To all our customers

# Regarding the change of names mentioned in the document, such as Mitsubishi Electric and Mitsubishi XX, to Renesas Technology Corp.

The semiconductor operations of Hitachi and Mitsubishi Electric were transferred to Renesas Technology Corporation on April 1st 2003. These operations include microcomputer, logic, analog and discrete devices, and memory chips other than DRAMs (flash memory, SRAMs etc.) Accordingly, although Mitsubishi Electric, Mitsubishi Electric Corporation, Mitsubishi Semiconductors, and other Mitsubishi brand names are mentioned in the document, these names have in fact all been changed to Renesas Technology Corp. Thank you for your understanding. Except for our corporate trademark, logo and corporate statement, no changes whatsoever have been made to the contents of the document, and these changes do not constitute any alteration to the contents of the document itself.

Note : Mitsubishi Electric will continue the business operations of high frequency & optical devices and power devices.

Renesas Technology Corp.
Customer Support Dept.
April 1, 2003

# M306H1SFP

SINGLE-CHIP 16-BIT CMOS MICROCOMPUTER
with DATA SLICER and ON-SCREEN DISPLAY CONTROLLER

## 1. Description

The M306H1SFP is single-chip microcomputer using the high-performance silicon gate CMOS process using a M16C/60 Series CPU core and is packaged in a 144-pin plastic molded QFP. This single-chip microcomputer operates using sophisticated instructions featuring a high level of instruction efficiency. With 1M bytes of address space, this is capable of executing instructions at high speed. This also features a built-in OSD display function and data slicer, making this correspondence to Teletext broadcasting service. This microcomputer is ROM less article, it can be used only at microprocessor mode.

### 1.1 Features

- Memory capacity ..... <ROM>ROM less
  <RAM>5K bytes
- Shortest instruction execution time ..... 100 ns (f($X_{IN}$)=10 MHz)
- Supply voltage ..... 4.75 V to 5.25V(at f($X_{IN}$)=10 MHz)
- Interrupts ..... 25 internal and 8 external interrupt sources, 4 software interrupt sources; 7 levels (Including key input interrupt)
- Multifunction 16-bit timer ..... 5 output timers + 6 input timers
- Serial I/O ..... 5 channels
  UART/clock synchronous: 3
  Clock synchronous: 2
- DMAC ..... 2 channels (trigger: 24 sources)
- A-D converter ..... 8 bits X 8 channels (Expandable up to 10 channels)
- D-A converter ..... 8 bits X 2 channels
- CRC calculation circuit ..... 1 circuit
- Watchdog timer ..... 1 line
- Programmable I/O ..... 50 lines
- Input port ..... 1 port ($P8_5$ shared with $\overline{NMI}$ pin)
- Chip select output ..... 3 lines
- Clock generating circuit ..... 2 built-in circuits
  (built-in feedback resistor, and external ceramic or crystal oscillator)
- OSD function ..... Screen composition 40 characters X 25 lines
  Characters available
  Font RAM : 256 characters, SYRAM : 15 characters
- Data slicer ..... For PDC, VPS and VBI
- Encoder ..... For VBI

### 1.2 Applications

VCR, etc

## Table of contents

## 1.3 Pin Configuration

Figures 1.3.1 shows the pin configuration (top view).

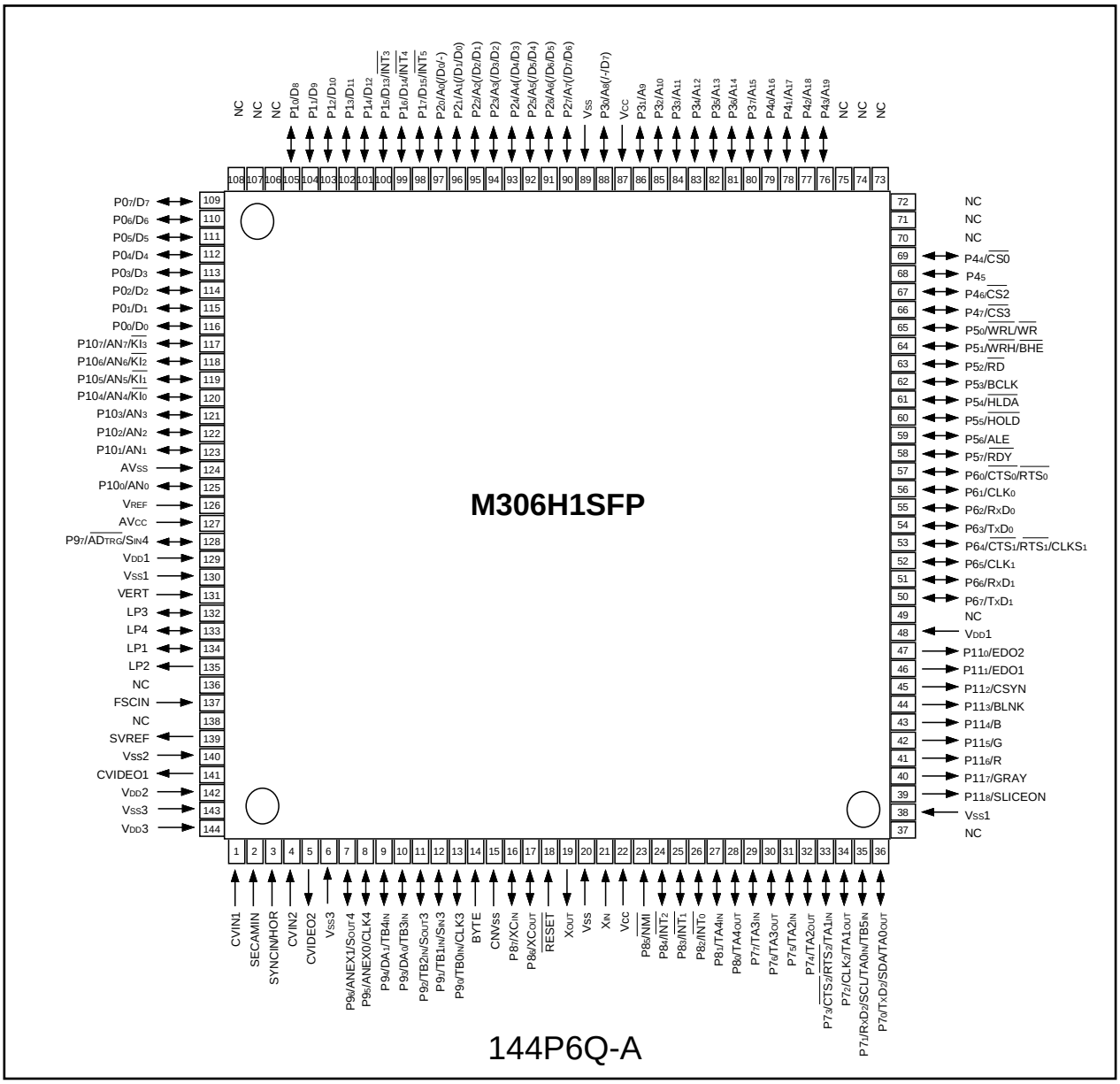



**Figure 1.3.1 Pin configuration (top view)**

## 1.4 Block Diagram

Figure 1.4.1 is a block diagram of the M306H1SFP.

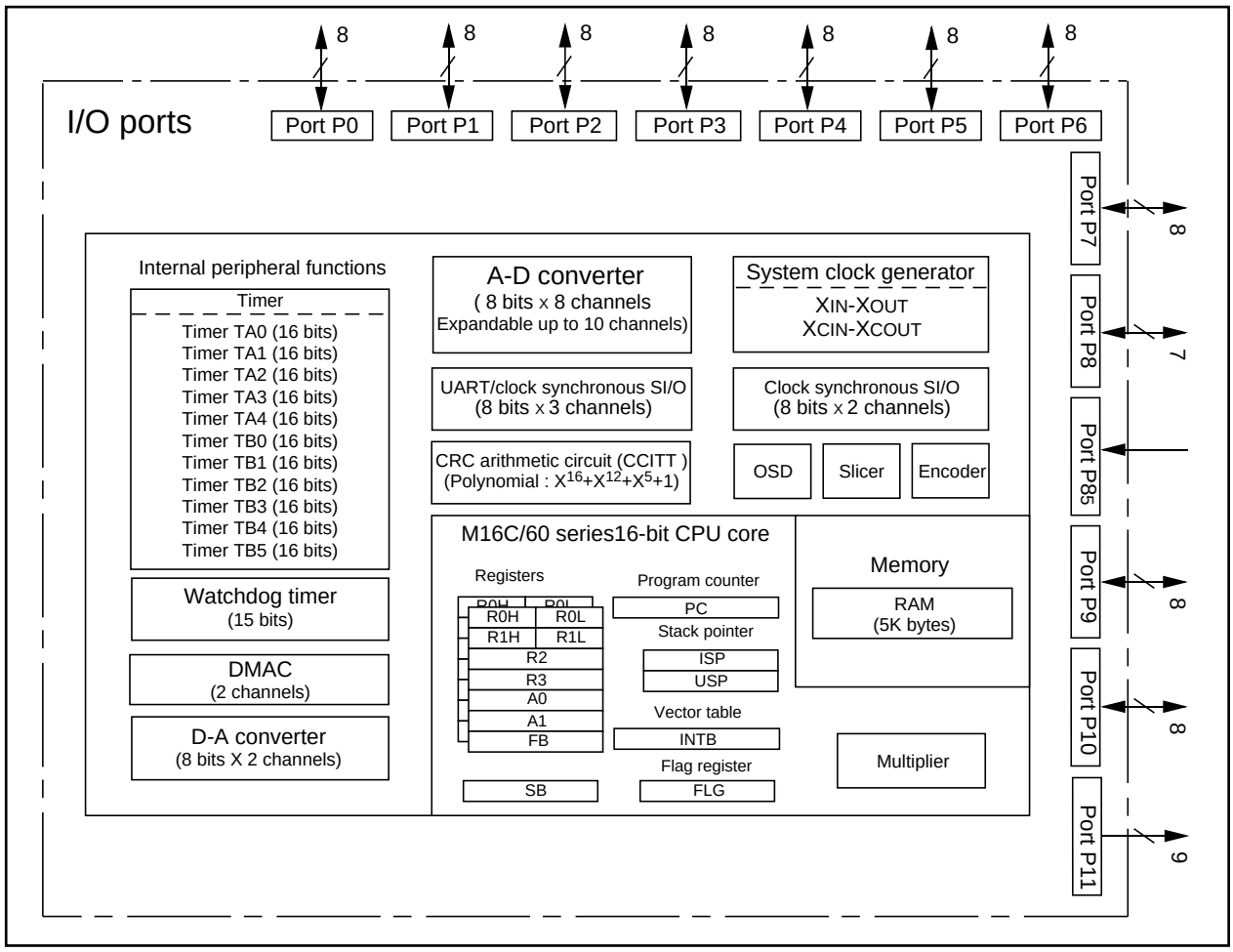



**Figure 1.4.1 Block diagram of M306H1SFP**

## 1.5 Performance Outline

Table 1.5.1 is a performance outline of M306H1SFP.

**Table 1.5.1 Performance outline of M306H1SFP**

| Item | | Performance |
|---|---|---|
| Number of basic instructions | | 91 instructions |
| Shortest instruction execution time | | 100ns (f(XIN)=10MHz) |
| Memory capacity | ROM | — |
| | RAM | 5K bytes |
| I/O port | P0 to P10 (except P85) | 8 bits x 10, 7 bits x 1 |
| Input port | P85 | 1 bit x 1 |
| Output port | P11 | 9 bit x 1 |
| Multifunction timer | TA0, TA1, TA2, TA3, TA4 | 16 bits x 5 |
| | TB0, TB1, TB2, TB3, TB4, TB5 | 16 bits x 6 |
| Serial I/O | UART0, UART1, UART2 | (UART or clock synchronous) x 3 |
| | SI/O3, SI/O4 | (Clock synchronous) x 2 |
| A-D converter | | 8 bits x (8 + 2) channels |
| D-A converter | | 8 bits x 2 channels |
| DMAC | | 2 channels (trigger: 24 sources) |
| CRC calculation circuit | | CRC-CCITT |
| Watchdog timer | | 15 bits x 1 (with prescaler) |
| Interrupt | | 25 internal and 8 external sources, 4 software sources, 7 levels |
| Clock generating circuit | | 2 built-in clock generation circuits (built-in feedback resistor, and external ceramic or crystal oscillator) |
| Supply voltage | | 4.75 to 5.25V (f(XIN)=10MHz) |
| Device configuration | | CMOS high performance silicon gate |
| Package | | 144-pin plastic mold QFP |
| OSD function | OSD display RAM | 2.75K Bytes (25 x 40 x 22-bit) |
| | Font RAM | 3.84K Bytes (12 x 10 x 256-bit) |
| | SYRAM | 260 Bytes (13 x 10 x 16-bit) |
| | Screen composition | 40 characters x 25 lines |
| | Character composition | 12 x 10 dots matrix |
| | Character coloring | 8 colors choices per character |
| | Character Background coloring | 8 colors choices per character |
| | Background coloring | 8 colors choices per screen |
| | SYRAM color | 8 colors choices per character |
| | Character Background coloring | 8 colors choices per character |
| | Synchronous signal | PAL |
| | Video signal | PAL |
| Data slicer | Slice RAM | 864 Bytes (48 x 18 x 8-bit) |
| | VBIRAM | 95 Bytes ((5 + 5 x 18) x 8-bit) |
| | Data slicer | for PDC, VPS and VBI |
| | Encoder | for VBI |

**Table 1.5.2 Pin Description**

| Pin name | Signal name | I/O type | Function |
|---|---|---|---|
| $V_{CC}$, $V_{SS}$ | Power supply input | | Supply 4.75 to 5.25 V to the Vcc pin. Supply 0 V to the Vss pin. |
| $CNV_{SS}$ | $CNV_{SS}$ | Input | This pin switches between processor modes. Connect it to the $V_{CC}$ pin when in microprocessor mode. |
| $\overline{RESET}$ | Reset input | Input | A "L" on this input resets the microcomputer. |
| $X_{IN}$<br>$X_{OUT}$ | Clock input<br>Clock output | Input<br>Output | These pins are provided for the main clock generating circuit. Connect a ceramic resonator or crystal between the $X_{IN}$ and the $X_{OUT}$ pins. To use an externally derived clock, input it to the $X_{IN}$ pin and leave the $X_{OUT}$ pin open. |
| BYTE | External data bus width select input | Input | This pin selects the width of an external data bus. A 16-bit width is selected when this input is "L"; an 8-bit width is selected when this input is "H". This input must be fixed to either "H" or "L". |
| $AV_{CC}$ | Analog power supply input | | This pin is a power supply input for the A-D converter. Connect this pin to $V_{CC}$. |
| $AV_{SS}$ | Analog power supply input | | This pin is a power supply input for the A-D converter. Connect this pin to $V_{SS}$. |
| $V_{REF}$ | Reference voltage input | Input | This pin is a reference voltage input for the A-D converter. |
| $P0_0$ to $P0_7$ | I/O port P0 | Input/output | This is an 8-bit CMOS I/O port. It has an input/output port direction register that allows the user to set each pin for input or output individually. |
| $D_0$ to $D_7$ | | Input/output | When set as a separate bus, these pins input and output data ($D_0$–$D_7$). |
| $P1_0$ to $P1_7$ | I/O port P1 | Input/output | This is an 8-bit I/O port equivalent to P0. Pins in this port also function as external interrupt pins as selected by software. |
| $D_8$ to $D_{15}$ | | Input/output | When set as a separate bus, these pins input and output data ($D_8$–$D_{15}$). |
| $P2_0$ to $P2_7$ | I/O port P2 | Input/output | This is an 8-bit I/O port equivalent to P0. |
| $A_0$ to $A_7$ | | Output | These pins output 8 low-order address bits ($A_0$–$A_7$). |
| $A_0/D_0$ to $A_7/D_7$ | | Input/output | If the external bus is set as an 8-bit wide multiplexed bus, these pins input and output data ($D_0$–$D_7$) and output 8 low-order address bits ($A_0$–$A_7$) separated in time by multiplexing. |
| $A_0$, $A_1/D_0$ to $A_7/D_6$ | | Output<br>Input/output | If the external bus is set as a 16-bit wide multiplexed bus, these pins input and output data ($D_0$–$D_6$) and output address ($A_1$–$A_7$) separated in time by multiplexing. They also output address ($A_0$). |
| $P3_0$ to $P3_7$ | I/O port P3 | Input/output | This is an 8-bit I/O port equivalent to P0. |
| $A_8$ to $A_{15}$ | | Output | These pins output 8 middle-order address bits ($A_8$–$A_{15}$). |
| $A_8/D_7$, $A_9$ to $A_{15}$ | | Input/output<br>Output | If the external bus is set as a 16-bit wide multiplexed bus, these pins input and output data ($D_7$) and output address ($A_8$) separated in time by multiplexing. They also output address ($A_9$–$A_{15}$). |
| $P4_0$ to $P4_7$ | I/O port P4 | Input/output | This is an 8-bit I/O port equivalent to P0. |
| $\overline{CS_0}$,$\overline{CS_2}$,$\overline{CS_3}$, $A_{16}$ to $A_{19}$ | | Output<br>Output | These pins output $\overline{CS_0}$,$\overline{CS_2}$,$\overline{CS_3}$ signals and $A_{16}$–$A_{19}$. $\overline{CS_0}$,$\overline{CS_2}$,$\overline{CS_3}$ are chip select signals used to specify an access space. $A_{16}$–$A_{19}$ are 4 high- order address bits. |

**Table 1.5.3 Pin Description**

| Pin name | Signal name | I/O type | Function |
|---|---|---|---|
| P5$_0$ to P5$_7$ | I/O port P5 | Input/output | This is an 8-bit I/O port equivalent to P0. |
| $\overline{WRL}$ / $\overline{WR}$,<br>$\overline{WRH}$ / $\overline{BHE}$,<br>$\overline{RD}$,<br>BCLK,<br>$\overline{HLDA}$,<br>$\overline{HOLD}$,<br><br>ALE,<br>$\overline{RDY}$ | | Output<br>Output<br>Output<br>Output<br>Output<br>Input<br><br>Output<br>Input | Output $\overline{WRL}$, $\overline{WRH}$ ($\overline{WR}$ and $\overline{BHE}$), $\overline{RD}$, BCLK, $\overline{HLDA}$, and ALE signals. $\overline{WRL}$ and $\overline{WRH}$, and $\overline{BHE}$ and $\overline{WR}$ can be switched using software control.<br>■ $\overline{WRL}$, $\overline{WRH}$, and $\overline{RD}$ selected<br>With a 16-bit external data bus, data is written to even addresses when the $\overline{WRL}$ signal is "L" and to the odd addresses when the $\overline{WRH}$ signal is "L". Data is read when $\overline{RD}$ is "L".<br>■ $\overline{WR}$, $\overline{BHE}$, and $\overline{RD}$ selected<br>Data is written when $\overline{WR}$ is "L". Data is read when $\overline{RD}$ is "L". Odd addresses are accessed when $\overline{BHE}$ is "L". Use this mode when using an 8-bit external data bus.<br>While the input level at the $\overline{HOLD}$ pin is "L", the microcomputer is placed in the hold state. While in the hold state, $\overline{HLDA}$ outputs a "L" level. ALE is used to latch the address. While the input level of the $\overline{RDY}$ pin is "L", the microcomputer is in the ready state. |
| P6$_0$ to P6$_7$ | I/O port P6 | Input/output | This is an 8-bit I/O port equivalent to P0. The port can be set to have or not have a pull-up resistor in units of four bits by software. Pins in this port also function as UART0 and UART1 I/O pins as selected by software. |
| P7$_0$ to P7$_7$ | I/O port P7 | Input/output | This is an 8-bit I/O port equivalent to P6 (P7$_0$ and P7$_1$ are N channel open-drain output). Pins in this port also function as timer A$_0$–A$_3$, timer B5 or UART2 I/O pins as selected by software. |
| P8$_0$ to P8$_4$,<br>P8$_6$,<br>P8$_7$,<br>P8$_5$ | I/O port P8<br><br><br>I/O port P8$_5$ | Input/output<br>Input/output<br>Input/output<br>Input | P8$_0$ to P8$_4$, P8$_6$, and P8$_7$ are I/O ports with the same functions as P6. Using software, they can be made to function as the I/O pins for timer A4 and the input pins for external interrupts. P8$_6$ and P8$_7$ can be set using software to function as the I/O pins for a sub clock generation circuit. In this case, connect a quartz oscillator between P8$_6$ (XCOUT pin) and P8$_7$ (XCIN pin). P8$_5$ is an input-only port that also functions for $\overline{NMI}$. The $\overline{NMI}$ interrupt is generated when the input at this pin changes from "H" to "L". The $\overline{NMI}$ function cannot be cancelled using software. The pull-up cannot be set for this pin. |
| P9$_0$ to P9$_7$ | I/O port P9 | Input/output | This is an 8-bit I/O port equivalent to P6. Pins in this port also function as SI/O3, 4 I/O pins, Timer B0–B4 input pins, D-A converter output pins, A-D converter extended input pins, or A-D trigger input pins as selected by software. |
| P10$_0$ to P10$_7$ | I/O port P10 | Input/output | This is an 8-bit I/O port equivalent to P6. Pins in this port also function as A-D converter input pins. Furthermore, P10$_4$–P10$_7$ also function as input pins for the key input interrupt function. |
| VDD1 | Power supply input | | Digital power supply pin. Connect to +5 V. |
| VDD2 | Power supply input | | Analog power supply pin. Connect to +5 V. |
| VDD3 | Power supply input | | Analog power supply pin. Connect to +5 V. |
| CVIDEO1 | Composite video output 1 | Output | This is composite video signal output pin. Output 2 Vp-p composite video signal. In superimpose mode, this pin's signal consists of CVIN1 signal of the display range combined with the character output signal. |

**Table 1.5.4 Pin Description**

| Pin name | Signal name | I/O type | Function |
|---|---|---|---|
| CVIDEO2 | Composite video output 2 | Output | This is composite video signal output pin. Output 2 Vp-p composite video signal. This pin's signal consists of CVIN2 signal of vertical blanking erase interval combined with the VBI output signal. |
| SVREF | Synchronous slice level input | Input | When slice the vertical synchronous signal, input slice power. |
| CVIN1 | Composite video signal input 1 | Input | This pin inputs the external composite video signal. In superimpose mode, this pin's signal consists of it's composite video signal combined with the character output signal. Data slices this signal internally by setting. |
| SECAMIN | SECAM input | Input | Carrier input pin for SECAM. |
| CVIN2 | Composite video signal input 2 | Input | This pin inputs the external composite video signal. In VBI encode, this pin's signal consists of it's composite video signal combined with the VBI output signal. Data slices this signal internally by setting. |
| SYNCIN | Composite video signal input 3 | Input | This pin inputs the external composite video signal. Synchronous devides this signal internally. |
| HOR | | | Input digital horizontal synchronous signal (5 V). |
| LP1 | Filter output 1 | Output | This is filter output pin 1 (for display). |
| LP2 | Filter output 2 | Output | This is filter output pin 2 (for synchronous). |
| LP3 | Filter output 3 | Output | This is filter output pin 3 (for VBI, VPS). |
| LP4 | Filter output 4 | Output | This is filter output pin 4 (for PDC). |
| FSCIN | fsc input pin for synchronous signal generation | Input | Sub-carrier (fsc) input pin for synchronous signal generation. |
| VERT | Vertical synchronous signal input | Input | Digital vertical synchronous signal input (5 V). |
| $P11_0$ to $P11_8$ | Output port P11 | Output | This is a 9-bit output-only port. Pins in this port also function as EDO2, EDO1,CSYN,BLNK,B,G,R,GRAY,SLICEON output pins as selected by software. |

## 2. Operation of Functional Blocks

The M306H0SFP accommodates certain units in a single chip. These units include RAM to store instructions and data and the central processing unit (CPU) to execute arithmetic/logic operations. Also included are peripheral units such as timers, serial I/O, D-A converter, DMAC, CRC calculation circuit, A-D converter, OSD circuit, Data slicer circuit, Data encode circuit and I/O ports.
The following explains each unit.

### 2.1 Memory

Figure 2.1.1 is a memory map of the M306H0SFP. The address space extends the 1M bytes from address $00000_{16}$ to $FFFFF_{16}$. From address $FFFFF_{16}$ down is ROM. In the M306H0SFP, can use from address from $04000_{16}$ to $FFFFF_{16}$ as external ROM area. The vector table for fixed interrupts such as the reset and $\overline{NMI}$ are mapped to from address $FFFDC_{16}$ to $FFFFF_{16}$. The starting address of the interrupt routine is stored here. The address of the vector table for timer interrupts, etc., can be set as desired using the internal register (INTB). See the section on interrupts for details.
5K bytes of internal RAM is mapped to from address $00400_{16}$ to $017FF_{16}$. In addition to storing data, the RAM also stores the stack used when calling subroutines and when interrupts are generated.
The SFR area is mapped to from address $00000_{16}$ 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 2.1.4 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 from address $FFE00_{16}$ 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.
Address $01800_{16}$ to $03FFF_{16}$ and address $28000_{16}$ to $2FFFF_{16}$ are reserved and cannot be used.

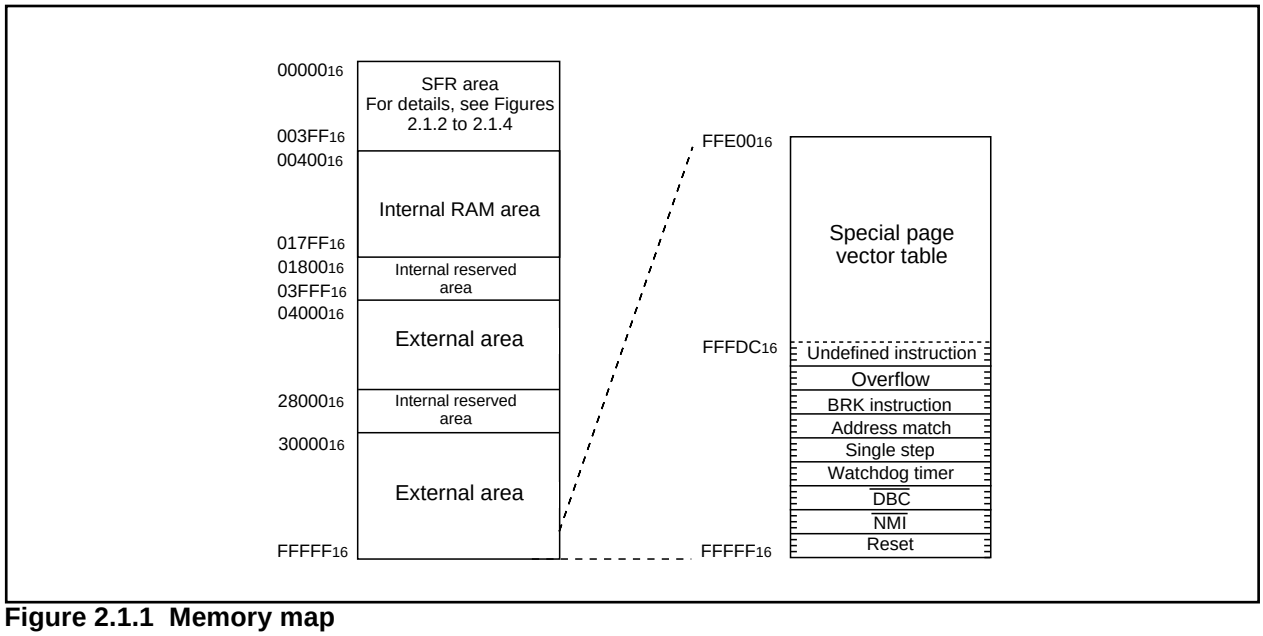



**Figure 2.1.1 Memory map**

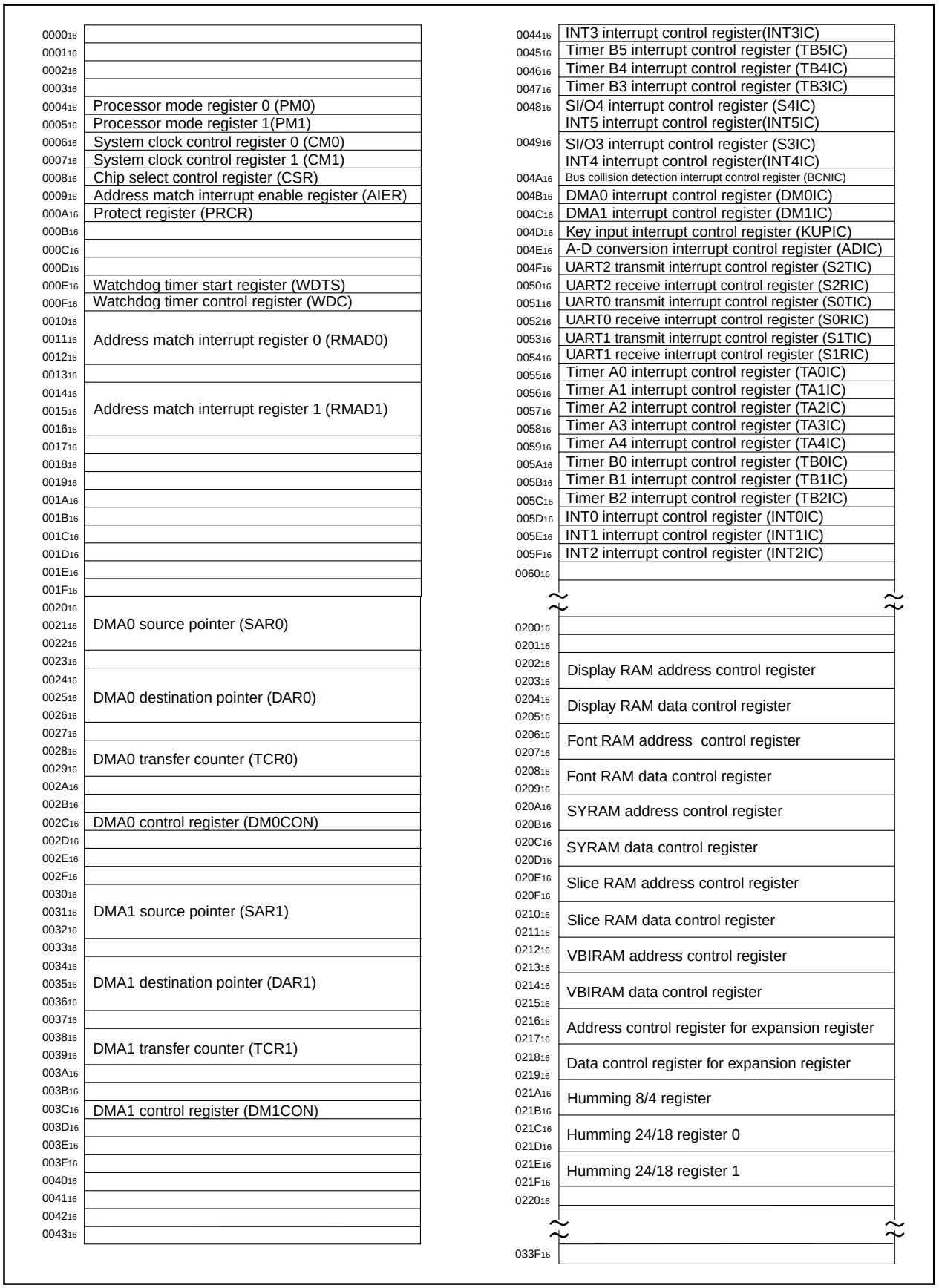



**Figure 2.1.2 Location of peripheral unit control registers (1)**

| | |
|---|---|
| 0340₁₆ | Timer B3, 4, 5 count start flag (TBSR) |
| 0341₁₆ | |
| 0342₁₆ 0343₁₆ | Timer A1-1 register (TA11) |
| 0344₁₆ 0345₁₆ | Timer A2-1 register (TA21) |
| 0346₁₆ 0347₁₆ | Timer A4-1 register (TA41) |
| 0348₁₆ | Reserved register |
| 0349₁₆ | Reserved register |
| 034A₁₆ | Reserved register |
| 034B₁₆ | Reserved register |
| 034C₁₆ | Reserved register |
| 034D₁₆ | Reserved register |
| 034E₁₆ | |
| 034F₁₆ | |
| 0350₁₆ 0351₁₆ | Timer B3 register (TB3) |
| 0352₁₆ 0353₁₆ | Timer B4 register (TB4) |
| 0354₁₆ 0355₁₆ | Timer B5 register (TB5) |
| 0356₁₆ | |
| 0357₁₆ | |
| 0358₁₆ | |
| 0359₁₆ | |
| 035A₁₆ | |
| 035B₁₆ | Timer B3 mode register (TB3MR) |
| 035C₁₆ | Timer B4 mode register (TB4MR) |
| 035D₁₆ | Timer B5 mode register (TB5MR) |
| 035E₁₆ | |
| 035F₁₆ | Interrupt cause select register (IFSR) |
| 0360₁₆ | SI/O3 transmit/receive register (S3TRR) |
| 0361₁₆ | |
| 0362₁₆ | SI/O3 control register (S3C) |
| 0363₁₆ | SI/O3 bit rate generator (S3BRG) |
| 0364₁₆ | SI/O4 transmit/receive register (S4TRR) |
| 0365₁₆ | |
| 0366₁₆ | SI/O4 control register (S4C) |
| 0367₁₆ | SI/O4 bit rate generator (S4BRG) |
| 0368₁₆ | |
| 0369₁₆ | |
| 036A₁₆ | |
| 036B₁₆ | |
| 036C₁₆ | |
| 036D₁₆ | |
| 036E₁₆ | |
| 036F₁₆ | |
| 0370₁₆ | |
| 0371₁₆ | |
| 0372₁₆ | |
| 0373₁₆ | |
| 0374₁₆ | |
| 0375₁₆ | UART2 special mode register 3(U2SMR3) |
| 0376₁₆ | UART2 special mode register 2(U2SMR2) |
| 0377₁₆ | UART2 special mode register (U2SMR) |
| 0378₁₆ | UART2 transmit/receive mode register (U2MR) |
| 0379₁₆ | UART2 bit rate generator (U2BRG) |
| 037A₁₆ 037B₁₆ | UART2 transmit buffer register (U2TB) |
| 037C₁₆ | UART2 transmit/receive control register 0 (U2C0) |
| 037D₁₆ | UART2 transmit/receive control register 1 (U2C1) |
| 037E₁₆ 037F₁₆ | UART2 receive buffer register (U2RB) |

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

**Figure 2.1.3 Location of peripheral unit control registers (2)**

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

**Figure 2.1.4 Location of peripheral unit control registers (3)**

## 2.2 Central Processing Unit (CPU)

The CPU has 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.

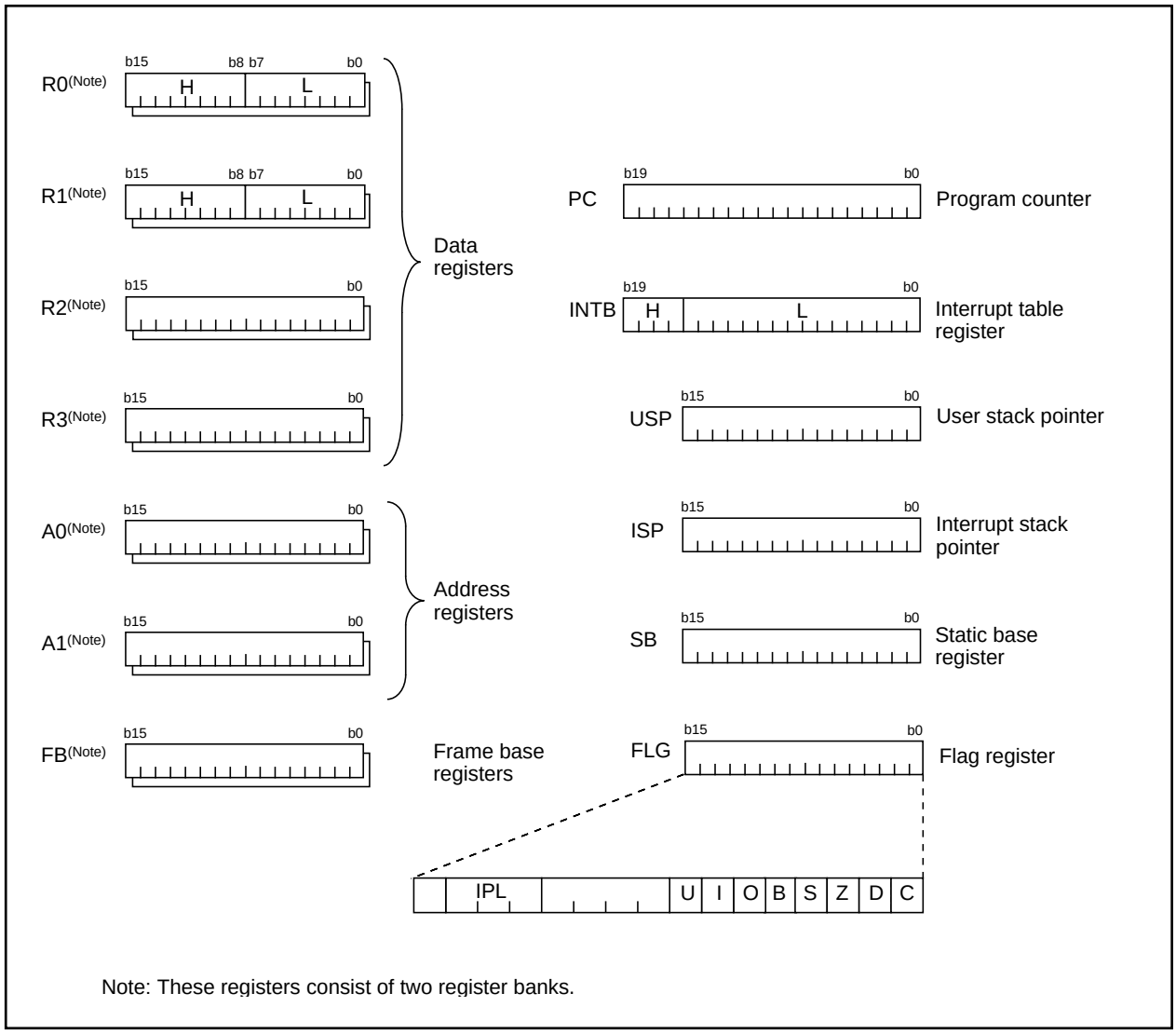



**Figure 2.2.1 Central processing unit register**

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

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

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

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

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

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

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

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

### (4) Program counter (PC)

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

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

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

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

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

Your desired type of stack pointer (USP or ISP) can be selected by a stack pointer select flag (U flag). This flag is located at the position of bit 7 in the flag register (FLG).

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

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

### (8) Flag register (FLG)

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

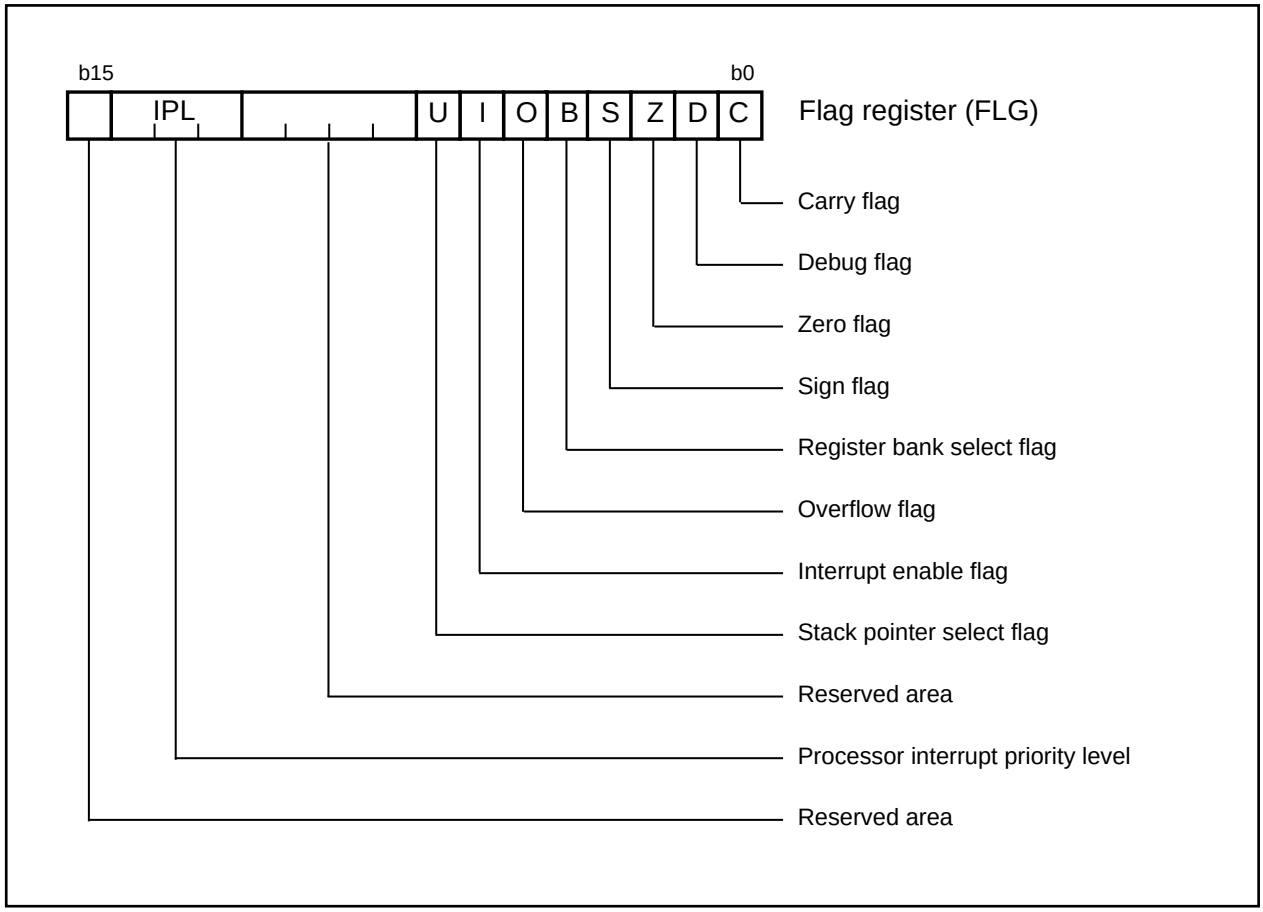



**Figure 2.2.2 Flag register (FLG)**

## 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” (0.2Vcc max.) for at least 20 cycles. When the reset pin level is then returned to the “H” level while main clock is stable, the reset status is cancelled and program execution resumes from the address in the reset vector table.

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

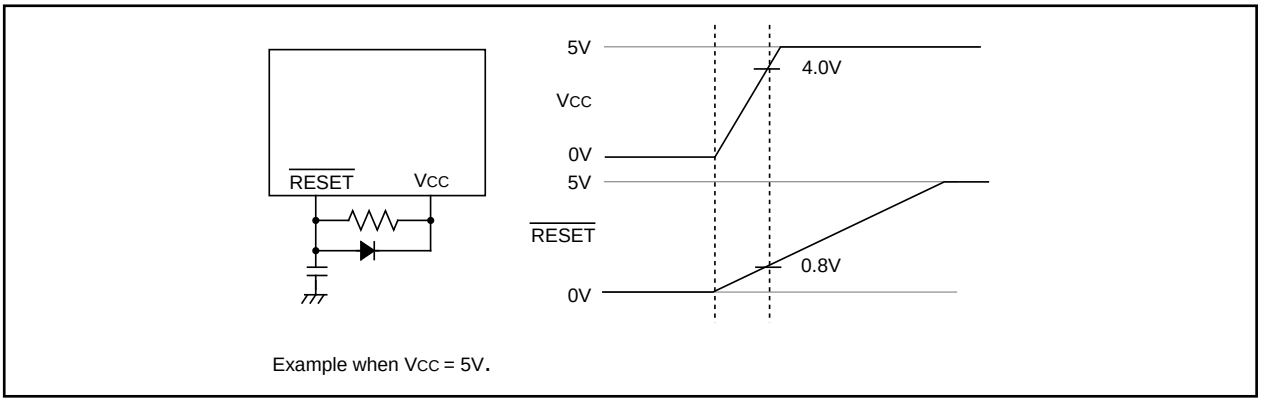



**Figure 2.3.1 Example reset circuit**

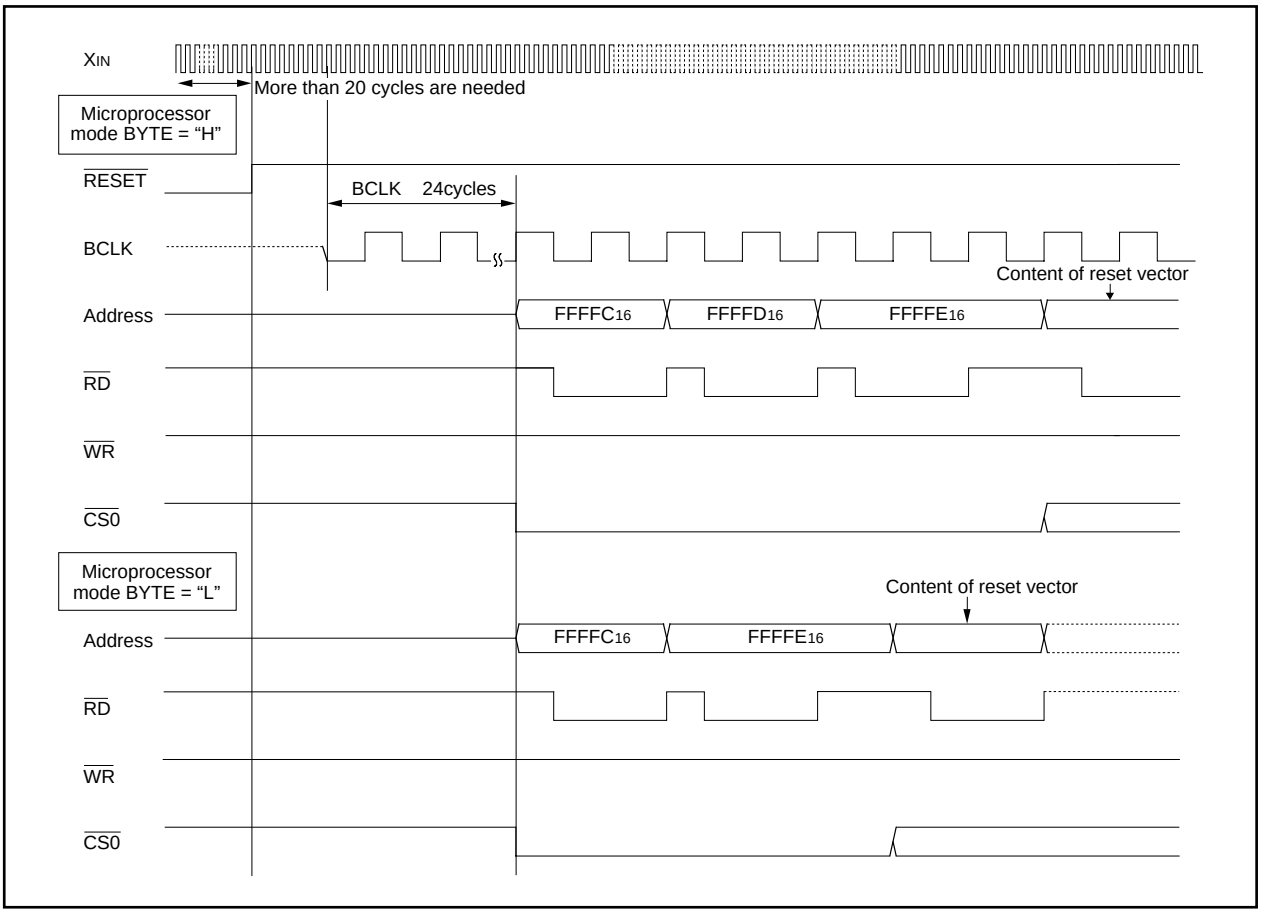



**Figure 2.3.2 Reset sequence**

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 = VCC | |
| | BYTE = VSS | BYTE = VCC |
| P0 | Data input (floating) | Data input (floating) |
| P1 | Data input (floating) | Input port (floating) |
| P2, P3, P40 to P43 | Address output (undefined) | Address output (undefined) |
| P44 | $\overline{\text{CS0}}$ output (“H” level is output) | $\overline{\text{CS0}}$ output (“H” level is output) |
| P45 to P47 | Input port (floating)<br>(pull-up resistor is on) | Input port (floating)<br>(pull-up resistor is on) |
| P50 | $\overline{\text{WR}}$ output (“H” level is output) | $\overline{\text{WR}}$ output (“H” level is output) |
| P51 | $\overline{\text{BHE}}$ output (undefined) | $\overline{\text{BHE}}$ output (undefined) |
| P52 | $\overline{\text{RD}}$ output (“H” level is output) | $\overline{\text{RD}}$ output (“H” level is output) |
| P53 | BCLK output | BCLK output |
| P54 | $\overline{\text{HLDA}}$ output (The output value depends on the input to the $\overline{\text{HOLD}}$ pin) | $\overline{\text{HLDA}}$ output (The output value depends on the input to the $\overline{\text{HOLD}}$ pin) |
| P55 | $\overline{\text{HOLD}}$ input (floating) | $\overline{\text{HOLD}}$ input (floating) |
| P56 | ALE output (“L” level is output) | ALE output (“L” level is output) |
| P57 | $\overline{\text{RDY}}$ input (floating) | $\overline{\text{RDY}}$ input (floating) |
| P6, P7, P80 to P84, P86, P87, P9, P10 | Input port (floating) | Input port (floating) |
| P110 to P118 | Output port | Output port |
| CVIDEO1, CVIDEO2 | Output port | Output port |
| CVIN1, CVIN2, SECAMIN, SVREF, SYNCIN, VERT, FSCIN | Input port | Input port |
| LP1, LP2, LP3, LP4 | Output port | Output port |

### 2.3.1 Software Reset

Writing “1” to bit 3 of the processor mode register 0 (address $0004_{16}$) applies a (software) reset to the microcomputer. A software reset has almost the same effect as a hardware reset. The contents of internal RAM are preserved.

| | | | | | |
|---|---|---|---|---|---|
| Processor mode register 0 (Note) | (0004$_{16}$)··· | 00$_{16}$ | Display RAM address control register | (0202$_{16}$)··· | 00$_{16}$ |
| Processor mode register 1 | (0005$_{16}$)··· | 0 0 0 0 0 x x 0 | | (0203$_{16}$)··· | 00$_{16}$ |
| System clock control register 0 | (0006$_{16}$)··· | 0 1 0 0 1 0 0 0 | Display RAM data control register | (0204$_{16}$)··· | 00$_{16}$ |
| System clock control register 1 | (0007$_{16}$)··· | 0 0 1 0 0 0 0 0 | | (0205$_{16}$)··· | 00$_{16}$ |
| Chip select control register | (0008$_{16}$)··· | 0 0 0 0 0 0 0 1 | Font RAM address control register | (0206$_{16}$)··· | 00$_{16}$ |
| Address match interrupt enable register | (0009$_{16}$)··· | x x x x x x 0 0 | | (0207$_{16}$)··· | 00$_{16}$ |
| Protect register | (000A$_{16}$)··· | x x x x x 0 0 0 | Font RAM data control register | (0208$_{16}$)··· | 00$_{16}$ |
| Watchdog timer control register | (000F$_{16}$)··· | 0 0 0 ? ? ? ? ? | | (0209$_{16}$)··· | 00$_{16}$ |
| Address match interrupt register 0 | (0010$_{16}$)··· | 00$_{16}$ | SYRAM address control register | (020A$_{16}$)··· | 00$_{16}$ |
| | (0011$_{16}$)··· | 00$_{16}$ | | (020B$_{16}$)··· | 00$_{16}$ |
| | (0012$_{16}$)··· | x x x x 0 0 0 0 | SYRAM data control register | (020C$_{16}$)··· | 00$_{16}$ |
| Address match interrupt register 1 | (0014$_{16}$)··· | 00$_{16}$ | | (020D$_{16}$)··· | 00$_{16}$ |
| | (0015$_{16}$)··· | 00$_{16}$ | Slice RAM address control register | (020E$_{16}$)··· | 00$_{16}$ |
| | (0016$_{16}$)··· | x x x x 0 0 0 0 | | (020F$_{16}$)··· | 00$_{16}$ |
| DMA0 control register | (002C$_{16}$)··· | 0 0 0 0 0 ? 0 0 | Slice RAM data control register | (0210$_{16}$)··· | 00$_{16}$ |
| DMA1 control register | (003C$_{16}$)··· | 0 0 0 0 0 ? 0 0 | | (0211$_{16}$)··· | 00$_{16}$ |
| INT3 interrupt control register | (0044$_{16}$)··· | x x 0 0 ? 0 0 0 | VBIRAM address control register | (0212$_{16}$)··· | 00$_{16}$ |
| Timer B5 interrupt control register | (0045$_{16}$)··· | x x x x ? 0 0 0 | | (0213$_{16}$)··· | 00$_{16}$ |
| Timer B4 interrupt control register | (0046$_{16}$)··· | x x x x ? 0 0 0 | VBIRAM data control register | (0214$_{16}$)··· | 00$_{16}$ |
| Timer B3 interrupt control register | (0047$_{16}$)··· | x x x x ? 0 0 0 | | (0215$_{16}$)··· | 00$_{16}$ |
| SI/O4 interrupt control register | (0048$_{16}$)··· | x x 0 0 ? 0 0 0 | Address control register for expansion register | (0216$_{16}$)··· | 00$_{16}$ |
| SI/O3 interrupt control register | (0049$_{16}$)··· | x x 0 0 ? 0 0 0 | | (0217$_{16}$)··· | 00$_{16}$ |
| Bus collision detection interrupt control register | (004A$_{16}$)··· | x x x x ? 0 0 0 | Data control register for expansion register | (0218$_{16}$)··· | 00$_{16}$ |
| DMA0 interrupt control register | (004B$_{16}$)··· | x x x x ? 0 0 0 | | (0219$_{16}$)··· | 00$_{16}$ |
| DMA1 interrupt control register | (004C$_{16}$)··· | x x x x ? 0 0 0 | Humming 8/4 | (021A$_{16}$)··· | 00$_{16}$ |
| Key input interrupt control register | (004D$_{16}$)··· | x x x x ? 0 0 0 | | (021B$_{16}$)··· | 00$_{16}$ |
| A-D conversion interrupt control register | (004E$_{16}$)··· | x x x x ? 0 0 0 | Humming 24/18 | (021C$_{16}$)··· | 00$_{16}$ |
| UART2 transmit interrupt control register | (004F$_{16}$)··· | x x x x ? 0 0 0 | | (021D$_{16}$)··· | 00$_{16}$ |
| UART2 receive interrupt control register | (0050$_{16}$)··· | x x x x ? 0 0 0 | | (021E$_{16}$)··· | 00$_{16}$ |
| UART0 transmit interrupt control register | (0051$_{16}$)··· | x x x x ? 0 0 0 | | (021F$_{16}$)··· | 00$_{16}$ |
| UART0 receive interrupt control register | (0052$_{16}$)··· | x x x x ? 0 0 0 | Timer B3,4,5 count start flag | (0340$_{16}$)··· | 0 0 0 x x x x x |
| UART1 transmit interrupt control register | (0053$_{16}$)··· | x x x x ? 0 0 0 | Reserved register | (0348$_{16}$)··· | 00$_{16}$ |
| UART1 receive interrupt control register | (0054$_{16}$)··· | x x x x ? 0 0 0 | Reserved register | (0349$_{16}$)··· | 00$_{16}$ |
| Timer A0 interrupt control register | (0055$_{16}$)··· | x x x x ? 0 0 0 | Reserved register | (034A$_{16}$)··· | 00$_{16}$ |
| Timer A1 interrupt control register | (0056$_{16}$)··· | x x x x ? 0 0 0 | Reserved register | (034B$_{16}$)··· | 00$_{16}$ |
| Timer A2 interrupt control register | (0057$_{16}$)··· | x x x x ? 0 0 0 | Timer B3 mode register | (035B$_{16}$)··· | 0 0 ? x 0 0 0 0 |
| Timer A3 interrupt control register | (0058$_{16}$)··· | x x x x ? 0 0 0 | Timer B4 mode register | (035C$_{16}$)··· | 0 0 ? x 0 0 0 0 |
| Timer A4 interrupt control register | (0059$_{16}$)··· | x x x x ? 0 0 0 | Timer B5 mode register | (035D$_{16}$)··· | 0 0 ? x 0 0 0 0 |
| Timer B0 interrupt control register | (005A$_{16}$)··· | x x x x ? 0 0 0 | Interrupt cause select register | (035F$_{16}$)··· | 00$_{16}$ |
| Timer B1 interrupt control register | (005B$_{16}$)··· | x x x x ? 0 0 0 | SI/O3 control register | (0362$_{16}$)··· | 40$_{16}$ |
| Timer B2 interrupt control register | (005C$_{16}$)··· | x x x x ? 0 0 0 | SI/O4 control register | (0366$_{16}$)··· | 40$_{16}$ |
| INT0 interrupt control register | (005D$_{16}$)··· | x x 0 0 ? 0 0 0 | UART2 special mode register 2 | (0376$_{16}$)··· | 00$_{16}$ |
| INT1 interrupt control register | (005E$_{16}$)··· | x x 0 0 ? 0 0 0 | UART2 special mode register | (0377$_{16}$)··· | 00$_{16}$ |
| INT2 interrupt control register | (005F$_{16}$)··· | x x 0 0 ? 0 0 0 | UART2 transmit/receive mode register | (0378$_{16}$)··· | 00$_{16}$ |
| | | | UART2 transmit/receive control register 0 | (037C$_{16}$)··· | 0 0 0 0 1 0 0 0 |
| | | | UART2 transmit/receive control register 1 | (037D$_{16}$)··· | 0 0 0 0 0 0 1 0 |

x : Nothing is mapped to this bit
? : Undefined

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

**Figure 2.3.3 Device's internal status after a reset is cleared**

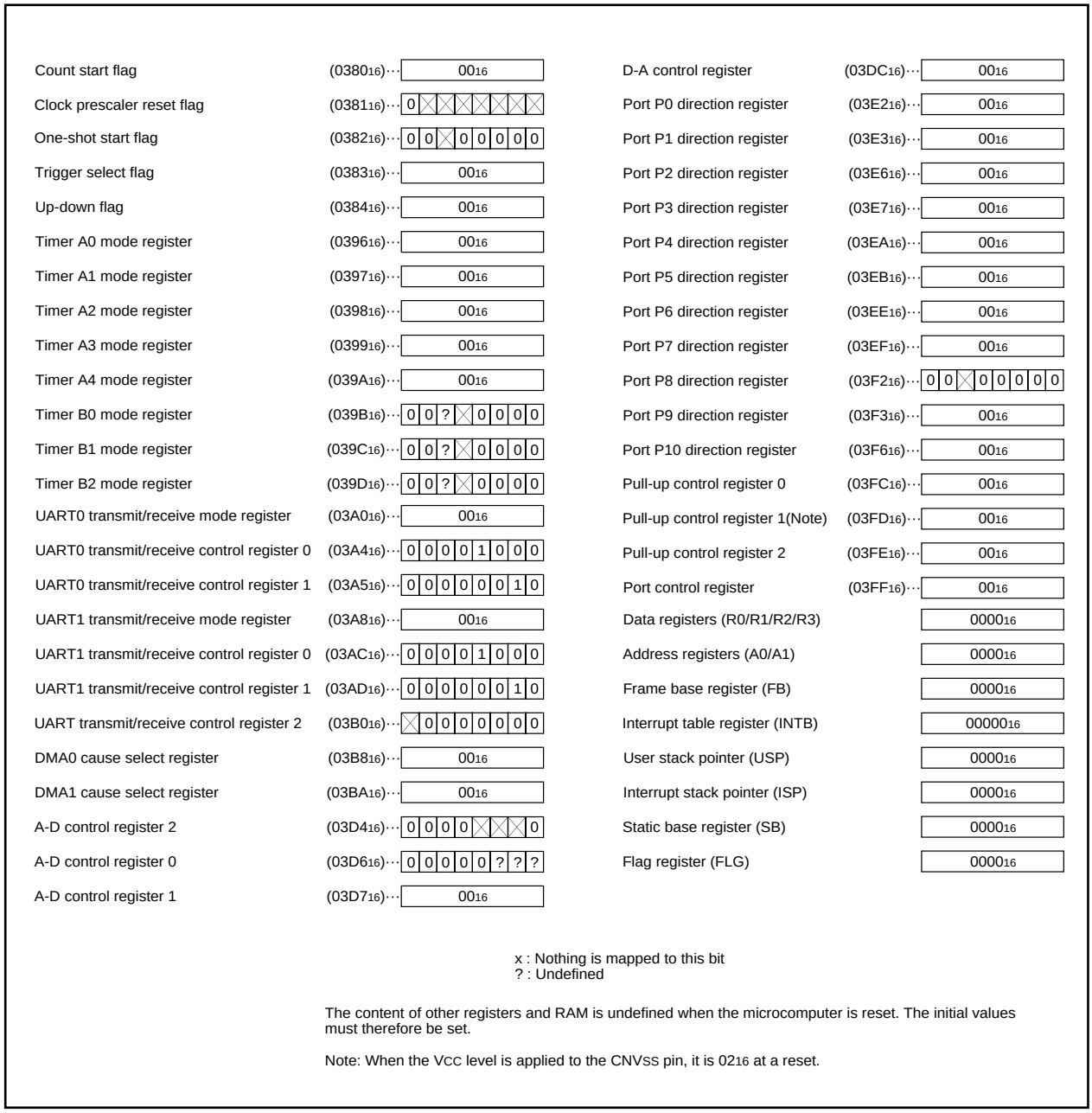



**Figure 2.3.4 Device's internal status after a reset is cleared**

## 2.4 Processor Mode

### (1) Types of Processor Mode

Processor mode can be used at microprocessor mode.

- **Microprocessor mode**

  In microprocessor mode, the SFR, internal RAM, and external memory space can 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 " 2.4.1 Bus Settings" for details.)

### (2) Setting Microprocessor Mode

Microprocessor mode is set using the CNVSS pin and the processor mode bits (bits 1 and 0 at address $0004_{16}$). Set the processor mode bits to "$11_2$".

Regardless of the level of the CNVSS pin, the processor mode bits can be changed by software. Therefore, never change the processor mode bits when changing the contents of other bits.

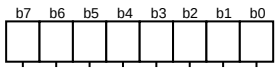
- **Applying Vcc to CNVss pin**

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

Figure 2.4.1 shows the processor mode register 0 and 1.

Figure 2.4.2 shows the memory maps applicable for microprocessor mode.

Processor mode register 0 (Note 1)

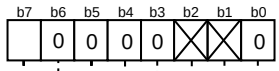



| Symbol | Address | When reset |
|---|---|---|
| PM0 | $0004_{16}$ | $00_{16}$ (Note 2) |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PM00 | Processor mode bit | b1 b0<br>0 0: Inhibited<br>0 1: Inhibited<br>1 0: Inhibited<br>1 1: Microprocessor mode | ○ | ○ |
| PM01 | | | ○ | ○ |
| PM02 | R/W mode select bit | 0 : $\overline{RD}$,$\overline{BHE}$,$\overline{WR}$<br>1 : $\overline{RD}$,$\overline{WRH}$,$\overline{WRL}$ | ○ | ○ |
| PM03 | Software reset bit | The device is reset when this bit is set to “1”. The value of this bit is “0” when read. | ○ | ○ |
| PM04 | Multiplexed bus space select bit | b5 b4<br>0 0 : Multiplexed bus is not used<br>0 1 : Allocated to CS2 space<br>1 0 : Inhibited<br>1 1 : Inhibited | ○ | ○ |
| PM05 | | | ○ | ○ |
| PM06 | Port $P4_0$ to $P4_3$ function select bit | 0 : Address output<br>1 : Port function<br>(Address is not output) | ○ | ○ |
| PM07 | BCLK output disable bit | 0 : BCLK is output<br>1 : BCLK is not output<br>(Pin is left floating) | ○ | ○ |

Notes 1: Set bit 1 of the protect register (address $000A_{16}$) to “1” when writing new values to this register.
2: If the $V_{CC}$ voltage is applied to the $CNV_{SS}$, the value of this register when reset is $03_{16}$. (PM00 and PM01 both are set to “1”.)

Processor mode register 1 (Note)




| Symbol | Address | When reset |
|---|---|---|
| PM1 | $0005_{16}$ | $00000XX0_2$ |

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

Note : Set bit 1 of the protect register (address $000A_{16}$) to “1” when writing new values to this register.

**Figure 2.4.1 Processor mode registers**

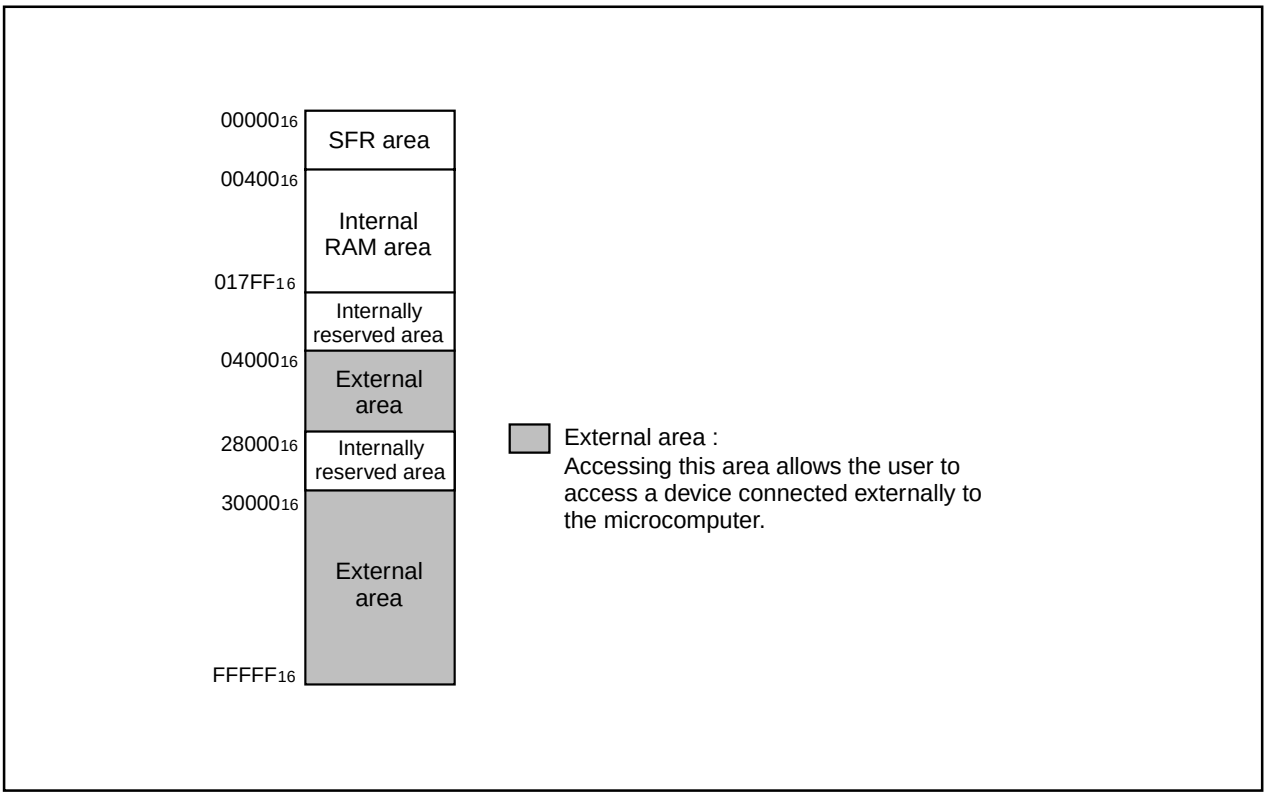



**Figure 2.4.2 Memory maps applicable for microprosessor mode**

### 2.4.1 Bus settings

The BYTE pin and bits 4 to 6 of the processor mode register 0 (address $0004_{16}$) are used to change the bus settings.Table 2.4.1 shows the factors used to change the bus settings.

**Table 2.4.1 Factors for switching bus settings**

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

**(1) Selecting external address bus width**

The address bus width for external output in the 1M bytes of address space can be set to 16 bits (64K bytes address space) or 20 bits (1M bytes address space). When bit 6 of the processor mode register 0 is set to “1”, the external address bus width is set to 16 bits, and P2 and P3 become part of the address bus. $P4_0$ to $P4_3$ can be used as programmable I/O ports. When bit 6 of processor mode register 0 is set to “0”, the external address bus width is set to 20 bits, and P2, P3, and $P4_0$ to $P4_3$ become part of the address bus.

**(2) Selecting external data bus width**

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

**(3) Selecting separate/multiplex bus**

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

**• Separate bus**

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

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

**• Multiplex bus**

In this mode, data and address I/O are time multiplexed. With an 8-bit data bus selected (BYTE pin = “H”), the 8 bits from $D_0$ to $D_7$ are multiplexed with $A_0$ to $A_7$.

With a 16-bit data bus selected (BYTE pin = “L”), the 8 bits from $D_0$ to $D_7$ are multiplexed with $A_1$ to $A_8$. $D_8$ to $D_{15}$ are not multiplexed. In this case, the external devices connected to the multiplexed bus are mapped to the microcomputer’s even addresses (every 2nd address). To access these external devices, access the even addresses as bytes.

The ALE signal latches the address. It is output from $P5_6$.

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

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

### Table 2.4.2 Pin functions for processor mode

| Processor mode | Microprocessor modes | | | |
|---|---|---|---|---|
| Multiplexed bus space select bit | "01" [CS2 is for multiplexed bus and others are for separate bus] | | "00" (separate bus) | |
| Data bus width BYTE pin level | 8 bits "H" | 16 bits "L" | 8 bits "H" | 16 bits "L" |
| $P0_0$ to $P0_7$ | Data bus | Data bus | Data bus | Data bus |
| $P1_0$ to $P1_7$ | I/O port | Data bus | I/O port | Data bus |
| $P2_0$ | Address bus /data bus (Note) | Address bus | Address bus | Address bus |
| $P2_1$ to $P2_7$ | Address bus data bus (Note) | Address bus data bus (Note) | Address bus | Address bus |
| $P3_0$ | Address bus | Address bus data bus (Note) | Address bus | Address bus |
| $P3_1$ to $P3_7$ | Address bus | Address bus | Address bus | Address bus |
| $P4_0$ to $P4_3$ Port $P4_0$ to $P4_3$ function select bit = 1 | I/O port | I/O port | I/O port | I/O port |
| $P4_0$ to $P4_3$ Port $P4_0$ to $P4_3$ function select bit = 0 | Address bus | Address bus | Address bus | Address bus |
| $P4_4$ to $P4_7$ | $\overline{CS}$ (chip select) or programmable I/O port (For details, refer to "Bus control") | | | |
| $P5_0$ to $P5_3$ | Outputs $\overline{RD}$, $\overline{WRL}$, $\overline{WRH}$, and BCLK or $\overline{RD}$, $\overline{BHE}$, $\overline{WR}$, and BCLK (For details, refer to "Bus control") | | | |
| $P5_4$ | $\overline{HLDA}$ | $\overline{HLDA}$ | $\overline{HLDA}$ | $\overline{HLDA}$ |
| $P5_5$ | $\overline{HOLD}$ | $\overline{HOLD}$ | $\overline{HOLD}$ | $\overline{HOLD}$ |
| $P5_6$ | ALE | ALE | ALE | ALE |
| $P5_7$ | $\overline{RDY}$ | $\overline{RDY}$ | $\overline{RDY}$ | $\overline{RDY}$ |

Note : Address bus when in separate bus mode.

### 2.4.2 Bus Control

The following explains the signals required for accessing external devices and software waits.

**(1) Address bus/data bus**

The address bus consists of the 20 pins $A_0$ to $A_{19}$ for accessing the 1M bytes of address space.

The data bus consists of the pins for data I/O. When the BYTE pin is "H", the 8 ports $D_0$ to $D_7$ function as the data bus. When BYTE is "L", the 16 ports $D_0$ to $D_{15}$ function as the data bus.

**(2) Chip select signal**

The chip select signal is output using the same pins as $P4_4$, $P4_6$ and to $P4_7$. Bits 0, 2 and 3 of the chip select control register (address $0008_{16}$) set each pin to function as a port or to output the chip select signal.

In microprocessor mode, only $\overline{CS0}$ outputs the chip select signal after the reset state has been cancelled. $\overline{CS2}$,$\overline{CS3}$ function as input ports. Figure 2.4.3 shows the chip select control register.

The chip select signal can be used to split the external area. Tables 2.4.3 show the external memory areas specified using the chip select signal.

**Table 2.4.3 External areas specified by the chip select signals**

| Processor mode | Chip select signal | | |
|---|---|---|---|
| | $\overline{CS0}$ | $\overline{CS2}$ | $\overline{CS3}$ |
| Microprocessor mode | $30000_{16}$ to FFFFF$_{16}$ (832K bytes) | $08000_{16}$ to 27FFF$_{16}$ (128K bytes) | $04000_{16}$ to 07FFF$_{16}$ (16K bytes) |

Note : Address $28000_{16}$ to 2FFFF$_{16}$ are reserved and cannot be used.

Chip select control register

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

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

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CS0 | $\overline{CS0}$ output enable bit | 0 : Chip select output disabled (Normal port pin)<br>1 : Chip select output enabled | ○ | ○ |
| Reserved bit | | Always set to "0". | ○ | ○ |
| CS2 | $\overline{CS2}$ output enable bit | 0 : Chip select output disabled (Normal port pin)<br>1 : Chip select output enabled | ○ | ○ |
| CS3 | $\overline{CS3}$ output enable bit | | ○ | ○ |
| CS0W | $\overline{CS0}$ wait bit | 0 : Wait state inserted<br>1 : No wait state | ○ | ○ |
| Reserved bit | | Always set to "0". | ○ | ○ |
| CS2W | $\overline{CS2}$ wait bit | 0 : Wait state inserted<br>1 : No wait state | ○ | ○ |
| CS3W | CS3 wait bit | | ○ | ○ |

**Figure 2.4.3 Chip select control register**

### (3) Read/write signals

With a 16-bit data bus (BYTE pin ="L"), bit 2 of the processor mode register 0 (address $0004_{16}$) select the combinations of $\overline{RD}$, $\overline{BHE}$, and $\overline{WR}$ signals or $\overline{RD}$, $\overline{WRL}$, and $\overline{WRH}$ signals. With an 8-bit data bus (BYTE pin = "H"), use the combination of $\overline{RD}$, $\overline{WR}$, and $\overline{BHE}$ signals. (Set bit 2 of the processor mode register 0 (address $0004_{16}$) to "0".) Tables 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 $0004_{16}$) has been set (Note).

Note: Before attempting to change the contents of the processor mode register 0, set bit 1 of the protect register (address $000A_{16}$) to "1".

**Table 2.4.4 Operation of $\overline{RD}$, $\overline{WRL}$, and $\overline{WRH}$ signals**

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

**Table 2.4.5 Operation of $\overline{RD}$, $\overline{WR}$, and $\overline{BHE}$ signals**

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

### (4) ALE signal

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

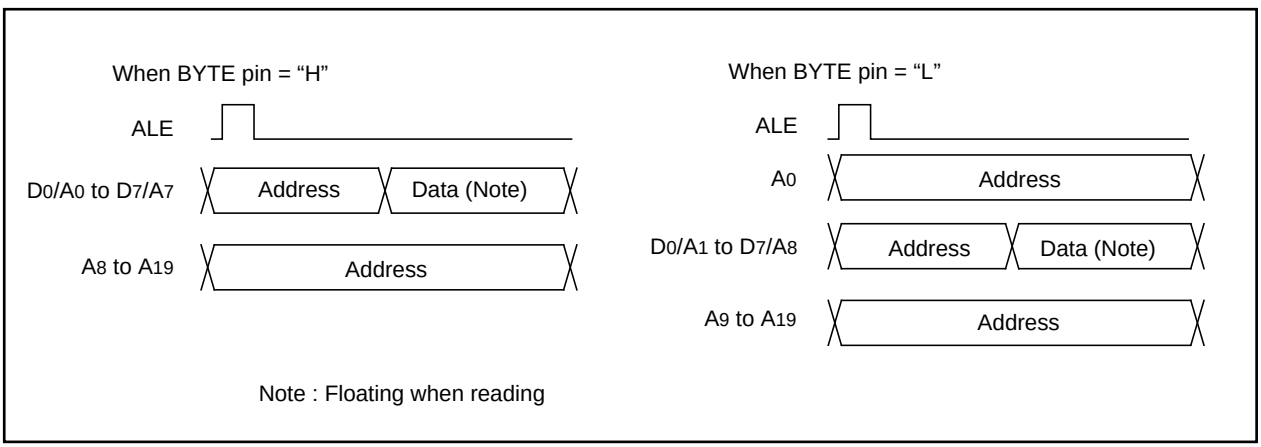



**Figure 2.4.4 ALE signal and address/data bus**

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

$\overline{RDY}$ is a signal that facilitates access to an external device that requires long access time. As shown in Figure 2.4.5, if an “L” is being input to the $\overline{RDY}$ at the BCLK falling edge, the bus turns to the wait state. If an “H” is being input to the $\overline{RDY}$ pin at the BCLK falling edge, the bus cancels the wait state. Table 2.4.6 shows the state of the microcomputer with the bus in the wait state, and Figure 2.4.5 shows an example in which the $\overline{RD}$ signal is prolonged by the $\overline{RDY}$ signal.

The $\overline{RDY}$ signal is valid when accessing the external area during the bus cycle in which bits 4, 6 and 7 of the chip select control register (address $0008_{16}$) are set to “0”. The $\overline{RDY}$ signal is invalid when setting “1” to all bits 4, 6 and 7 of the chip select control register (address $0008_{16}$), 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/$\overline{W}$ signal, address bus, data bus, $\overline{CS}$<br>ALE signal, $\overline{HLDA}$, programmable I/O ports | Maintain status when $\overline{RDY}$ signal received |
| Internal peripheral circuits | On |

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

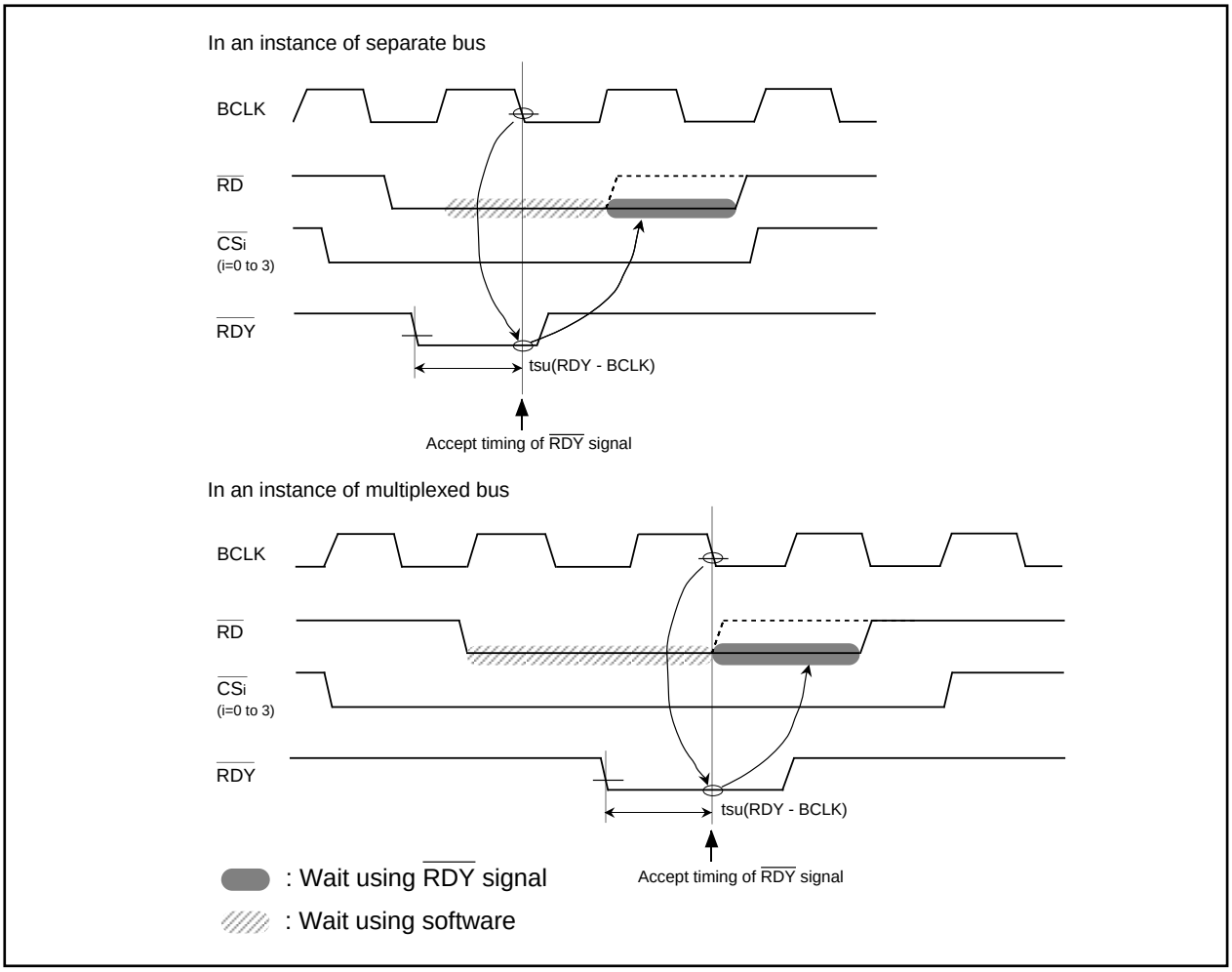



**Figure 2.4.5 Example of $\overline{RD}$ signal extended by $\overline{RDY}$ signal**

**(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.

$\overline{\text{HOLD}}$ > DMAC > 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 the 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 RAM accessed |
|---|---|---|---|
| Address bus | | Address output | Maintain status before accessed address of external area |
| Data bus | When read | Floating | Floating |
| | When write | Output data | Undefined |
| $\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{WRL}}$, $\overline{\text{WRH}}$ | | $\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{WRL}}$, $\overline{\text{WRH}}$ output | Output "H" |
| $\overline{\text{BHE}}$ | | $\overline{\text{BHE}}$ output | Maintain status before accessed status of external area |
| $\overline{\text{CS}}$ | | Output "H" | Output "H" |
| ALE | | Output "L" | Output "L" |

**(8) BCLK output**

The user can choose the BCLK output by use of bit 7 of processor mode register 0 ($0004_{16}$) (Note). When set to "1", the output floating.

Note: Before attempting to change the contents of the processor mode register 0, set bit 1 of the protectregister (address $000A_{16}$) to "1".

**(9) Software wait**

A software wait can be inserted by setting the wait bit (bit 7) of the processor mode register 1 (address $0005_{16}$) (Note) and bits 4, 6 and 7 of the chip select control register (address $0008_{16}$).

A software wait is inserted in the internal RAM area and in the external memory area by setting the wait bit of the processor mode register 1. When set to "0", each bus cycle is executed in one BCLK cycle. When set to "1", each bus cycle is executed in two or three BCLK cycles. After the microcomputer has been reset, this bit defaults to "0". When set to "1", a wait is applied to all memory areas (two or three BCLK cycles), regardless of the contents of bits 4, 6 and 7 of the chip select control register. Set this bit after referring to the recommended operating conditions (main clock input oscillation frequency) of the electric characteristics. However, when the user is using the $\overline{RDY}$ signal, the relevant bit in the chip select control register's bits 4, 6 and 7 must be set to "0".

When the wait bit of the processor mode register 1 is "0", software waits can be set independently for each areas selected using the chip select signal. Bits 4, 6 and 7 of the chip select control register correspond to chip selects $\overline{CS0}$, $\overline{CS2}$, and $\overline{CS3}$. When one of these bits is set to "1", the bus cycle is executed in one BCLK cycle. When set to "0", the bus cycle is executed in two or three BCLK cycles. These bits default to "0" after the microcomputer has been reset.

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

Table 2.4.9 shows the software wait and bus cycles. Figure 2.4.7 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 $000A_{16}$) to "1".

**Table 2.4.9 Software waits and bus cycles**

| Area | Bus status | Wait bit | Bits 4, 6 and 7 of chip select control register | Bus cycle |
|---|---|---|---|---|
| SFR | ——— | Invalid | Invalid | 2 BCLK cycles |
| Internal RAM | ——— | 0 | Invalid | 1 BCLK cycle |
| | ——— | 1 | Invalid | 2 BCLK cycles |
| External memory area | Separate bus | 0 | 1 | 1 BCLK cycle |
| | Separate bus | 0 | 0 | 2 BCLK cycles |
| | Separate bus | 1 | 0 (Note) | 2 BCLK cycles |
| | Multiplex bus | 0 | 0 | 3 BCLK cycles |
| | Multiplex bus | 1 | 0 (Note) | 3 BCLK cycles |

Note: When using the RDY signal, always set to "0".

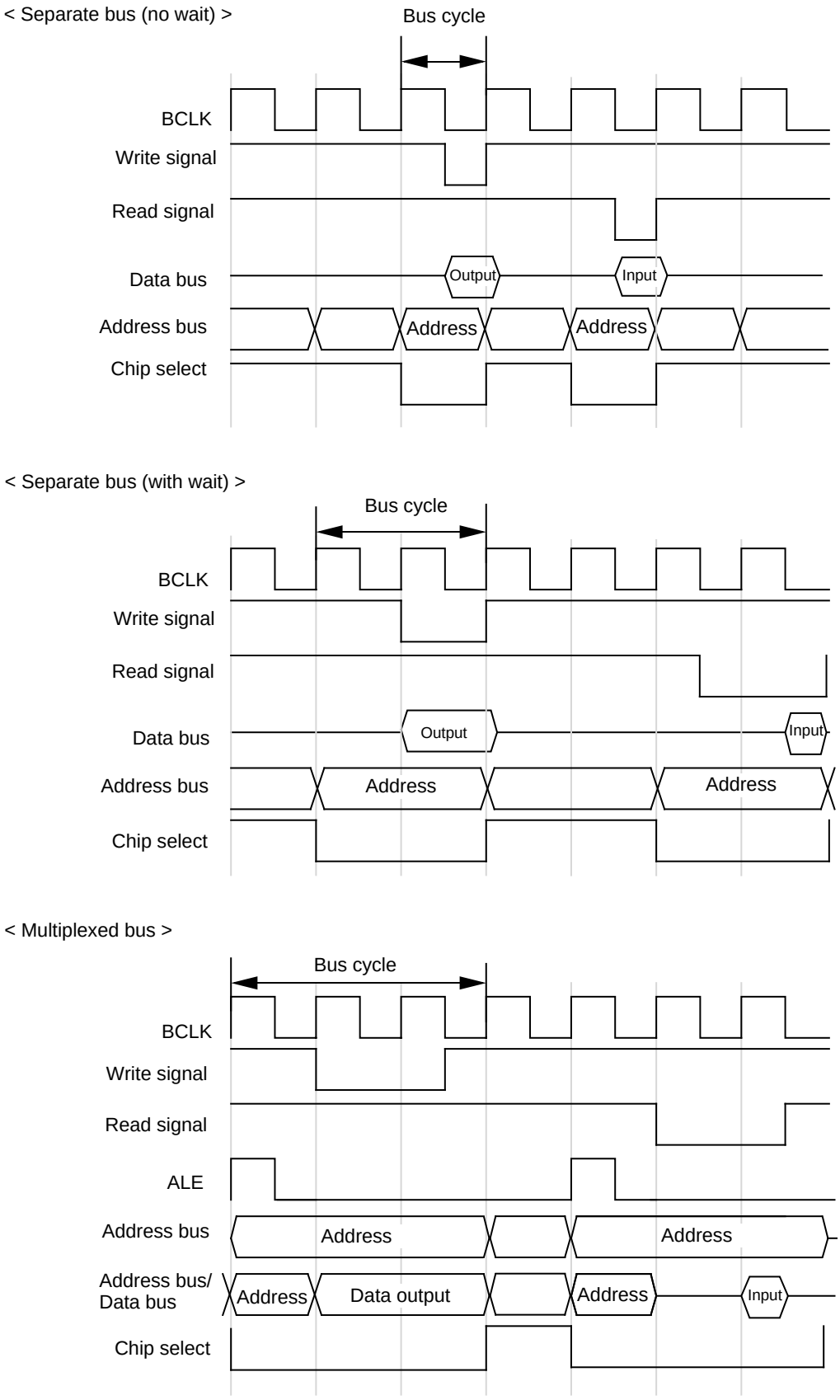



**Figure 2.4.7 Typical bus timings using software wait**

## 2.5 Clock Generating Circuit

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

**Table 2.5.1 Main clock and sub clock generating circuits**

| | Main clock generating circuit | Sub clock generating circuit |
|---|---|---|
| Use of clock | • CPU's operating clock source<br>• Internal peripheral units' operating clock source | • CPU's operating clock source<br>• Timer A/B's count clock source |
| Usable oscillator | Ceramic or crystal oscillator | Crystal oscillator |
| Pins to connect oscillator | XIN, XOUT | XCIN, XCOUT |
| Oscillation stop/restart function | Available | Available |
| Oscillator status immediately after reset | Oscillating | Stopped |
| Other | Externally derived clock can be input | |

### 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.

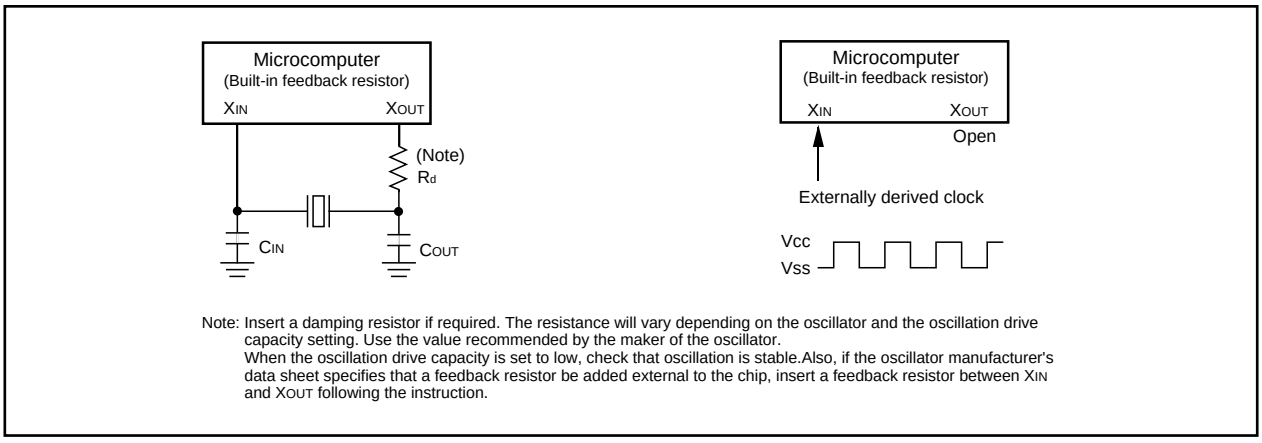



**Figure 2.5.1 Examples of main clock**

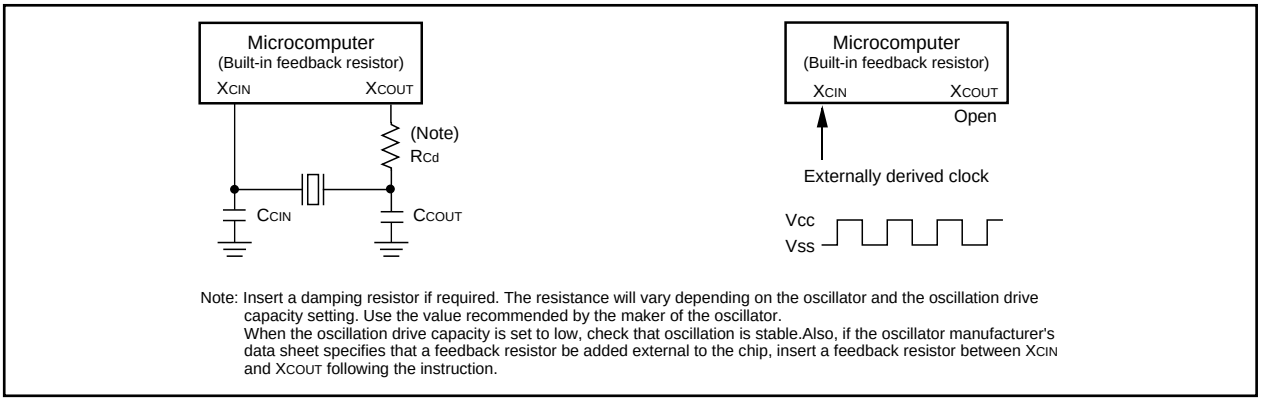



**Figure 2.5.2 Examples of sub clock**

### 2.5.2 Clock Control

Figure 2.5.3 shows the block diagram of the clock generating circuit.

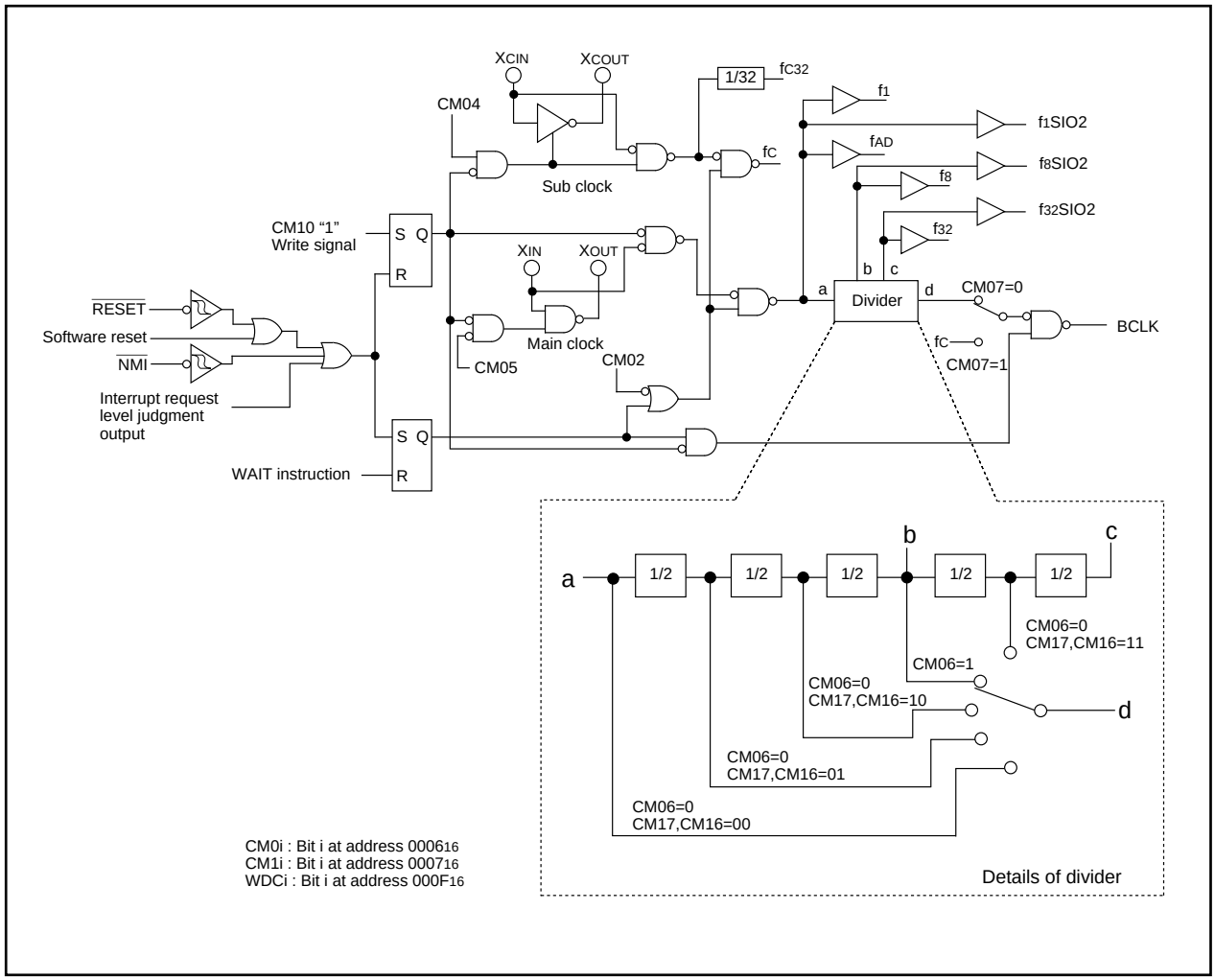



**Figure 2.5.3 Clock generating circuit**

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

**(1) Main clock**

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

**(2) Sub-clock**

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

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

**(3) BCLK**

The BCLK is the clock that drives the CPU, and is fc or the clock is derived by dividing the main clock by 1, 2, 4, 8, or 16. The BCLK is derived by dividing the main clock by 8 after a reset. The BCLK signal can be output from BCLK pin by the BCLK output disable bit (bit 7 at address $0004_{16}$).

The main clock division select bit 0(bit 6 at address $0006_{16}$) 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($f_1$, $f_8$, $f_{32}$, $f_{1SIO2}$, $f_{8SIO2}$,$f_{32SIO2}$,$f_{AD}$)**

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

**(5) $f_{C32}$**

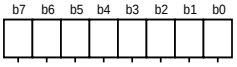
This clock is derived by dividing the sub-clock by 32. It is used for the timer A and timer B counts.

**(6) $f_C$**

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

Figure 2.5.4 shows the system clock control registers 0 and 1.

System clock control register 0 (Note 1)

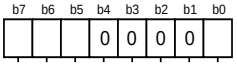



Symbol: CM0 | Address: 0006₁₆ | When reset: 48₁₆

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CM00 | Clock output function select bit | b1 b0<br>0 0 : I/O port P5₇<br>0 1 : Inhibited<br>1 0 : Inhibited<br>1 1 : Inhibited | O | O |
| CM01 | | | O | O |
| CM02 | WAIT peripheral function clock stop bit | 0 : Do not stop peripheral function clock in wait mode<br>1 : Stop peripheral function clock in wait mode (Note 8) | O | O |
| CM03 | XCIN-XCOUT drive capacity select bit (Note 2) | 0 : LOW<br>1 : HIGH | O | O |
| CM04 | Port XC select bit | 0 : I/O port<br>1 : XCIN-XCOUT generation | O | O |
| CM05 | Main clock (XIN-XOUT) stop bit (Notes 3,4,5) | 0 : On<br>1 : Off | O | O |
| CM06 | Main clock division select bit 0 (Note 7) | 0 : CM16 and CM17 valid<br>1 : Division by 8 mode | O | O |
| CM07 | System clock select bit (Note 6) | 0 : XIN, XOUT<br>1 : XCIN, XCOUT | O | O |

Note 1: Set bit 0 of the protect register (address 000A16) to "1" before writing to this register.
Note 2: Changes to "1" when shifting to stop mode and at a reset.
Note 3: When entering power saving mode, main clock stops using this bit. When returning from stop mode and operating with XIN, set this bit to "0". When main clock oscillation is operating by itself, set system clock select bit (CM07) to "1" before setting this bit to "1".
Note 4: When inputting external clock, only clock oscillation buffer is stopped and clock input is acceptable.
Note 5: If this bit is set to "1", XOUT turns "H". The built-in feedback resistor remains being connected, so XIN turns pulled up to XOUT ("H") via the feedback resistor.
Note 6: Set port Xc select bit (CM04) to "1" and stabilize the sub-clock oscillating before setting to this bit from "0" to "1". Do not write to both bits at the same time. And also, set the main clock stop bit (CM05) to "0" and stabilize the main clock oscillating before setting this bit from "1" to "0".
Note 7: 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.
Note 8: fC32 is not included.

System clock control register 1 (Note 1)




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

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

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

**Figure 2.5.4 Clock control registers 0 and 1**

### 2.5.3 Stop Mode

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

The internal oscillator circuit of expansion function (OSD function/ data slice function/ data encode function/ humming function) stops oscillation when expansion register CK_VCO, XTAL_VCO, PDC_VCO_ON, VPS_VCO_ON = "L".

Because the oscillation , BCLK, $f_1$ to $f_{32}$, $f_{1SIO2}$ to $f_{32SIO2}$, $f_C$, $f_{C32}$, and $f_{AD}$ stops in stop mode, peripheral functions such as the A-D converter and watchdog timer do not function. However, timer A and timer B operate provided that the event counter mode is set to an external pulse, and UARTi(i = 0 to 2) SI/O3,4 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 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 $0006_{16}$) 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 | Microprocessor mode |
|---|---|
| Address bus, data bus, $\overline{CS0}$, $\overline{CS2}$, $\overline{CS3}$ | Retains status before stop mode |
| $\overline{RD}$, $\overline{WR}$, $\overline{BHE}$, $\overline{WRL}$, $\overline{WRH}$ | “H” |
| $\overline{HLDA}$, BCLK | “H” |
| ALE | “H” |
| Port | Retains status before stop mode |

### 2.5.4 Wait Mode

When a WAIT instruction is executed, the BCLK stops and the microcomputer enters the wait mode. In this mode, oscillation continues but the BCLK and watchdog timer stop. Writing “1” to the WAIT peripheral function clock stop bit and executing a WAIT instruction stops the clock being supplied to the internal peripheral functions, allowing power dissipation to be reduced. Table 2.5.3 shows the status of the ports in wait mode.

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

**Table 2.5.3 Port status during wait mode**

| Pin | Microprocessor mode |
|---|---|
| Address bus, data bus, $\overline{CS0}$, $\overline{CS2}$, $\overline{CS3}$ | Retains status before stop mode |
| $\overline{RD}$, $\overline{WR}$, $\overline{BHE}$, $\overline{WRL}$, $\overline{WRH}$ | “H” |
| $\overline{HLDA}$, BCLK | “H” |
| ALE | “H” |
| Port | Retains status before stop mode |

### 2.5.5 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.

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

**(1) Division by 2 mode**

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

**(2) Division by 4 mode**

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

**(3) Division by 8 mode**

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

**(4) Division by 16 mode**

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

**(5) No-division mode**

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

**(6) Low-speed mode**

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

**(7) Low power dissipation mode**

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

Note :

Before the count source for BCLK can be changed from $X_{IN}$ to $X_{CIN}$ or vice versa, the clock to which the count source is going to be switched must be oscillating stably. Allow a wait time in software for the oscillation to stabilize before switching over the clock.

**Table 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 |

### 2.5.6 Power control

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

**Modes**

Power control is available in three modes.

**(a) Normal operation mode**

**• High-speed mode**

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

**• Medium-speed mode**

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

**• Low-speed mode**

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

**• Low power consumption mode**

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

**(b) Wait mode**

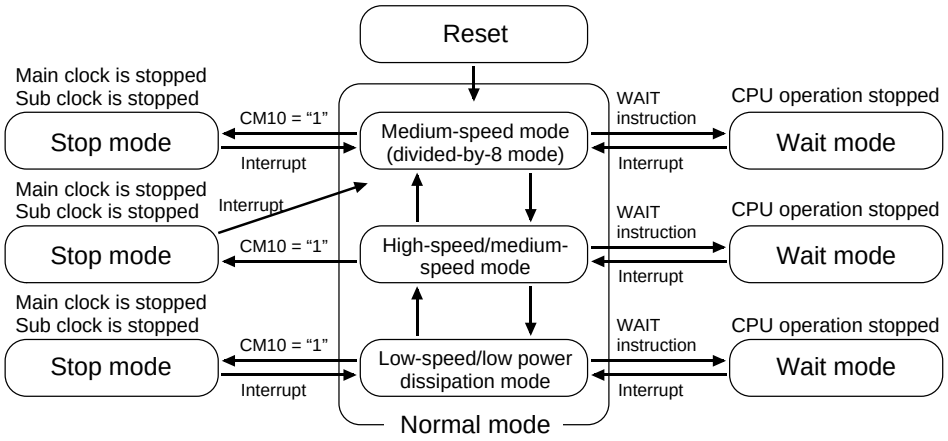
The CPU operation is stopped. The oscillators do not stop.

**(c) Stop mode**

The main clock and the sub-clock 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.

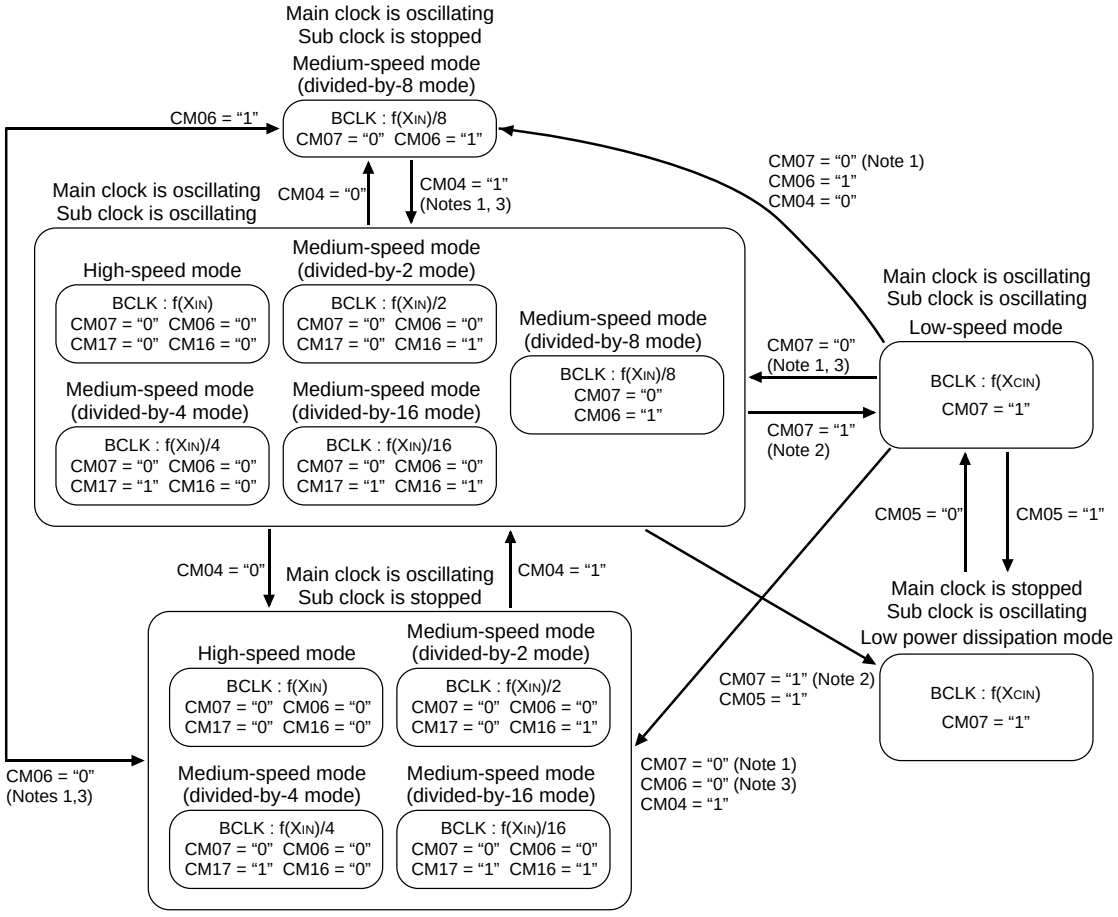
Figure 2.5.5 is the state transition diagram of the above modes.

**Transition of stop mode, wait mode**




(Refer to the following for the transition of normal mode.)

**Transition of normal mode**




Note 1: Switch clock after oscillation of main clock is sufficiently stable.
Note 2: Switch clock after oscillation of sub clock is sufficiently stable.
Note 3: Change CM06 after changing CM17 and CM16.
Note 4: Transit in accordance with arrow.

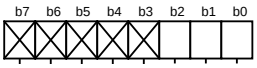
**Figure 2.5.5 State transition diagram of Power control mode**

## 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 $0004_{16}$), processor mode register 1 (address $0005_{16}$), system clock control register 0 (address $0006_{16}$), system clock control register 1 (address $0007_{16}$), port P9 direction register (address $03F3_{16}$) , SI/O3 control register (address $0362_{16}$) and SI/O4 control register (address $0366_{16}$) can only be changed when the respective bit in the protect register is set to “1”. Therefore, important outputs can be allocated to port P9.

If, after “1” (write-enabled) has been written to the port P9 direction register and SI/Oi control register (i=3,4) write-enable bit (bit 2 at address $000A_{16}$), a value is written to any address, the bit automatically reverts to “0” (write-inhibited). However, the system clock control registers 0 and 1 write-enable bit (bit 0 at $000A_{16}$) and processor mode register 0 and 1 write-enable bit (bit 1 at $000A_{16}$) do not automatically return to “0” after a value has been written to an address. The program must therefore be written to return these bits to “0”.

Protect register




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

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PRC0 | Enables writing to system clock control registers 0 and 1 (addresses $0006_{16}$ and $0007_{16}$) | 0 : Write-inhibited<br>1 : Write-enabled | ○ | ○ |
| PRC1 | Enables writing to processor mode registers 0 and 1 (addresses $0004_{16}$ and $0005_{16}$) | 0 : Write-inhibited<br>1 : Write-enabled | ○ | ○ |
| PRC2 | Enables writing to port P9 direction register (address $03F3_{16}$) and to SI/Oi control register (i=3,4) (addresses $0362_{16}$ and $0366_{16}$)(Note) | 0 : Write-inhibited<br>1 : Write-enabled | ○ | ○ |
| Nothing is assigned.<br>In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminate. | | | – | – |

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

**Figure 2.6.1 Protect register**

## 2.7 Interrupt

### 2.7.1 Interrupt

Figure 2.7.1 lists the types of interrupts.

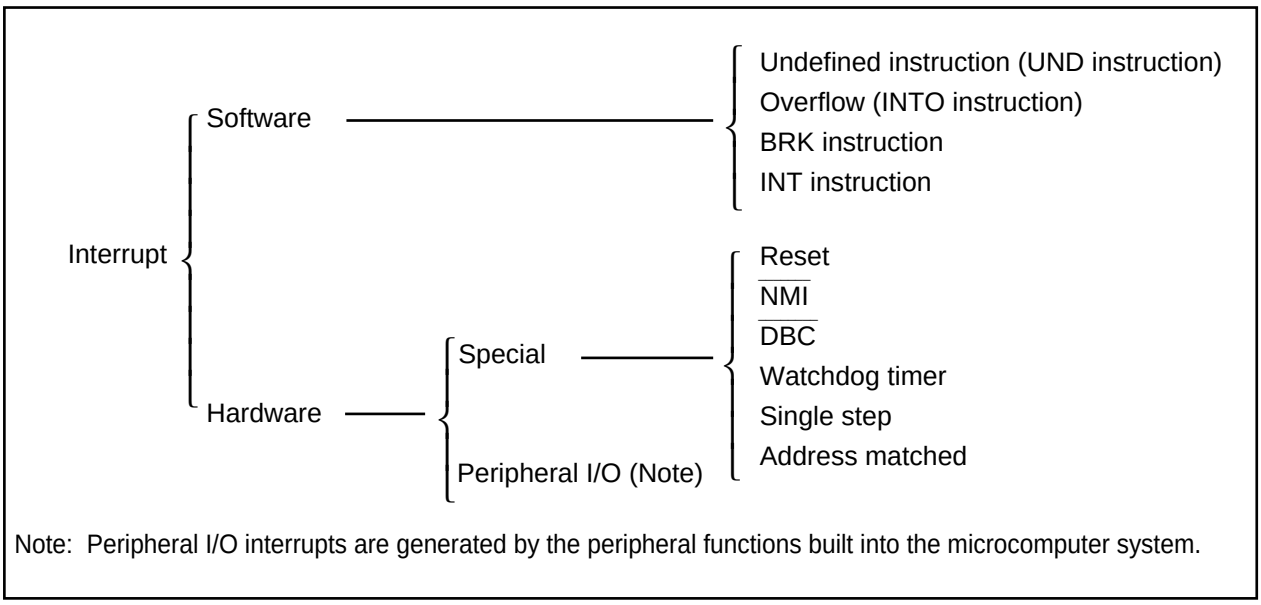



**Figure 2.7.1 Classification of interrupts**

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

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

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

### 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.

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

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

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

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

- **Watchdog timer interrupt**

  Generated by the watchdog timer.

- **Single-step interrupt**

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

- **Address match interrupt**

  An address match interrupt occurs immediately before the instruction held in the address indicated by the address match interrupt register is executed with the address match interrupt enable bit set to “1”.

  If an address other than the first address of the instruction in the address match interrupt register is set, no address match interrupt occurs. For address match interrupt, see 2.7.10 Address match Interrupt.

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

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

- **Bus collision detection interrupt**

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

- **DMA0 interrupt, DMA1 interrupt**

  These are interrupts that DMA generates.

- **Key-input interrupt**

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

- **A-D conversion interrupt**

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

- **UART0, UART1, UART2/NACK, SI/O3 and SI/O4 transmission interrupt**

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

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

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

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

  These are interrupts that timer A generates

- **Timer B0 interrupt through timer B5 interrupt**

  These are interrupts that timer B generates.

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

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

### 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.

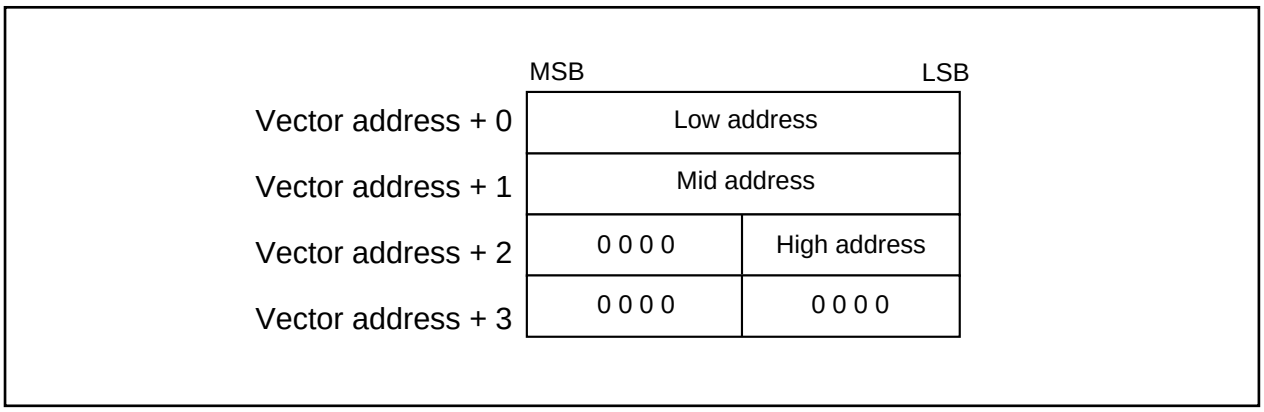



**Figure 2.7.2 Format for specifying interrupt vector addresses**

**• Fixed vector tables**

The fixed vector table is a table in which addresses are fixed. The vector tables are located in an area extending from FFFDC16 to FFFFF16. One vector table comprises four bytes. Set the first address of interrupt routine in each vector table. Table 2.7.1 shows the interrupts assigned to the fixed vector tables and addresses of vector tables.

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

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

Note: Interrupts used for debugging purposes only.

**• Variable vector tables**

The addresses in the variable vector table can be modified, according to the user's settings. Indicate the first address using the interrupt table register (INTB). The 256-byte area subsequent to the address the INTB indicates becomes the area for the variable vector tables. One vector table comprises four bytes. Set the first address of the interrupt routine in each vector table. Table 2.7.2 shows the interrupts assigned to the variable 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<br>Address (L) to address (H) | Interrupt source | Remarks |
|---|---|---|---|
| Software interrupt number 0 | +0 to +3 (Note 1) | BRK instruction | Cannot be masked I flag |
| | | | |
| Software interrupt number 4 | +16 to +19 (Note 1) | $\overline{\text{INT3}}$ | |
| Software interrupt number 5 | +20 to +23 (Note 1) | Timer B5 | |
| Software interrupt number 6 | +24 to +27 (Note 1) | Timer B4 | |
| Software interrupt number 7 | +28 to +31 (Note 1) | Timer B3 | |
| Software interrupt number 8 | +32 to +35 (Note 1) | SI/O4/$\overline{\text{INT5}}$ (Note 2) | |
| Software interrupt number 9 | +36 to +39 (Note 1) | SI/O3/$\overline{\text{INT4}}$ (Note 2) | |
| Software interrupt number 10 | +40 to +43 (Note 1) | Bus collision detection | |
| Software interrupt number 11 | +44 to +47 (Note 1) | DMA0 | |
| Software interrupt number 12 | +48 to +51 (Note 1) | DMA1 | |
| Software interrupt number 13 | +52 to +55 (Note 1) | Key input interrupt | |
| Software interrupt number 14 | +56 to +59 (Note 1) | A-D | |
| Software interrupt number 15 | +60 to +63 (Note 1) | UART2 transmit/NACK (Note 3) | |
| Software interrupt number 16 | +64 to +67 (Note 1) | UART2 receive/ACK (Note 3) | |
| Software interrupt number 17 | +68 to +71 (Note 1) | UART0 transmit | |
| Software interrupt number 18 | +72 to +75 (Note 1) | UART0 receive | |
| Software interrupt number 19 | +76 to +79 (Note 1) | UART1 transmit | |
| Software interrupt number 20 | +80 to +83 (Note 1) | UART1 receive | |
| Software interrupt number 21 | +84 to +87 (Note 1) | Timer A0 | |
| Software interrupt number 22 | +88 to +91 (Note 1) | Timer A1 | |
| Software interrupt number 23 | +92 to +95 (Note 1) | Timer A2 | |
| Software interrupt number 24 | +96 to +99 (Note 1) | Timer A3 | |
| Software interrupt number 25 | +100 to +103 (Note 1) | Timer A4 | |
| Software interrupt number 26 | +104 to +107 (Note 1) | Timer B0 | |
| Software interrupt number 27 | +108 to +111 (Note 1) | Timer B1 | |
| Software interrupt number 28 | +112 to +115 (Note 1) | Timer B2 | |
| Software interrupt number 29 | +116 to +119 (Note 1) | $\overline{\text{INT0}}$ | |
| Software interrupt number 30 | +120 to +123 (Note 1) | $\overline{\text{INT1}}$ | |
| Software interrupt number 31 | +124 to +127 (Note 1) | $\overline{\text{INT2}}$ | |
| Software interrupt number 32<br>to<br>Software interrupt number 63 | +128 to +131 (Note 1)<br>to<br>+252 to +255 (Note 1) | Software interrupt | Cannot be masked I flag |

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

### 2.7.5 Interrupt Control

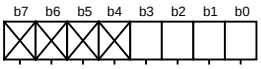
Descriptions are given here regarding how to enable or disable maskable interrupts and how to set the priority to be accepted. What is described here does not apply to non-maskable interrupts.

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

Interrupt control register

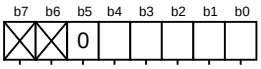
| Symbol | Address | When reset |
|---|---|---|
| TBiIC(i=3 to 5) | $0045_{16}$ to $0047_{16}$ | $XXXXX000_2$ |
| BCNIC | $004A_{16}$ | $XXXXX000_2$ |
| DMiIC(i=0, 1) | $004B_{16}$, $004C_{16}$ | $XXXXX000_2$ |
| KUPIC | $004D_{16}$ | $XXXXX000_2$ |
| ADIC | $004E_{16}$ | $XXXXX000_2$ |
| SiTIC(i=0 to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




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

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

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




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

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

**Figure 2.7.3 Interrupt control registers**

**(1) 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) 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").

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

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

| IPL ($IPL_2$ $IPL_1$ $IPL_0$) | Enabled interrupt priority levels |
|---|---|
| 0 0 0 | Interrupt levels 1 and above are enabled |
| 0 0 1 | Interrupt levels 2 and above are enabled |
| 0 1 0 | Interrupt levels 3 and above are enabled |
| 0 1 1 | Interrupt levels 4 and above are enabled |
| 1 0 0 | Interrupt levels 5 and above are enabled |
| 1 0 1 | Interrupt levels 6 and above are enabled |
| 1 1 0 | Interrupt levels 7 and above are enabled |
| 1 1 1 | All maskable interrupts are disabled |

**(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

### 2.7.6 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:

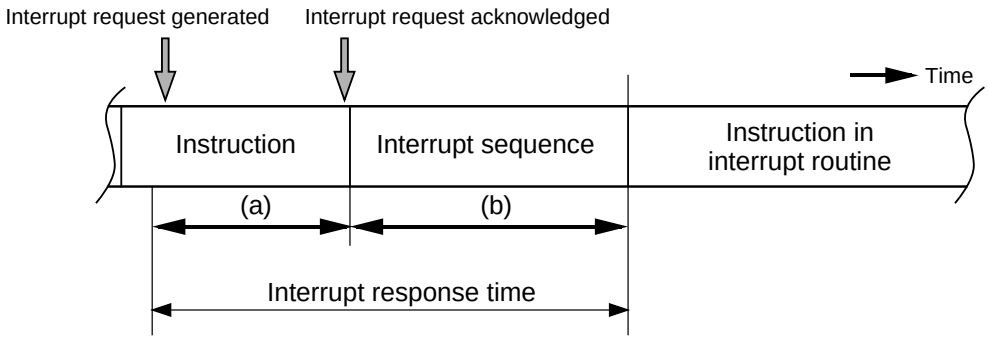
(a) CPU gets the interrupt information (the interrupt number and interrupt request level) by reading address $0000_{16}$.
(b) Saves the content of the flag register (FLG) as it was immediately before the start of interrupt sequence in the temporary register (Note) within the CPU.
(c) Sets the interrupt 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)
(d) Saves the content of the temporary register (Note) within the CPU in the stack area.
(e) Saves the content of the program counter (PC) in the stack area.
( f) 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.

#### (1) 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.




(a) Time from interrupt request is generated to when the instruction then under execution is completed.
(b) Time in which the instruction sequence is executed.

**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 $\overline{\text{DBC}}$ interrupt; add 1 cycle in the case either of an address coincidence interrupt or of a single-step interrupt.
Notes 2: Locate an interrupt vector address in an even address, if possible.

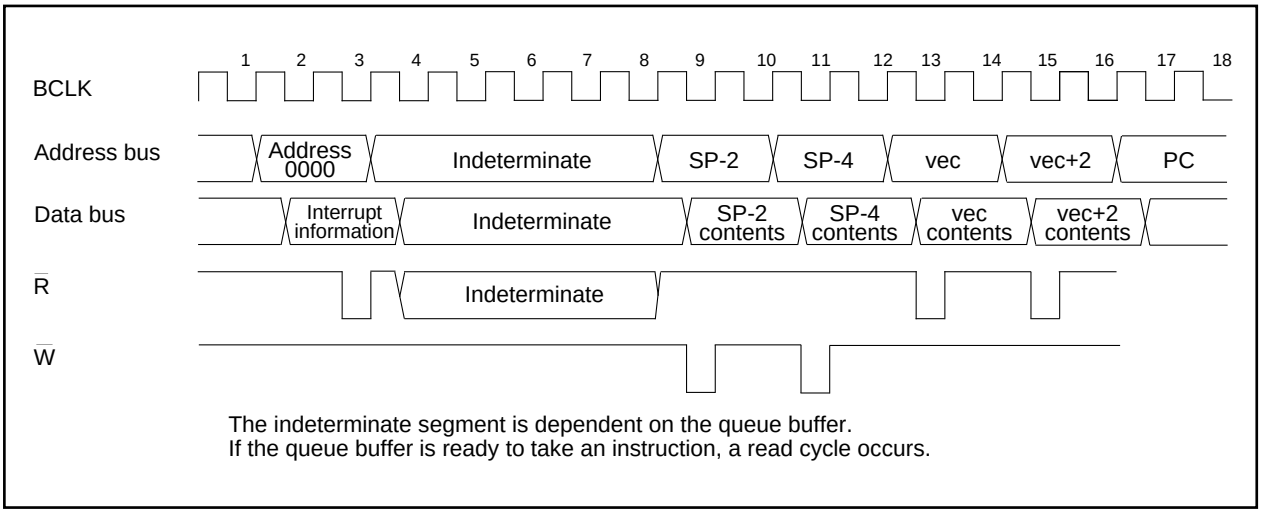



**Figure 2.7.5 Time required for executing the interrupt sequence**

**(2) 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, $\overline{\text{NMI}}$ | 7 |
| Reset | 0 |
| Other | Not changed |

**(3) 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).

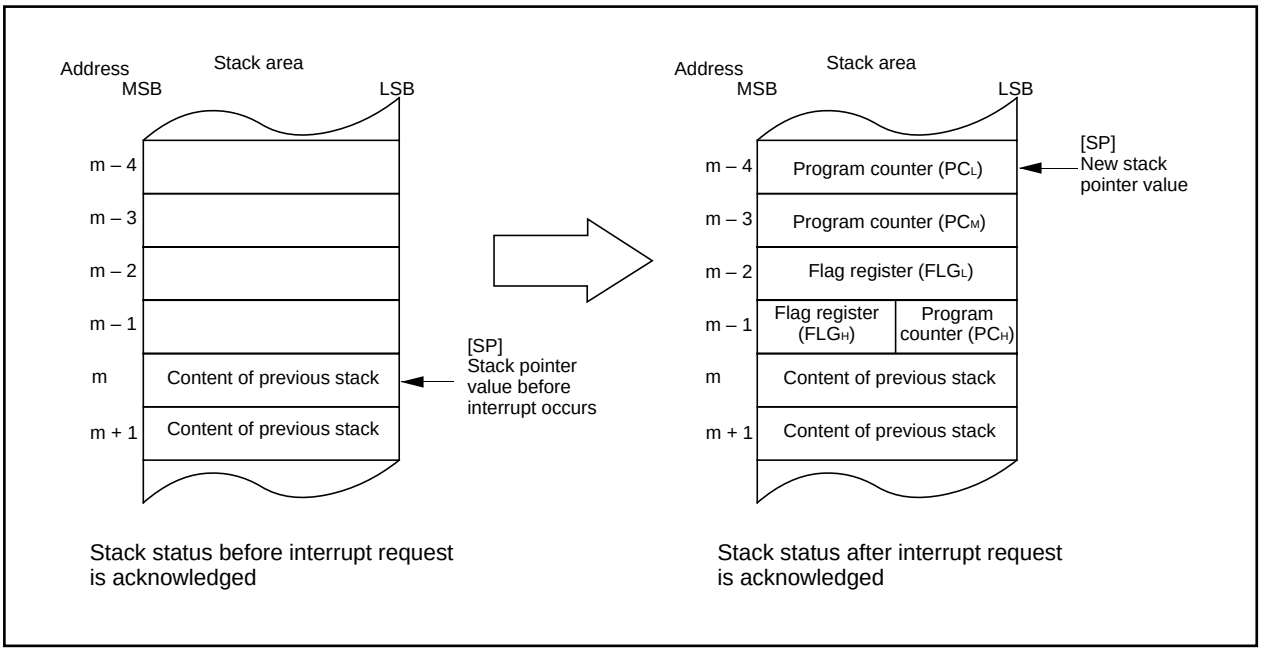



**Figure 2.7.6 State of stack before and after acceptance of interrupt request**

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.

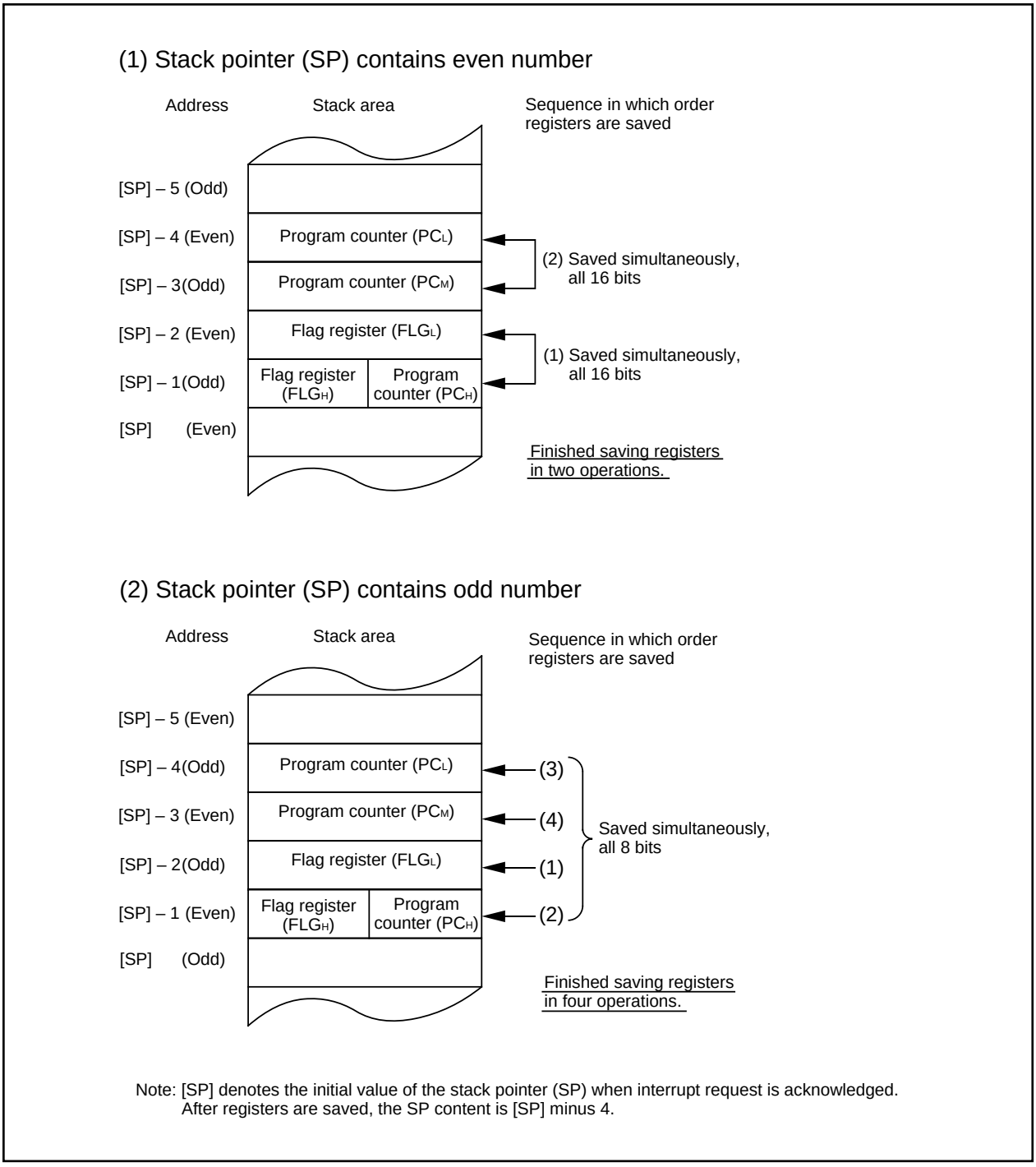



**Figure 2.7.7 Operation of saving registers**

**(4) 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.

**(5) 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.

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

**Figure 2.7.8 Hardware interrupts priorities**

**(6) Interrupt 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.

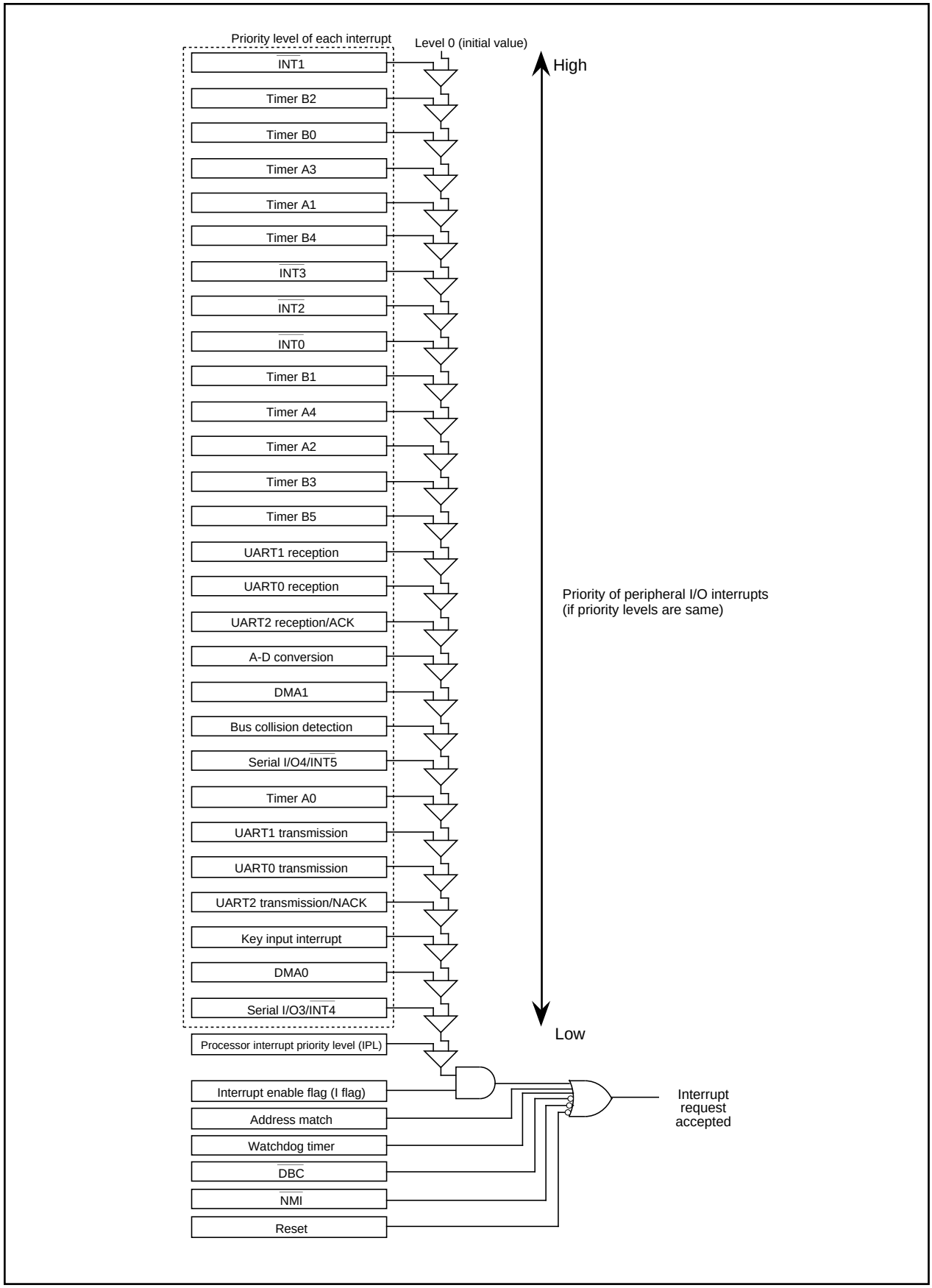



**Figure 2.7.9 Maskable interrupts priorities (peripheral I/O interrupts)**

### 2.7.7 $\overline{\text{INT}}$ Interrupt

$\overline{\text{INT0}}$ to $\overline{\text{INT5}}$ are triggered by the edges of external inputs. The edge polarity is selected using the polarity select bit.

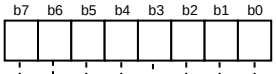
Of interrupt control registers, $0048_{16}$ is used both as serial I/O4 and external interrupt $\overline{\text{INT5}}$ input control register, and $0049_{16}$ is used both as serial I/O3 and as external interrupt $\overline{\text{INT4}}$ input control register. Use the interrupt request cause select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

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

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

Figure 2.7.10 shows the Interrupt request cause select register.

Interrupt request cause select register




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

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

**Figure 2.7.10 Interrupt request cause select register**

### 2.7.8 $\overline{\text{NMI}}$ Interrupt

An $\overline{\text{NMI}}$ interrupt is generated when the input to the P8$_5$/$\overline{\text{NMI}}$ pin changes from “H” to “L”. The $\overline{\text{NMI}}$ interrupt is a non-maskable external interrupt. The pin level can be checked in the port P8$_5$ register (bit 5 at address 03F0$_{16}$).

This pin cannot be used as a normal port input.

### 2.7.9 Key Input Interrupt

If the direction register of any of P10$_4$ to P10$_7$ is set for input and a falling edge is input to that port, a key input interrupt is generated. A key input interrupt can also be used as a key-on wakeup function for cancelling the wait mode or stop mode. However, if you intend to use the key input interrupt, do not use P10$_4$ to P10$_7$ as A-D input ports. Figure 2.7.11 shows the block diagram of the key input interrupt. Note that if an “L” level is input to any pin that has not been disabled for input, inputs to the other pins are not detected as an interrupt.

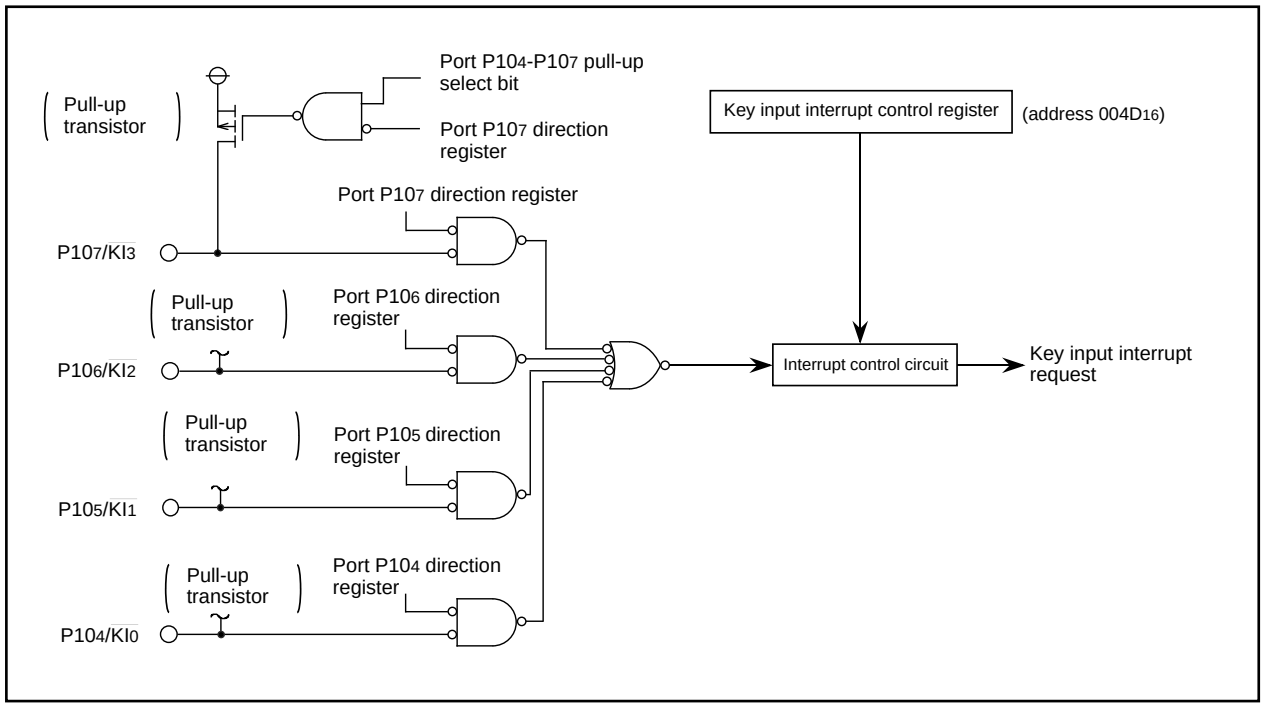



**Figure 2.7.11 Block diagram of key input interrupt**

### 2.7.10 Address Match Interrupt

An address match interrupt is generated when the address match interrupt address register contents match the program counter value. Two address match interrupts can be set, each of which can be enabled and disabled by an address match interrupt enable bit. Address match interrupts are not affected by the interrupt enable flag (I flag) and processor interrupt priority level (IPL). The value of the program counter (PC) for an address match interrupt varies depending on the instruction being executed. Figure 2.7.12 shows the address match interrupt-related registers.

Address match interrupt enable register

b7 b6 b5 b4 b3 b2 b1 b0

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

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

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

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

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

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

**Figure 2.7.12 Address match interrupt-related registers**

## 2.7.11 Precautions for Interrupts

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

• When maskable interrupt is occurred, CPU read the interrupt information (the interrupt number and interrupt request level) in the interrupt sequence. The interrupt request bit of the certain interrupt written in address $00000_{16}$ will then be set to "0". Reading address $00000_{16}$ by software sets enabled highest priority interrupt source request bit to "0".
Though the interrupt is generated, the interrupt routine may not be executed.
Do not read address $00000_{16}$ by software.

**(2) Setting the stack pointer**

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

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

• As for the $\overline{\text{NMI}}$ interrupt pin, an interrupt cannot be disabled. Connect it to the Vcc pin via a resistor (pull-up) if unused. Be sure to work on it.

• The $\overline{\text{NMI}}$ pin also serves as $P8_5$, which is exclusively input. Reading the contents of the P8 register allows reading the pin value. Use the reading of this pin only for establishing the pin level at the time when the $\overline{\text{NMI}}$ interrupt is input.

• Do not reset the CPU with the input to the $\overline{\text{NMI}}$ pin being in the "L" state.

• Do not attempt to go into stop mode with the input to the $\overline{\text{NMI}}$ pin being in the "L" state. With the input to the $\overline{\text{NMI}}$ being in the "L" state, the CM10 is fixed to "0", so attempting to go into stop mode is turned down.

• Do not attempt to go into wait mode with the input to the $\overline{\text{NMI}}$ pin being in the "L" state. With the input to the $\overline{\text{NMI}}$ pin being in the "L" state, the CPU stops but the oscillation does not stop, so no power is saved. In this instance, the CPU is returned to the normal state by a later interrupt.

• Signals input to the $\overline{\text{NMI}}$ pin require an "L" level of 1 clock or more, from the operation clock of the CPU.

**(4) External interrupt**

• Either an "L" level or an "H" level of at least 250 ns width is necessary for the signal input to pins $\overline{\text{INT}_0}$ through $\overline{\text{INT}_5}$ regardless of the CPU operation clock.

• When the polarity of the $\overline{\text{INT}_0}$ to $\overline{\text{INT}_5}$ pins is changed, the interrupt request bit is sometimes set to "1". After changing the polarity, set the interrupt request bit to "0". Figure 2.7.13 shows the procedure for changing the $\overline{\text{INT}}$ interrupt generate factor.

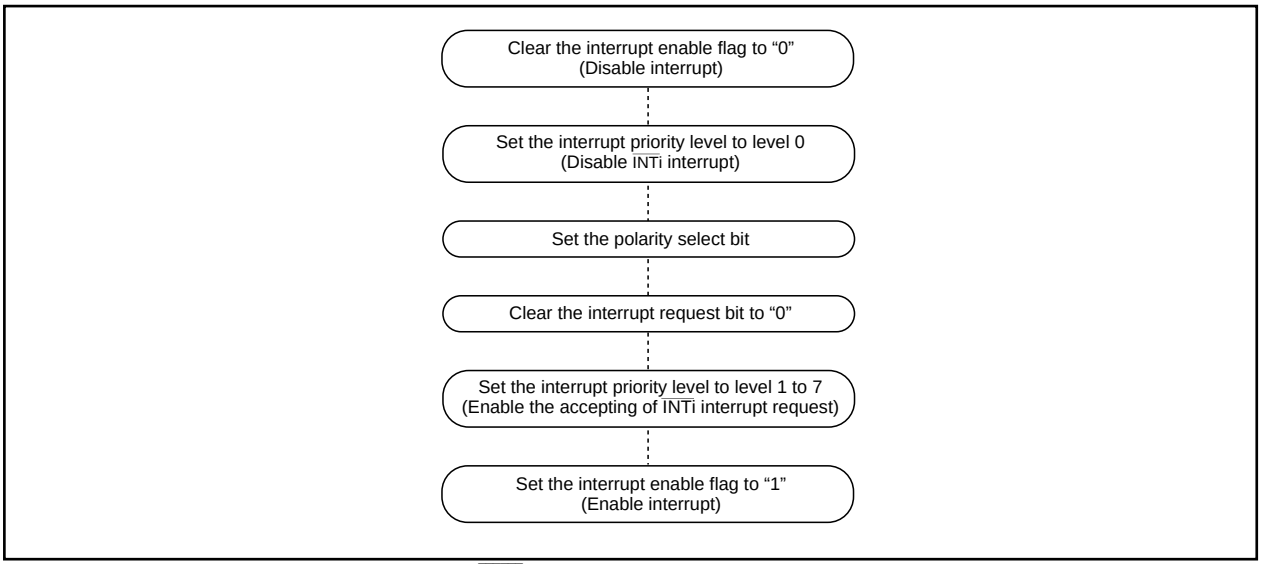



**Figure 2.7.13 Switching condition of $\overline{INT}$ interrupt request**

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

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

**Example 1:**

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

**Example 2:**

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

**Example 3:**

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

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

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

Instructions : AND, OR, BCLR, BSET

## 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 $X_{IN}$ is selected for the BCLK, bit 7 of the watchdog timer control register (address $000F_{16}$) selects the prescaler division ratio (by 16 or by 128). When $X_{CIN}$ is selected as the BCLK, the prescaler is set for division by 2 regardless of bit 7 of the watchdog timer control register (address $000F_{16}$). Thus the watchdog timer's period can be calculated as given below. The watchdog timer's period is, however, subject to an error due to the pre-scaler.

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

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

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

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

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

The watchdog timer is initialized by writing to the watchdog timer start register (address $000E_{16}$) and when a watchdog timer interrupt request is generated. The prescaler is initialized only when the microcomputer is reset. After a reset is cancelled, the watchdog timer and prescaler are both stopped. The count is started by writing to the watchdog timer start register (address $000E_{16}$).
Figure 2.8.1 shows the block diagram of the watchdog timer. Figure 2.8.2 shows the watchdog timer-related registers.

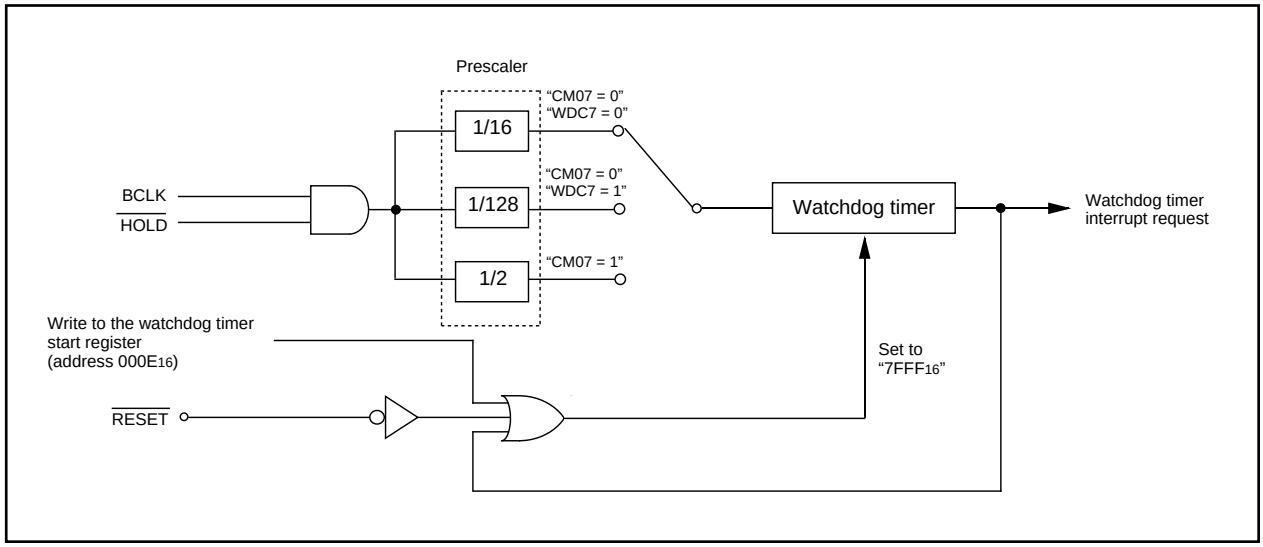



**Figure 2.8.1 Block diagram of watchdog timer**

Watchdog timer control register

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

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

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

Watchdog timer start register

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

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

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

**Figure 2.8.2 Watchdog timer control and start registers**

## 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 2.9.4 show the registers used by the DMAC.

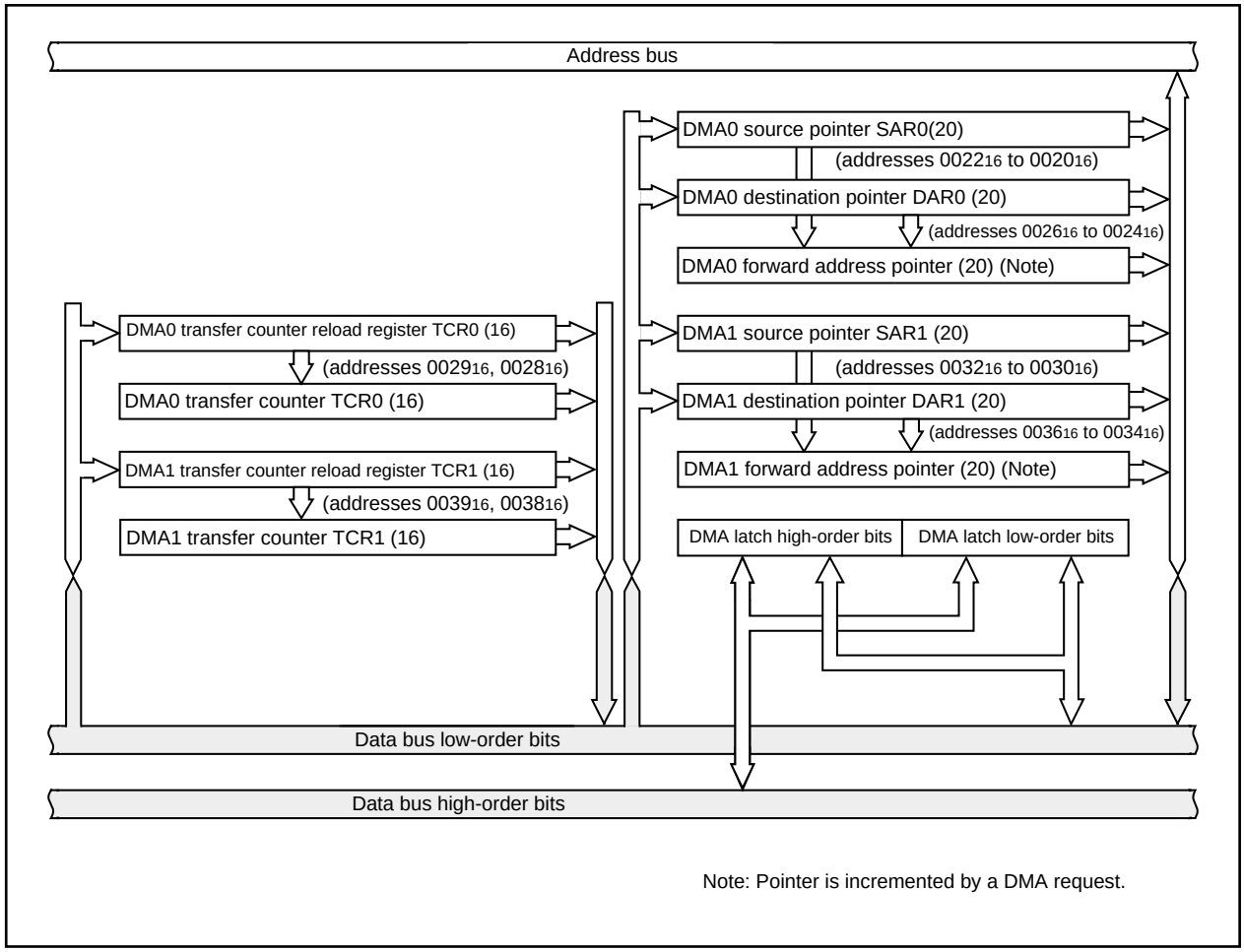



**Figure 2.9.1 Block diagram of DMAC**

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.

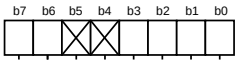
If the DMAC is active (the DMA enable bit is set to 1), data transfer starts every time a DMA transfer request signal occurs. If the cycle of the occurrences of DMA transfer request signals is higher than the DMA transfer cycle, there can be instances in which the number of transfer requests doesn't 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 | • From any address in the 1M bytes space to a fixed address<br>• From a fixed address to any address in the 1M bytes space<br>• From a fixed address to a fixed address<br>(Note that DMA-related registers [$0020_{16}$ to $003F_{16}$] cannot be accessed) |
| Maximum No. of bytes transferred | 128K bytes (with 16-bit transfers) or 64K bytes (with 8-bit transfers) |
| DMA request factors (Note) | Falling edge of $\overline{INT0}$ or $\overline{INT1}$ ($\overline{INT0}$ can be selected by DMA0, $\overline{INT1}$ by DMA1) or both edge<br>Timer A0 to timer A4 interrupt requests<br>Timer B0 to timer B5 interrupt requests<br>UART0 transfer and reception interrupt requests<br>UART1 transfer and reception interrupt requests<br>UART2 transfer and reception interrupt requests<br>Serial I/O3, 4 interrpt requests<br>A-D conversion interrupt requests<br>Software triggers |
| Channel priority | DMA0 takes precedence if DMA0 and DMA1 requests are generated simultaneously |
| Transfer unit | 8 bits or 16 bits |
| Transfer address direction | forward/fixed (forward direction cannot be specified for both source and destination simultaneously) |
| Transfer mode | • Single transfer mode<br>After the transfer counter underflows, the DMA enable bit turns to "0", and the DMAC turns inactive<br>• Repeat transfer mode<br>After the transfer counter underflows, the value of the transfer counter reload register is reloaded to the transfer counter.<br>The DMAC remains active unless a "0" is written to the DMA enable bit. |
| DMA interrupt request generation timing | When an underflow occurs in the transfer counter |
| Active | When the DMA enable bit is set to "1", the DMAC is active.<br>When the DMAC is active, data transfer starts every time a DMA transfer request signal occurs. |
| Inactive | • When the DMA enable bit is set to "0", the DMAC is inactive.<br>• After the transfer counter underflows in single transfer mode |
| Forward address pointer and reload timing for transfer counter | At the time of starting data transfer immediately after turning the DMAC active, the value of one of source pointer and destination pointer - the one specified for the forward direction - is reloaded to the forward direction address pointer,and the value of the transfer counter reload register is reloaded to the transfer counter. |
| Writing to register | Registers specified for forward direction transfer are always write enabled.<br>Registers specified for fixed address transfer are write-enabled when the DMA enable bit is "0". |
| Reading the register | Can be read at any time.<br>However, when the DMA enable bit is "1", reading the register set up as the forward register is the same as reading the value of the forward address pointer. |

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

DMA0 request cause select register

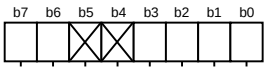



Symbol: DM0SL | Address: 03B8$_{16}$ | When reset: 00$_{16}$

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

**Figure 2.9.2 DMAC register (1)**

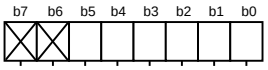
DMA1 request cause select register




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

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

DMAi control register



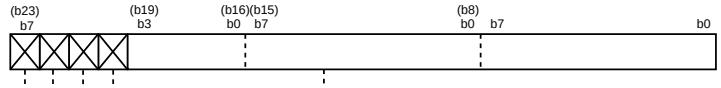

Symbol: DMiCON(i=0,1) | Address: 002C16, 003C16 | When reset: 00000X002

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

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

**Figure 2.9.3 DMAC register (2)**

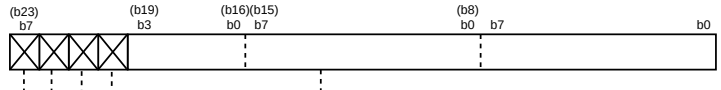
DMAi source pointer (i = 0, 1)




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

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

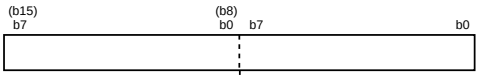
DMAi destination pointer (i = 0, 1)




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

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

DMAi transfer counter (i = 0, 1)




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

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

**Figure 2.9.4 DMAC register (3)**

## (1) Transfer cycle

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

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

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

**(b) Effect of BYTE pin level**

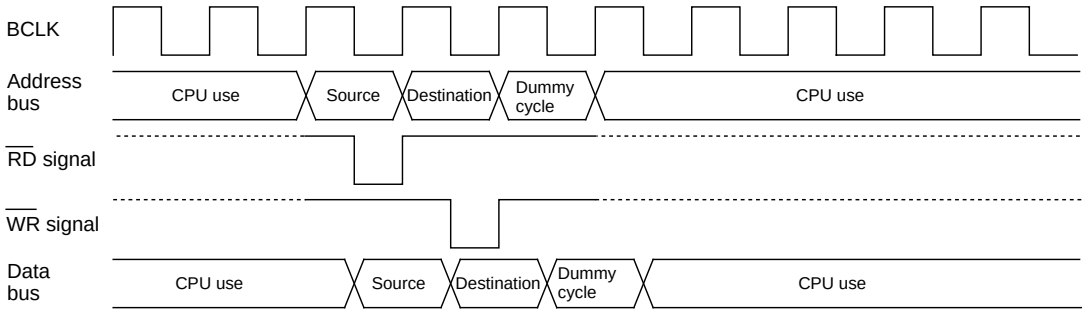
When transferring 16-bit data over an 8-bit data bus (BYTE pin = “H”) , the 16 bits of data are sent in two 8-bit blocks. Therefore, two bus cycles are required for reading the data and two are required for writing the data. Also, in contrast to when the CPU accesses internal memory, when the DMAC accesses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

**(c) Effect of software wait**

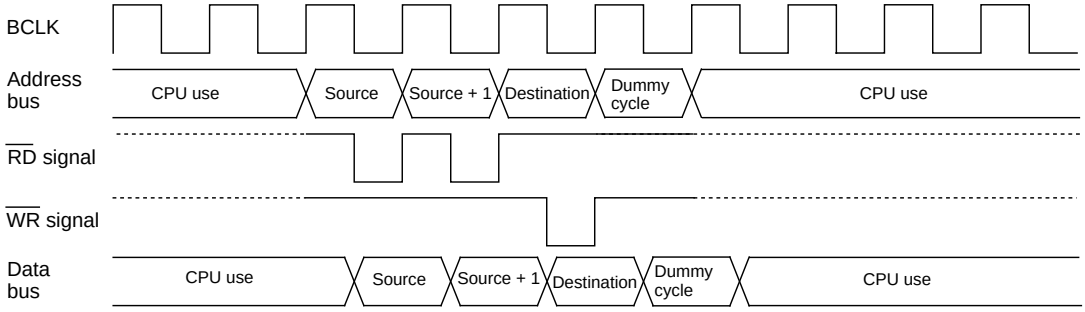
When the SFR area or a memory area with a software wait is accessed, the number of cycles is increased for the wait by 1 bus cycle. The length of the cycle is determined by BCLK.

Figure 2.9.5 shows the example of the transfer cycles for a source read. For convenience, the destination write cycle is shown as one cycle and the source read cycles for the different conditions are shown. In reality, the destination write cycle is subject to the same conditions as the source read cycle, with the transfer cycle changing accordingly. When calculating the transfer cycle, remember to apply the respective conditions to both the destination write cycle and the source read cycle. For example (2) in Figure 2.9.5, if data is being transferred in 16-bit units on an 8-bit bus, two bus cycles are required for both the source read cycle and the destination write cycle.

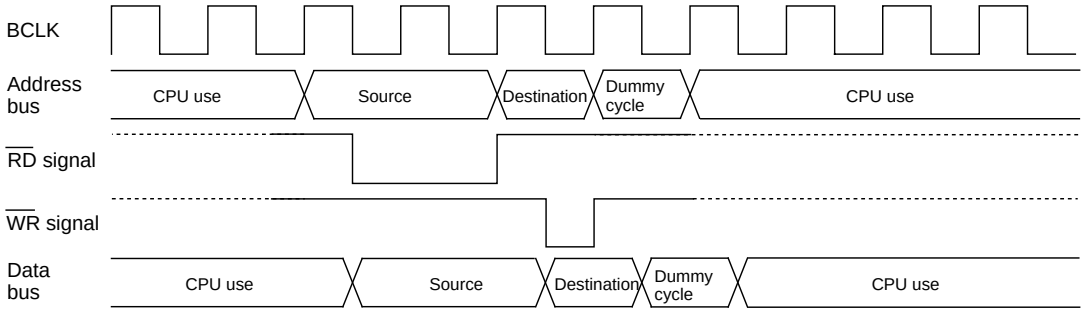
**(1) 8-bit transfers**
**16-bit transfers from even address and the source address is even.**



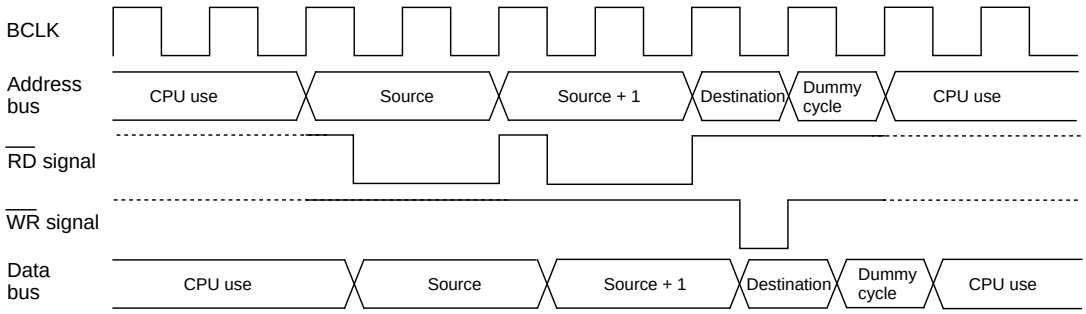

**(2) 16-bit transfers and the source address is odd**
**Transferring 16-bit data on an 8-bit data bus (In this case, there are also two destination write cycles).**




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




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




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

**Figure 2.9.5 Example of the transfer cycles for a source read**

## (2) DMAC transfer cycles

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

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

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

**Table 2.9.2 No. of DMAC transfer cycles**

| Transfer unit | Bus width | Access address | Microprocessor mode | |
|---|---|---|---|---|
| | | | No. of read cycles | No. of write cycles |
| 8-bit transfers (DMBIT= “1”) | 16-bit (BYTE= “L”) | Even | 1 | 1 |
| | | Odd | 1 | 1 |
| | 8-bit (BYTE = “H”) | Even | 1 | 1 |
| | | Odd | 1 | 1 |
| 16-bit transfers (DMBIT= “0”) | 16-bit (BYTE = “L”) | Even | 1 | 1 |
| | | Odd | 2 | 2 |
| | 8-bit (BYTE = “H”) | Even | 2 | 2 |
| | | Odd | 2 | 2 |

**Coefficient j, k**

| Internal memory | | | External memory | | |
|---|---|---|---|---|---|
| Internal RAM No wait | Internal RAM With wait | SFR area | Separate bus No wait | Separate bus With wait | Multiplex bus |
| 1 | 2 | 2 | 1 | 2 | 3 |

### 2.9.1 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.

### 2.9.2 DMA request bit

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

DMA request factors include the following.

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

For the selection of DMA request factors, see the descriptions of the DMAi factor selection register.

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

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

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

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

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

**(1) Internal factors**

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

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

**(2) External factors**

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

Selecting the INTi pins as external factors using the DMA request factor selection bit causes input from these pins to become the DMA transfer request signals.

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

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

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

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

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

Figure 2.9.6 An example of DMA transfer effected by external factors.

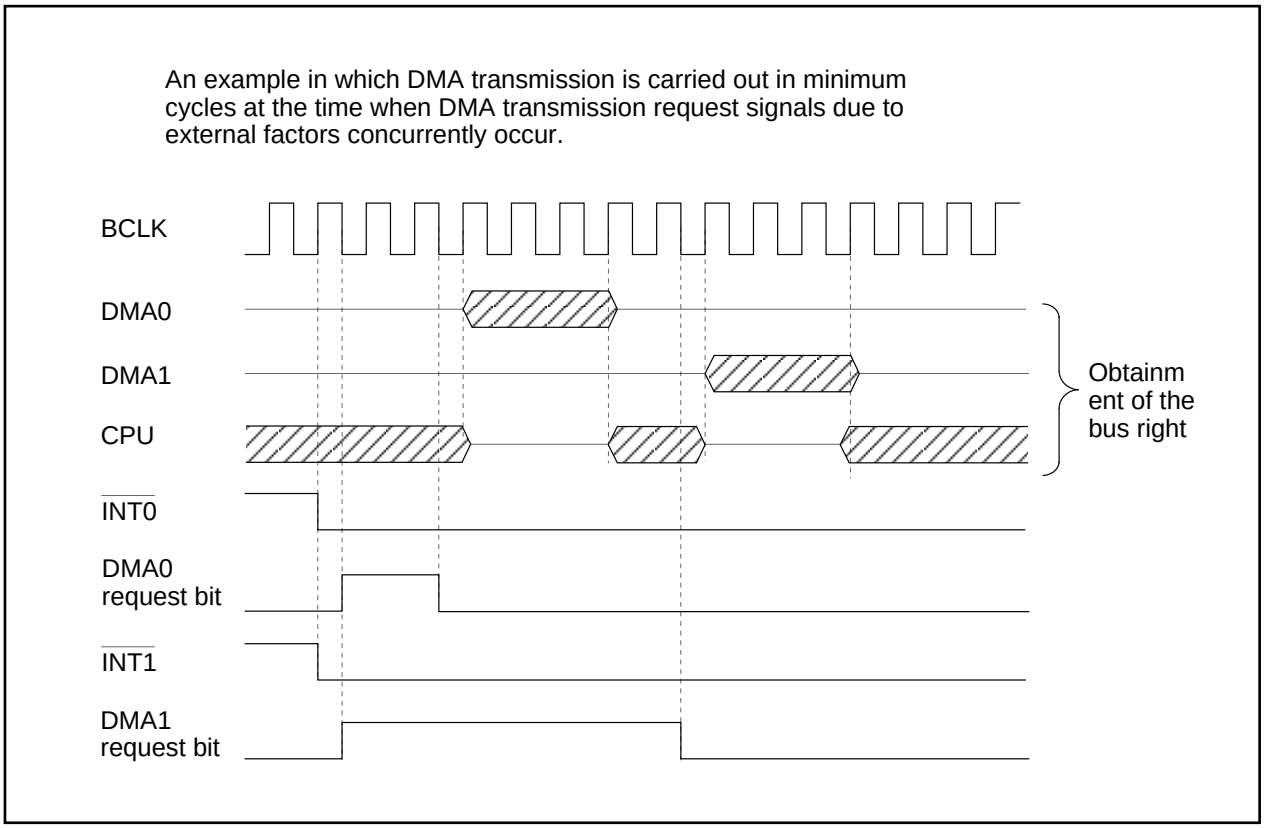
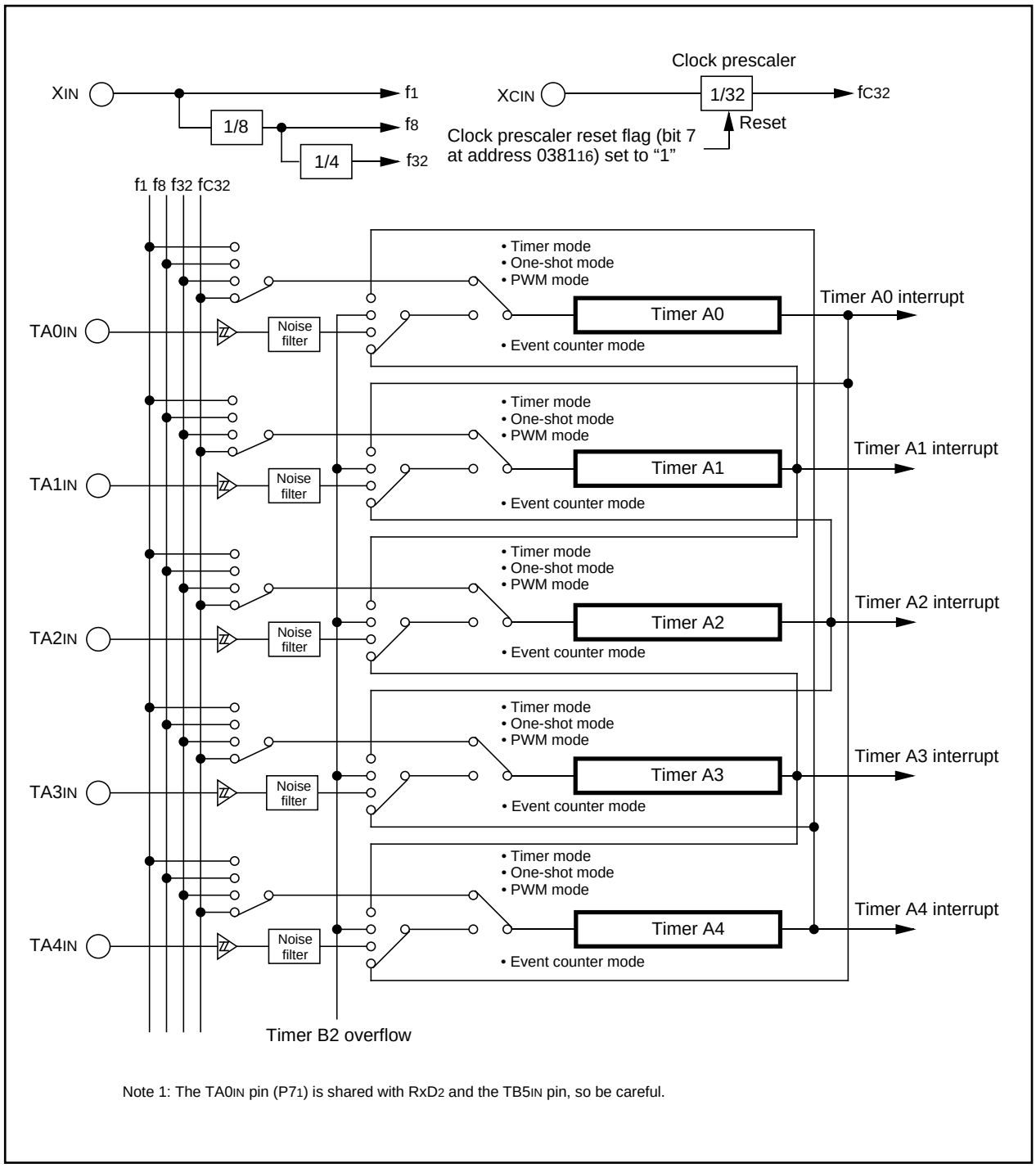



**Figure 2.9.6 An example of DMA transfer effected by external factors**

## 2.10 Timer

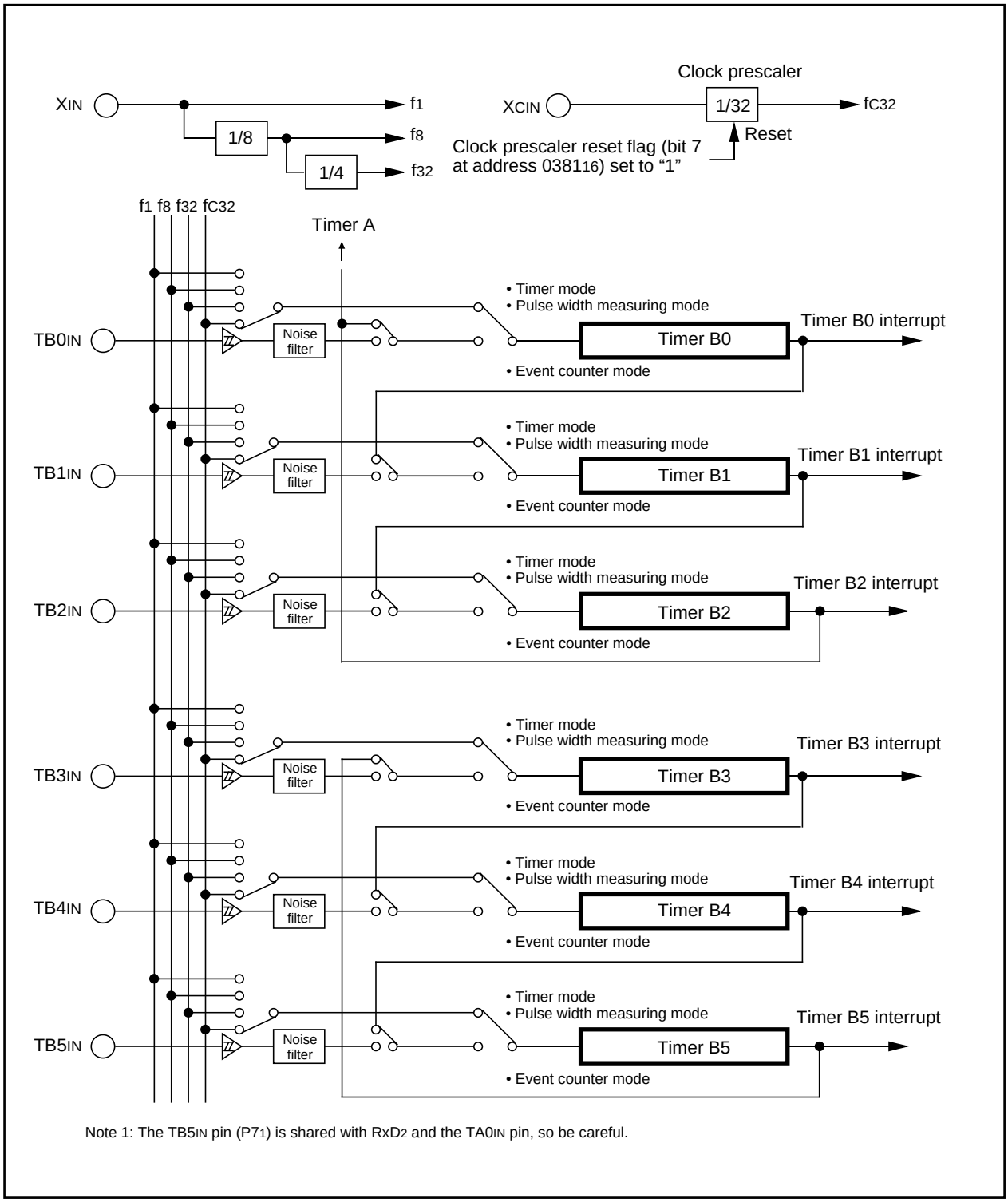
There are eleven 16-bit timers. These timers can be classified by function into timers A (five) and timers B (six). All these timers function independently.

Figures 2.10.1 and 2.10.2 show the block diagram of timers.




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

**Figure 2.10.1 Timer A block diagram**




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

**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 2.10.6 show the timer A-related registers.

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

Timer A has the four operation modes listed as follows:

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

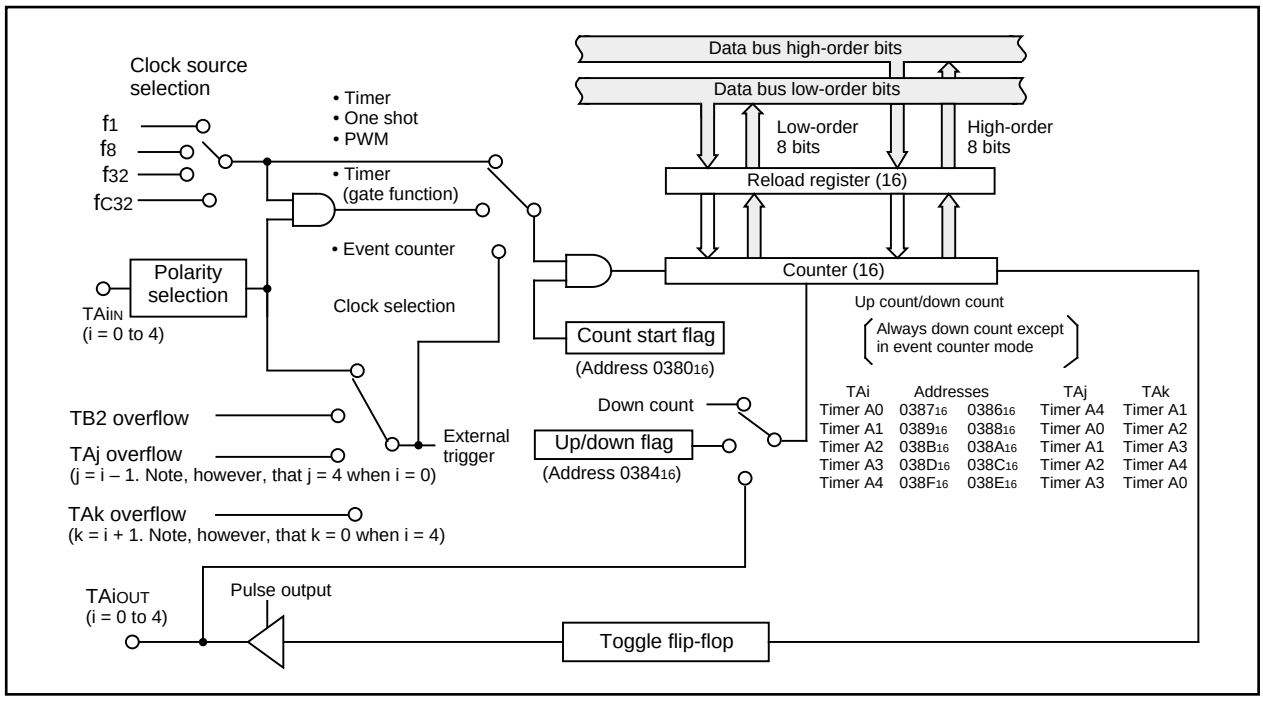



**Figure 2.10.3 Block diagram of timer A**

Timer Ai mode register

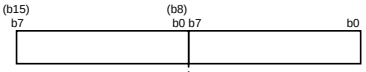
| Symbol | Address | When reset |
|---|---|---|
| TAiMR(i=0 to 4) | $0396_{16}$ to $039A_{16}$ | $00_{16}$ |

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

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

**Figure 2.10.4 Timer A-related registers (1)**

Timer Ai register (Note)




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

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

Note 1: Read and write data in 16-bit units.
Note 2: In the case of using “Event counter mode” as “Free-Run type”, the timer register contents may be unkown when counting begins.(Refer 3. Usage Precaution.)

Count start flag


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

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

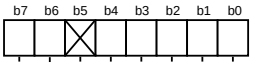
Up/down flag


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

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

**Figure 2.10.5 Timer A-related registers (2)**

### One-shot start flag

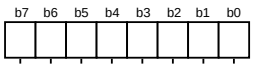



Symbol: ONSF | Address: $0382_{16}$ | When reset: $00X00000_2$

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

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

### Trigger select register

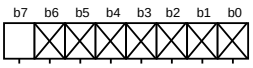



Symbol: TRGSR | Address: $0383_{16}$ | When reset: $00_{16}$

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

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

### Clock prescaler reset flag




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

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

**Figure 2.10.6 Timer A-related registers (3)**

**(1) Timer mode**

In this mode, the timer counts an internally generated count source. (See Table 2.10.1) Figure 2.10.7 shows the timer Ai mode register in timer mode.

**Table 2.10.1 Specifications of timer mode**

| Item | Specification |
|---|---|
| Count source | $f_1$, $f_8$, $f_{32}$, $f_{C32}$ |
| Count operation | • Down count<br>• When the timer underflows, it reloads the reload register contents before continuing counting |
| Divide ratio | 1/(n+1) n : Set value |
| Count start condition | Count start flag is set (= 1) |
| Count stop condition | Count start flag is reset (= 0) |
| Interrupt request generation timing | When the timer underflows |
| TAiIN pin function | Programmable I/O port or gate input |
| TAiOUT pin function | Programmable I/O port or pulse output |
| Read from timer | Count value can be read out by reading timer Ai register |
| Write to timer | • When counting stopped<br>When a value is written to timer Ai register, it is written to both reload register and counter<br>• When counting in progress<br>When a value is written to timer Ai register, it is written to only reload register (Transferred to counter at next reload time) |
| Select function | • Gate function<br>Counting can be started and stopped by the TAiIN pin's input signal<br>• Pulse output function<br>Each time the timer underflows, the TAiOUT pin's polarity is reversed |

Timer Ai mode register

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

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

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

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

**Figure 2.10.7 Timer Ai mode register in timer mode**

**(2) Event counter mode**

In this mode, the timer counts an external signal or an internal timer's overflow. Timers A0 and A1 can count a single-phase external signal. Timers A2, A3, and A4 can count a single-phase and a two-phase external signal. Table 2.10.2 lists timer specifications when counting a single-phase external signal. Figure 2.10.8 shows the timer Ai mode register in event counter mode.

Table 2.10.3 lists timer specifications when counting a two-phase external signal. Figure 2.10.9 shows the timer Ai mode register in event counter mode.

**Table 2.10.2 Timer specifications in event counter mode (when not processing two-phase pulse signal)**

| Item | Specification |
|---|---|
| Count source | • External signals input to TAiIN pin (effective edge can be selected by software)<br>• TB2 overflow, TAj overflow |
| Count operation | • Up count or down count can be selected by external signal or software<br>• When the timer overflows or underflows, it reloads the reload register contents before continuing counting (Note) |
| Divide ratio | 1/ ($FFFF_{16}$ - n + 1) for up count<br>1/ (n + 1) for down count n : Set value |
| Count start condition | Count start flag is set (= 1) |
| Count stop condition | Count start flag is reset (= 0) |
| Interrupt request generation timing | The timer overflows or underflows |
| TAiIN pin function | Programmable I/O port or count source input |
| TAiOUT pin function | Programmable I/O port, pulse output, or up/down count select input |
| Read from timer | Count value can be read out by reading timer Ai register |
| Write to timer | • When counting stopped<br>When a value is written to timer Ai register, it is written to both reload register and counter<br>• When counting in progress<br>When a value is written to timer Ai register, it is written to only reload register (Transferred to counter at next reload time) |
| Select function | • Free-run count function<br>Even when the timer overflows or underflows, the reload register content is not reloaded to it<br>• Pulse output function<br>Each time the timer overflows or underflows, the TAiOUT pin's polarity is reversed |

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

Timer Ai mode register

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

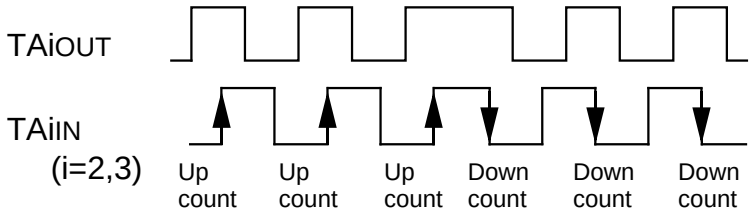
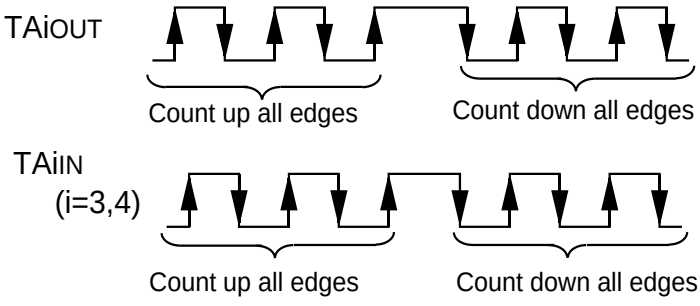
Symbol: TAiMR(i = 0, 1) Address: $0396_{16}$, $0397_{16}$ When reset: $00_{16}$

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

Note 1: In event counter mode, the count source is selected by the event / trigger select bit (addresses $0382_{16}$ and $0383_{16}$).
Note 2: The settings of the corresponding port register and port direction register are invalid.
Note 3: Valid only when counting an external signal.
Note 4: When an "L" signal is input to the TAiOUT pin, the downcount is activated. When "H", the upcount is activated. Set the corresponding port direction register to "0".
Note 5: In the case of using "Event counter mode" as "Free-Run type", the timer register contents may be unkown when counting begins.(Refer 3. Usage Precaution.)

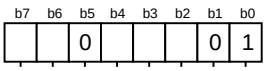
**Figure 2.10.8 Timer Ai mode register in event counter mode**

**Table 2.10.3 Timer specifications in event counter mode (when processing two-phase pulse signal with timers A2, A3, and A4)**

| Item | Specification |
|---|---|
| Count source | • Two-phase pulse signals input to TAiIN or TAiOUT pin |
| Count operation | • Up count or down count can be selected by two-phase pulse signal<br>• When the timer overflows or underflows, the reload register content is reloaded and the timer starts over again (Note) |
| Divide ratio | 1/ (FFFF16 - n + 1) for up count<br>1/ (n + 1) for down count  n : Set value |
| Count start condition | Count start flag is set (= 1) |
| Count stop condition | Count start flag is reset (= 0) |
| Interrupt request generation timing | Timer overflows or underflows |
| TAiIN pin function | Two-phase pulse input |
| TAiOUT pin function | Two-phase pulse input |
| Read from timer | Count value can be read out by reading timer A2, A3, or A4 register |
| Write to timer | • When counting stopped<br>When a value is written to timer A2, A3, or A4 register, it is written to both reload register and counter<br>• When counting in progress<br>When a value is written to timer A2, A3, or A4 register, it is written to only reload register. (Transferred to counter at next reload time.) |
| Select function | • Normal processing operation<br>The timer counts up rising edges or counts down falling edges on the TAiIN pin when input signal on the TAiOUT pin is “H”<br>  <br>• Multiply-by-4 processing operation<br>If the phase relationship is such that the TAiIN pin goes “H” when the input signal on the TAiOUT pin is “H”, the timer counts up rising and falling edges on the TAiOUT and TAiIN pins. If the phase relationship is such that the TAiIN pin goes “L” when the input signal on the TAiOUT pin is “H”, the timer counts down rising and falling edges on the TAiOUT and TAiIN pins.<br>  |

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

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

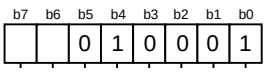



Symbol: TAiMR(i = 2 to 4) | Address: 0398$_{16}$ to 039A$_{16}$ | When reset: 00$_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>0 1 : Event counter mode | O | O |
| TMOD1 | | | O | O |
| MR0 | Pulse output function select bit | 0 : Pulse is not output (TAiOUT pin is a normal port pin)<br>1 : Pulse is output (Note 1) (TAiOUT pin is a pulse output pin) | O | O |
| MR1 | Count polarity select bit (Note 2) | 0 : Counts external signal's falling edges<br>1 : Counts external signal's rising edges | O | O |
| MR2 | Up/down switching cause select bit | 0 : Up/down flag's content<br>1 : TAiOUT pin's input signal (Note 3) | O | O |
| MR3 | 0 : (Must always be “0” in event counter mode) | | O | O |
| TCK0 | Count operation type select bit | 0 : Reload type<br>1 : Free-run type(Note 6) | O | O |
| TCK1 | Two-phase pulse signal processing operation select bit (Note 4)(Note 5) | 0 : Normal processing operation<br>1 : Multiply-by-4 processing operation | O | O |

Note 1: The settings of the corresponding port register and port direction register are invalid.
Note 2: This bit is valid when only counting an external signal.
Note 3: Set the corresponding port direction register to “0”.
Note 4: This bit is valid for the timer A3 mode register.
For timer A2 and A4 mode registers, this bit can be “0 ”or “1”.
Note 5: When performing two-phase pulse signal processing, make sure the two-phase pulse signal processing operation select bit (address 0384$_{16}$) is set to “1”. Also, always be sure to set the event/trigger select bit (addresses 0382$_{16}$ and 0383$_{16}$) to “00”.
Note 6: In the case of using “Event counter mode” as “Free-Run type”, the timer register contents may be unkown when counting begins.(Refer 3. Usage Precaution.)

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




Symbol: TAiMR(i = 2 to 4) | Address: 0398$_{16}$ to 039A$_{16}$ | When reset: 00$_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>0 1 : Event counter mode | O | O |
| TMOD1 | | | O | O |
| MR0 | 0 (Must always be “0” when using two-phase pulse signal processing) | | O | O |
| MR1 | 0 (Must always be “0” when using two-phase pulse signal processing) | | O | O |
| MR2 | 1 (Must always be “1” when using two-phase pulse signal processing) | | O | O |
| MR3 | 0 (Must always be “0” when using two-phase pulse signal processing) | | O | O |
| TCK0 | Count operation type select bit | 0 : Reload type<br>1 : Free-run type(Note 3) | O | O |
| TCK1 | Two-phase pulse processing operation select bit (Note 1)(Note 2) | 0 : Normal processing operation<br>1 : Multiply-by-4 processing operation | O | O |

Note 1: This bit is valid for timer A3 mode register.
For timer A2 and A4 mode registers, this bit can be “0” or “1”.
Note 2: When performing two-phase pulse signal processing, make sure the two-phase pulse signal processing operation select bit (address 0384$_{16}$) is set to “1”. Also, always be sure to set the event/trigger select bit (addresses 0382$_{16}$ and 0383$_{16}$) to “00”.
Note 3: In the case of using “Event counter mode” as “Free-Run type”, the timer register contents may be unkown when counting begins.(Refer 3. Usage Precaution.)

**Figure 2.10.9 Timer Ai mode register in event counter mode**

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

In this mode, the timer operates only once. (See Table 2.10.4) When a trigger occurs, the timer starts up and continues operating for a given period. Figure 2.10.10 shows the timer Ai mode register in one-shot timer mode.

**Table 2.10.4 Timer specifications in one-shot timer mode**

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

Timer Ai mode register

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

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

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

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

**Figure 2.10.10 Timer Ai mode register in one-shot timer mode**

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

In this mode, the timer outputs pulses of a given width in succession. (See Table 2.10.5) In this mode, the counter functions as either a 16-bit pulse width modulator or an 8-bit pulse width modulator.

Figure 2.10.11 shows the timer Ai mode register in pulse width modulation mode. Figure 2.10.12 shows the example of how a 16-bit pulse width modulator operates. Figure 2.10.13 shows the example of how an 8-bit pulse width modulator operates.

**Table 2.10.5 Timer specifications in pulse width modulation mode**

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

Timer Ai mode register

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

Symbol: TAiMR(i=0 to 4) Address: 039616 to 039A16 When reset: 0016

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

Note 1: Valid only when the TAiIN pin is selected by the event/trigger select bit (addresses 038216 and 038316). If timer overflow is selected, this bit can be "1" or "0".
Note 2: Set the corresponding port direction register to "0".

**Figure 2.10.11 Timer Ai mode register in pulse width modulation mode**

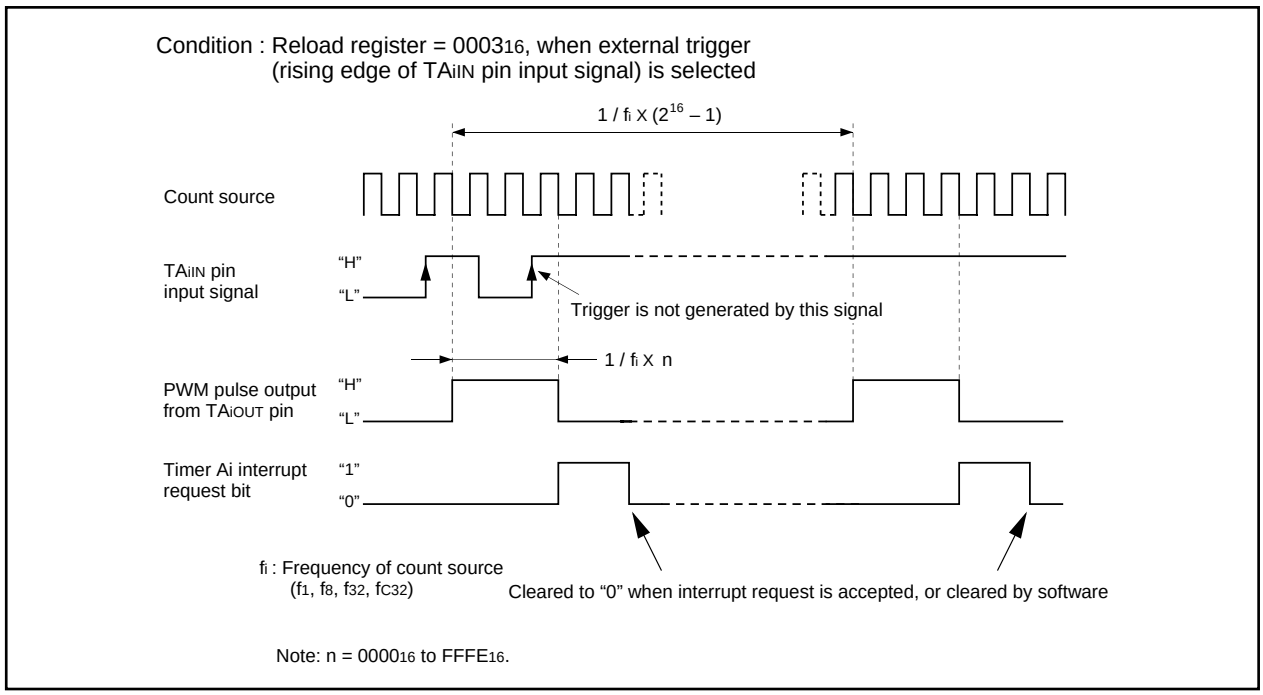



**Figure 2.10.12 Example of how a 16-bit pulse width modulator operates**

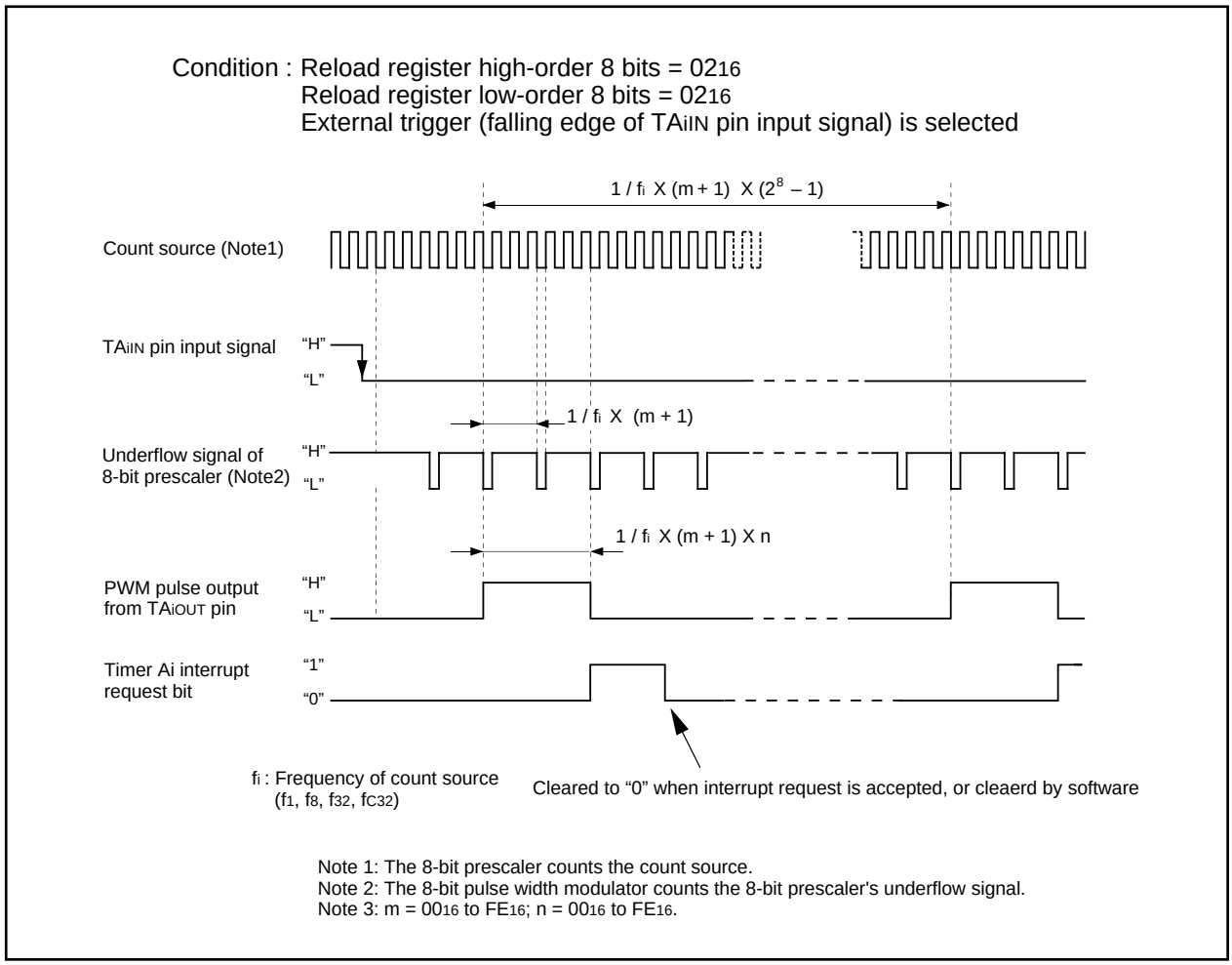



**Figure 2.10.13 Example of how an 8-bit pulse width modulator operates**

## 2.10.2 Timer B

Figure 2.10.14 shows the block diagram of timer B. Figures 2.10.15 and 2.10.16 show the timer B-related registers.

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

Timer B has three operation modes listed as follows:

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

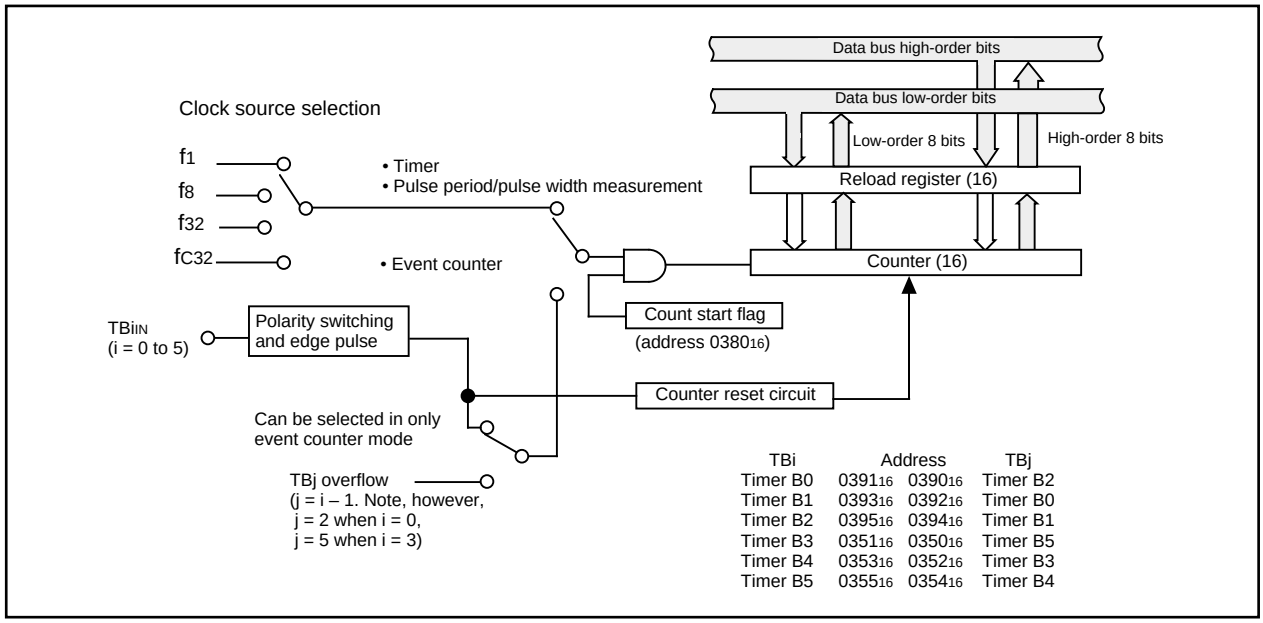



**Figure 2.10.14 Block diagram of timer B**

Timer Bi mode register

b7 b6 b5 b4 b3 b2 b1 b0

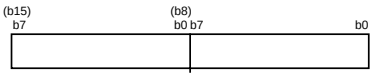
| Symbol | Address | When reset |
|---|---|---|
| TBiMR(i = 0 to 5) | 039B16 to 039D16 | 00XX00002 |
| | 035B16 to 035D16 | 00XX00002 |

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

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

**Figure 2.10.15 Timer B-related registers (1)**

Timer Bi register (Note)




| Symbol | Address | When reset |
|---|---|---|
| TB0 | $0391_{16}$, $0390_{16}$ | Indeterminate |
| TB1 | $0393_{16}$, $0392_{16}$ | Indeterminate |
| TB2 | $0395_{16}$, $0394_{16}$ | Indeterminate |
| TB3 | $0351_{16}$, $0350_{16}$ | Indeterminate |
| TB4 | $0353_{16}$, $0352_{16}$ | Indeterminate |
| TB5 | $0355_{16}$, $0354_{16}$ | Indeterminate |

| Function | Values that can be set | R | W |
|---|---|---|---|
| • Timer mode<br>Counts the timer's period | $0000_{16}$ to $FFFF_{16}$ | O | O |
| • Event counter mode<br>Counts external pulses input or a timer overflow | $0000_{16}$ to $FFFF_{16}$ | O | O |
| • Pulse period / pulse width measurement mode<br>Measures a pulse period or width | —— | O | X |

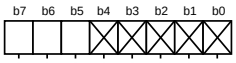
Note: Read and write data in 16-bit units.

Count start flag


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

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

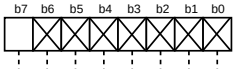
Timer B3, 4, 5 count start flag




| Symbol | Address | When reset |
|---|---|---|
| TBSR | $0340_{16}$ | $000XXXXX_2$ |

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

Clock prescaler reset flag




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

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

**Figure 2.10.16 Timer B-related registers (2)**

### (1) Timer mode

In this mode, the timer counts an internally generated count source. (See Table 2.10.6) Figure 2.10.17 shows the timer Bi mode register in timer mode.

**Table 2.10.6 Timer specifications in timer mode**

| Item | Specification |
|---|---|
| Count source | $f_1$, $f_8$, $f_{32}$, $f_{C32}$ |
| Count operation | • Counts down<br>• When the timer underflows, it reloads the reload register contents before continuing counting |
| Divide ratio | 1/(n+1) n : Set value |
| Count start condition | Count start flag is set (= 1) |
| Count stop condition | Count start flag is reset (= 0) |
| Interrupt request generation timing | The timer underflows |
| $TBi_{IN}$ pin function | Programmable I/O port |
| Read from timer | Count value is read out by reading timer Bi register |
| Write to timer | • When counting stopped<br>When a value is written to timer Bi register, it is written to both reload register and counter<br>• When counting in progress<br>When a value is written to timer Bi register, it is written to only reload register (Transferred to counter at next reload time) |

Timer Bi mode register

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

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

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

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

**Figure 2.10.17 Timer Bi mode register in timer mode**

**(2) Event counter mode**

In this mode, the timer counts an external signal or an internal timer's overflow. (See Table 2.10.7)

Figure 2.10.18 shows the timer Bi mode register in event counter mode.

**Table 2.10.7 Timer specifications in event counter mode**

| Item | Specification |
|---|---|
| Count source | • External signals input to TBiIN pin<br>• Effective edge of count source can be a rising edge, a falling edge, or falling and rising edges as selected by software |
| Count operation | • Counts down<br>• When the timer underflows, it reloads the reload register contents before continuing counting |
| Divide ratio | 1/(n+1) n : Set value |
| Count start condition | Count start flag is set (= 1) |
| Count stop condition | Count start flag is reset (= 0) |
| Interrupt request generation timing | The timer underflows |
| TBiIN pin function | Count source input |
| Read from timer | Count value can be read out by reading timer Bi register |
| Write to timer | • When counting stopped<br>When a value is written to timer Bi register, it is written to both reload register and counter<br>• When counting in progress<br>When a value is written to timer Bi register, it is written to only reload register (Transferred to counter at next reload time) |

Timer Bi mode register

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

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

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

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

**Figure 2.10.18 Timer Bi mode register in event counter mode**

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

In this mode, the timer measures the pulse period or pulse width of an external signal. (See Table 2.10.8) Figure 2.10.19 shows the timer Bi mode register in pulse period/pulse width measurement mode. Figure 2.10.20 shows the operation timing when measuring a pulse period. Figure 2.10.21 shows the operation timing when measuring a pulse width.

**Table 2.10.8 Timer specifications in pulse period/pulse width measurement mode**

| Item | Specification |
|---|---|
| Count source | $f_1$, $f_8$, $f_{32}$, $f_{C32}$ |
| Count operation | • Up count<br>• Counter value "$0000_{16}$" is transferred to reload register at measurement pulse's effective edge and the timer continues counting |
| Count start condition | Count start flag is set (= 1) |
| Count stop condition | Count start flag is reset (= 0) |
| Interrupt request generation timing | • When measurement pulse's effective edge is input (Note 1)<br>• When an overflow occurs. (Simultaneously, the timer Bi overflow flag changes to "1". The timer Bi overflow flag changes to "0" when the count start flag is "1" and a value is written to the timer Bi mode register.) |
| $TBi_{IN}$ pin function | Measurement pulse input |
| Read from timer | When timer Bi register is read, it indicates the reload register's content (measurement result) (Note 2) |
| Write to timer | Cannot be written to |

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

Timer Bi mode register

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

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

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

Note 1: The timer Bi overflow flag changes to "0" when the count start flag is "1" and a value is written to the timer Bi mode register. This flag cannot be set to "1" by software.
Note 2: Timer B0, timer B3.
Note 3: Timer B1, timer B2, timer B4, timer B5.

**Figure 2.10.19 Timer Bi mode register in pulse period/pulse width measurement mode**

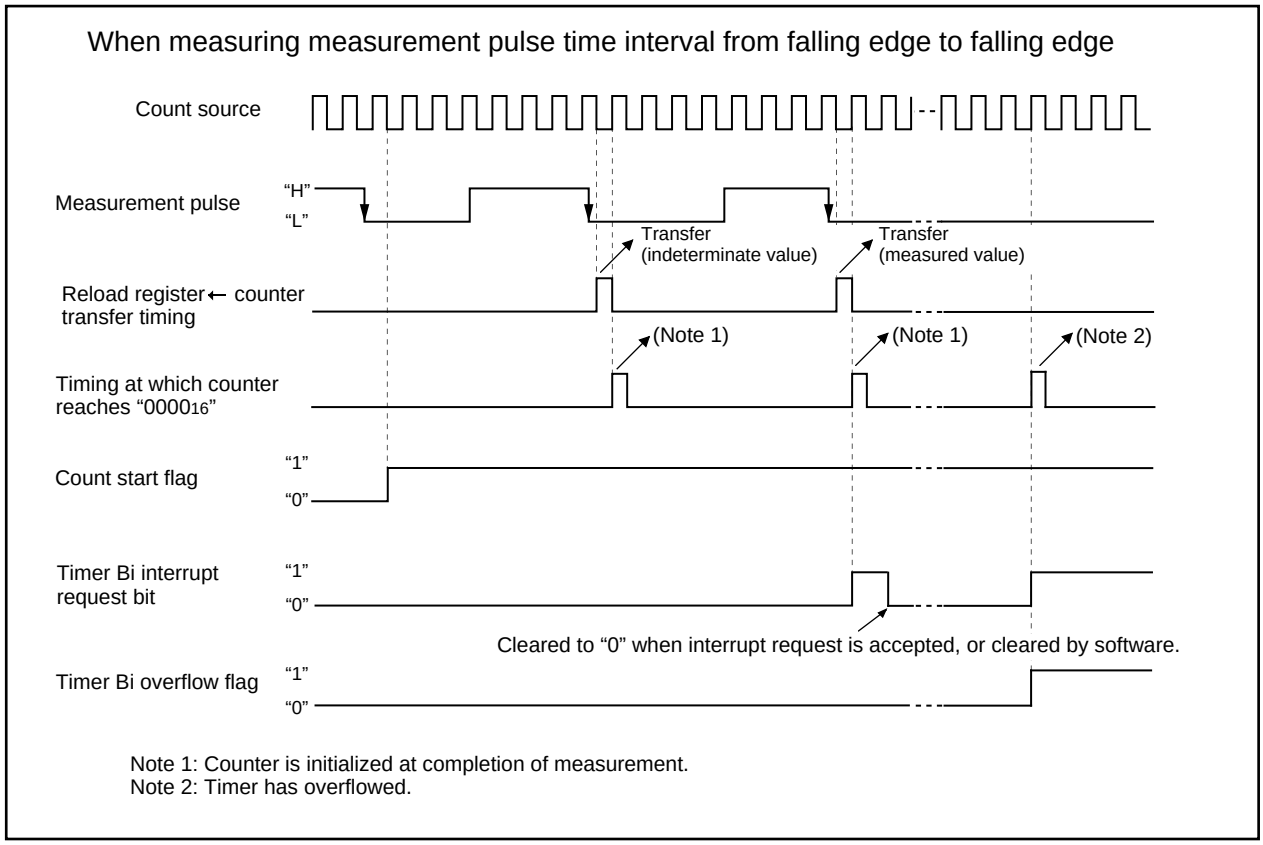



**Figure 2.10.20 Operation timing when measuring a pulse period**

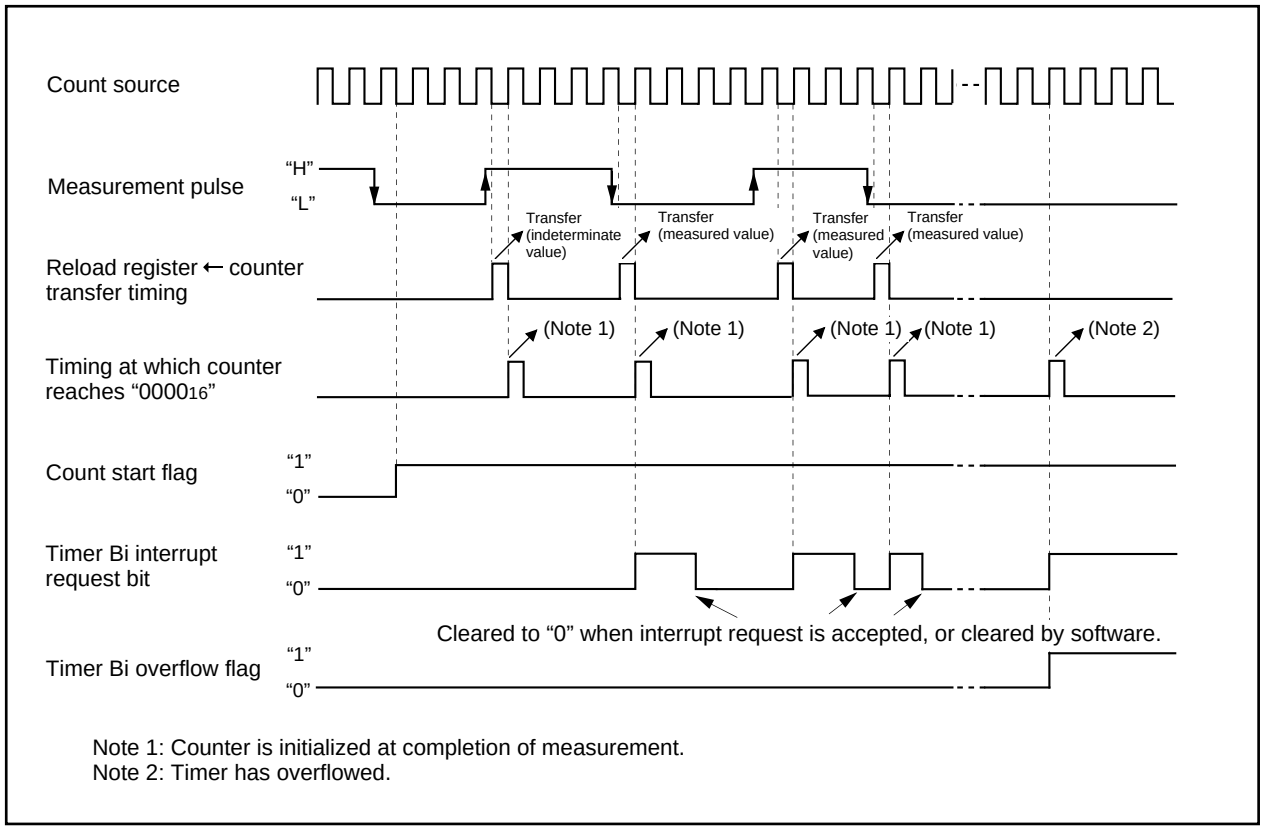



**Figure 2.10.21 Operation timing when measuring a pulse width**

## 2.11 Serial I/O

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

### 2.11.1 UART0 to 2

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

Figure 2.11.1 shows the block diagram of UART0, UART1 and UART2. Figures 2.11.2 and 2.11.3 show the block diagram of the transmit/receive unit.

UARTi (i = 0 to 2) has two operation modes: a clock synchronous serial I/O mode and a clock asynchronous serial I/O mode (UART mode). The contents of the serial I/O mode select bits (bits 0 to 2 at addresses $03A0_{16}$, $03A8_{16}$ and $0378_{16}$) determine whether UARTi is used as a clock synchronous serial I/O or as a UART. Although a few functions are different, UART0, UART1 and UART2 have almost the same functions.

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

Table 2.11.1 shows the comparison of functions of UART0 through UART2, and Figures 2.11.4 to 2.11.8 show the registers related to UARTi.

Note: SIM : Subscriber Identity Module

**Table 2.11.1 Comparison of functions of UART0 through UART2**

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

Note 1: Only when clock synchronous serial I/O mode.
Note 2: Only when clock synchronous serial I/O mode and 8-bit UART mode.
Note 3: Only when UART mode.
Note 4: Using for SIM interface.

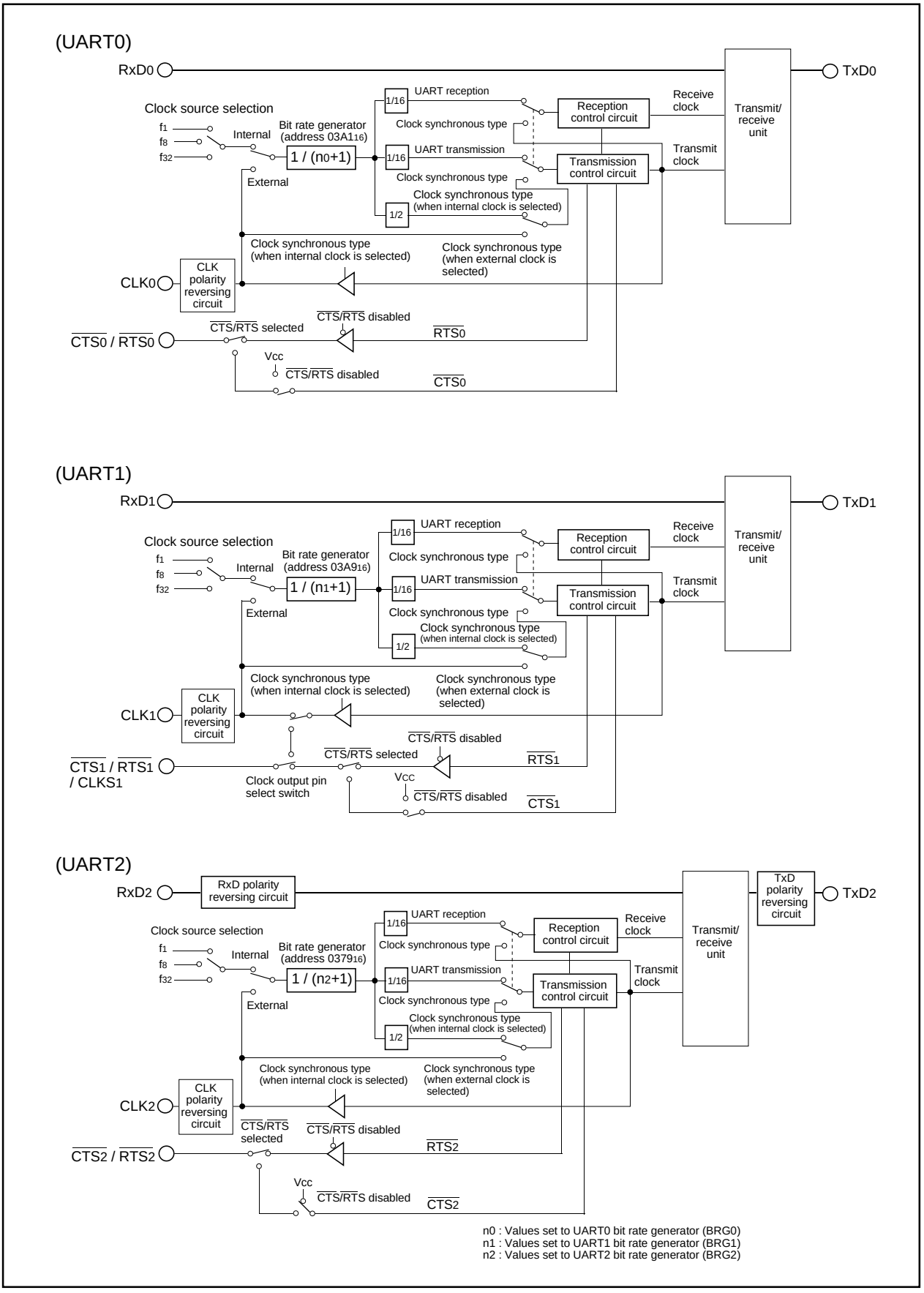



**Figure 2.11.1 Block diagram of UARTi (i = 0 to 2)**

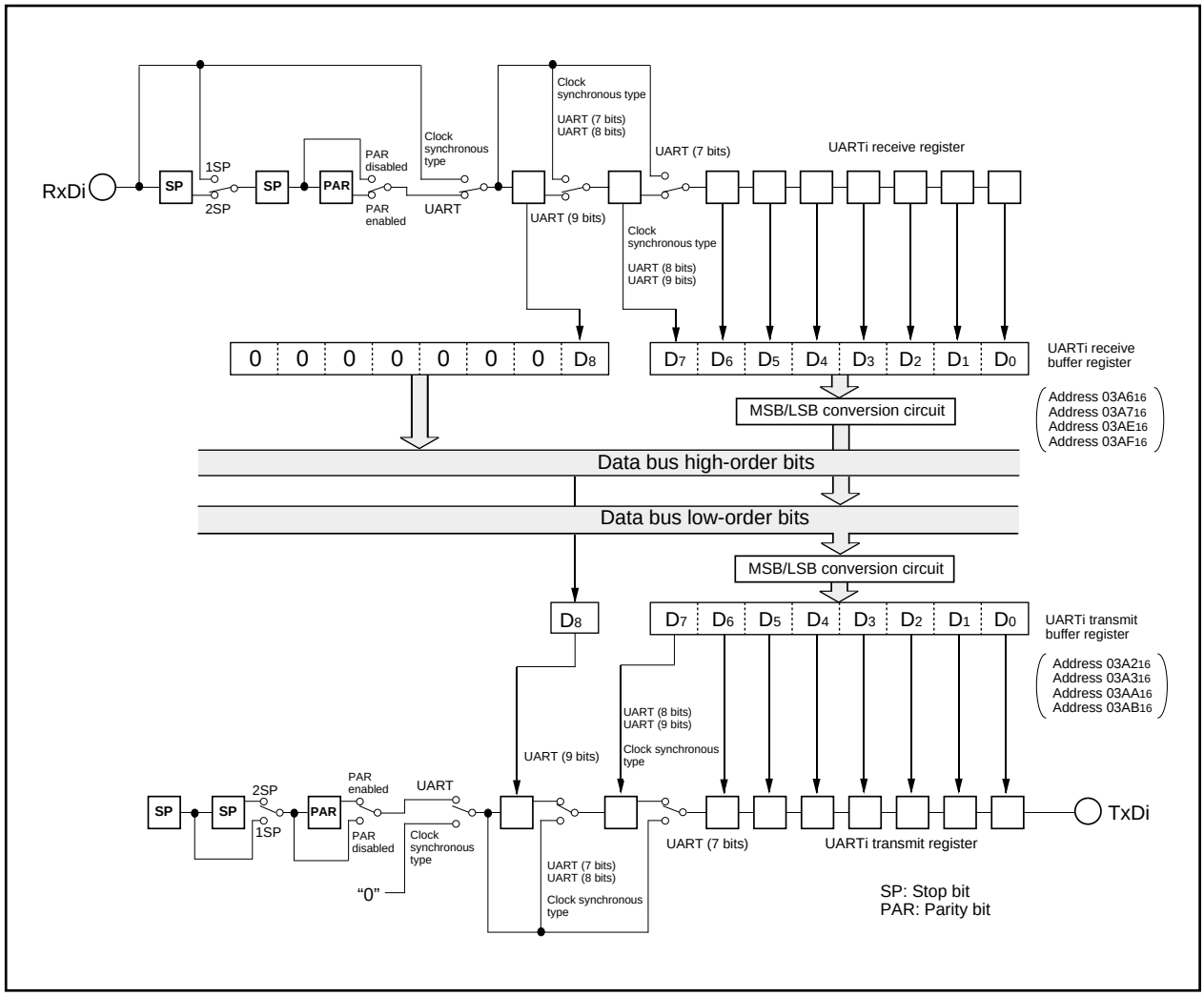



**Figure 2.11.2 Block diagram of UARTi (i = 0, 1) transmit/receive unit**

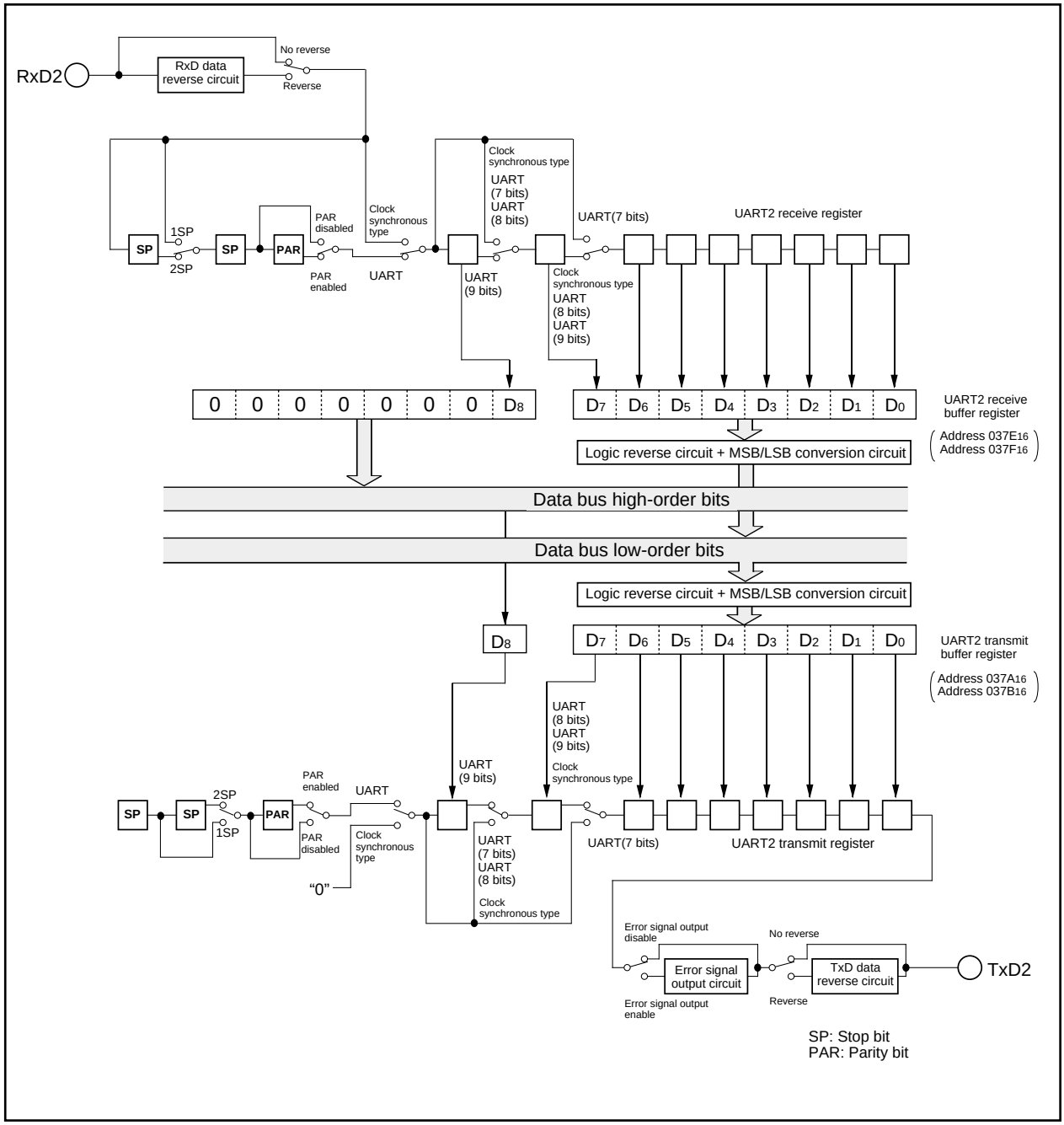
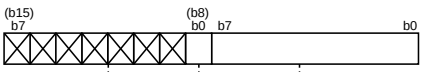



**Figure 2.11.3 Block diagram of UART2 transmit/receive unit**

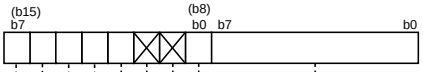
UARTi transmit buffer register




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

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

UARTi receive buffer register

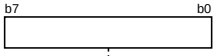



| Symbol | Address | When reset |
|---|---|---|
| U0RB | $03A7_{16}$, $03A6_{16}$ | Indeterminate |
| U1RB | $03AF_{16}$, $03AE_{16}$ | Indeterminate |
| U2RB | $037F_{16}$, $037E_{16}$ | Indeterminate |

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

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

UARTi bit rate generator




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

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

**Figure 2.11.4 UARTi I/O-related registers (1)**

UARTi transmit/receive mode register

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

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

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

UART2 transmit/receive mode register

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

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

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

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

**Figure 2.11.5 UARTil I/O-related registers (2)**

UARTi transmit/receive control register 0


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

| Bit symbol | Bit name | Function (During clock synchronous serial I/O mode) | Function (During UART mode) | R | W |
|---|---|---|---|---|---|
| CLK0 | BRG count source select bit | b1 b0<br>0 0 : $f_1$ is selected<br>0 1 : $f_8$ is selected<br>1 0 : $f_{32}$ is selected<br>1 1 : Inhibited | b1 b0<br>0 0 : $f_1$ is selected<br>0 1 : $f_8$ is selected<br>1 0 : $f_{32}$ is selected<br>1 1 : Inhibited | ○ | ○ |
| CLK1 | | | | ○ | ○ |
| CRS | $\overline{CTS}$/$\overline{RTS}$ function select bit | Valid when bit 4 = "0"<br>0 : $\overline{CTS}$ function is selected (Note 1)<br>1 : RTS function is selected (Note 2) | Valid when bit 4 = "0"<br>0 : $\overline{CTS}$ function is selected (Note 1)<br>1 : $\overline{RTS}$ function is selected (Note 2) | ○ | ○ |
| TXEPT | Transmit register empty flag | 0 : Data present in transmit register (during transmission)<br>1 : No data present in transmit register (transmission completed) | 0 : Data present in transmit register (during transmission)<br>1 : No data present in transmit register (transmission completed) | ○ | × |
| CRD | $\overline{CTS}$/$\overline{RTS}$ disable bit | 0 : $\overline{CTS}$/$\overline{RTS}$ function enabled<br>1 : CTS/RTS function disabled (P6$_0$ and P6$_4$ function as programmable I/O port) | 0 : $\overline{CTS}$/$\overline{RTS}$ function enabled<br>1 : CTS/RTS function disabled (P6$_0$ and P6$_4$ function as programmable I/O port) | ○ | ○ |
| NCH | Data output select bit | 0 : TXDi pin is CMOS output<br>1 : TXDi pin is N-channel open-drain output | 0: TXDi pin is CMOS output<br>1: TXDi pin is N-channel open-drain output | ○ | ○ |
| CKPOL | CLK polarity select bit | 0 : Transmit data is output at falling edge of transfer clock and receive data is input at rising edge<br>1 : Transmit data is output at rising edge of transfer clock and receive data is input at falling edge | Must always be "0" | ○ | ○ |
| UFORM | Transfer format select bit | 0 : LSB first<br>1 : MSB first | Must always be "0" | ○ | ○ |

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

UART2 transmit/receive control register 0

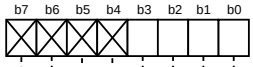

Symbol: U2C0 — Address: 037C$_{16}$ — When reset: 08$_{16}$

| Bit symbol | Bit name | Function (During clock synchronous serial I/O mode) | Function (During UART mode) | R | W |
|---|---|---|---|---|---|
| CLK0 | BRG count source select bit | b1 b0<br>0 0 : $f_1$ is selected<br>0 1 : $f_8$ is selected<br>1 0 : $f_{32}$ is selected<br>1 1 : Inhibited | b1 b0<br>0 0 : $f_1$ is selected<br>0 1 : $f_8$ is selected<br>1 0 : $f_{32}$ is selected<br>1 1 : Inhibited | ○ | ○ |
| CLK1 | | | | ○ | ○ |
| CRS | $\overline{CTS}$/$\overline{RTS}$ function select bit | Valid when bit 4 = "0"<br>0 : $\overline{CTS}$ function is selected (Note 1)<br>1 : RTS function is selected (Note 2) | Valid when bit 4 = "0"<br>0 : $\overline{CTS}$ function is selected (Note 1)<br>1 : $\overline{RTS}$ function is selected (Note 2) | ○ | ○ |
| TXEPT | Transmit register empty flag | 0 : Data present in transmit register (during transmission)<br>1 : No data present in transmit register (transmission completed) | 0 : Data present in transmit register (during transmission)<br>1 : No data present in transmit register (transmission completed) | ○ | × |
| CRD | $\overline{CTS}$/$\overline{RTS}$ disable bit | 0 : $\overline{CTS}$/$\overline{RTS}$ function enabled<br>1 : CTS/RTS function disabled (P7$_3$ functions programmable I/O port) | 0 : $\overline{CTS}$/$\overline{RTS}$ function enabled<br>1 : CTS/RTS function disabled (P7$_3$ functions programmable I/O port) | ○ | ○ |
| Nothing is assigned.<br>In an attempt to write to this bit, write "0". The value, if read, turns out to be "0". | | | | — | — |
| CKPOL | CLK polarity select bit | 0 : Transmit data is output at falling edge of transfer clock and receive data is input at rising edge<br>1 : Transmit data is output at rising edge of transfer clock and receive data is input at falling edge | Must always be "0" | ○ | ○ |
| UFORM | Transfer format select bit (Note 3) | 0 : LSB first<br>1 : MSB first | 0 : LSB first<br>1 : MSB first | ○ | ○ |

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

**Figure 2.11.6 UARTi I/O-related registers (3)**

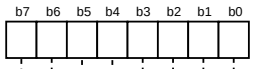
UARTi transmit/receive control register 1




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

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

UART2 transmit/receive control register 1




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

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

**Figure 2.11.7 UARTi I/O-related registers (4)**

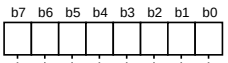
UART transmit/receive control register 2


Symbol: UCON | Address: 03B0₁₆ | When reset: X0000000₂

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

Note: When using multiple pins to output the transfer clock, the following requirements must be met:
• UART1 internal/external clock select bit (bit 3 at address 03A8 ₁₆) = "0".

UART2 special mode register



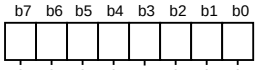

Symbol: U2SMR | Address: 0377₁₆ | When reset: 00₁₆

| Bit symbol | Bit name | Function (During clock synchronous serial I/O mode) | Function (During UART mode) | R | W |
|---|---|---|---|---|---|
| IICM | I²C mode selection bit | 0 : Normal mode<br>1 : I²C mode | Must always be "0" | O | O |
| ABC | Arbitration lost detecting flag control bit | 0 : Update per bit<br>1 : Update per byte | Must always be "0" | O | O |
| BBS | Bus busy flag | 0 : STOP condition detected<br>1 : START condition detected | Must always be "0" | O (Note 1) | O |
| LSYN | SCLL sync output enable bit | 0 : Disabled<br>1 : Enabled | Must always be "0" | O | O |
| ABSCS | Bus collision detect sampling clock select bit | Must always be "0" | 0 : Rising edge of transfer clock<br>1 : Underflow signal of timer A0 | O | O |
| ACSE | Auto clear function select bit of transmit enable bit | Must always be "0" | 0 : No auto clear function<br>1 : Auto clear at occurrence of bus collision | O | O |
| SSS | Transmit start condition select bit | Must always be "0" | 0 : Ordinary<br>1 : Falling edge of RxD2 | O | O |
| SDDS | SDA digital delay selection bit (Notes 2 and 3) | 0 : Analog delay output selection<br>1 : Digital delay output selection | Must always be "0" | O | O |

Notes 1: Nothing but "0" may be written.
2: Do not write "1" except at I²C mode. Must always be "0" at normal mode.
Bit 7 to bit5 (DL2 to DL0 = SDA digital delay value setting bit) of UART2 special mode register 3 (U2SMR3/address 0375 ₁₆) are initialized and become "000" when this bit is "0", analog delay circuit is selected. Reading and writing U2SMR are enable when SDDS = "0" .
3: Delaying ; Only analog delay value when analog delay is selected, and only digital delay value when digital delay is selected.

**Figure 2.11.8 UARTi I/O-related registers (5)**

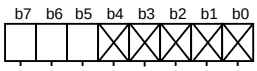
UART2 special mode register 2 ($I^2C$ bus exclusive register)




Symbol U2SMR2 | Address $037_{16}$ | When reset $00_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| IICM2 | $I^2C$ mode selection bit 2 | Refer to Table 2.11.11 | ○ | ○ |
| CSC | Clock-synchronous bit | 0 : Disabled<br>1 : Enabled | ○ | ○ |
| SWC | SCL wait output bit | 0 : Disabled<br>1 : Enabled | ○ | ○ |
| ALS | SDA output stop bit | 0 : Disabled<br>1 : Enabled | ○ | ○ |
| STAC | UART2 initialization bit | 0 : Disabled<br>1 : Enabled | ○ | ○ |
| SWC2 | SCL wait output bit 2 | 0: UART2 clock<br>1: 0 output | ○ | ○ |
| SDHI | SDA output disable bit | 0: Enabled<br>1: Disabled (high impedance) | ○ | ○ |
| SHTC | Start/stop condition control bit | Set this bit to "1" in $I^2C$ mode<br>(refer to Table 2.11.12) | ○ | ○ |

UART2 special mode register 3 ($I^2C$ bus exclusive register)




Symbol U2SMR3 | Address $037_{16}$ | When reset Indeterminate
(initializing value is "$00_{16}$" at SDDS = "1")

| Bit symbol | Bit name | Function ($I^2C$ bus exclusive) | R | W |
|---|---|---|---|---|
| Nothing is assigned.<br>In an attempt to write to this bit, write “0”. The value, if read, turns out to be “0”.<br>“0” is read out when SDDS = 1. | | | — | — |
| DL0 | SDA digital delay value setting bit | b7 b6 b5<br>0 0 0 : Analog delay<br>0 0 1 : 2 cycle of 1/f (Xin)(Digital delay)<br>0 1 0 : 3 cycle of 1/f (Xin)(Digital delay)<br>0 1 1 : 4 cycle of 1/f (Xin)(Digital delay)<br>1 0 0 : 5 cycle of 1/f (Xin)(Digital delay)<br>1 0 1 : 6 cycle of 1/f (Xin)(Digital delay)<br>1 1 0 : 7 cycle of 1/f (Xin)(Digital delay)<br>1 1 1 : 8 cycle of 1/f (Xin)(Digital delay) | ○ | ○ |
| DL1 | | | ○ | ○ |
| DL2 | | | ○ | ○ |

Notes 1: Reading and writing is possible when bit7 (SDDS = SDA digital delay selection bit) of UART2 special mode register (U2SMR/address $037_{16}$) is "1". When set SDDS = "1" and read out initialized value of UART2 special mode register 3(U2SMR3), this value is "$00_{16}$".When set SDDS = "1" and write to UART2 special mode register 3(U2SMR3), set "0" to bit 0 to bit 4. When SDDS = "0", writing is enable. When read out, this value is indeterminate.
2: When SDDS = "0" , this bit is initialized and become "000", selected analog delay circuit. This bit is become "000" after end reset released, and selected analog delay circuit. Reading out is possible when only SDDS = "1". when SDDS = "0", value which was read out is indeterminate.
3: Delaying ; Only analog delay value when analog delay is selected, and only digital delay value when digital delay is selected.
4: Delay level depends on SCL pin and SDA pin. And, when use external clock, delay is increase around 100ns. So test first, and use this.

**Figure 2.11.9 UARTi -related registers (6)**

### 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. Figur 2.11.10 shows the UARTi transmit/receive mode register.

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

| Item | Specification |
|---|---|
| Transfer data format | • Transfer data length: 8 bits |
| Transfer clock | • When internal clock is selected (bit 3 at addresses $03A0_{16}$, $03A8_{16}$, $0378_{16}$ = "0") : fi/ 2(n+1) (Note 1) fi = $f_1$, $f_8$, $f_{32}$<br>• When external clock is selected (bit 3 at addresses $03A0_{16}$, $03A8_{16}$, $0378_{16}$ = "1") : Input from CLKi pin |
| Transmission/reception control | • $\overline{CTS}$ function/$\overline{RTS}$ function/$\overline{CTS}$, $\overline{RTS}$ function chosen to be invalid |
| Transmission start condition | • To start transmission, the following requirements must be met:<br>– Transmit enable bit (bit 0 at addresses $03A5_{16}$, $03AD_{16}$, $037D_{16}$) = "1"<br>– Transmit buffer empty flag (bit 1 at addresses $03A5_{16}$, $03AD_{16}$, $037D_{16}$) = "0"<br>– When $\overline{CTS}$ function selected, $\overline{CTS}$ input level = "L"<br>• Furthermore, if external clock is selected, the following requirements must also be met:<br>– CLKi polarity select bit (bit 6 at addresses $03A4_{16}$, $03AC_{16}$, $037C_{16}$) = "0": CLKi input level = "H"<br>– CLKi polarity select bit (bit 6 at addresses $03A4_{16}$, $03AC_{16}$, $037C_{16}$) = "1": CLKi input level = "L" |
| Reception start condition | • To start reception, the following requirements must be met:<br>– Receive enable bit (bit 2 at addresses $03A5_{16}$, $03AD_{16}$, $037D_{16}$) = "1"<br>– Transmit enable bit (bit 0 at addresses $03A5_{16}$, $03AD_{16}$, $037D_{16}$) = "1"<br>– Transmit buffer empty flag (bit 1 at addresses $03A5_{16}$, $03AD_{16}$, $037D_{16}$) = "0"<br>• Furthermore, if external clock is selected, the following requirements must also be met:<br>– CLKi polarity select bit (bit 6 at addresses $03A4_{16}$, $03AC_{16}$, $037C_{16}$) = "0": CLKi input level = "H"<br>– CLKi polarity select bit (bit 6 at addresses $03A4_{16}$, $03AC_{16}$, $037C_{16}$) = "1": CLKi input level = "L" |
| Interrupt request generation timing | • When transmitting<br>– Transmit interrupt cause select bit (bits 0, 1 at address $03B0_{16}$, bit 4 at address $037D_{16}$) = "0": Interrupts requested when data transfer from UARTi transfer buffer register to UARTi transmit register is completed<br>– Transmit interrupt cause select bit (bits 0, 1 at address $03B0_{16}$, bit 4 at address $037D_{16}$) = "1": Interrupts requested when data transmission from UARTi transfer register is completed<br>• When receiving<br>– Interrupts requested when data transfer from UARTi receive register to UARTi receive buffer register is completed |
| Error detection | • Overrun error (Note 2)<br>This error occurs when the next data is ready before contents of UARTi receive buffer register are read out |

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

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

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

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

UARTi transmit/receive mode registers

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

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

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

UART2 transmit/receive mode register

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

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

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

Note: Usually set to “0”.

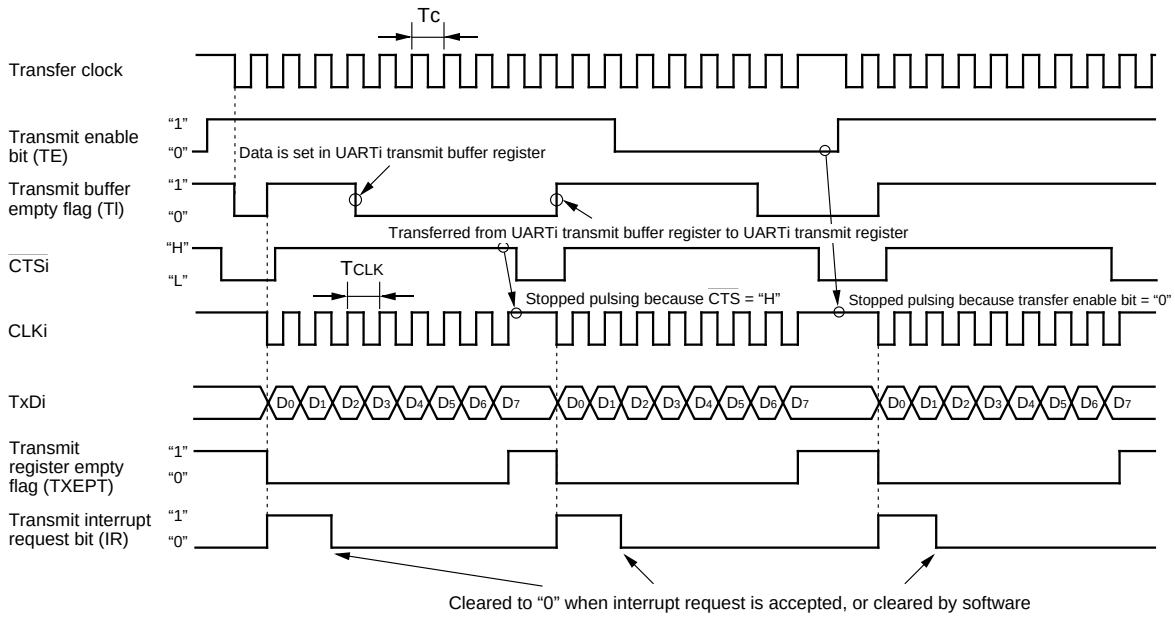
**Figure 2.11.10 UARTi transmit/receive mode register in clock synchronous serial I/O mode**

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

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

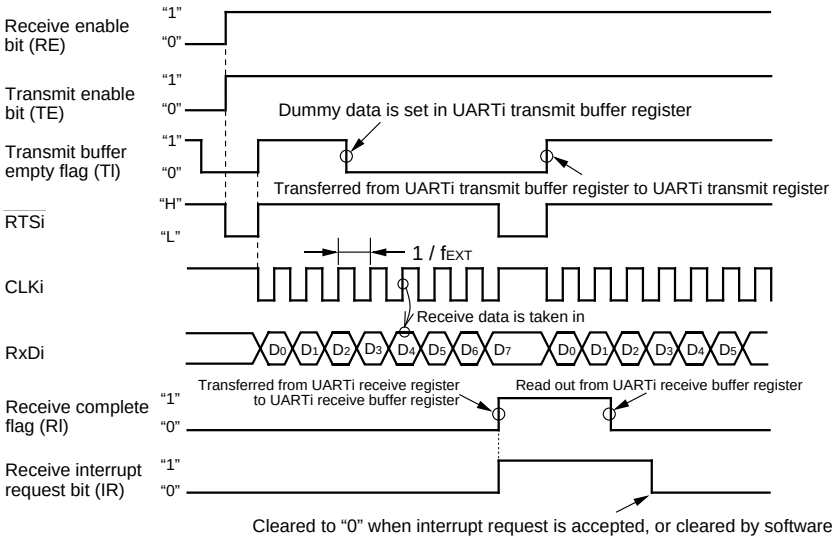



Shown in ( ) are bit symbols.

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".

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

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




Shown in ( ) are bit symbols.

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

fEXT: frequency of external clock

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

**Figure 2.11.11 Typical transmit/receive timings in clock synchronous serial I/O mode**

**(1) Polarity select function**

As shown in Figure 2.11.12 the CLK polarity select bit (bit 6 at addresses 03A4$_{16}$, 03AC$_{16}$, 037C$_{16}$) allows selection of the polarity of the transfer clock.

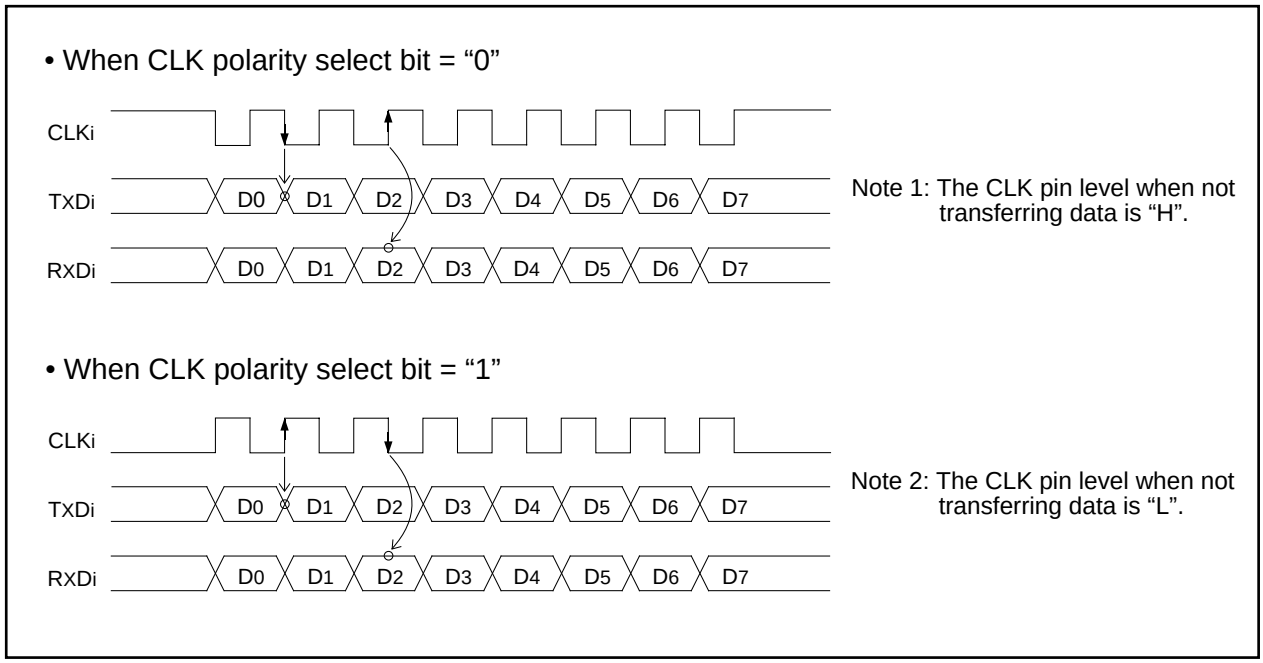



**Figure 2.11.12 Polarity of transfer clock**

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

As shown in Figure 2.11.13, when the transfer format select bit (bit 7 at addresses 03A4$_{16}$, 03AC$_{16}$, 037C$_{16}$) = “0”, the transfer format is “LSB first”; when the bit = “1”, the transfer format is “MSB first”.

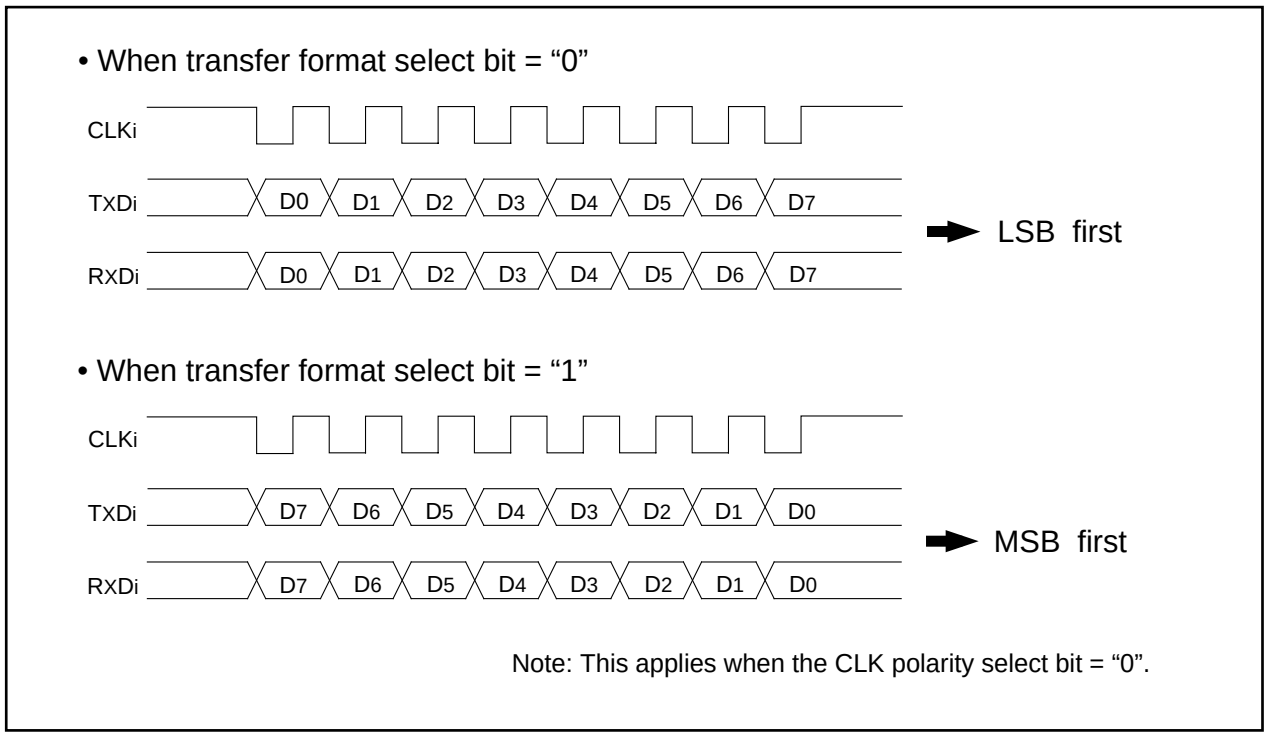



**Figure 2.11.13 Transfer format**

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

This function allows the setting two transfer clock output pins and choosing one of the two to output a clock by using the CLK and CLKS select bit (bits 4 and 5 at address $03B0_{16}$). (See Figure 2.11.14) The multiple pins function is valid only when the internal clock is selected for UART1. Note that when this function is selected, UART1 $\overline{CTS}/\overline{RTS}$ function cannot be used.

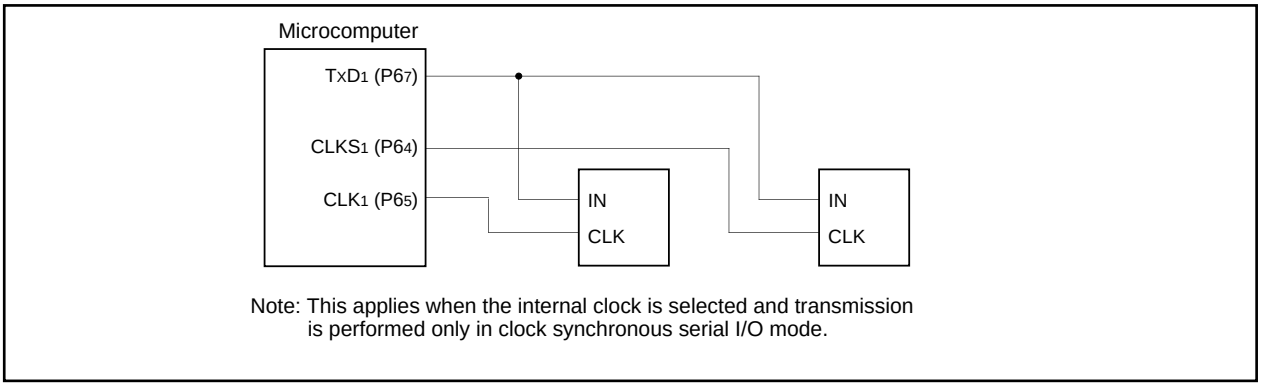



**Figure 2.11.14 The transfer clock output from the multiple pins function usage**

**(4) Continuous receive mode**

If the continuous receive mode enable bit (bits 2 and 3 at address $03B0_{16}$, bit 5 at address $037D_{16}$) is set to "1", the unit is placed in continuous receive mode. In this mode, when the receive buffer register is read out, the unit simultaneously goes to a receive enable state without having to set dummy data to the transmit buffer register back again.

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

When the data logic select bit (bit6 at address $037D_{16}$) = "1", and writing to transmit buffer register or reading from receive buffer register, data is reversed. Figure 2.11.15 shows the example of serial data logic switch timing.

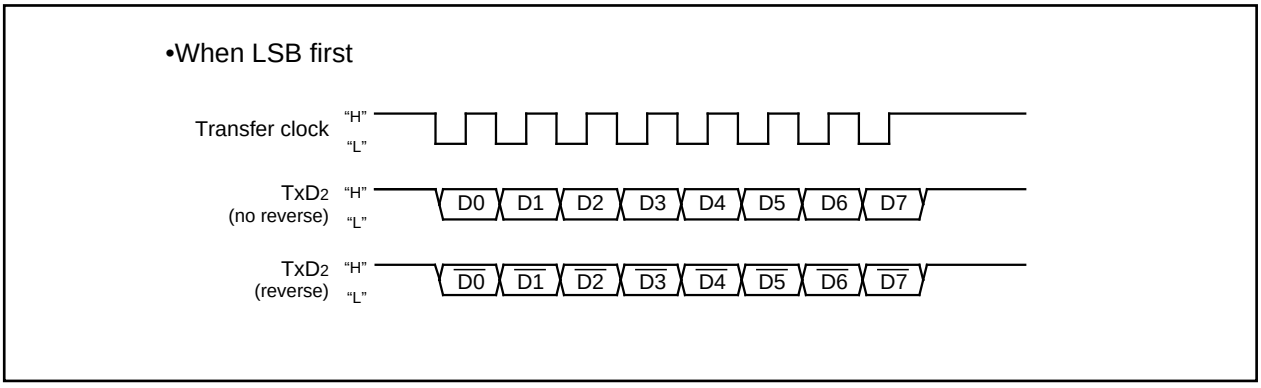



**Figure 1.11.15 Serial data logic switch timing**

### 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 2.11.16 shows the UARTi transmit/receive mode register.

**Table 2.11.5 Specifications of UART Mode (1)**

| Item | Specification |
|---|---|
| Transfer data format | • Character bit (transfer data): 7 bits, 8 bits, or 9 bits as selected<br>• Start bit: 1 bit<br>• Parity bit: Odd, even, or nothing as selected<br>• Stop bit: 1 bit or 2 bits as selected |
| Transfer clock | • When internal clock is selected (bit 3 at addresses $03A0_{16}$ ,$03A8_{16}$ ,$0378_{16}$ = "0") : fi/16(n+1) (Note 1) fi = $f_1$, $f_8$, $f_{32}$<br>• When external clock is selected (bit 3 at addresses $03A0_{16}$ and $03A8_{16}$ = "1") : $f_{EXT}$/16(n+1) (Note 1) (Note 2) (Do not set external clock for UART2) |
| Transmission/reception control | • $\overline{CTS}$ function/$\overline{RTS}$ function/$\overline{CTS}$, $\overline{RTS}$ function chosen to be invalid |
| Transmission start condition | • To start transmission, the following requirements must be met:<br>- Transmit enable bit (bit 0 at addresses $03A5_{16}$, $03AD_{16}$, $037D_{16}$) = "1"<br>- Transmit buffer empty flag (bit 1 at addresses $03A5_{16}$, $03AD_{16}$, $037D_{16}$) = "0"<br>- When $\overline{CTS}$ function selected, $\overline{CTS}$ input level = "L" |
| Reception start condition | • To start reception, the following requirements must be met:<br>- Receive enable bit (bit 2 at addresses $03A5_{16}$, $03AD_{16}$, $037D_{16}$) = "1"<br>- Start bit detection |
| Interrupt request generation timing | • When transmitting<br>- Transmit interrupt cause select bits (bits 0,1 at address $03B0_{16}$, bit4 at address $037D_{16}$) = "0": Interrupts requested when data transfer from UARTi transfer buffer register to UARTi transmit register is completed<br>- Transmit interrupt cause select bits (bits 0, 1 at address $03B0_{16}$, bit4 at address $037D_{16}$) = "1": Interrupts requested when data transmission from UARTi transfer register is completed<br>• When receiving<br>- Interrupts requested when data transfer from UARTi receive register to UARTi receive buffer register is completed |
| Error detection | • Overrun error (Note 3)<br>This error occurs when the next data is ready before contents of UARTi receive buffer register are read out<br>• Framing error<br>This error occurs when the number of stop bits set is not detected<br>• Parity error<br>This error occurs when if parity is enabled, the number of 1's in parity and character bits does not match the number of 1's set<br>• Error sum flag<br>This flag is set (= 1) when any of the overrun, framing, and parity errors is encountered |

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

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

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

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

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

UARTi transmit / receive mode registers

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

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

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

UART2 transmit / receive mode register

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

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

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

Note: Usually set to "0".

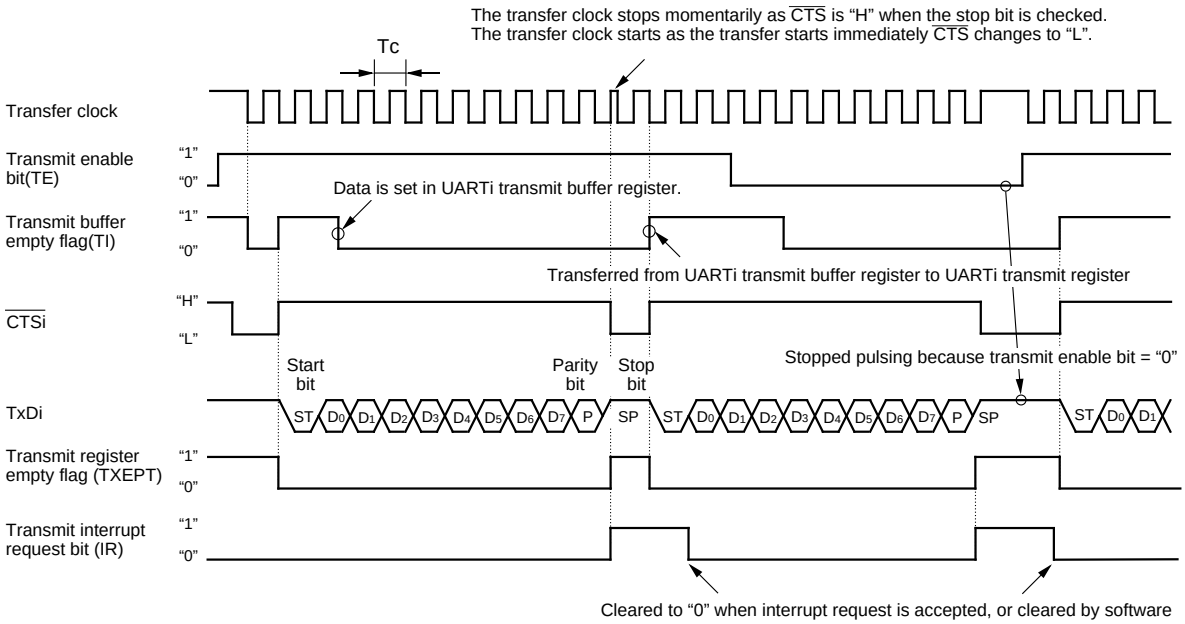
**Figure 2.11.16 UARTi transmit/receive mode register in UART mode**

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 (P6$_3$, P6$_7$, P7$_0$) | Serial data output | |
| RxDi (P6$_2$, P6$_6$, P7$_1$) | Serial data input | Port P6$_2$, P6$_6$ and P7$_1$ direction register (bits 2 and 6 at address 03EE$_{16}$, bit 1 at address 03EF$_{16}$)= “0” (Can be used as an input port when performing transmission only) |
| CLKi (P6$_1$, P6$_5$, P7$_2$) | Programmable I/O port | Internal/external clock select bit (bit 3 at address 03A0$_{16}$, 03A8$_{16}$, 037$8_{16}$) = “0” |
| | Transfer clock input | Internal/external clock select bit (bit 3 at address 03A0$_{16}$, 03A8$_{16}$) = “1” Port P6$_1$, P6$_5$ direction register (bits 1 and 5 at address 03EE$_{16}$) = “0” (Do not set external clock for UART2) |
| $\overline{\text{CTS}}$i/$\overline{\text{RTS}}$i (P6$_0$, P6$_4$, P7$_3$) | $\overline{\text{CTS}}$ input | $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit (bit 4 at address 03A4$_{16}$, 03AC$_{16}$, 037C$_{16}$) =“0” $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ function select bit (bit 2 at address 03A4$_{16}$, 03AC$_{16}$, 037C$_{16}$) = “0” Port P6$_0$, P6$_4$ and P7$_3$ direction register (bits 0 and 4 at address 03EE$_{16}$, bit 3 at address 03EF$_{16}$) = “0” |
| | $\overline{\text{RTS}}$ output | $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit (bit 4 at address 03A4$_{16}$, 03AC$_{16}$, 037C$_{16}$) = “0” $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ function select bit (bit 2 at address 03A4$_{16}$, 03AC$_{16}$, 037C$_{16}$) = “1” |
| | Programmable I/O port | $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit (bit 4 at address 03A4$_{16}$, 03AC$_{16}$, 037C$_{16}$) = “1” |

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

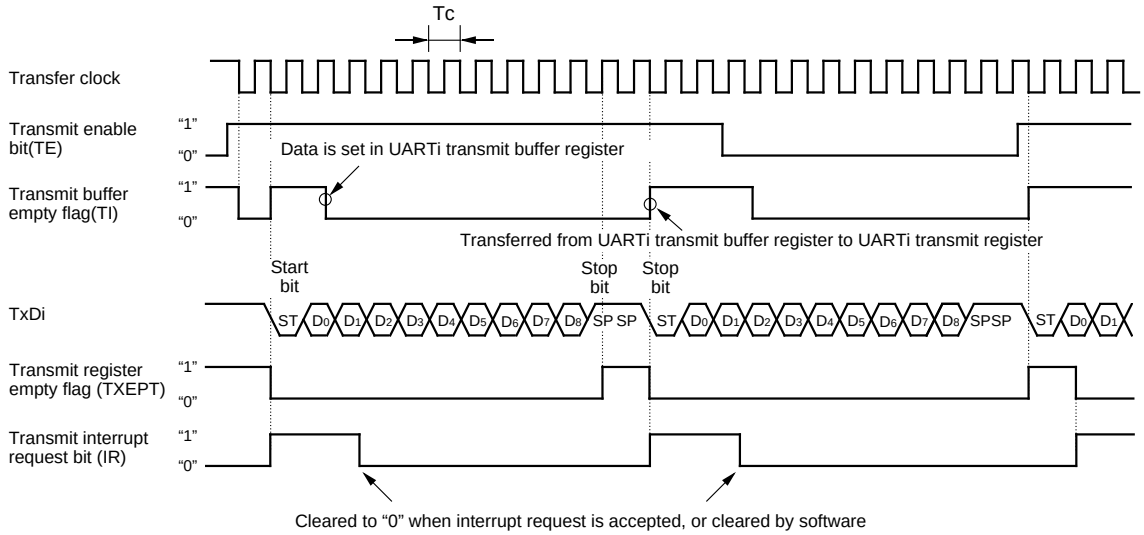



Shown in ( ) are bit symbols.

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

Tc = 16 (n + 1) / fi or 16 (n + 1) / $f_{EXT}$
fi : frequency of BRGi count source ($f_1$, $f_8$, $f_{32}$)
$f_{EXT}$ : frequency of BRGi count source (external clock)
n : value set to BRGi

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



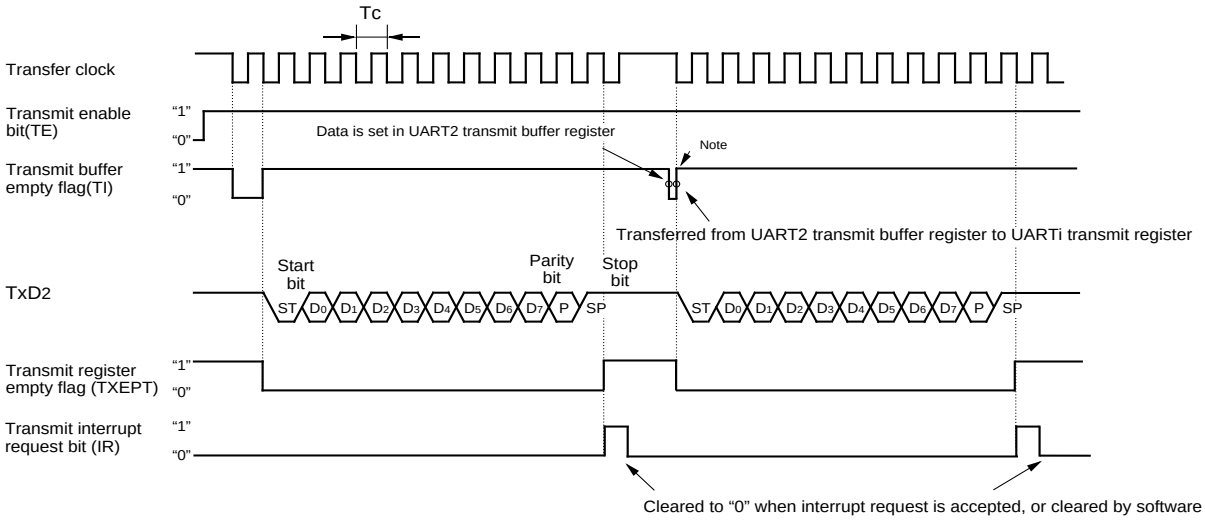

Shown in ( ) are bit symbols.

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

Tc = 16 (n + 1) / fi or 16 (n + 1) / $f_{EXT}$
fi : frequency of BRGi count source ($f_1$, $f_8$, $f_{32}$)
$f_{EXT}$ : frequency of BRGi count source (external clock)
n : value set to BRGi

**Figure 2.11.17 Typical transmit timings in UART mode(UART0,UART1)**

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




Shown in ( ) are bit symbols.

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

Tc = 16 (n + 1) / fi
fi : frequency of BRG2 count source ($f_1$, $f_8$, $f_{32}$)
n : value set to BRG2

Note: The transmit is started with overflow timing of BRG after having written in a value at the transmit buffer in the above timing.

**Figure 2.11.18 Typical transmit timings in UART mode(UART2)**

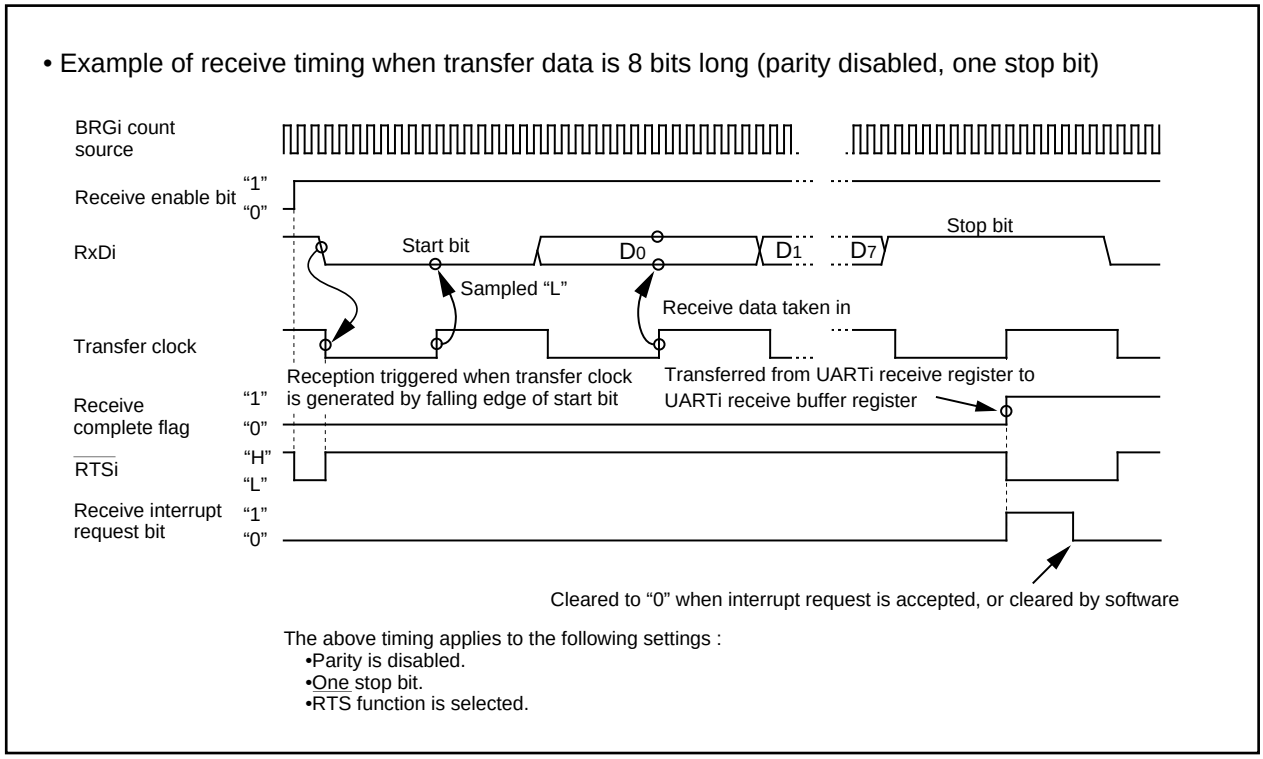



**Figure 2.11.19 Typical receive timing in UART mode**

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

This mode is used to transfer data between specific microcomputers among multiple microcomputers connected using UARTi. The sleep mode is selected when the sleep select bit (bit 7 at addresses $03A0_{16}$, $03A8_{16}$) is set to "1" during reception. In this mode, the unit performs receive operation when the MSB of the received data = "1" and does not perform receive operation when the MSB = "0".

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

When the data logic select bit (bit 6 of address $037D_{16}$) is assigned 1, data is inverted in writing to the transmission buffer register or reading the reception buffer register. Figure 2.11.20 shows the example of timing for switching serial data logic.

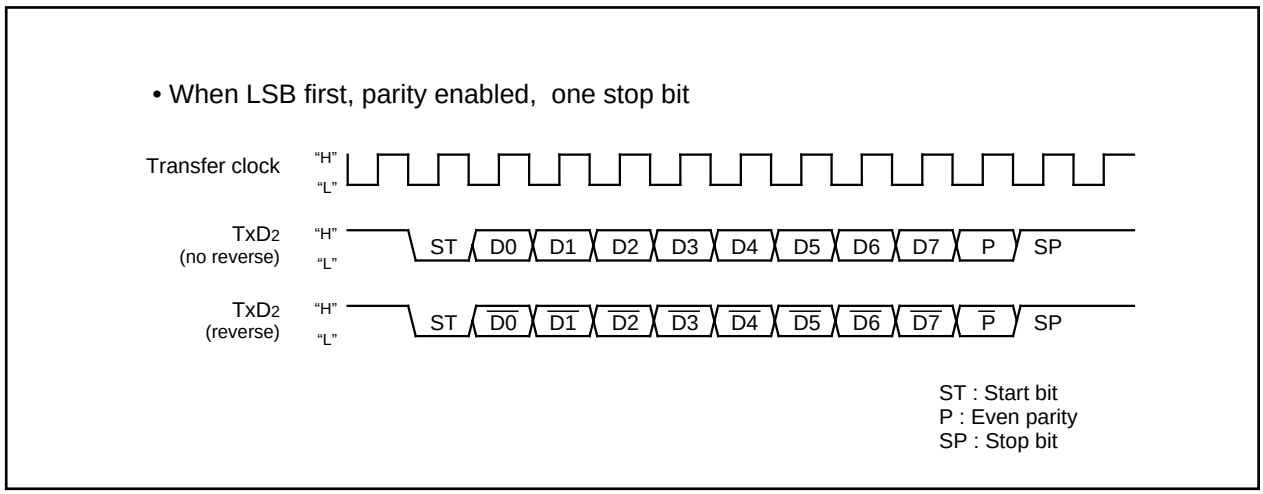



**Figure 2.11.20 Timing for switching serial data logic**

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

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

**(4) Bus collision detection function (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 2.11.21 shows the example of detection timing of a buss collision (in UART mode).

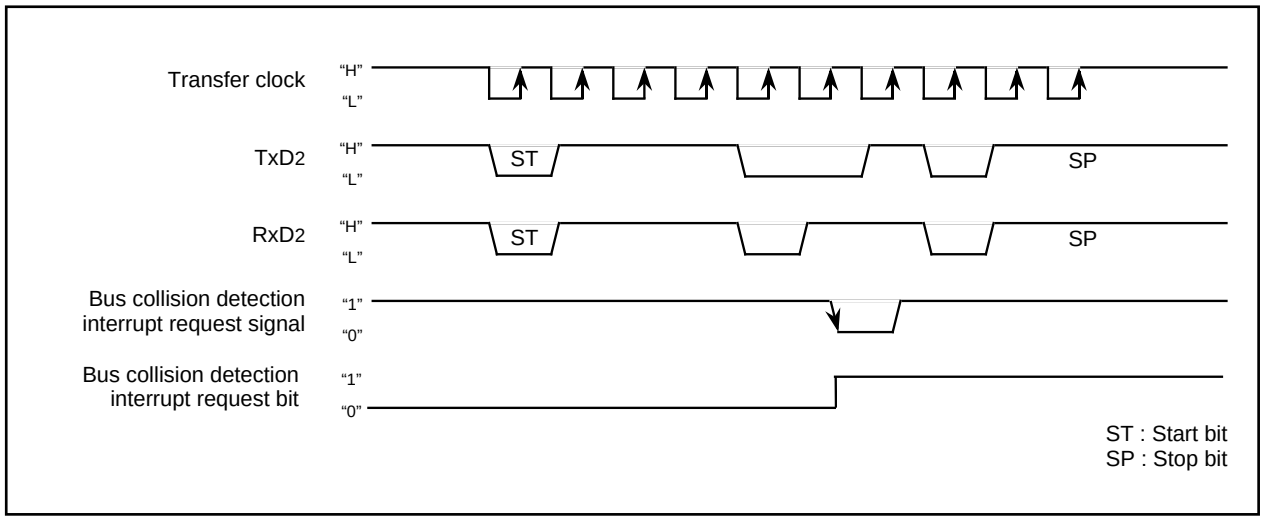



**Figure 2.11.21 Detection timing of a bus collision (in UART mode)**

### 2.11.4 Clock-asynchronous serial I/O mode (compliant with the SIM interface)

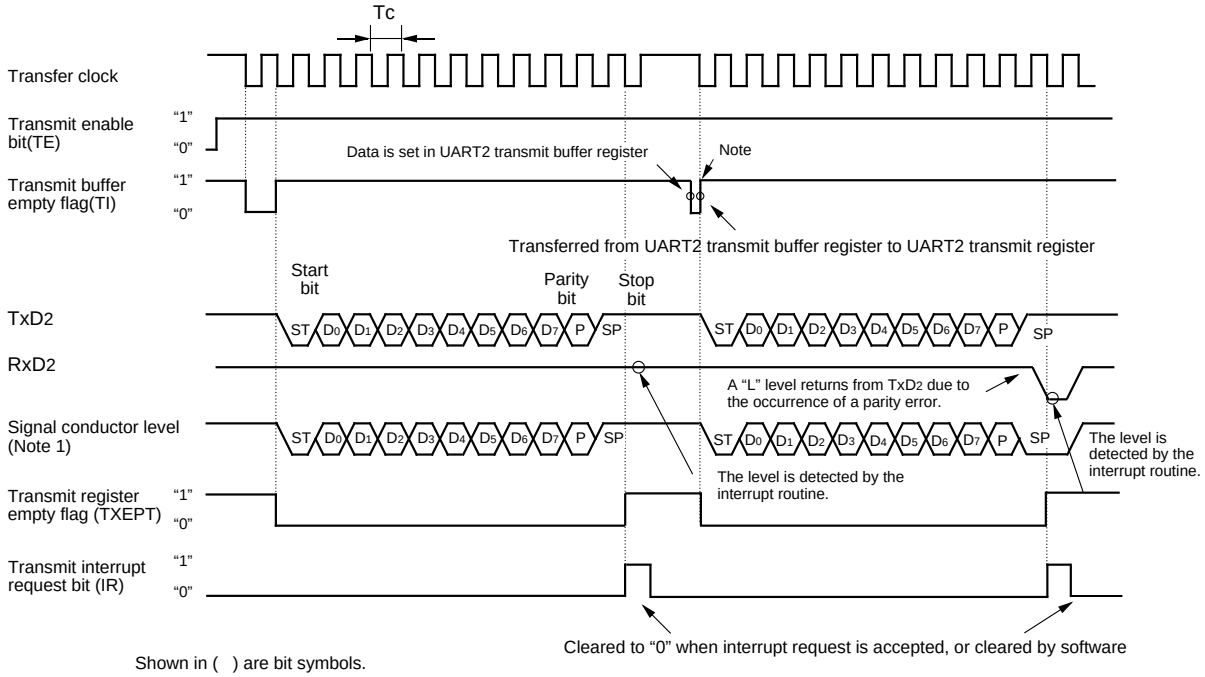
The SIM interface is used for connecting the microcomputer with a memory card or the like; adding some extra settings in UART2 clock-asynchronous serial I/O mode allows the user to effect this function. Table 2.11.8 shows the specifications of clock-asynchronous serial I/O mode (compliant with the SIM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

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

| Item | Specification |
|---|---|
| Transfer data format | • Transfer data 8-bit UART mode (bit 2 through bit 0 of address $0378_{16}$ = “$101_2$”)<br>• One stop bit (bit 4 of address $0378_{16}$ = “0”)<br>• With the direct format chosen<br>Set parity to “even” (bit 5 and bit 6 of address $0378_{16}$ = “1” and “1” respectively)<br>Set data logic to “direct” (bit 6 of address $037D_{16}$ = “0”).<br>Set transfer format to LSB (bit 7 of address $037C_{16}$ = “0”).<br>• With the inverse format chosen<br>Set parity to “odd” (bit 5 and bit 6 of address $0378_{16}$ = “0” and “1” respectively)<br>Set data logic to “inverse” (bit 6 of address $037D_{16}$ = “1”)<br>Set transfer format to MSB (bit 7 of address $037C_{16}$ = “1”) |
| Transfer clock | • With the internal clock chosen (bit 3 of address $0378_{16}$ = “0”) : fi / 16 (n + 1) (Note 1) : fi=$f_1$, $f_8$, $f_{32}$<br>(Do not set external clock) |
| Transmission / reception control | • Disable the $\overline{CTS}$ and $\overline{RTS}$ function (bit 4 of address $037C_{16}$ = “1”) |
| Other settings | • The sleep mode select function is not available for UART2<br>• Set transmission interrupt factor to “transmission completed” (bit 4 of address $037D_{16}$ = “1”) |
| Transmission start condition | • To start transmission, the following requirements must be met:<br>- Transmit enable bit (bit 0 of address $037D_{16}$) = “1”<br>- Transmit buffer empty flag (bit 1 of address $037D_{16}$) = “0” |
| Reception start condition | • To start reception, the following requirements must be met:<br>- Reception enable bit (bit 2 of address $037D_{16}$) = “1”<br>- Detection of a start bit |
| Interrupt request generation timing | • When transmitting<br>When data transmission from the UART2 transfer register is completed (bit 4 of address $037D_{16}$ = “1”)<br>• When receiving<br>When data transfer from the UART2 receive register to the UART2 receive buffer register is completed |
| Error detection | • Overrun error (see the specifications of clock-asynchronous serial I/O) (Note 2)<br>• Framing error (see the specifications of clock-asynchronous serial I/O)<br>• Parity error (see the specifications of clock-asynchronous serial I/O)<br>- On the reception side, an “L” level is output from the $T_XD_2$ pin by use of the parity error signal output function (bit 7 of address $037D_{16}$ = “1”) when a parity error is detected<br>- On the transmission side, a parity error is detected by the level of input to the $R_XD_2$ pin when a transmission interrupt occurs<br>• The error sum flag (see the specifications of clock-asynchronous serial I/O) |

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

Note 2: If an overrun error occurs, the UART2 receive buffer will have the next data written in. Note also that the UARTi receive interrupt request bit is not set to “1”.

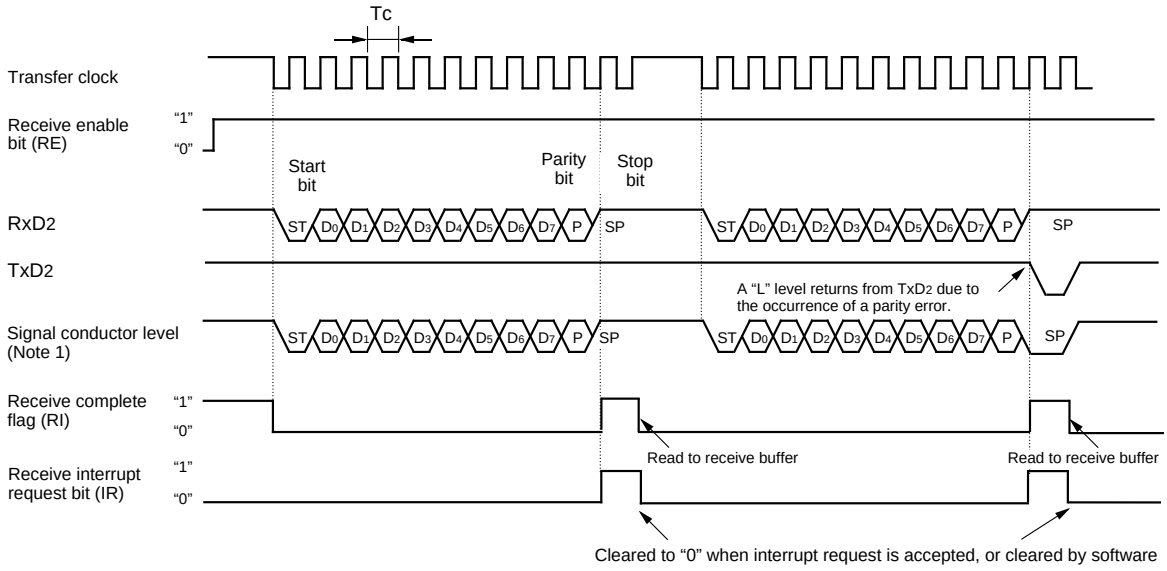



Shown in ( ) are bit symbols.

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

Tc = 16 (n + 1) / fi
fi : frequency of BRG2 count source (f1, f8, f32)
n : value set to BRG2

Note: The transmit is started with overflow timing of BRG after having written in a value at the transmit buffer in the above timing.




Shown in ( ) are bit symbols.

The above timing applies to the following settings :
- Parity is enabled.
- One stop bit.
- Transmit interrupt cause select bit = "0".

Tc = 16 (n + 1) / fi
fi : frequency of BRG2 count source (f1, f8, f32)
n : value set to BRG2

Note: Equal in waveform because TxD2 and RxD2 are connected.

**Figure 2.11.22 Typical transmit/receive timing in UART mode (compliant with the SIM interface)**

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

With the error signal output enable bit (bit 7 of address 037D16) assigned “1”, you can output an “L” level from the TxD2 pin when a parity error is detected. In step with this function, the generation timing of a transmission completion interrupt changes to the detection timing of a parity error signal. Figure 2.11.23 shows the output timing of the parity error signal.

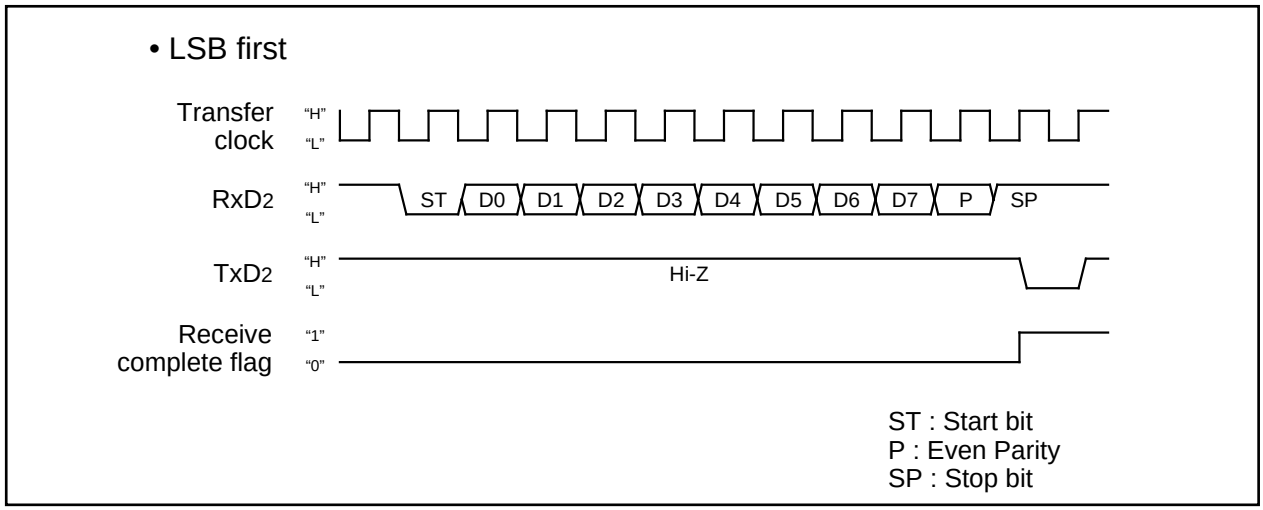



**Figure 2.11.23 Output timing of the parity error signal**

**(2) Direct format/inverse format**

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

Figure 2.11.24 shows the SIM interface format.

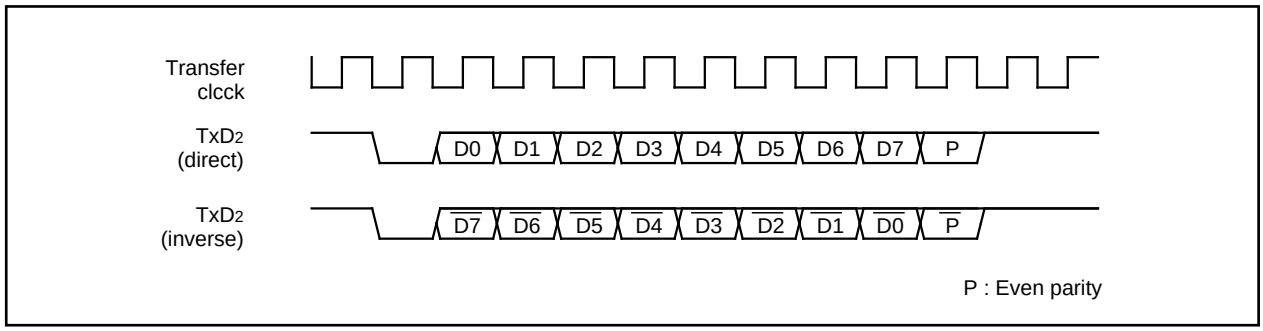



**Figure 2.11.24 SIM interface format**

Figure 2.11.25 shows the example of connecting the SIM interface. Connect TxD2 and RxD2 and apply pull-up.

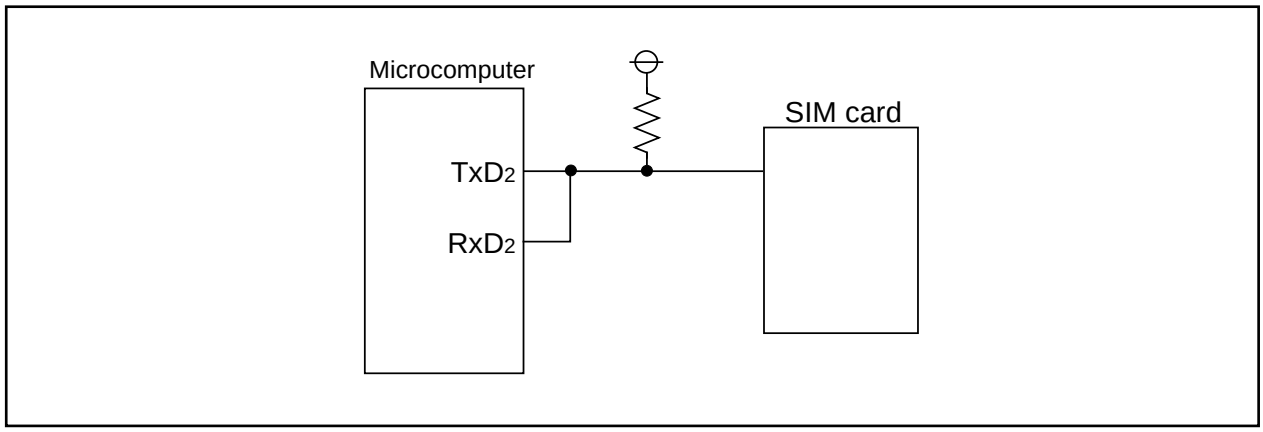



**Figure 2.11.25 Connecting the SIM interface**

## 2.11.5 UART2 Special Mode Register

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

Figure 2.11.26 shows the UART2 special mode register.

In the first place, the control bits related to the $I^2C$ bus(simplified $I^2C$ bus) interface are explained.

Bit 0 of the UART special mode register (0377$_{16}$) is used as the $I^2C$ mode selection bit.

Setting “1” in the $I^2C$ mode select bit (bit 0) goes the circuit to achieve the $I^2C$ bus interface effective.

Since this function uses clock-synchronous serial I/O mode, set this bit to “0” in UART mode.

UART2 special mode register

b7 b6 b5 b4 b3 b2 b1 b0

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

| Bit symbol | Bit name | Function (During clock synchronous serial I/O mode) | Function (During UART mode) | R | W |
|---|---|---|---|---|---|
| IICM | $I^2C$ mode selection bit | 0 : Normal mode<br>1 : $I^2C$ mode | Must always be “0” | ○ | ○ |
| ABC | Arbitration lost detecting flag control bit | 0 : Update per bit<br>1 : Update per byte | Must always be “0” | ○ | ○ |
| BBS | Bus busy flag | 0 : STOP condition detected<br>1 : START condition detected | Must always be “0” | ○ | ○ (Note 1) |
| LSYN | SCLL sync output enable bit | 0 : Disabled<br>1 : Enabled | Must always be “0” | ○ | ○ |
| ABSCS | Bus collision detect sampling clock select bit | Must always be “0” | 0 : Rising edge of transfer clock<br>1 : Underflow signal of timer A0 | ○ | ○ |
| ACSE | Auto clear function select bit of transmit enable bit | Must always be “0” | 0 : No auto clear function<br>1 : Auto clear at occurrence of bus collision | ○ | ○ |
| SSS | Transmit start condition select bit | Must always be “0” | 0 : Ordinary<br>1 : Falling edge of RxD2 | ○ | ○ |
| SDDS | SDA digital delay select bit (Notes 2 and 3) | 0 : Selects analog delay output<br>1 : Selects digital delay output (Must always be “0” except at $I^2C$ mode) | Must always be “0” | ○ | ○ |

Notes 1: Nothing but "0" may be written.
2: Do not write "1" except at $I^2C$ mode. Must always be “0” at normal mode.
Bit 7 to bit5 (DL2 to DL0 = SDA digital delay value setting bit) of UART2 special mode register 3 (U2SMR3/address 0375 $_{16}$) are initialized and become “000” when this bit is "0", analog delay circuit is selected. Reading and writing U2SMR are enable when SDDS = "0" .
3: Delaying ; Only analog delay value when analog delay is selected, and only digital delay value when digital delay is selected.

UART2 special mode register 3 ($I^2C$ bus exclusive register)

b7 b6 b5 b4 b3 b2 b1 b0

Symbol: U2SMR3 | Address: 0375$_{16}$ | When reset: Indeterminate (initializing value is "00 $_{16}$" at SDDS = "1")

| Bit symbol | Bit name | Function ($I^2C$ bus exclusive) | R | W |
|---|---|---|---|---|
| Nothing is assigned. In an attempt to write to this bit, write “0”. The value, if read, turns out to be “0”. “0” is read out when SDDS = 1. | | | ○ | ○ |
| DL0 | SDA digital delay value set bit | b7 b6 b5<br>0 0 0 : Selects analog delay<br>0 0 1 : 2 cycle of 1/f (Xin)(Digital delay)<br>0 1 0 : 3 cycle of 1/f (Xin)(Digital delay)<br>0 1 1 : 4 cycle of 1/f (Xin)(Digital delay)<br>1 0 0 : 5 cycle of 1/f (Xin)(Digital delay)<br>1 0 1 : 6 cycle of 1/f (Xin)(Digital delay)<br>1 1 0 : 7 cycle of 1/f (Xin)(Digital delay)<br>1 1 1 : 8 cycle of 1/f (Xin)(Digital delay) | ○ | ○ |
| DL1 | | | ○ | ○ |
| DL2 | | | ○ | ○ |

Notes 1: Reading and writing is possible when bit7 (SDDS = SDA digital delay selection bit) of UART2 special mode register (U2SMR/address 0377 $_{16}$) is "1". When set SDDS = "1" and read out initialized value of UART2 special mode register 3(U2SMR3), this value is "00 $_{16}$".When set SDDS = "1" and write to UART2 special mode register 3(U2SMR3), set "0" to bit 0 to bit 4. When SDDS = "0", writing is enable. When read out, this value is indeterminate.
2: When SDDS = "0" , this bit is initialized and become "000", selected analog delay circuit. This bit is become "000" after end reset released, and selected analog delay circuit. Reading out is possible when only SDDS = "1". when SDDS = "0", value which was read out is indeterminate.
3: Delaying ; Only analog delay value when analog delay is selected, and only digital delay value when digital delay is selected.
4: Delay level depends on SCL pin and SDA pin. And, when use external clock, delay is increase around 100ns. So test first, and use this.

**Figure 2.11.26 UART2 special mode register**

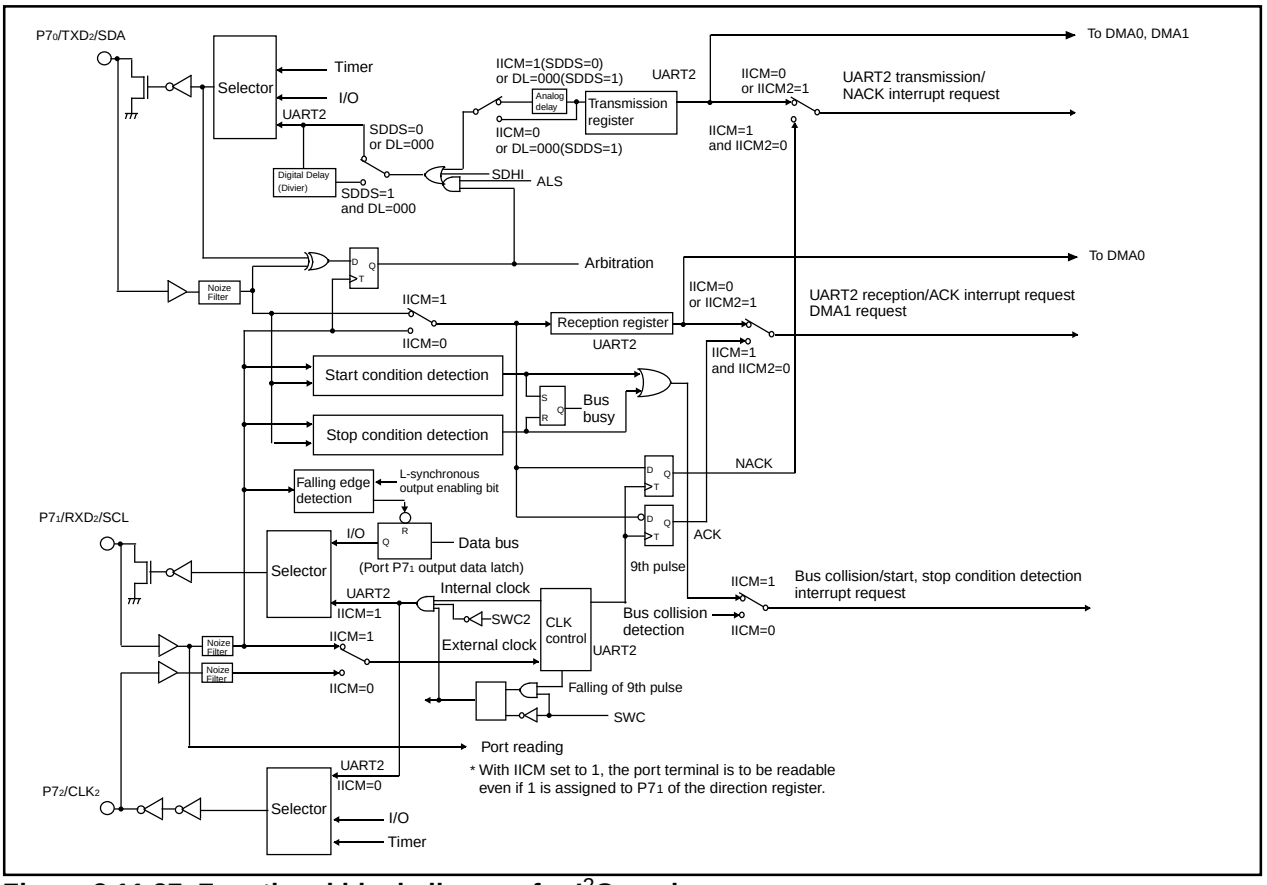



**Figure 2.11.27 Functional block diagram for I²C mode**

### Table 2.11.9 Features in I²C mode

| | Function | Normal mode | I²C mode (Note 1) |
|---|---|---|---|
| 1 | Factor of interrupt number 10 (Note 2) | Bus collision detection | Start condition detection or stop condition detection |
| 2 | Factor of interrupt number 15 (Note 2) | UART2 transmission | No acknowledgment detection (NACK) |
| 3 | Factor of interrupt number 16 (Note 2) | UART2 reception | Acknowledgment detection (ACK) |
| 4 | UART2 transmission output delay | Not delayed | Delayed |
| 5 | P70 at the time when UART2 is in use | TxD2 (output) | SDA (input/output) (Note 3) |
| 6 | P71 at the time when UART2 is in use | RxD2 (input) | SCL (input/output) |
| 7 | P72 at the time when UART2 is in use | CLK2 | P72 |
| 8 | DMA1 factor at the time when 1 1 0 1 is assigned to the DMA request factor selection bits | UART2 reception | Acknowledgment detection (ACK) |
| 9 | Noise filter width | 15ns | 50ns |
| 10 | Reading P71 | Reading the terminal when 0 is assigned to the direction register | Reading the terminal regardless of the value of the direction register |
| 11 | Initial value of UART2 output | H level (when 0 is assigned to the CLK polarity select bit) | The value set in latch P70 when the port is selected |

Note 1: Make the settings given below when I²C mode is in use.
Set 0 1 0 in bits 2, 1, 0 of the UART2 transmission/reception mode register.
Disable the RTS/CTS function. Choose the MSB First function.
Note 2: Follow the steps given below to switch from a factor to another.
1. Disable the interrupt of the corresponding number.
2. Switch from a factor to another.
3. Reset the interrupt request flag of the corresponding number.
4. Set an interrupt level of the corresponding number.
Note 3: Set an initial value of SDA transmission output when serial I/O is invalid.

Figure 2.11.27 hows the functional block diagram for $I^2C$ mode. Setting “1” in the $I^2C$ mode selection bit (IICM) causes ports $P7_0$, $P7_1$, and $P7_2$ to work as data transmission-reception terminal SDA, clock input-output terminal SCL, and port $P7_2$ respectively. A delay circuit is added to the SDA transmission output, so the SDA output changes after SCL fully goes to “L”. Can select analog delay or digital delay by SDA digital delay selection bit (7 bit of address $037_{16}$). When select digital delay, can select delay to 2 cycle to 8 cycle of f1 by UART2 special mode register 3 (address $037_{16}$) . Functions changed by $I^2C$ mode selection bit 2 is shown in below.

**Table 2.11.10 Delay circuit selection condition**

| | Register value | | | Contents |
|---|---|---|---|---|
| | IICM | SDDS | DL | |
| Digital delay selection | 1 | 1 | 001 ≀ 111 | When select digital delay, analog delay is not added. Only digital delay. |
| Analog delay selection | 1 | 1 | 000 | When select DL="000" , analog delay is chosen regardless of the value of SDDS. |
| | | 0 | (000) | When SDDS="0" , DL is initialized and DL="000". |
| No delay | 0 | 0 | (000) | Delay circuit is not selected when IICM="0". But, must set SDDS="0" when IICM="0". |

An attempt to read Port $P7_1$ (SCL) results in getting the terminal’s level regardless of the content of the port direction register. The initial value of SDA transmission output in this mode goes to the value set in port $P7_0$. The interrupt factors of the bus collision detection interrupt, UART2 transmission interrupt, and of UART2 reception interrupt turn to the start/stop condition detection interrupt, acknowledgment non-detection interrupt, and acknowledgment detection interrupt respectively.

The start condition detection interrupt refers to the interrupt that occurs when the falling edge of the SDA terminal ($P7_0$) is detected with the SCL terminal ($P7_1$) staying “H”. The stop condition detection interrupt refers to the interrupt that occurs when the rising edge of the SDA terminal ($P7_0$) is detected with the SCL terminal ($P7_1$) staying “H”. The bus busy flag (bit 2 of the UART2 special mode register) is set to “1” by the start condition detection, and set to “0” by the stop condition detection.

The acknowledgment non-detection interrupt refers to the interrupt that occurs when the SDA terminal level is detected still staying “H” at the rising edge of the 9th transmission clock. The acknowledgment detection interrupt refers to the interrupt that occurs when SDA terminal’s level is detected already went to “L” at the 9th transmission clock. Also, assigning 1101(UART2 reception) to the DMA1 request factor select bits provides the means to start up the DMA transfer by the effect of acknowledgment detection.

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

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

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

Some other functions added are explained here. Figure 2.11.28 shows their workings.

Bit 4 of the UART2 special mode register is used as the bus collision detect sampling clock select bit. The bus collision detect interrupt occurs when the RxD2 level and TxD2 level do not match, but the nonconformity is detected in synchronization with the rising edge of the transfer clock signal if the bit is set to “0”. If this bit is set to “1”, the nonconformity is detected at the timing of the overflow of timer A0 rather than at the rising edge of the transfer clock.

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

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

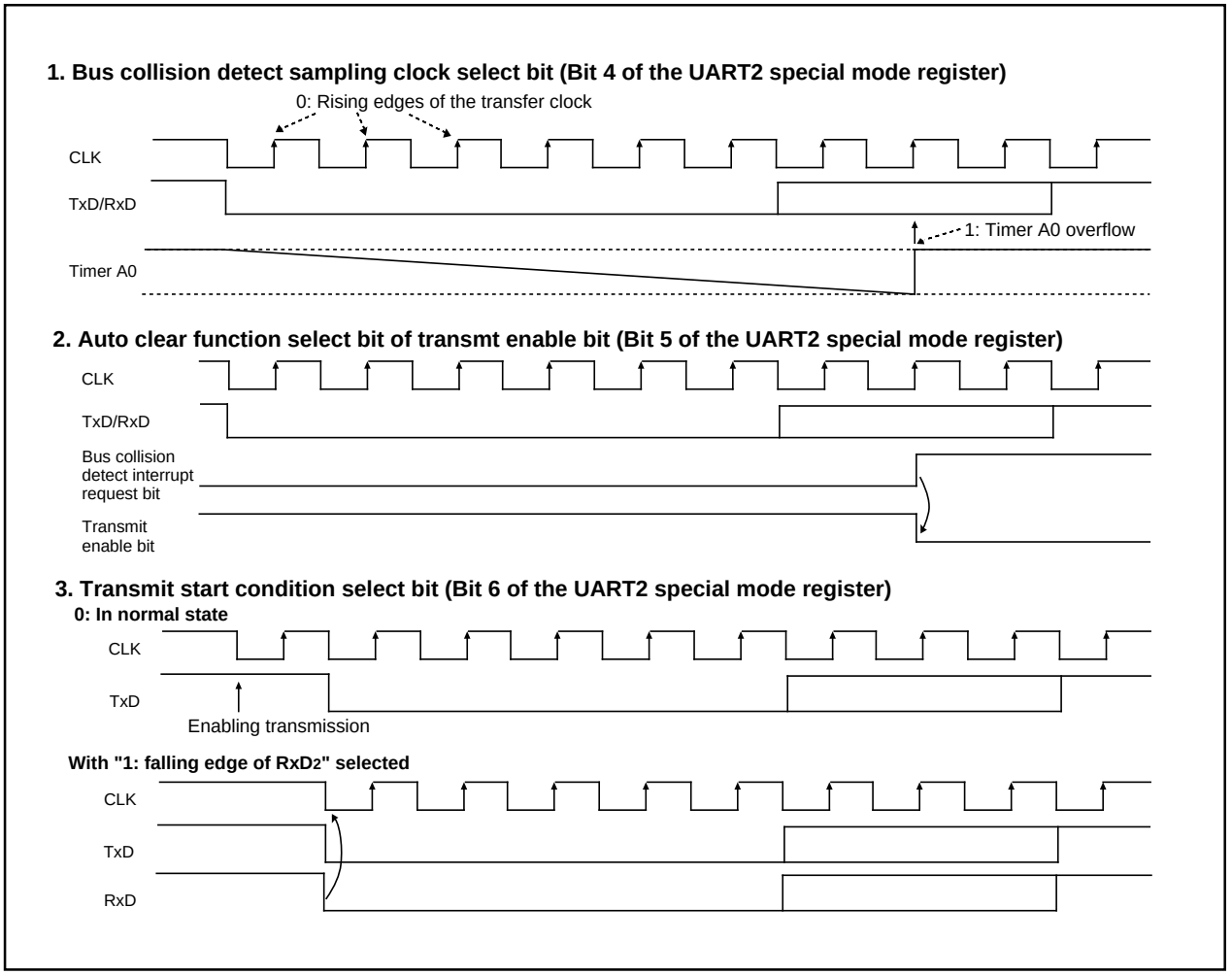
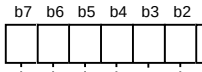



**Figure 2.11.28 Some other functions added**

### 2.11.6 UART2 Special Mode Register 2

UART2 special mode register 2 (address $0376_{16}$) is used to further control UART2 in $I^2C$ mode. Figure 2.11.29 shows the UART2 special mode register 2.

UART2 special mode register 2




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

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| IICM2 | $I^2C$ mode selection bit 2 | Refer to Table 2.11.11 | ○ | ○ |
| CSC | Clock-synchronous bit | 0 : Disabled<br>1 : Enabled | ○ | ○ |
| SWC | SCL wait output bit | 0 : Disabled<br>1 : Enabled | ○ | ○ |
| ALS | SDA output stop bit | 0 : Disabled<br>1 : Enabled | ○ | ○ |
| STAC | UART2 initialization bit | 0 : Disabled<br>1 : Enabled | ○ | ○ |
| SWC2 | SCL wait output bit 2 | 0: UART2 clock<br>1: 0 output | ○ | ○ |
| SDHI | SDA output disable bit | 0: Enabled<br>1: Disabled (high impedance) | ○ | ○ |
| SHTC | Start/stop condition control bit | Set this bit to "1" in $I^2C$ mode (refer to Table 2.11.12) | ○ | ○ |

**Figure 2.11.29 UART2 special mode register 2**

Bit 0 of the UART2 special mode register 2 (address $037_{16}$) is used as the $I^2C$ mode selection bit 2. Table 2.11.11 shows the types of control to be changed by $I^2C$ mode selection bit 2 when the $I^2C$ mode selection bit is set to "1". Table 2.11.12 shows the timing characteristics of detecting the start condition and the stop condition. Set the start/stop condition control bit (bit 7 of UART2 special mode register 2) to "1" in $I^2C$ mode.

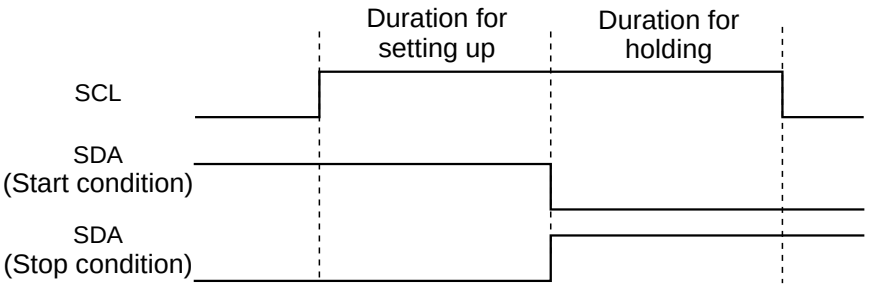
**Table 2.11.11 Functions changed by $I^2C$ mode selection bit 2**

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

**Table 2.11.12 Timing characteristics of detecting the start condition and the stop condition(Note1)**

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

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



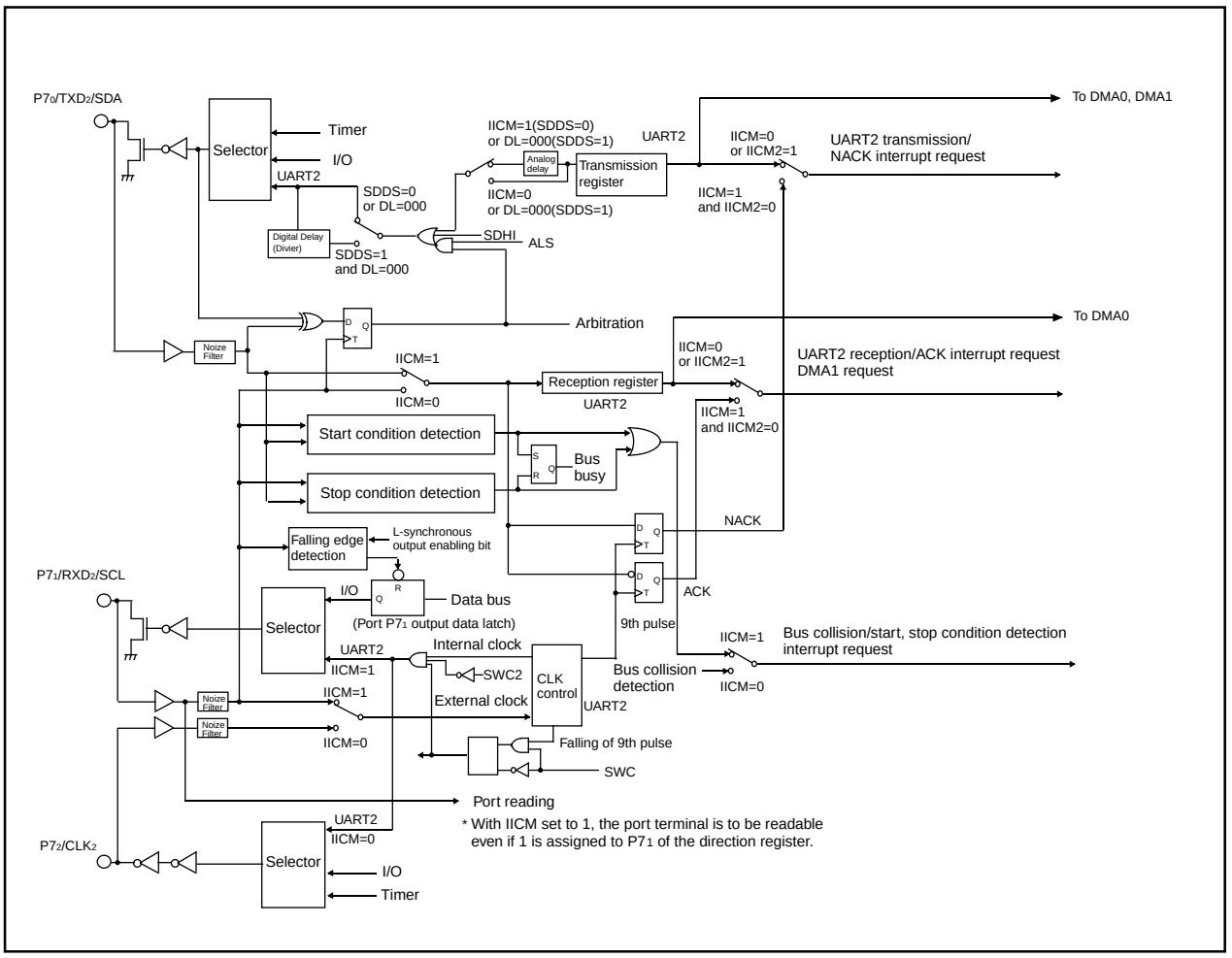



**Figure 2.11.30 Functional block diagram for I²C mode**

Functions available in I²C mode are shown in Figure 2.11.30— a functional block diagram.

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

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

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

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

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

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

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

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

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

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

### 2.11.7 S I/O3, 4

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

Figure 2.11.31 shows the S I/O3, 4 block diagram, and Figure 2.11.32 shows the S I/O3, 4 control register.Table 2.11.13 shows the specifications of S I/O3, 4.

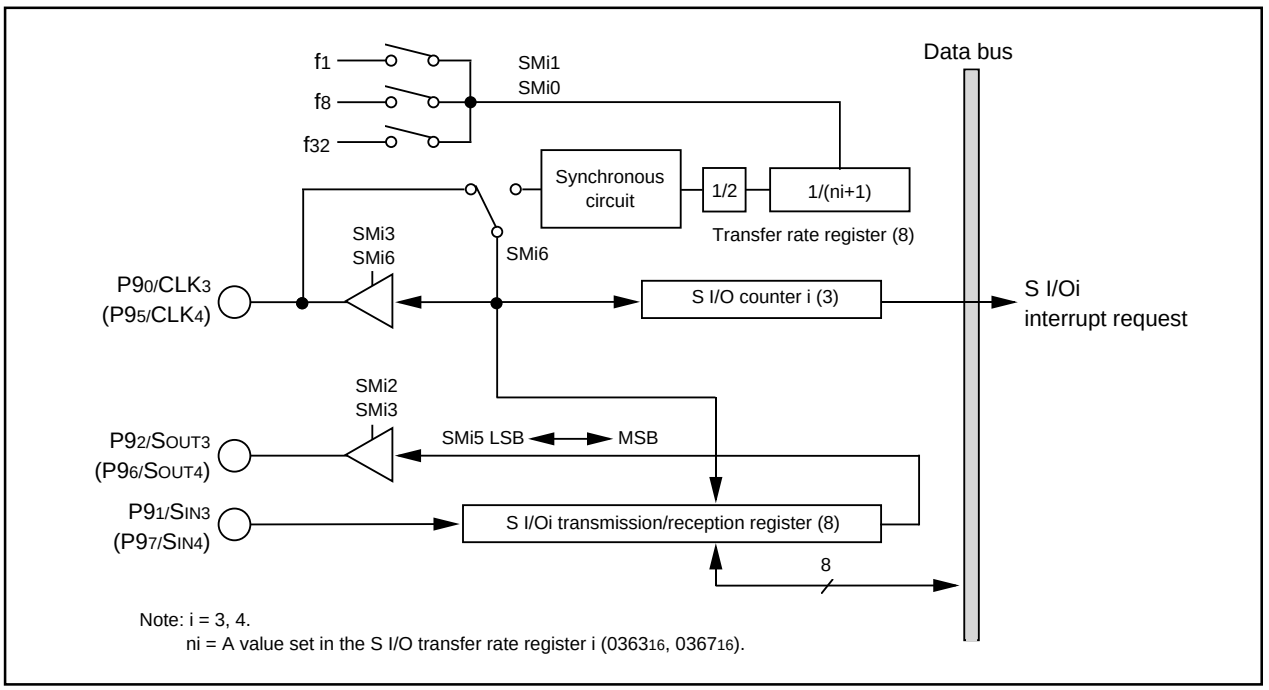
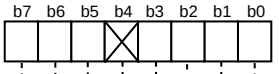



**Figure 2.11.31 S I/O3, 4 block diagram**

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

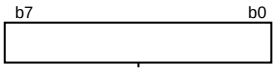



| Symbol | Address | When reset |
|---|---|---|
| SiC | $0362_{16}$, $0366_{16}$ | $40_{16}$ |

| Bit symbol | Bit name | Description | R | W |
|---|---|---|---|---|
| SMi0 | Internal synchronous clock select bit | b1 b0<br>0 0 : Selecting $f_1$<br>0 1 : Selecting $f_8$<br>1 0 : Selecting $f_{32}$<br>1 1 : Not to be used | ○ | ○ |
| SMi1 | | | ○ | ○ |
| SMi2 | SOUTi output disable bit | 0 : SOUTi output<br>1 : SOUTi output disable(high impedance) | ○ | ○ |
| SMi3 | S I/Oi port select bit (Note 2) | 0 : Input-output port<br>1 : SOUTi output, CLK function | ○ | ○ |
| Nothing is assigned. In an attempt to write to this bit, write "0". The value, if read, turns out to be "0". | | | — | — |
| SMi5 | Transfer direction lect bit | 0 : LSB first<br>1 : MSB first | ○ | ○ |
| SMi6 | Synchronous clock select bit (Note 2) | 0 : External clock<br>1 : Internal clock | ○ | ○ |
| SMi7 | SOUTi initial value set bit | Effective when SMi3 = 0<br>0 : L output<br>1 : H output | ○ | ○ |

Note 1: Set "1" in bit 2 of the protection register ($000A_{16}$) in advance to write to the S I/Oi control register (i = 3, 4).
Note 2: When using the port as an input/output port by setting the SI/Oi port select bit (i = 3, 4) to "1", be sure to set the sync clock select bit to "1".

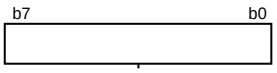
SI/Oi bit rate generator




| Symbol | Address | When reset |
|---|---|---|
| S3BRG | $0363_{16}$ | Indeterminate |
| S4BRG | $0367_{16}$ | Indeterminate |

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

SI/Oi transmit/receive register




| Symbol | Address | When reset |
|---|---|---|
| S3TRR | $0360_{16}$ | Indeterminate |
| S4TRR | $0364_{16}$ | Indeterminate |

| Indeterminate | R | W |
|---|---|---|
| Transmission/reception starts by writing data to this register. After transmission/reception finishes, reception data is input. | ○ | ○ |

**Figure 2.11.32 S I/O3, 4 related register**

**Table 2.11.13 Specifications of S I/O3, 4**

| Item | Specifications |
|---|---|
| Transfer data format | • Transfer data length: 8 bits |
| Transfer clock | • With the internal clock selected (bit 6 of $0362_{16}$, $0366_{16}$ = "1"): f1/2(ni+1), f8/2(ni+1), f32/2(ni+1) (Note 1)<br>• With the external clock selected (bit 6 of $0362_{16}$, $0366_{16}$ = 0):Input from the CLKi terminal (Note 2) |
| Conditions for transmission/ reception start | • To start transmit/reception, the following requirements must be met:<br>- Select the synchronous clock (use bit 6 of $0362_{16}$, $0366_{16}$).<br>Select a frequency dividing ratio if the internal clock has been selected (use bits 0 and 1 of $0362_{16}$, $0366_{16}$).<br>- SOUTi initial value set bit (use bit 7 of $0362_{16}$, $0366_{16}$)= 1.<br>- S I/Oi port select bit (bit 3 of $0362_{16}$, $0366_{16}$) = 1.<br>- Select the transfer direction (use bit 5 of $0362_{16}$, $0366_{16}$)<br>-Write transfer data to SI/Oi transmit/receive register ($0360_{16}$, $0364_{16}$)<br>• To use S I/Oi interrupts, the following requirements must be met:<br>- Clear the SI/Oi interrupt request bit before writing transfer data to the SI/Oi transmit/receive register (bit 3 of $0049_{16}$, $0048_{16}$) = 0. |
| Interrupt request generation timing | • Rising edge of the last transfer clock. (Note 3) |
| Select function | • LSB first or MSB first selection<br>Whether transmission/reception begins with bit 0 (LSB) or bit 7 (MSB) can be selected.<br>• Function for setting an SOUTi initial value selection<br>When using an external clock for the transfer clock, the user can choose the SOUTi pin output level during a non-transfer time. For details on how to set, see Figure 2.11.33. |
| Precaution | • Unlike UART0–2, SI/Oi (i = 3, 4) is not divided for transfer register and buffer. Therefore, do not write the next transfer data to the SI/Oi transmit/receive register (addresses $0360_{16}$, $0364_{16}$) during a transfer. When the internal clock is selected for the transfer clock, SOUTi holds the last data for a 1/2 transfer clock period after it finished transferring and then goes to a high-impedance state. However, if the transfer data is written to the SI/Oi transmit/receive register (addresses $0360_{16}$, $0364_{16}$) during this time, SOUTi is placed in the high-impedance state immediately upon writing and the data hold time is thereby reduced. |

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

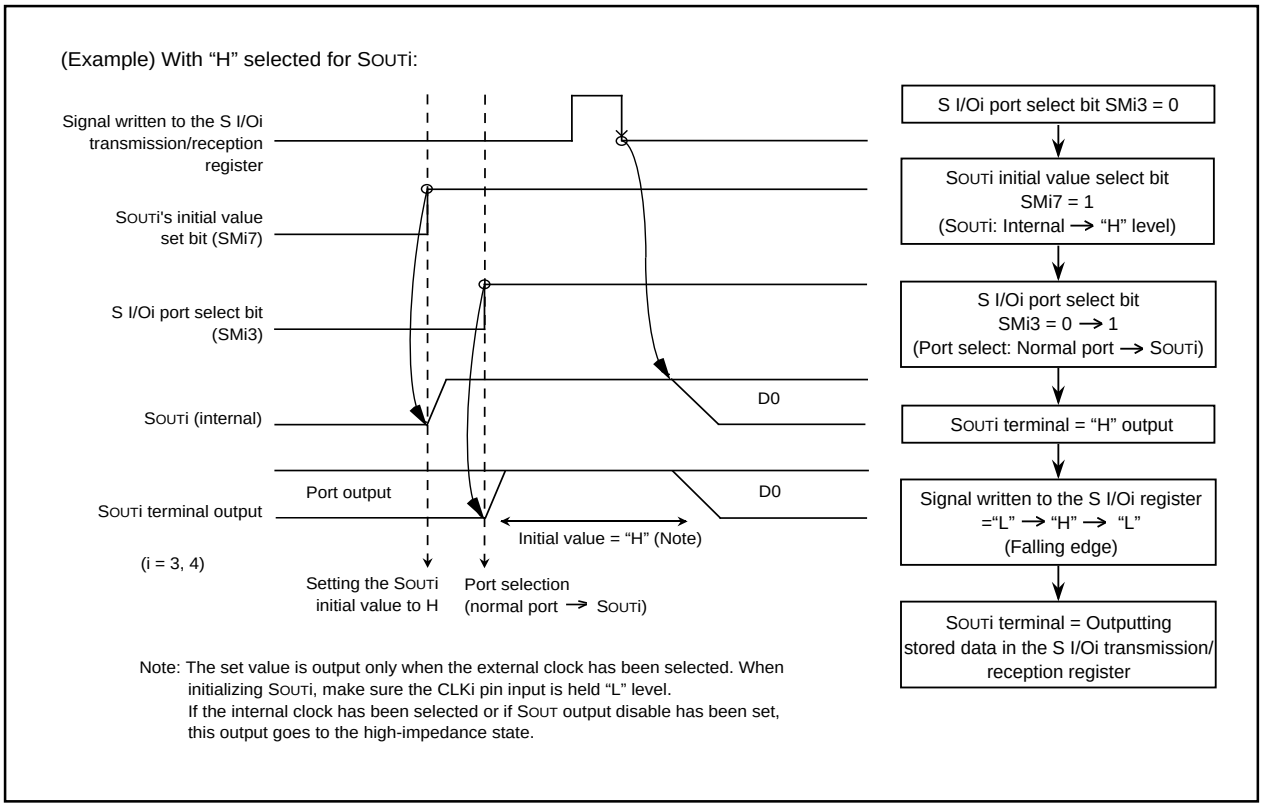
Note 2: With the external clock selected:

- Before data can be written to the SI/Oi transmit/receive register (addresses $0360_{16}$, $0364_{16}$), the CLKi pin input must be in the low state. Also, before rewriting the SI/Oi Control Register (addresses $0362_{16}$, $0366_{16}$)'s bit 7 (SOUTi initial value set bit), make sure the CLKi pin input is held low.
- The S I/Oi circuit keeps on with the shift operation as long as the synchronous clock is entered in it, so stop the synchronous clock at the instant when it counts to eight. The internal clock, if selected, automatically stops.

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

### (1) Functions for setting an SOUTi initial value

When using an external clock for the transfer clock, the SOUTi pin output level during a non-transfer time can be set to the high or the low state. Figure 2.11.33 shows the timing chart for setting an SOUTi initial value and how to set it.

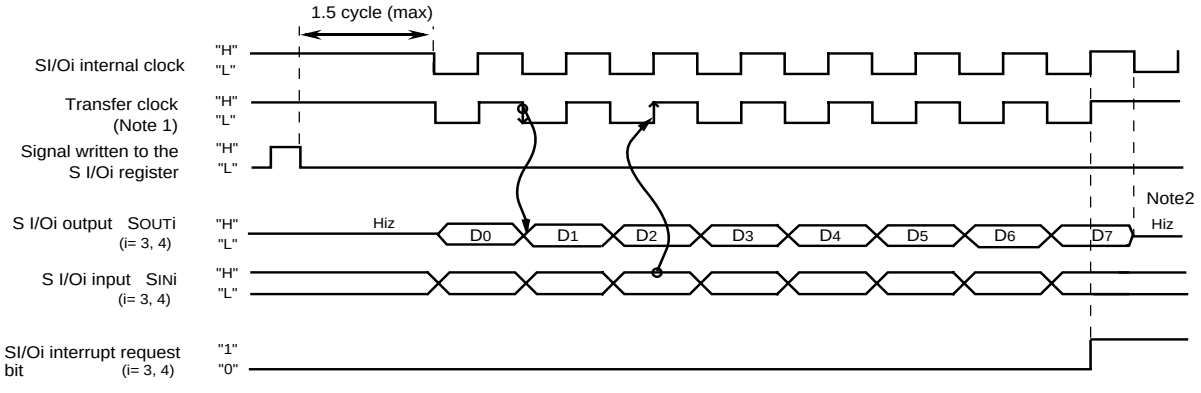



Note: The set value is output only when the external clock has been selected. When initializing SOUTi, make sure the CLKi pin input is held "L" level.
If the internal clock has been selected or if SOUT output disable has been set, this output goes to the high-impedance state.

**Figure 2.11.33 Timing chart for setting SOUTi's initial value and how to set it**

### (2) S I/Oi operation timing

Figure 2.11.34 shows the S I/Oi operation timing




Note 1: With the internal clock selected for the transfer clock, the frequency dividing ratio can be selected using bits 0 and 1 of the S I/Oi control register. (i=3,4) (No frequency division, 8-division frequency, 32-division frequency.)
Note 2: With the internal clock selected for the transfer clock, the SOUTi pin becomes to the high-impedance state after the transfer finishes.
Note 3: Shown above is the case where the SOUTi (i = 3, 4) port select bit ="1".

**Figure 22.11.34 S I/Oi operation timing chart**

## 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 $P10_0$ to $P10_7$, $P9_5$, and $P9_6$ also function as the analog signal input pins. The direction registers of these pins for A-D conversion must therefore be set to input. The Vref connect bit (bit 5 at address $03D7_{16}$) can be used to isolate the resistance ladder of the A-D converter from the reference voltage input pin ($V_{REF}$) when the A-D converter is not used. Doing so stops any current flowing into the resistance ladder from $V_{REF}$, reducing the power dissipation. When using the A-D converter, start A-D conversion only after setting bit 5 of $03D7_{16}$ to connect $V_{REF}$.

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

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 and 2.12.3 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 $AV_{CC}$ ($V_{CC}$) |
| Operating clock $\phi_{AD}$ (Note 2) | $f_{AD}$/divide-by-2 of $f_{AD}$/divide-by-4 of $f_{AD}$, $f_{AD}=f(X_{IN})$ |
| Resolution | 8-bit |
| Absolute precision | • Without sample and hold function<br>±3LSB<br>• With sample and hold function<br>±2LSB |
| Operating modes | One-shot mode, repeat mode, single sweep mode, repeat sweep mode 0, and repeat sweep mode 1 |
| Analog input pins | 8pins ($AN_0$ to $AN_7$) + 2pins (ANEX0 and ANEX1) |
| A-D conversion start condition | • Software trigger<br>A-D conversion starts when the A-D conversion start flag changes to “1”<br>• External trigger (can be retriggered)<br>A-D conversion starts when the A-D conversion start flag is “1” and the $\overline{ADTRG}$/$P9_7$ input changes from “H” to “L” |
| Conversion speed per pin | • Without sample and hold function<br>49 $\phi_{AD}$ cycles<br>• With sample and hold function<br>28 $\phi_{AD}$ cycles |

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

Note 2: Without sample and hold function, set the $\phi_{AD}$ frequency to 250kHz min.
With the sample and hold function, set the $\phi_{AD}$ frequency to 1MHz min.

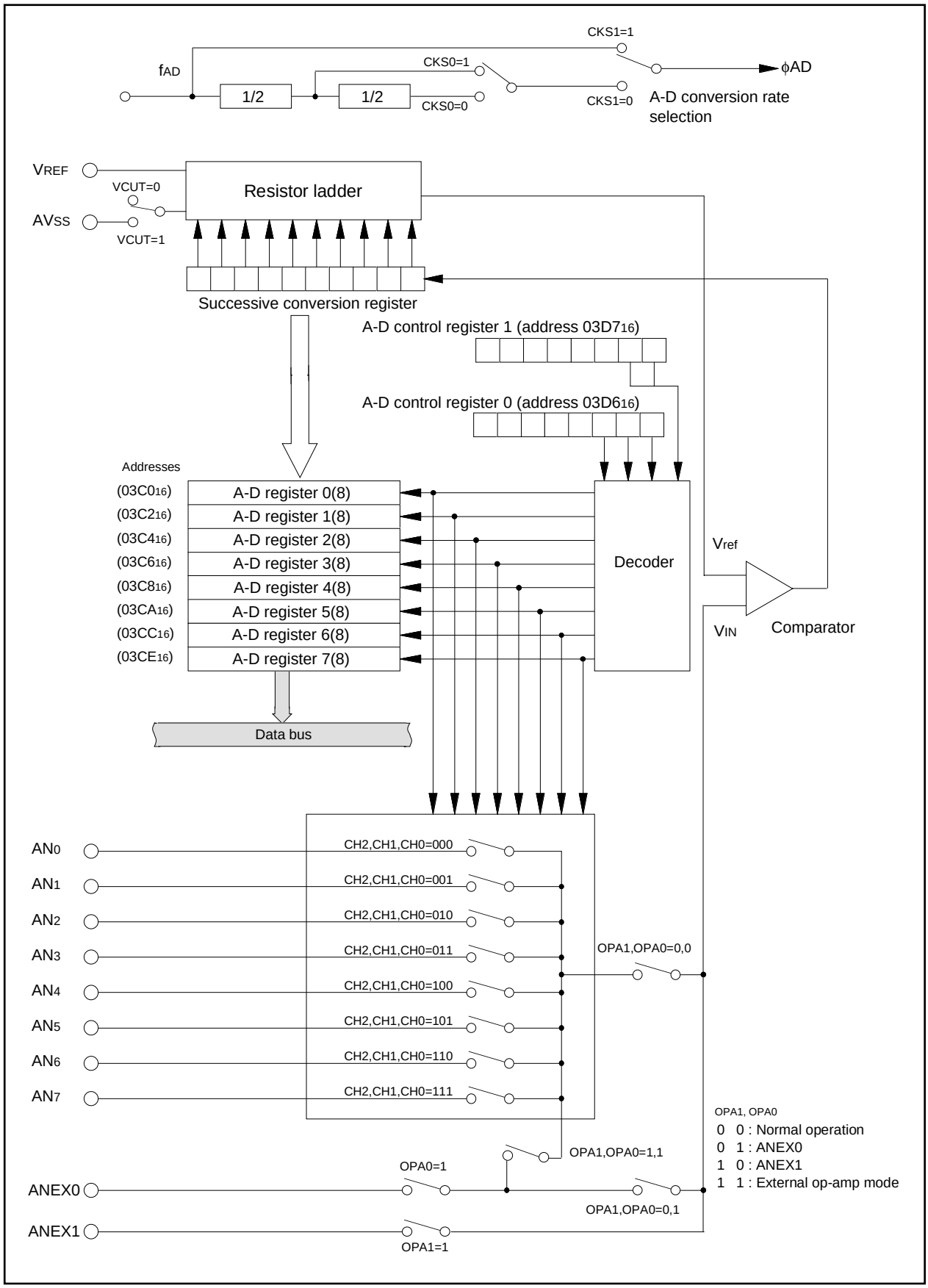



**Figure 2.12.1 Block diagram of A-D converter**

A-D control register 0 (Note 1)

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

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

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

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

A-D control register 1 (Note)

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

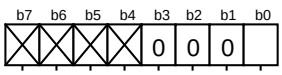
| Symbol | Address | When reset |
|---|---|---|
| ADCON1 | $03D7_{16}$ | $00_{16}$ |

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

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

**Figure 2.12.2 A-D converter-related registers (1)**

A-D control register 2 (Note)



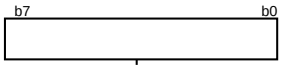

| Symbol | Address | When reset |
|---|---|---|
| ADCON2 | $03D4_{16}$ | $0000XXX0_2$ |

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

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

A-D register i

| Symbol | Address | When reset |
|---|---|---|
| ADi(i=0 to 7) | $03C0_{16}$,$03C2_{16}$,$03C4_{16}$,$03C6_{16}$,<br>$03C8_{16}$,$03CA_{16}$,$03CC_{16}$,$03CE_{16}$ | Indeterminate |




| Function | R | W |
|---|---|---|
| Eight bits of A-D conversion result | ○ | × |

**Figure 2.12.3 A-D converter-related registers (2)**

**(1) One-shot mode**

In one-shot mode, the pin selected using the analog input pin select bit is used for one-shot A-D conversion. Table 2.12.2 shows the specifications of one-shot mode. Figure 2.12.4 shows 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 | • End of A-D conversion (A-D conversion start flag changes to "0", except when external trigger is selected)<br>• Writing "0" to A-D conversion start flag |
| Interrupt request generation timing | End of A-D conversion |
| Input pin | One of $AN_0$ to $AN_7$, as selected |
| Reading of result of A-D converter | Read A-D register corresponding to selected pin |

A-D control register 0 (Note 1)

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

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

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

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

A-D control register 1 (Note)

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

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

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

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

**Figure 2.12.4 A-D conversion register in one-shot mode**

### (2) Repeat mode

In repeat mode, the pin selected using the analog input pin select bit is used for repeated A-D conversion. Table 2.12.3 shows the specifications of repeat mode. Figure 2.12.5 shows 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 |
| Star condition | Writing “1” to A-D conversion start flag |
| Stop condition | Writing “0” to A-D conversion start flag |
| Interrupt request generation timing | None generated |
| Input pin | One of $AN_0$ to $AN_7$, as selected |
| Reading of result of A-D converter | Read A-D register corresponding to selected pin |

A-D control register 0 (Note 1)

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

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

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

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

A-D control register 1 (Note)

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

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

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

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

**Figure 2.12.5 A-D conversion register in repeat mode**

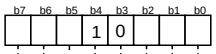
### (3) Single sweep mode

In single sweep mode, the pins selected using the A-D sweep pin select bit are used for one-by-one A-D conversion. Table 2.12.4 shows the specifications of single sweep mode. Figure 2.12.6 shows 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 | • End of A-D conversion (A-D conversion start flag changes to "0", except when external trigger is selected)<br>• Writing "0" to A-D conversion start flag |
| Interrupt request generation timing | End of A-D conversion |
| Input pin | AN0 and AN1 (2 pins), AN0 to AN3 (4 pins), AN0 to AN5 (6 pins), or AN0 to AN7 (8 pins) |
| Reading of result of A-D converter | Read A-D register corresponding to selected pin |

A-D control register 0 (Note)

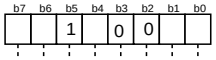



Symbol ADCON0 — Address 03D6₁₆ — When reset 00000XXX₂

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

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

A-D control register 1 (Note 1)




Symbol ADCON1 — Address 03D7₁₆ — When reset 00₁₆

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SCAN0 | A-D sweep pin select bit | When single sweep and repeat sweep mode 0 are selected<br>b1 b0<br>0 0 : AN0, AN1 (2 pins)<br>0 1 : AN0 to AN3 (4 pins)<br>1 0 : AN0 to AN5 (6 pins)<br>1 1 : AN0 to AN7 (8 pins) | O | O |
| SCAN1 | | | O | O |
| MD2 | A-D operation mode select bit 1 | 0 : Any mode other than repeat sweep mode 1 | O | O |
| Reserved bit | | Must always be set to "0". | O | O |
| CKS1 | Frequency select bit 1 | 0 : fAD/2 or fAD/4 is selected<br>1 : fAD is selected | O | O |
| VCUT | Vref connect bit | 1 : Vref connected | O | O |
| OPA0 | External op-amp connection mode bit (Note 2) | b7 b6<br>0 0 : ANEX0 and ANEX1 are not used<br>0 1 : ANEX0 input is A-D converted<br>1 0 : ANEX1 input is A-D converted<br>1 1 : External op-amp connection mode | O | O |
| OPA1 | | | O | O |

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

**Figure 2.12.6 A-D conversion register in single sweep mode**

### (4) Repeat sweep mode 0

In repeat sweep mode 0, the pins selected using the A-D sweep pin select bit are used for repeat sweep A-D conversion. Table 2.12.5 shows the specifications of repeat sweep mode 0. Figure 2.12.7 shows 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 | $AN_0$ and $AN_1$ (2 pins), $AN_0$ to $AN_3$ (4 pins), $AN_0$ to $AN_5$ (6 pins), or $AN_0$ to $AN_7$ (8 pins) |
| Reading of result of A-D converter | Read A-D register corresponding to selected pin (at any time) |

A-D control register 0 (Note)

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

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

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

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

A-D control register 1 (Note 1)

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

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

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

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

**Figure 2.12.7 A-D conversion register in repeat sweep mode 0**

**(5) Repeat sweep mode 1**

In repeat sweep mode 1, all pins are used for A-D conversion with emphasis on the pin or pins selected using the A-D sweep pin select bit. Table 2.12.6 shows the specifications of repeat sweep mode 1. Figure 2.12.8 shows 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<br>Example : $AN_0$ selected $AN_0 \rightarrow AN_1 \rightarrow AN_0 \rightarrow AN_2 \rightarrow AN_0 \rightarrow AN_3$, etc |
| Start condition | Writing "1" to A-D conversion start flag |
| Stop condition | Writing "0" to A-D conversion start flag |
| Interrupt request generation timing | None generated |
| Input pin | $AN_0$ (1 pin), $AN_0$ and $AN_1$ (2 pins), $AN_0$ to $AN_2$ (3 pins), $AN_0$ to $AN_3$ (4 pins) |
| Reading of result of A-D converter | Read A-D register corresponding to selected pin (at any time) |

A-D control register 0 (Note)

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

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

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

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

A-D control register 1 (Note 1)

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

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

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

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

**Figure 2.12.8 A-D conversion register in repeat sweep mode 1**

**(a) Sample and hold**

Sample and hold is selected by setting bit 0 of the A-D control register 2 (address 03D4$_{16}$) to “1”. When sample and hold is selected, the rate of conversion of each pin increases. As a result, a 28 $f_{AD}$ cycle is achieved. Sample and hold can be selected in all modes. However, in all modes, be sure to specify before starting A-D conversion whether sample and hold is to be used.

**(b) Extended analog input pins**

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

When bit 6 of the A-D control register 1 (address 03D7$_{16}$) is “1” and bit 7 is “0”, input via ANEX0 is converted from analog to digital. The result of conversion is stored in A-D register 0.

When bit 6 of the A-D control register 1 (address 03D7$_{16}$) is “0” and bit 7 is “1”, input via ANEX1 is converted from analog to digital. The result of conversion is stored in A-D register 1.

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

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

When bit 6 of the A-D control register 1 (address 03D7$_{16}$) is “1” and bit 7 is “1”, input via $AN_0$ to $AN_7$ is output from ANEX0. The input from ANEX1 is converted from analog to digital and the result stored in the corresponding A-D register. The speed of A-D conversion depends on the response of the external operation amp. Do not connect the ANEX0 and ANEX1 pins directly. Figure 2.12.9 is an example of how to connect the pins in external operation amp mode.

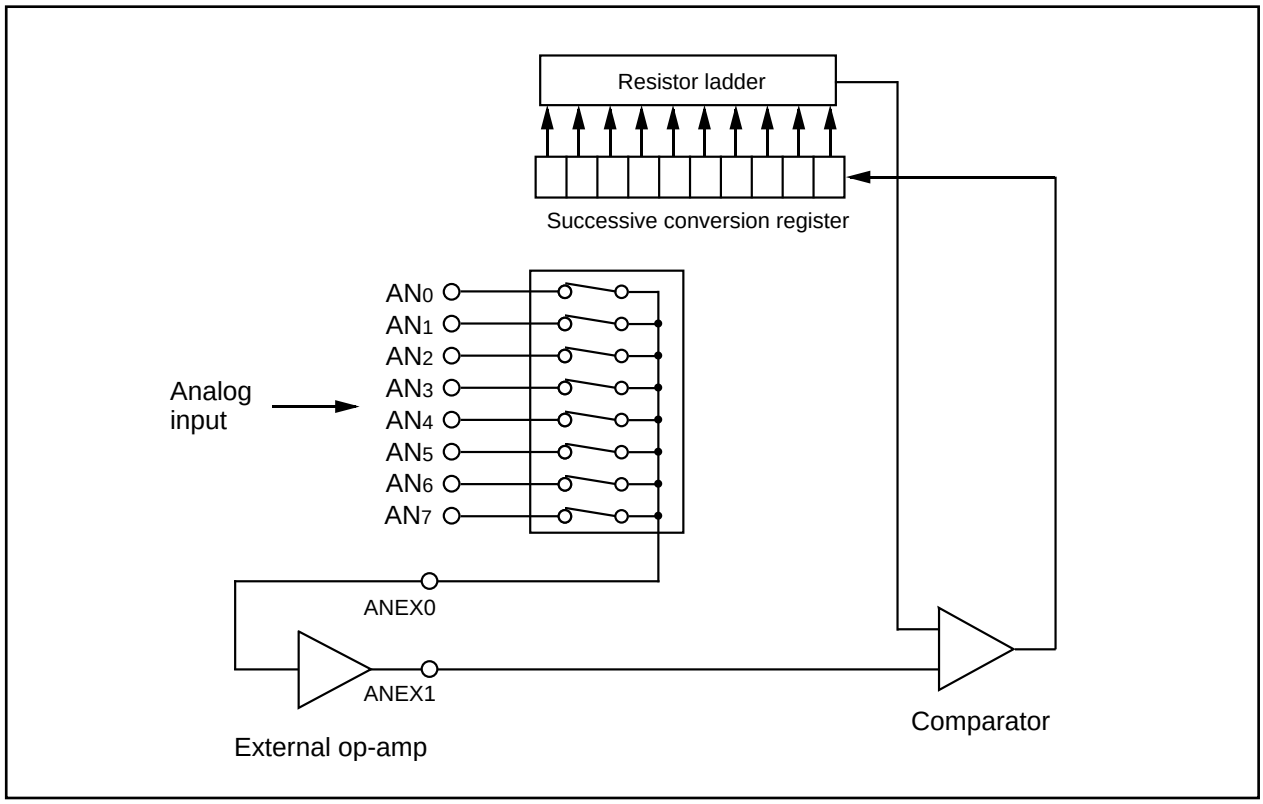



**Figure 2.12.9 Example of external op-amp connection mode**

## 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.

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

$V_{REF}$ : reference voltage

Table 2.13.1 lists the performance of the D-A converter. Figure 2.13.1 shows the block diagram of the D-A converter. Figure 2.13.2 shows the D-A control register. Figure 2.13.3 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 |

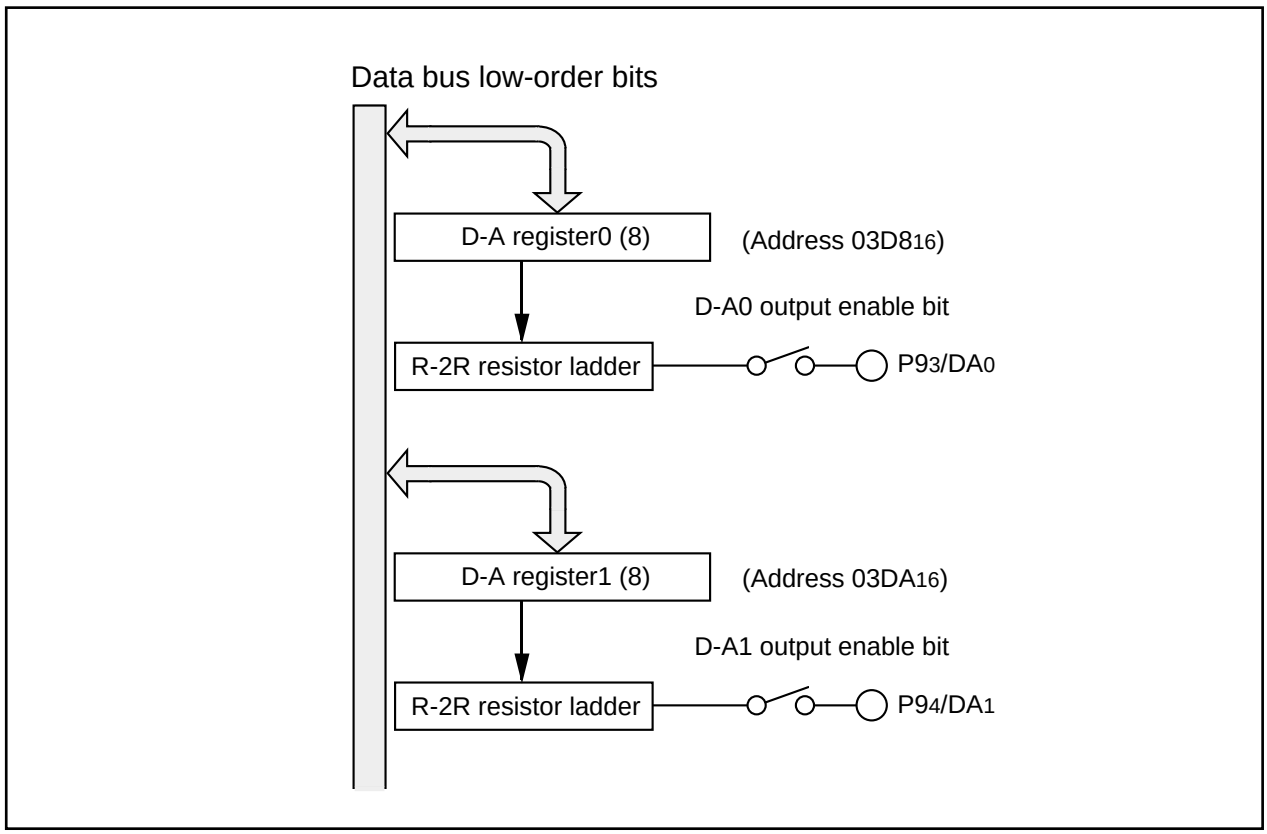



**Figure 2.13.1 Block diagram of D-A converter**

D-A control register

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

| Symbol | Address | When reset |
|---|---|---|
| DACON | $03DC_{16}$ | $00_{16}$ |

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

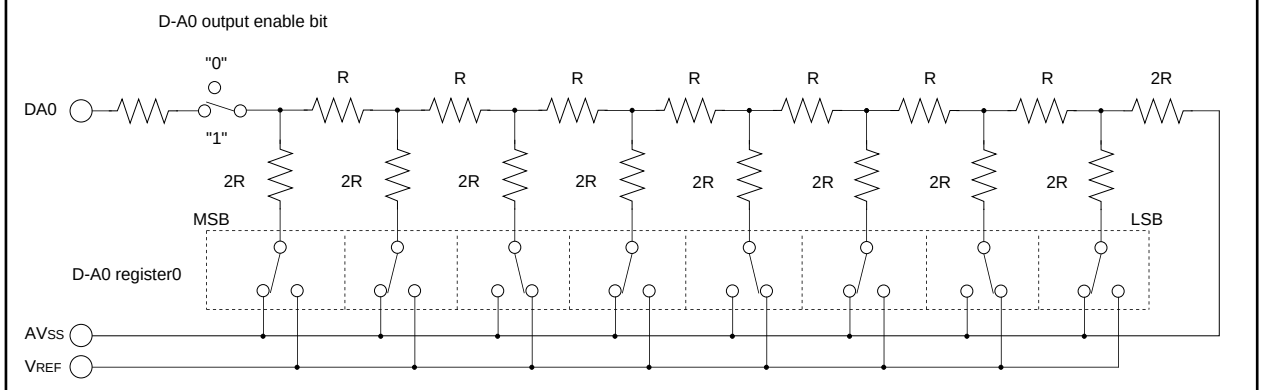
D-A register

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

| Symbol | Address | When reset |
|---|---|---|
| DAi (i = 0,1) | $03D8_{16}$, $03DA_{16}$ | Indeterminate |

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

**Figure 2.13.2 D-A control register**




Note 1: The above diagram shows an instance in which the D-A register is assigned $2A_{16}$.
Note 2: The same circuit as this is also used for D-A1.
Note 3: To reduce the current consumption when the D-A converter is not used, set the D-A output enable bit to 0 and set the D-A register to $00_{16}$ so that no current flows in the resistors Rs and 2Rs.

**Figure 2.13.3 D-A converter equivalent circuit**

## 2.14 CRC Calculation Circuit

The Cyclic Redundancy Check (CRC) calculation circuit detects an error in data blocks. The microcomputer uses a generator polynomial of CRC_CCITT ($X^{16} + X^{12} + X^5 + 1$) to generate CRC code.

The CRC code is a 16-bit code generated for a block of a given data length in multiples of 8 bits. The CRC code is set in a CRC data register each time one byte of data is transferred to a CRC input register after writing an initial value into the CRC data register. Generation of CRC code for one byte of data is completed in two machine cycles.

Figure 2.14.1 shows the block diagram of the CRC circuit. Figure 2.14.2 shows the CRC-related registers. Figure 2.14.3 shows the calculation example using the CRC calculation circuit

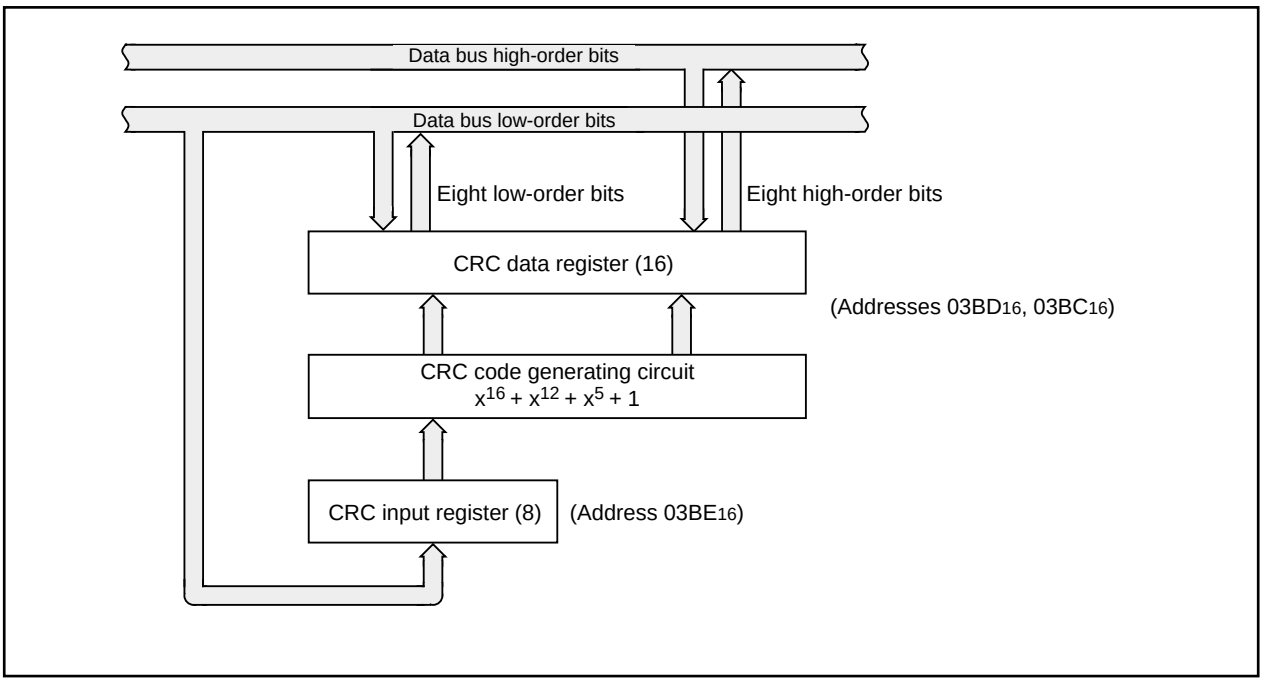
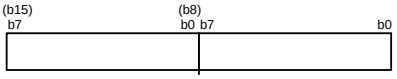



**Figure 2.14.1 Block diagram of CRC circuit**

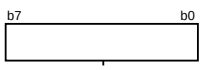
CRC data register




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

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

CRC input register




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

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

**Figure 2.14.2 CRC-related registers**

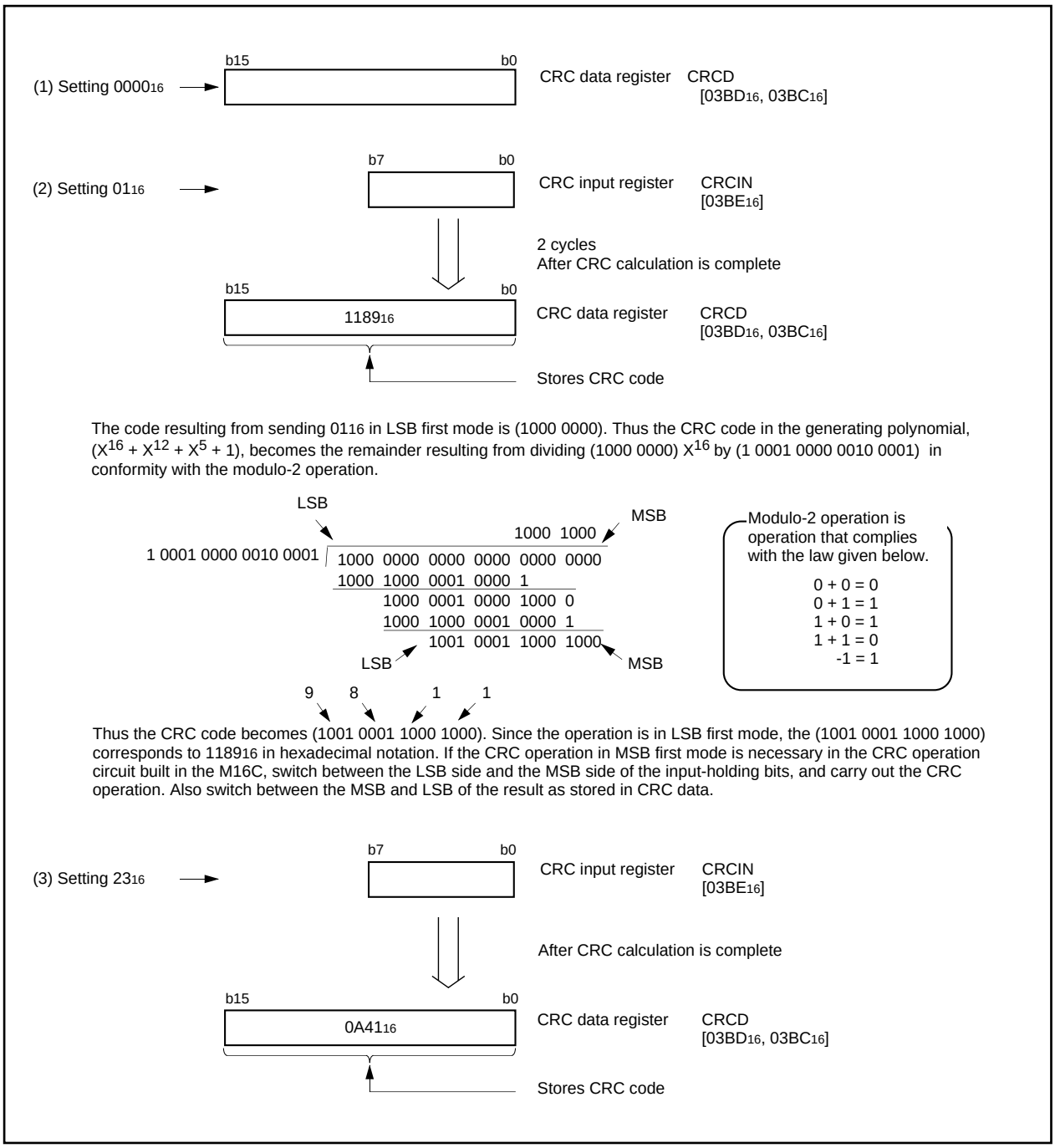



**Figure 2.14.3 Calculation example using the CRC calculation circuit**

## 2.15 Expansion Function

### 2.15.1 Expansion function description

Expansion function cousists of OSD display function, data slicer fanction, data encoder function and humming decoder function. Each function is controld by expansion memories.

**(1) OSD function**

Character is consisted of 12 X 10 dots, can display 40 (horizontal) X 25 (vertical) on the fixed line. And also, can be written over with built-in composite RAM.

M306H0SFP can be reduced external circuit by built-in SYNC-SEP (synchronous separate) and synchronous correction circuit. And it also can reduce error of character display at superimpose.

**Table 2.15.1 OSD function outline**

| | |
|---|---|
| Screen composition | 40 characters X 25 lines Fixed line display<br>(at scrolling 40 characters X 24 lines) |
| Number of characters displayed | 1000 (Max.) |
| Character composition | 12 X 10 dot matrix<br>(horizontal direction : 12 dots, vertical direction : 10 dots) |
| Characters available | Font RAM : 256 characters<br>Composite RAM(SYRAM) : 15 characters |
| Character sizes available | Horizontal : one time, two times<br>Vertical : one time, two times<br>setting by every line |
| Display locations available | Horizontal direction : 486 locations<br>Vertical direction : 235 locations |
| Blinking | Character units<br>Cycle : approximately 1 second, or approximately 0.5 seconds (per screen)<br>Duty 25%, 50% or 75% (per screen) |
| Coloring | Character coloring : 8 colors choices per character<br>Character Background coloring : 8 colors choices per character<br>Background coloring : 8 colors choices per screen |
| Blanking | Character blanking<br>Matrix-outline<br>Halftone blanking<br>Can be set by every line |
| Superimpose | Can be displayed<br>(PAL/SECAM)(monotone display) |
| Synchronous signal | Composite synchronous signal generate (only PAL)<br>Composite video signal generate (only PAL) |
| Scrolling | The top and bottom smooth scroll of the soft control |
| General-purpose output ports | Combined port output : 9<br>(switching to R,G,B,GRAY,BLNK,CSYN,SLICEON, EDO1, EDO2 output) |
| Synchronous correction circuit | Built-in |
| Synchronous separation circuit | Built-in |

**(2) Data slicer function**

Corresponds to TELETEXT, VPS, and VBI data

**(3) Data encoder function**

Encode VBI data

**(4) Humming decoder function**

8/4 humming and 24/18 humming

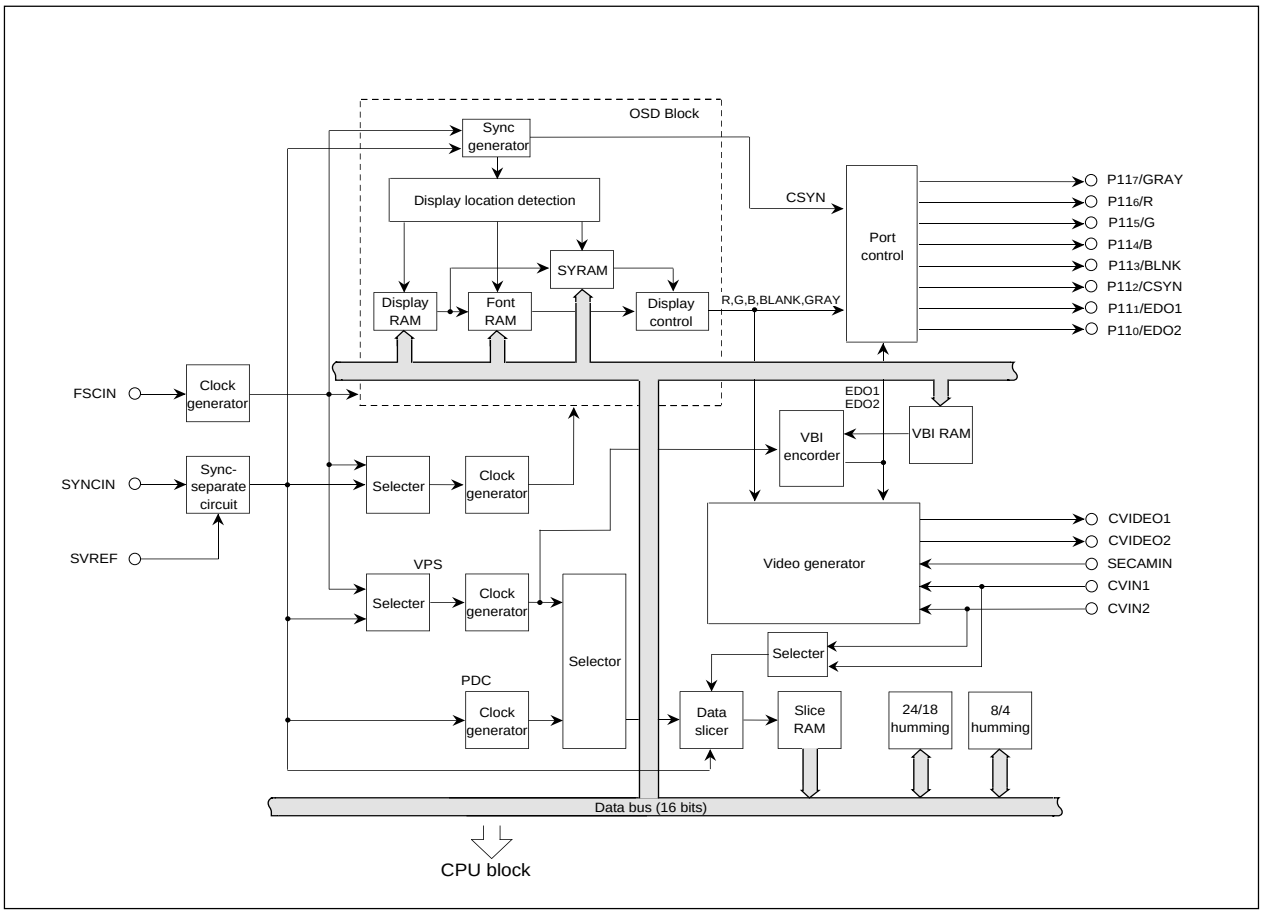



**Figure 2.15.1 Block diagram of expansion function**

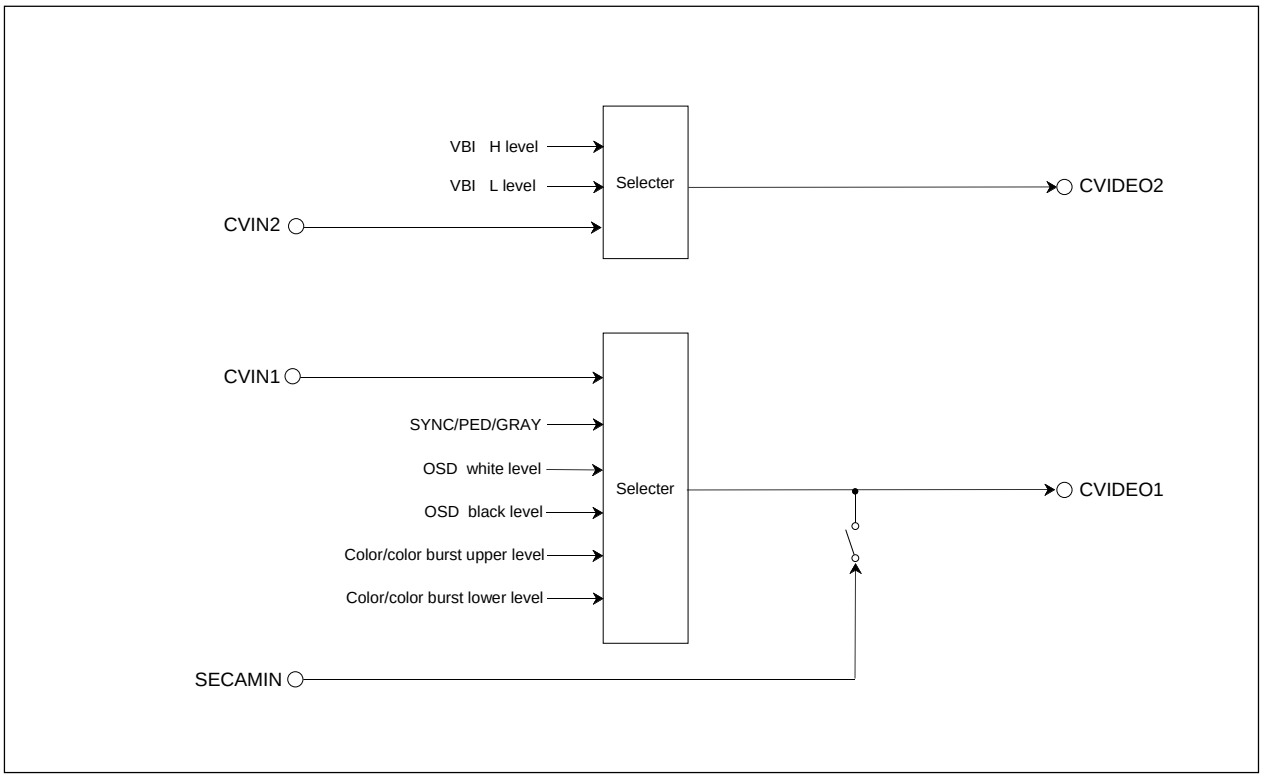



**Figure 2.15.2 Block diagram of video generator**

### 2.15.2 Expansion memory

Expansion function memory is divided by 6 patterns ; display RAM, Font RAM, SYRAM, Slice RAM, VBIRAM and expansion register. (Humming decoder operates by the register placed on SFR). Data writing and read out to these RAM and the expansion register are carried out 16 bit unit by the data setting register (addresses $0202_{16}$ to $0218_{16}$) placed on SFR.

Contents of each memory and data setting register are shown in Table 2.15.2.

**Table 2.15.2 Expansion memory composition**

| Expansion memory | Contents | Data setting register |
|---|---|---|
| Display RAM | 1 screen (40 characters × 25 lines) display character setting. RAM font (character code), character color, character backgroud color, blinking, SYRAM font (character code) and SYRAM character color are specified by 1 character unit. | Display RAM address control register ($0202_{16}$)<br>Display RAM data control register ($0204_{16}$) |
| Font RAM | 255 character fonts setting. | Font RAM address control register ($0206_{16}$)<br>Font RAM data control register ($0208_{16}$) |
| SYRAM | 15 composite character fonts setting. | SYRAM address control register ($020A_{16}$)<br>SYRAM data control register ($020C_{16}$) |
| Slice RAM | Store slice data. | Slice RAM address control register ($020E_{16}$)<br>Slice RAM data control register ($0210_{16}$) |
| VBIRAM | VBI encode data setting. | VBIRAM address control register ($0212_{16}$)<br>VBIRAM data control register ($0214_{16}$) |
| Expansion register | This register controls OSD display, data slicer and VBI encoder. | OSD register address control register ($0216_{16}$)<br>OSD register data control register ($0218_{16}$) |

### 2.15.3 Display RAM

Set 1 screen (40 characters × 25 lines) display character.

1 character display character setting is consists is 2 addresses (even address 16 bits + odd address 8 bits), set characters available, character color, blinking, character background color, SYRAM available and SYRAM color. Display RAM composition is shown in Table 2.15.3.

**Table 2.15.3 Display RAM composition**

| Address (CA10 to CA0) | CD15 | CD14 | CD13 | CD12 | CD11 | CD10 | CD9 | CD8 | CD7 | CD6 | CD5 | CD4 | CD3 | CD2 | CD1 | CD0 | Remarks |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $000_{16}$ | 0 | BB | BG | BR | BLINK | CB | CG | CR | C7 | C6 | C5 | C4 | C3 | C2 | C1 | C0 | Character setting of the 0th |
| | | Character background | | | Blinking | Character color | | | Font RAM character code | | | | | | | | |
| $001_{16}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SB | SG | SR | 0 | SYC3 | SYC2 | SYC1 | SYC0 | character of the 0th line. |
| | | | | | | | | | SYRAM character code(Note) | | | | SYRAM character code(Note) | | | | |
| $002_{16}$ | 0 | BB | BG | BR | BLINK | CB | CG | CR | C7 | C6 | C5 | C4 | C3 | C2 | C1 | C0 | Character setting of the first |
| $003_{16}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SB | SG | SR | 0 | SYC3 | SYC2 | SYC1 | SYC0 | character of the 0th line. |
| $004_{16}$ | 0 | BB | BG | BR | BLINK | CB | CG | CR | C7 | C6 | C5 | C4 | C3 | C2 | C1 | C0 | Character setting of the second |
| $005_{16}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SB | SG | SR | 0 | SYC3 | SYC2 | SYC1 | SYC0 | character of the 0th line. |
| $006_{16}$ ⋮ $7CB_{16}$ | ⋮ | | | | | | | | | | | | | | | | Character setting of the third character of the 0th line. ⋮ Character setting of the 37th character of the 24th line. |
| $7CC_{16}$ | 0 | BB | BG | BR | BLINK | CB | CG | CR | C7 | C6 | C5 | C4 | C3 | C2 | C1 | C0 | Character setting of the 38th |
| $7CD_{16}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SB | SG | SR | 0 | SYC3 | SYC2 | SYC1 | SYC0 | character of the 24th line. |
| $7CE_{16}$ | 0 | BB | BG | BR | BLINK | CB | CG | CR | C7 | C6 | C5 | C4 | C3 | C2 | C1 | C0 | Character setting of the 39th |
| $7CF_{16}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SB | SG | SR | 0 | SYC3 | SYC2 | SYC1 | SYC0 | Character of the 24th line. |

Note: SYRAM setting bit is G1character setting bit when set $00_{16}$ to font RAM character code.
(Refer to Teletext G1 character display for detal)

Set accessing address (CA10 to CA0) (shown in Table 2.15.3) to display RAM address control register (address $0202_{16}$), and write data (CD15 to CD0) from display RAM data control register (address $0204_{16}$). After data accessing fixed, display RAM address control register iuncrements address automatically. Then, writing next address data is possible.

Display RAM bit composition is shown in Figure 2.15.3, Display RAM access registers are shown in Figure 2.15.4, Display RAM data access block diagram is shown in Figure 2.15.5, and Address map is shown in Figure 2.15.6 and Figure 2.15.7.

Display RAM bit composition

| | CD15 | CD14 | CD13 | CD12 | CD11 | CD10 | CD9 | CD8 | CD7 | CD6 | CD5 | CD4 | CD3 | CD2 | CD1 | CD0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Even address | — | BB | BG | BR | BLINK | CB | CG | CR | C7 | C6 | C5 | C4 | C3 | C2 | C1 | C0 |
| Odd address | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SB | SG | SR | — | SYC3 | SYC2 | SYC1 | SYC0 |

| Bit | Bit name | Function |
|---|---|---|
| C0 | Font RAM bit | Set font RAM character code.<br><br>Character code $00_{16}$ is corresponded to teletext G1 character. (Refer to "Teletext G1 character display".) |
| C1 | | |
| C2 | | |
| C3 | | |
| C4 | | |
| C5 | | |
| C6 | | |
| C7 | | |
| CR | Character color bit | Set color code of font RAM character color (Note 2) |
| CG | | |
| CB | | |
| BLINK | Blinking bit | 0 : Do not blink<br>1 : Blink |
| BR | Character background color bit | Set color code of font RAM character background color (Note 2) |
| BG | | |
| BB | | |
| — | — | Must always be set to "0". |
| SYC0 | SYRAM bit | Set SYRAM character code which composes to font RAM setting by C0 to C7. When it is not composed, set character code $F_{16}$.<br>These bit are teletext G1 character setting bit when C7 to C0 is $00_{16}$ setting. |
| SYC1 | | |
| SYC2 | | |
| SYC3 | | |
| — | — | Must always be set to "0". |
| SR | SYRAM color bit | Set color code of SYRAM color (Note 2).<br>These bit are teletext G1 character setting bit when C7 to C0 is $00_{16}$ setting. |
| SG | | |
| SB | | |

Notes 1. The contents of display RAM is indefinite at reset.
2. Color code setting

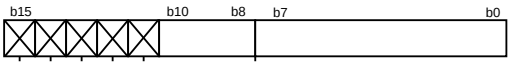
| Color code | | | Color setting |
|---|---|---|---|
| B | G | R | |
| 0 | 0 | 0 | Black |
| 0 | 0 | 1 | Red |
| 0 | 1 | 0 | Green |
| 0 | 1 | 1 | Yellow |
| 1 | 0 | 0 | Blue |
| 1 | 0 | 1 | Magenta |
| 1 | 1 | 0 | Cyan |
| 1 | 1 | 1 | White |

Color code (R, G, B) is corresponded to character color bit (CR, CG, CB), Character backgroud color bit (BR,BG, BB) and SYRAM color bit (SR, SG, SB).

Refer to expansion register composition (Address $0A_{16}$) for color setting at expansion register GRYON = "1".

**Figure 2.15.3 Display RAM bit composotion**

Display RAM address control register

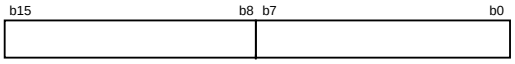



| Symbol | Address | When reset |
|---|---|---|
| CA | $0202_{16}$ | $0000_2$ |

| Function | Setting possible value | R | W |
|---|---|---|---|
| Specify accessing display RAM address | $000_{16}$ to $7CF_{16}$ | X | O |
| Nothing is assigned. In an attempt to write to these bits, write "0". The value, if read, turns out to be indeterminated. | | – | – |

Note : When access to display RAM, must be set display RAM address at first, then use display RAM data control register (0204 $_{16}$).
Display RAM address control register increments by accessing display RAM data control register. So, it is not neccesary to setting the next display RAM address.

Display RAM data control register




| Symbol | Address | When reset |
|---|---|---|
| CD | $0204_{16}$ | $0000_2$ |

| Function | Setting possible value | R | W |
|---|---|---|---|
| Write and read out the data of display RAM which is specified by display RAM address control register (address $0202_{16}$) | $0000_{16}$ to $7FFF_{16}$ | O | O |

Note : Data access must be 16-bit unit. 8-bit unit access is disable.

**Figure 2.15.4 Display RAM access registers**

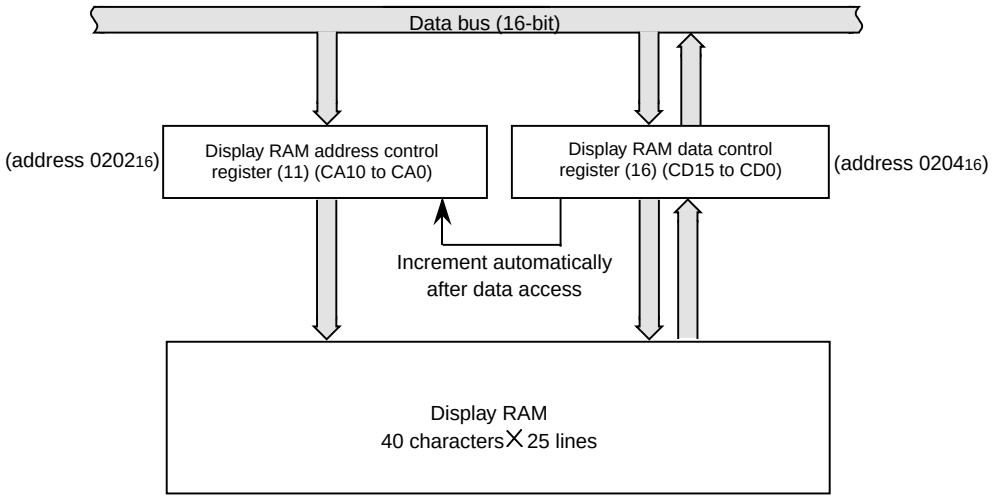
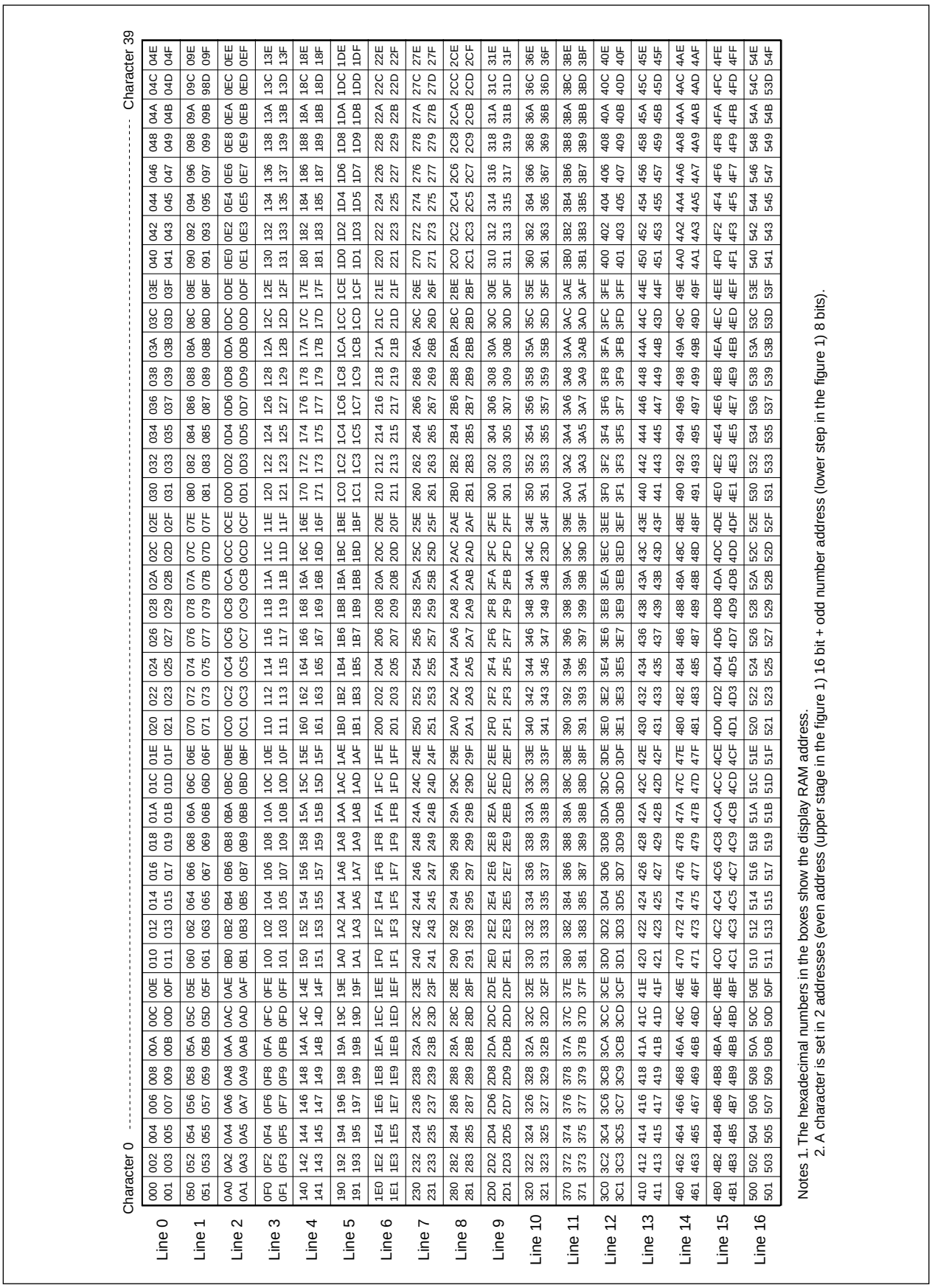
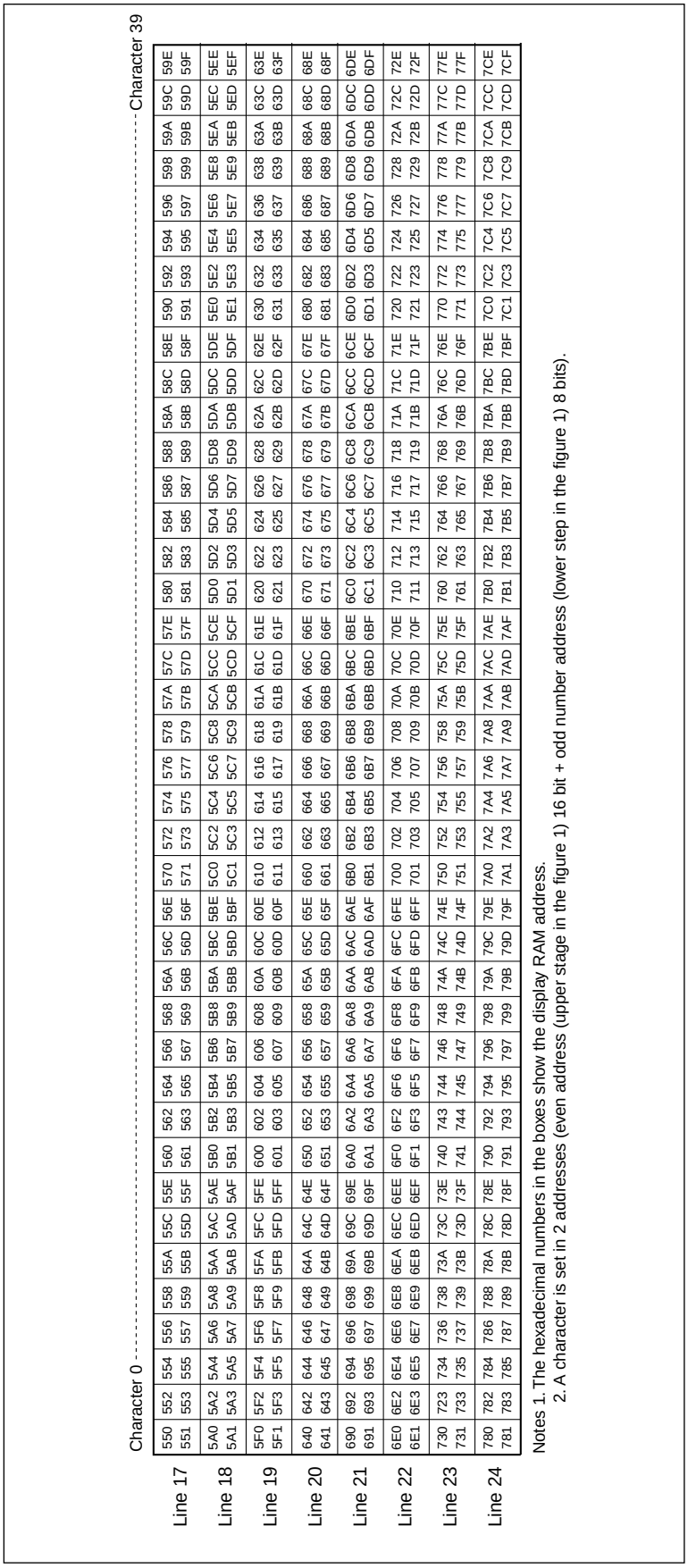



**Figure 2.15.5 Display RAM access block diagram**




Notes 1. The hexadecimal numbers in the boxes show the display RAM address.
2. A character is set in 2 addresses (even address (upper stage in the figure 1) 16 bit + odd number address (lower step in the figure 1) 8 bits).

**Figure 2.15.6 Address map 1 (continued)**




Notes 1. The hexadecimal numbers in the boxes show the display RAM address.
2. A character is set in 2 addresses (even address (upper stage in the figure 1) 16 bit + odd number address (lower step in the figure 1) 8 bits).

**Figure 2.15.7 Address map 2**

**Teletext G1 character display**

Can display teletext G1 character by setting character code $00_{16}$ to font RAM bit (C7 to C0) of display RAM. SYRAM setting is invalid when set $00_{16}$ to font RAM bit (C7 to C0), set G1 character by G1 character bit (G0 to G5) and G1character form bit(G6). At the time, set 0 to all addresses of font RAM code $00_{16}$ (font RAM addresses $000_{16}$ to $009_{16}$).

Display RAM composition at G1 character display is shown in Figure 2.15.8.

| | CD15 | CD14 | CD13 | CD12 | CD11 | CD10 | CD9 | CD8 | CD7 | CD6 | CD5 | CD4 | CD3 | CD2 | CD1 | CD0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Even address | — | BB | BG | BR | BLINK | CB | CG | CR | 0 (C7) | 0 (C6) | 0 (C5) | 0 (C4) | 0 (C3) | 0 (C2) | 0 (C1) | 0 (C0) |
| Odd address | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | G6 (SB) | G5 (SG) | G4 (SR) | — | G3 (SYC3) | G2 (SYC2) | G1 (SYC1) | G0 (SYC0) |

| Bit | Bit name | Function |
|---|---|---|
| 0(C0) | Font RAM bit | Set $00_{16}$ when display teletext G1 character.<br>At the time , set space in font RAM($00_{16}$). |
| 0(C1) | | |
| 0(C2) | | |
| 0(C3) | | |
| 0(C4) | | |
| 0(C5) | | |
| 0(C6) | | |
| 0(C7) | | |
| CR | G1 character color bit | Set color code of G1 character color. |
| CG | | |
| CB | | |
| BLINK | Blinking bit | 0 : Do not blink<br>1 : Blink |
| BR | G1 character background color bit | Set color code of G1 character background color. |
| BG | | |
| BB | | |
| — | — | Must always be set to "0". |
| G0(SYC0) | G1 character bit(1) | Set G1 character by G0 to G5.<br>(Refer to the next page.) |
| G1(SYC1) | | |
| G2(SYC2) | | |
| G3(SYC3) | | |
| — | — | Must always be set to "0". |
| G4(SR) | G1 character bit(1) | Set G1 character by G0 to G5.<br>(Refer to the next page.) |
| G5(SG) | | |
| G6(SB) | G1 character form bit | 0 : Contiguous form (Refer to the next page)<br>1 : Separated form |

**Figure 2.15.8 Display RAM bit composotion(at G1 character displaying )**

**G1 character composition**

Set G1 character by display RAM G1 character bit (G0 to G5) and G1 character form bit(G6). G1 character composition is shown in Figure 2.15.9. G1 character is divided to 6 blocks (refer to Figure 2.15.9), and set character by G0 to G5 in each block. Also, G1 character form is set by G6.

Can display 64 patterns G1 character by using G0 to G5. G1 character composition is shown in Figure 2.15.10.

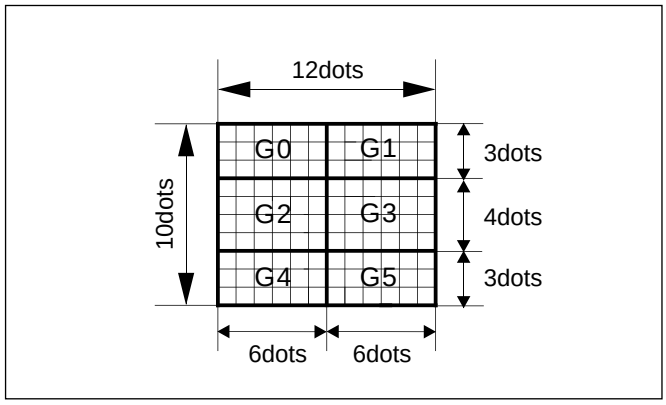



**Figure 2.15.9 G1 character composition**

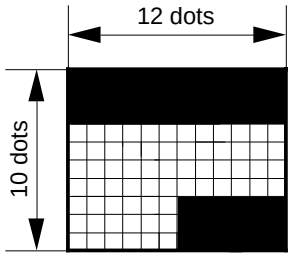
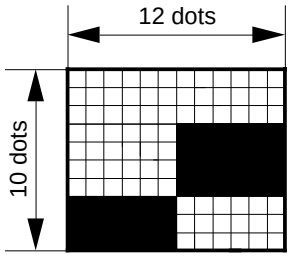
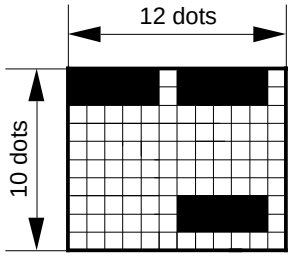
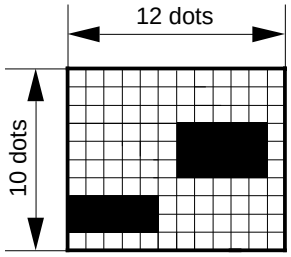
| | Example1:G0,G1,G5=1,G2,G3,G4=0 | Example2:G3,G4=1,G0,G1,G2,G5=0 |
|---|---|---|
| G6=0 (Contiguous form) |   |   |
| G6=1 (Separated form) |   |   |

**Figure 2.15.10 G1 character setting**

Set 0 to G0 to G5 when use font RAM code $00_{16}$ as normal character.

However, SYRAM can not be displayed.

### 2.15.4 Font RAM

Character font composited horizontal direction 12 dots × vertical direction 10 dots is set to font RAM code $00_{16}$ to $FF_{16}$ (255 available, $7F_{16}$:blank code).

1 character setting is 10 address composite (12-bit × 10 addresses).

Setting character is displayed by specifying font RAM code to font RAM bit of display RAM. Font RAM code $00_{16}$ is corresponds to Teletext G1 character. Then, font RAM code $7F_{16}$ is fixed by blank, character font setting to this code is disable. Font RAM composition is shown in Table 2.15.4.

**Table 2.15.4 Font RAM composition**

| Font RAM addresses (FA11 to FA0) | FD11 | FD10 | FD9 | FD8 | FD7 | FD6 | FD5 | FD4 | FD3 | FD2 | FD1 | FD0 | Remarks |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $000_{16}$ | F0B | F0A | F09 | F08 | F07 | F06 | F05 | F04 | F03 | F02 | F01 | F00 | Font RAM code ($00_{16}$) |
| $001_{16}$ | F1B | F1A | F19 | F18 | F17 | F16 | F15 | F14 | F13 | F12 | F11 | F10 | |
| $002_{16}$ | F2B | F2A | F29 | F28 | F27 | F26 | F25 | F24 | F23 | F22 | F21 | F20 | |
| $003_{16}$ | F3B | F3A | F39 | F38 | F37 | F36 | F35 | F34 | F33 | F32 | F31 | F30 | |
| $004_{16}$ | F4B | F4A | F49 | F48 | F47 | F46 | F45 | F44 | F43 | F42 | F41 | F40 | |
| $005_{16}$ | F5B | F5A | F59 | F58 | F57 | F56 | F55 | F54 | F53 | F52 | F51 | F50 | |
| $006_{16}$ | F6B | F6A | F69 | F68 | F67 | F66 | F65 | F64 | F63 | F62 | F61 | F60 | |
| $007_{16}$ | F7B | F7A | F79 | F78 | F77 | F76 | F75 | F74 | F73 | F72 | F71 | F70 | |
| $008_{16}$ | F8B | F8A | F89 | F88 | F87 | F86 | F85 | F84 | F83 | F82 | F81 | F80 | |
| $009_{16}$ | F9B | F9A | F99 | F98 | F97 | F96 | F95 | F94 | F93 | F92 | F91 | F90 | |
| $00A_{16}$ ⋮ $00F_{16}$ | Unused area | | | | | | | | | | | | |
| $010_{16}$ ⋮ $019_{16}$ | F0B ⋮ F9B | F0A ⋮ F9A | F09 ⋮ F99 | F08 ⋮ F98 | F07 ⋮ F97 | F06 ⋮ F96 | F05 ⋮ F95 | F04 ⋮ F94 | F03 ⋮ F93 | F02 ⋮ F92 | F01 ⋮ F91 | F00 ⋮ F90 | Font RAM code ($01_{16}$) |
| $020_{16}$ ⋮ $FD9_{16}$ | ⋮ | | | | | | | | | | | | Font RAM code ($02_{16}$) ⋮ Font RAM code ($FD_{16}$) |
| $FE0_{16}$ ⋮ $FE9_{16}$ | F0B ⋮ F9B | F0A ⋮ F9A | F09 ⋮ F99 | F08 ⋮ F98 | F07 ⋮ F97 | F06 ⋮ F96 | F05 ⋮ F95 | F04 ⋮ F94 | F03 ⋮ F93 | F02 ⋮ F92 | F01 ⋮ F91 | F00 ⋮ F90 | Font RAM code ($FE_{16}$) |
| $FF0_{16}$ ⋮ $FF9_{16}$ | F9B ⋮ F9B | F0A ⋮ F9A | F09 ⋮ F99F | F08 ⋮ F98F | F07 ⋮ F97 | F06 ⋮ F96 | F05 ⋮ F95 | F04 ⋮ F94 | F03 ⋮ F93 | F02 ⋮ F92 | F01 ⋮ F91 | F00 ⋮ F90 | Font RAM code ($FF_{16}$) |

For accessing to font RAM data, set accessing address (FA11 to FA0) (shown in Table 2.15.4) to font RAM address control register ($0206_{16}$). Then write data (FD11 to FD0) by font RAM data control register ($0208_{16}$. After data accessing fixed, font RAM address control register increments address automatically. Then, next address data writing is possible. Do not access to unused area (addresses x$A_{16}$ to x$F_{16}$) of each Font RAM codes. But, when write data in succession, jump unused area and increments address automatically. (ex. increment automatically from address $009_{16}$ to $010_{16}$).

Font composition is shown in Figure 2.15.11, Setting example is shown in Figure 2.15.12, Font RAM access registers are shown in Figure 2.15.13 and Font RAM access block diagram is shown in Figure 2.15.14.

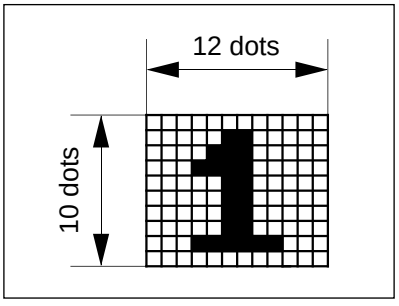
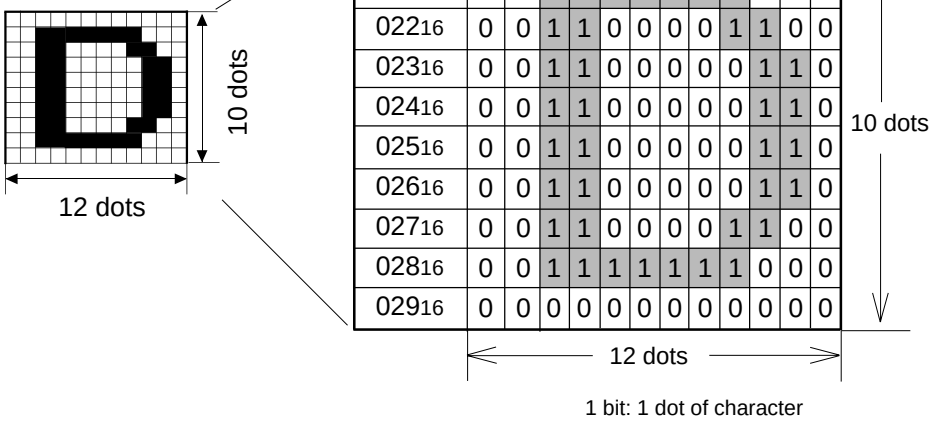



**Figure 2.15.11 Font composition**

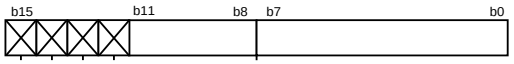
**Example of font RAM code $02_{16}$ ..... Set character by addresses $020_{16}$ to $029_{16}$ data setting.**




| Address | FD | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FA | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| $020_{16}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $021_{16}$ | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| $022_{16}$ | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| $023_{16}$ | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| $024_{16}$ | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| $025_{16}$ | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| $026_{16}$ | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| $027_{16}$ | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| $028_{16}$ | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| $029_{16}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 2.15.12 Setting example of font RAM**

Font RAM address control register

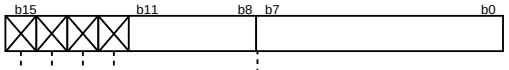



| Symbol | Address | When reset |
|---|---|---|
| FA | $0206_{16}$ | $0000_2$ |

| Function | Setting possible value | R | W |
|---|---|---|---|
| Specify accessing font RAM address | $000_{16}$ to $FF9_{16}$ | X | O |
| Nothing is assigned. In an attempt to write to these bits, write "0". The value, if read, turns out to be indeterminated. | | – | – |

Note : When access to font RAM, must be set font RAM at first, then use font RAM data control register ($0208_{16}$).
Font RAM address control register increments by accessing font RAM data control register. So, it is not neccesary to setting the next font RAM address.

Font RAM data control register




| Symbol | Address | When reset |
|---|---|---|
| FD | $0208_{16}$ | $0000_2$ |

| Function | Setting possible value | R | W |
|---|---|---|---|
| Write and read out the data of font RAM which is specified by font RAM address control register (address $0206_{16}$) | $000_{16}$ to $FFF_{16}$ | O | O |
| Nothing is assigned. In an attempt to write to these bits, write "0". The value, if read, turns out to be indeterminated. | | – | – |

Note : Data access must be 16-bit unit. 8-bit unit access is disable.

**Figure 2.15.13 Font RAM access registers**

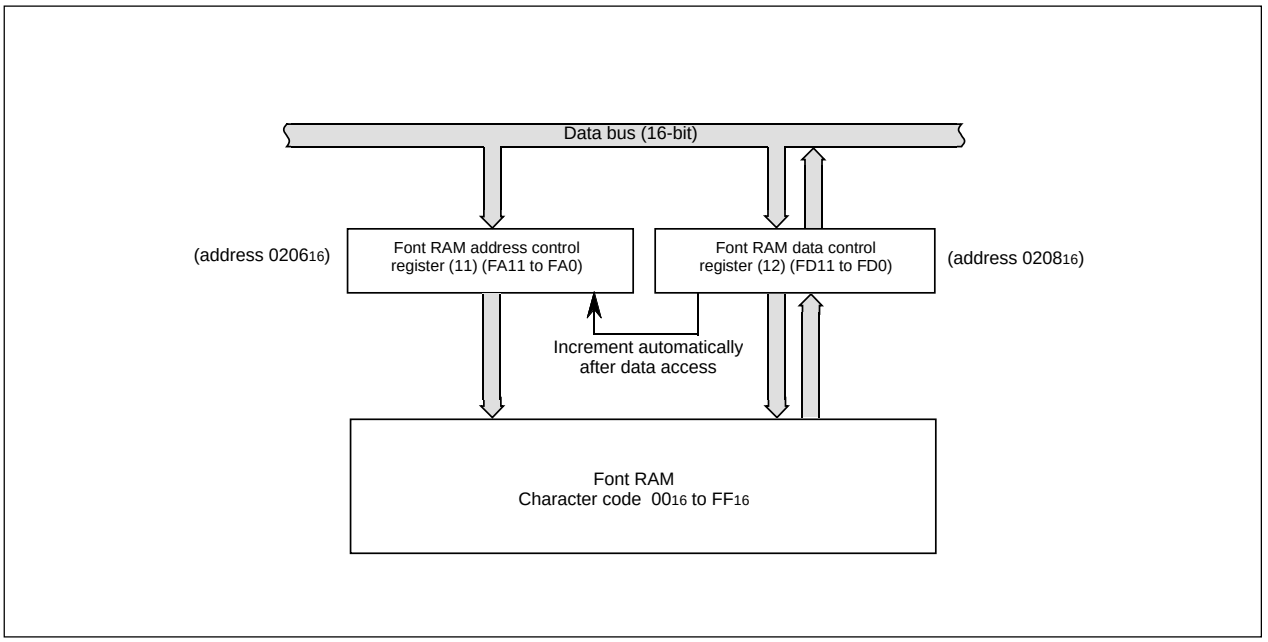



**Figure 2.15.14 Font RAM access block diagram**

### 2.15.5 SYRAM

Character font composite horizontal direction 12 dots × vertical direction 10 dots is set to SYRAM code $0_{16}$ to $E_{16}$ (15 available).

Setting composite character is composed to font RAM by specifying SYRAM code to SYRAM bit of display RAM. Then, SYRAM code $F_{16}$ is fixed by blank, character font setting to this code is disable. Use $F_{16}$ when SYRAM is not composed to character.

SYRAM composite is shown in Table 2.15.5.

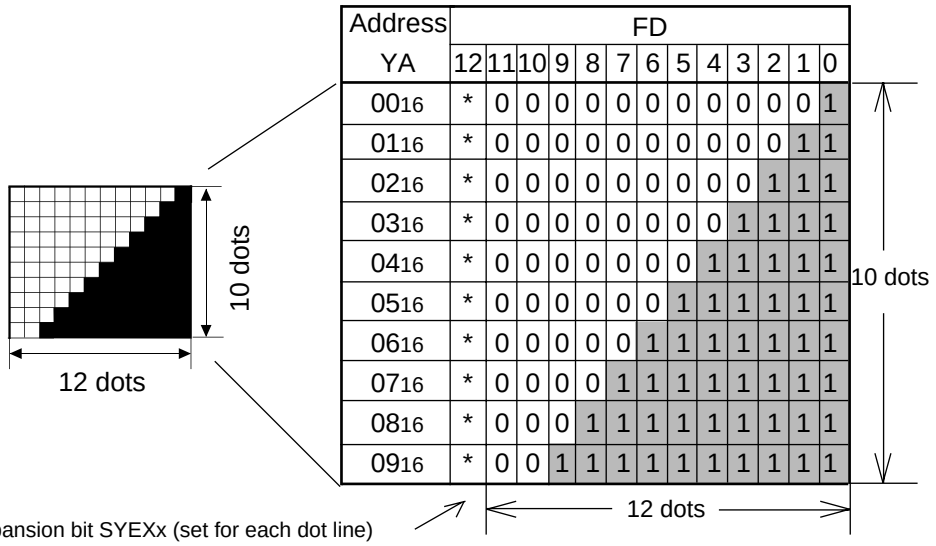
**Table 2.15.5 SYRAM composition**

| Font RAM addresses (FA10 to FA0) | YD12 | YD11 | YD10 | YD9 | YD8 | YD7 | YD6 | YD5 | YD4 | YD3 | YD2 | YD1 | YD0 | Remarks |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $00_{16}$ | SYEX0 | SY0B | SY0A | SY09 | SY08 | SY07 | SY06 | SY05 | SY04 | SY03 | SY02 | SY01 | SY00 | SYRAM code ($0_{16}$) |
| $01_{16}$ | SYEX1 | SY1B | SY1A | SY19 | SY18 | SY17 | SY16 | SY15 | SY14 | SY13 | SY12 | SY11 | SY10 | |
| $02_{16}$ | SYEX2 | SY2B | SY2A | SY29 | SY28 | SY27 | SY26 | SY25 | SY24 | SY23 | SY22 | SY21 | SY20 | |
| $03_{16}$ | SYEX3 | SY3B | SY3A | SY39 | SY38 | SY37 | SY36 | SY35 | SY34 | SY33 | SY32 | SY31 | SY30 | |
| $04_{16}$ | SYEX4 | SY4B | SY4A | SY49 | SY48 | SY47 | SY46 | SY45 | SY44 | SY43 | SY42 | SY41 | SY40 | |
| $05_{16}$ | SYEX5 | SY5B | SY5A | SY59 | SY58 | SY57 | SY56 | SY55 | SY54 | SY53 | SY52 | SY51 | SY50 | |
| $06_{16}$ | SYEX6 | SY6B | SY6A | SY69 | SY68 | SY67 | SY66 | SY65 | SY64 | SY63 | SY62 | SY61 | SY60 | |
| $07_{16}$ | SYEX7 | SY7B | SY7A | SY79 | SY78 | SY77 | SY76 | SY75 | SY74 | SY73 | SY72 | SY71 | SY70 | |
| $08_{16}$ | SYEX8 | SY8B | SY8A | SY89 | SY88 | SY87 | SY86 | SY85 | SY84 | SY83 | SY82 | SY81 | SY80 | |
| $09_{16}$ | SYEX9 | SY9B | SY9A | SY99 | SY98 | SY97 | SY96 | SY95 | SY94 | SY93 | SY92 | SY91 | SY90 | |
| $0A_{16}$ ⋮ $0F_{16}$ | Unused area | | | | | | | | | | | | | |
| $10_{16}$ ⋮ $19_{16}$ | SYEX0 ⋮ SYEX9 | SY0B ⋮ SY9B | SY0A ⋮ SY9A | SY09 ⋮ SY99 | SY08 ⋮ SY98 | SY07 ⋮ SY97 | SY06 ⋮ SY96 | SY05 ⋮ SY95 | SY04 ⋮ SY94 | SY03 ⋮ SY93 | SY02 ⋮ SY92 | SY01 ⋮ SY91 | SY00 ⋮ SY90 | SYRAM code ($1_{16}$) |
| $20_{16}$ ⋮ $C9_{16}$ | ⋮ | | | | | | | | | | | | | SYRAM code ($2_{16}$) ⋮ SYRAM code ($C_{16}$) |
| $D0_{16}$ ⋮ $D9_{16}$ | SYEX0 ⋮ SYEX9 | SY0B ⋮ SY9B | SY0A ⋮ SY9A | SY09 ⋮ SY99 | SY08 ⋮ SY98 | SY07 ⋮ SY97 | SY06 ⋮ SY96 | SY05 ⋮ SY95 | SY04 ⋮ SY94 | SY03 ⋮ SY93 | SY02 ⋮ SY92 | SY01 ⋮ SY91 | SY00 ⋮ SY90 | SYRAM code ($D_{16}$) |
| $E0_{16}$ ⋮ $E9_{16}$ | SYEX0 ⋮ SYEX9 | SY9B ⋮ SY9B | SY0A ⋮ SY9A | SY09 ⋮ SY99 | SY08 ⋮ SY98 | SY07 ⋮ SY97 | SY06 ⋮ SY96 | SY05 ⋮ SY95 | SY04 ⋮ SY94 | SY03 ⋮ SY93 | SY02 ⋮ SY92 | SY01 ⋮ SY91 | SY00 ⋮ SY90 | SYRAM code ($E_{16}$) |

For accessing to SYRAM data, set accessing address (YA7 to YA0) (shown in Table 2.15.5) to SYRAM address control register ($020A_{16}$). Then write data (YD12 to YD0) by SYRAM data control register ($020C_{16}$). When end the accessing, SYRAM address control register increments address automatically. Then, next address data writing is possible. Do not access to unused area (addresses $xA_{16}$ to $xF_{16}$) of each SYRAM codes. But, when write data in succession, jump unused area and increments address automatically. (ex. increment automatically from address $09_{16}$ to $10_{16}$).

Setting example is shown in Figure 2.15.15, SYRAM access registers are shown in Figure 2.15.16 and SYRAM access block diagram is shown in Figure 2.15.17.

**Example of SYRAM code $0_{16}$ ..... Set character by addresses $00_{16}$ to $09_{16}$ data setting.**



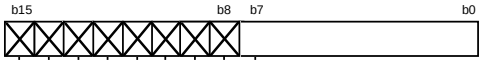

| Address | FD | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| YA | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| $00_{16}$ | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| $01_{16}$ | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| $02_{16}$ | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| $03_{16}$ | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| $04_{16}$ | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| $05_{16}$ | * | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| $06_{16}$ | * | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| $07_{16}$ | * | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| $08_{16}$ | * | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| $09_{16}$ | * | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Color expansion bit SYEXx (set for each dot line)
The HIDE register (address $0D_{16}$) becomes valid for only the dot line where * = 1.
For details, refer to the next section, “Compositing font RAM and SYRAM.”

1 bit: 1 dot of character

**Figure 2.15.15 Setting example of SYRAM**

SYRAM address control register

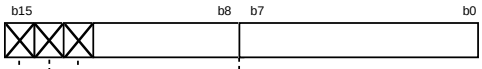



| Symbol | Address | When reset |
|---|---|---|
| YA | $020A_{16}$ | $0000_{16}$ |

| Function | Setting possible value | R | W |
|---|---|---|---|
| Specify accessing SYRAM address | $00_{16}$ to $E9_{16}$ | X | O |
| Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminated. | | – | – |

Note : When access to SYRAM, must be set SYRAM at first, then use SYRAM data control register ($020C_{16}$).
SYRAM address control register increments by accessing SYRAM data control register. So, it is not neccesary to setting the next SYRAM address.

SYRAM data control register




| Symbol | Address | When reset |
|---|---|---|
| YD | $020C_{16}$ | $0000_{16}$ |

| Function | Setting possible value | R | W |
|---|---|---|---|
| Write and read out the data of SYRAM which is specified by SYRAM address control register (address $020A_{16}$) | $0000_{16}$ to $1FFF_{16}$ | O | O |
| Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminated. | | – | – |

Note : Data access must be 16-bit unit. 8-bit unit access is disable.

**Figure 2.15.16 SYRAM access registers**

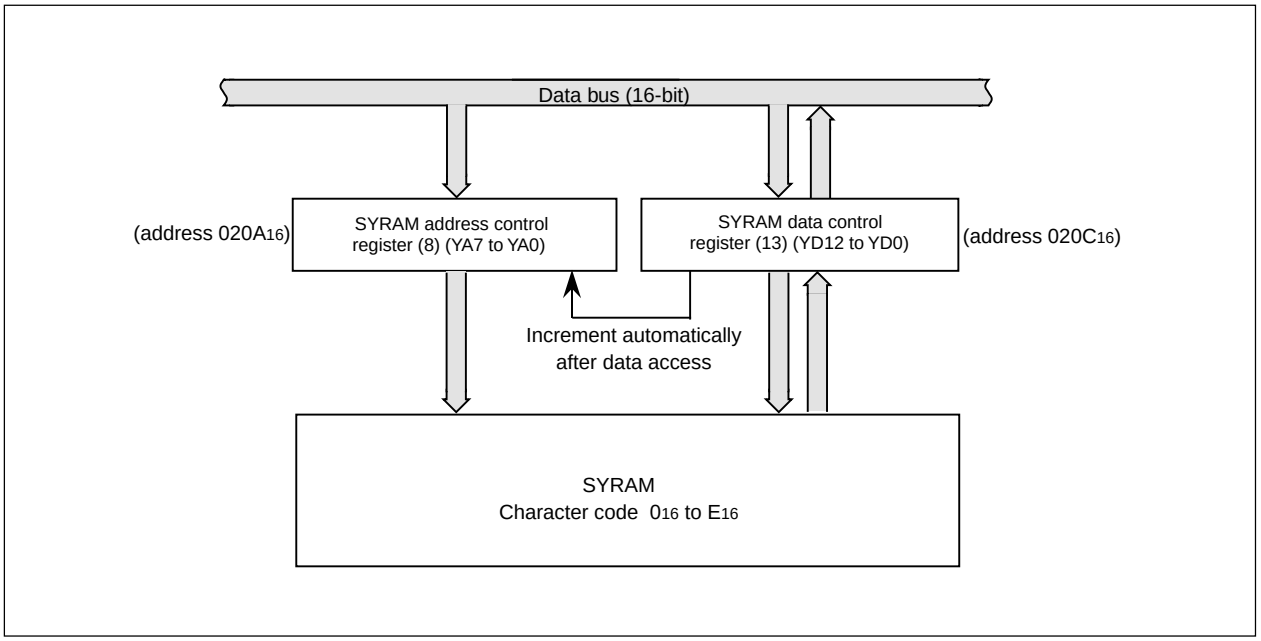



**Figure 2.15.17 SYRAM access block diagram**

**Compositing font RAM and SYRAM**

Can composite characters in font RAM with SYRAM.

The compositing method is determined by the SYEXx color expansion bit and the HIDE register (address $0D_{16}$).

For dot lines where SYEXx = 0, the SYRAM color is set by the display RAM's SR, SG, and SB irrespective of the HIDE register's content.

If the HIDE register's content is 0, the SYRAM color for dot lines where SYEXx = 1 is set by the registers LINER, LINEG, and LINEB (address $08_{16}$).

If the HIDE register's content is 1, the font RAM part of the dot lines where SYEXx = 1 is overwritten in HIDE mode with colors set by the registers LINER, LINEG, and LINEB irrespective of the font RAM's content and color. The color of the SYRAM part is set by the display RAM's SR, SG, and SB as in the case of dot lines where SYEXx = 0.

Figure 2.15.18 shows an example for each instance of compositing.

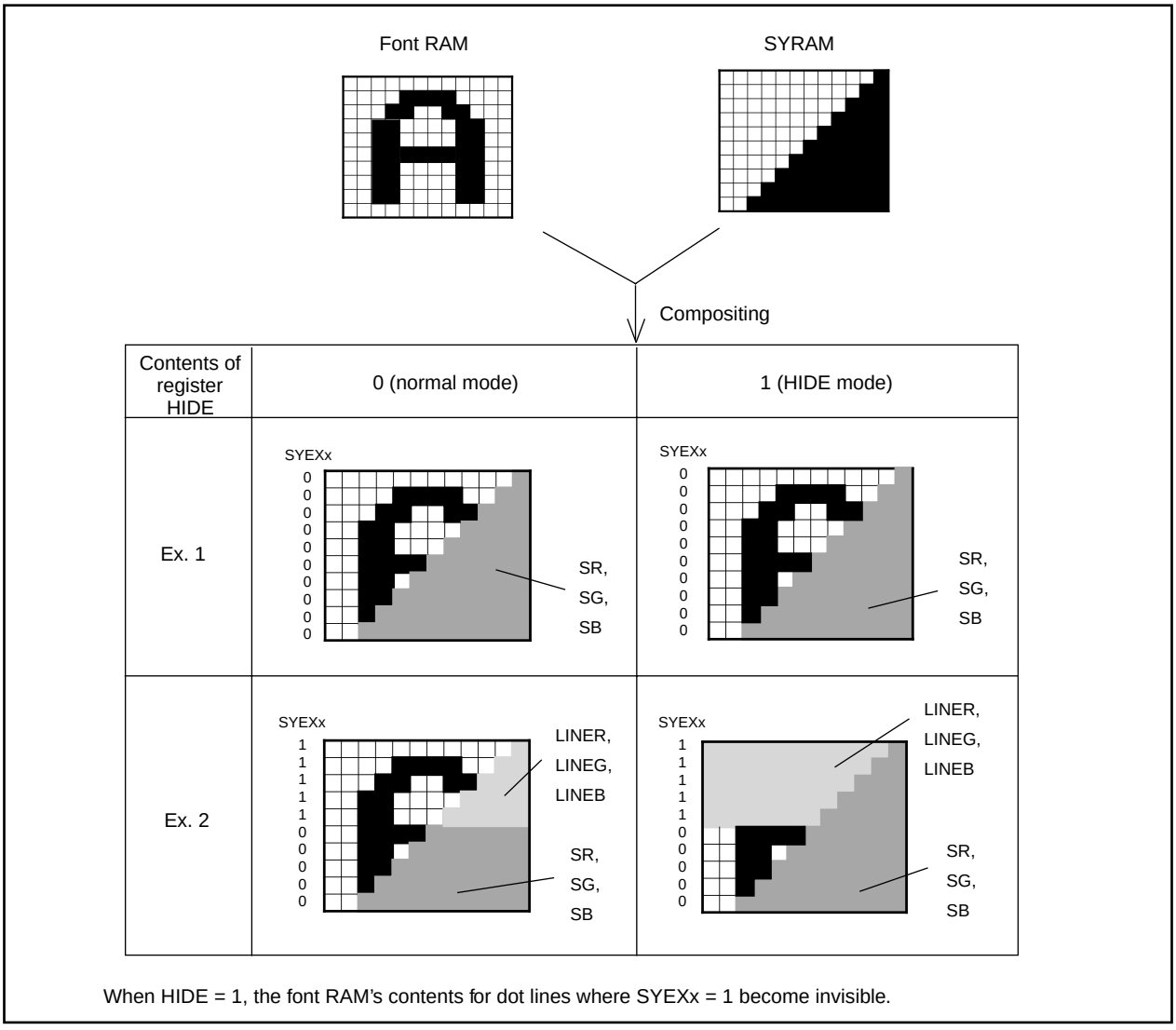



**Figure 2.15.18 Compositing example**

### 2.15.6 Slice RAM

Store 18-line slice data. There are 3 types of Slice data : PDC, VPS and VBI. All data are stored to addresses which corresponds to slicing line (ex. 22 line' data is stored to addresses $200_{16}$ to $217_{16}$ ). 24 addresses (SR00x to SR17x) are prepared for 1 line, slice data is stored in order from LSB side. Then, slice datas and field information are stored to the top address of each line.

Slice RAM composite is shown in Table 2.15.6.

**Table 2.15.6 Slice RAM composition**

| Slice RAM addresses (SA9 to SA0) | SD15 | SD14 | SD13 | SD12 | SD11 | SD10 | SD9 | SD8 | SD7 | SD6 | SD5 | SD4 | SD3 | SD2 | SD1 | SD0 | Remarks |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $000_{16}$ | SR00F | SR00E | SR00D | SR00C | SR00B | SR00A | SR009 | SR008 | SR007 | SR006 | SR005 | SR004 | SR003 | SR002 | SR001 | SR000 | 6th line or 318th line |
| $001_{16}$ | SR01F | SR01E | SR01D | SR01C | SR01B | SR01A | SR019 | SR018 | SR017 | SR016 | SR015 | SR014 | SR013 | SR012 | SR011 | SR010 | slice data |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | |
| $016_{16}$ | SR16F | SR16E | SR16D | SR16C | SR16B | SR16A | SR169 | SR168 | SR167 | SR166 | SR165 | SR164 | SR163 | SR162 | SR161 | SR160 | |
| $017_{16}$ | SR17F | SR17E | SR17D | SR17C | SR17B | SR17A | SR179 | SR178 | SR177 | SR176 | SR175 | SR174 | SR173 | SR172 | SR171 | SR170 | |
| $018_{16}$ ⋮ $01F_{16}$ | Unused area | | | | | | | | | | | | | | | | |
| $020_{16}$ | SR00F | SR00E | SR00D | SR00C | SR00B | SR00A | SR009 | SR008 | SR007 | SR006 | SR005 | SR004 | SR003 | SR002 | SR001 | SR000 | 7th line or 319 th line |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | slice data |
| $037_{16}$ | SR17F | SR17E | SR17D | SR17C | SR17B | SR17A | SR179 | SR178 | SR177 | SR176 | SR175 | SR174 | SR173 | SR172 | SR171 | SR170 | |
| $040_{16}$ ⋮ $1F7_{16}$ | ⋮ | | | | | | | | | | | | | | | | 8th line to 21th line or 320th line to 333 line slice data |
| $200_{16}$ | SR00F | SR00E | SR00D | SR00C | SR00B | SR00A | SR009 | SR008 | SR007 | SR006 | SR005 | SR004 | SR003 | SR002 | SR001 | SR000 | 22th line or 334th line |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | slice data |
| $217_{16}$ | SR17F | SR17E | SR17D | SR17C | SR17B | SR17A | SR179 | SR178 | SR177 | SR176 | SR175 | SR174 | SR173 | SR172 | SR171 | SR170 | |
| $220_{16}$ | SR00F | SR00E | SR00D | SR00C | SR00B | SR00A | SR009 | SR008 | SR007 | SR006 | SR005 | SR004 | SR003 | SR002 | SR001 | SR000 | 23th line or 335th line |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | slice data |
| $237_{16}$ | SR17F | SR17E | SR17D | SR17C | SR17B | SR17A | SR179 | SR178 | SR177 | SR176 | SR175 | SR174 | SR173 | SR172 | SR171 | SR170 | |

For accessing to slice RAM data, set accessing address (SA9 to SA0) (shown in Table 2.15.6) to slice RAM address control register (address $020E_{16}$ ). Then read out data from slice RAM data control register (address $0210_{16}$ ). When end the data reading, slice RAM address control register increments address automatically. Then, next address data reading is possible. Do not access to unused area of each character codes. Must set address to each line because unused area has no address' automatically increment.

Slice RAM bit composition is shown in Figure 2.15.19, Slice RAM access registers are shown in Figure 2.15.20 and Slice RAM access block diagram is shown in Figure 2.15.21.

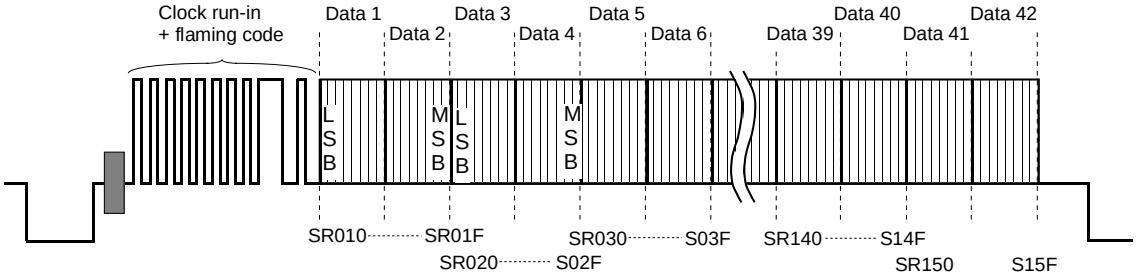
**The each head address of the address is corresponded to slicing line has stored next slice information.**

| | SR00F to SR004 | SR003 | SR002 | SR001 | SR000 |
|---|---|---|---|---|---|
| PDC | 0 | field * (Note) | 0 | 0 | 1 |
| VPS | 0 | field * (Note) | 0 | 1 | 0 |
| VBI | 0 | field * (Note) | 1 | 0 | 0 |
| Other | 0 | 0 | 0 | 0 | 0 |

Note : * the first field : 1
the second field : 0

(1) PDC

In case of the PDC data, 16 bits (2 data) are stored for the 1 address from the LSB side.



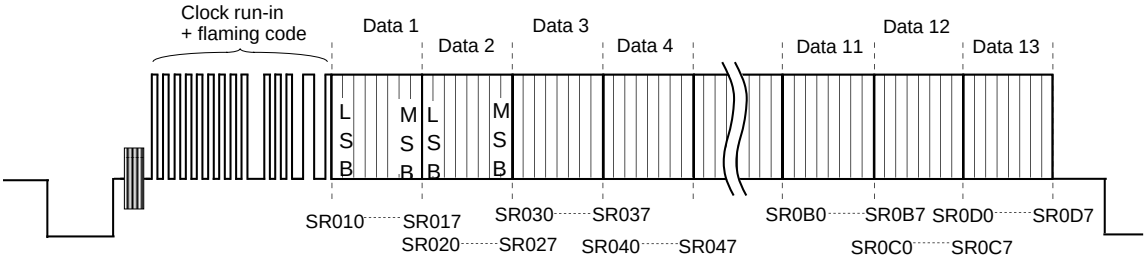

SR16x to SR17x are unused area.

(2) VPS

In case of the VPS data, 8 bits (a data) are stored for an address from the LSB side.

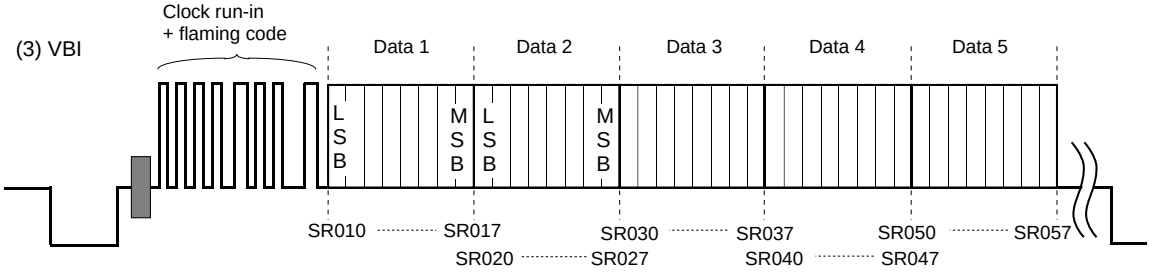
Low-order 8 bits stores the slice data. And, high-order 8 bits become warning bit, when the send data is not recognized as bi-phase type.

The case of bi-phase data ="1,0" or "0,1" (the bi-phase type) becomes "0" for this warning bit, and it becomes "1" in bi-phase data ="0,0" or "1,1" (it is not the bi-phase type). (For example, bi-phase data of SR011 is "0,0" or "1,1", "1" is set to SR019.)




SR0Ex to SR17x are unused area.

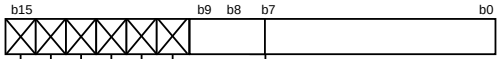
(3) VBI




SR06x to SR17x are unused area.

**Figure 2.15.19 Slice RAM bit composition**

Slice RAM address control register

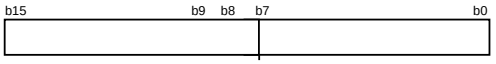



| Symbol | Address | When reset |
|---|---|---|
| SA | $020E_{16}$ | $0000_{16}$ |

| Function | Setting possible value | R | W |
|---|---|---|---|
| Specify accessing slice RAM address | $000_{16}$ to $237_{16}$ | X | O |
| Nothing is assigned. In an attempt to write to these bits, write "0". The value, if read, turns out to be indeterminated. | | – | – |

Note : When access to slice RAM, must be set slice RAM at first, then use slice RAM data control register ($0210_{16}$).
Slice RAM address control register increments by accessing slice RAM data control register. So, it is not neccesary to setting the next slice RAM address.

Slice RAM data control register




| Symbol | Address | When reset |
|---|---|---|
| SD | $0210_{16}$ | $0000_{16}$ |

| Function | R | W |
|---|---|---|
| Read out the data of slice RAM. Read out data of slice RAM which is specified by slice RAM address control register ( address $020E_{16}$) by reading this register. | O | X |

Note : Data access must be 16-bit unit. 8-bit unit access is disable.

**Figure 2.15.20 Slice RAM access registers**

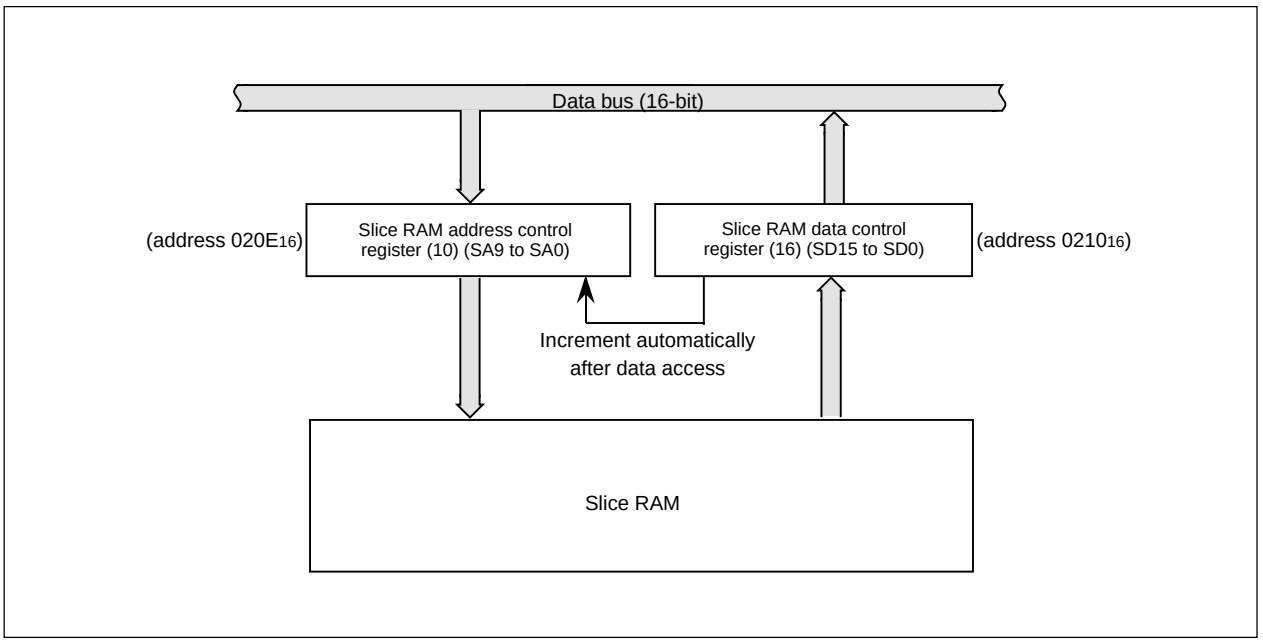



**Figure 2.15.21 Slice RAM access block diagram**

## 2.15.7 VBIRAM

Set 18-line VBI encode data. 5 addresses (8-bit × 5) are prepared for 1 line, out put data in order from LSB side in bi-phase type. Specifiy output pattern (the NRZ type) of header (clock-run in and framing code) (each line command) at addresses $00_{16}$ to $04_{16}$.

VBIRAM composite is shown in Table 2.15.7, VBI encode data composite is shown in Figure 2.15.20.

**Table 2.15.7 VBIRAM composition**

| VBIRAM addresses (EA6 to EA0) | ED7 | ED6 | ED5 | ED4 | ED3 | ED2 | ED1 | ED0 | Remarks |
|---|---|---|---|---|---|---|---|---|---|
| $00_{16}$ | VF07 | VF06 | VF05 | VF04 | VF03 | VF02 | VF01 | VF00 | Specify Clock-run in and Framing code pattern. |
| $01_{16}$ | VF17 | VF16 | VF15 | VF14 | VF13 | VF12 | VF11 | VF10 | 1-bit corresponds to 1T (Max.40 bits). |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | Outputs before data in each line (each line common). |
| $03_{16}$ | VF37 | VF36 | VF35 | VF34 | VF33 | VF32 | VF31 | VF30 | |
| $04_{16}$ | VF47 | VF46 | VF45 | VF44 | VF43 | VF42 | VF41 | VF40 | |
| $05_{16}$ | VR07 | VR06 | VR05 | VR04 | VR03 | VR02 | VR01 | VF00 | Specify output data of 6th line and 318th line. |
| $06_{16}$ | VR17 | VR16 | VR15 | VR14 | VR13 | VR12 | VR11 | VF10 | 1-bit corresponds to bi-phase 1-bit (4T). |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | |
| $08_{16}$ | VR37 | VR36 | VR35 | VR34 | VR33 | VR32 | VR31 | VF30 | |
| $09_{16}$ | VR47 | VR46 | VR45 | VR44 | VR43 | VR42 | VR41 | VF40 | |
| $0A_{16}$ | VR07 | VR06 | VR05 | VR04 | VR03 | VR02 | VR01 | VF00 | Specify output data of 7th line and 319th line. |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | 1-bit corresponds to bi-phase 1-bit (4T). |
| $0E_{16}$ | VR47 | VR46 | VR45 | VR44 | VR43 | VR42 | VR41 | VF40 | |
| ⋮ | ⋮ | | | | | | | | ⋮ |
| $55_{16}$ | VR07 | VR06 | VR05 | VR04 | VR03 | VR02 | VR01 | VF00 | Specify output data of 22th line and 334th line. |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | 1-bit corresponds to bi-phase 1-bit (4T). |
| $59_{16}$ | VR47 | VR46 | VR45 | VR44 | VR43 | VR42 | VR41 | VF40 | |
| $5A_{16}$ | VR07 | VR06 | VR05 | VR04 | VR03 | VR02 | VR01 | VF00 | Specify output data of 23th line and 335th line. |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | 1-bit corresponds to bi-phase 1-bit (4T). |
| $5E_{16}$ | VR47 | VR46 | VR45 | VR44 | VR43 | VR42 | VR41 | VF40 | |

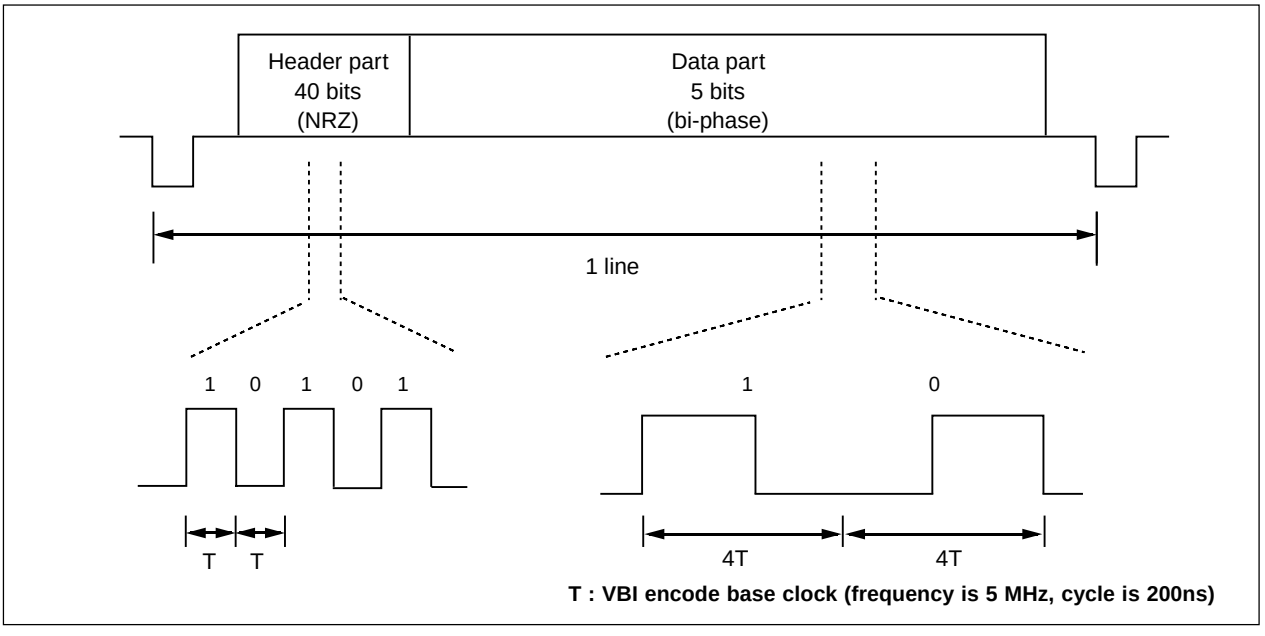
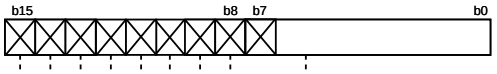



**Figure 2.15.22 VBIRAM encode data composition**

For accessing to VBIRAM data, set accessing address (EA) (shown in Table 2.15.7) to VBIRAM address control register (address $0212_{16}$). Then write data (ED) from VBIRAM data control register (address $0214_{16}$). When end the data accessing, VBIRAM address control register increments address automatically. Then, next address data writing is possible.

VBIRAM access registers are shown in Figure 2.15.23 and VBIRAM access block diagram is shown in Figure 2.15.24.

VBIRAM address control register

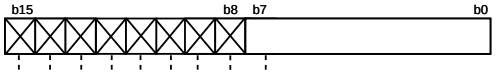



| Symbol | Address | When reset |
|---|---|---|
| EA | $0212_{16}$ | $0000_{16}$ |

| Function | Setting possible value | R | W |
|---|---|---|---|
| Specify accessing VBIRAM address | $000_{16}$ to $5E_{16}$ | X | O |
| Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminated. | | — | — |

Note : When access to VBIRAM, must be set VBIRAM at first, then use VBIRAM data control register ($0214_{16}$).
VBIRAM address control register increments by accessing VBIRAM data control register. So, it is not neccesary to setting the next VBIRAM address.

VBIRAM data control register




| Symbol | Address | When reset |
|---|---|---|
| ED | $0214_{16}$ | $0000_{16}$ |

| Function | Setting possible value | R | W |
|---|---|---|---|
| Write and read out the data of VBIRAM which is specified by SYRAM address control register (address $0212_{16}$) | $00_{16}$ to $FF_{16}$ | O | O |
| Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminated. | | — | — |

Note : Data access must be 16-bit unit. 8-bit unit access is disable.

**Figure 2.15.23 VBIRAM access registers**

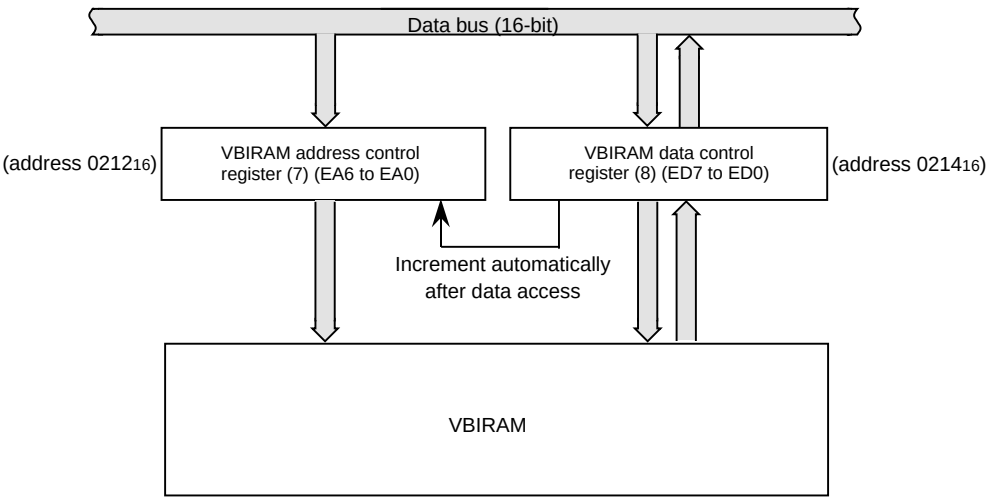
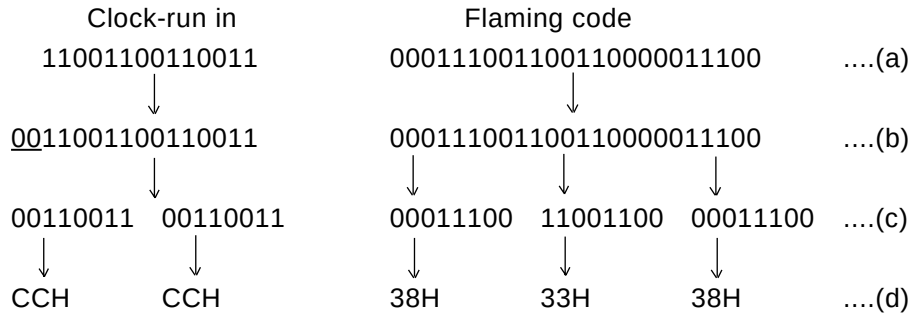



**Figure 2.15.24 VBIRAM access block**

**(1) Setting of Clock-run in and Flaming code**

Specify clock-run in and flamig code output pattern at VBIRAM addresses $00_{16}$ to $04_{16}$ (40 bits). Data 1-bit corresponds to 1T, every byte is output at LSB first.

When clock-run in and flaming code are less than 40 bits (40T), put "0" to the top (from the end, set "0" to unused bit). This pattern of every line is common, outputting before data of every line. Example of setting is shown in Figure 2.15.25.




| VBIRAM address | $00_{16}$ | $01_{16}$ | $02_{16}$ | $03_{16}$ | $04_{16}$ |
|---|---|---|---|---|---|
| Data | $00CC_{16}$ | $00CC_{16}$ | $0038_{16}$ | $0033_{16}$ | $0038_{16}$ |

....(e)

**(a) Clock-run in flaming code output pattern**
**(1-bit corresponds to T)**
**(b) Add 0 × 2 to the top for becoming 40 bits.**
**(c) Cut every 8 bits.**
**(d) Change upper and lower for LSB first mode.**
**(e) Write from VBIRAM address $00_{16}$ at 16-bit mode**

**Figure 2.15.25 Example of setting**

**(2) Data setting**

Set 5 bytes data for 1 line. Setting data is output in bi-phase method. VBI data 1 bit is corresponds to output bi-phase 1 bit (4T). Data specifying is set to RAM which is corresponds to RAM corresponding to the line specifying composition at expansion register VBIL0 to VBIL17. When set to RAM of unspecific line, output is disable.

**(3) EDO2 (VBI-BLNK) signal output specification**

EDO2 signal (BLNK signal for VBI signal) output including former 1.8µs and outer 6.4µs of VBI encode data. Example of output timing is shown in Figure 2.15.26.

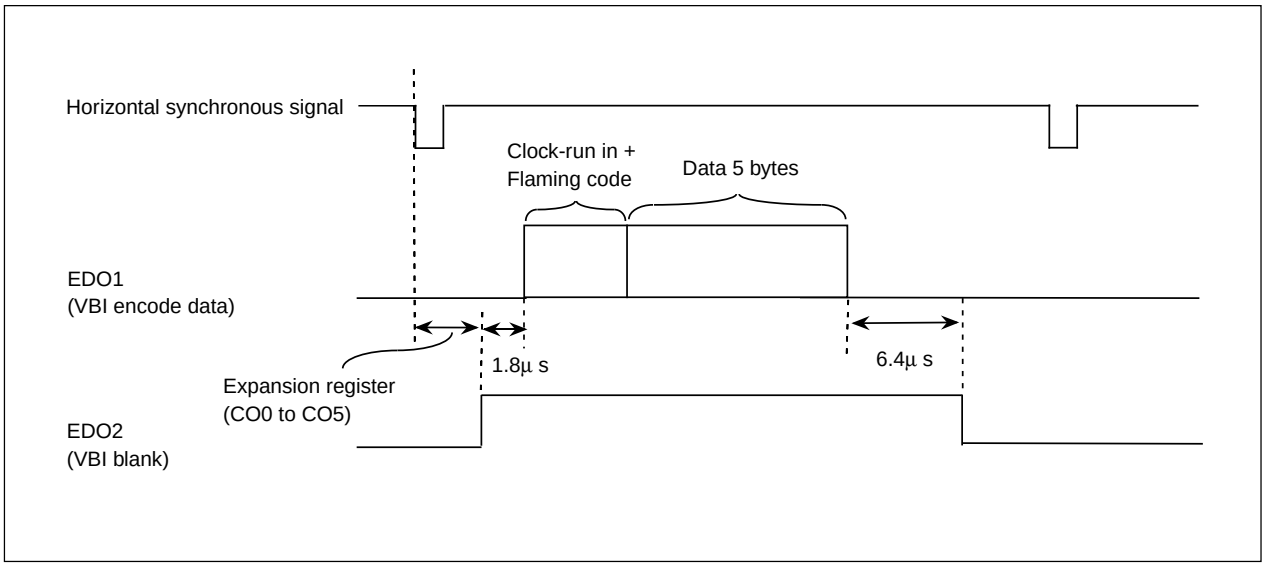



**Figure 2.15.26 Example of output timing**

## 2.15.8 Expansion Register

Control function of OSD function, Data slicer function and VBI encoder function. Expansion register composition is shown in Table 2.15.8.

**Table 2.15.8 Expansion register composition**

| DA5 to DA0 | DD15 | DD14 | DD13 | DD12 | DD11 | DD10 | DD9 | DD8 | DD7 | DD6 | DD5 | DD4 | DD3 | DD2 | DD1 | DD0 | Remarks |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 00₁₆ | PTC6 | PTC5 | PTC4 | PTC3 | PTC2 | PTC1 | PTC0 | STBY0 | VP7 | VP6 | VP5 | VP4 | VP3 | VP2 | VP1 | VP0 | Vertical display position, Port setting |
| 01₁₆ | PTD6 | PTD5 | PTD4 | PTD3 | PTD2 | PTD1 | PTD0 | HP8 | HP7 | HP6 | HP5 | HP4 | HP3 | HP2 | HP1 | HP0 | Horizontal display position, Port setting |
| 02₁₆ | HSZ15 | HSZ14 | HSZ13 | HSZ12 | HSZ11 | HSZ10 | HSZ9 | HSZ8 | HSZ7 | HSZ6 | HSZ5 | HSZ4 | HSZ3 | HSZ2 | HSZ1 | HSZ0 | Horizontal size setting |
| 03₁₆ | – | – | – | TEST2 | TEST1 | TEST0 | BCOL | HSZ24 | HSZ23 | HSZ22 | HSZ21 | HSZ20 | HSZ19 | HSZ18 | HSZ17 | HSZ16 | Horizontal size setting |
| 04₁₆ | VSZ15 | VSZ14 | VSZ13 | VSZ12 | VSZ11 | VSZ10 | VSZ9 | VSZ8 | VSZ7 | VSZ6 | VSZ5 | VSZ4 | VSZ3 | VSZ2 | VSZ1 | VSZ0 | Vertical size setting |
| 05₁₆ | – | – | – | – | BLINK2 | BLINK1 | BLINK0 | VSZ24 | VSZ23 | VSZ22 | VSZ21 | VSZ20 | VSZ19 | VSZ18 | VSZ17 | VSZ16 | Vertical size and blinking setting |
| 06₁₆ | DSP015 | DSP014 | DSP013 | DSP012 | DSP011 | DSP010 | DSP09 | DSP08 | DSP07 | DSP06 | DSP05 | DSP04 | DSP03 | DSP02 | DSP01 | DSP00 | Display mode setting |
| 07₁₆ | – | C05 | C04 | C03 | C02 | C01 | C00 | DSP024 | DSP023 | DSP022 | DSP021 | DSP020 | DSP019 | DSP018 | DSP017 | DSP016 | Display mode setting |
| 08₁₆ | DSP115 | DSP114 | DSP113 | DSP112 | DSP111 | DSP110 | DSP19 | DSP18 | DSP17 | DSP16 | DSP15 | DSP14 | DSP13 | DSP12 | DSP11 | DSP10 | Display mode setting |
| 09₁₆ | – | – | – | – | – | – | – | DSP124 | DSP123 | DSP122 | DSP121 | DSP120 | DSP119 | DSP118 | DSP117 | DSP116 | Display mode setting |
| 0A₁₆ | – | – | – | GRYB | GRYG | GRYR | GRYON | SLIN4 | SLIN3 | SLIN2 | SLIN1 | SLIN0 | SBIT3 | SBIT2 | SBIT1 | SBIT0 | Gray, scroll setting |
| 0B₁₆ | – | PTD8 | PTD7 | SEND4 | SEND3 | SEND2 | SEND1 | SEND0 | – | PTC8 | PTC7 | SST4 | SST3 | SST2 | SST1 | SST0 | Scroll, port setting |
| 0C₁₆ | SECAM | – | IN0 | TIMBAS | – | – | YON1 | – | PC7 | PC6 | PC5 | PC4 | PC3 | PC2 | PC1 | PC0 | Display frequency setting |
| 0D₁₆ | SELFLD | – | MPAL | NXP | EQP | HIDE | LEVEL0 | INTNON | PALH | – | – | – | DSPONV | DSPON | – | EX | Display control setting |
| 0E₁₆ | – | – | – | YON0 | ALL24 | – | – | – | – | LBLACK | LINEB | LINEG | LINER | PHASE2 | PHASE1 | PHASE0 | Color setting |
| 0F₁₆ | – | – | – | – | – | – | – | ADON | – | – | SEL_PDCH | – | – | – | – | – | Slicer control setting |
| 10₁₆ | SELSLI | – | – | – | – | SYNCSEP_ON0 | – | – | SLSLVL | SLI_VP2 | SLI_VP1 | SLI_VP0 | – | – | VPS_SUB | – | Sync separation, slice setting |
| 11₁₆ | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | — |
| 12₁₆ | SEL_VPSH | – | – | – | – | – | – | – | – | – | SEKI5 | SEKI4 | SEKI3 | SEKI2 | SEKI1 | SEKI0 | Slice setting |
| 13₁₆ | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | — |
| 14₁₆ | – | – | – | – | – | – | – | – | – | IN1 | – | – | – | – | – | – | Display setting |
| 15₁₆ | – | – | STBY1 | – | – | – | VPS_VCO_ON | – | – | PDC_VCO_ON | – | – | XTAL_VCO | – | – | CK_VCO | Oscillation ON/OFF setting |
| 16₁₆ | RGBWH | – | – | – | – | PD2 | PD1 | – | PDC_HP10 | PDC_HP9 | PDC_HP8 | PDC_HP7 | PDC_HP6 | PDC_HP5 | PDC_HP4 | PDC_HP3 | PDC slice position setting |
| 17₁₆ | HGSL | HGSLS | – | CCD | – | – | – | – | VPS_HP10 | VPS_HP9 | VPS_HP8 | VPS_HP7 | VPS_HP6 | VPS_HP5 | VPS_HP4 | VPS_HP3 | VPS slice position setting |
| 18₁₆ | VBIL15 | VBIL14 | VBIL13 | VBIL12 | VBIL11 | VBIL10 | VBIL9 | VBIL8 | VBIL7 | VBIL6 | VBIL5 | VBIL4 | VBIL3 | VBIL2 | VBIL1 | VBIL0 | VBI encode setting |
| 19₁₆ | – | VBIL17 | VBIL16 | VPS_LINE4 | VPS_LINE3 | VPS_LINE2 | VPS_LINE1 | VPS_LINE0 | ENCF2 | ENCF1 | VBIF2 | VBIF1 | VPSF2 | VPSF1 | PDCF2 | PDCF1 | Slice setting |
| 1A₁₆ | VPS_FLC7 | VPS_FLC6 | VPS_FLC5 | VPS_FLC4 | VPS_FLC3 | VPS_FLC2 | VPS_FLC1 | VPS_FLC0 | PDC_FLC7 | PDC_FLC6 | PDC_FLC5 | PDC_FLC4 | PDC_FLC3 | PDC_FLC2 | PDC_FLC1 | PDC_FLC0 | PDC, VPS flaming setting |
| 1B₁₆ | – | – | CHK_VPS5 | – | – | – | – | – | – | – | CHK_PDC5 | – | – | – | – | – | — |
| 1C₁₆ | – | – | – | SELPEEK | DIV_PDC8 | DIV_PDC7 | DIV_PDC6 | DIV_PDC5 | DIV_PDC4 | DIV_PDC3 | DIV_PDC2 | DIV_PDC1 | DIV_PDC0 | DIV_PDCS2 | DIV_PDCS1 | DIV_PDCS0 | PDC frequency setting |
| 1D₁₆ | – | – | – | – | DIV_VPS8 | DIV_VPS7 | DIV_VPS6 | DIV_VPS5 | DIV_VPS4 | DIV_VPS3 | DIV_VPS2 | DIV_VPS1 | DIV_VPS0 | DIV_VPSS2 | DIV_VPSS1 | DIV_VPSS0 | VPS frequency setting |
| 1E₁₆ | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | — |
| 1F₁₆ | – | – | – | – | – | – | – | – | MACRON | – | – | FLD | – | – | – | – | Macro, field flag |
| 20₁₆ | – | – | MIN5 | MIN4 | MIN3 | MIN2 | MIN1 | MIN0 | – | – | MAX5 | MAX4 | MAX3 | MAX2 | MAX1 | MAX0 | Slice setting |
| 21₁₆ | – | – | – | – | – | – | – | – | – | DBL_HEIGHT | – | – | – | – | – | – | — |
| 22₁₆ | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | — |

For accessing to expantion register data, set accessing address (DA5 to DA0) (shown in Table 2.15.8) to expantion register address control register (address $0216_{16}$). Then write data (DD15 to DD0) by expantion register data control register (address $0218_{16}$). When end the data accessing, expantion register address control register increments address automatically. Then, next address data writing is possible.

Expantion register access registers are shown in Figure 2.15.27, expansion register access block diagram is shown in Figure 2.15.28, and expansion register bit compositions are shown in p172 to p197.

Expansion register address control register

b15 b8 b7 b5 b0

| Symbol | Address | When reset |
|---|---|---|
| DA | $0216_{16}$ | $0000_{16}$ |

| Function | Setting possible value | R | W |
|---|---|---|---|
| Specify accessing expansion register address | $00_{16}$ to $22_{16}$ | ○ | ○ |
| Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminated. | | – | – |
| Expansion register address auto increments set 0:vaid / 1:invaid (Note2) | – | × | ○ |
| Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminated. | | – | – |

Note1 : When access to expansion register, must be set expansion register address at first, then use expansion register data control register ($0218_{16}$).
Note2 : When bit 8 =“0” setting,expansion register data control register increments by accessing expansion register data control register,so it is not neccesary to setting the next expansion register address.When bit 8 =“1” setting,the address is fixed.

Expansion register data control register

b15 b8 b7 b0

| Symbol | Address | When reset |
|---|---|---|
| DD | $0218_{16}$ | $0000_{16}$ |

| Function | Setting possible value | R | W |
|---|---|---|---|
| Write and read out the data of expansion register which is specified by expansion register address control register (address $0216_{16}$) | $0000_{16}$ to $FFFF_{16}$ | ○ | ○ |

Note : Data access must be 16-bit unit. 8-bit unit access is disable.

**Figure 2.15.27 Expansion register access registers composition**

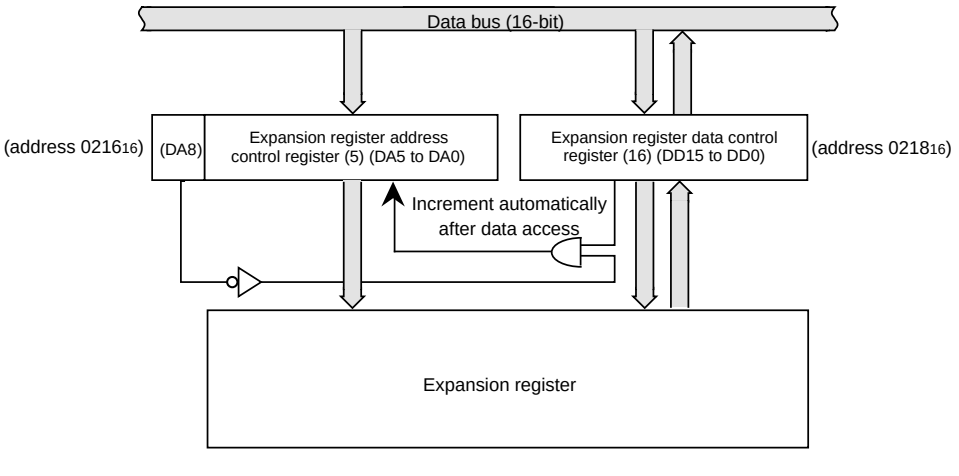
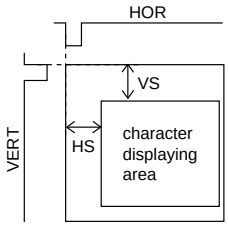


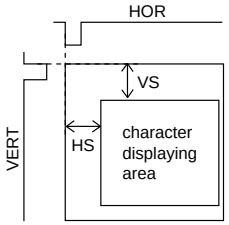

**Figure 2.15.28 Expansion register access block diagram**

## Expansion register construction

### (1) Address $00_{16}$ ( = DA5 to 0)


| Bit symbol | Bit name | Function | | R | W |
|---|---|---|---|---|---|
| VP0 | Vertical display start position selection bit | If VS is the vertical display start position, $VS = H \times \sum_{n=0}^{7} 2^n VPn$<br>H: Cycle with the horizontal synchronizing pulse<br>  <br><br>VP7 to VP0 ≤ ($00001110_2$) are disable | | ○ | ○ |
| VP1 | | | | | |
| VP2 | | | | | |
| VP3 | | | | | |
| VP4 | | | | | |
| VP5 | | | | | |
| VP6 | | | | | |
| VP7 | | | | | |
| STBY0 | Stand-by mode selection bit | 0 | Normal mode | ○ | ○ |
| | | 1 | Stand-by mode | | |
| PTC0 | Port $P11_0$ output selection bit | 0 | P0 output | ○ | ○ |
| | | 1 | EDO2 output | | |
| PTC1 | Port $P11_1$ output selection bit | 0 | P1 output | ○ | ○ |
| | | 1 | EDO1 output | | |
| PTC2 | Port $P11_2$ output selection bit | 0 | P2 output | ○ | ○ |
| | | 1 | CSYN output | | |
| PTC3 | Port $P11_3$ output selection bit | 0 | P3 output | ○ | ○ |
| | | 1 | BLNK output | | |
| PTC4 | Port $P11_4$ output selection bit | 0 | P4 output | ○ | ○ |
| | | 1 | B output | | |
| PTC5 | Port $P11_5$ output selection bit | 0 | P5 output | ○ | ○ |
| | | 1 | G output | | |
| PTC6 | Port $P11_6$ output selection bit | 0 | P6 output | ○ | ○ |
| | | 1 | R output | | |

### (2) Address $01_{16}$ ( = DA5 to 0)


| Bit symbol | Bit name | Function | | R | W |
|---|---|---|---|---|---|
| HP0 | Horizontal display start position selection bit | If HS is the vertical display start position,<br>$HS = T1 \times \sum_{n=0}^{8} 2^n HPn + 9$<br>T1 : Cycle with the display clock<br>  <br><br>HP8 to HP0 ≤ ($000010011_2$) are disable | | ○ | ○ |
| HP1 | | | | | |
| HP2 | | | | | |
| HP3 | | | | | |
| HP4 | | | | | |
| HP5 | | | | | |
| HP6 | | | | | |
| HP7 | | | | | |
| HP8 | | | | | |
| PTD0 | Port $P11_0$ data selection bit | 0 | When port output : fixed to L , when EDO2 output : specified negative polarity. | ○ | ○ |
| | | 1 | When port output : fixed to H , when EDO2 output : specified positive polarity. | | |
| PTD1 | Port $P11_1$ data selection bit | 0 | When port output : fixed to L , when EDO1 output : specified negative polarity. | ○ | ○ |
| | | 1 | When port output : fixed to H , when EDO1 output : specified positive polarity. | | |
| PTD2 | Port $P11_2$ data selection bit | 0 | When port output : fixed to L , when CSYN output : specified negative polarity. | ○ | ○ |
| | | 1 | When port output : fixed to H , when CSYN output : specified positive polarity. | | |
| PTD3 | Port $P11_3$ data selection bit | 0 | When port output : fixed to L , when BLNK output : specified negative polarity. | ○ | ○ |
| | | 1 | When port output : fixed to H , when BLNK output : specified positive polarity. | | |
| PTD4 | Port $P11_4$ data selection bit | 0 | When port output : fixed to L , when B output : specified negative polarity. | ○ | ○ |
| | | 1 | When port output : fixed to H , when B output : specified positive polarity. | | |
| PTD5 | Port $P11_5$ data selection bit | 0 | When port output : fixed to L , when G output : specified negative polarity. | ○ | ○ |
| | | 1 | When port output : fixed to H , when G output : specified positive polarity. | | |
| PTD6 | Port $P11_6$ data selection bit | 0 | When port output : fixed to L , when R output : specified negative polarity. | ○ | ○ |
| | | 1 | When port output : fixed to H , when R output : specified positive polarity. | | |

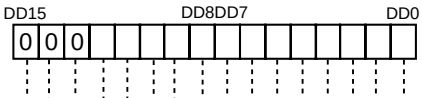
### (3) Address $02_{16}$ ( = DA5 to 0)


| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| HSZ0 | The 0th line horizontal character selection bit | Horizontal direction character size of the line n is set by HSZn (n = 0 to 24). Set at one time or two times in the each every line. | ○ | ○ |
| HSZ1 | The first line horizontal character selection bit | | ○ | ○ |
| HSZ2 | The second line horizontal character selection bit | | ○ | ○ |
| HSZ3 | The third line horizontal character selection bit | | ○ | ○ |
| HSZ4 | The 4th line horizontal character selection bit | | ○ | ○ |
| HSZ5 | The 5th line horizontal character selection bit | | ○ | ○ |
| HSZ6 | The 6th line horizontal character selection bit | | ○ | ○ |
| HSZ7 | The 7th line horizontal character selection bit | | ○ | ○ |
| HSZ8 | The 8th line horizontal character selection bit | | ○ | ○ |
| HSZ9 | The 9th line horizontal character selection bit | | ○ | ○ |
| HSZ10 | The 10th line horizontal character selection bit | | ○ | ○ |
| HSZ11 | The 11th line horizontal character selection bit | | ○ | ○ |
| HSZ12 | The 12th line horizontal character selection bit | | ○ | ○ |
| HSZ13 | The 13th line horizontal character selection bit | | ○ | ○ |
| HSZ14 | The 14th line horizontal character selection bit | | ○ | ○ |
| HSZ15 | The 15th line horizontal character selection bit | | ○ | ○ |

| HSZn | Horizontal direction character size |
|---|---|
| 0 | 1T/dot (one time) |
| 1 | 2T/dot (two times) |

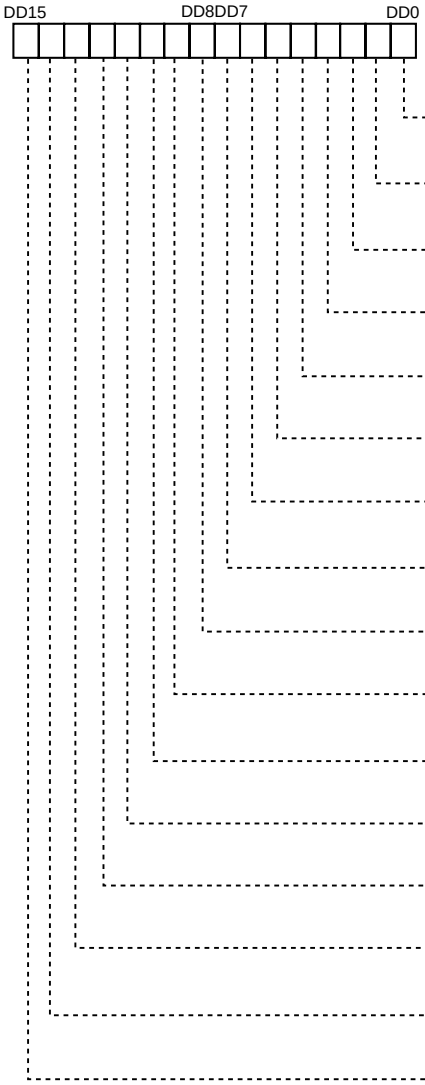
T : Display clock

### (4) Address $03_{16}$ ( = DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| HSZ16 | The 16th line horizontal character selection bit | Horizontal direction character size of the line n is set by HSZn (n = 0 to 24). Set at one time or two times in the each every line.<br>HSZn: 0 = 1T/dot (one time); 1 = 2T/dot (two times)<br>T : Display clock | ○ | ○ |
| HSZ17 | The 17th line horizontal character selection bit | | ○ | ○ |
| HSZ18 | The 18th line horizontal character selection bit | | ○ | ○ |
| HSZ19 | The 19th line horizontal character selection bit | | ○ | ○ |
| HSZ20 | The 20th line horizontal character selection bit | | ○ | ○ |
| HSZ21 | The 21th line horizontal character selection bit | | ○ | ○ |
| HSZ22 | The 22th line horizontal character selection bit | | ○ | ○ |
| HSZ23 | The 23th line horizontal character selection bit | | ○ | ○ |
| HSZ24 | The 24th line horizontal character selection bit | | ○ | ○ |
| BCOL | All blanking selection bit | 0: Blanking of DSP1n and DSP0n<br>1: All raster blanking | ○ | ○ |
| TEST0 | Test bit | Must always be set to "0". | ○ | ○ |
| TEST1 | | | ○ | ○ |
| TEST2 | | | ○ | ○ |
| Reserved bit | | Must always be set to "0". | × | ○ |

### (5) Address $04_{16}$ ( = DA5 to 0)

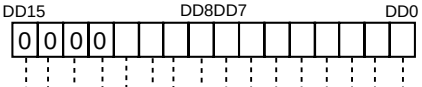



| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| VSZ0 | The 0th line vertical character size selection bit | Vertical direction character size of the line n is set by VSZn (n = 0 to 24) Set at one time or two times each every line. | ○ | ○ |
| VSZ1 | The first line vertical character size selection bit | | ○ | ○ |
| VSZ2 | The second line vertical character size selection bit | | ○ | ○ |
| VSZ3 | The third line vertical character size selection bit | | ○ | ○ |
| VSZ4 | The 4th line vertical character size selection bit | | ○ | ○ |
| VSZ5 | The 5th line vertical character size selection bit | | ○ | ○ |
| VSZ6 | The 6th line vertical character size selection bit | | ○ | ○ |
| VSZ7 | The 7th line vertical character size selection bit | | ○ | ○ |
| VSZ8 | The 8th line vertical character size selection bit | | ○ | ○ |
| VSZ9 | The 9th line vertical character size selection bit | | ○ | ○ |
| VSZ10 | The 10th line vertical character size selection bit | | ○ | ○ |
| VSZ11 | The 11th line vertical character size selection bit | | ○ | ○ |
| VSZ12 | The 12th line vertical character size selection bit | | ○ | ○ |
| VSZ13 | The 13th line vertical character size selection bit | | ○ | ○ |
| VSZ14 | The 14th line vertical character size selection bit | | ○ | ○ |
| VSZ15 | The 15th line vertical character size selection bit | | ○ | ○ |

| VSZn | Vertical direction character size |
|---|---|
| 0 | 1H/dot (one time) |
| 1 | 2H/dot (two times) |

H : Horizontal synchronous pulse

### (6) Address $05_{16}$ ( = DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| VSZ16 | The 16th line vertical character size selection bit | Vertical direction character size of the line n is set by VSZn (n = 0 to 24) Set at one time or two times in the each every line. | ○ | ○ |
| VSZ17 | The 17th line vertical character size selection bit | | ○ | ○ |
| VSZ18 | The 18th line vertical character size selection bit | | ○ | ○ |
| VSZ19 | The 19th line vertical character size selection bit | | ○ | ○ |
| VSZ20 | The 20th line vertical character size selection bit | | ○ | ○ |
| VSZ21 | The 21th line vertical character size selection bit | | ○ | ○ |
| VSZ22 | The 22th line vertical character size selection bit | | ○ | ○ |
| VSZ23 | The 23th line vertical character size selection bit | | ○ | ○ |
| VSZ24 | The 24th line vertical character size selection bit | | ○ | ○ |
| BLINK0 | Blinking duty selection bit | | ○ | ○ |
| BLINK1 | | | ○ | ○ |
| BLINK2 | Blinking cycle selection bit | | ○ | ○ |
| Reserved bit | | Must always be set to "0". | × | ○ |

| VSZn | Vertical direction character size |
|---|---|
| 0 | 1H/dot (one time) |
| 1 | 2H/dot (two times) |

H : Horizontal synchronous pulse

| BLINK1 | BLINK0 | DUTY |
|---|---|---|
| 0 | 0 | Blinking off |
| 0 | 1 | 25% |
| 1 | 1 | 50% |
| 1 | 0 | 75% |

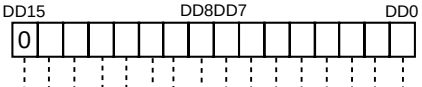
| | |
|---|---|
| 0 | Cycle approximatery 1 second. |
| 1 | Cycle approximatery 0.5 second. |

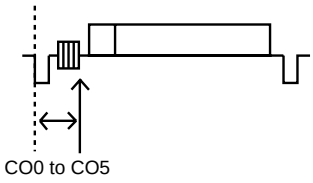
### (7) Address $06_{16}$ ( = DA5 to 0)


| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DSP00 | The 0th line display mode selection bit | Set the display mode of the line n (blanking mode) by combination of DSP0n (addresses $06_{16}$ and $07_{16}$) and DSP1n (addresses $08_{16}$ and $09_{16}$) (n = 0 to 24) 3 kinds of following setting are possible for the each every line. | ○ | ○ |
| DSP01 | The first line display mode selection bit | | ○ | ○ |
| DSP02 | The second line display mode selection bit | | ○ | ○ |
| DSP03 | The third line display mode selection bit | | ○ | ○ |
| DSP04 | The 4th line display mode selection bit | | ○ | ○ |
| DSP05 | The 5th line display mode selection bit | | ○ | ○ |
| DSP06 | The 6th line display mode selection bit | | ○ | ○ |
| DSP07 | The 7th line display mode selection bit | | ○ | ○ |
| DSP08 | The 8th line display mode selection bit | | ○ | ○ |
| DSP09 | The 9th line display mode selection bit | | ○ | ○ |
| DSP010 | The 10th line display mode selection bit | | ○ | ○ |
| DSP011 | The 11th line display mode selection bit | | ○ | ○ |
| DSP012 | The 12th line display mode selection bit | | ○ | ○ |
| DSP013 | The 13th line display mode selection bit | | ○ | ○ |
| DSP014 | The 14th line display mode selection bit | | ○ | ○ |
| DSP015 | The 15th line display mode selection bit | | ○ | ○ |

| DSP1n | DSP0n | Display mode |
|---|---|---|
| 0 | 0 | Character |
| 0 | 1 | Disable |
| 1 | 0 | Matrix-outline |
| 1 | 1 | Halftone |

### (8) Address $07_{16}$ ( = DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DSP016 | The 16th line display mode selection bit | Set the display mode of the line n (blanking mode) by combination of DSP0n (addresses $06_{16}$ and $07_{16}$) and DSP1n (addresses $08_{16}$ and $09_{16}$) ( n = 0 to 24)<br>3 kinds of following setting are possible for the each every line. | ○ | ○ |
| DSP017 | The 17th line display mode selection bit | | ○ | ○ |
| DSP018 | The 18th line display mode selection bit | | ○ | ○ |
| DSP019 | The 19th line display mode selection bit | | ○ | ○ |
| DSP020 | The 20th line display mode selection bit | | ○ | ○ |
| DSP021 | The 21th line display mode selection bit | | ○ | ○ |
| DSP022 | The 22th line display mode selection bit | | ○ | ○ |
| DSP023 | The 23th line display mode selection bit | | ○ | ○ |
| DSP024 | The24th line display mode selection bit | | ○ | ○ |
| CO0 | VBI encode horizontal start position selection bit | VBI encode horizontal start position<br>  <br>CO0 to CO5<br>(Each line are set to common) | ○ | ○ |
| CO1 | | | ○ | ○ |
| CO2 | | | ○ | ○ |
| CO3 | | | ○ | ○ |
| CO4 | | | ○ | ○ |
| CO5 | | | ○ | ○ |
| Reserved bit | | Must always be set to "0". | × | ○ |

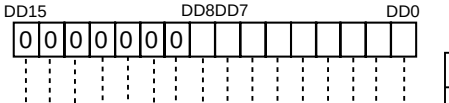
| DSP1n | DSP0n | Display mode |
|---|---|---|
| 0 | 0 | Character |
| 0 | 1 | Disable |
| 1 | 0 | Matrix-outline |
| 1 | 1 | Halftone |

### (9) Address $08_{16}$ ( = DA5 to 0)


| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DSP10 | The 10th line display mode selection bit | Set the display mode of the line n (blanking mode) by combination of DSP0n (addresses $06_{16}$ and $07_{16}$) and DSP1n (addresses $08_{16}$ and $09_{16}$) ( n = 0 to 24) 3 kinds of following setting are possible for the each every line. | ○ | ○ |
| DSP11 | The 11th line display mode selection bit | | ○ | ○ |
| DSP12 | The 12th line display mode selection bit | | ○ | ○ |
| DSP13 | The 13th line display mode selection bit | | ○ | ○ |
| DSP14 | The 14th line display mode selection bit | | ○ | ○ |
| DSP15 | The 15th line display mode selection bit | | ○ | ○ |
| DSP16 | The 16th line display mode selection bit | | ○ | ○ |
| DSP17 | The 17th line display mode selection bit | | ○ | ○ |
| DSP18 | The 18th line display mode selection bit | | ○ | ○ |
| DSP19 | The 19th line display mode selection bit | | ○ | ○ |
| DSP110 | The 20th line display mode selection bit | | ○ | ○ |
| DSP111 | The 21th line display mode selection bit | | ○ | ○ |
| DSP112 | The 22th line display mode selection bit | | ○ | ○ |
| DSP113 | The 23th line display mode selection bit | | ○ | ○ |
| DSP114 | The 24th line display mode selection bit | | ○ | ○ |
| DSP115 | The 25th line display mode selection bit | | ○ | ○ |

| DSP1n | DSP0n | Display mode |
|---|---|---|
| 0 | 0 | Character |
| 0 | 1 | Disable |
| 1 | 0 | Matrix-outline |
| 1 | 1 | Halftone |

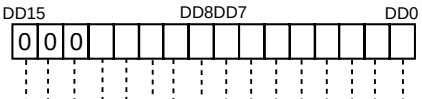
### (10) Address $09_{16}$ ( = DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DSP116 | The 16th line display mode selection bit | Set the display mode of the line n (blanking mode) by combination of DSP0n (addresses $06_{16}$ and $07_{16}$) and DSP1n (addresses $08_{16}$ and $09_{16}$) ( n = 0 to 24) 3 kinds of following setting are possible for the each every line. | ○ | ○ |
| DSP117 | The 17th line display mode selection bit | | ○ | ○ |
| DSP118 | The 18th line display mode selection bit | | ○ | ○ |
| DSP119 | The 19th line display mode selection bit | | ○ | ○ |
| DSP120 | The 20th line display mode selection bit | | ○ | ○ |
| DSP121 | The 21th line display mode selection bit | | ○ | ○ |
| DSP122 | The 22th line display mode selection bit | | ○ | ○ |
| DSP123 | The 23th line display mode selection bit | | ○ | ○ |
| DSP124 | The24th line display mode selection bit | | ○ | ○ |
| Reserved bit | | Must always be set to "0". | × | ○ |

| DSP1n | DSP0n | Display mode |
|---|---|---|
| 0 | 0 | Character |
| 0 | 1 | Disable |
| 1 | 0 | Matrix-outline |
| 1 | 1 | Halftone |

### (11) Address 0A16 ( = DA5 to 0)

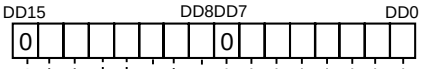



| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SBIT0 | Scroll display start dot selection bit | If SA is display start dot of scroll block,<br>$SA=\sum_{n=0}^{3}2^{n}SBITn$<br>SBIT3 to SBIT0 ≥ ($1010_2$) is disable | ○ | ○ |
| SBIT1 | | | ○ | ○ |
| SBIT2 | | | ○ | ○ |
| SBIT3 | | | ○ | ○ |
| SLIN0 | Scroll display start dot selection bit | If SB is display start dot of scroll block,<br>$SB=\sum_{n=0}^{4}2^{n}SBITn$<br>SLIN4 to SLIN0 ≥ ($11001_2$) is disable.<br>Set the value which is satisfies with shown below :<br>SST4 to SST0 ≤ SLIN4 to SLIN0 < SEND4 to SEND0 | ○ | ○ |
| SLIN1 | | | ○ | ○ |
| SLIN2 | | | ○ | ○ |
| SLIN3 | | | ○ | ○ |
| SLIN4 | | | ○ | ○ |
| GRYON | Gray display selection bit | 0 Normal display.<br>1 Gray display setting one color of eight colors. (Note 1) | ○ | ○ |
| GRYR | Gray display color selection bit | See color table below<br>Gray color is set by this register<br>Valid only ot GRYON = "1" | ○ | ○ |
| GRYG | | | ○ | ○ |
| GRYB | | | ○ | ○ |
| Reserved bit | | Must always be set to "0". | × | ○ |

| GRYB | GRYG | GRYR | Color |
|---|---|---|---|
| 0 | 0 | 0 | Black |
| 0 | 0 | 1 | Red |
| 0 | 1 | 0 | Green |
| 0 | 1 | 1 | Yellow |
| 1 | 0 | 0 | Blue |
| 1 | 0 | 1 | Magenta |
| 1 | 1 | 0 | Cyan |
| 1 | 1 | 1 | White |

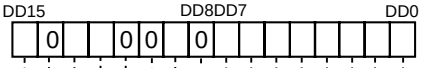
Note 1. Refer to register RGBWH (Address $16_{16}$) about RGB output.

### (12) Address 0B16 ( = DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SST0 | Scroll block start line selection bit | If SC is start line of scroll block, $SC=\sum_{n=0}^{4}2^{n}SSTn$ SST4 to SST0 ≥ ($11000_2$) is disable | ○ | ○ |
| SST1 | | | ○ | ○ |
| SST2 | | | ○ | ○ |
| SST3 | | | ○ | ○ |
| SST4 | | | ○ | ○ |
| PTC7 | Port P7 output selection bit | 0 P7 output<br>1 GRAY output | ○ | ○ |
| PTC8 | Port P8 output selection bit | 0 P8 output<br>1 SLICEON output | ○ | ○ |
| Reserved bit | | Must always be set to "0". | × | ○ |
| SEND0 | Scroll block last line selection bit | If SD is the fixation start line of the below the scroll block (last line of the scroll block + 1) $SD=\sum_{n=0}^{4}2^{n}SENDn$ Set the value which will be (SEND4 to SEND0) ≥ (SST4 to SST0) + 2 When scroll on, SEND4 to SEND0 ≤ ($00001_2$) and SEND4 to SEND0 ≥ ($11010_2$) are disable. When scroll off, SEND4 to SEND0 = ($00000_2$) is available. | ○ | ○ |
| SEND1 | | | ○ | ○ |
| SEND2 | | | ○ | ○ |
| SEND3 | | | ○ | ○ |
| SEND4 | | | ○ | ○ |
| PTD7 | Port P7 data selection bit | 0 When port output : fixed to "H" when GRAY output : specified negative polarity<br>1 When port output : fixed to "L" when GRAY output : specified positive polarity | ○ | ○ |
| PTD8 | Port P8 data selection bit | 0 When port output : fixed to "H" when SLICEON output : specified negative polarity<br>1 When port output : fixed to "L" when SLICEON output : specified positive polarity | ○ | ○ |
| Reserved bit | | Must always be set to "0". | × | ○ |

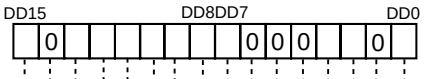
### (13) Address 0C16 ( = DA5 to 0)




| Bit symbol | Bit name | Function | | R | W |
|---|---|---|---|---|---|
| PC0 | Display frequency selection bit | Control display frequency $f_T$, $f_T = f_H \times \{\sum_{n=0}^{7} 2^n PCn+512\}$ $f_H$ : Horizontal synchronous signal frequency PC7 to PC0 ≤ ($01111111_2$) is disable. Set PC7 to PC0 = ($11110101_2$), normally. | | ○ | ○ |
| PC1 | | | | ○ | ○ |
| PC2 | | | | ○ | ○ |
| PC3 | | | | ○ | ○ |
| PC4 | | | | ○ | ○ |
| PC5 | | | | ○ | ○ |
| PC6 | | | | ○ | ○ |
| PC7 | | | | ○ | ○ |
| Reserved bit | | Must always be set to "0". | | × | ○ |
| YON1 | Color burst at internal synchronous selection bit (Note) | 0 | Color burst ON | ○ | ○ |
| | | 1 | Color burst OFF | | |
| Reserved bit | | Must always be set to "0". | | × | ○ |
| TIMBAS | Time base selection bit | 0 | Time base ON | ○ | ○ |
| | | 1 | Time base OFF | | |
| IN0 | Internal synchronous selection bit | 0 | External synchronous setting | ○ | ○ |
| | | 1 | Internal synchronous setting | | |
| Reserved bit | | Must always be set to "0". | | × | ○ |
| SECAM | Combination selection bit from SECAMIN pin | 0 | Do not superimpose the carrier from SECAMIN pin. | ○ | ○ |
| | | 1 | Superimpose the carrier from SECAMIN pin. | | |

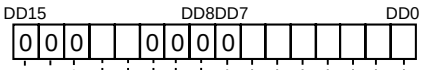
Note1. When moto-tone display (YON0(address $0E_{16}$)= "1") setting, must be set to "1".

### (14) Address 0D16 ( = DA5 to 0)




| Bit symbol | Bit name | Function | | R | W |
|---|---|---|---|---|---|
| EX | External/internal synchronous selection bit | 0 | External synchronization | ○ | ○ |
| | | 1 | Internal synchronization | | |
| Reserved bit | | Must always be set to "0". | | × | ○ |
| DSPON | Digital display selection bit | 0 | Digital output display OFF. | ○ | ○ |
| | | 1 | Digital outoput display ON. | | |
| DSPONV | Analog display selection bit | 0 | Composite video signal output display OFF. | ○ | ○ |
| | | 1 | Composite video signal output display ON. | | |
| Reserved bit | | Must always be set to "0". | | × | ○ |
| PALH | Number of scanning line selection bit | PALH | INT/NON | Number of scanning line | ○ | ○ |
| | | 0 | 0 | 625H | | |
| | | 0 | 1 | 626H | | |
| | | 1 | 0 | 624H | | |
| | | 1 | 1 | 628H | | |
| INTNON | | | | | ○ | ○ |
| LEVEL0 | Video signal generation selection bit | 0 | Composite video signal generation circuit OFF. | ○ | ○ |
| | | 1 | Composite video signal generation circuit ON. | | |
| HIDE | SYRAM expantion display selection bit | 0 | SYRAM writing over | ○ | ○ |
| | | 1 | SYRAM writing over or character erasing | | |
| EQP | Equivalent pulse selection bit | 0 | Do not include equivalent pulse. | ○ | ○ |
| | | 1 | Includes equivalent pulse. | | |
| NXP | Broadcast method selection bit | N/P | MPAL | Broadcasting method | ○ | ○ |
| | | 0 | 0 | NTSC | | |
| | | 0 | 1 | M-PAL | | |
| | | 1 | 0 | PAL | | |
| | | 1 | 1 | Disable | | |
| MPAL | | | | | ○ | ○ |
| Reserved bit | | Must always be set to "0". | | × | ○ |
| SELFLD | Field at non interlace selection bit | 0 | The secound field. | ○ | ○ |
| | | 1 | The first field. | | |

### (15) Address 0E16 ( = DA5 to 0)



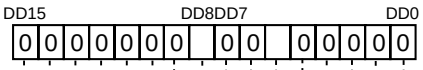

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PHASE0 | Raster color selection bit | See table below | ○ | ○ |
| PHASE1 | | | ○ | ○ |
| PHASE2 | | Raster color setting when Register GRYON = 0<br>Refer to address 0A16 when color setting at GRYON = 1 | ○ | ○ |
| LINER | SYRAM color selection bit | See table below | ○ | ○ |
| LINEG | | | ○ | ○ |
| LINEB | | SYRAM color setting when Register GRYON = 0.<br>Refer to address 0A16 when color setting at GRYON = 1. | ○ | ○ |
| LBLACK | Video signal black level selection bit | 0: 1.6V<br>1: 1.8V | ○ | ○ |
| Reserved bit | | Must always be set to "0". | × | ○ |
| ALL24 | Horizontal direction matrix outline range selection bit | 0: OSD horizontal display range (40 characters)<br>1: All range of horizontal display period. | ○ | ○ |
| YON0 | Internal synchronous moto-tone display selection bit | 0: Color display<br>1: Mono-ton display (Note1 ) | ○ | ○ |
| Reserved bit | | Must always be set to "0". | × | ○ |

| PHASE2 | PHASE1 | PHASE0 | Color |
|---|---|---|---|
| 0 | 0 | 0 | Black |
| 0 | 0 | 1 | Red |
| 0 | 1 | 0 | Green |
| 0 | 1 | 1 | Yellow |
| 1 | 0 | 0 | Blue |
| 1 | 0 | 1 | Magenta |
| 1 | 1 | 0 | Cyan |
| 1 | 1 | 1 | White |

| LINEB | LINEG | LINER | Color |
|---|---|---|---|
| 0 | 0 | 0 | Black |
| 0 | 0 | 1 | Red |
| 0 | 1 | 0 | Green |
| 0 | 1 | 1 | Yellow |
| 1 | 0 | 0 | Blue |
| 1 | 0 | 1 | Magenta |
| 1 | 1 | 0 | Cyan |
| 1 | 1 | 1 | White |

Note1. When moto-tone display(YON0="1") setting, must be set YON1(address OC16)="1".

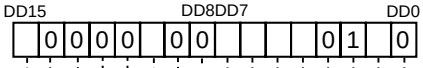
### (16) Address 0F16 ( = DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Reserved bit | | Must always be set to "0". | × | ○ |
| SEL_PDCH | PDC clock selection bit | 0: Generates PDC clock in based on external fH.<br>1: Generates PDC clock in based on FSCIN pin input signal. | ○ | ○ |
| Reserved bit | | Must always be set to "0". | × | ○ |
| ADON | Data slicer control bit | 0: Data slicer OFF<br>1: Data slicer ON | ○ | ○ |
| Reserved bit | | Must always be set to "0". | × | ○ |

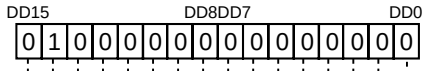
Note1. When ADLAT0="1" setting, must be set ADLAT1(address 1416)="1".

### (17) Address10$_{16}$ ( = DA5 to 0)



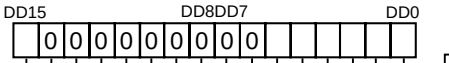

| Bit symbol | Bit name | Function | | R | W |
|---|---|---|---|---|---|
| Reserved bit | | Must always be set to "0". | | × | ○ |
| VPS_SUB | Flaming code check selection bit for VPS data. | 0 | Later 8bits of flaming code 16bits | ○ | ○ |
| | | 1 | Former 4bits and later 4bits of flaming code 16bits (Select 8bits which is set in VPS_FLC0 to 7) | | |
| Reserved bit | | Must always be set to "1". | | × | ○ |
| Reserved bit | | Must always be set to "0". | | × | ○ |
| SLI_VP0 | Slice start line selection bit (Field 1 and 2 are common) Stores data for 18 lines from the 6th line,normally. (SLI_VP2 to SLI_VPO = "3$_{16}$" fixed) | If the slice start line is SLI_VS, <Field 1> SLI_VS= $\sum_{n=0}^{2} 2^n$SLI_VPn+3 <Field 2> SLI_VS= $\sum_{n=0}^{2} 2^n$SLI_VPn+315 Stores data for 18 lines from line which is set by this register to slice RAM. | | ○ | ○ |
| SLI_VP1 | | | | ○ | ○ |
| SLI_VP2 | | | | ○ | ○ |
| SLSLVL | Slice level control bit | 0 | Auto level for data slice | ○ | ○ |
| | | 1 | Fix level for data slice | | |
| Reserved bit | | Must always be set to "0". | | × | ○ |
| SYNCSEP_ON0 | Synchronous separation control bit | 0 | Sync-sep circuit OFF | ○ | ○ |
| | | 1 | Sync-sep circuit ON | | |
| Reserved bit | | Must always be set to "0". | | × | ○ |
| SELSLI | Slice signal input pin selection bit | 0 | CVIN1 pin | ○ | ○ |
| | | 1 | CVIN2 pin | | |

### (18) Address 11$_{16}$( = DA5 to 0)



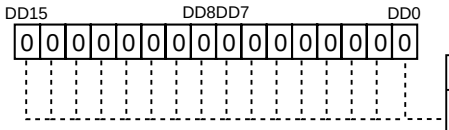

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Reserved bit | | Must always be set to "0". | × | ○ |
| Reserved bit | | Must always be set to "1". | × | ○ |
| Reserved bit | | Must always be set to "0". | × | ○ |

### (19) Address 12$_{16}$ ( = DA5 to 0)



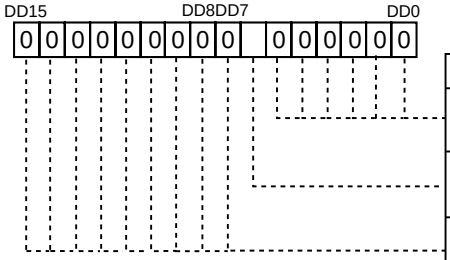

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SEKI0 | Data slicer control bit 1 | SEKI1 SEKI0 N: 0 0 5; 0 1 4; 1 0 3; 1 1 2. N times of the digital value after AD is done. | ○ | ○ |
| SEKI1 | | | ○ | ○ |
| SEKI2 | Data slicer control bit 2 | SEKI3 SEKI2 N: 0 0 4; 0 1 3; 1 0 1; 1 1 Not differentiate. It is differentiated for digital value after the SEKI0, 1 operation at digital value in the before N/8 period(clock run-in period). | ○ | ○ |
| SEKI3 | | | ○ | ○ |
| SEKI4 | Data slicer control bit 3 | SEKI5 SEKI4 N: 0 0 4; 0 1 3; 1 0 1; 1 1 Not differentiate. It is differentiated for digital value after the SEKI3, 2 operation at digital value in the after N/8 period(clock run-in period). | ○ | ○ |
| SEKI5 | | | ○ | ○ |
| Reserved bit | | Must always be set to "0" | × | ○ |
| SEL_VPSH | VPS clock selection bit | 0: Generats VPS clock in based on external fH. 1: Generats VPS clock in based on FSCIN pin input signal. | ○ | ○ |

### (20) Address 13$_{16}$ ( = DA5 to 0)



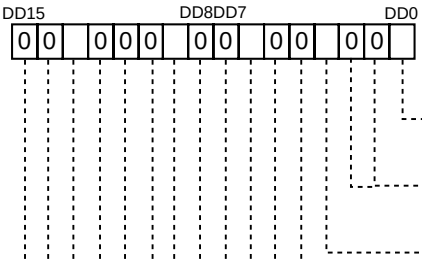

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Reserved bit | | Must always be set to "0". | × | ○ |

### (21) Address $14_{16}$ ( = DA5 to 0)



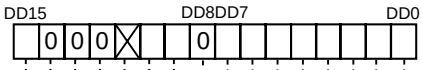

| Bit symbol | Bit name | Function | | R | W |
|---|---|---|---|---|---|
| Reserved bit | | Must always be set to "0". | | × | ○ |
| IN1 | Internal synchronous selection bit | 0 | External synchronous setting | ○ | ○ |
| | | 1 | Internal synchronous setting | | |
| Reserved bit | | Must always be set to "0". | | × | ○ |

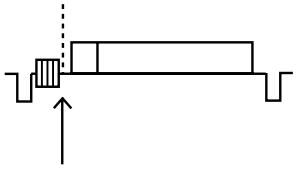
### (22) Address $15_{16}$ ( = DA5 to 0)



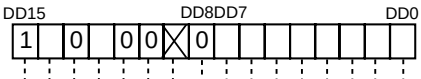

| Bit symbol | Bit name | Function | | R | W |
|---|---|---|---|---|---|
| CK_VCO | Display clock oscillation selection bit | **0** | Display clock OFF | ○ | ○ |
| | | **1** | Display clock oscillation | | |
| Reserved bit | | Must always be set to "0". | | × | ○ |
| XTAL_VCO | Synchronous clock oscillation selection bit | 0 | Synchronizing clock OFF | ○ | ○ |
| | | 1 | Synchronizing clock oscillation | | |
| Reserved bit | | Must always be set to "0". | | × | ○ |
| PDC_VCO_ON | PDC clock oscillation selection bit | 0 | PDC clock OFF | ○ | ○ |
| | | 1 | PDC clock oscillation | | |
| Reserved bit | | Must always be set to "0". | | × | ○ |
| VPS_VCO_ON | VPS and VBI clock oscillation selection bit | 0 | VPS and VBI clock OFF | ○ | ○ |
| | | 1 | VPS and VBI clock oscillation | | |
| Reserved bit | | Must always be set to "0". | | × | ○ |
| STBY1 | Stand-by mode selection bit | 0 | Normal mode | ○ | ○ |
| | | 1 | Stand-by mode. | | |
| Reserved bit | | Must always be set to "0". | | × | ○ |

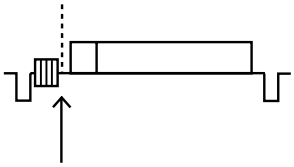
### (23) Address $16_{16}$ ( = DA5 to 0)



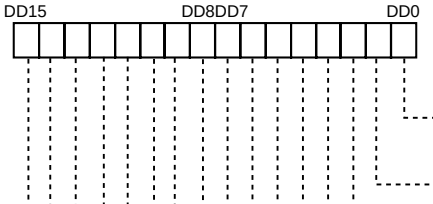

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PDC_HP3 | PDC slice check start position selection bit | If the PDC slice check start position is PDC_HS,<br>$PDC\_HS = T3 \times \sum_{n=3}^{10} 2^{(n-3)} PDC\_HPn$<br>T3 : PDC clock run-in cycle ÷2<br>  <br>Set to flaming code check start position<br>Set by the 144ns (1bit) | ○ | ○ |
| PDC_HP4 | | | ○ | ○ |
| PDC_HP5 | | | ○ | ○ |
| PDC_HP6 | | | ○ | ○ |
| PDC_HP7 | | | ○ | ○ |
| PDC_HP8 | | | ○ | ○ |
| PDC_HP9 | | | ○ | ○ |
| PDC_HP10 | | | ○ | ○ |
| Reserved bit | | Must always be set to "0". | × | ○ |
| PD1 | PDC, VPS, VBI clock phase control bit | Adjust clock phase for Data slicer.<br>Normaly, PD2 to PD1=$(10)_2$ fixed. | ○ | ○ |
| PD2 | | | ○ | ○ |
| Nothing is assigned. | | | × | × |
| Reserved bit | | Must always be set to "0". | × | ○ |
| RGBWH | RGB out put (gray display) selection bit | 0 Normal<br>1 RGB output of gray display color is white. | ○ | ○ |

### (24) Address $17_{16}$ ( = DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| VPS_HP3 | VPS and VBI slice check start position selection bit | If VPS and VBI slice check start position is VPS_HS,<br>$VPS\_HS = T2 \times \sum_{n=3}^{10} 2^{(n-3)} VPS\_HPn$<br>T2 : VPS or VBI clock run-in cycle ÷2<br>  <br>Set to flaming code check start position<br>Set by the 200ns (1bit)....VPS<br>Set by the 800ns (1bit)....VBI | ○ | ○ |
| VPS_HP4 | | | ○ | ○ |
| VPS_HP5 | | | ○ | ○ |
| VPS_HP6 | | | ○ | ○ |
| VPS_HP7 | | | ○ | ○ |
| VPS_HP8 | | | ○ | ○ |
| VPS_HP9 | | | ○ | ○ |
| VPS_HP10 | | | ○ | ○ |
| Reserved bit | | Must always be set to "0". | × | ○ |
| Nothing is assigned. | | | × | × |
| Reserved bit | | Must always be set to "0". | × | ○ |
| CCD | CCD slicer selection bit | 0 PDC, VPS, VBI<br>1 CCD | ○ | ○ |
| Reserved bit | | Must always be set to "0". | × | ○ |
| HGSLS | Data slicer control bit | 0 PDC, VPS<br>1 VBI | ○ | ○ |
| HGSL | Data slicer control bit | Must always be set to "1". | ○ | ○ |

### (25) Address 18$_{16}$ ( = DA5 to 0)



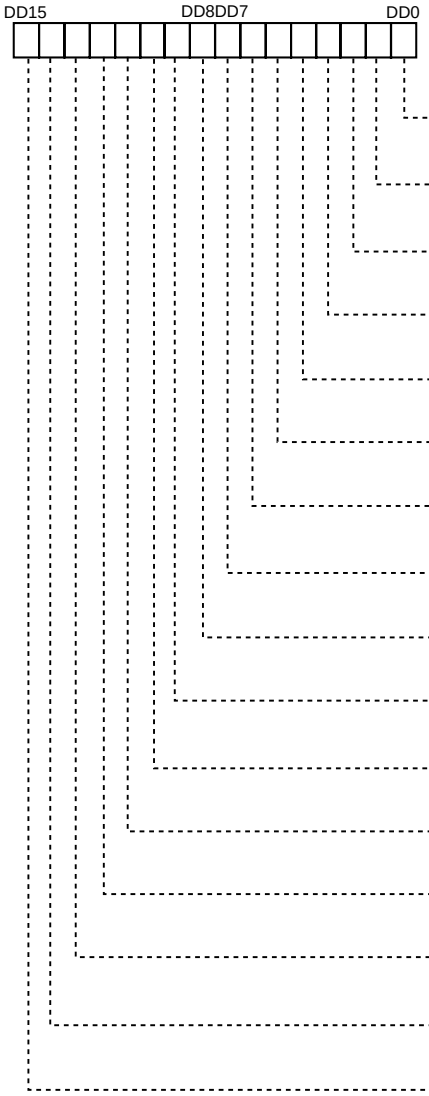

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| VBIL0 | 6th line or 318th line VBI encode selection bit | Set the line which encodes by VBIn ( n = 0 to 17).<br>It can be setin the each every line<br>VBILn: 0 = Do not set; 1 = Set<br>N : (n+6) or (n+318) | ○ | ○ |
| VBIL1 | 7th line or 319th line VBI encode selection bit | | ○ | ○ |
| VBIL2 | 8th line or 320th line VBI encode selection bit | | ○ | ○ |
| VBIL3 | 9th line or 321th line VBI encode selection bit | | ○ | ○ |
| VBIL4 | 10th line or 322th line VBI encode selection bit | | ○ | ○ |
| VBIL5 | 11th line or 323th line VBI encode selection bit | | ○ | ○ |
| VBIL6 | 12th line or 324th line VBI encode selection bit | | ○ | ○ |
| VBIL7 | 13th line or 325th line VBI encode selection bit | | ○ | ○ |
| VBIL8 | 14th line or 326th line VBI encode selection bit | | ○ | ○ |
| VBIL9 | 15th line or 327th line VBI encode selection bit | | ○ | ○ |
| VBIL10 | 16th line or 328th line VBI encode selection bit | | ○ | ○ |
| VBIL11 | 17th line or 329th line VBI encode selection bit | | ○ | ○ |
| VBIL12 | 18th line or 330th line VBI encode selection bit | | ○ | ○ |
| VBIL13 | 19th line or 331th line VBI encode selection bit | | ○ | ○ |
| VBIL14 | 20th line or 332th line VBI encode selection bit | | ○ | ○ |
| VBIL15 | 21th line or 333th line VBI encode selection bit | | ○ | ○ |

| VBILn | Encode of N line |
|---|---|
| 0 | Do not set |
| 1 | Set |

### (26) Address 19$_{16}$ ( = DA5 to 0)

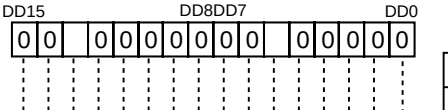

| Bit symbol | Bit name | Function | | R | W |
|---|---|---|---|---|---|
| PDCF1 | PDC data slice selection bit (field1) | 0 | Do not slice field 1 PDC data | ○ | ○ |
| | | 1 | Slice field 1 PDC data | | |
| PDCF2 | PDC data sline selection bit (field2) | 0 | Do not slice field 2 PDC data | ○ | ○ |
| | | 1 | Slice field 2 PDC data | | |
| VPSF1 | VPS data slice selection bit (field1) | 0 | Do not slice field 1 VPS data | ○ | ○ |
| | | 1 | Slice field 1 VPS data | | |
| VPSF2 | VPS data slice selection bit (field2) | 0 | Do not slice field 2 VPS data | ○ | ○ |
| | | 1 | Slice field 2 VPS data | | |
| VBIF1 | VBI data slice selection bit (field1) | 0 | Do not slice field 1 VBI data | ○ | ○ |
| | | 1 | Slice field 1 VBI data | | |
| VBIF2 | VBI data slice selection bit (field2) | 0 | Do not slice field 2 VBI data | ○ | ○ |
| | | 1 | Slice field 2 VBI data | | |
| ENCF1 | VBI data encode selection bit (field1) | 0 | Do not slice field 1 VBI data | ○ | ○ |
| | | 1 | Slice field 1 VBI data | | |
| ENCF2 | VBI data encode selection bit (field2) | 0 | Do not slice field 2 VBI data | ○ | ○ |
| | | 1 | Slice field 2 VBI data | | |
| VPSF_LINE0 | VPS data slice line selection bit | When VPS data slice line is VPS_LINES, VPS_LINES = $\sum_{n=0}^{4} 2^n$ VPS_LINEn + 7 Fix to 16th line normally. (VPS_LINE4 to VPS LINE0 = "01001$_2$" fixed) Setting value from 00000$_2$ to 10000$_2$ (7th line to 23 line) | | ○ | ○ |
| VPSF_LINE1 | | | | ○ | ○ |
| VPSF_LINE2 | | | | ○ | ○ |
| VPSF_LINE3 | | | | ○ | ○ |
| VPSF_LINE4 | | | | ○ | ○ |
| VBIL16 | 22th line or 334th line VBI encode selection bit | Set encode line by VBILn (n = 0 to 17) Refer to address 18$_{16}$ | | ○ | ○ |
| VBIL17 | 23th line or 335th line VBI encode selection bit | | | ○ | ○ |
| Reserved bit | | Must always be set to "0". | | × | ○ |

### (27) Addrres 1A16 ( = DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PDC_FLC0 | Flaming code selection bit at PDC slice | [PDC]<br>Flaming code (8 bits)<br>Clock run - in<br>Data<br>Setting<br>PDC_FLC0 to PDC_FLC7<br>PDC_FLC0 to 7 = 11100100 | ○ | ○ |
| PDC_FLC1 | | | ○ | ○ |
| PDC_FLC2 | | | ○ | ○ |
| PDC_FLC3 | | | ○ | ○ |
| PDC_FLC4 | Flaming code selection bit at VBI slice | [VBI]<br>Flaming code (24 bits)<br>Clock run - in<br>Data<br>PDC_FLC4 to 7<br>VPS_FLC0 to 7<br>Set last 8bits | ○ | ○ |
| PDC_FLC5 | | | ○ | ○ |
| PDC_FLC6 | | | ○ | ○ |
| PDC_FLC7 | | | ○ | ○ |
| VPS_FLC0 | Flaming code selection bit at VPS and VBI slice | [VPS]<br>When VPS_SUB (address12$_{16}$) = 0<br>Flaming code (16 bits)<br>Crock run - in<br>Data<br>VPS_FLC0 to VPS_FLC7<br>Set last 8bits<br>VPS_FLC0 to 7 = 10011001<br>VPS_SUB = 1<br>Flaming code (16 bits)<br>Data<br>VPS_FLC0 to 3<br>VPS_FLC4 to 7<br>(Set first 4bits) (Set last 4 bits) = 8bits<br>VPS_FLC0 to 7 = 10001001 | ○ | ○ |
| VPS_FLC1 | | | ○ | ○ |
| VPS_FLC2 | | | ○ | ○ |
| VPS_FLC3 | | | ○ | ○ |
| VPS_FLC4 | | | ○ | ○ |
| VPS_FLC5 | | | ○ | ○ |
| VPS_FLC6 | | | ○ | ○ |
| VPS_FLC7 | | | ○ | ○ |

### (28) Address 1B16 ( = DA5 to 0)



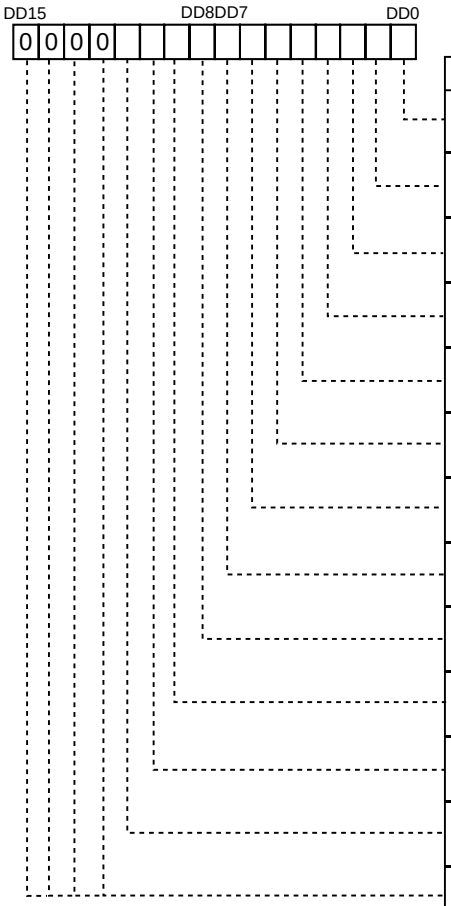

| Bit symbol | Bit name | Function | | R | W |
|---|---|---|---|---|---|
| Reserved bit | | Must always be set to "0". | | × | ○ |
| CHK_PDC5 | Flaming code check selection bit | 0 | PDC_FLC5 valid | ○ | ○ |
| | | 1 | PDC_FLC5 invalid (Note1) | | |
| Reserved bit | | Must always be set to "0". | | × | ○ |
| CHK_VPS5 | Flaming code check selection bit | 0 | VPS_FLC5 valid | ○ | ○ |
| | | 1 | VPS_FLC5 invalid (Note1) | | |
| Reserved bit | | Must always be set to "0". | | × | ○ |

Note1. At VBI slice, must be set to "1".

### (29) Address 1C16 ( = DA5 to 0)

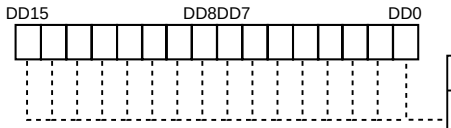

| Bit symbol | Bit name | Function | | R | W |
|---|---|---|---|---|---|
| DIV_PDCS0 | PLL control bit for PDC | Contorl the slice clock frequency $f_{PDC}$ for PDC.<br>$f_{PDC} = f_H \times (\sum_{n=0}^{8} 2^n DIV\_VPSn + \sum_{m=0} 2^{m-3} DIV\_PDCSm)$<br>$f_H$ : Horizontal synchronized signal frequency<br>When SEL_PDCH (address 0F16) = "0",<br>DIV_PDC8 to DIV_PDC0 = $(110111011)_2$<br>DIV_PDC2 to DIV_PDC0 = $(110)_2$<br>When SEL_PDCH = "1"<br>DIV_PDC8 to DIV_PDC0 = $(000010010)_2$<br>DIV_PDC2 to DIV_PDCS0 = $(101)_2$ | | ○ | ○ |
| DIV_PDCS1 | | | | ○ | ○ |
| DIV_PDCS2 | | | | ○ | ○ |
| DIV_PDC0 | PLL divided value selection bit for PDC | | | ○ | ○ |
| DIV_PDC1 | | | | ○ | ○ |
| DIV_PDC2 | | | | ○ | ○ |
| DIV_PDC3 | | | | ○ | ○ |
| DIV_PDC4 | | | | ○ | ○ |
| DIV_PDC5 | | | | ○ | ○ |
| DIV_PDC6 | | | | ○ | ○ |
| DIV_PDC7 | | | | ○ | ○ |
| DIV_PDC8 | | | | ○ | ○ |
| SELPEEK | Peek point detect selection bit | 0 | Detect from A/D data | ○ | ○ |
| | | 1 | Detect from data of digital calculation after normally "1"setting. | | |
| Reserved bit | | Must always be set to "0". | | × | ○ |

### (30) Address 1D16 ( = DA5 to 0)



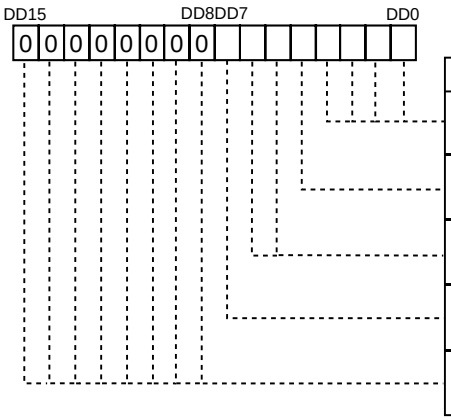

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DIV_VPSS0 | PLL control bit for VPS and VBI | Control the slice clock frequency $f_{VPS}$ for VPS and VBI.<br>$f_{PDC} = f_H \times (\sum_{n=0}^{8} 2^n DIV\_VPSn + \sum_{m=0}^{2} 2^{m-3} DIV\_VPSSm)$<br>$f_H$ : Horizontal synchronized signal frequency<br>When SEL_VPSH (address $12_{16}$) = "0",<br>DIV_VPS8 to DIV_VPS0 = $(100111111)_2$<br>DIV_VPSS2 to DIV_VPSS0 = $(110)_2$ | ○ | ○ |
| DIV_VPSS1 | | | ○ | ○ |
| DIV_VPSS2 | | | ○ | ○ |
| DIV_VPS0 | PLL divided value selection bit for VPS and VBI | When SEL_VPSH = "1",<br>DIV_VPS8 to DIV_VPS0 = $(000001111)_2$<br>DIV_VPSS2 to DIV_VPSS0 = $(110)_2$ | ○ | ○ |
| DIV_VPS1 | | | ○ | ○ |
| DIV_VPS2 | | | ○ | ○ |
| DIV_VPS3 | | | ○ | ○ |
| DIV_VPS4 | | | ○ | ○ |
| DIV_VPS5 | | | ○ | ○ |
| DIV_VPS6 | | | ○ | ○ |
| DIV_VPS7 | | | ○ | ○ |
| DIV_VPS8 | | | ○ | ○ |
| Reserved bit | | Must always be set to "0". | × | ○ |

### (31) Address 1E16 ( = DA5 to 0)



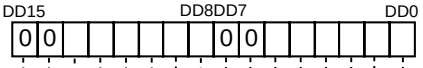

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Reserved bit | | Writing is disable.<br>Reading exclusive bit. | × | × |

### (32) Address 1F16 ( = DA5 to 0)



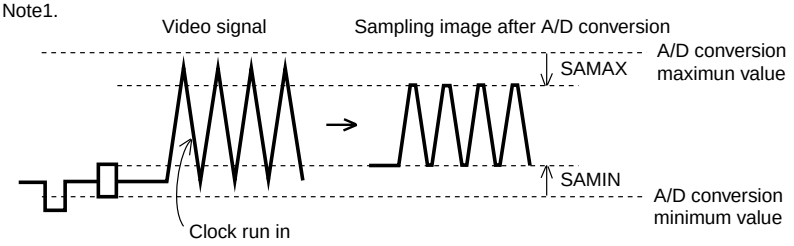

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Reserved bit | | Writing is disable.<br>Reading exclusive bit. | × | × |
| FLD | Fild flag | 0 The secound field.<br>1 The first field. | ○ | × |
| Reserved bit | | Writing is disable.<br>Reading exclusive bit. | × | × |
| MACRON | Macro vision flag | 0 No macro vision.<br>1 Macro vision | ○ | × |
| Reserved bit | | Writing is disable.<br>Reading exclusive bit. | × | × |

### (33) Address 20₁₆ ( = DA5 to 0)

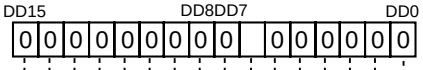



| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| MAX0 | Slice data sampling maximum value selection bit | Set slice data sampling maximum value after A/D conversion. $SAMAX = \sum_{n=0}^{5} 2^n \times MAXn$ (Note1) | ○ | ○ |
| MAX1 | | | ○ | ○ |
| MAX2 | | | ○ | ○ |
| MAX3 | | | ○ | ○ |
| MAX4 | | | ○ | ○ |
| MAX5 | | | ○ | ○ |
| Reserved bit | | Must always be set to "0". | × | ○ |
| MIN0 | Slice data sampling minimum value selection bit | Set slice data sampling minimun value after A/D conversion. $SAMIN = \sum_{n=0}^{5} 2^n \times MINn$ (Note1) | ○ | ○ |
| MIN1 | | | ○ | ○ |
| MIN2 | | | ○ | ○ |
| MIN3 | | | ○ | ○ |
| MIN4 | | | ○ | ○ |
| MIN5 | | | ○ | ○ |
| Reserved bit | | Must always be set to "0". | × | ○ |

Note1.




### (34) Address 21₁₆ ( = DA5 to 0)




| Bit symbol | Bit name | Function | | R | W |
|---|---|---|---|---|---|
| Reserved bit | | Must always be set to "0". | | × | ○ |
| DBL_HEIGHT | Double height display selection bit | 0 | Display next line, when vertical direction character size is two times. | ○ | ○ |
| | | 1 | Do not display next line, when vertical direction character size is two times. | | |
| Reserved bit | | Must always be set to "0". | | × | ○ |

### (35) Address 22₁₆ ( = DA5 to 0)


| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Reserved bit | | Must always be set to "0". | × | ○ |

### 2.15.9 Expansion Register Construction Composition

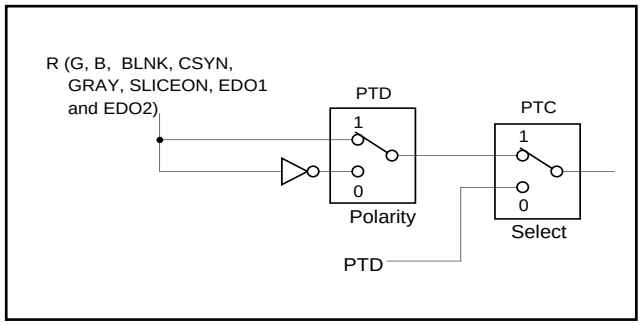



**Figure 2.15.29 Switching of port output, R, G and B output**

**Table 2.15.9 Video signal level**

| Color name | Phase (rad) | Luminance level (V) (Note1) | | | Chroma level (mV) (Note1) | | | Chroma amplitude (Notes 1 and 2) | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Min. | Typ. | Max. | Min. | Typ. | Max. | Min. | Typ. | Max. |
| Sync-chip | — | 0.90 | 1.00 | 1.10 | — | — | — | — | — | — |
| Pedestal | — | 1.50 | 1.60 | 1.70 | — | — | — | — | — | — |
| Color burst | ±4π/16 | 1.50 | 1.60 | 1.70 | 480 | 600 | 720 | — | 1.00 | — |
| Black | — | 1.50 | 1.60 | 1.70 | — | — | — | — | — | — |
| Red | ± 7π/16 ± 2π/16 | 1.70 | 1.80 | 1.90 | 1020 | 1200 | 1380 | 1.70 | 2.00 | 2.30 |
| Green | ∓5π/16 ± 2π/16 | 1.95 | 2.05 | 2.15 | 930 | 1100 | 1270 | 1.55 | 1.83 | 2.11 |
| Yellow | ± π/16 ± 2π/16 | 2.25 | 2.35 | 2.45 | 670 | 800 | 920 | 1.13 | 1.33 | 1.53 |
| Blue | ∓15π/16 ± 2π/16 | 1.60 | 1.70 | 1.80 | 670 | 800 | 920 | 1.13 | 1.33 | 1.53 |
| Magenta | ∓11π/16 ± 2π/16 | 1.80 | 1.90 | 2.00 | 930 | 1100 | 1270 | 1.55 | 1.83 | 2.11 |
| Cyan | ∓ 9π/16 ± 2π/16 | 2.10 | 2.20 | 2.30 | 1020 | 1200 | 1380 | 1.70 | 2.00 | 2.30 |
| Gray | — | 2.10 | 2.20 | 2.30 | — | — | — | — | — | — |
| White | — | 2.40 | 2.50 | 2.60 | — | — | — | — | — | — |

Notes. 1 The luminance level and the chroma amplitude of this video signal are ruled only for PAL method.
2 The chroma amplitude is ruled as shown below,
[Each color's chroma ÷ Color burst's chroma]

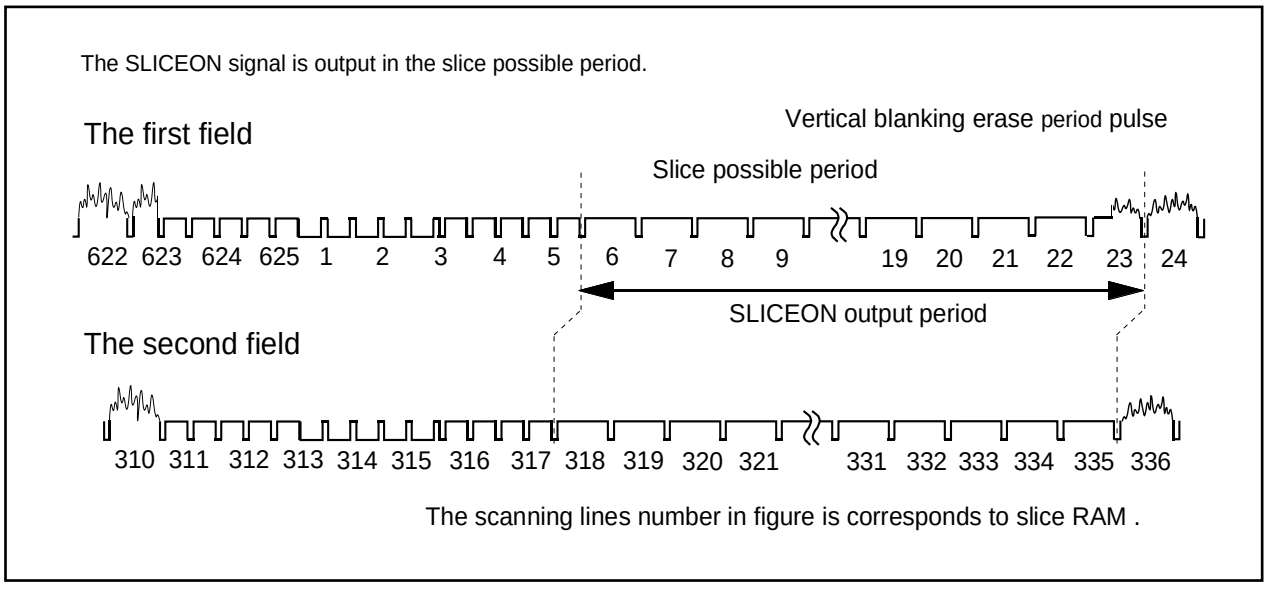



**Figure 2.15.30 Slice timing**

## 2.15.10 Display Forms

### (1) Blanking mode

Display forms are shown in Table 2.15.10, display forms at each display mode are shown in Figure 2.15.31.

**Table 2.15.26 Display forms**

| Display mode | DSP1 XX (Addresses $08_{16}$ and $09_{16}$) | DSP0 XX (Addresses $06_{16}$ and $07_{16}$) | BLNK output |
|---|---|---|---|
| Character | 0 | 0 | Character size |
| Disable | 0 | 1 | — |
| Matrix-outline | 1 | 0 | All blanking |
| Halftone | 1 | 1 | Blanking OFF |

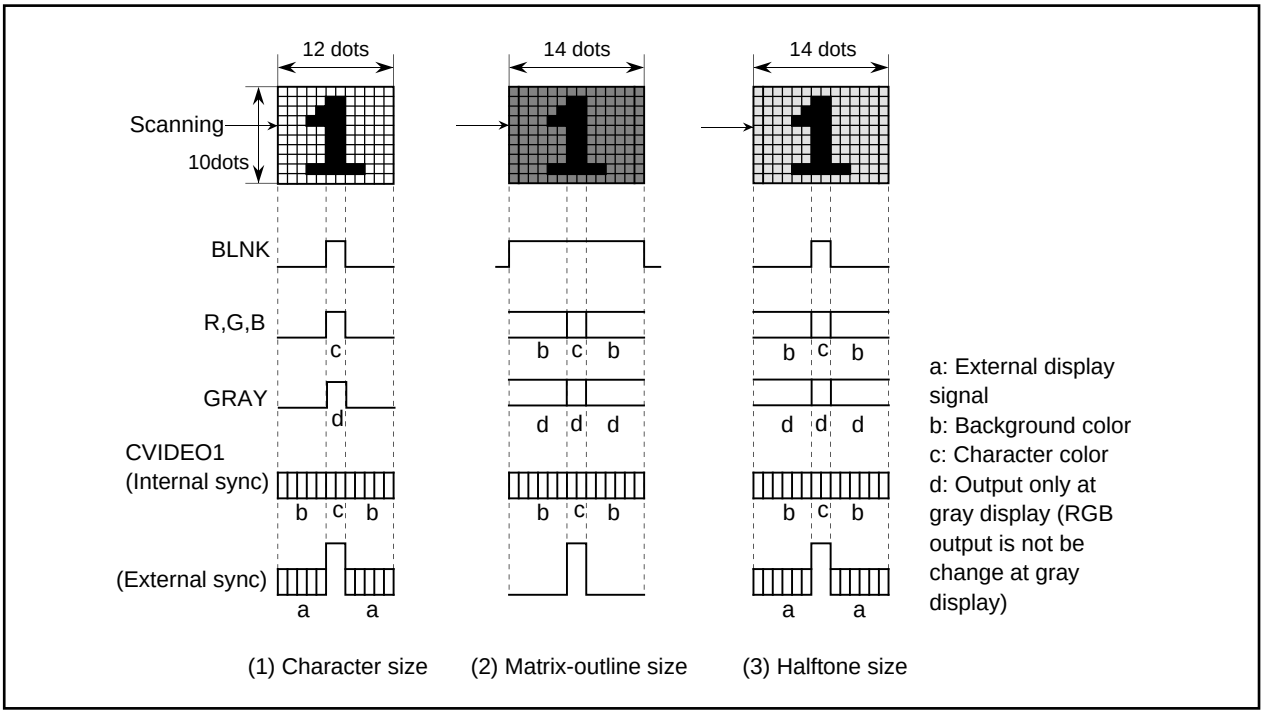



**Figure 2.15.31 Blanking mode display**

For matrix and halftone, a character's number of dots in the horizontal direction increases to 14. Figure 2.15.32 shows a display example for a case where adjacent characters have different background colors and for character code $7F_{16}$.

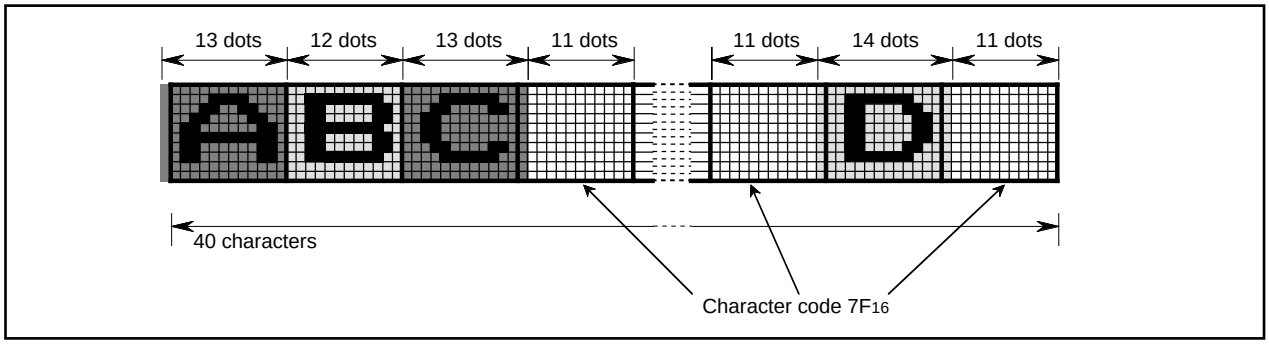



**Figure 2.15.32 Number of dots in the horizontal direction at matrix-outline or halftone**

### (2) Setting matrix outline

Matrix outline is set by using register ALL24 (address $0E_{16}$).Matrix outline can be set for each line by using the register DSP1XX (addresses $08_{16}$ and $09_{16}$) .

However, this setting is disabled if the register EX (address $0D_{16}$) is 0 (external sync). An example of setting example of all matrix-outline area is shown in Figure 2.15.33.

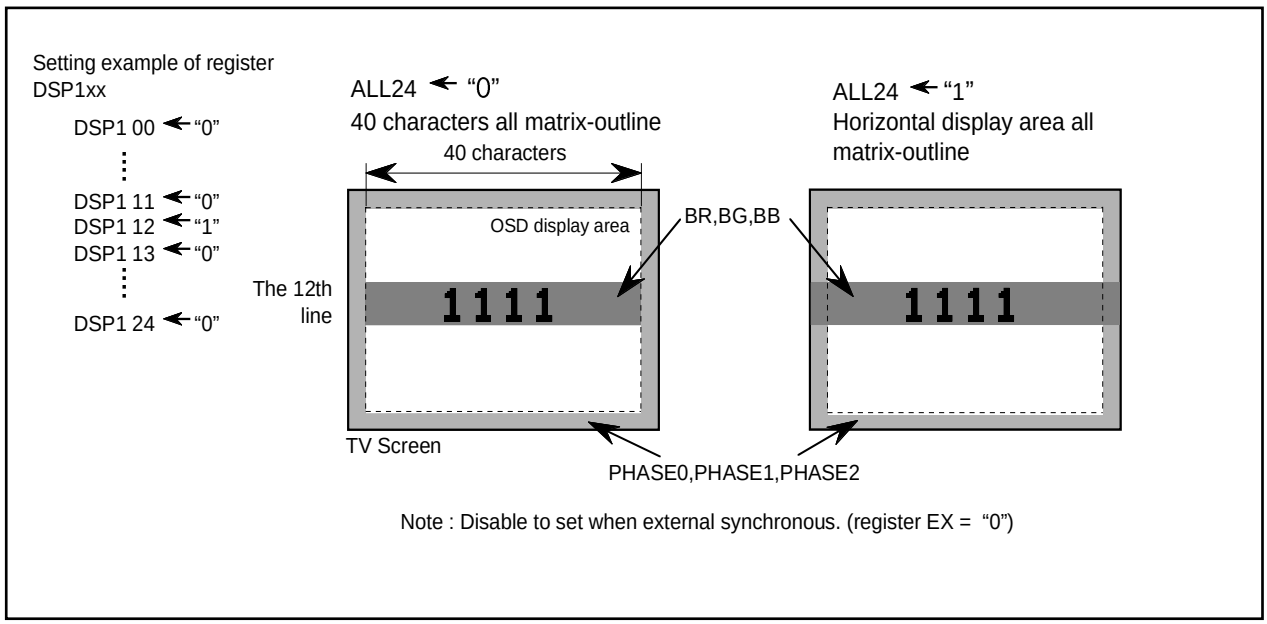



**Figure 2.15.33 Setting example of all matrix-outline area**

### (3) Blinking mode

Blinking by BLINK bit of display RAM.

And, use registers BLINK0, 1, and 2 (address $05_{16}$) to set the duty ratio and period that determines the blinking time.

Blinking mode is shown in Table 2.15.11(SYRAM do not blink).

The register settings and the duty ratio and period are shown in tables 2.15.12 and 2.15.13.

**Table 2.15.11 Blinking mode**

| Blinking mode | at blinking OFF |
|---|---|
| Blinking | |

**Table 2.15.12 Setting of duty ratio**

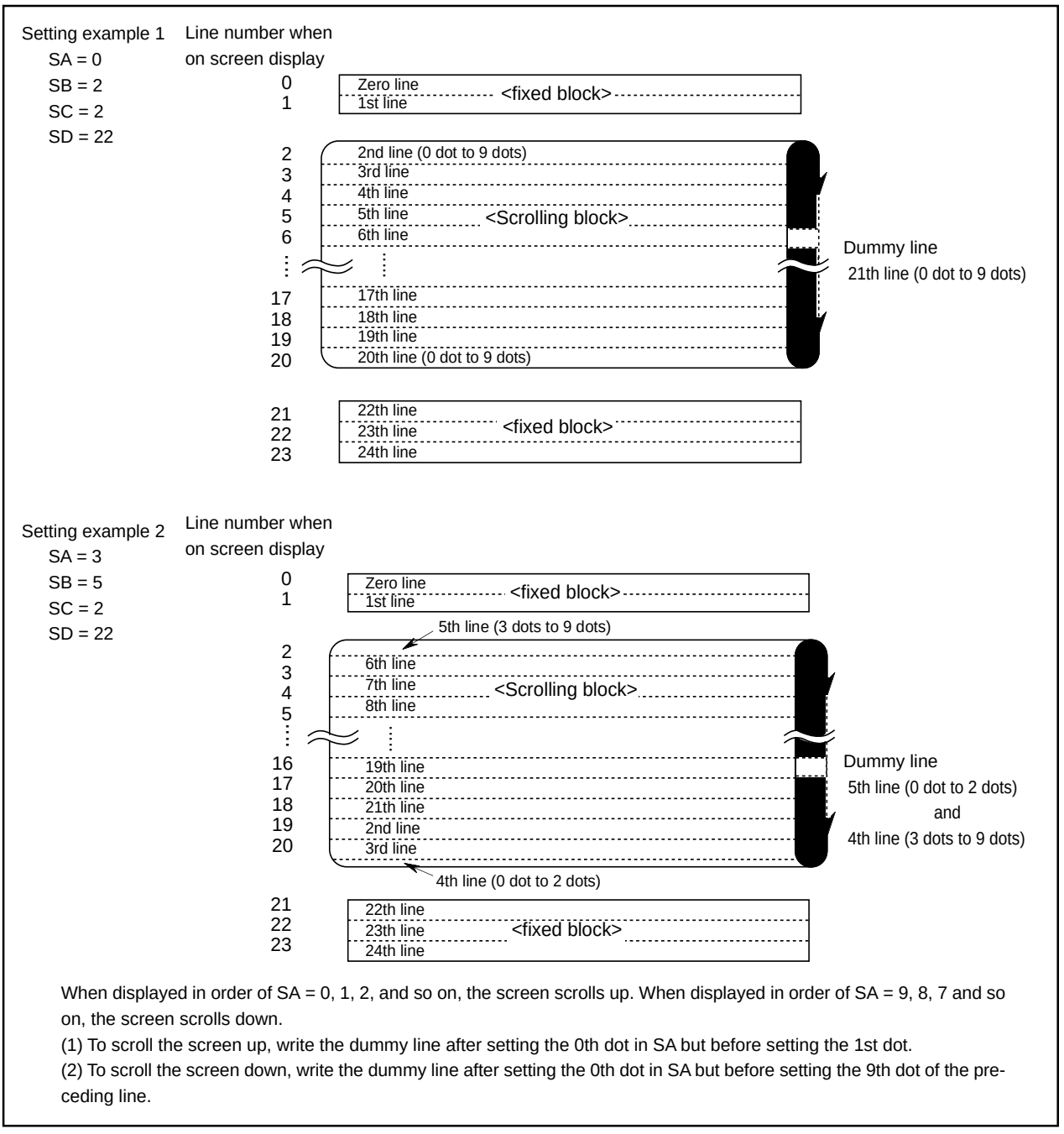
| BLINK1 \ BLINK0 | 0 | 1 |
|---|---|---|
| 0 | Blink OFF | Duty 25% |
| 1 | Duty 50% | Duty 75% |

**Table 2.15.13 Setting of cycle**

| BLINK2 | Cycle |
|---|---|
| 0 | Approximately 1 second (Vertical sync divided into 1/64) |
| 1 | Approximately 0.5 second (Vertical sync divided into 1/32) |

### (4) Scroll display mode

The scroll display mode is entered by setting registers SBIT0 to 3 (SA), SLIN0 to 4 (SB) (address $0A_{16}$), SST0 to 4 (SC), and SEND0 to 4 (SD) (address $0B_{16}$). (Scroll is turned off when SD = 0.)
The screen is scrolled in the range from the (SC)'th line to the (SD-1)'th line, and sections above and below this range are fixed. The beginning line and beginning dot of scroll are the (SA)'th dot on the (SB)'th line.
The screen can be scrolled up or down by successively incrementing or decrementing SA and SB.
Figure 2.15.34 shows examples of how the display is scrolled. The scroll range in these examples contains 20 lines (second to the 21th lines). However, the screen can display only 19 lines at a time, and the remaining one line is handled as a dummy line and not displayed.




When displayed in order of SA = 0, 1, 2, and so on, the screen scrolls up. When displayed in order of SA = 9, 8, 7 and so on, the screen scrolls down.
(1) To scroll the screen up, write the dummy line after setting the 0th dot in SA but before setting the 1st dot.
(2) To scroll the screen down, write the dummy line after setting the 0th dot in SA but before setting the 9th dot of the preceding line.

**Figure 2.15.34 Scrolling example**

### 2.15.11 8/4 Humming Decoder

8/4 humming decoder opetates only by written the data which 8/4 humming- decoded to 8/4 humming register (address 021A16). 8/4 humming register consists of 16 bits, can decode two data at a time. Can obtain the decoded result by reading 8/4 humming register, and the decoded value and error information are output. Corrects and outputs the decoded value for single error, and outputs only error information for double error. Decoded result is shown in Figure 2.15.35 and humming 8/4 register composition is shown in Figure 2.15.36.

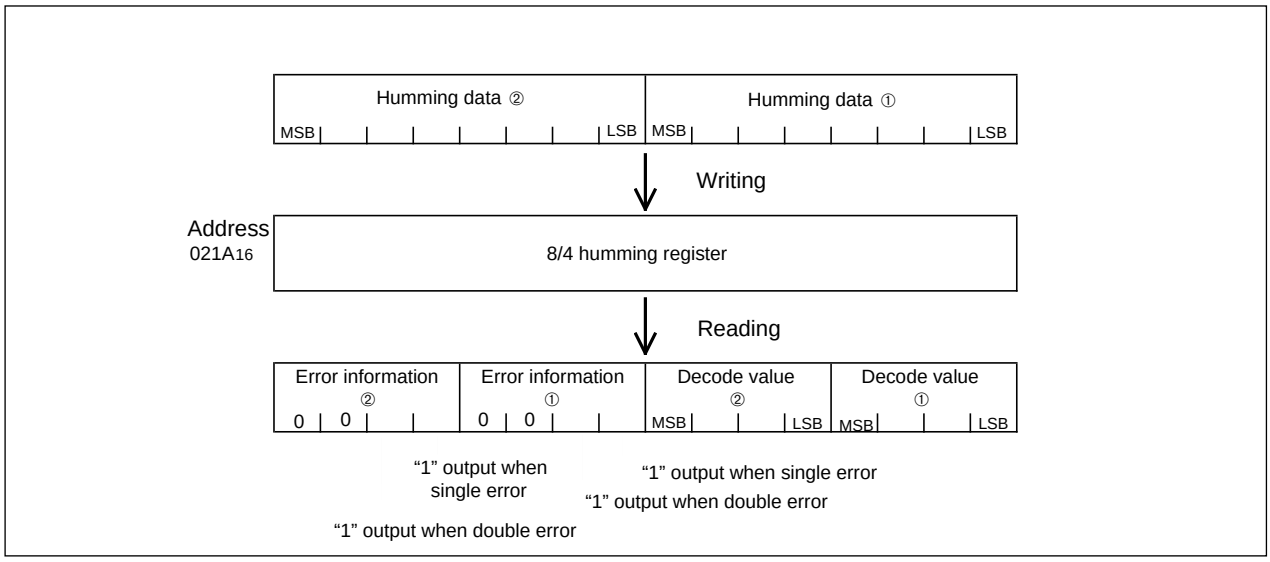



**Figure 2.15.35 Decoded result**

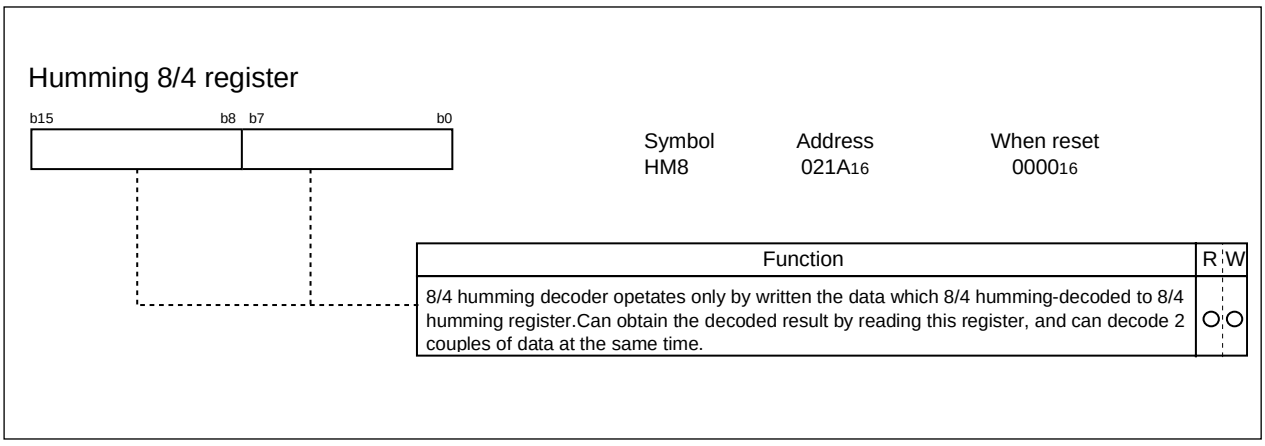



**Figure 2.15.36 Humming 8/4 register composition**

### 2.15.12 24/18Humming Decoder

24/18 humming decoder operates only by written the data which 24/18 humming-encoded to 24/18 humming register 0 (address 021C16) and 1 (address 021E16). Can obtain the decoded result by reading the same 24/18 humming register. Decoded result is shown in Figure 2.15.37 and humming 24/18 register composition is shown in Figure 2.15.38.

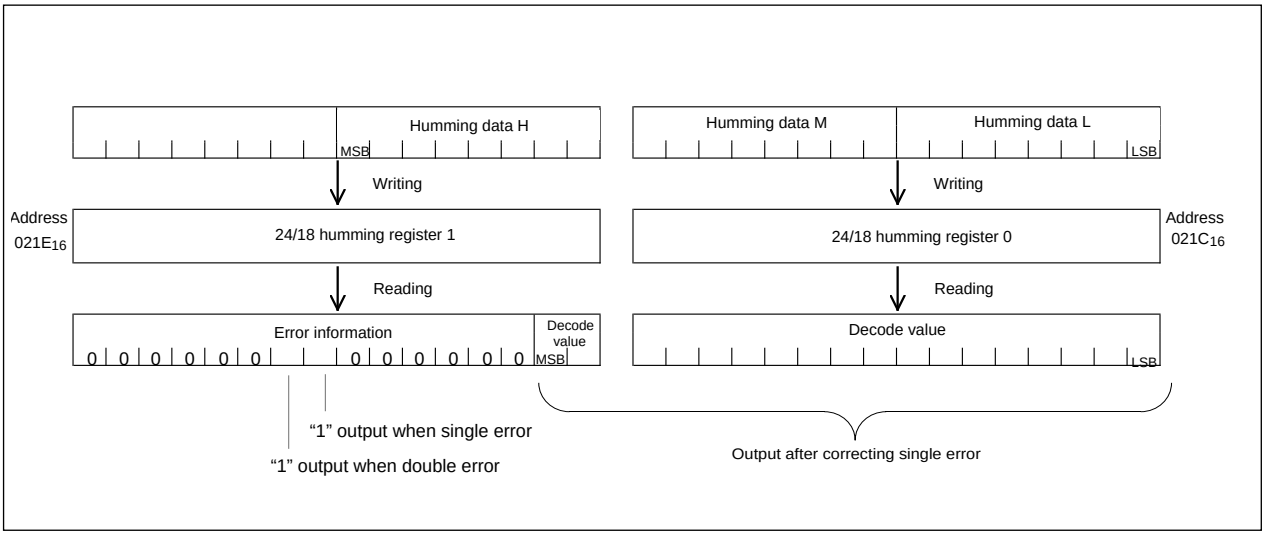



**Figure 2.15.37 Decoded result**

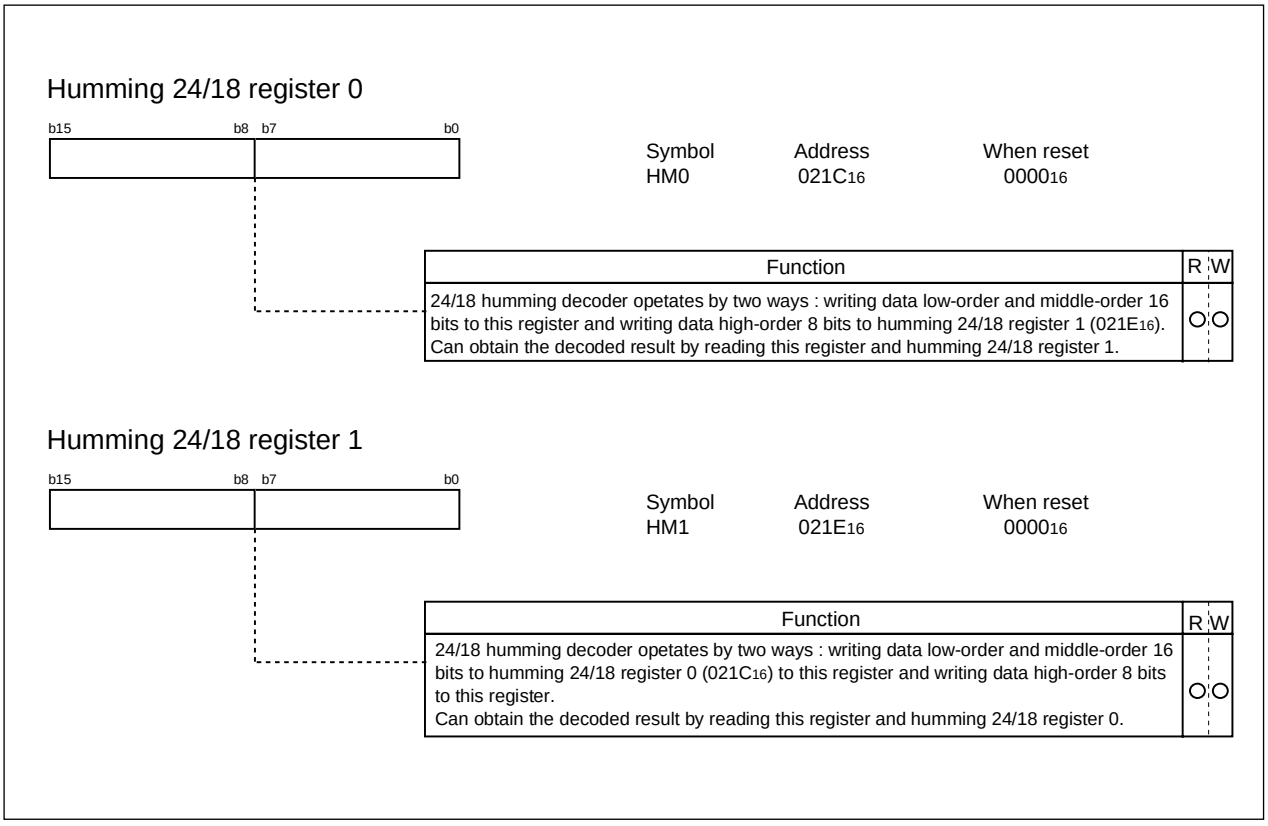



**Figure 2.15.38 Humming 24/18 register composition**

**Continuous error correction**

When uses humming 8/4 (address 021A16) at tha same time as humming 24/18, can do the continuous error correction.

Continuous error correction sequence is shown in Figure 2.15.39.

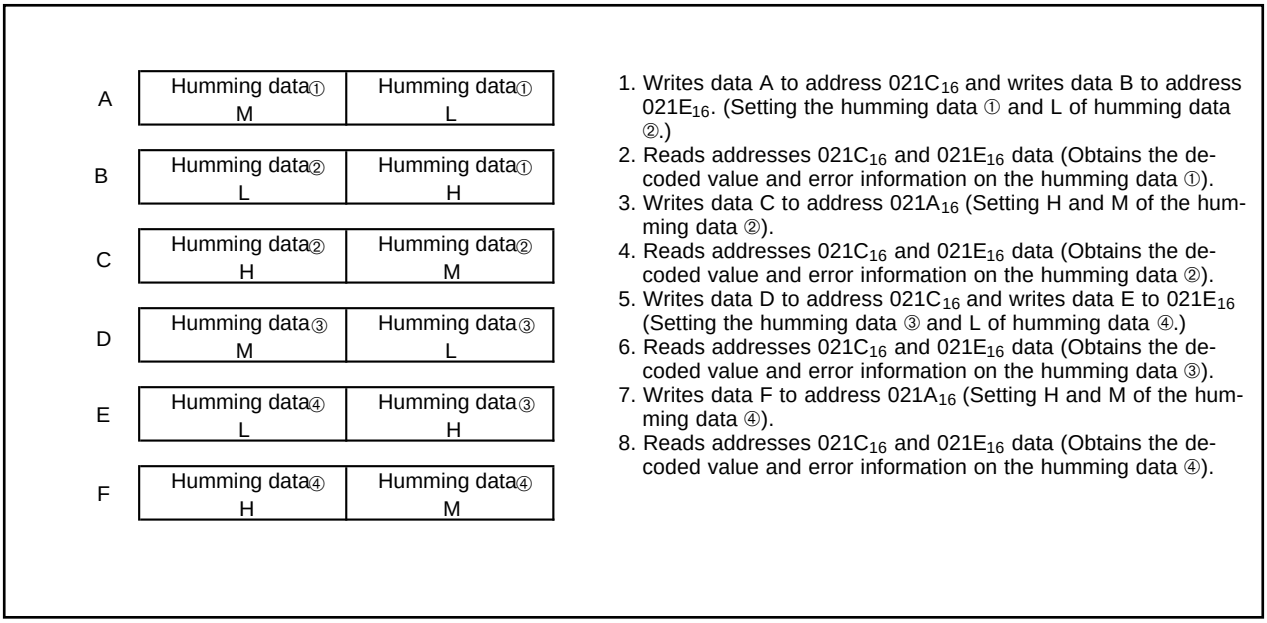



**Figure 2.15.39 Continuous error correction sequence**

Then, because using a part of circuit of humming 8/4 about this operation, cannot use this operation at the same time.

When using the humming circuit, do the decoded result reading operation at once after the setting data of humming. And do not access other memories (Including the humming circuit) before reading of the decoded result.

## 2.15.13 I/O Composition of pins for Expansion Memory

Figure 2.15.40 and figure 2.15.41 show pins for expansion memory.

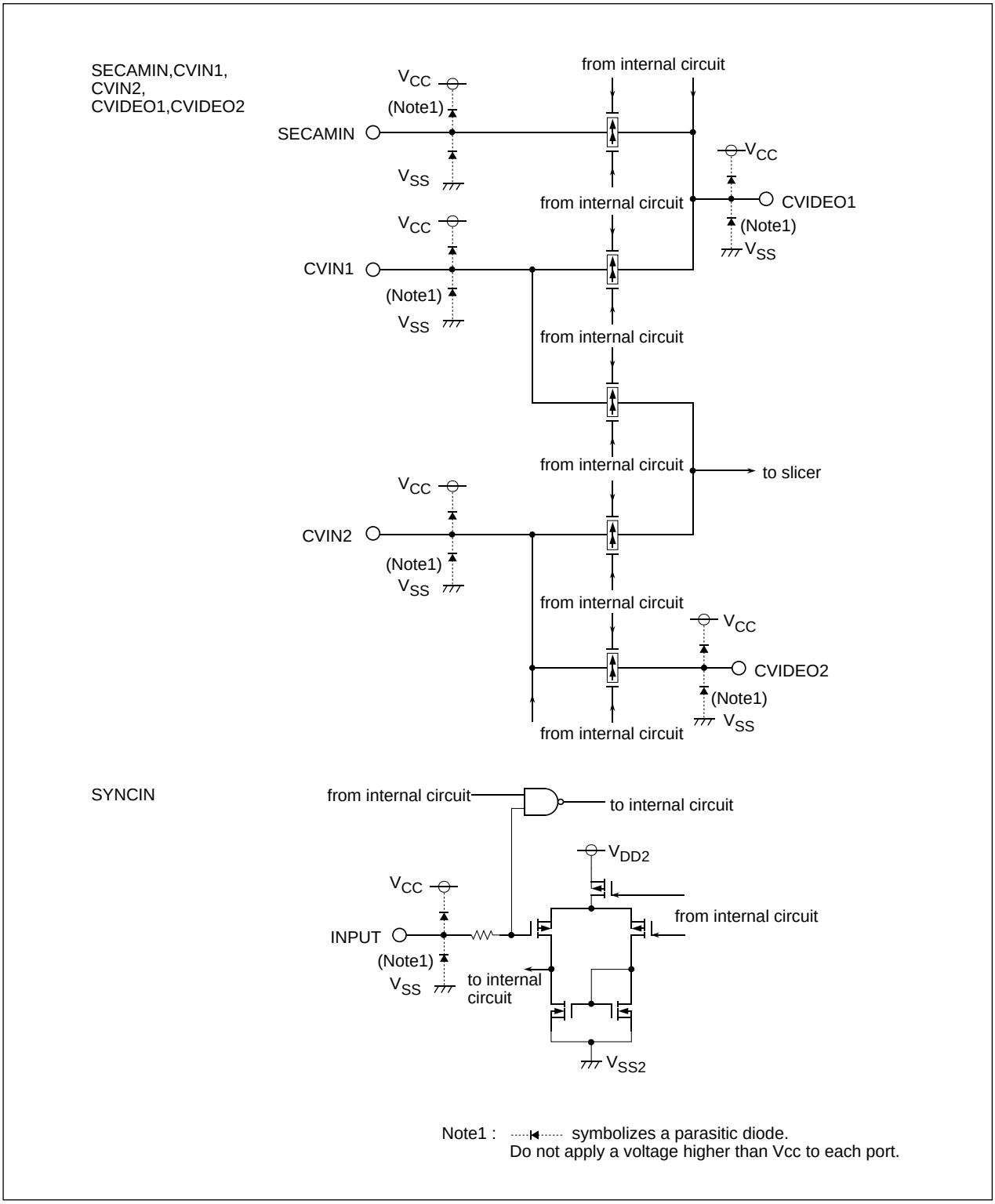
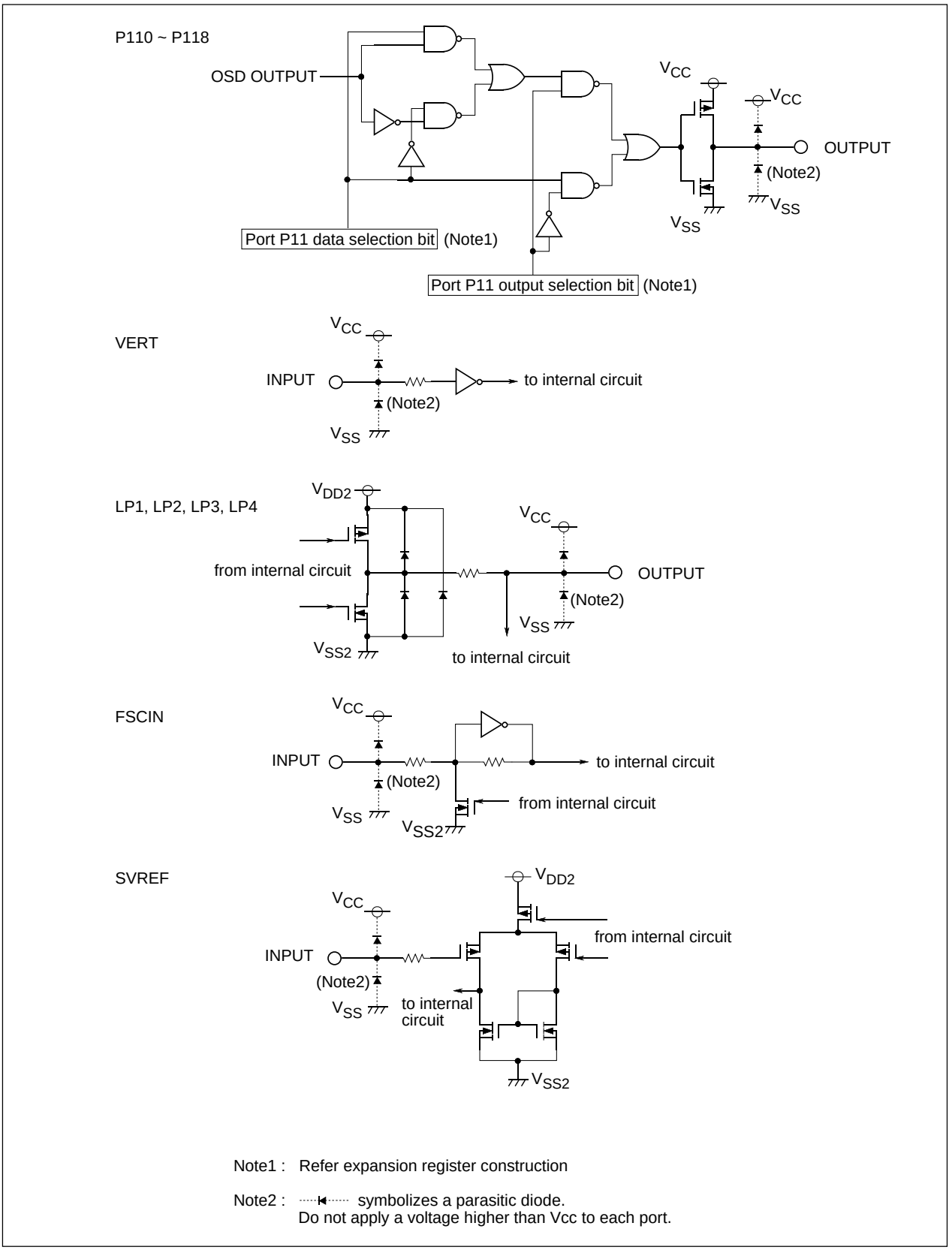



**Figure 2.15.40 Pins for expansion memory(1)**




Note1 : Refer expansion register construction

Note2 : ⋯⋯ symbolizes a parasitic diode.
Do not apply a voltage higher than Vcc to each port.

**Figure 2.15.41 Pins for expansion memory(2)**

## 2.16 Programmable I/O Ports

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

Figures 2.16.1 to 2.16.4 show the programmable I/O ports. Figure 2.16.5 shows the I/O pins.

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

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

### (1) Direction registers

Figure 2.16.6 shows the direction registers.

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

Note: There is no direction register bit for P85.

### (2) Port registers

Figure 2.16.7 shows the port registers.

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

### (3) Pull-up control registers

Figure 2.16.8 shows the pull-up control registers.

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

However, the pull-up control register of P0 to P5 is invalid.

### (4) Port control register

Figure 2.16.9 shows the port control register.

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

0 : When port P1 is input port, port input level is read.

When port P1 is output port , the contents of port P1 register is read.

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

This register is valid in the following:

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

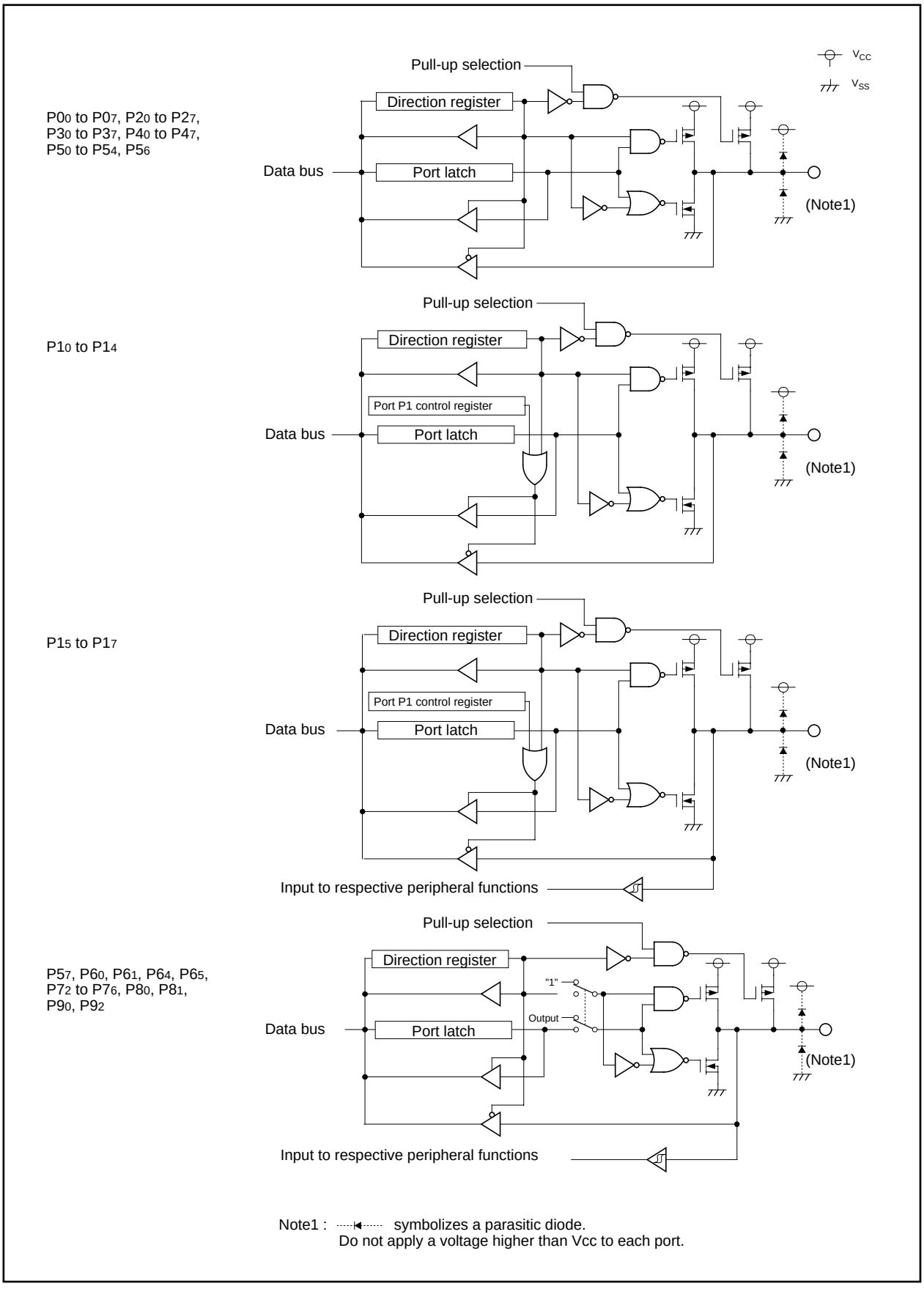



**Figure 2.16.1 Programmable I/O ports (1)**

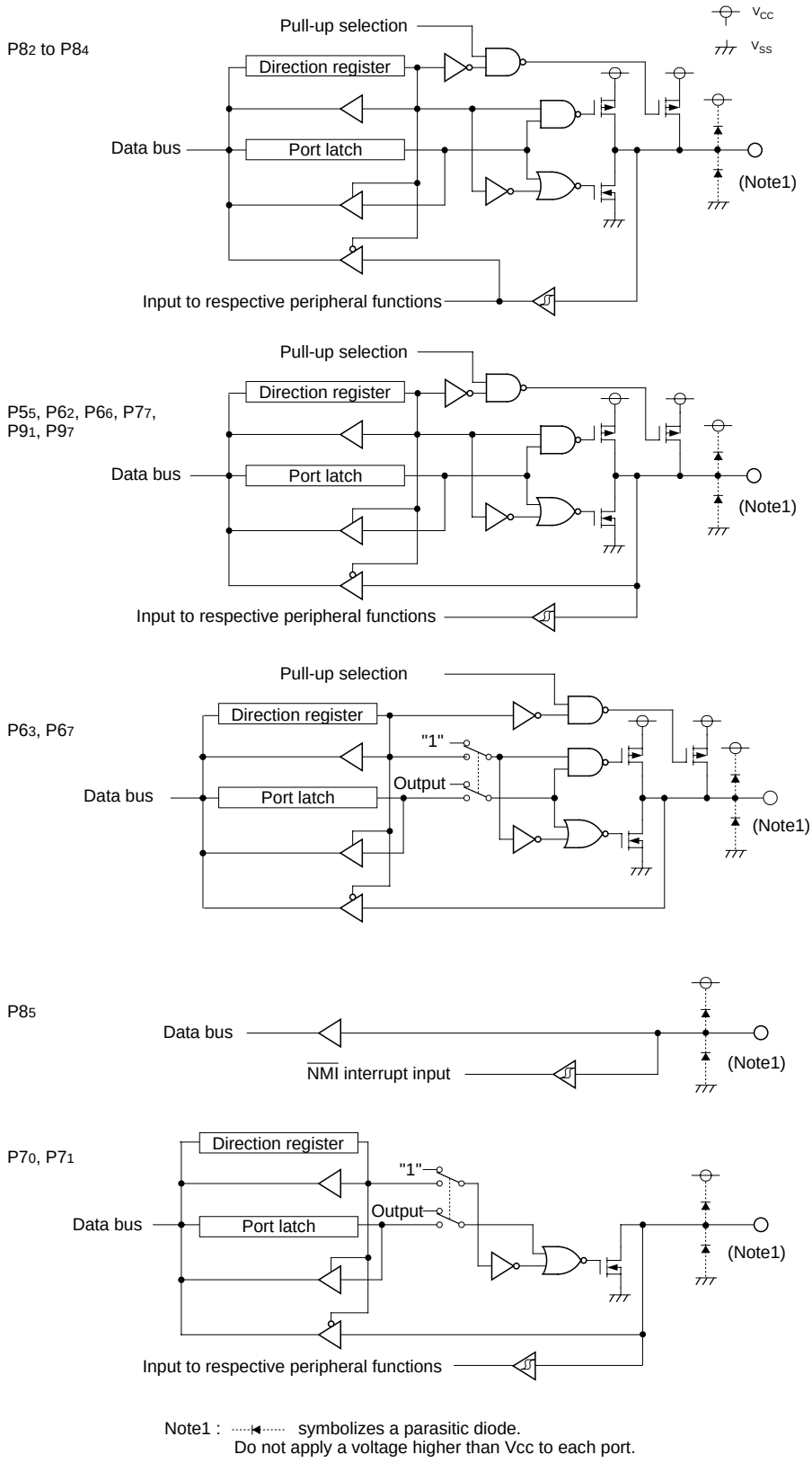



**Figure 2.16.2 Programmable I/O ports (2)**

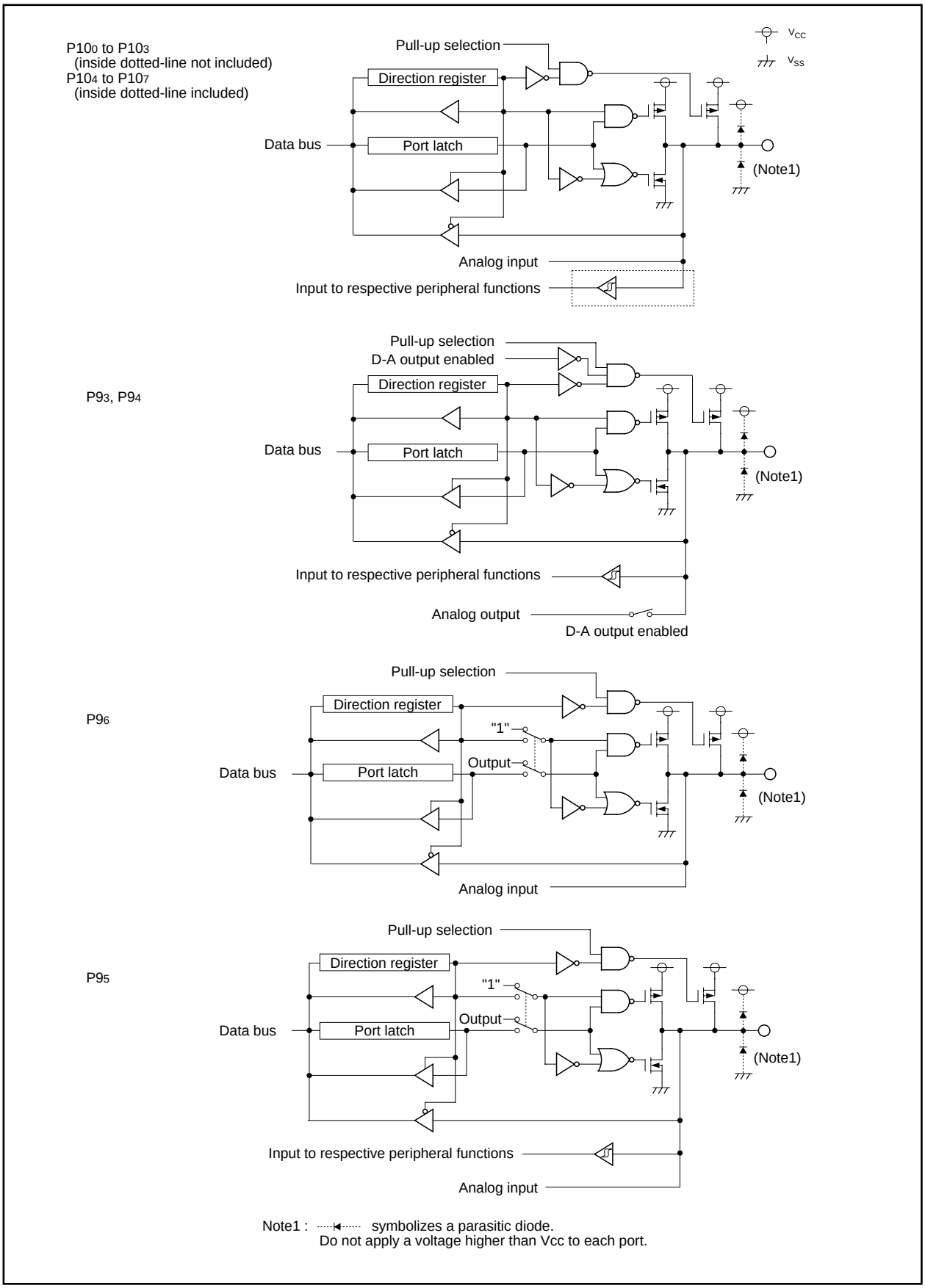



**Figure 2.16.3 Programmable I/O ports (3)**

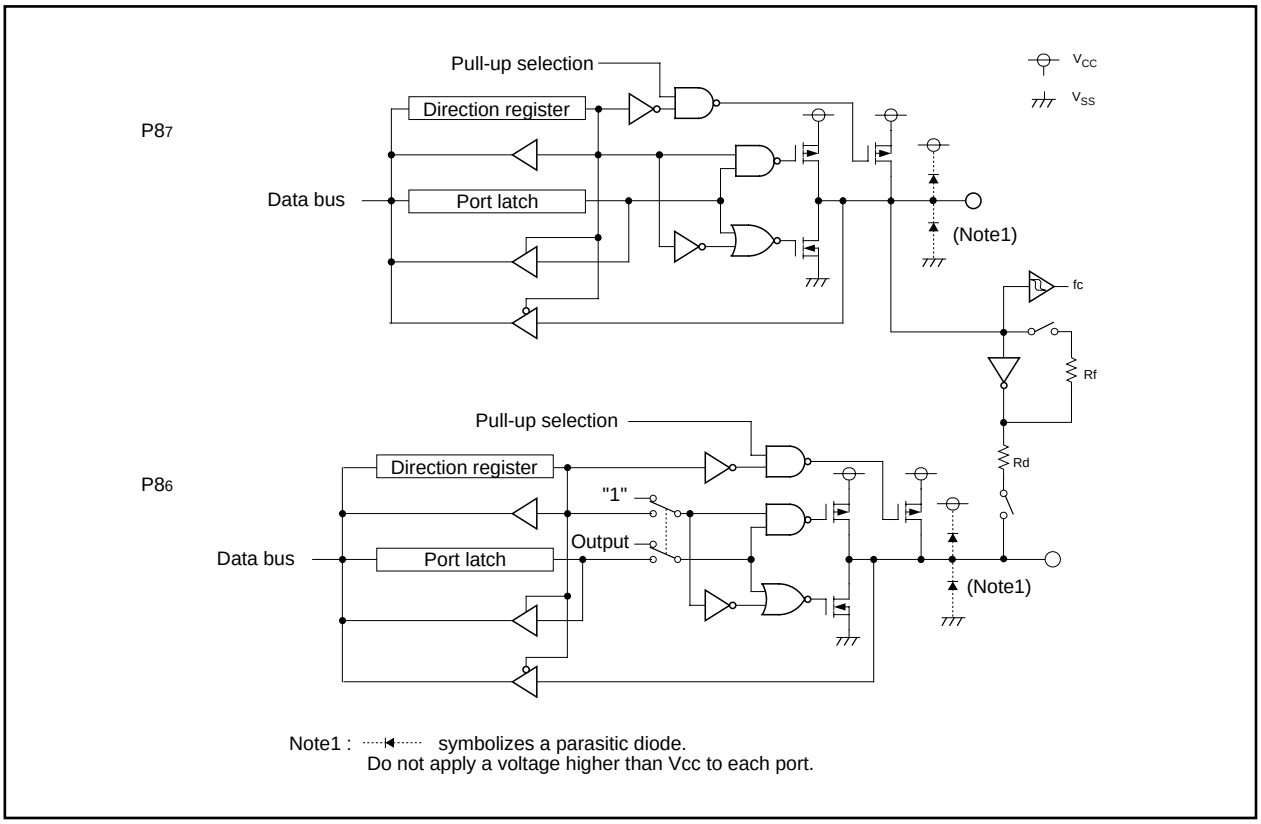



**Figure 2.16.4 Programmable I/O ports (4)**

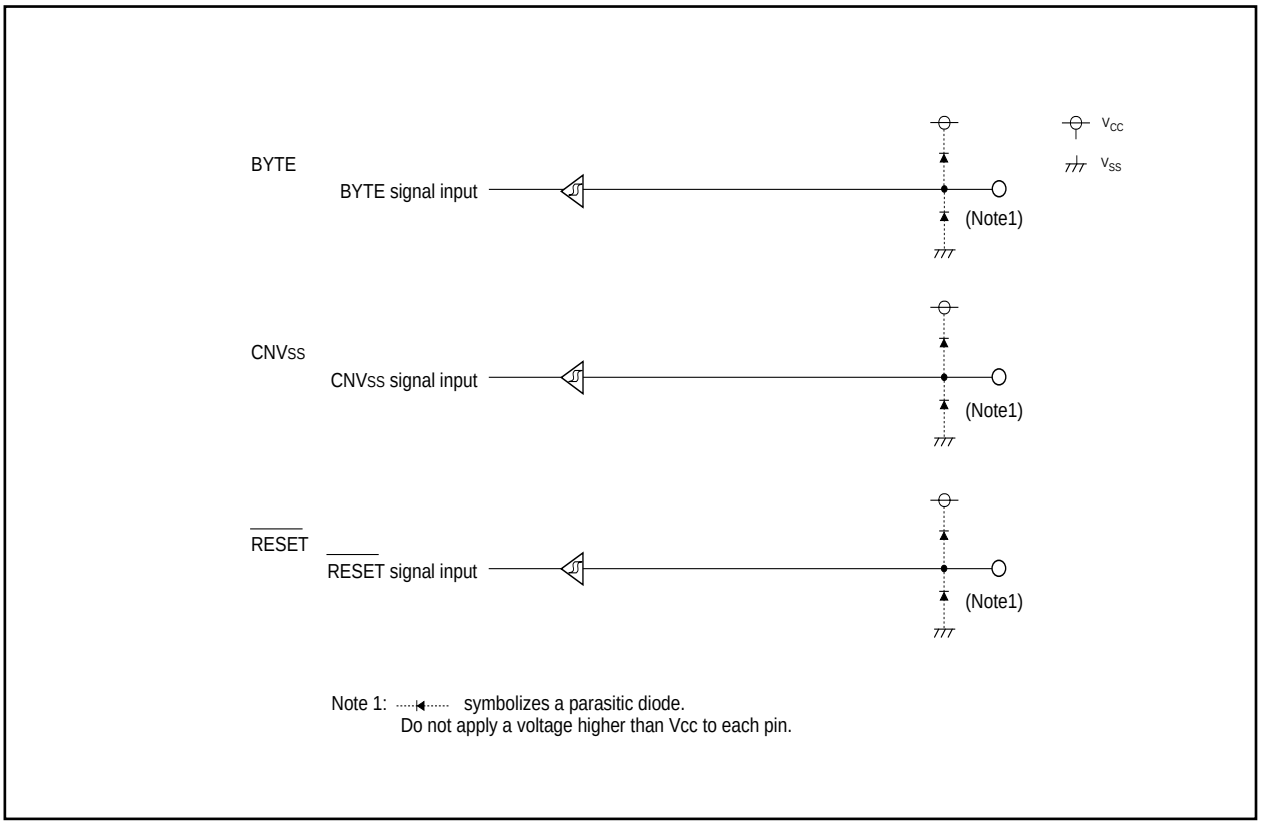
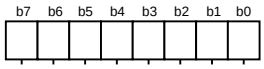



**Figure 2.16.5 I/O pins**

Port Pi direction register (Note)

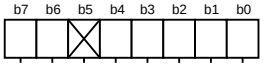



| Symbol | Address | When reset |
|---|---|---|
| PDi (i = 0 to 10, except 8) | 03E2$_{16}$, 03E3$_{16}$, 03E6$_{16}$, 03E7$_{16}$, 03EA$_{16}$ 03EB$_{16}$, 03EE$_{16}$, 03EF$_{16}$, 03F3$_{16}$, 03F6$_{16}$ | 00$_{16}$ |

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

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

Port P8 direction register




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

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

**Figure 2.16.6 Direction register**

Port Pi register

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

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

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Pi_0 | Port Pi$_0$ register | Data is input and output to and from each pin by reading and writing to and from each corresponding bit<br>0 : "L" level data<br>1 : "H" level data (Note)<br><br>(i = 0 to 10 except 8) | ○ | ○ |
| Pi_1 | Port Pi$_1$ register | | ○ | ○ |
| Pi_2 | Port Pi$_2$ register | | ○ | ○ |
| Pi_3 | Port Pi$_3$ register | | ○ | ○ |
| Pi_4 | Port Pi$_4$ register | | ○ | ○ |
| Pi_5 | Port Pi$_5$ register | | ○ | ○ |
| Pi_6 | Port Pi$_6$ register | | ○ | ○ |
| Pi_7 | Port Pi$_7$ register | | ○ | ○ |

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

Port P8 register

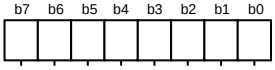
| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

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

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| P8_0 | Port P8$_0$ register | Data is input and output to and from each pin by reading and writing to and from each corresponding bit (except for P8$_5$)<br>0 : "L" level data<br>1 : "H" level data | ○ | ○ |
| P8_1 | Port P8$_1$ register | | ○ | ○ |
| P8_2 | Port P8$_2$ register | | ○ | ○ |
| P8_3 | Port P8$_3$ register | | ○ | ○ |
| P8_4 | Port P8$_4$ register | | ○ | ○ |
| P8_5 | Port P8$_5$ register | | ○ | × |
| P8_6 | Port P8$_6$ register | | ○ | ○ |
| P8_7 | Port P8$_7$ register | | ○ | ○ |

**Figure 2.16.7 Port register**

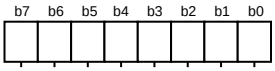
Pull-up control register 0




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

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

Pull-up control register 1

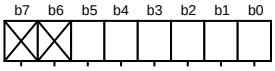



| Symbol | Address | When reset |
|---|---|---|
| PUR1 | 03FD16 | 0016 (Note 2) |

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

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

Pull-up control register 2

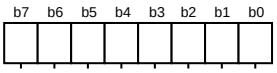



| Symbol | Address | When reset |
|---|---|---|
| PUR2 | 03FE16 | 0016 |

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

**Figure 2.16.8 Pull-up control register**

Port control register




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

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

**Figure 2.16.9 Port control register**

**Table 2.16.1 Example connection of unused pins.**

| Pin name | Connection |
|---|---|
| Ports P6 to P10 (excluding $P8_5$) | After setting for input mode, connect every pin to $V_{SS}$ or $V_{CC}$ via a resistor; or after setting for output mode, leave these pins open. |
| $P4_5$ ,$P4_6$/$\overline{CS2}$, $P4_7$/$\overline{CS3}$ | Sets ports to input mode, sets bits $\overline{CS2}$, $\overline{CS3}$ to 0, and connects to Vcc via resistors (pull-up). |
| $\overline{BHE}$, ALE, $\overline{HLDA}$, $X_{OUT}$(Note), BCLK | Open |
| $\overline{HOLD}$, $\overline{RDY}$, $\overline{NMI}$ | Connect via resistor to $V_{CC}$ (pull-up) |
| $AV_{CC}$ | Connect to $V_{CC}$ |
| $AV_{SS}$, $V_{REF}$ | Connect to $V_{SS}$ |
| $CNV_{SS}$ | Connect via resistor to $V_{CC}$ (pull-up) |

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

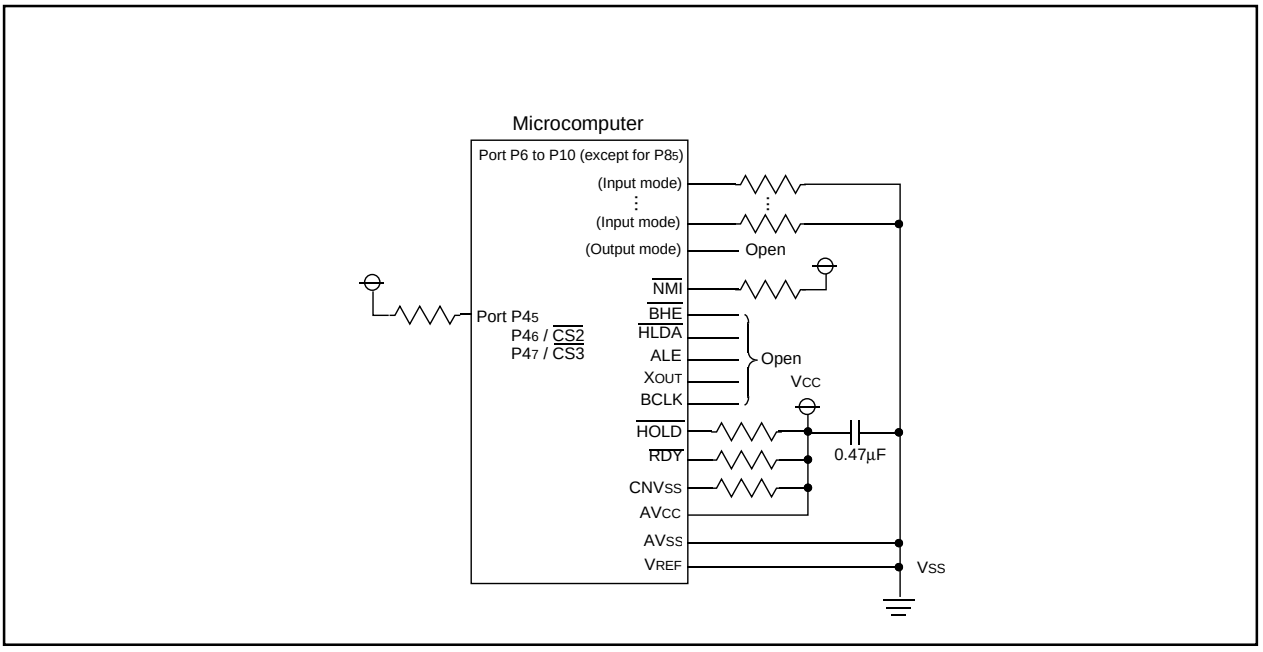



**Figure 2.16.10 Example connection of unused pins**

## 3. Usage Precaution

### Timer A (timer mode)

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

### Timer A (event counter mode)

(1) Reading the timer Ai register while a count is in progress allows reading, with arbitrary timing, the value of the counter. Reading the timer Ai register with the reload timing gets "FFFF$_{16}$" by underflow or "0000$_{16}$" 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) In the case of using "Event counter mode" as "Free-Run type" for timer A, the timer register contents may be unkown when counting begins. If the timer register is set before counting has started, then the starting value will be unkown.

This issue will occuer only for the "Event counter mode" operating as "Free-Run type". The value of the timer register will not be unkown during counting.

### 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 TAi$_{OUT}$ pin outputs "L" level.
- The interrupt request generated and the timer Ai interrupt request bit goes to "1".

(2) The timer Ai interrupt request bit goes to "1" if the timer's operation mode is set using any of the following procedures:

- Selecting one-shot timer mode after reset.
- Changing operation mode from timer mode to one-shot timer mode.
- Changing operation mode from event counter mode to one-shot timer mode.

Therefore, to use timer Ai interrupt (interrupt request bit), set timer Ai interrupt request bit to "0" after the above listed changes have been made.

### Timer A (pulse width modulation mode)

(1) The timer Ai interrupt request bit becomes "1" if setting operation mode of the timer in compliance with any of the following procedures:

- Selecting PWM mode after reset.
- Changing operation mode from timer mode to PWM mode.
- Changing operation mode from event counter mode to PWM mode.

Therefore, to use timer Ai interrupt (interrupt request bit), set timer Ai interrupt request bit to "0" after the above listed changes have been made.

(2) Setting the count start flag to "0" while PWM pulses are being output causes the counter to stop counting. If the TAi$_{OUT}$ 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 TAi$_{OUT}$ pin is outputting an "L" level in this instance, the level does not change, and the timer Ai interrupt request bit does not becomes "1".

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

(1) Reading the timer Bi register while a count is in progress allows reading , with arbitrary timing, the value of the counter. Reading the timer Bi register with the reload timing gets "FFFF16". Reading the timer Bi register after setting a value in the timer Bi register with a count halted but before the counter starts counting gets a proper value.

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

(1) If changing the measurement mode select bit is set after a count is started, the timer Bi interrupt request bit goes to "1".

(2) When the first effective edge is input after a count is started, an indeterminate value is transferred to the reload register. At this time, timer Bi interrupt request is not generated.

### A-D Converter

(1) Write to each bit (except bit 6) of A-D control register 0, to each bit of A-D control register 1, and to bit 0 of A-D control register 2 when A-D conversion is stopped (before a trigger occurs).
In particular, when the Vref connection bit is changed from "0" to "1", start A-D conversion after an elapse of 1 μs or longer.

(2) When changing A-D operation mode, select analog input pin again.

(3) Using one-shot mode or single sweep mode
Read the correspondence A-D register after confirming A-D conversion is finished. (It is known by A-D conversion interrupt request bit.)

(4) Using repeat mode, repeat sweep mode 0 or repeat sweep mode 1
Use the undivided main clock as the internal CPU clock.

### Stop Mode and Wait Mode

(1) When returning from stop mode by hardware reset, $\overline{\text{RESET}}$ pin must be set to "L" level until main clock oscillation is stabilized.

(2) When switching to either wait mode or stop mode, instructions occupying four bytes either from the WAIT instruction or from the instruction that sets the every-clock stop bit to "1" within the instruction queue are prefetched and then the program stops. So put at least four NOPs in succession either to the WAIT instruction or to the instruction that sets the every-clock stop bit to "1".

### Interrupts

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

• When maskable interrupt is occurred, CPU read the interrupt information (the interrupt number and interrupt request level) in the interrupt sequence.
The interrupt request bit of the certain interrupt written in address 00000$_{16}$ will then be set to "0".
Reading address 00000$_{16}$ by software sets enabled highest priority interrupt source request bit to "0".
Though the interrupt is generated, the interrupt routine may not be executed.
Do not read address 00000$_{16}$ by software.

(2) Setting the stack pointer

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

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

• As for the $\overline{NMI}$ interrupt pin, an interrupt cannot be disabled. Connect it to the $V_{CC}$ pin via a resistor (pull-up) if unused. Be sure to work on it.

• Do not get either into stop mode with the $\overline{NMI}$ pin set to “L”.

(4) External interrupt

• When the polarity of the INT0 to $\overline{INT5}$ pins is changed, the interrupt request bit is sometimes set to “1”. After changing the polarity, set the interrupt request bit to “0”.

(5) Rewrite the interrupt control register

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

**Example 1:**

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

**Example 2:**

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

**Example 3:**

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

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

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

Instructions : AND, OR, BCLR, BSET

## Other Notes

(1) Timing of power supplying

The power need to supply to $V_{CC}$, $V_{DD}1$, $V_{DD}2$, $V_{DD}3$ and $AV_{CC}$ at a time. While operating, must set same voltage.

(2) Power supply noise and latch-up

In order to avoid power supply noise and latch-up, connect a bypass capacitor (more than 0.1μF) directly between the $V_{CC}$ pin and $V_{SS}$ pin, $V_{DD1}$ pin and $V_{SS1}$ pin, $V_{DD2}$ pin and $V_{SS2}$ pin, $V_{DD3}$ pin and $V_{SS3}$ pin, $AV_{CC}$ pin and $AV_{SS}$ pin using a heavy wire.

(3) After the reset

After the reset, until the oscillator circuit stabilizes, data is sometimes not set correctly in the display RAM, font RAM, SYRAM and VBIRAM. Therefore, use the following start-up procedure.

(a) Reset release.
(b) Set expansion register CK_VCO, XTAL_VCO, PDC_VCO_ON,VPS_VCO_ON = “H”. (oscillation start)
(c) Set expansion register SYNCSEP_ON0 = “H”.
(d) Set expansion register NXP = “H”.
(e) Set expansion register PCn, DIV_PDCn, DIV_PDCSn, DIV_VPSn, DIV_VPSSn.
(f) Disable data input for a 20 m sec (time enough to allow theinternal oscillator circuit to stabilize).
(g) Set other expansion registers.
(h) Set the SYRAM.
(i) Set the display RAM.
(j) Set expansionregister DSPON and DSPONV to display ON.
(k) Possible to access slice RAM.

(4) When resuming internal oscillation from the off state

The each internal oscillator circuit of expansion function stops oscillating when expansion register CK_VCO,XTAL_VCO,PDC_VCO_ON,VPS_VCO_ON = “L”.

When resuming internal oscillation from the off state, up until the oscillator circuit stabilizes, data is sometimes not set correctly in the display RAM , font RAM, SYRAM and VBIRAM. Therefore, start oscillation as follows.

(a) Set expansion register CK_VCO = “H”.
(b) Set expansion register XTAL_VCO = “H”.
(c) Set expansion register PDC_VCO_ON= “H”, VPS_VCO_ON = “H”.
(Necessity none when data sliceris not used)
(d) Wait for a 20 m sec. (time enough to allow the internal oscilla-tor circuit to stabilize)
(e) Access the other memories.

Especially, set expansion register XTAL_VCO = “H” when access to display RAM, font RAM, SYRAM, VBIRAM and slice RAM. And input 4.43 MHz sub carrier frequency clock from the FSCIN pin.

Access the memory after waiting for 20ms certaninly when resuming synchronous oscillation from the off state, and begin to input clock into the FSCIN pin.

(5) Other notes on oscillation

Make note of the fact that the internal oscillator circuit cannot stabilize in the below situations.

(a) When the external composite video signal is discontinuous. (when changing channels, etc.)

(b) When expansion register PCn setting is changed.

(c) When expansion register SYNCSEP_ON0 setting is changed.

Before changing settings, turn expansion registers DSPON and DSPONV off. Also, disable data input for 20 m sec after making settings.

(6) When no external composite video signal is input

Without a signal, characters cannot be displayed by external synchronization. Therefore, switch to internal synchronization.

(7) When signal level of the external composite video signal is extremely poor

With a weak electric field, character display is uncontrollable by external synchronization. Therefore, switch to internal synchronization.

(8) When oscillation circuit stop for data slicer

Expansion register PDC_VCO_ON,VPS_VCO_ON is set at “L”, when the data slicer is not used, and the oscillation is stopped. When starting oscillation again, set data at the folowing order.

(a) Set expansion register PDC_VCO_ON,VPS_VCO_ON = “L”.

(b) Set expansion register PDC_VCO_ON,VPS_VCO_ON = “H”.

(c) 60 ms or more is a waiting state (stability period of internal oscillation circuit + data slice preparation).

To operate slice RAM , set expansion register XTAL_VCO = “H”. And input 4.43 MHz sub carrier frequency clock from the FSCIN pin.

Access the memories after wating for 20 ms certainly when resuming synchronous oscillation from the off state , and begin to input clock into the FSCIN pin.

(9) When the data slicer is used without displaying OSD

If expansion register DSPON is set in “L”, the OSD display is turned off.

Expansion register CK_VCO must be set “H” in that case

(10) At stop mode (clock is stopped)

Set each input pins to as follows.

(a) Set VERT pin = VSS.

(b) Stop the FSCIN pin input.

(c) Set expansion register STBY0 and STBY1 = “H”.

Set all expansion registers to “L” except for the superscription register.

(11) When operation start from stop mode (clock is stopped)

Input FSCIN pin clock after set “L” to register STBY0 and STBY1.

At next, set expansion register as notes (4).

## 4. Electrical characteristic

### Table 4.1 Absolute maximum ratings

| Symbol | Parameter | | Condition | Rated value | Unit |
|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | $V_{CC}$=AV$_{CC}$ | -0.3 to 5.75 | V |
| AVcc | Analog supply voltage | | $V_{CC}$=AV$_{CC}$ | -0.3 to 5.75 | V |
| $V_I$ | Input voltage | $\overline{RESET}$, CNV$_{SS}$, BYTE, P0$_0$ to P0$_7$, P1$_0$ to P1$_7$, P2$_0$ to P2$_7$, P3$_0$ to P3$_7$, P4$_0$ to P4$_7$, P5$_0$ to P5$_7$, P6$_0$ to P6$_7$, P7$_2$ to P7$_7$, P8$_0$ to P8$_7$, P9$_0$ to P9$_7$, P10$_0$ to P10$_7$, $V_{REF}$, $X_{IN}$, HOR,VERT | | -0.3 to Vcc+0.3 | V |
| | | P7$_0$, P7$_1$ | | -0.3 to 5.75 | V |
| $V_O$ | Output voltage | P0$_0$ to P0$_7$, P1$_0$ to P1$_7$, P2$_0$ to P2$_7$, P3$_0$ to P3$_7$,P4$_0$ to P4$_7$, P5$_0$ to P5$_7$, P6$_0$ to P6$_7$,P7$_2$ to P7$_7$, P8$_0$ to P8$_4$, P8$_6$, P8$_7$, P9$_0$ to P9$_7$, P10$_0$ to P10$_7$, $X_{OUT}$, P11$_0$ to P11$_8$ | | -0.3 to Vcc+0.3 | V |
| | | P7$_0$, P7$_1$ | | -0.3 to 5.75 | V |
| $P_d$ | Power dissipation | | Ta=25°C | 1000 | mW |
| $T_{opr}$ | Operating ambient temperature | | | -20 to 70 | °C |
| $T_{stg}$ | Storage temperature | | | -40 to 125 | °C |

**Tabl 4.2 Recommended operating conditions (referenced to Vcc = 4.75V to 5.25V at Ta = – 20 to 70°C unless otherwise specified)**

| Symbol | Parameter | | | Standard | | | Unit |
|---|---|---|---|---|---|---|---|
| | | | | Min | Typ. | Max. | |
| Vcc | Supply voltage | | | 4.75 | 5.0 | 5.25 | V |
| AVcc | Analog supply voltage | | | | Vcc | | V |
| Vss | Supply voltage | | | | 0 | | V |
| AVss | Analog supply voltage | | | | 0 | | V |
| $V_{IH}$ | HIGH input voltage | $P3_1$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, $X_{IN}$, RESET, $CNV_{SS}$, BYTE, HOR, VERT | | 0.8Vcc | | Vcc | V |
| | | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ | | 0.5Vcc | | Vcc | V |
| $V_{IL}$ | LOW input voltage | $P3_1$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, $X_{IN}$, RESET, $CNV_{SS}$, BYTE, HOR, VERT | | 0 | | 0.2Vcc | V |
| | | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ | | 0 | | 0.16Vcc | V |
| $V_{CVIN}$ | Composite video input voltage | CVIN1, CVIN2 | | | 2V P-P | | V |
| $V_{FSCIN}$ | Input voltage | FSCIN(Note 1) | | 0.3V P-P | | 4.0V P-P | V |
| $I_{OH (peak)}$ | HIGH peak output current (Note 2.3) | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, $P11_0$ to $P11_8$ | | | | -10.0 | mA |
| $I_{OH (avg)}$ | HIGH average output current | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, $P11_0$ to $P11_8$ | | | | -5.0 | mA |
| $I_{OL (peak)}$ | LOW peak output current | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, $P11_0$ to $P11_8$ | | | | 10.0 | mA |
| $I_{OL (avg)}$ | LOW average output current | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, $P11_0$ to $P11_8$ | | | | 5.0 | mA |
| f ($X_{IN}$) | Main clock input oscillation frequency | No wait | Vcc=4.75V to 5.25V | 0 | | 10 | MHz |
| | | with wait | | | | | |
| f ($X_{CIN}$) | Subclock oscillation frequency | | | | 32.768 | 50 | kHz |
| f (FSCIN) | Oscillation frequency for synchronous signal(Duty 40% to 60%) | | | | 4.434 | | MHz |

Note 1: Noise component is within 30mV.

Note 2: The mean output current is the mean value within 100ms.

Note 3: The total IOL (peak) for ports P0, P1, P2, $P8_6$, $P8_7$, P9, and P10 must be 80mA max. The total $I_{OH}$ (peak) for ports P0, P1,
P2, $P8_6$, $P8_7$, P9, and P10 must be 80mA max. The total IOL (peak) for ports P3, P4, P5, P6, P7, and $P8_0$ to $P8_4$ must be
80mA max. The total IOH (peak) for ports P3, P4, P5, P6, $P7_2$ to $P7_7$, and $P8_0$ to $P8_4$ must be 80mA max.

**Table 4.3 Electrical characteristics (referenced to $V_{CC}$ = 5V, $V_{SS}$ = 0V at Ta = 25°C, f($X_{IN}$) =10MHz unless otherwise specified)**

| Symbol | Parameter | | Measuring condition | Standard | | | Unit |
|---|---|---|---|---|---|---|---|
| | | | | Min | Typ. | Max. | |
| $V_{OH}$ | HIGH output voltage | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, $P11_0$ to $P11_8$ | $I_{OH}$=-5mA | 3.0 | | | V |
| $V_{OH}$ | HIGH output voltage | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, $P11_0$ to $P11_8$ | $I_{OH}$=-200µA | 4.7 | | | V |
| $V_{OH}$ | HIGH output voltage | LP1 to LP4 | $V_{CC}$=4.75V, $I_{OH}$=-0.5mA | 3.75 | | | V |
| $V_{OH}$ | HIGH output voltage | $X_{OUT}$ HIGHPOWER | $I_{OH}$=-1mA | 3.0 | | | V |
| | | $X_{OUT}$ LOWPOWER | $I_{OH}$=-0.5mA | 3.0 | | | |
| | HIGH output voltage | $X_{COUT}$ HIGHPOWER | With no load applied | | 3.0 | | V |
| | | $X_{COUT}$ LOWPOWER | With no load applied | | 1.6 | | |
| $V_{OL}$ | LOW output voltage | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, $P11_0$ to $P11_8$ | $I_{OL}$=5mA | | | 2.0 | V |
| $V_{OL}$ | LOW output voltage | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, $P11_0$ to $P11_8$ | $I_{OL}$=200µA | | | 0.45 | V |
| $V_{OL}$ | LOW output voltage | LP1 to LP4 | $V_{CC}$=4.75V, $I_{OH}$=-0.5mA | | | 0.4 | V |
| $V_{OL}$ | LOW output voltage | $X_{OUT}$ HIGHPOWER | $I_{OL}$=1mA | | | 2.0 | V |
| | | $X_{OUT}$ LOWPOWER | $I_{OL}$=0.5mA | | | 2.0 | |
| | LOW output voltage | $X_{COUT}$ HIGHPOWER | With no load applied | | 0 | | V |
| | | $X_{COUT}$ LOWPOWER | With no load applied | | 0 | | |
| $V_{T+}$-$V_{T-}$ | Hysteresis | $\overline{HOLD}$, $\overline{RDY}$, $TA0_{IN}$ to $TA4_{IN}$, $TB0_{IN}$ to $TB2_{IN}$, $\overline{INT_0}$ to $\overline{INT_5}$, $\overline{ADTRG}$, $\overline{CTS_1}$, $CLK_1$, $\overline{NMI}$ $TA2_{OUT}$ to $TA4_{OUT}$,$\overline{KI_0}$ to $\overline{KI_3}$ | | 0.2 | | 0.8 | V |
| $V_{T+}$-$V_{T-}$ | Hysteresis | $\overline{CTS_0}$, $CLK_0$ | | 0.2 | | 1.4 | V |
| $V_{T+}$-$V_{T-}$ | Hysteresis | $\overline{RESET}$ | | 0.2 | | 1.8 | V |
| $I_{IH}$ | HIGH input current | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, $X_{IN}$, RESET, CNVss, BYTE, HOR, VERT | $V_I$=5V | | | 5.0 | µA |
| $I_{IL}$ | LOW input current | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, $X_{IN}$, RESET, CNVss, BYTE, HOR, VERT | $V_I$=0V | | | -5.0 | µA |
| $R_{PULLUP}$ | Pull-up resistance | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$ | $V_I$=0V | 30.0 | 50.0 | 167.0 | kΩ |
| $V_{SYNCIN}$ | Sync voltage amplitude | | | 0.3 | 0.6 | 1.2 | V |
| $V_{dat(text)}$ | Teletext data voltage amplitude | | | 0.6 | 0.9 | 1.4 | V |
| △ f/ f | Range for display oscillator circuit | | | ±7 | | | % |
| $f_H$ | Horizontal synchronous signal frequency | | | 14.6 | 15.625 | 17.0 | kHz |

**Table 4.4 Electrical characteristics (referenced to $V_{CC}$ = 5V, $V_{SS}$ = 0V at Ta = 25°C, f($X_{IN}$) = 10MHz unless otherwise specified)**

| Symbol | Parameter | Measuring condition | Standard | | | Unit |
|---|---|---|---|---|---|---|
| | | | Min | Typ. | Max. | |
| $R_{fXIN}$ | Feedback resistance $X_{IN}$ | | | 1.0 | | MΩ |
| $R_{fXCIN}$ | Feedback resistance $X_{CIN}$ | | | 6.0 | | MΩ |
| $V_{RAM}$ | RAM retention voltage | When clock is stopped | 2.0 | | | V |
| $I_{cc}$ | Power supply current | When OSD operate, f($X_{IN}$)=10MHz | | 150 | 180 | mA |
| | | When clock is stopped | | | 3 | mA |

**Tabl 4.5 Video signal input conditions ($V_{CC}$ = 5.0V, Ta = –20 to 70°C)**

| Symbol | Parameter | Measuring condition | Standard | | | Unit |
|---|---|---|---|---|---|---|
| | | | Min | Typ. | Max. | |
| $V_{IN-cu}$ | Composite video signal input clamp voltage | Sync-chip voltage | | 1.0 | | V |

**Table 4.6 A-D conversion characteristics (referenced to $V_{CC}$ = $AV_{CC}$ = $V_{REF}$ = 5V, $V_{SS}$ = $AV_{SS}$ = 0V at Ta = 25°C, f($X_{IN}$) = 10MHz unless otherwise specified)**

| Symbol | Parameter | | Measuring condition | Standard | | | Unit |
|---|---|---|---|---|---|---|---|
| | | | | Min. | Typ. | Max. | |
| – | Resolution | | $V_{REF}$ = $V_{CC}$ | | | 8 | Bits |
| – | Absolute accuracy | Sample & hold function not available | $V_{REF}$ = $V_{CC}$ = 5V | | | ±3 | LSB |
| | | Sample & hold function available(8bit) | $V_{REF}$ = $V_{CC}$ = 5V | | | ±2 | LSB |
| $R_{LADDER}$ | Ladder resistance | | $V_{REF}$ = $V_{CC}$ | 10 | | 40 | kΩ |
| $t_{CONV}$ | Conversion time(8bit) | | | 2.8 | | | µs |
| $t_{SAMP}$ | Sampling time | | | 0.3 | | | µs |
| $V_{REF}$ | Reference voltage | | | 2 | | $V_{CC}$ | V |
| $V_{IA}$ | Analog input voltage | | | 0 | | $V_{REF}$ | V |

**Table 4.7 D-A conversion characteristics (referenced to $V_{CC}$ = 5V, $V_{SS}$ = $AV_{SS}$ = 0V, $V_{REF}$ = 5V at Ta = 25°C, f($X_{IN}$) = 10MHz unless otherwise specified)**

| Symbol | Parameter | Measuring condition | Standard | | | Unit |
|---|---|---|---|---|---|---|
| | | | Min. | Typ. | Max. | |
| – | Resolution | | | | 8 | Bits |
| – | Absolute accuracy | | | | 1.0 | % |
| $t_{su}$ | Setup time | | | | 3 | µs |
| $R_O$ | Output resistance | | 4 | 10 | 20 | kΩ |
| $I_{VREF}$ | Reference power supply input current | (Note) | | | 1.5 | mA |

Note: This applies when using one D-A converter, with the D-A register for the unused D-A converter set to "$00_{16}$".
The A-D converter's ladder resistance is not included.
Also, when the Vref is unconnected at the A-D control register, $I_{VREF}$ is sent.

**Timing requirements (referenced to $V_{CC}$ = 5V, $V_{SS}$ = 0V at Ta = 25°C unless otherwise specified)**

**Table 4.8 External clock input**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_c$ | External clock input cycle time | 100 | | ns |
| $t_{w(H)}$ | External clock input HIGH pulse width | 40 | | ns |
| $t_{w(L)}$ | External clock input LOW pulse width | 40 | | ns |
| $t_r$ | External clock rise time | | 18 | ns |
| $t_f$ | External clock fall time | | 18 | ns |

**Table 4.9 $\overline{RDY}$, $\overline{HOLD}$, $\overline{HLDA}$ imput**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{ac1(RD-DB)}$ | Data input access time (no wait) | | (Note) | ns |
| $t_{ac2(RD-DB)}$ | Data input access time (with wait) | | (Note) | ns |
| $t_{ac3(RD-DB)}$ | Data input access time (when accessing multiplex bus area) | | (Note) | ns |
| $t_{su(DB-RD)}$ | Data input setup time | 40 | | ns |
| $t_{su(RDY-BCLK)}$ | $\overline{RDY}$ input setup time | 30 | | ns |
| $t_{su(HOLD-BCLK)}$ | $\overline{HOLD}$ input setup time | 40 | | ns |
| $t_{h(RD-DB)}$ | Data input hold time | 0 | | ns |
| $t_{h(BCLK-RDY)}$ | $\overline{RDY}$ input hold time | 0 | | ns |
| $t_{h(BCLK-HOLD)}$ | $\overline{HOLD}$ input hold time | 0 | | ns |
| $t_{d(BCLK-HLDA)}$ | $\overline{HLDA}$ output delay time | | 40 | ns |

Note: Calculated according to the BCLK frequency as follows:

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

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

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

**Timing requirements (referenced to $V_{CC}$ = 5V, $V_{SS}$ = 0V at Ta = 25°C unless otherwise specified)**

**Table 4.10 Timer A input (counter input in event counter mode)**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TA)}$ | $TAi_{IN}$ input cycle time | 100 | | ns |
| $t_{w(TAH)}$ | $TAi_{IN}$ input HIGH pulse width | 40 | | ns |
| $t_{w(TAL)}$ | $TAi_{IN}$ input LOW pulse width | 40 | | ns |

**Table 4.11 Timer A input (gating input in timer mode)**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TA)}$ | $TAi_{IN}$ input cycle time | 400 | | ns |
| $t_{w(TAH)}$ | $TAi_{IN}$ input HIGH pulse width | 200 | | ns |
| $t_{w(TAL)}$ | $TAi_{IN}$ input LOW pulse width | 200 | | ns |

**Table 4.12 Timer A input (external trigger input in one-shot timer mode)**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TA)}$ | $TAi_{IN}$ input cycle time | 200 | | ns |
| $t_{w(TAH)}$ | $TAi_{IN}$ input HIGH pulse width | 100 | | ns |
| $t_{w(TAL)}$ | $TAi_{IN}$ input LOW pulse width | 100 | | ns |

**Table 4.13 Timer A input (external trigger input in pulse width modulation mode)**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{w(TAH)}$ | $TAi_{IN}$ input HIGH pulse width | 100 | | ns |
| $t_{w(TAL)}$ | $TAi_{IN}$ input LOW pulse width | 100 | | ns |

**Table 4.14 Timer A input (up/down input in event counter mode)**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(UP)}$ | $TAi_{OUT}$ input cycle time | 2000 | | ns |
| $t_{w(UPH)}$ | $TAi_{OUT}$ input HIGH pulse width | 1000 | | ns |
| $t_{w(UPL)}$ | $TAi_{OUT}$ input LOW pulse width | 1000 | | ns |
| $t_{su(UP-TIN)}$ | $TAi_{OUT}$ input setup time | 400 | | ns |
| $t_{h(TIN-UP)}$ | $TAi_{OUT}$ input hold time | 400 | | ns |

**Timing requirements (referenced to $V_{CC}$ = 5V, $V_{SS}$ = 0V at Ta = 25°C unless otherwise specified)**

### Table 4.15 Timer B input (counter input in event counter mode)

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TB)}$ | TBiIN input cycle time (counted on one edge) | 100 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width (counted on one edge) | 40 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width (counted on one edge) | 40 | | ns |
| $t_{c(TB)}$ | TBiIN input cycle time (counted on both edges) | 200 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width (counted on both edges) | 80 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width (counted on both edges) | 80 | | ns |

### Table 4.16 Timer B input (pulse period measurement mode)

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TB)}$ | TBiIN input cycle time | 400 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width | 200 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width | 200 | | ns |

### Table 4.17 Timer B input (pulse width measurement mode)

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TB)}$ | TBiIN input cycle time | 400 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width | 200 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width | 200 | | ns |

### Table 4.18 A-D trigger input

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(AD)}$ | $\overline{ADTRG}$ input cycle time (trigger able minimum) | 1000 | | ns |
| $t_{w(ADL)}$ | $\overline{ADTRG}$ input LOW pulse width | 125 | | ns |

### Table 4.19 Serial I/O

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

### Table 4.20 External interrupt $\overline{INTi}$ inputs

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

**Switching characteristics (referenced to $V_{CC}$ = 5V, $V_{SS}$ = 0V at Ta = 25°C, CM15 = "1" unless otherwise specified)**

**Table 4.21 No wait**

| Symbol | Parameter | Measuring condition | Standard | | Unit |
|---|---|---|---|---|---|
| | | | Min. | Max. | |
| $t_{d(BCLK-AD)}$ | Address output delay time | Figure 4.1 | | 25 | ns |
| $t_{h(BCLK-AD)}$ | Address output hold time (BCLK standard) | | 4 | | ns |
| $t_{h(RD-AD)}$ | Address output hold time (RD standard) | | 0 | | ns |
| $t_{h(WR-AD)}$ | Address output hold time (WR standard) | | 0 | | ns |
| $t_{d(BCLK-CS)}$ | Chip select output delay time | | | 25 | ns |
| $t_{h(BCLK-CS)}$ | Chip select output hold time (BCLK standard) | | 4 | | ns |
| $t_{d(BCLK-ALE)}$ | ALE signal output delay time | | | 25 | ns |
| $t_{h(BCLK-ALE)}$ | ALE signal output hold time | | – 4 | | ns |
| $t_{d(BCLK-RD)}$ | RD signal output delay time | | | 25 | ns |
| $t_{h(BCLK-RD)}$ | RD signal output hold time | | 0 | | ns |
| $t_{d(BCLK-WR)}$ | WR signal output delay time | | | 25 | ns |
| $t_{h(BCLK-WR)}$ | WR signal output hold time | | 0 | | ns |
| $t_{d(BCLK-DB)}$ | Data output delay time (BCLK standard) | | | 40 | ns |
| $t_{h(BCLK-DB)}$ | Data output hold time (BCLK standard) | | 4 | | ns |
| $t_{d(DB-WR)}$ | Data output delay time (WR standard) | | (Note1) | | ns |
| $t_{h(WR-DB)}$ | Data output hold time (WR standard)(Note2) | | 0 | | ns |

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

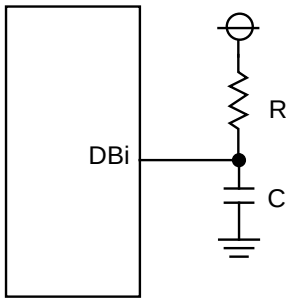
$$td(DB - WR) = \frac{10^9}{f(BCLK) \times 2} - 40 \quad [ns]$$

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

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

by a circuit of the right figure.
For example, when $V_{OL} = 0.2V_{CC}$, C = 30pF, R = 1kΩ, hold time of output "L" level is

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



**Switching characteristics (refer to $V_{CC}$ = 5V, $V_{SS}$ = 0V at Ta = 25°C, CM15 = "1" unless otherwise specified)**

**Table 4.22 With wait, accessing external memory**

| Symbol | Parameter | Measuring condition | Standard | | Unit |
|---|---|---|---|---|---|
| | | | Min. | Max. | |
| $t_{d(BCLK-AD)}$ | Address output delay time | Figure 4.1 | | 25 | ns |
| $t_{h(BCLK-AD)}$ | Address output hold time (BCLK standard) | | 4 | | ns |
| $t_{h(RD-AD)}$ | Address output hold time (RD standard) | | 0 | | ns |
| $t_{h(WR-AD)}$ | Address output hold time (WR standard) | | 0 | | ns |
| $t_{d(BCLK-CS)}$ | Chip select output delay time | | | 25 | ns |
| $t_{h(BCLK-CS)}$ | Chip select output hold time (BCLK standard) | | 4 | | ns |
| $t_{d(BCLK-ALE)}$ | ALE signal output delay time | | | 25 | ns |
| $t_{h(BCLK-ALE)}$ | ALE signal output hold time | | – 4 | | ns |
| $t_{d(BCLK-RD)}$ | RD signal output delay time | | | 25 | ns |
| $t_{h(BCLK-RD)}$ | RD signal output hold time | | 0 | | ns |
| $t_{d(BCLK-WR)}$ | WR signal output delay time | | | 25 | ns |
| $t_{h(BCLK-WR)}$ | WR signal output hold time | | 0 | | ns |
| $t_{d(BCLK-DB)}$ | Data output delay time (BCLK standard) | | | 40 | ns |
| $t_{h(BCLK-DB)}$ | Data output hold time (BCLK standard) | | 4 | | ns |
| $t_{d(DB-WR)}$ | Data output delay time (WR standard) | | (Note1) | | ns |
| $t_{h(WR-DB)}$ | Data output hold time (WR standard)(Note2) | | 0 | | ns |

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

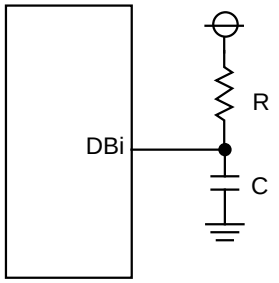
$$td(DB - WR) = \frac{10^9}{f(BCLK)} - 40 \quad [ns]$$

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

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

by a circuit of the right figure.
For example, when $V_{OL} = 0.2V_{CC}$, C = 30pF, R = 1kΩ, hold time of output "L" level is

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



**Switching characteristics (referenced to $V_{CC}$ = 5V, $V_{SS}$ = 0V at Ta = 25°C, CM15 = "1" unless otherwise specified)**

**Table 4.23 With wait, accessing external memory, multiplex bus area selected**

| Symbol | Parameter | Measuring condition | Standard | | Unit |
|---|---|---|---|---|---|
| | | | Min. | Max. | |
| $t_{d(BCLK-AD)}$ | Address output delay time | Figure 4.1 | | 25 | ns |
| $t_{h(BCLK-AD)}$ | Address output hold time (BCLK standard) | | 4 | | ns |
| $t_{h(RD-AD)}$ | Address output hold time (RD standard) | | (Note) | | ns |
| $t_{h(WR-AD)}$ | Address output hold time (WR standard) | | (Note) | | ns |
| $t_{d(BCLK-CS)}$ | Chip select output delay time | | | 25 | ns |
| $t_{h(BCLK-CS)}$ | Chip select output hold time (BCLK standard) | | 4 | | ns |
| $t_{h(RD-CS)}$ | Chip select output hold time (RD standard) | | (Note) | | ns |
| $t_{h(WR-CS)}$ | Chip select output hold time (WR standard) | | (Note) | | ns |
| $t_{d(BCLK-RD)}$ | RD signal output delay time | | | 25 | ns |
| $t_{h(BCLK-RD)}$ | RD signal output hold time | | 0 | | ns |
| $t_{d(BCLK-WR)}$ | WR signal output delay time | | | 25 | ns |
| $t_{h(BCLK-WR)}$ | WR signal output hold time | | 0 | | ns |
| $t_{d(BCLK-DB)}$ | Data output delay time (BCLK standard) | | | 40 | ns |
| $t_{h(BCLK-DB)}$ | Data output hold time (BCLK standard) | | 4 | | ns |
| $t_{d(DB-WR)}$ | Data output delay time (WR standard) | | (Note) | | ns |
| $t_{h(WR-DB)}$ | Data output hold time (WR standard) | | (Note) | | ns |
| $t_{d(BCLK-ALE)}$ | ALE signal output delay time (BCLK standard) | | | 25 | ns |
| $t_{h(BCLK-ALE)}$ | ALE signal output hold time (BCLK standard) | | – 4 | | ns |
| $t_{d(AD-ALE)}$ | ALE signal output delay time (Address standard) | | (Note) | | ns |
| $t_{h(ALE-AD)}$ | ALE signal output hold time (Adderss standard) | | 50 | | ns |
| $t_{d(AD-RD)}$ | Post-address RD signal output delay time | | 0 | | ns |
| $t_{d(AD-WR)}$ | Post-address WR signal output delay time | | 0 | | ns |
| $t_{dZ(RD-AD)}$ | Address output floating start time | | | 8 | ns |

Note: Calculated according to the BCLK frequency as follows:

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

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

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

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

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

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

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

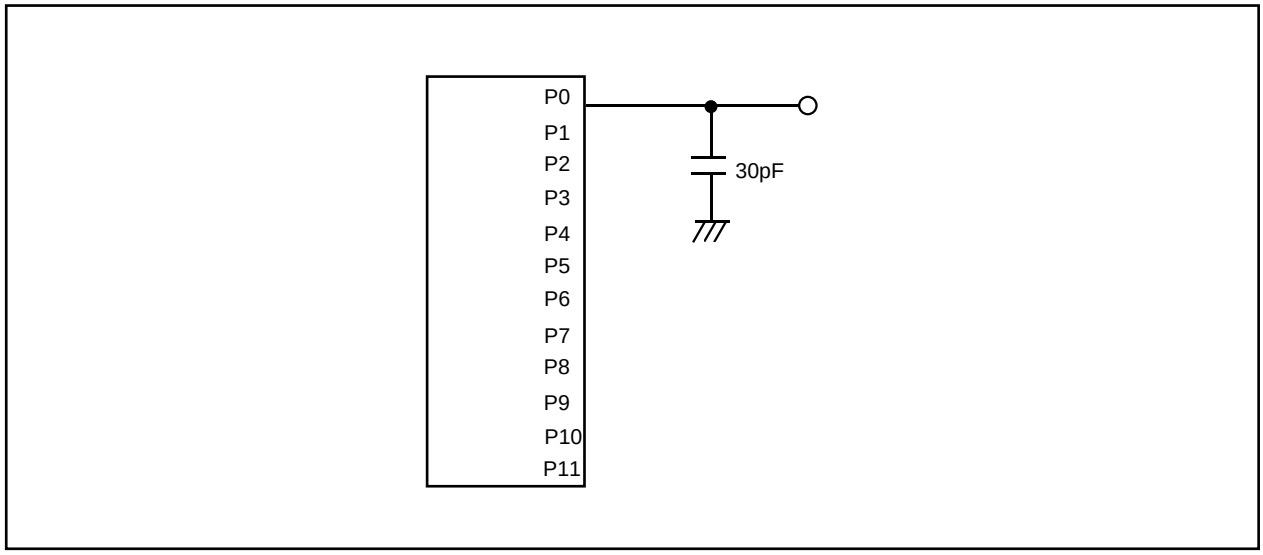



**Figure 4.1 Port P0 to P11 measurement circuit**

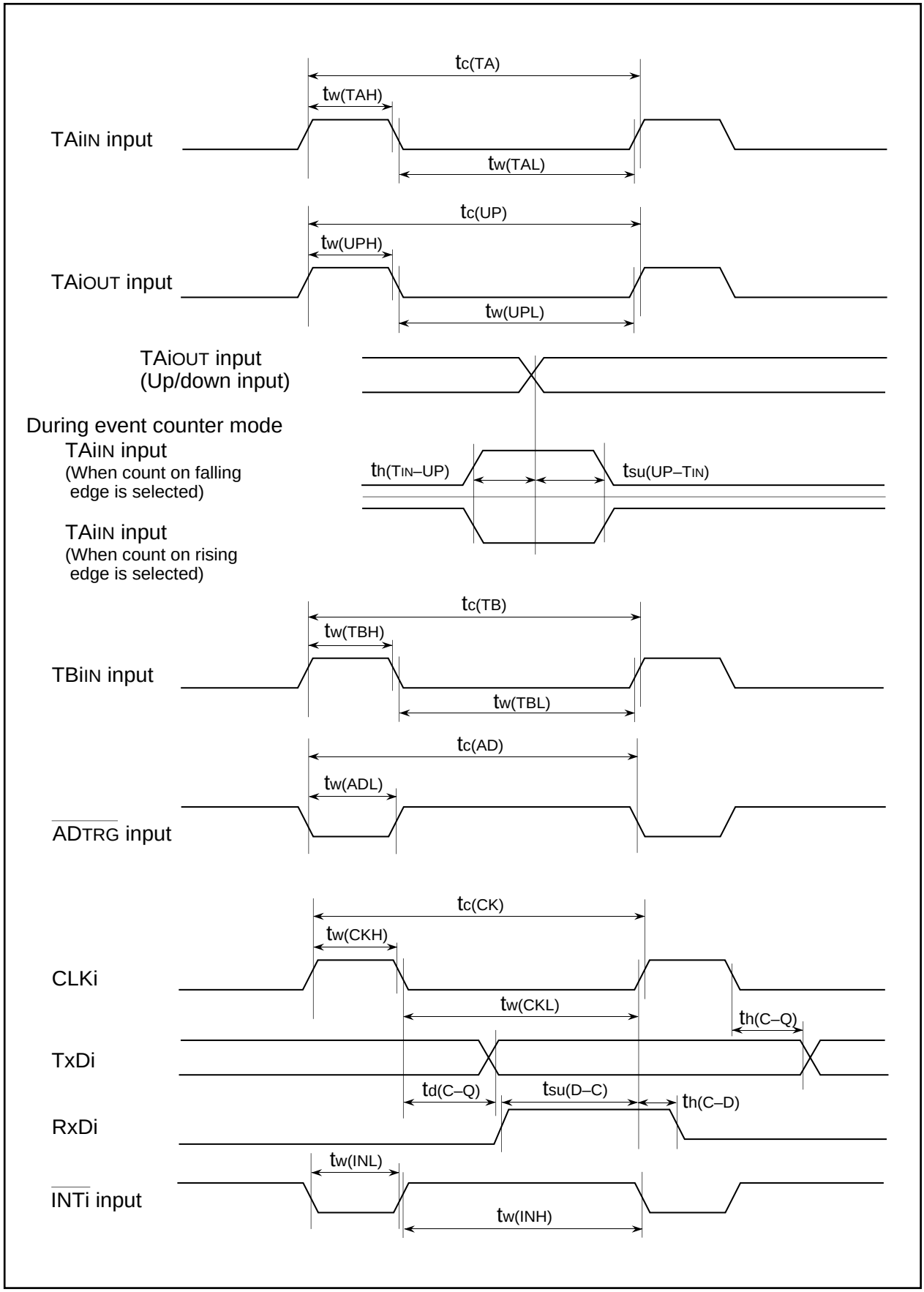
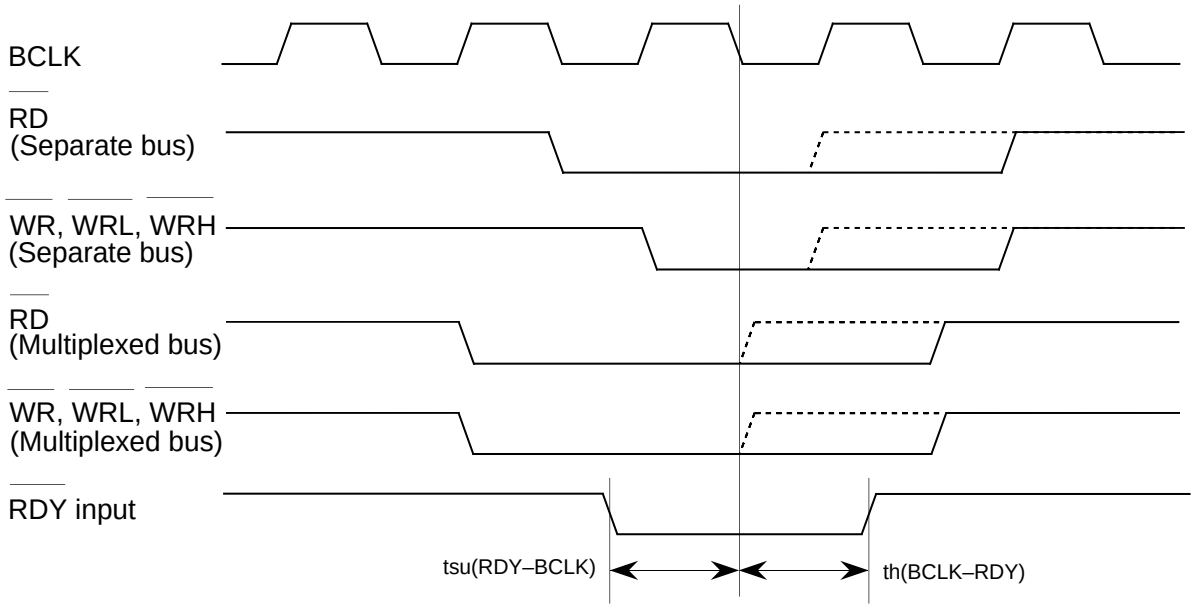
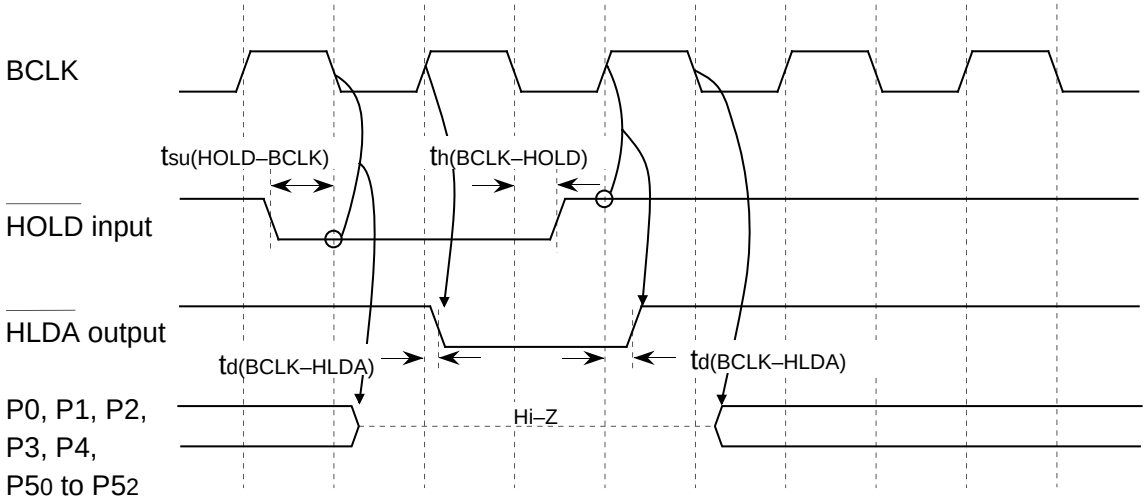



**Figure 4.2 Timing diagram (1)**

**Valid Only With Wait**




**Valid With Or Without Wait**



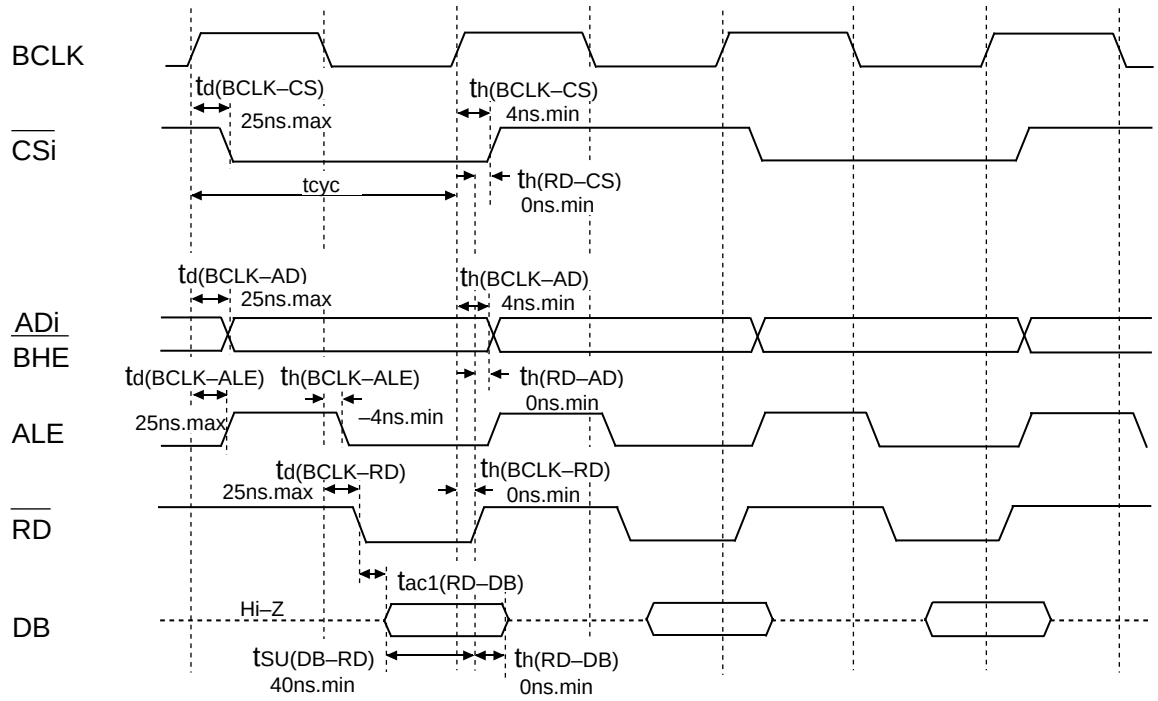

Note: The above pins are set to high-impedance regardless of the input level of the BYTE pin and bit (PM06) of processor mode register 0 selects the function of ports P40 to P43.

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

**Figure 4.3 Timing diagram (2)**

**With No Wait**

**Read timing**




**Write timing**

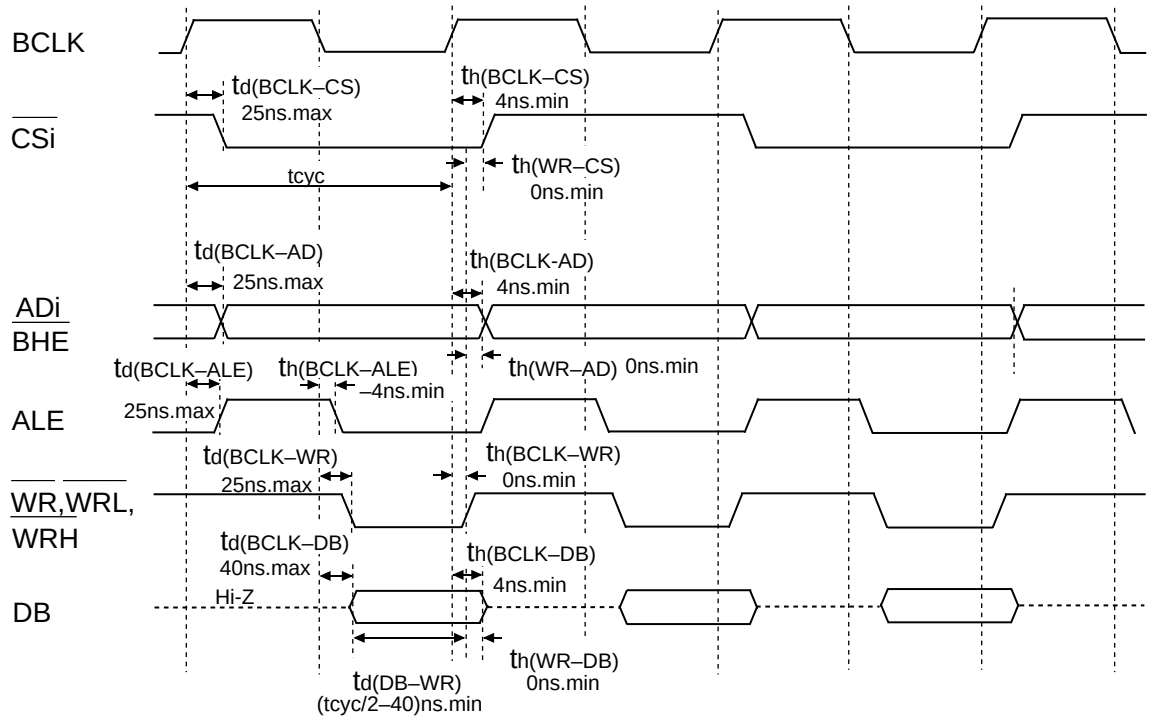
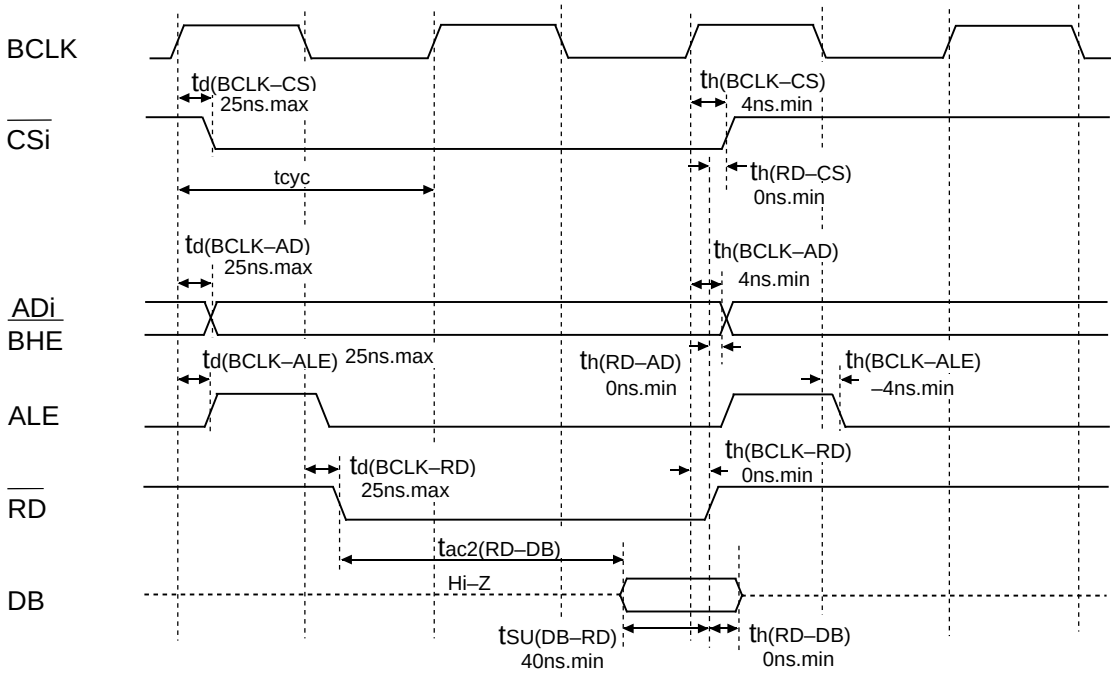


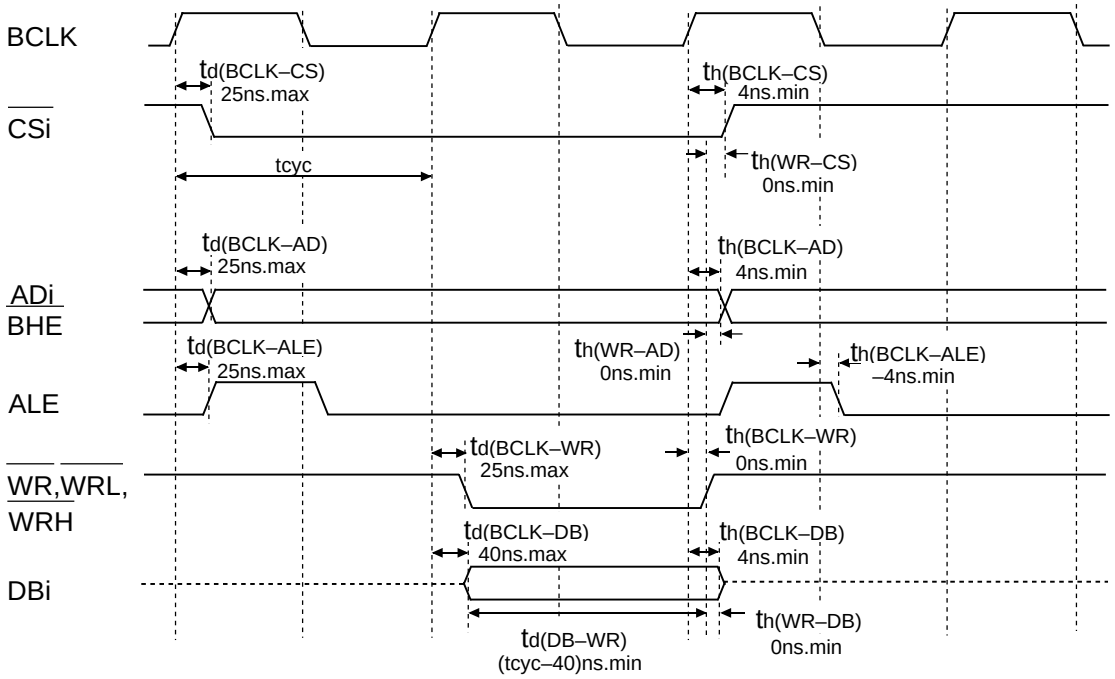

**Figure 4.4 Timing diagram (3)**

**When Accessing External Memory Area With Wait**

**Read timing**




**Write timing**

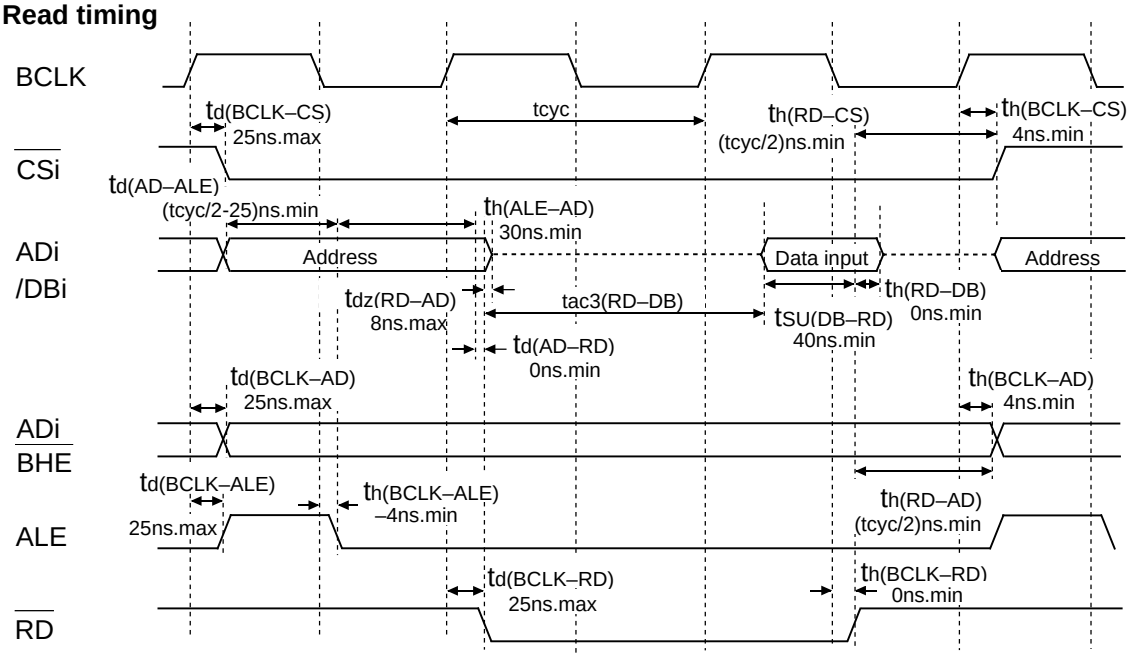



Measuring conditions :
- $V_{CC}$=5V
- Input timing voltage : Determined with: $V_{IL}$=0.8V, $V_{IH}$=2.5V
- Output timing voltage : Determined with: $V_{OL}$=0.8V, $V_{OH}$=2.0V

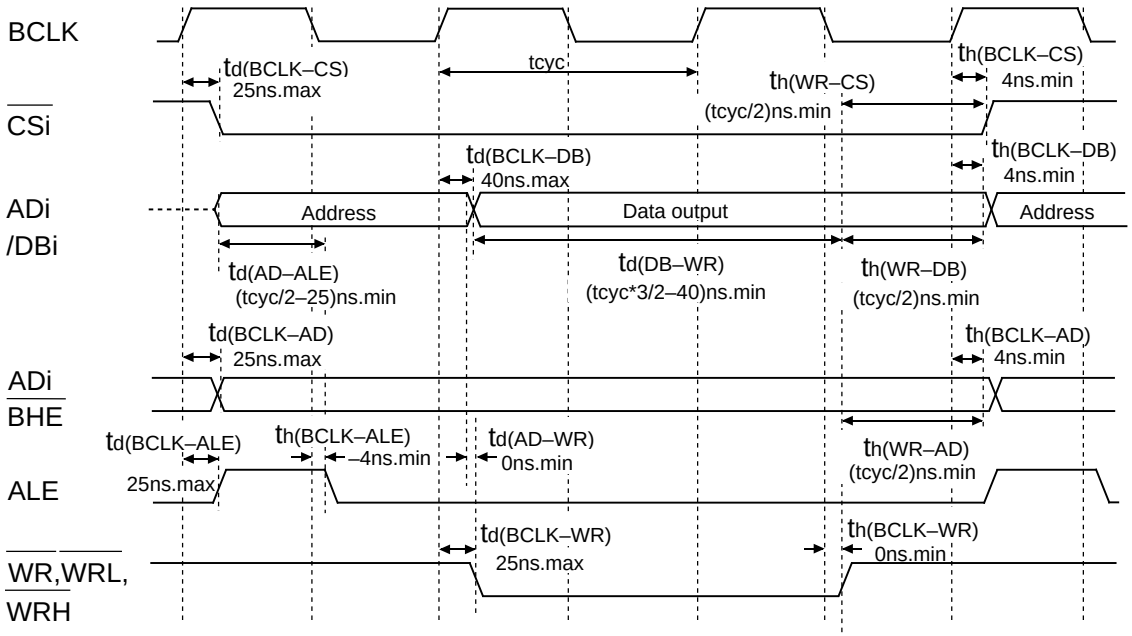
**Figure 4.5 Timing diagram (4)**

**When Accessing External Memory Area With Wait, And Select Multiplexed bus**

**Read timing**



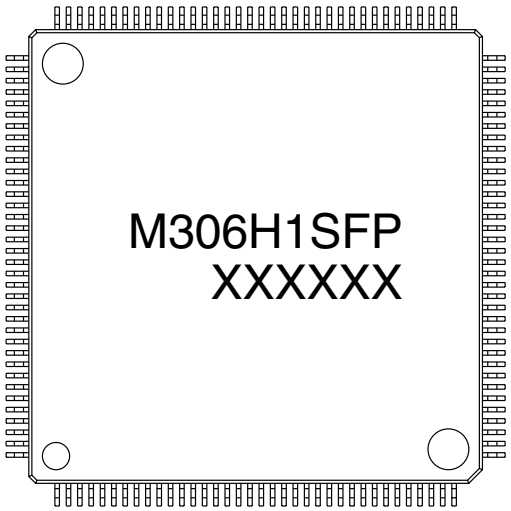

**Write timing**




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

**Figure 4.6 Timing diagram (5)**

## 5. Marking Figure



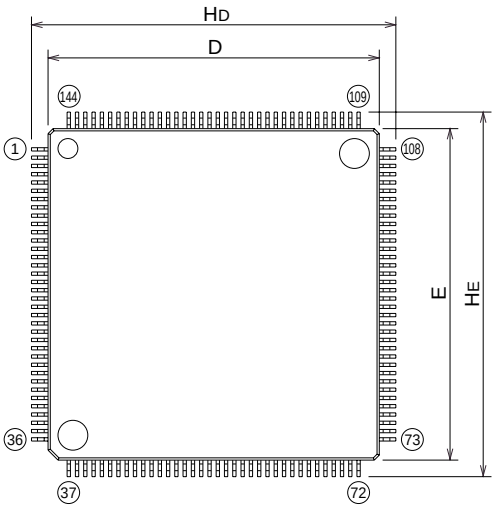

XXXXXX: Mitsubishi lot number

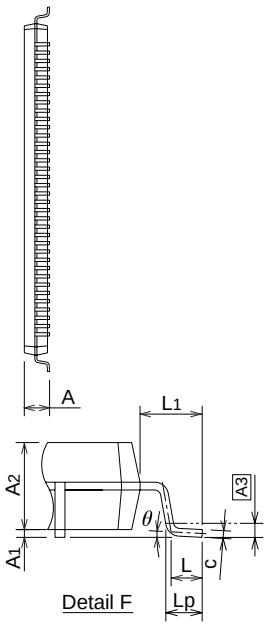
## 6. Package Outline

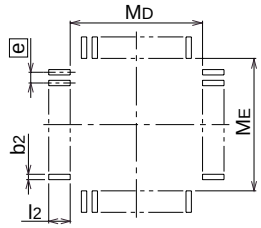
### 144P6Q-A

**Plastic 144pin 20×20mm body LQFP**

| EIAJ Package Code | JEDEC Code | Weight(g) | Lead Material |
|---|---|---|---|
| LQFP144-P-2020-0.50 | – | 1.23 | Cu Alloy |




Recommended Mount Pad

| Symbol | Dimension in Millimeters | | |
|---|---|---|---|
| | Min | Nom | Max |
| A | – | – | 1.7 |
| A1 | 0.05 | 0.125 | 0.2 |
| A2 | – | 1.4 | – |
| b | 0.17 | 0.22 | 0.27 |
| c | 0.105 | 0.125 | 0.175 |
| D | 19.9 | 20.0 | 20.1 |
| E | 19.9 | 20.0 | 20.1 |
| e | – | 0.5 | – |
| HD | 21.8 | 22.0 | 22.2 |
| HE | 21.8 | 22.0 | 22.2 |
| L | 0.35 | 0.5 | 0.65 |
| L1 | – | 1.0 | – |
| Lp | 0.45 | 0.6 | 0.75 |
| A3 | – | 0.25 | – |
| x | – | – | 0.08 |
| y | – | – | 0.1 |
| θ | 0° | – | 8° |
| b2 | – | 0.225 | – |
| l2 | 0.95 | – | – |
| MD | – | 20.4 | – |
| ME | – | 20.4 | – |

# M306H1SFP

SINGLE-CHIP 16-BIT CMOS MICROCOMPUTER
with DATA SLICER and ON-SCREEN DISPLAY CONTROLLER

Renesas Technology Corp.
Nippon Bldg.,6-2,Otemachi 2-chome,Chiyoda-ku,Tokyo,100-0004 Japan



RENESAS
Renesas Technology Corp.

| REVISION HISTORY | M306H1SFP (Rev.1.1) DATA SHEET |
|---|---|

| Rev. No. | Revision Description | Rev. date |
|---|---|---|
| 1.0 | PDF First Edition | 0006 |
| 1.1 | ・Expansion register construction corrected<br>(28) Address $1B_{16}$ ( = DA5 to 0) (page 195)<br>(29) Address $1C_{16}$ ( = DA5 to 0) (page 195)<br>(34) Address $21_{16}$ ( = DA5 to 0) (page 197)<br>(35) Address $22_{16}$ ( = DA5 to 0) (page 197)<br>・The change of the page layout<br>Usage precaution (page 219 and 220) | 0010 |
| | | |