# M32C/83 GROUP

# Datasheet

# Preliminarily

# Rev.B3

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

Mitsubishi Electric proprietary information

MITSUBISHI
ELECTRIC

# About M32C/83 Group

The M32C/83 group of single-chip microcomputers are built using a high-performance silicon gate CMOS process uses a M32C/80 Series CPU core and are packaged in a 144-pin and 100-pin plastic molded QFP. These single-chip microcomputers operate using sophisticated instructions featuring a high level of instruction efficiency. With 16M bytes of address space, they are capable of executing instructions at high speed. They also feature a built-in multiplier and DMAC, making them ideal for controlling office, communications, industrial equipment, and other high-speed processing applications.

## Applications

Audio, cameras, office equipment, communications equipment, portable equipment, etc.

## Index

About M32C/83 Group ..... 1
Central Processing Unit (CPU) ..... 20
Reset ..... 24
SFR ..... 37
Software Reset ..... 48
Processor Mode ..... 48
Bus Settings ..... 52
Bus Control ..... 55
System Clock ..... 65
Power Saving ..... 76
Protection ..... 81
Interrupt Outline ..... 83
$\overline{\text{INT}}$ Interrupts ..... 98
$\overline{\text{NMI}}$ Interrupt ..... 99
Key Input Interrupt ..... 99
Address Match Interrupt ..... 100
Intelligent I/O and CAN Interrupt ..... 101
Precautions for Interrupts ..... 104
Watchdog Timer ..... 106
DMAC ..... 109
DMAC II ..... 121
Timer ..... 129
Timer A ..... 131
Timer B ..... 147
Three-phase motor control timers' functions ..... 155
Serial I/O ..... 168
CAN Module ..... 198
Intelligent I/O ..... 235
A-D Converter ..... 281
D-A Converter ..... 296
CRC Calculation Circuit ..... 298
X-Y Converter ..... 300
DRAM Controller ..... 303
Programmable I/O Ports ..... 310
VDC ..... 334
Usage Precaution ..... 335
Electrical characteristics ..... 344
Electrical characteristics (Vcc = 3V) ..... 364
Outline Performance ..... 381
Flash Memory ..... 383
CPU Rewrite Mode ..... 384
Outline Performance of CPU Rewrite Mode ..... 384
Parallel I/O Mode ..... 399
Standard serial I/O mode ..... 400

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

## Performance Outline

Table 1.1.1 and 1.1.2 are performance outline of M32C/83 group.

**Table 1.1.1. Performance outline of M32C/83 group (144-pin version) (1/2)**

| Item | | | Performance |
|---|---|---|---|
| CPU | Number of basic instructions | | 108 instructions |
| | Shortest instruction execution time | | 33 ns(f(XIN)=30MHz) |
| | Operation mode | | Single-chip, memory expansion and microprocessor modes |
| | Memory space | | 16 M bytes |
| | Memory capacity | | See ROM/RAM expansion figure. |
| Peripheral function | | | |
| | I/O port | | 123 pins (P0 to P15 except P85) |
| | Input port | | 1 pin (P85) |
| | Multifunction timer | Output | 16 bits x 5 (TA0, TA1, TA2, TA3, TA4) |
| | | Input | 16 bits x 6 (TB0, TB1, TB2, TB3, TB4, TB5) |
| | Intelligent I/O<br>Time measurement<br>Waveform generation<br>Bit-modulation PWM<br>Real time port<br>Communication function | | 4 groups<br>8 channels (group 0) + 4 channels (group 1)<br>4 channels (group 0) + 8 channels X 3 (group 1, 2 and 3)<br>8 channels X 2 (group 2 and 3)<br>8 channels X 2 (group 2 and 3)<br>• Clock synchronous serial I/O, UART (group 0 and 1)<br>• HDLC data process (group 0 and 1)<br>• Clock synchronous variable length serial I/O (group 2)<br>• IE bus (Note 1) (group 2) |
| | Serial I/O | | 5 channels (UART0 to UART4)<br>IE Bus (Note 1, 3), I2C Bus (Note 2, 3) |
| | CAN module | | 1 channel, 2.0B specification |
| | A-D converter | | 10-bit A-D x 2 circuits, standard 18 inputs, max 34 inputs |
| | D-A converter | | 8-bit D-A x 2 circuits |
| | DMAC | | 4 channels |
| | DMAC II | | Start by all variable vector interrupt factor<br>Immediate transfer, operation transfer and chain transfer function |
| | DRAM controller | | CAS before RAS refresh, self-refresh, EDO, FP |
| | CRC calculation circuit | | CRC-CCITT |
| | X-Y converter | | 16 bits X 16 bits |
| | Watchdog timer | | 15 bits x 1 (with prescaler) |
| | Interrupt | | 42 internal and 8 external sources, 5 software sources, interrupt priority level 7 levels |
| | Clock generating circuit | | 3 built-in clock generation circuits<br>• Main/sub-clock generating circuit :built-in feedback resistance, and external ceramic or quartz oscillator<br>• Ring oscillator for detecting main clock oscillation stop |
| Electric characteristics | | | |
| | Supply voltage | | 4.2 to 5.5V (f(XIN)=30MHz without wait), 3.0 to 3.6V (f(XIN)=20MHz without wait) |
| | Power consumption | | 26mA (f(XIN)=20MHz without software wait,Vcc=3V)<br>38mA (f(XIN)=30MHz without software wait,Vcc=5V) |
| | I/O characteristics | | I/O withstand voltage :5V<br>I/O current :5mA |
| Operating ambient temperature | | | –40 to 85°C |
| Device configuration | | | CMOS high performance silicon gate |
| Package | | | 144-pin plastic mold QFP |

Note 1 :IE Bus is a trademark of NEC corporation.

Note 2 :I2C Bus is a registered trademark of Philips.

Note 3 :This function is executed by using software and hardware.

**Table 1.1.2. Performance outline of M32C/83 group (100-pin version) (1/2)**

| Item | | | Performance |
|---|---|---|---|
| CPU | Number of basic instructions | | 108 instructions |
| | Shortest instruction execution time | | 33 ns (f(XIN)=30MHz) |
| | Operation mode | | Single-chip, memory expansion and microprocessor modes |
| | Memory space | | 16 M bytes |
| | Memory capacity | | See ROM/RAM expansion figure. |
| Peripheral function | | | |
| | I/O port | | 87 pins (P0 to P10 except P85) |
| | Input port | | 1 pin (P85) |
| | Multifunction timer | Output | 16 bits x 5 (TA0, TA1, TA2, TA3, TA4) |
| | | Input | 16 bits x 6 (TB0, TB1, TB2, TB3, TB4, TB5) |
| | Intelligent I/O<br>Time measurement<br>Waveform generation<br>Bit-modulation PWM<br>Real time port<br>Communication function | | 4 groups<br>3 channels (group 0) + 2 channels (group 1)<br>2 channels X 2 (group 0 and 3) + 3 channels X 2 (group 1 and 2)<br>3 channels (group 2) + 2 channels (group 3)<br>3 channels (group 2) + 2 channels (group 3)<br>• Clock synchronous serial I/O, UART (group 0 and 1)<br>• HDLC data process (group 0 and 1)<br>• Clock synchronous variable length serial I/O (group 2)<br>• IE bus (Note 1) (group 2) |
| | Serial I/O | | 5 channels (UART0 to UART4)<br>IE Bus (Note 1, 3), I2C Bus (Note 2, 3) |
| | CAN module | | 1 channel, 2.0B specification |
| | A-D converter | | 10 bits A-Dx 2 circuits, standard 10 inputs, max 26 inputs |
| | D-A converter | | 8 bits D-A x 2 circuits |
| | DMAC | | 4 channels |
| | DMAC II | | Start by all variable vector interrupt factor<br>Immediate transfer, operation function and chain transfer function |
| | DRAM controller | | CAS before RAS refresh, self-refresh, EDO, FP |
| | CRC calculation circuit | | CRC-CCITT |
| | X-Y converter | | 16 bits X 16 bits |
| | Watchdog timer | | 15 bits x 1 (with prescaler) |
| | Interrupt | | 42 internal and 8 external sources, 5 software sources, interrupt priority level 7 levels |
| | Clock generating circuit | | 3 built-in clock generation circuits<br>• Main/sub-clock generating circuit :built-in feedback resistance, and external ceramic or quartz oscillator<br>• Ring oscillator for detecting main clock oscillation stop |
| Electric characteristics | | | |
| | Supply voltage | | 4.2 to 5.5V (f(XIN)=30MHz without wait), 3.0 to 3.6V (f(XIN)=20MHz without wait) |
| | Power consumption | | 26mA (f(XIN)=20MHz without software wait,Vcc=3V)<br>38mA (f(XIN)=30MHz without software wait,Vcc=5V) |
| | I/O characteristics | | I/O withstand voltage :5V<br>I/O current :5mA |
| Operating ambient temperature | | | –40 to 85°C |
| Device configuration | | | CMOS high performance silicon gate |
| Package | | | 100-pin plastic mold QFP |

Note 1 :IE Bus is a trademark of NEC corporation.

Note 2 :I2C Bus is a registered trademark of Philips.

Note 3 :This function is executed by using software and hardware.

Mitsubishi plans to release the following products in the M32C/83 group:

(1) Support for mask ROM version and flash memory version
(2) ROM capacity
(3) Package
100P6S-A : Plastic molded QFP (mask ROM version and flash memory version)
100P6Q-A : Plastic molded QFP (mask ROM version and flash memory version)
144P6Q-A : Plastic molded QFP (mask ROM version and flash memory version)

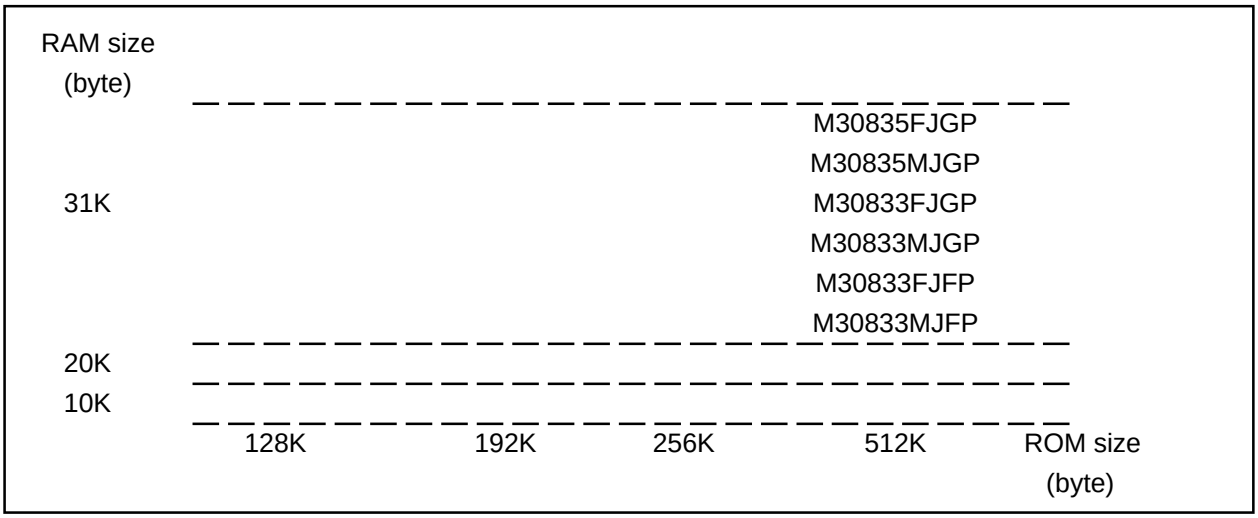



**Figure 1.1.1. ROM expansion**

The M32C/83 group products currently supported are listed in Table 1.1.3.

**Table 1.1.3. M32C/83 group** **As of April, 2002**

| Type No | ROM capacity | RAM capacity | Package type | Remarks |
|---|---|---|---|---|
| M30835MJGP *** | 512K | 31K | 144P6Q-A | Mask ROM version |
| M30833MJGP *** | | | 100P6Q-A | |
| M30833MJFP *** | | | 100P6S-A | |
| M30835FJGP ** | | | 144P6Q-A | Flash memory version |
| M30833FJGP ** | | | 100P6Q-A | |
| M30833FJFP ** | | | 100P6S-A | |

** :Under development
*** :Under planning

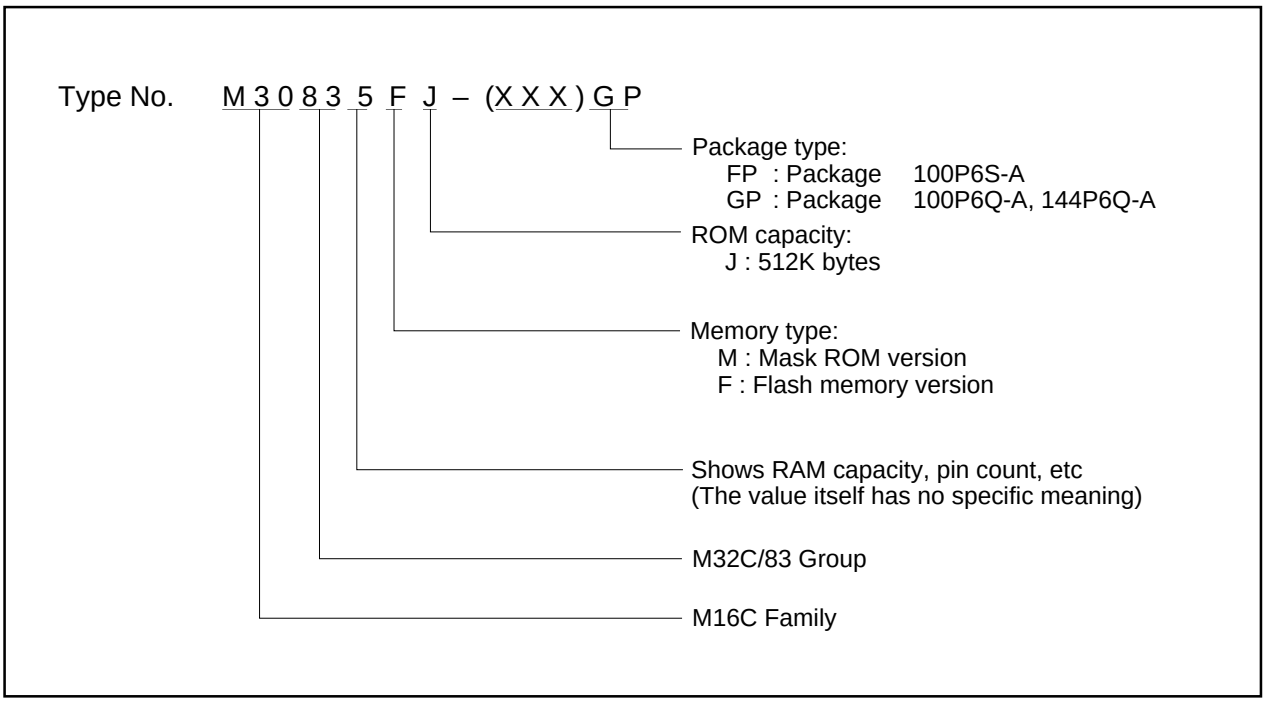
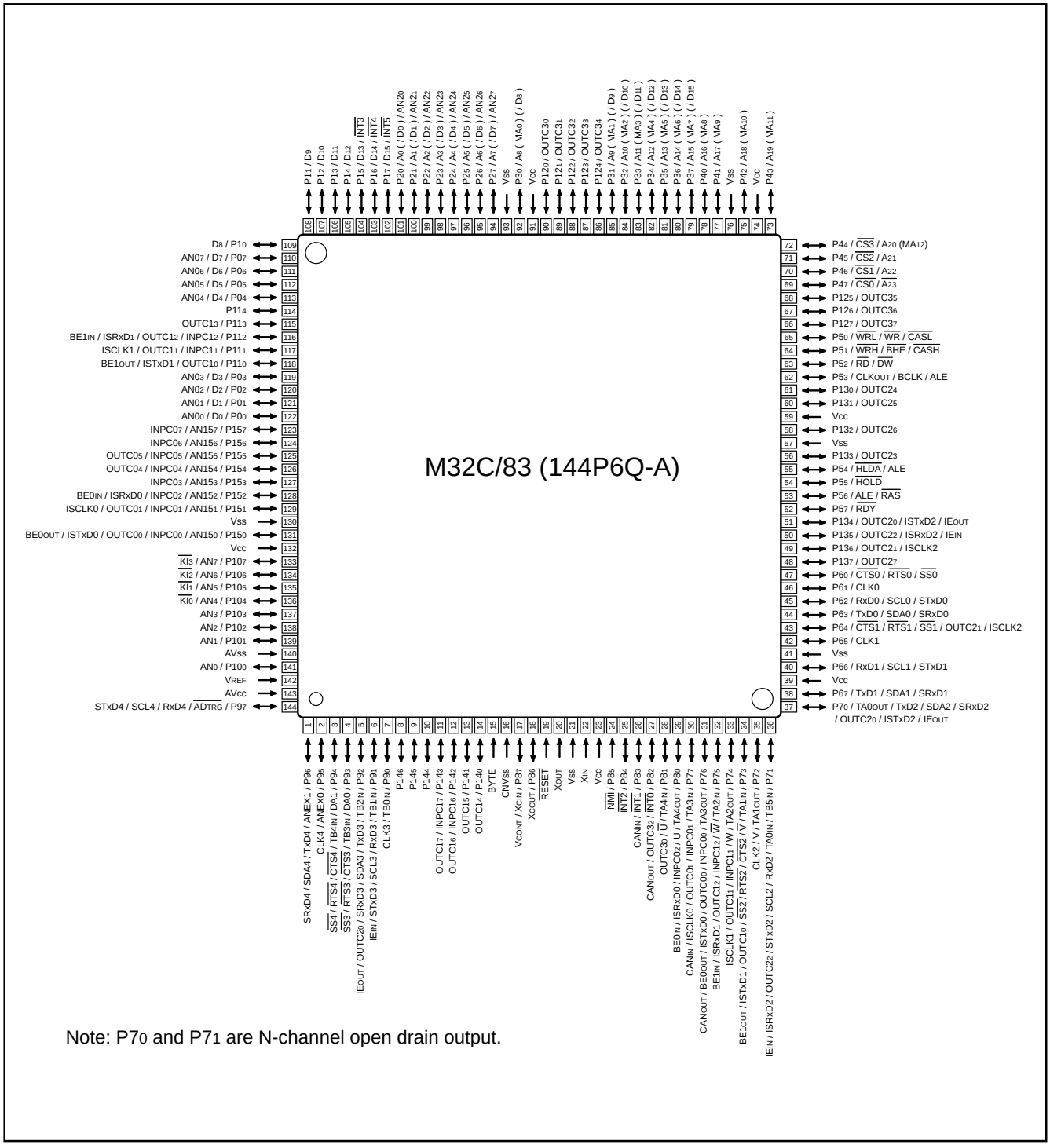



**Figure 1.1.2. Type No., memory size, and package**

## Pin Configuration and Pin Description

Figure 1.1.3 to 1.1.5 show the pin configurations (top view), Table 1.1.3 list pin names, and Table 1.1.4 list pin description.




Note: P70 and P71 are N-channel open drain output.

**Figure 1.1.3. 144-pin version pin configuration (top view)**

**Table 1.1.4. 144-pin version pin description (1/3)**

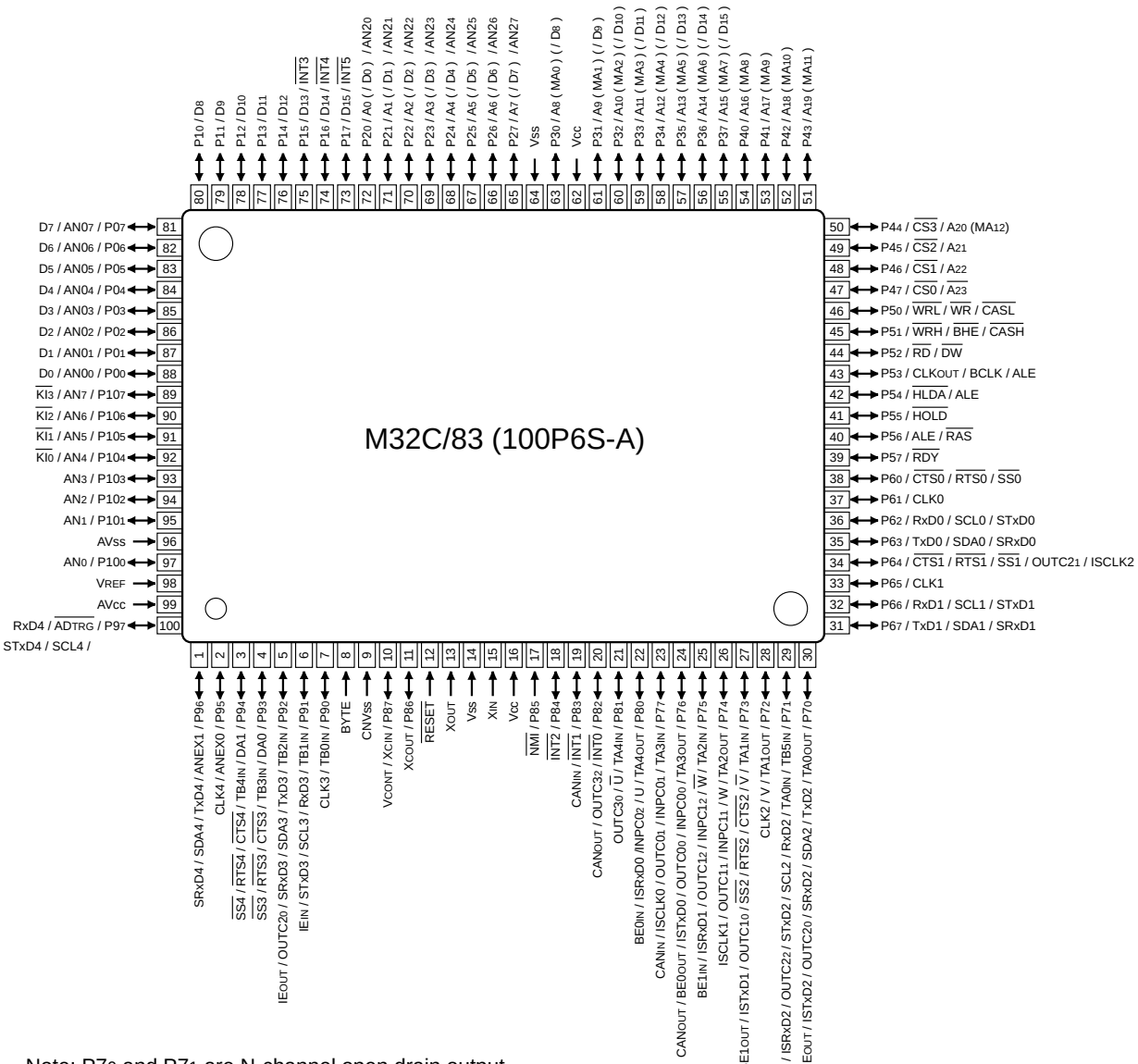
| Pin No | Control | Port | Interrupt | Timer | UART/CAN | Intelligent I/O | Analog | Bus control |
|---|---|---|---|---|---|---|---|---|
| 1 | | P9$_6$ | | | TxD4/SDA4/SRxD4 | | ANEX1 | |
| 2 | | P9$_5$ | | | CLK4 | | ANEX0 | |
| 3 | | P9$_4$ | | TB4$_{IN}$ | $\overline{CTS4}$/$\overline{RTS4}$/$\overline{SS4}$ | | DA1 | |
| 4 | | P9$_3$ | | TB3$_{IN}$ | $\overline{CTS3}$/$\overline{RTS3}$/$\overline{SS3}$ | | DA0 | |
| 5 | | P9$_2$ | | TB2$_{IN}$ | TxD3/SDA3/SRxD3 | OUTC2$_0$/IE$_{OUT}$ | | |
| 6 | | P9$_1$ | | TB1$_{IN}$ | RxD3/SCL3/STxD3 | IE$_{IN}$ | | |
| 7 | | P9$_0$ | | TB0$_{IN}$ | CLK3 | | | |
| 8 | | P14$_6$ | | | | | | |
| 9 | | P14$_5$ | | | | | | |
| 10 | | P14$_4$ | | | | | | |
| 11 | | P14$_3$ | | | | INPC1$_7$/OUTC1$_7$ | | |
| 12 | | P14$_2$ | | | | INPC1$_6$/OUTC1$_6$ | | |
| 13 | | P14$_1$ | | | | OUTC1$_5$ | | |
| 14 | | P14$_0$ | | | | OUTC1$_4$ | | |
| 15 | BYTE | | | | | | | |
| 16 | CNV$_{SS}$ | | | | | | | |
| 17 | X$_{CIN}$/V$_{CONT}$ | P8$_7$ | | | | | | |
| 18 | X$_{COUT}$ | P8$_6$ | | | | | | |
| 19 | $\overline{RESET}$ | | | | | | | |
| 20 | X$_{OUT}$ | | | | | | | |
| 21 | V$_{SS}$ | | | | | | | |
| 22 | X$_{IN}$ | | | | | | | |
| 23 | V$_{CC}$ | | | | | | | |
| 24 | | P8$_5$ | $\overline{NMI}$ | | | | | |
| 25 | | P8$_4$ | $\overline{INT2}$ | | | | | |
| 26 | | P8$_3$ | $\overline{INT1}$ | | CAN$_{IN}$ | | | |
| 27 | | P8$_2$ | $\overline{INT0}$ | | CAN$_{OUT}$ | OUTC3$_2$ | | |
| 28 | | P8$_1$ | | TA4$_{IN}$/$\overline{U}$ | | OUTC3$_0$ | | |
| 29 | | P8$_0$ | | TA4$_{OUT}$/U | | INPC0$_2$/ISRxD0/BE0$_{IN}$ | | |
| 30 | | P7$_7$ | | TA3$_{IN}$ | CAN$_{IN}$ | INPC0$_1$/OUTC0$_1$/ISCLK0 | | |
| 31 | | P7$_6$ | | TA3$_{OUT}$ | CAN$_{OUT}$ | INPC0$_0$/OUTC0$_0$/ISTxD0/BE0$_{OUT}$ | | |
| 32 | | P7$_5$ | | TA2$_{IN}$/$\overline{W}$ | | INPC1$_2$/OUTC1$_2$/ISRxD1/BE1$_{IN}$ | | |
| 33 | | P7$_4$ | | TA2$_{OUT}$/W | | INPC1$_1$/OUTC1$_1$/ISCLK1 | | |
| 34 | | P7$_3$ | | TA1$_{IN}$/$\overline{V}$ | $\overline{CTS2}$/$\overline{RTS2}$/$\overline{SS2}$ | OUTC1$_0$/ISTxD1/BE1$_{OUT}$ | | |
| 35 | | P7$_2$ | | TA1$_{OUT}$/V | CLK2 | | | |
| 36 | | P7$_1$ | | TB5$_{IN}$/TA0$_{IN}$ | RxD2/SCL2/STxD2 | OUTC2$_2$/ISRxD2/IE$_{IN}$ | | |
| 37 | | P7$_0$ | | TA0$_{OUT}$ | TxD2/SDA2/SRxD2 | OUTC2$_0$/ISTxD2/IE$_{OUT}$ | | |
| 38 | | P6$_7$ | | | TxD1/SDA1/SRxD1 | | | |
| 39 | V$_{CC}$ | | | | | | | |
| 40 | | P6$_6$ | | | RxD1/SCL1/STxD1 | | | |
| 41 | V$_{SS}$ | | | | | | | |
| 42 | | P6$_5$ | | | CLK1 | | | |
| 43 | | P6$_4$ | | | $\overline{CTS1}$/$\overline{RTS1}$/$\overline{SS1}$ | OUTC2$_1$/ISCLK2 | | |
| 44 | | P6$_3$ | | | TxD0/SDA0/SRxD0 | | | |
| 45 | | P6$_2$ | | | RxD0/SCL0/STxD0 | | | |
| 46 | | P6$_1$ | | | CLK0 | | | |
| 47 | | P6$_0$ | | | $\overline{CTS0}$/$\overline{RTS0}$/$\overline{SS0}$ | | | |
| 48 | | P13$_7$ | | | | OUTC2$_7$ | | |

**Table 1.1.5. 144-pin version pin description (2/3)**

| Pin No | Control | Port | Interrupt | Timer | UART/CAN | Intelligent I/O | Analog | Bus control |
|---|---|---|---|---|---|---|---|---|
| 49 | | P13$_6$ | | | | OUTC2$_1$/ISCLK2 | | |
| 50 | | P13$_5$ | | | | OUTC2$_2$/ISRxD2/IE$_{IN}$ | | |
| 51 | | P13$_4$ | | | | OUTC2$_0$/ISTxD2/IE$_{OUT}$ | | |
| 52 | | P5$_7$ | | | | | | $\overline{RDY}$ |
| 53 | | P5$_6$ | | | | | | ALE/$\overline{RAS}$ |
| 54 | | P5$_5$ | | | | | | $\overline{HOLD}$ |
| 55 | | P5$_4$ | | | | | | $\overline{HLDA}$/ALE |
| 56 | | P13$_3$ | | | | OUTC2$_3$ | | |
| 57 | V$_{SS}$ | | | | | | | |
| 58 | | P13$_2$ | | | | OUTC2$_6$ | | |
| 59 | V$_{CC}$ | | | | | | | |
| 60 | | P13$_1$ | | | | OUTC2$_5$ | | |
| 61 | | P13$_0$ | | | | OUTC2$_4$ | | |
| 62 | | P5$_3$ | | | | | | CLK$_{OUT}$/BCLK/ALE |
| 63 | | P5$_2$ | | | | | | $\overline{RD}$/$\overline{DW}$ |
| 64 | | P5$_1$ | | | | | | $\overline{WRH}$/$\overline{BHE}$/$\overline{CASH}$ |
| 65 | | P5$_0$ | | | | | | $\overline{WRL}$/$\overline{WR}$/$\overline{CASL}$ |
| 66 | | P12$_7$ | | | | OUTC3$_7$ | | |
| 67 | | P12$_6$ | | | | OUTC3$_6$ | | |
| 68 | | P12$_5$ | | | | OUTC3$_5$ | | |
| 69 | | P4$_7$ | | | | | | $\overline{CS0}$/$\overline{A_{23}}$ |
| 70 | | P4$_6$ | | | | | | $\overline{CS1}$/A$_{22}$ |
| 71 | | P4$_5$ | | | | | | $\overline{CS2}$/A$_{21}$ |
| 72 | | P4$_4$ | | | | | | $\overline{CS3}$/A$_{20}$(MA$_{12}$) |
| 73 | | P4$_3$ | | | | | | A$_{19}$(MA$_{11}$) |
| 74 | V$_{CC}$ | | | | | | | |
| 75 | | P4$_2$ | | | | | | A$_{18}$(MA$_{10}$) |
| 76 | V$_{SS}$ | | | | | | | |
| 77 | | P4$_1$ | | | | | | A$_{17}$(MA$_9$) |
| 78 | | P4$_0$ | | | | | | A$_{16}$(MA$_8$) |
| 79 | | P3$_7$ | | | | | | A$_{15}$(MA$_7$)(/D$_{15}$) |
| 80 | | P3$_6$ | | | | | | A$_{14}$(MA$_6$)(/D$_{14}$) |
| 81 | | P3$_5$ | | | | | | A$_{13}$(MA$_5$)(/D$_{13}$) |
| 82 | | P3$_4$ | | | | | | A$_{12}$(MA$_4$)(/D$_{12}$) |
| 83 | | P3$_3$ | | | | | | A$_{11}$(MA$_3$)(/D$_{11}$) |
| 84 | | P3$_2$ | | | | | | A$_{10}$(MA$_2$)(/D$_{10}$) |
| 85 | | P3$_1$ | | | | | | A$_9$(MA$_1$)(/D$_9$) |
| 86 | | P12$_4$ | | | | OUTC3$_4$ | | |
| 87 | | P12$_3$ | | | | OUTC3$_3$ | | |
| 88 | | P12$_2$ | | | | OUTC3$_2$ | | |
| 89 | | P12$_1$ | | | | OUTC3$_1$ | | |
| 90 | | P12$_0$ | | | | OUTC3$_0$ | | |
| 91 | V$_{CC}$ | | | | | | | |
| 92 | | P3$_0$ | | | | | | A$_8$(MA$_0$)(/D$_8$) |
| 93 | V$_{SS}$ | | | | | | | |
| 94 | | P2$_7$ | | | | | AN3$_7$ | A$_7$(/D$_7$) |
| 95 | | P2$_6$ | | | | | AN3$_6$ | A$_6$(/D$_6$) |
| 96 | | P2$_5$ | | | | | AN3$_5$ | A$_5$(/D$_5$) |

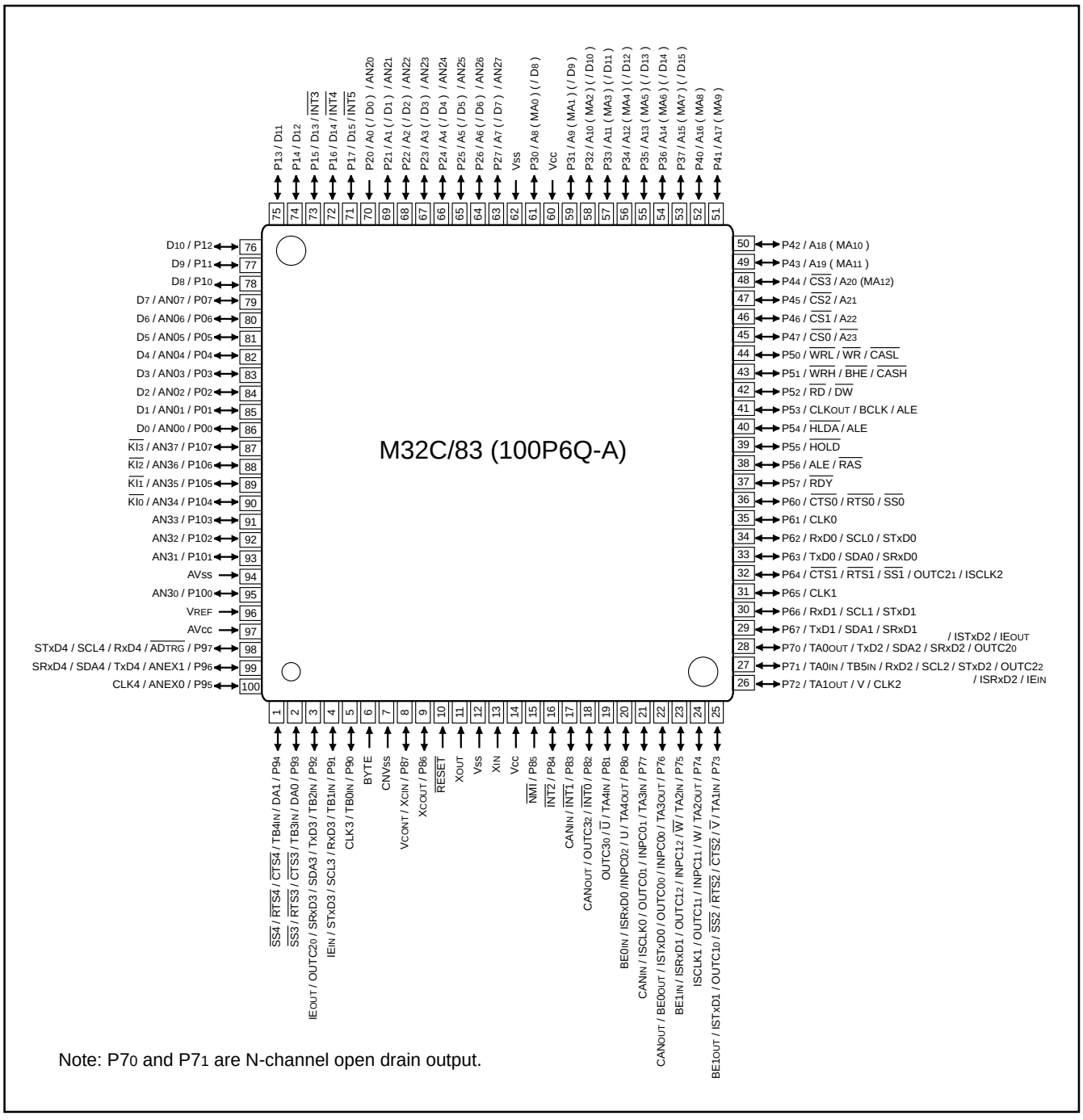
**Table 1.1.6. 144-pin version pin description (3/3)**

| Pin No | Control | Port | Interrupt | Timer | UART/CAN | Intelligent I/O | Analog | Bus control |
|---|---|---|---|---|---|---|---|---|
| 97 | | $P2_4$ | | | | | $AN2_4$ | $A_4$(/$D_4$) |
| 98 | | $P2_3$ | | | | | $AN2_3$ | $A_3$(/$D_3$) |
| 99 | | $P2_2$ | | | | | $AN2_2$ | $A_2$(/$D_2$) |
| 100 | | $P2_1$ | | | | | $AN2_1$ | $A_1$(/$D_1$) |
| 101 | | $P2_0$ | | | | | $AN2_0$ | $A_0$(/$D_0$) |
| 102 | | $P1_7$ | $\overline{INT5}$ | | | | | $D_{15}$ |
| 103 | | $P1_6$ | $\overline{INT4}$ | | | | | $D_{14}$ |
| 104 | | $P1_5$ | $\overline{INT3}$ | | | | | $D_{13}$ |
| 105 | | $P1_4$ | | | | | | $D_{12}$ |
| 106 | | $P1_3$ | | | | | | $D_{11}$ |
| 107 | | $P1_2$ | | | | | | $D_{10}$ |
| 108 | | $P1_1$ | | | | | | $D_9$ |
| 109 | | $P1_0$ | | | | | | $D_8$ |
| 110 | | $P0_7$ | | | | | $AN0_7$ | $D_7$ |
| 111 | | $P0_6$ | | | | | $AN0_6$ | $D_6$ |
| 112 | | $P0_5$ | | | | | $AN0_5$ | $D_5$ |
| 113 | | $P0_4$ | | | | | $AN0_4$ | $D_4$ |
| 114 | | $P11_4$ | | | | | | |
| 115 | | $P11_3$ | | | | $OUTC1_3$ | | |
| 116 | | $P11_2$ | | | | $INPC1_2$/$OUTC1_2$/ISRxD1/$BE1_{IN}$ | | |
| 117 | | $P11_1$ | | | | $INPC1_1$/$OUTC1_1$/ISCLK1 | | |
| 118 | | $P11_0$ | | | | $OUTC1_0$/ISTxD1/$BE1_{OUT}$ | | |
| 119 | | $P0_3$ | | | | | $AN0_3$ | $D_3$ |
| 120 | | $P0_2$ | | | | | $AN0_2$ | $D_2$ |
| 121 | | $P0_1$ | | | | | $AN0_1$ | $D_1$ |
| 122 | | $P0_0$ | | | | | $AN0_0$ | $D_0$ |
| 123 | | $P15_7$ | | | | $INPC0_7$ | $AN15_7$ | |
| 124 | | $P15_6$ | | | | $INPC0_6$ | $AN15_6$ | |
| 125 | | $P15_5$ | | | | $INPC0_5$/$OUTC0_5$ | $AN15_5$ | |
| 126 | | $P15_4$ | | | | $INPC0_4$/$OUTC0_4$ | $AN15_4$ | |
| 127 | | $P15_3$ | | | | $INPC0_3$ | $AN15_3$ | |
| 128 | | $P15_2$ | | | | $INPC0_2$/ISRxD0/$BE0_{IN}$ | $AN15_2$ | |
| 129 | | $P15_1$ | | | | $INPC0_1$/$OUTC0_1$/ISCLK0 | $AN15_1$ | |
| 130 | $V_{SS}$ | | | | | | | |
| 131 | | $P15_0$ | | | | $INPC0_0$/$OUTC0_0$/ISTxD0/$BE0_{OUT}$ | $AN15_0$ | |
| 132 | $V_{CC}$ | | | | | | | |
| 133 | | $P10_7$ | $\overline{KI_3}$ | | | | $AN_7$ | |
| 134 | | $P10_6$ | $\overline{KI_2}$ | | | | $AN_6$ | |
| 135 | | $P10_5$ | $\overline{KI_1}$ | | | | $AN_5$ | |
| 136 | | $P10_4$ | $\overline{KI_0}$ | | | | $AN_4$ | |
| 137 | | $P10_3$ | | | | | $AN_3$ | |
| 138 | | $P10_2$ | | | | | $AN_2$ | |
| 139 | | $P10_1$ | | | | | $AN_1$ | |
| 140 | $AV_{SS}$ | | | | | | | |
| 141 | | $P10_0$ | | | | | $AN_0$ | |
| 142 | $V_{REF}$ | | | | | | | |
| 143 | $AV_{CC}$ | | | | | | | |
| 144 | | $P9_7$ | | | RxD4/SCL4/STxD4 | | $\overline{ADTRG}$ | |




Note: P70 and P71 are N-channel open drain output.

**Figure 1.1.4. 100-pin version pin configuration (top view)**




Note: P70 and P71 are N-channel open drain output.

**Figure 1.1.5. 100-pin version pin configuration (top view)**

**Table 1.1.7. 100-pin version pin description (1/2)**

| Package Pin No | | Control | Port | Interrupt | Timer | UART/CAN | Intelligent I/O | Analog | Bus control |
|---|---|---|---|---|---|---|---|---|---|
| FP | GP | | | | | | | | |
| 1 | 99 | | P9$_6$ | | | TxD4/SDA4/SRxD4 | | ANEX1 | |
| 2 | 100 | | P9$_5$ | | TB4$_{IN}$ | CLK4 | | ANEX0 | |
| 3 | 1 | | P9$_4$ | | TB3$_{IN}$ | $\overline{CTS4}$/$\overline{RTS4}$/$\overline{SS4}$ | | DA1 | |
| 4 | 2 | | P9$_3$ | | TB2$_{IN}$ | $\overline{CTS3}$/$\overline{RTS3}$/$\overline{SS3}$ | | DA0 | |
| 5 | 3 | | P9$_2$ | | TB1$_{IN}$ | TxD3/SDA3/SRxD3 | OUTC2$_0$/IE$_{OUT}$ | | |
| 6 | 4 | | P9$_1$ | | TB0$_{IN}$ | RxD3/SCL3/STxD3 | IE$_{IN}$ | | |
| 7 | 5 | | P9$_0$ | | | CLK3 | | | |
| 8 | 6 | BYTE | | | | | | | |
| 9 | 7 | CNV$_{SS}$ | | | | | | | |
| 10 | 8 | X$_{CIN}$/V$_{CONT}$ | P8$_7$ | | | | | | |
| 11 | 9 | X$_{COUT}$ | P8$_6$ | | | | | | |
| 12 | 10 | $\overline{RESET}$ | | | | | | | |
| 13 | 11 | X$_{OUT}$ | | | | | | | |
| 14 | 12 | V$_{SS}$ | | | | | | | |
| 15 | 13 | X$_{IN}$ | | | | | | | |
| 16 | 14 | V$_{CC}$ | | | | | | | |
| 17 | 15 | | P8$_5$ | $\overline{NMI}$ | | | | | |
| 18 | 16 | | P8$_4$ | $\overline{INT2}$ | | | | | |
| 19 | 17 | | P8$_3$ | $\overline{INT1}$ | | CAN$_{IN}$ | | | |
| 20 | 18 | | P8$_2$ | $\overline{INT0}$ | | CAN$_{OUT}$ | OUTC3$_2$ | | |
| 21 | 19 | | P8$_1$ | | TA4$_{IN}$/$\overline{U}$ | | OUTC3$_0$ | | |
| 22 | 20 | | P8$_0$ | | TA4$_{OUT}$/$\overline{U}$ | | INPC0$_2$/ISRxD0/BE0$_{IN}$ | | |
| 23 | 21 | | P7$_7$ | | TA3$_{IN}$ | CAN$_{IN}$ | INPC0$_1$/OUTC0$_1$/ISCLK0 | | |
| 24 | 22 | | P7$_6$ | | TA3$_{OUT}$ | CAN$_{OUT}$ | INPC0$_0$/OUTC0$_0$/ISTxD0/BE0$_{OUT}$ | | |
| 25 | 23 | | P7$_5$ | | TA2$_{IN}$/W | | INPC1$_2$/OUTC1$_2$/ISRxD1/BE1$_{IN}$ | | |
| 26 | 24 | | P7$_4$ | | TA2$_{OUT}$/$\overline{W}$ | | INPC1$_1$/OUTC1$_1$/ISCLK1 | | |
| 27 | 25 | | P7$_3$ | | TA1$_{IN}$/V | $\overline{CTS2}$/$\overline{RTS2}$/$\overline{SS2}$ | OUTC1$_0$/ISTxD1/BE1$_{OUT}$ | | |
| 28 | 26 | | P7$_2$ | | TA1$_{OUT}$/$\overline{V}$ | CLK2 | | | |
| 29 | 27 | | P7$_1$ | | TB5$_{IN}$/TA0$_{IN}$ | RxD2/SCL2/STxD2 | OUTC2$_2$/ISRxD2/IE$_{IN}$ | | |
| 30 | 28 | | P7$_0$ | | TA0$_{OUT}$ | TxD2/SDA2/SRxD2 | OUTC2$_0$/ISTxD2/IE$_{OUT}$ | | |
| 31 | 29 | | P6$_7$ | | | TxD1/SDA1/SRxD1 | | | |
| 32 | 30 | | P6$_6$ | | | RxD1/SCL1/STxD1 | | | |
| 33 | 31 | | P6$_5$ | | | CLK1 | | | |
| 34 | 32 | | P6$_4$ | | | $\overline{CTS1}$/$\overline{RTS1}$/$\overline{SS1}$ | OUTC2$_1$/ISCLK2 | | |
| 35 | 33 | | P6$_3$ | | | TxD0/SDA0/SRxD0 | | | |
| 36 | 34 | | P6$_2$ | | | RxD0/SCL0/STxD0 | | | |
| 37 | 35 | | P6$_1$ | | | CLK0 | | | |
| 38 | 36 | | P6$_0$ | | | $\overline{CTS0}$/$\overline{RTS0}$/$\overline{SS0}$ | | | |
| 39 | 37 | | P5$_7$ | | | | | | $\overline{RDY}$ |
| 40 | 38 | | P5$_6$ | | | | | | ALE/$\overline{RAS}$ |
| 41 | 39 | | P5$_5$ | | | | | | $\overline{HOLD}$ |
| 42 | 40 | | P5$_4$ | | | | | | $\overline{HLDA}$/ALE |
| 43 | 41 | | P5$_3$ | | | | | | CLK$_{OUT}$/BCLK/ALE |
| 44 | 42 | | P5$_2$ | | | | | | $\overline{RD}$/$\overline{DW}$ |
| 45 | 43 | | P5$_1$ | | | | | | $\overline{WRH}$/$\overline{BHE}$/$\overline{CASH}$ |
| 46 | 44 | | P5$_0$ | | | | | | $\overline{WRL}$/$\overline{WR}$/$\overline{CASL}$ |
| 47 | 45 | | P4$_7$ | | | | | | $\overline{CS0}$/A$_{23}$ |
| 48 | 46 | | P4$_6$ | | | | | | $\overline{CS1}$/A$_{22}$ |
| 49 | 47 | | P4$_5$ | | | | | | $\overline{CS2}$/A$_{21}$ |
| 50 | 48 | | P4$_4$ | | | | | | $\overline{CS3}$/A$_{20}$(MA$_{12}$) |

**Table 1.1.8. 100-pin version pin description (2/2)**

| Package pin No | | Control | Port | Interrupt | Timer | UART/CAN | Intelligent I/O | Analog | Bus control |
|---|---|---|---|---|---|---|---|---|---|
| FP | GP | | | | | | | | |
| 51 | 49 | | P4$_3$ | | | | | | A$_{19}$(MA$_{11}$) |
| 52 | 50 | | P4$_2$ | | | | | | A$_{18}$(MA$_{10}$) |
| 53 | 51 | | P4$_1$ | | | | | | A$_{17}$(MA$_9$) |
| 54 | 52 | | P4$_0$ | | | | | | A$_{16}$(MA$_8$) |
| 55 | 53 | | P3$_7$ | | | | | | A$_{15}$(MA$_7$)(/D$_{15}$) |
| 56 | 54 | | P3$_6$ | | | | | | A$_{14}$(MA$_6$)(/D$_{14}$) |
| 57 | 55 | | P3$_5$ | | | | | | A$_{13}$(MA$_5$)(/D$_{13}$) |
| 58 | 56 | | P3$_4$ | | | | | | A$_{12}$(MA$_4$)(/D$_{12}$) |
| 59 | 57 | | P3$_3$ | | | | | | A$_{11}$(MA$_3$)(/D$_{11}$) |
| 60 | 58 | | P3$_2$ | | | | | | A$_{10}$(MA$_2$)(/D$_{10}$) |
| 61 | 59 | | P3$_1$ | | | | | | A$_9$(MA$_1$)(/D$_9$) |
| 62 | 60 | V$_{CC}$ | | | | | | | |
| 63 | 61 | | P3$_0$ | | | | | | A$_8$(MA$_0$)(/D$_8$) |
| 64 | 62 | V$_{SS}$ | | | | | | | |
| 65 | 63 | | P2$_7$ | | | | | AN2$_7$ | A$_7$(/D$_7$) |
| 66 | 64 | | P2$_6$ | | | | | AN2$_6$ | A$_6$(/D$_6$) |
| 67 | 65 | | P2$_5$ | | | | | AN2$_5$ | A$_5$(/D$_5$) |
| 68 | 66 | | P2$_4$ | | | | | AN2$_4$ | A$_4$(/D$_4$) |
| 69 | 67 | | P2$_3$ | | | | | AN2$_3$ | A$_3$(/D$_3$) |
| 70 | 68 | | P2$_2$ | | | | | AN2$_2$ | A$_2$(/D$_2$) |
| 71 | 69 | | P2$_1$ | | | | | AN2$_1$ | A$_1$(/D$_1$) |
| 72 | 70 | | P2$_0$ | | | | | AN2$_0$ | A$_0$(/D$_0$) |
| 73 | 71 | | P1$_7$ | $\overline{\text{INT5}}$ | | | | | D$_{15}$ |
| 74 | 72 | | P1$_6$ | $\overline{\text{INT4}}$ | | | | | D$_{14}$ |
| 75 | 73 | | P1$_5$ | $\overline{\text{INT3}}$ | | | | | D$_{13}$ |
| 76 | 74 | | P1$_4$ | | | | | | D$_{12}$ |
| 77 | 75 | | P1$_3$ | | | | | | D$_{11}$ |
| 78 | 76 | | P1$_2$ | | | | | | D$_{10}$ |
| 79 | 77 | | P1$_1$ | | | | | | D$_9$ |
| 80 | 78 | | P1$_0$ | | | | | | D$_8$ |
| 81 | 79 | | P0$_7$ | | | | | AN0$_7$ | D$_7$ |
| 82 | 80 | | P0$_6$ | | | | | AN0$_6$ | D$_6$ |
| 83 | 81 | | P0$_5$ | | | | | AN0$_5$ | D$_5$ |
| 84 | 82 | | P0$_4$ | | | | | AN0$_4$ | D$_4$ |
| 85 | 83 | | P0$_3$ | | | | | AN0$_3$ | D$_3$ |
| 86 | 84 | | P0$_2$ | | | | | AN0$_2$ | D$_2$ |
| 87 | 85 | | P0$_1$ | | | | | AN0$_1$ | D$_1$ |
| 88 | 86 | | P0$_0$ | | | | | AN0$_0$ | D$_0$ |
| 89 | 87 | | P10$_7$ | $\overline{\text{KI}_3}$ | | | | AN$_7$ | |
| 90 | 88 | | P10$_6$ | $\overline{\text{KI}_2}$ | | | | AN$_6$ | |
| 91 | 89 | | P10$_5$ | $\overline{\text{KI}_1}$ | | | | AN$_5$ | |
| 92 | 90 | | P10$_4$ | $\overline{\text{KI}_0}$ | | | | AN$_4$ | |
| 93 | 91 | | P10$_3$ | | | | | AN$_3$ | |
| 94 | 92 | | P10$_2$ | | | | | AN$_2$ | |
| 95 | 93 | | P10$_1$ | | | | | AN$_1$ | |
| 96 | 94 | AV$_{SS}$ | | | | | | | |
| 97 | 95 | | P10$_0$ | | | | | AN$_0$ | |
| 98 | 96 | V$_{REF}$ | | | | | | | |
| 99 | 97 | AV$_{CC}$ | | | | | | | |
| 100 | 98 | | P9$_7$ | | | RxD4/SCL4/STxD4 | | $\overline{\text{AD}_{TRG}}$ | |

**Table 1.1.9. Pin description (1/4)**

| Port | Function | Pin name | I/O type | Description |
|---|---|---|---|---|
| | Power supply input | $V_{CC}$<br>$V_{SS}$ | I<br>I | 4.2 to 5.5 V or 3.0V to 3.6V.<br>0 V. |
| | CPU mode switch | $CNV_{SS}$ | I | Connect it to $V_{SS}$ : Single-chip or memory expansion mode<br>Connect it to $V_{CC}$ : Microprocessor mode |
| | External data bus width select input | BYTE | I | Selects the width of the data bus for external memory.<br>Connect it to $V_{SS}$ : A 16-bit width<br>Connect it to $V_{CC}$ : An 8-bit width |
| | Reset input | $\overline{RESET}$ | I | A “L” on this input resets the microcomputer. |
| | Clock input<br>Clock output | $X_{IN}$<br>$X_{OUT}$ | I<br>O | 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. |
| | Analog power supply input | $AV_{CC}$<br>$AV_{SS}$ | I<br>I | Connect this pin to $V_{CC}$.<br>Connect this pin to $V_{SS}$. |
| | Reference voltage input | $V_{REF}$ | I | This pin is a reference voltage input for the A-D converter. |
| P0 | I/O port | $P0_0$ to $P0_7$ | I/O | An 8-bit CMOS I/O port.<br>It has an input/output port direction register that allows the user to set each pin for input or output individually.<br>The user can specify in units of four bits via software whether or not they are tied to a pull-up resistor. |
| | Data bus | $D_0$ to $D_7$ | I/O | When set as a separate bus, these pins input and output 8 low-order data bits. |
| | Analog input port | $AN0_0$ to $AN0_7$ | I | $P0_0$ to $P0_7$ are analog input ports for the A-D converter. |
| P1 | I/O port | $P1_0$ to $P1_7$ | I/O | This is an 8-bit I/O port equivalent to P0. |
| | External interrupt input port | $INT_3$ to $INT_5$ | I | $P1_5$ to $P1_7$ function as external interrupt pins. |
| | Data bus | $D_8$ to $D_{15}$ | I/O | When set as a separate bus, these pins input and output 8 high-order data bits. |
| P2 | I/O port | $P2_0$ to $P2_7$ | I/O | This is an 8-bit I/O port equivalent to P0. |
| | Address bus | $A_0$ to $A_7$ | O | These pins output 8 low-order address bits. |
| | Address bus/data bus | $A_0/D_0$ to $A_7/D_7$ | I/O | If a multiplexed bus is set, these pins input and output data and output 8 low-order address bits separated in time by multiplexing. |
| | Analog input port | $AN2_0$ to $AN2_7$ | I | $P2_0$ to $P2_7$ are analog input ports for the A-D converter. |
| P3 | I/O port | $P3_0$ to $P3_7$ | I/O | This is an 8-bit I/O port equivalent to P0. |
| | Address bus | $A_8$ to $A_{15}$ | O | These pins output 8 middle-order address bits. |
| | Address bus/data bus | $A_8/D_8$ to $A_{15}/D_{15}$ | I/O | If the external bus is set as a 16-bit wide multiplexed bus, these pins output 8 middle-order address bits, and input and output 8 middle-order data separated in time by multiplexing. |
| | Address bus | MA0 to MA7 | O | If accessing to DRAM area, these pins output row address and column address separated in time by multiplexing. |
| P4 | I/O port | $P4_0$ to $P4_7$ | I/O | This is an 8-bit I/O port equivalent to P0. |
| | Address bus | $A_{16}$ to $A_{22}$<br>$\overline{A_{23}}$ | O | These pins output 8 high-order address bits.<br>Highest address bit ($\overline{A_{23}}$) outputs inversely. |
| | Chip select | $\overline{CS_0}$ to $\overline{CS_3}$ | O | $P4_0$ to $P4_7$ are chip select output pins to specify access area. |
| | Address bus | MA8 to MA12 | O | If accessing to DRAM area, these pins output row address and column address separated in time by multiplexing. |

**Table 1.1.10. Pin description (2/4)**

| Port | Function | Pin name | I/O type | Description |
|---|---|---|---|---|
| P5 | I/O port | P5$_0$ to P5$_7$ | I/O | This is an 8-bit I/O port equivalent to P0. |
| | Clock output | CLKOUT | I/O | P5$_3$ in this port outputs a divide-by-8 or divide-by-32 clock of XIN or a clock of the same frequency as XCIN. |
| | Bus control | $\overline{WRL}$ / $\overline{WR}$,<br>$\overline{WRH}$ / BHE,<br>$\overline{RD}$ | O<br>O<br>O | Output $\overline{WRL}$, $\overline{WRH}$ and $\overline{RD}$, or $\overline{WR}$, $\overline{BHE}$ and $\overline{RD}$ bus control signals.<br>■ $\overline{WRL}$, $\overline{WRH}$, and $\overline{RD}$ selected<br>In 16-bit data bus, data is written to even addresses when the $\overline{WRL}$ signal is “L”.<br>Data is written to odd addresses when the $\overline{WRH}$ signal is “L”.<br>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”.<br>Data is read when $\overline{RD}$ is “L”.<br>Odd addresses are accessed when $\overline{BHE}$ is “L”. Even addresses are accessed when BHE is “H”.<br>Use $\overline{WR}$, $\overline{BHE}$, and $\overline{RD}$ when all external memory is an 8-bit data bus. |
| | | BCLK,<br>HOLD,<br>$\overline{HLDA}$<br>ALE,<br>RDY | O<br>I<br>O<br>O<br>I | Output operation clock for CPU.<br>While the input level at the $\overline{HOLD}$ pin is “L”, the microcomputer is placed in the hold state.<br>While in the hold state, $\overline{HLDA}$ outputs a “L” level.<br>ALE is used to latch the address.<br>While the input level of the RDY pin is “L”, the microcomputer is in the ready state. |
| | Bus control for DRAM | $\overline{DW}$,<br>$\overline{CASL}$,<br>$\overline{CASH}$,<br>$\overline{RAS}$ | O<br>O<br>O<br>O | When $\overline{DW}$ signal is “L”, write to DRAM.<br>Timing signal when latching to line address of even address.<br>Timing signal when latching to line address of odd address.<br>Timing signal when latching to row address. |
| P6 | I/O port | P6$_0$ to P6$_7$ | I/O | This is an 8-bit I/O port equivalent to P0. |
| | UART port | $\overline{CTS}$/$\overline{RTS}$/$\overline{SS}$<br>CLK<br>RxD/SCL/STxD<br>TxD/SDA/SRxD | I/O | P6$_0$ to P6$_3$ are I/O ports for UART0.<br>P6$_4$ to P6$_7$ are I/O ports for UART1. |
| | Intelligent I/O port | OUTC/ISCLK | I/O | ISCLK is a clock I/O port for intelligent I/O communication.<br>OUTC is an output port for waveform generation function. |
| P7 | I/O port | P7$_0$ to P7$_7$ | I/O | This is an 8-bit I/O port equivalent to P0.<br>However, P7$_0$ and P7$_1$ are N-channel open drain outputs. |
| | Timer A port | TAOUT<br>TAIN | O<br>I | P7$_0$ to P7$_7$ are I/O ports for timers A0–A3. |
| | Timer B port | TBIN | I | P7$_1$ is an input port for timer B5. |
| | Three phase motor control output port | V, $\overline{V}$<br>W, $\overline{W}$ | O | P7$_2$ and P7$_3$ are V phase outputs.<br>P7$_4$ and P7$_5$ are W phase outputs. |
| | UART port | $\overline{CTS}$/$\overline{RTS}$/$\overline{SS}$<br>CLK<br>RxD/SCL/STxD<br>TxD/SDA/SRxD | I/O | P7$_0$ to P7$_3$ are I/O ports for UART2. |
| | Intelligent I/O port | INPC/OUTC<br>ISCLK/ISTxD/<br>ISRxD<br>IEOUT/IEIN<br>BEOUT/BEIN | I/O | INPC is an input port for time measurement function.<br>OUTC is an output port for waveform generation function.<br>ISCLK is a clock I/O port for intelligent I/O communication.<br>ISTxD/IEOUT/BEOUT is transmit data output port for intelligent I/O communication.<br>ISRxD/IEIN/BEIN is receive data input port for intelligent I/O communication. |
| | CANOUT<br>CANIN | CAN | O<br>I | P7$_6$ and P7$_7$ are I/O ports for CAN communication function. |

**Table 1.1.11. Pin description (3/4)**

| Port | Function | Pin name | I/O type | Description |
|---|---|---|---|---|
| P8 | I/O port | P80-P84, P86, P87 | I/O | This is a 7-bit I/O port equivalent to P0. |
| | Sub clock input | XCIN | I | P86 and P87 function as I/O ports for the sub clock generating circuit by software. Connect a crystal between the XCIN and the XCOUT pins. |
| | Sub clock output | XCOUT | O | |
| | Low-pass filter connect pin for PLL frequency synthesizer | VCOUT | O | When using PLL frequency synthesizer, connect P87 to a low-pass filter. To stabilize PLL frequency, connect P86 to Vss. |
| | Timer A port | TA4OUT<br>TA4IN | O<br>I | P80 to P81 are I/O ports for timer A4. |
| | Three phase motor control output port | U, $\overline{U}$ | O | P80 and P81 are U phase output ports. |
| | External interrupt input port | INT0 to INT2 | I | P82 to P84 are external interrupt input ports. |
| | Intelligent I/O port | INPC/ISRxD/BEIN | I | INPC is an input port for time measurement function. ISRxD/BEIN is receive data input port for intelligent I/O communication. |
| | Input port | P85/$\overline{NMI}$ | I | Input port and input ports for $\overline{NMI}$ interrupt. |
| P9 | I/O port | P90 to P97 | I/O | This is an 8-bit I/O port equivalent to P0. |
| | Timer B port | TB0IN to TB4IN | I | P90 to P94 are input port for timer B4. |
| | UART port | $\overline{CTS}$/$\overline{RTS}$/$\overline{SS}$<br>CLK<br>RxD/SCL/STxD<br>TxD/SDA/SRxD | I/O<br>I/O<br>I/O<br>I/O | P90 to P93 are I/O ports for UART3.<br>P94 to P97 are I/O ports for UART4. |
| | D-A output port | DA0, DA1 | O | P93 and P94 are D-A output ports. |
| | A-D related port | ANEX1, ANEX2<br>ADTRG | I<br>I | P95 to P96 are expanded input port for A-D converter.<br>P97 is A-D trigger input port. |
| | Intelligent I/O port | OUTC/IEOUT<br><br><br>IEIN | I/O<br><br><br>I | OUTC is an output port for waveform generation function.<br>IEOUT is transmit data output port for intelligent I/O communication.<br>IEIN is receive data input port for intelligent I/O communication. |
| | The protect register prevents a false write to P9 direction register and function select register A3. | | | |
| P10 | I/O port | P100 to P107 | I/O | This is an 8-bit I/O port equivalent to P0. |
| | Key input interrupt port | KI0 to KI3 | I | P104 to P107 are key input interrupt ports. |
| | Analog input port | AN0 to AN7 | I | P100 to P107 are analog input ports for A-D convertor. |

**Table 1.1.12. Pin description (4/4)**

| Port | Function | Pin name | I/O type | Description |
|---|---|---|---|---|
| P11 (Note) | I/O port | $P11_0$ to $P11_4$ | I/O | This is an 5-bit I/O port equivalent to P0. |
| | Intelligent I/O port | INPC/OUTC<br>ISCLK<br>ISTxD/ISRxD<br>$BE_{OUT}$/$BE_{IN}$ | I/O<br>I/O<br><br>I/O | INPC is an input port for time measurement function.<br>OUTC is an output port for waveform generation function.<br>ISCLK is a clock I/O port for intelligent I/O communication.<br>ISTxD/$BE_{OUT}$ is transmit data output port for intelligent I/O communication.<br>ISRxD/$BE_{IN}$ is receive data input port for intelligent I/O communication. |
| P12 (Note) | I/O port | $P12_0$ to $P12_7$ | I/O | This is an 8-bit I/O port equivalent to P0. |
| | Intelligent I/O port | OUTC | O | OUTC is an output port for waveform generation function. |
| P13 (Note) | I/O port | $P13_0$ to $P13_7$ | I/O | This is an 8-bit I/O port equivalent to P0. |
| | Intelligent I/O port | OUTC<br>ISCLK/ISTxD/<br>ISRxD<br>$IE_{OUT}$/$IE_{IN}$ | I/O<br>I/O<br>I/O<br>I/O | OUTC is an output port for waveform generation function.<br>ISCLK is a clock I/O port for intelligent I/O communication.<br>ISTxD/$IE_{OUT}$ is transmit data output port for intelligent I/O communication.<br>ISRxD/$IE_{IN}$ is receive data input port for intelligent I/O communication. |
| P14 (Note) | I/O port | $P14_0$ to $P14_6$ | I/O | This is a 7-bit I/O port equivalent to P0. |
| | Intelligent I/O port | INPC/OUTC | I/O | INPC is an input port for time measurement function.<br>OUTC is an output port for waveform generation function. |
| P15 (Note) | I/O port | $P15_0$ to $P15_7$ | I/O | This is an 8-bit I/O port equivalent to P0. |
| | Intelligent I/O port | INPC/OUTC<br>ISCLK/ISTxD/<br>ISRxD<br>$BE_{OUT}$/$BE_{IN}$ | I/O<br>I/O<br><br>I/O | INPC is an input port for time measurement function.<br>OUTC is an output port for waveform generation function.<br>ISCLK is a clock I/O port for intelligent I/O communication.<br>ISTxD/$BE_{OUT}$ is transmit data output port for intelligent I/O communication.<br>ISRxD/$BE_{IN}$ is receive data input port for intelligent I/O communication. |
| | Analog input port | $AN15_0$ to $AN15_7$ | I | $P15_0$ to $P15_7$ are analog input ports for A-D convertor. |

Note :Port P11 to P15 exist in 144-pin version.

## Block Diagram

The M32C/83 group includes the following devices in a single-chip. ROM and RAM for code instructions and data, storage, CPU for executing operation and peripheral functions such as timer, serial I/O, D-A converter, DMAC, CRC operation circuit, A-D converter, DRAM controller, intelligent I/O and I/O ports. Figure 1.1.6 is a block diagram of the M32C/83 group (144-pin version).

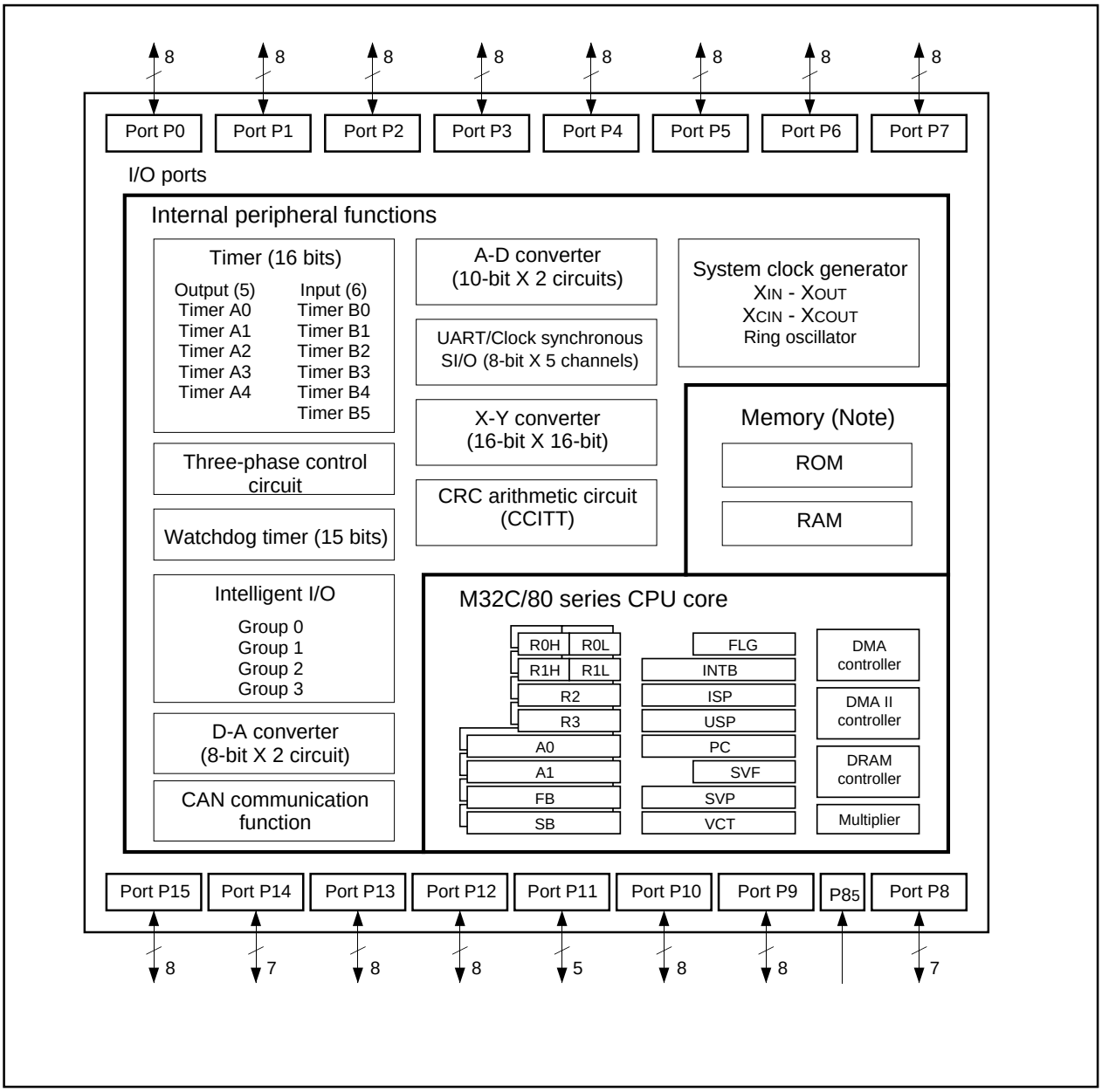



**Figure 1.1.6. Block diagram of the M32C/83 group (144-pin version)**

## Memory

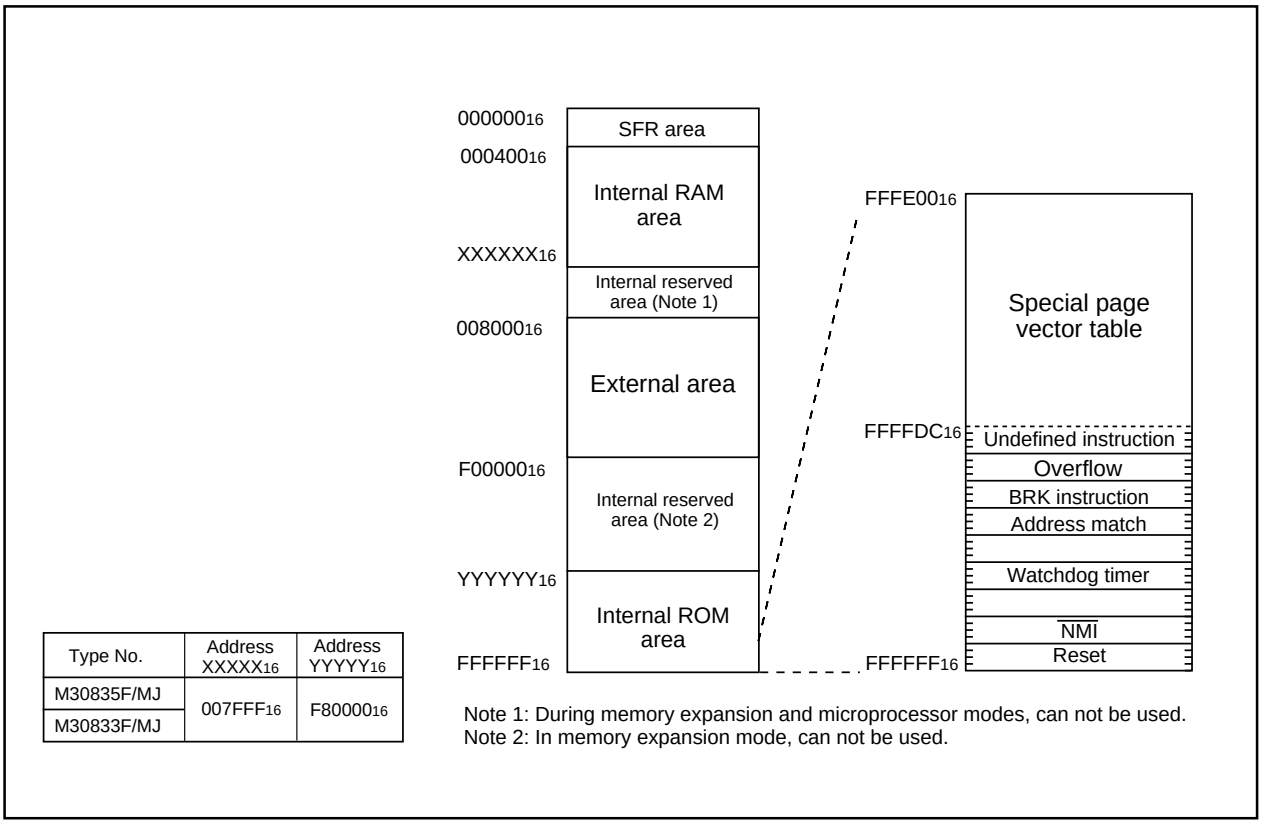
Figure 1.2.1 is a memory map of the M32C/83 group. The address space extends 16 Mbytes from address 000000$_{16}$ to FFFFFF$_{16}$. From FFFFFF$_{16}$ down is ROM. For example, in the M30835FJGP, there are 512K bytes of internal ROM from F80000$_{16}$ to FFFFFF$_{16}$. The vector table for fixed interrupts such as the reset and $\overline{\text{NMI}}$ are mapped to FFFFDC$_{16}$ to FFFFFF$_{16}$. The starting address of the interrupt routine is stored here. The address of the vector table for timer interrupts, etc., can be set as desired using the internal register (INTB). See the section on interrupts for details.

From 000400$_{16}$ up is RAM. For example, in the M30835FJGP, 31 Kbytes of internal RAM are mapped to the space from 000400$_{16}$ to 007FFF$_{16}$. 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 from 000000$_{16}$ to 0003FF$_{16}$. This area accommodates the control registers for peripheral devices such as I/O ports, A-D converter, serial I/O, and timers, etc. Any part of the SFR area that is not occupied is reserved and cannot be used for any other purpose.

The special page vector table is mapped from FFFE00$_{16}$ to FFFFDB$_{16}$. If the starting addresses of subroutines or the destination addresses of jumps are stored here, subroutine call instructions and jump instructions can be used as 2-byte instructions, reducing the number of program steps.

In memory expansion mode and microprocessor mode, a part of the spaces are reserved and cannot be used.




| Type No. | Address XXXXX$_{16}$ | Address YYYYY$_{16}$ |
|---|---|---|
| M30835F/MJ | 007FFF$_{16}$ | F80000$_{16}$ |
| M30833F/MJ | | |

**Figure 1.2.1. Memory map**

## Central Processing Unit (CPU)

The CPU has a total of 28 registers shown in Figure 1.3.1. Eight of these registers (R0, R1, R2, R3, A0, A1, SB and FB) come in two sets; therefore, these have two register banks.

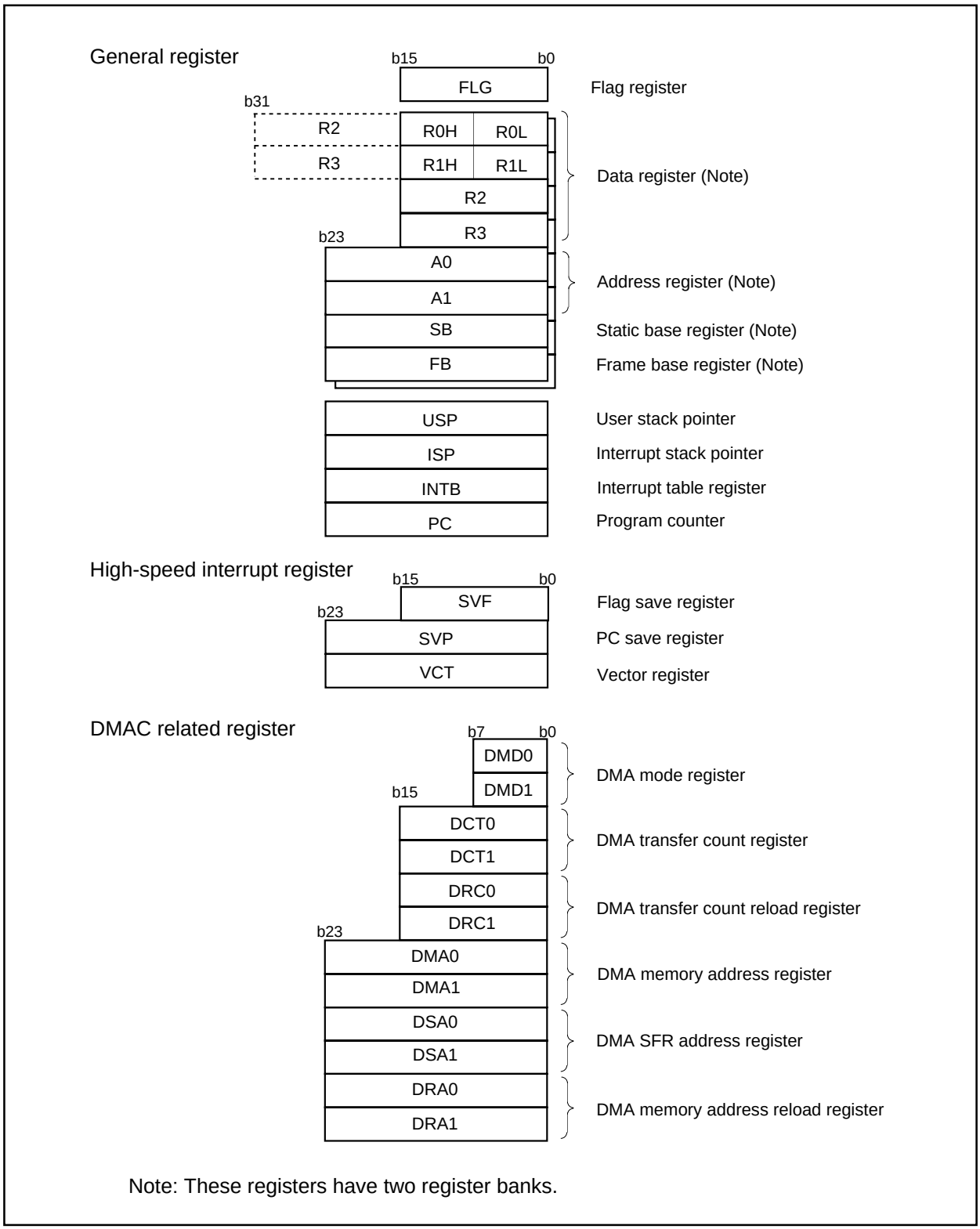



**Figure 1.3.1. Central processing unit register**

### (1) Data registers (R0, R0H, R0L, R1, R1H, R1L, R2, R3, R2R0 and R3R1)

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). Registers R2 and R0, as well as R3 and R1 can function as 32-bit data registers (R2R0/R3R1).

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

Address registers (A0 and A1) are configured with 24 bits, and have functions equivalent to those of data registers. These registers can also function as address register, indirect addressing and address register relative addressing.

### (3) Static base register (SB)

Static base register (SB) is configured with 24 bits, and is used for SB relative addressing.

### (4) Frame base register (FB)

Frame base register (FB) is configured with 24 bits, and is used for FB relative addressing.

### (5) Program counter (PC)

Program counter (PC) is configured with 24 bits, indicating the address of an instruction to be executed.

### (6) Interrupt table register (INTB)

Interrupt table register (INTB) is configured with 24 bits, indicating the start address of an interrupt vector table.

### (7) User stack pointer (USP), interrupt stack pointer (ISP)

Stack pointer comes in two types: user stack pointer (USP) and interrupt stack pointer (ISP), each configured with 24 bits.

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

To execute efficienly set USP and ISP to an even number.

### (8) Save flag register (SVF)

This register consists of 16 bits and is used to save the flag register when a high-speed interrupt is generated.

### (9) Save PC register (SVP)

This register consists of 24 bits and is used to save the program counter when a high-speed interrupt is generated.

This register consist of 24 bits and is used to indicate a jump address when a high-speed interrupt is generated.

### (10) Vector register (VCT)

This register consists of 24 bits and is used to indicate the jump address when a high-speed interrupt is generated.

### (11) DMA mode registers (DMD0/DMD1)

These registers consist of 8 bits and are used to set the transfer mode, etc. for DMA.

### (12) DMA transfer count registers (DCT0/DCT1)

These registers consist of 16 bits and are used to set the number of DMA transfers performed.

### (13) DMA transfer count reload registers (DRC0/DRC1)

These registers consist of 16 bits and are used to reload the DMA transfer count registers.

### (14) DMA memory address registers (DMA0/DMA1)

These registers consist of 24 bits and are used to set a memory address at the source or destination of DMA transfer.

### (15) DMA SFR address registers (DSA0/DSA1)

These registers consist of 24 bits and are used to set a fixed address at the source or destination of DMA transfer.

### (16) DMA memory address reload registers (DRA0/DRA1)

These registers consist of 24 bits and are used to reload the DMA memory address registers.

### (17) Flag register (FLG)

Flag register (FLG) is configured with 11 bits, each bit is used as a flag. Figure 1.3.2 shows the flag register (FLG). The following explains the function of each flag:

- **Bit 0: Carry flag (C)**
  This flag retains a carry, borrow, or shift-out bit that has occurred in the arithmetic/logic unit.

- **Bit 1: Debug flag (D)**
  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)**
  This flag is set to “1” when an arithmetic operation resulted in 0; otherwise, cleared to “0”.

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

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)**

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

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

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)**

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 Numbers. 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**

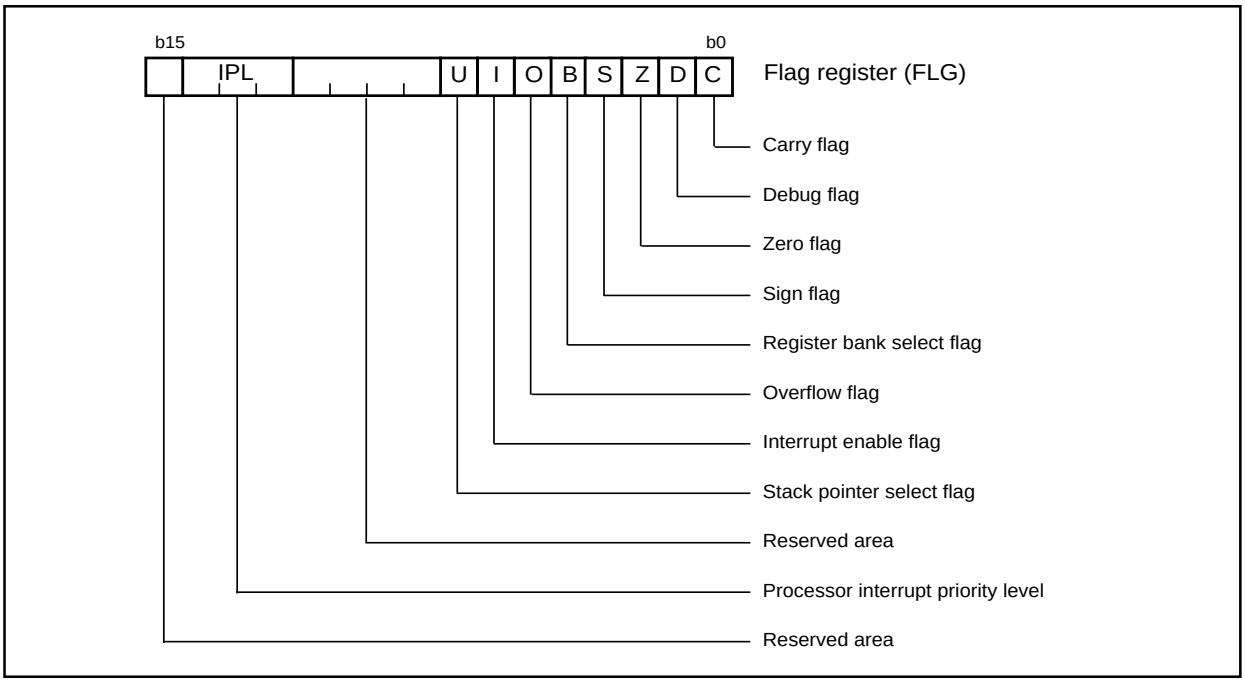



**Figure 1.3.2. Flag register (FLG)**

## Reset

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

When the supply voltage is in the range where operation is guaranteed, a reset is enabled by holding the reset pin Low (0.2Vcc max.) for at least 20 cycles. When the reset pin level is then returned to High while main clock is stable, the reset status is cancelled and program execution resumes from the address in the reset vector table.

Since the value of RAM is indeterminate when power is applied, the initial values must be set. Also, if a reset signal is input during write to RAM, the access to the RAM will be interrupted. Consequently, the value of the RAM being written may change to an unintended value due to the interruption.

Figure 1.4.1 shows the example reset circuit. Figure 1.4.2 shows the reset sequence.

Table 1.4.1 shows the status of other pins while the $\overline{\text{RESET}}$ pin level is Low. Figures 1.4.3 and 1.4.4 show the internal status of the microcomputer immediately after the reset is cancelled.

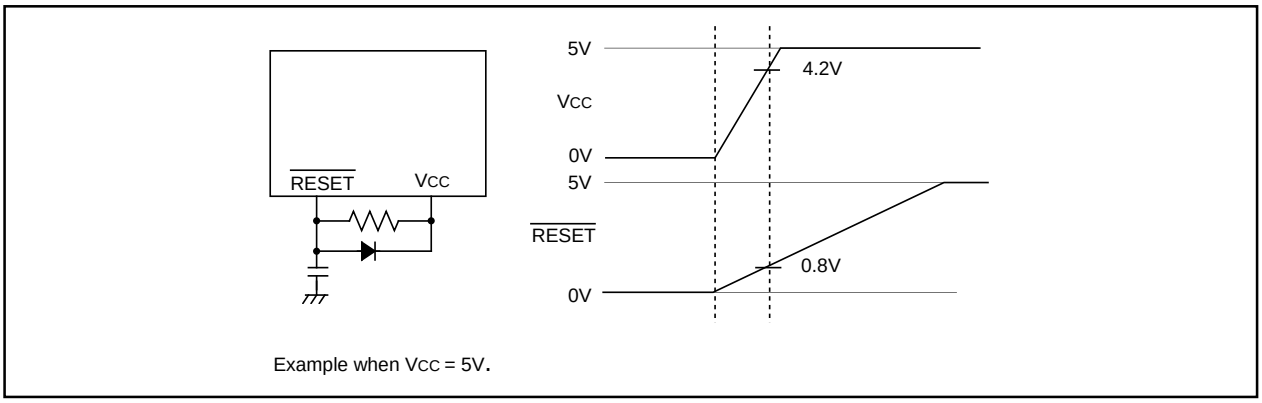



**Figure 1.4.1. Example reset circuit**

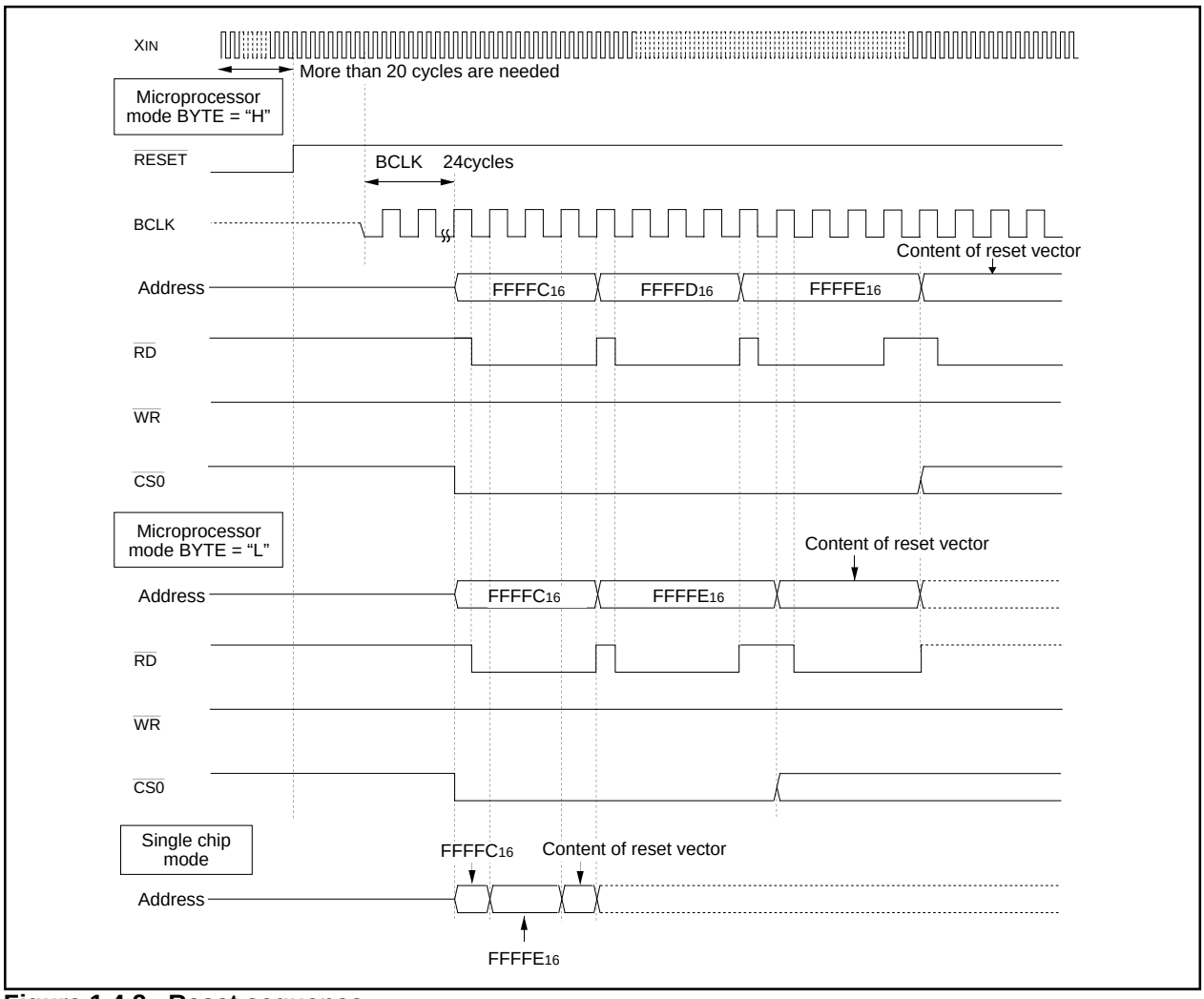



**Figure 1.4.2. Reset sequence**

**Table 1.4.1. Pin status when $\overline{\text{RESET}}$ pin level is "L"**

| Pin name | Status | | |
|---|---|---|---|
| | CNVSS = VSS | CNVSS = VCC | |
| | | BYTE = VSS | BYTE = VCC |
| P0 | Input port (floating) | Data input (floating) | |
| P1 | Input port (floating) | Data input (floating) | Input port (floating) |
| P2, P3, P4 | Input port (floating) | Address output (undefined) | |
| P50 | Input port (floating) | $\overline{\text{WR}}$ output ("H" level output) | |
| P51 | Input port (floating) | $\overline{\text{BHE}}$ output (undefined) | |
| P52 | Input port (floating) | $\overline{\text{RD}}$ output ("H" level output) | |
| P53 | Input port (floating) | BCLK output | |
| P54 | Input port (floating) | $\overline{\text{HLDA}}$ output (The output value depends on the input to the $\overline{\text{HOLD}}$ pin) | |
| P55 | Input port (floating) | $\overline{\text{HOLD}}$ input (floating) | |
| P56 | Input port (floating) | $\overline{\text{RAS}}$ output | |
| P57 | Input port (floating) | $\overline{\text{RDY}}$ input (floating) | |
| P6 to P15 (Note) | Input port (floating) | Input port (floating) | |

Note :Port P11 to P15 exists in 144-pin version.

| | | | | | |
|---|---|---|---|---|---|
| (1) | Processor mode register 0 | (Note 1) (0004$_{16}$) | 80$_{16}$ | (26) UART2 receive /ACK interrupt control register | (006B$_{16}$) XXXX?000 |
| (2) | Processor mode register 1 | (0005$_{16}$) | X00000XX | (27) Timer A0 interrupt control register | (006C$_{16}$) XXXX?000 |
| (3) | System clock control register 0 | (0006$_{16}$) | 0000X000 | (28) UART3 receive/ACK interrupt control register | (006D$_{16}$) XXXX?000 |
| (4) | System clock control register 1 | (0007$_{16}$) | 20$_{16}$ | (29) Timer A2 interrupt control register | (006E$_{16}$) XXXX?000 |
| (5) | Wait control register | (0008$_{16}$) | FF$_{16}$ | (30) UART4 receive/ACK interrupt control register | (006F$_{16}$) XXXX?000 |
| (6) | Address match interrupt control register | (0009$_{16}$) | XXXX0000 | (31) Timer A4 interrupt control register | (0070$_{16}$) XXXX?000 |
| (7) | Protect register | (000A$_{16}$) | XXXX0000 | (32) UART0/UART3 bus collision detection interrupt control register | (0071$_{16}$) XXXX?000 |
| (8) | External data bus width control register | (Note 2) (000B$_{16}$) | XXXXX000 | (33) UART0 receive/ACK interrupt control register | (0072$_{16}$) XXXX?000 |
| (9) | Main clock divided register | (000C$_{16}$) | XXX01000 | (34) A-D0 interrupt control register | (0073$_{16}$) XXXX?000 |
| (10) | Oscillation stop detect register | (000D$_{16}$) | 00$_{16}$ | (35) UART1 receive/ACK interrupt control register | (0074$_{16}$) XXXX?000 |
| (11) | Watchdog timer start register | (000E$_{16}$) | ??$_{16}$ | (36) Intelligent I/O interrupt control register 0 | (0075$_{16}$) XXXX?000 |
| (12) | Watchdog timer control register | (000F$_{16}$) | 000?????? | (37) Timer B1 interrupt control register | (0076$_{16}$) XXXX?000 |
| (13) | Address match interrupt register 0 | (0010$_{16}$) | 00$_{16}$ | (38) Intelligent I/O interrupt control register 2 | (0077$_{16}$) XXXX?000 |
| | | (0011$_{16}$) | 00$_{16}$ | (39) Timer B3 interrupt control register | (0078$_{16}$) XXXX?000 |
| | | (0012$_{16}$) | 00$_{16}$ | (40) Intelligent I/O interrupt control register 4 | (0079$_{16}$) XXXX?000 |
| (14) | Address match interrupt register 1 | (0014$_{16}$) | 00$_{16}$ | (41) INT5 interrupt control register | (007A$_{16}$) XX00?000 |
| | | (0015$_{16}$) | 00$_{16}$ | (42) Intelligent I/O interrupt control register 6 | (007B$_{16}$) XXXX?000 |
| | | (0016$_{16}$) | 00$_{16}$ | (43) INT3 interrupt control register | (007C$_{16}$) XX00?000 |
| (15) | VDC control register for PLL | (0017$_{16}$) | XXXXXX01 | (44) Intelligent I/O interrupt control register 8 | (007D$_{16}$) XXXX?000 |
| (16) | Address match interrupt register 2 | (0018$_{16}$) | 00$_{16}$ | (45) INT1 interrupt control register | (007E$_{16}$) XX00?000 |
| | | (0019$_{16}$) | 00$_{16}$ | (46) Intelligent I/O interrupt control register 10/ CAN interrupt 1 control register | (007F$_{16}$) XXXX?000 |
| | | (001A$_{16}$) | 00$_{16}$ | (47) Intelligent I/O interrupt control register 11/ CAN interrupt 2 control register | (0081$_{16}$) XXXX?000 |
| (17) | VDD control register 0 | (001B$_{16}$) | 00$_{16}$ | (48) A-D1 interrupt control register | (0086$_{16}$) XXXX?000 |
| (18) | Address match interrupt register 3 | (001C$_{16}$) | 00$_{16}$ | (49) DMA1 interrupt control register | (0088$_{16}$) XXXX?000 |
| | | (001D$_{16}$) | 00$_{16}$ | (50) UART2 transmit /NACK interrupt control register | (0089$_{16}$) XXXX?000 |
| | | (001E$_{16}$) | 00$_{16}$ | (51) DMA3 interrupt control register | (008A$_{16}$) XXXX?000 |
| (19) | VDD control register 1 | (001F$_{16}$) | 00$_{16}$ | (52) UART3 transmit /NACK interrupt control register | (008B$_{16}$) XXXX?000 |
| (20) | DRAM control register | (0040$_{16}$) | ?XXX???? | (53) Timer A1 interrupt control register | (008C$_{16}$) XXXX?000 |
| (21) | DRAM refresh interval set register | (0041$_{16}$) | ??$_{16}$ | (54) UART4 transmit /NACK interrupt control register | (008D$_{16}$) XXXX?000 |
| (22) | Flash memory control register 0 | (0057$_{16}$) | XX000001 | (55) Timer A3 interrupt control register | (008E$_{16}$) XXXX?000 |
| (23) | DMA0 interrupt control register | (0068$_{16}$) | XXXX?000 | (56) UART2 bus collision detection interrupt control register | (008F$_{16}$) XXXX?000 |
| (24) | Timer B5 interrupt control register | (0069$_{16}$) | XXXX?000 | (57) UART0 transmit /NACK interrupt control register | (0090$_{16}$) XXXX?000 |
| (25) | DMA2 interrupt control register | (006A$_{16}$) | XXXX?000 | (58) UART1/UART4 bus collision detection interrupt control register | (0091$_{16}$) XXXX?000 |

x : Nothing is mapped to this bit
? : Undefined

The content of other registers and RAM is undefined when the microcomputer is reset. The initial values must therefore be set.
Note 1: When the $V_{CC}$ level is applied to the $CNV_{SS}$ pin, it is 03$_{16}$ at a reset.
Note 2: When the BYTE pin is "L", bit 3 is "1". When the BYTE pin is "H", bit 3 is "0".

**Figure 1.4.3. Device's internal status after a reset is cleared (1/10)**

| | | | | | |
|---|---|---|---|---|---|
| (59) | UART1 transmit/NACK interrupt control register | (0092$_{16}$) XXXX?000 | (92) | Interrupt enable register 7 | (00B7$_{16}$) 0XX00000 |
| (60) | Key input interrupt control register | (0093$_{16}$) XXXX?000 | (93) | Interrupt enable register 8 | (00B8$_{16}$) 00X00000 |
| (61) | Timer B0 interrupt control register | (0094$_{16}$) XXXX?000 | (94) | Interrupt enable register 9 | (00B9$_{16}$) 0XX00000 |
| (62) | Intelligent I/O interrupt control register 1 | (0095$_{16}$) XXXX?000 | (95) | Interrupt enable register 10 | (00BA$_{16}$) 0XX00000 |
| (63) | Timer B2 interrupt control register | (0096$_{16}$) XXXX?000 | (96) | Interrupt enable register 11 | (00BB$_{16}$) 0XX00000 |
| (64) | Intelligent I/O interrupt control register 3 | (0097$_{16}$) XXXX?000 | (97) | Group 0 time measurement/waveform generate register 0 | (00C0$_{16}$) ??$_{16}$ |
| (65) | Timer B4 interrupt control register | (0098$_{16}$) XXXX?000 | | | (00C1$_{16}$) ??$_{16}$ |
| (66) | Intelligent I/O interrupt control register 5 | (0099$_{16}$) XXXX?000 | (98) | Group 0 time measurement/waveform generate register 1 | (00C2$_{16}$) ??$_{16}$ |
| (67) | INT4 interrupt control register | (009A$_{16}$) XX00?000 | | | (00C3$_{16}$) ??$_{16}$ |
| (68) | Intelligent I/O interrupt control register 7 | (009B$_{16}$) XXXX?000 | (99) | Group 0 time measurement/waveform generate register 2 | (00C4$_{16}$) ??$_{16}$ |
| (69) | INT2 interrupt control register | (009C$_{16}$) XX00?000 | | | (00C5$_{16}$) ??$_{16}$ |
| (70) | Intelligent I/O interrupt control register 9/ CAN interrupt 0 control register | (009D$_{16}$) XXXX?000 | (100) | Group 0 time measurement/waveform generate register 3 | (00C6$_{16}$) ??$_{16}$ |
| (71) | INT0 interrupt control register | (009E$_{16}$) XX00?000 | | | (00C7$_{16}$) ??$_{16}$ |
| (72) | Exit priority register | (009F$_{16}$) XX0X0000 | (101) | Group 0 time measurement/waveform generate register 4 | (00C8$_{16}$) ??$_{16}$ |
| (73) | Interrupt request register 0 | (00A0$_{16}$) XX00X00X | | | (00C9$_{16}$) ??$_{16}$ |
| (74) | Interrupt request register 1 | (00A1$_{16}$) XX00X00X | (102) | Group 0 time measurement/waveform generate register 5 | (00CA$_{16}$) ??$_{16}$ |
| (75) | Interrupt request register 2 | (00A2$_{16}$) XX00X0XX | | | (00CB$_{16}$) ??$_{16}$ |
| (76) | Interrupt request register 3 | (00A3$_{16}$) XX00000X | (103) | Group 0 time measurement/waveform generate register 6 | (00CC$_{16}$) ??$_{16}$ |
| (77) | Interrupt request register 4 | (00A4$_{16}$) 00X0000X | | | (00CD$_{16}$) ??$_{16}$ |
| (78) | Interrupt request register 5 | (00A5$_{16}$) XX00000X | (104) | Group 0 time measurement/waveform generate register 7 | (00CE$_{16}$) ??$_{16}$ |
| (79) | Interrupt request register 6 | (00A6$_{16}$) XX00000X | | | (00CF$_{16}$) ??$_{16}$ |
| (80) | Interrupt request register 7 | (00A7$_{16}$) 0X00000X | (105) | Group 0 waveform generate control register 0 | (00D0$_{16}$) 0X00X000 |
| (81) | Interrupt request register 8 | (00A8$_{16}$) 00X0000X | (106) | Group 0 waveform generate control register 1 | (00D1$_{16}$) 0X00X000 |
| (82) | Interrupt request register 9 | (00A9$_{16}$) 0X00000X | (107) | Group 0 waveform generate control register 4 | (00D4$_{16}$) 0X00X000 |
| (83) | Interrupt request register 10 | (00AA$_{16}$) 0X00000X | (108) | Group 0 waveform generate control register 5 | (00D5$_{16}$) 0X00X000 |
| (84) | Interrupt request register 11 | (00AB$_{16}$) 0X00000X | (109) | Group 0 time measurement control register 0 | (00D8$_{16}$) 00$_{16}$ |
| (85) | Interrupt enable register 0 | (00B0$_{16}$) XX00X000 | (110) | Group 0 time measurement control register 1 | (00D9$_{16}$) 00$_{16}$ |
| (86) | Interrupt enable register 1 | (00B1$_{16}$) XX00X000 | (111) | Group 0 time measurement control register 2 | (00DA$_{16}$) 00$_{16}$ |
| (87) | Interrupt enable register 2 | (00B2$_{16}$) XX00X0X0 | (112) | Group 0 time measurement control register 3 | (00DB$_{16}$) 00$_{16}$ |
| (88) | Interrupt enable register 3 | (00B3$_{16}$) XX000000 | (113) | Group 0 time measurement control register 4 | (00DC$_{16}$) 00$_{16}$ |
| (89) | Interrupt enable register 4 | (00B4$_{16}$) 00X00000 | (114) | Group 0 time measurement control register 5 | (00DD$_{16}$) 00$_{16}$ |
| (90) | Interrupt enable register 5 | (00B5$_{16}$) XX000000 | (115) | Group 0 time measurement control register 6 | (00DE$_{16}$) 00$_{16}$ |
| (91) | Interrupt enable register 6 | (00B6$_{16}$) XX000000 | (116) | Group 0 time measurement control register 7 | (00DF$_{16}$) 00$_{16}$ |

x : Nothing is mapped to this bit
? : Undefined

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

**Figure 1.4.3. Device's internal status after a reset is cleared (2/10)**

| | | |
|---|---|---|
| (117) Group 0 base timer register | (00E0$_{16}$) | ??$_{16}$ |
| | (00E1$_{16}$) | ??$_{16}$ |
| (118) Group 0 base timer control register 0 | (00E2$_{16}$) | 00$_{16}$ |
| (119) Group 0 base timer control register 1 | (00E3$_{16}$) | 00$_{16}$ |
| (120) Group 0 time measurement prescaler register 6 | (00E4$_{16}$) | 00$_{16}$ |
| (121) Group 0 time measurement prescaler register 7 | (00E5$_{16}$) | 00$_{16}$ |
| (122) Group 0 function enable register | (00E6$_{16}$) | 00$_{16}$ |
| (123) Group 0 function select register | (00E7$_{16}$) | 00$_{16}$ |
| (124) Group 0 SI/O receive buffer register | (00E8$_{16}$) | ??$_{16}$ |
| | (00E9$_{16}$) | XX00XXXX |
| (125) Group 0 transmit buffer/receive data register | (00EA$_{16}$) | ??$_{16}$ |
| (126) Group 0 receive input register | (00EC$_{16}$) | ??$_{16}$ |
| (127) Group 0 SI/O communication mode register | (00ED$_{16}$) | 00$_{16}$ |
| (128) Group 0 transmit output register | (00EE$_{16}$) | ??$_{16}$ |
| (129) Group 0 SI/O communication control register | (00EF$_{16}$) | 0000X011 |
| (130) Group 0 data compare register 0 | (00F0$_{16}$) | ??$_{16}$ |
| (131) Group 0 data compare register 1 | (00F1$_{16}$) | ??$_{16}$ |
| (132) Group 0 data compare register 2 | (00F2$_{16}$) | ??$_{16}$ |
| (133) Group 0 data compare register 3 | (00F3$_{16}$) | ??$_{16}$ |
| (134) Group 0 data mask register 0 | (00F4$_{16}$) | ??$_{16}$ |
| (135) Group 0 data mask register 1 | (00F5$_{16}$) | ??$_{16}$ |
| (136) Group 0 receive CRC code register | (00F8$_{16}$) | ??$_{16}$ |
| | (00F9$_{16}$) | ??$_{16}$ |
| (137) Group 0 transmit CRC code register | (00FA$_{16}$) | 00$_{16}$ |
| | (00FB$_{16}$) | 00$_{16}$ |
| (138) Group 0 SI/O expansion mode register | (00FC$_{16}$) | 00$_{16}$ |
| (139) Group 0 SI/O expansion receive control register | (00FD$_{16}$) | 00$_{16}$ |
| (140) Group 0 SI/O special communication interrupt detect register | (00FE$_{16}$) | 00000XXX |
| (141) Group 0 SI/O expansion transmit control register | (00FF$_{16}$) | 00000XXX |
| (142) Group 1 time measurement/waveform generate register 0 | (0100$_{16}$) | ??$_{16}$ |
| | (0101$_{16}$) | ??$_{16}$ |
| (143) Group 1 time measurement/waveform generate register 1 | (0102$_{16}$) | ??$_{16}$ |
| | (0103$_{16}$) | ??$_{16}$ |
| (144) Group 1 time measurement/waveform generate register 2 | (0104$_{16}$) | ??$_{16}$ |
| | (0105$_{16}$) | ??$_{16}$ |
| (145) Group 1 time measurement/waveform generate register 3 | (0106$_{16}$) | ??$_{16}$ |
| | (0107$_{16}$) | ??$_{16}$ |
| (146) Group 1 time measurement/waveform generate register 4 | (0108$_{16}$) | ??$_{16}$ |
| | (0109$_{16}$) | ??$_{16}$ |
| (147) Group 1 time measurement/waveform generate register 5 | (010A$_{16}$) | ??$_{16}$ |
| | (010B$_{16}$) | ??$_{16}$ |
| (148) Group 1 time measurement/waveform generate register 6 | (010C$_{16}$) | ??$_{16}$ |
| | (010D$_{16}$) | ??$_{16}$ |
| (149) Group 1 time measurement/waveform generate register 7 | (010E$_{16}$) | ??$_{16}$ |
| | (010F$_{16}$) | ??$_{16}$ |
| (150) Group 1 waveform generate control register 0 | (0110$_{16}$) | 0X00X000 |
| (151) Group 1 waveform generate control register 1 | (0111$_{16}$) | 0X00X000 |
| (152) Group 1 waveform generate control register 2 | (0112$_{16}$) | 0X00X000 |
| (153) Group 1 waveform generate control register 3 | (0113$_{16}$) | 0X00X000 |
| (154) Group 1 waveform generate control register 4 | (0114$_{16}$) | 0X00X000 |
| (155) Group 1 waveform generate control register 5 | (0115$_{16}$) | 0X00X000 |
| (156) Group 1 waveform generate control register 6 | (0116$_{16}$) | 0X00X000 |
| (157) Group 1 waveform generate control register 7 | (0117$_{16}$) | 0X00X000 |
| (158) Group 1 time measurement control register 1 | (0119$_{16}$) | 00$_{16}$ |
| (159) Group 1 time measurement control register 2 | (011A$_{16}$) | 00$_{16}$ |
| (160) Group 1 time measurement control register 6 | (011E$_{16}$) | 00$_{16}$ |
| (161) Group 1 time measurement control register 7 | (011F$_{16}$) | 00$_{16}$ |
| (162) Group 1 base timer register | (0120$_{16}$) | ??$_{16}$ |
| | (0121$_{16}$) | ??$_{16}$ |
| (163) Group 1 base timer control register 0 | (0122$_{16}$) | 00$_{16}$ |
| (164) Group 1 base timer control register 1 | (0123$_{16}$) | 00$_{16}$ |

x : Nothing is mapped to this bit
? : Undefined

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

**Figure 1.4.3. Device's internal status after a reset is cleared (3/10)**

| | | | |
|---|---|---|---|
| (165) Group 1 time measurement prescaler register 6 | (0124₁₆) | 00₁₆ | |
| (166) Group 1 time measurement prescaler register 7 | (0125₁₆) | 00₁₆ | |
| (167) Group 1 function enable register | (0126₁₆) | 00₁₆ | |
| (168) Group 1 function select register | (0127₁₆) | 00₁₆ | |
| (169) Group 1 SI/O receive buffer register | (0128₁₆) | ??₁₆ | |
| | (0129₁₆) | XX0XXXXX | |
| (170) Group 1 transmit buffer/receive data register | (012A₁₆) | ??₁₆ | |
| (171) Group 1 receive input register | (012C₁₆) | ??₁₆ | |
| (172) Group 1 SI/O communication mode register | (012D₁₆) | 00₁₆ | |
| (173) Group 1 transmit output register | (012E₁₆) | ??₁₆ | |
| (174) Group 1 SI/O communication control register | (012F₁₆) | 0000X011 | |
| (175) Group 1 data compare register 0 | (0130₁₆) | ??₁₆ | |
| (176) Group 1 data compare register 1 | (0131₁₆) | ??₁₆ | |
| (177) Group 1 data compare register 2 | (0132₁₆) | ??₁₆ | |
| (178) Group 1 data compare register 3 | (0133₁₆) | ??₁₆ | |
| (179) Group 1 data mask register 0 | (0134₁₆) | ??₁₆ | |
| (180) Group 1 data mask register 1 | (0135₁₆) | ??₁₆ | |
| (181) Group 1 receive CRC code register | (0138₁₆) | ??₁₆ | |
| | (0139₁₆) | ??₁₆ | |
| (182) Group 1 transmit CRC code register | (013A₁₆) | 00₁₆ | |
| | (013B₁₆) | 00₁₆ | |
| (183) Group 1 SI/O expansion mode register | (013C₁₆) | 00₁₆ | |
| (184) Group 1 SI/O expansion receive control register | (013D₁₆) | 00₁₆ | |
| (185) Group 1 SI/O special communication interrupt detect register | (013E₁₆) | 00000XXX | |
| (186) Group 1 SI/O expansion transmit control register | (013F₁₆) | 00000XXX | |
| (187) Group 2 waveform generate register 0 | (0140₁₆) | ??₁₆ | |
| | (0141₁₆) | ??₁₆ | |
| (188) Group 2 waveform generate register 1 | (0142₁₆) | ??₁₆ | |
| | (0143₁₆) | ??₁₆ | |
| (189) Group 2 waveform generate register 2 | (0144₁₆) | ??₁₆ | |
| | (0145₁₆) | ??₁₆ | |
| (190) Group 2 waveform generate register 3 | (0146₁₆) | ??₁₆ | |
| | (0147₁₆) | ??₁₆ | |
| (191) Group 2 waveform generate register 4 | (0148₁₆) | ??₁₆ | |
| | (0149₁₆) | ??₁₆ | |
| (192) Group 2 waveform generate register 5 | (014A₁₆) | ??₁₆ | |
| | (014B₁₆) | ??₁₆ | |
| (193) Group 2 waveform generate register 6 | (014C₁₆) | ??₁₆ | |
| | (014D₁₆) | ??₁₆ | |
| (194) Group 2 waveform generate register 7 | (014E₁₆) | ??₁₆ | |
| | (014F₁₆) | ??₁₆ | |
| (195) Group 2 waveform generate control register 0 | (0150₁₆) | 00₁₆ | |
| (196) Group 2 waveform generate control register 1 | (0151₁₆) | 00₁₆ | |
| (197) Group 2 waveform generate control register 2 | (0152₁₆) | 00₁₆ | |
| (198) Group 2 waveform generate control register 3 | (0153₁₆) | 00₁₆ | |
| (199) Group 2 waveform generate control register 4 | (0154₁₆) | 00₁₆ | |
| (200) Group 2 waveform generate control register 5 | (0155₁₆) | 00₁₆ | |
| (201) Group 2 waveform generate control register 6 | (0156₁₆) | 00₁₆ | |
| (202) Group 2 waveform generate control register 7 | (0157₁₆) | 00₁₆ | |
| (203) Group 2 base timer register | (0160₁₆) | ??₁₆ | |
| | (0161₁₆) | ??₁₆ | |
| (204) Group 2 base timer control register 0 | (0162₁₆) | 00₁₆ | |
| (205) Group 2 base timer control register 1 | (0163₁₆) | 00₁₆ | |
| (206) Base timer start register | (0164₁₆) | XXXX0000 | |
| (207) Group 2 function enable register | (0166₁₆) | 00₁₆ | |
| (208) Group 2 RTP output buffer register | (0167₁₆) | 00₁₆ | |
| (209) Group 2 SI/O communication mode register | (016A₁₆) | 00XXX000 | |
| (210) Group 2 SI/O communication control register | (016B₁₆) | 0000X110 | |
| (211) Group 2 SI/O transmit buffer register | (016C₁₆) | ??₁₆ | |
| | (016D₁₆) | ???X???? | |
| (212) Group 2 SI/O receive buffer register | (016E₁₆) | ??₁₆ | |
| | (016F₁₆) | XXX?XXXX | |
| (213) Group 2 IEBus address register | (0170₁₆) | ??₁₆ | |
| | (0171₁₆) | XXXX???? | |
| (214) Group 2 IEBus control register | (0172₁₆) | 00XXX000 | |

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 1.4.3. Device's internal status after a reset is cleared (4/10)**

| | | | |
|---|---|---|---|
| (215) Group 2 IEBus transmit interrupt cause detect register | ($0173_{16}$) | xxx00000 | |
| (216) Group 2 IEBus receive interrupt cause detect register | ($0174_{16}$) | xxx00000 | |
| (217) Input function select register | ($0178_{16}$) | $00_{16}$ | |
| (218) Group 3 SI/O communication mode register | ($017A_{16}$) | 00XX0000 | |
| (219) Group 3 SI/O communication control register | ($017B_{16}$) | 00?0X??0 | |
| (220) Group 3 SI/O transmit buffer register | ($017C_{16}$) | $??_{16}$ | |
| | ($017D_{16}$) | $??_{16}$ | |
| (221) Group 3 SI/O receive buffer register | ($017E_{16}$) | $??_{16}$ | |
| | ($017F_{16}$) | $??_{16}$ | |
| (222) Group 3 waveform generate register 0 | ($0180_{16}$) | $??_{16}$ | |
| | ($0181_{16}$) | $??_{16}$ | |
| (223) Group 3 waveform generate register 1 | ($0182_{16}$) | $??_{16}$ | |
| | ($0183_{16}$) | $??_{16}$ | |
| (224) Group 3 waveform generate register 2 | ($0184_{16}$) | $??_{16}$ | |
| | ($0185_{16}$) | $??_{16}$ | |
| (225) Group 3 waveform generate register 3 | ($0186_{16}$) | $??_{16}$ | |
| | ($0187_{16}$) | $??_{16}$ | |
| (226) Group 3 waveform generate register 4 | ($0188_{16}$) | $??_{16}$ | |
| | ($0189_{16}$) | $??_{16}$ | |
| (227) Group 3 waveform generate register 5 | ($018A_{16}$) | $??_{16}$ | |
| | ($018B_{16}$) | $??_{16}$ | |
| (228) Group 3 waveform generate register 6 | ($018C_{16}$) | $??_{16}$ | |
| | ($018D_{16}$) | $??_{16}$ | |
| (229) Group 3 waveform generate register 7 | ($018E_{16}$) | $??_{16}$ | |
| | ($018F_{16}$) | $??_{16}$ | |
| (230) Group 3 waveform generate control register 0 | ($0190_{16}$) | $00_{16}$ | |
| (231) Group 3 waveform generate control register 1 | ($0191_{16}$) | $00_{16}$ | |
| (232) Group 3 waveform generate control register 2 | ($0192_{16}$) | $00_{16}$ | |
| (233) Group 3 waveform generate control register 3 | ($0193_{16}$) | $00_{16}$ | |
| (234) Group 3 waveform generate control register 4 | ($0194_{16}$) | $00_{16}$ | |
| (235) Group 3 waveform generate control register 5 | ($0195_{16}$) | $00_{16}$ | |
| (236) Group 3 waveform generate control register 6 | ($0196_{16}$) | $00_{16}$ | |
| (237) Group 3 waveform generate control register 7 | ($0197_{16}$) | $00_{16}$ | |
| (238) Group 3 waveform generate mask register 4 | ($0198_{16}$) | $??_{16}$ | |
| | ($0199_{16}$) | $??_{16}$ | |
| (239) Group 3 waveform generate mask register 5 | ($019A_{16}$) | $??_{16}$ | |
| | ($019B_{16}$) | $??_{16}$ | |
| (240) Group 3 waveform generate mask register 6 | ($019C_{16}$) | $??_{16}$ | |
| | ($019D_{16}$) | $??_{16}$ | |
| (241) Group 3 waveform generate mask register 7 | ($019E_{16}$) | $??_{16}$ | |
| | ($019F_{16}$) | $??_{16}$ | |
| (242) Group 3 base timer register | ($01A0_{16}$) | $??_{16}$ | |
| | ($01A1_{16}$) | $??_{16}$ | |
| (243) Group 3 base timer control register 0 | ($01A2_{16}$) | $00_{16}$ | |
| (244) Group 3 base timer control register 1 | ($01A3_{16}$) | 0X0X0000 | |
| (245) Group 3 function enable register | ($01A6_{16}$) | $00_{16}$ | |
| (246) Group 3 RTP output buffer register | ($01A7_{16}$) | $00_{16}$ | |
| (247) Group 3 high-speed HDLC communication control register 1 | ($01AB_{16}$) | 00XXXXX0 | |
| (248) Group 3 high-speed HDLC communication control register | ($01AC_{16}$) | $00_{16}$ | |
| (249) Group 3 high-speed HDLC communication register | ($01AD_{16}$) | $??_{16}$ | |
| (250) Group 3 high-speed HDLC transmit counter | ($01AE_{16}$) | $00_{16}$ | |
| | ($01AF_{16}$) | $00_{16}$ | |
| (251) Group 3 high-speed HDLC data compare register 0 | ($01B0_{16}$) | $00_{16}$ | |
| | ($01B1_{16}$) | $00_{16}$ | |
| (252) Group 3 high-speed HDLC data mask register 0 | ($01B2_{16}$) | $00_{16}$ | |
| | ($01B3_{16}$) | $00_{16}$ | |
| (253) Group 3 high-speed HDLC data compare register 1 | ($01B4_{16}$) | $00_{16}$ | |
| | ($01B5_{16}$) | $00_{16}$ | |
| (254) Group 3 high-speed HDLC data mask register 1 | ($01B6_{16}$) | $00_{16}$ | |
| | ($01B7_{16}$) | $00_{16}$ | |
| (255) Group 3 high-speed HDLC data compare register 2 | ($01B8_{16}$) | $00_{16}$ | |
| | ($01B9_{16}$) | $00_{16}$ | |
| (256) Group 3 high-speed HDLC data mask register 2 | ($01BA_{16}$) | $00_{16}$ | |
| | ($01BB_{16}$) | $00_{16}$ | |
| (257) Group 3 high-speed HDLC data compare register 3 | ($01BC_{16}$) | $00_{16}$ | |
| | ($01BD_{16}$) | $00_{16}$ | |

x : Nothing is mapped to this bit
? : Undefined
The content of other registers and RAM is undefined when the microcomputer is reset. The initial values must therefore be set.

**Figure 1.4.3. Device's internal status after a reset is cleared (5/10)**

| | | | | | |
|---|---|---|---|---|---|
| (258) Group 3 high-speed HDLC data mask register 3 | (01BE16) | 0016 | (282) CAN0 message slot buffer 0 data 6 | (01EC16) (Note) | ??16 |
| | (01BF16) | 0016 | (283) CAN0 message slot buffer 0 data 7 | (01ED16) (Note) | ??16 |
| (259) A-D1 register 0 | (01C016) | ??16 | (284) CAN0 message slot buffer 0 time stamp high | (01EE16) (Note) | ??16 |
| | (01C116) | ??16 | (285) CAN0 message slot buffer 0 time stamp low | (01EF16) (Note) | ??16 |
| (260) A-D1 register 1 | (01C216) | ??16 | (286) CAN1 message slot buffer 0 standard ID 0 | (01F016) (Note) | XXX????? |
| | (01C316) | ??16 | (287) CAN1 message slot buffer 0 standard ID 1 | (01F116) (Note) | XX?????? |
| (261) A-D1 register 2 | (01C416) | ??16 | (288) CAN1 message slot buffer 0 extended ID 0 | (01F216) (Note) | XXXX???? |
| | (01C516) | ??16 | (289) CAN1 message slot buffer 0 extended ID 1 | (01F316) (Note) | ??16 |
| (262) A-D1 register 3 | (01C616) | ??16 | (290) CAN1 message slot buffer 0 extended ID 2 | (01F416) (Note) | XX?????? |
| | (01C716) | ??16 | (291) CAN1 message slot buffer 0 data length code | (01F516) (Note) | XXXX???? |
| (263) A-D1 register 4 | (01C816) | ??16 | (292) CAN1 message slot buffer 0 data 0 | (01F616) (Note) | ??16 |
| | (01C916) | ??16 | (293) CAN1 message slot buffer 0 data 1 | (01F716) (Note) | ??16 |
| (264) A-D1 register 5 | (01CA16) | ??16 | (294) CAN1 message slot buffer 0 data 2 | (01F816) (Note) | ??16 |
| | (01CB16) | ??16 | (295) CAN1 message slot buffer 0 data 3 | (01F916) (Note) | ??16 |
| (265) A-D1 register 6 | (01CC16) | ??16 | (296) CAN1 message slot buffer 0 data 4 | (01FA16) (Note) | ??16 |
| | (01CD16) | ??16 | (297) CAN1 message slot buffer 0 data 5 | (01FB16) (Note) | ??16 |
| (266) A-D1 register 7 | (01CE16) | ??16 | (298) CAN1 message slot buffer 0 data 6 | (01FC16) (Note) | ??16 |
| | (01CF16) | ??16 | (299) CAN1 message slot buffer 0 data 7 | (01FD16) (Note) | ??16 |
| (267) A-D1 control register 2 | (01D416) | X00XX000 | (300) CAN1 message slot buffer 0 time stamp high | (01FE16) (Note) | ??16 |
| (268) A-D1 control register 0 | (01D616) | 0016 | (301) CAN1 message slot buffer 0 time stamp low | (01FF16) (Note) | ??16 |
| (269) A-D1 control register 1 | (01D716) | XX000000 | (302) CAN0 control register 0 | (020016) (Note) | XX010X01 |
| (270) CAN0 message slot buffer 0 standard ID 0 | (01E016) (Note) | XXX????? | | (020116) | XXXX0000 |
| (271) CAN0 message slot buffer 0 standard ID 1 | (01E116) (Note) | XX?????? | (303) CAN0 status register | (020216) (Note) | 0016 |
| (272) CAN0 message slot buffer 0 extended ID 0 | (01E216) (Note) | XXXX???? | | (020316) | X0000X01 |
| (273) CAN0 message slot buffer 0 extended ID 1 | (01E316) (Note) | ??16 | (304) CAN0 expansion ID register | (020416) (Note) | 0016 |
| (274) CAN0 message slot buffer 0 extended ID 2 | (01E416) (Note) | XX?????? | | (020516) | 0016 |
| (275) CAN0 message slot buffer 0 data length code | (01E516) (Note) | XXXX???? | (305) CAN0 configuration register | (020616) (Note) | 0000XXXX |
| (276) CAN0 message slot buffer 0 data 0 | (01E616) (Note) | ??16 | | (020716) | 0016 |
| (277) CAN0 message slot buffer 0 data 1 | (01E716) (Note) | ??16 | (306) CAN0 time stamp register | (020816) (Note) | 0016 |
| (278) CAN0 message slot buffer 0 data 2 | (01E816) (Note) | ??16 | | (020916) | 0016 |
| (279) CAN0 message slot buffer 0 data 3 | (01E916) (Note) | ??16 | (307) CAN0 transmit error count register | (020A16) (Note) | 0016 |
| (280) CAN0 message slot buffer 0 data 4 | (01EA16) (Note) | ??16 | (308) CAN0 receive error count register | (020B16) (Note) | 0016 |
| (281) CAN0 message slot buffer 0 data 5 | (01EB16) (Note) | ??16 | | | |

x : Nothing is mapped to this bit
? : Undefined

The content of other registers and RAM is undefined when the microcomputer is reset. The initial values must therefore be set.
Note: This applies when the CAN module is supplied with a clock by setting the sleep mode control bit (bit 0 at address 024216) to 1 after reset.

**Figure 1.4.3. Device's internal status after a reset is cleared (6/10)**

| | | | | | |
|---|---|---|---|---|---|
| (309) CAN0 slot interrupt status register | (020C16) (Note) | 0016 | (339) X0 register/Y0 register | (02C016) | ??16 |
| | (020D16) | 0016 | | (02C116) | ??16 |
| (310) CAN0 slot interrupt mask register | (021016) (Note) | 0016 | (340) X1 register/Y1 register | (02C216) | ??16 |
| | (021116) | 0016 | | (02C316) | ??16 |
| (311) CAN0 error interrupt mask register | (021416) (Note) | XXXXX000 | (341) X2 register/Y2 register | (02C416) | ??16 |
| (312) CAN0 error interrupt status register | (021516) (Note) | XXXXX000 | | (02C516) | ??16 |
| (313) CAN0 baud rate prescaler | (021716) (Note) | 0116 | (342) X3 register/Y3 register | (02C616) | ??16 |
| (314) CAN0 global mask register standard ID0 | (022816) (Note) | XXX00000 | | (02C716) | ??16 |
| (315) CAN0 global mask register standard ID1 | (022916) (Note) | XX000000 | (343) X4 register/Y4 register | (02C816) | ??16 |
| (316) CAN0 global mask register extended ID0 | (022A16) (Note) | ??16 | | (02C916) | ??16 |
| (317) CAN0 global mask register extended ID1 | (022B16) (Note) | ??16 | (344) X5 register/Y5 register | (02CA16) | ??16 |
| (318) CAN0 global mask register extended ID2 | (022C16) (Note) | ??16 | | (02CB16) | ??16 |
| (319) CAN0 message slot 0 control register / CAN0 local mask register A standard ID0 | (023016) (Note) | XXX00000 | (345) X6 register/Y6 register | (02CC16) | ??16 |
| (320) CAN0 message slot 1 control register / CAN0 local mask register A standard ID1 | (023116) (Note) | XX000000 | | (02CD16) | ??16 |
| (321) CAN0 message slot 2 control register / CAN0 local mask register A extended ID0 | (023216) (Note) | 0016 | (346) X7 register/Y7 register | (02CE16) | ??16 |
| (322) CAN0 message slot 3 control register / CAN0 local mask register A extended ID1 | (023316) (Note) | 0016 | | (02CF16) | ??16 |
| (323) CAN0 message slot 4 control register / CAN0 local mask register A extended ID2 | (023416) (Note) | 0016 | (347) X8 register/Y8 register | (02D016) | ??16 |
| (324) CAN0 message slot 5 control register | (023516) (Note) | 0016 | | (02D116) | ??16 |
| (325) CAN0 message slot 6 control register | (023616) (Note) | 0016 | (348) X9 register/Y9 register | (02D216) | ??16 |
| (326) CAN0 message slot 7 control register | (023716) (Note) | 0016 | | (02D316) | ??16 |
| (327) CAN0 message slot 8 control register / CAN0 local mask register B standard ID0 | (023816) (Note) | XXX00000 | (349) X10 register/Y10 register | (02D416) | ??16 |
| (328) CAN0 message slot 9 control register / CAN0 local mask register B standard ID1 | (023916) (Note) | XX000000 | | (02D516) | ??16 |
| (329) CAN0 message slot 10 control register / CAN0 local mask register B extended ID0 | (023A16) (Note) | 0016 | (350) X11 register/Y11 register | (02D616) | ??16 |
| (330) CAN0 message slot 11 control register / CAN0 local mask register B extended ID1 | (023B16) (Note) | 0016 | | (02D716) | ??16 |
| (331) CAN0 message slot 12 control register / CAN0 local mask register B extended ID2 | (023C16) (Note) | 0016 | (351) X12 register/Y12 register | (02D816) | ??16 |
| (332) CAN0 message slot 13 control register | (023D16) (Note) | 0016 | | (02D916) | ??16 |
| (333) CAN0 message slot 14 control register | (023E16) (Note) | 0016 | (352) X13 register/Y13 register | (02DA16) | ??16 |
| (334) CAN0 message slot 15 control register | (023F16) (Note) | 0016 | | (02DB16) | ??16 |
| (335) CAN0 slot buffer select register | (024016) (Note) | 0016 | (353) X14 register/Y14 register | (02DC16) | ??16 |
| (336) CAN0 control register 1 | (024116) (Note) | XX000XX | | (02DD16) | ??16 |
| (337) CAN0 sleep control register | (024216) (Note) | XXXXXXX0 | (354) X15 register/Y15 register | (02DE16) | ??16 |
| (338) CAN0 acceptance filter support register | (024416) (Note) | 0016 | | (02DF16) | ??16 |
| | (024516) | 0116 | (355) XY control register | (02E016) | XXXXXX00 |

x : Nothing is mapped to this bit
? : Undefined

The content of other registers and RAM is undefined when the microcomputer is reset. The initial values must therefore be set.
Note: This applies when the CAN module is supplied with a clock by setting the sleep mode control bit (bit 0 at address 024216) to 1 after reset.

**Figure 1.4.3. Device's internal status after a reset is cleared (7/10)**

| | | | | | |
|---|---|---|---|---|---|
| (356) UART1 special mode register 4 | (02E4$_{16}$) | 00$_{16}$ | (382) Three-phase output buffer register 0 | (030A$_{16}$) | XX000000 |
| (357) UART1 special mode register 3 | (02E5$_{16}$) | 00$_{16}$ | (383) Three-phase output buffer register 1 | (030B$_{16}$) | XX000000 |
| (358) UART1 special mode register 2 | (02E6$_{16}$) | 00$_{16}$ | (384) Dead time timer | (030C$_{16}$) | ??$_{16}$ |
| (359) UART1 special mode register | (02E7$_{16}$) | 00$_{16}$ | (385) Timer B2 interrupt occurrence frequency set counter | (030D$_{16}$) | XXXX???? |
| (360) UART1 transmit-receive mode register | (02E8$_{16}$) | 00$_{16}$ | (386) Timer B3 register | (0310$_{16}$) | ??$_{16}$ |
| (361) UART1 bit rate generator | (02E9$_{16}$) | ??$_{16}$ | | (0311$_{16}$) | ??$_{16}$ |
| (362) UART1 transmit buffer register | (02EA$_{16}$) | ??$_{16}$ | (387) Timer B4 register | (0312$_{16}$) | ??$_{16}$ |
| | (02EB$_{16}$) | XXXXXXXX | | (0313$_{16}$) | ??$_{16}$ |
| (363) UART1 transmit-receive control register 0 | (02EC$_{16}$) | 08$_{16}$ | (388) Timer B5 register | (0314$_{16}$) | ??$_{16}$ |
| (364) UART1 transmit-receive control register 1 | (02ED$_{16}$) | 02$_{16}$ | | (0315$_{16}$) | ??$_{16}$ |
| (365) UART1 receive buffer register | (02EE$_{16}$) | ??$_{16}$ | (389) Timer B3 mode register | (031B$_{16}$) | 00?X0000 |
| | (02EF$_{16}$) | ?????XX? | (390) Timer B4 mode register | (031C$_{16}$) | 00?X0000 |
| (366) UART4 special mode register 4 | (02F4$_{16}$) | 00$_{16}$ | (391) Timer B5 mode register | (031D$_{16}$) | 00?00000 |
| (367) UART4 special mode register 3 | (02F5$_{16}$) | 00$_{16}$ | (392) External interrupt cause select register | (031F$_{16}$) | 00$_{16}$ |
| (368) UART4 special mode register 2 | (02F6$_{16}$) | 00$_{16}$ | (393) UART3 special mode register 4 | (0324$_{16}$) | 00$_{16}$ |
| (369) UART4 special mode register | (02F7$_{16}$) | 00$_{16}$ | (394) UART3 special mode register 3 | (0325$_{16}$) | 00$_{16}$ |
| (370) UART4 transmit-receive mode register | (02F8$_{16}$) | 00$_{16}$ | (395) UART3 special mode register 2 | (0326$_{16}$) | 00$_{16}$ |
| (371) UART4 bit rate generator | (02F9$_{16}$) | ??$_{16}$ | (396) UART3 special mode register | (0327$_{16}$) | 00$_{16}$ |
| (372) UART4 transmit buffer register | (02FA$_{16}$) | ??$_{16}$ | (397) UART3 transmit-receive mode register | (0328$_{16}$) | 00$_{16}$ |
| | (02FB$_{16}$) | XXXXXXXX | (398) UART3 bit rate generator | (0329$_{16}$) | ??$_{16}$ |
| (373) UART4 transmit-receive control register 0 | (02FC$_{16}$) | 08$_{16}$ | (399) UART3 transmit buffer register | (032A$_{16}$) | ??$_{16}$ |
| (374) UART4 transmit-receive control register 1 | (02FD$_{16}$) | 02$_{16}$ | | (032B$_{16}$) | XXXXXXXX |
| (375) UART4 receive buffer register | (02FE$_{16}$) | ??$_{16}$ | (400) UART3 transmit-receive control register 0 | (032C$_{16}$) | 08$_{16}$ |
| | (02FF$_{16}$) | ?????XX? | (401) UART3 transmit-receive control register 1 | (032D$_{16}$) | 02$_{16}$ |
| (376) Timer B3,B4,B5 count start flag | (0300$_{16}$) | 000XXXXX | (402) UART3 receive buffer register | (032E$_{16}$) | ??$_{16}$ |
| (377) Timer A1-1 register | (0302$_{16}$) | ??$_{16}$ | | (032F$_{16}$) | ?????XX? |
| | (0303$_{16}$) | ??$_{16}$ | (403) UART2 special mode register 4 | (0334$_{16}$) | 00$_{16}$ |
| (378) Timer A2-1 register | (0304$_{16}$) | ??$_{16}$ | (404) UART2 special mode register 3 | (0335$_{16}$) | 00$_{16}$ |
| | (0305$_{16}$) | ??$_{16}$ | (405) UART2 special mode register 2 | (0336$_{16}$) | 00$_{16}$ |
| (379) Timer A4-1 register | (0306$_{16}$) | ??$_{16}$ | (406) UART2 special mode register | (0337$_{16}$) | 00$_{16}$ |
| | (0307$_{16}$) | ??$_{16}$ | (407) UART2 transmit-receive mode register | (0338$_{16}$) | 00$_{16}$ |
| (380) Three-phase PWM control register 0 | (0308$_{16}$) | 00$_{16}$ | (408) UART2 bit rate generator | (0339$_{16}$) | ??$_{16}$ |
| (381) Three-phase PWM control register 1 | (0309$_{16}$) | 00$_{16}$ | | | |

x : Nothing is mapped to this bit
? : Undefined

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

**Figure 1.4.3. Device's internal status after a reset is cleared (8/10)**

| | | | | | |
|---|---|---|---|---|---|
| (409) UART2 transmit buffer register | (033A16) | ??16 | (432) Timer B1 mode register | (035C16) | 00?X0000 |
| | (033B16) | XXXXXXX? | (433) Timer B2 mode register | (035D16) | 00?X0000 |
| (410) UART2 transmit/receive control register 0 | (033C16) | 0816 | (434) Timer B2 special mode register | (035E16) | XXXXXXX0 |
| (411) UART2 transmit/receive control register 1 | (033D16) | 0216 | (435) Count source prescaler register | (035F16) | 0XXX0000 |
| (412) UART2 receive buffer register | (033E16) | ??16 | (436) UART0 pecial mode register 4 | (036416) | 0016 |
| | (033F16) | ?????XX? | (437) UART0 special mode register 3 | (036516) | 0016 |
| (413) Count start flag | (034016) | 0016 | (438) UART0 special mode register 2 | (036616) | 0016 |
| (414) Clock prescaler reset flag | (034116) | 0XXXXXXX | (439) UART0 special mode register | (036716) | 0016 |
| (415) One-shot start flag | (034216) | 0016 | (440) UART0 transmit/receive mode register | (036816) | 0016 |
| (416) Trigger select register | (034316) | 0016 | (441) UART0 bit rate generator | (036916) | ??16 |
| (417) Up-down flag | (034416) | 0016 | (442) UART0 transmit buffer register | (036A16) | ??16 |
| (418) Timer A0 | (034616) | ??16 | | (036B16) | XXXXXXX? |
| | (034716) | ??16 | (443) UART0 transmit/receive control register 0 | (036C16) | 0816 |
| (419) Timer A1 | (034816) | ??16 | (444) UART0 transmit/receive control register 1 | (036D16) | 0216 |
| | (034916) | ??16 | (445) UART0 receive buffer register | (036E16) | ??16 |
| (420) Timer A2 | (034A16) | ??16 | | (036F16) | ?????XX? |
| | (034B16) | ??16 | (446) PLL control register 0 | (037616) | 01101100 |
| (421) Timer A3 | (034C16) | ??16 | (447) DMA0 cause select register | (037816) | 0X000000 |
| | (034D16) | ??16 | (448) DMA1 cause select register | (037916) | 0X000000 |
| (422) Timer A4 | (034E16) | ??16 | (449) DMA2 cause select register | (037A16) | 0X000000 |
| | (034F16) | ??16 | (450) DMA3 cause select register | (037B16) | 0X000000 |
| (423) Timer B0 | (035016) | ??16 | (451) CRC data register | (037C16) | ??16 |
| | (035116) | ??16 | | (037D16) | ??16 |
| (424) Timer B1 | (035216) | ??16 | (452) CRC input register | (037E16) | ??16 |
| | (035316) | ??16 | (453) A-D0 register 0 | (038016) | ??16 |
| (425) Timer B2 | (035416) | ??16 | | (038116) | ??16 |
| | (035516) | ??16 | (454) A-D0 register 1 | (038216) | ??16 |
| (426) Timer A0 mode register | (035616) | 00000X00 | | (038316) | ??16 |
| (427) Timer A1 mode register | (035716) | 00000X00 | (455) A-D0 register 2 | (038416) | ??16 |
| (428) Timer A2 mode register | (035816) | 00000X00 | | (038516) | ??16 |
| (429) Timer A3 mode register | (035916) | 00000X00 | (456) A-D0 register 3 | (038616) | ??16 |
| (430) Timer A4 mode register | (035A16) | 00000X00 | | (038716) | ??16 |
| (431) Timer B0 mode register | (035B16) | 00?00000 | | | |

x : Nothing is mapped to this bit
? : Undefined

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

**Figure 1.4.3. Device's internal status after a reset is cleared (9/10)**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| (457) A-D0 register 4 | | (0388$_{16}$) | ??$_{16}$ | (486) Port P9 | | (03C5$_{16}$) | ??$_{16}$ |
| | | (0389$_{16}$) | ??$_{16}$ | (487) Port P8 direction register | | (03C6$_{16}$) | 00X0000 |
| (458) A-D0 register 5 | | (038A$_{16}$) | ??$_{16}$ | (488) Port P9 direction register | | (03C7$_{16}$) | 00$_{16}$ |
| | | (038B$_{16}$) | ??$_{16}$ | (489) Port P10 | | (03C8$_{16}$) | ??$_{16}$ |
| (459) A-D0 register 6 | | (038C$_{16}$) | ??$_{16}$ | (490) Port P11 | (Note) | (03C9$_{16}$) | XXX????? |
| | | (038D$_{16}$) | ??$_{16}$ | (491) Port P10 direction register | (Note) | (03CA$_{16}$) | 00$_{16}$ |
| (460) A-D0 register 7 | | (038E$_{16}$) | ??$_{16}$ | (492) Port P11 direction register | (Note) | (03CB$_{16}$) | XXX0000 |
| | | (038F$_{16}$) | ??$_{16}$ | (493) Port P12 | (Note) | (03CC$_{16}$) | ??$_{16}$ |
| (461) A-D0 control register 2 | | (0394$_{16}$) | X0000000 | (494) Port P13 | (Note) | (03CD$_{16}$) | ??$_{16}$ |
| (462) A-D0 control register 0 | | (0396$_{16}$) | 00$_{16}$ | (495) Port P12 direction register | (Note) | (03CE16) | 00$_{16}$ |
| (463) A-D0 control register 1 | | (0397$_{16}$) | 00$_{16}$ | (496) Port P13 direction register | (Note) | (03CF$_{16}$) | 00$_{16}$ |
| (464) D-A register 0 | | (0398$_{16}$) | ??$_{16}$ | (497) Port P14 | (Note) | (03D0$_{16}$) | X?????? |
| (465) D-A register 1 | | (039A$_{16}$) | ??$_{16}$ | (498) Port P15 | (Note) | (03D1$_{16}$) | ??$_{16}$ |
| (466) D-A control register | | (039C$_{16}$) | XXXXXX00 | (499) Port P14 direction register | (Note) | (03D2$_{16}$) | X0000000 |
| (467) Function select register A8 | (Note) | (03A0$_{16}$) | XXXX0000 | (500) Port P15 direction register | (Note) | (03D3$_{16}$) | 00$_{16}$ |
| (468) Function select register A9 | (Note) | (03A1$_{16}$) | 00$_{16}$ | (501) Pull-up control register 2 | | (03DA$_{16}$) | 00$_{16}$ |
| (469) Function select register C | | (03AF$_{16}$) | 00X00000 | (502) Pull-up control register 3 | | (03DB$_{16}$) | 00$_{16}$ |
| (470) Function select register A0 | | (03B0$_{16}$) | 00$_{16}$ | (503) Pull-up control register 4 | (Note) | (03DC$_{16}$) | XXXX0000 |
| (471) Function select register A1 | | (03B1$_{16}$) | 00$_{16}$ | (504) Port P0 | | (03E0$_{16}$) | ??$_{16}$ |
| (472) Function select register B0 | | (03B2$_{16}$) | 00$_{16}$ | (505) Port P1 | | (03E1$_{16}$) | ??$_{16}$ |
| (473) Function select register B1 | | (03B3$_{16}$) | 00$_{16}$ | (506) Port P0 direction register | | (03E2$_{16}$) | 00$_{16}$ |
| (474) Function select register A2 | | (03B4$_{16}$) | XXXXX000 | (507) Port P1 direction register | | (03E3$_{16}$) | 00$_{16}$ |
| (475) Function select register A3 | | (03B5$_{16}$) | 00$_{16}$ | (508) Port P2 | | (03E4$_{16}$) | ??$_{16}$ |
| (476) Function select register B2 | | (03B6$_{16}$) | XXXXX000 | (509) Port P3 | | (03E5$_{16}$) | ??$_{16}$ |
| (477) Function select register B3 | | (03B7$_{16}$) | 00$_{16}$ | (510) Port P2 direction register | | (03E6$_{16}$) | 00$_{16}$ |
| (478) Function select register A5 | (Note) | (03B9$_{16}$) | XXXX0000 | (511) Port P3 direction register | | (03E7$_{16}$) | 00$_{16}$ |
| (479) Function select register A6 | (Note) | (03BC$_{16}$) | 00$_{16}$ | (512) Port P4 | | (03E8$_{16}$) | ??$_{16}$ |
| (480) Function select register A7 | (Note) | (03BD$_{16}$) | 00$_{16}$ | (513) Port P5 | | (03E9$_{16}$) | ??$_{16}$ |
| (481) Port P6 | | (03C0$_{16}$) | ??$_{16}$ | (514) Port P4 direction register | | (03EA$_{16}$) | 00$_{16}$ |
| (482) Port P7 | | (03C1$_{16}$) | ??$_{16}$ | (515) Port P5 direction register | | (03EB$_{16}$) | 00$_{16}$ |
| (483) Port P6 direction register | | (03C2$_{16}$) | 00$_{16}$ | (516) Pull-up control register 0 | | (03F0$_{16}$) | 00$_{16}$ |
| (484) Port P7 direction register | | (03C3$_{16}$) | 00$_{16}$ | (517) Pull-up control register 1 | | (03F1$_{16}$) | XXXX0000 |
| (485) Port P8 | | (03C4$_{16}$) | ??$_{16}$ | (518) Port control register | | (03FF$_{16}$) | XXXXXXX0 |

x : Nothing is mapped to this bit
? : Undefined

The content of other registers and RAM is undefined when the microcomputer is reset. The initial values must therefore be set.
Note :This register exists in 144-pin version.

**Figure 1.4.3. Device's internal status after a reset is cleared (10/10)**

# SFR

| Address | Register | |
|---|---|---|
| 0000₁₆ | | |
| 0001₁₆ | | |
| 0002₁₆ | | |
| 0003₁₆ | | |
| 0004₁₆ | Processor mode register 0 | PM0 |
| 0005₁₆ | Processor mode register 1 | PM1 |
| 0006₁₆ | System clock control register 0 | CM0 |
| 0007₁₆ | System clock control register 1 | CM1 |
| 0008₁₆ | Wait control register | WCR |
| 0009₁₆ | Address match interrupt control register | AIER |
| 000A₁₆ | Protect register | PRCR |
| 000B₁₆ | External data bus width control register | DS |
| 000C₁₆ | Main clock divided register | MCD |
| 000D₁₆ | Oscillation stop detect register | CM2 |
| 000E₁₆ | Watchdog timer start register | WDTS |
| 000F₁₆ | Watchdog timer control register | WDC |
| 0010₁₆ | | |
| 0011₁₆ | Address match interrupt register 0 | RMAD0 |
| 0012₁₆ | | |
| 0013₁₆ | | |
| 0014₁₆ | | |
| 0015₁₆ | Address match interrupt register 1 | RMAD1 |
| 0016₁₆ | | |
| 0017₁₆ | VDC control register for PLL | PLV |
| 0018₁₆ | | |
| 0019₁₆ | Address match interrupt register 2 | RMAD2 |
| 001A₁₆ | | |
| 001B₁₆ | VDC control register 0 | VDC0 * |
| 001C₁₆ | | |
| 001D₁₆ | Address match interrupt register 3 | RMAD3 |
| 001E₁₆ | | |
| 001F₁₆ | VDC control register 1 | VDC1 * |
| 0020₁₆ | | |
| 0021₁₆ | Emulator interrupt vector table register | EIAD0 * |
| 0022₁₆ | | |
| 0023₁₆ | Emulator interrupt detect register | EITD * |
| 0024₁₆ | Emulator protect register | EPRR * |
| 0025₁₆ | | |
| 0026₁₆ | | |
| 0027₁₆ | | |
| 0028₁₆ | | |
| 0029₁₆ | | |
| 002A₁₆ | | |
| 002B₁₆ | | |
| 002C₁₆ | | |
| 002D₁₆ | | |
| 002E₁₆ | | |
| 002F₁₆ | | |

| Address | Register | |
|---|---|---|
| 0030₁₆ | ROM area set register | ROA * |
| 0031₁₆ | Debug monitor area set register | DBA * |
| 0032₁₆ | Expansion area set register 0 | EXA0 * |
| 0033₁₆ | Expansion area set register 1 | EXA1 * |
| 0034₁₆ | Expansion area set register 2 | EXA2 * |
| 0035₁₆ | Expansion area set register 3 | EXA3 * |
| 0036₁₆ | | |
| 0037₁₆ | | |
| 0038₁₆ | | |
| 0039₁₆ | | |
| 003A₁₆ | | |
| 003B₁₆ | | |
| 003C₁₆ | | |
| 003D₁₆ | | |
| 003E₁₆ | | |
| 003F₁₆ | | |
| 0040₁₆ | DRAM control register | DRAMCONT |
| 0041₁₆ | DRAM refresh interval set register | REFCNT |
| 0042₁₆ | | |
| 0043₁₆ | | |
| 0044₁₆ | | |
| 0045₁₆ | | |
| 0046₁₆ | | |
| 0047₁₆ | | |
| 0048₁₆ | | |
| 0049₁₆ | | |
| 004A₁₆ | | |
| 004B₁₆ | | |
| 004C₁₆ | | |
| 004D₁₆ | | |
| 004E₁₆ | | |
| 004F₁₆ | | |
| 0050₁₆ | | |
| 0051₁₆ | | |
| 0052₁₆ | | |
| 0053₁₆ | | |
| 0054₁₆ | | |
| 0055₁₆ | Flash memory control register 2 | FMR2 * |
| 0056₁₆ | Flash memory control register 1 | FMR1 * |
| 0057₁₆ | Flash memory control register 0 | FMR0 |
| 0058₁₆ | | |
| 0059₁₆ | | |
| 005A₁₆ | | |
| 005B₁₆ | | |
| 005C₁₆ | | |
| 005D₁₆ | | |
| 005E₁₆ | | |
| 005F₁₆ | | |

The blank area is reserved and cannot be used by user.

*: User cannot use this. Do not access to the register.

| Address | Register | |
|---|---|---|
| 006016 | | |
| 006116 | | |
| 006216 | | |
| 006316 | | |
| 006416 | | |
| 006516 | | |
| 006616 | | |
| 006716 | | |
| 006816 | DMA0 interrupt control register | DM0IC |
| 006916 | Timer B5 interrupt control register | TB5IC |
| 006A16 | DMA2 interrupt control register | DM2IC |
| 006B16 | UART2 receive /ACK interrupt control register | S2RIC |
| 006C16 | Timer A0 interrupt control register | TA0IC |
| 006D16 | UART3 receive /ACK interrupt control register | S3RIC |
| 006E16 | Timer A2 interrupt control register | TA2IC |
| 006F16 | UART4 receive /ACK interrupt control register | S4RIC |
| 007016 | Timer A4 interrupt control register | TA4IC |
| 007116 | UART0/UART3 bus collision detection interrupt control register | BCN0IC |
| 007216 | UART0 receive/ACK interrupt control register | S0RIC |
| 007316 | A-D0 interrupt control register | AD0IC |
| 007416 | UART1 receive/ACK interrupt control register | S1RIC |
| 007516 | Intelligent I/O interrupt control register 0 | IIO0IC |
| 007616 | Timer B1 interrupt control register | TB1IC |
| 007716 | Intelligent I/O interrupt control register 2 | IIO2IC |
| 007816 | Timer B3 interrupt control register | TB3IC |
| 007916 | Intelligent I/O interrupt control register 4 | IIO4IC |
| 007A16 | INT5 interrupt control register | INT5IC |
| 007B16 | Intelligent I/O interrupt control register 6 | IIO6IC |
| 007C16 | INT3 interrupt control register | INT3IC |
| 007D16 | Intelligent I/O interrupt control register 8 | IIO8IC |
| 007E16 | INT1 interrupt control register | INT1IC |
| 007F16 | Intelligent I/O interrupt control register 10/ | IIO10IC |
| | CAN interrupt 1 control register | CAN1ICI |
| 008016 | | |
| 008116 | Intelligent I/O interrupt control register 11/ | IIO11IC |
| | CAN interrupt 2 control register | CAN2IC |
| 008216 | | |
| 008316 | | |
| 008416 | | |
| 008516 | | |
| 008616 | A-D1 interrupt control register | AD1IC |
| 008716 | | |
| 008816 | DMA1 interrupt control register | DM1IC |
| 008916 | UART2 transmit /NACK interrupt control register | S2TIC |
| 008A16 | DMA3 interrupt control register | DM3IC |
| 008B16 | UART3 transmit /NACK interrupt control register | S3TIC |
| 008C16 | Timer A1 interrupt control register | TA1IC |
| 008D16 | UART4 transmit /NACK interrupt control register | S4TIC |
| 008E16 | Timer A3 interrupt control register | TA3IC |
| 008F16 | UART2 bus collision detection interrupt control register | BCN2IC |

| Address | Register | |
|---|---|---|
| 009016 | UART0 transmit /NACK interrupt control register | S0TIC |
| 009116 | UART1/UART4 bus collision detection interrupt control register | BCN1IC |
| 009216 | UART1 transmit/NACK interruptcontrol register | S1TIC |
| 009316 | Key input interrupt control register | KUPIC |
| 009416 | Timer B0 interrupt control register | TB0IC |
| 009516 | Intelligent I/O interrupt control register 1 | IIO1IC |
| 009616 | Timer B2 interrupt control register | TB2IC |
| 009716 | Intelligent I/O interrupt control register 3 | IIO3IC |
| 009816 | Timer B4 interrupt control register | TB4IC |
| 009916 | Intelligent I/O interrupt control register 5 | IIO5IC |
| 009A16 | INT4 interrupt control register | INT4IC |
| 009B16 | Intelligent I/O interrupt control register 7 | IIO7IC |
| 009C16 | INT2 interrupt control register | INT2IC |
| 009D16 | Intelligent I/O interrupt control register 9/ | IIO9IC |
| | CAN interrupt 0 control register | CAN0ICI |
| 009E16 | INT0 interrupt control register | INT0IC |
| 009F16 | Exit priority register | RLVL |
| 00A016 | Interrupt request register 0 | IIO0IR |
| 00A116 | Interrupt request register 1 | IIO1IR |
| 00A216 | Interrupt request register 2 | IIO2IR |
| 00A316 | Interrupt request register 3 | IIO3IR |
| 00A416 | Interrupt request register 4 | IIO4IR |
| 00A516 | Interrupt request register 5 | IIO5IR |
| 00A616 | Interrupt request register 6 | IIO6IR |
| 00A716 | Interrupt request register 7 | IIO7IR |
| 00A816 | Interrupt request register 8 | IIO8IR |
| 00A916 | Interrupt request register 9 | IIO9IR |
| 00AA16 | Interrupt request register 10 | IIO10IR |
| 00AB16 | Interrupt request register 11 | IIO11IR |
| 00AC16 | | |
| 00AD16 | | |
| 00AE16 | | |
| 00AF16 | | |
| 00B016 | Interrupt enable register 0 | IIO0IE |
| 00B116 | Interrupt enable register 1 | IIO1IE |
| 00B216 | Interrupt enable register 2 | IIO2IE |
| 00B316 | Interrupt enable register 3 | IIO3IE |
| 00B416 | Interrupt enable register 4 | IIO4IE |
| 00B516 | Interrupt enable register 5 | IIO5IE |
| 00B616 | Interrupt enable register 6 | IIO6IE |
| 00B716 | Interrupt enable register 7 | IIO7IE |
| 00B816 | Interrupt enable register 8 | IIO8IE |
| 00B916 | Interrupt enable register 9 | IIO9IE |
| 00BA16 | Interrupt enable register 10 | IIO10IE |
| 00BB16 | Interrupt enable register 11 | IIO11IE |
| 00BC16 | | |
| 00BD16 | | |
| 00BE16 | | |
| 00BF16 | | |

The blank area is reserved and cannot be used by user.

| Address | Register | |
|---|---|---|
| 00C016<br>00C116 | Group 0 TM /WG register 0 | G0TM0/G0PO0 |
| 00C216<br>00C316 | Group 0 TM /WG register 1 | G0TM1/G0PO1 |
| 00C416<br>00C516 | Group 0 TM /WG register 2 | G0TM2/G0PO2 |
| 00C616<br>00C716 | Group 0 TM /WG register 3 | G0TM3/G0PO3 |
| 00C816<br>00C916 | Group 0 TM /WG register 4 | G0TM4/G0PO4 |
| 00CA16<br>00CB16 | Group 0 TM /WG register 5 | G0TM5/G0PO5 |
| 00CC16<br>00CD16 | Group 0 TM /WG register 6 | G0TM6/G0PO6 |
| 00CE16<br>00CF16 | Group 0 TM /WG register 7 | G0TM7/G0PO7 |
| 00D016 | Group 0 waveform generate control register 0 | G0POCR0 |
| 00D116 | Group 0 waveform generate control register 1 | G0POCR1 |
| 00D216 | | |
| 00D316 | | |
| 00D416 | Group 0 waveform generate control register 4 | G0POCR4 |
| 00D516 | Group 0 waveform generate control register 5 | G0POCR5 |
| 00D616 | | |
| 00D716 | | |
| 00D816 | Group 0 time measurement control register 0 | G0TMCR0 |
| 00D916 | Group 0 time measurement control register 1 | G0TMCR1 |
| 00DA16 | Group 0 time measurement control register 2 | G0TMCR2 |
| 00DB16 | Group 0 time measurement control register 3 | G0TMCR3 |
| 00DC16 | Group 0 time measurement control register 4 | G0TMCR4 |
| 00DD16 | Group 0 time measurement control register 5 | G0TMCR5 |
| 00DE16 | Group 0 time measurement control register 6 | G0TMCR6 |
| 00DF16 | Group 0 time measurement control register 7 | G0TMCR7 |
| 00E016<br>00E116 | Group 0 base timer register | G0BT |
| 00E216 | Group 0 base timer control register 0 | G0BCR0 |
| 00E316 | Group 0 base timer control register 1 | G0BCR1 |
| 00E416 | Group 0 time measurement prescaler register 6 | G0TPR6 |
| 00E516 | Group 0 time measurement prescaler register 7 | G0TPR7 |
| 00E616 | Group 0 function enable register | G0FE |
| 00E716 | Group 0 function select register | G0FS |
| 00E816<br>00E916 | Group 0 SI/O receive buffer register | G0RB |
| 00EA16 | Group 0 transmit buffer/receive data register | G0TB |
| 00EB16 | | |
| 00EC16 | Group 0 receive input register | G0RI |
| 00ED16 | Group 0 SI/O communication mode register | G0MR |
| 00EE16 | Group 0 transmit output register | G0TO |
| 00EF16 | Group 0 SI/O communication control register | G0CR |

| Address | Register | |
|---|---|---|
| 00F016 | Group 0 data compare register 0 | G0CMP0 |
| 00F116 | Group 0 data compare register 1 | G0CMP1 |
| 00F216 | Group 0 data compare register 2 | G0CMP2 |
| 00F316 | Group 0 data compare register 3 | G0CMP3 |
| 00F416 | Group 0 data mask register 0 | G0MSK0 |
| 00F516 | Group 0 data mask register 1 | G0MSK1 |
| 00F616 | | |
| 00F716 | | |
| 00F816<br>00F916 | Group 0 receive CRC code register | G0RCRC |
| 00FA16<br>00FB16 | Group 0 transmit CRC code register | G0TCRC |
| 00FC16 | Group 0 SI/O expansion mode register | G0EMR |
| 00FD16 | Group 0 SI/O expansion receive control register | G0ERC |
| 00FE16 | Group 0 SI/O special communication interrupt detect register | G0IRF |
| 00FF16 | Group 0 SI/O expansion transmit control register | G0ETC |
| 010016<br>010116 | Group 1 TM /WG register 0 | G1TM0/G1PO0 |
| 010216<br>010316 | Group 1 TM /WG register 1 | G1TM1/G1PO1 |
| 010416<br>010516 | Group 1 TM /WG register 2 | G1TM2/G1PO2 |
| 010616<br>010716 | Group 1 TM /WG register 3 | G1TM3/G1PO3 |
| 010816<br>010916 | Group 1 TM /WG register 4 | G1TM4/G1PO4 |
| 010A16<br>010B16 | Group 1 TM /WG register 5 | G1TM5/G1PO5 |
| 010C16<br>010D16 | Group 1 TM /WG register 6 | G1TM6/G1PO6 |
| 010E16<br>010F16 | Group 1 TM /WG register 7 | G1TM7/G1PO7 |
| 011016 | Group 1 waveform generate control register 0 | G1POCR0 |
| 011116 | Group 1 waveform generate control register 1 | G1POCR1 |
| 011216 | Group 1 waveform generate control register 2 | G1POCR2 |
| 011316 | Group 1 waveform generate control register 3 | G1POCR3 |
| 011416 | Group 1 waveform generate control register 4 | G1POCR4 |
| 011516 | Group 1 waveform generate control register 5 | G1POCR5 |
| 011616 | Group 1 waveform generate control register 6 | G1POCR6 |
| 011716 | Group 1 waveform generate control register 7 | G1POCR7 |
| 011816 | | |
| 011916 | Group 1 time measurement control register 1 | G1TMCR1 |
| 011A16 | Group 1 time measurement control register 2 | G1TMCR2 |
| 011B16 | | |
| 011C16 | | |
| 011D16 | | |
| 011E16 | Group 1 time measurement control register 6 | G1TMCR6 |
| 011F16 | Group 1 time measurement control register 7 | G1TMCR7 |

The blank area is reserved and cannot be used by user.

| Address | Register | |
|---|---|---|
| 0120₁₆ 0121₁₆ | Group 1 base timer register | G1BT |
| 0122₁₆ | Group 1 base timer control register 0 | G1BCR0 |
| 0123₁₆ | Group 1 base timer control register 1 | G1BCR1 |
| 0124₁₆ | Group 1 time measurement prescaler register 6 | G1TPR6 |
| 0125₁₆ | Group 1 time measurement prescaler register 7 | G1TPR7 |
| 0126₁₆ | Group 1 function enable register | G1FE |
| 0127₁₆ | Group 1 function select register | G1FS |
| 0128₁₆ 0129₁₆ | Group 1 SI/O receive buffer register | G1RB |
| 012A₁₆ | Group 1 transmit buffer/receive data register | G1TB |
| 012B₁₆ | | |
| 012C₁₆ | Group 1 receive input register | G1RI |
| 012D₁₆ | Group 1 SI/O communication mode register | G1MR |
| 012E₁₆ | Group 1 transmit output register | G1TO |
| 012F₁₆ | Group 1 SI/O communication control register | G1CR |
| 0130₁₆ | Group 1 data compare register 0 | G1CMP0 |
| 0131₁₆ | Group 1 data compare register 1 | G1CMP1 |
| 0132₁₆ | Group 1 data compare register 2 | G1CMP2 |
| 0133₁₆ | Group 1 data compare register 3 | G1CMP3 |
| 0134₁₆ | Group 1 data mask register 0 | G1MSK0 |
| 0135₁₆ | Group 1 data mask register 1 | G1MSK1 |
| 0136₁₆ | | |
| 0137₁₆ | | |
| 0138₁₆ 0139₁₆ | Group 1 receive CRC code register | G1RCRC |
| 013A₁₆ 013B₁₆ | Group 1 transmit CRC code register | G1TCRC |
| 013C₁₆ | Group 1 SI/O expansion mode register | G1EMR |
| 013D₁₆ | Group 1 SI/O expansion receive control register | G1ERC |
| 013E₁₆ | Group 1 SI/O special communication interrupt detect register | G1IRF |
| 013F₁₆ | Group 1 SI/O expansion transmit control register | G1ETC |
| 0140₁₆ 0141₁₆ | Group 2 waveform generate register 0 | G2PO0 |
| 0142₁₆ 0143₁₆ | Group 2 waveform generate register 1 | G2PO1 |
| 0144₁₆ 0145₁₆ | Group 2 waveform generate register 2 | G2PO2 |
| 0146₁₆ 0147₁₆ | Group 2 waveform generate register 3 | G2PO3 |
| 0148₁₆ 0149₁₆ | Group 2 waveform generate register 4 | G2PO4 |
| 014A₁₆ 014B₁₆ | Group 2 waveform generate register 5 | G2PO5 |
| 014C₁₆ 014D₁₆ | Group 2 waveform generate register 6 | G2PO6 |
| 014E₁₆ 014F₁₆ | Group 2 waveform generate register 7 | G2PO7 |

| Address | Register | |
|---|---|---|
| 0150₁₆ | Group 2 waveform generate control register 0 | G2POCR0 |
| 0151₁₆ | Group 2 waveform generate control register 1 | G2POCR1 |
| 0152₁₆ | Group 2 waveform generate control register 2 | G2POCR2 |
| 0153₁₆ | Group 2 waveform generate control register 3 | G2POCR3 |
| 0154₁₆ | Group 2 waveform generate control register 4 | G2POCR4 |
| 0155₁₆ | Group 2 waveform generate control register 5 | G2POCR5 |
| 0156₁₆ | Group 2 waveform generate control register 6 | G2POCR6 |
| 0157₁₆ | Group 2 waveform generate control register 7 | G2POCR7 |
| 0158₁₆ | | |
| 0159₁₆ | | |
| 015A₁₆ | | |
| 015B₁₆ | | |
| 015C₁₆ | | |
| 015D₁₆ | | |
| 015E₁₆ | | |
| 015F₁₆ | | |
| 0160₁₆ 0161₁₆ | Group 2 base timer register | G2BT |
| 0162₁₆ | Group 2 base timer control register 0 | G2BCR0 |
| 0163₁₆ | Group 2 base timer control register 1 | G2BCR1 |
| 0164₁₆ | Base timer start register | BTSR |
| 0165₁₆ | | |
| 0166₁₆ | Group 2 function enable register | G2FE |
| 0167₁₆ | Group 2 RTP output buffer register | G2RTP |
| 0168₁₆ | | |
| 0169₁₆ | | |
| 016A₁₆ | Group 2 SI/O communication mode register | G2MR |
| 016B₁₆ | Group 2 SI/O communication control register | G2CR |
| 016C₁₆ 016D₁₆ | Group 2 SI/O transmit buffer register | G2TB |
| 016E₁₆ 016F₁₆ | Group 2 SI/O receive buffer register | G2RB |
| 0170₁₆ 0171₁₆ | Group 2 IEBus address register | IEAR |
| 0172₁₆ | Group 2 IEBus control register | IECR |
| 0173₁₆ | Group 2 IEBus transmit interrupt cause detect register | IETIF |
| 0174₁₆ | Group 2 IEBus receive interrupt cause detect register | IERIF |
| 0175₁₆ | | |
| 0176₁₆ | | |
| 0177₁₆ | | |
| 0178₁₆ | Input function select register | IPS |
| 0179₁₆ | | |
| 017A₁₆ | Group 3 SI/O communication mode register | G3MR |
| 017B₁₆ | Group 3 SI/O communication control register | G3CR |
| 017C₁₆ 017D₁₆ | Group 3 SI/O transmit buffer register | G3TB |
| 017E₁₆ 017F₁₆ | Group 3 SI/O receive buffer register | G3RB |

The blank area is reserved and cannot be used by user.

| Address | Register | |
|---|---|---|
| 0180₁₆ 0181₁₆ | Group 3 waveform generate register 0 | G3PO0 |
| 0182₁₆ 0183₁₆ | Group 3 waveform generate register 1 | G3PO1 |
| 0184₁₆ 0185₁₆ | Group 3 waveform generate register 2 | G3PO2 |
| 0186₁₆ 0187₁₆ | Group 3 waveform generate register 3 | G3PO3 |
| 0188₁₆ 0189₁₆ | Group 3 waveform generate register 4 | G3PO4 |
| 018A₁₆ 018B₁₆ | Group 3 waveform generate register 5 | G3PO5 |
| 018C₁₆ 018D₁₆ | Group 3 waveform generate register 6 | G3PO6 |
| 018E₁₆ 018F₁₆ | Group 3 waveform generate register 7 | G3PO7 |
| 0190₁₆ | Group 3 waveform generate control register 0 | G3POCR0 |
| 0191₁₆ | Group 3 waveform generate control register 1 | G3POCR1 |
| 0192₁₆ | Group 3 waveform generate control register 2 | G3POCR2 |
| 0193₁₆ | Group 3 waveform generate control register 3 | G3POCR3 |
| 0194₁₆ | Group 3 waveform generate control register 4 | G3POCR4 |
| 0195₁₆ | Group 3 waveform generate control register 5 | G3POCR5 |
| 0196₁₆ | Group 3 waveform generate control register 6 | G3POCR6 |
| 0197₁₆ | Group 3 waveform generate control register 7 | G3POCR7 |
| 0198₁₆ 0199₁₆ | Group 3 waveform generate mask register 4 | G3MK4 |
| 019A₁₆ 019B₁₆ | Group 3 waveform generate mask register 5 | G3MK5 |
| 019C₁₆ 019D₁₆ | Group 3 waveform generate mask register 6 | G3MK6 |
| 019E₁₆ 019F₁₆ | Group 3 waveform generate mask register 7 | G3MK7 |
| 01A0₁₆ 01A1₁₆ | Group 3 base timer register | G3BT |
| 01A2₁₆ | Group 3 base timer control register 0 | G3BCR0 |
| 01A3₁₆ | Group 3 base timer control register 1 | G3BCR1 |
| 01A4₁₆ | | |
| 01A5₁₆ | | |
| 01A6₁₆ | Group 3 function enable register | G3FE |
| 01A7₁₆ | Group 3 RTP output buffer register | G3RTP |
| 01A8₁₆ | | |
| 01A9₁₆ | | |
| 01AA₁₆ | | |
| 01AB₁₆ | Group 3 high-speed HDLC communication control register 1 | HDLC1 |
| 01AC₁₆ | Group 3 high-speed HDLC communication control register | HDLC |
| 01AD₁₆ | Group 3 high-speed HDLC communication register | HDLCF |
| 01AE₁₆ 01AF₁₆ | Group 3 high-speed HDLC transmit counter | HDLCC |

| Address | Register | |
|---|---|---|
| 01B0₁₆ 01B1₁₆ | Group 3 high-speed HDLC data compare register 0 | HDLCCP0 |
| 01B2₁₆ 01B3₁₆ | Group 3 high-speed HDLC data mask register 0 | HDLCMK0 |
| 01B4₁₆ 01B5₁₆ | Group 3 high-speed HDLC data compare register1 | HDLCCP1 |
| 01B6₁₆ 01B7₁₆ | Group 3 high-speed HDLC data mask register 1 | HDLCMK1 |
| 01B8₁₆ 01B9₁₆ | Group 3 high-speed HDLC data compare register 2 | HDLCCP2 |
| 01BA₁₆ 01BB₁₆ | Group 3 high-speed HDLC data mask register 2 | HDLCMK2 |
| 01BC₁₆ 01BD₁₆ | Group 3 high-speed HDLC data compare register 3 | HDLCCP3 |
| 01BE₁₆ 01BF₁₆ | Group 3 high-speed HDLC data mask register 3 | HDLCMK3 |
| 01C0₁₆ 01C1₁₆ | A-D1 register 0 | AD10 |
| 01C2₁₆ 01C3₁₆ | A-D1 register 1 | AD11 |
| 01C4₁₆ 01C5₁₆ | A-D1 register 2 | AD12 |
| 01C6₁₆ 01C7₁₆ | A-D1 register 3 | AD13 |
| 01C8₁₆ 01C9₁₆ | A-D1 register 4 | AD14 |
| 01CA₁₆ 01CB₁₆ | A-D1 register 5 | AD15 |
| 01CC₁₆ 01CD₁₆ | A-D1 register 6 | AD16 |
| 01CE₁₆ 01CF₁₆ | A-D1 register 7 | AD17 |
| 01D0₁₆ | | |
| 01D1₁₆ | | |
| 01D2₁₆ | | |
| 01D3₁₆ | | |
| 01D4₁₆ | A-D1 control register 2 | AD1CON2 |
| 01D5₁₆ | | |
| 01D6₁₆ | A-D1 control register 0 | AD1CON0 |
| 01D7₁₆ | A-D1 control register 1 | AD1CON1 |
| 01D8₁₆ | | |
| 01D9₁₆ | | |
| 01DA₁₆ | | |
| 01DB₁₆ | | |
| 01DC₁₆ | | |
| 01DD₁₆ | | |
| 01DE₁₆ | | |
| 01DF₁₆ | | |

The blank area is reserved and cannot be used by user.

| Address | Register | |
|---|---|---|
| 01E0₁₆ | CAN0 message slot buffer 0 standard ID0 | C0SLOT0_0 |
| 01E1₁₆ | CAN0 message slot buffer 0 standard ID1 | C0SLOT0_1 |
| 01E2₁₆ | CAN0 message slot buffer 0 extend ID0 | C0SLOT0_2 |
| 01E3₁₆ | CAN0 message slot buffer 0 extend ID1 | C0SLOT0_3 |
| 01E4₁₆ | CAN0 message slot buffer 0 extend ID2 | C0SLOT0_4 |
| 01E5₁₆ | CAN0 message slot buffer 0 data length code | C0SLOT0_5 |
| 01E6₁₆ | CAN0 message slot buffer 0 data 0 | C0SLOT0_6 |
| 01E7₁₆ | CAN0 message slot buffer 0 data 1 | C0SLOT0_7 |
| 01E8₁₆ | CAN0 message slot buffer 0 data 2 | C0SLOT0_8 |
| 01E9₁₆ | CAN0 message slot buffer 0 data 3 | C0SLOT0_9 |
| 01EA₁₆ | CAN0 message slot buffer 0 data 4 | C0SLOT0_10 |
| 01EB₁₆ | CAN0 message slot buffer 0 data 5 | C0SLOT0_11 |
| 01EC₁₆ | CAN0 message slot buffer 0 data 6 | C0SLOT0_12 |
| 01ED₁₆ | CAN0 message slot buffer 0 data 7 | C0SLOT0_13 |
| 01EE₁₆ | CAN0 message slot buffer 0 time stamp high | C0SLOT0_14 |
| 01EF₁₆ | CAN0 message slot buffer 0 time stamp low | C0SLOT0_15 |
| 01F0₁₆ | CAN0 message slot buffer 1 standard ID0 | C0SLOT1_0 |
| 01F1₁₆ | CAN0 message slot buffer 1 standard ID1 | C0SLOT1_1 |
| 01F2₁₆ | CAN0 message slot buffer 1 extend ID0 | C0SLOT1_2 |
| 01F3₁₆ | CAN0 message slot buffer 1 extend ID1 | C0SLOT1_3 |
| 01F4₁₆ | CAN0 message slot buffer 1 extend ID2 | C0SLOT1_4 |
| 01F5₁₆ | CAN0 message slot buffer 1 data length code | C0SLOT1_5 |
| 01F6₁₆ | CAN0 message slot buffer 1 data 0 | C0SLOT1_6 |
| 01F7₁₆ | CAN0 message slot buffer 1 data 1 | C0SLOT1_7 |
| 01F8₁₆ | CAN0 message slot buffer 1 data 2 | C0SLOT1_8 |
| 01F9₁₆ | CAN0 message slot buffer 1 data 3 | C0SLOT1_9 |
| 01FA₁₆ | CAN0 message slot buffer 1 data 4 | C0SLOT1_10 |
| 01FB₁₆ | CAN0 message slot buffer 1 data 5 | C0SLOT1_11 |
| 01FC₁₆ | CAN0 message slot buffer 1 data 6 | C0SLOT1_12 |
| 01FD₁₆ | CAN0 message slot buffer 1 data 7 | C0SLOT1_13 |
| 01FE₁₆ | CAN0 message slot buffer 1 time stamp high | C0SLOT1_14 |
| 01FF₁₆ | CAN0 message slot buffer 1 time stamp low | C0SLOT1_15 |
| 0200₁₆<br>0201₁₆ | CAN0 control register 0 | C0CTLR0 |
| 0202₁₆<br>0203₁₆ | CAN0 status register | C0STR |
| 0204₁₆<br>0205₁₆ | CAN0 expansion ID register | C0IDR |
| 0206₁₆<br>0207₁₆ | CAN0 configuration register | C0CONR |
| 0208₁₆<br>0209₁₆ | CAN0 time stamp register | C0TSR |
| 020A₁₆ | CAN0 transmit error count register | C0TEC |
| 020B₁₆ | CAN0 receive error count register | C0REC |
| 020C₁₆<br>020D₁₆ | CAN0 slot interrupt status register | C0SISTR |
| 020E₁₆ | | |
| 020F₁₆ | | |

| Address | Register | |
|---|---|---|
| 0210₁₆<br>0211₁₆ | CAN0 slot interrupt mask register | C0SIMKR |
| 0212₁₆ | | |
| 0213₁₆ | | |
| 0214₁₆ | CAN0 error interrupt mask register | C0EIMKR |
| 0215₁₆ | CAN0 error interrupt status register | C0EISTR |
| 0216₁₆ | | |
| 0217₁₆ | CAN0 baud rate prescaler | C0BPR |
| 0218₁₆ | | |
| 0219₁₆ | | |
| 021A₁₆ | | |
| 021B₁₆ | | |
| 021C₁₆ | | |
| 021D₁₆ | | |
| 021E₁₆ | | |
| 021F₁₆ | | |
| 0220₁₆ | | |
| 0221₁₆ | | |
| 0222₁₆ | | |
| 0223₁₆ | | |
| 0224₁₆ | | |
| 0225₁₆ | | |
| 0226₁₆ | | |
| 0227₁₆ | | |
| 0228₁₆ | CAN0 global mask register standard ID0 | C0GMR0 |
| 0229₁₆ | CAN0 global mask register standard ID1 | C0GMR1 |
| 022A₁₆ | CAN0 global mask register extend ID0 | C0GMR2 |
| 022B₁₆ | CAN0 global mask register extend ID1 | C0GMR3 |
| 022C₁₆ | CAN0 global mask register extend ID2 | C0GMR4 |
| 022D₁₆ | | |
| 022E₁₆ | | |
| 022F₁₆ | | |
| 0230₁₆ | CAN0 message slot 0 control register / CAN0 local mask register A standard ID0 | C0MCTL0/ C0LMAR0 |
| 0231₁₆ | CAN0 message slot 1 control register / CAN0 local mask register A standard ID1 | C0MCTL1/ C0LMAR1 |
| 0232₁₆ | CAN0 message slot 2 control register / CAN0 local mask register A extend ID0 | C0MCTL2/ C0LMAR2 |
| 0233₁₆ | CAN0 message slot 3 control register / CAN0 local mask register A extend ID1 | C0MCTL3/ C0LMAR3 |
| 0234₁₆ | CAN0 message slot 4 control register / CAN0 local mask register A extend ID2 | C0MCTL4/ C0LMAR4 |
| 0235₁₆ | CAN0 message slot 5 control register | C0MCTL5 |
| 0236₁₆ | CAN0 message slot 6 control register | C0MCTL6 |
| 0237₁₆ | CAN0 message slot 7 control register | C0MCTL7 |
| 0238₁₆ | CAN0 message slot 8 control register / CAN0 local mask register B standard ID0 | C0MCTL8/ C0LMBR0 |

The blank area is reserved and cannot be used by user.

Note 1: CAN0 message slot i control registers (i=0 to 15) are allocated to addresses 0230₁₆ to 023F₁₆ by switching banks.

| Address | Register | |
|---|---|---|
| 0239₁₆ | CAN0 message slot 9 control register / CAN0 local mask register B standard ID1 | C0MCTL9/ C0LMBR1 |
| 023A₁₆ | CAN0 message slot 10 control register / CAN0 local mask register B extend ID0 | C0MCTL10/ C0LMBR2 |
| 023B₁₆ | CAN0 message slot 11 control register / CAN0 local mask register B extend ID1 | C0MCTL11/ C0LMBR3 |
| 023C₁₆ | CAN0 message slot 12 control register / CAN0 local mask register B extend ID2 | C0MCTL12/ C0LMBR4 |
| 023D₁₆ | CAN0 message slot 13 control register | C0MCTL13 |
| 023E₁₆ | CAN0 message slot 14 control register | C0MCTL14 |
| 023F₁₆ | CAN0 message slot 15 control register | C0MCTL15 |
| 0240₁₆ | CAN0 slot buffer select register | C0SBS |
| 0241₁₆ | CAN0 control register 1 | C0CTLR1 |
| 0242₁₆ | CAN0 sleep control register | C0SLPR |
| 0243₁₆ | | |
| 0244₁₆ 0245₁₆ | CAN0 acceptance filter support register | C0AFS |

| Address | Register | |
|---|---|---|
| 02C0₁₆ 02C1₁₆ | X0 register/Y0 register | X0R/Y0R |
| 02C2₁₆ 02C3₁₆ | X1 register/Y1 register | X1R/Y1R |
| 02C4₁₆ 02C5₁₆ | X2 register/Y2 register | X2R/Y2R |
| 02C6₁₆ 02C7₁₆ | X3 register/Y3 register | X3R/Y3R |
| 02C8₁₆ 02C9₁₆ | X4 register/Y4 register | X4R/Y4R |
| 02CA₁₆ 02CB₁₆ | X5 register/Y5 register | X5R/Y5R |
| 02CC₁₆ 02CD₁₆ | X6 register/Y6 register | X6R/Y6R |
| 02CE₁₆ 02CF₁₆ | X7 register/Y7 register | X7R/Y7R |
| 02D0₁₆ 02D1₁₆ | X8 register/Y8 register | X8R/Y8R |
| 02D2₁₆ 02D3₁₆ | X9 register/Y9 register | X9R/Y9R |
| 02D4₁₆ 02D5₁₆ | X10 register/Y10 register | X10R/Y10R |
| 02D6₁₆ 02D7₁₆ | X11 register/Y11 register | X11R/Y11R |
| 02D8₁₆ 02D9₁₆ | X12 register/Y12 register | X12R/Y12R |
| 02DA₁₆ 02DB₁₆ | X13 register/Y13 register | X13R/Y13R |
| 02DC₁₆ 02DD₁₆ | X14 register/Y14 register | X14R/Y14R |
| 02DE₁₆ 02DF₁₆ | X15 register/Y15 register | X15R/Y15R |
| 02E0₁₆ | XY control register | XYC |
| 02E1₁₆ | | |
| 02E2₁₆ | | |
| 02E3₁₆ | | |
| 02E4₁₆ | UART1 special mode register 4 | U1SMR4 |
| 02E5₁₆ | UART1 special mode register 3 | U1SMR3 |
| 02E6₁₆ | UART1 special mode register 2 | U1SMR2 |
| 02E7₁₆ | UART1 special mode register | U1SMR |
| 02E8₁₆ | UART1 transmit-receive mode register | U1MR |
| 02E9₁₆ | UART1 bit rate generator | U1BRG |
| 02EA₁₆ 02EB₁₆ | UART1 transmit buffer register | U1TB |
| 02EC₁₆ | UART1 transmit-receive control register 0 | U1C0 |
| 02ED₁₆ | UART1 transmit-receive control register 1 | U1C1 |
| 02EE₁₆ 02EF₁₆ | UART1 receive buffer register | U1RB |

The blank area is reserved and cannot be used by user.

| Address | Register | |
|---|---|---|
| 02F0₁₆ | | |
| 02F1₁₆ | | |
| 02F2₁₆ | | |
| 02F3₁₆ | | |
| 02F4₁₆ | UART4 special mode register 4 | U4SMR4 |
| 02F5₁₆ | UART4 special mode register 3 | U4SMR3 |
| 02F6₁₆ | UART4 special mode register 2 | U4SMR2 |
| 02F7₁₆ | UART4 special mode register | U4SMR |
| 02F8₁₆ | UART4 transmit-receive mode register | U4MR |
| 02F9₁₆ | UART4 bit rate generator | U4BRG |
| 02FA₁₆<br>02FB₁₆ | UART4 transmit buffer register | U4TB |
| 02FC₁₆ | UART4 transmit-receive control register 0 | U4C0 |
| 02FD₁₆ | UART4 transmit-receive control register 1 | U4C1 |
| 02FE₁₆<br>02FF₁₆ | UART4 receive buffer register | U4RB |
| 0300₁₆ | Timer B3,B4,B5 count start flag | TBSR |
| 0301₁₆ | | |
| 0302₁₆<br>0303₁₆ | Timer A1-1 register | TA11 |
| 0304₁₆<br>0305₁₆ | Timer A2-1 register | TA21 |
| 0306₁₆<br>0307₁₆ | Timer A4-1 register | TA41 |
| 0308₁₆ | Three-phase PWM control register 0 | INVC0 |
| 0309₁₆ | Three-phase PWM control register 1 | INVC1 |
| 030A₁₆ | Three-phase output buffer register 0 | IDB0 |
| 030B₁₆ | Three-phase output buffer register 1 | IDB1 |
| 030C₁₆ | Dead time timer | DTT |
| 030D₁₆ | Timer B2 interrupt occurrence frequency set counter | ICTB2 |
| 030E₁₆ | | |
| 030F₁₆ | | |
| 0310₁₆<br>0311₁₆ | Timer B3 register | TB3 |
| 0312₁₆<br>0313₁₆ | Timer B4 register | TB4 |
| 0314₁₆<br>0315₁₆ | Timer B5 register | TB5 |
| 0316₁₆ | | |
| 0317₁₆ | | |
| 0318₁₆ | | |
| 0319₁₆ | | |
| 031A₁₆ | | |
| 031B₁₆ | Timer B3 mode register | TB3MR |
| 031C₁₆ | Timer B4 mode register | TB4MR |
| 031D₁₆ | Timer B5 mode register | TB5MR |
| 031E₁₆ | | |
| 031F₁₆ | External interrupt cause select register | IFSR |

| Address | Register | |
|---|---|---|
| 0320₁₆ | | |
| 0321₁₆ | | |
| 0322₁₆ | | |
| 0323₁₆ | | |
| 0324₁₆ | UART3 special mode register 4 | U3SMR4 |
| 0325₁₆ | UART3 special mode register 3 | U3SMR3 |
| 0326₁₆ | UART3 special mode register 2 | U3SMR2 |
| 0327₁₆ | UART3 special mode register | U3SMR |
| 0328₁₆ | UART3 transmit-receive mode register | U3MR |
| 0329₁₆ | UART3 bit rate generator | U3BRG |
| 032A₁₆<br>032B₁₆ | UART3 transmit buffer register | U3TB |
| 032C₁₆ | UART3 transmit-receive control register 0 | U3C0 |
| 032D₁₆ | UART3 transmit-receive control register 1 | U3C1 |
| 032E₁₆<br>032F₁₆ | UART3 receive buffer register | U3RB |
| 0330₁₆ | | |
| 0331₁₆ | | |
| 0332₁₆ | | |
| 0333₁₆ | | |
| 0334₁₆ | UART2 special mode register 4 | U2SMR4 |
| 0335₁₆ | UART2 special mode register 3 | U2SMR3 |
| 0336₁₆ | UART2 special mode register 2 | U2SMR2 |
| 0337₁₆ | UART2 special mode register | U2SMR |
| 0338₁₆ | UART2 transmit-receive mode register | U2MR |
| 0339₁₆ | UART2 bit rate generator | U2BRG |
| 033A₁₆<br>033B₁₆ | UART2 transmit buffer register | U2TB |
| 033C₁₆ | UART2 transmit/receive control register 0 | U2C0 |
| 033D₁₆ | UART2 transmit/receive control register 1 | U2C1 |
| 033E₁₆<br>033F₁₆ | UART2 receive buffer register | U2RB |
| 0340₁₆ | Count start flag | TABSR |
| 0341₁₆ | Clock prescaler reset flag | CPSRF |
| 0342₁₆ | One-shot start flag | ONSF |
| 0343₁₆ | Trigger select register | TRGSR |
| 0344₁₆ | Up-down flag | UDF |
| 0345₁₆ | | |
| 0346₁₆<br>0347₁₆ | Timer A0 register | TA0 |
| 0348₁₆<br>0349₁₆ | Timer A1 register | TA1 |
| 034A₁₆<br>034B₁₆ | Timer A2 register | TA2 |
| 034C₁₆<br>034D₁₆ | Timer A3 register | TA3 |
| 034E₁₆<br>034F₁₆ | Timer A4 register | TA4 |

The blank area is reserved and cannot be used by user.

| Address | Register | |
|---|---|---|
| 0350₁₆ 0351₁₆ | Timer B0 register | TB0 |
| 0352₁₆ 0353₁₆ | Timer B1 register | TA1 |
| 0354₁₆ 0355₁₆ | Timer B2 register | TA2 |
| 0356₁₆ | Timer A0 mode register | TA0MR |
| 0357₁₆ | Timer A1 mode register | TA1MR |
| 0358₁₆ | Timer A2 mode register | TA2MR |
| 0359₁₆ | Timer A3 mode register | TA3MR |
| 035A₁₆ | Timer A4 mode register | TA4MR |
| 035B₁₆ | Timer B0 mode register | TB0MR |
| 035C₁₆ | Timer B1 mode register | TB1MR |
| 035D₁₆ | Timer B2 mode register | TB2MR |
| 035E₁₆ | Timer B2 special mode register | TB2SC |
| 035F₁₆ | Count source prescaler register | TCSPR |
| 0360₁₆ | | |
| 0361₁₆ | | |
| 0362₁₆ | | |
| 0363₁₆ | | |
| 0364₁₆ | UART0 special mode register 4 | U0SMR4 |
| 0365₁₆ | UART0 special mode register 3 | U0SMR3 |
| 0366₁₆ | UART0 special mode register 2 | U0SMR2 |
| 0367₁₆ | UART0 special mode register | U0SMR |
| 0368₁₆ | UART0 transmit/receive mode register | U0MR |
| 0369₁₆ | UART0 bit rate generator | U0BRG |
| 036A₁₆ 036B₁₆ | UART0 transmit buffer register | U0TB |
| 036C₁₆ | UART0 transmit/receive control register 0 | U0C0 |
| 036D₁₆ | UART0 transmit/receive control register 1 | U0C1 |
| 036E₁₆ 036F₁₆ | UART0 receive buffer register | U0RB |
| 0370₁₆ | | |
| 0371₁₆ | | |
| 0372₁₆ | | |
| 0373₁₆ | | |
| 0374₁₆ | | |
| 0375₁₆ | | |
| 0376₁₆ | PLL control register 0 | PLC0 |
| 0377₁₆ | | |
| 0378₁₆ | DMA0 cause select register | DM0SL |
| 0379₁₆ | DMA1 cause select register | DM1SL |
| 037A₁₆ | DMA2 cause select register | DM2SL |
| 037B₁₆ | DMA3 cause select register | DM3SL |
| 037C₁₆ 037D₁₆ | CRC data register | CRCD |
| 037E₁₆ | CRC input register | CRCIN |
| 037F₁₆ | | |

| Address | Register | |
|---|---|---|
| 0380₁₆ 0381₁₆ | A-D0 register 0 | AD00 |
| 0382₁₆ 0383₁₆ | A-D0 register 1 | AD01 |
| 0384₁₆ 0385₁₆ | A-D0 register 2 | AD02 |
| 0386₁₆ 0387₁₆ | A-D0 register 3 | AD03 |
| 0388₁₆ 0389₁₆ | A-D0 register 4 | AD04 |
| 038A₁₆ 038B₁₆ | A-D0 register 5 | AD05 |
| 038C₁₆ 038D₁₆ | A-D0 register 6 | AD06 |
| 038E₁₆ 038F₁₆ | A-D0 register 7 | AD07 |
| 0390₁₆ | | |
| 0391₁₆ | | |
| 0392₁₆ | | |
| 0393₁₆ | | |
| 0394₁₆ | A-D0 control register 2 | AD0CON2 |
| 0395₁₆ | | |
| 0396₁₆ | A-D0 control register 0 | AD0CON0 |
| 0397₁₆ | A-D0 control register 1 | AD0CON1 |
| 0398₁₆ | D-A register 0 | DA0 |
| 0399₁₆ | | |
| 039A₁₆ | D-A register 1 | DA1 |
| 039B₁₆ | | |
| 039C₁₆ | D-A control register | DACON |
| 039D₁₆ | | |
| 039E₁₆ | | |
| 039F₁₆ | | |

The blank area is reserved and cannot be used by user.

<144-pin version>

| Address | Register | |
|---|---|---|
| 03A0$_{16}$ | Function select register A8 | PS8 |
| 03A1$_{16}$ | Function select register A9 | PS9 |
| 03A2$_{16}$ | | |
| 03A3$_{16}$ | | |
| 03A4$_{16}$ | | |
| 03A5$_{16}$ | | |
| 03A6$_{16}$ | | |
| 03A7$_{16}$ | | |
| 03A8$_{16}$ | | |
| 03A9$_{16}$ | | |
| 03AA$_{16}$ | | |
| 03AB$_{16}$ | | |
| 03AC$_{16}$ | | |
| 03AD$_{16}$ | | |
| 03AE$_{16}$ | | |
| 03AF$_{16}$ | Function select register C | PSC |
| 03B0$_{16}$ | Function select register A0 | PS0 |
| 03B1$_{16}$ | Function select register A1 | PS1 |
| 03B2$_{16}$ | Function select register B0 | PSL0 |
| 03B3$_{16}$ | Function select register B1 | PSL1 |
| 03B4$_{16}$ | Function select register A2 | PS2 |
| 03B5$_{16}$ | Function select register A3 | PS3 |
| 03B6$_{16}$ | Function select register B2 | PSL2 |
| 03B7$_{16}$ | Function select register B3 | PSL3 |
| 03B8$_{16}$ | | |
| 03B9$_{16}$ | Function select register A5 | PS5 |
| 03BA$_{16}$ | | |
| 03BB$_{16}$ | | |
| 03BC$_{16}$ | Function select register A6 | PS6 |
| 03BD$_{16}$ | Function select register A7 | PS7 |
| 03BE$_{16}$ | | |
| 03BF$_{16}$ | | |
| 03C0$_{16}$ | Port P6 register | P6 |
| 03C1$_{16}$ | Port P7 register | P7 |
| 03C2$_{16}$ | Port P6 direction register | PD6 |
| 03C3$_{16}$ | Port P7 direction register | PD7 |
| 03C4$_{16}$ | Port P8 register | P8 |
| 03C5$_{16}$ | Port P9 register | P9 |
| 03C6$_{16}$ | Port P8 direction register | PD8 |
| 03C7$_{16}$ | Port P9 direction register | PD9 |
| 03C8$_{16}$ | Port P10 register | P10 |
| 03C9$_{16}$ | Port P11 register | P11 |
| 03CA$_{16}$ | Port P10 direction register | PD10 |
| 03CB$_{16}$ | Port P11 direction register | PD11 |
| 03CC$_{16}$ | Port P12 register | P12 |
| 03CD$_{16}$ | Port P13 register | P13 |
| 03CE$_{16}$ | Port P12 direction register | PD12 |
| 03CF$_{16}$ | Port P13 direction register | PD13 |

| Address | Register | |
|---|---|---|
| 03D0$_{16}$ | Port P14 register | P14 |
| 03D1$_{16}$ | Port P15 register | P15 |
| 03D2$_{16}$ | Port P14 direction register | PD14 |
| 03D3$_{16}$ | Port P15 direction register | PD15 |
| 03D4$_{16}$ | | |
| 03D5$_{16}$ | | |
| 03D6$_{16}$ | | |
| 03D7$_{16}$ | | |
| 03D8$_{16}$ | | |
| 03D9$_{16}$ | | |
| 03DA$_{16}$ | Pull-up control register 2 | PUR2 |
| 03DB$_{16}$ | Pull-up control register 3 | PUR3 |
| 03DC$_{16}$ | Pull-up control register 4 | PUR4 |
| 03DD$_{16}$ | | |
| 03DE$_{16}$ | | |
| 03DF$_{16}$ | | |
| 03E0$_{16}$ | Port P0 register | P0 |
| 03E1$_{16}$ | Port P1 register | P1 |
| 03E2$_{16}$ | Port P0 direction register | PD0 |
| 03E3$_{16}$ | Port P1 direction register | PD1 |
| 03E4$_{16}$ | Port P2 register | P2 |
| 03E5$_{16}$ | Port P3 register | P3 |
| 03E6$_{16}$ | Port P2 direction register | PD2 |
| 03E7$_{16}$ | Port P3 direction register | PD3 |
| 03E8$_{16}$ | Port P4 register | P4 |
| 03E9$_{16}$ | Port P5 register | P5 |
| 03EA$_{16}$ | Port P4 direction register | PD4 |
| 03EB$_{16}$ | Port P5 direction register | PD5 |
| 03EC$_{16}$ | | |
| 03ED$_{16}$ | | |
| 03EE$_{16}$ | | |
| 03EF$_{16}$ | | |
| 03F0$_{16}$ | Pull-up control register 0 | PUR0 |
| 03F1$_{16}$ | Pull-up control register 1 | PUR1 |
| 03F2$_{16}$ | | |
| 03F3$_{16}$ | | |
| 03F4$_{16}$ | | |
| 03F5$_{16}$ | | |
| 03F6$_{16}$ | | |
| 03F7$_{16}$ | | |
| 03F8$_{16}$ | | |
| 03F9$_{16}$ | | |
| 03FA$_{16}$ | | |
| 03FB$_{16}$ | | |
| 03FC$_{16}$ | | |
| 03FD$_{16}$ | | |
| 03FE$_{16}$ | | |
| 03FF$_{16}$ | Port control register | PCR |

The blank area is reserved and cannot be used by user.

<100-pin version>

| Address | Register | |
|---|---|---|
| 03A0$_{16}$ | | |
| 03A1$_{16}$ | | |
| 03A2$_{16}$ | | |
| 03A3$_{16}$ | | |
| 03A4$_{16}$ | | |
| 03A5$_{16}$ | | |
| 03A6$_{16}$ | | |
| 03A7$_{16}$ | | |
| 03A8$_{16}$ | | |
| 03A9$_{16}$ | | |
| 03AA$_{16}$ | | |
| 03AB$_{16}$ | | |
| 03AC$_{16}$ | | |
| 03AD$_{16}$ | | |
| 03AE$_{16}$ | | |
| 03AF$_{16}$ | Function select register C | PSC |
| 03B0$_{16}$ | Function select register A0 | PS0 |
| 03B1$_{16}$ | Function select register A1 | PS1 |
| 03B2$_{16}$ | Function select register B0 | PSL0 |
| 03B3$_{16}$ | Function select register B1 | PSL1 |
| 03B4$_{16}$ | Function select register A2 | PS2 |
| 03B5$_{16}$ | Function select register A3 | PS3 |
| 03B6$_{16}$ | Function select register B2 | PSL2 |
| 03B7$_{16}$ | Function select register B3 | PSL3 |
| 03B8$_{16}$ | | |
| 03B9$_{16}$ | | |
| 03BA$_{16}$ | | |
| 03BB$_{16}$ | | |
| 03BC$_{16}$ | | |
| 03BD$_{16}$ | | |
| 03BE$_{16}$ | | |
| 03BF$_{16}$ | | |
| 03C0$_{16}$ | Port P6 register | P6 |
| 03C1$_{16}$ | Port P7 register | P7 |
| 03C2$_{16}$ | Port P6 direction register | PD6 |
| 03C3$_{16}$ | Port P7 direction register | PD7 |
| 03C4$_{16}$ | Port P8 register | P8 |
| 03C5$_{16}$ | Port P9 register | P9 |
| 03C6$_{16}$ | Port P8 direction register | PD8 |
| 03C7$_{16}$ | Port P9 direction register | PD9 |
| 03C8$_{16}$ | Port P10 register | P10 |
| 03C9$_{16}$ | | |
| 03CA$_{16}$ | Port P10 direction register | PD10 |
| 03CB$_{16}$ | | |
| 03CC$_{16}$ | | |
| 03CD$_{16}$ | | |
| 03CE$_{16}$ | | |
| 03CF$_{16}$ | | |

| Address | Register | |
|---|---|---|
| 03D0$_{16}$ | | |
| 03D1$_{16}$ | | |
| 03D2$_{16}$ | | |
| 03D3$_{16}$ | | |
| 03D4$_{16}$ | | |
| 03D5$_{16}$ | | |
| 03D6$_{16}$ | | |
| 03D7$_{16}$ | | |
| 03D8$_{16}$ | | |
| 03D9$_{16}$ | | |
| 03DA$_{16}$ | Pull-up control register 2 | PUR2 |
| 03DB$_{16}$ | Pull-up control register 3 | PUR3 |
| 03DC$_{16}$ | | |
| 03DD$_{16}$ | | |
| 03DE$_{16}$ | | |
| 03DF$_{16}$ | | |
| 03E0$_{16}$ | Port P0 register | P0 |
| 03E1$_{16}$ | Port P1 register | P1 |
| 03E2$_{16}$ | Port P0 direction register | PD0 |
| 03E3$_{16}$ | Port P1 direction register | PD1 |
| 03E4$_{16}$ | Port P2 register | P2 |
| 03E5$_{16}$ | Port P3 register | P3 |
| 03E6$_{16}$ | Port P2 direction register | PD2 |
| 03E7$_{16}$ | Port P3 direction register | PD3 |
| 03E8$_{16}$ | Port P4 register | P4 |
| 03E9$_{16}$ | Port P5 register | P5 |
| 03EA$_{16}$ | Port P4 direction register | PD4 |
| 03EB$_{16}$ | Port P5 direction register | PD5 |
| 03EC$_{16}$ | | |
| 03ED$_{16}$ | | |
| 03EE$_{16}$ | | |
| 03EF$_{16}$ | | |
| 03F0$_{16}$ | Pull-up control register 0 | PUR0 |
| 03F1$_{16}$ | Pull-up control register 1 | PUR1 |
| 03F2$_{16}$ | | |
| 03F3$_{16}$ | | |
| 03F4$_{16}$ | | |
| 03F5$_{16}$ | | |
| 03F6$_{16}$ | | |
| 03F7$_{16}$ | | |
| 03F8$_{16}$ | | |
| 03F9$_{16}$ | | |
| 03FA$_{16}$ | | |
| 03FB$_{16}$ | | |
| 03FC$_{16}$ | | |
| 03FD$_{16}$ | | |
| 03FE$_{16}$ | | |
| 03FF$_{16}$ | Port control register | PCR |

The blank area is reserved and cannot be used by user.

Note 1: Addresses 03CB$_{16}$, 03CE$_{16}$, 03CF$_{16}$, 03D2$_{16}$, 03D3$_{16}$ does not exist in 100-pin version. Must set "FF$_{16}$" to the addresses at initial setting.

Note 2: Addresses 03DC$_{16}$ area does not exist in 100-pin version. Must set "00$_{16}$" to addresses 03DC$_{16}$ at initial setting.

Note 3: Addresses 03A0$_{16}$, 03A1$_{16}$, 03B9$_{16}$, 03BC$_{16}$, 03BD$_{16}$, 03C9$_{16}$, 03CC$_{16}$, 03CD$_{16}$, 03D30$_{16}$, 03D1$_{16}$ does not exist in 100-pin version.

## 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 the same effect as a hardware reset. The contents of internal RAM are preserved.

## Processor Mode

### (1) Types of Processor Mode

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

- **Single-chip mode**

  In single-chip mode, only internal memory space (SFR, internal RAM, and internal ROM) can be accessed. Ports P0 to P15 can be used as programmable I/O ports or as I/O ports for the internal peripheral functions.

- **Memory expansion mode**

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

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

- **Microprocessor mode**

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

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

### (2) Setting Processor Modes

The processor mode is set using the CNV$_{SS}$ pin and the processor mode bits (bits 1 and 0 at address $0004_{16}$). Do not set the processor mode bits to “$10_2$”.

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

- **Applying V$_{SS}$ to CNV$_{SS}$ pin**

  The microcomputer begins operation in single-chip mode after being reset. Memory expansion mode is selected by writing “$01_2$” to the processor mode is selected bits.

- **Applying V$_{CC}$ to CNV$_{SS}$ pin**

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

Figure 1.6.1 and 1.6.2 show the processor mode register 0 and 1.

Figure 1.6.3 shows the memory maps applicable for each processor modes.

## Processor mode register 0 (Note 1)

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

Symbol: PM0
Address: 0004₁₆
When reset: 80₁₆ (CNVss = "L"), 03₁₆ (CNVss = "H")

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PM00 | Processor mode bit (Note 2) | b1 b0<br>0 0: Single-chip mode<br>0 1: Memory expansion mode<br>1 0: Must not be set<br>1 1: Microprocessor mode | ○ | ○ |
| PM01 | | | ○ | ○ |
| PM02 | R/W mode select bit (Note 3) | 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 (Note 4) | b5 b4<br>0 0 : Multiplexed bus is not used<br>0 1 : Allocated to $\overline{CS2}$ space<br>0 1 : Allocated to $\overline{CS1}$ space<br>1 1 : Allocated to entire $\overline{CS}$ space (Note 5) | ○ | ○ |
| PM05 | | | ○ | ○ |
| ——— | Reserved bit | Must always be set to "0" | ○ | ○ |
| PM07 | BCLK output disable bit (Note 6) | 0 : BCLK is output (Note 7)<br>1 : Function set by bit 0,1 of system clock control register 0 | ○ | ○ |

Note 1: Set bit 1 of the protect register (address 000A₁₆) to "1" when writing new values to this register.
Note 2: Do not set the processor mode bits and other bits simultaneously when setting the processor mode bits to 012 or 112 . Set the other bits first,and then change the processor mode bits.
Note 3: When using 16-bit bus width in DRAM controler, must set this bit to "1".
Note 4: Valid in microprocessor and memory expansion modes 1, 2 and 3. Do not use multiplex bus when mode 0 is selected. Do not set to allocated to $\overline{CS2}$ space when mode 2 is selected.
Note 5: After the reset has been released, the M32C/83 group MCU operates using the separate bus. As a result, in microprocessor mode, you cannot select the full $\overline{CS}$ space multiplex bus.
When you select the full $\overline{CS}$ space multiplex bus in memory expansion mode, the address bus operates with 64 Kbytes boundaries, for each chip select.
Mode 0: Multiplexed bus cannot be used.
Mode 1: $\overline{CS0}$ to $\overline{CS2}$ when you select full $\overline{CS}$ space.
Mode 2: $\overline{CS0}$ to $\overline{CS1}$ when you select full $\overline{CS}$ space.
Mode 3: $\overline{CS0}$ to $\overline{CS3}$ when you select full $\overline{CS}$ space.
Note 6: No BCLK is output in single chip mode even when "0" is set in PM07. When stopping clock output in microprocessor or memory expansion mode, make the following settings: PM07="1", bit 0 (CM00) and bit 1 (CM01) of system clock control register 0 (address 0006₁₆) = "0". "L" is now output from P5₃.
Note 7: When selecting BCLK, set bits 0 and 1 of system clock control register 0 (CM00, CM01) to "0".

**Figure 1.6.1. Processor mode register 0**

## Processor mode register 1 (Note 1)

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

| Symbol | Address | When reset |
|---|---|---|
| PM1 | $0005_{16}$ | $0X000000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PM10 | External memory area mode bit (Note 2) | b1 b0<br>0 0 : Mode 0 ($P4_4$ to $P4_7$ : $A_{20}$ to $\overline{A_{23}}$)<br>0 1 : Mode 1 ($P4_4$: $A_{20}$, $P4_5$ to $P4_7$: $\overline{CS2}$ to $\overline{CS0}$)<br>1 0 : Mode 2 ($P4_4$, $P4_5$ : $A_{20}$, $A_{21}$, $P4_6$, $P4_7$ : $\overline{CS1}$, $\overline{CS0}$)<br>1 1 : Mode 3 (Note 3) ($P4_4$ to $P4_7$ : $\overline{CS3}$ to $\overline{CS0}$) | ○ | ○ |
| PM11 | | | ○ | ○ |
| PM12 | Internal memory wait bit | 0 : No wait state<br>1 : Wait state inserted | ○ | ○ |
| PM13 | SFR area wait bit 0 | 0 : One wait state inserted<br>1 : Two wait states inserted (Note 4) | ○ | ○ |
| PM14 | ALE pin select bit (Note 2) | b5 b4<br>0 0 : No ALE<br>0 1 : $P5_3$/BCLK (Note 5)<br>1 0 : $P5_6$/$\overline{RAS}$<br>1 1 : $P5_4$/$\overline{HLDA}$ | ○ | ○ |
| PM15 | | | ○ | ○ |
| Nothing is assigned. When write, set "0". When read, its content is indeterminate. | | | – | – |
| PM17 | Reserved bit | Must set to "0" | ○ | ○ |

Note 1: Set bit 1 of the protect register (address $000A_{16}$) to “1” when writing new values to this register.
Note 2: Valid in memory expansion mode or in microprocessor mode.
Note 3: When mode 3 is selected, DRAMC is not used.
Note 4: When accessing SFR area for CAN, PM13 must be set to "1".
Note 5: When selecting $P5_3$/BCLK, set bits 0 and 1 of system clock control register 0 (CM00, CM01) to "0".

**Figure 1.6.2. Processor mode register 1**

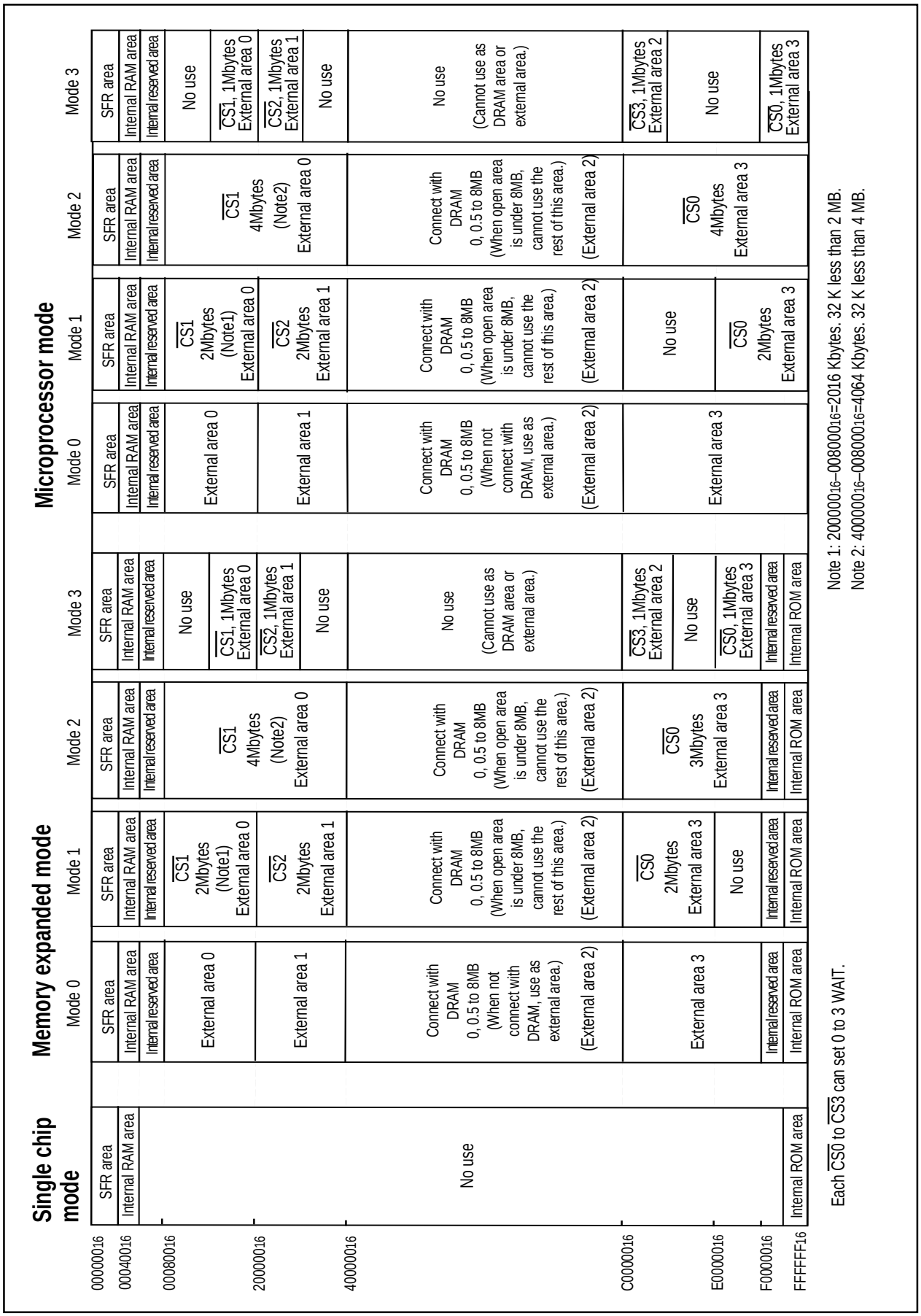



**Figure 1.6.3. Memory maps in each processor mode**

## Bus Settings

The BYTE pin, bit 0 to 3 of the external data bus width control register (address $000B_{16}$), bits 4 and 5 of the processor mode register 0 (address $0004_{16}$) and bit 0 and 1 of the processor mode register 1 (address $0005_{16}$) are used to change the bus settings.
Table 1.7.1 shows the factors used to change the bus settings, figure 1.7.1 shows external data bus width control register and table 1.7.2 shows external area 0 to 3 and external area mode.

**Table 1.7.1. Factors for switching bus settings**

| Bus setting | Switching factor |
|---|---|
| Switching external address bus width | External data bus width control register |
| Switching external data bus width | BYTE pin (external area 3 only) |
| Switching between separate and multiplex bus | Bits 4 and 5 of processor mode register 0 |
| Selecting external area | Bits 0 and 1 of processor mode register 1 |

### (1) Selecting external address bus width

You can select the width of the address bus output externally from the 16 Mbytes address space, the number of chip select signals, and the address area of the chip select signals. (Note, however, that when you select "Full $\overline{CS}$ space multiplex bus", addresses $A_0$ to $A_{15}$ are output.) The combination of bits 0 and 1 of the processor mode register 1 allow you to set the external area mode.
When using DRAM controller, the DRAM area is output by multiplexing of the time splitting of the row and column addresses.

### (2) Selecting external data bus width

You can select 8-bit or 16-bit for the width of the external data bus for external areas 0, 1, 2, and 3. When the data bus width bit of the external data bus width control register is "0", the data bus width is 8 bits; when "1", it is 16 bits. The width can be set for each of the external areas. The default bus width for external area 3 is 16 bits when the BYTE pin is "L" after a reset, or 8 bits when the BYTE pin is "H" after a reset. The bus width selection is valid only for the external bus (the internal bus width is always 16 bits).
During operation, fix the level of the BYTE pin to "H" or "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 bus configuration, input and output is performed on separate data and address buses. The data bus width can be set to 8 bits or 16 bits using the external data bus width control register. For all programmable external areas, P0 is the data bus when the external data bus is set to 8 bits, and P1 is a programmable IO port. When the external data bus width is set to 16 bits for any of the external areas, P0 and P1 (although P1 is undefined for any 8-bit bus areas) are the data buses.
  When accessing memory using the separate bus configuration, you can select a software wait using the wait control register.

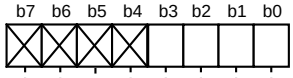
- **Multiplex bus**

  In this bus configuration, data and addresses are input and output on a time-sharing basis. For areas for which 8-bit has been selected using the external data bus width control register, the 8 bits $D_0$ to $D_7$ are multiplexed with the 8 bits $A_0$ to $A_7$. For areas for which 16-bit has been selected using the external data bus width control register, the 16 bits $D_0$ to $D_{15}$ are multiplexed with the 16 bits $A_0$ to $A_{15}$. When

accessing memory using the multiplex bus configuration, two waits are inserted regardless of whether you select “No wait” or “1 wait” in the appropriate bit of the wait control register.

The default after a reset is a separate bus configuration, and the full $\overline{CS}$ space multiplex bus configuration cannot be selected in microprocessor mode. If you select “Full $\overline{CS}$ space multiplex bus”, the 16 bits from A0 to A15 are output for the address

External data bus width control register




Symbol: DS | Address: 000B16 | When reset: XXXXX0002

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DS0 | External area 0 data bus width bit | 0 : 8 bits data bus width<br>1 : 16 bits data bus width | ○ | ○ |
| DS1 | External area 1 data bus width bit | 0 : 8 bits data bus width<br>1 : 16 bits data bus width | ○ | ○ |
| DS2 | External area 2 data bus width bit | 0 : 8 bits data bus width<br>1 : 16 bits data bus width | ○ | ○ |
| DS3 | External area 3 data bus width bit (Note) | 0 : 8 bits data bus width<br>1 : 16 bits data bus width | ○ | ○ |
| Nothing is assigned.<br>When write, set "0". When read, their contents are indeterminate. | | | — | — |

Note: The value after a reset is determined by the input via the BYTE pin.
When BYTE pin is "L", DS3 is "1". When "H", it is "0".

**Figure 1.7.1. External data bus width control register**

**Table 1.7.2. External area 0 to 3 and external area mode**

| | External area mode (Note 2) | Mode 0 | Mode 1 | Mode 2 | Mode 3 |
|---|---|---|---|---|---|
| External area 0 | Memory expansion mode, Microprocessor mode | 008000$_{16}$ to 1FFFFF$_{16}$ | <CS1 area> 008000$_{16}$ to 1FFFFF$_{16}$ | <CS1 area> 008000$_{16}$ to 1FFFFF$_{16}$ | <CS1 area> 100000$_{16}$ to 1FFFFF$_{16}$ |
| External area 1 | Memory expansion mode, Microprocessor mode | 200000$_{16}$ to 3FFFFF$_{16}$ | <CS2 area> 200000$_{16}$ to 3FFFFF$_{16}$ | No area is selected. | <CS2 area> 200000$_{16}$ to 2FFFFF$_{16}$ |
| External area 2 | Memory expansion mode, Microprocessor mode | 400000$_{16}$ to BFFFFF$_{16}$ (Note 1) | <DRAMC area> 400000$_{16}$ to BFFFFF$_{16}$ | <DRAMC area> 400000$_{16}$ to BFFFFF$_{16}$ | <CS3 area> C00000$_{16}$ to CFFFFF$_{16}$ |
| External area 3 | Memory expansion mode | C00000$_{16}$ to EFFFFF$_{16}$ | <CS0 area> C00000$_{16}$ to EFFFFF$_{16}$ | <CS0 area> C00000$_{16}$ to EFFFFF$_{16}$ | <CS0 area> E00000$_{16}$ to EFFFFF$_{16}$ |
| | Microprocessor mode | C00000$_{16}$ to FFFFFF$_{16}$ | <CS0 area> E00000$_{16}$ to FFFFFF$_{16}$ | <CS0 area> C00000$_{16}$ to FFFFFF$_{16}$ | <CS0 area> F00000$_{16}$ to FFFFFF$_{16}$ |

Note 1: DRAMC area when using DRAMC.

Note 2: Set the external area mode (modes 0, 1, 2, and 3) using bits 0 and 1 of the processor mode register 1 (address 0005$_{16}$).

**Table 1.7.3. Each processor mode and port function**

| Processor mode | Single-chip mode | Memory expansion mode/microprocessor modes | | | | Memory expansion mode | |
|---|---|---|---|---|---|---|---|
| Multiplexed bus space select bit | | "01", "10"<br>CS1 or CS2 : multiplexed bus, and the other : separate bus | | "00"<br>Separate bus | | "11" (Note 1)<br>All space multiplexed bus | |
| Data bus width BYTE pin level | | All external area is 8 bits | Some external area is 16 bits | All external area is 8 bits | Some external area is 16 bits | All external area is 8 bits | Some external area is 16 bits |
| $P0_0$ to $P0_7$ | I/O port | Data bus | Data bus | Data bus | Data bus | I/O port | I/O port |
| $P1_0$ to $P1_7$ | I/O port | I/O port | I/O port | Data bus | I/O port | I/O port | I/O port |
| $P2_0$ to $P2_7$ | I/O port | Address bus /data bus (Note 2) | Address bus /data bus (Note 2) | Address bus | Address bus | Address bus /data bus | Address bus /data bus |
| $P3_0$ to $P3_7$ | I/O port | Address bus | Address bus /data bus (Note 2) | Address bus | Address bus | Address bus | Address bus /data bus |
| $P4_0$ to $P4_3$ | I/O port | Address bus | Address bus | Address bus | Address bus | I/O port | I/O port |
| $P4_4$ to $P4_6$ | I/O port | CS (chip select) or address bus ($A_{23}$) (For details, refer to "Bus control") (Note 5) | | | | | |
| $P4_7$ | I/O port | $\overline{CS}$ (chip select) or address bus ($A_{23}$) (For details, refer to "Bus control") (Note 5) | | | | | |
| $P5_0$ to $P5_3$ | I/O port | Outputs $\overline{RD}$, $\overline{WRL}$, $\overline{WRH}$, and BCLK or $\overline{RD}$, $\overline{BHE}$, $\overline{WR}$, and BCLK (For details, refer to "Bus control") (Note 3,4) | | | | | |
| $P5_4$ | I/O port | $\overline{HLDA}$(Note 3) | $\overline{HLDA}$(Note 3) | $\overline{HLDA}$(Note 3) | $\overline{HLDA}$(Note 3) | $\overline{HLDA}$(Note 3) | $\overline{HLDA}$(Note 3) |
| $P5_5$ | I/O port | $\overline{HOLD}$ | $\overline{HOLD}$ | $\overline{HOLD}$ | $\overline{HOLD}$ | $\overline{HOLD}$ | $\overline{HOLD}$ |
| $P5_6$ | I/O port | RAS (Note 3) | RAS (Note 3) | RAS (Note 3) | RAS (Note 3) | RAS (Note 3) | RAS (Note 3) |
| $P5_7$ | I/O port | $\overline{RDY}$ | $\overline{RDY}$ | $\overline{RDY}$ | $\overline{RDY}$ | $\overline{RDY}$ | $\overline{RDY}$ |

Note 1:The default after a reset is the separate bus configuration, and "Full $\overline{CS}$ space multiplex bus" cannot be selected in microprocessor mode. When you select "Full $\overline{CS}$ space multiplex bus" in extended memory mode, the address bus operates with 64 Kbytes boundaries for each chip select.
Note 2: Address bus in separate bus configuration.
Note 3: The ALE output pin is selected using bits 4 and 5 of the processor mode register 1.
Note 4: When you have selected the DRAM controller and access the DRAM area, these are outputs $\overline{CASL}$, $\overline{CASH}$, $\overline{DW}$, and BCLK.
Note 5: The $\overline{CS}$ signal and address bus selection are set by the external area mode.

## Bus Control

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

### (1) Address bus/data bus

There are 24 pins, $A_0$ to $A_{22}$ and $\overline{A_{23}}$ for the address bus for accessing the 16 Mbytes address space. $\overline{A_{23}}$ is an inverted output of the MSB of the address.

The data bus consists of  pins for data IO. The external data bus control register (address $000B_{16}$) selects the 8-bit data bus, $D_0$ to $D_7$ for each external area, or the 16-bit data bus, $D_0$ to $D_{15}$. After a reset, there is by default an 8-bit data bus for the external area 3 when the BYTE pin is High, or a 16-bit data bus when the BYTE pin is Low.

When shifting from single-chip mode to extended memory mode, the value on the address bus is undefined until an external area is accessed.

When accessing a DRAM area with DRAM control in use, a multiplexed signal consisting of row address and column address is output to $A_8$ to $A_{20}$.

### (2) Chip select signals

The chip select signals share $A_0$ to $A_{22}$ and $\overline{A_{23}}$. You can use bits 0 and 1 of the processor mode register 1 (address $0005_{16}$) to set the external area mode, then select the chip select area and number of address outputs.

In microprocessor mode, external area mode 0 is selected after a reset. The external area can be split into a maximum of four Blocks or Areas using the chip select signals. Table 1.7.4 shows the external areas specified by the chip select signals.

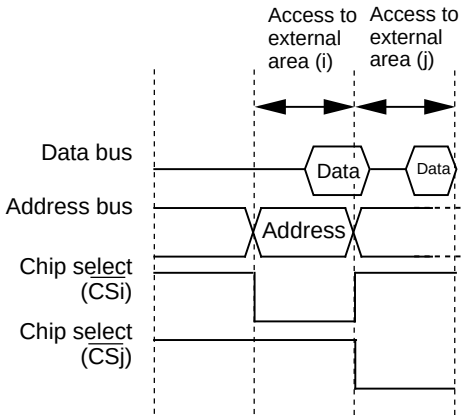
**Table 1.7.4. External areas specified by the chip select signals**

| Memory space expansion mode | | Processor mode | Chip select signal | | | |
|---|---|---|---|---|---|---|
| | | | $\overline{CS0}$ | $\overline{CS1}$ | $\overline{CS2}$ | $\overline{CS3}$ |
| Specified address range | Mode 0 | | $\overline{\overline{(A23)}}$ | $\overline{(A22)}$ | $\overline{(A21)}$ | $\overline{(A20)}$ |
| | Mode 1 | Memory expansion mode | $C00000_{16}$ to $DFFFFF_{16}$ (2 Mbytes) | $008000_{16}$ to $1FFFFF_{16}$ (2016 Kbytes) | $200000_{16}$ to $3FFFFF_{16}$ (2 Mbytes) | $\overline{(A20)}$ |
| | | Microprocessor mode | $E00000_{16}$ to $FFFFFF_{16}$ (2 Mbytes) | | | |
| | Mode 2 | Memory expansion mode | $C00000_{16}$ to $EFFFFF_{16}$ (3 Mbytes) | $008000_{16}$ to $3FFFFF_{16}$ (4064 Kbytes) | $\overline{(A21)}$ | $\overline{(A20)}$ |
| | | Microprocessor mode | $C00000_{16}$ to $FFFFFF_{16}$ (4 Mbytes) | | | |
| | Mode 3 | Memory expansion mode | $E00000_{16}$ to $EFFFFF_{16}$ (1 Mbytes) | $100000_{16}$ to $1FFFFF_{16}$ (1 Mbytes) | $200000_{16}$ to $2FFFFF_{16}$ (1 Mbytes) | $C00000_{16}$ to $CFFFFF_{16}$ (1 Mbytes) |
| | | Microprocessor mode | $F00000_{16}$ to $FFFFFF_{16}$ (1 Mbytes) | | | |

The chip select signal turns Low (active) in synchronize with the address bus. However, its turning High depends on the area accessed in the next cycle. Figure 1.7.2 shows the output examples of the address bus and chip select signals.

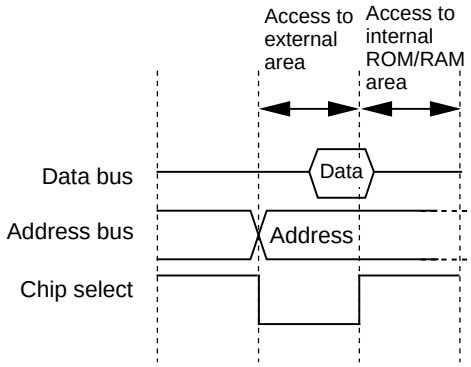
**Example 1:** After accessing the external area, the address bus and chip select signal both are changed in the next cycle.

The following example shows the other chip select signal accessing area (j) in the cycle after having accessed external area (i). In this case, the address bus and chip select signal both change between the two cycles.



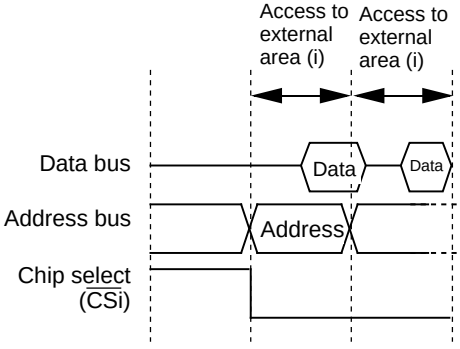

**Example 2:** After accessing the external area, only the chip select signal is changed in the next cycle. (The address bus does not change.)

The following example shows the CPU accesses the internal ROM/RAM area in the cycle after having accessed external area. In this case, the chip select signal changes between the two cycles but the address bus does not.



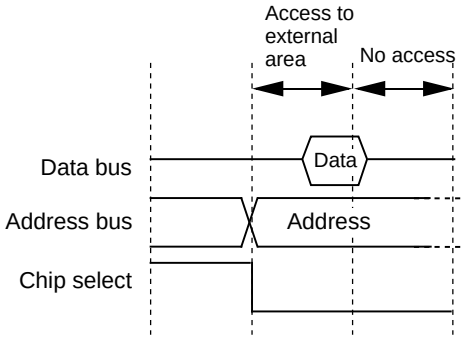

**Example 3:** After accessing the external area, only the address bus is changed in the next cycle. (The chip select signal does not change.)

The following example shows the same chip select signal accessing area (i) in the cycle after having accessed external area (i). In this case, the address bus changes between the two cycles, but the chip select signal does not.




**Example 4:** After accessing the external area, the address bus and chip select signal both are not changed in the next cycle.

The following example shows CPU does not access any area in the cycle after having accessed external area (no instruction pre-fetch is occurred). In this case, the address bus and the chip select signal do not change between the two cycles.




Note: These examples show the address bus and chip select signal for two consecutive cycles. By combining these examples, chip select signal can be extended beyond two cycles.

**Figure 1.7.2. Example of address bus and chip select signal outputs (Separate bus)**

### (3) Read/write signals

With a 16-bit data bus, bit 2 of the processor mode register 0 (address $0004_{16}$) selects the combinations of $\overline{RD}$, $\overline{BHE}$, and $\overline{WR}$ signals or $\overline{RD}$, $\overline{WRL}$, and $\overline{WRH}$ signals. With a 8-bit full space data bus, use the combination of $\overline{RD}$, $\overline{WR}$, and $\overline{BHE}$ signals as read/write signals. (Set "0" to bit 2 of the processor mode register 0 (address $0004_{16}$).) When using both 8-bit and 16-bit data bus widths to access a 8-bit data bus area, the $\overline{RD}$, $\overline{WR}$ and $\overline{BHE}$ signals combination is selected regardless of the value of bit 2 of the processor mode register 0 (address $0004_{16}$).

Tables 1.7.5 and 1.7.6 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 1: Before attempting to change the contents of the processor mode register 0, set bit 1 of the protect register (address $000A_{16}$) to “1”.

Note 2: When using 16-bit data bus width for DRAM controller, select $\overline{RD}$, $\overline{WRL}$, and $\overline{WRH}$ signals.

**Table 1.7.5. 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 | 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 |
| 8-bit | H | L (Note) | Not used | Write 1 byte of data |
| | L | H (Note) | Not used | Read 1 byte of data |

Note: It becomes $\overline{WR}$ signal.

**Table 1.7.6. 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 | 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 | 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. The ALE output pin is selected using bits 4 and 5 of the processor mode register 1 (address $0005_{16}$).

The ALE signal is occurred regardless of internal area and external area.

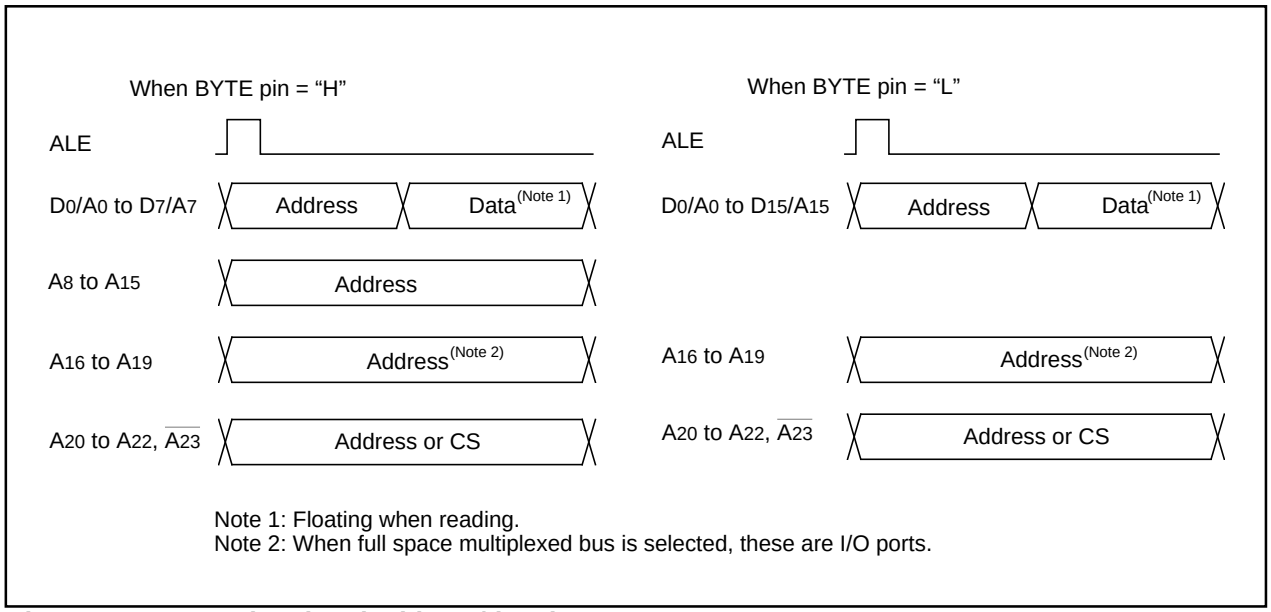



**Figure 1.7.3. ALE signal and address/data bus**

### (5) Ready signal

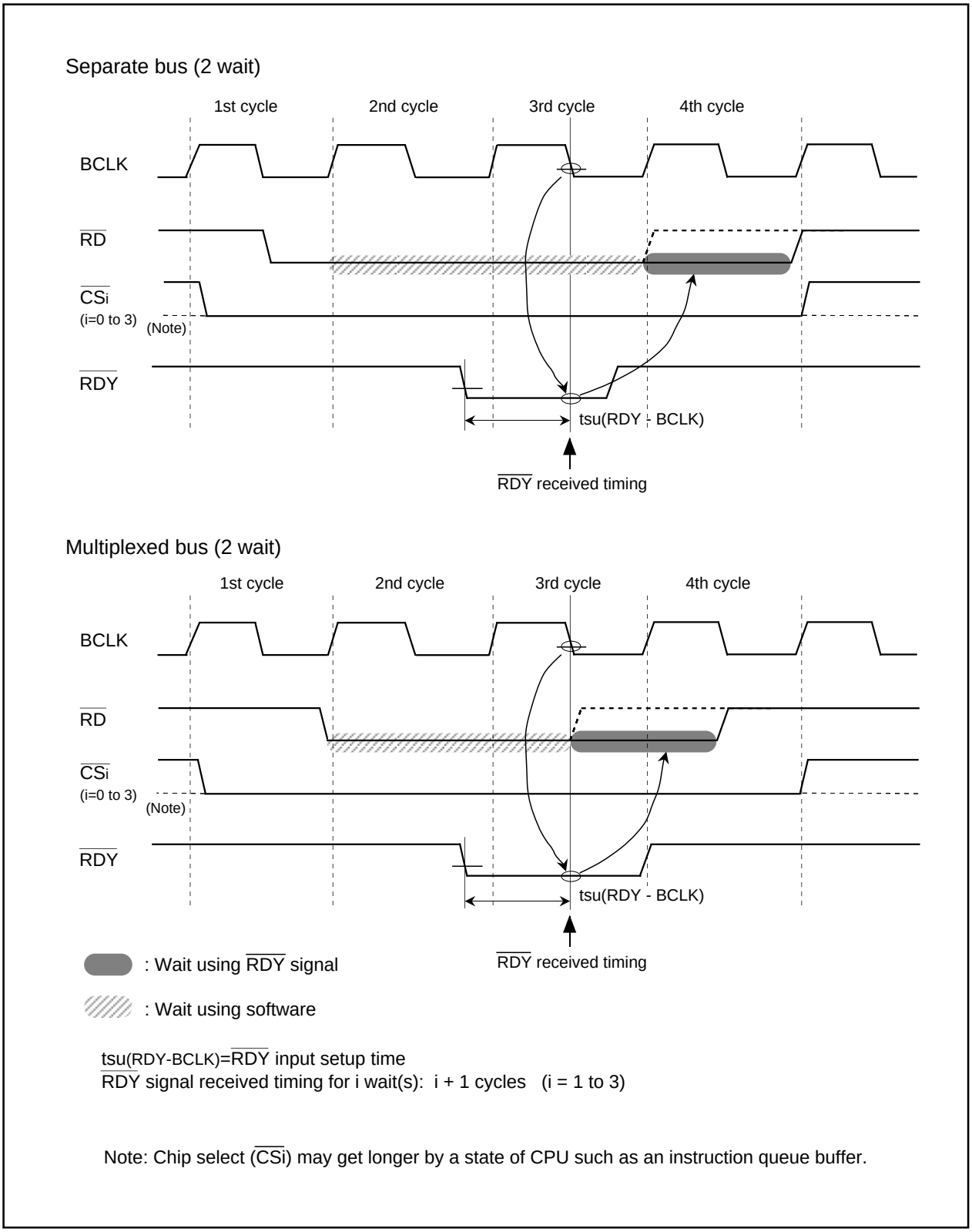
The ready signal facilitates access of external devices that require a long time for access. As shown in Figure 1.7.2, inputting "L" to the $\overline{RDY}$ pin at the falling edge of BCLK causes the microcomputer to enter the ready state. Inputting "H" to the $\overline{RDY}$ pin at the falling edge of BCLK cancels the ready state. Table 1.7.7 shows the microcomputer status in the ready state. Figure 1.7.4 shows the example of the $\overline{RD}$ signal being extended using the $\overline{RDY}$ signal.

Ready is valid when accessing the external area during the bus cycle in which the software wait is applied. When no software wait is operating, the $\overline{RDY}$ signal is ignored, but even in this case, unused pins must be pulled up.

**Table 1.7.7. Microcomputer status in ready state** (Note)

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

Note: The ready signal cannot be received immediately prior to a software wait.




tsu(RDY-BCLK)=RDY input setup time
RDY signal received timing for i wait(s): i + 1 cycles (i = 1 to 3)

Note: Chip select (CSi) may get longer by a state of CPU such as an instruction queue buffer.

**Figure 1.7.4. Example of RD signal extended by RDY signal**

## (6) Hold signal

The hold signal is used to transfer the bus privileges from the CPU to the external circuits. Inputting “L” to the $\overline{\text{HOLD}}$ pin places the microcomputer in the hold state at the end of the current bus access. This status is maintained and “L” is output from the $\overline{\text{HLDA}}$ pin as long as “L” is input to the $\overline{\text{HOLD}}$ pin. Table 1.7.8 shows the microcomputer status in the hold state. The bus is used in the following descending order of priority: $\overline{\text{HOLD}}$, DMAC, CPU.

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

**Figure 1.7.5. Example of $\overline{\text{RD}}$ signal extended by $\overline{\text{RDY}}$ signal**

**Table 1.7.8. Microcomputer status in hold state**

| Item | Status |
|---|---|
| Oscillation | ON |
| $\overline{\text{RD}}$/$\overline{\text{WR}}$ signal, address bus, data bus, $\overline{\text{CS}}$, $\overline{\text{BHE}}$ | Floating |
| Programmable I/O ports: P0 to P15 | Maintains status when hold signal is received |
| $\overline{\text{HLDA}}$ | Output “L” |
| Internal peripheral circuits | ON (but watchdog timer stops) |
| ALE signal | Output “L” |

## (7) External bus status when accessing to internal area

Table 1.7.9 shows external bus status when accessing to internal area

**Table 1.7.9. External bus status when accessing to internal area**

| Item | | SFR accessing status | Internal ROM/RAM accessing status |
|---|---|---|---|
| Address bus | | Remain address of external area accessed immediately before | |
| Data bus | When read | Floating | |
| | When write | Floating | |
| $\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{WRL}}$, $\overline{\text{WRH}}$ | | Output "H" | |
| $\overline{\text{BHE}}$ | | Remain external area status accessed immediately before | |
| $\overline{\text{CS}}$ | | Output "H" | |
| ALE | | ALE output | |

## (8) BCLK output

BCLK output can be selected by bit 7 of the processor mode register 0 (address $0004_{16}$ :PM07) and bit 1 and bit 0 of the system clock select register 0 (address $0006_{16}$ :CM01, CM00). Setting PM07 to “0” and CM01 and CM00 to “00” outputs the BCLK signal from $P5_3$. However, in single chip mode, BCLK signal is inactive. When setting PM07 to “1”, the function is set by CM01 and CM00.

## (9) DRAM controller signals ($\overline{RAS}$, $\overline{CASL}$, $\overline{CASH}$, and $\overline{DW}$)

Bits 1, 2, and 3 of the DRAM control register (address $0004_{16}$) select the DRAM space and enable the DRAM controller. The DRAM controller signals are output when the DRAM area is accessed. Table 1.7.10 shows the operation of the respective signals.

**Table 1.7.10. Operation of $\overline{RAS}$, $\overline{CASL}$, $\overline{CASH}$, and $\overline{DW}$ signals**

| Data bus width | $\overline{RAS}$ | $\overline{CASL}$ | $\overline{CASH}$ | $\overline{DW}$ | Status of external data bus |
|---|---|---|---|---|---|
| 16-bit | L | L | L | H | Read data from both even and odd addresses |
| | L | L | H | H | Read 1 byte of data from even address |
| | L | L | H | H | Read 1 byte of data from odd address |
| | L | L | L | L | Write data to both even and odd addresses |
| | L | L | H | L | Write 1 byte of data to even address |
| | L | H | L | L | Write 1 byte of data to odd address |
| 8-bit | L | L | Not used | H | Read 1 byte of data |
| | L | L | Not used | L | Write 1 byte of data |

## (10) Software wait

A software wait can be inserted by setting the wait control register (address $0008_{16}$). Figure 1.7.6 shows wait control register.

You can use the external area i wait bits (where i = 0 to 3) of the wait control register to specify from “No wait” to “3 waits” for the external memory area. When you select “No wait”, the read cycle is executed in the BCLK1 cycle. The write cycle is executed in the BCLK2 cycle (which has 1 wait). When accessing external memory using the multiplex bus, access has two waits regardless of whether you specify “No wait” or “1 wait” in the appropriate external area i wait bits in the wait control register.

Software waits in the internal memory (internal RAM and internal ROM) can be set using the internal memory wait bits of the processor mode register 1 (address $0005_{16}$). Setting the internal memory wait bit = “0” sets “No wait”. Setting the internal memory wait bit = “1” specifies a wait.

SFR area is accessed with either "1 wait" (BCLK 2-cycle) or "2 waits" (BCLK 3-cycle) by setting the SFR wait bit (bit 3) of the processor mode register 1 (address $0005_{16}$). SFR area of CAN must be accessed with "2 waits".

Table 1.7.11 shows the software waits and bus cycles. Figures 1.7.7 and 1.7.8 show example bus timing when using software waits.

Wait control register (Note 1, 2)

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

Symbol: WCR
Address: 0008$_{16}$
When reset: FF$_{16}$

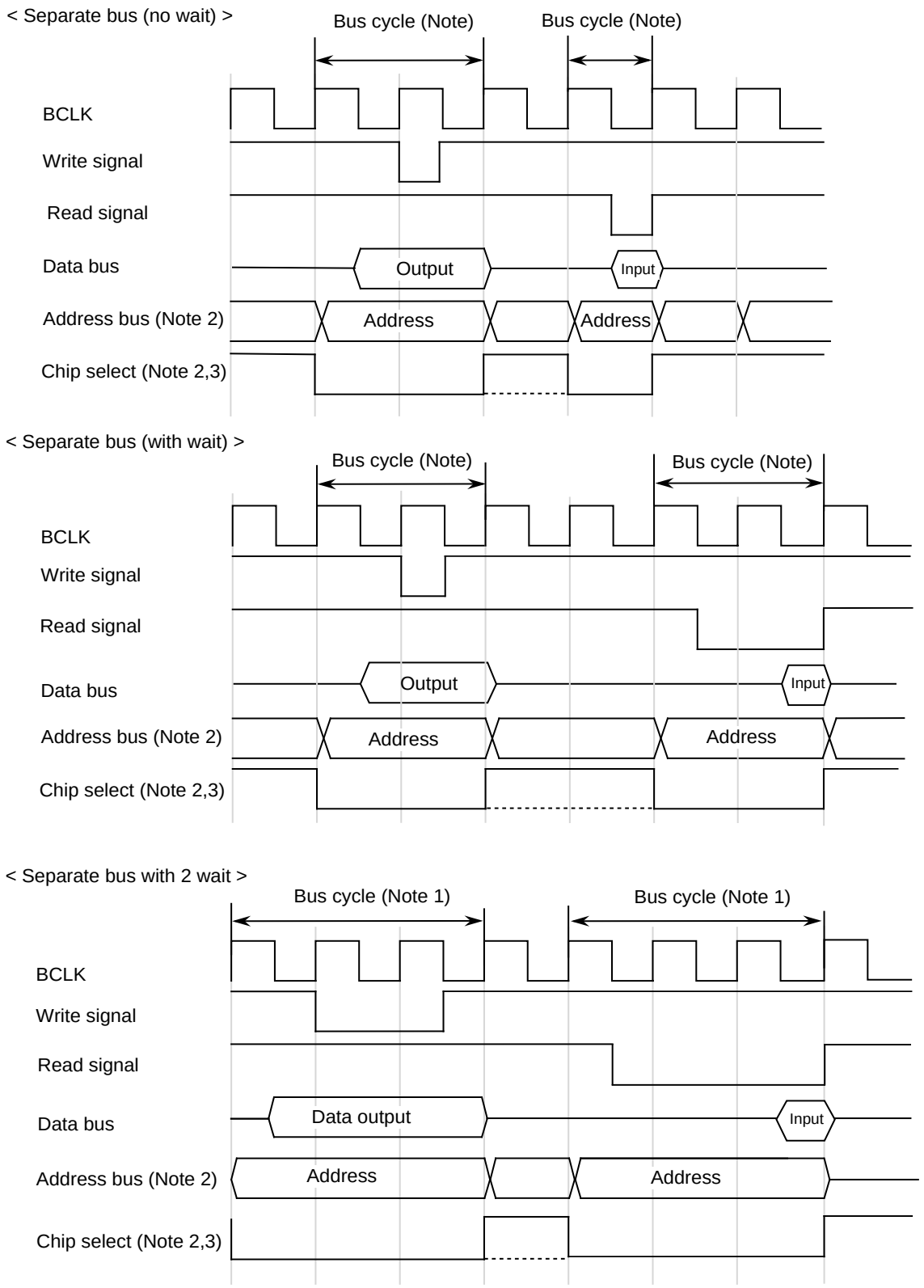
| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| WCR0 | External area 0 wait bit | b1 b0<br>0 0: Without wait<br>0 1: With 1 wait<br>1 0: With 2 waits<br>1 1: With 3 waits | ○ | ○ |
| WCR1 | | | ○ | ○ |
| WCR2 | External area 1 wait bit | b3 b2<br>0 0: Without wait<br>0 1: With 1 wait<br>1 0: With 2 waits<br>1 1: With 3 waits | ○ | ○ |
| WCR3 | | | ○ | ○ |
| WCR4 | External area 2 wait bit | b5 b4<br>0 0: Without wait<br>0 1: With 1 wait<br>1 0: With 2 waits<br>1 1: With 3 waits | ○ | ○ |
| WCR5 | | | ○ | ○ |
| WCR6 | External area 3 wait bit | b7 b6<br>0 0: Without wait<br>0 1: With 1 wait<br>1 0: With 2 waits<br>1 1: With 3 waits | ○ | ○ |
| WCR7 | | | ○ | ○ |

Note 1: When using the multiplex bus configuration, there are two waits regardless of whether you have specified "No wait" or "1 wait". However, you can specify "2 waits" or "3 waits".
Note 2: When using the separate bus configuration, the read bus cycle is executed in the BCLK1 cycle, and the write cycle is executed in the BCLK2 cycle (with 1 wait).

**Figure 1.7.6. Wait control register**

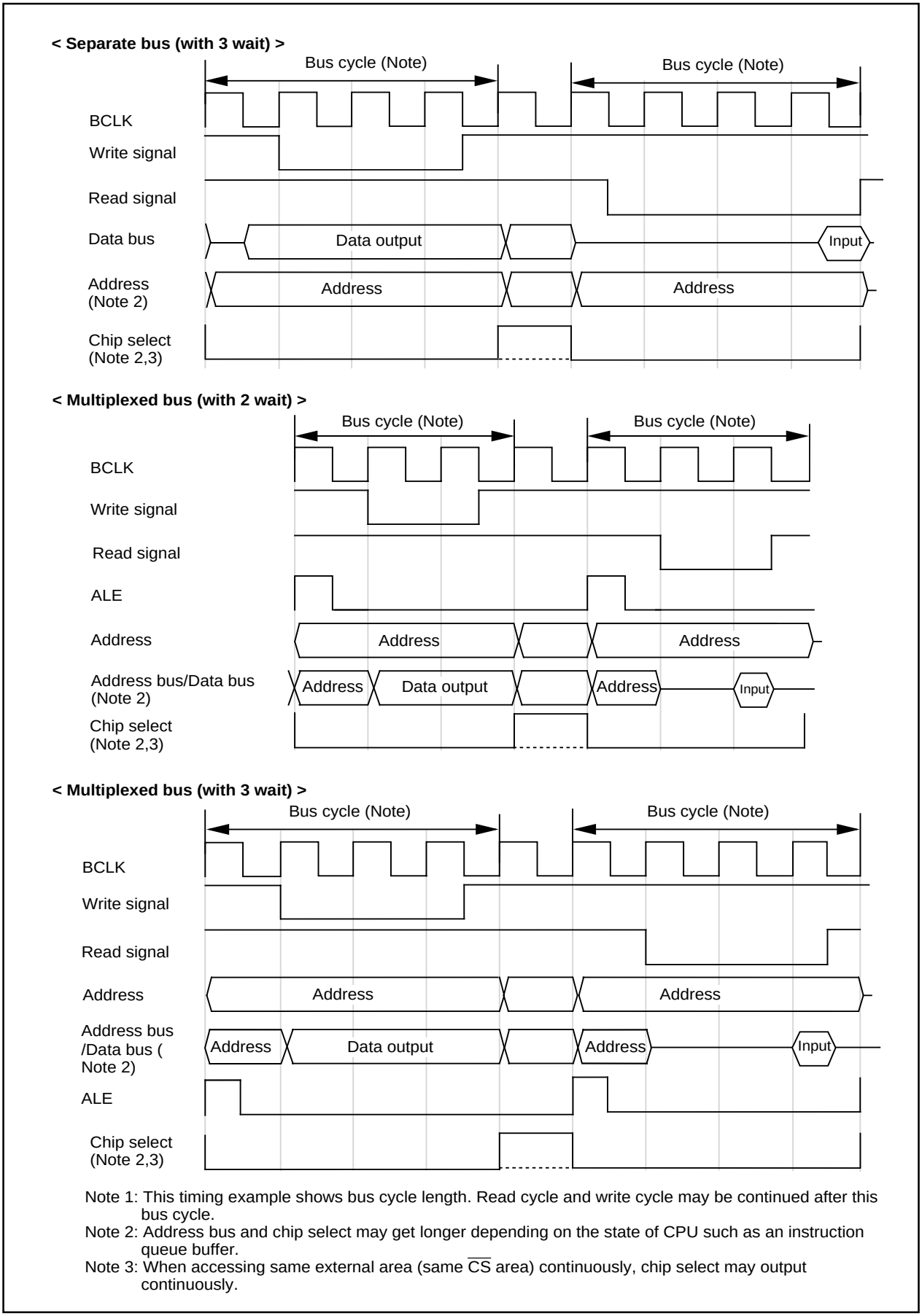
**Table 1.7.11. Software waits and bus cycles**

| Area | Bus status | SFR area wait bit | Internal memory wait bit | External memory area i wait bit | Bus cycle |
|---|---|---|---|---|---|
| SFR | ——— | 0 | ——— | ——— | 2 BCLK cycles |
| | ——— | 1 | ——— | ——— | 3 BCLK cycles |
| Internal ROM/RAM | ——— | ——— | 0 | ——— | 1 BCLK cycle |
| | ——— | ——— | 1 | ——— | 2 BCLK cycles |
| External memory area | Separate bus | ——— | ——— | 00$_2$ | Read :1 BCLK cycle<br>Write : 2 BCLK cycles |
| | | | | 01$_2$ | 2 BCLK cycles |
| | | | | 10$_2$ | 3 BCLK cycles |
| | | | | 11$_2$ | 4 BCLK cycles |
| | Multiplex bus | ——— | ——— | 00$_2$ | 3 BCLK cycle |
| | | | | 01$_2$ | 3 BCLK cycles |
| | | | | 10$_2$ | 3 BCLK cycles |
| | | | | 11$_2$ | 4 BCLK cycles |




Note 1: This timing example shows bus cycle length. Read cycle and write cycle may be continued after this bus cycle.
Note 2: Address bus and chip select may get longer depending on the state of CPU such as an instruction queue buffer.
Note 3: When accessing same external area (same $\overline{CS}$ area) continuously, chip select may output continuously.

**Figure 1.7.7. Typical bus timings using software wait**




Note 1: This timing example shows bus cycle length. Read cycle and write cycle may be continued after this bus cycle.
Note 2: Address bus and chip select may get longer depending on the state of CPU such as an instruction queue buffer.
Note 3: When accessing same external area (same $\overline{CS}$ area) continuously, chip select may output continuously.

**Figure 1.7.8. Typical bus timings using software wait**

## System Clock

### Clock Generating Circuit

The clock generating circuit contains three oscillator circuits as follows:

(1) Main clock generating circuit
(2) Sub clock generating circuit
(3) Ring oscillator (oscillation stop detect function)

Table 1.8.1 lists the clock generating circuit specifications and Table 1.8.2 lists registers controlling each clock generating circuit. Figure 1.8.1 shows block diagram of the system clock generating circuit. Figure 1.8.2 to 1.8.5 show clock control related registers.

**Table 1.8.1. The clock oscillation circuit specifications**

| Item | Main clock generating circuit | Sub clock generating circuit | Ring oscillator |
|---|---|---|---|
| Use of clock | • CPU's operating clock source<br>• Internal peripheral unit's operating clock source | • CPU's operating clock source<br>• Timer A/B's count clock source | • CPU's operating clock source when main clock frequency stops |
| Clock frequency | 0 to 30 MHz | 32.768 kHz | About 1 MHz |
| Usable oscillator | • Ceramic oscillator<br>• Crystal oscillator | • Crystal oscillator | ——— |
| Pins to connect oscillator | XIN, XOUT | XCIN, XCOUT | ——— |
| Oscillation stop/ restart function | Presence | Presence | Presence |
| Oscillator status after reset | Oscillating | Stopped | Stopped |
| Other | Externally derived clock can be input | | ——— |

**Table 1.8.2. Control registers for each clock generating circuits**

| Clock generating circuit | Control register |
|---|---|
| Main clock | System clock control register 0 (address $0006_{16}$) :CM0<br>System clock control register 1 (address $0007_{16}$) :CM1<br>Main clock divide register (address $000C_{16}$) : MCD |
| Sub clock | System clock control register 0 (address $0006_{16}$) : CM0<br>System clock control register 1 (address $0007_{16}$) :CM1 |
| Oscillation stop detect function | Oscillation stop detect register (address $000D_{16}$) : CM2 |

Note : CM0, CM1, CM2 and MCD registers are protected from a false write by program runaway. When you want to rewrite these registers, set "1" to bit 0 of protect register (address $000A_{16}$) to release protect, then rewrite the register.

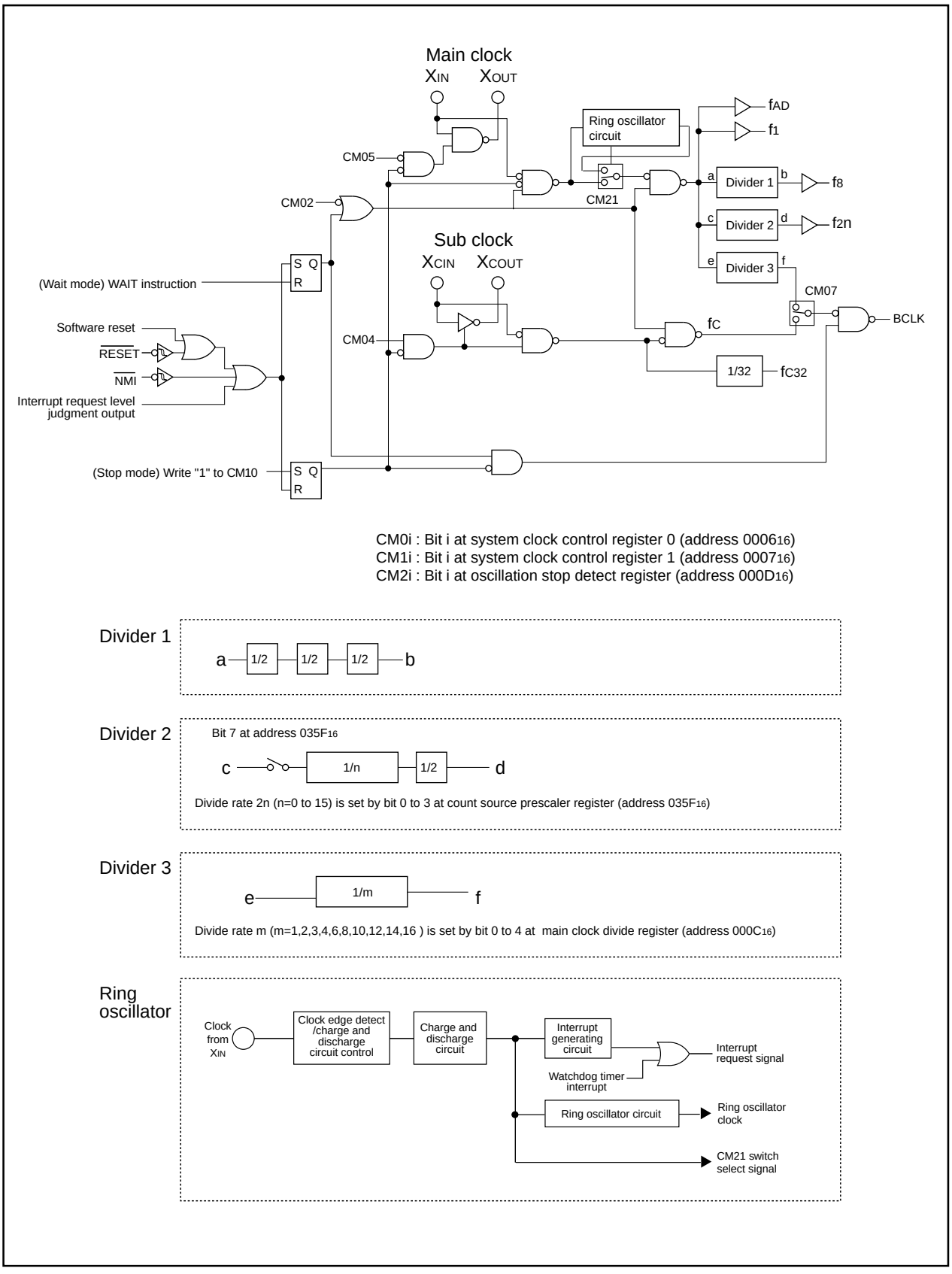
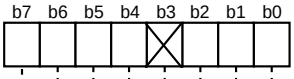



**Figure 1.8.1. Clock generating circuit**

System clock control register 0 (Note 1)




| Symbol | Address | When reset |
|---|---|---|
| CM0 | 0006$_{16}$ | 0000 X000$_2$ |

| Bit symbol | Bit | Function | R | W |
|---|---|---|---|---|
| CM00 | Clock output function select bit (Note 2) | b1 b0<br>0 0 : I/O port P5$_3$<br>0 1 : f$_C$ output<br>1 0 : f$_8$ output<br>1 1 : f$_{32}$ output | ○ | ○ |
| CM01 | | | ○ | ○ |
| CM02 | WAIT peripheral function clock stop bit | 0 : Do not stop peripheral clock in wait mode<br>1 : Stop peripheral clock in wait mode (Note 3) | ○ | ○ |
| Nothing is assigned. When write, set "0". When read, their contents are indeterminate. | | | – | – |
| CM04 | Port X$_C$ select bit | 0 : I/O port<br>1 : X$_{CIN}$-X$_{COUT}$ generation (Note 4) | ○ | ○ |
| CM05 | Main clock (X$_{IN}$-X$_{OUT}$) stop bit (Note 5) | 0 : Main clock On<br>1 : Main clock Off (Note 6) | ○ | ○ |
| CM06 | Watchdog timer function select bit | 0 : Watchdog timer interrupt<br>1 : Reset (Note 7) | ○ | ○ |
| CM07 | System clock select bit (Note 8) | 0 : X$_{IN}$, X$_{OUT}$<br>1 : X$_{CIN}$, X$_{COUT}$ | ○ | ○ |

Note 1: Set bit 0 of the protect register (address 000A$_{16}$) to “1” before writing to this register.
Note 2: The port P5$_3$ dose not function as an I/O port in microprocessor or memory expansion mode.
When outputting ALE to P5$_3$ (bits 5 and 4 of processor mode register 0 is "01"), set these bits to "00".
The port P5$_3$ function is not selected, even when you set "00" in microprocessor or memory expansion mode and bit 7 of the processor mode register 0 is "1".
Note 3: fc$_{32}$ is not included. When this bit is set to "1", PLL cannot be used in WAIT.
Note 4: When X$_{CIN}$-X$_{COUT}$ is used, set port P8$_6$ and P8$_7$ to no pull-up resistance with the input port.
Note 5: When entering the power saving mode, the main clock is stopped using this bit. To stop the main clock, set system clock stop bit (CM07) to "1" while an oscillation of sub clock is stable. Then set this bit to "1".
When X$_{IN}$ is used after returning from stop mode, set this bit to "0".
When this bit is "1", X$_{OUT}$ is "H". Also, the internal feedback resistance remains ON, so X$_{IN}$ is pulled up to X$_{OUT}$ ("H" level) via the feedback resistance.
Note 6: When the main clock is stopped, the main clock division register (address 000C$_{16}$) is set to the division by 8 mode.
However, in ring oscillator mode, the main clock division register is not set to the division by 8 mode when X$_{IN}$-X$_{OUT}$ is stopped by this bit.
Note 7: When "1" has been set once, "0" cannot be written by software.
Note 8: Set this bit "0" to "1" when sub clock oscillation is stable by setting CM04 to "1".
Set this bit "1" to "0" when main clock oscillation is stable by setting CM05 to "0".
Do not set CM04 and CM05 simultaneously.

**Figure 1.8.2. Clock control related register (1)**

System clock control register 1 (Note 1)

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

Symbol: CM1 — Address: 0007$_{16}$ — When reset: 00100000$_2$

| Bit symbol | Bit | Function | R | W |
|---|---|---|---|---|
| CM10 | All clock stop control bit (Note 2) | 0 : Clock on<br>1 : All clocks off (stop mode) (Note 3) | ○ | ○ |
| Reserved bit | | Must set to “0” | ○ | ○ |
| Reserved bit | | Must set to “1” | ○ | ○ |
| Reserved bit | | Must set to “0” | ○ | ○ |

Note 1: Set bit 0 of the protect register (address 000A$_{16}$) to “1” before writing to this register.
Note 2: When this bit is "1", XOUT is "H", and the internal feedback resistance is disabled. XCIN and XCOUT are high-inpedance.
Note 3: When all clocks are stopped (stop mode), the main clock division register (address 000C$_{16}$) is set to the division by 8 mode.

Main clock division register (Note 1)

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

Symbol: MCD — Address: 000C$_{16}$ — When reset: XXX01000$_2$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| MCD0 | Main clock division select bit (Note 2, 4) | b4 b3 b2 b1 b0<br>1 0 0 1 0 : No division mode<br>0 0 0 1 0 : Division by 2 mode<br>0 0 0 1 1 : Division by 3 mode<br>0 0 1 0 0 : Division by 4 mode<br>0 0 1 1 0 : Division by 6 mode<br>0 1 0 0 0 : Division by 8 mode<br>0 1 0 1 0 : Division by 10 mode<br>0 1 1 0 0 : Division by 12 mode<br>0 1 1 1 0 : Division by 14 mode<br>0 0 0 0 0 : Division by 16 mode | ○ | ○ |
| MCD1 | | | ○ | ○ |
| MCD2 | | | ○ | ○ |
| MCD3 | | | ○ | ○ |
| MCD4 | | | ○ | ○ |
| Nothing is assigned.<br>When write, set "0". When read, their contents are indeterminate. | | | – | – |

Note 1: Set bit 0 of the protect register (address 000A$_{16}$) to “1” before writing to this register.
Note 2: These bits are "01000$_2$" (8-division mode) when main clock is stopped or you shift to stop mode. However, in ring oscillator mode, this register is not set to the division by 8 mode when XIN-XOUT is stopped by main clock stop bit.
Note 3: Do not attempt to set combinations of values other than those shown in this figure.
Note 4: SFR area of CAN is accessed with no division mode.

**Figure 1.8.3. Clock control related registers (2)**

## Oscillation stop detect register (Note 1)

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

| Symbol | Address | When reset |
|---|---|---|
| CM2 | $000D_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CM20 | Oscillation stop detect enable bit | 0: Oscillation stop detect function disabled<br>1: Oscillation stop detect function enabled | ○ | ○ |
| CM21 | Main clock switching bit (Note 2,3) | 0: $X_{IN}$ selected<br>1: Ring oscillator selected | ○ | ○ |
| CM22 | Oscillation stop detect flag (Note 4) | 0: Ignored<br>1: Detect oscillation stop | ○ | ○ |
| CM23 | $X_{IN}$ clock monitor flag (Note 5) | 0: $X_{IN}$ oscillating<br>1: $X_{IN}$ not oscillating | ○ | – |
| —— | Reserved bit | Must set to "0" | ○ | ○ |
| —— | | | ○ | ○ |
| —— | | | ○ | ○ |
| —— | | | ○ | ○ |

Note 1: Set bit 0 of the protect register (address $000A_{16}$) to "1" before writing to this register.
Note 2: When $X_{IN}$ oscillation stop is detected in CM20="1", this bit becomes "1".
After this, although $X_{IN}$ starts oscillating, this bit does not become "0". When you change to $X_{IN}$ as system clock after $X_{IN}$ restarts oscillating, write "0" to this bit.
Note 3: When CM20="1" and CM22="1", this bit cannot be written.
Note 4: When detecting oscillation stop, this bit becomes "1". "0" can be written by software.
When "0" is written during $X_{IN}$ oscillation stop, this bit does not becomes "1" although $X_{IN}$ oscillating stops.
Note 5: $X_{IN}$ state is judged by reading this bit several times in oscillation stop interrupt process program.

**Figure 1.8.4. Clock control related register (3)**

## Count source prescale register

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

| Symbol | Address | When reset |
|---|---|---|
| TCSPR | $035F_{16}$ | $0XXX\ 0000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CNT0 | Division rate select bit | b3 b2 b1 b0<br>0 0 0 0: No-division<br>0 0 0 1: Division by 2<br>0 0 1 0: Division by 4<br>0 0 1 1: Division by 6<br>•<br>•<br>•<br>1 1 0 1: Division by 26<br>1 1 1 0: Division by 28<br>1 1 1 1: Division by 30 (Note) | ○ | ○ |
| CNT1 | | | ○ | ○ |
| CNT2 | | | ○ | ○ |
| CNT3 | | | ○ | ○ |
| —— | Nothing is assigned. When write, set to "0". When read, their contents are indeterminate. | | – | – |
| —— | | | – | – |
| —— | | | – | – |
| CST | Operation enable bit | 0: Divider stops<br>1: Divider starts | ○ | ○ |

Note : Write to these bits during the count stop.

## VDC control register for PLL (Note 1)

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

| Symbol | Address | When reset |
|---|---|---|
| PLV | $0017_{16}$ | $XXXXXX01_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PLV00 | PLL VDC enable bit (Note 2) | 0 : Cut off power to PLL<br>1 : Power to PLL | ○ | ○ |
| —— | Reserved bit | Must set to "0" | ○ | ○ |
| —— | Nothing is assigned. When write, set to "0". When read, their contents are indeterminate. | | – | – |

Note 1: When rewriting this register, set bit 3 of protect regiser (address $000A_{16}$) to "1".
Note 2: Set this bit to "0" before shifting to stop mode.

**Figure 1.8.5. Clock control related register (4)**

## (1) Main clock

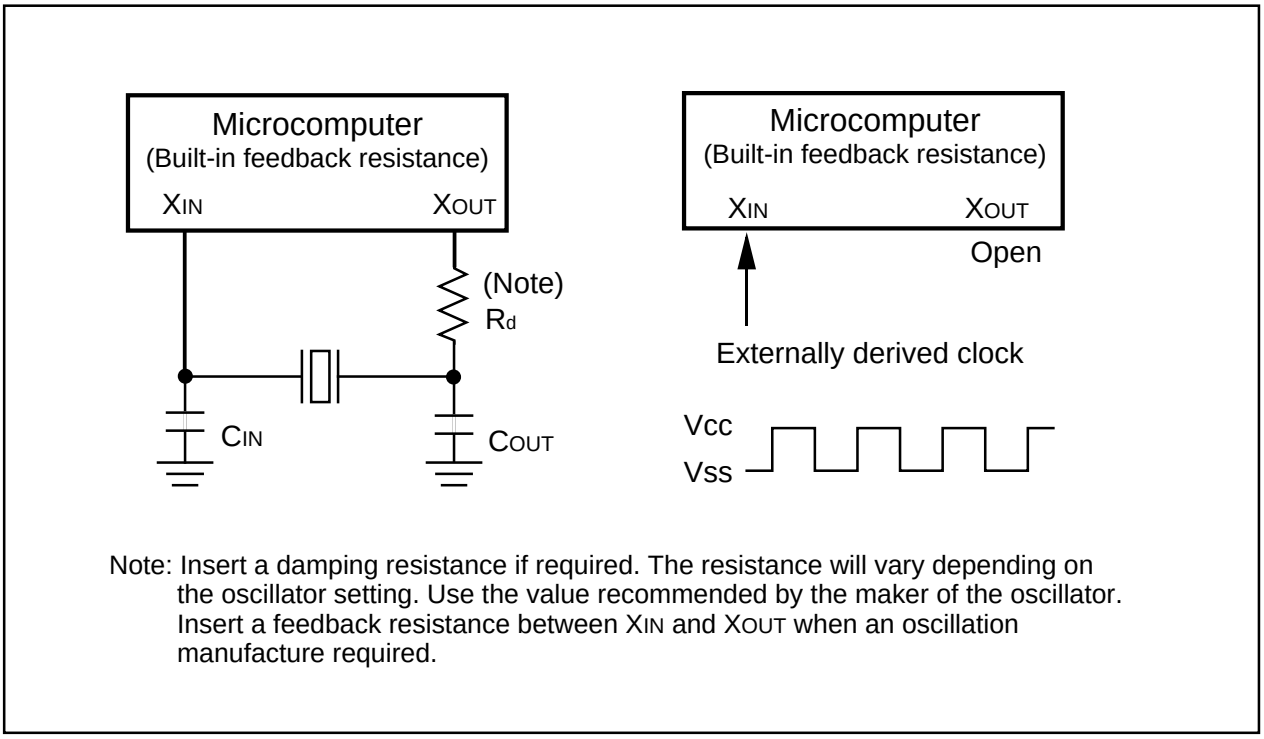
The main clock is a clock source for CPU operation and peripheral I/O. Figure 1.8.6 shows example of a main clock. When a reset, the clock oscillates and after a reset, the clock is divided by 8 to the BCLK (CPU operating clock).

**(a) Main clock On/Off function**

- Main clock ($X_{IN}$-$X_{OUT}$) stop bit of system control register 0 (bit 5 at address $0006_{16}$)
  0: Main clock On
  1: Main clock Off

Also, the clock is stopped by shifting to the stop mode.

- All clock stop control bit of system control register 1 (bit 0 at address $0007_{16}$)
  0: Clock on
  1: All clocks off (stop mode)




Note: Insert a damping resistance if required. The resistance will vary depending on the oscillator setting. Use the value recommended by the maker of the oscillator. Insert a feedback resistance between $X_{IN}$ and $X_{OUT}$ when an oscillation manufacture required.

**Figure 1.8.6. Examples of main clock**

## (2) Sub clock

The sub clock is a clock source for CPU operation and count source for timer A and B. Figure 1.8.7 shows example of sub clock. When the sub clock is used, set ports P8$_6$ and P8$_7$ to no pull-up resistance with the input port. No sub clock is generated during and after a reset.

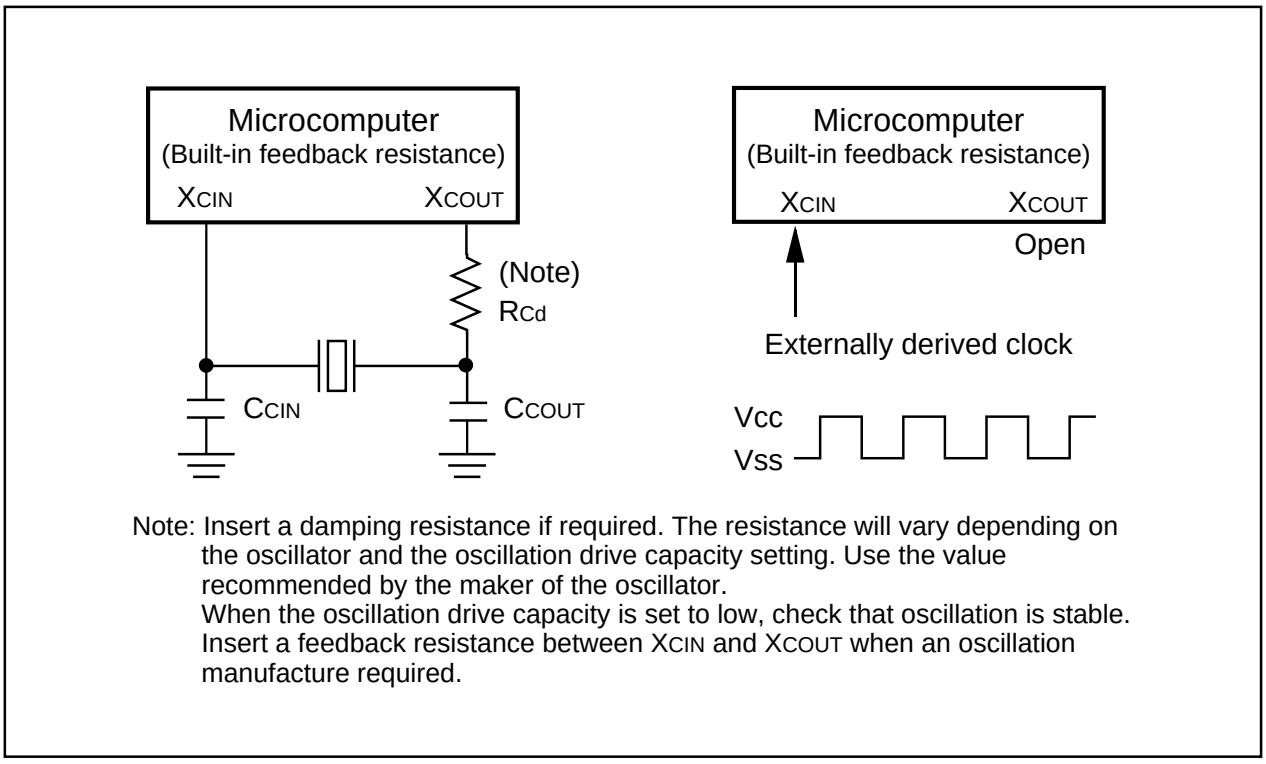
### (a) Sub clock On/Off function

When you want to use sub clock, set the following bit and sub clock enabled.

- Port Xc select bit of system control register 0 (bit 4 at address 0006$_{16}$)
  - 0: I/O port (sub clock off)
  - 1: XIN-XOUT generation (sub-clock on)

Also, shifting to the stop mode stops the clock.

- All clock stop control bit of system control register 1 (bit 0 at address 0007$_{16}$)
  - 0: Clock On
  - 1: All clock stop (stop mode)




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

**Figure 1.8.7. Examples of sub clock**

## (3) Oscillation stop detect function (OSD function)

This function monitor the main clock (XIN pin). When main clock is stopped, internal ring oscillator starts ocsillation and replace the main clock. Then oscillation stop detect interrupt process is operated.
When frequency of main clock is less or equal than 2MHz, this function does not work.

### (a) OSD function enable/disable

• OSD enable bit of oscillation stop detect register (bit 0 at address 000D16)
0: OSD function disabled
1: OSD function enabled

Set OSD enable bit (bit 0) of oscillation stop detect register to "0" to disable OSD function before setting stop mode. Stop mode is canceled before setting this bit to "1".

### (b) Operation when oscillation stop detects

1) When XIN oscillation stops, a built in ring oscillation starts as a main clock automatically.
2) OSD interrupt request is generated, jump to an address FFFFF016 to FFFFF316 allocated fixed vector table (watchdog timer interrupt vector) and execute program of jump address.
3) OSD interrupt shares vector table with watchdog timer interrupt. When using both OSD and watchdog timer interrupts, read and judge OSD flag in interrupt process routine.
OSD flag of oscillation stop detect register (bit 2 at address 000D16)
1: Oscillation stop detects
4) XIN does not become main clock although XIN On after oscillation stop detects. When you want XIN to be main clock, execute a process shown in Figure 1.8.8.

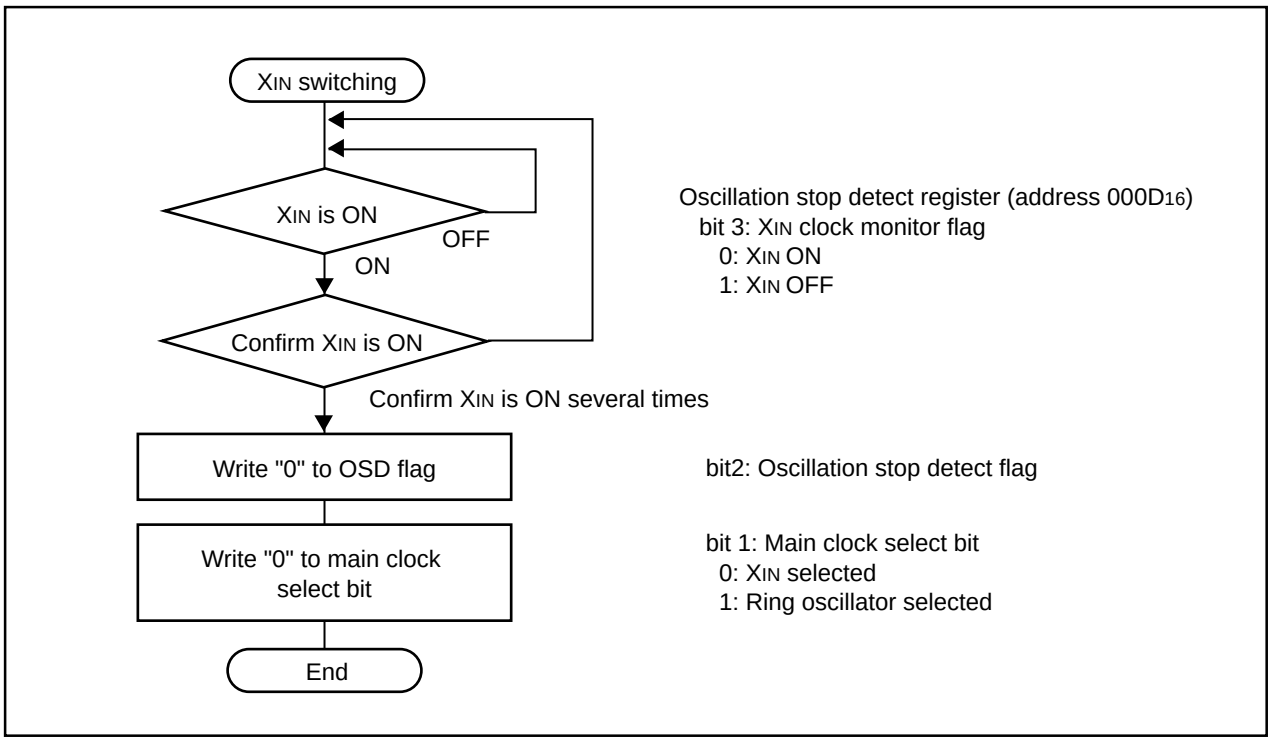



**Figure 1.8.8. Main clock switching sequence**

## CPU clock (BCLK)

Main clock, sub clock or clock from ring oscillator can be selected as clock source for BCLK.

System clock select bit of system clock control register (bit 7 at address $0006_{16}$)

0: Main clock is selected (XIN-XOUT)

1: Sub clock is selected (XCIN-XCOUT)

Main clock select bit of oscillation stop detect register (bit 1 at address $000D_{16}$)

0: Main clock is selected (XIN-XOUT)

1: Clock from ring oscillator is selected

**Table 1.8.3. BCLK source and setting bit**

| BCLK source | System clock select bit (Bit 7 of address $0006_{16}$) | Main clock select bit (Bit 1 of address $000D_{16}$) |
|---|---|---|
| Main clock (XIN-XOUT) | 0 | 0 |
| Sub clock (XCIN-XCOUT) | 1 | 0 |
| Ring oscillator | 0 | 1 |

When main clock or ring oscillator clock is selected as clock source for BCLK, the BCLK is the clock derived by dividing the main clock or ring oscillator clock by 1, 2, 3, 4, 6, 8, 10, 12, 14 or 16.

Main clock divide rate select bit of main clock division register (bit 0 to 4 at address $000C_{16}$)

The BCLK is derived by dividing the main clock (XIN-XOUT) by 8 after a reset. (Main clock division register = "$XXX01000_2$")

When main clock is stopped under changing to stop mode or selecting XIN-XOUT (main clock select bit = "0"), the main clock division register is set to the division by 8 ("$XXX01000_2$").

When ring oscillator clock is selected as clock source for BCLK, although main clock is stoped, the contents of main clock division register is maintained.

## Peripheral function clock

Main clock, sub clock, PLL clock or ring oscillator clock can be selected as clock source for peripheral function.

**(1) $f_1$, $f_8$, $f_{2n}$**

The clock is derived from the main clock or by dividing it by 1, 8 or 2n (n=1 to 15). It is used for the timer A and timer B counts and serial I/O and UART operation clock.

The $f_{2n}$ division rate is set by the count source prescaler register. Figure 1.8.5 shows the count source prescaler register.

**(2) $f_{AD}$**

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

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

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

**(4) $f_{PLL}$**

This clock is 80 MHz generated by PLL synthesizer. It is used for the intelligent I/O group 3.

## Clock Output

You can output clock from the P5$_3$ pin.

- BCLK output function select bit of processor mode register 0 (bit 7 at address 0004$_{16}$)
- ALE select bits of processor mode register 1 (bit 4 and 5 at address 0005$_{16}$)
- Clock output function select bits of system clock select register (bits 1 and 0 at address 0006$_{16}$)

Table 1.8.4 shows clock output setting (single chip mode) and Table 1.8.5 shows clock output setting (memory expansion/microprocessor mode).

**Table 1.8.4. Clock output setting (single chip mode)**

| BCLK output function select bit | Clock output function select bit | | ALE pin select bit | | P5$_3$/BCLK/ALE/CLKOUT pin function |
|---|---|---|---|---|---|
| PM07 | CM01 | CM00 | PM15 | PM14 | |
| Ignored | 0 | 0 | Ignored | Ignored | P5$_3$ I/O port |
| 1 | 0 | 1 | Ignored | Ignored | fc output (Note) |
| 1 | 1 | 0 | Ignored | Ignored | f$_8$ output (Note) |
| 1 | 1 | 1 | Ignored | Ignored | f$_{32}$ output (Note) |

Note :Must use P5$_7$ as input port.

**Table 1.8.5. Clock output setting (memory expansion/microprocessor mode)**

| BCLK output function select bit | Clock output function select bit | | ALE pin select bit | | P5$_3$/BCLK/ALE/CLKOUT pin function |
|---|---|---|---|---|---|
| PM07 | CM01 | CM00 | PM15 | PM14 | |
| 0 | 0 | 0 | "0, 0" "1, 0" "1, 1" | | BCLK output |
| 1 | 0 | 0 | | | "L" output (not P5$_3$) |
| 1 | 0 | 1 | | | fc output |
| 1 | 1 | 0 | | | f$_8$ output |
| 1 | 1 | 1 | | | f$_{32}$ output |
| Ignored | 0 | 0 | 0 | 1 | ALE output |

Note: The processor mode register 0 and 1 are protected from false write by program run away. Set bit 1 to "1" at protect register (address 000A$_{16}$) and release protect before rewriting processor mode register 0 and 1.

## Power Saving

There are three power save modes. Figure 1.8.9 shows the clock transition between each of the three modes, (1), (2), and (3).

- Normal operating mode
  CPU and peripheral function operate when supplying clock. Power dissipation is reduced by making BCLK slow.
- Wait mode
  BCLK is stopped. Peripheral function clock is stopped as desired. Main clock and sub clock isn't stopped. Power dissipation is reduced than normal operating mode.
- Stop mode (Note 1)
  Main clock, sub clock and PLL synthesizer are stopped. CPU and peripheral function clock are stopped. Power dissipation is the most few in this mode.
  Note :When using stop mode, oscillation stop detect function must be canceled.

### (1) Normal operating mode

**High-speed mode**

Main clock one cycle forms CPU operating clock.

**Medium-speed mode**

The main clock divided into 2, 3, 4, 6, 8, 10, 12, 14, or 16 forms CPU operating clock.

**Low-speed mode**

Subclock (fc) forms CPU operating clock.

**Low power-dissipation mode**

This mode is selected when the main clock is stopped from low-speed mode. Only the peripheral functions for which the subclock was selected as the count source continue to run.

**Ring oscillator mode**

The ring oscillator clock divided into 2, 3, 4, 6, 8, 10, 12, 14, or 16 forms CPU operating clock.

**Ring oscillator low power-dissipation mode**

This mode is selected when the main clock is stopped from low-speed mode.

When switching BCLK from ring oscillator to main clock, switch clock after main clock oscillates fully stable. After setting divided by 8 (main clock division register =$08_{16}$) in ring oscilltor mode, switching to the middle mode (divided by 8) is recommended.

### (2) Wait mode

In wait mode, BCLK is stopped and CPU and watchdog timer operated by BCLK are halted. The main clock, subclock and ring oscillator clock continue to run.

**(a) Shifting to wait mode**

Execute WAIT instruction.

**(b) Peripheral function clock stop function**

The $f_1$, $f_8$ and $f_{2n}$ being supplied to the internal peripheral functions stops. The internal peripheral functions operated by the clock stop.

WAIT peripheral function clock stop bit of system clock control register 0 (bit 2 at address $0006_{16}$)
0: Do not stop $f_1$, $f_8$ and $f_{2n}$ in wait mode and do not stop supplying clock to PLL circuit
1: Stop $f_1$, $f_8$ and $f_{2n}$ in wait mode and stop supplying clock to PLL circuit

**(c) The status of the ports in wait mode**

Table 1.8.6 shows the status of the ports in wait mode.

**(d) Exit from wait mode**

Wait mode is cancelled by a hardware reset or interrupt. If a peripheral function interrupt is used to cancel wait mode, set the following registers.

Interrupt priority set bits for exiting a stop/wait state of exit priority register (bits 0 to 2 at address $009F_{16}$) :RLVL0 to RLVL2
Set the same level as the flag register (FLG) processor interrupt level (IPL).
Interrupt priority set bits of interrupt control register (bits 0 to 2)
Set to a priority level above the level set by RLVL0 to RLVL2 bits
Interrupt enable flag of FLG register
I = 1

When using an interrupt to exit Wait mode, the microcomputer resumes operating the clock that was operating when the WAIT command was executed as BCLK from the interrupt routine.

**Table 1.8.6. Port status during wait mode**

| Pin | | Memory expansion mode<br>Microprocessor mode | Single-chip mode |
|---|---|---|---|
| Address bus, data bus, $\overline{CS0}$ to $\overline{CS3}$, $\overline{BHE}$ | | Retains status before wait mode | |
| $\overline{RD}$, $\overline{WR}$, $\overline{WRL}$, $\overline{WRH}$, $\overline{DW}$, $\overline{CASL}$, $\overline{CASH}$ | | "H" (Note) | |
| $\overline{RAS}$ | | "H" (Note) | |
| $\overline{HLDA}$,BCLK | | "H" | |
| ALE | | "L" | |
| Port | | Retains status before wait mode | |
| $CLK_{OUT}$ | When $f_C$ selected | Does not stop | |
| | When $f_8$, $f_{32}$ selected | Does not stop when the WAIT peripheral function clock stop bit is "0". When the WAIT peripheral function clock stop bit is "1", the status immediately prior to entering wait mode is maint ained. | |

Note :When self-refresh is done in operating DRAM control, $\overline{CAS}$ and $\overline{RAS}$ becomes "L".

## (3) Stop mode

All oscillation, main clock, subclock, and PLL synthesizer stop in this mode. Because the oscillation of BCLK and peripheral clock stops in stop mode, peripheral functions such as the A-D converter, timer A and B, serial I/O, intelligent I/O and watchdog timer do not function.

The content of the internal RAM is retained provided that $V_{CC}$ remains above 2.5V.

When changing to stop mode, the main clock division register ($000C_{16}$) is set to "$XXX01000_2$" (division by 8 mode).

### (a) Changing to stop mode

All clock stop control bit of system clock control register 1 (bit 0 at address $0007_{16}$)

0: Clock ON

1: All clocks off (stop mode)

Before changing to stop mode, set bit 7 of PLL control register 0 (address $0376_{16}$) to "0" to stop PLL. Also, set bit 0 of VDC control register for PLL (address $0017_{16}$) to "1" to turn PLL circuit power off.

### (b) The status of the ports in stop mode

Table 1.8.7 shows the status of the ports in stop mode.

### (c) Exit from stop mode

Stop mode is cancelled by a hardware reset or interrupt. If a peripheral function interrupt is used to cancel stop mode, set the following registers.

- Interrupt priority set bits for exiting a stop/wait state of exit priority register (bits 0 to 2 at address $009F_{16}$) :RLVL0 to RLVL2
  Set the same level as the flag register (FLG) processor interrupt level (IPL).
- Interrupt priority set bits of interrupt control register (bits 0 to 2)
  Set to a priority level above the level set by RLVL0 to RLVL2 bits
- Interrupt enable flag of FLG register
  I = 1

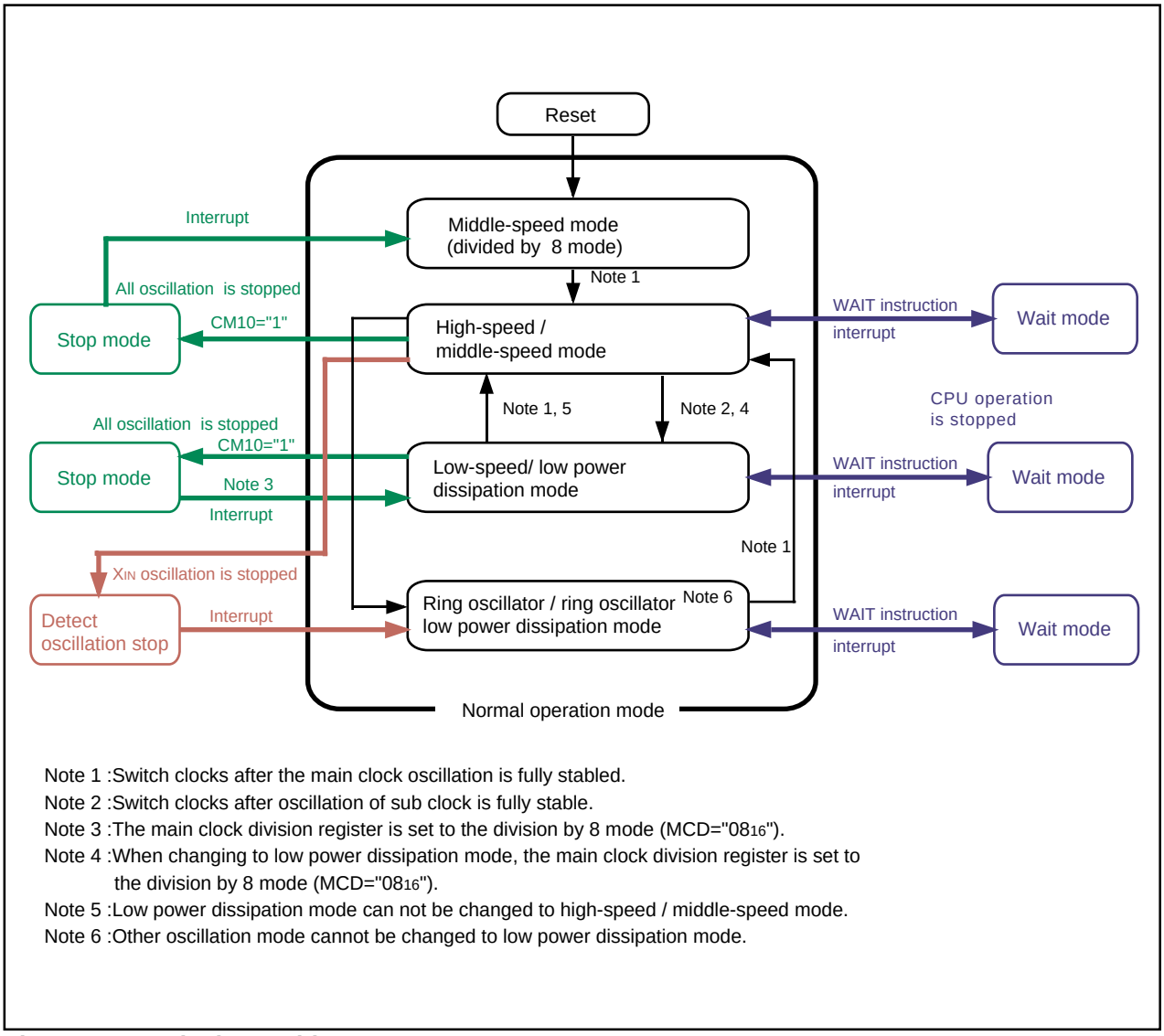
When exiting from stop mode using peripheral interrupt request, CPU operates the following BCLK and the relevant interrupt routine is executed.

- When subclock was set as BCLK before changing to stop mode, subclock is set to BCLK after cancelled stop mode
- When main clock was set as BCLK before changing to stop mode, the main clock division by 8 is set to BCLK after cancelled stop mode.

**Table 1.8.7. Port status during stop mode**

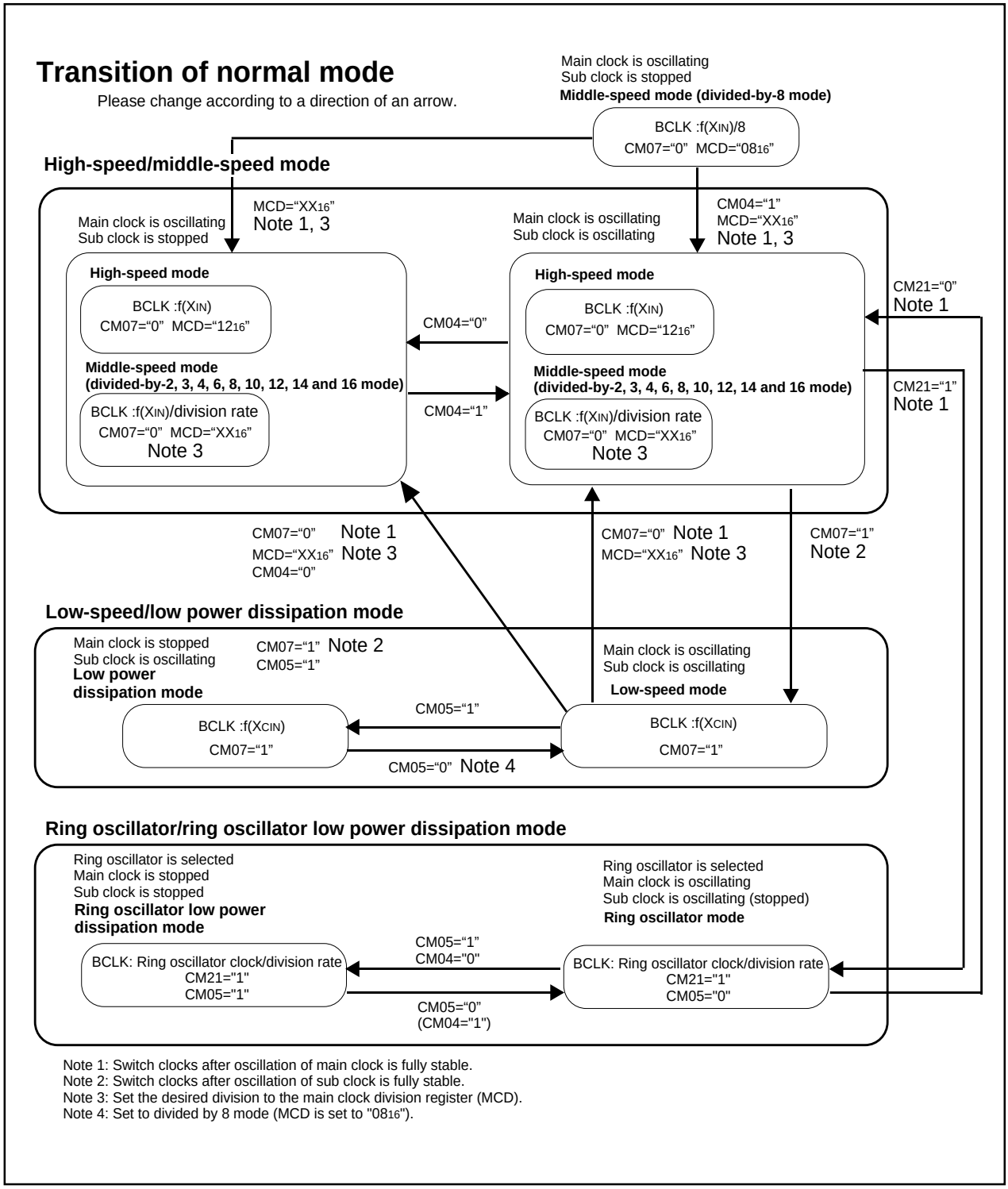
| Pin | | Memory expansion mode<br>Microprocessor mode | Single-chip mode |
|---|---|---|---|
| Address bus, data bus, $\overline{CS0}$ to $\overline{CS3}$, $\overline{BHE}$ | | Retains status before stop mode | |
| $\overline{RD}$, $\overline{WR}$, $\overline{WRL}$, $\overline{WRH}$, $\overline{DW}$, $\overline{CASL}$, $\overline{CASH}$ | | "H" (Note) | |
| $\overline{RAS}$ | | "H" (Note) | |
| $\overline{HLDA}$, BCLK | | "H" | |
| ALE | | "H" | |
| Port | | Retains status before stop mode | |
| $CLK_{OUT}$ | When fc selected | "H" | |
| | When $f_8$, $f_{32}$ selected | Retains status before stop mode | |

Note :When self-refresh is done in operating DRAM control, $\overline{CAS}$ and $\overline{RAS}$ becomes "L".




Note 1 :Switch clocks after the main clock oscillation is fully stabled.
Note 2 :Switch clocks after oscillation of sub clock is fully stable.
Note 3 :The main clock division register is set to the division by 8 mode (MCD="08$_{16}$").
Note 4 :When changing to low power dissipation mode, the main clock division register is set to the division by 8 mode (MCD="08$_{16}$").
Note 5 :Low power dissipation mode can not be changed to high-speed / middle-speed mode.
Note 6 :Other oscillation mode cannot be changed to low power dissipation mode.

**Figure 1.8.9. Clock transition**




Note 1: Switch clocks after oscillation of main clock is fully stable.
Note 2: Switch clocks after oscillation of sub clock is fully stable.
Note 3: Set the desired division to the main clock division register (MCD).
Note 4: Set to divided by 8 mode (MCD is set to "0816").

**Figure 1.8.10. Clock transition**

## Protection

The protection function is provided so that the values in important registers cannot be changed in the event that the program runs out of control. Figure 1.8.11 shows the protect register. The following registers are protected by the protect register.

**(1) Registers protected by PRC0 (bit 0)**

- System clock control registers 0 and 1 (addresses $0006_{16}$ and $0007_{16}$)
- Main clock division register (address $000C_{16}$)
- Oscillation stop detect register (address $000D_{16}$)
- PLL control register 0 (address $0376_{16}$)

**(2) Registers protected by PRC1 (bit 1)**

- Processor mode registers 0 and 1 (addresses $0004_{16}$ and $0005_{16}$)
- Three-phase PWM control registers 0 and 1 (addresses $0308_{16}$ and $0309_{16}$)

**(3) Registers protected by PRC2 (bit 2)**

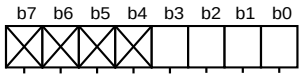
- Port P9 direction register (address $03C7_{16}$)
- Function select register A3 (address $03B5_{16}$)

**(4) Registers protected by PRC3 (bit 3)**

- VDC control register for PLL (address $0017_{16}$)
- VDC control register 0 (address $001B_{16}$)
- VDC control register 1 (address $001F_{16}$)

If, after “1” (write-enabled) has been written to the PRC2, a value is written to any address, the bit automatically reverts to “0” (write-inhibited). Change port P9 input/output and function select register A3 immediately after setting "1" to PRC2. Interrupt and DMA transfer should not be inserted between instructions. However, the PRC0, PRC1 and PRC3 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: 000A16
When reset: XXXX00002

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PRC0 | Protect bit 0 | Enables writing to system clock control registers 0 and 1 (addresses 000616 and 000716), main clock division register (address 000C16), oscillation stop detect register (address 000D16) and PLL control register 0 (address 037616)<br>0 : Write-inhibited<br>1 : Write-enabled | ○ | ○ |
| PRC1 | Protect bit 1 | Enables writing to processor mode registers 0 and 1 (addresses 000416 and 000516) and three-phase PWM control register 0 and 1 (addresses 030816 and 030916)<br>0 : Write-inhibited<br>1 : Write-enabled | ○ | ○ |
| PRC2 | Protect bit 2 (Note 1) | Enables writing to port P9 direction register ( address 03C716) and function select register A3 (address 03B516)<br>0 : Write-inhibited<br>1 : Write-enabled | ○ | ○ |
| PRC3 | Protect bit 3 | Enables writing to VDC control register for PLL ( address 001716), VDC control register 0 and 1 ( addresses 001F16 and 001B16)<br>0 : Write-inhibited<br>1 : Write-enabled (Note 2) | ○ | ○ |
| Nothing is assigned.<br>When write, set “0”. When read, their contents are indeterminate. | | | – | – |

Note 1: 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.
Note 2: User cannot use. Writing to VDC control registers 0 and 1 (addresses 001F16, 001B16) is enabled so that a careful handling is required.

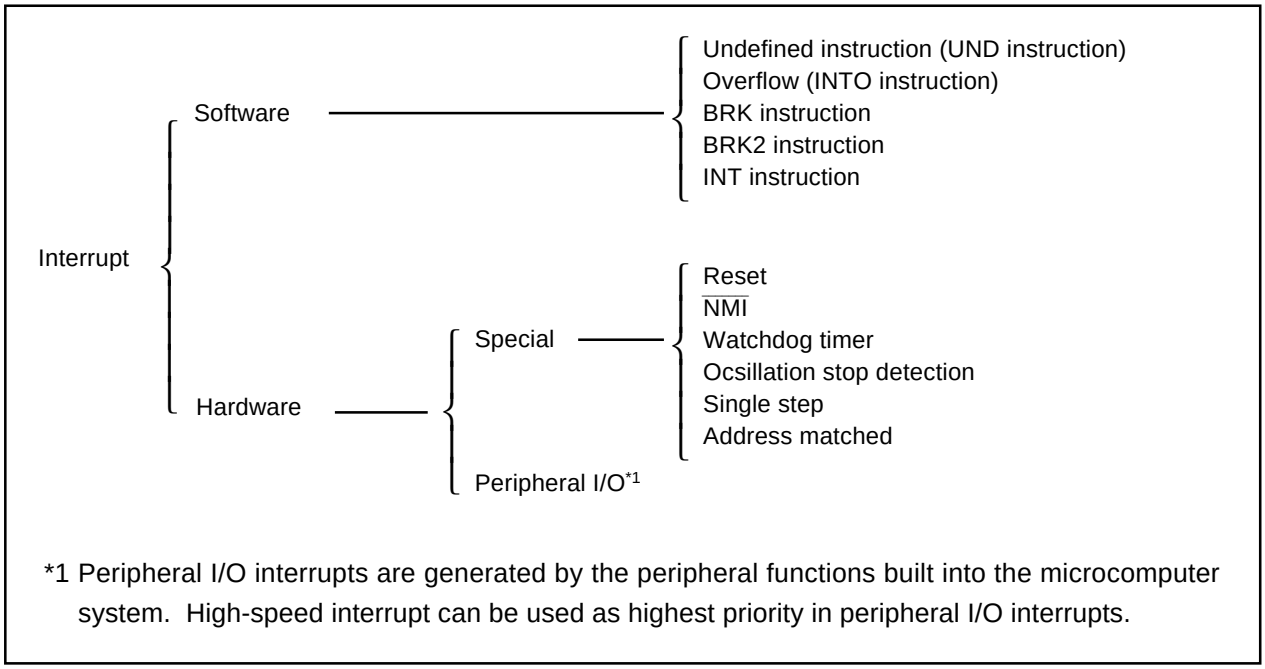
**Figure 1.8.11. Protect register**

## Interrupt Outline

### Types of Interrupts

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

Figure 1.9.1 lists the types of interrupts.




*1 Peripheral I/O interrupts are generated by the peripheral functions built into the microcomputer system. High-speed interrupt can be used as highest priority in peripheral I/O interrupts.

**Figure 1.9.1. Classification of interrupts**

### Software Interrupts

Software interrupts are generated by some instruction that generates an interrupt request when executed. Software interrupts are nonmaskable interrupts.

**(1) Undefined-instruction interrupt**

This interrupt occurs when the UND instruction is executed.

**(2) Overflow interrupt**

This interrupt occurs if the INTO instruction is executed when the O flag is 1.

The following lists the instructions that cause the O flag to change:

ABS, ADC, ADCF, ADD, ADDX, CMP, CMPX, DIV, DIVU, DIVX, NEG, RMPA, SBB, SCMPU, SHA, SUB, SUBX

**(3) BRK interrupt**

This interrupt occurs when the BRK instruction is executed.

**(4) BRK2 interrupt**

This interrupt occurs when the BRK2 instruction is executed. This interrupt is used exclusively for debugger purposes. You normally do not need to use this interrupt.

**(5) INT instruction interrupt**

This interrupt occurs when the INT instruction is executed after specifying a software interrupt number from 0 to 63. Note that software interrupt numbers 7 to 54 and 57 are assigned to peripheral I/O interrupts. This means that by executing the INT instruction, you can execute the same interrupt routine as used in peripheral I/O interrupts.

The stack pointer used in INT instruction interrupt varies depending on the software interrupt number. For software interrupt numbers 0 to 31, the U flag is saved when an interrupt occurs and the U flag is cleared to 0 to choose the interrupt stack pointer (ISP) before executing the interrupt sequence. The previous U flag before the interrupt occurred is restored when control returns from the interrupt routine. For software interrupt numbers 32 to 63, such stack pointer switchover does not occur.

However, in peripheral I/O interrupts, the U flag is saved when an interrupt occurs and the U flag is cleared to 0 to choose ISP.

Therefore movement of U flag is different by peripheral I/O interrupt or INT instruction in software interrupt number 32 to 54 and 57.

## Hardware Interrupts

There are Two types of hardware Interrupts; special interrupts and Peripheral I/O interrupts.

**(1) Special interrupts**

Special interrupts are nonmaskable interrupts.

- **Reset**
  A reset occurs when the $\overline{\text{RESET}}$ pin is pulled low.
- **$\overline{\text{NMI}}$ interrupt**
  This interrupt occurs when the $\overline{\text{NMI}}$ pin is pulled low.
- **Watchdog timer interrupt**
  This interrupt is caused by the watchdog timer.
- **Ocsillation stop detect interrupt**
  This interrupt is caused by the ocsillation stop detect function.
  It occurs when detecting the $X_{IN}$ ocsillation is stopped.
- **Single-step interrupt**
  This interrupt is used exclusively for debugger purposes. These interrupts normally do not need to use this interrupt. A single-step interrupt occurs when the D flag is set (= 1); in this case, an interrupt is generated each time an instruction is executed.
- **Address-match interrupt**
  This interrupt occurs when the program's execution address matches the contents of the address match register while the address match interrupt enable bit is set (= 1).
  This interrupt does not occur if any address other than the start address of an instruction is set in the address match register.

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

A peripheral I/O interrupt is generated by one of the 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 7 through 54 and 57 the INT instruction uses. Peripheral I/O interrupts are maskable interrupts.

- **UART related interrupt (UART0 to 4)**
  - UART transmission/NACK interrupt
  - UART reception/ACK interrupt
  - Bus collision detection, start/stop condition detection interrupts

  This is an interrupt that the serial I/O bus collision detection generates. When $I^2C$ mode is selected, start, stop condition interrupt is selected.
- **DMA0 through DMA3 interrupts**
- **Key-input interrupt**

  A key-input interrupt occurs if an “L” is input to the $\overline{KI}$ pin.
- **A-D conversion interrupt (AD0, 1)**
- **Timer A interrupt (TA0 to 4)**
- **Timer B interrupt (TB0 to 5)**
- **$\overline{INT}$ interrupt ($\overline{INT0}$ to $\overline{INT5}$ )**

  An $\overline{INT}$ interrupt selects an edge sense or a level sense. In edge sense, an $\overline{INT}$ interrupt occurs if either a rising edge or a falling edge is input to the $\overline{INT}$ pin. In level sense, an $\overline{INT}$ interrupt occurs if either a "H" level or a "L" level is input to the $\overline{INT}$ pin.
- **Intelligent I/O interrupt**
- **CAN interrupt**

## High-speed interrupts

High-speed interrupts are interrupts in which the response is executed at 5 cycles and the return is 3 cycles.

When a high-speed interrupt is received, the flag register (FLG) and program counter (PC) are saved to the save flag register (SVF) and save PC register (SVP) and the program is executed from the address shown in the vector register (VCT).

Execute an FREIT instruction to return from the high-speed interrupt routine.

High-speed interrupts can be set by setting “1” in the high-speed interrupt specification bit allocated to bit 3 of the exit priority register. Setting “1” in the high-speed interrupt specification bit makes the interrupt set to level 7 in the interrupt control register a high-speed interrupt.

You can only set one interrupt as a high-speed interrupt. When using a high-speed interrupt, do not set multiple interrupts as level 7 interrupts. When using high speed interrupt, DMA II cannot be used.

The interrupt vector for a high-speed interrupt must be set in the vector register (VCT).

When using a high-speed interrupt, you can use a maximum of two DMAC channels.

The execution speed is improved when register bank 1 is used with high speed interrupt register selected by not saving registers to the stack but to the switching register bank. In this case, switch register bank mode for high-speed interrupt routine.

## Interrupts and Interrupt Vector Tables

If an interrupt request is accepted, a program branches to the interrupt routine set in the interrupt vector table. Set the first address of the interrupt routine in each vector table. Figure 1.9.2 shows the format for specifying the address.

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

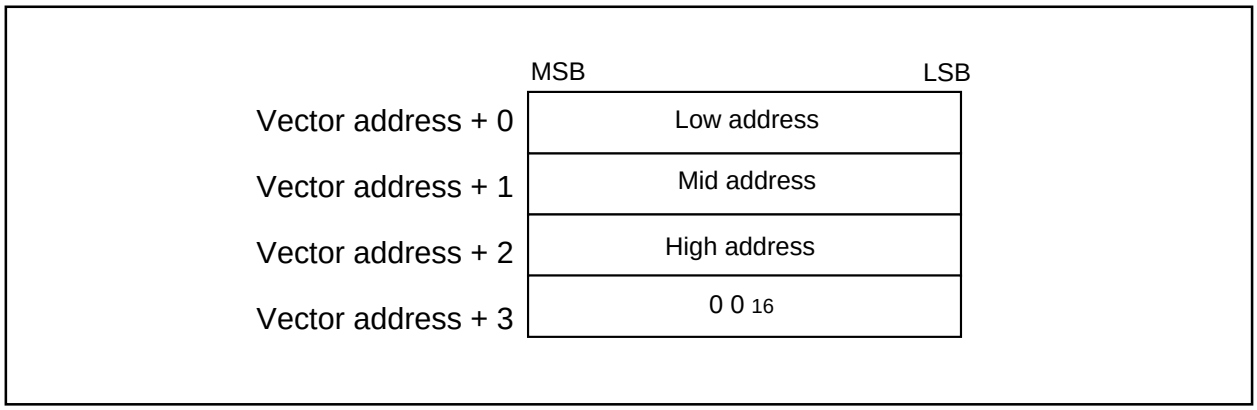



**Figure 1.9.2. Format for specifying interrupt vector addresses**

**• 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 FFFFDC16 to FFFFFF16. Each vector comprises four bytes. Set the first address of interrupt routine in each vector table. Table 1.9.1 shows the interrupts assigned to the fixed vector tables and addresses of vector tables.

**Table 1.9.1. Interrupt factors (fixed interrupt vector addresses)**

| Interrupt source | Vector table addresses Address (L) to address (H) | Remarks |
|---|---|---|
| Undefined instruction | FFFFDC16 to FFFFDF16 | Interrupt on UND instruction |
| Overflow | FFFFE016 to FFFFE316 | Interrupt on INTO instruction |
| BRK instruction | FFFFE416 to FFFFE716 | If contents of FFFFE716 is filled with FF16, program execution starts from the address shown by the vector in the relocatable vector table |
| Address match | FFFFE816 to FFFFEB16 | There is an address-matching interrupt enable bit |
| Watchdog timer interrupt | FFFFF016 to FFFFF316 | Share it with watchdog timer and oscillation stop detect |
| $\overline{\text{NMI}}$ | FFFFF816 to FFFFFB16 | External interrupt by input to $\overline{\text{NMI}}$ pin |
| Reset | FFFFFC16 to FFFFFF16 | |

**• Vector table dedicated for emulator**

Table 1.9.2 shows interrupt vector address, which is vector table register dedicated for emulator (address $000020_{16}$ to $000022_{16}$). These instructions are not effected with interrupt enable flag (I flag) (non maskable interrupt).

This interrupt is used exclusively for debugger purposes. You normally do not need to use this interrupt. Do not access the interrupt vector table register dedicated for emulator (address $000020_{16}$ to $000022_{16}$).

**Table 1.9.2. Interrupt vector table register for emulator**

| Interrupt source | Vector table addresses<br>Address (L) to address (H) | Remarks |
|---|---|---|
| BRK2 instruction | Interrupt vector table register for emulator<br>$000020_{16}$ to $000022_{16}$ | Interrupt for debugger |
| Single step | | |

**• Relocatable vector tables**

The addresses in the relocatable 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 relocatable vector tables. One vector table comprises four bytes. Set the first address of the interrupt routine in each vector table. Table 1.9.3 shows the interrupts assigned to the relocatable vector tables and addresses of vector tables.

Set an even address to the start address of vector table setting in INTB so that operating efficiency is increased.

**Table 1.9.3. Interrupt causes (variable interrupt vector addresses) (1/2)**

| Software interrupt number | Vector table address Address(L)to address(H) (Note 1) | Interrupt source |
|---|---|---|
| Software interrupt number 0 (Note 2) | +0 to +3 ($0000_{16}$ to $0003_{16}$) | BRK instruction |
| | | |
| Software interrupt number 7 | +28 to +31 ($001C_{16}$ to $001F_{16}$) | A-D channel 1 |
| Software interrupt number 8 | +32 to +35 ($0020_{16}$ to $0023_{16}$) | DMA0 |
| Software interrupt number 9 | +36 to +39 ($0024_{16}$ to $0027_{16}$) | DMA1 |
| Software interrupt number 10 | +40 to +43 ($0028_{16}$ to $002B_{16}$) | DMA2 |
| Software interrupt number 11 | +44 to +47 ($002C_{16}$ to $002F_{16}$) | DMA3 |
| Software interrupt number 12 | +48 to +51 ($0030_{16}$ to $0033_{16}$) | Timer A0 |
| Software interrupt number 13 | +52 to +55 ($0034_{16}$ to $0037_{16}$) | Timer A1 |
| Software interrupt number 14 | +56 to +59 ($0038_{16}$ to $003B_{16}$) | Timer A2 |
| Software interrupt number 15 | +60 to +63 ($003C_{16}$ to $003F_{16}$) | Timer A3 |
| Software interrupt number 16 | +64 to +67 ($0040_{16}$ to $0043_{16}$) | Timer A4 |
| Software interrupt number 17 | +68 to +71 ($0044_{16}$ to $0047_{16}$) | UART0 transmit/NACK (Note 3) |
| Software interrupt number 18 | +72 to +75 ($0048_{16}$ to $004B_{16}$) | UART0 receive/ACK (Note 3) |
| Software interrupt number 19 | +76 to +79 ($004C_{16}$ to $004F_{16}$) | UART1 transmit/NACK (Note 3) |
| Software interrupt number 20 | +80 to +83 ($0050_{16}$ to $0053_{16}$) | UART1 receive/ACK (Note 3) |
| Software interrupt number 21 | +84 to +87 ($0054_{16}$ to $0057_{16}$) | Timer B0 |
| Software interrupt number 22 | +88 to +91 ($0058_{16}$ to $005B_{16}$) | Timer B1 |
| Software interrupt number 23 | +92 to +95 ($005C_{16}$ to $005F_{16}$) | Timer B2 |
| Software interrupt number 24 | +96 to +99 ($0060_{16}$ to $0063_{16}$) | Timer B3 |
| Software interrupt number 25 | +100 to +103 ($0064_{16}$ to $0067_{16}$) | Timer B4 |
| Software interrupt number 26 | +104 to +107 ($0068_{16}$ to $006B_{16}$) | INT5 |
| Software interrupt number 27 | +108 to +111 ($006C_{16}$ to $006F_{16}$) | INT4 |
| Software interrupt number 28 | +112 to +115 ($0070_{16}$ to $0073_{16}$) | INT3 |
| Software interrupt number 29 | +116 to +119 ($0074_{16}$ to $0077_{16}$) | INT2 |
| Software interrupt number 30 | +120 to +123 ($0078_{16}$ to $007B_{16}$) | INT1 |
| Software interrupt number 31 | +124 to +127 ($007C_{16}$ to $007F_{16}$) | INT0 |
| Software interrupt number 32 | +128 to +131 ($0080_{16}$ to $0083_{16}$) | Timer B5 |
| Software interrupt number 33 | +132 to +135 ($0084_{16}$ to $0087_{16}$) | UART2 transmit/NACK (Note 3) |
| Software interrupt number 34 | +136 to +139 ($0088_{16}$ to $008B_{16}$) | UART2 receive/ACK (Note 3) |
| Software interrupt number 35 | +140 to +143 ($008C_{16}$ to $008F_{16}$) | UART3 transmit/NACK (Note 3) |
| Software interrupt number 36 | +144 to +147 ($0090_{16}$ to $0093_{16}$) | UART3 receive/ACK (Note 3) |
| Software interrupt number 37 | +148 to +151 ($0094_{16}$ to $0097_{16}$) | UART4 transmit/NACK (Note 3) |
| Software interrupt number 38 | +152 to +155 ($0098_{16}$ to $009B_{16}$) | UART4 receive/ACK (Note 3) |
| Software interrupt number 39 | +156 to +159 ($009C_{16}$ to $009F_{16}$) | Bus collision detection, start/stop condition detection (UART2)(Note 3) |
| Software interrupt number 40 | +160 to +163 ($00A0_{16}$ to $00A3_{16}$) | Bus collision detection, start/stop condition detection (UART3/UART0)(Note 3) |
| Software interrupt number 41 | +164 to +167 ($00A4_{16}$ to $00A7_{16}$) | Bus collision detection, start/stop condition detection (UART4/UART1)(Note 3) |
| Software interrupt number 42 | +168 to +171 ($00A8_{16}$ to $00AB_{16}$) | A-D channel 0 |
| Software interrupt number 43 | +172 to +175 ($00AC_{16}$ to $00AF_{16}$) | Key input interrupt |
| Software interrupt number 44 | +176 to +179 ($00B0_{16}$ to $00B3_{16}$) | Intelligent I/O interrupt 0 |
| Software interrupt number 45 | +180 to +183 ($00B4_{16}$ to $00B7_{16}$) | Intelligent I/O interrupt 1 |
| Software interrupt number 46 | +184 to +187 ($00B8_{16}$ to $00BB_{16}$) | Intelligent I/O interrupt 2 |
| Software interrupt number 47 | +188 to +191 ($00BC_{16}$ to $00BF_{16}$) | Intelligent I/O interrupt 3 |
| Software interrupt number 48 | +192 to +195 ($00C0_{16}$ to $00C3_{16}$) | Intelligent I/O interrupt 4 |
| Software interrupt number 49 | +196 to +199 ($00C4_{16}$ to $00C7_{16}$) | Intelligent I/O interrupt 5 |
| Software interrupt number 50 | +200 to +203 ($00C8_{16}$ to $00CB_{16}$) | Intelligent I/O interrupt 6 |
| Software interrupt number 51 | +204 to +207 ($00CC_{16}$ to $00CF_{16}$) | Intelligent I/O interrupt 7 |
| Software interrupt number 52 | +208 to +211 ($00D0_{16}$ to $00D3_{16}$) | Intelligent I/O interrupt 8 |
| Software interrupt number 53 | +212 to +215 ($00D4_{16}$ to $00D7_{16}$) | Intelligent I/O interrupt 9/CAN interrupt 0 |
| Software interrupt number 54 | +216 to +219 ($00D8_{16}$ to $00DB_{16}$) | Intelligent I/O interrupt 10/CAN interrupt 1 |

**Table 1.9.3. Interrupt causes (variable interrupt vector addresses) (2/2)**

| Software interrupt number | Vector table address Address(L)to address(H) (Note 1) | Interrupt source |
|---|---|---|
| Software interrupt number 55 | +220 to +223 (00DC16 to 00DF16) | Softwea interrupt |
| Software interrupt number 56 | +224 to +227 (00E016 to 00E316) | Softwea interrupt |
| Software interrupt number 57 | +228 to +231 (00E416 to 00E716) | Intelligent I/O interrupt 11/CAN interrupt 2 |
| Software interrupt number 58 (Note 2) to Software interrupt number 63 | +232 to +235 (00E816 to 00EB16) to +252 to +255 (00FC16 to 00FF16) | Softwea interrupt |

Note 1: Address relative to address in interrupt table register (INTB).

Note 2: Cannot be masked by I flag.

Note 3: When IIC mode is selected, NACK/ACK, start/stop condition detection interrupts are selected. The fault error interrupt is selected when $\overline{SS}$ pin is selected.

## Interrupt request reception

The following lists the conditions under which an interrupt request is acknowledged:

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

The interrupt enable flag (I flag), the processor interrupt priority level (IPL), interrupt request bit and interrupt priority level select bit are all independent of each other, so they do not affect any other bit. There are I flag and IPL in flag register (FLG). This flag and bit are described below.

### Interrupt Enable Flag (I Flag) and processor Interrupt Priority Level (IPL)

I flag is used to disable/enable maskable interrupts. When this flag is set (= 1), all maskable interrupts are enabled; when the flag is cleared to 0, they are disabled. This flag is automatically cleared to 0 after a reset.

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.

Table 1.9.4 shows interrupt enable levels in relation to the processor interrupt priority level (IPL).

**Table 1.9.4. IPL and Interrupt Enable Levels**

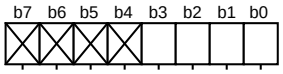
| Processor interrupt priority level (IPL) | | | Enabled interrupt priority levels |
|---|---|---|---|
| $IPL_2$ | $IPL_1$ | $IPL_0$ | |
| 0 | 0 | 0 | Interrupt levels 1 and above are enabled. |
| 0 | 0 | 1 | Interrupt levels 2 and above are enabled. |
| 0 | 1 | 0 | Interrupt levels 3 and above are enabled.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

## Interrupt control registers and Exit priority register

Peripheral I/O interrupts have their own interrupt control registers. Figure 1.9.3 and 1.9.4 show the interrupt control registers and figure 1.9.5 shows exit priority register.

### Interrupt control register

| Symbol | Address | When reset |
|---|---|---|
| TAiIC(i=0 to 4) | $006C_{16}$, $008C_{16}$, $006E_{16}$, $008E_{16}$, $0070_{16}$ | $XXXXX000_2$ |
| TBiIC(i=0 to 5) | $0094_{16}$, $0076_{16}$, $0096_{16}$, $0078_{16}$, $0098_{16}$, $0069_{16}$ | $XXXXX000_2$ |
| SiTIC(i=0 to 4) | $0090_{16}$, $0092_{16}$, $0089_{16}$, $008B_{16}$, $008D_{16}$ | $XXXXX000_2$ |
| SiRIC(i=0 to 4) | $0072_{16}$, $0074_{16}$, $006B_{16}$, $006D_{16}$, $006F_{16}$ | $XXXXX000_2$ |
| BCNiIC(i=0 to 4) | $0071_{16}$, $0091_{16}$, $008F_{16}$, $0071_{16}$(Note 1), $0091_{16}$(Note 2) | $XXXXX000_2$ |
| DMiIC(i=0 to 3) | $0068_{16}$, $0088_{16}$, $006A_{16}$, $008A_{16}$ | $XXXXX000_2$ |
| ADiIC(i=0,1) | $0073_{16}$, $0086_{16}$ | $XXXXX000_2$ |
| KUPIC | $0093_{16}$ | $XXXXX000_2$ |
| IIOiIC(i=0 to 5) | $0075_{16}$, $0095_{16}$, $0077_{16}$, $0097_{16}$, $0079_{16}$, $0099_{16}$ | $XXXXX000_2$ |
| IIOiIC(i=6 to 11) | $007B_{16}$, $009B_{16}$, $007D_{16}$, $009D_{16}$, $007F_{16}$, $0081_{16}$ | $XXXXX000_2$ |
| CANiIC(i=0 to 2) | $009D_{16}$, $007F_{16}$, $0081_{16}$ | $XXXXX000_2$ |

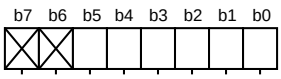



| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| ILVL0 | Interrupt priority level select bit | b2 b1 b0<br>0 0 0 : Level 0 (interrupt disabled)<br>0 0 1 : Level 1<br>0 1 0 : Level 2<br>0 1 1 : Level 3<br>1 0 0 : Level 4<br>1 0 1 : Level 5<br>1 1 0 : Level 6<br>1 1 1 : Level 7 | ○ | ○ |
| ILVL1 | | | ○ | ○ |
| ILVL2 | | | ○ | ○ |
| IR | Interrupt request bit | 0 : Interrupt not requested<br>1 : Interrupt requested | ○ | ○ (Note 3) |
| Nothing is assigned.<br>When write, set "0". When read, their contents are indeterminate. | | | – | – |
| | | | – | – |
| | | | – | – |
| | | | – | – |

Note 1: UART0 bus collision and start/stop condition detection interrupt control register is shared with UART3.
Note 2: UART1 bus collision and start/stop condition detection interrupt control register is shared with UART4.
Note 3: This bit can only be accessed for reset (= 0), but cannot be accessed for set (= 1).

**Figure 1.9.3. Interrupt control register (1)**

Interrupt control register




| Symbol | Address | When reset |
|---|---|---|
| INTiIC(i=0 to 2) | 009E16, 007E16, 009C16 | XX00 X0002 |
| INTiIC(i=3 to 5)(*1) | 007C16, 009A16, 007A16 | XX00 X0002 |

| 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 (Note 2) | 0 : Selects falling edge or L level<br>1 : Selects rising edge or H level | ○ | ○ |
| LVS | Level sense/edge sense select bit | 0 : Edge sense<br>1 : Level sense (Note 3) | ○ | ○ |
| Nothing is assigned.<br>When write, set "0". When read, their contents are indeterminate. | | | – | – |
| | | | – | – |

Note 1: This bit can only be accessed for reset (= 0), but cannot be accessed for set (= 1).
Note 2: When related bit of external interrupt cause select register (address 031F16) are used for both edge, select the falling edge (=0).
Note 3: When level sense is selected, set related bit of external interrupt cause select register (address 031F16) to one edge.

*1 When using 16-bit data bus width in microprocessor mode or memory expansion mode, $\overline{INT3}$ to $\overline{INT5}$ are used for data bus. In this case, set the interrupt disabled to INT3IC, INT4IC and INT5IC.

**Figure 1.9.4. Interrupt control register (2)**

**Bit 0 to 2: Interrupt Priority Level Select Bits (ILVL0 to ILVL2)**

Interrupt priority levels are set by ILVL0 to ILVL2 bits. When an interrupt request is generated, the interrupt priority level of this interrupt is compared with IPL. This interrupt is enabled only when its interrupt priority level is greater than IPL. This means that you can disable any particular interrupt by setting its interrupt priority level to 0.

**Bit 3: Interrupt Request Bit (IR)**

This bit is set (= 1) by hardware when an interrupt request is generated. The bit is cleared (= 0) by hardware when the interrupt request is acknowledged and jump to the interrupt vector.
This bit can be cleared (= 0) (but never be set to 1) in software.

## Exit priority register

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

Symbol: RLVL
Address: 009F16
When reset: XX0X00002

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| RLVL0 | Interrupt priority set bits for exiting Stop/Wait state (Note 1) | b2 b1 b0<br>0 0 0 : Level 0<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 | ○ | ○ |
| RLVL1 | | | ○ | ○ |
| RLVL2 | | | ○ | ○ |
| FSIT | High-speed interrupt set bit (Note 2) | 0: Interrupt priority level 7 = normal interrupt<br>1: Interrupt priority level 7 = high-speed interrupt | ○ | ○ |
| Nothing is assigned. When write, set "0". When read, its content is indeterminate. | | | — | — |
| DMA II | DMA II select bit (Note 3) | 0: Interrupt priority level 7 = normal interrupt or high-speed interrupt<br>1: Interrupt priority level 7 = DMA II transfer | ○ | ○ |
| Nothing is assigned. When write, set "0". When read, their contents are indeterminate. | | | — | — |
| | | | — | — |

Note 1: Exits the Stop or Wait mode when the requested interrupt priority level is higher than that set in the exit priority register.
Set to the same value as the processor interrupt priority level (IPL) set in the flag register (FLG).
Note 2: The high-speed interrupt can only be specified for interrupts with interrupt priority level 7. Specify interrupt priority level 7 for only one interrupt.
Note 3: Do not set this bit to 0 after once setting it to 1.
When this bit is 1, set the high-speed interrupt select bit to 0. (This cannot be used simultaneously with the high-speed interrupt.)
Transfers by DMAC II are unaffected by the interrupt enable flag (I flag) and processor interrupt priority level (IPL).

**Figure 1.9.5. Exit priority register**

**Bit 0 to 2: Interrupt priority set bits for exiting Stop/Wait state (RLVL0 to RLVL2)**

When using an interrupt to exit Stop mode or Wait mode, the relevant interrupt must be enabled and set to a priority level above the level set by the RLVL0 to RLVL2 bits. Set the RLVL0 to RLVL2 bits to the same level as the flag register (FLG) IPL.

## 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 SCMPU, SIN, SMOVB, SMOVF, SMOVU, SSTR, SOUT or RMPA instruction, the processor temporarily suspends the instruction being executed, and transfers control to the interrupt sequence.

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

(1) CPU gets the interrupt information (the interrupt number and interrupt request level) by reading address $00000_{16}$ (address $00002_{16}$ when high-speed interrupt). After this, the related interrupt request bit is "0".

(2) Saves the contents of the flag register (FLG) immediately before the start of interrupt sequence in the temporary register (Note) within the CPU.

(3) Sets the interrupt enable flag (I flag), the debug flag (D flag), and the stack pointer select flag (U flag) to “0” (the U flag, however does not change if the INT instruction, in software interrupt numbers 32 through 63, is executed)

(4) Saves the contents of the temporary register (Note) within the CPU in the stack area. Saves in the flag save register (SVF) in high-speed interrupt.

(5) Saves the content of the program counter (PC) in the stack area. Saves in the PC save register (SVP) in high-speed interrupt.

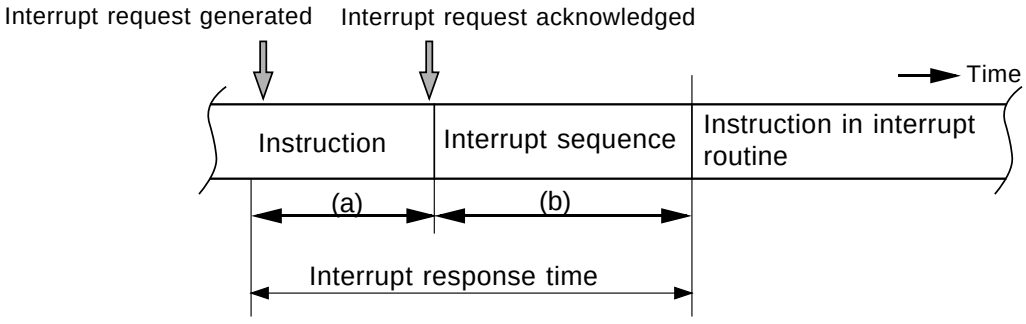
(6) Sets the interrupt priority level of the accepted instruction in the IPL.

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

Note: This register cannot be utilized by the user.

## Interrupt Response Time

'Interrupt response time' is the period between the instant an interrupt occurs and the instant the first instruction within the interrupt routine has been executed. This time comprises the period from the occurrence of an interrupt to the completion of the instruction under execution at that moment (a) and the time required for executing the interrupt sequence (b). Figure 1.9.6 shows the interrupt response time.




(a) The period from the occurrence of an interrupt to the completion of the instruction under execution.
(b) The time required for executing the interrupt sequence.

**Figure 1.9.6. Interrupt response time**

Time (a) varies with each instruction being executed. The DIVX instruction requires a maximum time of 29* cycles.
Time (b) is shown in table 1.9.5.

* It is when the divisor is immediate or register. When the divisor is memory, the following value is added.
  - Normal addressing : 2 + X
  - Index addressing : 3 + X
  - Indirect addressing : 5 + X + 2Y
  - Indirect index addressing : 6 + X + 2Y

X is number of wait of the divisor area. Y is number of wait of the indirect address stored area. When X and Y are in odd address or in 8 bit bus area, double the value of X and Y.

**Table 1.9.5 Interrupt Sequence Execution Time**

| Interrupt | Interrupt vector address | 16 bits data bus | 8 bits data bus |
|---|---|---|---|
| Peripheral I/O | Even address | 14 cycles | 16 cycles |
| | Odd address (Note 1) | 16 cycles | 16 cycles |
| INT instruction | Even address | 12 cycles | 14 cycles |
| | Odd address (Note 1) | 14 cycles | 14 cycles |
| $\overline{\text{NMI}}$<br>Watchdog timer<br>Undefined instruction<br>Address match | Even address (Note 2) | 13 cycles | 15 cycles |
| Overflow | Even address (Note 2) | 14 cycles | 16 cycles |
| BRK instruction (Relocatable vector table) | Even address | 17 cycles | 19 cycles |
| | Odd address (Note 1) | 19 cycles | 19 cycles |
| Single step<br>BRK2 instruction<br>BRK instruction (Fixed vector table) | Even address (Note 2) | 19 cycles | 21 cycles |
| High-speed interrupt (Note 3) | Vector table is internal register | 5 cycles | |

Note 1: Allocate interrupt vector addresses in even addresses as much as possible.
Note 2: The vector table is fixed to even address.
Note 3: The high-speed interrupt is independent of these conditions.

## Changes of IPL When Interrupt Request Acknowledged

When an interrupt request is acknowledged, the interrupt priority level of the acknowledged interrupt is set to the processor interrupt priority level (IPL).

If an interrupt request is acknowledged that does not have an interrupt priority level, the value shown in Table 1.9.6 is set to the IPL.

**Table 1.9.6 Relationship between Interrupts without Interrupt Priority Levels and IPL**

| Interrupt sources without interrupt priority levels | Value that is set to IPL |
|---|---|
| Watchdog timer, $\overline{NMI}$ | 7 |
| Reset | 0 |
| Other | Not changed |

## Saving Registers

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

The order in which these contents are saved are as follows: First, the FLG register is saved to the stack area. Next, the 16 high-order bits and 16 low-order bits of the program counter expanded to 32-bit are saved. Figure 1.9.7 shows the stack status before an interrupt request is acknowledged and the stack status after an interrupt request is acknowledged.

In a high-speed interrupt sequence, the contents of the flag register (FLG) are saved to the flag save register (SVF) and program counter (PC) are saved to PC save register (SVP).

If there are any other registers you want to be saved, save them in software at the beginning of the interrupt routine. The PUSHM instruction allows you to save all registers except the stack pointer (SP) by a single instruction.

In high speed interrupt, switch register bank, then register bank 1 is used as high speed interrupt register. In this case, switch register bank mode for high-speed interrupt routine.

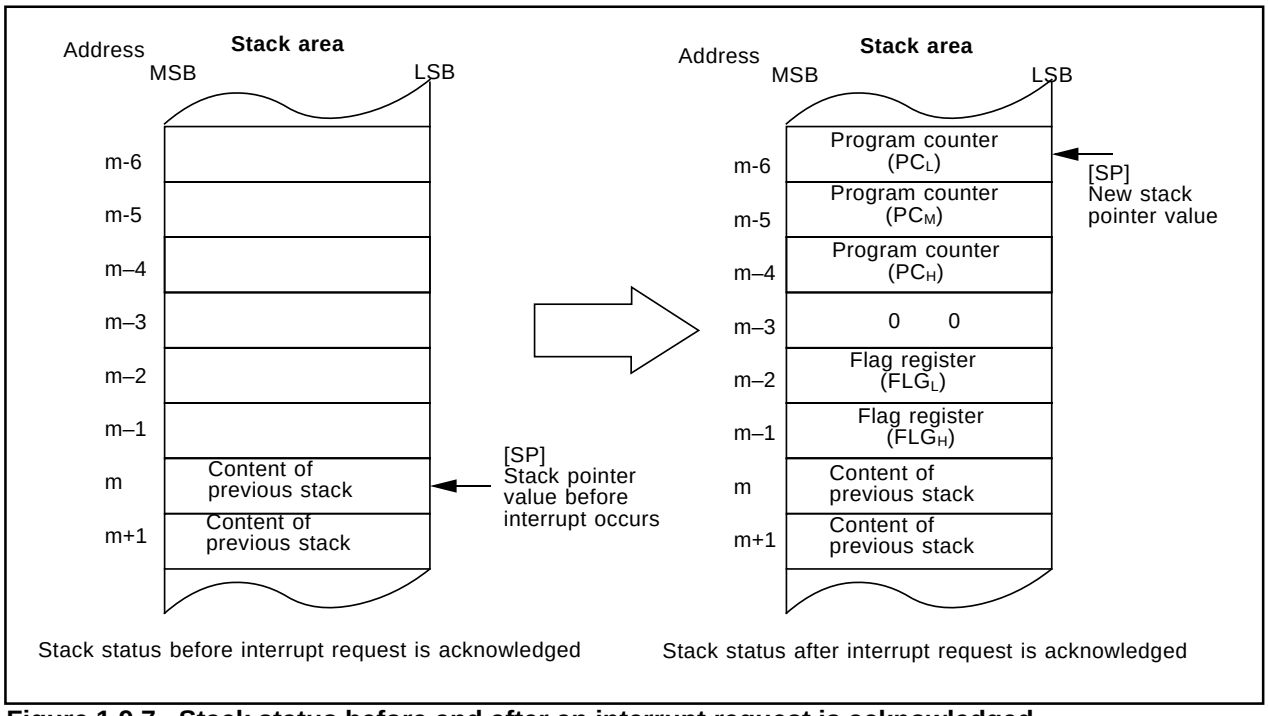



**Figure 1.9.7. Stack status before and after an interrupt request is acknowledged**

## Return from Interrupt Routine

As you execute the REIT instruction at the end of the interrupt routine, the contents of the flag register (FLG) and program counter (PC) that have been saved to the stack area immediately preceding the interrupt sequence are automatically restored. In high-speed interrupt, as you execute the FREIT instruction at the end of the interrupt routine, the contents of the flag register (FLG) and program counter (PC) that have been saved to the save registers immediately preceding the interrupt sequence are automatically restored.

Then control returns to the routine that was under execution before the interrupt request was acknowledged, and processing is resumed from where control left off.

If there are any registers you saved via software in the interrupt routine, be sure to restore them using an instruction (e.g., POPM instruction) before executing the REIT or FREIT instruction.

When switching the register bank before executing REIT and FREIT instruction, switched to the register bank immediately before the interrupt sequence.

## Interrupt Priority

If two or more interrupt requests are sampled active at the same time, the interrupt with the highest priority will be acknowledged.

Maskable interrupts (Peripheral I/O interrupts) can be assigned any desired priority by setting the interrupt priority level select bit accordingly. If some maskable interrupts are assigned the same priority level, the priority between these interrupts are resolved by the priority that is set in hardware.

Certain nonmaskable interrupts such as a reset (reset is given the highest priority) and watchdog timer interrupt have their priority levels set in hardware. Figure 1.9.8 lists the hardware priority levels of these interrupts.

Software interrupts are not subjected to interrupt priority. They always cause control to branch to an interrupt routine whenever the relevant instruction is executed.

## Interrupt Resolution Circuit

Interrupt resolution circuit selects the highest priority interrupt when two or more interrupt requests are sampled active at the same time.

Figure 1.9.9 shows the interrupt resolution circuit.

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

**Figure 1.9.8. Interrupt priority that is set in hardware**

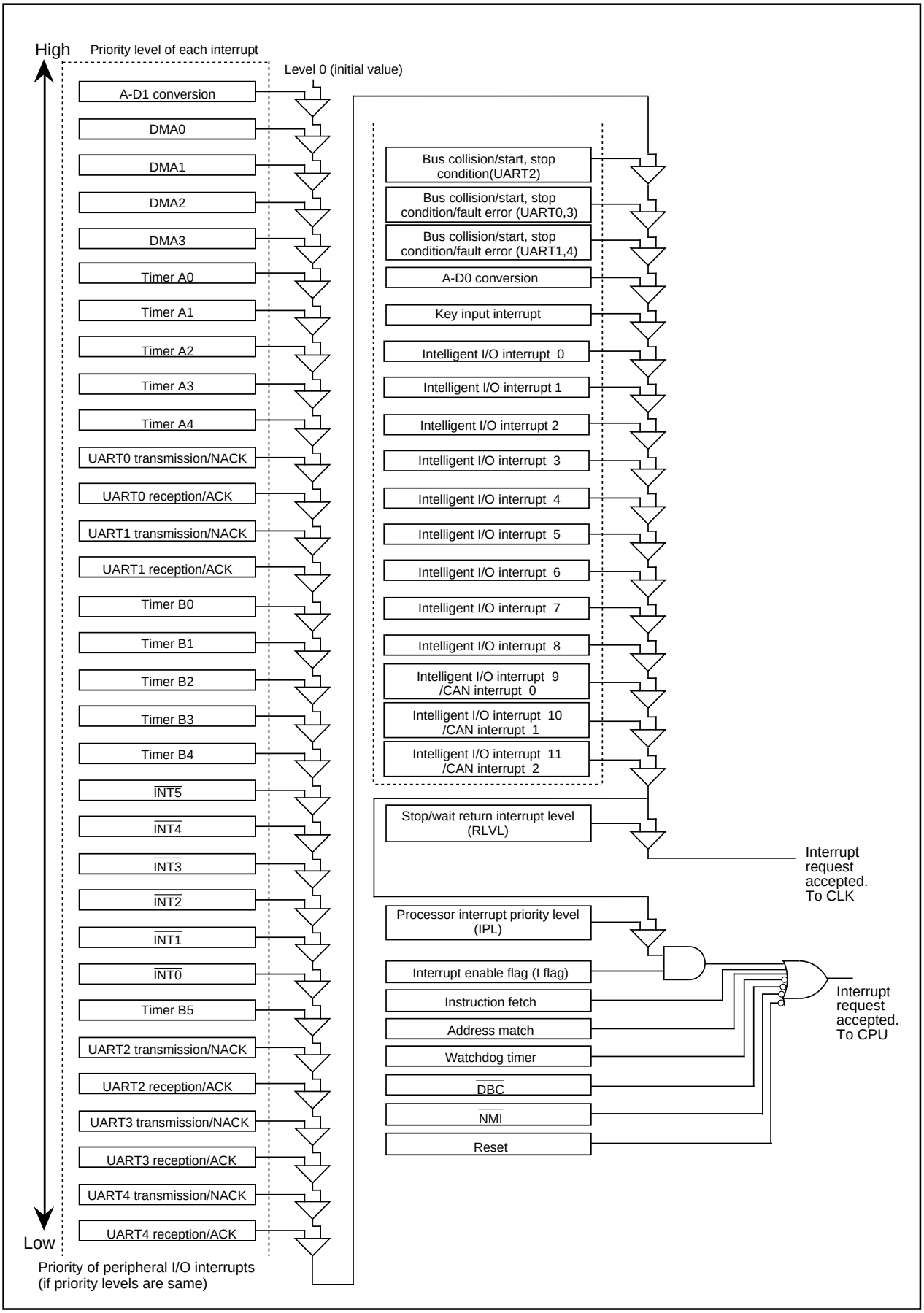



**Figure 1.9.9. Interrupt resolution circuit**

## INT Interrupts

INT0 to INT5 are external input interrupts. The level sense/edge sense switching bits of the interrupt control register select the input signal level and edge at which the interrupt can be set to occur on input signal level and input signal edge. The polarity bit selects the polarity.

With the external interrupt input edge sense, the interrupt can be set to occur on both rising and falling edges by setting the INTi interrupt polarity switch bit of the interrupt request select register (address 031F16) to “1”. When you select both edges, set the polarity switch bit of the corresponding interrupt control register to the falling edge (“0”).

When you select level sense, set the INTi interrupt polarity switch bit of the interrupt request select register (address 031F16) to “0”.

Figure 1.9.10 shows the interrupt request select register.

External interrupt request cause select register

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

Symbol: IFSR — Address: 031F16 — When reset: 0016

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| IFSR0 | INT0 interrupt polarity select bit (Note) | 0 : One edge<br>1 : Both edges | ○ | ○ |
| IFSR1 | INT1 interrupt polarity select bit (Note) | 0 : One edge<br>1 : Both edges | ○ | ○ |
| IFSR2 | INT2 interrupt polarity select bit (Note) | 0 : One edge<br>1 : Both edges | ○ | ○ |
| IFSR3 | INT3 interrupt polarity select bit | 0 : One edge<br>1 : Both edges | ○ | ○ |
| IFSR4 | INT4 interrupt polarity select bit | 0 : One edge<br>1 : Both edges | ○ | ○ |
| IFSR5 | INT5 interrupt polarity select bit | 0 : One edge<br>1 : Both edges | ○ | ○ |
| IFSR6 | UART0/3 interrupt cause select bit | 0 : UART3 bus collision /start,stop detect/false error detect<br>1 : UART0 bus collision /start,stop detect/false error detect | ○ | ○ |
| IFSR7 | UART1/4 interrupt cause select bit | 0 : UART4 bus collision /start,stop detect/false error detect<br>1 : UART1 bus collision /start,stop detect/false error detect | ○ | ○ |

Note :When level sense is selected, set this bit to "0". When both edges are selected, set the corresponding polarity switching bit of INT interrupt control register to "0" (falling edge).

**Figure 1.9.10. External interrupt request cause select register**

## $\overline{\text{NMI}}$ 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 03C4$_{16}$).

This pin cannot be used as a normal port input.

### Notes:

When not intending to use the $\overline{\text{NMI}}$ function, be sure to connect the $\overline{\text{NMI}}$ pin to $V_{CC}$ (pulled-up). The $\overline{\text{NMI}}$ interrupt is non-maskable. Because it cannot be disabled, the pin must be pulled up.

## 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 1.9.11 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.

Setting the key input interrupt disable bit (bit 7 at address 03AF$_{16}$) to “1” disables key input interrupts from occurring, regardless of the setting in the interrupt control register. When “1” is set in the key input interrupt disable register, there is no input via the port pin even when the direction register is set to input.

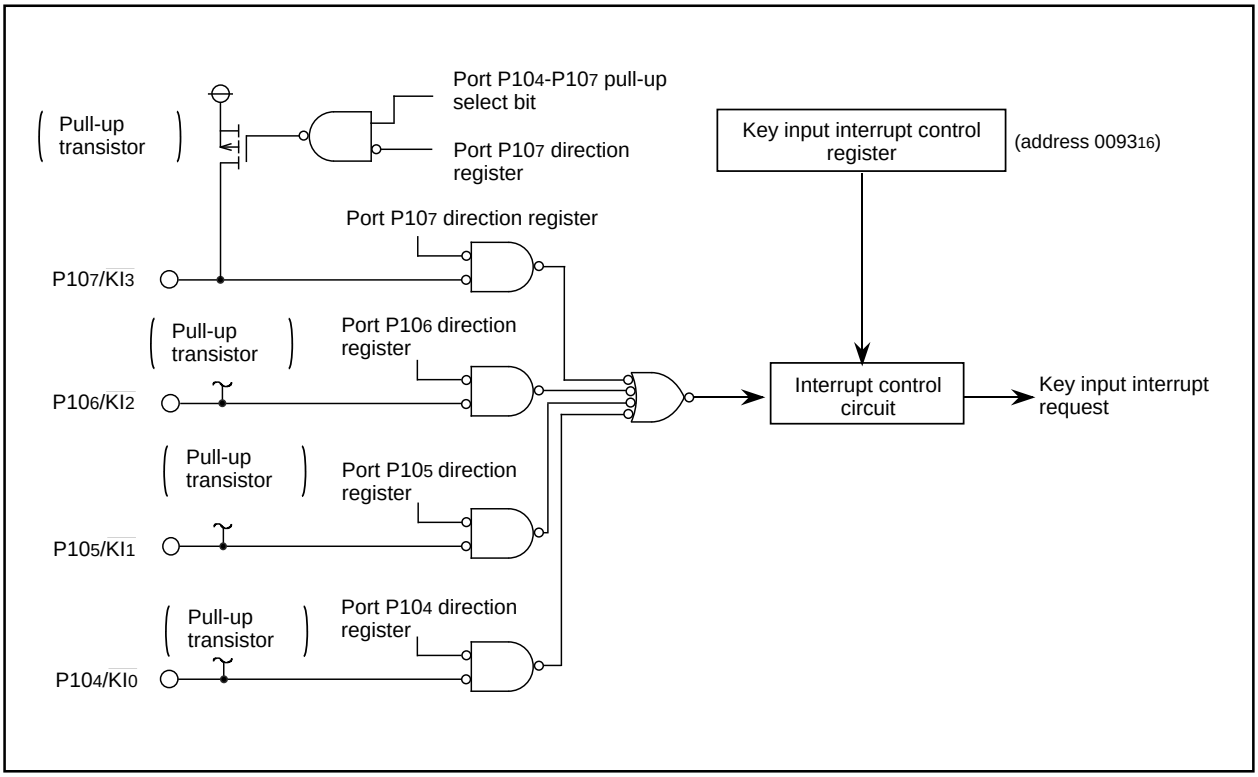



**Figure 1.9.11. Block diagram of key input interrupt**

## Address Match Interrupt

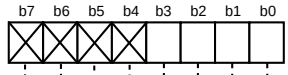
An address match interrupt is generated when the address match interrupt address register contents match the program counter value. Four address match interrupts can be set, each of which can be enabled and disabled by an address match interrupt enable bit. Address match interrupts are not affected by the interrupt enable flag (I flag) and processor interrupt priority level (IPL).

Figure 1.9.12 shows the address match interrupt-related registers.

Set the start address of an instruction to the address match interrupt register.

Address match interrupt is not generated when address such as the middle of instruction or table data is set.

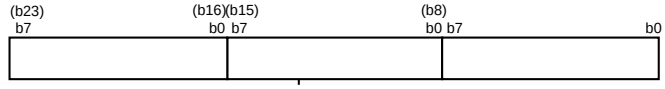
Address match interrupt enable register




| Symbol | Address | When reset |
|---|---|---|
| AIER | $0009_{16}$ | $XXXX0000_2$ |

| 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 | ○ | ○ |
| AIER2 | Address match interrupt 2 enable bit | 0 : Interrupt disabled<br>1 : Interrupt enabled | ○ | ○ |
| AIER3 | Address match interrupt 3 enable bit | 0 : Interrupt disabled<br>1 : Interrupt enabled | ○ | ○ |
| Nothing is assigned.<br>When write, set "0". When read, their contents are indeterminate. | | | — | — |

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




| Symbol | Address | When reset |
|---|---|---|
| RMAD0 | $0012_{16}$ to $0010_{16}$ | $000000_{16}$ |
| RMAD1 | $0016_{16}$ to $0014_{16}$ | $000000_{16}$ |
| RMAD2 | $001A_{16}$ to $0018_{16}$ | $000000_{16}$ |
| RMAD3 | $001E_{16}$ to $001C_{16}$ | $000000_{16}$ |

| Function | Values that can be set | R | W |
|---|---|---|---|
| Address setting register for address match interrupt | $000000_{16}$ to $FFFFFF_{16}$ | ○ | ○ |

**Figure 1.9.12. Address match interrupt-related registers**

## Intelligent I/O and CAN Interrupt

Group 0 to 3 intelligent I/O interrupts and CAN interrupt are assigned to software interrupt numbers 44 to 54 and 57.

As intelligent I/O interrupt request, there are base timer interrupt request signals, time measurement interrupt request signals, waveform generation interrupt request signals and interrupt request signals from various communication circuits.

Figure 1.9.13 shows the intelligent I/O interrupts and CAN interrupt block diagram, figure 1.9.14 shows the interrupt request register and figure 1.9.15 shows interrupt enable register.

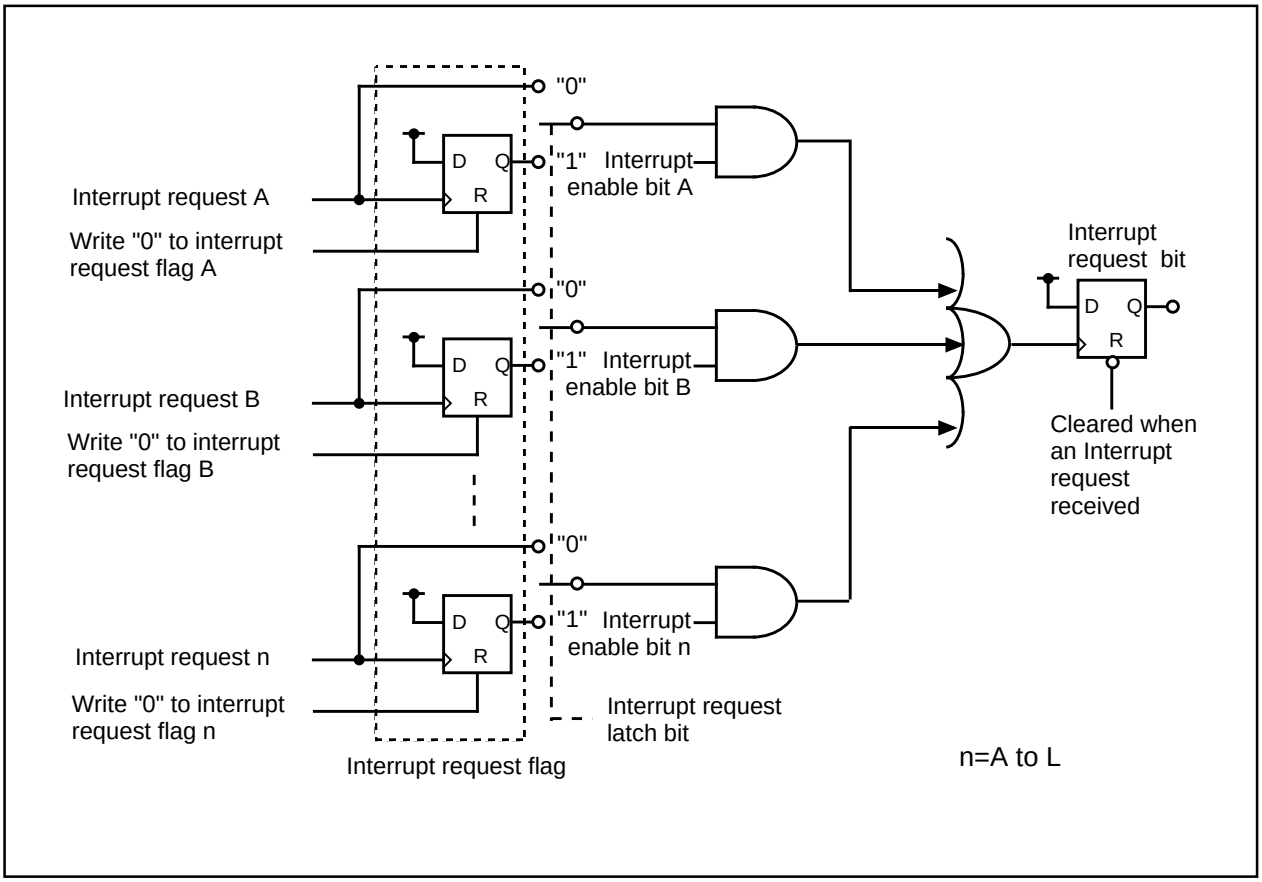
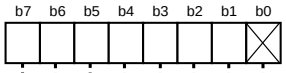



**Figure 1.9.13. Intelligent I/O and CAN interrupt block diagram**

When using the intelligent I/O or CAN interrupt as an starting factor for DMA II, the interrupt latch bit must be set to "0" in order to enable only the interrupt request factor used by the interrupt enable register.

## Interrupt request register




| Symbol | Address | When reset |
|---|---|---|
| IIOiIR | See below | 0000 000X$_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| —— | Nothing is assigned.<br>When write, set "0". When read, the content is indeterminate. | | — | — |
| IRF1 | Interrupt request flag 1 | 0 : Interrupt request not present<br>1 : Interrupt request present (Note) | ○ | ○ |
| IRF2 | Interrupt request flag 2 | 0 : Interrupt request not present<br>1 : Interrupt request present (Note) | ○ | ○ |
| IRF3 | Interrupt request flag 3 | 0 : Interrupt request not present<br>1 : Interrupt request presence (Note) | ○ | ○ |
| IRF4 | Interrupt request flag 4 | 0 : Interrupt request not present<br>1 : Interrupt request present (Note) | ○ | ○ |
| IRF5 | Interrupt request flag 5 | 0 : Interrupt request not present<br>1 : Interrupt request present (Note) | ○ | ○ |
| IRF6 | Interrupt request flag 6 | 0 : Interrupt request not present<br>1 : Interrupt request present (Note) | ○ | ○ |
| IRF7 | Interrupt request flag 7 | 0 : Interrupt request not present<br>1 : Interrupt request present (Note) | ○ | ○ |

Note: "0" can be written.

Interrupt request register table

| Symbol | Address | bit7 (IRF7) | bit6 (IRF6) | bit5 (IRF5) | bit4 (IRF4) | bit3 (IRF3) | bit2 (IRF2) | bit1 (IRF1) | bit0 - |
|---|---|---|---|---|---|---|---|---|---|
| IIO0IR | 00A0$_{16}$ | - | - | SIO0r | G0RI | - | PO13 | TM02 | - |
| IIO1IR | 00A1$_{16}$ | - | - | SIO0t | G0TO | - | PO14 | TM00/PO00 | - |
| IIO2IR | 00A2$_{16}$ | - | - | SIO1r | G1RI | - | TM12/PO12 | - | - |
| IIO3IR | 00A3$_{16}$ | - | - | SIO1t | G1TO | PO27 | PO10 | TM03 | - |
| IIO4IR | 00A4$_{16}$ | BEAN0 | BEAN1 | - | BT1 | PO32 | TM17/PO17 | TM04/PO04 | - |
| IIO5IR | 00A5$_{16}$ | - | - | - | SIO2r | PO33 | PO21 | TM05/PO05 | - |
| IIO6IR | 00A6$_{16}$ | - | - | - | SIO2t | PO34 | PO20 | TM06 | - |
| IIO7IR | 00A7$_{16}$ | IE0 | - | - | BT0 | PO35 | PO22 | TM07 | - |
| IIO8IR | 00A8$_{16}$ | IE1 | IE2 | - | BT2 | PO36 | PO23 | TM11/PO11 | - |
| IIO9IR | 00A9$_{16}$ | CAN0 | - | - | - | PO31 | PO24 | PO15 | - |
| IIO10IR | 00AA$_{16}$ | CAN1 | - | - | - | PO30 | PO25 | TM16/PO16 | - |
| IIO11IR | 00AB$_{16}$ | CAN2 | - | - | BT3 | PO37 | PO26 | TM01/PO01 | - |

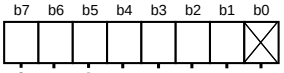
BTi : Interrupt request from base timer of intelligent I/O group i
TMij : Interrupt request from time measurement function ch j of intelligent I/O group i
POij : Interrupt request from waveform generator function ch j of intelligent I/O group i
SIOir/SIOit : Interrupt request from communication function of intelligent I/O group i (r:reception, t:transmission)
GiTO/GiRI : Interrupt request from HDLC data processing function of intelligent I/O group i (RI:reception input, TO:transmission output)
BEANi : Interrupt request from special communication function of intelligent I/O group i (i=0,1)
IE : Interrupt request from IEBus communication function of intelligent I/O group 2
CANi : Interrupt request from AN communication function (i=0 to 2)
- : Nothing is assigned in this bit.

**Figure 1.9.14. Interrupt request registers**

**Bit 1 to bit 7: Interrupt request flag (IRF1 to IRF7)**

To retain respective interrupt requests and judge interrupt kind occurred in the interrupt process routine.

## Interrupt enable register




| Symbol | Address | When reset |
|---|---|---|
| IIOiIE | See below | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| IRLT | Interrupt request latch bit | 0: Interrupt request is not latched(used by DMA II)<br>1: Interrupt request is latched(used by interrupt) | ○ | ○ |
| ITE1 | Interrupt enable bit 1 | 0: Interrupt of corresponding interrupt request flag (IRF1) disabled<br>1: Interrupt of corresponding interrupt request flag (IRF1) enabled | ○ | ○ |
| ITE2 | Interrupt enable bit 2 | 0: Interrupt of corresponding interrupt request flag (IRF2) disabled<br>1: Interrupt of corresponding interrupt request flag (IRF2) enabled | ○ | ○ |
| ITE3 | Interrupt enable bit 3 | 0: Interrupt of corresponding interrupt request flag (IRF3) disabled<br>1: Interrupt of corresponding interrupt request flag (IRF3) enabled | ○ | ○ |
| ITE4 | Interrupt enable bit 4 | 0: Interrupt of corresponding interrupt request flag (IRF4) disabled<br>1: Interrupt of corresponding interrupt request flag (IRF4) enabled | ○ | ○ |
| ITE5 | Interrupt enable bit 5 | 0: Interrupt of corresponding interrupt request flag (IRF5) disabled<br>1: Interrupt of corresponding interrupt request flag (IRF5) enabled | ○ | ○ |
| ITE6 | Interrupt enable bit 6 | 0: Interrupt of corresponding interrupt request flag (IRF6) disabled<br>1: Interrupt of corresponding interrupt request flag (IRF6) enabled | ○ | ○ |
| ITE7 | Interrupt enable bit 7 | 0: Interrupt of corresponding interrupt request flag (IRF7) disabled<br>1: Interrupt of corresponding interrupt request flag (IRF7) enabled | ○ | ○ |

Interrupt enable register table

| Symbol | Address | bit7 (ITE7) | bit6 (ITE6) | bit5 (ITE5) | bit4 (ITE4) | bit3 (ITE3) | bit2 (ITE2) | bit1 (ITE1) | bit1 (IRLT) |
|---|---|---|---|---|---|---|---|---|---|
| IIO0IE | $00B0_{16}$ | - | - | SIO0r | G0RI | - | PO13 | TM02 | IRLT |
| IIO1IE | $00B1_{16}$ | - | - | SIO0t | G0TO | - | PO14 | TM00/PO00 | IRLT |
| IIO2IE | $00B2_{16}$ | - | - | SIO1r | G1RI | - | TM12/PO12 | - | IRLT |
| IIO3IE | $00B3_{16}$ | - | - | SIO1t | G1TO | PO27 | PO10 | TM03 | IRLT |
| IIO4IE | $00B4_{16}$ | BEAN0 | BEAN1 | - | BT1 | PO32 | TM17/PO17 | TM04/PO04 | IRLT |
| IIO5IE | $00B5_{16}$ | - | - | - | SIO2r | PO33 | PO21 | TM05/PO05 | IRLT |
| IIO6IE | $00B6_{16}$ | - | - | - | SIO2t | PO34 | PO20 | TM06 | IRLT |
| IIO7IE | $00B7_{16}$ | IE0 | - | - | BT0 | PO35 | PO22 | TM07 | IRLT |
| IIO8IE | $00B8_{16}$ | IE1 | IE2 | - | BT2 | PO36 | PO23 | TM11/PO11 | IRLT |
| IIO9IE | $00B9_{16}$ | CAN0 | - | - | - | PO31 | PO24 | PO15 | IRLT |
| IIO10IE | $00BA_{16}$ | CAN1 | - | - | - | PO30 | PO25 | TM16/PO16 | IRLT |
| IIO11IE | $00BB_{16}$ | CAN2 | - | - | BT3 | PO37 | PO26 | TM01/PO01 | IRLT |

BTi : Interrupt request from base timer of intelligent I/O group i is enabled
TMij : Interrupt request from time measurement function ch j of intelligent I/O group i is enabled
POij : Interrupt request from waveform generator function ch j of intelligent I/O group i is enabled
SIOir/SIOit : Interrupt request from communication function of intelligent I/O group i (r:reception, t:transmission) is enabled
GiTO/GiRI : Interrupt request from HDLC data processing function of intelligent I/O group i (RI:reception input, TO:transmission output) is enabled
BEANi : Interrupt request from special communication function of intelligent I/O group i (i=0,1) is enabled
IE : Interrupt request from IEBus communication function of intelligent I/O group 2 is enabled
CANi : Interrupt request from CAN communication function (i=0 to 2) is enabled
- : Nothing is assigned in this bit. (Set "0" to these bits.)

**Figure 1.9.15. Interrupt enable registers**

**Bit 0: Interrupt request latch bit (IRLT)**

An interrupt signal or latched signal of the interrupt signal is selected as an interrupt request signal. When the latched signal of an interrupt signal is used, this flag must be set to "0" after interrupt request flag is read in interrupt process routine, . If this flag is not set to "0" and interrupt process is completed, although interrupt request occurs again, interrupt will not occur.

**Bit 1 to bit 7: Interrupt enable bit (ITE 1 to ITE 7)**

To enable/disable respective interrupts.

## Precautions for Interrupts

### (1) Reading addresses $000000_{16}$ and $000002_{16}$

- When maskable interrupt occurs, CPU reads the interrupt information (the interrupt number and interrupt request level) in the interrupt sequence from address $000000_{16}$. When a high-speed interrupt occurs, CPU reads from address $000002_{16}$.
  The interrupt request bit of the certain interrupt will then be set to “0”.
  However, reading addresses $000000_{16}$ and $000002_{16}$ by software does not set request bit to “0”.

### (2) Setting the stack pointer

- The value of the stack pointer immediately after reset is initialized to $000000_{16}$. Accepting an interrupt before setting a value in the stack pointer may cause 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. Any interrupt including the $\overline{\text{NMI}}$ interrupt is generated immediately after executing the first instruction after reset. Set an even number to the stack pointer. Set an even address to the stack pointer so that operating efficiency is increased.

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

- As for the $\overline{\text{NMI}}$ interrupt pin, this interrupt cannot be disabled. Connect it to the Vcc pin via a pull-up resistor if unused.
- 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.
- A low level signal with more than 1 clock cycle (BCLK) is necessary for $\overline{\text{NMI}}$ pin.

### (4) External interrupt

- Edge sense
  Either a low level or a high level for at least 250 ns is necessary for the signal input to pins $\text{INT}_0$ to $\text{INT}_5$ regardless of the CPU operation clock.
- Level sense
  Either a low level or a high level of 1 cycle of BCLK + at least 200 ns width is necessary for the signal input to pins $\text{INT}_0$ to $\text{INT}_5$ regardless of the CPU operation clock. (When $X_{IN}$=20MHz and no division mode, at least 250 ns width is necessary.)

• When the polarity of the INT0 to INT5 pins is changed, the interrupt request bit is sometimes set to "1". After changing the polarity, set the interrupt request bit to "0". Figure 1.9.12 shows the procedure for changing the $\overline{\text{INT}}$ interrupt generate factor.

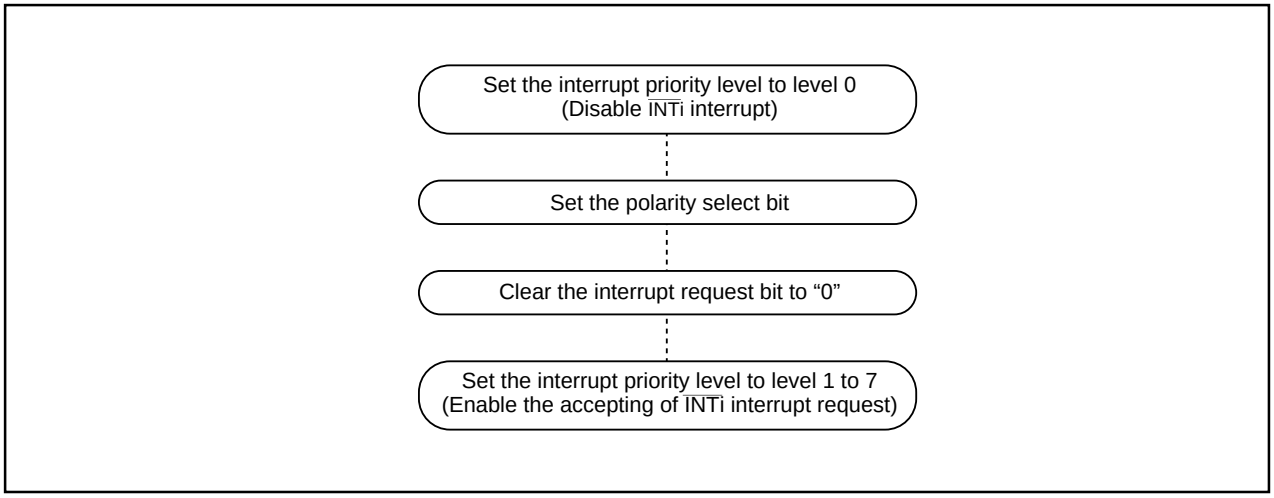



**Figure 1.9.16. Switching condition of $\overline{\text{INT}}$ interrupt request**

## (5) Rewrite the interrupt control register

• When an 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

## (6) Rewrite interrupt request register

• When writing to "0" to this register, the following instructions must be used.
Instructions : AND, BCLR

## 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. Whether a watchdog timer interrupt is generated or reset is selected when an underflow occurs in the watchdog timer. Watchdog timer interrupt is selected when bit 6 (CM06) of the system control register 0 (address $0008_{16}$) is "0" and reset is selected when CM06 is "1". No value other than "1" can be written in CM06. Once reset is selected (CM06="1"), watchdog timer interrupt cannot be selected by software.
When $X_{IN}$ is selected for the BCLK, bit 7 (WDC7) 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 WDC7. Therefore, the watchdog timer cycle can be calculated as follows. However, errors can arise in the watchdog timer cycle due to the prescaler.

When $X_{IN}$ is selected in BCLK

$$\text{Watchdog timer cycle} = \frac{\text{Prescaler division ratio (16 or 128) x watchdog timer count (32768)}}{\text{BCLK}}$$

When $X_{CIN}$ is selected in BCLK

$$\text{Watchdog timer cycle} = \frac{\text{Prescaler division ratio (2) x watchdog timer count (32768)}}{\text{BCLK}}$$

For example, when BCLK is 20MHz and the prescaler division ratio is set to 16, the monitor timer cycle is approximately 26.2 ms, and approximately 17.5 ms when BCLK is 30MHz.

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 $000E_{16}$). CM06 is initialized only at reset. After reset, watchdog timer interrupt is selected.
The watchdog timer and the prescaler stop in stop mode, wait mode and hold status. After exiting these modes and status, counting starts from the previous value.
In the stop mode, wait mode and hold state, the watchdog timer and prescaler are stopped. Counting is resumed from the held value when the modes or state are released. Figure 1.10.1 shows the block diagram of the watchdog timer. Figure 1.10.2 and 1.10.3 show the watchdog timer-related registers.

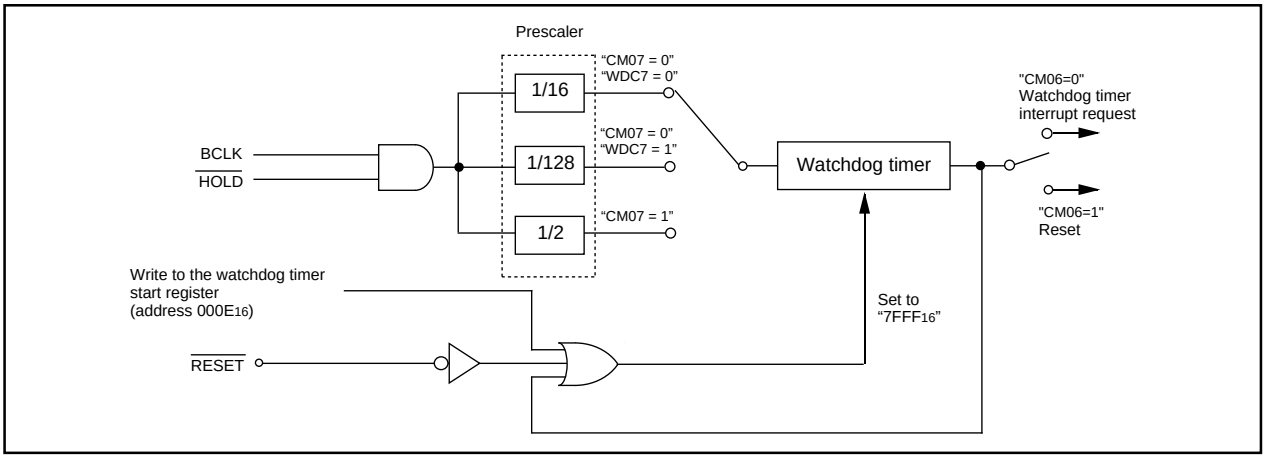



**Figure 1.10.1. Block diagram of watchdog timer**

## Watchdog timer control register

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

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

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| —— | High-order bit of watchdog timer | | ○ | — |
| —— | | | ○ | — |
| —— | | | ○ | — |
| —— | | | ○ | — |
| —— | | | ○ | — |
| —— | 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

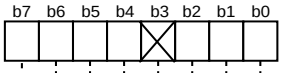
| b7 | b0 |
|---|---|
| | |

| Symbol | Address | When reset |
|---|---|---|
| WDTS | $000E_{16}$ | 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 the value written. | — | ○ |

**Figure 1.10.2. Watchdog timer control and start registers**

System clock control register 0 (Note 1)




Symbol: CM0 — Address: 000616 — When reset: 0000 X0002

| Bit symbol | Bit | Function | R | W |
|---|---|---|---|---|
| CM00 | Clock output function select bit (Note 2) | b1 b0<br>0 0 : I/O port P53<br>0 1 : fC output<br>1 0 : f8 output<br>1 1 : f32 output | O | O |
| CM01 | | | O | O |
| CM02 | WAIT peripheral function clock stop bit | 0 : Do not stop peripheral clock in wait mode<br>1 : Stop peripheral clock in wait mode (Note 3) | O | O |
| Nothing is assigned. When write, set "0". When read, their contents are indeterminate. | | | — | — |
| CM04 | Port XC select bit | 0 : I/O port<br>1 : XCIN-XCOUT generation (Note 4) | O | O |
| CM05 | Main clock (XIN-XOUT) stop bit (Note 5) | 0 : Main clock On<br>1 : Main clock Off (Note 6) | O | O |
| CM06 | Watchdog timer function select bit | 0 : Watchdog timer interrupt<br>1 : Reset (Note 7) | O | O |
| CM07 | System clock select bit (Note 8) | 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: The port P53 dose not function as an I/O port in microprocessor or memory expansion mode.
When outputting ALE to P53 (bits 5 and 4 of processor mode register 0 is "01"), set these bits to "00".
The port P53 function is not selected, even when you set "00" in microprocessor or memory expansion mode and bit 7 of the processor mode register 0 is "1".
Note 3: fc32 is not included. When this bit is set to "1", PLL cannot be used in WAIT.
Note 4: When XCIN-XCOUT is used, set port P86 and P87 to no pull-up resistance with the input port.
Note 5: When entering the power saving mode, the main clock is stopped using this bit. To stop the main clock, set system clock stop bit (CM07) to "1" while an oscillation of sub clock is stable. Then set this bit to "1".
When XIN is used after returning from stop mode, set this bit to "0".
When this bit is "1", XOUT is "H". Also, the internal feedback resistance remains ON, so XIN is pulled up to XOUT ("H" level) via the feedback resistance.
Note 6: When the main clock is stopped, the main clock division register (address 000C16) is set to the division by 8 mode.
However, in ring oscillator mode, the main clock division register is not set to the division by 8 mode when XIN-XOUT is stopped by this bit.
Note 7: When "1" has been set once, "0" cannot be written by software.
Note 8: Set this bit "0" to "1" when sub clock oscillation is stable by setting CM04 to "1".
Set this bit "1" to "0" when main clock oscillation is stable by setting CM05 to "0".
Do not set CM04 and CM05 simultaneously.

**Figure 1.10.3. System clock control register 0**

## DMAC

This microcomputer has four DMAC (direct memory access controller) channels that allow data to be sent to memory without using the CPU. DMAC is a function that transmit delete data of a source address (8 bits /16 bits) to a destination address when transmission request occurs. When using three or more DMAC channels, the register bank 1 and high-speed interrupt register are used as DMAC registers. If you are using three or more DMAC channels, you cannot use high-speed interrupts. The CPU and DMAC use the same data bus, but the DMAC has a higher bus access privilege than the CPU, and because of the use of cycle-steeling, operations are performed at high-speed from the occurrence of a transfer request until one word (16 bits) or 1 byte (8 bits) of data have been sent. Figure 1.11.1 shows the mapping of registers used by the DMAC. Table 1.11.1 shows DMAC specifications. Figures 1.11.2 to 1.11.5 show the structures of the registers used.

As the registers shown in Figure 1.11.1 are allocated in the CPU, use LDC instruction when writing. When writing to DCT2, DCT3, DRC2, DRC3, DMA2 and DMA3, set register bank select flag (B flag) to "1" and use MOV instruction to set R0 to R3, A0 and A1 registers. When writing to DSA2 and DSA3, set register bank select flag (B flag) to "1" and use LDC instruction to set SB and FB registers.

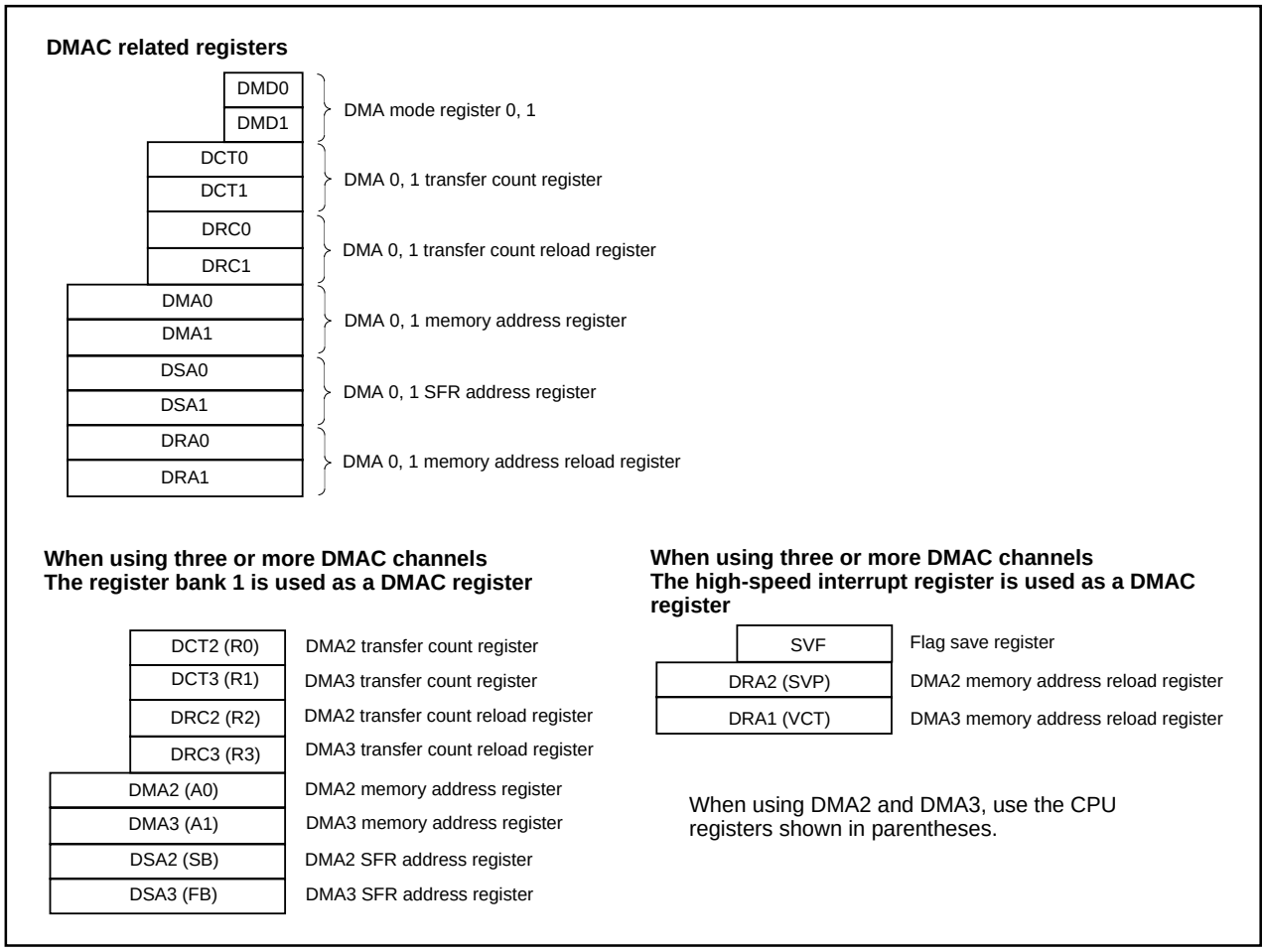



**Figure 1.11.1. Register map using DMAC**

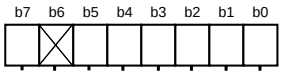
In addition to writing to the software DMA request bit to start DMAC transfer, the interrupt request signals output from the functions specified in the DMA request factor select bits are also used. However, in contrast to the interrupt requests, repeated DMA requests can be received, regardless of the interrupt flag.
(Note, however, that the number of actual transfers may not match the number of transfer requests if the DMA request cycle is shorter than the DMR transfer cycle. For details, see the description of the DMAC request bit.)

**Table 1.11.1. DMAC specifications**

| Item | Specification |
|---|---|
| No. of channels | 4 (cycle steal method) |
| Transfer memory space | • From any address in the 16 Mbytes space to a fixed address (16 Mbytes space)<br>• From a fixed address (16 Mbytes space) to any address in the 16 M bytes space |
| Maximum No. of bytes transferred | 128 Kbytes (with 16-bit transfers) or 64 Kbytes (with 8-bit transfers) |
| DMA request factors (Note) | Falling edge of $\overline{INT0}$ to $\overline{INT3}$ or both edge<br>Timer A0 to timer A4 interrupt requests<br>Timer B0 to timer B5 interrupt requests<br>UART0 to UART4 transmission and reception interrupt requests<br>A-D conversion interrupt requests<br>Intelligent I/O interrupt<br>Software triggers |
| Channel priority | DMA0 > DMA1 > DMA2 > DMA3 (DMA0 is the first priority) |
| 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<br>Transfer ends when the transfer count register is "$0000_{16}$".<br>• Repeat transfer<br>When the transfer counter is "$0000_{16}$", the value in the transfer counter reload register is reloaded into the transfer counter and the DMA transfer is continued |
| DMA interrupt request generation timing | When the transfer counter register changes from "$0001_{16}$" to "$0000_{16}$". |
| DMA startup | • Single transfer<br>Transfer starts when DMA transfer count register is more than "$0001_{16}$" and the DMA is requested after "$01_2$" is written to the channel i transfer mode select bits<br>• Repeat transfer<br>Transfer starts when the DMA is requested after "$11_2$" is written to the channel i transfer mode select bits |
| DMA shutdown | • Single transfer<br>When "$00_2$" is written to the channel i transfer mode select bits and DMA transfer count register becomes "$0000_{16}$" by DMA transfer or write<br>• Repeat transfer<br>When "$00_2$" is written to the channel i transfer mode select bits |
| Reload timing | When the transfer counter register changes from "$0001_{16}$" to "$0000_{16}$" in repeat transfer mode. |
| Reading / writing the register | Registers are always read/write enabled. |
| Number of DMA transfer cycles | Between SFR and internal RAM : 3 cycles<br>Between external I/O and external memory : minimum 3 cycles |

Note: DMA transfer does not affect any interrupt.

## DMAi request cause select register (i=0 to 3)




| Symbol | Address | When reset |
|---|---|---|
| DMiSL(i=0 to 3) | $0378_{16}$, $0379_{16}$, $037A_{16}$, $037B_{16}$ | $0X000000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DSEL0 | DMA request cause select bit (Note 1) | Refer to function table | ○ | ○ |
| DSEL1 | | | ○ | ○ |
| DSEL2 | | | ○ | ○ |
| DSEL3 | | | ○ | ○ |
| DSEL4 | | | ○ | ○ |
| DSR | Software DMA request bit (Note 2) | 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") | ○ | ○ |
| ——— | Nothing is assigned. When write, set "0". When read, its content is indeterminate. | | – | – |
| DRQ | DMA request bit (Note 2, 3) | 0 : Not requested<br>1 : Requested | ○ | ○ |

Note 1: Set DMA inhibit before changing the DMA request cause. Set DRQ bit to "1" simultaneously.
e.g.) MOV.B #083h, DMiSL ; Set timer A0
Note 2: When setting DSR to "1", set DRQ bit to "1" using OR instruction etc. simultaneously.
e.g.) OR.B #0A0h, DMiSL
Note 3: Do not write "0" to this bit. There is no need to clear the DMA request bit.

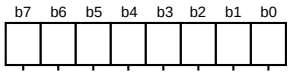
**Figure 1.11.2. DMAC register (1)**

**Table 1.11.2. DMAi request cause select register function**

| Setting value | DMA request cause | | | | |
|---|---|---|---|---|---|
| b4 b3 b2 b1 b0 | DMA0 | DMA1 | DMA2 | DMA3 | |
| 0 0 0 0 0 | Software trigger | | | | |
| 0 0 0 0 1 | Falling edge of INT0 pin | Falling edge of INT1 pin | Falling edge of INT2 pin | Falling edge of INT3 pin | (Note 1) |
| 0 0 0 1 0 | Both edges of INT0 | Both edges of INT1 | Both edges of INT2 | Both edges of INT3 | (Note 1) |
| 0 0 0 1 1 | Timer A0 | | | | |
| 0 0 1 0 0 | Timer A1 | | | | |
| 0 0 1 0 1 | Timer A2 | | | | |
| 0 0 1 1 0 | Timer A3 | | | | |
| 0 0 1 1 1 | Timer A4 | | | | |
| 0 1 0 0 0 | Timer B0 | | | | |
| 0 1 0 0 1 | Timer B1 | | | | |
| 0 1 0 1 0 | Timer B2 | | | | |
| 0 1 0 1 1 | Timer B3 | | | | |
| 0 1 1 0 0 | Timer B4 | | | | |
| 0 1 1 0 1 | Timer B5 | | | | |
| 0 1 1 1 0 | UART0 transmit | | | | |
| 0 1 1 1 1 | UART0 receive /ACK | | | | (Note 2) |
| 1 0 0 0 0 | UART1 transmit | | | | |
| 1 0 0 0 1 | UART1 receive /ACK | | | | (Note 2) |
| 1 0 0 1 0 | UART2 transmit | | | | |
| 1 0 0 1 1 | UART2 receive /ACK | | | | (Note 2) |
| 1 0 1 0 0 | UART3 transmit | | | | |
| 1 0 1 0 1 | UART3 receive /ACK | | | | (Note 2) |
| 1 0 1 1 0 | UART4 transmit | | | | |
| 1 0 1 1 1 | UART4 receive /ACK | | | | (Note 2) |
| 1 1 0 0 0 | A-D0 | A-D1 | A-D0 | A-D1 | |
| 1 1 0 0 1 | Intelligent I/O interrupt control register 0 | Intelligent I/O interrupt control register 7 | Intelligent I/O interrupt control register 2 | Intelligent I/O interrupt control register 9 | |
| 1 1 0 1 0 | Intelligent I/O interrupt control register 1 | Intelligent I/O interrupt control register 8 | Intelligent I/O interrupt control register 3 | Intelligent I/O interrupt control register 10 | |
| 1 1 0 1 1 | Intelligent I/O interrupt control register 2 | Intelligent I/O interrupt control register 9 | Intelligent I/O interrupt control register 4 | Intelligent I/O interrupt control register 11 | |
| 1 1 1 0 0 | Intelligent I/O interrupt control register 3 | Intelligent I/O interrupt control register 10 | Intelligent I/O interrupt control register 5 | Intelligent I/O interrupt control register 0 | |
| 1 1 1 0 1 | Intelligent I/O interrupt control register 4 | Intelligent I/O interrupt control register 11 | Intelligent I/O interrupt control register 6 | Intelligent I/O interrupt control register 1 | |
| 1 1 1 1 0 | Intelligent I/O interrupt control register 5 | Intelligent I/O interrupt control register 0 | Intelligent I/O interrupt control register 7 | Intelligent I/O interrupt control register 2 | |
| 1 1 1 1 1 | Intelligent I/O interrupt control register 6 | Intelligent I/O interrupt control register 1 | Intelligent I/O interrupt control register 8 | Intelligent I/O interrupt control register 3 | |

Note 1: When $\overline{INT3}$ pin is data bus in microprocessor mode, INT3 edge cannot be used as DMA3 request cause.
Note 2: UARTi receive /ACK switched by setting of UARTi special mode register and UARTi special mode register 2 (i=0 to 4)

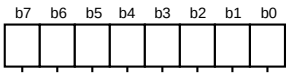
DMA mode register 0
(CPU internal register)




Symbol: DMD0
When reset: $00_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| MD00 | Channel 0 transfer mode select bit | b1 b0<br>0 0 : DMA inhibit<br>0 1 : Single transfer<br>1 0 : Must not be set<br>1 1 : Repeat transfer | ○ | ○ |
| MD01 | | | ○ | ○ |
| BW0 | Channel 0 transfer unit select bit | 0 : 8 bits<br>1 : 16 bits | ○ | ○ |
| RW0 | Channel 0 transfer direction select bit | 0 : Fixed address to memory (forward direction)<br>1 : Memory (forward direction) to fixed address | ○ | ○ |
| MD10 | Channel 1 transfer mode select bit | b5 b4<br>0 0 : DMA inhibit<br>0 1 : Single transfer<br>1 0 : Must not be set<br>1 1 : Repeat transfer | ○ | ○ |
| MD11 | | | ○ | ○ |
| BW1 | Channel 1 transfer unit select bit | 0 : 8 bits<br>1 : 16 bits | ○ | ○ |
| RW1 | Channel 1 transfer direction select bit | 0 : Fixed address to memory (forward direction)<br>1 : Memory (forward direction) to fixed address | ○ | ○ |

DMA mode register 1
(CPU internal register)

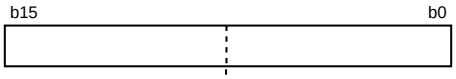



Symbol: DMD1
When reset: $00_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| MD20 | Channel 2 transfer mode select bit | b1 b0<br>0 0 : DMA inhibit<br>0 1 : Single transfer<br>1 0 : Must not be set<br>1 1 : Repeat transfer | ○ | ○ |
| MD21 | | | ○ | ○ |
| BW2 | Channel 2 transfer unit select bit | 0 : 8 bits<br>1 : 16 bits | ○ | ○ |
| RW2 | Channel 2 transfer direction select bit | 0 : Fixed address to memory (forward direction)<br>1 : Memory (forward direction) to fixed address | ○ | ○ |
| MD30 | Channel 3 transfer mode select bit | b5 b4<br>0 0 : DMA inhibit<br>0 1 : Single transfer<br>1 0 : Must not be set<br>1 1 : Repeat transfer | ○ | ○ |
| MD31 | | | ○ | ○ |
| BW3 | Channel 3 transfer unit select bit | 0 : 8 bits<br>1 : 16 bits | ○ | ○ |
| RW3 | Channel 3 transfer direction select bit | 0 : Fixed address to memory (forward direction)<br>1 : Memory (forward direction) to fixed address | ○ | ○ |

**Figure 1.11.3. DMAC register (2)**

DMAi transfer count register (i = 0 to 3)
(CPU internal register)

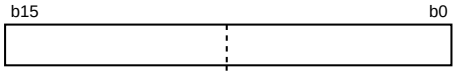



| Symbol | | Address | When reset |
|---|---|---|---|
| DCT0 | (Note 2) | (CPU internal register) | XXXX16 |
| DCT1 | (Note 2) | (CPU internal register) | XXXX16 |
| DCT2 (bank 1;R0) | (Note 3) | (CPU internal register) | 000016 |
| DCT3 (bank 1;R1) | (Note 3) | (CPU internal register) | 000016 |

| Function | Setting range | R | W |
|---|---|---|---|
| Set transfer number | 000016 to FFFF16 | O | O |

Note 1: When "0" is set to this register, data transfer is not done even if DMA is requested.
Note 2: Use LDC instruction to write to this register.
Note 3: When setting DCT2 and DCT3, set "1" to the register bank select flag (B flag) of flag register (FLG), then set desired value to R0 and R1 of register bank 1. Use MOV instruction to write to this register.

DMAi transfer count reload register (i = 0 to 3)
(CPU internal register)




| Symbol | | Address | When reset |
|---|---|---|---|
| DRC0 | (Note 1) | (CPU internal register) | XXXX16 |
| DRC1 | (Note 1) | (CPU internal register) | XXXX16 |
| DRC2 (bank 1;R2) | (Note 2) | (CPU internal register) | 000016 |
| DRC3 (bank 1;R3) | (Note 2) | (CPU internal register) | 000016 |

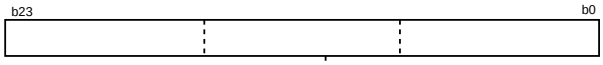
| Function | Setting range | R | W |
|---|---|---|---|
| Set transfer number | 000016 to FFFF16 | O | O |

Note 1: Use LDC instruction to write to this register.
Note 2: When setting DRC2 and DRC3, set "1" to the register bank select flag (B flag) of flag register (FLG), then set desired value to R2 and R3 of register bank 1. Use MOV instruction to write to this register.

**Figure 1.11.4. DMAC register (3)**

DMAi memory address register (i = 0 to 3)
(CPU internal register)

| Symbol | | Address | When reset |
|---|---|---|---|
| DMA0 | (Note 2) | (CPU internal register) | XXXXXX16 |
| DMA1 | (Note 2) | (CPU internal register) | XXXXXX16 |
| DMA2 (bank 1;A0) | (Note 3) | (CPU internal register) | 00000016 |
| DMA3 (bank 1;A1) | (Note 3) | (CPU internal register) | 00000016 |

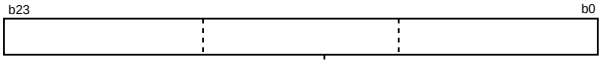



| Function | Setting range | R | W |
|---|---|---|---|
| Set source or destination memory address (Note 1) | 00000016 to FFFFFF16 (16 Mbytes area) | O | O |

Note 1: When the transfer direction select bit is "0" (fixed address to memory), this register is destination memory address.
When the transfer direction select bit is "1" (memory to fixed address), this register is source memory address.
Note 2: Use LDC instruction to write to this register.
Note 3: When setting DMA2 and DMA3, set "1" to the register bank select flag (B flag) of flag register (FLG), and set desired value to A0 and A1 of register bank 1. Use MOV instruction to write to this register.

DMAi SFR address register (i = 0 to 3)
(CPU internal register)

| Symbol | | Address | When reset |
|---|---|---|---|
| DSA0 | (Note 2) | (CPU internal register) | XXXXXX16 |
| DSA1 | (Note 2) | (CPU internal register) | XXXXXX16 |
| DSA2 (bank 1;SB) | (Note 3) | (CPU internal register) | 00000016 |
| DSA3 (bank 1;FB) | (Note 4) | (CPU internal register) | 00000016 |

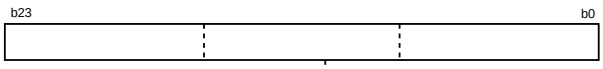



| Function | Setting range | R | W |
|---|---|---|---|
| Set source or destination fixed address (Note 1) | 00000016 to FFFFFF16 (16 Mbytes area) | O | O |

Note 1: When the transfer direction select bit is "0" (fixed address to memory), this register is source fixed address.
When the transfer direction select bit is "1" (memory to fixed address), this register is destination fixed address.
Note 2: Use LDC instruction to write to this register.
Note 3: When setting DSA2, set "1" to the register bank select flag (B flag) of flag register (FLG), and set desired value to SB of register bank 1. Use LDC instruction to write to this register.
Note 4: When setting DSA3, set "1" to the register bank select flag (B flag) of flag register (FLG), and set desired value to FB of register bank 1. Use LDC instruction to write to this register.

DMAi memory address reload register (i = 0 to 3) (Note 1)
(CPU internal register)

| Symbol | Address | When reset |
|---|---|---|
| DRA0 | (CPU internal register) | XXXXXX16 |
| DRA1 | (CPU internal register) | XXXXXX16 |
| DRA2 (bank 1;SVP) (Note 2) | (CPU internal register) | XXXXXX16 |
| DRA3 (bank 1;VCT) (Note 3) | (CPU internal register) | XXXXXX16 |




| Function | Setting range | R | W |
|---|---|---|---|
| Set source or destination memory address | 00000016 to FFFFFF16 (16 Mbytes area) | O | O |

Note 1: Use LDC instruction to write to this register.
Note 2: When setting DRA2, set desired value to save PC register (SVP).
Note 3: When setting DRA3, set desired value to vector register (VCT).

**Figure 1.11.5. DMAC register (4)**

## (1) Transfer cycle

The transfer cycle consists of the bus cycle in which data is read from memory or from the SFR area (source read) and the bus cycle in which the data is written to memory or to the SFR area (destination write). The number of read and write bus cycles depends on the source and destination addresses. In memory expansion mode and microprocessor mode, the number of read and write bus cycles also depends on the level of the BYTE pin. Also, the bus cycle 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 external data bus width control register**

When in memory expansion mode or microprocessor mode, the transfer cycle changes according to the data bus width at the source and destination.

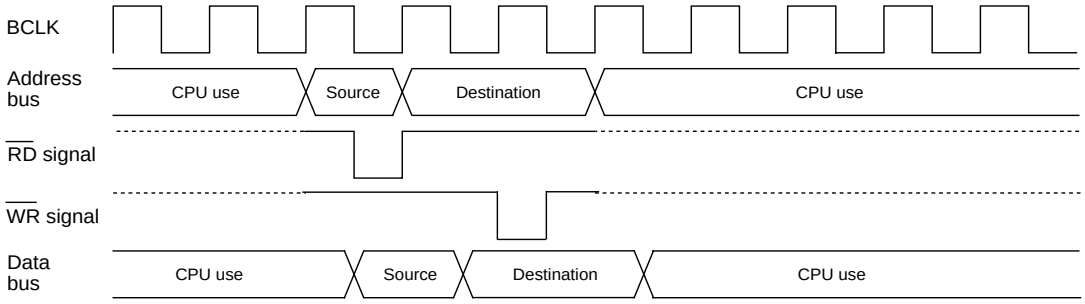
1. When transferring 16 bits of data and the data bus width at the source and at the destination is 8 bits (data bus width bit = “0”), there are two 8-bit data transfers. Therefore, two bus cycles are required for reading and two cycles for writing.
2. When transferring 16 bits of data and the data bus width at the source is 8 bits (data bus width bit = “0”) and the data bus width at the destination is 16 bits (data bus width bit = “1”), the data is read in two 8-bit blocks and written as 16-bit data. Therefore, two bus cycles are required for reading and one cycle for writing.
3. When transferring 16 bits of data and the data bus width at the source is 16 bits (data bus width bit = “1”) and the data bus width at the destination is 8 bits (data bus width bit = “0”), 16 bits of data are read and written as two 8-bit blocks. Therefore, one bus cycle is required for reading and two cycles for writing.

**(c) Effect of software wait**

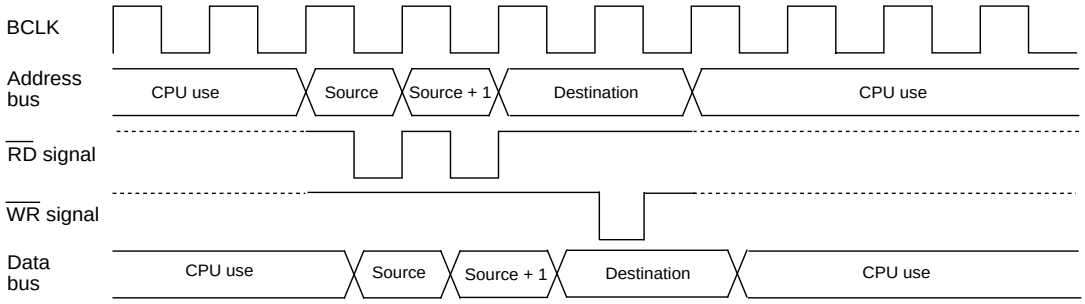
When the SFR area or a memory area with a software wait is accessed, the number of cycles is increased for the software wait by 1 bus cycle. The length of the cycle is determined by BCLK.

Figure 1.11.6 shows the example of the transfer cycles for a source read. Figure 1.11.6 shows the destination is external area, the destination write cycle is shown as two cycle (one bus cycle) and the source read cycles for the different conditions. In reality, the destination write cycle is subject to the same conditions as the source read cycle, with the transfer cycle changing accordingly. When calculating the transfer cycle, remember to apply the respective conditions to both the destination write cycle and the source read cycle. For example (2) in Figure 1.11.6, 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.

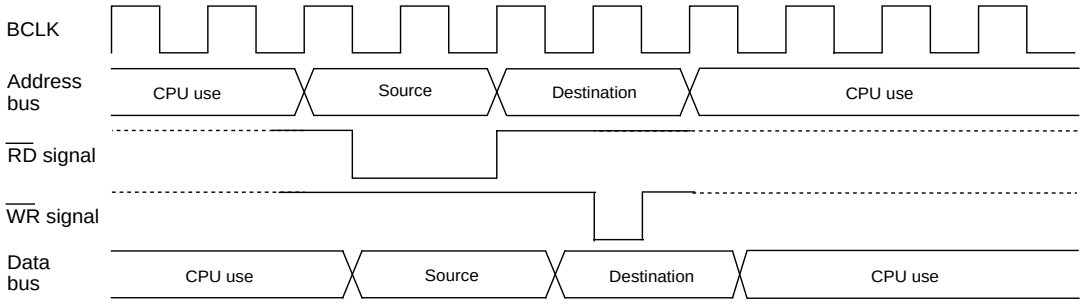
**(1) •When 8-bit data is transferred**
**•When 16-bit data is transferred on a 16-bit data bus and the source address is even**



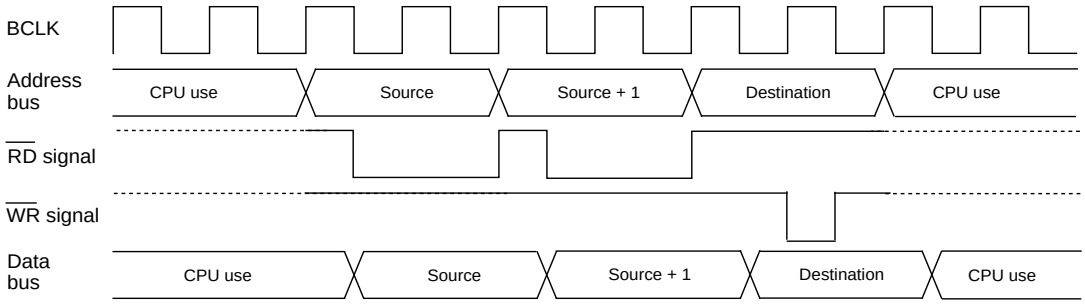

**(2) •When 16-bit data is transferred and the source address is odd**
**•When 16-bit data is transferred and the width of data bus at the source is 8-bit**
**(When the width of data bus at the destination is 8-bit, there are also two destination write cycles).**




**(3) •When one wait is inserted into the source read under the conditions in (1)**




**(4) •When one wait is inserted into the source read under the conditions in (2)**
**(When 16-bit data is transferred and the width of data but at the destination is 8-bit, there are two destination write cycles).**




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

**Figure 1.11.6. Example of the transfer cycles for a source read**

## (2) DMAC transfer cycles

Any combination of even or odd transfer read and write addresses is possible. Table 1.11.2 shows the number of DMAC transfer cycles.

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

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

**Table 1.11.2. No. of DMAC transfer cycles**

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

**Coefficient j, k**

| | | | Coefficient j | Coefficient k |
|---|---|---|---|---|
| Internal memory | Internal ROM/RAM | No wait | 1 | 1 |
| | Internal ROM/RAM | One wait | 2 | 2 |
| | SFR area | | 2 | 2 |
| External memory | Separate bus | No wait | 1 | 2 |
| | Separate bus | One wait | 2 | 2 |
| | Separate bus | Two waits | 3 | 3 |
| | Separate bus | Three waits | 4 | 4 |
| | Multiplex bus | | 3 | 3 |

## DMA Request Bit

The DMAC can issue DMA requests using preselected DMA request factors for each channel as triggers. The DMA transfer request factors include the reception of DMA request signals from the internal peripheral functions, software DMA factors generated by the program, and external factors using input from external interrupt signals.

See the description of the DMAi factor selection register for details of how to select DMA request factors. DMA requests are received as DMA requests when the DMAi request bit is set to "1" and the channel i transfer mode select bits are "01" or "11". Therefore, even if the DMAi request bit is "1", no DMA request is received if the channel i transfer mode select bit is "00". In this case, DMAi request bit is cleared. Because the channel i transfer mode select bits default to "00" after a reset, remember to set the channel i transfer mode select bit for the channel to be activated after setting the DMAC related registers. This enables receipt of the DMA requests for that channel, and DMA transfers are then performed when the DMAi request bit is set.

The following describes when the DMAi request bit is set and cleared.

**(1) Internal factors**

The DMAi request flag is set to “1” in response to internal factors at the same time as the interrupt request bit of the interrupt control register for each factor is set. This is because, except for software trigger DMA factors, they use the interrupt request signals output by each function.

The DMAi request bit is cleared to "0" when the DMA transfer starts or the DMA transfer is disabled (channel i transfer mode select bits are "00" and the DMAi transfer count register is "0").

**(2) External factors**

These are DMA request factors that are generated by the input edge from the $\overline{\text{INTi}}$ pin (where i indicates the DMAC channel). When the $\overline{\text{INTi}}$ pin is selected by the DMAi request factor select bit as an external factor, the inputs from these pins become the DMA request signals.

When an external factor is selected, the DMAi request bit is set, according to the function specified in the DMA request factor select bit, on either the falling edge of the signal input via the $\overline{\text{INTi}}$ pins, or both edges.

When an external factor is selected, the DMAi request bit is cleared, in the same way as the DMAi request bit is cleared for internal factors, when the DMA transfer starts or the DMA transfer is in disable state.

**(3) Relationship between external factor request input and DMAi request bits, and DMA transfer timing**

When the request inputs to DMAi occur in the same sampling cycle (between the falling edge of BCLK and the next falling edge), the DMAi request bits are set simultaneously, but if the DMAi enable bits are all set, DMA0 takes priority and the transfer starts. When one transfer unit is complete, the bus privilege is returned to the CPU. When the CPU has completed one bus access, DMA1 transfer starts, and, when one transfer unit is complete, the privilege is again returned to the CPU.

The priority is as follows: DMA0 > DMA1 > DMA2 > DMA3.

Figure 1.11.7. DMA transfer example by external factors shows what happens when DMA0 and DMA1 requests occur in the same sampling cycle.

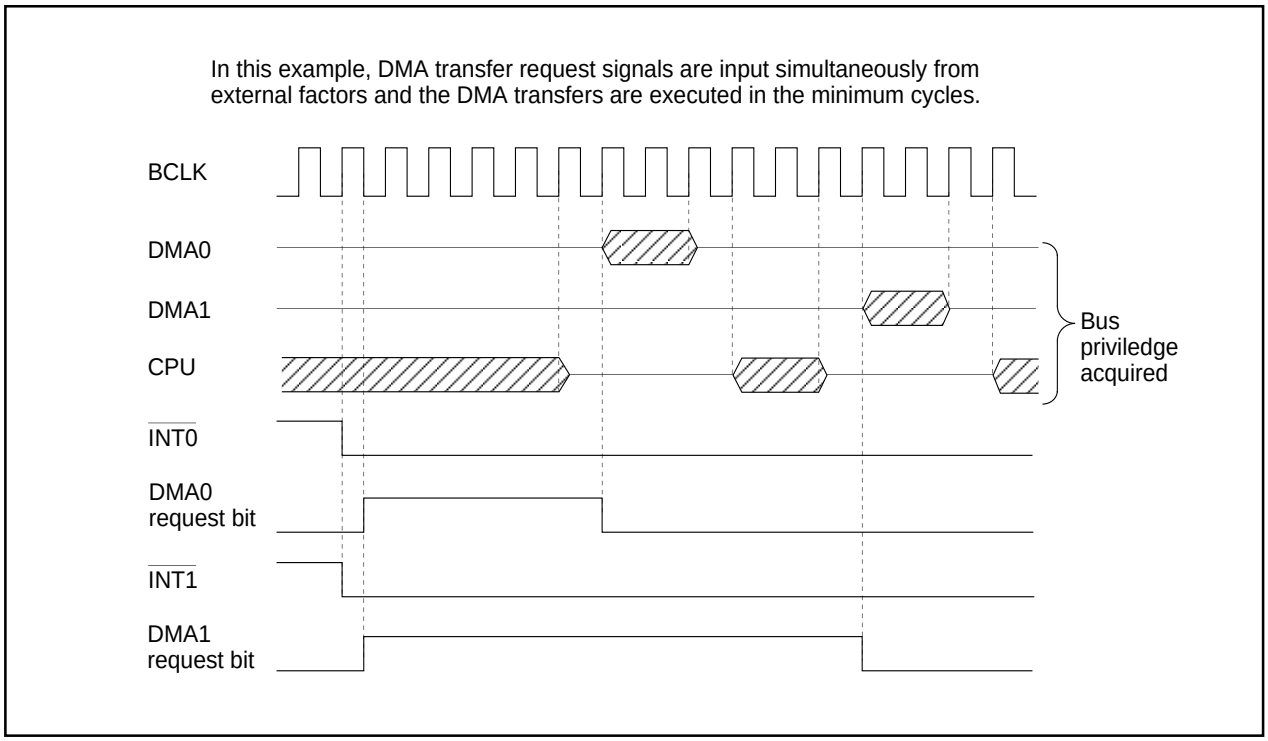



**Figure 1.11.7. DMA transfer example by external factors**

## Precautions for DMAC

(1) Do not clear the DMA request bit of the DMAi request cause select register.
In M32C/83, when a DMA request is generated while the channel is disabled (Note), the DMA transfer is not executed and the DMA request bit is cleared automatically.
Note :The DMA is disabled or the transfer count register is "0".

(2) When DMA transfer is done by a software trigger, set DSR and DRQ of the DMAi request cause select register to "1" simultaneously using the OR instruction.

```
e.g.) OR.B   #0A0h, DMiSL          ; DMiSL is DMAi request cause select register
```

(3) When changing the DMAi request cause select bit of the DMAi request cause select register, set "1" to the DMA request bit, simultaneously. In this case, the corresponding DMA channel is set to disabled. At least 8 + 6 x N (N: enabled channel number) clock cycles are needed from the instruction to write to the DMAi request cause select bit to enable DMA.

e.g.) When DMA request cause is changed to timer A0 and using DMA0 in single transfer after DMA initial setting

```
push.w   R0                 ; Store R0 register
stc      DMD0, R0           ; Read DMA mode register 0
and.b    #11111100b, R0L    ; Clear DMA0 transfer mode select bit to "00"
ldc      R0, DMD0           ; DMA0 disabled
mov.b    #10000011b, DM0SL  ; Select timer A0
                            ; (Write "1" to DMA request bit simultaneously)
nop                                  } At least 8 + 6 x N cycles
 :                                   } (N: enabled channel number)
ldc      R0, DMD0           ; DMA0 enabled
pop.w    R0                 ; Restore R0 register
```

## DMAC II

When requested by an interrupt from any peripheral I/O, the DMAC performs a memory-to-memory transfer, an immediate data transfer, or an arithmetic transfer (to transfer the sum of two data added). Specifications of DMAC II are shown in Table 1.12.1.

**Table 1.12.1 Specifications of DMAC II**

| Item | Specification |
|---|---|
| Causes to activate DMAC II | Interrupt request from any peripheral I/O whose interrupt priority is set to "level 7" by the Interrupt Control Register |
| Transfer data | (1) Memory -> memory (memory-to-memory transfer)<br>(2) Immediate data -> memory (immediate data transfer)<br>(3) Memory (or immediate data) + memory -> memory (arithmetic transfer) |
| Unit of transfer | Transferred in 8 or 16 bits |
| Transfer space | 64-Kbyte space at address up to $0FFFF_{16}$ (Note) |
| Direction of transfer | Fixed or forward address<br>Can be selected individually for the source and the destination of transfer. |
| Transfer mode | (1) Single transfer<br>(2) Burst transfer |
| Chained transfer function | Parameters (transfer count, transfer address, and other information) are switched over when the transfer counter reaches zero. |
| Interrupt at end of transfer | Interrupt is generated when the transfer counter reaches zero. |
| Multiple transfer function | Multiple data transfers can be performed by one DMA II transfer request generated. |

Note : When transfer unit is 16 bits and destination address is $0FFFF_{16}$, data is transfered to addresses $0FFFF_{16}$ and $10000_{16}$. When source address is $0FFFF_{16}$, data is transfered as in the previous.

## Settings of DMAC II

DMAC II can be enabled for use by setting up the following registers and tables.

- Exit Priority Register (address $009F_{16}$)
- DMAC II Index
- Interrupt Control Register for the peripheral I/O that requests a transfer by DMAC II
- Relocatable Vector Table for the peripheral I/O that requests a transfer by DMAC II
- When using an intelligent I/O or CAN interrupt, Interrupt Enable Register's interrupt request latch bit (bit 0)

**(1) Exit priority register (address $009F_{16}$)**

If this register's DMAC II select bit (bit 5) and fast interrupt select bit (bit 3) respectively are set to 1 and 0, DMAC II is activated by an interrupt request from any peripheral I/O whose interrupt priority is set to "level 7" by the interrupt priority level select bit.

The configuration of the exit priority register is shown in Figure 1.12.1.

## Exit priority register

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

| Symbol | Address | When reset |
|---|---|---|
| RLVL | $009F_{16}$ | $XX0X0000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| RLVL0 | Interrupt priority set bits for exiting Stop/Wait state (Note 1) | b2 b1 b0<br>0 0 0 : Level 0<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 | ○ | ○ |
| RLVL1 | | | ○ | ○ |
| RLVL2 | | | ○ | ○ |
| FSIT | High-speed interrupt set bit (Note 2) | 0: Interrupt priority level 7 = normal interrupt<br>1: Interrupt priority level 7 = high-speed interrupt | ○ | ○ |
| Nothing is assigned. When write, set "0". When read, its content is indeterminate. | | | — | — |
| DMA II | DMA II select bit (Note 3) | 0: Interrupt priority level 7 = normal interrupt or high-speed interrupt<br>1: Interrupt priority level 7 = DMA II transfer | ○ | ○ |
| Nothing is assigned. When write, set "0". When read, their contents are indeterminate. | | | — | — |
| | | | — | — |

Note 1: Exits the Stop or Wait mode when the requested interrupt priority level is higher than that set in the exit priority register.
Set to the same value as the processor interrupt priority level (IPL) set in the flag register (FLG).

Note 2: The high-speed interrupt can only be specified for interrupts with interrupt priority level 7. Specify interrupt priority level 7 for only one interrupt.

Note 3: Do not set this bit to 0 after once setting it to 1.
When this bit is 1, set the high-speed interrupt select bit to 0. (This cannot be used simultaneously with the high-speed interrupt.)
Transfers by DMAC II are unaffected by the interrupt enable flag (I flag) and processor interrupt priority level (IPL).

**Figure 1.12.1. Exit priority register**

**(2) DMAC II Index**

The DMAC II Index is a data table, comprised of 8 to 18 bytes (max. 32 kbytes when multiple transfer function is selected), which contains such parameters as transfer mode, transfer counter, transfer source address (or immediate data), operation address, transfer destination address, chained transfer address, and end-of-transfer interrupt address.

This DMAC II Index is located in the RAM area.

Configuration of the DMAC II Index is shown in Figure 1.12.2. The configuration of the DMAC II Index by transfer mode is shown in Table 1.12.2.

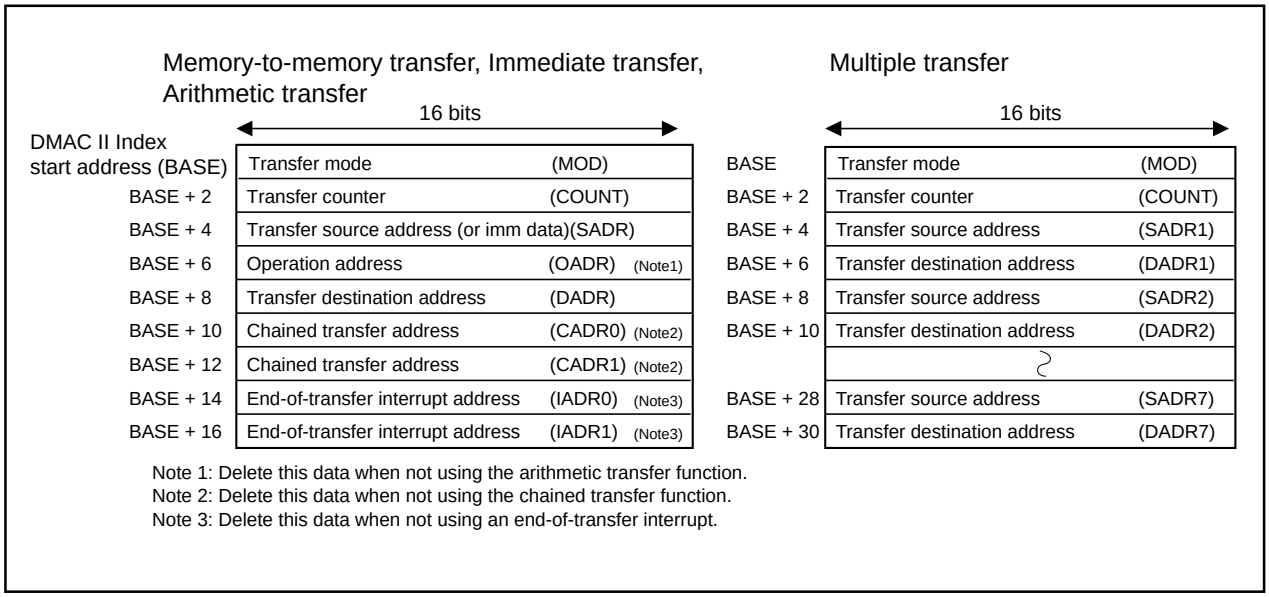



**Figure 1.12.2. DMAC II index**

- Transfer mode (MOD)

  This two-byte data sets DMAC II transfer mode. Configuration of transfer modes is shown in Figure 1.12.3.
- Transfer counter (COUNT)

  This two-byte data sets the number of times transfer is performed.
- Transfer source address (SADR)

  This two-byte data sets the memory address from which data is transferred or immediate data.
- Operation address (OADR)

  This two-byte data sets the memory address to be operated on for calculation. This data is added to the table only when using the arithmetic transfer function.
- Transfer destination address (DADR)

  This two-byte data sets the memory address to which data is transferred.
- Chained transfer address (CADR)

  This four-byte data sets the DMAC II Index start address for the next DMAC II transfer to be performed. This data is added to the table only when using the chained transfer function.
- End-of-transfer interrupt address (IADR)

  This four-byte data sets the jump address for end-of-transfer interrupt processing. This data is added to the table only when using an end-of-transfer interrupt.

**Table 1.12.2. The configuration of the DMAC II Index by transfer mode**

| Transmit data | Memory-to-memory transfer /immediate data transfer | | | | Arithmetic transfer | | | | Multiple transfer |
|---|---|---|---|---|---|---|---|---|---|
| Chained transfer | Not use | Use | Not use | Use | Not use | Use | Not use | Use | Cannot use |
| Interrupt at end of transfer | Not use | Not use | Use | Use | Not use | Not use | Use | Use | Cannot use |
| DMAC II index | MOD<br>COUNT<br>SADR<br>DADR<br>8 bytes | MOD<br>COUNT<br>SADR<br>DADR<br>CADR0<br>CADR1<br>12 bytes | MOD<br>COUNT<br>SADR<br>DADR<br>IADR0<br>IADR1<br>12 bytes | MOD<br>COUNT<br>SADR<br>DADR<br>CADR0<br>CADR1<br>IADR0<br>IADR1<br>16 bytes | MOD<br>COUNT<br>SADR<br>OADR<br>DADR<br>10 bytes | MOD<br>COUNT<br>SADR<br>OADR<br>DADR<br>CADR0<br>CADR1<br>14 bytes | MOD<br>COUNT<br>SADR<br>OADR<br>DADR<br>IADR0<br>IADR1<br>14 bytes | MOD<br>COUNT<br>SADR<br>OADR<br>DADR<br>CADR0<br>CADR1<br>IADR0<br>IADR1<br>18 bytes | MOD<br>COUNT<br>SADR1<br>DADR1<br>SADRi<br>DADRi<br>i=1 to 7<br>Max. 32 bytes<br>(when i=7) |

## Transfer mode(MOD)

b15 (b7) b8 (b0) b7 b0

| Bit symbol | Bit name | Function (MULT=0) | Function (MULT=1) | R | W |
|---|---|---|---|---|---|
| SIZE | Transfer unit select bit | 0: 8 bits<br>1: 16 bits | | ○ | ○ |
| IMM | Transfer data select bit | 0: Immediate<br>1: Memory | Must set to "1" | ○ | ○ |
| UPDS | Transfer source direction select bit | 0: Fixed address<br>1: Forward address | | ○ | ○ |
| UPDD | Transfer destination direction select bit | 0: Fixed address<br>1: Forward address | | ○ | ○ |
| OPER /CNT0 | Arithmetic transfer function select bit | 0: Not used<br>1: Used | b6 b5 b4<br>0 0 0: Do not set this<br>0 0 1: Once<br>0 1 0: Twice<br>:<br>:<br>1 1 0: 6 times<br>1 1 1: 7 times | ○ | ○ |
| BRST /CNT1 | Burst transfer select bit | 0: Single transfer<br>1: Burst transfer | | ○ | ○ |
| INTE /CNT2 | End of transfer interrupt select bit | 0: Interrupt not used<br>1: Interrupt used | | ○ | ○ |
| CHAIN | Chained transfer select bit | 0: Chained transfer not used<br>1: Chained transfer used | Must set to "0" | ○ | ○ |
| ____ | Nothing is assigned. When write, set "0".<br>When read, their contents are indeterminate. | | | _ | _ |
| MULT | Multiple transfer select bit | 0: Not multiple transfer | 1: Multiple transfer | ○ | ○ |

**Figure 1.12.3. Transfer mode**

**(3) Interrupt Control Register for Peripheral I/O**

For peripheral I/O interrupts used to request a transfer by DMAC II, set the Interrupt Control Register for each peripheral I/O to select “level 7” for their interrupt priority.

**(4) Relocatable Vector Table for Peripheral I/O**

In the relocatable vector table for each peripheral I/O that requests a transfer by DMAC II, set the DMAC II Index start address. (When using chained transfers, the relocatable vector table must be located in the RAM.)

**(5) Interrupt Enable Register’s interrupt request latch bit (bit 0)**

When using an intelligent I/O or CAN interrupt to activate DMAC II, set to 0 the Interrupt Enable Register’s interrupt request latch bit (bit 0) for the intelligent I/O or CAN interrupt that requests a transfer by DMAC II.

## Operation of DMAC II

The DMAC II function is selected by setting the DMAC II select bit (bit 5 at address $009F_{16}$) to 1. All peripheral I/O interrupt requests which have had their interrupt priorities set to “level 7” by the Interrupt Control Register comprise DMAC II interrupt requests. These interrupt requests (priority level = 7) do not generate an interrupt, however.

When an interrupt request is generated by any peripheral I/O whose interrupt priority is set to “level 7,” DMAC II is activated no matter which state the I flag and processor interrupt priority level(IPL) is in. If an interrupt request with higher priority than that (e.g., $\overline{\text{NMI}}$ or watchdog timer) occurs, this higher priority interrupt has precedence over and is accepted before DMAC II transfers. The pending DMAC II transfer is started after the interrupt processing sequence for that interrupt finishes.

## Transfer data

DMAC II transfers data in units of 8 or 16 bits as described below.

- Memory-to-memory transfer: Data is transferred from any memory location in the 64-Kbyte space to any memory location in the same space.
- Immediate data transfer: Data is transferred as immediate data to any memory location in the 64-Kbyte space.
- Arithmetic transfer: Two 8 or16 bits of data are added together and the result is transferred to any memory location in the 64-Kbyte space.

When transfer unit is 16 bits and destination address is $0FFFF_{16}$, data is transfered to addresses $0FFFF_{16}$ and $10000_{16}$. When source address is $0FFFF_{16}$, data is transfered as in the previous.

**(1) Memory-to-memory transfer**

Data can be transferred from any memory location in the 64-Kbyte space to any memory location in the same space in one of the following four ways:

- Transfer from a fixed address to another fixed address
- Transfer from a fixed address to a variable address
- Transfer from a variable address to a fixed address
- Transfer from a variable address to another variable address

If variable address mode is selected, the transfer address is incremented for the next DMA II transfer to be performed. When transferred in units of 8 bits, the transfer address is incremented by one; when transferred in units of 16 bits, the transfer address is incremented by two. If the transfer source or destination address exceeds $0FFFF_{16}$ as a result of address incrementation, the transfer source or destination address recycles back to $00000_{16}$.

**(2) Immediate data transfer**

Data is transferred as immediate data to any memory location in the 64-Kbyte space. A fixed or variable address can be selected for the transfer destination address. Store the immediate data in the DMAC II Index's transfer source address. When transferring 8-bit immediate data, set the data in the lower byte position of the transfer source address. (The upper byte is ignored.)

**(3) Arithmetic transfer**

Data in two memory locations of the 64-Kbyte space or immediate data and data in any memory location of the 64-Kbyte space are added together and the result is transferred to any memory location in the 64-Kbyte space. Set the memory location to be operated on or immediate data in the DMAC II Index's transfer source address field and the other memory location to be operated on in the DMAC II Index's operation address field. When performing this mode of transfer on two memory locations, a fixed or variable address can be selected for the transfer source and transfer destination addresses. If the transfer source address is chosen to be variable, the operation address also becomes variable. When performing this mode of transfer on immediate data and any memory location, a fixed or variable address can be selected for the transfer destination address.

## Transfer modes

DMAC II supports single and burst transfers. Use the burst transfer select bit (bit 5) for transfer mode setup in the DMAC II index to choose single or burst transfer mode. Use the DMAC II index transfer counter to set the number of times a transfer is performed. Neither single transfer nor burst transfer is performed if the value "$0000_{16}$" is set in the transfer counter.

**(1) Single transfer**

For a DMAC II transfer request, 8 or 16 bits of data (one transfer unit) is transferred once. If the transfer source or transfer destination address is chosen to be variable, the next DMA II transfer is performed on an incremented memory address.

The transfer counter is decremented by each DMA II transfer performed. When using the end-of-transfer interrupt facility, an end-of-transfer interrupt is generated at the time the transfer counter reaches zero.

**(2) Burst transfer**

For a DMAC II transfer request, data transfers are performed in succession a number of times as set by the DMAC II Index transfer counter. When using the end-of-transfer interrupt facility, an end-of-transfer interrupt is generated at the time a burst transfer finishes (i.e., when the transfer counter reaches zero after being decremented for each data transfer performed).

**(3) Multiple transfers**

For multiple transfers, use the multiple transfer select bit (bit 15) for transfer mode setup in the DMAC II Index. Setting this bit to 1 selects the multiple transfer function. For the multiple transfer function, memory to memory transfer can be performed.

Multiple transfers are performed for one DMAC II transfer request received. Use DMAC II Index transfer mode bits 4–6 to set the number of transfers to be performed. (Setting these bits to 001 performs one transfer; setting these bits to 111 performs 7 transfers. Setting these bits to 000 is inhibited.)

The transfer source and transfer destination addresses are alternately incremented beginning with the DMAC II Index BASE address + 4 (as many times as the number of transfers performed).

When using multiple transfer function, arithmetic transfer, burst transfer, end-of-transfer interrupt and chained transfer cannot be used.

**(4) Chained transfer**

For chained transfers, use the chained transfer select bit (bit 7) for transfer mode setup in the DMAC II Index. Setting this bit to 1 selects the chained transfer function. The following describes how a chained transfer is performed.

1) When a DMA II transfer request (interrupt request from any peripheral I/O) is received, a DMAC II Index transfer is performed corresponding to the received request.
2) When the DMAC II Index transfer counter reaches zero, the chained transfer address in the DMAC II Index (i.e., the start address of the DMAC II Index that contains a description of the next DMAC II transfer to be performed) is written to the relocatable vector table for the peripheral I/O.
3) From the next DMA II transfer request on, transfers are performed based on the DMAC II Index indicated by the rewritten relocatable vector table of the peripheral I/O.

Before the chained transfer function can be used, the relocatable vector table must be located in the RAM area.

**(5) End-of-transfer interrupt**

For end-of-transfer interrupts, use the end-of-transfer interrupt select bit (bit 6) for transfer mode setup in the DMAC II Index. Setting this bit to 1 selects the end-of-transfer interrupt function. Set the jump address for end-of-transfer interrupt processing in the DMAC II Index's end-of-transfer interrupt address field. An end-of-transfer interrupt is generated when the DMAC II Index transfer counter reaches zero.

## Execution time

The number of DMAC II execution cycles is calculated by the equation below.

For other than multiple transfers, t = 6 + (26 + A + B + C + D) X m + (4 + E) X n (cycles)
For multiple transfers, t = 21 + (11 + B + C) X k (cycles)
where
A: If the source of transfer is immediate data, A = 0; if it is memory, A = –1
B: If the source address of transfer is a variable address, B = 0; if it is a fixed address, B = 1
C: If the destination address of transfer is a variable address, C = 0; if it is a fixed address, C = 1
D: If the arithmetic function is not selected, D = 0; if the arithmetic function is selected and the source of transfer is immediate data or fixed address memory, D = 7; if the arithmetic function is selected and the source of transfer is variable address memory, D = 8
E: If the chained transfer function is not selected, E = 0; if the chained transfer function is selected, E = 4
m: For single transfer, m = 1; for burst transfer, m = the value set by the transfer counter
n: If the transfer count is one, n = 0; if the transfer count is two or greater, n = 1
k: Number of transfers set by transfer mode bits 4–7

The above equation applies only when all of the following conditions are met, however.
• No bus wait states are inserted.
• The DMAC II Index is set to an even address.
• During word transfer, the transfer source address, transfer destination address, and operation address all are set to an even address.
Note that the first instruction in end-of-transfer interrupt processing is executed 7 cycles after DMAC II transfers are completed.

When using an end-of-transfer interrupt (transfer counter = 2) after performing a memory to memory single transfer twice from a variable source address to a fixed destination address, with the chained transfer function unselected

A=0 B=1 C=0 D=0 E=0

First DMAC II transfer t=6+27X1+4X1=37 cycle
Second DMAC II transfer t=6+27X1+4X0=33 cycle

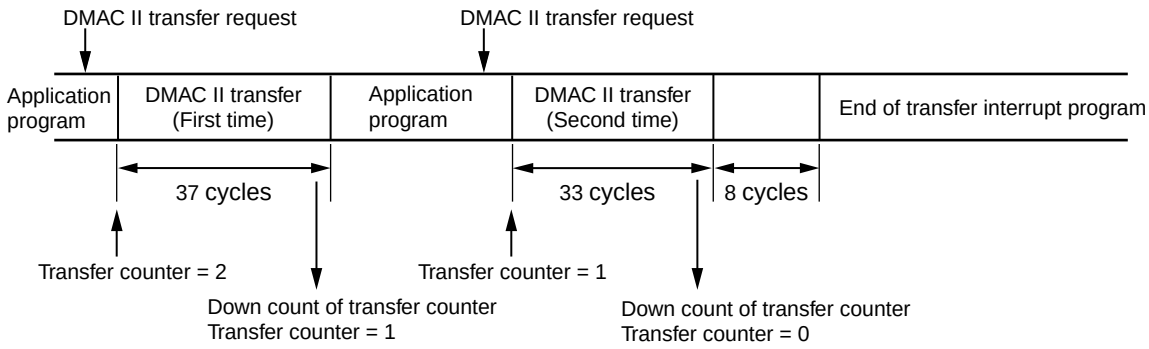



**Figure 1.12.4. Transfer Time**

## Timer

There are eleven 16-bit timers. These timers can be classified by function into timers A (five) and timers B (six). All these timers function independently. Figures 1.13.1 and 1.13.2 show the block diagram of timers.

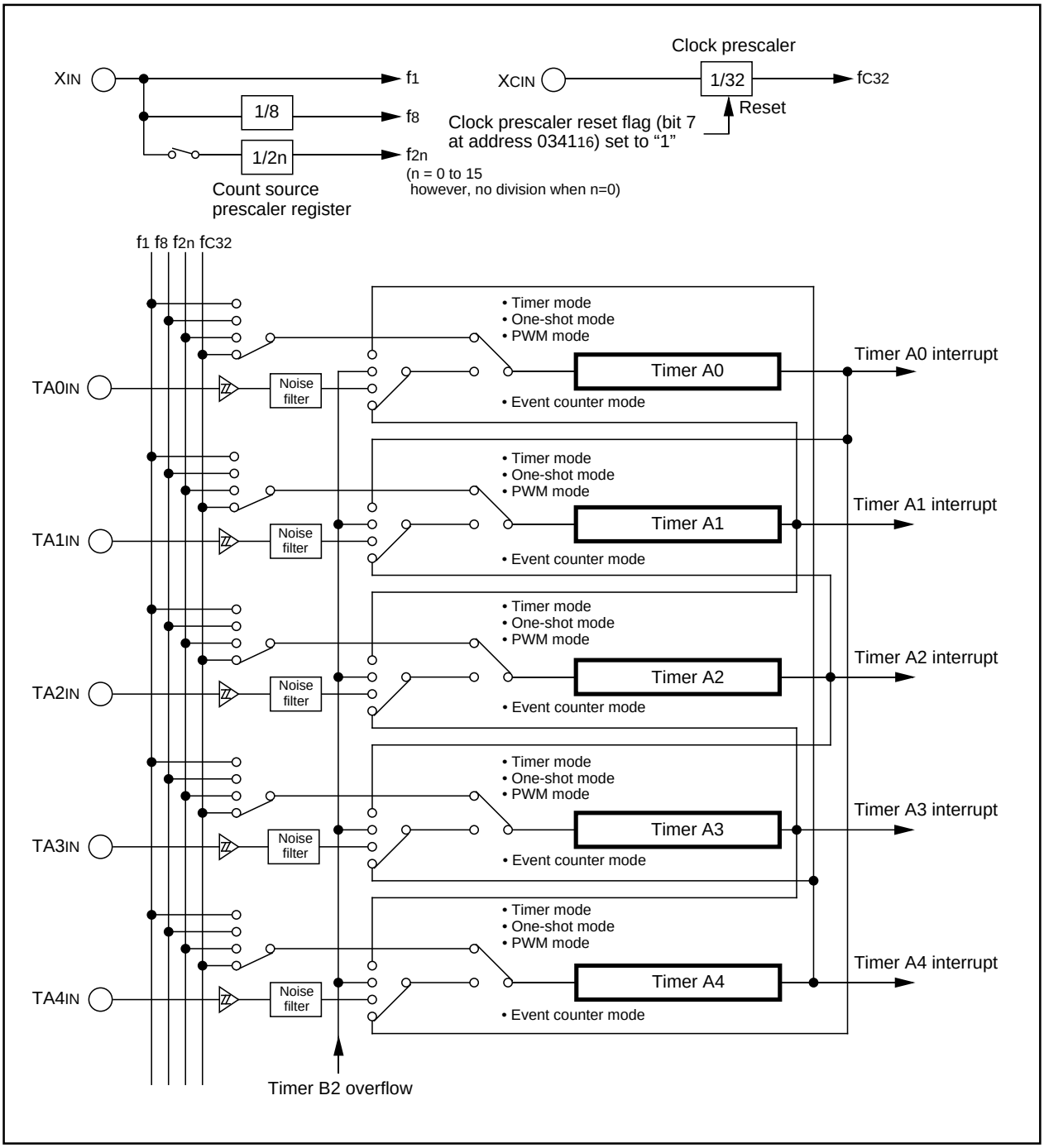



**Figure 1.13.1. Timer A block diagram**

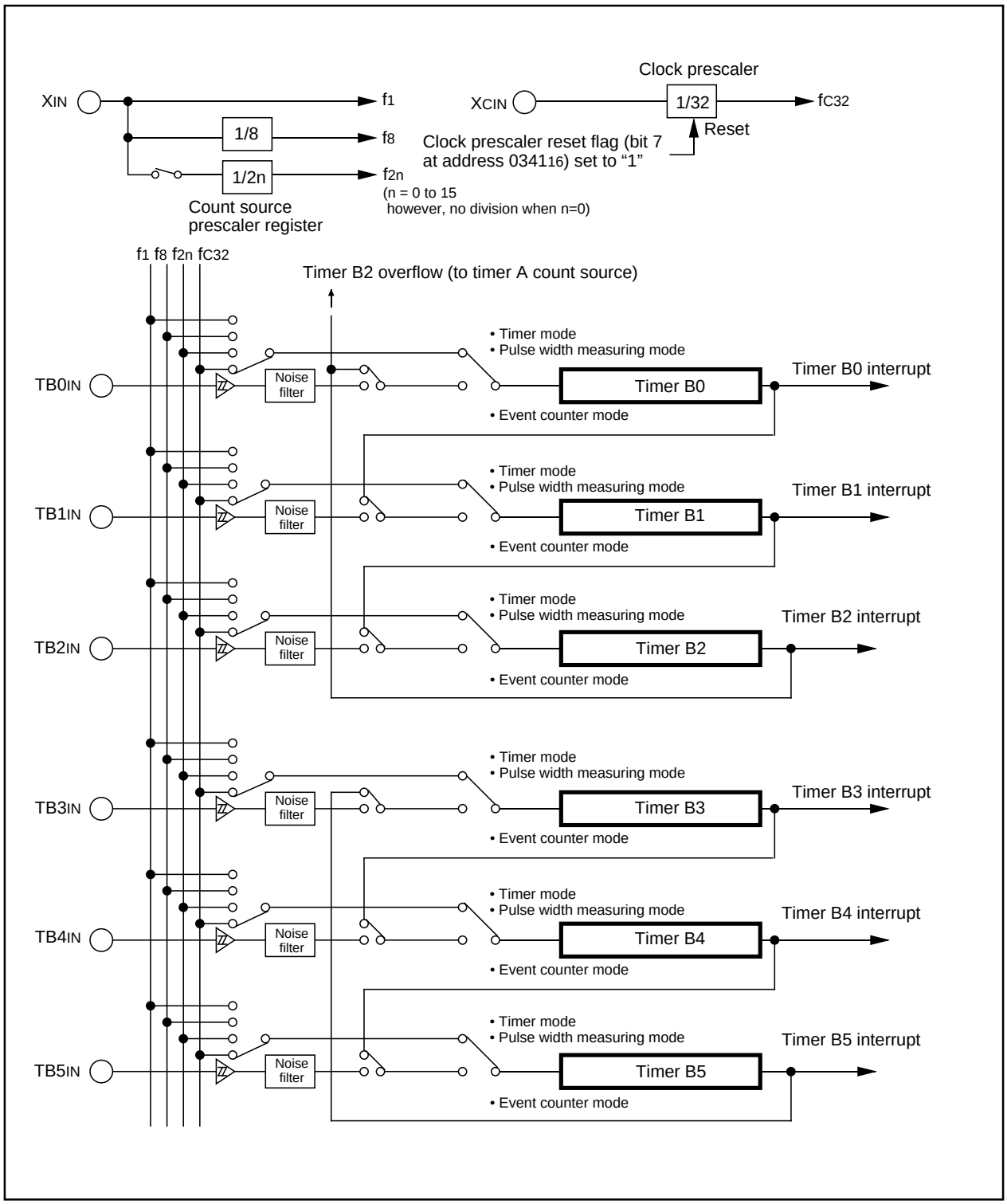



**Figure 1.13.2. Timer B block diagram**

## Timer A

Figure 1.14.1 shows the block diagram of timer A. Figures 1.14.2 to 1.14.6 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 outputs one effective pulse until the count reaches "$0000_{16}$".
- Pulse width modulation (PWM) mode: The timer outputs pulses of a given width.

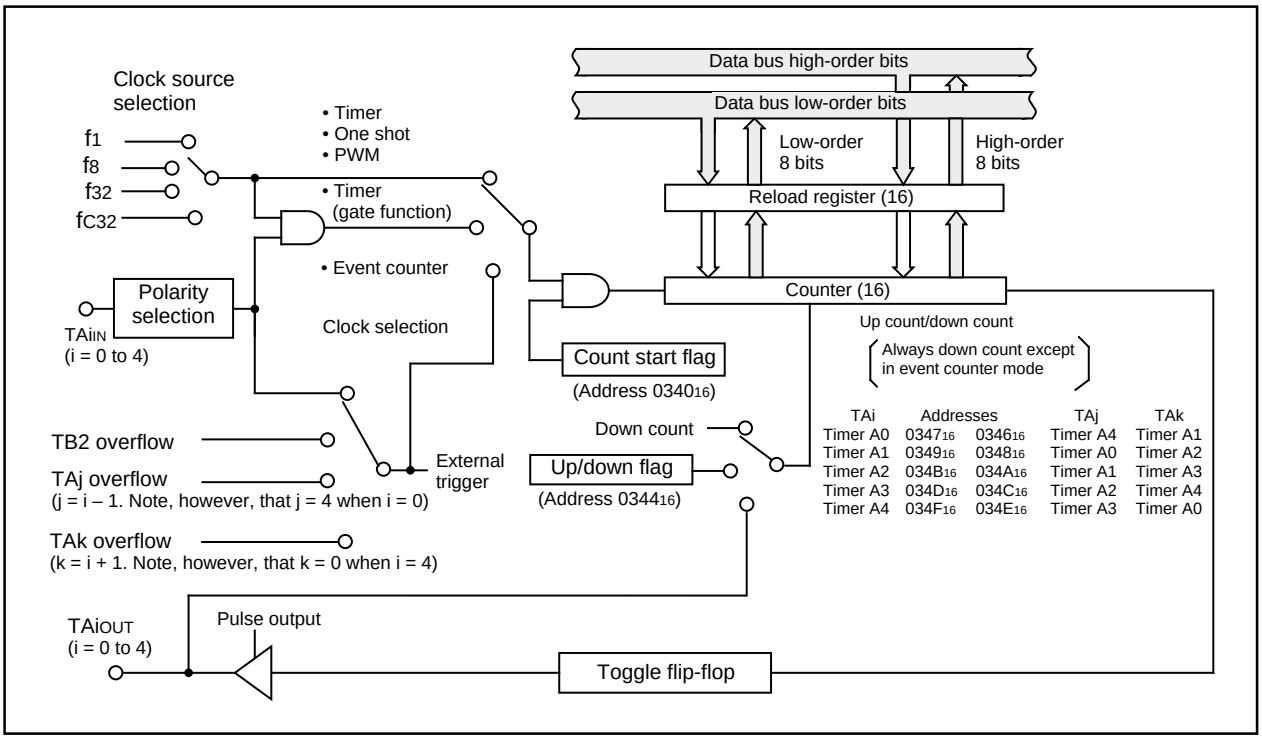
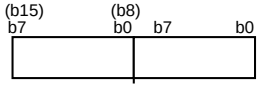



**Figure 1.14.1. Block diagram of timer A**

Timer Ai register (i = 0 to 4) (Note 1)




| Symbol | Address | When reset |
|---|---|---|
| TAi (i = 0 to 2) | 0347$_{16}$,0346$_{16}$, 0349$_{16}$,0348$_{16}$, 034B$_{16}$,034A$_{16}$ | Indeterminate |
| TAi (i = 3, 4) | 034D$_{16}$,034C$_{16}$, 034F$_{16}$,034E$_{16}$ | Indeterminate |

| | Function | Values that can be set | R | W |
|---|---|---|---|---|
| Timer mode | 16-bit counter (set to dividing ratio) | 0000$_{16}$ to FFFF$_{16}$ | ○ | ○ |
| Event counter mode | 16-bit counter (set to dividing ratio) (Note 2) | 0000$_{16}$ to FFFF$_{16}$ | ○ | ○ |
| One-shot timer mode | 16-bit counter (set to one shot width) (Note 6) | 0000$_{16}$ to FFFF$_{16}$ (Note 3) | – | ○ |
| Pulse width modulation mode (16-bit PWM) | 16-bit pulse width modulator (set to PWM pulse “H” width) (Note 4, 7) | 0000$_{16}$ to FFFE$_{16}$ (Note 3) | – | ○ |
| Pulse width modulation mode (8-bit PWM) | Low-order 8 bits : 8-bit prescaler (Note 5, 7) (set to PWM period) High-order 8 bits : 8-bit pulse width modulator (set to PWM pulse “H” width) | 00$_{16}$ to FE$_{16}$ (High-order address) 00$_{16}$ to FF$_{16}$ (Low-order address) (Note 3) | – | ○ |

Note 1: Read and write data in 16-bit units.
Note 2: Counts pulses from an external source or timer overflow.
Note 3: Use MOV instruction to write to this register.
Note 4: When setting value is n, PWM period and “H” width of PWM pulse are as follows:
PWM period : ($2^{16}$ - 1) / fi
PWM pulse “H” width : n / fi
Note 5: When setting value of high-order address is n and setting value of low-order address is m, PWM period and “H” width of PWM pulse are as follows:
PWM period : ($2^8$ - 1) X (m + 1) / fi
PWM pulse “H” width : (m + 1)n / fi
Note 6: When the timer Ai register is set to "0000$_{16}$", the counter does not operate and the timer Ai interrupt request is not generated. When the pulse is set to output, the pulse does not output from the TAi$_{OUT}$ pin.
Note 7: When the timer Ai register is set to "0000$_{16}$", the pulse width modulator does not operate and the output level of the TAi$_{OUT}$ pin remains "L" level, therefore the timer Ai interrupt request is not generated. This also occurs in the 8-bit pulse width modulator mode when the significant 8 high-order bits in the timer Ai register are set to "00$_{16}$".

**Figure 1.14.2. Timer A-related registers (1)**

Timer Ai mode register (i = 0 to 4)

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

| Symbol | Address | When reset |
|---|---|---|
| TAiMR(i=0 to 4) | $0356_{16}$, $0357_{16}$, $0358_{16}$, $0359_{16}$, $035A_{16}$ | $00000X00_2$ |

| 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 | This bit is invalid in M32C/80 series. Port output control is set by the function select registers A, B and C. | | – | – |
| MR1 | ——— | Function varies with each operation mode | ○ | ○ |
| MR2 | | | ○ | ○ |
| MR3 | | | ○ | ○ |
| TCK0 | Count source select bit | Function varies with each operation mode | ○ | ○ |
| TCK1 | | | ○ | ○ |

Count start flag

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

| Symbol | Address | When reset |
|---|---|---|
| TABSR | $0340_{16}$ | $00_{16}$ |

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

**Figure 1.14.3. Timer A-related registers (2)**

Up/down flag (Note 1)

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

Symbol: UDF
Address: $0344_{16}$
When reset: $00_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TA0UD | Timer A0 up/down flag | 0 : Down count<br>1 : Up count (Note 2) | O | O |
| TA1UD | Timer A1 up/down flag | 0 : Down count<br>1 : Up count (Note 2) | O | O |
| TA2UD | Timer A2 up/down flag | 0 : Down count<br>1 : Up count (Note 2) | O | O |
| TA3UD | Timer A3 up/down flag | 0 : Down count<br>1 : Up count (Note 2) | O | O |
| TA4UD | Timer A4 up/down flag | 0 : Down count<br>1 : Up count (Note 2) | 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 (Note 3) | — | O |
| TA3P | Timer A3 two-phase pulse signal processing select bit | 0 : two-phase pulse signal processing disabled<br>1 : two-phase pulse signal processing enabled (Note 3) | — | O |
| TA4P | Timer A4 two-phase pulse signal processing select bit | 0 : two-phase pulse signal processing disabled<br>1 : two-phase pulse signal processing enabled (Note 3) | — | O |

Note 1: Use MOV instruction to write to this register.
Note 2: This specification becomes valid when the up/down flag content is selected for up/down switching cause.
Note 3: When not using the two-phase pulse signal processing function, set the select bit to “0”.

One-shot start flag

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

Symbol: ONSF
Address: $0342_{16}$
When reset: $00_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TA0OS | Timer A0 one-shot start flag | 0 : Invalid<br>1 : Timer start (Note 1) | O | O |
| TA1OS | Timer A1 one-shot start flag | 0 : Invalid<br>1 : Timer start (Note 1) | O | O |
| TA2OS | Timer A2 one-shot start flag | 0 : Invalid<br>1 : Timer start (Note 1) | O | O |
| TA3OS | Timer A3 one-shot start flag | 0 : Invalid<br>1 : Timer start (Note 1) | O | O |
| TA4OS | Timer A4 one-shot start flag | 0 : Invalid<br>1 : Timer start (Note 1) | O | O |
| TAZIE | Z phase input enable bit | 0 : Invalid<br>1 : Valid | O | O |
| TA0TGL | Timer A0 event/trigger select bit | b7 b6<br>0 0 : Input on $TA_{0IN}$ is selected (Note 2)<br>0 1 : TB2 overflow is selected<br>1 0 : TA4 overflow is selected<br>1 1 : TA1 overflow is selected | O | O |
| TA0TGH | | | O | O |

Note 1: When read, the value is “0”.
Note 2: Set the corresponding pin output function select register to I/O port, and port direction register to “0”.

**Figure 1.14.4. Timer A-related registers (3)**

Trigger select register

b7 b6 b5 b4 b3 b2 b1 b0

| Symbol | Address | When reset |
|---|---|---|
| TRGSR | 0343₁₆ | 00₁₆ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TA1TGL | Timer A1 event/trigger select bit | b1 b0<br>0 0 : Input on TA1IN 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 TA2IN 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 TA3IN 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 TA4IN 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 function select register A to I/O port, and port direction register to “0”.

Clock prescaler reset flag

b7 b6 b5 b4 b3 b2 b1 b0

| Symbol | Address | When reset |
|---|---|---|
| CPSRF | 0341₁₆ | 0XXXXXXX₂ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| — | Nothing is assigned.<br>When write, set “0”. When read, their contents are indeterminate. | | — | — |
| — | | | — | — |
| — | | | — | — |
| — | | | — | — |
| — | | | — | — |
| — | | | — | — |
| — | | | — | — |
| CPSR | Clock prescaler reset flag | 0 : Ignored<br>1 : Prescaler is reset<br>(When read, the value is “0”) | ○ | ○ |

**Figure 1.14.5. Timer A-related registers (4)**

Count source prescaler register

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

| Symbol | Address | When reset |
|---|---|---|
| TCSPR | $035F_{16}$ | $0XXXXXXX_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CNT0 | Count value set bits | b3 b2 b1 b0<br>0 0 0 0 : No division<br>0 0 0 1 : Division by 2<br>0 0 1 0 : Division by 4<br>0 0 1 1 : Division by 6<br>⋮<br>1 1 0 1 : Division by 26<br>1 1 1 0 : Division by 28<br>1 1 1 1 : Division by 30<br>(Note 1) | ○ | ○ |
| CNT1 | | | ○ | ○ |
| CNT2 | | | ○ | ○ |
| CNT3 | | | ○ | ○ |
| — | Nothing is assigned.<br>When write, set “0”. When read, their contents are indeterminate. | | — | — |
| — | | | — | — |
| — | | | — | — |
| CST | Count start bit | 0 : Stops counting<br>1 : Starts counting (Note 2) | ○ | ○ |

Note 1: Set count start bit to “0” before writing to count value set bits.
Note 2: When this bit is set to “0”, divider circuit is inactive.

**Figure 1.14.6. Timer A-related registers (5)**

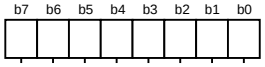
## (1) Timer mode

In this mode, the timer counts an internally generated count source. (See Table 1.14.1.) Figure 1.14.7 shows the timer Ai mode register in timer mode.

**Table 1.14.1. Specifications of timer mode**

| Item | Specification |
|---|---|
| Count source | $f_1$, $f_8$, $f_{2n}$, $f_{C32}$ |
| Count operation | • Down count<br>• When the timer underflows, it reloads the reload register contents before continuing counting |
| Divide ratio | 1/(m+1) m : 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 (Setting by corresponding function select registers A, B and C) |
| 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 (i = 0 to 4) (Timer mode)




Symbol: TAiMR(i=0 to 4) | Address: $0356_{16}$, $0357_{16}$, $0358_{16}$, $0359_{16}$, $035A_{16}$ | When reset: $00000X00_2$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>0 0 : Timer mode | ○ | ○ |
| TMOD1 | | | ○ | ○ |
| MR0 | This bit is invalid in M32C/80 series.<br>Port output control is set by the function select registers A, B and C. | | – | – |
| MR1 | Gate function select bit | b4 b3<br>0 X : Gate function not available (Note 1)<br>(TAiIN pin is a normal port pin)<br>1 0 : Timer counts only when TAiIN pin is held "L" (Note 2) | ○ | ○ |
| MR2 | | 1 1 : Timer counts only when TAiIN pin is held "H" (Note 2) | ○ | ○ |
| MR3 | 0 (Set 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 : $f_{2n}$ (Note 3)<br>1 1 : $f_{C32}$ | ○ | ○ |
| TCK1 | | | ○ | ○ |

Note 1: X value can be "0" or "1".
Note 2: Set the corresponding function select register A to I/O port, and port direction register to "0".
Note 3: n = 0 to 15. n is set by the count source prescaler register (address $035F_{16}$).

**Figure 1.14.7. Timer Ai mode register in timer mode**

## (2) Event counter mode

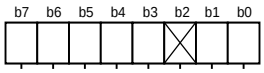
In this mode, the timer counts an external signal or an internal timer's overflow. Timers A0 and A1 can count a single-phase external signal. Timers A2, A3, and A4 can count a single-phase and a two-phase external signal. Table 1.14.2 lists timer specifications when counting a single-phase external signal. Table 1.14.3 lists timer specifications when counting a two-phase external signal. Figure 1.14.8 shows the timer Ai mode register in event counter mode.

**Table 1.14.2. 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 overflows or underflows, TAj overflows or underflows |
| 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 (Setting by corresponding function select registers A, B and C) |
| 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 (i = 0 to 4) (Event counter mode)



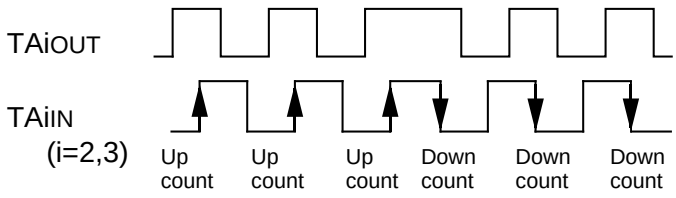
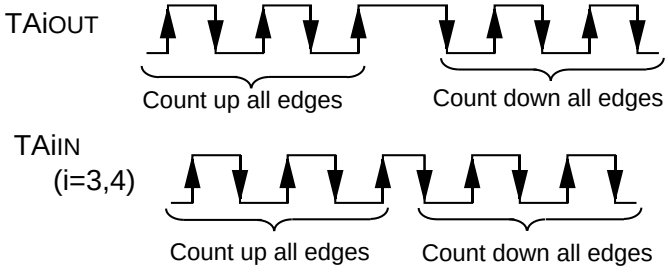

Symbol: TAiMR(i=0 to 4) | Address: $0356_{16}$, $0357_{16}$, $0358_{16}$, $0359_{16}$, $035A_{16}$ | When reset: $00000X00_2$

| Bit symbol | Bit name | Function (When not using two-phase pulse signal processing) | Function (When using two-phase pulse signal processing) | R | W |
|---|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>0 1 : Event counter mode (Note 1) | | O | O |
| TMOD1 | | | | O | O |
| MR0 | This bit is invalid in M32C/80 series.<br>Port output control is set by the function select registers A, B and C. | | | – | – |
| MR1 | Count polarity select bit (Note 2) | 0 : Counts external signal's falling edges<br>1 : Counts external signal's rising edges | 0 (Set to “0” when using two-phase pulse signal processing) | O | O |
| MR2 | Up/down switching cause select bit | 0 : Up/down flag's content<br>1 : TAiOUT pin's input signal (Note 3) | 1 (Set to “1” when using two-phase pulse signal processing) | O | O |
| MR3 | 0 (Set to “0” in Event counter mode) | | | O | O |
| TCK0 | Count operation type select bit | 0 : Reload type<br>1 : Free-run type | | O | O |
| TCK1 | Two-phase pulse signal processing operation select bit (Note 4,Note 5) | 0 (Set to “0” when not using two-phase pulse signal processing) | 0 : Normal processing operation<br>1 : Multiply-by-4 processing operation | O | O |

Note 1: Count source is select by the event/trigger select bit (addresses $0342_{16}$, $0343_{16}$) in event counter mode.
Note 2: This bit is valid when only counting an external signal.
Note 3: Set the corresponding function select register A to I/O port, and port direction register to “0”.
Signal of $TAi_{OUT}$ pin counts down at the time of “L” and counts up at the time of “H”.
Note 4: This bit is valid for timer A3 mode register.
Timer A0 and A1 can be “0” or “1”.
Timer A2 is fixed to normal processing operation and timer A4 is fixed to multiply-by-4 processing operation.
Note 5: When performing two-phase pulse signal processing, make sure the two-phase pulse signal processing operation select bit (address $0344_{16}$) is set to “1”. Also, always be sure to set the event/trigger select bit (address $0343_{16}$) to “00”.

**Figure 1.14.8. Timer Ai mode register in event counter mode**

**Table 1.14.3. Timer specifications in event counter mode**

| 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 1) |
| Divide ratio | • 1/ ($FFFF_{16}$ - n + 1) for up count<br>• 1/ (n + 1) for down count  n : Set value |
| Count start condition | Count start flag is set (= 1) |
| Count stop condition | Count start flag is reset (= 0) |
| Interrupt request generation timing | Timer overflows or underflows |
| TAiIN pin function | Two-phase pulse input |
| TAiOUT pin function | Two-phase pulse input (Set corresponding function select register A for I/O port) |
| 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 (Note 2) | • Normal processing operation (TimerA2 and timer A3)<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 (TimerA3 and timer A4)<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>  |

(when processing two-phase pulse signal with timers A2, A3, and A4)

Note 1: This does not apply when the free-run function is selected.

Note 2: Timer A3 is selectable. Timer A2 is fixed to normal processing operation and timer A4 is fixed to multiply-by-4 operation.

**• Counter Resetting by Two-Phase Pulse Signal Processing**

This function resets the timer counter to “0” when the Z-phase (counter reset) is input during two-phase pulse signal processing.

This function can only be used in timer A3 event counter mode, two-phase pulse signal processing, free-run type, and multiply-by-4 processing. The Z phase is input to the INT2 pin.

When the Z-phase input enable bit (bit 5 at address $0342_{16}$) is set to “1”, the counter can be reset by Z-phase input. For the counter to be reset to “0” by Z-phase input, you must first write “$0000_{16}$” to the timer A3 register (addresses $034D_{16}$ and $034C_{16}$).

The Z-phase is input when the INT2 input edge is detected. The edge polarity is selected by the INT2 polarity switch bit (bit 4 at address $009C_{16}$). The Z-phase must have a pulse width greater than 1 cycle of the timer A3 count source. Figure 1.14.9 shows the relationship between the two-phase pulse (A phase and B phase) and the Z phase.

The counter is reset at the count source following Z-phase input. Figure 1.14.10 shows the timing at which the counter is reset to “0”.

Note that timer A3 interrupt requests occur successively two times when timer A3 underflow and INT2 input reload occures at the same time.

Do not use timer A3 interrupt request when this function is used.

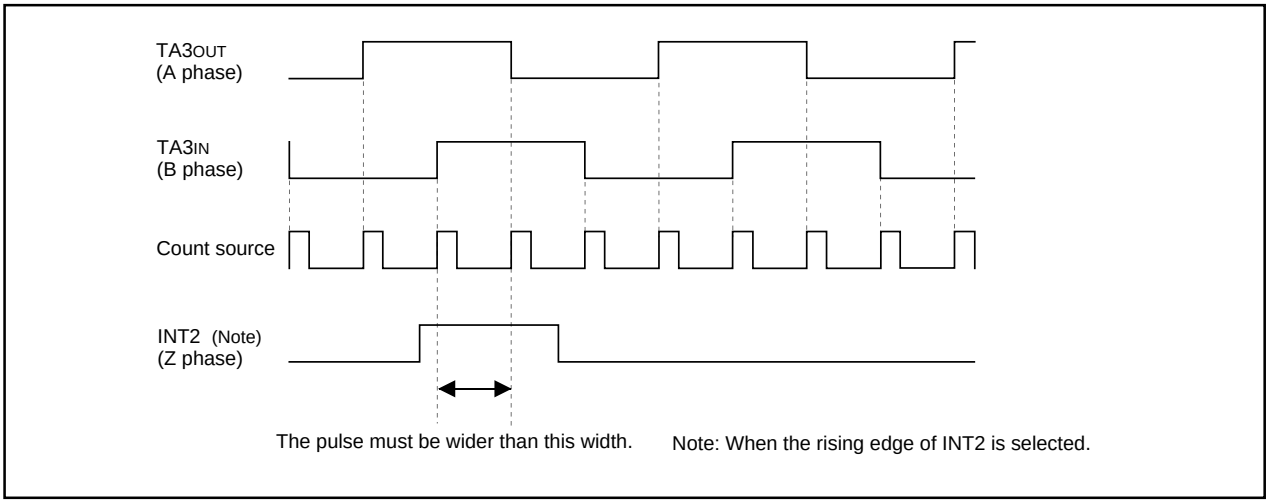



**Figure 1.14.9. The relationship between the two-phase pulse (A phase and B phase) and the Z phase**

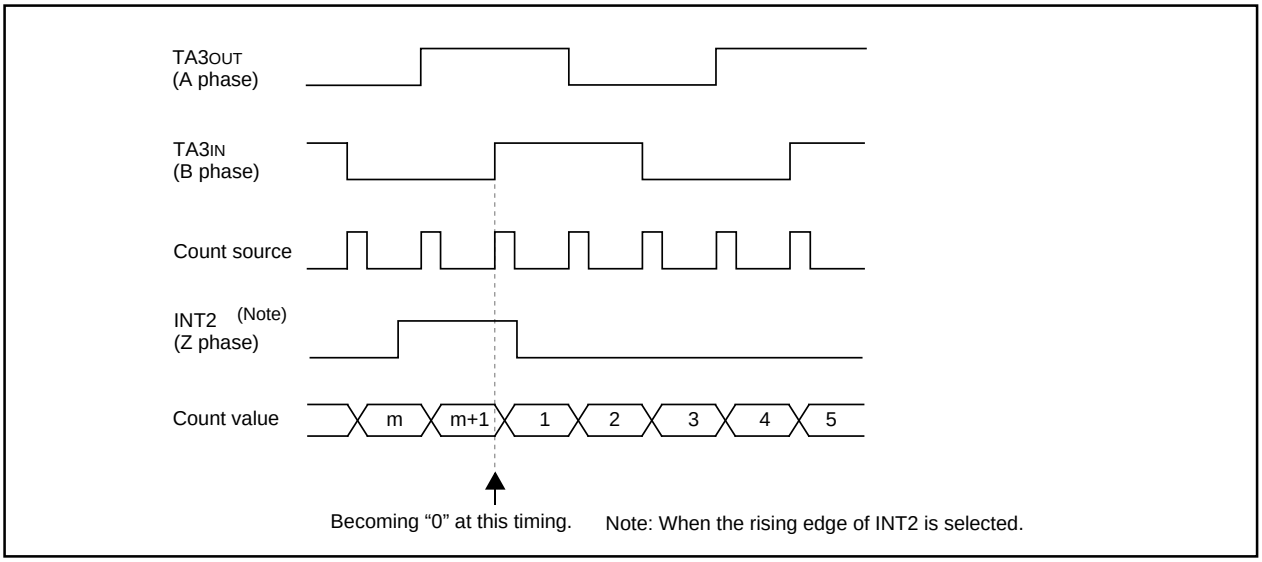



**Figure 1.14.10. The counter reset timing**

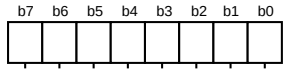
### (3) One-shot timer mode

In this mode, the timer operates only once. (See Table 1.14.4.) When a trigger occurs, the timer starts up and continues operating for a given period. Figure 1.14.11 shows the timer Ai mode register in one-shot timer mode.

**Table 1.14.4. Timer specifications in one-shot timer mode**

| Item | Specification |
|---|---|
| Count source | $f_1$, $f_8$, $f_{2n}$, $f_{C32}$ |
| Count operation | • The timer counts down<br>• When the count reaches $0000_{16}$, the timer stops counting after reloading a new count<br>• If a trigger occurs when counting, the timer reloads a new count and restarts counting |
| Divide ratio | 1/n n : Set value |
| Count start condition | • An external trigger is input<br>• The timer overflows<br>• The one-shot start flag is set (= 1) |
| Count stop condition | • A new count is reloaded after the count has reached $0000_{16}$<br>• The count start flag is reset (= 0) |
| Interrupt request generation timing | The count reaches $0000_{16}$ |
| $TAi_{IN}$ pin function | Programmable I/O port or trigger input |
| $TAi_{OUT}$ pin function | Programmable I/O port or pulse output (Setting by corresponding function select registers A, B and C) |
| 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 (i = 0 to 4) (One-shot timer mode)




| Symbol | Address | When reset |
|---|---|---|
| TAiMR(i=0 to 4) | $0356_{16}$, $0357_{16}$, $0358_{16}$, $0359_{16}$, $035A_{16}$ | $00000X00_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>1 0 : One-shot timer mode | ○ | ○ |
| TMOD1 | | | ○ | ○ |
| MR0 | This bit is invalid in M32C/80 series.<br>Port output control is set by the function select registers A, B and C. | | – | – |
| MR1 | External trigger select bit (Note 1) | 0 : Falling edge of $TA_{iIN}$ pin's input signal (Note 2)<br>1 : Rising edge of $TA_{iIN}$ pin's input signal (Note 2) | ○ | ○ |
| MR2 | Trigger select bit | 0 : One-shot start flag is valid<br>1 : Selected by event/trigger select register | ○ | ○ |
| MR3 | 0 (Set to "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 : $f_{2n}$ (Note 3)<br>1 1 : $f_{C32}$ | ○ | ○ |
| TCK1 | | | ○ | ○ |

Note 1: Valid only when the $TA_{iIN}$ pin is selected by the event/trigger select bit (addresses $0342_{16}$ and $0343_{16}$). If timer overflow is selected, this bit can be "1" or "0".
Note 2: Set the corresponding function select register A to I/O port, and port direction register to "0".
Note 3: n = 0 to 15. n is set by the count source prescaler register (address $035F_{16}$).

**Figure 1.14.11. Timer Ai mode register in one-shot timer mode**

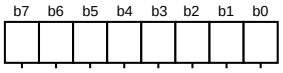
### (4) Pulse width modulation (PWM) mode

In this mode, the timer outputs pulses of a given width in succession. (See Table 1.14.5.) In this mode, the counter functions as either a 16-bit pulse width modulator or an 8-bit pulse width modulator. Figure 1.14.12 shows the timer Ai mode register in pulse width modulation mode. Figure 1.14.13 shows the example of how a 16-bit pulse width modulator operates. Figure 1.14.14 shows the example of how an 8-bit pulse width modulator operates.

**Table 1.14.5. Timer specifications in pulse width modulation mode**

| Item | Specification |
|---|---|
| Count source | $f_1$, $f_8$, $f_{2n}$, $f_{C32}$ |
| Count operation | • The timer counts down (operating as an 8-bit or a 16-bit pulse width modulator)<br>• The timer reloads a new count at a rising edge of PWM pulse and continues counting<br>• The timer is not affected by a trigger that occurs when counting |
| 16-bit PWM | • High level width n / fi n : Set value<br>• Cycle time $(2^{16}-1)$ / fi fixed |
| 8-bit PWM | • High level width n×(m+1) / fi n : values set to timer Ai register's high-order address<br>• Cycle time $(2^8-1)$ ×(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" |
| $TAi_{IN}$ pin function | Programmable I/O port or trigger input |
| $TAi_{OUT}$ pin function | Pulse output ($TAi_{OUT}$ is selected by corresponding function select registers A, B and C) |
| 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 (i = 0 to 4) (Pulse width modulator mode)




| Symbol | Address | When reset |
|---|---|---|
| TAiMR(i=0 to 4) | $0356_{16}$, $0357_{16}$, $0358_{16}$, $0359_{16}$, $035A_{16}$ | $00000X00_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>1 1 : Pulse width modulator (PWM) mode | ○ | ○ |
| TMOD1 | | | ○ | ○ |
| MR0 | This bit is invalid in M32C/80 series.<br>Port output control is set by the function select registers A, B and C. | | – | – |
| MR1 | External trigger select bit (Note 1) | 0: Falling edge of $TA_{iIN}$ pin's input signal (Note 2)<br>1: Rising edge of $TA_{iIN}$ 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 : $f_{2n}$ (Note 3)<br>1 1 : $f_{C32}$ | ○ | ○ |
| TCK1 | | | ○ | ○ |

Note 1: Valid only when the $TA_{iIN}$ pin is selected by the event/trigger select bit (addresses $0342_{16}$ and $0343_{16}$). If timer overflow is selected, this bit can be “1” or “0”.
Note 2: Set the corresponding function select register A to I/O port, and port direction register to “0”.
Note 3: n = 0 to 15. n is set by the count source prescaler register (address $035F_{16}$).

**Figure 1.14.12. Timer Ai mode register in pulse width modulation mode**

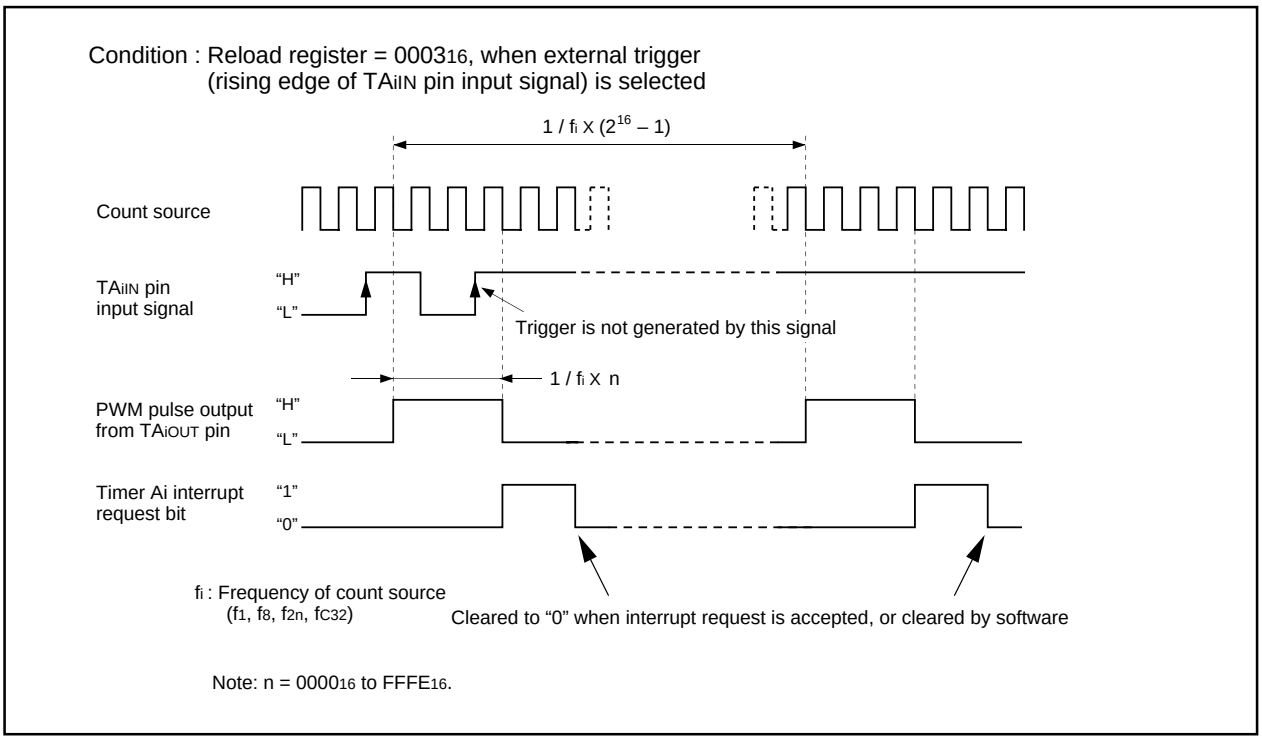



**Figure 1.14.13. Example of how a 16-bit pulse width modulator operates**

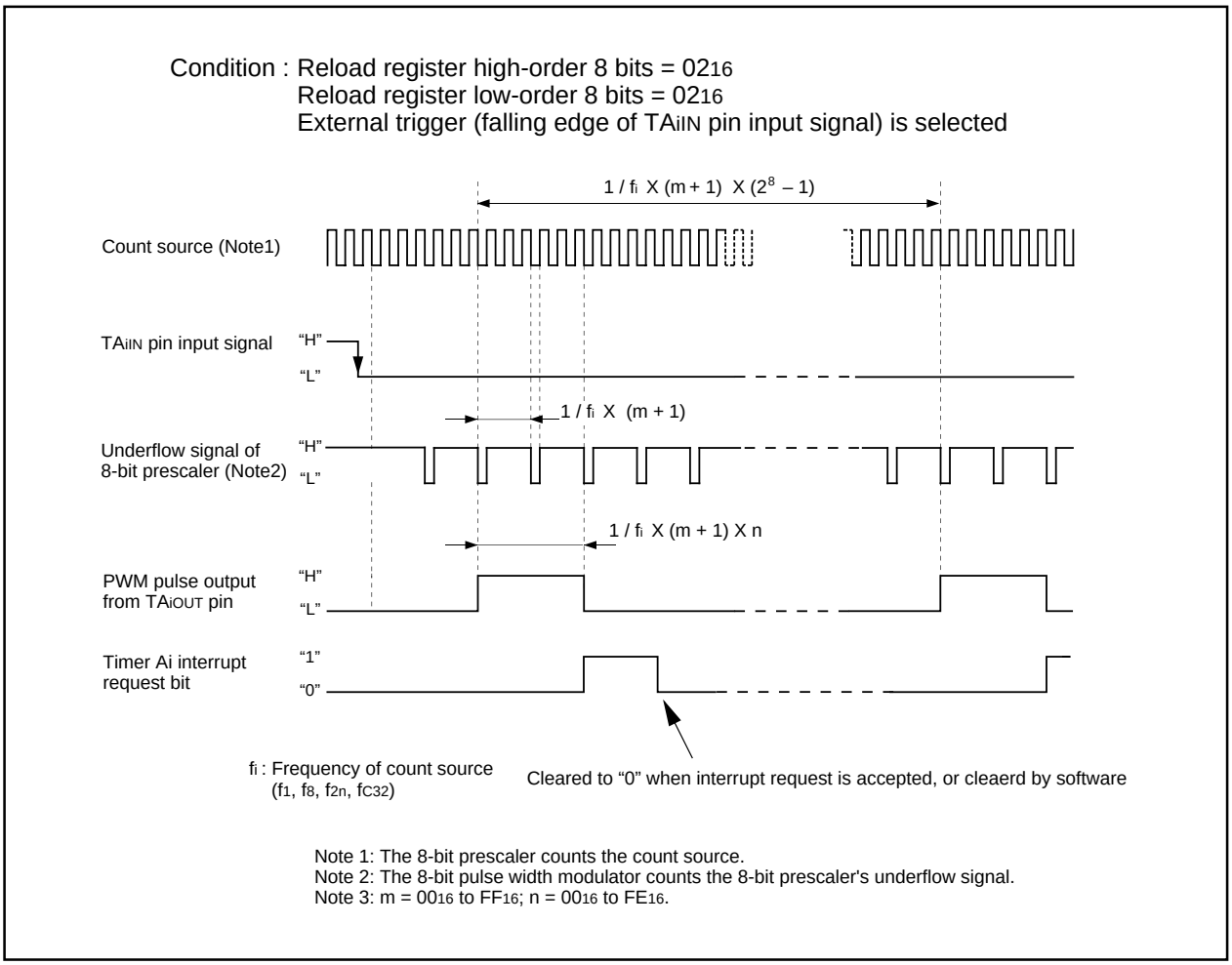



**Figure 1.14.14. Example of how an 8-bit pulse width modulator operates**

## Timer B

Figure 1.15.1 shows the block diagram of timer B. Figures 1.15.2 and 1.15.4 show the timer B-related registers. Use the timer Bi mode register (i = 0 to 5) bits 0 and 1 to choose the desired mode.
Timer B has three operation modes listed as follows:

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

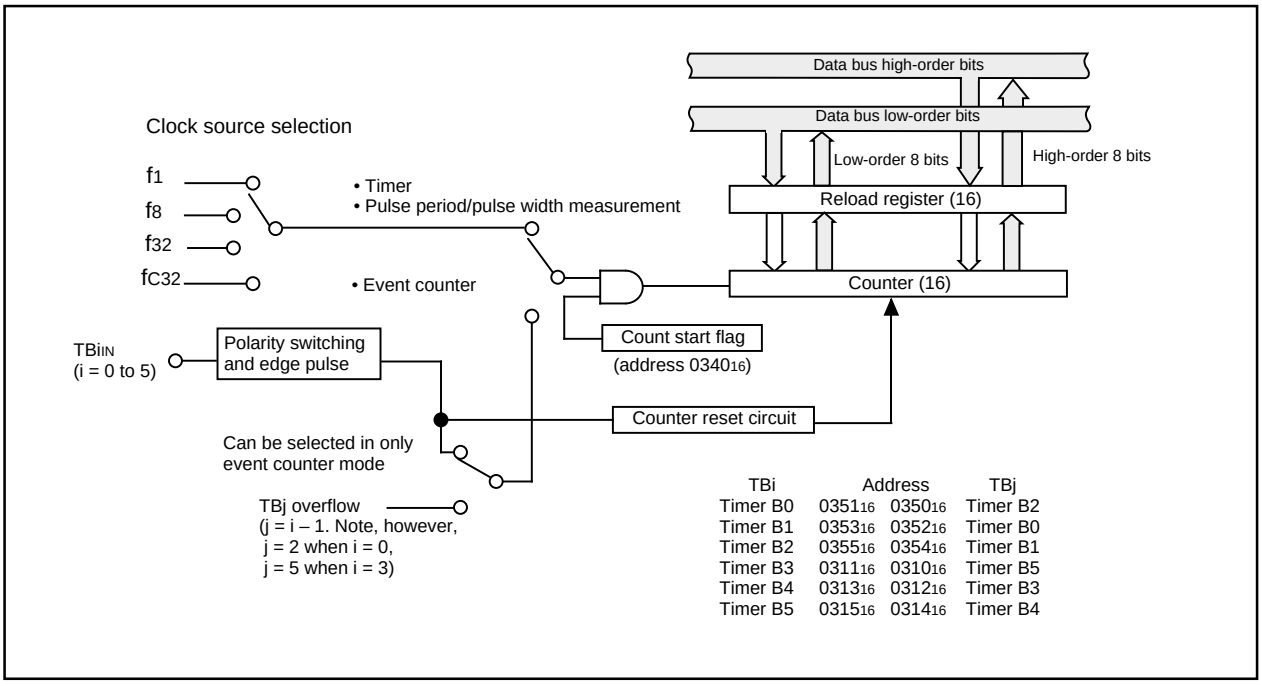
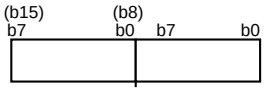



**Figure 1.15.1. Block diagram of timer B**

Timer Bi register (i = 0 to 6) (Note 1)




| Symbol | Address | When reset |
|---|---|---|
| TBi (i = 0 to 2) | 035116,035016, 035316,035216, 035516,035416 | Indeterminate |
| TBi (i = 3 to 5) | 031116,031016, 031316,031216, 031516,031416 | Indeterminate |

| | Function | Values that can be set | R | W |
|---|---|---|---|---|
| Timer mode | 16-bit counter (set to dividing ratio) | 000016 to FFFF16 | ○ | ○ |
| Event counter mode | 16-bit counter (set to dividing ratio) (Note 2) | 000016 to FFFF16 | ○ | ○ |
| Pulse period / pulse width measurement mode | Measures a pulse period or width | ——— | ○ | – |

Note 1: Read and write data in 16-bit units.
Note 2: Counts external pulses input or a timer overflow.

**Figure 1.15.2. Timer B-related registers (1)**

Timer Bi mode register (i = 0 to 5)

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

| Symbol | Address | When reset |
|---|---|---|
| TBiMR(i=0 to 5) | 035B16, 035C16, 035D16, 031B16, 031C16, 031DB16 | 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 : Don't set it up | ○ | ○ |
| TMOD1 | | | ○ | ○ |
| MR0 | ——— | Function varies with each operation mode (Note 1) (Note 2) | ○ | ○ |
| MR1 | | | ○ | ○ |
| MR2 | | | ○ | ○ |
| MR3 | | | ○ | ○ |
| TCK0 | Count source select bit | Function varies with each operation mode | ○ | ○ |
| TCK1 | | | ○ | ○ |

Note 1: Bit 4 is valid only by timer B0 and timer B3.
Note 2: In timer B1, timer B2, timer B4 and timer B5, nothing is assigned by bit 4(There is not R/W).
When write, set “0”. When read, its content is indeterminate.

Count start flag

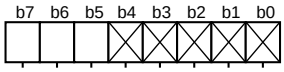
| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

| Symbol | Address | When reset |
|---|---|---|
| TABSR | 034016 | 0016 |

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

**Figure 1.15.3. Timer B-related registers (2)**

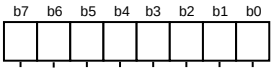
### Timer B3, B4,B5 count start flag




| Symbol | Address | When reset |
|---|---|---|
| TBSR | $0300_{16}$ | $000XXXXX_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| — | | | — | — |
| — | | | — | — |
| — | Nothing is assigned.<br>When write, set "0". When read, its content is indeterminate. | | — | — |
| — | | | — | — |
| — | | | — | — |
| TB3S | Timer B3 count start flag | 0 : Stops counting<br>1 : Starts counting | ○ | ○ |
| TB4S | Timer B4 count start flag | 0 : Stops counting<br>1 : Starts counting | ○ | ○ |
| TB5S | Timer B5 count start flag | 0 : Stops counting<br>1 : Starts counting | ○ | ○ |

### One-shot start flag




| Symbol | Address | When reset |
|---|---|---|
| ONSF | $0342_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TA0OS | Timer A0 one-shot start flag | 0 : Invalid<br>1 : Timer start (Note 1) | ○ | ○ |
| TA1OS | Timer A1 one-shot start flag | 0 : Invalid<br>1 : Timer start (Note 1) | ○ | ○ |
| TA2OS | Timer A2 one-shot start flag | 0 : Invalid<br>1 : Timer start (Note 1) | ○ | ○ |
| TA3OS | Timer A3 one-shot start flag | 0 : Invalid<br>1 : Timer start (Note 1) | ○ | ○ |
| TA4OS | Timer A4 one-shot start flag | 0 : Invalid<br>1 : Timer start (Note 1) | ○ | ○ |
| TAZIE | Z phase input enable bit | 0 : Invalid<br>1 : Valid | ○ | ○ |
| TA0TGL | Timer A0 event/trigger select bit | b7 b6<br>0 0 : Input on $TA_{0IN}$ is selected (Note 2)<br>0 1 : TB2 overflow is selected<br>1 0 : TA4 overflow is selected<br>1 1 : TA1 overflow is selected | ○ | ○ |
| TA0TGH | | | ○ | ○ |

Note 1: When read, the value is “0”.
Note 2: Set the corresponding pin output function select register to I/O port, and port direction register to “0”.

**Figure 1.15.4. Timer B-related registers (3)**

## (1) Timer mode

In this mode, the timer counts an internally generated count source. (See Table 1.15.1.) Figure 1.15.5 shows the timer Bi mode register in timer mode.

**Table 1.15.1. Timer specifications in timer mode**

| Item | Specification |
|---|---|
| Count source | $f_1$, $f_8$, $f_{2n}$, $f_{C32}$ |
| Count operation | • Counts down<br>• When the timer underflows, it reloads the reload register contents before continuing counting |
| Divide ratio | 1/(m+1) m : Set value |
| Count start condition | Count start flag is set (= 1) |
| Count stop condition | Count start flag is reset (= 0) |
| Interrupt request generation timing | The timer underflows |
| TBiIN pin function | Programmable I/O port |
| Read from timer | Count value is read out by reading timer Bi register |
| Write to timer | • 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 (i = 0 to 5) (Timer mode)

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

Symbol: TBiMR(i=0 to 5) — Address: 035B16, 035C16, 035D16, 031B16, 031C16, 031D16 — When reset: 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. Can be "0" or "1". | | ○ | ○ |
| MR1 | | | ○ | ○ |
| MR2 | 0 (Set to "0" in timer mode) (Note 1) | | ○ | ○ |
| | Nothing is assigned. (i = 1, 2, 4, 5) (Note 2) When write, set "0". When read, its content is indeterminate. | | — | — |
| MR3 | Invalid in timer mode. When write, set "0". When read in timer mode, its content is indeterminate. | | — | — |
| TCK0 | Count source select bit | b7 b6<br>0 0 : $f_1$<br>0 1 : $f_8$<br>1 0 : $f_{2n}$ (Note 3)<br>1 1 : $f_{C32}$ | ○ | ○ |
| TCK1 | | | ○ | ○ |

Note 1: R/W is valid only in timer B0 and timer B3.
Note 2: In timer B1, timer B2, timer B4 and timer B5, nothing is assigned by bit 4(There is not R/W). When write, set "0". When read, its content is indeterminate.
Note 3: n = 0 to 15. n is set by the count source prescaler register (address 035F16).

**Figure 1.15.5. Timer Bi mode register in timer mode**

## (2) Event counter mode

In this mode, the timer counts an external signal or an internal timer's overflow. (See Table 1.15.2.) Figure 1.15.6 shows the timer Bi mode register in event counter mode.

Table **1.15.2**. 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<br>• TBj overflows or underflows |
| 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 (Set the corresponding function select register A to I/O port.) |
| 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 (i = 0 to 5) (Event counter mode)

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

Symbol: TBiMR(i=0 to 5) — Address: 035B16, 035C16, 035D16, 031B16, 031C16, 031D16 — When reset: 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 | b3 b2<br>0 0 : Counts external signal's falling edges<br>0 1 : Counts external signal's rising edges | ○ | ○ |
| MR1 | | 1 0 : Counts external signal's falling and rising edges<br>1 1 : Inhibited (Note 1) | ○ | ○ |
| MR2 | 0 (Set to "0" in event counter mode) (Note 2) | | ○ | ○ |
| | Nothing is assigned. (i = 1, 2, 4, 5) (Note 3)<br>When write, set "0". When read, its content is indeterminate. | | — | — |
| MR3 | Invalid in event counter mode. When write, set "0". When read in event counter mode, its content is indeterminate. | | — | — |
| TCK0 | Invalid in event counter mode.<br>Can be "0" or "1". | | ○ | ○ |
| TCK1 | Event clock select bit | 0 : Input from TBiIN pin (Note 4)<br>1 : TBj overflow (Note 5) | ○ | ○ |

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: R/W is valid only in timer B0 and timer B3.
Note 3: In timer B1, timer B2, timer B4 and timer B5, nothing is assigned by bit 4(There is not R/W).
When write, set "0". When read, its content is indeterminate.
Note 4: Set the corresponding function select register A to I/O port, and port direction register to "0".
Note 5: j = i – 1; however, j = 2 when i = 0, j = 5 when i = 3.

**Figure 1.15.6. Timer Bi mode register in event counter mode**

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

In this mode, the timer measures the pulse period or pulse width of an external signal. (See Table 1.15.3.) Figure 1.15.7 shows the timer Bi mode register in pulse period/pulse width measurement mode. Figure 1.15.8 shows the operation timing when measuring a pulse period. Figure 1.15.9 shows the operation timing when measuring a pulse width.

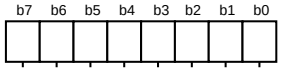
**Table 1.15.3. Timer specifications in pulse period/pulse width measurement mode**

| Item | Specification |
|---|---|
| Count source | $f_1$, $f_8$, $f_{2n}$, $f_{C32}$ |
| Count operation | • Count up<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". Assume that the count start flag condition is "1" and then the Timer Bi overflow flag becomes "1". If the Timer Bi mode register has a write-access after next count cycle of the timer from the above condition, the Timer Bi overflow flag becomes "0".) |
| TBiIN pin function | Measurement pulse input (Set the corresponding function select register A to I/O port.) |
| 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 (i = 0 to 5) (Pulse period / pulse width measurement mode)




| Symbol | Address | When reset |
|---|---|---|
| TBiMR(i=0 to 5) | $035B_{16}$, $035C_{16}$, $035D_{16}$, $031B_{16}$, $031C_{16}$, $031D_{16}$ | $00XX0000_2$ |

| 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 1<br>0 1 : Pulse period measurement 2<br>1 0 : Pulse width measurement<br>1 1 : Must not be set (Note 1) | ○ | ○ |
| MR1 | | | ○ | ○ |
| MR2 | 0 (Set to “0” in pulse period/pulse width measurement mode) (Note 2) | | ○ | ○ |
| | Nothing is assigned (i = 1, 2, 4, 5). (Note 3)<br>When write, set "0". When read, its content is indeterminate. | | — | — |
| MR3 | Timer Bi overflow flag (Note 4) | 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 : $f_{2n}$ (Note 5)<br>1 1 : $f_{C32}$ | ○ | ○ |
| TCK1 | | | ○ | ○ |

Note 1: Do the next measurement, In the measurement mode select bit.
Pulse period measurement 1 (bit 3, bit 2=“0 0”) : Interval between measurement pulse's falling edge to falling edge.
Pulse period measurement 2 (bit 3, bit 2=“0 1”) : Interval between measurement pulse's rising edge to rising edge.
Pulse width measurement (bit 3, bit 2=“1 0”) : Interval between measurement pulse's falling edge to rising edge, and between rising edge to falling edge.
Note 2: R/W is valid only in timer B0 and timer B3.
Note 3: In timer B1, timer B2, timer B4 and timer B5, nothing is assigned by bit 4(There is not R/W).
When write, set “0”. When read, its content is indeterminate.
Note 4: It is indeterminate when reset.
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 5: n = 0 to 15. n is set by the count source prescaler register (address $035F_{16}$).

**Figure 1.15.7. Timer Bi mode register in pulse period/pulse width measurement mode**

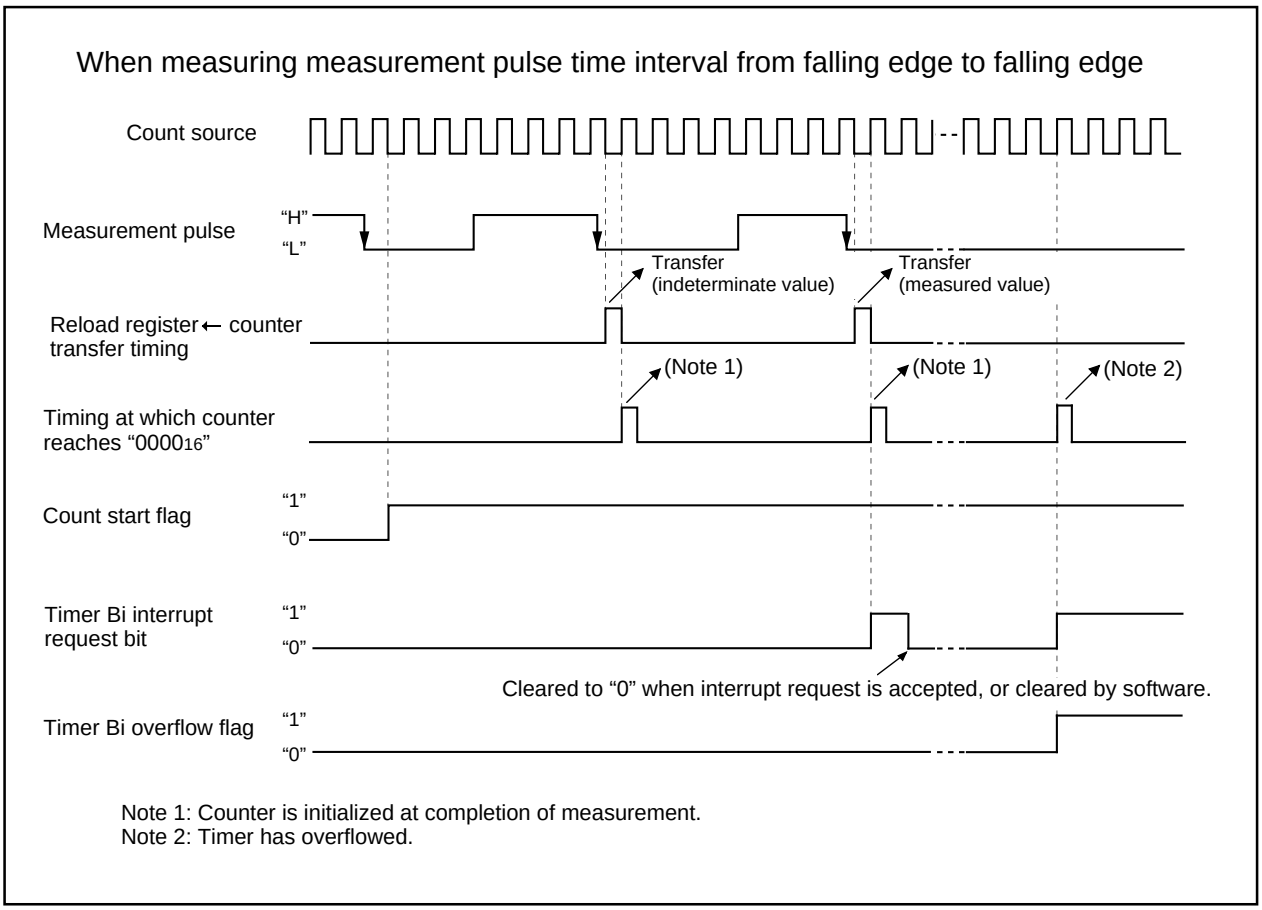



**Figure 1.15.8. Operation timing when measuring a pulse period**

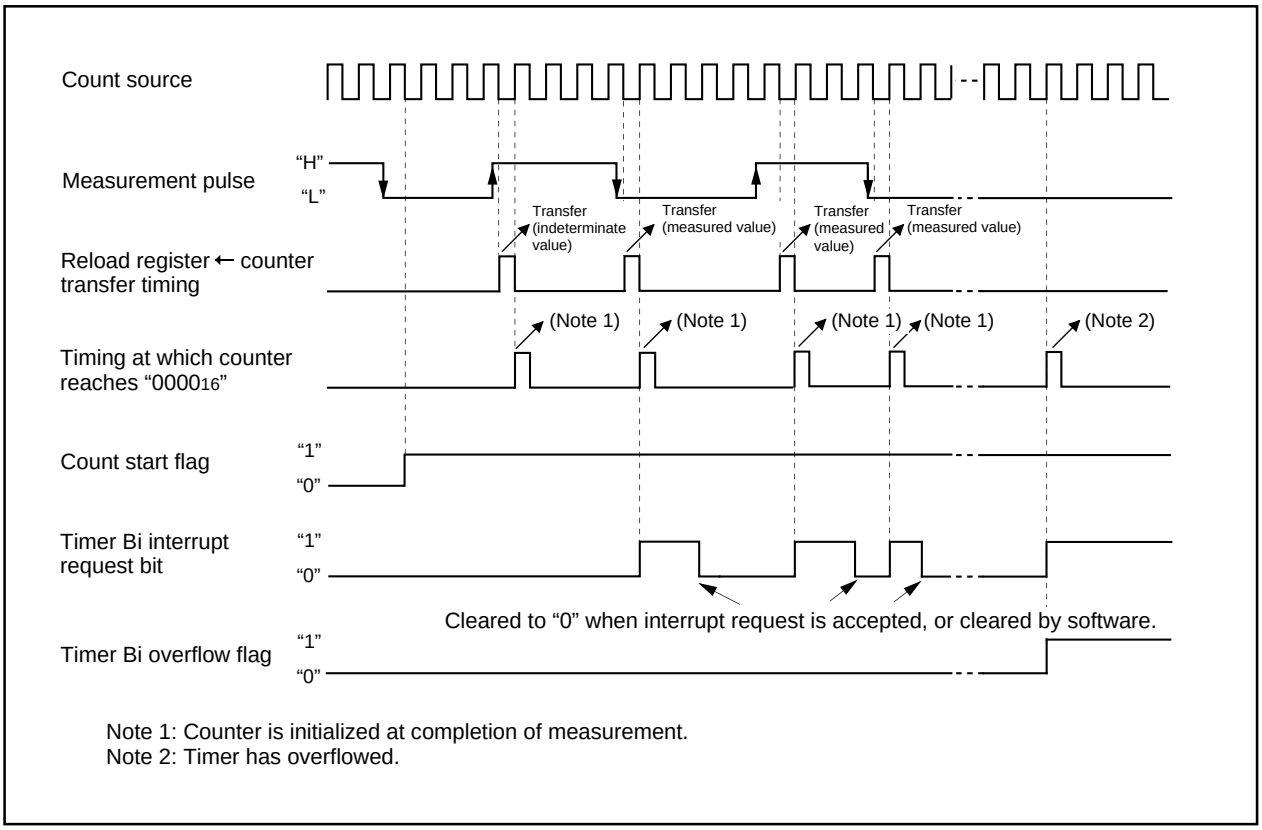



**Figure 1.15.9. Operation timing when measuring a pulse width**

## Three-phase motor control timers' functions

Use of more than one built-in timer A and timer B provides the means of outputting three-phase motor driving waveforms.
Figures 1.16.1 through 1.16.5 show registers related to timers for three-phase motor control.

Three-phase PWM control register 0 (Note 4)

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

| Symbol | Address | When reset |
|---|---|---|
| INVC0 | $0308_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Description | R | W |
|---|---|---|---|---|
| INV00 | Effective interrupt output polarity select bit | 0: A timer B2 interrupt occurs when the timer A1 reload control signal is "1".<br>1: A timer B2 interrupt occurs when the timer A1 reload control signal is "0". (Note 3) | ○ | ○ |
| INV01 | Effective interrupt output specification bit (Note 2) | 0: Not specified.<br>1: Selected by the INV00 bit. (Note 3) | ○ | ○ |
| INV02 | Mode select bit (Note 4) | 0: Normal mode<br>1: Three-phase PWM output mode | ○ | ○ |
| INV03 | Output control bit | 0: Output disabled<br>1: Output enabled | ○ | ○ |
| INV04 | Positive and negative phases concurrent L output disable function enable bit | 0: Feature disabled<br>1: Feature enabled | ○ | ○ |
| INV05 | Positive and negative phases concurrent L output detect flag | 0: Not detected yet<br>1: Already detected (Note 5) | ○ | ○ |
| INV06 | Modulation mode select bit | 0: Triangular wave modulation mode (Note 6)<br>1: Sawtooth wave modulation mode (Note 7) | ○ | ○ |
| INV07 | Software trigger bit | 0: Ignored<br>1: Trigger generated (Note 8) | ○ | ○ |

Note 1: Set bit 1 of the protect register (address $000A_{16}$) to "1" before writing to this register.
Note 2: Set bit 1 of this register to "1" after setting timer B2 interrupt occurrences frequency set counter.
Note 3: Effective only in three-phase mode 1(Three-phase PWM control register's bit 1 = "1").
Note 4: Selecting three-phase PWM output mode causes the dead time timer, the U, V, W phase output control circuits, and the timer B2 interrupt frequency set circuit works.
For U, $\overline{U}$, V, $\overline{V}$, W and $\overline{W}$ output from $P8_0$, $P8_1$, and $P7_2$ through $P7_5$, setting of function select registers A, B and C is required.
Note 5: No value other than "0" can be written.
Note 6: The dead time timer starts in synchronization with the falling edge of timer Ai output. The data transfer from the three-phase buffer register to the three-phase output shift register is made only once in synchronization with the transfer trigger signal after writing to the three-phase output buffer register.
Note 7: The dead time timer starts in synchronization with the falling edge of timer A output and with the transfer trigger signal. The data transfer from the three-phase output buffer register to the three-phase output shift register is made with respect to every transfer trigger.
Note 8: The value, when read, is "0".

**Figure 1.16.1. Registers related to timers for three-phase motor control**

Three-phase PWM control register 1 (Note 1)

b7 b6 b5 b4 b3 b2 b1 b0

| Symbol | Address | When reset |
|---|---|---|
| INVC1 | $0309_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Description | R | W |
|---|---|---|---|---|
| INV10 | Timer Ai start trigger signal select bit | 0: Timer B2 overflow signal<br>1: Timer B2 overflow signal, signal for writing to timer B2 | ○ | ○ |
| INV11 | Timer A1-1, A2-1, A4-1 control bit (Note 2) | 0: Three-phase mode 0<br>1: Three-phase mode 1 | ○ | ○ |
| INV12 | Dead time timer count source select bit | 0 : $f_1$<br>1 : $f_1/2$ | ○ | ○ |
| INV13 | Carrier wave detect flag | 0: Rising edge of triangular waveform<br>1: Falling edge of triangular waveform | ○ | — |
| INV14 | Output polarity control bit | 0 : Low active<br>1 : High active | ○ | ○ |
| INV15 | Dead time invalid bit | 0: Dead time valid bit<br>1: Dead time invalid bit | ○ | ○ |
| INV16 | Dead time timer trigger select bit | 0: Triggers from corresponding timer<br>1: Rising edge of corresponding phase output (Note 3) | ○ | ○ |
| INV17 | Waveform reflect timing select bit | 0: Synchronized with raising edge of triangular waveform<br>1: Synchronized with timer B2 overflow (Note 4) | ○ | ○ |

Note 1: Set bit 1 of the protect register (address $000A_{16}$) to "1" before writing to this register.
Note 2: INV13 is valid only in triangular waveform mode (INV06=0) and three-phase mode (INV11=1).
Note 3:Usually set to "1".
Note 4:INV17 is valid only in three-phase mode 1.

Three-phase output buffer register i (i=0, 1) (Note)

b7 b6 b5 b4 b3 b2 b1 b0

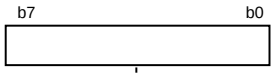
| Symbol | Address | When reset |
|---|---|---|
| IDBi (i=0,1) | $030A_{16}$, $030B_{16}$ | $XX00\ 0000_2$ |

| Bit Symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DUi | U phase output buffer i | Setting in U phase output buffer i | ○ | ○ |
| DUBi | $\overline{U}$ phase output buffer i | Setting in $\overline{U}$ phase output buffer i | ○ | ○ |
| DVi | V phase output buffer i | Setting in V phase output buffer i | ○ | ○ |
| DVBi | $\overline{V}$ phase output buffer i | Setting in $\overline{V}$ phase output buffer i | ○ | ○ |
| DWi | W phase output buffer i | Setting in W phase output buffer i | ○ | ○ |
| DWBi | $\overline{W}$ phase output buffer i | Setting in $\overline{W}$ phase output buffer i | ○ | ○ |
| — | Nothing is assigned.<br>When write, set "0". When read, their contents are "0". | | — | — |
| — | | | — | — |

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

**Figure 1.16.2. Registers related to timers for three-phase motor control**

### Dead time timer (Note)

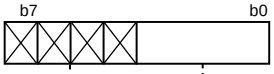



| Symbol | Address | When reset |
|---|---|---|
| DTT | 030C16 | Indeterminate |

| Function | Values that can be set | R | W |
|---|---|---|---|
| 8-bits counter (set dead time timer) | 1 to 255 | — | ○ |

Note: Use MOV instruction to write to this register.

### Timer B2 interrupt occurrences frequency set counter (Note 1 to 4)

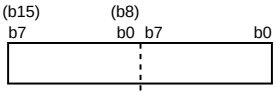



| Symbol | Address | When reset |
|---|---|---|
| ICTB2 | 030D16 | Indeterminate |

| Function | Values that can be set | R | W |
|---|---|---|---|
| Set occurrence frequency of timer B2 interrupt request | 1 to 15 | — | ○ |
| Nothing is assigned. When write, set to "0". | | — | — |

Note 1: Use MOV instruction to write to this register.
Note 2: When the effective interrupt output specification bit (INV01: bit 1 at 030816) is set to "1" and three-phase motor control timer is operating, do not rewrite to this register.
Note 3: Do not write to this register at the timing of timer B2 overflow.
Note 4: Setting of this register is valid only when bit 2 (INV02) of three-phase PWM control register 0 is set to "1".

### Timer Ai, Ai-1 register (Note 1 to 3)

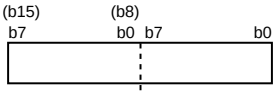



| Symbol | Address | When reset |
|---|---|---|
| TAi (i 1, 2, 4) | 034916,034816, 034B16,034A16, 034F16,034E16 | Indeterminate |
| TAi1 (i 1, 2, 4) | 030316,030216, 030516,030416, 030716,030616 | Indeterminate |

| Function | Values that can be set | R | W |
|---|---|---|---|
| Three-phase PWM pulse width modulator (decide PWM output pulse width) | 000016 to FFFF16 | — | ○ |

Note 1: Read and write data in 16-bit units.
Note 2: When set "000016" to the timer Ai register, counter doesn't move, and timer Ai interrupt isn't generated.
Note 3: Use MOV instruction to write to this register.

### Timer B2 register (Note)




| Symbol | Address | When reset |
|---|---|---|
| TB2 | 035516,035416 | Indeterminate |

| Function | Values that can be set | R | W |
|---|---|---|---|
| Set the period of carrier wave | 000016 to FFFF16 | ○ | ○ |

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

**Figure 1.16.3. Registers related to timers for three-phase motor control**

### Timer B2 special mode register

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

| Symbol | Address | When reset |
|---|---|---|
| TB2SC | $035E_{16}$ | $XXXXXXX0_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PWCOM | Timer B2 reload timing switching bit | 0 : Next underflow<br>1 : Synchronized rising edge of triangular wave | ○ | ○ |
| Nothing is assigned.<br>When write, set "0". When read, its content is "0". | | | — | — |

### Trigger select register

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

| Symbol | Address | When reset |
|---|---|---|
| TRGSR | $0343_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TA1TGL | Timer A1 event/trigger select bit | Set bit 1 and bit 0 to "0 1" before using to the V phase output control circuit. (Note) | ○ | ○ |
| TA1TGH | | | ○ | ○ |
| TA2TGL | Timer A2 event/trigger select bit | Set bit 3 and bit 2 to "0 1" before using to the W phase output control circuit. (Note) | ○ | ○ |
| TA2TGH | | | ○ | ○ |
| TA3TGL | Inhibited in Three-phase PWM mode. | | — | — |
| TA3TGH | | | — | — |
| TA4TGL | Timer A4 event/trigger select bit | Set bit 7 and bit 6 to "0 1" before using to the U phase output control circuit. (Note) | ○ | ○ |
| TA4TGH | | | ○ | ○ |

Note: Set the corresponding port function select register A to I/O port, and port direction register to "0".

### Count start flag

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

| Symbol | Address | When reset |
|---|---|---|
| TABSR | $0340_{16}$ | $00_{16}$ |

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

**Figure 1.16.4. Registers related to timers for three-phase motor control**

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

Setting "1" in the mode select bit (bit 2 at $0308_{16}$) shown in Figure 1.16.1 causes three-phase PWM output mode that uses four timers A1, A2, A4, and B2. As shown in Figure 1.16.4 and 1.16.5 set timers A1, A2, and A4 in one-shot timer mode, set the trigger in timer B2, and set timer B2 in timer mode using the respective timer mode registers.

Timer Ai mode register (i = 1, 2, 4)

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

Symbol: TAiMR(i=1, 2, 4) — Address: $0357_{16}$, $0358_{16}$, $035A_{16}$ — When reset: $00000X00_2$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>1 0 : One-shot timer mode | ○ | ○ |
| TMOD1 | | | ○ | ○ |
| MR0 | This bit is invalid in M32C/80 series.<br>Port output control is set by the function select registers A, B and C. | | — | — |
| MR1 | External trigger select bit | Invalid in Three-phase PWM output mode. | ○ | ○ |
| MR2 | Trigger select bit | 1 : Selected by event/trigger select register | ○ | ○ |
| MR3 | 0 (Set to "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 : $f_{2n}$ (Note)<br>1 1 : $f_{C32}$ | ○ | ○ |
| TCK1 | | | ○ | ○ |

Note: n = 0 to 15. n is set by the count source prescaler register (address $035F_{16}$).

Timer B2 mode register

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

Symbol: TB2MR — Address: $035D_{16}$ — When reset: $00XX0000_2$

| 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 (Set to "0" in timer mode) | | ○ | ○ |
| MR3 | Invalid in timer mode.<br>When write, set "0". When read in timer mode, its content is indeterminate. | | — | — |
| TCK0 | Count source select bit | b7 b6<br>0 0 : $f_1$<br>0 1 : $f_8$<br>1 0 : $f_{2n}$ (Note)<br>1 1 : $f_{C32}$ | ○ | ○ |
| TCK1 | | | ○ | ○ |

Note: n = 0 to 15. n is set by the count source prescaler register (address $035F_{16}$).

**Figure 1.16.5. Timer mode registers in three-phase PWM output mode**

Figure 1.16.6 shows the block diagram for three-phase waveform mode. The Low active output polarity in three-phase waveform mode, the positive-phase waveforms (U phase, V phase, and W phase) and negative waveforms ($\overline{U}$ phase, $\overline{V}$ phase, and $\overline{W}$ phase), six waveforms in total, are output from $P8_0$, $P8_1$, $P7_2$, $P7_3$, $P7_4$, and $P7_5$ as active on the "L" level. Of the timers used in this mode, timer A4 controls the U phase and $\overline{U}$ phase, timer A1 controls the V phase and $\overline{V}$ phase, and timer A2 controls the W phase and $\overline{W}$ phase respectively; timer B2 controls the periods of one-shot pulse output from timers A4, A1, and A2.

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

To set short circuit time, use three 8-bit timers, sharing the reload register, for setting dead time. A value from 1 through 255 can be set as the count of the timer for setting dead time. The timer for setting dead time works as a one-shot timer. If a value is written to the dead timer ($030C_{16}$), the value is written to the reload register shared by the three timers for setting dead time.

Any of the timers for setting dead time takes the value of the reload register into its counter, if a start trigger comes from its corresponding timer, and performs a down count in line with the clock source selected by the dead time timer count source select bit (bit 2 at $0309_{16}$). The timer can receive another trigger again before the workings due to the previous trigger are completed. In this instance, the timer performs a down count from the reload register's content after its transfer, provoked by the trigger, to the timer for setting dead time.

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

The positive waveforms (U phase, V phase, and W phase) and the negative waveforms ($\overline{U}$ phase, $\overline{V}$ phase, and $\overline{W}$ phase) in three-phase waveform mode are output, from respective ports by means of setting "1" in the output control bit (bit 3 at $0308_{16}$). Setting "0" in this bit causes the ports to be the high-impedance state. This bit can be set to "0" not only by use of the applicable instruction, but by entering a falling edge in the $\overline{NMI}$ terminal or by resetting. Also, if "1" is set in the positive and negative phases concurrent L output disable function enable bit (bit 4 at $0308_{16}$) causes one of the pairs of U phase and $\overline{U}$ phase, V phase and $\overline{V}$ phase, and W phase and $\overline{W}$ phase concurrently go to "L", as a result, the output control bit becomes the high-impedance state.

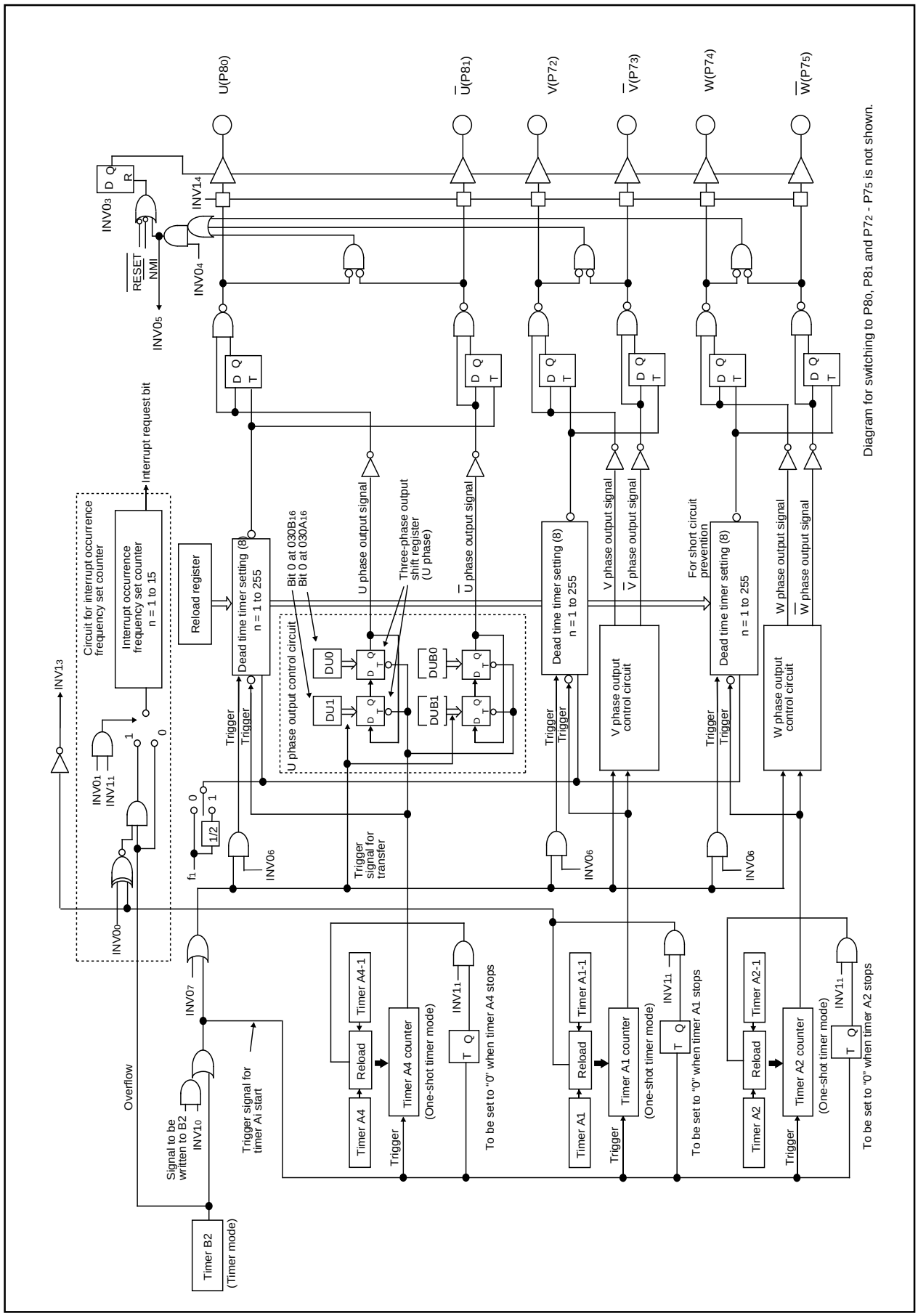



**Figure 1.16.6. Block diagram for three-phase waveform mode**

### Triangular wave modulation

To generate a PWM waveform of triangular wave modulation, set "0" in the modulation mode select bit (bit 6 at $0308_{16}$). Also, set "1" in the timers A4-1, A1-1, A2-1 control bit (bit 1 at $0309_{16}$). In this mode, each of timers A4, A1, and A2 has two timer registers, and alternately reloads the timer register's content to the counter every time timer B2 counter's content becomes $0000_{16}$. If "0" is set to the effective interrupt output specification bit (bit 1 at $0308_{16}$), the frequency of interrupt requests that occur every time the timer B2 counter's value becomes $0000_{16}$ can be set by use of the timer B2 counter ($030D_{16}$) for setting the frequency of interrupt occurrences. The frequency of occurrences is given by (setting; setting $\neq 0$).

Setting "1" in the effective interrupt output specification bit (bit 1 at $0308_{16}$) provides the means to choose which value of the timer A1 reload control signal to use, "0" or "1", to cause timer B2's interrupt request to occur. To make this selection, use the effective interrupt output polarity selection bit (bit 0 at $0308_{16}$).

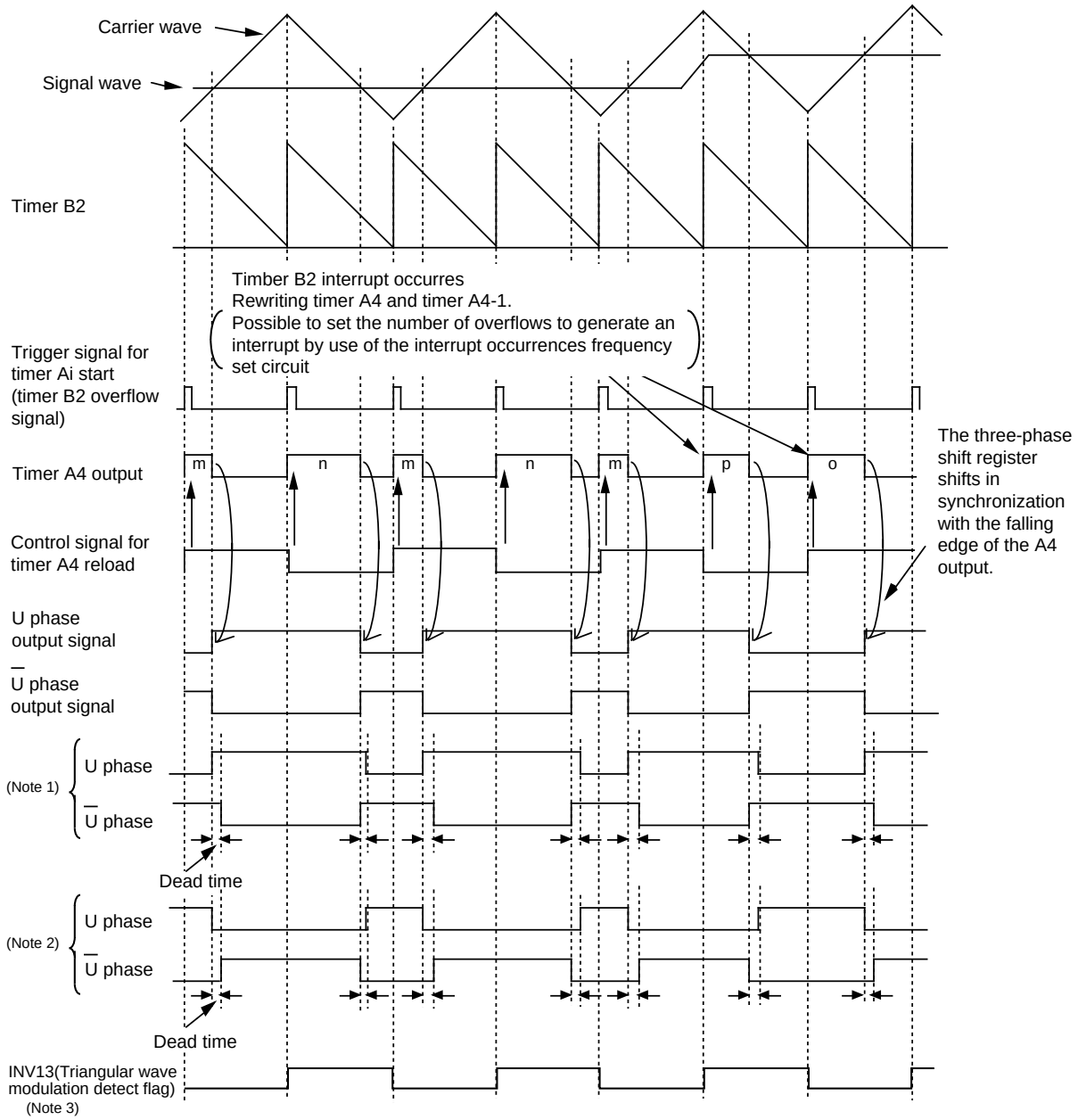
An example of U phase waveform is shown in Figure 1.16.7, and the description of waveform output workings is given below. Set "1" in DU0 (bit 0 at $030A_{16}$). And set "0" in DUB0 (bit 1 at $030A_{16}$). In addition, set "0" in DU1 (bit 0 at $030B_{16}$) and set "1" in DUB1 (bit 1 at $030B_{16}$). Also, set "0" in the effective interrupt output specification bit (bit 1 at $0308_{16}$) to set a value in the timer B2 interrupt occurrence frequency set counter. By this setting, a timer B2 interrupt occurs when the timer B2 counter's content becomes $0000_{16}$ as many as (setting) times. Furthermore, set "1" in the effective interrupt output specification bit (bit 1 at $0308_{16}$), set in the effective interrupt polarity select bit (bit 0 at $0308_{16}$) and set "1" in the interrupt occurrence frequency set counter ($030D_{16}$). These settings cause a timer B2 interrupt to occur every other interval when the U phase output goes to "H".

When the timer B2 counter's content becomes $0000_{16}$, timer A4 starts outputting one-shot pulses. In this instance, the content of DU1 (bit 0 at $030B_{16}$) and that of DU0 (bit 0 at $030A_{16}$) are set in the three-phase output shift register (U phase), the content of DUB1 (bit 1 at $030B_{16}$) and that of DUB0 (bit 1 at $030A_{16}$) are set in the three-phase shift register ($\overline{U}$ phase). After triangular wave modulation mode is selected, however, no setting is made in the shift register even though the timer B2 counter's content becomes $0000_{16}$.

The value of DU0 and that of DUB0 are output to the U terminal ($P8_0$) and to the $\overline{U}$ terminal ($P8_1$) respectively. When the timer A4 counter counts the value written to timer A4 ($034F_{16}$, $034E_{16}$) and when timer A4 finishes outputting one-shot pulses, the three-phase shift register's content is shifted one position, and the value of DU1 and that of DUB1 are output to the U phase output signal and to $\overline{U}$ phase output signal respectively. At this time, one-shot pulses are output from the timer for setting dead time used for setting the time over which the "L" level of the U phase waveform doesn't overlap the Low level of the $\overline{U}$ phase waveform, which has the opposite phase of the former. The U phase waveform output that started from the "H" level keeps its level until the timer for setting dead time finishes outputting one-shot pulses even though the three-phase output shift register's content changes from "1" to "0" by the effect of the one-shot pulses. When the timer for setting dead time finishes outputting one-shot pulses, "0" already shifted in the three-phase shift register goes active, and the U phase waveform changes to the Low level. When the timer B2 counter's content becomes $0000_{16}$, the timer A4 counter starts counting the value written to timer A4-1 ($0307_{16}$, $0306_{16}$), and starts outputting one-shot pulses. When timer A4 finishes outputting one-shot pulses, the three-phase shift register's content is shifted one position, but if the three-phase output shift register's content changes from "0" to "1" as a result of the shift, the output level changes from "L" to "H" without waiting for the timer for setting dead time to finish outputting one-shot pulses. A U phase waveform is generated by these workings repeatedly. With the exception that the three-phase output shift register on the U phase side is used, the workings in generating a U

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

**A carrier wave of triangular waveform**




Note 1: When INV14="0" (output wave Low active)
Note 2: When INV14="1" (output wave High active)
Note 3: Set to triangular wave modulation mode and to three-phase mode 1.

**Figure 1.16.7. Timing chart of operation (1)**

Assigning certain values to DU0 (bit 0 at 030A16) and DUB0 (bit 1 at 030A16), and to DU1 (bit 0 at 030B16) and DUB1 (bit 1 at 030B16) allows you to output the waveforms as shown in Figure 1.16.8, that is, to output the U phase alone, to fix $\overline{U}$ phase to "H", to fix the U phase to "H," or to output the $\overline{U}$ phase alone.

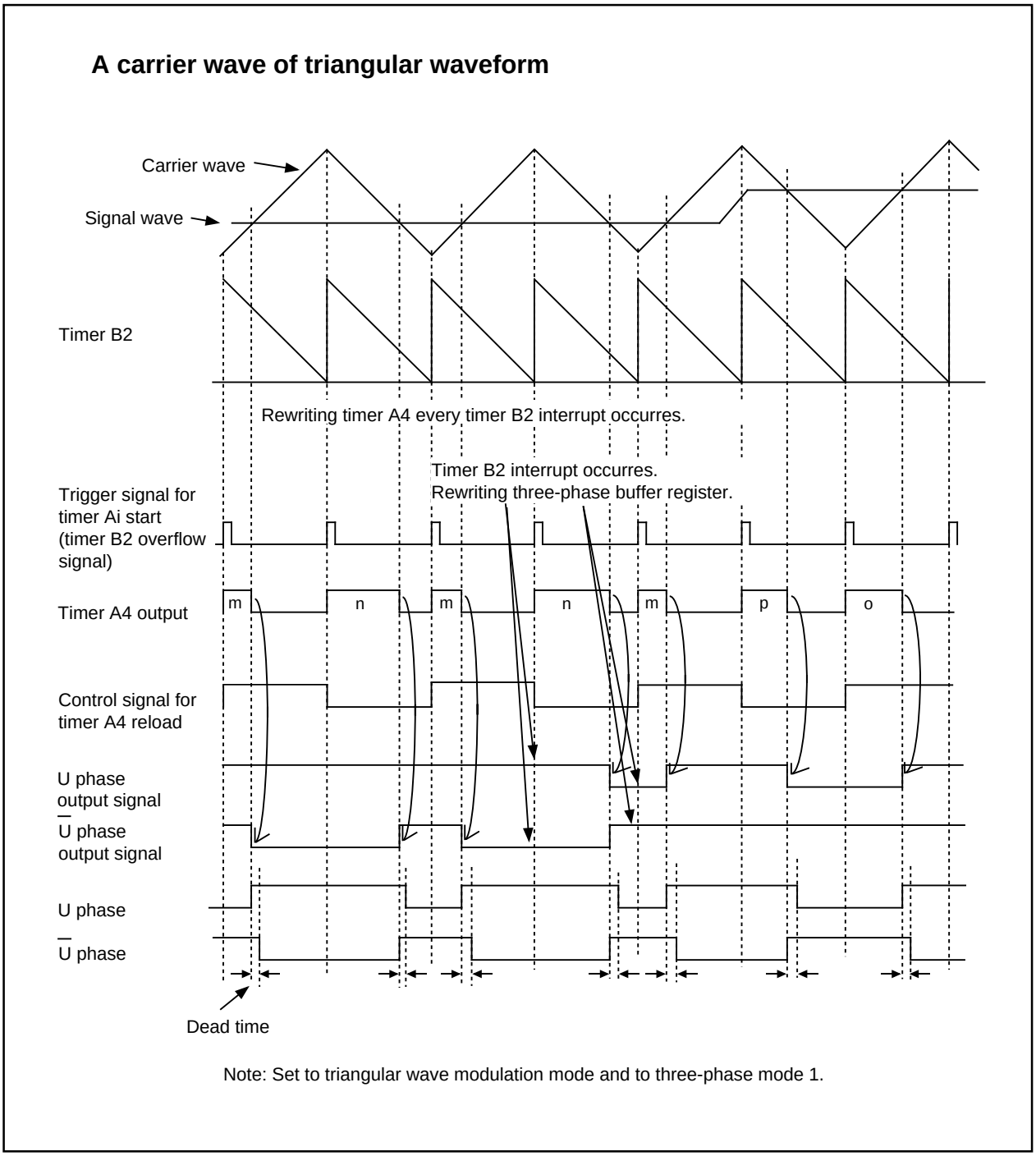



**Figure 1.16.8. Timing chart of operation (2)**

## Sawtooth modulation

To generate a PWM waveform of sawtooth wave modulation, set "1" in the modulation mode select bit (bit 6 at $0308_{16}$). Also, set "0" in the timers A4, A1, and A2-1 control bit (bit 1 at $0309_{16}$). In this mode, the timer registers of timers A4, A1, and of A2 comprise conventional timers A4, A1, and A2 alone, and reload the corresponding timer register's content to the counter every time the timer B2 counter's content becomes $0000_{16}$. The effective interrupt output specification bit (bit 1 at $0308_{16}$) and the effective interrupt output polarity select bit (bit 0 at $0308_{16}$) go nullified.

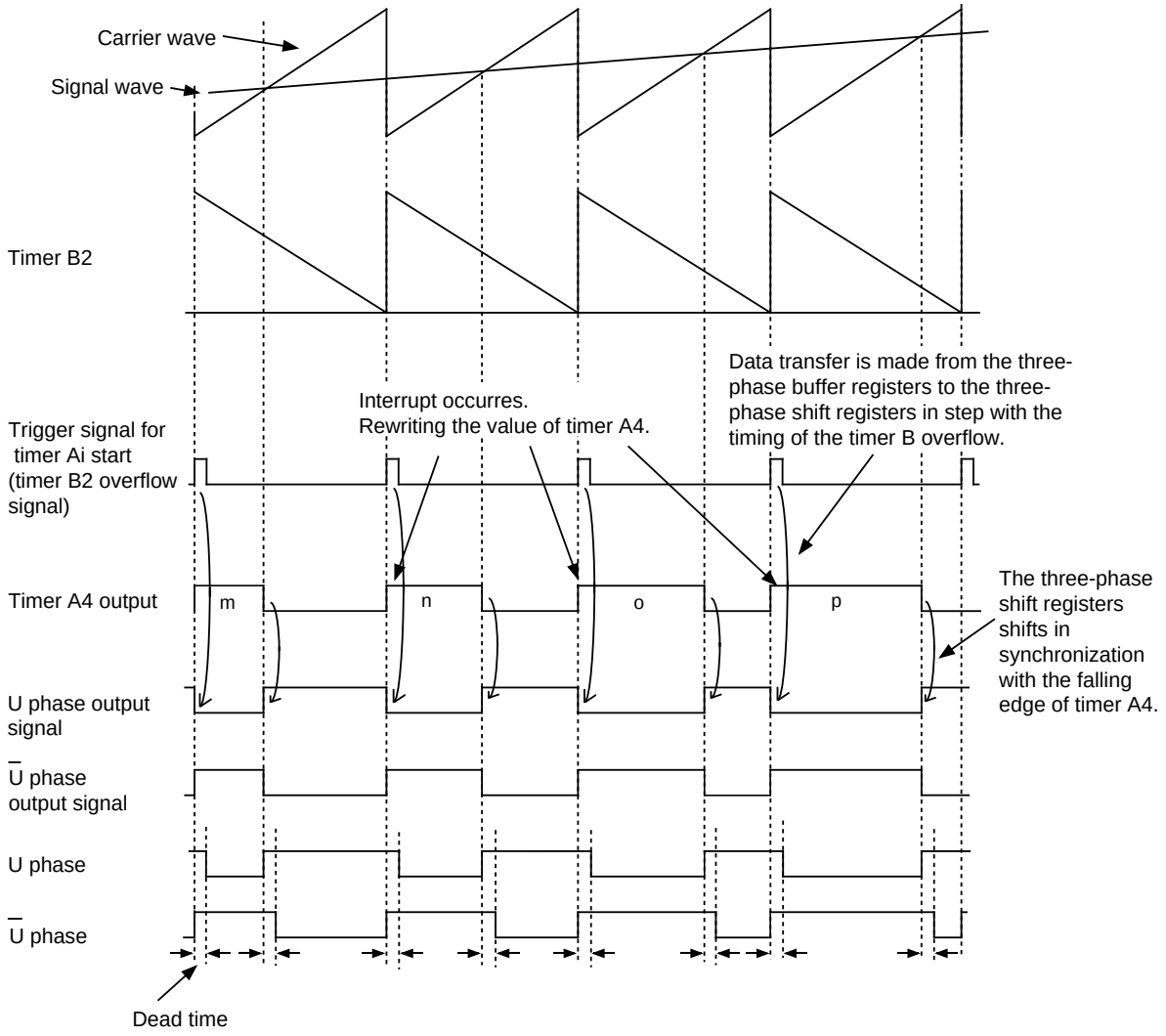
An example of U phase waveform is shown in Figure 1.16.9, and the description of waveform output workings is given below. Set "1" in DU0 (bit 0 at $030A_{16}$), and set "0" in DUB0 (bit 1 at $030A_{16}$). In addition, set "0" in DU1 (bit 0 at $030B_{16}$) and set "1" in DUB1 (bit 1 at $030B_{16}$).

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

The value of DU0 and that of DUB0 are output to the U terminal ($P8_0$) and to the $\overline{U}$ terminal ($P8_1$) respectively. When the timer A4 counter counts the value written to timer A4 ($034F_{16}$, $034E_{16}$) and when timer A4 finishes outputting one-shot pulses, the three-phase output shift register's content is shifted one position, and the value of DU1 and that of DUB1 are output to the U phase output signal and to the $\overline{U}$ output signal respectively. At this time, one-shot pulses are output from the timer for setting dead time used for setting the time over which the "L" level of the U phase waveform doesn't lap over the "L" level of the $\overline{U}$ phase waveform, which has the opposite phase of the former. The U phase waveform output that started from the "H" level keeps its level until the timer for setting dead time finishes outputting one-shot pulses even though the three-phase output shift register's content changes from "1" to "0 "by the effect of the one-shot pulses. When the timer for setting dead time finishes outputting one-shot pulses, 0 already shifted in the three-phase shift register goes effective, and the U phase waveform changes to the "L" level. When the timer B2 counter's content becomes $0000_{16}$, the contents of the three-phase buffer registers DU1 and DU0 are set in the three-phase shift register (U phase), and the contents of DUB1 and DUB0 are set in the three-phase shift register ($\overline{U}$ phase) again.

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

**A carrier wave of sawtooth waveform**

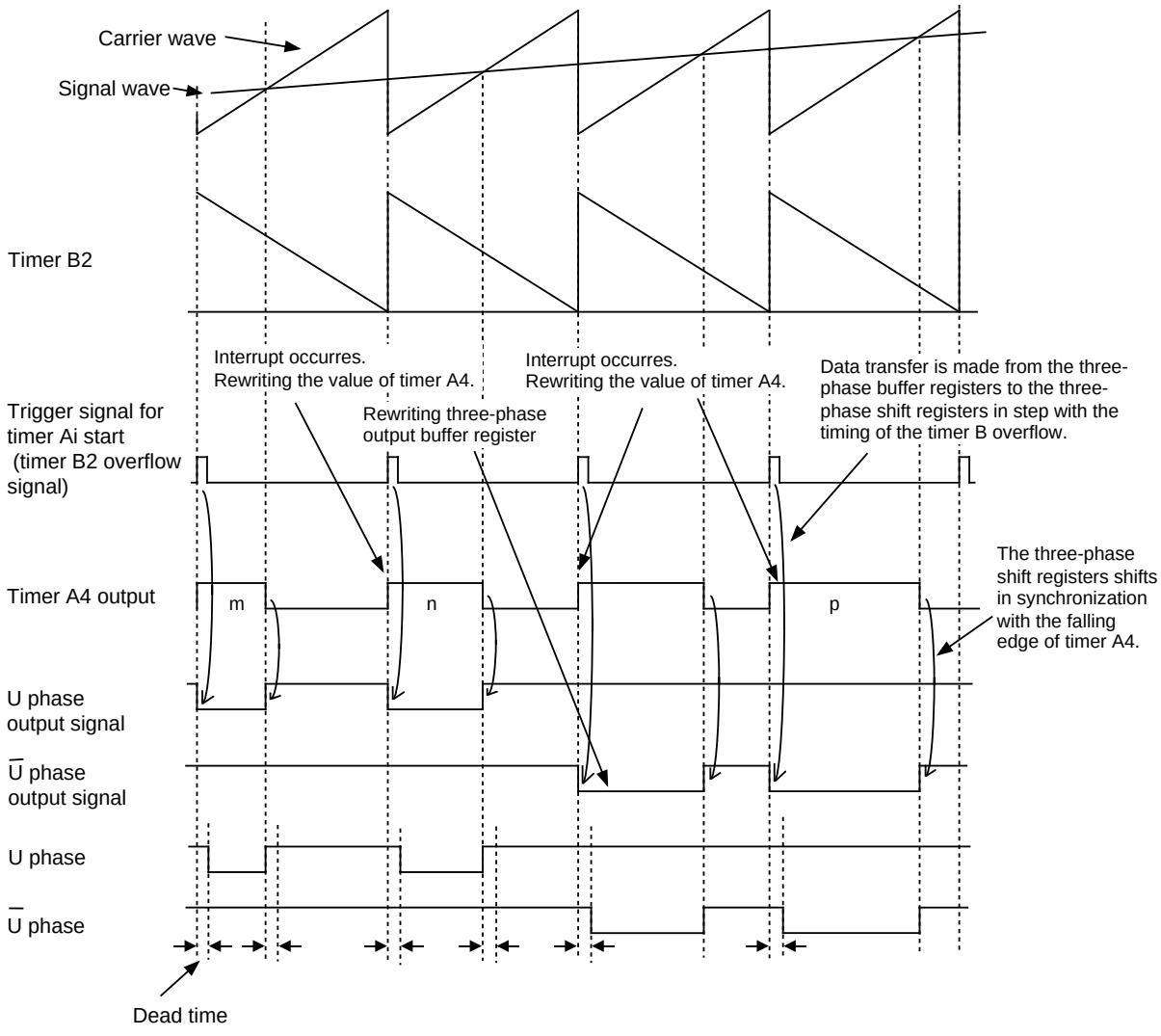



Note: Set to sawtooth modulation mode and to three-phase mode 0.

**Figure 1.16.9. Timing chart of operation (3)**

Setting "1" both in DUB0 and in DUB1 provides a means to output the U phase alone and to fix the $\overline{U}$ phase output to "H" as shown in Figure 1.16.10.

**A carrier wave of sawtooth waveform**




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

**Figure 1.16.10. Timing chart of operation (4)**

## Serial I/O

Serial I/O is configured as five channels: UART0 to UART4.

UARTi (i=0 to 4) each have an exclusive timer to generate a transfer clock, so they operate independently of each other.

Figure 1.17.1 shows the block diagram of UARTi.

UARTi 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 $0368_{16}$, $02E8_{16}$, $0338_{16}$, $0328_{16}$ and $02F8_{16}$) determine whether UARTi is used as a clock synchronous serial I/O or as a UART.

It has the bus collision detection function that generates an interrupt request if the TxD pin and the RxD pin are different in level.

Figures 1.17.2 through 1.17.8 show the registers related to UARTi.

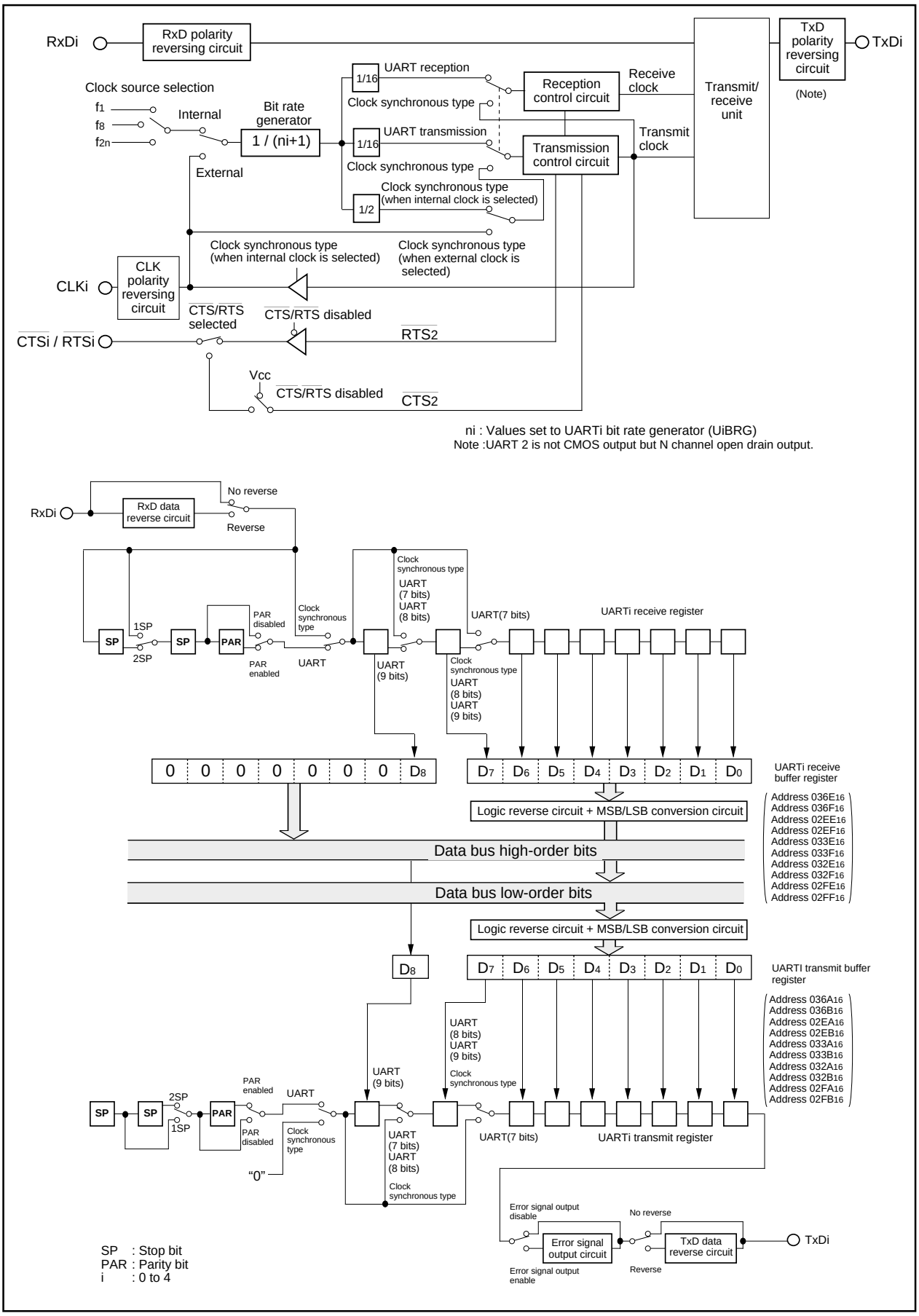
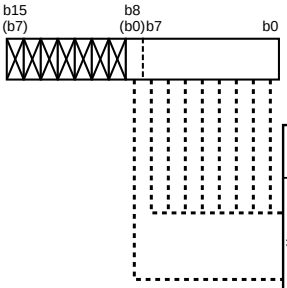



**Figure 1.17.1. Block diagram of UARTi**

## UARTi transmit buffer register (i=0 to 4) (Note)

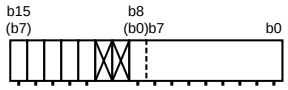



| Symbol | Address | When reset |
|---|---|---|
| UiTB(i=0,1,2) | $036B_{16}$, $036A_{16}$, $02EB_{16}$, $02EA_{16}$, $033B_{16}$, $033A_{16}$ | Indeterminate |
| UiTB(i=3,4) | $032B_{16}$, $032A_{16}$, $02FB_{16}$, $02FA_{16}$ | Indeterminate |

| Bit symbol | Function (Clock synchronous serial I/O mode) | Function (UART mode) | R | W |
|---|---|---|---|---|
| —— | Transmit data | Transmit data | — | ○ |
| —— | —— | Transmit data (9th bit) | — | ○ |
| Nothing is assigned. When write, set "0". When read, their contents are indeterminate. | | | — | — |

Note: Use MOV instruction to write to this register.

## UARTi receive buffer register (i = 0 to 4)




| Symbol | Address | When reset |
|---|---|---|
| UiRB(i=0,1,2) | $036F_{16}$,$036E_{16}$, $02EF_{16}$,$02EE_{16}$, $033F_{16}$,$033E_{16}$ | Indeterminate |
| UiRB(i=3,4) | $032F_{16}$,$032E_{16}$, $02FF_{16}$,$02FE_{16}$ | Indeterminate |

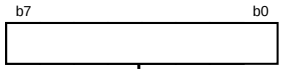
| Bit symbol | Bit name | Function (Clock synchronous serial I/O mode) | Function (UART mode) | R | W |
|---|---|---|---|---|---|
| —— | —— | Receive data | Receive data | ○ | — |
| —— | —— | —— | Receive data (9th bit) | ○ | — |
| Nothing is assigned. When write, set "0". When read, their contents are indeterminate. | | | | — | — |
| | | | | — | — |
| ABT | Arbitration lost detecting flag (Note 1) | 0: Not detectet<br>1: Detected | Invalid | ○ | ○ |
| OER | Overrun error flag (Note 2) | 0: No overrun error<br>1: Overrun error found | 0: No overrun error<br>1: Overrun error found | ○ | — |
| FER | Framing error flag (Note 2) | Invalid | 0: No framing error<br>1: Framing error found | ○ | — |
| PER | Parity error flag (Note 2) | Invalid | 0: No parity error<br>1: Parity error found | ○ | — |
| SUM | Error sum flag (Note 2) | Invalid | 0: No error<br>1: Error found | ○ | — |

Note 1: Arbitration lost detecting flag must always write "0".

Note 2: Bits 15 through 12 are set to $000_2$ when the serial I/O mode select bit (bits 2 to 0 at addresses $0368_{16}$, $02E8_{16}$, $0338_{16}$, $0328_{16}$, $02F8_{16}$) 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 $036E_{16}$, $02EE_{16}$, $033E_{16}$, $032E_{16}$, $02FE_{16}$) is read.

**Figure 1.17.2. Serial I/O-related registers (1)**

## UARTi bit rate generator (i=0 to 4) (Note 1, 2)

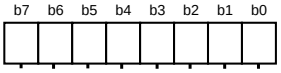



| Symbol | Address | When reset |
|---|---|---|
| UiBRG(i=0 to 4) | $0369_{16}$, $02E9_{16}$, $0339_{16}$, $0329_{16}$, $02F9_{16}$ | Indeterminate |

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

Note 1: Use MOV instruction to write to this register.
Note 2: Write a value to this register while transmit/receive halts.

## UARTi transmit/receive mode register (i=0 to 4)



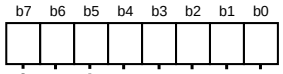

| Symbol | Address | When reset |
|---|---|---|
| UiMR(i=0 to 4) | $0368_{16}$, $02E8_{16}$, $0338_{16}$, $0328_{16}$, $02F8_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function (Clock synchronous serial I/O mode) | Function (UART mode) | R | W |
|---|---|---|---|---|---|
| SMD0 | Serial I/O mode select bit | b2 b1 b0<br>0 0 0: Serial I/O invalid<br>0 0 1: Serial I/O mode<br>0 1 0: $I^2C$ mode<br>Must not be set except above | b2 b1 b0<br>0 0 0: Serial I/O invalid<br>1 0 0: Transfer data 7 bits long<br>1 0 1: Transfer data 8 bits long<br>1 1 0: Transfer data 9 bits long<br>Must not be set except above | ○ | ○ |
| SMD1 | | | | ○ | ○ |
| SMD2 | | | | ○ | ○ |
| CKDIR | Internal/external clock select bit | 0 : Internal clock (Note 1)<br>1 : External clock (Note 2) | 0 : Internal clock<br>1 : External clock (Note 2) | ○ | ○ |
| STPS | Stop bit length select bit | Invalid | 0 : One stop bit<br>1 : Two stop bits | ○ | ○ |
| PRY | Odd/even parity select bit | Invalid | Valid when bit 6 = "1"<br>0 : Odd parity<br>1 : Even parity | ○ | ○ |
| PRYE | Parity enable bit | Invalid | 0 : Parity disabled<br>1 : Parity enabled | ○ | ○ |
| IOPOL | TxD,RxD input/ output polarity switch bit (Note 3) | 0: No reversed<br>1: Reversed | | ○ | ○ |

Note 1: Select CLK output by the corresponding function select registers A, B and C.
Note 2: Set the corresponding function select register A to the I/O port.
Note 3: Normally set "0".

**Figure 1.17.3. Serial I/O-related registers (2)**

## UARTi transmit/receive control register 0 (i=0 to 4)



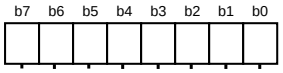

| Symbol | Address | When reset |
|---|---|---|
| UiC0(i=0 to 4) | $036C_{16}$, $02EC_{16}$, $033C_{16}$, $032C_{16}$, $02FC_{16}$ | $08_{16}$ |

| Bit symbol | Bit name | Function (Clock synchronous serial I/O mode) | Function (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: $f_{2n}$ is selected<br>1 1: Must not be set | | ○ | ○ |
| CLK1 | | | | ○ | ○ |
| CRS | $\overline{CTS}/\overline{RTS}$ function select bit | Valid when bit 4 = "0"<br>0 : $\overline{CTS}$ function is selected (Note 1)<br>1 : $\overline{RTS}$ function is selected (Note 2) | | ○ | ○ |
| TXEPT | Transmit register empty flag | 0 : Data present in transmit register (during transmission)<br>1 : No data present in transmit register (transmission completed) | | ○ | — |
| CRD | $\overline{CTS}/\overline{RTS}$ disable bit | 0 : $\overline{CTS}/\overline{RTS}$ function enabled<br>1 : $\overline{CTS}/\overline{RTS}$ function disabled | | ○ | ○ |
| NCH (Note 3) | Data output select bit | 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 | Set to "0" | ○ | ○ |
| UFORM | Transfer format select bit (Note 4) | 0 : LSB first<br>1 : MSB first | | ○ | ○ |

Note 1: Set the corresponding function select register A to I/O port, and port direction register to "0"
Note 2: Select $\overline{RTS}$ output using the corresponding function select registers A, B and C.
Note 3: UART2 transfer pin (TxD2:$P7_0$) is N-channel open drain output. It is not set to CMOS output.
Note 4: Valid only in clock syncronous serial I/O mode and 8 bits UART mode.

**Figure 1.17.4. Serial I/O-related registers (3)**

## UARTi transmit/receive control register 1 (i=0 to 4)

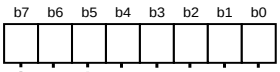



| Symbol | Address | When reset |
|---|---|---|
| UiC1(i=0 to 4) | $036D_{16}$, $02ED_{16}$, $033D_{16}$, $032D_{16}$, $02FD_{16}$ | $02_{16}$ |

| Bit symbol | Bit name | Function (Clock synchronous serial I/O mode) | Function (UART mode) | R | W |
|---|---|---|---|---|---|
| TE | Transmit enable bit | 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 | | ○ | — |
| RE | Receive enable bit | 0: Reception disabled<br>1: Reception enabled | | ○ | ○ |
| RI | Receive complete flag | 0: Data present in receive buffer register<br>1: No data present in receive buffer register | | ○ | — |
| UiIRS | UARTi transmit interrupt cause select bit | 0: Transmit buffer empty (TI = 1)<br>1: Transmit is completed (TXEPT = 1) | | ○ | ○ |
| UiRRM | UARTi continuous receive mode enable bit | 0: Continuous receive mode disabled<br>1: Continuous receive mode enabled | Set to "0" | ○ | ○ |
| UiLCH | Data logic select bit | 0: No reverse<br>1: Reverse | | ○ | ○ |
| SCLKSTPB /UiERE | Clock divide synchronizing stop bit /error signal output enable bit | Clock divide synchronizing stop bit<br>0: Synchronizing stop<br>1: Synchronous start (Note) | Set to "0" | ○ | ○ |

Note :When this bit and bit 7 of UARTi special mode register 2 are set, clock synchronizing function is used.

## UARTi special mode register (i=0 to 4)



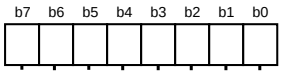

| Symbol | Address | When reset |
|---|---|---|
| UiSMR(i=0 to 4) | $0367_{16}$, $02E7_{16}$, $0337_{16}$, $0327_{16}$, $02F7_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function (Clock synchronous serial I/O mode) | Function (UART mode) | R | W |
|---|---|---|---|---|---|
| IICM | IIC mode select bit | 0: Normal mode<br>1: IIC mode | Set to "0" | ○ | ○ |
| ABC | Arbitration lost detecting flag control bit | 0: Update per bit<br>1: Update per byte | Set to "0" | ○ | ○ |
| BBS | Bus busy flag | 0: STOP condition detected<br>1: START condition detected | Set to "0" | ○ | ○ (Note 1) |
| LSYN | SCLL sync output enable bit | 0: Disabled<br>1: Enabled | Set to "0" | ○ | ○ |
| ABSCS | Bus collision detect sampling clock select bit | Set to "0" | 0: Rising edge of transfer clock<br>1: Underflow signal of timer Ai (Note 2) | ○ | ○ |
| ACSE | Auto clear function select bit of transmit enable bit | Set to "0" | 0: No auto clear function<br>1: Auto clear at occurrence of bus | ○ | ○ |
| SSS | Transmit start condition select bit | Set to "0" | 0: Ordinary<br>1: Falling edge of RxDi | ○ | ○ |
| SCLKDIV | Clock divide set bit | 0: No divided<br>1: Divided-by-2 (Note 3) | Set to "0" | ○ | ○ |

Note 1: Nothing but "0" may be written.
Note 2: UART0: timer A3 underflow signal, UART1: timer A4 underflow signal, UART2: timer A0 underflow signal, UART3: timer A3 underflow signal, UART4: timer A4 underflow signal.
Note 3: When this bit and bit 7 of UARTi transmit/receive control register 1 are set, clock synchronizing function is used.

**Figure 1.17.5. Serial I/O-related registers (4)**

## UARTi special mode register 2 (i=0 to 4)




| Symbol | Address | When reset |
|---|---|---|
| UiSMR2(i=0 to 4) | $0366_{16}$, $02E6_{16}$, $0336_{16}$, $0326_{16}$, $02F6_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| IICM2 | IIC mode select bit 2 | 0: NACK/ACK interrupt (DMA source - ACK)<br>Transfer to receive buffer at the rising edge of last bit of receive clock<br>Receive interrupt occurs at the rising edge of last bit of receive clock<br>1: UART transfer/receive interrupt (DMA source - UART receive)<br>Transfer to receive buffer at the falling edge of last bit of receive clock<br>Receive interrupt occurs at the falling edge of last bit of receive clock | ○ | ○ |
| 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 | ○ | ○ |
| STC | UARTi initialize bit | 0: Disabled<br>1: Enabled | ○ | ○ |
| SWC2 | SCL wait output bit 2 | 0: UARTi clock<br>1: 0 output | ○ | ○ |
| SDHI | SDA output inhibit bit | 0: Disabled<br>1: Enabled (high impedance) | ○ | ○ |
| SU1HIM | External clock synchronizing enable bit | 0: Synchronous disabled<br>1: Synchronous enabled | ○ | ○ |

**Figure 1.17.6. Serial I/O-related registers (5)**

## UARTi special mode register 3 (i=0 to 4)

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

| Symbol | Address | When reset |
|---|---|---|
| UiSMR3(i=0 to 4) | $0365_{16}$, $02E5_{16}$, $0335_{16}$, $0325_{16}$, $02F5_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SSE | $\overline{SS}$ port function enable bit (Note 1) | 0: $\overline{SS}$ function disabled<br>1: $\overline{SS}$ function enabled | ○ | ○ |
| CKPH | Clock phase set bit | 0: Without clock delay<br>1: With clock delay | ○ | ○ |
| DINC | Serial input port set bit | 0: Select TxDi and RxDi (master mode)<br>1: Select STxDi and SRxDi (slave mode) | ○ | ○ |
| NODC | Clock output select bit | 0: CLKi is CMOS output (Note 2)<br>1: CLKi is N-channel open drain output (Note 3) | ○ | ○ |
| ERR | Fault error flag | 0: Without fault error<br>1: With fault error (Note 4) | ○ | ○ |
| DL0 | SDAi(TxDi) digital delay time set bit (Note 5,6) | b7 b6 b5<br>000 :Without delay<br>001 :2-cycle of BRG count source<br>010 :3-cycle of BRG count source<br>011 :4-cycle of BRG count source<br>100 :5-cycle of BRG count source<br>101 :6-cycle of BRG count source<br>110 :7-cycle of BRG count source<br>111 :8-cycle of BRG count source | ○ | ○ |
| DL1 | | | ○ | ○ |
| DL2 | | | ○ | ○ |

Note 1: Set $\overline{SS}$ function after setting $\overline{CTS}/\overline{RTS}$ disable bit (bit 4 of UARTi transfer/receive control register 0) to "1".
Note 2: Set CLKi and TxDi both for output using the CLKi and TxDi function select register A. Set the RxDi function select register A for input/output port and the port direction register to "0".
Note 3: Set STxDi for output using the STxDi function select registers A and B. Set the CLKi and SRxDi function select register A for input/output port and the port direction register to "0".
Note 4: Nothing but "0" may be written.
Note 5: These bits are used for SDAi (TxDi) output digital delay when using UARTi for IIC interface. Otherwise, must set to "000".
Note 6: When external clock is selected, delay is increased approximately 100ns.

**Figure 1.17.7. Serial I/O-related registers (6)**

## UARTi special mode register 4 (i=0 to 4)

b7 b6 b5 b4 b3 b2 b1 b0

| Symbol | Address | When reset |
|---|---|---|
| UiSMR4(i=0 to 4) | $0364_{16}$, $02E4_{16}$, $0334_{16}$, $0324_{16}$, $02F4_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| STAREQ | Start condition generate bit (Note) | 0: Clear<br>1: Start | ○ | ○ |
| RSTAREQ | Restart condition generate bit (Note) | 0: Clear<br>1: Start | ○ | ○ |
| STPREQ | Stop condition generate bit (Note) | 0: Clear<br>1: Start | ○ | ○ |
| STSPSEL | SCL, SDA output select bit | 0: Ordinal block<br>1: Start/stop condition generate block | ○ | ○ |
| ACKD | ACK data bit | 0: ACK<br>1: NACK | ○ | ○ |
| ACKC | ACK data output enable bit | 0: SI/O data output<br>1: ACKD output | ○ | ○ |
| SCLHI | SCL output stop enable bit | 0: Disabled<br>1: Enabled | ○ | ○ |
| SWC9 | SCL wait output bit 3 | 0: SCL "L" hold disabled<br>1: SCL "L" hold enabled | ○ | ○ |

Note :When start condition is generated, these bits automatically become "0".

## External interrupt request cause select register

b7 b6 b5 b4 b3 b2 b1 b0

| Symbol | Address | When reset |
|---|---|---|
| IFSR | $031F_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| IFSR0 | INT0 interrupt polarity select bit (Note) | 0 : One edge<br>1 : Both edges | ○ | ○ |
| IFSR1 | INT1 interrupt polarity select bit (Note) | 0 : One edge<br>1 : Both edges | ○ | ○ |
| IFSR2 | INT2 interrupt polarity select bit (Note) | 0 : One edge<br>1 : Both edges | ○ | ○ |
| IFSR3 | INT3 interrupt polarity select bit | 0 : One edge<br>1 : Both edges | ○ | ○ |
| IFSR4 | INT4 interrupt polarity select bit | 0 : One edge<br>1 : Both edges | ○ | ○ |
| IFSR5 | INT5 interrupt polarity select bit | 0 : One edge<br>1 : Both edges | ○ | ○ |
| IFSR6 | UART0/3 interrupt cause select bit | 0 : UART3 bus collision /start,stop detect/false error detect<br>1 : UART0 bus collision /start,stop detect/false error detect | ○ | ○ |
| IFSR7 | UART1/4 interrupt cause select bit | 0 : UART4 bus collision /start,stop detect/false error detect<br>1 : UART1 bus collision /start,stop detect/false error detect | ○ | ○ |

Note :When level sense is selected, set this bit to "0". When both edges are selected, set the corresponding polarity switching bit of INT interrupt control register to "0" (falling edge).

**Figure 1.17.8. Serial I/O-related registers (7)**

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

The clock synchronous serial I/O mode uses a transfer clock to transmit and receive data. Tables 1.18.1 and 1.18.2 list the specifications of the clock synchronous serial I/O mode.

**Table 1.18.1. Specifications of clock synchronous serial I/O mode (1/2)**

| Item | Specification |
|---|---|
| Transfer data format | • Transfer data length: 8 bits |
| Transfer clock | • When internal clock is selected (bit 3 at addresses $0368_{16}$, $02E8_{16}$, $0338_{16}$, $0328_{16}$, $02F8_{16}$ = "0") : fi/ 2(m+1) (Note 1) fi = $f_1$, $f_8$, $f_{2n}$<br>– CLK is selected by the corresponding peripheral function select register A, B and C.<br>• When external clock is selected (bit 3 at addresses $0368_{16}$, $02E8_{16}$, $0338_{16}$ , $0328_{16}$, $02F8_{16}$= "1") : Input from CLKi pin<br>– Set the corresponding function select register A to I/O port |
| 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 $036D_{16}$, $02ED_{16}$, $033D_{16}$, $032D_{16}$, $02FD_{16}$) = "1"<br>– Transmit buffer empty flag (bit 1 at addresses $036D_{16}$, $02ED_{16}$, $033D_{16}$, $032D_{16}$, $02FD_{16}$) = "0"<br>– When $\overline{CTS}$ function selected, $\overline{CTS}$ input level = "L"<br>– TxD output is selected by the corresponding peripheral function select register A, B and C.<br>• Furthermore, if external clock is selected, the following requirements must also be met:<br>– CLKi polarity select bit (bit 6 at addresses $036C_{16}$, $02EC_{16}$, $033C_{16}$, $032C_{16}$, $02FC_{16}$) = "0": CLKi input level = "H"<br>– CLKi polarity select bit (bit 6 at addresses $036C_{16}$, $02EC_{16}$, $033C_{16}$, $032C_{16}$, $02FC_{16}$) = "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 $036D_{16}$, $02ED_{16}$, $033D_{16}$, $032D_{16}$, $02FD_{16}$) = "1"<br>– Transmit enable bit (bit 0 at addresses $036D_{16}$, $02ED_{16}$, $033D_{16}$, $032D_{16}$, $02FD_{16}$) = "1"<br>– Transmit buffer empty flag (bit 1 at addresses $036D_{16}$, $02ED_{16}$, $033D_{16}$, $032D_{16}$, $02FD_{16}$) = "0"<br>• Furthermore, if external clock is selected, the following requirements must also be met:<br>– CLKi polarity select bit (bit 6 at addresses $036C_{16}$, $02EC_{16}$, $033C_{16}$, $032C_{16}$, $02FC_{16}$) = "0": CLKi input level = "H"<br>– CLKi polarity select bit (bit 6 at addresses $036C_{16}$, $02EC_{16}$, $033C_{16}$, $032C_{16}$, $02FC_{16}$) = "1": CLKi input level = "L" |
| Interrupt request generation timing | • When transmitting<br>– Transmit interrupt cause select bit (bit 4 at address $036D_{16}$, $02ED_{16}$, $033D_{16}$, $032D_{16}$, $02FD_{16}$) = "0": Interrupts requested when data transfer from UARTi transfer buffer register to UARTi transmit register is completed<br>– Transmit interrupt cause select bit (bit 4 at address $036D_{16}$, $02ED_{16}$, $033D_{16}$, $032D_{16}$, $02FD_{16}$) = "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 |

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

**Table 1.18.2. Specifications of clock synchronous serial I/O mode (2/2)**

| Item | Specification |
|---|---|
| Error detection | • Overrun error (Note)<br>This error occurs when the next data is started to receive and 6.5 transfer clock is elapsed before UARTi receive buffer register are read out. |
| 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>• Reversing serial data logic<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<br>This function is reversing TxD port output and RxD port input. All I/O data level is reversed. |

Note : If an overrun error occurs, the UARTi receive buffer will have the next data written in.

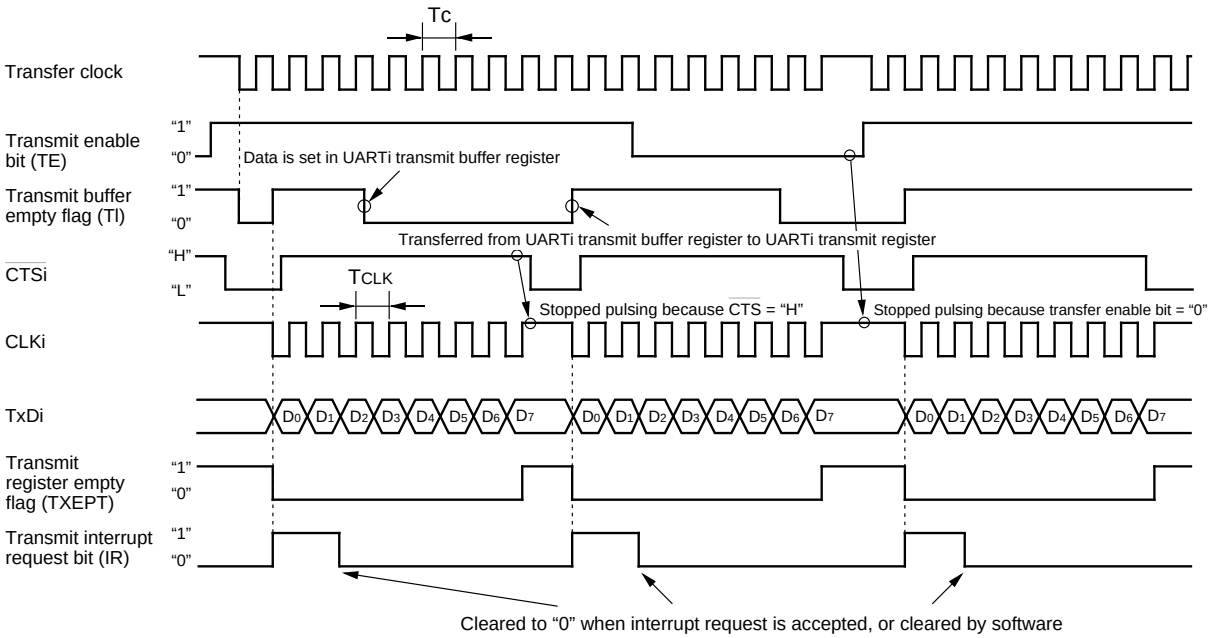
Table 1.18.3 lists the functions of the input/output pins during clock synchronous serial I/O mode. Note that for a period from when the UARTi operation mode is selected to when transfer starts, the TxDi pin outputs a “H”. (If the N-channel open drain is selected, this pin is in floating state.)

**Table 1.18.3. Input/output pin functions in clock synchronous serial I/O mode**

| Pin name | Function | Method of selection |
|---|---|---|
| TxDi (P6$_3$, P6$_7$, P7$_0$, P9$_2$, P9$_6$) | Serial data output (Note 1) | (Outputs dummy data when performing reception only) |
| RxDi (P6$_2$, P6$_6$, P7$_1$, P9$_1$, P9$_7$) | Serial data input (Note 2) | Port P6$_2$, P6$_6$, P7$_1$, P9$_1$ and P9$_7$ direction register (bits 2 and 6 at address 03C2$_{16}$, bit 1 at address 03C3$_{16}$, bit 1 and 7 at address 03C7$_{16}$)= “0”<br>(Can be used as an input port when performing transmission only) |
| CLKi (P6$_1$, P6$_5$, P7$_2$, P9$_0$, P9$_5$) | Transfer clock output (Note 1) | Internal/external clock select bit (bit 3 at addresses 036$8_{16}$, 02E$8_{16}$, 033$8_{16}$, 032$8_{16}$, 02F$8_{16}$) = “0” |
| | Transfer clock input (Note 2) | Internal/external clock select bit (bit 3 at addresses 036$8_{16}$, 02E$8_{16}$, 033$8_{16}$, 032$8_{16}$, 02F$8_{16}$) = “1”<br>Port P6$_1$, P6$_5$, P7$_2$, P9$_0$ and P9$_5$ direction register (bits 1 and 5 at address 03C2$_{16}$, bit 2 at address 03C3$_{16}$, bit 0 and 5 at address 03C7$_{16}$) = “0” |
| $\overline{\text{CTS}}$i/$\overline{\text{RTS}}$i (P6$_0$, P6$_4$, P7$_3$, P9$_3$, P9$_4$) | $\overline{\text{CTS}}$ input (Note 2) | $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit (bit 4 at addresses 036C$_{16}$, 02EC$_{16}$, 033C$_{16}$, 032C$_{16}$, 02FC$_{16}$) =“0”<br>$\overline{\text{CTS}}$/$\overline{\text{RTS}}$ function select bit (bit 2 at addresses 036C$_{16}$, 02EC$_{16}$, 033C$_{16}$, 032C$_{16}$, 02FC$_{16}$) = “0”<br>Port P6$_0$, P6$_4$, P7$_3$, P9$_3$ and P9$_4$ direction register (bits 0 and 4 at address 03C2$_{16}$, bit 3 at address 03C3$_{16}$, bits 3 and 4 at address 03C7$_{16}$) = “0” |
| | $\overline{\text{RTS}}$ output (Note 1) | $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit (bit 4 at addresses 036C$_{16}$, 02EC$_{16}$, 033C$_{16}$, 032C$_{16}$, 02FC$_{16}$) = “0”<br>$\overline{\text{CTS}}$/$\overline{\text{RTS}}$ function select bit (bit 2 at addresses 036C$_{16}$, 02EC$_{16}$, 033C$_{16}$, 032C$_{16}$, 02FC$_{16}$) = “1” |
| | Programmable I/O port (Note 2) | $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit (bit 4 at addresses 036C$_{16}$, 02EC$_{16}$, 033C$_{16}$, 032C$_{16}$, 02FC$_{16}$) = “1” |

Note 1: Select TxD output, CLK output and $\overline{\text{RTS}}$ output by the corresponding function select registers A, B and C.
Note 2: Select I/O port by the corresponding function select register A.

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

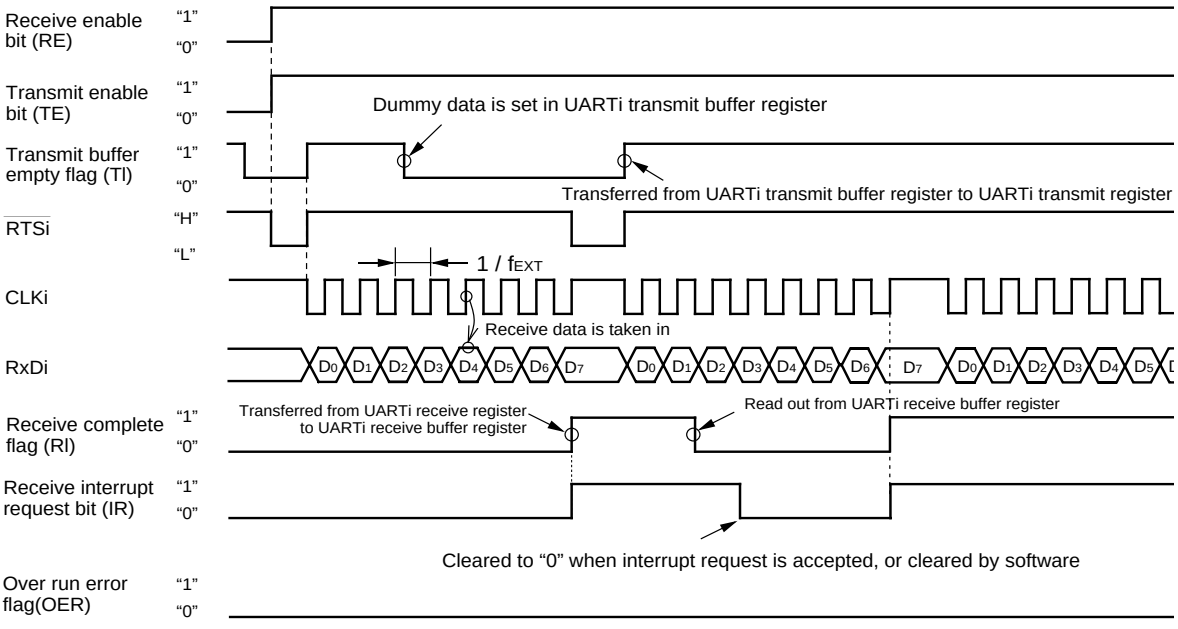



Shown in ( ) are bit symbols.

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(m + 1) / fi
fi: frequency of BRGi count source (f1, f8, f2n)
m: 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

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

**Figure 1.18.1. Typical transmit/receive timings in clock synchronous serial I/O mode**

**(a) Polarity select function**

As shown in Figure 1.18.2, the CLK polarity select bit (bit 6 at addresses 036C16, 02EC16, 033C16, 032C16, 02FC16) allows selection of the polarity of the transfer clock.

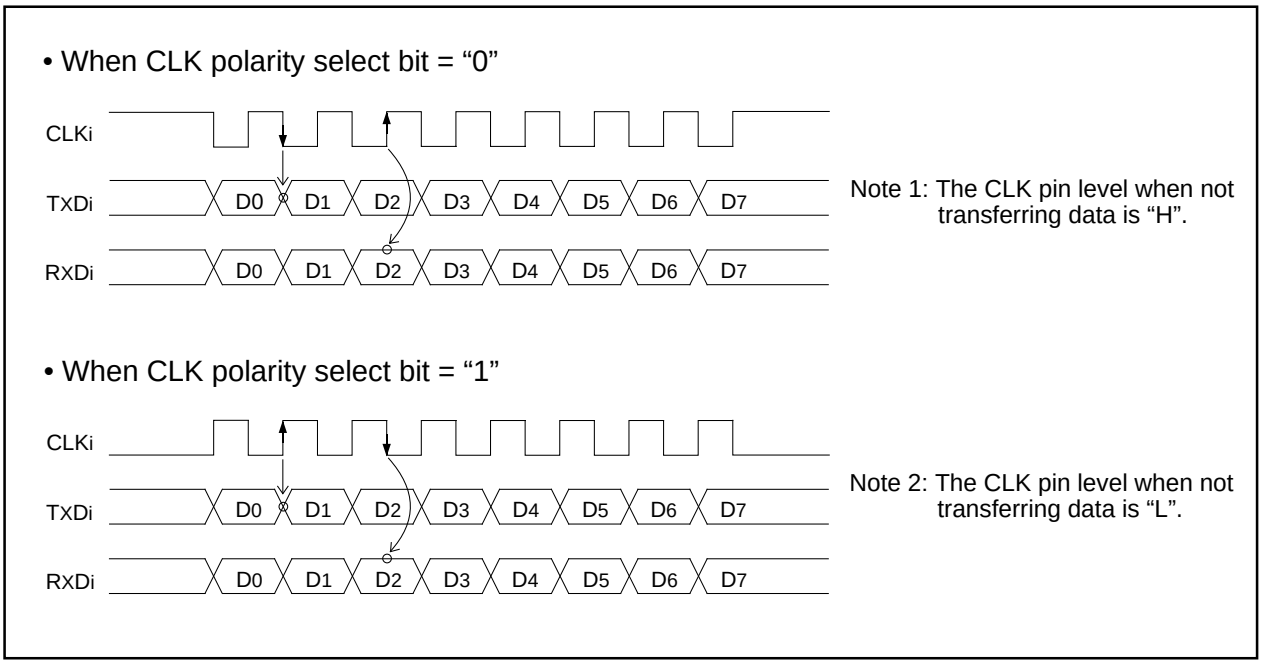



**Figure 1.18.2. Polarity of transfer clock**

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

As shown in Figure 1.18.3, when the transfer format select bit (bit 7 at addresses 036C16, 02EC16, 033C16, 032C16, 02FC16) = “0”, the transfer format is “LSB first”; when the bit = “1”, the transfer format is “MSB first”.

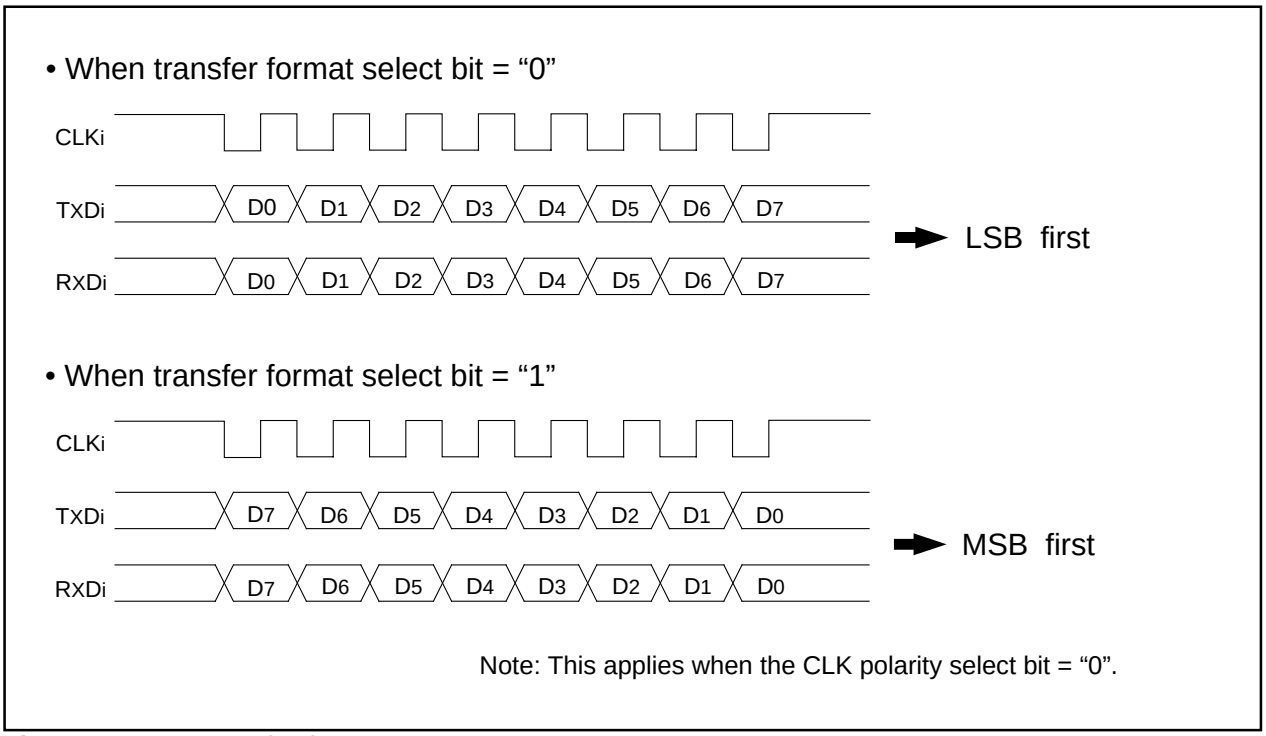



**Figure 1.18.3. Transfer format**

**(c) Continuous receive mode**

If the continuous receive mode enable bit (bit 5 at address 036D16, 02ED16, 033D16, 032D16, 02FD16) 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 back to the transmit buffer register again.

**(d) Serial data logic switch function**

When the data logic select bit (bit6 at address 036D16, 02ED16, 033D16, 032D16, 02FD16) = "1", and writing to transmit buffer register or reading from receive buffer register, data is reversed. Figure 1.18.4 shows the timing example of serial data logic switch.

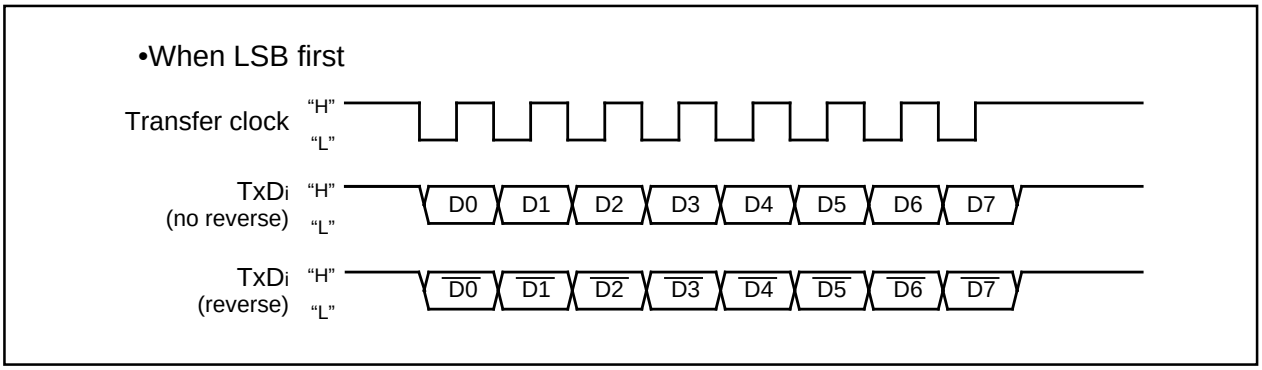



**Figure 1.18.4. Timing for switching serial data logic**

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

The UART mode allows transmitting and receiving data after setting the desired transfer rate and transfer data format. Tables 1.19.1 and 1.19.2 list the specifications of the UART mode. Figure 1.19.1 shows the UARTi transmit/receive mode register.

**Table 1.19.1. Specifications of UART Mode (1/2)**

| 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 $0368_{16}$, $02E8_{16}$, $0338_{16}$, $0328_{16}$, $02F8_{16}$ = "0") : fi/16(m+1) (Note 1) fi = $f_1$, $f_8$, $f_{2n}$<br>• When external clock is selected (bit 3 at addresses $0368_{16}$, $02E8_{16}$, $0338_{16}$, $0328_{16}$, $02F8_{16}$ ="1") : $f_{EXT}$/16(m+1)(Note 1, 2) |
| Transmission/reception control | • $\overline{CTS}$ function, $\overline{RTS}$ function, $\overline{CTS}$/$\overline{RTS}$ function chosen to be invalid |
| Transmission start condition | • To start transmission, the following requirements must be met:<br>- Transmit enable bit (bit 0 at addresses $036D_{16}$, $02ED_{16}$, $033D_{16}$, $032D_{16}$, $02FD_{16}$) = "1"<br>- Transmit buffer empty flag (bit 1 at addresses $036D_{16}$, $02ED_{16}$, $033D_{16}$, $032D_{16}$, $02FD_{16}$) = "0"<br>- When $\overline{CTS}$ function selected, $\overline{CTS}$ input level = "L"<br>- TxD output is selected by the corresponding peripheral function select register A, B and C. |
| Reception start condition | • To start reception, the following requirements must be met:<br>- Receive enable bit (bit 2 at addresses $036D_{16}$, $02ED_{16}$, $033D_{16}$, $032D_{16}$, $02FD_{16}$) = "1"<br>- Start bit detection |
| Interrupt request generation timing | • When transmitting<br>- Transmit interrupt cause select bits (bit 4 at address $036D_{16}$, $02ED_{16}$, $033D_{16}$, $032D_{16}$, $02FD_{16}$) = "0": Interrupts requested when data transfer from UARTi transfer buffer register to UARTi transmit register is completed<br>- Transmit interrupt cause select bits (bit 4 at address $036D_{16}$, $02ED_{16}$, $033D_{16}$, $032D_{16}$, $02FD_{16}$) = "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 started to receive and 6.5 transfer clock is elapsed before UARTi receive buffer register are read out. |

Note 1: 'm' 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 be over written with the next data.

**Table 1.19.2. Specifications of UART Mode (2/2)**

| Item | Specification |
|---|---|
| Error detection | • Framing error<br>This error occurs when the number of stop bits set is not detected<br>• Parity error<br>If parity is enabled this error occurs when, 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 |
| Select function | • Serial data logic switch<br>This function reveres the logic value of transferring data. Start bit, parity bit and stop bit are not reversed.<br>• TxD, RxD I/O polarity switch<br>This function reveres the TxD port output and RxD port input. All I/O data level is reversed. |

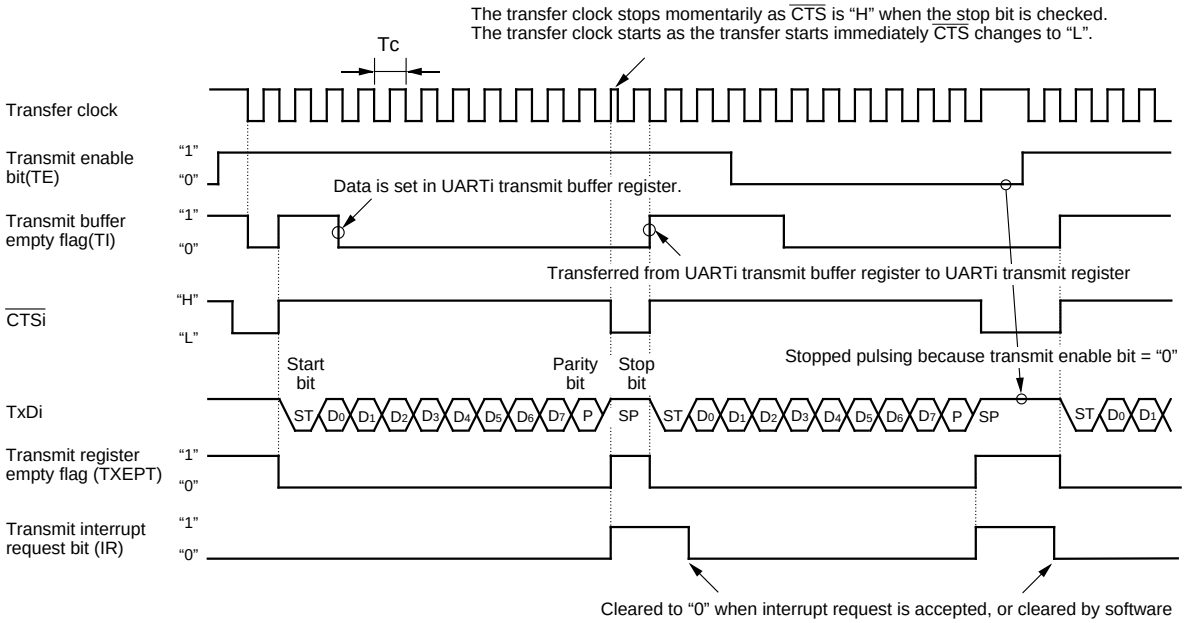
Table 1.19.3 lists the functions of the input/output pins in UART mode. Note that for a period from when the UARTi operation mode is selected to when transfer starts, the TxDi pin outputs a "H". (If the N-channel open drain is selected, this pin is in floating state.)

**Table 1.19.3. Input/output pin functions in UART mode**

| Pin name | Function | Method of selection |
|---|---|---|
| TxDi (P6$_3$, P6$_7$, P7$_0$, P9$_2$, P9$_6$) | Serial data output (Note 1) | |
| RxDi (P6$_2$, P6$_6$, P7$_1$, P9$_1$, P9$_7$) | Serial data input (Note 2) | Port P6$_2$, P6$_6$, P7$_1$, P9$_1$ and P9$_7$ direction register (bits 2 and 6 at address 03C2$_{16}$, bit 1 at address 03C3$_{16}$, bit 1 and 7 at address 03C7$_{16}$)= "0"<br>(Can be used as an input port when performing transmission only) |
| CLKi (P6$_1$, P6$_5$, P7$_2$, P9$_0$, P9$_5$) | Programmable I/O port (Note 2) | Internal/external clock select bit (bit 3 at addresses 036$8_{16}$, 02E$8_{16}$, 033$8_{16}$, 032$8_{16}$, 02F$8_{16}$) = "0" |
| | Transfer clock input (Note 2) | Internal/external clock select bit (bit 3 at addresses 036$8_{16}$, 02E$8_{16}$, 033$8_{16}$, 032$8_{16}$, 02F$8_{16}$) = "1"<br>Port P6$_1$, P6$_5$, P7$_2$, P9$_0$ and P9$_5$ direction register (bits 1 and 5 at address 03C2$_{16}$, bit 2 at address 03C3$_{16}$, bits 0 and 5 at address 03C7$_{16}$) = "0" |
| $\overline{\text{CTS}}$i/$\overline{\text{RTS}}$i (P6$_0$, P6$_4$, P7$_3$, P9$_3$, P9$_4$) | $\overline{\text{CTS}}$ input (Note 2) | $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit (bit 4 at addresses 036C$_{16}$, 02EC$_{16}$, 033C$_{16}$, 032C$_{16}$, 02FC$_{16}$) ="0"<br>$\overline{\text{CTS}}$/$\overline{\text{RTS}}$ function select bit (bit 2 at addresses 036C$_{16}$, 02EC$_{16}$, 033C$_{16}$, 032C$_{16}$, 02FC$_{16}$) = "0"<br>Port P6$_0$, P6$_4$, P7$_3$, P9$_3$ and P9$_4$ direction register (bits 0 and 4 at address 03C2$_{16}$, bit 3 at address 03C3$_{16}$, bits 3 and 4 at address 03C7$_{16}$) = "0" |
| | $\overline{\text{RTS}}$ output (Note 1) | $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit (bit 4 at addresses 036C$_{16}$, 02EC$_{16}$, 033C$_{16}$, 032C$_{16}$, 02FC$_{16}$) = "0"<br>$\overline{\text{CTS}}$/$\overline{\text{RTS}}$ function select bit (bit 2 at addresses 036C$_{16}$, 02EC$_{16}$, 033C$_{16}$, 032C$_{16}$, 02FC$_{16}$) = "1" |
| | Programmable I/O port (Note 2) | $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit (bit 4 at addresses 036C$_{16}$, 02EC$_{16}$, 033C$_{16}$, 032C$_{16}$, 02FC$_{16}$) = "1" |

Note 1: Select TxD output, CLK output and $\overline{\text{RTS}}$ output by the corresponding function select registers A, B and C.
Note 2: Select I/O port by the corresponding function select register A.

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

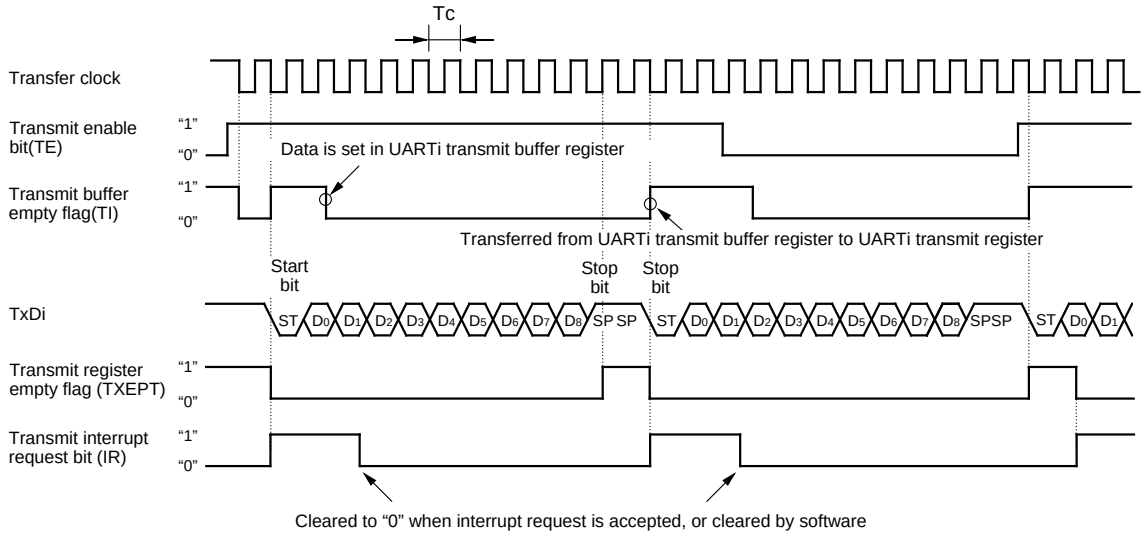



Shown in ( ) are bit symbols.

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 (m + 1) / fi or 16 (m + 1) / fEXT
fi : frequency of BRGi count source (f1, f8, f2n)
fEXT : frequency of BRGi count source (external clock)
m : 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 (m + 1) / fi or 16 (m + 1) / fEXT
fi : frequency of BRGi count source (f1, f8, f2n)
fEXT : frequency of BRGi count source (external clock)
m : value set to BRGi

**Figure 1.19.1. Typical transmit timings in UART mode**

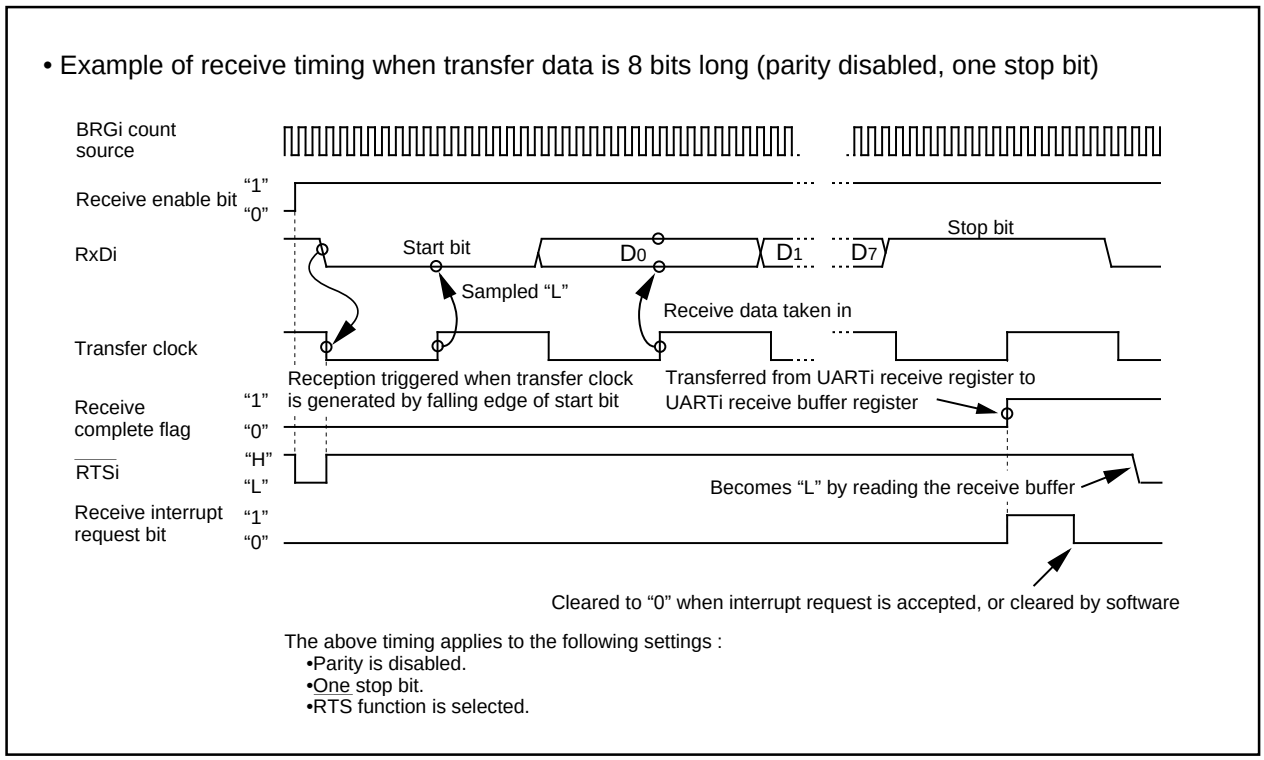



**Figure 1.19.2. Typical receive timing in UART mode**

**(a) Function for switching serial data logic**

When the data logic select bit (bit 6 of address 036D16, 02ED16, 033D16, 032D16, 02FD16) is assigned 1, data is inverted in writing to the transmission buffer register or reading the reception buffer register. Figure 1.19.3 shows the example of timing for switching serial data logic.

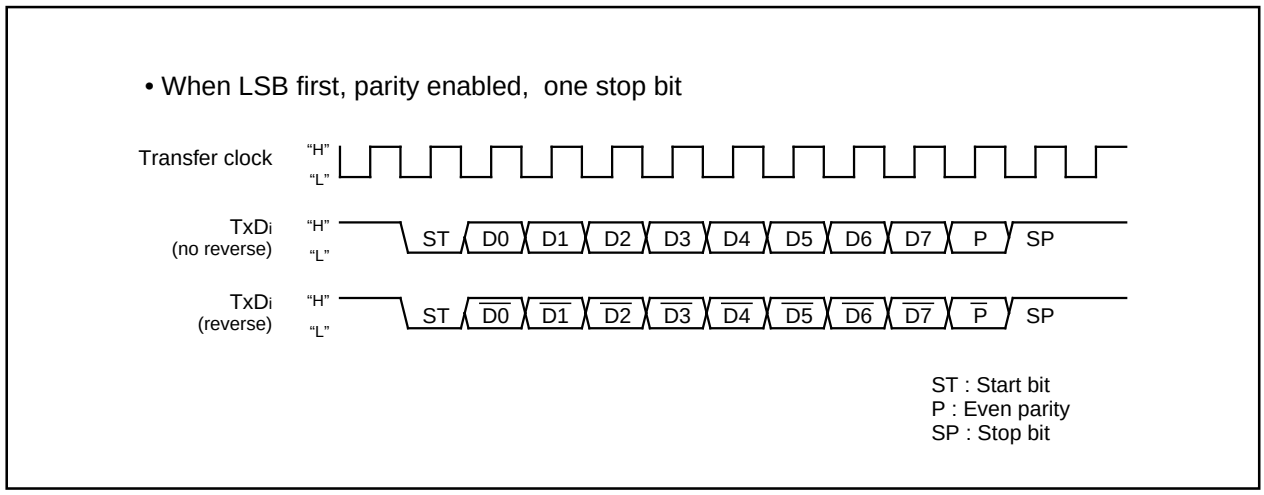



**Figure 1.19.3. Timing for switching serial data logic**

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

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 normal use.

**(c) Bus collision detection function**

This function is to sample the output level of the TxD pin and the input level of the RxD pin at the rising edge of the transfer clock; if their values are different, then an interrupt request occurs. Figure 1.19.4 shows the example of detection timing of a bus collision (in UART mode).

UART0 and UART3 are allocated to software interrupt number 40. UART1 and UART4 are allocated to software interrupt number 41. When selecting UART 0, 3, 1 or 4 bus collision detect function, bit 6 or 7 of external interrupt cause select register (address 031F16) must be set.

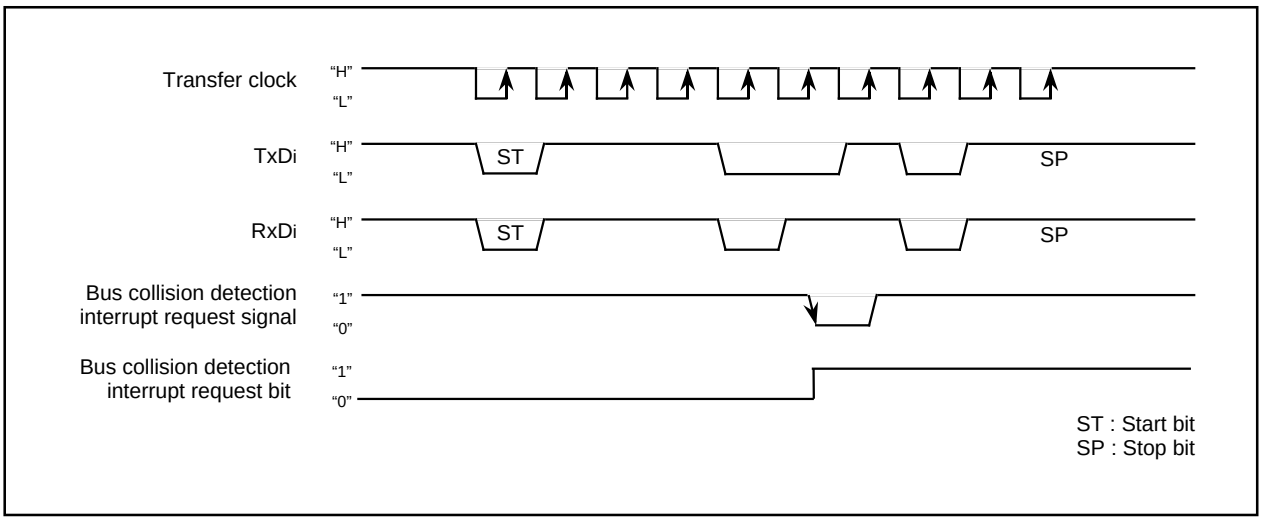



**Figure 1.19.4. Detection timing of a bus collision (in UART mode)**

## UARTi Special Mode Register

UARTi (i=0 to 4) operate the IIC bus interface (simple IIC bus) using the UARTi special mode register (addresses 036716, 02E716, 033716, 032716 and 02F716) and UARTi special mode register 2 (addresses 036616, 02E616, 033616, 032616 and 02F616). UARTi add special functions using UARTi special mode resister 3 (addresses 036516, 02E516, 035516, 032516 and 02F516).

## (1) IIC Bus Interface Mode

The $I^2C$ bus interface mode is provided with UARTi.

Table 1.21.1 shows the construction of the UARTi special mode register and UARTi special mode register 2.

When the $I^2C$ mode select bit (bit 0 in addresses 036716, 02E716, 033716, 032716 and 02F716) is set to "1", the $I^2C$ bus (simple $I^2C$ bus) interface circuit is enabled.

To use the $I^2C$ bus, set the SCLi and the SDAi of both master and slave to output with the function select register. Also, set the data output select bit (bit 5 in address 036C16, 02EC16, 033C16, 032C16 and 02FC16) to N-channel open drain output.

Table 1.21.1 shows the relationship of the IIC mode select bit to control. To use the chip in the clock synchronized serial I/O mode or UART mode, always set this bit to "0".

**Table 1.21.1. Features in $I^2C$ mode**

| | Function | Normal mode (IICM=0) | $I^2C$ mode (IICM=1) (Note 1) |
|---|---|---|---|
| 1 | Factor of interrupt number 39 to 41 (Note 2) | Bus collision detection | Start condition detection or stop condition detection |
| 2 | Factor of interrupt number 17, 19, 33, 35, 37 (Note 2) | UARTi transmission | No acknowledgment detection (NACK) |
| 3 | Factor of interrupt number 18, 20, 34, 36, 38 (Note 2) | UARTi reception | Acknowledgment detection (ACK) |
| 4 | UARTi transmission output delay | Not delayed | Delayed |
| 5 | P63, P67, P70, P92, P96 at the time when UARTi is in use | TxDi (output) | SDAi (input/output) |
| 6 | P62, P66, P71, P91, P97 at the time when UARTi is in use | RxDi (input) | SCLi (input/output) |
| 7 | P61, P65, P72, P90, P95 at the time when UARTi is in use | CLKi | P61, P65, P72, P90, P95 (Note 3) |
| 8 | DMA factor at the time | UARTi reception | Acknowledgment detection (ACK) |
| 9 | Noise filter width | 15ns | 50ns |
| 10 | Reading P62, P66, P71, P91, P97 | 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 UARTi output | H level (when 0 is assigned to the CLK polarity select bit) | The value set in latch P63, P67, P70, P92, P96 when the port is selected (Note 3) |

Note 1: Make the settings given below when $I^2C$ mode is used.
Set 0 1 0 in bits 2, 1, 0 of the UARTi transmission/reception mode register.
Disable the $\overline{RTS}/\overline{CTS}$ function. Choose the MSB First function.

Note 2: Follow the steps given below to switch from one 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 IIC mode (IIC mode select bit = "1") is valid and serial I/O is invalid.

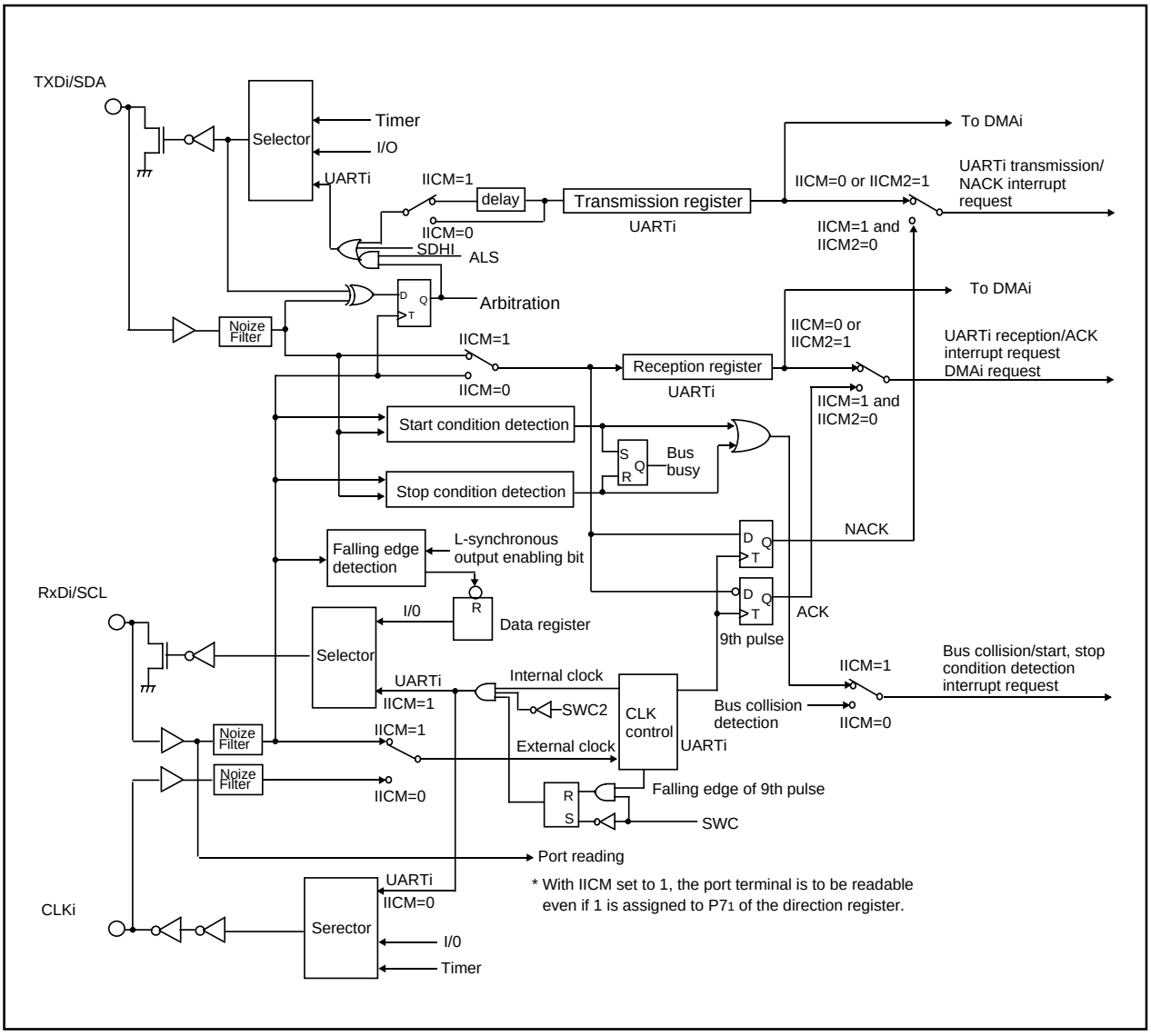



**Figure 1.21.1. Functional block diagram for I²C mode**

Figure 1.21.1 is a block diagram of the IIC bus interface.
The control bits of the IIC bus interface is explained as follow:

**UARTi Special Mode Register (UiSMR:Addresses 0367₁₆, 02E7₁₆, 0337₁₆, 0327₁₆, 02F7₁₆)**

Bit 0 is the IIC mode select bit. When set to “1”, ports operate respectively as the SDAi data transmission-reception pin, SCLi clock I/O pin and port. A delay circuit is added to SDAi transmission output, therefore after SCLi is sufficiently L level, SDAi output changes. Port (SCLi) is designed to read pin level regardless of the content of the port direction register. SDAi transmission output is initially set to port in this mode. Furthermore, interrupt factors for the bus collision detection interrupt, UARTi transmission interrupt and UARTi reception interrupt change respectively to the start/stop condition detection interrupts, acknowledge non-detection interrupt and acknowledge detection interrupt.

The start condition detection interrupt is generated when the falling edge at the SDAi pin is detected while the SCLi pin is in the H state. The stop condition detection interrupt is generated when the rising edge at the SDAi pin is detected while the SCLi pin is in the H state.
The acknowledge non-detection interrupt is generated when the H level at the SDAi pin is detected at the 9th rise of the transmission clock.
The acknowledge detection interrupt is generated when the L level at the SDAi pin is detected at the 9th rise of the transmission clock. Also, DMA transfer can be started when the acknowledge is detected and UARTi transmission is selected as the DMAi request factor.

Bit 1 is the arbitration lost detection flag control bit (ABC). Arbitration detects a conflict between data transmitted at SCLi rise and data at the SDAi pin. This detection flag is allocated to bit 11 in UARTi transmission buffer register (addresses $036F_{16}$, $02EF_{16}$, $033F_{16}$, $032F_{16}$, $02FF_{16}$). It is set to “1” when a conflict is detected. With the arbitration lost detection flag control bit, it can be selected to update the flag in units of bits or bytes. When this bit is set to “1”, update is set to units of byte. If a conflict is then detected, the arbitration lost detection flag control bit will be set to “1” at the 9th rise of the clock. When updating in units of byte, always clear (“0” interrupt) the arbitration lost detection flag control bit after the 1st byte has been acknowledged but before the next byte starts transmitting.

Bit 2 is the bus busy flag (BBS). It is set to “1” when the start condition is detected, and reset to “0” when the stop condition is detected.

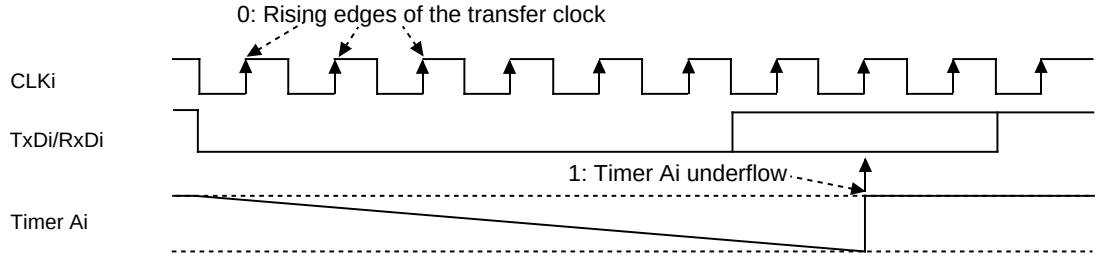
Bit 3 is the SCLi L synchronization output enable bit (LSYN). When this bit is set to “1”, the port data register is set to “0” in sync with the L level at the SCLi pin.

Bit 4 is the bus collision detection sampling clock select bit (ABSCS). The bus collision detection interrupt is generated when RxDi and TxDi level do not conflict with one another. When this bit is “0”, a conflict is detected in sync with the rise of the transfer clock. When this bit is “1”, detection is made when timer Ai (timer A3 with UART0, timer A4 with UART1, timer A0 with UART2, timer A3 with UART3 and timer A4 with UART4) underflows. Operation is shown in Figure 1.21.2.

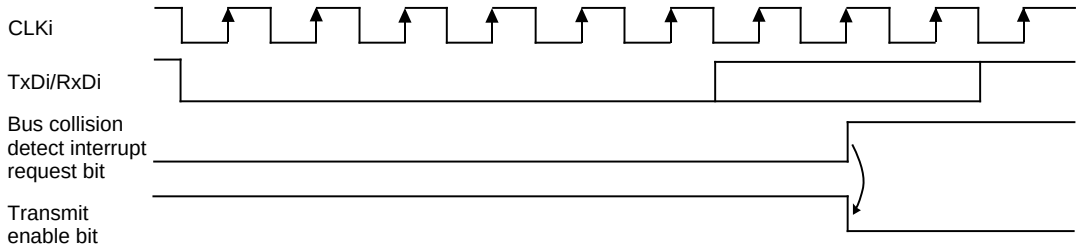
Bit 5 is the transmission enable bit automatic clear select bit (ACSE). By setting this bit to “1”, the transmission bit is automatically reset to “0” when the bus collision detection interrupt factor bit is “1” (when a conflict is detected).

Bit 6 is the transmission start condition select bit (SSS). By setting this bit to “1”, TxDi transmission starts in sync with the rise at the RxDi pin.

**1. Bus collision detect sampling clock select bit (Bit 4 of the UARTi special mode register)**

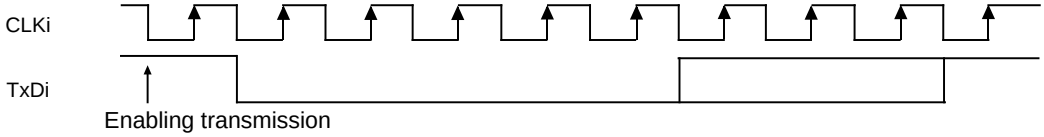



**2. Auto clear function select bit of transmit enable bit (Bit 5 of the UARTi special mode register)**




**3. Transmit start condition select bit (Bit 6 of the UARTi special mode register)**

**0: In normal state**




**With "1: falling edge of RxDi" selected**

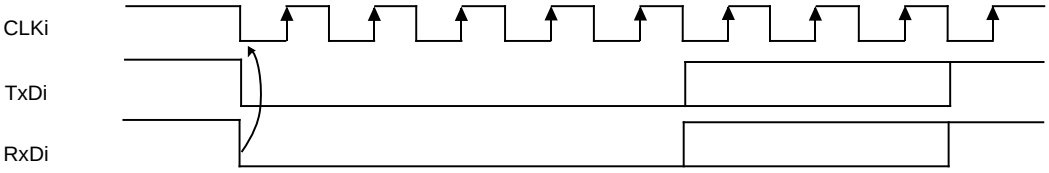



**Figure 1.21.2. Some other functions added**

**UARTi Special Mode Register 2 (UiSMR2:Addresses 0366$_{16}$, 02E6$_{16}$, 0336$_{16}$, 0326$_{16}$, 02F6$_{16}$)**

Bit 0 is the IIC mode select bit 2 (IICM2). Table 1.21.2 gives control changes by bit when the IIC mode select bit is “1”. Start and stop condition detection timing characteristics are shown in Figure 1.21.4. Always set bit 7 (start/stop condition control bit) to “1”.

Bit 1 is the clock synchronizing bit (CSC). When this bit is set to “1”, and the rising edge is detected at pin SCLi while the internal SCL is High level, the internal SCL is changed to Low level, the baud rate generator value is reloaded and the Low sector count starts. Also, while the SCLi pin is Low level, and the internal SCL changes from Low level to High, baud rate generator stops counting. If the SCLi pin is H level, counting restarts. Because of this function, the UARTi transmission-reception clock takes the AND condition for the internal SCL and SCLi pin signals. This function operates from the clock half period before the 1st rise of the UARTi clock to the 9th rise. To use this function, select the internal clock as the transfer clock.

Bit 2 is the SCL wait output bit (SWC). When this bit is set to “1”, output from the SCLi pin is fixed to L level at the clock’s 9th rise. When set to “0”, the Low output lock is released.

Bit 3 is the SDA output stop bit (ALS). When this bit is set to “1”, an arbitration lost is generated. If the arbitration lost detection flag is “1”, then the SDAi pin simultaneously becomes high impedance.

Bit 4 is the UARTi initialize bit (STC). While this bit is set to “1”, the following operations are performed when the start condition is detected.

1. The transmission shift register is initialized and the content of the transmission register is transmitted to the transmission shift register. As such, transmission starts with the 1st bit of the next input clock. However, the UARTi output value remains the same as when the start condition was detected, without changing from when the clock is input to when the 1st bit of data is output.
2. The reception shift register is initialized and reception starts with the 1st bit of the next input clock.
3. The SCL wait output bit is set to “1”. As such, the SCLi pin becomes Low level at the rise of the 9th bit of the clock.

When UART transmission-reception has started using this function, the content of the transmission buffer available flag does not change. Also, to use this function, select an external clock as the transfer clock.

Bit 5 is SCL wait output bit 2 (SWC2). When this bit is set to “1” and serial I/O is selected, an Low level can be forcefully output from the SCLi pin even during UART operation. When this bit is set to “0', the Low output from the SCLi pin is canceled and the UARTi clock is input and output.

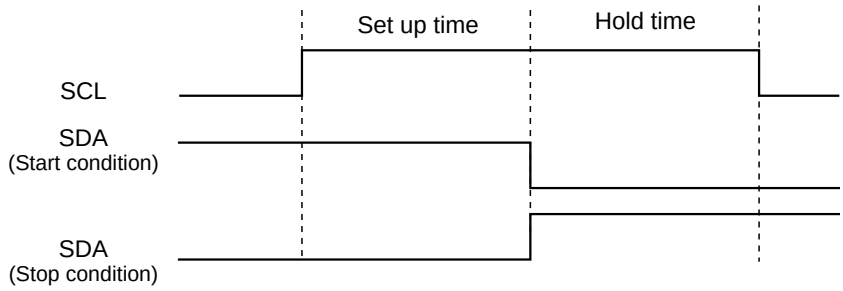
Bit 6 is the SDA output disable bit (SDHI). When this bit is set to “1”, the SDAi pin is forced to high impedance. To overwrite this bit, do so at the rise of the UARTi transfer clock. The arbitration lost detection flag may be set.

**Table 1.21.2. Functions changed by I²C mode select bit 2**

| Function | IICM2 = 0 | IICM2 = 1 |
|---|---|---|
| Interrupt no. 17, 19, 33, 35, 37 factor | Acknowledge not detect (NACK) | UARTi transfer (rising edge of the last bit) |
| Interrupt no. 18, 20, 34, 36, 38 factor | Acknowledge detect (ACK) | UARTi receive (falling edge of the last bit) |
| DMA factor | Acknowledge detect (ACK) | UARTi receive (falling edge of the last bit) |
| Data transfer timing from UART receive shift register to receive buffer | Rising edge of the last bit of receive clock | Rising edge of the last bit of receive clock |
| UART receive / ACK interrupt request generation timing | Rising edge of the last bit of receive clock | Rising edge of the last bit of receive clock |

3 to 6 cycles < set up time (Note)

3 to 6 cycles < hold time (Note)




Note : Cycle number shows main clock input oscillation frequency f(XIN) cycle number.

**Figure 1.21.3. Start/stop condition detect timing characteristics**

**UARTi Special Mode Register 3 (UiSMR3:Addresses 0365$_{16}$, 02E5$_{16}$, 0335$_{16}$, 0325$_{16}$, 02F5$_{16}$)**

Bit 1 is clock phase set bit (CKPH). When both the IIC mode select bit (bit 0 of UARTi special mode select register) and the IIC mode select bit 2 (bit 0 of UiSMR2 register) are "1", functions changed by these bits are shown in table 1.21.3 and figure 1.21.4.

Bits 5 to 7 are SDAi digital delay setting bits (DL0 to DL2). By setting these bits, it is possible to turn the SDAi delay OFF or set the BRG count source delay to 2 to 8 cycles.

**Table 1.21.3. Functions changed by clock phase set bits**

| Function | CKPH = 0, IICM = 1, IICM2 = 1 | CKPH = 1, IICM = 1, IICM2 = 1 |
|---|---|---|
| SCL initial and last value | Initial value = H, last value = L | Initial value = L, last value = L |
| Transfer interrupt factor | Rising edge of 9th bit | Falling edge of 10th bit |
| Data transfer times from UART receive shift register to receive buffer register | Falling edge of 9th bit | Two times :falling edge of 9th bit and rising edge of 9th bit |

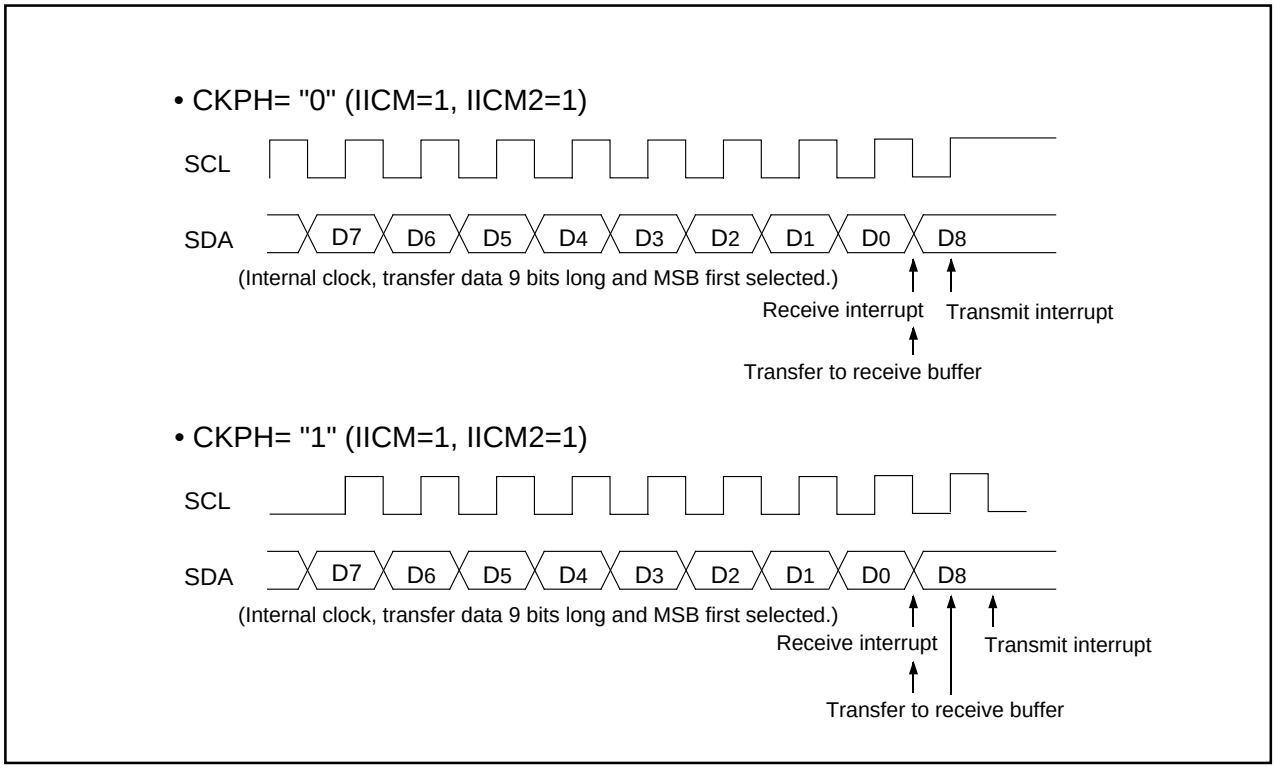



**Figure 1.21.4. Functions changed by clock phase set bits**

**UARTi Special Mode Register 4 (UiSMR4:Addresses 0364$_{16}$, 02E4$_{16}$, 0334$_{16}$, 0324$_{16}$, 02F4$_{16}$)**

Bit 0 is the start condition generate bit (STAREQ). When the SCL, SDA output select bit (bit 3 of UiSMR4 register) is "1" and this bit is "1", then the start condition is generated.

Bit 1 is the restart condition generate bit (RSTAREQ). When the SCL, SDA output select bit (bit 3 of UiSMR4 register) is "1" and this bit is "1", then the restart condition is generated.

Bit 2 is the stop condition generate bit (STPREQ). When the SCL, SDA output select bit (bit 3 of UiSMR4 register) is "1" and this bit is "1", then the stop condition is generated.

Bit 3 is SCL, SDA output select bit (STSPSEL). Functions changed by these bits are shown in table 1.21.4 and figure 1.21.5.

**Table 1.21.4. Functions changed by SCL, SDA output select bit**

| Function | STSPSEL = 0 | STSPSEL = 1 |
|---|---|---|
| SCL, SDA output | Output of SI/O control circuit | Output of start/stop condition control circuit |
| Start/stop condition interrupt factor | Start/stop condition detection | Completion of start/stop condition generation |

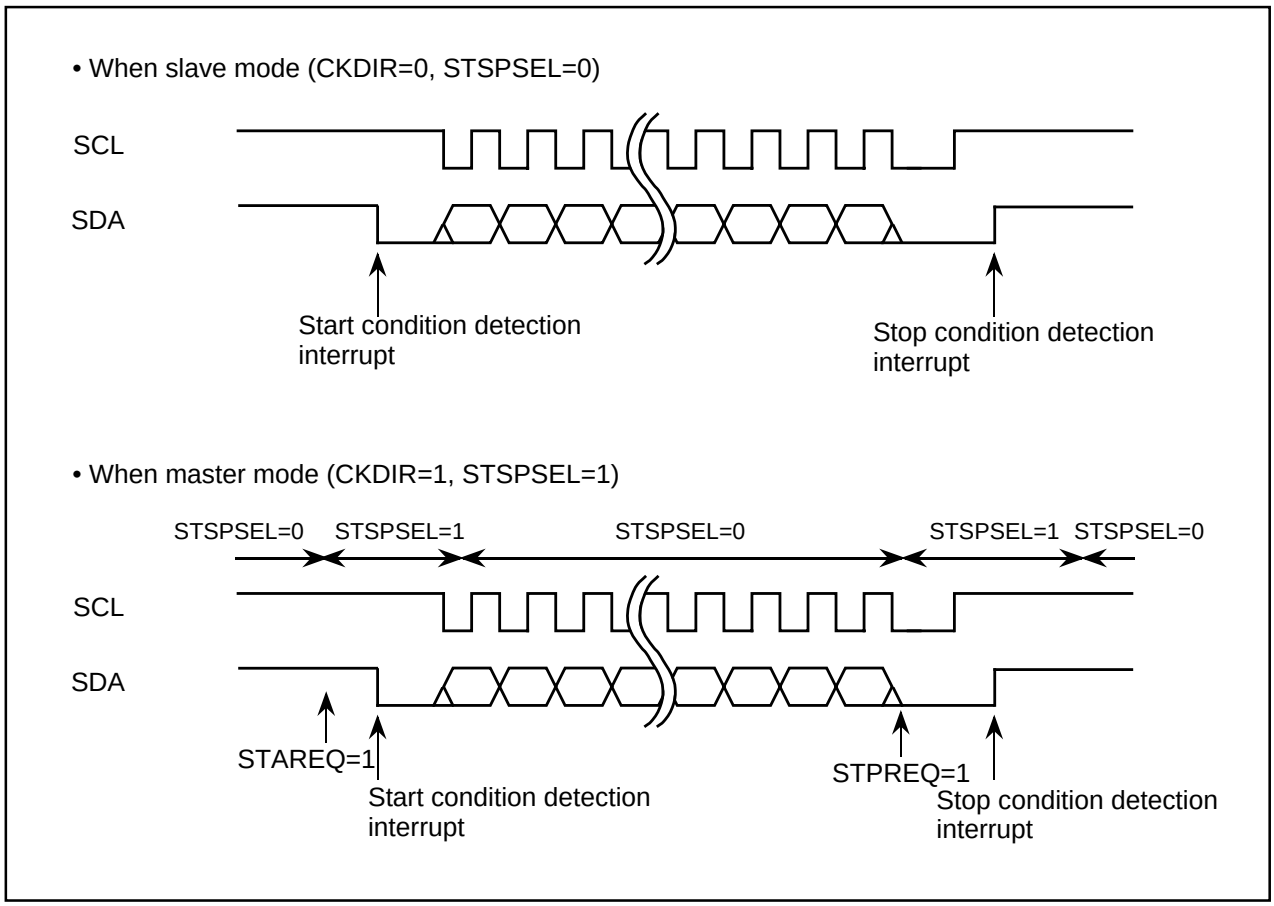



**Figure 1.21.5 Functions changed by SCL, SDA output select bit**

Bit 4 is ACK data bit (ACKD). When the SCL, SDA output select bit (bit 3 of UiSMR4 register) is "0" and the ACK data output enable bit (bit 5 of UiSMR4 register) is "1", then the content of ACK data bit is output to SDAi pin.

Bit 5 is ACK data output enable bit (ACKC). When the SCL, SDA output select bit (bit 3 of UiSMR4 register) is "0" and this bit is "1", then the content of ACK data bit is output to SDAi pin.

Bit 6 is SCL output stop bit (SCLHI). When this bit is "1", SCLi output is stopped at stop condition detection. (Hi-impedance status).

Bit 7 is SCL wait output bit 3 (SWC9). When this bit is "1", SCLi output is fixed to "L" at falling edge of 10th bit of clock. When this bit is "0", SCLi output fixed to "L" is released.

## (2) Serial Interface Special Function

UARTi can control communications on the serial bus using the $\overline{SSi}$ input pins (Figure 1.21.6). The master outputting the transfer clock transfers data to the slave inputting the transfer clock. In this case, in order to prevent a data collision on the bus, the master floats the output pin of other slaves/masters using the $\overline{SSi}$ input pins.

$\overline{SSi}$ input pins function between the master and slave are as follows.

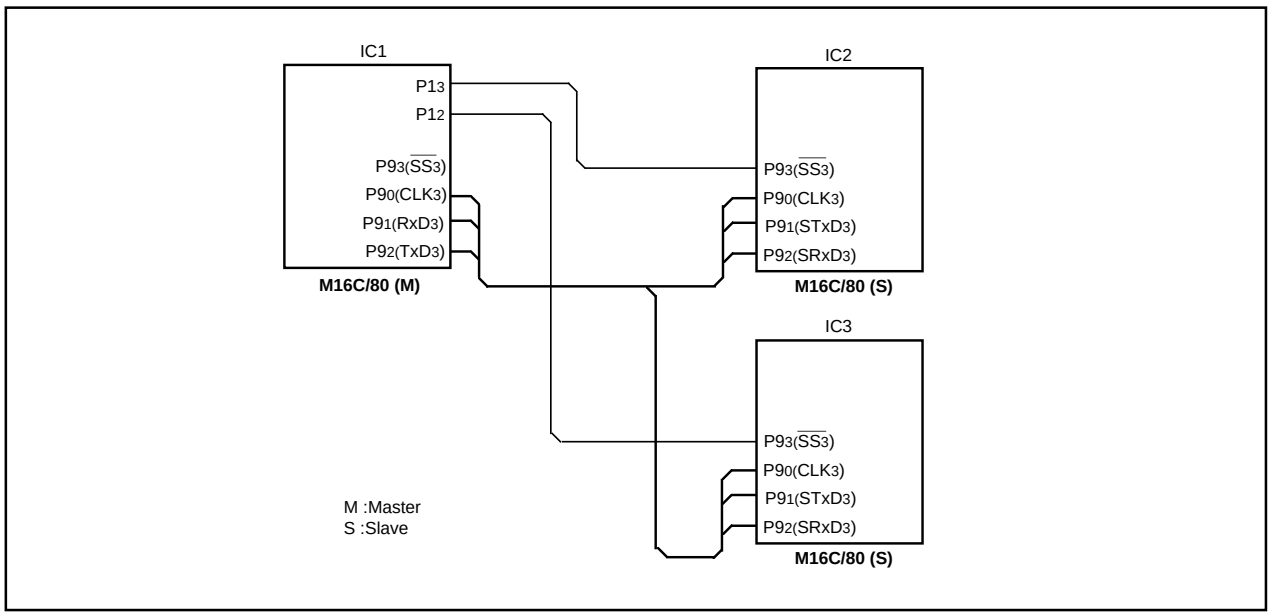



**Figure 1.21.6. Serial bus communication control example using the SS input pins**

**< Slave Mode (STxDi and SRxDi are selected, DINC = 1) >**

When an H level signal is input to an $\overline{SSi}$ input pin, the STxDi and SRxDi pins both become high impedance, hence the clock input is ignored. When an "L" level signal is input to an $\overline{SSi}$ input pin, the clock input becomes effective and serial communications are enabled.

**< Master Mode (TxDi and RxDi are selected, DINC = 0) >**

The $\overline{SSi}$ input pins are used with a multiple master system. When an $\overline{SSi}$ input pin is H level, transmission has priority and serial communications are enabled. When an L signal is input to an $\overline{SSi}$ input pin, another master exists, and the TxDi, RxDi and CLKi pins all become high impedance. Moreover, the trouble error interrupt request bit becomes “1”. Communications do not stop even when a trouble error is generated during communications. To stop communications, set bits 0, 1 and 2 of the UARTi transmission-reception mode register (addresses $0368_{16}$, $02E8_{16}$, $0338_{16}$, $0328_{16}$ and $02F8_{16}$) to “0”.

### Clock Phase Setting

With bit 1 of UARTi special mode register 3 (UiSMR3:addresses 0365$_{16}$, 02E5$_{16}$, 0335$_{16}$, 0325$_{16}$, 02F5$_{16}$) and bit 6 of UARTi transmission-reception control register 0 (addresses 036C$_{16}$, 02EC$_{16}$, 033C$_{16}$, 032C$_{16}$, 02FC$_{16}$), four combinations of transfer clock phase and polarity can be selected.
Bit 6 of UARTi transmission-reception control register 0 sets transfer clock polarity, whereas bit 1 of UiSMR3 register sets transfer clock phase.
Transfer clock phase and polarity must be the same between the master and slave involved in the transfer.

**< Master (Internal Clock) (DINC = 0) >**

Figure 1.21.7 shows the transmission and reception timing.

**< Slave (External Clock) (DINC = 1) >**

- With “0” for CKPH bit (bit 1 of UiSMR3 register), when an $\overline{SSi}$ input pin is H level, output data is high impedance. When an $\overline{SSi}$ input pin is L level, the serial transmission start condition is satisfied, though output is indeterminate. After that, serial transmission is synchronized with the clock. Figure 1.21.8 shows the timing.
- With “1” for CKPH bit, when an $\overline{SSi}$ input pin is H level, output data is high impedance. When an $\overline{SSi}$ input pin is L level, the first data is output. After that, serial transmission is synchronized with the clock.Figure 1.21.9 shows the thing.

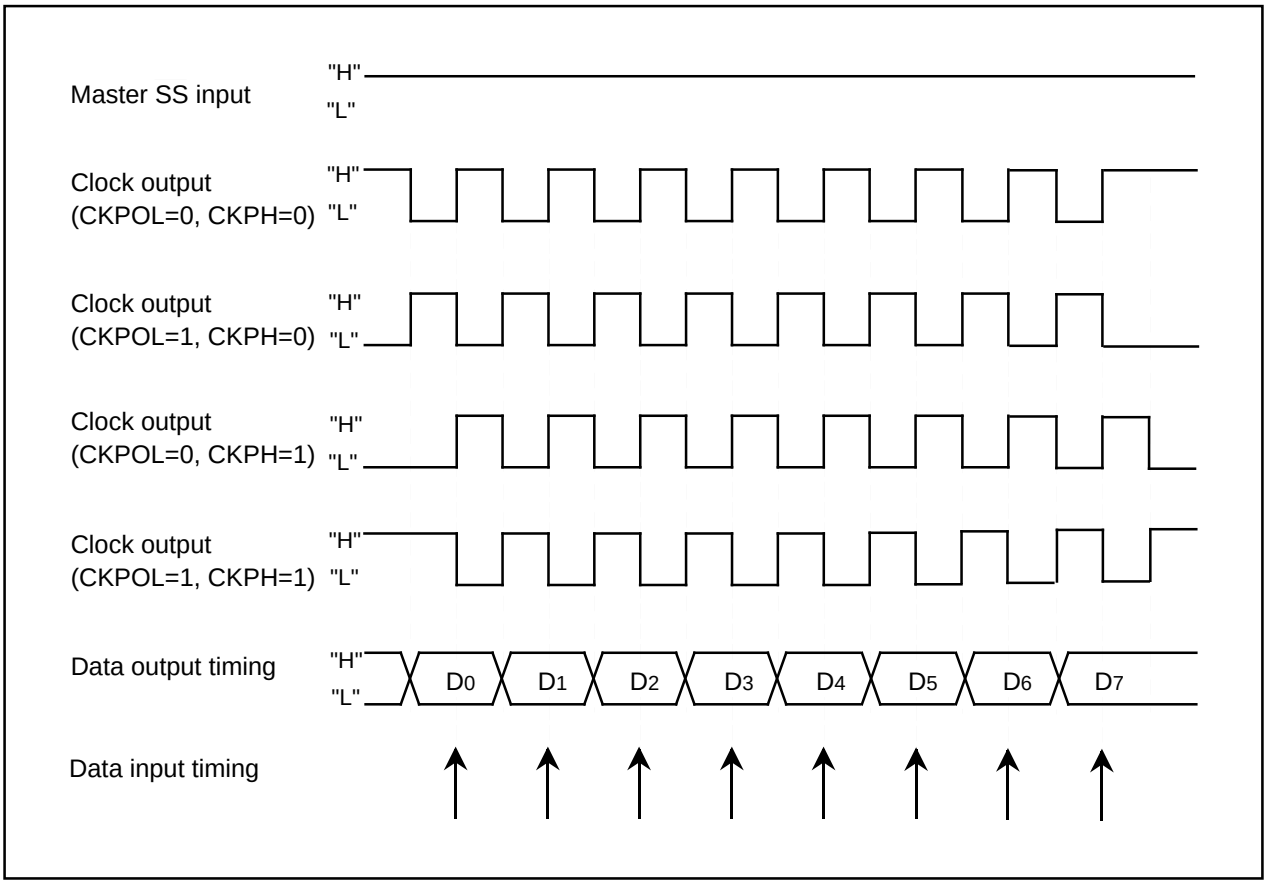



**Figure 1.21.7. The transmission and reception timing in master mode (internal clock)**

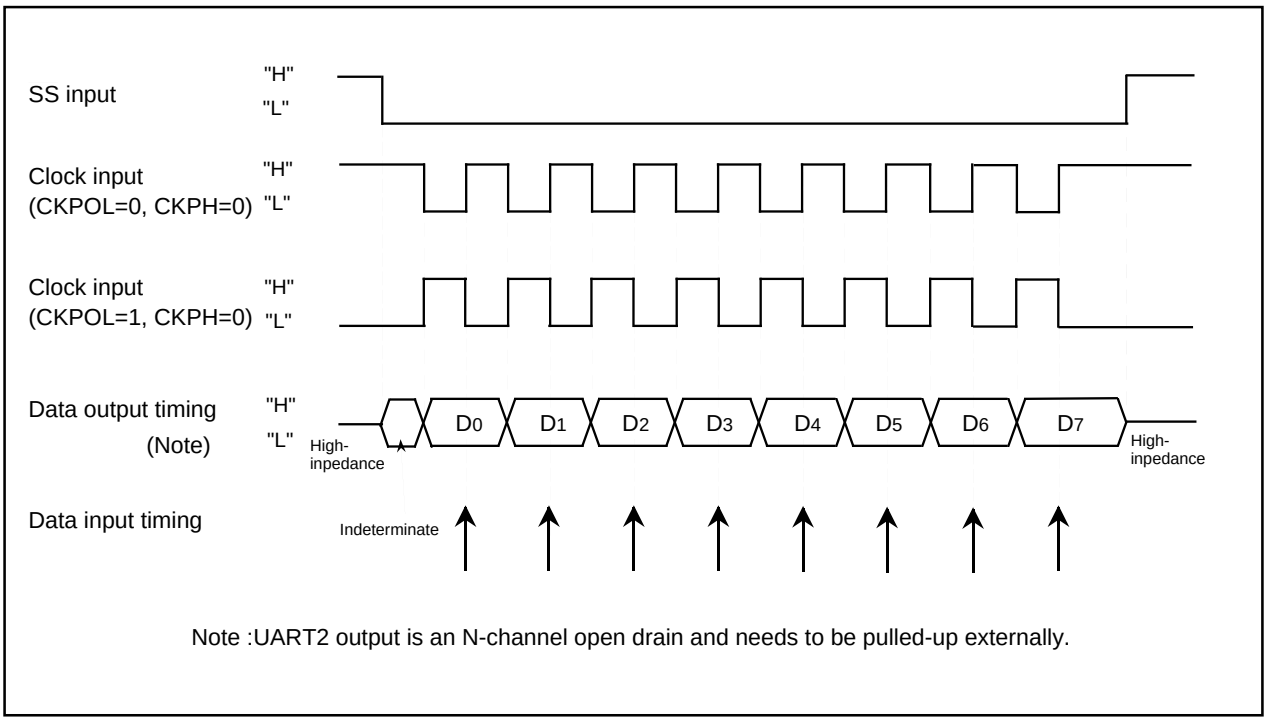



**Figure 1.21.8. The transmission and reception timing (CKPH=0) in slave mode (external clock)**

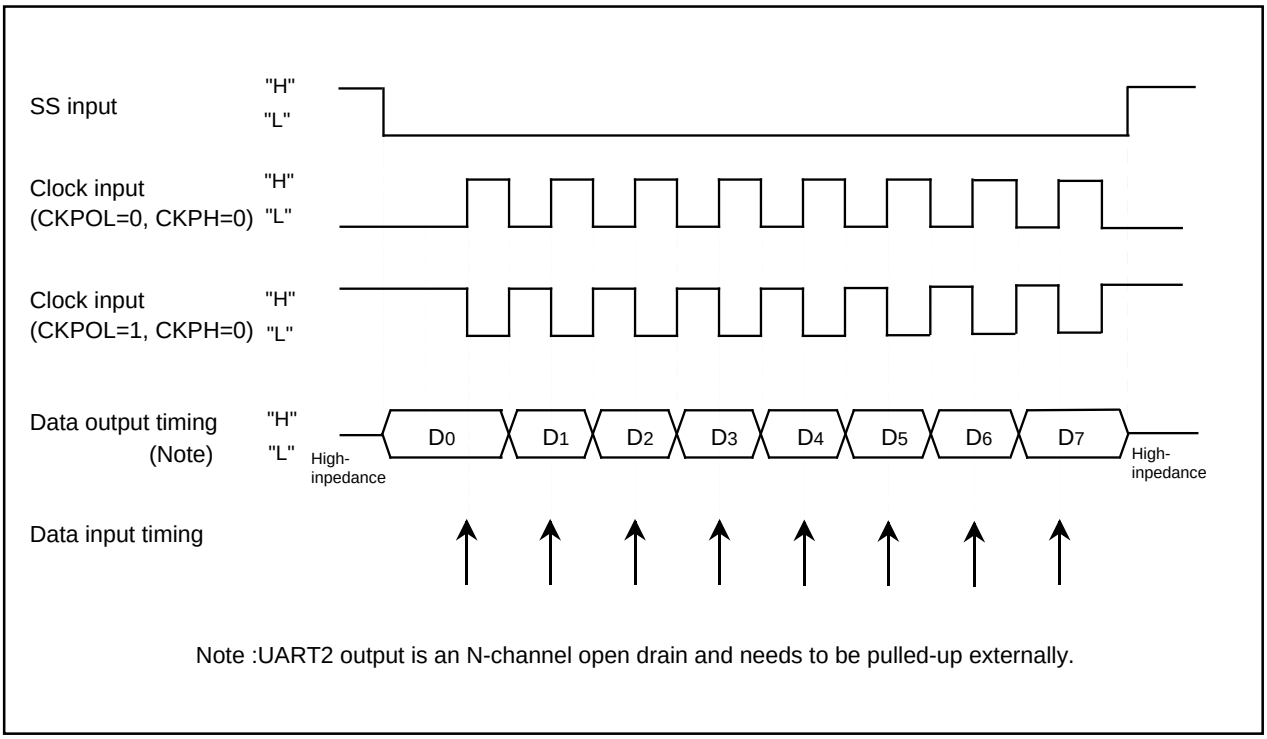



**Figure 1.21.9. The transmission and reception timing (CKPH=1) in slave mode (external clock)**

## CAN Module

The microcomputer incorporates Full-CAN modules compliant with CAN (Controller Area Network) 2.0B specification.

These Full-CAN modules are outlined below.

**Table 1.22.1 Outline of the CAN module**

| Item | Description |
|---|---|
| Protocol | Compliant with CAN 2.0B specification |
| Number of message slots | 16 slots |
| Polarity | 0: Dominant<br>1: Recessive |
| Acceptance filter | Global mask: 1 mask (for message slots 0–13)<br>Local mask: 2 masks (for message slots 14 and 15 each) |
| Baud rate | 1 time quantum (Tq) = (BRP + 1) / CPU clock (Note)<br>(BRP = baud rate prescaler set value)<br>Baud rate = 1 / (Tq period x number of Tq's in one bit) ---Max. 1 Mbps<br>BRP: 1-255 (0: Inhibited)<br>Number of Tq's in one bit = Synchronization Segment + Propagation Time Segment + Phase Buffer Segment 1 + Phase Buffer Segment 2<br>Synchronization Segment : 1 Tq (fixed)<br>Propagation Time Segment : 1 to 8 Tq<br>Phase Buffer Segment 1 : 2 to 8 Tq<br>Phase Buffer Segment 2 : 2 to 8 Tq |
| Remote frame automatic answering function | The message slot that received a remote frame automatically transmits it. |
| Timestamp function | This timestamp function is based on a 16-bit counter. A count period can be derived from the CAN bus bit period (as the fundamental period) by dividing it by 1, 2, 3, or 4. |
| BasicCAN mode | The BasicCAN function is realized by using message slots 14 and 15. |
| Transmit abort function | This function is used to cancel a transmit request. |
| Loopback function | The data the CAN module itself transmitted is received. |
| Return from bus-off function | Forcibly placed into an error active state from a bus-off state. |

Note: Use a specification conforming resonator whose maximum permissible error of oscillation is not greater than 1.58%

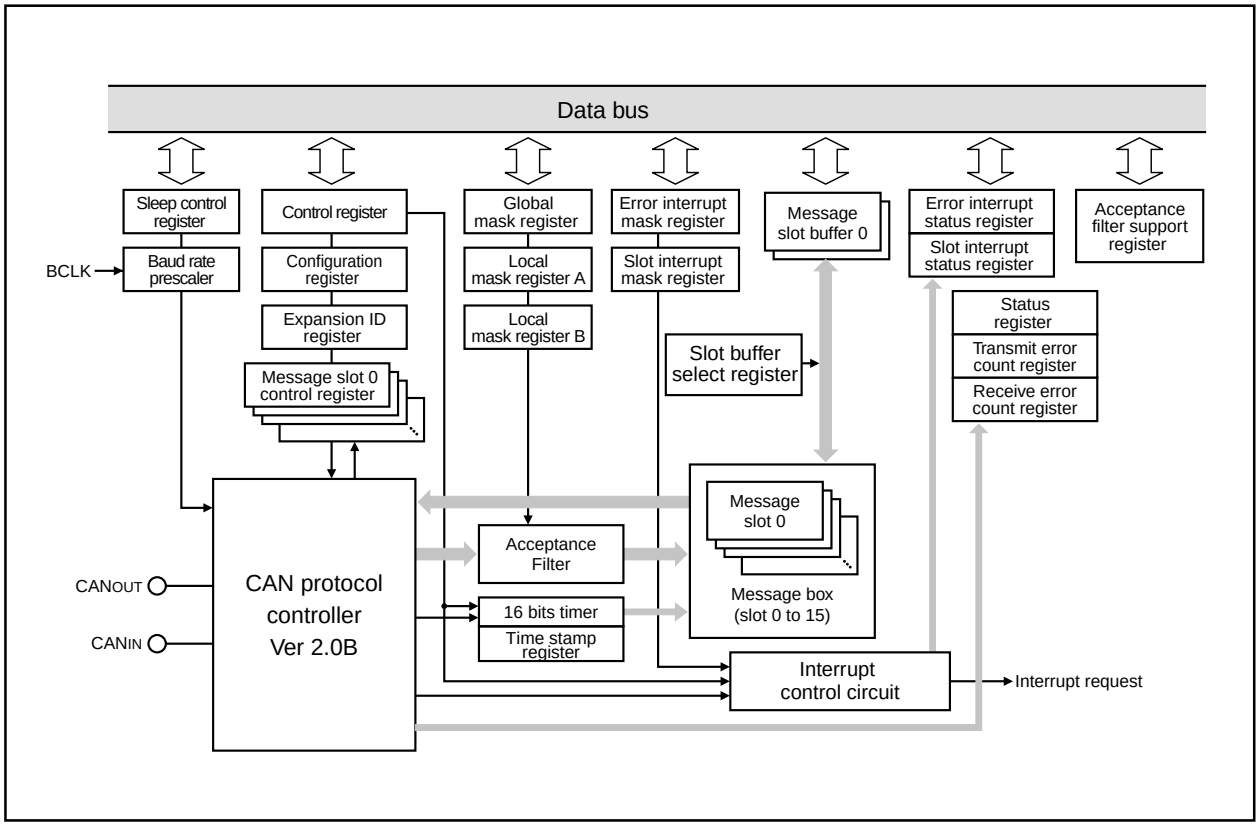



**Figure 1.22.1 CAN module blobk diagram**

CAN0 message slot buffer 0 and 1 can be selected by setting of slot buffer select register. Figure 1.22.2 shows the message slot buffer and 16 bytes of message slots. Figure 1.22.26 to 1.22.30 show related registers.

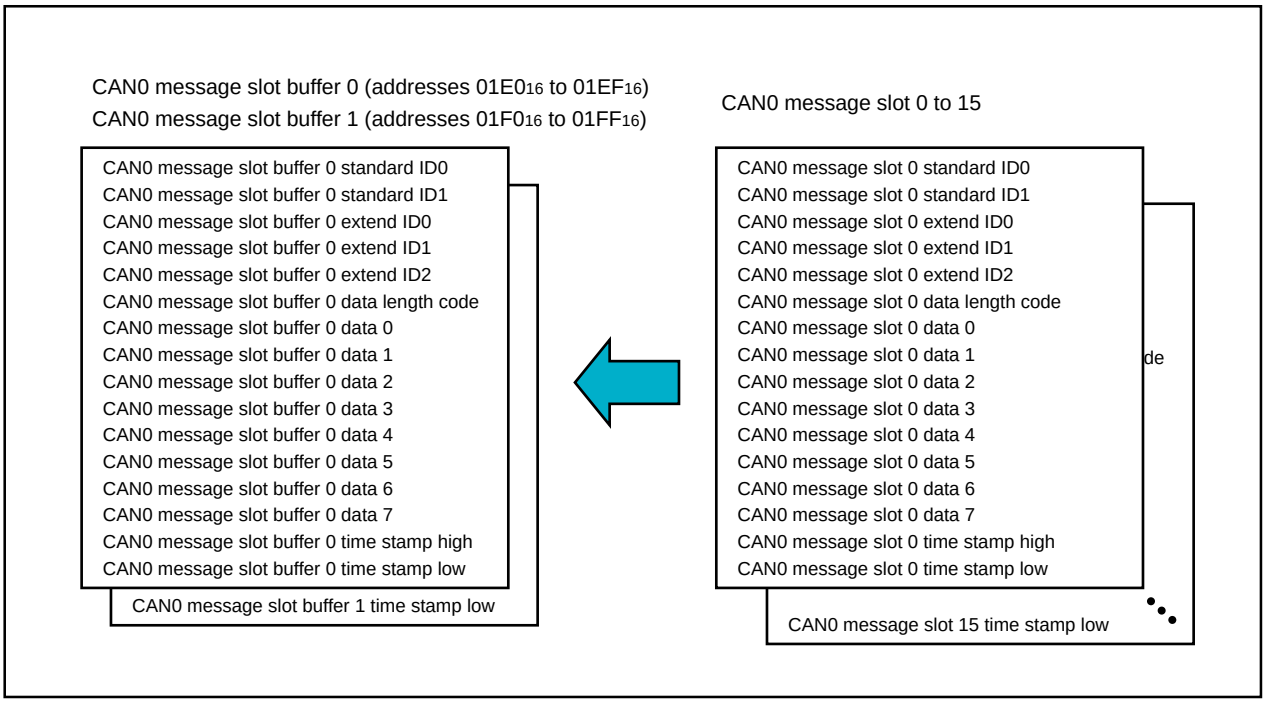
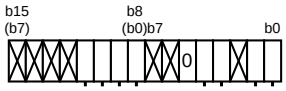



**Figure 1.22.2. Message slot buffer and message slots**

## CAN0 control register 0




| Symbol | Address | When reset (Note 1) |
|---|---|---|
| C0CTLR0 | $0201_{16}$, $0200_{16}$ | XXXX 0000 XX01 0X01$_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Reset 0 | CAN reset bit 0 | 0: Reset released<br>1: Reset requested | ○ | ○ |
| Loopback | Loop back mode select bit | 0: Loop back function disabled<br>1: Loop back function enabled | ○ | ○ |
| —— | Nothing is assigned. When write, set to "0". When read, its contents is indeterminate. | | — | — |
| BasicCAN | Basic CAN mode select bit | 0 : Basic CAN mode function disabled<br>1 : Basic CAN mode function enabled | ○ | ○ |
| Reset 1 | CAN reset bit 1 | 0 : Reset released<br>1 : Reset requested | ○ | ○ |
| —— | Reserved bit | Must set to "0". | ○ | ○ |
| —— | Nothing is assigned. When write, set to "0". When read, their contents are indeterminate. | | — | — |
| —— | | | — | — |
| TSPre0 | Time stamp prescaler select bit | b9 b8<br>0 0: CAN bus bit clock is selected<br>0 1: Division by 2 of CAN bus bit clock is selected<br>1 0: Division by 3 of CAN bus bit clock is selected<br>1 1: Division by 4 of CAN bus bit clock is selected | ○ | ○ |
| TSPre1 | | | ○ | ○ |
| TSReset | Time stamp counter reset bit | 0 : Count enabled<br>1 : Count reset (set $0000_{16}$) (Note 2) | ○ | ○ |
| ECReset | Error counter reset bit | 0 : Normal operation mode<br>1 : Error counter reset (Note 2) | ○ | ○ |
| —— | Nothing is assigned. When write, set to "0". When read, their contents are indeterminate. | | — | — |
| —— | | | — | — |
| —— | | | — | — |
| —— | | | — | — |

Note 1: This applies when the CAN module is supplied with a clock by setting the sleep mode control bit (bit 0 at address $0242_{16}$) to 1 after reset.
Note 2: Only writing 1 is accepted. The bit is automatically cleared to 0 in hardware.

**Figure 1.22.3 CAN0 control register 0**

## 1. CAN0 control register 0

**Bit 0: CAN reset bits 0 and 1 (Reset0 and Reset1)**

If the Reset0 and Reset1 bits both are set from 1 to 0, CAN communication is enabled after detecting 11 consecutive recessive bits. The CAN Timestamp Register starts counting at the same time communication is enabled.

In no case will the CAN be reset unless transmission of all messages are completed.

Note 1: Reset0 and Reset1 bits must both be cleared to "0" or set to "1" simutnously.

Note 2: Setting a new transmit request is inhibited before the CAN Status Register State_Reset bit is set to 1 and the CAN module is reset after setting the Reset0 and Reset1 bits to 1.

Note 3: When the CAN module is reset by setting the Reset0 and Reset1 bits to 1, the CAN Timestamp Register (C0TSR), CAN Transmit Error Counter (C0TEC), and CAN Receive Error Counter (C0REC) are initialized to 0.

Note 4: If Reset0 and Reset1 bits sre set to "1" during communication, the CANOUT pin output goes "H" immediately after that. Therefore, setting these bits to 1 while the CAN module is sending a frame may cause a CAN bus error.

Note 5: To CAN communication, function select register A1 (PS1), function select register A2 (PS2), function select register B1 (PSL1), function select register B2 (PSL2), function select register C (PSC) and input function select register (IPS) must be set. These registers must be set when CAN module is reset.

**Bit 1: Loopback mode select bit (LoopBack)**

Setting the LoopBack bit to 1 enables loopback mode, so that if any receive slot whose ID matches that of a frame the CAN module itself transmitted exists, the frame is received.

Note 1: ACK is not returned for the transmit frame.

Note 2: Do not set or reset the LoopBack bit while the CAN module is operating (CAN Status Register State_Reset bit = 0).

**Bit 3: BasicCAN mode select bit (BasicCAN)**

If this bit is set to 1, message slots 14 and 15 operate in BasicCAN mode.

**• Operation during BasicCAN mode**

In BasicCAN mode, message slots 14 and 15 are used with a dual-structured buffer. The received frames whose IDs are found matching by acceptance filtering are stored in slots 14 and 15 alternately. When slot 14 is active (i.e., the next received frame is to be stored in slot 14), this acceptance filtering is accomplished using the ID that is set in slot 14 and local mask A; when slot 15 is active, it is accomplished using the ID that is set in slot 15 and local mask B. Frame types of both data frame and remote frame can be received.

When using BasicCAN mode, setting the IDs of two slots and the mask registers the same way helps to reduce the possibility of causing an overrun error.

**• Procedure for entering BasicCAN mode**

Make the following settings during initialization.

(1) Set the BasicCAN bit to 1.

(2) Set the IDs of slots 14 and 15 and Local Mask Registers A and B. (We recommend setting the same value)

(3) Set the frame format to be handled with slots 14 and 15 (standard or extended) in the CAN Extended ID Register. (We recommend setting the same format)

(4) Set the Message Slot Control Registers for slots 14 and 15 to receive data frames.

Note 1: Do not set or reset the BasicCAN bit while the CAN module is operating (CAN Status Register State_Reset bit = 0).

Note 2: Slot 14 is the first slot to become active after clearing the Reset0 bit.

Note 3: Even during BasicCAN mode, slot 0 through slot 13 can be used in the same way as when operating normally.

**Bit 8, 9: Timestamp prescaler select bits (TSPre0, 1)**

These bits select the count clock source for the timestamp counter.

Note 1: Do not set or reset these TSPre0, 1 bits while the CAN module is operating (CAN Status Register State_Reset bit = 0).

**Bit 10: Timestamp counter reset bit (TSReset)**

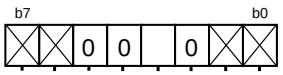
Setting this bit to 1 clears the value of the CAN Timestamp Register (C0TSR) to $0000_{16}$. This bit is automatically cleared after the CAN Timestamp Register (C0TSR) has its value cleared to $0000_{16}$.

**Bit 11: Error counter reset bit (ECReset)**

Setting this bit to 1 clears the Receive Error Counter Register (C0REC) and Transmit Error Counter Register (C0TEC), with the CAN module forcibly placed in an error active state. This bit is automatically cleared upon entering an error active state.

Note 1: When in an error active state, the CAN module becomes ready to communicate when it detects 11 consecutive recessive bits on the CAN bus.

CAN0 control register 1




| Symbol | Address | When reset (Note) |
|---|---|---|
| C0CTLR1 | $0241_{16}$ | $XX0000XX_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| — | Nothing is assigned. When write, set to "0". When read, their contents are indeterminate. | | — | — |
| — | | | — | — |
| — | Reserved bit | Must set to "0". | ○ | ○ |
| BankSel | CAN0 bank select bit | 0 : Message slot control register selected<br>1 : Mask register selected | ○ | ○ |
| — | Reserved bit | Must set to "0". | ○ | ○ |
| — | | | ○ | ○ |
| — | Nothing is assigned. When write, set to "0". When read, their contents are indeterminate. | | — | — |
| — | | | — | — |

Note: This applies when the CAN module is supplied with a clock by setting the sleep mode control bit (bit 0 at address $0242_{16}$) to 1 after reset.

**Figure 1.22.4. CAN0 control register 1**

## 2. CAN0 control register 1

### Bit 3: CAN0 bank select bit (BankSel)

This bit selects between registers allocated to the addresses $0220_{16}$ through $023F_{16}$.

Setting the BankSel bit to 0 selects the CAN0 Message Slot Control Register. Setting the BankSel bit to 1 selects the CAN0 Mask Register.

CAN0 sleep control register

b7 b0

| Symbol | Address | When reset |
|---|---|---|
| C0SLPR | 0242$_{16}$ | XXXXXXX0$_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Sleep | Sleep mode control bit | 0 : Sleep mode On<br>1 : Sleep mode Off (Note) | ○ | ○ |
| — | Nothing is assigned. When write, set to "0".<br>When read, their contents are indeterminate. | | — | — |
| — | | | — | — |
| — | | | — | — |
| — | | | — | — |
| — | | | — | — |
| — | | | — | — |
| — | | | — | — |

Note: After CAN sleep mode is canceled, set up the CAN configuration. While the CAN module is in sleep mode, no SFR registers for the CAN, except the sleep mode control register, can be accessed for read or write.

**Figure 1.22.5. CAN0 sleep control register**

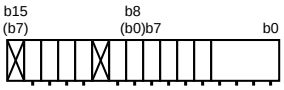
## 3. CAN0 sleep control register

**Bit 0: Sleep mode control bit (Sleep)**

The CAN module isn't supplied with a clock by setting the Sleep bit to 0, and is shifted to sleep mode.
The CAN module is supplied with a clock by setting the Sleep bit to 1, and is released from sleep mode.

Note: Sleep mode can be shifted to only after CAN is reset (State_Reset bit = 1).

CAN0 status register




| Symbol | Address | When reset (Note) |
|---|---|---|
| C0STR | $0203_{16}$,$0202_{16}$ | X000 0X01 0000 $0000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| MBox0 | Active slot determination bit | b3 b2 b1 b0<br>0 0 0 0 : Slot 0<br>0 0 1 0 : Slot 1<br>0 0 1 1 : Slot 2<br>0 1 0 0 : Slot 3<br>• •<br>• •<br>1 1 0 1 : Slot 13<br>1 1 1 0 : Slot 14<br>1 1 1 1 : Slot 15 | ○ | – |
| MBox1 | | | ○ | – |
| MBox2 | | | ○ | – |
| MBox3 | | | ○ | – |
| TrmSucc | Transmission-finished status | 0: Transmission not finished<br>1: Transmission finished | ○ | – |
| RecSucc | Reception-finished status | 0: Reception not finished<br>1: Reception finished | ○ | – |
| TrmState | Transmission status | 0: Not transmitting<br>1: Transmitting | ○ | – |
| RecState | Reception status | 0: Not receiving<br>1: Receiving | ○ | – |
| State_Reset | CAN reset status | 0: Operating<br>1: Reset | ○ | – |
| State_LoopBack | Loop back status | 0: Normal mode<br>1: Loop back mode | ○ | – |
| ____ | Nothing is assigned. When write, set to "0".<br>When read, its content is indeterminate. | | – | – |
| State_BasicCAN | Basic CAN status | 0: Normal mode<br>1: Basic CAN mode | ○ | – |
| State_BusError | CAN bus error | 0: No error occurred<br>1: Error occurred | ○ | – |
| State_ErrPas | Error passive status | 0: Not error passive state<br>1: Error passive state | ○ | – |
| State_BusOff | Bus-off status | 0: Not bus-off state<br>1: Bus-off state | ○ | – |
| ____ | Nothing is assigned. When write, set to "0".<br>When read, its content is indeterminate. | | – | – |

Note: This applies when the CAN module is supplied with a clock by setting the sleep mode control bit (bit 0 at address $0242_{16}$) to 1 after reset.

**Figure 1.22.6. CAN0 status register**

## 4. CAN0 status register

**Bits 0–3: Active slot determination bits (MBox)**

When the CAN module finished transmitting data or finished storing received data, the relevant slot number is stored in these bits.

The MBox bits cannot be cleared to 0 in software.

**Bit 4: Transmission-finished status (TrmSucc)**

[Set condition]

This bit is set to 1 when the CAN module finished transmitting data normally.

[Clear condition]

This bit is cleared when the CAN module finished receiving data normally.

**Bit 5: Reception-finished status (RecSucc)**

[Set condition]

This bit is set to 1 when the CAN module finished receiving data normally (regardless of whether the received message has been stored in a message slot). However, this bit is not set if the received message is one that was transmitted in loopback mode.

[Clear condition]

This bit is cleared when the CAN module finished transmitting data normally.

**Bit 6: Transmission status (TrmState)**

[Set condition]

This bit is set to 1 when the CAN module is operating as a transmit node.

[Clear condition]

This bit is cleared when the CAN module goes to a bus-idle state or starts operating as a receive node.

**Bit 7: Reception status (RecState)**

[Set condition]

This bit is set to 1 when the CAN module is operating as a receive node.

[Clear condition]

This bit is cleared when the CAN module goes to a bus-idle state or starts operating as a transmit node.

**Bit 8: CAN reset status (State_Reset)**

When the State_Reset bit = 1, it means that the CAN module is in a reset state.

[Set condition]

This bit is set to 1 when CAN module is in a reset state.

[Clear condition]

This bit is cleared by clearing the Reset0 or Reset1 bits to 0.

**Bit 9: Loopback status (State_loopBack)**

When the State_loopBack bit = 1, it means that the CAN module is operating in loopback mode.

[Set condition]

This bit is set to 1 by setting the CAN control register LoopBack bit to 1.

[Clear condition]

This bit is cleared by clearing the LoopBack bit to 0.

**Bit 11: BasicCAN status (State_BasicCAN)**

When the State_BasicCAN bit = 1, it means that the CAN module is operating in BasicCAN mode.

[Set condition]

This bit is set to 1 when the CAN module is operating in BasicCAN mode.

Conditions for the CAN module to operate in BasicCAN mode are as follows:

• The CAN Control Register BasicCAN bit is set to 1.

• Slots 14 and 15 both are set for data frame reception.

[Clear condition]

This bit is cleared by clearing the BasicCAN bit to 0.

**Bit 12: CAN bus error (State_BusError)**

[Set condition]

This bit is set to 1 when an error on the CAN bus is detected.

[Clear condition]

This bit is cleared when the CAN module finished transmitting or receiving normally. Clearing of this bit does not depend on whether the received message has been stored in a message slot.

Note :When this bit is 1, although CAN module is reset, this bit does not become to 0.

**Bit 13: Error passive status (State_ErrPas)**

When the State_ErrPas bit = 1, it means that the CAN module is in an error-passive state.

[Set condition]

This bit is set to 1 when the value of C0TEC register or C0REC register exceeds 127, with the CAN module in an error-passive state.

[Clear condition]

This bit is cleared when the CAN module goes from the error-passive state to any other error state.

Note :When this bit is 1, then CAN module is reset, this bit becomes 0 automatically.

**Bit 14: Bus-off status (State_BusOff)**

When the State_BusOff bit = 1, it means that the CAN module is in a bus-off state.

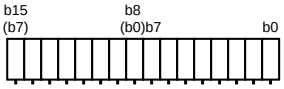
[Set condition]

This bit is set to 1 when the value of the C0TEC register exceeds 255, with the CAN module in a bus-off state.

[Clear condition]

This bit is cleared when the CAN module returns from the bus-off state.

CAN0 extended ID register




| Symbol | Address | When reset (Note) |
|---|---|---|
| C0IDR | $0205_{16}$,$0204_{16}$ | $0000_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| IDE15 | Expansion ID15 (slot 15) | 0: Standard ID format<br>1: Extended ID format | ○ | ○ |
| IDE14 | Expansion ID14 (slot 14) | 0: Standard ID format<br>1: Extended ID format | ○ | ○ |
| IDE13 | Expansion ID13 (slot 13) | 0: Standard ID format<br>1: Extended ID format | ○ | ○ |
| IDE12 | Expansion ID12 (slot 12) | 0: Standard ID format<br>1: Extended ID format | ○ | ○ |
| IDE11 | Expansion ID11 (slot 11) | 0: Standard ID format<br>1: Extended ID format | ○ | ○ |
| IDE10 | Expansion ID10 (slot 10) | 0: Standard ID format<br>1: Extended ID format | ○ | ○ |
| IDE9 | Expansion ID9 (slot 9) | 0: Standard ID format<br>1: Extended ID format | ○ | ○ |
| IDE8 | Expansion ID8 (slot 8) | 0: Standard ID format<br>1: Extended ID format | ○ | ○ |
| IDE7 | Expansion ID7 (slot 7) | 0: Standard ID format<br>1: Extended ID format | ○ | ○ |
| IDE6 | Expansion ID6 (slot 6) | 0: Standard ID format<br>1: Extended ID format | ○ | ○ |
| IDE5 | Expansion ID5 (slot 5) | 0: Standard ID format<br>1: Extended ID format | ○ | ○ |
| IDE4 | Expansion ID4 (slot 4) | 0: Standard ID format<br>1: Extended ID format | ○ | ○ |
| IDE3 | Expansion ID3 (slot 3) | 0: Standard ID format<br>1: Extended ID format | ○ | ○ |
| IDE2 | Expansion ID2 (slot 2) | 0: Standard ID format<br>1: Extended ID format | ○ | ○ |
| IDE1 | Expansion ID1 (slot 1) | 0: Standard ID format<br>1: Extended ID format | ○ | ○ |
| IDE0 | Expansion ID0 (slot 0) | 0: Standard ID format<br>1: Extended ID format | ○ | ○ |

Note: This applies when the CAN module is supplied with a clock by setting the sleep mode control bit (bit 0 at address $0242_{16}$) to 1 after reset.

**Figure 1.22.7. CAN0 extended ID register**

## 5. CAN0 extended ID register

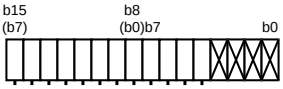
This register selects the format of a frame handled by the message slot that corresponds to each bit in this register.

Setting any bit to 0 selects the standard (Standard ID) format.

Setting any bit to 1 selects the extended (Extended ID) format.

Note 1: When setting or resetting any bit in this register, make sure the corresponding slot has no transmit or receive request.

## CAN0 configuration register




| Symbol | Address | When reset (Note) |
|---|---|---|
| C0CONR | $0207_{16}$,$0206_{16}$ | $000X_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| ——— | Nothing is assigned. When write, set to "0". When read, their contents are indeterminate. | | — | — |
| ——— | | | — | — |
| ——— | | | — | — |
| ——— | | | — | — |
| SAM | Sampling number | 0: Sampled once<br>1: Sampled three times | ○ | ○ |
| PTS0 | Propagation Time Segment | b7 b6 b5<br>0 0 0: Propagation Time Segment = 1Tq<br>0 0 1: Propagation Time Segment = 2Tq<br>0 1 0: Propagation Time Segment = 3Tq<br>0 1 1: Propagation Time Segment = 4Tq<br>1 0 0: Propagation Time Segment = 5Tq<br>1 0 1: Propagation Time Segment = 6Tq<br>1 1 0: Propagation Time Segment = 7Tq<br>1 1 1: Propagation Time Segment = 8Tq | ○ | ○ |
| PTS1 | | | ○ | ○ |
| PTS2 | | | ○ | ○ |
| PBS10 | Phase Buffer Segment 1 | b10 b9 b8<br>0 0 0: Must not be set<br>0 0 1: Phase Buffer Segment 1 = 2Tq<br>0 1 0: Phase Buffer Segment 1 = 3Tq<br>0 1 1: Phase Buffer Segment 1 = 4Tq<br>1 0 0: Phase Buffer Segment 1 = 5Tq<br>1 0 1: Phase Buffer Segment 1 = 6Tq<br>1 1 0: Phase Buffer Segment 1 = 7Tq<br>1 1 1: Phase Buffer Segment 1 = 8Tq | ○ | ○ |
| PBS11 | | | ○ | ○ |
| PBS12 | | | ○ | ○ |
| PBS20 | Phase Buffer Segment 2 | b13 b12 b11<br>0 0 0: Must not be set<br>0 0 1: Phase Buffer Segment 2 = 2Tq<br>0 1 0: Phase Buffer Segment 2 = 3Tq<br>0 1 1: Phase Buffer Segment 2 = 4Tq<br>1 0 0: Phase Buffer Segment 2 = 5Tq<br>1 0 1: Phase Buffer Segment 2 = 6Tq<br>1 1 0: Phase Buffer Segment 2 = 7Tq<br>1 1 1: Phase Buffer Segment 2 = 8Tq | ○ | ○ |
| PBS21 | | | ○ | ○ |
| PBS22 | | | ○ | ○ |
| SJW0 | reSynchronization Jump Width | b15 b14<br>0 0: SJW = 1Tq<br>0 1: SJW = 2Tq<br>1 0: SJW = 3Tq<br>1 1: SJW = 4Tq | ○ | ○ |
| SJW1 | | | ○ | ○ |

Note: This applies when the CAN module is supplied with a clock by setting the sleep mode control bit (bit 0 at address $0242_{16}$) to 1 after reset.

**Figure 1.22.8. CAN0 configuration register**

## 6. CAN0 configuration register

**Bit 4: SAM bit (SAM)**

This bit sets the sampling number per one bit.

0: The value sampled at the last of the Phase Buffer Segment 1 becomes the bit value.

1: The bit value is determined by the majority operation circuit using values sampled at the following three points: the last of the Phase Buffer Segment 1, before 1Tq, and before 2Tq.

**Bits 5–7: PTS bits (RTS00-RTS02)**

These bits set the width of Propagation Time Segment.

**Bits 8–10: PBS1 bits (PBS10-PBS12)**

These bits set the width of Phase Buffer Segment 1. The PBS1 bits must be set to 1 or greater.

**Bits 11–13: PBS2 bits (PBS20-PBS22)**

These bits set the width of Phase Buffer Segment 2. The PBS2 bits must be set to 1 or greater.

**Bits 14, 15: SJW bits (SJW0, SJW1)**

These bits set the width of reSynchronization Jump Width. The SJW bits must be set to a value equal to or less than PBS2.

**Table 1.22.2 Bit Timing Setup Example when the CPU Clock = 30 MHz**

| Baud rate | BRP | Tq period (ns) | 1 bit's Tq number | PTS+PBS1 | PBS2 | Sample point |
|---|---|---|---|---|---|---|
| 1Mbps | 1 | 66.7 | 15 | 12 | 2 | 87% |
| | 1 | 66.7 | 15 | 11 | 3 | 80% |
| | 1 | 66.7 | 15 | 10 | 4 | 73% |
| | 2 | 100 | 10 | 7 | 2 | 80% |
| | 2 | 100 | 10 | 6 | 3 | 70% |
| | 2 | 100 | 10 | 5 | 4 | 60% |
| 500Kbps | 2 | 100 | 20 | 16 | 3 | 85% |
| | 2 | 100 | 20 | 15 | 4 | 80% |
| | 2 | 100 | 20 | 14 | 5 | 75% |
| | 3 | 133.3 | 15 | 12 | 2 | 87% |
| | 3 | 133.3 | 15 | 11 | 3 | 80% |
| | 3 | 133.3 | 15 | 10 | 4 | 73% |
| | 4 | 166.7 | 12 | 9 | 2 | 83% |
| | 4 | 166.7 | 12 | 8 | 3 | 75% |
| | 4 | 166.7 | 12 | 7 | 4 | 67% |
| | 5 | 200 | 10 | 7 | 2 | 80% |
| | 5 | 200 | 10 | 6 | 3 | 70% |
| | 5 | 200 | 10 | 5 | 4 | 60% |

CAN0 time stamp register

b15 (b7) b8 (b0) b7 b0

(Upper 8-bit) (Lower 8-bit)

| Symbol | Address | When reset (Note) |
|---|---|---|
| C0TSR | $0209_{16}$,$0208_{16}$ | $0000_{16}$ |

| Function | Setting range | R | W |
|---|---|---|---|
| 16 bits count value | $0000_{16}$ to $FFFF_{16}$ | ○ | — |

Note: This applies when the CAN module is supplied with a clock by setting the sleep mode control bit (bit 0 at address $0242_{16}$) to 1 after reset.

**Figure 1.22.9. CAN0 time stamp register**

## 7. CAN0 Timestamp register

The CAN module incorporates a 16-bit counter. The count period for this counter can be derived from the CAN bus bit period by dividing it by 1, 2, 3, or 4 using the CAN0 control register0 (C0CTLR0)'s TSPre0, 1 bits.

When the CAN module finishes transmitting or receiving, the CAN0 Timestamp Register (C0TSR) value is captured and the value is automatically stored in a message slot.

The C0TSR register starts counting upon clearing the C0CTLR register's Reset and Reset1 bits to 0.

Note 1: Setting the C0CTLR0 register's Reset0 and Reset1 bits to 1 resets CAN, and the C0TSR register thereby initialized to $0000_{16}$. Also, setting the TSReset (timestamp counter reset) bit to 1 initializes the C0TSR register to $0000_{16}$ on-the-fly (while the CAN remains operating; CAN0 status register's State_Reset bit is "0").

Note 2: During loopback mode, if any receive slot exists in which a message can be stored, the C0TSR register value is stored in the corresponding slot when the CAN module finished receiving. (This storing of the C0TSR register value does not occur at completion of transmission.)

CAN0 transmit error count register

b7 b0

8-bit

| Symbol | Address | When reset (Note) |
|---|---|---|
| C0TEC | $020A_{16}$ | $00_{16}$ |

| Function | R | W |
|---|---|---|
| Transmit error count value | ○ | — |

Note: This applies when the CAN module is supplied with a clock by setting the sleep mode control bit (bit 0 at address $0242_{16}$) to 1 after reset.

**Figure 1.22.10. CAN0 transmit error count register**

## 8. CAN0 transmit error count register

When in an error active or an error passive state, the transmit error count value is stored in this register. The count is decremented when the CAN module finished transmitting normally or incremented when an error occurred while transmitting.

When in a bus-off state, an indeterminate value is stored in this register. The register is reset to $00_{16}$ upon returning to an error active state.

CAN0 reception error count register

| b7 ... b0 | Symbol | Address | When reset (Note) |
|---|---|---|---|
| 8-bit | C0REC | 020B16 | 0016 |

| Function | R | W |
|---|---|---|
| Reception error count value | ○ | — |

Note: This applies when the CAN module is supplied with a clock by setting the sleep mode control bit (bit 0 at address 024216) to 1 after reset.

**Figure 1.22.11. CAN0 reception error count register**

## 9. CAN0 reception error count register

When in an error active or an error passive state, the receive error count value is stored in this register. The count is decremented when the CAN module finished receiving normally or incremented when an error occurred while receiving.

When C0REC ≥ 128 (error passive state) at the time the CAN module finished receiving normally, the C0REC register is set to 127.

When in a bus-off state, an indeterminate value is stored in this register. The register is reset to 0016 upon returning to an error active state.

CAN0 baud rate prescaler

| b7 ... b0 | Symbol | Address | When reset (Note 1) |
|---|---|---|---|
| 8-bit | C0BRP | 021716 | 0116 |

| Function | Setting range | R | W |
|---|---|---|---|
| Baud rate prescaler value selected | 0116 to FF16 (Note 2) | ○ | ○ |

Note 1: This applies when the CAN module is supplied with a clock by setting the sleep mode control bit (bit 0 at address 024216) to 1 after reset.

Note 2: Do not set to "0016" (division by 1).

**Figure 1.22.12. CAN0 baud rate register**

## 10. CAN0 baud rate prescaler

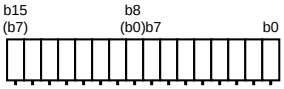
This register is used to set the Tq period, the CAN bit time. The CAN baud rate is determined by (Tq period x number of Tq's in one bit).

Tq period = (C0BRP+1)/CPU clock

CAN baud rate = 1 / (Tq period x number of Tq's in one bit)

Number of Tq's in one bit = Synchronization Segment +
Propagation Time Segment +
Phase Buffer Segment 1 +
Phase Buffer Segment 2

## CAN0 slot interrupt status register




| Symbol | Address | When reset (Note 1) |
|---|---|---|
| C0SISTR | $020D_{16}$,$020C_{16}$ | $0000_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SIS15 | Slot 15 interrupt request status bit | 0: Interrupt not requested<br>1: Interrupt requested (Note 2) | ○ | ○ |
| SIS14 | Slot 14 interrupt request status bit | 0: Interrupt not requested<br>1: Interrupt requested (Note 2) | ○ | ○ |
| SIS13 | Slot 13 interrupt request status bit | 0: Interrupt not requested<br>1: Interrupt requested (Note 2) | ○ | ○ |
| SIS12 | Slot 12 interrupt request status bit | 0: Interrupt not requested<br>1: Interrupt requested (Note 2) | ○ | ○ |
| SIS11 | Slot 11 interrupt request status bit | 0: Interrupt not requested<br>1: Interrupt requested (Note 2) | ○ | ○ |
| SIS10 | Slot 10 interrupt request status bit | 0: Interrupt not requested<br>1: Interrupt requested (Note 2) | ○ | ○ |
| SIS9 | Slot 9 interrupt request status bit | 0: Interrupt not requested<br>1: Interrupt requested (Note 2) | ○ | ○ |
| SIS8 | Slot 8 interrupt request status bit | 0: Interrupt not requested<br>1: Interrupt requested (Note 2) | ○ | ○ |
| SIS7 | Slot 7 interrupt request status bit | 0: Interrupt not requested<br>1: Interrupt requested (Note 2) | ○ | ○ |
| SIS6 | Slot 6 interrupt request status bit | 0: Interrupt not requested<br>1: Interrupt requested (Note 2) | ○ | ○ |
| SIS5 | Slot 5 interrupt request status bit | 0: Interrupt not requested<br>1: Interrupt requested (Note 2) | ○ | ○ |
| SIS4 | Slot 4 interrupt request status bit | 0: Interrupt not requested<br>1: Interrupt requested (Note 2) | ○ | ○ |
| SIS3 | Slot 3 interrupt request status bit | 0: Interrupt not requested<br>1: Interrupt requested (Note 2) | ○ | ○ |
| SIS2 | Slot 2 interrupt request status bit | 0: Interrupt not requested<br>1: Interrupt requested (Note 2) | ○ | ○ |
| SIS1 | Slot 1 interrupt request status bit | 0: Interrupt not requested<br>1: Interrupt requested (Note 2) | ○ | ○ |
| SIS0 | Slot 0 interrupt request status bit | 0: Interrupt not requested<br>1: Interrupt requested (Note 2) | ○ | ○ |

Note 1: This applies when the CAN module is supplied with a clock by setting the sleep mode control bit (bit 0 at address $0242_{16}$) to 1 after reset.

Note 2: "0" can be set. When set to "1", the previous value is remained.

**Figure 1.22.13. CAN slot interrupt status register**

## 9. CAN0 slot interrupt status register

When using CAN interrupts, the CAN0 Slot Interrupt Status Register helps to know which slot requested an interrupt.

**• For transmit slots**

The status is set to 1 when the CAN module finished storing the CAN Timestamp Register value in the message slot after completing transmission.
To clear this bit, write 0 in software (Note 1).

**• For receive slots**

The status is set to 1 when the CAN module finished storing the received message in the message slot after completing reception.
To clear this bit, write 0 in software (Note 1).

Note 1: To clear any bit of the CAN Interrupt Status Register, write 0 to the bit to be cleared and 1 to all other bits, without using bit clear instructions.

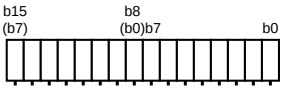
```
Example : Assembler language    mov.w   #07FFFh, C0SISTR
          C language            c0sister = 0x7FFF;
```

Note 2: For remote frame receive slots whose automatic answering function is enabled, the slot interrupt status bit is set when the CAN module finished receiving a remote frame and when it finished transmitting a data frame.

Note 3: For remote frame transmit slots, the slot interrupt status bit is set when the CAN module finished transmitting a remote frame and when it finished receiving a data frame.

Note 4: If the slot interrupt status bit is set by an interrupt request at the same time it is cleared by writing in software, the former has priority, i.e., the slot interrupt status bit is set.

CAN0 slot interrupt mask register




| Symbol | Address | When reset (Note) |
|---|---|---|
| C0SIMKR | $0211_{16}$,$0210_{16}$ | $0000_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SIM15 | Slot 15 interrupt request mask bit | 0: Interrupt request masked (disabled)<br>1: Interrupt request enabled | ○ | ○ |
| SIM14 | Slot 14 interrupt request mask bit | 0: Interrupt request masked (disabled)<br>1: Interrupt request enabled | ○ | ○ |
| SIM13 | Slot 13 interrupt request mask bit | 0: Interrupt request masked (disabled)<br>1: Interrupt request enabled | ○ | ○ |
| SIM12 | Slot 12 interrupt request mask bit | 0: Interrupt request masked (disabled)<br>1: Interrupt request enabled | ○ | ○ |
| SIM11 | Slot 11 interrupt request mask bit | 0: Interrupt request masked (disabled)<br>1: Interrupt request enabled | ○ | ○ |
| SIM10 | Slot 10 interrupt request mask bit | 0: Interrupt request masked (disabled)<br>1: Interrupt request enabled | ○ | ○ |
| SIM9 | Slot 9 interrupt request mask bit | 0: Interrupt request masked (disabled)<br>1: Interrupt request enabled | ○ | ○ |
| SIM8 | Slot 8 interrupt request mask bit | 0: Interrupt request masked (disabled)<br>1: Interrupt request enabled | ○ | ○ |
| SIM7 | Slot 7 interrupt request mask bit | 0: Interrupt request masked (disabled)<br>1: Interrupt request enabled | ○ | ○ |
| SIM6 | Slot 6 interrupt request mask bit | 0: Interrupt request masked (disabled)<br>1: Interrupt request enabled | ○ | ○ |
| SIM5 | Slot 5 interrupt request mask bit | 0: Interrupt request masked (disabled)<br>1: Interrupt request enabled | ○ | ○ |
| SIM4 | Slot 4 interrupt request mask bit | 0: Interrupt request masked (disabled)<br>1: Interrupt request enabled | ○ | ○ |
| SIM3 | Slot 3 interrupt request mask bit | 0: Interrupt request masked (disabled)<br>1: Interrupt request enabled | ○ | ○ |
| SIM2 | Slot 2 interrupt request mask bit | 0: Interrupt request masked (disabled)<br>1: Interrupt request enabled | ○ | ○ |
| SIM1 | Slot 1 interrupt request mask bit | 0: Interrupt request masked (disabled)<br>1: Interrupt request enabled | ○ | ○ |
| SIM0 | Slot 0 interrupt request mask bit | 0: Interrupt request masked (disabled)<br>1: Interrupt request enabled | ○ | ○ |

Note: This applies when the CAN module is supplied with a clock by setting the sleep mode control bit (bit 0 at address $0242_{16}$) to 1 after reset.

**Figure 1.22.14. CAN0 slot interrupt mask register**

## 12. CAN0 slot interrupt mask register

This register controls CAN interrupts by enabling or disabling interrupt requests generated by each corresponding slot at completion of transmission or reception. Setting any bit of this register (SIMn where n = 0–15) to 1 enables the interrupt request to be generated by the corresponding slot at completion of transmission or reception.

CAN0 error interrupt mask register

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

| Symbol | Address | When reset (Note) |
|---|---|---|
| C0EIMKR | $0214_{16}$ | XXXX $X000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| BOIM | Bus off interrupt mask bit | 0: Interrupt request masked (disabled)<br>1: Interrupt request enabled | ○ | ○ |
| EPIM | Error passive interrupt mask bit | 0: Interrupt request masked (disabled)<br>1: Interrupt request enabled | ○ | ○ |
| BEIM | CAN bus error interrupt mask bit | 0: Interrupt request masked (disabled)<br>1: Interrupt request enabled | ○ | ○ |
| ——— | Nothing is assigned. When write, set to "0". When read, their contents are indeterminate. | | – | – |
| ——— | | | – | – |
| ——— | | | – | – |
| ——— | | | – | – |
| ——— | | | – | – |

Note: This applies when the CAN module is supplied with a clock by setting the sleep mode control bit (bit 0 at address $0242_{16}$) to 1 after reset.

**Figure 1.22.15. CAN0 error interrupt mask register**

## 13. CAN0 error interrupt mask register

**Bit 0: Bus-off interrupt mask bit (BOIM)**

This bit controls CAN interrupts by enabling or disabling interrupt requests generated when the CAN module goes to a bus-off state. Setting this bit to 1 enables a bus-off interrupt request.

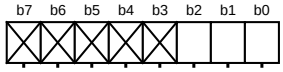
**Bit 1: Error passive interrupt mask bit (EPIM)**

This bit controls CAN interrupts by enabling or disabling interrupt requests generated when the CAN module goes to an error passive state. Setting this bit to 1 enables an error passive interrupt request.

**Bit 2: CAN bus error interrupt mask bit (BEIM)**

This bit controls CAN interrupts by enabling or disabling interrupt requests generated by occurrence of a CAN bus error. Setting this bit to 1 enables a CAN bus error interrupt request.

CAN0 error interrupt status register




| Symbol | Address | When reset (Note 1) |
|---|---|---|
| C0EISTR | $0215_{16}$ | $XXXX\ X000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| BOIS | Bus off interrupt status bit | 0: Interrupt not requested<br>1: Interrupt requested (Note 2) | ○ | ○ |
| EPIS | Error passive interrupt status bit | 0: Interrupt not requested<br>1: Interrupt requested (Note 2) | ○ | ○ |
| BEIS | CAN bus error interrupt status bit | 0: Interrupt not requested<br>1: Interrupt requested (Note 2) | ○ | ○ |
| — | Nothing is assigned. When write, set to "0". When read, their contents are indeterminate. | | — | — |
| — | | | — | — |
| — | | | — | — |
| — | | | — | — |
| — | | | — | — |

Note 1: This applies when the CAN module is supplied with a clock by setting the sleep mode control bit (bit 0 at address $0242_{16}$) to 1 after reset.

Note 2: "0" can be set. When set to "1", the previous value is remained.

**Figure 1.22.16. CAN0 error interrupt status register**

## 14. CAN0 error interrupt status register

When using CAN interrupts, the CAN Error Interrupt Status Register helps to verify the causes of error-derived interrupts.

**Bit 0: Bus-off interrupt status bit (BOIS)**

This bit is set to 1 when the CAN module goes to a bus-off state.

To clear this bit, write 0 in software (Note 1).

**Bit 1: Error passive interrupt status bit (EPIS)**

This bit is set to 1 when the CAN module goes to an error passive state.

To clear this bit, write 0 in software (Note 1).

**Bit 2: CAN bus error interrupt status bit (BEIS)**

This bit is set to 1 when a CAN communication error is detected.

To clear this bit, write 0 in software (Note 1).

Note 1: To clear any bit of the CAN Error Interrupt Status Register, write 0 to the bit to be cleared and 1 to all other bits, without using bit clear instructions.

Example: Assembler language mov.B #006h, C0EISTR

C language c0eistr = 0x06;

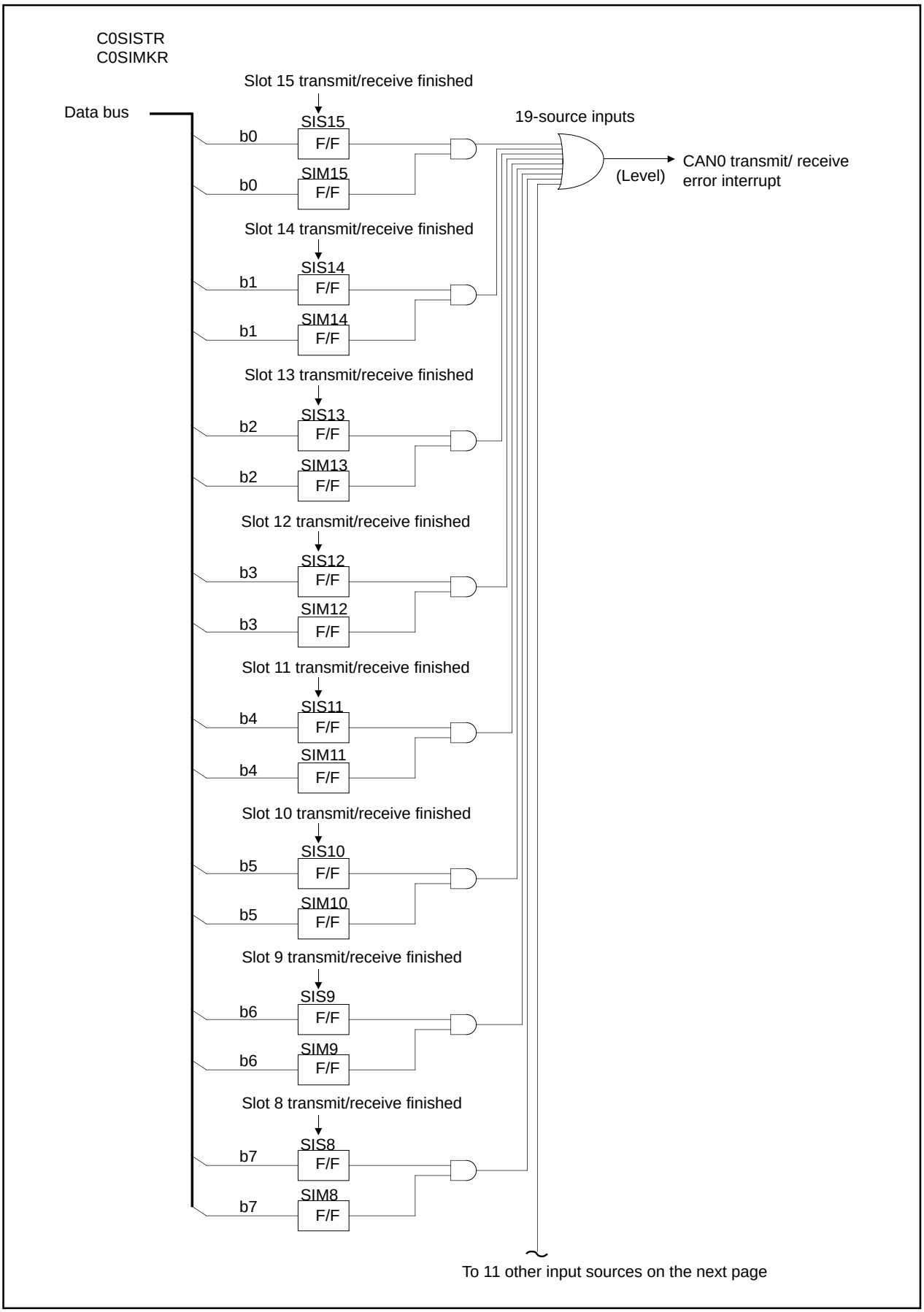



**Figure 1.22.17. CAN0 transmit, receive and error interrupt block diagram (1/3)**

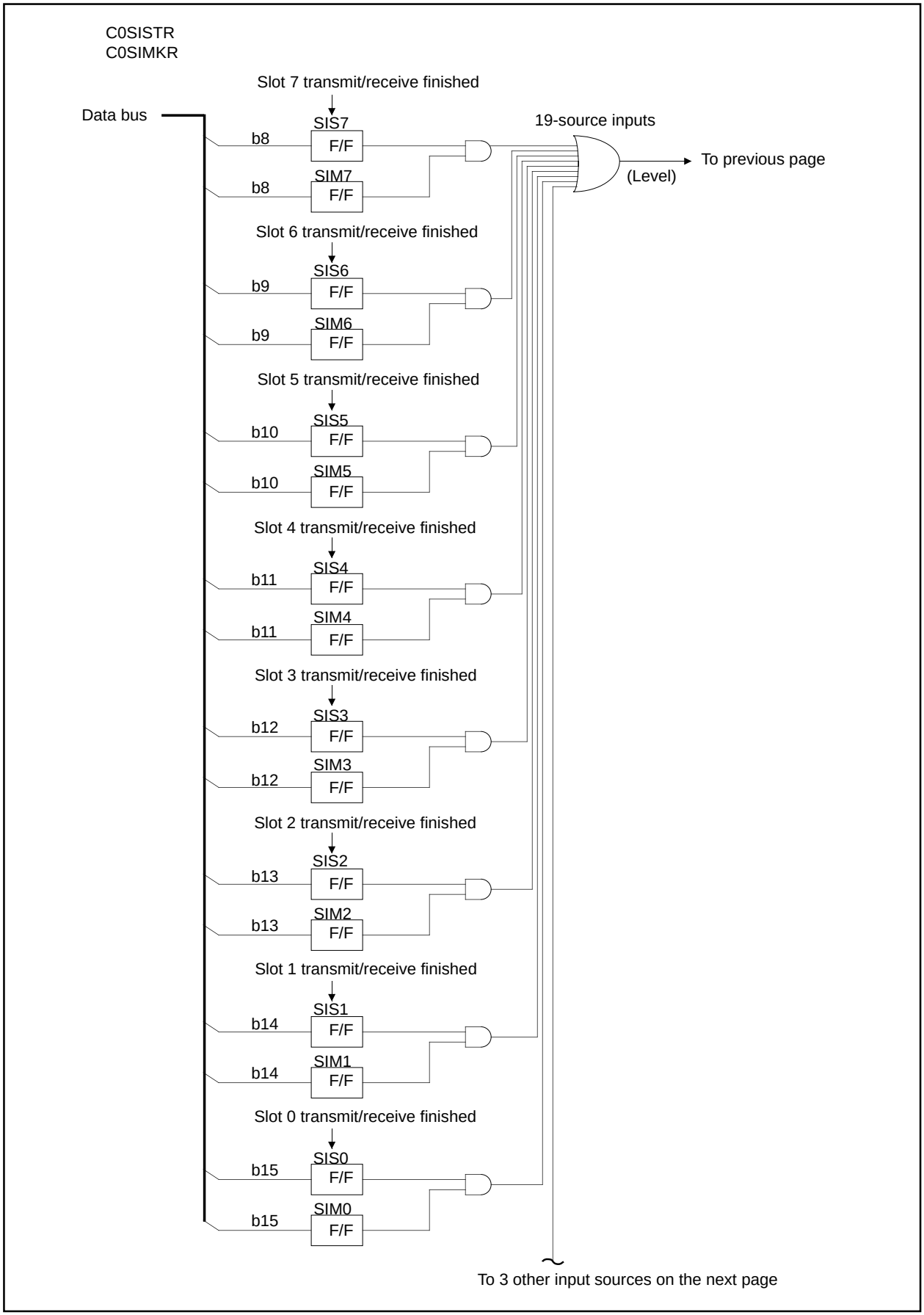



**Figure 1.22.18. CAN0 transmit, receive and error interrupt block diagram (2/3)**

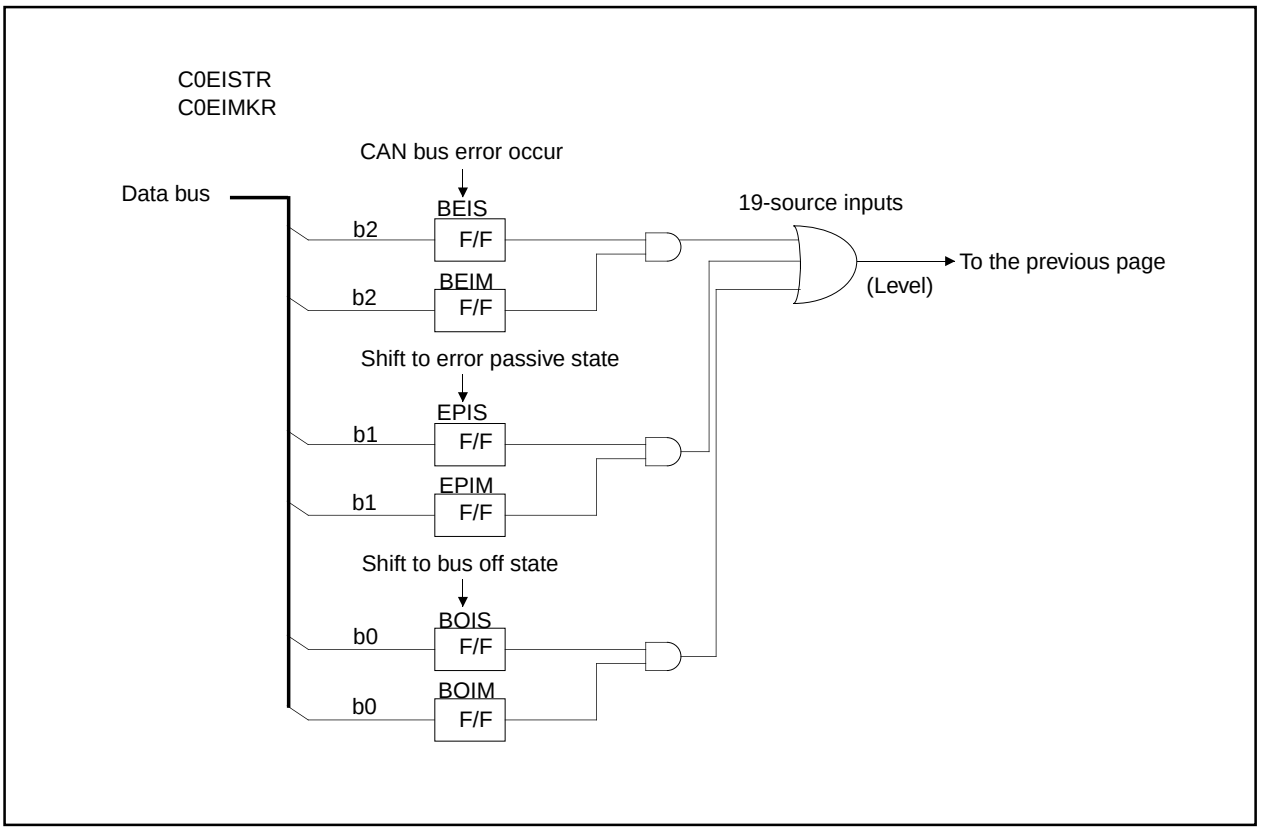
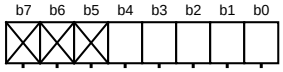



**Figure 1.22.19. CAN0 transmit, receive and error interrupt block diagram (3/3)**

CAN0 global mask register standard ID0
CAN0 local mask register A, B standard ID0




| Symbol | Address | When reset (Note) |
|---|---|---|
| C0GMR0 | $0228_{16}$ | $XXX0\ 0000_2$ |
| C0LMAR0 | $0230_{16}$ | $XXX0\ 0000_2$ |
| C0LMBR0 | $0238_{16}$ | $XXX0\ 0000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SID6M | Standard ID6 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| SID7M | Standard ID7 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| SID8M | Standard ID8 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| SID9M | Standard ID9 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| SID10M | Standard ID10 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| —— | Nothing is assigned. When write, set to "0".<br>When read, their contents are indeterminate. | | – | – |
| —— | | | – | – |
| —— | | | – | – |

Note: This applies when the CAN module is supplied with a clock by setting the sleep mode control bit (bit 0 at address $0242_{16}$) to 1 after reset.

**Figure 1.22.20. CAN0 global mask register standard ID0 and CAN0 local mask register A, B standard ID0**

## 15. CAN0 global mask register standard ID0

### CAN0 local mask register A, B standard ID0

The mask registers used for acceptance filtering consist of the global mask register, local mask register A, and local mask register B.

The global mask register takes care of message slots 0–13 whereas local mask registers A and B are used for message slots 14 and 15, respectively.

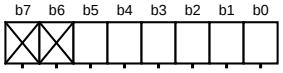
- If any bit of this register is set to 0, its corresponding ID bit is masked during acceptance filtering. (The masked bit is not checked for ID; the ID is assumed to be matching.)
- If any bit of this register is set to 1, its corresponding ID bit is compared with the received ID during acceptance filtering. If it matches the ID that is set in any message slot, the received data is stored in that slot.

Note 1: The global mask register can only be modified when none of the slots 0–13 has receive requests set.

Note 2: The local mask register A can only be modified when slot 14 has no receive requests set.

Note 3: The local mask register B can only be modified when slot 15 has no receive requests set.

CAN0 global mask register standard ID1
CAN0 local mask register A, B standard ID1




| Symbol | Address | When reset (Note) |
|---|---|---|
| C0GMR1 | $0229_{16}$ | $XX00\ 0000_2$ |
| C0LMAR1 | $0231_{16}$ | $XX00\ 0000_2$ |
| C0LMBR1 | $0239_{16}$ | $XX00\ 0000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SID0M | Standard ID0 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| SID1M | Standard ID1 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| SID2M | Standard ID2 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| SID3M | Standard ID3 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| SID4M | Standard ID4 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| SID5M | Standard ID5 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| — | Nothing is assigned. When write, set to "0".<br>When read, their contents are indeterminate. | | — | — |
| — | | | — | — |

Note: This applies when the CAN module is supplied with a clock by setting the sleep mode control bit (bit 0 at address $0242_{16}$) to 1 after reset.

**Figure 1.22.21. CAN0 global mask register standard ID1 and CAN0 local mask register A, B standard ID1**

## 16. CAN0 global mask register standard ID1
## CAN0 local mask register A, B standard ID1

The mask registers used for acceptance filtering consist of the global mask register, local mask register A, and local mask register B.

The global mask register takes care of message slots 0–13 whereas local mask registers A and B are used for message slots 14 and 15, respectively.

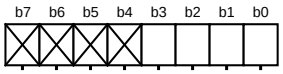
- If any bit of this register is set to 0, its corresponding ID bit is masked during acceptance filtering. (The masked bit is not checked for ID; the ID is assumed to be matching.)
- If any bit of this register is set to 1, its corresponding ID bit is compared with the received ID during acceptance filtering. If it matches the ID that is set in any message slot, the received data is stored in that slot.

Note 1: The global mask register can only be modified when none of the slots 0–13 has receive requests set.

Note 2: The local mask register A can only be modified when slot 14 has no receive requests set.

Note 3: The local mask register B can only be modified when slot 15 has no receive requests set.

CAN0 global mask register extend ID0
CAN0 local mask register A, B extend ID0




| Symbol | Address | When reset (Note) |
|---|---|---|
| C0GMR2 | $022A_{16}$ | $XXXX\ 0000_2$ |
| C0LMAR2 | $0232_{16}$ | $XXXX\ 0000_2$ |
| C0LMBR2 | $023A_{16}$ | $XXXX\ 0000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| EID14M | Extend ID14 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| EID15M | Extend ID15 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| EID16M | Extend ID16 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| EID17M | Extend ID17 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| — | Nothing is assigned. When write, set to "0".<br>When read, their contents are indeterminate. | | — | — |
| — | | | — | — |
| — | | | — | — |
| — | | | — | — |

Note: This applies when the CAN module is supplied with a clock by setting the sleep mode control bit (bit 0 at address $0242_{16}$) to 1 after reset.

**Figure 1.22.22. CAN0 global mask register extend ID0 and CAN0 local mask register A, B extend ID0**

## 17. CAN0 global mask register extend ID0

### CAN0 local mask register A, B extend ID0

The mask registers used for acceptance filtering consist of the global mask register, local mask register A, and local mask register B.

The global mask register takes care of message slots 0–13 whereas local mask registers A and B are used for message slots 14 and 15, respectively.

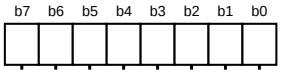
- If any bit of this register is set to 0, its corresponding ID bit is masked during acceptance filtering. (The masked bit is not checked for ID; the ID is assumed to be matching.)
- If any bit of this register is set to 1, its corresponding ID bit is compared with the received ID during acceptance filtering. If it matches the ID that is set in any message slot, the received data is stored in that slot.

Note 1: The global mask register can only be modified when none of the slots 0–13 has receive requests set.

Note 2: The local mask register A can only be modified when slot 14 has no receive requests set.

Note 3: The local mask register B can only be modified when slot 15 has no receive requests set.

CAN0 global mask register extend ID1
CAN0 local mask register A, B extend ID1




| Symbol | Address | When reset (Note) |
|---|---|---|
| C0GMR3 | $022B_{16}$ | $00_{16}$ |
| C0LMAR3 | $0233_{16}$ | $00_{16}$ |
| C0LMBR3 | $023B_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| EID6M | Extend ID6 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| EID7M | Extend ID7 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| EID8M | Extend ID8 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| EID9M | Extend ID9 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| EID10M | Extend ID10 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| EID11M | Extend ID11 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| EID12M | Extend ID12 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| EID13M | Extend ID13 | 0: ID not checked<br>1: ID checked | ○ | ○ |

Note: This applies when the CAN module is supplied with a clock by setting the sleep mode control bit (bit 0 at address $0242_{16}$) to 1 after reset.

**Figure 1.22.23. CAN0 global mask register extend ID1 and CAN0 local mask register A, B extend ID1**

## 18. CAN0 global mask register extend ID1

### CAN0 local mask register A, B extend ID1

The mask registers used for acceptance filtering consist of the global mask register, local mask register A, and local mask register B.

The global mask register takes care of message slots 0–13, whereas local mask registers A and B are used for message slots 14 and 15, respectively.

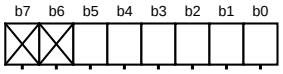
- If any bit of this register is set to 0, its corresponding ID bit is masked during acceptance filtering. (The masked bit is not checked for ID; the ID is assumed to be matching.)
- If any bit of this register is set to 1, its corresponding ID bit is compared with the received ID during acceptance filtering. If it matches the ID that is set in any message slot, the received data is stored in that slot.

Note 1: The global mask register can only be modified when none of the slots 0–13 has receive requests set.

Note 2: The local mask register A can only be modified when slot 14 has no receive requests set.

Note 3: The local mask register B can only be modified when slot 15 has no receive requests set.

CAN0 global mask register extend ID2
CAN0 local mask register A, B extend ID2




| Symbol | Address | When reset (Note) |
|---|---|---|
| C0GMR4 | 022C16 | XX00 00002 |
| C0LMAR4 | 023416 | XX00 00002 |
| C0LMBR4 | 023C16 | XX00 00002 |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| EID0M | Extend ID0 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| EID1M | Extend ID1 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| EID2M | Extend ID2 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| EID3M | Extend ID3 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| EID4M | Extend ID4 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| EID5M | Extend ID5 | 0: ID not checked<br>1: ID checked | ○ | ○ |
| ——— | Nothing is assigned. When write, set to "0".<br>When read, their contents are indeterminate. | | – | – |
| ——— | | | – | – |

Note: This applies when the CAN module is supplied with a clock by setting the sleep mode control bit (bit 0 at address 024216) to 1 after reset.

**Figure 1.22.24. CAN0 global mask register extend ID2 and CAN0 local mask register A, B extend ID2**

## 19. CAN0 global mask register extend ID2

### CAN0 local mask register A, B extend ID2

The mask registers used for acceptance filtering consist of the global mask register, local mask register A, and local mask register B.

The global mask register takes care of message slots 0–13, whereas local mask registers A and B are used for message slots 14 and 15, respectively.

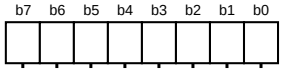
- If any bit of this register is set to 0, its corresponding ID bit is masked during acceptance filtering. (The masked bit is not checked for ID; the ID is assumed to be matching.)
- If any bit of this register is set to 1, its corresponding ID bit is compared with the received ID during acceptance filtering. If it matches the ID that is set in any message slot, the received data is stored in that slot.

Note 1: The global mask register can only be modified when none of the slots 0–13 has receive requests set.

Note 2: The local mask register A can only be modified when slot 14 has no receive requests set.

Note 3: The local mask register B can only be modified when slot 15 has no receive requests set.

## CAN0 message slot i control register (i=0 to 15)




| Symbol | Address | When reset (Note 1) |
|---|---|---|
| C0MCTLi(i=0 to 5) | $0230_{16}$, $0231_{16}$, $0232_{16}$, $0233_{16}$, $0234_{16}$, $0235_{16}$ | $00_{16}$ |
| C0MCTLi(i=6 to 11) | $0236_{16}$, $0237_{16}$, $0238_{16}$, $0239_{16}$, $023A_{16}$, $023B_{16}$ | $00_{16}$ |
| C0MCTLi(i=12 to 15) | $023C_{16}$, $023D_{16}$, $023E_{16}$, $023F_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| When receive, NewData<br>When transmit, SentData | Transmit/receive finished flag | When transmitting<br>0: Not transmitted yet<br>1: Finished transmitting<br>When receiving (Note 2)<br>0: Not received yet<br>1: Finished receiving | ○ | ○ |
| When receive, InvalData<br>When transmit, TrmActive | Transmitting/ receiving flag | When transmitting<br>0: Stopped transmitting<br>1: Accepted transmit request<br>When receiving<br>0: Stopped receiving<br>1: Storing received data | ○ | — |
| MsgLost | Overwrite flag | 0: Over run error not occurred<br>1: Over run error occurred (Note 2) | ○ | ○ |
| RemActive | Remote flame transmit/receive status flag | Using BasicCan mode<br>0: Data flame received (status) (Note 2)<br>1: Remote flame received (status)<br>Not using BasicCan mode<br>0: Data flame<br>1: Remote flame | ○ | ○ |
| RspLock | Automatic answering disable bit | 0: Automatic answering of remote flame enable<br>1: Automatic answering of remote flame disable | ○ | ○ |
| Remote | Remote frame set bit | 0: Transmit/receive data flame<br>1: Transmit/receive remote flame | ○ | ○ |
| RecReq | Receive request bit | 0: Reception not requested<br>1: Reception requested | ○ | ○ |
| TrmReq | Transmit request bit | 0: Transmission not requested<br>1: Transmission requested | ○ | ○ |

Note 1: This applies when the CAN module is supplied with a clock by setting the sleep mode control bit (bit 0 at address $0242_{16}$) to 1 after reset.

Note 2: "0" can be set. When set to "1", the previous value is remained.

**Figure 1.22.25. CAN0 message slot i control register**

## 20. CAN0 message slot i control register

**Bit 0: Transmission finished flag /reception finished flag (SentData, NewData)**

This bit indicates that the CAN module finished transmitting or receiving a message.

• For transmit slots

The bit is set to 1 when the CAN module finished transmitting from the message slot.

This bit is cleared by writing 0 in software. However, it cannot be cleared when the TrmActive (transmit/receive status) bit = 1.

• For receive slots

The bit is set to 1 when the CAN module finished receiving a message normally that is to be stored in the message slot.

This bit is cleared by writing 0 in software. However, it cannot be cleared when the InvalData (transmit/receive status) bit = 1.

Note 1: Before reading received data from the message slot, be sure to clear the NewData (transmission/reception finished status) bit. Also, if the NewData bit is set to 1 after readout, it means that new received data has been stored in the message slot while reading out from the slot, and that the read data contains an indeterminate value. In this case, discard the read data and clear the NewData bit before reading out from the slot again.

Note 2: The NewData bit is not set by a completion of remote frame transmission or reception.

**Bit 1: Transmitting flag /receiving flag (TrmActive, InvalData)**

This bit indicates that the CAN module is transmitting or receiving a message, with the message slot being accessed. The bit is set to 1 when the CAN module is accessing the message slot and set to 0 when not accessing the message slot.

• For transmit slots

This bit is set to 1 when the message slot has its transmit request accepted. If the message slot failed in arbitration, this bit is cleared to 0 by occurrence of a CAN bus error or completion of transmission.

• For receive slots

This bit is set to 1 when the CAN module is receiving a message, with the received message being stored in the message slot. Note that the value read out from the message slot while this bit remains set is indeterminate.

**Bit 2: Overwrite flag (MsgLost)**

This bit is useful for the receive slots, those that are set for reception. This bit is set to 1 when while the message slot contains an unread received message, it is overwritten by a new received message.

This bit is cleared by writing 0 in software.

**Bit 3: Remote frame transmit/receive status flag (RemActive)**

This bit functions differently for slots 0–13 and slots 14, 15.

• For slots 0–13

If the slot is set for remote frame transmission (or reception), this bit is set to 1. Then, when the slot finished transmitting (or receiving) a remote frame, this bit is cleared to 0.

• For slots 14 and 15

The RemActive bit functions differently depending on how the CAN Control Register's BasicCAN (BasicCAN mode) bit is set.

When BasicCAN = 0 (operating normally), if the slot is set for remote frame transmission (or reception), the RemActive bit is set to 1.

When BasicCAN = 1 (operating in BasicCAN mode), the RemActive bit indicates which frame type of message was received. During BasicCAN mode, slots 14 and 15 store the received data whether it be a data frame or a remote frame.

If RemActive = 0, it means that the message stored in the slot is a data frame.

If RemActive = 1, it means that the message stored in the slot is a remote frame.

**Bit 4: Automatic answering disable bit (RspLock)**

This bit is useful for the slots set for remote frame reception, indicating the processing to be performed after receiving a remote frame.

If this bit is set to 0, the slot automatically changes to a transmit slot after receiving a remote frame and the message stored in the slot is transmitted as a data frame.

If this bit is set to 1, the slot stops operating after receiving a remote frame.

Note 1: This bit must always be set to 0 for any slots other than those set for remote frame reception.

**Bit 5: Remote frame set bit (Remote)**

Set this bit to 1 for the message slots that handle a remote frame.

Message slots can be set to handle a remote frame in the following two ways.

• Set to transmit a remote frame and receive a data frame

The message stored in the message slot is transmitted as a remote frame. The slot automatically changes to a data frame receive slot after it finished transmitting.

However, if it receives a data frame before it finishes transmitting a remote frame, the data frame is stored in the message slot and the remote frame is not transmitted.

• Set to receive a remote frame and transmit a data frame

The slot receives a remote frame. The processing to be performed after receiving a remote frame depends on how the RspLock (automatic answering disable) bit is set.

**Bit 6: Receive request bit (RecReq)**

Set this bit to 1 when using any message slot as a receive slot.

Set this bit to 0 when using any message slot as a data frame transmit or remote frame transmit slot.

If the TrmReq (transmit request) bit and RecReq (receive request) bit both are set to 1, the operation of the CAN module is indeterminate.

**Bit 7: Transmit request bit (TrmReq)**

Set this bit to 1 when using any message slot as a transmit slot.

Set this bit to 0 when using any message slot as a data frame receive or remote frame receive slot.

CAN0 slot buffer select register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 | Symbol | Address | When reset (Note 2) |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | C0SBS | $0240_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SBS00 | CAN0 message slot buffer 0 number select bit | b3 b2 b1 b0<br>0 0 0 0 : slot 0<br>0 0 0 1 : slot 1<br>0 0 1 0 : slot 2<br>0 0 1 1 : slot 3<br>0 1 0 0 : slot 4<br>⋮ ⋮ (Note 1)<br>1 1 0 0 : slot 12<br>1 1 0 1 : slot 13<br>1 1 1 0 : slot 14<br>1 1 1 1 : slot 15 | ○ | ○ |
| SBS01 | | | ○ | ○ |
| SBS02 | | | ○ | ○ |
| SBS03 | | | ○ | ○ |
| SBS10 | CAN0 message slot buffer 1 number select bit | b3 b2 b1 b0<br>0 0 0 0 : slot 0<br>0 0 0 1 : slot 1<br>0 0 1 0 : slot 2<br>0 0 1 1 : slot 3<br>0 1 0 0 : slot 4<br>⋮ ⋮ (Note 1)<br>1 1 0 0 : slot 12<br>1 1 0 1 : slot 13<br>1 1 1 0 : slot 14<br>1 1 1 1 : slot 15 | ○ | ○ |
| SBS11 | | | ○ | ○ |
| SBS12 | | | ○ | ○ |
| SBS13 | | | ○ | ○ |

Note 1: There is a total of 16 CAN0 message slots for transmission and reception uses, respectively. Each message slot can be selected for use as a transmit or a receive slot.
Note 2: This applies when the CAN module is supplied with a clock by setting the sleep mode control bit (bit 0 at address $0242_{16}$) to 1 after reset.

**Figure 1.22.26. CAN0 slot buffer select register**

## 21. CAN0 slot buffer select register

**Bits 0-3: CAN0 message slot buffer 0 slot number select bits (SBS0)**

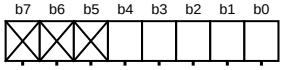
The message slot whose number is selected with these bits appears in CAN0 message slot buffer 0.

**Bits 4-7: CAN0 message slot buffer 1 slot number select bits (SBS1)**

The message slot whose number is selected with these bits appears in CAN0 message slot buffer 1.

The selected message slot can be identified by reading the message slot buffer.
A message written to the message slot buffer is stored in the selected message slot.

## CAN0 message slot buffer i standard ID0 (i=0,1) (Note)

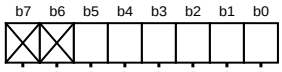



| Symbol | Address | When reset |
|---|---|---|
| C0SLOTi_0(i=0,1) | $01E0_{16}$, $01F0_{16}$ | Indeterminate |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SID6 | Standard ID6 | Message slot j (j=0 to 15) | ○ | ○ |
| SID7 | Standard ID7 | Message slot j (j=0 to 15) | ○ | ○ |
| SID8 | Standard ID8 | Message slot j (j=0 to 15) | ○ | ○ |
| SID9 | Standard ID9 | Message slot j (j=0 to 15) | ○ | ○ |
| SID10 | Standard ID10 | Message slot j (j=0 to 15) | ○ | ○ |
| — | Nothing is assigned. When write, set to "0". When read, their contents are indeterminate. | | — | — |
| — | | | — | — |
| — | | | — | — |

Note: CAN0 message slot j standard ID0 (j=0 to 15) is stored in this register. j is selected with the slot buffer select register.

## CAN0 message slot buffer i standard ID1 (i=0,1) (Note)



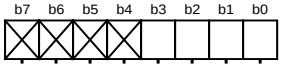

| Symbol | Address | When reset |
|---|---|---|
| C0SLOTi_1(i=0,1) | $01E1_{16}$, $01F1_{16}$ | Indeterminate |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SID0 | Standard ID0 | Message slot j (j=0 to 15) | ○ | ○ |
| SID1 | Standard ID1 | Message slot j (j=0 to 15) | ○ | ○ |
| SID2 | Standard ID2 | Message slot j (j=0 to 15) | ○ | ○ |
| SID3 | Standard ID3 | Message slot j (j=0 to 15) | ○ | ○ |
| SID4 | Standard ID4 | Message slot j (j=0 to 15) | ○ | ○ |
| SID5 | Standard ID5 | Message slot j (j=0 to 15) | ○ | ○ |
| — | Nothing is assigned. When write, set to "0". When read, their contents are indeterminate. | | — | — |
| — | | | — | — |

Note: CAN0 message slot j standard ID1 (j=0 to 15) is stored in this register. j is selected with the slot buffer select register.

**Figure 1.22.27. CAN0 message slot buffer i standard ID0 and ID1**

## CAN0 message slot buffer i extend ID0 (i=0,1) (Note 1, 2)

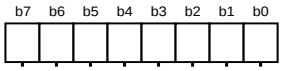



| Symbol | Address | When reset |
|---|---|---|
| C0SLOTi_2(i=0,1) | $01E2_{16}$, $01F2_{16}$ | Indeterminate |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| EID14 | Extended ID14 | Message slot j (j=0 to 15) | ○ | ○ |
| EID15 | Extended ID15 | Message slot j (j=0 to 15) | ○ | ○ |
| EID16 | Extended ID16 | Message slot j (j=0 to 15) | ○ | ○ |
| EID17 | Extended ID17 | Message slot j (j=0 to 15) | ○ | ○ |
| — | Nothing is assigned. When write, set to "0". When read, their contents are indeterminate. | | — | — |
| — | | | — | — |
| — | | | — | — |
| — | | | — | — |

Note 1: When receive slot is standard ID format, EID bits are indeterminate when saving received data.
Note 2: CAN0 message slot j extend ID0 (j=0 to 15) is stored in this register. j is selected with the slot buffer select register.

## CAN0 message slot buffer i extend ID1 (i=0,1) (Note 1,2)



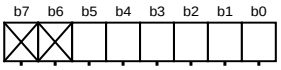

| Symbol | Address | When reset |
|---|---|---|
| C0SLOTi_3(i=0,1) | $01E3_{16}$, $01F3_{16}$ | Indeterminate |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| EID6 | Extended ID6 | Message slot j (j=0 to 15) | ○ | ○ |
| EID7 | Extended ID7 | Message slot j (j=0 to 15) | ○ | ○ |
| EID8 | Extended ID8 | Message slot j (j=0 to 15) | ○ | ○ |
| EID9 | Extended ID9 | Message slot j (j=0 to 15) | ○ | ○ |
| EID10 | Extended ID10 | Message slot j (j=0 to 15) | ○ | ○ |
| EID11 | Extended ID11 | Message slot j (j=0 to 15) | ○ | ○ |
| EID12 | Extended ID12 | Message slot j (j=0 to 15) | ○ | ○ |
| EID13 | Extended ID13 | Message slot j (j=0 to 15) | ○ | ○ |

Note 1: When receive slot is standard ID format, EID bits are indeterminate when saving received data.
Note 2: CAN0 message slot j extend ID1 (j=0 to 15) is stored in this register. j is selected with the slot buffer select register.

**Figure 1.22.28. CAN0 message slot buffer i extended ID0 and ID1**

## CAN0 message slot buffer i extend ID2 (i=0,1) (Note 1,2)

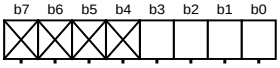



| Symbol | Address | When reset |
|---|---|---|
| C0SLOTi_4(i=0,1) | $01E4_{16}$, $01F4_{16}$ | Indeterminate |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| EID0 | Extended ID0 | Message slot j (j=0 to 15) | ○ | ○ |
| EID1 | Extended ID1 | Message slot j (j=0 to 15) | ○ | ○ |
| EID2 | Extended ID2 | Message slot j (j=0 to 15) | ○ | ○ |
| EID3 | Extended ID3 | Message slot j (j=0 to 15) | ○ | ○ |
| EID4 | Extended ID4 | Message slot j (j=0 to 15) | ○ | ○ |
| EID5 | Extended ID5 | Message slot j (j=0 to 15) | ○ | ○ |
| —— | Nothing is assigned. When write, set to "0". When read, their contents are indeterminate. | | — | — |
| —— | | | — | — |

Note 1: When receive slot is standard ID format, EID bits are indeterminate when saving received data.
Note 2: CAN0 message slot j extend ID2 (j=0 to 15) is stored in this register. j is selected with the slot buffer select register.

## CAN0 message slot buffer i data length code (i=0,1)(Note)




| Symbol | Address | When reset |
|---|---|---|
| C0SLOTi_5(i=0,1) | $01E5_{16}$, $01F5_{16}$ | Indeterminate |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DLC0 | Data length set bit | Message slot j (j=0 to 15) | ○ | ○ |
| DLC1 | | | ○ | ○ |
| DLC2 | | | ○ | ○ |
| DLC3 | | | ○ | ○ |
| —— | Nothing is assigned. When write, set to "0". When read, their contents are indeterminate. | | — | — |
| —— | | | — | — |
| —— | | | — | — |
| —— | | | — | — |

Note : CAN0 message slot j data length code (j=0 to 15) is stored in this register. j is selected with the slot buffer select register.

**Figure 1.22.29. CAN0 message slot buffer i extended ID2 and CAN0 message slot buffer i data lengthcode**

## CAN0 message slot buffer i data m (i=0,1 m=0 to 7) (Note)

b7 b0

| Symbol | Address | When reset |
|---|---|---|
| C0SLOT0_n(n=m+6,m=0 to 3) | 01E616, 01E716, 01E816, 01E916 | Indeterminate |
| C0SLOT0_n(n=m+6,m=4 to 7) | 01EA16, 01EB16, 01EC16, 01ED16 | Indeterminate |
| C0SLOT1_n(n=m+6,m=0 to 3) | 01F616, 01F716, 01F816, 01F916 | Indeterminate |
| C0SLOT1_n(n=m+6,m=4 to 7) | 01FA16, 01FB16, 01FC16, 01FD16 | Indeterminate |

| Function | Setting range | R | W |
|---|---|---|---|
| Message slot j data m (j=0 to 15, m=0 to 7) | $00_{16}$ to $FF_{16}$ | ○ | ○ |

Note: j is selected with the slot buffer select register

## CAN0 message slot buffer i time stamp high (i=0,1) (Note)

b7 b0

| Symbol | Address | When reset |
|---|---|---|
| C0SLOTi_14(i=0,1) | $01EE_{16}$, $01FE_{16}$ | Indeterminate |

| Function | Setting range | R | W |
|---|---|---|---|
| Message slot j time stamp high (j=0 to 15) | $00_{16}$ to $FF_{16}$ | ○ | ○ |

Note : CAN0 message slot j time stamp high (j=0 to 15) is stored in this register. j is selected with the slot buffer select register.

## CAN0 message slot buffer i time stamp low (i=0,1) (Note)

b7 b0

| Symbol | Address | When reset |
|---|---|---|
| C0SLOTi_15(i=0,1) | $01EF_{16}$, $01FF_{16}$ | Indeterminate |

| Function | Setting range | R | W |
|---|---|---|---|
| Message slot j time stamp low (j=0 to 15) | $00_{16}$ to $FF_{16}$ | ○ | ○ |

Note : CAN0 message slot j time stamp low (j=0 to 15) is stored in this register. j is selected with the slot buffer select register.

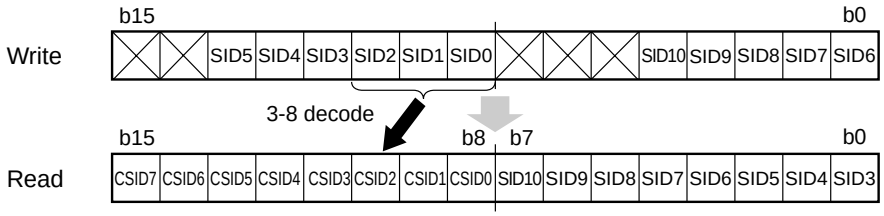
**Figure 1.22.30. CAN0 message slot buffer i data m and CAN0 message slot buffer i time stamp**

## CAN0 acceptance filter support register

| Symbol | Address | When reset (Note) |
|---|---|---|
| C0AFS | $0245_{16}$,$0244_{16}$ | $0100_{16}$ |

| Function | Setting range | R | W |
|---|---|---|---|
| Produces receive ID determination data | $0000_{16}$ to $FFFF_{16}$ | ○ | ○ |

Note: This applies when the CAN module is supplied with a clock by setting the sleep mode control bit (bit 0 at address $0242_{16}$) to 1 after reset.

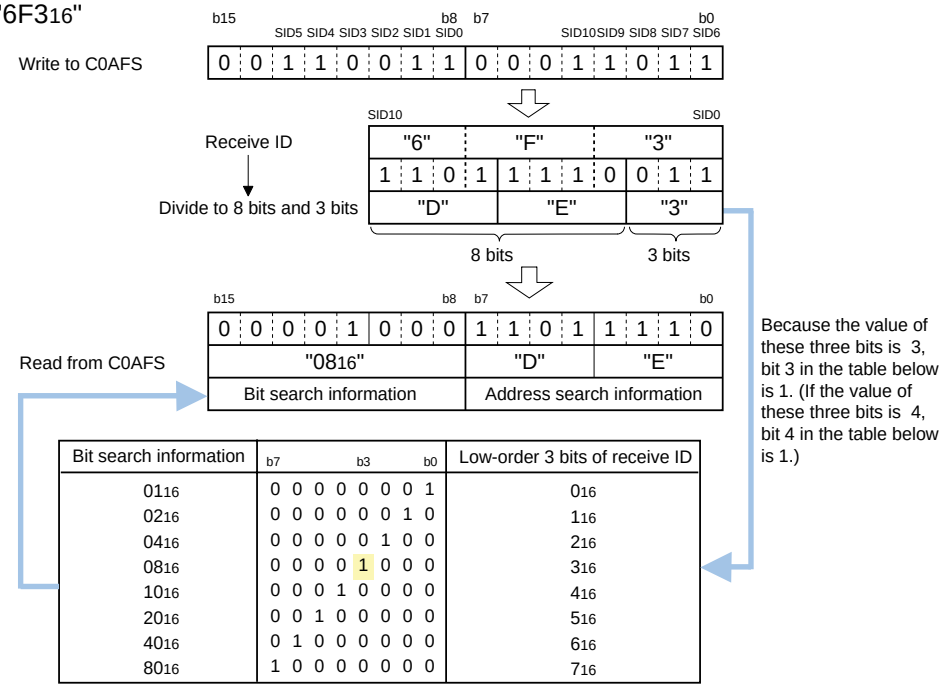



From the receive ID of the standard format, this register produces data with which to search the data table. After searching the table using this data, the CAN module determines whether the receive ID is valid or not.

| | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Top+$00_{16}$ | $007_{16}$ "0" | $006_{16}$ "0" | $005_{16}$ "0" | $004_{16}$ "0" | $003_{16}$ "0" | $002_{16}$ "0" | $001_{16}$ "1" | $000_{16}$ "0" |
| Top+$01_{16}$ | $00F_{16}$ "1" | $00E_{16}$ "0" | $00D_{16}$ "0" | $00C_{16}$ "0" | $00B_{16}$ "0" | $00A_{16}$ "0" | $009_{16}$ "0" | $008_{16}$ "0" |
| | ≈ | | | | | | | ≈ |
| Top+$DE_{16}$ | $6F7_{16}$ "0" | $6F6_{16}$ "0" | $6F5_{16}$ "0" | $6F4_{16}$ "0" | $6F3_{16}$ "1" | $6F2_{16}$ "0" | $6F1_{16}$ "0" | $6F0_{16}$ "0" |
| | ≈ | | | | | | | ≈ |
| Top+$FE_{16}$ | $7F7_{16}$ "0" | $7F6_{16}$ "0" | $7F5_{16}$ "0" | $7F4_{16}$ "0" | $7F3_{16}$ "0" | $7F2_{16}$ "0" | $7F1_{16}$ "0" | $7F0_{16}$ "1" |
| Top+$FF_{16}$ | $7FF_{16}$ "0" | $7FE_{16}$ "0" | $7FD_{16}$ "1" | $7FC_{16}$ "0" | $7FB_{16}$ "0" | $7FA_{16}$ "0" | $7F9_{16}$ "0" | $7F8_{16}$ "0" |

Address search information — Bit search information

When receive ID is "$6F3_{16}$"




| Bit search information | b7 b3 b0 | Low-order 3 bits of receive ID |
|---|---|---|
| $01_{16}$ | 0 0 0 0 0 0 0 1 | $0_{16}$ |
| $02_{16}$ | 0 0 0 0 0 0 1 0 | $1_{16}$ |
| $04_{16}$ | 0 0 0 0 0 1 0 0 | $2_{16}$ |
| $08_{16}$ | 0 0 0 0 1 0 0 0 | $3_{16}$ |
| $10_{16}$ | 0 0 0 1 0 0 0 0 | $4_{16}$ |
| $20_{16}$ | 0 0 1 0 0 0 0 0 | $5_{16}$ |
| $40_{16}$ | 0 1 0 0 0 0 0 0 | $6_{16}$ |
| $80_{16}$ | 1 0 0 0 0 0 0 0 | $7_{16}$ |

**Figure 1.22.31. CAN0 acceptance filter support register**

## Intelligent I/O

Intelligent I/O uses multifunctional I/O ports for time measurement, waveform generation, clock-synchronous/asynchronous (UART) serial I/O, IE bus (Note) communications, HDLC data processing and more. A single Intelligent I/O group comes with one 16-bit base timer for free running, eight 16-bit registers for time measurement and waveform generation, and two shift registers for 8-bit and 16-bit communications.
The M32C/83 has four internal Intelligent I/O groups. Table 1.23.1 lists functions by group.

**Table 1.23.1. List of functions of intelligent I/O**

| Function | Group 0 | Group 1 | Group 2 | Group 3 | Group 0,1 cascaded |
|---|---|---|---|---|---|
| Configuration | | | | | |
| •Base timer | 1 | 1 | 1 | 1 | 1 |
| •TM | 4ch(2ch) | – | – | – | – |
| •TM/WG register (shared) | 4chs(1ch) | 4chs(2chs) | – | – | 8chs(3chs) |
| •WG register | – | 4chs(1ch) | 8chs | 8chs(3chs) | 8chs(2chs) |
| •Communication shift register | 8bits X 2chs | 8bits X 2chs | 8bits X 2chs | – | – |
| Time measurement functions | Max. 8chs (3chs) | Max. 4chs (2chs) | – | – | Max. 8chs (3chs) |
| •Digital filter function | √ | √ | – | – | √ |
| •Trigger input prescale function | 2chs | 2chs | – | – | 2chs |
| •Gate function for trigger input | 2chs | 2chs | – | – | 2chs |
| WG function | Max. 4chs (1ch) | Max. 8chs (3chs) | Max. 8chs (3chs) | Max. 8chs (2chs) | Max. 8chs (1ch) |
| •Single phase waveform output | √ | √ | √ | √ | √ |
| •Phase delayed waveform output | √ | √ | √ | √ | √ |
| •Set/reset waveform output | √ | √ | √ | √ | √ |
| •Bit modulation PWM output | – | – | √ | √ | – |
| •Real-time port output | – | – | √ | √ | – |
| •Parallel real-time port output | – | – | √ | √ | – |
| Communication functions | | | | | |
| •Bit length | 8 bits fixed | 8 bits fixed | Variable length | – | – |
| •Communication mode | | | | | |
| 1. Clock synchronous serial I/O | √ | √ | √ | – | – |
| 2. UART | √ | √ | – | – | – |
| 3. HDLC data processing | √ | √ | – | – | – |
| 4. IE Bus sub set | – | – | √ | – | – |

Note 1: IE Bus is a trademark of NEC.

Note 2: 100-pin specification are in parentheses.

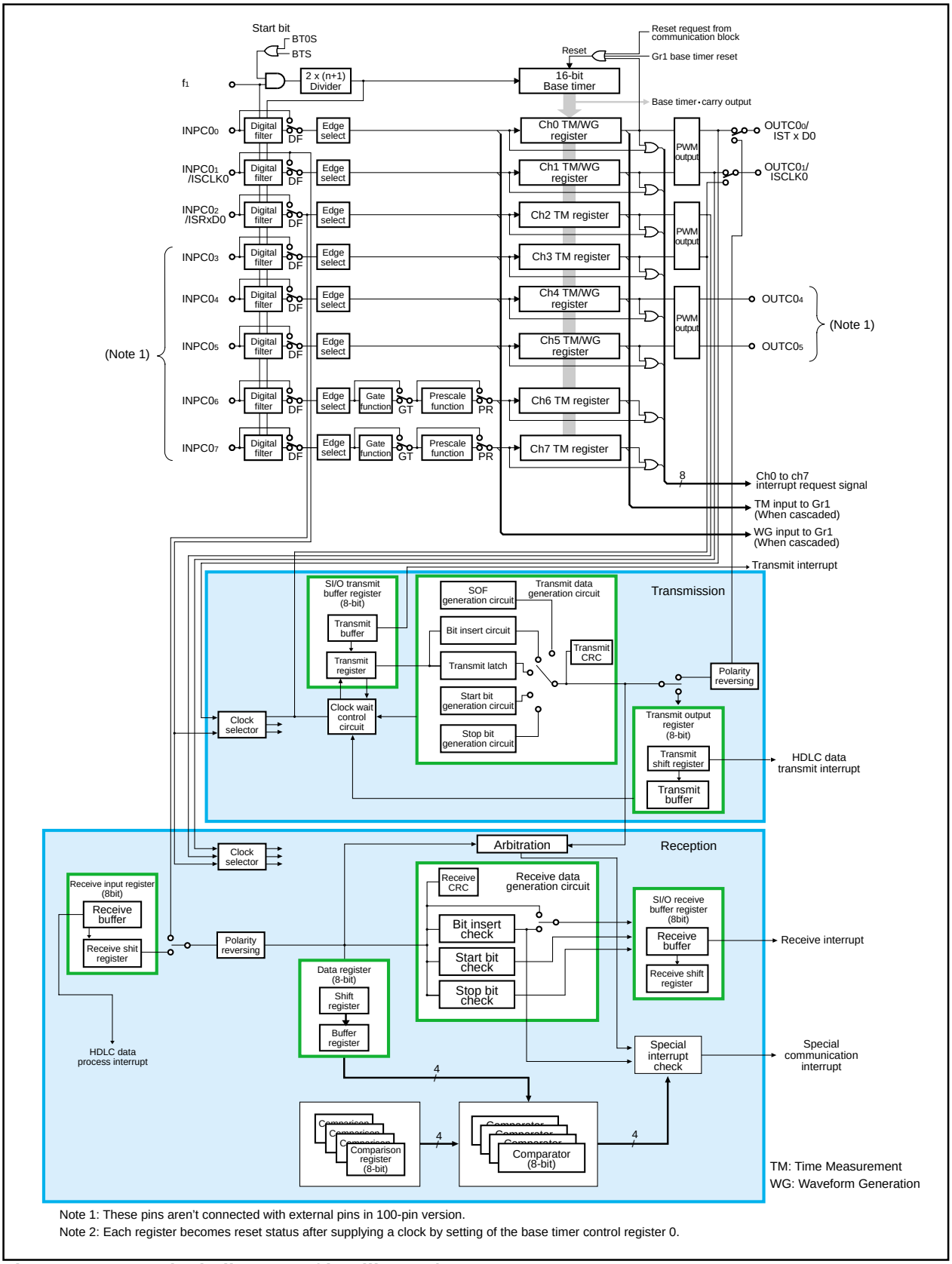
√ : Present

– : Not present

TM:Time Measurement

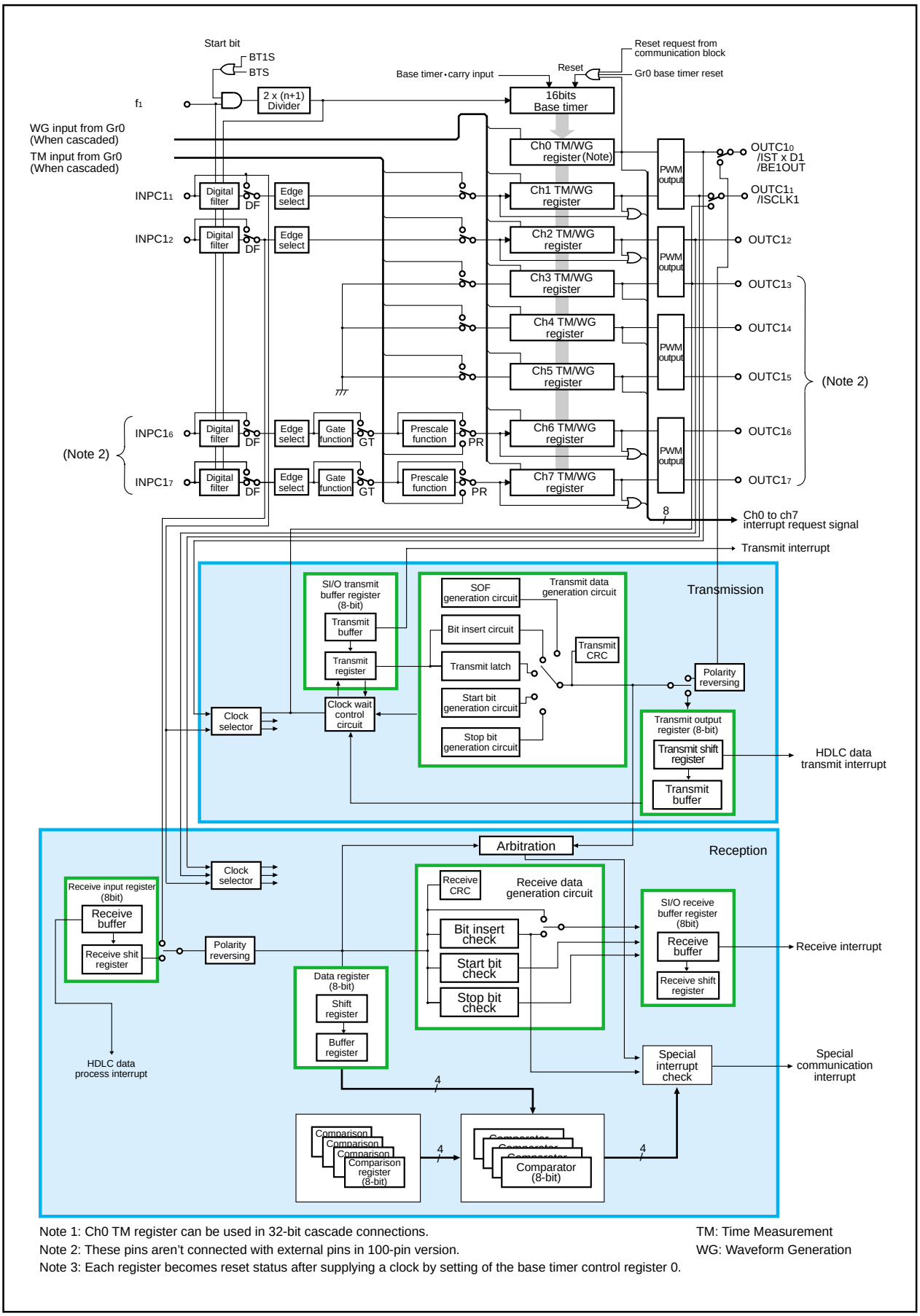
WG:Waveform Genaration

Block diagrams for groups 0 to 3 are given in Figures 1.23.1 to 1.23.4.




Note 1: These pins aren't connected with external pins in 100-pin version.
Note 2: Each register becomes reset status after supplying a clock by setting of the base timer control register 0.

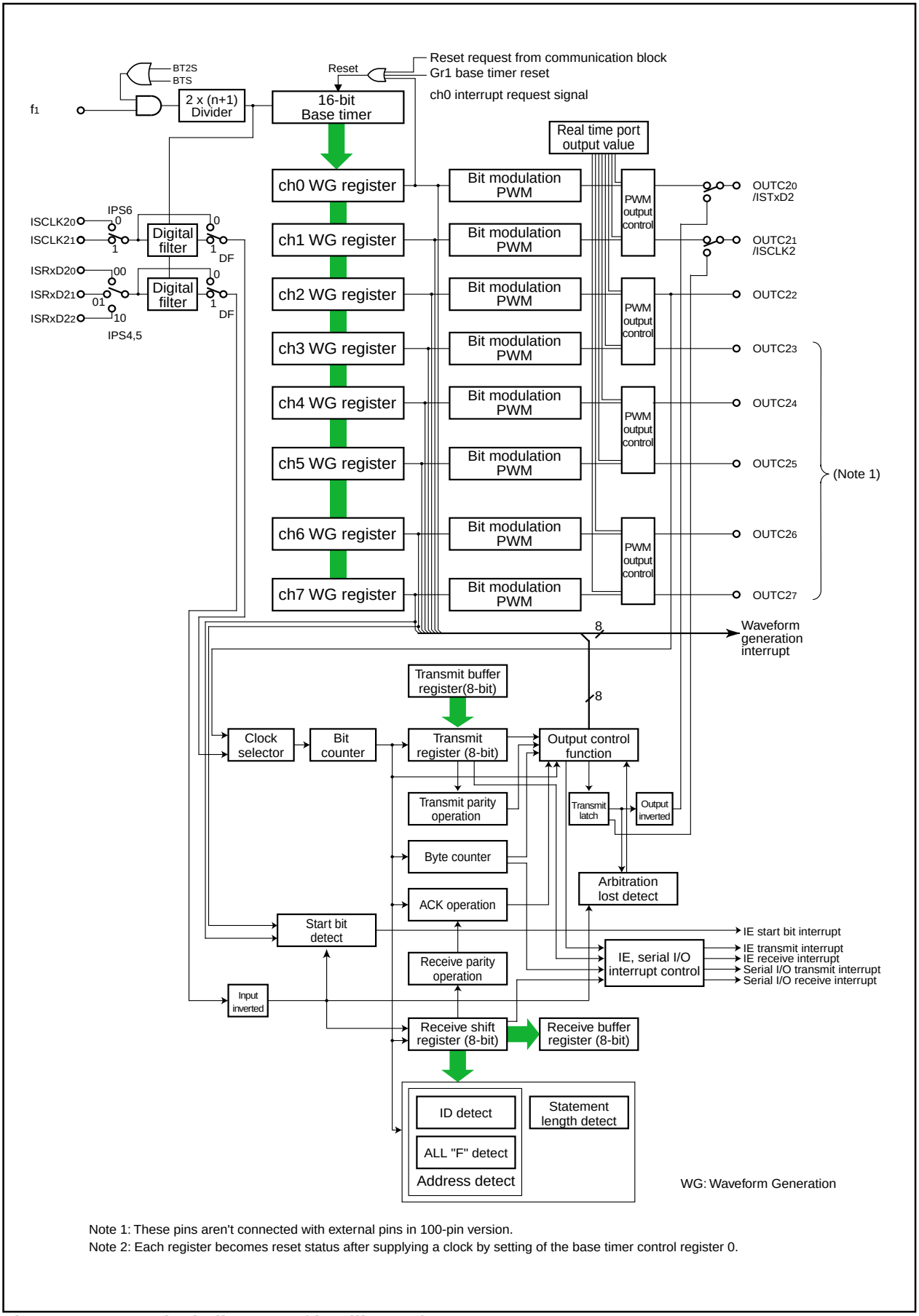
**Figure 1. 23. 1. Block diagram of intelligent I/O group 0**




Note 1: Ch0 TM register can be used in 32-bit cascade connections.
Note 2: These pins aren't connected with external pins in 100-pin version.
Note 3: Each register becomes reset status after supplying a clock by setting of the base timer control register 0.

TM: Time Measurement
WG: Waveform Generation

**Figure 1. 23. 2. Block diagram of intelligent I/O group 1**




Note 1: These pins aren't connected with external pins in 100-pin version.
Note 2: Each register becomes reset status after supplying a clock by setting of the base timer control register 0.

**Figure 1. 23. 3. Block diagram of intelligent I/O group 2**

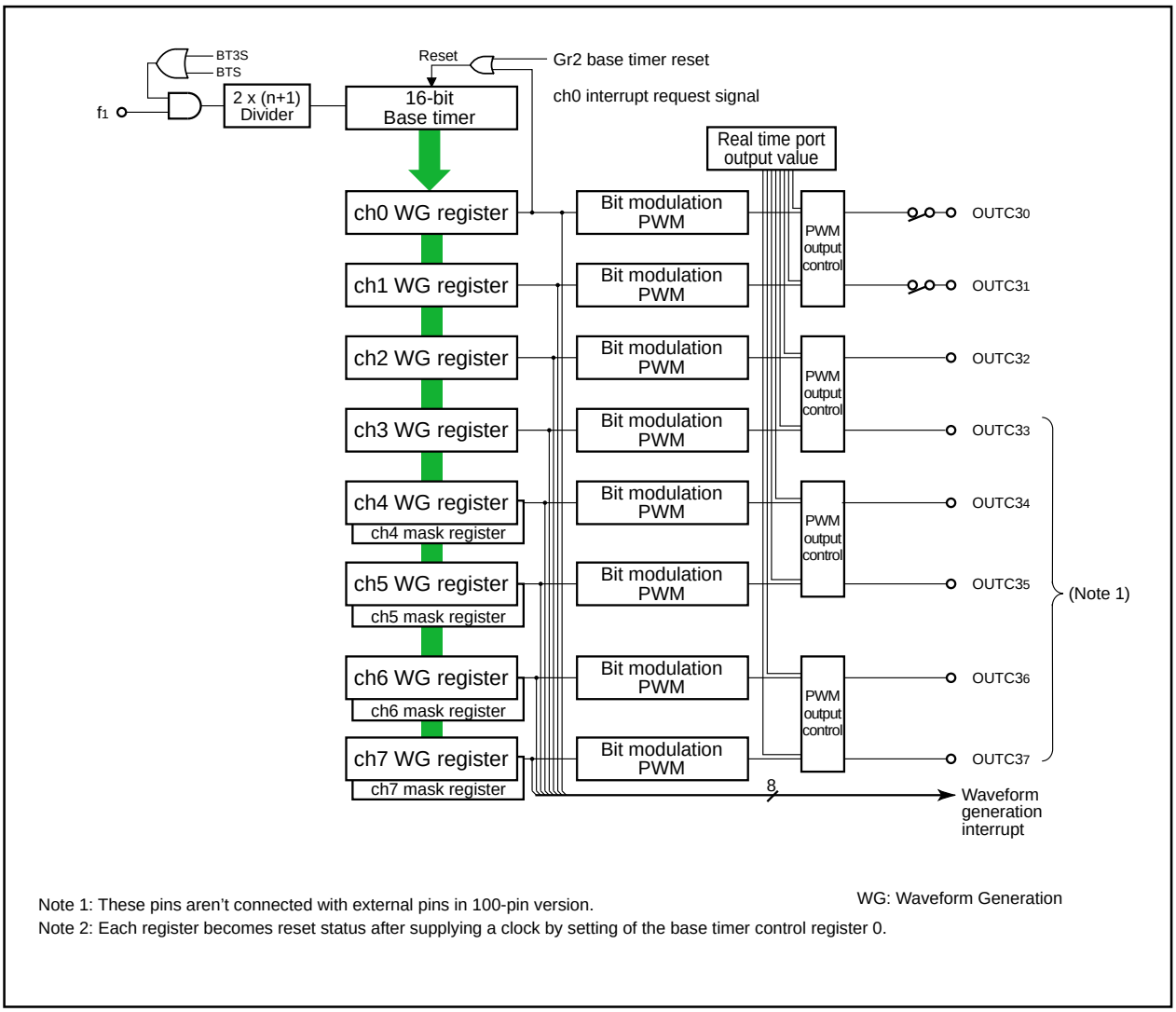
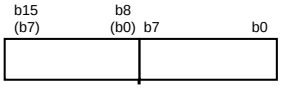



**Figure 1. 23 . 4. Block diagram of intelligent I/O group 3**

## Base timer (group 0 to 3)

The internally generated count source is a free run source. Base timer specifications are given in Table 1.23.2, base timer registers in Figures 1.23.5 to 1.23.9 and a block diagram in Figure 1.23.10.

### Group i base timer register (i=0 to 3)

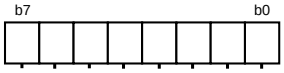



| Symbol | Address | When reset |
|---|---|---|
| GiBT (i=0,1) | $00E1_{16}$, $00E0_{16}$, $0121_{16}$, $0120_{16}$ | Indeterminate |
| GiBT (i=2,3) | $0161_{16}$, $0160_{16}$, $01A1_{16}$, $01A0_{16}$ | Indeterminate |

| Function | Setting range | R | W |
|---|---|---|---|
| Count value of the 16-bit base timer (Note) | $0000_{16}$ to $FFFF_{16}$ | ○ | ○ |

Note : When this register is read while the base timer is being reset, the value is indeterminate.
The counter value is read if the register is output while the timer is running.
Written value while the base timer is being reset is ignored. The count starts from "$0000_{16}$" after starting the base timer. When writing value while the base timer is operating, the count starts from the written value immediately after written.

### Group i base timer control register 0 (i=0 to 3) (Note)



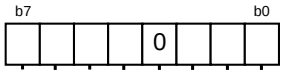

| Symbol | Address | When reset |
|---|---|---|
| GiBCR0 (i=0 to 3) | $00E2_{16}$, $0122_{16}$, $0162_{16}$, $01A2_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| BCK0 | Count source select bit | b1b0<br>0 0: Clock stop<br>0 1: Must not be set<br>1 0: Must not be set<br>1 1: $f_1$ | ○ | ○ |
| BCK1 | | | ○ | ○ |
| DIV0 | Count source division ratio select bit | Divides the count source by 2x(n + 1) for a setting value n (n = 0 to 31).<br>b6 b5 b4 b3 b2<br>(n=0) 0 0 0 0 0 : Division by 2<br>(n=1) 0 0 0 0 1 : Division by 4<br>(n=2) 0 0 0 1 0 : Division by 6<br>:<br>(n=30) 1 1 1 1 0 : Division by 62<br>(n=31) 1 1 1 1 1 : No division | ○ | ○ |
| DIV1 | | | ○ | ○ |
| DIV2 | | | ○ | ○ |
| DIV3 | | | ○ | ○ |
| DIV4 | | | ○ | ○ |
| IT | Base timer interrupt select bit | 0 : Bit 15 overflow<br>1 : Bit 14 overflow | ○ | ○ |

Note: In cascade connections, set the same value to the base timer control register 0 of groups 0 and 1.

**Figure 1. 23. 5. Base timer-related register (1)**

## Group i base timer control register 1 (i=0,1)




| Symbol | Address | When reset |
|---|---|---|
| GiBCR1 (i=0,1) | $00E3_{16}$, $0123_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| RST0 | Base timer reset cause select bit 0 | 0: Synchronizes the base timer reset without resetting the timer<br>1: Synchronizes the base timer reset with resetting the timer (Note1) | ○ | ○ |
| RST1 | Base timer reset cause select bit 1 | 0: Does not reset the base timer when it matches WG register ch0<br>1: Reset the base timer when it matches WG register ch0 (Note 2) | ○ | ○ |
| RST2 | Base timer reset cause select bit 2 | 0: Does not reset the base timer when input to the $\overline{INT}$ pin is "L" level<br>1: Reset the base timer when input to the $\overline{INT}$ pin is "L" level (Note 3) | ○ | ○ |
| —— | Reserved bit | Must always set to "0". | ○ | ○ |
| BTS | Base timer start bit | 0: Base timer reset<br>1: Base timer count start | ○ | ○ |
| UD0 | Up / down control bit | b6b5<br>0 0 : Up mode<br>0 1 : Up / down mode (triangle wave)<br>1 0 : Two-phase pulse signal processing mode (Note 4)<br>1 1 : Must not be set | ○ | ○ |
| UD1 | | | ○ | ○ |
| CAS | Groups 0 and 1 cascaded function select bit | 0: 16-bit TM / WG function<br>1: 32-bit TM / WG function (Note 5) | ○ | ○ |

Note 1: With group 0, reset synchronizing with group 1 base timer. With group 1, reset synchronizing with group 0 base timer.
Note 2: The base timer is reset 2 clock cycles after it matches waveform generation register ch0.
Note 3: With group 0, the base timer is reset when "L" level is input to $\overline{INT0}$. With group 1, it resets when "L" level is input to $\overline{INT1}$.
Note 4: Operation of this mode is equal to Timer A two-phase pulse signal processing except count value.
Note 5: In cascade connections, set to "$81_{16}$" for group 0 base timer control register 1. Set to "$1000\ 0XX0_2$" for group 1 base timer control register 1.

**Figure 1. 23. 6. Base timer-related register (2)**

## Group 2 base timer control register 1

| b7 | | | | | | | b0 |
|---|---|---|---|---|---|---|---|
| | 0 | 0 | | 0 | | | |

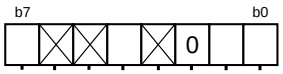
| Symbol | Address | When reset |
|---|---|---|
| G2BCR1 | $0163_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| RST0 | Base timer reset cause select bit 0 | 0 : Synchronizes the group 1 base timer reset without resetting the timer<br>1 : Synchronizes the group 1 base timer reset with resetting the timer | ○ | ○ |
| RST1 | Base timer reset cause select bit 1 | 0 : Does not reset the base timer when it matches WG register ch0<br>1 : Reset the base timer when it matches WG register ch0 (Note) | ○ | ○ |
| RST2 | Base timer reset cause select bit 2 | 0 : Does not reset the base timer when a reset is requested from the communication additional circuit<br>1 : Reset the base timer when a reset is requested from the communication additional circuit | ○ | ○ |
| RST3 | Reserve bit | Must always set to "0". | ○ | ○ |
| BTS | Base timer start bit | 0 : Base timer reset<br>1 : Base timer count start | ○ | ○ |
| UD0 | Reserve bit | Must always set to "0". | ○ | ○ |
| UD1 | | | ○ | ○ |
| PRP | Parallel real-time port function select bit | 0 : Not use<br>1 : Use | ○ | ○ |

Note : The base timer is reset 2 clock cycles after it matches waveform generation register ch0.

**Figure 1. 23. 7. Base timer-related register (3)**

## Group 3 base timer control register 1




Symbol: G3BCR1 | Address: $01A3_{16}$ | When reset: $0XX0\ X000_2$

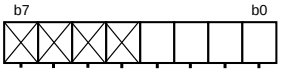
| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| RST0 | Base timer reset cause select bit 0 | 0 : Synchronizes the base timer 2 reset without resetting the timer<br>1 : Synchronizes the base timer 2 reset with resetting the timer | ○ | ○ |
| RST1 | Base timer reset cause select bit 1 | 0 : Does not reset the base timer when it matches WG register ch0<br>1 : Reset the base timer when it matches WG register ch0 (Note) | ○ | ○ |
| —— | Reserved bit | Must always set to "0". | ○ | ○ |
| —— | Nothing is assigned. When write, set to "0". When read, the content is indeterminate. | | — | — |
| BTS | Base timer start bit | 0 : Base timer reset<br>1 : Base timer count start | ○ | ○ |
| —— | Nothing is assigned. When write, set to "0". When read, their contents are indeterminate. | | — | — |
| —— | | | — | — |
| PRP | Parallel real-time port function select bit | 0 : Not use<br>1 : Use | ○ | ○ |

WG: Waveform Generation

Note : The base timer is reset 2 clock cycles after it matches waveform generation register ch0.

**Figure 1. 23. 8. Base timer-related register (4)**

## Base timer start register (Note 1, 2)




| Symbol | Address | When reset |
|---|---|---|
| BSTR | $0164_{16}$ | XXXX $0000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| BT0S | Group 0 base timer start bit | 0 : Base timer reset<br>1 : Base timer count start | ○ | ○ |
| BT1S | Group 1 base timer start bit | 0 : Base timer reset<br>1 : Base timer count start | ○ | ○ |
| BT2S | Group 2 base timer start bit (Note 2) | 0 : Base timer reset<br>1 : Base timer count start | ○ | ○ |
| BT3S | Group 3 base timer start bit | 0 : Base timer reset<br>1 : Base timer count start | ○ | ○ |
| ____ | Nothing is assigned. When write, set "0".<br>When read, their contents are indeterminate. | | _ | _ |

Note 1: When starting multiple base timer with this register at the same time (including group 0 and 1 cascaded connection), do the followings. Do not need when starting base timer individually.
* Set the same values to each group's base timer clock division ratio ( bits 6 to 0 of base timer control register).
* When changing base timer clock division ratio, start base timer twice with the following procedure.
(1) Start each group base timer using the base timer start register.
(2) After one clock, stop base timer by setting "$00_{16}$" to base timer start register.
(3) Further after one clock, restart each group base timer using the base timer start register.

Note 2: This register is enabled after when group 2 base timer control register 0 is set.

**Figure 1. 23. 9. Base timer-related register (5)**

**Table 1. 23.2. Base timer specifications**

| Item | | Specifications |
|---|---|---|
| Count source | | f1/2(n+1)<br>n: Set by count source division ratio select bit<br>(n=0 to 31, however, please note when n=31, the counter source is not divided.) |
| Count operation | | Up count / down count |
| Count start condition | | Writes "1" for the start bit in the base timer start register or base timer control register 1. (After writing the bit, the base timer resets to "0000$_{16}$" and counting starts.) |
| Count stop condition | | Writes "0" for both the start bit in the base timer start register and base timer control register 1. |
| Count reset condition | Group 0, 1 | (1) Synchronizes and resets the base timer with that of another group.<br>Group 0: Synchronizes base timer reset with the group 1 base timer.<br>Group 1: Synchronizes base timer reset with the group 0 base timer.<br>(2) Matches the value of the base timer to the value of WG register 0.<br>(3) Input "L" to INT pin<br>Group 0 : INT 0 pin Group 1 : INT 1 pin<br>The above 3 factors can be used in conjunction with one another. |
| | Group 2, 3 | (1) Synchronizes and resets the base timer with that of another group.<br>Group 2: Synchronizes base timer reset with the group 1 base timer.<br>Group 3: Synchronizes base timer reset with the group 2 base timer.<br>(2) Matches the value of the base timer to the value of WG register 0.<br>(3) Reset request from communication additional circuit (group 2 only)<br>The above 3 factors can be used in conjunction with one another. |
| Interrupt request generation timing | | When bit 14 or bit 15 overflows |
| Read from timer | | •When the base timer is running<br>The count is output when the base timer is read.<br>•When the base timer not running<br>An undefined value is output when the base timer is read. |
| Write to timer | | Possible. Values that are written while the base timer is resetting are ignored. If values are written while the base timer is running, counting continues after the values are written. |

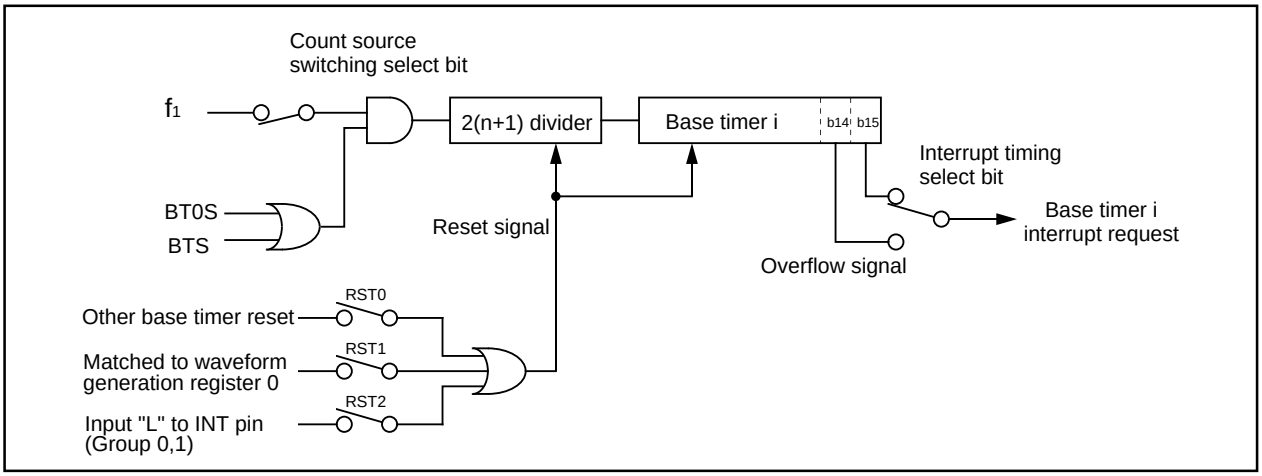



**Figure 1. 23.10. Base timer block diagram**

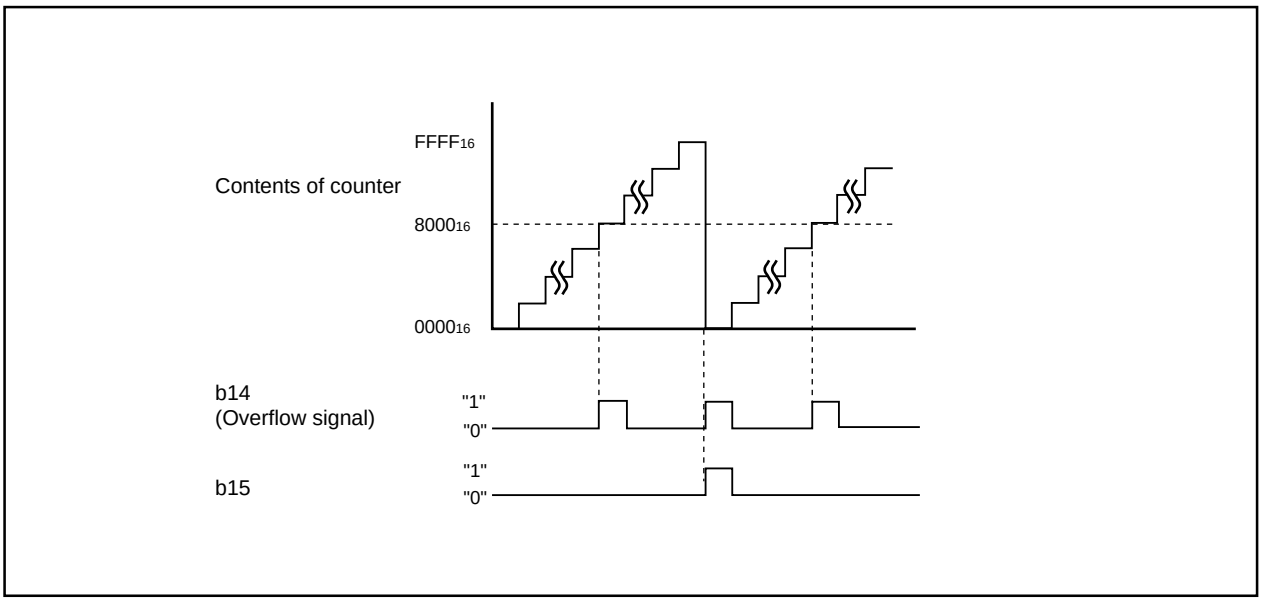
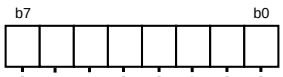



**Figure 1. 23.11. Operation timing of base timer**

## Time measurement (group 0 and 1)

Synchronizes external trigger input and stores the base timer value in the time measurement register j. Specifications for the time measurement function are given in Table 1.23.3, the time measurement control registers in Figures 1.23.12 to 1.23.13, and the operating timing of the time measurement function in Figure 1.23.14 and 15.

Group i time measurement control register j (i=0,1/j=0 to 7) (Note 1)




| Symbol | Address | When reset |
|---|---|---|
| GiTMCRj(i=0/j=0 to 3) | $00D8_{16}$, $00D9_{16}$, $00DA_{16}$, $00DB_{16}$ | $00_{16}$ |
| GiTMCRj(i=0/j=4 to 7) | $00DC_{16}$, $00DD_{16}$, $00DE_{16}$, $00DF_{16}$ | $00_{16}$ |
| GiTMCRj(i=1/j=1, 2) | $0119_{16}$, $011A_{16}$ | $00_{16}$ |
| GiTMCRj(i=1/j=6, 7) | $011E_{16}$, $011F_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CTS0 | Time measurement trigger select bit | b1 b0<br>0 0 : No time measurement<br>0 1 : Rising edge<br>1 0 : Falling edge<br>1 1 : Both edges | ○ | ○ |
| CTS1 | | | ○ | ○ |
| DF0 | Digital filter function select bit | b3 b2<br>0 0 : No digital filter<br>0 1 : Must not be set<br>1 0 : Base timer clock<br>1 1 : $f_1$ | ○ | ○ |
| DF1 | | | ○ | ○ |
| GT | Gate function select bit (Note 2, 4) | 0 : Gate function not used<br>1 : Gate function used | ○ | ○ |
| GOC | Gate function release select bit (Note 2, 3) | 0 : No effect<br>1 : Release the gate when it matches WG register | ○ | ○ |
| GSC | Gate function release bit (Note 2, 3) | 0 : No effect<br>1 : Gate released | ○ | ○ |
| PR | Prescaler function select bit (Note 2) | 0 : Not used<br>1 : Used | ○ | ○ |

WG: Waveform Generation

Note 1: The 16-bit time measurement function is available for 8 channels (ch0 to 7) with group 0 and 4 channels (ch1, 2, 6 and 7) with group 1. When using the 16-bit time measurement function, use the time measurement register values for ch0, 3, 4 and 5 of group 1 as they are, or, if writing values, write "$00_{16}$". The 32-bit time measurement function can be used with 8 channels (ch0 to 7) by linking groups 0 and 1. When using the 32-bit time measurement function, write the same value for time measurement registers of similar channels in groups 0 and 1.

Note 2: These functions are available only for time measurement ch6 and 7 (time measurement registers 6 and 7). For ch0 to 5, set "0" for bits 4 to 7 of the time measurement register.

Note 3: These bits are valid only when "1" is set for the gate function select bit.

Note 4: The gate function cannot be used at the same time as the 32-bit time measurement function.

**Figure 1. 23. 12. Time measurement-related register (1)**

## Group i time measurement prescale register j (i=0,1/j=6,7)

b7 b0

| Symbol | Address | When reset |
|---|---|---|
| GiTPRj(i=0/j=6, 7) | $00E4_{16}$, $00E5_{16}$ | $00_{16}$ |
| GiTPRj(i=1/j=6, 7) | $0124_{16}$, $0125_{16}$ | $00_{16}$ |

| Function | Setting range | R | W |
|---|---|---|---|
| Prescales time measurement events. (Generates the time measurement request after an n + 1 count.) (Note) | $00_{16}$ to $FF_{16}$ | ○ | ○ |

Note : This function is only built into time measurement ch6 and 7 of Intelligent I/O groups 0 and 1.

## Group i time measurement register j (i=0,1/j=0 to 7)

b15 b8
(b7) (b0) b7 b0

| Symbol | Address | When reset |
|---|---|---|
| GiTMj(i=0/j=0 to 2) | $00C1_{16}$,$00C0_{16}$, $00C3_{16}$,$00C2_{16}$, $00C5_{16}$,$00C4_{16}$ | Indeterminate |
| GiTMj(i=0/j=3 to 5) | $00C7_{16}$,$00C6_{16}$, $00C9_{16}$,$00C8_{16}$, $00CB_{16}$,$00CA_{16}$ | Indeterminate |
| GiTMj(i=0/j=6,7) | $00CD_{16}$,$00CC_{16}$, $00CF_{16}$,$00CE_{16}$ | Indeterminate |
| GiTMj(i=1/j=0 to 2) | $0101_{16}$,$0100_{16}$, $0103_{16}$,$0102_{16}$, $0105_{16}$,$0104_{16}$ | Indeterminate |
| GiTMj(i=1/j=3 to 5) | $0107_{16}$,$0106_{16}$, $0109_{16}$,$0108_{16}$, $010B_{16}$,$010A_{16}$ | Indeterminate |
| GiTMj(i=1/j=6,7) | $010D_{16}$,$010C_{16}$, $010F_{16}$,$010E_{16}$ | Indeterminate |

| Function | Setting range | R | W |
|---|---|---|---|
| When an event occures, the value of the base timer is stored. | ———— | ○ | – |

## Group i function select register (i=0, 1)

b7 b0

| Symbol | Address | When reset |
|---|---|---|
| GiFS (i=0,1) | $00E7_{16}$, $0127_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| FSC0 | Ch0 TM/WG function select bit | Whether the corresponding port functions as TM or WG is selected<br>0 : WG function is selected<br>1 : TM function is selected | ○ | ○ |
| FSC1 | Ch1 TM/WG function select bit | | ○ | ○ |
| FSC2 | Ch2 TM/WG function select bit | | ○ | ○ |
| FSC3 | Ch3 TM/WG function select bit | | ○ | ○ |
| FSC4 | Ch4 TM/WG function select bit | | ○ | ○ |
| FSC5 | Ch5 TM/WG function select bit | | ○ | ○ |
| FSC6 | Ch6 TM/WG function select bit | | ○ | ○ |
| FSC7 | Ch7 TM/WG function select bit | | ○ | ○ |

Note : In group 0, channles 2, 3, 6 and 7 cannot be selected in 16-bit mode. All channel can be selected in 32-bit mode.
In group 1, channles 0 and 3 to 5 cannot be selected in 16-bit mode. All channel can be selected in 32-bit mode.

**Figure 1. 23. 13. Time measurement-related register (2)**

Table 1. 23.3. Specifications of time measurement function

| Item | Specifications |
|---|---|
| Time resolution | t=1/(base timer count source) |
| Trigger input polarity select | •Rising edge •Falling edge •Both edges |
| Measurement start condition (Note) | Write "1" to the function enable bit |
| Measurement stop condition | Write "0" to the function enable bit |
| Time measurement timing | •Prescaler (only ch6 and ch7) : Every the (m+1) trigger input<br>•No prescaler : Every trigger input |
| Interrupt request generation timing | Same timing as time measurement |
| INPC pin function | Trigger input pin<br>(Set the corresponding pin to input with the function select register) |
| Select function | •Digital filter function<br>Pulses will pass when they match either $f_1$ or the base timerclock 3 times .<br>•Prescaler function (only for ch6 and ch7)<br>Counts trigger inputs and measures time by inputting a trigger of +1 the value of the time measurement prescale register.<br>•Gate function (only for ch6 and ch7)<br>Prohibits the reception of trigger inputs after the time measurement starts for the first trigger input. Trigger input is newly enabled when the below conditions are satisfied.<br>(1) When the base timer i matches the value in WG register j<br>(2) When “1” is written for the gate function release bit<br>This bit automatically becomes “0” after the gate function is released. |

Note: On channels where both the time measurement function and waveform output function can be used, select the time measurement function for the function select register (addresses $00E7_{16}$ and $0127_{16}$).

**Table 1. 23.4. List of time measurement channels with prescaler function and gate function**

| Group | Channel | TM register | WG register matehes signal to release gate function |
|---|---|---|---|
| Group 0 | ch6 | TM register 6 | Base timer 0 matches to WG register 4 |
| | ch7 | TM register 7 | Base timer 0 matches to WG register 5 |
| Group 1 | ch6 | TM register 6 | Base timer 1 matches to WG register 4 |
| | ch7 | TM register 7 | Base timer 1 matches to WG register 5 |

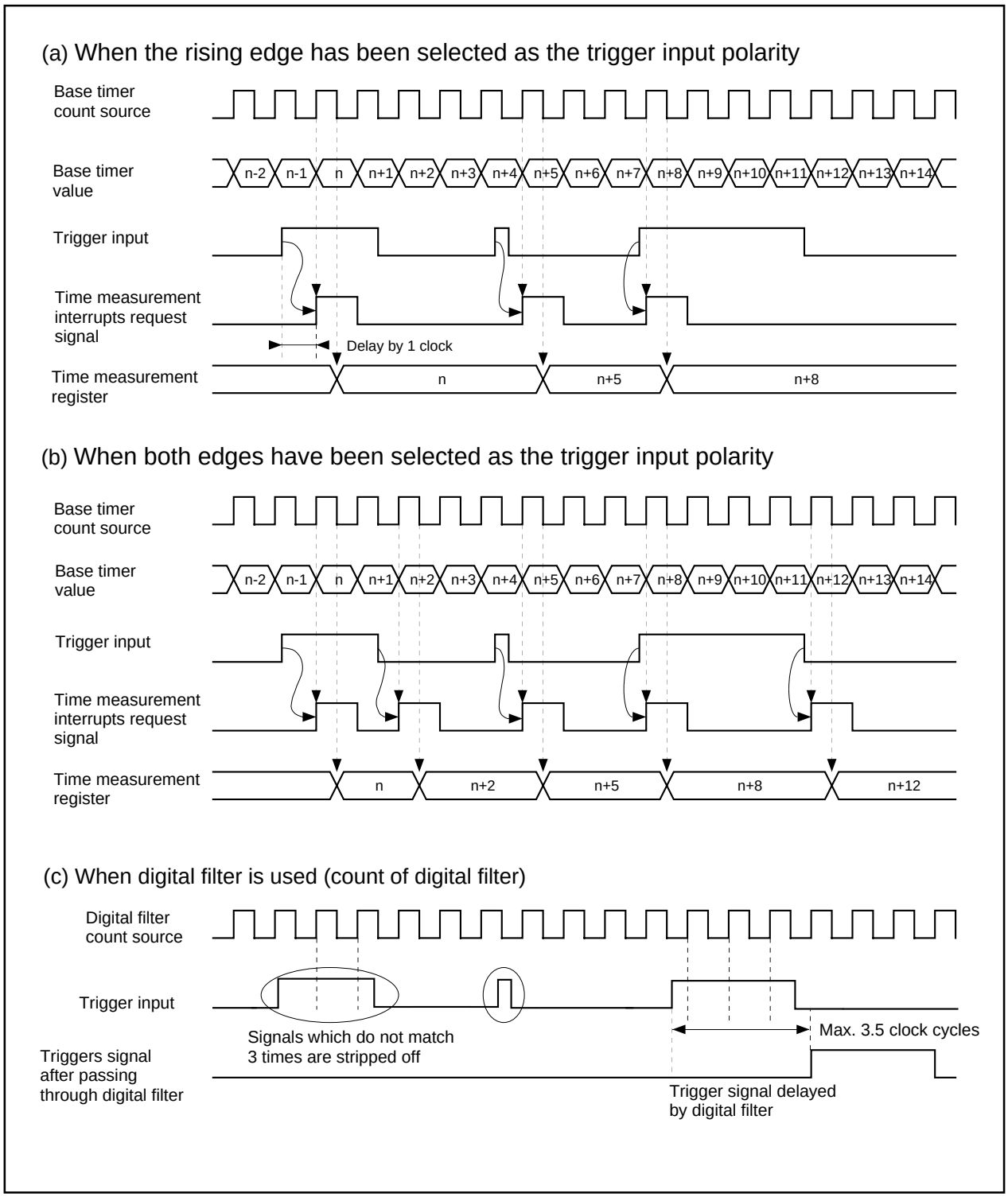
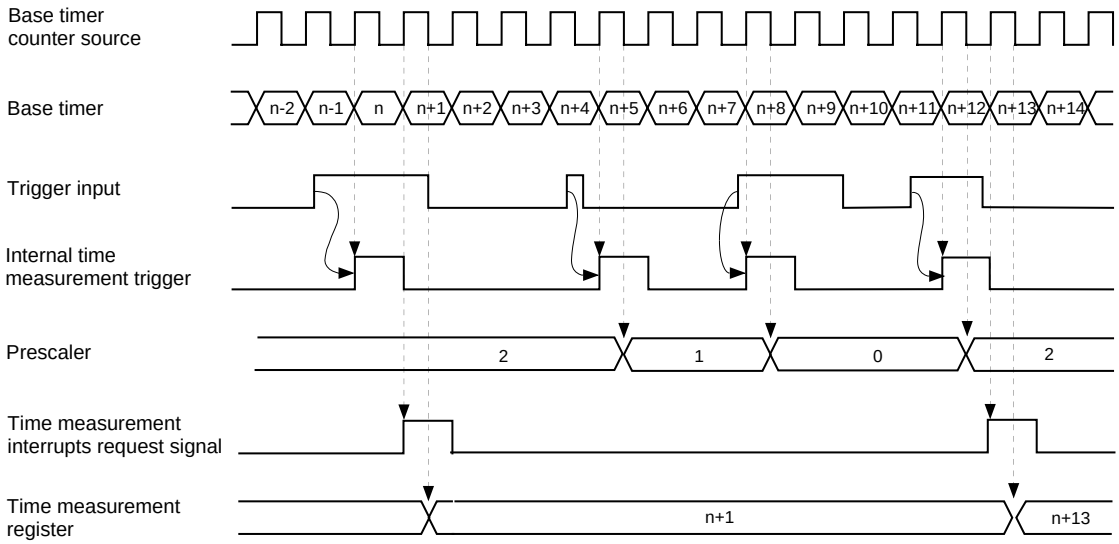



**Figure 1. 23. 14 Operation timing of time measurement function**

(a) When prescaler function is used (the value of time measurement prescaler register is "2".)




(b) When gate function is used (gate function released by matching WG register)

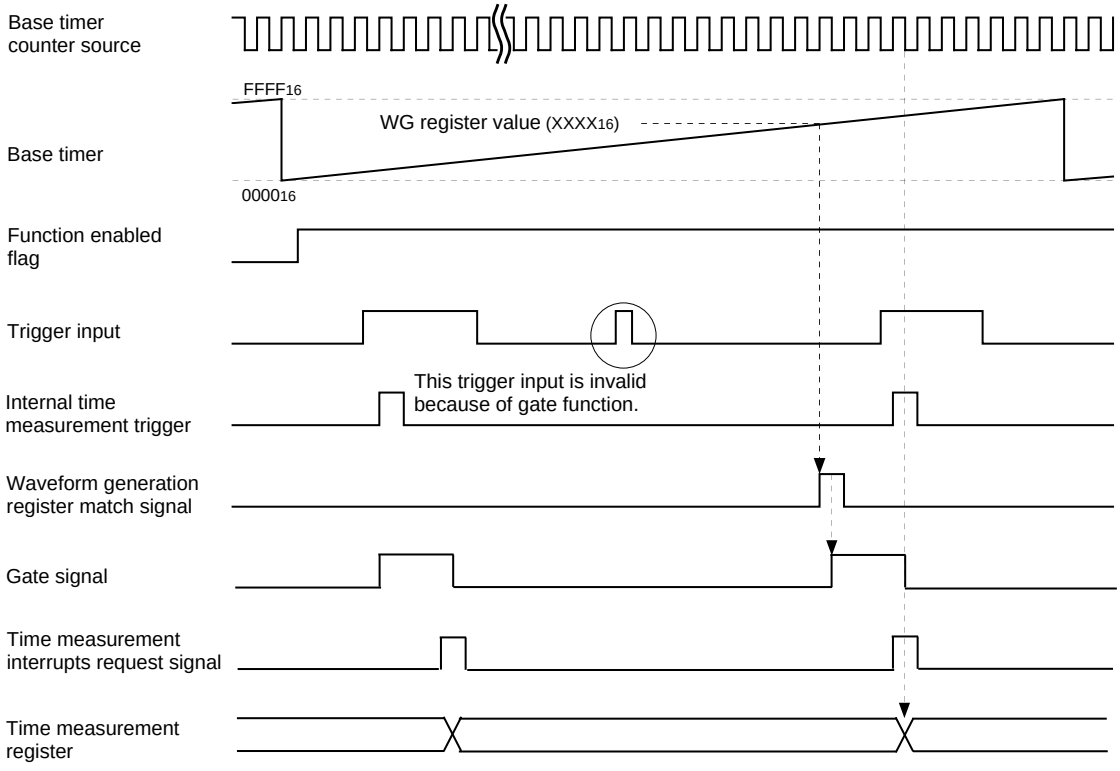



**Figure 1. 23. 15. Operation timing when gate function and prescaler function is used**

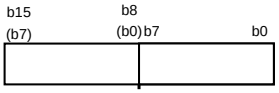
## Waveform generation (WG) function (group 0 to 3)

Waveforms are generated when the base timer value matches the value of WG register j.

There are five mode in WG function: single phase waveform output mode (group 0 to 3), phase delayed waveform output mode (group 0 to 3), SR (Set/Reset) waveform output mode (group 0 to 3), bit modulation PWM output mode (group 2 and 3) and parallel real-time port output mode (group 2 and 3).

The WG function related registers are shown in Figures 1.23.16 to 1.23.19.

### Group i waveform generation register j (i=0 to 3/j=0 to 7)



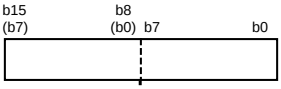

| Symbol | Address | When reset |
|---|---|---|
| GiPOj(i=0/j=0 to 2) | 00C1$_{16}$,00C0$_{16}$, 00C3$_{16}$,00C2$_{16}$, 00C5$_{16}$,00C4$_{16}$ | Indeterminate |
| GiPOj(i=0/j=3 to 5) | 00C7$_{16}$,00C6$_{16}$, 00C9$_{16}$,00C8$_{16}$, 00CB$_{16}$,00CA$_{16}$ | Indeterminate |
| GiPOj(i=0/j=6,7) | 00CD$_{16}$,00CC$_{16}$, 00CF$_{16}$,00CE$_{16}$ | Indeterminate |
| GiPOj(i=1/j=0 to 2) | 0101$_{16}$,0100$_{16}$, 0103$_{16}$,0102$_{16}$, 0105$_{16}$,0104$_{16}$ | Indeterminate |
| GiPOj(i=1/j=3 to 5) | 0107$_{16}$,0106$_{16}$, 0109$_{16}$,0108$_{16}$, 010B$_{16}$,010A$_{16}$ | Indeterminate |
| GiPOj(i=1/j=6,7) | 010D$_{16}$,010C$_{16}$, 010F$_{16}$,010E$_{16}$ | Indeterminate |
| GiPOj(i=2/j=0 to 2) | 0141$_{16}$,0140$_{16}$, 0143$_{16}$,0142$_{16}$, 0145$_{16}$,0144$_{16}$ | Indeterminate |
| GiPOj(i=2/j=3 to 5) | 0147$_{16}$,0146$_{16}$, 0149$_{16}$,0148$_{16}$, 014B$_{16}$,014A$_{16}$ | Indeterminate |
| GiPOj(i=2/j=6,7) | 014D$_{16}$,014C$_{16}$, 014F$_{16}$,014E$_{16}$ | Indeterminate |
| GiPOj(i=3/j=0 to 2) | 0181$_{16}$,0180$_{16}$, 0183$_{16}$,0182$_{16}$, 0185$_{16}$,0184$_{16}$ | Indeterminate |
| GiPOj(i=3/j=3 to 5) | 0187$_{16}$,0186$_{16}$, 0189$_{16}$,0188$_{16}$, 018B$_{16}$,018A$_{16}$ | Indeterminate |
| GiPOj(i=3/j=6,7) | 018D$_{16}$,018C$_{16}$, 018F$_{16}$,018E$_{16}$ | Indeterminate |

| Function | Setting range | R | W |
|---|---|---|---|
| A compared value for waveform generation is stored. (Note) | 0000$_{16}$ to FFFF$_{16}$ | ○ | ○ |

WG: Waveform Generation

Note: When resetting the base timer on ch0, the timer is reset 2 clock cycles after it matches the waveform generation register of ch0.

### Group 3 waveform generation mask register j (j=4 to 7)




| Symbol | Address | When reset |
|---|---|---|
| G3MKj (j=4,5) | 0199$_{16}$,0198$_{16}$, 019B$_{16}$,019A$_{16}$ | Indeterminate |
| G3MKj (j=6,7) | 019D$_{16}$,019C$_{16}$, 019F$_{16}$,019E$_{16}$ | Indeterminate |

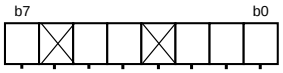
| Function | Setting range | R | W |
|---|---|---|---|
| Masks base timer value (Note 1) (Note 2) | 0000$_{16}$ to FFFF$_{16}$ | ○ | ○ |

Note 1: This function is provided only for the waveform generation functions on ch 4 to 7 of Intelligent I/O group 3.

Note 2: Comparison results are masked in bit positions where a "1" has been set for the register bits.

**Figure 1. 23. 16. WG-related register (1)**

## Group i waveform generation control register j (i=0 to 1/ j=0 to 7) (Note 1)




| Symbol | Address | When reset |
|---|---|---|
| GiPOCRj (i=0/j=0,1) | $00D0_{16}$, $00D1_{16}$ | $0X00X000_2$ |
| GiPOCRj (i=0/j=4,5) | $00D4_{16}$, $00D5_{16}$ | $0X00X000_2$ |
| GiPOCRj (i=1/j=0 to 3) | $0110_{16}$, $0111_{16}$, $0112_{16}$, $0113_{16}$ | $0X00X000_2$ |
| GiPOCRj (i=1/j=4 to 7) | $0114_{16}$, $0115_{16}$, $0116_{16}$, $0117_{16}$ | $0X00X000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| MOD0 | Operation mode select bit | b2 b1 b0<br>0 0 0: Single PWM mode<br>0 0 1: S-R PWM mode (Note 2)<br>0 1 0: Phase delayed PWM mode<br>0 1 1: Must not be set<br>1 0 0: Must not be set<br>1 0 1: Must not be set<br>1 1 0: Must not be set (Note 3)<br>1 1 1: Assigns communication output to a port (Note 4) | ○ | ○ |
| MOD1 | | | ○ | ○ |
| MOD2 | | | ○ | ○ |
| ——— | Must always set to "0"<br>When read, the value of this bit is indeterminate. | | — | — |
| IVL | Output initial value select bit | 0: Outputs "0" as the initial value<br>1: Outputs "1" as the initial value | ○ | ○ |
| RLD | Reload timing select bit | 0: Reloads a new count when CPU writes the count<br>1: Reloads a new count when the base timer i is reset | ○ | ○ |
| ——— | Must always set "0"<br>When read, the value of this bit is indeterminate. | | — | — |
| INV | Inverted output function select bit (Note 5) | 0: Output is not inverted<br>1: Output is inverted | ○ | ○ |

Note 1: Group 0 and 1 have 16-bit WG function and 32-bit WG function.
The 16-bit WG function is available for 4 channels (ch=0,1,4,5) with group 0 and 8 channels (ch=0 to 7) with group 1. When using the 16-bit WG function, use the WG register values for ch2, 3, 6 and 7 of group 0 as they are, or, if writing values, write "$00_{16}$".
The 32-bit WG function can be used with 8 channels (ch0 to 7) by linking groups 0 and 1. When using the 32-bit WG function, write the same value for WG registers of similar channels in groups 0 and 1.

Note 2: This setting is valid only on even-numbered channels. When this mode is selected, settings for corresponding odd-numbered (even number + 1) channels are ignored. Waveforms are output for even-numbered channels, not output for odd-numbered channels.

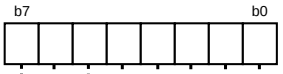
Note 3: When receiving in UART mode of group 0 and 1, group i WG control register 2 is set to be "$00000110_2$".

Note 4: This setting is valid only for WG function ch0 and 1. Do not set this value for other channels.

Note 5: Inverted output function is allocated at the final stage of WG circuit. Therefore, when selecting

**Figure 1. 23. 17. WG-related register (2)**

## Group i waveform generation control register j (i=2 to 3/ j=0 to 7)

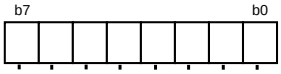



| Symbol | Address | When reset |
|---|---|---|
| GiPOCRj (i=2/j=0 to 3) | $0150_{16}$, $0151_{16}$, $0152_{16}$, $0153_{16}$ | $00_{16}$ |
| GiPOCRj (i=2/j=4 to 7) | $0154_{16}$, $0155_{16}$, $0156_{16}$, $0157_{16}$ | $00_{16}$ |
| GiPOCRj (i=3/j=0 to 3) | $0190_{16}$, $0191_{16}$, $0192_{16}$, $0193_{16}$ | $00_{16}$ |
| GiPOCRj (i=3/j=4 to 7) | $0194_{16}$, $0195_{16}$, $0196_{16}$, $0197_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| MOD0 | Operation mode select bit | b2 b1 b0<br>0 0 0: Single PWM mode<br>0 0 1: S-R PWM mode (Note 1)<br>0 1 0: Phase delayed PWM mode<br>0 1 1: Must not be set<br>1 0 0: Bit modulation PWM mode<br>1 0 1: Must not be set<br>1 1 0: Must not be set<br>1 1 1: Assigns communication output to a port (Note 2) | ○ | ○ |
| MOD1 | | | ○ | ○ |
| MOD2 | | | ○ | ○ |
| PRT | Parallel RTP output trigger select bit | 0: Match of WG register j isn't trigger<br>1: Match of WG register j is trigger | ○ | ○ |
| IVL | Output initial value select bit | 0: Outputs "0" as the initial value<br>1: Outputs "1" as the initial value | ○ | ○ |
| RLD | Reload timing select bit | 0: Reloads a new count when CPU writes the count<br>1: Reloads a new count when the base timer i is reset | ○ | ○ |
| RTP | RTP port function select bit | 0: Not use<br>1: Use | ○ | ○ |
| INV | Inverted output function select bit (Note 3) | 0: Output is not inverted<br>1: Output is inverted | ○ | ○ |

Note 1: This setting is valid only on even-numbered channels. When this mode is selected, settings for corresponding odd-numbered (even number + 1) channels are ignored. Waveforms are output for even-numbered channels, not output for odd-numbered channels.
Note 2: This setting is valid only for group 2 WG function ch0 and 1. Do not set this value for other channels.
Note 3: Inverted output function is allocated at the final stage of WG circuit. Therefore, when selecting "0" output by IVL bit and inverted output by INV bit, "1" is output.

## Group i function enable register (i=0 to 3)

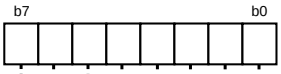



| Symbol | Address | When reset |
|---|---|---|
| GiFE (i=0 to 3) | $00E6_{16}$, $0126_{16}$, $0166_{16}$, $01A6_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| IFE0 | Ch0 function enable bit | Whether the corresponding port functions is selected<br>0 : Disables function on ch i<br>1 : Enables function on ch i | ○ | ○ |
| IFE1 | Ch1 function enable bit | | ○ | ○ |
| IFE2 | Ch2 function enable bit | | ○ | ○ |
| IFE3 | Ch3 function enable bit | | ○ | ○ |
| IFE4 | Ch4 function enable bit | | ○ | ○ |
| IFE5 | Ch5 function enable bit | | ○ | ○ |
| IFE6 | Ch6 function enable bit | | ○ | ○ |
| IFE7 | Ch7 function enable bit | | ○ | ○ |

**Figure 1. 23. 18. WG-related register (3)**

## Group i function select register (i=0, 1)

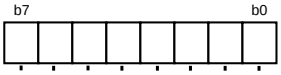



| Symbol | Address | When reset |
|---|---|---|
| GiFS (i=0,1) | $00E7_{16}$, $0127_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| FSC0 | Ch0 TM/WG function select bit | Whether the corresponding port functions as TM or WG is selected<br>0 : WG function is selected<br>1 : TM function is selected | ○ | ○ |
| FSC1 | Ch1 TM/WG function select bit | | ○ | ○ |
| FSC2 | Ch2 TM/WG function select bit | | ○ | ○ |
| FSC3 | Ch3 TM/WG function select bit | | ○ | ○ |
| FSC4 | Ch4 TM/WG function select bit | | ○ | ○ |
| FSC5 | Ch5 TM/WG function select bit | | ○ | ○ |
| FSC6 | Ch6 TM/WG function select bit | | ○ | ○ |
| FSC7 | Ch7 TM/WG function select bit | | ○ | ○ |

Note : In group 0, channles 2, 3, 6 and 7 cannot be selected in 16-bit mode. All channel can be selected in 32-bit mode.
In group 1, channles 0 and 3 to 5 cannot be selected in 16-bit mode. All channel can be selected in 32-bit mode.

## Group i RTP output buffer register (i=2,3)




| Symbol | Address | When reset |
|---|---|---|
| GiRTP (i=2,3) | $0167_{16}$, $01A7_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| RTP0 | Ch0 RTP output buffer | The corresponding port's output value is set<br>0 : Output "0"<br>1 : Output "1" | ○ | ○ |
| RTP1 | Ch1 RTP output buffer | | ○ | ○ |
| RTP2 | Ch2 RTP output buffer | | ○ | ○ |
| RTP3 | Ch3 RTP output buffer | | ○ | ○ |
| RTP4 | Ch4 RTP output buffer | | ○ | ○ |
| RTP5 | Ch5 RTP output buffer | | ○ | ○ |
| RTP6 | Ch6 RTP output buffer | | ○ | ○ |
| RTP7 | Ch7 RTP output buffer | | ○ | ○ |

**Figure 1. 23. 19. WG-related register (4)**

### (1) Single phase waveform output mode (group 0 to 3)

This mode is set when the base timer value matches the value of WG register j, and reset when the base timer overflows or the count is reset. Specifications for the single phase waveform output mode are given in Table 1.23.5 and an operating chart for the single phase waveform output mode in Figure 1.23.20.

**Table 1. 23.5. Specifications of single phase waveform output mode**

| Item | Specifications |
|---|---|
| Output waveform | •When free run operation<br>Period : Base timer count source x 1/65536<br>"H" level width : 1/base timer count source x (65536 - m)<br>•Resetting when the base timer matches WG register 0 (ch0)<br>Period : Base timer count source x 1/(k+2)<br>"H" level width : 1/base timer count source x (k+2-m)<br>m : values set to WG register j k: values set to WG register 0 |
| Waveform output start condition | Write "1" to the function enable bit(Note) |
| Waveform output stop condition | Write "0" to the function enable bit |
| Interrupt generation timing | When the base timer value matches the WG register j |
| OUTC pin | Pulse output (Corresponding pins are set with the function select register.) |
| Read from the WG register 0 | The set value is output |
| Write to the WG register 0 | Can always write |
| Select function | •Initial value setting function<br>Sets output level used at waveform output start<br>•Inverted output function<br>Inverts waveform output level and outputs the waveform from the OUTC pin |

Note: On channels where both the time measurement function and waveform output function can be used, select the waveform output function for the function select register (addresses 00E7$_{16}$ and 0127$_{16}$).

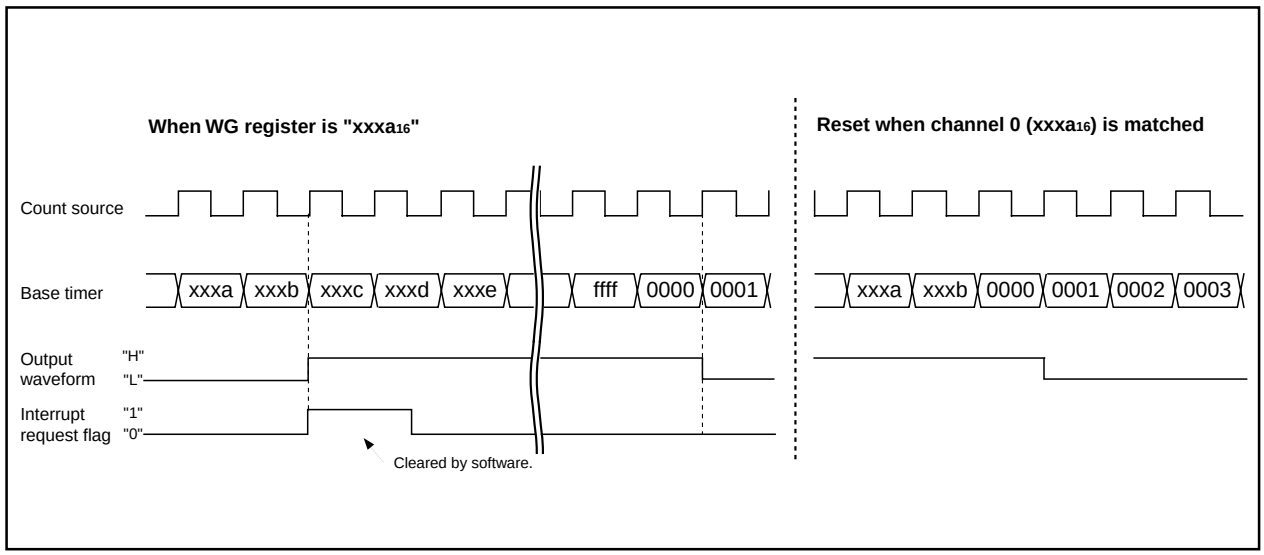



**Figure 1. 23. 20. Operation timing in single phase waveform output mode**

## (2) Phase delayed waveform output mode (group 0 to 3)

This mode is repeatedly set and reset when the base timer value matches the value of WG register j. Specifications for the phase delayed waveform output mode are given in Table 1.23.6 and an operation timing in phase delayed waveform output mode in Figure 1.23.21.

**Table 1. 23.6. Specifications of phase delayed waveform output mode**

| Item | Specifications |
|---|---|
| Output waveform | •When free run operation<br>Period : Base timer count source x 1/65536 x 1/2<br>"H" and "L" level width : 1/base timer count source x 65536<br>•Resetting when group i base timer matches WG register 0 (ch0)<br>Period : Base timer count source x 1/(k+2) x 1/2<br>"H" and "L" level width : 1/base timer count source x (k+2)<br>k : values set to WG register 0 |
| Waveform output start condition | Write "1" to the function enable bit (Note) |
| Waveform output stop condition | Write "0" to the function enable bit |
| Interrupt generation timing | When the base timer value matches the WG register j |
| OUTCij pin | Pulse output (Corresponding pins are set with the function select register.) |
| Read from the WG register | The set value is output |
| Write to the WG register | Can always write |
| Select function | •Initial value setting function<br>Sets output level used at waveform output start<br>•Inverted output function<br>Inverts waveform output level and outputs the waveform from the OUTC pin |

Note : On channels where both the time measurement function and waveform output function can be used, select the waveform output function for the function select register (addresses $00E7_{16}$ and $0127_{16}$).

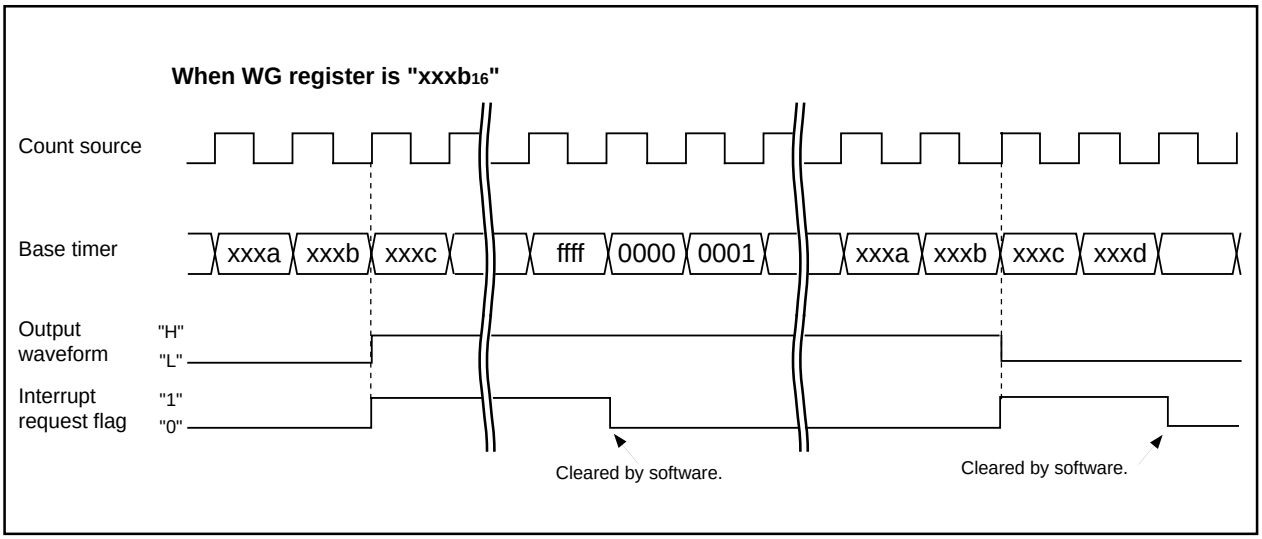



**Figure 1. 23. 21. Operation timing in phase delayed waveform output mode**

### (3) SR (Set/Reset) waveform output mode (group 0 to 3)

This mode is set when the base timer value matches the value of WG register j (j is an even-numbered channel), and reset when the base timer matches the WG register (j + 1) or the base timer value is “0”. Specifications for the SR waveform output mode are given in Table 1.23.7 and an operating chart for the SR waveform output mode in Figure 1.23.22.

**Table 1. 23.7. Specifications of SR waveform output mode**

| Item | Specifications |
|---|---|
| Output waveform | •When free run operation<br>Period : Base timer count source x 1/65536<br>"H" level width : 1/base timer count source x (m-p)<br>•Resetting when base timer matches WG register 0 (ch0)<br>Period : Base timer count source x 1/(k+2) (Note 1)<br>"H" level width : 1/base timer count source x (m-p)<br>m : values set to WG register j p : values set to WG register i(j+1)<br>k : values set to WG register 0 (j is an even-numbered channel) (Note 2) |
| Waveform output start condition | Write "1" to the function enable bit (Note 3) |
| Waveform output stop condition | Write "0" to the function enable bit |
| Interrupt generation timing | When the base timer value matches the WG register j |
| OUTC pin (Note 4) | Pulse output (Corresponding pins are set with the function select register.) |
| Read from the WG register | The set value is output |
| Write to the WG register | Can always write |
| Select function (Note 5) | •Initial value setting function<br>Sets output level used at waveform output start<br>•Inverted output function<br>Inverts waveform output level and outputs the waveform from the OUTC pin |

Note 1: The SR waveform output function that sets and resets the mode on ch0 and 1 cannot be used when the base timer is reset by WG register 0 (ch0).

Note 2: Set WG register values for odd-numbered channels that are lower than even-numbered channels.

Note 3: On channels where both the time measurement function and waveform output function can be used, select the waveform output function for the function select register (addresses $00E7_{16}$ and $0127_{16}$).

Note 4: SR waveforms are output for even-numbered channels only.

Note 5: Settings for the WG control register on the odd-numbered channels are ignored.

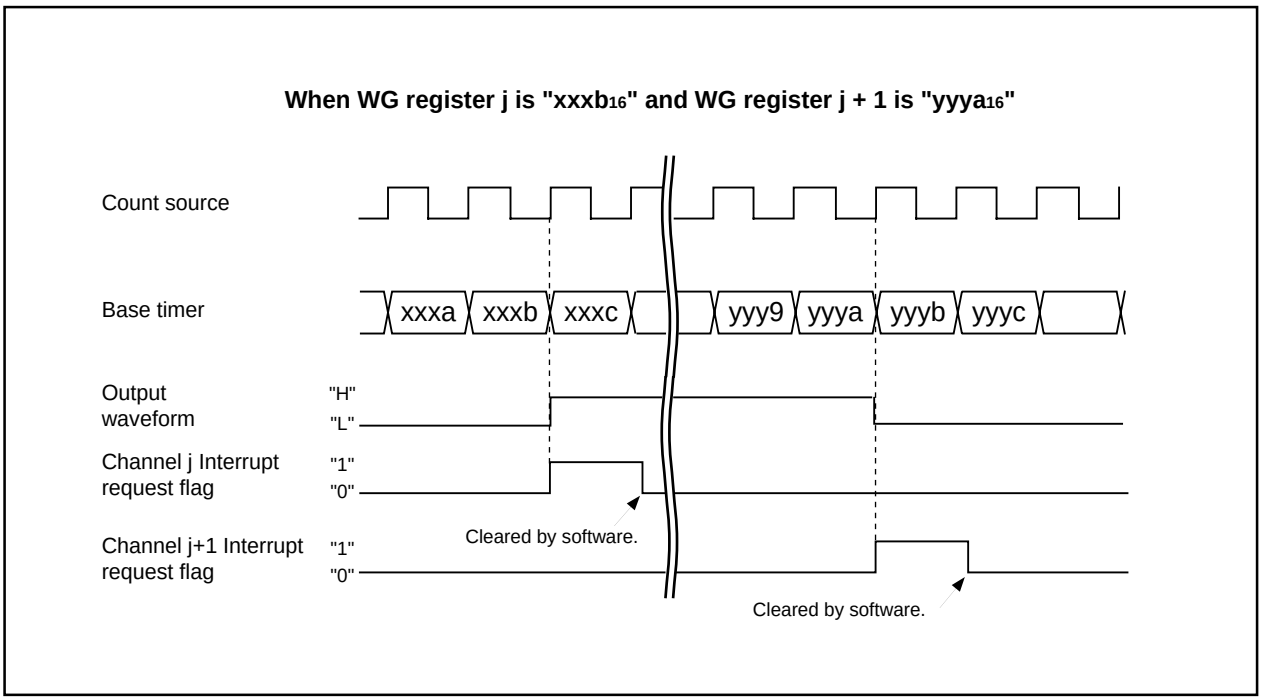



**Figure 1. 23. 22. Operation timing in SR waveform output mode**

## (4) Bit modulation PWM output mode (group 2 and 3)

This mode performs PWM to improve output resolution. Specifications for the bit modulation PWM mode are given in Table 1.23.8 and an operating chart for the bit modulation PWM mode in Figure 1.23.23.

**Table 1. 23.8. Specifications of bit modulation PWM mode**

| Item | Specifications |
|---|---|
| Output waveform | Period : Base timer count source x 1/64<br>"H" level width (avelage) : 1/base timer count source x [k+(m/1024)]<br>k : values set to WG register j (six high-order bits)<br>m : values set to WG register j (ten lower-order bits) |
| Waveform output start condition | Write "1" to the function enable bit |
| Waveform output stop condition | Write "0" to the function enable bit |
| Interrupt generation timing | When the base timer value matches the WG register j |
| OUTC pin | Pulse output (Corresponding pins are set with the function select register.) |
| Read from the WG register j | The set value is output |
| Write to the WG register j | Can always write |
| Select function | •Initial value setting function<br>Sets output level used at waveform output start<br>•Inverted output function<br>Inverts waveform output level and outputs the waveform from the OUTC pin |

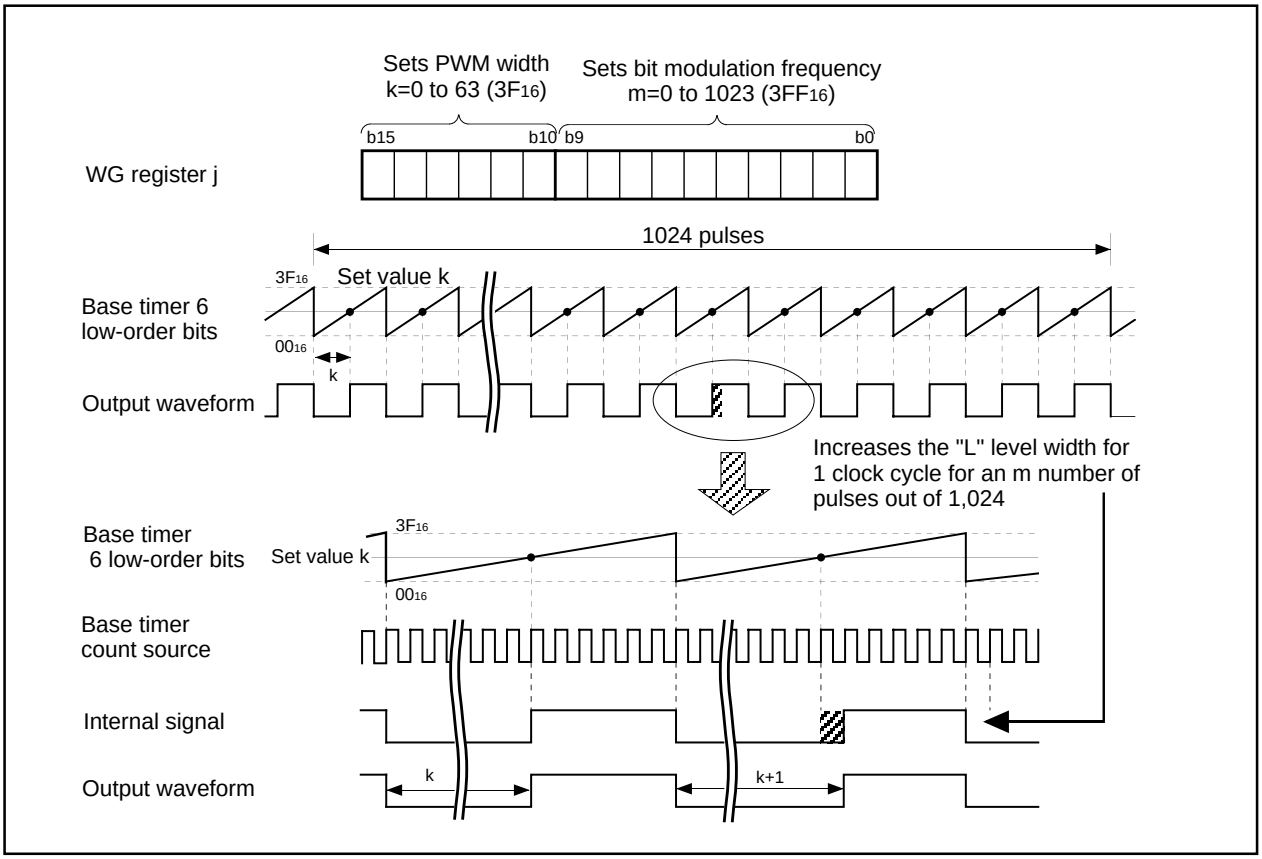



**Figure 1. 23. 23. Operation timing in bit modulation PWM mode**

## (5) Real-time port output mode (group 2 and 3)

This mode outputs the value set in the real-time port register from the OUTC pin when the base timer value matches the value of WG register j. Specifications for the real-time port output mode are given in Table 1.23.9 and a block diagram and timing chart of the real-time port output function in Figure 1.23.24.

**Table 1. 23.9. Specifications of real-time port output mode**

| Item | Specifications |
|---|---|
| Waveform output start condition | Write "1" to the function enable bit |
| Waveform output stop condition | Write "0" to the function enable bit |
| Interrupt generation timing | When the base timer value matches the WG register j |
| OUTC pin | RTP output (Corresponding pins are set with the function select register.) |
| Read from the WG register j | The set value is output |
| Write to the WG register j | Can always write |
| Read from the RTP output buffer register | The set value is output |
| Write to the RTP output buffer register | Can always write |
| Select function | •Initial value setting function<br>Sets output level used at waveform output start<br>•Inverted output function<br>Inverts waveform output level and outputs the waveform from the OUTC pin |

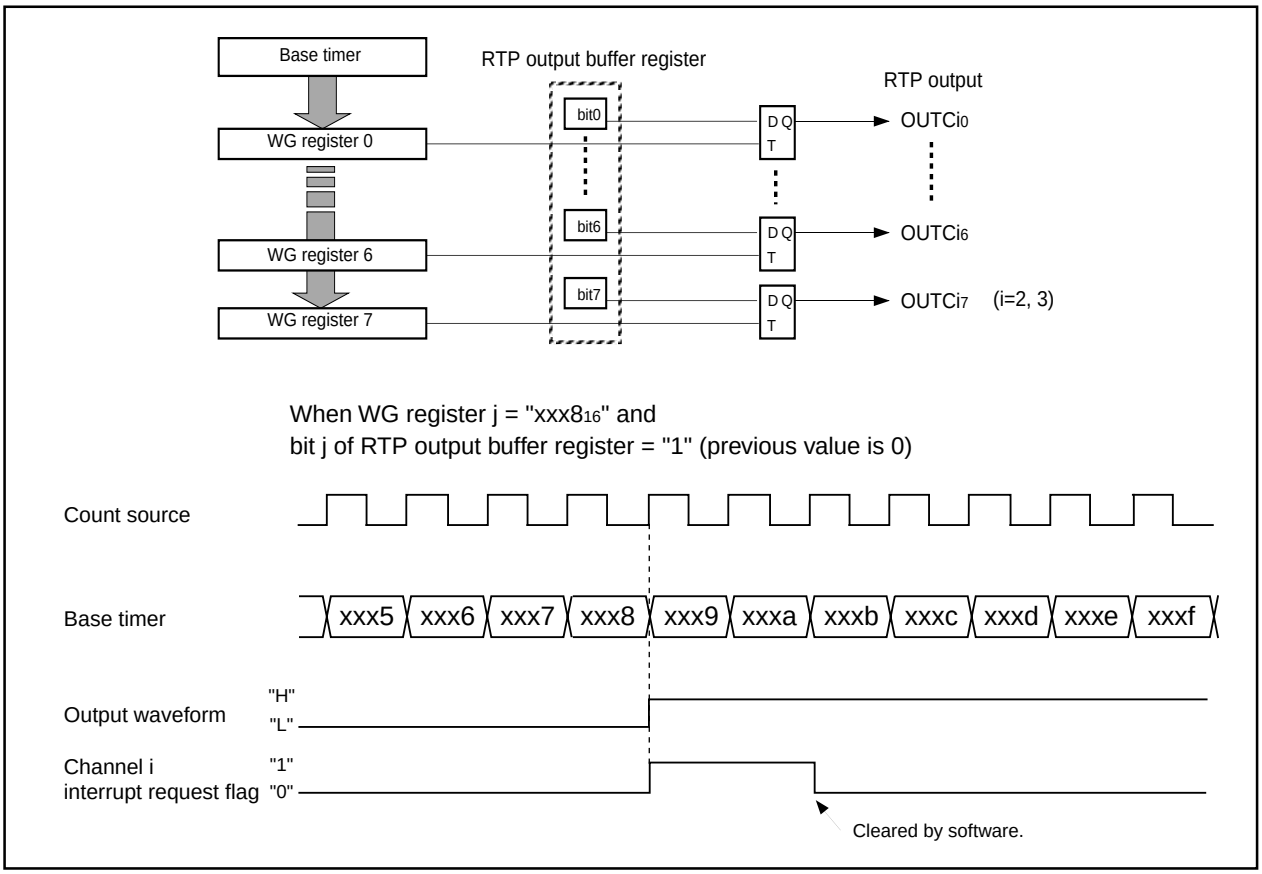


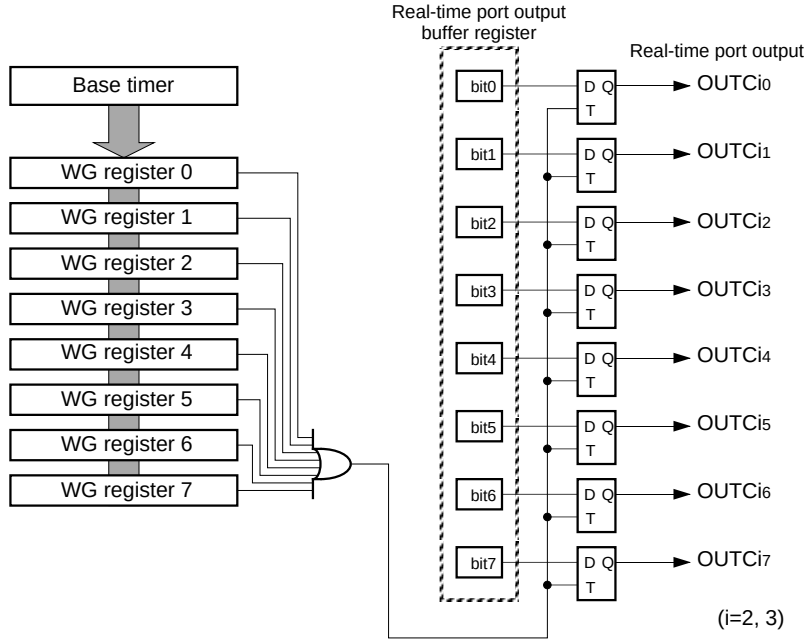

**Figure 1. 23. 24. Block diagram and operation timing of real-time port output function**

### (6) Parallel real-time port output mode (group 2 and 3)

This mode outputs the value set in the real-time port register from the OUTC pin when the base timer value matches the value of WG register j. Specifications for the parallel real-time port output mode are given in Table 1.23.10 and a block diagram and timing chart of the real-time port output function in Figure 1.23.25.

**Table 1. 23.10. Specifications of parallel real-time port output mode**

| Item | Specifications |
|---|---|
| Waveform output start condition | Write "1" to the function enable bit |
| Waveform output stop condition | Write "0" to the function enable bit |
| Interrupt generation timing | When the base timer value matches the WG register |
| OUTC pin | RTP output (Corresponding pins are set with the function select register.) |
| Read from the WG register | The set value is output |
| Write to the WG register | Can always write |
| Read from the RTP output buffer register | The set value is output |
| Write to the RTP output buffer register | Can always write |
| Select function | •Initial value setting function<br>Sets output level used at waveform output start<br>•Inverted output function<br>Inverts waveform output level and outputs the waveform from the OUTC pin |




**When WG register j = "xxx1$_{16}$" RTP output buffer register = "0xxx xx01$_2$"**
**WG register j + 1 = "xxx5$_{16}$" RTP output buffer register = "1xxx xx10$_2$"**
**(this value is saved to RTP output buffer register as channel j interrupt request trigger)**
**WG register j + 2 = "xxxC$_{16}$" RTP output buffer register = "1xxx xx11$_2$"**
**(this value is saved to RTP output buffer register as channel j+1 interrupt request trigger)**

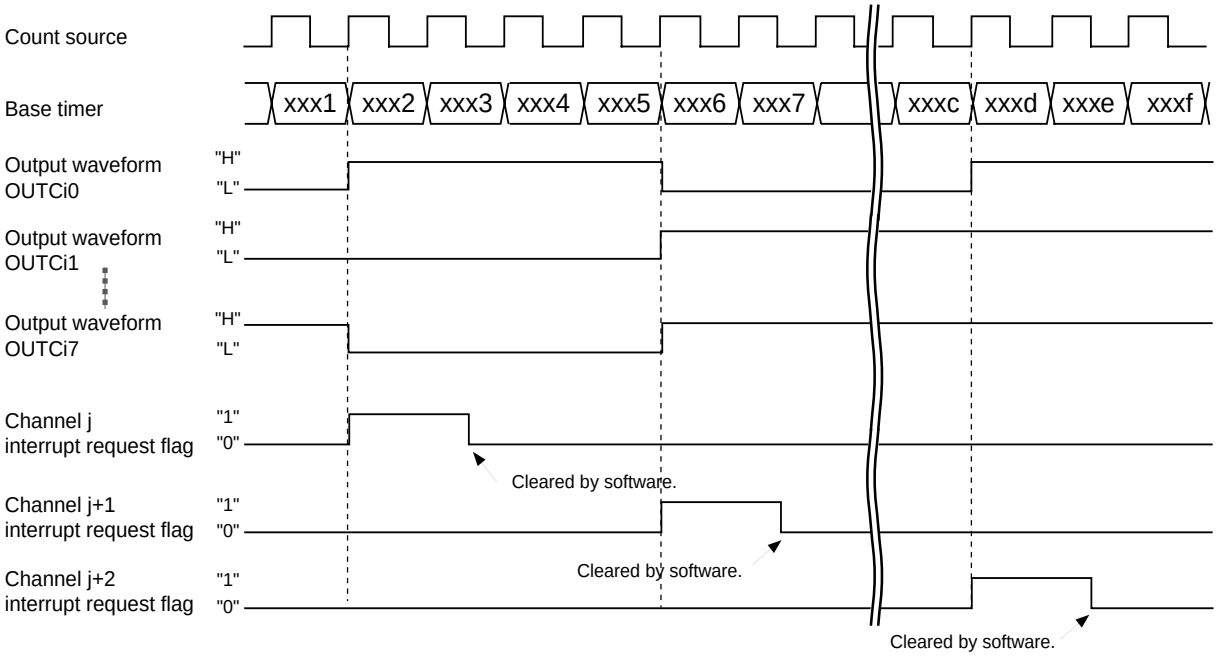



**Figure 1. 23. 25. Block diagram and operation timing of parallel real-time port output function**

## Serial I/O (group 0 to 2)

Intelligent I/O groups 0 to 2 each have two internal 8-bit shift registers. When used in conjunction with the time measurement (TM) function or WG function, these shift registers enable clock synchronous/asynchronous serial communications.

### (1) Clock synchronous serial I/O mode (group 0, 1)

Intelligent I/O groups 0 and 1 each have communication block that have two internal 8-bit shift registers. When used in conjunction with the communication block and WG function, these shift registers enable 8-bit clock synchronous and HDLC data process function. When used in conjunction with the communication block, TM function and WG function, these shift registers enable 8-bit clock asynchronous communication.

Table 1.23.11 lists using registers in group 0 and 1, figure 1.23.26 to 1.23.29 shows the related registers.

**Table 1.23.11. Using registers in group 0 and 1**

| | Clock synchronous serial I/O | UART | HDLC |
|---|---|---|---|
| Base timer control register 0 | √ | √ | √ |
| Base timer control register 1 | √ | √ | √ |
| Time measument control register 2 | – | √ | – |
| Waveform generate control register 0 | √ | √ | √ |
| Waveform generate control register 1 | – | – | √ |
| Waveform generate control register 2 | √ | √ | – |
| Waveform generate control register 3 | √ | √ | – |
| Waveform generate register 0 | √ | √ | √ |
| Waveform generate register 1 | √ | – | √ |
| Time measument /Waveform generate register 2 | √ | √ | – |
| Waveform generate register 3 | √ | √ | – |
| Function select register | √ | √ | √ |
| Function enable register | √ | √ | √ |
| SI/O communication mode register | √ | √ | √ |
| SI/O extended mode register | – | – | √ |
| SI/O communication control register | √ | √ | √ |
| SI/O extended transmit control register | – | – | √ |
| SI/O extended receive control register | – | – | √ |
| SI/O special communication interrupt detect register | – | – | √ |
| SI/O receive buffer register | √ | √ | √ |
| Transmit buffer | √ | √ | √ |
| (Receive data register ) | – | – | √ |
| Data compare register j (j=0 to 3) | – | – | √ |
| Data mask register j (j=0, 1) | – | – | √ |
| Transmit CRC code register | – | – | √ |
| Receive CRC code register | – | – | √ |
| Transmit output register | – | – | √ |
| Receive input register | – | – | √ |

√ : Use – : Not use

## Group i receive input register (i=0,1)

b7 b0

| Symbol | Address | When reset |
|---|---|---|
| GiRI (i=0, 1) | $00EC_{16}$, $012C_{16}$ | Indeterminate |

| Function | Setting range | R | W |
|---|---|---|---|
| Data that is input to the receive data process unit | $00_{16}$ to $FF_{16}$ | — | ○ |

## Group i transmit output register (i=0,1)

b7 b0

| Symbol | Address | When reset |
|---|---|---|
| GiTO (i=0, 1) | $00EE_{16}$, $012E_{16}$ | Indeterminate |

| Function | Setting range | R | W |
|---|---|---|---|
| Data that is output from the transmit data process unit | —— | ○ | — |

## Group i SI/O communication control register (i=0,1)

b7 b0

| Symbol | Address | When reset |
|---|---|---|
| GiCR (i=0,1) | $00EF_{16}$, $012F_{16}$ | 0000 $X000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TI | Transmit buffer empty flag | 0 : Data present in transmit buffer register<br>1 : No data present in transmit buffer register | ○ | — |
| TXEPT | Transmit register empty flag | 0 : Data present in transmit register (during transmission)<br>1 : No data present in transmit register (transmission completed) | ○ | — |
| RI | Receive complete flag | 0 : No data present in receive buffer register<br>1 : Data present in receive buffer register | ○ | — |
| —— | Nothing is assigned. When write, set to "0". When read, the contents is indeterminate. | | — | — |
| TE | Transmit enable bit | 0 : Transmission disabled<br>1 : Transmission enabled | ○ | ○ |
| RE | Receive enable bit | 0 : Reception disabled<br>1 : Reception enabled | ○ | ○ |
| IPOL | RxD input polarity reverse select bit | 0 : No reverse (Usually set to "0")<br>1 : Reverse (Note) | ○ | ○ |
| OPOL | TxD output polarity reverse select bit | 0 : No reverse (Usually set to "0")<br>1 : Reverse (Note) | ○ | ○ |

Note :This bit is set to "1" in UART mode.

**Figure 1. 23. 26. Group 0 and 1 related register (1)**

## Group i SI/O receive buffer register (i=0,1)

b15 b14 b13 b12 b11 b10 b9 b8 (b7)(b6)(b5)(b4)(b3)(b2)(b1)(b0) b7 ... b0

| Symbol | Address | When reset |
|---|---|---|
| GiRB(i=0,1) | $00E9_{16}$,$00E8_{16}$, $0129_{16}$,$0128_{16}$ | XX00 XXXX XXXX XXXX |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| ——— | Receive buffer | Receive data | ○ | ○ |
| ——— | Nothing is assigned. When read, their value are indeterminate. | | – | – |
| OER | Overrun error flag (Note) | 0 : No overrun error<br>1 : Overrun error found | ○ | – |
| FER | Framing error flag (Note) | 0 : No Framing error<br>1 : Framing error found | ○ | – |
| ——— | Nothing is assigned. When read, its value is indeterminate. | | – | – |
| ——— | | | – | – |

Note: Only effective for receive data.

## Group i SI/O communication mode register (i=0,1)

b7 ... b0

| Symbol | Address | When reset |
|---|---|---|
| GiMR (i=0,1) | $00ED_{16}$, $012D_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| GMD0 | Communication mode select bit | b1 b0<br>0 0 : UART mode<br>0 1 : Serial I/O mode<br>1 0 : Special communication mode<br>1 1 : HDLC data process mode | ○ | ○ |
| GMD1 | | | ○ | ○ |
| CKDIR | Internal/external clock select bit | 0 : Internal clock (Note 2)<br>1 : External clock (Note 3) | ○ | ○ |
| STPS | Stop bit length select bit (Note 1) | 0 : 1 stop bit<br>1 : 2 stop bits | ○ | ○ |
| ——— | Reserved bit | Must set to "0". | ○ | ○ |
| ——— | | | ○ | ○ |
| UFORM | Transfer direction select bit | 0 : LSB first<br>1 : MSB first | ○ | ○ |
| IRS | Transmit interrupt cause select bit | 0 : Transmit buffer is empty<br>1 : Transmit is completed | ○ | ○ |

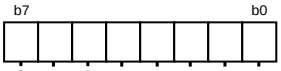
Note 1: Can be used only in the UART mode.

Note 2: Select a pin for clock output by setting the waveform generation control register, input function select register, and function select registers A, B and C.
Data transmission pins are the same as clock output pins.

Note 3: Select which pins will input the clock with the input function select register and set those pins to the input port using function select register A.
Data input pins are the same as with clock input pins.

**Figure 1. 23. 27. Group 0 and 1 related register (2)**

## Group i SI/O expansion mode register (i=0,1) (Note 1)

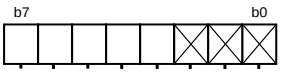



| Symbol | Address | When reset |
|---|---|---|
| GiEMR (i=0,1) | $00FC_{16}$, $013C_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SMODE | Synchronous mode select bit | 0 : Normal mode<br>1 : Resynchronous mode | ○ | ○ |
| CRCV | CRC initial value select bit | 0 : "$0000_{16}$" is set<br>1 : "$FFFF_{16}$" is set | ○ | ○ |
| ACRC | CRC initialization select bit | 0 : Not initialize<br>1 : Initialize (Note 2) | ○ | ○ |
| BSINT | Bit stuffing error interrupt select bit | 0 : Not use<br>1 : Use | ○ | ○ |
| RXSL | Reception source select bit | 0 : RxD pin<br>1 : Receive input register | ○ | ○ |
| TXSL | Transmission source select bit | 0 : TxD pin<br>1 : Transmit output register | ○ | ○ |
| CRC0 | CRC polynomial select bit | b7 b6<br>0 0 : $X^8+X^4+X+1$<br>0 1 : Must not be set<br>1 0 : $X^{16}+X^{15}+X^2+1$<br>1 1 : $X^{16}+X^{12}+X^5+1$ | ○ | ○ |
| CRC1 | | | ○ | ○ |

Note 1: Other than when in the special communication mode or HDLC data process mode, either use the reset state as is or write "$00_{16}$".
Note 2: Initialized when the data compare register matches.

## Group i SI/O expansion transmit control register (i=0,1) (Note)



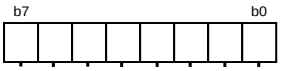

| Symbol | Address | When reset |
|---|---|---|
| GiETC (i=0,1) | $00FF_{16}$, $013F_{16}$ | $00000XXX_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| ——— | Nothing is assigned. When write, set "0".<br>When read, their contents are indeterminate. | | — | — |
| ——— | | | — | — |
| ——— | | | — | — |
| SOF | SOF transmit request bit | 0 : No SOF transmit request<br>1 : SOF transmit request | ○ | ○ |
| TCRCE | Transmit CRC enable bit | 0 : Not use<br>1 : Use | ○ | ○ |
| ABTE | Arbitration enable bit | 0 : Not use<br>1 : Use | ○ | ○ |
| TBSF0 | Transmit bit stuffing "1" insert select bit | 0 : "1" is not inserted<br>1 : "1" is inserted | ○ | ○ |
| TBSF1 | Transmit bit stuffing "0" insert select bit | 0 : "0" is not inserted<br>1 : "0" is inserted | ○ | ○ |

Note : Other than when in the special communication mode or HDLC data processing mode, either use the reset state as is or write "$00_{16}$".

**Figure 1. 23. 28. Group 0 and 1 related register (3)**

## Group i SI/O expansion receive control register (i=0,1) (Note 1)

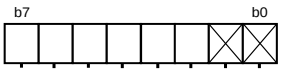



| Symbol | Address | When reset |
|---|---|---|
| GiERC (i=0,1) | 00FD16, 013D16 | 0016 |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CMP0E | Data compare function 0 select bit | 0 : Does not compare the received data with data compare register 0<br>1 : Compare the received data with data compare register 0 | ○ | ○ |
| CMP1E | Data compare function 1 select bit | 0 : Does not compare the received data with data compare register 1<br>1 : Compare the received data with data compare register 1 | ○ | ○ |
| CMP2E | Data compare function 2 select bit | 0 : Does not compare the received data with data compare register 2<br>1 : Compare the received data with data compare register 2 | ○ | ○ |
| CMP3E | Data compare function 3 select bit | 0 : Does not compare the received data with data compare register 3 (Note 2)<br>1 : Compare the received data with data compare register 3 | ○ | ○ |
| RCRCE | Receive CRC enable bit | 0 : Not enable<br>1 : Enable | ○ | ○ |
| RSHTE | Receive shift operation enable bit | 0 : Receive shift operation disabled<br>1 : Receive shift operation enabled | ○ | ○ |
| RBSF0 | Receive bit stuffing "1" delete select bit | 0 : "1" is not deleted<br>1 : "1" is deleted | ○ | ○ |
| RBSF1 | Receive bit stuffing "1" delete select bit | 0 : "0" is not deleted<br>1 : "0" is deleted | ○ | ○ |

Note 1: Other than when in the special communication mode or HDLC data processing mode, either use the reset state as is or write "0016".
Note 2: To use the CRC initialization function (when bit 2 of SI/O expansion mode register is set to "1"), set bit 3 to "1".

## Group i special communication interrupt detect register (i=0,1) (Note)




| Symbol | Address | When reset |
|---|---|---|
| GiIRF (i=0,1) | 00FE16, 013E16 | 0000 00XX2 |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| ——— | Nothing is assigned. When write, set "0". When read, their contents are indeterminate. | | — | — |
| ——— | | | — | — |
| BSERR | Bit stuffing error detecting flag | 0 : Not detected<br>1 : Detected | ○ | ○ |
| ABT | Arbitration lost detecting flag | 0 : Not detected<br>1 : Detected | ○ | ○ |
| IRF0 | Interrupt cause determination flag 0 | 0 : Received data does not match data compare register 0<br>1 : Received data matches data compare register 0 | ○ | ○ |
| IRF1 | Interrupt cause determination flag 1 | 0 : Received data does not match data compare register 1<br>1 : Received data matches data compare register 1 | ○ | ○ |
| IRF2 | Interrupt cause determination flag 2 | 0 : Received data does not match data compare register 2<br>1 : Received data matches data compare register 2 | ○ | ○ |
| IRF3 | Interrupt cause determination flag 3 | 0 : Received data does not match data compare register 3<br>1 : Received data matches data compare register 3 | ○ | ○ |

Note : Other than when in the special communication mode or HDLC data processing mode, either use the reset state as is or write "0016".

**Figure 1. 23. 29. Group 0 and 1 related register (4)**

## Group i transmit buffer(receive data) register (i=0,1)

b7 b0

| Symbol | Address | When reset |
|---|---|---|
| GiTB (i=0,1) | $00EA_{16}$, $012A_{16}$ | Indeterminate |

| Function | Setting range | R | W |
|---|---|---|---|
| Transmit data for data compare is stored<br>Receive data for data compare is stored | —— | ○ | ○ |

## Group i data compare register j (i=0,1/j=0 to 3)

b7 b0

| Symbol | Address | When reset |
|---|---|---|
| GiCMPj (i=0/j=o to 3) | $00F0_{16}$, $00F1_{16}$, $00F2_{16}$, $00F3_{16}$ | Indeterminate |
| GiCMPj (i=1/j=o to 3) | $0130_{16}$, $0131_{16}$, $0132_{16}$, $0133_{16}$ | Indeterminate |

| Function | Setting range | R | W |
|---|---|---|---|
| Compare data | $00_{16}$ to $FF_{16}$ | ○ | ○ |

Note : When using the data compare registers 0 and 1, the data mask registers 0 and 1 must be set.

## Group i data mask register j (i=0,1/j=0,1)

b7 b0

| Symbol | Address | When reset |
|---|---|---|
| GiMSKj (i=0/j=0, 1) | $00F4_{16}$, $00F5_{16}$ | Indeterminate |
| GiMSKj (i=1/j=0, 1) | $0134_{16}$, $0135_{16}$ | Indeterminate |

| Function | Setting range | R | W |
|---|---|---|---|
| Mask data for receive data (masked by "1") | $00_{16}$ to $FF_{16}$ | ○ | ○ |

## Group i transmit CRC code register (i=0,1)

b15 b8
(b7) (b0) b7 b0

| Symbol | Address | When reset |
|---|---|---|
| GiTCRC (i=0, 1) | $00FB_{16}$,$00FA_{16}$, $013B_{16}$, $013A_{16}$ | $0000_{16}$ |

| Function | Setting range | R | W |
|---|---|---|---|
| Transmit CRC calculation results (Note) | —— | ○ | — |

Note : Computed results are initialized when the transmit CRC enable bit (bit 4 of group i expanded transmit control register) is set to "0".

## Group i receive CRC code register (i=0,1)

b15 b8
(b7) (b0) b7 b0

| Symbol | Address | When reset |
|---|---|---|
| GiRCRC (i=0, 1) | $00F9_{16}$,$00F8_{16}$, $0139_{16}$, $0138_{16}$ | $0000_{16}$ |

| Function | Setting range | R | W |
|---|---|---|---|
| Receive CRC calculation results | —— | ○ | — |

Note 1: Computed results are initialized when the receive CRC enable bit (bit 4 of group i expanded reseive control register) is set to "0", or when the CRC initialization bit (bit 2 of group i SI/O expansion mode register) is set to "1" and values match the data comparison register.
Note 2: Initialize to selected value when starting to receive.

**Figure 1. 23. 30. Group 0 and 1 related register (5)**

## • Clock synchronous serial I/O mode (group 0 and 1)

Table 1.23.12 gives specifications for the clock synchronous serial I/O mode.

**Table 1.23.12. Specifications of clock synchronous serial I/O mode (group 0 and 1)**

| Item | Specification |
|---|---|
| Transfer data format | • Transfer data length: 8 bits fixed |
| Transfer clock | • When internal clock is selected<br>– Transfer speed is determined when the base timer is reset by the ch0 WG function<br>Transfer rate (bps) = base timer count source (frequency) / (k+2) / 2<br>k : values set to WG register 0<br>– Transfer clock is generated when the transfer clock in the phase delayed waveform output mode<br>Transmit clock : ch3 WG function<br>Receive clock : ch2 WG function<br>Sets the same value in the WG registers on ch2 and ch3<br>• When external clock is selected<br>– Transfer rate (bps) = Clock input to ISCLK pin |
| Transmission start condition | To start transmission, the following requirements must be met:<br>• Transmit enable bit = “1”<br>• Write data to transmit buffer |
| Reception start condition | To start reception, the following requirements must be met:<br>• Receive enable bit = “1” |
| Interrupt request generation timing | • When transmitting<br>– When transmit buffer is empty, transmit interrupt cause select bit = “0”<br>– When transmission is completed, transmit interrupt cause select bit = “1”<br>• When receiving<br>When data is transferred to SI/O receive buffer register |
| Error detection | • Overrun error<br>This error occurs when the next data is ready before the contents of SI/O receive buffer register are read out |
| Select function | • LSB first/MSB first selection<br>When transmission/reception begins with bit 0 or bit 7, it can be selected<br>• Transmit/receive data polarity switching<br>This function is reversing ISTxD pin output and ISRxD pin input.<br>(All I/O data level is reversed.) |

Note: Set the transmission clock to at least 6 divisions of the base timer clock.

Table 1.23.13 lists I/O pin functions for the clock synchronous serial I/O mode of groups 0 and 1.
From when the operating mode is selected until transmission starts, the ISTxDi pin is "H" level. Figure 1.23.31 shows typical transmit/receive timings in clock synchronous serial I/O mode in group 0 and 1.

**Table 1.23.13. I/O pin functions in clock synchronous serial I/O mode of group 0, group 1**

| Pin name | Function | Selected method |
|---|---|---|
| ISTxD<br>($P7_6$, $P15_0$, $P7_3$, $P11_0$) | Serial data output | • Use the ch0 WG function<br>• Sets "111" for the operating mode select bit (bits 2, 1 and 0) in WG control register 0<br>• Selects ISTxD output for the port using function select registers A, B and C |
| ISRxD<br>($P8_0$, $P15_2$, $P7_5$, $P11_2$) | Serial data input | • Selects a using port with input function select register<br>• Selects I/O with function select register A<br>• Sets a selected port to input using the port direction register |
| ISCLK<br>($P7_7$, $P15_1$, $P7_4$, $P11_1$) | Transfer clock output | • Use the ch1 WG function<br>• Sets "111" for the operating mode select bit (bits 2, 1 and 0) in WG control register 1<br>• Sets "0" for the internal/external clock select bit (bit 2) of the SI/O communication mode register<br>• Selects ISCLK output for the port using function select registers A, B and C |
| | Transfer clock input | • Selects a using port with input function select register |
| | | • Sets "1" for the internal/external clock select bit (bit 2) of the SI/O communication mode register<br>• Sets a selected port to input using the port direction register<br>• Selects I/O port with function select register A |

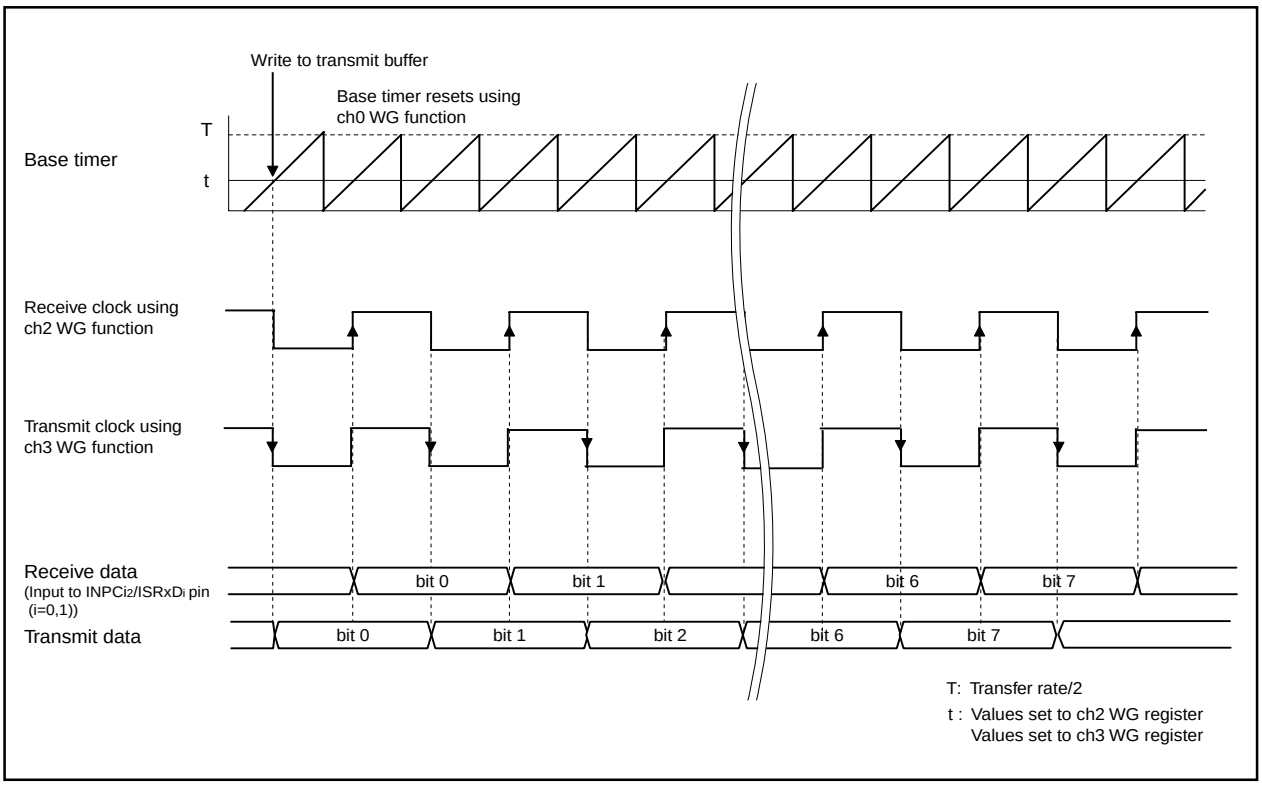



**Figure 1.23.31. Typical transmit/receive timings in clock synchronous serial I/O mode in group 0 and 1**

## (2) Clock asynchronous serial I/O mode (UART) (group 0 and 1)

Table 1.23.14 lists the specifications for the UART mode.

**Table 1.23.14. Specifications of UART mode**

| Item | Specification |
|---|---|
| Transfer data format | • Character bit (transfer data) : 8 bits<br>• Start bit : 1 bit<br>• Stop bit : 1 bit or 2 bits selected |
| Transfer clock | • When internal clock is selected (Generates the transmit/receive clock in the phase delayed waveform output mode)<br>– Transfer speed is determined when the base timer is reset by the ch0 WG function<br>Transfer rate (bps) = base timer count source (frequency) / (k+2) / 2<br>k : values set to WG register 0<br>– Transfer clock is generated when the transfer clock in the phase delayed waveform output mode<br>Transmit clock : ch3 WG function<br>Receive clock : Change ch2 TM function to WG function<br>Detects falling edge of start bit<br>Changes to the WG mode when the time measurement interrupt arrives<br>• When external clock is selected<br>– Transfer rate (bps) = Clock input to ISCLK pin |
| Transmission start condition | To start transmission, the following requirements must be met:<br>• Transmit enable bit = “1”<br>• Write data to transmit buffer |
| Reception start condition | To start reception, the following requirements must be met:<br>• Receive enable bit = “1” |
| Interrupt request generation timing | • When transmitting<br>– When transmit buffer is empty, transmit interrupt cause select bit = “0”<br>– When transmission is completed, transmit interrupt cause select bit = “1”<br>• When receiving<br>– When data is transferred to SI/O receive buffer register |
| Error detection | • Overrun error : This error occurs when the next data is ready before contents of SI/O receive buffer register are read out<br>• Framing error : This error occurs when the number of stop bits set is not detected |
| Select function | • Stop bit length : Stop bit length can be selected as 1 bit or 2 bits<br>• LSB first/MSB first selection :<br>Whether transmit/receive begins with bit 0 or bit 7 can be selected<br>• Transmit/receive data polarity switching :<br>This function is reversing ISTxD port output and ISRxD port input. (All I/O data level are reversed.)<br>• Data transfer bit length : Transmission data length can be set between 1 to 8 bits |

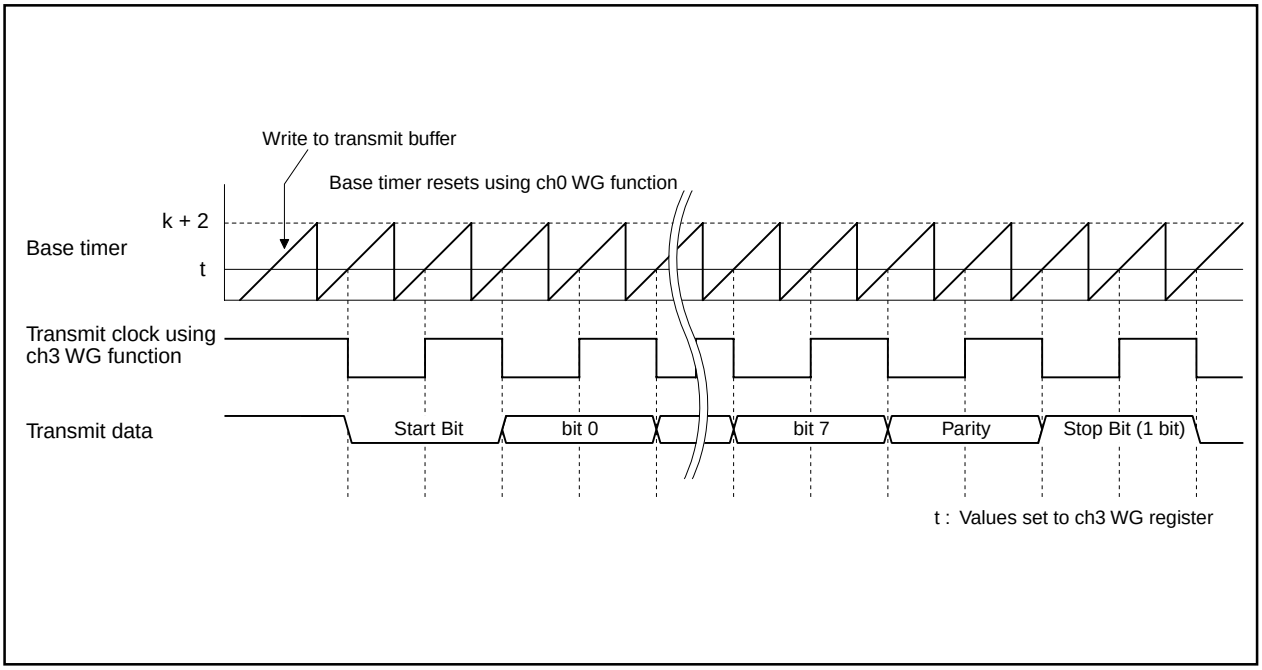



**Figure 1.23.32. Typical transmit timings in UART mode**

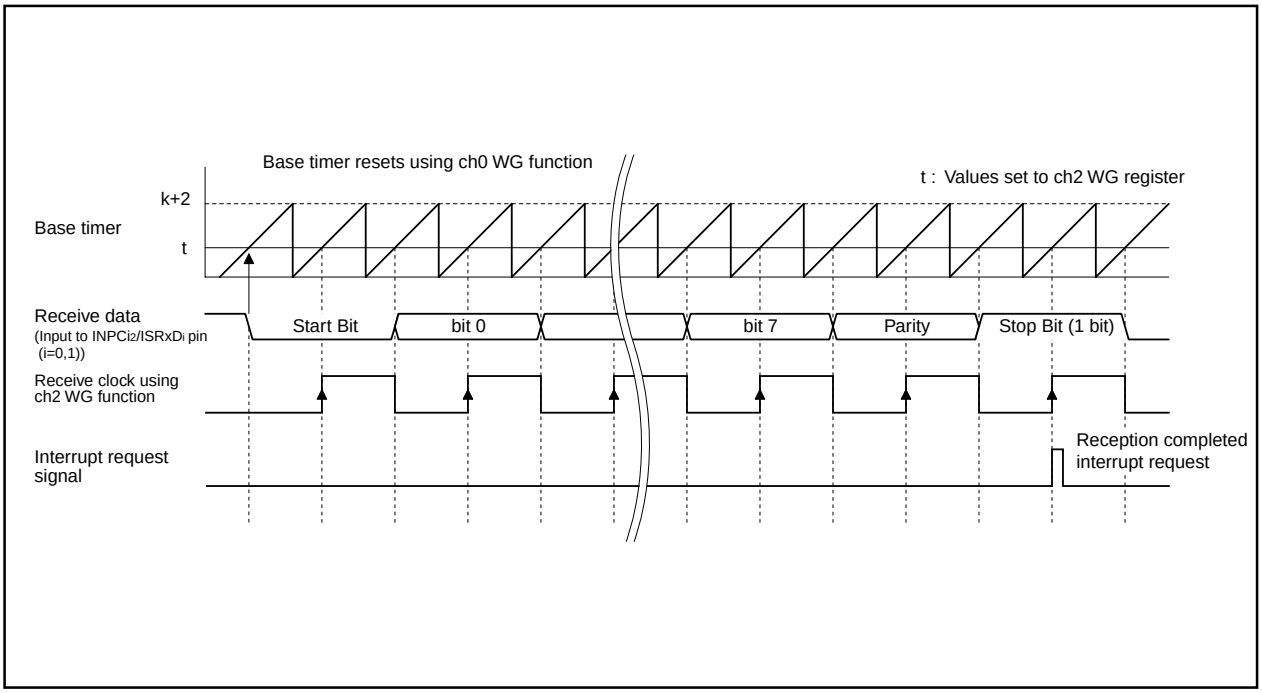



**Figure 1.23.33. Typical receive timing in UART mode**

**TxD, RxD I/O polarity reverse function**

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) are reversed. TxD output polarity reverse select bit is set to “1” (reverse) for usual use.

## (2) Clock synchronous serial I/O mode (group 2)

Intelligent I/O groups 2 has communication block that have two internal 8-bit shift registers. When used in conjunction with the communication block and WG function, these shift registers enable variable clock synchronous and IE Bus (Note) communications.

Table 1.23.16 lists using registers in group 2, figure 1.23.34 to 1.23.37 shows the related registers.

Note : IE Bus is a trademark of NEC corporation.

**Table 1.23.16. Using registers in group 2**

| | Clock synchronous serial I/O | IE Bus |
|---|---|---|
| Base timer control register 0 | √ | √ |
| Base timer control register 1 | √ | √ |
| Waveform generate control register 0 | √ | √ |
| Waveform generate control register 1 | – | √ |
| Waveform generate control register 2 | √ | √ |
| Waveform generate control register 3 | – | √ |
| Waveform generate control register 4 | – | √ (Note 1) |
| Waveform generate control register 5 | – | √ |
| Waveform generate control register 6 | – | √ |
| Waveform generate control register 7 | – | √ |
| Waveform generate register 0 | √ | √ |
| Waveform generate register 1 | – | √ |
| Waveform generate register 2 | √ | √ |
| Waveform generate register 3 | – | √ |
| Waveform generate register 4 | – | √ |
| Waveform generate register 5 | – | √ |
| Waveform generate register 6 | – | √ |
| Waveform generate register 7 | – | √ |
| Function enable register | √ | √ |
| SI/O communication mode register | √ | √ |
| SI/O communication control register | √ | √ |
| IE Bus control register | – | √ |
| IE Bus address register | – | √ |
| IE Bus transmit interrupt cause detect register | – | √ |
| IE Bus receive interrupt cause detect register | – | √ |
| SI/O receive buffer register | √ | √ |
| SI/O transmit buffer register | √ | √ |

√ : Use – : Not use

Note 1: When receiving slave, set corresponding value with 32.5 µs. Don't set 170 µs.

## Group 2 SI/O transmit buffer register

b15 b14 b13 b12 b11 b10 b9 b8
(b7)(b6)(b5)(b4)(b3)(b2)(b1)(b0) b7 ... b0

| Symbol | Address | When reset |
|---|---|---|
| G2TB | 016D16, 016C16 | Indeterminate |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| —— | Transmit buffer | Transmit data | – | ○ |
| SZ0 | Transfer bit length select bit | b10 b9 b8<br>0 0 0 : 8-bit long<br>0 0 1 : 1-bit long<br>0 1 0 : 2-bit long<br>0 1 1 : 3-bit long<br>1 0 0 : 4-bit long<br>1 0 1 : 5-bit long<br>1 1 0 : 6-bit long<br>1 1 1 : 7-bit long | ○ | ○ |
| SZ1 | | | ○ | ○ |
| SZ2 | | | ○ | ○ |
| —— | Nothing is assigned. When write, set "0". When read, their contents are indeterminate. | | – | – |
| —— | | | – | – |
| A | ACK function select bit | 0 : No function<br>1 : Adds an ACK bit after the final transmission bit | ○ | ○ |
| PC | Parity operation continuing bit (Note) | 0 : Adds the parity bit after the transmitted data<br>1 : Repeats the parity check with the next transmission | ○ | ○ |
| P | Parity function select bit | 0 : No parity<br>1 : Parity (Only even parity) | ○ | ○ |

Note: When this bit is set to "1", set the parity function select bit to "0".

## Group 2 SI/O receive buffer register

b15 b14 b13 b12 b11 b10 b9 b8
(b7)(b6)(b5)(b4)(b3)(b2)(b1)(b0) b7 ... b0

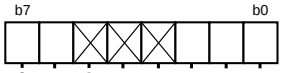
| Symbol | Address | When reset |
|---|---|---|
| G2RB | 016F16, 016E16 | Indeterminate |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| —— | Receive buffer | Receive data | ○ | – |
| —— | Nothing is assigned. When write, set "0". When read, their contents are indeterminate. | | – | – |
| OER | Overrun error flag (Note) | 0 : No overrun error<br>1 : Overrun error found | ○ | – |
| —— | Nothing is assigned. When write, set "0". When read, their contents are indeterminate. | | – | – |

Note : This bit is automatically set to "0" when communication unit reset is selected for the communication mode select bit and the reception enable bit is set to "0".

**Figure 1. 23. 34. Group 2 Intelligent I/O-related register (1)**

## Group 2 IE Bus control register

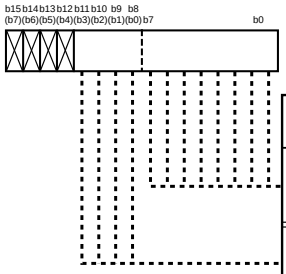



| Symbol | Address | When reset |
|---|---|---|
| IECR | $0172_{16}$ | $00XXX000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| IEB | IE Bus enable bit | 0 : IE Bus disabled (Note)<br>1 : IE Bus enabled | ○ | ○ |
| IETS | IE Bus transmit start request bit | 0 : Transmit completed<br>1 : Transmit start | ○ | ○ |
| IEBBS | IE Bus busy flag | 0 : Idle state<br>1 : Busy state (start condition detected) | ○ | — |
| —— | Nothing is assigned. When write, set "0".<br>When read, their contents are indeterminate. | | — | — |
| —— | | | — | — |
| —— | | | — | — |
| DF | Digital filter select bit | 0 : No digital filter<br>1 : Digital filter | ○ | ○ |
| IEM | IE Bus mode select bit | 0 : Mode 1<br>1 : Mode 2 | ○ | ○ |

Note :When this bit is set to "0", hold "0" for at least 1 cycle of base timer .

## Group 2 IE Bus address register

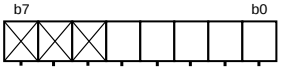



| Symbol | Address | When reset |
|---|---|---|
| IEAR | $0171_{16}$, $0170_{16}$ | Indeterminate |

| Function | R | W |
|---|---|---|
| Address data | ○ | ○ |
| Address data | ○ | ○ |
| Nothing is assigned. When write, set "0".<br>When read, their contents are indeterminate. | — | — |
| | — | — |
| | — | — |
| | — | — |

**Figure 1. 23. 35. Group 2 Intelligent I/O-related register (2)**

## Group 2 IE Bus transmit interrupt cause determination register

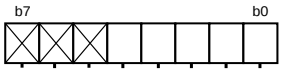



| Symbol | Address | When reset |
|---|---|---|
| IETIF | $0173_{16}$ | $XXX00000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| IETNF | Normal termination flag | 0 : Terminated in error<br>1 : Terminated normally (Note) | ○ | ○ |
| IEACK | ACK error flag | 0 : No error<br>1 : Error found (Note) | ○ | ○ |
| IETMB | Max. transfer byte error flag | 0 : No error<br>1 : Error found (Note) | ○ | ○ |
| IETT | Timing error flag | 0 : No error<br>1 : Error found (Note) | ○ | ○ |
| IEABL | Arbitration lost flag | 0 : No error<br>1 : Error found (Note) | ○ | ○ |
| —— | Nothing is assigned. When write, set "0".<br>When read, their contents are indeterminate. | | – | – |
| —— | | | – | – |
| —— | | | – | – |

Note : Only "0" can be written for this bit. Also, it is cleared to "0" when "0" is written for bit 0 of the IE Bus control register. At this time, hold "0" for at least 1 cycle of base timer clock.

## Group 2 IE Bus receive interrupt cause determination register



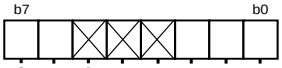

| Symbol | Address | When reset |
|---|---|---|
| IERIF | $0174_{16}$ | $XXX00000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| IERNF | Normal termination flag | 0 : Terminated in error<br>1 : Terminated normally (Note) | ○ | ○ |
| IEPAR | Parity error flag | 0 : No error<br>1 : Error found (Note) | ○ | ○ |
| IERMB | Max. transfer byte error flag | 0 : No error<br>1 : Error found (Note) | ○ | ○ |
| IERT | Timing error flag | 0 : No error<br>1 : Error found (Note) | ○ | ○ |
| IERETC | Other cause receive completed flag | 0 : No error<br>1 : Error found (Note) | ○ | ○ |
| —— | Nothing is assigned. When write, set "0".<br>When read, their contents are indeterminate. | | – | – |
| —— | | | – | – |
| —— | | | – | – |

Note : Only "0" can be written for this bit. Also, it is cleared to "0" when "0" is written for bit 0 of the IE Bus control register. At this time, hold "0" for at least 1 cycle of base timer clock.

**Figure 1. 23. 36. Group 2 Intelligent I/O-related register (3)**

## Group 2 SI/O communication mode register

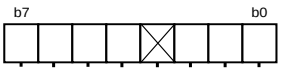



| Symbol | Address | When reset |
|---|---|---|
| G2MR | $016A_{16}$ | $00XXX000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| GMD0 | Communication mode select bit | b1 b0<br>0 0 : Communication part is reset (Overrun error flag is cleared)<br>0 1 : Serial I/O mode<br>1 0 : Special communication mode<br>1 1 : HDLC data process mode | ○ | ○ |
| GMD1 | | | ○ | ○ |
| CKDIR | Internal/external clock select bit | 0 : Internal clock (Note 2)<br>1 : External clock (Note 3) | ○ | ○ |
| ——— | Nothing is assigned. When write, set "0". When read, their contents are indeterminate. | | – | – |
| ——— | | | – | – |
| ——— | | | – | – |
| UFORM | Transfer direction select bit | 0 : LSB first<br>1 : MSB first | ○ | ○ |
| IRS | Transmit interrupt cause select bit | 0 : Transmit buffer is empty<br>1 : Transmit is completed | ○ | ○ |

Note 1: Intelligent I/O group 2 has IE bus communication function as special communication function.
Note 2: Select a pin for clock output by setting the waveform generation control register, input function select register, and function select registers A, B and C. Data transmission pins are the same as clock output pins.
Note 3: Select which pins will input the clock with the input function select register and set those pins to the input port using function select register A. Data input pins are the same as with clock input pins.

## Group 2 SI/O communication control register




| Symbol | Address | When reset |
|---|---|---|
| G2CR | $016B_{16}$ | $0000\ X000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TE | Transmit enable bit | 0 : Transmission disabled<br>1 : Transmission enabled | ○ | ○ |
| TXEPT | Transmit register empty flag | 0 : Data present in transmit register (during transmission)<br>1 : No data present in transmit register (transmission completed) | ○ | – |
| TI | Transmit buffer empty flag | 0 : Data present in transmit buffer register<br>1 : No data present in transmit buffer register | ○ | – |
| ——— | Nothing is assigned. When write, set to "0". When read, the contents is indeterminate. | | – | – |
| RE | Receive enable bit | 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 | ○ | – |
| OPOL | TxD output polarity reverse select bit | 0 : No reverse (Usually set to "0")<br>1 : Reverse | ○ | ○ |
| IPOL | RxD input polarity reverse select bit | 0 : No reverse (Usually set to "0")<br>1 : Reverse | ○ | ○ |

**Figure 1. 23. 37. Group 2 Intelligent I/O-related register (4)**

## • Clock synchronous serial I/O mode (group 2)

Table 1.23.17 gives specifications for the group 2 clock synchronous serial I/O mode.

**Table 1.23.17. Specifications of clock synchronous serial I/O mode**

| Item | Specification |
|---|---|
| Transfer data format | • Transfer data length: Variable length (group2) |
| Transfer clock | • When internal clock is selected, the transfer clock in the single waveform output mode is generated.<br>– Transfer speed is determined when the base timer is reset by the ch0 WG function<br>Transfer rate (bps) = base timer count source (frequency) / (k+2)<br>k : values set to WG register 0<br>– Transfer clock is generated by ch2 single phase WG function<br>Ch3 WG register = (k+2)/2 (Note 1)<br>• When external clock is selected<br>– Transfer rate (bps) = Clock input to ISCLK pin (Note 2) |
| Transmission start condition | • To start transmission, the following requirements must be met:<br>– Transmit enable bit = “1”<br>– Write data to SI/O transmit buffer register |
| Reception start condition | • To start reception, the following requirements must be met:<br>– Receive enable bit = “1”<br>– Transmit enable bit = “1”<br>– Write data to SI/O transmit buffer register |
| Interrupt request generation timing | • When transmitting<br>– When SI/O communication buffer register is empty, transmit interrupt cause select bit = “0”<br>– When transmission is completed, transmit interrupt cause select bit = “1”<br>• When receiving<br>– When data is transferred to SI/O receive buffer register |
| Error detection | • Overrun error<br>This error occurs when the next data is ready before the contents of SI/O receive buffer register are read out |
| Select function | • LSB first/MSB first selection<br>When transmission/reception begins with bit 0 or bit 7, it can be selected.<br>• Transmit/receive data polarity switching<br>– This function is reversing ISTxD pin output and ISRxD pin input.<br>(All I/O data level is reversed.)<br>• Data transfer bit length<br>– Transmission data length can be set between 1 to 8 bits |

Note 1: When the transfer clock and transfer data are transmission, transfer clock is set to at least 6 divisions of the base timer clock. Except this, transfer clock is set to at least 20 divisions of the base timer clock.

Note 2: Transfer clock is set to at least 20 divisions of the base timer clock.

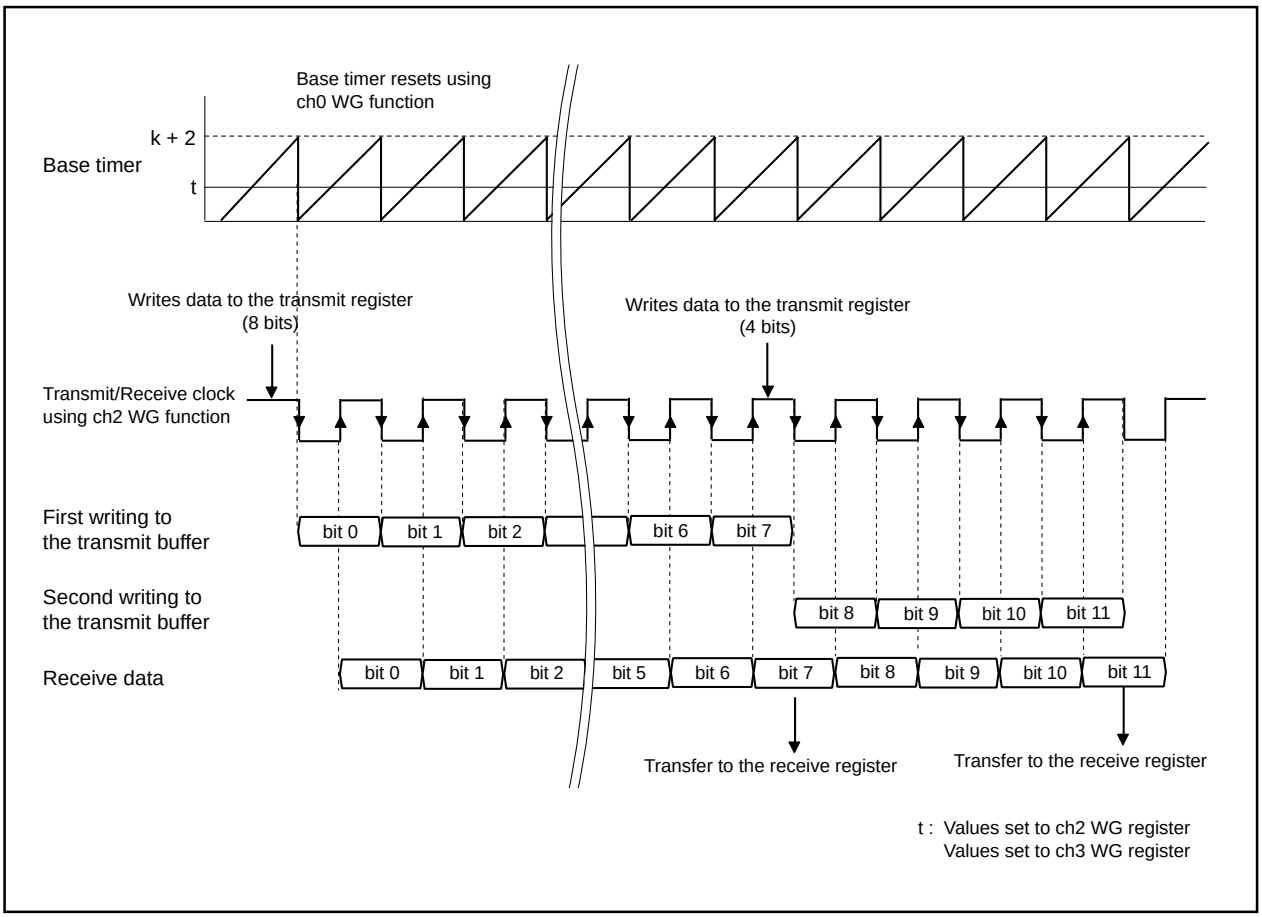



**Figure 1. 23. 38. Typical transmit/receive timings in clock synchronous serial I/O mode in group 2**

## A-D Converter

The A-D converter consists of two 10-bit successive approximation A-D converter circuit with a capacitive coupling amplifier. Pins $P10_0$ to $P10_7$, $P15_0$ to $P15_7$, $P0_0$ to $P0_7$, $P2_0$ to $P2_7$, $P9_5$, and $P9_6$ are shared as the analog signal input pins. Pins $P15_0$ to $P15_7$, $P0_0$ to $P0_7$ and $P2_0$ to $P2_7$ can be used as the analog signal input pins and switched by analog input port select bit. However, $P0_0$ to $P0_7$ and $P2_0$ to $P2_7$ can be used in single chip mode. Set input to direction register corresponding to a pin doing A-D conversion.

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

Table 1.24.1 shows the performance of the A-D converter. Figure 1.24.1 shows the block diagram of the A-D converter, and Figures 1.24.2 to 1.24.7 show the A-D converter-related registers.

This section is described to 144-pin version as example.

In 100-pin version, $AN1_0$ to $AN1_7$ cannot be selected because there is no P15.

## A-D Converter

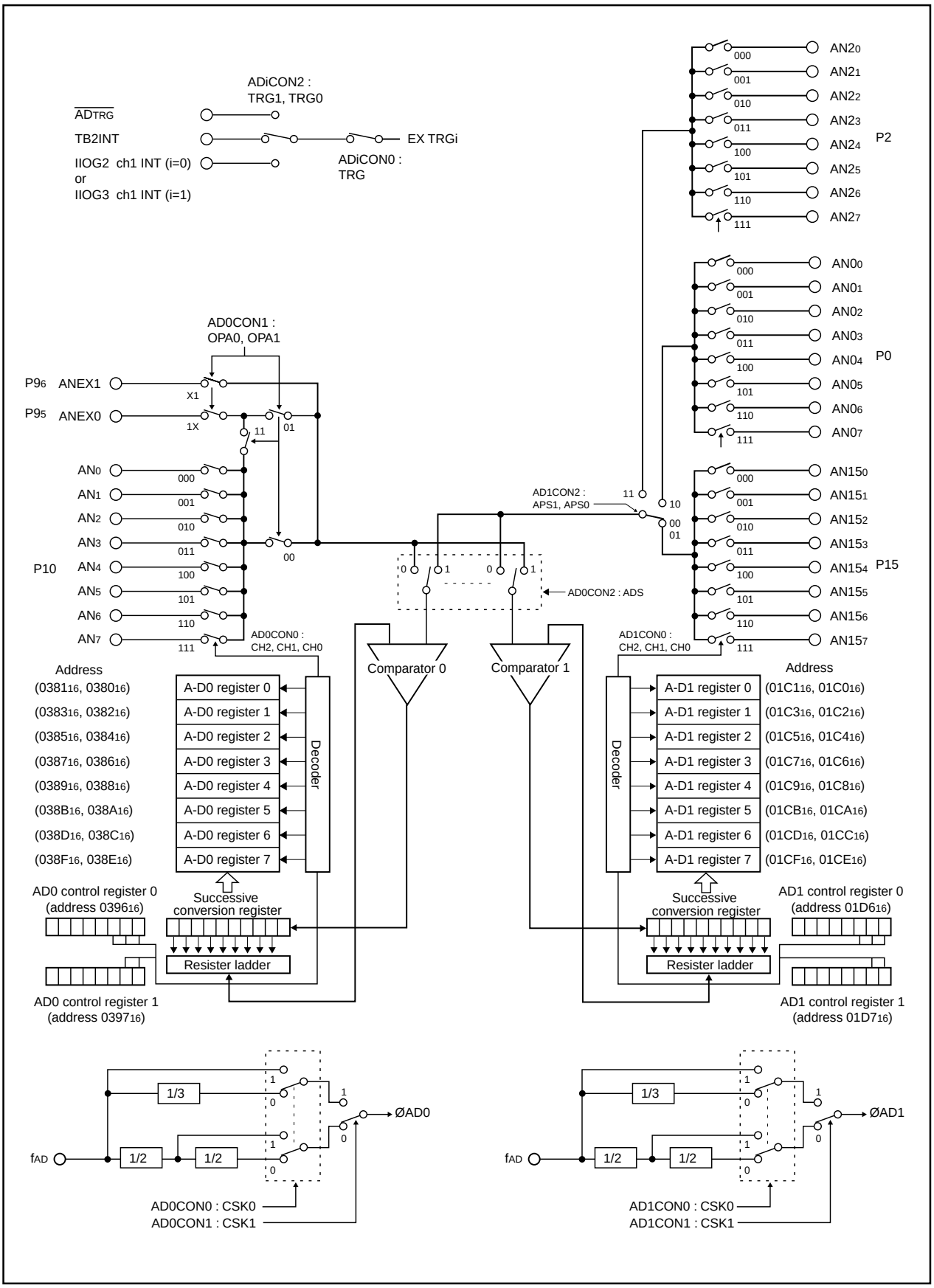



**Figure 1.24.1. Block diagram of A-D converter**

**Table 1.24.1. Performance of A-D converter**

| Item | Performance |
|---|---|
| Method of A-D conversion | Successive approximation (capacitive coupling amplifier) |
| Analog input voltage (Note 1) | 0V to AVCC (VCC) |
| Operating clock ØAD (Note 2) | fAD, fAD/2, fAD/3 , fAD/4 fAD=f(XIN) |
| Resolution | 8-bit or 10-bit (selectable) |
| Operating modes | One-shot mode, repeat mode, single sweep mode, repeat sweep mode 0, and repeat sweep mode 1 |
| Analog input pins | 34 pins<br>AN, AN0, AN2, AN15(Note 3) each 8 pins<br>Extended input 2 pins (ANEX0(Note 4) and ANEX1(Note 5)) |
| 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 by outbreak of the following factors chosen among in three (Note 6)<br>· ADTRG/P97 input changes from “H” to “L”<br>· Timer B2 interrupt occurrences frequency counter overflow<br>· Interrupt of Intelligent I/O group 2 or 3 channel 1 |
| Conversion speed per pin | • Without sample and hold function<br>8-bit resolution: 49 ØAD cycles<br>10-bit resolution: 59 ØAD cycles<br>• With sample and hold function<br>8-bit resolution: 28 ØAD cycles<br>10-bit resolution: 33 ØAD cycles |

Note 1: Does not depend on use of sample and hold function.
Note 2: When f(XIN) is over 10 MHz, the fAD frequency must be under 10 MHz by dividing.
Without sample and hold function, set the fAD frequency to 250kHz or more.
With the sample and hold function, set the fAD frequency to 1MHz or more.
Note 3: When port P15 is used as analog input port, port P15 input peripheral function select bit (bit 2 of address $0178_{16}$) must set to be "1".
Note 4: When port P95 is used as analog input port, port P95 output peripheral function select bit (bit 5 of address $03B7_{16}$) must set to be "1".
Note 5: When port P96 is used as analog input port, port P96 output peripheral function select bit (bit 6 of address $03B7_{16}$) must set to be "1".
Note 6: Set the port direction register to input.

## A-D0 control register 0 (Note 1)

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

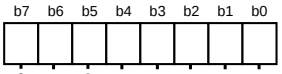
| Symbol | Address | When reset |
|---|---|---|
| AD0CON0 | $0396_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CH0 | Analog input pin select bit | b2 b1 b0<br>0 0 0 : AN0<br>0 0 1 : AN1<br>0 1 0 : AN2<br>0 1 1 : AN3 (Note 2, 3)<br>1 0 0 : AN4<br>1 0 1 : AN5<br>1 1 0 : AN6<br>1 1 1 : AN7 | ○ | ○ |
| 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 (Note 2)<br>1 1 : Repeat sweep mode 0<br>Repeat sweep mode 1 | ○ | ○ |
| MD1 | | | ○ | ○ |
| TRG | Trigger select bit | 0 : Software trigger<br>1 : External trigger (Note 4) | ○ | ○ |
| ADST | A-D conversion start flag | 0 : A-D conversion disabled<br>1 : A-D conversion started (Note 5) | ○ | ○ |
| CKS0 | Frequency select bit (Note 6) | 0 : $f_{AD}/3$ or $f_{AD}/4$ is selected<br>1 : $f_{AD}/1$ or $f_{AD}/2$ is selected | ○ | ○ |

Note 1: If the A-D0 control register 0 is rewritten during A-D conversion, the conversion result is indeterminate.
Note 2: When changing A-D operation mode, set analog input pin again.
Note 3: This bit is disabled in single sweep mode, repeat sweep mode 0 and repeat sweep mode 1.
Note 4: External trigger request cause can be selected in external trigger request cause select bit (bit5 and bit 6 of address $0394_{16}$).
Note 5: When External trigger is selected, set to "1" after selecting the external trigger request cause using the external trigger request cause select bit.
Note 6: When f($X_{IN}$) is over 10 MHz, the $\varnothing_{AD}$ frequency must be under 10 MHz by dividing.

**Figure 1.24.2. A-D converter-related registers (1)**

## A-D0 control register 1 (Note 1)




| Symbol | Address | When reset |
|---|---|---|
| AD0CON1 | $0397_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SCAN0 | A-D sweep pin select bit | b0 b1<br>0 0 : ANj0, ANj1 (ANj0)<br>0 1 : ANj0 to ANj3 (ANj0, ANj1) (Note 2, 3)<br>1 0 : ANj0 to ANj5 (ANj0 to ANj2)<br>1 1 : ANj0 to ANj7 (ANj0 to ANj3) | ○ | ○ |
| SCAN1 | | | ○ | ○ |
| MD2 | A-D operation mode select bit 1 | 0 : Any mode other than repeat sweep mode 1<br>1 : Repeat sweep mode 1 | ○ | ○ |
| BITS | 8/10-bit mode select bit | 0 : 8-bit mode<br>1 : 10-bit mode | ○ | ○ |
| CKS1 | Frequency select bit (Note 3) | 0 : fAD/2 or fAD/4 is selected<br>1 : fAD/1 or fAD/3 is selected | ○ | ○ |
| VCUT | VREF connect bit | 0 : VREF not connectec<br>1 : VREF connectec | ○ | ○ |
| OPA0 | External op-amp connection mode bit (Note 4) | b6 b7<br>0 0 : ANEX0 and ANEX1 are not used (Note 5)<br>0 1 : ANEX0 input is A-D converted (Note 6)<br>1 0 : ANEX1 input is A-D converted (Note 7)<br>1 1 : External op-amp connection mode (Note 8) | ○ | ○ |
| OPA1 | | | ○ | ○ |

Note 1: If the A-D0 control register 1 is rewritten during A-D conversion, the conversion result is indeterminate.
Note 2: This bit is invalid in One-shot mode and Repeat mode. Channel shown in the parentheses, becomes valid when repeat sweep mode 1(bit2="1") is selected.
Note 3: When f(XIN) is over 10 MHz, the ∅AD frequency must be under 10 MHz by dividing.
Note 4: In single sweep mode and repeat sweep mode 0, 1, bit 7 and bit 6 cannot be set "01" and "10".
Note 5: When this bit is set, set "00" to bit6 and bit5 of function select register B3.
Note 6: When this bit is set, set "1" to bit5 of function select register B3.
Note 7: When this bit is set, set "1" to bit6 of function select register B3.
Note 8: When this bit is set, set "11" to bit6 and bit5 of function select register B3.

**Figure 1.24.3. A-D converter-related registers (2)**

## A-D0 control register 2 (Note 1)

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

| Symbol | Address | When reset |
|---|---|---|
| AD0CON2 | $0394_{16}$ | $X000\ 0000_2$ |

| 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 | Must always set to "0" | ○ | ○ |
| —— | | | ○ | ○ |
| —— | | | ○ | ○ |
| ADS | A-D Channel replace select bit (Note 2) | 0 : Channel replace is invalid<br>1 : Channel replace is valid | ○ | ○ |
| TRG0 | External trigger request cause select bit | b6 b5<br>0 0 : $\overline{ADTRG}$ is selected<br>0 1 : Timer B2 interrupt occurrence frequency counter overflow is selected (Note 3)<br>1 0 : Group 2 channel 1 interrupt is selected<br>1 1 : Must not be set | ○ | ○ |
| TRG1 | | | ○ | ○ |
| PST | Simultaneous start bit (Note 4) (Note 5) (Note 6) | 0 : Invalid<br>1 : 2 circuit A-D simultaneous start | — | ○ |

Note 1: If the A-D0 control register 2 is rewritten during A-D conversion, the conversion result is indeterminate
Note 2: When the A-D circuit of either of A-D0 and A-D1 are operated, do not write "1" to this bit.
Note 3: This bit is valid when software trigger is selected.
Note 4: When this bit read, the value is indeterminate.
Note 5: This is valid in three-phase PWM mode.
Note 6: Turn every setting of A-D0 and A-D1 into same, and start at the same time in sweep mode.

## A-D0 register j (j=0 to 7)

| Symbol | Address | When reset |
|---|---|---|
| AD0j(j=0 to 2) | $0381_{16}$,$0380_{16}$, $0383_{16}$,$0382_{16}$, $0385_{16}$,$0384_{16}$ | indeterminate |
| AD0j(j=3 to 5) | $0387_{16}$,$0386_{16}$, $0389_{16}$,$0388_{16}$, $038B_{16}$,$038A_{16}$ | indeterminate |
| AD0j(j=6,7) | $038D_{16}$,$038C_{16}$, $038F_{16}$,$038E_{16}$ | indeterminate |

| Function | R | W |
|---|---|---|
| Eight low-order bits of A-D conversion result | ○ | — |
| During 10-bit mode : Two high-order bits of A-D conversion result<br>During 8-bit mode : When read, their contents are indeterminate | ○ | — |
| Nothing is assigned.<br>When write, set "0". When read, their contents are indeterminate. | — | — |

**Figure 1.24.4. A-D converter-related registers (3)**

## A-D1 control register 0 (Note 1)

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

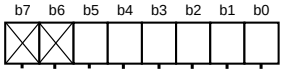
| Symbol | Address | When reset |
|---|---|---|
| AD1CON0 | $01D6_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CH0 | Analog input pin select bit | b2 b1 b0<br>0 0 0 : ANj0<br>0 0 1 : ANj1<br>0 1 0 : ANj2<br>0 1 1 : ANj3 (Note 2, 3, 4)<br>1 0 0 : ANj4<br>1 0 1 : ANj5<br>1 1 0 : ANj6<br>1 1 1 : ANj7 (j=0, 2, 15) | ○ | ○ |
| 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 (Note 2)<br>1 1 : Repeat sweep mode 0<br>Repeat sweep mode 1 | ○ | ○ |
| MD1 | | | ○ | ○ |
| TRG | Trigger select bit | 0 : Software trigger<br>1 : External trigger (Note 5, 6) | ○ | ○ |
| ADST | A-D conversion start flag | 0 : A-D conversion disabled<br>1 : A-D conversion started (Note 7) | ○ | ○ |
| CKS0 | Frequency select bit (Note 8) | 0 : $f_{AD}/3$ or $f_{AD}/4$ is selected<br>1 : $f_{AD}/1$ or $f_{AD}/2$ is selected | ○ | ○ |

Note 1: If the A-D1 control register 0 is rewritten during A-D conversion, the conversion result is indeterminate.
Note 2: When changing A-D operation mode, set analog input pin again.
Note 3: This bit is disabled in single sweep mode, repeat sweep mode 0 and repeat sweep mode 1.
Note 4: j=0, 2, 15 is selected by analog input port select bits (bit1 and bit 2 of address $01D4_{16}$).
Note 5: External trigger request cause can be selected in external trigger request cause select bit (bit5 and bit 6 of address $01D4_{16}$).
Note 6: After selecting external trigger request cause, set to "1".
Note 7: When External trigger is selected, set to "1" after selecting the external trigger.
Note 8: When f($X_{IN}$) is over 10 MHz, the $\varnothing_{AD}$ frequency must be under 10 MHz by dividing.

**Figure 1.24.5. A-D converter-related registers (4)**

## A-D1 control register 1 (Note 1)



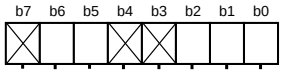

| Symbol | Address | When reset |
|---|---|---|
| AD1CON1 | $01D7_{16}$ | $XX000000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SCAN0 | A-D sweep pin select bit | b1 b0<br>0 0 : ANj0,ANj1 (ANj0)<br>0 1 : ANj0 to ANj3 (ANj0,ANj1) (Note 2, 3)<br>1 0 : ANj0 to ANj5 (ANj0 to ANj2)<br>1 1 : ANj0 to ANj7 (ANj0 to ANj3) (j=0, 2, 15) | ○ | ○ |
| SCAN1 | | | ○ | ○ |
| MD2 | A-D operation mode select bit 1 | 0 : Any mode except repeat sweep mode 1<br>1 : Repeat sweep mode 1 | ○ | ○ |
| BITS | 8/10-bit mode select bit | 0 : 8-bit mode<br>1 : 10-bit mode | ○ | ○ |
| CKS1 | Frequency select bit (Note 4) | 0 : $f_{AD}/2$ or $f_{AD}/4$ is selected<br>1 : $f_{AD}/1$ or $f_{AD}/3$ is selected | ○ | ○ |
| VCUT | $V_{REF}$ connect bit | 0 : $V_{REF}$ not connectec<br>1 : $V_{REF}$ connectec | ○ | ○ |
| —— | Nothing is assigned. When write, set to "0". When read, their contents are indeterminate. | | — | — |
| —— | | | — | — |

Note 1: If the A-D1 control register 1 is rewritten during A-D conversion, the conversion result is indeterminate.
Note 2: This bit is invalid in one-shot mode and repeat mode. Channel shown in the parentheses, becomes valid when repeat sweep mode 1(bit 2 = "1") is selected.
Note 3: j=0, 2, 15 is selected by analog input port select bits (bit1 and bit 2 of address $01D4_{16}$).
Note 4: When f($X_{IN}$) is over 10 MHz, the $\varnothing_{AD}$ frequency must be under 10 MHz by dividing.

**Figure 1.24.6. A-D converter-related registers (5)**

## A-D1 control register 2 (Note 1)

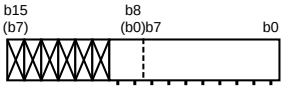



| Symbol | Address | When reset |
|---|---|---|
| AD1CON2 | $01D4_{16}$ | $X00XX000_2$ |

| Bit Symbol | Bit name | Bit name | R | W |
|---|---|---|---|---|
| SMP | A-D conversion method select bit | 0 : Without sample and hold<br>1 : With sample and hold | ○ | ○ |
| APS0 | Analog input port select bit | b2 b1<br>0 0 : P15<br>0 1 : Must not be set<br>1 0 : P0<br>1 1 : P2 | ○ | ○ |
| APS1 | | | ○ | ○ |
| ——— | Nothing is assigned.<br>When write, set to "0". When read, their contents are indeterminate. | | — | — |
| ——— | | | — | — |
| TRG0 | External trigger request cause select bit | b6 b5<br>0 0 : $\overline{\text{AD}_{\text{TRG}}}$ is selected<br>0 1 : Timer B2 interrupt occurrence frequency counter overflow is selected (Note 2)<br>1 0 : Group 3 channel 1 interrupt is selected<br>1 1 : Must not be set | ○ | ○ |
| TRG1 | | | ○ | ○ |
| ——— | Nothing is assigned.<br>When write, set to "0". When read, its content is indeterminate. | | — | — |

Note 1: If the A-D1 control register 2 is rewritten during A-D conversion, the conversion.
Note 2: This is valid in three-phase PWM mode.

## A-D1 register j (j=0 to 7)




| Symbol | Address | When reset |
|---|---|---|
| AD1j(j=0 to 2) | $01C1_{16}$,$01C0_{16}$, $01C3_{16}$,$01C2_{16}$, $01C5_{16}$,$01C4_{16}$ | indeterminate |
| AD1j(j=3 to 5) | $01C7_{16}$,$01C6_{16}$, $01C9_{16}$,$01C8_{16}$, $01CB_{16}$,$01CA_{16}$ | indeterminate |
| AD1j(j=6,7) | $01CD_{16}$,$01CC_{16}$, $01CF_{16}$,$01CE_{16}$ | indeterminate |

| Function | R | W |
|---|---|---|
| Eight low-order bits of A-D conversion result | ○ | — |
| During 10-bit mode : Two high-order bits of A-D conversion result<br>During 8-bit mode : When read, their contents are indeterminate | ○ | — |
| Nothing is assigned.<br>When write, set to "0". When read, their contents are indeterminate. | — | — |

**Figure 1.24.7. A-D converter-related registers (6)**

### (1) One-shot mode

In one-shot mode, the pin selected using the analog input pin select bit is used for one-shot A-D conversion. Table 1.24.2 shows the specifications of one-shot mode.

**Table 1.24.2. 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-Di conversion start flag, external trigger |
| Stop condition | • End of A-Di conversion (A-Di 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 ANj0 to ANj7 (j =non, 0, 2, 15), ANEX0, ANEX1 |
| Reading of result of A-D converter | Read A-D register corresponding to selected pin |

### (2) Repeat mode

In repeat mode, the pin selected using the analog input pin select bit is used for repeated A-D conversion. Table 1.24.3 shows the A-D control register in repeat mode.

**Table 1.24.3. Repeat mode specifications**

| Item | Specification |
|---|---|
| Function | The pin selected by the analog input pin select bit is used for repeated A-D conversion |
| Start condition | Writing “1” to A-D conversion start flag, external trigger |
| Stop condition | Writing “0” to A-D conversion start flag |
| Interrupt request generation timing | None generated |
| Input pin | One of ANj0 to ANj7 (j =non, 0, 2, 15), ANEX0, ANEX1 |
| Reading of result of A-D converter | Read A-D register corresponding to selected pin |

### (3) Single sweep mode

In single sweep mode, the pins selected using the A-D sweep pin select bit are used for one-by-one A-D conversion. Table 1.24.4 shows the A-D control register in single sweep mode.

**Table 1.24.4. Single sweep mode specifications**

| Item | Specification |
|---|---|
| Function | The pins selected by the A-Di sweep pin select bit are used for one-by-one A-D conversion |
| Start condition | Writing “1” to A-D converter start flag, external trigger |
| Stop condition | • End of A-Di conversion (A-D conversion start flag changes to “0”, except when external trigger is selected)<br>• Writing “0” to A-Di conversion start flag |
| Interrupt request generation timing | End of sweep |
| Input pin | ANj0 and ANj1 (2 pins), ANj0 to ANj3 (4 pins), ANj0 to ANj5 (6 pins), or ANj0 to ANj7 (8 pins) (j =non, 0, 2, 15) |
| Reading of result of A-D converter | Read A-D register corresponding to selected pin |

## (4) Repeat sweep mode 0

In repeat sweep mode 0, the pins selected using the A-D sweep pin select bit are used for repeat sweep A-D conversion. Table 1.24.5 shows the specifications of repeat sweep mode 0.

**Table 1.24.5. 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 | ANj0 and ANj1 (2 pins), ANj0 to ANj3 (4 pins), ANj0 to ANj5 (6 pins), or ANj0 to AN7 (8 pins) (j =non, 0, 2, 15) |
| Reading of result of A-D converter | Read A-D register corresponding to selected pin |

## (5) Repeat sweep mode 1

In repeat sweep mode 1, all pins are used for A-D conversion with emphasis on the pin or pins selected using the A-D sweep pin select bit. Table 1.26.6 shows the specifications of repeat sweep mode 1.

**Table 1.26.6. 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 : AN0 selected<br>ANj0 → ANj1 → ANj0 → ANj2 → ANj0 → ANj3 etc. (j =non, 0, 2, 15) |
| 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 | ANj0 to ANj7 (j =non, 0, 2, 15) |
| With emphasis on the pin | ANj0 (1 pin), ANj0 and ANj1 (2 pins), ANj0 to ANj2 (3 pins), ANj0 to ANj3 (4 pins) (j =non, 0, 2, 15) |
| Reading of result of A-D converter | Read A-D register corresponding to selected pin |

## (a) Resolution select function

**8/10-bit mode select bit of A-D control register 1 (bit 3 at address 0397$_{16}$, 01D7$_{16}$)**

When set to 10-bit precision, the low 8 bits are stored in the even addresses and the high 2 bits in the odd addresses. When set to 8-bit precision, the low 8 bits are stored in the even addresses.

## (b) Sample and hold

Sample and hold are selected by setting bit 0 of the A-D control register 2 (address 0394$_{16}$, 01D4$_{16}$) to “1”. When sample and hold are selected, the rate of conversion of each pin increases. As a result, a 28 ØAD cycle is achieved with 8-bit resolution and 33 ØAD with 10-bit resolution. Sample and hold can be selected in all modes. However, in all modes, be sure to specify before starting A-D conversion whether sample and hold are to be used.

## (c) Trigger select function

Can appoint start of conversion, by a combination of setting of trigger select bit (bit 5 at address 0396$_{16}$, 01D6$_{16}$) and external trigger request cause select bit (bit 5 and bit 6 at address 0394$_{16}$, 01D4$_{16}$), as follows.

**Table 1.24.7. Trigger select function setting**

| | Trigger select bit="0" | Trigger select bit="1" | | |
|---|---|---|---|---|
| | | External trigger cause select bits | | |
| | | 00 | 01 | 10 |
| A-D0 | Software trigger | ADTRG | Timer B2 OFCOI(Note) | Group 2 channel 1 interrupt |
| A-D1 | Software trigger | ADTRG | Timer B2 OFCOI(Note) | Group 3 channel 1 interrupt |

Timer B2 OFCOI : Timer B2 occurrence frequency counter overflow interrupt
Note :Valid in three-phase PWM mode.

## (d) Two circuit same time start (software trigger)

Two A-D converters can start at the same time by setting simultaneous start bit (bit 7 of address 0394$_{16}$) to “1”.

During the A-D circuit of either of A-D0 and A-D1 are operated, do not set “1” to the simultaneous start bit. Do not set to "1" when external trigger is selected. When using this bit, do not set A-D conversion start flag (bit 6 of address 0396$_{16}$, 01D6$_{16}$) to "1".

## (e) Replace function of input pin

Setting "1" to A-D channel replace select bit of A-D0 control register 2 (ADS:bit 4 at address 0394$_{16}$) can replace channel of A-D0 and A-D1. A-D conversion reliability is confirmed by replacing channels.

When ADS bit is "1", a corresponding pin of A-D0 register i is selected by analog input port select bits of A-D1 control register 2 (bits 2 and 1 at address 01D4$_{16}$). In this case, A-D0 control register 0 and A-D1 control register 0 must be set to same value.

**Table 1.24.8. Setting of analog input port replace of A-D converter**

| Setting value / A-D conversion stored register | A-D channel replace select bit / Analog output port select bit | 1 | | |
|---|---|---|---|---|
| | | 00 | 10 | 11 |
| A-D0 register 0 | | $AN15_0$ | $AN0_0$ | $AN2_0$ |
| A-D0 register 1 | | $AN15_1$ | $AN0_1$ | $AN2_1$ |
| A-D0 register 2 | | $AN15_2$ | $AN0_2$ | $AN2_2$ |
| A-D0 register 3 | | $AN15_3$ | $AN0_3$ | $AN2_3$ |
| A-D0 register 4 | | $AN15_4$ | $AN0_4$ | $AN2_4$ |
| A-D0 register 5 | | $AN15_5$ | $AN0_5$ | $AN2_5$ |
| A-D0 register 6 | | $AN15_6$ | $AN0_6$ | $AN2_6$ |
| A-D0 register 7 | | $AN15_7$ | $AN0_7$ | $AN2_7$ |
| A-D1 register 0 | | $AN_0$ | | |
| A-D1 register 1 | | $AN_1$ | | |
| A-D1 register 2 | | $AN_2$ | | |
| A-D1 register 3 | | $AN_3$ | | |
| A-D1 register 4 | | $AN_4$ | | |
| A-D1 register 5 | | $AN_5$ | | |
| A-D1 register 6 | | $AN_6$ | | |
| A-D1 register 7 | | $AN_7$ | | |

## (f) 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 as $AN_0$ and $AN_1$ analog input signal respectively.

Set the related input peripheral function of the function select register B3 to disabled.

## (g) 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 and bit 7 of the A-D control register 1 (address $0397_{16}$) is “11”, input via $AN_0$ to $AN_7$ is output from ANEX0.

The input from ANEX1 is converted from analog to digital and the result stored in the corresponding A-D register. The speed of A-D conversion depends on the response of the external operation amp. Do not connect the ANEX0 and ANEX1 pins directly. Figure 1.24.8 is an example of how to connect the pins in external operation amp mode.

Set the related input peripheral function of the function select register B3 to disabled.

**Table 1.24.9. Setting of extended analog input pins**

| A-D0 control register 1 | | ANEX0 function | ANEX1 function |
|---|---|---|---|
| Bit 7 | Bit 6 | | |
| 0 | 0 | Not used | Not used |
| 0 | 1 | $P9_5$ analog input | Not used |
| 1 | 0 | Not used | $P9_6$ analog input |
| 1 | 1 | Output to external ope-amp | Input from external ope-amp |

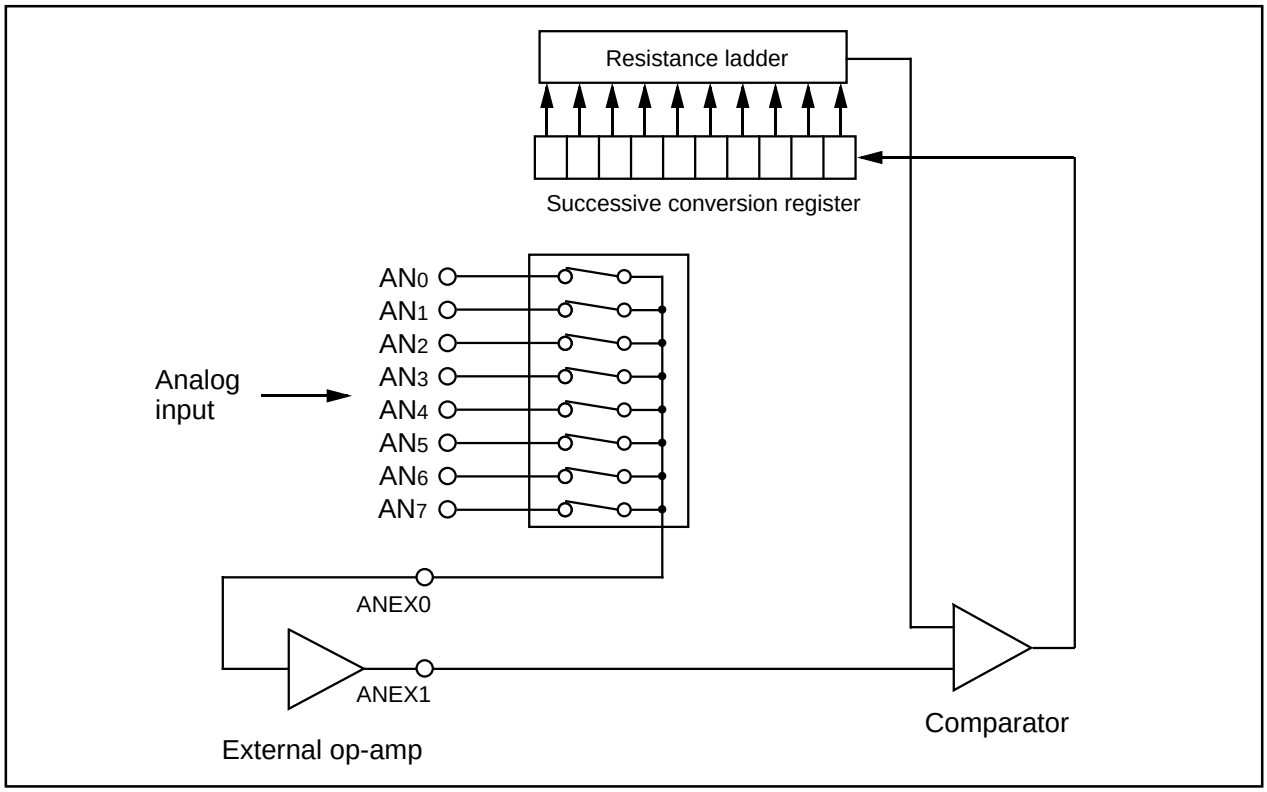



**Figure 1.24.8. Example of external op-amp connection mode**

## (h) Power consumption reduction function

**$V_{REF}$ connect bit (bit 5 at addresses $0397_{16}$, $01D7_{16}$)**

The $V_{REF}$ connect bit (bit 5 at address $0397_{16}$, $01D7_{16}$) 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 connecting $V_{REF}$.

Do not write A-D conversion start flag and $V_{REF}$ connect bit to “1” at the same time. Do not clear $V_{REF}$ connect bit to “0” during A-D conversion. This $V_{REF}$ is without reference to D-A converter's $V_{REF}$.

## Precaution

After A-D conversion is complete, if the CPU reads the A-D register at the same time as the A-D conversion result is being saved to A-D register, wrong A-D conversion value is saved into the A-D register. This happens when the internal CPU clock is selected from divided main clock or sub-clock.

- **When using the one-shot or single sweep mode**
  Confirm that A-D conversion is complete before reading the A-D register.
  (Note: When A-D conversion interrupt request bit is set, it shows that A-D conversion is completed.)

- **When using the repeat mode or repeat sweep mode 0 or 1**
  Use the undivided main clock as the internal CPU clock.

## 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. Set the function select register A3 to I/O port, the related input peripheral function of the function select register B3 to disabled and the direction register to input mode. Do not set the target port to pulled-up when D-A output is enabled.

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 (This is unrelated to bit 5 of A-D control register 1 (addresses $0397_{16}$, $01D7_{16}$)

Table 1.25.1 lists the performance of the D-A converter. Figure 1.25.1 shows the block diagram of the D-A converter. Figure 1.25.2 shows the D-A control register. Figure 1.25.3 shows the D-A converter equivalent circuit.

When the D-A converter is not used, set the D-A register to "00" and D-A output enable bit to "0".

**Table 1.25.1. Performance of D-A converter**

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

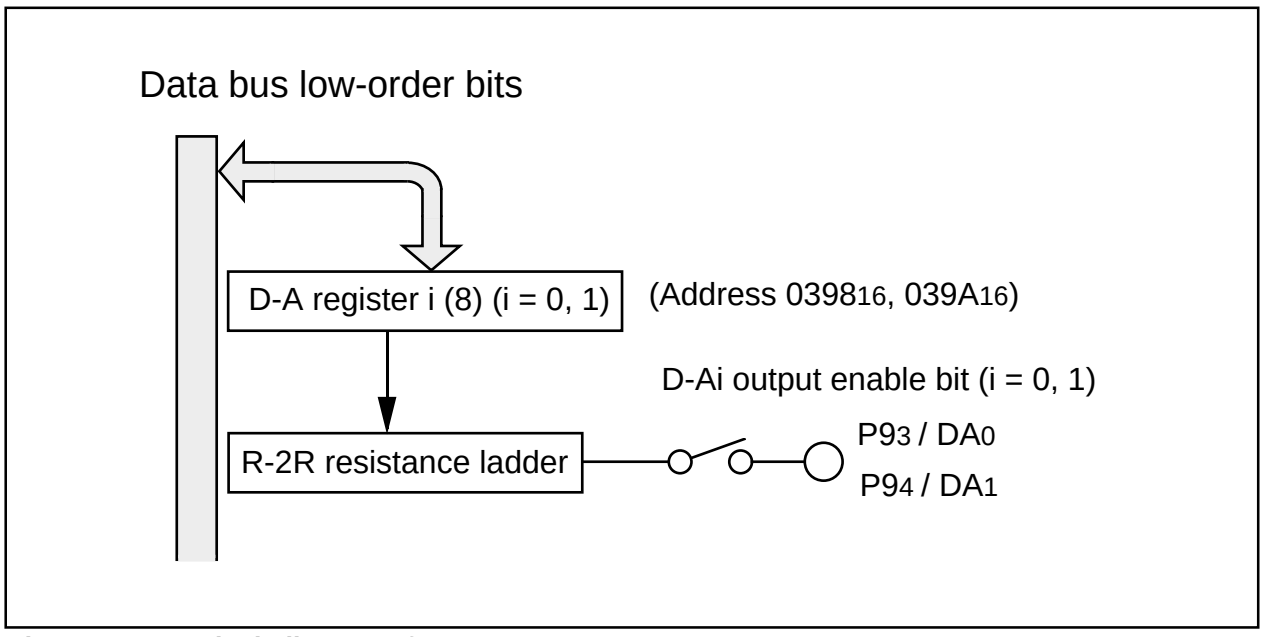



**Figure 1.25.1. Block diagram of D-A converter**

## D-A control register

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

| Symbol | Address | When reset |
|---|---|---|
| DACON | $039C_{16}$ | $XXXXXX00_2$ |

| 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>When write, set to "0". When read, their contents are "0". | | — | — |
| — | | | — | — |
| — | | | — | — |
| — | | | — | — |
| — | | | — | — |
| — | | | — | — |

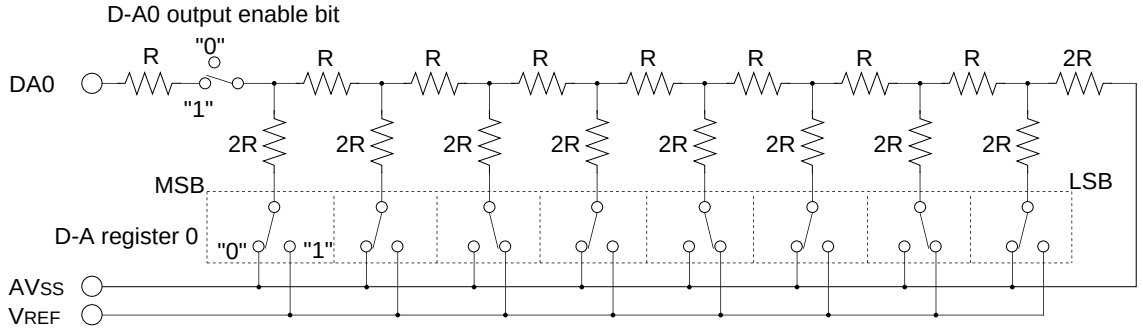
## D-A register i

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

| Symbol | Address | When reset |
|---|---|---|
| DAi(i=0,1) | $0398_{16}$, $039A_{16}$ | Indeterminate |

| Function | Setting range | R | W |
|---|---|---|---|
| Output value of D-A conversion | $00_{16}$ to $FF_{16}$ | ○ | ○ |

**Figure 1.25.2. D-A control register**




Note 1: In the above figure, the D-A register value is "$2A_{16}$".
Note 2: This circuit is the same in D-A1.
Note 3: To save power dissipation when not using the D-A converter, set the D-A output enable bit to "0" and the D-A register to "$00_{16}$", and prevent current flowing to the R-2R resistance.

**Figure 1.25.3. D-A converter equivalent circuit**

## CRC Calculation Circuit

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

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

Figure 1.26.1 shows the block diagram of the CRC circuit. Figure 1.26.2 shows the CRC-related registers. Figure 1.26.3 shows the CRC example.

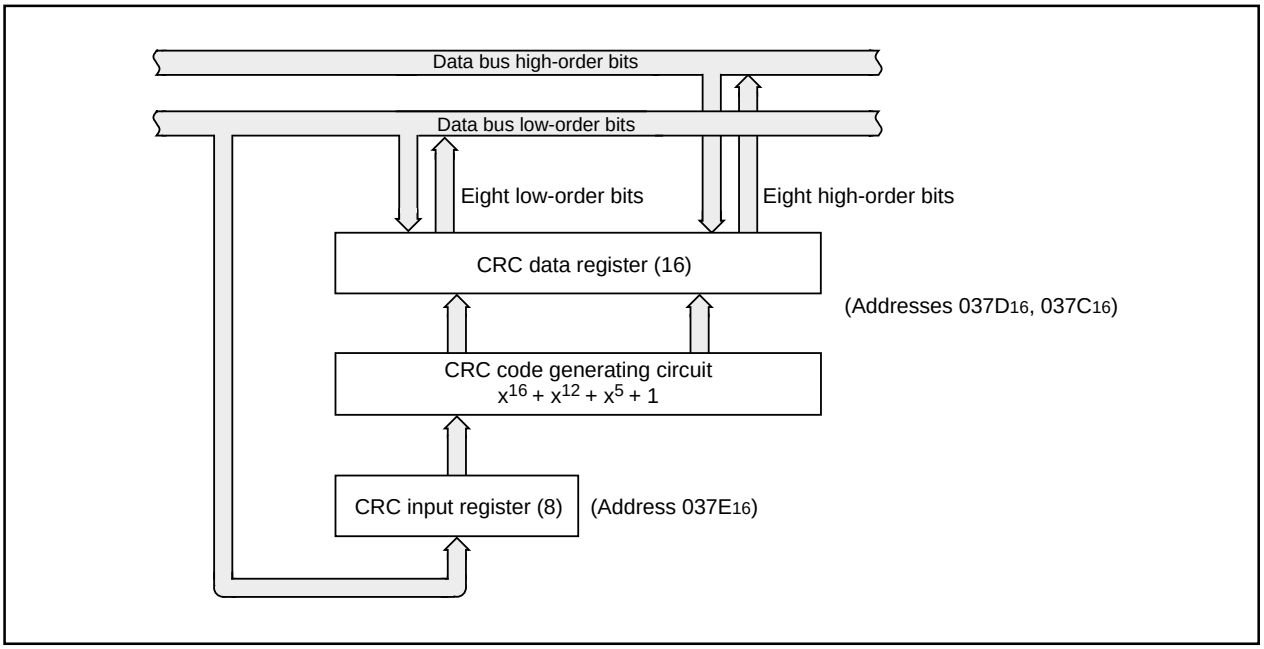



**Figure 1.26.1. Block diagram of CRC circuit**

CRC data register

b15 (b7) b8 (b0) b7 b0

| Symbol | Address | When reset |
|---|---|---|
| CRCD | 037D16, 037C16 | Indeterminate |

| Function | Setting range | R | W |
|---|---|---|---|
| CRC calculation output register | 000016 to FFFF16 | ○ | ○ |

CRC input register

b7 b0

| Symbol | Address | When reset |
|---|---|---|
| CRCIN | 037E16 | Indeterminate |

| Function | Setting range | R | W |
|---|---|---|---|
| Data input register | 0016 to FF16 | ○ | ○ |

**Figure 1.26.2. CRC-related registers**

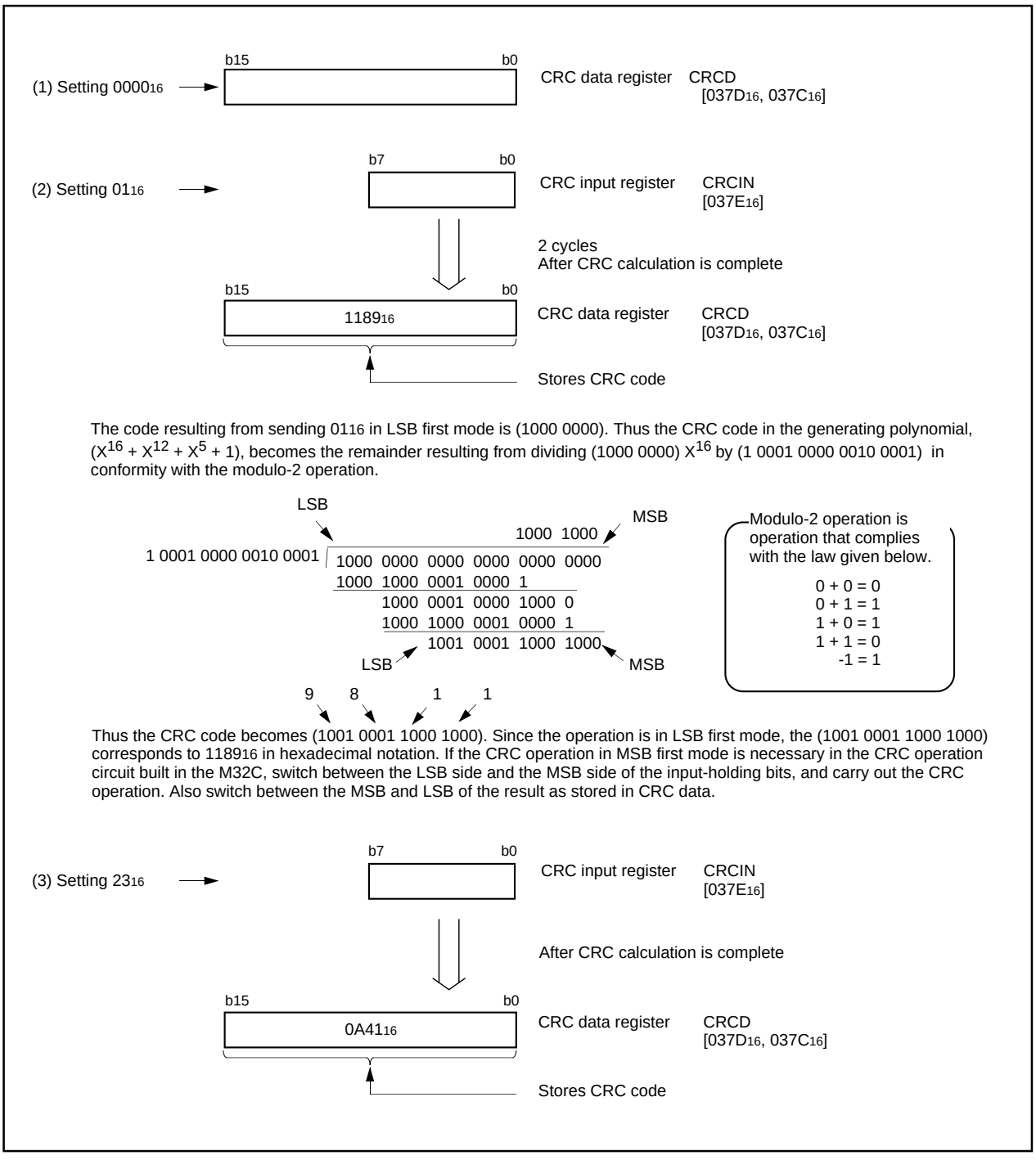



**Figure 1.26.3. CRC example**

## X-Y Converter

X-Y conversion rotates the 16 x 16 matrix data by 90 degrees. It can also be used to invert the top and bottom of the 16-bit data. Figure 1.27.1 shows the XY control register.

The Xi and the Yi registers are 16-bit registers. There are 16 of each (where i= 0 to 15).

The Xi and Yi registers are mapped to the same address. The Xi register is a write-only register, while the Yi register is a read-only register. Be sure to access the Xi and Yi registers in 16-bit units from an even address. Operation cannot be guaranteed if you attempt to access these registers in 8-bit units.

XY control register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 | Symbol | Address | When reset |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | XYC | $02E0_{16}$ | $XXXXXX00_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| XYC0 | Read-mode set bit | 0 : Data conversion<br>1 : No data conversion | ○ | ○ |
| XYC1 | Write-mode set bit | 0 : No bit mapping conversion<br>1 : Bit mapping conversion | ○ | ○ |
| — | Noting is assigned. When write, set to "0".<br>When read, their contents are indeterminate. | | — | — |
| — | | | — | — |
| — | | | — | — |
| — | | | — | — |
| — | | | — | — |
| — | | | — | — |

**Figure 1.27.1. XY control register**

The reading of the Yi register is controlled by the read-mode set bit (bit 0 at address 02E0$_{16}$).
When the read-mode set bit (bit 0 at address 02E0$_{16}$) is “0”, specific bits in the Xi register can be read at the same time as the Yi register is read.
For example, when you read the Y0 register, bit 0 is read as bit 0 of the X0 register, bit 1 is read as bit 0 of the X1 register, ..., bit 14 is read as bit 0 of the X14 register, bit 15 as bit 0 of the X15 register. Similarly, when you read the Y15 register, bit 0 is bit 15 of the X0 register, bit 1 is bit 15 of the X1 register, ..., bit 14 is bit 15 of the X14 register, bit 15 is bit 15 of the X15 register.
Figure 1.27.2 shows the conversion table when the read mode set bit = “0”. Figure 1.27.3 shows the X-Y conversion example.

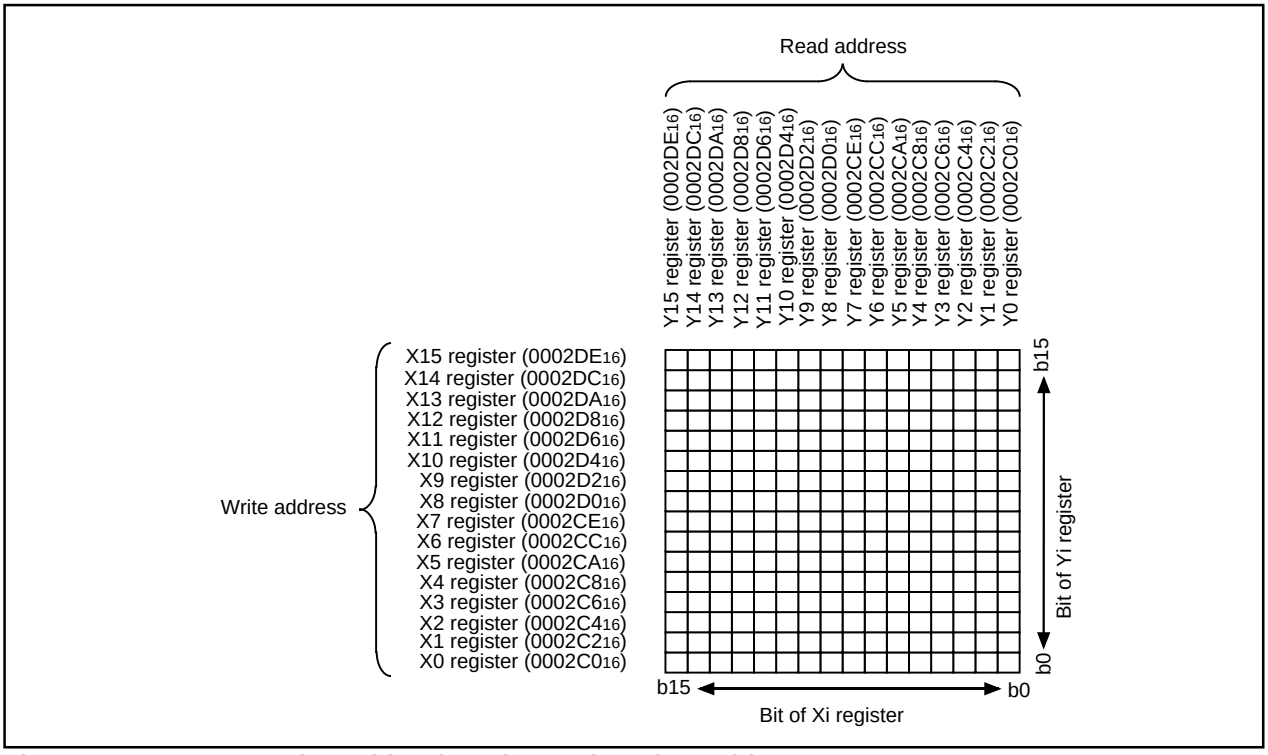



**Figure 1.27.2. Conversion table when the read mode set bit = “0”**

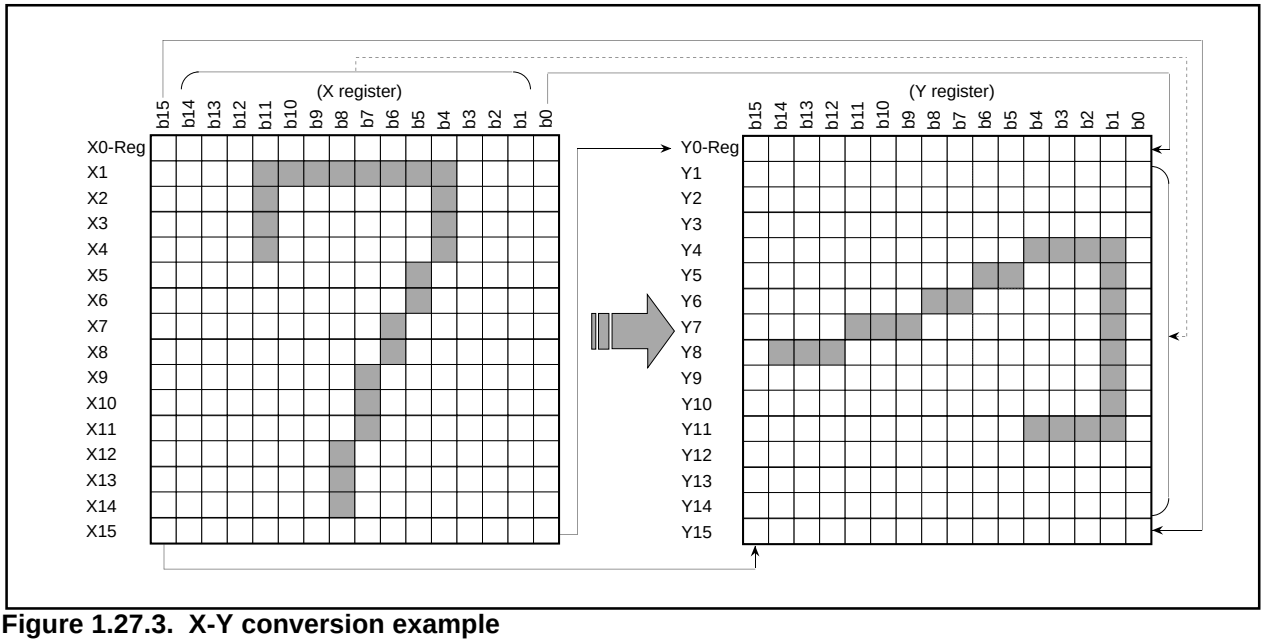



**Figure 1.27.3. X-Y conversion example**

When the read-mode set bit (bit 0 at address 02E0$_{16}$) is “1”, you can read the value written to the Xi register by reading the Yi register. Figure 1.27.4 shows the conversion table when the read mode set bit = “1”.

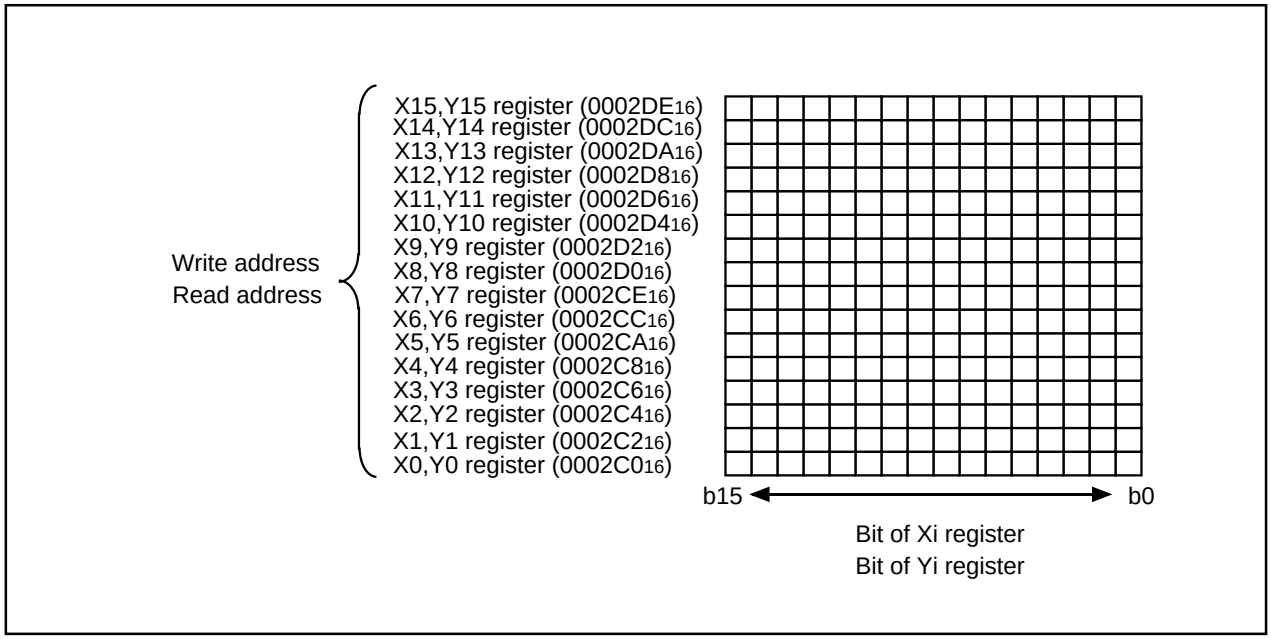



**Figure 1.27.4. Conversion table when the read mode set bit = “1”**

The value written to the Xi register is controlled by the write mode set bit (bit 1 at address 02E0$_{16}$).
When the write mode set bit (bit 1 at address 02E0$_{16}$) is “0” and data is written to the Xi register, the bit stream is written directly.
When the write mode set bit (bit 1 at address 02E0$_{16}$) is “1” and data is written to the Xi register, the bit sequence is reversed so that the high becomes low and vice versa. Figure 1.27.5 shows the conversion table when the write mode set bit = “1”.

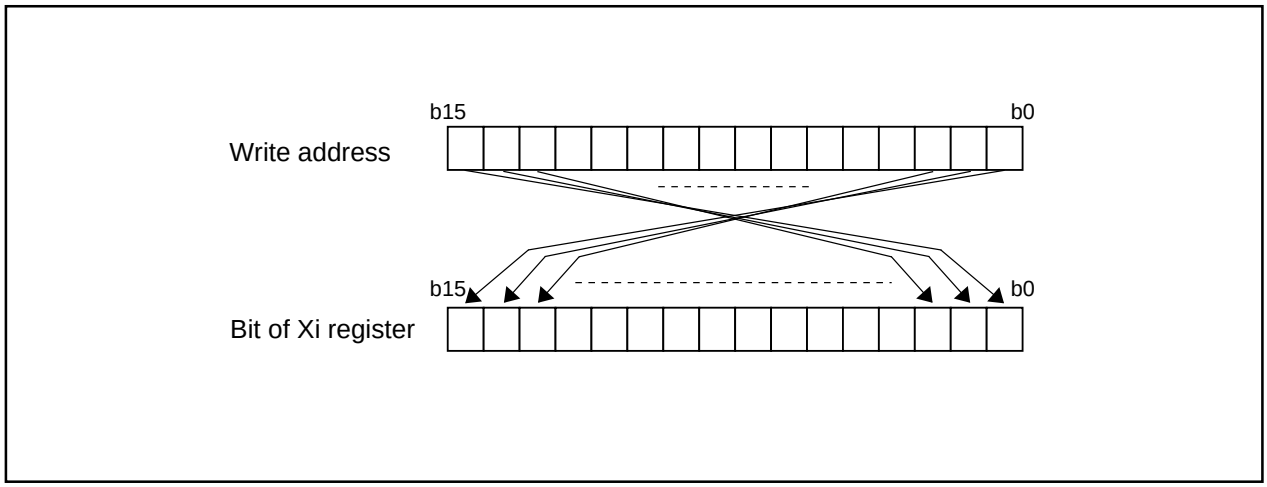



**Figure 1.27.5. Conversion table when the write mode set bit = “1”**

## DRAM Controller

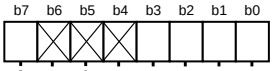
There is a built in DRAM controller to which it is possible to connect between 512 Kbytes and 8 Mbytes of DRAM. Table 1.28.1 shows the functions of the DRAM controller.

**Table 1.28.1. DRAM Controller Functions**

| | |
|---|---|
| DRAM space | 512KB, 1MB, 2MB, 4MB, 8MB |
| Bus control | 2CAS/1W |
| Refresh | $\overline{CAS}$ before $\overline{RAS}$ refresh, Self refresh-compatible |
| Function modes | EDO-compatible, fast page mode-compatible |
| Waits | 1 wait or 2 waits, programmable |

To use the DRAM controller, use the DRAM space select bit of the DRAM control register (address $0040_{16}$) to specify the DRAM size. Figure 1.28.1 shows the DRAM control register.
The DRAM controller cannot be used in external memory mode 3 (bits 1 and 2 at address $0005_{16}$ are "$11_2$"). Always use the DRAM controller in external memory modes 0, 1, or 2.
When the data bus width is 16-bit in DRAM area, set "1" to R/W mode select bit (bit 2 at address $0004_{16}$). Set wait time between after DRAM power ON and before memory processing, and processing necessary for dummy cycle to refresh DRAM by software.

DRAM Control register




Symbol: DRAMCONT — Address: $0040_{16}$ — When reset: Indeterminate (Note 1)

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| WT | Wait select bit (Note 2) | 0 : Two wait<br>1 : One wait | ○ | ○ |
| AR0 | DRAM space select bit | b3 b2 b1<br>0 0 0 : DRAM ignored<br>0 0 1 : Must not be set<br>0 1 0 : 0.5MB<br>0 1 1 : 1MB<br>1 0 0 : 2MB<br>1 0 1 : 4MB<br>1 1 0 : 8MB<br>1 1 1 : Must not be set | ○ | ○ |
| AR1 | | | ○ | ○ |
| AR2 | | | ○ | ○ |
| —— | Nothing is assigned. When write, set to "0".<br>When read, their contents are indeterminate. | | – | – |
| —— | | | – | – |
| —— | | | – | – |
| SREF | Self-refresh mode bit (Note 3) | 0 : Self-refresh OFF<br>1 : Self-refresh ON | ○ | ○ |

Note 1: After reset, the content of this register is indeterminate. DRAM controller starts operation after writing to this register.
Note 2: The number of cycles with 2 waits is 3-2-2. With 1 wait, it is 2-1-1.
Note 3: When you set to "1", both $\overline{RAS}$ and $\overline{CAS}$ change to "L". When you set to "0", $\overline{RAS}$ and $\overline{CAS}$ change to "H" and then normal operation (read/write, refresh) is resumed. In stop mode, there is no control.
Note 4: Set the bus width using the external data bus width control register (address $000B_{16}$). When selecting 8-bit bus width, CASH is indeterminate.

**Figure 1.28.1. DRAM control register**

## • DRAM Controller Multiplex Address Output

The DRAM controller outputs the row addresses and column addresses as a multiplexed signal to the address bus $A_8$ to $A_{20}$. Figure 1.28.2 shows the output format for multiplexed addresses.

### 8-bit bus mode

| Pin function | MA12 (A20) | MA11 (A19) | MA10 (A18) | MA9 (A17) | MA8 (A16) | MA7 (A15) | MA6 (A14) | MA5 (A13) | MA4 (A12) | MA3 (A11) | MA2 (A10) | MA1 (A9) | MA0 (A8) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Row address | (A20) | (A19) | A18 | A17 | A16 | A15 | A14 | A13 | A12 | A11 | A10 | A9 | – |
| Column address | (A22) | (A22) | A19 | A8 | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 | – |
| | | | ← 512KB, 1MB → | | | | | | | | | | |
| Row address | (A20) | A19 | A18 | A17 | A16 | A15 | A14 | A13 | A12 | A11 | A10 | A9 | – |
| Column address | (A22) | A21 | A20 | A8 | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 | – |
| | | | ← 2MB, 4MB → | | | | | | | | | | |
| Row address | A20 | A19 | A18 | A17 | A16 | A15 | A14 | A13 | A12 | A11 | A10 | A9 | – |
| Column address | (A22) | A22 | A21 | A8 | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 | – |
| | | | ← 8MB → | | | | | | | | | | |

### 16-bit bus mode

| Pin function | MA12 (A20) | MA11 (A19) | MA10 (A18) | MA9 (A17) | MA8 (A16) | MA7 (A15) | MA6 (A14) | MA5 (A13) | MA4 (A12) | MA3 (A11) | MA2 (A10) | MA1 (A9) | MA0 (A8) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Row address | (A20) | (A19) | A18 | A17 | A16 | A15 | A14 | A13 | A12 | A11 | A10 | (A9) | – |
| Column address | (A22) | (A20) | $\overline{A9}$ | A8 | A7 | A6 | A5 | A4 | A3 | A2 | A1 | (A0) | – |
| | | | ← 512KB → | | | | | | | | | | |
| Row address | (A20) | A19 | A18 | A17 | A16 | A15 | A14 | A13 | A12 | A11 | A10 | (A9) | – |
| Column address | (A22) | A20 | $\overline{A9}$ | A8 | A7 | A6 | A5 | A4 | A3 | A2 | A1 | (A0) | – |
| | | | ← 1MB, 2MB → | | | | | | | | | | |
| Row address | A20 | A19 | A18 | A17 | A16 | A15 | A14 | A13 | A12 | A11 | A10 | (A9) | – |
| Column address | A22 | A21 | $\overline{A9}$ | A8 | A7 | A6 | A5 | A4 | A3 | A2 | A1 | (A0) | – |
| | | | ← 4MB, 8MB (Note 2) → | | | | | | | | | | |

Note 1: ( ) invalid bit: bits that change according to selected mode (8-bit/16-bit bus mode, DRAM space).
Note 2: The figure is for 4Mx1 or 4Mx4 memory configuration. If you are using a 4Mx16 configuration, use combinations of the following: For row addresses, MA0 to MA12; for column addresses MA2 to MA8, MA11, and MA12. Or for row addresses MA1 to MA12; for column addresses MA2 to MA9, MA11, MA12.
Note 3: "–" is indetermimate.

**Figure 1.28.2. Output format for multiplexed addresses**

## • Refresh

The refresh method is $\overline{CAS}$ before $\overline{RAS}$. The refresh interval is set by the DRAM refresh interval set register (address $0041_{16}$). The refresh signal is not output in HOLD state. Figure 1.28.3 shows the DRAM refresh interval set register.

Use the following formula to determine the value to set in the refresh interval set register.

Refresh interval set register value (0 to 255) = refresh interval time / (BCLK frequency X 32) - 1

DRAM refresh interval set register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 | Symbol | Address | When reset |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | REFCNT | $0041_{16}$ | Indeterminate |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| REFCNT0 | Refresh interval set bit | b7 b6 b5 b4 b3 b2 b1 b0<br>0 0 0 0 0 0 0 0 : 1.1 µs (Note)<br>0 0 0 0 0 0 0 1 : 2.1 µs<br>0 0 0 0 0 0 1 0 : 3.2 µs<br>⋮<br>1 1 1 1 1 1 1 1 : 272.8 µs | ○ | ○ |
| REFCNT1 | | | ○ | ○ |
| REFCNT2 | | | ○ | ○ |
| REFCNT3 | | | ○ | ○ |
| REFCNT4 | | | ○ | ○ |
| REFCNT5 | | | ○ | ○ |
| REFCNT6 | | | ○ | ○ |
| REFCNT7 | | | ○ | ○ |

Note. Refresh interval at 30 MHz operating (no division).
Refresh interval = BCLK frequency X (refresh interval set bit + 1) X 32

**Figure 1.28.3. DRAM refresh interval set register**

The DRAM self-refresh operates in STOP mode, etc.

When shifting to self-refresh, select DRAM ignored by the DRAM space select bit. In the next instruction, simultaneously set the DRAM space select bit and self-refresh ON by self-refresh mode bit. Also, insert two NOPs after the instruction that sets the self-refresh mode bit to "1".

Do not access external memory while operating in self-refresh. (All external memory space access is inhibited. )

When disabling self-refresh, simultaneously select DRAM ignored by the DRAM space select bit and self-refresh OFF by self-refresh mode bit. In the next instruction, set the DRAM space select bit.

Do not access the DRAM space immediately after setting the DRAM space select bit.

Example) One wait is selected by the wait select bit and 4MB is selected by the DRAM space select bit

Shifting to self-refresh

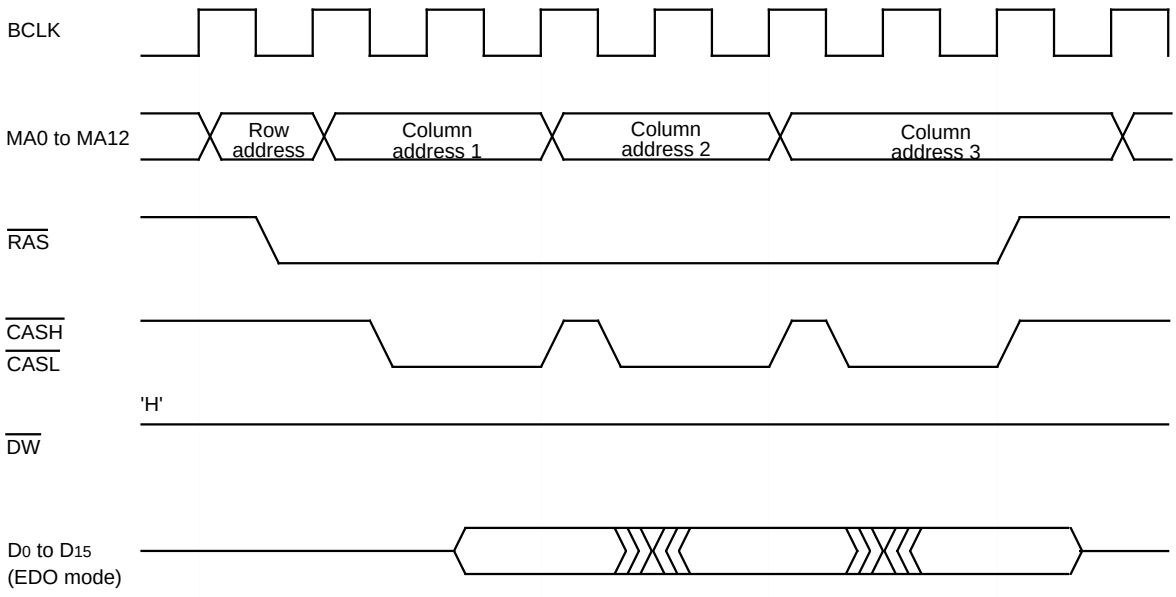
```
•••
mov.b   #00000001b,DRAMCONT     ;DRAM ignored, one wait is selected
mov.b   #10001011b,DRAMCONT     ;Set self-refresh, select 4MB and one wait
nop                             ;Two nops are needed
nop                             ;
•••
```

Disable self-refresh

```
•••
mov.b   #00000001b,DRAMCONT     ;Disable self-refresh, DRAM ignored, one wait is
                                ;selected
mov.b   #00001011b,DRAMCONT     ;Select 4MB and one wait
nop                             ;Inhibit instruction to access DRAM area
nop
•••
```

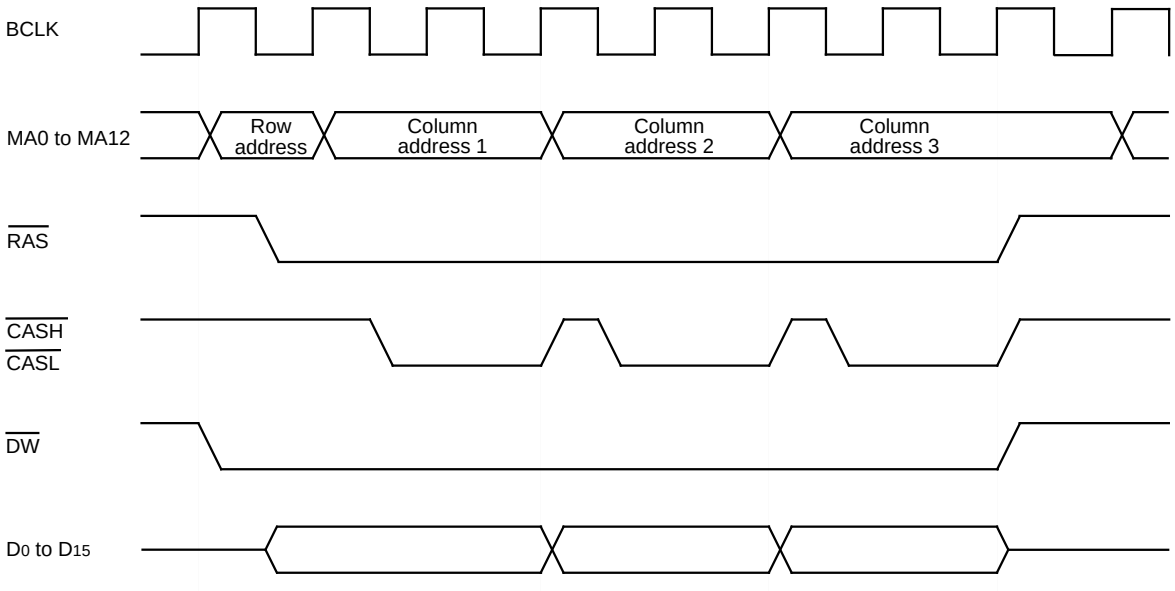
Figures 1.28.4 to 1.28.6 show the bus timing during DRAM access.

**< Read cycle (wait control bit = 0) >**




Note : Only $\overline{CASL}$ is operating in 8-bit data bus width.

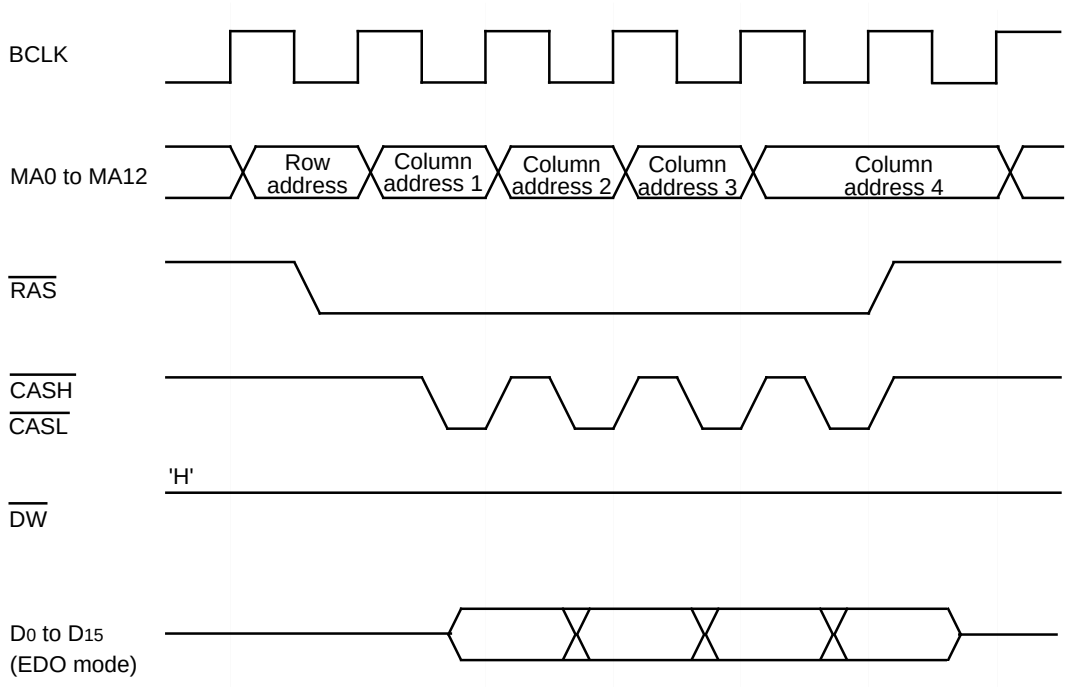
**< Write cycle (wait control bit = 0) >**




Note : Only $\overline{CASL}$ is operating in 8-bit data bus width.

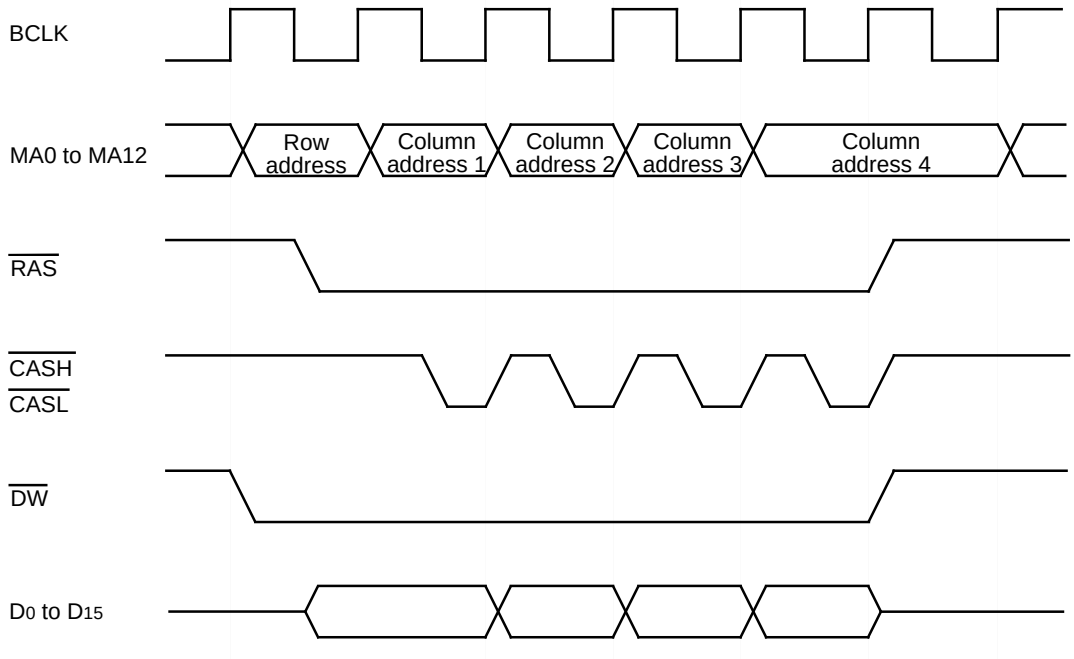
**Figure 1.28.4. The bus timing during DRAM access (1)**

**< Read cycle (wait control bit = 1) >**



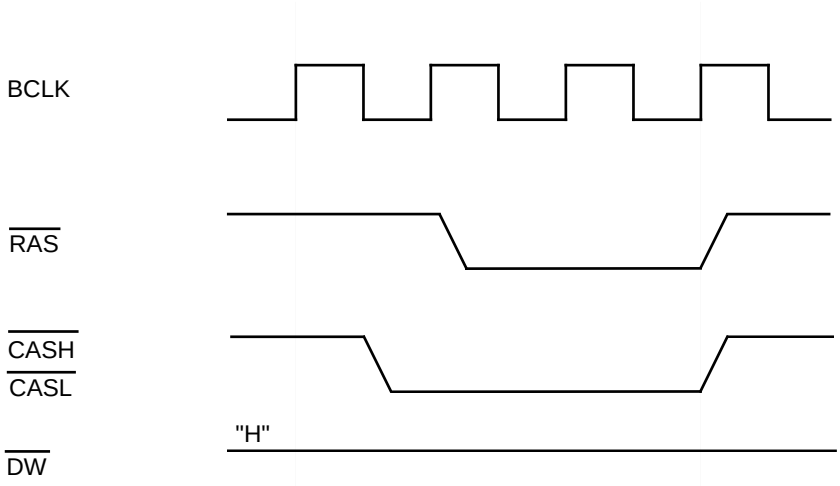

Note : Only $\overline{CASL}$ is operating in 8-bit data bus width.

**< Write cycle (wait control bit = 1) >**



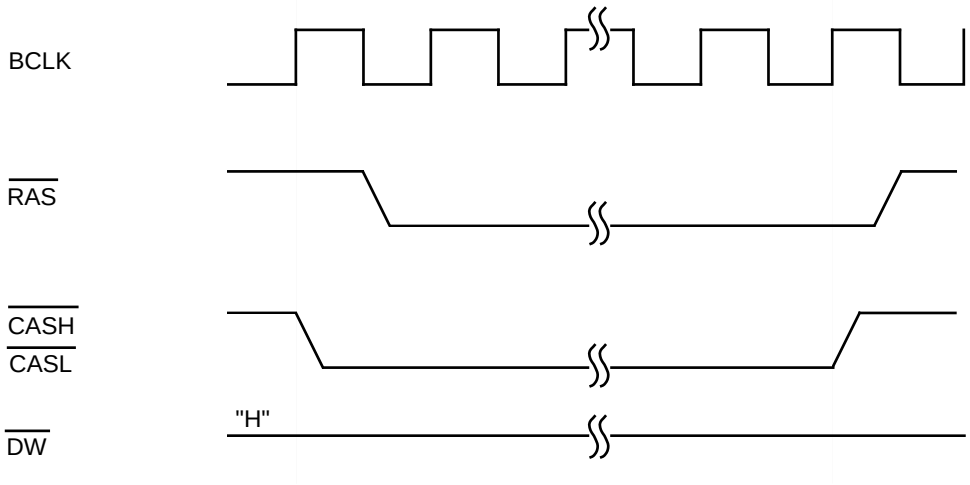

Note : Only $\overline{CASL}$ is operating in 8-bit data bus width.

**Figure 1.28.5. The bus timing during DRAM access (2)**




< $\overline{CAS}$ before $\overline{RAS}$ refresh cycle >

Note : Only $\overline{CASL}$ is operating in 8-bit data bus width.




< Self refresh cycle >

Note : Only $\overline{CASL}$ is operating in 8-bit data bus width.

**Figure 1.28.6. The bus timing during DRAM access (3)**

## Programmable I/O Ports

There are 123 programmable I/O ports in 144-pin version: P0 to P15 (excluding $P8_5$). There are 87 programmable I/O ports in 100-pin version: P0 to P10 (excluding $P8_5$). Each port can be set independently for input or output using the direction register. A pull-up resistance for each block of 4 ports can be set. $P8_5$ is an input-only port and has no built-in pull-up resistance.

Figures 1.29.1 to 1.29.4 show the programmable I/O ports.

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

To use the pins as the inputs for the built-in peripheral devices, set the direction register of each pin to input mode. When the pins are used as the outputs for the built-in peripheral devices (other than the D-A converter), set the corresponding function select registers A, B and C. When pins are to be used as the outputs for the D-A converter, set the function select register A3 of each pin to I/O port, and set the direction registers to input mode.

See the descriptions of the respective functions for how to set up the built-in peripheral devices.

### (1) Direction registers

Figurs 1.29.5 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.

In memory expansion and microprocessor mode, the contents of corresponding direction register of pins $A_0$ to $A_{22}$, $\overline{A_{23}}$, $D_0$ to $D_{15}$, $MA_0$ to $MA_{12}$, $\overline{CS0}$ to $\overline{CS3}$, $\overline{WRL}/\overline{WR}/\overline{CASL}$, $\overline{WRH}/\overline{BHE}/\overline{CASH}$, $\overline{RD}/\overline{DW}$, BCLK/ALE/CLKOUT, $\overline{HLDA}$/ALE, $\overline{HOLD}$, ALE/$\overline{RAS}$, and $\overline{RDY}$ are not changed.

Note: There is no direction register bit for $P8_5$.

### (2) Port registers

Figure 1.29.6 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 a port register corresponds one for one to each I/O pin.

In memory expansion and microprocessor mode, the contents of corresponding port register of pins $A_0$ to $A_{22}$, $\overline{A_{23}}$, $D_0$ to $D_{15}$, $MA_0$ to $MA_{12}$, $\overline{CS0}$ to $\overline{CS3}$, $\overline{WRL}/\overline{WR}/\overline{CASL}$, $\overline{WRH}/\overline{BHE}/\overline{CASH}$, $\overline{RD}/\overline{DW}$, BCLK/ALE/CLKOUT, $\overline{HLDA}$/ALE, $\overline{HOLD}$, ALE/$\overline{RAS}$, and $\overline{RDY}$ are not changed.

### (3) Function select register A

Figures 1.29.7 to 1.29.11 show the function select registers A.

The register is used to select port output and peripheral function output when the port functions for both port output and peripheral function output.

Each bit of this register corresponds to each pin that functions for both port output and peripheral function output.

### (4) Function select register B

Figures 1.29.12 and 1.29.13 show the function select registers B.

This register selects the first peripheral function output and second peripheral function output when multiple peripheral function outputs are assigned to a pin. For pins with a third peripheral function, this register selects whether to enable the function select register C, or output the second peripheral function.

Each bit of this register corresponds to each pin that has multiple peripheral function outputs assigned to it.

This register is enabled when the bits of the corresponding function select register A are set for peripheral functions.

The bit 3 to bit 6 of function select register B3 is ignored bit for input peripheral function. When using DA0/DA1 and ANEX0/ANEX1, set related bit to“1”. When not using DA0/DA1 or ANEX0/ANEX1, set related bit to “0”.

### (5) Function select register C

Figure 1.29.14 shows the function select register C.

This register is used to select the first peripheral function output and the third peripheral function output when three peripheral function outputs are assigned to a pin.

This register is effective when the bits of the function select register A of the counterpart pin have selected a peripheral function and when the function select register B has made effective the function select register C.

The bit 7 (PSC_7) is assigned the key-in interrupt inhibit bit. Setting “1” in the key-in interrupt inhibit bit causes no key-in interrupts regardless of the settings in the interrupt control register even if “L” is entered in pins $\overline{KI_0}$ to $\overline{KI_3}$. With “1” set in the key-in interrupt inhibit bit, input from a port pin cannot be effected even if the port direction register is set to input mode.

### (6) Pull-up control registers

Figures 1.29.15 to 1.29.17 show the pull-up control registers.

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

Since P0 to P5 operate as the bus in memory expansion mode and microprocessor mode, do not set the pull-up control register. However, it is possible to select pull-up resistance presence to the usable port as I/O port by setting.

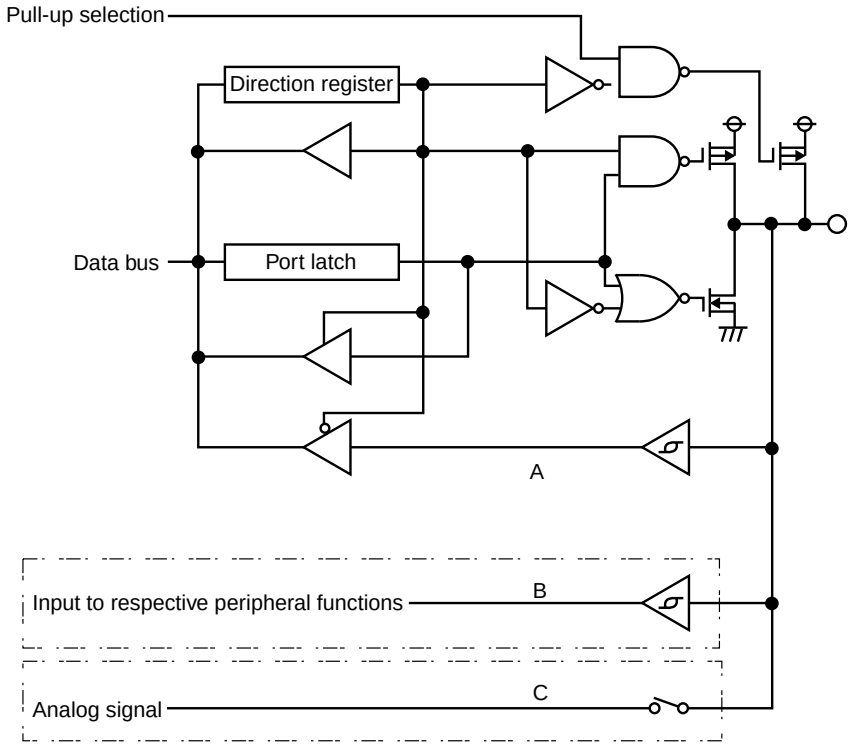
### (7) Port control register

Figure 1.29.18 shows the port control register.

This register is used to choose whether to make port P1 a CMOS port or an Nch open drain. In the Nch open drain, the CMOS port’s Pch is kept always turned off so that the port P1 cannot be a complete open drain. Thus the absolute maximum rating of the input voltage falls within the range from “- 0.3 V to Vcc + 0.3 V”.

The port control register functions similarly to the above. Also in the case in which port P1 can be used as a port when the bus width in the full external areas comprises 8 bits in either microprocessor mode or in memory expansion mode.

## Programmable I/O ports



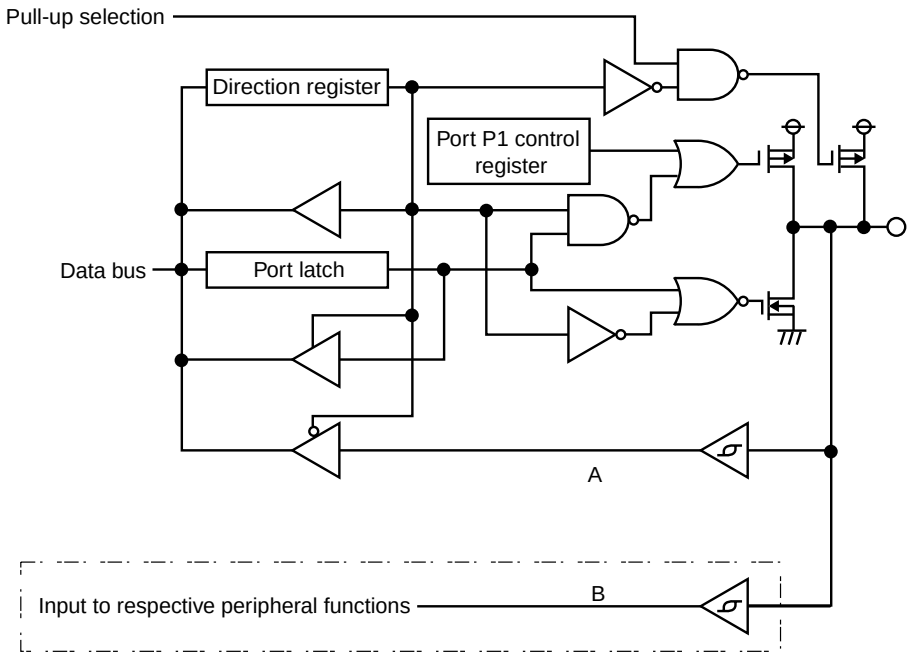

| Port \ Option | (A) Hysteresis presence | Circuit (B) Input to respective peripheral functions | Circuit (C) Analog I/F |
|---|---|---|---|
| $P0_0$ to $P0_7$<br>$P2_0$ to $P2_7$ | – | – | ○ |
| $P3_0$ to $P3_7$<br>$P4_0$ to $P4_7$<br>$P5_0$ to $P5_2$<br>$P5_4$ | – | – | – |
| $P5_5$ | – | ○ | – |
| $P5_6$ | – | – | – |
| $P5_7$ | – | ○ | – |
| $P8_3$, $P8_4$ | ○ | ○ | – |
| $P8_6$ | – | – | – |
| $P8_7$ | – | ○ | ○ |
| $P10_0$ to $P10_3$ | – | – | ○ |
| $P10_4$ to $P10_7$ | ○ | ○ | ○ |
| (Note) $P11_4$<br>$P14_4$ to $P14_6$ | – | – | – |
| (Note) $P15_2$, $P15_3$<br>$P15_6$, $P15_7$ | – | ○ | ○ |

○ : Present, – : Not present

Note: These ports exist in 144-pin version.

**Figure 1.29.1. Programmable I/O ports (1)**

Programmable I/O ports with port control register

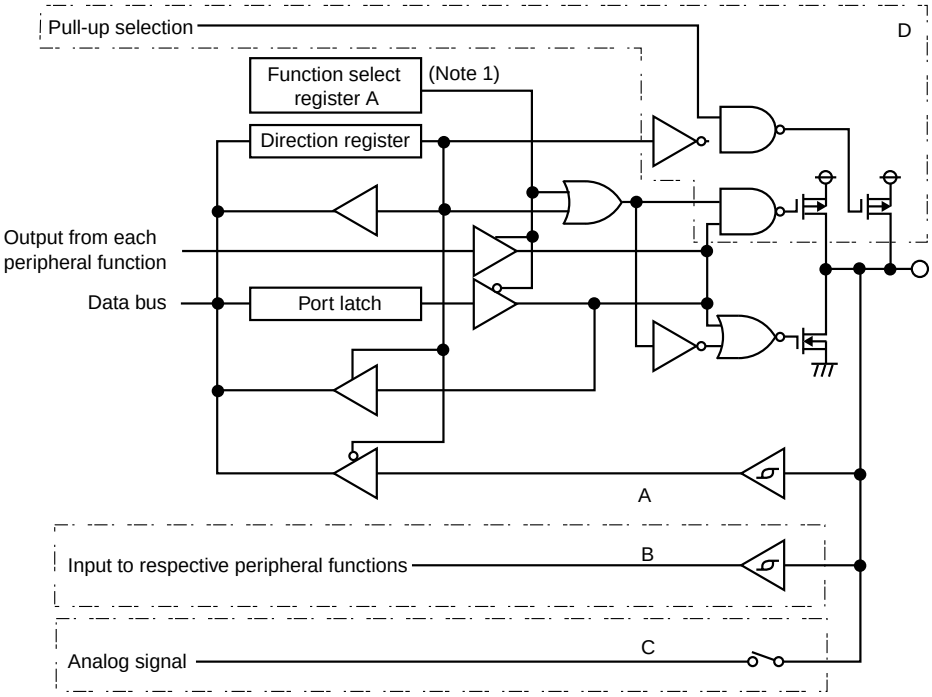



| Port \ Option | (A) Hysteresis presence | Circuit (B) Input to respective peripheral functions |
|---|---|---|
| P1$_0$ to P1$_4$ | — | — |
| P1$_5$ to P1$_7$ | ○ | ○ |

○ : Present, — : Not present

**Figure 1.29.2. Programmable I/O ports (2)**

## Programmable I/O ports with function select register




| Port \ Option | (A) Hysteresis presence | Circuit (B) Input to respective peripheral functions | Circuit (C) Analog I/F | Circuit (D) |
|---|---|---|---|---|
| $P5_3$ (Note 1) | – | – | – | ○ |
| $P6_0$, $P6_1$<br>$P6_3$ to $P6_5$, $P6_7$ | – | ○ | – | ○ |
| $P7_0$, $P7_1$ (Note 2) | – | ○ | – | – |
| $P7_2$ to $P7_7$<br>$P8_0$, $P8_1$ | – | ○ | – | ○ |
| $P8_2$ | ○ | ○ | – | ○ |
| $P9_0$ to $P9_2$ | – | ○ | – | ○ |
| $P9_3$ to $P9_6$ | – | ○ | ○ | ○ |
| $P9_7$ | – | ○ | – | ○ |
| (Note 3) $P11_0$ | – | – | – | ○ |
| (Note 3) $P11_1$, $P11_2$ | – | ○ | – | ○ |
| (Note 3) $P11_3$<br>$P12_0$ | – | – | – | ○ |
| (Note 3) $P12_1$, $P12_2$ | – | – | – | ○ |
| (Note 3) $P12_3$ to $P12_7$<br>$P13_0$ to $P13_4$ | – | – | – | ○ |
| (Note 3) $P13_5$, $P13_6$ | – | ○ | – | ○ |
| (Note 3) $P13_7$<br>$P14_0$, $P14_1$ | – | – | – | ○ |
| (Note 3) $P14_2$, $P14_3$ | – | ○ | – | ○ |
| (Note 3) $P15_0$, $P15_1$<br>$P15_4$, $P15_5$ | – | ○ | ○ | ○ |

○ : Present, – : Not present

Note 1: $P5_3$ is clock output select bit for BCLK.
Note 2: $P7_0$ and $P7_1$ are N-channel open drain output.
Note 3: These ports exist in 144-pin version.

**Figure 1.29.3. Programmable I/O ports (3)**

Input-only port

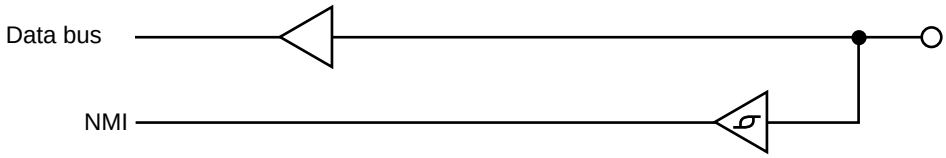
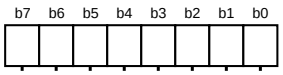



**Figure 1.29.4. Programmable I/O ports (4)**

Port Pi direction register (Note 1, 2, 3)




| Symbol | Address | When reset |
|---|---|---|
| PDi(i=0 to 5) | 03E2$_{16}$, 03E3$_{16}$, 03E6$_{16}$, 03E7$_{16}$, 03EA$_{16}$, 03EB$_{16}$ | 00$_{16}$ |
| PDi(i=6 to 11) | 03C2$_{16}$, 03C3$_{16}$, 03C6$_{16}$, 03C7$_{16}$, 03CA$_{16}$, 03CB$_{16}$ | 00$_{16}$ |
| PDi(i=12 to 15) | 03CE$_{16}$, 03CF$_{16}$, 03D2$_{16}$, 03D3$_{16}$ | 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) | ○ | ○ |
| PDi_1 | Port Pi$_1$ direction register | 0 : Input mode (Functions as an input port)<br>1 : Output mode (Functions as an output port) | ○ | ○ |
| PDi_2 | Port Pi$_2$ direction register | 0 : Input mode (Functions as an input port)<br>1 : Output mode (Functions as an output port) | ○ | ○ |
| PDi_3 | Port Pi$_3$ direction register | 0 : Input mode (Functions as an input port)<br>1 : Output mode (Functions as an output port) | ○ | ○ |
| PDi_4 | Port Pi$_4$ direction register | 0 : Input mode (Functions as an input port)<br>1 : Output mode (Functions as an output port) | ○ | ○ |
| PDi_5 | Port Pi$_5$ direction register | 0 : Input mode (Functions as an input port)<br>1 : Output mode (Functions as an output port) (Note 4) | ○ | ○ |
| PDi_6 | Port Pi$_6$ direction register | 0 : Input mode (Functions as an input port)<br>1 : Output mode (Functions as an output port) (Note 4) | ○ | ○ |
| PDi_7 | Port Pi$_7$ direction register | 0 : Input mode (Functions as an input port)<br>1 : Output mode (Functions as an output port) (Note 4) | ○ | ○ |

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

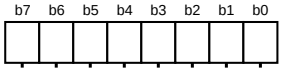
Note 2: In memory expansion and microprocessor mode, the contents of corresponding port direction register of pins A$_0$ to A$_{22}$, $\overline{A_{23}}$, D$_0$ to D$_{15}$, MA0 to MA12, $\overline{CS0}$ to $\overline{CS3}$, $\overline{WRL}$/$\overline{WR}$/$\overline{CASL}$, $\overline{WRH}$/$\overline{BHE}$/$\overline{CASH}$, $\overline{RD}$/$\overline{DW}$, BCLK/ALE/CLKOUT, $\overline{HLDA}$/ALE, $\overline{HOLD}$, ALE/$\overline{RAS}$, and $\overline{RDY}$ are not changed.

Note 3: Port 11 to 15 registers exist in 144-pin version.

Note 4: Nothing is assigned in bit5 of Port P8 direction register, bit7 to bit5 of port P11 direction register and bit7 of port P14 direction register.
When write, set to "0". When read, its content is indeterminate.

**Figure 1.29.5. Direction register**

## Port Pi register (Note 1, 2, 3)



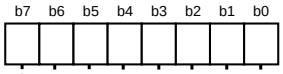

| Symbol | Address | When reset |
|---|---|---|
| Pi(i=0 to 5) | 03E0$_{16}$, 03E1$_{16}$, 03E4$_{16}$, 03E5$_{16}$, 03E8$_{16}$, 03E9$_{16}$ | Indeterminate |
| Pi(i=6 to 11) | 03C0$_{16}$, 03C1$_{16}$, 03C4$_{16}$, 03C5$_{16}$, 03C8$_{16}$, 03C9$_{16}$ | Indeterminate |
| Pi(i=12 to 15) | 03CC$_{16}$, 03CD$_{16}$, 03D0$_{16}$, 03D1$_{16}$ | Indeterminate |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Pi_0 | Port Pi$_0$ register | 0 : "L" level<br>1 : "H" level (Note 4) | ○ | ○ |
| Pi_1 | Port Pi$_1$ register | 0 : "L" level<br>1 : "H" level (Note 4) | ○ | ○ |
| Pi_2 | Port Pi$_2$ register | 0 : "L" level<br>1 : "H" level | ○ | ○ |
| Pi_3 | Port Pi$_3$ register | 0 : "L" level<br>1 : "H" level | ○ | ○ |
| Pi_4 | Port Pi$_4$ register | 0 : "L" level<br>1 : "H" level | ○ | ○ |
| Pi_5 | Port Pi$_5$ register | 0 : "L" level<br>1 : "H" level (Note 5) (Note 6) | ○ | ○ |
| Pi_6 | Port Pi$_6$ register | 0 : "L" level<br>1 : "H" level (Note 6) | ○ | ○ |
| Pi_7 | Port Pi$_7$ register | 0 : "L" level<br>1 : "H" level (Note 6) | ○ | ○ |

Note 1: Data is input and output to and from each pin by reading and writing to and from each corresponding bit.
Note 2: In memory expansion and microprocessor mode, the contents of corresponding port direction register of pins A$_0$ to A$_{22}$, $\overline{A_{23}}$, D$_0$ to D$_{15}$, MA0 to MA12, $\overline{CS0}$ to $\overline{CS3}$, $\overline{WRL}$/$\overline{WR}$/$\overline{CASL}$, $\overline{WRH}$/$\overline{BHE}$/$\overline{CASH}$, $\overline{RD}$/$\overline{DW}$, BCLK/ALE/CLKOUT, $\overline{HLDA}$/ALE, $\overline{HOLD}$, ALE/$\overline{RAS}$, and $\overline{RDY}$ are not changed.
Note 3: Port 11 to 15 direction registers exist in 144-pin version.
Note 4: Port P7$_0$ and P7$_1$ output high impedance because of N-channel open drain output.
Note 5: Port P8$_5$ is read only (There is not W).
Note 6: Nothing is assigned in bit7 to bit5 of port P11 and bit7 of port P14.
When write, set to "0". When read, its content is indeterminate.

**Figure 1.29.6. Port register**

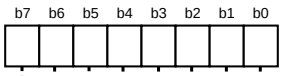
## Function select register A0




| Symbol | Address | When reset |
|---|---|---|
| PS0 | $03B0_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PS0_0 | Port $P6_0$ function select bit | 0 : I/O port<br>1 : UART0 output ($\overline{RTS_0}$) | ○ | ○ |
| PS0_1 | Port $P6_1$ function select bit | 0 : I/O port<br>1 : UART0 output ($CLK_0$ output) | ○ | ○ |
| PS0_2 | Port $P6_2$ function select bit | 0 : I/O port<br>1 : Function that was selected in bit2 of function select register B0 | ○ | ○ |
| PS0_3 | Port $P6_3$ function select bit | 0 : I/O port<br>1 : UART0 output ($TxD_0$/$SDA_0$) | ○ | ○ |
| PS0_4 | Port $P6_4$ function select bit | 0 : I/O port<br>1 : Function that was selected in bit4 of function select register B0 | ○ | ○ |
| PS0_5 | Port $P6_5$ function select bit | 0 : I/O port<br>1 : UART1 output ($CLK_1$ output) | ○ | ○ |
| PS0_6 | Port $P6_6$ function select bit | 0 : I/O port<br>1 : Function that was selected in bit6 of function select register B0 | ○ | ○ |
| PS0_7 | Port $P6_7$ function select bit | 0 : I/O port<br>1 : UART1 output ($TxD_1$/$SDA_1$) | ○ | ○ |

## Function select register A1




| Symbol | Address | When reset |
|---|---|---|
| PS1 | $03B1_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PS1_0 | Port $P7_0$ function select bit | 0 : I/O port<br>1 : Function that was selected in bit0 of function select register B1 | ○ | ○ |
| PS1_1 | Port $P7_1$ function select bit | 0 : I/O port<br>1 : Function that was selected in bit1 of function select register B1 | ○ | ○ |
| PS1_2 | Port $P7_2$ function select bit | 0 : I/O port<br>1 : Function that was selected in bit2 of function select register B1 | ○ | ○ |
| PS1_3 | Port $P7_3$ function select bit | 0 : I/O port<br>1 : Function that was selected in bit3 of function select register B1 | ○ | ○ |
| PS1_4 | Port $P7_4$ function select bit | 0 : I/O port<br>1 : Function that was selected in bit4 of function select register B1 | ○ | ○ |
| PS1_5 | Port $P7_5$ function select bit | 0 : I/O port<br>1 : Function that was selected in bit5 of function select register B1 | ○ | ○ |
| PS1_6 | Port $P7_6$ function select bit | 0 : I/O port<br>1 : Function that was selected in bit6 of function select register B1 | ○ | ○ |
| PS1_7 | Port $P7_7$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 0 output ($OUTC0_1$/$ISCLK_0$) | ○ | ○ |

**Figure 1.29.7. Function select register A (1)**

## Function select register A2

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

| Symbol | Address | When reset |
|---|---|---|
| PS2 | $03B4_{16}$ | $00X00000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PS2_0 | Port $P8_0$ function select bit | 0 : I/O port<br>1 : Function that was selected in bit0 of function select register B2 | ○ | ○ |
| PS2_1 | Port $P8_1$ function select bit | 0 : I/O port<br>1 : Function that was selected in bit1 of function select register B2 | ○ | ○ |
| PS2_2 | Port $P8_2$ function select bit | 0 : I/O port<br>1 : Function that was selected in bit2 of function select register B2 | ○ | ○ |
| —— | Reserve bit | Must always be "0". | ○ | ○ |
| —— | | | ○ | ○ |
| —— | Noting is assigned. When write, set to "0".<br>When read, their contents are indeterminate. | | – | – |
| —— | Reserve bit | Must always be "0". | ○ | ○ |
| —— | | | ○ | ○ |

## Function select register A3 (Note)

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

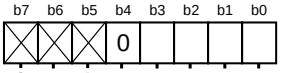
| Symbol | Address | When reset |
|---|---|---|
| PS3 | $03B5_{16}$ | $00_{16}$ |

| Bit symbol | Bit Name | Function | R | W |
|---|---|---|---|---|
| PS3_0 | Port $P9_0$ function select bit | 0 : I/O port<br>1 : UART3 output ($CLK_3$) | ○ | ○ |
| PS3_1 | Port $P9_1$ function select bit | 0 : I/O port<br>1 : Function that was selected in bit1 of function select register B3 | ○ | ○ |
| PS3_2 | Port $P9_2$ function select bit | 0 : I/O port<br>1 : Function that was selected in bit2 of function select register B3 | ○ | ○ |
| PS3_3 | Port $P9_3$ function select bit | 0 : I/O port<br>1 : UART3 output ($\overline{RTS_3}$) | ○ | ○ |
| PS3_4 | Port $P9_4$ function select bit | 0 : I/O port<br>1 : UART4 output ($\overline{RTS_4}$) | ○ | ○ |
| PS3_5 | Port $P9_5$ function select bit | 0 : I/O port<br>1 : UART4 output ($CLK_4$) | ○ | ○ |
| PS3_6 | Port $P9_6$ function select bit | 0 : I/O port<br>1 : UART4 output ($TxD_4$/$SDA_4$) | ○ | ○ |
| PS3_7 | Port $P9_7$ function select bit | 0 : I/O port<br>1 : Function that was selected in bit7 of function select register B3 | ○ | ○ |

Note :Set bit 2 of protect register (address $000A_{16}$) to "1" before rewriting to this register.

**Figure 1.29.8. Function select register A (2)**

## Function select register A5 (Note)

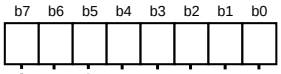



| Symbol | Address | When reset |
|---|---|---|
| PS5 | $03B9_{16}$ | $XXX0\ 0000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PS5_0 | Port $P11_0$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 1 output ($OUTC1_0$/ $ISTxD1/BE1_{OUT}$) | ○ | ○ |
| PS5_1 | Port $P11_1$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 1 output ($OUTC1_1$/ $ISCLK_1$) | ○ | ○ |
| PS5_2 | Port $P11_2$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 1 output ($OUTC1_2$) | ○ | ○ |
| PS5_3 | Port $P11_3$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 1 output ($OUTC1_3$) | ○ | ○ |
| —— | Reserve bit | Must always be "0". | ○ | ○ |
| —— | Noting is assigned. When write, set to "0". When read, their contents are indeterminate. | | – | – |
| —— | | | – | – |
| —— | | | – | – |

Note: This register exists in 144-pin version.

## Function select register A6 (Note)



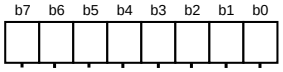

| Symbol | Address | When reset |
|---|---|---|
| PS6 | $03BC_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PS6_0 | Port $P12_0$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 3 output ($OUTC3_0$) | ○ | ○ |
| PS6_1 | Port $P12_1$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 3 output ($OUTC3_1$) | ○ | ○ |
| PS6_2 | Port $P12_2$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 3 output ($OUTC3_2$) | ○ | ○ |
| PS6_3 | Port $P12_3$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 3 output ($OUTC3_3$) | ○ | ○ |
| PS6_4 | Port $P12_4$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 3 output ($OUTC3_4$) | ○ | ○ |
| PS6_5 | Port $P12_5$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 3 output ($OUTC3_5$) | ○ | ○ |
| PS6_6 | Port $P12_6$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 3 output ($OUTC3_6$) | ○ | ○ |
| PS6_7 | Port $P12_7$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 3 output ($OUTC3_7$) | ○ | ○ |

Note: This register exists in 144-pin version.

**Figure 1.29.9. Function select register A (3)**

## Function select register A7 (Note)

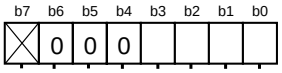



| Symbol | Address | When reset |
|---|---|---|
| PS7 | $03BD_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PS7_0 | Port $P13_0$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 2 output ($OUTC2_4$) | ○ | ○ |
| PS7_1 | Port $P13_1$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 2 output ($OUTC2_5$) | ○ | ○ |
| PS7_2 | Port $P13_2$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 2 output ($OUTC2_6$) | ○ | ○ |
| PS7_3 | Port $P13_3$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 2 output ($OUTC2_3$) | ○ | ○ |
| PS7_4 | Port $P13_4$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 2 output ($OUTC2_0$/$IST_{XD2}$/$IE_{OUT}$) | ○ | ○ |
| PS7_5 | Port $P13_5$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 2 output ($OUTC2_2$) | ○ | ○ |
| PS7_6 | Port $P13_6$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 2 output ($OUTC2_1$/$ISCLK_2$) | ○ | ○ |
| PS7_7 | Port $P13_7$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 2 output ($OUTC2_7$) | ○ | ○ |

Note: This register exists in 144-pin version.

## Function select register A8 (Note)




| Symbol | Address | When reset |
|---|---|---|
| PS8 | $03A0_{16}$ | $X0000000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PS8_0 | Port $P14_0$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 1 output ($OUTC1_4$) | ○ | ○ |
| PS8_1 | Port $P14_1$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 1 output ($OUTC1_5$) | ○ | ○ |
| PS8_2 | Port $P14_2$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 1 output ($OUTC1_6$) | ○ | ○ |
| PS8_3 | Port $P14_3$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 1 output ($OUTC1_7$) | ○ | ○ |
| ——— | Reserve bit | Must always be "0". | ○ | ○ |
| ——— | | | ○ | ○ |
| ——— | | | ○ | ○ |
| ——— | Noting is assigned. When write, set to "0".<br>When read, their contents are indeterminate. | | – | – |

Note: This register exists in 144-pin version.

**Figure 1.29.10. Function select register A (4)**

## Function select register A9 (Note)

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

| Symbol | Address | When reset |
|---|---|---|
| PS9 | $03A1_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PS9_0 | Port $P15_0$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 0 output ($OUTC0_0$/ $IST_XD_0$/$BE0_{OUT}$) | ○ | ○ |
| PS9_1 | Port $P15_1$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 0 output ($OUTC0_1$/ $ISCLK_0$) | ○ | ○ |
| —— | Reserve bit | Must always be "0". | ○ | ○ |
| —— | | | ○ | ○ |
| PS9_4 | Port $P15_4$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 0 output ($OUTC0_4$) | ○ | ○ |
| PS9_5 | Port $P15_5$ function select bit | 0 : I/O port<br>1 : Intelligent I/O group 0 output ($OUTC0_5$) | ○ | ○ |
| —— | Reserve bit | Must always be "0". | ○ | ○ |
| —— | | | ○ | ○ |

Note: This register exists in 144-pin version.

**Figure 1.29.11. Function select register A (5)**

## Function select register B0

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

| Symbol | Address | When reset |
|---|---|---|
| PSL0 | $03B2_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| —— | Reserve bit | Must always be "0". | ○ | ○ |
| —— | | | ○ | ○ |
| PSL0_2 | Port $P6_2$ peripheral function select bit | 0 : UART0 output (SCL0)<br>1 : UART0 output (STxD0) | ○ | ○ |
| —— | Reserve bit | Must always be "0". | ○ | ○ |
| PSL0_4 | Port $P6_4$ peripheral function select bit | 0 : UART1 output ($\overline{RTS1}$)<br>1 : Intelligent I/O group 2 output (OUTC21/ISCLK2) | ○ | ○ |
| —— | Reserve bit | Must always be "0". | ○ | ○ |
| PSL0_6 | Port $P6_6$ peripheral function select bit | 0 : UART1 output (SCL1)<br>1 : UART1 output (STxD1) | ○ | ○ |
| —— | Reserve bit | Must always be "0". | ○ | ○ |

## Function select register B1

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

| Symbol | Address | When reset |
|---|---|---|
| PSL1 | $03B3_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PSL1_0 | Port $P7_0$ peripheral function select bit | 0 : Function that was selected in bit0 of function select register C<br>1 : Timer output (TA0OUT) | ○ | ○ |
| PSL1_1 | Port $P7_1$ peripheral function select bit | 0 : Function that was selected in bit1 of function select register C<br>1 : UART2 output (STxD2) | ○ | ○ |
| PSL1_2 | Port $P7_2$ peripheral function select bit | 0 : Function that was selected in bit2 of function select register C<br>1 : Timer output (TA1OUT) | ○ | ○ |
| PSL1_3 | Port $P7_3$ peripheral function select bit | 0 : Function that was selected in bit3 of function select register C<br>1 : Three-phase PWM output ($\overline{V}$) | ○ | ○ |
| PSL1_4 | Port $P7_4$ peripheral function select bit | 0 : Function that was selected in bit4 of function select register C<br>1 : Three-phase PWM output (W) | ○ | ○ |
| PSL1_5 | Port $P7_5$ peripheral function select bit | 0 : Three-phase PWM output ($\overline{W}$)<br>1 : Intelligent I/O group 1 output (OUTC12) | ○ | ○ |
| PSL1_6 | Port $P7_6$ peripheral function select bit | 0 : Function that was selected in bit6 of function select register C<br>1 : Timer output (TA3OUT) | ○ | ○ |
| —— | Reserve bit | Must always be "0". | ○ | ○ |

**Figure 1.29.12. Function select register B (1)**

## Function select register B2

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

| Symbol | Address | When reset |
|---|---|---|
| PSL2 | $03B6_{16}$ | $00X00000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PSL2_0 | Port $P8_0$ peripheral function select bit | 0 : Timer output ($TA4_{OUT}$)<br>1 : Three-phase PWM output (U) | ○ | ○ |
| PSL2_1 | Port $P8_1$ peripheral function select bit | 0 : Three-phase PWM output ($\overline{U}$)<br>1 : Intelligent I/O group 3 output ($OUTC3_0$) | ○ | ○ |
| PSL2_2 | Port $P8_2$ peripheral function select bit | 0 : Intelligent I/O group 3 output ($OUTC3_2$)<br>1 : CAN output ($CAN_{OUT}$) | ○ | ○ |
| —— | Reserve bit | Must always be "0". | ○ | ○ |
| —— | | | ○ | ○ |
| —— | Noting is assigned. When write, set to "0". When read, their contents are indeterminate. | | – | – |
| —— | Reserve bit | Must always be "0". | ○ | ○ |
| —— | | | ○ | ○ |

## Function select register B3

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

| Symbol | Address | When reset |
|---|---|---|
| PSL3 | $03B7_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| —— | Reserve bit | Must always be "0". | ○ | ○ |
| PSL3_1 | Port $P9_1$ peripheral function select bit | 0 : UART3 output ($SCL_3$)<br>1 : UART3 output ($STxD_3$) | ○ | ○ |
| PSL3_2 | Port $P9_2$ peripheral function select bit | 0 : UART3 output ($TxD_3/SDA_3$)<br>1 : Intelligent I/O group 2 output ($OUTC2_0/IE_{OUT}$) | ○ | ○ |
| PSL3_3 | Port $P9_3$ peripheral function select bit | 0 : Input peripheral function enabled (Expect DA0 output) (Note)<br>1 : Input peripheral function disabled (DA0 output) | ○ | ○ |
| PSL3_4 | Port $P9_4$ peripheral function select bit | 0 : Input peripheral function enabled (Expect DA1 output) (Note)<br>1 : Input peripheral function disabled (DA1 output) | ○ | ○ |
| PSL3_5 | Port $P9_5$ peripheral function select bit | 0 : Input peripheral function enabled (Expect ANEX0 output) (Note)<br>1 : Input peripheral function disabled (ANEX0 output) | ○ | ○ |
| PSL3_6 | Port $P9_6$ peripheral function select bit | 0 : Input peripheral function enabled (Expect ANEX1 output) (Note)<br>1 : Input peripheral function disabled (ANEX1 output) | ○ | ○ |
| PSL3_7 | Port $P9_7$ peripheral function select bit | 0 : UART4 output ($SCL_4$)<br>1 : UART4 output ($STxD_4$) | ○ | ○ |

Note: Although DA0, DA1, ANEX0 and ANEX1 can be used when "0" is set in these bits, the power supply may be increased.

**Figure 1.29.13. Function select register B (2)**

## Function select register C

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

| Symbol | Address | When reset |
|---|---|---|
| PSC | $03AF_{16}$ | $00X00000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PSC_0 | Port $P7_0$ peripheral function select bit | 0 : UART2 output ($TxD_2$/$SDA_2$)<br>1 : Intelligent I/O group 2 output ($OUTC2_0$/ $ISTxD_2$/$IEOUT$) | ○ | ○ |
| PSC_1 | Port $P7_1$ peripheral function select bit | 0 : UART2 output ($SCL_2$)<br>1 : Intelligent I/O group 2 output ($OUTC2_2$) | ○ | ○ |
| PSC_2 | Port $P7_2$ peripheral function select bit | 0 : UART2 output ($CLK_2$)<br>1 : Three-phase PWM output (V) | ○ | ○ |
| PSC_3 | Port $P7_3$ peripheral function select bit | 0 : UART2 output ($\overline{RTS_2}$)<br>1 : Intelligent I/O group 1 output ($OUTC1_0$/ $ISTxD_1$/$BE1OUT$) | ○ | ○ |
| PSC_4 | Port $P7_4$ peripheral function select bit | 0 : Timer output ($TA2OUT$)<br>1 : Intelligent I/O group 1 output ($OUTC1_1$/ $ISCLK_1$) | ○ | ○ |
| ——— | Noting is assigned. When write, set to "0".<br>When read, its content is indeterminate. | | — | — |
| PSC_6 | Port $P7_6$ peripheral function select bit | 0 : Intelligent I/O group 0 output ($OUTC0_0$/$ISTxD_0$/$BE0OUT$)<br>1 : CAN output ($CANOUT$) | ○ | ○ |
| PSC_7 | Key input interrupt disable bit | 0 : Enabled (Note)<br>1 : Disabled | — | ○ |

Note: Although DA0, DA1, ANEX0 and ANEX1 can be used when "0" is set in this bit, the power supply may be increased.

**Figure 1.29.14. Function select register C**

## Pull-up control register 0 (Note)

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

| Symbol | Address | When reset |
|---|---|---|
| PUR0 | $03F0_{16}$ | $00000000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PU00 | $P0_0$ to $P0_3$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU01 | $P0_4$ to $P0_7$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU02 | $P1_0$ to $P1_3$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU03 | $P1_4$ to $P1_7$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU04 | $P2_0$ to $P2_3$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU05 | $P2_4$ to $P2_7$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU06 | $P3_0$ to $P3_3$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU07 | $P3_4$ to $P3_7$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |

Note: Since P0 to P5 operate as the bus in memory expansion mode and microprocessor mode, do not set the pull-up control register. However, it is possible to select pull-up resistance presence to the usable port as I/O port by setting.

## Pull-up control register 1 (Note)

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

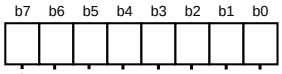
| Symbol | Address | When reset |
|---|---|---|
| PUR1 | $03F1_{16}$ | $XXXX0000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PU10 | $P4_0$ to $P4_3$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU11 | $P4_4$ to $P4_7$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU12 | $P5_0$ to $P5_3$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU13 | $P5_4$ to $P5_7$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| —— | Noting is assigned. When write, set to "0".<br>When read, their contents are indeterminate. | | – | – |
| —— | | | – | – |
| —— | | | – | – |
| —— | | | – | – |

Note: Since P0 to P5 operate as the bus in memory expansion mode and microprocessor mode, do not set the pull-up control register. However, it is possible to select pull-up resistance presence to the usable port as I/O port by setting.

**Figure 1.29.15. Pull-up control register (1)**

## Pull-up control register 2 (Note 1)

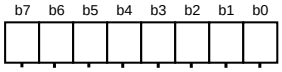



| Symbol | Address | When reset |
|---|---|---|
| PUR2 | $03DA_{16}$ | $00000000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PU20 | $P6_0$ to $P6_3$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU21 | $P6_4$ to $P6_7$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU22 | $P7_0$ to $P7_3$ pull-up | 0 : Not pulled high<br>1 : Pulled high (Note 2) | ○ | ○ |
| PU23 | $P7_4$ to $P7_7$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU24 | $P8_0$ to $P8_3$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU25 | $P8_4$ to $P8_7$ pull-up | 0 : Not pulled high<br>1 : Pulled high (Note 3) | ○ | ○ |
| PU26 | $P9_0$ to $P9_3$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU27 | $P9_4$ to $P9_7$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |

Note 1: Since $P7_0$ and $P7_1$ are N-channel open drain ports, pull-up is not available for them.
Note 2: Except port $P8_5$.

## Pull-up control register 3 <144-pin version>




| Symbol | Address | When reset |
|---|---|---|
| PUR3 | $03DB_{16}$ | $00000000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PU30 | $P10_0$ to $P10_3$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU31 | $P10_4$ to $P10_7$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU32 | $P11_0$ to $P11_3$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU33 | $P11_4$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU34 | $P12_0$ to $P12_3$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU35 | $P12_4$ to $P12_7$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU36 | $P13_0$ to $P13_3$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU37 | $P13_4$ to $P13_7$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |

**Figure 1.29.16. Pull-up control register (2)**

## Pull-up control register 3 <100-pin version>

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

| Symbol | Address | When reset |
|---|---|---|
| PUR3 | $03DB_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PU30 | $P10_0$ to $P10_3$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU31 | $P10_4$ to $P10_7$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| —— | Reserve bit | Must always be "0". | ○ | ○ |
| —— | | | ○ | ○ |
| —— | | | ○ | ○ |
| —— | | | ○ | ○ |
| —— | | | ○ | ○ |
| —— | | | ○ | ○ |

## Pull-up control register 4 (Note)

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

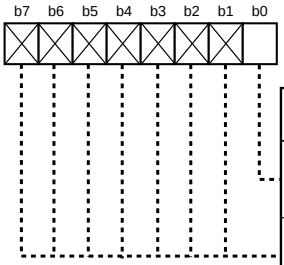
| Symbol | Address | When reset |
|---|---|---|
| PUR4 | $03DC_{16}$ | $XXXX0000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PU40 | $P14_0$ to $P14_3$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU41 | $P14_4$ to $P14_6$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU42 | $P15_0$ to $P15_3$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU43 | $P15_4$ to $P15_7$ pull-up | 0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| —— | Noting is assigned. When write, set to "0".<br>When read, their contents are indeterminate. | | – | – |
| —— | | | – | – |
| —— | | | – | – |
| —— | | | – | – |

Note: This register exists in 144-pin version..

**Figure 1.29.17. Pull-up control register (3)**

## Port control register (Note 1)



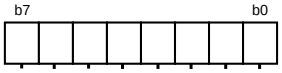

| Symbol | Address | When reset |
|---|---|---|
| PCR | $03FF_{16}$ | $XXXXXXX0_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PCR0 | Port P1 control register | 0 : Function as common CMOS port<br>1 : Function as N-ch open drain port (Note 2) | ○ | ○ |
| —— | Noting is assigned. When write, set to "0".<br>When read, their contents are indeterminate. | | — | — |

Note 1: Since P1 operates as the data bus in memory expansion mode and microprocessor mode, do not set the port control register. However, it is possible to select CMOS port or N-channel open drain pin to the usable port as I/O port by setting.

Note 2: This function is designed to permanently turn OFF the Pch of the CMOS port.
It dose not make port P1 a full open drain.
Therefore, the absolute maximum input voltage rating is [-3 to $V_{CC}$ + 3.0V].

## Input function select register




| Symbol | Address | When reset |
|---|---|---|
| IPS | $0178_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| IPS0 | Group 0 input pin select bit 0 | Assigns functions of $INPC0_0$, $INPC0_1$ /ISCLK0 and $INPC0_2$/ISRxD0/$BE0_{IN}$ to the following ports.<br>0 : $P7_6$, $P7_7$, $P8_0$<br>1 : $P15_0$, $P15_1$, $P15_2$ | ○ | ○ |
| IPS1 | Group 1 input pin select bit 1 | Assigns functions of $INPC1_1$/ISCLK1 and $INPC1_2$/ISRxD1/$BE1_{IN}$ to the following ports.<br>0 : $P7_4$, $P7_5$<br>1 : $P11_1$, $P11_2$ | ○ | ○ |
| IPS2 | P15 input peripheral function select bit | 0 : Input peripheral function is enabled<br>1 : Input peripheral function is disabled (Note) | ○ | ○ |
| IPS3 | $CAN_{IN}$ function pin select bit | 0 : $P7_7$<br>1 : $P8_3$ | ○ | ○ |
| IPS4 | ISRxD2/$IE_{IN}$ function pin select bit | b5 b4<br>0 0 : $P7_1$<br>0 1 : $P9_1$<br>1 0 : $P13_5$<br>1 1 : Must not be set | ○ | ○ |
| IPS5 | | | ○ | ○ |
| IPS6 | ISCLK2 function pin select bit | 0 : $P6_4$<br>1 : $P13_6$ | ○ | ○ |
| —— | Reserve bit | Must always be "0". | ○ | ○ |

Note: Although AD input pin can be used when "0" is set in this bit, the power supply may be increased.

**Figure 1.29.18. Port control register and input function select register**

**Table 1.29.1. Example connection of unused pins in single-chip mode**

| Pin name | Connection |
|---|---|
| Ports P0 to P15 (excluding P85) (Note 1) | After setting for input mode, connect every pin to VSS via a resistance (pull-down); or after setting for output mode, leave these pins open. |
| XOUT (Note 2) | Open |
| $\overline{\text{NMI}}$ | Connect via resistance to VCC (pull-up) |
| AVCC | Connect to VCC |
| AVSS, VREF, BYTE | Connect to VSS |

Note 1: Ports P11 to P15 exist in 144-pin version.
Note 2: With external clock input to XIN pin.

**Table 1.29.2. Example connection of unused pins in memory expansion mode and microprocessor mode**

| Pin name | Connection |
|---|---|
| Ports P6 to P15 (excluding P85) (Note 1) | After setting for input mode, connect every pin to VSS via a resistance (pull-down); or after setting for output mode, leave these pins open. |
| $\overline{\text{BHE}}$, ALE, $\overline{\text{HLDA}}$, XOUT(Note 2), BCLK | Open |
| $\overline{\text{HOLD}}$, $\overline{\text{RDY}}$, $\overline{\text{NMI}}$ | Connect via resistance to VCC (pull-up) |
| AVCC | Connect to VCC |
| AVSS, VREF | Connect to VSS |

Note 1: Ports P11 to P15 exist in 144-pin version.
Note 2: With external clock input to XIN pin.

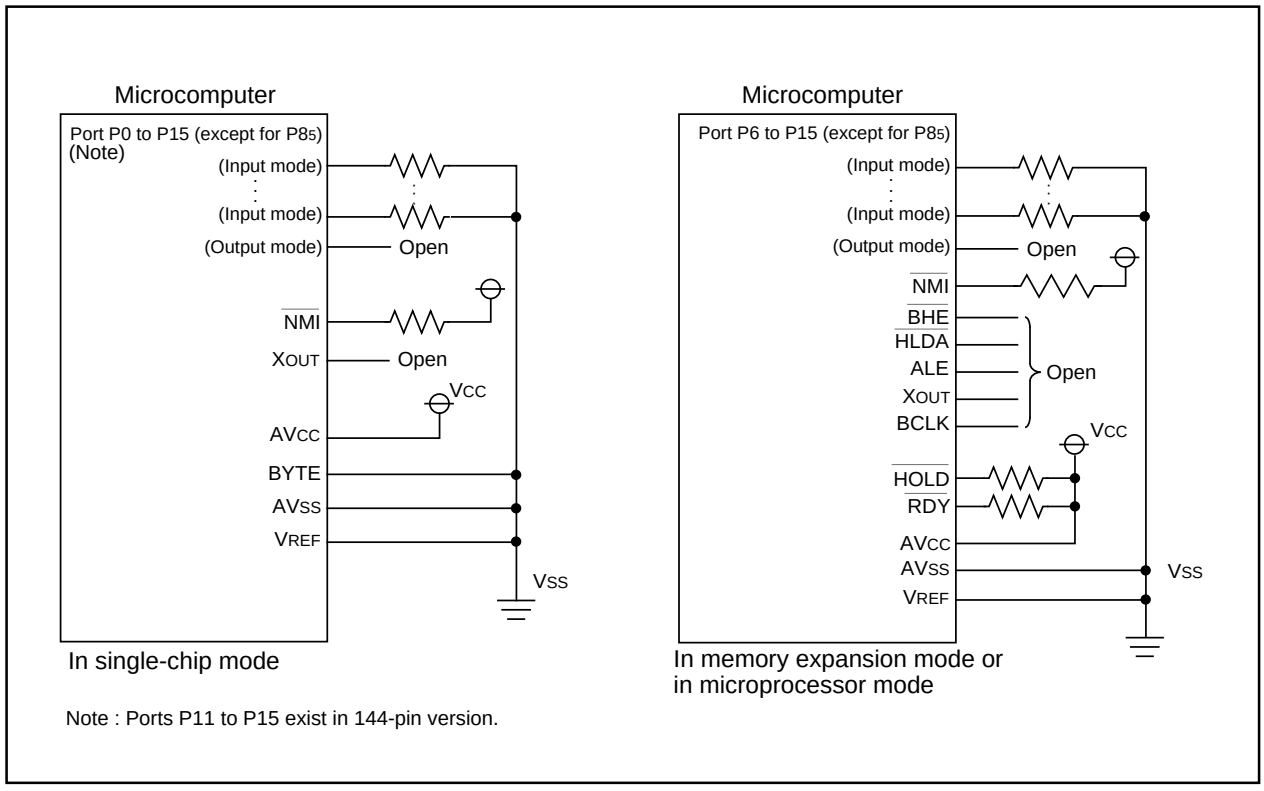



**Figure 1.29.19. Example connection of unused pins**

**Table 1.29.3. Port P6 output control**

| | PS0 register | PSL0 register |
|---|---|---|
| Bit 0 | 0: P6$_0$<br>1: UART0 output ($\overline{RTS0}$)(Note) | Must set to "0" |
| Bit 1 | 0: P6$_1$<br>1: UART0 output (CLK0)(Note) | Must set to "0" |
| Bit 2 | 0: P6$_2$<br>1: Selected by PSL0 register | 0: UART0 output (SCL0)<br>1: UART0 output (STxD0) |
| Bit 3 | 0: P6$_3$<br>1: UART0 output (TxD0/SDA0)(Note) | Must set to "0" |
| Bit 4 | 0: P6$_4$<br>1: Selected by PSL0 register | 0: UART1 output ($\overline{RTS1}$)<br>1: Intelligent I/O group 2 (OUTC2$_1$/ISCLK2) |
| Bit 5 | 0: P6$_5$<br>1: UART1 output (CLK1)(Note) | Must set to "0" |
| Bit 6 | 0: P6$_6$<br>1: Selected by PSL0 register | 0: UART1 output (SCL1)<br>1: UART1 output (STxD1) |
| Bit 7 | 0: P6$_7$<br>1: UART1 output (TxD1/SDA1)(Note) | Must set to "0" |

PS0 register: Function select register A0
PSL0 register: Function select register B0
Note : Select "0" in corresponding bit of PSL0 register.

**Table 1.29.4. Port P7 output control**

| | PS1 register | PSL1 register | PSC register |
|---|---|---|---|
| Bit 0 | 0: P7$_0$<br>1: Selected by PSL1 register | 0: Selected by PSC register<br>1: TImer output (TA0$_{OUT}$)(Note 1) | 0: UART2 output (TxD2/SDA2)<br>1: Intelligent I/O group 2 (OUTC2$_0$/ISTxD2/IE$_{OUT}$) |
| Bit 1 | 0: P7$_1$<br>1: Selected by PSL1 register | 0: Selected by PSC register<br>1: UART2 output (STxD2)(Note 1) | 0: UART2 output (SCL2)<br>1: Intelligent I/O group 2 (OUTC2$_2$) |
| Bit 2 | 0: P7$_2$<br>1: Selected by PSL1 register | 0: Selected by PSC register<br>1: TImer output (TA1$_{OUT}$)(Note 1) | 0: UART2 output (CLK2)<br>1: Three-phase PWM output (V) |
| Bit 3 | 0: P7$_3$<br>1: Selected by PSL1 register | 0: Selected by PSC register<br>1: Three-phase PWM output ($\overline{V}$)(Note 1) | 0: UART2 output ($\overline{RTS2}$)<br>1: Intelligent I/O group 1 (OUTC1$_0$/ISTxD1/BE1$_{OUT}$) |
| Bit 4 | 0: P7$_4$<br>1: Selected by PSL1 register | 0: Selected by PSC register<br>1: Three-phase PWM output (W)(Note 1) | 0: TImer output (TA2$_{OUT}$)<br>1: Intelligent I/O group 1 (OUTC1$_1$/ISCLK1) |
| Bit 5 | 0: P7$_5$<br>1: Selected by PSL1 register | 0: Three-phase PWM output ($\overline{W}$)(Note 1)<br>1: Intelligent I/O group 1 (OUTC1$_2$) | Must set to "0" |
| Bit 6 | 0: P7$_6$<br>1: Selected by PSL1 register | 0: Selected by PSC register<br>1: TImer output (TA3$_{OUT}$) | 0: Intelligent I/O group 0 (OUTC0$_0$/ISTxD0/BE0$_{OUT}$)<br>1: CAN output (CAN$_{OUT}$) |
| Bit 7 | 0: P7$_7$<br>1: Intelligent I/O group 0 (OUTC0$_1$/ISCLK0) | Must set to "0" | 0: Key input interrupt signal enabled<br>1: Key input interrupt signal disabled |

PS1 register: Function select register A1
PSL1 register: Function select register B1
PSC register: Function select register C
Note 1: Select "0" in corresponding bit of PSC register.
Note 2: Select "0" in corresponding bit of PSL1 register.

**Table 1.29.5. Port P8 output control**

| | PS2 register | PSL2 register |
|---|---|---|
| Bit 0 | 0: $P8_0$<br>1: Selected by PSL2 register | 0: Timer output ($TA4_{OUT}$)<br>1: Three-phase PWM output (U) |
| Bit 1 | 0: $P8_1$<br>1: Selected by PSL2 register | 0: Three-phase PWM output ($\overline{U}$)<br>1: Intelligent I/O group 3($OUTC3_0$) |
| Bit 2 | 0: $P8_2$<br>1: Selected by PSL2 register | 0: Intelligent I/O group 3($OUTC3_2$)<br>1: CAN output ($CAN_{OUT}$) |
| Bit 3 to 7 Must set to "0" | | |

PS2 register: Function select register A2
PSL2 register: Function select register B2

**Table 1.29.6. Port P9 output control**

| | PS3 register | PSL3 register |
|---|---|---|
| Bit 0 | 0: $P9_0$<br>1: UART3 output (CLK3)(Note) | Must set to "0" |
| Bit 1 | 0: $P9_1$<br>1: Selected by PSL3 register | 0: UART3 output (SCL3)<br>1: UART3 output (STxD3) |
| Bit 2 | 0: $P9_2$<br>1: Selected by PSL3 register | 0: UART3 output (TxD3/SDA3)<br>1: Intelligent I/O group 2 ($OUTC2_0$/$IE_{OUT}$) |
| Bit 3 | 0: $P9_3$<br>1: UART3 output ($\overline{RTS3}$)(Note) | 0: Except DA0 output<br>1: DA0 output |
| Bit 4 | 0: $P9_4$<br>1: UART4 output ($\overline{RTS4}$)(Note) | 0: Except DA1 output<br>1: DA1 output |
| Bit 5 | 0: $P9_5$<br>1: UART4 output (CLK4)(Note) | 0: Except ANEX0<br>1: ANEX0 |
| Bit 6 | 0: $P9_6$<br>1: UART4 output (TxD4/SDA4)(Note) | 0: Except ANEX1<br>1: ANEX1 |
| Bit 7 | 0: $P9_7$<br>1: Selected by PSL3 register | 0: UART4 output (SCL4)<br>1: UART4 output (STxD4) |

PS3 register: Function select register A3
PSL3 register: Function select register B3
Note : Select "0" in corresponding bit of PSL3 register.

**Table 1.29.7. Port P11 output control**

| | PS5 register |
|---|---|
| Bit 0 | 0: $P11_0$<br>1: Intelligent I/O group 1($OUTC1_0$/ISTxD1/$BE1_{OUT}$) |
| Bit 1 | 0: $P11_1$<br>1: Intelligent I/O group 1($OUTC1_1$/ISCLK1) |
| Bit 2 | 0: $P11_2$<br>1: Intelligent I/O group 1($OUTC1_2$) |
| Bit 3 | 0: $P11_3$<br>1: Intelligent I/O group 1($OUTC1_3$) |
| Bit 4 to 7 Must set to "0" | |

PS5 register: Function select register A5

**Table 1.29.8. Port P12 output control**

| | PS6 register |
|---|---|
| Bit 0 | 0: $P12_0$<br>1: Intelligent I/O group 3($OUTC3_0$) |
| Bit 1 | 0: $P12_1$<br>1: Intelligent I/O group 3($OUTC3_1$) |
| Bit 2 | 0: $P12_2$<br>1: Intelligent I/O group 3($OUTC3_2$) |
| Bit 3 | 0: $P12_3$<br>1: Intelligent I/O group 3($OUTC3_3$) |
| Bit 4 | 0: $P12_4$<br>1: Intelligent I/O group 3($OUTC3_4$) |
| Bit 5 | 0: $P12_5$<br>1: Intelligent I/O group 3($OUTC3_5$) |
| Bit 6 | 0: $P12_6$<br>1: Intelligent I/O group 3($OUTC3_6$) |
| Bit 7 | 0: $P12_7$<br>1: Intelligent I/O group 3($OUTC3_7$) |

PS6 register: Function select register A6

**Table 1.29.9. Port P13 output control**

| | PS7 register |
|---|---|
| Bit 0 | 0: $P13_0$<br>1: Intelligent I/O group 2($OUTC2_4$) |
| Bit 1 | 0: $P13_1$<br>1: Intelligent I/O group 2($OUTC2_5$) |
| Bit 2 | 0: $P13_2$<br>1: Intelligent I/O group 2($OUTC2_6$) |
| Bit 3 | 0: $P13_3$<br>1: Intelligent I/O group 2($OUTC2_3$) |
| Bit 4 | 0: $P13_4$<br>1: Intelligent I/O group 2($OUTC2_0$/ISTx$D_2$/$IE_{OUT}$) |
| Bit 5 | 0: $P13_5$<br>1: Intelligent I/O group 2($OUTC2_2$) |
| Bit 6 | 0: $P13_6$<br>1: Intelligent I/O group 2($OUTC2_1$/$ISCLK_2$) |
| Bit 7 | 0: $P13_7$<br>1: Intelligent I/O group 2($OUTC2_7$) |

PS7 register: Function select register A7

**Table 1.29.10. Port P14 output control**

| | PS8 register |
|---|---|
| Bit 0 | 0: $P14_0$<br>1: Intelligent I/O group 1($OUTC1_4$) |
| Bit 1 | 0: $P14_1$<br>1: Intelligent I/O group 1($OUTC1_5$) |
| Bit 2 | 0: $P14_2$<br>1: Intelligent I/O group 1($OUTC1_6$) |
| Bit 3 | 0: $P14_3$<br>1: Intelligent I/O group 1($OUTC1_7$) |
| Bit 4 to 7 Must set to "0" | |

PS8 register: Function select register A8

**Table 1.29.11. Port P15 output control**

| | PS9 register |
|---|---|
| Bit 0 | 0: $P15_0$<br>1: Intelligent I/O group 0 ($OUTC0_0$/$ISTxD_0$/$BE_{OUT}$) |
| Bit 1 | 0: $P15_1$<br>1: Intelligent I/O group 0 ($OUTC0_1$/$ISCLK_0$) |
| Bit 2 to 3 Must set to "0" | |
| Bit 4 | 0: $P15_4$<br>1: Intelligent I/O group 0 ($OUTC0_4$) |
| Bit 5 | 0: $P15_5$<br>1: Intelligent I/O group 0 ($OUTC0_5$) |
| Bit 6 to 7 Must set to "0" | |

PS9 register: Function select register A9

## VDC

When power-supply voltage is 3.3V or under, set the internal VDC (Voltage Down Converter) unused.
Follow the steps given below to disable the VDC.

(1) Set bit 3 of the protect register to "1".
(2) Set the VDC control register 0 to "0F16".
(3) Set the VDC control register 0 to "8F16".
(4) Set bit 3 of the protect register to "0".

These steps must be performed after reset as immediately as possible with divide-by-8 clock. When the VDC select bit has been set to "112" once, do not set any other values.

Figure 1.30.1 shows the VDC control register 0.

VDC control register 0 (Note 1)

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

| Symbol | Address | When reset |
|---|---|---|
| VDC0 | 001B16 | 0016 |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| VDC00 | VDC select bit | b1b0<br>1 1: VDC unused<br>Do not set any values other than "11". | ○ | ○ |
| VDC01 | | | ○ | ○ |
| VDC02 | VDC reference voltage select bit | b3 b2<br>1 1: VDC reference voltage Off<br>Do not set any values other than "11". | ○ | ○ |
| VDC03 | | | ○ | ⊖ |
| VDC04 | Reserved bit | Must set to "0" | ○ | ○ |
| VDC05 | | | ○ | ○ |
| VDC06 | | | ○ | ○ |
| VDC07 | VDC enable bit (Note 2) | 0: VDC Off<br>1: VDC On | ○ | ○ |

Note 1: Set bit 3 of the protect register (address 000A16) to "1" before rewriting this register.
Rewriting this register should be performed only when the VDC is to be off.
Note 2: This bit enables the setting of bit 0 to bit 3.
Set bit 7 to "0" first, and then write values to bit 0 to bit 3. After that, write "1" to bit 7.
The state changes at the time "1" is written to bit 7.

**Figure 1.30.1. VDC control register**

## 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 while reloading 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 while reloading 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 as "Free-Run type", the timer register contents may be unknown when counting begins. If the timer register is set before counting has started, then the starting value will be unknown.

- In the case where the up/down count will not be changed.

  Enable the "Reload" function and write to the timer register before counting begins. Rewrite the value to the timer register immediately after counting has started. If counting up, rewrite "0000$_{16}$" to the timer register. If counting down, rewrite "FFFF$_{16}$" to the timer register. This will cause the same operation as "Free-Run type" mode.

- In the case where the up/down count has changed.

  First set to "Reload type" operation. Once the first counting pulse has occurred, the timer may be changed to "Free-Run type".

### 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 output from the one-shot timer synchronizes with the count source generated internally. Therefore, when an external trigger has been selected, a delay of one cycle of count source as maximum occurs between the trigger input to the TAi$_{IN}$ pin and the one-shot timer output.

(3) 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.

(4) If a trigger occurs while a count is in progress, after the counter performs one down count following the reoccurrence of a trigger, the reload register contents are reloaded, and the count continues. To generate a trigger while a count is in progress, generate the second trigger after an elapse longer than one cycle of the timer's count source after the previous trigger occurred.

### Timer A (pulse width modulation mode)

(1) The timer Ai interrupt request bit becomes “1” if setting operation mode of the timer in compliance with any of the following procedures:
- Selecting PWM mode after reset.
- Changing operation mode from timer mode to PWM mode.
- Changing operation mode from event counter mode to PWM mode.

Therefore, to use timer Ai interrupt (interrupt request bit), set timer Ai interrupt request bit to “0” after the above listed changes have been made.

(2) Setting the count start flag to “0” while PWM pulses are being output causes the counter to stop counting. If the TAiOUT pin is outputting an “H” level in this instance, the output level goes to “L”, and the timer Ai interrupt request bit goes to “1”. If the TAiOUT pin is outputting an “L” level in this instance, the level does not change, and the timer Ai interrupt request bit does not becomes “1”.

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

(1) Reading the timer Bi register while a count is in progress allows reading, with arbitrary timing, the value of the counter. Reading the timer Bi register while reloading gets “$FFFF_{16}$”. 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.

(3) The value of the counter is indeterminate at the beginning of a count. Therefore, the timer Bi overflow flag may go to “1” and timer Bi interrupt request may be generated during the interval between a count start and an effective edge input.

### 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 shifting to WAIT mode or STOP mode, the program stops after reading from the WAIT instruction and the instruction that sets all clock stop control bits to “1” in the instruction queue. Therefore, insert a minimum of 4 NOPs after the WAIT instruction and the instruction that sets all clock stop control bits to “1” in order to flush the instruction queue.

## A-D Converter

(1) Write to each bit (except bit 6) of A-D i (i=0,1) control register 0, to each bit of A-D i control register 1, and to each bit of A-D i 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.

(5) When f($X_{IN}$) is faster than 10 MHz, make the frequency 10 MHz or less by dividing.

(6) Output impedance of sensor at A-D conversion (Reference value)
To carry out A-D conversion properly, charging the internal capacitor C shown in Figure 1.31.1 has to be completed within a specified period of time T. Let output impedance of sensor equivalent circuit be R0, microcomputer's internal resistance be R, precision (error) of the A-D converter be X, and the A-D converter's resolution be Y (Y is 1024 in the 10-bit mode, and 256 in the 8-bit mode).

Vc is generally $V_C = V_{IN} \{1 - e^{-\frac{t}{C(R0 + R)}}\}$

And when t = T, $V_C = V_{IN} - \frac{X}{Y} V_{IN} = V_{IN}(1 - \frac{X}{Y})$

$$e^{-\frac{T}{C(R0 + R)}} = \frac{X}{Y}$$

$$-\frac{T}{C(R0 + R)} = \ln \frac{X}{Y}$$

Hence, $R0 = -\frac{T}{C \cdot \ln \frac{X}{Y}} - R$

With the model shown in Figure 1.31.1 as an example, when the difference between $V_{IN}$ and $V_C$ becomes 0.1LSB, we find impedance R0 when voltage between pins $V_C$ changes from 0 to $V_{IN}$-(0.1/1024) $V_{IN}$ in time T. (0.1/1024) means that A-D precision drop due to insufficient capacitor charge is held to 0.1LSB at time of A-D conversion in the 10-bit mode. Actual error however is the value of absolute precision added to 0.1LSB. When f($X_{IN}$) = 10 MHz, T = 0.3 μs in the A-D conversion mode with sample & hold. Output impedance R0 for sufficiently charging capacitor C within time T is determined as follows.

T = 0.3 μs, R = 7.8 kΩ, C = 3 pF, X = 0.1, and Y = 1024 . Hence,

$$R0 = -\frac{0.3 \times 10^{-6}}{3.0 \times 10^{-12} \cdot \ln \frac{0.1}{1024}} - 7.8 \times 10^3 \fallingdotseq 3.0 \times 10^3$$

Thus, the allowable output impedance of the sensor circuit capable of thoroughly driving the A-D converter turns out to be approximately 3.0 kΩ. Tables 1.31.1 and 1.31.2 show output impedance values based on the LSB values.

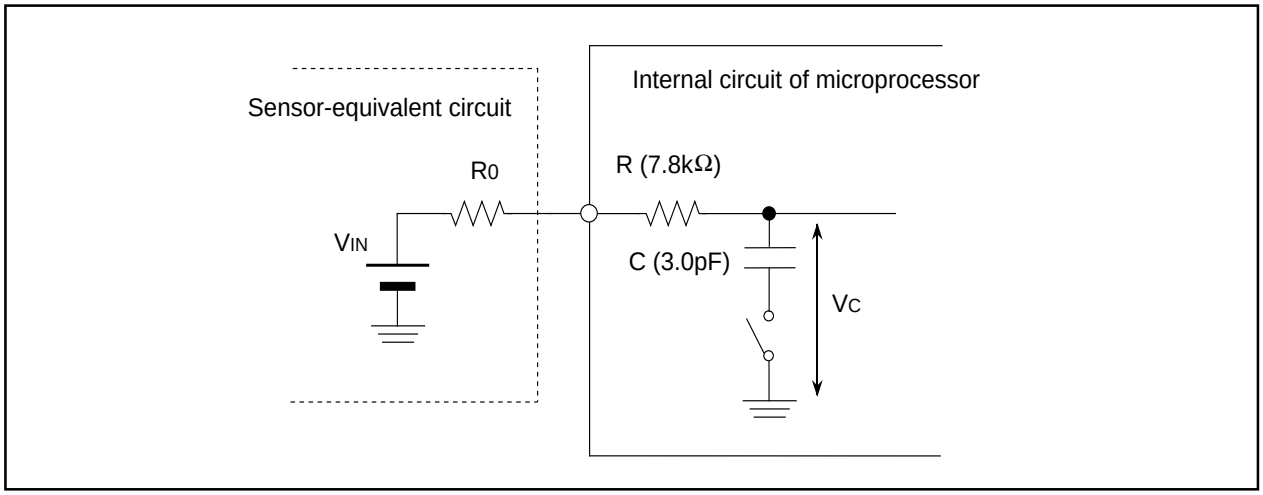



**Figure 1.31.1 A circuit equivalent to the A-D conversion terminal**

(7) After A-D conversion is complete, if the CPU reads the A-D register at the same time as the A-D conversion result is being saved to A-D register, wrong A-D conversion value is saved into the A-D register. This happens when the internal CPU clock is selected from divided main clock or sub-clock.

- **When using the one-shot or single sweep mode**
  Confirm that A-D conversion is complete before reading the A-D register.
  (Note: When A-D conversion interrupt request bit is set, it shows that A-D conversion is completed.)
- **When using the repeat mode or repeat sweep mode 0 or 1**
  Use the undivided main clock as the internal CPU clock.

## Interrupts

(1) Setting the stack pointer

- The value of the stack pointer is initialized to $000000_{16}$ immediately after reset. Accepting an interrupt before setting a value in the stack pointer may cause 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 pointer at the beginning of a program. Regarding the first instruction immediately after reset, generating any interrupts including the $\overline{\text{NMI}}$ interrupt is prohibited.
  Set an even address to the stack pointer so that operating efficiency is increased.

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

- As for the $\overline{\text{NMI}}$ interrupt pin, an interrupt cannot be prohibited. Connect it to the $V_{CC}$ pin via a resistance (pulled-up) if unused.
- 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.
- Signal of "L" level width more than 1 clock of CPU operation clock (BCLK) is necessary for $\overline{\text{NMI}}$ pin.

**Tables 1.31.1. Output impedance values based on the LSB values (10-bit mode) Reference value**

| f(XIN) (MHz) | Cycle (µs) | Sampling time (µs) | R (kΩ) | C (pF) | Resolution (LSB) | R0max (kΩ) |
|---|---|---|---|---|---|---|
| 10 | 0.1 | 0.3 (3 X cycle, Sample & hold bit is enabled) | 7.8 | 3.0 | 0.1 | 3.0 |
| | | | | | 0.3 | 4.5 |
| | | | | | 0.5 | 5.3 |
| | | | | | 0.7 | 5.9 |
| | | | | | 0.9 | 6.4 |
| | | | | | 1.1 | 6.8 |
| | | | | | 1.3 | 7.2 |
| | | | | | 1.5 | 7.5 |
| | | | | | 1.7 | 7.8 |
| | | | | | 1.9 | 8.1 |
| 10 | 0.1 | 0.2 (2 X cycle, Sample & hold bit is disabled) | 7.8 | 3.0 | 0.3 | 0.4 |
| | | | | | 0.5 | 0.9 |
| | | | | | 0.7 | 1.3 |
| | | | | | 0.9 | 1.7 |
| | | | | | 1.1 | 2.0 |
| | | | | | 1.3 | 2.2 |
| | | | | | 1.5 | 2.4 |
| | | | | | 1.7 | 2.6 |
| | | | | | 1.9 | 2.8 |

**Tables 1.31.2. Output impedance values based on the LSB values (8-bit mode) Reference value**

| f(XIN) (MHz) | Cycle (µs) | Sampling time (µs) | R (kΩ) | C (pF) | Resolution (LSB) | R0max (kΩ) |
|---|---|---|---|---|---|---|
| 10 | 0.1 | 0.3 (3 X cycle, Sample & hold bit is enabled) | 7.8 | 3.0 | 0.1 | 4.9 |
| | | | | | 0.3 | 7.0 |
| | | | | | 0.5 | 8.2 |
| | | | | | 0.7 | 9.1 |
| | | | | | 0.9 | 9.9 |
| | | | | | 1.1 | 10.5 |
| | | | | | 1.3 | 11.1 |
| | | | | | 1.5 | 11.7 |
| | | | | | 1.7 | 12.1 |
| | | | | | 1.9 | 12.6 |
| 10 | 0.1 | 0.2 (2 X cycle, Sample & hold bit is disabled) | 7.8 | 3.0 | 0.1 | 0.7 |
| | | | | | 0.3 | 2.1 |
| | | | | | 0.5 | 2.9 |
| | | | | | 0.7 | 3.5 |
| | | | | | 0.9 | 4.0 |
| | | | | | 1.1 | 4.4 |
| | | | | | 1.3 | 4.8 |
| | | | | | 1.5 | 5.2 |
| | | | | | 1.7 | 5.5 |
| | | | | | 1.9 | 5.8 |

(3) External interrupt

- Edge sense

  Either an “L” level or an “H” level of at least 250 ns width is necessary for the signal input to pins $\overline{INT_0}$ to $\overline{INT_5}$ regardless of the CPU operation clock.
- Level sense

  Either an “L” level or an “H” level of 1 cycle of BCLK + at least 200 ns width is necessary for the signal input to pins $\overline{INT_0}$ to $\overline{INT_5}$ regardless of the CPU operation clock. (When $X_{IN}$=30MHz and no division mode, at least 233 ns width is necessary.)
- When the polarity of the $\overline{INT_0}$ to $\overline{INT_5}$ pins is changed, the interrupt request bit is sometimes set to "1". After changing the polarity, set the interrupt request bit to "0". Figure 1.31.2 shows the procedure for changing the $\overline{INT}$ interrupt generate factor.

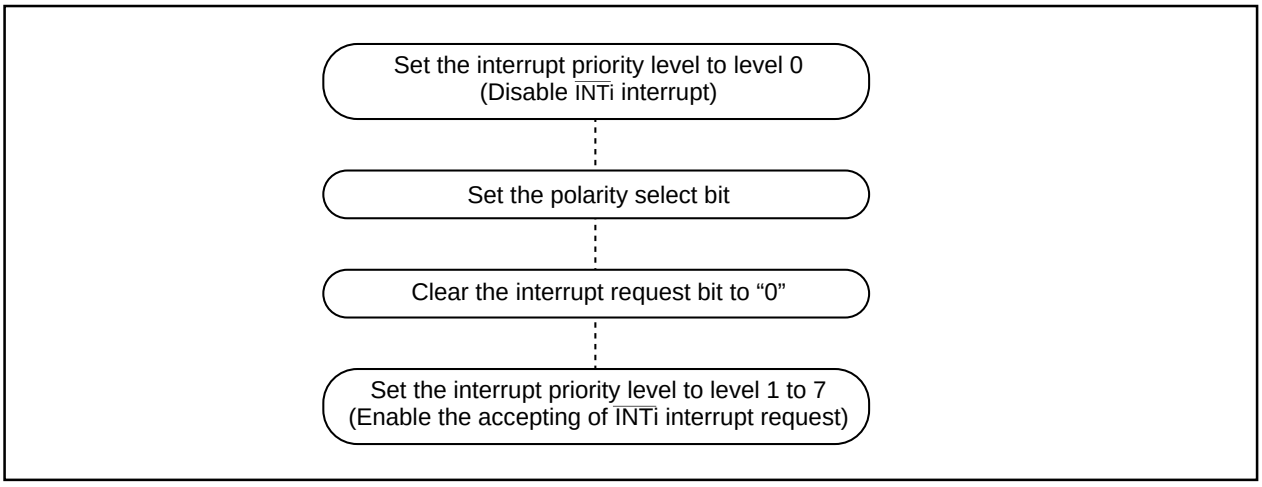



**Figure 1.31.2. Switching condition of $\overline{INT}$ interrupt request**

(4) Rewrite the interrupt control register

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 instructions. If this creates problems, use the below instructions to change the register.

Instructions : AND, OR, BCLR, BSET

## DMAC

(1) Do not clear the DMA request bit of the DMAi request cause select register.

In M32C/80 series, when a DMA request is generated while the channel is disabled (Note), the DMA transfer is not executed and the DMA request bit is cleared automatically.

Note :The DMA is disabled or the transfer count register is "0".

(2) When DMA transfer is done by a software trigger, set DSR and DRQ of the DMAi request cause select register to "1" simultaneously using the OR instruction.

```
e.g.) OR.B   #0A0h, DMiSL      ; DMiSL is DMAi request cause select register
```

(3) When changing the DMAi request cause select bit of the DMAi request cause select register, set "1" to the DMA request bit, simultaneously. In this case, disable the corresponding DMA channel to disabled before changing the DMAi request cause select bit. To enable DMA at least 8+6xN cycles (N: enabled channel number) following the instruction to write to the DMAi request cause select register are needed.

Example) When DMA request cause is changed to timer A0 and using DMA0 in single transfer after DMA initial setting

```
push.w    R0                ; Store R0 register
stc       DMD0, R0          ; Read DMA mode register 0
and.b     #11111100b, R0L   ; Clear DMA0 transfer mode select bit to "00"
ldc       R0, DMD0          ; DMA0 disabled
mov.b     #10000011b, DM0SL ; Select timer A0
                            ; (Write "1" to DMA request bit simultaneously)
nop                         } At least 8 + 6 x N cycles
 :                          } (N: enabled channel number)
ldc       R0, DMD0          ; DMA0 enabled
pop.w R0                    ; Restore R0 register
```

## Noise

A bypass capacitor should be inserted between Vcc-Vss line for reducing noise and latch-up
Connect a bypass capacitor (approx. 0.1µF) between the Vcc and Vss pins using short wiring and thicker circuit traces.

## Precautions for using CLKOUT pin

When using the Clock Output function of P53/CLKOUT pin ($f_8$, $f_{32}$ or fc output) in single chip mode, use port P57 as an input only port (port P57 direction register is "0").
Although port P57 may be set as an output port (port P57 direction register is "1"), it will become high impedance and will not output "H" or "L" levels.

## $\overline{\text{HOLD}}$ signal

When using the $\overline{\text{HOLD}}$ input while P40 to P47 and P50 to P52 are set as output ports in single-chip mode, you must first set all pins for P40 to P47 and P50 to P52 as input ports, then shift to microprocessor mode or memory expansion mode.

## Reducing power consumption

(1) When A-D conversion is not performed, select the Vref not connected with the Vref connect bit of A-D control register 1. When A-D conversion is performed, start the A-D conversion at least 1 µs or longer after connecting Vref.

(2) When using AN4 (P104) to AN7 (P107), select the input disable of the key input interrupt signal with the key input interrupt disable bit of the function select register C .
When selecting the input disable of the key input interrupt signal, the key input interrupt cannot be used. Also, the port cannot be input even if the direction register of P104 to P107 is set to input (the input result becomes undefined). When the input disable of the key input interrupt signal is selected, use all AN4 to AN7 as A-D inputs.

(3) When ANEX0 and ANEX1 are used, select the input peripheral function disable with port P95 and P96 input peripheral function select bit of the function select register B3.
When the input peripheral function disable is selected, the port cannot be input even if the port direction register is set to input (the input result becomes undefined).
Also, it is not possible to input a peripheral function except ANEX0 and ANEX1.

(4) When D-A converter is not used, set output disabled with the D-A output enable bit of D-A control register and set the D-A register to "00$_{16}$".

(5) When D-A conversion is used, select the input peripheral function disabled with port P9$_3$ and P9$_4$ input peripheral function select bit of the function select register B3.
When the input peripheral function disabled is selected, the port cannot be input even if the port direction register is set to input (the input result becomes undefined).
Also, it is not possible to input a peripheral function.

## DRAM controller

The DRAM self-refresh operates in stop mode, etc.

When shifting to self-refresh, select DRAM is ignored by the DRAM space select bit. In the next instruction, simultaneously set the DRAM space select bit and self-refresh ON by self-refresh mode bit. Also, insert two NOPs after the instruction that sets the self-refresh mode bit to "1".

Do not access external memory while operating in self-refresh. (All external memory space access is inhibited. )

When disabling self-refresh, simultaneously select DRAM is ignored by the DRAM space select bit and self-refresh OFF by self-refresh mode bit. In the next instruction, set the DRAM space select bit.

Do not access the DRAM space immediately after setting the DRAM space select bit.

Example) One wait is selected by the wait select bit and 4MB is selected by the DRAM space select bit

Shifting to self-refresh

```
•••
mov.b   #00000001b,DRAMCONT     ;DRAM is ignored, one wait is selected
mov.b   #10001011b,DRAMCONT     ;Set self-refresh, select 4MB and one wait
nop                             ;Two nops are needed
nop                             ;
•••
```

Disable self-refresh

```
•••
mov.b   #00000001b,DRAMCONT     ;Disable self-refresh, DRAM ignored, one wait is
                                ;selected
mov.b   #00001011b,DRAMCONT     ;Select 4MB and one wait
nop                             ;Inhibit instruction to access DRAM area
nop
•••
```

## Setting the registers

The registers shown in Table 1.31.3 include indeterminate bit when read. Set immidiate to these registers.

Store the content of the frequently used register to RAM, change the content of RAM, then transfer to the register.

**Table 1.31.3 The object registers**

| Register name | Symbol | Address |
|---|---|---|
| Watchdog timer start register | WDTS | 000E16 |
| Group0 receive input register | G0RI | 00EC16 |
| Group1 receive input register | G1RI | 012C16 |
| Group2 SI/O transmit buffer register | G2TB | 016D16, 016C16 |
| UART4 bit rate generator | U4BRG | 02F916 |
| UART4 transfer buffer register | U4TB | 02FB16, 02FA16 |
| Timer A1-1 register | TA11 | 030316, 030216 |
| Timer A2-1 register | TA21 | 030516, 030416 |
| Timer A4-1 register | TA41 | 030716, 030616 |
| Dead time timer | DTT | 030C16 |
| Timer B2 interrupt occurrence frequency set counter | ICTB2 | 030D16 |
| UART3 bit rate generator | U3BRG | 032916 |
| UART3 transfer buffer register | U3TB | 032B16, 032A16 |
| UART2 bit rate generator | U2BRG | 033916 |
| UART2 transfer buffer register | U2TB | 033B16, 033A16 |
| Up-down flag | UDF | 034416 |
| Timer A0 register (Note) | TA0 | 034716, 034616 |
| Timer A1 register (Note) | TA1 | 034916, 034816 |
| Timer A2 register (Note) | TA2 | 034B16, 034A16 |
| Timer A3 register (Note) | TA3 | 034D16, 034C16 |
| Timer A4 register (Note) | TA4 | 034F16, 034E16 |
| UART0 bit rate generator | U0BRG | 036916 |
| UART0 transfer buffer register | U0TB | 036B16, 036A16 |
| UART1 bit rate generator | U1BRG | 02E916 |
| UART1 transfer buffer register | U1TB | 02EB16, 02EA16 |
| A-D0 control register 2 | ADCON2 | 039416 |

Note: In one-shot timer mode and pulse width modulation mode.

## Notes on the microprocessor mode and transition after shifting from the microprocessor mode to the memory expansion mode / single-chip mode

In microprocessor mode, the SFR, internal RAM, and external memory space can be accessed.
For that reason, the internal ROM area cannot be accessed.
After the reset has been released and the operation of shifting from the microprocessor mode has started ("H" applied to the CNVSS pin), the internal ROM area cannot be accessed even if the CPU shifts to the memory expansion mode or single-chip mode.

## Notes on CNVss pin reset at "H" level

When the CNVss pin is reset at "H" level, the contents of internal ROM cannot be read out.

## Electrical characteristics

**Table 1.32.1. Absolute maximum ratings**

| Symbol | Parameter | | Condition | Rated value | Unit |
|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | $V_{CC}$=AV$_{CC}$ | -0.3 to 6.0 | V |
| AV$_{CC}$ | Analog supply voltage | | $V_{CC}$=AV$_{CC}$ | -0.3 to 6.0 | V |
| $V_I$ | Input voltage | $\overline{RESET}$, CNVss, BYTE, $P0_0$-$P0_7$, $P1_0$-$P1_7$, $P2_0$-$P2_7$, $P3_0$-$P3_7$, $P4_0$-$P4_7$, $P5_0$-$P5_7$, $P6_0$-$P6_7$, $P7_2$-$P7_7$, $P8_0$-$P8_7$, $P9_0$-$P9_7$, $P10_0$-$P10_7$, $P11_0$-$P11_4$, $P12_0$-$P12_7$, $P13_0$-$P13_7$, $P14_0$-$P14_6$, $P15_0$-$P15_7$(Note1), $V_{REF}$, $X_{IN}$ | | -0.3 to Vcc+0.3 | V |
| | | $P7_0$, $P7_1$ | | -0.3 to 6.0 | V |
| $V_O$ | Output voltage | $P0_0$-$P0_7$, $P1_0$-$P1_7$, $P2_0$-$P2_7$, $P3_0$-$P3_7$, $P4_0$-$P4_7$, $P5_0$-$P5_7$, $P6_0$-$P6_7$, $P7_2$-$P7_7$, $P8_0$-$P8_7$, $P9_0$-$P9_7$, $P10_0$-$P10_7$, $P11_0$-$P11_4$, $P12_0$-$P12_7$, $P13_0$-$P13_7$, $P14_0$-$P14_6$, $P15_0$-$P15_7$(Note1), $V_{REF}$, $X_{IN}$ | | -0.3 to Vcc+0.3 | V |
| | | $P7_0$, $P7_1$ | | -0.3 to 6.0 | V |
| Pd | Power dissipation | | Topr=25°C | 500 | mW |
| Topr | Operating ambient temperature | | | -20 to 85/-40 to 85(Note 2) | °C |
| Tstg | Storage temperature | | | -65 to 150 | °C |

Note 1: Ports P11 to P15 exist in 144-pin version.

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

**Table 1.32.2. Recommended operating conditions (referenced to $V_{CC}$ = 3.0V to 5.5V at Topr = – 20 to 85°C / – 40 to 85°C(Note3) unless otherwise specified)**

| Symbol | Parameter | | | Standard | | | Unit |
|---|---|---|---|---|---|---|---|
| | | | | Min. | Typ. | Max. | |
| $V_{CC}$ | Supply voltage(When VDC-ON) | | | 3.0 | 5.0 | 5.5 | V |
| | Supply voltage(When VDC-pass through) | | | 3.0 | 3.3 | 3.6 | V |
| $AV_{CC}$ | Analog supply voltage | | | | $V_{CC}$ | | V |
| $V_{SS}$ | Supply voltage | | | | 0 | | V |
| $AV_{SS}$ | Analog supply voltage | | | | 0 | | V |
| $V_{IH}$ | "H" input voltage | $P2_0$-$P2_7$, $P3_0$-$P3_7$, $P4_0$-$P4_7$, $P5_0$-$P5_7$, $P6_0$-$P6_7$, $P7_2$-$P7_7$, $P8_0$-$P8_7$, $P9_0$-$P9_7$, $P10_0$-$P10_7$, $P11_0$-$P11_4$, $P12_0$-$P12_7$, $P13_0$-$P13_7$, $P14_0$-$P14_6$, $P15_0$-$P15_7$(Note5), $X_{IN}$, $\overline{RESET}$, CNVss, BYTE | | 0.8Vcc | | Vcc | V |
| | | $P7_0$, $P7_1$ | | 0.8Vcc | | 6.0 | V |
| | | $P0_0$-$P0_7$, $P1_0$-$P1_7$ (during single-chip mode) | | 0.8Vcc | | Vcc | V |
| | | $P0_0$-$P0_7$, $P1_0$-$P1_7$ (during memory-expansion and microprocessor modes) | | 0.5Vcc | | Vcc | V |
| $V_{IL}$ | "L" input voltage | $P2_0$-$P2_7$, $P3_0$-$P3_7$, $P4_0$-$P4_7$, $P5_0$-$P5_7$, $P6_0$-$P6_7$, $P7_0$-$P7_7$, $P8_0$-$P8_7$, $P9_0$-$P9_7$, $P10_0$-$P10_7$, $P11_0$-$P11_4$, $P12_0$-$P12_7$, $P13_0$-$P13_7$, $P14_0$-$P14_6$, $P15_0$-$P15_7$(Note5), $X_{IN}$, $\overline{RESET}$, CNVss, BYTE | | 0 | | 0.2Vcc | V |
| | | $P0_0$-$P0_7$, $P1_0$-$P1_7$ (during single-chip mode) | | 0 | | 0.2Vcc | V |
| | | $P0_0$-$P0_7$, $P1_0$-$P1_7$ (during memory-expansion and microprocessor modes) | | 0 | | 0.16Vcc | V |
| $I_{OH(peak)}$ | "H" peak output current | $P0_0$-$P0_7$, $P1_0$-$P1_7$, $P2_0$-$P2_7$, $P3_0$-$P3_7$, $P4_0$-$P4_7$, $P5_0$-$P5_7$, $P6_0$-$P6_7$, $P7_0$-$P7_7$, $P8_0$-$P8_4$, $P8_6$, $P8_7$, $P9_0$-$P9_7$, $P10_0$-$P10_7$, $P11_0$-$P11_4$, $P12_0$-$P12_7$, $P13_0$-$P13_7$, $P14_0$-$P14_6$, $P15_0$-$P15_7$(Note5) | | | | -10.0 | mA |
| $I_{OH(avg)}$ | "H" average output current | $P0_0$-$P0_7$, $P1_0$-$P1_7$, $P2_0$-$P2_7$, $P3_0$-$P3_7$, $P4_0$-$P4_7$, $P5_0$-$P5_7$, $P6_0$-$P6_7$, $P7_0$-$P7_7$, $P8_0$-$P8_4$, $P8_6$, $P8_7$, $P9_0$-$P9_7$, $P10_0$-$P10_7$, $P11_0$-$P11_4$, $P12_0$-$P12_7$, $P13_0$-$P13_7$, $P14_0$-$P14_6$, $P15_0$-$P15_7$(Note5) | | | | -5.0 | mA |
| $I_{OL(peak)}$ | "L" peak output current | $P0_0$-$P0_7$, $P1_0$-$P1_7$, $P2_0$-$P2_7$, $P3_0$-$P3_7$, $P4_0$-$P4_7$, $P5_0$-$P5_7$, $P6_0$-$P6_7$, $P7_0$-$P7_7$, $P8_0$-$P8_4$, $P8_6$, $P8_7$, $P9_0$-$P9_7$, $P10_0$-$P10_7$, $P11_0$-$P11_4$, $P12_0$-$P12_7$, $P13_0$-$P13_7$, $P14_0$-$P14_6$, $P15_0$-$P15_7$(Note5) | | | | 10.0 | mA |
| $I_{OL(avg)}$ | "L" average output current | $P0_0$-$P0_7$, $P1_0$-$P1_7$, $P2_0$-$P2_7$, $P3_0$-$P3_7$, $P4_0$-$P4_7$, $P5_0$-$P5_7$, $P6_0$-$P6_7$, $P7_0$-$P7_7$, $P8_0$-$P8_4$, $P8_6$, $P8_7$, $P9_0$-$P9_7$, $P10_0$-$P10_7$, $P11_0$-$P11_4$, $P12_0$-$P12_7$, $P13_0$-$P13_7$, $P14_0$-$P14_6$, $P15_0$-$P15_7$(Note5) | | | | 5.0 | mA |
| $f(X_{IN})$ | Main clock input frequency | VDC-ON | Vcc=4.2 to 5.5V | 0 | | 30 | MHz |
| | | | Vcc=3.0 to 4.2V | 0 | | 20 | MHz |
| | | VDC-pass through | Vcc=3.0 to 3.6V | 0 | | 20 | MHz |
| $f(X_{CIN})$ | Sub-clock oscillation frequency | | | | 32.768 | | kHz |

Note 1: The mean output current is the mean value within 100ms.
Note 2: The total $I_{OL}$ (peak) for ports P0, P1, P2, $P8_6$, $P8_7$, P9, P10, P11, P14 and P15 must be 80mA max. The total $I_{OH}$ (peak) for ports P0, P1, P2, $P8_6$, $P8_7$, P9, P10, P11, P14 and P15 must be -80mA max. The total $I_{OL}$ (peak) for ports P3, P4, P5, P6, $P7_2$ to $P7_7$, $P8_0$ to $P8_4$, P12 and P13 must be 80mA max. The total $I_{OH}$ (peak) for ports P3, P4, P5, P6, $P7_2$ to $P7_7$, $P8_0$ to $P8_4$, P12 and P13 must be -80mA max.
Note 3: Specify a product of -40 to 85°C to use it.
Note 4: The specification of $V_{IH}$ and $V_{IL}$ of $P8_7$ is not when using as $X_{CIN}$ but when using programmable input port.
Note 5: Port P11 to P15 exist in 144-pin version.

# Electrical characteristics (Vcc = 5V)

**Table 1.32.3. Electrical characteristics (referenced to $V_{CC}$=4.2 to 5.5V, $V_{SS}$=0V at Topr= –20 to 85°C, f($X_{IN}$)=30MHz unless otherwise specified)** $V_{CC}$ = 5V

| Symbol | Parameter | | Condition | Standard | | | Unit |
|---|---|---|---|---|---|---|---|
| | | | | Min. | Typ. | Max. | |
| $V_{OH}$ | "H" output voltage | $P0_0$-$P0_7$, $P1_0$-$P1_7$, $P2_0$-$P2_7$, $P3_0$-$P3_7$, $P4_0$-$P4_7$, $P5_0$-$P5_7$, $P6_0$-$P6_7$, $P7_0$-$P7_7$, $P8_0$-$P8_4$, $P8_6$, $P8_7$, $P9_0$-$P9_7$, $P10_0$-$P10_7$, $P11_0$-$P11_4$, $P12_0$-$P12_7$, $P13_0$-$P13_7$, $P14_0$-$P14_6$, $P15_0$-$P15_7$(Note) | $I_{OH}$=-5mA | 3.0 | | | V |
| $V_{OH}$ | "H" output voltage | $P0_0$-$P0_7$, $P1_0$-$P1_7$, $P2_0$-$P2_7$, $P3_0$-$P3_7$, $P4_0$-$P4_7$, $P5_0$-$P5_7$, $P6_0$-$P6_7$, $P7_0$-$P7_7$, $P8_0$-$P8_4$, $P8_6$, $P8_7$, $P9_0$-$P9_7$, $P10_0$-$P10_7$, $P11_0$-$P11_4$, $P12_0$-$P12_7$, $P13_0$-$P13_7$, $P14_0$-$P14_6$, $P15_0$-$P15_7$(Note) | $I_{OH}$=-200µA | 4.7 | | | V |
| $V_{OH}$ | "H" output voltage | $X_{OUT}$ | $I_{OH}$=-1mA | 3.0 | | | V |
| | "H" output voltage | $X_{COUT}$ | No load applied | | 3.3 | | V |
| $V_{OL}$ | "L" output voltage | $P0_0$-$P0_7$, $P1_0$-$P1_7$, $P2_0$-$P2_7$, $P3_0$-$P3_7$, $P4_0$-$P4_7$, $P5_0$-$P5_7$, $P6_0$-$P6_7$, $P7_0$-$P7_7$, $P8_0$-$P8_4$, $P8_6$, $P8_7$, $P9_0$-$P9_7$, $P10_0$-$P10_7$, $P11_0$-$P11_4$, $P12_0$-$P12_7$, $P13_0$-$P13_7$, $P14_0$-$P14_6$, $P15_0$-$P15_7$(Note) | $I_{OH}$=5mA | | | 2.0 | V |
| $V_{OL}$ | "L" output voltage | $P0_0$-$P0_7$, $P1_0$-$P1_7$, $P2_0$-$P2_7$, $P3_0$-$P3_7$, $P4_0$-$P4_7$, $P5_0$-$P5_7$, $P6_0$-$P6_7$, $P7_0$-$P7_7$, $P8_0$-$P8_4$, $P8_6$, $P8_7$, $P9_0$-$P9_7$, $P10_0$-$P10_7$, $P11_0$-$P11_4$, $P12_0$-$P12_7$, $P13_0$-$P13_7$, $P14_0$-$P14_6$, $P15_0$-$P15_7$(Note) | $I_{OH}$=200µA | | | 0.45 | V |
| $V_{OL}$ | "L" output voltage | $X_{OUT}$ | $I_{OL}$=1mA | | | 2.0 | V |
| | "L" output voltage | $X_{COUT}$ | No load applied | | 0 | | V |
| $V_{T+}$-$V_{T-}$ | Hysteresis | $\overline{HOLD}$, $\overline{RDY}$, $TA0_{IN}$-$TA4_{IN}$, $TB0_{IN}$-$TB5_{IN}$, $\overline{INT0}$-$\overline{INT5}$, $AD_{TRG}$, $\overline{CTS0}$-$\overline{CTS4}$, CLK0-CLK4, $TA0_{OUT}$-$TA4_{OUT}$, $\overline{NMI}$, $\overline{KI0}$-$\overline{KI3}$, RxD0-RxD4, SCL0-SCL4, SDA0-SDA4 | | 0.2 | | 1.0 | V |
| $V_{T+}$-$V_{T-}$ | Hysteresis | $\overline{RESET}$ | | 0.2 | | 1.8 | V |
| $I_{IH}$ | "H" input current | $P0_0$-$P0_7$, $P1_0$-$P1_7$, $P2_0$-$P2_7$, $P3_0$-$P3_7$, $P4_0$-$P4_7$, $P5_0$-$P5_7$, $P6_0$-$P6_7$, $P7_2$-$P7_7$, $P8_0$-$P8_7$, $P9_0$-$P9_7$, $P10_0$-$P10_7$, $P11_0$-$P11_4$, $P12_0$-$P12_7$, $P13_0$-$P13_7$, $P14_0$-$P14_6$, $P15_0$-$P15_7$(Note), $X_{IN}$, $\overline{RESET}$, CNVss, BYTE | $V_I$=5V | | | 5.0 | µA |
| $I_{IL}$ | "L" input current | $P0_0$-$P0_7$, $P1_0$-$P1_7$, $P2_0$-$P2_7$, $P3_0$-$P3_7$, $P4_0$-$P4_7$, $P5_0$-$P5_7$, $P6_0$-$P6_7$, $P7_2$-$P7_7$, $P8_0$-$P8_7$, $P9_0$-$P9_7$, $P10_0$-$P10_7$, $P11_0$-$P11_4$, $P12_0$-$P12_7$, $P13_0$-$P13_7$, $P14_0$-$P14_6$, $P15_0$-$P15_7$(Note), $X_{IN}$, $\overline{RESET}$, CNVss, BYTE | $V_I$=0V | | | -5.0 | µA |
| $R_{PULLUP}$ | Pull-up resistance | $P0_0$-$P0_7$, $P1_0$-$P1_7$, $P2_0$-$P2_7$, $P3_0$-$P3_7$, $P4_0$-$P4_7$, $P5_0$-$P5_7$, $P6_0$-$P6_7$, $P7_2$-$P7_7$, $P8_0$-$P8_4$, $P8_6$, $P8_7$, $P9_0$-$P9_7$, $P10_0$-$P10_7$, $P11_0$-$P11_4$, $P12_0$-$P12_7$, $P13_0$-$P13_7$, $P14_0$-$P14_6$, $P15_0$-$P15_7$(Note) | $V_I$=0V | 30 | 50 | 167 | kΩ |
| $Rf_{XIN}$ | Feedback resistance | $X_{IN}$ | | | 1.5 | | MW |
| $Rf_{XCIN}$ | Feedback resistance | $X_{CIN}$ | | | 10 | | MΩ |
| $V_{RAM}$ | RAM retention voltage | VDC-ON | | 2.5 | | | V |
| $I_{CC}$ | Power supply current | Measuring condition: In sigle-chip mode, the output pins are open and other pins are Vss. | f($X_{IN}$)=30MHz, square wave, no division | | 38 | 54 | mA |
| | | | f($X_{CIN}$)=32kHz, with WAIT instruction executed, Topr=25°C | | 470 | | µA |
| | | | When clock is stopped Topr=25°C | | 0.4 | 20 | µA |

Note: Port P11 to P15 exist in 144-pin version.

## VCC = 5V

**Table 1.32.4. A-D conversion characteristics (referenced to VCC = AVCC = VREF = 4.2 to 5.5V, Vss = AVSS = 0V at Topr = –20 to 85°C, f(XIN) = 30MHz unless otherwise specified)**

| Symbol | Parameter | Measuring condition | | Standard | | | Unit |
|---|---|---|---|---|---|---|---|
| | | | | Min. | Typ. | Max. | |
| – | Resolution | VREF = VCC | | | | 10 | Bits |
| INL | Integral nonlinearity error | VREF = VCC = 5V | AN0 to AN7 ANEX0, ANEX1 | | | ±3 | LSB |
| | | | External op-amp connection mode | | | ±7 | LSB |
| DNL | Differential nonlinearity error | | | | | ±1 | LSB |
| – | Offset error | | | | | ±3 | LSB |
| – | Gain error | | | | | ±3 | LSB |
| RLADDER | Ladder resistance | VREF = VCC | | 8 | | 40 | kW |
| tCONV | Conversion time(10bit) | | | 3.3 | | | ms |
| tCONV | Conversion time(8bit) | | | 2.8 | | | ms |
| tSAMP | Sampling time | | | 0.3 | | | ms |
| VREF | Reference voltage | | | 2 | | VCC | V |
| VIA | Analog input voltage | | | 0 | | VREF | V |

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

**Table 1.32.5. D-A conversion characteristics (referenced to VCC = VREF = 4.2 to 5.5V, VSS = AVSS = 0V at Topr = –20 to 85°C, f(XIN) = 30MHz unless otherwise specified)**

| Symbol | Parameter | Measuring condition | Standard | | | Unit |
|---|---|---|---|---|---|---|
| | | | Min. | Typ. | Max. | |
| – | Resolution | | | | 8 | Bits |
| – | Absolute accuracy | | | | 1.0 | % |
| tsu | Setup time | | | | 3 | ms |
| RO | Output resistance | | 4 | 10 | 20 | kW |
| IVREF | Reference power supply input current | (Note) | | | 1.5 | mA |

Note: This applies when using one D-A converter, with the D-A register for the unused D-A converter set to "00$_{16}$".
The A-D converter's ladder resistance is not included.
Also, when the Vref is unconnected at the A-D control register 1, IVREF is sent.

# VCC = 5V

**Timing requirements (referenced to VCC = 4.2 to 5.5V, VSS = 0V at Topr = –20 to 85°C unless otherwise specified)**

**Table 1.32.6. External clock input**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| tc | External clock input cycle time | 33 | | ns |
| tw(H) | External clock input HIGH pulse width | 13 | | ns |
| tw(L) | External clock input LOW pulse width | 13 | | ns |
| tr | External clock rise time | | 5 | ns |
| tf | External clock fall time | | 5 | ns |

**Table 1.32.7. Memory expansion and microprocessor modes**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| tac1(RD-DB) | Data input access time (RD standard, no wait) | | (Note) | ns |
| tac1(AD-DB) | Data input access time (AD standard, CS standard, no wait) | | (Note) | ns |
| tac2(RD-DB) | Data input access time (RD standard, with wait) | | (Note) | ns |
| tac2(AD-DB) | Data input access time (AD standard, CS standard, with wait) | | (Note) | ns |
| tac3(RD-DB) | Data input access time (RD standard, when accessing multiplex bus area) | | (Note) | ns |
| tac3(AD-DB) | Data input access time (AD standard, CS standard, when accessing multiplex bus area) | | (Note) | ns |
| tac4(RAS-DB) | Data input access time (RAS standard, DRAM access) | | (Note) | ns |
| tac4(CAS-DB) | Data input access time (CAS standard, DRAM access) | | (Note) | ns |
| tac4(CAD-DB) | Data input access time (CAD standard, DRAM access) | | (Note) | ns |
| tsu(DB-BCLK) | Data input setup time | 26 | | ns |
| tsu(RDY-BCLK ) | $\overline{\text{RDY}}$ input setup time | 26 | | ns |
| tsu(HOLD-BCLK ) | $\overline{\text{HOLD}}$ input setup time | 30 | | ns |
| th(RD-DB) | Data input hold time | 0 | | ns |
| th(CAS -DB) | Data input hold time | 0 | | ns |
| th(BCLK -RDY) | $\overline{\text{RDY}}$ input hold time | 0 | | ns |
| th(BCLK-HOLD ) | $\overline{\text{HOLD}}$ input hold time | 0 | | ns |
| td(BCLK-HLDA ) | $\overline{\text{HLDA}}$ output delay time | | 25 | ns |

Note: Calculated according to the BCLK frequency as follows:
Note that inserting wait or using lower operation frequency f(BCLK) is needed when calculated value is negative.

$$t_{ac1(RD-DB)} = \frac{10^9}{f(BCLK) \times 2} - 35 \quad [ns]$$

$$t_{ac1(AD-DB)} = \frac{10^9}{f(BCLK)} - 35 \quad [ns]$$

$$t_{ac2(RD-DB)} = \frac{10^9 \times m}{f(BCLK) \times 2} - 35 \quad [ns] \text{ (m=3, 5 and 7 when 1 wait, 2 wait and 3 wait, respectively)}$$

$$t_{ac2(AD-DB)} = \frac{10^9 \times n}{f(BCLK)} - 35 \quad [ns] \text{ (n=2, 3 and 4 when 1 wait, 2 wait and 3 wait, respectively)}$$

$$t_{ac3(RD-DB)} = \frac{10^9 \times m}{f(BCLK) \times 2} - 35 \quad [ns] \text{ (m=3 and 5 when 2 wait and 3 wait, respectively)}$$

$$t_{ac3(AD-DB)} = \frac{10^9 \times n}{f(BCLK) \times 2} - 35 \quad [ns] \text{ (n=5 and 7 when 2 wait and 3 wait, respectively)}$$

$$t_{ac4(RAS-DB)} = \frac{10^9 \times m}{f(BCLK) \times 2} - 35 \quad [ns] \text{ (m=3 and 5 when 1 wait and 2 wait, respectively)}$$

$$t_{ac4(CAS-DB)} = \frac{10^9 \times n}{f(BCLK) \times 2} - 35 \quad [ns] \text{ (n=1 and 3 when 1 wait and 2 wait, respectively)}$$

$$t_{ac4(CAD-DB)} = \frac{10^9 \times l}{f(BCLK)} - 35 \quad [ns] \text{ (l=1 and 2 when 1 wait and 2 wait, respectively)}$$

# VCC = 5V

**Timing requirements (referenced to VCC = 4.2 to 5.5V, VSS = 0V at Topr = –20 to 85°C unless otherwise specified)**

**Table 1.32.8. Timer A input (count input in event counter mode)**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TA)}$ | TAiIN input cycle time | 100 | | ns |
| $t_{w(TAH)}$ | TAiIN input HIGH pulse width | 40 | | ns |
| $t_{w(TAL)}$ | TAiIN input LOW pulse width | 40 | | ns |

**Table 1.32.9. Timer A input (gating input in timer mode)**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TA)}$ | TAiIN input cycle time | 400 | | ns |
| $t_{w(TAH)}$ | TAiIN input HIGH pulse width | 200 | | ns |
| $t_{w(TAL)}$ | TAiIN input LOW pulse width | 200 | | ns |

**Table 1.32.10. Timer A input (external trigger input in one-shot timer mode)**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TA)}$ | TAiIN input cycle time | 200 | | ns |
| $t_{w(TAH)}$ | TAiIN input HIGH pulse width | 100 | | ns |
| $t_{w(TAL)}$ | TAiIN input LOW pulse width | 100 | | ns |

**Table 1.32.11. Timer A input (external trigger input in pulse width modulation mode)**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{w(TAH)}$ | TAiIN input HIGH pulse width | 100 | | ns |
| $t_{w(TAL)}$ | TAiIN input LOW pulse width | 100 | | ns |

**Table 1.32.12. Timer A input (up/down input in event counter mode)**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(UP)}$ | TAiOUT input cycle time | 2000 | | ns |
| $t_{w(UPH)}$ | TAiOUT input HIGH pulse width | 1000 | | ns |
| $t_{w(UPL)}$ | TAiOUT input LOW pulse width | 1000 | | ns |
| $t_{su(UP-TIN)}$ | TAiOUT input setup time | 400 | | ns |
| $t_{h(TIN-UP)}$ | TAiOUT input hold time | 400 | | ns |

# VCC = 5V

**Timing requirements (referenced to VCC = 4.2 to 5.5V, VSS = 0V at Topr = –20 to 85°C unless otherwise specified)**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TB)}$ | TBiIN input cycle time (counted on one edge) | 100 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width (counted on one edge) | 40 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width (counted on one edge) | 40 | | ns |
| $t_{c(TB)}$ | TBiIN input cycle time (counted on both edges) | 200 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width (counted on both edges) | 80 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width (counted on both edges) | 80 | | ns |

**Table 1.32.13. Timer B input (count input in event counter mode)**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TB)}$ | TBiIN input cycle time | 400 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width | 200 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width | 200 | | ns |

**Table 1.28.14. Timer B input (pulse period measurement mode)**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TB)}$ | TBiIN input cycle time | 400 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width | 200 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width | 200 | | ns |

**Table 1.32.15. Timer B input (pulse width measurement mode)**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(AD)}$ | $\overline{ADTRG}$ input cycle time (trigger able minimum) | 1000 | | ns |
| $t_{w(ADL)}$ | $\overline{ADTRG}$ input LOW pulse width | 125 | | ns |

**Table 1.32.16. A-D trigger input**

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

**Table 1.32.17. Serial I/O**

**Table 1.32.18. External interrupt $\overline{INTi}$ inputs**

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

# VCC = 5V

**Switching characteristics (referenced to VCC = 4.2 to 5.5V, VSS = 0V at Topr = –20 to 85°C unless otherwise specified)**

**Table 1.32.19. Memory expansion mode and microprocessor mode (no wait)**

| Symbol | Parameter | Measuring condition | Standard | | Unit |
|---|---|---|---|---|---|
| | | | Min. | Max. | |
| td(BCLK-AD) | Address output delay time | Figure 1.32.1 | | 18 | ns |
| th(BCLK-AD) | Address output hold time (BCLK standard) | | -3 | | ns |
| th(RD-AD) | Address output hold time (RD standard) | | 0 | | ns |
| th(WR-AD) | Address output hold time (WR standard) | | (Note) | | ns |
| td(BCLK-CS) | Chip select output delay time | | | 18 | ns |
| th(BCLK-CS) | Chip select output hold time (BCLK standard) | | -3 | | ns |
| th(RD-CS) | Chip select output hold time (RD standard) | | 0 | | ns |
| th(WR-CS) | Chip select output hold time (WR standard) | | (Note) | | ns |
| td(BCLK-ALE) | ALE signal output delay time | | | 18 | ns |
| th(BCLK-ALE) | ALE signal output hold time | | – 2 | | ns |
| td(BCLK-RD) | RD signal output delay time | | | 18 | ns |
| th(BCLK-RD) | RD signal output hold time | | -5 | | ns |
| td(BCLK-WR) | WR signal output delay time | | | 18 | ns |
| th(BCLK-WR) | WR signal output hold time | | -3 | | ns |
| td(DB-WR) | Data output delay time (WR standard) | | (Note) | | ns |
| th(WR-DB) | Data output hold time (WR standard) | | (Note) | | ns |
| tw(WR) | WR signal width | | (Note) | | ns |

Note: Calculated according to the BCLK frequency as follows:

$$t_{d(DB-WR)} = \frac{10^9}{f_{(BCLK)}} - 20 \quad [ns]$$

$$t_{h(WR-DB)} = \frac{10^9}{f_{(BCLK)} \times 2} - 10 \quad [ns]$$

$$t_{h(WR-AD)} = \frac{10^9}{f_{(BCLK)} \times 2} - 10 \quad [ns]$$

$$t_{h(WR-CS)} = \frac{10^9}{f_{(BCLK)} \times 2} - 10 \quad [ns]$$

$$t_{w(WR)} = \frac{10^9}{f_{(BCLK)} \times 2} - 15 \quad [ns]$$

# VCC = 5V

**Switching characteristics (referenced to VCC = 4.2 to 5.5V, VSS = 0V at Topr = –20 to 85°C unless otherwise specified)**

**Table 1.32.20. Memory expansion mode and microprocessor mode (with wait, accessing external memory)**

| Symbol | Parameter | Measuring condition | Standard | | Unit |
|---|---|---|---|---|---|
| | | | Min. | Max. | |
| td(BCLK-AD) | Address output delay time | Figure 1.32.1 | | 18 | ns |
| th(BCLK-AD) | Address output hold time (BCLK standard) | | – 3 | | ns |
| th(RD-AD) | Address output hold time (RD standard) | | 0 | | ns |
| th(WR-AD) | Address output hold time (WR standard) | | (Note) | | ns |
| td(BCLK-CS) | Chip select output delay time | | | 18 | ns |
| th(BCLK-CS) | Chip select output hold time (BCLK standard) | | – 3 | | ns |
| th(RD-CS) | Chip select output hold time (RD standard) | | 0 | | ns |
| th(WR-CS) | Chip select output hold time (WR standard) | | (Note) | | ns |
| td(BCLK-ALE) | ALE signal output delay time | | | 18 | ns |
| th(BCLK-ALE) | ALE signal output hold time | | – 2 | | ns |
| td(BCLK-RD) | RD signal output delay time | | | 18 | ns |
| th(BCLK-RD) | RD signal output hold time | | – 5 | | ns |
| td(BCLK-WR) | WR signal output delay time | | | 18 | ns |
| th(BCLK-WR) | WR signal output hold time | | – 3 | | ns |
| td(DB-WR) | Data output delay time (WR standard) | | (Note) | | ns |
| th(WR-DB) | Data output hold time (WR standard) | | (Note) | | ns |
| tw(WR) | WR signal width | | (Note) | | ns |

Note: Calculated according to the BCLK frequency as follows:

$$t_{d(DB-WR)} = \frac{10^9 \times n}{f_{(BCLK)}} - 20 \quad [ns] \quad (n=1, 2 \text{ and } 3 \text{ when 1 wait, 2 wait and 3 wait, respectively})$$

$$t_{h(WR-DB)} = \frac{10^9}{f_{(BCLK)} \times 2} - 10 \quad [ns]$$

$$t_{h(WR-AD)} = \frac{10^9}{f_{(BCLK)} \times 2} - 10 \quad [ns]$$

$$t_{h(WR-CS)} = \frac{10^9}{f_{(BCLK)} \times 2} - 10 \quad [ns]$$

$$t_{w(WR)} = \frac{10^9 \times n}{f_{(BCLK)} \times 2} - 15 \quad [ns] \quad (n=1, 3 \text{ and } 5 \text{ when 1 wait, 2 wait and 3 wait, respectively})$$

# VCC = 5V

**Switching characteristics (referenced to VCC = 4.2 to 5.5V, VSS = 0V at Topr = –20 to 85°C unless otherwise specified)**

**Table 1.32.21. Memory expansion mode and microprocessor mode (with wait, accessing external memory, multiplex bus area selected)**

| Symbol | Parameter | Measuring condition | Standard | | Unit |
|---|---|---|---|---|---|
| | | | Min. | Max. | |
| td(BCLK-AD) | Address output delay time | Figure 1.32.1 | | 18 | ns |
| th(BCLK-AD) | Address output hold time (BCLK standard) | | -3 | | ns |
| th(RD-AD) | Address output hold time (RD standard) | | (Note) | | ns |
| th(WR-AD) | Address output hold time (WR standard) | | (Note) | | ns |
| td(BCLK-CS) | Chip select output delay time | | | 18 | ns |
| th(BCLK-CS) | Chip select output hold time (BCLK standard) | | -3 | | ns |
| th(RD-CS) | Chip select output hold time (RD standard) | | (Note) | | ns |
| th(WR-CS) | Chip select output hold time (WR standard) | | (Note) | | ns |
| td(BCLK-RD) | RD signal output delay time | | | 18 | ns |
| th(BCLK-RD) | RD signal output hold time | | -5 | | ns |
| td(BCLK-WR) | WR signal output delay time | | | 18 | ns |
| th(BCLK-WR) | WR signal output hold time | | -3 | | ns |
| td(DB-WR) | Data output delay time (WR standard) | | (Note) | | ns |
| th(WR-DB) | Data output hold time (WR standard) | | (Note) | | ns |
| td(BCLK-ALE) | ALE signal output delay time (BCLK standard) | | | 18 | ns |
| th(BCLK-ALE) | ALE signal output hold time (BCLK standard) | | – 2 | | ns |
| td(AD-ALE) | ALE signal output delay time (address standard) | | (Note) | | ns |
| th(ALE-AD) | ALE signal output hold time (address standard) | | (Note) | | ns |
| tdz(RD-AD) | Address output flowting start time | | | 8 | ns |

Note: Calculated according to the BCLK frequency as follows:

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

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

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

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

$$td(DB-WR) = \frac{10^9 \times m}{f(BCLK) \times 2} - 25 \quad [ns] \quad (m=3 \text{ and } 5 \text{ when 2 wait and 3 wait, respectively})$$

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

$$td(AD-ALE) = \frac{10^9}{f(BCLK) \times 2} - 20 \quad [ns]$$

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

# VCC = 5V

**Switching characteristics (referenced to VCC = 4.2 to 5.5V, VSS = 0V at Topr = –20 to 85°C unless otherwise specified)**

**Table 1.32.22. Memory expansion mode and microprocessor mode (with wait, accessing external memory, DRAM area selected)**

| Symbol | Parameter | Measuring condition | Standard | | Unit |
|---|---|---|---|---|---|
| | | | Min. | Max. | |
| $t_{d(BCLK-RAD)}$ | Row address output delay time | Figure 1.32.1 | | 18 | ns |
| $t_{h(BCLK-RAD)}$ | Row address output hold time (BCLK standard) | | -3 | | ns |
| $t_{d(BCLK-CAD)}$ | String address output delay time | | | 18 | ns |
| $t_{h(BCLK-CAD)}$ | String address output hold time (BCLK standard) | | -3 | | ns |
| $t_{h(RAS-RAD)}$ | Row address output hold time after RAS output | | (Note) | | ns |
| $t_{d(BCLK-RAS)}$ | RAS output delay time (BCLK standard) | | | 18 | ns |
| $t_{h(BCLK-RAS)}$ | RAS output hold time (BCLK standard) | | -3 | | ns |
| $t_{RP}$ | RAS "H" hold time | | (Note) | | ns |
| $t_{d(BCLK-CAS)}$ | CAS output delay time (BCLK standard) | | | 18 | ns |
| $t_{h(BCLK-CAS)}$ | CAS output hold time (BCLK standard) | | -3 | | ns |
| $t_{d(BCLK-DW)}$ | DW output delay time (BCLK standard) | | | 18 | ns |
| $t_{h(BCLK-DW)}$ | DW output hold time (BCLK standard) | | -5 | | ns |
| $t_{su(DB-CAS)}$ | CAS output setup time after DB output | | (Note) | | ns |
| $t_{h(BCLK-DB)}$ | DB signal output hold time (BCLK standard) | | -7 | | ns |
| $t_{su(CAS-RAS)}$ | CAS output setup time before RAS output (refresh) | | (Note) | | ns |

Note: Calculated according to the BCLK frequency as follows:

$$t_{h(RAS-RAD)} = \frac{10^9}{f_{(BCLK)} \times 2} - 13 \quad [ns]$$

$$t_{RP} = \frac{10^9}{f_{(BCLK)} \times 2} \times 3 - 20 \quad [ns]$$

$$t_{su(DB-CAS)} = \frac{10^9}{f_{(BCLK)}} - 20 \quad [ns]$$

$$t_{su(CAS-RAS)} = \frac{10^9}{f_{(BCLK)} \times 2} - 13 \quad [ns]$$

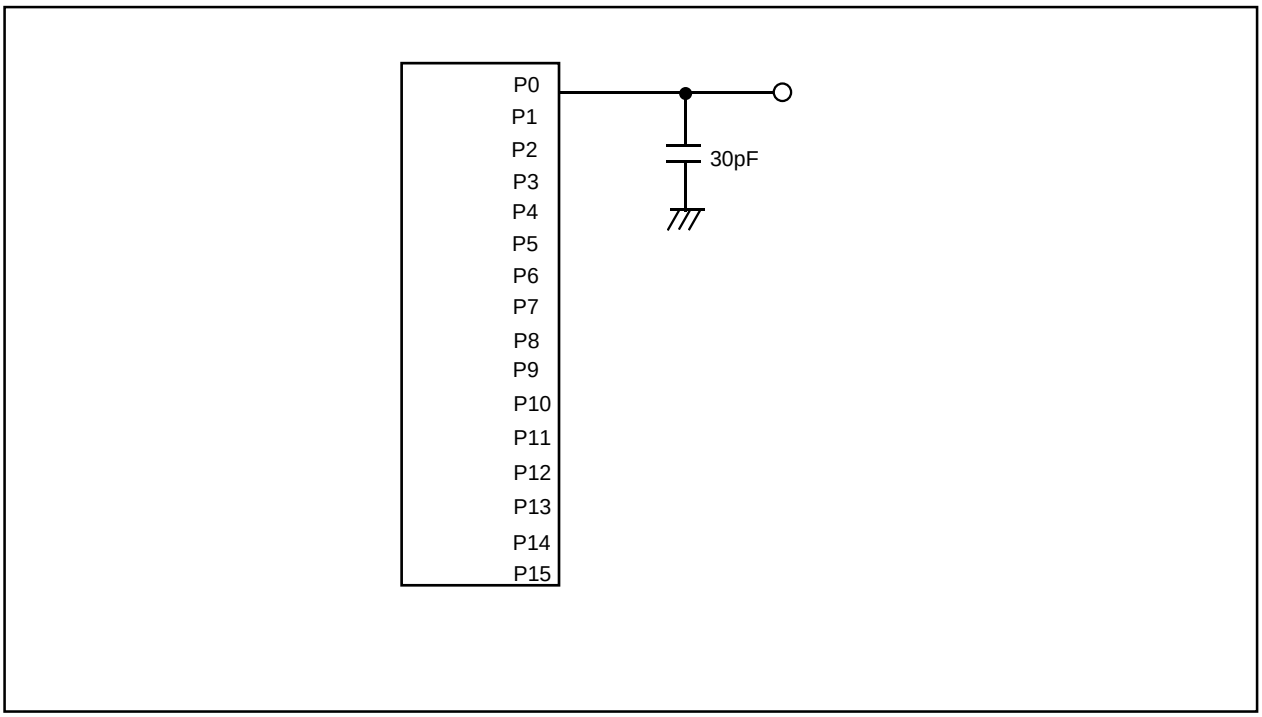
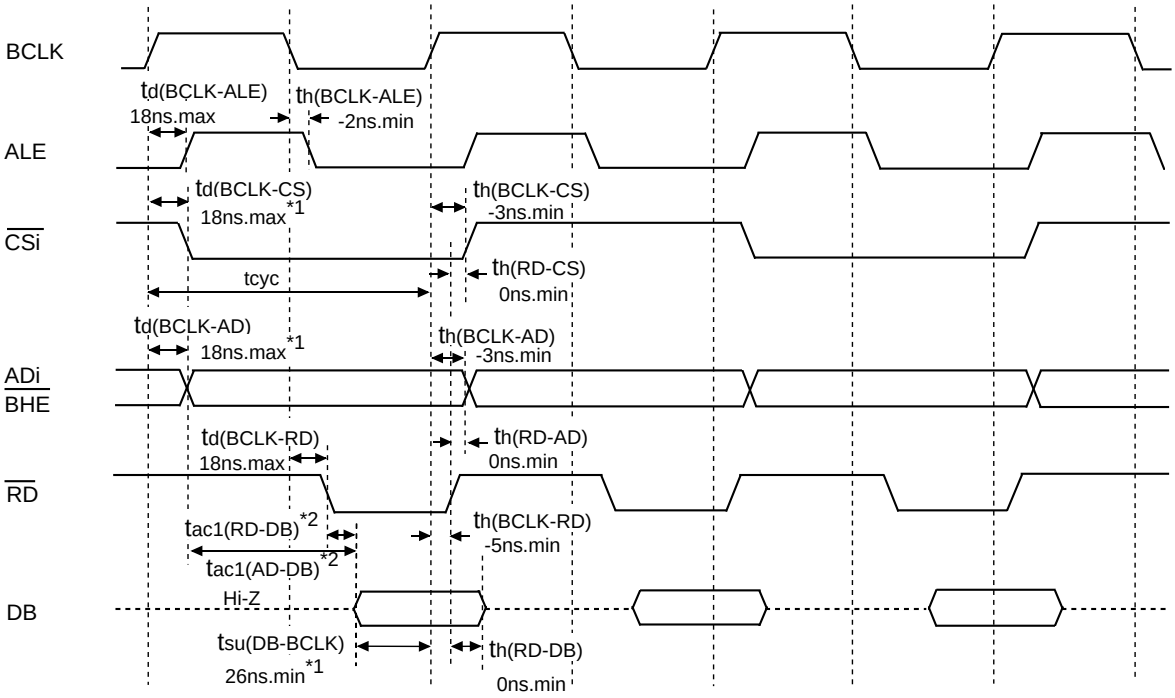



**Figure 1.32.1. Port P0 to P15 measurement circuit**

Vcc=5V

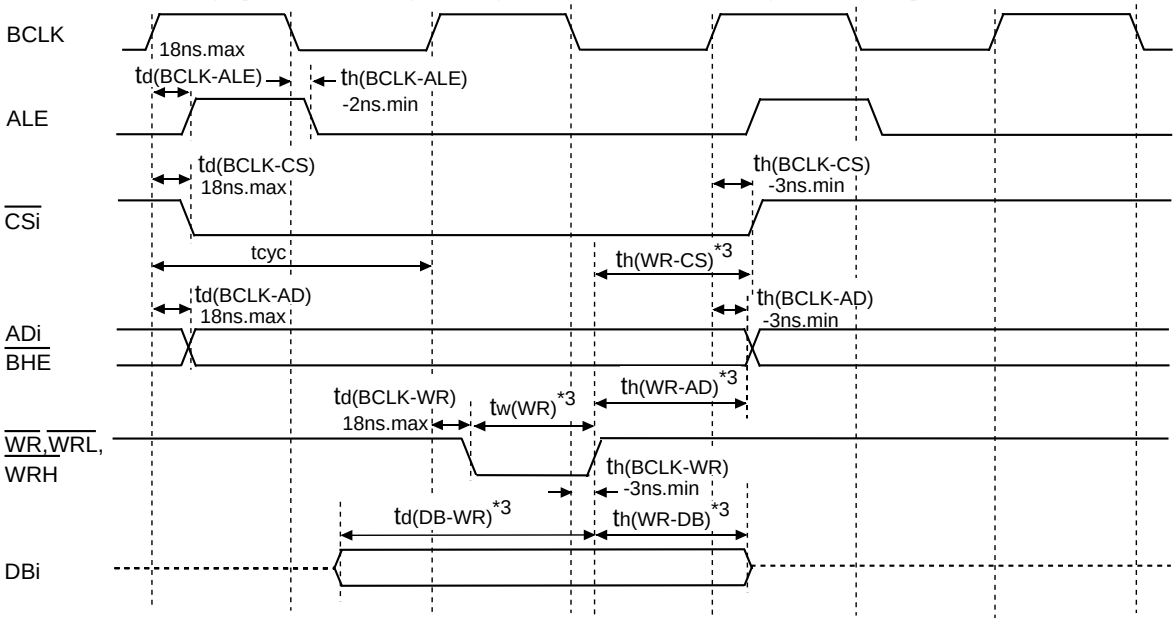
## Memory Expansion Mode and Microprocessor Mode (without wait)

### Read Timing




*1:It is a guarantee value with being alone. 35ns.max garantees as $t_{d(BCLK-AD)}+t_{su(DB-BCLK)}$.
*2:It depends on operation frequency.
$t_{ac1(RD-DB)}$=(tcyc/2-35)ns.max
$t_{ac1(AD-DB)}$=(tcyc-35)ns.max

### Write Timing ( Written by 2 cycles in selecting no wait)




*3:It depends on operation frequency.
$t_{d(DB-WR)}$=(tcyc-20)ns.min
$t_{h(WR-DB)}$=(tcyc/2-10)ns.min
$t_{h(WR-AD)}$=(tcyc/2-10)ns.min
$t_{h(WR-CS)}$=(tcyc/2-10)ns.min
$t_{w(WR)}$=(tcyc/2-15)ns.min

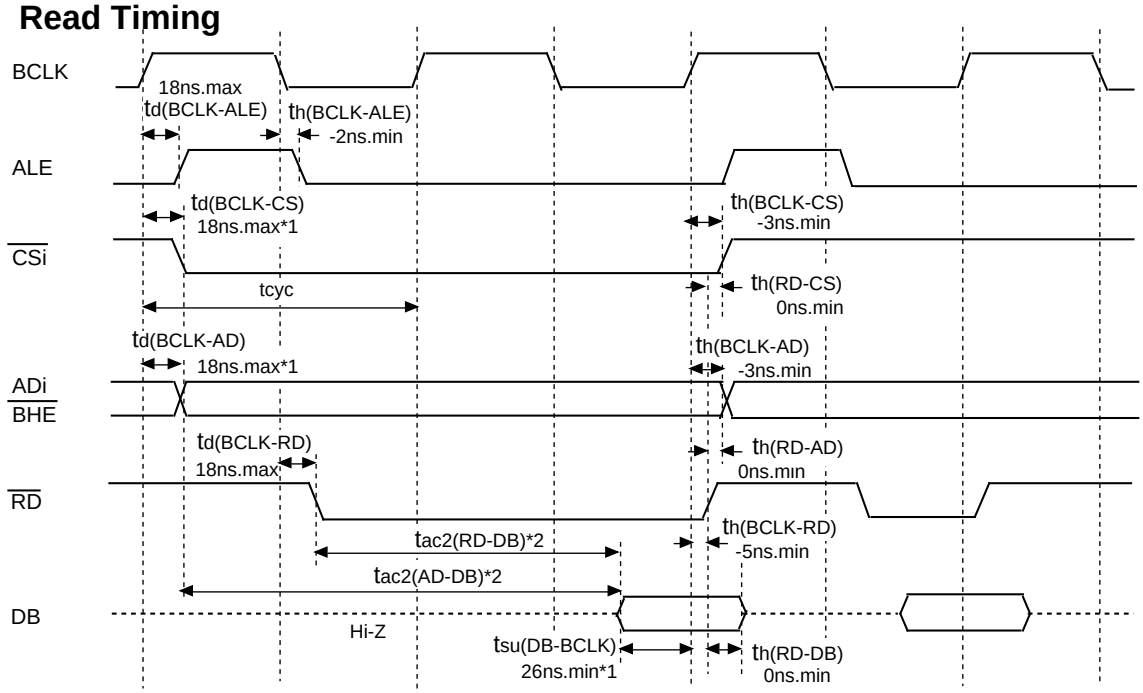
Measuring conditions
- $V_{CC}$=4.2V to 5.5V
- Input timing voltage :Determined with $V_{IH}$=2.5V, $V_{IL}$=0.8V
- Output timing voltage :Determined with $V_{OH}$=2.0V, $V_{OL}$=0.8V

**Figure 1.32.2. Vcc=5V timing diagram (1)**

Vcc=5V

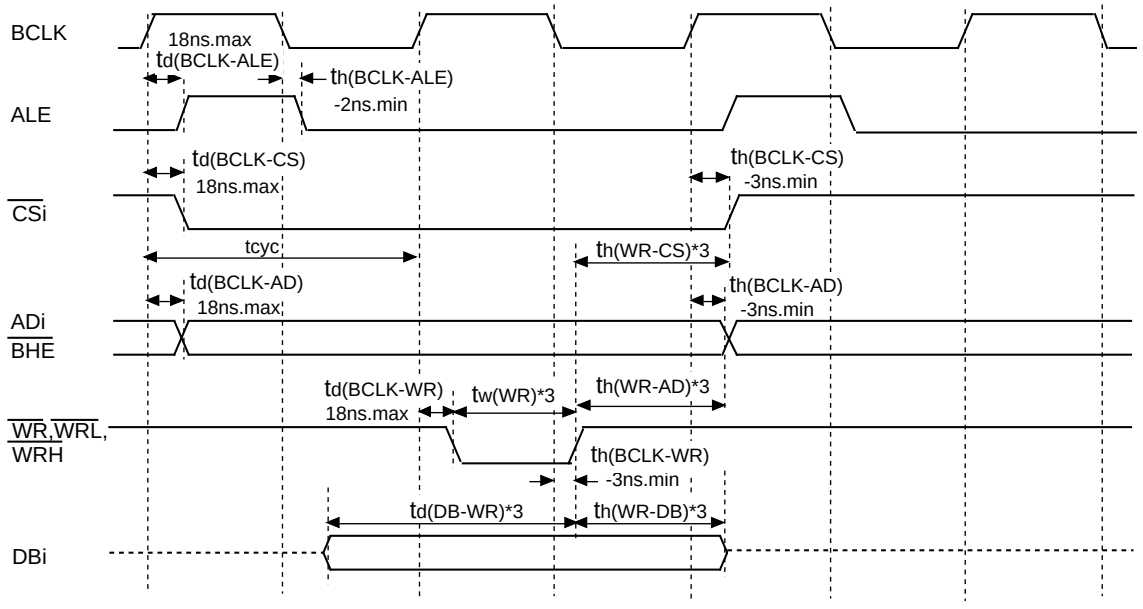
## Memory Expansion Mode and Microprocessor Mode (with wait)

### Read Timing




*1:It is a guarantee value with being alone. 35ns.max garantees as $t_{d(BCLK-AD)}+t_{su(DB-BCLK)}$.
*2:It depends on operation frequency.
$t_{ac2(RD-DB)}$=(tcyc/2 x m-35)ns.max (m=3, 5 and 7 when 1 wait, 2 wait and 3 wait, respectively.)
$t_{ac2(AD-DB)}$=(tcyc x n-35)ns.max (n=2, 3 and 4 when 1 wait, 2 wait and 3 wait, respectively.)

### Write Timing ( Written by 2 cycles in selecting no wait)




*3:It depends on operation frequency.
$t_{d(DB-WR)}$=(tcyc x n-20)ns.min
(n=1, 2 and 3 when 1 wait, 2 wait and 3 wait, respectively.)
$t_{h(WR-DB)}$=(tcyc/2-10)ns.min
$t_{h(WR-AD)}$=(tcyc/2-10)ns.min
$t_{h(WR-CS)}$=(tcyc/2-10)ns.min
$t_{w(WR)}$=(tcyc/2 x n-15)ns.min
(n=1, 3 and 5 when 1 wait, 2 wait and 3 wait, respectively.)

Measuring conditions
- $V_{CC}$=4.2V to 5.5V
- Input timing voltage
  :Determined with $V_{IH}$=2.5V, $V_{IL}$=0.8V
- Output timing voltage
  :Determined with $V_{OH}$=2.0V, $V_{OL}$=0.8V

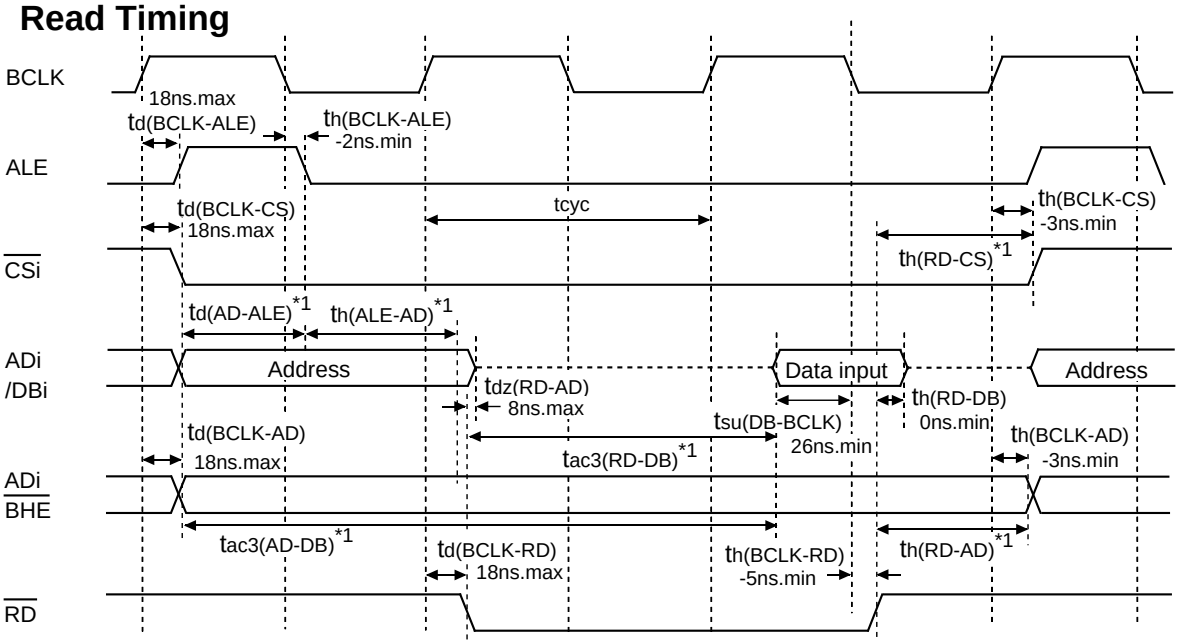
**Figure 1.32.3. Vcc=5V timing diagram (2)**

## Memory Expansion Mode and Microprocessor Mode

Vcc=5V

**(When accessing external memory area with wait, and select multiplexed bus))**

### Read Timing




*1:It depends on operation frequency.

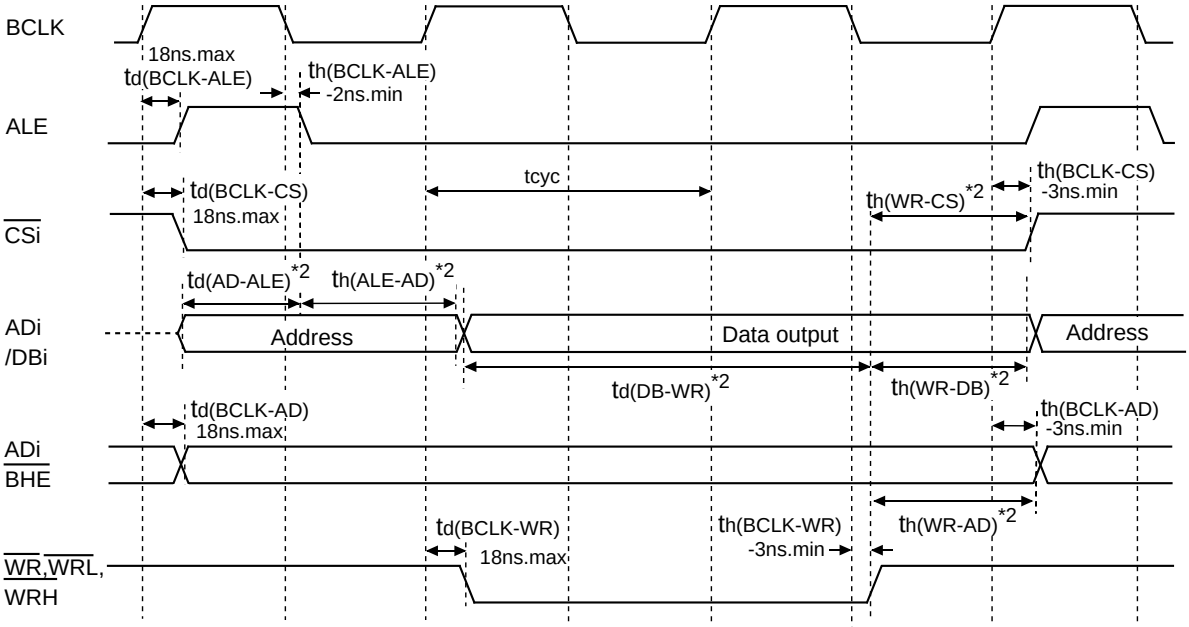
$t_{d(AD-ALE)}$=(tcyc/2-20)ns.min

$t_{h(ALE-AD)}$=(tcyc/2-10)ns.min, $t_{h(RD-AD)}$=(tcyc/2-10)ns.min, $t_{h(RD-CS)}$=(tcyc/2-10)ns.min

$t_{ac3(RD-DB)}$=(tcyc/2 x m-35)ns.max (m=3 and 5 when 2 wait and 3 wait, respectively.)

$t_{ac3(AD-DB)}$=(tcyc/2 x n-35)ns.max (n=5 and 7 when 2 wait and 3 wait, respectively.)

### Write Timing (Written by 2 cycles in selecting no wait)




*2:It depends on operation frequency.

$t_{d(AD-ALE)}$=(tcyc/2-20)ns.min

$t_{h(ALE-AD)}$=(tcyc/2-10)ns.min, $t_{h(WR-AD)}$=(tcyc/2-10)ns.min

$t_{h(WR-CS)}$=(tcyc/2-10)ns.min, $t_{h(WR-DB)}$=(tcyc/2-10)ns.min

$t_{d(DB-WR)}$=(tcyc/2 x m-25)ns.min

(m=3 and 5 when 2 wait and 3 wait, respectively.)

Measuring conditions

- $V_{CC}$=4.2V to 5.5V
- Input timing voltage
  :Determined with $V_{IH}$=2.5V, $V_{IL}$=0.8V
- Output timing voltage
  :Determined with $V_{OH}$=2.0V, $V_{OL}$=0.8V

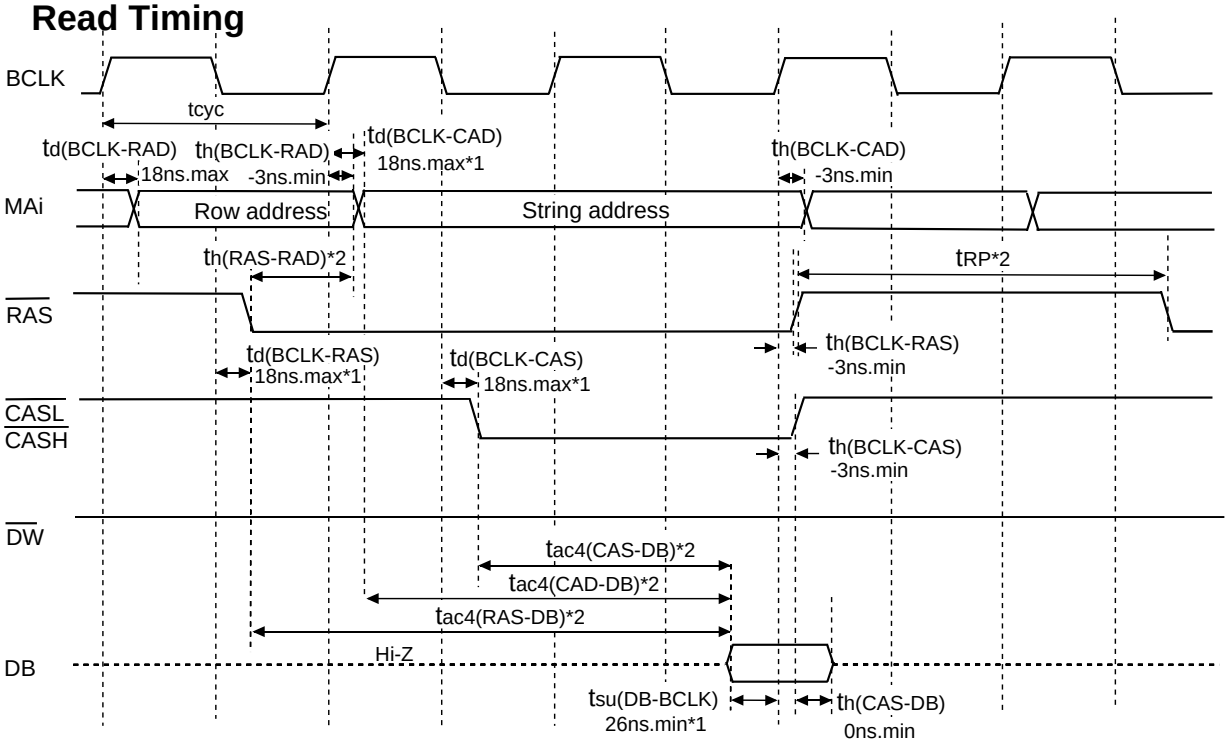
**Figure 1.32.4. Vcc=5V timing diagram (3)**

## Memory Expansion Mode and Microprocessor Mode

Vcc=5V

**(When accessing DRAM area)**

### Read Timing




*1:It is a guarantee value with being alone. 35ns.max garantees as follows:

$t_{d(BCLK-RAS)} + t_{su(DB-BCLK)}$

$t_{d(BCLK-CAS)} + t_{su(DB-BCLK)}$

$t_{d(BCLK-CAD)} + t_{su(DB-BCLK)}$

*2:It depends on operation frequency.

$t_{ac4(RAS-DB)}$=(tcyc/2 x m-35)ns.max (m=3 and 5 when 1 wait and 2 wait, respectively.)

$t_{ac4(CAS-DB)}$=(tcyc/2 x n-35)ns.max (n=1 and 3 when 1 wait and 2 wait, respectively.)

$t_{ac4(CAD-DB)}$=(tcyc x l-35)ns.max (l=1 and 2 when 1 wait and 2 wait, respectively.)

$t_{h(RAS-RAD)}$=(tcyc/2-13)ns.min

$t_{RP}$=(tcyc/2 x 3-20)ns.min

Measuring conditions

- $V_{CC}$=4.2V to 5.5V
- Input timing voltage
  :Determined with $V_{IH}$=2.5V, $V_{IL}$=0.8V
- Output timing voltage
  :Determined with $V_{OH}$=2.0V, $V_{OL}$=0.8V

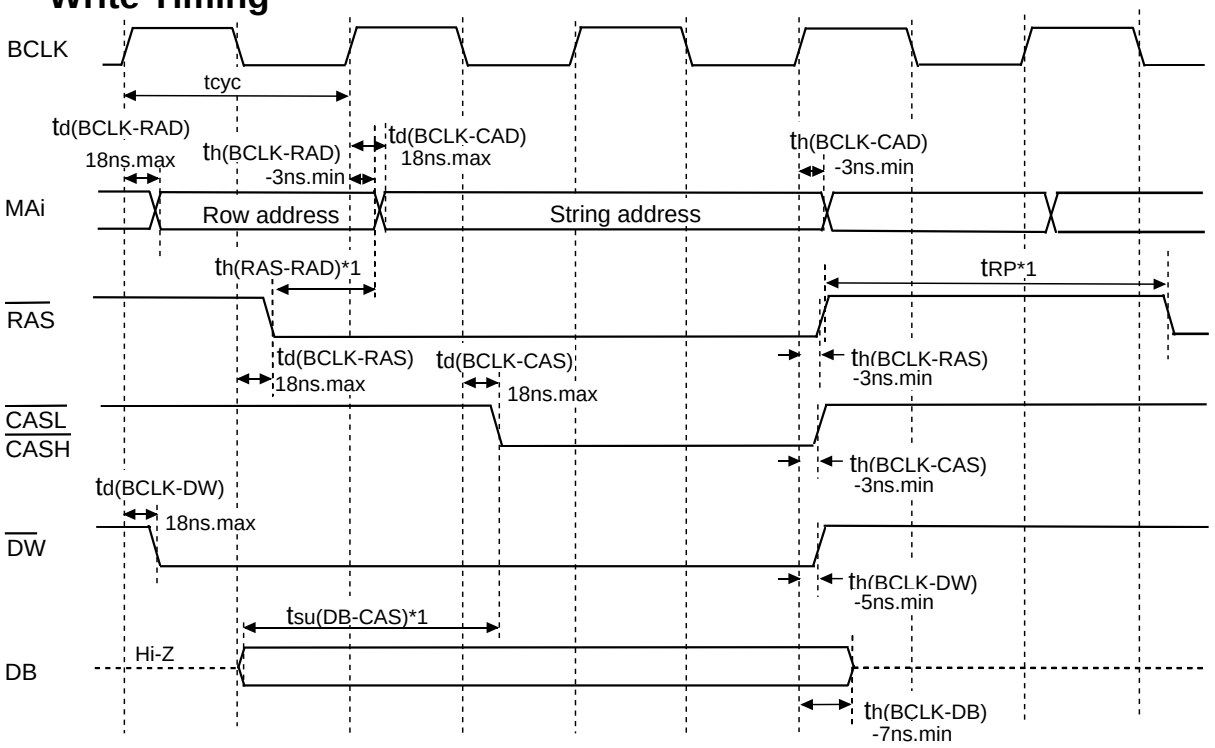
**Figure 1.32.5. Vcc=5V timing diagram (4)**

## Memory Expansion Mode and Microprocessor Mode

Vcc=5V

**(When accessing DRAM area)**

### Write Timing




*1:It depends on operation frequency.

$t_{h(RAS-RAD)}$=(tcyc/2-13)ns.min

$t_{RP}$=(tcyc/2 x 3-20)ns.min

$t_{su(DB-CAS)}$=(tcyc-20)ns.min

Measuring conditions

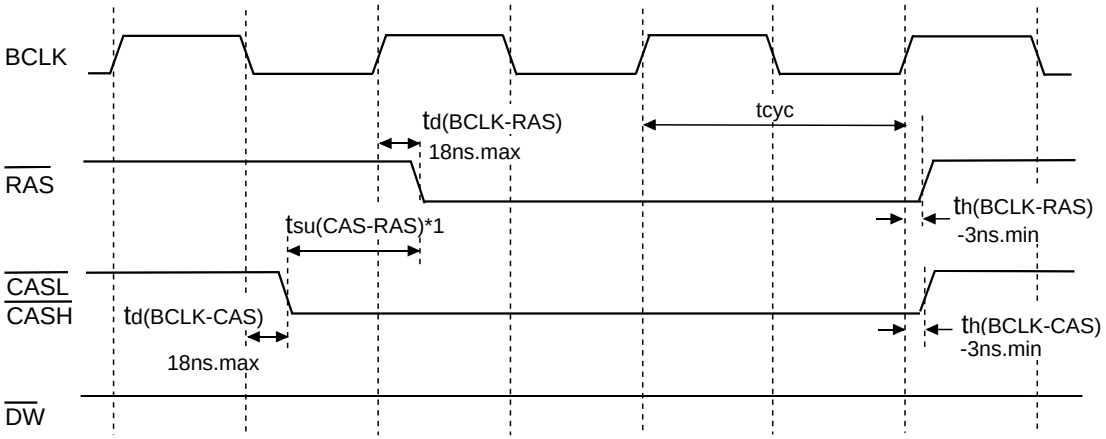
- $V_{CC}$=4.2V to 5.5V
- Input timing voltage
  :Determined with $V_{IH}$=2.5V, $V_{IL}$=0.8V
- Output timing voltage
  :Determined with $V_{OH}$=2.0V, $V_{OL}$=0.8V

**Figure 1.32.6. Vcc=5V timing diagram (5)**

## Memory expansion Mode and Microprocessor Mode

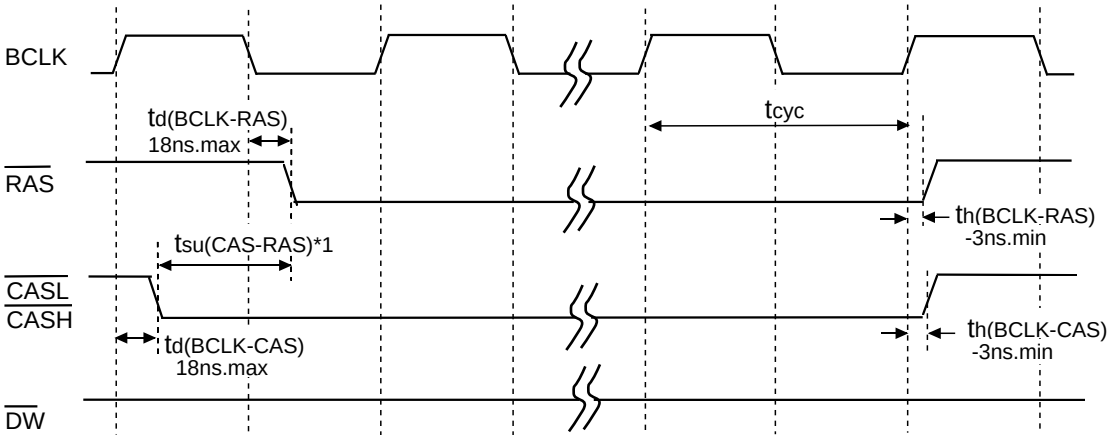
Vcc=5V

### Refresh Timing (CAS before RAS refresh)




*1:It depends on operation frequency.
$t_{su(CAS-RAS)}$=(tcyc/2-13)ns.min

### Refresh Timing (Self-refresh)




*1:It depends on operation frequency.
$t_{su(CAS-RAS)}$=(tcyc/2-13)ns.min

Measuring conditions
- $V_{CC}$=4.2V to 5.5V
- Input timing voltage
  :Determined with $V_{IH}$=2.5V, $V_{IL}$=0.8V
- Output timing voltage
  :Determined with $V_{OH}$=2.0V, $V_{OL}$=0.8V

**Figure 1.32.7. Vcc=5V timing diagram (6)**

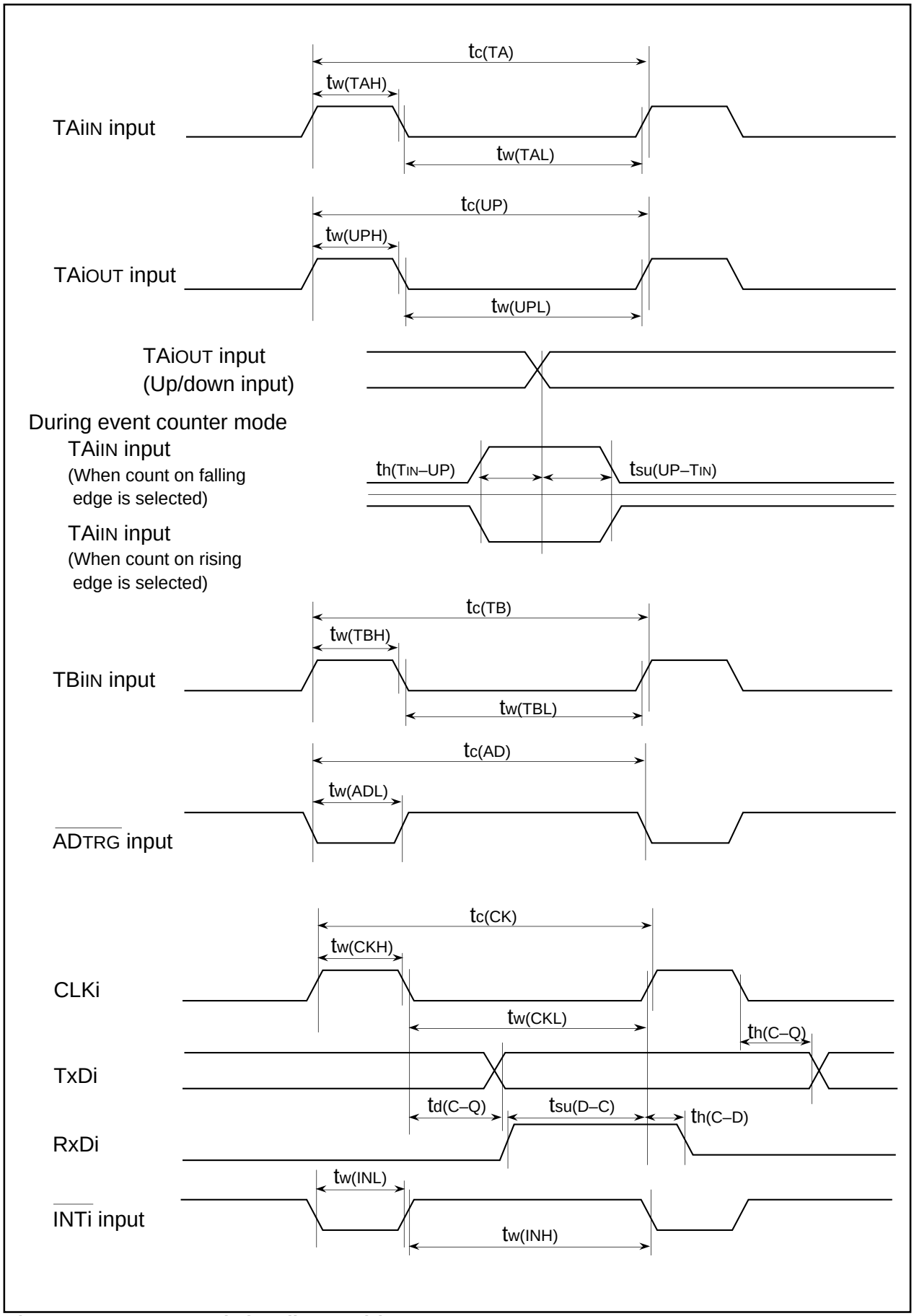
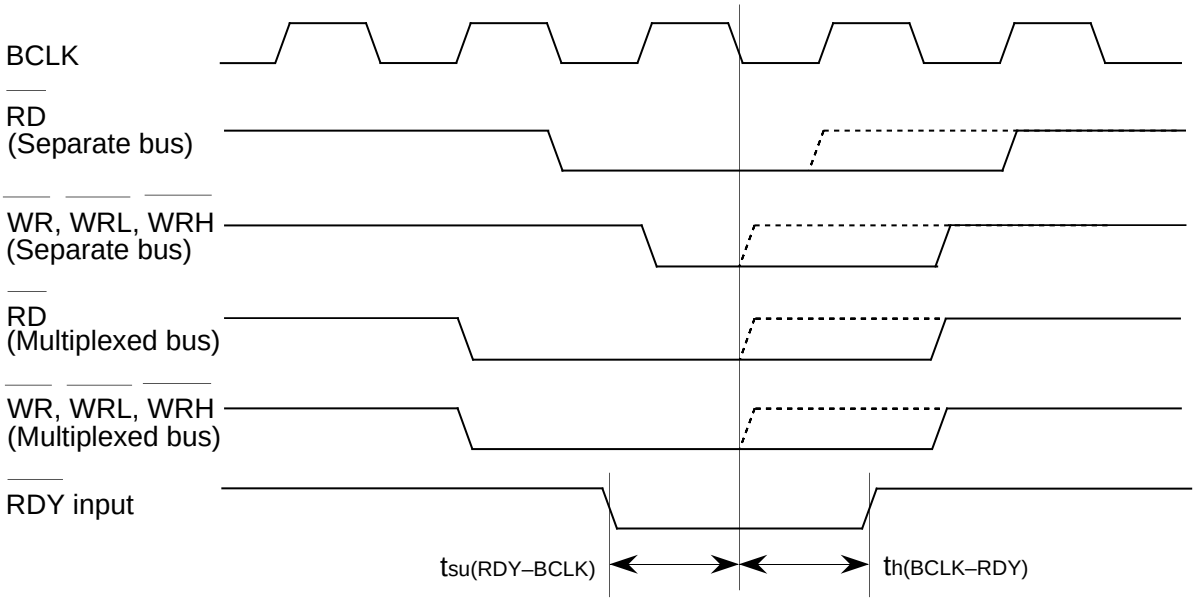


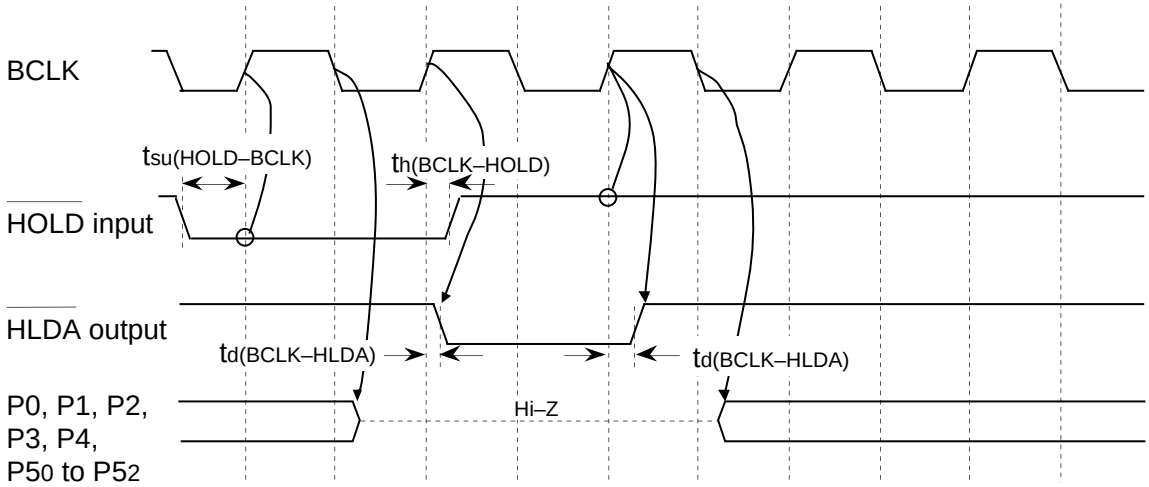

**Figure 1.32.8. Vcc=5V timing diagram (7)**

## Memory Expansion Mode and Microprocessor Mode

**(Valid only with wait)**




**(Valid with or without wait)**




Note: Regardless of the level of the BYTE pin input and the setting of the port P40 to P43 function select bit (PM06) of the processor mode register 0, all ports above become the high-impedance state.

Measuring conditions :
- VCC=4.2V to 5.5V
- Input timing voltage : Determined with VIH=4.0V, VIL=1.0V
- Output timing voltage : Determined with VOH=2.5V, VOL=2.5V

**Figure 1.32.9. Vcc=5V timing diagram (8)**

## Electrical characteristics (Vcc = 3.3V)

**Table 1.32.23. Electrical characteristics (referenced to $V_{CC}$=3.0 to 3.6V, $V_{SS}$=0V at Topr = –20 to 85°C, f($X_{IN}$)=20MHz unless otherwise specified)**

$V_{CC}$ = 3.3V

| Symbol | Parameter | | Condition | Standard | | | Unit |
|---|---|---|---|---|---|---|---|
| | | | | Min. | Typ. | Max. | |
| $V_{OH}$ | "H" output voltage | $P0_0$-$P0_7$, $P1_0$-$P1_7$, $P2_0$-$P2_7$, $P3_0$-$P3_7$, $P4_0$-$P4_7$, $P5_0$-$P5_7$, $P6_0$-$P6_7$, $P7_0$-$P7_7$, $P8_0$-$P8_4$, $P8_6$, $P8_7$, $P9_0$-$P9_7$, $P10_0$-$P10_7$, $P11_0$-$P11_4$, $P12_0$-$P12_7$, $P13_0$-$P13_7$, $P14_0$-$P14_6$, $P15_0$-$P15_7$(Note) | $I_{OH}$=-1mA | 2.7 | | | V |
| $V_{OH}$ | "H" output voltage | $X_{OUT}$ | $I_{OH}$=-0.1mA | 2.7 | | | V |
| | "H" output voltage | $X_{COUT}$ | No load applied | | 3.3 | | V |
| $V_{OL}$ | "L" output voltage | $P0_0$-$P0_7$, $P1_0$-$P1_7$, $P2_0$-$P2_7$, $P3_0$-$P3_7$, $P4_0$-$P4_7$, $P5_0$-$P5_7$, $P6_0$-$P6_7$, $P7_0$-$P7_7$, $P8_0$-$P8_4$, $P8_6$, $P8_7$, $P9_0$-$P9_7$, $P10_0$-$P10_7$, $P11_0$-$P11_4$, $P12_0$-$P12_7$, $P13_0$-$P13_7$, $P14_0$-$P14_6$, $P15_0$-$P15_7$(Note) | $I_{OL}$=1mA | | | 0.5 | |
| $V_{OL}$ | "L" output voltage | $X_{OUT}$ | $I_{OL}$=0.1mA | | | 0.5 | V |
| | "L" output voltage | $X_{COUT}$ | No load applied | | 0 | | V |
| $V_{T+}$-$V_{T-}$ | Hysteresis | $\overline{HOLD}$, $\overline{RDY}$, $TA0_{IN}$-$TA4_{IN}$, $TB0_{IN}$-$TB5_{IN}$, $\overline{INT0}$-$\overline{INT5}$, $AD_{TRG}$, $\overline{CTS0}$-$\overline{CTS4}$, CLK0-CLK4, $TA0_{OUT}$-$TA4_{OUT}$, $\overline{NMI}$, $\overline{KI0}$-$\overline{KI3}$, RxD0-RxD4, SCL0-SCL4, SDA0-SDA4 | | 0.2 | | 1.0 | V |
| $V_{T+}$-$V_{T-}$ | Hysteresis | $\overline{RESET}$ | | 0.2 | | 1.8 | V |
| $I_{IH}$ | "H" input current | $P0_0$-$P0_7$, $P1_0$-$P1_7$, $P2_0$-$P2_7$, $P3_0$-$P3_7$, $P4_0$-$P4_7$, $P5_0$-$P5_7$, $P6_0$-$P6_7$, $P7_2$-$P7_7$, $P8_0$-$P8_7$, $P9_0$-$P9_7$, $P10_0$-$P10_7$, $P11_0$-$P11_4$, $P12_0$-$P12_7$, $P13_0$-$P13_7$, $P14_0$-$P14_6$, $P15_0$-$P15_7$(Note), $X_{IN}$, $\overline{RESET}$, CNVss, BYTE | $V_I$=3V | | | 4.0 | µA |
| $I_{IL}$ | "L" input current | $P0_0$-$P0_7$, $P1_0$-$P1_7$, $P2_0$-$P2_7$, $P3_0$-$P3_7$, $P4_0$-$P4_7$, $P5_0$-$P5_7$, $P6_0$-$P6_7$, $P7_2$-$P7_7$, $P8_0$-$P8_7$, $P9_0$-$P9_7$, $P10_0$-$P10_7$, $P11_0$-$P11_4$, $P12_0$-$P12_7$, $P13_0$-$P13_7$, $P14_0$-$P14_6$, $P15_0$-$P15_7$(Note), $X_{IN}$, $\overline{RESET}$, CNVss, BYTE | $V_I$=0V | | | -4.0 | µA |
| $R_{PULLUP}$ | Pull-up resistance | $P0_0$-$P0_7$, $P1_0$-$P1_7$, $P2_0$-$P2_7$, $P3_0$-$P3_7$, $P4_0$-$P4_7$, $P5_0$-$P5_7$, $P6_0$-$P6_7$, $P7_2$-$P7_7$, $P8_0$-$P8_4$, $P8_6$, $P8_7$, $P9_0$-$P9_7$, $P10_0$-$P10_7$, $P11_0$Å`$P11_4$, $P12_0$-$P12_7$, $P13_0$-$P13_7$, $P14_0$-$P14_6$, $P15_0$-$P15_7$(Note) | $V_I$=0V | 66 | 120 | 500 | kΩ |
| $R_{fXIN}$ | Feedback resistance | $X_{IN}$ | | | 3.0 | | MΩ |
| $R_{fXCIN}$ | Feedback resistance | $X_{CIN}$ | | | 20.0 | | MW |
| $V_{RAM}$ | RAM retention voltage | VDC-ON | | 2.5 | | | V |
| | | VDC-pass through | | 2.0 | | | V |
| $I_{CC}$ | Power supply current | Measuring condition: In sigle-chip mode, the output pins are open and other pins are Vss. — f($X_{IN}$)=20MHz, square wave, no division | | | 26 | 38 | mA |
| | | f($X_{CIN}$)=32kHz, with WAIT, VDC-pass through, Topr=25°C | | | 5.0 | | µA |
| | | f($X_{CIN}$)=32kHz, with WAIT, VDC-ON, Topr=25°C | | | 340 | | µA |
| | | When clock is stopped Topr=25°C | | | 0.4 | 20 | µA |

Note: Port P11 to P15 exist in 144-pin version.

# $V_{CC}$ = 3.3V

**Table 1.32.24. A-D conversion characteristics (referenced to $V_{CC}$ = $AV_{CC}$ = $V_{REF}$ = 3.0 to 3.6V, $V_{SS}$ = $AV_{SS}$ = 0V at Topr = –20 to 85°C, f($X_{IN}$) = 20MHz unless otherwise specified)**

| Symbol | Parameter | | Measuring condition | Standard | | | Unit |
|---|---|---|---|---|---|---|---|
| | | | | Min. | Typ. | Max | |
| – | Resolution | | $V_{REF}$ = $V_{CC}$ | | | 10 | Bits |
| ISL | Integral nonlinearity error | No S&H function(8-bit) | | | | ±2 | LSB |
| DSL | Differential nonlinearity error | No S&H function(8-bit) | | | | ±1 | LSB |
| – | Offset error | No S&H function(8-bit) | | | | ±2 | LSB |
| – | Gain error | No S&H function(8-bit) | | | | ±2 | LSB |
| $R_{LADDER}$ | Ladder resistance | | $V_{REF}$ = $V_{CC}$ | 8 | | 40 | kΩ |
| $t_{CONV}$ | Conversion time(8bit) | | | 9.8 | | | µs |
| $V_{REF}$ | Reference voltage | | | 2.7 | | $V_{CC}$ | V |
| $V_{IA}$ | Analog input voltage | | | 0 | | $V_{REF}$ | V |

S&H: Sample and hold

Note: Divide the frequency if f($X_{IN}$) exceeds 10 MHz, and make ØAD equal to or lower than 10 MHz.

**Table 1.32.25. D-A conversion characteristics (referenced to $V_{CC}$ = $V_{REF}$ = 3.0 to 3.6V, $V_{SS}$ = $AV_{SS}$ = 0V at Topr = –20 to 85°C, f($X_{IN}$) = 20MHz 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) | | | 1.0 | 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, the Vref is unconnected at the A-D control register 1, $I_{VREF}$ is sent.

# VCC = 3.3V

**Timing requirements (referenced to VCC = 3.0 to 3.6V, VSS = 0V at Topr = –20 to 85°C unless otherwise specified)**

**Table 1.32.26. External clock input**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| tc | External clock input cycle time | 50 | | ns |
| tw(H) | External clock input HIGH pulse width | 22 | | ns |
| tw(L) | External clock input LOW pulse width | 22 | | ns |
| tr | External clock rise time | | 5 | ns |
| tf | External clock fall time | | 5 | ns |

**Table 1.32.27. Memory expansion and microprocessor modes**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| tac1(RD-DB) | Data input access time (RD standard, no wait) | | (Note) | ns |
| tac1(AD-DB) | Data input access time (AD standard, CS standard, no wait) | | (Note) | ns |
| tac2(RD-DB) | Data input access time (RD standard, with wait) | | (Note) | ns |
| tac2(AD-DB) | Data input access time (AD standard, CS standard, with wait) | | (Note) | ns |
| tac3(RD-DB) | Data input access time (RD standard, when accessing multiplex bus area) | | (Note) | ns |
| tac3(AD-DB) | Data input access time (AD standard, CS standard, when accessing multiplex bus area) | | (Note) | ns |
| tac4(RAS-DB) | Data input access time (RAS standard, DRAM access) | | (Note) | ns |
| tac4(CAS-DB) | Data input access time (CAS standard, DRAM access) | | (Note) | ns |
| tac4(CAD-DB) | Data input access time (CAD standard, DRAM access) | | (Note) | ns |
| tsu(DB-BCLK) | Data input setup time | 30 | | ns |
| tsu(RDY-BCLK ) | $\overline{RDY}$ input setup time | 40 | | ns |
| tsu(HOLD-BCLK ) | $\overline{HOLD}$ input setup time | 60 | | ns |
| th(RD-DB) | Data input hold time | 0 | | ns |
| th(CAS-DB) | Data input hold time | 0 | | ns |
| th(BCLK -RDY) | $\overline{RDY}$ input hold time | 0 | | ns |
| th(BCLK-HOLD ) | $\overline{HOLD}$ input hold time | 0 | | ns |
| td(BCLK-HLDA ) | $\overline{HLDA}$ output delay time | | 25 | ns |

Note: Calculated according to the BCLK frequency as follows:
Note that inserting wait or using lower operation frequency f(BCLK) is needed when calculated value is negative.

$$t_{ac1(RD-DB)} = \frac{10^9}{f(BCLK) \times 2} - 35 \quad [ns]$$

$$t_{ac1(AD-DB)} = \frac{10^9}{f(BCLK)} - 35 \quad [ns]$$

$$t_{ac2(RD-DB)} = \frac{10^9 \times m}{f(BCLK) \times 2} - 35 \quad [ns] \text{ (m=3, 5 and 7 when 1 wait, 2 wait and 3 wait, respectively)}$$

$$t_{ac2(AD-DB)} = \frac{10^9 \times n}{f(BCLK)} - 35 \quad [ns] \text{ (n=2, 3 and 4 when 1 wait, 2 wait and 3 wait, respectively)}$$

$$t_{ac3(RD-DB)} = \frac{10^9 \times m}{f(BCLK) \times 2} - 35 \quad [ns] \text{ (m=3 and 5 when 2 wait and 3 wait, respectively)}$$

$$t_{ac3(AD-DB)} = \frac{10^9 \times n}{f(BCLK) \times 2} - 35 \quad [ns] \text{ (n=5 and 7 when 2 wait and 3 wait, respectively)}$$

$$t_{ac4(RAS-DB)} = \frac{10^9 \times m}{f(BCLK) \times 2} - 35 \quad [ns] \text{ (m=3 and 5 when 1 wait and 2 wait, respectively)}$$

$$t_{ac4(CAS-DB)} = \frac{10^9 \times n}{f(BCLK) \times 2} - 35 \quad [ns] \text{ (n=1 and 3 when 1 wait and 2 wait, respectively)}$$

$$t_{ac4(CAD-DB)} = \frac{10^9 \times l}{f(BCLK)} - 35 \quad [ns] \text{ (l=1 and 2 when 1 wait and 2 wait, respectively)}$$

# VCC = 3.3V

**Timing requirements (referenced to VCC = 3.0 to 3.6V, VSS = 0V at Topr = –20 to 85°C unless otherwise specified)**

**Table 1.32.28. Timer A input (counter input in event counter mode)**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TA)}$ | TAiIN input cycle time | 100 | | ns |
| $t_{w(TAH)}$ | TAiIN input HIGH pulse width | 40 | | ns |
| $t_{w(TAL)}$ | TAiIN input LOW pulse width | 40 | | ns |

**Table 1.32.29. Timer A input (gating input in timer mode)**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TA)}$ | TAiIN input cycle time | 400 | | ns |
| $t_{w(TAH)}$ | TAiIN input HIGH pulse width | 200 | | ns |
| $t_{w(TAL)}$ | TAiIN input LOW pulse width | 200 | | ns |

**Table 1.32.30. Timer A input (external trigger input in one-shot timer mode)**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TA)}$ | TAiIN input cycle time | 200 | | ns |
| $t_{w(TAH)}$ | TAiIN input HIGH pulse width | 100 | | ns |
| $t_{w(TAL)}$ | TAiIN input LOW pulse width | 100 | | ns |

**Table 1.32.31. Timer A input (external trigger input in pulse width modulation mode)**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{w(TAH)}$ | TAiIN input HIGH pulse width | 100 | | ns |
| $t_{w(TAL)}$ | TAiIN input LOW pulse width | 100 | | ns |

**Table 1.32.32. Timer A input (up/down input in event counter mode)**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(UP)}$ | TAiOUT input cycle time | 2000 | | ns |
| $t_{w(UPH)}$ | TAiOUT input HIGH pulse width | 1000 | | ns |
| $t_{w(UPL)}$ | TAiOUT input LOW pulse width | 1000 | | ns |
| $t_{su(UP-TIN)}$ | TAiOUT input setup time | 400 | | ns |
| $t_{h(TIN-UP)}$ | TAiOUT input hold time | 400 | | ns |

# VCC = 3.3V

**Timing requirements (referenced to VCC = 3.0 to 3.6V, VSS = 0V at Topr = –20 to 85°C unless otherwise specified)**

**Table 1.32.33. Timer B input (counter input in event counter mode)**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TB)}$ | TBiIN input cycle time (counted on one edge) | 100 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width (counted on one edge) | 40 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width (counted on one edge) | 40 | | ns |
| $t_{c(TB)}$ | TBiIN input cycle time (counted on both edges) | 200 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width (counted on both edges) | 80 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width (counted on both edges) | 80 | | ns |

**Table 1.32.34. Timer B input (pulse period measurement mode)**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TB)}$ | TBiIN input cycle time | 400 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width | 200 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width | 200 | | ns |

**Table 1.32.35. Timer B input (pulse width measurement mode)**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TB)}$ | TBiIN input cycle time | 400 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width | 200 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width | 200 | | ns |

**Table 1.32.36. A-D trigger input**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(AD)}$ | $\overline{ADTRG}$ input cycle time (trigger able minimum) | 1000 | | ns |
| $t_{w(ADL)}$ | $\overline{ADTRG}$ input LOW pulse width | 125 | | ns |

**Table 1.32.37. Serial I/O**

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

**Table 1.32.38. External interrupt $\overline{INTi}$ inputs**

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

# VCC = 3.3V

**Switching characteristics (referenced to VCC = 3.0 to 3.6V, VSS = 0V at Topr = –20 to 85°C, unless otherwise specified)**

**Table 1.32.39. Memory expansion and microprocessor modes (with no wait)**

| Symbol | Parameter | Measuring condition | Standard | | Unit |
|---|---|---|---|---|---|
| | | | Min. | Max. | |
| $t_{d(BCLK-AD)}$ | Address output delay time | Figure 1.32.1 | | 18 | ns |
| $t_{h(BCLK-AD)}$ | Address output hold time (BCLK standard) | | 0 | | ns |
| $t_{h(RD-AD)}$ | Address output hold time (RD standard) | | 0 | | ns |
| $t_{h(WR-AD)}$ | Address output hold time (WR standard) | | (Note) | | ns |
| $t_{d(BCLK-CS)}$ | Chip select output delay time | | | 18 | ns |
| $t_{h(BCLK-CS)}$ | Chip select output hold time (BCLK standard) | | 0 | | ns |
| $t_{h(RD-CS)}$ | Chip select output hold time (RD standard) | | 0 | | ns |
| $t_{h(WR-CS)}$ | Chip select output hold time (WR standard) | | (Note) | | ns |
| $t_{d(BCLK-ALE)}$ | ALE signal output delay time | | | 18 | ns |
| $t_{h(BCLK-ALE)}$ | ALE signal output hold time | | – 2 | | ns |
| $t_{d(BCLK-RD)}$ | RD signal output delay time | | | 18 | ns |
| $t_{h(BCLK-RD)}$ | RD signal output hold time | | – 3 | | ns |
| $t_{d(BCLK-WR)}$ | WR signal output delay time | | | 18 | ns |
| $t_{h(BCLK-WR)}$ | WR signal output hold time | | 0 | | 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 |
| $t_{w(WR)}$ | Write pulse width | | (Note) | | ns |

Note: Calculated according to the BCLK frequency as follows:

$$t_{d(DB-WR)} = \frac{10^9}{f_{(BCLK)}} - 20 \quad [ns]$$

$$t_{h(WR-DB)} = \frac{10^9}{f_{(BCLK)} \times 2} - 10 \quad [ns]$$

$$t_{h(WR-AD)} = \frac{10^9}{f_{(BCLK)} \times 2} - 10 \quad [ns]$$

$$t_{h(WR-CS)} = \frac{10^9}{f_{(BCLK)} \times 2} - 10 \quad [ns]$$

$$t_{w(WR)} = \frac{10^9}{f_{(BCLK)} \times 2} - 15 \quad [ns]$$

# VCC = 3.3V

**Switching characteristics (referenced to VCC = 3.0 to 3.6V, VSS = 0V at Topr = –20 to 85°C unless otherwise specified)**

**Table 1.32.40. Memory expansion and microprocessor modes (with wait, accessing external memory)**

| Symbol | Parameter | Measuring condition | Standard | | Unit |
|---|---|---|---|---|---|
| | | | Min. | Max. | |
| $t_{d(BCLK-AD)}$ | Address output delay time | Figure 1.32.1 | | 18 | ns |
| $t_{h(BCLK-AD)}$ | Address output hold time (BCLK standard) | | 0 | | ns |
| $t_{h(RD-AD)}$ | Address output hold time (RD standard) | | 0 | | ns |
| $t_{h(WR-AD)}$ | Address output hold time (WR standard) | | (Note) | | ns |
| $t_{d(BCLK-CS)}$ | Chip select output delay time | | | 18 | ns |
| $t_{h(BCLK-CS)}$ | Chip select output hold time (BCLK standard) | | 0 | | ns |
| $t_{h(RD-CS)}$ | Chip select output hold time (RD standard) | | 0 | | ns |
| $t_{h(WR-CS)}$ | Chip select output hold time (WR standard) | | (Note) | | ns |
| $t_{d(BCLK-ALE)}$ | ALE signal output delay time | | | 18 | ns |
| $t_{h(BCLK-ALE)}$ | ALE signal output hold time | | – 2 | | ns |
| $t_{d(BCLK-RD)}$ | RD signal output delay time | | | 18 | ns |
| $t_{h(BCLK-RD)}$ | RD signal output hold time | | – 3 | | ns |
| $t_{d(BCLK-WR)}$ | WR signal output delay time | | | 18 | ns |
| $t_{h(BCLK-WR)}$ | WR signal output hold time | | 0 | | 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 |
| $t_{w(WR)}$ | Write pulse width | | (Note) | | ns |

Note: Calculated according to the BCLK frequency as follows:

$$t_{d(DB-WR)} = \frac{10^9 \times n}{f_{(BCLK)}} - 20 \quad [ns] \quad (n=1, 2 \text{ and } 3 \text{ when 1 wait, 2 wait and 3 wait, respectively})$$

$$t_{h(WR-DB)} = \frac{10^9}{f_{(BCLK)} \times 2} - 10 \quad [ns]$$

$$t_{h(WR-AD)} = \frac{10^9}{f_{(BCLK)} \times 2} - 10 \quad [ns]$$

$$t_{h(WR-CS)} = \frac{10^9}{f_{(BCLK)} \times 2} - 10 \quad [ns]$$

$$t_{w(WR)} = \frac{10^9 \times n}{f_{(BCLK)} \times 2} - 15 \quad [ns] \quad (n=1, 3 \text{ and } 5 \text{ when 1 wait, 2 wait and 3 wait, respectively})$$

# VCC = 3.3V

**Switching characteristics (referenced to VCC = 3.0 to 3.6V, VSS = 0V at Topr = –20 to 85°C unless otherwise specified)**

**Table 1.32.41. Memory expansion and microprocessor modes (with wait, accessing external memory, multiplex bus area selected)**

| Symbol | Parameter | Measuring condition | Standard | | Unit |
|---|---|---|---|---|---|
| | | | Min. | Max. | |
| td(BCLK-AD) | Address output delay time | Figure 1.32.1 | | 18 | ns |
| th(BCLK-AD) | Address output hold time (BCLK standard) | | 0 | | ns |
| th(RD-AD) | Address output hold time (RD standard) | | (Note) | | ns |
| th(WR-AD) | Address output hold time (WR standard) | | (Note) | | ns |
| td(BCLK-CS) | Chip select output delay time | | | 18 | ns |
| th(BCLK-CS) | Chip select output hold time (BCLK standard) | | 0 | | ns |
| th(RD-CS) | Chip select output hold time (RD standard) | | (Note) | | ns |
| th(WR-CS) | Chip select output hold time (WR standard) | | (Note) | | ns |
| td(BCLK-RD) | RD signal output delay time | | | 18 | ns |
| th(BCLK-RD) | RD signal output hold time | | – 3 | | ns |
| td(BCLK-WR) | WR signal output delay time | | | 18 | ns |
| th(BCLK-WR) | WR signal output hold time | | 0 | | ns |
| td(DB-WR) | Data output delay time (WR standard) | | (Note) | | ns |
| th(WR-DB) | Data output hold time (WR standard) | | (Note) | | ns |
| td(BCLK-ALE) | ALE signal output delay time (BCLK standard) | | | 18 | ns |
| th(BCLK-ALE) | ALE signal output hold time (BCLK standard) | | – 2 | | ns |
| td(AD-ALE) | ALE signal output delay time (address standard) | | (Note) | | ns |
| th(ALE-AD) | ALE signal output hold time (address standard) | | (Note) | | ns |
| tdz(RD-AD) | Address output flowting start time | | | 8 | ns |

Note: Calculated according to the BCLK frequency as follows:

$$t_{h(RD-AD)} = \frac{10^9}{f_{(BCLK)} \times 2} - 10 \quad [ns]$$

$$t_{h(WR-AD)} = \frac{10^9}{f_{(BCLK)} \times 2} - 10 \quad [ns]$$

$$t_{h(RD-CS)} = \frac{10^9}{f_{(BCLK)} \times 2} - 10 \quad [ns]$$

$$t_{h(WR-CS)} = \frac{10^9}{f_{(BCLK)} \times 2} - 10 \quad [ns]$$

$$t_{d(DB-WR)} = \frac{10^9 \times m}{f_{(BCLK)} \times 2} - 25 \quad [ns] \quad (m=3 \text{ and } 5 \text{ when 2 wait and 3 wait, respectively})$$

$$t_{h(WR-DB)} = \frac{10^9}{f_{(BCLK)} \times 2} - 10 \quad [ns]$$

$$t_{d(AD-ALE)} = \frac{10^9}{f_{(BCLK)} \times 2} - 20 \quad [ns]$$

$$t_{h(ALE-AD)} = \frac{10^9}{f_{(BCLK)} \times 2} - 10 \quad [ns]$$

# VCC = 3.3V

**Switching characteristics (referenced to VCC = 3.0 to 3.6V, VSS = 0V at Topr = –20 to 85°C unless otherwise specified)**

**Table 1.32.42. Memory expansion and microprocessor modes (with wait, accessing external memory, DRAM area selected)**

| Symbol | Parameter | Measuring condition | Standard | | Unit |
|---|---|---|---|---|---|
| | | | Min. | Max. | |
| $t_{d(BCLK-RAD)}$ | Row address output delay time | Figure 1.32.1 | | 18 | ns |
| $t_{h(BCLK-RAD)}$ | Row address output hold time (BCLK standard) | | 0 | | ns |
| $t_{d(BCLK-CAD)}$ | String address output delay time | | | 18 | ns |
| $t_{h(BCLK-CAD)}$ | String address output hold time (BCLK standard) | | 0 | | ns |
| $t_{h(RAS-RAD)}$ | Row address output hold time after RAS output | | (Note) | | ns |
| $t_{d(BCLK-RAS)}$ | RAS output delay time (BCLK standard) | | | 18 | ns |
| $t_{h(BCLK-RAS)}$ | RAS output hold time (BCLK standard) | | 0 | | ns |
| $t_{RP}$ | RAS "H" hold time | | (Note) | | ns |
| $t_{d(BCLK-CAS)}$ | CAS output delay time (BCLK standard) | | | 18 | ns |
| $t_{h(BCLK-CAS)}$ | CAS output hold time (BCLK standard) | | 0 | | ns |
| $t_{d(BCLK-DW)}$ | Data output delay time (BCLK standard) | | | 18 | ns |
| $t_{h(BCLK-DW)}$ | Data output hold time (BCLK standard) | | – 3 | | ns |
| $t_{su(DB-CAS)}$ | CAS after DB output setup time | | (Note) | | ns |
| $t_{h(BCLK-DB)}$ | DB signal output hold time (BCLK standard) | | – 7 | | ns |
| $t_{su(CAS-RAS)}$ | CAS output setup time before RAS output (refresh) | | (Note) | | ns |

Note: Calculated according to the BCLK frequency as follows:

$$t_{h(RAS-RAD)} = \frac{10^9}{f_{(BCLK)} \times 2} - 13 \quad [ns]$$

$$t_{RP} = \frac{10^9 \times 3}{f_{(BCLK)} \times 2} - 20 \quad [ns]$$

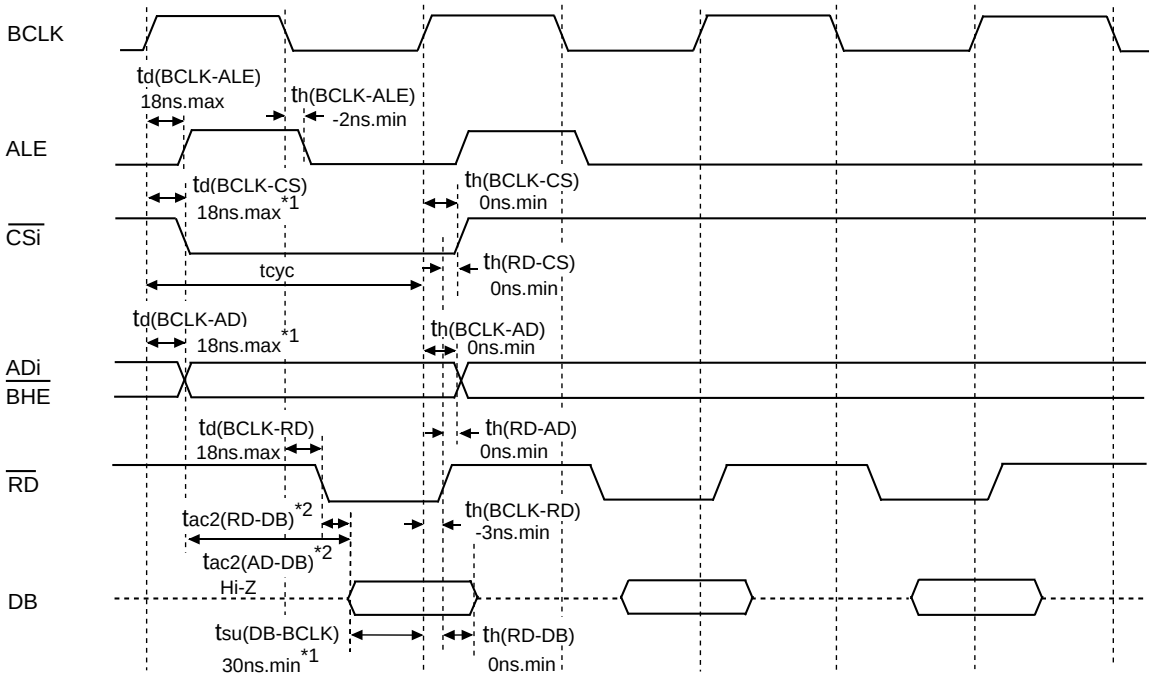
$$t_{su(DB-CAS)} = \frac{10^9}{f_{(BCLK)}} - 20 \quad [ns]$$

$$t_{su(CAS-RAS)} = \frac{10^9}{f_{(BCLK)} \times 2} - 13 \quad [ns]$$

Vcc=3.3V

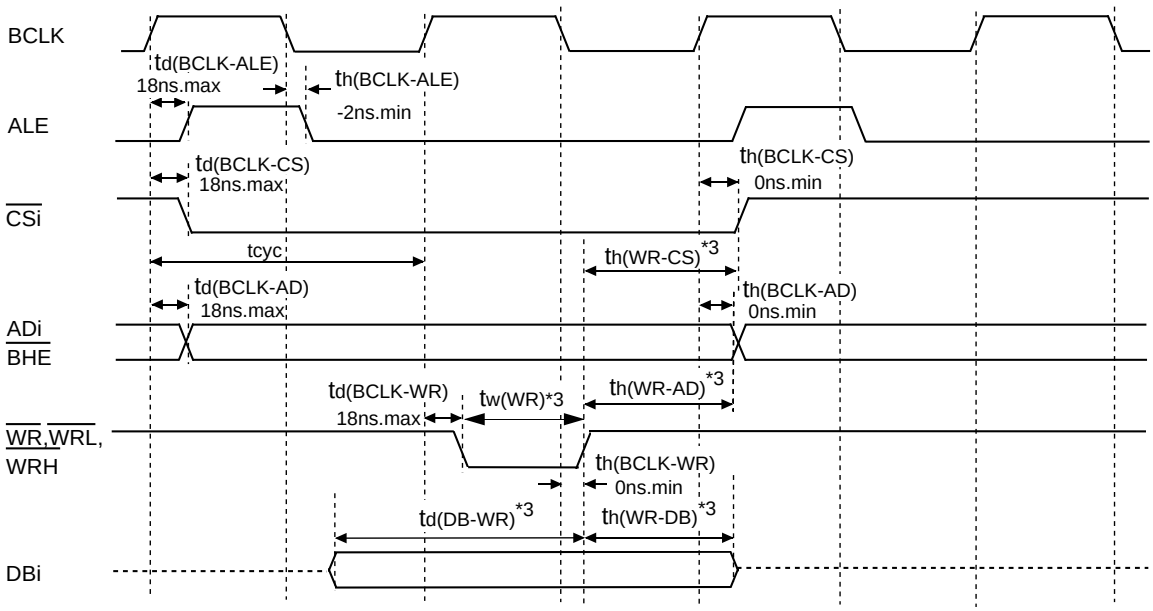
## Memory expansion Mode and Microprocessor Mode (without wait)

### Read Timing




*1:It is a guarantee value with being alone. 35ns.max garantees as $t_{d(BCLK-AD)}+t_{su(DB-BCLK)}$.
*2:It depends on operation frequency.
$t_{ac2(RD-DB)}$=(tcyc/2-35)ns.max
$t_{ac2(AD-DB)}$=(tcyc-35)ns.max

### Write Timing




*3:It depends on operation frequency.
$t_{d(DB-WR)}$=(tcyc-20)ns.min
$t_{h(WR-DB)}$=(tcyc/2-10)ns.min
$t_{h(WR-AD)}$=(tcyc/2-10)ns.min
$t_{h(WR-CS)}$=(tcyc/2-10)ns.min
$t_{w(WR)}$=(tcyc/2-15)ns.min

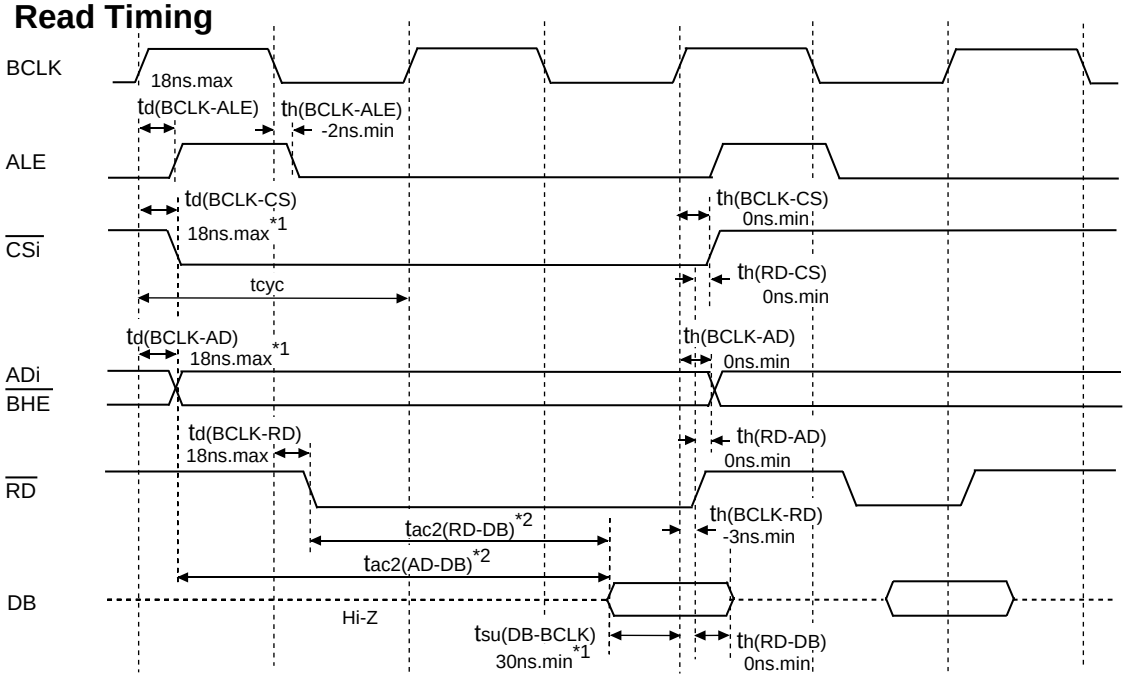
Measuring conditions
- $V_{CC}$=3.0V to 3.6V
- Input timing voltage :Determined with $V_{IH}$=1.5V, $V_{IL}$=0.5V
- Output timing voltage :Determined with $V_{OH}$=1.5V, $V_{OL}$=1.5V

**Figure 1.32.10. $V_{CC}$=3.3V timing diagram (1)**

Vcc=3.3V

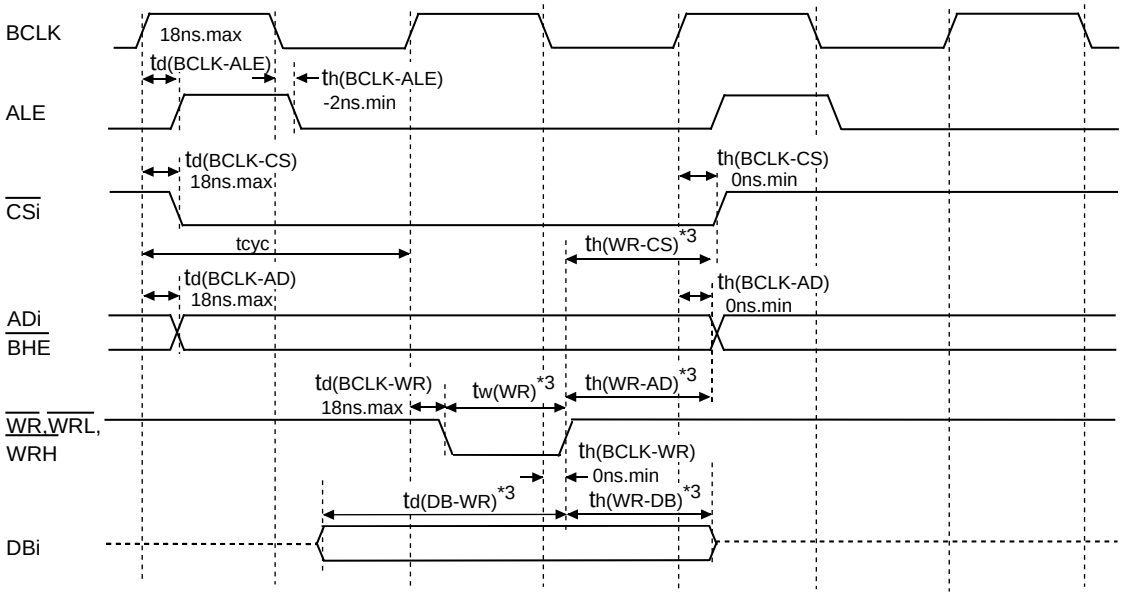
## Memory expansion Mode and Microprocessor Mode (with wait)

### Read Timing




*1:It is a guarantee value with being alone. 35ns.max garantees as $t_{d(BCLK-AD)}+t_{su(DB-BCLK)}$.
*2:It depends on operation frequency.
$t_{ac2(RD-DB)}$=(tcyc/2 x m-35)ns.max (m=3, 5 and 7 when 1 wait, 2 wait and 3 wait, respectively.)
$t_{ac2(AD-DB)}$=(tcyc x n-35)ns.max (n=2, 3 and 4 when 1 wait, 2 wait and 3 wait, respectively.)

### Write Timing




*3:It depends on operation frequency.
$t_{d(DB-WR)}$=(tcyc x n-20)ns.min
(n=1, 2 and 3 when 1 wait, 2 wait and 3 wait, respectively.)
$t_{h(WR-DB)}$=(tcyc/2-10)ns.min
$t_{h(WR-AD)}$=(tcyc/2-10)ns.min
$t_{h(WR-CS)}$=(tcyc/2-10)ns.min
$t_{w(WR)}$=(tcyc/2 x n-15)ns.min
(n=1, 3 and 5 when 1 wait, 2 wait and 3 wait, respectively.)

Measuring conditions
- $V_{CC}$=3.0V to 3.6V
- Input timing voltage
  :Determined with $V_{IH}$=1.5V, $V_{IL}$=0.5V
- Output timing voltage
  :Determined with $V_{OH}$=1.5V, $V_{OL}$=1.5V

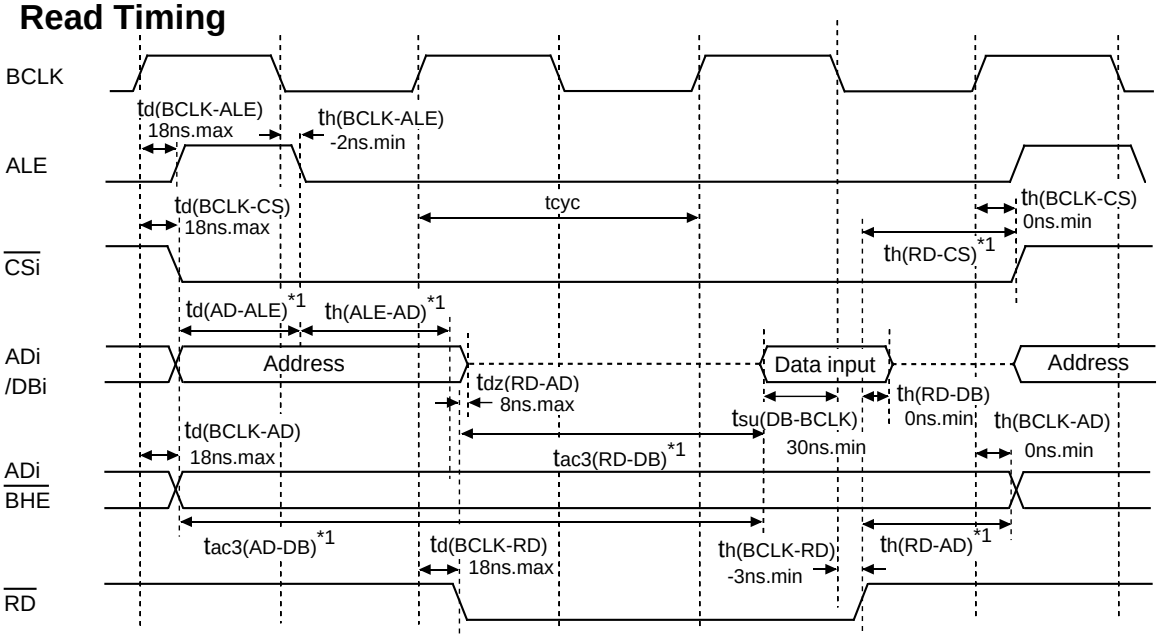
**Figure 1.32.11. $V_{CC}$=3.3V timing diagram (2)**

## Memory expansion Mode and Microprocessor Mode

Vcc=3.3V

**(When accessing external memory area with wait, and select multiplexed bus)**

### Read Timing




*1:It depends on operation frequency.

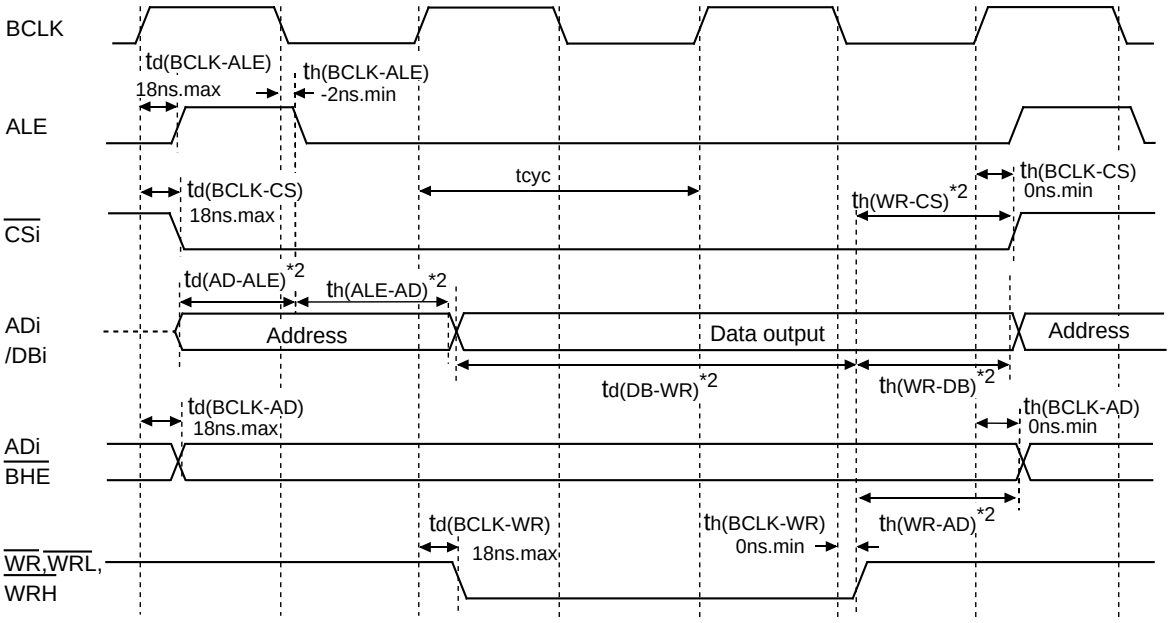
$t_{d(AD-ALE)}$=(tcyc/2-20)ns.min

$t_{h(ALE-AD)}$=(tcyc/2-10)ns.min, $t_{h(RD-AD)}$=(tcyc/2-10)ns.min, $t_{h(RD-CS)}$=(tcyc/2-10)ns.min

$t_{ac3(RD-DB)}$=(tcyc/2 x m-35)ns.max (m=3 and 5 when 2 wait and 3 wait, respectively.)

$t_{ac3(AD-DB)}$=(tcyc/2 x n-35)ns.max (n=5 and 7 when 2 wait and 3 wait, respectively.)

### Write Timing




*2:It depends on operation frequency.

$t_{d(AD-ALE)}$=(tcyc/2-20)ns.min

$t_{h(ALE-AD)}$=(tcyc/2-10)ns.min, $t_{h(WR-AD)}$=(tcyc/2-10)ns.min

$t_{h(WR-CS)}$=(tcyc/2-10)ns.min, $t_{h(WR-DB)}$=(tcyc/2-10)ns.min

$t_{d(DB-WR)}$=(tcyc/2 x m-25)ns.min

(m=3 and 5 when 2 wait and 3 wait, respectively.)

Measuring conditions

- $V_{CC}$=3.0V to 3.6V
- Input timing voltage
  :Determined with $V_{IH}$=1.5V, $V_{IL}$=0.5V
- Output timing voltage
  :Determined with $V_{OH}$=1.5V, $V_{OL}$=1.5V

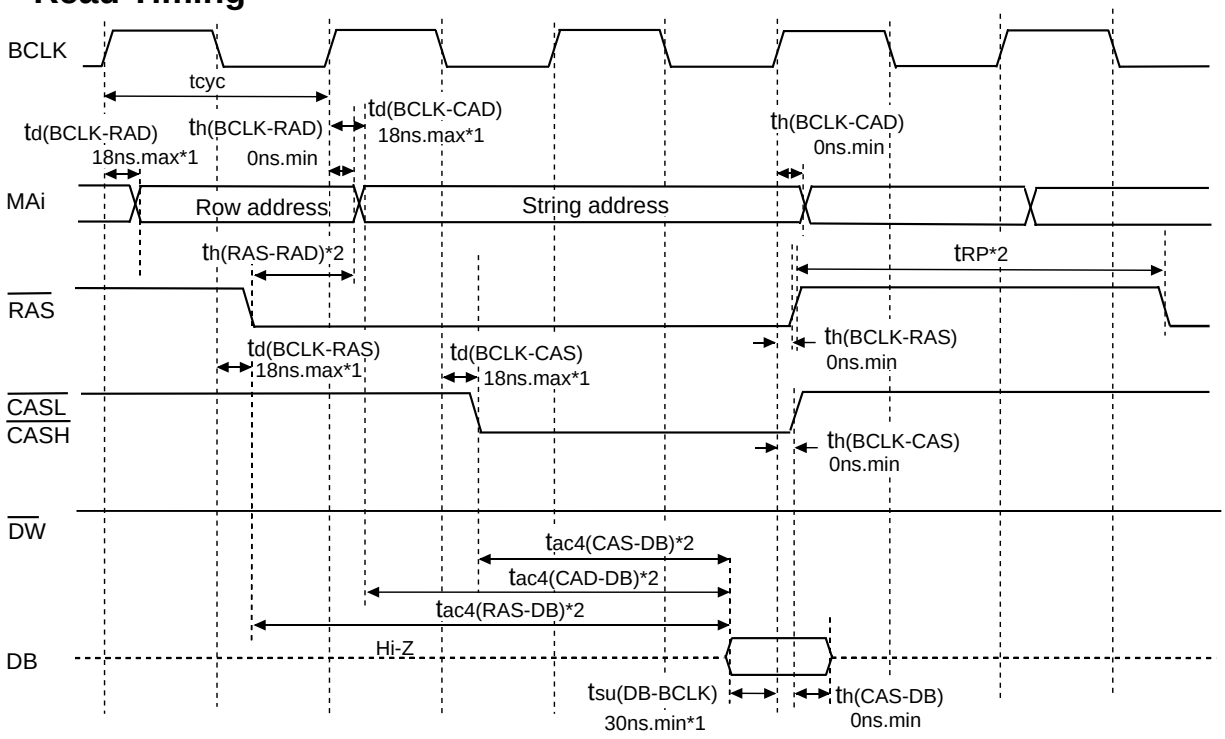
**Figure 1.32.12. $V_{CC}$=3.3V timing diagram (3)**

## Memory expansion Mode and Microprocessor Mode

Vcc=3.3V

**(When accessing DRAM area with 2 wait)**

### Read Timing




*1:It is a guarantee value with being alone. 35ns.max garantees as follows:

$t_{d(BCLK-RAS)} + t_{su(DB-BCLK)}$

$t_{d(BCLK-CAS)} + t_{su(DB-BCLK)}$

$t_{d(BCLK-CAD)} + t_{su(DB-BCLK)}$

*2:It depends on operation frequency.

$t_{ac4(RAS-DB)}$=(tcyc/2 x m-35)ns.max (m=3 and 5 when 1 wait and 2 wait, respectively.)

$t_{ac4(CAS-DB)}$=(tcyc/2 x n-35)ns.max (n=1 and 3 when 1 wait and 2 wait, respectively.)

$t_{ac4(CAD-DB)}$=(tcyc x l-35)ns.max (l=1 and 2 when 1 wait and 2 wait, respectively.)

$t_{h(RAS-RAD)}$=(tcyc/2-13)ns.min

$t_{RP}$=(tcyc/2 x 3-20)ns.min

Measuring conditions

- $V_{CC}$=3.0V to 3.6V
- Input timing voltage
  :Determined with $V_{IH}$=1.5V, $V_{IL}$=0.5V
- Output timing voltage
  :Determined with $V_{OH}$=1.5V, $V_{OL}$=1.5V

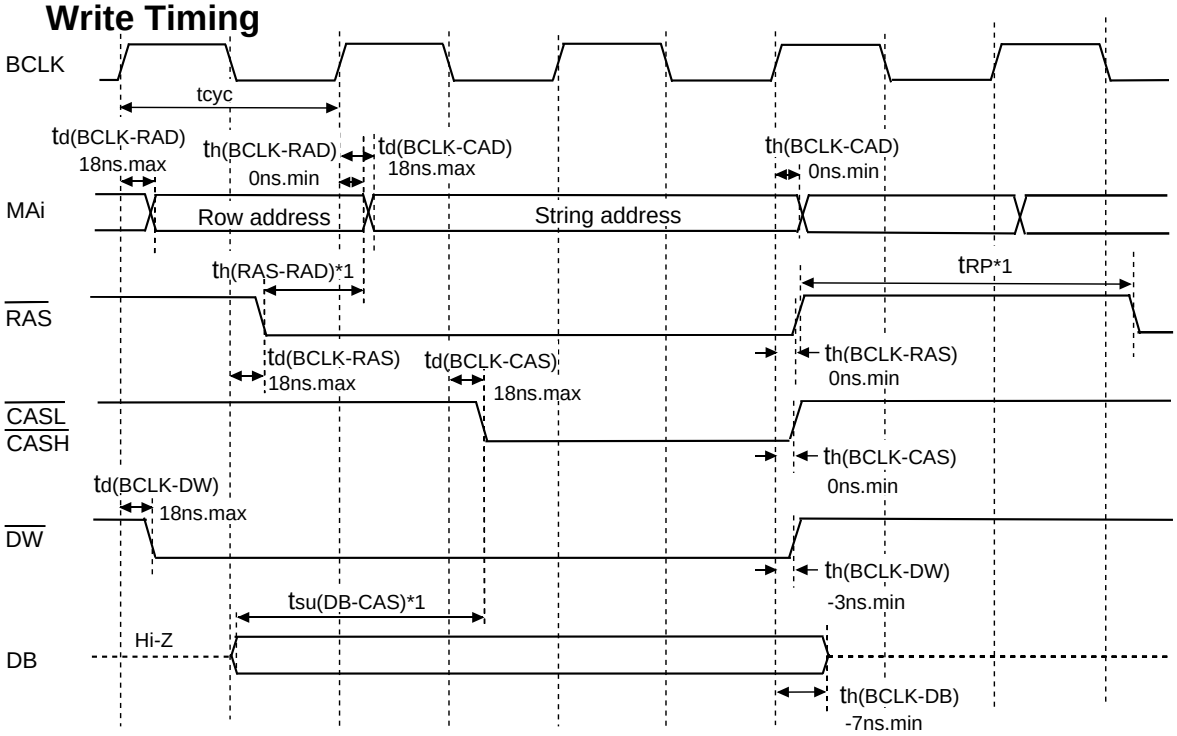
**Figure 1.32.13. $V_{CC}$=3.3V timing diagram (4)**

## Memory expansion Mode and Microprocessor Mode

Vcc=3.3V

**(When accessing DRAM area with 2 wait)**

### Write Timing




*1:It depends on operation frequency.

$t_{h(RAS-RAD)}$=(tcyc/2-13)ns.min

$t_{RP}$=(tcyc/2 x 3-20)ns.min

$t_{su(DB-CAS)}$=(tcyc-20)ns.min

Measuring conditions

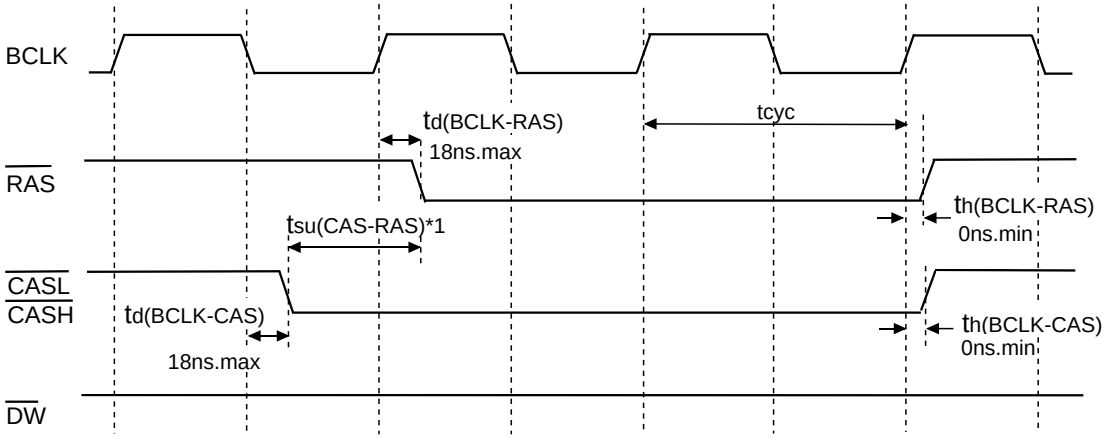
- $V_{CC}$=3.0V to 3.6V
- Input timing voltage
  :Determined with $V_{IH}$=1.5V, $V_{IL}$=0.5V
- Output timing voltage
  :Determined with $V_{OH}$=1.5V, $V_{OL}$=1.5V

**Figure 1.32.14. $V_{CC}$=3.3V timing diagram (5)**

## Memory expansion Mode and Microprocessor Mode

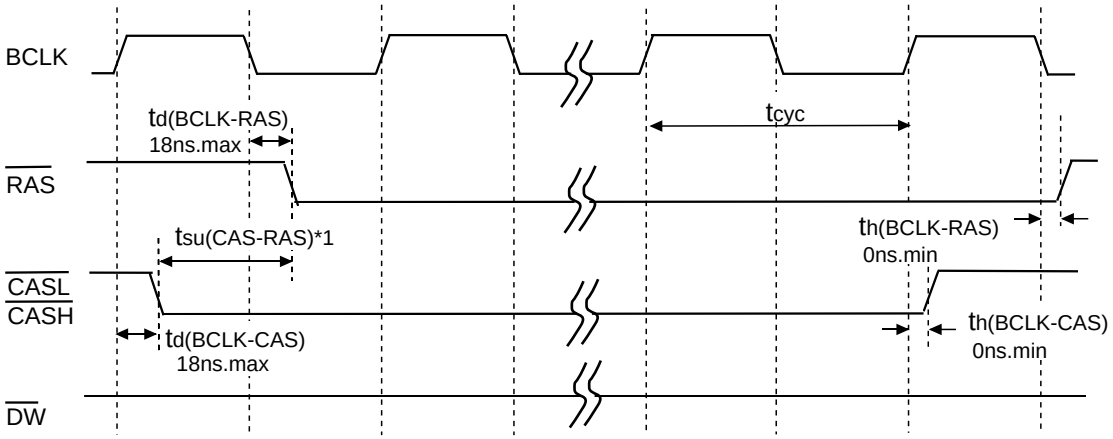
Vcc=3.3V

### Refresh Timing (CAS before RAS refresh)




*1:It depends on operation frequency.
$t_{su(CAS-RAS)}$=(tcyc/2-13)ns.min

### Refresh Timing (Self-refresh)




*1:It depends on operation frequency.
$t_{su(CAS-RAS)}$=(tcyc/2-13)ns.min

Measuring conditions
- $V_{CC}$=3.0V to 3.6V
- Input timing voltage
  :Determined with $V_{IH}$=1.5V, $V_{IL}$=0.5V
- Output timing voltage
  :Determined with $V_{OH}$=1.5V, $V_{OL}$=1.5V

**Figure 1.32.15. $V_{CC}$=3.3V timing diagram (6)**

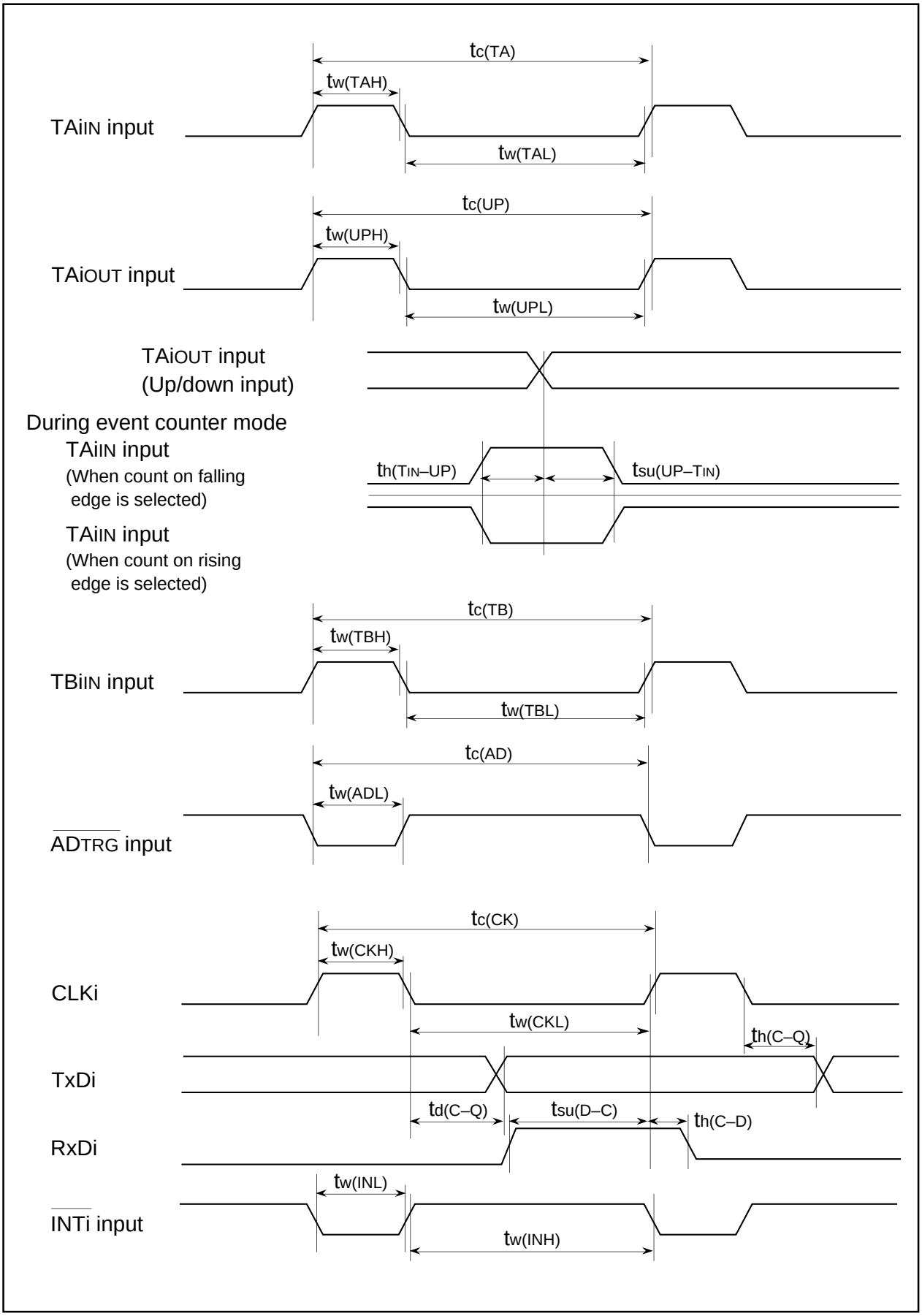
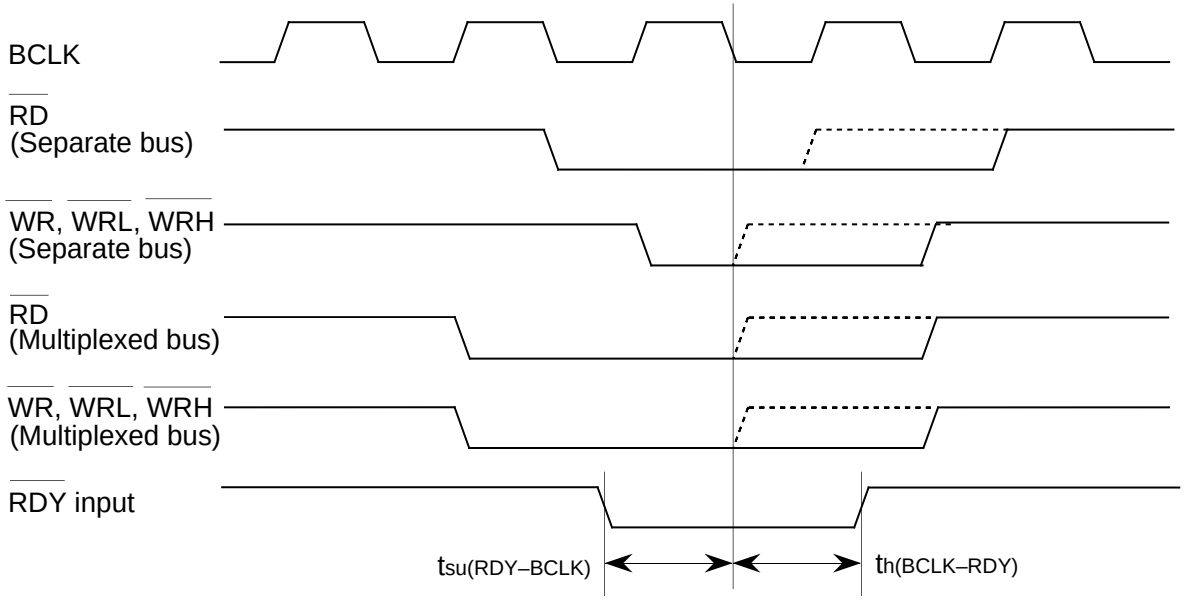


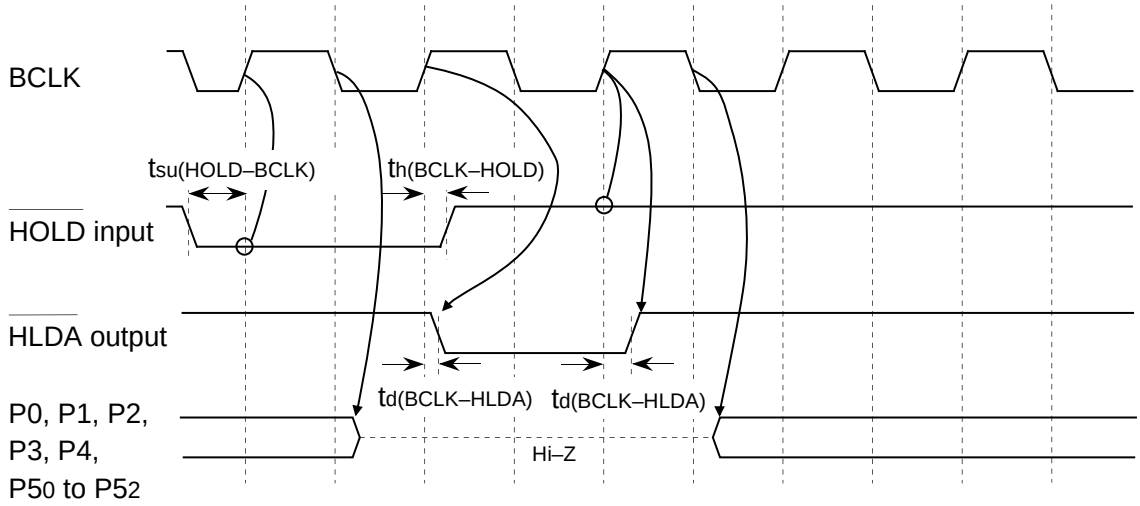

**Figure 1.32.16. Vcc=3.3V timing diagram (7)**

## Memory Expansion Mode and Microprocessor Mode

**(Valid only with wait)**




**(Valid with or without wait)**




Measuring conditions :

- $V_{CC}$=3.0 to 3.6V
- Input timing voltage : Determined with $V_{IH}$=2.4V, $V_{IL}$=0.6V
- Output timing voltage : Determined with $V_{OH}$=1.5V, $V_{OL}$=1.5V

**Figure 1.32.17. $V_{CC}$=3.3V timing diagram (8)**

## Outline Performance

Table 1.33.1 shows the outline performance of the M32C/83 (flash memory version).

**Table 1.33.1. Outline Performance of the M32C/83 (flash memory version)**

| Item | | Performance |
|---|---|---|
| Power supply voltage | | f(XIN)=30MHz, without wait, 4.2V to 5.5V<br>f(XIN)=20MHz, without wait, 3.0V to 3.6V |
| Program/erase voltage | | 4.2V to 5.5 V : f(BCLK)=12.5MHz, with one wait<br>: f(BCLK)=6.25MHz, without wait |
| Flash memory operation mode | | Three modes (parallel I/O, standard serial I/O, CPU rewrite) |
| Erase block division | User ROM area | See Figure 1.33.3 |
| | Boot ROM area | One division (8 Kbytes) (Note 1) |
| Program method | | In units of pages (in units of 256 bytes) |
| Erase method | | Collective erase/block erase |
| Program/erase control method | | Program/erase control by software command |
| Protect method | | Protected for each block by lock bit |
| Number of commands | | 8 commands |
| Program/erase count | | 100 times |
| Data holding | | 10 years |
| ROM code protect | | Parallel I/O and standard serial modes are supported. |

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

The following shows Mitsubishi plans to develop a line of M32C/83 products (flash memory version).

(1) ROM capacity

(2) Package 100P6S-A ... Plastic molded QFP
100P6Q-A ... Plastic molded QFP
144P6Q-A ... Plastic molded QFP

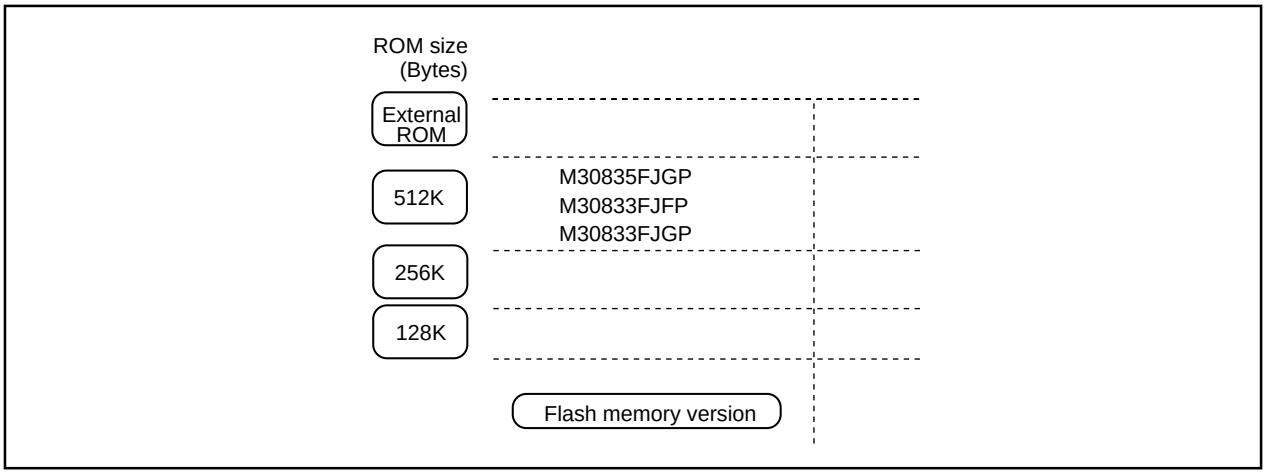



**Figure 1.33.1. ROM Expansion**

The following lists the M32C/83 products to be supported in the future.

**Table 1.33.2. Product List**

As of April, 2002

| Type No | ROM capacity | RAM capacity | Package type | Remarks |
|---|---|---|---|---|
| M30835FJGP ** | 512 Kbytes | 31 Kbytes | 144P6Q-A | |
| M30833FJGP ** | | | 100P6Q-A | |
| M30833FJFP ** | | | 100P6S-A | |

** : Under development

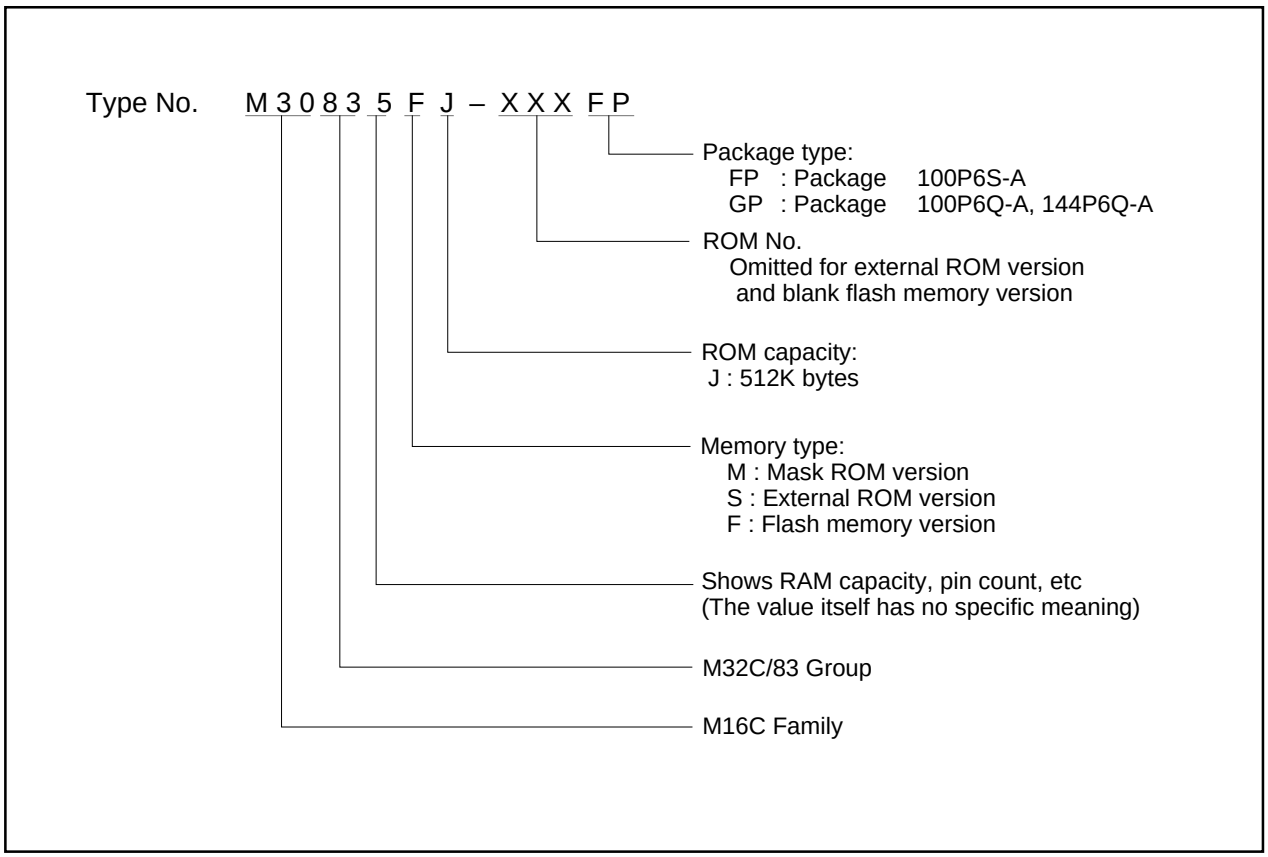



**Figure 1.33.2. Type No., memory size, and package**

## Flash Memory

The M32C/83 (flash memory version) contains the flash memory that can be rewritten with a single voltage of 5 V. For this flash memory, three flash memory modes are available in which to read, program, and erase: parallel I/O and standard serial I/O modes in which the flash memory can be manipulated using a programmer and a CPU rewrite mode in which the flash memory can be manipulated by the Central Processing Unit (CPU). Each mode is detailed in the pages to follow.

The flash memory is divided into several blocks as shown in Figure 1.33.3, so that memory can be erased one block at a time. Each block has a lock bit to enable or disable execution of an erase or program operation, allowing for data in each block to be protected.

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

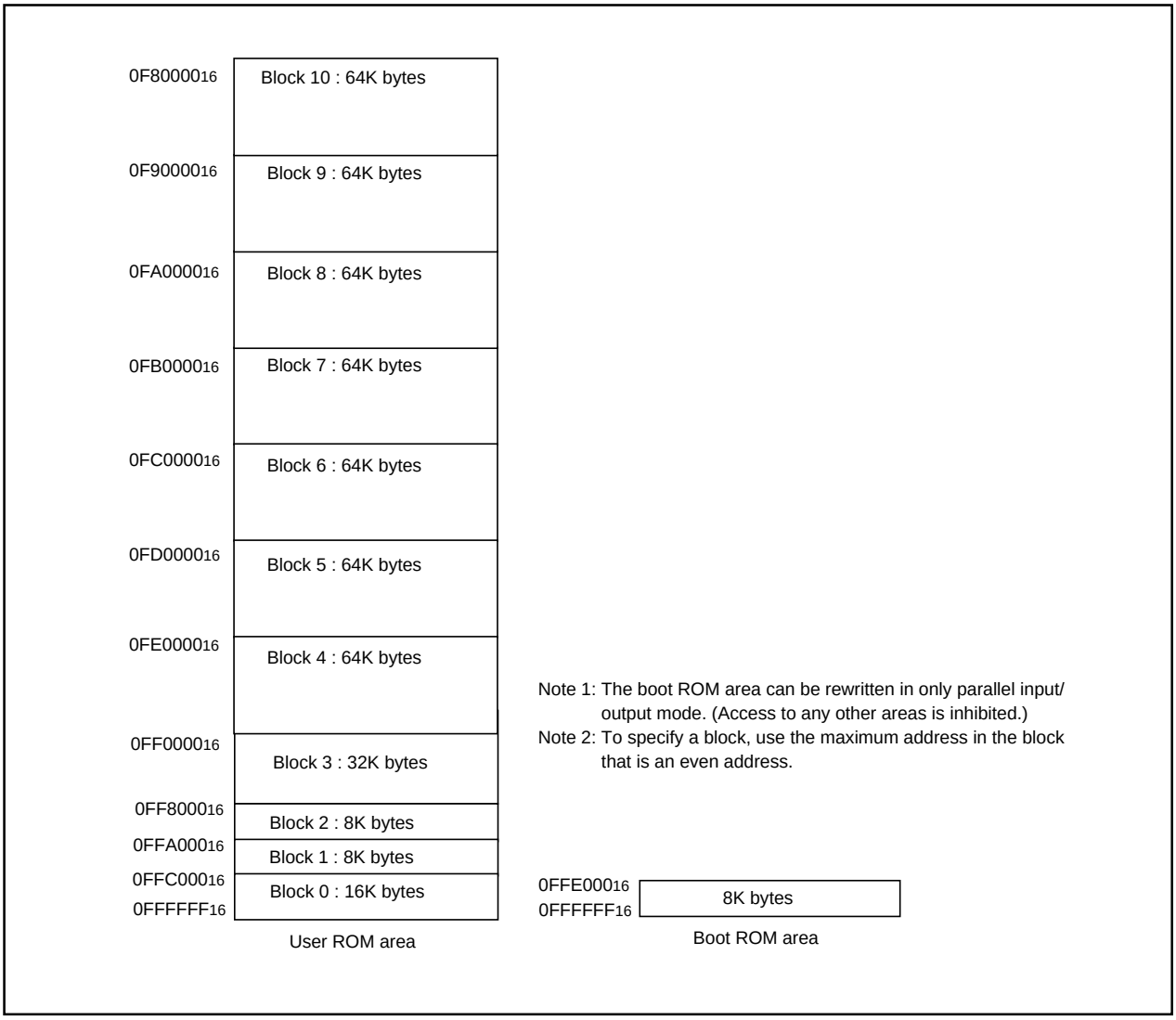



**Figure 1.33.3. Block diagram of flash memory version**

## CPU Rewrite Mode

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

In CPU rewrite mode, only the user ROM area shown in Figure 1.33.3 can be rewritten; the boot ROM area cannot be rewritten. Make sure the program and block erase commands are issued for only the user ROM area and each block area.

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

### Microcomputer Mode and Boot Mode

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

See Figure 1.33.3 for details about the boot ROM area.

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

When the microcomputer is reset by pulling the $P5_5$ pin low, the $CNV_{SS}$ pin high, and the $P5_0$ pin high, the CPU starts operating using the control program in the boot ROM area. This mode is called the “boot” mode. The control program in the boot ROM area can also be used to rewrite the user ROM area.

### Block Address

Block addresses refer to the maximum even address of each block. These addresses are used in the block erase command, lock bit program command, and read lock status command.

## Outline Performance of CPU Rewrite Mode

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

When the CPU rewrite mode select bit (bit 1 at address $0377_{16}$) is set to “1”, transition to CPU rewrite mode occurs and software commands can be accepted.

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

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

Figure 1.34.1 shows the flash memory control register 0.

Flash memory control register 0

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

| Symbol | Address | When reset |
|---|---|---|
| FMR0 | $0057_{16}$ | $XX000001_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| FMR00 | RY/$\overline{BY}$ signal status bit | 0: Busy (being written or erased)<br>1: Ready | ○ | — |
| FMR01 | CPU rewrite mode select bit (Note 1) | 0: Normal mode (Software commands invalid)<br>1: CPU rewrite mode (Software commands acceptable) | ○ | ○ |
| FMR02 | Lock bit disable bit (Note 2) | 0: Block lock by lock bit data is enabled<br>1: Block lock by lock bit data is disabled | ○ | ○ |
| FMR03 | Flash memory reset bit (Note 3) | 0: Normal operation<br>1: Reset | ○ | ○ |
| Reserved bit | | Must always be set to “0” | ○ | ○ |
| FMR05 | User ROM area select bit ( Note 4) (Effective in only boot mode) | 0: Boot ROM area is accessed<br>1: User ROM area is accessed | ○ | ○ |
| ——— | Noting is assigned. When write, set to "0". When read, their contents are indeterminate. | | — | — |

Note 1: For this bit to be set to “1”, the user needs to write a “0” and then a “1” to it in succession. When it is not this procedure, it is not enacted in “1”. This is necessary to ensure that no interrupt or DMA transfer will be executed during the interval. Use the control program except in the internal flash memory for write to this bit. Also write to this bit when NMI pin is "H" level.
Note 2: For this bit to be set to “1”, the user needs to write a “0” and then a “1” to it in succession when the CPU rewrite mode select bit = “1”. When it is not this procedure, it is not enacted in “1”. This is necessary to ensure that no interrupt or DMA transfer will be executed during the interval.
Note 3: Effective only when the CPU rewrite mode select bit = 1. Set this bit to 0 subsequently after setting it to 1 (reset).
Note 4: Use the control program except in the internal flash memory for write to this bit.

**Figure 1.34.1. Flash memory control register**

## Flash memory control register (address $0057_{16}$)

Bit 0 of the flash memory control register 0 is the RY/$\overline{BY}$ signal status bit used exclusively to read the operating status of the flash memory. During programming and erase operations, it is “0”. Otherwise, it is “1”.

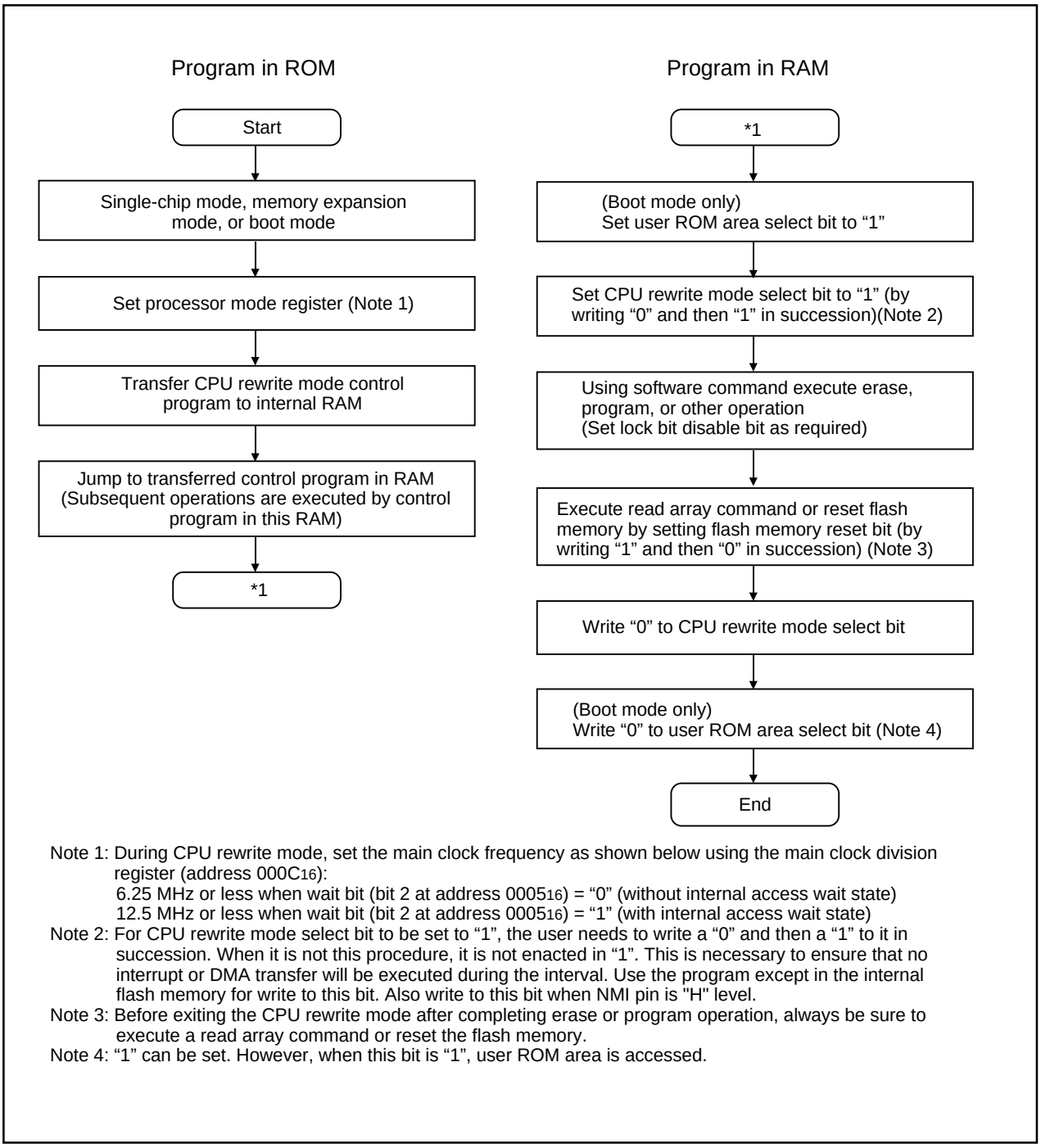
Bit 1 of the flash memory control register 0 is the CPU rewrite mode select bit. The CPU rewrite mode is entered by setting this bit to “1”, so that software commands become acceptable. In CPU rewrite mode, the CPU becomes unable to access the internal flash memory directly. Therefore, write bit 1 in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

Bit 2 of the flash memory control register 0 is a lock bit disable bit. By setting this bit to “1”, it is possible to disable erase and write protect (block lock) effectuated by the lock bit data. The lock bit disable select bit only disables the lock bit function; it does not change the lock data bit value. However, if an erase operation is performed when this bit =“1”, the lock bit data that is “0” (locked) is set to “1” (unlocked) after erasure. To set this bit to “1”, it is necessary to write “0” and then write “1” in succession. This bit can be manipulated only when the CPU rewrite mode select bit = “1”.

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

Bit 5 of the flash memory control register 0 is a user ROM area select bit which is effective in only boot mode. If this bit is set to “1” in boot mode, the area to be accessed is switched from the boot ROM area to the user ROM area. When the CPU rewrite mode needs to be used in boot mode, set this bit to “1”. Note that if the microcomputer is booted from the user ROM area, it is always the user ROM area that can be accessed and this bit has no effect. When in boot mode, the function of this bit is effective regardless of whether the CPU rewrite mode is on or off. Use the control program except in the internal flash memory to rewrite this bit.

Figure 1.34.2 shows a flowchart for setting/releasing the CPU rewrite mode. Always perform operation as indicated in these flowcharts.




Note 1: During CPU rewrite mode, set the main clock frequency as shown below using the main clock division register (address 000C16):
6.25 MHz or less when wait bit (bit 2 at address 000516) = “0” (without internal access wait state)
12.5 MHz or less when wait bit (bit 2 at address 000516) = “1” (with internal access wait state)
Note 2: For CPU rewrite mode select bit to be set to “1”, the user needs to write a “0” and then a “1” to it in succession. When it is not this procedure, it is not enacted in “1”. This is necessary to ensure that no interrupt or DMA transfer will be executed during the interval. Use the program except in the internal flash memory for write to this bit. Also write to this bit when NMI pin is "H" level.
Note 3: Before exiting the CPU rewrite mode after completing erase or program operation, always be sure to execute a read array command or reset the flash memory.
Note 4: “1” can be set. However, when this bit is “1”, user ROM area is accessed.

**Figure 1.34.2. CPU rewrite mode set/reset flowchart**

## Precautions on CPU Rewrite Mode

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

**(1) Operation speed**

During CPU rewrite mode, set the main clock frequency as shown below using the main clock division register (address $000C_{16}$):

6.25 MHz or less when wait bit (bit 2 at address $0005_{16}$) = 0 (without internal access wait state)

12.5 MHz or less when wait bit (bit 2 at address $0005_{16}$) = 1 (with internal access wait state)

**(2) Instructions inhibited against use**

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

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

**(3) Interrupts inhibited against use**

The address match interrupt cannot be used during CPU rewrite mode because they refer to the internal data of the flash memory. If interrupts have their vector in the variable vector table, they can be used by transferring the vector into the RAM area. The $\overline{\text{NMI}}$ and watchdog timer interrupts each can be used to change the CPU rewrite mode select bit forcibly to normal mode (FMR01="0") upon occurrence of the interrupt. Since the rewrite operation is halted when the $\overline{\text{NMI}}$ and watchdog timer interrupts occur, set the CPU rewite mode select bit to "1" and the erase/program operation needs to be performed over again.

**(4) Reset**

Reset input is always accepted.

**(5) Access disable**

Write CPU rewrite mode select bit and user ROM area select bit in an area other than the internal flash memory.

**(6) How to access**

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

Write to the CPU rewrite mode select bit when NMI pin is "H" level.

**(7)Writing in the user ROM area**

If power is lost while rewriting blocks that contain the flash rewrite program with the CPU rewrite mode, those blocks may not be correctly rewritten and it is possible that the flash memory can no longer be rewritten after that. Therefore, it is recommended to use the standard serial I/O mode or parallel I/O mode to rewrite these blocks.

**(8)Using the lock bit**

To use the CPU rewrite mode, use a boot program that can set and cancel the lock command.

## Software Commands

Table 1.34.1 lists the software commands available with the M16C/62A (flash memory version).

After setting the CPU rewrite mode select bit to 1, write a software command to specify an erase or program operation. Note that when entering a software command, the upper byte ($D_8$ to $D_{15}$) is ignored.

The content of each software command is explained below.

**Table 1.34.1. List of software commands (CPU rewrite mode)(Note 1)**

| Command | First bus cycle | | | Second bus cycle | | | Third bus cycle | | |
|---|---|---|---|---|---|---|---|---|---|
| | Mode | Address | Data ($D_0$ to $D_7$) | Mode | Address | Data ($D_0$ to $D_7$) | Mode | Address | Data ($D_0$ to $D_7$) |
| Read array | Write | X (Note 6) | $FF_{16}$ | | | | | | |
| Read status register | Write | X | $70_{16}$ | Read | X | SRD (Note 2) | | | |
| Clear status register | Write | X | $50_{16}$ | | | | | | |
| Page program (Note 3) | Write | X | $41_{16}$ | Write | WA0 (Note 3) | WD0 (Note 3) | Write | WA1 | WD1 |
| Block erase | Write | X | $20_{16}$ | Write | BA (Note 4) | $D0_{16}$ | | | |
| Erase all unlock block | Write | X | $A7_{16}$ | Write | X | $D0_{16}$ | | | |
| Lock bit program | Write | X | $77_{16}$ | Write | BA | $D0_{16}$ | | | |
| Read lock bit status | Write | X | $71_{16}$ | Read | BA | $D_6$ (Note 5) | | | |

Note 1: When a software command is input, the high-order byte of data ($D_8$ to $D_{15}$) is ignored.
Note 2: SRD = Status Register Data
Note 3: WA = Write Address, WD = Write Data
WA and WD must be set sequentially from $00_{16}$ to $FE_{16}$ (byte address; however, an even address). The page size is 256 bytes.
Note 4: BA = Block Address (Enter the maximum address of each block that is an even address.)
Note 5: $D_6$ corresponds to the block lock status. Block not locked when $D_6$ = 1, block locked when $D_6$ = 0.
Note 6: X denotes a given address in the user ROM area (that is an even address).

### Read Array Command ($FF_{16}$)

The read array mode is entered by writing the command code "$FF_{16}$" in the first bus cycle. When an even address to be read is input in one of the bus cycles that follow, the content of the specified address is read out at the data bus ($D_0$–$D_{15}$), 16 bits at a time.

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

### Read Status Register Command ($70_{16}$)

When the command code "$70_{16}$" is written in the first bus cycle, the content of the status register is read out at the data bus ($D_0$–$D_7$) by a read in the second bus cycle.

The status register is explained in the next section.

### Clear Status Register Command ($50_{16}$)

This command is used to clear the bits SR3 to 5 of the status register after they have been set. These bits indicate that operation has ended in an error. To use this command, write the command code "$50_{16}$" in the first bus cycle.

**Page Program Command (41$_{16}$)**

Page program allows for high-speed programming in units of 256 bytes. Page program operation starts when the command code "41$_{16}$" is written in the first bus cycle. In the second bus cycle through the 129th bus cycle, the write data is sequentially written 16 bits at a time. At this time, the addresses A$_0$-A$_7$ need to be incremented by 2 from "00$_{16}$" to "FE$_{16}$." When the system finishes loading the data, it starts an auto write operation (data program and verify operation).

Whether the auto write operation is completed can be confirmed by reading the status register or the flash memory control register 0. At the same time the auto write operation starts, the read status register mode is automatically entered, so the content of the status register can be read out. The status register bit 7 (SR7) is set to 0 at the same time the auto write operation starts and is returned to 1 upon completion of the auto write operation. In this case, the read status register mode remains active until the Read Array command (FF$_{16}$) or Read Lock Bit Status command (71$_{16}$) is written or the flash memory is reset using its reset bit.

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

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

Figure 1.34.3 shows an example of a page program flowchart.

Each block of the flash memory can be write protected by using a lock bit. For details, refer to the section where the data protect function is detailed.

Additional writes to the already programmed pages are prohibited.

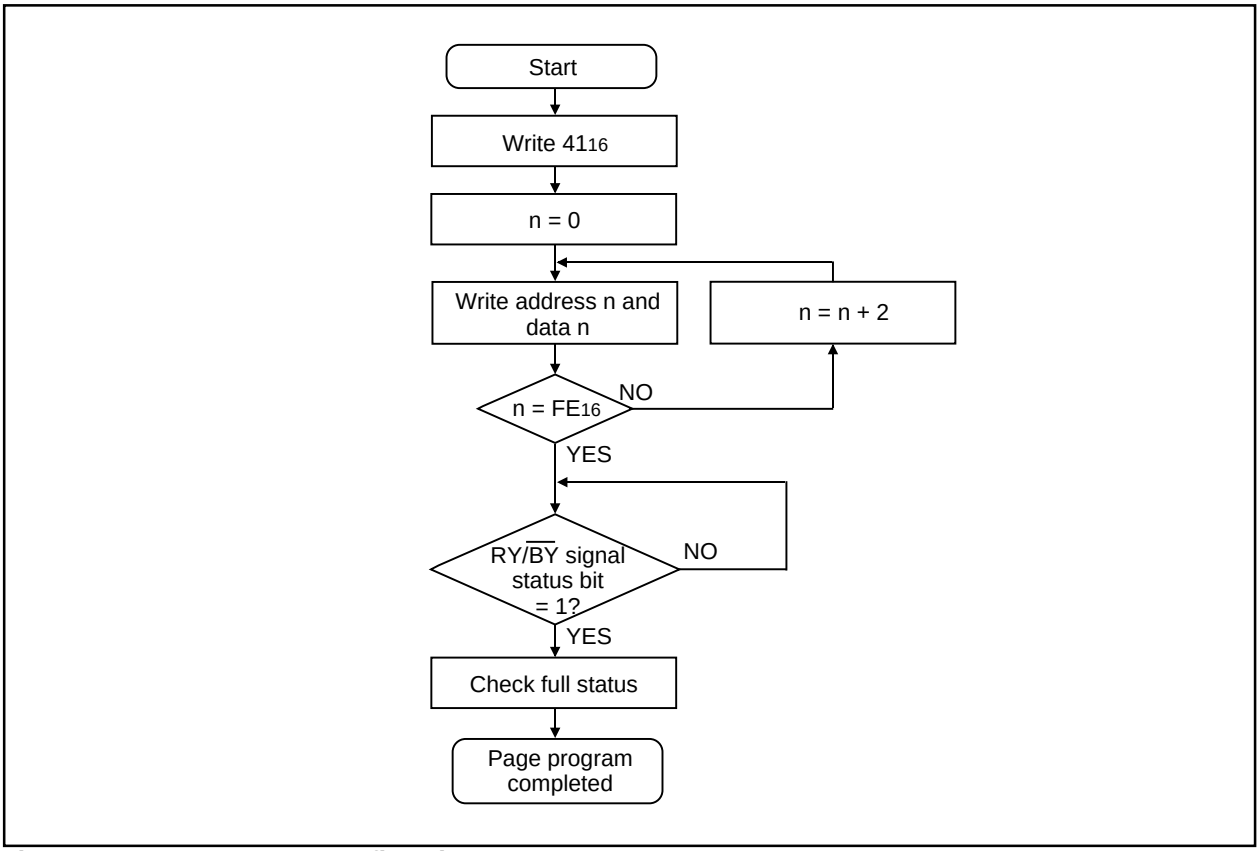



**Figure 1.34.3. Page program flowchart**

**Block Erase Command ($20_{16}$/$D0_{16}$)**

By writing the command code "$20_{16}$" in the first bus cycle and the confirmation command code "$D0_{16}$" in the second bus cycle that follows to the block address of a flash memory block, the system initiates an auto erase (erase and erase verify) operation.

Whether the auto erase operation is completed can be confirmed by reading the status register or the flash memory control register 0. At the same time the auto erase operation starts, the read status register mode is automatically entered, so the content of the status register can be read out. The status register bit 7 (SR7) is set to 0 at the same time the auto erase operation starts and is returned to 1 upon completion of the auto erase operation. In this case, the read status register mode remains active until the Read Array command ($FF_{16}$) or Read Lock Bit Status command ($71_{16}$) is written or the flash memory is reset using its reset bit.

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

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

Figure 1.34.4 shows an example of a block erase flowchart.

Each block of the flash memory can be protected against erasure by using a lock bit. For details, refer to the section where the data protect function is detailed.

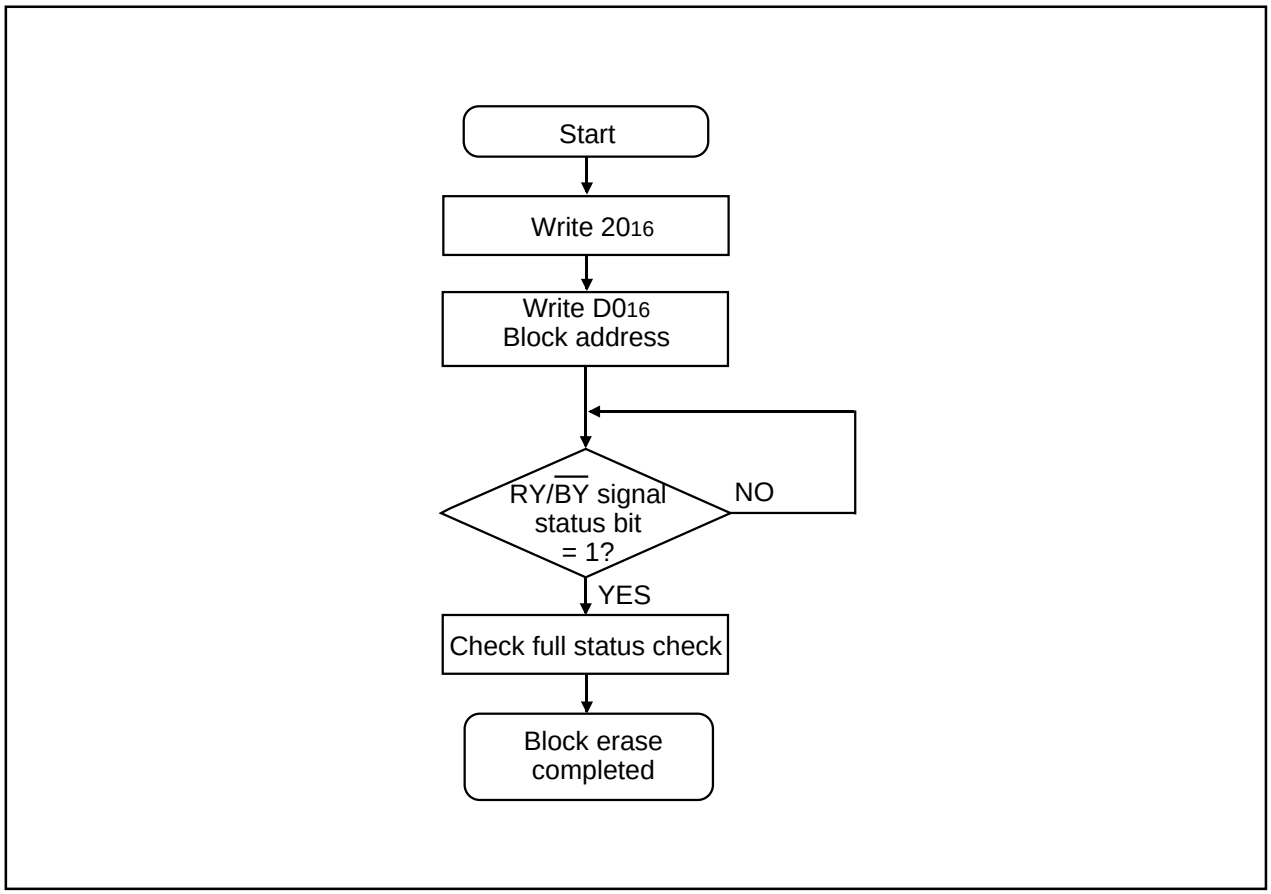



**Figure 1.34.4. Block erase flowchart**

**Erase All Unlock Blocks Command ($A7_{16}/D0_{16}$)**

By writing the command code "$A7_{16}$" in the first bus cycle and the confirmation command code "$D0_{16}$" in the second bus cycle that follows, the system starts erasing blocks successively.

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

When the lock bit disable bit of the flash memory control register 0 = 1, all blocks are erased no matter how the lock bit is set. On the other hand, when the lock bit disable bit = 0, the function of the lock bit is effective and only nonlocked blocks (where lock bit data = 1) are erased.

**Lock Bit Program Command ($77_{16}/D0_{16}$)**

By writing the command code "$77_{16}$" in the first bus cycle and the confirmation command code "$D0_{16}$" in the second bus cycle that follows to the block address of a flash memory block, the system sets the lock bit for the specified block to 0 (locked).

Figure 1.34.5 shows an example of a lock bit program flowchart. The status of the lock bit (lock bit data) can be read out by a read lock bit status command.

Whether the lock bit program command is terminated can be confirmed by reading the status register or the flash memory control register 0, in the same way as for page program.

For details about the function of the lock bit and how to reset the lock bit, refer to the section where the data protect function is detailed.

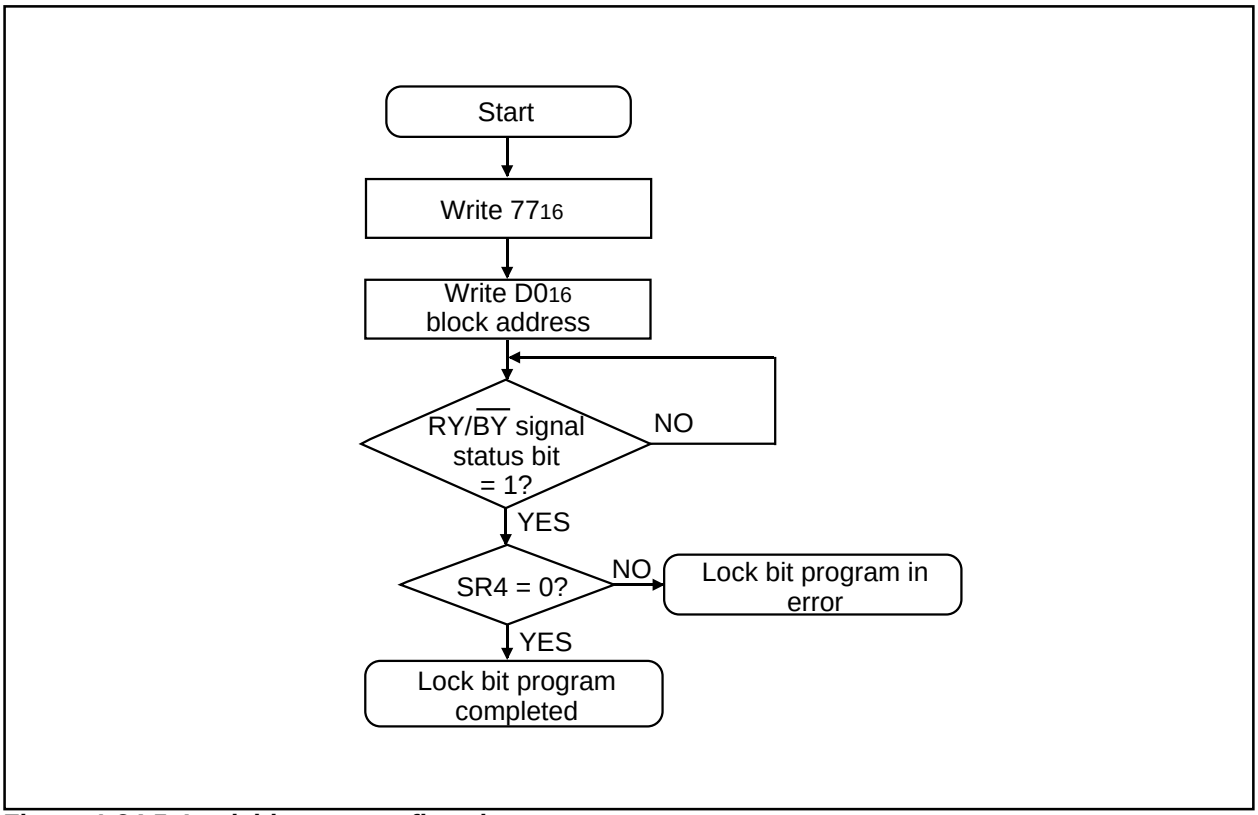



**Figure 1.34.5. Lock bit program flowchart**

**Read Lock Bit Status Command (7116)**

By writing the command code "7116" in the first bus cycle and then the block address of a flash memory block in the second bus cycle that follows, the system reads out the status of the lock bit of the specified block on to the data (D6).

Figure 1.34.6 shows an example of a read lock bit program flowchart.

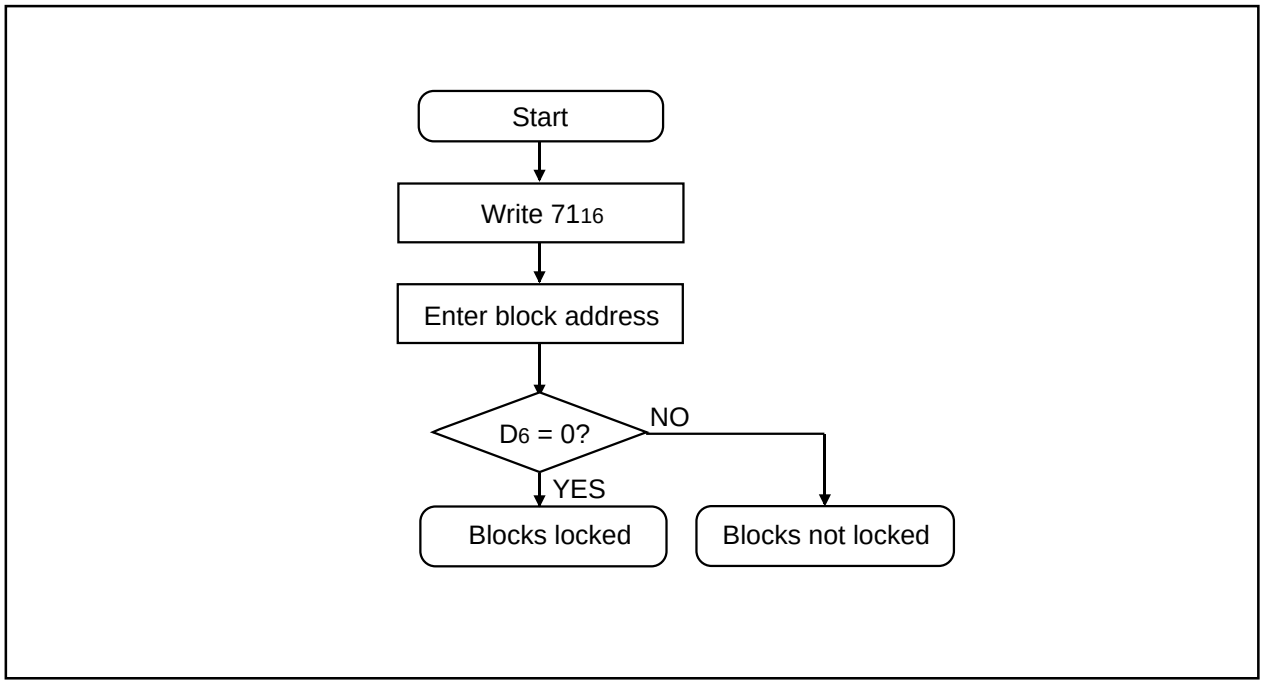



**Figure 1.34.6. Read lock bit status flowchart**

## Data Protect Function (Block Lock)

Each block in Figure 1.33.3 has a nonvolatile lock bit to specify that the block be protected (locked) against erase/write. The lock bit program command is used to set the lock bit to 0 (locked). The lock bit of each block can be read out using the read lock bit status command.

Whether block lock is enabled or disabled is determined by the status of the lock bit and how the flash memory control register 0's lock bit disable bit is set.

(1) When the lock bit disable bit = 0, a specified block can be locked or unlocked by the lock bit status (lock bit data). Blocks whose lock bit data = 0 are locked, so they are disabled against erase/write. On the other hand, the blocks whose lock bit data = 1 are not locked, so they are enabled for erase/write.

(2) When the lock bit disable bit = 1, all blocks are nonlocked regardless of the lock bit data, so they are enabled for erase/write. In this case, the lock bit data that is 0 (locked) is set to 1 (nonlocked) after erasure, so that the lock bit-actuated lock is removed.

## Status Register

The status register indicates the operating status of the flash memory and whether an erase or program operation has terminated normally or in an error. The content of this register can be read out by only writing the read status register command ($70_{16}$). Table 1.34.2 details the status register.

The status register is cleared by writing the Clear Status Register command ($50_{16}$).

After a reset, the status register is set to "$80_{16}$."

Each bit in this register is explained below.

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

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

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

**Erase status (SR5)**

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

The erase status is reset to 0 when cleared.

**Program status (SR4)**

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

The program status is reset to 0 when cleared.

When an erase command is in error (which occurs if the command entered after the block erase command ($20_{16}$) is not the confirmation command ($D0_{16}$), both the program status and erase status (SR5) are set to 1.

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

Also, in one of the following cases, both SR4 and SR5 are set to 1 (command sequence error):

(1) When the valid command is not entered correctly

(2) When the data entered in the second bus cycle of lock bit program ($77_{16}$/$D0_{16}$), block erase ($20_{16}$/$D0_{16}$), or erase all unlock blocks ($A7_{16}$/$D0_{16}$) is not the $D0_{16}$ or $FF_{16}$. However, if $FF_{16}$ is entered, read array is assumed and the command that has been set up in the first bus cycle is canceled.

**Block status after program (SR3)**

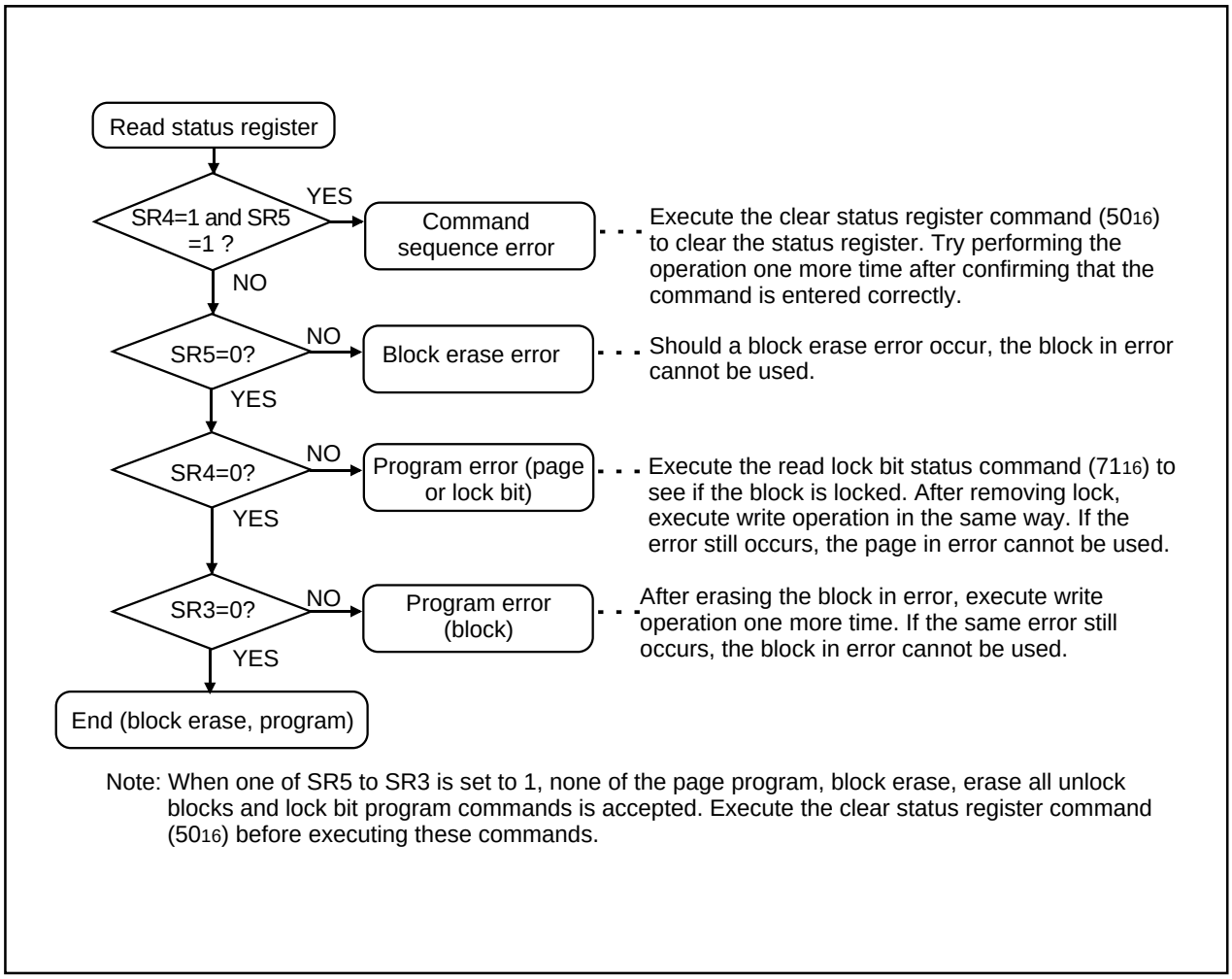
If excessive data is written (phenomenon whereby the memory cell becomes depressed which results in data not being read correctly), “1” is set for the program status after-program at the end of the page write operation. In other words, when writing ends successfully, “$80_{16}$” is output; when writing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

**Table 1.34.2. Definition of each bit in status register**

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

**Full Status Check**

By performing full status check, it is possible to know the execution results of erase and program operations. Figure 1.34.7 shows a full status check flowchart and the action to be taken when each error occurs.




Note: When one of SR5 to SR3 is set to 1, none of the page program, block erase, erase all unlock blocks and lock bit program commands is accepted. Execute the clear status register command ($50_{16}$) before executing these commands.

**Figure 1.34.7. Full status check flowchart and remedial procedure for errors**

## Parallel I/O Mode

In this mode, the M32C/83 (flash memory version) operates in a manner similar to the flash memory M5M29FB/T800 from Mitsubishi. Since there are some differences with regard to the functions not available with the microcomputer and matters related to memory capacity, the M32C/83 cannot be programed by a programer for the flash memory.

Use an exclusive programer supporting M32C/83 (flash memory version).

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

## User ROM and Boot ROM Areas

In parallel I/O mode, the user ROM and boot ROM areas shown in Figure 1.33.3 can be rewritten. Both areas of flash memory can be operated on in the same way.

Program and block erase operations can be performed in the user ROM area. The user ROM area and its blocks are shown in Figure 1.33.3.

The boot ROM area is 8 Kbytes in size. In parallel I/O mode, it is located at addresses $0FFE000_{16}$ through $0FFFFFF_{16}$. Make sure program and block erase operations are always performed within this address range. (Access to any location outside this address range is prohibited.)

In the boot ROM area, an erase block operation is applied to only one 8 Kbyte block. The boot ROM area has had a standard serial I/O mode control program stored in it when shipped from the Mitsubishi factory. Therefore, using the device in standard serial input/output mode, you do not need to write to the boot ROM area.

## Standard serial I/O mode

The standard serial I/O mode inputs and outputs the software commands, addresses and data needed to operate (read, program, erase, etc.) the internal flash memory. This I/O is serial. There are actually two standard serial I/O modes: mode 1, which is clock synchronized, and mode 2, which is asynchronized. Both modes require a purpose-specific peripheral unit.

The standard serial I/O mode is different from the parallel I/O mode in that the CPU controls flash memory rewrite (uses the CPU's rewrite mode), rewrite data input and so forth. It is started when the reset is released, which is done when the $P5_0$ ($\overline{CE}$) pin is "H" level, the $P5_5$ ($\overline{EPM}$) pin "L" level and the CNVss pin "H" level. (In the ordinary command mode, set CNVss pin to "L" level.)
This control program is written in the boot ROM area when the product is shipped from Mitsubishi. Accordingly, make note of the fact that the standard serial I/O mode cannot be used if the boot ROM area is rewritten in the parallel I/O mode. Figures 1.35.1 to 1.35.3 show the pin connections for the standard serial I/O mode. Serial data I/O uses UART1 and transfers the data serially in 8-bit units. Standard serial I/O switches between mode 1 (clock synchronized) and mode 2 (clock asynchronized) according to the level of $CLK_1$ pin when the reset is released.

To use standard serial I/O mode 1 (clock synchronized), set the $CLK_1$ pin to "H" level and the $TxD_1$ pin to "L" level, and release the reset. The $CLK_1$ pin is connected to Vcc via pull-up resistance and the $TxD_1$ is connected to Vss via pull-down resistance. The operation uses the four UART1 pins $CLK_1$, $RxD_1$, $TxD_1$ and $RTS_1$ (BUSY). The $CLK_1$ pin is the transfer clock input pin through which an external transfer clock is input. The $TxD_1$ pin is for CMOS output. The $RTS_1$ (BUSY) pin outputs an "L" level when ready for reception and an "H" level when reception starts.

To use standard serial I/O mode 2 (clock asynchronized), set the $CLK_1$ pin to "L" level and release the reset. The operation uses the two UART1 pins $RxD_1$ and $TxD_1$.

In the standard serial I/O mode, only the user ROM area indicated in Figure 1.35.20 can be rewritten. The boot ROM cannot.
In the standard serial I/O mode, a 7-byte ID code is used. When there is data in the flash memory, commands sent from the peripheral unit (programmer) are not accepted unless the ID code matches.

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

| Pin | Name | I/O | Description |
|---|---|---|---|
| VCC,VSS | Power input | | Apply 4.2V to 5.5V to Vcc pin and 0 V to Vss pin. |
| CNVSS | CNVSS | I | Connect to Vcc pin. |
| $\overline{\text{RESET}}$ | Reset input | I | Reset input pin. While reset is "L" level, a 20 cycle or longer clock must be input to XIN pin. |
| XIN | Clock input | I | Connect a ceramic resonator or crystal oscillator between XIN and XOUT pins. To input an externally generated clock, input it to XIN pin and open XOUT pin. |
| XOUT | Clock output | O | |
| BYTE | BYTE | I | Connect this pin to Vcc or Vss. |
| AVCC, AVSS | Analog power supply input | I | Connect AVSS to Vss and AVcc to Vcc, respectively. |
| VREF | Reference voltage input | I | Enter the reference voltage for A-D converter from this pin. |
| P00 to P07 | Input port P0 | I | Input "H" or "L" level signal or open. |
| P10 to P17 | Input port P1 | I | Input "H" or "L" level signal or open. |
| P20 to P27 | Input port P2 | I | Input "H" or "L" level signal or open. |
| P30 to P37 | Input port P3 | I | Input "H" or "L" level signal or open. |
| P40 to P47 | Input port P4 | I | Input "H" or "L" level signal or open. |
| P51 to P54, P56, P57 | Input port P5 | I | Input "H" or "L" level signal or open. |
| P50 | $\overline{\text{CE}}$ input | I | Input "H" level signal. |
| P55 | $\overline{\text{EPM}}$ input | I | Input "L" level signal. |
| P60 to P63 | Input port P6 | I | Input "H" or "L" level signal or open. |
| P64 | BUSY output | O | Standard serial mode 1: BUSY signal output pin<br>Standard serial mode 2: Monitors the program operation check |
| P65 | SCLK input | I | Standard serial mode 1: Serial clock input pin<br>Standard serial mode 2: Input "L" level signal. |
| P66 | RxD input | I | Serial data input pin |
| P67 | TxD output | O | Serial data output pin. When using standard serial mode 1, an "L" level must be input to TxD pin while the $\overline{\text{RESET}}$ pin is “L”. For this reason, this pin should be pulled down. After being reset, this pin functions as a data output pin. Thus adjust pull-down resistance value with the system not to affect data transfer. |
| P70 to P77 | Input port P7 | I | Input "H" or "L" level signal or open. |
| P80 to P84, P86, P87 | Input port P8 | I | Input "H" or "L" level signal or open. |
| P85 | $\overline{\text{NMI}}$ input | I | Connect this pin to Vcc. |
| P90 to P97 | Input port P9 | I | Input "H" or "L" level signal or open. |
| P100 to P107 | Input port P10 | I | Input "H" or "L" level signal or open. |
| P110 to P114 | Input port P11 | I | Input "H" or "L" level signal or open. (Note) |
| P120 to P127 | Input port P12 | I | Input "H" or "L" level signal or open. (Note) |
| P130 to P137 | Input port P13 | I | Input "H" or "L" level signal or open. (Note) |
| P140 to P146 | Input port P14 | I | Input "H" or "L" level signal or open. (Note) |
| P150 to P157 | Input port P15 | I | Input "H" or "L" level signal or open. (Note) |

Note: Port P11 to P15 exist in 144-pin version.

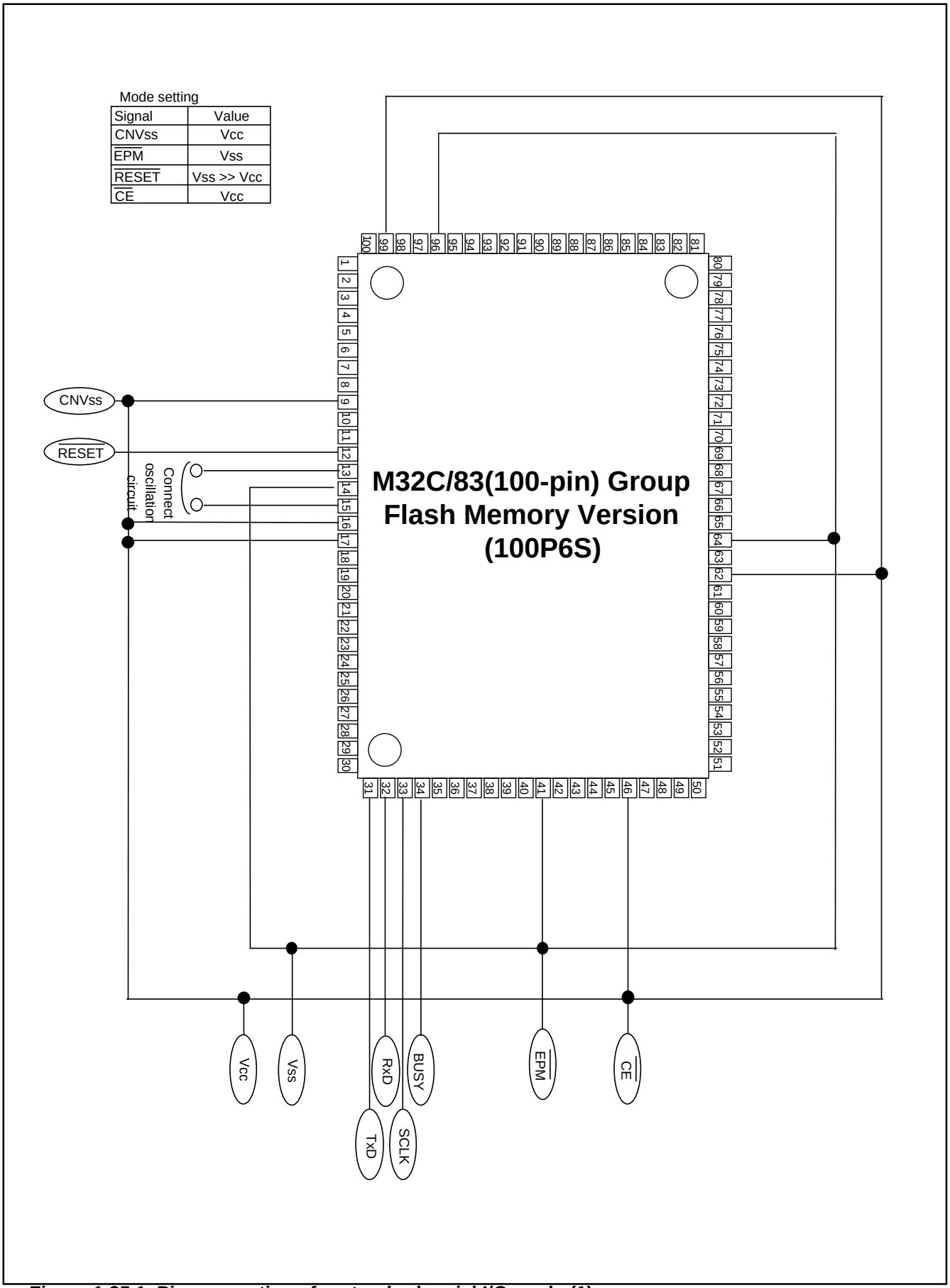



**Figure 1.35.1. Pin connections for standard serial I/O mode (1)**

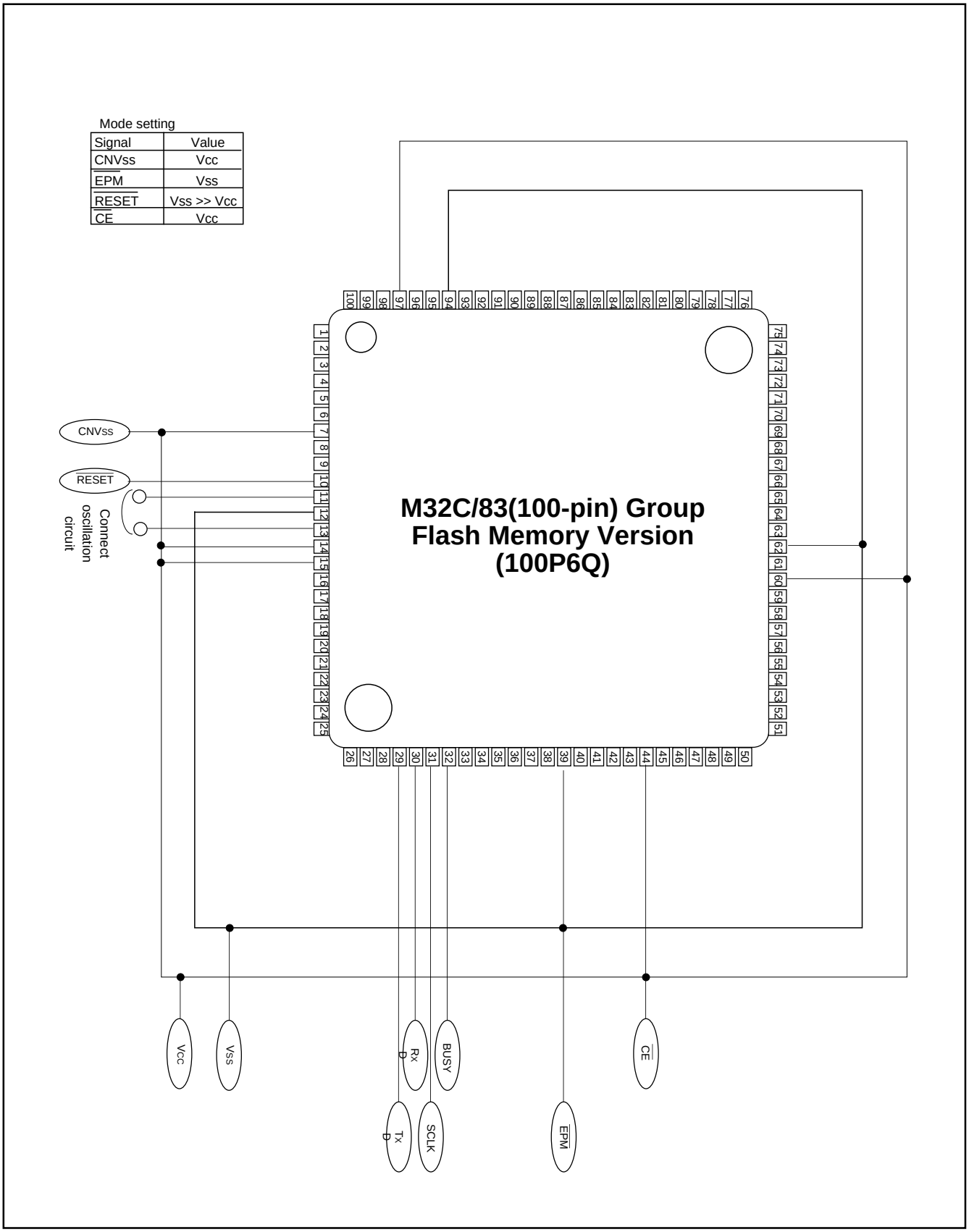



**Figure 1.35.2. Pin connections for standard serial I/O mode (2)**

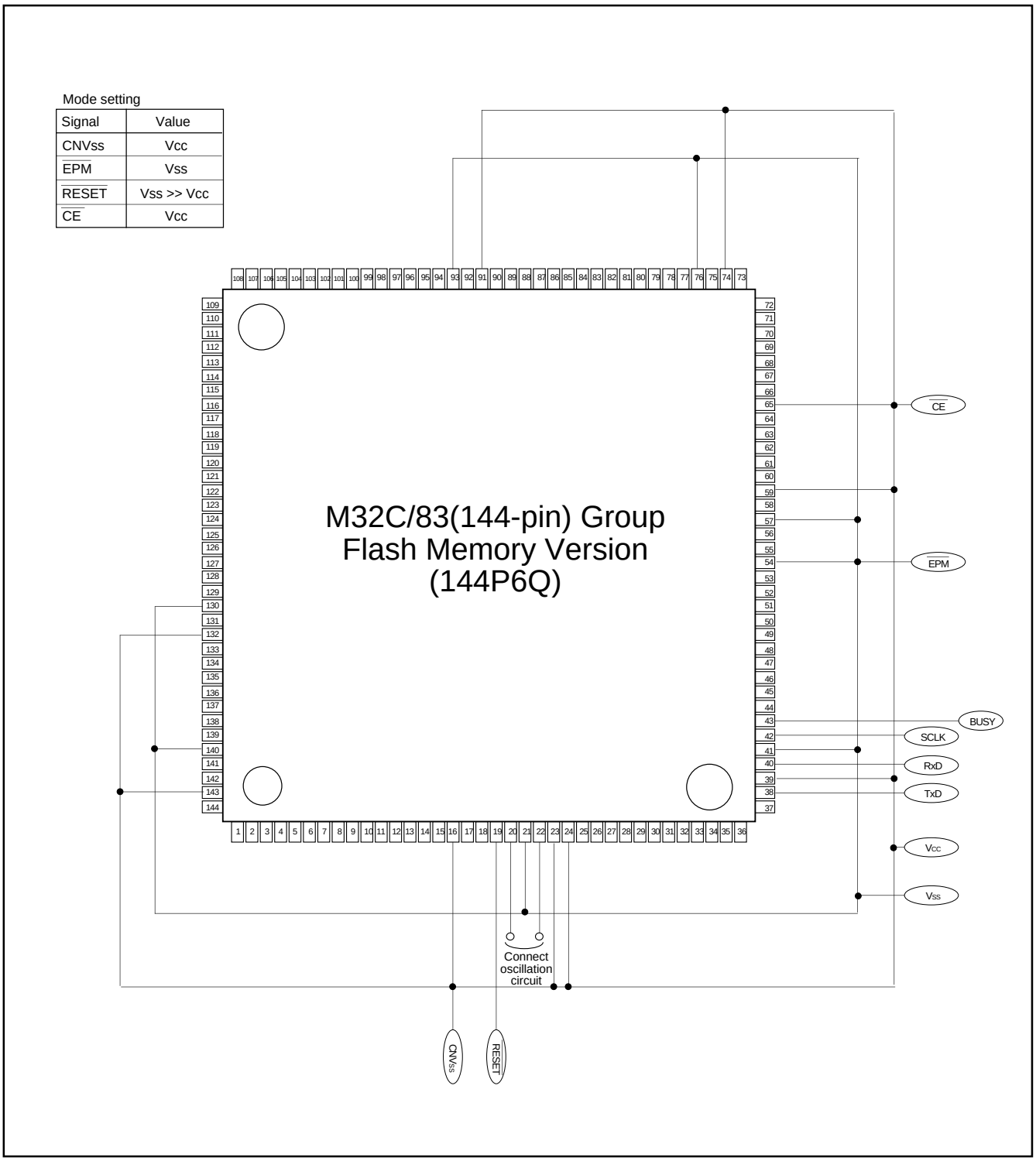



**Figure 1.35.3. Pin connections for standard serial I/O mode (3)**

## Overview of standard serial I/O mode 1 (clock synchronized)

In standard serial I/O mode 1, software commands, addresses and data are input and output between the MCU and peripheral units (serial programer, etc.) using 4-wire clock-synchronized serial I/O (UART1). Standard serial I/O mode 1 is engaged by releasing the reset with the P6$_5$ (CLK1) pin "H" level.

In reception, software commands, addresses and program data are synchronized with the rise of the transfer clock that is input to the CLK1 pin, and are then input to the MCU via the RxD1 pin. In transmission, the read data and status are synchronized with the fall of the transfer clock, and output from the TxD1 pin. The TxD1 pin is for CMOS output. Transfer is in 8-bit units with LSB first.

When busy, such as during transmission, reception, erasing or program execution, the RTS1 (BUSY) pin is "H" level. Accordingly, always start the next transfer after the RST1 (BUSY) pin is "L" level.

Also, data and status registers in memory can be read after inputting software commands. Status, such as the operating state of the flash memory or whether a program or erase operation ended successfully or not, can be checked by reading the status register. Here following are explained software commands, status registers, etc.

## Software Commands

Table 1.35.1 lists software commands. In the standard serial I/O mode 1, erase operations, programs and reading are controlled by transferring software commands via the $RxD_1$ pin. Software commands are explained here below.

**Table 1.35.1. Software commands (Standard serial I/O mode 1)**

| | Control command | 1st byte transfer | 2nd byte | 3rd byte | 4th byte | 5th byte | 6th byte | | When ID is not verified |
|---|---|---|---|---|---|---|---|---|---|
| 1 | Page read | $FF_{16}$ | Address (middle) | Address (high) | Data output | Data output | Data output | Data output to 259th byte | Not acceptable |
| 2 | Page program | $41_{16}$ | Address (middle) | Address (high) | Data input | Data input | Data input | Data input to 259th byte | Not acceptable |
| 3 | Block erase | $20_{16}$ | Address (middle) | Address (high) | $D0_{16}$ | | | | Not acceptable |
| 4 | Erase all unlocked blocks | $A7_{16}$ | $D0_{16}$ | | | | | | Not acceptable |
| 5 | Read status register | $70_{16}$ | SRD output | SRD1 output | | | | | Acceptable |
| 6 | Clear status register | $50_{16}$ | | | | | | | Not acceptable |
| 7 | Read lock bit status | $71_{16}$ | Address (middle) | Address (high) | Lock bit data output | | | | Not acceptable |
| 8 | Lock bit program | $77_{16}$ | Address (middle) | Address (high) | $D0_{16}$ | | | | Not acceptable |
| 9 | Lock bit enable | $7A_{16}$ | | | | | | | Not acceptable |
| 10 | Lock bit disable | $75_{16}$ | | | | | | | Not acceptable |
| 11 | Code processing function | $F5_{16}$ | Address (low) | Address (middle) | Address (high) | ID size | ID1 | To ID7 | Acceptable |
| 12 | Download function | $FA_{16}$ | Size (low) | Size (high) | Check-sum | Data input | To required number of times | | Not acceptable |
| 13 | Version data output function | $FB_{16}$ | Version data output | Version data output | Version data output | Version data output | Version data output | Version data output to 9th byte | Acceptable |
| 14 | Boot ROM area output function | $FC_{16}$ | Address (middle) | Address (high) | Data output | Data output | Data output | Data output to 259th byte | Not acceptable |
| 15 | Read check data | $FD_{16}$ | Check data (low) | Check data (high) | | | | | Not acceptable |

Note 1: Shading indicates transfer from flash memory microcomputer to peripheral unit. All other data is transferred from the peripheral unit to the flash memory microcomputer.

Note 2: SRD refers to status register data. SRD1 refers to status register data1 .

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

**Page Read Command**

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

(1) Transfer the "FF16" command code with the 1st byte.

(2) Transfer addresses A8 to A15 and A16 to A23 with the 2nd and 3rd bytes respectively.

(3) From the 4th byte onward, data (D0–D7) for the page (256 bytes) specified with addresses A8 to A23 will be output sequentially from the smallest address first in sync with the rise of the clock.

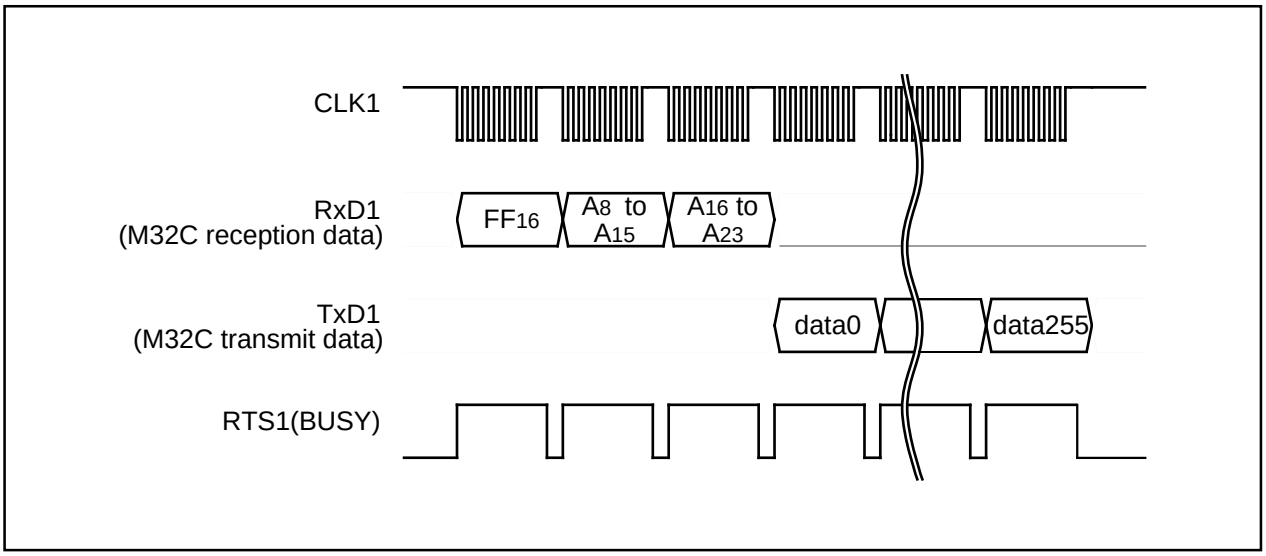



**Figure 1.35.4. Timing for page read**

**Page Program Command**

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

(1) Transfer the "4116" command code with the 1st byte.

(2) Transfer addresses A8 to A15 and A16 to A23 with the 2nd and 3rd bytes respectively.

(3) From the 4th byte onward, as write data (D0–D7) for the page (256 bytes) specified with addresses A8 to A23 is input sequentially from the smallest address first, that page is automatically written.

When reception setup for the next 256 bytes ends, the RTS1 (BUSY) signal changes from the "H" to the "L" level. The result of the page program can be known by reading the status register. For more information, see the section on the status register.

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

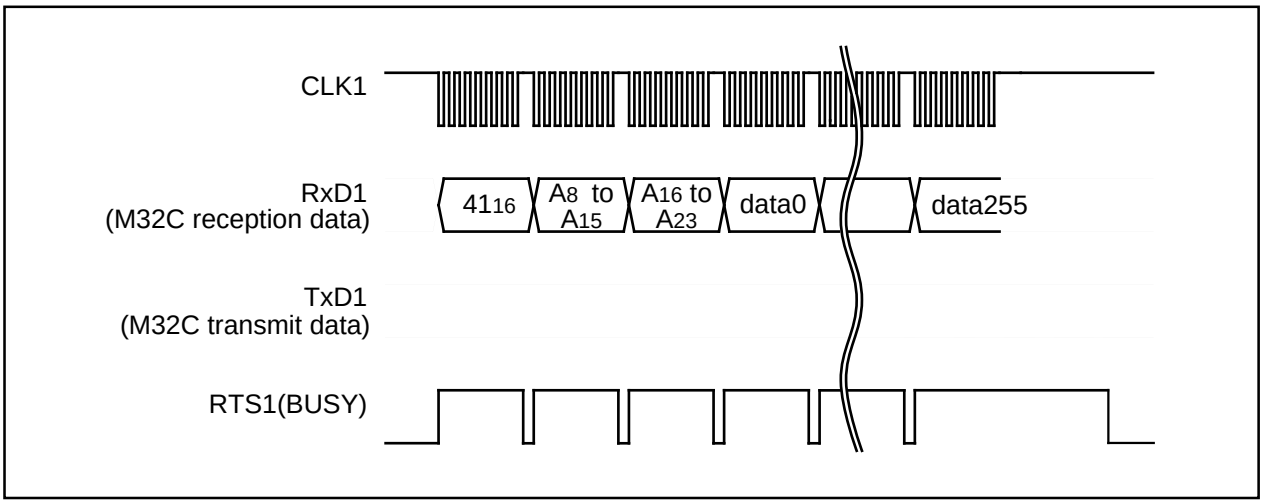



**Figure 1.35.5. Timing for the page program**

**Block Erase Command**

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

(1) Transfer the "$20_{16}$" command code with the 1st byte.
(2) Transfer addresses $A_8$ to $A_{15}$ and $A_{16}$ to $A_{23}$ with the 2nd and 3rd bytes respectively.
(3) Transfer the verify command code "$D0_{16}$" with the 4th byte. With the verify command code, the erase operation will start for the specified block in the flash memory. Write the highest address of the specified block for addresses $A_{16}$ to $A_{23}$.

When block erasing ends, the RTS1 (BUSY) signal changes from the "H" to the "L" level. After block erase ends, the result of the block erase operation can be known by reading the status register. For more information, see the section on the status register.

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

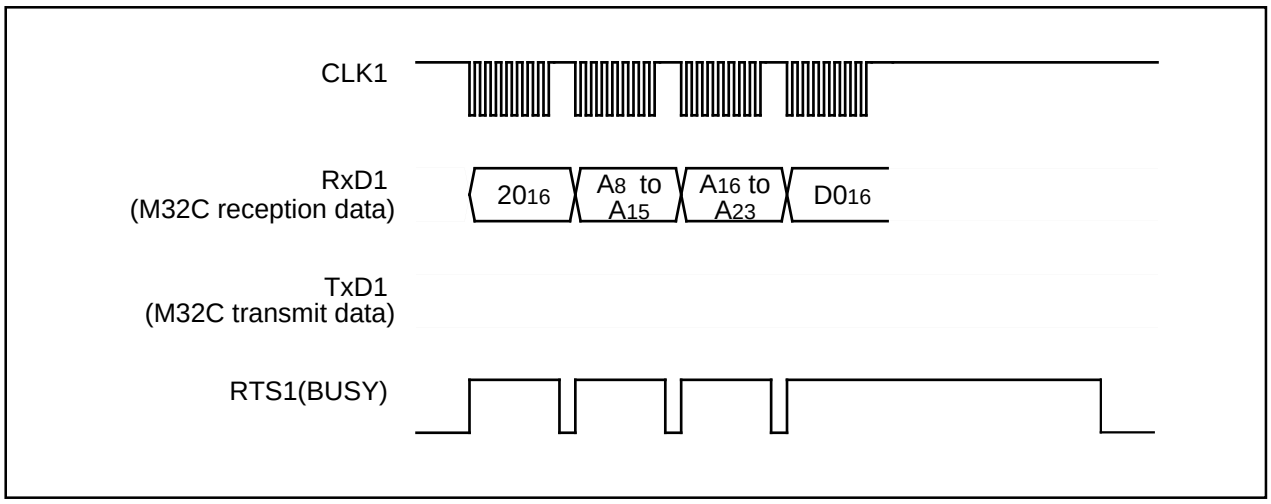



**Figure 1.35.6. Timing for block erasing**

**Erase All Unlocked Blocks Command**

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

(1) Transfer the "A7$_{16}$" command code with the 1st byte.

(2) Transfer the verify command code "D0$_{16}$" with the 2nd byte. With the verify command code, the erase operation will start and continue for all blocks in the flash memory.

When block erasing ends, the RTS1 (BUSY) signal changes from the "H" to the "L" level. The result of the erase operation can be known by reading the status register. Each block can be erase-protected with the lock bit. For more information, see the section on the data protection function.

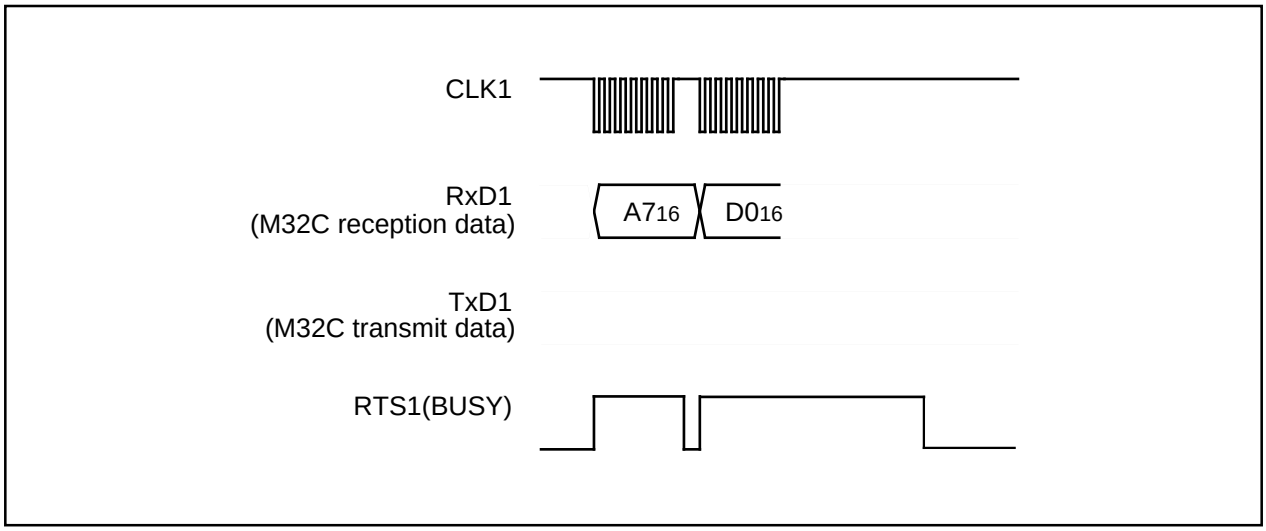



**Figure 1.35.7. Timing for erasing all unlocked blocks**

**Read Status Register Command**

This command reads status information. When the "70$_{16}$" command code is sent with the 1st byte, the contents of the status register (SRD) specified with the 2nd byte and the contents of status register 1 (SRD1) specified with the 3rd byte are read.

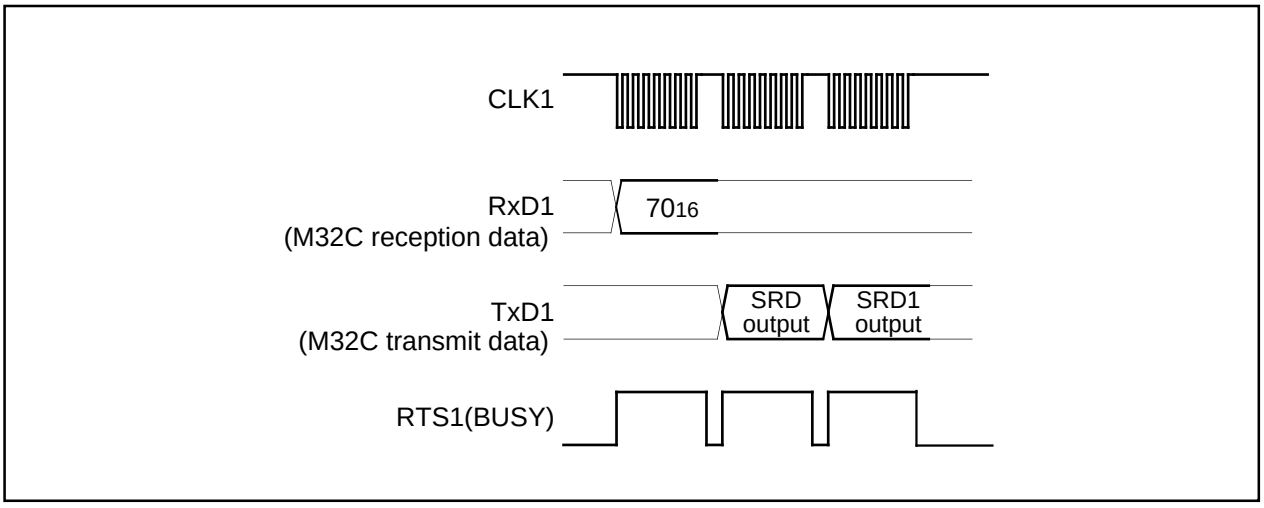



**Figure 1.35.8. Timing for reading the status register**

**Clear Status Register Command**

This command clears the bits (SR3–SR5) which are set when the status register operation ends in error. When the "$50_{16}$" command code is sent with the 1st byte, the aforementioned bits are cleared. When the clear status register operation ends, the $RTS_1$ (BUSY) signal changes from the "H" to the "L" level.

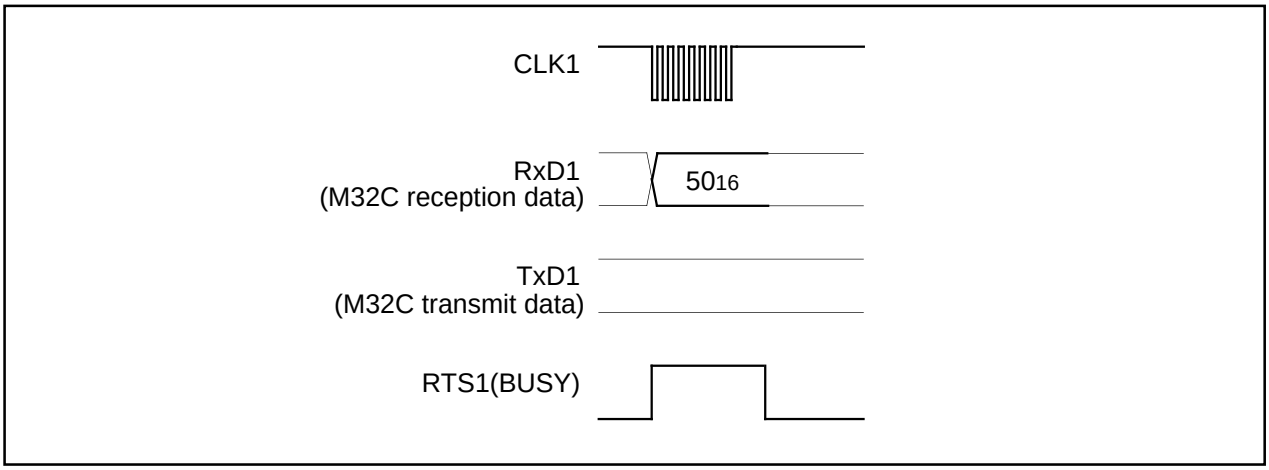



**Figure 1.35.9. Timing for clearing the status register**

**Read Lock Bit Status Command**

This command reads the lock bit status of the specified block. Execute the read lock bit status command as explained here following.

(1) Transfer the "$71_{16}$" command code with the 1st byte.
(2) Transfer addresses $A_8$ to $A_{15}$ and $A_{16}$ to $A_{23}$ with the 2nd and 3rd bytes respectively.
(3) The lock bit data of the specified block is output with the 4th byte. The 6th bit (D6) of output data is the lock bit data. Write the highest address of the specified block for addresses $A_8$ to $A_{23}$.

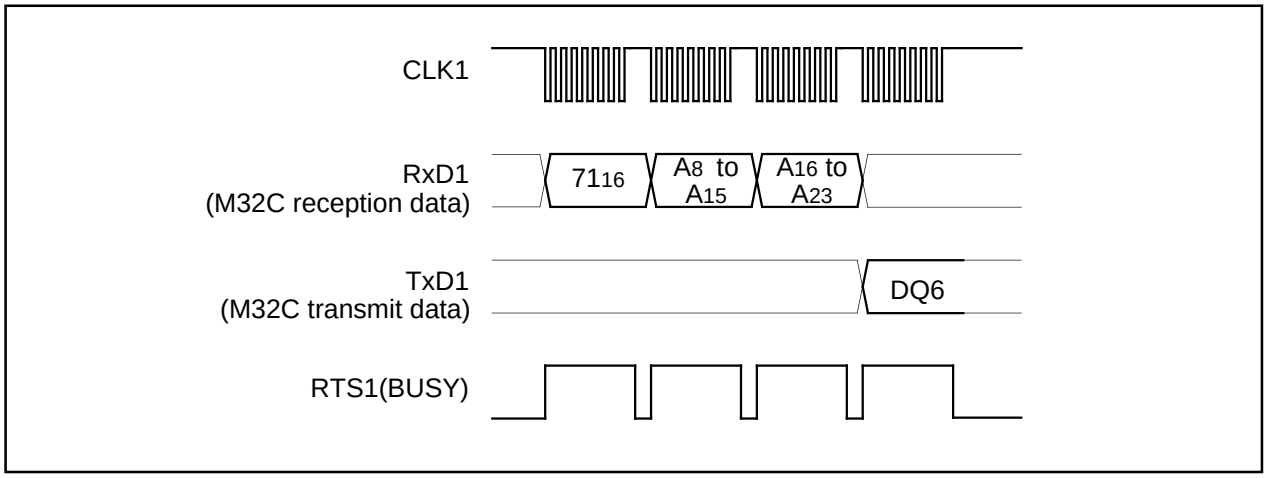



**Figure 1.35.10. Timing for reading lock bit status**

**Lock Bit Program Command**

This command writes “0” (lock) for the lock bit of the specified block. Execute the lock bit program command as explained here following.

(1) Transfer the “$77_{16}$” command code with the 1st byte.

(2) Transfer addresses $A_8$ to $A_{15}$ and $A_{16}$ to $A_{23}$ with the 2nd and 3rd bytes respectively.

(3) Transfer the verify command code “$D0_{16}$” with the 4th byte. With the verify command code, “0” is written for the lock bit of the specified block. Write the highest address of the specified block for addresses $A_8$ to $A_{23}$.

When writing ends, the $RTS_1$ (BUSY) signal changes from the “H” to the “L” level. Lock bit status can be read with the read lock bit status command. For information on the lock bit function, reset procedure and so on, see the section on the data protection function.

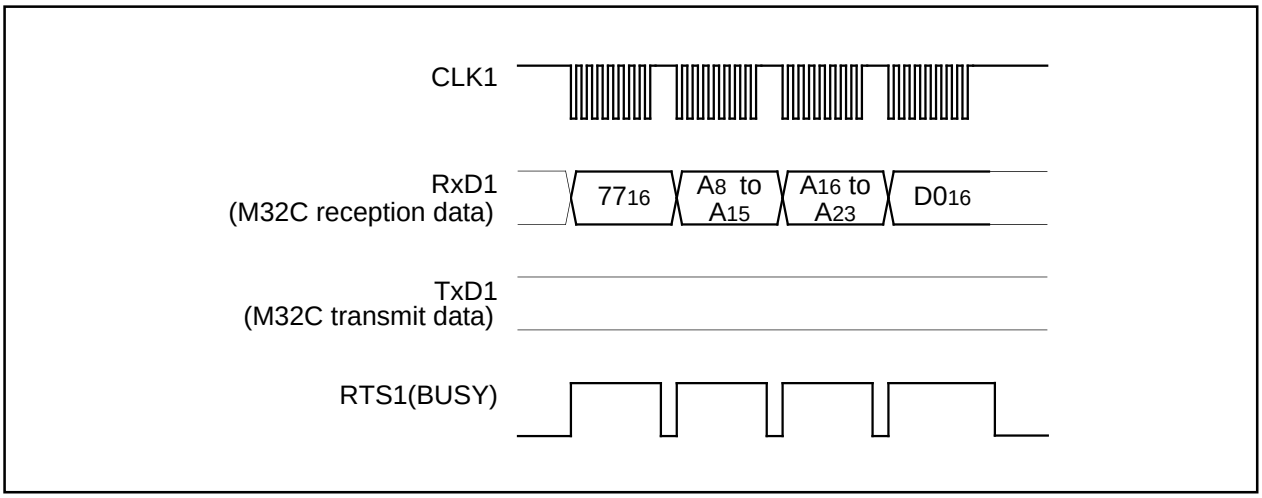



**Figure 1.35.11. Timing for the lock bit program**

**Lock Bit Enable Command**

This command enables the lock bit in blocks whose bit was disabled with the lock bit disable command. The command code “$7A_{16}$” is sent with the 1st byte of the serial transmission. This command only enables the lock bit function; it does not set the lock bit itself.

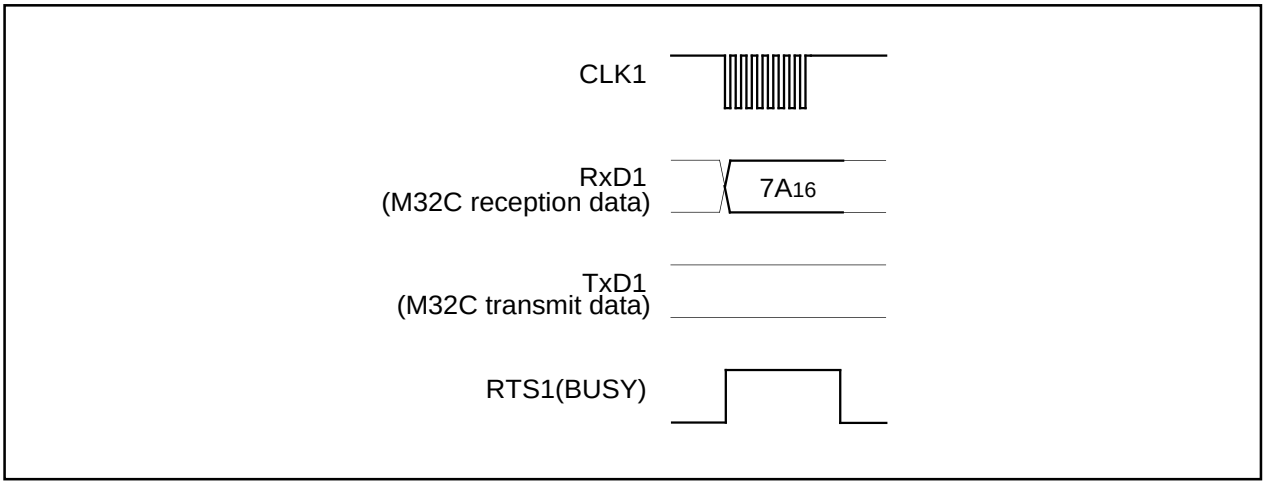



**Figure 1.35.12. Timing for enabling the lock bit**

**Lock Bit Disable Command**

This command disables the lock bit. The command code "$75_{16}$" is sent with the 1st byte of the serial transmission. This command only disables the lock bit function; it does not set the lock bit itself. However, if an erase command is executed after executing the lock bit disable command, "0" (locked) lock bit data is set to "1" (unlocked) after the erase operation ends. In any case, after the reset is cancelled, the lock bit is enabled.

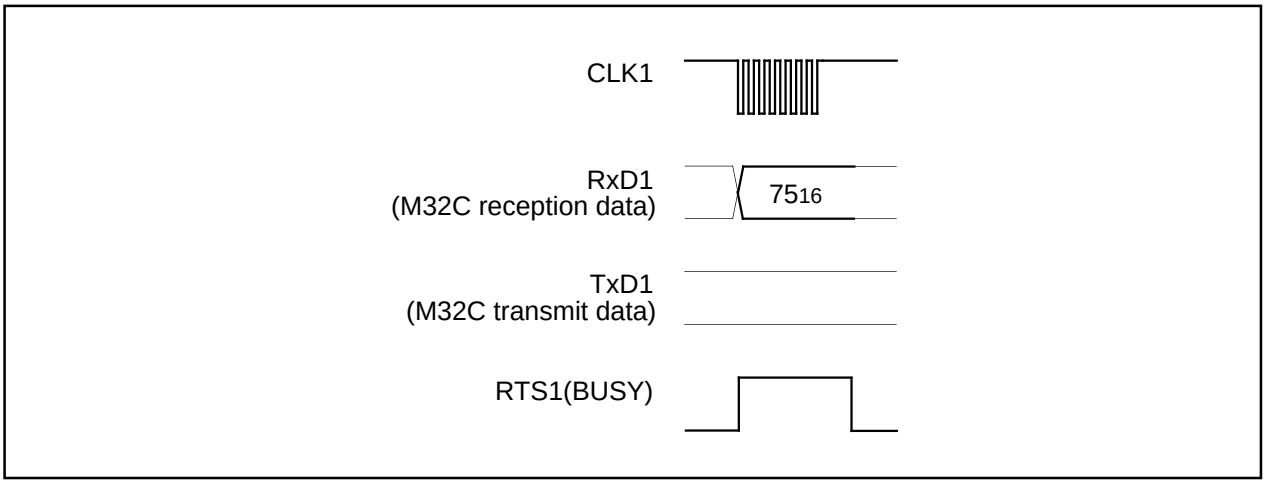



**Figure 1.35.13. Timing for disabling the lock bit**

**ID Check**

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

(1) Transfer the "$F5_{16}$" command code with the 1st byte.

(2) Transfer addresses $A_0$ to $A_7$, $A_8$ to $A_{15}$ and $A_{16}$ to $A_{23}$ of the 1st byte of the ID code with the 2nd, 3rd and 4th bytes respectively.

(3) Transfer the number of data sets of the ID code with the 5th byte.

(4) The ID code is sent with the 6th byte onward, starting with the 1st byte of the code.

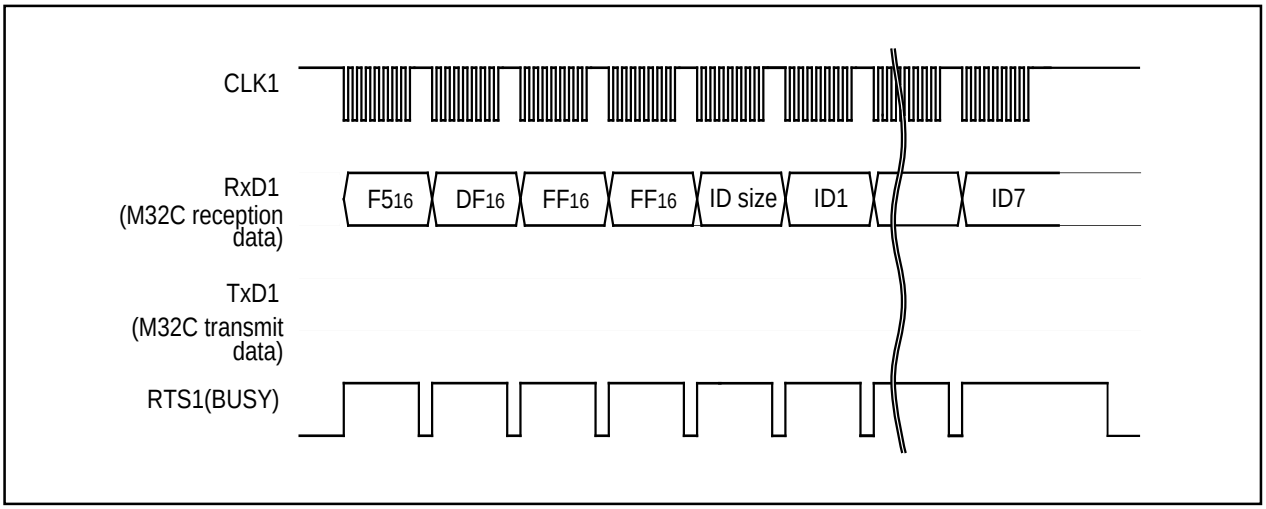



**Figure 1.35.14. Timing for the ID check**

**Download Command**

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

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

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

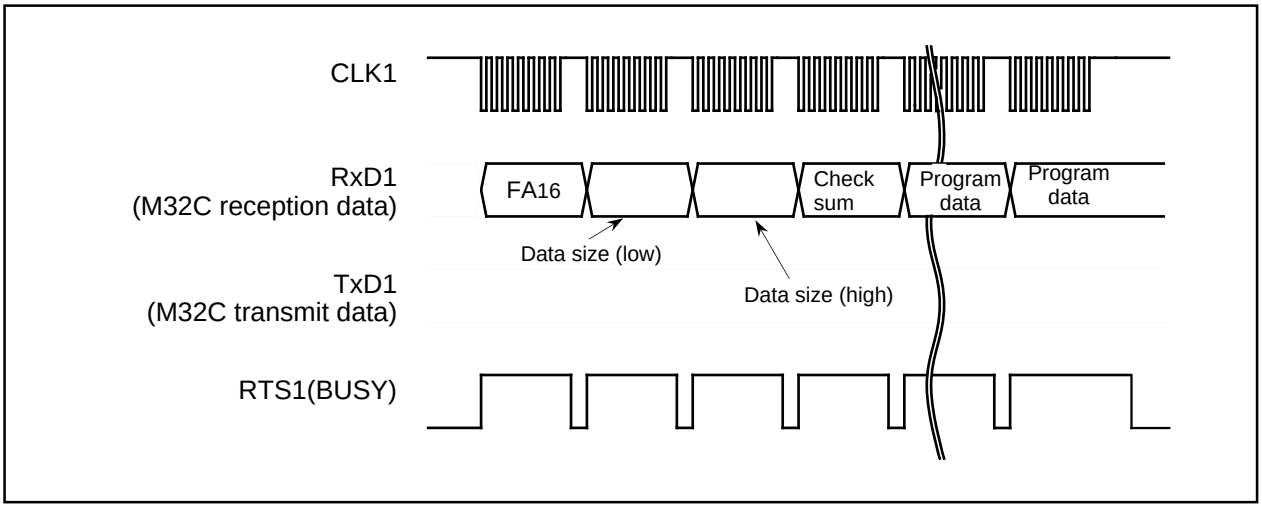



**Figure 1.35.15. Timing for download**

**Version Information Output Command**

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

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

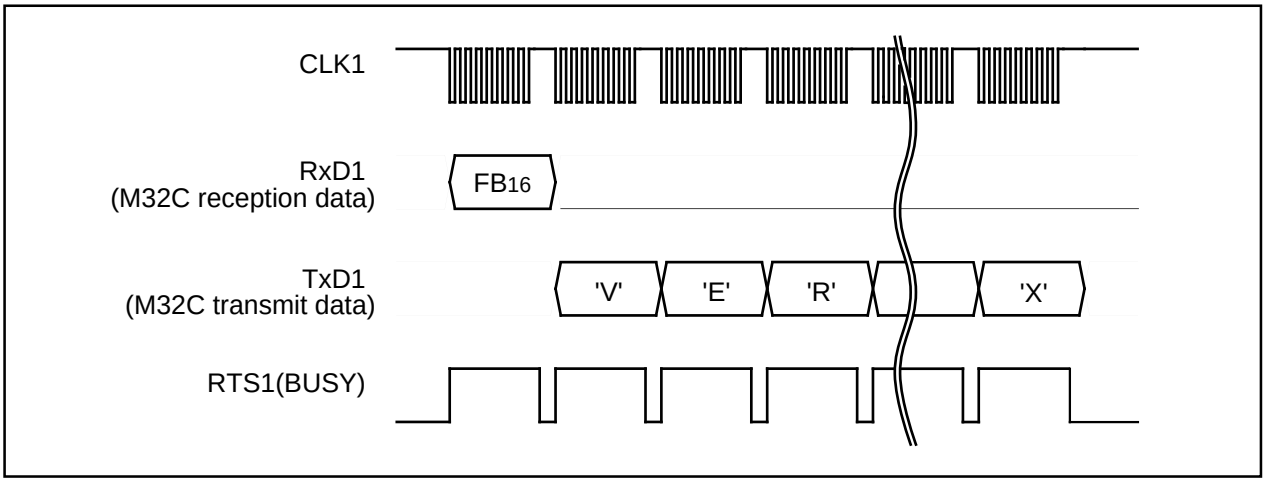



**Figure 1.35.16. Timing for version information output**

**Boot ROM Area Output Command**

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

(1) Transfer the "$FC_{16}$" command code with the 1st byte.

(2) Transfer addresses $A_8$ to $A_{15}$ and $A_{16}$ to $A_{23}$ with the 2nd and 3rd bytes respectively.

(3) From the 4th byte onward, data ($D_0$–$D_7$) for the page (256 bytes) specified with addresses $A_8$ to $A_{23}$ will be output sequentially from the smallest address first, in sync with the rise of the clock.

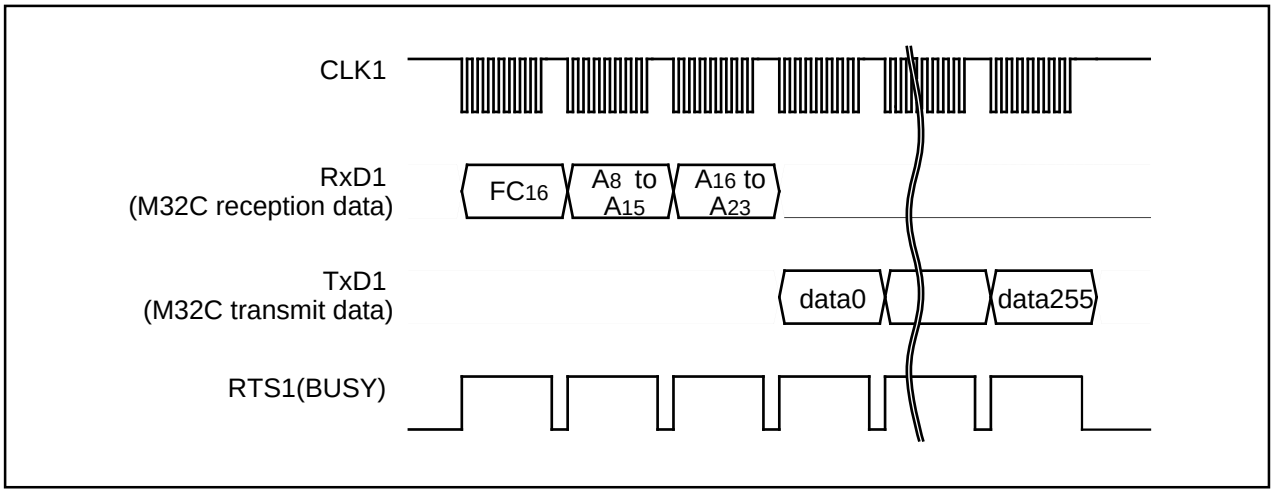



**Figure 1.35.17. Timing for boot ROM area output**

**Read Check Data**

This command reads the check data that confirms that the write data, which was sent with the page program command, was successfully received.

(1) Transfer the "$FD_{16}$" command code with the 1st byte.

(2) The check data (low) is received with the 2nd byte and the check data (high) with the 3rd.

To use this read check data command, first execute the command and then initialize the check data. Next, execute the page program command the required number of times. After that, when the read check command is executed again, the check data for all of the read data that was sent with the page program command during this time is read. The check data is the result of CRC operation of write data.

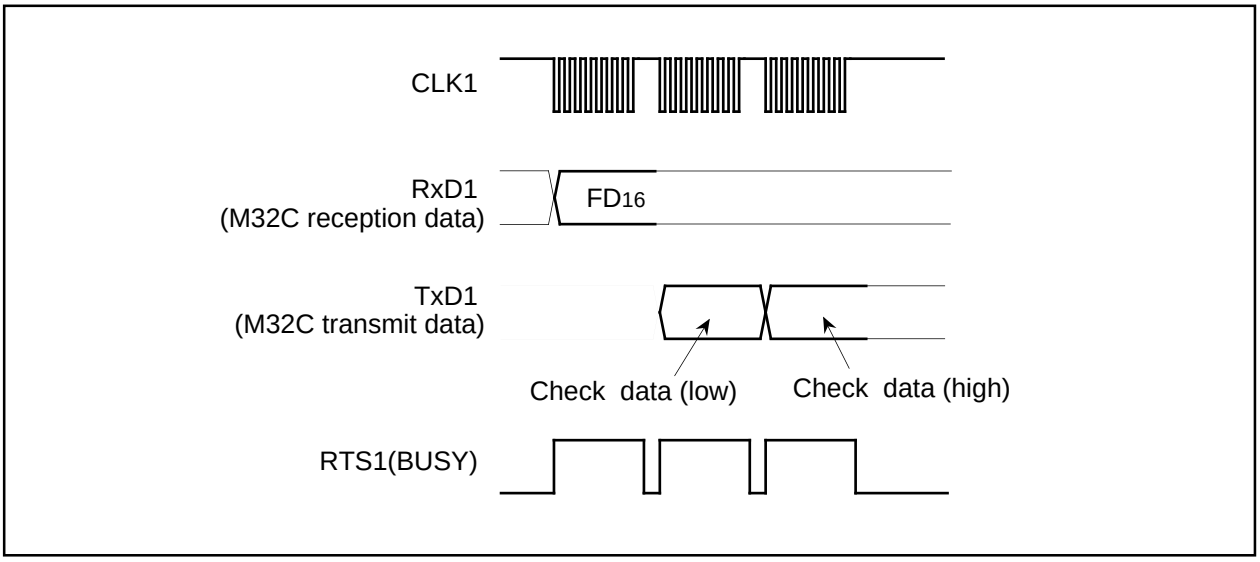



**Figure 1.35.18. Timing for the read check data**

**ID Code**

When the flash memory is not blank, the ID code sent from the peripheral units and the ID code written in the flash memory are compared to see if they match. If the codes do not match, the command sent from the peripheral units is not accepted. An ID code contains 8 bits of data. Area is, from the 1st byte, addresses 0FFFFDF16, 0FFFFE316, 0FFFFEB16, 0FFFFEF16, 0FFFFF316, 0FFFFF716 and 0FFFFFB16. Write a program into the flash memory, which already has the ID code set for these addresses.

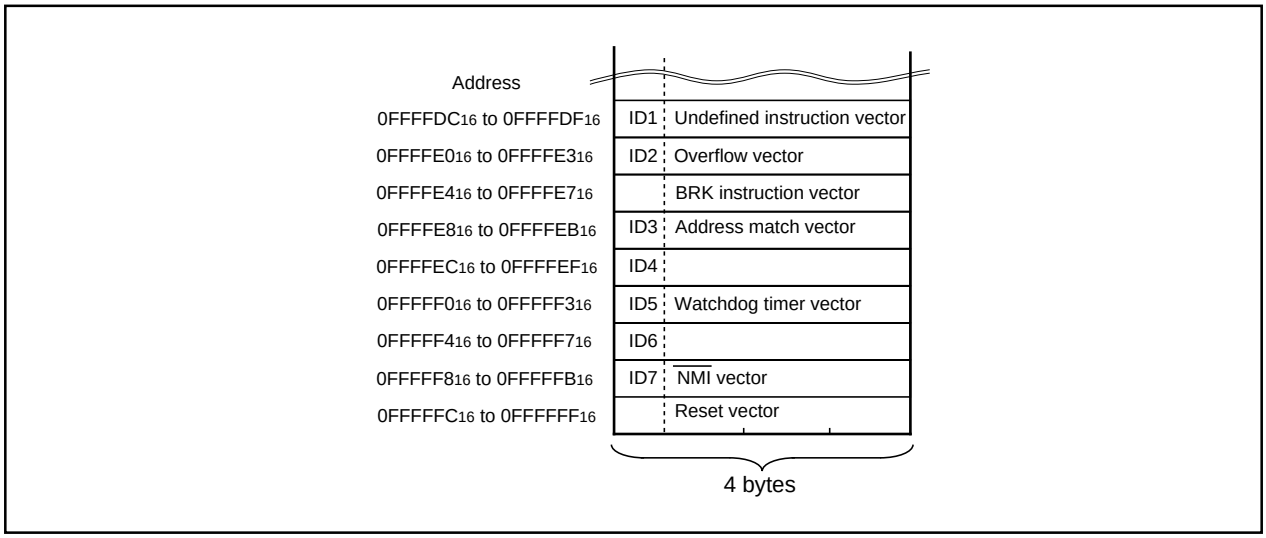



**Figure 1.35.19. ID code storage addresses**

## Data Protection (Block Lock)

Each of the blocks in Figure 1.35.20 have a nonvolatile lock bit that specifies protection (block lock) against erasing/writing. A block is locked (writing “0” for the lock bit) with the lock bit program command. Also, the lock bit of any block can be read with the read lock bit status command.

Block lock disable/enable is determined by the status of the lock bit itself and execution status of the lock bit disable and lock enable bit commands.

(1) After the reset has been cancelled and the lock bit enable command executed, the specified block can be locked/unlocked using the lock bit (lock bit data). Blocks with a “0” lock bit data are locked and cannot be erased or written in. On the other hand, blocks with a “1” lock bit data are unlocked and can be erased or written in.

(2) After the lock bit enable command has been executed, all blocks are unlocked regardless of lock bit data status and can be erased or written in. In this case, lock bit data that was “0” before the block was erased is set to “1” (unlocked) after erasing, therefore the block is actually unlocked with the lock bit.

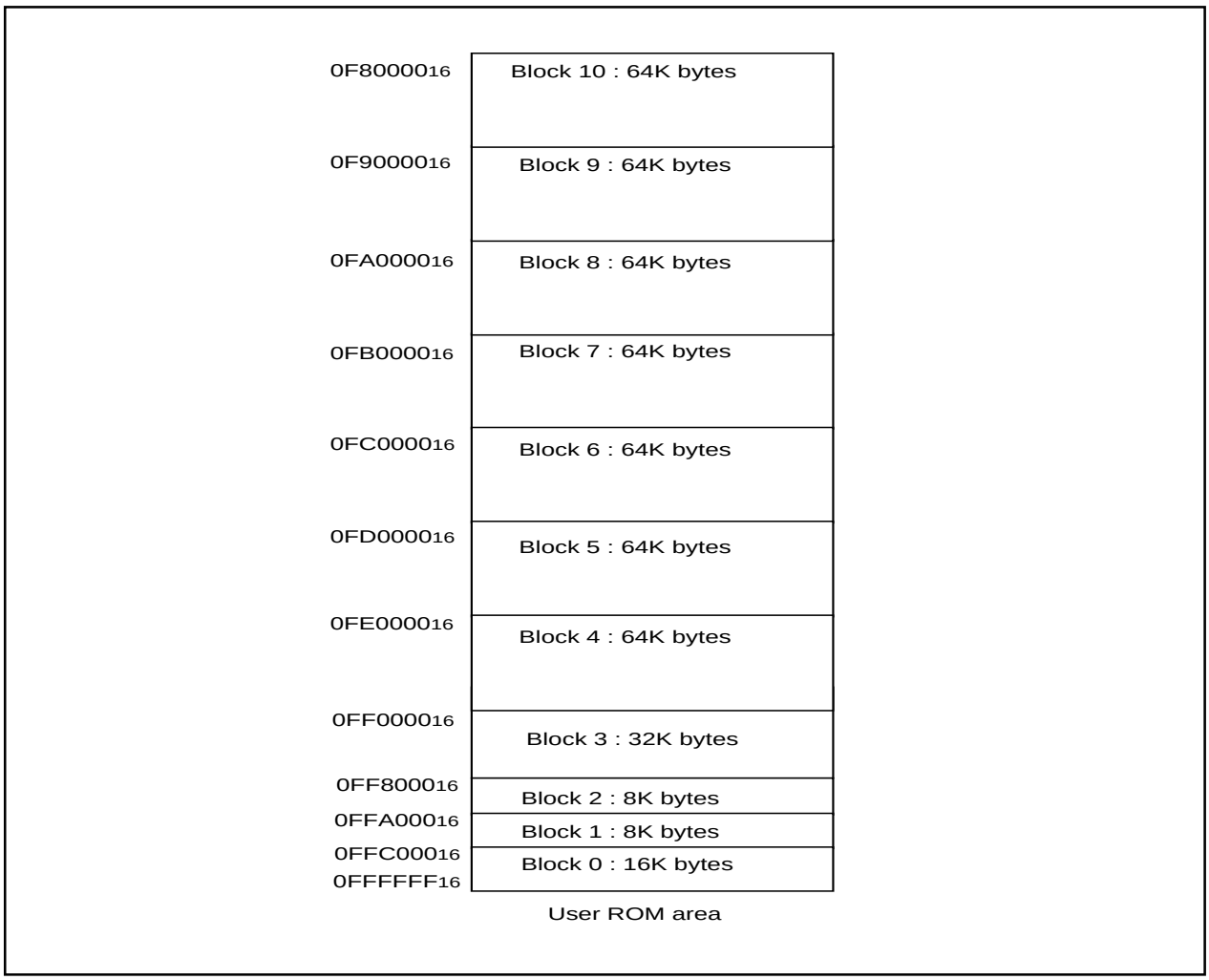



**Figure 1.35.20. Blocks in the user area**

## Status Register (SRD)

The status register indicates operating status of the flash memory and status such as whether an erase operation or a program ended successfully or in error. It can be read by writing the read status register command ($70_{16}$). Also, the status register is cleared by writing the clear status register command ($50_{16}$). Table 1.35.2 gives the definition of each status register bit. After clearing the reset, the status register outputs "$80_{16}$".

**Table 1.35.2. Status register (SRD)**

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

**Program Status After Program (SR3)**

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

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

**Program Status (SR4)**

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

**Erase Status (SR5)**

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

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

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

## Status Register 1 (SRD1)

Status register 1 indicates the status of serial communications, results from ID checks and results from check sum comparisons. It can be read after the SRD by writing the read status register command ($70_{16}$). Also, status register 1 is cleared by writing the clear status register command ($50_{16}$).

Table 1.35.3 gives the definition of each status register 1 bit. "$00_{16}$" is output when power is turned ON and the flag status is maintained even after the reset.

**Table 1.35.3. Status register 1 (SRD1)**

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

**Data Reception Time Out (SR9)**

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

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

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

**Check Sum Consistency Bit (SR12)**

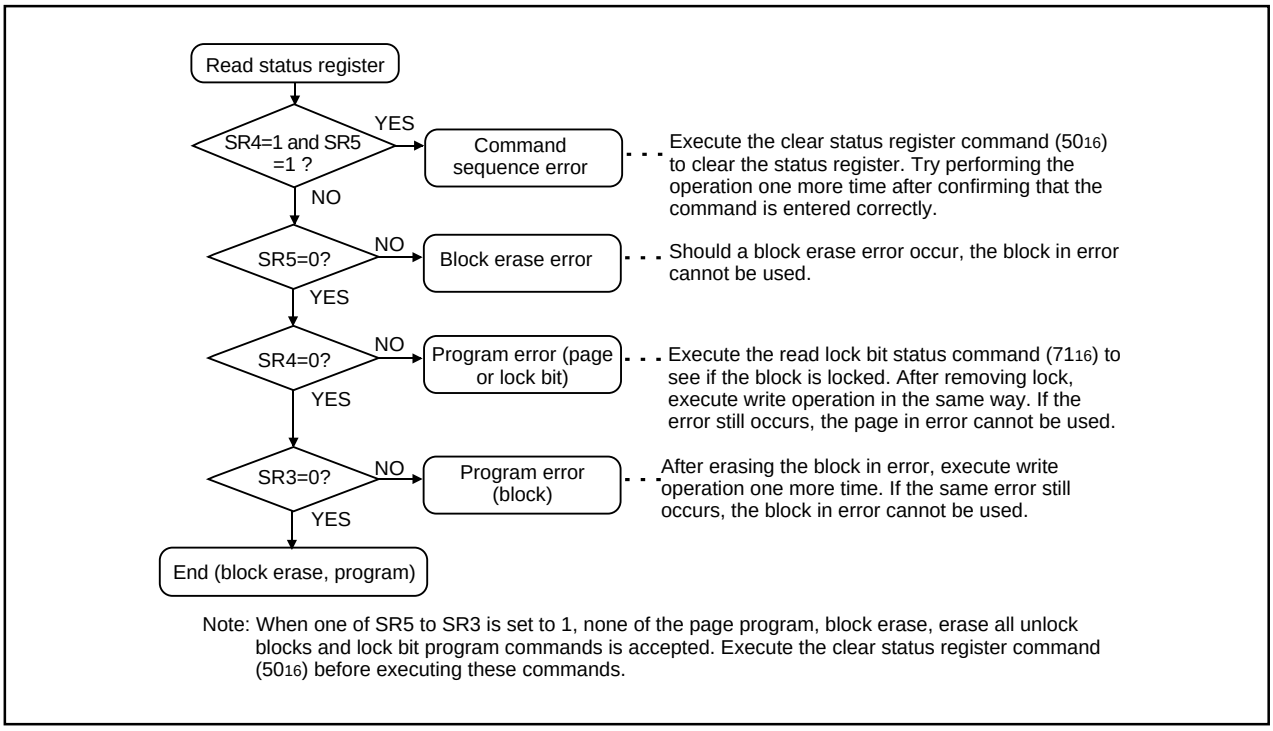
This flag indicates whether the check sum matches or not when a program, is downloaded for execution using the download function.

**Boot Update Completed Bit (SR15)**

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

## Full Status Check

Results from executed erase and program operations can be known by running a full status check. Figure 1.35.21 shows a flowchart of the full status check and explains how to remedy errors which occur.

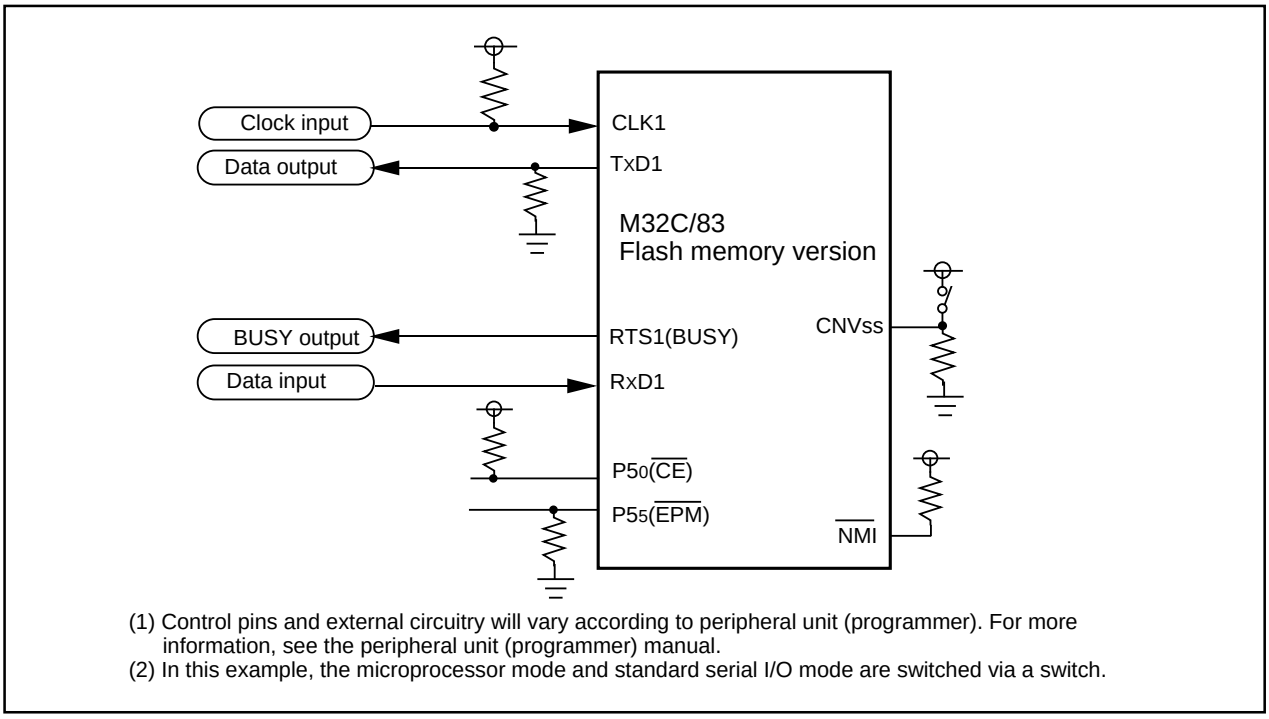



Note: When one of SR5 to SR3 is set to 1, none of the page program, block erase, erase all unlock blocks and lock bit program commands is accepted. Execute the clear status register command ($50_{16}$) before executing these commands.

**Figure 1.35.21. Full status check flowchart and remedial procedure for errors**

## Example Circuit Application for The Standard Serial I/O Mode 1

The below figure shows a circuit application for the standard serial I/O mode 1. Control pins will vary according to peripheral unit (programmer), therefore see the peripheral unit (programmer) manual for more information.




(1) Control pins and external circuitry will vary according to peripheral unit (programmer). For more information, see the peripheral unit (programmer) manual.
(2) In this example, the microprocessor mode and standard serial I/O mode are switched via a switch.

**Figure 1.35.22. Example circuit application for the standard serial I/O mode 1**

## Overview of standard serial I/O mode 2 (clock asynchronized)

In standard serial I/O mode 2, software commands, addresses and data are input and output between the MCU and peripheral units (serial programer, etc.) using 2-wire clock-asynchronized serial I/O (UART1). Standard serial I/O mode 2 is engaged by releasing the reset with the $P6_5$ ($CLK_1$) pin "L" level.

The $TxD_1$ pin is for CMOS output. Data transfer is in 8-bit units with LSB first, 1 stop bit and parity OFF.

After the reset is released, connections can be established at 9,600 bps when initial communications (Figure 1.35.23) are made with a peripheral unit. However, this requires a main clock with a minimum 2 MHz input oscillation frequency. Baud rate can be changed from 9,600 bps to 19,200, 38,400, 57,600 or 115,200 bps by executing software commands. However, communication errors may occur because of the oscillation frequency of the main clock. If errors occur, change the main clock's oscillation frequency and the baud rate.

After executing commands from a peripheral unit that requires time to erase and write data, as with erase and program commands, allow a sufficient time interval or execute the read status command and check how processing ended, before executing the next command.

Data and status registers in memory can be read after transmitting software commands. Status, such as the operating state of the flash memory or whether a program or erase operation ended successfully or not, can be checked by reading the status register. Here following are explained initial communications with peripheral units, how frequency is identified and software commands.

## Initial communications with peripheral units

After the reset is released, the bit rate generator is adjusted to 9,600 bps to match the oscillation frequency of the main clock, by sending the code as prescribed by the protocol for initial communications with peripheral units (Figure 1.35.23).

(1) Transmit "$00_{16}$" from a peripheral unit 16 times. (The MCU with internal flash memory sets the bit rate generator so that "$00_{16}$" can be successfully received.)

(2) The MCU with internal flash memory outputs the "$B0_{16}$" check code and initial communications end successfully *1. Initial communications must be transmitted at a speed of 9,600 bps and a transfer interval of a minimum 15 ms. Also, the baud rate at the end of initial communications is 9,600 bps.

*1. If the peripheral unit cannot receive "$B0_{16}$" successfully, change the oscillation frequency of the main clock.

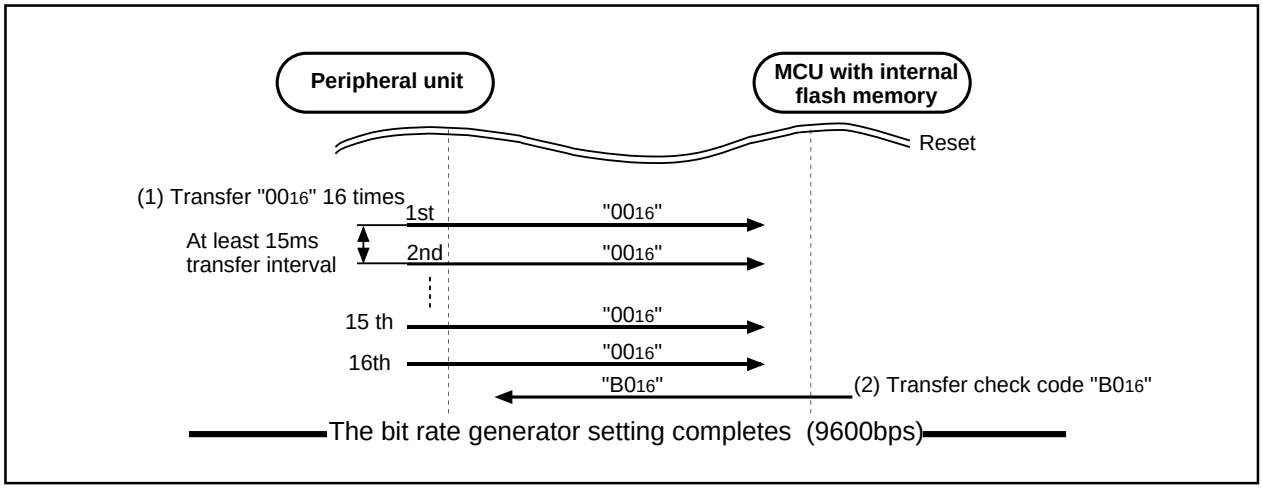



**Figure 1.35.23. Peripheral unit and initial communication**

## How frequency is identified

When "$00_{16}$" data is received 16 times from a peripheral unit at a baud rate of 9,600 bps, the value of the bit rate generator is set to match the operating frequency (2 - 30 MHz). The highest speed is taken from the first 8 transmissions and the lowest from the last 8. These values are then used to calculate the bit rate generator value for a baud rate of 9,600 bps.

Baud rate cannot be attained with some operating frequencies. Table 1.35.4 gives the operation frequency and the baud rate that can be attained for.

**Table 1.35.4 Operation frequency and the baud rate**

| Operation frequency (MHz) | Baud rate 9,600bps | Baud rate 19,200bps | Baud rate 38,400bps | Baud rate 57,600bps | Baud rate 115,200bps |
|---|---|---|---|---|---|
| 30MHz | √ | √ | √ | √ | – |
| 20MHz | √ | √ | √ | √ | √ |
| 16MHz | √ | √ | √ | √ | – |
| 12MHz | √ | √ | √ | √ | – |
| 11MHz | √ | √ | √ | √ | – |
| 10MHz | √ | √ | √ | √ | – |
| 8MHz | √ | √ | √ | √ | – |
| 7.3728MHz | √ | √ | √ | √ | – |
| 6MHz | √ | √ | √ | – | – |
| 5MHz | √ | √ | √ | – | – |
| 4.5MHz | √ | √ | √ | √ | – |
| 4.194304MHz | √ | √ | √ | – | – |
| 4MHz | √ | √ | – | – | – |
| 3.58MHz | √ | √ | √ | √ | – |
| 3MHz | √ | √ | √ | – | – |
| 2MHz | √ | – | – | – | – |

√ : Communications possible
– : Communications not possible

## Software Commands

Table 1.35.5 lists software commands. In the standard serial I/O mode 2, erase operations, programs and reading are controlled by transferring software commands via the $RxD_1$ pin. Standard serial I/O mode 2 adds five transmission speed commands - 9,600, 19,200, 38,400, 57,600 and 115,200 bps - to the software commands of standard serial I/O mode 1. Software commands are explained here below.

**Table 1.35.5. Software commands (Standard serial I/O mode 2)**

| | Control command | 1st byte transfer | 2nd byte | 3rd byte | 4th byte | 5th byte | 6th byte | | When ID is not verified |
|---|---|---|---|---|---|---|---|---|---|
| 1 | Page read | $FF_{16}$ | Address (middle) | Address (high) | Data output | Data output | Data output | Data output to 259th byte | Not acceptable |
| 2 | Page program | $41_{16}$ | Address (middle) | Address (high) | Data input | Data input | Data input | Data input to 259th byte | Not acceptable |
| 3 | Block erase | $20_{16}$ | Address (middle) | Address (high) | $D0_{16}$ | | | | Not acceptable |
| 4 | Erase all unlocked blocks | $A7_{16}$ | $D0_{16}$ | | | | | | Not acceptable |
| 5 | Read status register | $70_{16}$ | SRD output | SRD1 output | | | | | Acceptable |
| 6 | Clear status register | $50_{16}$ | | | | | | | Not acceptable |
| 7 | Read lock bit status | $71_{16}$ | Address (middle) | Address (high) | Lock bit data output | | | | Not acceptable |
| 8 | Lock bit program | $77_{16}$ | Address (middle) | Address (high) | $D0_{16}$ | | | | Not acceptable |
| 9 | Lock bit enable | $7A_{16}$ | | | | | | | Not acceptable |
| 10 | Lock bit disable | $75_{16}$ | | | | | | | Not acceptable |
| 11 | Code processing function | $F5_{16}$ | Address (low) | Address (middle) | Address (high) | ID size | ID1 | To ID7 | Acceptable |
| 12 | Download function | $FA_{16}$ | Size (low) | Size (high) | Check-sum | Data input | To required number of times | | Not acceptable |
| 13 | Version data output function | $FB_{16}$ | Version data output | Version data output | Version data output | Version data output | Version data output | Version data output to 9th byte | Acceptable |
| 14 | Boot ROM area output function | $FC_{16}$ | Address (middle) | Address (high) | Data output | Data output | Data output | Data output to 259th byte | Not acceptable |
| 15 | Read check data | $FD_{16}$ | Check data (low) | Check data (high) | | | | | Not acceptable |
| 16 | Baud rate 9600 | $B0_{16}$ | $B0_{16}$ | | | | | | Acceptable |
| 17 | Baud rate 19200 | $B1_{16}$ | $B1_{16}$ | | | | | | Acceptable |
| 18 | Baud rate 38400 | $B2_{16}$ | $B2_{16}$ | | | | | | Acceptable |
| 19 | Baud rate 57600 | $B3_{16}$ | $B3_{16}$ | | | | | | Acceptable |
| 20 | Baud rate 115200 | $B4_{16}$ | $B4_{16}$ | | | | | | Acceptable |

Note 1: Shading indicates transfer from flash memory microcomputer to peripheral unit. All other data is transferred from the peripheral unit to the flash memory microcomputer.

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

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

**Page Read Command**

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

(1) Transfer the "$FF_{16}$" command code with the 1st byte.

(2) Transfer addresses $A_8$ to $A_{15}$ and $A_{16}$ to $A_{23}$ with the 2nd and 3rd bytes respectively.

(3) From the 4th byte onward, data ($D_0$–$D_7$) for the page (256 bytes) specified with addresses $A_8$ to $A_{23}$ will be output sequentially from the smallest address first in sync with the rise of the clock.

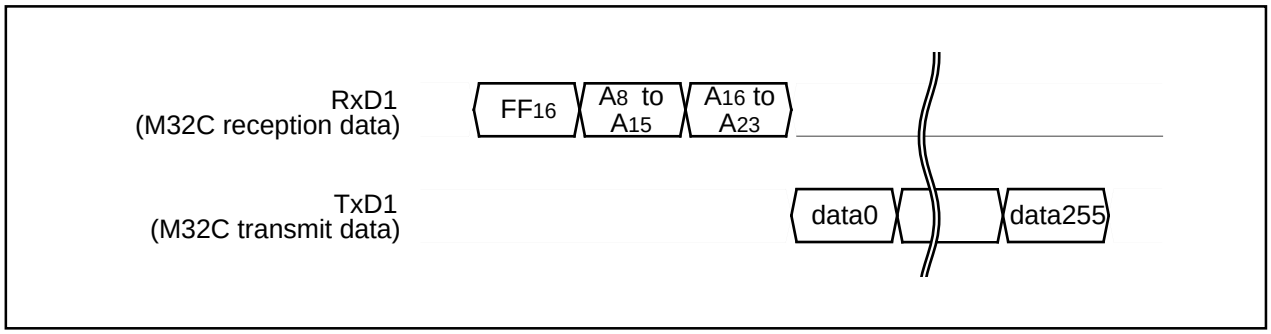



**Figure 1.35.24. Timing for page read**

**Page Program Command**

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

(1) Transfer the "$41_{16}$" command code with the 1st byte.

(2) Transfer addresses $A_8$ to $A_{15}$ and $A_{16}$ to $A_{23}$ with the 2nd and 3rd bytes respectively.

(3) From the 4th byte onward, as write data ($D_0$–$D_7$) for the page (256 bytes) specified with addresses $A_8$ to $A_{23}$ is input sequentially from the smallest address first, that page is automatically written.

The result of the page program can be known by reading the status register. For more information, see the section on the status register.

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

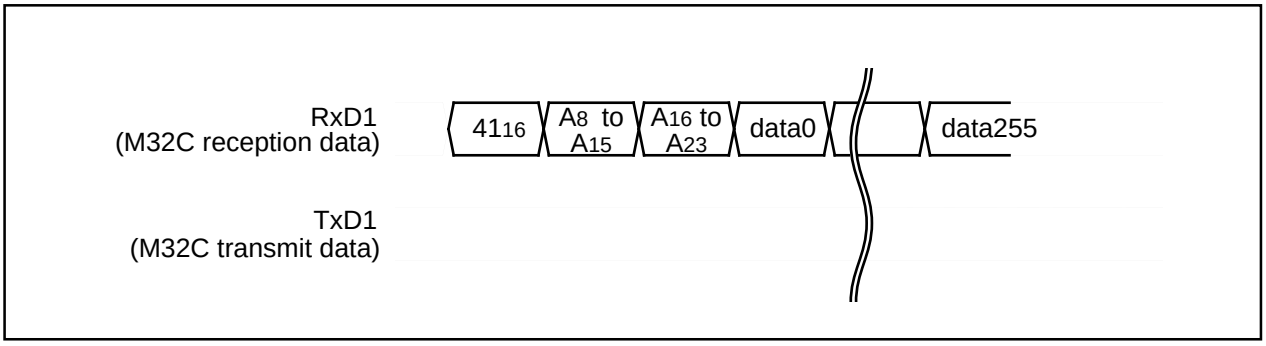



**Figure 1.35.25. Timing for the page program**

**Block Erase Command**

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

(1) Transfer the "$20_{16}$" command code with the 1st byte.

(2) Transfer addresses $A_8$ to $A_{15}$ and $A_{16}$ to $A_{23}$ with the 2nd and 3rd bytes respectively.

(3) Transfer the verify command code "$D0_{16}$" with the 4th byte. With the verify command code, the erase operation will start for the specified block in the flash memory. Write the highest address of the specified block for addresses $A_{16}$ to $A_{23}$.

After block erase ends, the result of the block erase operation can be known by reading the status register. For more information, see the section on the status register.

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

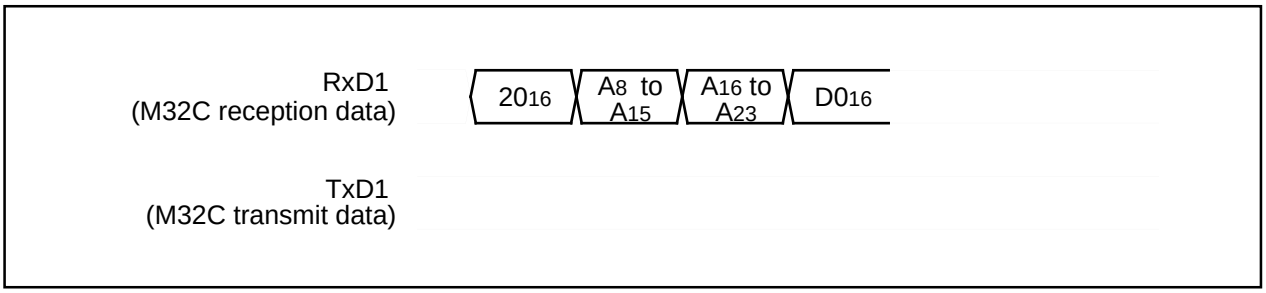



**Figure 1.35.26. Timing for block erasing**

**Erase All Unlocked Blocks Command**

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

(1) Transfer the "$A7_{16}$" command code with the 1st byte.

(2) Transfer the verify command code "$D0_{16}$" with the 2nd byte. With the verify command code, the erase operation will start and continue for all blocks in the flash memory.

The result of the erase operation can be known by reading the status register. Each block can be erase-protected with the lock bit. For more information, see the section on the data protection function.

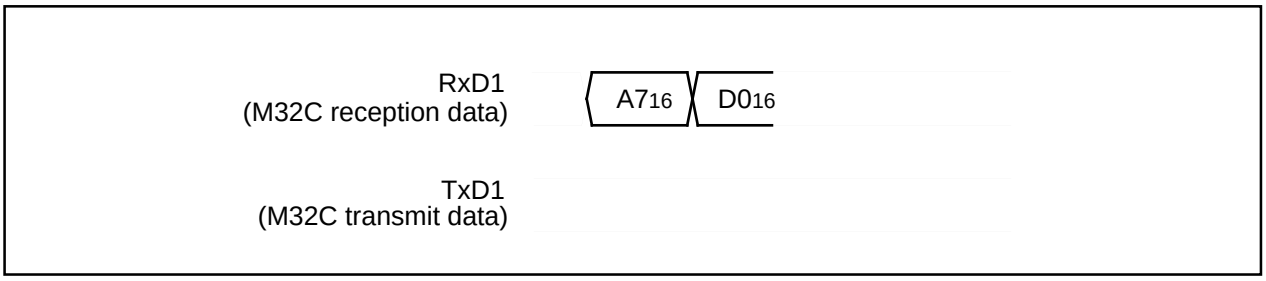



**Figure 1.35.27. Timing for erasing all unlocked blocks**

**Read Status Register Command**

This command reads status information. When the "$70_{16}$" command code is sent with the 1st byte, the contents of the status register (SRD) specified with the 2nd byte and the contents of status register 1 (SRD1) specified with the 3rd byte are read.

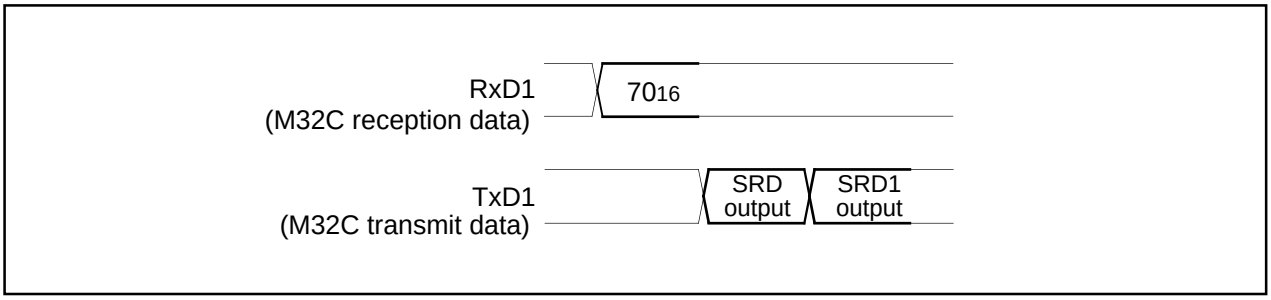



**Figure 1.35.28. Timing for reading the status register**

**Clear Status Register Command**

This command clears the bits (SR3–SR5) which are set when the status register operation ends in error. When the "$50_{16}$" command code is sent with the 1st byte, the aforementioned bits are cleared.

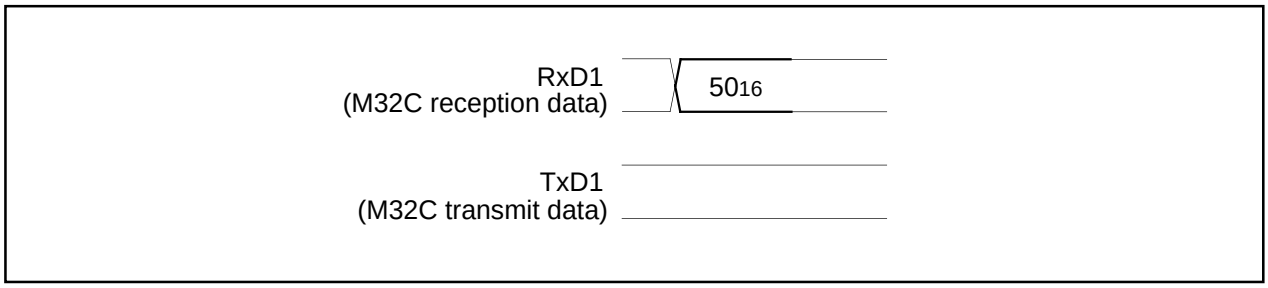



**Figure 1.35.29. Timing for clearing the status register**

**Read Lock Bit Status Command**

This command reads the lock bit status of the specified block. Execute the read lock bit status command as explained here following.

(1) Transfer the "$71_{16}$" command code with the 1st byte.

(2) Transfer addresses $A_8$ to $A_{15}$ and $A_{16}$ to $A_{23}$ with the 2nd and 3rd bytes respectively.

(3) The lock bit data of the specified block is output with the 4th byte. The 6th bit (D6) of output data is the lock bit data. Write the highest address of the specified block for addresses $A_8$ to $A_{23}$.

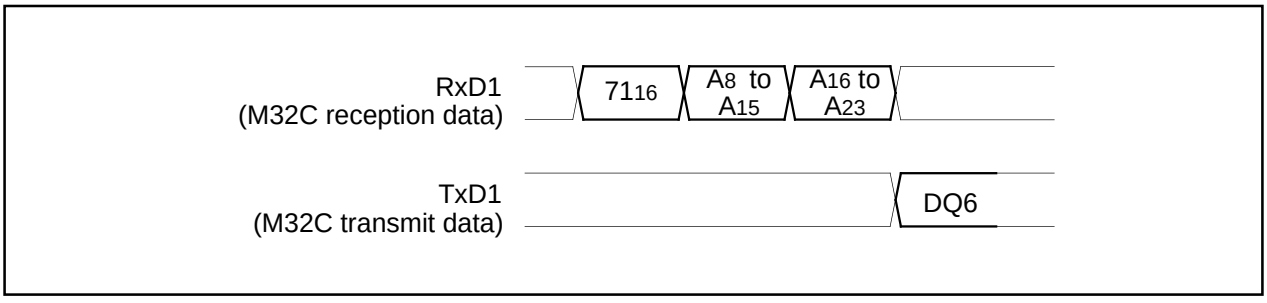



**Figure 1.35.30. Timing for reading lock bit status**

**Lock Bit Program Command**

This command writes "0" (lock) for the lock bit of the specified block. Execute the lock bit program command as explained here following.

(1) Transfer the "$77_{16}$" command code with the 1st byte.

(2) Transfer addresses $A_8$ to $A_{15}$ and $A_{16}$ to $A_{23}$ with the 2nd and 3rd bytes respectively.

(3) Transfer the verify command code "$D0_{16}$" with the 4th byte. With the verify command code, "0" is written for the lock bit of the specified block. Write the highest address of the specified block for addresses $A_8$ to $A_{23}$.

Lock bit status can be read with the read lock bit status command. For information on the lock bit function, reset procedure and so on, see the section on the data protection function.

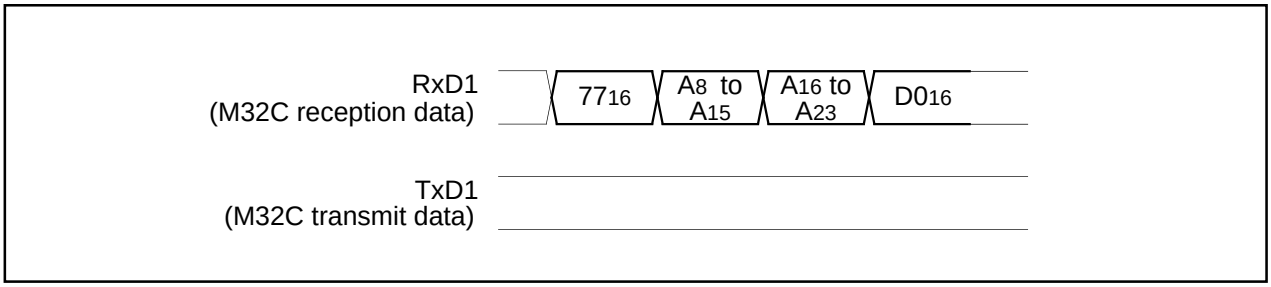



**Figure 1.35.31. Timing for the lock bit program**

**Lock Bit Enable Command**

This command enables the lock bit in blocks whose bit was disabled with the lock bit disable command. The command code "$7A_{16}$" is sent with the 1st byte of the serial transmission. This command only enables the lock bit function; it does not set the lock bit itself.

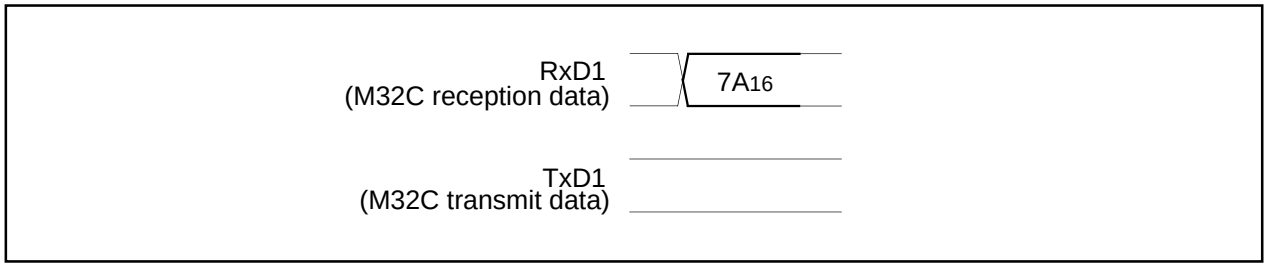



**Figure 1.35.32. Timing for enabling the lock bit**

**Lock Bit Disable Command**

This command disables the lock bit. The command code "$75_{16}$" is sent with the 1st byte of the serial transmission. This command only disables the lock bit function; it does not set the lock bit itself. However, if an erase command is executed after executing the lock bit disable command, "0" (locked) lock bit data is set to "1" (unlocked) after the erase operation ends. In any case, after the reset is cancelled, the lock bit is enabled.

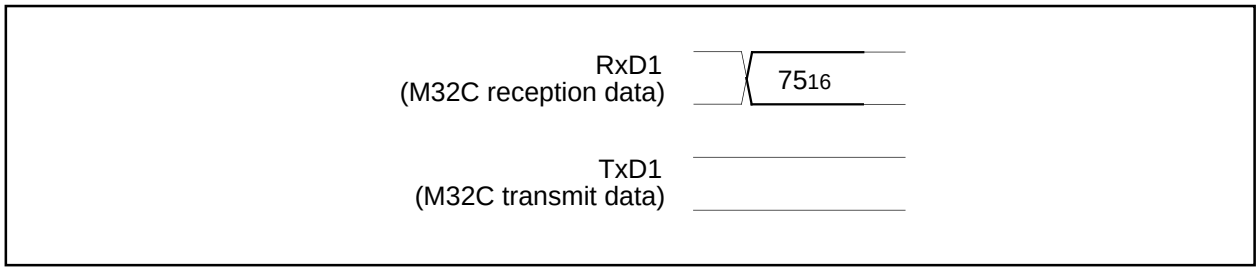



**Figure 1.35.33. Timing for disabling the lock bit**

**ID Check**

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

(1) Transfer the "$F5_{16}$" command code with the 1st byte.

(2) Transfer addresses $A_0$ to $A_7$, $A_8$ to $A_{15}$ and $A_{16}$ to $A_{23}$ of the 1st byte of the ID code with the 2nd, 3rd and 4th bytes respectively.

(3) Transfer the number of data sets of the ID code with the 5th byte.

(4) The ID code is sent with the 6th byte onward, starting with the 1st byte of the code.

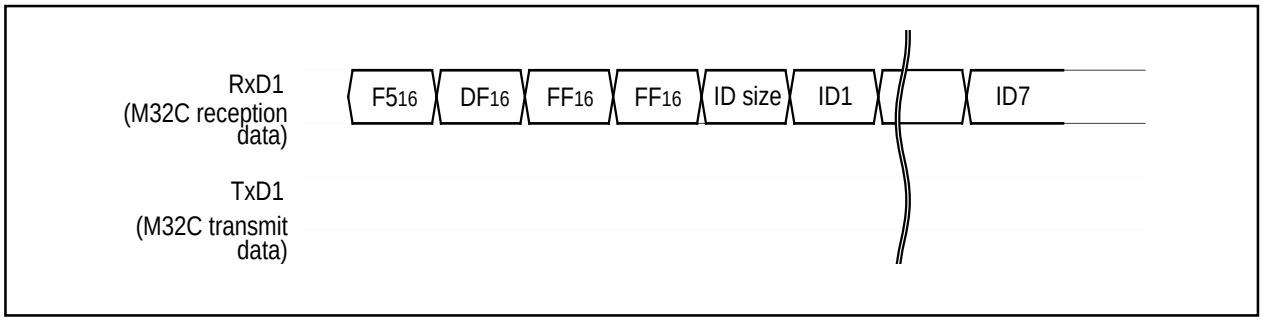



**Figure 1.35.34. Timing for the ID check**

**Download Command**

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

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

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

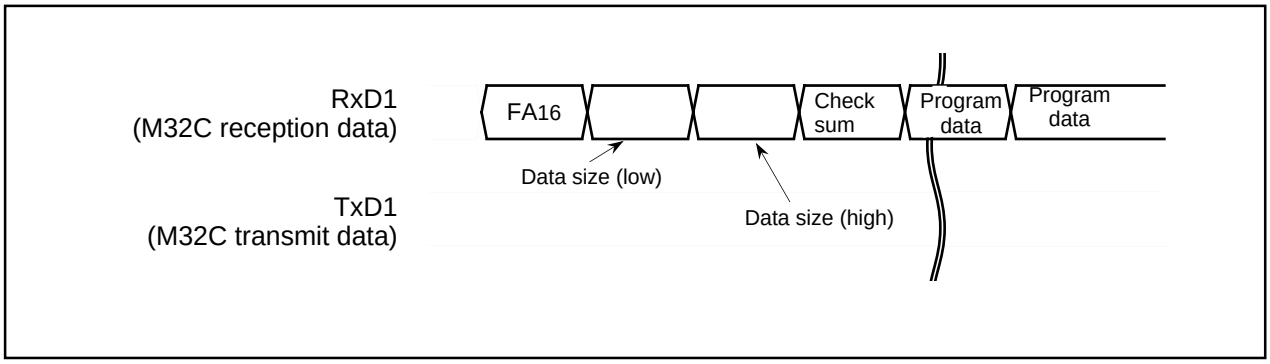



**Figure 1.35.35. Timing for download**

**Version Information Output Command**

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

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

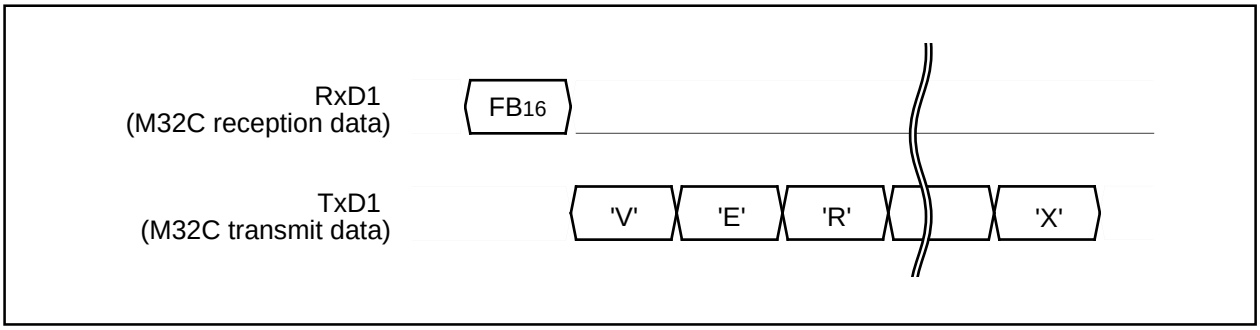



**Figure 1.35.36. Timing for version information output**

**Boot ROM Area Output Command**

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

(1) Transfer the "$FC_{16}$" command code with the 1st byte.

(2) Transfer addresses $A_8$ to $A_{15}$ and $A_{16}$ to $A_{23}$ with the 2nd and 3rd bytes respectively.

(3) From the 4th byte onward, data ($D_0$–$D_7$) for the page (256 bytes) specified with addresses $A_8$ to $A_{23}$ will be output sequentially from the smallest address first, in sync with the rise of the clock.

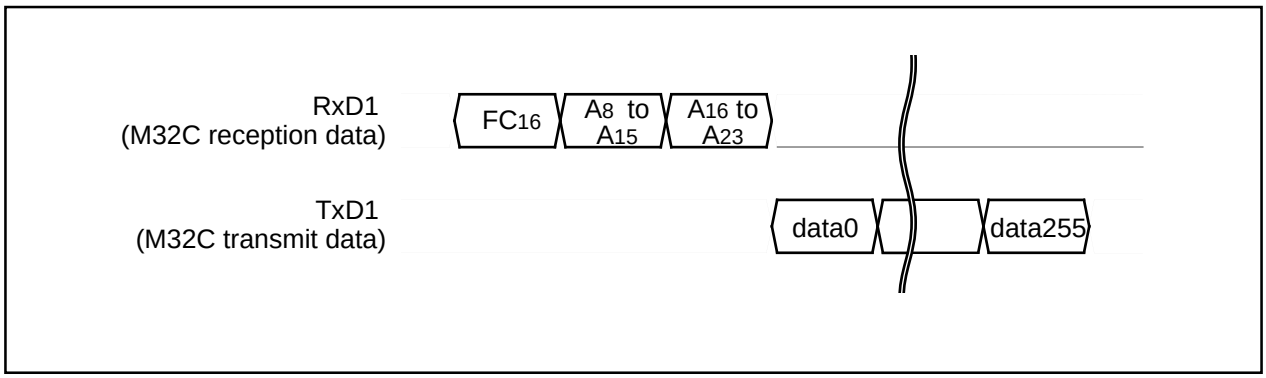



**Figure 1.35.37. Timing for boot ROM area output**

**Read Check Data**

This command reads the check data that confirms that the write data, which was sent with the page program command, was successfully received.

(1) Transfer the "$FD_{16}$" command code with the 1st byte.

(2) The check data (low) is received with the 2nd byte and the check data (high) with the 3rd.

To use this read check data command, first execute the command and then initialize the check data. Next, execute the page program command the required number of times. After that, when the read check command is executed again, the check data for all of the read data that was sent with the page program command during this time is read. The check data is the result of CRC operation of write data.

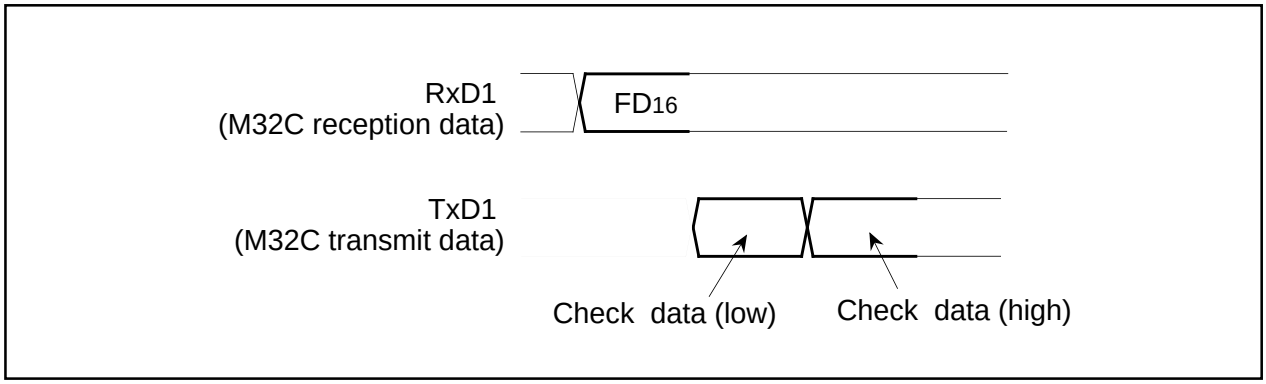



**Figure 1.35.38. Timing for the read check data**

**Baud Rate 9600**

This command changes baud rate to 9,600 bps. Execute it as follows.

(1) Transfer the "B0$_{16}$" command code with the 1st byte.

(2) After the "B0$_{16}$" check code is output with the 2nd byte, change the baud rate to 9,600 bps.

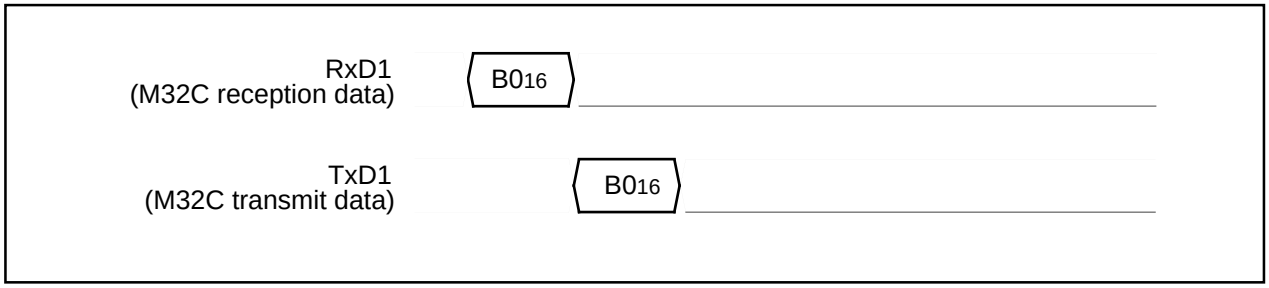



**Figure 1.35.39. Timing of baud rate 9600**

**Baud Rate 19200**

This command changes baud rate to 19,200 bps. Execute it as follows.

(1) Transfer the "B1$_{16}$" command code with the 1st byte.

(2) After the "B1$_{16}$" check code is output with the 2nd byte, change the baud rate to 19,200 bps.

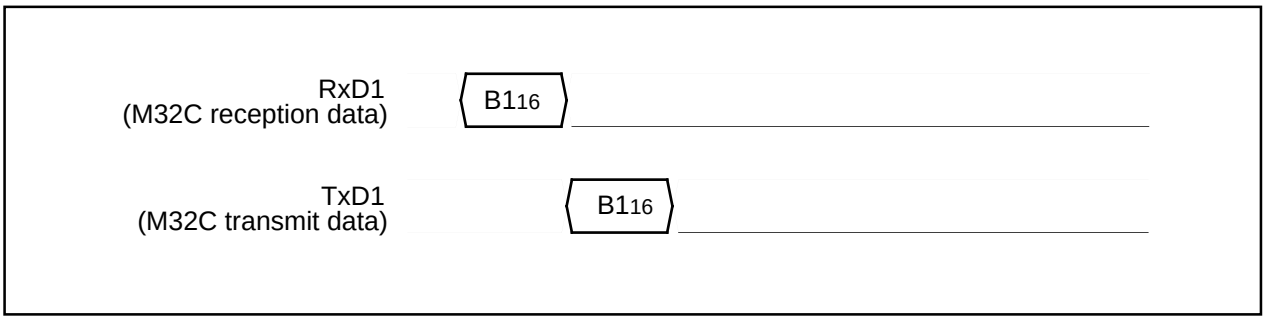



**Figure 1.35.40. Timing of baud rate 19200**

**Baud Rate 38400**

This command changes baud rate to 38,400 bps. Execute it as follows.

(1) Transfer the "B2$_{16}$" command code with the 1st byte.

(2) After the "B2$_{16}$" check code is output with the 2nd byte, change the baud rate to 38,400 bps.

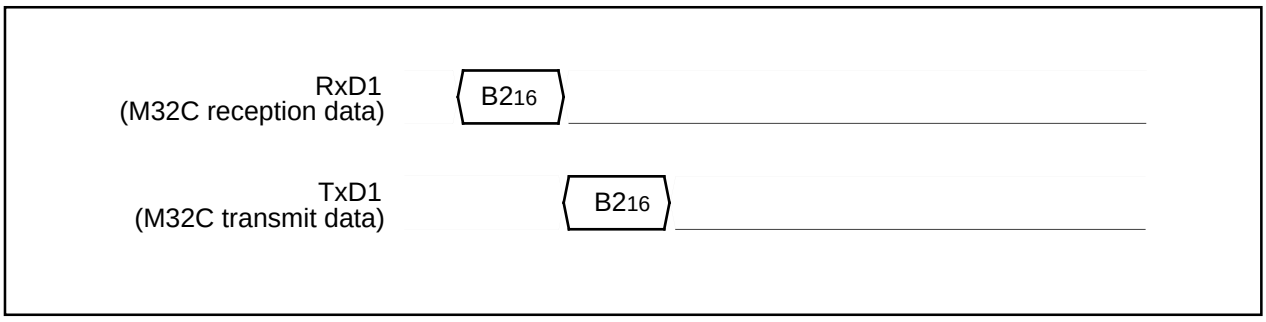



**Figure 1.35.41. Timing of baud rate 38400**

**Baud Rate 57600**

This command changes baud rate to 57,600 bps. Execute it as follows.

(1) Transfer the "B3$_{16}$" command code with the 1st byte.

(2) After the "B3$_{16}$" check code is output with the 2nd byte, change the baud rate to 57,600 bps.

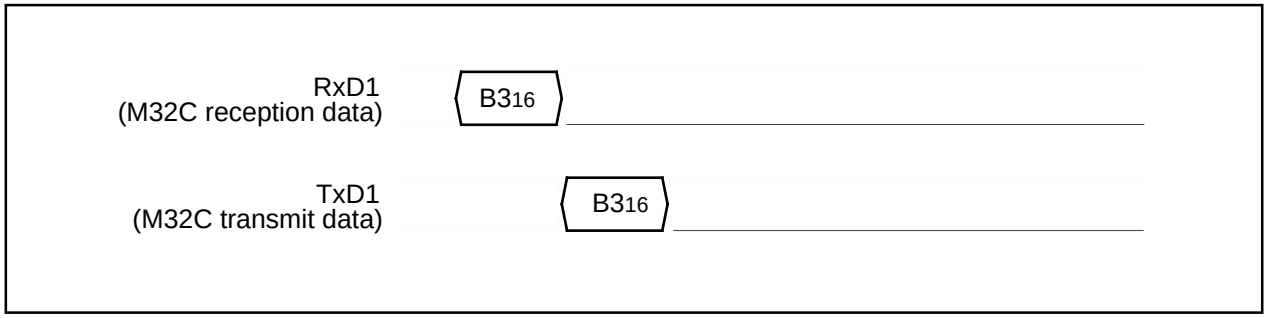



**Figure 1.35.42. Timing of baud rate 57600**

**Baud Rate 115200**

This command changes baud rate to 115,200 bps. Execute it as follows.

(1) Transfer the "B4$_{16}$" command code with the 1st byte.

(2) After the "B4$_{16}$" check code is output with the 2nd byte, change the baud rate to 19,200 bps.

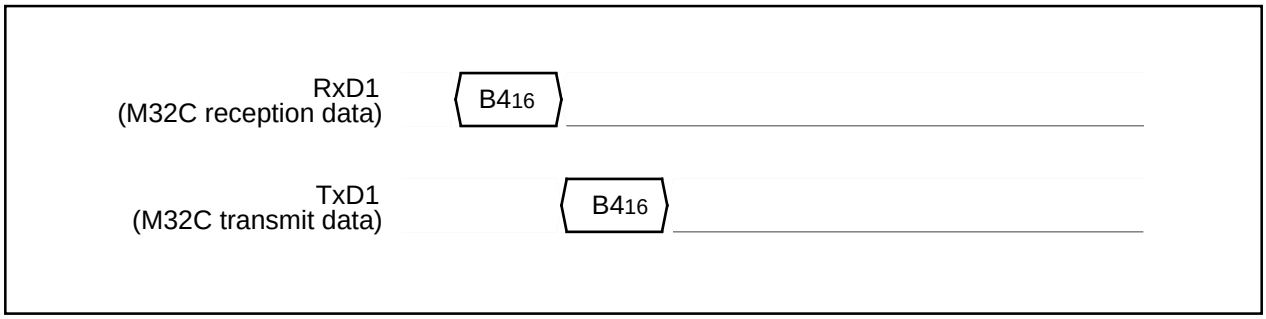



**Figure 1.35.43. Timing of baud rate 115200**

**ID Code**

When the flash memory is not blank, the ID code sent from the peripheral units and the ID code written in the flash memory are compared to see if they match. If the codes do not match, the command sent from the peripheral units is not accepted. An ID code contains 8 bits of data. Area is, from the 1st byte, addresses 0FFFFDF$_{16}$, 0FFFFE3$_{16}$, 0FFFFEB$_{16}$, 0FFFFEF$_{16}$, 0FFFFF3$_{16}$, 0FFFFF7$_{16}$ and 0FFFFFB$_{16}$. Write a program into the flash memory, which already has the ID code set for these addresses.

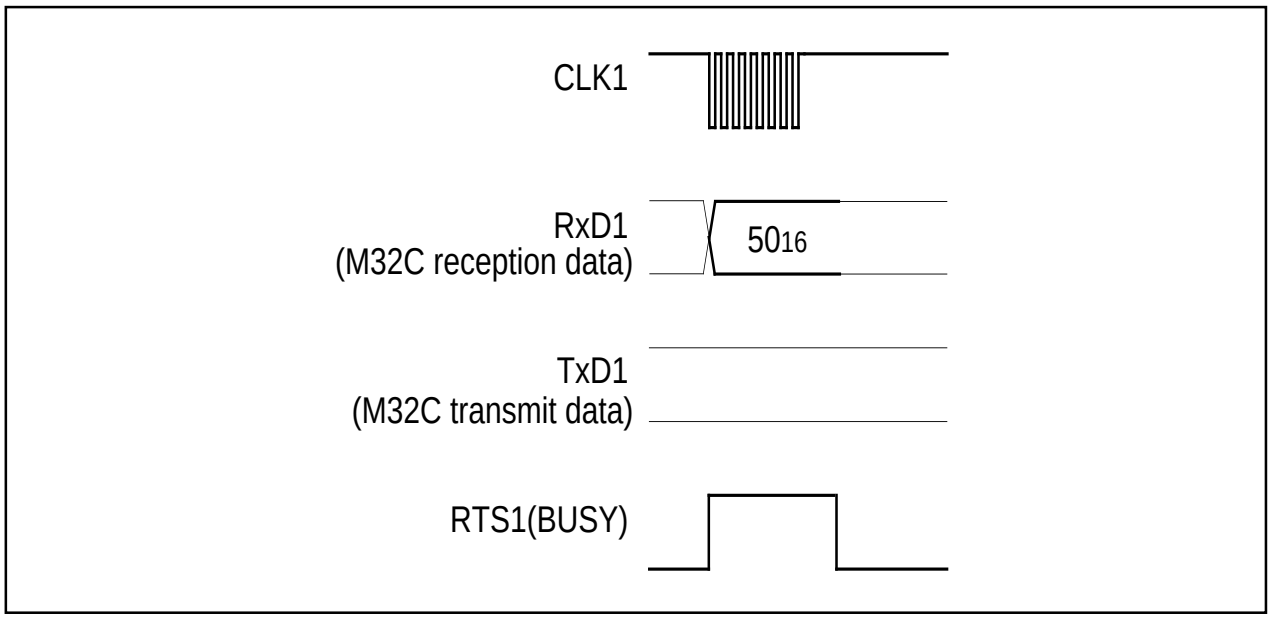



**Figure 1.35.44. ID code storage addresses**

## Example Circuit Application for The Standard Serial I/O Mode 2

The below figure shows a circuit application for the standard serial I/O mode 2.

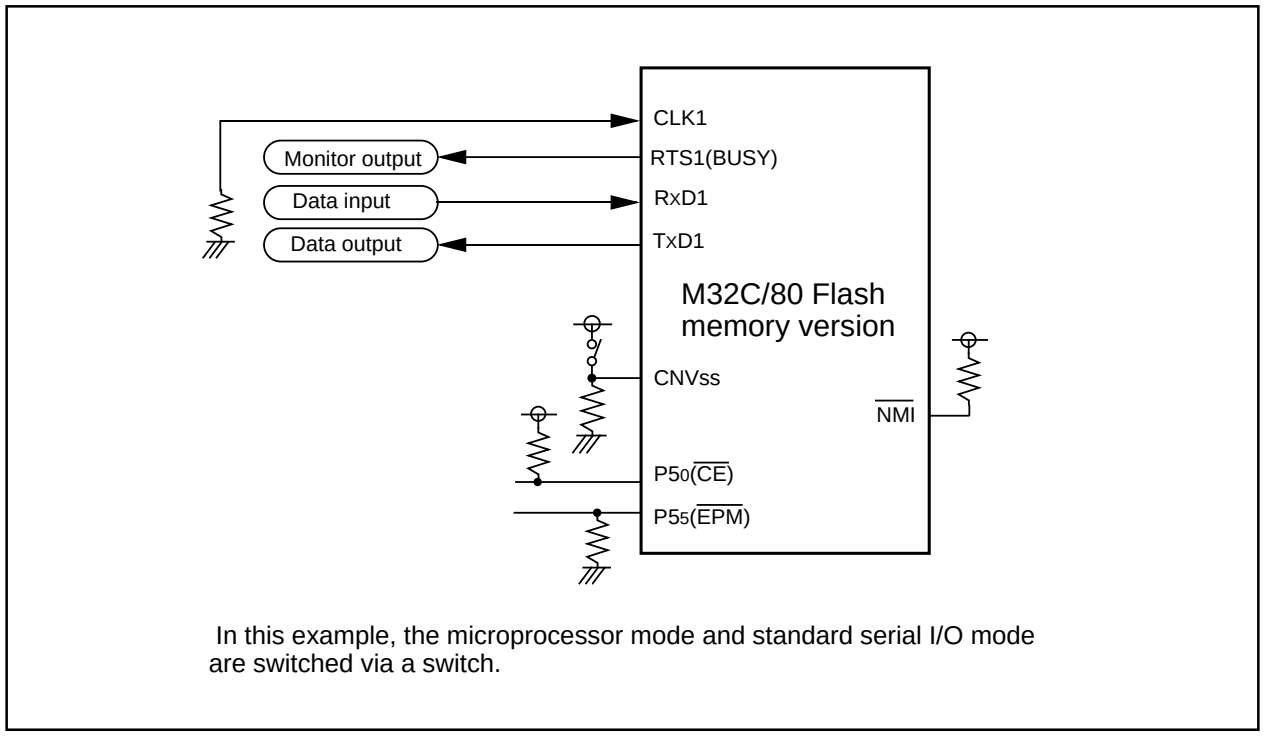



**Figure 1.35.45. Example circuit application for the standard serial I/O mode 2**

| CORRECT AND ERROR | M32C/83 GROUP DATA SHEET |
|---|---|

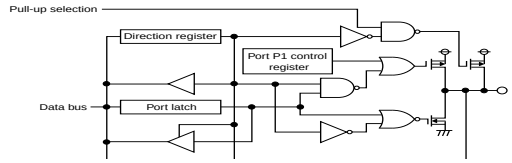
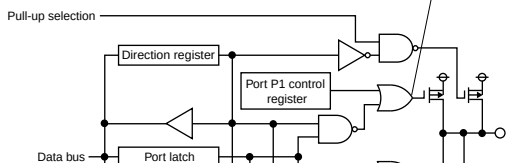
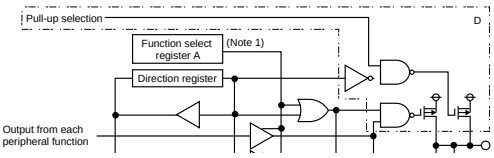
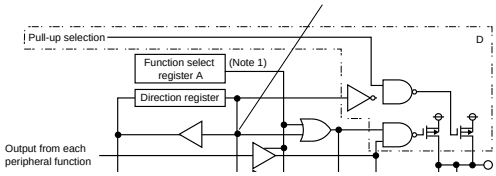
| Rev. Date | Page | Description | |
|---|---|---|---|
| | | Errror | Correct |
| B1<br>1 / 8 / 2001 | | | 100-pin version is added.<br>Flash memory version is added.<br>Others |
| B1<br>30/8/ 2001 | 2,3 | Tables 1.1.1 and 1.1.2<br>Interrupt: 12 internal/external sources<br>(intelligent I/O and CAN module)<br>Supply voltage 3.0 to 3.6V (f($X_{IN}$)=20MHz | <br><br>Delate<br>without wait) add |
| | 3 | A-D converter<br>10 bits (8 channels) x 2 circuits, max 26 inputs | <br>10 bits x 2 circuits, standard 10 inputs, max 26 inputs |
| | 7 | Table 1.1.3 Pin 26 | $CAN_{IN}$ addition |
| | 10-12 | Figures 1.1.4, 1.1.5, Table 1.1.7 | $CAN_{IN}$ is added to Pin 17(GP) and pin 19(FP) |
| | 11 | Figure 1.1.5 Pin 97 $AN0_0$ | $AN_0$ |
| | 12 | Pin 32 (FP) Vcc<br>Pin 34 (FP) Vss | Delate<br>Delate |
| | 13 | Vcc position to pin 64(FP)<br>Vss position to pin 66(FP)<br>RxD4/SCL4/STxD4 position to pin 98 (FP) | Pin 62<br>Pin 64<br>Pin 100 |
| | 14 | Table 1.1.5 $AN2_0$ to $AN2_7$<br>$AN3_0$ to $AN3_7$ | $AN0_0$ to $AN0_7$<br>$AN2_0$ to $AN2_7$ |
| | 17 | Table1.1.12 $P12_0$ to $P12_7$ ISCLK description<br>$AN1_0$ to $AN1_7$ | Delate<br>$AN15_0$ to $AN15_7$ |
| | 18 | Figure 1.1.6 System clock oscillation circuit | PLL oscillation stop detect addition |
| | 28, 29 | Figure 1.4.3 (122), (167)<br>(123), (168) | Group0 receive buffer register, Group1 receive buffer register<br>Group0 transmit buffer/receive data register, Group1 transmit buffer/receive data register |
| | 46 | Note 1: Addresses $03C9_{16}$, $03CB_{16}$ to $03D3_{16}$ | Addresses $03A0_{16}$, $03A1_{16}$, $03B9_{16}$, $03BC_{16}$, $03BD_{16}$, $03C9_{16}$, $03CB_{16}$ to $03D3_{16}$ |
| | 48 | Figure 1.6.1 Note 2 | Addition. Displase after the former Note 2 |
| | 70 | Figure 1.8.6 When reset of PLL control register<br>0X11 0100 | 0<br>0011 0100 |
| | 72 | Figure 1.8.8 Count value set bit<br>Count start bit<br>Count stop/start<br>Note 2 | Division rate select bit<br>Operation enable bit<br>Divider stops/starts<br>Delate |
| | 76 | Line 10 Addition | Stop mode is canceled before setting this bit to "1". |
| | 77<br>135 | Line 8 1:Sub clock is selected<br>Figure 1.14.2 Values that can be set Pulse width modulation mode (8-bit PWM)<br>$00_{16}$ to $FF_{16}$(High-order and low-order address) | 1: Clock from ring oscillator is selected<br><br>$00_{16}$ to $FE_{16}$(High-order address) $00_{16}$ to $FF_{16}$(Low-order address) |
| | 230<br>266 | Line 5, Bit 1 TrmActive<br>Table1.23.11 Waveform generate control register<br>1when clock synchronous serial I/O | TrmData |
| | 280 | -<br>Table1.23.17 Note 1: | √<br>When the transfer clock and transfer data are trans- |

| Rev. Date | Page | Description | |
|---|---|---|---|
| | | Errror | Correct |
| B1 30/8/ 2001 (continue from preced- ing page) | | | mission, transfer clock is set to at least 6 divisions of the base timer clock. Except this, transfer clock is set to at least 20 divisions of the base timer clock. |
| | | Note 2 | Addition |
| | 285 | Figure 1.23.37 Delay timing of base timer | |
| | 284 | Table1.24.1 A-D conversion start condition | |
| | | • Timer B2 interrupt | • Timer B2 interrupt occurrences frequency counter overflow |
| B2 Feb/1/ 2002 (con- tinue from preced- ing page) | 2, 3, 4 | Table 1.1.1, 1.1.2 | |
| | | Clock generating circuit | 4 built-in...circuit 3 built-in clock generation circuits |
| | | PLL freq. synthe. | Delete |
| | | Power consumption 29mA | 26mA |
| | | 44mA | 38mA |
| | 6,10, 11 | Fig 1.1.3-1.1.5 Note: P7$_0$ and P7$_1$ are N-channel...output.-> Add | |
| | 18 | Fig 1.1.6 System clock generator | |
| | | PLL | Delete |
| | | Oscillation stop detection | Ring oscillator |
| | 24 | 7th line | Since the value.....due to the interruption. -> Add |
| | 27 | Fig 1.4.3 (1) | |
| | | (2) Processor mode register 1 | XX00 X000 -> X000 00XX |
| | | (3) System clock control register 0 | 80 -> 0000 X000 |
| | | (10) Oscillation stop detect register | XXXX 0000 -> 00 |
| | | (17) VDC control register 1 | Add |
| | | (21) DRAM refresh interval set register | XXXX ?000 -> ?? |
| | | (46) CAN interrupt 1 control register | Add |
| | | (47) CAN interrupt 2 control register | Add |
| | 28 | Fig 1.4.3 (2) | |
| | | (70) CAN interrupt 0 control register | Add |
| | 28-31 | Fig 1.4.3(2) (97)-(104), Fig 1.4.3(3) (142)-(149), Fig 1.4.3(4) (187)-(194), Fig 1.4.3(5) (222)-(229) Group 0 -3 time measurement/ waveform generation register 0-7 | 00 -> ?? |
| | 29, 30 | Fig 1.4.3(3) (124), Fig 1.4.3(4) (169) | |
| | | Group 0,1 SI/O communication buffer register | Group 0,1 SI/O receive buffer register |
| | | Fig 1.4.3(3) (125), Fig 1.4.3(4) (170) | |
| | | Group 0,1 receive data register | Group 0,1 transmit buffer/receive data register |
| | | (129) Group 0 SI/O comm cont register | X000 XXX -> 000 X011 |
| | | (186) Group 1 SI/O expansion trans cont register | 0000 00XX -> 0000 0XXX |
| | 31 | Fig 1.4.3(5) (238)-(241) | |
| | | Group 3 waveform generate mask register 4-7 | 00 -> ?? |
| | 32 | Fig 1.4.3(6) (270)-(308) | Note added |
| | | (270)-(302) | Reset value changed |
| | 33 | Fig 1.4.3(7) (309)-(338) | Note added |
| | | (314)-(318),(321),(323),(329),(331),(336) | Reset values changed |
| | | (337) CAN0 clock control register | CAN0 sleep control register |
| | 36 | Fig 1.4.3(10) (461) A-D control register 2 | X000 XXX0 -> X000 0000 |

| Rev. Date | Page | Description | |
|---|---|---|---|
| | | Errror | Correct |
| B2 Feb/1/ 2002 (continue from preced-ng page) | 38 | Address 007F16 CAN interrupt 1 control register added<br>Address 008116 CAN interrupt 2 control register added<br>Address 009D16 CAN interrupt 0 control register added | |
| | 61 | (10) Software wait, 11th line<br>SFR area is accessed.....with “2 waits”. | <br>Add |
| | 67 | Fig 1.8.2 System clock control register 0<br>When reset: 0816<br>Note 3: When selecting fc,.....as input port. | <br>0000 X0002<br>Delete |
| | 79 | Fig 1.8.9<br>Note 7: When using PLL.....cannot be used. | <br>Delete |
| | 90 | Fig 1.9.3, Symbol CAN0ICi | CANiIC |
| | 110 | Table 1.11.1, DMA request factors | Intelligent I/O interrupt -> add |
| | 128 | Fig 1.12.4, the number of cycles | Change |
| | 133 | Fig 1.14.3, Timer Ai mode register, MR0<br>Port output.....registers A and B. | <br>Port output.....registers A, B and C. |
| | 137, 138, 142, 144 | Table 1.14.1, 1.14.2, 1.14,4, 1.14,5<br>TAiOUT pin function | <br>Function select register C -> add |
| | 137 | Fig 1.14.7 Timer Ai mode register<br>bit 2 (MR0)<br>Location of Note 3 (b7, b6): 11 | <br>Function select register C -> add<br>10 |
| | 139 | Fig 1.14.8 Timer Ai mode register<br>bit 2 (MR0) | <br>Function select register C -> add |
| | 143, 145 | Fig 1.14.11, 1.14.12 Timer Ai mode register<br>bit 2 (MR0)<br>Location of Note 3 (b7, b6): 11 | <br>Function select register C -> add<br>10 |
| | 159 | Fig 1.16.5 Timer Ai mode register<br>bit 2 (MR0) | <br>Function select register C -> add |
| | 161 | Fig 1.16.6 Reload register<br>n = 1 to 255 | Reload register |
| | 172 | Fig 1.17.4 UARTi transmit/receive control register 0<br>Note 2 | <br>Function select register C -> add |
| | 173 | Fig 1.17.5 UARTi transmit/receive control register 1<br>Function of bit 7: Error signal output enable bit | <br>Set to “0” |
| | 199 | Fig 1.22.1 Clock control register<br>Time stamp count register | Sleep control register<br>Time stamp register |
| | 200 | Fig 1.22.3 Bit 4 0: Forced reset<br>Bit 10 Time stamp count reset bit | 0: Reset requested<br>Time stamp counter reset bit |
| | 201 | 5th line: In no case will the CAN module be .....<br>Bit 3: BasicCAN mode bit | In no case will the CAN be.....<br>Bit 3: BasicCAN mode select bit |
| | 202 | Bit 8,9: Timestamp prescaler bits<br>Bit 11, 1st line: Receive Error Counter<br>Transmit Error Counter | Bit 8, 9: Timestamp prescaler select bits<br>Receive Error Counter Register<br>Transmit Error Counter Register |
| | 209 | Fig 1.22.8 bit 4: Reserved bit | Sampling number |
| | 210 | 6. CAN0 configuration register | Explanation of Bit 4 -> add |

| Rev. Date | Page | Description | |
|---|---|---|---|
| | | Errror | Correct |
| B2 Feb/1/ 2002 (con-tinue from preced-ing page) | 211 | Note:1 Setting the C0CTLR0 register's Reset0 bit to 1 resets the CAN protocol control unit, with the C0TSR register thereby initialized to $0000_{16}$. Also, setting the TSReset (timestamp count reset) bit to 1 initializes the C0TSR register to $0000_{16}$ on-the-fly (while the CAN protocol control unit remains operating). | Note 1: Setting the C0CTLR0 register's Reset0 and Reset1 bits to 1 resets the CAN, and the C0TSR register is thereby initialized to $0000_{16}$. Also, setting the TSReset (timestamp counter reset) bit to 1 initializes the C0TSR register to $0000_{16}$ on-the-fly (while the CAN remains operating; CAN0 status register's State_Reset bit is "0"). |
| | 212 | Tq period = (C0BRP+1) | Tq period = (C0BRP+1)/CPU clock |
| | 220 | Fig 1.22.19 b0<br>b2 | b2<br>b1 |
| | 226 | Fig 1.22.25 bit 0<br>bit 1, When transmit, TrmData<br>bit 3<br>bit 6, 7, Transmit request flag | Note 2 -> add<br>When transmit, TrmActive<br>Note 2 -> add<br>Transmit request bit |
| | 229 | Fig 1.22.26, explanation of function | Change |
| | 230, 231, 232 | Fig 1.22.27, 1.22.28, 1.22.29<br>Explanation of function | Message slot j (j=0 to 15) -> change |
| | 233 | Fig 1.22.30, CAN0 message slot butter i data m<br>Symbol C0SLOT0_m (m=0 to 3)<br>C0SLOT0_m (m=4 to 7)<br>C0SLOT1_m (m=0 to 3)<br>C0SLOT1_m (m=4 to 7) | <br>C0SLOT0_n (n=m+6, m=0 to 3)<br>C0SLOT0_n (n=m+6, m=4 to 7)<br>C0SLOT1_n (n=m+6, m=0 to 3)<br>C0SLOT1_n (n=m+6, m=4 to 7) |
| | 235 | Table 1.23.1 Group 2, WG register<br>Group 3 Comm shift register | - -> 8chs<br>16bits x 2chs -> - |
| | 240 | Fig 1.23.5, Group i base timer cont reg 0<br>Bit 2 to bit 6, explanations on $f_{PLL}$ | <br>Delete |
| | 245 | Table 1.23.2, Count reset condition, Group 2, 3<br>(3) Reset request ..... circuit | <br>(3) Reset request ..... circuit (group 2 only) |
| | 245 | Fig 1.23.10 $f_{PLL}$ | Delete |
| | 246 | Fig 1.23.11 | Newly added |
| | 248 | Fig 1.23.13, the values when reset: $00_{16}$ | $0000_{16}$ |
| | 249 | Table 1.23.3, select function, digital filter function<br>Strips off pulses less than 3 cycles long from $f_1$ and the base timerclock. | <br>Pulses will pass when they match either $f_1$ or the base timerclock 3 times. |
| | 250 | Fig 1.23.14, (c) Change | |
| | 252 | Fig 1.23.16, reset values for both registers | $0000_{16}$ -> $XXXX_{16}$ |
| | 256 | Fig 1.23.20, When WG register is "$xxxb_{16}$" | When WG register is "$xxxa_{16}$" |
| | 270 | Table 1.23.12<br>Transmission start condition<br>• Write data to transmit buffer register<br>Interrupt request generation timing<br>•When transmitting<br>- When SI/O transmit buffer register is.....<br>•When receiving<br>When....to SI/O communication buffer register | <br><br>• Write data to transmit buffer<br><br><br>- When transmit buffer is .....<br><br>When.....to SI/O receive buffer register |

| Rev. Date | Page | Description | |
|---|---|---|---|
| | | Errror | Correct |
| B2 Feb/1/ 2002 (con-tinue from preced-ing page) | 270 | Select function<br>This.....TxD pin output and RxD pin input. | <br>This.....ISTxD pin output and ISRxD pin input. |
| | 271 | Table 1.23.13, Transfer clock input<br>•Selects I/O with function..... | <br>•Select I/O port with function..... |
| | 271 | Fig 1.23.31<br>Write to communication buffer<br>(Input to INPC2/ISRxD0 pin) | <br>Write to receive buffer<br>(Input to INPCi2/ISRxDi pin (i=0, 1)) |
| | 272 | Table 1.23.14<br>Transmission start condition<br>• Write data to transmit buffer register<br>Interrupt request generation timing<br>•When transmitting<br>- When SI/O transmit buffer register is.....<br>•When receiving<br>When....to SI/O communication buffer register<br>Error detection<br>• Overrun error:<br>.....before contents of receive buffer register.... | <br><br>• Write data to transmit buffer<br><br><br>- When transmit buffer is .....<br><br>When.....to SI/O receive buffer register<br><br><br>.....before contents o SI/O receive buffer register..... |
| | 273 | Fig 1.23.32<br>Write to communication buffer | <br>Write to receive buffer |
| | 273 | Fig 1.23.33<br>(Input to INPC2/ISRxD0 pin) | <br>(Input to INPCi2/ISRxDi pin (i=0, 1)) |
| | 279 | Table 1.23.17<br>Transmission start condition<br>• Write data to transmit buffer register<br>Reception start condition<br>• Write data to transmit buffer register<br>Interrupt request generation timing<br>•When receiving<br>When....to SI/O communication buffer register<br>Select function<br>This.....TxD pin output and RxD pin input. | <br><br>• Write data to SI/O transmit buffer register<br><br>• Write data to SI/O transmit buffer register<br><br><br>When.....to SI/O receive buffer register<br><br>This.....ISTxD pin output and ISRxD pin input. |
| | 286 | Fig 1.24.4, A-D control register 2<br>When reset: X000 XXX02 | <br>X000 00002 |
| | 287, 288 | Fig 1.24.5, Note 4 and Fig 1.24.6, Note 3<br>..... by A-D sweep pin select bits..... | <br>.....by analog input port select bits..... |
| | 292 | (e) Replace function of input pin<br>2nd line: .....of A-D0 and A-D2. | <br>.....of A-D0 and A-D1. |
| | 293 | (f) , at the end of 2nd line<br>(g) 3rd line: ....., input via AN00 to AN07 is..... | as AN0.....respectively. -> add<br>, input via AN0 to AN7 is..... |
| | 294 | Table 1.24.9 P00 analog input<br>P01 analog input | P95 analog input<br>P96 analog input |
| | 312 | Fig 1.29.1, P00 to P07, P20 to P27: - | ○ |

| Rev. Date | Page | Description | |
|---|---|---|---|
| | | Errror | Correct |
| B2 Feb/1/ 2002 (con- tinue from preced- ing page) | 313 | Fig 1.29.2<br>  <br>Circuit (C)<br>P15 to P17, Circuit (B): - | Delete<br>  <br>Delete<br>○ |
| | 314 | Fig 1.29.3<br>  <br>P121, P122, Circuit (B): - | Add<br>  <br>○ |
| | 326 | Fig 1.29.16, Pull-up register 2, Note 1 | Delete |
| | 331 | Table 1.29.5<br>Bit 0, 1: Three-phase PWM output $\overline{(U)}$<br>Bit 1, 0: Three-phase PWM output (U) | <br>1: Three-phase PWM output (U)<br>0: Three-phase PWM output $\overline{(U)}$ |
| | 331 | Table 1.29.6, PS4<br>PSL4<br>Bit 1, UART0<br>Bit 2, UART4<br>Bit 3, UART1<br>Bit 4, 5 UART1 UART4<br>A4<br>B4 | PS3<br>PSL3<br>UART3<br>UART3<br>UART3<br><br>A3<br>B3 |
| | 334 | VDC | Add |
| | 337 | A-D Converter<br>1st line: A-D<br>1st line: A-D<br>2nd line: .....and to bit 0 of A-D control register 2... | <br>A-D i (i=0,1)<br>A-D i<br>.....and to each bit of A-D i control register 2..... |
| | 340 | (3) External interrupt<br>• Level sense, 2nd line: (When XIN=20MHz and..)<br>3rd line: (....., at least 250 ns.....)<br>• When the polarity of INT0 to INT5 pins is..... | <br>(When XIN=30MHz and .....)<br>(....., at least 233 ns.....)<br>• When the polarity of $\overline{INT0}$ and $\overline{INT5}$ pins is..... |
| | 341 | Reducing power consumption, (2)<br>1st line, last line: AN04, AN07 | <br>AN4, AN7 |
| | 343 | Table 1.30.3 G0CR 00EF16<br>G1RI 012F16<br>U0BRG 036116<br>U0TB 036316, 036216<br>U1BRG 036916<br>U1TB 036B16, 036A16 | G0RI 00EC16<br>G1RI 012C16<br>U0BRG 036916<br>U0TB 036B16, 036A16<br>U1BRG 02E916<br>U1TB 02EB16, 02EA16 |
| | 343 | Notes on CNVss pin reset at “H” level | Add |

| Rev. Date | Page | Description | |
|---|---|---|---|
| | | Errror | Correct |
| B2 Feb/1/ 2002 (continue from preced-ng page) | 344-380 | Electric characteristics | Add |
| | 385 | Fig 1.34.1, Address $0377_{16}$<br>Bit 0: RY/$\overline{BY}$ status bit | Address $0057_{16}$<br>RY/$\overline{BY}$ signal status bit |
| | 385 | Flash memory control register (address $0057_{16}$)<br>1st line: .....the RY/$\overline{BY}$ status flag..... | .....the RY/$\overline{BY}$ signal status bit..... |
| | 390 | 13th line of Page Program Command ($41_{16}$) and<br>Fig 1.34.3: RY/$\overline{BY}$ status flag | RY/$\overline{BY}$ signal status bit |
| | 391 | 11th line of Block Erase Command ($20_{16}$/$D0_{16}$)<br>and Fig 1.34.4: RY/$\overline{BY}$ status flag | RY/$\overline{BY}$ signal status bit |
| | 392 | Fig 1.34.5: RY/$\overline{BY}$ status flag | RY/$\overline{BY}$ signal status bit |
| | 400 | 3rd paragraph, 1st line<br>....., set the $CLK_1$ pin to "H" level and.... | .....,set the $CLK_1$ pin to "H" level and the TxD1 pin to "L" level, and..... |
| | 400 | 3rd paragraph, 2nd line<br>The $CLK_1$ pin is connected to Vcc.....resistance. | Add |
| | 401 | $P6_7$ When using standard.....transfer. | Add |
| | 419 | Fig 1.35.22, Data output | Pulled down |
| | 421 | How frequency is identified, 2nd line: (2 - 20MHz) | (2 - 30MHz) |
| B3 28/3/ 2002 | 2, 3 | Table 1.1.1-2 Power consumption<br>26mA (f($X_{IN}$)=20MHz without software wait,Vcc=5V) | 26mA (f($X_{IN}$)=20MHz without software wait,Vcc=3V) |
| | 37 | SFR registers naming RAMD1, RAMD2, RAMD3 | RMAD1, RMAD2, RMAD3 |
| | 27, 37 | $001B_{16}$ VDC control register 1 VDC1<br>$001F_{16}$ VDC control register 0 VDC0 | VDC control register 0 VDC0<br>VDC control register 1 VDC1 |
| | 39, 40 | GiBF<br>GiDR | GiRB<br>GiTB |
| | 82 | (4) VDC control register 0 (address $001F_{16}$)<br>VDC control register 1 (address $001F_{16}$) | VDC control register 0 (address $001B_{16}$)<br>Add |
| | 92, 122 | Figure 1.9.5, 1.12.1 RLVL register Note 3:<br>When this bit is 1, do not set the high-speed interrupt select bit to 0. | When this bit is 1, set the high-speed interrupt select bit to 0. |
| | 248 | Figure 1.23.13. GiTMj register<br>When reset $0000_{16}$ | Indeterminate |
| | 247 | Figure 1.23.11 GiTMCRj register CST0, CST1 | CTS0, CTS1 |
| | 252 | Figure 1.23.16. GiPOj register, G3MKj register<br>When reset $XXXX_{16}$ | Indeterminate |
| | 258 | Figure 1.23.18. GiPOCR register<br>When reset $0X00\ X000_2$ | $00_{16}$ |
| | 270 | Figure 1.23.26 GiBF register | GiRB register |
| | 269 | Figure 1.23.30 GiDR register | GiTB register |
| | 273 | Line 4<br>TxD output polarity reverse select bit is set to "0" (not to reverse) for usual use. | TxD output polarity reverse select bit is set to "1" (reverse) for usual use. |
| | 328 | Figure 1.29.18 IPS register when reset $0000\ 0X00_2$ | $00_{16}$ |
| | 412 | Figure 1.35.12 RxD1 $0F_{16}$ | $FF_{16}$ |
| | | | |

| REVISION HISTORY | M32C/83 GROUP DATA SHEET |
|---|---|

| Rev. Date | Page | Description | |
|---|---|---|---|
| | | Old version | New version |
| B3 28/3/ 2002 | 152 | Table 1.15.3<br>Interrupt request generation timing<br>•••• 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.) | •••• Assume that the count start flag condition is “1” and then the Timer Bi overflow flag becomes “1”. If the Timer Bi mode register has a write-access after next count cycle of the timer from the above condition, the Timer Bi overflow flag becomes “0”. |
| | 173 | Figure 1.17.5 Bit 7 of UiSMR rgister<br>0: Divided-by-2<br>1: No divided | <br>0: No divided<br>1: Divided-by-2 |
| | 174 | Figure 1.17.6 Bit 7 of UiSMR2 register<br>Clock divide synchronizing enable bit | <br>External clock synchronizing enable bit |
| | | Delate “parity” in intelligent I/O | |
| | 29,30 | Addresses $00E9_{16}$, $0129_{16}$ X000 XXXX | XX00 XXXX |
| | 236, | Figures 1.23.1, 1.23.2 | |
| | 237 | Parity bit generation circuit | Delate |
| | | Parity bit check | Delate |
| | 266 | Figure 1.23.27 GiRB register | |
| | | When reset Indeterminate | XX00 XXXX XXXX $XXXX_2$ |
| | | Parity error flag | Nothing is assigned. •••• |
| | | Odd/even parity select bit | Reserved bit |
| | | Parity enable select bit | Reserved bit |
| | 272 | Table 1.23.14 | |
| | | Transfer data format • Parity bit | Delate |
| | | Error detection • Parity error | Delate |
| | | Select function • Parity | Delate |
| | 399 | Figure 1.34.8 ROMCP2 | Reserved bit |
| B3 12/6/ 2002 | Header | 16-BIT MICROCOMPUTER | 16/32-BIT MICROCOMPUTER |
| | 346 | Figures 1.32.3, 1.32.23 | |
| | 364 | "H" output voltage $X_{COUT}$ 3.0V | 3.3V |
| | 347 | Figures 1.32.4, 1.32.24 Ladder resistance | |
| | 365 | 10K | 8K |
| | 345- | When 5V, Vcc = 5V, Topr = 25 ºC | Vcc=4.2V to 5.5V, Topr=20 to 85 ºC |
| | 351, 369 | CM15=“1” | Delate |
| | 356- | Vcc=5V±10% | 4.2V to 5.5V |
| | 364- | Vcc voltage 3V<br>When 3.3V, Vcc=3.3V Topr=25ºC | 3.3V<br>Vcc=3.0V to 3.6V Topr=-20 to 85ºC |
| | 373 | Vcc=3V±10% Vcc=3.0V to 3.6V | |

## Keep safety first in your circuit designs!

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

## Notes regarding these materials

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

MITSUBISHI SEMICONDUCTORS
M32C/83 Group DATA SHEET REV. B3

June Second Edition 2002

Editioned by
Committee of editing of Mitsubishi Semiconductor DATA SHEET

Published by
Mitsubishi Electric Corp., Kitaitami Works

This book, or parts thereof, may not be reproduced in any form without permission of Mitsubishi Electric Corporation.
©2002 MITSUBISHI ELECTRIC CORPORATION