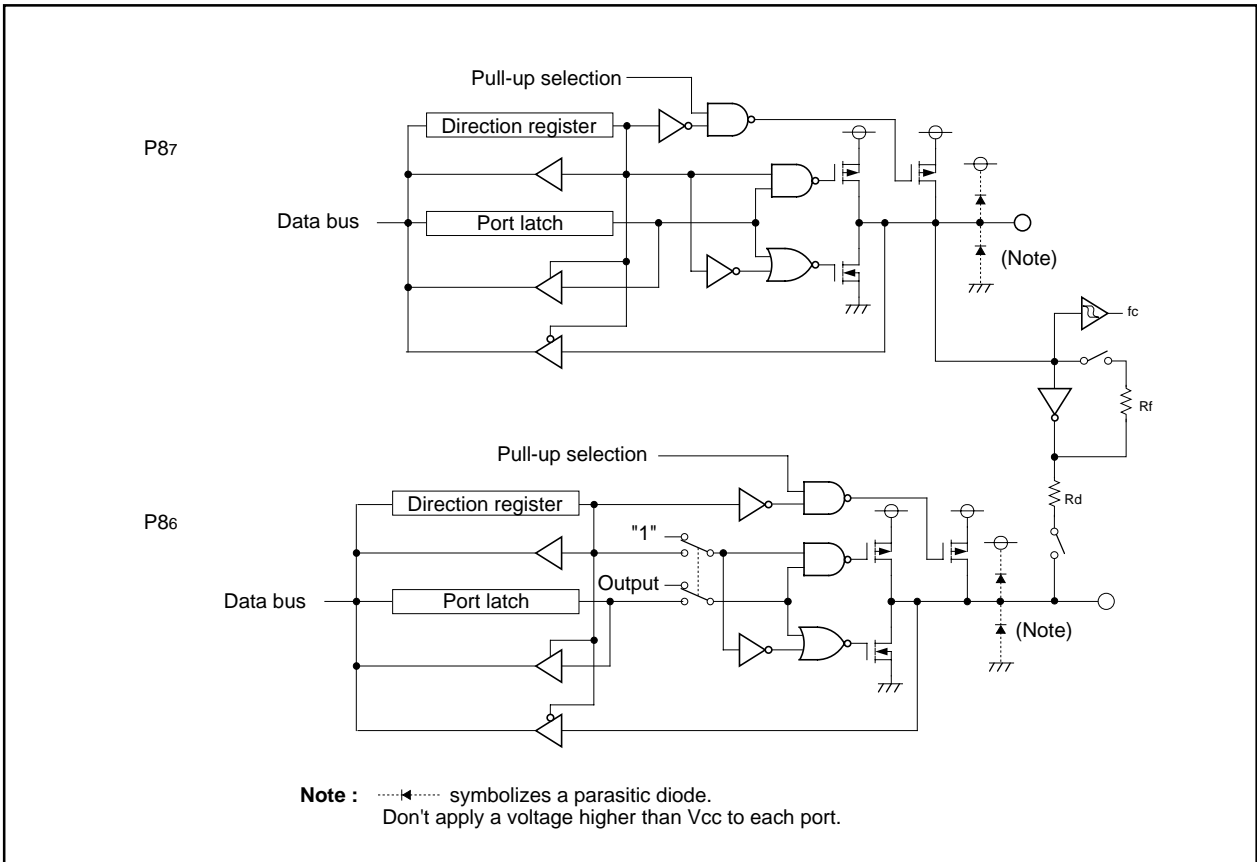


Figure 2.17.4 Programmable I/O ports (4)

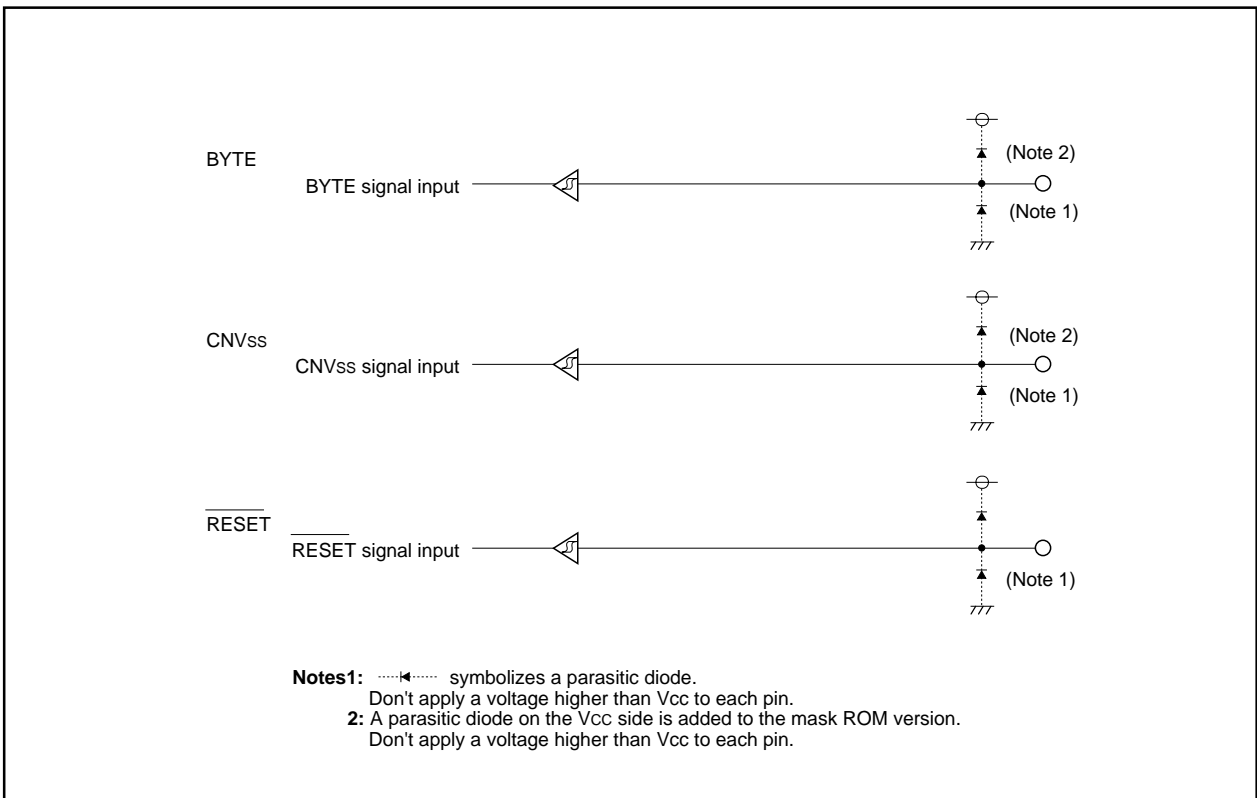


Figure 2.17.5 I/O pins

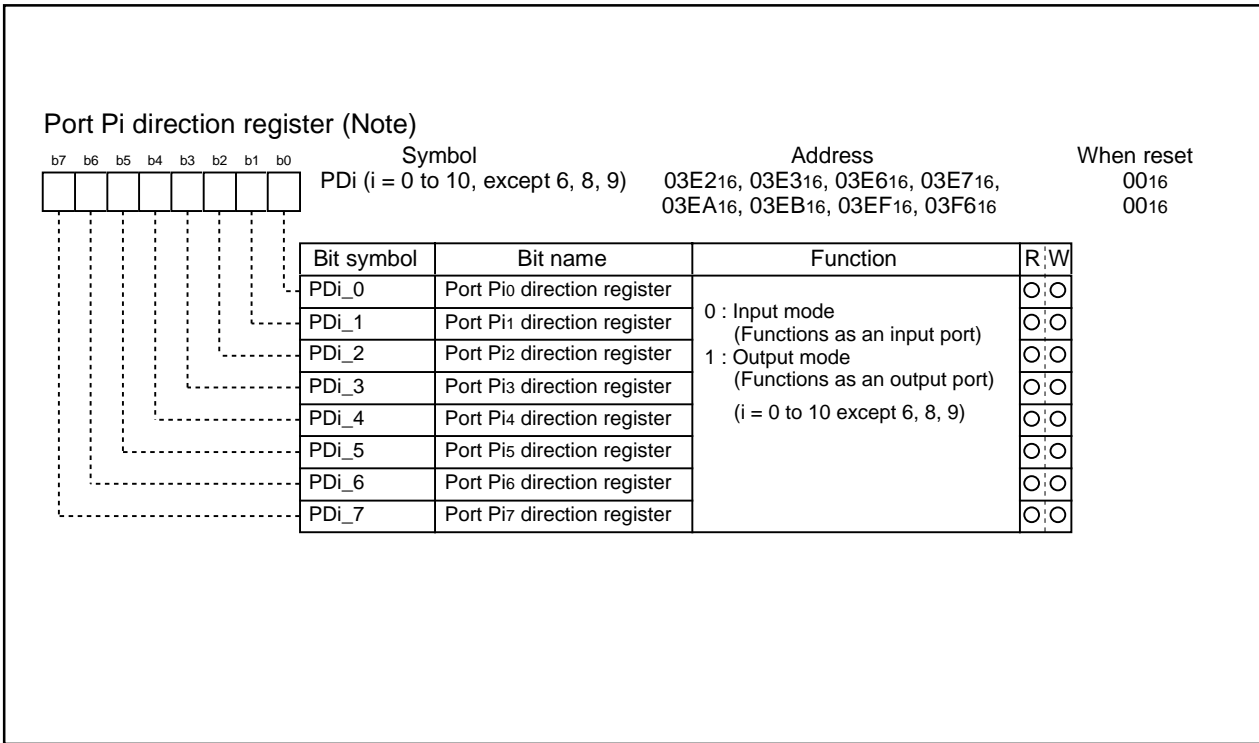


Figure 2.17.6 Port Pi direction register (i = 0 to 10, except 6, 8, 9)

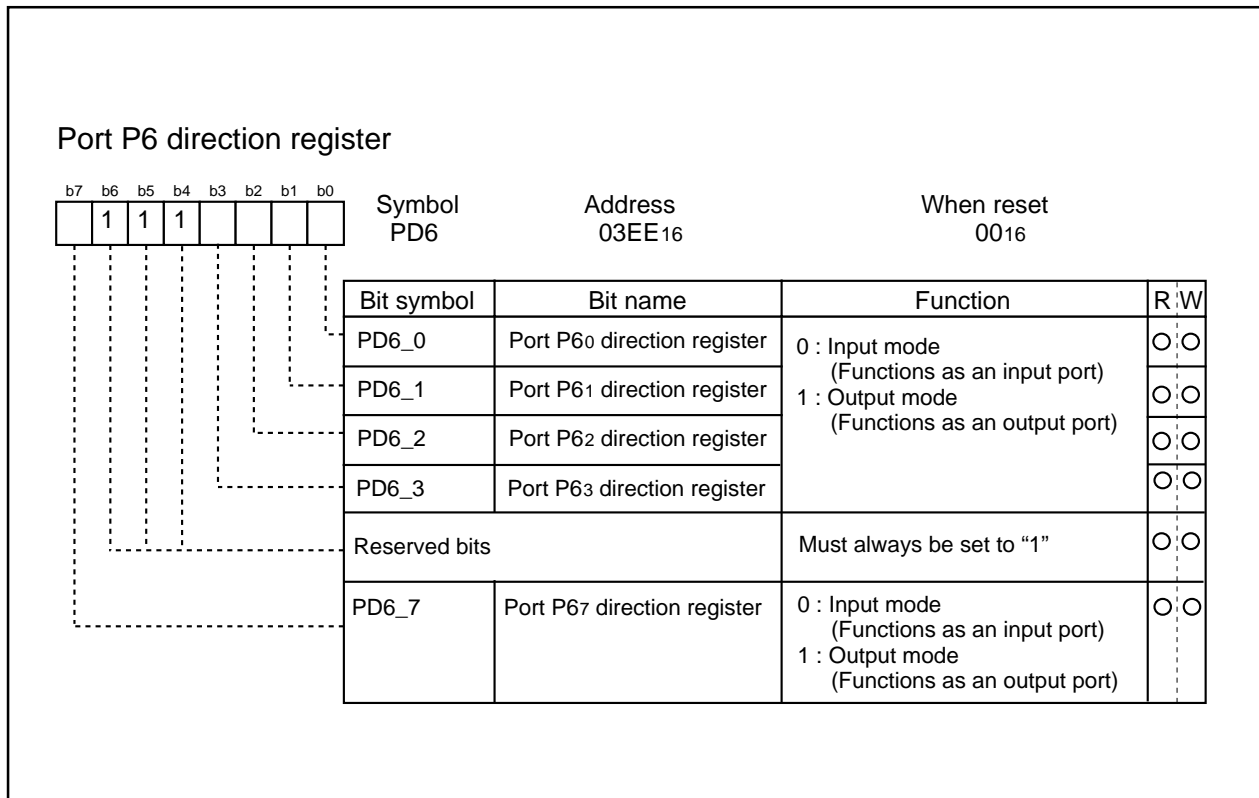


Figure 2.17.7 Port P6 direction register

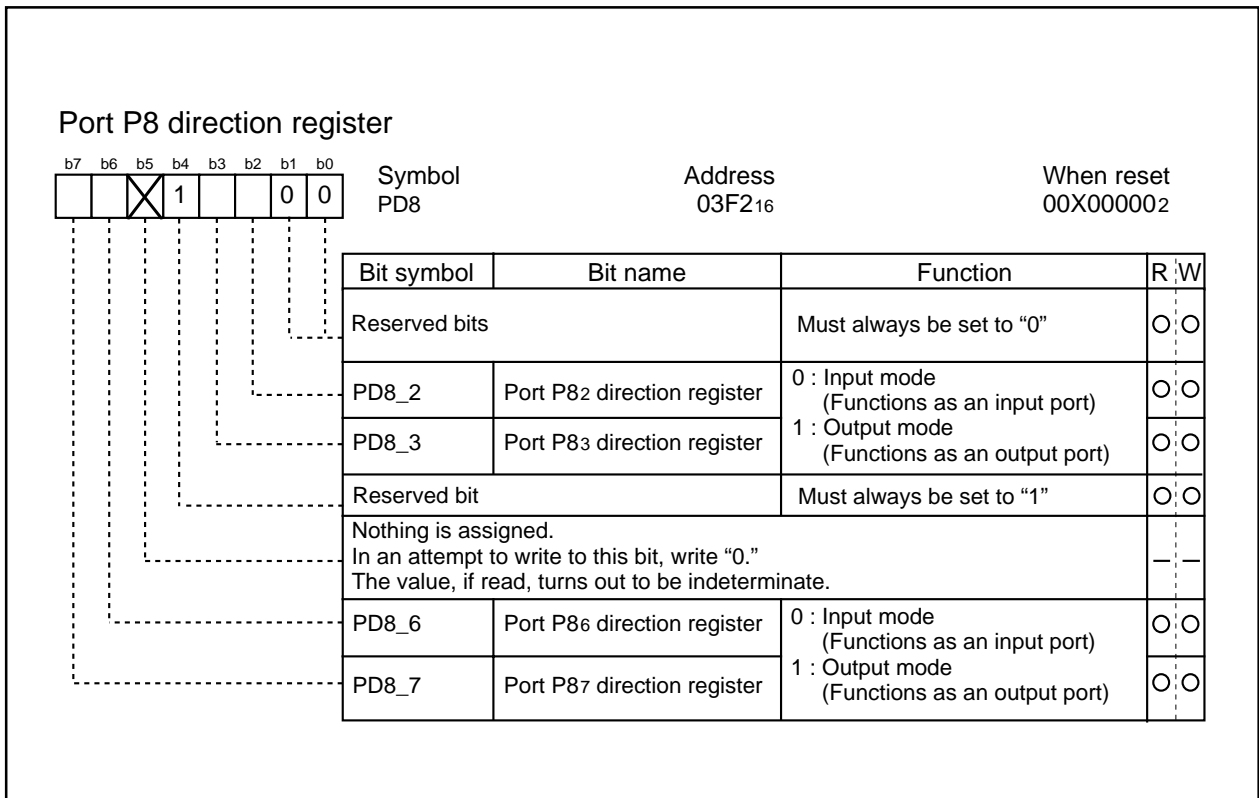


Figure 2.17.8 Port P8 direction register

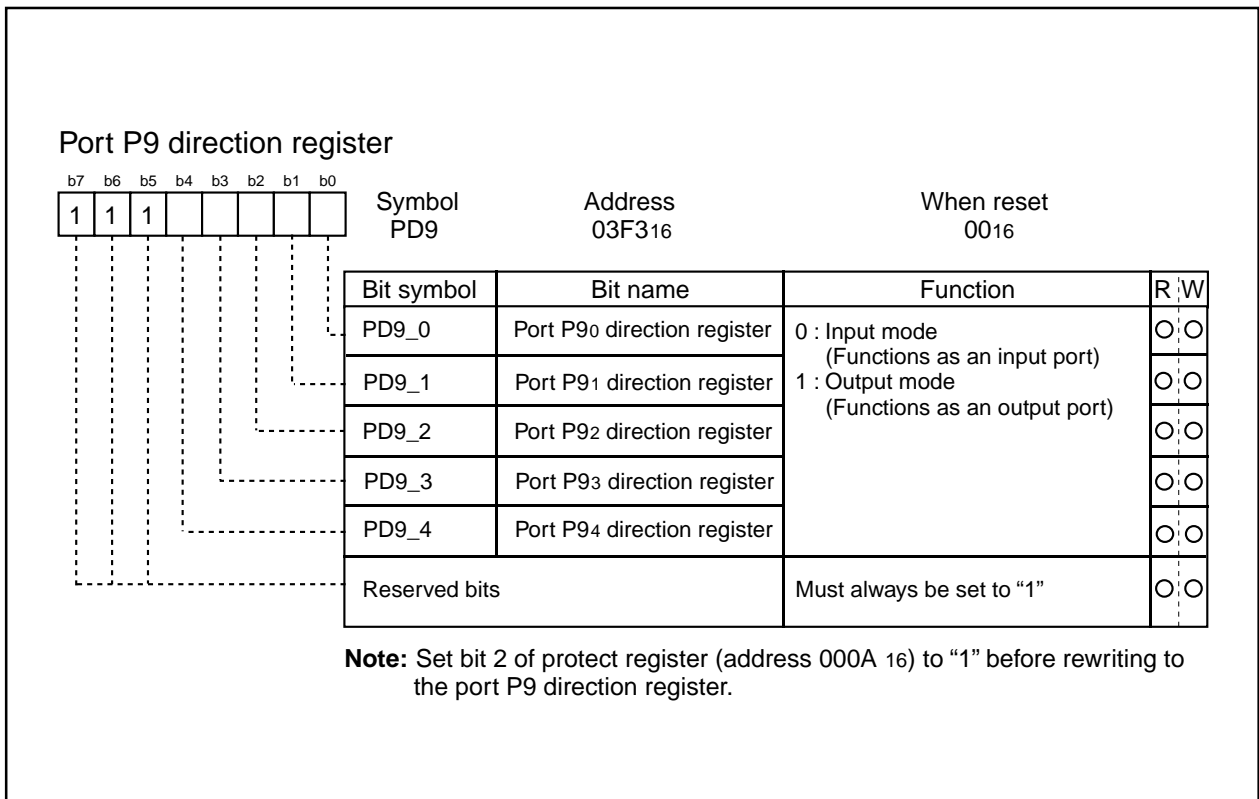


Figure 2.17.9 Port P9 direction register

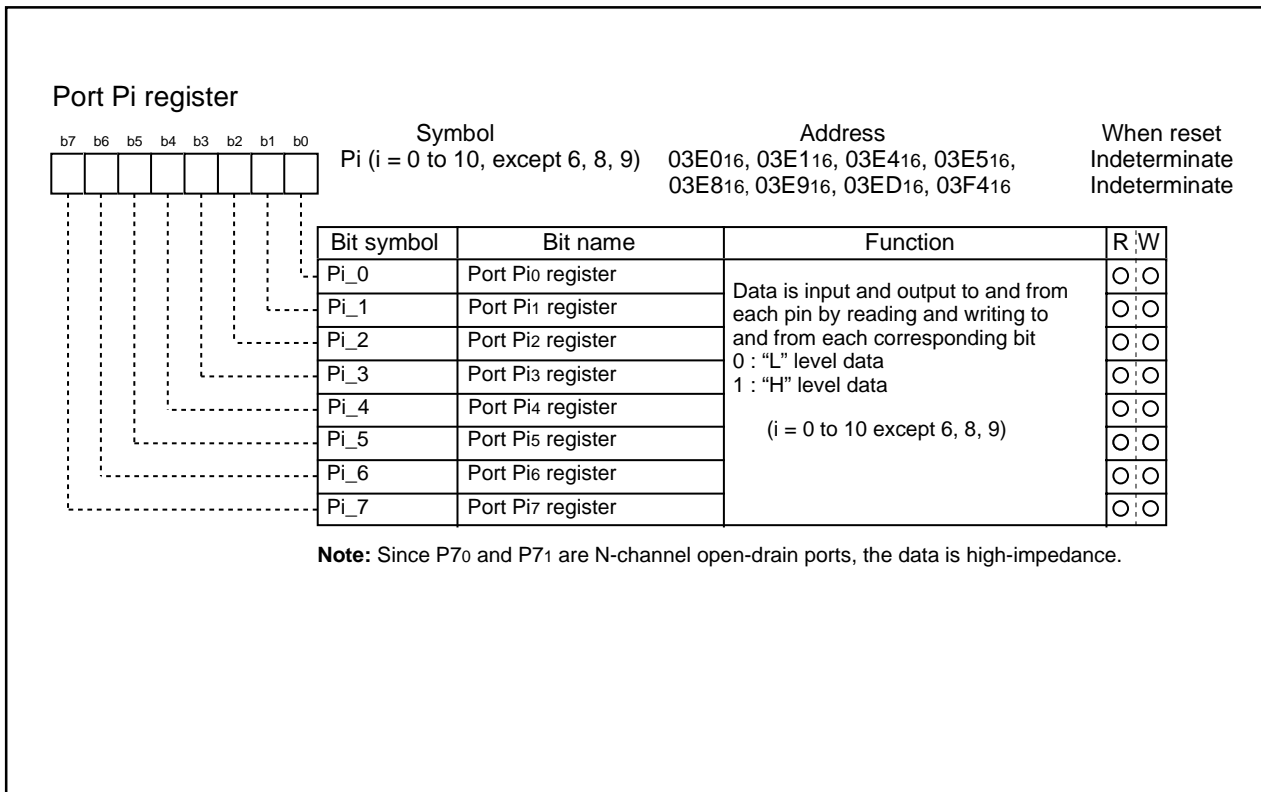


Figure 2.17.10 Port Pi register (i = 0 to 10, except 6, 8, 9)

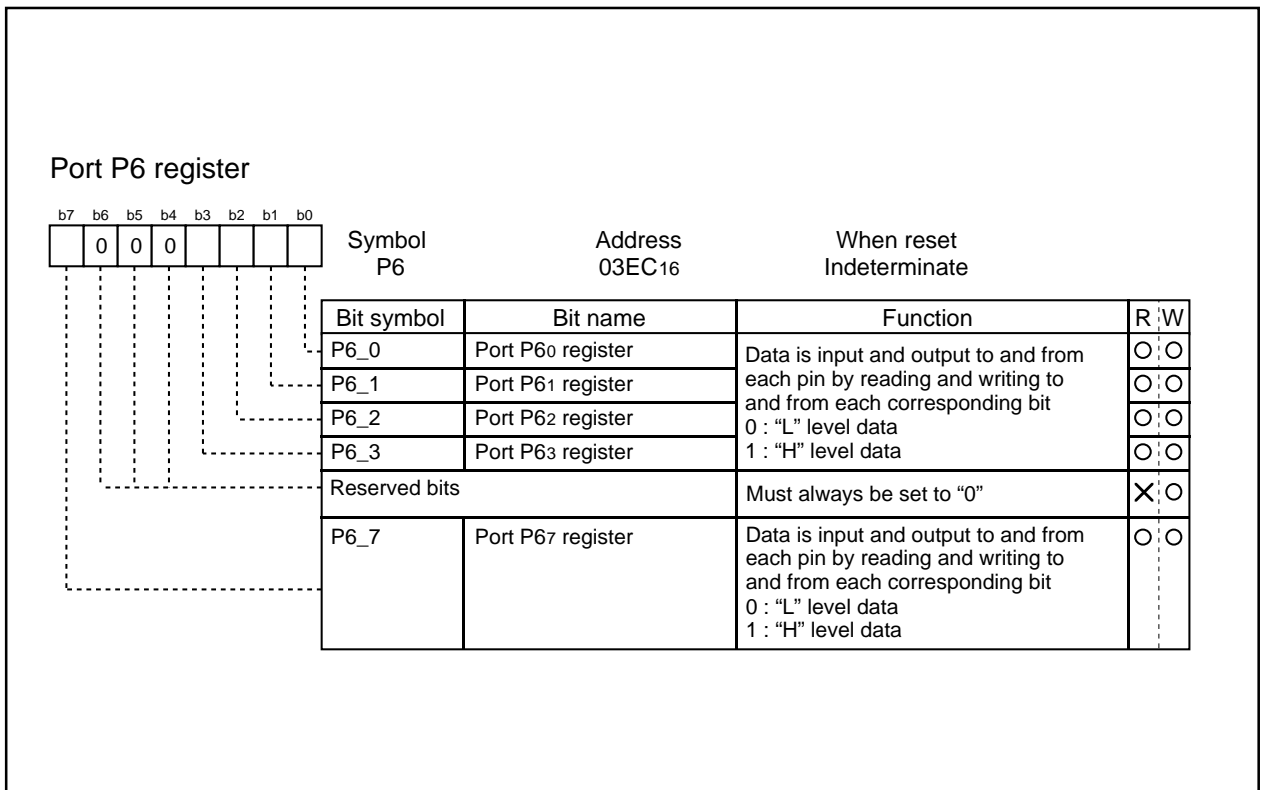


Figure 2.17.11 Port P6 register

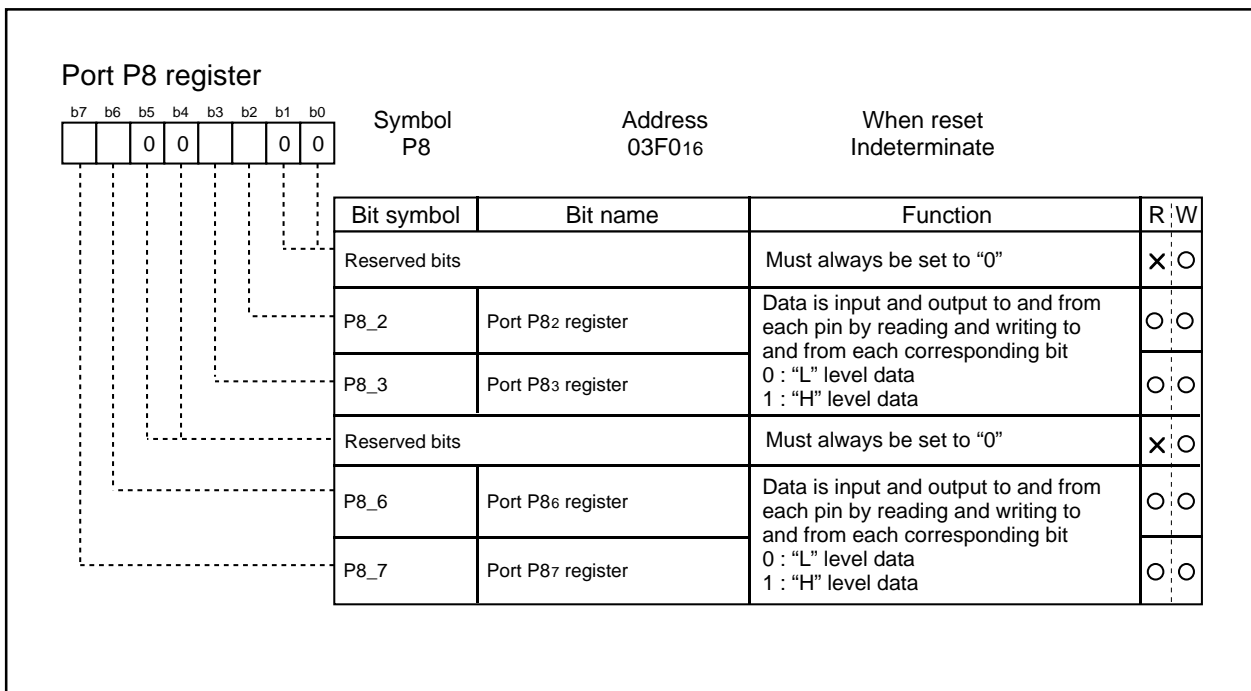


Figure 2.17.12 Port P8 register

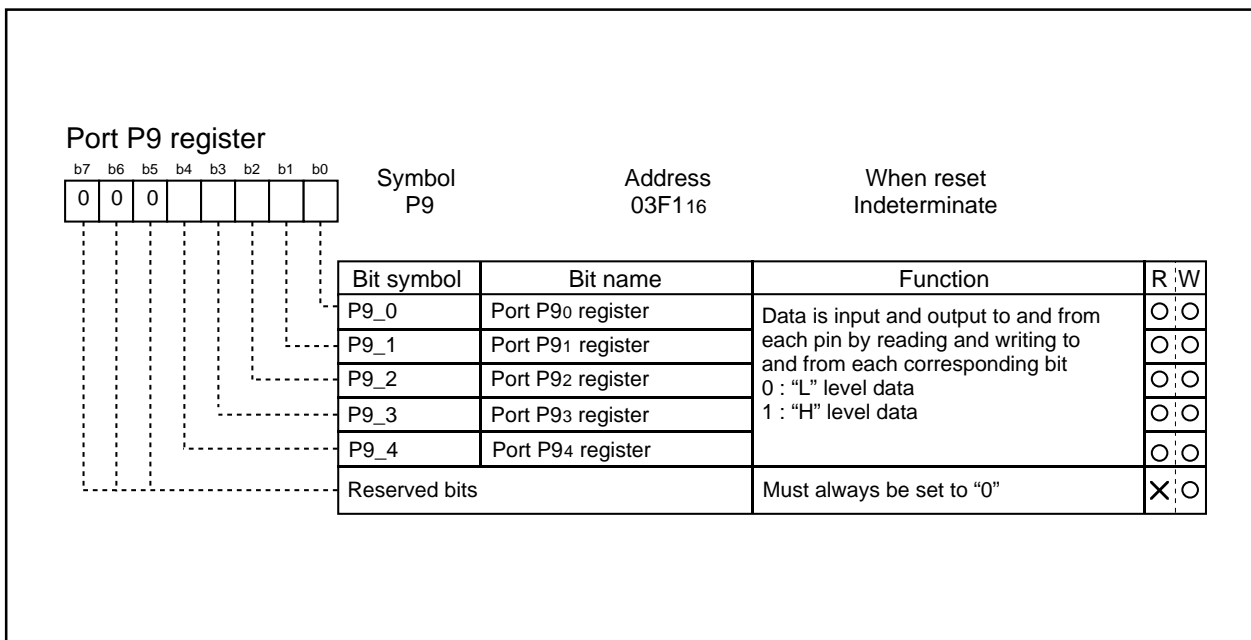


Figure 2.17.13 Port P9 register 0

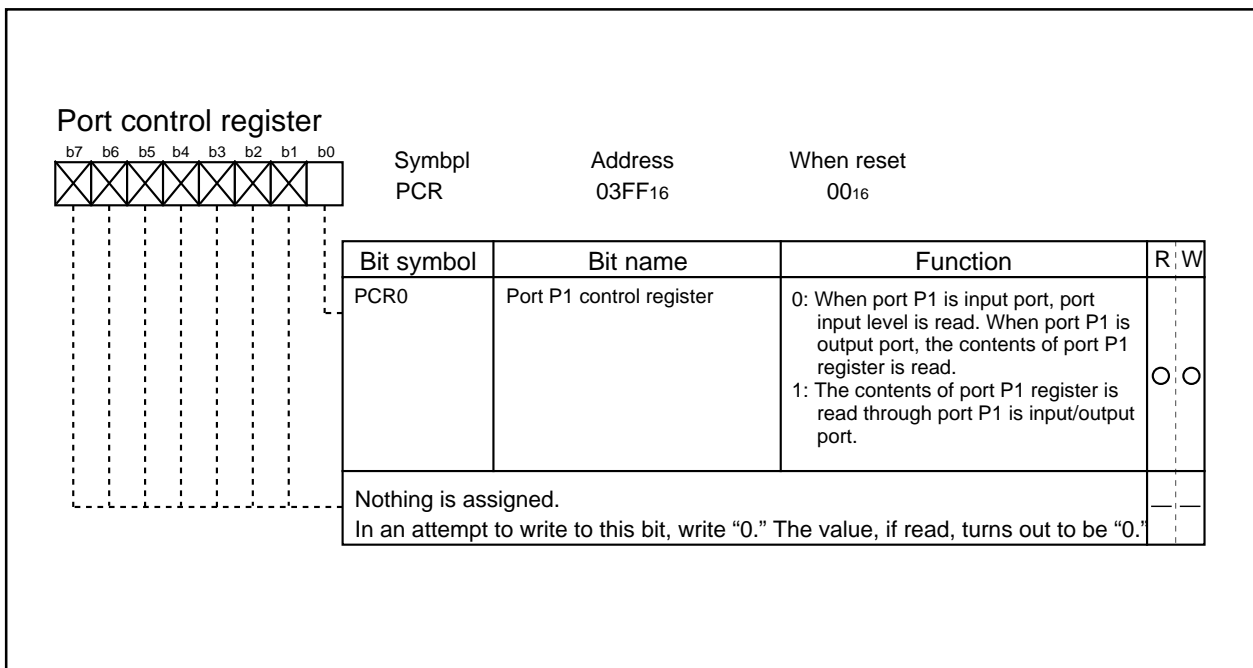


Figure 2.17.14 Port control register

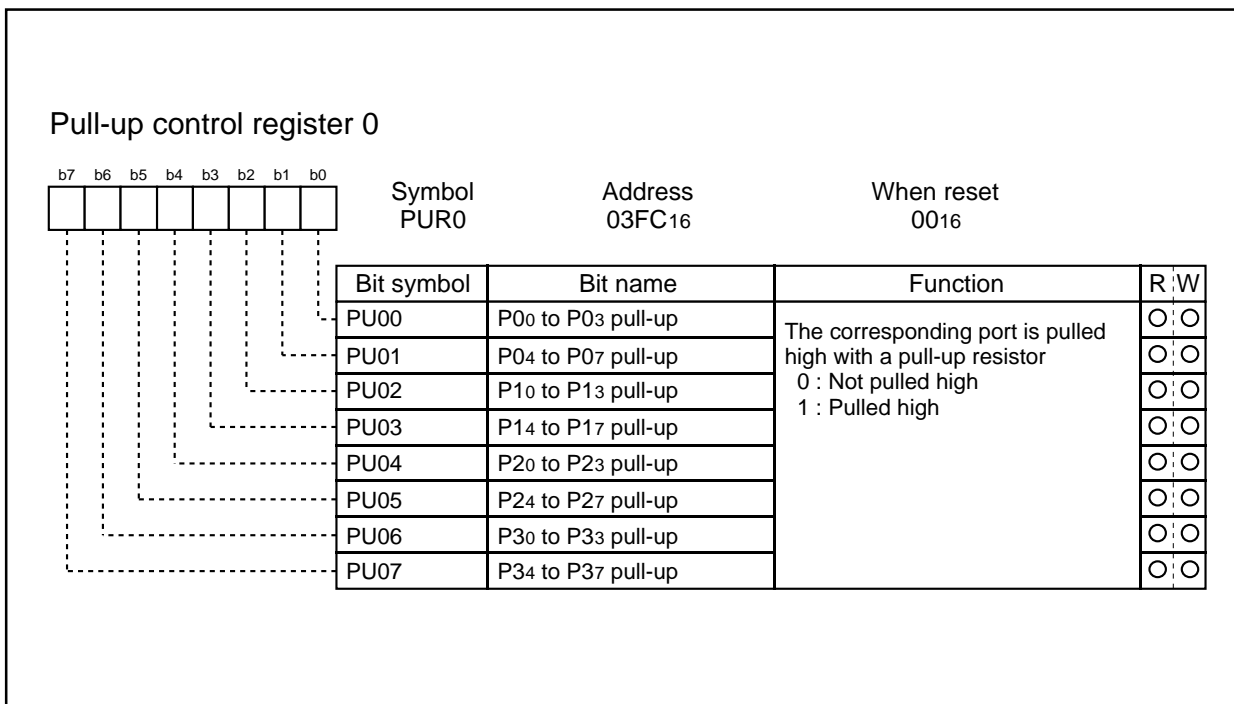


Figure 2.17.15 Pull-up control register 0

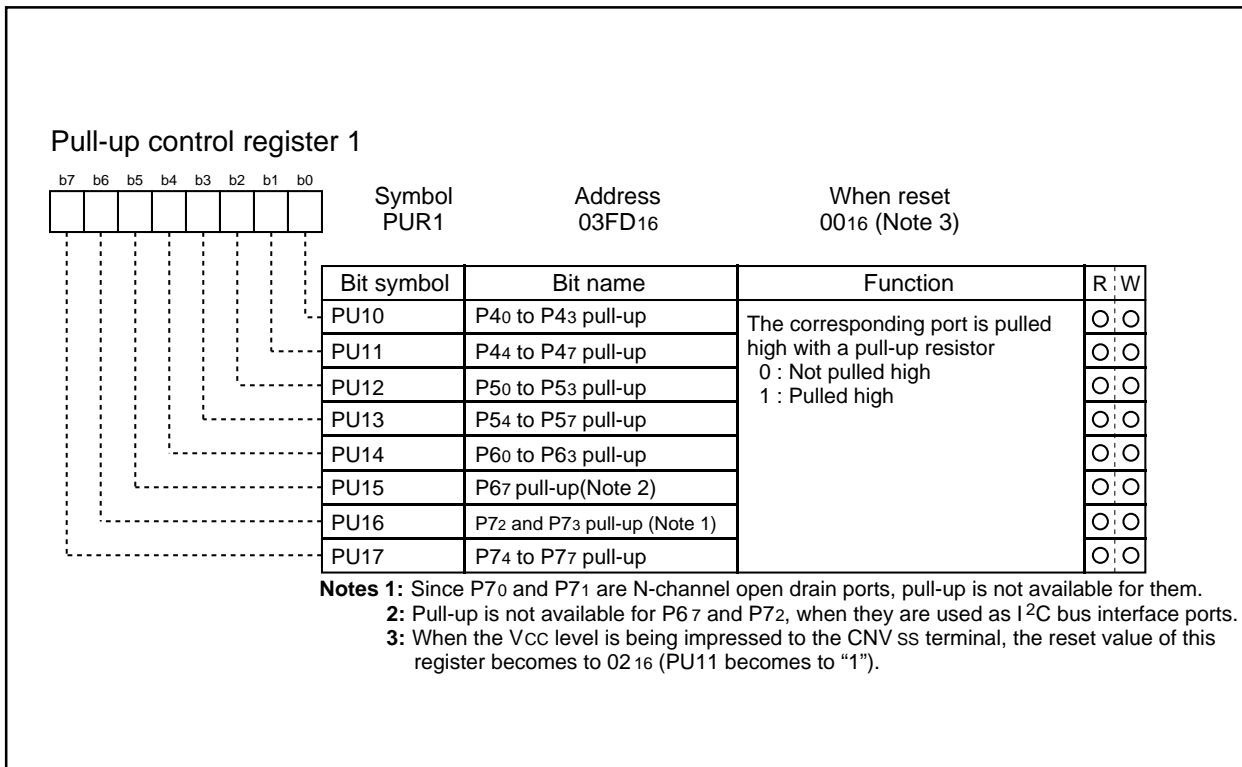


Figure 2.17.16 Pull-up control register 1

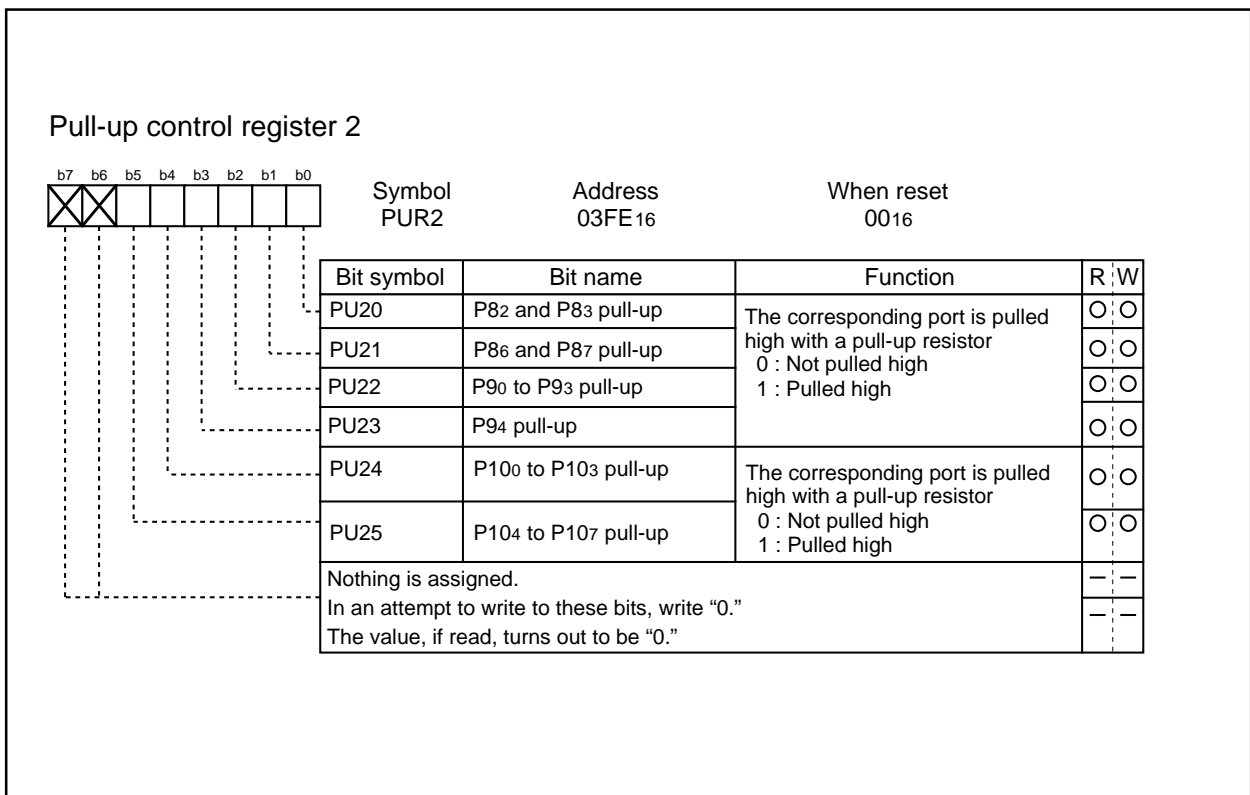


Figure 2.17.17 Pull-up control register 2

Table 2.17.1 Example connection of unused pins in single-chip mode

| Pin name | Connection |
|-----------------|--|
| Ports P0 to P10 | After setting for input mode, connect every pin to Vss or Vcc via a resistor; or after setting for output mode, leave these pins open. |
| XOUT (Note) | Open |
| AVcc | Connect to Vcc |
| BYTE | Connect to Vss |
| CNVss | Connect via resistor to Vss (pull-down) |

Note: With external clock input to XIN pin.

Table 2.17.2 Example connection of unused pins in memory expansion mode and microprocessor mode

| Pin name | Connection |
|--|---|
| Ports P6 to P10 | After setting for input mode, connect every pin to Vss or Vcc via a resistor; or after setting for output mode, leave these pins open. |
| P45/ $\overline{CS1}$ to P47/ $\overline{CS3}$ | 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} , XOUT(Note), BCLK | Open |
| \overline{HOLD} , \overline{RDY} | Connect via resistor to Vcc (pull-up) |
| AVcc | Connect to Vcc |
| CNVss | Connect via resistor to Vss (pull-down) in the memory expansion mode. Connect via resistor to Vcc (pull-up) in the microprocessor mode. |

Note: With external clock input to XIN pin.

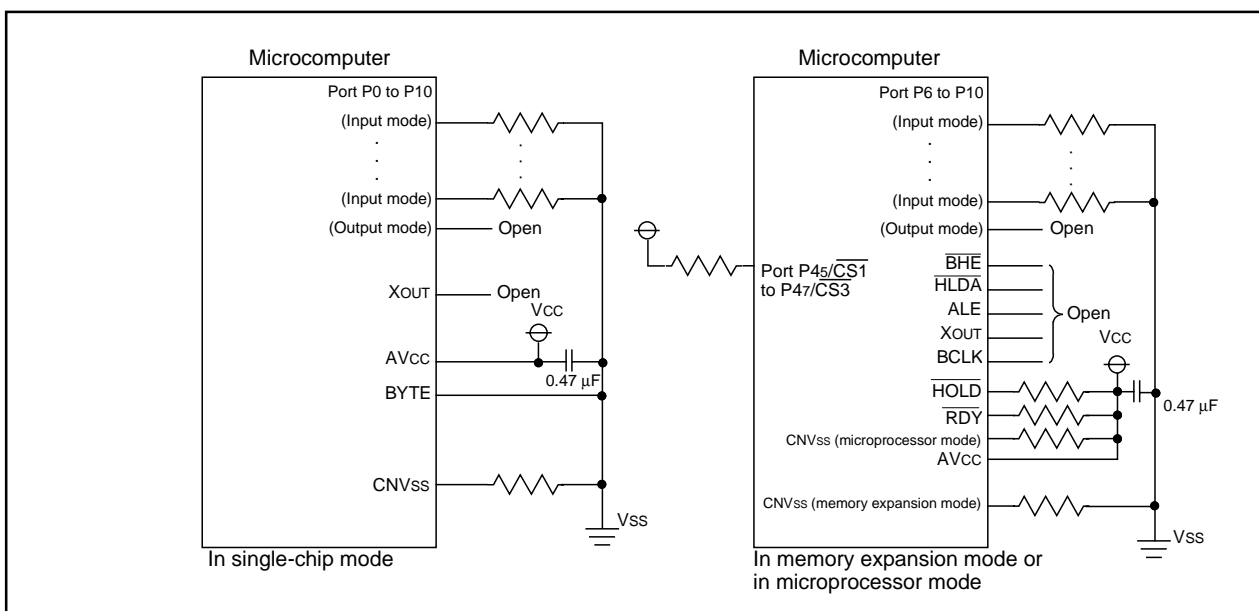


Figure 2.17.18 Example connection of unused pins

3. USAGE PRECAUTION

3.1 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.

3.2 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.

3.3 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.

3.4 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".

3.5 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.

3.6 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.

3.7 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 μ s or longer.
- (2) When changing A-D operation mode, select analog input pin again.
- (3) When using A-D converter in the one-shot mode and in the single sweep mode
After confirming the completion of A-D conversion, read the A-D register (the completion of A-D conversion is determined by A-D interrupt request bit).
- (4) When using A-D converter in the repeat mode and in the repeat sweep mode
Use the main clock without dividing as the internal clock of CPU.
- (5) The A-D conversion in the sweep mode needs the time as follows; (number of sweep pins + 2 pins) \times repeat times \times A-D conversion time for 1 pin.
- (6) When operating OSD or operating data slicer using the HSYNC and VSYNC input, do not use the A-D sweep mode (single sweep mode, repeat sweep mode 0, and repeat sweep mode 1).

3.8 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 perfected 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."

3.9 Interrupts

(1) Reading address 00000₁₆

- 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 00000₁₆ will then be set to "0". Reading address 00000₁₆ 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 00000₁₆ by software.

(2) Setting the stack pointer

- The value of the stack pointer immediately after reset is initialized to 0000₁₆. 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.

(3) External interrupt

- When the polarity of the $\overline{\text{INT}}_0$ and $\overline{\text{INT}}_1$ pins is changed, the interrupt request bit is sometimes set to "1." After changing the polarity, set the interrupt request bit to "0."

(4) 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

3.10 Built-in PROM Version

3.10.1 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.

3.10.2 One Time PROM Version

One Time PROM versions shipped in blank, 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 3.10.1 before use.

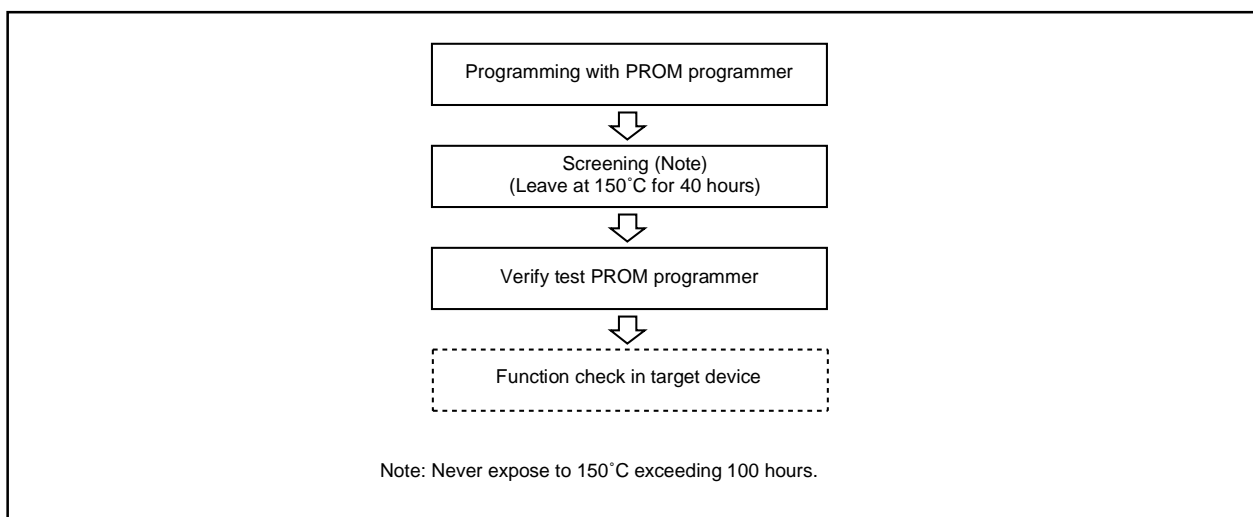


Figure 3.10.1 Programming and test flow for One Time PROM version

4. ITEMS TO BE SUBMITTED WHEN ORDERING MASKED ROM VERSION

Please submit the following when ordering masked ROM products.

- (1) Mask ROM confirmation form
- (2) Mask specification sheet
- (3) ROM data : EPROMs (3 sets)

*: In the case of EPROMs, three 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.

5. ELECTRICAL CHARACTERISTICS

5.1. Absolute Maximum Ratings

Table 5.1.1 Absolute maximum ratings

| Symbol | Parameter | Condition | Rated value | Unit |
|------------------|---|-----------------------|------------------------------|------|
| V _{cc} | Supply voltage | | -0.3 to 6.0 | V |
| AV _{cc} | Analog supply voltage | | -0.3 to 6.0 | V |
| V _i | Input 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 ₃ , P6 ₇ , P7 ₀ to P7 ₇ , P8 ₂ , P8 ₃ , P8 ₆ , P8 ₇ , P9 ₀ to P9 ₄ , P10 ₀ to P10 ₇ , X _{IN} , OSC1, RESET | | -0.3 to V _{cc} +0.3 | V |
| V _i | Input voltage CNV _{ss} , BYTE | | -0.3 to 6.0 (Note) | 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 ₃ , P6 ₇ , P7 ₀ to P7 ₇ , P8 ₂ , P8 ₃ , P8 ₆ , P8 ₇ , P9 ₀ to P9 ₄ , P10 ₀ to P10 ₇ , R, G, B, OUT1, OUT2, OSC2, X _{OUT} | | -0.3 to V _{cc} +0.3 | V |
| P _d | Power dissipation | T _a =25 °C | 500 | mW |
| T _{opr} | Operating ambient temperature | | -10 to 70 | °C |
| T _{stg} | Storage temperature | | -40 to 125 | °C |

Note: When writing to EPROM, only CNV_{ss} is -0.3 to 13(V).

5.2 Recommended Operating Conditions

Table 5.2.1 Recommended operating conditions (referenced to Vcc = 4.5 V to 5.5 V at Ta = – 10 °C to 70 °C unless otherwise specified)

| Symbol | Parameter | Standard | | | Unit |
|------------------------|---|--------------------------|--------|---------|------|
| | | Min | Typ. | Max. | |
| Vcc | Supply voltage (Note 3) | 4.5 | 5.0 | 5.5 | V |
| AVcc | Analog supply voltage (Note 3) | | Vcc | | V |
| Vss | Supply | | 0 | | V |
| V _{IH} | HIGH input voltage P31 to P37, P40 to P47, P50 to P57, P60 to P63, P67, P70 to P77, P82, P83, P86, P87, P90 to P94, P100 to P107. HLF, V _{HOLD} , CV _{IN} , TVSETB, X _{IN} , OSC1, RESET, CNV _{SS} , BYTE | 0.8Vcc | | Vcc | V |
| V _{IH} | HIGH input voltage P00 to P07, P10 to P17, P20 to P27, P30 (during single-chip mode) | 0.8Vcc | | Vcc | V |
| V _{IH} | HIGH input voltage P00 to P07, P10 to P17, P20 to P27, P30 (data input function during memory expansion and microprocessor modes) | 0.5Vcc | | Vcc | V |
| V _{IL} | LOW input voltage P31 to P37, P40 to P47, P50 to P57, P60 to P63, P67, P70 to P77, P82, P83, P86, P87, P90 to P94, P100 to P107. X _{IN} , OSC1, RESET, CNV _{SS} , BYTE | 0 | | 0.2Vcc | V |
| V _{IL} | LOW input voltage P00 to P07, P10 to P17, P20 to P27, P30 (during single-chip mode) | 0 | | 0.2Vcc | V |
| V _{IL} | LOW input voltage P00 to P07, P10 to P17, P20 to P27, P30 (data input function during memory expansion and microprocessor modes) | 0 | | 0.16Vcc | V |
| I _{OH (peak)} | HIGH peak output current P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P63, P72 to P77, P82, P83, P86, P87, P90 to P94, P100 to P107, R, G, B, OUT1, OUT2 | | | –10.0 | mA |
| I _{OH (avg)} | HIGH average output current P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P63, P67, P72 to P77, P82, P83, P86, P87, P90 to P94, P100 to P107, R, G, B, OUT1, OUT2 | | | –5.0 | mA |
| I _{OL (peak)} | LOW peak output current P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P63, P73 to P77, P82, P83, P86, P87, P90 to P94, P100 to P107, R, G, B, OUT1, OUT2 | | | 10.0 | mA |
| I _{OL (avg)} | LOW average output current P67, P70 to P72, P93, P94 | | | 6.0 | mA |
| I _{OL (avg)} | LOW average output current P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P63, P67, P72 to P77, P82, P83, P86, P87, P90 to P94, P100 to P107, R, G, B, OUT1, OUT2 | | | 5.0 | mA |
| f (X _{IN}) | Main clock input oscillation frequency | | | 10 | MHz |
| f (X _{ClN}) | Sub-clock oscillation frequency | | 32.768 | 50 | kHz |
| f _{osc} | Oscillation frequency (for OSD) OSC1 | LC oscillating mode | 11.0 | 27.0 | MHz |
| | | Ceramic oscillating mode | 15.0 | 27.0 | |
| f _{CVIN} | Input frequency Horizontal sync. signal of video signal | 15.262 | 15.743 | 16.206 | kHz |
| V _i | Input amplitude video signal CV _{IN} | 1.5 | 2.0 | 2.5 | V |

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

2: The total I_{OL (peak)} for ports P0, P1, P2, P86, P87, P9, and P10 must be 80 mA max. The total I_{OH (peak)} for ports P0, P1, P2, P86, P87, P9, and P10 must be 80 mA max. The total I_{OL (peak)} for ports P3, P4, P5, P6, P7, P82 and P83 must be 80 mA max. The total I_{OH (peak)} for ports P3, P4, P5, P6, P72 to P77, P82 and P83 must be 80 mA max.

3: Connect 0.1 mF or more capacitor externally between the power source pins Vcc–Vss and AVcc–Vss so as to reduce power source noise. Also connect 0.1 mF or more capacitor externally between the power source pins

Vcc–CNVss

5.3 Electrical Characteristics

Table 5.3.1 Electrical characteristics (referenced to VCC = 5 V, VSS = 0 V at Ta = 25 °C, f(XIN) = 10 MHz unless otherwise specified)

| Symbol | Parameter | | Measuring condition | Standard | | | Unit | |
|----------------------------------|--|--|---|--|------|-------|------|----|
| | | | | Min. | Typ. | Max. | | |
| VOH | HIGH output voltage | P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P63, P67, P72 to P77, P82, P83, P86, P87, P90 to P94, P100 to P107, R, G, B, OUT1, OUT2 | I _{OH} = -5 mA | 3.0 | | | V | |
| VOH | HIGH output voltage | P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57 | I _{OH} = -200 μA | 4.7 | | | V | |
| VOH | HIGH output voltage | X _{OUT} | HIGH POWER | I _{OH} = -1 mA | 3.0 | | V | |
| | | | LOW POWER | I _{OH} = -0.5 mA | 3.0 | | | |
| VOL | LOW output voltage | P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P63, P73 to P77, P82, P83, P86, P87, P90 to P94, P100 to P107, R, G, B, OUT1, OUT2 | I _{OL} = 5 mA | | | 2.0 | V | |
| VOL | LOW output voltage | P67, P70 to P72, P93, P94 | I _{OL} = 6.0 mA | | | 0.6 | V | |
| VOL | LOW output voltage | P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P53 | I _{OL} = 200 μA | | | 0.45 | V | |
| VOL | LOW output voltage | X _{OUT} | HIGHPOWER | I _{OL} = 1 mA | | 2.0 | V | |
| | | | LOWPOWER | I _{OL} = 0.5 mA | | 2.0 | | |
| V _{T+} -V _{T-} | Hysteresis | HOLD, RDY, TB0IN to TB2IN, INT0, INT1, CTS0, CTS2, CLK0, CLK2, SCL1 to SCL3, SDA1 to SDA3, HSYNC, VSYNC, HC0, HC1, RxD0 | | 0.2 | | 0.8 | V | |
| V _{T+} -V _{T-} | Hysteresis | RESET | | 0.2 | | 1.8 | V | |
| V _{T+} -V _{T-} | Hysteresis | XIN | | 0.2 | | 0.8 | V | |
| I _{IH} | HIGH input current | P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P63, P67, P70 to P77, P82, P83, P86, P87, P90 to P94, P100 to P107, XIN, RESET, CNVSS, BYTE, OSC1 | V _I = 5 V | | | 5.0 | μA | |
| I _{IL} | LOW input current | P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P63, P67, P70 to P77, P82, P83, P86, P87, P90 to P94, P100 to P107, XIN, RESET, CNVSS, BYTE, OSC1 | V _I = 0 V | | | -5.0 | μA | |
| R _{PULLUP} | Pull-up resistor | P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P63, P67, P72 to P77, P82, P83, P86, P87, P90 to P94, P100 to P107 | V _I = 0 V | 30.0 | 50.0 | 167.0 | kΩ | |
| I _{CC} | Power supply current | In single-chip mode, the output pins are open and other pins are VSS | f(XIN) = 10 MHz Square wave, no division | OSD ON, Data slicer ON | 70 | 90 | mA | |
| | | | | OSD OFF, Data slicer OFF | 30 | 50 | | |
| | | | f(XIN) = 10 MHz Square wave, division by 8 | OSD OFF, Data slicer OFF | 10 | | mA | |
| | | | | f(XCIN) = 32kHz When a WAIT instruction is executed | 10 | | | μA |
| | | | | Ta=25 °C when clock is stopped | | 10 | | |
| Ta = 70 °C when clock is stopped | | 200 | | | | | | |
| R _{BS} | I ² C-BUS • BUS switch connection resistor (between SCL1 and SCL2, SDA1 and SDA2) | | V _{CC} = 4.5 V | | | 130 | Ω | |
| R _{IXIN} | Feedback resistor XIN | | | | 1.0 | | MΩ | |
| R _{IXCIN} | Feedback resistor XCIN | | | | 6.0 | | MΩ | |

5.4 A-D Conversion Characteristics

Table 5.4.1 A-D conversion characteristics (referenced to $V_{CC} = AV_{CC} = 5V$, $V_{SS} = 0V$ at $T_a = 25\text{ }^\circ\text{C}$, $f(X_{IN}) = 10\text{ MHz}$ unless otherwise specified)

| Symbol | Parameter | | Measuring condition | Standard | | | Unit |
|---------|----------------------|--|-------------------------|----------|----------|----------|---------------|
| | | | | Min. | Typ. | Max. | |
| — | Resolution | | $V_{REF} = V_{CC}$ | | | 8 | Bits |
| — | Absolute accuracy | Sample & hold function not available | $V_{REF} = V_{CC} = 5V$ | | | ± 5 | LSB |
| | | Sample & hold function available (8 bit) | $V_{REF} = V_{CC} = 5V$ | | | ± 5 | LSB |
| RLADDER | Ladder resistance | | $V_{REF} = V_{CC}$ | 10 | | 40 | k Ω |
| tCONV | Conversion time | | | 2.8 | | | μs |
| tsAMP | Sampling time | | | 0.3 | | | μs |
| VREF | Reference voltage | | | | V_{CC} | | V |
| VIA | Analog input voltage | | | 0 | | V_{CC} | V |

5.5 D-A Conversion Characteristics

Table 5.5.1 D-A conversion characteristics (referenced to $V_{CC} = 5V$, $V_{SS} = 0V$, at $T_a = 25\text{ }^\circ\text{C}$, $f(X_{IN}) = 10\text{ MHz}$ unless otherwise specified)

| Symbol | Parameter | | Measuring condition | Standard | | | Unit |
|--------|--------------------------------------|--|---------------------|----------|------|------|---------------|
| | | | | Min. | Typ. | Max. | |
| — | Resolution | | | | | 8 | Bits |
| — | Absolute accuracy | | | | | 10 | % |
| tsu | Setup time | | | | | 3 | μs |
| Ro | Output resistance | | | 4 | 10 | 20 | k Ω |
| 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 "0016."
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.

5.6 Analog R, G, B Output Characteristics

Table 5.6.1 Analog R, G, B output characteristics ($V_{CC} = 5V$, $V_{SS} = 0V$, at $T_a = 25\text{ }^\circ\text{C}$, $f(X_{IN}) = 10\text{ MHz}$ unless otherwise specified)

| Symbol | Parameter | | Test conditions | Standard | | Unit |
|--------|------------------|--|--|----------|-----------|------------|
| | | | | Min. | Max. | |
| Ro | Output impedance | | | | 2 | k Ω |
| VOE | Output deviation | | | | ± 0.5 | V |
| TST | Settling time | | load capacity of 10 pF, load resistance of 20 k Ω , 70 % DC level | | 50 | ns |

5.7 Timing Requirements

Table 5.7.1 External clock input (referenced to Vcc = 5 V, Vss = 0 V at Ta = 25 °C unless otherwise specified)

| Symbol | Parameter | Standard | | Unit |
|-------------------|---------------------------------------|----------|------|------|
| | | Min. | Max. | |
| t _c | External clock input cycle time | 100 | | ns |
| t _{w(H)} | External clock input HIGH pulse width | 40 | | ns |
| t _{w(L)} | External clock input LOW pulse width | 40 | | ns |
| t _r | External clock rise time | | 15 | ns |
| t _f | External clock fall time | | 15 | ns |

Table 5.7.2 Memory expansion and microprocessor modes (referenced to Vcc = 5 V, Vss = 0 V at Ta = 25 °C unless otherwise specified)

| 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) | $\overline{\text{RDY}}$ input setup time | 30 | | ns |
| t _{su} (HOLD-BCLK) | $\overline{\text{HOLD}}$ input setup time | 40 | | ns |
| t _h (RD-DB) | Data input hold time | 0 | | ns |
| t _h (BCLK-RDY) | $\overline{\text{RDY}}$ input hold time | 0 | | ns |
| t _h (BCLK-HOLD) | $\overline{\text{HOLD}}$ input hold time | 0 | | ns |
| t _d (BCLK-HLDA) | $\overline{\text{HLDA}}$ output delay time | | 40 | ns |

Note: Calculated according to the BCLK frequency as follows:

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

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

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

Table 5.7.3 Timer B input (counter input in event counter mode)(referenced to $V_{CC} = 5\text{ V}$, $V_{SS} = 0\text{ V}$ at $T_a = 25\text{ }^\circ\text{C}$ unless otherwise specified)

| 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 5.7.4 Timer B input (pulse period measurement mode)(referenced to $V_{CC} = 5\text{ V}$, $V_{SS} = 0\text{ V}$ at $T_a = 25\text{ }^\circ\text{C}$ unless otherwise specified)

| 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 5.7.5 Timer B input (pulse width measurement mode)(referenced to $V_{CC} = 5\text{ V}$, $V_{SS} = 0\text{ V}$ at $T_a = 25\text{ }^\circ\text{C}$ unless otherwise specified)

| 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 5.7.6 Serial I/O (referenced to $V_{CC} = 5\text{ V}$, $V_{SS} = 0\text{ V}$ at $T_a = 25\text{ }^\circ\text{C}$ unless otherwise specified)

| 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 5.7.7 External interrupt $\overline{\text{INTi}}$ inputs (referenced to $V_{CC} = 5\text{ V}$, $V_{SS} = 0\text{ V}$ at $T_a = 25\text{ }^\circ\text{C}$ unless otherwise specified)

| Symbol | Parameter | Standard | | Unit |
|--------------------------------|---|----------|------|------|
| | | Min. | Max. | |
| $t_{w(\overline{\text{INH}})}$ | $\overline{\text{INTi}}$ input HIGH pulse width | 250 | | ns |
| $t_{w(\overline{\text{INL}})}$ | $\overline{\text{INTi}}$ input LOW pulse width | 250 | | ns |

5.8 Switching Characteristics

Table 5.8.1 Memory expansion mode and microprocessor mode (no wait) (referenced to Vcc = 5 V, Vss = 0 V at Ta = 25 °C, CM15 = "1" unless otherwise specified)

| Symbol | Parameter | Measuring condition | Standard | | Unit |
|---------------------------|--|---------------------|----------|------|------|
| | | | Min. | Max. | |
| t _d (BCLK-AD) | Address output delay time | Figure 5.9.1 | | 35 | 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 | | | 35 | ns |
| t _h (BCLK-CS) | Chip select output hold time (BCLK standard) | | 4 | | ns |
| t _d (BCLK-ALE) | ALE signal output delay time | | | 35 | ns |
| t _h (BCLK-ALE) | ALE signal output hold time | | - 4 | | ns |
| t _d (BCLK-RD) | RD signal output delay time | | | 35 | ns |
| t _h (BCLK-RD) | RD signal output hold time | | 0 | | ns |
| t _d (BCLK-WR) | WR signal output delay time | | | 35 | 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 1) | | ns |
| t _h (WR-DB) | Data output hold time (WR standard)(Note 2) | | 0 | | ns |

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

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

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

Hold time of data bus is expressed in

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

by a circuit of the right figure.

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

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

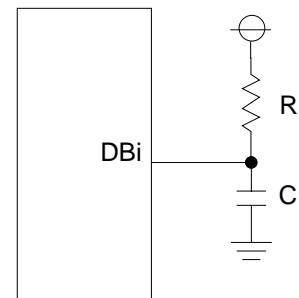


Table 5.8.2 Memory expansion mode and microprocessor mode**(with wait, accessing external memory)****(referenced to Vcc = 5 V, Vss = 0 V at Ta = 25 °C, CM15 = "1" unless otherwise specified)**

| Symbol | Parameter | Measuring condition | Standard | | Unit |
|---------------------------|--|---------------------|----------|------|------|
| | | | Min. | Max. | |
| t _d (BCLK-AD) | Address output delay time | Figure 5.9.1 | | 35 | 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 | | | 35 | ns |
| t _h (BCLK-CS) | Chip select output hold time (BCLK standard) | | 4 | | ns |
| t _d (BCLK-ALE) | ALE signal output delay time | | | 35 | ns |
| t _h (BCLK-ALE) | ALE signal output hold time | | -4 | | ns |
| t _d (BCLK-RD) | RD signal output delay time | | | 35 | ns |
| t _h (BCLK-RD) | RD signal output hold time | | 0 | | ns |
| t _d (BCLK-WR) | WR signal output delay time | | | 35 | 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 1) | | ns |
| t _h (WR-DB) | Data output hold time (WR standard)(Note 2) | | 0 | | ns |

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

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

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

Hold time of data bus is expressed in

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

by a circuit of the right figure.

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

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

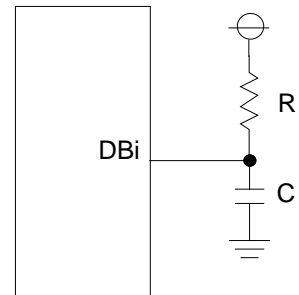


Table 5.8.3 Memory expansion mode and microprocessor mode
(with wait, accessing external memory, multiplex bus area selected)
(referenced to Vcc = 5 V, Vss = 0 V at Ta = 25 °C, CM15 = "1" unless otherwise specified)

| Symbol | Parameter | Measuring condition | Standard | | Unit |
|---------------------------|---|---------------------|----------|------|------|
| | | | Min. | Max. | |
| t _d (BCLK-AD) | Address output delay time | Figure 5.9.1 | | 35 | 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 | | | 35 | 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 | | | 35 | ns |
| t _h (BCLK-RD) | RD signal output hold time | | 0 | | ns |
| t _d (BCLK-WR) | WR signal output delay time | | | 35 | 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) | | | 35 | 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) | | 30 | | 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:

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

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

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

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

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

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

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

5.9 Measurement Circuit

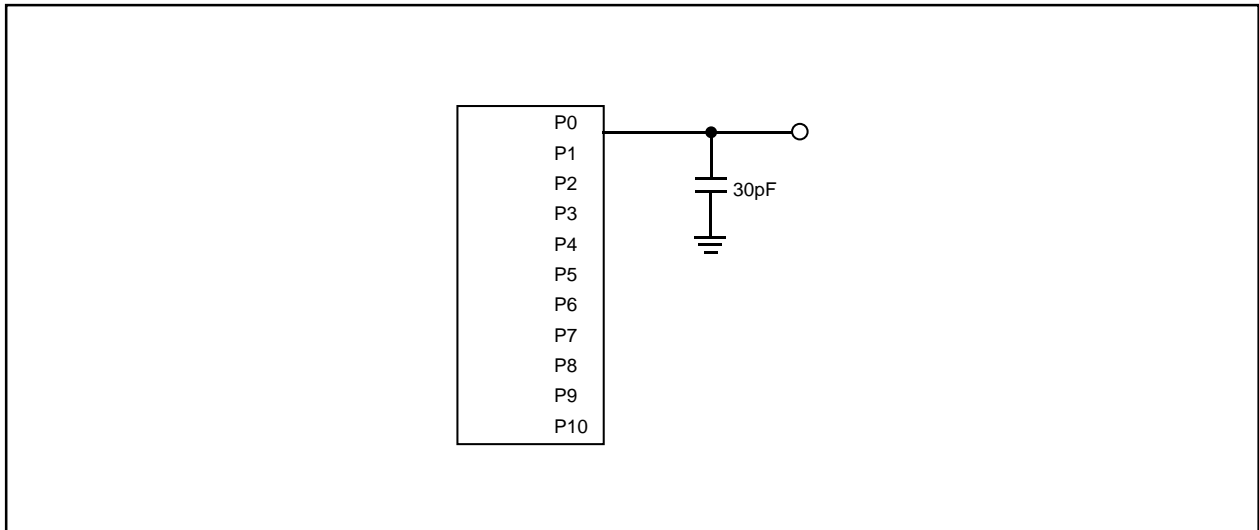


Figure 5.9.1 Port P0 to P10 measurement circuit

5.10 Timing Diagram

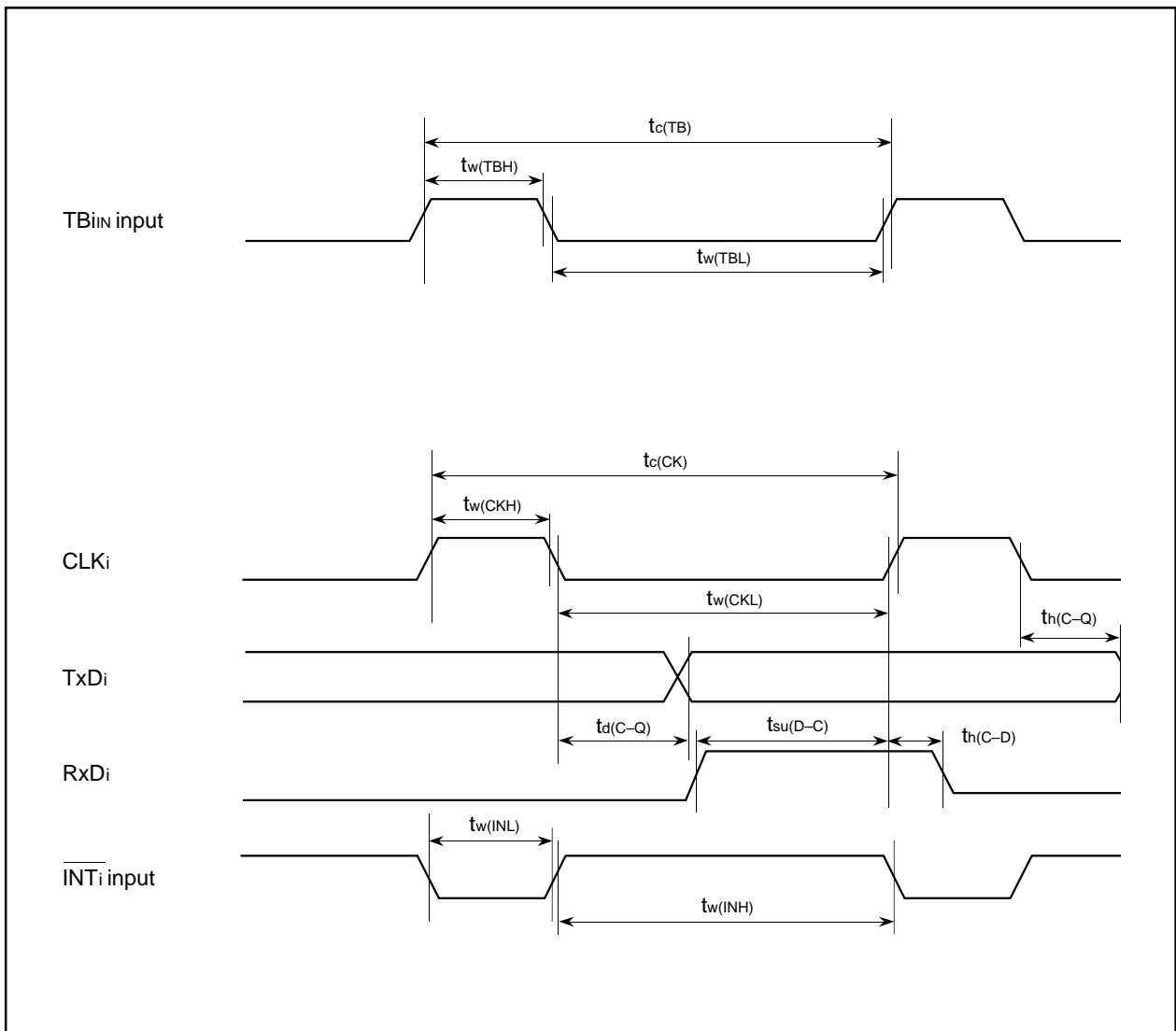
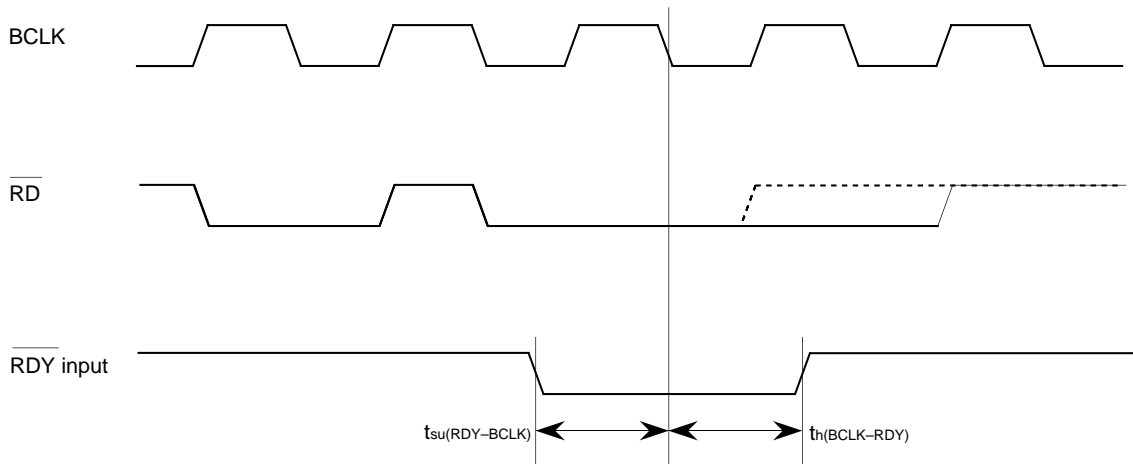
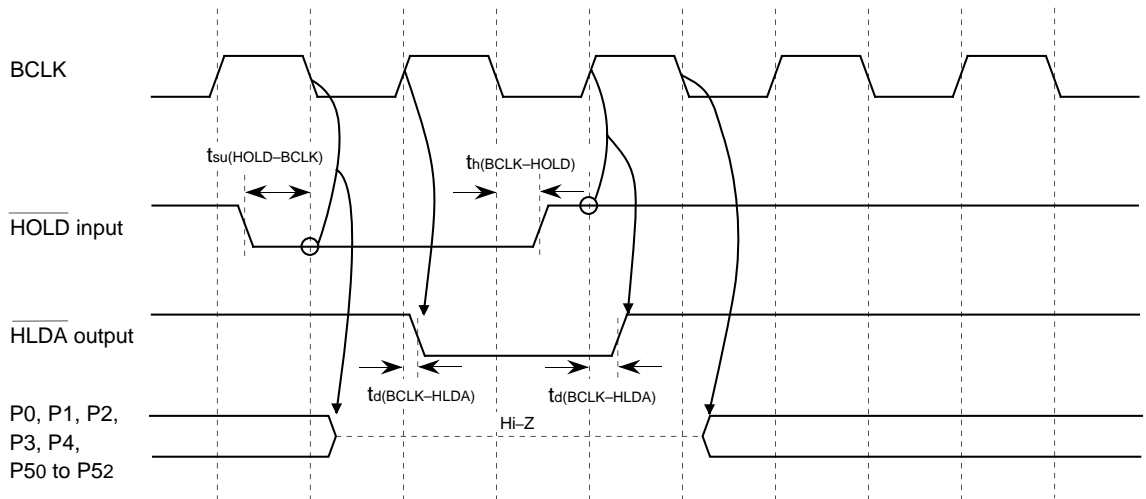


Figure 5.10.1 Timing diagram

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

- $V_{CC}=5V$
- Input timing voltage : Determined with $V_{IL}=1.0V$, $V_{IH}=4.0V$
- Output timing voltage : Determined with $V_{OL}=2.5V$, $V_{OH}=2.5V$

Figure 5.10.2 Timing diagram in memory expansion mode and microprocessor mode (1)

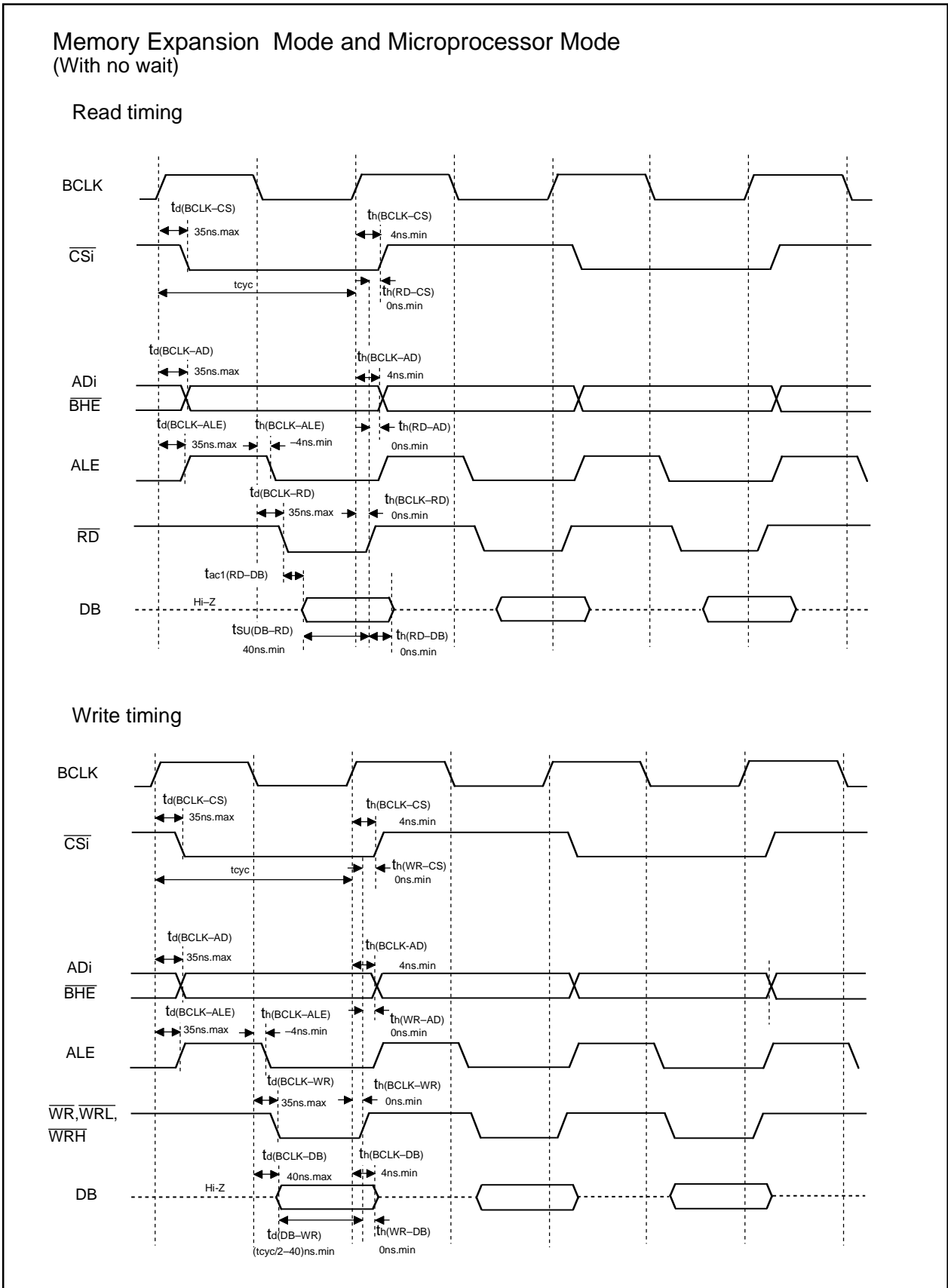
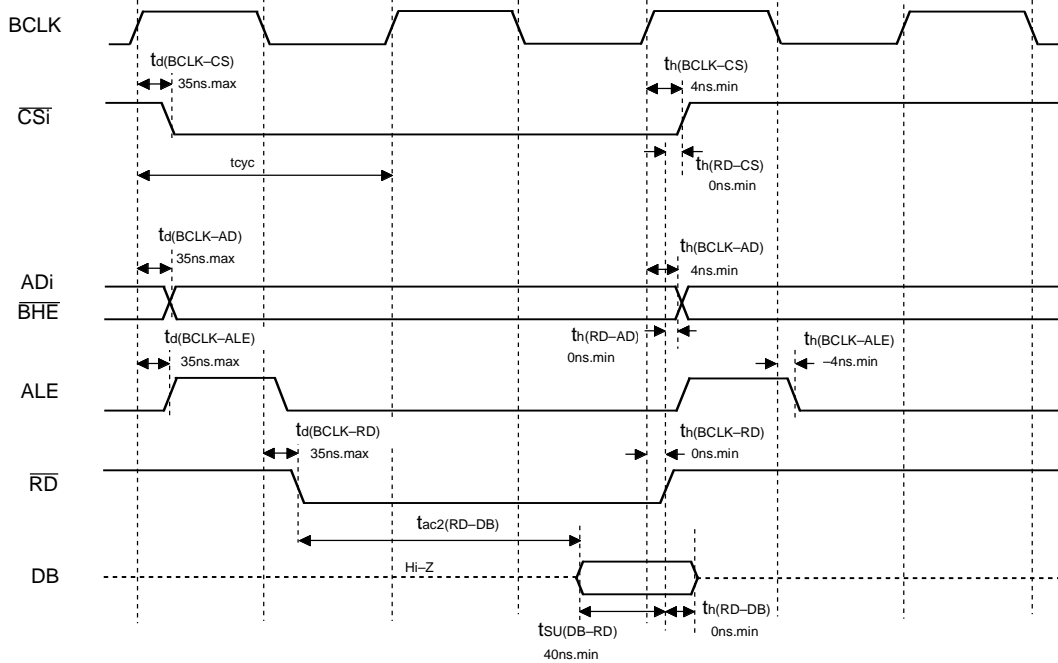


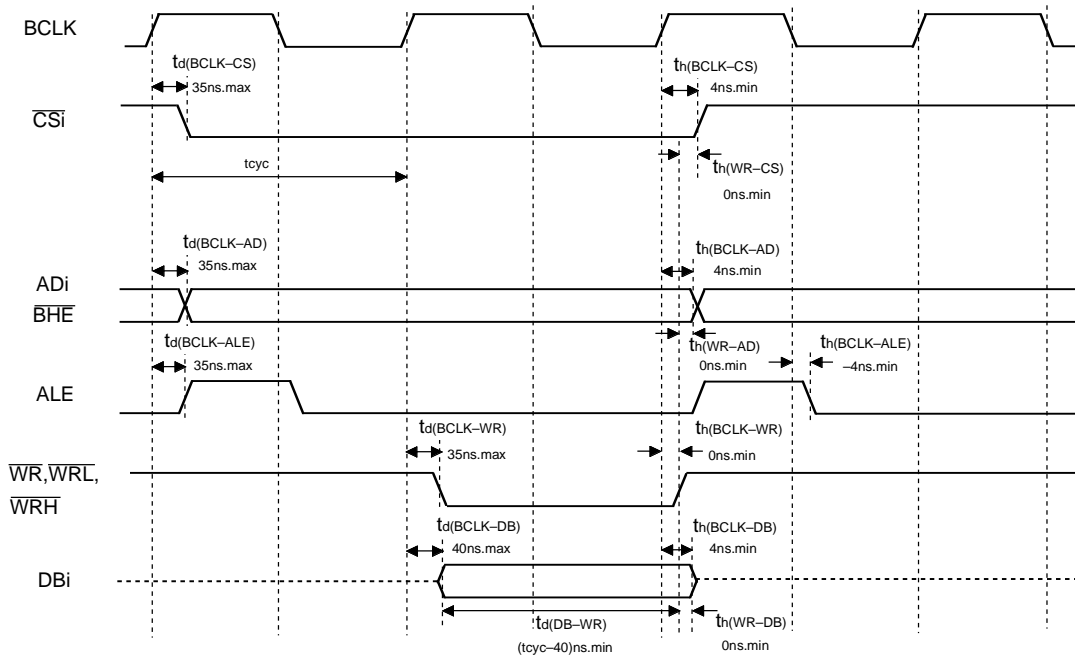
Figure 5.10.3 Timing diagram in memory expansion mode and microprocessor mode (2)

Memory Expansion Mode and Microprocessor Mode
(When accessing external memory area with wait)

Read timing



Write timing



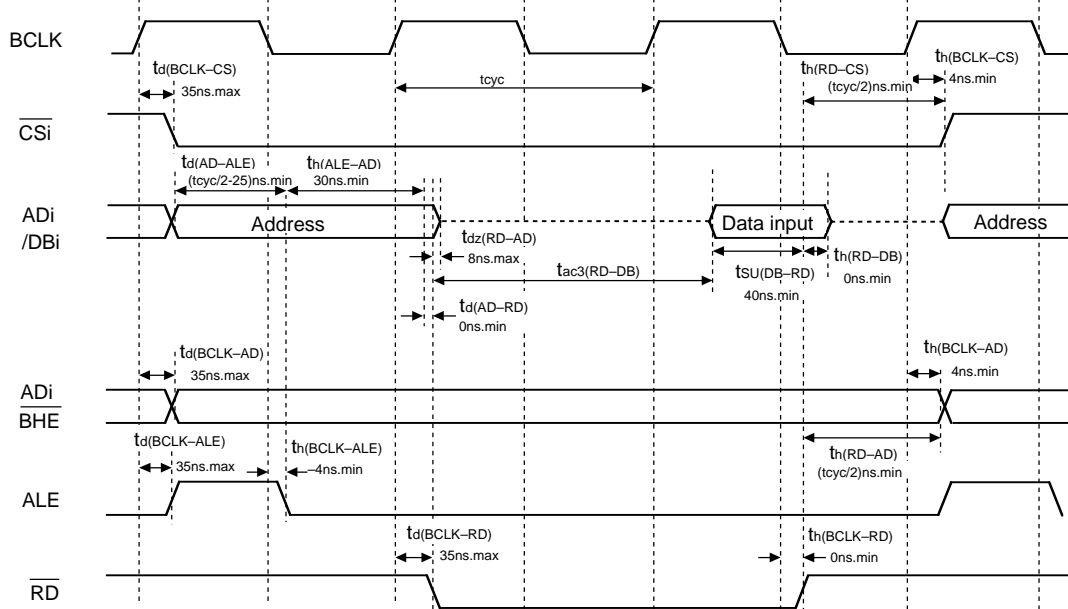
Measuring conditions

- VCC=5V
- Input timing voltage : Determined with: $V_{IL}=0.8V$, $V_{IH}=2.5V$
- Output timing voltage : Determined with: $V_{OL}=0.8V$, $V_{OH}=2.0V$

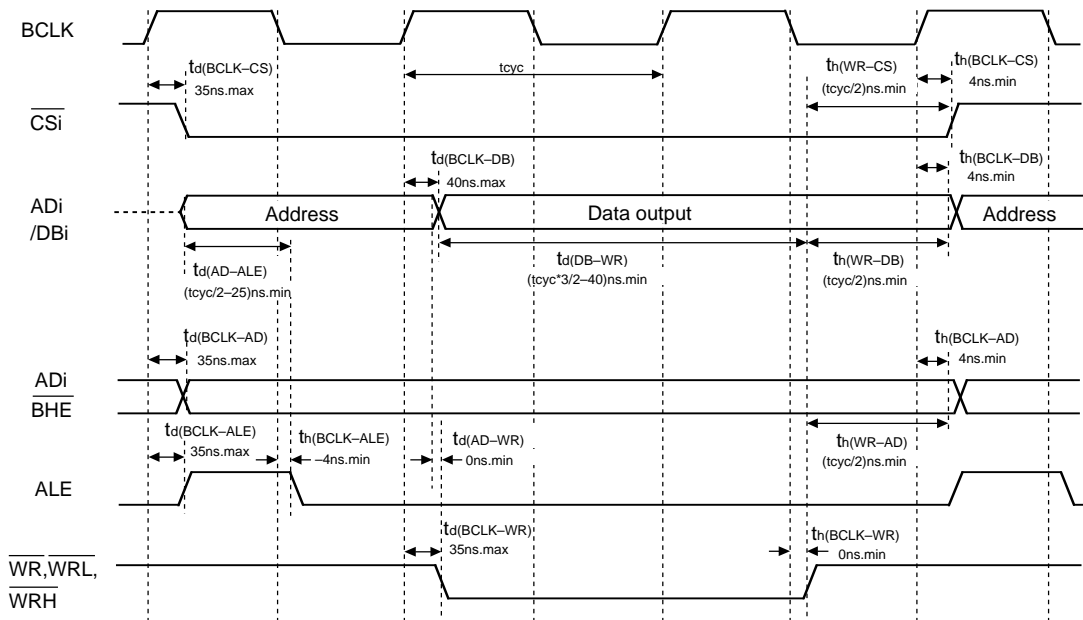
Figure 5.10.4 Timing diagram in memory expansion mode and microprocessor mode (3)

Memory Expansion Mode and Microprocessor Mode
 (When accessing external memory area with wait, and select multiplexed bus)

Read timing



Write timing

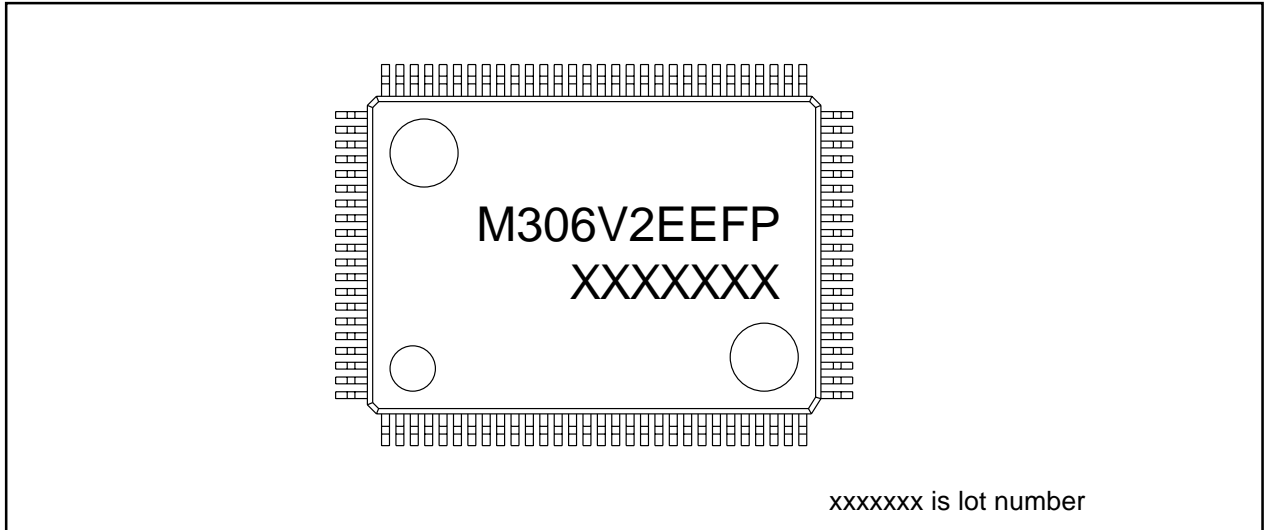


Measuring conditions

- $V_{\text{CC}}=5\text{V}$
- Input timing voltage : Determined with $V_{\text{IL}}=0.8\text{V}$, $V_{\text{IH}}=2.5\text{V}$
- Output timing voltage : Determined with $V_{\text{OL}}=0.8\text{V}$, $V_{\text{OH}}=2.0\text{V}$

Figure 5.10.5 Timing diagram in memory expansion mode and microprocessor mode (4)

6. ONE TIME PROM VERSION M306V2EEFP MARKING

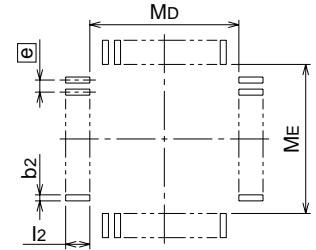
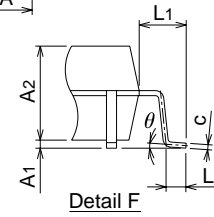
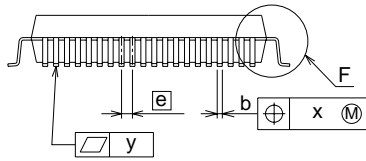
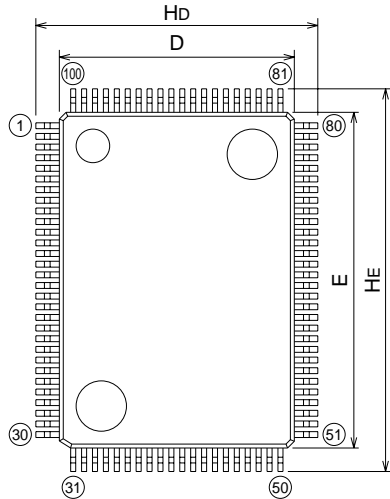


7. PACKAGE OUTLINE

100P6S-A (MMP)

Plastic 100pin 14X20mm body QFP

| | | | |
|--------------------|------------|-----------|---------------|
| EIAJ Package Code | JEDEC Code | Weight(g) | Lead Material |
| QFP100-P-1420-0.65 | - | 1.58 | Alloy 42 |



Recommended Mount Pad

| Symbol | Dimension in Millimeters | | |
|----------|--------------------------|------|------|
| | Min | Nom | Max |
| A | - | - | 3.05 |
| A1 | 0 | 0.1 | 0.2 |
| A2 | - | 2.8 | - |
| b | 0.25 | 0.3 | 0.4 |
| c | 0.13 | 0.15 | 0.2 |
| D | 13.8 | 14.0 | 14.2 |
| E | 19.8 | 20.0 | 20.2 |
| e | - | 0.65 | - |
| HD | 16.5 | 16.8 | 17.1 |
| HE | 22.5 | 22.8 | 23.1 |
| L | 0.4 | 0.6 | 0.8 |
| L1 | - | 1.4 | - |
| x | - | - | 0.13 |
| y | - | - | 0.1 |
| θ | 0° | - | 10° |
| b2 | - | 0.35 | - |
| l2 | 1.3 | - | - |
| MD | - | 14.6 | - |
| ME | - | 20.6 | - |

Structure of Register

Refer to the figure below as for each register.

<Example>

Processor mode register 1 (Note)

Values immediately after reset release (Note 1)

When reset
00000X002

Bit attributes (Note 2)

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

Symbol PM1 Address 000516

| Bit symbol | Bit name | Function | R | W |
|------------|---|--|---|---|
| | Reserved bit | Must always be set to "0" | ○ | ○ |
| | Reserved bit | Must always be set to "1" | ○ | ○ |
| | Nothing is assigned. In an attempt to write to this bit, write "0." The value, if read, turns out to be indeterminate. | | — | — |
| | Reserved bits | Must always be set to "0" | — | ○ |
| PM17 | Wait bit | 0 : No wait state 1 : Wait state inserted | ○ | ○ |

Note: As bit 1 of this register becomes "0" at reset, must always be set to "1" after reset release. Set bit 1 of the protect register (address 000A16) to "1" when writing new values to this register.

X : Bit in which nothing is assigned

Notes 1: Values immediately after reset release

- 0 "0" after reset release
- 1 "1" after reset release
- ? Indeterminate after reset release
- X Bit in which nothing is assigned

2: Bit attributes.....The attributes of control register bits are classified into 3 types : read-only, write-only and read and write. In the figure, these attributes are represented as follows :

R.....Read

-Read enabled
- X.....Read disabled
-Bit in which nothing is assigned (The read value is indeterminate unless otherwise mentioned.)

W.....Write

-Write enabled
- X.....Write disabled
-Bit in which nothing is assigned

-----Register Index-----

| | |
|--|------------------------------|
| [A] | |
| A-D conversion interrupt control register (ADIC) | 54 |
| Address match interrupt enable register (AIER) | 64 |
| Address match interrupt register i (RMADi) .. | 64 |
| A-D register i (ADi) | 152 |
| A-D control register 2 (ADCON2) | 152 |
| A-D control register 1 (ADCON1) | 151, 154, 156, 158, 160, 162 |
| A-D control register 0 (ADCON0) | 151, 154, 156, 158, 160, 161 |

| | |
|--|-----|
| [B] | |
| Block control register i (BCi) | 182 |
| Bottom border control register (BB) | 226 |
| Bus collision detection interrupt control register (BCNIC) | 54 |

| | |
|--|--------|
| [C] | |
| Caption data register i (CDi) | 12 |
| Caption position register (CPS) | 172 |
| Chip select control register (CSR) | 30 |
| Clock control register (CS) | 190 |
| Clock prescaler reset flag (CPSRF) | 84, 92 |
| Clock run-in detect register (CRD) | 173 |
| Color palette register i (CRi) | 210 |
| Count start flag (TABSR) | 82, 92 |

| | |
|--|-----|
| [D] | |
| D-A control register (DACON) | 165 |
| D-A register i (DAi) | 165 |
| Data clock position register (DPS) | 174 |
| Data slicer control register 1 (DSC1) | 168 |
| Data slicer control register 2 (DSC2) | 168 |
| Data slicer interrupt control register (DSIC) .. | 54 |
| Data slicer reserved register i (DRi) | 175 |
| DMA0 request cause select register (DM0SL) | 71 |
| DMA1 request cause select register (DM1SL) .. | 72 |

| | |
|---|----|
| DMAi control register (DMiCON) | 72 |
| DMAi interrupt control register (DMiIC) | 54 |
| DMAi destination pointer (DARi) | 73 |
| DMAi transfer counter (TCRi) | 73 |
| DMAi source pointer (SARi) | 73 |

| | |
|---|-----|
| [H] | |
| Horizontal position register (HP) | 187 |
| HSYNC counter register (HC) | 176 |
| HSYNC counter latch | 12 |

| | |
|--|-----|
| [I] | |
| I ² Ci data shift register (IICiS0) | 134 |
| I ² Ci address register (IICiS0D) | 135 |
| I ² Ci status register (IICiS1) | 142 |
| I ² Ci control register (IICiS1D) | 139 |
| I ² Ci clock control register (IICiS2) | 137 |
| I ² Ci port selection register (IICiS2D) | 132 |
| I ² Ci transmit buffer register (IICiS0S) | 134 |
| I/O polarity control register (PC) | 191 |
| Interrupt control reserved register i (REiIC) | 63 |
| Interrupt request cause select register (IFSR) | 63 |
| INTi interrupt control register (INTiIC) | 54 |

| | |
|---|-----|
| [L] | |
| Left border control register (LB) | 227 |

| | |
|---|----|
| [M] | |
| Multi-master I ² C-BUS interface i interrupt control register (IICiIC) | 54 |

| | |
|--|-----|
| [O] | |
| One-shot start flag (ONSF) | 83 |
| OSD control register 1 (OC1) | 181 |
| OSD control register 2 (OC2) | 184 |
| OSD control register 3 (OC3) | 209 |
| OSD control register 4 (OR4) | 193 |
| OSD reserved register i (ORi) | 231 |
| OSDi interrupt control register (OSDiIC) | 54 |

| | |
|--|-----|
| [P] | |
| Peripheral mode register (PM) | 98 |
| Port control register (PUR) | 242 |
| Port P0 to P5, P7, P10 register (P0 to P5, P7, P10) | 240 |
| Port P0 to P5, P7, P10 direction register (PD0 to PD5, PD7, PD10) | 238 |
| Port P6 register (P6) | 240 |
| Port P6 direction register (PD6) | 238 |
| Port P8 register (P8) | 241 |
| Port P9 register (P9) | 241 |
| Port P8 direction register (PD8) | 239 |
| Port P9 direction register (PD9) | 239 |
| Processor mode register 0 (PM0) | 24 |
| Processor mode register 1 (PM1) | 25 |
| Protect register (PRCR) | 46 |
| Pull-up control register 0 (PUR0) | 242 |
| Pull-up control register 1 (PUR1) | 243 |
| Pull-up control register 2 (PUR2) | 243 |

| | |
|--|-----|
| [R] | |
| Raster color register (RSC) | 228 |
| Reserved register i (INVCi) | 97 |
| Right border control register (RB) | 227 |

| | |
|--|-----|
| [S] | |
| SPRITE horizontal position register (HS) | 223 |
| SPRITE OSD control register (SC) | 222 |
| SPRITE vertical position register i (VSi) | 223 |
| System clock control register 0 (CM0) | 40 |
| System clock control register 1 (CM1) | 40 |

| | |
|--|--------------------|
| [T] | |
| Timer Ai interrupt control register (TAiIC) | 54 |
| Timer Bi interrupt control register (TBiIC) | 54 |
| Timer Ai register (TAi) | 82 |
| Timer Bi register (TBi) | 92 |
| Timer Ai mode register (TAiMR) | 81, 85, 87, 88, 89 |
| Timer Bi mode register (TBiMR) | 91, 93, 94, 95 |
| Top border control register (TB) | 226 |

| | |
|---------------------------------------|----|
| Trigger select register (TRGSR) | 84 |
|---------------------------------------|----|

| | |
|--|---------------|
| [U] | |
| UART transmit/receive control register 2 (UCON) | 108 |
| UART0 transmit/receive control register 0 (U0C0) | 105 |
| UART0 transmit/receive control register 1 (U0C1) | 107 |
| UART0 transmit/receive mode register (U0MR) | 111, 118 |
| UART2 special mode register (U2SMR) | 108 |
| UART2 transmit/receive mode register (U2MR) | 104, 111, 118 |
| UART2 transmit/receive control register 0 (U2C0) | 106 |
| UART2 transmit/receive control register 1 (U2C1) | 107 |
| UARTi bit rate generator (UiBRG) | 103 |
| UARTi receive buffer register (UiRB) | 103 |
| UARTi receive interrupt control register (SiRIC) | 54 |
| UARTi transmit buffer register (UiTB) | 103 |
| UARTi transmit interrupt control register (SiTIC) | 54 |
| Up/down flag (UDF) | 83 |

| | |
|--|-----|
| [V] | |
| VSYNC interrupt control register (VSYNCIC) | 54 |
| Vertical position register i (VPi) | 187 |

| | |
|---|----|
| [W] | |
| Watchdog timer control register (WDC) | 68 |
| Watchdog timer start register (WDTS) | 68 |

Keep safety first in your circuit designs!

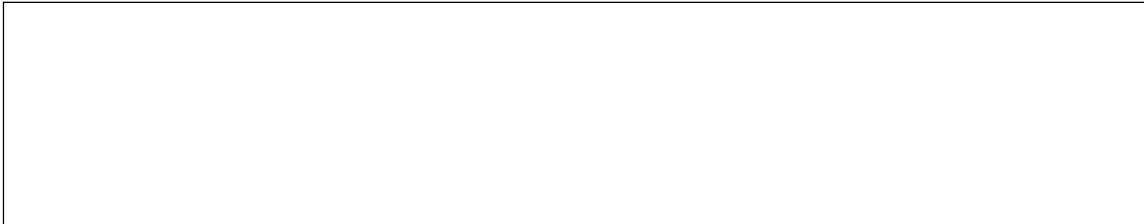
1. Renesas Technology Corporation puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage. Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

1. These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corporation product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corporation or a third party.
 2. Renesas Technology Corporation assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
 3. All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor for the latest product information before purchasing a product listed herein.
The information described here may contain technical inaccuracies or typographical errors.
Renesas Technology Corporation assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.
Please also pay attention to information published by Renesas Technology Corporation by various means, including the Renesas Technology Corporation Semiconductor home page (<http://www.renesas.com>).
 4. When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corporation assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
 5. Renesas Technology Corporation semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
 6. The prior written approval of Renesas Technology Corporation is necessary to reprint or reproduce in whole or in part these materials.
 7. If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
 8. Please contact Renesas Technology Corporation for further details on these materials or the products contained therein.
-

RENESAS

<http://www.renesas.com>



REVISION HISTORY

M306V2ME-XXXFP, M306V2EEFP (REV.1.30) DATA SHEET

| Rev. No. | Revision Description | Rev. date | | | | | | | | | | | | | | | | | | | | | | | | |
|---|--|---------------|---------------|--------------|--|-----------------------------|----------|------------|--|-----------------------------|----------|-------------|--|------------------|--|---------------|--------------|---|--|---------------|-----------|--------------------------|--|---------------|-----------|------|
| 1.0 | First Edition of PDF File | 0006 | | | | | | | | | | | | | | | | | | | | | | | | |
| 1.1 | Serial I/O: <u>4</u> units. (P1) | 0008 | | | | | | | | | | | | | | | | | | | | | | | | |
| 1.2 | <table border="0" style="width: 100%;"> <tr> <td style="width: 40%;"></td> <td style="text-align: center;"><u>BEFORE</u></td> <td style="text-align: center;"><u>AFTER</u></td> <td style="width: 20%;"></td> </tr> <tr> <td>G signal output control bit</td> <td style="text-align: center;">b2 b1 b0</td> <td style="text-align: center;">→ b6 b5 b4</td> <td></td> </tr> <tr> <td>B signal output control bit</td> <td style="text-align: center;">b2 b1 b0</td> <td style="text-align: center;">→ b10 b9 b8</td> <td></td> </tr> </table> <hr/> <table border="0" style="width: 100%;"> <tr> <td style="width: 40%;">P251 Table 5.2.1</td> <td style="width: 20%;"></td> <td style="text-align: center;"><u>BEFORE</u></td> <td style="text-align: center;"><u>AFTER</u></td> </tr> <tr> <td>f_{osc} Oscillation frequency (for OSD) OSC1</td> <td></td> <td style="text-align: center;">Min. 24.0 MHz</td> <td style="text-align: center;">→ 15.0MHz</td> </tr> <tr> <td>Ceramic oscillating mode</td> <td></td> <td style="text-align: center;">Max. 25.0 MHz</td> <td style="text-align: center;">→ 27.0MHz</td> </tr> </table> | | <u>BEFORE</u> | <u>AFTER</u> | | G signal output control bit | b2 b1 b0 | → b6 b5 b4 | | B signal output control bit | b2 b1 b0 | → b10 b9 b8 | | P251 Table 5.2.1 | | <u>BEFORE</u> | <u>AFTER</u> | f _{osc} Oscillation frequency (for OSD) OSC1 | | Min. 24.0 MHz | → 15.0MHz | Ceramic oscillating mode | | Max. 25.0 MHz | → 27.0MHz | 0108 |
| | <u>BEFORE</u> | <u>AFTER</u> | | | | | | | | | | | | | | | | | | | | | | | | |
| G signal output control bit | b2 b1 b0 | → b6 b5 b4 | | | | | | | | | | | | | | | | | | | | | | | | |
| B signal output control bit | b2 b1 b0 | → b10 b9 b8 | | | | | | | | | | | | | | | | | | | | | | | | |
| P251 Table 5.2.1 | | <u>BEFORE</u> | <u>AFTER</u> | | | | | | | | | | | | | | | | | | | | | | | |
| f _{osc} Oscillation frequency (for OSD) OSC1 | | Min. 24.0 MHz | → 15.0MHz | | | | | | | | | | | | | | | | | | | | | | | |
| Ceramic oscillating mode | | Max. 25.0 MHz | → 27.0MHz | | | | | | | | | | | | | | | | | | | | | | | |
| 1.30 | P176 Figure 2.15.1 is changed | 0307 | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | |