

### 1. DESCRIPTION

The M306V7MG/MH-XXXFP and M306V7FG/FHFP are single-chip microcomputers using the high-performance silicon gate CMOS process using a M16C/60 Series CPU core and are packaged in a 100-pin plastic molded QFP. These single-chip microcomputers operate using sophisticated instructions featuring a high level of instruction efficiency. With 1M bytes of address space, they are capable of executing instructions at high speed. They also feature a built-in OSD display function and data slicer, making them ideal for closed caption and ID1 for 525p.

#### 1.1 Features

- Memory size ..... Refer to Figure 1.5.1 ROM development
  - <RAM> 10K bytes
  - <OSD ROM> 60K bytes
  - <OSD RAM> 2.2K bytes
- Shortest instruction execution time ..... 62.5 ns ( $f(X_{IN})=16$  MHz)
- Power source voltage .....
 

V <sub>CCE</sub> (5V system I/O power supply)	4.75 V to 5.25V
V <sub>CC1</sub> (internal logic power supply)	3.15 V to 3.45V
- Power consumption ..... 415mW
- Interrupts ..... 22 internal and 3 external interrupt sources, 4 software interrupt sources; 7 levels
- Multifunction 16-bit timer ..... 2 output timers + 3 input timers + 3 timers
- Serial I/O ..... 4 units
  - UART/clock synchronous: 2
  - Multi-master I<sup>2</sup>C-BUS interface 0 (2 systems): 1
  - Multi-master I<sup>2</sup>C-BUS interface 1 (1 system): 1
- DMAC ..... 2 channels (trigger: 23 sources)
- A-D converter ..... 8 bits X 6 channels
- D-A converter ..... 8 bits X 2 channels
- Data slicer ..... 2 circuits (closed caption and video ID for 525p are available)
- HSYNC counter ..... 1 circuit (2 systems)
- OSD function ..... 1 circuit
- Watchdog timer ..... 1 circuit
- Programmable I/O ..... 76 lines
- Memory expansion ..... Available
- Chip select output ..... 4 lines
- Clock generating circuit ..... 3 built-in clock generation circuits

#### 1.2 Applications

TV with a closed caption and ID1

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

<ul style="list-style-type: none"> <li>1. DESCRIPTION ..... 1           <ul style="list-style-type: none"> <li>1.1 Features ..... 1</li> <li>1.2 Applications ..... 1</li> <li>1.3 Pin Configuration ..... 3</li> <li>1.4 Block Diagram ..... 4</li> <li>1.5 Performance Outline ..... 5</li> </ul> </li> <li>2. OPERATION OF FUNCTIONAL BLOCKS .... 9           <ul style="list-style-type: none"> <li>2.1 Memory ..... 9</li> <li>2.2 Central Processing Unit (CPU) ..... 15</li> <li>2.3 Reset ..... 18</li> <li>2.4 Processor Mode ..... 23</li> <li>2.5 Clock Generating Circuit ..... 36</li> <li>2.6 Protection ..... 46</li> <li>2.7 Interrupts ..... 47</li> <li>2.8 Watchdog Timer ..... 68</li> <li>2.9 DMAC ..... 70</li> <li>2.10 Timer ..... 80</li> <li>2.11 Serial I/O ..... 100</li> <li>2.12 A-D Converter ..... 146</li> <li>2.13 D-A Converter ..... 161</li> <li>2.14 Data Slicer ..... 163</li> <li>2.15 HSYNC Counter ..... 176</li> <li>2.16 OSD Functions ..... 177               <ul style="list-style-type: none"> <li>2.16.1 Triple Layer OSD ..... 183</li> <li>2.16.2 Display Position ..... 185</li> <li>2.16.3 Dot Size ..... 184</li> <li>2.16.4 Clock for OSD ..... 190</li> <li>2.16.5 Field Determination Display ..... 193</li> <li>2.16.6 Memory for OSD ..... 195</li> <li>2.16.7 Character Color ..... 205</li> <li>2.16.8 Character Background Color ..... 205</li> <li>2.16.9 OUT1, OUT2 Signals ..... 210</li> <li>2.16.10 Attribute ..... 211</li> <li>2.16.11 Automatic Solid Space Function .... 216</li> <li>2.16.12 Particular OSD Mode Block ..... 217</li> <li>2.16.13 Multiline Display ..... 219</li> <li>2.16.14 SPRITE OSD Function ..... 220</li> <li>2.16.15 Window Function ..... 223</li> <li>2.16.16 Blank Function ..... 224</li> <li>2.16.17 Raster Coloring Function ..... 227</li> <li>2.16.18 Scan Mode ..... 229</li> </ul> </li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>2.16.19 R, G, B Signal Output Control ... 229</li> <li>2.16.20 OSD Reserved Register ..... 230</li> <li>2.17 Programmable I/O Ports ..... 231</li> <li>3. USAGE PRECAUTION ..... 247           <ul style="list-style-type: none"> <li>3.1 Timer A (timer mode) ..... 247</li> <li>3.2 Timer A (event counter mode) ..... 247</li> <li>3.3 Timer A (one-shot timer mode) ..... 247</li> <li>3.4 Timer A (pulse width modulation mode) .. 247</li> <li>3.5 Timer B (timer mode, event counter mode) ... 247</li> <li>3.6 Timer B (pulse period, pulse width measurement mode) ..... 247</li> <li>3.7 A-D Converter ..... 247</li> <li>3.8 Stop Mode and Wait Mode ..... 247</li> <li>3.9 Interrupts ..... 248</li> <li>3.10 About Flash memory version and mask ROM ..... 249</li> </ul> </li> <li>4. ITEMS TO BE SUBMITTED WHEN ORDERING MASKED ROM VERSION .... 250</li> <li>5. ELECTRICAL CHARACTERISTICS ..... 251           <ul style="list-style-type: none"> <li>5.1. Absolute Maximum Ratings ..... 251</li> <li>5.2 Recommended Operating Conditions .. 252</li> <li>5.3 Electrical Characteristics ..... 253</li> <li>5.4 A-D Conversion Characteristics ..... 254</li> <li>5.5 D-A Conversion Characteristics ..... 254</li> <li>5.6 Analog R, G, B Output Characteristics 254</li> <li>5.7 Timing Requirements ..... 256</li> <li>5.8 Switching Characteristics ..... 258</li> <li>5.9 Measurement Circuit ..... 259</li> <li>5.10 Timing Diagram ..... 260</li> </ul> </li> <li>6. PACKAGE OUTLINE ..... 264</li> <li>7. Flash Memory ..... 265           <ul style="list-style-type: none"> <li>7.1 Description ..... 265</li> <li>7.2 CPU Rewrite Mode ..... 267</li> <li>7.3 Parallel I/O Mode ..... 279</li> <li>7.4 Standard Serial I/O Mode ..... 280</li> </ul> </li> </ul>
--	--

### 1.3 Pin Configuration

Figure 1.3.1 shows the pin configuration (top view).

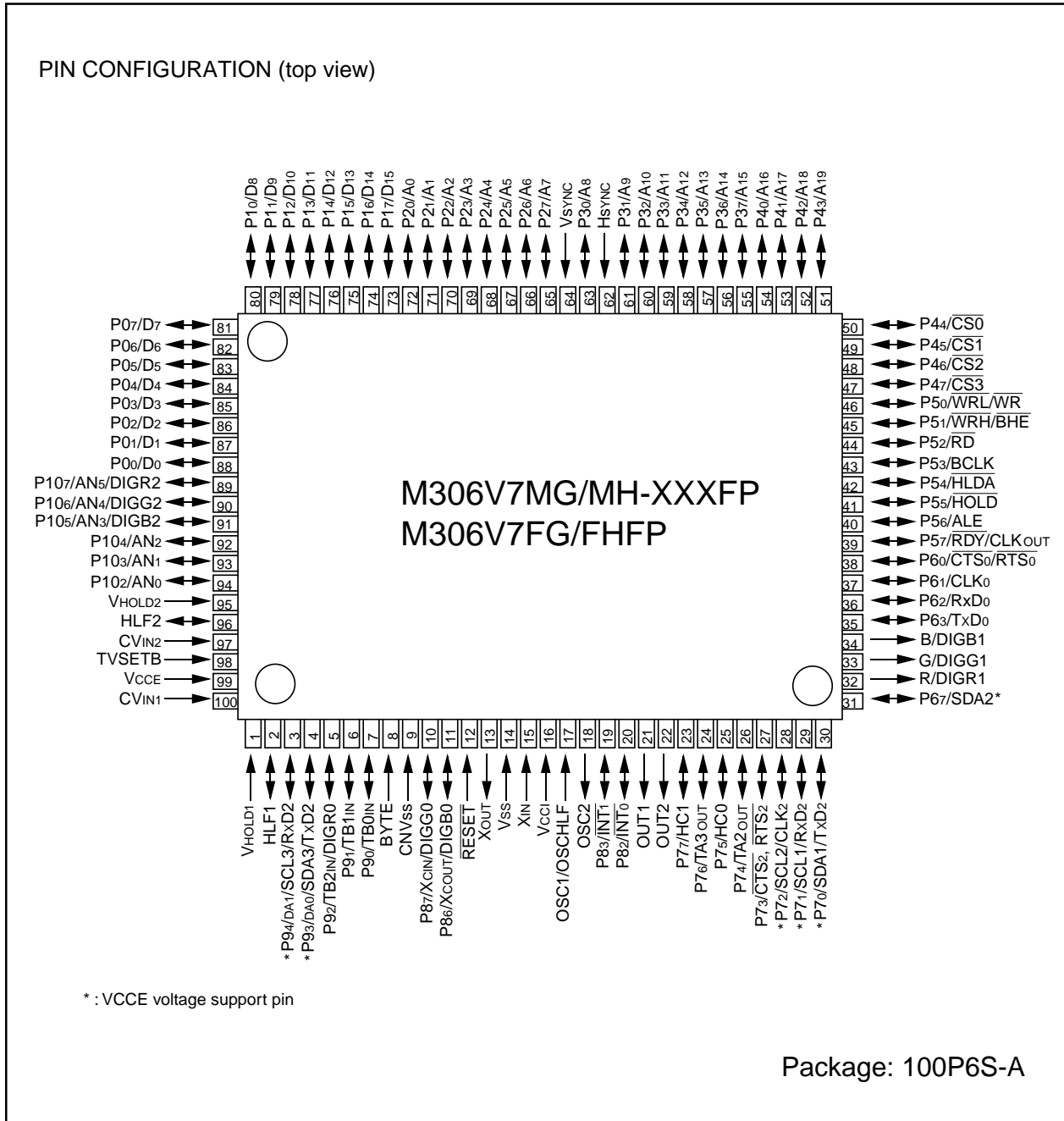


Figure 1.3.1 Pin configuration (top view)

### 1.4 Block Diagram

Figure 1.4.1 is a block diagram.

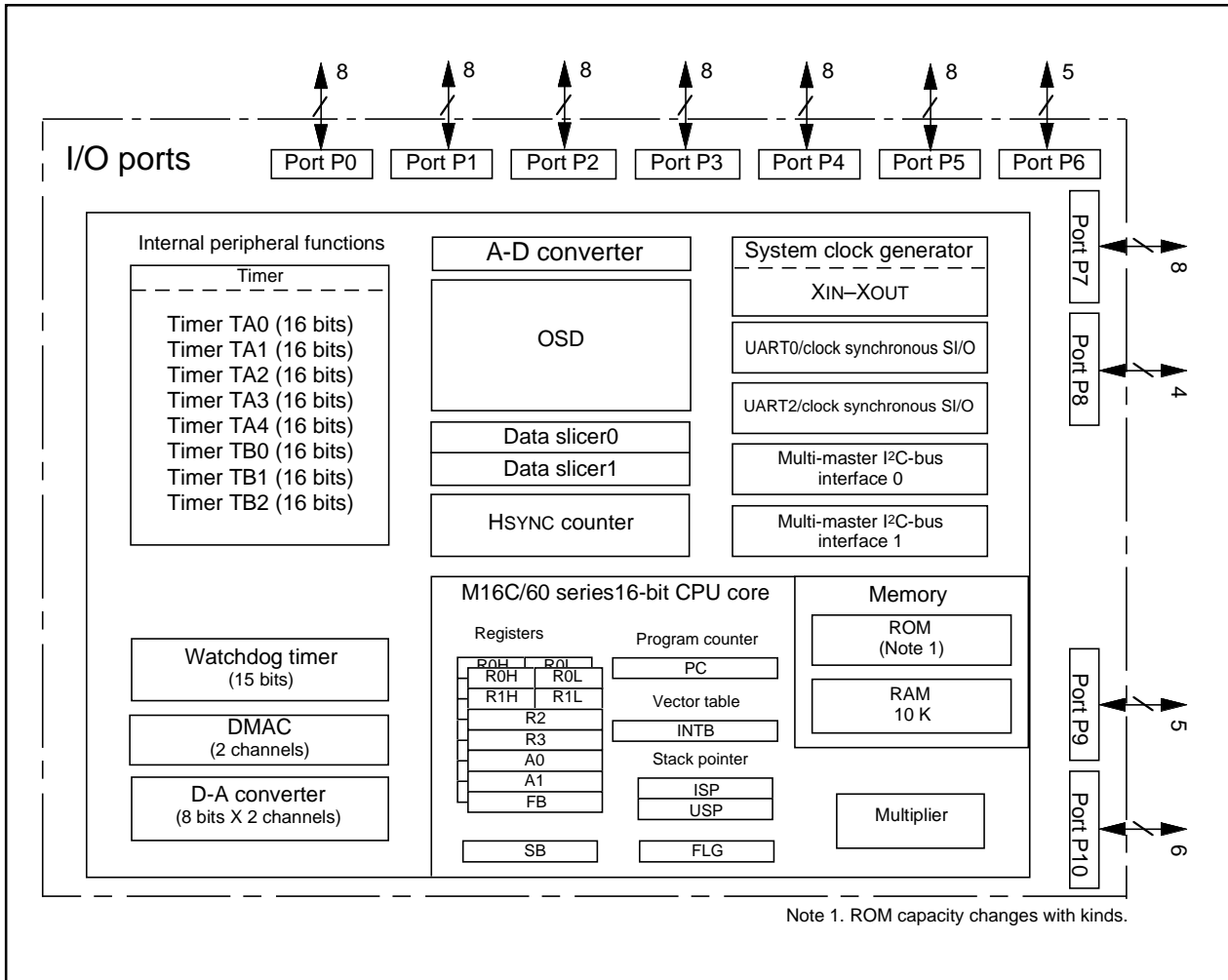


Figure 1.4.1 Block diagram

## 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		62.5 ns( $f(XIN)=16$ MHz)
Memory size	ROM	Refer to Figure 1.5.1 ROM development
	RAM	10K bytes
	OSD ROM	60K bytes
	OSD RAM	2.2K bytes
I/O port	P0 to P10	8 bits × 7, 6 bits × 1, 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 <sup>2</sup> C-BUS interface 0	1 unit (2 channels)
	Multi-master I <sup>2</sup> 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 characters × 16 lines
Data slicer	Data slicer0	16-bit × 2, or data buffer of 16-bit and 20-bit
	Data slicer1	16-bit × 2, or data buffer of 16-bit and 20-bit
HSYNC counter		8 bits × 2 channels
Watchdog timer		15 bits × 1 (with prescaler)
Interrupt		22 internal and 3 external sources, 4 software sources, 7 levels
Clock generating circuit		3 built-in clock generation circuits
Power source voltage	VCCE(5V system I/O power supply)	4.75 V to 5.25V
	VCCI(internal logic power supply)	3.15 V to 3.45V
Power consumption		415mW
I/O characteristics	I/O withstand voltage	3.3 V (only P67, P70, P71, P72, P93 and P94 : 5V )
	Output current	5 mA
Memory expansion		Available
Operating ambient temperature		-20 °C to 70 °C
Device configuration		CMOS high performance silicon gate
Package		100-pin plastic molded QFP

Currently supported products are listed below.

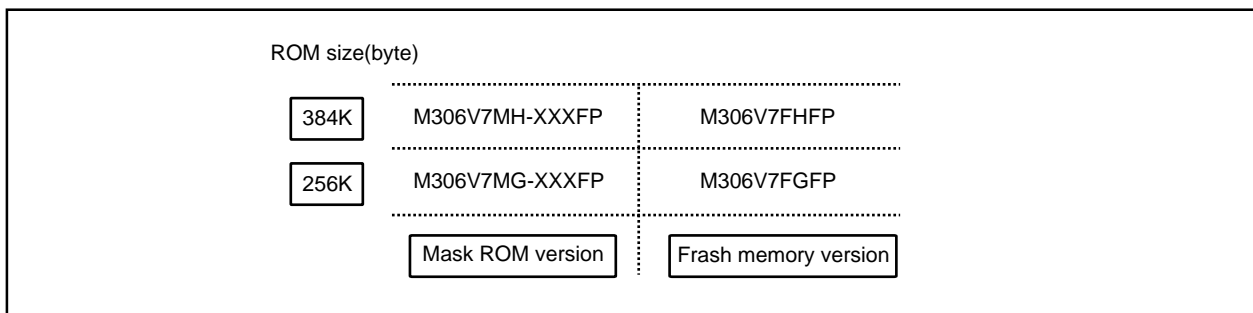


Figure 1.5.1 ROM development

Table 1.5.2 List of supported products

Type No	ROM capacity	RAM capacity	Package type	Remarks
M306V7MG-XXXFP	256K bytes	10K bytes	100P6S-A	Mask ROM version
M306V7MH-XXXFP	384K bytes			
M306V7FGFP	256K bytes			Flash Memory version
M306V7FHFP	384K bytes			

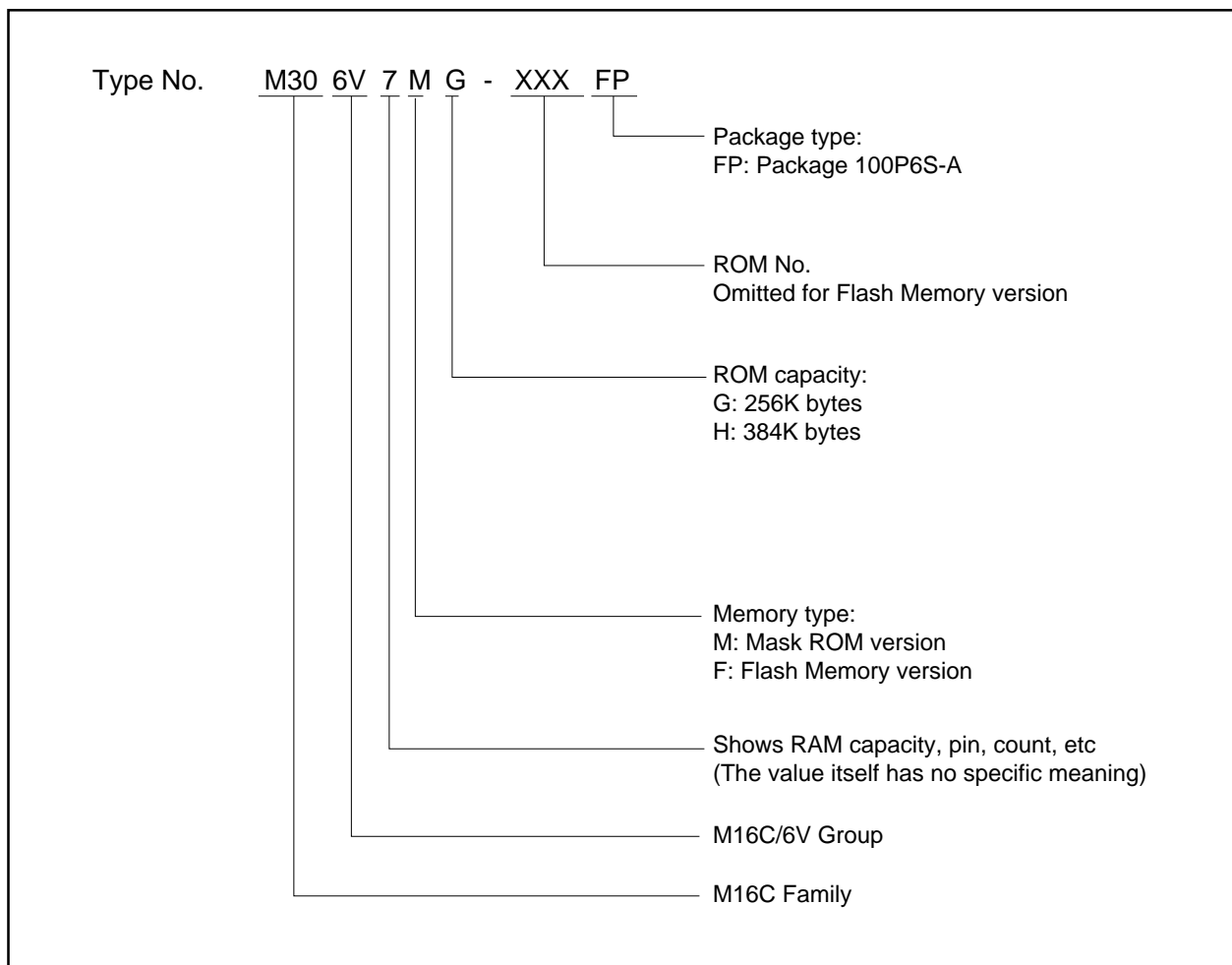


Figure 1.5.2 Type No., memory size, and package

Table 1.5.3 Pin description (1)

Pin name	Signal name	I/O type	Function
VCCE, VCCI, VSS	Power supply input		Supply 4.75V to 5.25V to the VCCE pin. Supply 3.15V to 3.45V to the VCCI pin. Supply 0V 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 VCCI pin when in microprocessor mode.
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.
P00 to P07	I/O port P0	Input/output	This is an 8-bit CMOS I/O port. It has an input/output port direction register that allows the user to set each pin for input or output individually. When 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	These pins input and output data (D0–D7).
P10 to P17	I/O port P1	Input/output	This is an 8-bit I/O port equivalent to P0.
D8 to D15		Input/output	These pins input and output data (D8–D15).
P20 to P27	I/O port P2	Input/output	This is an 8-bit I/O port equivalent to P0.
A0 to A7		Output	These pins output 8 low-order address bits (A0–A7).
P30 to P37	I/O port P3	Input/output	This is an 8-bit I/O port equivalent to P0.
A8 to A15		Output	These pins output 8 middle-order address bits (A8–A15).
P40 to P47	I/O port P4	Input/output	This is an 8-bit I/O port equivalent to P0.
CS0 to CS3, A16 to A19		Output Output	These pins output CS0–CS3 signals and A16–A19. CS0–CS3 are chip select signals used to specify an access space. A16–A19 are 4 high-order address bits.
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.
WRL / WR, WRH / BHE, RD, BCLK, HLDA, HOLD, ALE, RDY		Output Output Output Output Input Output Input	Output WRL, WRH (WR and BHE), RD, BCLK, HLDA, and ALE signals. WRL and WRH, and BHE and WR can be switched using software control. <ul style="list-style-type: none"> <li>■ WRL, WRH, and RD selected With a 16-bit external data bus, data is written to even addresses when the WRL signal is "L" and to the odd addresses when the WRH signal is "L". Data is read when RD is "L".</li> <li>■ WR, BHE, and RD selected Data is written when WR is "L". Data is read when RD is "L". Odd addresses are accessed when BHE is "L". Use this mode when using an 8-bit external data bus.</li> </ul> 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. While the input level of the RDY pin is "L", the microcomputer is in the ready state. ALE output is indefinite.

**Table 1.5.4 Pin description (continued) (2)**

Pin name	Signal name	I/O type	Function
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 and multi-master I <sup>2</sup> 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 <sup>2</sup> 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, G0 and B0 output pins of digital RGB output. 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 <sup>2</sup> C-BUS interface 1 I/O pins, RXD2, and TXD2 pins. P92 can be set using software to function as the R0 output pin of digital RGB output.
P102 to P107	I/O port P10	Input/output	This is an 6-bit I/O port equivalent to P6. Pins in this port also function as A-D converter input pins. P105 to P107 can be set using software to function as the B2, G2, and R2 output pins of digital RGB output.
Hsync	Synchronous signal input for OSD	Input	This is horizontal synchronous signal pin for OSD.
Vsync	Synchronous signal input for OSD	Input	This is vertical synchronous signal pin for OSD.
R, G, B	OSD output	Output	These are OSD output pins (Digital/analog outputs selectable).
OUT1, OUT2	OSD output	Output	These are OSD output pins (digital output).
OSC1/ OSCHLF	Clock for OSD	Input	OSD clock input or filter pin.
OSC2	Clock for OSD	Output	This is an OSD clock output pin.
CVIN1 CVIN2	I/O for data slicer	Input	Input composite video signal through a capacitor.
VHOLD1/ VHOLD2		Input	Connect a capacitor between VHOLD and Vss.
HLF1/HLF2		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 BLOCKS

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. For example, when M306V7MG-XXXFP is used, there is 256K bytes of internal ROM from  $C0000_{16}$  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.

10K bytes of internal RAM is mapped to the space from  $02C00_{16}$  to  $053FF_{16}$ . 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. For example, area shown below cannot be used in M306V7MG-XXXFP.

- The space between  $01400_{16}$  and  $02BFF_{16}$  (in memory expansion and microprocessor modes)
- The space between  $05400_{16}$  and  $07FFF_{16}$  (in memory expansion and microprocessor modes)
- The space between  $60000_{16}$  and  $BFFFF_{16}$  (in memory expansion mode)

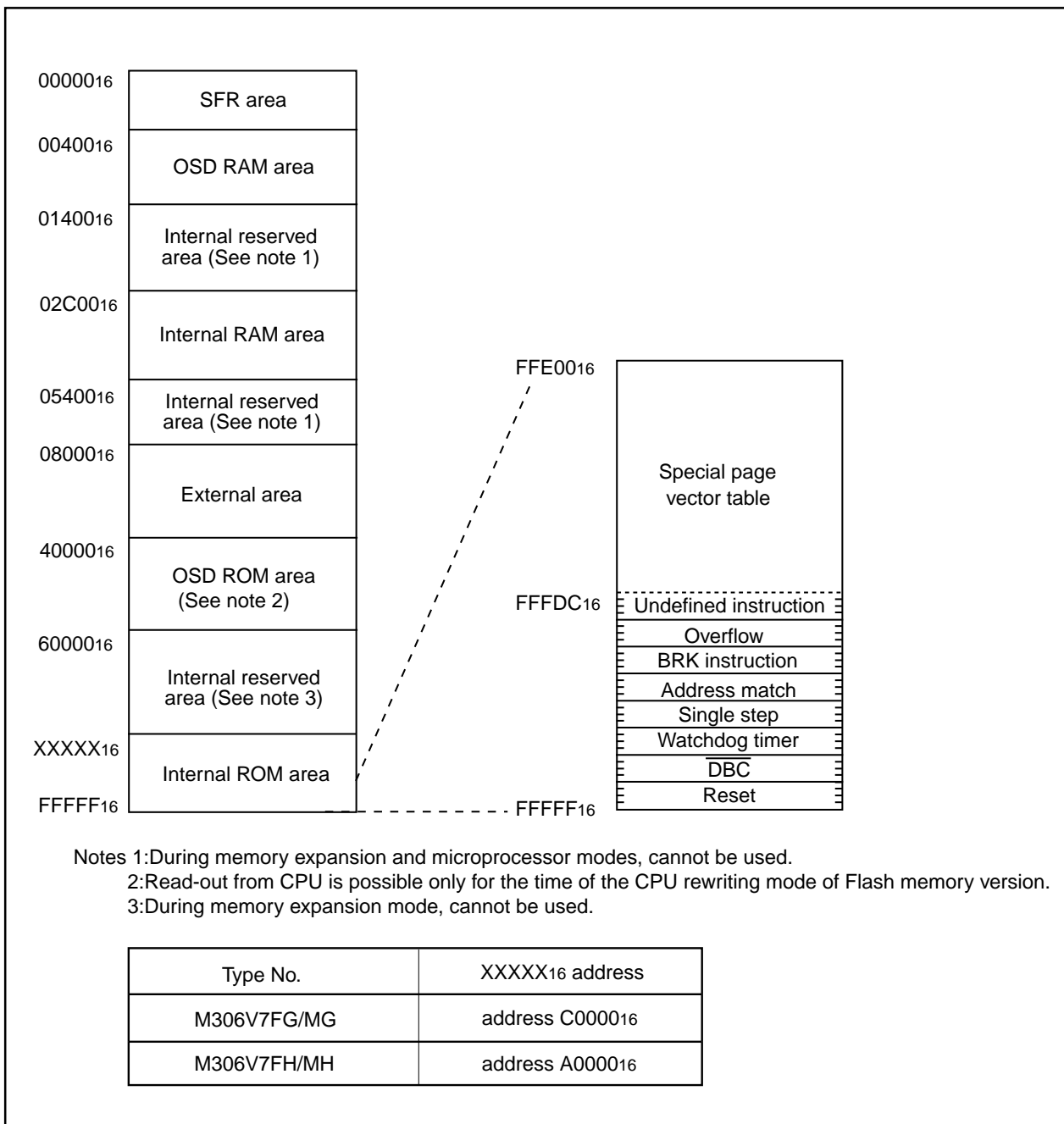


Figure 2.1.1 Memory map

0000 <sub>16</sub>		0040 <sub>16</sub>	
0001 <sub>16</sub>		0041 <sub>16</sub>	
0002 <sub>16</sub>		0042 <sub>16</sub>	
0003 <sub>16</sub>		0043 <sub>16</sub>	
0004 <sub>16</sub>	Processor mode register 0 (PM0)	0044 <sub>16</sub>	OSD1 interrupt control register (OSD1IC)
0005 <sub>16</sub>	Processor mode register 1 (PM1)	0045 <sub>16</sub>	Data slicer 1 interrupt control register (DS1IC)
0006 <sub>16</sub>	System clock control register 0 (CM0)	0046 <sub>16</sub>	Interrupt control reserved register 1 (RE1IC)
0007 <sub>16</sub>	System clock control register 1 (CM1)	0047 <sub>16</sub>	Interrupt control reserved register 2 (RE2IC)
0008 <sub>16</sub>	Chip select control register (CSR)	0048 <sub>16</sub>	OSD2 interrupt control register (OSD2IC)
0009 <sub>16</sub>	Address match interrupt enable register (AIER)	0049 <sub>16</sub>	Multi-master I <sup>2</sup> C-BUS interface 1 interrupt control register (IIC01C)
000A <sub>16</sub>	Protect register (PRCR)	004A <sub>16</sub>	Bus collision detection interrupt control register (BCNIC)
000B <sub>16</sub>		004B <sub>16</sub>	DMA0 interrupt control register (DM0IC)
000C <sub>16</sub>		004C <sub>16</sub>	DMA1 interrupt control register (DM1IC)
000D <sub>16</sub>		004D <sub>16</sub>	Multi-master I <sup>2</sup> C-BUS interface 0 interrupt control register (IIC00C)
000E <sub>16</sub>	Watchdog timer start register (WDTS)	004E <sub>16</sub>	A-D conversion interrupt control register (ADIC)
000F <sub>16</sub>	Watchdog timer control register (WDC)	004F <sub>16</sub>	UART2 transmit interrupt control register (S2TIC)
0010 <sub>16</sub>		0050 <sub>16</sub>	UART2 receive interrupt control register (S2RIC)
0011 <sub>16</sub>	Address match interrupt register 0 (RMAD0)	0051 <sub>16</sub>	UART0 transmit interrupt control register (S0TIC)
0012 <sub>16</sub>		0052 <sub>16</sub>	UART0 receive interrupt control register (S0RIC)
0013 <sub>16</sub>		0053 <sub>16</sub>	Data slicer 0 interrupt control register (DS0IC)
0014 <sub>16</sub>		0054 <sub>16</sub>	VSYNC interrupt control register (VSYNCIC)
0015 <sub>16</sub>	Address match interrupt register 1 (RMAD1)	0055 <sub>16</sub>	Timer A0 interrupt control register (TA0IC)
0016 <sub>16</sub>		0056 <sub>16</sub>	Timer A1 interrupt control register (TA1IC)
0017 <sub>16</sub>		0057 <sub>16</sub>	Timer A2 interrupt control register (TA2IC)
0018 <sub>16</sub>		0058 <sub>16</sub>	Timer A3 interrupt control register (TA3IC)
0019 <sub>16</sub>		0059 <sub>16</sub>	Timer A4 interrupt control register (TA4IC)
001A <sub>16</sub>		005A <sub>16</sub>	Timer B0 interrupt control register (TB0IC)
001B <sub>16</sub>		005B <sub>16</sub>	Timer B1 interrupt control register (TB1IC)
001C <sub>16</sub>		005C <sub>16</sub>	Timer B2 interrupt control register (TB2IC)
001D <sub>16</sub>		005D <sub>16</sub>	INT0 interrupt control register (INT0IC)
001E <sub>16</sub>		005E <sub>16</sub>	INT1 interrupt control register (INT1IC)
001F <sub>16</sub>		005F <sub>16</sub>	Interrupt control reserved register 3 (RE3IC)
0020 <sub>16</sub>		0060 <sub>16</sub>	
0021 <sub>16</sub>	DMA0 source pointer (SAR0)		
0022 <sub>16</sub>			
0023 <sub>16</sub>			
0024 <sub>16</sub>			
0025 <sub>16</sub>	DMA0 destination pointer (DAR0)		
0026 <sub>16</sub>			
0027 <sub>16</sub>			
0028 <sub>16</sub>	DMA0 transfer counter (TCR0)		
0029 <sub>16</sub>			
002A <sub>16</sub>			
002B <sub>16</sub>			
002C <sub>16</sub>	DMA0 control register (DM0CON)		
002D <sub>16</sub>			
002E <sub>16</sub>			
002F <sub>16</sub>			
0030 <sub>16</sub>			
0031 <sub>16</sub>	DMA1 source pointer (SAR1)		
0032 <sub>16</sub>			
0033 <sub>16</sub>			
0034 <sub>16</sub>			
0035 <sub>16</sub>	DMA1 destination pointer (DAR1)		
0036 <sub>16</sub>			
0037 <sub>16</sub>			
0038 <sub>16</sub>	DMA1 transfer counter (TCR1)		
0039 <sub>16</sub>			
003A <sub>16</sub>			
003B <sub>16</sub>			
003C <sub>16</sub>	DMA1 control register (DM1CON)		
003D <sub>16</sub>			
003E <sub>16</sub>			
003F <sub>16</sub>			
		01FF <sub>16</sub>	

Figure 2.1.2 Location of peripheral unit control registers (1)

0200 <sub>16</sub>		0240 <sub>16</sub>	Color palette register 1 (CR1)
0201 <sub>16</sub>	SPRITE OSD control register (SC)	0241 <sub>16</sub>	Color palette register 2 (CR2)
0202 <sub>16</sub>	OSD control register 1 (OC1)	0242 <sub>16</sub>	Color palette register 2 (CR2)
0203 <sub>16</sub>	OSD control register 2 (OC2)	0243 <sub>16</sub>	Color palette register 2 (CR2)
0204 <sub>16</sub>	Horizontal position register (HP)	0244 <sub>16</sub>	Color palette register 3 (CR3)
0205 <sub>16</sub>	Clock control register 1 (CS)	0245 <sub>16</sub>	Color palette register 3 (CR3)
0206 <sub>16</sub>	I/O polarity control register (PC)	0246 <sub>16</sub>	Color palette register 4 (CR4)
0207 <sub>16</sub>	OSD control register 3 (OC3)	0247 <sub>16</sub>	Color palette register 4 (CR4)
0208 <sub>16</sub>		0248 <sub>16</sub>	Color palette register 5 (CR5)
0209 <sub>16</sub>	Raster color register (RSC)	0249 <sub>16</sub>	Color palette register 5 (CR5)
020A <sub>16</sub>	OSD reserved register 5 (OR5)	024A <sub>16</sub>	Color palette register 6 (CR6)
020B <sub>16</sub>	Clock control register 2 (CG)	024B <sub>16</sub>	Color palette register 6 (CR6)
020C <sub>16</sub>		024C <sub>16</sub>	Color palette register 7 (CR7)
020D <sub>16</sub>	Top border control register (TBR)	024D <sub>16</sub>	Color palette register 7 (CR7)
020E <sub>16</sub>		024E <sub>16</sub>	Color palette register 9 (CR9)
020F <sub>16</sub>	Bottom border control register (BBR)	024F <sub>16</sub>	Color palette register 9 (CR9)
0210 <sub>16</sub>	Block control register 1 (BC1)	0250 <sub>16</sub>	Color palette register 10 (CR10)
0211 <sub>16</sub>	Block control register 2 (BC2)	0251 <sub>16</sub>	Color palette register 10 (CR10)
0212 <sub>16</sub>	Block control register 3 (BC3)	0252 <sub>16</sub>	Color palette register 11 (CR11)
0213 <sub>16</sub>	Block control register 4 (BC4)	0253 <sub>16</sub>	Color palette register 11 (CR11)
0214 <sub>16</sub>	Block control register 5 (BC5)	0254 <sub>16</sub>	Color palette register 12 (CR12)
0215 <sub>16</sub>	Block control register 6 (BC6)	0255 <sub>16</sub>	Color palette register 12 (CR12)
0216 <sub>16</sub>	Block control register 7 (BC7)	0256 <sub>16</sub>	Color palette register 13 (CR13)
0217 <sub>16</sub>	Block control register 8 (BC8)	0257 <sub>16</sub>	Color palette register 13 (CR13)
0218 <sub>16</sub>	Block control register 9 (BC9)	0258 <sub>16</sub>	Color palette register 14 (CR14)
0219 <sub>16</sub>	Block control register 10 (BC10)	0259 <sub>16</sub>	Color palette register 14 (CR14)
021A <sub>16</sub>	Block control register 11 (BC11)	025A <sub>16</sub>	Color palette register 15 (CR15)
021B <sub>16</sub>	Block control register 12 (BC12)	025B <sub>16</sub>	Color palette register 15 (CR15)
021C <sub>16</sub>	Block control register 13 (BC13)	025C <sub>16</sub>	
021D <sub>16</sub>	Block control register 14 (BC14)	025D <sub>16</sub>	OSD reserved register 1 (OR1)
021E <sub>16</sub>	Block control register 15 (BC15)	025E <sub>16</sub>	
021F <sub>16</sub>	Block control register 16 (BC16)	025F <sub>16</sub>	OSD control register 4 (OC4)
0220 <sub>16</sub>	Vertical position register 1 (VP1)	0260 <sub>16</sub>	Data slicer 0 control register 1 (DSC01)
0221 <sub>16</sub>		0261 <sub>16</sub>	Data slicer 0 control register 2 (DSC02)
0222 <sub>16</sub>	Vertical position register 2 (VP2)	0262 <sub>16</sub>	
0223 <sub>16</sub>		0263 <sub>16</sub>	Caption data register 01 (CD01)
0224 <sub>16</sub>	Vertical position register 3 (VP3)	0264 <sub>16</sub>	
0225 <sub>16</sub>		0265 <sub>16</sub>	Caption data register 02 (CD02)
0226 <sub>16</sub>	Vertical position register 4 (VP4)	0266 <sub>16</sub>	Caption position register 0 (CPS0)
0227 <sub>16</sub>		0267 <sub>16</sub>	Slice standard voltage selection register 0 (SBV0)
0228 <sub>16</sub>	Vertical position register 5 (VP5)	0268 <sub>16</sub>	Data slicer 0 reserved register 1 (DR01)
0229 <sub>16</sub>		0269 <sub>16</sub>	Clock run-in detect register 0 (CRD0)
022A <sub>16</sub>	Vertical position register 6 (VP6)	026A <sub>16</sub>	Data clock position register 0 (DPS0)
022B <sub>16</sub>		026B <sub>16</sub>	ID1 control register 0 (IDC0)
022C <sub>16</sub>	Vertical position register 7 (VP7)	026C <sub>16</sub>	Standard clock detection register 0 (BCD0)
022D <sub>16</sub>		026D <sub>16</sub>	CRCC data register 0 (CRC0)
022E <sub>16</sub>	Vertical position register 8 (VP8)	026E <sub>16</sub>	Test reserved register 0 (IDT0)
022F <sub>16</sub>		026F <sub>16</sub>	Reserved register (RSV0)
0230 <sub>16</sub>	Vertical position register 9 (VP9)	0270 <sub>16</sub>	Left border control register (LBR)
0231 <sub>16</sub>		0271 <sub>16</sub>	
0232 <sub>16</sub>	Vertical position register 10 (VP10)	0272 <sub>16</sub>	Right border control register (RBR)
0233 <sub>16</sub>		0273 <sub>16</sub>	
0234 <sub>16</sub>	Vertical position register 11 (VP11)	0274 <sub>16</sub>	SPRITE vertical position register 1 (VS1)
0235 <sub>16</sub>		0275 <sub>16</sub>	
0236 <sub>16</sub>	Vertical position register 12 (VP12)	0276 <sub>16</sub>	SPRITE vertical position register 2 (VS2)
0237 <sub>16</sub>		0277 <sub>16</sub>	
0238 <sub>16</sub>	Vertical position register 13 (VP13)	0278 <sub>16</sub>	SPRITE horizontal position register (HS)
0239 <sub>16</sub>		0279 <sub>16</sub>	
023A <sub>16</sub>	Vertical position register 14 (VP14)	027A <sub>16</sub>	OSD reserved register 4 (OR4)
023B <sub>16</sub>		027B <sub>16</sub>	OSD reserved register 3 (OR3)
023C <sub>16</sub>	Vertical position register 15 (VP15)	027C <sub>16</sub>	OSD reserved register 2 (OR2)
023D <sub>16</sub>		027D <sub>16</sub>	Peripheral mode register (PM)
023E <sub>16</sub>	Vertical position register 16 (VP16)	027E <sub>16</sub>	Hsync counter register (HC)
023F <sub>16</sub>		027F <sub>16</sub>	Hsync counter latch
		0280 <sub>16</sub>	Internal oscillation control register 1 (DIV0)
		0281 <sub>16</sub>	Internal oscillation control register 2 (DIV1)
		0282 <sub>16</sub>	Internal oscillation control register 3 (VCO)
		029E	R0T reserved register (R0TRSV1)
		029F	R0T reserved register (R0TRSV2)
		02DF <sub>16</sub>	

Figure 2.1.3 Location of peripheral unit control registers (2)

02E0 <sub>16</sub>	I <sup>2</sup> C0 data shift register (IIC0S0)	0380 <sub>16</sub>	Count start flag (TABSR)
02E1 <sub>16</sub>	I <sup>2</sup> C0 address register (IIC0S0D)	0381 <sub>16</sub>	Clock prescaler reset flag (CPSRF)
02E2 <sub>16</sub>	I <sup>2</sup> C0 status register (IIC0S1)	0382 <sub>16</sub>	One-shot start flag (ONSF)
02E3 <sub>16</sub>	I <sup>2</sup> C0 control register (IIC0S1D)	0383 <sub>16</sub>	Trigger select register (TRGSR)
02E4 <sub>16</sub>	I <sup>2</sup> C0 clock control register (IIC0S2)	0384 <sub>16</sub>	Up-down flag (UDF)
02E5 <sub>16</sub>	I <sup>2</sup> C0 port selection register (IIC0S2D)	0385 <sub>16</sub>	
02E6 <sub>16</sub>	I <sup>2</sup> C0 transmit buffer register (IIC0S0S)	0386 <sub>16</sub>	Timer A0 register (TA0)
02E7 <sub>16</sub>		0387 <sub>16</sub>	
02E8 <sub>16</sub>	I <sup>2</sup> C1 data shift register (IIC1S0)	0388 <sub>16</sub>	Timer A1 register (TA1)
02E9 <sub>16</sub>	I <sup>2</sup> C1 address register (IIC1S0D)	0389 <sub>16</sub>	
02EA <sub>16</sub>	I <sup>2</sup> C1 status register (IIC1S1)	038A <sub>16</sub>	Timer A2 register (TA2)
02EB <sub>16</sub>	I <sup>2</sup> C1 control register (IIC1S1D)	038B <sub>16</sub>	
02EC <sub>16</sub>	I <sup>2</sup> C1 clock control register (IIC1S2)	038C <sub>16</sub>	Timer A3 register (TA3)
02ED <sub>16</sub>	I <sup>2</sup> C1 port selection register (IIC1S2D)	038D <sub>16</sub>	
02EE <sub>16</sub>	I <sup>2</sup> C1 transmit buffer register (IIC1S0S)	038E <sub>16</sub>	Timer A4 register (TA4)
0300 <sub>16</sub>	Data slicer 1 control register 1 (DSC11)	038F <sub>16</sub>	
0301 <sub>16</sub>	Data slicer 1 control register 2 (DSC12)	0390 <sub>16</sub>	Timer B0 register (TB0)
0302 <sub>16</sub>		0391 <sub>16</sub>	
0303 <sub>16</sub>	Caption data register 11 (CD11)	0392 <sub>16</sub>	Timer B1 register (TB1)
0304 <sub>16</sub>		0393 <sub>16</sub>	
0305 <sub>16</sub>	Caption data register 12 (CD12)	0394 <sub>16</sub>	Timer B2 register (TB2)
0306 <sub>16</sub>		0395 <sub>16</sub>	
0307 <sub>16</sub>	Caption position register 1 (CPS1)	0396 <sub>16</sub>	Timer A0 mode register (TA0MR)
0308 <sub>16</sub>	Slice standard voltage selection register 1 (SBV1)	0397 <sub>16</sub>	Timer A1 mode register (TA1MR)
0309 <sub>16</sub>	Data slicer 1 reserved register 1 (DR11)	0398 <sub>16</sub>	Timer A2 mode register (TA2MR)
0309 <sub>16</sub>	Clock run-in detection register 1 (CRD1)	0399 <sub>16</sub>	Timer A3 mode register (TA3MR)
030A <sub>16</sub>		039A <sub>16</sub>	Timer A4 mode register (TA4MR)
030A <sub>16</sub>	Data clock position register 1 (DPS1)	039B <sub>16</sub>	Timer B0 mode register (TB0MR)
030B <sub>16</sub>	ID1 control register 1 (IDC1)	039C <sub>16</sub>	Timer B1 mode register (TB1MR)
030C <sub>16</sub>	Standard clock detection register 1 (BCD1)	039D <sub>16</sub>	Timer B2 mode register (TB2MR)
030D <sub>16</sub>	CRCC data register (CRC1)	039E <sub>16</sub>	
030E <sub>16</sub>	Test reserved register 1 (IDT1)	039F <sub>16</sub>	
030F <sub>16</sub>	Reserved register (RSV1)	03A0 <sub>16</sub>	UART0 transmit/receive mode register (U0MR)
0310 <sub>16</sub>		03A1 <sub>16</sub>	UART0 bit rate generator (U0BRG)
0311 <sub>16</sub>	FM control 3 reserved register (FMRU3)	03A2 <sub>16</sub>	
0312 <sub>16</sub>	FM control 2 reserved register (FMRU2)	03A3 <sub>16</sub>	UART0 transmit buffer register (U0TB)
0313 <sub>16</sub>	Flash USER control register (FMRU)	03A4 <sub>16</sub>	UART0 transmit/receive control register 0 (U0C0)
0314 <sub>16</sub>		03A5 <sub>16</sub>	UART0 transmit/receive control register 1 (U0C1)
0315 <sub>16</sub>	FO control 3 reserved register (FMRD3)	03A6 <sub>16</sub>	
0316 <sub>16</sub>	FO control 2 reserved register (FMRD2)	03A7 <sub>16</sub>	UART0 receive buffer register (U0RB)
0317 <sub>16</sub>	Flash OSD control register (FMRD)	03A8 <sub>16</sub>	
0318 <sub>16</sub>	Flash memory change register (FMSEL)	03A9 <sub>16</sub>	
0319 <sub>16</sub>	ID1 reserved register 0 (IRSV0)	03AA <sub>16</sub>	
031C <sub>16</sub>	ID1 reserved register 1 (IRSV1)	03AB <sub>16</sub>	
031D <sub>16</sub>		03AC <sub>16</sub>	Reserved register (RUS0S4)
035E <sub>16</sub>		03AD <sub>16</sub>	Reserved register (RUS0S3)
035F <sub>16</sub>	Interrupt request cause select register (IFSR)	03AE <sub>16</sub>	Reserved register (RUS0S2)
0360 <sub>16</sub>		03AF <sub>16</sub>	Reserved register (RUS0S1)
		03B0 <sub>16</sub>	
		03B1 <sub>16</sub>	
		03B2 <sub>16</sub>	
		03B3 <sub>16</sub>	
0373 <sub>16</sub>		03B4 <sub>16</sub>	
0374 <sub>16</sub>	Reserved register (RUS2S4)	03B5 <sub>16</sub>	
0375 <sub>16</sub>	Reserved register (RUS2S3)	03B6 <sub>16</sub>	
0376 <sub>16</sub>	Reserved register (RUS2S2)	03B7 <sub>16</sub>	
0377 <sub>16</sub>	UART2 special mode register (U2SMR)	03B8 <sub>16</sub>	DMA0 request cause select register (DM0SL)
0378 <sub>16</sub>	UART2 transmit/receive mode register (U2MR)	03B9 <sub>16</sub>	
0379 <sub>16</sub>	UART2 bit rate generator (U2BRG)	03BA <sub>16</sub>	DMA1 request cause select register (DM1SL)
037A <sub>16</sub>		03BB <sub>16</sub>	
037B <sub>16</sub>	UART2 transmit buffer register (U2TB)	03BC <sub>16</sub>	
037C <sub>16</sub>	UART2 transmit/receive control register 0 (U2C0)	03BD <sub>16</sub>	
037D <sub>16</sub>	UART2 transmit/receive control register 1 (U2C1)	03BE <sub>16</sub>	
037E <sub>16</sub>		03BF <sub>16</sub>	
037F <sub>16</sub>	UART2 receive buffer register (U2RB)		

Note: Flash USER control register (FMRU), Flash OSD control register (FMRD) and Flash memory change register (FMSEL) are only for M306V7FGFP.

Figure 2.1.4 Location of peripheral unit control registers (3)

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

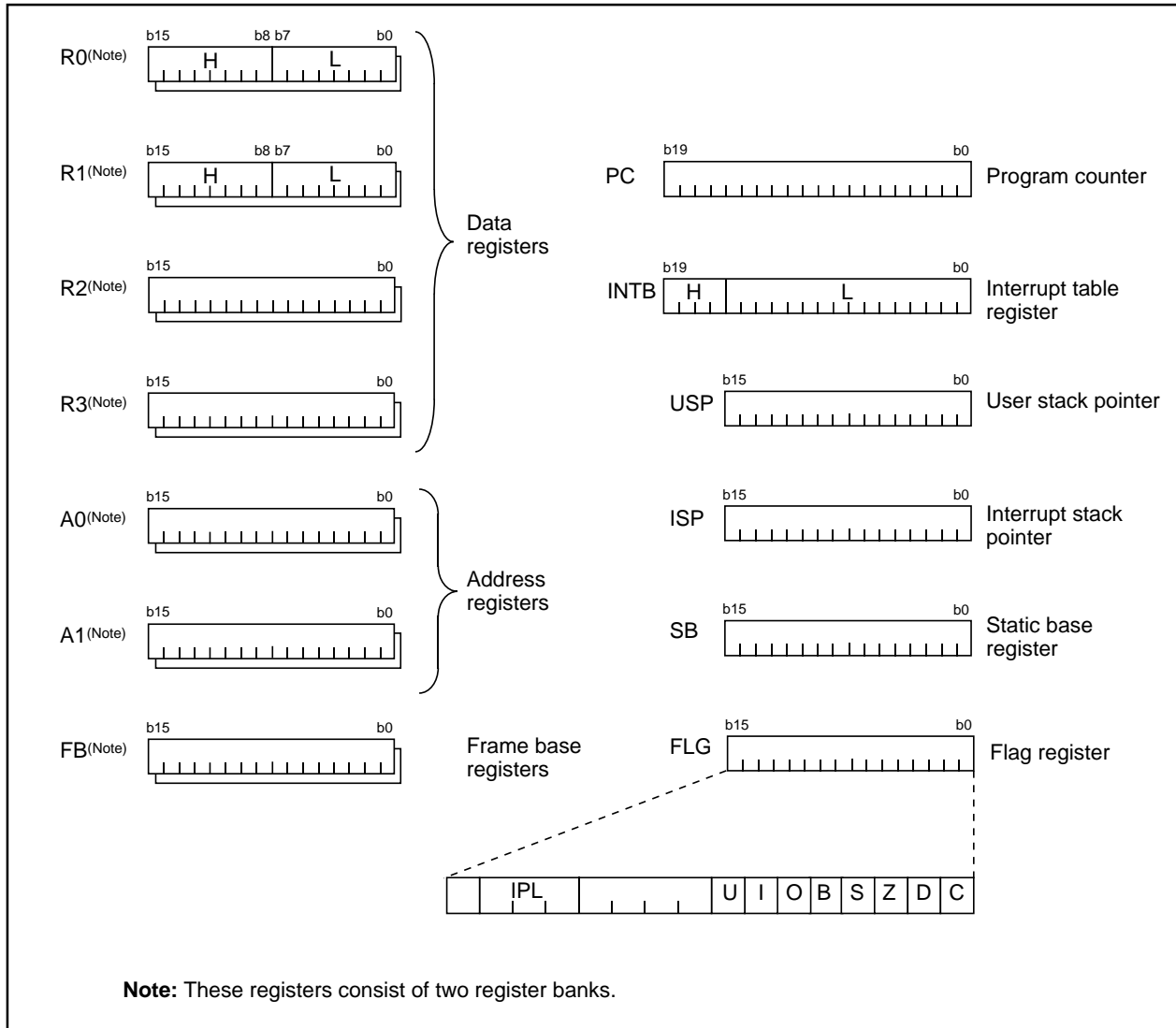


Figure 2.2.1 Central processing unit register

### 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**

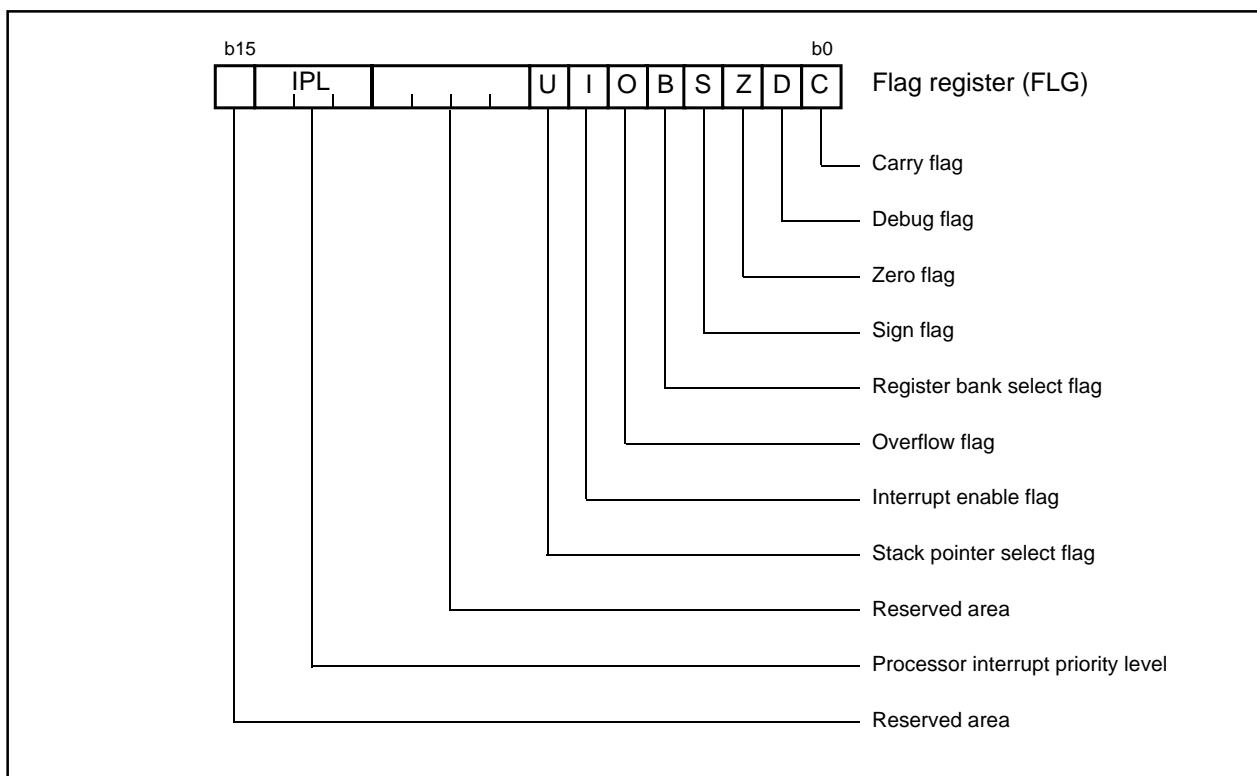
- **Bits 12 to 14: Processor interrupt priority level (IPL)**

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<sub>CC</sub> max.) for at least 20 cycles. When the reset pin level is then returned to the “H” level while main clock is stable, the reset status is cancelled and program execution resumes from the address in the reset vector table.

Figure 2.3.1 shows the example reset circuit. Figure 2.3.2 shows the reset sequence.

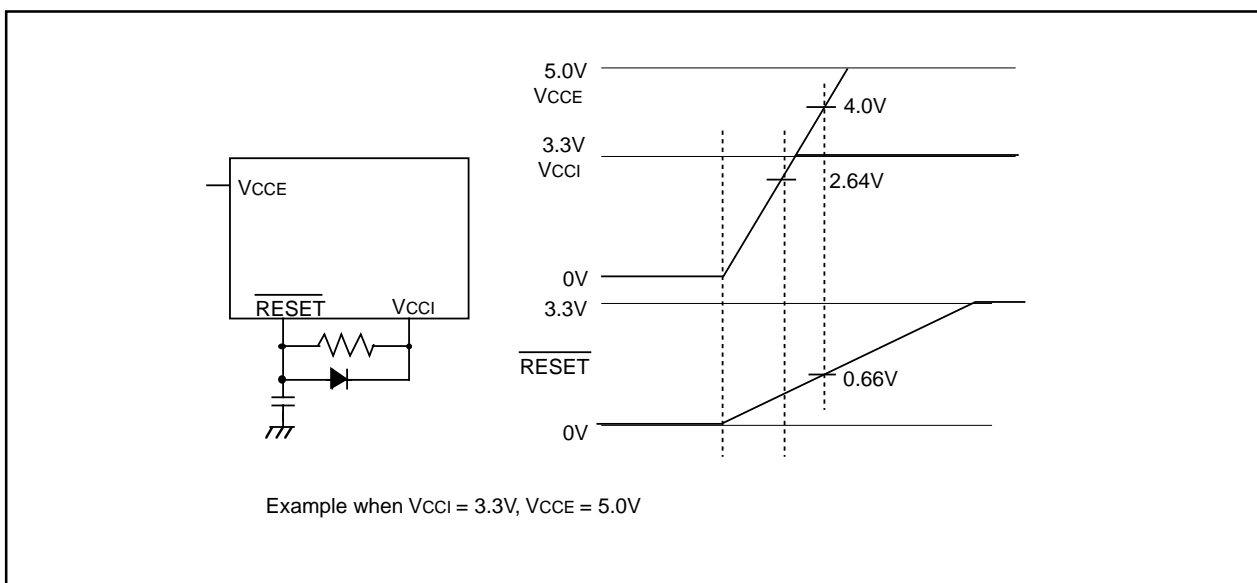


Figure 2.3.1 Example reset circuit

### 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.

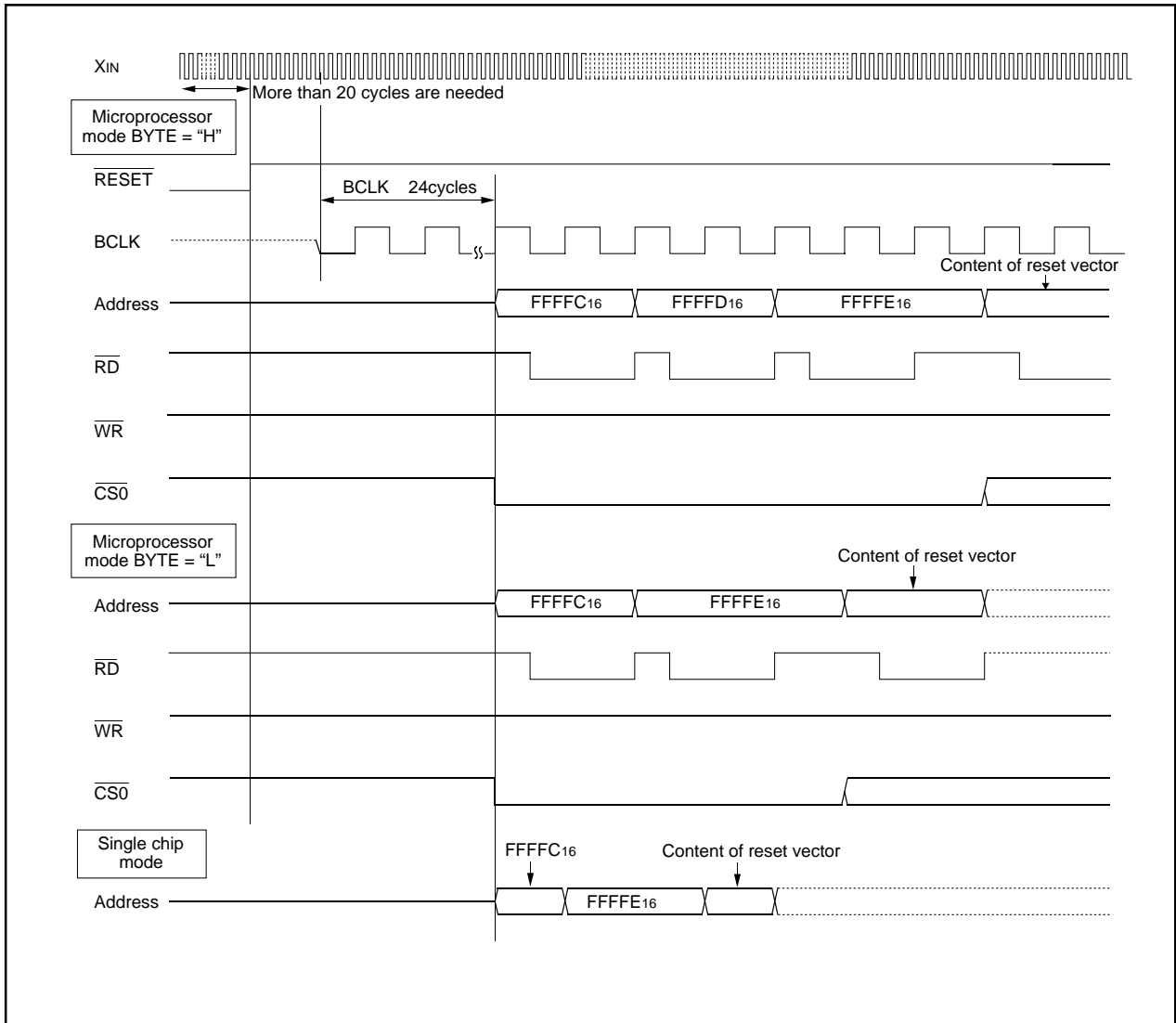


Figure 2.3.2 Reset sequence

### 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, P102 to P107	Input port (floating)	Input port (floating)	Input port (floating)
R, G, B, OUT1, OUT2	Output port		
CVIN1, VHOLD1, HLF1 CVIN2, VHOLD2, HLF2	Input/output port		
OSC1/OSCHLF	Input port		
OSC2	Output port		
HSYNC, VSYNC	Input port		

Processor mode register 0 (Note)	(0004 <sub>16</sub> )...	00 <sub>16</sub>	Timer B0 interrupt control register	(005A <sub>16</sub> )...	XXXXXXXX?000
Processor mode register 1	(0005 <sub>16</sub> )...	000000X00	Timer B1 interrupt control register	(005B <sub>16</sub> )...	XXXXXXXX?000
System clock control register 0	(0006 <sub>16</sub> )...	48 <sub>16</sub>	Timer B2 interrupt control register	(005C <sub>16</sub> )...	XXXXXXXX?000
System clock control register 1	(0007 <sub>16</sub> )...	20 <sub>16</sub>	INT0 interrupt control register	(005D <sub>16</sub> )...	XXXX00?000
Chip select control register	(0008 <sub>16</sub> )...	01 <sub>16</sub>	INT1 interrupt control register	(005E <sub>16</sub> )...	XXXX00?000
Address match interrupt enable register	(0009 <sub>16</sub> )...	XXXXXXXX00	SPRITE OSD control register	(0201 <sub>16</sub> )...	XXXX000000
Protect register	(000A <sub>16</sub> )...	XXXXXXXX0000	OSD control register 1	(0202 <sub>16</sub> )...	00 <sub>16</sub>
Watchdog timer control register	(000F <sub>16</sub> )...	0000????	OSD control register 2	(0203 <sub>16</sub> )...	00 <sub>16</sub>
Address match interrupt register 0	(0010 <sub>16</sub> )...	00 <sub>16</sub>	Horizontal position register	(0204 <sub>16</sub> )...	00 <sub>16</sub>
	(0011 <sub>16</sub> )...	00 <sub>16</sub>	Clock control register	(0205 <sub>16</sub> )...	00 <sub>16</sub>
	(0012 <sub>16</sub> )...	XXXXXXXX0000	I/O polarity control register	(0206 <sub>16</sub> )...	1000000000
Address match interrupt register 1	(0014 <sub>16</sub> )...	00 <sub>16</sub>	OSD control register 3	(0207 <sub>16</sub> )...	00 <sub>16</sub>
	(0015 <sub>16</sub> )...	00 <sub>16</sub>	Raster color register	(0208 <sub>16</sub> )...	00 <sub>16</sub>
	(0016 <sub>16</sub> )...	XXXXXXXX0000		(0209 <sub>16</sub> )...	00 <sub>16</sub>
DMA0 control register	(002C <sub>16</sub> )...	000000?000	OSD reserved register 5	(020A <sub>16</sub> )...	00 <sub>16</sub>
DMA1 control register	(003C <sub>16</sub> )...	000000?000	OSD reserved register 6	(020B <sub>16</sub> )...	00 <sub>16</sub>
OSD1 interrupt control register	(0044 <sub>16</sub> )...	XXXXXXXX?000	OSD reserved register 1	(025D <sub>16</sub> )...	00 <sub>16</sub>
Data slicer 1 interrupt control register	(0045 <sub>16</sub> )...	XXXXXXXX?000	OSD control register 4	(025F <sub>16</sub> )...	XXXXXXXX00
OSD2 interrupt control register	(0048 <sub>16</sub> )...	XXXXXXXX?000	Data slicer 0 control register 1	(0260 <sub>16</sub> )...	00 <sub>16</sub>
Multi-master I <sup>2</sup> C-BUS interface 1 interrupt control register	(0049 <sub>16</sub> )...	XXXX00?000	Data slicer 0 control register 2	(0261 <sub>16</sub> )...	?0?0?0?0?
Bus collision detection interrupt control register	(004A <sub>16</sub> )...	XXXXXXXX?000	Caption data register 01	(0262 <sub>16</sub> )...	???0??0???
DMA0 interrupt control register	(004B <sub>16</sub> )...	XXXXXXXX?000	Caption data register 01	(0263 <sub>16</sub> )...	???0??0???
DMA1 interrupt control register	(004C <sub>16</sub> )...	XXXXXXXX?000	Caption data register 02	(0264 <sub>16</sub> )...	???0??0???
Multi-master I <sup>2</sup> C-BUS interface 0 interrupt control register	(004D <sub>16</sub> )...	XXXXXXXX?000	Caption data register 02	(0265 <sub>16</sub> )...	???0??0???
A-D conversion interrupt control register	(004E <sub>16</sub> )...	XXXXXXXX?000	Caption position register 0	(0266 <sub>16</sub> )...	000?000000
UART2 transmit interrupt control register	(004F <sub>16</sub> )...	XXXXXXXX?000	Slice standard voltage selection register (SBV0)	(0267 <sub>16</sub> )...	00 <sub>16</sub>
UART2 receive interrupt control register	(0050 <sub>16</sub> )...	XXXXXXXX?000	Data slicer 0 reserved register 1	(0268 <sub>16</sub> )...	00 <sub>16</sub>
UART0 transmit interrupt control register	(0051 <sub>16</sub> )...	XXXXXXXX?000	Clock run-in detect register 0	(0269 <sub>16</sub> )...	00 <sub>16</sub>
UART0 receive interrupt control register	(0052 <sub>16</sub> )...	XXXXXXXX?000	Data clock position register 0	(026A <sub>16</sub> )...	XXXX000001
Data slicer 0 interrupt control register	(0053 <sub>16</sub> )...	XXXXXXXX?000	ID1 control register 0	(026B <sub>16</sub> )...	00 <sub>16</sub>
VSYNC interrupt control register	(0054 <sub>16</sub> )...	XXXXXXXX?000	Standard clock detection register 0	(026C <sub>16</sub> )...	XX??0??0???
Timer A0 interrupt control register	(0055 <sub>16</sub> )...	XXXXXXXX?000	CRCC data register 0	(026D <sub>16</sub> )...	XX00000000
Timer A1 interrupt control register	(0056 <sub>16</sub> )...	XXXXXXXX?000	Test reserved register 0	(026E <sub>16</sub> )...	00 <sub>16</sub>
Timer A2 interrupt control register	(0057 <sub>16</sub> )...	XXXXXXXX?000	Reserved register	(026F <sub>16</sub> )...	XXXXXXXXXX0
Timer A3 interrupt control register	(0058 <sub>16</sub> )...	XXXXXXXX?000	Left border control register	(0270 <sub>16</sub> )...	01 <sub>16</sub>
Timer A4 interrupt control register	(0059 <sub>16</sub> )...	XXXXXXXX?000		(0271 <sub>16</sub> )...	XXXXXXXX0000
			Right border control register	(0272 <sub>16</sub> )...	00 <sub>16</sub>
				(0273 <sub>16</sub> )...	XXXXXXXX0000
			SPRITE horizontal position register (high-order)	(0279 <sub>16</sub> )...	XXXXXXXX0000
			OSD reserved register 4	(027A <sub>16</sub> )...	0000000000
			OSD reserved register 3	(027B <sub>16</sub> )...	00 <sub>16</sub>
			OSD reserved register 2	(027C <sub>16</sub> )...	00 <sub>16</sub>
			Peripheral mode register	(027D <sub>16</sub> )...	0XX0000000
			HSYNC counter register	(027E <sub>16</sub> )...	XXXX00XX00

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 03<sub>16</sub> at a reset.

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

Internal oscillation control register 1	(02E116)...	0016	Timer A0 mode register	(039616)...	0016
Internal oscillation control register 2	(02E116)...	0016	Timer A1 mode register	(039716)...	0016
Internal oscillation control register 3	(02E116)...	0016	Timer A2 mode register	(039816)...	0016
ROT reserved register 1	(029E16)...	XXXXXXXXXX0	Timer A3 mode register	(039916)...	0016
ROT reserved register 2	(029F16)...	XXXXXXXXXX0	Timer A4 mode register	(039A16)...	0016
I <sup>2</sup> C0 address register	(02E116)...	0016	Timer B0 mode register	(039B16)...	00?X0000
I <sup>2</sup> C0 status register	(02E216)...	0001000?	Timer B1 mode register	(039C16)...	00?X0000
I <sup>2</sup> C0 control register	(02E316)...	0016	Timer B2 mode register	(039D16)...	00?X0000
I <sup>2</sup> C0 clock control register	(02E416)...	0016	UART0 transmit/receive mode register	(03A016)...	0016
I <sup>2</sup> C0 port selection register	(02E516)...	00??0000	UART0 transmit/receive control register 0	(03A416)...	0816
I <sup>2</sup> C1 address register	(02E916)...	0016	UART0 transmit/receive control register 1	(03A516)...	0216
I <sup>2</sup> C1 status register	(02EA16)...	0001000?	Reserved register	(03AC16)...	0016
I <sup>2</sup> C1 control register	(02EB16)...	0016	Reserved register	(03AD16)...	0016
I <sup>2</sup> C1 clock control register	(02EC16)...	0016	Reserved register	(03AE16)...	0016
I <sup>2</sup> C1 port selection register	(02ED16)...	00??0000	Reserved register	(03AF16)...	0016
Data slicer 1 control register 1	(030016)...	0016	DMA0 request cause select register	(03B816)...	0016
Data slicer 1 control register 2	(030116)...	?0?0??0?	DMA1 request cause select register	(03BA16)...	0016
Caption data register 11	(030216)...	???0??0?	A-D control register 2	(03D416)...	0000??0?
Caption data register 11	(030316)...	???0??0?	A-D control register 0	(03D616)...	0000??0?
Caption data register 12	(030416)...	???0??0?	A-D control register 1	(03D716)...	0016
Caption data register 12	(030516)...	???0??0?	D-A control register	(03DC16)...	0016
Caption position register 1	(030616)...	00?00000	Port P0 direction register	(03E216)...	0016
Slice standard voltage selection register	(030716)...	0016	Port P1 direction register	(03E316)...	0016
Data slicer 1 reserved register 1	(030816)...	0016	Port P2 direction register	(03E616)...	0016
Clock run-in detect register 1	(030916)...	0016	Port P3 direction register	(03E716)...	0016
Data clock position register 1	(030A16)...	X0000000	Port P4 direction register	(03EA16)...	0016
ID1 control register 1	(030B16)...	0016	Port P5 direction register	(03EB16)...	0016
Standard clock detection register 1	(030C16)...	XXXX????	Port P6 direction register	(03EE16)...	0016
CRCC data register 1	(030D16)...	XXXX0000	Port P7 direction register	(03EF16)...	0016
Test reserved register 1	(030E16)...	0016	Port P8 direction register	(03F216)...	00X00000
Reserved register	(030F16)...	XXXXXXXXXX0	Port P9 direction register	(03F316)...	0016
FM Control 3 reserved register	(031116)...	XXXXXXXXXX0	Port P10 direction register	(03F616)...	0016
FM Control 2 reserved register	(031216)...	XXXX?X??0	Pull-up control register 0	(03FC16)...	0016
Flash USER control register	(031316)...	X?000001	Pull-up control register 1(Note)	(03FD16)...	0016
FO Control 3 reserved register	(031516)...	XXXXXXXXXX0	Pull-up control register 2	(03FE16)...	0016
FO Control 2 reserved register	(031616)...	XXXX?X??0	Port control register	(03FF16)...	0016
Flash OSD control register	(031716)...	X?000001	Data registers (R0/R1/R2/R3)		000016
Flash memory change register	(031816)...	X0000000	Address registers (A0/A1)		000016
ID1 Reserved register 0	(031C16)...	00?0??0?	Frame base register (FB)		000016
ID1 Reserved register 1	(031D16)...	00?0??0?	Interrupt table register (INTB)		0000016
Interrupt request cause select register	(035F16)...	0016	User stack pointer (USP)		000016
Reserved register	(037416)...	0016	Interrupt stack pointer (ISP)		000016
Reserved register	(037516)...	0016	Static base register (SB)		000016
Reserved register	(037616)...	0016	Flag register (FLG)		000016
UART2 special mode register	(037716)...	0016			
UART2 transmit/receive mode register	(037816)...	0016			
UART2 transmit/receive control register 0	(037C16)...	0816			
UART2 transmit/receive control register 1	(037D16)...	0216			
Count start flag	(038016)...	0016			
Clock prescaler reset flag	(038116)...	0XXXXXXXXX			
One-shot start flag	(038216)...	00X00000			
Trigger select register	(038316)...	0016			
Up-down flag	(038416)...	0016			

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.

**Notes1:** When the VCC level is applied to the CNVSS pin, it is 0216 at a reset.  
**2:** Flash USER control register (FMRU), Flash OSD control register (FMRD) and Flash memory change register (FMSEL) are only for M306V7FGFP.

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

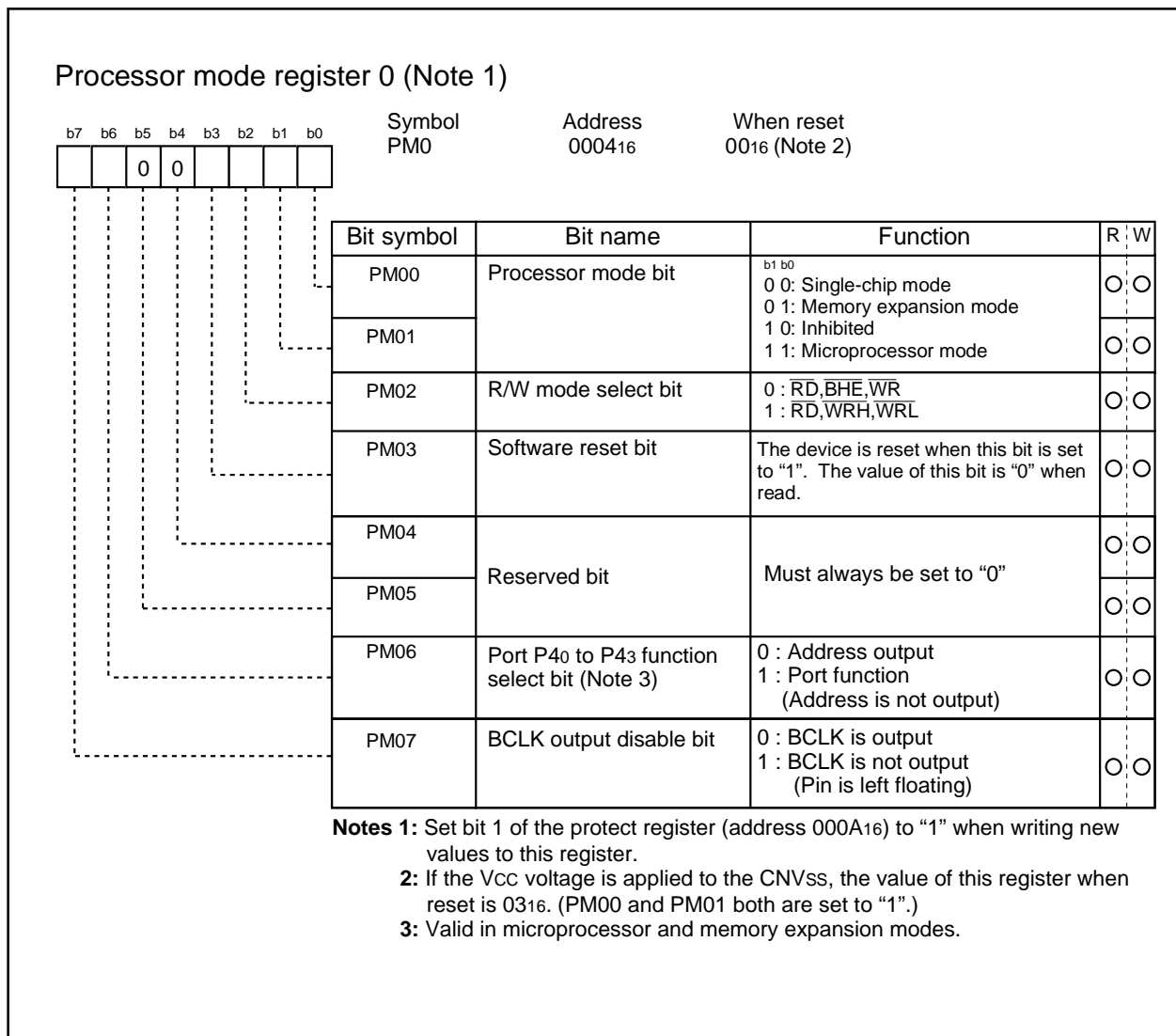
The microcomputer begins operation in single-chip mode after being reset. Memory expansion mode is selected by writing "012" to the processor mode bits.

#### (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**



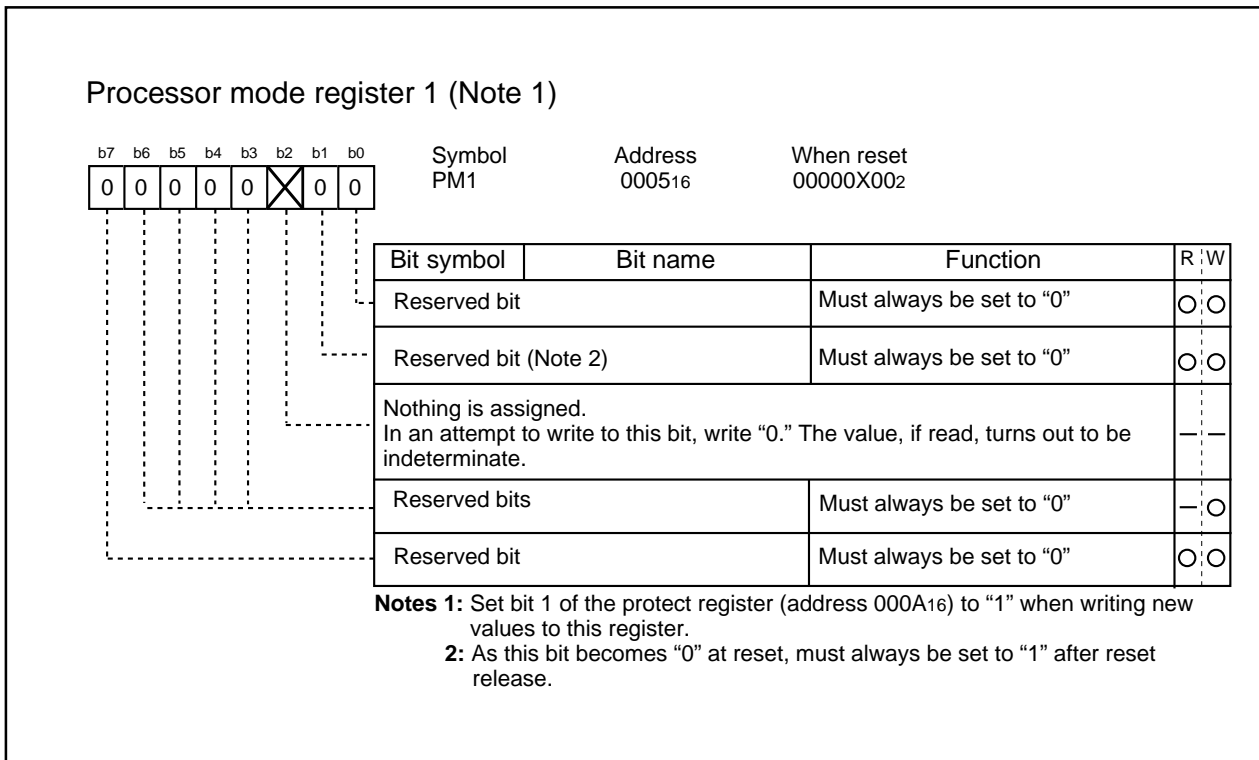
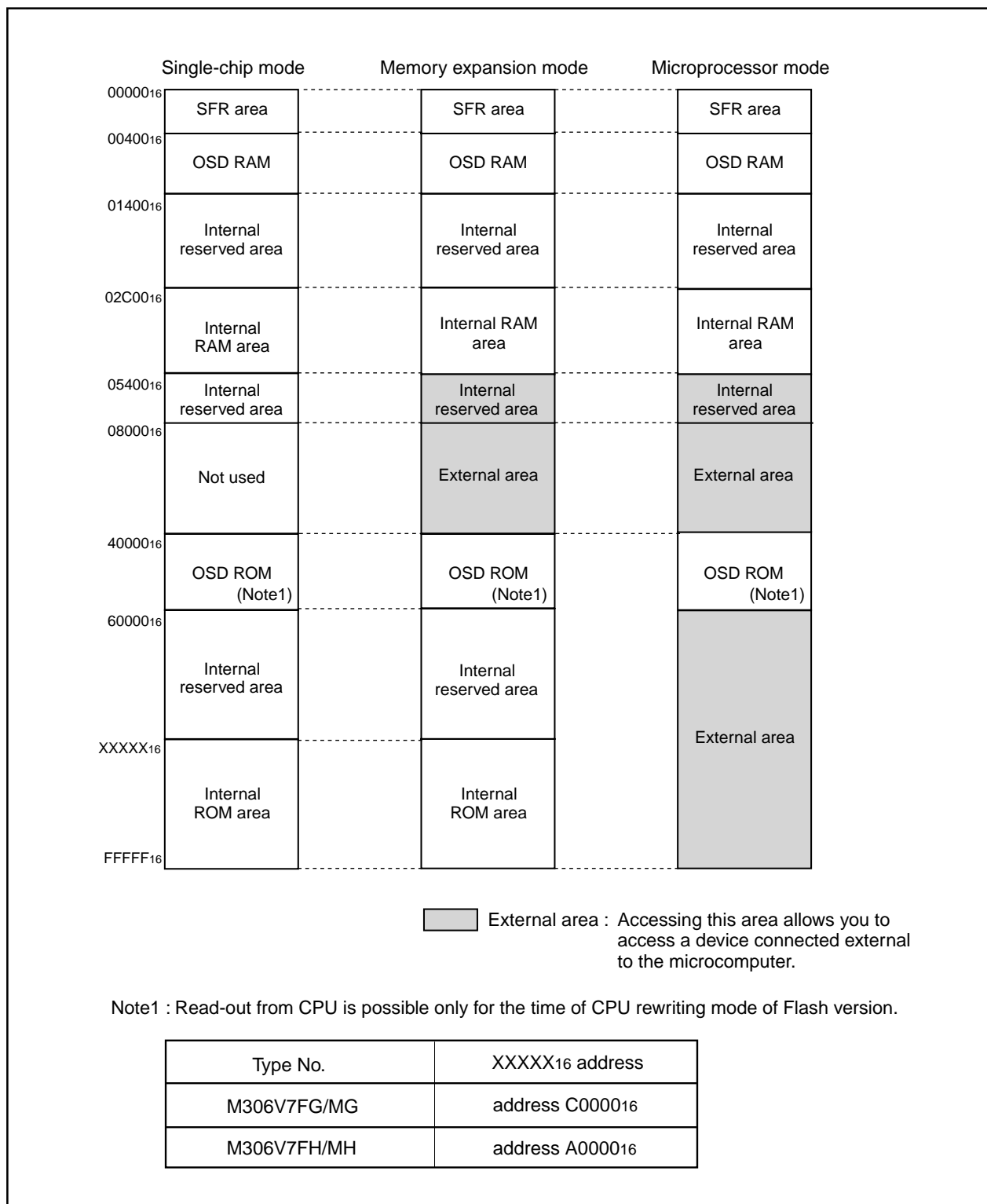


Figure 2.4.2 Processor mode register 1



**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 0004<sub>16</sub>) 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

#### (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. 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) Bus format

The bus format is separate bus .

##### • 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.

**Table 2.4.2 Pin functions for each processor mode**

Processor mode	Single-chip mode	Memory expansion mode/microprocessor modes	
Data bus width BYTE pin level		8 bits = "H"	16 bits = "L"
P00 to P07	I/O port	Data bus	Data bus
P10 to P17	I/O port	I/O port	Data bus
P20	I/O port	Address bus	Address bus
P21 to P27	I/O port	Address bus	Address bus
P30	I/O port	Address bus	Address bus
P31 to P37	I/O port	Address bus	Address bus
P40 to P43 Port P40 to P43 function select bit = 1	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
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	$\overline{HLDA}$	$\overline{HLDA}$
P55	I/O port	$\overline{HOLD}$	$\overline{HOLD}$
P56	I/O port	ALE	ALE
P57	I/O port	$\overline{RDY}$	$\overline{RDY}$

## 2.4.4 Bus Control

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

### (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 P4 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 2.4.3 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
$\overline{CS0}$	18000 <sub>16</sub> to 3FFFF <sub>16</sub> (160K)	18000 <sub>16</sub> to 3FFFF <sub>16</sub> (160K), 60000 <sub>16</sub> to FFFFF <sub>16</sub> (640K)
$\overline{CS1}$	10000 <sub>16</sub> to 17FFF <sub>16</sub> (32K)	10000 <sub>16</sub> to 17FFF <sub>16</sub> (32K)
$\overline{CS2}$	0C000 <sub>16</sub> to 0FFFF <sub>16</sub> (16K)	0C000 <sub>16</sub> to 0FFFF <sub>16</sub> (16K)
$\overline{CS3}$	08000 <sub>16</sub> to 0BFFF <sub>16</sub> (16K)	08000 <sub>16</sub> to 0BFFF <sub>16</sub> (16K)

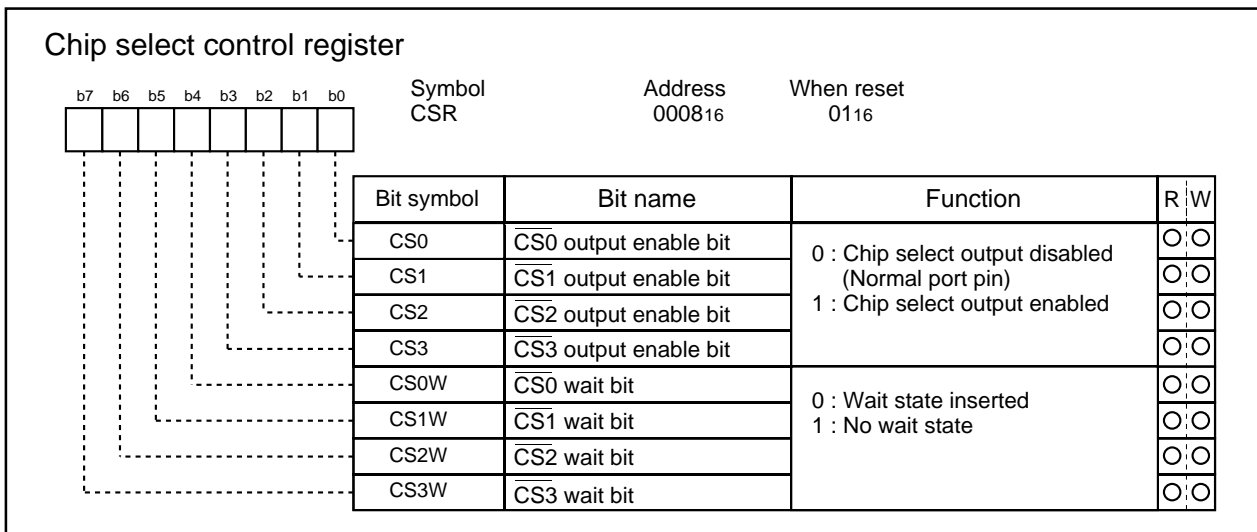


Figure 2.4.4 Chip select control register

(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**

ALE output is indefinite.

**Note:** The output is flouting when reading.

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

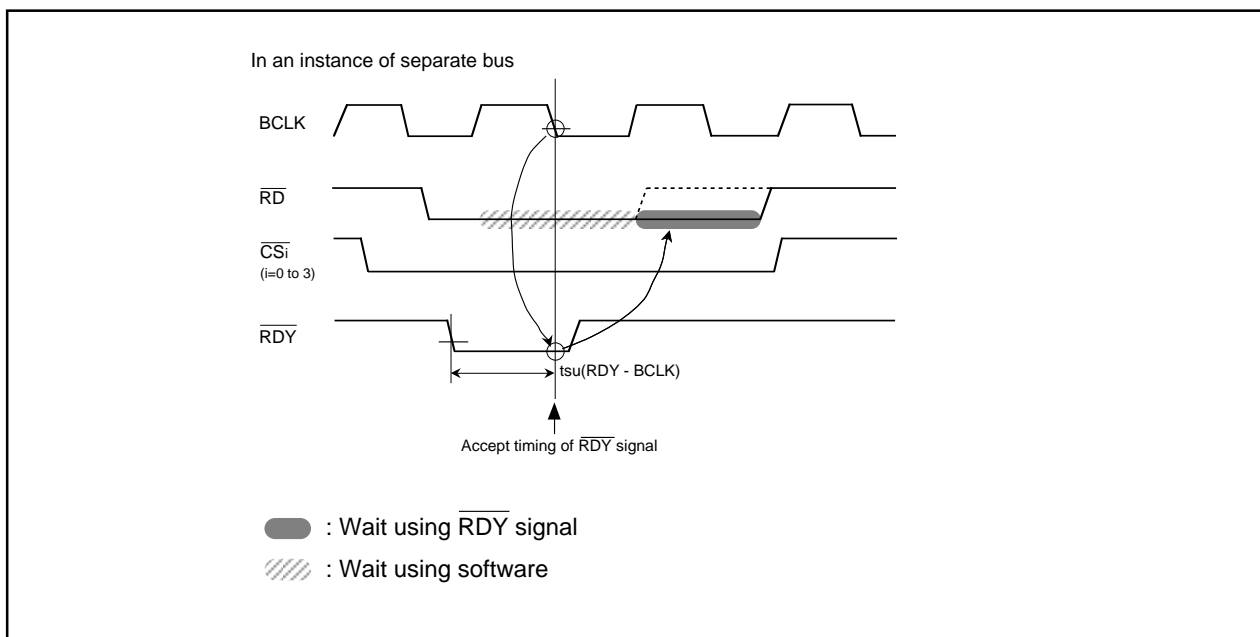
$\overline{RDY}$  signal facilitates access of external devices that require a long time for access. As shown in Figure 2.4.6, if an “L” is being input to the  $\overline{RDY}$  pin at the BCLK falling edge, the bus turns to the wait state. If an “H” is being input to the  $\overline{RDY}$  pin at the BCLK falling edge, the bus cancels the wait state. Table 2.4.6 shows the microcomputer state in the wait state. Figure 2.4.5 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 2.4.5 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 2.4.6 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	Indefinite
$\overline{\text{RD}}$ , $\overline{\text{WR}}$ , $\overline{\text{WRL}}$ , $\overline{\text{WRH}}$		$\overline{\text{RD}}$ , $\overline{\text{WR}}$ , $\overline{\text{WRL}}$ , $\overline{\text{WRH}}$ output	Indefinite
$\overline{\text{BHE}}$		$\overline{\text{BHE}}$ output	Maintain status before accessed status of external area
$\overline{\text{CS}}$		Output “H”	Output “H”
ALE		Indefinite	Indefinite

**(8) BCLK output**

The output of the internal clock  $\phi$  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 bits 4 to 7 of the chip select control register (address 000816).

Software waits can be set independently for each of the 4 areas selected using the chip select signal. Bits 4 to 7 of the chip select control register correspond to chip selects  $\overline{CS0}$  to  $\overline{CS3}$ . When one of these bits is set to "1", the bus cycle is executed in one BCLK cycle. When set to "0", the bus cycle is executed in two 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.

Table 2.4.9 shows the software wait and bus cycles. Figure 2.4.7 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**

Area	Bits 4 to 7 of chip select control register	Bus cycle
SFR/ OSD RAM	Invalid	2 BCLK cycles
Internal ROM/RAM	Invalid	1 BCLK cycle
External memory area	1	1 BCLK cycle
	0	2 BCLK cycles

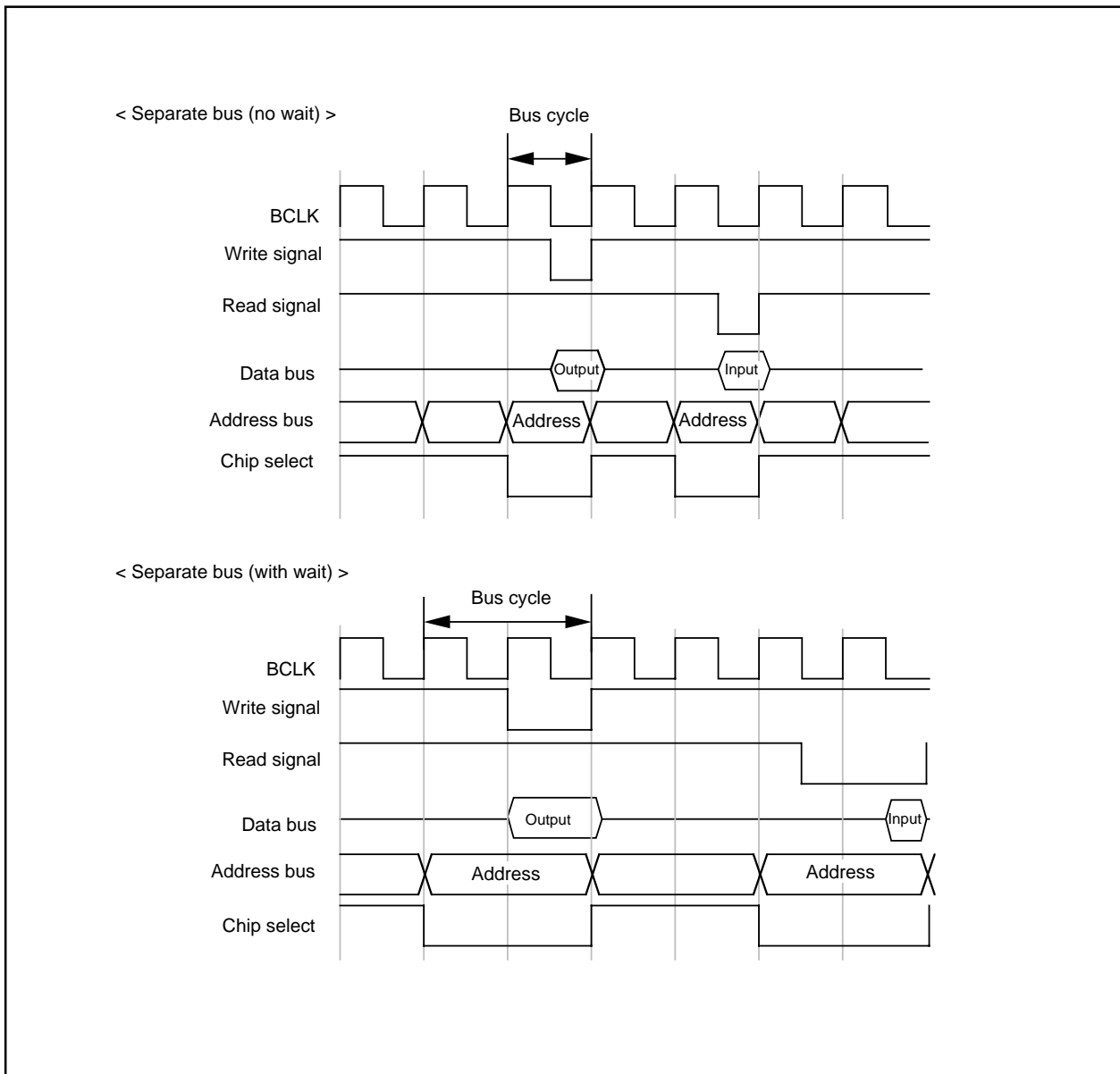


Figure 2.4.7 Typical bus timings using software wait

## 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"> <li>• CPU's operating clock source</li> <li>• Internal peripheral units' operating clock source</li> </ul>	<ul style="list-style-type: none"> <li>• CPU's operating clock source</li> <li>• Timer A/B's count clock source</li> </ul>	<ul style="list-style-type: none"> <li>• OSD's operating clock source</li> </ul>
Usable oscillator	<ul style="list-style-type: none"> <li>• Ceramic resonator (or quartz-crystal oscillator)</li> </ul>	<ul style="list-style-type: none"> <li>• Quartz-crystal oscillator</li> </ul>	<ul style="list-style-type: none"> <li>• Ceramic resonator (or quartz-crystal oscillator)</li> <li>• LC oscillator (Note)</li> </ul>
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		

**Note:** The OSD clock can be selected between an external resonator and an internal oscillator circuit. For details, see a description of the Clock Control Register (address 0205<sub>16</sub>).

## 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.

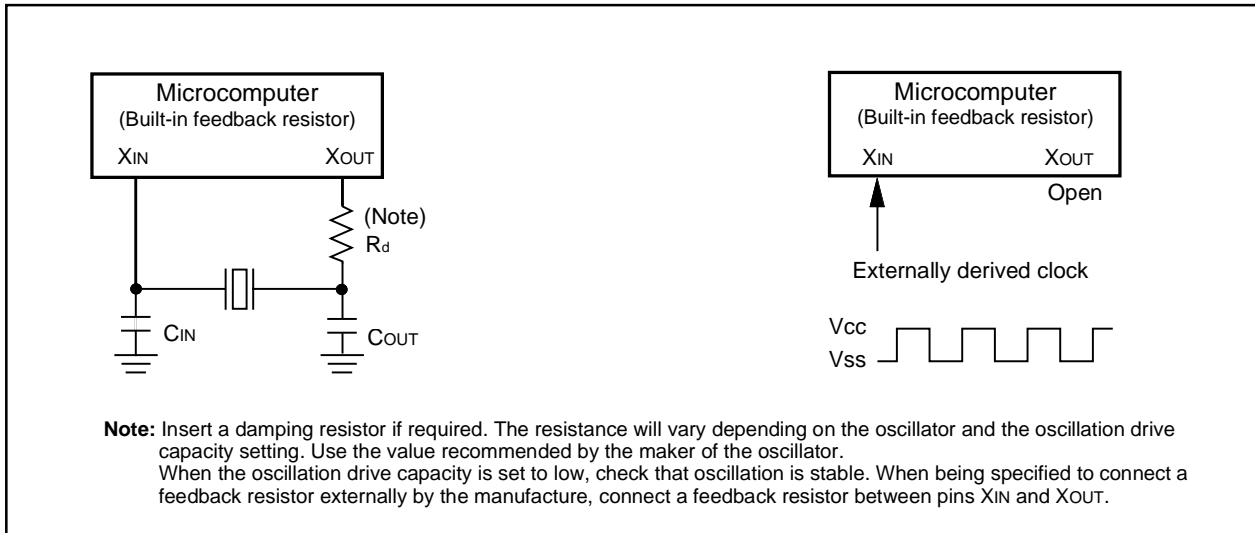


Figure 2.5.1 Examples of main clock

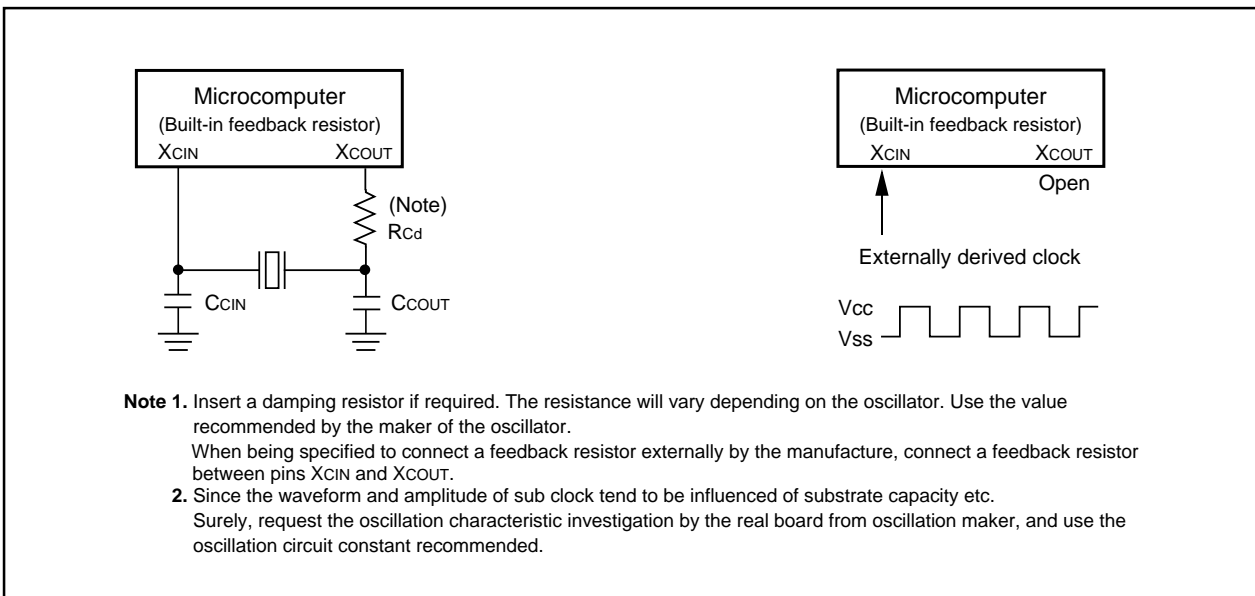


Figure 2.5.2 Examples of sub-clock

### 2.5.2 OSD Oscillation Circuit

The OSD clock oscillation circuit can be chosen to be an external oscillator circuit comprised of an LC oscillator or a ceramic resonator (or a quartz-crystal oscillator) connected between the OSC1 and OSC2 pins, or an internal oscillator circuit with a filter connected to the OSC1 pin. 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).

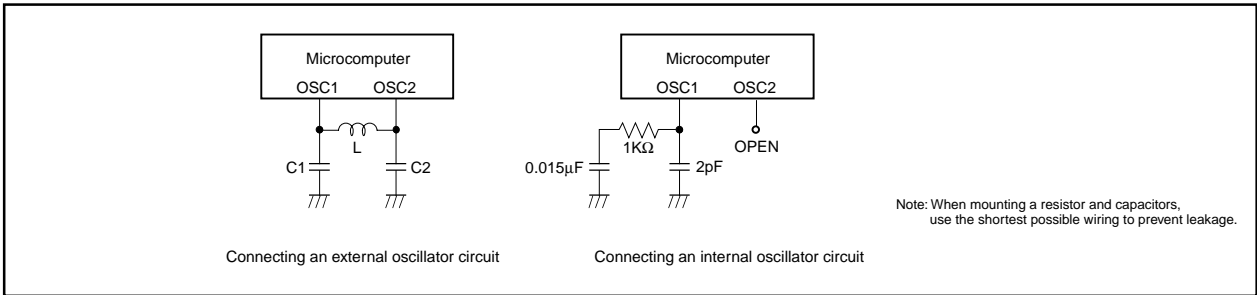


Figure 2.5.3 OSD clock connection example

### 2.5.3 Clock Control

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

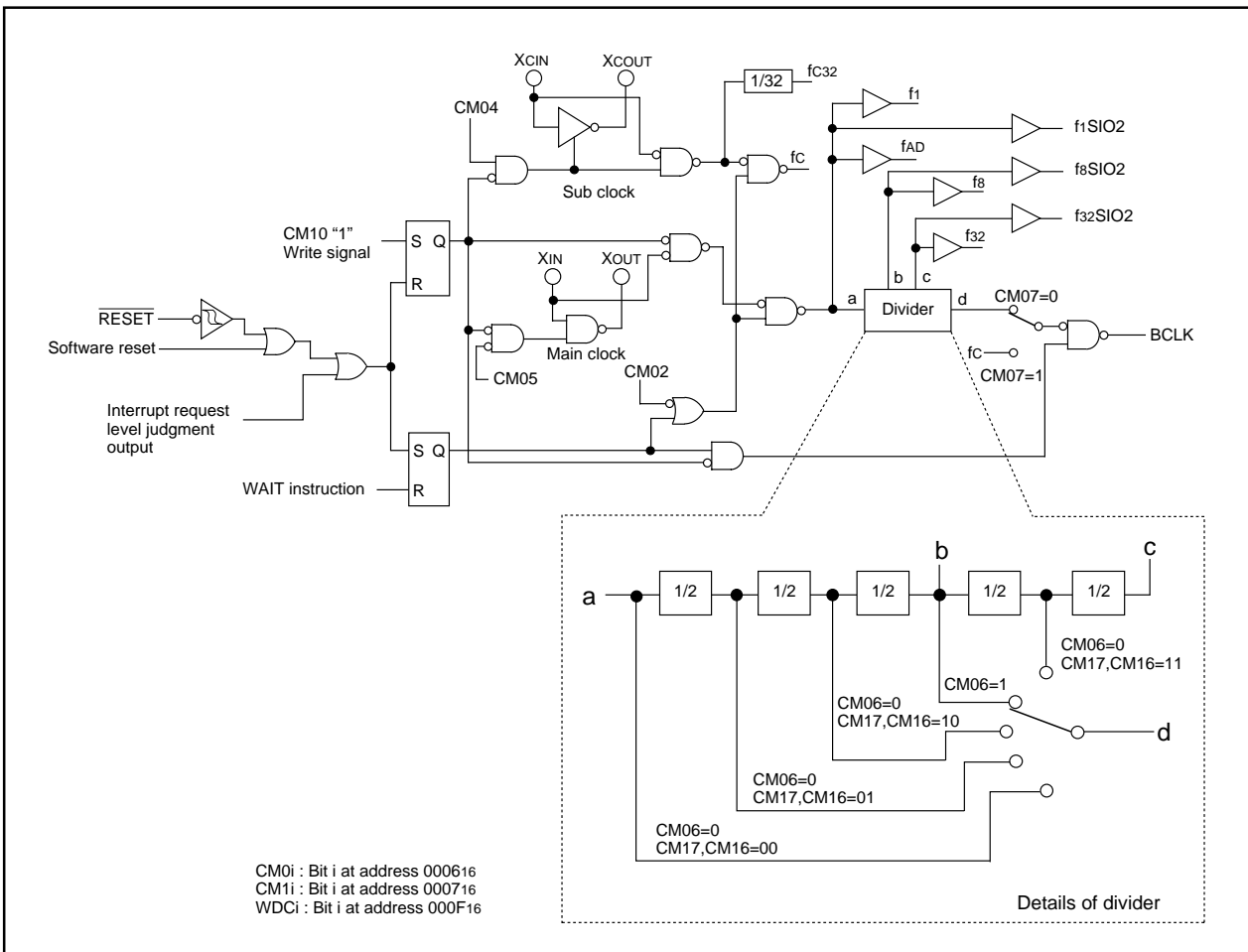


Figure 2.5.4 Clock generating circuit

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<sub>IN</sub>-X<sub>OUT</sub> 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<sub>c</sub> 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.

**(3) BCLK**

The internal clock  $\phi$  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 (f<sub>1</sub>, f<sub>8</sub>, f<sub>32</sub>, f<sub>1SIO2</sub>, f<sub>8SIO2</sub>, f<sub>32SIO2</sub>, f<sub>AD</sub>)**

The clock for the peripheral devices is derived 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.

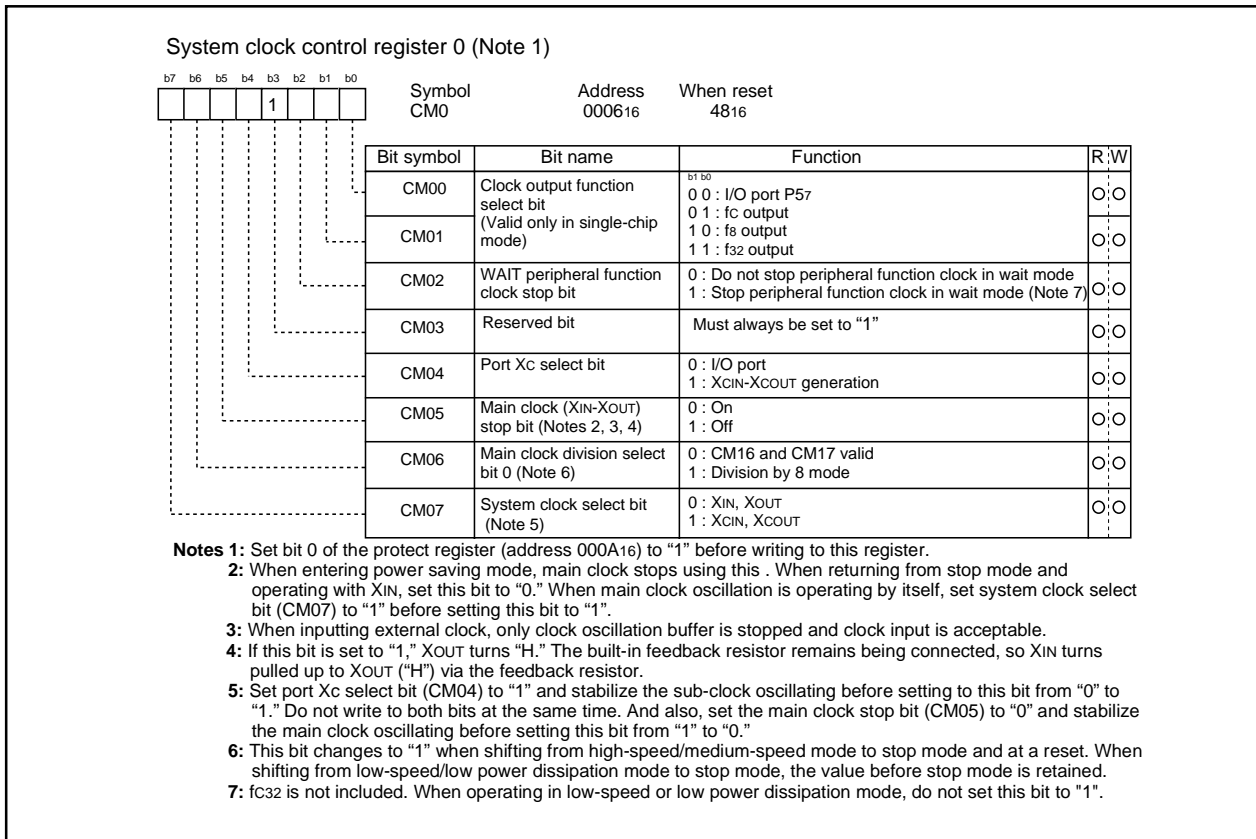
**(5) f<sub>c32</sub>**

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

**(6) f<sub>c</sub>**

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

Figures 2.5.5 and 2.5.6 shows the system clock control registers 0 and 1.



Figures 2.5.5 System clock control register 0

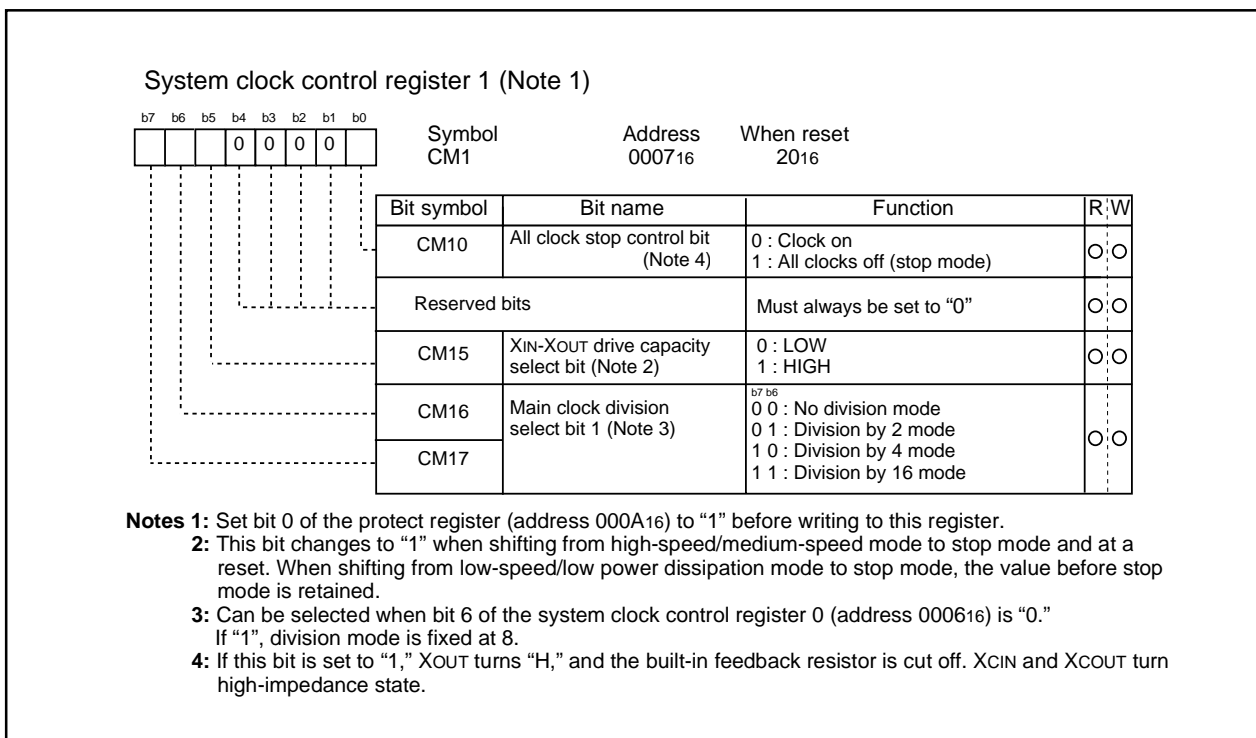


Figure 2.5.6 System clock control register 1



## 2.5.4 Clock Output

In single-chip mode, the clock output function select bits (bits 0 and 1 at address 0006<sub>16</sub>) enable f<sub>8</sub>, f<sub>32</sub>, or f<sub>c</sub> to be output from the P57/CLKOUT pin. When the WAIT peripheral function clock stop bit (bit 2 at address 0006<sub>16</sub>) is set to "1," the output of f<sub>8</sub> and f<sub>32</sub> 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 0007<sub>16</sub>) stops all oscillation and the microcomputer enters stop mode. In stop mode, the content of the internal RAM is retained provided that V<sub>CC</sub> remains above 3.0V.

Because the oscillation, BCLK, f<sub>1</sub> to f<sub>32</sub>, f<sub>1</sub>SIO<sub>2</sub> to f<sub>32</sub>SIO<sub>2</sub>, f<sub>c</sub>, f<sub>c</sub>32, and f<sub>AD</sub> 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<sub>i</sub> (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 0006<sub>16</sub>) 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	/
RD, WR, BHE, WRL, WRH		"H"	
$\overline{HLDA}$ , BCLK		"H"	
ALE		Unfixed	
Port		Retains status before stop mode	Retains status before stop mode
CLKOUT	When f <sub>c</sub> selected	Valid only in single-chip mode	"H"
	When f <sub>8</sub> , f <sub>32</sub> 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. However, peripheral function clock fc32 does not stop so that the peripherals using fc32 do not contribute to the power saving. When the MCU running in low-speed or low power dissipation mode, do not enter WAIT mode with this bit set to "1".

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, WRL, WRH		"H"	
HLDA, BCLK		"H"	
ALE		Unfixed	
Port		Retains status before wait mode	
CLKOUT	When fc selected	Valid only in single-chip mode	Does not stop
	When f8, f32 selected	Valid only in single-chip mode	Does not stop when the WAIT peripheral function clock stop bit is "0". When the WAIT peripheral function clock stop bit is "1", the status immediately prior to entering wait mode is maintained.

## 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  $\phi$ .

### (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 XIN and XCIN, 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.

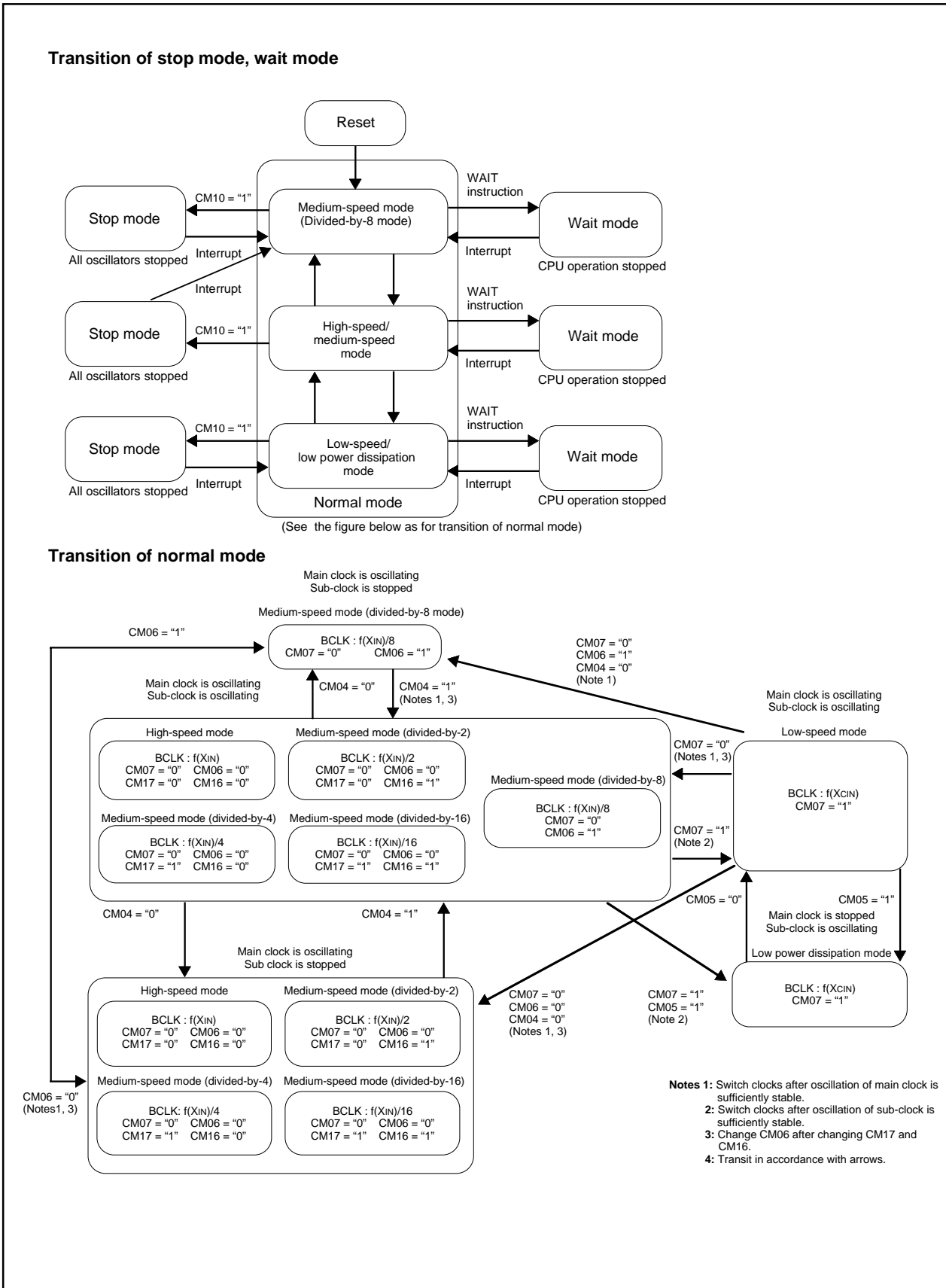
#### (2) Wait mode

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 0004<sub>16</sub>), processor mode register 1 (address 0005<sub>16</sub>), system clock control register 0 (address 0006<sub>16</sub>), system clock control register 1 (address 0007<sub>16</sub>) and port P9 direction register (address 03F3<sub>16</sub>) can only be changed when the respective bit in the protect register is set to "1". Therefore, important outputs can be allocated to port P9.

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

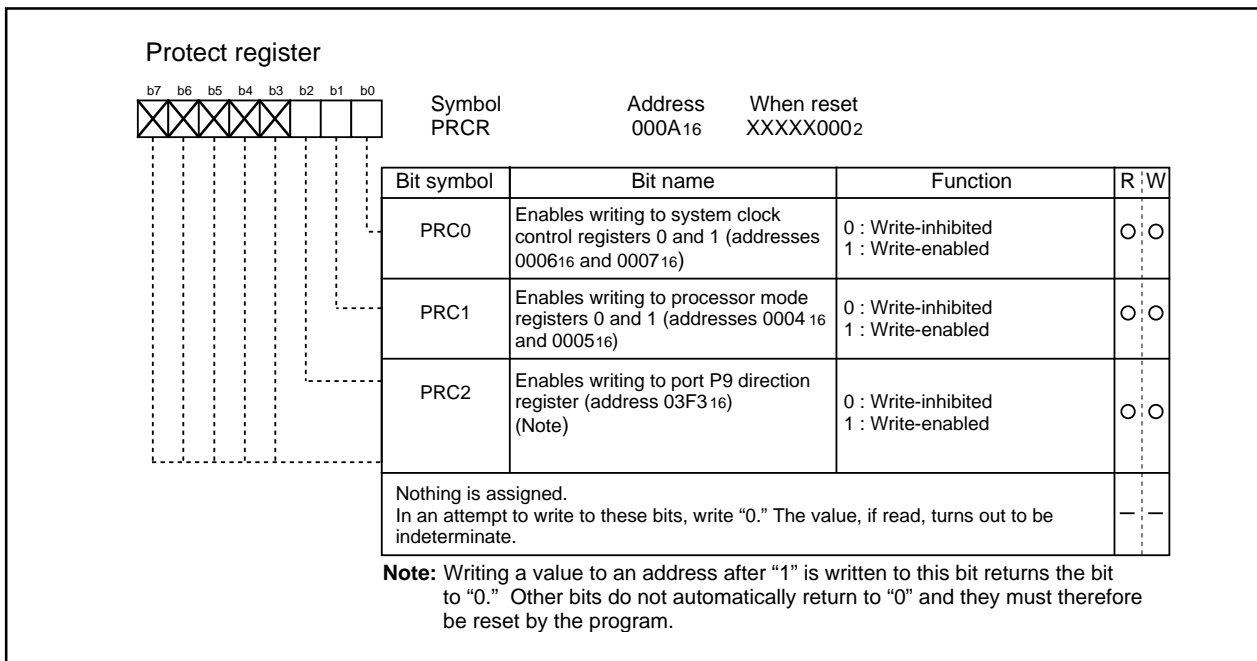
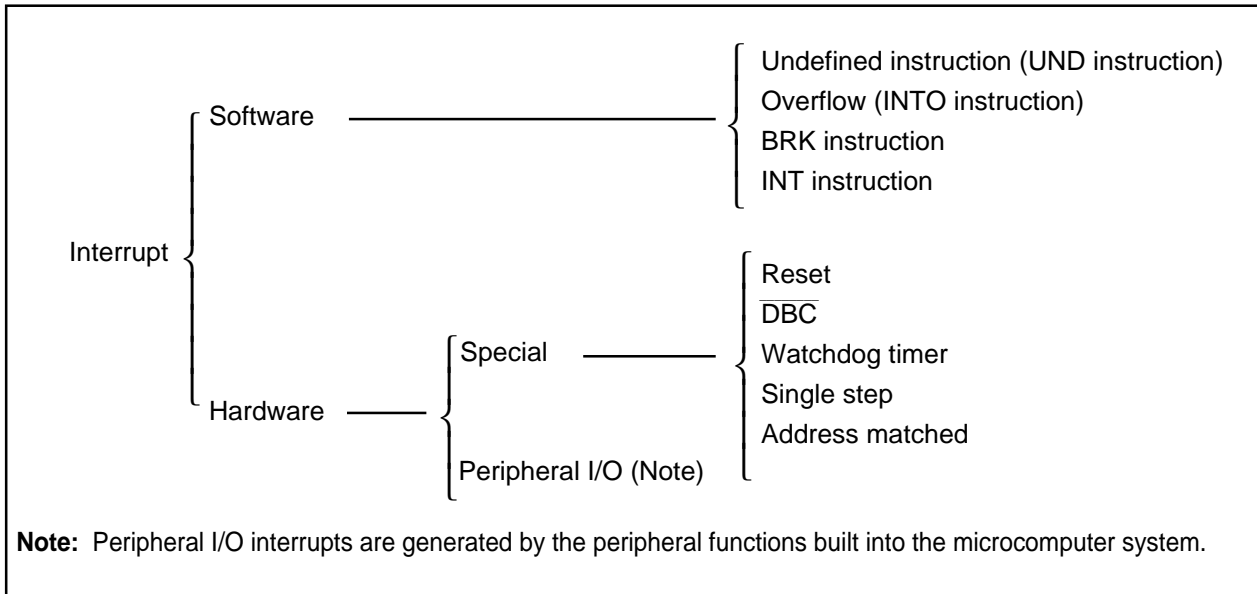


Figure 2.6.1 Protect register

## 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.

#### (2) Peripheral I/O interrupts

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

- **Bus collision detection interrupt**

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

- **DMA0 interrupt, DMA1 interrupt**

These are interrupts DMA generates.

- **V<sub>SYNC</sub> interrupt**

V<sub>SYNC</sub> interrupt occurs if a V<sub>SYNC</sub> 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 I<sup>2</sup>C-BUS interface 0 and multi-master I<sup>2</sup>C-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.

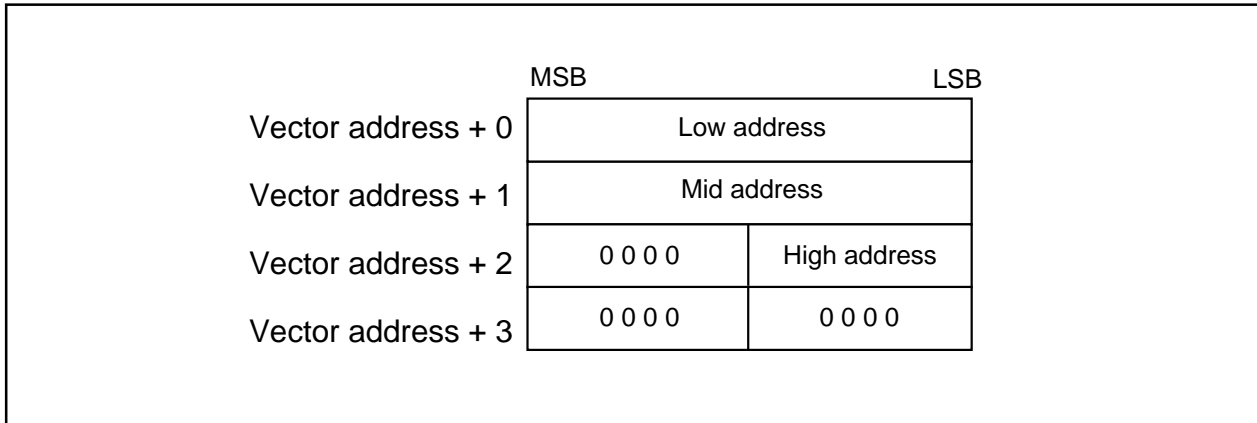
- **Data slicer 0 interrupt and Data slicer 1 interrupt**

These are interrupts 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 FFFDC<sub>16</sub> to FFFFF<sub>16</sub>. One vector table comprises four bytes. Set the first address of interrupt routine in each vector table. Table 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 <sub>16</sub> to FFFD <sub>16</sub>	Interrupt on UND instruction
Overflow	FFFE <sub>0</sub> <sub>16</sub> to FFFE <sub>3</sub> <sub>16</sub>	Interrupt on INTO instruction
BRK instruction	FFFE <sub>4</sub> <sub>16</sub> to FFFE <sub>7</sub> <sub>16</sub>	If the vector is filled with FF <sub>16</sub> , program execution starts from the address shown by the vector in the variable vector table
Address match	FFFE <sub>8</sub> <sub>16</sub> to FFFE <sub>B</sub> <sub>16</sub>	There is an address-matching interrupt enable bit
Single step (Note)	FFFE <sub>C</sub> <sub>16</sub> to FFFE <sub>F</sub> <sub>16</sub>	Do not use
Watchdog timer	FFFF <sub>0</sub> <sub>16</sub> to FFFF <sub>3</sub> <sub>16</sub>	
DBC (Note)	FFFF <sub>4</sub> <sub>16</sub> to FFFF <sub>7</sub> <sub>16</sub>	Do not use
Reserved source	FFFF <sub>8</sub> <sub>16</sub> to FFFF <sub>B</sub> <sub>16</sub>	Do not use
Reset	FFFF <sub>C</sub> <sub>16</sub> to FFFF <sub>F</sub> <sub>16</sub>	

**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 FFFDC<sub>16</sub> to FFFF<sub>16</sub>. 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)	Data slicer 1	
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 <sup>2</sup> 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 <sup>2</sup> 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 slicer0	
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.

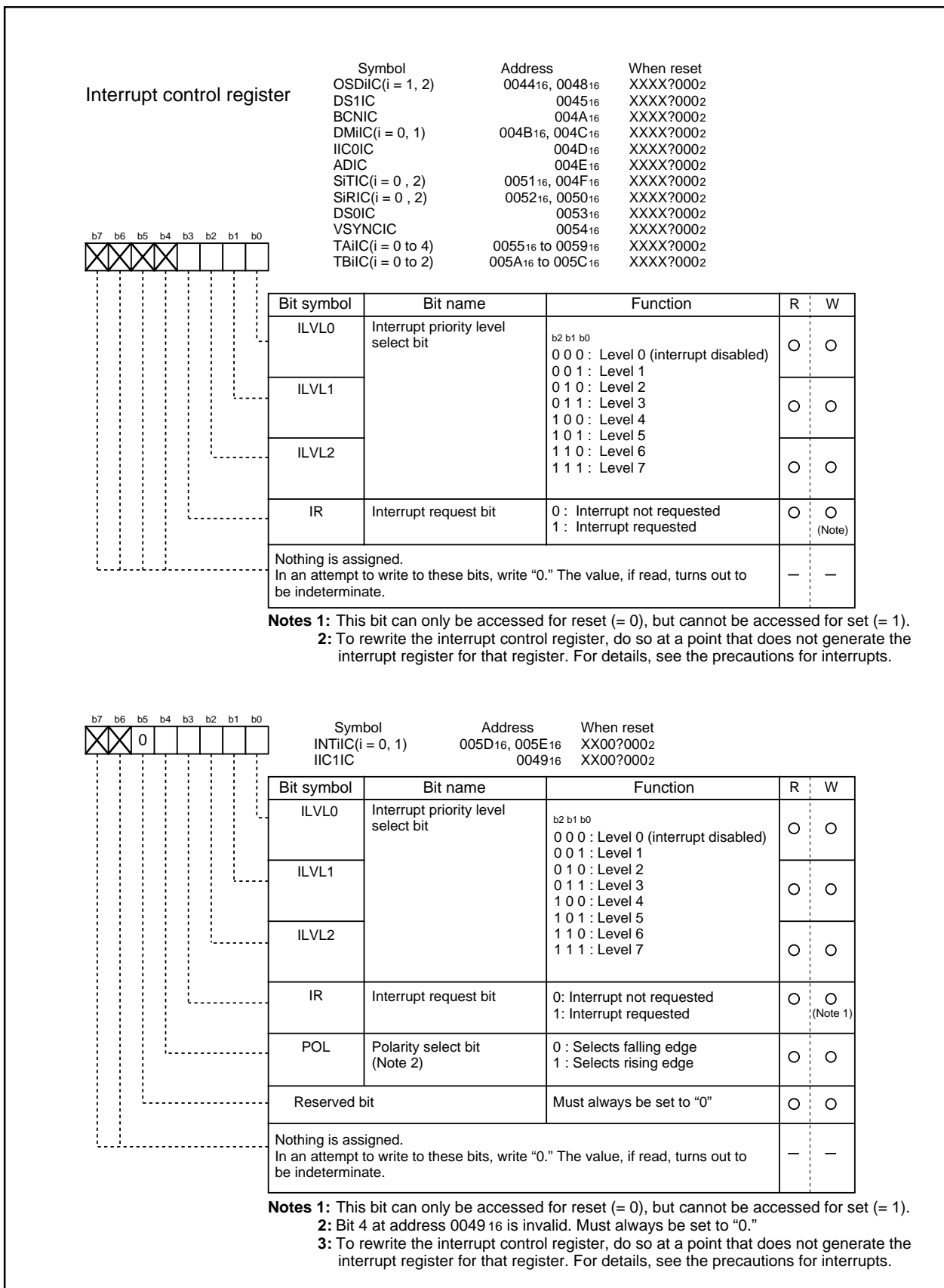


Figure 2.7.3 Interrupt control registers

## 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.

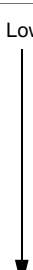
Table 2.7.3 shows the settings of interrupt priority levels and Table 2.7.4 shows the interrupt levels enabled, according to the consist of the IPL.

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

## 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 00000<sub>16</sub>.
- (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.

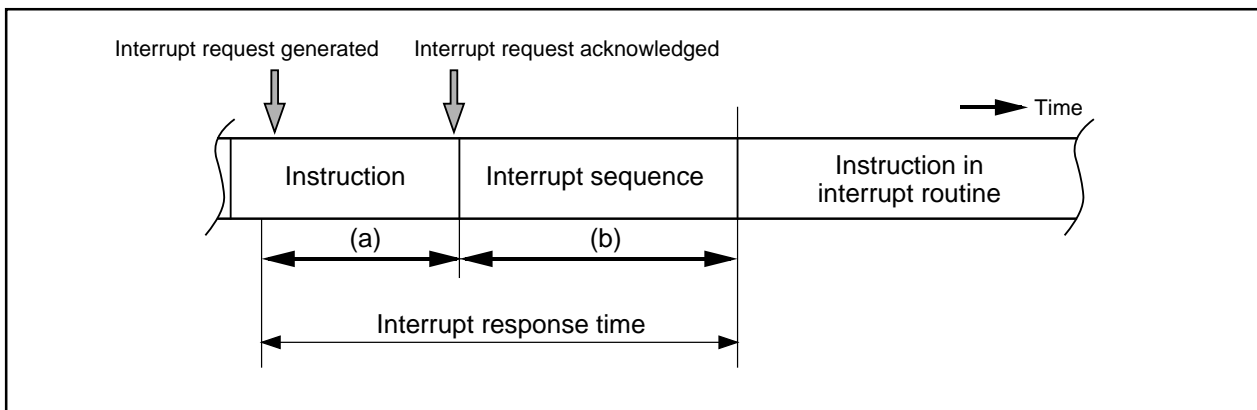


Figure 2.7.4 Interrupt response time

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

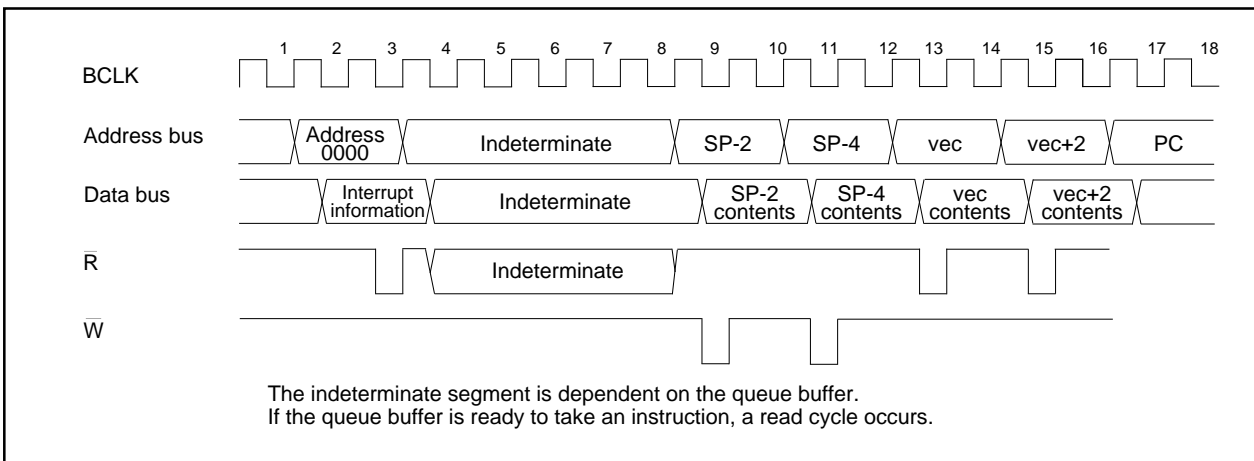
Time (b) is as shown in Table 2.7.5.

**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).

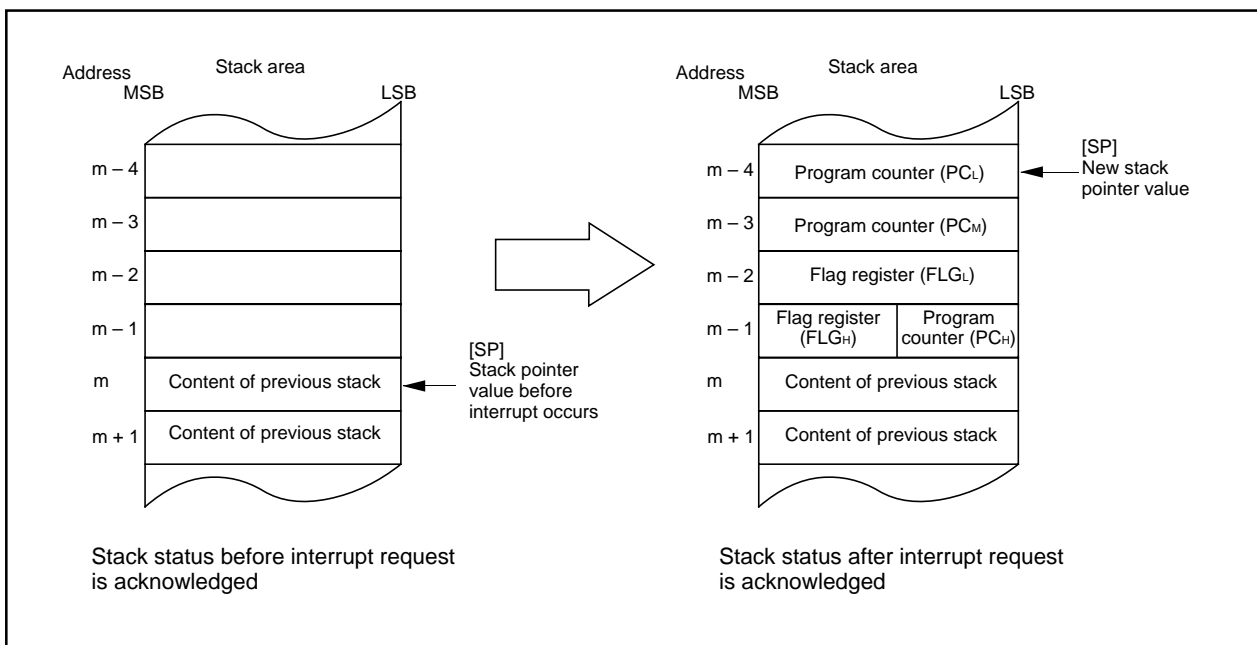
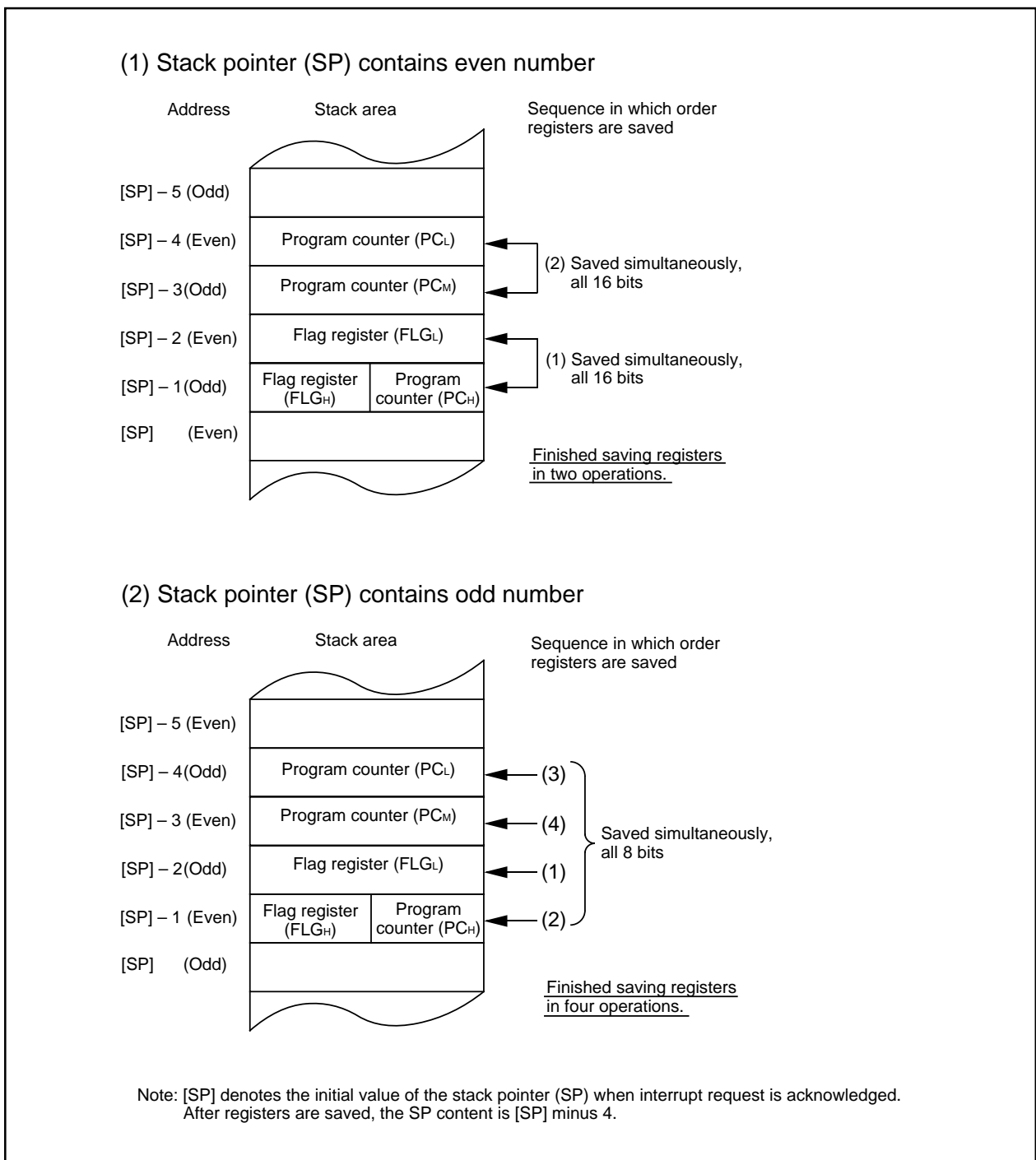


Figure 2.7.6 State of stack before and after acceptance of interrupt request

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.

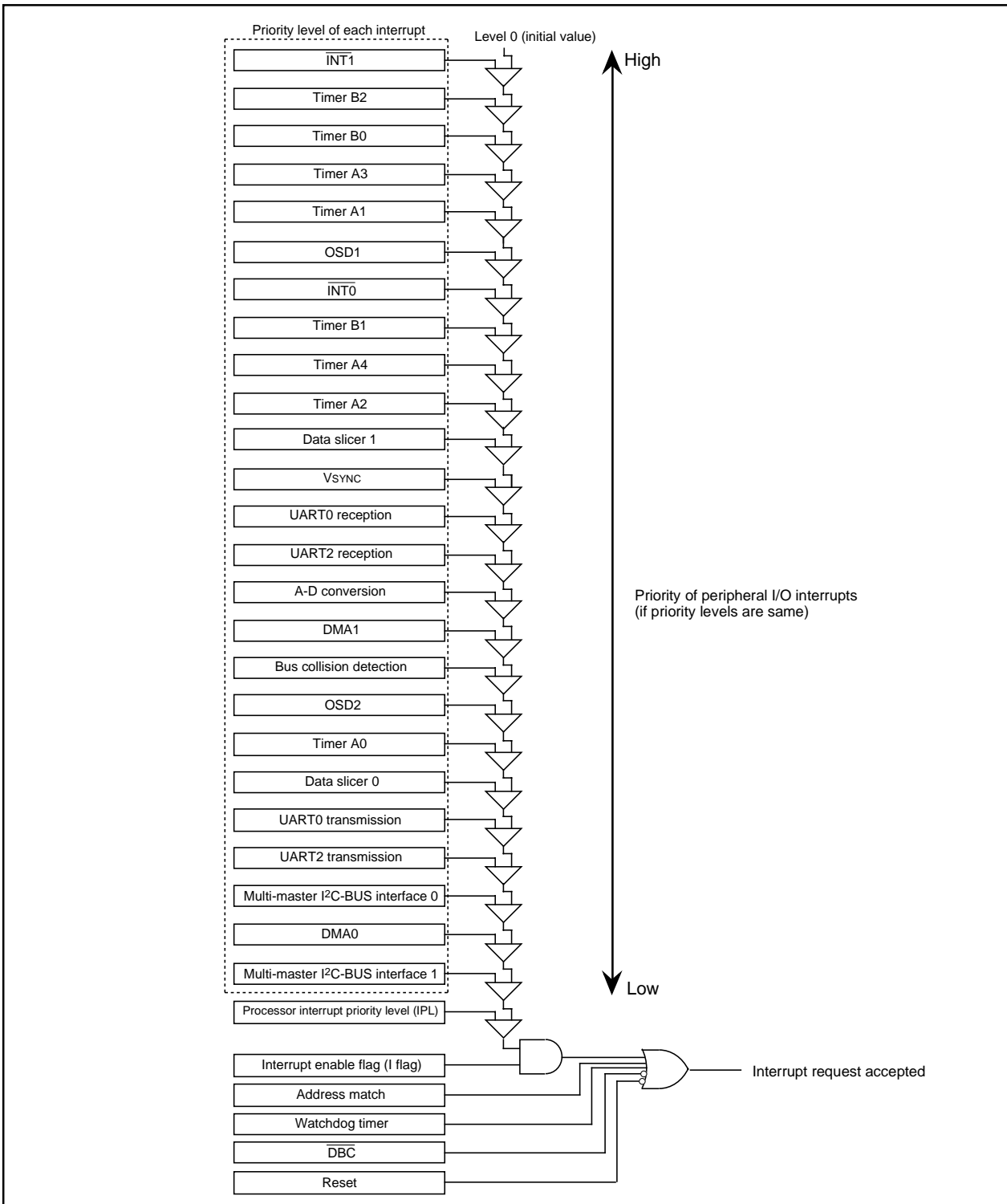
### 2.7.16 Interrupt Priority Level Resolution Circuit

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 (035F<sub>16</sub>). 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.

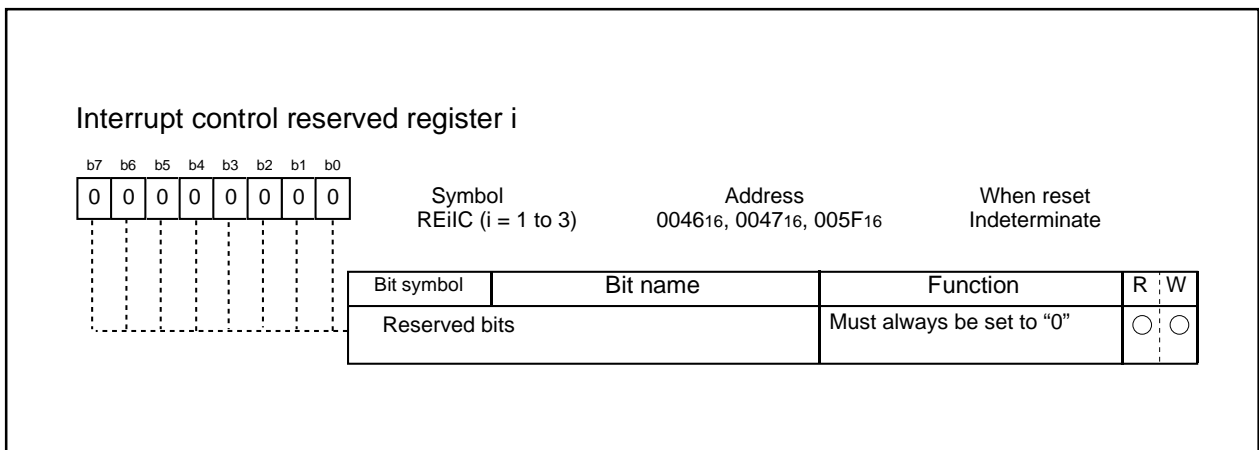


Figure 2.7.10 Interrupt control reserved register i (i = 0 to 3)

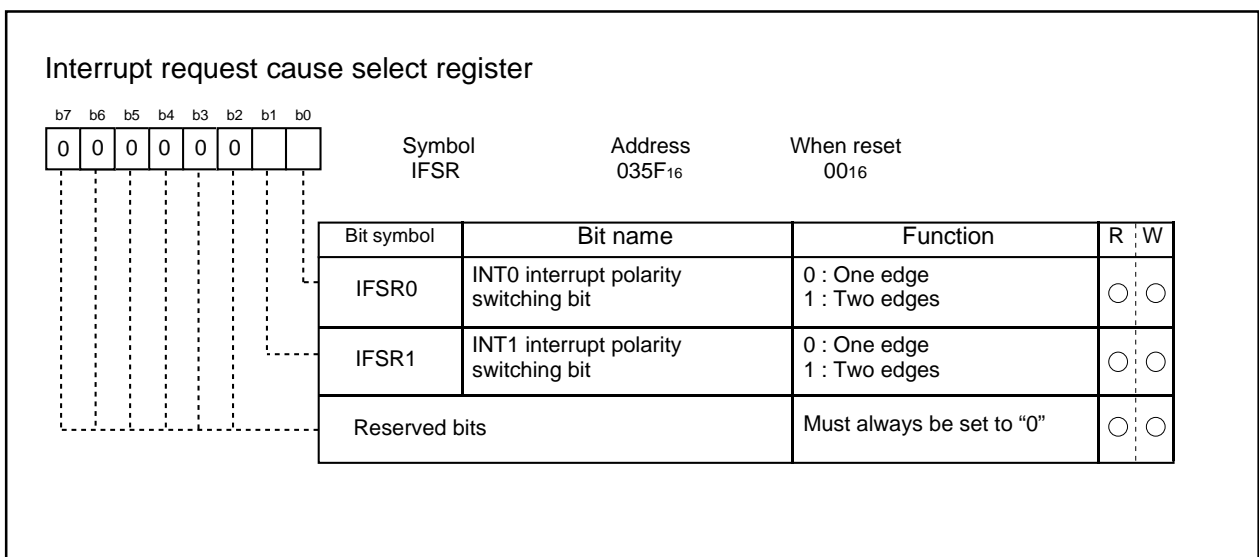
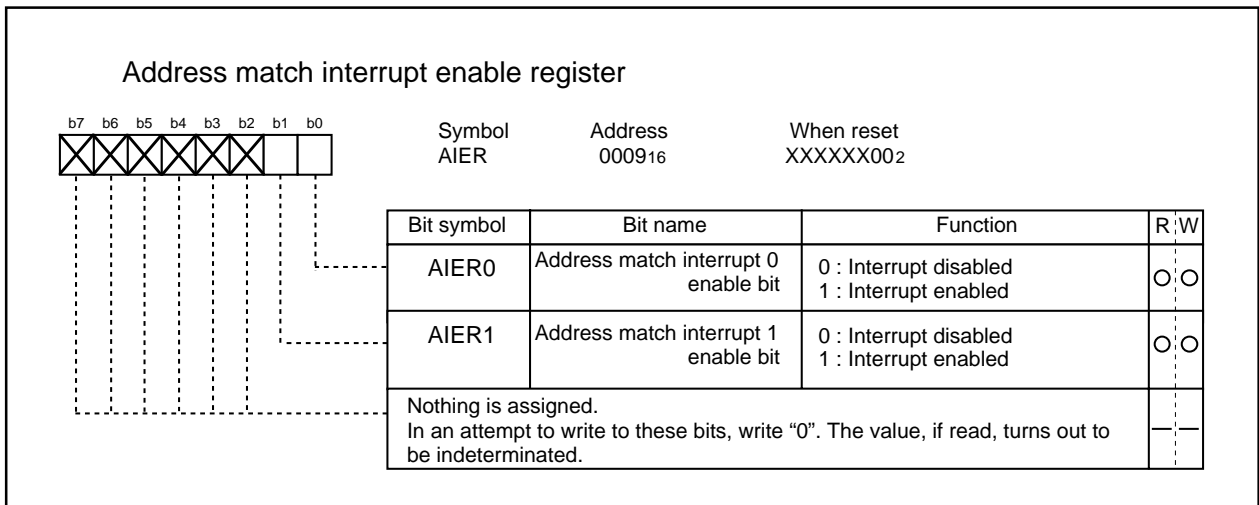


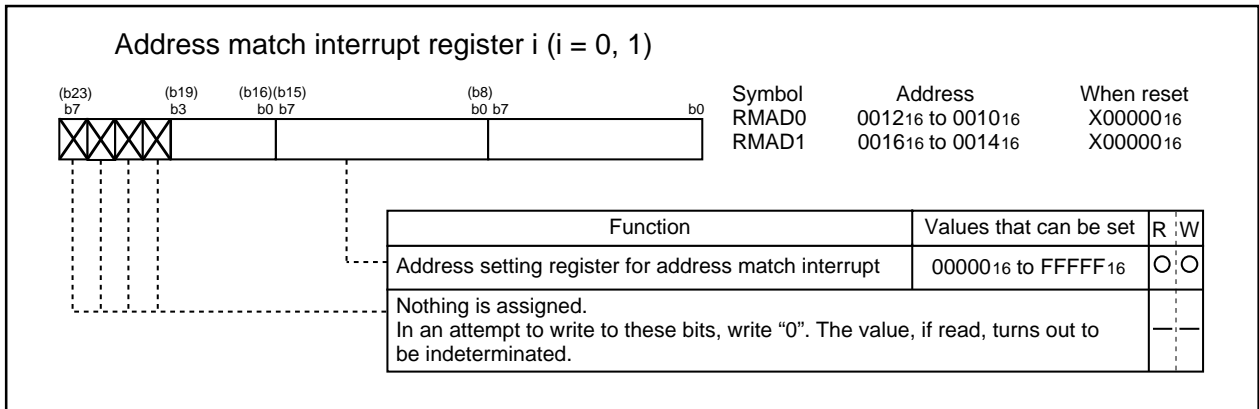
Figure 2.7.11 Interrupt request cause select register

### 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 00000<sub>16</sub>

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

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

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

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

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

### (2) Setting the stack pointer

- The value of the stack pointer immediately after reset is initialized to 0000<sub>16</sub>. Accepting an interrupt before setting a value in the stack pointer may become a factor of runaway. Be sure to set a value in the stack pointer before accepting an interrupt.

### (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.

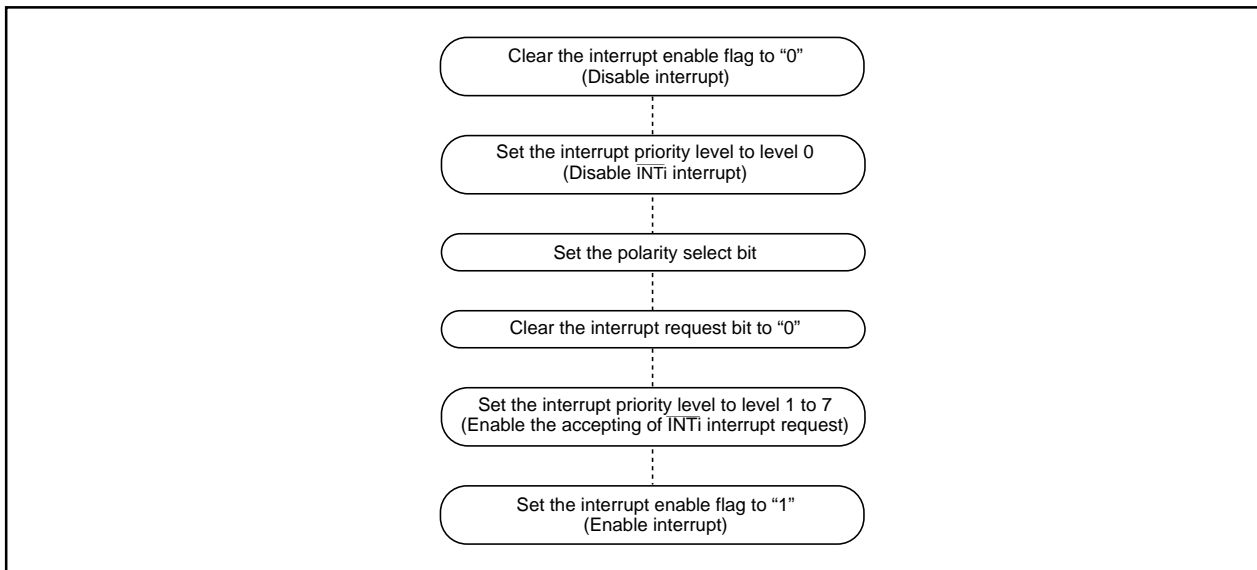


Figure 2.7.14 Switching condition of INT interrupt request

#### (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

**(5) Notes**

When clearing interrupt request bit of interrupt control register, depending on the instruction to be used, it interrupts and interrupt request bit may not be cleared. Use MOV command, when clear interrupt request bit, and change interrupt control register. When change interrupt control register in M16C/60 series and M16C/20 series, interrupt control register be sure to change in the part which corresponding interrupt request does not generate, and change interrupt control register after changing interruption into a prohibition state.

The example of a program which clears interrupt request bit in M16C/60 series.

Example 1: The case where interrupt control register is rewritten with a immediate value

```
FCLR    I                ; Interrupt is forbidden
MOV.B   #00H,0055H      ; Timer A0 interrupt request bit clear
MOV.W   MEM,R0          ; Dummy read
FSET    I                ; Interrupt is permitted
```

Example 2: The case where only interrupt request bit is cleared

```
FCLR    I                ; Interrupt is forbidden
MOV.B   0055H,R0L       ; Timer A0 interrupt control register read-out
AND.B   #0F7H,R0L       ; Only timer A0 interruption request bit is clear
MOV.B   R0L,0055H       ; Timer A0 interrupt control register writing
MOV.W   MEM,R0          ; Dummy read
FSET    I                ; Interrupt is permitted
```

There is a dummy read in Example 1 and Example 2 for preventing the set of interrupt permission flag (I flag) interrupting under the influence of command cue, and performing before the writing of interrupt control register.

## 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 16 MHz and that 16 has been chosen for the dividing ratio of the pre-scaler, then the watchdog timer's period becomes approximately 32.8 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.

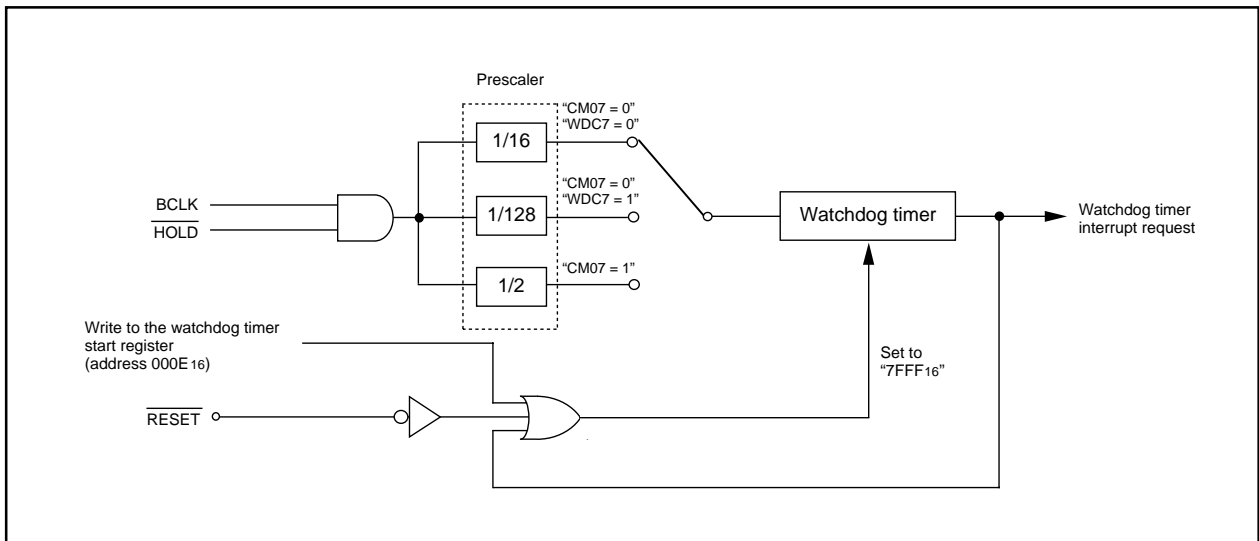


Figure 2.8.1 Block diagram of watchdog timer

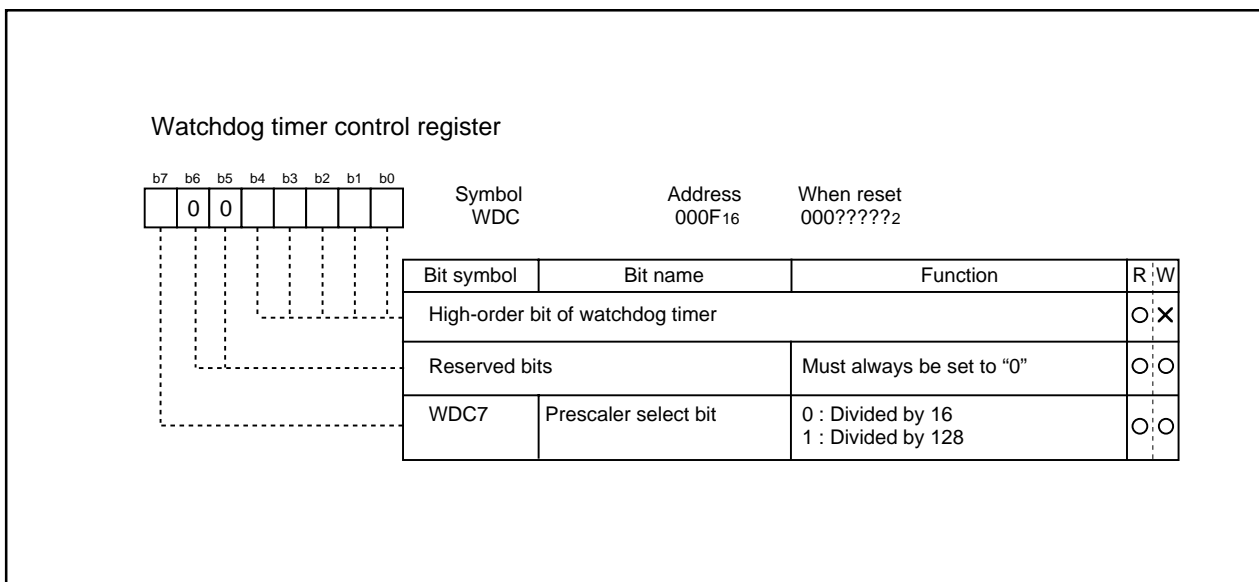


Figure 2.8.2 Watchdog timer control register

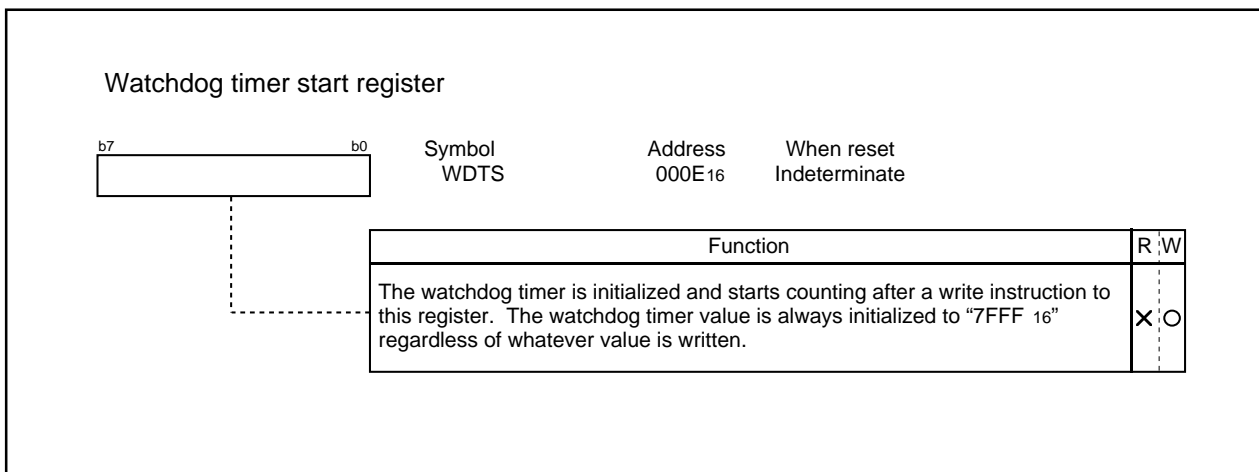


Figure 2.8.3 Watchdog timer start register

## 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.

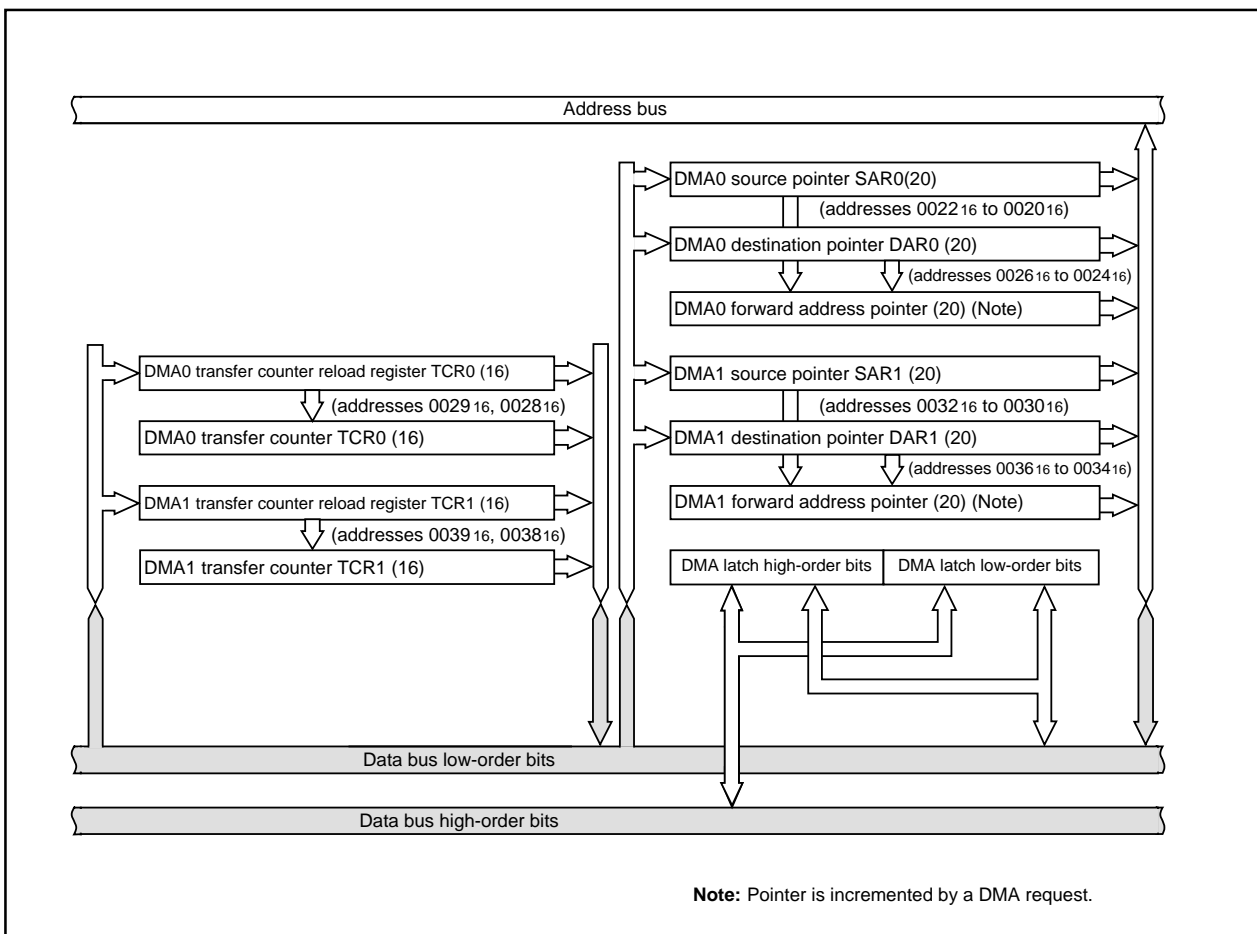


Figure 2.9.1 Block diagram of DMAC

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.

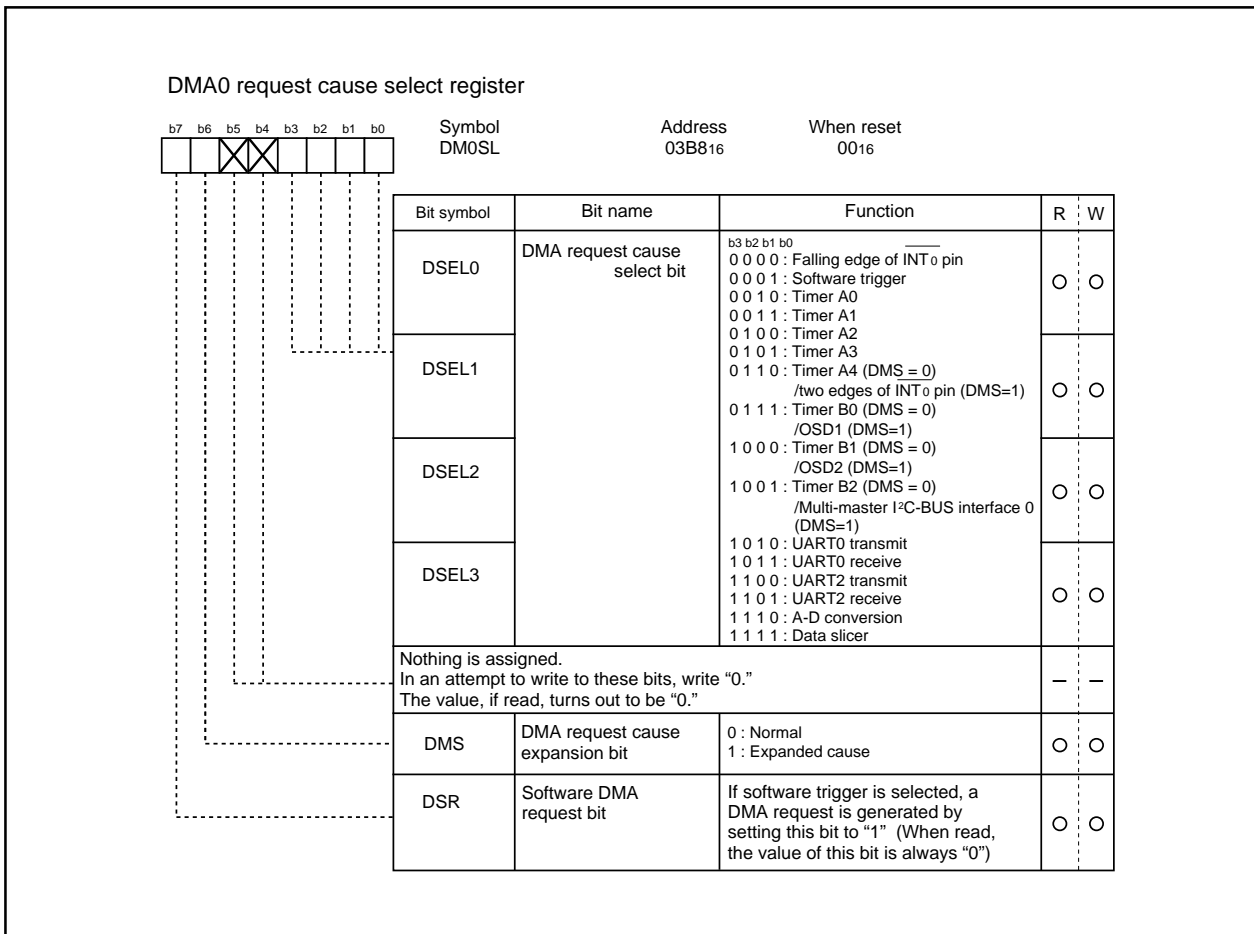
If the DMAC is active (the DMA enable bit is set to 1), data transfer starts every time a DMA transfer request signal occurs. If the cycle of the occurrences of DMA transfer request signals is higher than the DMA transfer cycle, there can be instances in which the number of transfer requests doesn't 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"> <li>From any address in the 1M bytes space to a fixed address</li> <li>From a fixed address to any address in the 1M bytes space</li> <li>From a fixed address to a fixed address</li> </ul> (Note that DMA-related registers [0020 <sub>16</sub> to 003F <sub>16</sub> ] cannot be accessed)
Maximum No. of bytes transferred	128K bytes (with 16-bit transfers) or 64K bytes (with 8-bit transfers)
DMA request factors (Note)	Falling edge or both edge of pin $\overline{INT0}$ Falling edge of pin $\overline{INT1}$ 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 <sup>2</sup> C-BUS interface 0 interrupt request Multi-master I <sup>2</sup> C-BUS interface 1 interrupt request A-D conversion interrupt request OSD1 and OSD2 interrupt requests Data slicer 0 interrupt request (Note2) V <sub>SYNC</sub> 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"> <li>Single transfer mode After the transfer counter underflows, the DMA enable bit turns to "0", and the DMAC turns inactive</li> <li>Repeat transfer mode After the transfer counter underflows, the value of the transfer counter reload register is reloaded to the transfer counter. The DMAC remains active unless a "0" is written to the DMA enable bit.</li> </ul>
DMA interrupt request generation timing	When an underflow occurs in the transfer counter
Active	When the DMA enable bit is set to "1", the DMAC is active. When the DMAC is active, data transfer starts every time a DMA transfer request signal occurs.
Inactive	<ul style="list-style-type: none"> <li>When the DMA enable bit is set to "0", the DMAC is inactive.</li> <li>After the transfer counter underflows in single transfer mode</li> </ul>
Forward address pointer and 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.

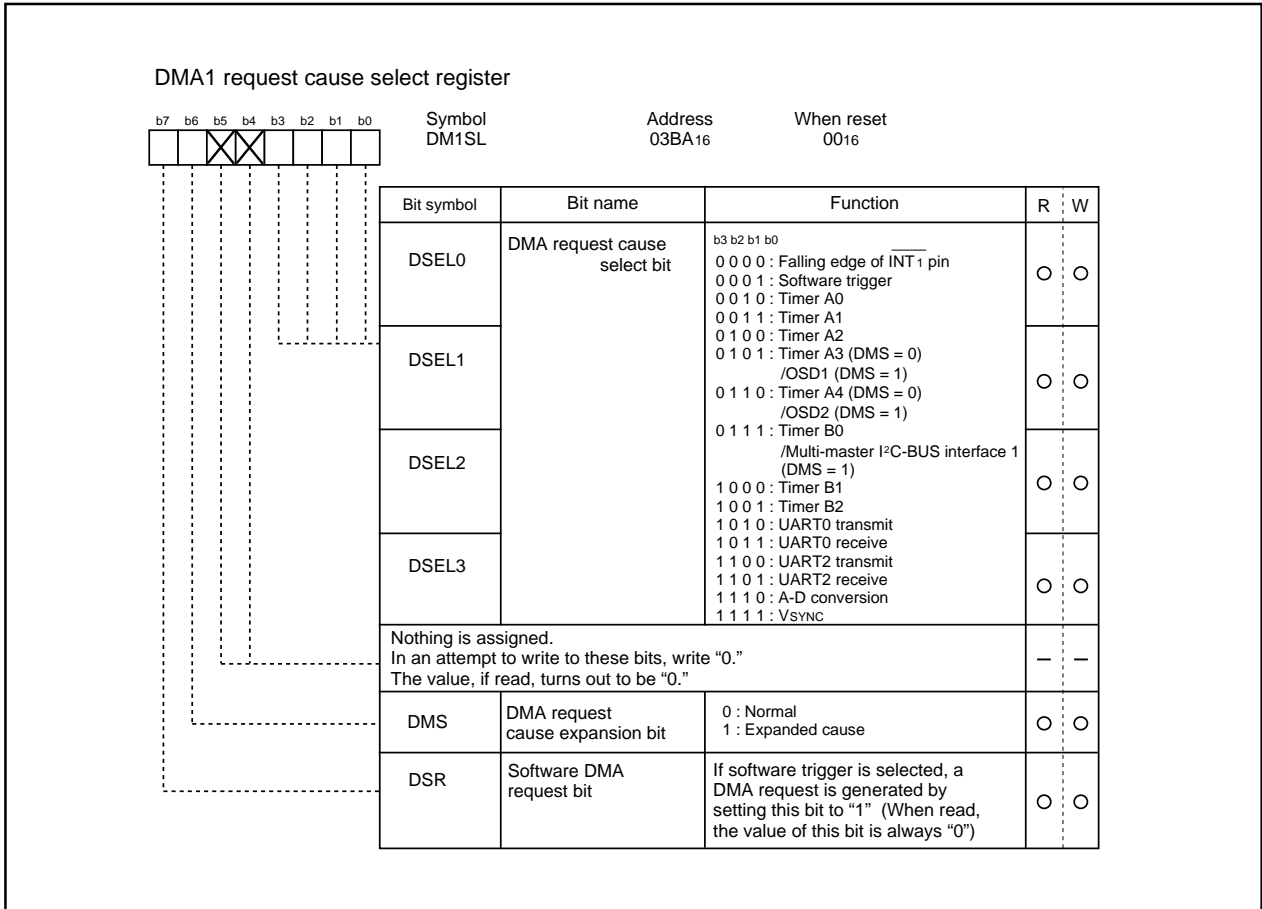
**Note1:** DMA transfer is not effective to any interrupt. DMA transfer is affected neither by the interrupt enable flag (I flag) nor by the interrupt priority level.

**2:** No DMA request sources for data slicer 1 are available.

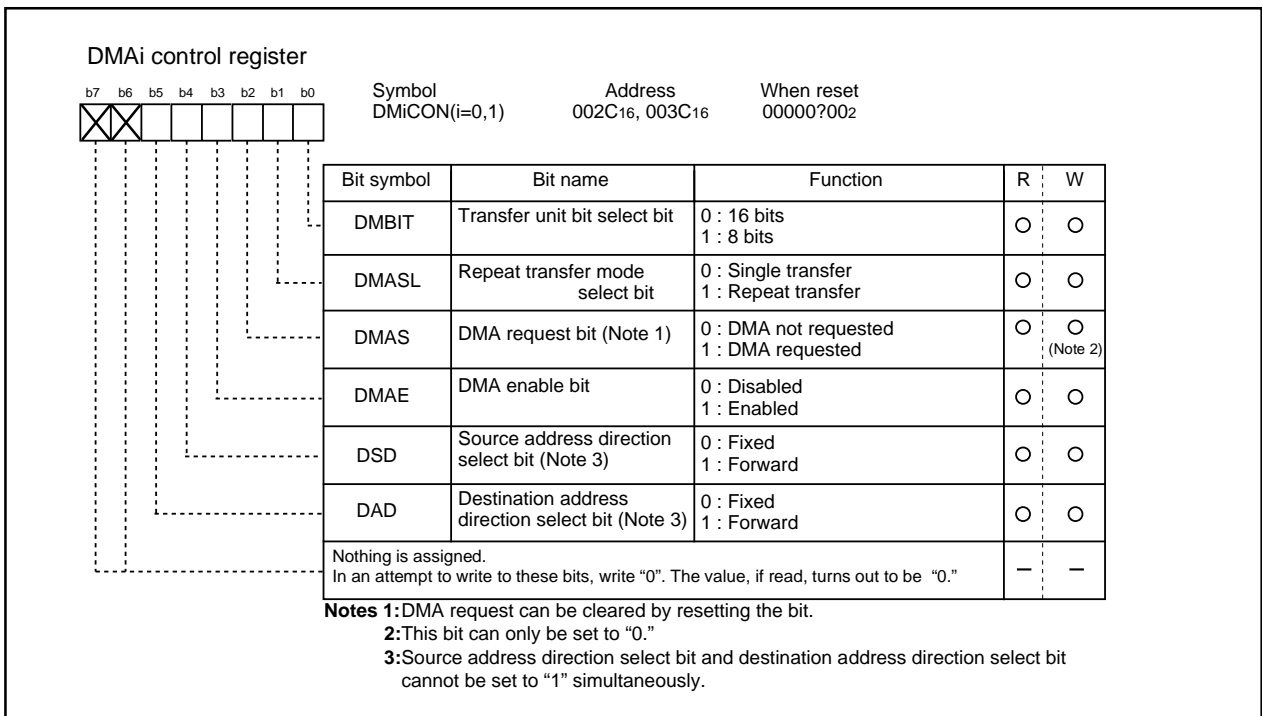


**Figure 2.9.2 DMA0 request cause select register**

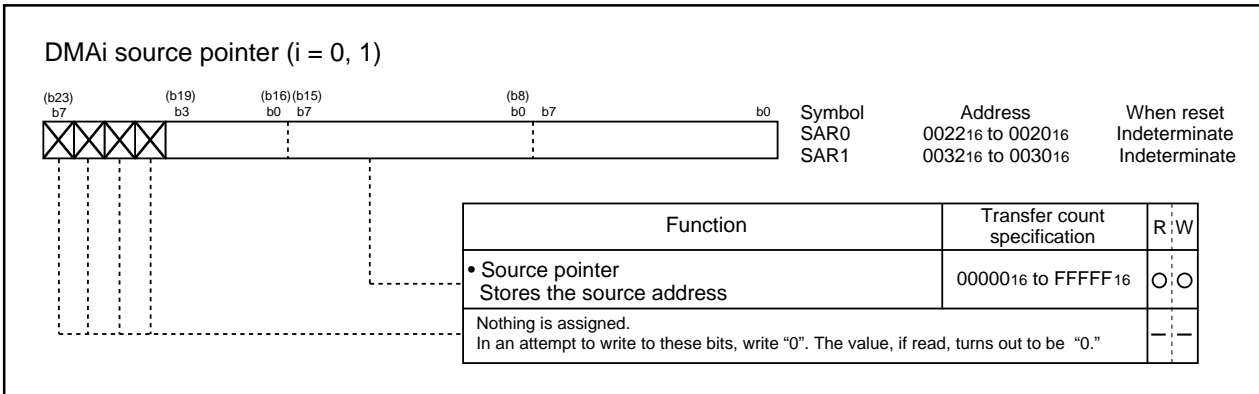




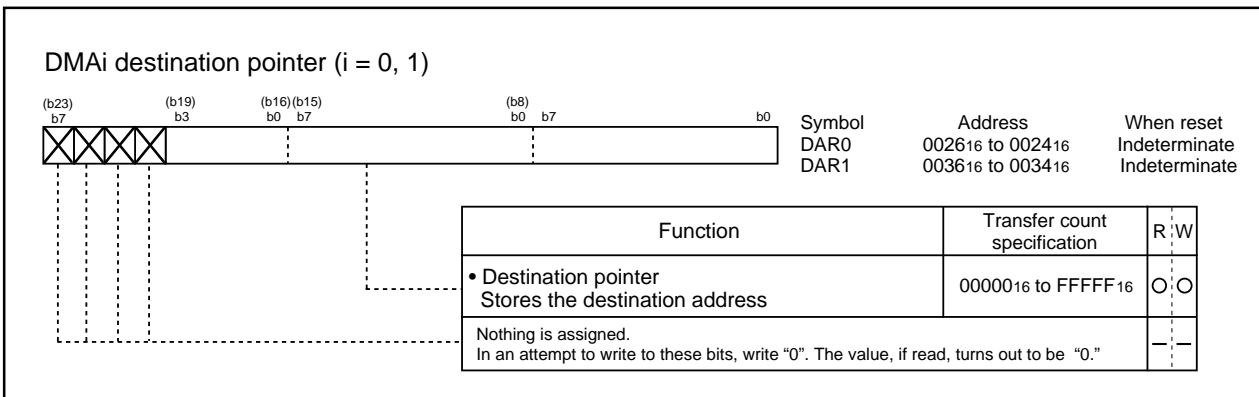
**Figure 2.9.3 DMA1 request cause select register**



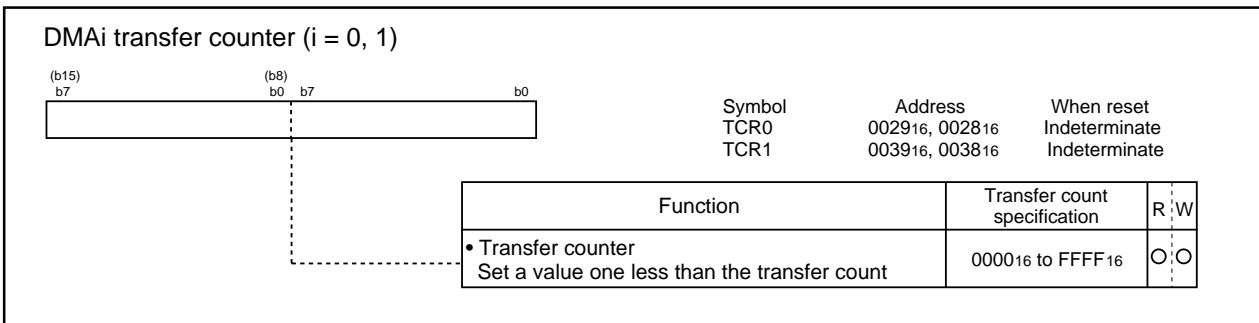
**Figure 2.9.4 DMAi control register (i = 0, 1)**



**Figure 2.9.5 DMAi source pointer (i = 0, 1)**



**Figure 2.9.6 DMAi destination pointer (i = 0, 1)**



**Figure 2.9.7 DMAi 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.

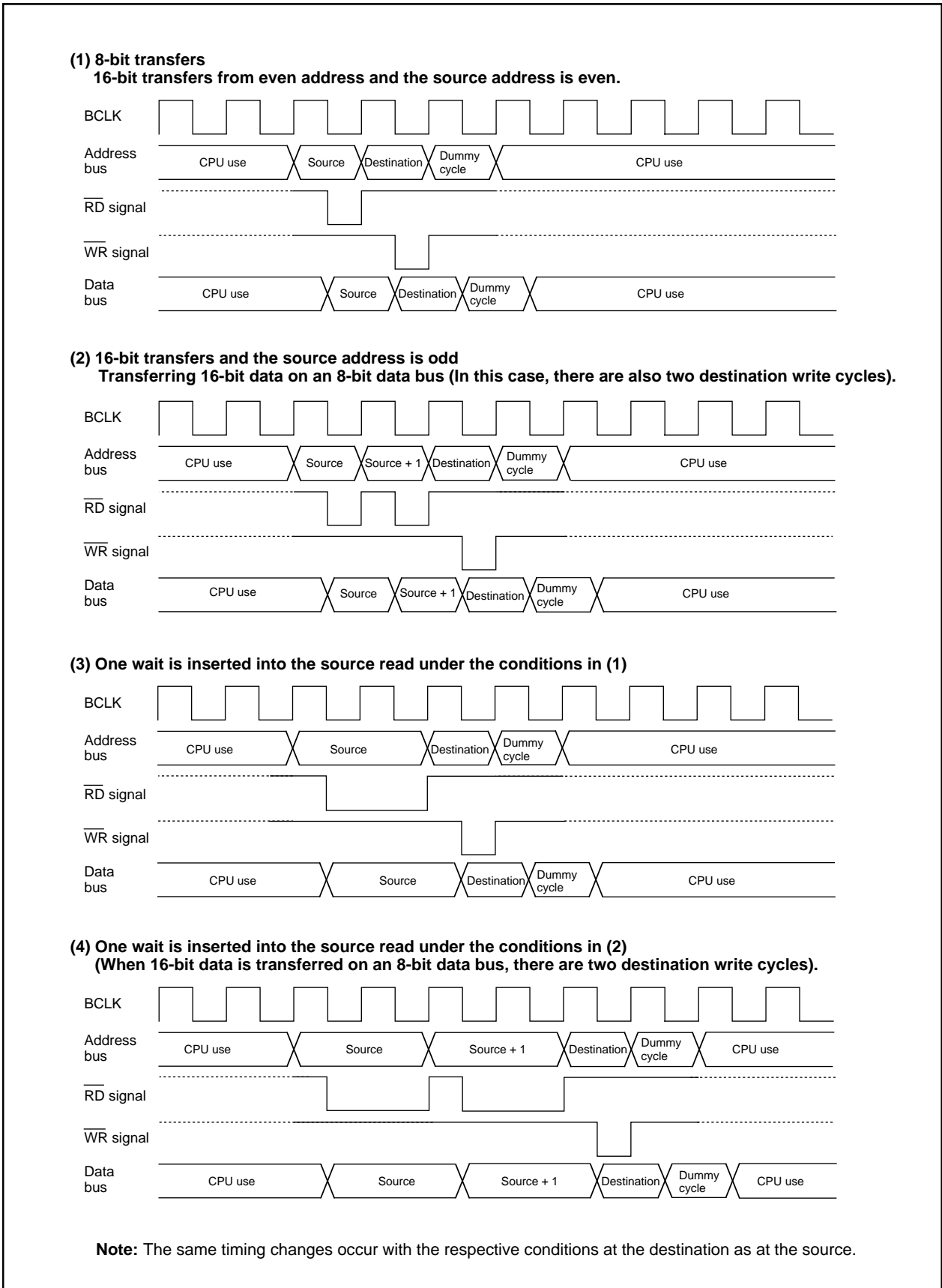
### (2) Effect of BYTE pin level

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	SFR area	Separate bus /OSD RAM No wait	Separate bus With wait
1	2	1	2

### 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<sub>i</sub> factor selection register.

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

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

There can be instances in which a change in DMA request factor selection bit causes the DMA request bit to turn to 1. So be sure to set the DMA request bit to 0 after the DMA request factor selection bit is changed.

The DMA request bit turns to 1 if a DMA transfer request signal occurs, and turns to 0 immediately before data transfer starts. If the DMAC is active, data transfer starts immediately, so the value of the DMA request bit, if read by use of a program, turns out to be 0 in most cases. To examine whether the DMAC is active, read the DMA enable bit.

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

#### (1) Internal factors

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

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

#### (2) External factors

An external factor is a factor caused to occur by the leading edge of input from the  $\overline{\text{INT}}_i$  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.

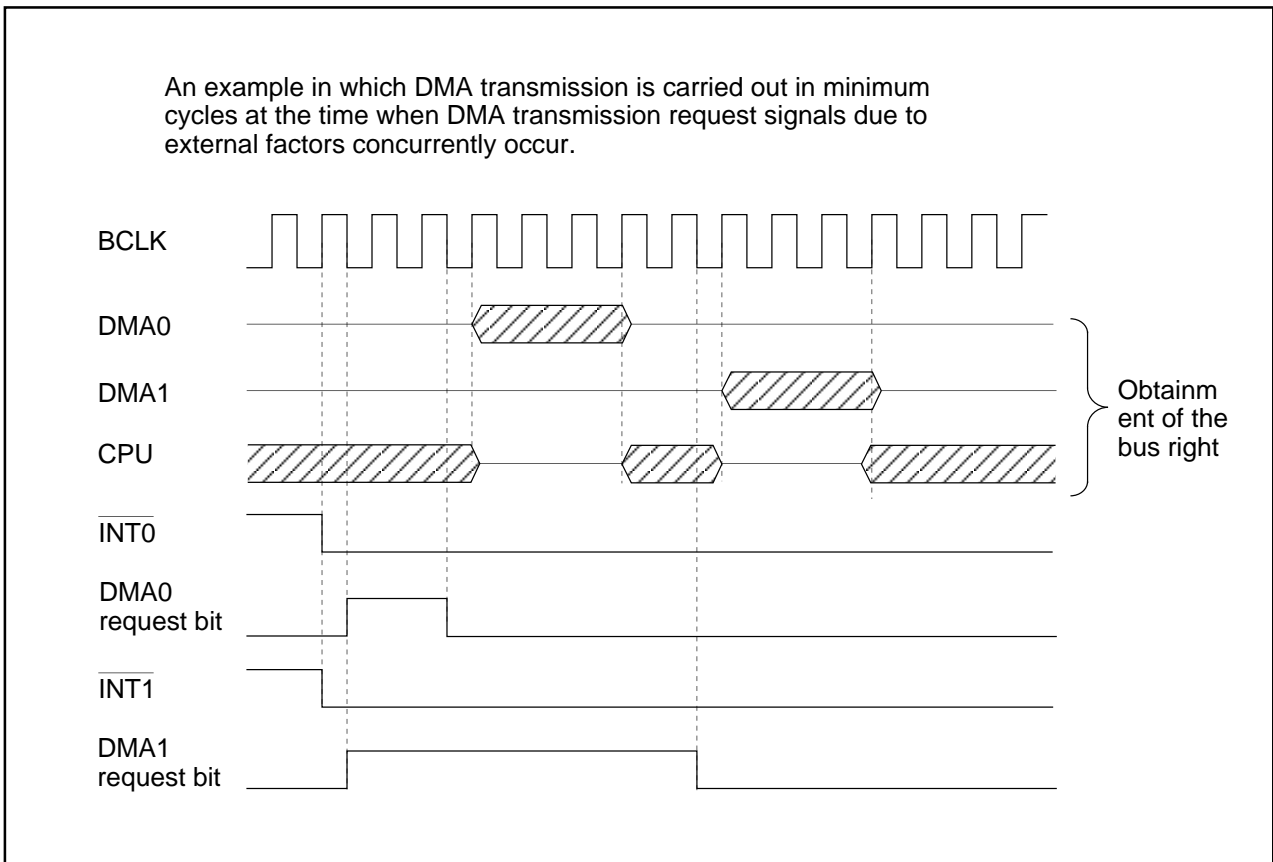
The timing for the DMA request bit to turn to 1 when an external factor is selected synchronizes with the signal's edge applicable to the function specified by the DMA request factor selection bit (synchronizes with the trailing edge of the input signal to each  $\overline{INTi}$  pin, for example).

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

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

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

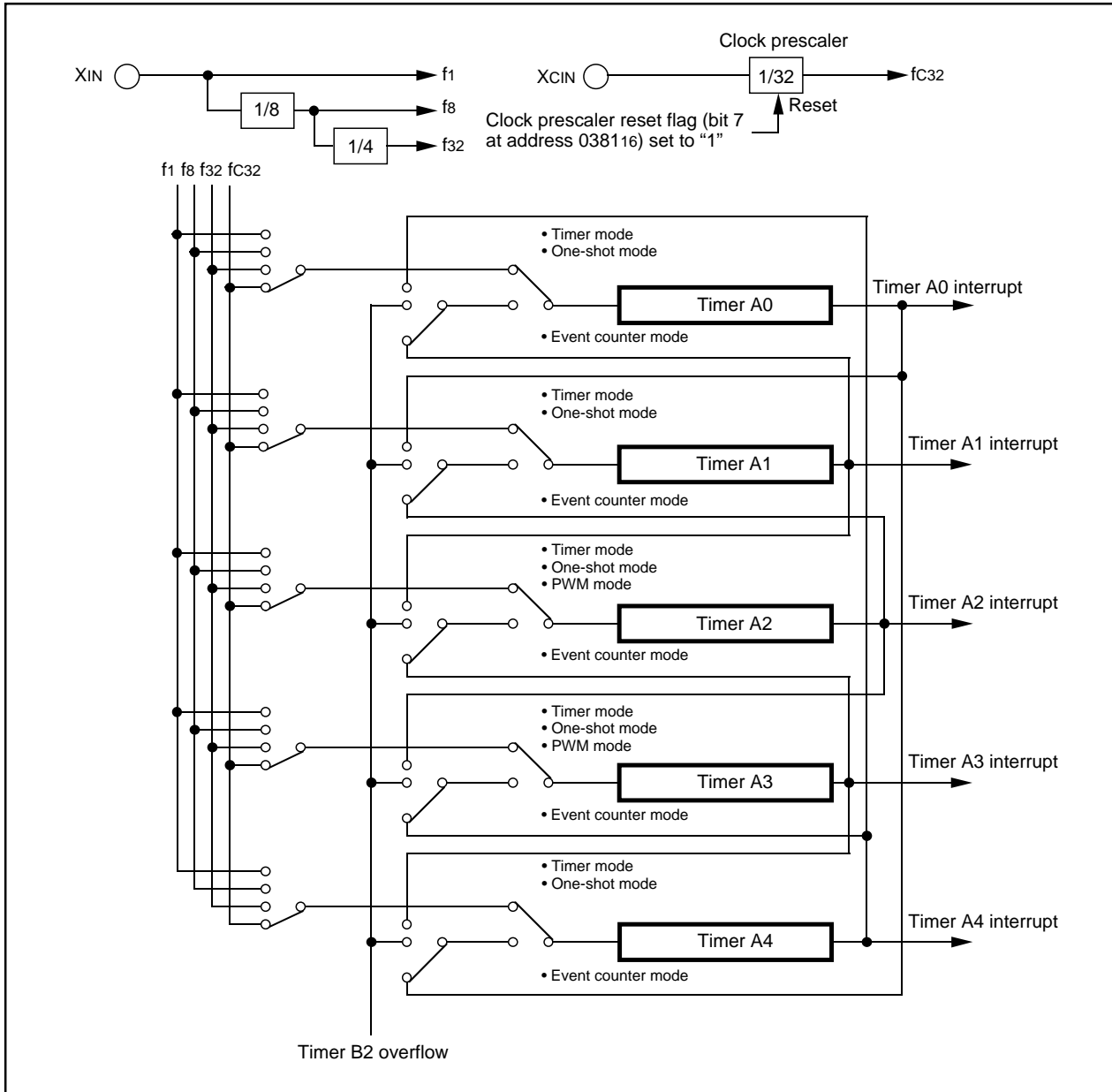


Figure 2.10.1 Timer A block diagram



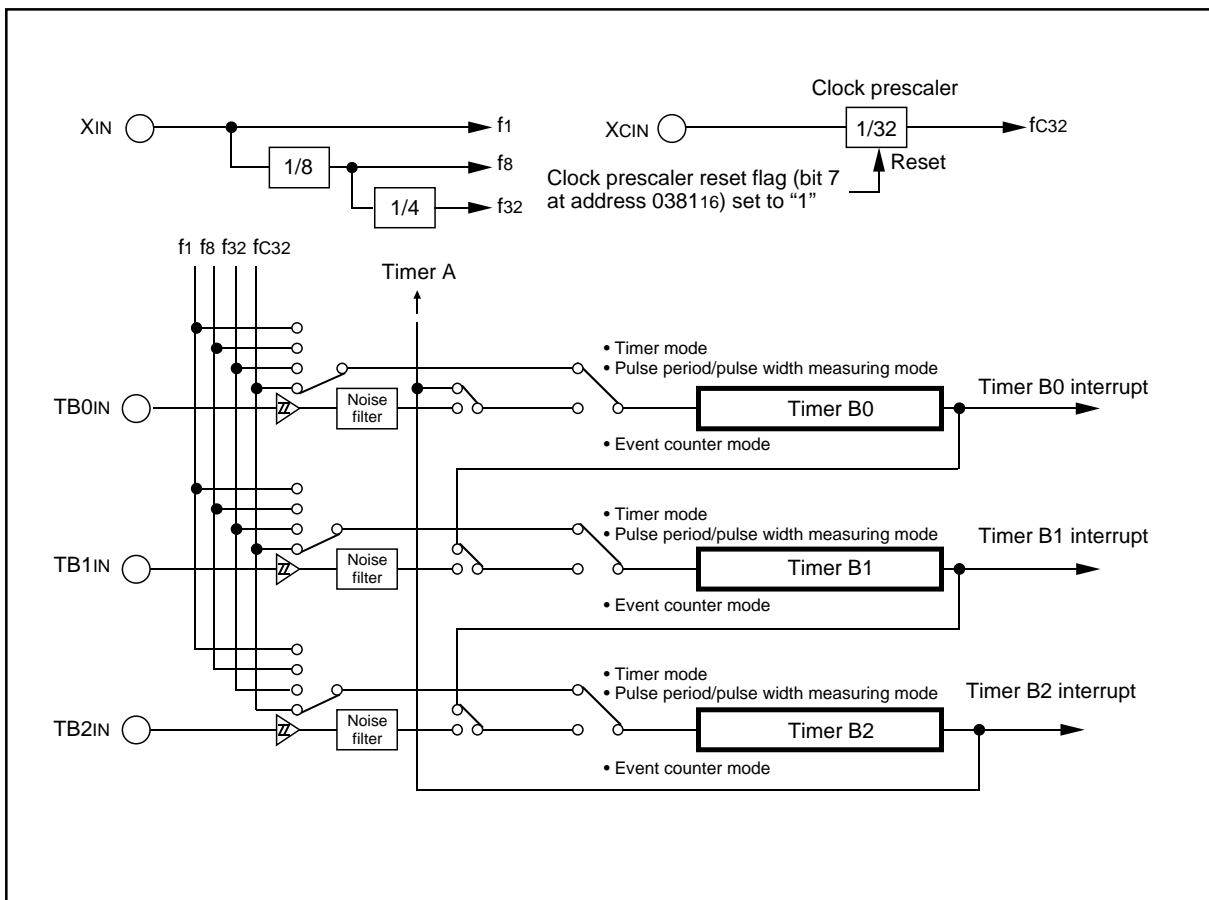


Figure 2.10.2 Timer B block diagram

### 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.

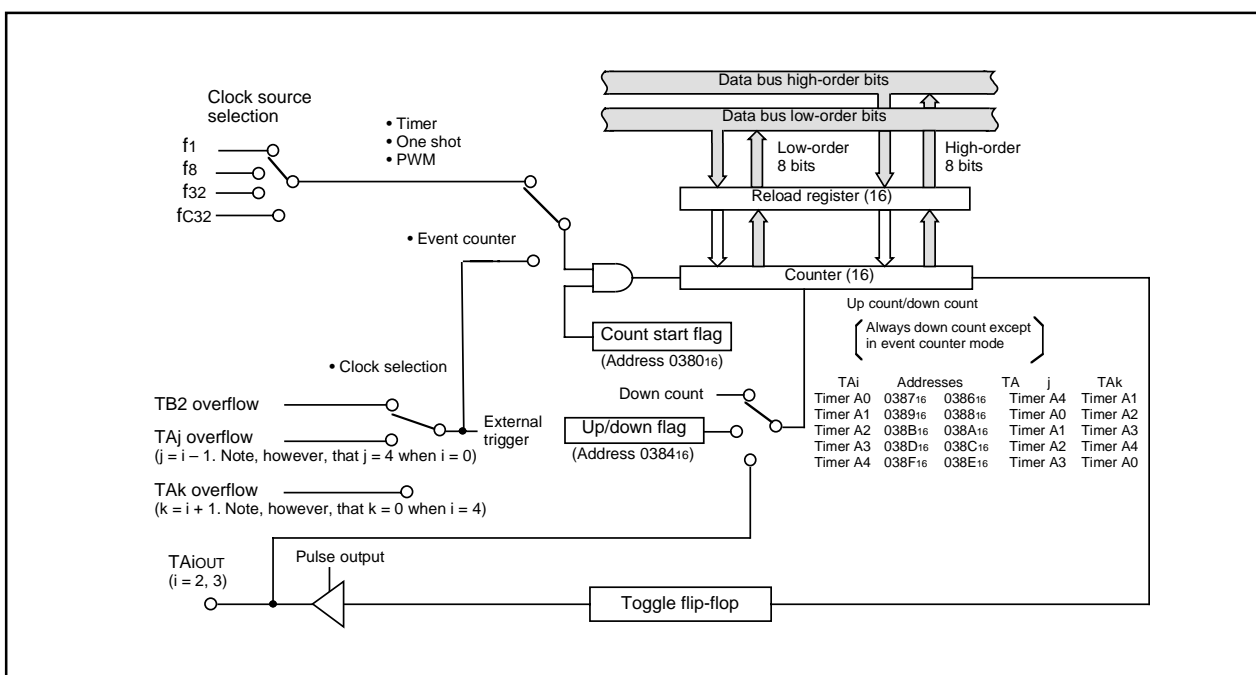


Figure 2.10.3 Block diagram of timer A

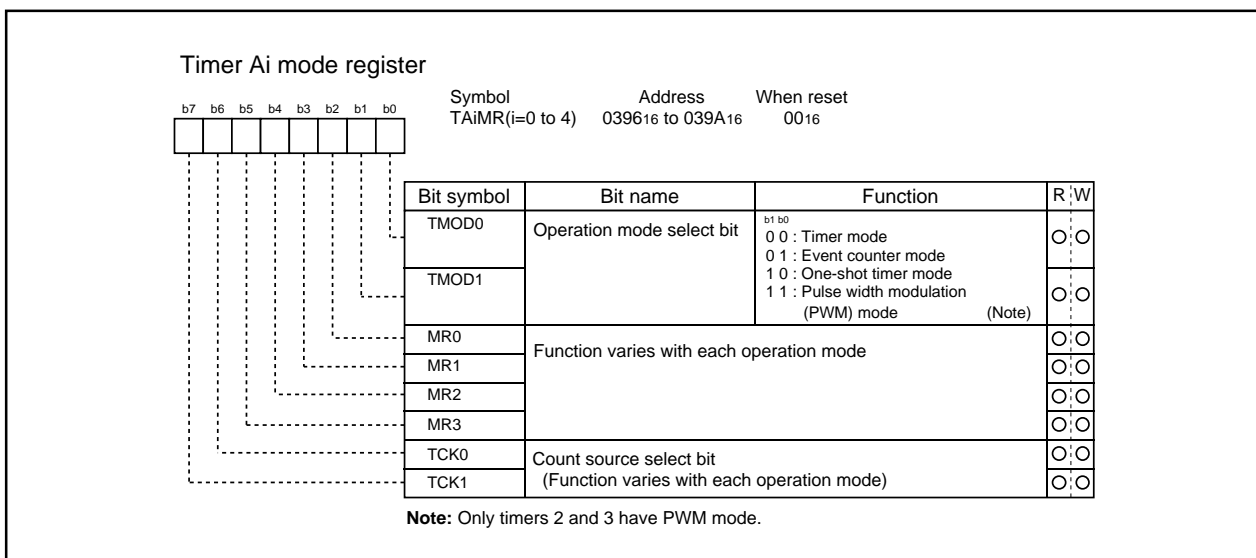


Figure 2.10.4 Timer Ai mode register (i = 0 to 4)

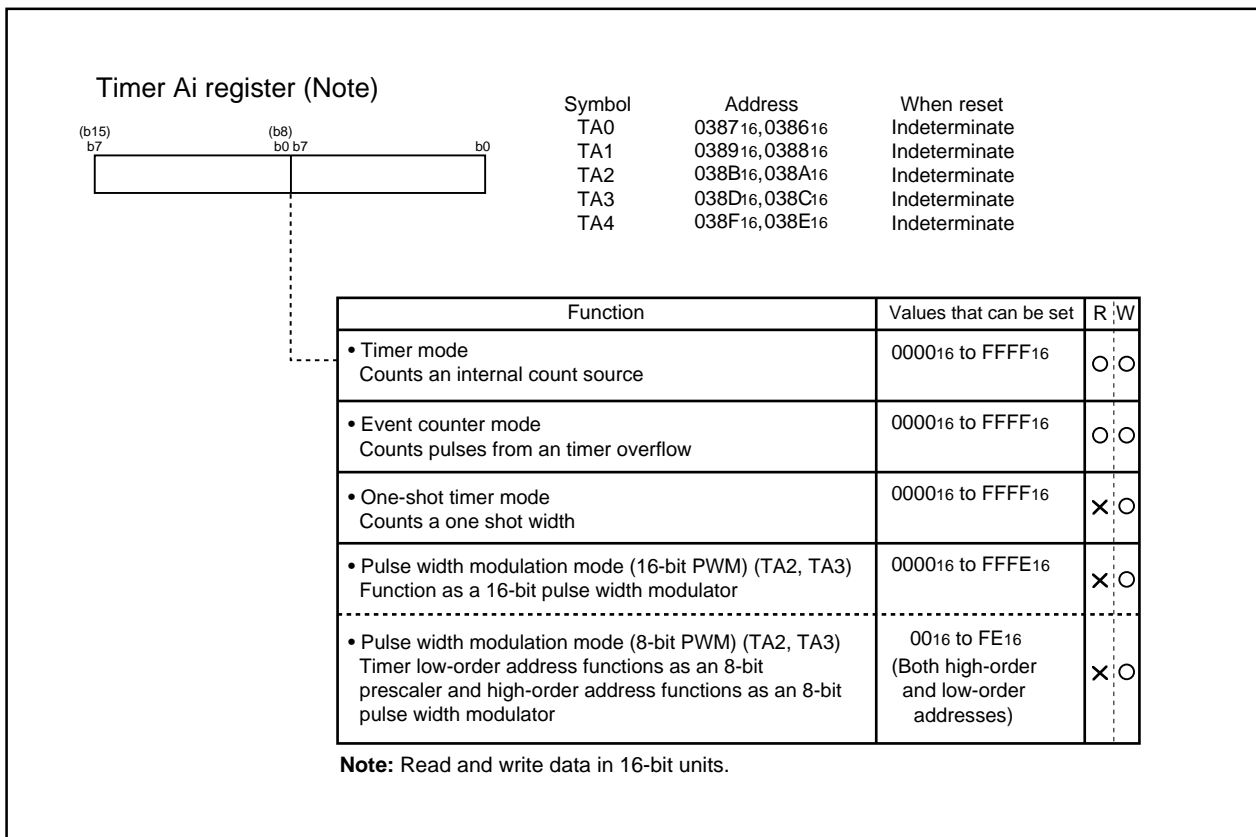


Figure 2.10.5 Timer Ai register (i = 0 to 4)

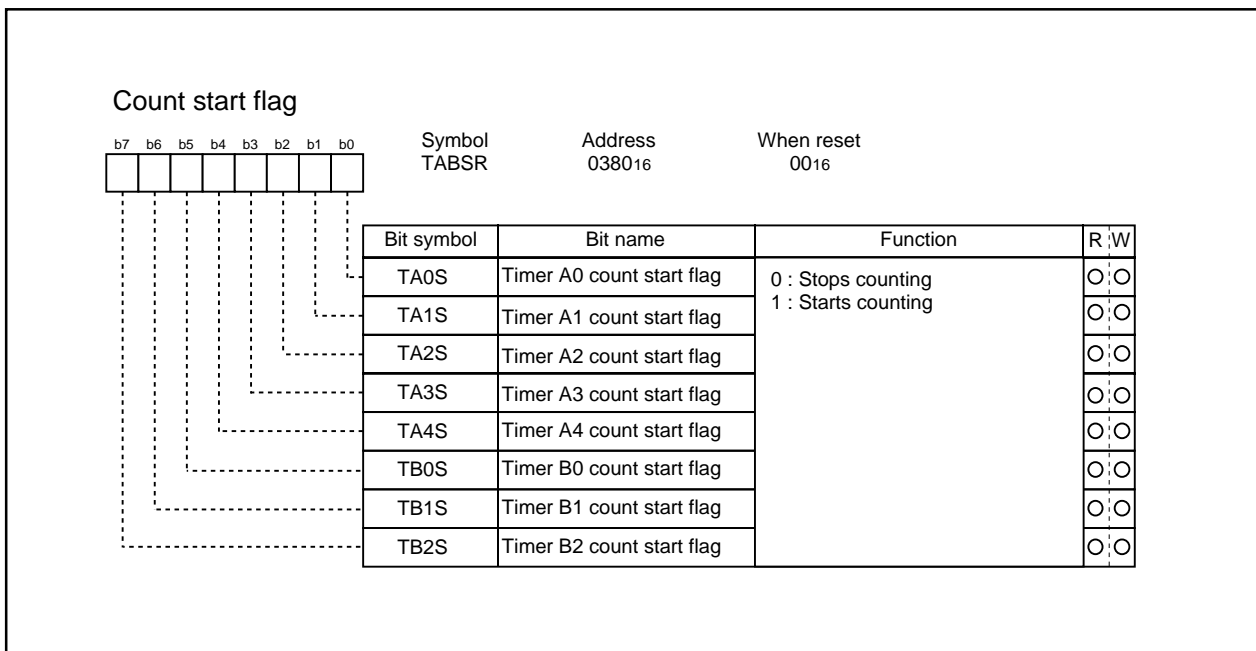


Figure 2.10.6 Count start flag

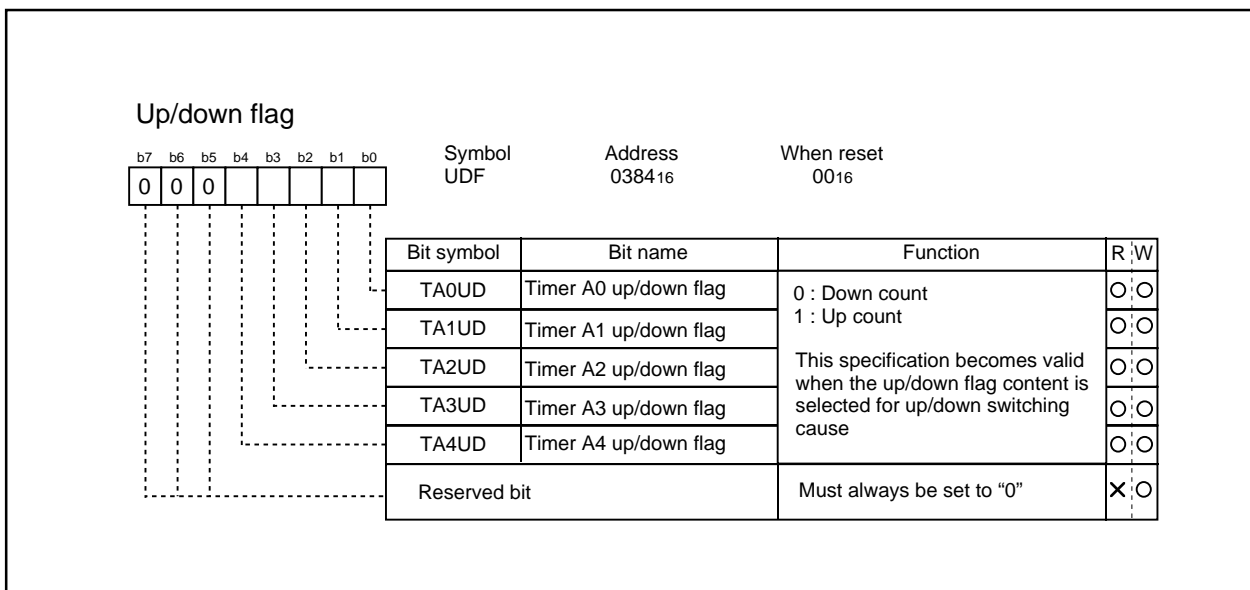


Figure 2.10.7 Up/down flag

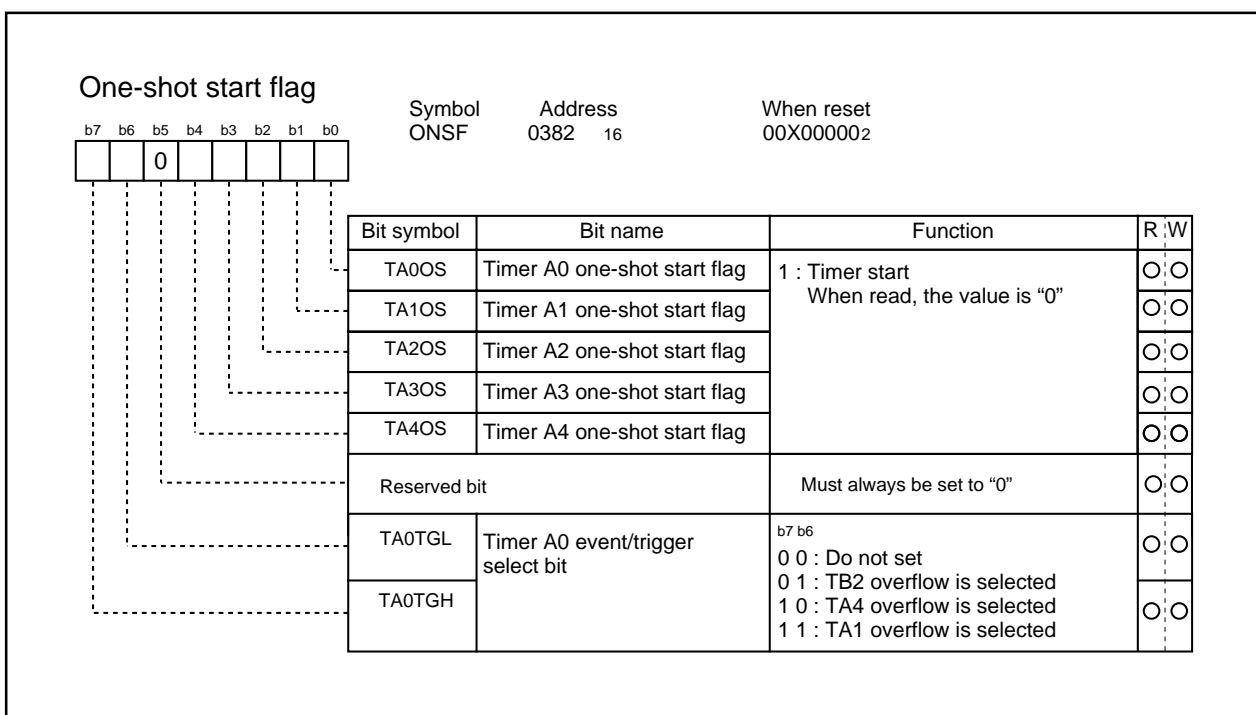


Figure 2.10.8 One-shot start flag

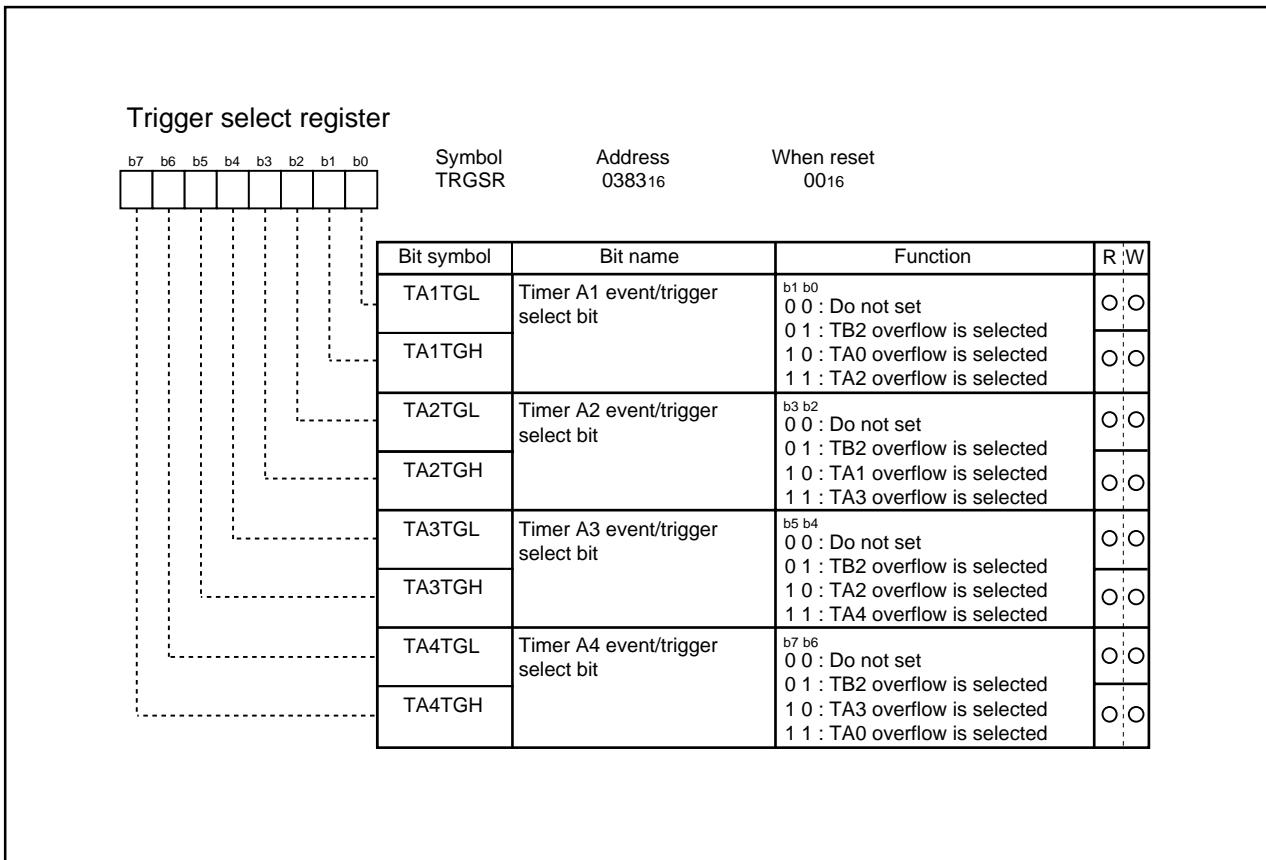


Figure 2.10.9 Trigger select register

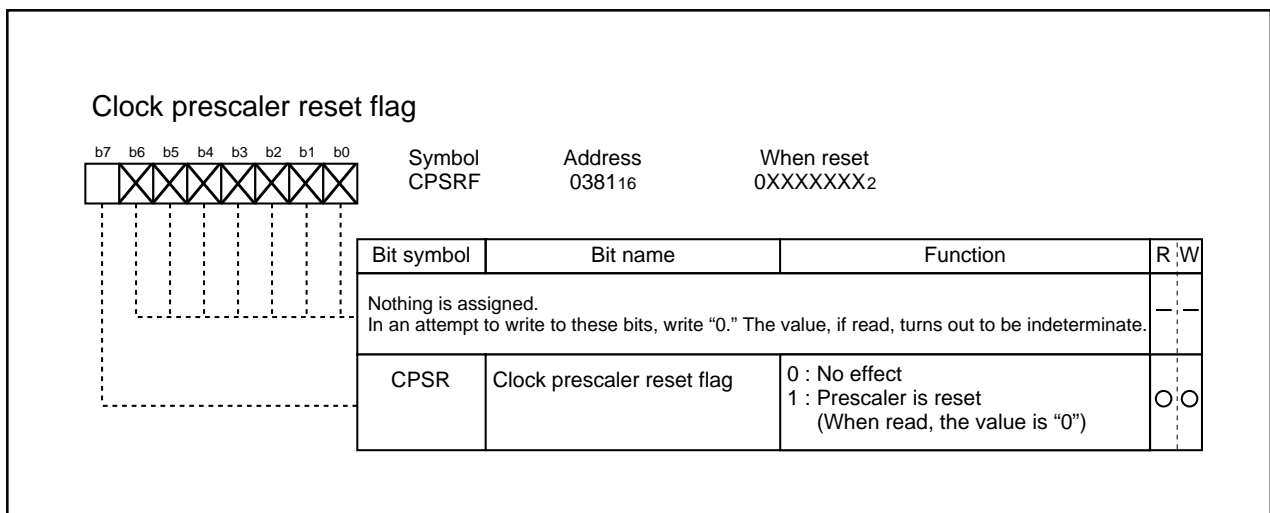


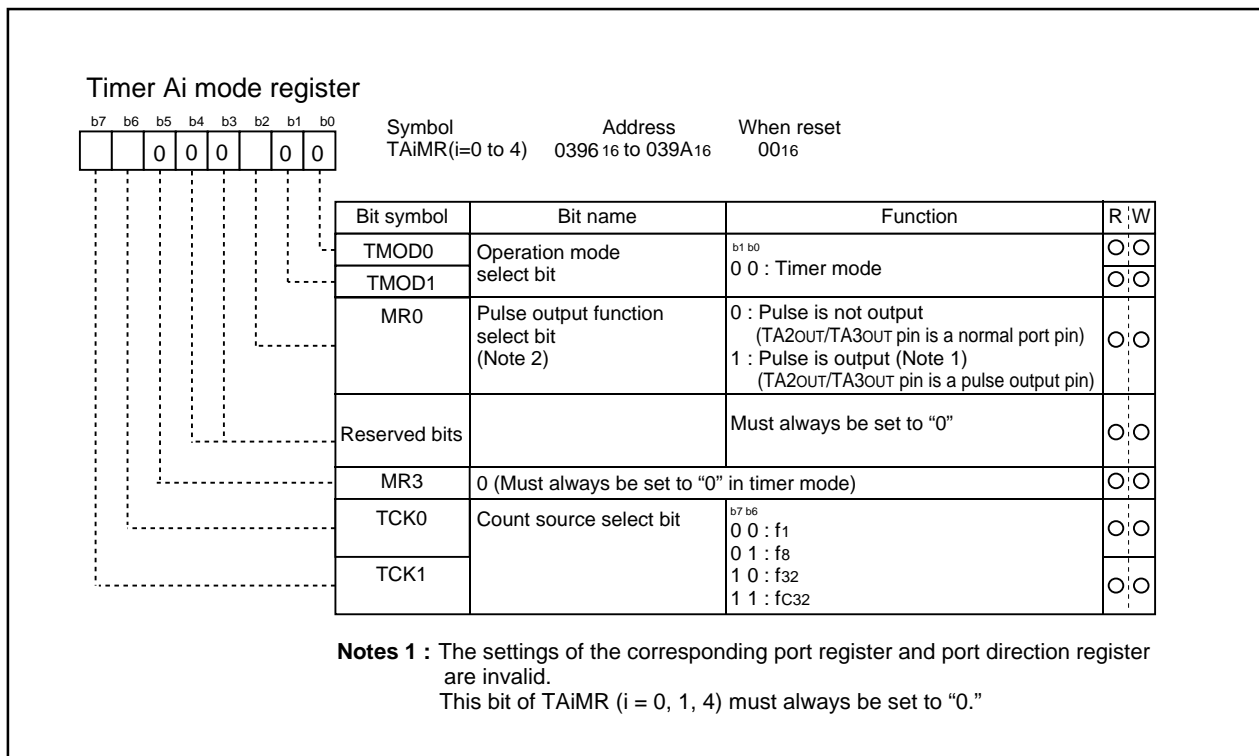
Figure 2.10.10 Clock prescaler reset flag

### (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 <sub>1</sub> , f <sub>8</sub> , f <sub>32</sub> , f <sub>c32</sub>
Count operation	<ul style="list-style-type: none"> <li>Down count</li> <li>When the timer underflows, it reloads the reload register contents before continuing counting</li> </ul>
Divide ratio	1/(n+1)    n : Set value
Count start condition	Count start flag is set (= 1)
Count stop condition	Count start flag is reset (= 0)
Interrupt request generation timing	When the timer underflows
TA2 <sub>OUT</sub> /TA3 <sub>OUT</sub> pin function	Programmable I/O port or pulse output
Read from timer	Count value can be read out by reading timer Ai register
Write to timer	<ul style="list-style-type: none"> <li>When counting stopped When a value is written to timer Ai register, it is written to both reload register and counter</li> <li>When counting in progress When a value is written to timer Ai register, it is written to only reload register (Transferred to counter at next reload time)</li> </ul>
Select function	<ul style="list-style-type: none"> <li>Pulse output function Each time the timer underflows, the TAI<sub>OUT</sub> pin's polarity is reversed</li> </ul>



**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"> <li>• TB2 overflow, TAj overflow, TAk overflow</li> </ul>
Count operation	<ul style="list-style-type: none"> <li>• Up count or down count can be selected by external signal or software</li> <li>• When the timer overflows or underflows, it reloads the reload register contents before continuing counting (Note)</li> </ul>
Divide ratio	$1 / (FFFF_{16} - n + 1)$ for up count $1 / (n + 1)$ for down count                      n : Set value
Count start condition	Count start flag is set (= 1)
Count stop condition	Count start flag is reset (= 0)
Interrupt request generation timing	The timer overflows or underflows
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"> <li>• When counting stopped When a value is written to timer Ai register, it is written to both reload register and counter</li> <li>• When counting in progress When a value is written to timer Ai register, it is written to only reload register (Transferred to counter at next reload time)</li> </ul>
Select function	<ul style="list-style-type: none"> <li>• Free-run count function Even when the timer overflows or underflows, the reload register content is not reloaded to it</li> <li>• Pulse output function Each time the timer overflows or underflows, the TAIOUT pin's polarity is reversed</li> </ul>

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

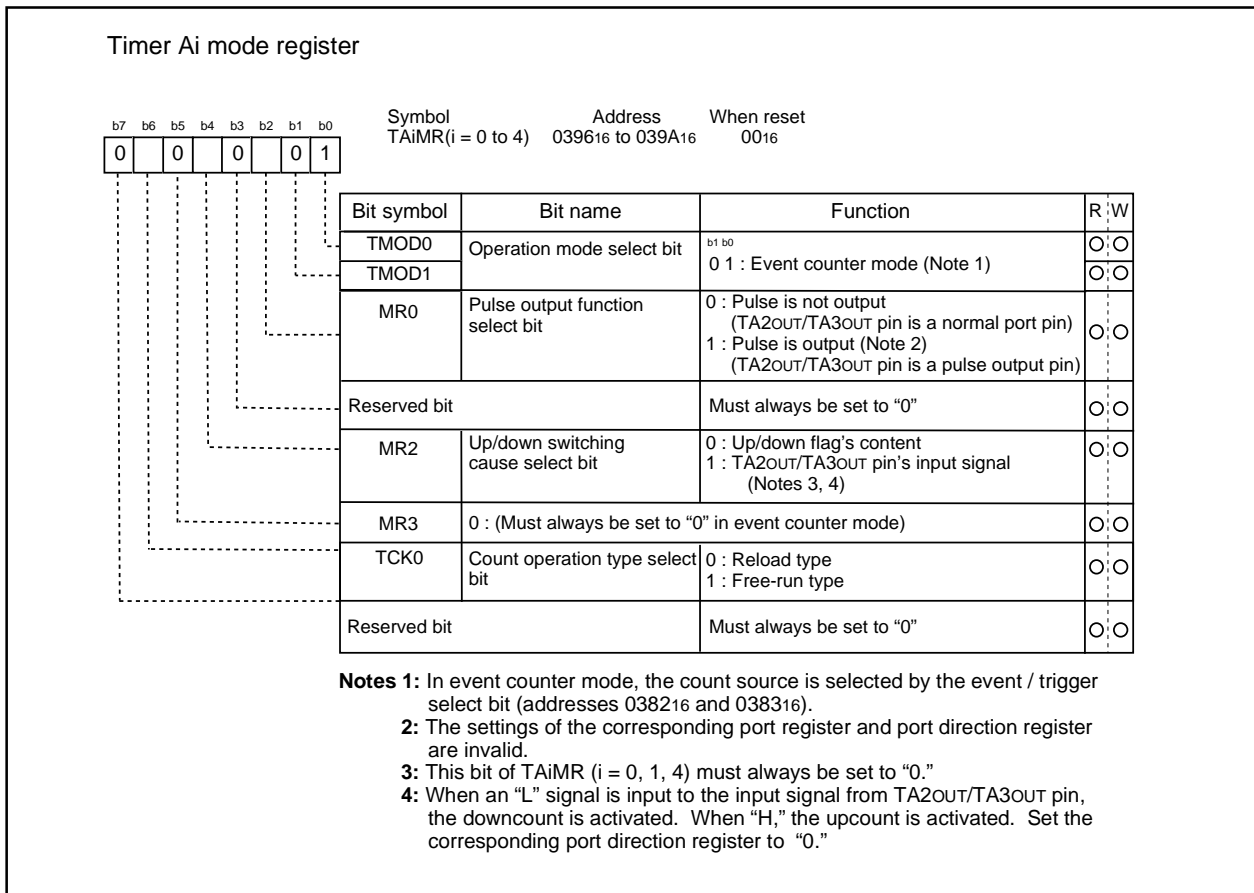


Figure 2.10.12 Timer Ai mode register in event counter mode (i = 0 to 4)

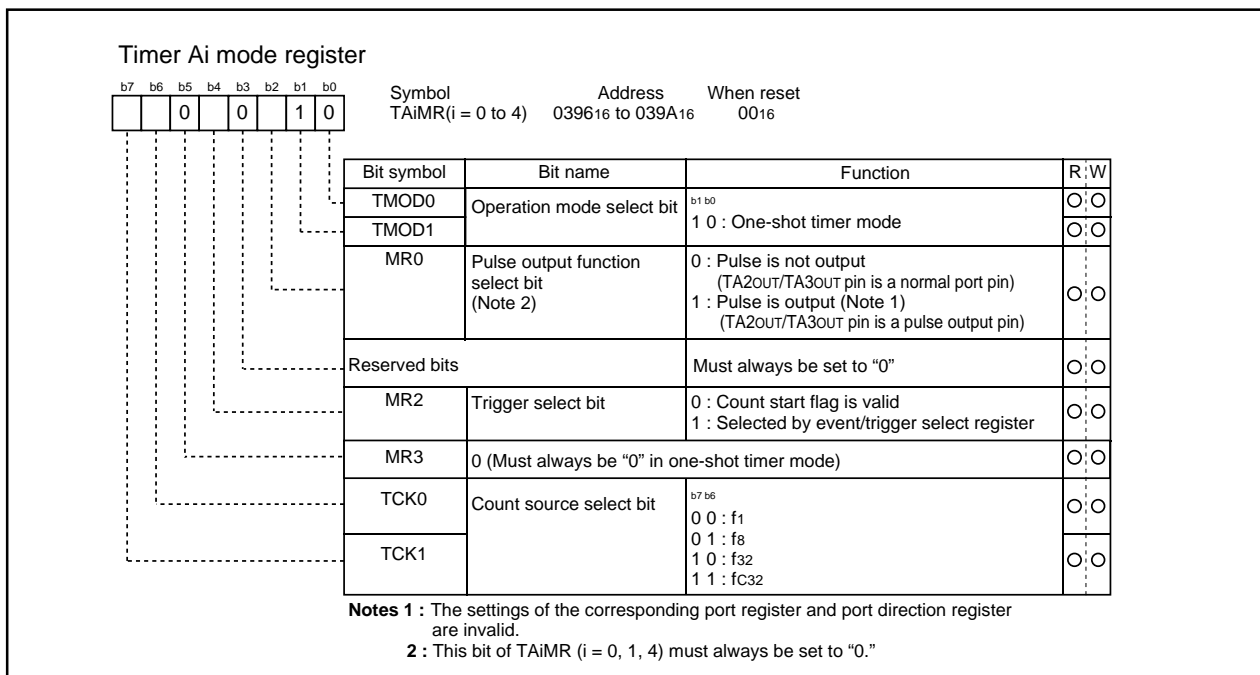


### (3) One-shot timer mode

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"> <li>The timer counts down</li> <li>When the count reaches 0000<sub>16</sub>, the timer stops counting after reloading a new count</li> <li>If a trigger occurs when counting, the timer reloads a new count and restarts counting</li> </ul>
Divide ratio	1/n    n : Set value
Count start condition	<ul style="list-style-type: none"> <li>The timer overflows</li> <li>The one-shot start flag is set (= 1)</li> </ul>
Count stop condition	<ul style="list-style-type: none"> <li>A new count is reloaded after the count has reached 0000<sub>16</sub></li> <li>The count start flag is reset (= 0)</li> </ul>
Interrupt request generation timing	The count reaches 0000 <sub>16</sub>
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"> <li>When counting stopped When a value is written to timer Ai register, it is written to both reload register and counter</li> <li>When counting in progress When a value is written to timer Ai register, it is written to only reload register (Transferred to counter at next reload time)</li> </ul>



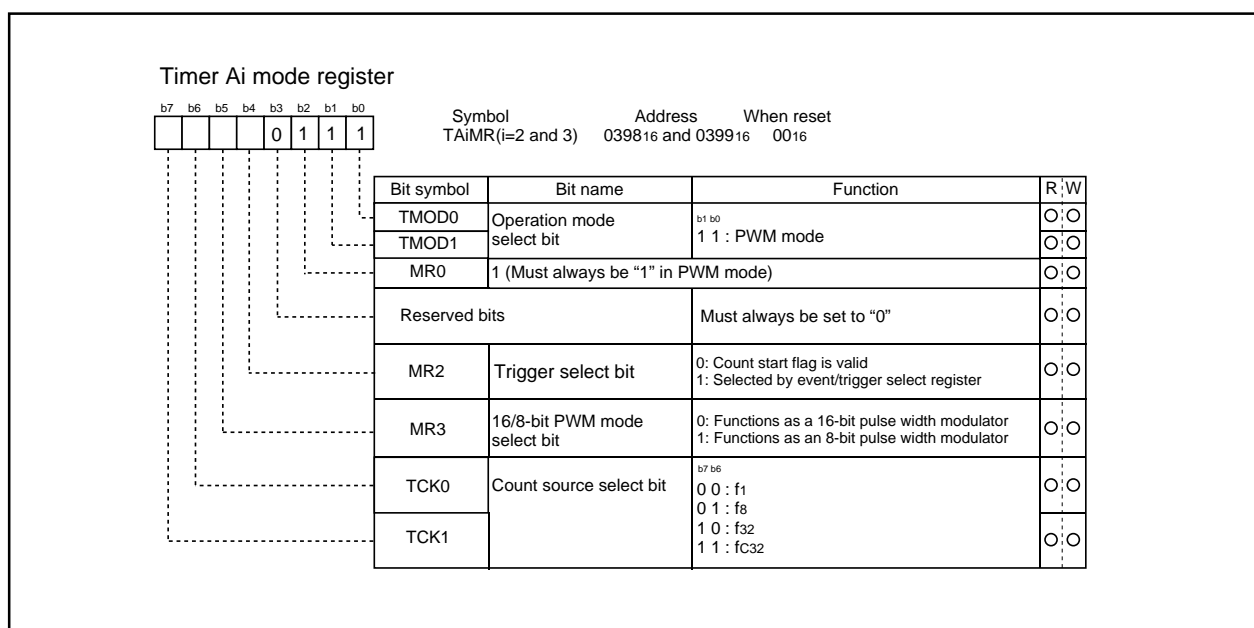
**Figure 2.10.13** Timer Ai mode register in one-shot timer mode (i = 0 to 4)

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



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

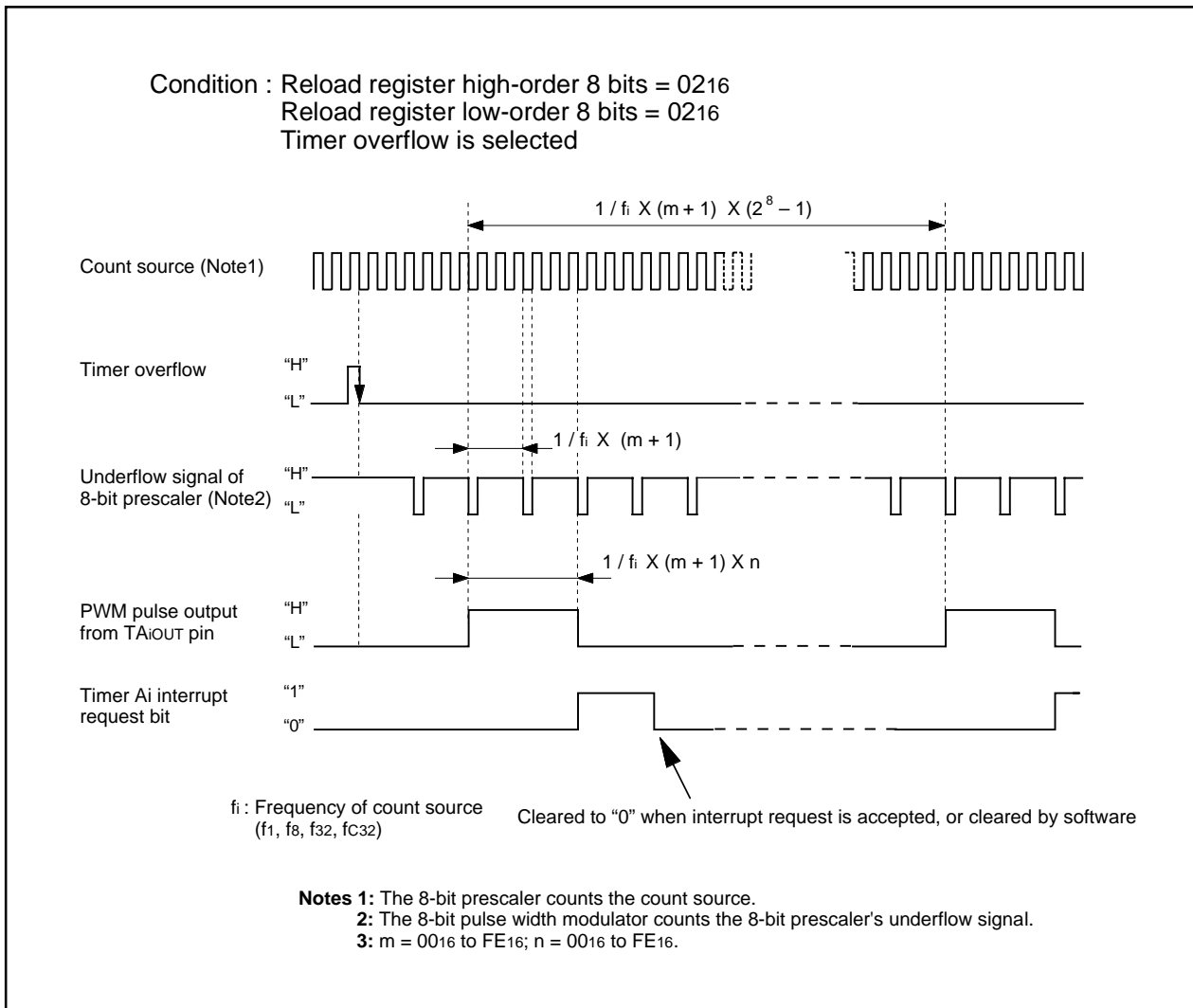


Figure 2.10.15 Example of how an 8-bit pulse width modulator operates

### 2.10.2 Timer B

Figure 2.10.16 shows the block diagram of timer B. Figures 2.10.17 and 2.10.18 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.

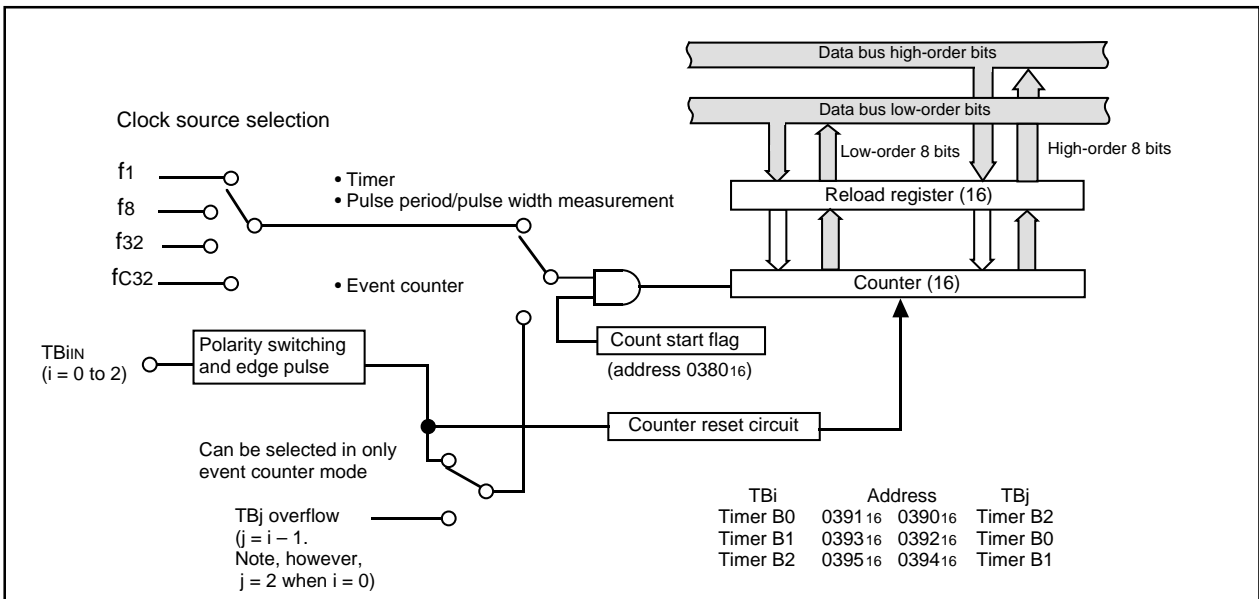


Figure 2.10.16 Block diagram of timer B

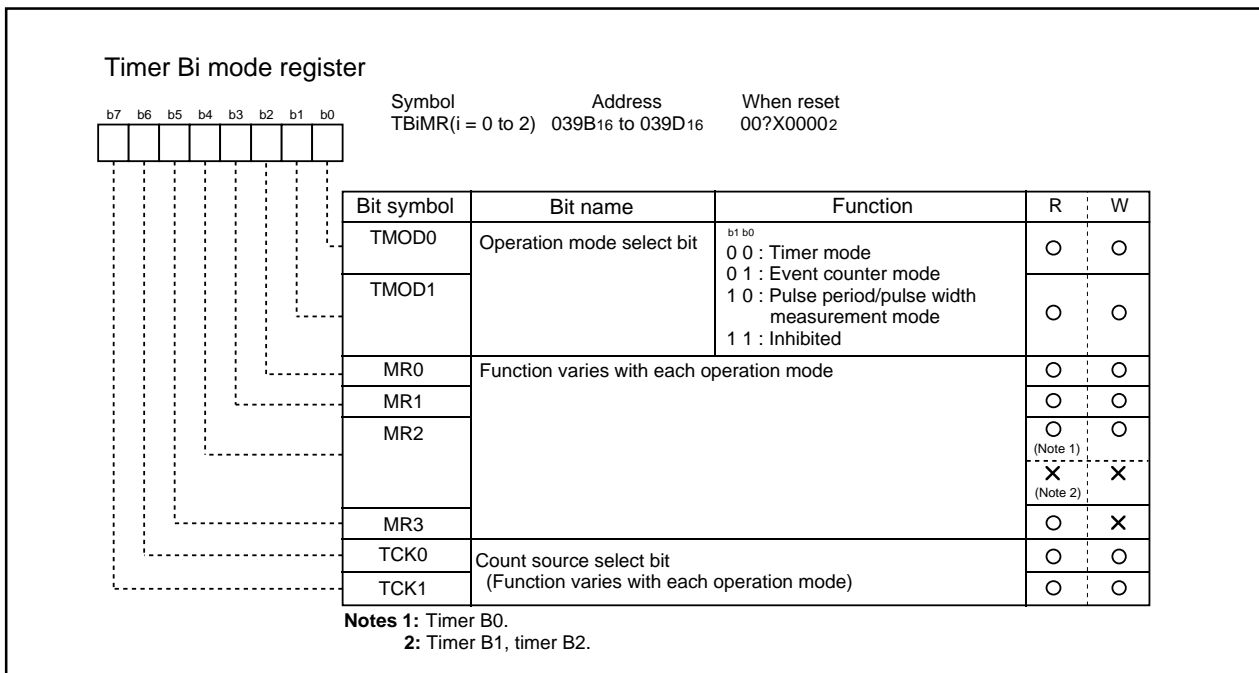


Figure 2.10.17 Timer Bi mode register (i = 0 to 2)

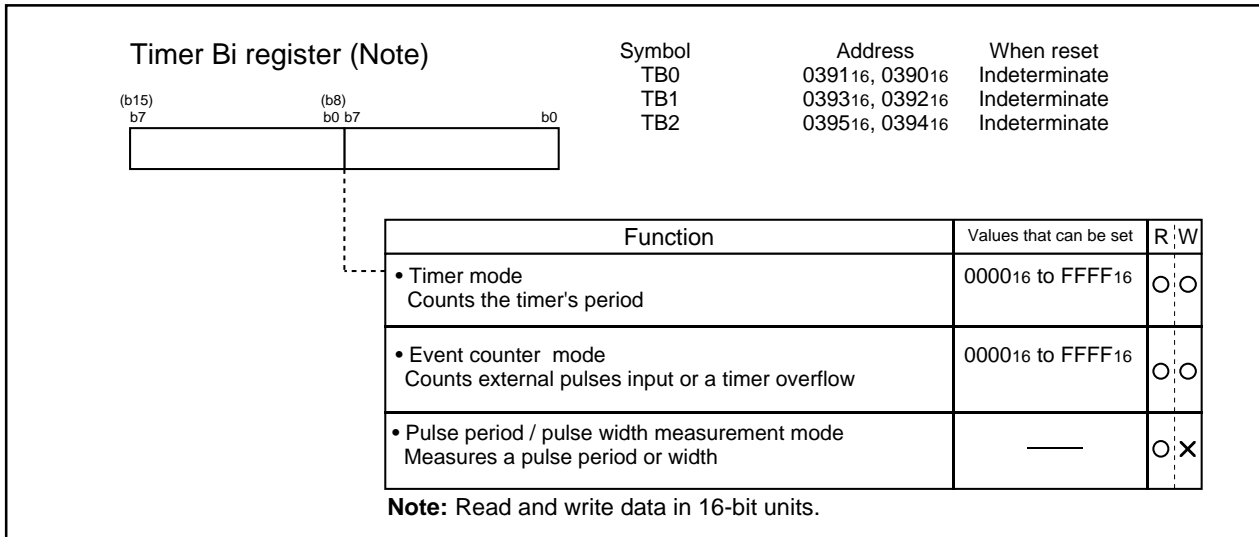


Figure 2.10.18 Timer Bi register (i = 0 to 2)

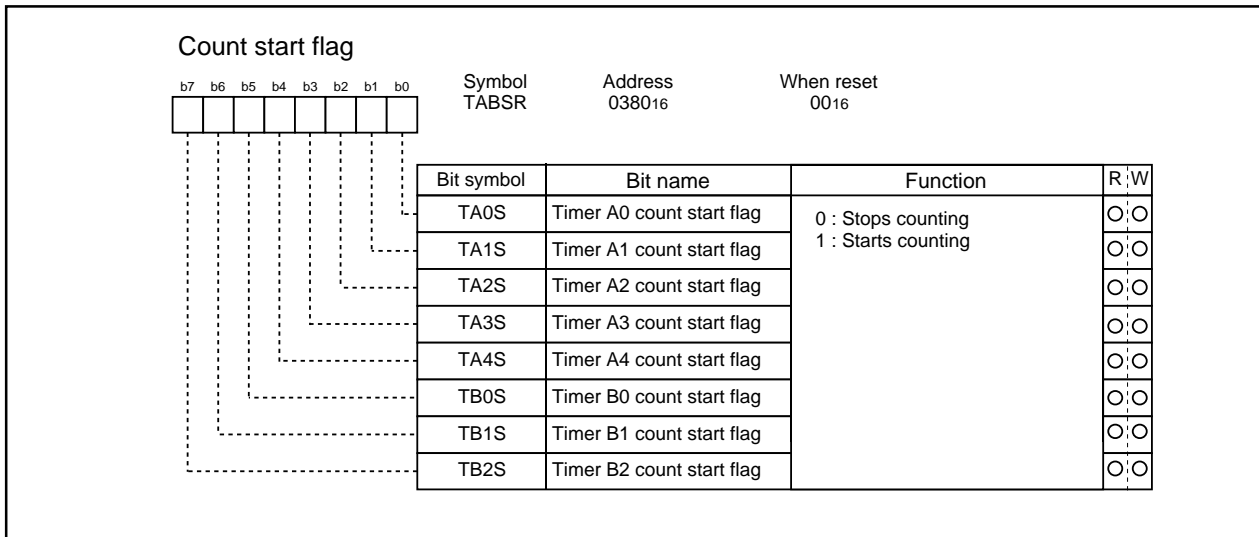


Figure 2.10.19 Count start flag

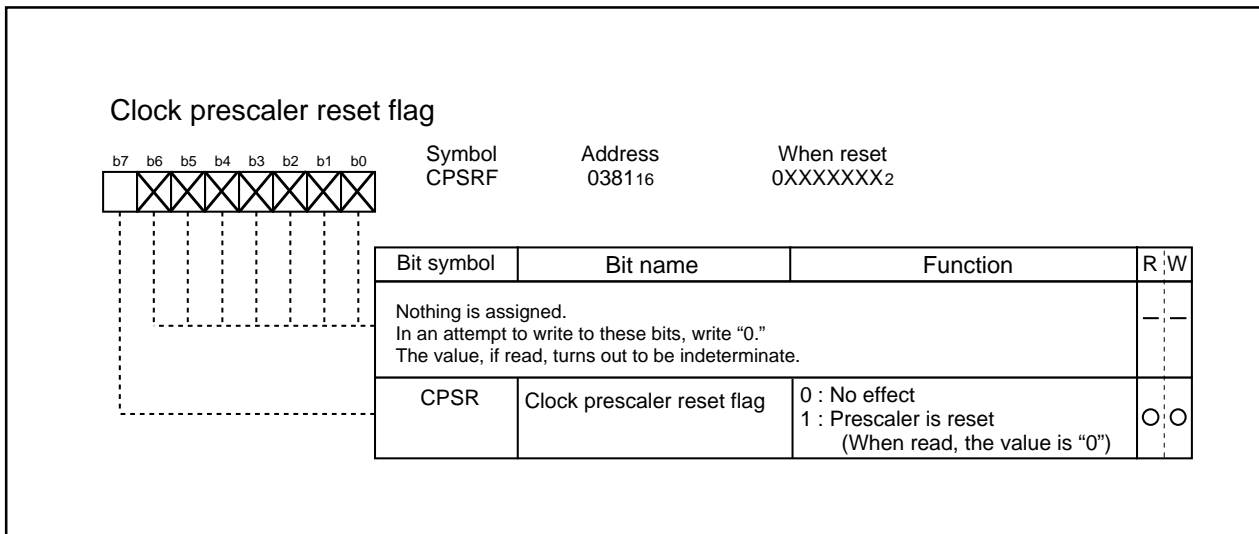


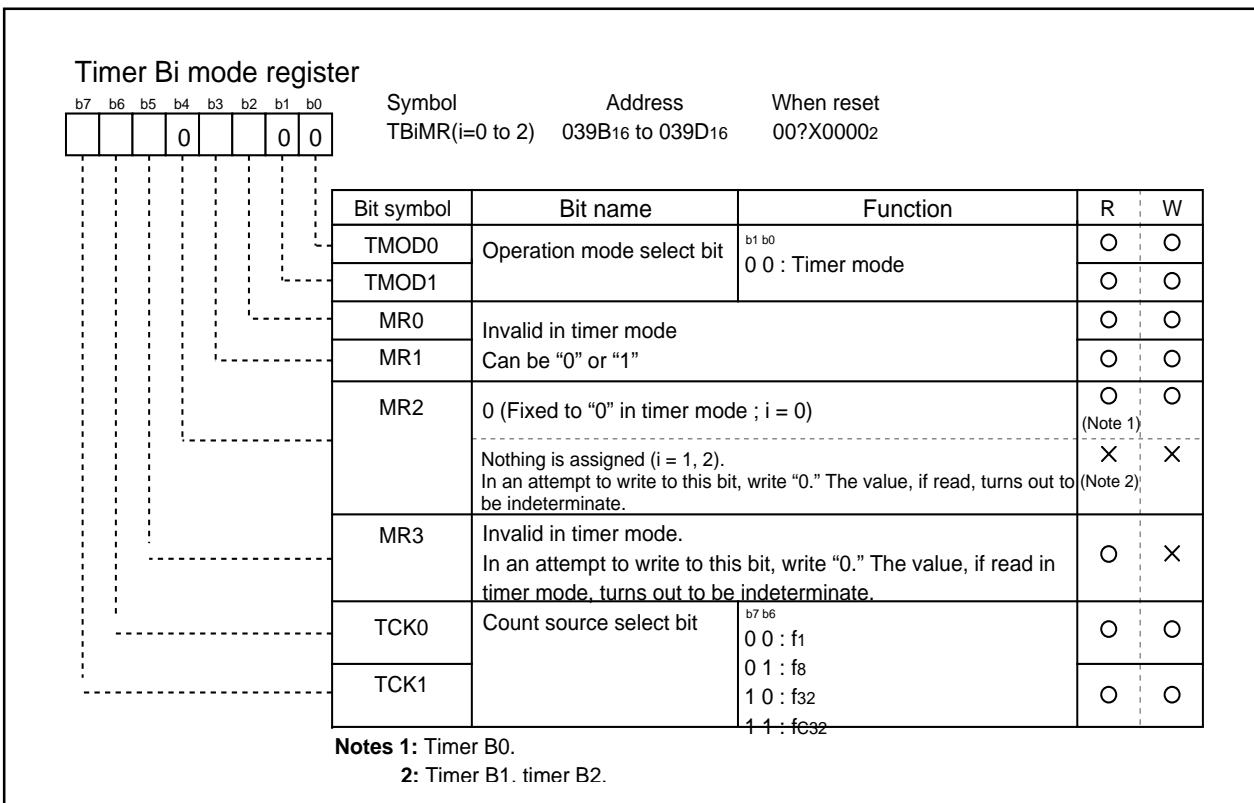
Figure 2.10.20 Clock prescaler reset flag

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



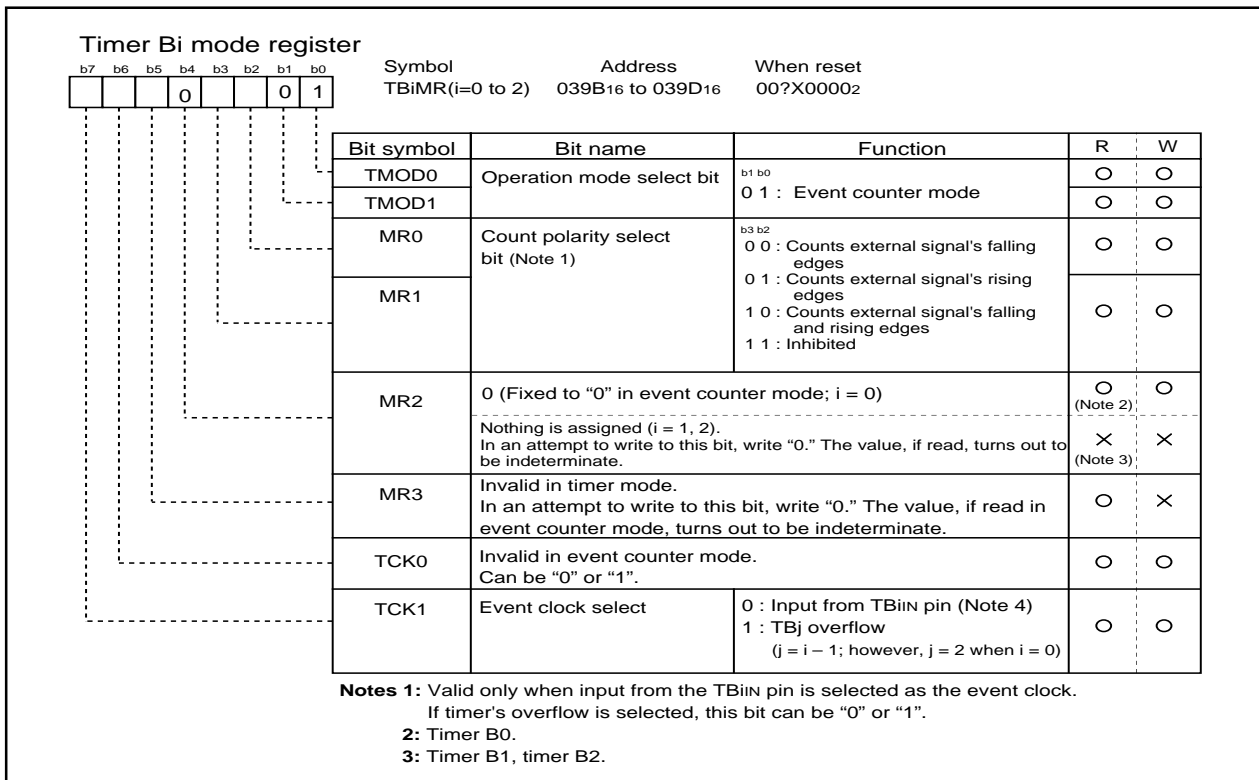
**Figure 2.10.21 Timer Bi mode register in timer mode (i = 0 to 2)**

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



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

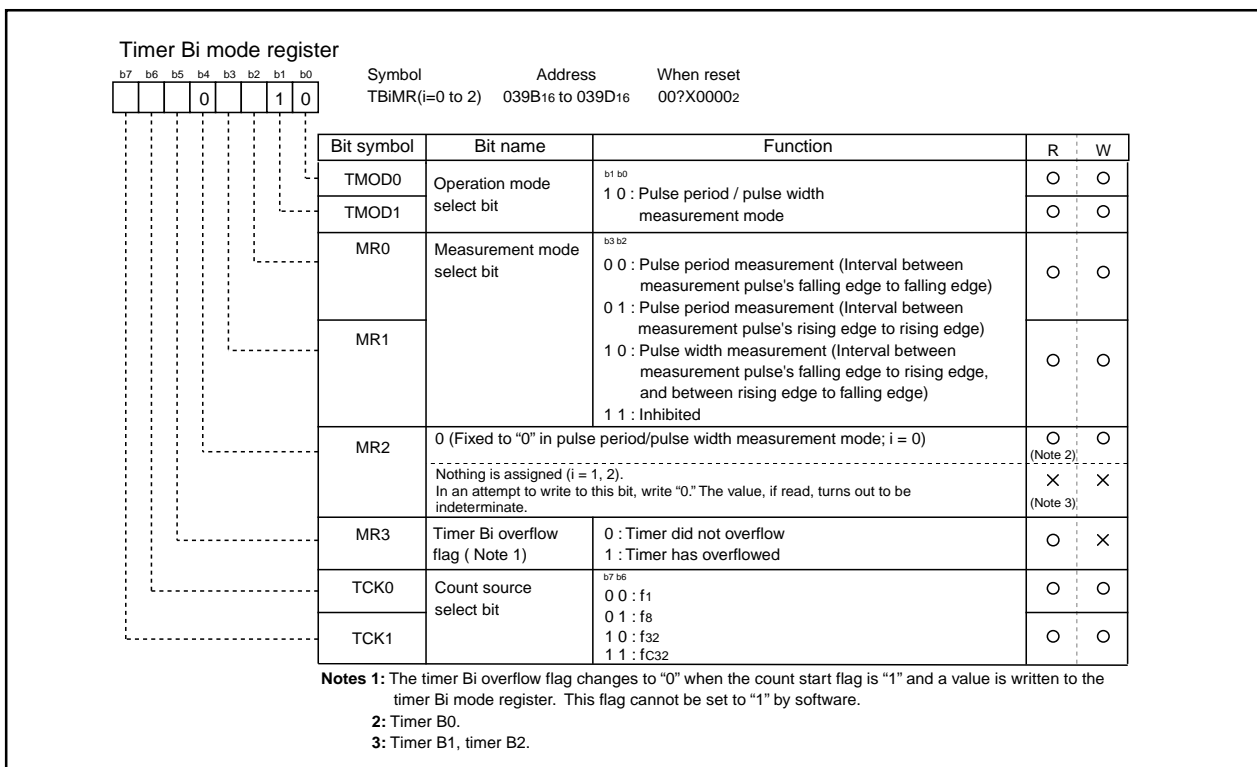
### (3) Pulse period/pulse width measurement mode

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**

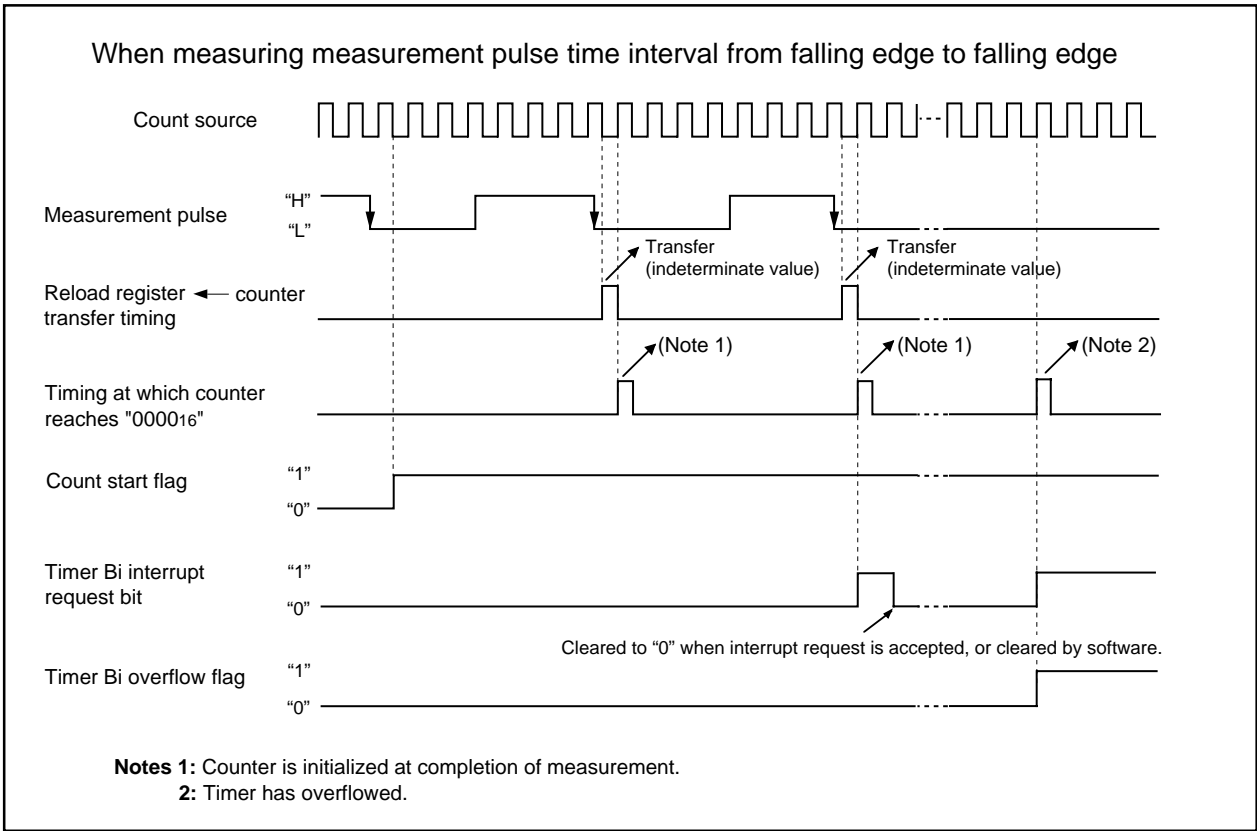
Item	Specification
Count source	f1, f8, f32, fc32
Count operation	<ul style="list-style-type: none"> <li>Up count</li> <li>Counter value "0000<sub>16</sub>" is transferred to reload register at measurement pulse's effective edge and the timer continues counting</li> </ul>
Count start condition	Count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

**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.

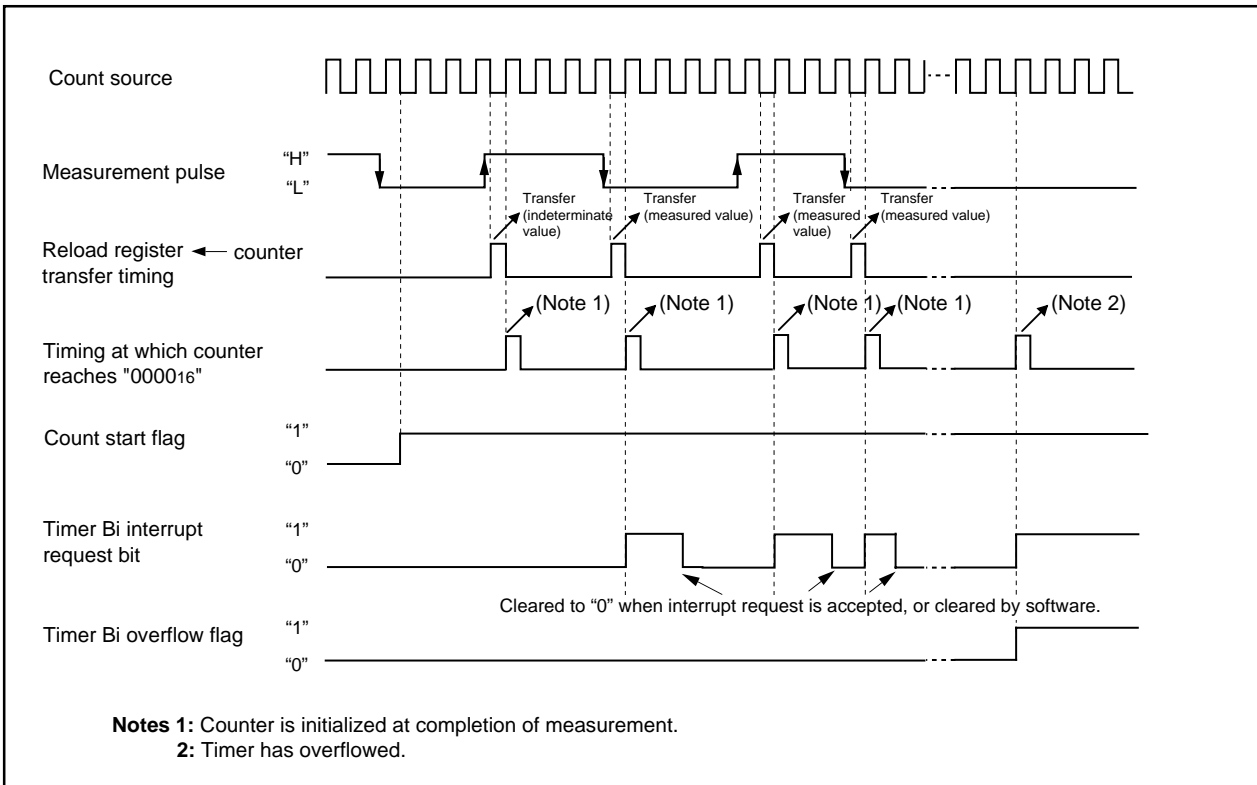


**Figure 2.10.23 Timer Bi mode register in pulse period/pulse width measurement mode (i = 0 to 2)**





**Figure 2.10.24** Operation timing when measuring a pulse period



**Figure 2.10.25** Operation timing when measuring a pulse width

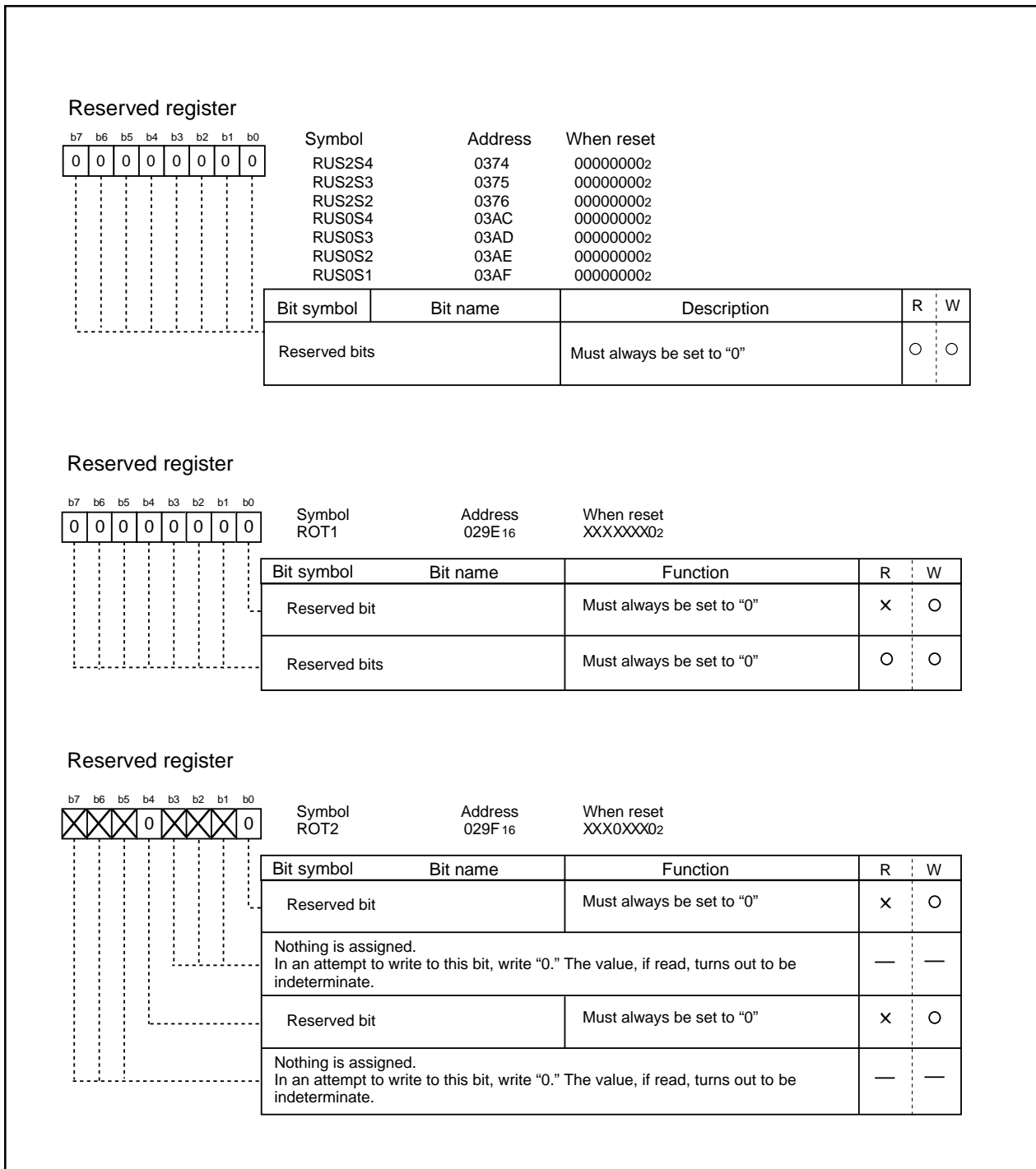


Figure 2.10.26 Reserved register

#### (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.

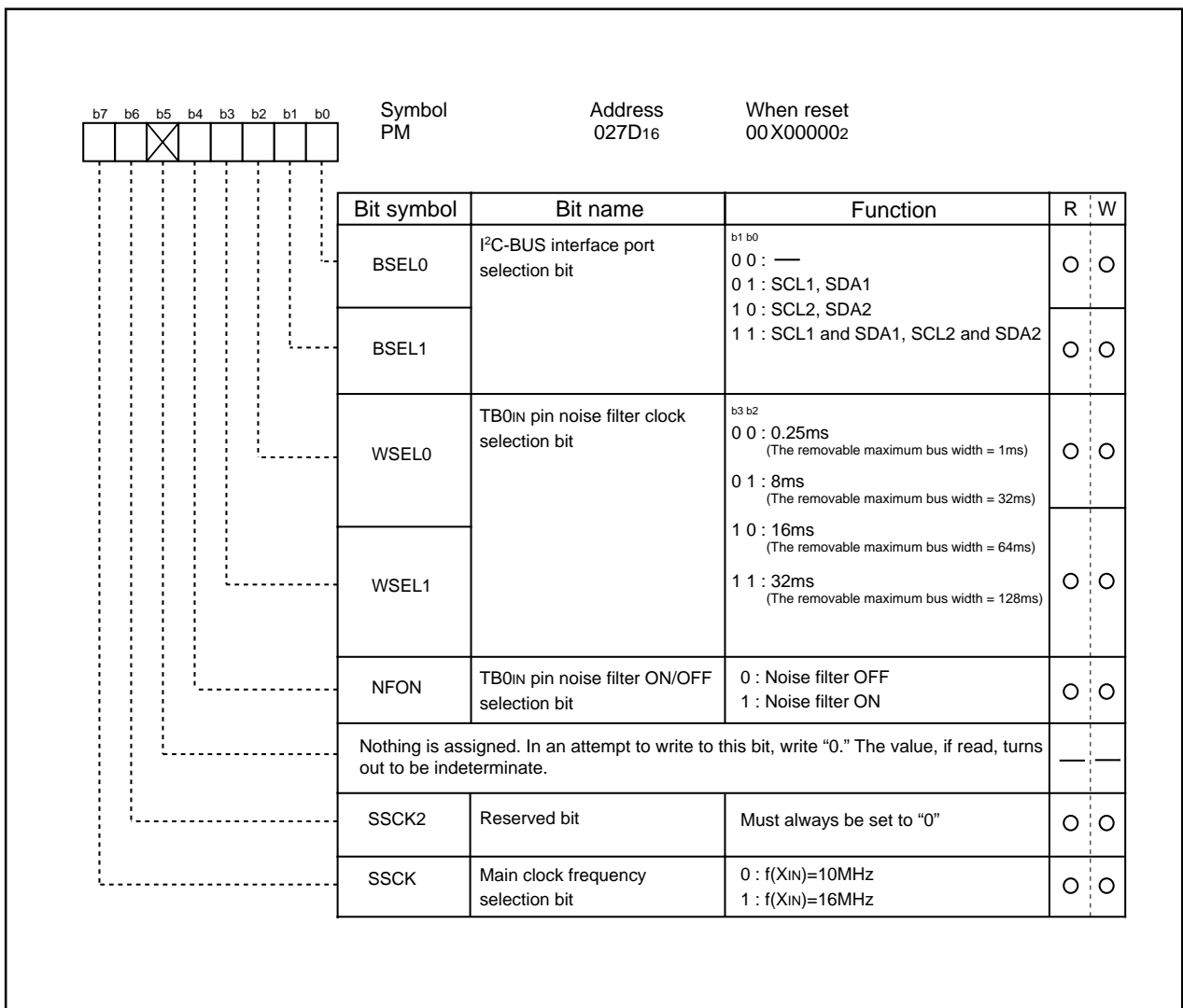


Figure 2.10.27 Peripheral mode register

## 2.11 Serial I/O

Serial I/O is configured as 4 unites: UART0, UART2, multi-master I<sup>2</sup>C-BUS interface 0, and multi-master I<sup>2</sup>C-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 03A0<sub>16</sub> and 0378<sub>16</sub>) 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, 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 2.11.13 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
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
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.

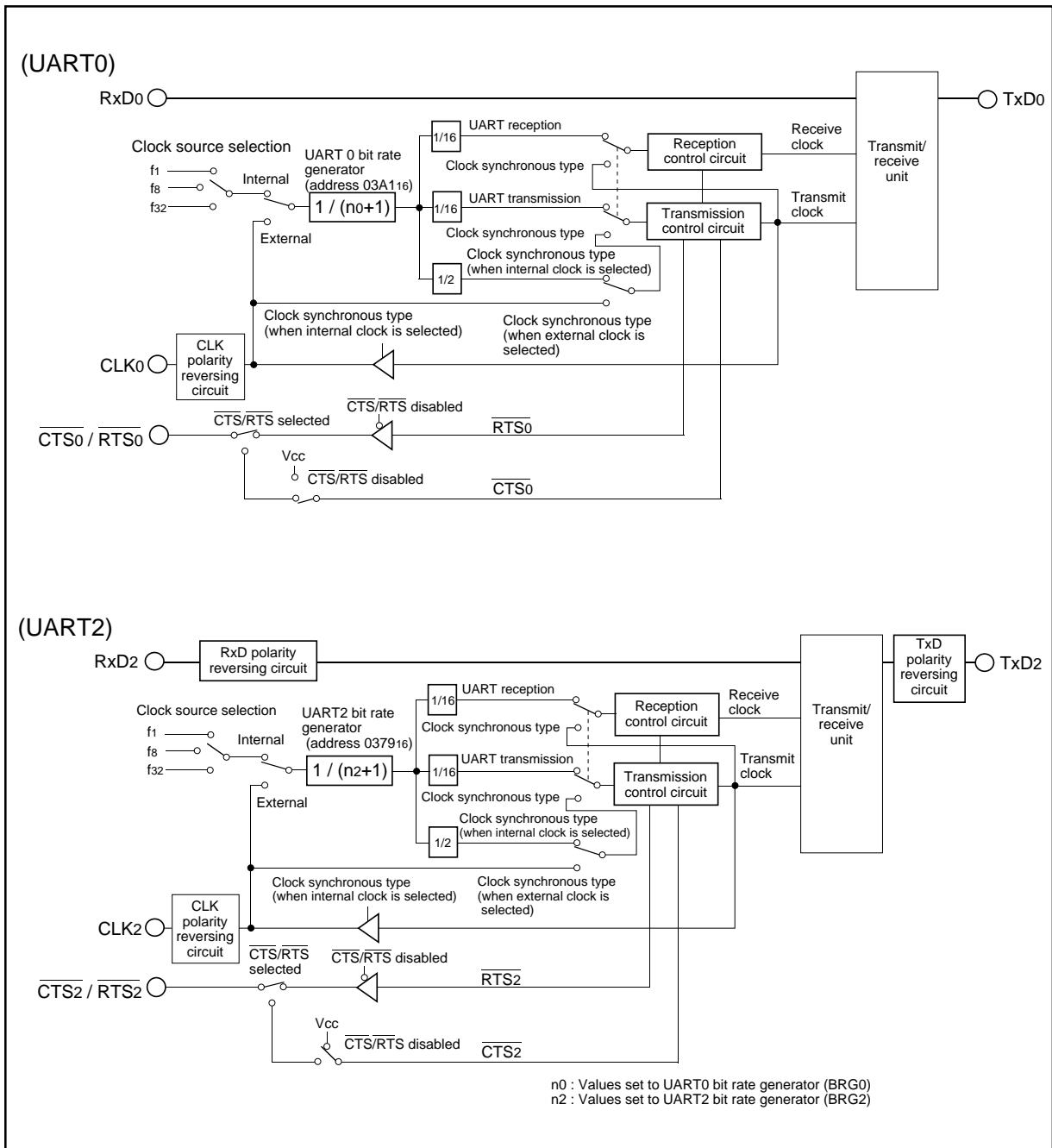


Figure 2.11.1 Block diagram of UARTi (i = 0 and 2)

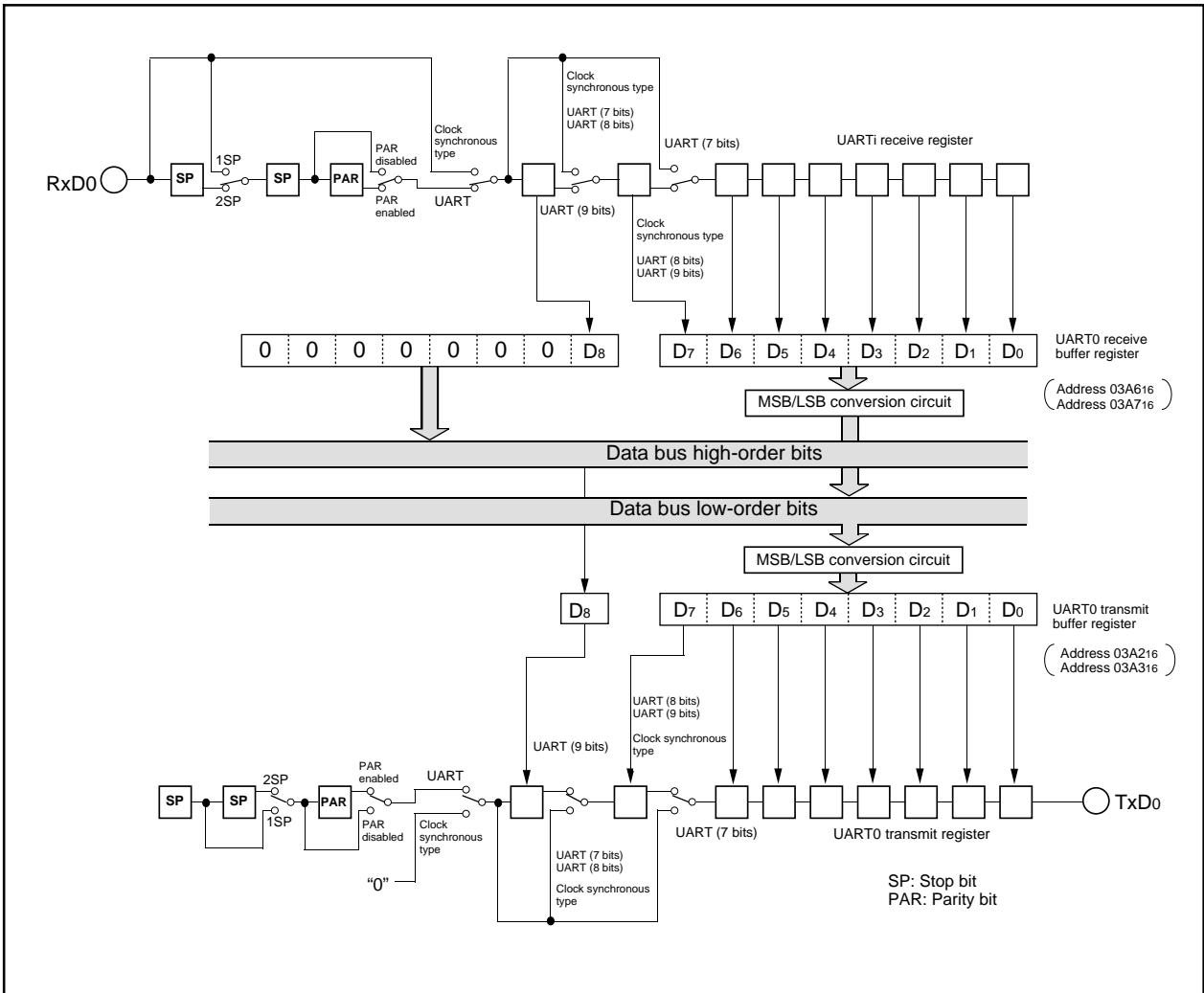


Figure 2.11.2 Block diagram of UART0 transmit/receive unit

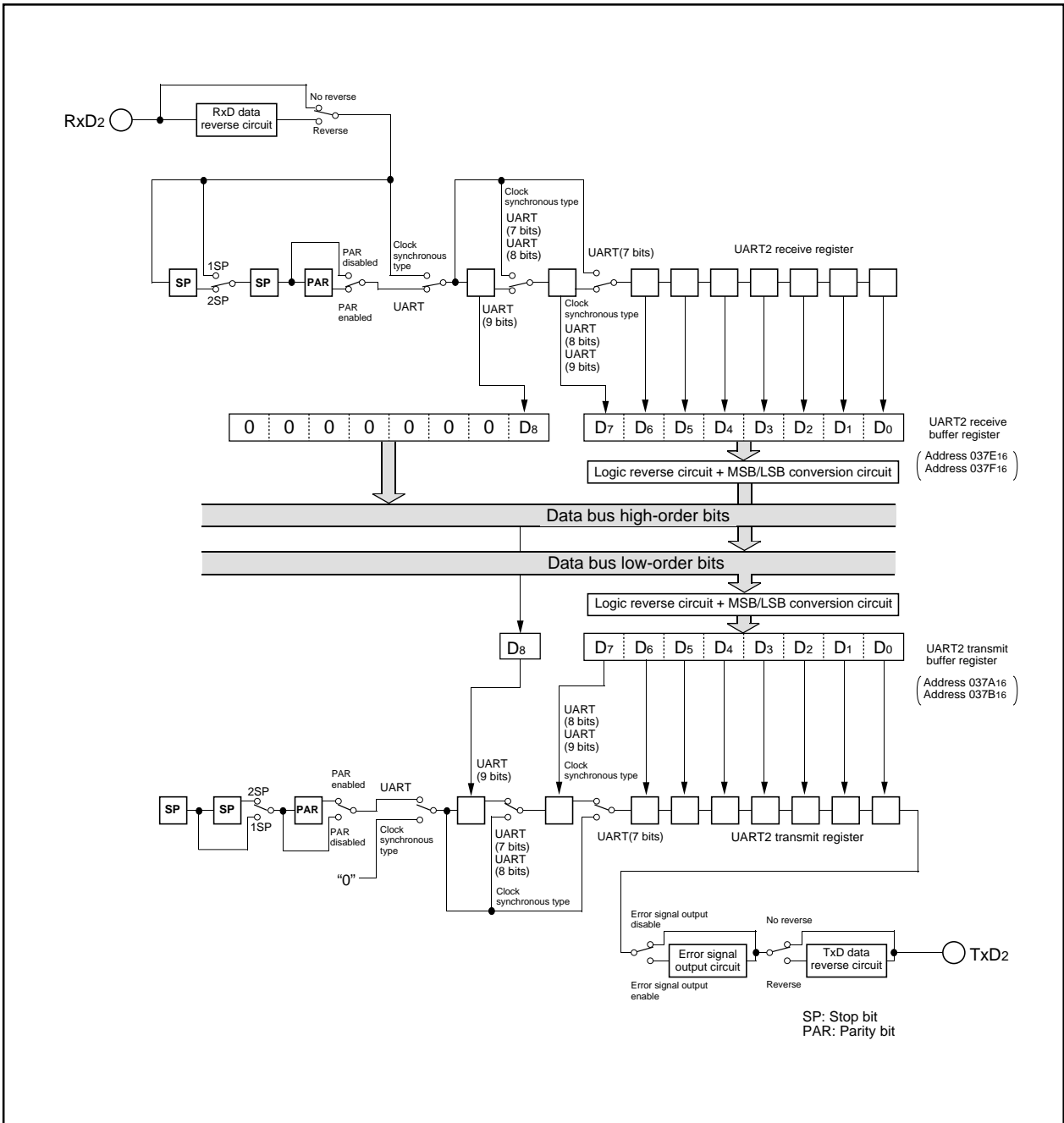


Figure 2.11.3 Block diagram of UART2 transmit/receive unit

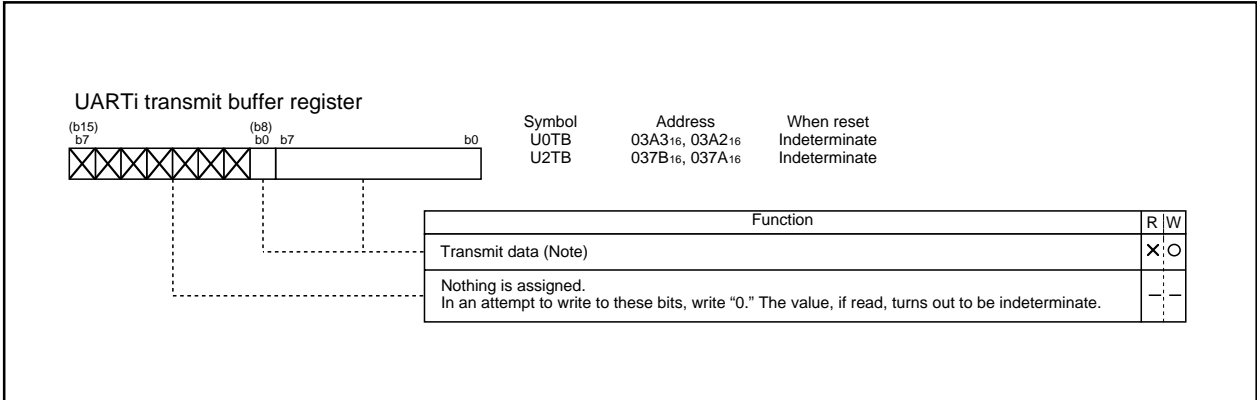


Figure 2.11.4 UARTi transmit buffer register (i = 0 and 2)

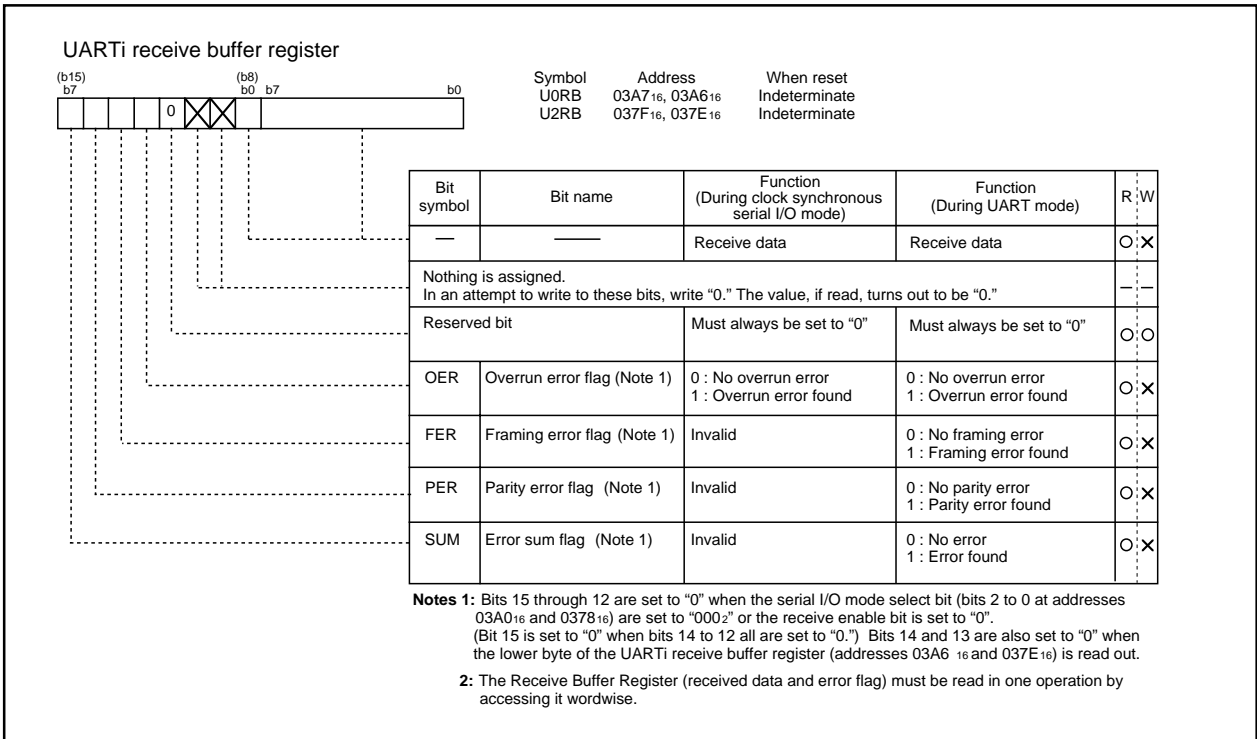


Figure 2.11.5 UARTi receive buffer register (i = 0 and 2)

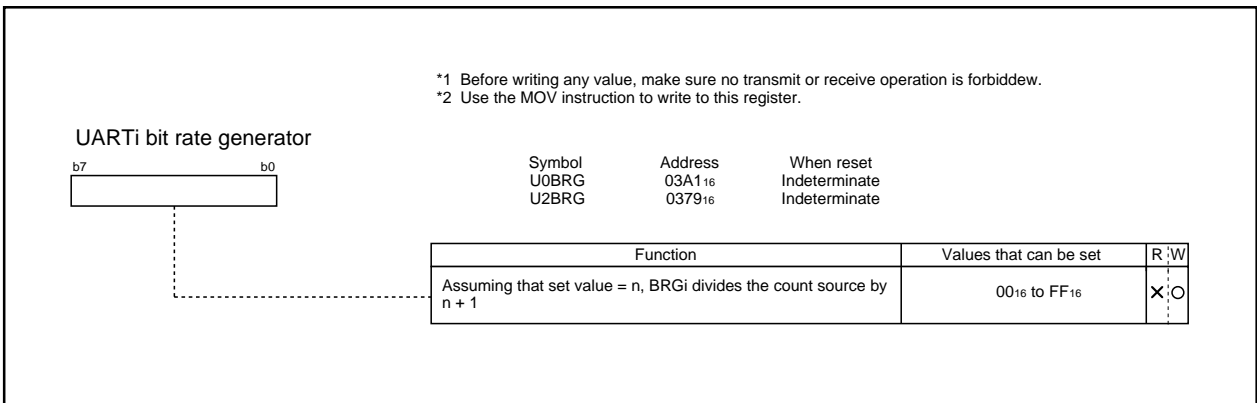


Figure 2.11.6 UARTi bit rate generator (i = 0 and 2)



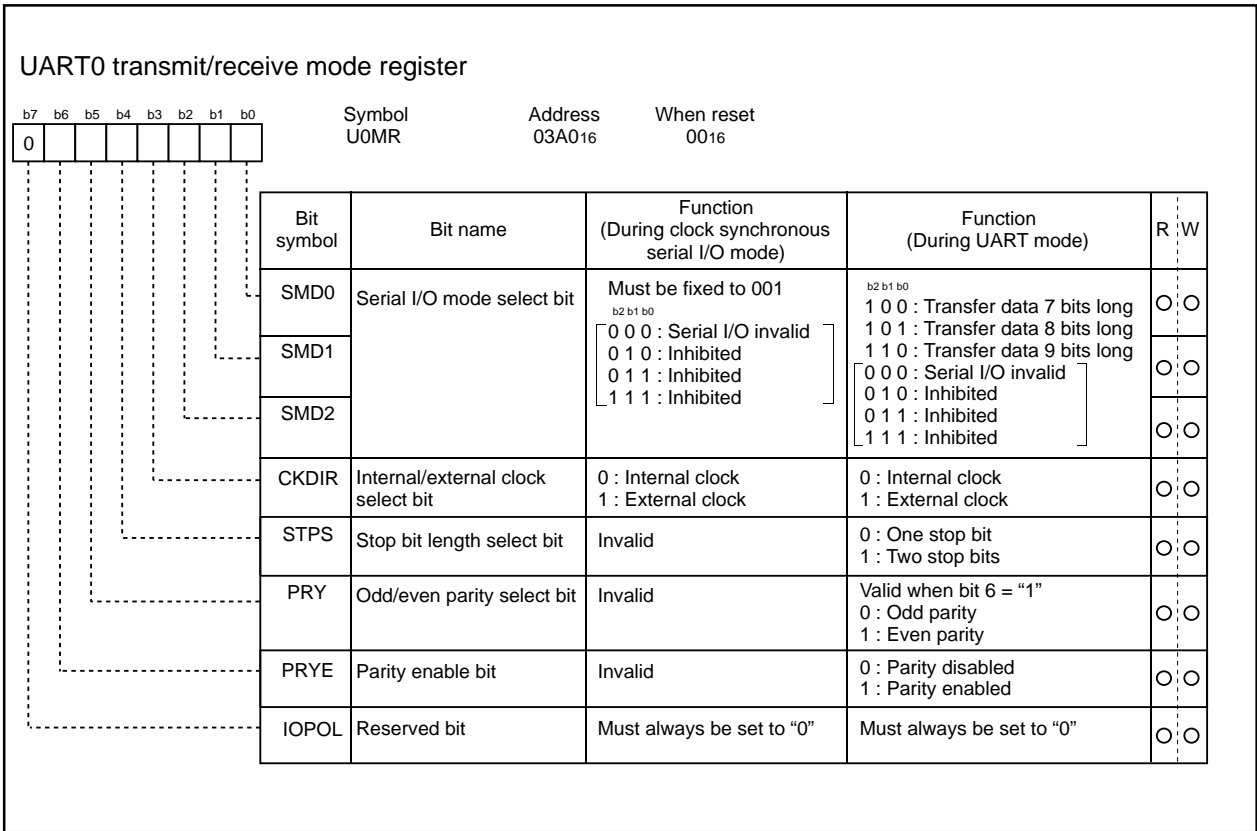


Figure 2.11.7 UART0 transmit/receive mode register

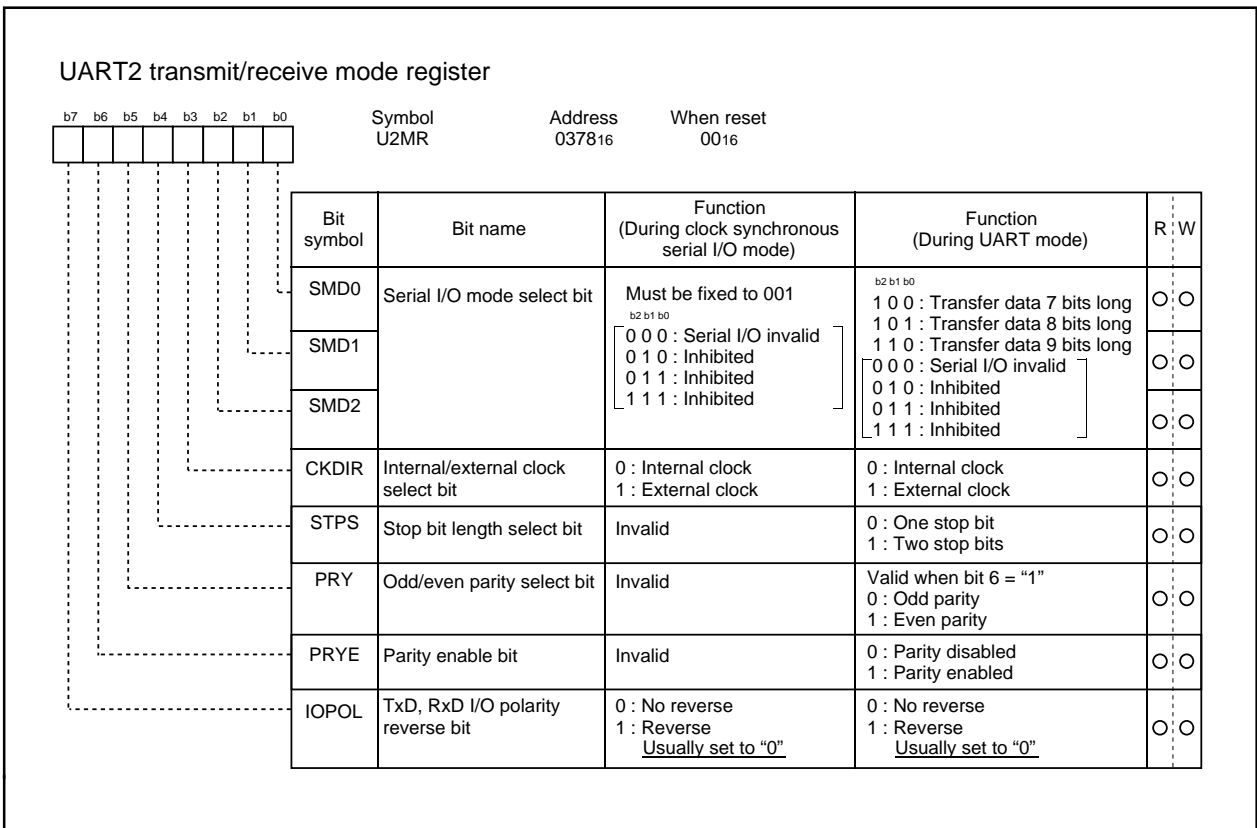


Figure 2.11.8 UART2 transmit/receive mode register

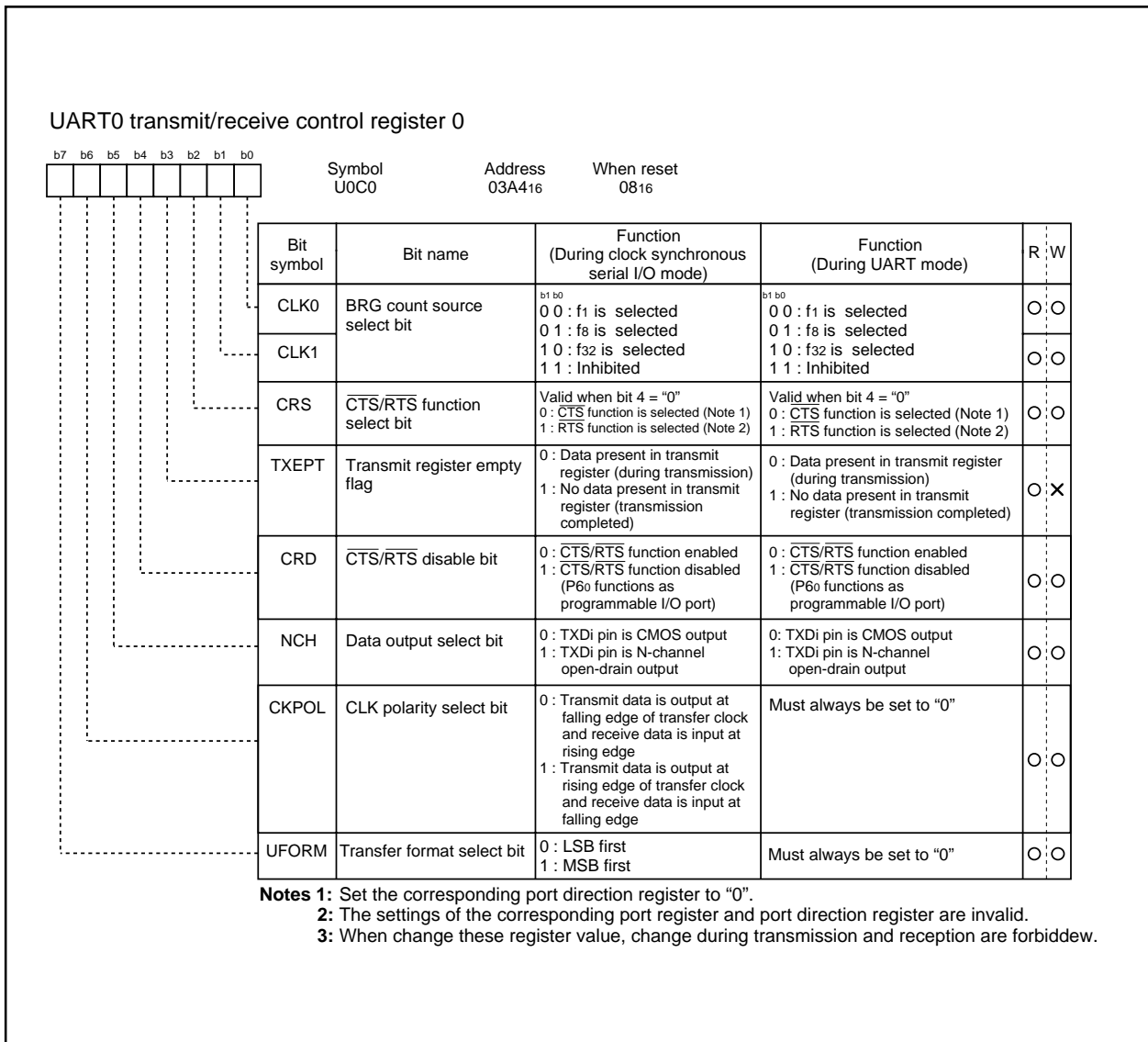
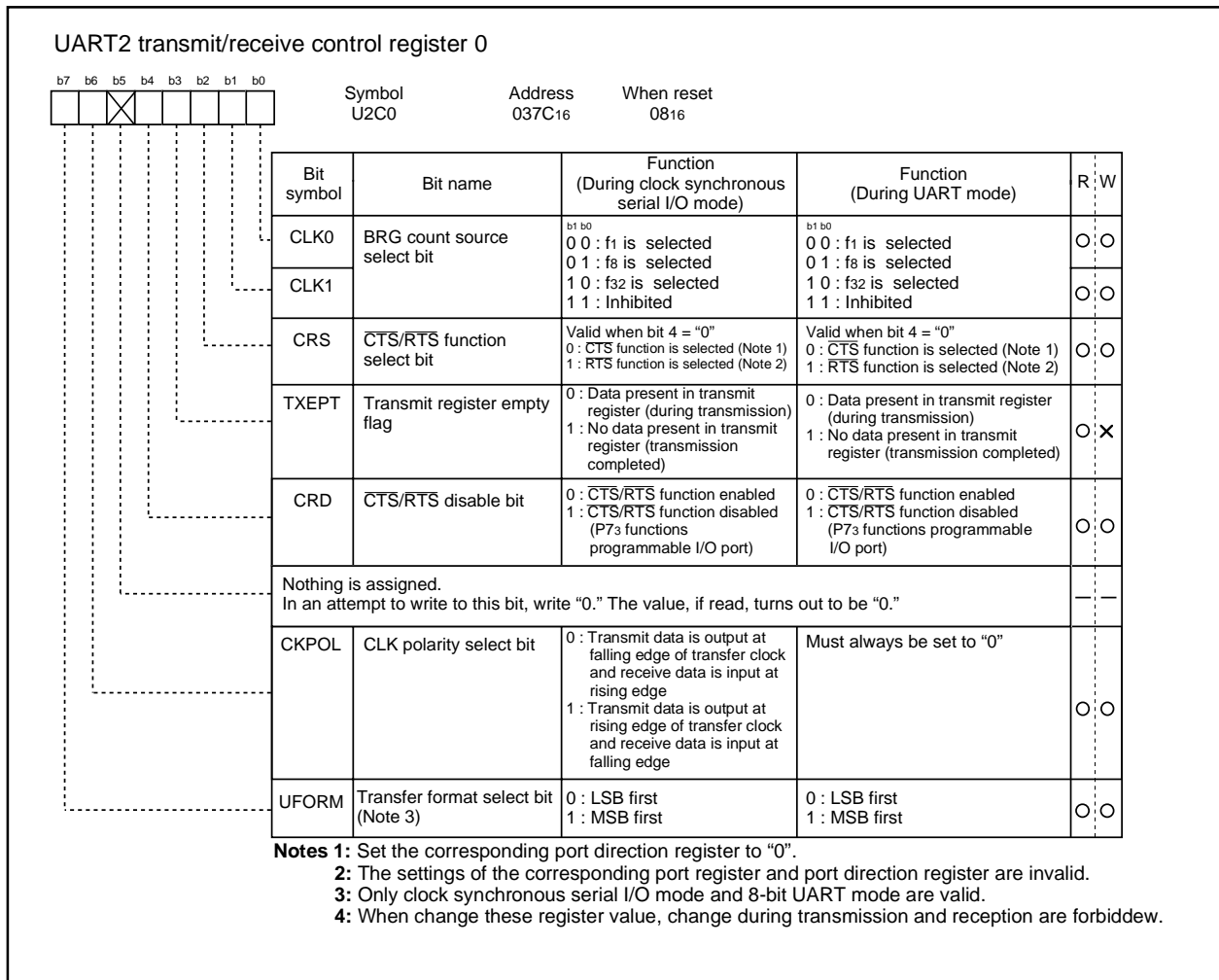
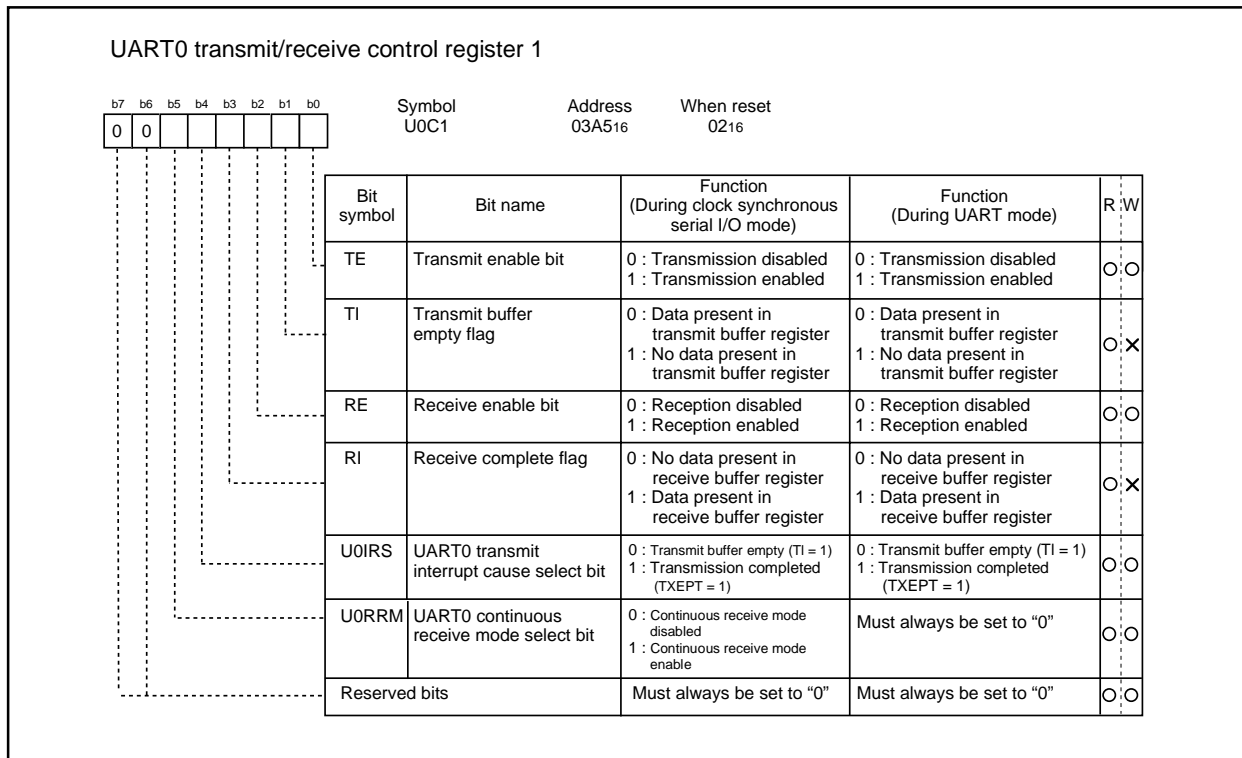


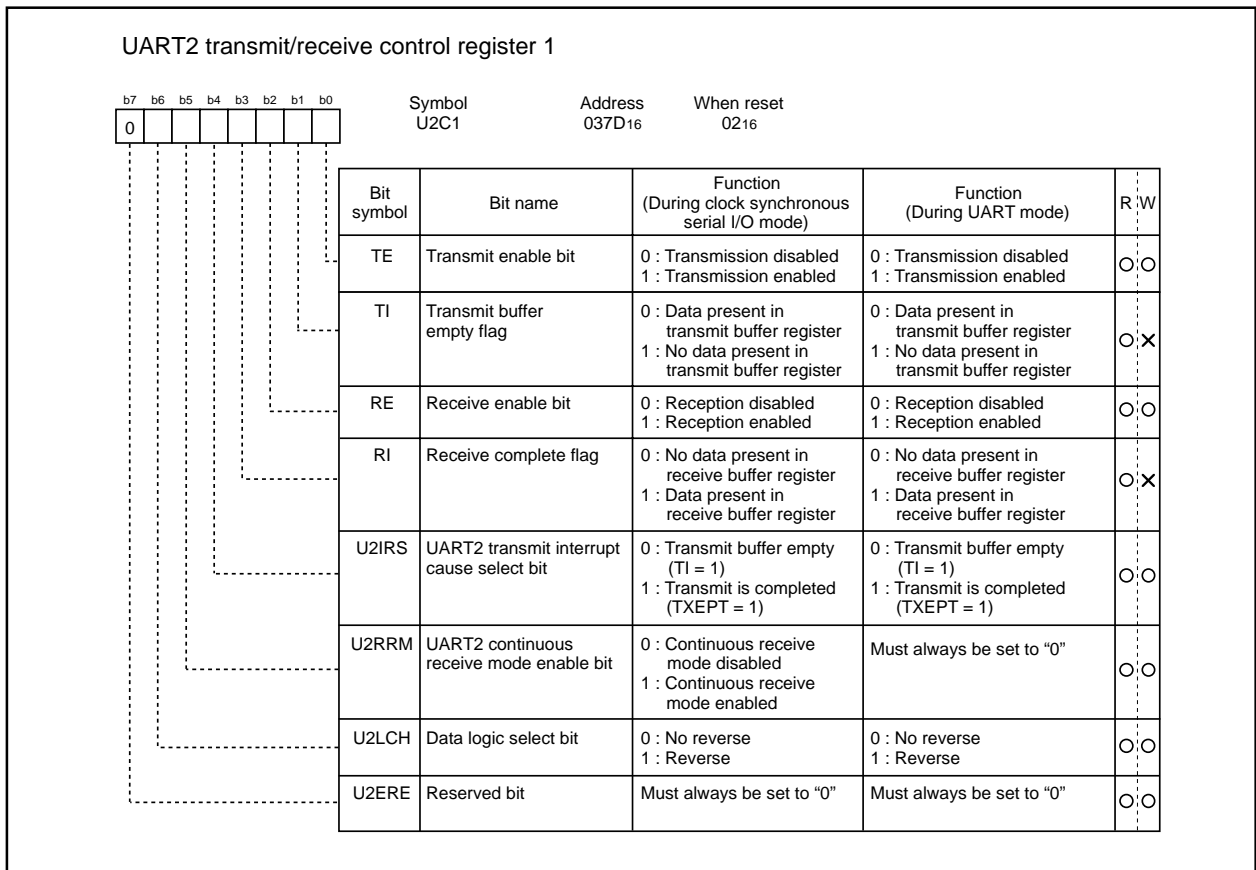
Figure 2.11.9 UART0 transmit/receive control register 0



**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**

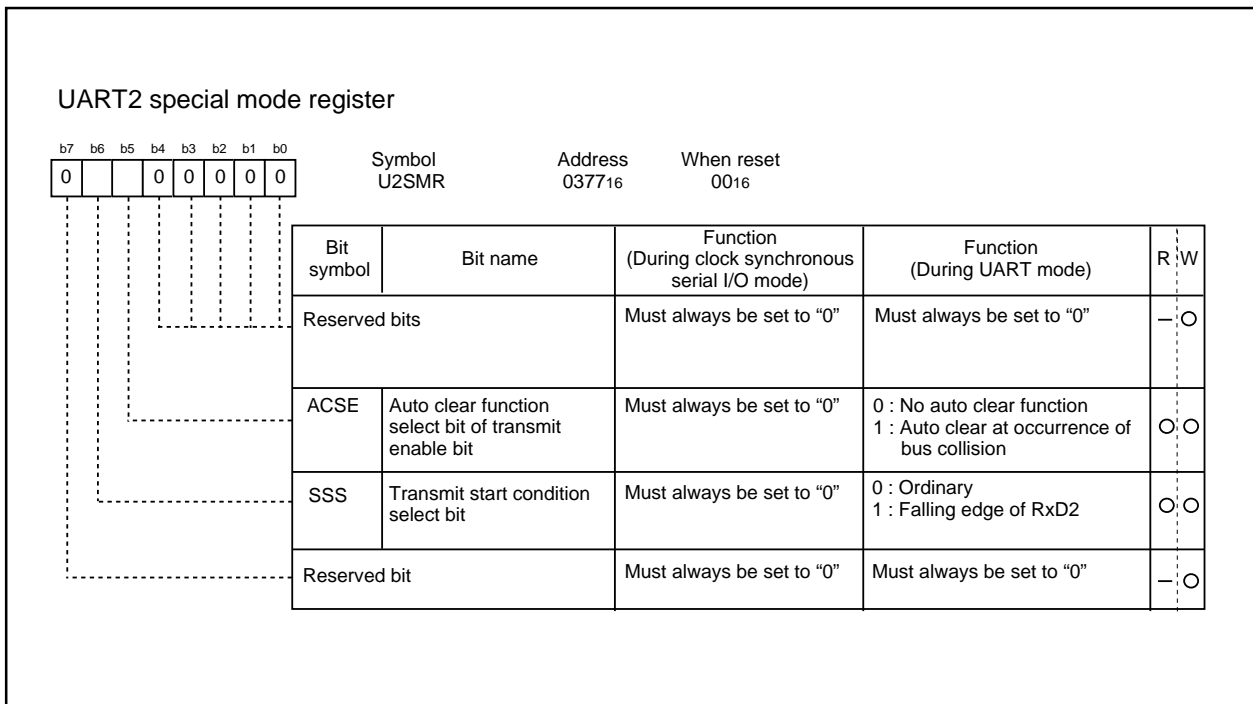


Figure 2.11.13 UART2 special mode register

## 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 2.11.14 and 2.11.15 show the UART<sub>i</sub> transmit/receive mode register in clock synchronous serial I/O mode.

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

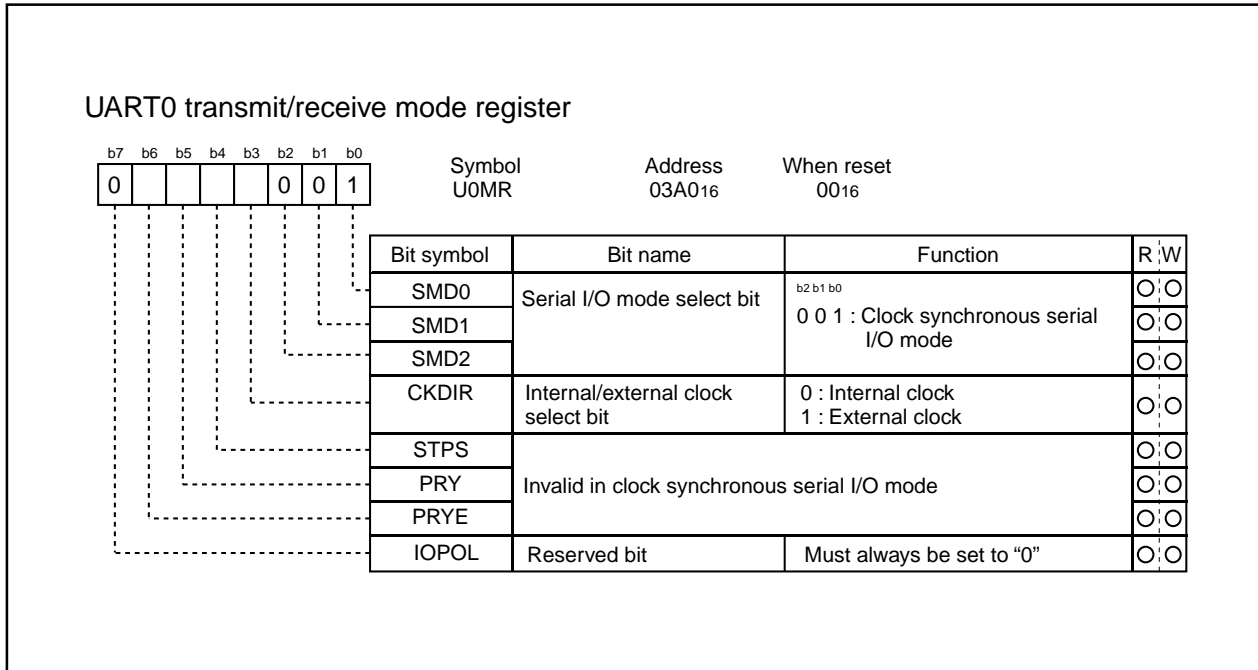
Item	Specification
Transfer data format	<ul style="list-style-type: none"> <li>Transfer data length: 8 bits</li> </ul>
Transfer clock	<ul style="list-style-type: none"> <li>When internal clock is selected (bit 3 at addresses 03A0<sub>16</sub>, 0378<sub>16</sub> = "0") :  <math>f_i / 2^{(n+1)}</math>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

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

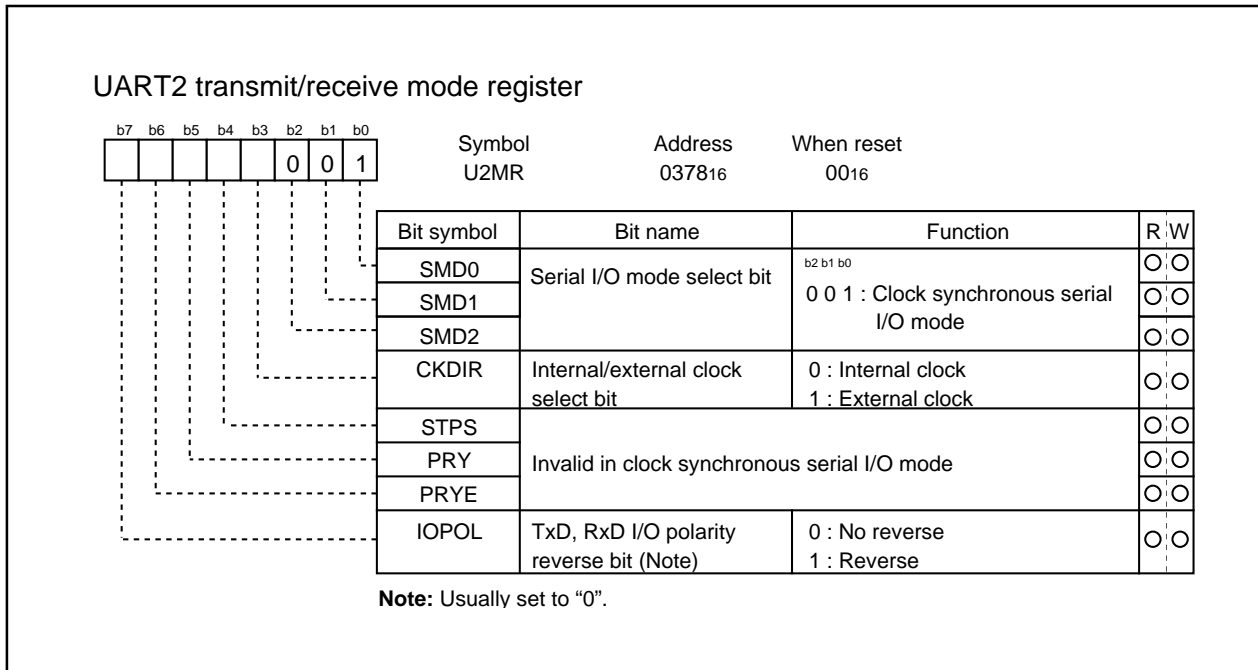
Item	Specification
Select function	<ul style="list-style-type: none"> <li>• CLK polarity selection Whether transmit data is output/input at the rising edge or falling edge of the transfer clock can be selected</li> <li>• LSB first/MSB first selection Whether transmission/reception begins with bit 0 or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

**Notes 1:** “n” denotes the value 00<sub>16</sub> to FF<sub>16</sub> that is set to the UART bit rate generator.

**2:** If an overrun error occurs, the UART<sub>i</sub> receive buffer will have the next data written in. Note also that the UART<sub>i</sub> receive interrupt request bit is not set to “1”.



**Figure 2.11.14 UART0 transmit/receive mode registers in clock synchronous serial I/O mode**



**Figure 2.11.15 UART2 transmit/receive mode register in clock synchronous serial I/O mode**

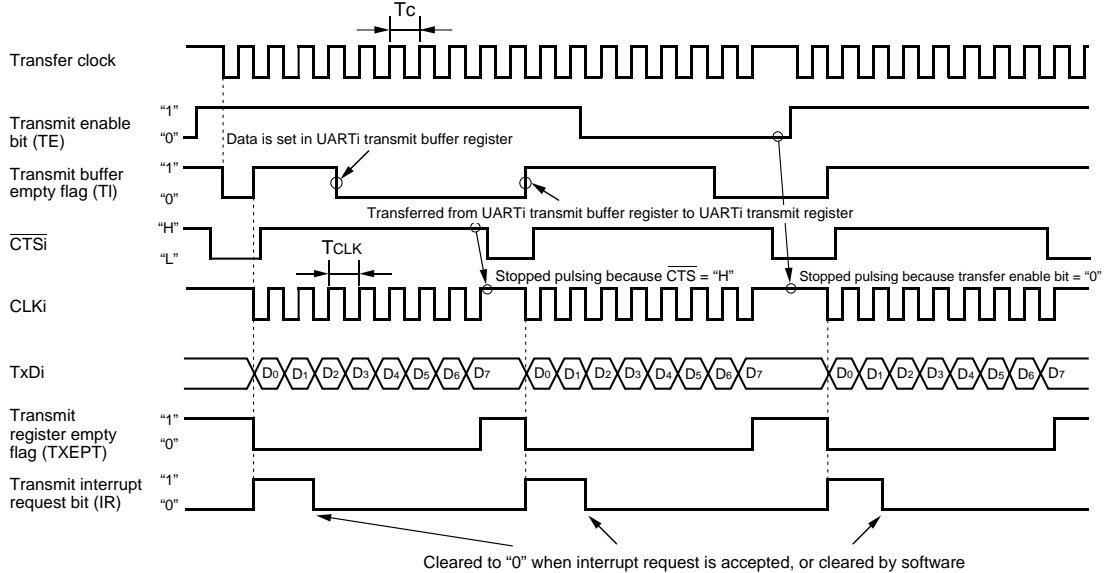


Table 2.11.4 lists the functions of the input/output pins during clock synchronous serial I/O mode. Note that for a period from when the UARTi operation mode is selected to when transfer starts, the TxDi pin outputs a “H”. (If the N-channel open-drain is selected, this pin is in floating state.)

**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”
CTS <sub>i</sub> /RTS <sub>i</sub> (P60, P73)	CTS input	CTS/RTS disable bit (bit 4 at address 03A4 16, 037C 16) = “0” CTS/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”
	RTS output	CTS/RTS disable bit (bit 4 at address 03A4 16, 037C 16) = “0” CTS/RTS function select bit (bit 2 at address 03A4 16, 037C 16) = “1”
	Programmable I/O port	CTS/RTS disable bit (bit 4 at address 03A4 16, 037C 16) = “1”

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



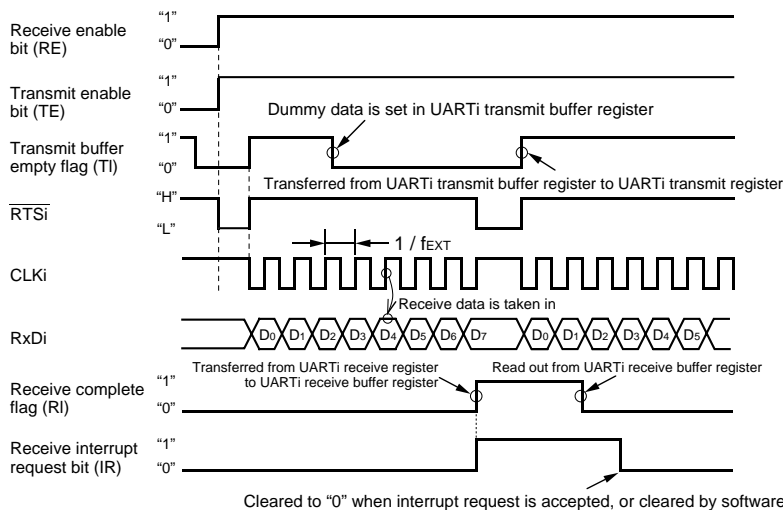
Shown in ( ) are bit symbols.

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

$$T_c = T_{CLK} = 2(n + 1) / f_i$$

$f_i$ : frequency of BRGi count source ( $f_1, f_8, f_{32}$ )  
 $n$ : value set to BRGi

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



Shown in ( ) are bit symbols.

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

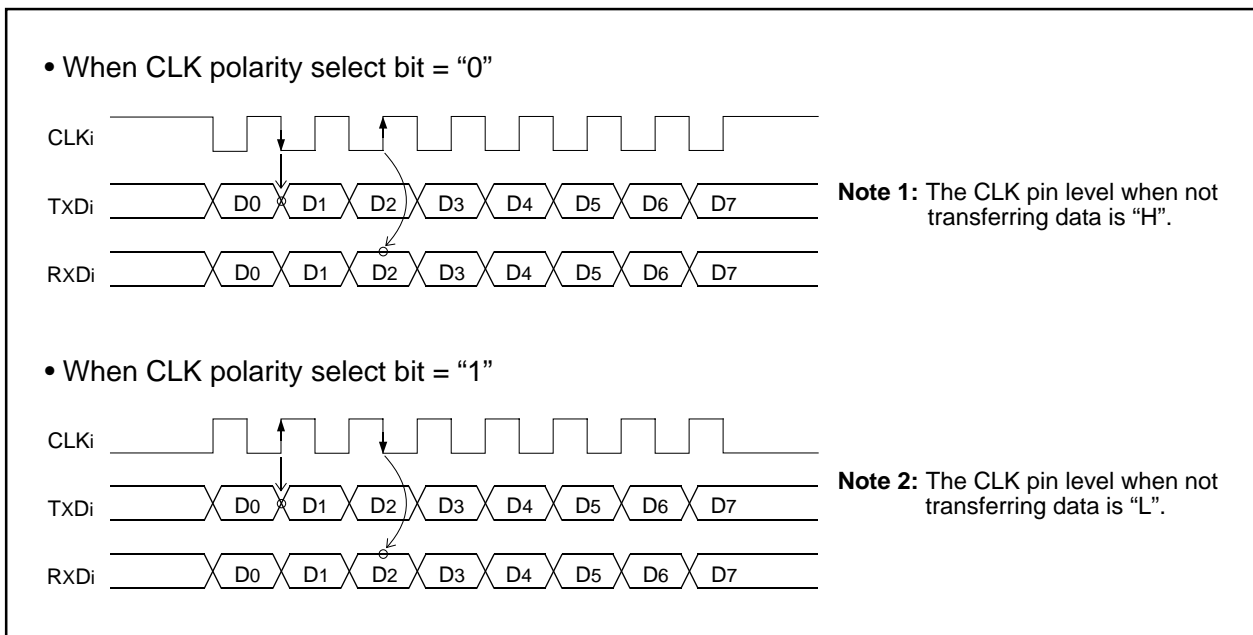
$f_{EXT}$ : frequency of external clock

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

Figure 2.11.16 Typical transmit/receive timings in clock synchronous serial I/O mode

**(1) Polarity select function**

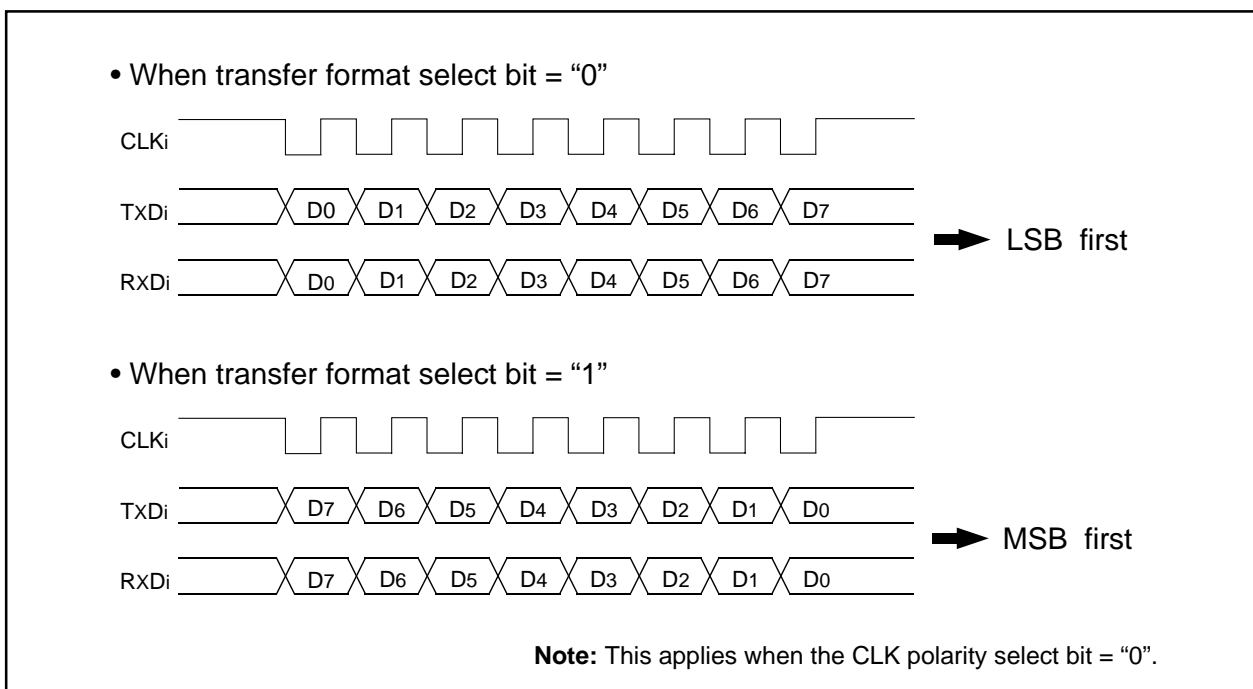
As shown in Figure 2.11.17, the CLK polarity select bit (bit 6 at addresses 03A416, 037C16) allows selection of the polarity of the transfer clock.



**Figure 2.11.17 Polarity of transfer clock**

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

As shown in Figure 2.11.18, 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”.



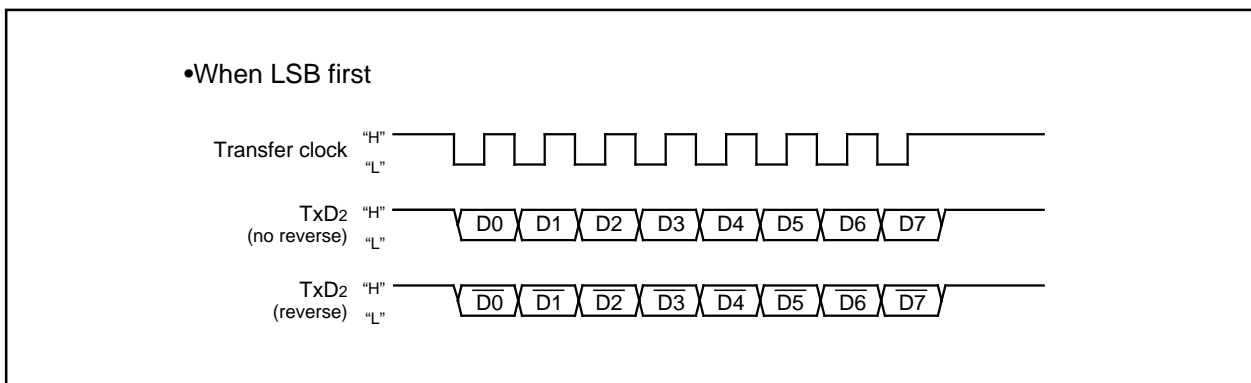
**Figure 2.11.18 Transfer format**

**(3) Continuous receive mode**

If the continuous receive mode enable bit (bits 2 at address 03A516, 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 2.11.19 shows the example of serial data logic switch timing.



**Figure 2.11.19 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 2.11.20 and 2.11.21 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"> <li>• Character bit (transfer data): 7 bits, 8 bits, or 9 bits as selected</li> <li>• Start bit: 1 bit</li> <li>• Parity bit: Odd, even, or nothing as selected</li> <li>• Stop bit: 1 bit or 2 bits as selected</li> </ul>
Transfer clock	<ul style="list-style-type: none"> <li>• When internal clock is selected (bit 3 at addresses 03A0<sub>16</sub>, 0378<sub>16</sub> = "0") :  <math>f_i/16(n+1)</math> (Note 1) <math>f_i = f_1, f_8, f_{32}</math></li> <li>• When external clock is selected (bit 3 at addresses 03A0<sub>16</sub>, 0378<sub>16</sub> = "1") :  <math>f_{EXT}/16(n+1)</math>(Note 1) (Note 2)</li> </ul>
Transmission/reception control	<ul style="list-style-type: none"> <li>• CTS function/RTS function/CTS, RTS function chosen to be invalid</li> </ul>
Transmission start condition	<ul style="list-style-type: none"> <li>• To start transmission, the following requirements must be met:            Transmit enable bit (bit 0 at addresses 03A5<sub>16</sub>, 037D<sub>16</sub>) = "1"            Transmit buffer empty flag (bit 1 at addresses 03A5<sub>16</sub>, 037D<sub>16</sub>) = "0"            When CTS function selected, <math>\overline{CTS}</math> input level = "L"</li> </ul>
Reception start condition	<ul style="list-style-type: none"> <li>• To start reception, the following requirements must be met:            Receive enable bit (bit 2 at addresses 03A5<sub>16</sub>, 037D<sub>16</sub>) = "1"            Start bit detection</li> </ul>
Interrupt request generation timing	<ul style="list-style-type: none"> <li>• When transmitting            Transmit interrupt cause select bits (bits 0 at address 03B0<sub>16</sub>, bit4 at address 037D<sub>16</sub>) = "0": Interrupts requested when data transfer from UARTi transfer buffer register to UARTi transmit register is completed            Transmit interrupt cause select bits (bits 0 at address 03B0<sub>16</sub>, bit4 at address 037D<sub>16</sub>) = "1": Interrupts requested when data transmission from UARTi transfer register is completed</li> <li>• When receiving            Interrupts requested when data transfer from UARTi receive register to UARTi receive buffer register is completed</li> </ul>
Error detection	<ul style="list-style-type: none"> <li>• Overrun error (Note 3)            This error occurs when the next data is ready before contents of UARTi receive buffer register are read out</li> <li>• Framing error            This error occurs when the number of stop bits set is not detected</li> <li>• 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</li> <li>• Error sum flag            This flag is set (= 1) when any of the overrun, framing, and parity errors is encountered</li> </ul>

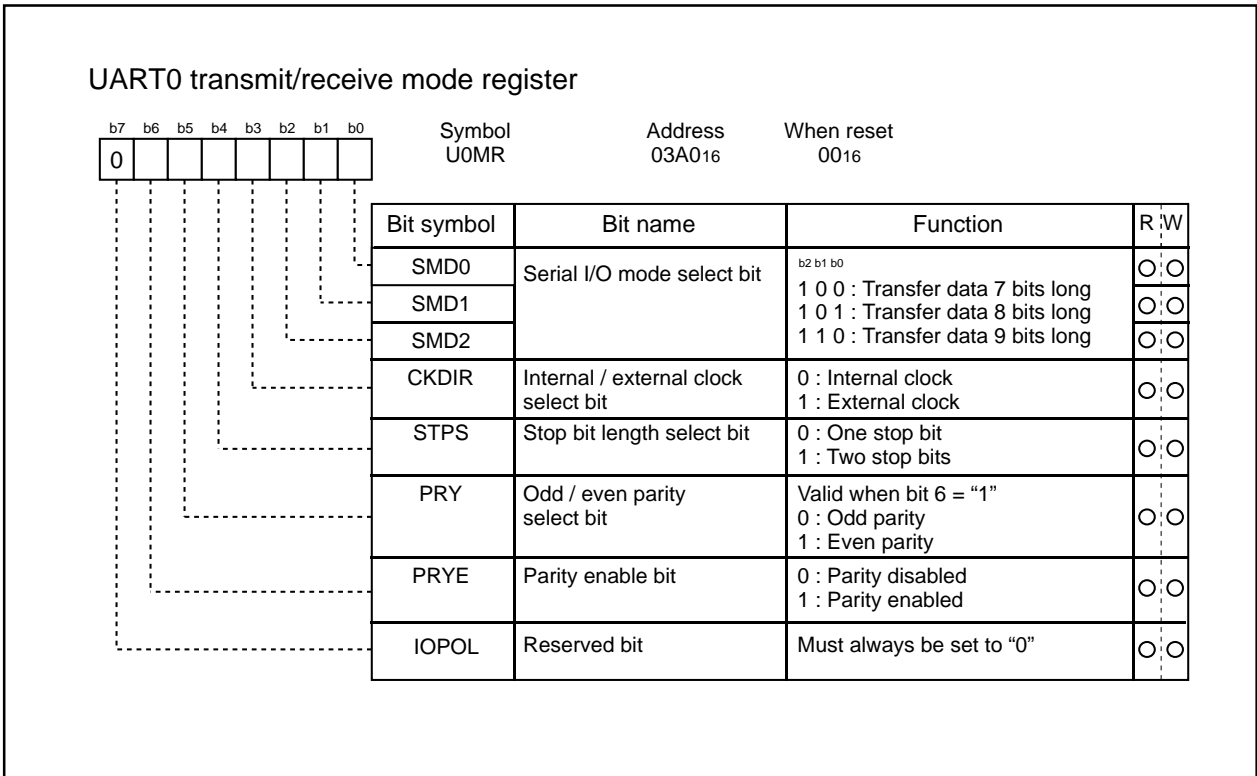
**Table 2.11.6 Specifications of UART Mode (2)**

Item	Specification
Select function	<ul style="list-style-type: none"><li data-bbox="507 398 1426 510">• Serial data logic switch (UART2) This function is reversing logic value of transferring data. Start bit, parity bit and stop bit are not reversed.</li><li data-bbox="507 517 1426 631">• TxD, RxD I/O polarity switch This function is reversing TxD port output and RxD port input. All I/O data level is reversed.</li></ul>

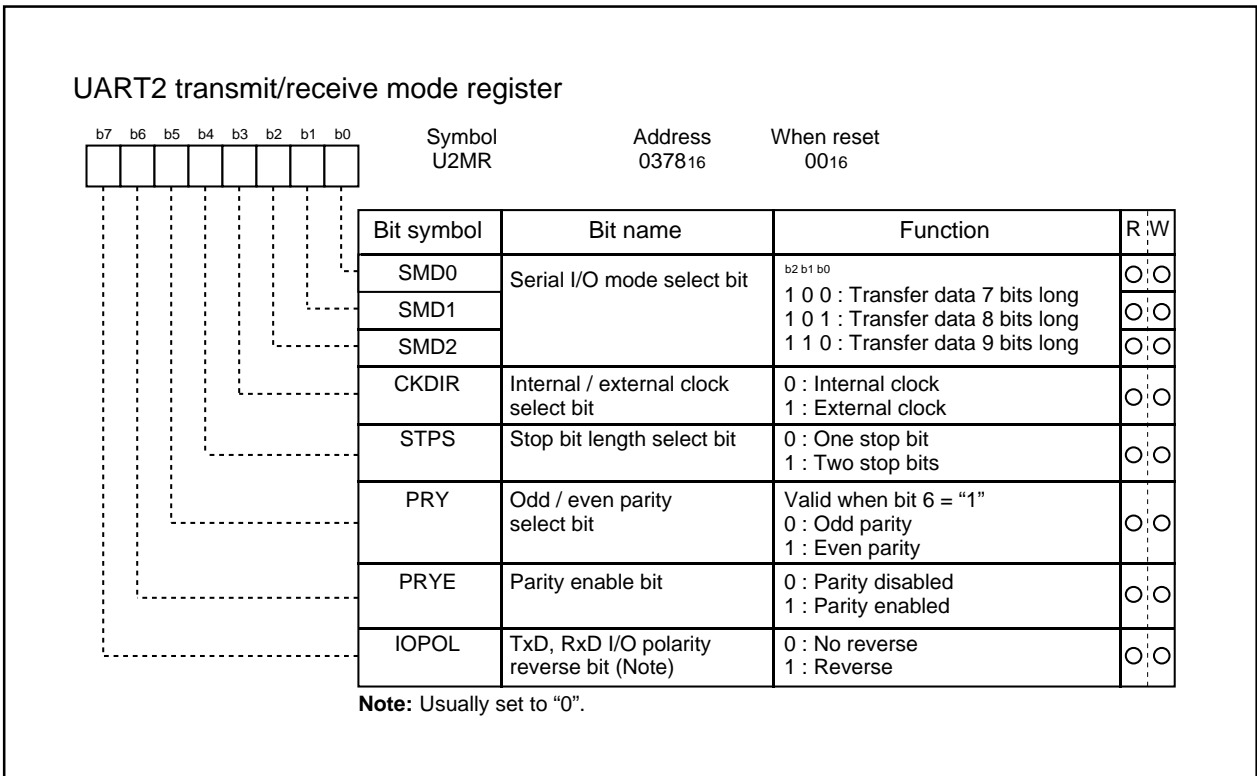
**Notes 1:** 'n' denotes the value 00<sub>16</sub> to FF<sub>16</sub> that is set to the UART<sub>i</sub> bit rate generator.

**2:** fEXT is input from the CLK<sub>i</sub> pin.

**3:** If an overrun error occurs, the UART<sub>i</sub> receive buffer will have the next data written in. Note also that the UART<sub>i</sub> receive interrupt request bit is not set to "1".



**Figure 2.11.20 UART0 transmit/receive mode register in UART mode**



**Figure 2.11.21 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 <sub>16</sub> , bit 1 at address 03EF <sub>16</sub> ) = "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 <sub>16</sub> , 0378 <sub>16</sub> ) = "0"
	Transfer clock input	Internal/external clock select bit (bit 3 at address 03A0 <sub>16</sub> , 0378 <sub>16</sub> ) = "1" Port P61 and P72 direction register (bit 1 at address 03EE <sub>16</sub> , bit 2 at address 03EF <sub>16</sub> ) = "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 <sub>16</sub> , 037C <sub>16</sub> ) = "0" $\overline{\text{CTS}}/\overline{\text{RTS}}$ function select bit (bit 2 at address 03A4 <sub>16</sub> , 037C <sub>16</sub> ) = "0" Port P60 and P73 direction register (bit 0 at address 03EE <sub>16</sub> , bit 3 at address 03EF <sub>16</sub> ) = "0"
	RTS output	$\overline{\text{CTS}}/\overline{\text{RTS}}$ disable bit (bit 4 at address 03A4 <sub>16</sub> , 037C <sub>16</sub> ) = "0" $\overline{\text{CTS}}/\overline{\text{RTS}}$ function select bit (bit 2 at address 03A4 <sub>16</sub> , 037C <sub>16</sub> ) = "1"
	Programmable I/O port	$\overline{\text{CTS}}/\overline{\text{RTS}}$ disable bit (bit 4 at address 03A4 <sub>16</sub> , 037C <sub>16</sub> ) = "1"



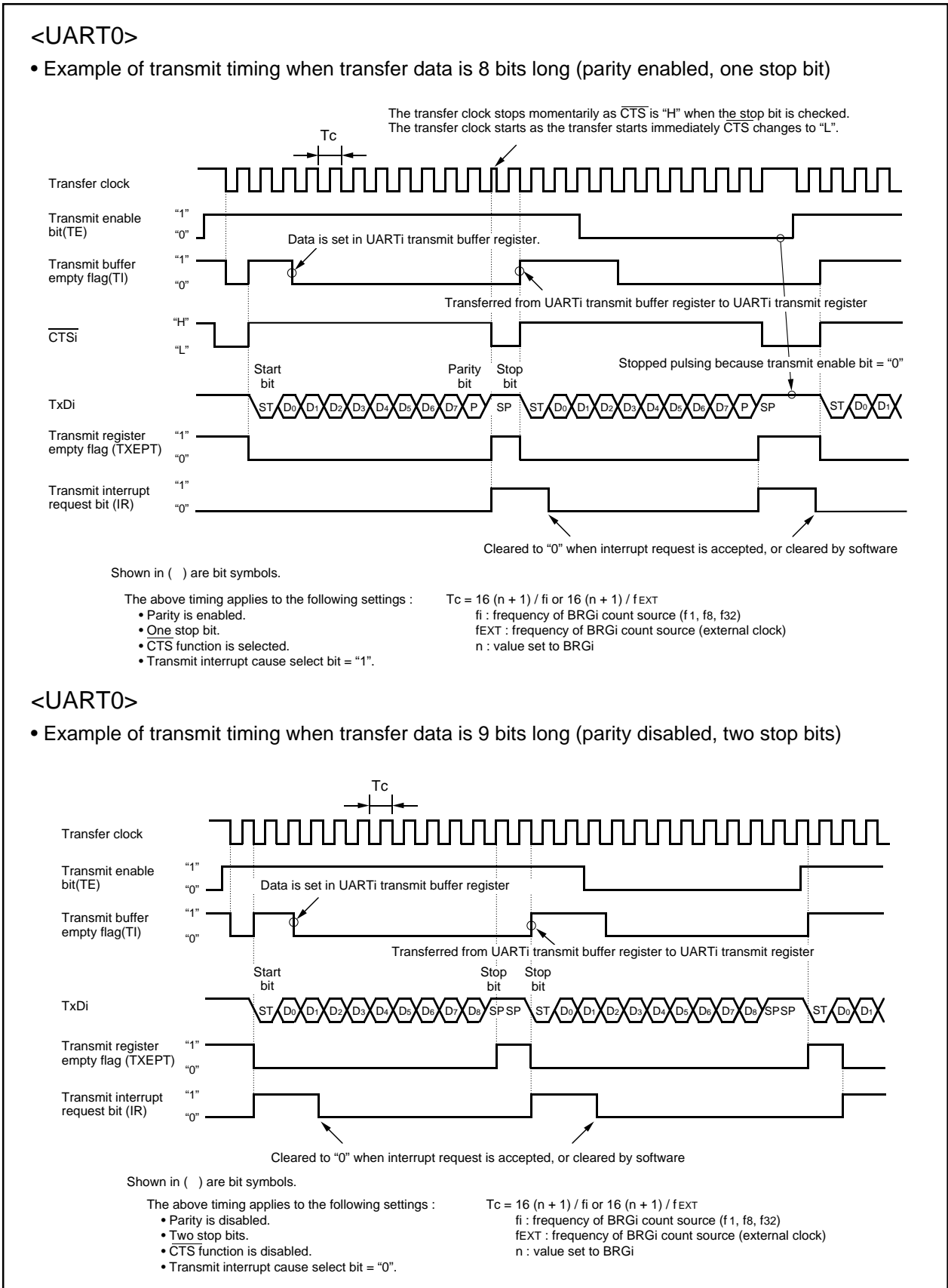


Figure 2.11.22 Typical transmit/receive timings in UART mode (1)

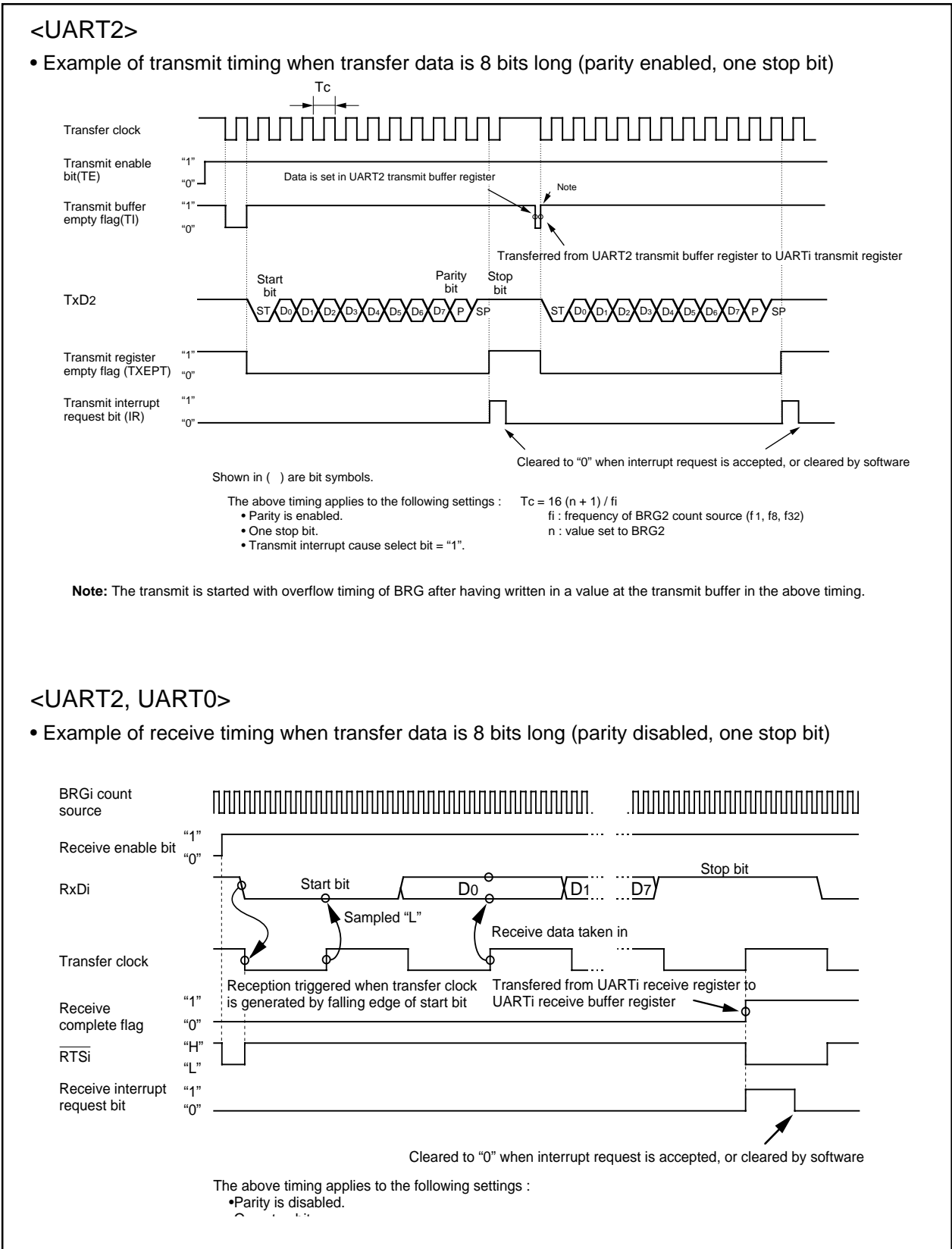
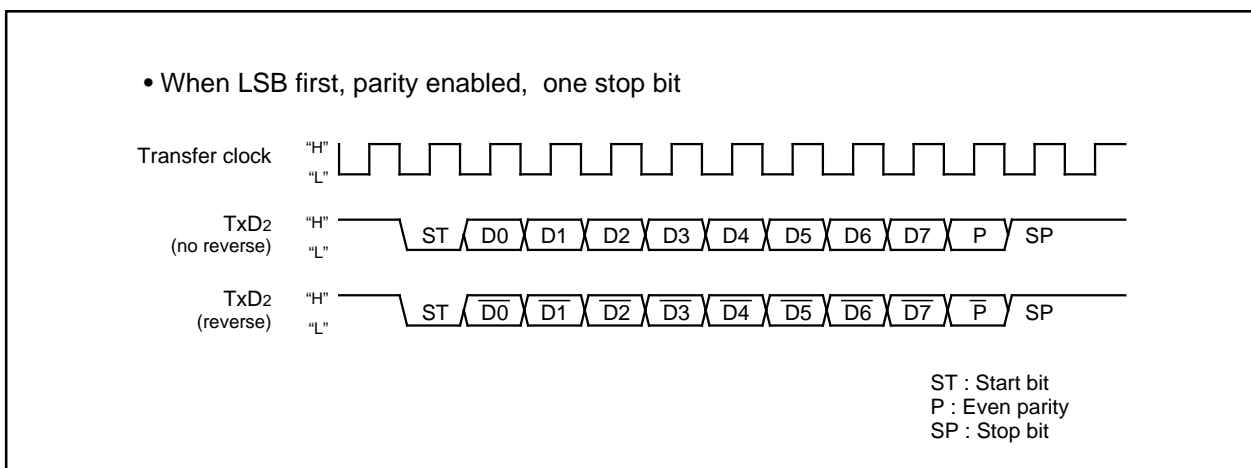


Figure 2.11.23 Typical transmit/receive timings in UART mode (2)

**(1) 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**

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

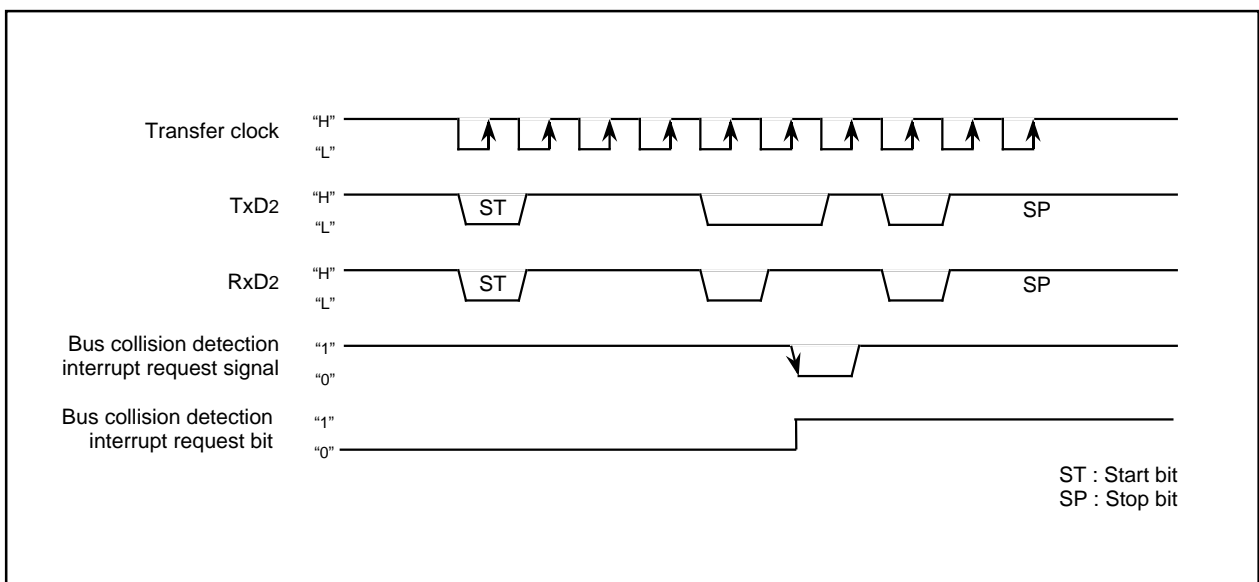
This function is to reverse TxD pin output and RxD pin input. The level of any data to be input or output (including the start bit, stop bit(s), and parity bit) is reversed. Set this function to “0” (not to reverse) for usual use.

**(3) Bus collision detection function and other functions (UART2)**

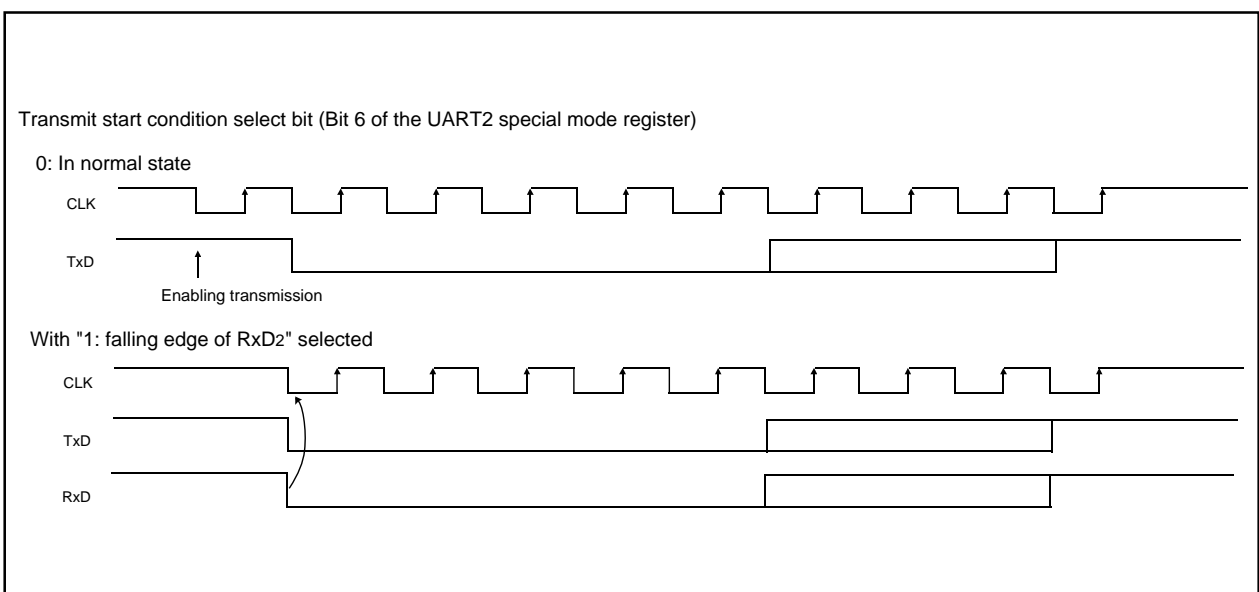
This function is to sample the output level of the TxD pin and the input level of the RxD pin at the rising edge of the transfer clock; if their values are different, then an interrupt request occurs. Figure 2.11.25 shows the example of detection timing of a buss collision (in UART mode).

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 Serial Interface Ports

The I/O ports (P67, P70 to P72, P93, P94) function as I/O ports of UART2 and multi-master I<sup>2</sup>C-BUS interface 0 and 1 (refer to “2.11.6 Multi-master I<sup>2</sup>C-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), bits 0 and 2 (PSEL0 and FIICON) of the I<sup>2</sup>C0 port selection register (address 02E516), and bits 2 (FIICON) of the I<sup>2</sup>C1 port selection register (address 02ED16).

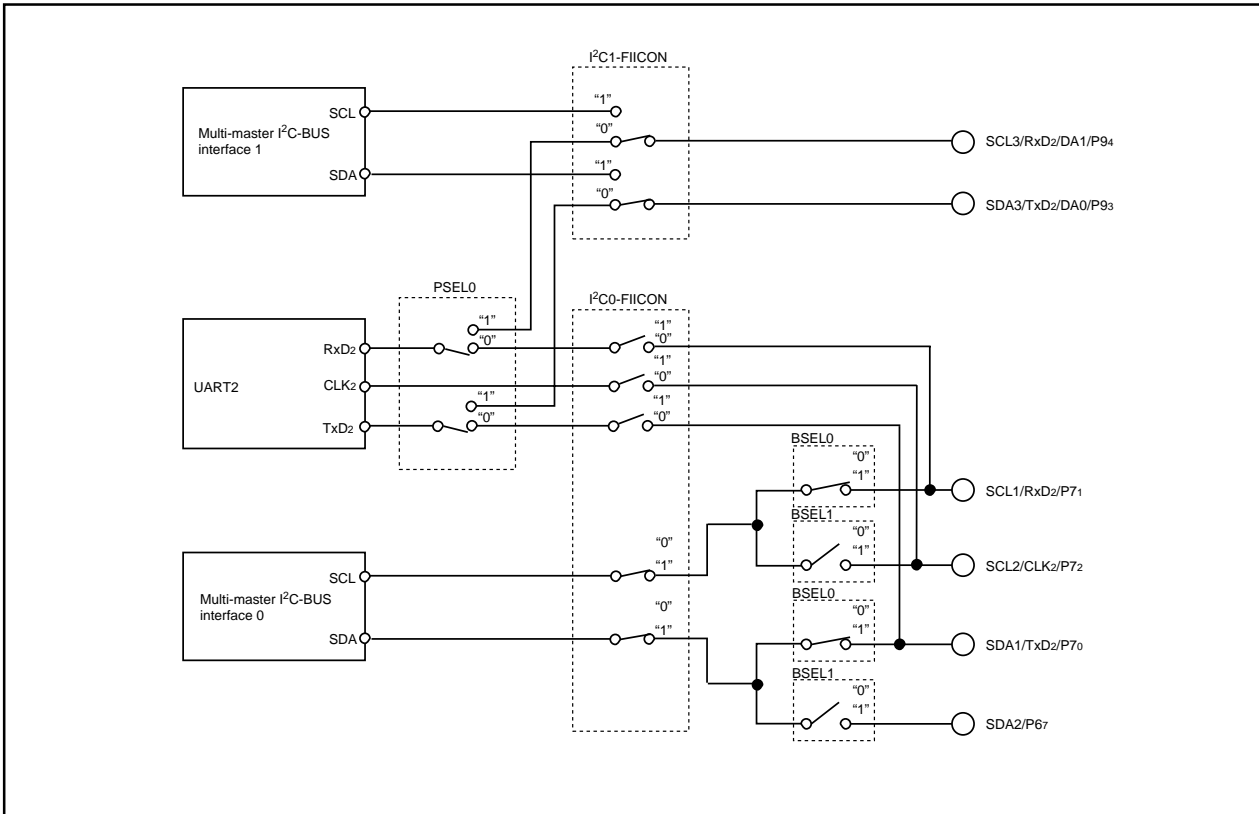


Figure 2.11.27 Serial interface port control

### 2.11.5 Multi-master I<sup>2</sup>C-BUS Interface 0 and Multi-master I<sup>2</sup>C-BUS Interface 1

The multi-master I<sup>2</sup>C-BUS interface 0 and 1 have each dedicated circuit and operate independently.

The multi-master I<sup>2</sup>C-BUS interface *i* is a serial communications circuit, conforming to the Philips I<sup>2</sup>C-BUS data transfer format. This interface *i*, offering both arbitration lost detection and a synchronous functions, is useful for the multi-master serial communications.

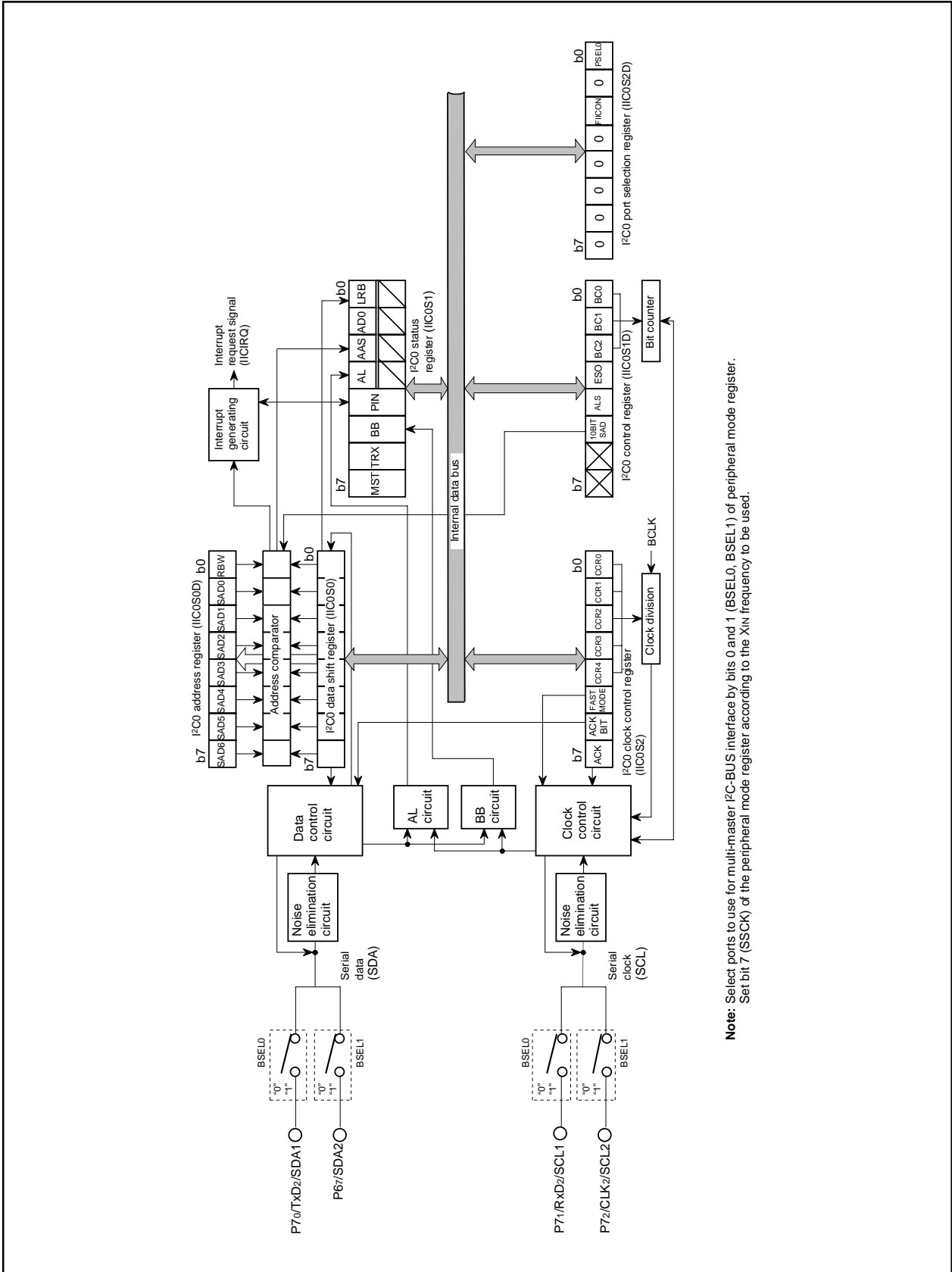
Figures 2.11.28 and 2.11.29 show a block diagram of the multi-master I<sup>2</sup>C-BUS interface *i* and Table 2.11.8 shows multi-master I<sup>2</sup>C-BUS interface *i* functions.

This multi-master I<sup>2</sup>C-BUS interface *i* consists of the I<sup>2</sup>C<sub>i</sub> address register, the I<sup>2</sup>C<sub>i</sub> data shift register, the I<sup>2</sup>C<sub>i</sub> clock control register, the I<sup>2</sup>C<sub>i</sub> control register, the I<sup>2</sup>C<sub>i</sub> status register, the I<sup>2</sup>C<sub>i</sub> port selection register and other control circuits.

**Table 2.11.8 Multi-master I<sup>2</sup>C-BUS Interface Functions**

Item	Function
Format	In conformity with Philips I <sup>2</sup> 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 <sup>2</sup> C-BUS standard: Master transmission Master reception Slave transmission Slave reception
SCL clock frequencyn	16.1 kHz to 400 kHz (at BCLK = 16 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 I<sup>2</sup>C control register at address 027D16) for connections between the I<sup>2</sup>C-BUS interface 0 and ports (SCL1, SCL2, SDA1, SDA2).



**Note:** Select ports to use for multi-master I2C-BUS interface by bits 0 and 1 (BSEL0, BSEL1) of peripheral mode register. Set bit 7 (SSCK) of the peripheral mode register according to the XIN frequency to be used.

Fig. 2.11.28 Block Diagram of Multi-master I2C-BUS Interface 0

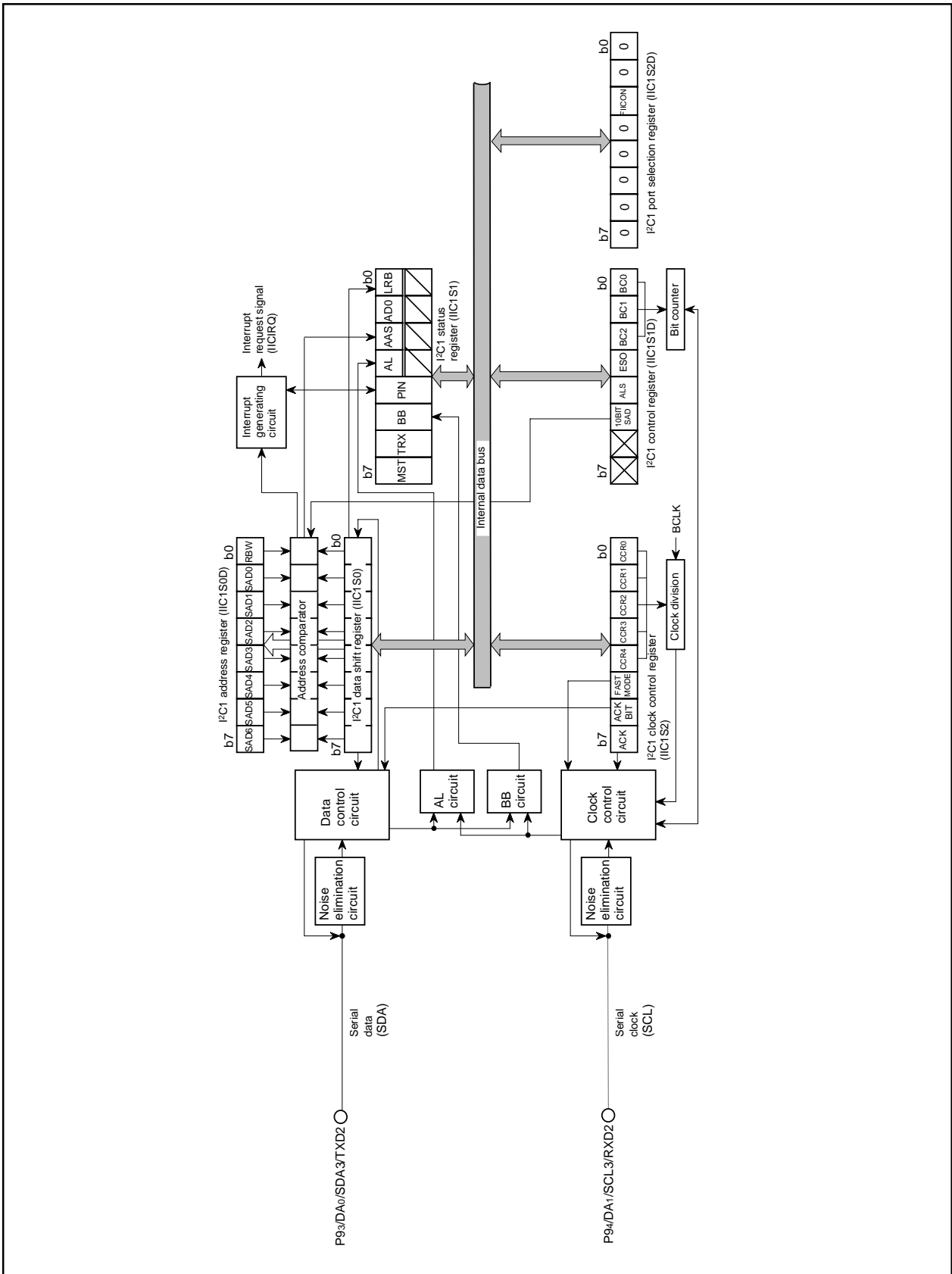


Fig. 2.11.29 Block Diagram of Multi-master I<sup>2</sup>C-BUS Interface 1



**(1) I<sup>2</sup>C<sub>i</sub> port selection register (i = 0, 1)**

The I<sup>2</sup>C<sub>i</sub> port selection register consists of bit to validate the multi-master I<sup>2</sup>C-BUS interface i function.

■ **Bit 0: Tx<sub>D2</sub>, Rx<sub>D2</sub> port select bit**      **Note: I<sup>2</sup>C<sub>0</sub> Port Selection Register only**

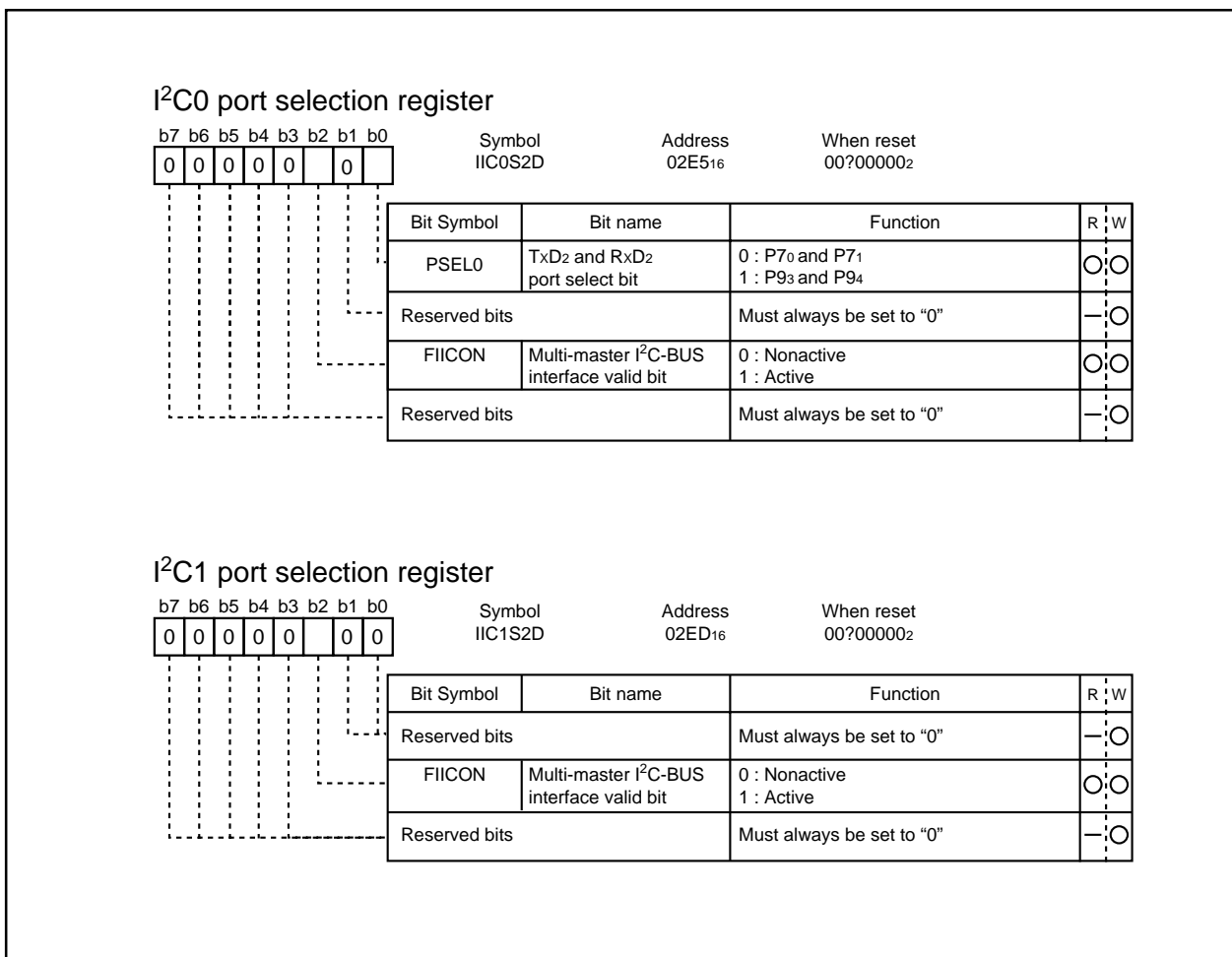
When using UART2, setting bit to “0” causes P7<sub>0</sub> and P7<sub>1</sub> to function respectively as Tx<sub>D2</sub> and Rx<sub>D2</sub>, and setting bit to “1” causes P9<sub>3</sub> and P9<sub>4</sub> to function respectively as Tx<sub>D2</sub> and Rx<sub>D2</sub>.

Note that the multimaster I<sup>2</sup>C-BUS interface enable bit (FIICON) described later has priority over this bit.

■ **Bit 2: Multi-master I<sup>2</sup>C-BUS interface valid bit (FIICON)**

When this bit is “0”, the multi-master I<sup>2</sup>C-BUS interface i is nonactive; when “1”, it is active. When selecting active, multi-master I<sup>2</sup>C-BUS interface 0 is connected with the ports selected by bits 0 and 1 of the peripheral mode register (address 027D16) and multi-master I<sup>2</sup>C-BUS interface 1 is connected with the ports P9<sub>3</sub> and P9<sub>4</sub>.

**Note:** It needs 20-BCLK cycles from setting this bit to “1” to being active of multi-master I<sup>2</sup>C-BUS interface i. Accordingly, do not access multi-master I<sup>2</sup>C-BUS interface i-related registers in this period.



**Fig. 2.11.30 I<sup>2</sup>C<sub>i</sub> port selection register (i = 0, 1)**

**(2) I<sup>2</sup>Ci data shift register, I<sup>2</sup>Ci transmit buffer register (i = 0, 1)**

The I<sup>2</sup>Ci 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 I<sup>2</sup>Ci data shift register is in a write enable status only when the ESO bit of the I<sup>2</sup>Ci control register is "1." The bit counter is reset by a write instruction to the I<sup>2</sup>Ci data shift register. When both the ESO bit and the MST bit of the I<sup>2</sup>Ci status register are "1," the SCL is output by a write instruction to the I<sup>2</sup>Ci data shift register. Reading data from the I<sup>2</sup>Ci data shift register is always enabled regardless of the ESO bit value.

The I<sup>2</sup>Ci transmit buffer register is a register to store transmit data (slave address) to the I<sup>2</sup>Ci data shift register before RESTART condition generation. That is, in master, transmit data written to the I<sup>2</sup>Ci transmit buffer register is written to the I<sup>2</sup>Ci data shift register simultaneously. However, the SCL is not output. The I<sup>2</sup>Ci transmit buffer register can be written only when the ESO bit is "1," reading data from the I<sup>2</sup>Ci transmit buffer register is disabled regardless of the ESO bit value.

**Notes 1:** To write data into the I<sup>2</sup>Ci data shift register or the I<sup>2</sup>Ci 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 I<sup>2</sup>Ci data shift register or the I<sup>2</sup>Ci transmit buffer register is written, keep an interval of 4 BCLK or more.

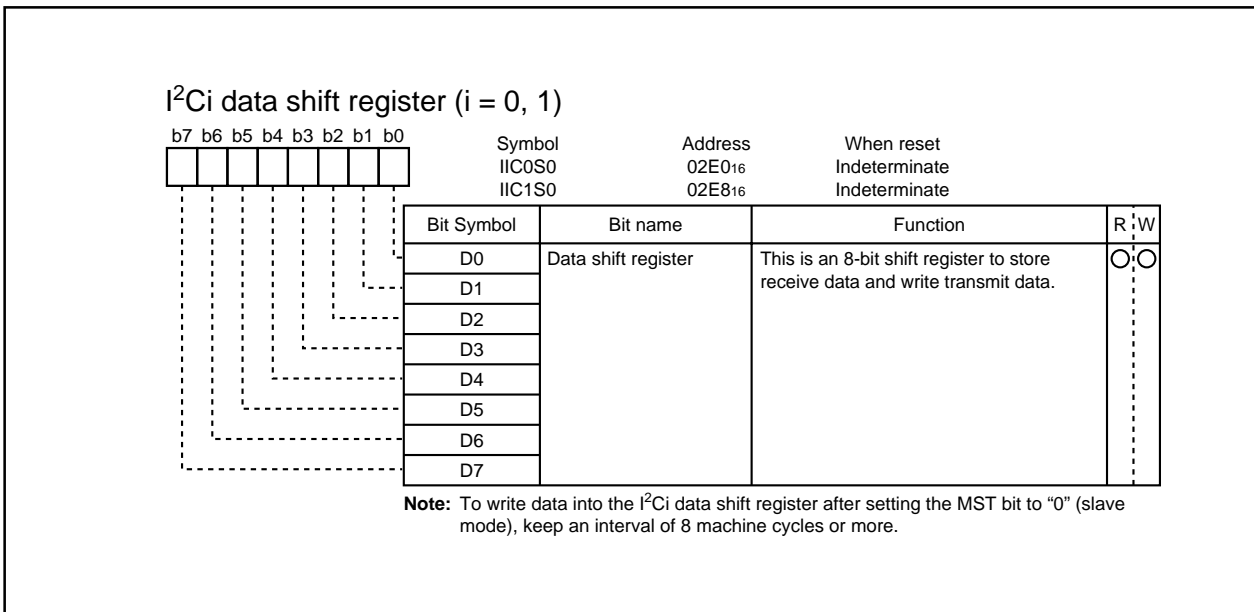


Fig. 2.11.31 I<sup>2</sup>Ci data shift register (i = 0, 1)

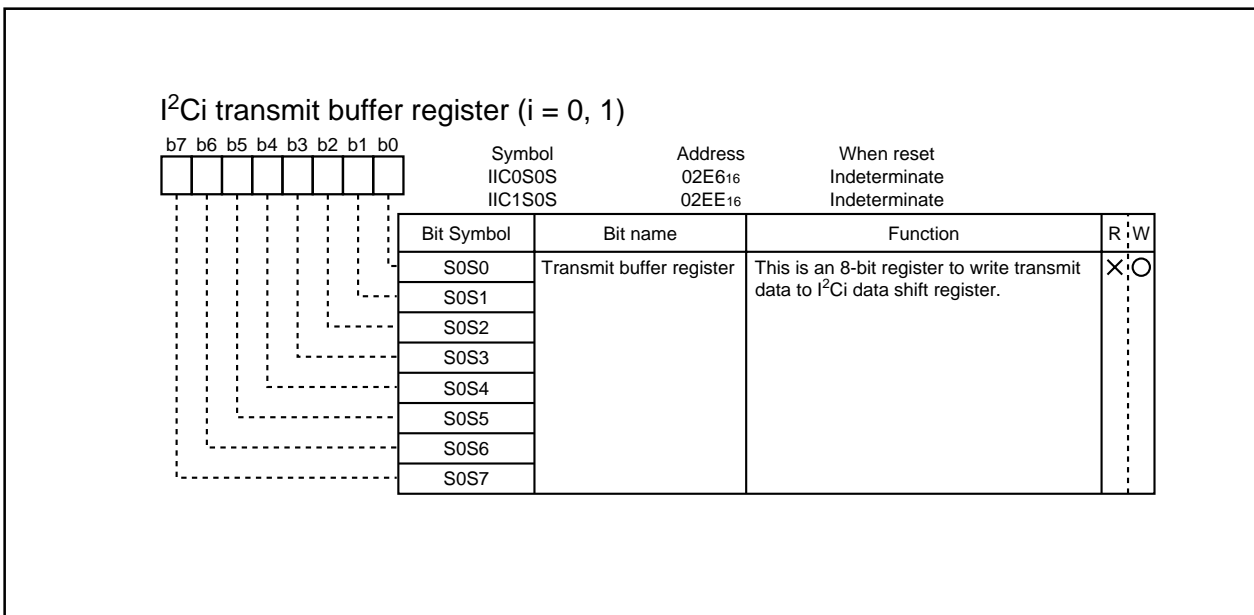


Fig. 2.11.32 I<sup>2</sup>Ci transmit buffer register (i = 0, 1)

**(3) I<sup>2</sup>Ci address register (i = 0, 1)**

The I<sup>2</sup>Ci 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.

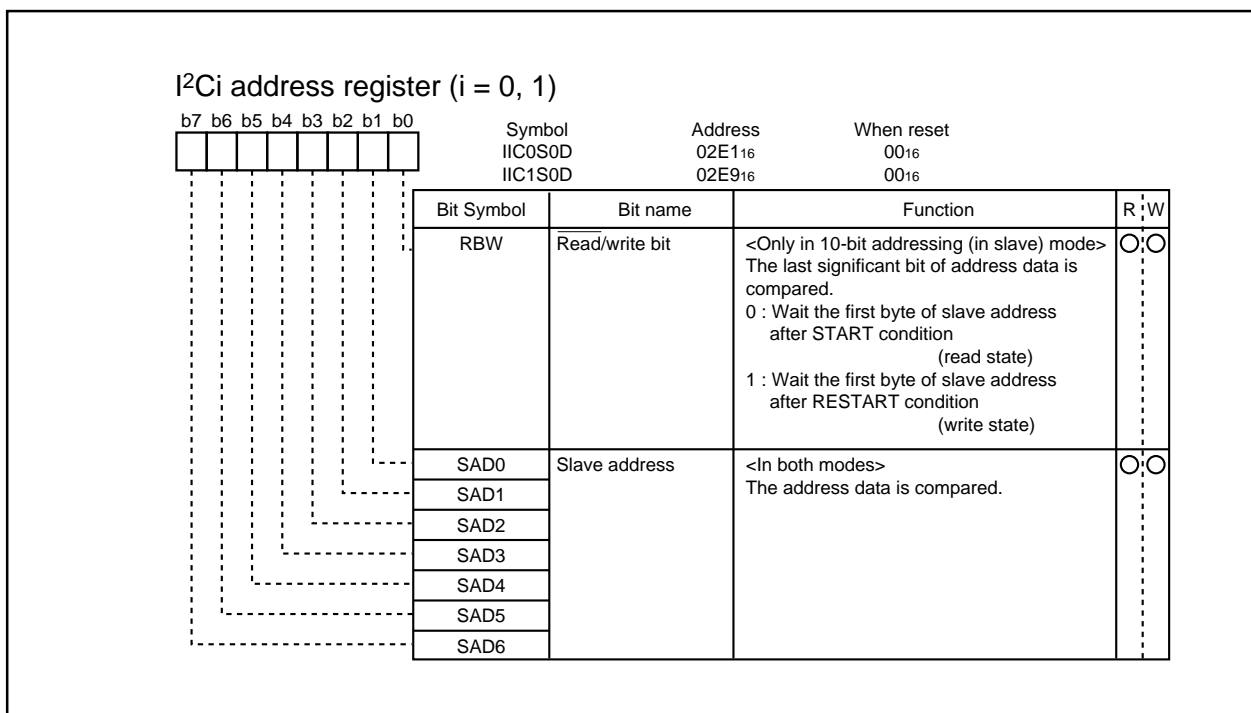
**■ Bit 0:  $\overline{\text{read/write}}$  bit (RBW)**

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 I<sup>2</sup>Ci 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. 2.11.33 I<sup>2</sup>Ci address register (i = 0, 1)**

**(4) I<sup>2</sup>Ci clock control register (i = 0, 1)**

The I<sup>2</sup>Ci 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 I<sup>2</sup>Ci clock control register during transmission. If data is written during transmission, the I<sup>2</sup>Ci clock generator is reset, so that data cannot be transmitted normally.

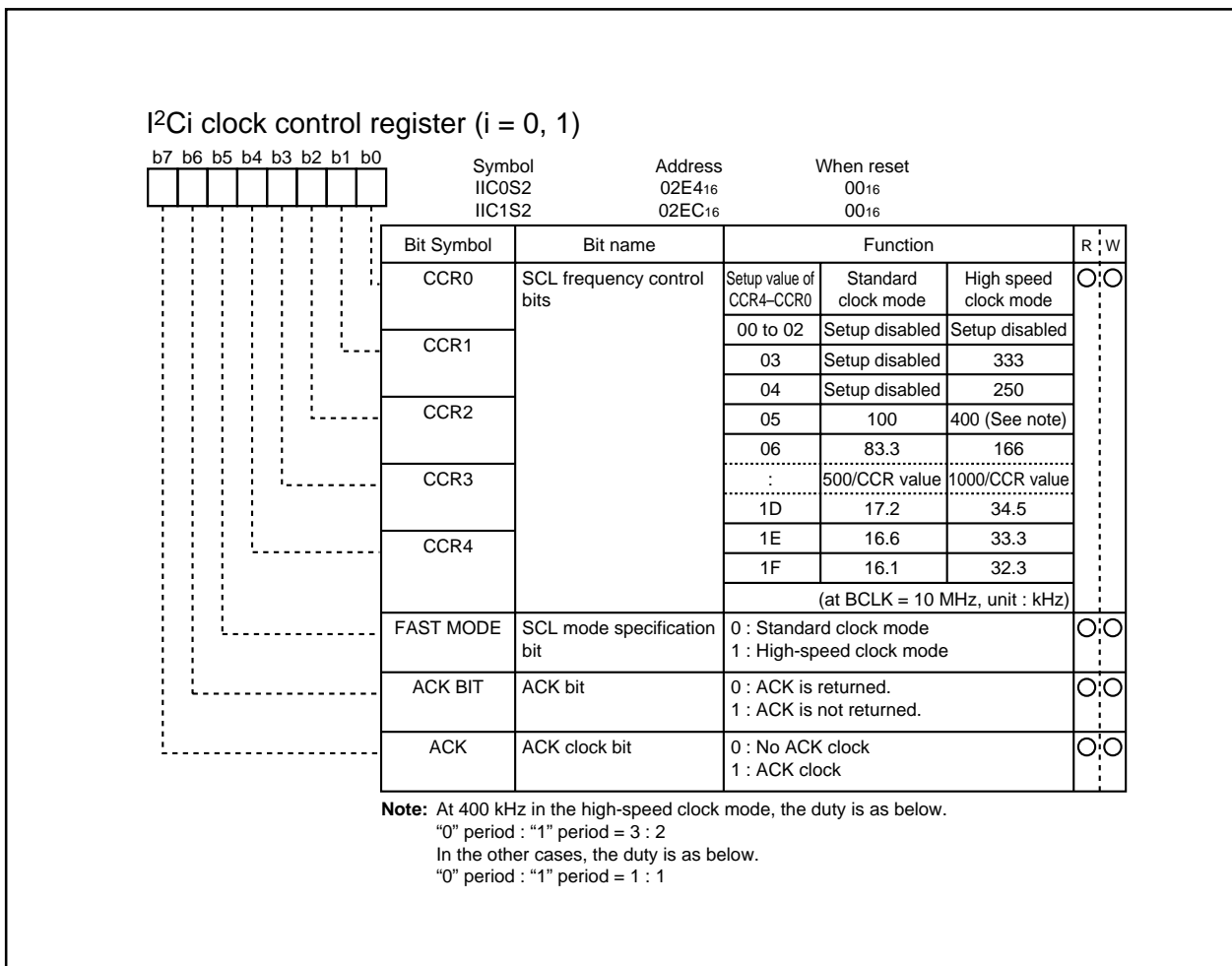


Fig. 2.11.34 I<sup>2</sup>Ci clock control register (i = 0, 1)

**(5) I<sup>2</sup>Ci control register (i = 0, 1)**

The I<sup>2</sup>Ci 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: I<sup>2</sup>C-BUS interface i use enable bit (ESO)**

This bit enables usage of the multimaster I<sup>2</sup>C-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 I<sup>2</sup>Ci status register).
- Writing data to the I<sup>2</sup>Ci data shift register and the I<sup>2</sup>Ci 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) I<sup>2</sup>Ci 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 I<sup>2</sup>Ci 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 I<sup>2</sup>Ci address register are compared with address data.

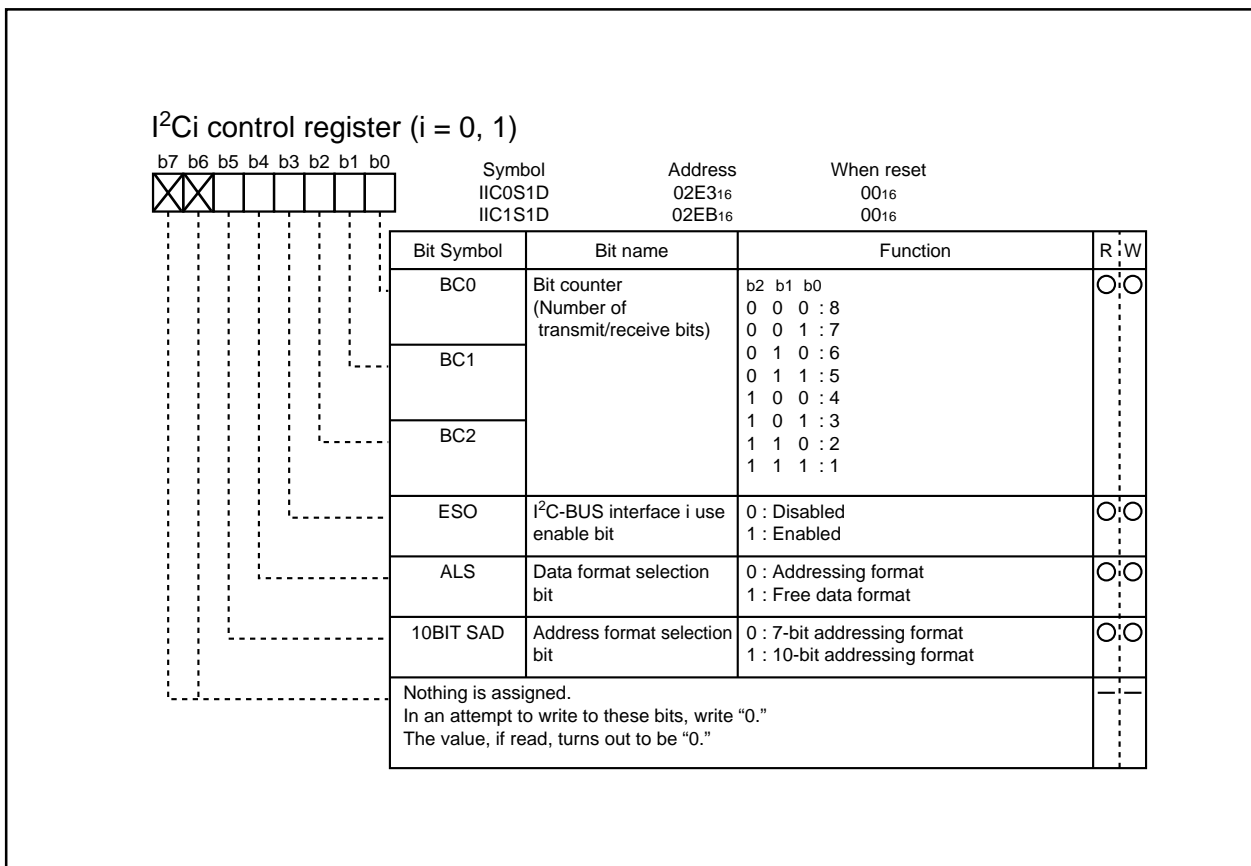


Fig. 2.11.35 I<sup>2</sup>C<sub>i</sub> control register (i = 0, 1)



**(6) I<sup>2</sup>Ci status register (i = 0, 1)**

The I<sup>2</sup>Ci status register controls the I<sup>2</sup>C-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 I<sup>2</sup>Ci data shift register or the I<sup>2</sup>Ci 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 I<sup>2</sup>Ci 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 I<sup>2</sup>Ci 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 I<sup>2</sup>Ci data shift register or the I<sup>2</sup>Ci 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: I<sup>2</sup>C-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 I<sup>2</sup>C-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 2.11.37 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 I<sup>2</sup>Ci data shift register or the I<sup>2</sup>Ci transmit buffer register (See note).
- When the ESO bit is “0”
- At reset

**Note:** It takes 12 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 I<sup>2</sup>Ci 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 I<sup>2</sup>Ci 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>.

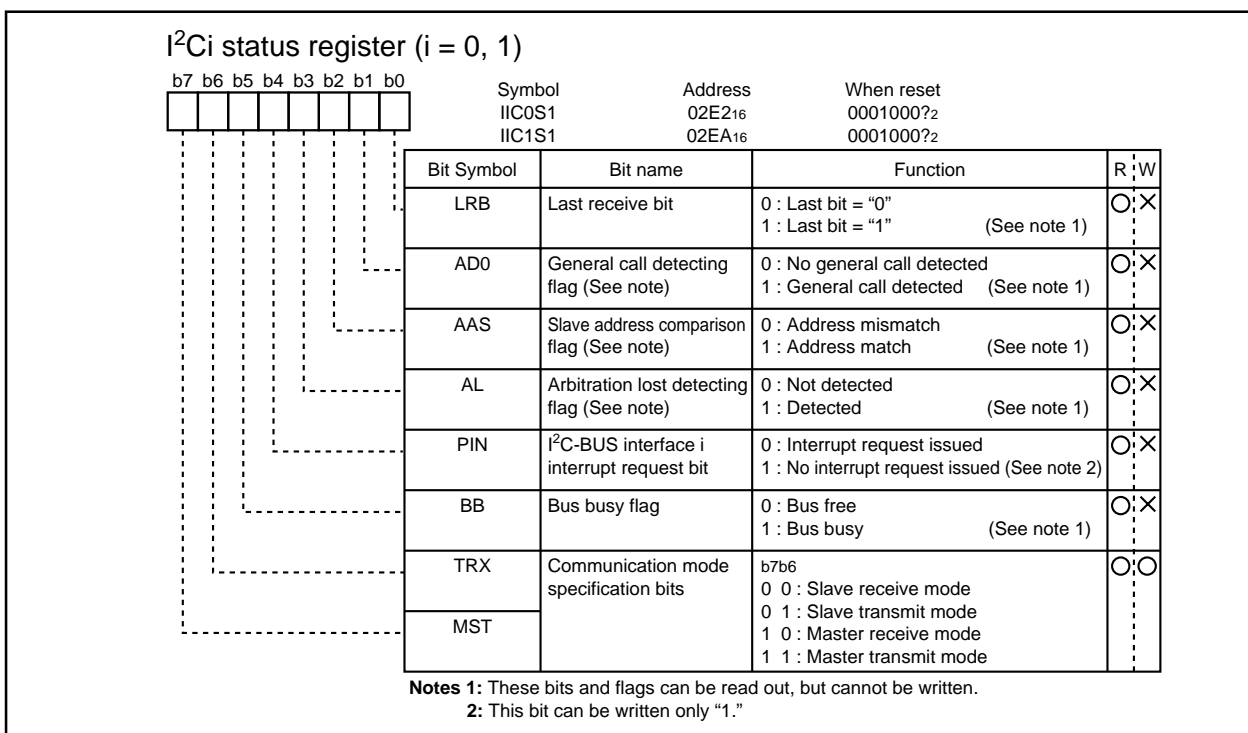


Fig. 2.11.36 I<sup>2</sup>Ci status register (i = 0, 1)

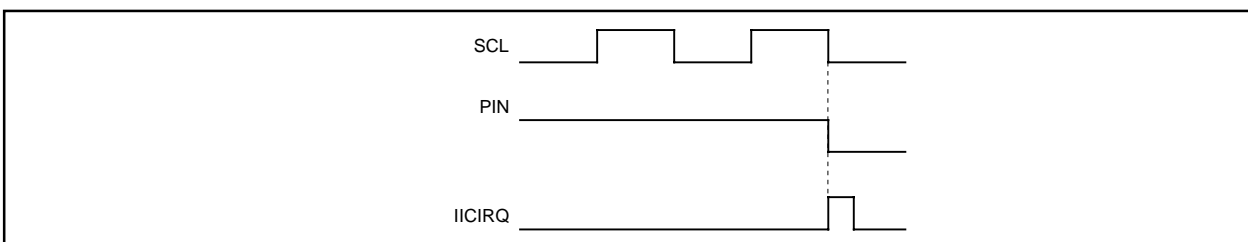
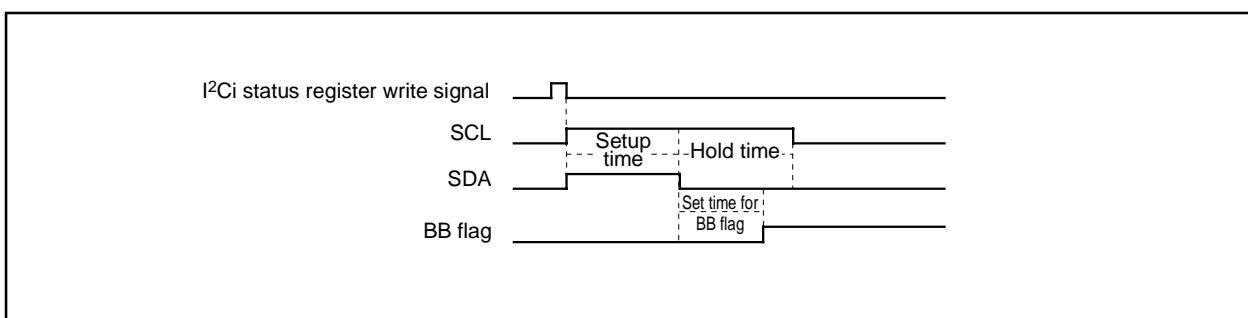


Fig. 2.11.37 Interrupt request signal generation timing

**(7) START condition generation method**

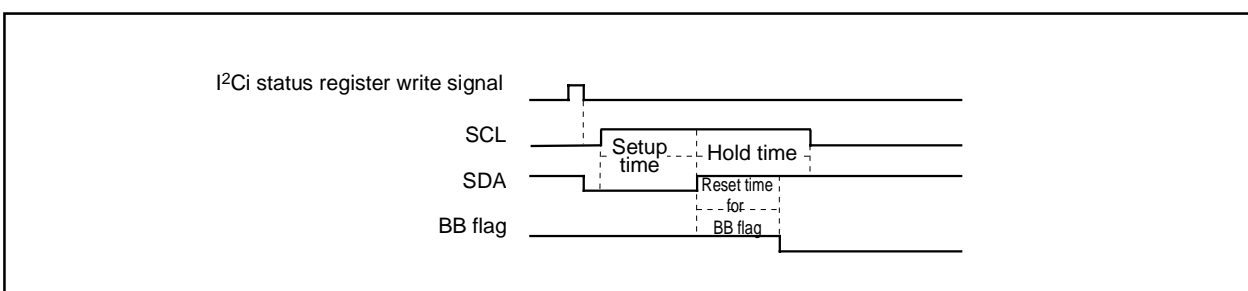
When the ESO bit of the I<sup>2</sup>Ci control register is “1,” execute a write instruction to the I<sup>2</sup>Ci status register to set the MST, TRX and BB bits to “1.” A START condition will then be generated. After that, the bit counter becomes “0002” and an SCL for 1 byte is output. The START condition generation timing and BB bit set timing are different in the standard clock mode and the high-speed clock mode. Refer to Figure 2.11.38 for the START condition generation timing diagram, and Table 2.11.9 for the START condition/STOP condition generation timing table.



**Fig. 2.11.38 START condition generation timing diagram**

**(8) STOP condition generation method**

When the ESO bit of the I<sup>2</sup>Ci control register is “1,” execute a write instruction to the I<sup>2</sup>Ci 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 2.11.39 for the STOP condition generation timing diagram, and Table 2.11.9 for the START condition/STOP condition generation timing table.



**Fig. 2.11.39 STOP condition generation timing diagram**

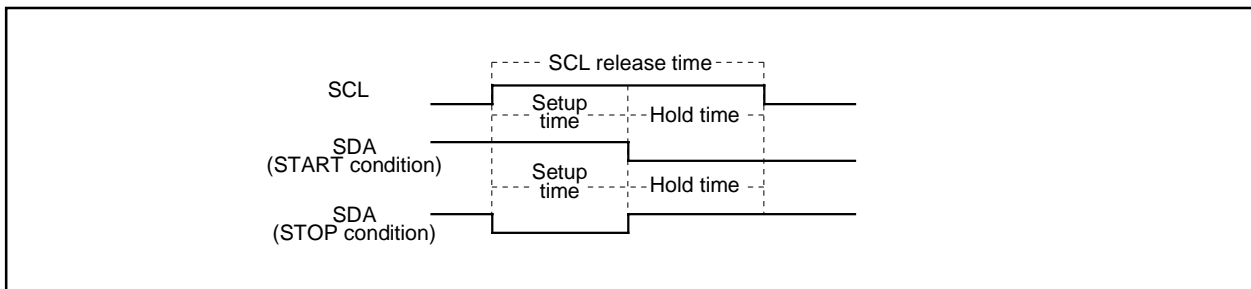
**Table 2.11.9 START condition/STOP condition generation timing table**

Item	Standard Clock Mode	High-speed Clock Mode
Setup time (Min.)	5.6 μs	2.1 μs
Hold time (Min.)	4.8 μs	2.3 μs
Set/reset time for BB flag	3.5 μs	0.75 μs

**(9) START/STOP condition detect conditions**

The START/STOP condition detect conditions are shown in Figure 2.11.40 and Table 2.11.10. Only when the 3 conditions of Table 2.11.10 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. 2.11.40 START condition/STOP condition detect timing diagram**

**Table 2.11.10 START condition/STOP condition detect conditions**

Standard Clock Mode	High-speed Clock Mode
6.5 μs < SCL release time	1.0 μs < SCL release time
3.25 μs < Setup time	0.5 μs < Setup time
3.25 μs < Hold time	0.5 μs < Hold time

**(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 I<sup>2</sup>Ci 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 I<sup>2</sup>Ci address register. At the time of this comparison, address comparison of the RBW bit of the I<sup>2</sup>Ci address register is not made. For the data transmission format when the 7-bit addressing format is selected, refer to Figure 2.11.41, (1) and (2).

**■ 10-bit addressing format**

To meet the 10-bit addressing format, set the 10BIT SAD bit of the I<sup>2</sup>Ci 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 I<sup>2</sup>Ci address register. At the time of this comparison, an address comparison between the RBW bit of the I<sup>2</sup>Ci 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 I<sup>2</sup>Ci status register is set to “1.” After the second-byte address data is stored into the I<sup>2</sup>Ci 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 I<sup>2</sup>Ci 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 I<sup>2</sup>Ci address register. For the data transmission format when the 10-bit addressing format is selected, refer to Figure 2.11.41, (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 I<sup>2</sup>Ci address register and “0” in the RBW bit.
- ② Set the ACK return mode and SCL = 100 kHz by setting “85<sub>16</sub>” in the I<sup>2</sup>Ci clock control register.
- ③ Set “10<sub>16</sub>” in the I<sup>2</sup>Ci status register and hold the SCL at the HIGH.
- ④ Set a communication enable status by setting “08<sub>16</sub>” in the I<sup>2</sup>Ci control register.
- ⑤ Set the address data of the destination of transmission in the high-order 7 bits of the I<sup>2</sup>Ci data shift register and set “0” in the least significant bit.
- ⑥ Set “F0<sub>16</sub>” in the I<sup>2</sup>Ci 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 I<sup>2</sup>Ci 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 “D0<sub>16</sub>” in the I<sup>2</sup>Ci 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 I<sup>2</sup>Ci address register and “0” in the RBW bit.
- ② Set the no ACK clock mode and SCL = 400 kHz by setting “25<sub>16</sub>” in the I<sup>2</sup>Ci clock control register.
- ③ Set “10<sub>16</sub>” in the I<sup>2</sup>Ci status register and hold the SCL at the HIGH.
- ④ Set a communication enable status by setting “08<sub>16</sub>” in the I<sup>2</sup>Ci 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 I<sup>2</sup>Ci 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 I<sup>2</sup>Ci status register is set to “1” and an interrupt request signal occurs.
  - In the cases other than the above:  
AD0 and AAS of the I<sup>2</sup>Ci status register are set to “0” and no interrupt request signal occurs.
- ⑦ Set dummy data in the I<sup>2</sup>Ci 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.

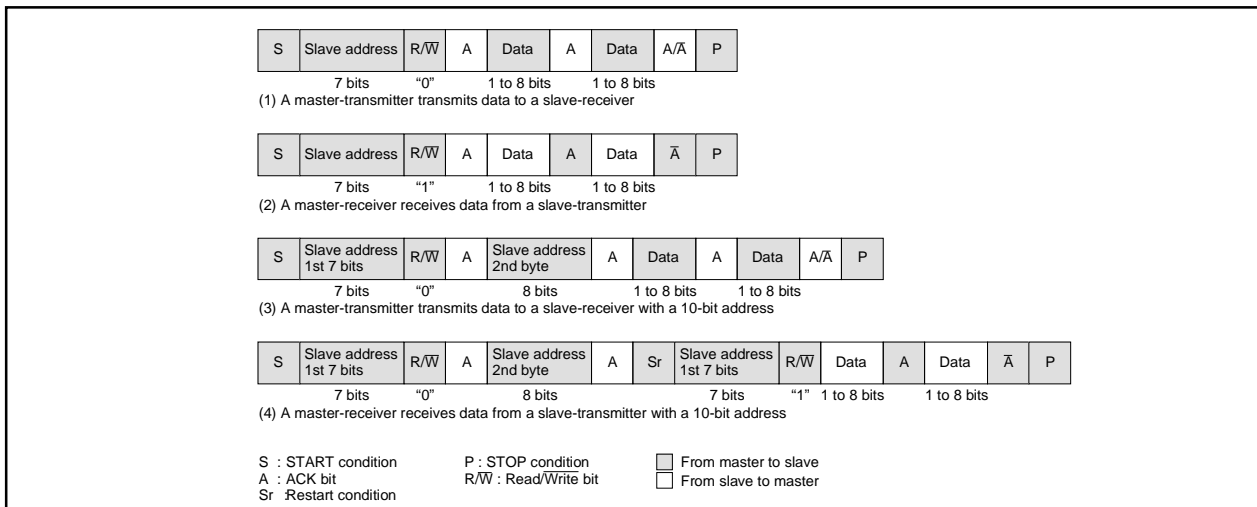


Fig. 2.11.41 Address data communication format

### (13) Precautions when using multi-master I<sup>2</sup>C-BUS interface i

#### ■ BCLK operation mode

Select the no-division mode and set the main clock frequency to  $f(X_{IN}) = 16 \text{ MHz}$  or  $10 \text{ MHz}$ . In this case, make sure the Peripheral Mode Register (address 027D16) bit 7 is set according to the frequency.

#### ■ Used instructions

Specify byte (.B) as data size to access multi-master I<sup>2</sup>C-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 I<sup>2</sup>C-BUS interface are described below.

##### • I<sup>2</sup>Ci data shift register (IICiS0)

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

##### • I<sup>2</sup>Ci 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.

##### • I<sup>2</sup>Ci status register (IICiS1)

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

##### • I<sup>2</sup>Ci 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.

##### • I<sup>2</sup>Ci clock control register (IICiS2)

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

##### • I<sup>2</sup>Ci port selection register (IICiS2D)

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

##### • I<sup>2</sup>Ci 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
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 4 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 I<sup>2</sup>Ci transmit buffer register to write the slave address value to the I<sup>2</sup>Ci data shift register. And also, be sure to add NOP instruction X 4.

### ■ Writing to I<sup>2</sup>Ci 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 I<sup>2</sup>Ci data shift register (IICiS0) and the I<sup>2</sup>Ci 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 VCCI
Operating clock $\phi_{AD}$ (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	<ul style="list-style-type: none"> <li>• Without sample and hold function: <math>\pm 5</math> LSB</li> <li>• With sample and hold function: <math>\pm 5</math> LSB</li> </ul>
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"> <li>• Software trigger</li> </ul> A-D conversion starts when the A-D conversion start flag changes to "1"
Conversion speed per pin	<ul style="list-style-type: none"> <li>• Without sample and hold function</li> <li style="padding-left: 20px;">49 <math>\phi_{AD}</math> cycles</li> <li>• With sample and hold function</li> <li style="padding-left: 20px;">28 <math>\phi_{AD}</math> cycles</li> </ul>

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

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

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

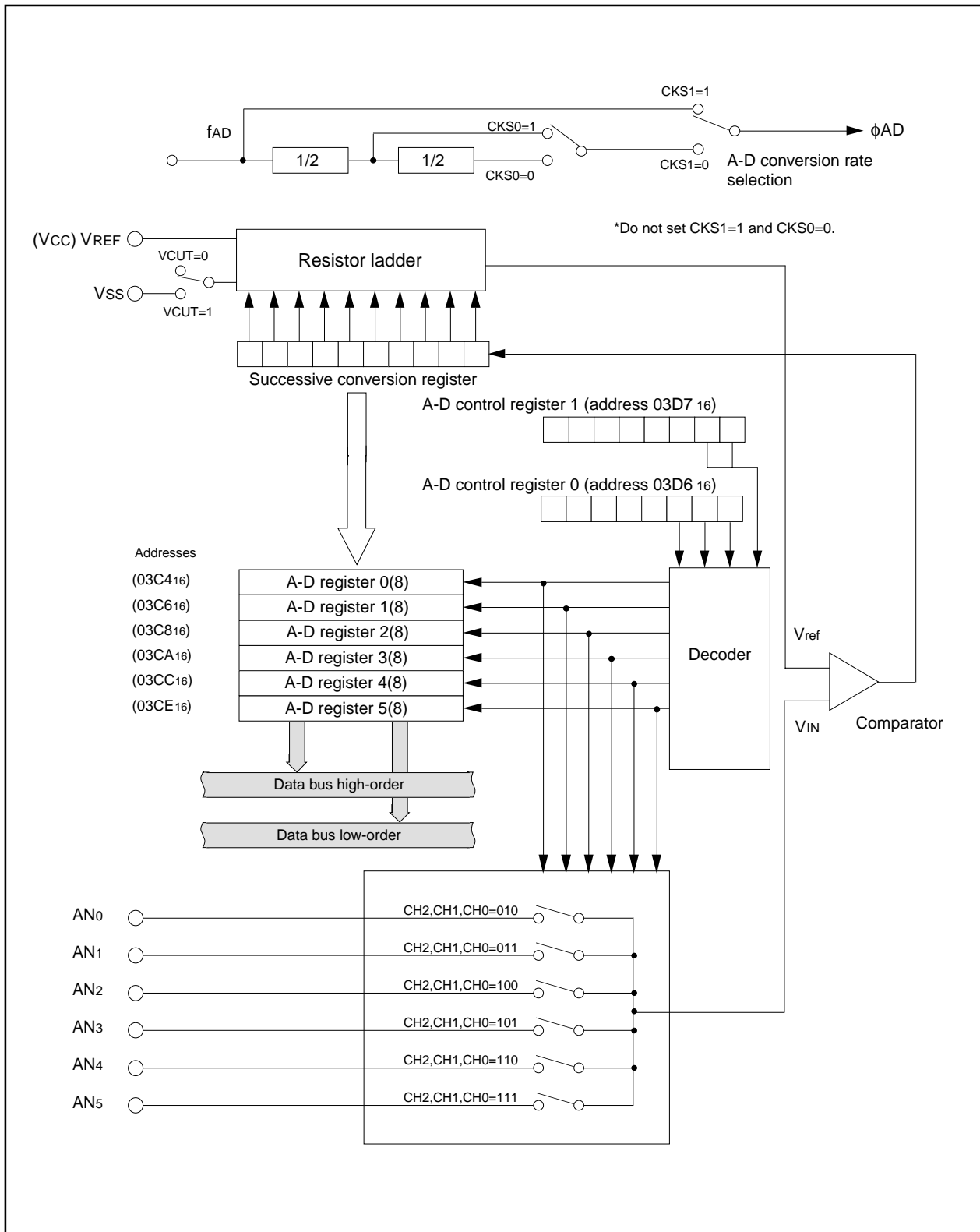


Figure 2.12.1 Block diagram of A-D converter

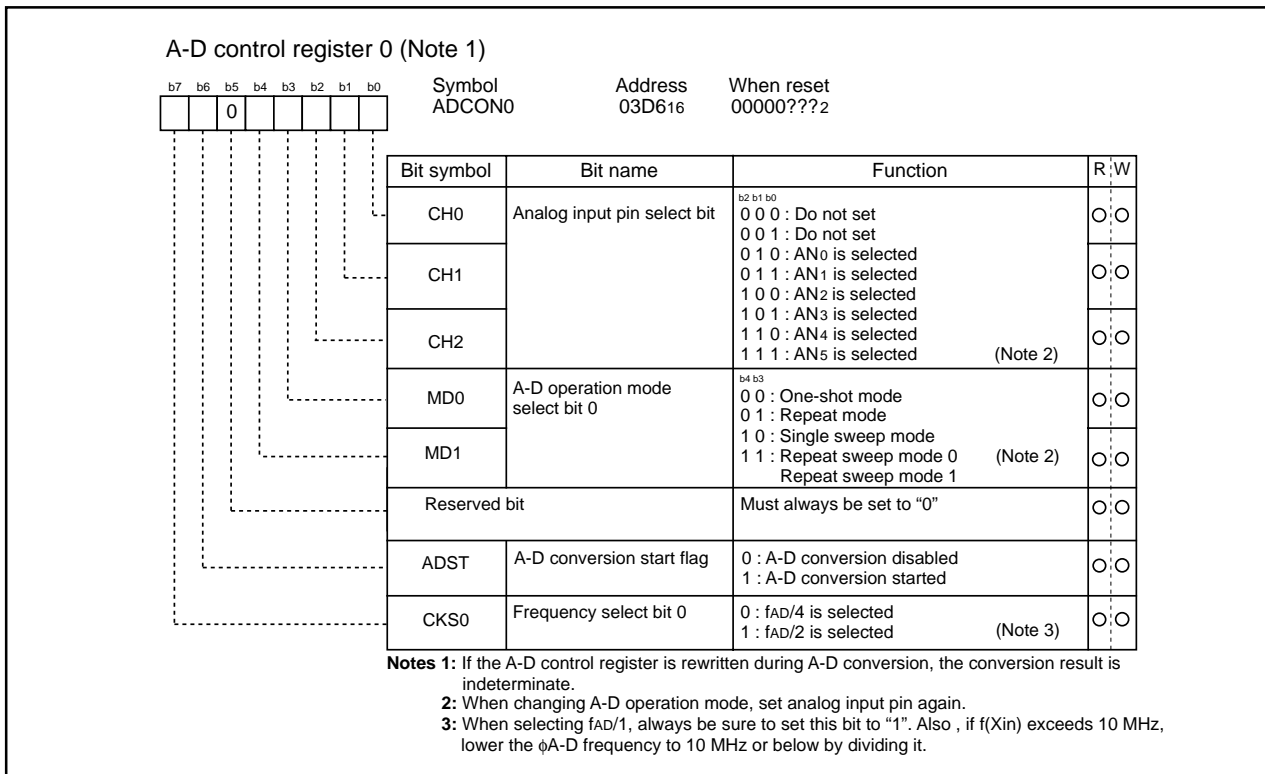


Figure 2.12.2 A-D control register 0

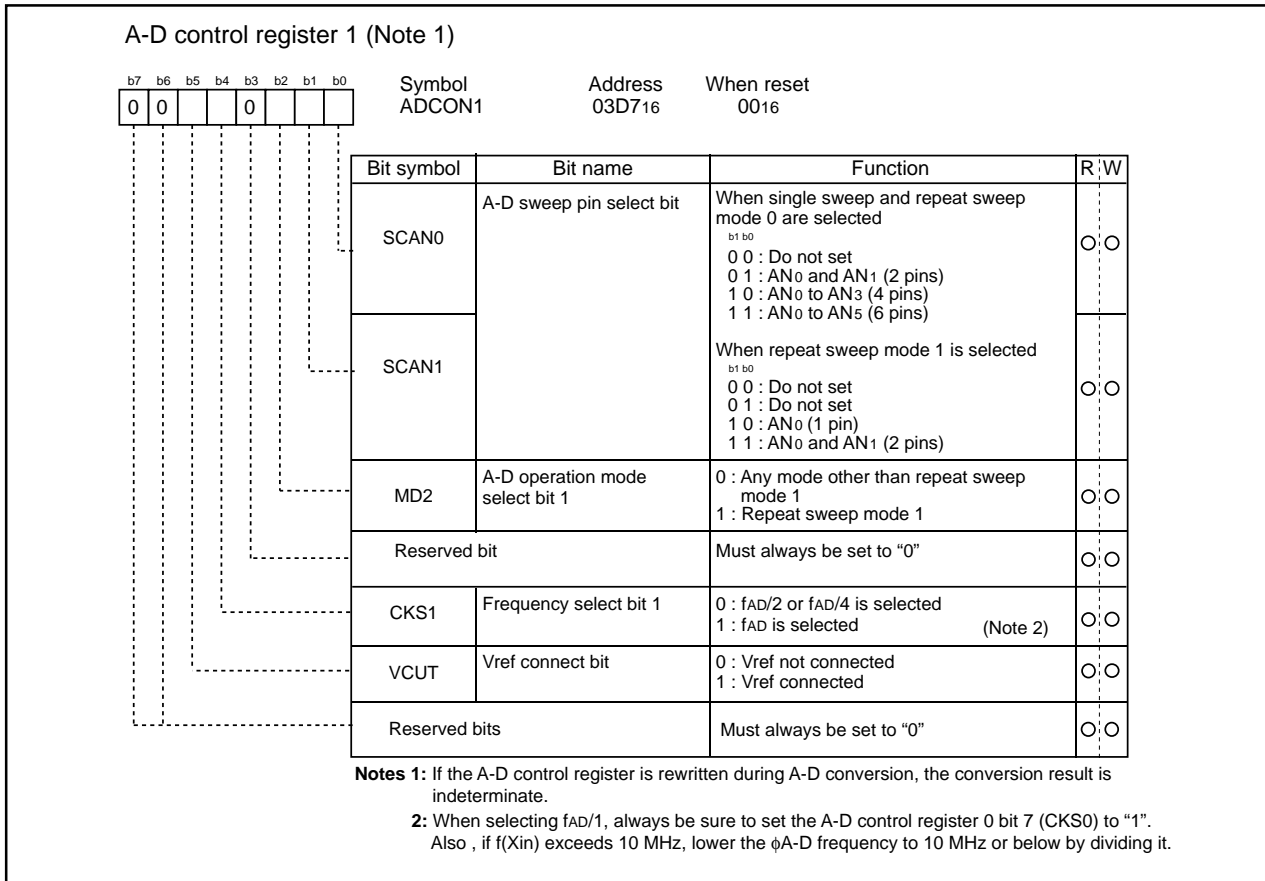


Figure 2.12.3 A-D control register 1

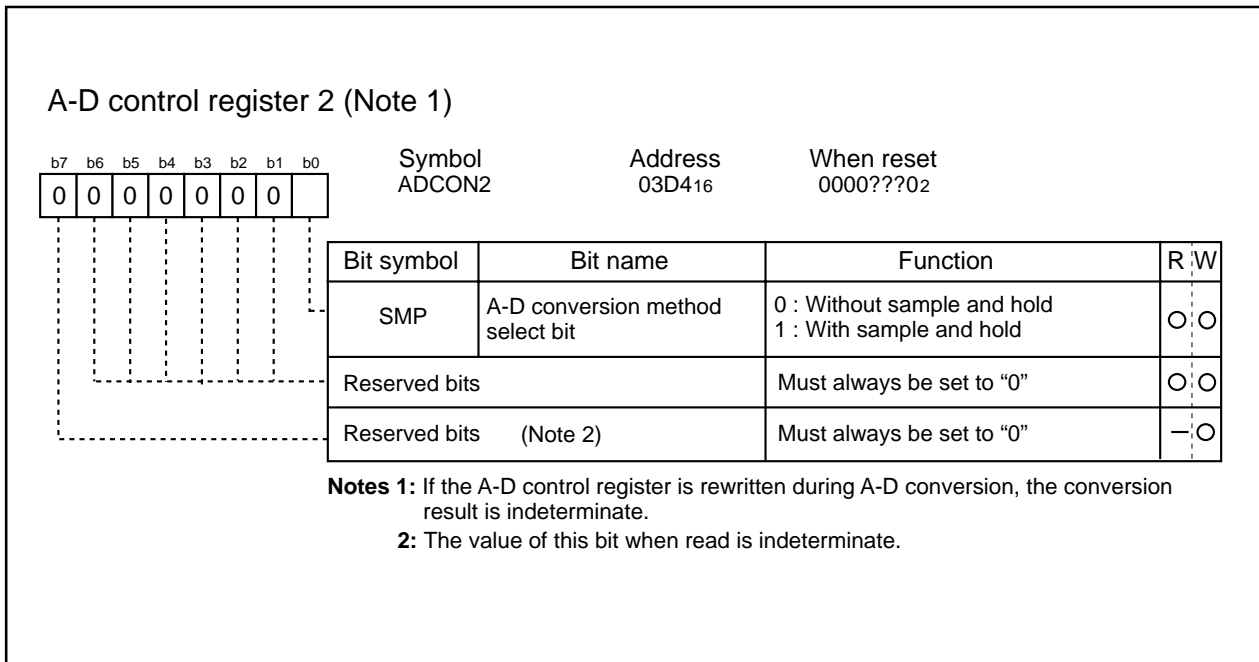


Figure 2.12.4 A-D control register 2

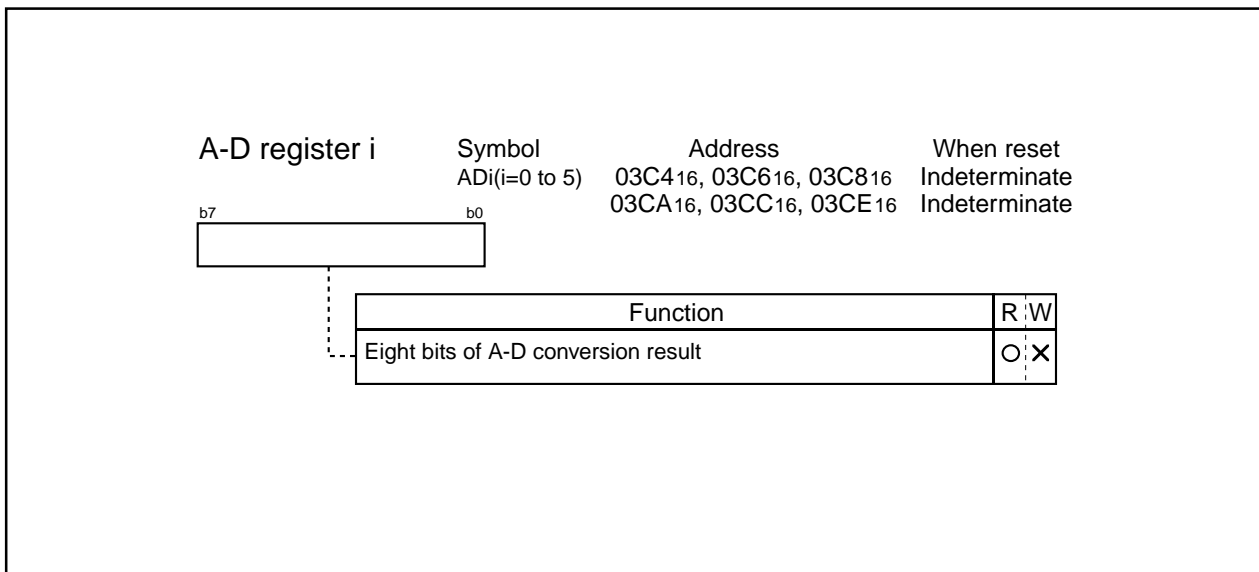


Figure 2.12.5 A-D register i (i = 0 to 5)

### 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"> <li>• End of A-D conversion</li> <li>• Writing "0" to A-D conversion start flag</li> </ul>
Interrupt request generation timing	End of A-D conversion
Input pin	One of AN0 to AN5, as selected
Reading of result of A-D converter	Read A-D register corresponding to selected pin

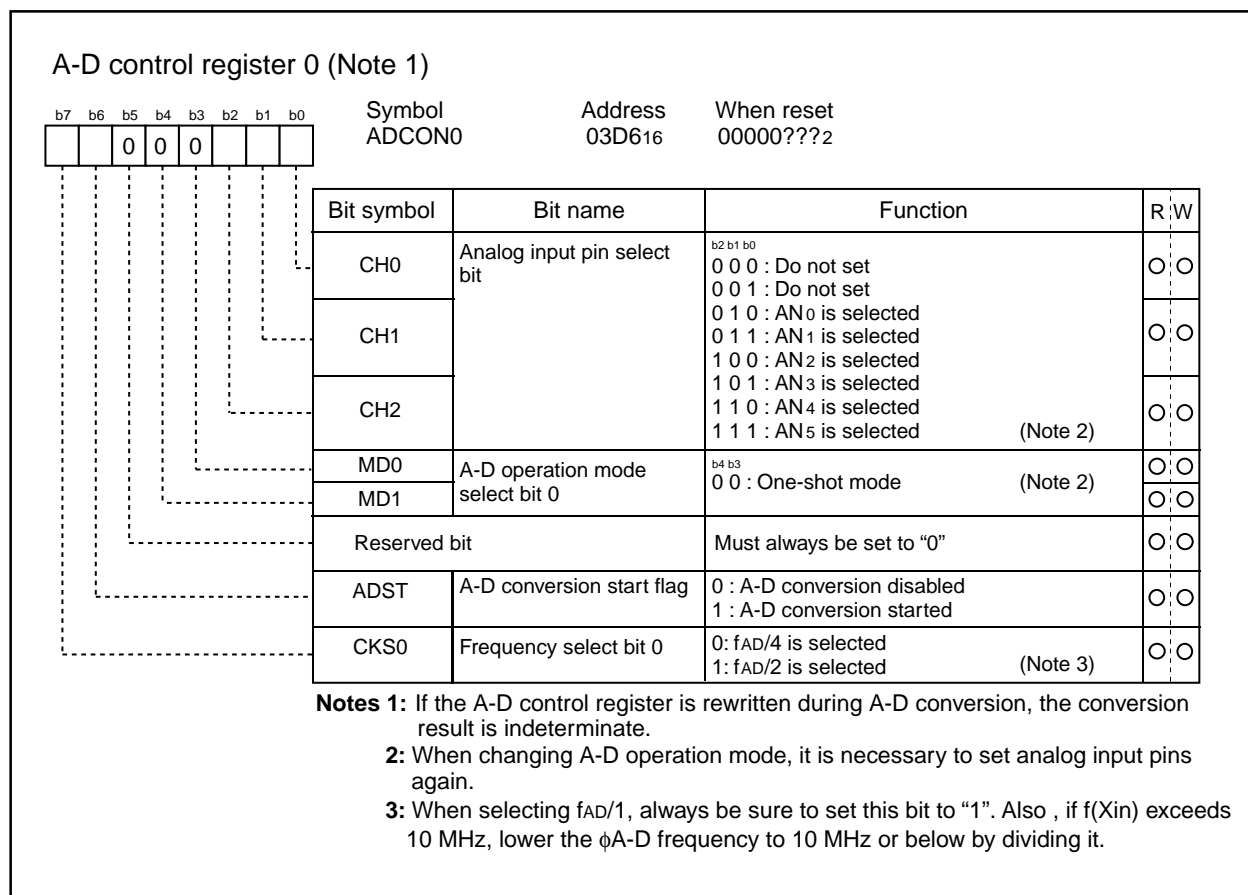


Figure 2.12.6 A-D control register 0 in one-shot mode

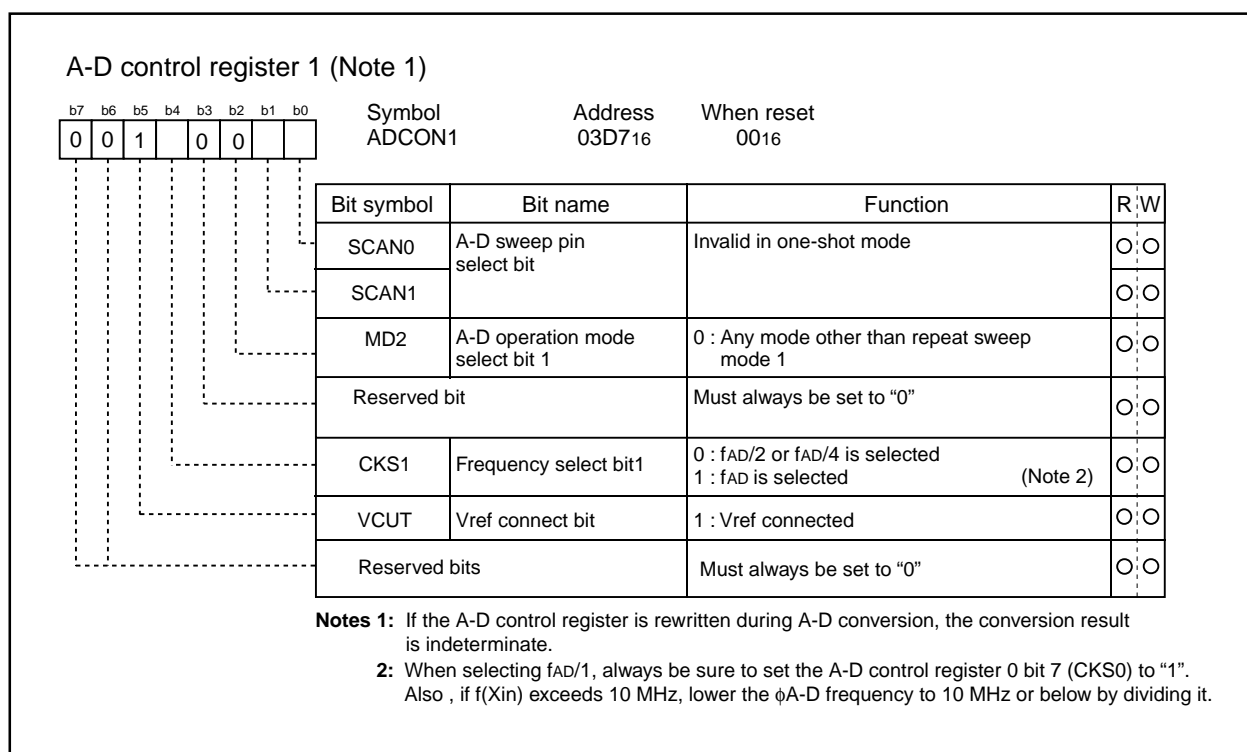


Figure 2.12.7 A-D control register 1 in one-shot mode

## 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



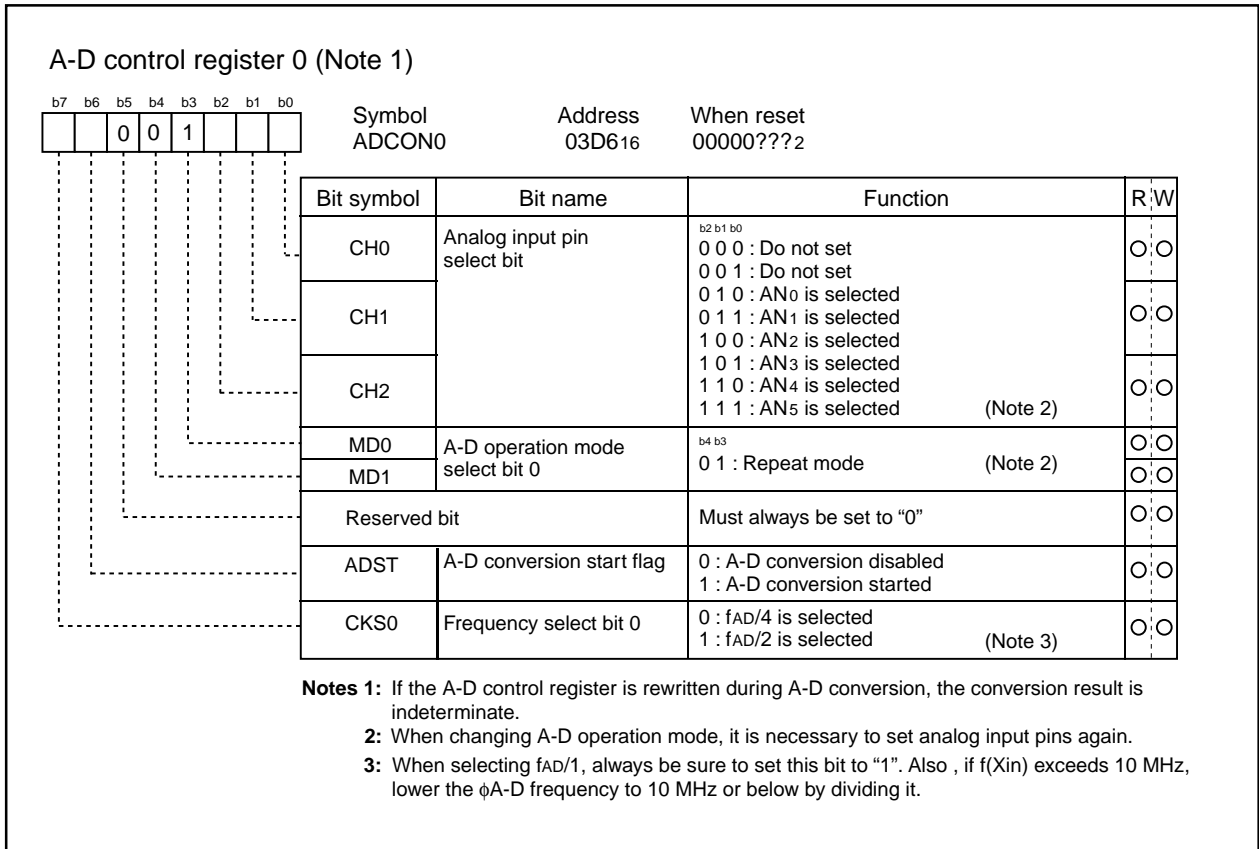


Figure 2.12.8 A-D conversion register 0 in repeat mode

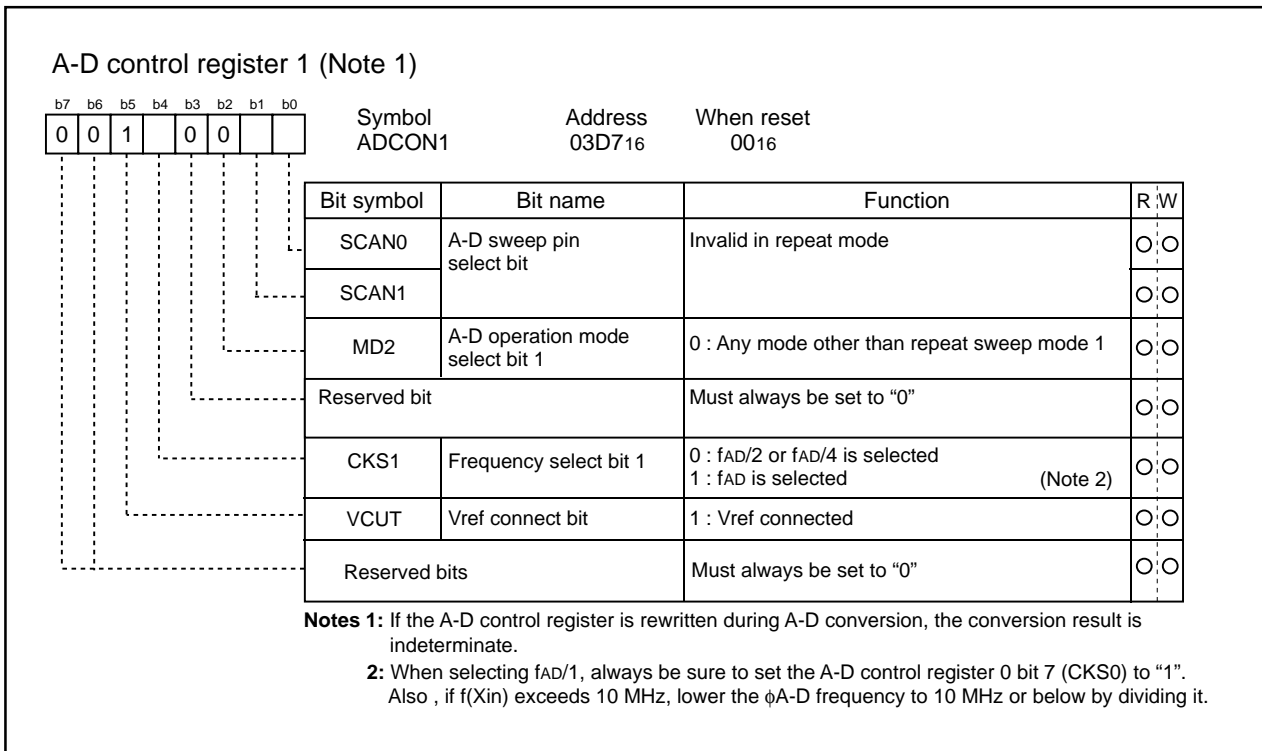


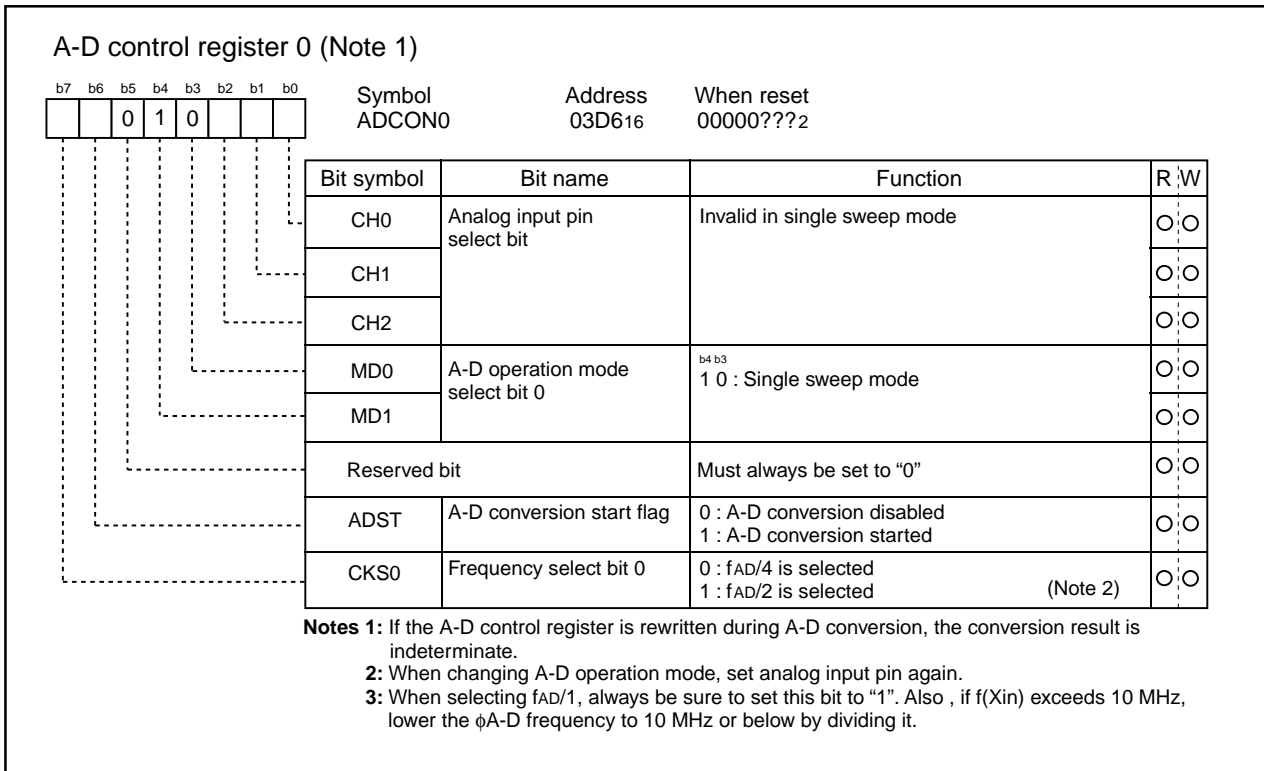
Figure 2.12.9 A-D conversion register 1 in repeat mode

### 2.12.3 Single Sweep Mode

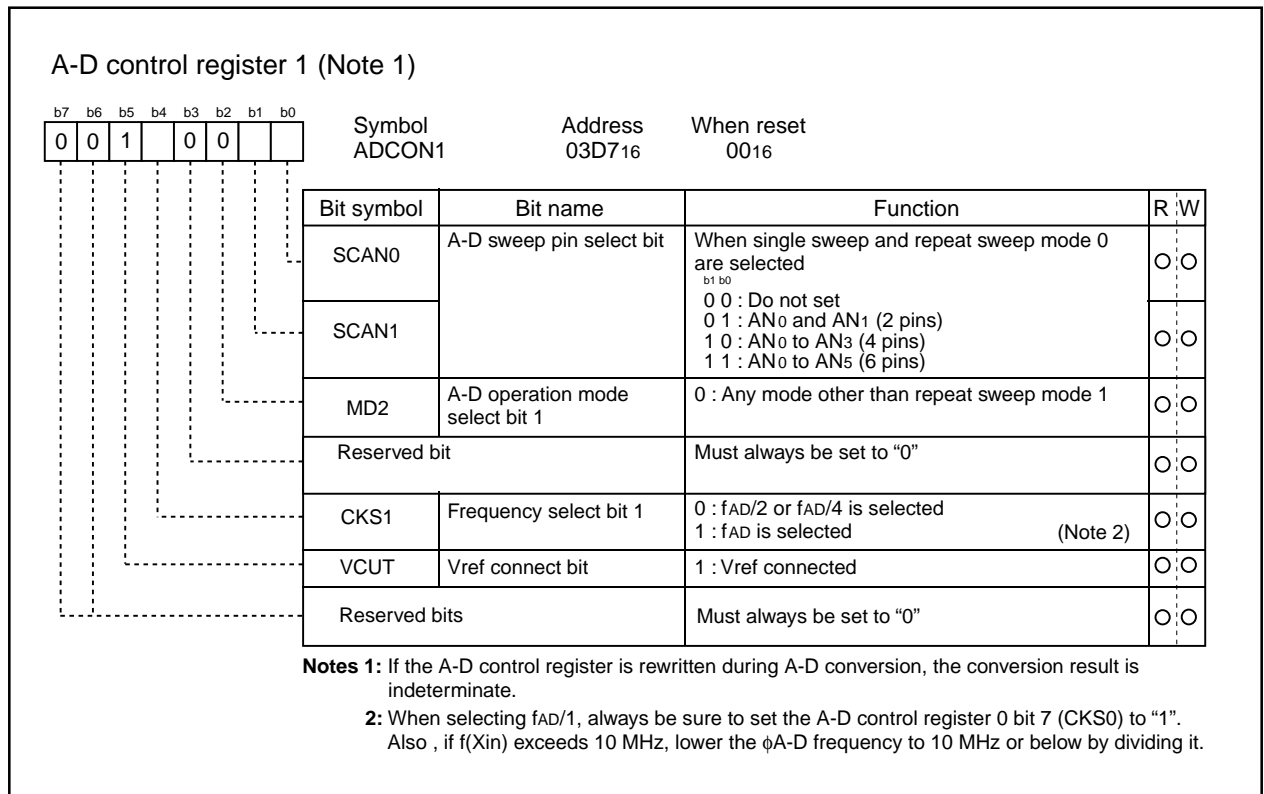
In single sweep mode, the pins selected using the A-D sweep pin select bit are used for one-by-one A-D conversion. Table 2.12.4 shows the specifications of single sweep mode. Figures 2.12.10 and 2.12.11 show the A-D control register in single sweep mode.

**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"><li>• End of A-D conversion</li><li>• Writing "0" to A-D conversion start flag</li></ul>
Interrupt request generation timing	End of A-D conversion
Input pin	AN <sub>0</sub> and AN <sub>1</sub> (2 pins), AN <sub>0</sub> to AN <sub>3</sub> (4 pins), AN <sub>0</sub> to AN <sub>5</sub> (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**



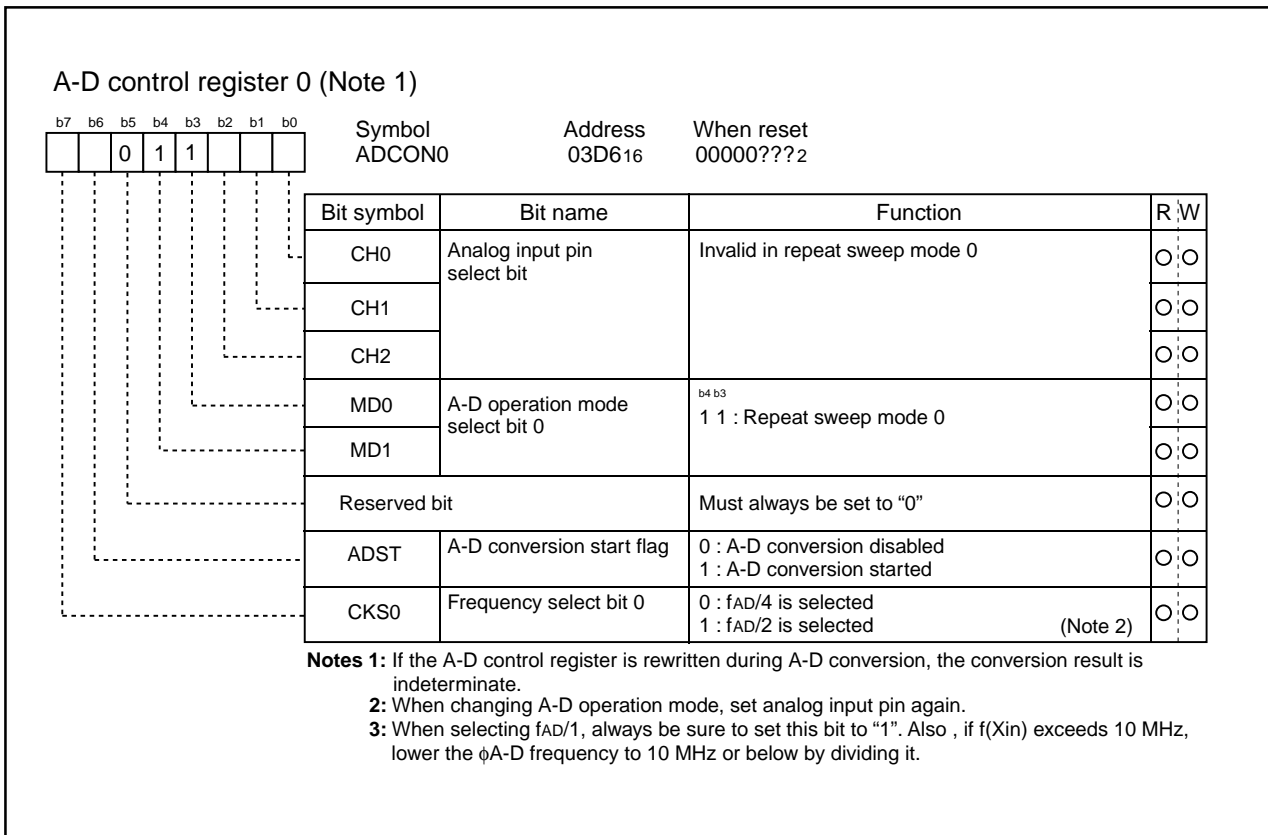
**Figure 2.12.11 A-D control register 1 in single sweep mode**

### 2.12.4 Repeat Sweep Mode 0

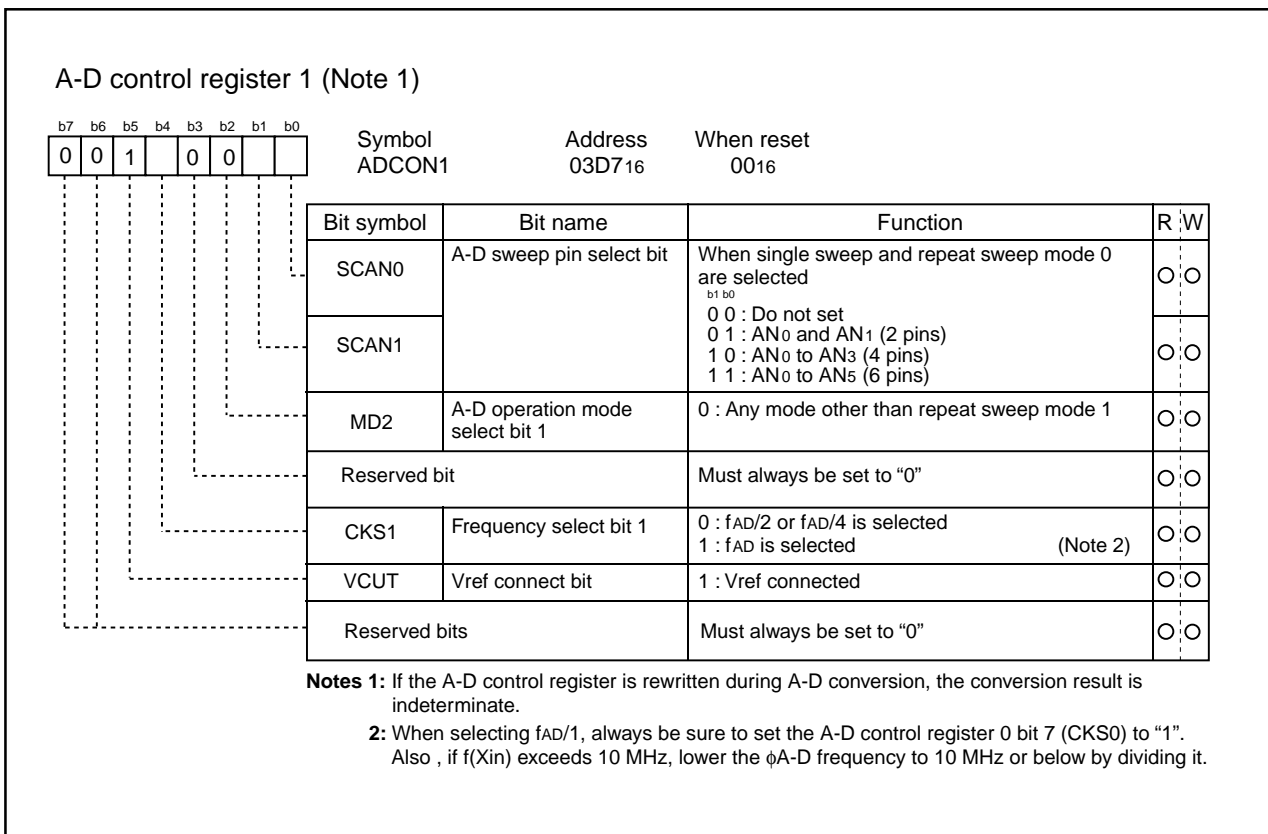
In repeat sweep mode 0, the pins selected using the A-D sweep pin select bit are used for repeat sweep A-D conversion. Table 2.12.5 shows the specifications of repeat sweep mode 0. Figures 2.12.12 and 2.12.13 show the A-D control register in repeat sweep mode 0.

**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**



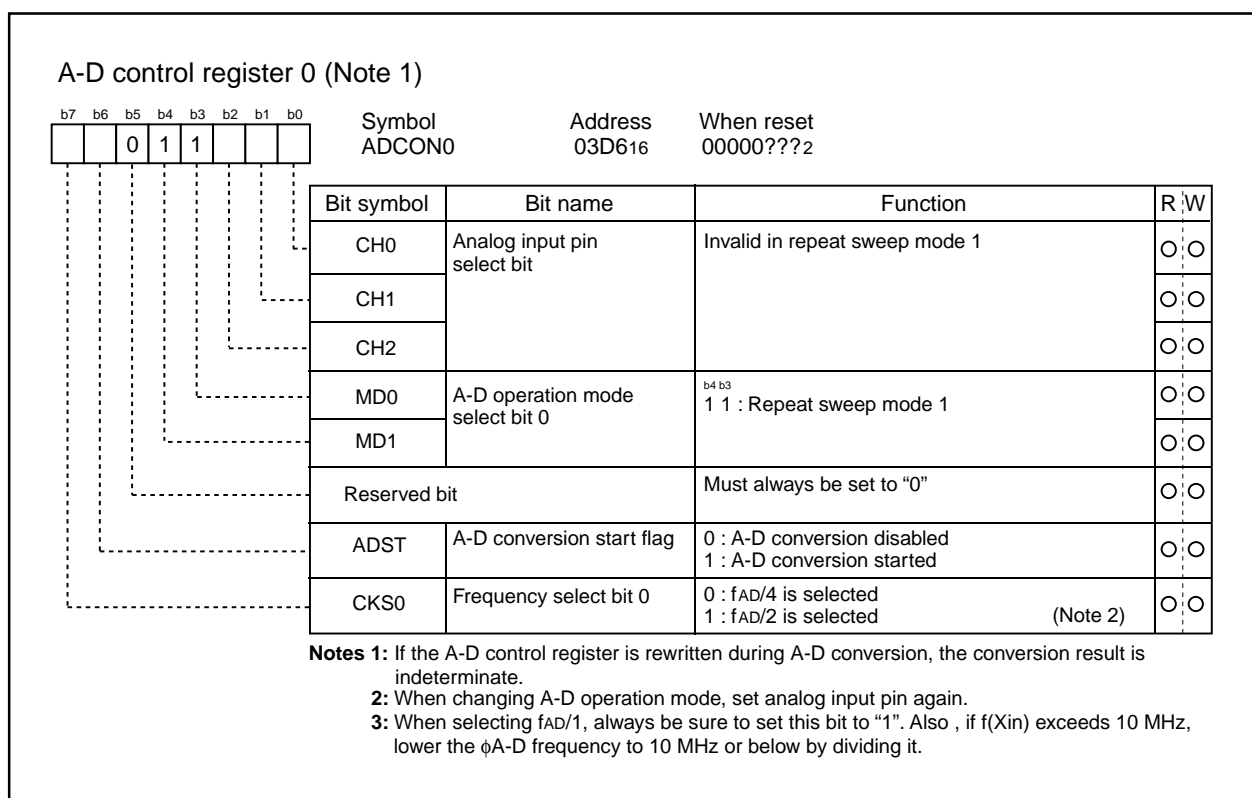
**Figure 2.12.13 A-D control register 1 in repeat sweep mode 0**

### 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 <sub>0</sub> selected AN <sub>0</sub> → AN <sub>1</sub> → AN <sub>0</sub> → AN <sub>2</sub> → AN <sub>0</sub> → AN <sub>3</sub> , etc
Start condition	Writing "1" to A-D conversion start flag
Stop condition	Writing "0" to A-D conversion start flag
Interrupt request generation timing	None generated
Input pin	AN <sub>0</sub> (1 pin), AN <sub>0</sub> and AN <sub>1</sub> (2 pins)
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**

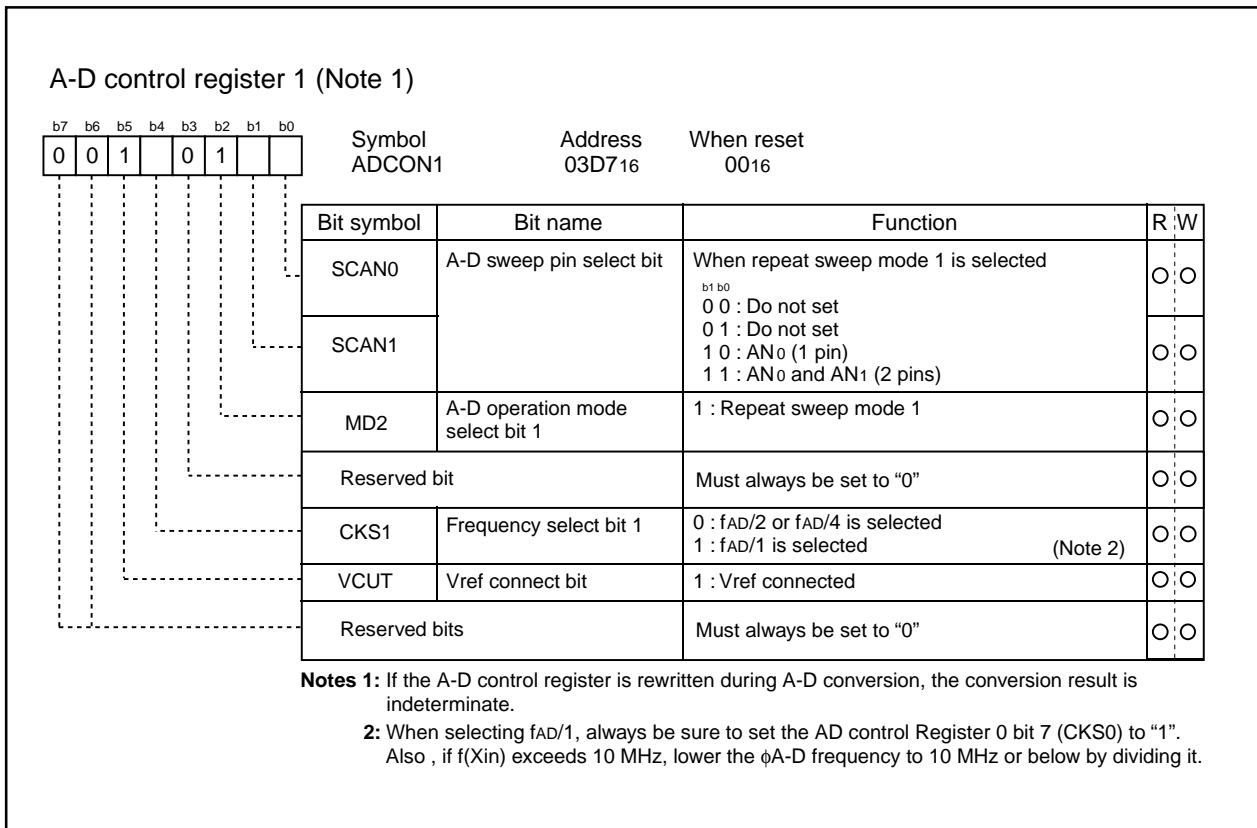


Figure 2.12.15 A-D control register 1 in repeat sweep mode 1

### **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  $\phi_{AD}$  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.

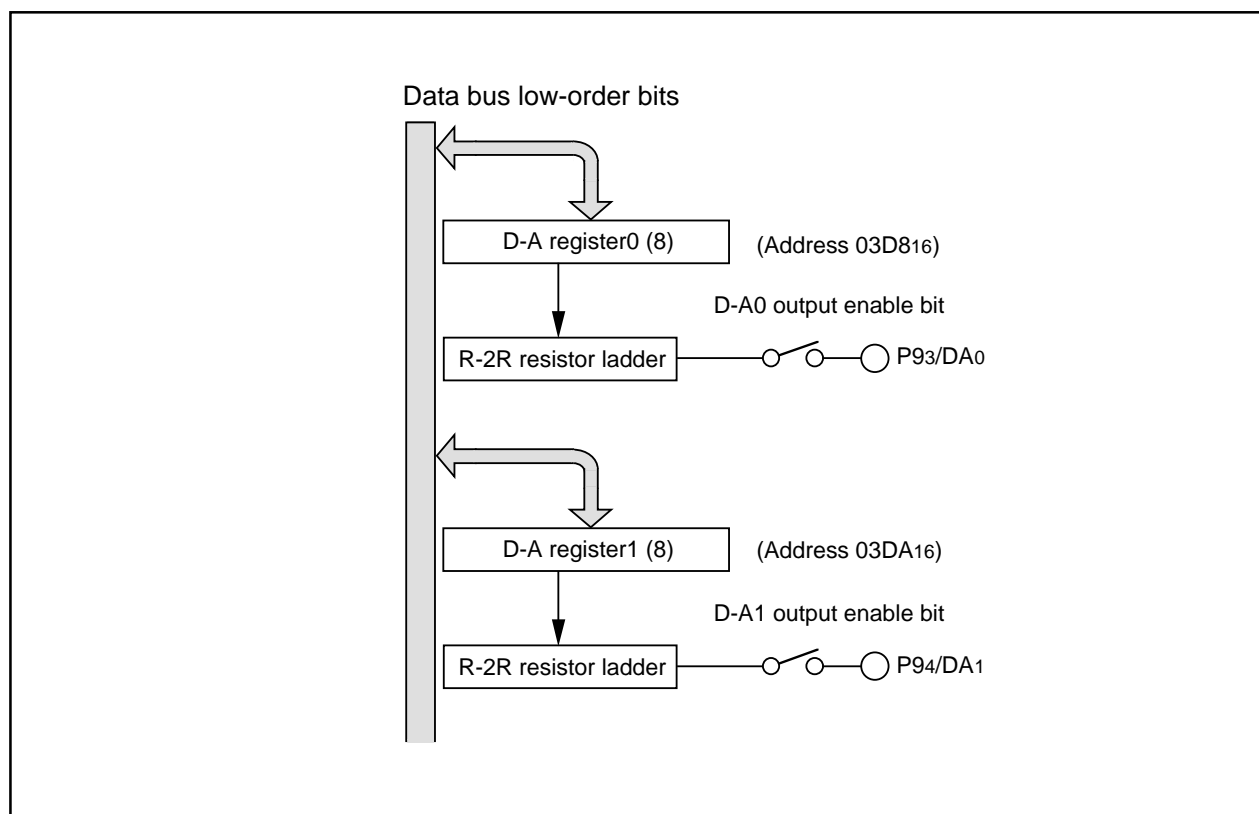
$$V = V_{REF} \times n / 256 \quad (n = 0 \text{ to } 255)$$

V<sub>REF</sub> : 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**

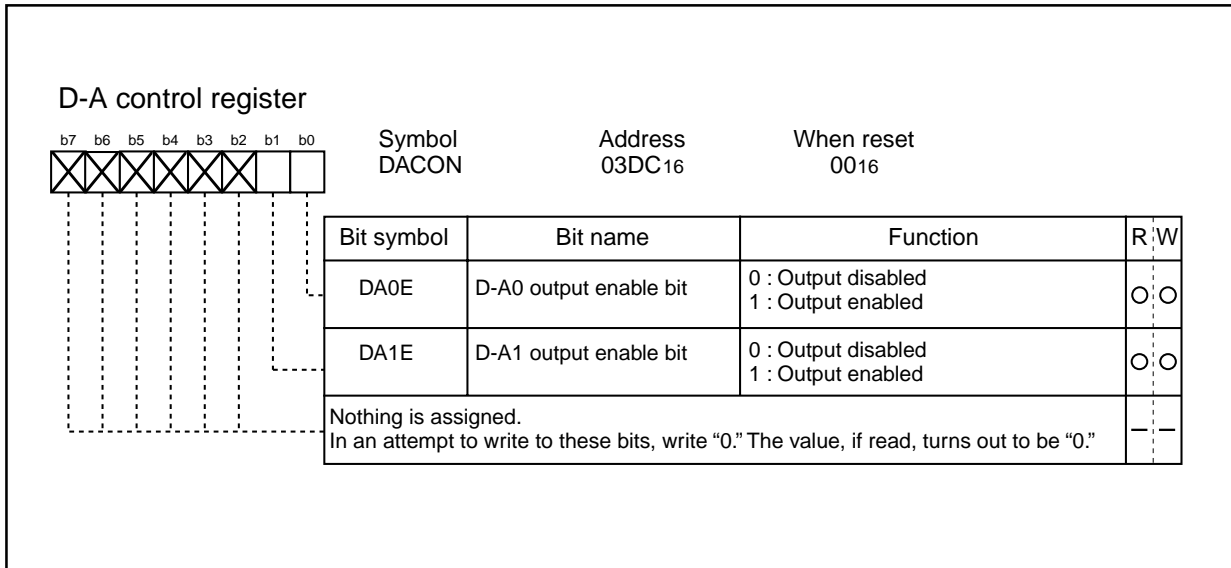


Figure 2.13.2 D-A control register

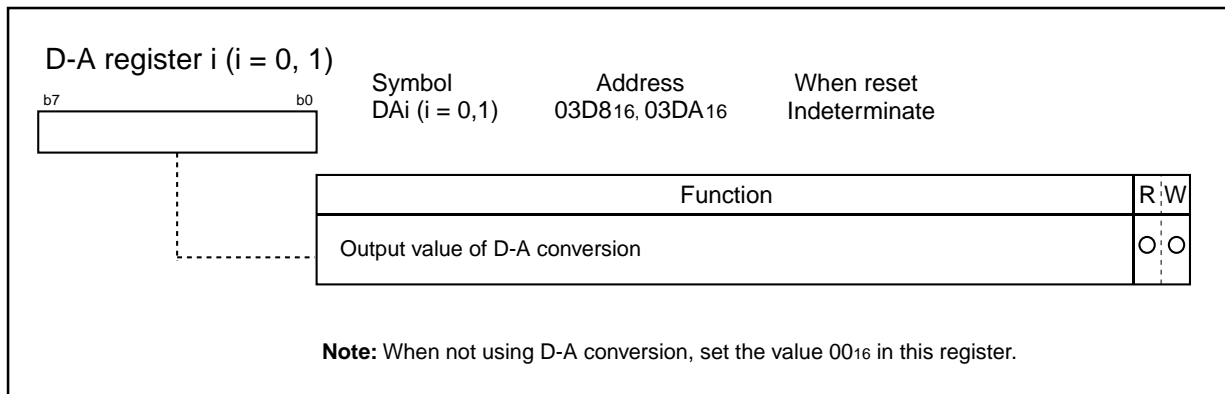


Figure 2.13.3 D-A register i (i = 0 and 1)

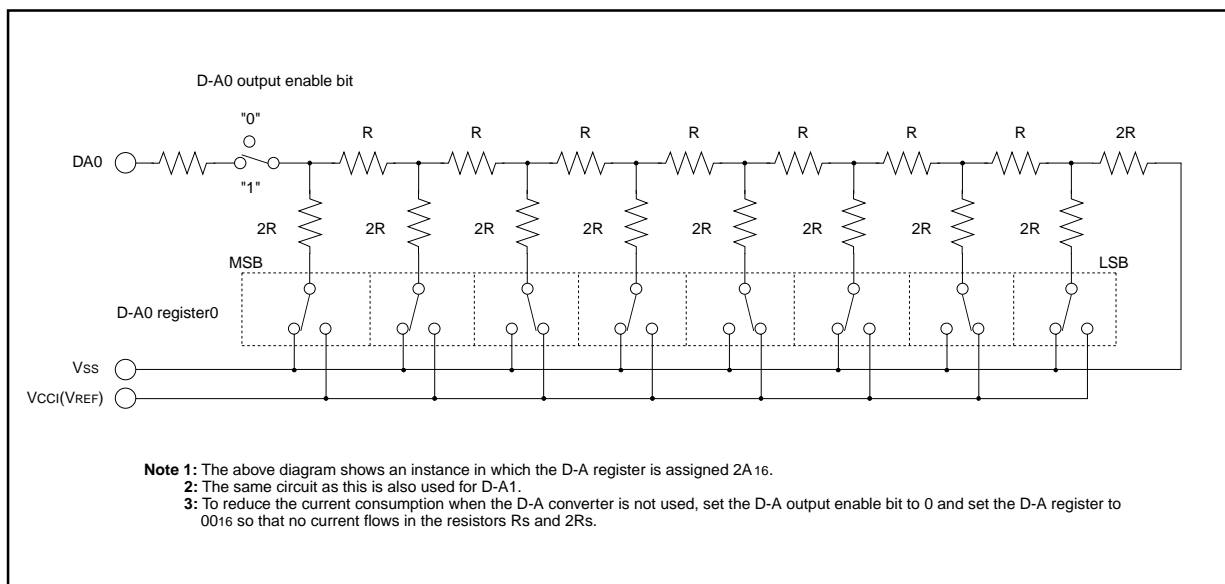


Figure 2.13.4 D-A converter equivalent circuit

### 2.14 Data Slicer

This microcomputer includes the data slicer function for the closed caption decoder (referred to as the CCD) and video ID (referred to as the ID1). This function takes out CC and ID1 (note 2) 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 CV<sub>IN</sub> 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 0260<sub>16</sub>/0300<sub>16</sub>) to "0." These settings can realize the low-power dissipation.

- Notes 1.** When using the data slicer, set bit 7 of the peripheral mode register (address 027D<sub>16</sub>) according to the main clock frequency.
- 2.** 525i/p:ID1 data slice can be performed. No CC data slice at 525p.
- 3.** When there is no specification, it becomes the publication about 525i below.

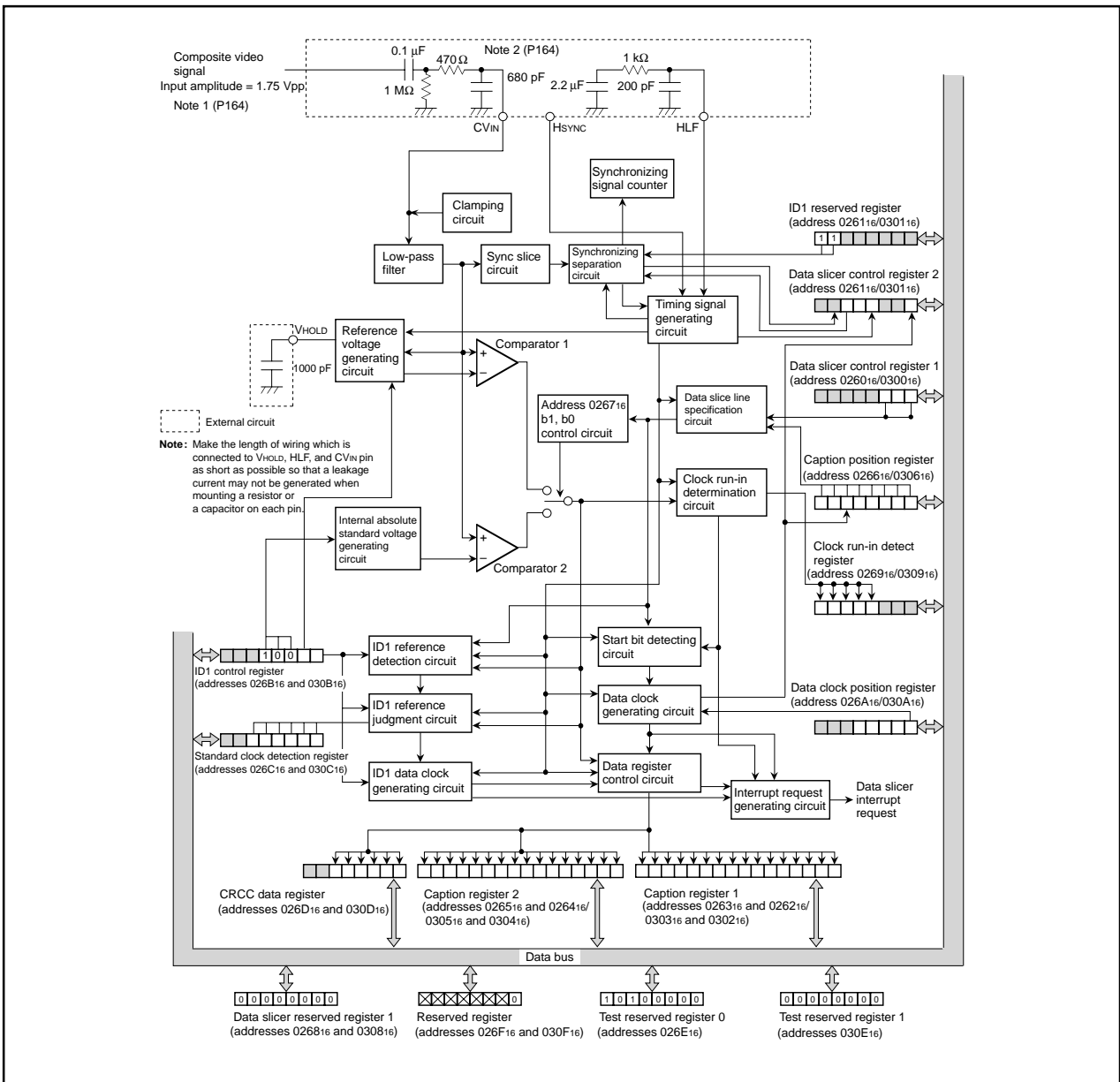


Figure 2.14.1 Data slicer block diagram

Notes 1 : Set up the amplitude inputted from CVIN pin to satisfy the following conditions.

(1) Set up as below :

input amplitude + synchronized chip clamp potential <math>< V\_{CCI} + 0.3 V.</math>

$V_{CCI}$  shows  $V_{CCI}$  power supply pin voltage.

Sink tip clamp pin serves as  $(43/120) \times V_{CCI}$  .

Example) In the case of  $V_{CCI} = 3.3V$  input amplitude = 2.0V

$$2.0V + 1.18 V = 3.18 V < 3.6 V = 3.3 V + 0.3 V$$

(2) Each signal level to input amplitude of CVIN pin is shown in Figure 2.14.2.

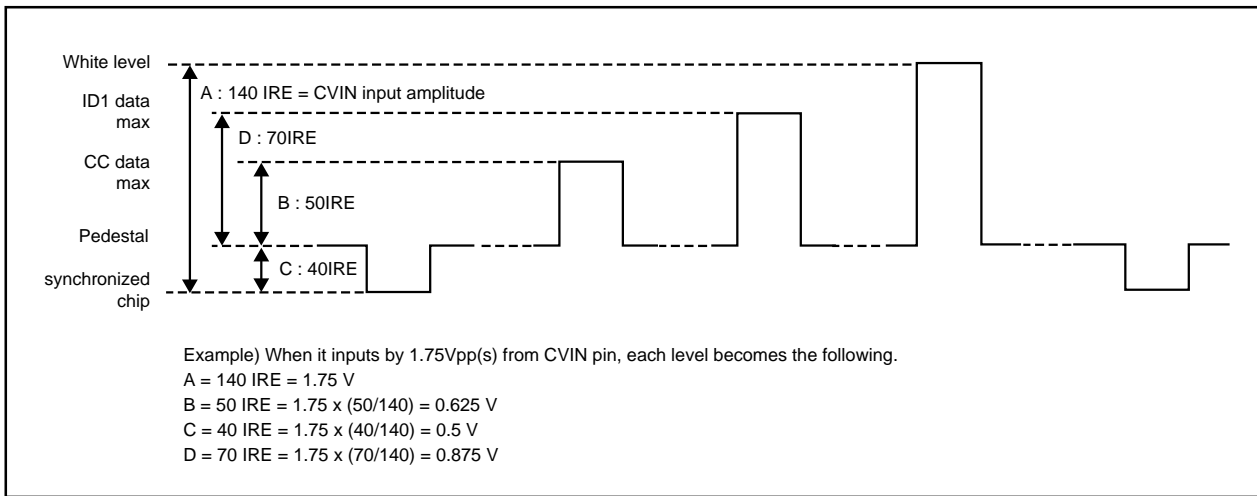


Figure 2.14.2 Each signal level to input amplitude of CVIN pin

Notes 2 : External each constant shown in Figure 2.14.1 is an example, and is greatly influenced by video signal output impedance, substrate capacity, etc. on a system. Evaluate input amplitude and external each constant perfectly, and determine it.

### 2.14.1 Notes when not Using Data Slicer

When bit 0 of data slicer control register 1 (address 026016/030016) is "0," terminate the pins as shown in Figure 2.14.3

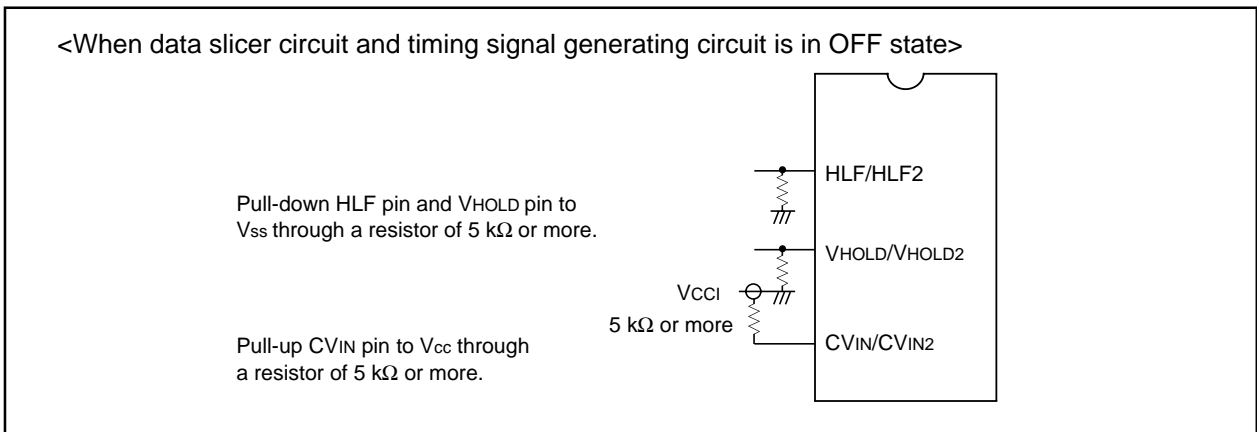


Figure 2.14.3 Termination of data slicer input/output pins when data slicer circuit and timing generating circuit is in OFF state

Figures 2.14.4 and 2.14.5 the data slicer control registers.

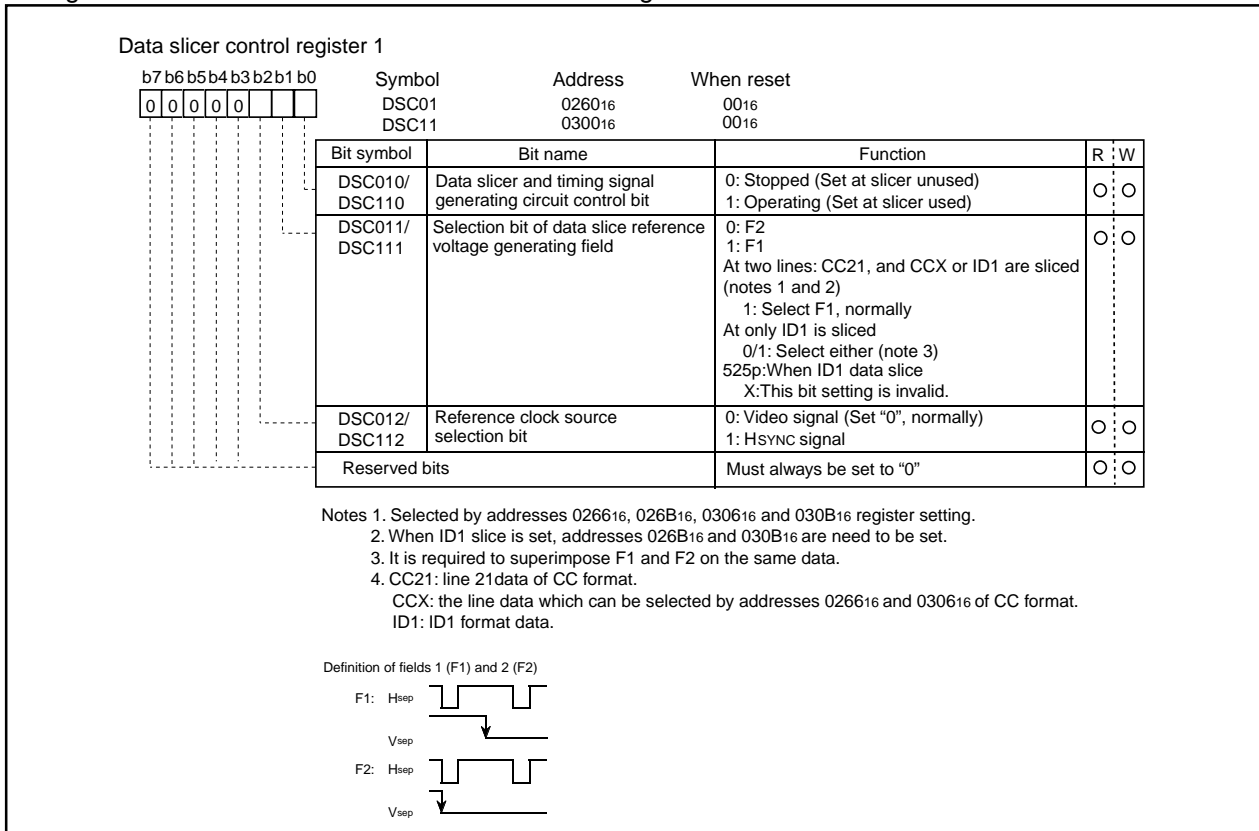


Figure 2.14.4 Data slicer control register 1

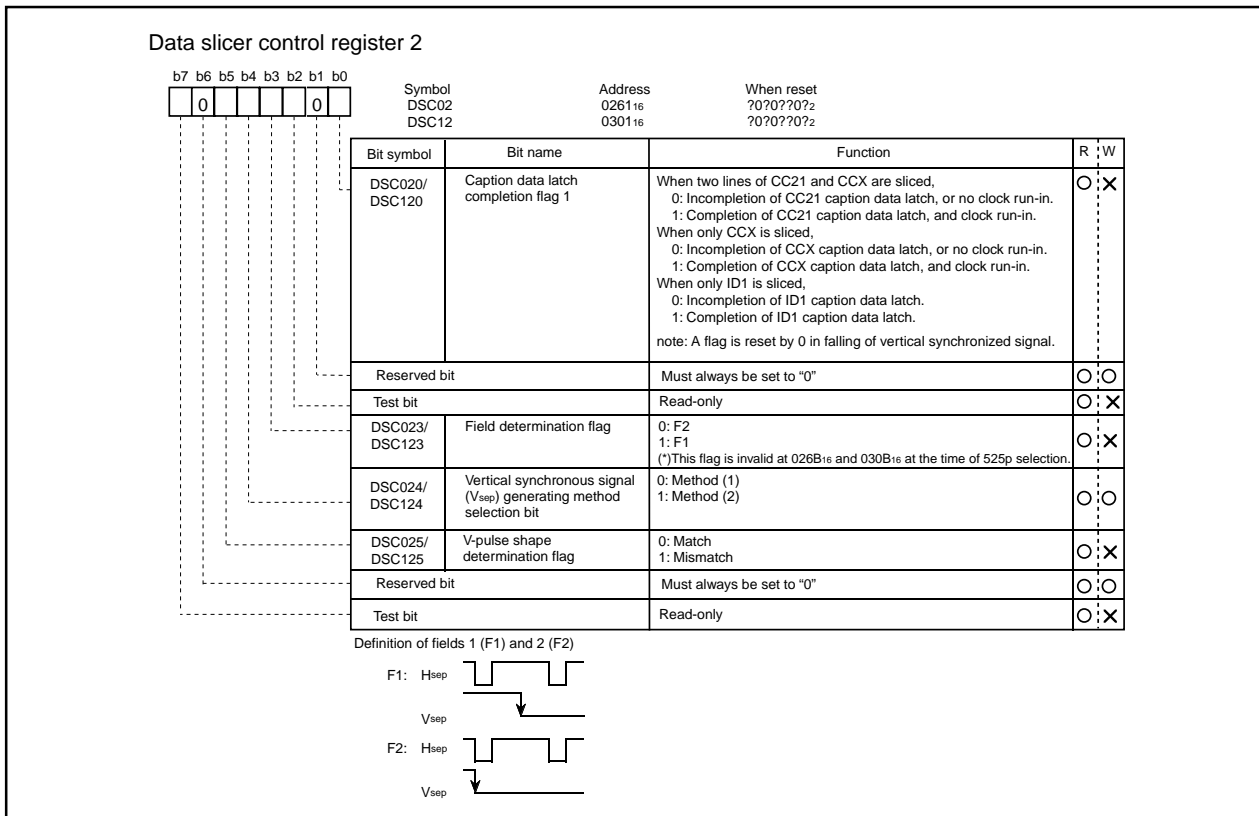


Figure 2.14.5 Data slicer control register 2

## 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  $\mu$ F) coupling outside. Pull down the CVIN pin with a resistor of hundreds of kilohms to 1 M $\Omega$ . 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 and notes).

## 2.14.3 Sync Slice Circuit

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

Set bit 6 and 7 to 11b of ID1 reserved register (addresses 037C16 and 031D16) show in Fig 2.14.21.

## 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 026116/030116).

- 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 exits 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."

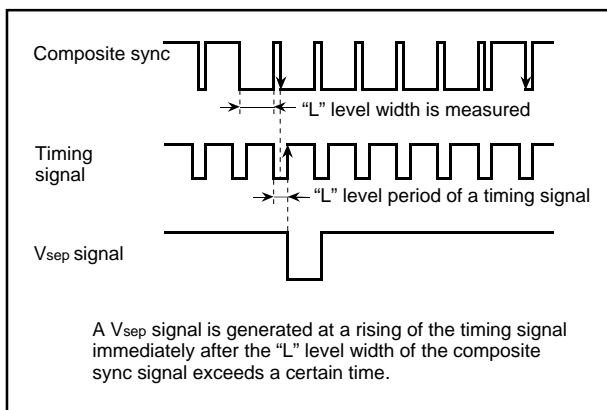


Figure 2.14.6 Vsep generating timing (method 2)

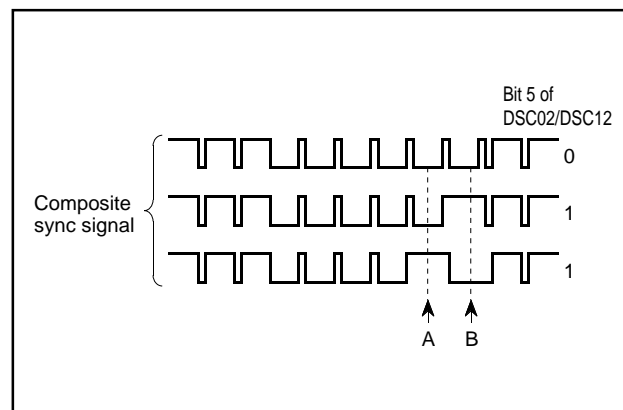


Figure 2.14.7 Determination of v-pulse waveform

### 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 0260<sub>16</sub>/0300<sub>16</sub>) to "1."

The reference clock is 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 0260<sub>16</sub>/0300<sub>16</sub>).

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, H<sub>sep</sub> signals and V<sub>sep</sub> signals become unstable. For this reason, take stabilization time into consideration when programming.

### 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 0266<sub>16</sub>/0306<sub>16</sub>) is used for each setting (refer to Table 2.14.1).

The counter is reset at the falling edge of V<sub>sep</sub> and is incremented by 1 every H<sub>sep</sub> pulse. When the counter value matched the value specified by bits 4 to 0 of the caption position register, this H<sub>sep</sub> is sliced.

The values of "00<sub>16</sub>" to "1F<sub>16</sub>" can be set in the caption position register (at setting only 1 appropriate line, refer to Table 2.14.1). Figure 2.14.8 shows the signals in the vertical blanking interval. Figure 2.14.9 shows the caption position register.

When slice ID1, set bits 0 to 4 of addresses 0266<sub>16</sub> and 0306<sub>16</sub> = 10000b.

525p:When ID1 data slice, set up addresses 0266<sub>16</sub>/0306<sub>16</sub> bit 4-0 = 00001b and the data clock position register (addresses 026A<sub>16</sub> and 030A<sub>16</sub>) bit 6, and 5 = 01b.

#### (2) Specification of line to set slice voltage

When slice CC21 and CCX, 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).

When slice ID1, set bit 6 and 7 of addresses 0266<sub>16</sub> and 0306<sub>16</sub> = 00b or 01b.

525p:When ID1 data slice, set up the addresses 0266<sub>16</sub> and 0306<sub>16</sub> bit 7 and 6 = 01b.

#### (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<sub>sep</sub>.

525p:When ID1 data slice, this bit setting is invalid.

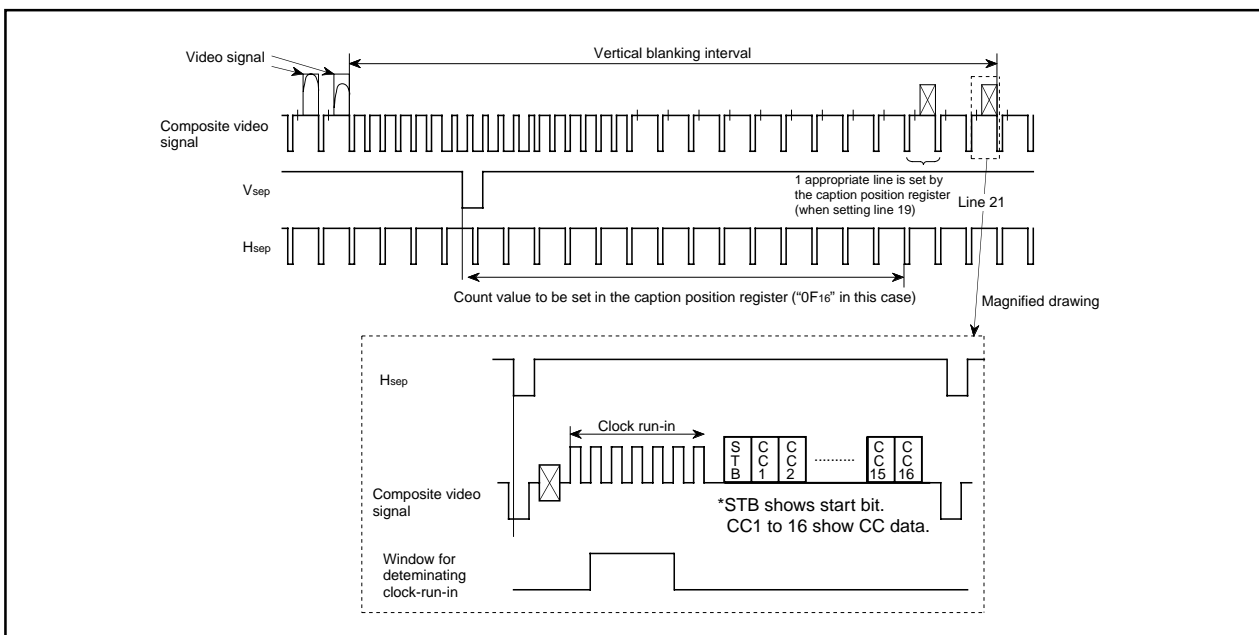


Figure 2.14.8 Signals in vertical blanking interval



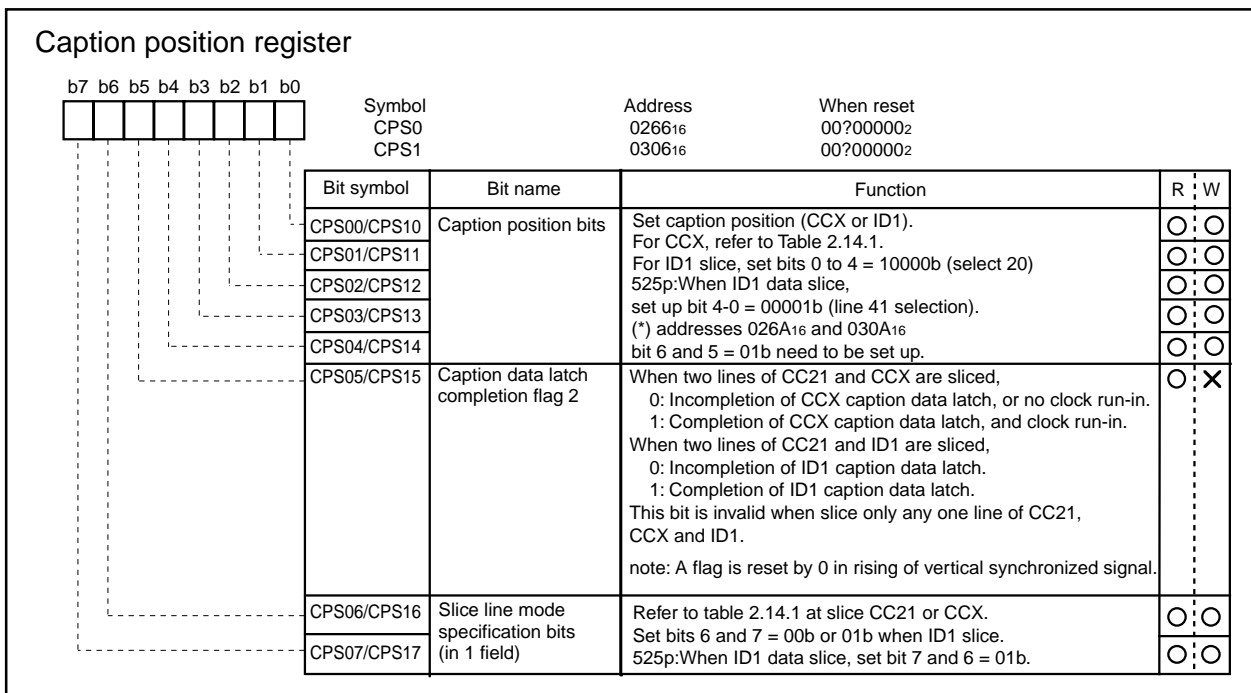


Figure 2.14.9 Caption position register

Table 2.14.1 Specification of data slice line

CPS0/CPS1		Field and Line to Be Sliced Data	Field and Line to Generate Slice Voltage
b7	b6		
0	0	<ul style="list-style-type: none"> <li>Both fields of F1 and F2</li> <li>Line 21 and a line specified by bits 4 to 0 of CPS0/CPS1 (total 2 lines) (See note 2)</li> </ul>	<ul style="list-style-type: none"> <li>Field specified by bit 1 of DSC01/DSC11</li> <li>Line 21 (total 1 line)</li> </ul>
0	1	<ul style="list-style-type: none"> <li>Both fields of F1 and F2</li> <li>A line specified by bits 4 to 0 of CPS0/CPS1 (total 1 line) (See note 3)</li> </ul>	<ul style="list-style-type: none"> <li>Field specified by bit 1 of DSC01/DSC11</li> <li>A line specified by bits 4 to 0 of CPS0/CPS1 (total 1 line) (See note 3)</li> </ul>
1	0	<ul style="list-style-type: none"> <li>Both fields of F1 and F2</li> <li>Line 21 (total 1 line)</li> </ul>	<ul style="list-style-type: none"> <li>Field specified by bit 1 of DSC01/DSC11</li> <li>Line 21 (total 1 line)</li> </ul>
1	1	<ul style="list-style-type: none"> <li>Both fields of F1 and F2</li> <li>Line 21 and a line specified by bits 4 to 0 of CPS0/CPS1 (total 2 lines) (See note 2)</li> </ul>	<ul style="list-style-type: none"> <li>Field specified by bit 1 of DSC01/DSC11</li> <li>Line 21 and a line specified by bits 4 to 0 of CPS0/CPS1 (total 2 lines) (See note 2)</li> </ul>

- Notes 1: DSC01/DSC11 is data slicer control register 1.  
 CPS0/CPS1 is caption position register.  
 2: Set the value of "0016" – "1016" to bits 4 to 0 of CPS0/CPS1.  
 3: Set the value of "0016" – "1F16" to bits 4 to 0 of CPS0/CPS1.

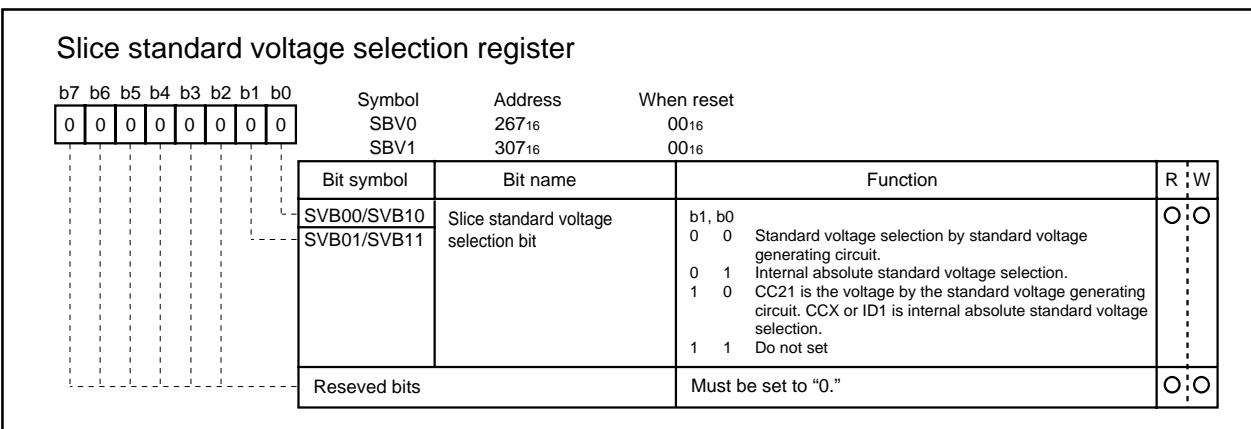


Figure 2.14.10 Slice standard voltage selection register

### 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 and 2.

#### (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.

**Note:** It takes a few tens of lines to generate slice voltage until the slice voltage becomes stable after the data slicer is started. In this period, the slice data becomes unstable. For this reason, take stabilization time into consideration when programming.

#### (2) Comparator 1

The comparator 1 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.

#### (3) Comparator 2

The comparator 2 compares the absolute standard voltage generated inside from the voltage and power supply voltage of a composite video signal, and converts the composite video signal into a digital value.

### 2.14.8 CC Start Bit • ID1 Reference Detecting Circuit

This circuit detects a CC start bit • ID1 reference bit at line decided in the data slice line specification circuit.

In the case of CC start bit

- 1) Detect a clock run impulse at counting the input pulse of a data slice line.
- 2) When a clock run impulse is detected, the sampling clock outputted from a timing generating circuit detects a start bit pattern, and judge CC start bit.

In the case of ID1 reference bit

- 1) Detect ID1 reference bit all over the window generated after fixed time from Hsep in a timing signal generating circuit.

### 2.14.9 Clock Run-in Determination Circuit

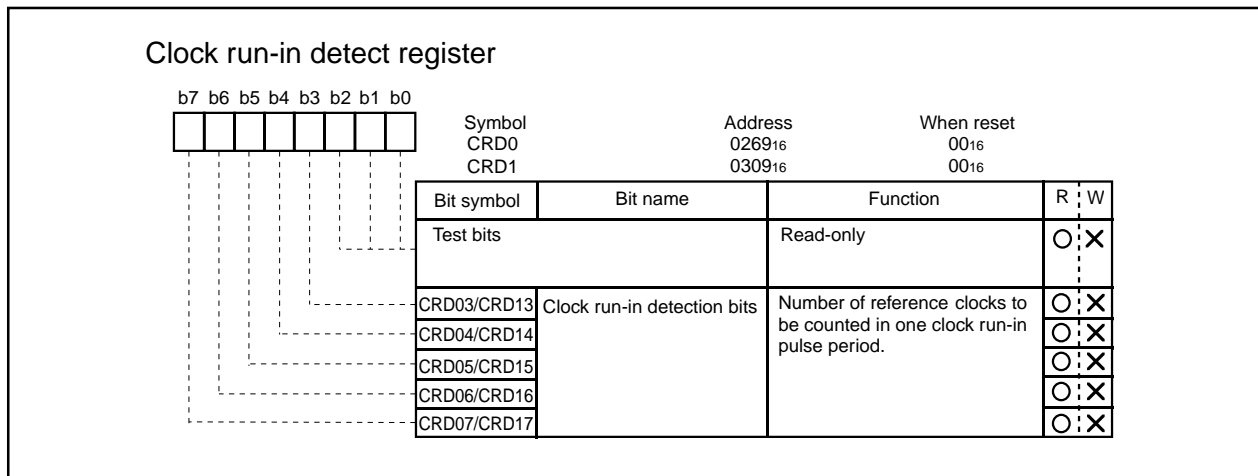
Clock run in judging

By counting the number of pulses all over the specific window of a data slice line, it judges that it is clock run in. When it judges with having no clock run in, the completion flag of a caption data latch is not set to 1. Moreover, the number of standard clocks counted in clock run impulse 1 cycle is stored in the bits 7-3 of a clock run in detection register (addresses 0269<sub>16</sub>/0309<sub>16</sub>).

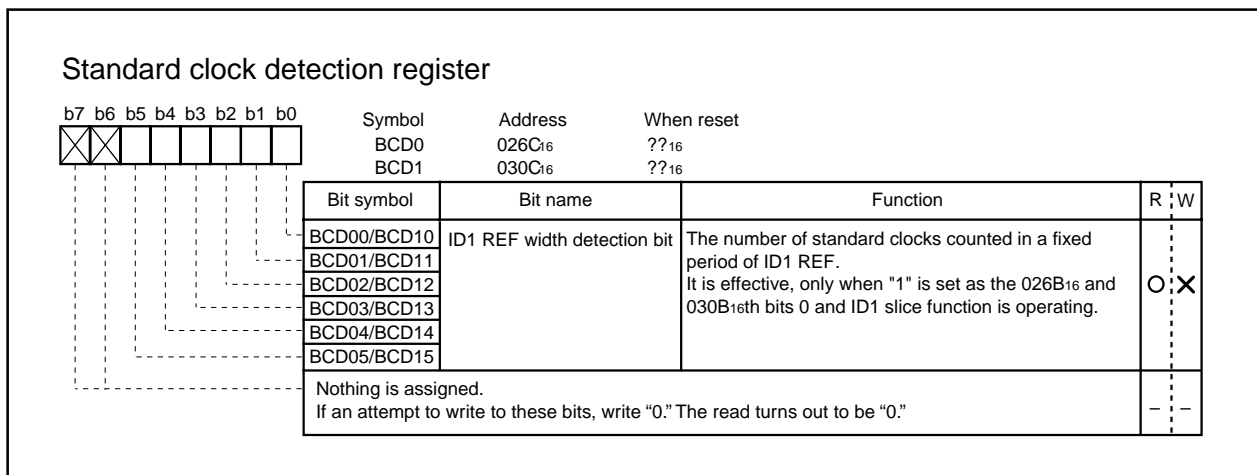
ID1 reference bit judging

The number of standard clocks counted during fixed of ID1 reference bit is stored in the bits 5-0 of a standard clock detection register (addresses 026C<sub>16</sub>/the 030C<sub>16</sub>). Read these bits after generating of data slicer interruption ("(12) interruption demand generating circuit").

Clock run-in detection register is shown in Fig. 2.14.11, standard clock detection register is shown in Fig. 2.14.12.



**Figure 2.14.11 Clock run-in detect register**



**Figure 2.14.12 Standard clock detection register**

### 2.14.10 Data Clock Generating Circuit

At the time of CC data slice

It synchronizes with CC start bit detected in CC start bit detection circuit, and a data clock is generated after the fixed offset set up by the data clock position register (addresses 026A<sub>16</sub>/030A<sub>16</sub>). A data clock is a clock for storing caption data in a caption register. When 16-bit data is stored in a caption register and judged in a clock run in judging circuit that has clock run in, the completion flag of a caption data latch is set.

A data clock position register is shown in Fig. 2.14.13.

At the time of ID1 data slice

The data clock which synchronized with ID1 reference bit is generated. With this data clock, the 6 bit data of the remaining CRCC is stored in a caption register for 14-bit data among 20-bit data at a CRCC data register (addresses 026D<sub>16</sub>/030D<sub>16</sub>). If 20-bit data is stored in each register, the completion flag of a caption data latch will be set.

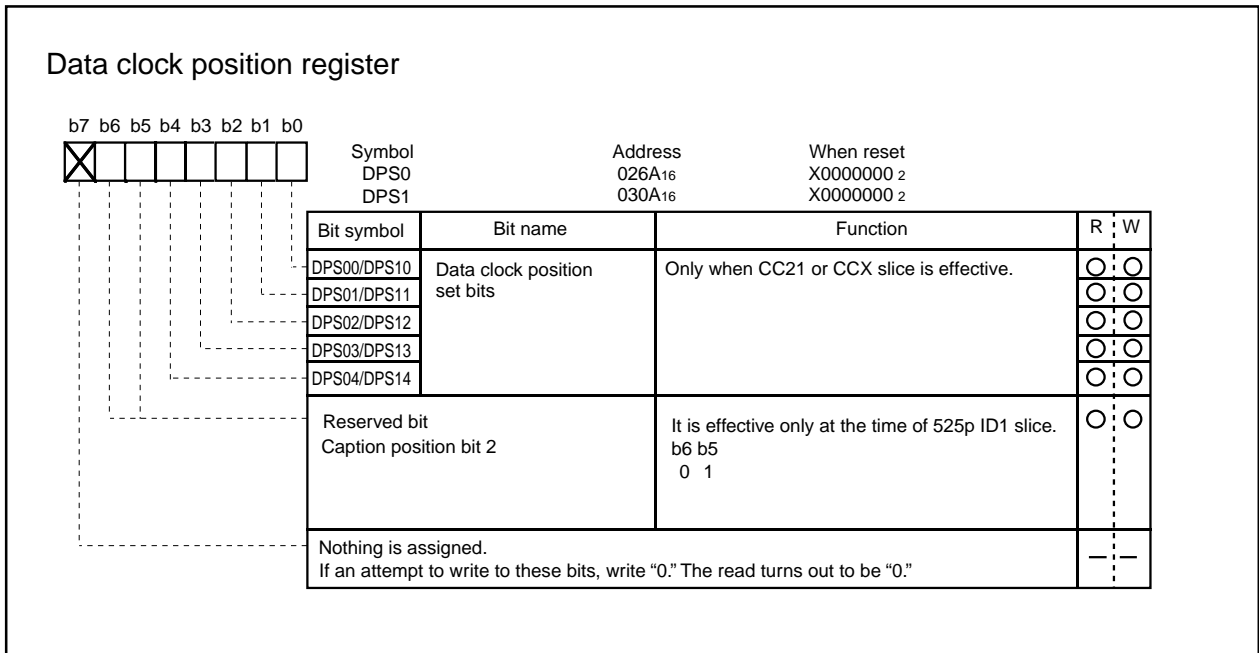


Figure 2.14.13 Data clock position register

### 2.14.11 Caption Register and CRCC Data Register

The caption data converted into a digital value by the comparator is stored into the caption register and CRCC data register in synchronization with the data clock. The contents of the stored caption data can be obtained by reading out the caption data register and CRCC data register. These registers are reset to "0" at a falling of V<sub>sep</sub>. Read out these registers after the occurrence of a data slicer interrupt (refer to "2.14.12 Interrupt request generating circuit").

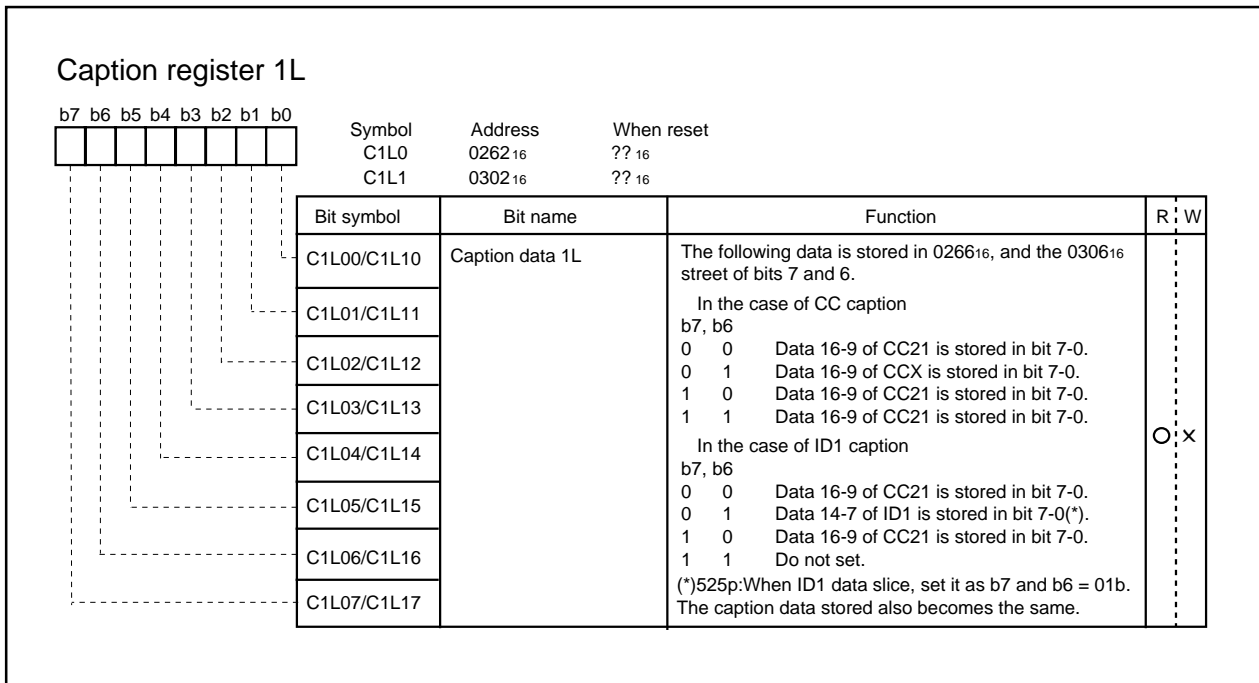


Figure 2.14.14 Caption register 1L

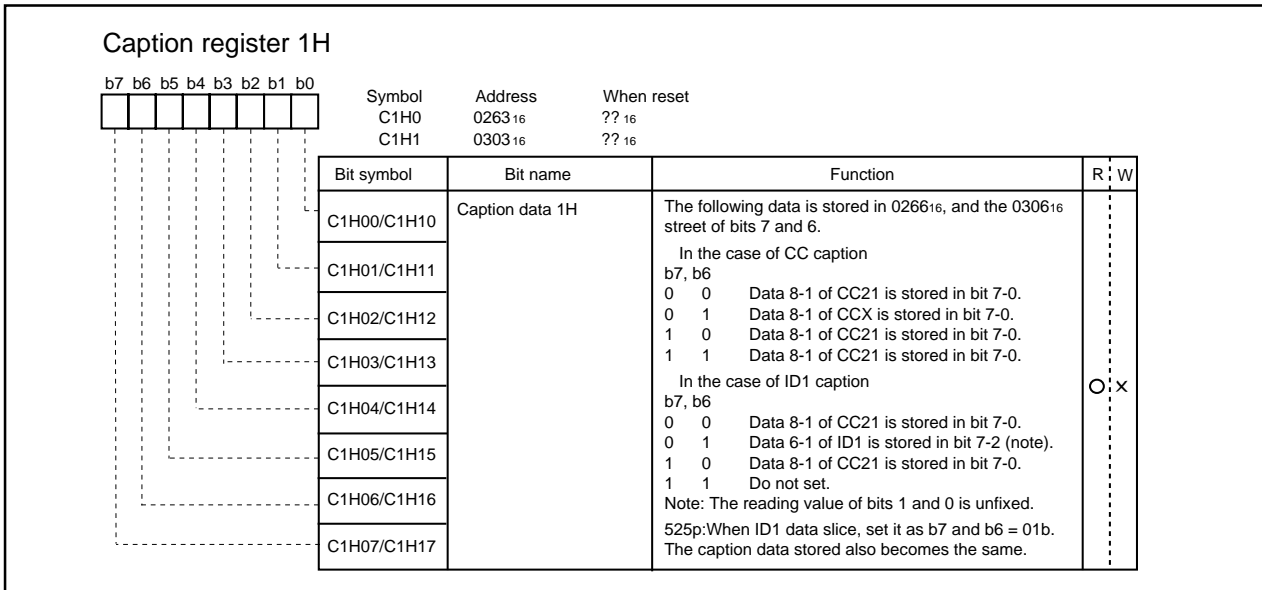


Figure 2.14.15 Caption register 1H

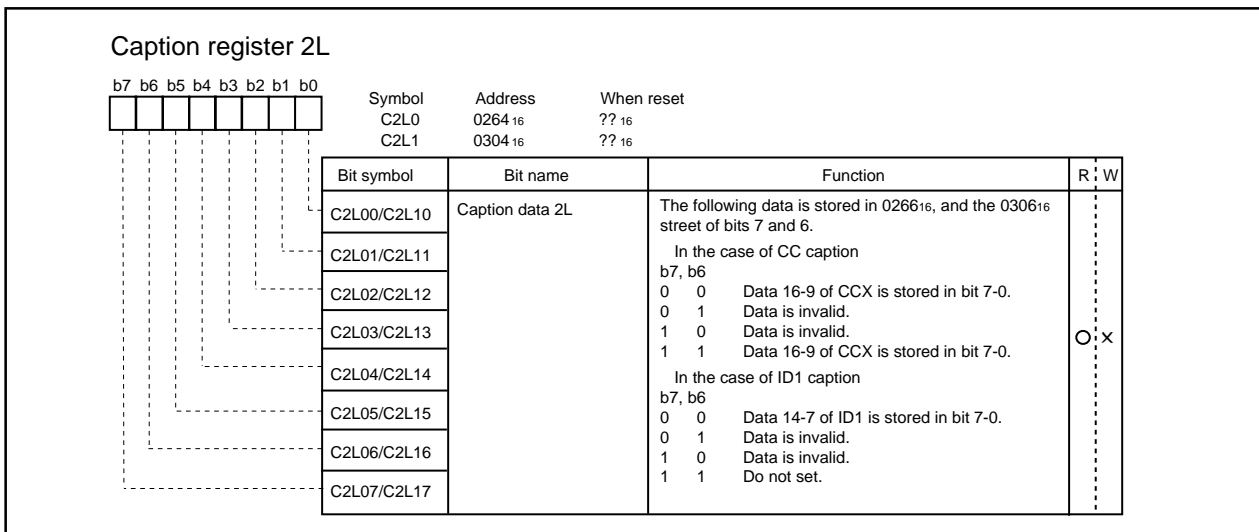


Figure 2.14.16 Caption register 2L

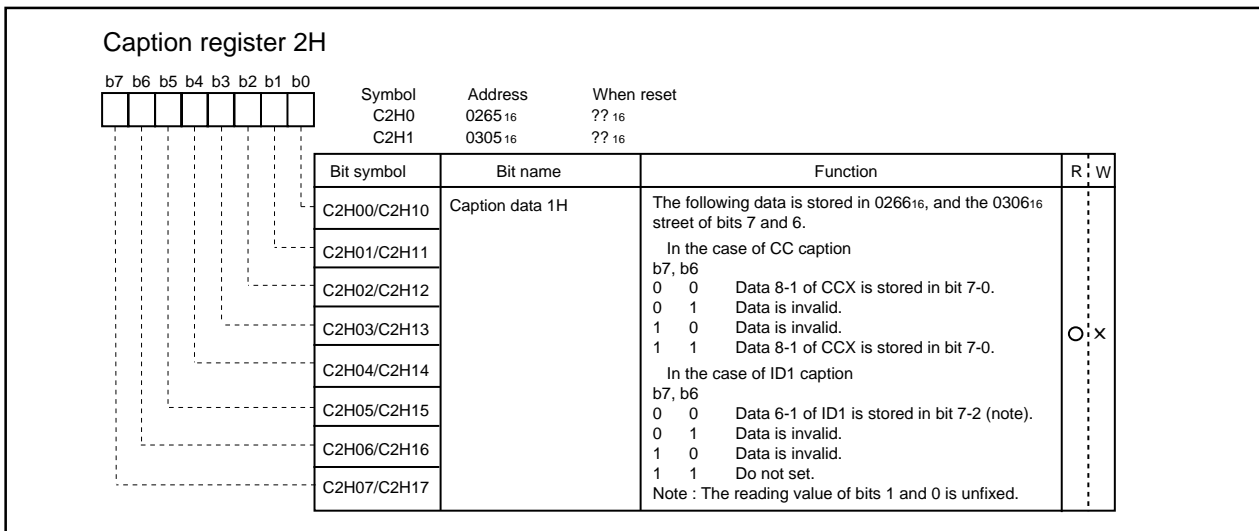


Figure 2.14.17 Caption register 2H

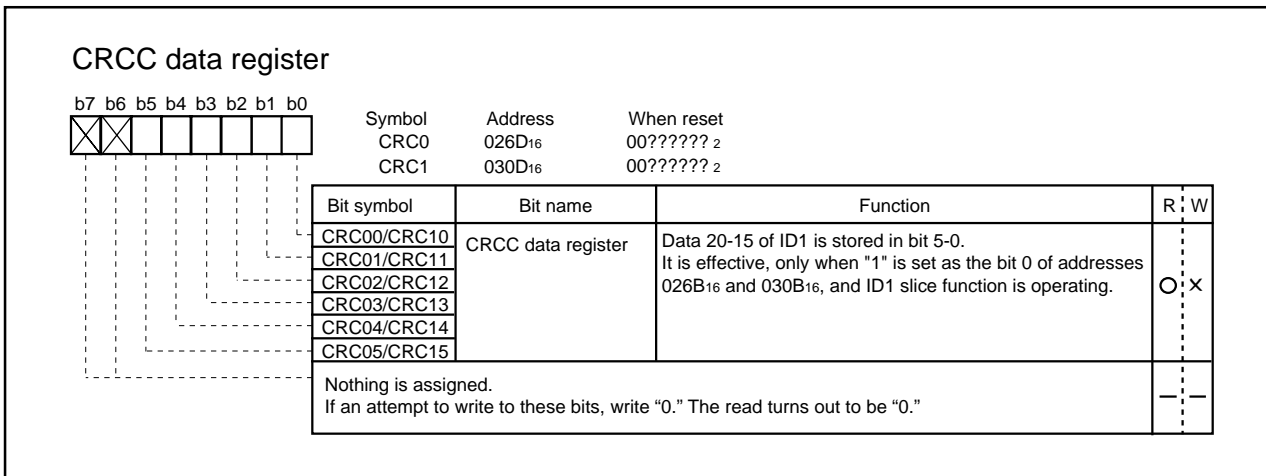


Figure 2.14.18 CRCC data register

### 2.14.12 Interrupt Request Generating Circuit

The interrupt requests as shown in Table 2.14.2 are generated by combination of the following bits; bits 6 and 7 of the caption position register (addresses 0266<sub>16</sub>/0306<sub>16</sub>). Read out the contents of caption data registers 1 and 2, CRCC data register, clock run-in detect register and standard clock detect register after the occurrence of a data slicer interrupt request.

Table 2.14.2 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 (Note)
1	0	After slicing line 21
	1	After slicing line 21

CPS: Caption position register

**Note:** It becomes the one-line back specified in 525p caption position register bits 4 to 0 and the data clock position register bits 6 and 5.

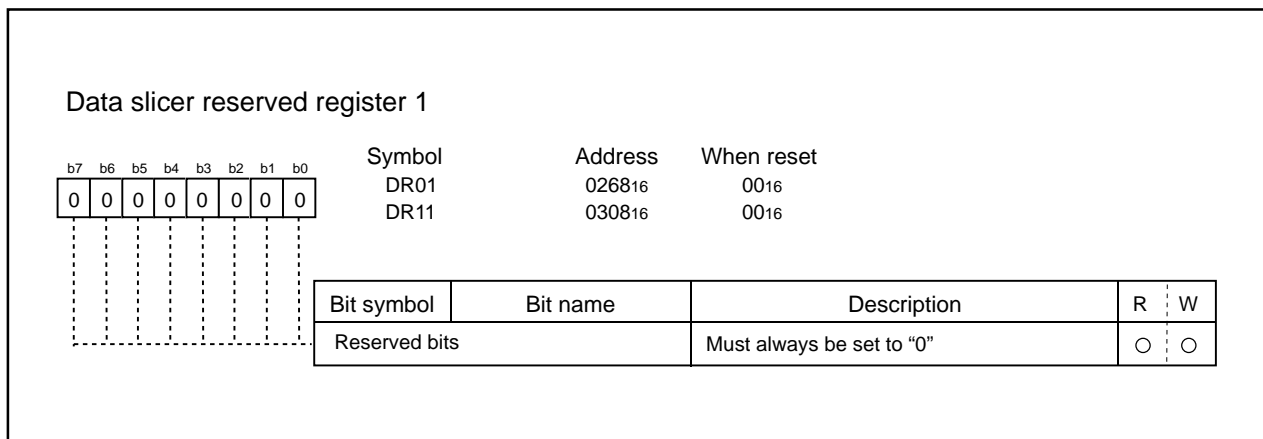


Figure 2.14.19 Data slicer reserved register i (i = 1, 2)

### 2.14.13 ID1 data slice

When data slice ID1, ID1 control register of Fig 2.14.20 needs to be set.

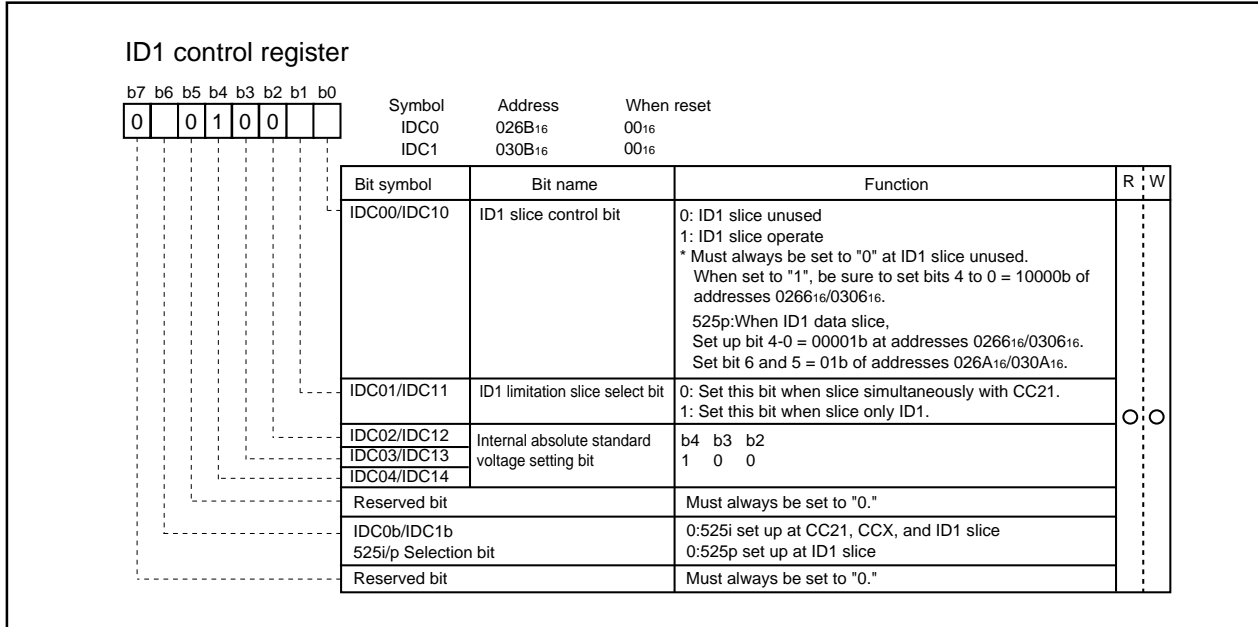


Figure 2.14.20 ID1 control register

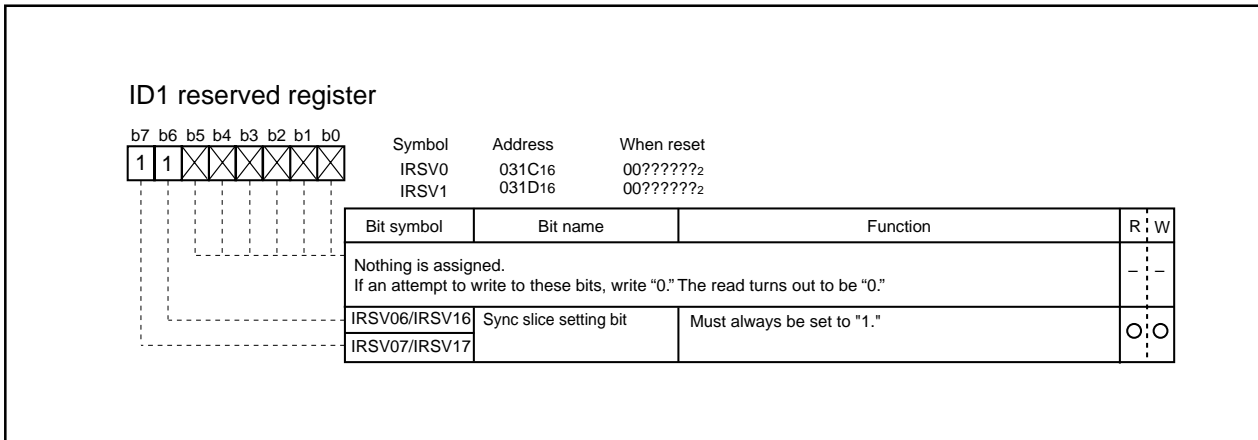


Figure 2.14.21 ID1 reserved register

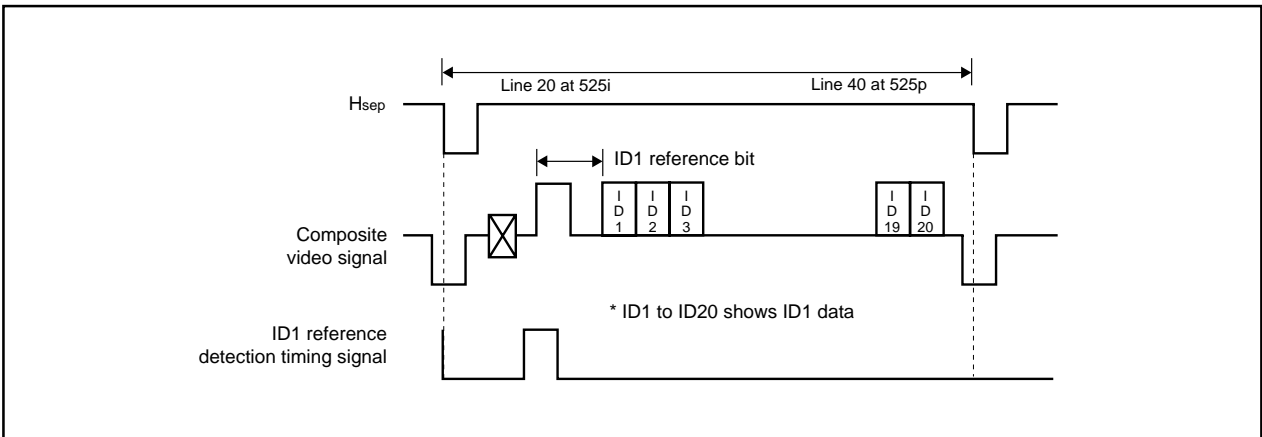


Figure 2.14.22 ID1 signal in vertical blanking interval

### 2.15 HSYNC Counter

The synchronous signal counter counts HSYNC from HSYNC count input pins (HC0/P7<sub>5</sub>, HC1/P7<sub>7</sub>) 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 is stored into the 8-bit latch.

Accordingly, the latch value changes in the cycle of T time. When the count value exceeds “FF<sub>16</sub>,” “FF<sub>16</sub>” is stored into the latch.

The latch value can be obtained by reading out the HSYNC counter latch (address 027F<sub>16</sub>). 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.

- Notes 1:** When using the HSYNC counter, set bit 7 of the peripheral mode register (address 027D<sub>16</sub>) according to the main clock frequency.
- 2:** HSYNC counter latch is a register only for read-out.

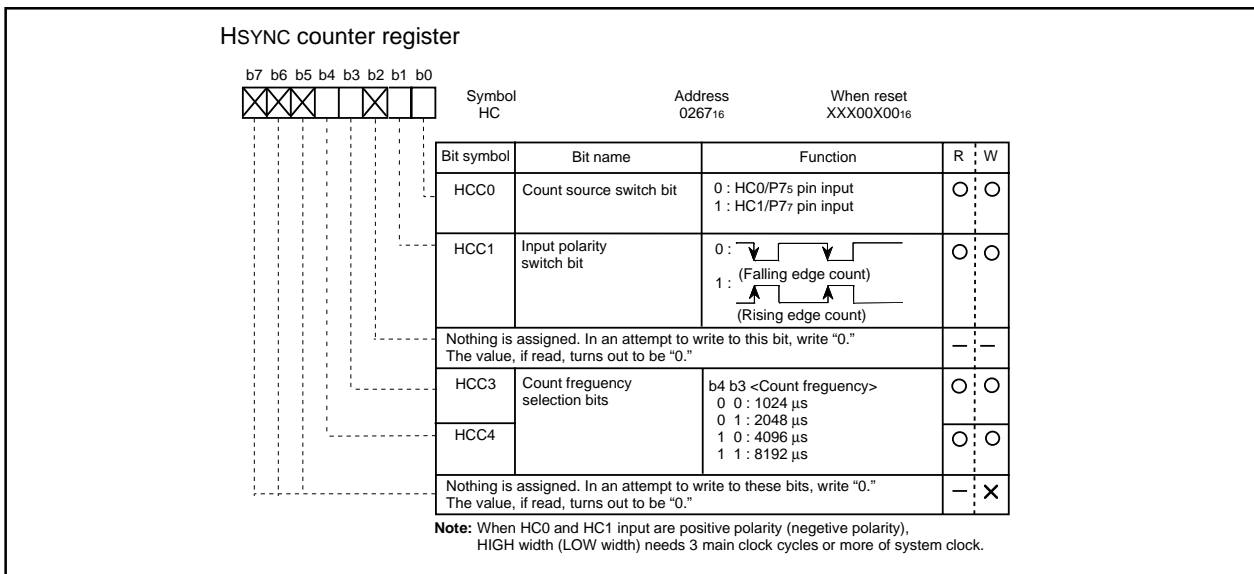


Figure 2.15.1 HSYNC counter register

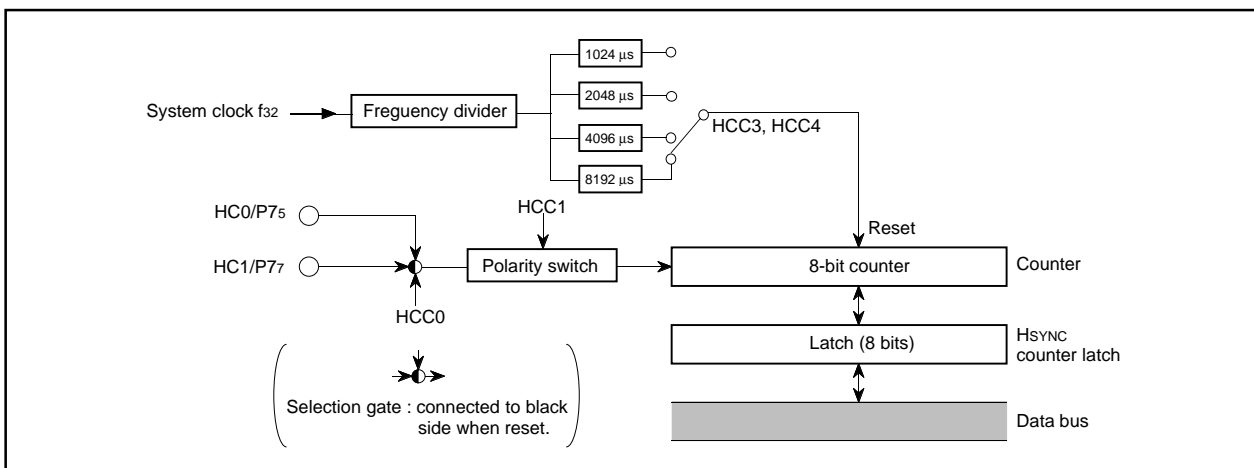


Figure 2.15.2 HSYNC counter block diagram



## 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  $f(XIN) = 16\text{ MHz}$  or  $10\text{ MHz}$ . At this time, set bit 7 (SSCK) of the peripheral mode register according to the XIN frequency to be used.

**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			
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	
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

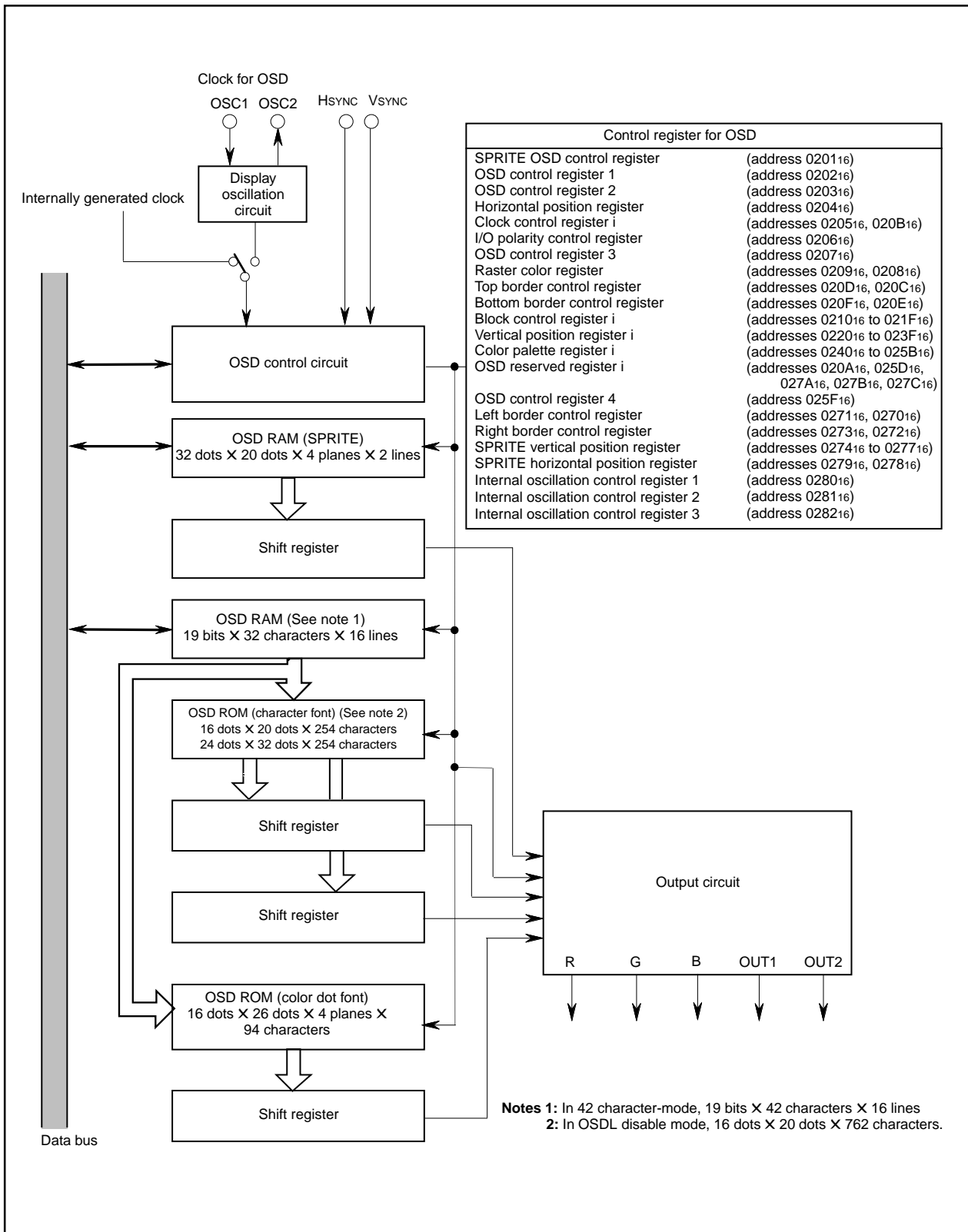


Figure 2.16.2 Block diagram of OSD circuit

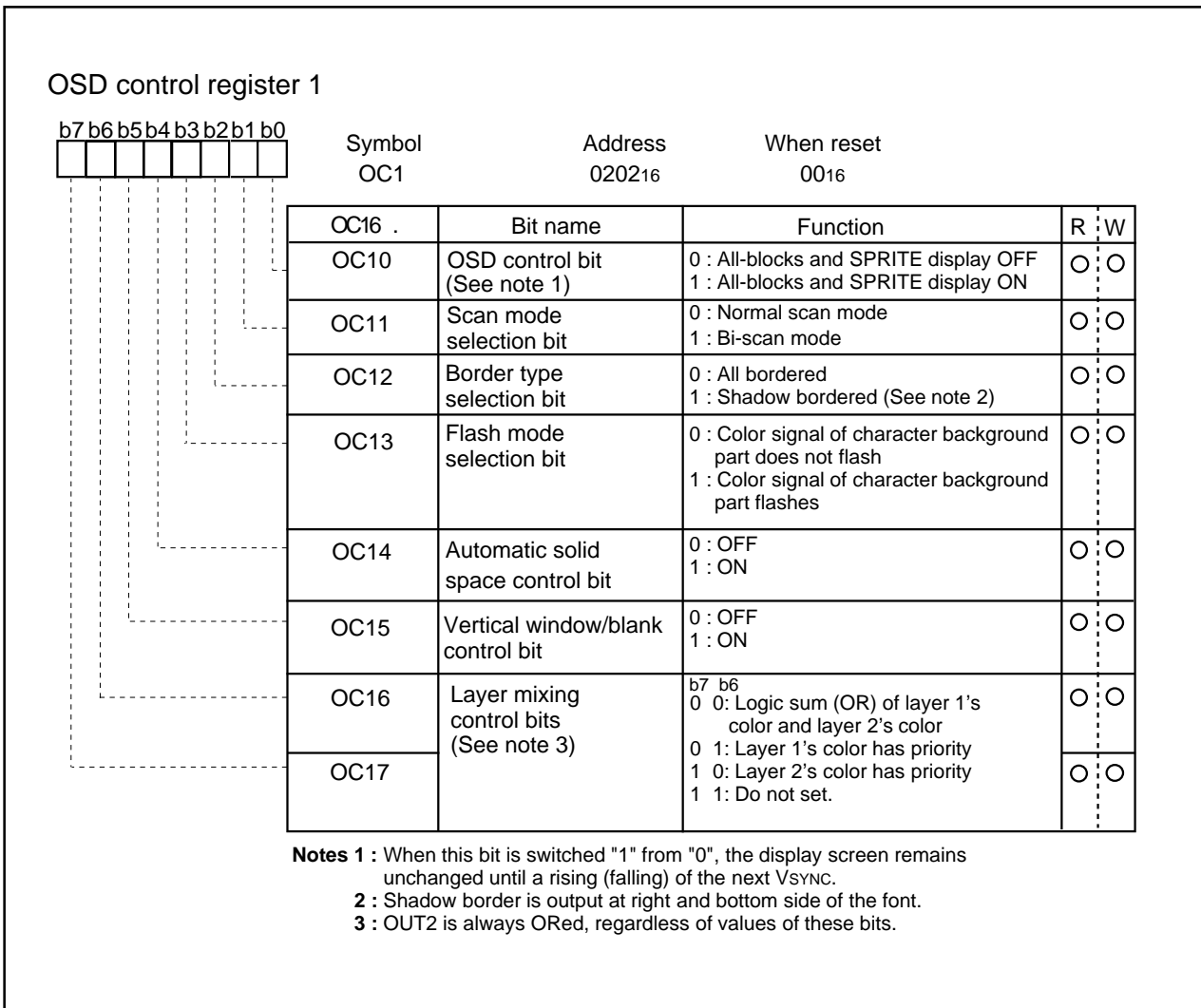


Figure 2.16.3 OSD control register 1

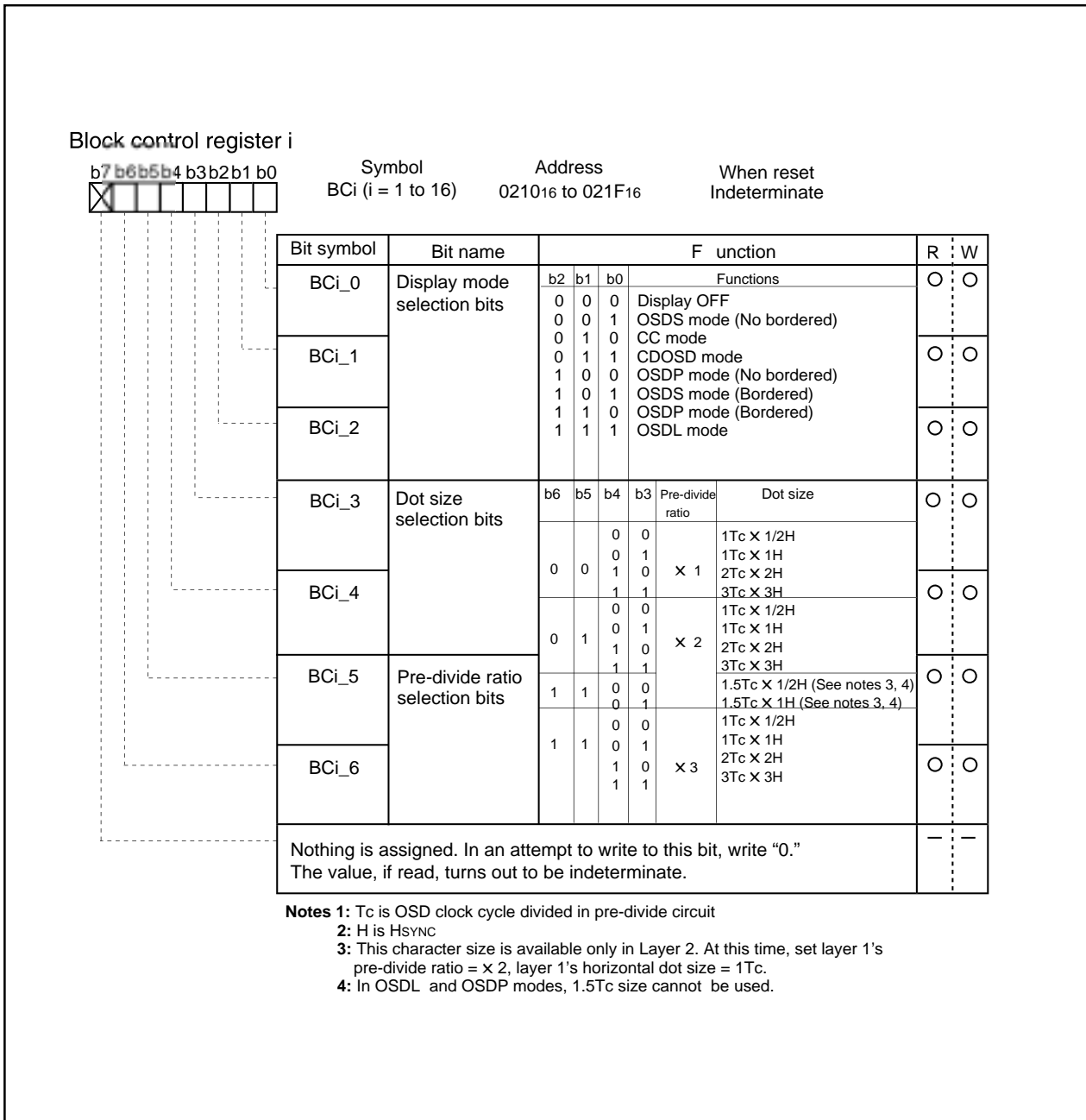


Figure 2.16.4 Block control register i (i = 0 to 16)

### 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.

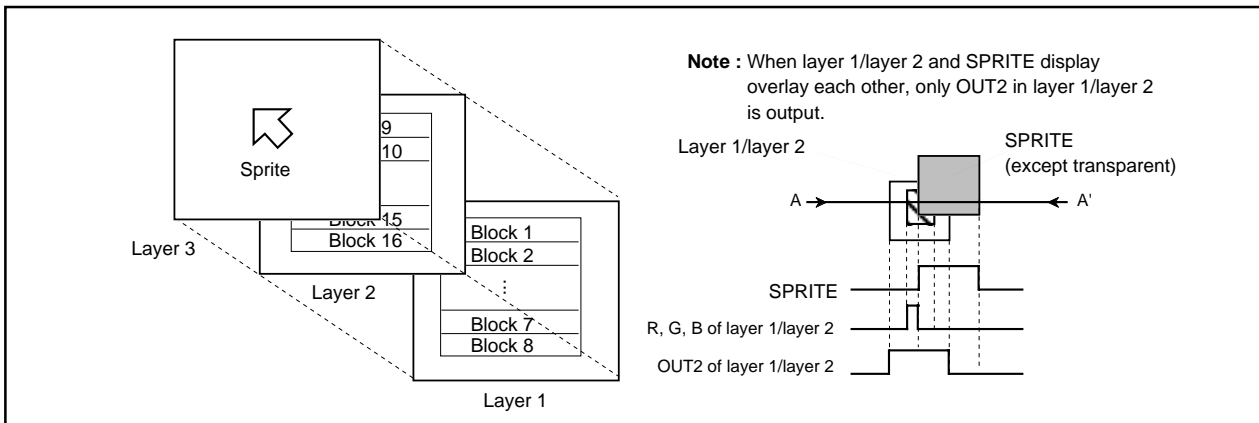
**2:** OSDP mode is always displayed on layer 1. And also, it cannot be overlapped with layer 2's block.

**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"> <li>• Same size as layer 1</li> <li>• 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.</li> </ul>	
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"> <li>• 2Tc X 2H: 2H units</li> <li>• 3Tc X 3H: 3H units</li> </ul>		

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



**Fig 2.16.5 Triple layer OSD**

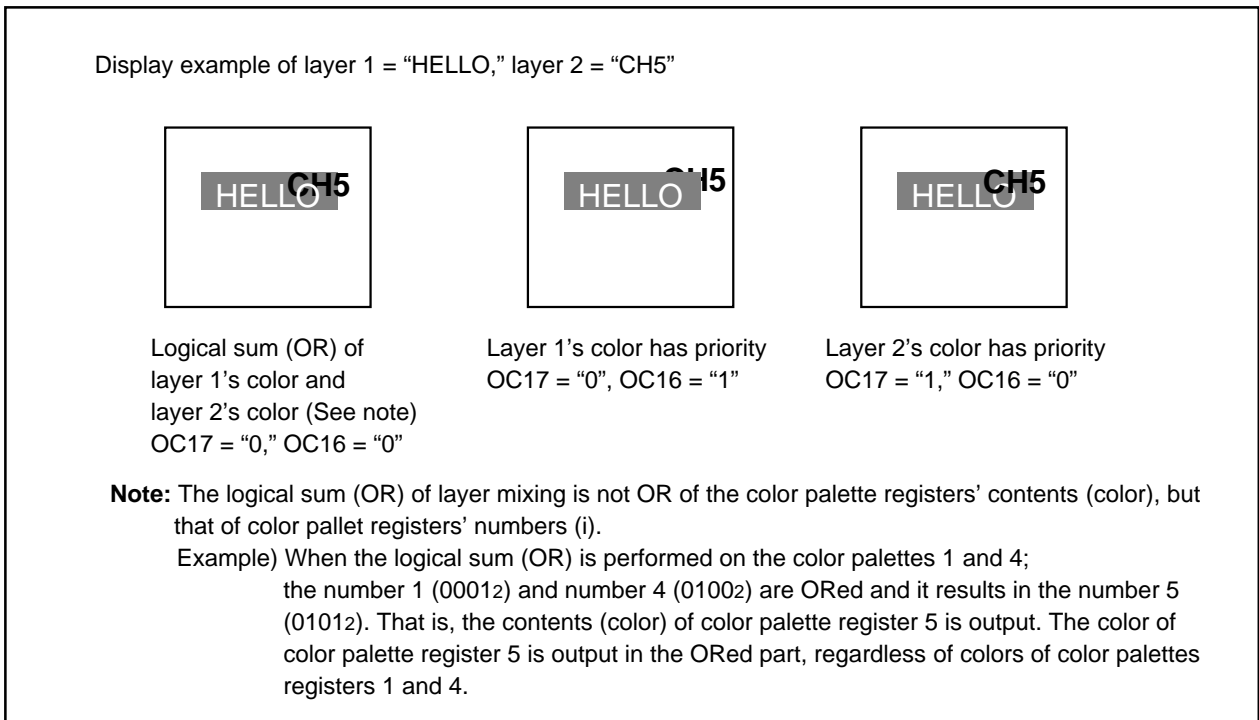


Figure 2.16.6 Display example of triple layer OSD

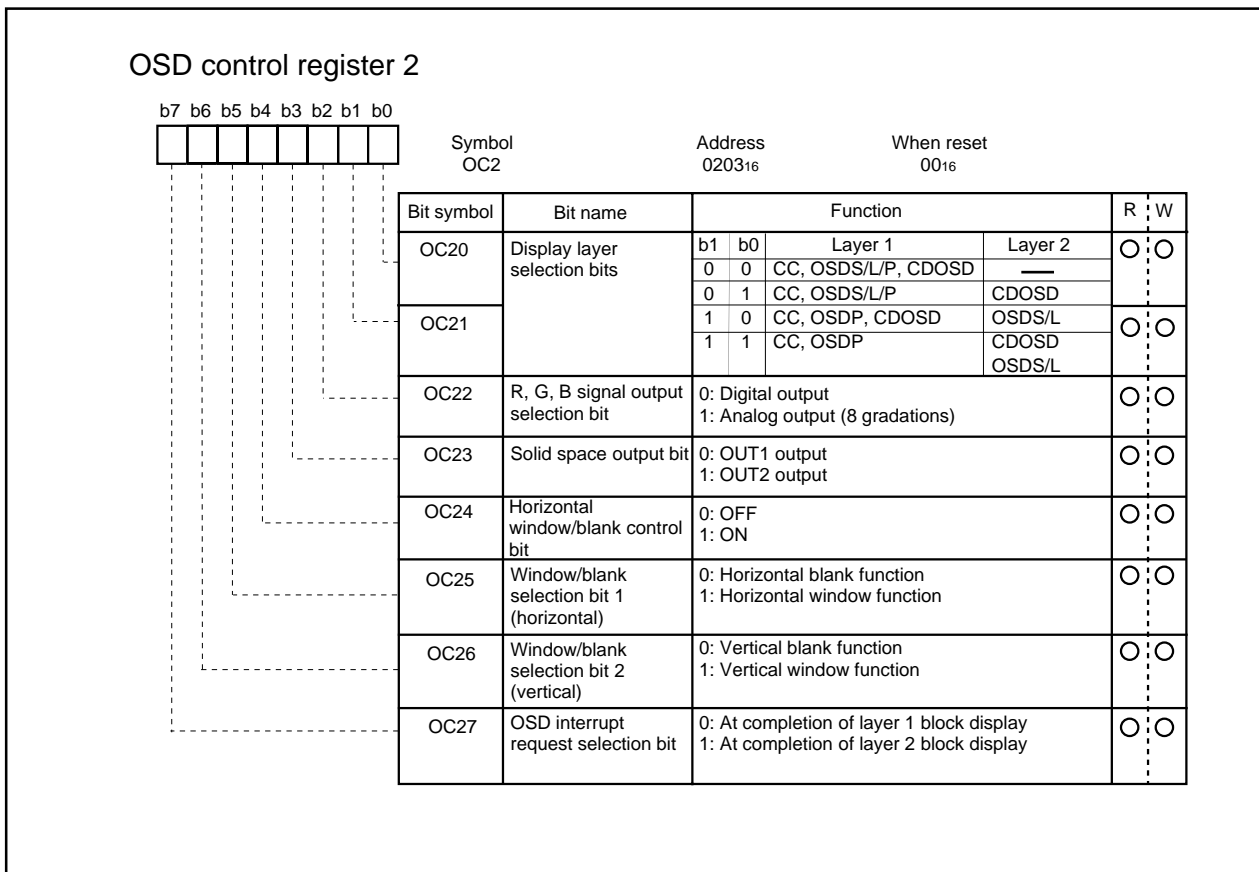


Figure 2.16.7 OSD control register 2



## 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).

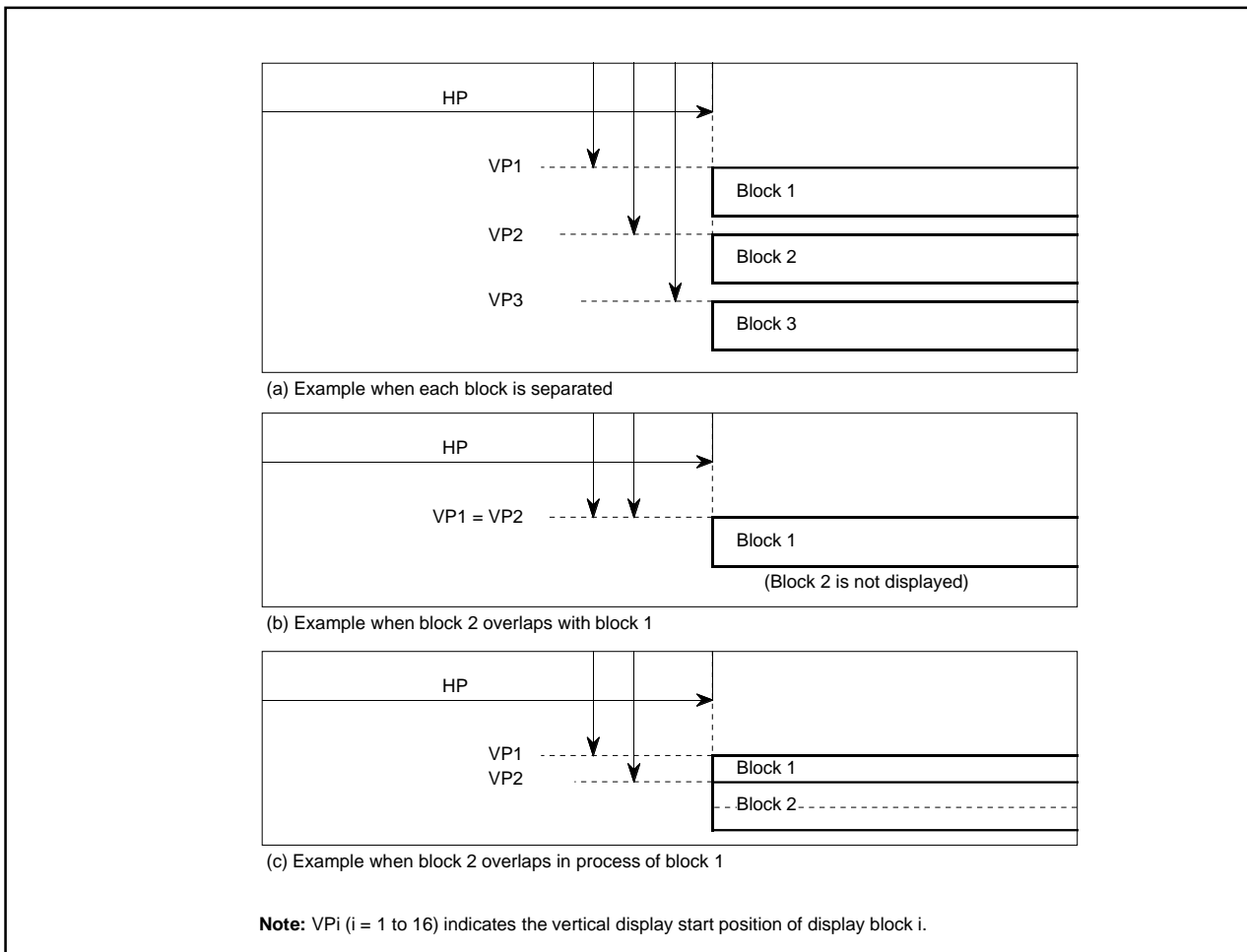
The display position of each block can be set in both horizontal and vertical directions by software.

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 ( $2 \times \text{BCLK}$  cycles or more) for avoiding jitter. The polarity of HSYNC and VSYNC signals can select with the I/O polarity control register (address 0206<sub>16</sub>).

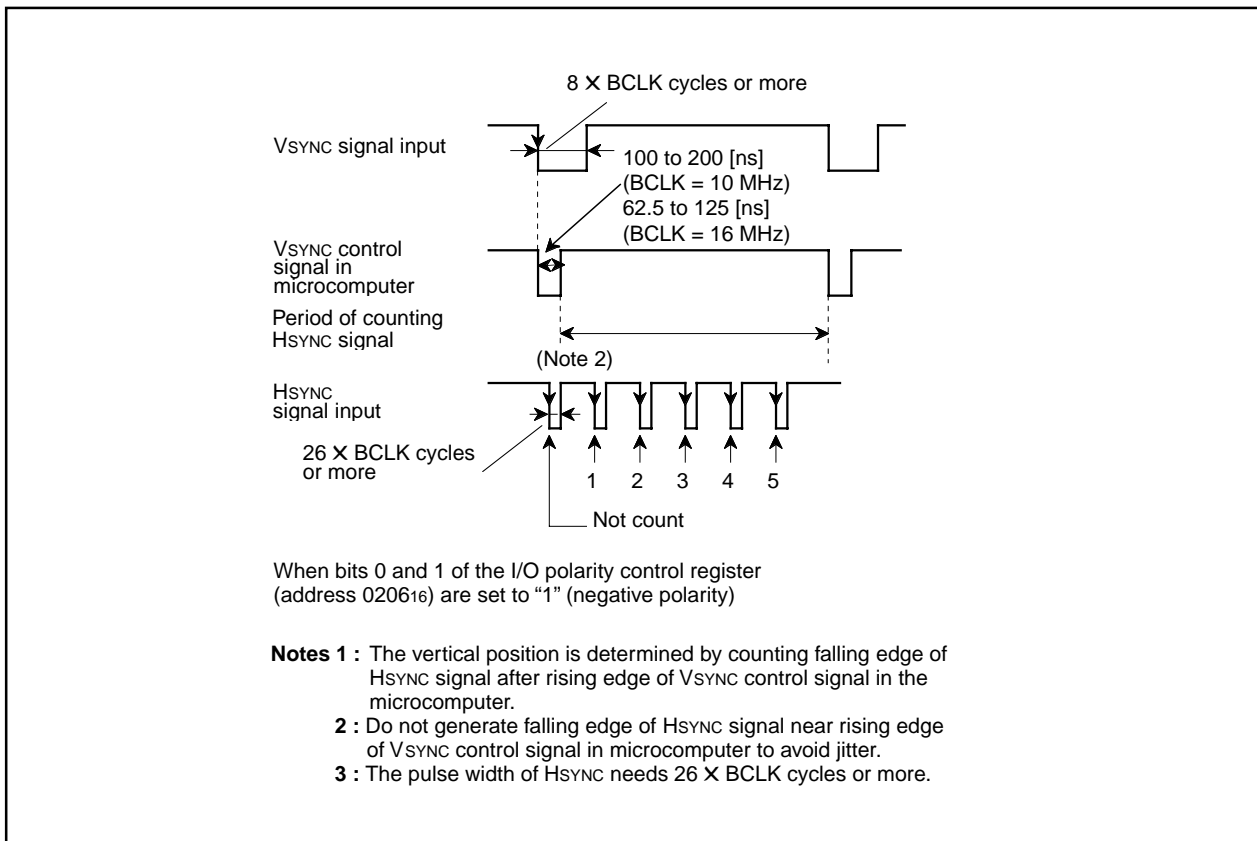
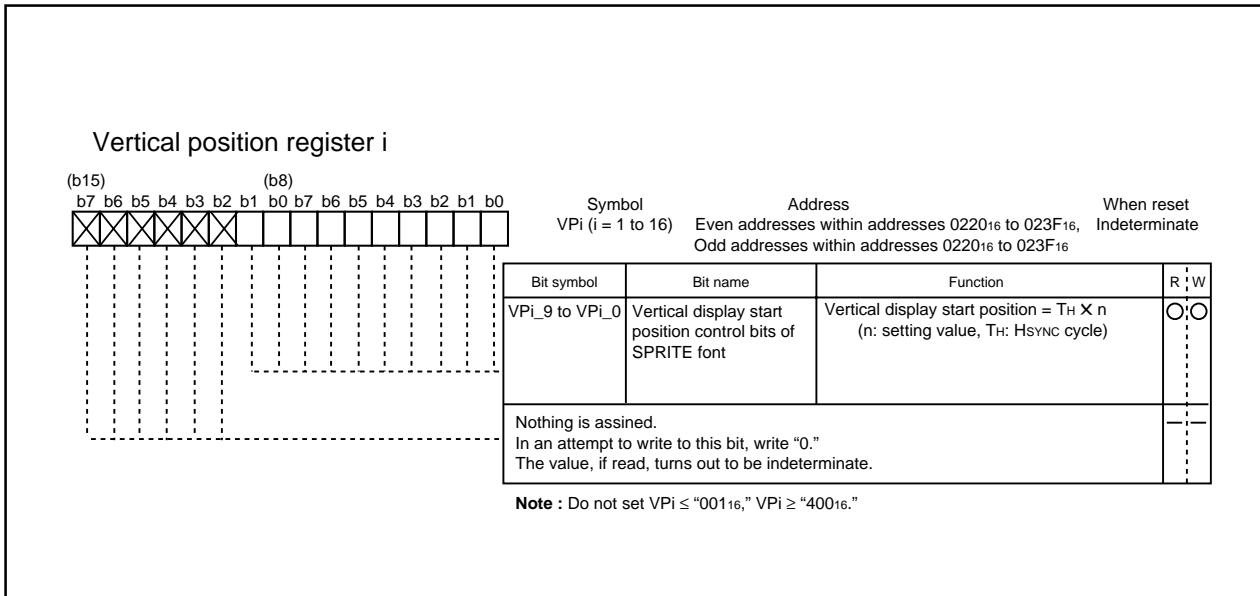


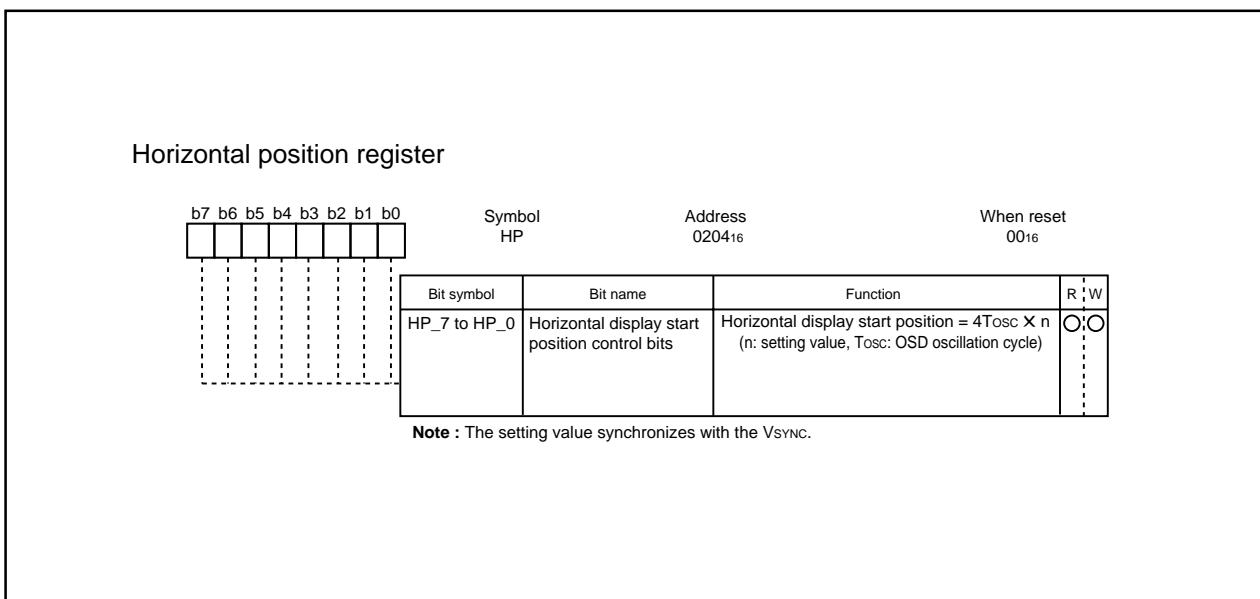
Figure 2.16.9 Supplement explanation for display position

The vertical position for each block can be set in 1024 steps (where each step is 1<sub>TH</sub> (TH: Hsync cycle)) as values “002<sub>16</sub>” to “3FF<sub>16</sub>” in vertical position register i (i = 1 to 16) (addresses 0220<sub>16</sub> to 023F<sub>16</sub>). The vertical position register i is shown in Figure 2.16.10.



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

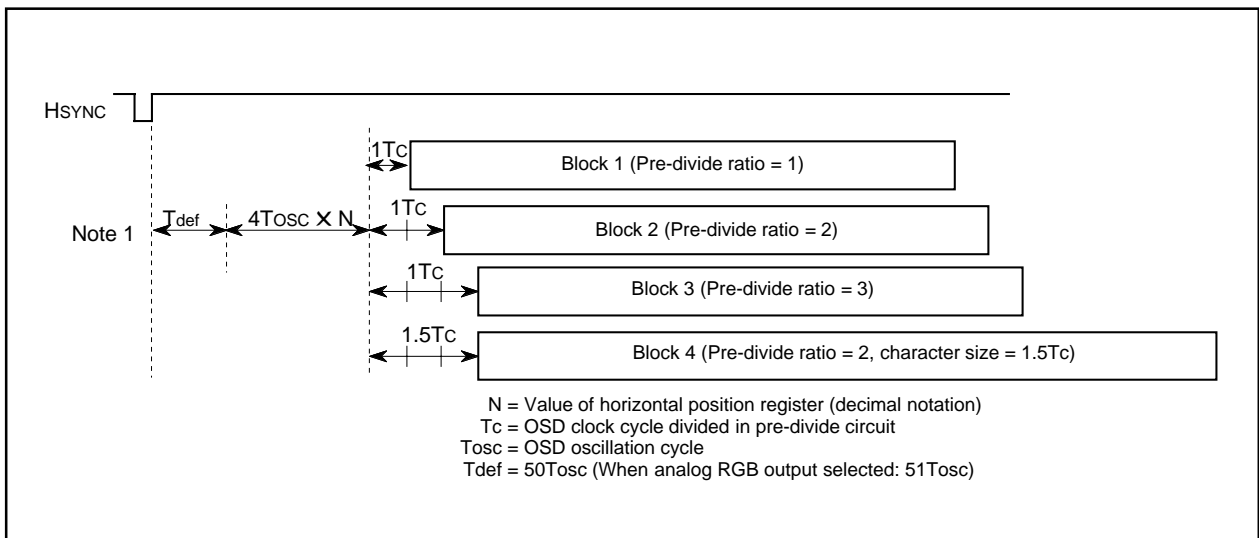
The horizontal position is common to all blocks, and can be set in 256 steps (where 1 step is 4<sub>Tosc</sub>, T<sub>osc</sub> being OSD oscillation cycle) as values “00<sub>16</sub>” to “FF<sub>16</sub>” in bits 0 to 7 of the horizontal position register (address 0204<sub>16</sub>). The horizontal 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 (internally generated 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.

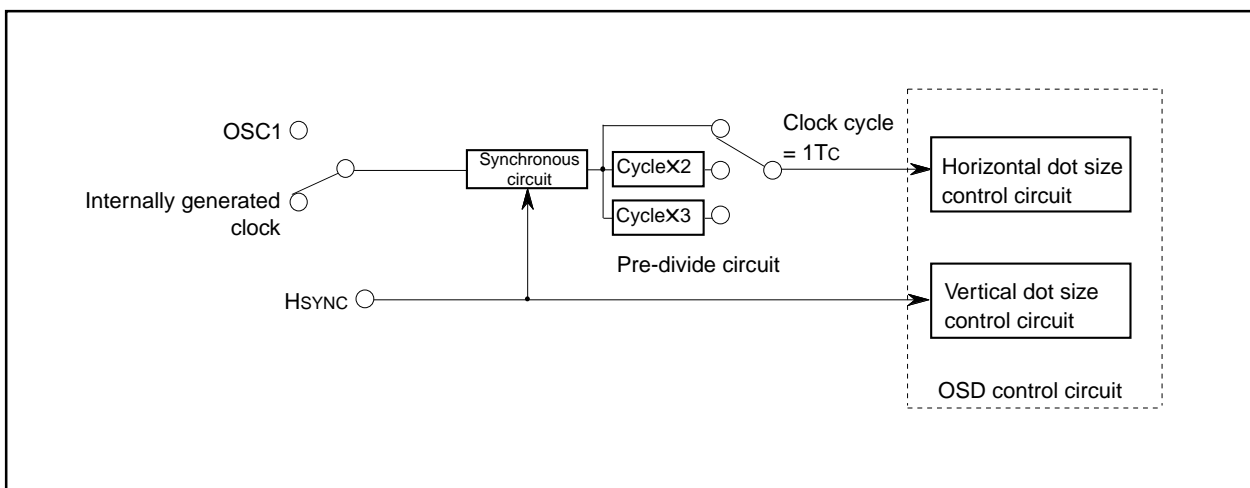


Figure 2.16.13 Block diagram of dot size control circuit

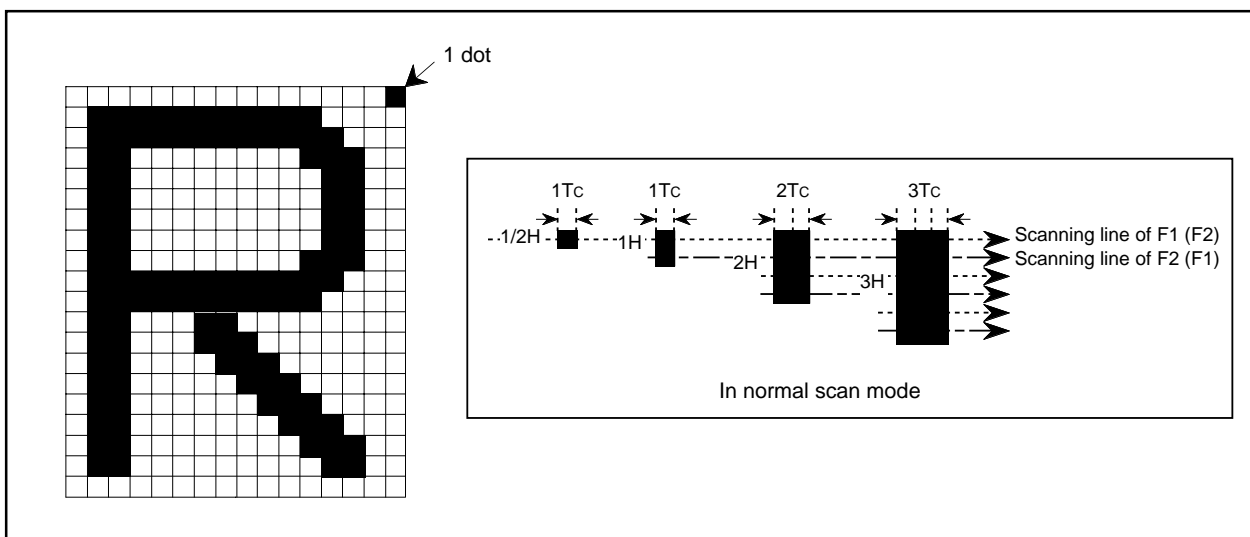


Figure 2.16.14 Definition of dot sizes

### 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.

- Internally generated clock (20 MHz to 40 MHz) output by the internal oscillator
- 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

When the clock control register i (i=1-2) is set to choose an internally generated clock for the OSD clock, use the internal oscillation control register i (i=1-3) to select the oscillation frequency.

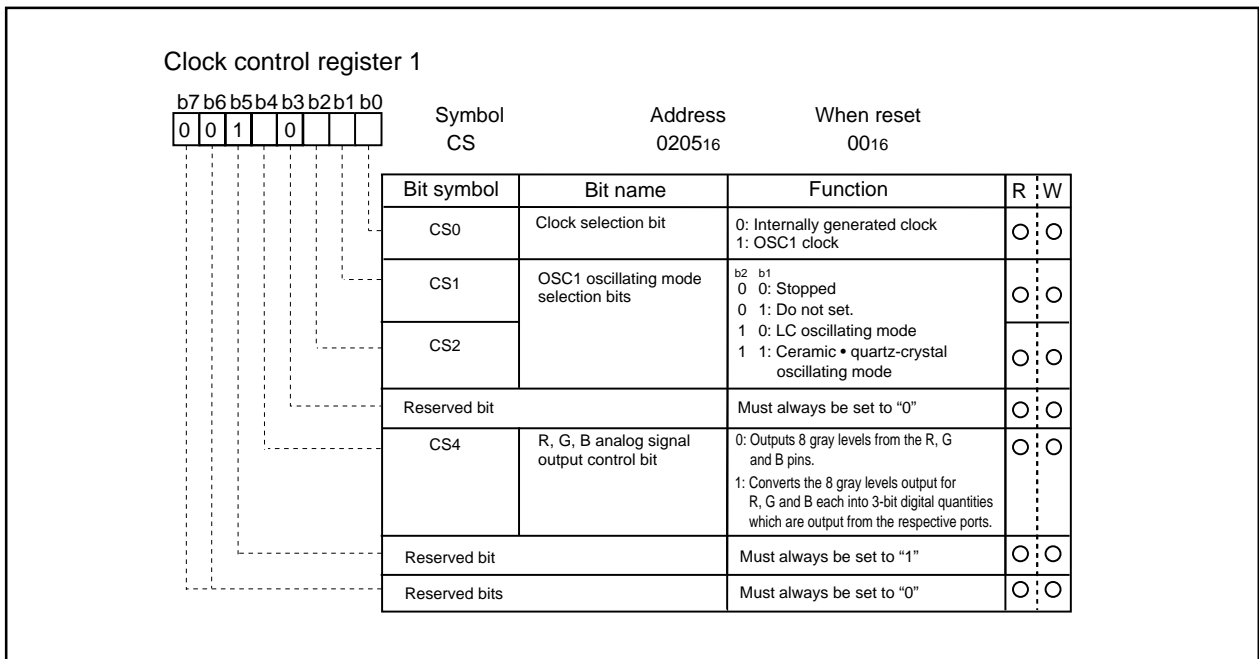


Figure 2.16.15 Clock control register 1

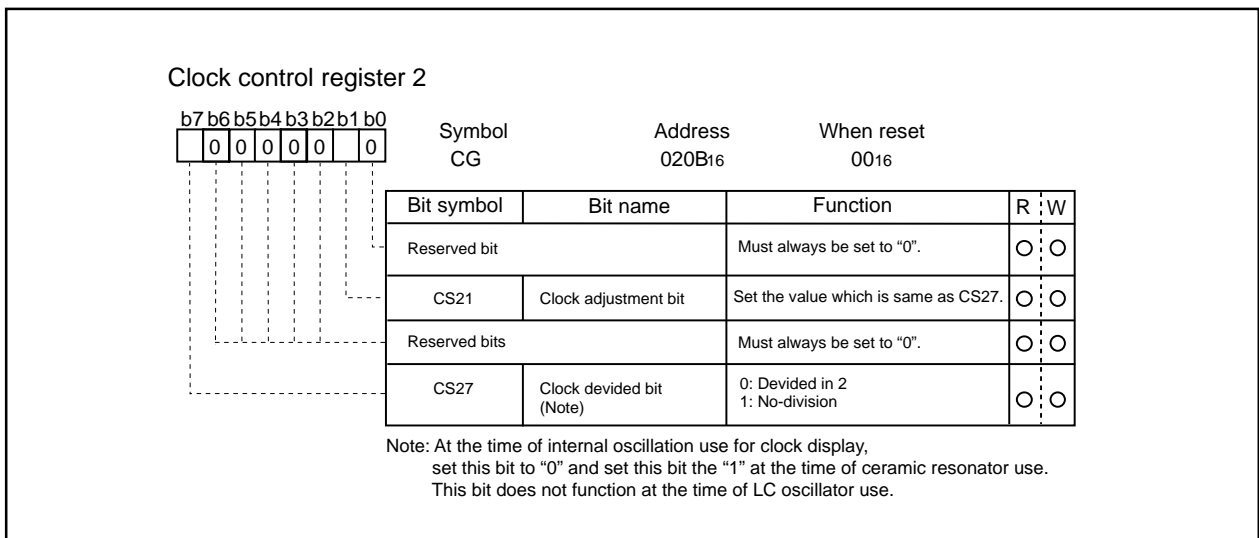


Figure 2.16.16 Clock control register 2

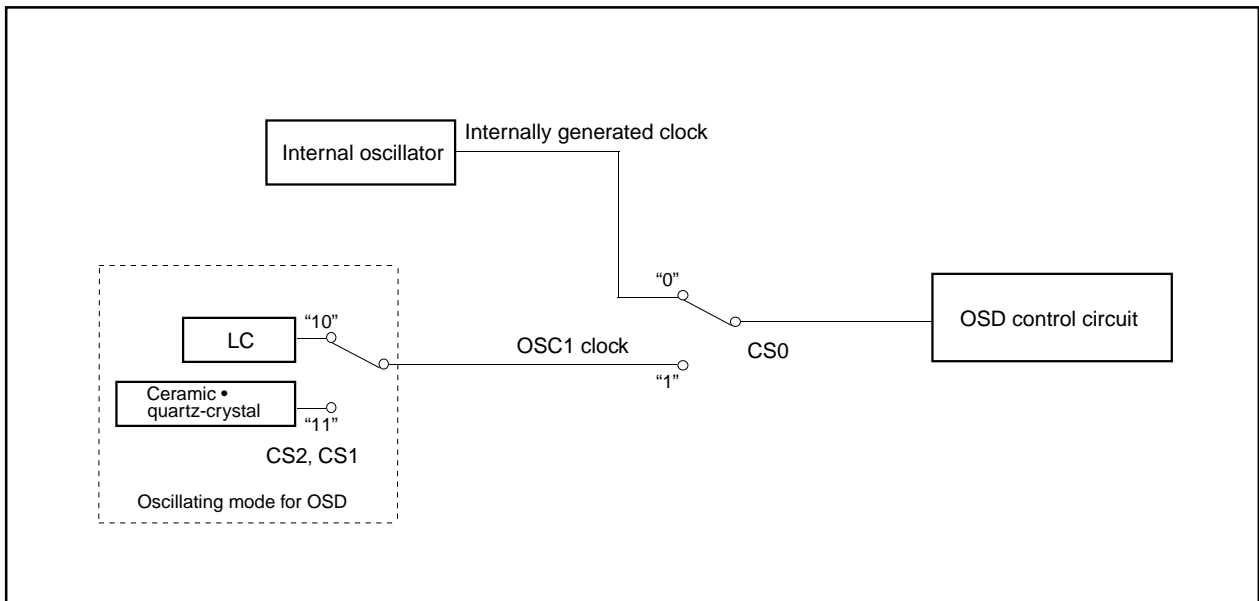
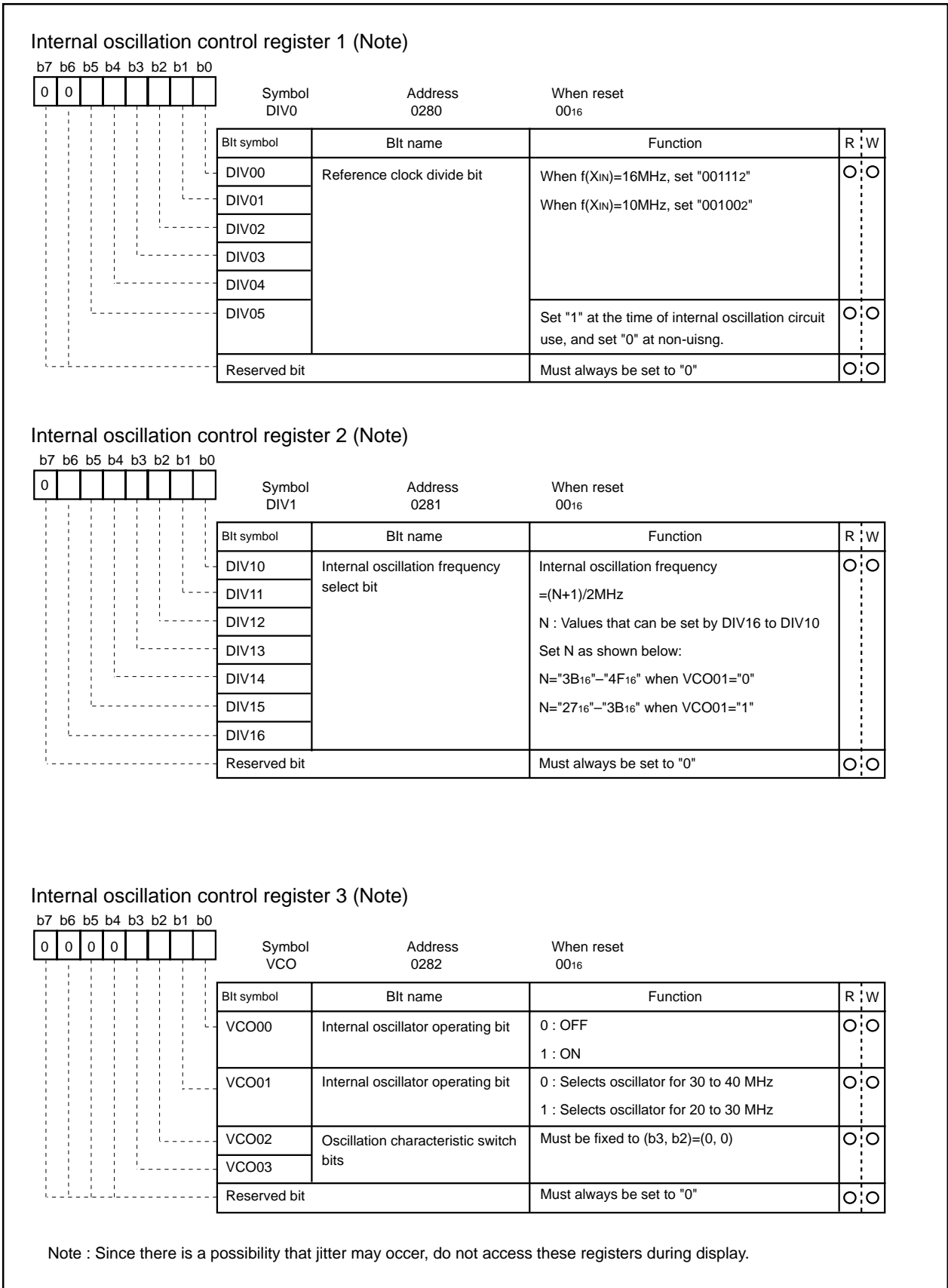


Figure 2.16.17 Block Diagram of OSD selection circuit



**Figure 2.16.18 Internal oscillation control register i (i=1 to 3)**



## 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 2.16.20) 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 0206<sub>16</sub>). A dot line is specified by bit 6 of the I/O polarity control register (refer to Figure 2.16.19).

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.

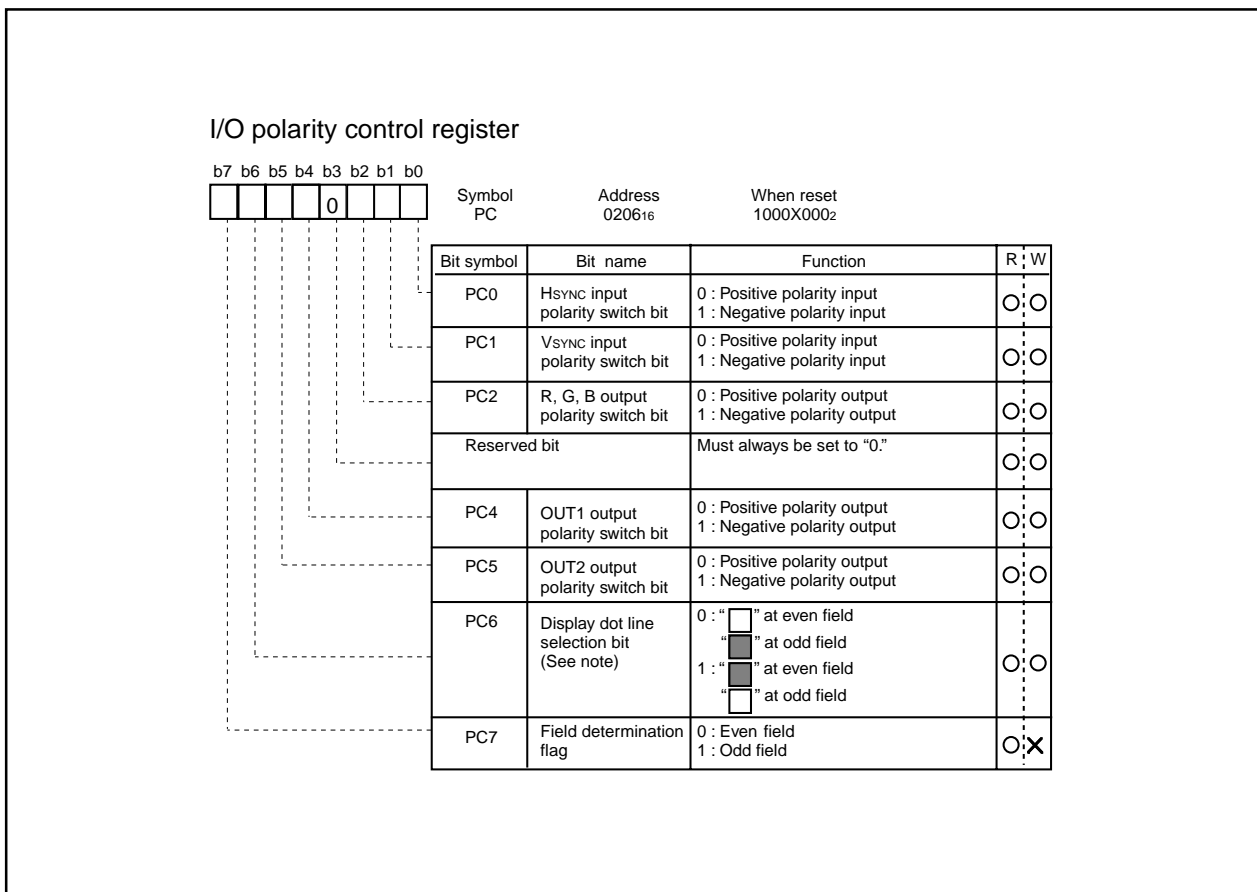
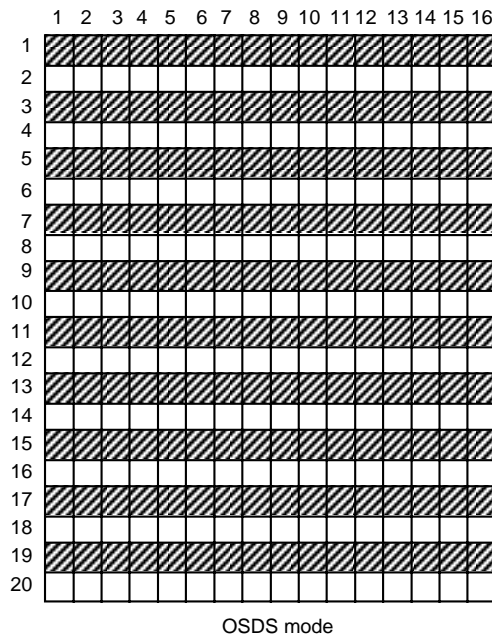
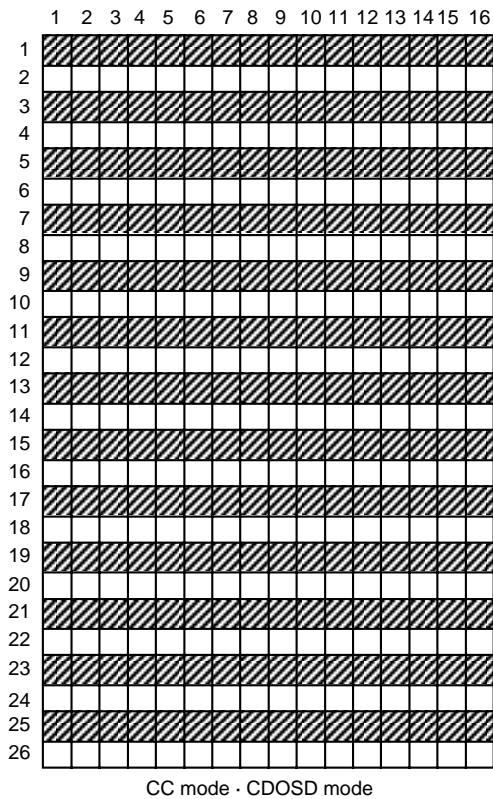


Figure 2.16.19 I/O polarity control register

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  Upper : Vsync signal  Lower : Vsync control signal in microcomputer	<p>(n - 1) field (Odd-numbered)</p> <p>T1</p> <p>0.5 to 0.1 [ms] at f(BCLK) = 10 MHz</p>	Odd	/	/	/
	<p>(n) field (Even-numbered)</p> <p>T2</p>	Even	0 (T2 > T1)	0	Dot line 1 <input type="checkbox"/>
	<p>(n + 1) field (Odd-numbered)</p> <p>T3</p>	Odd	1 (T3 < T2)	0	Dot line 0 <input checked="" type="checkbox"/>
				1	Dot line 1 <input type="checkbox"/>

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



When the display dot line selection bit is "0," the "□" font is displayed at even field, the "▨" font is displayed at odd field. Bit 7 of the I/O polarity control register can be read as the field determination flag : "1" is read at odd field, "0" is read at even field.

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 2.16.20 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 40000<sub>16</sub> to 5FFFF<sub>16</sub>) used to store character dot data and OSD RAM (addresses 0400<sub>16</sub> to 13FF<sub>16</sub>) 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 40000<sub>16</sub> to 5FFFF<sub>16</sub>)

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 0 of the OSD control register 4 for each screen.

The conditions for each OSDL enable/disable mode are shown in Figure 2.16.22.

During OSDL enable mode, character codes 000<sub>16</sub> through 1FF<sub>16</sub> can be used. In this case, the character codes 000<sub>16</sub> through 0FF<sub>16</sub> are turned to 16 X 20-dot fonts, whereas the character codes 100<sub>16</sub> through 1FF<sub>16</sub> are turned to 24 X 32-dot fonts. Of these, however, character codes 0FE<sub>16</sub>, 0FF<sub>16</sub>, 100<sub>16</sub>, and 180<sub>16</sub> cannot be used.

During OSDL disable mode, character codes 000<sub>16</sub> through 2FF<sub>16</sub> can be used. In this case, all characters are turned to 16 X 20-dots. Of these, however, character codes 0FE<sub>16</sub>, 0FF<sub>16</sub>, 100<sub>16</sub>, 180<sub>16</sub>, 200<sub>16</sub>, and 0FE<sub>16</sub> cannot be used.

CD codes 00<sub>16</sub> through 7F<sub>16</sub> can be used. In this case, all characters are turned to 16 X 26-dot fonts. Of these, however, CD codes 3F<sub>16</sub> and 40<sub>16</sub> cannot be used.

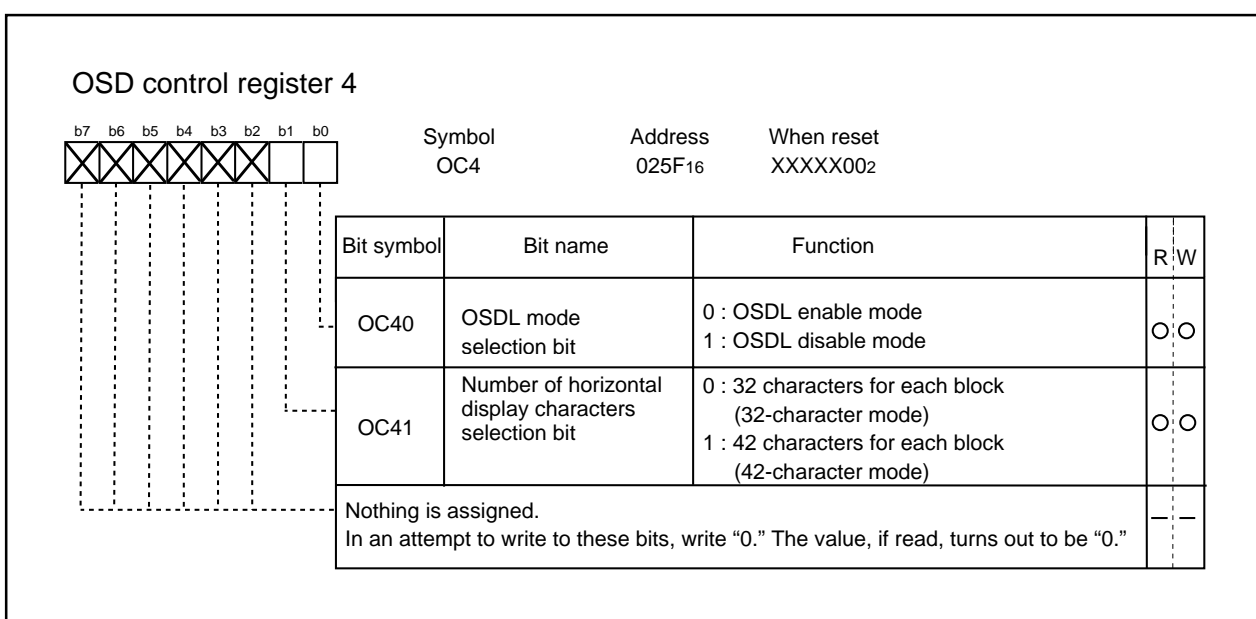
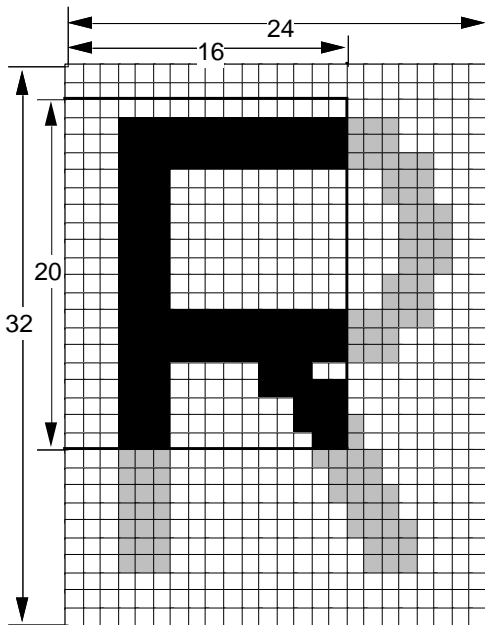


Figure 2.16.21 OSD control register 4

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 <sub>16</sub> to 0FF <sub>16</sub>	S	Used	Used	Not used (See note 3)	S	Used	Used	Display OFF
	100 <sub>16</sub> to 1FF <sub>16</sub>	L	Used (See note 1)	Used (See note 1)	Used	S	Used	Used	Display OFF
	200 <sub>16</sub> to 27F <sub>16</sub>	Not used (See note 3)	Not used (See note 3)			S	Not used	Used	Display OFF
	280 <sub>16</sub> to 2FF <sub>16</sub>					S	Not used (See note 3)	Used (No border ) (See note 2)	Display OFF
	300 <sub>16</sub> to 3FF <sub>16</sub>					S	Not used	Display OFF	



- Notes 1:** Part of 24 X 32 font is displayed.
- 2:** In OSDL disable mode, character codes "280<sub>16</sub>" to "2FF<sub>16</sub>" are used in OSDS/P mode (no border).
- 3:** As setting this make output of font data indeterminate, do not use. However, "3FE<sub>16</sub>" and "3FF<sub>16</sub>" can be used as character codes of blank font output in OSDP mode.

Figure 2.16.22 Conditions for each OSDL enable/disable mode

**(2) OSD RAM (OSD RAM for character, addresses 0400<sub>16</sub> to 0EFF<sub>16</sub>)**

The OSD RAM for character is allocated at addresses 0400<sub>16</sub> to 0EFF<sub>16</sub>, 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 0400<sub>16</sub>, write color code 1 at 0401<sub>16</sub>, and write color code 2 at 0480<sub>16</sub>. The structure of the OSD RAM is shown in Figure 2.16.23.

**Note :** For blocks of the following dot sizes, the 3<sup>rd</sup> (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 3<sup>rd</sup> 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 28<sup>th</sup>'s character area in 42-character mode, set ordinarily.

- In OSDS mode

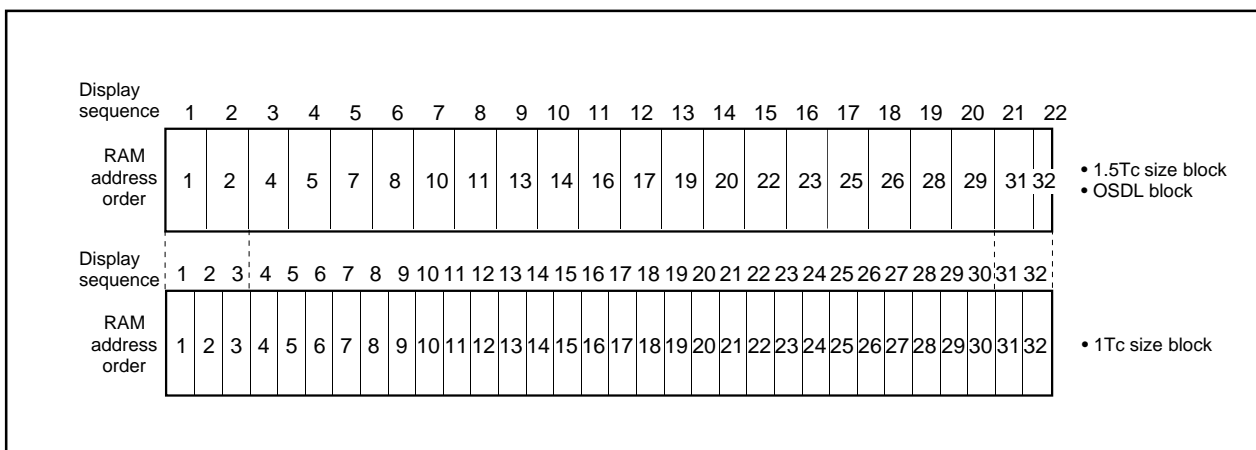
The character is not displayed, and only the left 1/3 part of the 22<sup>nd</sup> character background is displayed in the 22<sup>nd</sup>'s character area. When not displaying this background, set transparent for character background color.

- In OSDL mode

Set a blank character or a character of transparent color to the 22<sup>nd</sup> 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 22<sup>nd</sup>'s character area (left 1/3 part).



**Figure 2.16.23 RAM data for 3<sup>rd</sup> 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 <sub>16</sub>	0401 <sub>16</sub>	0480 <sub>16</sub>
	2nd character	0402 <sub>16</sub>	0403 <sub>16</sub>	0482 <sub>16</sub>
	⋮	⋮	⋮	⋮
	32nd character	043E <sub>16</sub>	043F <sub>16</sub>	04BE <sub>16</sub>
Block 2	1st character	0440 <sub>16</sub>	0441 <sub>16</sub>	04C0 <sub>16</sub>
	2nd character	0442 <sub>16</sub>	0443 <sub>16</sub>	04C2 <sub>16</sub>
	⋮	⋮	⋮	⋮
	32nd character	047E <sub>16</sub>	047F <sub>16</sub>	04FE <sub>16</sub>
Block 3	1st character	0500 <sub>16</sub>	0501 <sub>16</sub>	0580 <sub>16</sub>
	2nd character	0502 <sub>16</sub>	0503 <sub>16</sub>	0582 <sub>16</sub>
	⋮	⋮	⋮	⋮
	32nd character	053E <sub>16</sub>	053F <sub>16</sub>	05BE <sub>16</sub>
Block 4	1st character	0540 <sub>16</sub>	0541 <sub>16</sub>	05C0 <sub>16</sub>
	2nd character	0542 <sub>16</sub>	0543 <sub>16</sub>	05C2 <sub>16</sub>
	⋮	⋮	⋮	⋮
	32nd character	057E <sub>16</sub>	057F <sub>16</sub>	05FE <sub>16</sub>
Block 5	1st character	0600 <sub>16</sub>	0601 <sub>16</sub>	0680 <sub>16</sub>
	2nd character	0602 <sub>16</sub>	0603 <sub>16</sub>	0682 <sub>16</sub>
	⋮	⋮	⋮	⋮
	32nd character	063E <sub>16</sub>	063F <sub>16</sub>	06BE <sub>16</sub>
Block 6	1st character	0640 <sub>16</sub>	0641 <sub>16</sub>	06C0 <sub>16</sub>
	2nd character	0642 <sub>16</sub>	0643 <sub>16</sub>	06C2 <sub>16</sub>
	⋮	⋮	⋮	⋮
	32nd character	067E <sub>16</sub>	067F <sub>16</sub>	06FE <sub>16</sub>
Block 7	1st character	0700 <sub>16</sub>	0701 <sub>16</sub>	0780 <sub>16</sub>
	2nd character	0702 <sub>16</sub>	0703 <sub>16</sub>	0782 <sub>16</sub>
	⋮	⋮	⋮	⋮
	32nd character	073E <sub>16</sub>	073F <sub>16</sub>	07BE <sub>16</sub>
Block 8	1st character	0740 <sub>16</sub>	0741 <sub>16</sub>	07C0 <sub>16</sub>
	2nd character	0742 <sub>16</sub>	0743 <sub>16</sub>	07C2 <sub>16</sub>
	⋮	⋮	⋮	⋮
	32nd character	077E <sub>16</sub>	077F <sub>16</sub>	07FE <sub>16</sub>
Block 9	1st character	0800 <sub>16</sub>	0801 <sub>16</sub>	0880 <sub>16</sub>
	2nd character	0802 <sub>16</sub>	0803 <sub>16</sub>	0882 <sub>16</sub>
	⋮	⋮	⋮	⋮
	32nd character	083E <sub>16</sub>	083F <sub>16</sub>	08BE <sub>16</sub>
Block 10	1st character	0840 <sub>16</sub>	0841 <sub>16</sub>	08C0 <sub>16</sub>
	2nd character	0842 <sub>16</sub>	0843 <sub>16</sub>	08C2 <sub>16</sub>
	⋮	⋮	⋮	⋮
	32nd character	087E <sub>16</sub>	087F <sub>16</sub>	08FE <sub>16</sub>

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

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



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

**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 <sub>16</sub>	0D61 <sub>16</sub>	0DE0 <sub>16</sub>
	34th character	0D62 <sub>16</sub>	0D63 <sub>16</sub>	0DE2 <sub>16</sub>
	⋮	⋮	⋮	⋮
	39th character	0D6C <sub>16</sub>	0D6D <sub>16</sub>	0DEC <sub>16</sub>
	40th character	0D6E <sub>16</sub>	0D6F <sub>16</sub>	0DEE <sub>16</sub>
	41st character	0E70 <sub>16</sub>	0E71 <sub>16</sub>	0EF0 <sub>16</sub>
Block 16	42nd character	0E72 <sub>16</sub>	0E73 <sub>16</sub>	0EF2 <sub>16</sub>
	33rd character	0D70 <sub>16</sub>	0D71 <sub>16</sub>	0DF0 <sub>16</sub>
	34th character	0D72 <sub>16</sub>	0D73 <sub>16</sub>	0DF2 <sub>16</sub>
	⋮	⋮	⋮	⋮
	39th character	0D7C <sub>16</sub>	0D7D <sub>16</sub>	0DFC <sub>16</sub>
	40th character	0D7E <sub>16</sub>	0D7F <sub>16</sub>	0DFE <sub>16</sub>
Block 16	41st character	0E78 <sub>16</sub>	0E79 <sub>16</sub>	0EF8 <sub>16</sub>
	42nd character	0E7A <sub>16</sub>	0E7B <sub>16</sub>	0EFA <sub>16</sub>

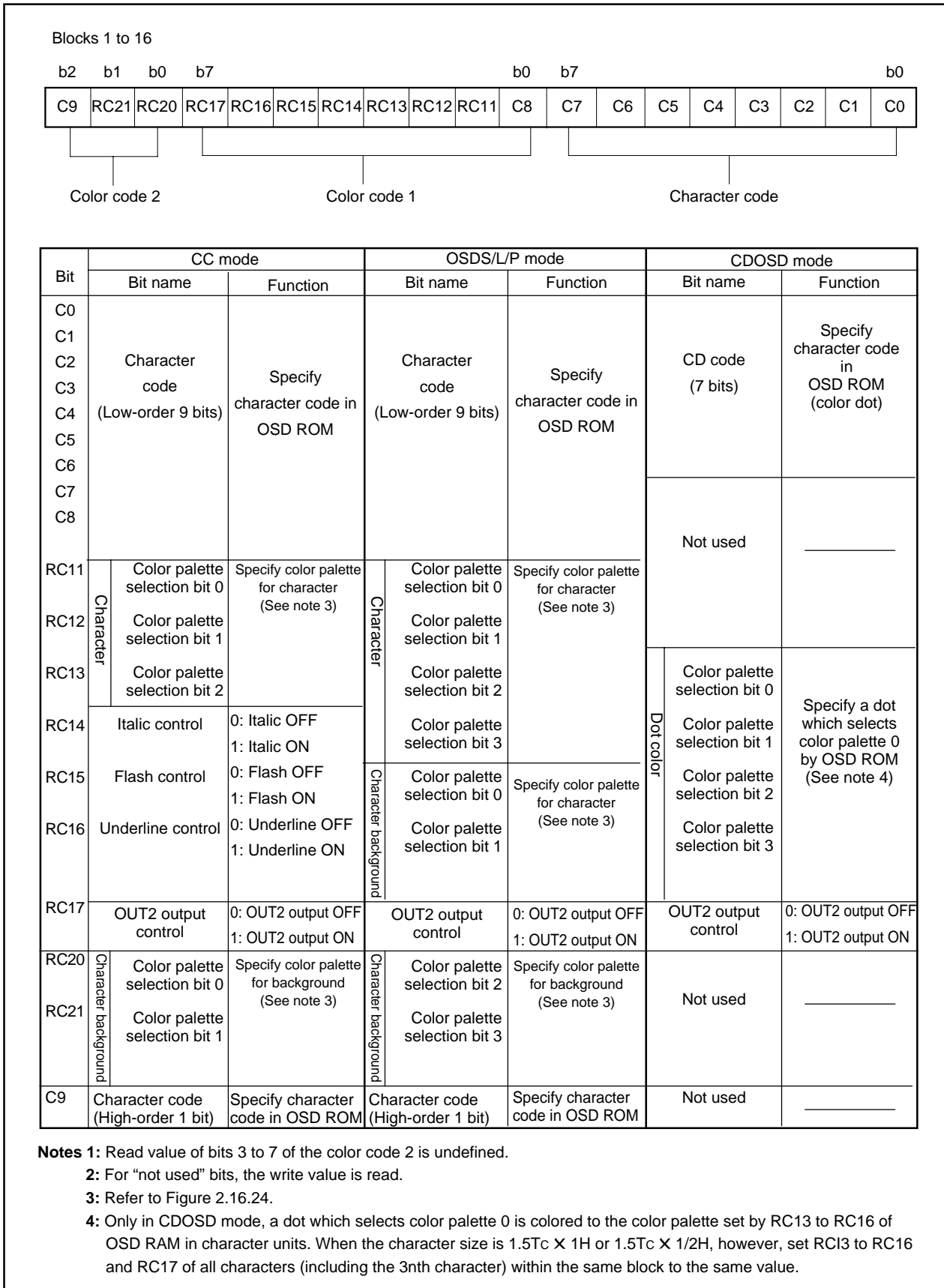


Figure 2.16.24 Structure of OSD RAM

**(3) OSD RAM (OSD RAM for SPRITE, addresses 1000<sub>16</sub> to 13E7<sub>16</sub>)**

The OSD RAM for SPRITE fonts 1 and 2, consisting of 4 planes for each font, is assigned to addresses 1000<sub>16</sub> to 13E7<sub>16</sub>. Each plane corresponds to each color palette selection bit and the color palette of each dot is determined from among 16 kinds.

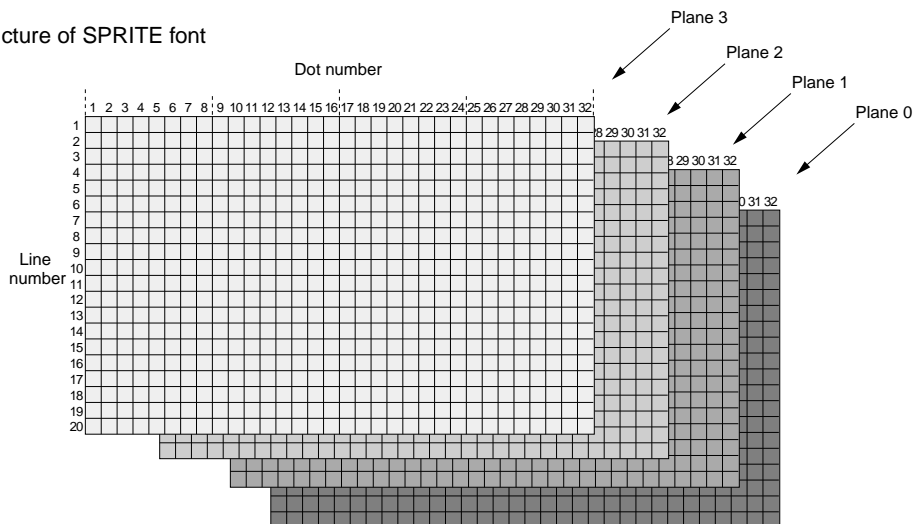
**Table 2.16.8 OSD RAM address (SPRITE font 1)**

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 <sub>16</sub>	10C1 <sub>16</sub>	11C0 <sub>16</sub>	11C1 <sub>16</sub>	1080 <sub>16</sub>	1081 <sub>16</sub>	1180 <sub>16</sub>	1181 <sub>16</sub>	1040 <sub>16</sub>	1041 <sub>16</sub>	1140 <sub>16</sub>	1141 <sub>16</sub>	1000 <sub>16</sub>	1001 <sub>16</sub>	1100 <sub>16</sub>	1101 <sub>16</sub>
Line 2	10C2 <sub>16</sub>	10C3 <sub>16</sub>	11C2 <sub>16</sub>	11C3 <sub>16</sub>	1082 <sub>16</sub>	1083 <sub>16</sub>	1182 <sub>16</sub>	1183 <sub>16</sub>	1042 <sub>16</sub>	1043 <sub>16</sub>	1142 <sub>16</sub>	1143 <sub>16</sub>	1002 <sub>16</sub>	1003 <sub>16</sub>	1102 <sub>16</sub>	1103 <sub>16</sub>
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
Line 19	10E4 <sub>16</sub>	10E5 <sub>16</sub>	11E4 <sub>16</sub>	11E5 <sub>16</sub>	10A4 <sub>16</sub>	10A5 <sub>16</sub>	11A4 <sub>16</sub>	11A5 <sub>16</sub>	1064 <sub>16</sub>	1065 <sub>16</sub>	1164 <sub>16</sub>	1165 <sub>16</sub>	1024 <sub>16</sub>	1025 <sub>16</sub>	1124 <sub>16</sub>	1125 <sub>16</sub>
Line 20	10E6 <sub>16</sub>	10E7 <sub>16</sub>	11E6 <sub>16</sub>	11E7 <sub>16</sub>	10A6 <sub>16</sub>	10A7 <sub>16</sub>	11A6 <sub>16</sub>	11A7 <sub>16</sub>	1066 <sub>16</sub>	1067 <sub>16</sub>	1166 <sub>16</sub>	1167 <sub>16</sub>	1026 <sub>16</sub>	1027 <sub>16</sub>	1126 <sub>16</sub>	1127 <sub>16</sub>

**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 <sub>16</sub>	12C1 <sub>16</sub>	13C0 <sub>16</sub>	13C1 <sub>16</sub>	1280 <sub>16</sub>	1281 <sub>16</sub>	1380 <sub>16</sub>	1381 <sub>16</sub>	1240 <sub>16</sub>	1241 <sub>16</sub>	1340 <sub>16</sub>	1341 <sub>16</sub>	1200 <sub>16</sub>	1201 <sub>16</sub>	1300 <sub>16</sub>	1301 <sub>16</sub>
Line 2	12C2 <sub>16</sub>	12C3 <sub>16</sub>	13C2 <sub>16</sub>	13C3 <sub>16</sub>	1282 <sub>16</sub>	1283 <sub>16</sub>	1382 <sub>16</sub>	1383 <sub>16</sub>	1242 <sub>16</sub>	1243 <sub>16</sub>	1342 <sub>16</sub>	1343 <sub>16</sub>	1202 <sub>16</sub>	1203 <sub>16</sub>	1302 <sub>16</sub>	1303 <sub>16</sub>
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
Line 19	12E4 <sub>16</sub>	12E5 <sub>16</sub>	13E4 <sub>16</sub>	13E5 <sub>16</sub>	12A4 <sub>16</sub>	12A5 <sub>16</sub>	13A4 <sub>16</sub>	13A5 <sub>16</sub>	1264 <sub>16</sub>	1265 <sub>16</sub>	1364 <sub>16</sub>	1365 <sub>16</sub>	1224 <sub>16</sub>	1225 <sub>16</sub>	1324 <sub>16</sub>	1325 <sub>16</sub>
Line 20	12E6 <sub>16</sub>	12E7 <sub>16</sub>	13E6 <sub>16</sub>	13E7 <sub>16</sub>	12A6 <sub>16</sub>	12A7 <sub>16</sub>	13A6 <sub>16</sub>	13A7 <sub>16</sub>	1266 <sub>16</sub>	1267 <sub>16</sub>	1366 <sub>16</sub>	1367 <sub>16</sub>	1226 <sub>16</sub>	1227 <sub>16</sub>	1326 <sub>16</sub>	1327 <sub>16</sub>

Dot structure of SPRITE font



### 2.16.7 Character Color

As shown in Figure 2.16.25, 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 0207<sub>16</sub>). 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 2.16.25).
- 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 2.16.27).

### 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 0207<sub>16</sub>). 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.

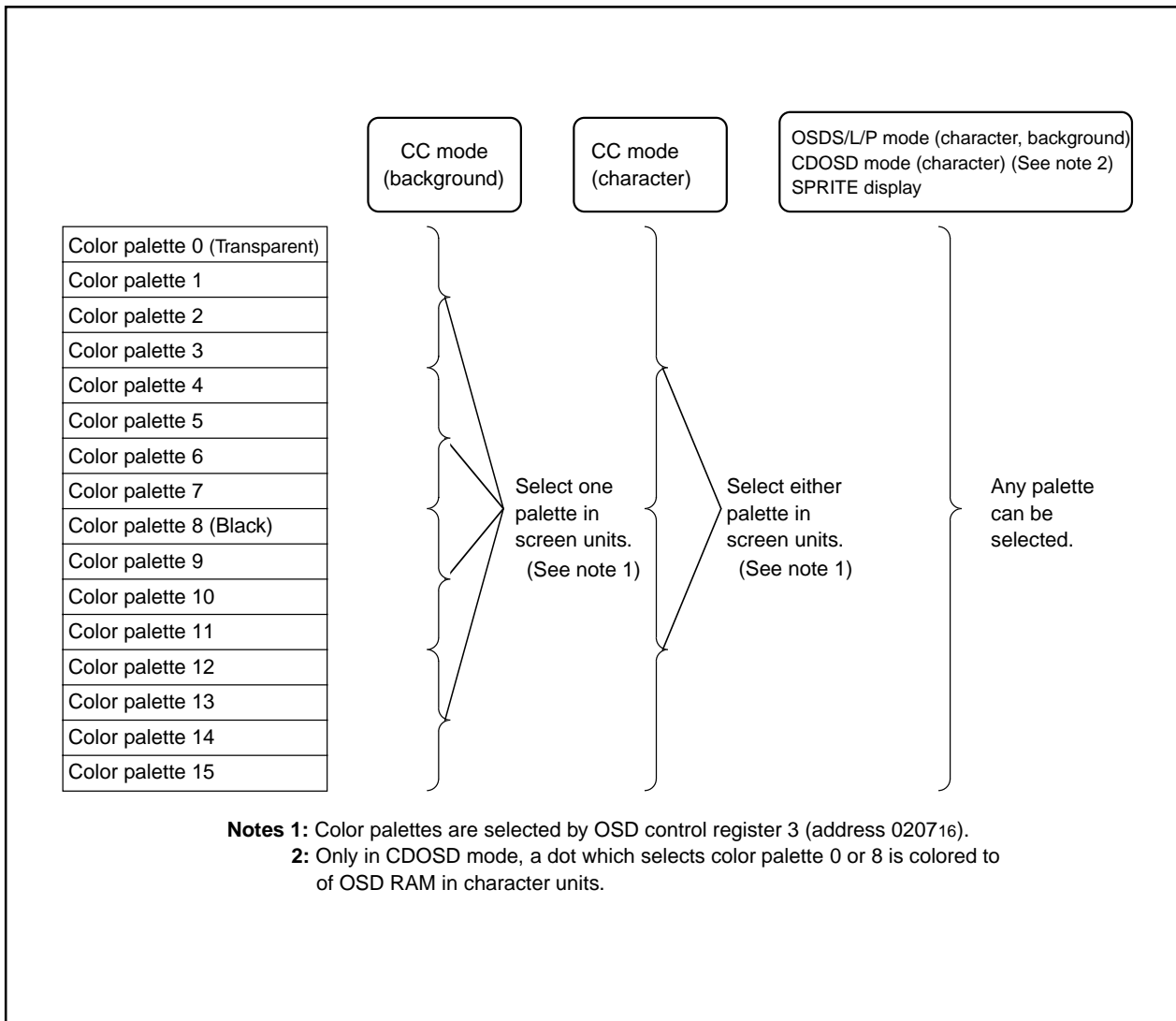


Figure 2.16.25 Color palette selection

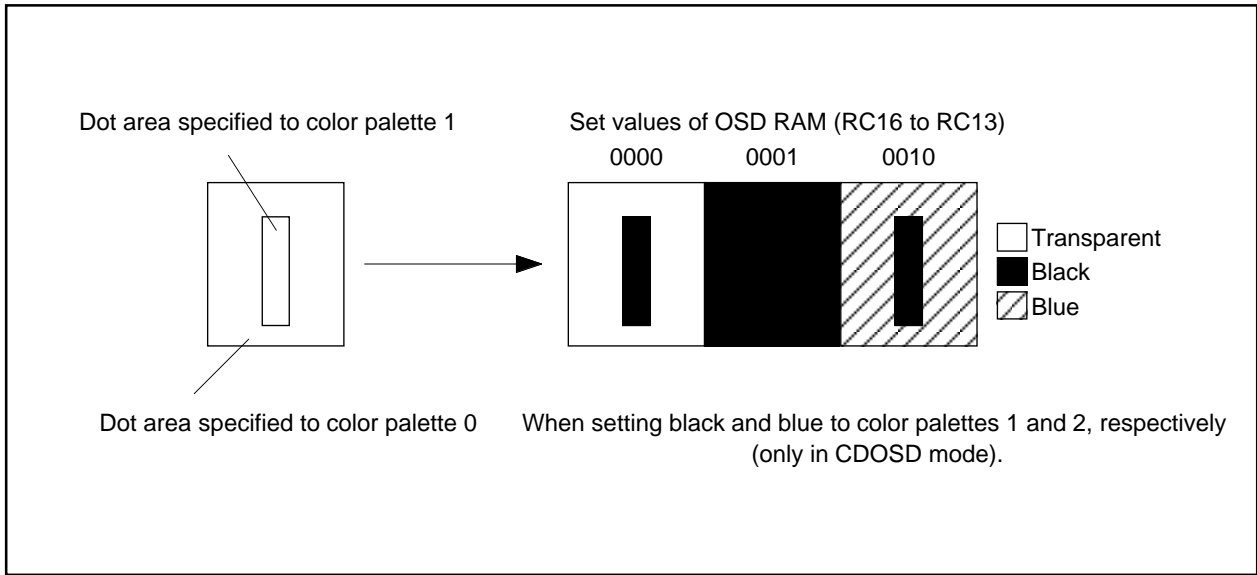


Figure 2.16.26 Set of color palette 0 or 8 in CDOSD mode

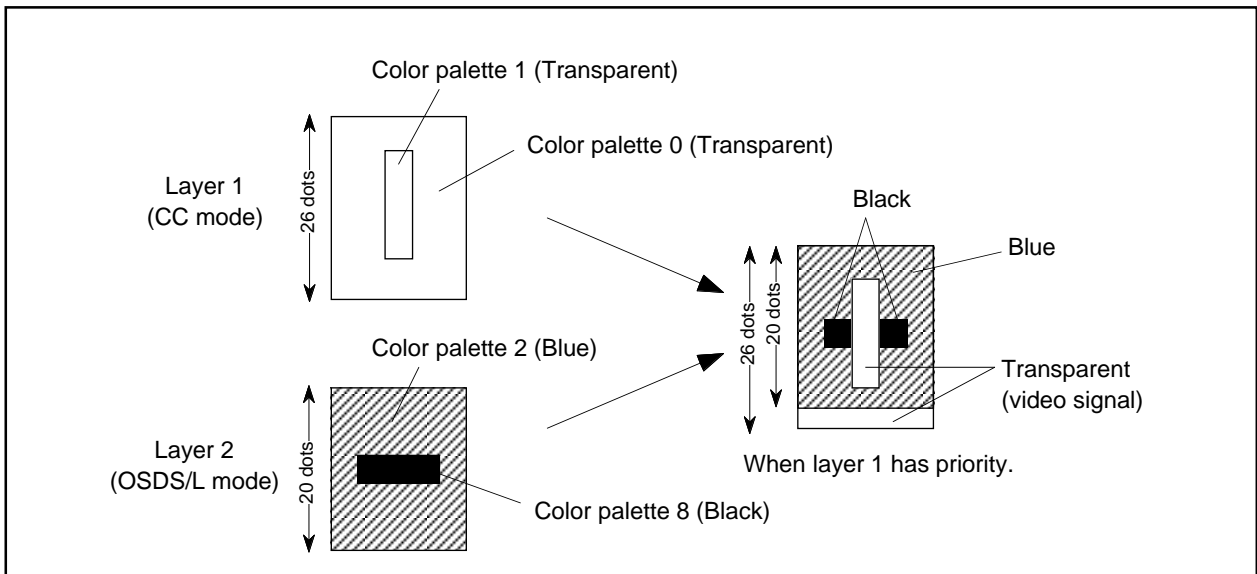
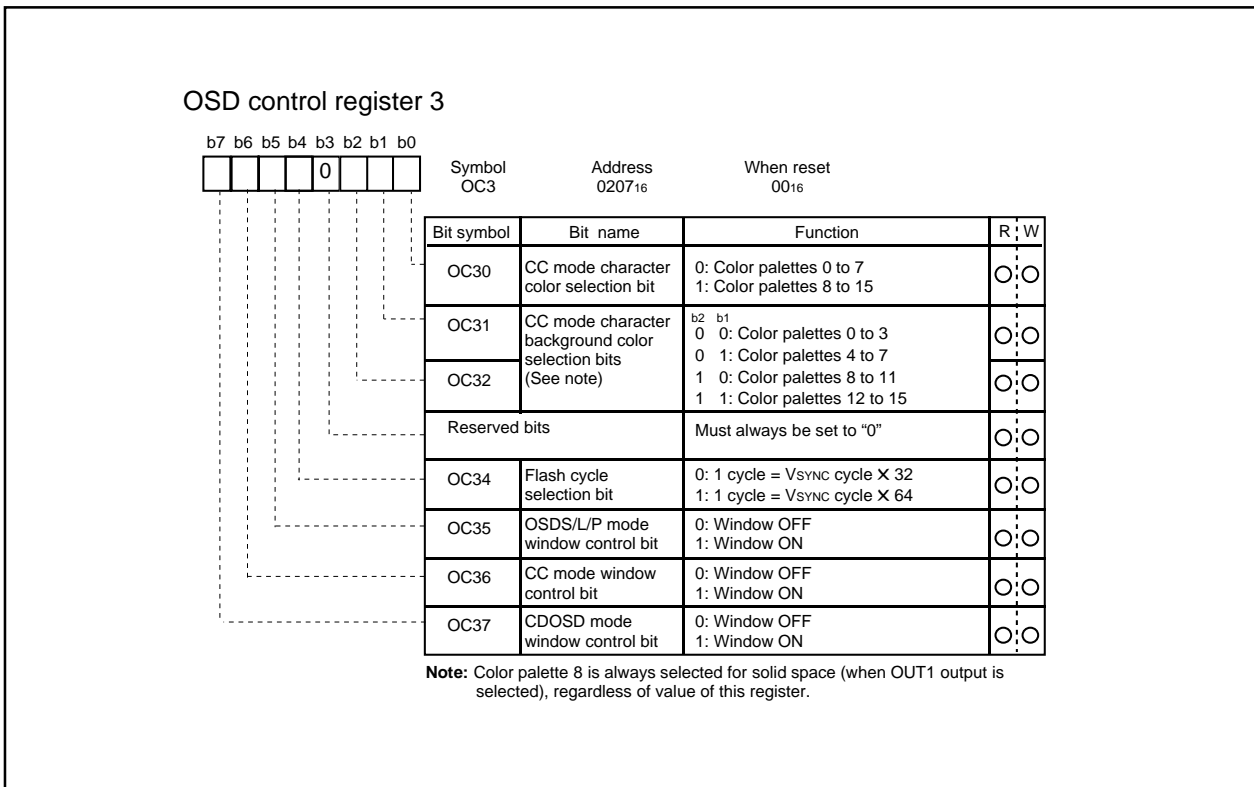
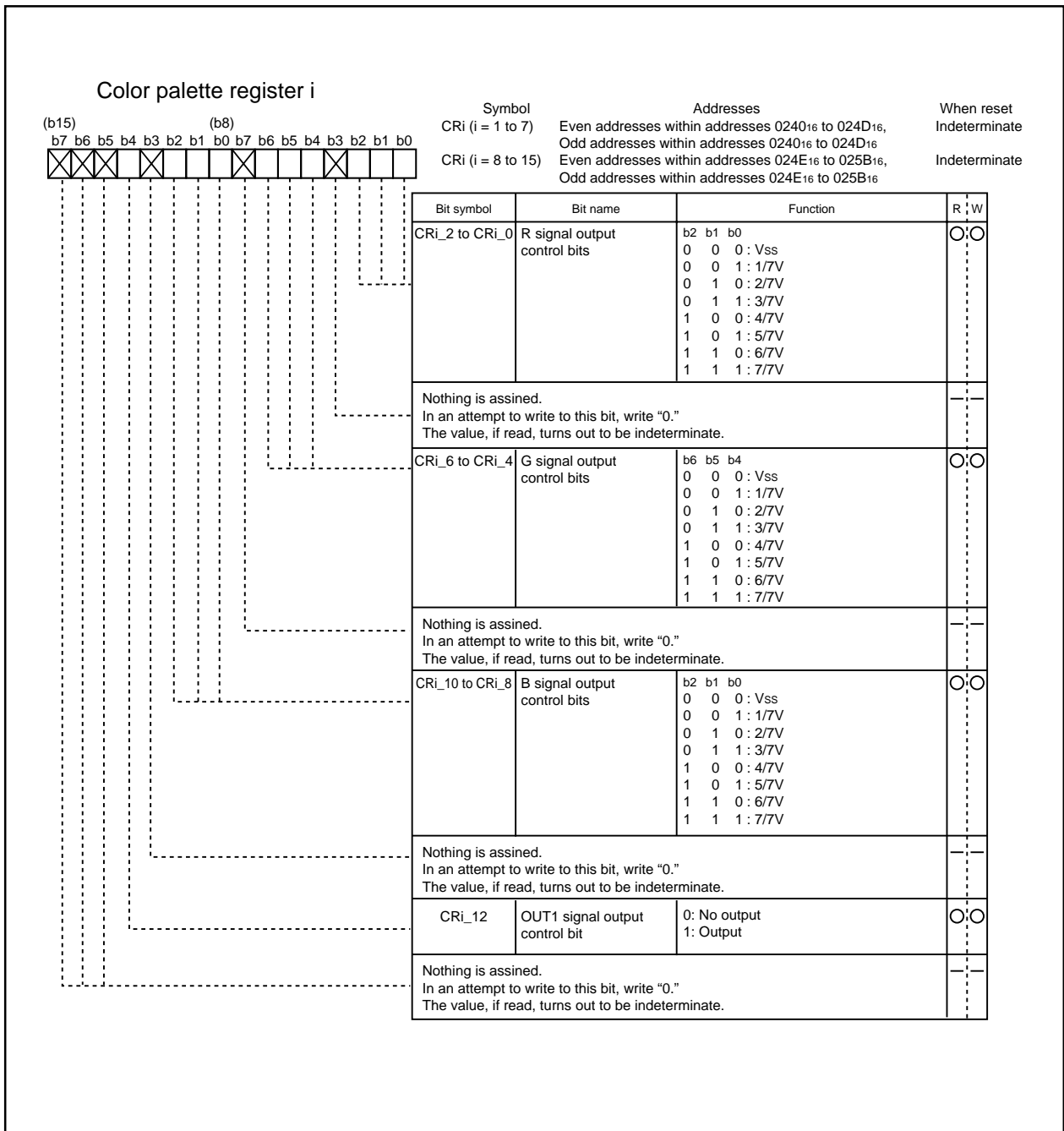


Figure 2.16.27 Difference between color palette 0 (transparent) and transparent setting of other color palettes



**Figure 2.16.28** OSD control register 3





**Figure 2.16.29 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 2.16.29), 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 2.16.30.

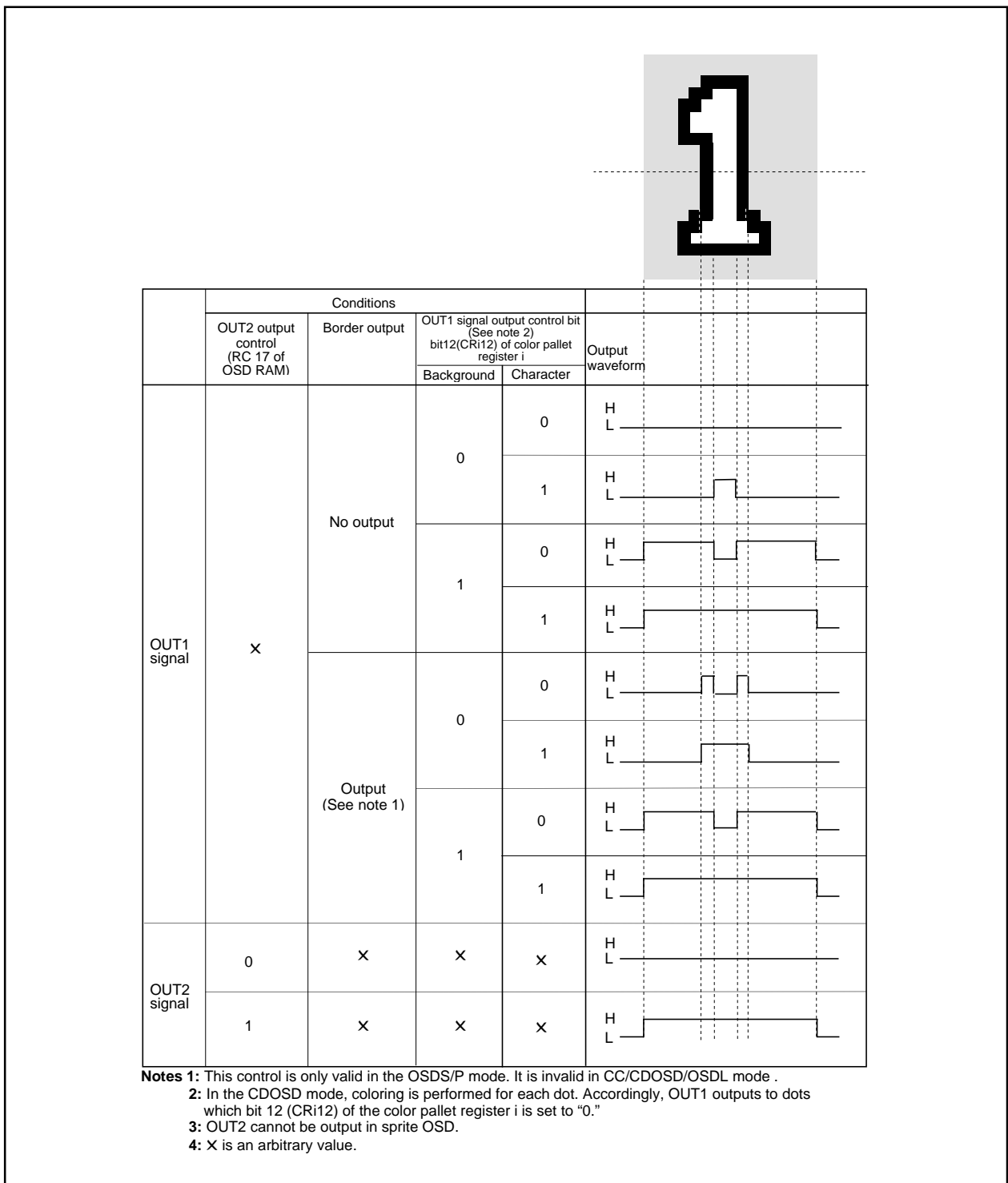


Figure 2.16.30 Setting value for controlling OUT1, OUT2 and corresponding output waveform

## 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 2.16.30. 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 2.16.30 (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 2.16.31).

**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 2.16.31).

**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 2.16.31).

**8:** When use the italic character which the pre-divide ratio = 1, do not use the character in which dot data exists for the right end of a font.

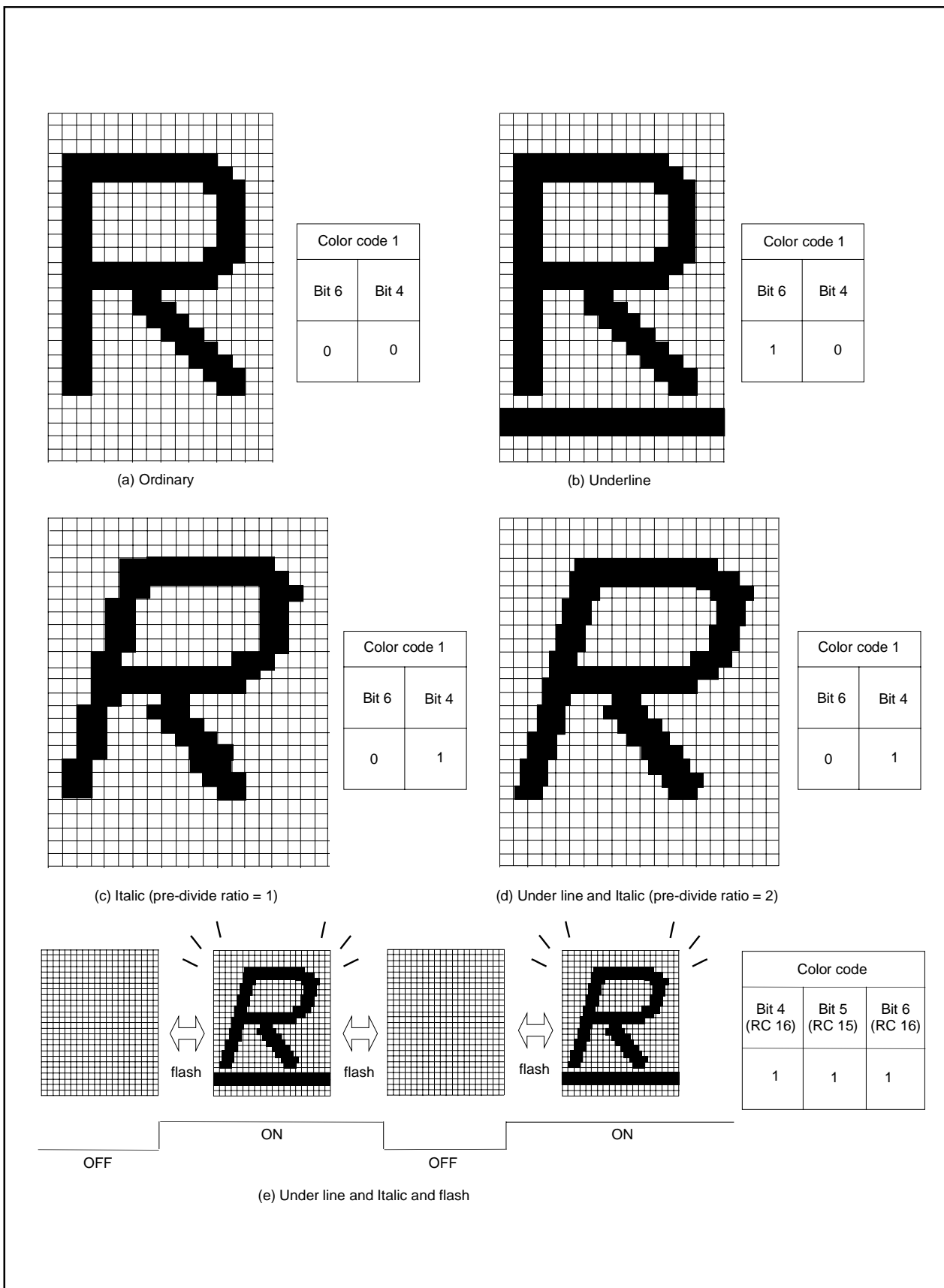
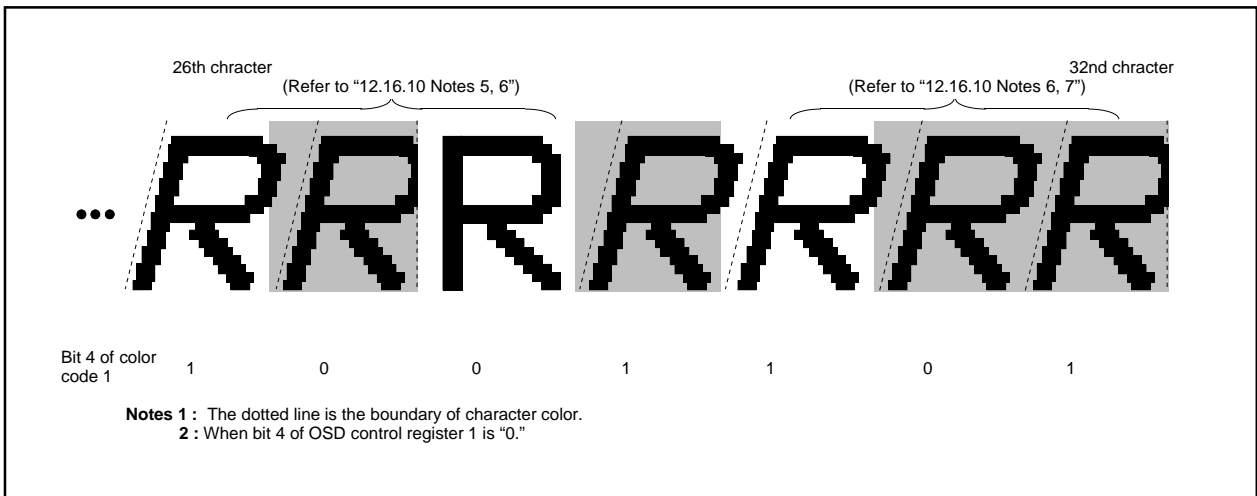


Figure 2.16.31 Example of attribute display (in CC mode)



**Figure 2.16.32 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 2.16.33) 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).

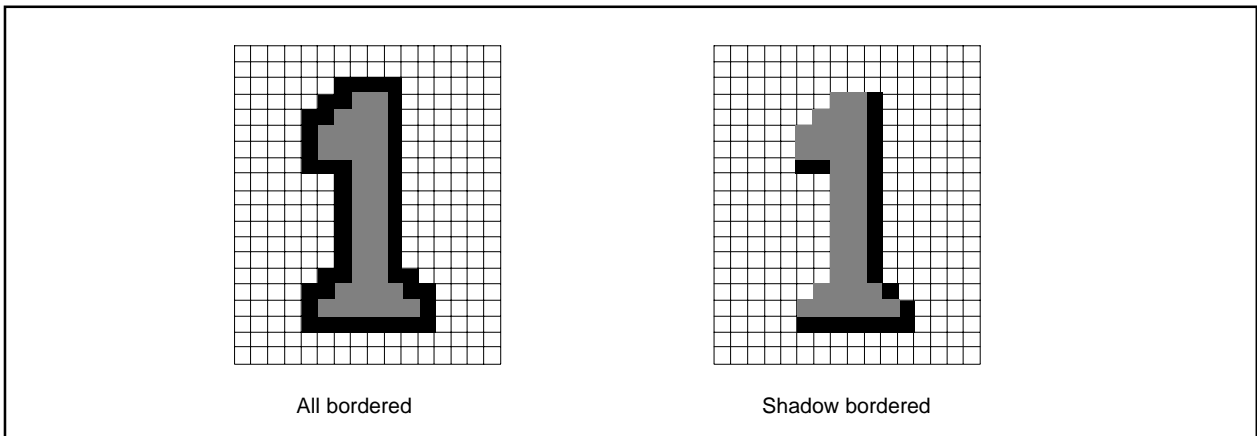
The OUT1 signal is used for border output. The border color is fixed at color palette 8 (black). The border color for each screen is specified by the border color register i.

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.

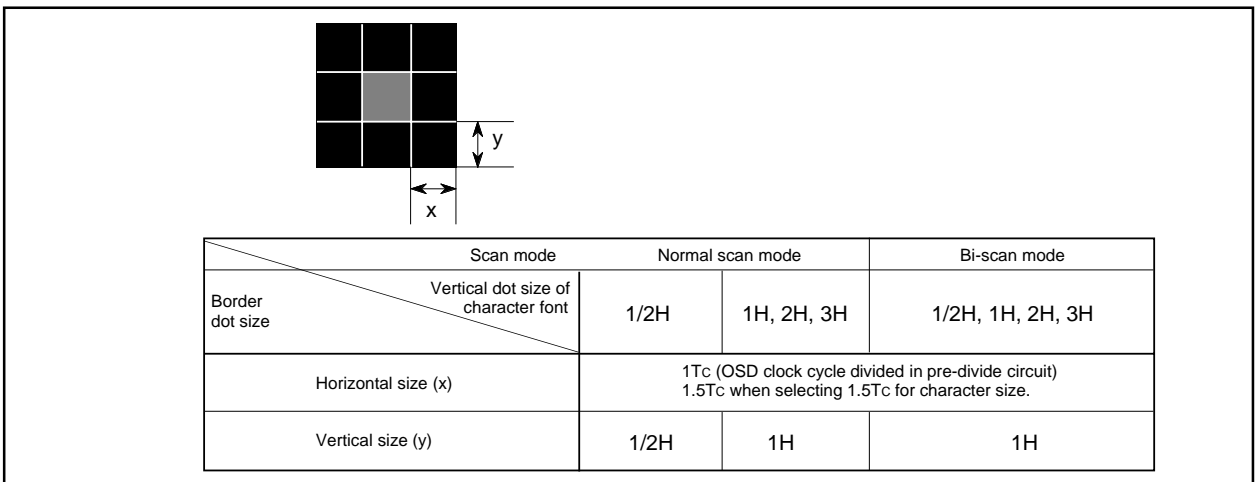
**Notes 1 :** The border dot area is the shaded area as shown in Figure 2.16.35.

**2 :** When the border dot overlaps on the next character font, the character font has priority (refer to Figure 2.16.36 A). When the border dot overlaps on the next character back ground, the border has priority (refer to Figure 2.16.36 B).

**3 :** The border in vertical out of character area is not displayed (refer to Figure 2.16.36).



**Figure 2.16.33 Example of border display**



**Figure 2.16.34 Horizontal and vertical size of border**

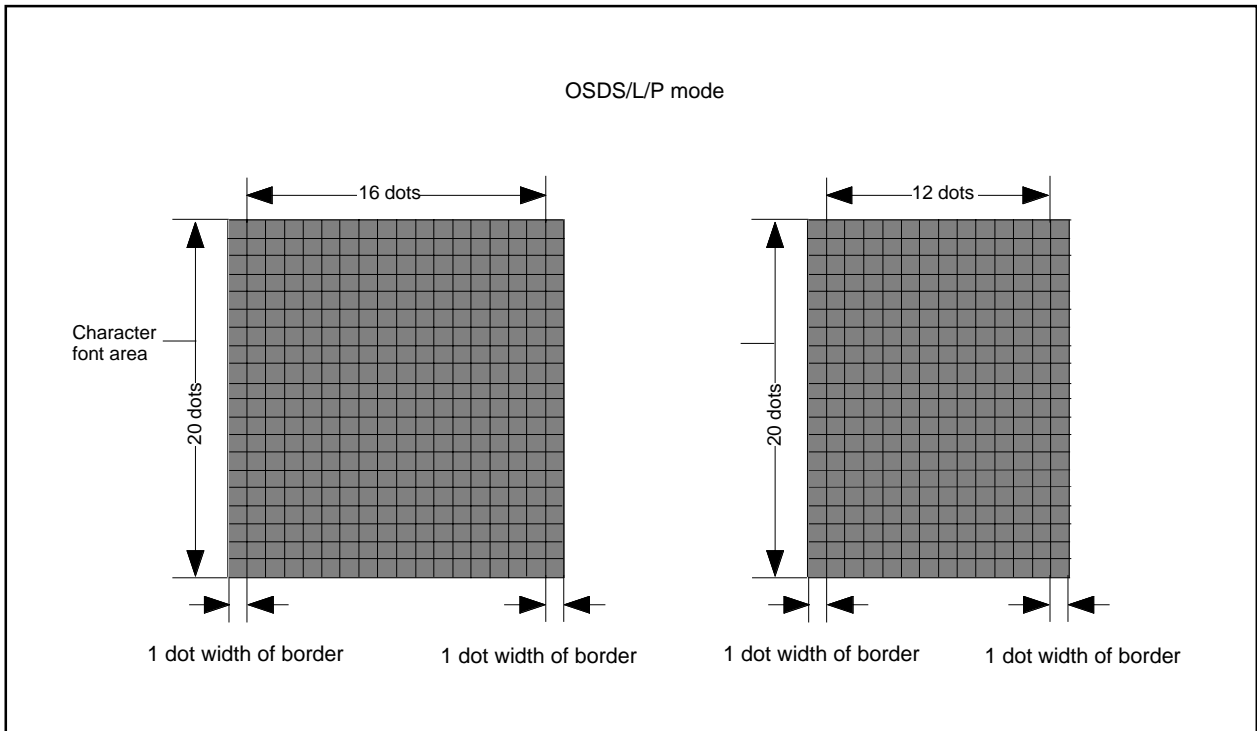


Figure 2.16.35 Border area

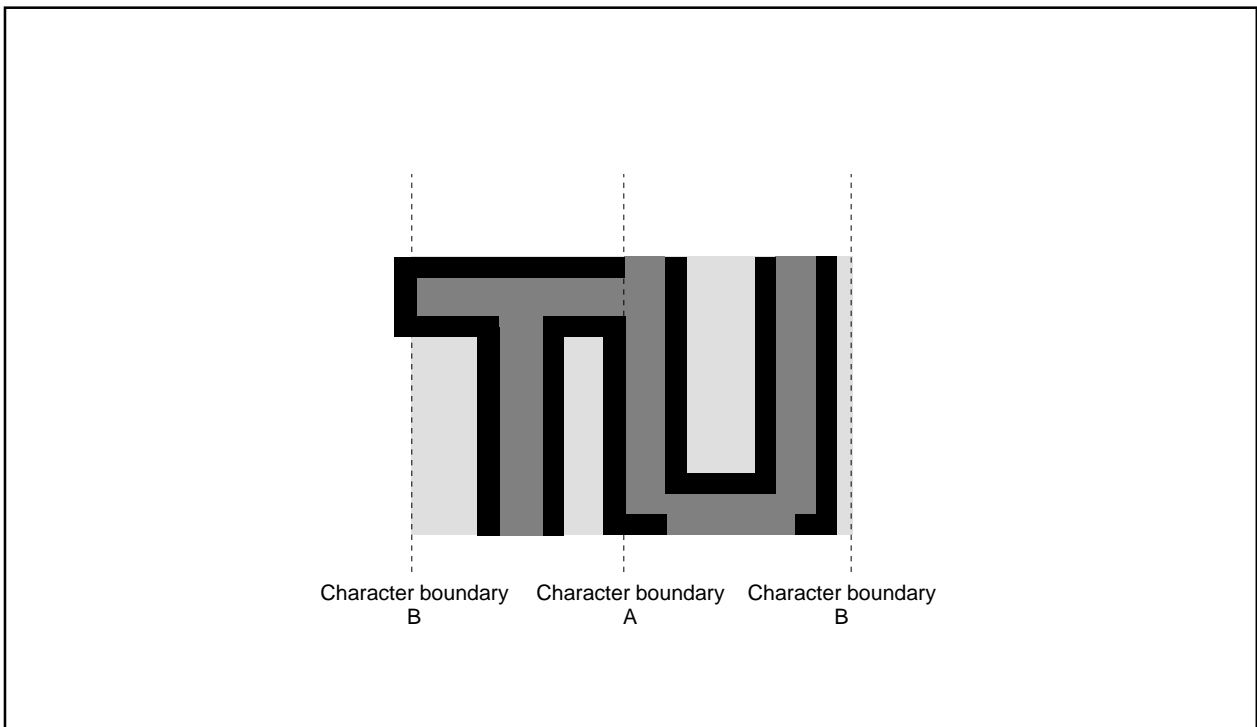


Figure 2.16.36 Border priority

### 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).

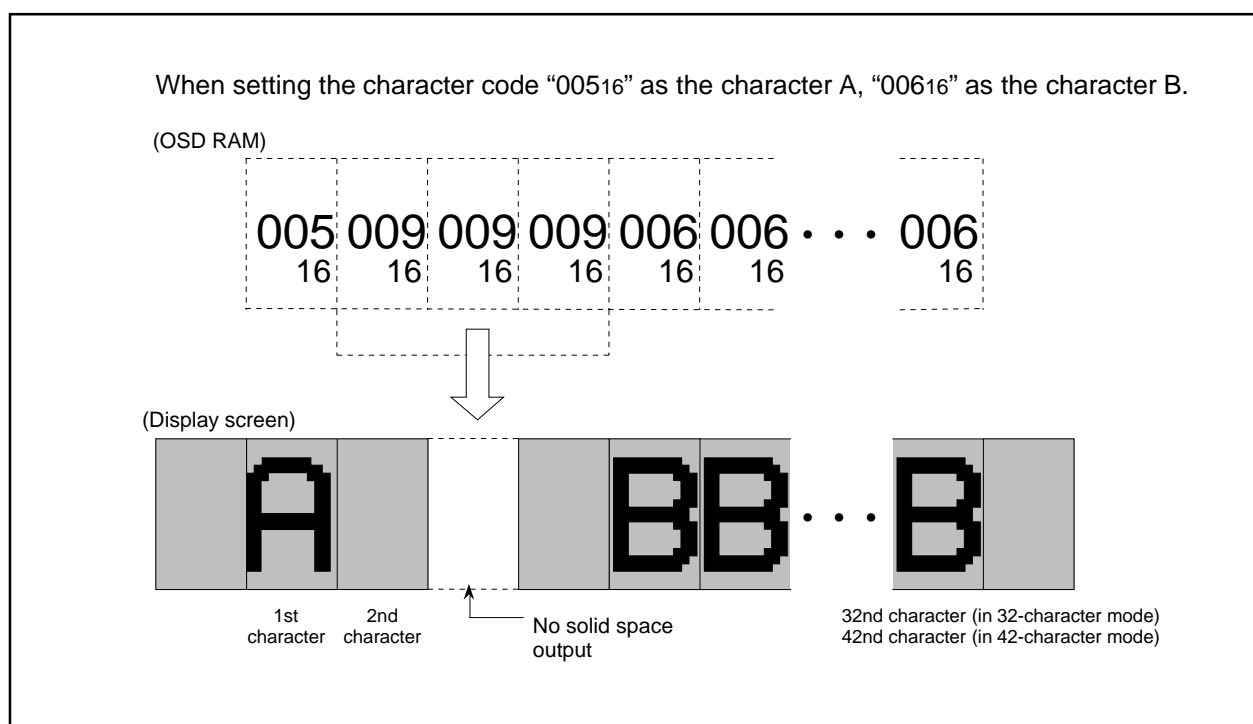
OUT1 or OUT2 output is selected by bit 3 of the OSD control register 2.

**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 2.16.37 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 2.16.38 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.

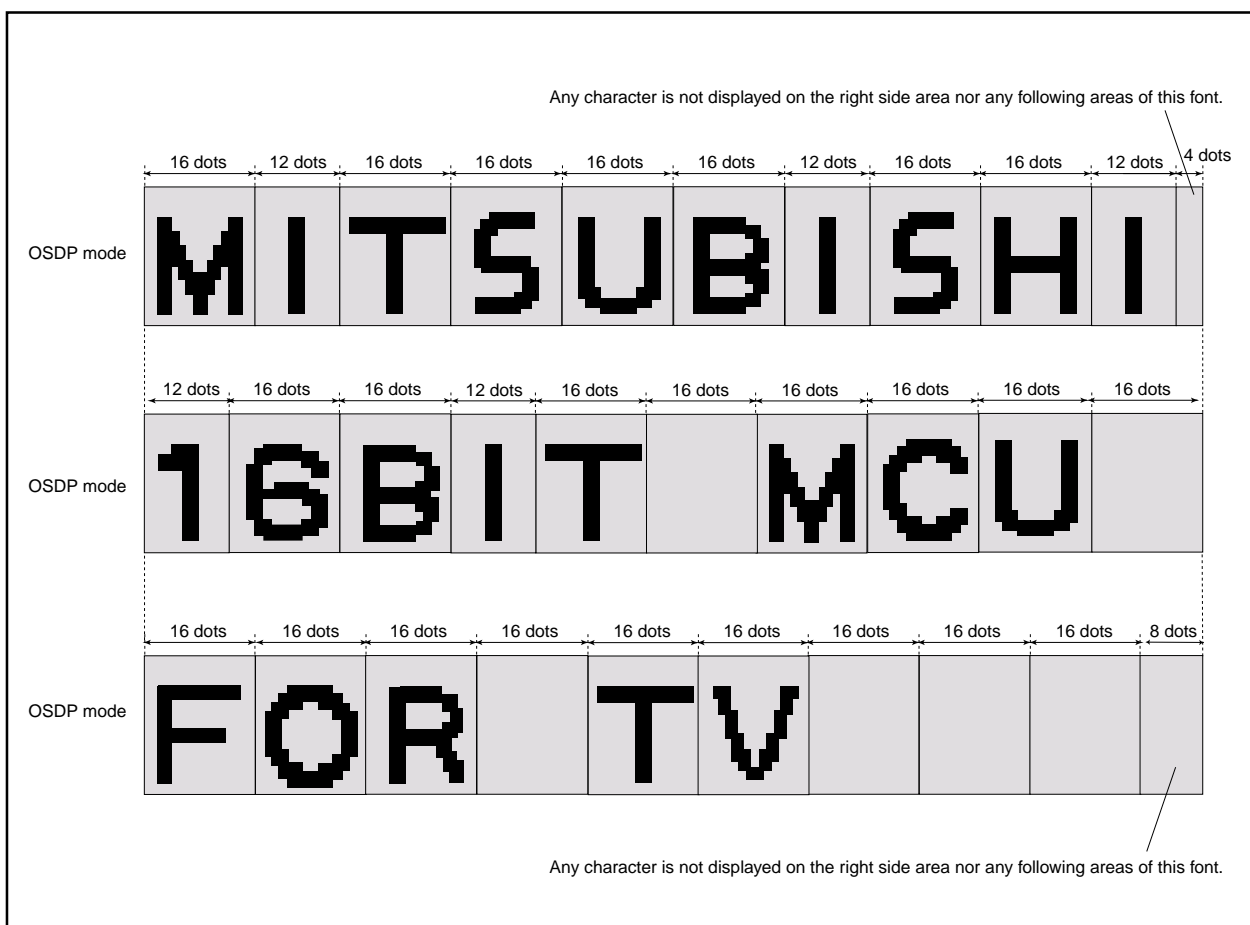
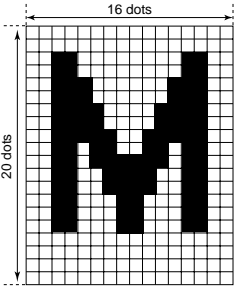
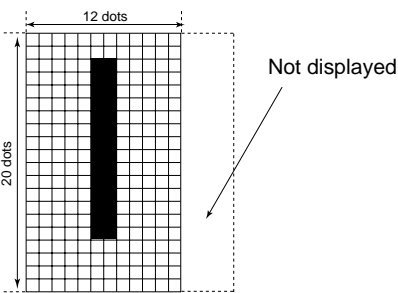
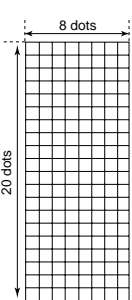
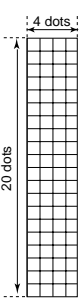


Figure 2.16.38 Display example of OSD mode block

**Table 2.16.11 Corresponding between character codes and display fonts**

Character code	Display fonts	Notes
000 <sub>16</sub> to 0EF <sub>16</sub> , 100 <sub>16</sub> to 2FF <sub>16</sub>  (except 100 <sub>16</sub> , 180 <sub>16</sub> , 200 <sub>16</sub> , 280 <sub>16</sub> )		
0F0 <sub>16</sub> to 0FD <sub>16</sub>		<ul style="list-style-type: none"> <li>• The left 12-dot part (16 X 12 dots) of set font is displayed.</li> <li>• In CC and OSDS modes, entire part (16 X 20 dots) of set font is displayed.</li> </ul>
3FE <sub>16</sub>		<ul style="list-style-type: none"> <li>• The blank font (only character background) is displayed.</li> <li>• Any character is not displayed on the right side area nor any following areas of this font.</li> <li>• Do not set this font for the 1st character (left edge) of a block.</li> </ul>
3FF <sub>16</sub>		<ul style="list-style-type: none"> <li>• The blank font (only character background) is displayed.</li> <li>• Any character is not displayed on the right side area nor any following areas of this font.</li> <li>• Do not set this font for the 1st character (left edge) of a block.</li> </ul>

### 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.

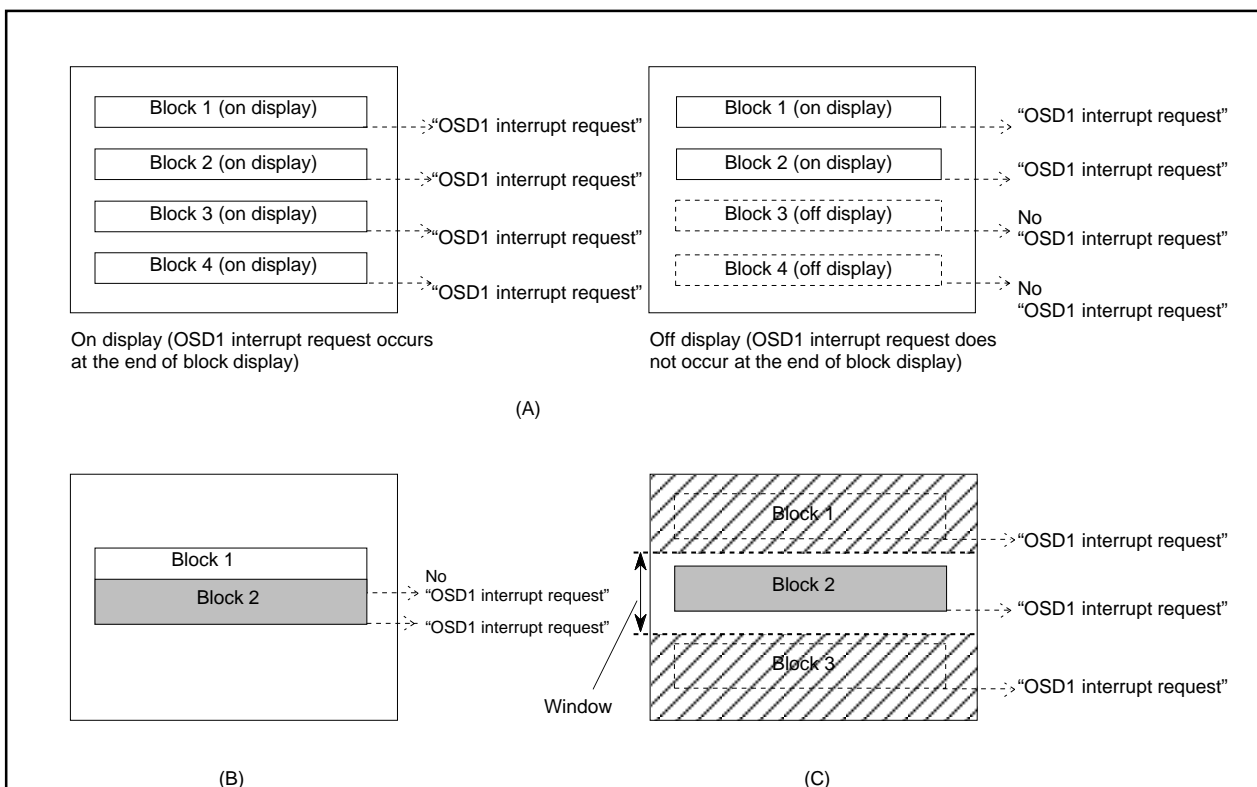
An OSD1 interrupt request occurs at the point at which display of each block has been completed. In other words, when a scanning line reaches the point of the display position (specified by the vertical position registers) of a certain block, the character display of that block starts, and an interrupt 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 2.16.38 (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 2.16.38 (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 2.16.38 (C)).



**Figure 2.16.39 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.

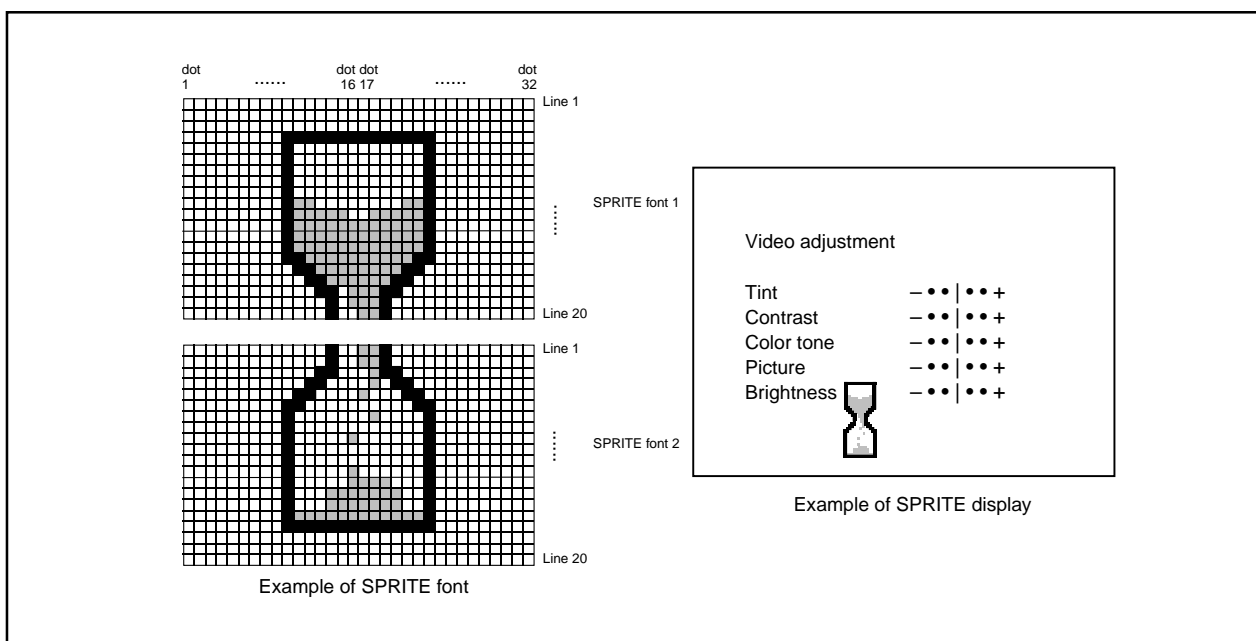
When SPRITE display overlaps with other OSD displays, SPRITE display is always given priority. However, the SPRITE display overlaps with the display which includes OUT2 output, OUT2 in the OSD is output without masking.

**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 2.16.40** SPRITE OSD display example

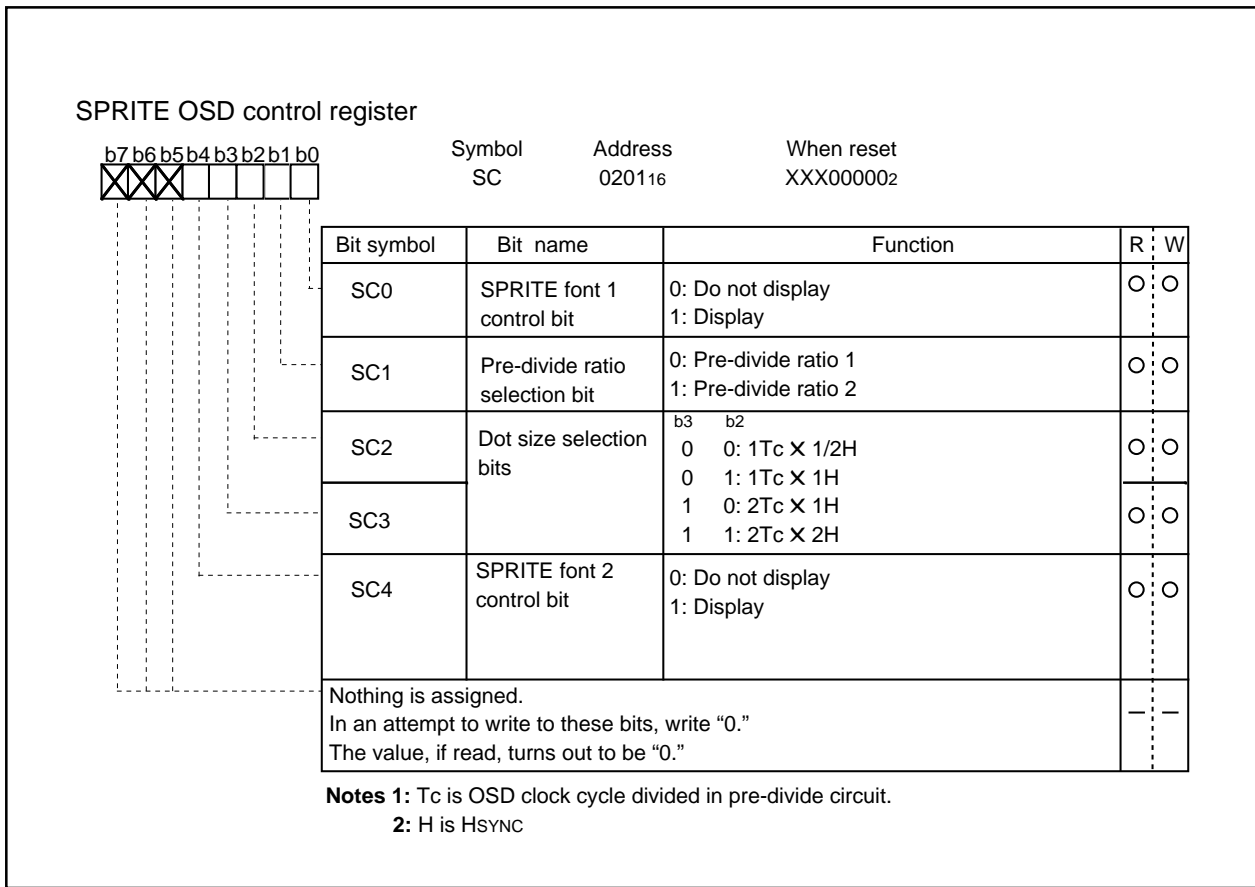


Figure 2.16.41 SPRITE OSD control register

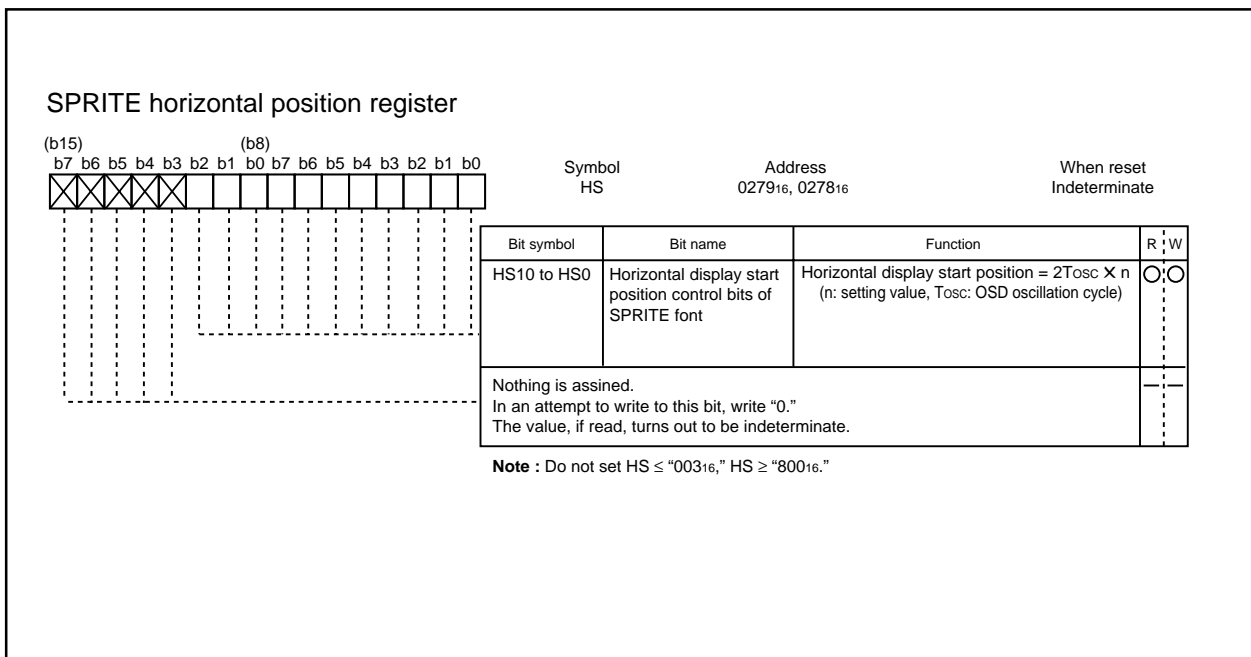


Figure 2.16.42 SPRITE horizontal position register

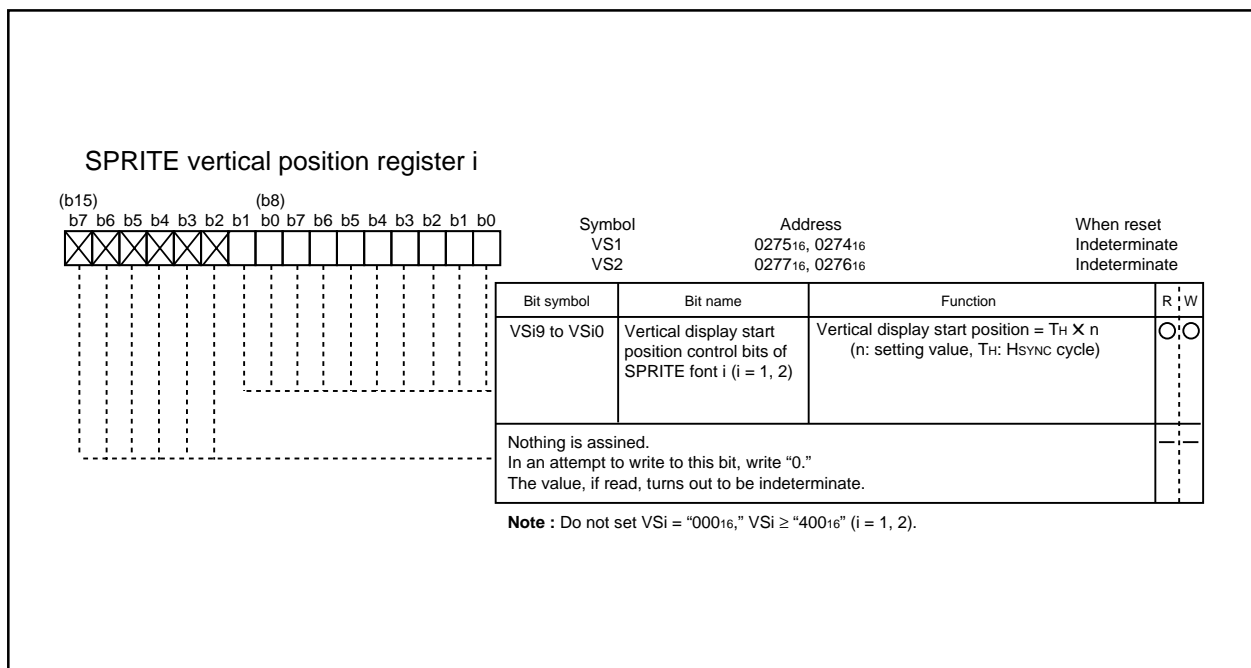


Figure 2.16.43 SPRITE vertical position register i (i = 1, 2)

### 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.

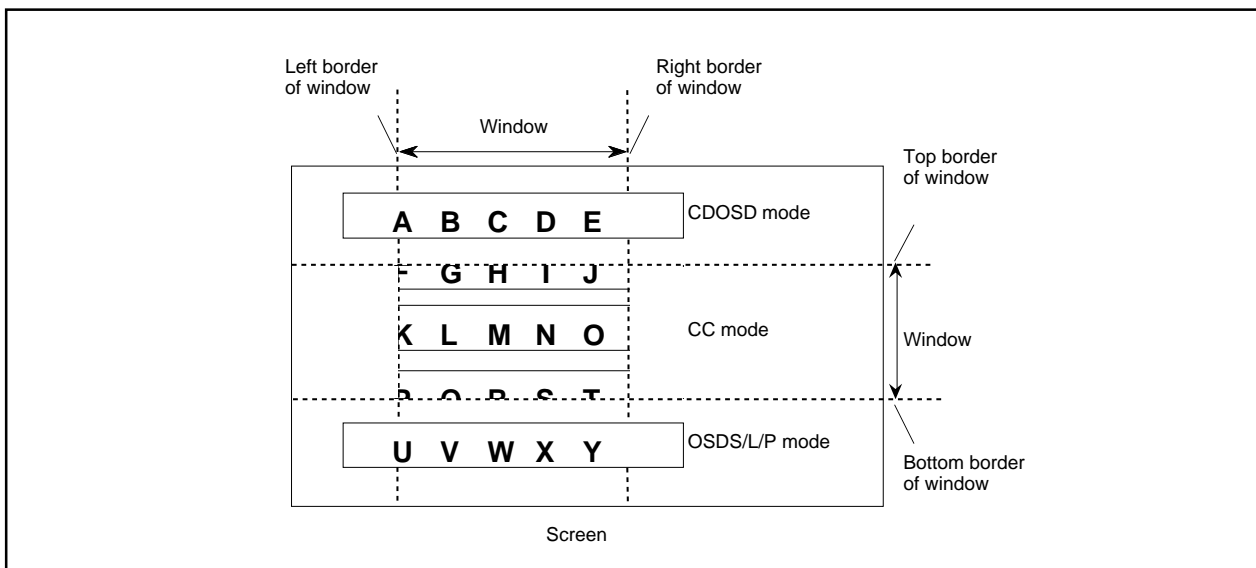
The ON/OFF for vertical window function is performed by bit 5 of the OSD control register 1 and is used to select vertical window function or vertical blank function by bit 6 of the OSD control register 2. Accordingly, the vertical window function cannot be used simultaneously with the vertical blank 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 2.16.44 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.

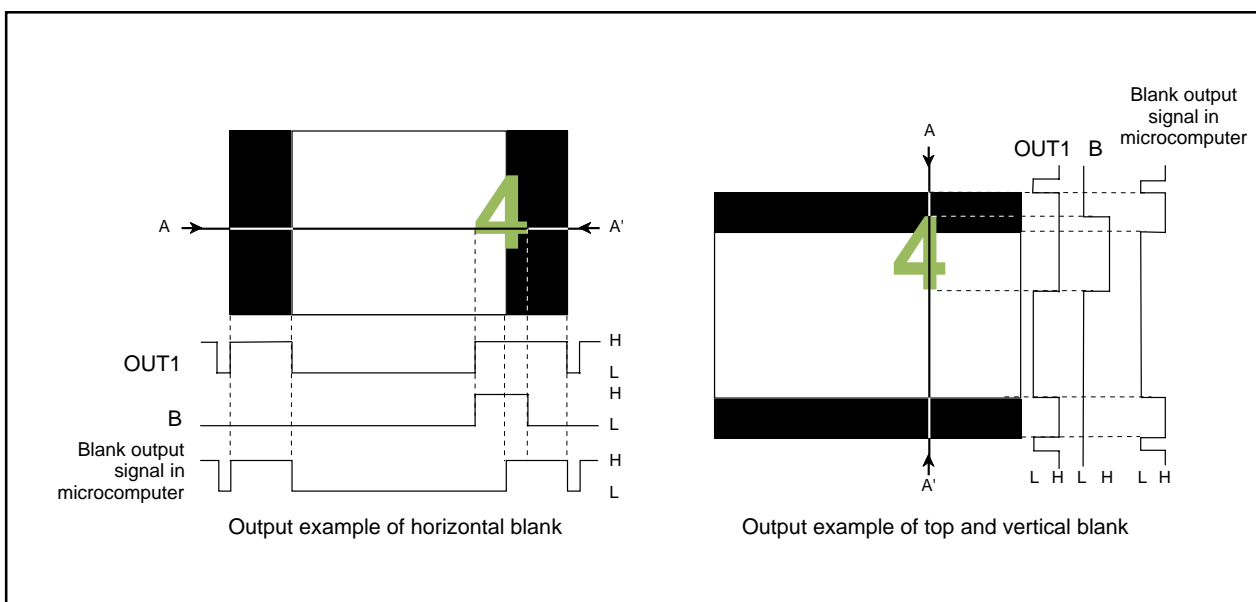
The ON/OFF for vertical blank function is performed by bit 5 of the OSD control register 1 and is used to select vertical window function or vertical blank function by bit 6 of the OSD control register 2. Accordingly, the vertical blank function cannot be used simultaneously with the vertical window 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.

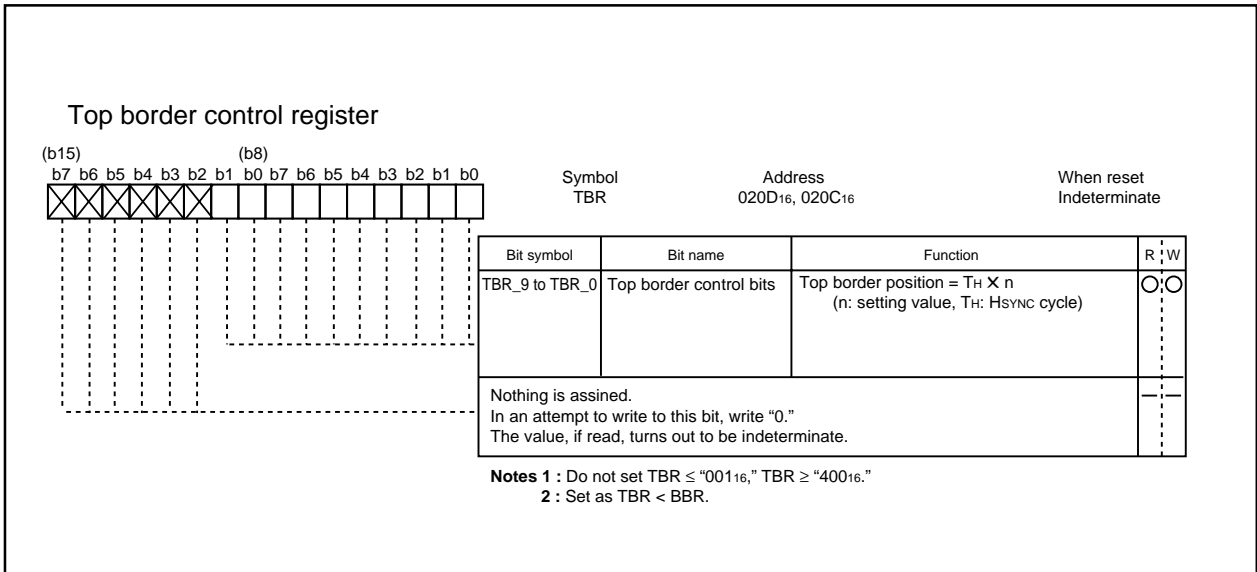
These blank signals are not output in the horizontal/vertical blanking interval.

- 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.

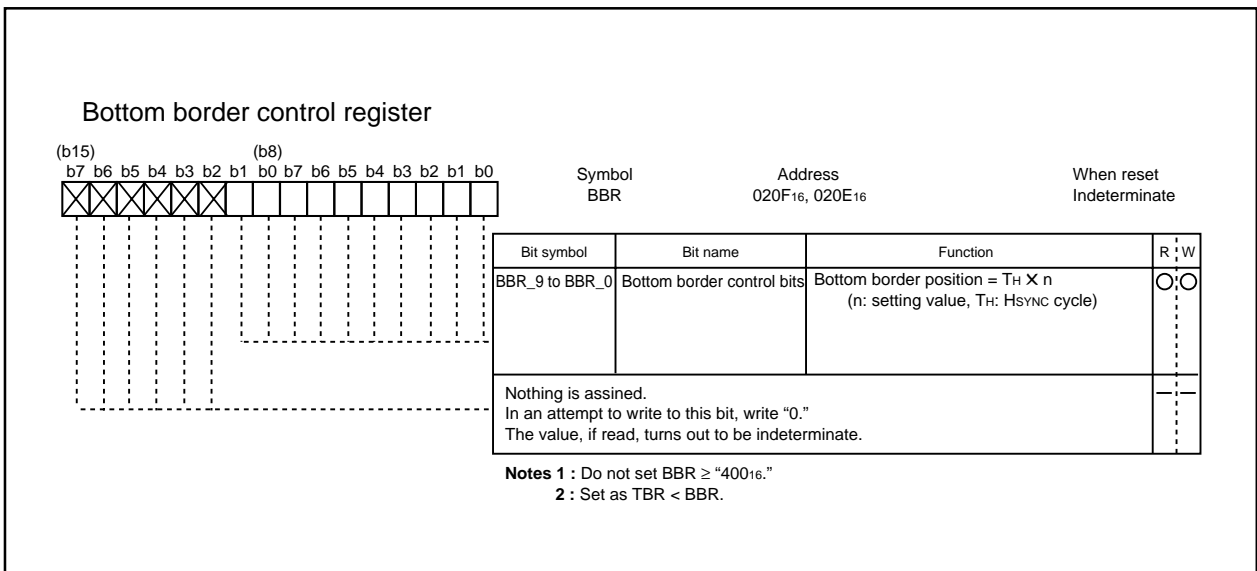


**Figure 2.16.45 Blank output example (when OSD output is B + OUT1)**





**Figure 2.16.46 Top border control register**



**Figure 2.16.47 Bottom border control register**

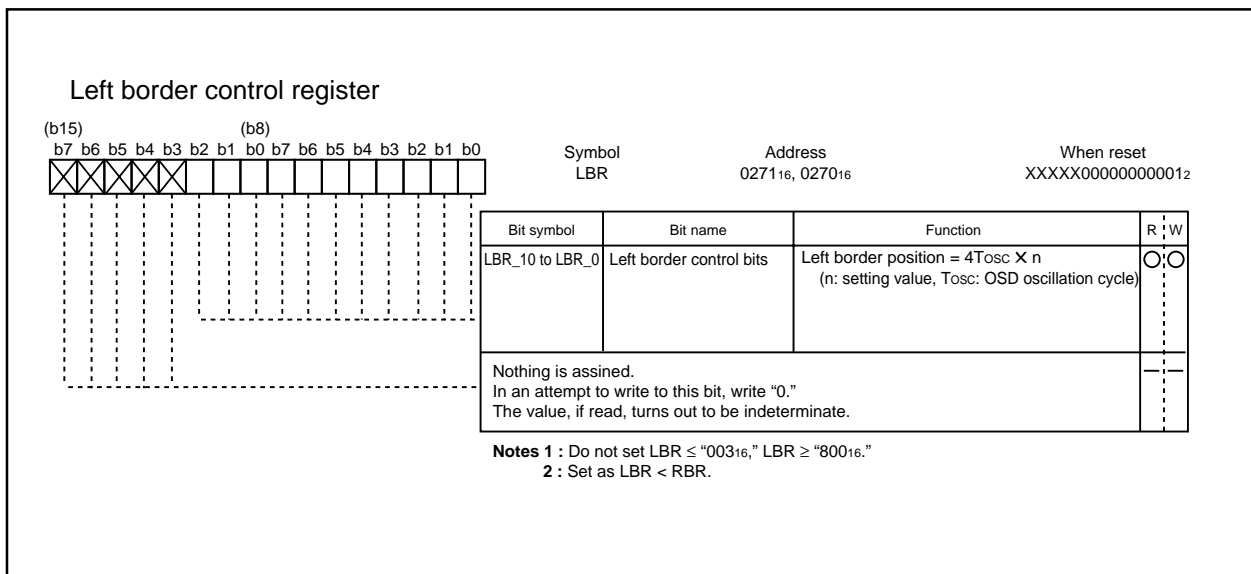


Figure 2.16.48 Left border control register

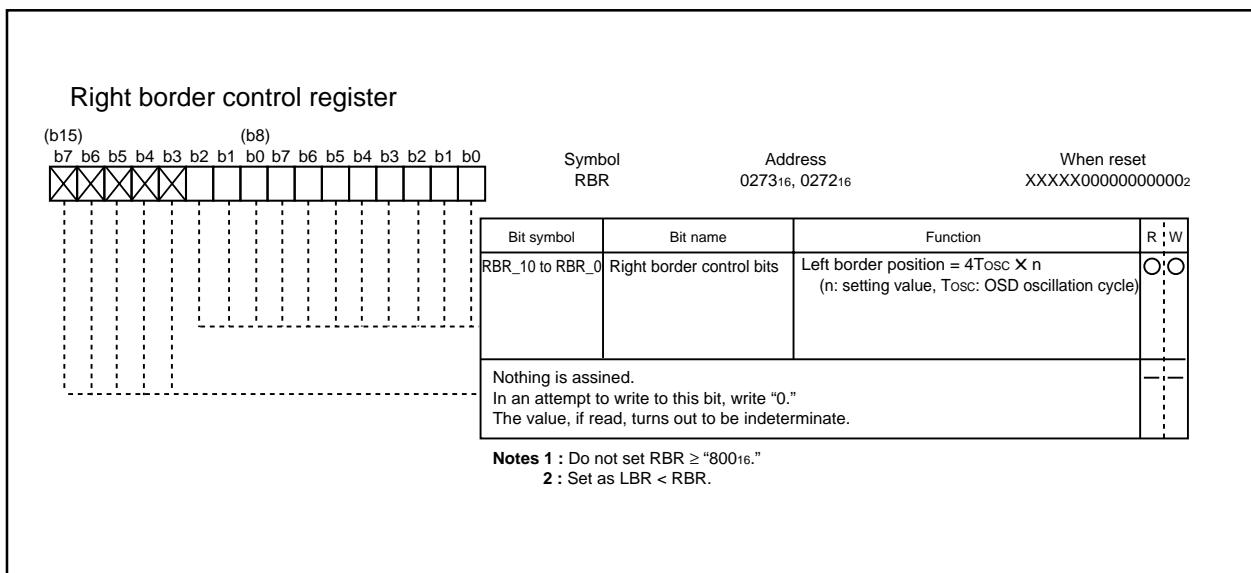


Figure 2.16.49 Right border control register

### 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 2.16.50, the example of raster coloring is shown in Figure 2.16.51.

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

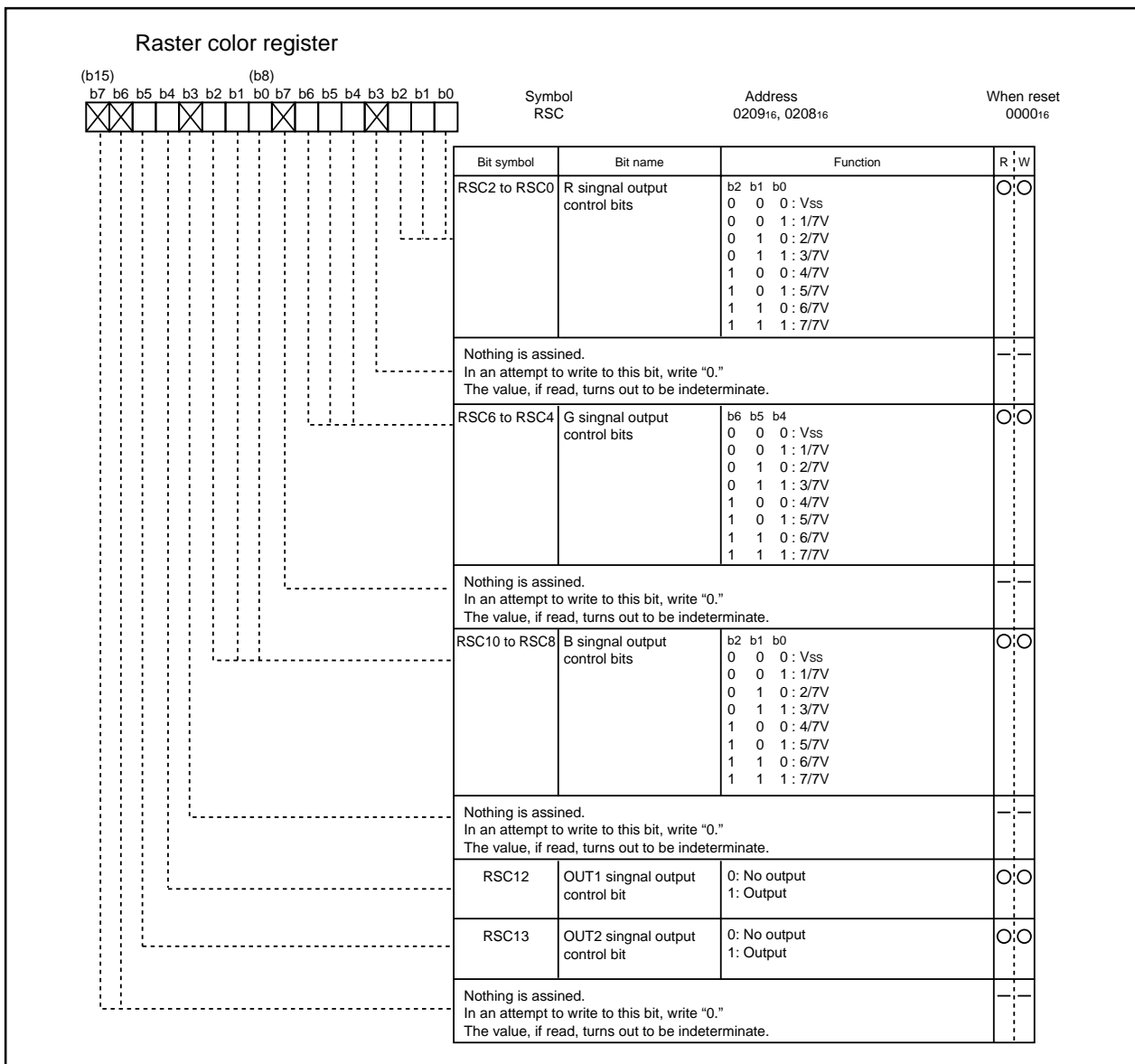


Figure 2.16.50 Raster color register

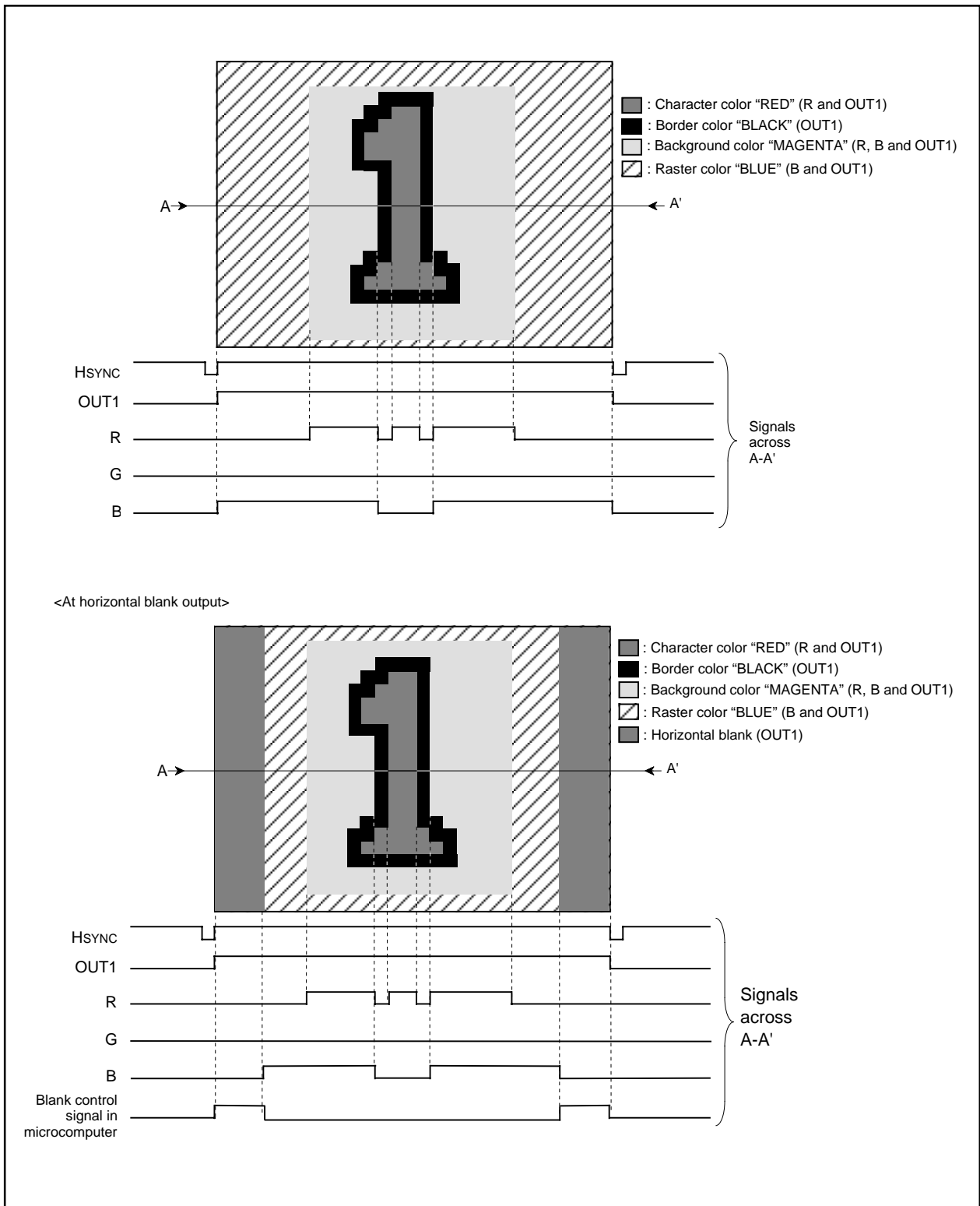


Figure 2.16.51 Example of raster coloring

### 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	0	DIGR0 (P9 <sub>2</sub> ), DIGR1 (R), DIGR2 (P10 <sub>7</sub> ) DIGG0 (P8 <sub>7</sub> ), DIGG1 (G), DIGG2 (P10 <sub>6</sub> ) DIGB0 (P8 <sub>6</sub> ), DIGB1 (B), DIGB2 (P10 <sub>5</sub> ) Each of these pins output two-level values. (Corresponding to each signal output control bit in color palette register i) DIGR0~2 correspond to CRi0~2, respectively. DIGG0~2 correspond to CRi4~6, respectively. DIGB0~2 correspond to CRi8~10, respectively.

### 2.16.20 OSD Reserved Register

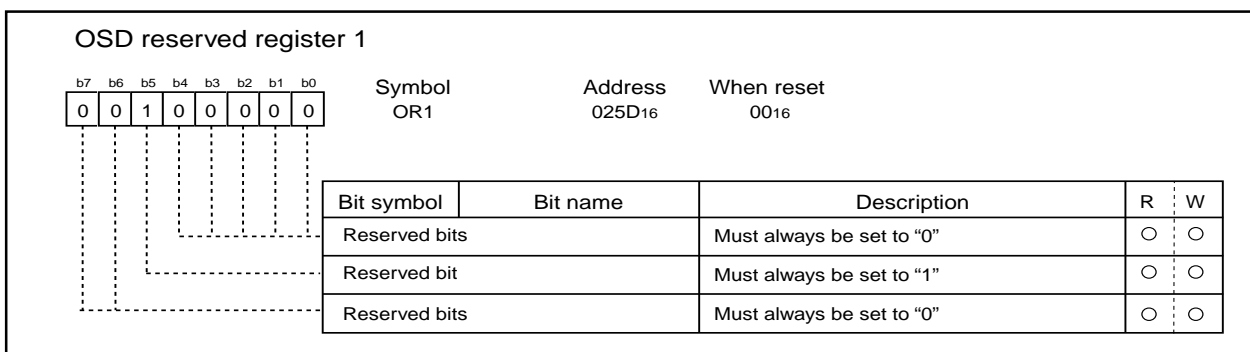


Figure 2.16.52 OSD reserved register 1

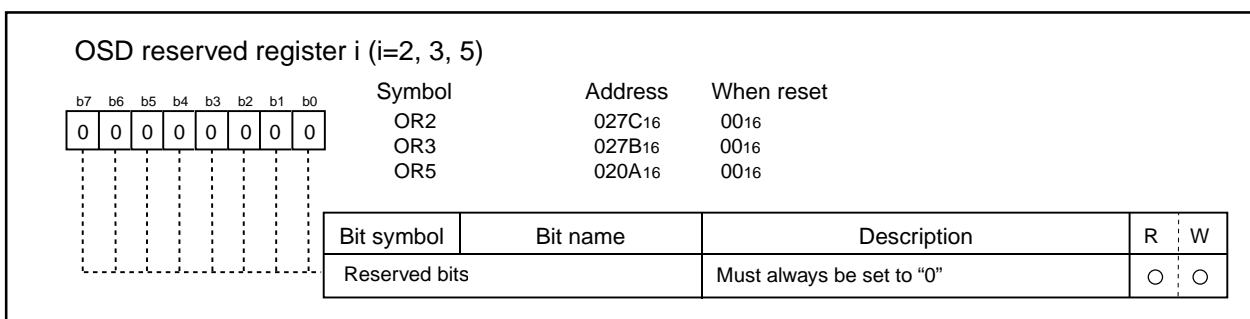


Figure 2.16.53 OSD reserved register i (i=2, 3, 5)

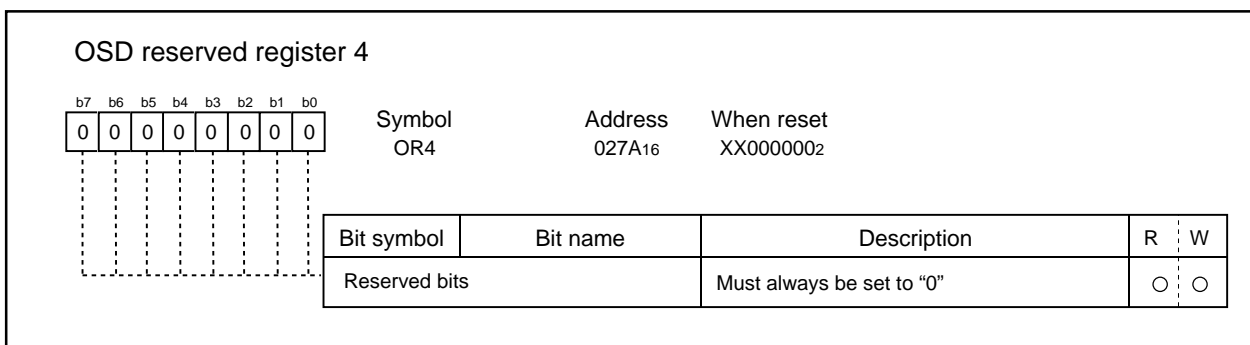


Figure 2.16.54 OSD reserved register 4

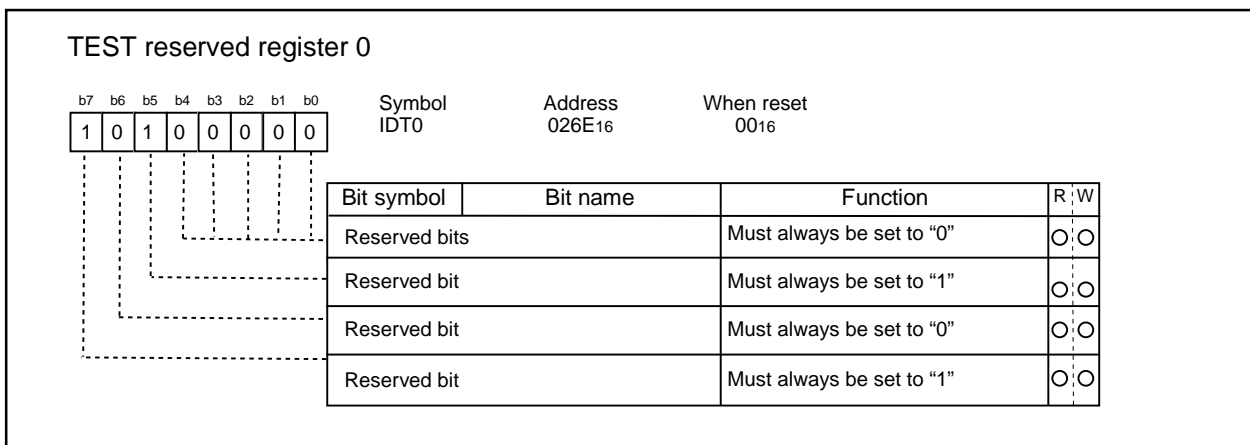


Figure 2.16.55 TEST reserved register 0

## 2.17 Programmable I/O Ports

There are 76 programmable I/O ports: P0–P5, P60–P63, P67, P7, P82, P83, P86, P87, P90–P94, P102–P107. 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 2.17.10 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 2.17.11 to 2.17.15 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 2.17.17 to 2.17.19 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 2.17.16 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.

When external bus width is 8 bits in microprocessor mode or memory expansion mode, this register is valid.



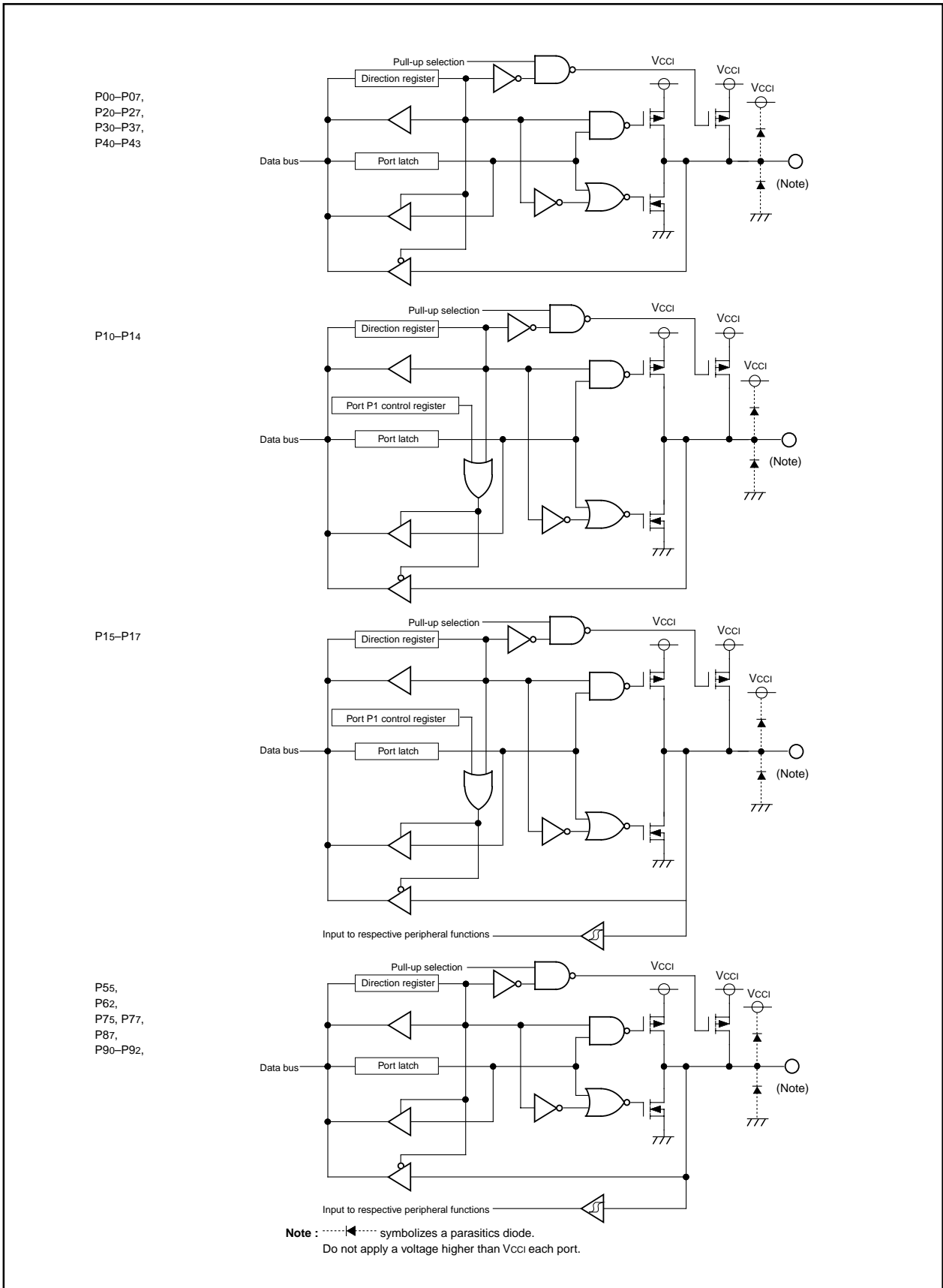


Figure 2.17.1 Programmable I/O ports (1)

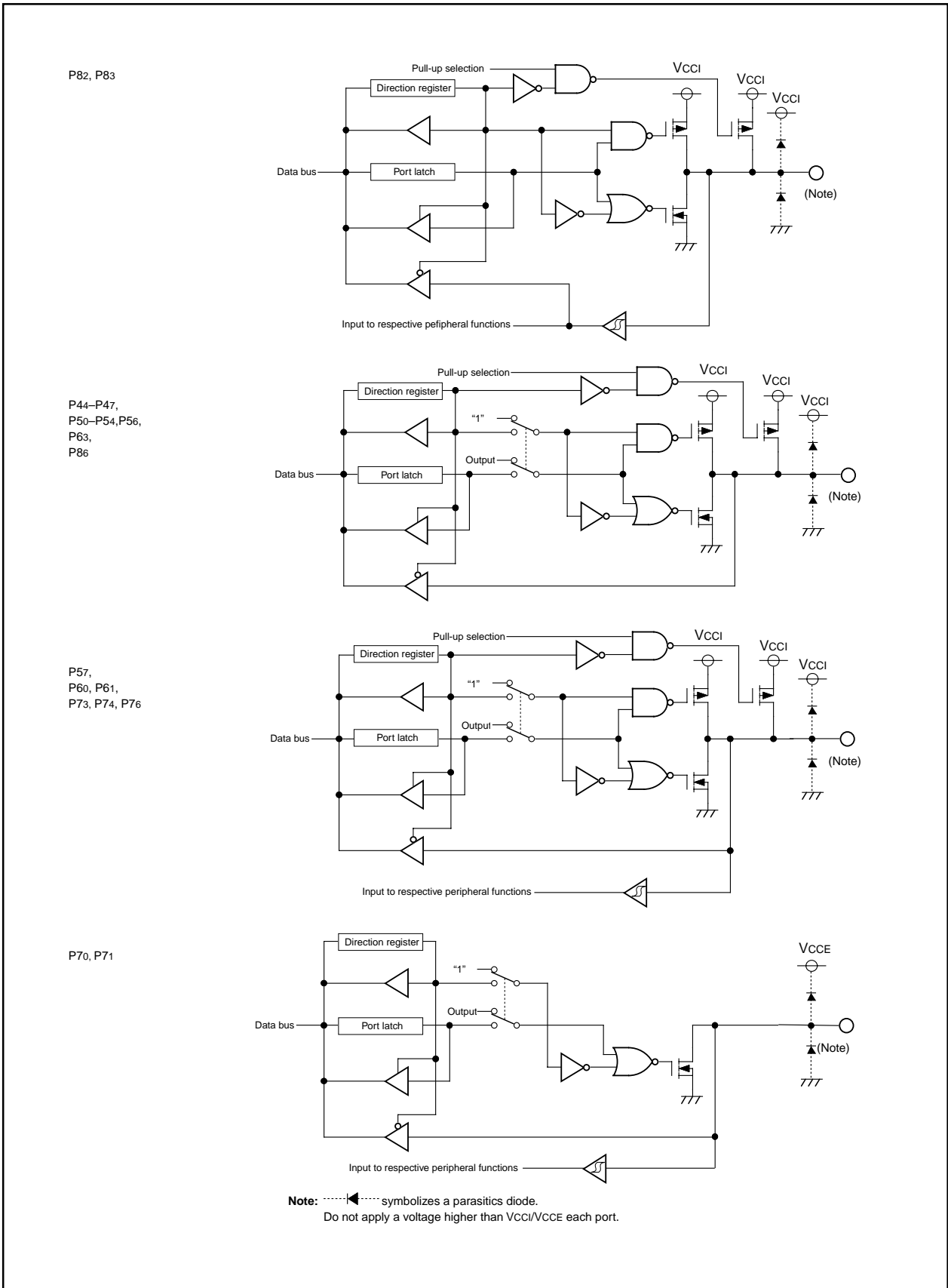


Figure 2.17.2 Programmable I/O ports (3)

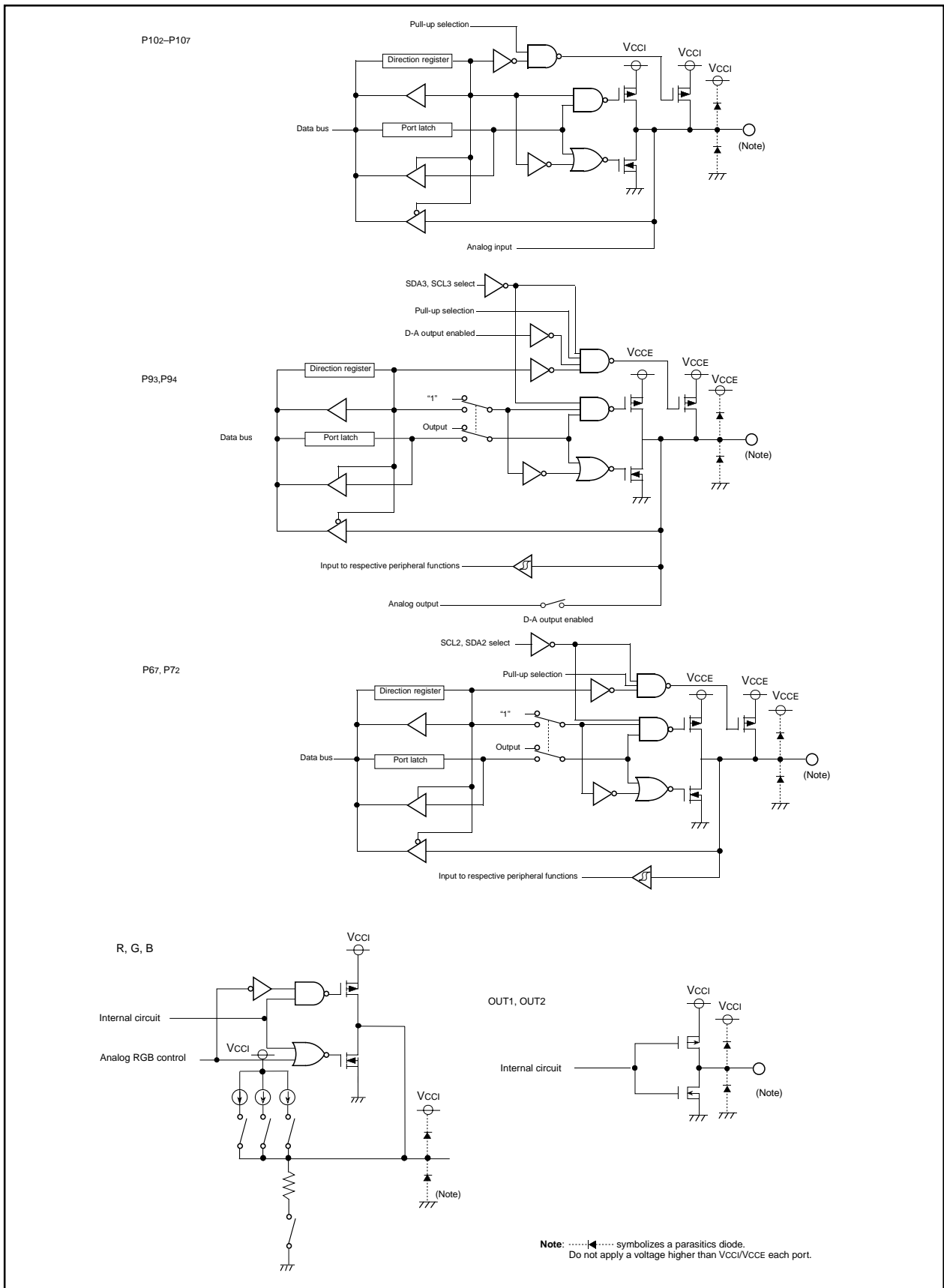


Figure 2.17.3 Programmable I/O ports (2)

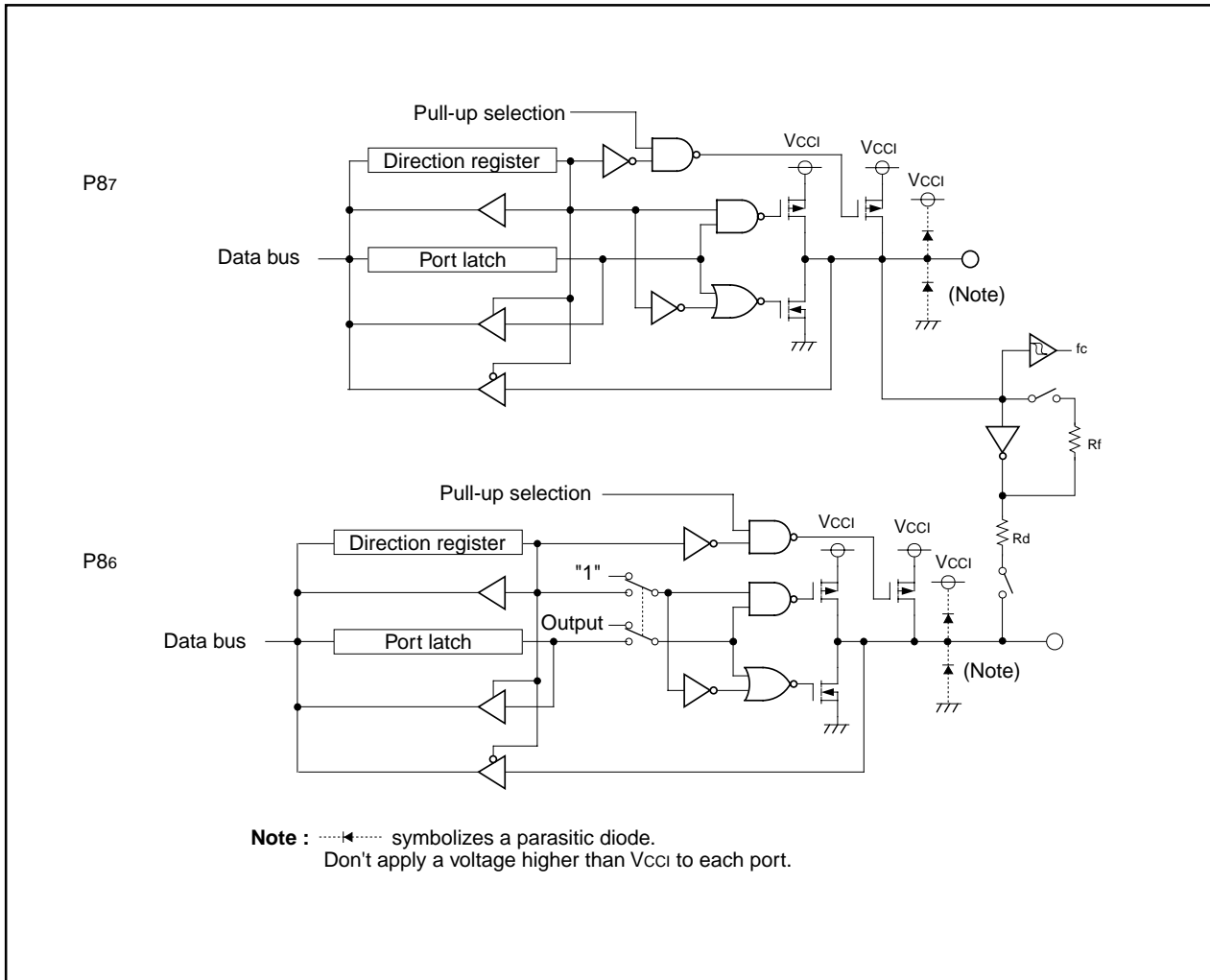


Figure 2.17.4 Programmable I/O ports (4)

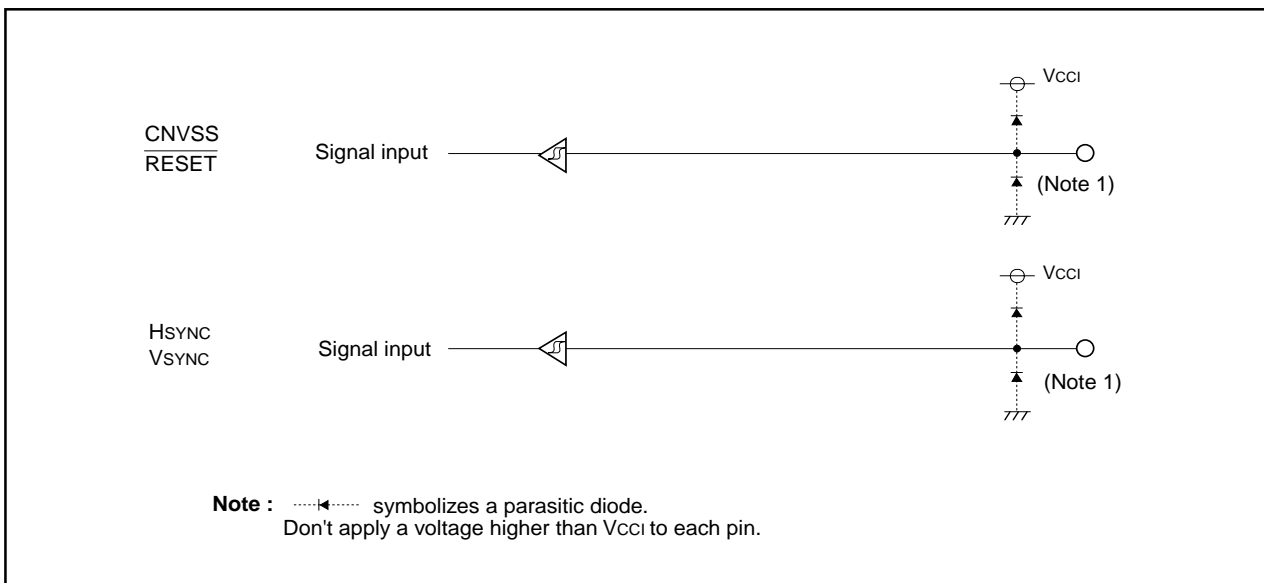


Figure 2.17.5 I/O pins

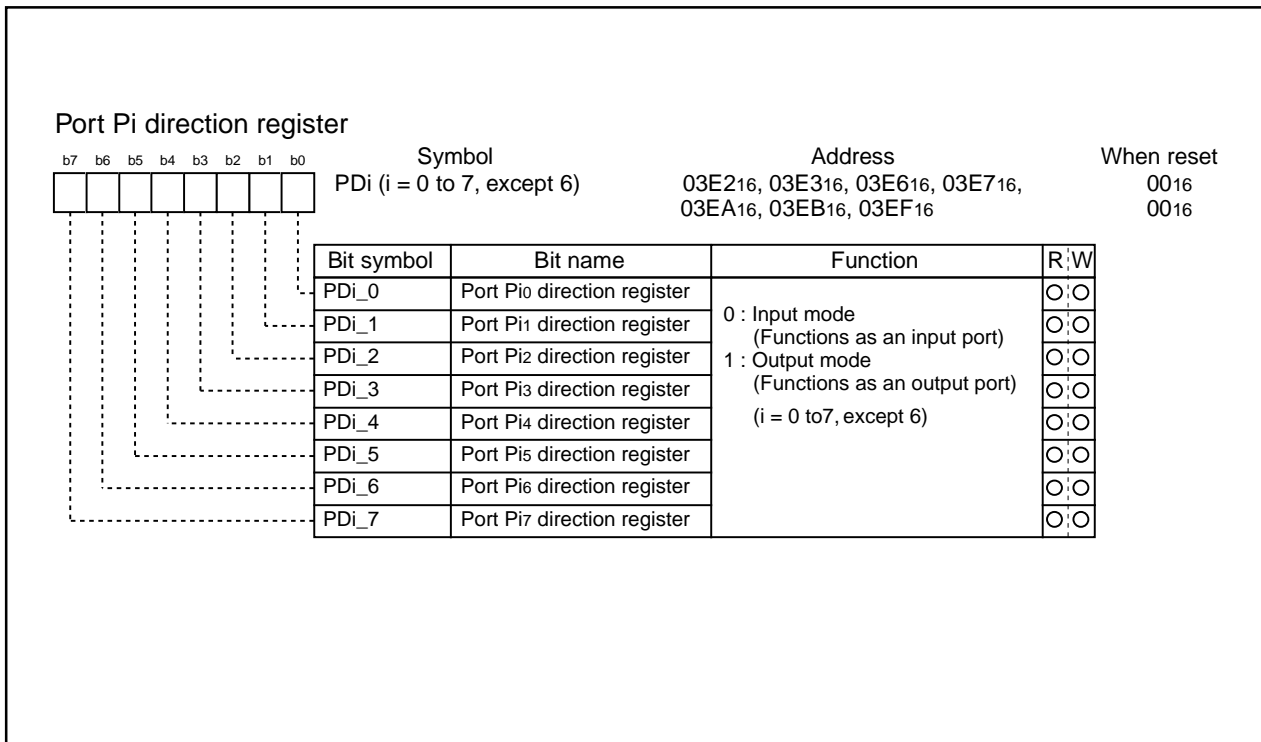


Figure 2.17.6 Port Pi direction register (i = 0 to 7, except 6)

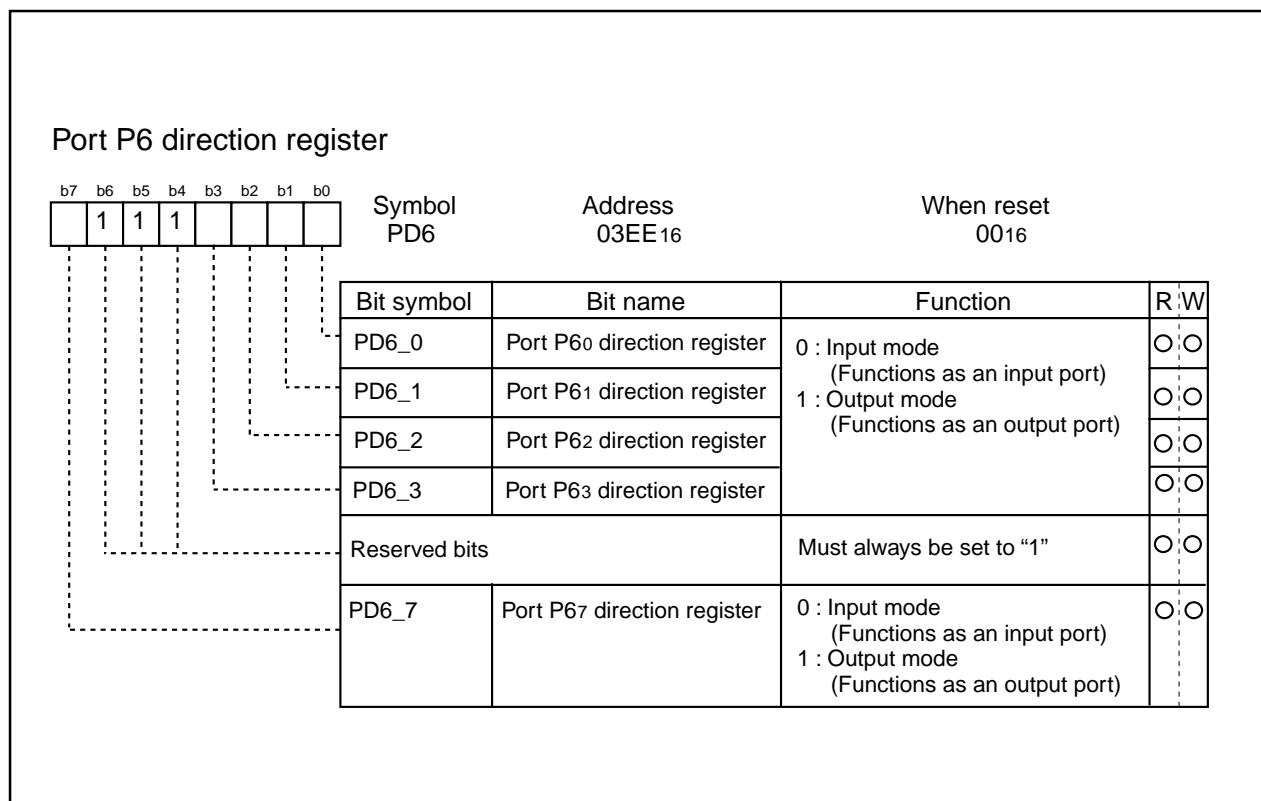


Figure 2.17.7 Port P6 direction register

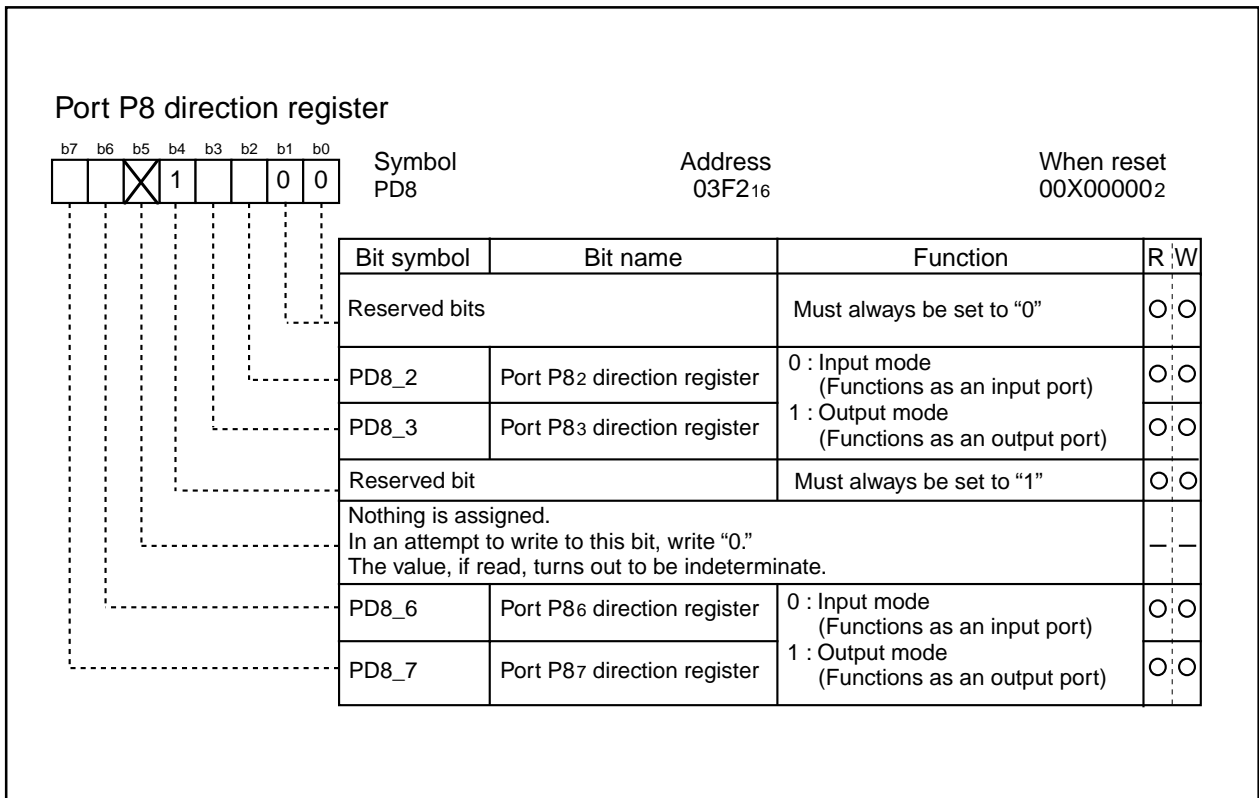


Figure 2.17.8 Port P8 direction register

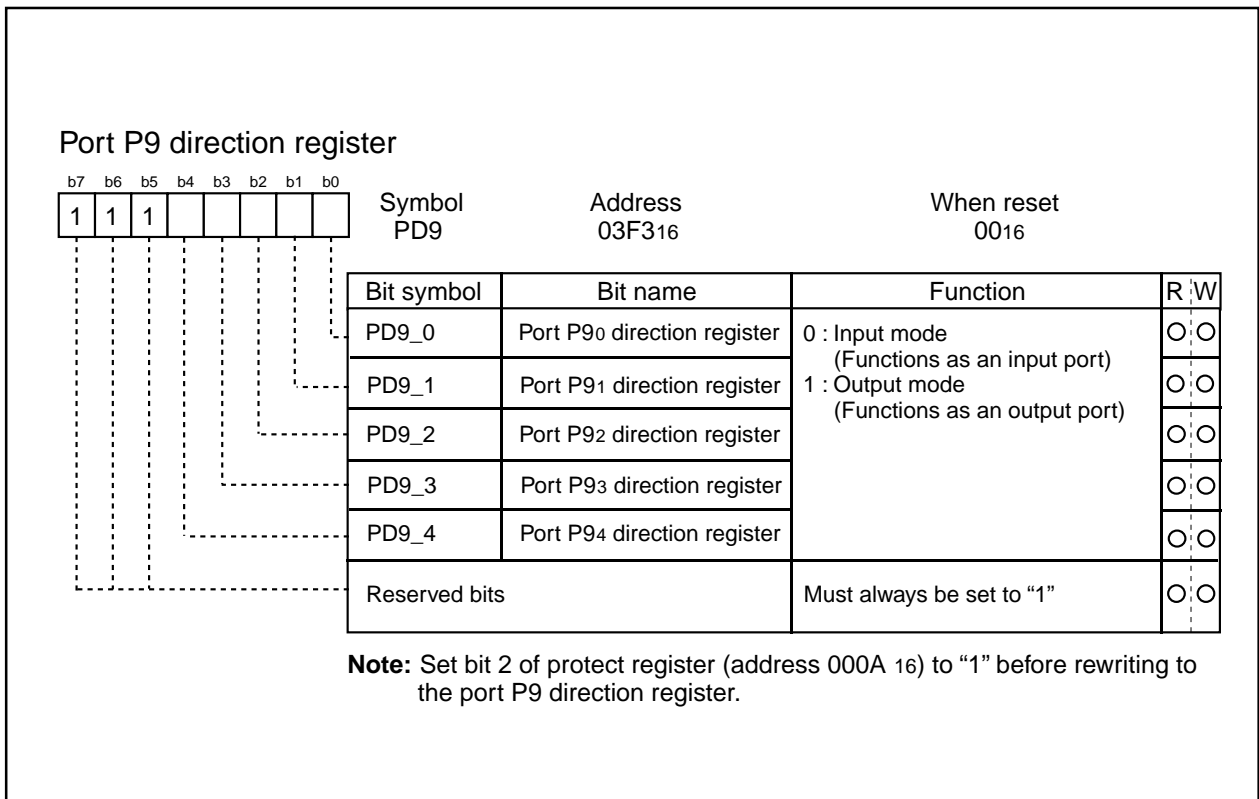


Figure 2.17.9 Port P9 direction register

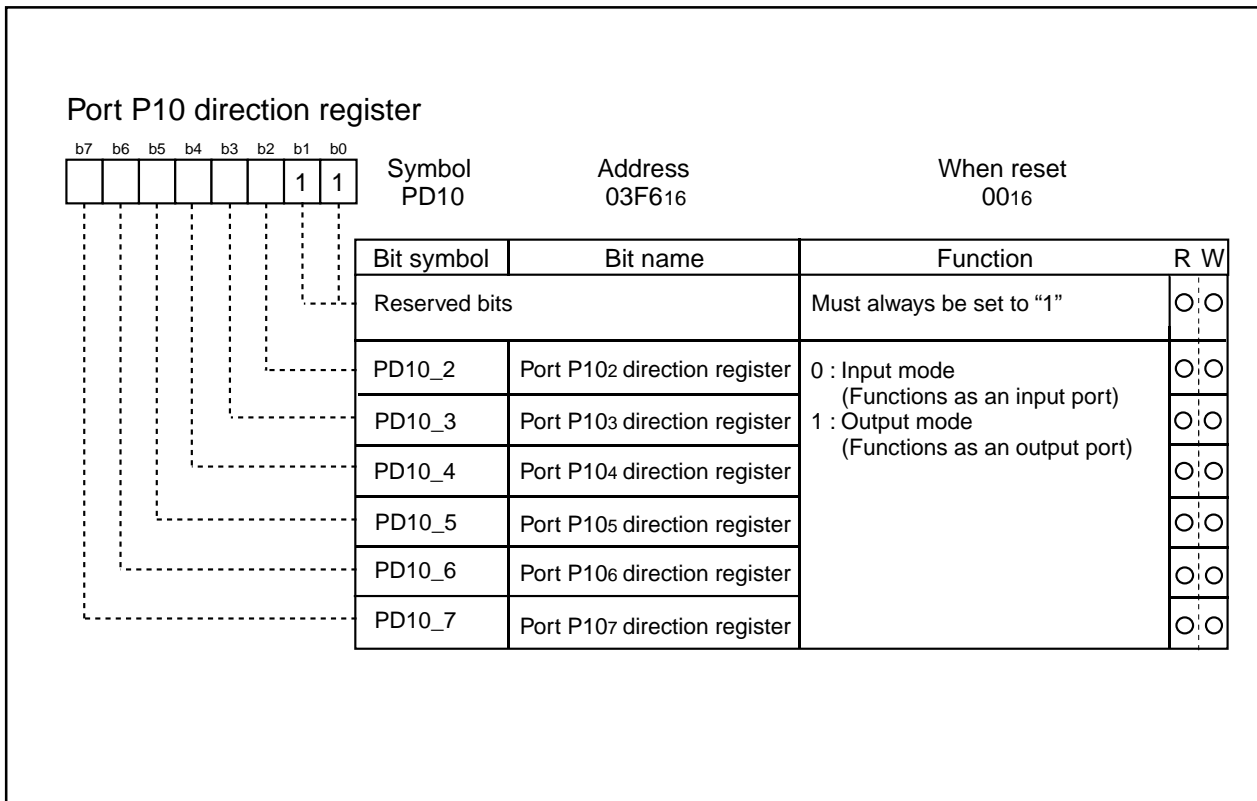


Figure 2.17.10 Port P10 direction register

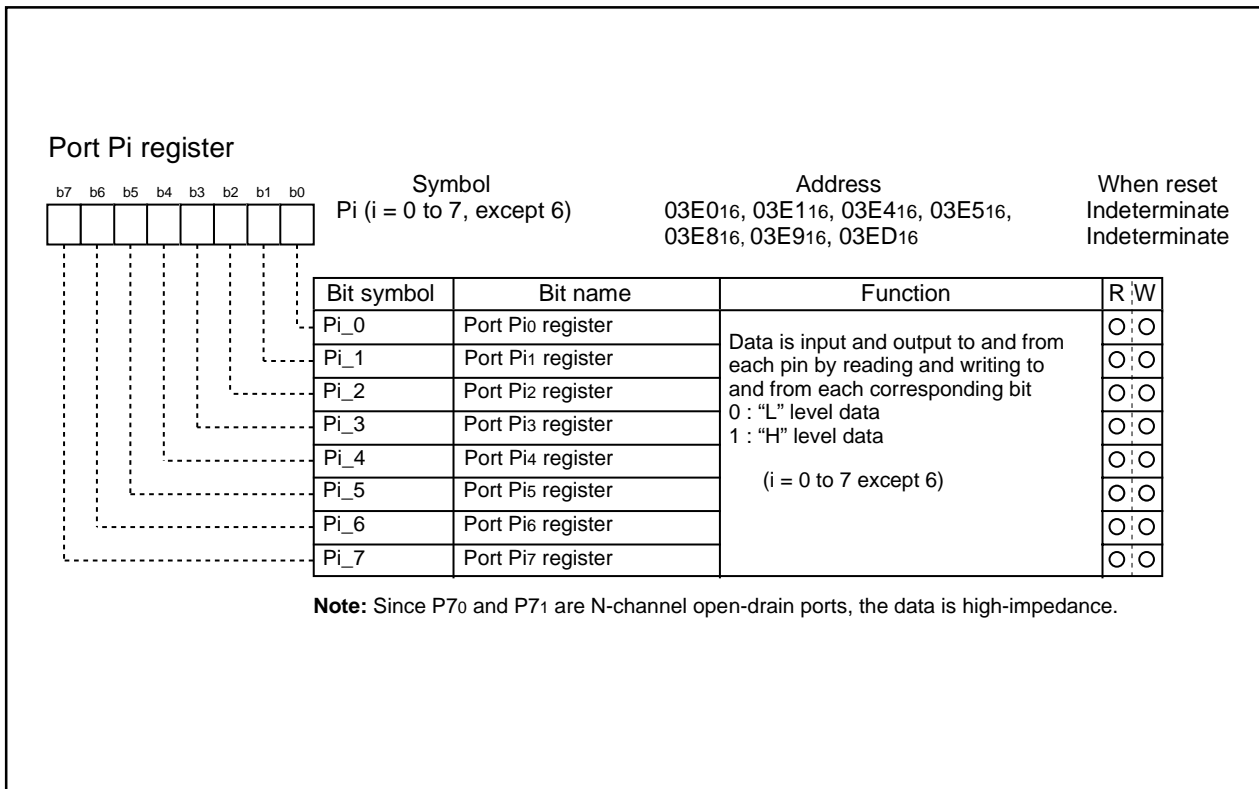


Figure 2.17.11 Port Pi register (i = 0 to 7, except 6)

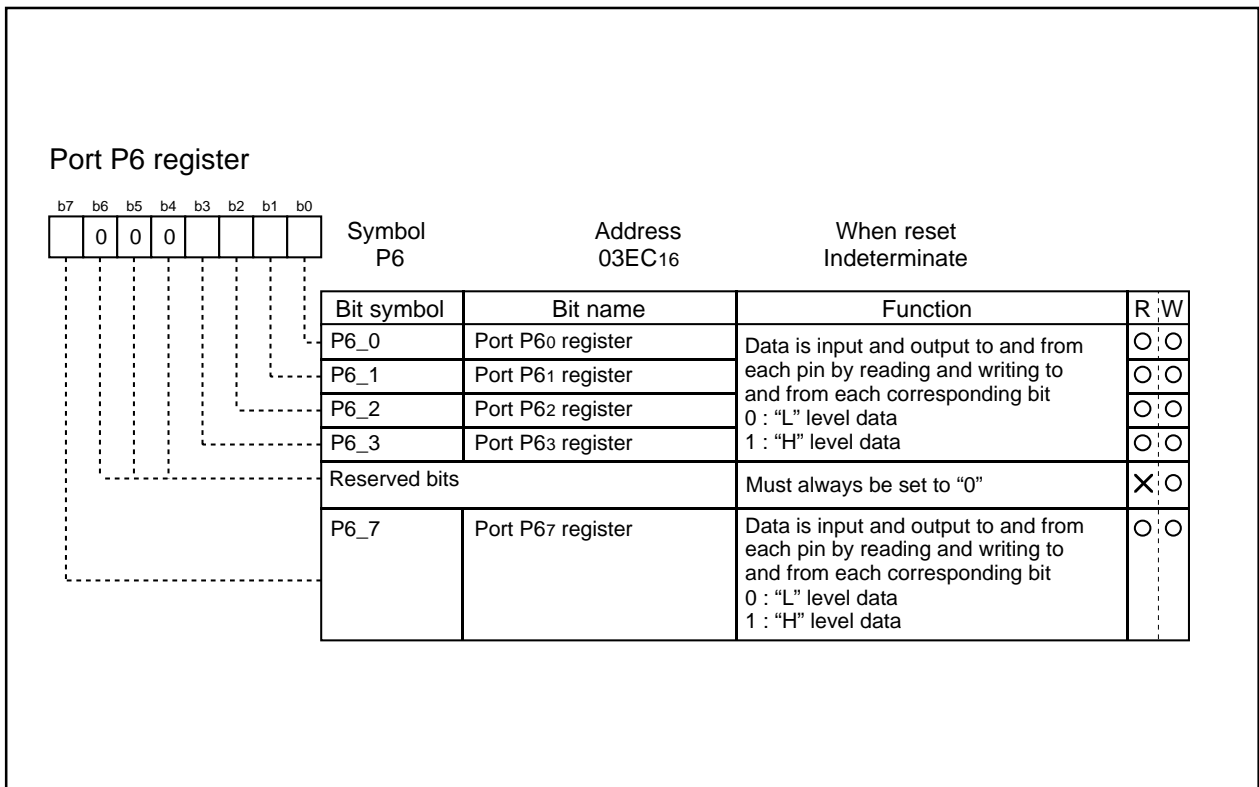


Figure 2.17.12 Port P6 register



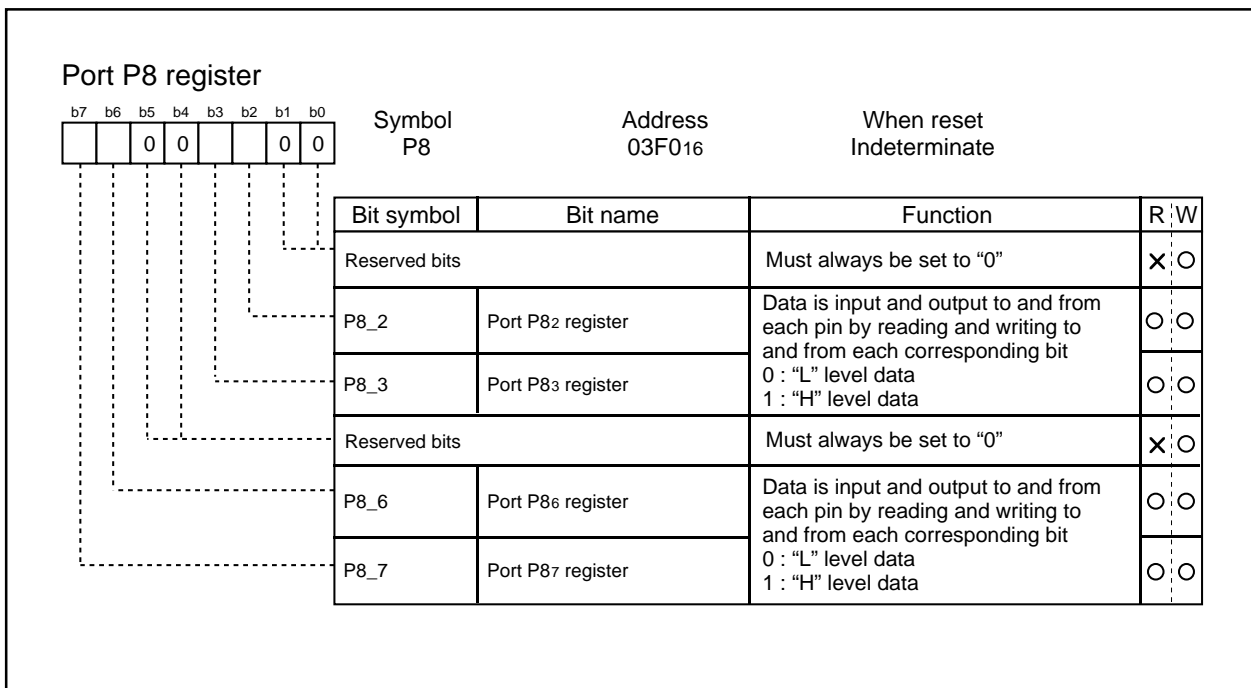


Figure 2.17.13 Port P8 register

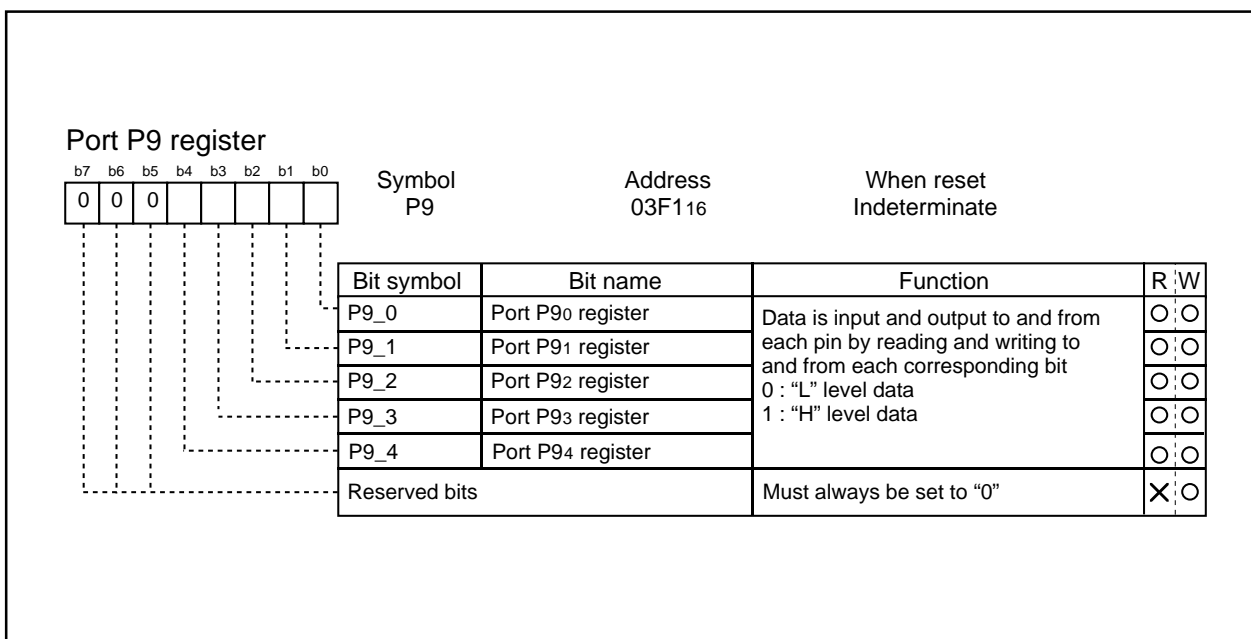


Figure 2.17.14 Port P9 register 0

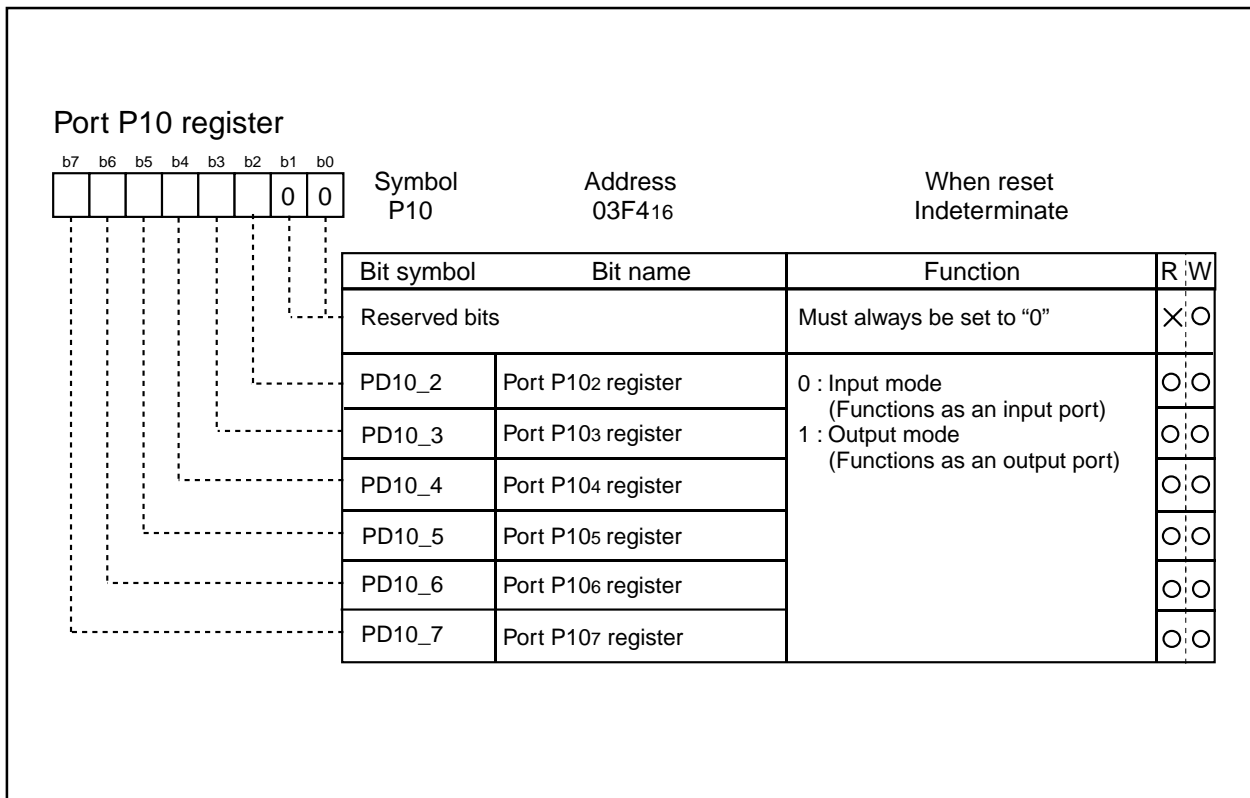


Figure 2.17.15 Port P10 register

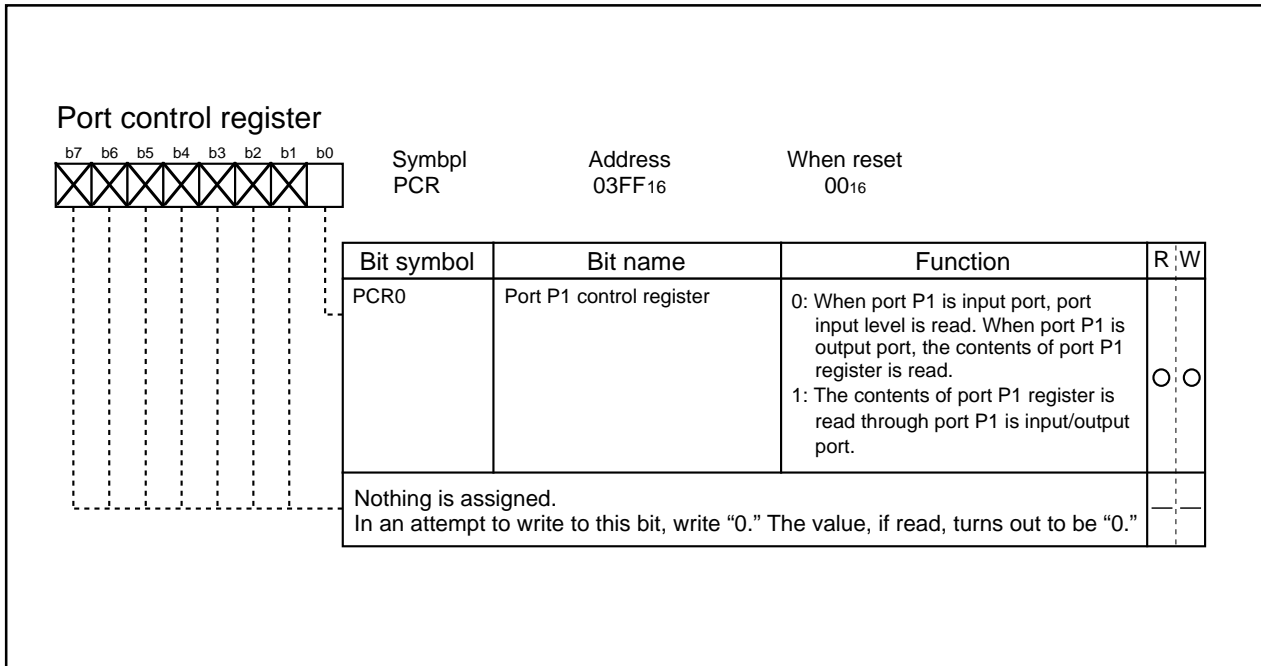


Figure 2.17.16 Port control register

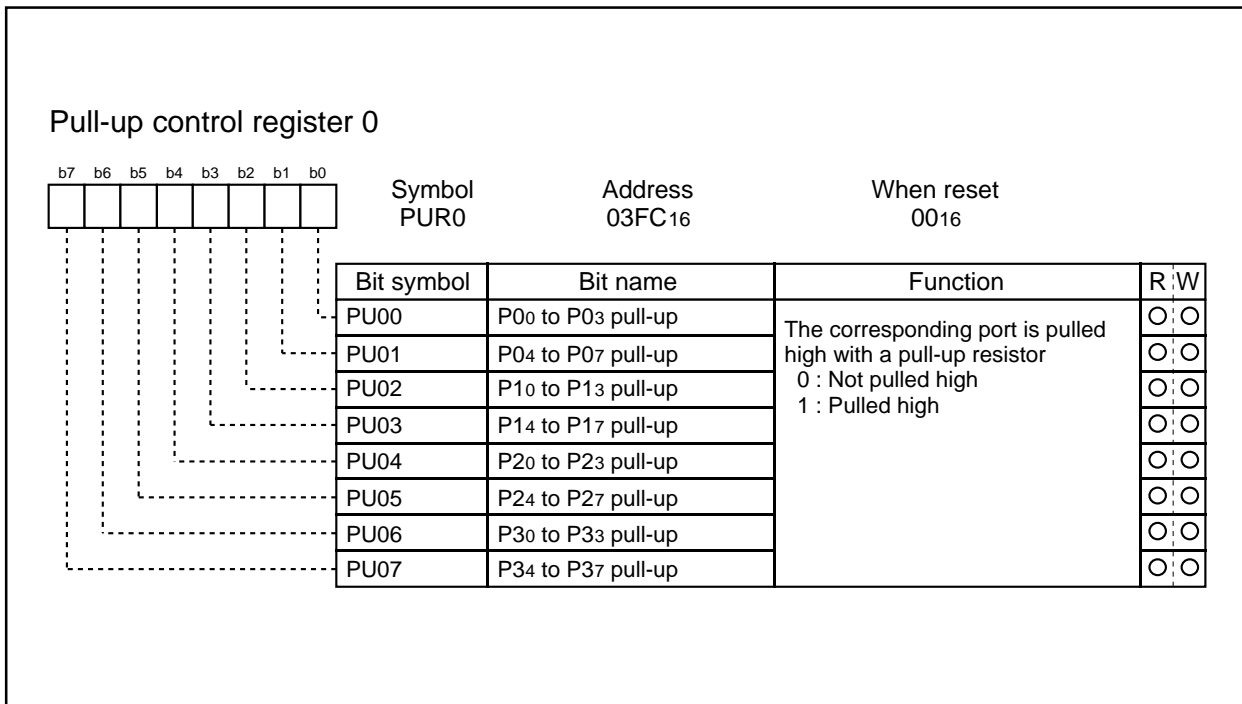


Figure 2.17.17 Pull-up control register 0

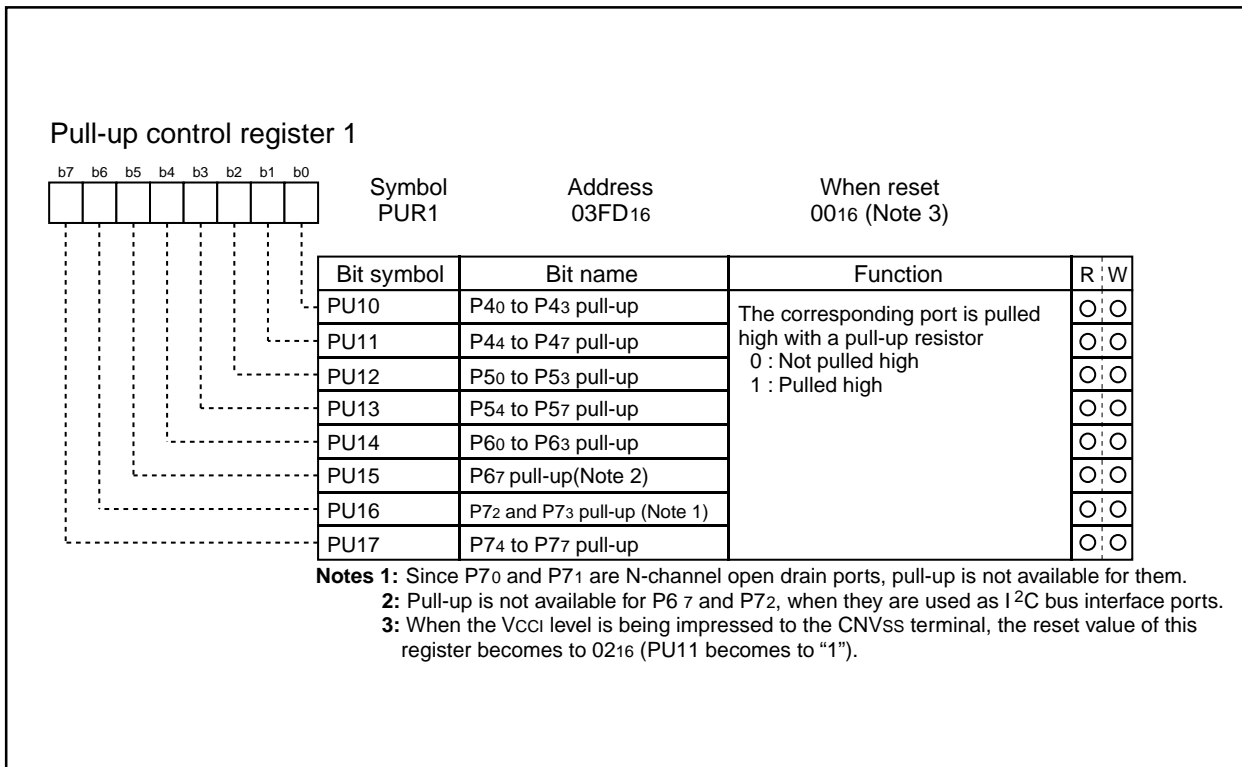


Figure 2.17.18 Pull-up control register 1

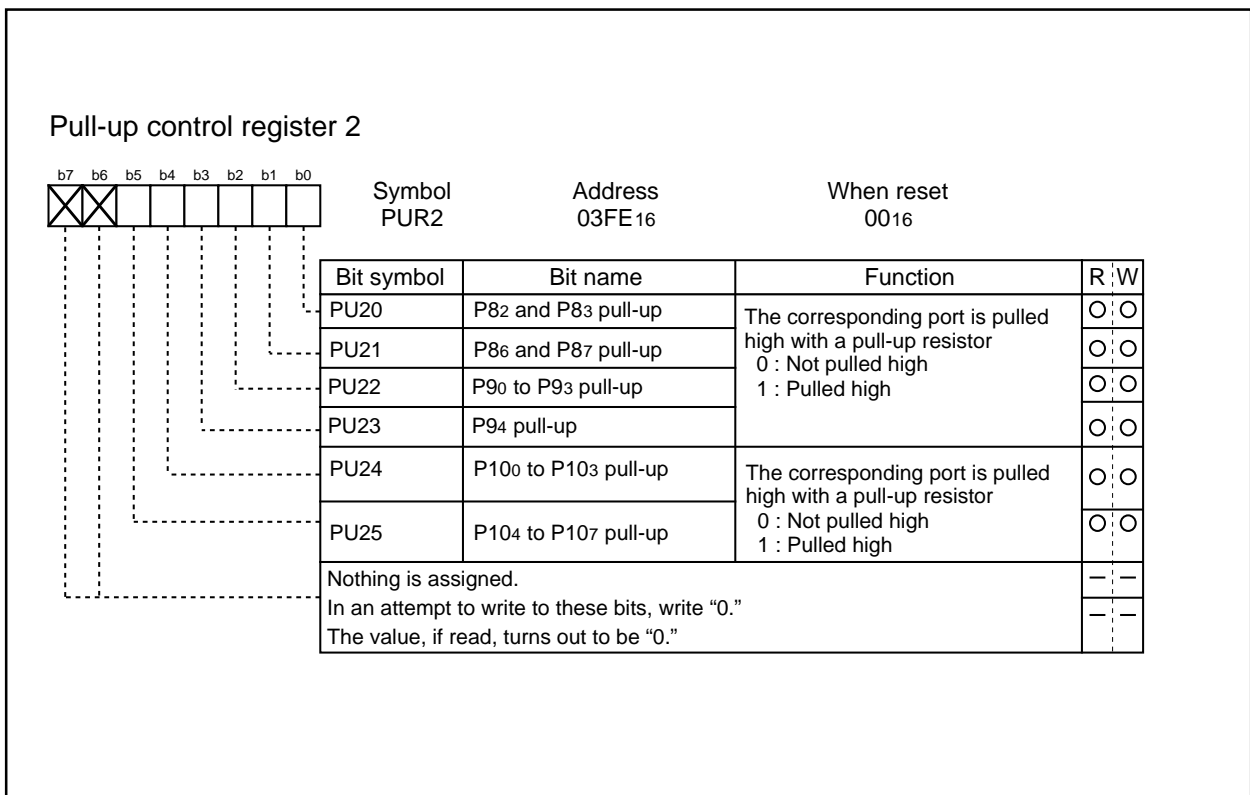


Figure 2.17.19 Pull-up control register 2

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

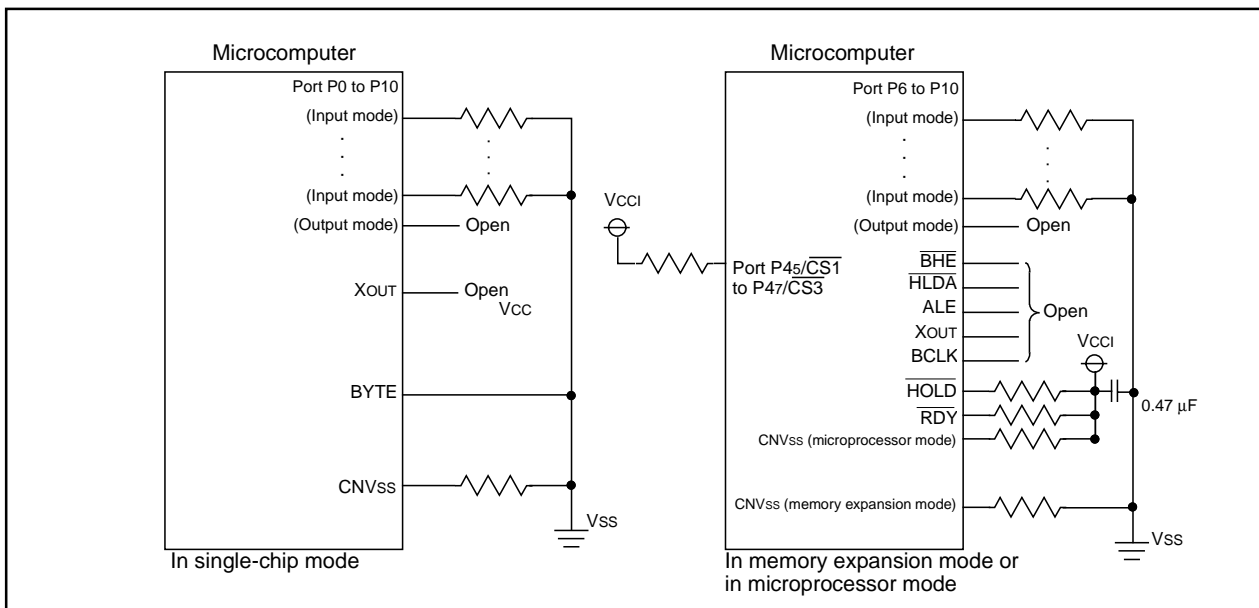
Pin name	Connection
Ports P0 to P10	After setting for input mode, connect every pin to Vss or Vcc via a resistor; or after setting for output mode, leave these pins open.
XOUT (Note)	Open
BYTE	Connect to Vss
CNVss	Connect via resistor to Vss (pull-down)

**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 VccI via resistors (pull-up).
$\overline{BHE}$ , $\overline{ALE}$ , $\overline{HLDA}$ , XOUT(Note), BCLK	Open
$\overline{HOLD}$ , $\overline{RDY}$	Connect via resistor to VccI (pull-up)
CNVss	Connect via resistor to Vss (pull-down) in the memory expansion mode. Connect via resistor to VccI (pull-up) in the microprocessor mode.

**Note:** With external clock input to XIN pin.



**Figure 2.17.20 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 "FFFF<sub>16</sub>". Reading the timer Ai register after setting a value in the timer Ai register with a count halted but before the counter starts counting gets a proper value.

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

#### 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  $\mu$ 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) When operating in low speed or low power consumption mode, do not go to wait mode by setting the peripheral function clock stop bit (CM02) to 1.
- (4) When shift to wait mode and stop mode, set the OSD control bit OC10 of the OSD control register 1 as "0."

### 3.9 Interrupts

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

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

The interrupt request bit of the certain interrupt written in address 00000<sub>16</sub> will then be set to "0". Reading address 00000<sub>16</sub> by software sets enabled highest priority interrupt source request bit to "0".

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

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

#### (2) Setting the stack pointer

- The value of the stack pointer immediately after reset is initialized to 0000<sub>16</sub>. Accepting an interrupt before setting a value in the stack pointer may become a factor of runaway. Be sure to set a value in the stack pointer before accepting an interrupt.

#### (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



**(5) Notes**

When clearing interrupt request bit of interrupt control register, depending on the instruction to be used, it interrupts and interrupt request bit may not be cleared. Use MOV command, when clear interrupt request bit, and change interrupt control register. When change interrupt control register in M16C/60 series and M16C/20 series, interrupt control register be sure to change in the part which corresponding interrupt request does not generate, and change interrupt control register after changing interruption into a prohibition state.

The example of a program which clears interrupt request bit in M16C/60 series.

Example 1: The case where interrupt control register is rewritten with a immediate value

```
FCLR    I                ; Interrupt is forbidden
MOV.B   #00H,0055H      ; Timer A0 interrupt request bit clear
MOV.W   MEM,R0          ; Dummy read
FSET    I                ; Interrupt is permitted
```

Example 2: The case where only interrupt request bit is cleared

```
FCLR    I                ; Interrupt is forbidden
MOV.B   0055H,R0L       ; Timer A0 interrupt control register read-out
AND.B   #0F7H,R0L       ; Only timer A0 interruption request bit is clear
MOV.B   R0L,0055H       ; Timer A0 interrupt control register writing
MOV.W   MEM,R0          ; Dummy read
FSET    I                ; Interrupt is permitted
```

There is a dummy read in Example 1 and Example 2 for preventing the set of interrupt permission flag (I flag) interrupting under the influence of command cue, and performing before the writing of interrupt control register.

**3.10 About Flash memory version and mask ROM**

Characteristic value, margin of operation, etc. of versions with built-in Flash memory, and built-in mask ROM may differ from each other within the limits of an electrical characteristics by manufacture process, built-in ROM, difference of a layout pattern, etc.

Carry out and check an examination equivalent to the system evaluation examination carried out with the version with built-in Flash memory at the time of a change for a version with built-in mask ROM.

#### **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

## 5. ELECTRICAL CHARACTERISTICS

### 5.1. Absolute Maximum Ratings

**Table 5.1.1 Absolute maximum ratings**

Symbol	Parameter	Condition	Rated value	Unit
V <sub>CCI</sub>	Internal logic supply voltage		-0.3 to 4.0	V
V <sub>CCE</sub>	External I/O buffer voltage(P93/P94/P72/P71/P70/P67)		-0.3 to 6.0	V
V <sub>I1</sub>	Input 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-P92, P102 to P107, X <sub>IN</sub> , OSC1, $\overline{\text{RESET}}$ , CNV <sub>SS</sub> , BYTE, Hsync, Vsync		-0.3 to V <sub>CCI</sub> +0.3	V
V <sub>I2</sub>	P6, P70, P71, P72/P93, P94 (Note)		-0.3 to V <sub>CCE</sub> +0.3	V
V <sub>I3</sub>	TVSETB		-0.3 to 0.3	V
V <sub>O1</sub>	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 P92, P102 to P107, R, G, B, OUT1, OUT2, OSC2, Xout		-0.3 to V <sub>CCI</sub> +0.3	V
V <sub>O2</sub>	P67, P70, P71, P72, P93, P94		-0.3 to V <sub>CCE</sub> +0.3	V
P <sub>d</sub>	Power dissipation (In the single chip mode)	T <sub>a</sub> =25°C	415	mW
T <sub>opr</sub>	Operating ambient temperature		-20 to 70	°C
T <sub>stg</sub>	Storage temperature		-40 to 125	°C

Note. When using P93 and P94 as DA pin, V<sub>I2</sub> is set to -0.3 to V<sub>CCI</sub> + 0.3(V).

## 5.2 Recommended Operating Conditions

**Table 5.2.1 Recommended operating conditions (referenced to  $V_{CCI}=3.3V\pm 0.15V$ ,  $V_{CCE}=5.0V\pm 0.25V$ ,  $T_a = -20\text{ }^\circ\text{C}$  to  $70\text{ }^\circ\text{C}$  unless otherwise specified)**

Symbol	Parameter		Standard			Unit
			Min	Typ.	Max.	
$V_{CCI}$	Internal logic supply voltage (Note 3)		3.15	3.30	3.45	V
$V_{CCE}$	External I/O buffer voltage (Note 3) P93, P94, P72, P71, P70, P67		4.75	5.00	5.25	V
$V_{SS}$	Supply voltage			0		V
$V_{IH}$	HIGH Input voltage	P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P63, P73 to P77, P82, P83, P86, P87, P90 to P92, P102 to P107, X <sub>IN</sub> , OSC1, RESET, CNV <sub>SS</sub> , BYTE, Hsync, Vsync, X <sub>CIN</sub>	0.8V <sub>CCI</sub>		V <sub>CCI</sub>	V
$V_{IH}$	HIGH Input voltage	P67, P70, P71, P72, P93, P94	0.8V <sub>CCE</sub>		V <sub>CCE</sub>	V
$V_{IH}$	HIGH Input voltage	P00 to P07, P10 to P17 (In the single-chip mode)	0.8V <sub>CCI</sub>		V <sub>CCI</sub>	V
$V_{IH}$	HIGH Input voltage	P00 to P07, P10 to P17 (data input function during memory expansion and microprocessor modes)	0.5V <sub>CCI</sub>		V <sub>CCI</sub>	V
$V_{IL}$	LOW Input voltage	P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P63, P73 to P77, P82, P83, P86, P87, P90 to P92, P102 to P107, X <sub>IN</sub> , OSC1, RESET, CNV <sub>SS</sub> , BYTE, Hsync, Vsync, X <sub>CIN</sub>	0		0.2V <sub>CCI</sub>	V
$V_{IL}$	LOW Input voltage	P67, P70, P71, P72, P93, P94	0		0.2V <sub>CCE</sub>	V
$V_{IL}$	LOW Input voltage	P00 to P07, P10 to P17 (In the single-chip mode)	0		0.2V <sub>CCI</sub>	V
$V_{IL}$	LOW Input voltage	P00 to P07, P10 to P17 (data input function during memory expansion and microprocessor modes)	0		0.16V <sub>CCI</sub>	V
I <sub>OH</sub> (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, P67, P72 to P77, P82, P83, P86, P87, P90 to P94, P102 to P107, R, G, B, OUT1, OUT2			-10.0	mA
I <sub>OH</sub> (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, P102 to P107, R, G, B, OUT1, OUT2			-5.0	mA
I <sub>OL</sub> (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, P67, P70 to P77, P82, P83, P86, P87, P90 to P94, P102 to P107, R, G, B, OUT1, OUT2			10.0	mA
I <sub>OL</sub> (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, P73 to P77, P82, P83, P86, P87, P90 to P92, P102 to P107, R, G, B, OUT1, OUT2			5.0	mA
I <sub>OL</sub> (avg)	LOW average output current	P67, P70 to P72, P93, P94			6.0	mA
f (X <sub>IN</sub> )	Main clock input oscillation frequency				16.1	MHz
f (X <sub>CIN</sub> )	Sub-clock oscillation frequency			32.768	50.0	kHz
f <sub>osc</sub>	Oscillation frequency (for OSD) OSC1	LC oscillating mode	7.9		30.1	MHz
		Ceramic oscillating mode	14.9		30.1	
		Internal oscillation mode (X <sub>IN</sub> =10 MHz)	19.9		31.1	
		Internal oscillation mode (X <sub>IN</sub> =16 MHz)	19.9		40.1	
f <sub>CVIN</sub>	Input frequency	Horizontal sync. signal of 525i video signal	15.262	15.734	16.206	kHz
		Horizontal sync. signal of 525p video signal	–	31.47	–	
V <sub>I</sub>	Input amplitude video signal	CV <sub>IN1</sub> , CV <sub>IN2</sub>	1.5	1.75	2.00	V

**Notes1:** The mean output current is the mean value within 100 ms.

**2:** The total I<sub>OH</sub> (peak) must be 80 mA max.

**3:** Connect 0.1 μF or more capacitor externally between the power source pins V<sub>CCI</sub>-V<sub>SS</sub>, V<sub>CCI</sub>-CNV<sub>SS</sub>, V<sub>CCI</sub>-TVSETB, and V<sub>CCE</sub>-V<sub>SS</sub> so as to reduce power source noise.

### 5.3 Electrical Characteristics

**Table 5.3.1 Electrical characteristics (referenced to  $V_{CCI}=3.3V$ ,  $V_{CCE}=5.0V$ ,  $V_{SS} = 0V$  at  $T_a = 25^\circ C$ ,  $f(X_{IN}) = 16\text{ MHz}$  unless otherwise specified)**

Symbol	Parameter	Measuring condition	Standard			Unit		
			Min.	Typ.	Max.			
$V_{OH}$	HIGH output voltage P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to 47, P50 to P57, P60 to P63, P73 to P77, P82, P83, P86, P87, P90 to P92, P102 to P107, R, G, B, OUT1, OUT2	$I_{OH} = -5\text{mA}$	$V_{CCI}-1.5$ (Note 1)			V		
$V_{OH}$	HIGH output voltage P67, P72, P93, P94	$I_{OH} = -5\text{mA}$	$V_{CCE}-1.5$ (Note 2)			V		
$V_{OL}$	LOW output voltage P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to 47, P50 to P57, P60 to P63, P73 to P77, P82, P83, P86, P87, P90 to P92, P102 to P107, R, G, B, OUT1, OUT2	$I_{OL} = 5\text{mA}$			1.5 (Note 3)	V		
$V_{OL}$	LOW output voltage P67, P70, P71, P72, P93, P94	$I_{OL} = 6\text{mA}$			0.6 (Note 4)	V		
$V_{T+}-V_{T-}$	Hysteresis $\overline{HOLD}$ , $\overline{RDY}$ , $\overline{TB0IN}$ to $\overline{TB2IN}$ , $\overline{INT0}$ , $\overline{INT1}$ , $\overline{CTS0}$ , $\overline{CTS2}$ , $\overline{CLK0}$ , $\overline{RxD0}$ , $\overline{Hsync}$ , $\overline{Vsync}$ , $\overline{HC0}$ , $\overline{HC1}$ , $X_{IN}$		0.2		0.6	V		
$V_{T+}-V_{T-}$	Hysteresis $\overline{SCL1}$ , $\overline{SCL2}$ , $\overline{SCL3}$ , $\overline{CLK2}$ , $\overline{SDA1}$ , $\overline{SDA2}$ , $\overline{SDA3}$ , $\overline{RxD2}$		0.2		0.8	V		
$V_{T+}-V_{T-}$	Hysteresis $\overline{RESET}$		0.2		1.2	V		
$I_{IH}$	HIGH input current P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to 47, P50 to P57, P60 to P63, P73 to P77, P82, P83, P86, P87, P90 to P92, P102 to P107, $X_{IN}$ , $\overline{RESET}$ , $\overline{CNVss}$ , $\overline{BYTE}$ , $\overline{OSC1}$ , $\overline{Hsync}$ , $\overline{Vsync}$	$V_I = 3.3V$			10.0	$\mu\text{A}$		
$I_{IH}$	HIGH input current P67, P70, P71, P72, P93, P94	$V_I = 5.0V$			10.0	$\mu\text{A}$		
$I_{IL}$	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, P102 to P107, $X_{IN}$ , $\overline{RESET}$ , $\overline{CNVss}$ , $\overline{BYTE}$ , $\overline{OSC1}$ , $\overline{Hsync}$ , $\overline{Vsync}$	$V_I = 0V$			-10.0	$\mu\text{A}$		
$R_{PULLUP}$	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, P102 to P107,	$V_I = 0V$	30.0	50.0	167.0	$\text{k}\Omega$		
$I_{CC}$	Power supply current	In single-chip mode, the output pins are open and other pins are $V_{SS}$	$f(X_{IN}) = 16\text{ MHz}$ Square wave, no division	OSD (40MHz) ON, Data slicer ON	90	120	$\text{mA}$	
					OSD OFF, Data slicer OFF	40		60
			$f(X_{IN}) = 16\text{ MHz}$ Square wave, division by 8	OSD OFF, Data slicer OFF	15		$\text{mA}$	
			$f(X_{CIN}) = 32\text{ kHz}$ In the weight		100	400		$\mu\text{A}$
			$T_a = 25^\circ\text{C}$ when clock is stopped		50	200		$\mu\text{A}$
$T_a = 70^\circ\text{C}$ when clock is stopped	0.5	2	$\text{mA}$					
$R_{BS}$	$I^2C$ -BUS • BUS switch connection resistor (between $\overline{SCL1}$ and $\overline{SCL2}$ , $\overline{SDA1}$ and $\overline{SDA2}$ )	$V_{CCE} = 5.0V$			130	$\Omega$		
$R_{IXIN}$	Feedback resistor $X_{IN}$				3.0	$\text{M}\Omega$		
$R_{IXCIN}$	Feedback resistor $X_{CIN}$				6.0	$\text{M}\Omega$		

**Notes1:** The minimum value standard for every  $V_{OH}$  serves as a straight line which connected ( $I_{OH} = -5\text{ mA}$ ,  $V_{OH} = V_{CCI}-1.5\text{ V}$ ) and ( $I_{OH} = 0\text{ mA}$ ,  $V_{OH} = V_{CCI}$ ).  
**2:** The minimum value standard for every  $V_{OH}$  serves as a straight line which connected ( $I_{OH} = -5\text{ mA}$ ,  $V_{OH} = V_{CCE}-1.5\text{ V}$ ) and ( $I_{OH} = 0\text{ mA}$ ,  $V_{OH} = V_{CCE}$ ).  
**3:** The maximum value standard for every  $V_{OL}$  serves as a straight line which connected ( $I_{OH} = 5\text{ mA}$ ,  $V_{OL} = 1.5\text{ V}$ ) and ( $I_{OL} = 0\text{ mA}$ ,  $V_{OL} = 0.0\text{ V}$ ).  
**4:** The maximum value standard for every  $V_{OL}$  serves as a straight line which connected ( $I_{OH} = 6\text{ mA}$ ,  $V_{OL} = 0.6\text{ V}$ ) and ( $I_{OL} = 0\text{ mA}$ ,  $V_{OL} = 0.0\text{ V}$ ).

## 5.4 A-D Conversion Characteristics

**Table 5.4.1 A-D conversion characteristics (referenced to  $V_{CC1} = 3.3V$ ,  $V_{SS} = 0V$  at  $T_a = 25\text{ }^\circ\text{C}$ ,  $f(X_{IN}) = 16\text{ MHz}$  unless otherwise specified)**

Symbol	Parameter		Measuring condition	Standard			Unit
				Min.	Typ.	Max.	
—	Resolution		$V_{REF} = V_{CC1}$			8	Bits
—	Absolute accuracy	Sample & hold function not available	$V_{REF} = V_{CC1} = 3.3\text{ V}$			$\pm 5$	LSB
		Sample & hold function available (8 bit)	$V_{REF} = V_{CC1} = 3.3\text{ V}$			$\pm 5$	LSB
$t_{CONV}$	Conversion time			2.8			$\mu\text{s}$
$t_{SAMP}$	Sampling time			0.3			$\mu\text{s}$
$V_{REF}$	Reference voltage				$V_{CC1}$		V
$V_{IA}$	Analog input voltage			0		$V_{CC1}$	V

## 5.5 D-A Conversion Characteristics

**Table 5.5.1 D-A conversion characteristics (referenced to  $V_{CC1}=3.3V$ ,  $V_{SS} = 0V$ , at  $T_a = 25\text{ }^\circ\text{C}$ ,  $f(X_{IN}) = 16\text{ MHz}$  unless otherwise specified)**

Symbol	Parameter		Measuring condition	Standard			Unit
				Min.	Typ.	Max.	
—	Resolution					8	Bits
—	Absolute accuracy					10	%
$t_{su}$	Setup time					3	$\mu\text{s}$
$R_o$	Output resistance			4	10	20	$\text{k}\Omega$

## 5.6 Analog R, G, B Output Characteristics

**Table 5.6.1 Analog R, G, B output characteristics ( $V_{CC1} = 3.3V$ ,  $V_{SS} = 0V$  at  $T_a = 25\text{ }^\circ\text{C}$ , Load register  $RI = \text{No}$ , Load capacity  $CI = \text{No}$ )**

Symbol	Parameter		Measuring condition	Standard			Unit
				Min.	Typ.	Max.	
$V_{ppm}$	Maximum output amplitude		RGB each output control bit =111b setting	0.9	1.0	1.2	V
$V_{oe}$	Output deviation					$\pm 20$	%
$I_o$	Maximum output supply voltage		RGB each output control bit =111b setting	2.2	4.0	5.8	mA
$R_o$	Output register			190		400	$\Omega$
$T_{st}$	Settling time		30 % $\rightarrow$ 70 % or 70 % $\rightarrow$ 30 %			33	ns

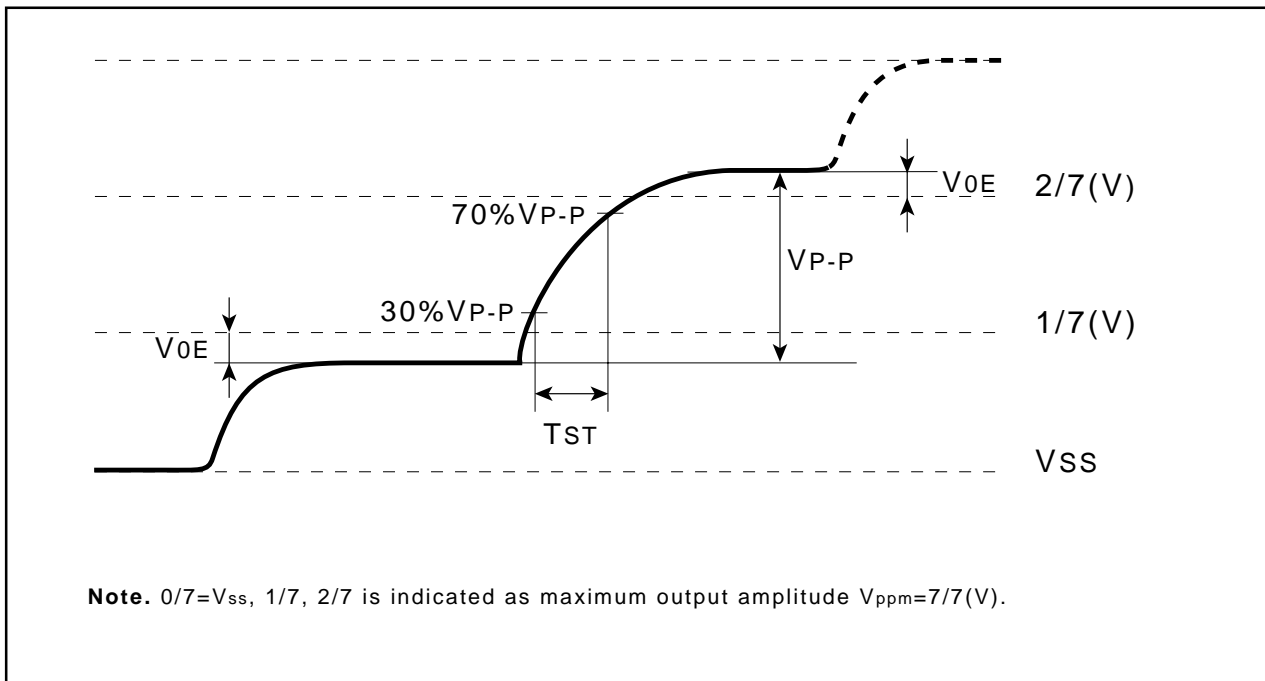


Figure 5.6.1 Analog R, G, B output characteristics

## 5.7 Timing Requirements

**Table 5.7.1 External clock input (referenced to VCCI=3.3V, VCC E=5.0V, VSS = 0 V at Ta = 25 °C unless otherwise specified)**

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

**Table 5.7.2 Memory expansion and microprocessor modes (referenced to VCCI=3.3V, VCC E=5.0V, VSS = 0 V at Ta = 25 °C unless otherwise specified)**

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

**Note.** According to BCLK frequency, it is computed in the following formula.

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

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



**Table 5.7.3 Timer B input (counter input in event counter mode)**(referenced to  $V_{CC1}=3.3V$ ,  $V_{SS} = 0 V$  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  $V_{CC1}=3.3V$ ,  $V_{SS} = 0 V$  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  $V_{CC1}=3.3V$ ,  $V_{SS} = 0 V$  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  $V_{CC1}=3.3V$ ,  $V_{CCE}=5.0V$ ,  $V_{SS} = 0 V$  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  $\overline{INTi}$  inputs (referenced to  $V_{CC1}=3.3V$ ,  $V_{SS} = 0 V$  at  $T_a = 25\text{ }^\circ\text{C}$  unless otherwise specified)**

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

## 5.8 Switching Characteristics

**Table 5.8.1 Memory expansion mode and microprocessor mode (no wait) (referenced to VCCI=3.3V, VCCE=5.0V, VSS = 0 V at Ta = 25 °C, CM15 = “1” unless otherwise specified)**

Symbol	Parameter	Measuring condition	Standard		Unit
			Min.	Max.	
t <sub>d</sub> (BCLK-AD)	Address output delay time	Figure 5.9.1		35	ns
t <sub>h</sub> (BCLK-AD)	Address output hold time (BCLK standard)		4		ns
t <sub>h</sub> (RD-AD)	Address output hold time (RD standard)		0		ns
t <sub>h</sub> (WR-AD)	Address output hold time (WR standard)		0		ns
t <sub>d</sub> (BCLK-CS)	Chip select output delay time			35	ns
t <sub>h</sub> (BCLK-CS)	Chip select output hold time (BCLK standard)		4		ns
t <sub>d</sub> (BCLK-RD)	RD signal output delay time			35	ns
t <sub>h</sub> (BCLK-RD)	RD signal output hold time		0		ns
t <sub>d</sub> (BCLK-WR)	WR signal output delay time			35	ns
t <sub>h</sub> (BCLK-WR)	WR signal output hold time		0		ns
t <sub>d</sub> (BCLK-DB)	Data output delay time (BCLK standard)			40	ns
t <sub>h</sub> (BCLK-DB)	Data output hold time (BCLK standard)		4		ns
t <sub>d</sub> (DB-WR)	Data output delay time (WR standard)		(Note1)		ns
t <sub>h</sub> (WR-DB)	Data output hold time (WR standard)(Note 2)		0		ns

**Notes1.** According to BCLK frequency, it is computed in the following formula.

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

**2.** This standard shows the timing which an output turns off and does not show the maintenance time of a data bus.

**Table 5.8.2 Memory expansion mode and microprocessor mode**

**(with wait, accessing external memory)**

**(referenced to V<sub>CCi</sub>=3.3V, V<sub>CCe</sub>=5.0V, V<sub>SS</sub> = 0 V at Ta = 25 °C, CM15 = “1” unless otherwise specified)**

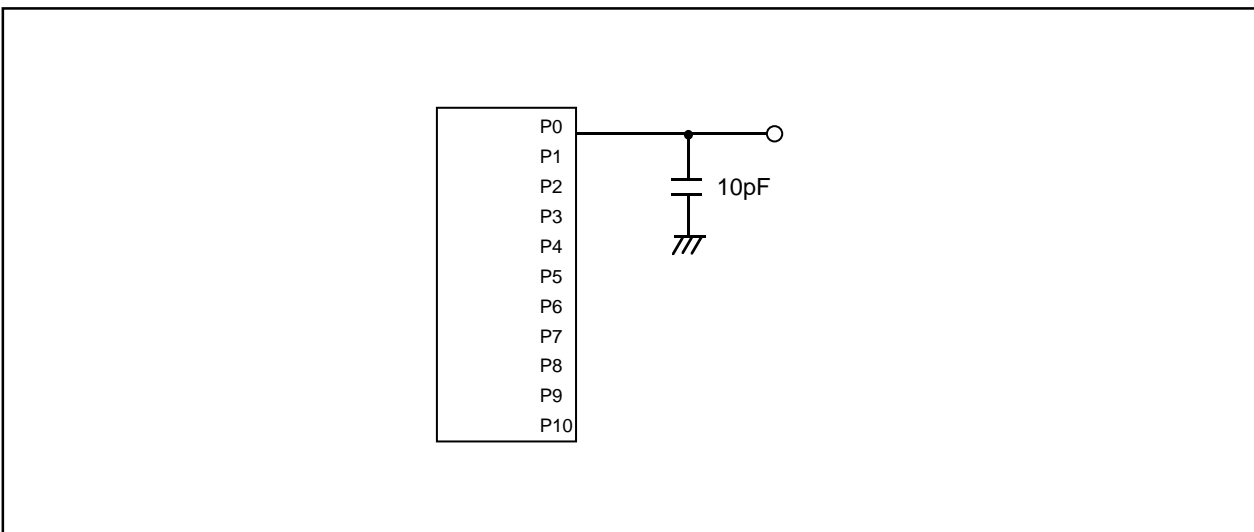
Symbol	Parameter	Measuring condition	Standard		Unit
			Min.	Max.	
t <sub>d</sub> (BCLK-AD)	Address output delay time	Figure 5.9.1		35	ns
t <sub>h</sub> (BCLK-AD)	Address output hold time (BCLK standard)		4		ns
t <sub>h</sub> (RD-AD)	Address output hold time (RD standard)		0		ns
t <sub>h</sub> (WR-AD)	Address output hold time (WR standard)		0		ns
t <sub>d</sub> (BCLK-CS)	Chip select output delay time			35	ns
t <sub>h</sub> (BCLK-CS)	Chip select output hold time (BCLK standard)		4		ns
t <sub>d</sub> (BCLK-RD)	RD signal output delay time			35	ns
t <sub>h</sub> (BCLK-RD)	RD signal output hold time		0		ns
t <sub>d</sub> (BCLK-WR)	WR signal output delay time			35	ns
t <sub>h</sub> (BCLK-WR)	WR signal output hold time		0		ns
t <sub>d</sub> (BCLK-DB)	Data output delay time (BCLK standard)			40	ns
t <sub>h</sub> (BCLK-DB)	Data output hold time (BCLK standard)		4		ns
t <sub>d</sub> (DB-WR)	Data output delay time (WR standard)		(Note1)		ns
t <sub>h</sub> (WR-DB)	Data output hold time (WR standard)		0		ns

**Notes1.** According to BCLK frequency, it is computed in the following formula.

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

2. This standard shows the timing which an output turns off and does not show the maintenance time of a data bus.

### 5.9 Measurement Circuit



**Figure 5.9.1 Port P0 to P10 measurement circuit**

### 5.10 Timing Diagram

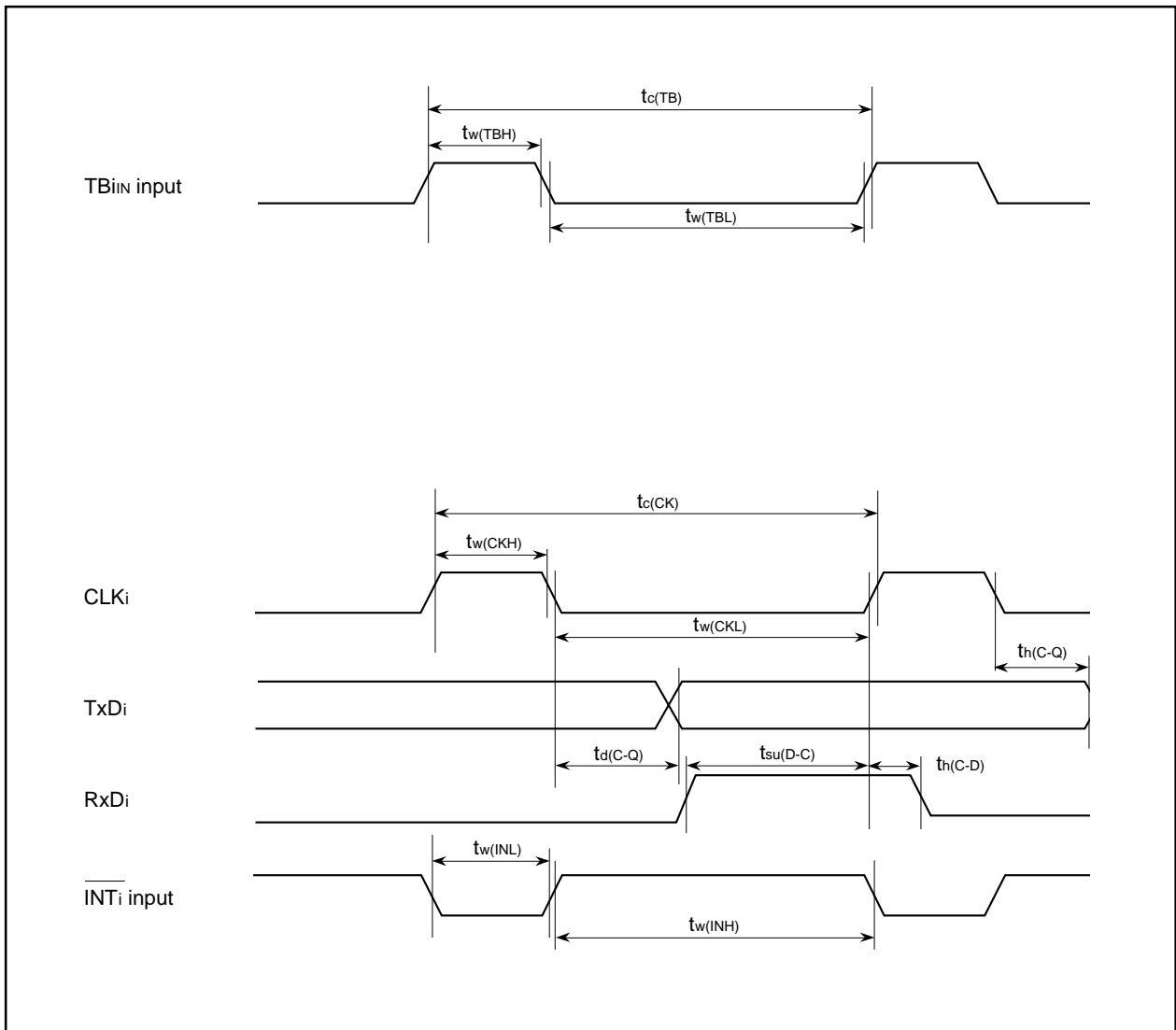


Figure 5.10.1 Timing diagram

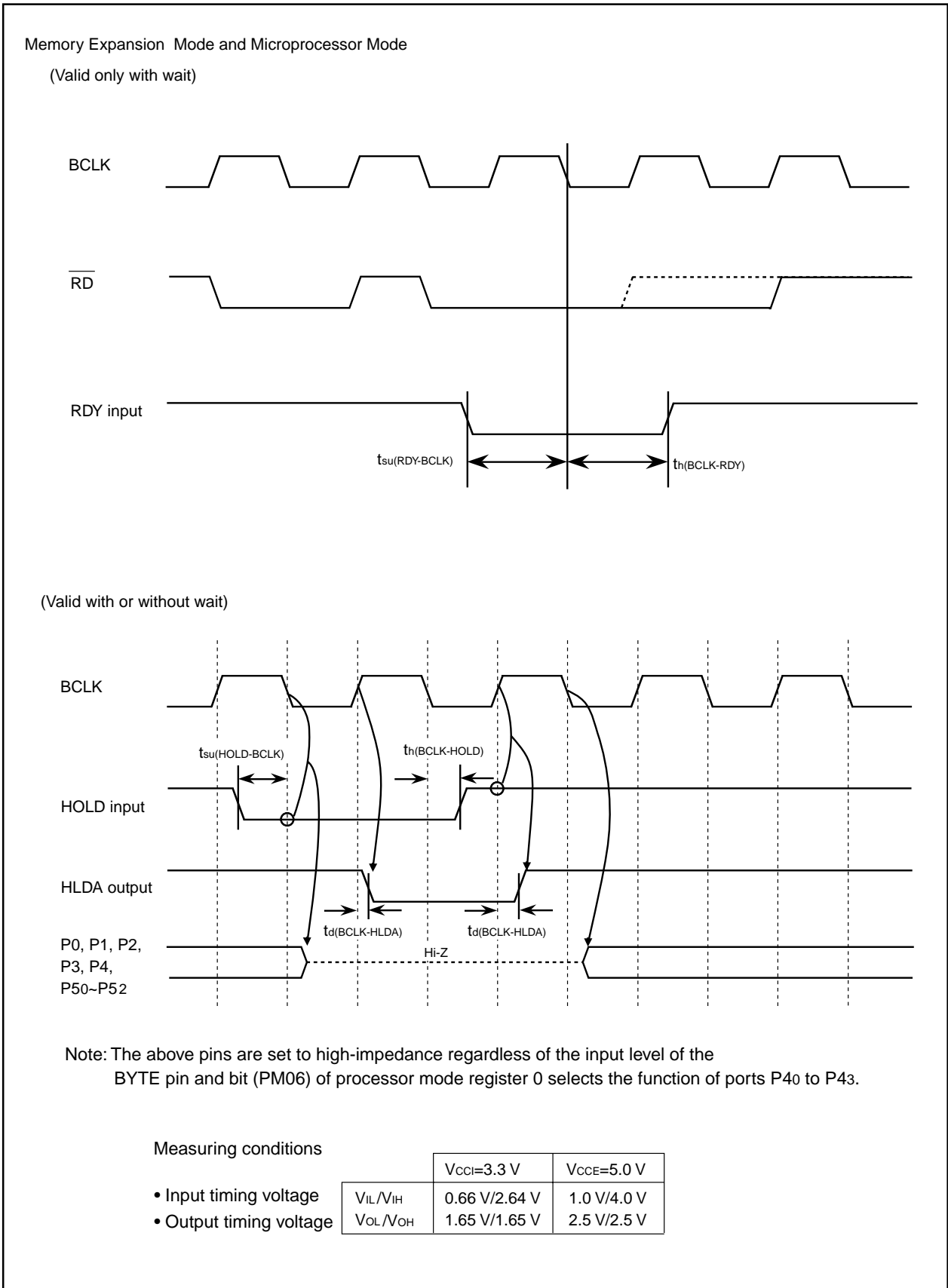


Figure 5.10.2 Timing diagram in memory expansion mode and microprocessor mode (1)

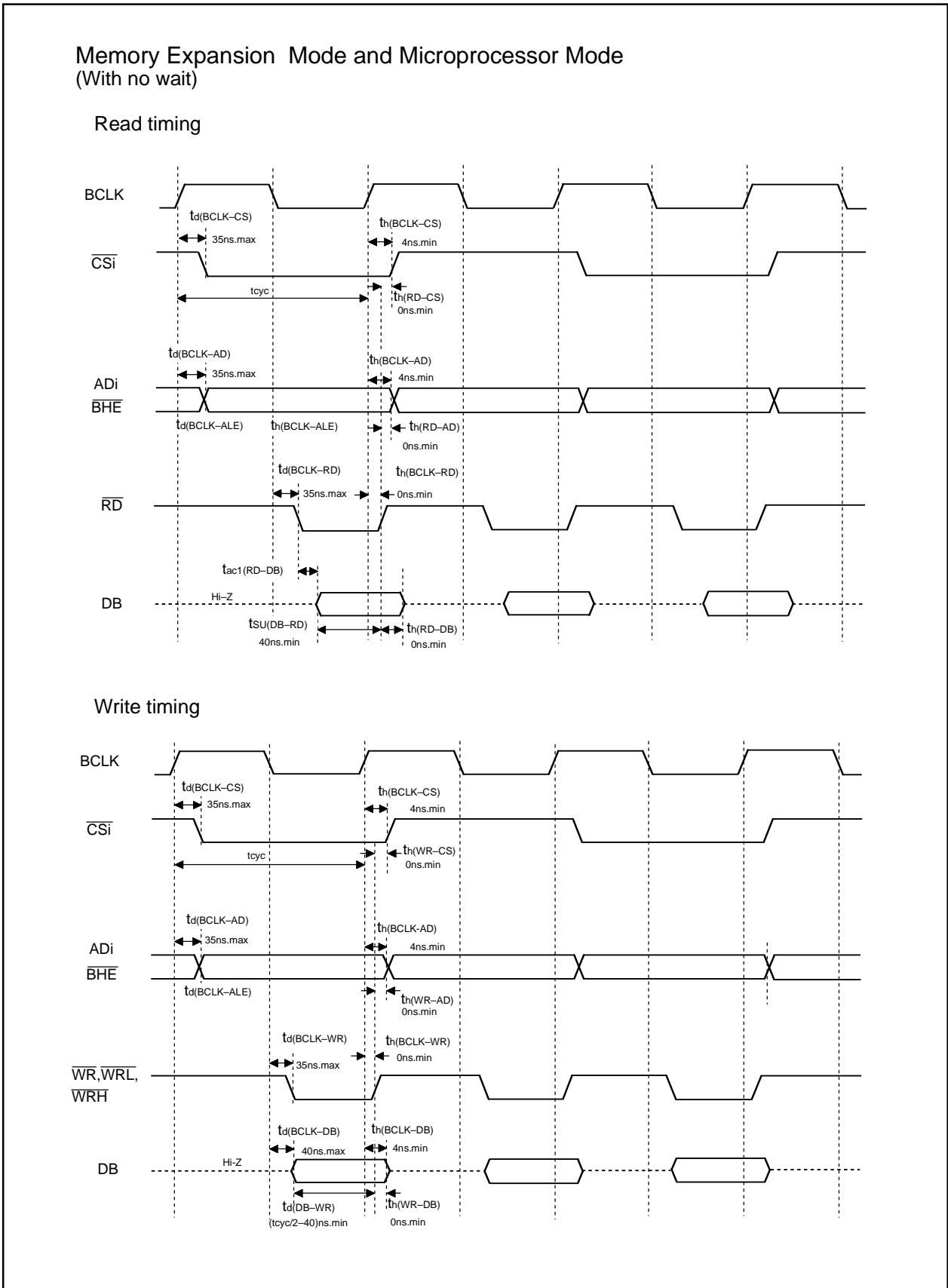
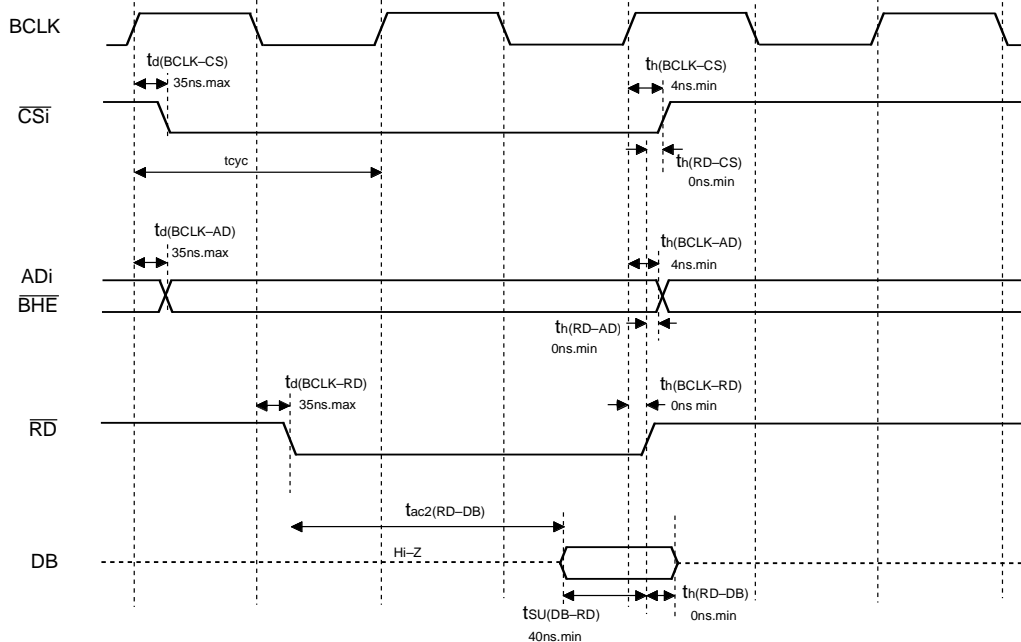


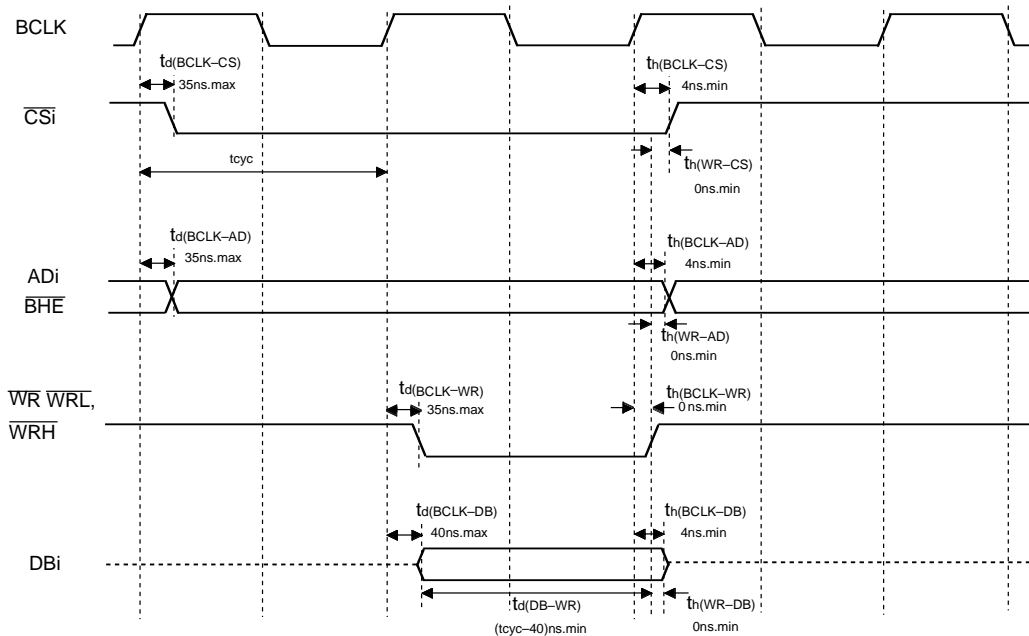
Figure 5.10.3 Timing diagram in memory expansion mode and microprocessor mode (2)

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

Read timing



Write timing



Measuring conditions

Input timing voltage  
Output timing voltage

	$V_{\text{CCI}}=3.3 \text{ V}$	$V_{\text{CCE}}=5.0 \text{ V}$
$V_{\text{IL}}/V_{\text{IH}}$	0.52 V/1.65 V	1.0 V/4.0 V
$V_{\text{OL}}/V_{\text{OH}}$	1.65 V	1.65 V

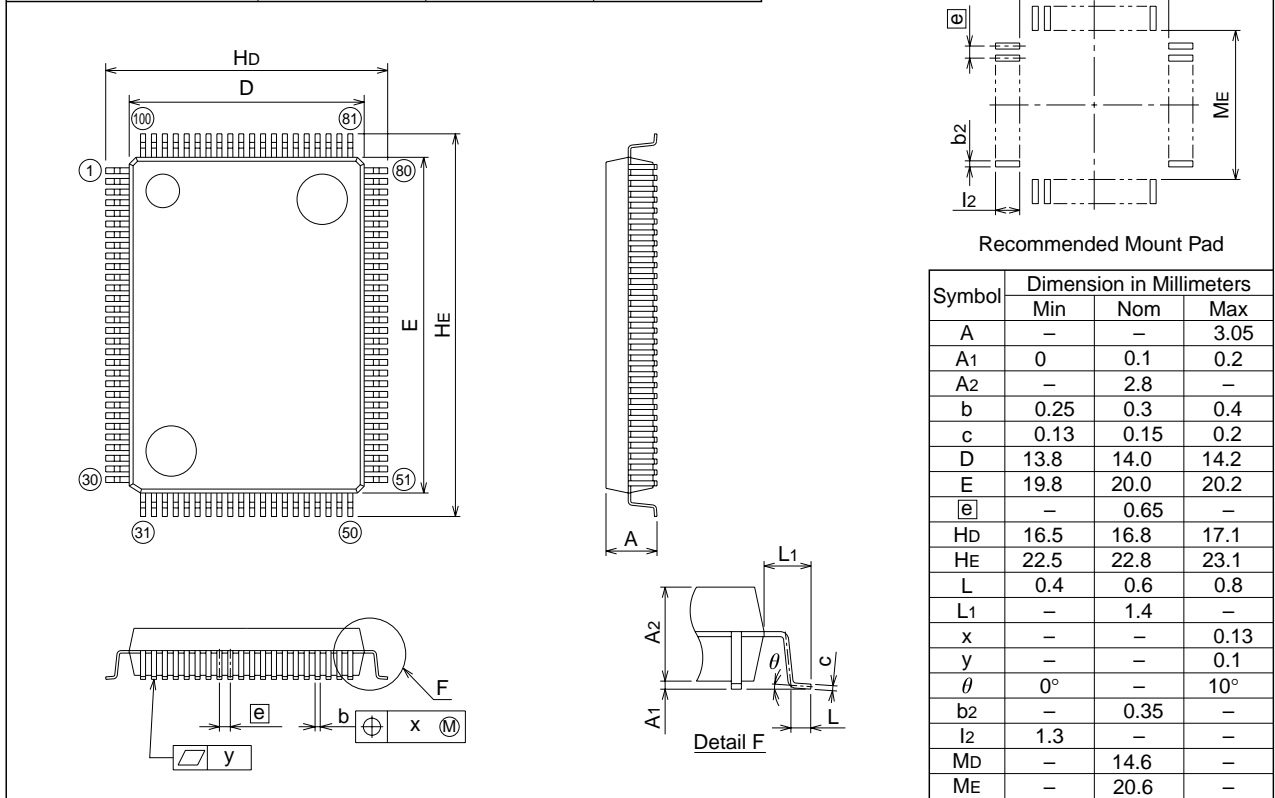
Figure 5.10.4 Timing diagram in memory expansion mode and microprocessor mode (3)

## 6. 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





## 7. Flash Memory

### 7.1 Description

The M306V7 (flash memory version) contains the DINOR (Divided bit line NOR) type of flash memory that can be rewritten with a single voltage of 3.3 V. For this flash memory, three flash memory modes are available in which to read, program, and erase: parallel I/O and standard serial I/O modes in which the flash memory can be manipulated using a programmer and a CPU rewrite mode in which the flash memory can be manipulated by the Central Processing Unit (CPU). Each mode is detailed in the pages to follow.

The flash memory is divided into several blocks as shown in Figure 7.1.1, so that memory can be erased one block at a time.

In addition to the ordinary user ROM area to store a microcomputer operation control program, the flash memory has a boot ROM area that is used to store a program to control rewriting in CPU rewrite and standard serial I/O modes. This boot ROM area has had a standard serial I/O mode control program stored in it when shipped from the factory. However, the user can write a rewrite control program in this area that suits the user's application system. This boot ROM area can be rewritten in only parallel I/O mode.

Table 7.1.1 is a performance outline.

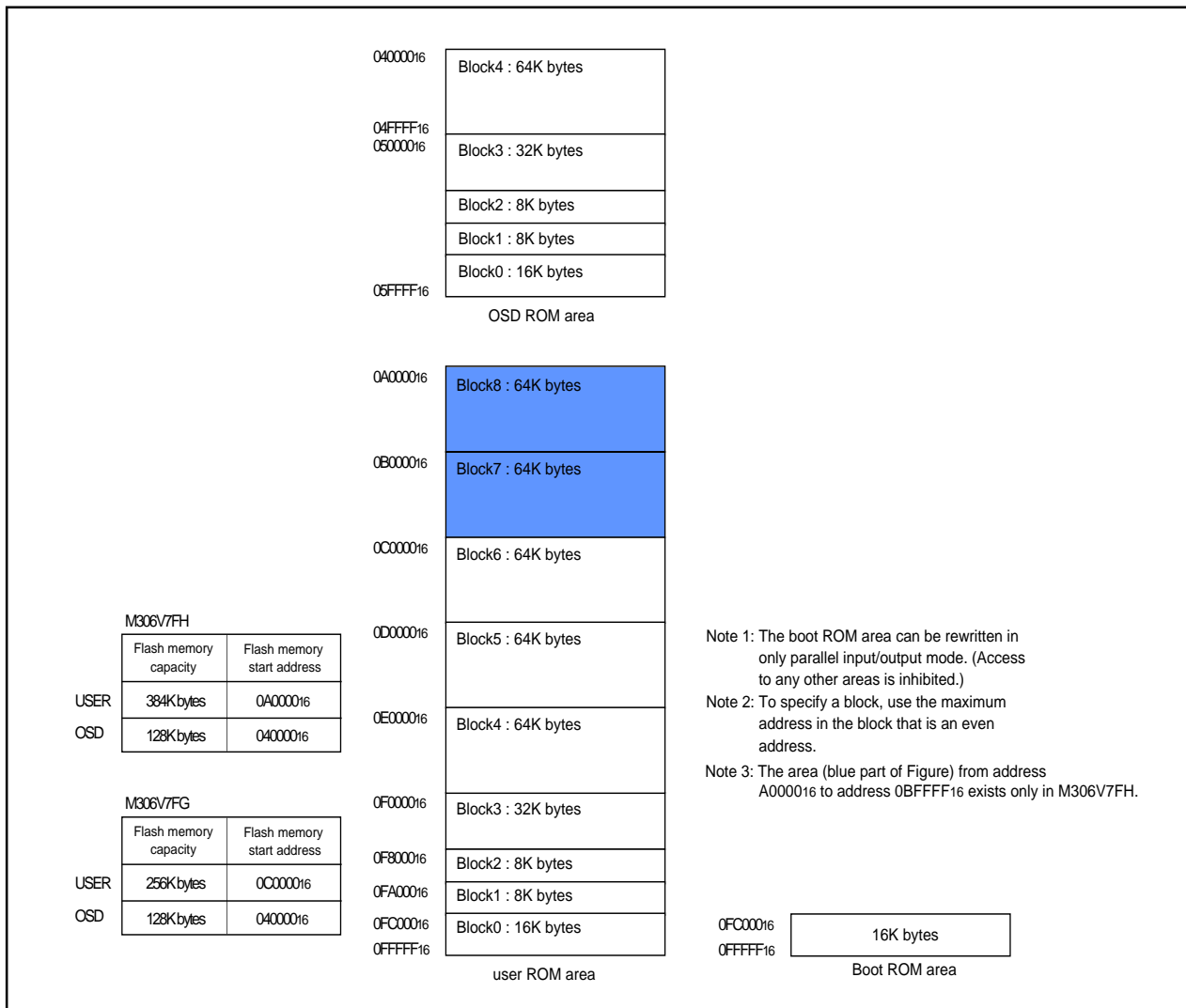


Figure 7.1.1. Block diagram of flash memory version

**Table 7.1.1. Outline Performance of the M306V7 (flash memory version)**

Item		Performance
Power supply voltage		f(XIN)=16MHz, without wait, 3.15V to 3.45V
Program/erase voltage		3.15V to 3.45V : f(BCLK)=6.25MHz, without wait
Flash memory operation mode		Three modes (parallel I/O, standard serial I/O, CPU rewrite)
Erase block division	User ROM area	See Figure 7.1.1
	OSD ROM area	See Figure 7.1.1
	Boot ROM area	One division (16 Kbytes) <sup>(Note 1)</sup>
Program method		In units of pages (in units of 256 bytes)
Erase method		Collective erase/block erase
Program/erase control method		Program / erase control by software command
Number of commands		6 commands
Program/erase count		100 times
ROM code protect		Standard serial mode

Note: The boot ROM area contains a standard serial I/O mode control program which is stored in it when shipped from the factory. This area can be erased and programmed in only parallel I/O mode.

## 7.2 CPU Rewrite Mode

In CPU rewrite mode, the on-chip flash memory can be operated on (read, program, or erase) under control of the Central Processing Unit (CPU).

In CPU rewrite mode, only the user ROM area shown in Figure 1.28.1 can be rewritten; the boot ROM area cannot be rewritten. Make sure the program and block erase commands are issued for only the user ROM area and each block area.

The control program for CPU rewrite mode can be stored in either user ROM or boot ROM area. In the CPU rewrite mode, because the flash memory cannot be read from the CPU, the rewrite control program must be transferred to any area other than the internal flash memory before it can be executed.

### 7.2.1 Microcomputer Mode and Boot Mode

The control program for CPU rewrite mode must be written into the user ROM or boot ROM area in parallel I/O mode beforehand. (If the control program is written into the boot ROM area, the standard serial I/O mode becomes unusable.)

See Figure 1.28.1 for details about the boot ROM area.

Normal microcomputer mode is entered when the microcomputer is reset with pulling CNVss pin low. In this case, the CPU starts operating using the control program in the user ROM area.

When the microcomputer is reset by pulling the P55 pin low, the CNVss pin high, and the P50 pin high, the CPU starts operating using the control program in the boot ROM area. This mode is called the "boot" mode. The control program in the boot ROM area can also be used to rewrite the user ROM area.

### 7.2.2 Block Address

Block addresses refer to the maximum even address of each block. These addresses are used in the block erase command.

### 7.2.3 Outline Performance (CPU Rewrite Mode)

In the CPU rewrite mode, the CPU erases, programs and reads the internal flash memory as instructed by software commands. Operations must be executed from a memory other than the internal flash memory, such as the internal RAM.

When the CPU rewrite mode select bit (bit 1 at address 0313<sub>16</sub> for USER area/0317<sub>16</sub> for OSD area) is set to "1", transition to CPU rewrite mode occurs and software commands can be accepted.

In the CPU rewrite mode, write to and read from software commands and data into even-numbered address ("0" for byte address A0) in 16-bit units. Always write 8-bit software commands into even-numbered address. Commands are ignored with odd-numbered addresses.

Use software commands to control program and erase operations. Whether a program or erase operation has terminated normally or in error can be verified by reading the status register.

Figure 7.2.1 shows the flash memory control register and the flash memory switch register.

Bit 0 of the flash memory control register is the RY/B $\bar{Y}$  status flag used exclusively to read the operating status of the flash memory. During programming and erase operations, it is "0". Otherwise, it is "1".

Bit 1 of the flash memory control register is the CPU rewrite mode select bit. The CPU rewrite mode is entered by setting this bit to "1", so that software commands become acceptable. In CPU rewrite mode, the CPU becomes unable to access the internal flash memory directly. Therefore, write bit 1 in an area other than the internal flash memory. To set this bit to "1", it is necessary to write "0" and then write "1" in succession. The bit can be set to "0" by only writing a "0".

Bit 3 of the flash memory control register is the flash memory reset bit used to reset the control circuit of the internal flash memory. This bit is used when exiting CPU rewrite mode and when flash memory access has failed. When the CPU rewrite mode select bit is "1", writing "1" for this bit resets the control circuit. To release the reset, it is necessary to set this bit to "0".

Bit 5 of the flash memory control register is a user ROM area select bit which is effective in only boot mode. If this bit is set to "1" in boot mode, the area to be accessed is switched from the boot ROM area to the user ROM area. When the CPU rewrite mode needs to be used in boot mode, set this bit to "1". Note that if the microcomputer is booted from the user ROM area, it is always the user ROM area that can be accessed and this bit has no effect. When in boot mode, the function of this bit is effective regardless of whether the CPU rewrite mode is on or off. Use the control program except in the internal flash memory to rewrite this bit.

The bit 1 of flash memory change register is a change bit of USER domain and OSD domain. An access domain changes according to the contents of a setting of this bit. Access to the domain which is not chosen cannot be performed including a memory lead. Moreover, after changing this bit before access of an object domain is attained, the waiting time of 50 clock cycle is required.

Figure 7.2.2 shows a flowchart for setting/releasing the CPU rewrite mode. Always perform operation as indicated in this flowchart.

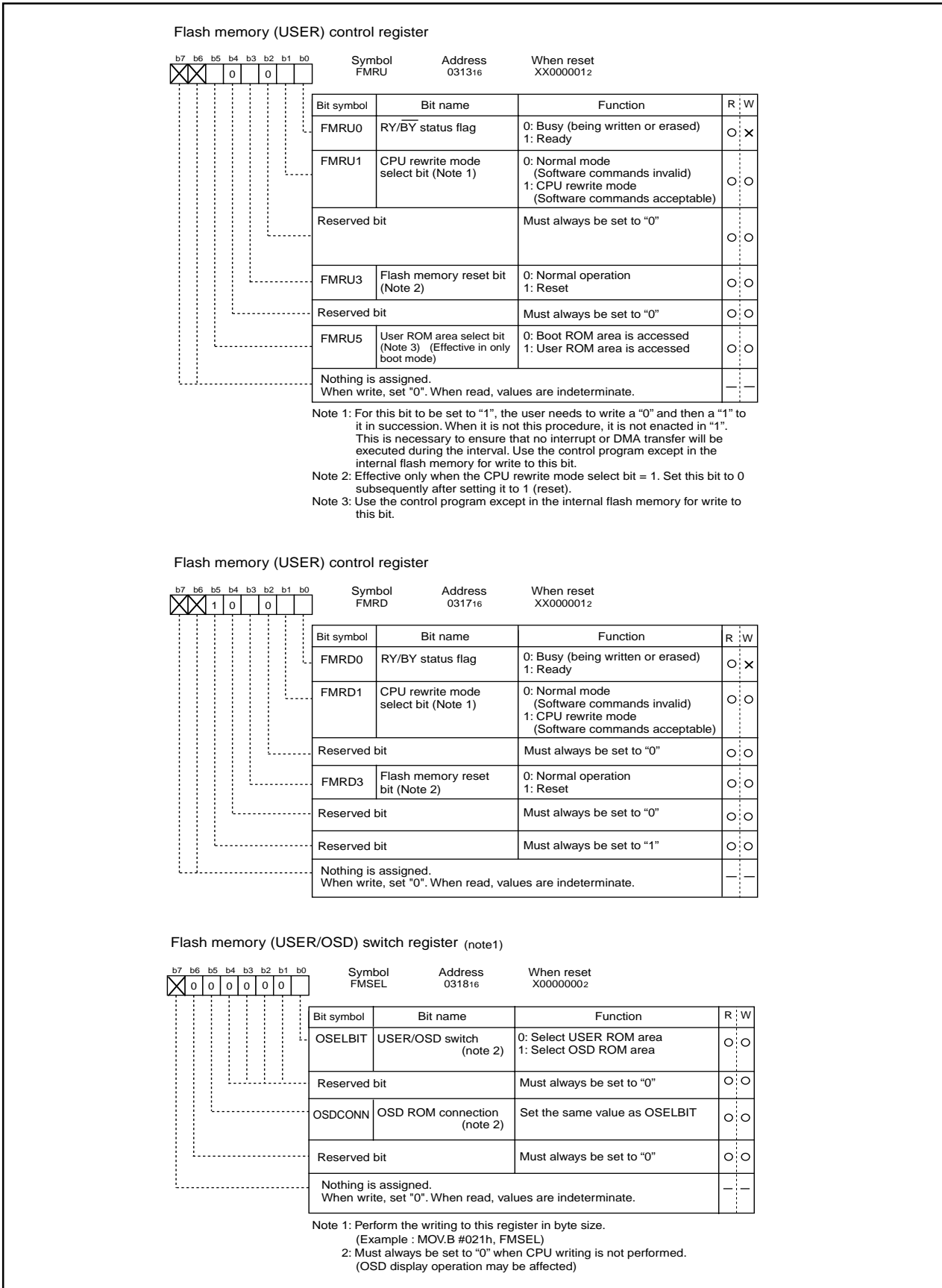


Figure 7.2.1. Flash memory control register, Flash memory switch register

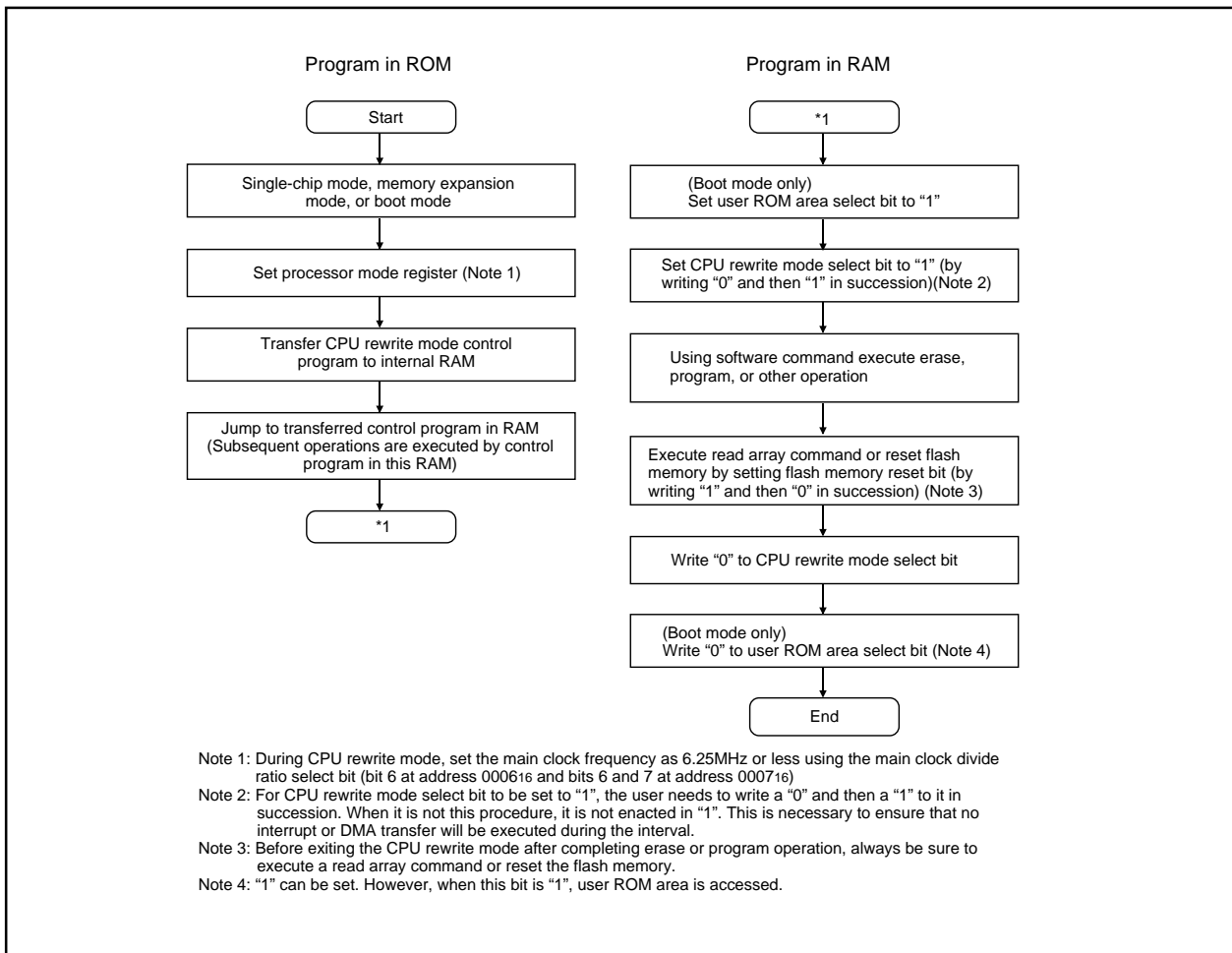


Figure 7.2.2. CPU Rewrite Mode Set/Reset Flowchart

## 7.2.4 Precautions on CPU Rewrite Mode

Described below are the precautions to be observed when rewriting the flash memory in CPU rewrite mode.

### (1) Operation speed

During CPU rewrite mode, set the main clock frequency 6.25 MHz using the main clock divide ratio select bit (bit 6 at address 0006<sub>16</sub> and bits 6 and 7 at address 0007<sub>16</sub>.)

### (2) Instructions inhibited against use

The instructions listed below cannot be used during CPU rewrite mode because they refer to the internal data of the flash memory:

UND instruction, INTO instruction, JMPS instruction, JSRS instruction, and BRK instruction

### (3) Interrupts inhibited against use

The address match interrupt cannot be used during CPU rewrite mode because they refer to the internal data of the flash memory. If interrupts have their vector in the variable vector table, they can be used by transferring the vector into the RAM area. The watchdog timer interrupts each can be used to change the flash memory's operation mode forcibly to read array mode upon occurrence of the interrupt. Since the rewrite operation is halted when the watchdog timer interrupts occur, the erase/program operation needs to be performed over again.

Disabling erase or rewrite operations for address FC000<sub>16</sub> to address FFFFF<sub>16</sub> in the user ROM block disables these operations for all subsequent blocks as well. Therefore, it is recommended to rewrite this block in the standard serial I/O mode.

### (4) Reset

Reset input is always accepted. After a reset, the addresses 060000<sub>16</sub> through (flash memory start address-1) are made a reserved area and cannot be accessed. Therefore, if your product has this area in the user ROM area, do not write any address of this area to the reset vector.

### (5) Access disable

Write CPU rewrite mode select bit, user ROM area select bit and USER/OSD change bit in an area other than the internal flash memory.

### (6) How to access

For CPU rewrite mode select bit to be set to "1", the user needs to write a "0" and then a "1" to it in succession. When it is not this procedure, it is not enacted in "1". This is necessary to ensure that no interrupt or DMA transfer will be executed during the interval.

### (7) Change time

When change an access area from USER/OSD change bit's, insert the waiting time about 50 clock cycle until access of an object area is attained.

## 7.2.5 Software Commands

Table 7.2.1 lists the software commands available with the M16C/62 (flash memory version). After setting the CPU rewrite mode select bit to 1, write a software command to specify an erase or program operation. Note that when entering a software command, the upper byte (D8 to D15) is ignored. The content of each software command is explained below.

**Table 7.2.1. List of Software Commands (CPU Rewrite Mode)**

Command	First bus cycle			Second bus cycle			Third bus cycle		
	Mode	Address	Data (D <sub>0</sub> to D <sub>7</sub> )	Mode	Address	Data (D <sub>0</sub> to D <sub>7</sub> )	Mode	Address	Data (D <sub>0</sub> to D <sub>7</sub> )
Read array	Write	X (Note 5)	FF <sub>16</sub>						
Read status register	Write	X	70 <sub>16</sub>	Read	X	SRD (Note 2)			
Clear status register	Write	X	50 <sub>16</sub>						
Page program (Note 3)	Write	X	41 <sub>16</sub>	Write	WA0 (Note 3)	WD0 (Note 3)	Write	WA1	WD1
Block erase	Write	X	20 <sub>16</sub>	Write	BA (Note 4)	D0 <sub>16</sub>			
Erase all unlock block	Write	X	A7 <sub>16</sub>	Write	X	D0 <sub>16</sub>			

Note 1: When a software command is input, the high-order byte of data (D<sub>8</sub> to D<sub>15</sub>) is ignored.

Note 2: SRD = Status Register Data

Note 3: WA = Write Address, WD = Write Data

WA and WD must be set sequentially from 00<sub>16</sub> to FE<sub>16</sub> (byte address; however, an even address). The page size is 256 bytes.

Note 4: BA = Block Address (Enter the maximum address of each block that is an even address.)

Note 5: X denotes a given address in the user ROM area (that is an even address).

### Read Array Command (FF<sub>16</sub>)

The read array mode is entered by writing the command code “FF<sub>16</sub>” in the first bus cycle. When an even address to be read is input in one of the bus cycles that follow, the content of the specified address is read out at the data bus (D<sub>0</sub>–D<sub>15</sub>), 16 bits at a time.

The read array mode is retained intact until another command is written.

### Read Status Register Command (70<sub>16</sub>)

When the command code “70<sub>16</sub>” is written in the first bus cycle, the content of the status register is read out at the data bus (D<sub>0</sub>–D<sub>7</sub>) by a read in the second bus cycle.

The status register is explained in the next section.

### Clear Status Register Command (50<sub>16</sub>)

This command is used to clear the bits SR3 to 5 of the status register after they have been set. These bits indicate that operation has ended in an error. To use this command, write the command code “50<sub>16</sub>” in the first bus cycle.



### Page Program Command (41<sub>16</sub>)

Page program allows for high-speed programming in units of 256 bytes. Page program operation starts when the command code "41<sub>16</sub>" is written in the first bus cycle. In the second bus cycle through the 129th bus cycle, the write data is sequentially written 16 bits at a time. At this time, the addresses A<sub>0</sub>-A<sub>7</sub> need to be incremented by 2 from "00<sub>16</sub>" to "FE<sub>16</sub>." When the system finishes loading the data, it starts an auto write operation (data program and verify operation).

Whether the auto write operation is completed can be confirmed by reading the status register or the flash memory control register. At the same time the auto write operation starts, the read status register mode is automatically entered, so the content of the status register can be read out. The status register bit 7 (SR7) is set to 0 at the same time the auto write operation starts and is returned to 1 upon completion of the auto write operation. In this case, the read status register mode remains active until the Read Array command (FF<sub>16</sub>) is written or the flash memory is reset using its reset bit.

The RY/ $\overline{\text{BY}}$  status flag of the flash memory control register 0 is 0 during auto write operation and 1 when the auto write operation is completed as is the status register bit 7.

After the auto write operation is completed, the status register can be read out to know the result of the auto write operation. For details, refer to the section where the status register is detailed.

Figure 7.2.3 shows an example of a page program flowchart.

And, Additional writes to the already programmed pages are prohibited.

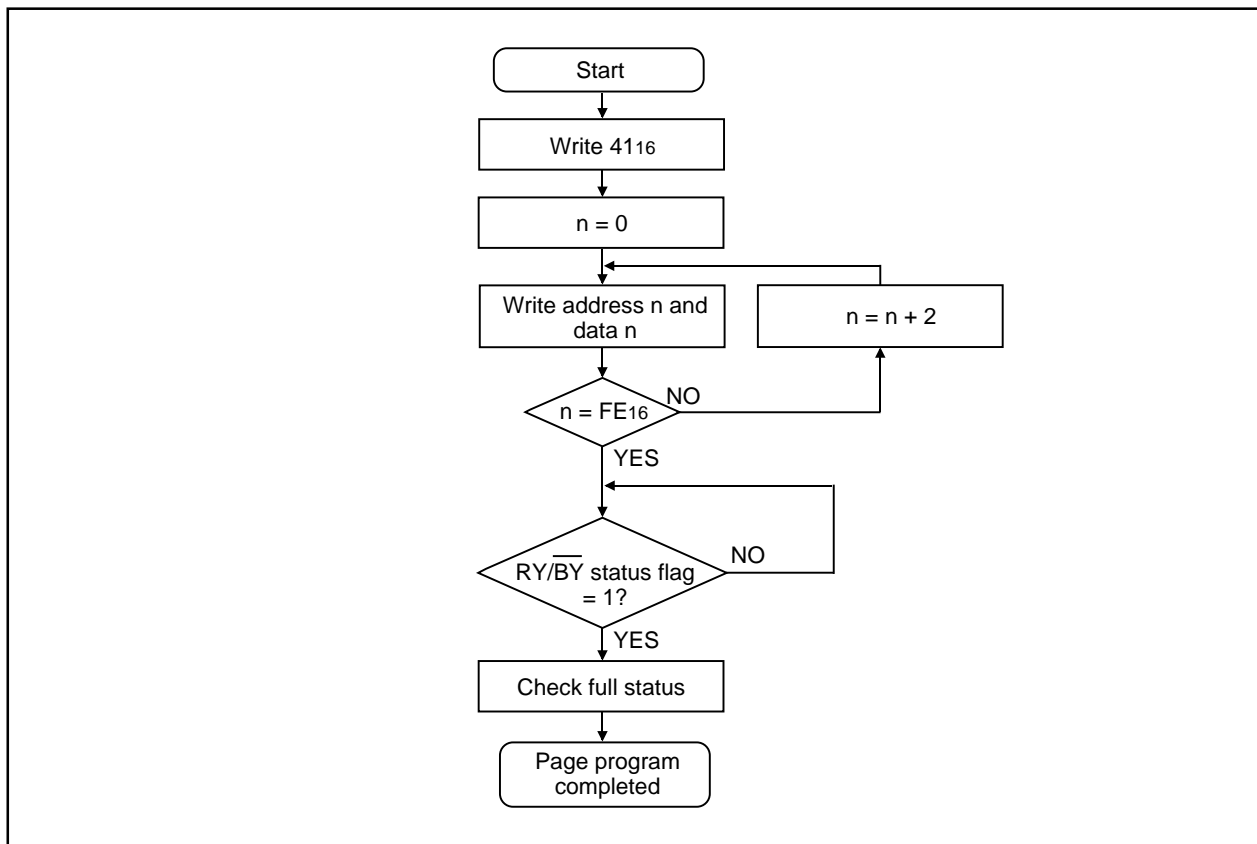


Figure 7.2.3. Page program flowchart

**Block Erase Command (2016/D016)**

By writing the command code "2016" in the first bus cycle and the confirmation command code "D016" in the second bus cycle that follows to the block address of a flash memory block, the system initiates an auto erase (erase and erase verify) operation.

Whether the auto erase operation is completed can be confirmed by reading the status register or the flash memory control register. At the same time the auto erase operation starts, the read status register mode is automatically entered, so the content of the status register can be read out. The status register bit 7 (SR7) is set to 0 at the same time the auto erase operation starts and is returned to 1 upon completion of the auto erase operation. In this case, the read status register mode remains active until the Read Array command (FF16) is written or the flash memory is reset using its reset bit. The RY/ $\overline{\text{BY}}$  status flag of the flash memory control register is 0 during auto erase operation and 1 when the auto erase operation is completed as is the status register bit 7.

After the auto erase operation is completed, the status register can be read out to know the result of the auto erase operation. For details, refer to the section where the status register is detailed.

Figure 7.2.4 shows an example of a block erase flowchart.

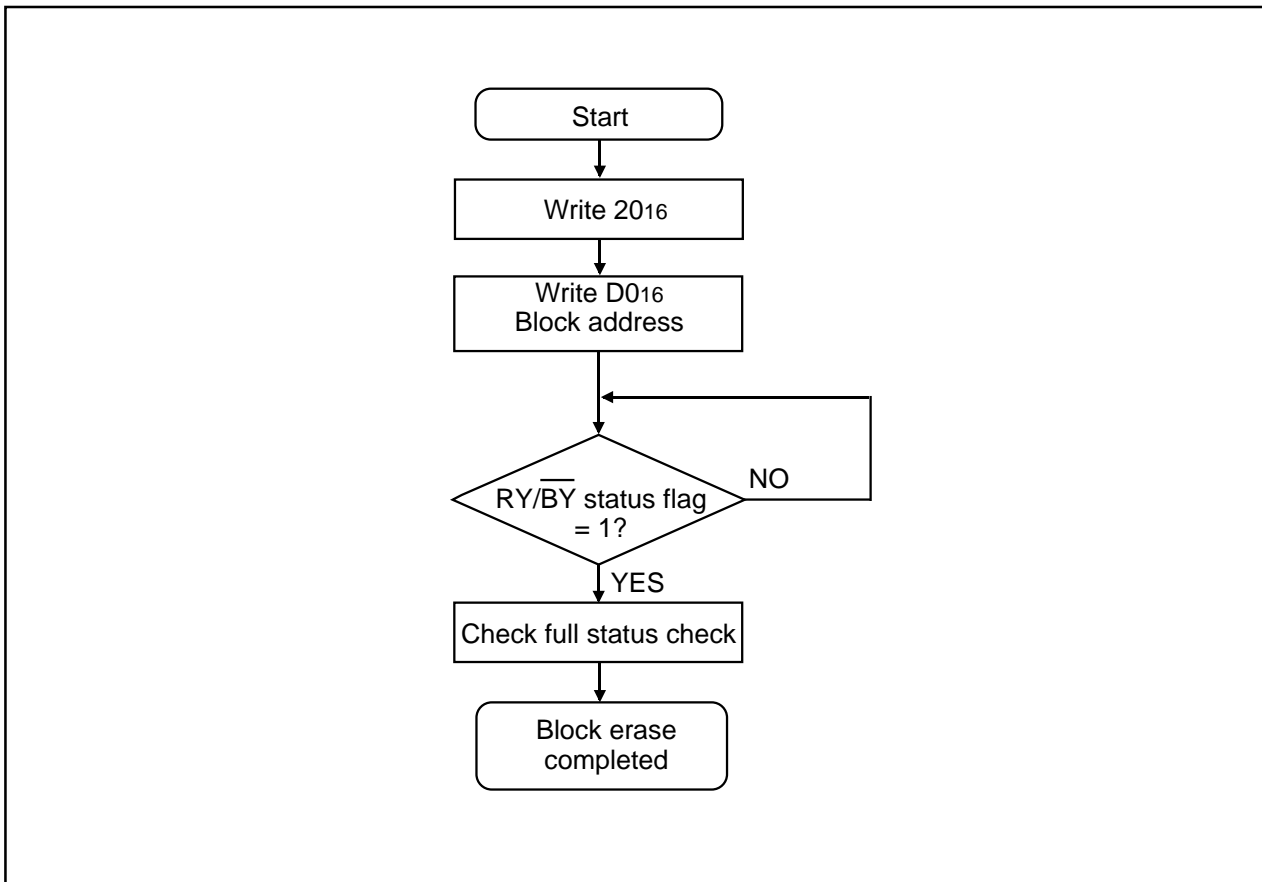


Figure 7.2.4. Block erase flowchart

**Erase All Unlock Blocks Command (A7<sub>16</sub>/D0<sub>16</sub>)**

By writing the command code "A7<sub>16</sub>" in the first bus cycle and the confirmation command code "D0<sub>16</sub>" in the second bus cycle that follows, the system starts erasing blocks successively.

Whether the erase all unlock blocks command is terminated can be confirmed by reading the status register or the flash memory control register 0, in the same way as for block erase. Also, the status register can be read out to know the result of the auto erase operation.

## 7.2.6 Status Register

The status register indicates the operating status of the flash memory and whether an erase or program operation has terminated normally or in an error. The content of this register can be read out by only writing the read status register command (70<sub>16</sub>). Table 7.2.2 details the status register.

The status register is cleared by writing the Clear Status Register command (50<sub>16</sub>).

After a reset, the status register is set to "80<sub>16</sub>."

Each bit in this register is explained below.

### **Write state machine (WSM) status (SR7)**

After power-on, the write state machine (WSM) status is set to 1.

The write state machine (WSM) status indicates the operating status of the device, as for output on the RY/ $\overline{\text{BY}}$  pin. This status bit is set to 0 during auto write or auto erase operation and is set to 1 upon completion of these operations.

### **Erase status (SR5)**

The erase status informs the operating status of auto erase operation to the CPU. When an erase error occurs, it is set to 1.

The erase status is reset to 0 when cleared.

### **Program status (SR4)**

The program status informs the operating status of auto write operation to the CPU. When a write error occurs, it is set to 1.

The program status is reset to 0 when cleared.

When an erase command is in error (which occurs if the command entered after the block erase command (20<sub>16</sub>) is not the confirmation command (D0<sub>16</sub>), both the program status and erase status (SR5) are set to 1.

When the program status or erase status = 1, the following commands entered by command write are not accepted.

Also, when the valid command is not entered correctly, both SR4 and SR5 are set to 1 (command sequence error).

**Block status after program (SR3)**

If excessive data is written (phenomenon whereby the memory cell becomes depressed which results in data not being read correctly), "1" is set for the program status after-program at the end of the page write operation. In other words, when writing ends successfully, "8016" is output; when writing fails, "9016" is output; and when excessive data is written, "8816" is output.

**Table 7.2.2. Definition of each bit in status register**

Each bit of SRD	Status name	Definition	
		"1"	"0"
SR7 (bit7)	Write state machine (WSM) status	Ready	Busy
SR6 (bit6)	Reserved	-	-
SR5 (bit5)	Erase status	Terminated in error	Terminated normally
SR4 (bit4)	Program status	Terminated in error	Terminated normally
SR3 (bit3)	Block status after program	Terminated in error	Terminated normally
SR2 (bit2)	Reserved	-	-
SR1 (bit1)	Reserved	-	-
SR0 (bit0)	Reserved	-	-

### 7.2.7 Full Status Check

By performing full status check, it is possible to know the execution results of erase and program operations. Figure 7.2.5 shows a full status check flowchart and the action to be taken when each error occurs.

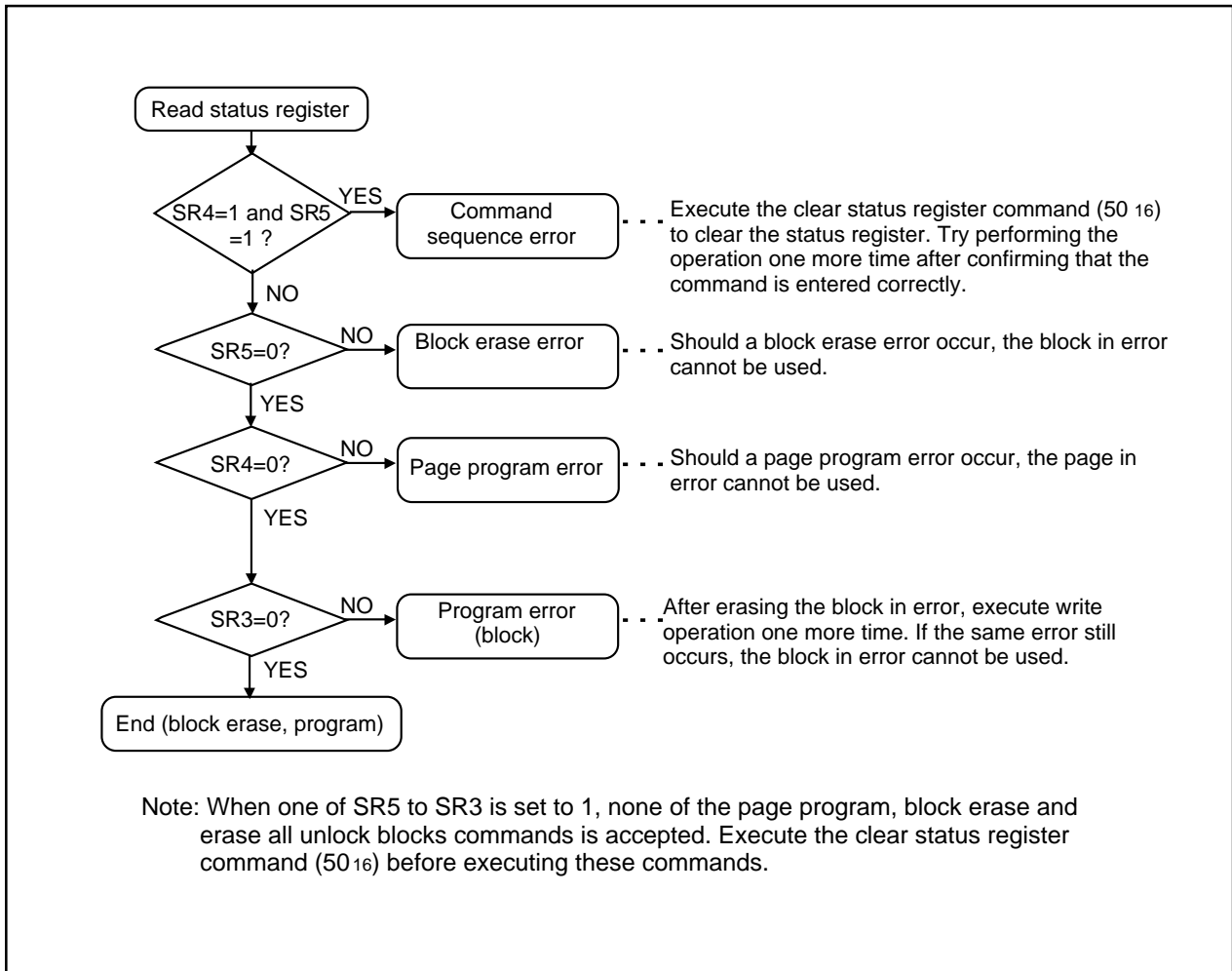


Figure 7.2.5. Full status check flow chart and the solution at the time of each error generating

### 7.2.8 Built-in flash memory rewriting prohibition function

For standard serial I/O mode, the ID code check function is built-in so that the contents of a built-in flash memory cannot be rewritten easily.

## 7.3 Parallel I/O Mode

In this mode, the M306V7 (flash memory version) operates in a manner similar to the flash memory M5M29FB/T800 from Mitsubishi. Since there are some differences with regard to the functions not available with the microcomputer and matters related to memory capacity, the M16C/62 cannot be programmed by a programmer for the flash memory.

Use an exclusive programmer supporting M306V7 (flash memory version).

Refer to the instruction manual of each programmer maker for the details of use.

### 7.3.1 User ROM and Boot ROM Areas

In parallel I/O mode, the user ROM and boot ROM areas shown in Figure 7.1.1 can be rewritten. Both areas of flash memory can be operated on in the same way.

Program and block erase operations can be performed in the user ROM area. The user ROM area and its blocks are shown in Figure 7.1.1.

The boot ROM area is 16 Kbytes in size. In parallel I/O mode, it is located at addresses 0FC000<sub>16</sub> through 0FFFFFF<sub>16</sub>. Make sure program and block erase operations are always performed within this address range. (Access to any location outside this address range is prohibited.)

In the boot ROM area, an erase block operation is applied to only one 16 Kbyte block. The boot ROM area has had a standard serial I/O mode control program stored in it when shipped from the Mitsubishi factory. Therefore, using the device in standard serial input/output mode, you do not need to write to the boot ROM area.

## 7.4 Standard Serial I/O Mode

### Pin functions (Flash memory standard serial I/O mode)

Pin	Name	I/O	Description
VCCI, VSS	Power input		Apply program/erase protection voltage to VCCI pin and 0 V to Vss pin.
CNVss	CNVss	I	Connect to VCCI pin.
$\overline{\text{RESET}}$	Reset input	I	Reset input pin. While reset is "L" level, a 20 cycle or longer clock must be input to XIN pin.
XIN	Clock input	I	Connect a ceramic resonator or crystal oscillator between XIN and XOUT pins. To input an externally generated clock, input it to XIN pin and open XOUT pin.
XOUT	Clock output	O	
BYTE	BYTE	I	Connect this pin to VCCI or Vss.
VcCE	Analog power supply input		Apply protection voltage to VcCE.
TVSETB	Set input	I	Connect a VSS.
P00 to P07	Input port P0	I	Input "H" or "L" level signal or open.
P10 to P17	Input port P1	I	Input "H" or "L" level signal or open.
P20 to P27	Input port P2	I	Input "H" or "L" level signal or open.
P30 to P37	Input port P3	I	Input "H" or "L" level signal or open.
P40 to P47	Input port P4	I	Input "H" or "L" level signal or open.
P51 to P54, P56, P57	Input port P5	I	Input "H" or "L" level signal or open.
P50	$\overline{\text{CE}}$ input	I	Input "H" level signal.
P55	$\overline{\text{EPM}}$ input	I	Input "L" level signal.
P64 to P67	Input port P6	I	Input "H" or "L" level signal or open.
P60	BUSY output	O	BUSY signal output pin
P61	SCLK input	I	Serial clock input pin
P62	RxD input	I	Serial data input pin
P63	TxD output	O	Serial data output pin
P70 to P77	Input port P7	I	Input "H" or "L" level signal or open.
P82, P83, P86, P87	Input port P8	I	Input "H" or "L" level signal or open.
P90 to P94	Input port P9	I	Input "H" or "L" level signal or open.
P102–P107	Input port P10	I	Input "H" or "L" level signal or open.
HLF, VHOLD, CVIN		I	Input "H" or "L" level signal or open.
HLF2, VHOLD2, CVIN2		I	Input "H" or "L" level signal or open.
VSUNC, HSUNC		I	Input "H" or "L" level signal or open.



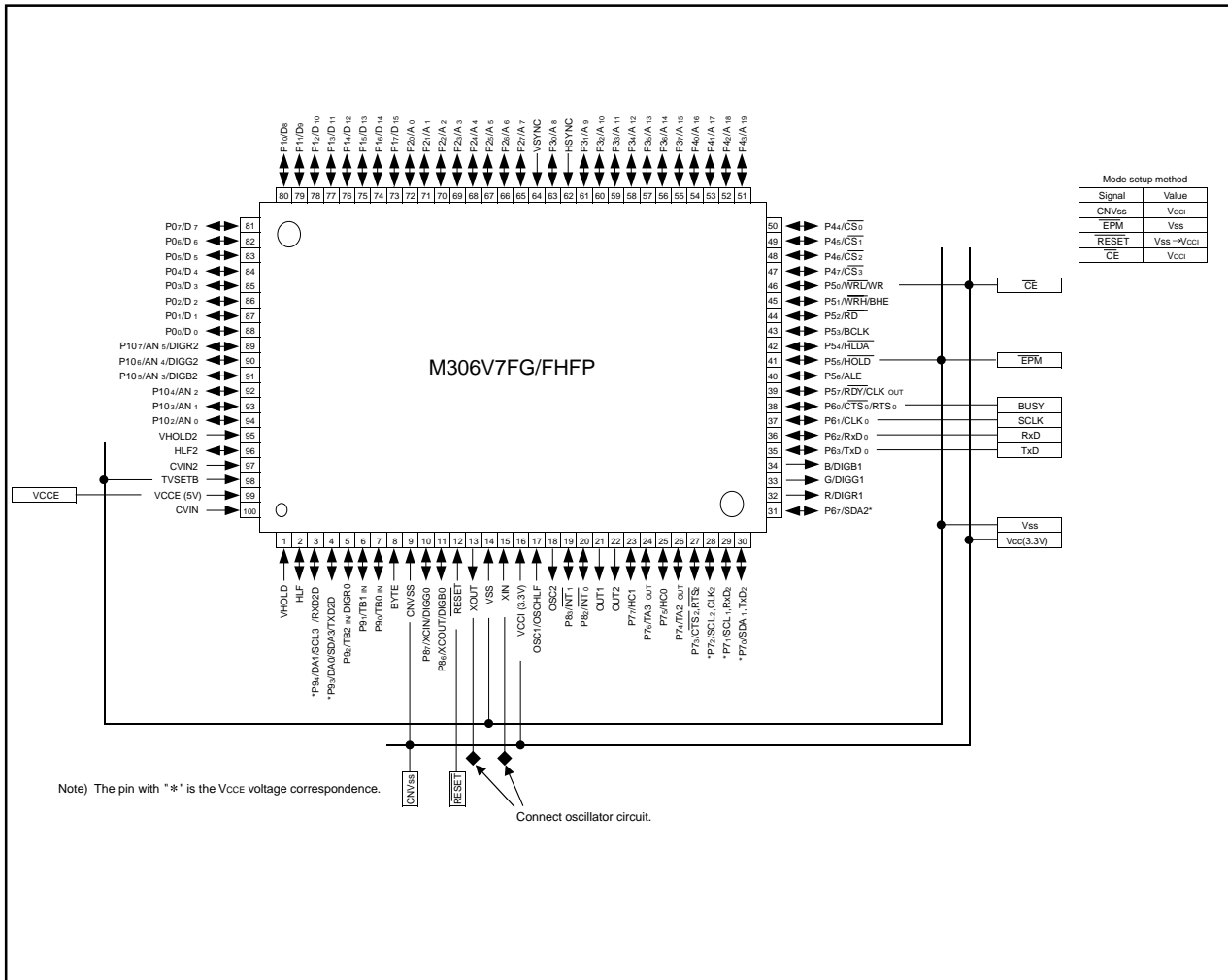


Figure 7.4.1. Pin connections for serial I/O mode (1)

### 7.4.1 Standard Serial I/O Mode

The standard serial I/O mode serially inputs and outputs the software commands, addresses and data necessary for operating (read, program, erase, etc.) the internal flash memory. It uses a purpose-specific peripheral unit.

The standard serial I/O mode differs from the parallel I/O mode in that the CPU controls operations like rewriting (uses the CPU rewrite mode) in the flash memory or serial input for rewriting data. The standard serial I/O mode is started by clearing the reset with an “H” level signal at the P50 ( $\overline{CE}$ ) pin, an “L” signal at the P55 ( $\overline{EPM}$ ) pin and an “H” level at the CNVss pin. (For the normal microprocessor mode, set CNVss to “L”.)

This control program is written in the boot ROM area when shipped from Mitsubishi Electric. Therefore, if the boot ROM area is rewritten in the parallel I/O mode, the standard serial I/O mode cannot be used. Figure 7.4.1 shows the pin connections for the standard serial I/O mode. Serial data I/O uses four UART1 pins: CLK0, RxD0, TxD0 and RTS0 (BUSY).

The CLK0 pin is the transfer clock input pin and it inputs the external transfer clock. The TxD0 pin outputs the CMOS signal. The RTS0 (BUSY) pin outputs an “L” level when reception setup ends and an “H” level when the reception operation starts. Transmission and reception data is transferred serially in 8-byte blocks.

In the standard serial I/O mode, only the user ROM area shown in Figure 7.1.1 can be rewritten, the boot ROM area cannot.

### Function Overview (Standard Serial I/O Mode)

In the standard serial I/O mode, software commands, addresses and data are input and output between the flash memory and an external device (peripheral unit, etc.) using a 4-wire clock synchronized serial I/O (UART0). In reception, the software commands, addresses and program data are synchronized with the rise of the transfer clock input to the CLK0 pin and input into the flash memory via the RxD0 pin.

In transmission, the read data and status are synchronized with the fall of the transfer clock and output to the outside from the TxD0 pin.

The TxD0 pin is CMOS output. Transmission is in 8-bit blocks and LSB first.

When busy, either during transmission or reception, or while executing an erase operation or program, the RTS0 (BUSY) pin is “H” level. Accordingly, do not start the next transmission until the RTS0 (BUSY) pin is “L” level.

Also, data in memory and the status register can be read after inputting a software command. It is possible to check flash memory operating status or whether a program or erase operation ended successfully or in error by reading the status register.

Software commands and the status register are explained here following.

## Software Commands

Table 7.4.1 lists software commands. In the standard serial I/O mode, erase operations, programs and reading are controlled by transferring software commands via the RxD pin. Software commands are explained here below.

**Table 7.4.1. Software commands (Standard serial I/O mode)**

	Control command	Transmission of the 1st byte	2rd byte	3rd byte	4rd byte	5rd byte	6rd byte	.....	When ID is not verificate
1	Page read	FF <sub>16</sub>	Address (middle)	Address (high)					Not acceptable
2	Page program	41 <sub>16</sub>	Address (middle)	Address (high)	Data input	Data input	Data input	Data output to 259th byte	Not acceptable
3	Block erase	20 <sub>16</sub>	Address (middle)	Address (high)	D0 <sub>16</sub>				Not acceptable
4	Erase all unlocked blocks	A7 <sub>16</sub>	D0 <sub>16</sub>						Not acceptable
5	Read status register	70 <sub>16</sub>							
6	Clear status register	50 <sub>16</sub>							Not acceptable
7	ID check function	F5 <sub>16</sub>	Address (high)	Address (middle)	Address (low)	ID size	ID1	to ID7	Acceptable
8	Download function	FA <sub>16</sub>	Size (low)	Size (high)	Check -sum		To required number of times		Not acceptable
9	Version data output function	FB <sub>16</sub>	Version data output	Version data output	Version data output	Version data output	Version data output	Version data output to 9th byte	Acceptable
10	Boot area output function	FC <sub>16</sub>	Address (middle)	Address (high)	Data output	Data output	Data output	Data output to 259th byte	Not acceptable
11	User ROM area selection function	E0 <sub>16</sub>							Acceptable
12	OSD ROM area selection function	E1 <sub>16</sub>							Acceptable

Notes1: Shading indicates transfer from flash memory microcomputer to peripheral unit.

All other data is transferred from the peripheral unit to the flash memory microcomputer.

2: SRD refers to status register data. SRD1 refers to status register 1 data.

3: All commands can be accepted when the flash memory is totally blank.

4: All commands can be accepted when the flash memory is totally blank.

\* In page lead, page program, and block erase, a domain is automatically changed before command execution.

\* With all erase unlocks, only the domain chosen serves as a candidate for execution.

\* After boot ROM domain output execution is ended where USER ROM domain is chosen.

### Page Read Command

This command reads the specified page (256 bytes) in the flash memory sequentially one byte at a time. Execute the page read command as explained here following.

- (1) Send the “FF<sub>16</sub>” command code in the 1st byte of the transmission.
- (2) Send addresses A<sub>8</sub> to A<sub>15</sub> and A<sub>16</sub> to A<sub>23</sub> in the 2nd and 3rd bytes of the transmission respectively.
- (3) From the 4th byte onward, data (D<sub>0</sub>–D<sub>7</sub>) for the page (256 bytes) specified with addresses A<sub>8</sub> to A<sub>23</sub> will be output sequentially from the smallest address first in sync with the rise of the clock.

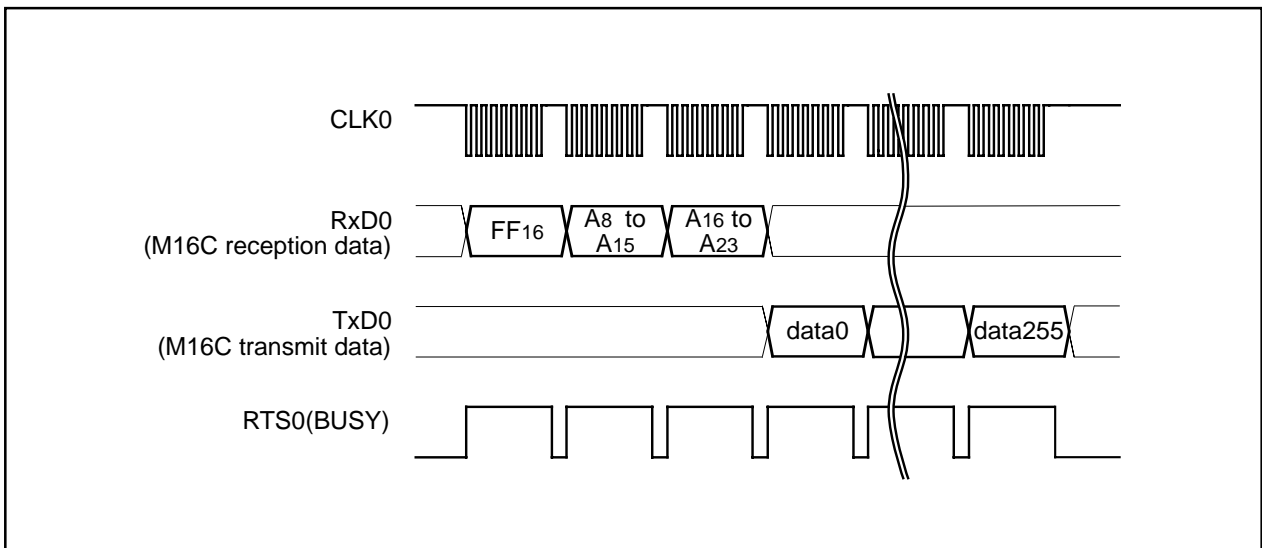


Figure 7.4.2. Timing for page read

### Read Status Register Command

This command reads status information. When the “70<sub>16</sub>” command code is sent in the 1st byte of the transmission, the contents of the status register (SRD) specified in the 2nd byte of the transmission and the contents of status register 1 (SRD1) specified in the 3rd byte of the transmission are read.

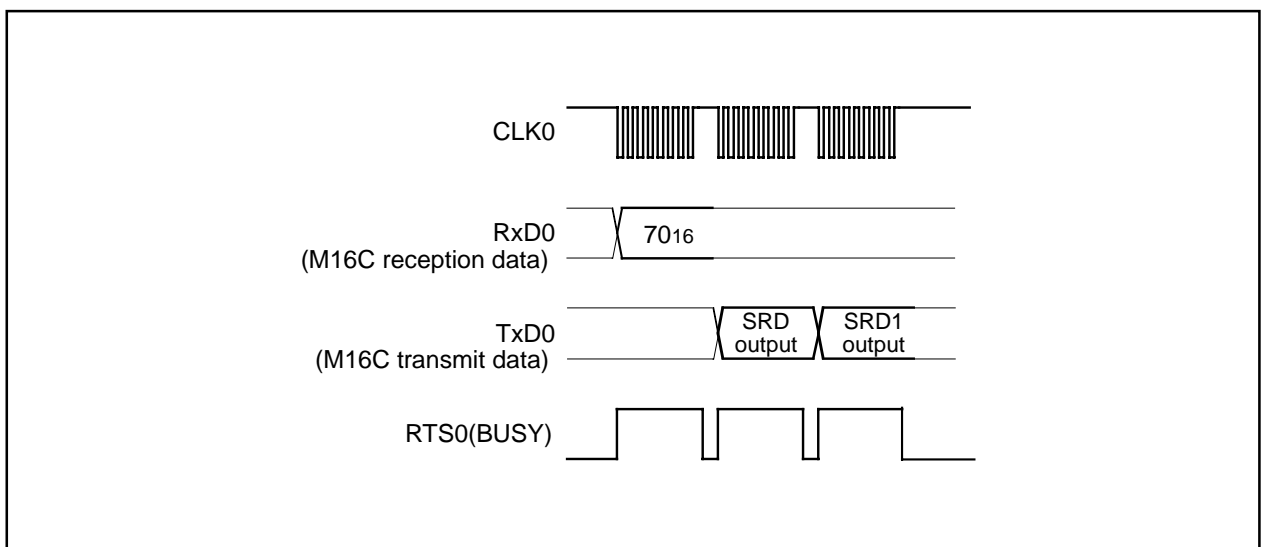


Figure 7.4.3. Timing for reading the status register

### Clear Status Register Command

This command clears the bits (SR3–SR5) which are set when the status register operation ends in error. When the “50<sub>16</sub>” command code is sent in the 1st byte of the transmission, the aforementioned bits are cleared. When the clear status register operation ends, the RTS0 (BUSY) signal changes from the “H” to the “L” level.

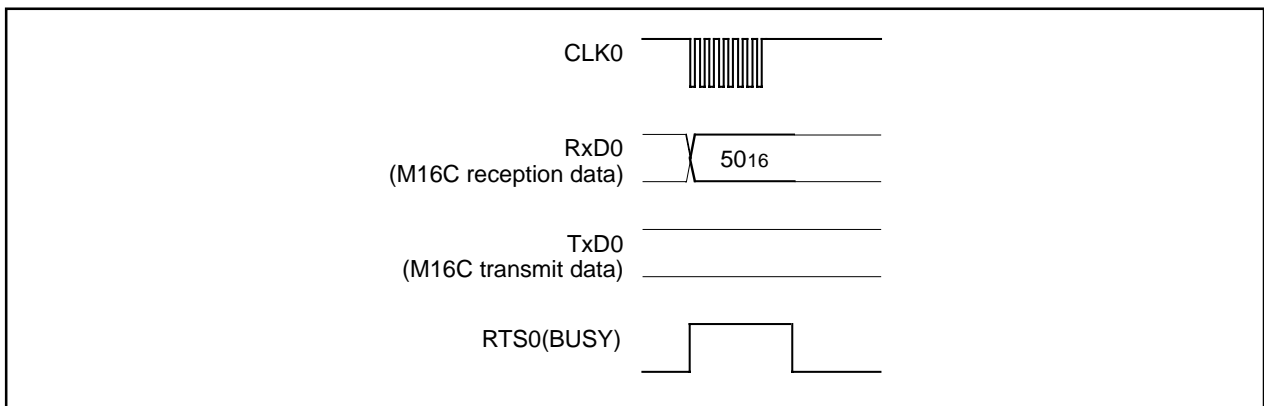


Figure 7.4.4. Timing for clearing the status register

### Page Program Command

This command writes the specified page (256 bytes) in the flash memory sequentially one byte at a time. Execute the page program command as explained here following.

- (1) Send the “41<sub>16</sub>” command code in the 1st byte of the transmission.
- (2) Send addresses A8 to A15 and A16 to A23 in the 2nd and 3rd bytes of the transmission respectively.
- (3) From the 4th byte onward, as write data (D<sub>0</sub>–D<sub>7</sub>) for the page (256 bytes) specified with addresses A8 to A23 is input sequentially from the smallest address first, that page is automatically written.

When reception setup for the next 256 bytes ends, the RTS0 (BUSY) signal changes from the “H” to the “L” level. The result of the page program can be known by reading the status register. For more information, see the section on the status register.

Each block can be write-protected with the lock bit. For more information, see the section on the data protection function. Additional writing is not allowed with already programmed pages.

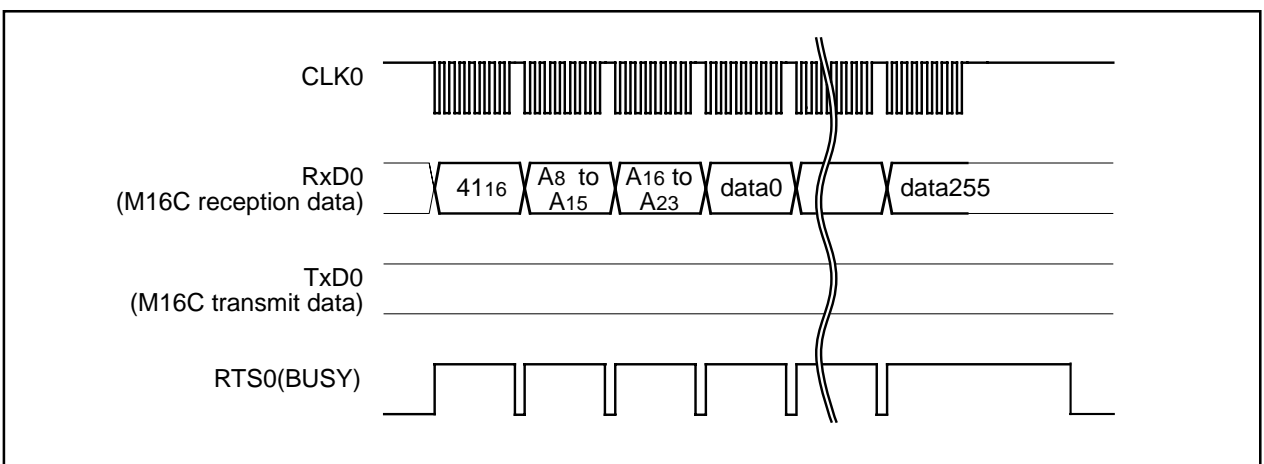


Figure 7.4.5. Timing for the page program

### Block Erase Command

This command erases the data in the specified block. Execute the block erase command as explained here following.

- (1) Send the "2016" command code in the 1st byte of the transmission.
- (2) Send addresses A8 to A15 and A16 to A23 in the 2nd and 3rd bytes of the transmission respectively.
- (3) Send the verify command code "D016" in the 4th byte of the transmission. With the verify command code, the erase operation will start for the specified block in the flash memory. Write the highest address of the specified block for addresses A16 to A23.

When block erasing ends, the RTS0 (BUSY) signal changes from the "H" to the "L" level. After block erase ends, the result of the block erase operation can be known by reading the status register. For more information, see the section on the status register.

Each block can be erase-protected with the lock bit. For more information, see the section on the data protection function.

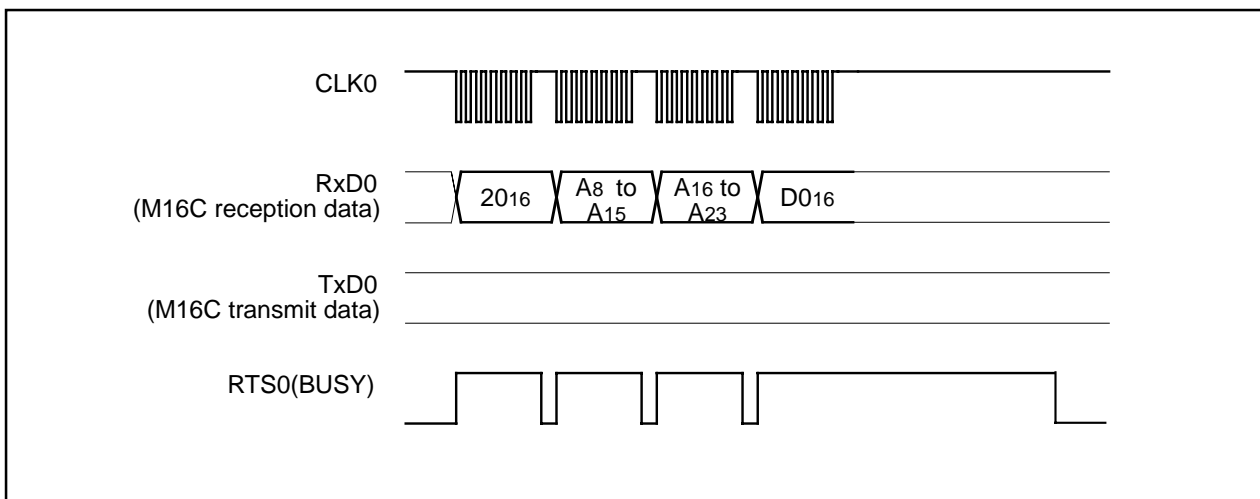


Figure 7.4.6. Timing for block erasing

### Erase All Unlocked Blocks Command

This command erases the content of all blocks. Execute the erase all unlocked blocks command as explained here following.

- (1) Send the "A716" command code in the 1st byte of the transmission.
- (2) Send the verify command code "D016" in the 2nd byte of the transmission. With the verify command code, the erase operation will start and continue for all blocks in the flash memory.

When block erasing ends, the RTS0 (BUSY) signal changes from the "H" to the "L" level. The result of the erase operation can be known by reading the status register. Each block can be erase-protected with the lock bit. For more information, see the section on the data protection function.

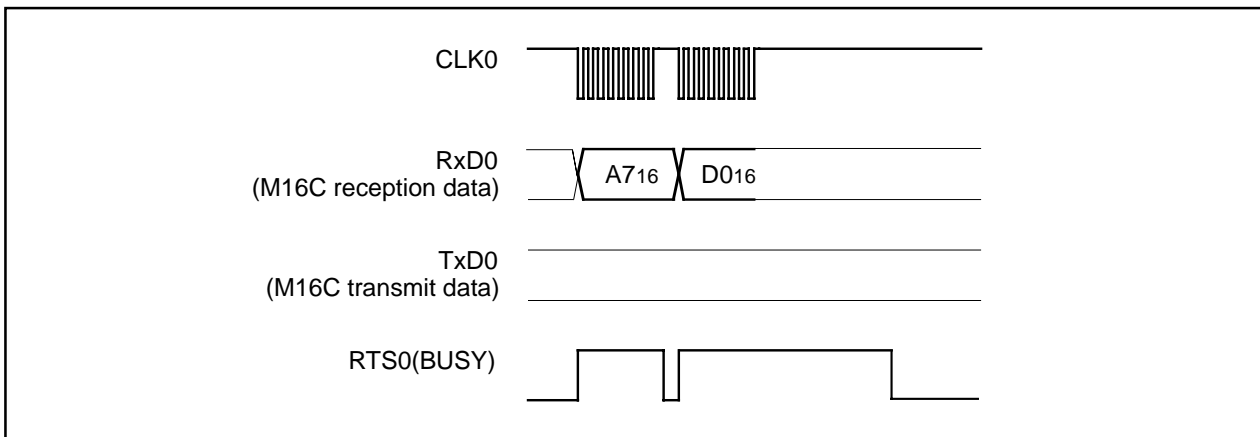


Figure 7.4.7. Timing for erasing all unlocked blocks

### ROM area selection function

This is a command for switching of USER ROM area and OSD ROM area. If command code "E0" or "E1" is transmitted at the 1st byte transmission, USER ROM area or OSD ROM area is chosen.

When end change operation, RTS0 (BUSY) signal changes from "H" to "L."

In addition, the area under selection can be known by reading a status register. Refer to the paragraph of a status register for details.

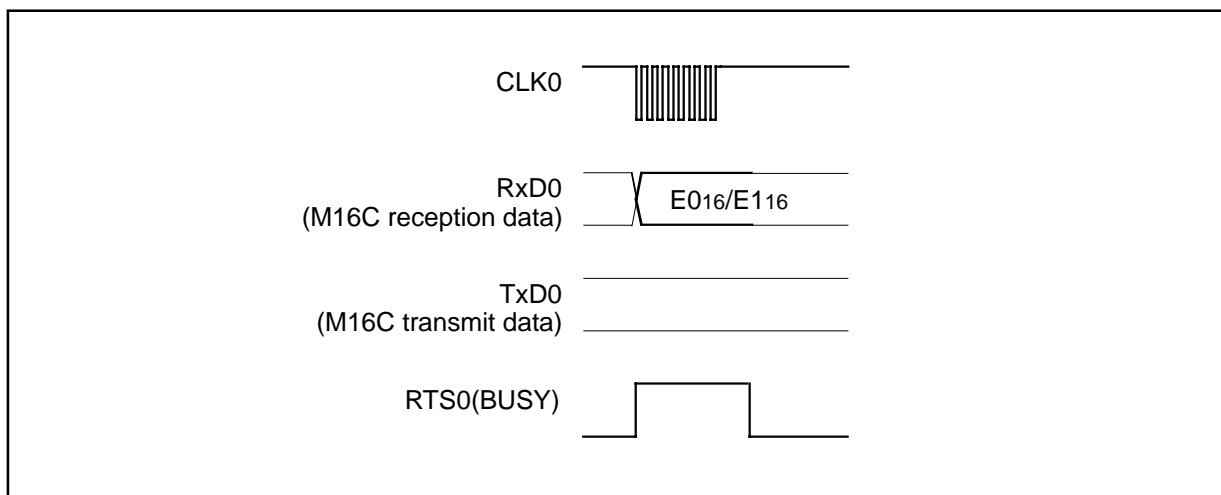


Figure 7.4.8. Timing for download

### **Download Command**

This command downloads a program to the RAM for execution. Execute the download command as explained here following.

- (1) Send the "FA16" command code in the 1st byte of the transmission.
- (2) Send the program size in the 2nd and 3rd bytes of the transmission.
- (3) Send the check sum in the 4th byte of the transmission. The check sum is added to all data sent in the 5th byte onward.
- (4) The program to execute is sent in the 5th byte onward.

When all data has been transmitted, if the check sum matches, the downloaded program is executed. The size of the program will vary according to the internal RAM.



### Version Information Output Command

This command outputs the version information of the control program stored in the boot area. Execute the version information output command as explained here following.

- (1) Send the "FB16" command code in the 1st byte of the transmission.
- (2) The version information will be output from the 2nd byte onward. This data is composed of 8 ASCII code characters.

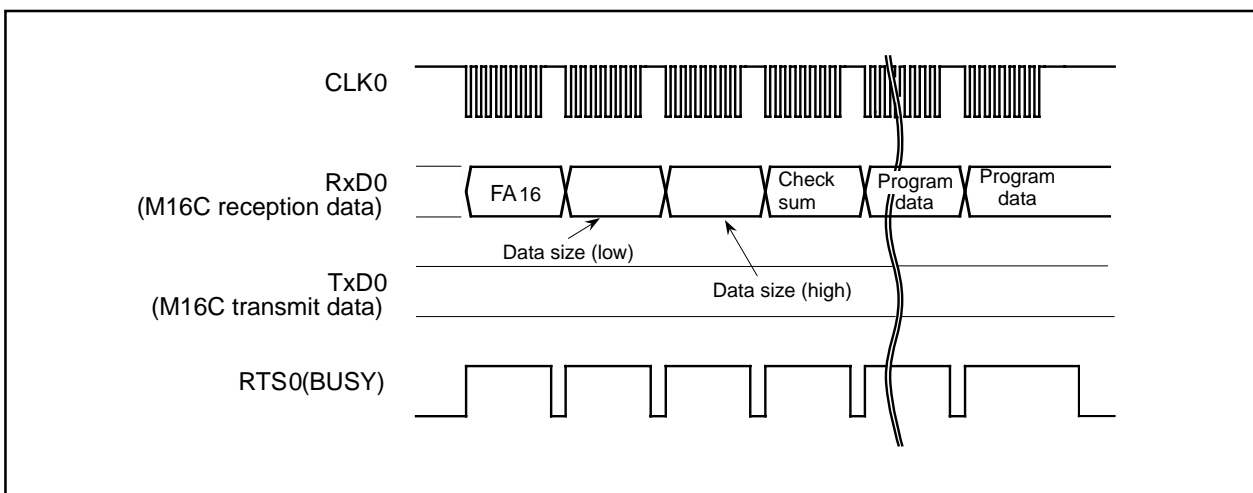


Figure 7.4.9. Timing for version information output

### Boot Area Output Command

This command outputs the control program stored in the boot area in one page blocks (256 bytes). Execute the boot area output command as explained here following.

- (1) Send the "FC16" command code in the 1st byte of the transmission.
- (2) Send addresses A8 to A15 and A16 to A23 in the 2nd and 3rd bytes of the transmission respectively.
- (3) From the 4th byte onward, data (D0–D7) for the page (256 bytes) specified with addresses A8 to A23 will be output sequentially from the smallest address first, in sync with the rise of the clock.

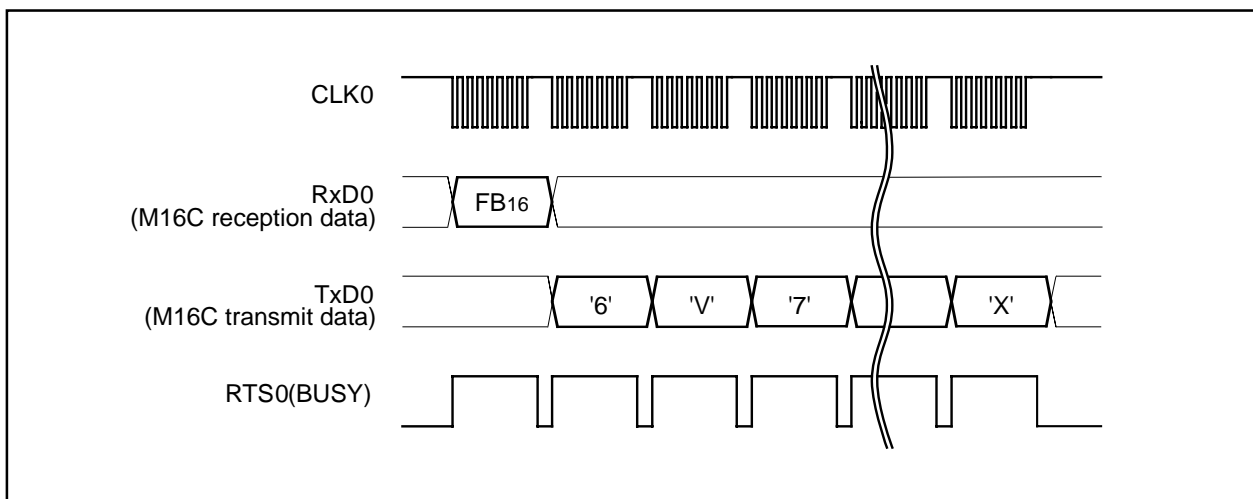
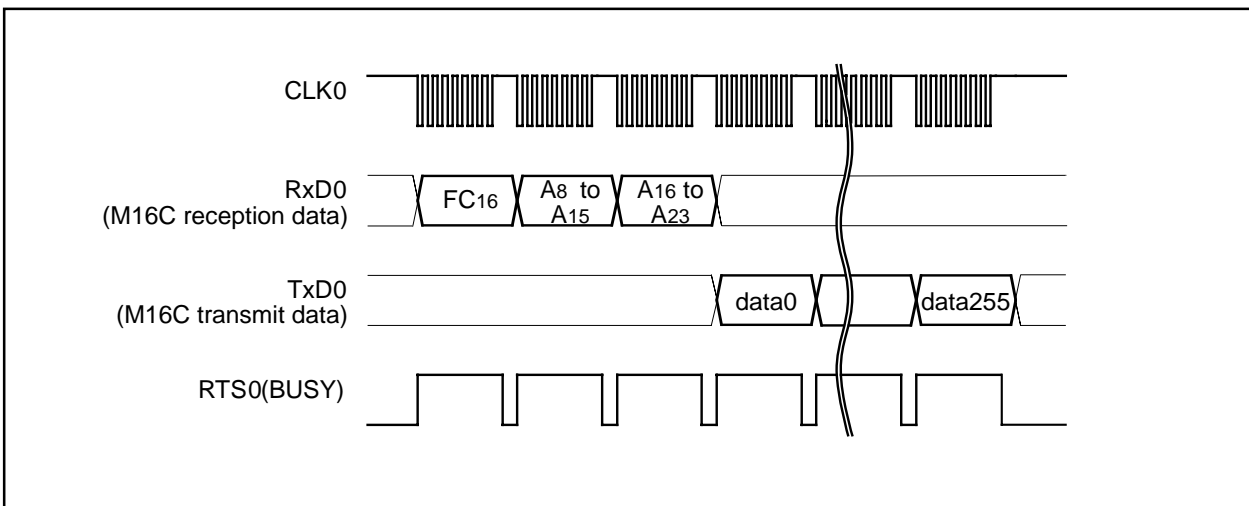


Figure 7.4.10. Timing for boot area output

**ID Check**

This command checks the ID code. Execute the boot ID check command as explained here following.

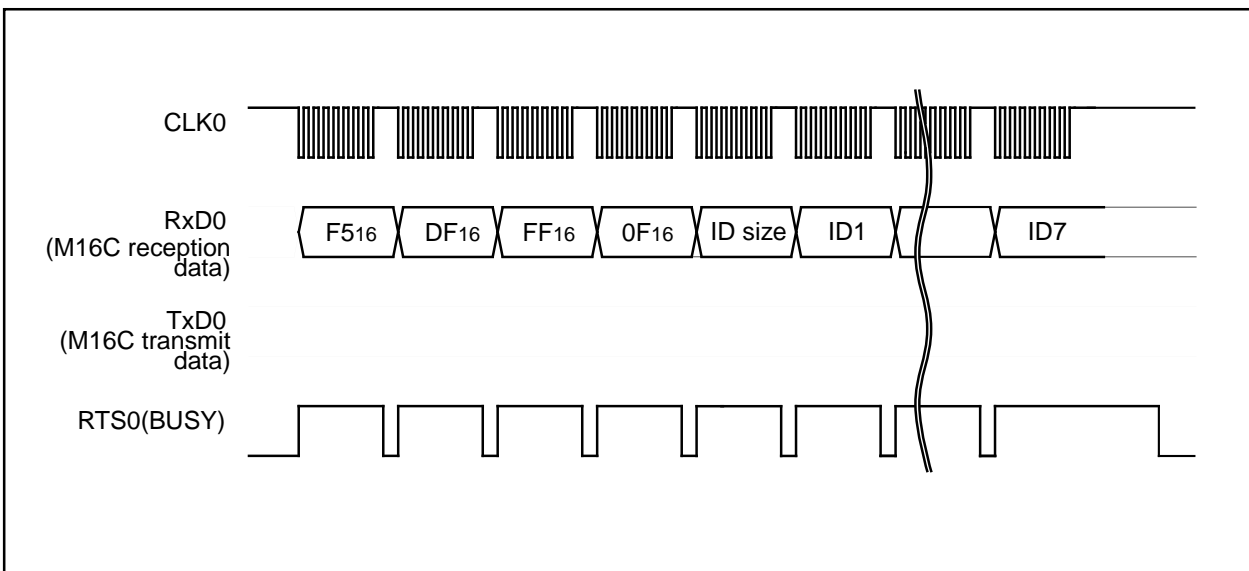
- (1) Send the "F516" command code in the 1st byte of the transmission.
- (2) Send addresses A0 to A7, A8 to A15 and A16 to A23 of the 1st byte of the ID code in the 2nd, 3rd and 4th bytes of the transmission respectively.
- (3) Send the number of data sets of the ID code in the 5th byte.
- (4) The ID code is sent in the 6th byte onward, starting with the 1st byte of the code.



**Figure 7.4.11. Timing for the ID check**

**ID Code**

When the flash memory is not blank, the ID code sent from the peripheral unit and the ID code written in the flash memory are compared to see if they match. If the codes do not match, the command sent from the peripheral unit is not accepted. An ID code contains 8 bits of data. Area is, from the 1st byte, addresses 0FFFDF16, 0FFFE316, 0FFFE216, 0FFFE116, 0FFFF316, 0FFFF716 and 0FFFFB16. Write a program into the flash memory, which already has the ID code set for these addresses.



**Figure 7.4.12. ID code storage addresses**

## Status Register (SRD)

The status register indicates operating status of the flash memory and status such as whether an erase operation or a program ended successfully or in error. It can be read by writing the read status register command (7016). Also, the status register is cleared by writing the clear status register command (5016). Table 7.4.2 gives the definition of each status register bit. After clearing the reset, the status register outputs "8016".

**Table 7.4.2. Status register (SRD)**

SRD0 bits	Status name	Definition	
		"1"	"0"
SR7 (bit7)	Write state machine (WSM) status	Ready	Busy
SR6 (bit6)	Reserved	-	-
SR5 (bit5)	Erase status	Terminated in error	Terminated normally
SR4 (bit4)	Program status	Terminated in error	Terminated normally
SR3 (bit3)	Block status after program	Terminated in error	Terminated normally
SR2 (bit2)	Reserved	-	-
SR1 (bit1)	Reserved	-	-
SR0 (bit0)	Reserved	-	-

### Write State Machine (WSM) Status (SR7)

The write state machine (WSM) status indicates the operating status of the flash memory. When power is turned on, "1" (ready) is set for it. The bit is set to "0" (busy) during an auto write or auto erase operation, but it is set back to "1" when the operation ends.

### Erase Status (SR5)

The erase status reports the operating status of the auto erase operation. If an erase error occurs, it is set to "1". When the erase status is cleared, it is set to "0".

### Program Status (SR4)

The program status reports the operating status of the auto write operation. If a write error occurs, it is set to "1". When the program status is cleared, it is set to "0".

### **Block Status After Program (SR3)**

If excessive data is written (phenomenon whereby the memory cell becomes depressed which results in data not being read correctly), "1" is set for the program status after-program at the end of the page write operation. In other words, when writing ends successfully, "80<sub>16</sub>" is output; when writing fails, "90<sub>16</sub>" is output; and when excessive data is written, "88<sub>16</sub>" is output.

If "1" is written for any of the SR5, SR4 or SR3 bits, the page program, block erase, erase all unlocked blocks and lock bit program commands are not accepted. Before executing these commands, execute the clear status register command (50<sub>16</sub>) and clear the status register.

## Status Register 1 (SRD1)

Status register 1 indicates the status of serial communications, results from ID checks and results from check sum comparisons. It can be read after the SRD by writing the read status register command (7016). Also, status register 1 is cleared by writing the clear status register command (5016).

Table 7.4.3 gives the definition of each status register 1 bit. "0016" is output when power is turned ON and the flag status is maintained even after the reset.

**Table 7.4.3. Status register 1 (SRD1)**

SRD1 bits	Status name	Definition	
		"1"	"0"
SR15 (bit7)	Boot update completed bit	Update completed	Not update
SR14 (bit6)	Reserved	-	-
SR13 (bit5)	Reserved	-	-
SR12 (bit4)	Check sum match bit	Match	Mismatch
SR11 (bit3) SR10 (bit2)	ID check completed bits	00 01 10 11	Not verified Verification mismatch Reserved Verified
SR9 (bit1)	Data receive time out	Time out	Normal operation
SR8 (bit0)	Selection area	OSD ROM	USER ROM

### Boot Update Completed Bit (SR15)

This flag indicates whether the control program was downloaded to the RAM or not, using the download function.

### Check Sum Consistency Bit (SR12)

This flag indicates whether the check sum matches or not when a program, is downloaded for execution using the download function.

### ID Check Completed Bits (SR11 and SR10)

These flags indicate the result of ID checks. Some commands cannot be accepted without an ID check.

### Data Reception Time Out (SR9)

This flag indicates when a time out error is generated during data reception. If this flag is attached during data reception, the received data is discarded and the microcomputer returns to the command wait state.

### Selection area (SR8)

It is the flag which shows ROM area under present selection.

### Full Status Check

Results from executed erase and program operations can be known by running a full status check. Figure 7.4.13 shows a flowchart of the full status check and explains how to remedy errors which occur.

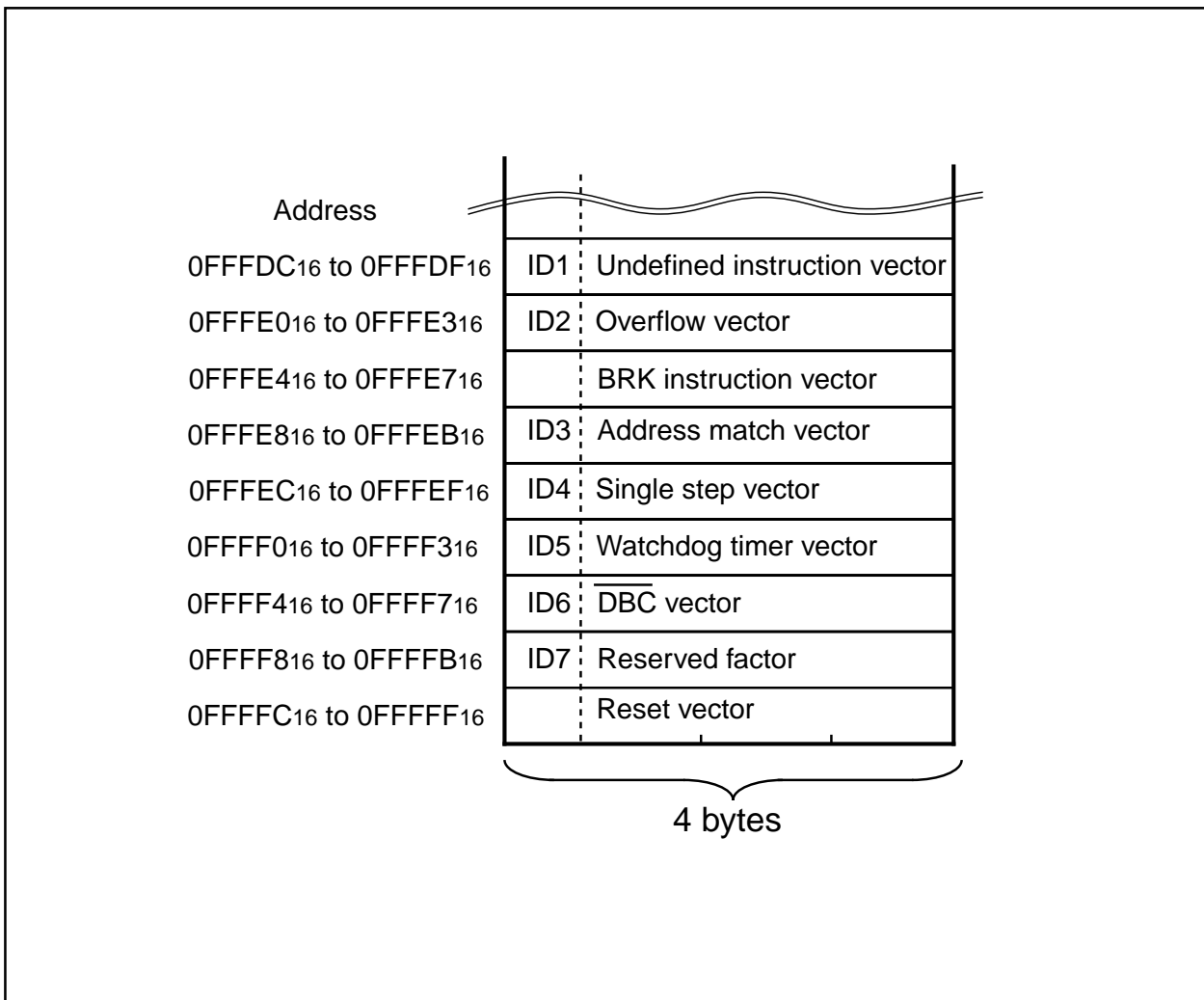


Figure 7.4.13. Full status check flowchart and remedial procedure for errors

### Example Circuit Application for The Standard Serial I/O Mode

The below figure shows a circuit application for the standard serial I/O mode. Control pins will vary according to peripheral unit (programmer), therefore see the peripheral unit (programmer) manual for more information.

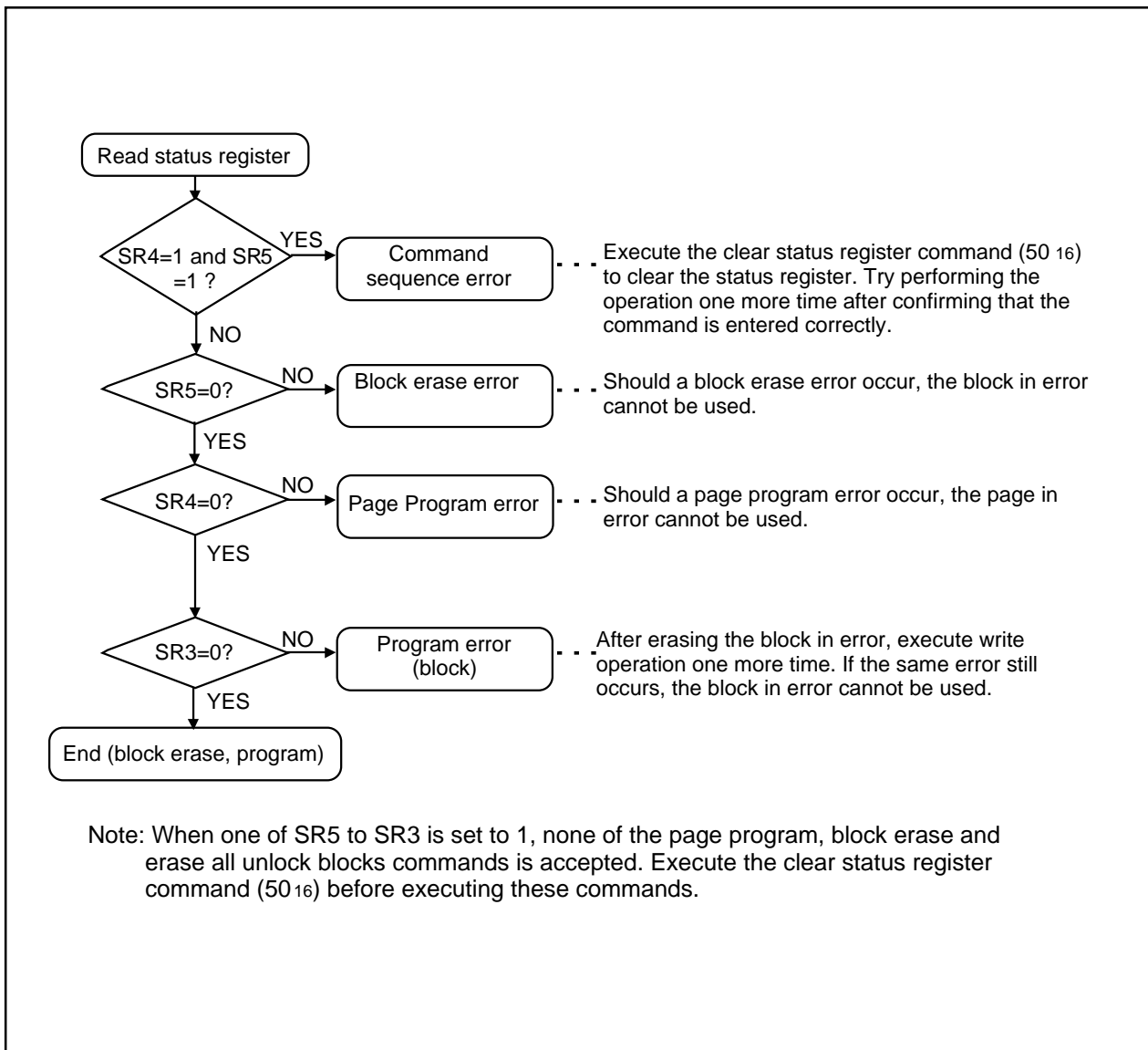


Figure 7.4.14. Example circuit application for the standard serial I/O mode

## 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 (Note 2)

Bit attributes (Note 2)

Symbol: PM1

Address: 000516

00000X002

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

Bit symbol	Bit name	Function	R	W
	Reserved bit	Must always be set to "0"	0	0
	Reserved bit	Must always be set to "1"	0	0
	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"	-	0
PM17	Wait bit	0 : No wait state 1 : Wait state inserted	0	0

**Notes 1:** Set bit 1 of the protect register (address 000A16) to "1" when writing new values to this register.

**2:** As this bit becomes "0" at reset, must always be set to "1" after reset release.

: 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**

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

**W.....Write**

- 0.....Write enabled
- X.....Write disabled
- .....Bit in which nothina is assianed



Sales Strategic Planning Div. Nippon Bldg., 2-6-2, Ohte-machi, Chiyoda-ku, Tokyo 100-0004, Japan

**Keep safety first in your circuit designs!**

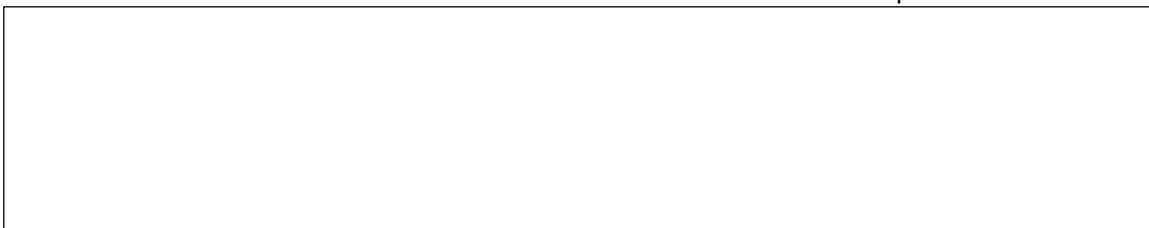
1. Renesas Technology Corporation puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage. Remember to give due consideration 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>



Copyright © 2003. Renesas Technology Corporation, All rights reserved. Printed in Japan.

REVISION DESCRIPTION LIST

M306V7MG/MH-XXXFP, M306V7FG/FHFP

Rev. No.	Revision Description	Rev. date
1.00	First Edition of PDF File	0709
	P63 Interrupt control reserved register i Symbol REiIC (i = 0 to 3) ] ---> REiIC (i = 1 to 3) ] Address 004516, 004616, 004716, 005F16 ---> Address 004616, 004716, 005F16	0709