查询M30222供应商
Under development



## Description

The M30222 single-chip microcomputers are built using the high-performance silicon gate CMOS process using a M16C/60 Series CPU core and are packaged in a 100-pin plastic molded QFP. These single-chip microcomputers operate using sophisticated instructions featuring a high-level of instruction efficiency and are capable of executing instructions at high speed. They also feature a built-in multiplier and DMAC, making them ideal for controlling office, communications, industrial equipment, and other high-speed processing applications.

The M30222 group includes a range of products with various package types.

### Features

- Memory capacity ..... Flash ROM 260 Kbytes
  ..... RAM 20 Kbytes
- Shortest instruction execution time ..... 62.5ns (f(XIN)=16MHz)
- Supply voltage ..... 2.7 to 5.5V
- Low power consumption ..... TBD
- Interrupts ..... 25 internal and 8 external interrupt sources
  4 software interrupt sources
  7 levels (including key input interrupt)
- Multifunction 16-bit timer ..... 5 output timers, 6 input timers, three phase motor control, real-time port
- Serial I/O ..... 5 channel
  3 for UART or clock synchronous (1 channel for $I^2C$ or SPI)
  2 for clock synchronous
- DMAC ..... 2 channels (trigger: 24 sources)
- A-D converter ..... 10 bits X 8 channels (expandable up to 10 channels)
- D-A converter ..... 8 bits X 2 channels
- CRC calculation circuit ..... 1 circuit
- Watchdog timer ..... 1 timer
- Key-on Wake up ..... 8 inputs
- Programmable I/O ..... 54 lines
- Input port ..... 1 line (P83 shared with $\overline{NMI}$ pin)
- Clock generating circuit ..... 2 built-in clock generation circuits
  (built-in feedback resistor, and external ceramic or quartz oscillator)
- LCD Drive ..... 1/2, 1/3 bias
  4 common outputs
  40 segment outputs
  Built-in charge pump
  1/2, 1/3, 1/4 duty
  Expansion CLK output
  Static/direct drive mode

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

### Applications

Audio, cameras, office, industrial, communications and, portable equipment

# Table of Contents

Description ..... 1-2
Operation of Functional Blocks ..... 1-10
Memory ..... 1-10
Central Processing Unit (CPU) ..... 1-11
Reset ..... 1-14
Special function registers ..... 1-15
Software Reset ..... 1-20
Clock generating Circuit ..... 1-21
Clock Output ..... 1-25
Wait Mode ..... 1-26
Stop Mode ..... 1-27
Status Transition Of BCLK ..... 1-28
Voltage Down Converter ..... 1-30
Power control ..... 1-32
Protection ..... 1-34
Software wait ..... 1-35
Overview of Interrupts ..... 1-36
Watchdog Timer ..... 1-57
DMAC ..... 1-59
Timers ..... 1-69
Timer A ..... 1-71
Timer B ..... 1-85
Timer functions for three-phase motor control ..... 1-93
Serial Communications ..... 1-105
(1) Clock synchronous serial I/O mode ..... 1-114
(2) Clock Asynchronous Serial I/O (UART) Mode 1-120
UART2 in I2C Mode ..... 1-130
UART2 in SPI mode ..... 1-138
S I/O 3, 4 ..... 1-143
LCD Drive Control Circuit ..... 1-147
A-D Converter ..... 1-157
D-A Converter ..... 1-168
CRC Calculation Circuit ..... 1-170
Programmable I/O Ports ..... 1-172
Electrical Characteristics ..... 1-179
Flash Memory ..... 1-186
CPU Rewrite Mode ..... 1-188
Parallel I/O Mode ..... 1-202
Standard serial I/O mode 1 ..... 1-206
Standard serial I/O mode 2 ..... 1-226

## Pin Configuration

Figure 1.1 shows the pin configurations for M30222 group.

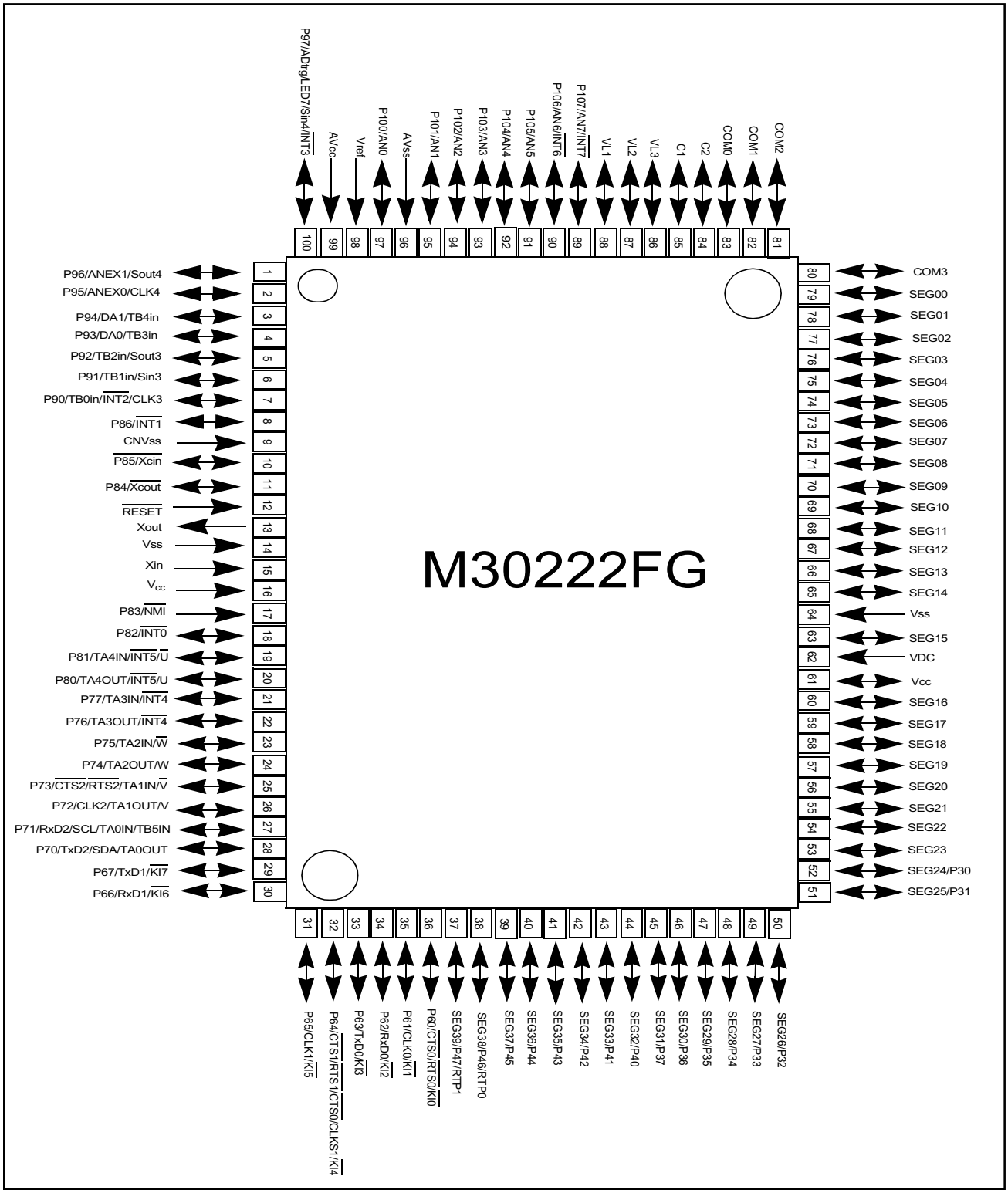
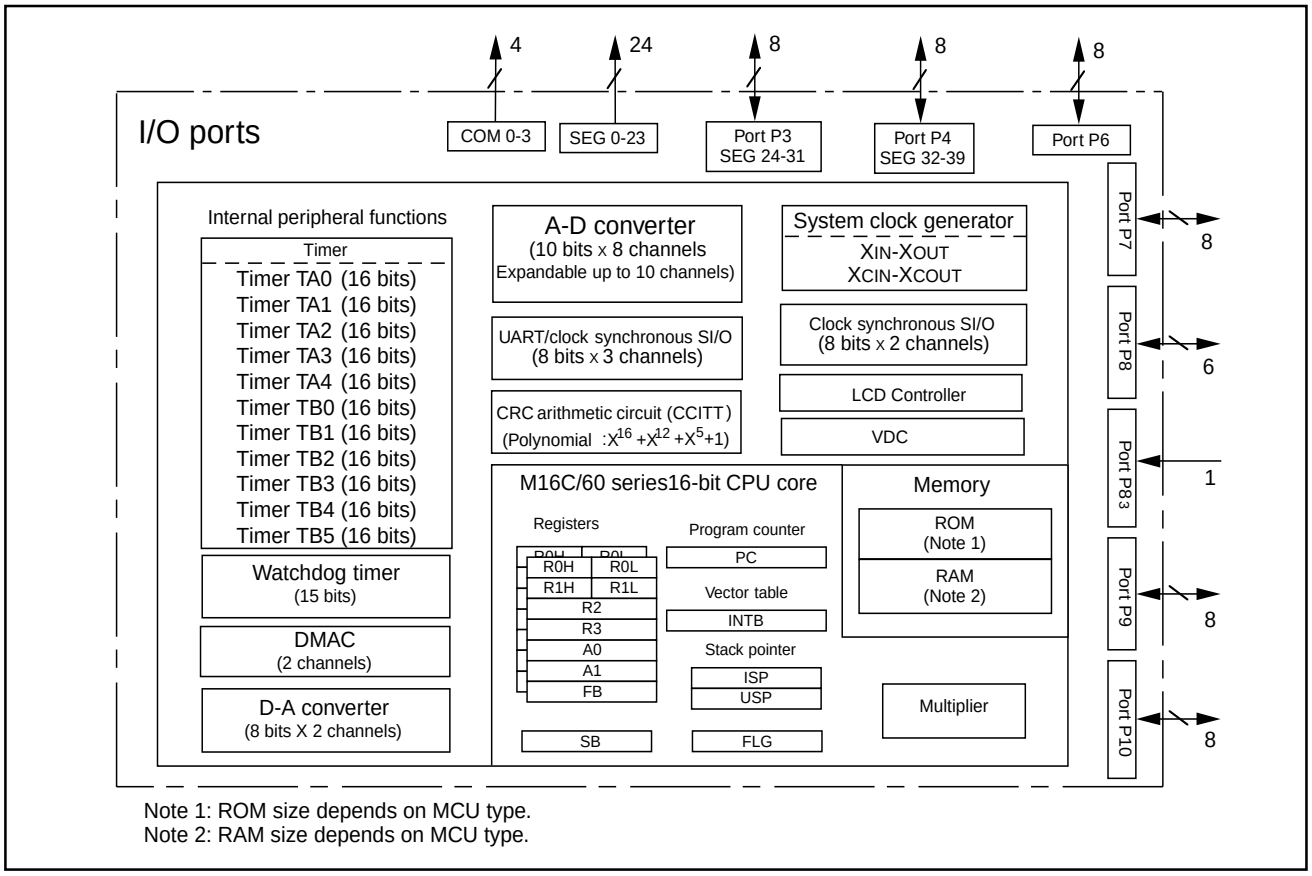



**Fig. 1.1. Pin configuration (top view)**

## Block Diagram

Figure 1.2 is a block diagram of the M30222 group.




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

**Fig. 1.2. Block diagram of M30222 group**

## Memory Expansion

Figure 1.3 shows the Memory expansion for the M30222 group.

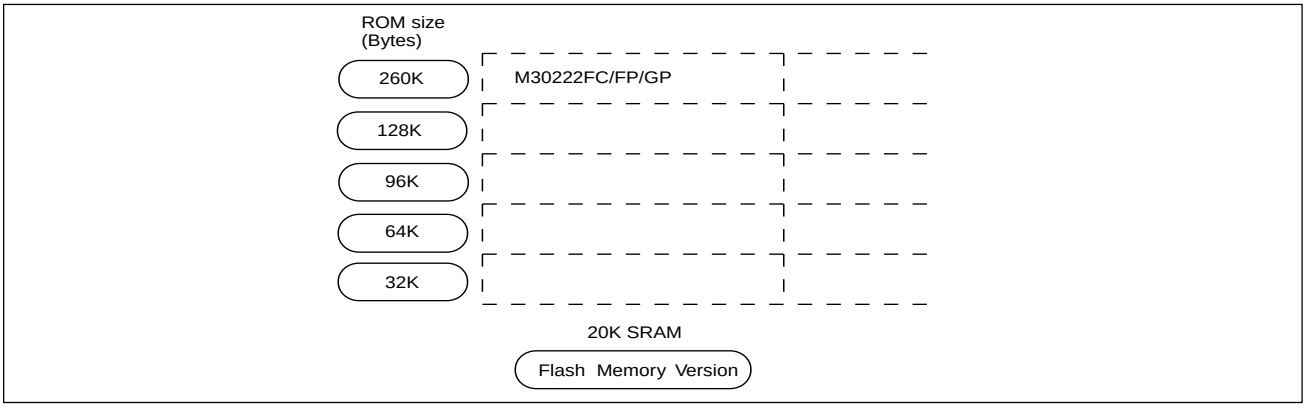



**Fig. 1.3. Memory Expansion**

## Performance Outline

**Table 1.1. Performance outline of the M30222 group**

| Parameters | | | Functions |
|---|---|---|---|
| Number of basic instructions | | | 91 |
| Shortest instruction execution time | | | 62.5ns f(Xin) = 16MHz |
| Memory size | ROM | | 260K bytes |
| | RAM | | 20K bytes |
| Input/Output | P3-P4, P6-P10 except P83 | I/O | 8 bits x 6, 7 bits x 1 |
| | P83 | I | 1 bit x 1 |
| Multifunctional timer | TA0, TA1, TA2, TA3, TA4 | | 16 bits x 5 |
| | TB0, TB1, TB2, TB3, TB4, TB5 | | 16 bits x 6, three-phase motor control |
| Serial I/O | UART0, UART1, UART2 | | (UART or clock synchronous) x 3, or $I^2C$ x 1 |
| | SIO3, SIO4 | | (Clock synchronous) x 2 |
| A-D converter | | | 10 bits x (8 + 2) channels |
| D-A converter | | | 8 bits x 2 |
| CRC calculation circuit | | | CRC-CCITT |
| Watchdog timer | | | 15 bits x 1 (with prescaler) |
| Interrupts | | | 25 external, 8 internal sources, 4 software, 7 levels |
| Clock generating circuit | | | 2 built-in clock generation circuits |
| Supply voltage | | | 2.7 to 5.5V f(Xin) = 16 MHz, without software wait |
| Power consumption | | | TBD |
| I/O characteristics | I/O withstand voltage | | 5.5V |
| | Output current | P3, P4 | 0.1 mA (high output), 2.5 mA (low output) |
| | | P6-P10 | 5 mA at 5V (excluding pins $P7_0$, $P7_1$, $P8_3$) |
| Device configuration | | | CMOS high performance silicon gate |
| Package | | | 100-pin plastic mold QFP |
| LCD | COM0 to COM3 | | 4 lines |
| | SEG0 to SEG39 | | 40 lines (16 lines shared with I/O ports) |

Mitsubishi plans to release the following products in the M30222 group:

(1) Support for Flash memory version and mask ROM versions

(2) ROM capacity: 260 K bytes

(3) Package

100P6S-A : Plastic molded QFP (mask ROM version)

100P6Q-A: Plastic molded QFP

## M16C Family Group

Figure 1.4 shows the M30222 family.

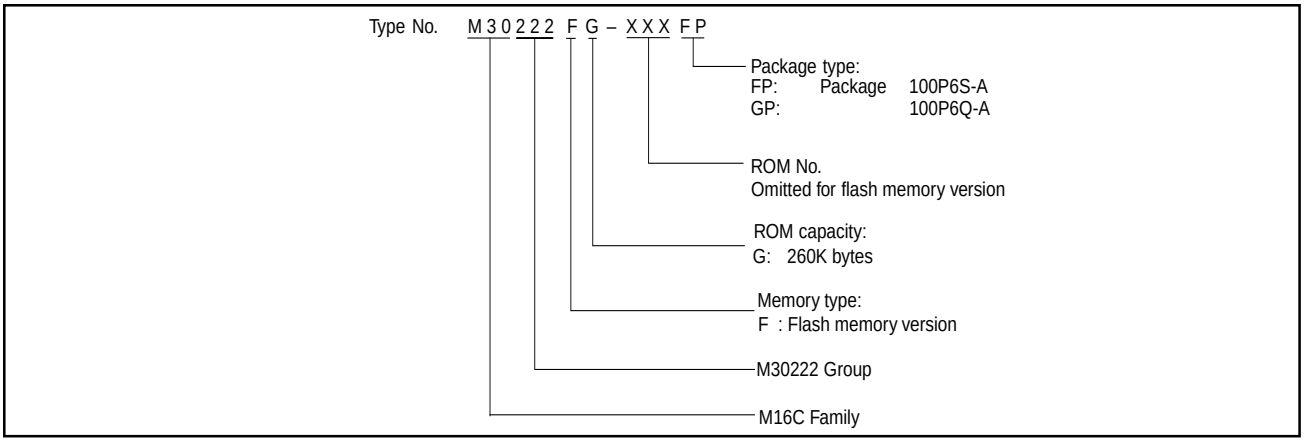



**Fig. 1.4. Type No., memory size, and package**

Table 1.2 shows the product list for the M30222 family.

**Table 1.2. Product list**

| Type No. | ROM Capacity | RAM Capacity | Package Type | Remarks |
|---|---|---|---|---|
| M30222FGFP | 260 Kbytes | 20 Kbytes | 100P6S-A | Flash |
| M30222FGGP | | | 100P6Q-A | |

## Pin Description

### Table 1.3. Pin Description for M30222 group

| Pin name | Signal name | I/O type | Function |
|---|---|---|---|
| Vcc, Vss | Power supply | Input | Supply 2.7 to 5.5V to the Vcc pin and 0V to Vss |
| VDC | Voltage Down Converter | Input | Connects capacitor from VDC to Vss; or if not using VDC, connect 3.3V to VDC pin. |
| CNVss | CNVss | | This pin is used to enable flash programming. Connect the pull-down resistor from CNVss to Vss. Connect CNVss to enable flash programming. |
| $\overline{\text{RESET}}$ | Reset input | Input | An "L" on this input resets the microcomputer. |
| Xin, Xout | Main Clock | Input/Output | These pins are provided for the main clock generating circuit. Connect a ceramic resonator or crystal between the Xin and the Xout pins. To use an externally derived clock, input it to the Xin. |
| $P8_4/P8_5$ | I/O Port | Input/Output | These pins are provided for the subclock generating circuit. Connect a ceramic resonator or crystal between the Xcin pin and leave the Xcout pin open. These pins also function as CMOS I/O ports. |
| Xcout/Xcin | Subclock | Input/Output | |
| AVcc | Analog power supply + reference | Input | This pin is a power supply input for the A-D converter. Connect this pin to Vcc. |
| AVss | Analog power supply + reference | Input | This pin is a power supply input for A-D converter. Connect this pin to Vss. |
| Vref | Reference voltage input | Input | This pin is a reference voltage input for the A-D converter. |
| $P3_0$ to $P3_7$ | I/O Port P3 | Input/Output | This is an 8-bit CMOS I/O port. It has an input/output direction register that allows the user to set each pin for input or output individually. When used for input, the port can be set by software to have or not have a pull resistor in units of four bits. |
| | RTP0_0 to RTP3_1 | Output | |
| | SEG24 to SEG31 | Output | Pins in this port also function as SEG output for LCD and output for Real-time port. |
| $P4_0$ to $P4_7$ | I/O Port P4 | Input/Output | This is an 8-bit I/O port equivalent to P3. |
| | SEG32 to SEG39 | Output | Pins in Port 4 also function as SEG outputs for LCD. |
| | RTP4_0 to RTP7_1 | Output | Pins in Port 4 also function as Real-time port. |
| $P6_0$ to $P6_7$ | I/O Port P6 | Input/Output | This is an 8-bit I/O port equivalent to P3. |
| | $\overline{\text{KI0}}$ to $\overline{\text{KI7}}$ | Input | Pins in Port 6 also function as key-input interrupts. |
| | UART0, UART1 | Input/Output | Pins in Port 6 also function as transmit, receive, clock, and $\overline{\text{CTS}}/\overline{\text{RTS}}$ pins for UART0, UART1. |
| $P7_0$ to $P7_7$ | I/O Port P7 | Input/Output | This is an 8-bit I/O port equivalent to P3. |
| | UART2 | Input/Output | Some pins in Port 7 serve as transmit, receive, clock, and $\overline{\text{CTS}}/\overline{\text{RTS}}$ for UART2. UART2 provides $I^2C$ serial communications. |
| | Timer A/B | Input/Output | Some pins in Port 7 serve as input/output for Timer A and Timer B. |
| | $\overline{\text{INT4}}$ | Input | Pins $P7_6$ and $P7_7$ function as inputs for $\overline{\text{INT4}}$. |
| | Three-phase | Output | Some pins in Port 7 function as three-phase outputs for V, $\overline{\text{V}}$, W, and $\overline{\text{W}}$. |
| $P8_0$ to $P8_2$, $P8_6$ | I/O Port P8 | Input/Output | $P8_0$ to $P8_2$, $P8_6$ are I/O ports equivalent to P3. |
| | Timer A | Input/Output | Some pins in Port 8 serve as input/output for Timer A and Timer B. |
| | $\overline{\text{INT5}}$ | Input | Pins P80 and P81 function as inputs for $\overline{\text{INT5}}$. |
| | Three-phase | Output | Pins $P8_0$ and $P8_1$ function as inputs three-phase outputs for U and $\overline{\text{U}}$. |
| $P8_3$ | $\overline{\text{NMI}}$ | Input | $P8_3$ is an input only port that also functions for $\overline{\text{NMI}}$. The $\overline{\text{NMI}}$ interrupt is generated when the input at this pin changes from "H" to "L". The $\overline{\text{NMI}}$ function cannot be cancelled using software. The pull-up resistor cannot be set for this pin. |

| Pin name | Signal name | I/O type | Function |
|---|---|---|---|
| $P9_0$ to $P9_7$ | I/O Port P9 | Input/Output | This is an 8-bit I/O equivalent to P3. |
| | SIO 3/4 | Input/Output | Pins in Port 9 function as transmit, receive and clock for SIO3 and SIO4. |
| | Timer B | Input | Some pins in Port 9 serve as TB3 and TB4 pins. |
| | D-A | Output | $P9_3$ and $P9_4$ can be configured to function as a digital to analog output. |
| | $\overline{INT2}$, $\overline{INT3}$ | Input | Pin $P9_0$ and $P9_7$ can be configured as $\overline{INT2}$ and $\overline{INT3}$. |
| | ANEX0 | Output | These pins are used to connect to an optional external op amp. |
| | ANEX1 | Input | |
| $P10_0$ to $P10_7$ | I/O Port 10 | Input/Output | This is an 8-bit I/O port equivalent to P3. |
| | AN0 to AN7 | Input | Pins in Port 10 function as analog inputs. |
| | $\overline{INT6}$, $\overline{INT7}$ | Input | $P10_6$ and $P10_7$ function as inputs for $\overline{INT6}$ and $\overline{INT7}$. |
| SEG0 to SEG23 | SEG drive pins | | Pins in this port function as SEG output for LCD drive circuit. |
| COM0 to COM3 | COM ports | | Pins in this port function as COM output for LCD drive circuit. |
| VL1 to VL3 | Power supply for LCD driver | | Power supply input for LCD drive circuit. |

## Operation of Functional Blocks

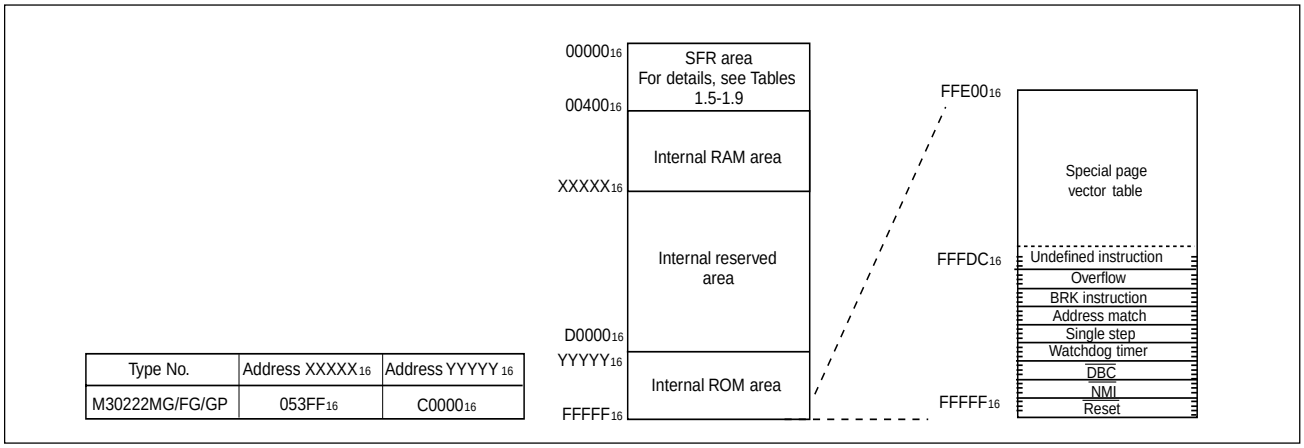
The M30222 group accommodates certain units in a single chip. These units include ROM and RAM to store instructions and data and the central processing unit (CPU) to execute arithmetic/logic operations. Also included are peripheral units such as timers, serial I/O, D-A converter, DMAC, CRC calculation circuit, A-D converter, LCD, and I/O ports. The following explains each unit.

## Memory

Figure 1.5 is a memory map of the M30222 group. The linear address space of 1M bytes extends from address $00000_{16}$ to $FFFFF_{16}$. From $FFFFF_{16}$ down is ROM. For example, in the M30222FG-XXXFP, there is 256K bytes of internal ROM from $C0000_{16}$ to $FFFFF_{16}$. The vector table for fixed interrupts such as the reset and NMI are mapped to $FFFDC_{16}$ to $FFFFF_{16}$. The starting address of the interrupt routine is stored here. The address of the vector table for timer interrupts, etc., can be set as desired using the internal register (INTB). See the section on interrupts for details.

From $00400_{16}$ up is RAM. For example, in the M30222FG-XXXFP, 20K bytes of internal RAM is mapped to the space from $00400_{16}$ to $053FF_{16}$. In addition to storing data, the RAM also stores the stack used when calling subroutines and when interrupts are generated.

The SFR area is mapped to $00000_{16}$ to $003FF_{16}$. This area accommodates the control registers for peripheral devices such as I/O ports, A-D converter, serial I/O, and timers, etc. Tables 1.5 to 1.9 show the location of peripheral unit control registers. Any part of the SFR area that is not occupied is reserved and cannot be used for other purposes.




| Type No. | Address XXXXX$_{16}$ | Address YYYYY$_{16}$ |
|---|---|---|
| M30222MG/FG/GP | $053FF_{16}$ | $C0000_{16}$ |

**Fig. 1.5. Memory Map**

## Central Processing Unit (CPU)

The CPU has a total of 13 registers shown in Figure 1.6. Seven of these registers (R0, R1, R2, R3, A0, A1, and FB) come in two sets; therefore, these have two register banks.

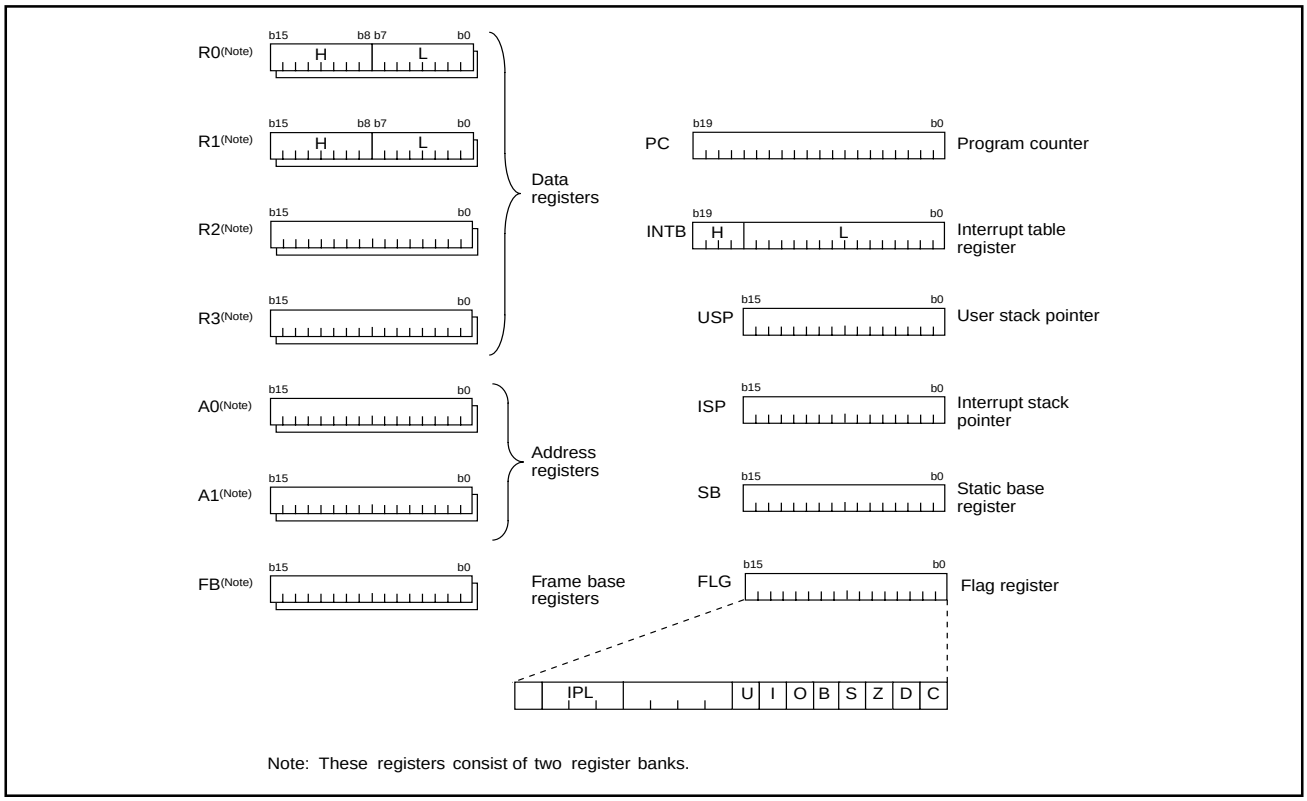



**Fig. 1.6. Central Processing Unit Register**

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

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

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

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

Address registers (A0 and A1) are configured with 16 bits, and have functions equivalent to those of data registers. These registers can also be used for address register indirect addressing and address register relative addressing. In some instructions, registers A1 and A0 can be combined for use as a 32-bit address register (A1A0).

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

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

**(4) Program counter (PC)**

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

**(5) Interrupt table register (INTB)**

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

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

Stack pointer comes in two types: user stack pointer (USP) and interrupt stack pointer (ISP), each configured with 16 bits. Your desired type of stack pointer (USP or ISP) can be selected by a stack pointer select flag (U flag). This flag is located at the position of bit 7 in the flag register (FLG).

**(7) Static base register (SB)**

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

**(8) Flag register (FLG)**

Flag register (FLG) is configured with 11 bits, each bit is used as a flag. Figure 1.7 shows the flag register (FLG). The following explains the function of each flag:

• Bit 0: Carry flag (C flag)
This flag retains a carry, borrow, or shift-out bit that has occurred in the arithmetic/logic unit.
• Bit 1: Debug flag (D flag)
This flag enables a single-step interrupt.
When this flag is “1”, a single-step interrupt is generated after instruction execution. This flag is cleared to “0” when the interrupt is acknowledged.
• Bit 2: Zero flag (Z flag)
This flag is set to “1” when an arithmetic operation resulted in 0; otherwise, cleared to “0”.
• Bit 3: Sign flag (S flag)
This flag is set to “1” when an arithmetic operation resulted in a negative value; otherwise, cleared to “0”.
• Bit 4: Register bank select flag (B flag)
This flag chooses a register bank. Register bank 0 is selected when this flag is “0” ; register bank 1 is selected when this flag is “1”.
• Bit 5: Overflow flag (O flag)
This flag is set to “1” when an arithmetic operation resulted in overflow; otherwise, cleared to “0”.
• Bit 6: Interrupt enable flag (I flag)
This flag enables a maskable interrupt.
An interrupt is disabled when this flag is “0”, and is enabled when this flag is “1”. This flag is cleared to “0” when the interrupt is acknowledged.
• Bit 7: Stack pointer select flag (U flag)
Interrupt stack pointer (ISP) is selected when this flag is “0” ; user stack pointer (USP) is selected when this flag is “1”.

This flag is cleared to “0” when a hardware interrupt is acknowledged or an INT instruction of software interrupt Nos. 0 to 31 is executed.

• Bits 8 to 11: Reserved area
• Bits 12 to 14: Processor interrupt priority level (IPL)
Processor interrupt priority level (IPL) is configured with three bits, for specification of up to eight processor interrupt priority levels from level 0 to level 7.
If a requested interrupt has priority greater than the processor interrupt priority level (IPL), the interrupt is enabled.
• Bit 15: Reserved area.

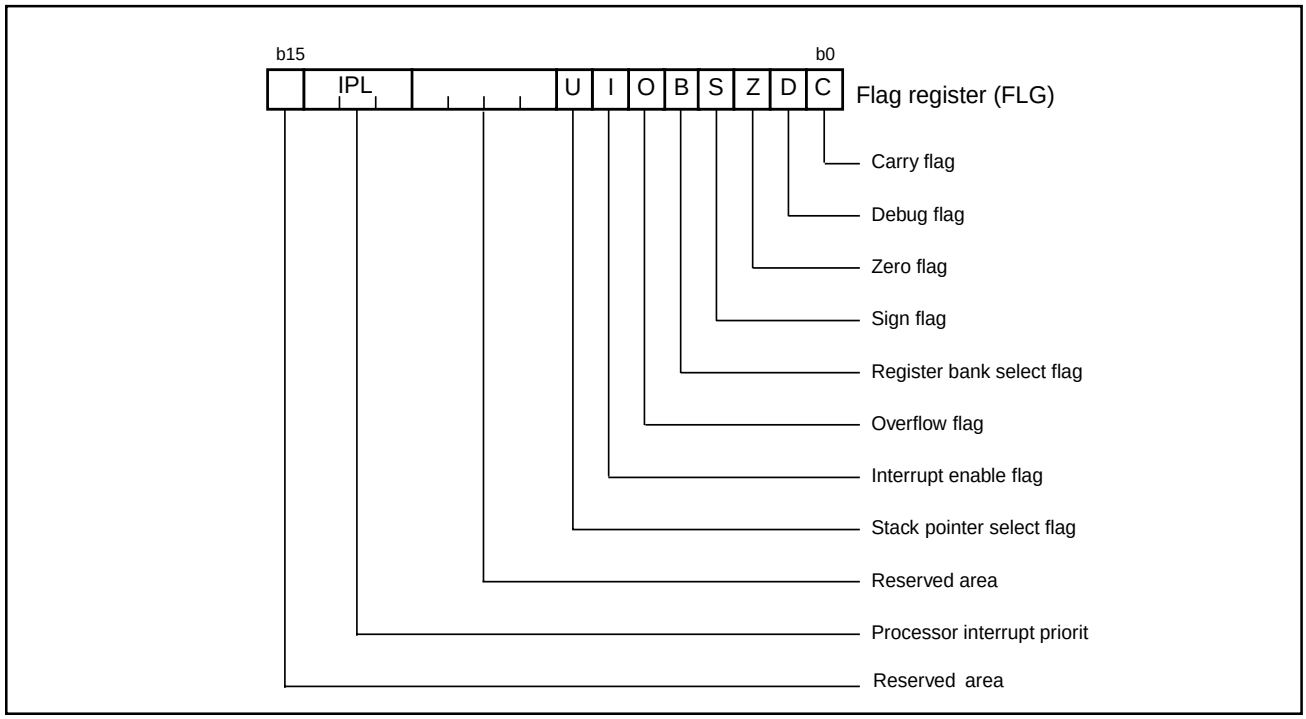



**Fig. 1.7. Flag Register**

## 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.) This section explains on hardware resets.

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

Figure 1.8 shows an example reset circuit. Figure 1.9 shows a reset sequence. Table 1.4 shows the pin status when reset pin level is "L".

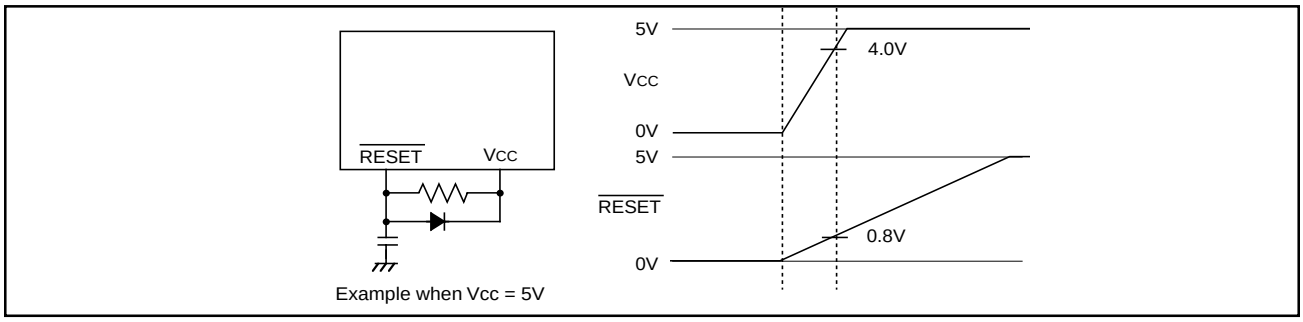



**Fig. 1.8. Example of Reset Circuit**

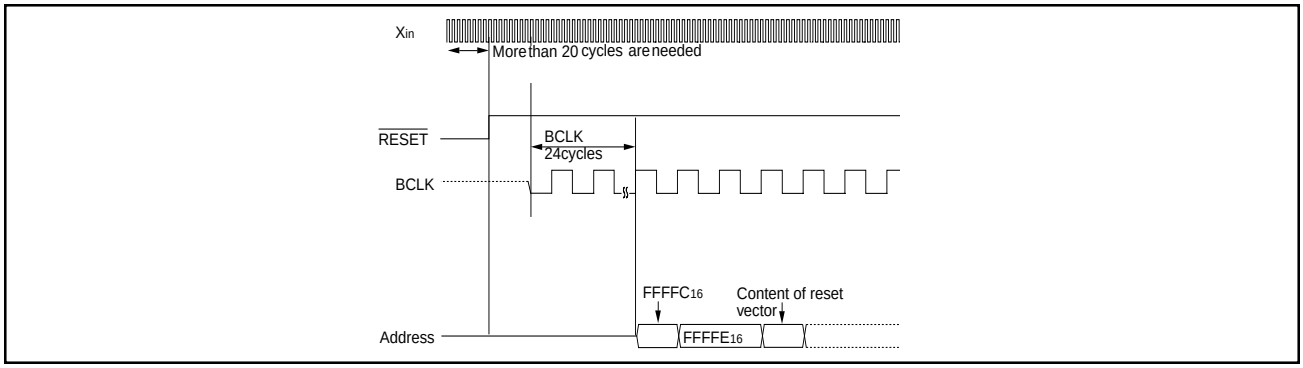



**Fig. 1.9. Reset sequence**

**Table 1.4. Pin status when $\overline{\text{Reset}}$ pin level is "L"**

| Pin name | Status |
|---|---|
| P3, P4 | Input port (with a pull-up resistor) |
| P6 to P10 | Input port (floating) |
| SEG0 to SEG23 | "H" level is output |
| COM0 to COM3 | "H" level is output |

## Special function registers

### Table 1.5. Location and value after reset of peripheral unit control registers (1)

| SFR Address | Register Name | Acronym | Value after Reset | | | | | | | | Page Number |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 | |
| $0000_{16}$ | | | | | | | | | | | |
| $0001_{16}$ | | | | | | | | | | | |
| $0002_{16}$ | | | | | | | | | | | |
| $0003_{16}$ | | | | | | | | | | | |
| $0004_{16}$ | Processor mode register 0 | PM0 | $00_{16}$ | | | | | | | | 1.19 |
| $0005_{16}$ | Processor mode register 1 | PM1 | 0 | | | | | | 0 | 0 | 1.19 |
| $0006_{16}$ | System clock control register 0 | CM0 | $48_{16}$ | | | | | | | | 1.23 |
| $0007_{16}$ | System clock control register 1 | CM1 | $20_{16}$ | | | | | | | | 1.23 |
| $0008_{16}$ | | | | | | | | | | | |
| $0009_{16}$ | Address match interrupt enable register | AIER | | | | | | | 0 | 0 | 1.53 |
| $000A_{16}$ | Protect register | PRCR | | | | | 0 | 0 | 0 | 0 | 1.34 |
| $000B_{16}$ | | | | | | | | | | | |
| $000C_{16}$ | | | | | | | | | | | |
| $000D_{16}$ | | | | | | | | | | | |
| $000E_{16}$ | Watchdog timer start register | WDTS | | | | | | | | | 1.57 |
| $000F_{16}$ | Watchdog timer control register | WDC | 0 | 0 | 0 | | | | | | 1.57 |
| $0010_{16}$ | Address match interrupt register 0 | RMAD0 | $00_{16}$ | | | | | | | | 1.53 |
| $0011_{16}$ | | | | | | | | | | | 1.53 |
| $0012_{16}$ | | | | | | | | | | | 1.53 |
| $0013_{16}$ | | | | | | | | | | | |
| $0014_{16}$ | Address match interrupt register 1 | RMAD1 | $00_{16}$ | | | | | | | | 1.53 |
| $0015_{16}$ | | | | | | | | | | | 1.53 |
| $0016_{16}$ | | | | | | | | | | | 1.53 |
| $0017_{16}$ | | | | | | | | | | | |
| $0018_{16}$ | VDC control register | VDCC | | | | 0 | 0 | | 0 | 0 | 1.29 |
| $0019_{16}$ | | | | | | | | | | | |
| $001A_{16}$ | | | | | | | | | | | |
| $001B_{16}$ | | | | | | | | | | | |
| $001C_{16}$ | | | | | | | | | | | |
| $001D_{16}$ | | | | | | | | | | | |
| $001E_{16}$ | | | | | | | | | | | |
| $001F_{16}$ | | | | | | | | | | | |
| $0020_{16}$ | DMA0 source pointer | SAR0 | ? | | | | | | | | 1.62 |
| $0021_{16}$ | | | | | | | | | | | 1.62 |
| $0022_{16}$ | | | | | | | | | | | 1.62 |
| $0023_{16}$ | | | | | | | | | | | |
| $0024_{16}$ | DMA0 destination pointer | DAR0 | ? | | | | | | | | 1.62 |
| $0025_{16}$ | | | | | | | | | | | 1.62 |
| $0026_{16}$ | | | | | | | | | | | 1.62 |
| $0027_{16}$ | | | | | | | | | | | |
| $0028_{16}$ | DMA0 transfer counter | TRC0 | ? | | | | | | | | 1.62 |
| $0029_{16}$ | | | | | | | | | | | 1.62 |
| $002A_{16}$ | | | | | | | | | | | |
| $002B_{16}$ | | | | | | | | | | | |
| $002C_{16}$ | DMA0 control register | DM0CON | | | 0 | 0 | 0 | 0 | 0 | 0 | 1.61 |
| $002D_{16}$ | | | | | | | | | | | |
| $002E_{16}$ | | | | | | | | | | | |
| $002F_{16}$ | | | | | | | | | | | |
| $0030_{16}$ | DMA1 source pointer | SAR1 | ? | | | | | | | | 1.62 |
| $0031_{16}$ | | | | | | | | | | | 1.62 |
| $0032_{16}$ | | | | | | | | | | | 1.62 |
| $0033_{16}$ | | | | | | | | | | | |
| $0034_{16}$ | DMA1 destination pointer | DAR1 | ? | | | | | | | | 1.62 |
| $0035_{16}$ | | | | | | | | | | | 1.62 |
| $0036_{16}$ | | | | | | | | | | | 1.62 |
| $0037_{16}$ | | | | | | | | | | | |
| $0038_{16}$ | DMA1 transfer counter | TCR1 | ? | | | | | | | | 1.62 |
| $0039_{16}$ | | | | | | | | | | | 1.62 |
| $003A_{16}$ | | | | | | | | | | | |
| $003B_{16}$ | | | | | | | | | | | |
| $003C_{16}$ | DMA1 control register | DM1CON | | | 0 | 0 | 0 | 0 | 0 | 0 | 1.61 |
| $003D_{16}$ | | | | | | | | | | | |
| $003E_{16}$ | | | | | | | | | | | |
| $003F_{16}$ | | | | | | | | | | | |

? = Undefined

**Table 1.6. Location and value after reset of peripheral unit control registers (2)**

| SFR Address | Register Name | Acronym | Value after Reset b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 | Page Number |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | |
| $0044_{16}$ | INT3 interrupt control register<br>SI/O4 interrupt control register | INT3IC<br>S4IC | | | 0 | 0 | 0 | 0 | 0 | 0 | 1.39 |
| $0045_{16}$ | Timer B5 interrupt control register | TB5IC | | | | | 0 | 0 | 0 | 0 | 1.39 |
| $0046_{16}$ | Timer B4 interrupt control register | TB 4IC | | | | | 0 | 0 | 0 | 0 | 1.39 |
| $0047_{16}$ | Timer B3 interrupt control register | TB3IC | | | | | 0 | 0 | 0 | 0 | 1.39 |
| $0048_{16}$ | INT7 interrupt control register | INT7IC | | | 0 | 0 | 0 | 0 | 0 | 0 | 1.39 |
| $0049_{16}$ | INT6 interrupt control register | INT6IC | | | 0 | 0 | 0 | 0 | 0 | 0 | 1.39 |
| $004A_{16}$ | Bus collision detection interrupt control register | BCNIC | | | | | 0 | 0 | 0 | 0 | 1.39 |
| $004B_{16}$ | DMA0 interrupt control register | DM0IC | | | | | 0 | 0 | 0 | 0 | 1.39 |
| $004C_{16}$ | DMA1 interrupt control register | DM1IC | | | | | 0 | 0 | 0 | 0 | 1.39 |
| $004D_{16}$ | Key input interrupt control register | KUPIC | | | | | 0 | 0 | 0 | 0 | 1.39 |
| $004E_{16}$ | A-D conversion interrupt control register | ADIC | | | | | 0 | 0 | 0 | 0 | 1.39 |
| $004F_{16}$ | UART2 transmit interrupt control register | S2TIC | | | | | 0 | 0 | 0 | 0 | 1.39 |
| $0050_{16}$ | UART2 receive interrupt control register | S2RIC | | | | | 0 | 0 | 0 | 0 | 1.39 |
| $0051_{16}$ | UART0 transmit interrupt control register | S0TIC | | | | | 0 | 0 | 0 | 0 | 1.39 |
| $0052_{16}$ | UART0 receive interrupt control register | S0RIC | | | | | 0 | 0 | 0 | 0 | 1.39 |
| $0053_{16}$ | UART1 transmit interrupt control register | S1TIC | | | | | 0 | 0 | 0 | 0 | 1.39 |
| $0054_{16}$ | UART1 receive interrupt control register | S1RIC | | | | | 0 | 0 | 0 | 0 | 1.39 |
| $0055_{16}$ | Timer A0 interrupt control register | TA0IC | | | | | 0 | 0 | 0 | 0 | 1.39 |
| $0056_{16}$ | Timer A1 interrupt control register | TA1IC | | | | | 0 | 0 | 0 | 0 | 1.39 |
| $0057_{16}$ | Timer A2 interrupt control register | TA2IC | | | | | 0 | 0 | 0 | 0 | 1.39 |
| $0058_{16}$ | Timer A3 interrupt control register<br>INT4 interrupt control register | TA3IC<br>INT4IC | | | 0 | 0 | 0 | 0 | 0 | 0 | 1.39 |
| $0059_{16}$ | Timer A4 interrupt control register<br>INT5 interrupt control register | TA4IC<br>INT5IC | | | 0 | 0 | 0 | 0 | 0 | 0 | 1.39 |
| $005A_{16}$ | Timer B0 interrupt control register | TB0IC | | | | | 0 | 0 | 0 | 0 | 1.39 |
| $005B_{16}$ | Timer B1 interrupt control register | TB1IC | | | | | 0 | 0 | 0 | 0 | 1.39 |
| $005C_{16}$ | Timer B2 interrupt control register | TB2IC | | | | | 0 | 0 | 0 | 0 | 1.39 |
| $005D_{16}$ | INT0 interrupt control register | INT0IC | | | 0 | 0 | 0 | 0 | 0 | 0 | 1.39 |
| $005E_{16}$ | INT1 interrupt control register | INT1IC | | | 0 | 0 | 0 | 0 | 0 | 0 | 1.39 |
| $005F_{16}$ | INT2 interrupt control register<br>SI/O3 interrupt control register | INT2IC<br>S3IC | | | 0 | 0 | 0 | 0 | 0 | 0 | 1.39 |
| | | | | | | | | | | | |
| | | | | | | | | | | | |
| $0100_{16}$ | LCD RAM0 | LRAM0 | ? | ? | ? | ? | ? | ? | ? | ? | 1.146 |
| $0101_{16}$ | LCD RAM1 | LRAM1 | ? | ? | ? | ? | ? | ? | ? | ? | 1.146 |
| $0102_{16}$ | LCD RAM2 | LRAM2 | ? | ? | ? | ? | ? | ? | ? | ? | 1.146 |
| $0103_{16}$ | LCD RAM3 | LRAM3 | ? | ? | ? | ? | ? | ? | ? | ? | 1.146 |
| $0104_{16}$ | LCD RAM4 | LRAM4 | ? | ? | ? | ? | ? | ? | ? | ? | 1.146 |
| $0105_{16}$ | LCD RAM5 | LRAM5 | ? | ? | ? | ? | ? | ? | ? | ? | 1.146 |
| $0106_{16}$ | LCD RAM6 | LRAM6 | ? | ? | ? | ? | ? | ? | ? | ? | 1.146 |
| $0107_{16}$ | LCD RAM7 | LRAM7 | ? | ? | ? | ? | ? | ? | ? | ? | 1.146 |
| $0108_{16}$ | LCD RAM8 | LRAM8 | ? | ? | ? | ? | ? | ? | ? | ? | 1.146 |
| $0109_{16}$ | LCD RAM9 | LRAM9 | ? | ? | ? | ? | ? | ? | ? | ? | 1.146 |
| $010A_{16}$ | LCD RAM10 | LRAM10 | ? | ? | ? | ? | ? | ? | ? | ? | 1.146 |
| $010B_{16}$ | LCD RAM11 | LRAM11 | ? | ? | ? | ? | ? | ? | ? | ? | 1.146 |
| $010C_{16}$ | LCD RAM12 | LRAM12 | ? | ? | ? | ? | ? | ? | ? | ? | 1.146 |
| $010D_{16}$ | LCD RAM13 | LRAM13 | ? | ? | ? | ? | ? | ? | ? | ? | 1.146 |
| $010E_{16}$ | LCD RAM14 | LRAM14 | ? | ? | ? | ? | ? | ? | ? | ? | 1.146 |
| $010F_{16}$ | LCD RAM15 | LRAM15 | ? | ? | ? | ? | ? | ? | ? | ? | 1.146 |
| $0110_{16}$ | LCD RAM16 | LRAM16 | ? | ? | ? | ? | ? | ? | ? | ? | 1.146 |
| $0111_{16}$ | LCD RAM17 | LRAM17 | ? | ? | ? | ? | ? | ? | ? | ? | 1.146 |
| $0112_{16}$ | LCD RAM18 | LRAM18 | ? | ? | ? | ? | ? | ? | ? | ? | 1.146 |
| $0113_{16}$ | LCD RAM19 | LRAM19 | ? | ? | ? | ? | ? | ? | ? | ? | 1.146 |
| | | | | | | | | | | | |
| | | | | | | | | | | | |
| $0120_{16}$ | LCD mode register | LCDM | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 1.143 |
| $0121_{16}$ | | | | | | | | | | | |
| $0122_{16}$ | Segment output enable register | SEG | $00_{16}$ | | | | | | | | 1.143 |
| $0123_{16}$ | | | | | | | | | | | |
| $0124_{16}$ | LCD frame frequency counter | LCDTIM | | | | | | | | | 1.143 |
| $0125_{16}$ | | | | | | | | | | | |
| $0126_{16}$ | Key input mode register | KUPM | | | | | | | 0 | 0 | 1.52 |
| $0127_{16}$ | | | | | | | | | | | |
| $0128_{16}$ | | | | | | | | | | | |
| $0129_{16}$ | | | | | | | | | | | |
| $0130_{16}$ | LCD expansion register | LEXP | $00_{16}$ | | | | | | | | 1.144 |
| $0131_{16}$ | | | | | | | | | | | |
| $0132_{16}$ | LCD clock divide counter | LCDC | | | | | | | | | 1.144 |
| $0133_{16}$ | | | | | | | | | | | |

? = Undefined

**Table 1.7. Location and value after reset of peripheral unit control registers (3)**

| SFR Address | Register Name | Acronym | Value after Reset b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 | Page Number |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $0340_{16}$ | Timer B3, 4, 5 count start flag | TBSR | 0 | 0 | 0 | | | | | | 1.85 |
| $0341_{16}$ | | | | | | | | | | | |
| $0342_{16}$ | Timer A1-1 register | TA11 | | | | | | | | | 1.94 |
| $0343_{16}$ | | | | | | | | | | | |
| $0344_{16}$ | Timer A2-1 register | TA21 | | | | | | | | | 1.94 |
| $0345_{16}$ | | | | | | | | | | | |
| $0346_{16}$ | Timer A4-1 register | TA41 | | | | | | | | | 1.94 |
| $0347_{16}$ | | | | | | | | | | | |
| $0348_{16}$ | Three-phase PWM control register 0 | INVC0 | $00_{16}$ | | | | | | | | 1.92 |
| $0349_{16}$ | Three-phase PWM control register 1 | INVC1 | $00_{16}$ | | | | | | | | 1.92 |
| $034A_{16}$ | Three-phase output buffer register 0 | IDB0 | $3F_{16}$ | | | | | | | | 1.93 |
| $034B_{16}$ | Three-phase output buffer register 1 | IDB1 | $3F_{16}$ | | | | | | | | 1.93 |
| $034C_{16}$ | Dead time timer | DTT | | | | | | | | | 1.93 |
| $034D_{16}$ | Timer B2 interrupt occurrence frequency set counter | ICTB2 | | | | | | | | | 1.93 |
| $034E_{16}$ | | | | | | | | | | | |
| $034F_{16}$ | | | | | | | | | | | |
| $0350_{16}$ | Timer B3 register | TB3 | | | | | | | | | 1.85 |
| $0351_{16}$ | | | | | | | | | | | |
| $0352_{16}$ | Timer B4 register | TB4 | | | | | | | | | 1.85 |
| $0353_{16}$ | | | | | | | | | | | |
| $0354_{16}$ | Timer B5 register | TB5 | | | | | | | | | 1.85 |
| $0355_{16}$ | | | | | | | | | | | |
| | | | | | | | | | | | |
| $035B_{16}$ | Timer B3 mode register | TB3MR | 0 | 0 | | | 0 | 0 | 0 | 0 | 1.87 1.88, 1.90 |
| $035C_{16}$ | Timer B4 mode register | TB4MR | 0 | 0 | | | 0 | 0 | 0 | 0 | 1.87 1.88, 1.90 |
| $035D_{16}$ | Timer B5 mode register | TB5MR | 0 | 0 | | | 0 | 0 | 0 | 0 | 1.87 1.88, 1.90 |
| $035E_{16}$ | Interrupt cause select register 0 | IFSR0 | $00_{16}$ | | | | | | | | 1.49 |
| $035F_{16}$ | Interrupt cause select register 1 | IFSR1 | $00_{16}$ | | | | | | | | 1.49 |
| $0360_{16}$ | SI/O3 transmit/receive register | S3TRR | | | | | | | | | 1.138 |
| $0361_{16}$ | | | | | | | | | | | |
| $0362_{16}$ | SI/O3 control register | S3C | $40_{16}$ | | | | | | | | 1.138 |
| $0363_{16}$ | SI/O3 bit rate generator | S3BRG | | | | | | | | | 1.138 |
| $0364_{16}$ | SI/O4 transmit/receive register | S4TRR | | | | | | | | | 1.138 |
| $0365_{16}$ | | | | | | | | | | | |
| $0366_{16}$ | SI/O4 control register | S4C | $40_{16}$ | | | | | | | | 1.138 |
| $0367_{16}$ | SI/O4 bit rate generator | S4BRG | | | | | | | | | 1.138 |
| | | | | | | | | | | | |
| $036C_{16}$ | Clock divided control register | CDCC | 0 | | | | | | | | 1.24 |
| $036D_{16}$ | | | | | | | | | | | |
| $036E_{16}$ | Clock divided counter | CDC | | | | | | | | | 1.24 |
| $0375_{16}$ | UART2 special mode register 3 | U2SMR3 | $00_{16}$ | | | | | | | | 1.112 1.130 |
| $0376_{16}$ | UART2 special mode register 2 | U2SMR2 | $00_{16}$ | | | | | | | | 1.112, 1.134 |
| $0377_{16}$ | UART2 special mode register | U2SMR | $00_{16}$ | | | | | | | | 1.111, 1.130 |
| $0378_{16}$ | UART2 transmit/receive mode register | U2MR | $00_{16}$ | | | | | | | | 1.108, 1.114, 1.120 |
| $0379_{16}$ | UART2 bit rate generator | U2BRG | | | | | | | | | 1.107 |
| $037A_{16}$ | | | | | | | | | | | |
| $037B_{16}$ | UART2 transmit buffer register | U2TB | | | | | | | | | 1.107 |
| $037C_{16}$ | UART2 transmit/receive control register 0 | U2C0 | $08_{16}$ | | | | | | | | |
| $037D_{16}$ | UART2 transmit/receive control register 1 | U2C1 | $02_{16}$ | | | | | | | | 1.110 |
| $037E_{16}$ | | | | | | | | | | | |
| $037F_{16}$ | UART2 receive buffer register | U2RB | | | | | | | | | 1.102 |

? = Undefined

| SFR Address | Register Name | Acronym | Value after Reset b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 | Page Number |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $0380_{16}$ | Count start flag | TABSR | $00_{16}$ | | | | | | | | 1.71, 1.85, 1.94 |
| $0381_{16}$ | Clock prescaler reset flag | CPSRF | 0 | | | | | | | | 1.72, 1.85 |
| $0382_{16}$ | One-shot start flag | ONSF | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 1.72 |
| $0383_{16}$ | Trigger select register | TRGSR | $00_{16}$ | | | | | | | | 1.72, 1.94 |
| $0384_{16}$ | Up-down flag | UDF | $00_{16}$ | | | | | | | | 1.71 |
| $0385_{16}$ | | | | | | | | | | | |
| $0386_{16}$ | Timer A0 | TA0 | | | | | | | | | |
| $0387_{16}$ | | | | | | | | | | | 1.71 |
| $0388_{16}$ | Timer A1 | TA1 | | | | | | | | | |
| $0389_{16}$ | | | | | | | | | | | 1.71, 1.90 |
| $038A_{16}$ | Timer A2 | TA2 | | | | | | | | | |
| $038B_{16}$ | | | | | | | | | | | 1.71, 1.90 |
| $038C_{16}$ | Timer A3 | TA3 | | | | | | | | | |
| $038D_{16}$ | | | | | | | | | | | 1.71 |
| $038E_{16}$ | Timer A4 | TA4 | | | | | | | | | |
| $038F_{16}$ | | | | | | | | | | | 1.71, 1.90 |
| $0390_{16}$ | Timer B0 | TB0 | | | | | | | | | |
| $0391_{16}$ | | | | | | | | | | | 1.83 |
| $0392_{16}$ | Timer B1 | TB1 | | | | | | | | | |
| $0393_{16}$ | | | | | | | | | | | 1.83 |
| $0394_{16}$ | Timer B2 | TB2 | | | | | | | | | |
| $0395_{16}$ | | | | | | | | | | | 1.83, 1.90 |
| $0396_{16}$ | Timer A0 mode register | TA0MR | $00_{16}$ | | | | | | | | 1.70, 1.73, 1.74, 1.77, 1.78 |
| $0397_{16}$ | Timer A1 mode register | TA1MR | $00_{16}$ | | | | | | | | 1.70, 1.73, 1.74, 1.77, 1.78, 1.95 |
| $0398_{16}$ | Timer A2 mode register | TA2MR | $00_{16}$ | | | | | | | | 1.70, 1.73, 1.74, 1.77, 1.78, 1.95 |
| $0399_{16}$ | Timer A3 mode register | TA3MR | $00_{16}$ | | | | | | | | 1.70, 1.72, 1.73, 1.77, 1.78 |
| $039A_{16}$ | Timer A4 mode register | TA4MR | $00_{16}$ | | | | | | | | 1.70, 1.73, 1.74, 1.77, 1.78, 1.95 |
| $039B_{16}$ | Timer B0 mode register | TB0MR | 0 | 0 | | | 0 | 0 | 0 | 0 | 1.84, 1.87, 1.90 |
| $039C_{16}$ | Timer B1 mode register | TB1MR | 0 | 0 | | | 0 | 0 | 0 | 0 | 1.84, 1.87, 1.90 |
| $039D_{16}$ | Timer B2 mode register | TB2MR | 0 | 0 | | | 0 | 0 | 0 | 0 | 1.84, 1.87, 1.90, 1.95 |
| $039E_{16}$ | | | | | | | | | | | |
| $039F_{16}$ | | | | | | | | | | | |
| $03A0^{16}$ | UART0 transmit/receive mode register | U0MR | $00_{16}$ | | | | | | | | 1.108, 1.114, 1.120 |
| $03A1_{16}$ | UART0 bit rate generator | U0BRG | | | | | | | | | 1.107 |
| $03A2_{16}$ | UART0 transmit buffer register | U0TB | | | | | | | | | 1.107 |
| $03A3_{16}$ | | | | | | | | | | | |
| $03A4_{16}$ | UART0 transmit/receive control register 0 | U0C0 | $08_{16}$ | | | | | | | | 1.109 |
| $03A5_{16}$ | UART0 transmit/receive control register 1 | U0C1 | $02_{16}$ | | | | | | | | 1.110 |
| $03A6_{16}$ | UART0 receive buffer register | U0RB | | | | | | | | | 1.107 |
| $03A7_{16}$ | | | | | | | | | | | |
| $03A8_{16}$ | UART1 transmit/receive mode register | U1MR | $00_{16}$ | | | | | | | | 1.108, 1.114, 1.120 |
| $03A9_{16}$ | UART1 bit rate generator | U1BRG | | | | | | | | | 1.107 |
| $03AA_{16}$ | | | | | | | | | | | |
| $03AB_{16}$ | UART1 transmit buffer register | U1TB | | | | | | | | | 1.107 |
| $03AC_{16}$ | UART1 transmit/receive control register 0 | U1C0 | $08_{16}$ | | | | | | | | 1.109 |
| $03AD_{16}$ | UART1 transmit/receive control register 1 | U1C1 | $02_{16}$ | | | | | | | | 1.110 |
| $03AE_{16}$ | | | | | | | | | | | |
| $03AF_{16}$ | UART1 receive buffer register | U1RB | | | | | | | | | 1.107 |
| $03B0_{16}$ | UART transmit/receive control register 2 | UCON | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1.111 |
| $03B1_{16}$ | | | | | | | | | | | |
| $03B2_{16}$ | | | | | | | | | | | |
| $03B3_{16}$ | | | | | | | | | | | |
| $03B4_{16}$ | Flash memory control register (Note) | FMCR | | | | | 0 | 0 | 0 | 1 | 1.176 |
| $03B5_{16}$ | | | | | | | | | | | |
| $03B6_{16}$ | | | | | | | | | | | |
| $03B7_{16}$ | | | | | | | | | | | |
| $03B8_{16}$ | DMA0 request cause select register | DM0SL | $00_{16}$ | | | | | | | | 1.60 |
| $03B9_{16}$ | | | | | | | | | | | |
| $03BA_{16}$ | DMA1 | DM1SL | $00_{16}$ | | | | | | | | 1.61 |
| $03BB_{16}$ | | | | | | | | | | | |
| $03BC_{16}$ | CRC data register | CRCD | | | | | | | | | 1.164 |
| $03BD_{16}$ | | | | | | | | | | | |
| $03BE_{16}$ | CRC input register | CRCIN | | | | | | | | | 1.164 |

Note: This register only exists in flash memory version

? = Undefined

**Table 1.8. Location and value after reset of peripheral unit control registers (1)**

| SFR Address | Register Name | Acronym | Value after Reset b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 | Page Number |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $03C0_{16}$ | | | | | | | | | | | 1.154 |
| $03C1_{16}$ | A-D register 0 | AD0 | | | | | | | | | |
| $03C2_{16}$ | | | | | | | | | | | 1.154 |
| $03C3_{16}$ | A-D register 1 | AD1 | | | | | | | | | |
| $03C4_{16}$ | | | | | | | | | | | 1.154 |
| $03C5_{16}$ | A-D register 2 | AD2 | | | | | | | | | |
| $03C6_{16}$ | | | | | | | | | | | 1.154 |
| $03C7_{16}$ | A-D register 3 | AD3 | | | | | | | | | |
| $03C8_{16}$ | | | | | | | | | | | 1.154 |
| $03C9_{16}$ | A-D register 4 | AD4 | | | | | | | | | |
| $03CA_{16}$ | | | | | | | | | | | 1.154 |
| $03CB_{16}$ | A-D register 5 | AD5 | | | | | | | | | |
| $03CC_{16}$ | | | | | | | | | | | 1.154 |
| $03CD_{16}$ | A-D register 6 | AD6 | | | | | | | | | |
| $03CE_{16}$ | | | | | | | | | | | 1.154 |
| $03CF_{16}$ | A-D register 7 | AD7 | | | | | | | | | |
| $03D0_{16}$ | | | | | | | | | | | |
| $03D1_{16}$ | | | | | | | | | | | |
| $03D2_{16}$ | | | | | | | | | | | |
| $03D3_{16}$ | | | | | | | | | | | |
| $03D4_{16}$ | A-D control register 2 | ADCON2 | 0 | 0 | 0 | 0 | | | | 0 | 1.154 |
| $03D5_{16}$ | | | | | | | | | | | |
| $03D6_{16}$ | A-D control register 0 | ADCON0 | 0 | 0 | 0 | 0 | 0 | | | | 1.153, 1.155, 1.156, 1.157, 1.158, 1.159 |
| $03D7_{16}$ | A-D control register 1 | ADCON1 | $00_{16}$ | | | | | | | | 1.153, 1.155, 1.156, 1.157, 1.158, 1.159 |
| $03D8_{16}$ | D-A register 0 | DA0 | | | | | | | | | 1.163 |
| $03D9_{16}$ | | | | | | | | | | | |
| $03DA_{16}$ | D-A register 1 | DA1 | | | | | | | | | 1.163 |
| $03DB_{16}$ | | | | | | | | | | | |
| $03DC_{16}$ | D-A control register | DACON | $00_{16}$ | | | | | | | | 1.163 |
| $03DD_{16}$ | | | | | | | | | | | |
| $03DE_{16}$ | | | | | | | | | | | |
| $03DF_{16}$ | | | | | | | | | | | |
| $03E0_{16}$ | | | | | | | | | | | |
| $03E1_{16}$ | | | | | | | | | | | |
| $03E2_{16}$ | | | | | | | | | | | |
| $03E3_{16}$ | | | | | | | | | | | |
| $03E4_{16}$ | | | | | | | | | | | |
| $03E5_{16}$ | Port P3 | P3 | | | | | | | | | 1.170 |
| $03E6_{16}$ | | | | | | | | | | | |
| $03E7_{16}$ | Port P3 direction register | PD3 | $00_{16}$ | | | | | | | | 1.170 |
| $03E8_{16}$ | Port P4 | P4 | | | | | | | | | 1.170 |
| $03E9_{16}$ | | | | | | | | | | | |
| $03EA_{16}$ | Port P4 direction register | PD4 | $00_{16}$ | | | | | | | | 1.170 |
| $03EB_{16}$ | | | | | | | | | | | |
| $03EC_{16}$ | Port P6 | P6 | | | | | | | | | 1.170 |
| $03ED_{16}$ | Port P7 | P7 | | | | | | | | | 1.170 |
| $03EE_{16}$ | Port P6 direction register | PD6 | $00_{16}$ | | | | | | | | 1.170 |
| $03EF_{16}$ | Port P7 direction register | PD7 | $00_{16}$ | | | | | | | | 1.170 |
| $03F0_{16}$ | Port P8 | P8 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1.170 |
| $03F1_{16}$ | Port P9 | P9 | | | | | | | | | 1.170 |
| $03F2_{16}$ | Port P8 direction register | PD8 | | 0 | | 0 | 0 | 0 | 0 | 0 | 1.170 |
| $03F3_{16}$ | Port P9 direction register | PD9 | $00_{16}$ | | | | | | | | 1.170 |
| $03F4_{16}$ | Port P10 | P10 | | | | | | | | | 1.170 |
| $03F5_{16}$ | | | | | | | | | | | |
| $03F6_{16}$ | Port P10 direction register | PD10 | $00_{16}$ | | | | | | | | 1.170 |
| $03F7_{16}$ | | | | | | | | | | | |
| $03F8_{16}$ | | | | | | | | | | | |
| $03F9_{16}$ | | | | | | | | | | | |
| $03FA_{16}$ | | | | | | | | | | | |
| $03FB_{16}$ | | | | | | | | | | | |
| $03FC_{16}$ | Pull-up control register 0 | PUR0 | $00_{16}$ | | | | | | | | 1.171 |
| $03FD_{16}$ | Pull-up control register 1 | PUR1 | $00_{16}$ | | | | | | | | 1.171 |
| $03FE_{16}$ | Pull-up control register 2 | PUR2 | $00_{16}$ | | | | | | | | 1.171 |
| $03FF_{16}$ | Real-time port control register | RTP | | | | | 0 | 0 | 0 | 0 | 1.83 |

? = Undefined

## 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.

Figure 1.10 shows processor mode register 0 and 1.

Processor mode register 0 (Note)

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

Symbol: PM0 — Address: $0004_{16}$ — When reset: $00_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Nothing is assigned. Write "0" when writing to this these bits. If read, the value is indeterminate. | | | — | — |
| PM03 | Software reset bit | The device is reset when this bit is set to “1”. The value of this bit is “0” when read. | O | O |
| Nothing is assigned. Write "0" when writing to this these bits. If read, the value is indeterminate. | | | — | — |
| Reserved bit | | Must always be set to "0" | O | O |

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

Processor mode register 1 (Note)

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

Symbol: PM1 — Address: $0005_{16}$ — When reset: $0XXXXX00_2$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Reserved bit | | Must always be set to “0” | O | O |
| Nothing is assigned. Write "0" when writing to this these bits. If read, the value is indeterminate. | | | — | — |
| PM17 | Wait bit | 0 : No wait state<br>1 : Wait state inserted | O | O |

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

**Fig. 1.10. Processor mode register 0 and 1**

## Clock generating Circuit

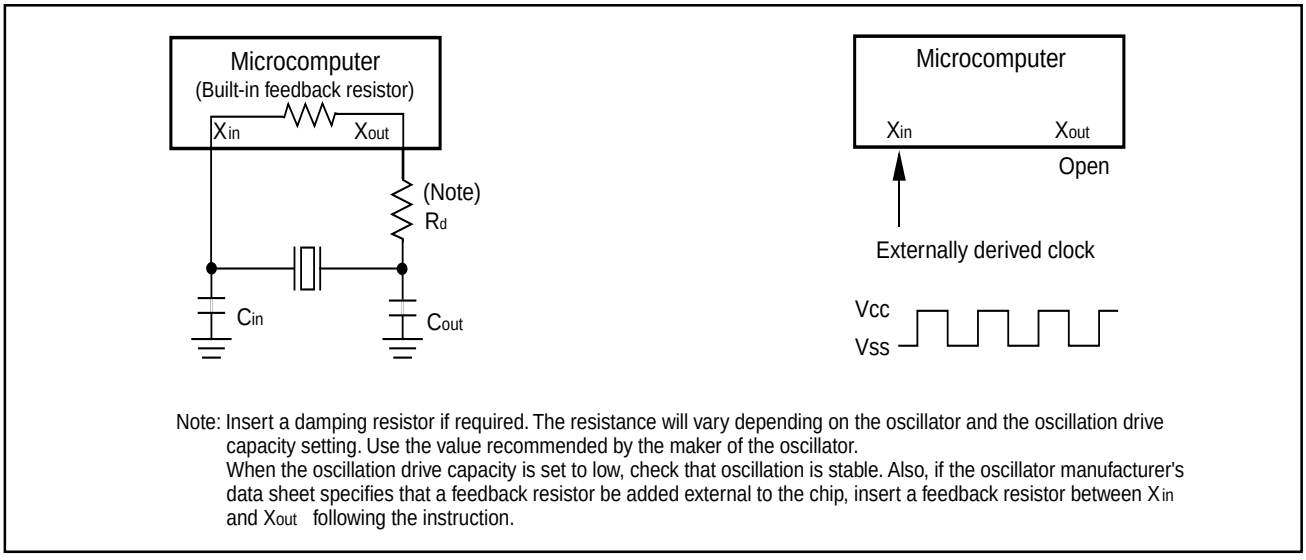
The clock generating circuit contains two oscillator circuits that supply the operating clock sources to the CPU and internal peripheral units. Table 1.10 shows some examples of the main clock and subclock generating circuits.

**Table 1.10. Main clock and sub-clock generating circuits**

| | Main clock generating circuit | Sub-clock generating circuit |
|---|---|---|
| Use of clock | Operating clock source for CPU<br>Operating clock source for Internal peripheral | Operating clock source<br>Count clock source for Timers A/B<br>Operating clock source for LCD |
| Usable oscillator | Ceramic or crystal oscillator | Crystal oscillator |
| Pins to connect oscillator | Xin, Xout | Xcin, Xcout |
| Oscillation stop/restart function | Available | Available |
| Oscillator status immediately after Reset | Oscillating | Stopped |
| Other | Externally derived clock can be input (Note) | |

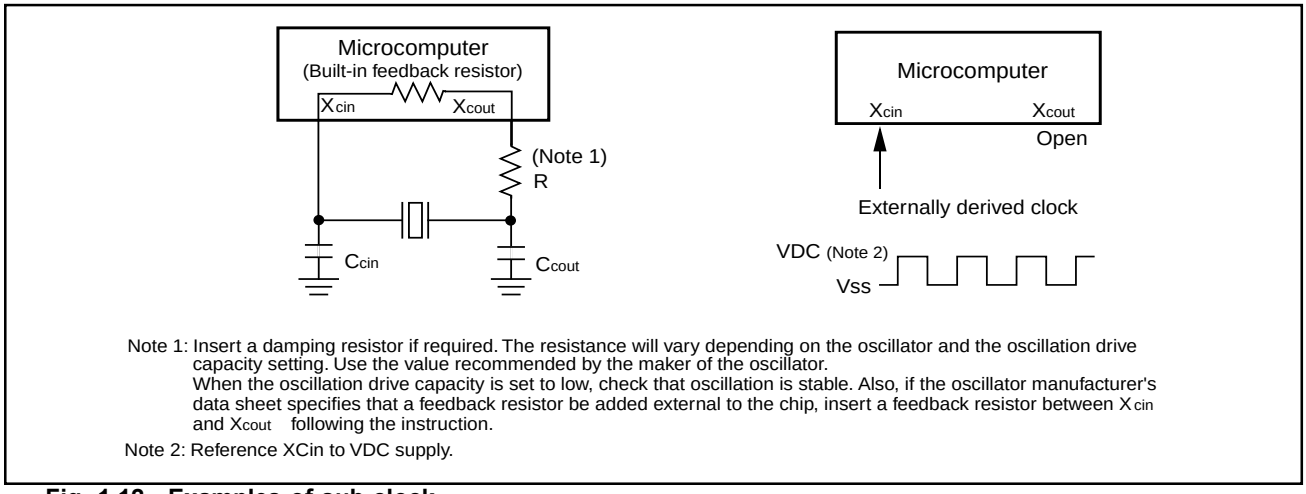
Note: Max. voltage is the same as VDC

Figure 1.11 shows some examples of the main clock circuit, one using an oscillator connected to the circuit, and the other one using an externally derived clock for input. Figure 1.12 shows some examples of sub-clock circuits, one using an oscillator connected to the circuit, and the other one using an externally derived clock for input. Circuit constants in Figures 1.11 and 1.12 vary with each oscillator used. Use the values recommended by the manufacturer of your oscillator.




Note: Insert a damping resistor if required. The resistance will vary depending on the oscillator and the oscillation drive capacity setting. Use the value recommended by the maker of the oscillator.
When the oscillation drive capacity is set to low, check that oscillation is stable. Also, if the oscillator manufacturer's data sheet specifies that a feedback resistor be added external to the chip, insert a feedback resistor between Xin and Xout following the instruction.

**Fig. 1.11. Examples of main clock**




Note 1: Insert a damping resistor if required. The resistance will vary depending on the oscillator and the oscillation drive capacity setting. Use the value recommended by the maker of the oscillator.
When the oscillation drive capacity is set to low, check that oscillation is stable. Also, if the oscillator manufacturer's data sheet specifies that a feedback resistor be added external to the chip, insert a feedback resistor between Xcin and Xcout following the instruction.

Note 2: Reference XCin to VDC supply.

**Fig. 1.12. Examples of sub-clock**

Figure 1.13 shows a block diagram of the clock generating circuit.

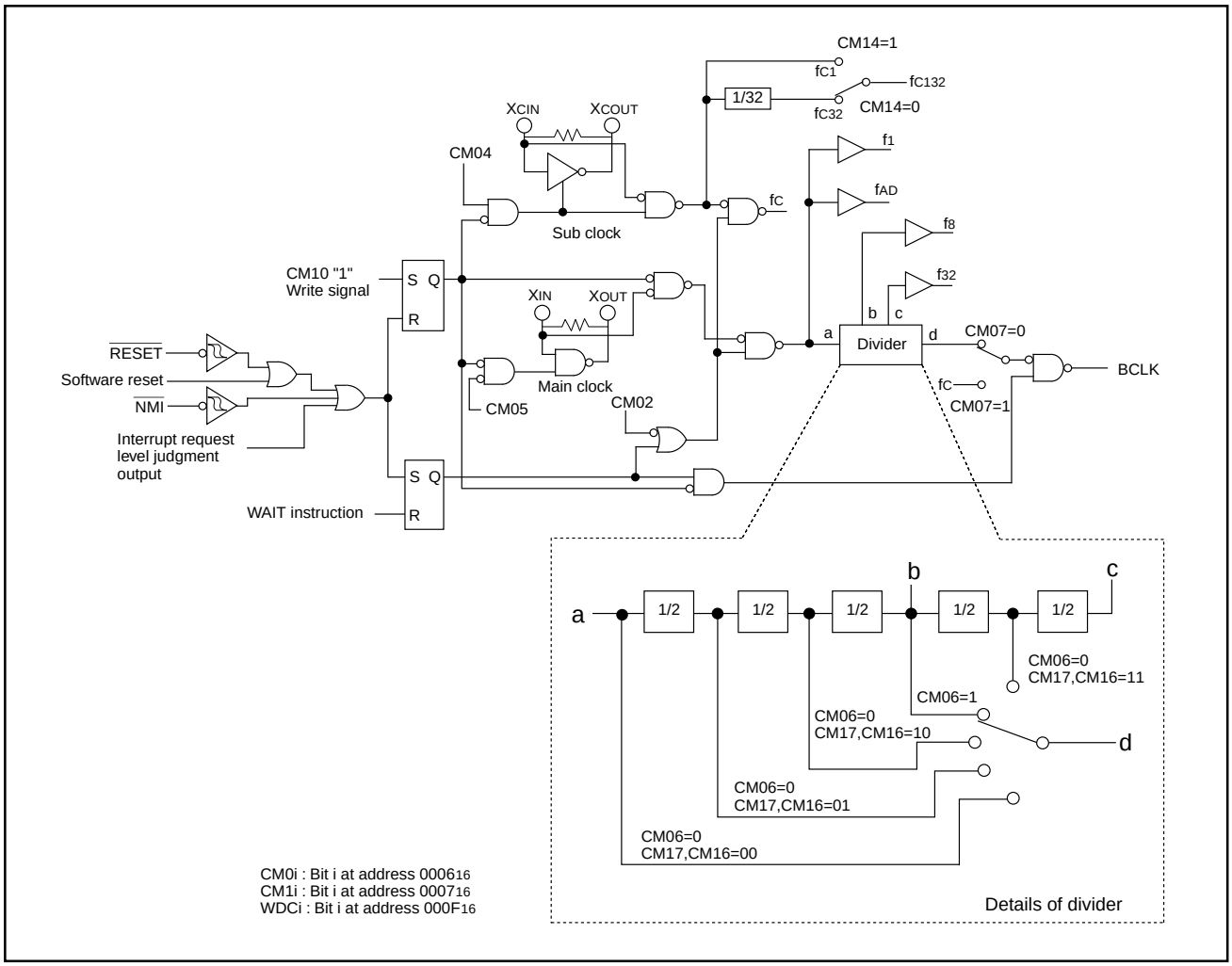



**Fig. 1.13. Clock generating circuit**

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

**(1) Main clock**

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

**(2) Sub-clock**

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

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

**(3) BCLK**

The BCLK is the clock that drives the CPU, and is fc or the clock is derived by dividing the main clock by 1, 2, 4, 8, or 16. The BCLK is derived by dividing the main clock by 8 after a reset.

The main clock division select bit 0(bit 6 at address $0006_{16}$) changes to "1" when shifting from high-speed/ medium-speed to stop mode and at reset. When shifting from low-speed/low power dissipation mode to stop mode, the value before stop mode is retained.

**(4) Peripheral function clock ($f_1$, $f_8$, $f_{32}$, $f_{AD}$)**

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

**(5) $f_{C132}$**

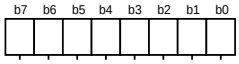
This clock is derived by dividing the sub-clock by 1 or 32. The clock is selected by $f_{C132}$ clock select bit (bit4 at address $0007_{16}$). It is used for the Timer A and Timer B counts, intermittent pull up operation of key input.

**(6) $f_C$**

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

Figure 1.14 shows the system clock control registers 0 and 1.

System clock control register 0 (Note 1)

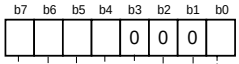



Symbol CM0 | Address 000616 | When reset 4816

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CM00 | Clock output function select bits | b1 b0<br>0 0 : I/O port P75<br>0 1 : fC1 output<br>1 0 : f1 output<br>1 1 : Clock divide counter output | O | O |
| CM01 | | | O | O |
| CM02 | WAIT peripheral function clock stop bit | 0 : Do not stop peripheral function clock in wait mode<br>1 : Stop peripheral function clock in wait mode (Note 7) | O | O |
| CM03 | Xcin-Xout drive capacity select bit (Note 2) | 0 : LOW<br>1 : HIGH | O | O |
| CM04 | Port Xc Select Bit | 0 : I/O port<br>1 : Xcin - Xcout generation | O | O |
| CM05 | Main clock (Xin -Xout ) stop bit (Note 3, 4) | 0 : Main clock on<br>1 : Main clock off | O | O |
| CM06 | Main clock division select bit 0 (Note 6) | 0 : CM16 and CM17 valid<br>1 : Division by 8 mode | O | O |
| CM07 | System clock select bit (Note 5) | 0 : Xin, Xout<br>1 : Xcin, Xcout | O | O |

Note 1: Set bit 0 of the protect register (address 000A16) to "1" before writing to this register.
Note 2: Changes to "1" when shifting to stop mode and at a reset.
Note 3: When inputting external clock, only clock oscillation buffer is stopped and clock input is acceptable.
Note 4: If this bit is set to "1", Xout turns "H". The built-in feedback resistor remains being connected, so X IN turns pulled up to Xout ("H") via the feedback resistor.
Note 5: Set subclock ($X_{cin}$ - $X_{cout}$) enable bit (CM04) to "1" and allow the subclock to stabilize before setting CM07 from from "0" to "1". Do not write to both bits at the same time. Likewise, set the main clock stop bit (CM05) to "0" and allow the subclock to stabilize before settng CM07 bit from "1" to "0".
Note 6: This bit changes to "1" when shifting from high-speed/medium-speed mode to stop mode and at a reset. When shifting from low-speed/low power dissipation mode to stop mode, the value before stop mode is retained.
Note 7: fC, fC132, fC1, fC32 is not included.

System clock control register 1 (Note 1)




Symbol CM1 | Address 000716 | When reset 2016

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CM10 | All clock stop control bit (Note 4) | 0 : Clock on<br>1 : All clocks off (stop mode) | O | O |
| Reserved bit | | Always set to "0" | O | O |
| CM14 | fC132 clock select bit | 0 : fC32<br>1 : fC1 | O | O |
| CM15 | Xin -Xout drive capacity select bit (Note 2) | 0 : LOW<br>1 : HIGH | O | O |
| CM16 | Main clock division select bit 1 (Note 3) | b7 b6<br>0 0 : No division mode<br>0 1 : Division by 2 mode<br>1 0 : Division by 4 mode<br>1 1 : Division by 16 mode | | |
| CM17 | | | O | O |

Note 1: Set bit 0 of the protect register (address 000A16) to "1" before writing to this register.
Note 2: This bit changes to "1" when shifting from high-speed/medium-speed mode to stop mode and at a reset. When shifting from low-speed/low power dissipation mode to stop mode, the value before stop mode is retained.
Note 3: Can be selected when bit 6 of the system clock control register 0 (address 000616) is "0". If "1", division mode is fixed at 8.
Note 4: If this bit is set to "1", Xout goes "H", and the built-in feedback resistor is cut off. Xcin and Xcout goes into high impedance state.

**Fig. 1.14. Clock control registers 0 and 1**

## Clock Output

The M30222 provides for a clock output signal (P73/CLKOUT pin) of user defined frequency. The clock output function select bit (CM00, CM01) allows you to choose the clock source from f1, fC1, or a divide-by-n clock for output to the P73/CLKOUT pin. The clock divide counter is an 8-bit counter whose count source is f32, and its divide ratio can be set in the range of 0016 to FF16. Also, the clock divided counter can be controlled for start or stop by the clock divide counter start flag. Figure 1.15 shows a block diagram of clock output. Figure 1.16 shows a clock divided counter related register.

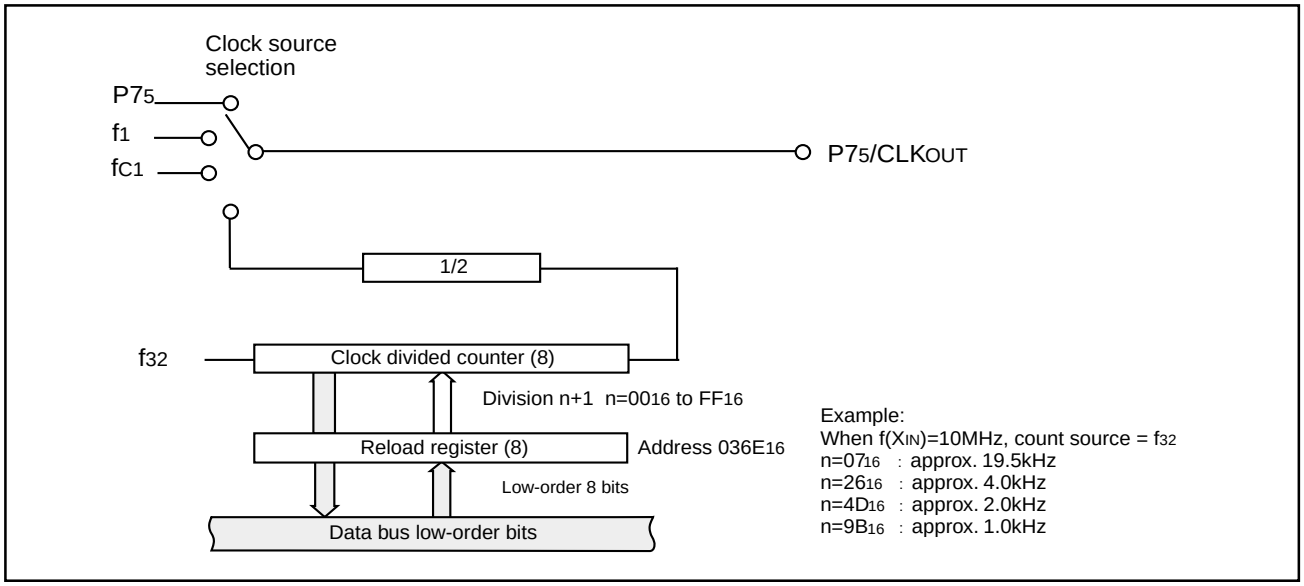



**Fig. 1.15. Block diagram of clock output**

Clock divided counter

b7 b0

Symbol: CDC
Address: 036E16
When reset: XX16

| Function | Values that can be set | R | W |
|---|---|---|---|
| 8-bit timer | 0016 to FF16 | ○ | ○ |

Clock divided counter control register

b7 b6 b5 b4 b3 b2 b1 b0

Symbol: CDCC
Address: 036C16
When reset: 0XXXXXXX2

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Nothing is assigned. Write "0" when writing to these bits. When read, the value is indeterminate. | | | — | — |
| CDCS | Clock divided counter start flg | 0 : Stop<br>1 : Start | ○ | ○ |

**Fig. 1.16. Clock divided counter related register**

## Wait Mode

When a WAIT instruction is executed, the BCLK stops and the microcomputer enters the wait mode. In this mode, oscillation continues but the BCLK and Watchdog timer stop. Writing “1” to the WAIT peripheral function clock stop bit and executing a WAIT instruction stops the clock being supplied to the internal peripheral functions, allowing power dissipation to be reduced. Table 1.11 shows the status of the ports in wait mode.

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

### Usage Precautions

When switching to either wait mode or stop mode, instructions occupying four bytes either from the WAIT instruction or from the instruction that sets the every-clock stop bit to “1” within the instruction queue are prefetched and then the program stops. So put at least four NOPs in succession either to the WAIT instruction or to the instruction that sets the every-clock stop bit to “1”.

**Table 1.11. Port Status during wait mode**

| Pin | Mode | Single-chip mode |
|---|---|---|
| Port | | Retains status before wait mode |
| CLKOUT/$P7_5$ | When $f_{C1}$ selected | Does not stop |
| | When f1, clock divided counter output selected | Retains status before stop mode. Does not stop when the WAIT peripheral function clock stop bit is "0". When the WAIT peripheral function clock stop bit is "1", the status immediately prior to entering wait mode is maintained. |

## Stop Mode

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

Because the oscillation , BCLK, $f_1$ to $f_{32}$, $f_C$, $f_{C32}$, $f_{C1}$, $f_{C32}$ and $f_{AD}$ stops in stop mode, peripheral functions such as the A-D converter and watchdog timer do not function. However, Timer A and Timer B operate provided that the event counter mode is set to an external pulse, and UART0 to UART2 functions provided an external clock is selected. Table 1.12 shows the status of the ports in stop mode. Stop mode is cancelled by a hardware reset or an interrupt. If an interrupt is to be used to cancel stop mode, that interrupt must first have been enabled. If coming out of stop mode is caused by an interrupt, that interrupt routine is executed.

When shifting from high-speed/medium-speed mode to stop mode and at a reset, the main clock division select bit 0 (bit 6 at address $0006_{16}$) is set to “1”. When shifting from low-speed/low power dissipation mode to stop mode, the value before stop mode is retained.

### Usage Precautions

(1) When returning from stop mode by hardware reset, $\overline{\text{RESET}}$ pin must be set to “L” level until main clock oscillation is stabilized.

(2) When switching to either wait mode or stop mode, instructions occupying four bytes either from the WAIT instruction or from the instruction that sets the every-clock stop bit to “1” within the instruction queue are prefetched and then the program stops. Put at least four NOPs in succession either to the WAIT instruction or to the instruction that sets the every-clock stop bit to “1”.

**Table 1.12 Port status during stop mode**

| Pin | Mode | Status |
|---|---|---|
| Port | | Retains status before stop mode |
| CLKOUT/ $P7_5$ | When fc1 selected | "H" |
| | When f1, clock divided output selected | Retains status before stop mode |

## Status Transition Of BCLK

Power dissipation can be reduced and low-voltage operation achieved by changing the count source for BCLK. Table 1.13 shows the operating modes corresponding to the settings of system clock control registers 0 and 1.

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

### (1) Division by 2 mode

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

### (2) Division by 4 mode

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

### (3) Division by 8 mode

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

### (4) Division by 16 mode

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

### (5) No-division mode

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

### (6) Low-speed mode

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

### (7) Low power dissipation mode

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

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

**Table 1.13. Operating modes dictated by settings of system clock control registers 0 and 1**

| CM17 | CM16 | CM07 | CM06 | CM05 | CM04 | BCLK operating mode |
|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 | Invalid | Divide by 2 |
| 1 | 0 | 0 | 0 | 0 | Invalid | Divide by 4 |
| Invalid | Invalid | 0 | 1 | 0 | Invalid | Divide by 8 |
| 1 | 1 | 0 | 0 | 0 | Invalid | Divide by 16 |
| 0 | 1 | 0 | 0 | 0 | Invalid | None |
| Invalid | Invalid | 1 | Invalid | 0 | 1 | Low-speed |
| Invalid | Invalid | 1 | Invalid | 1 | 1 | Low power dissipation |

## Voltage Down Converter

The Voltage Down Converter (VDC) is a bandgap reference based voltage regulator used for generating a low-voltage supply. The VDC block inputs the external supply Vcc (up to 5.5 volts) and generates a 3.3-volt (nominal) supply (VDD). Table 1.14 describes the specified voltage regulation. The VDC is programmable in terms of drive limit and power level. In low power mode, the VDC can source up to 20mA and uses less than 10uA bias current. In high-power mode, the VDC can source up to 200mA. There is a programmable option to limit the current of the VDC in high-power mode to about 80mA. The VDC default state (from reset) is high-power mode with current limiting enabled. The current limiting is enabled at reset in order to avoid a large in-rush current to an external hold capacitor (required) on the VDC pin. Once the external hold capacitor is charged, the current limiter can be disabled in software. Figures 1.17 and 1.18 describe the programmable features of the VDC. The external hold capacitor is required to stabilize the VDC and to minimize voltage ripple on the 3.3 volt supply during operation. Table 1.15 describes the external hold capacitor requirements.

**Table 1.14. VDC voltage regulations**

| Signal | Description |
|---|---|
| Package Supply (Vcc) | Range: 2.7v to 5.5v (input to VDC) |
| Internal Supply (Vdd) | 3.3v (nominal) +/- 10% (output from VDC) OR<br>Vcc - 200mV @ $Icc_{(AVG)}$ <15 mA (Note) |

Note: Whichever is smaller

Voltage Down Converter control register

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

Symbol: VDCC — Address: 0018₁₆ — When reset: XXX00X00₂

| Bit symbol | Function | R | W |
|---|---|---|---|
| VDCC0 | b1 b0<br>0 0 : VDC enabled<br>0 1 : Reserved<br>1 0 : Reserved<br>1 1 : VDC disabled | ○ | ○ |
| VDCC1 | | ○ | ○ |
| Nothing is assigned. Write "0" when writing to this bit. If read, the value is indeterminate. | | – | – |
| HPOWER | 0 : High power<br>1: Low power | ○ | ○ |
| ILIMEN | 0 : Current limit enabled<br>1 : Current limit disabled | ○ | ○ |
| Nothing is assigned. Write "0" when writing to these bits. If read, the value is indeterminate. | | – | – |

**Figure 1.17. VDC Control/Status Register**

**Table 1.15. Required External Components**

| Component | Value | Material |
|---|---|---|
| External Hold Capacitor | 0.1µF +/- 20% | Ceramic |

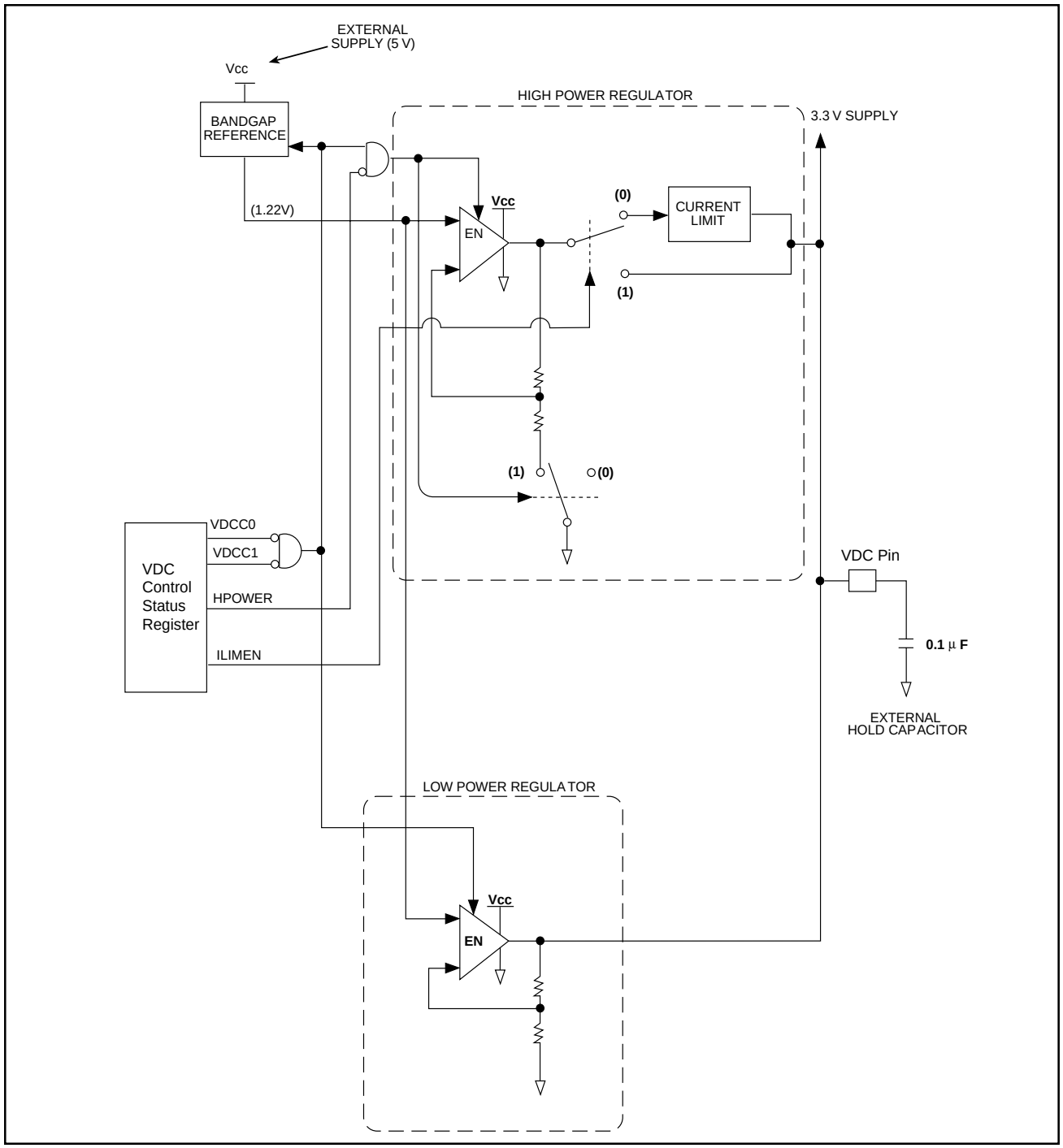



**Fig. 1.18. VDC Functional block diagram**

## Power control

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

### Modes

Power control is available in three modes.

**(a) Normal operation mode**

- **High-speed mode**

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

- **Medium-speed mode**

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

- **Low-speed mode**

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

- **Low power consumption mode**

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

**(b) Wait mode**

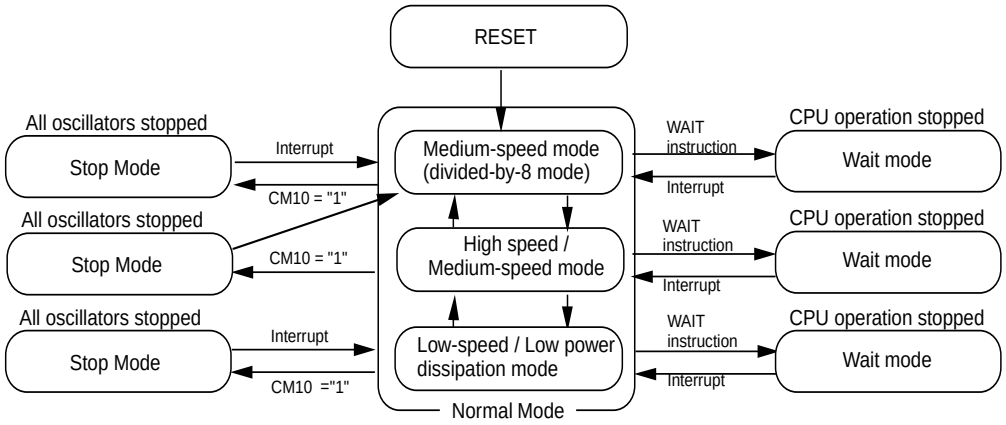
The CPU operation is stopped. The oscillators do not stop.

**(c) Stop mode**

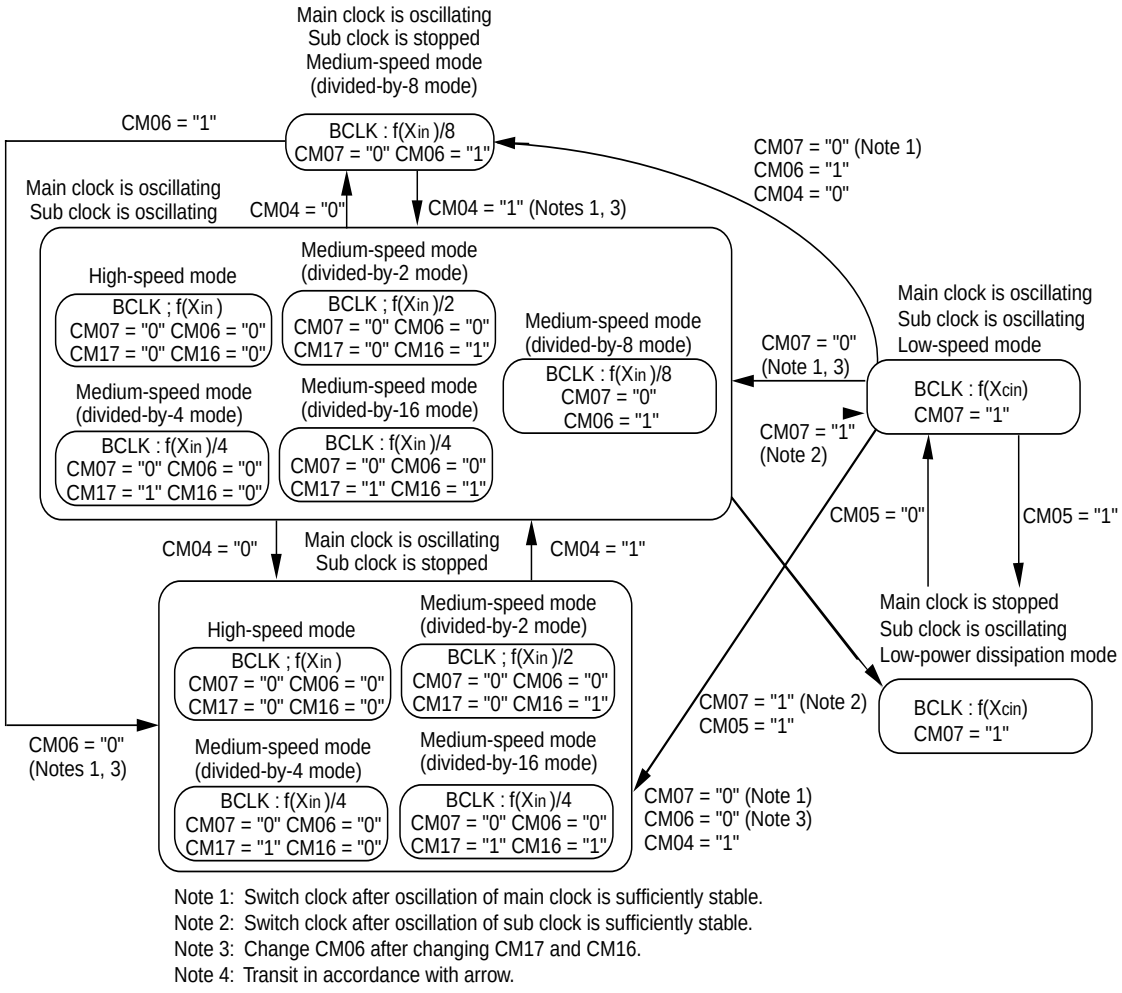
All oscillators stop. The CPU and all built-in peripheral functions stop. This mode, among the three modes listed here, is the most effective in decreasing power consumption.

Figure 1.19 is the state transition diagram of the above modes.

**State Transitions for Stop and Wait modes**




**State Transitions for normal mode**




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

**Fig. 1.19. State Transition diagram of power control mode**

## Software wait

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

A software wait is inserted in the internal ROM/RAM area by setting the wait bit of the processor mode register 1. When set to “0”, each bus cycle is executed in one BCLK cycle. When set to “1”, each bus cycle is executed in two or three BCLK cycles. After the microcomputer has been reset, this bit defaults to “0”. When set to “1”, a wait is applied to all memory areas (two or three BCLK cycles), regardless of the contents of bits 4 to 7 of the chip select control register. Set this bit after referring to the recommended operating conditions (main clock input oscillation frequency) of the electric characteristics.

The SFR area is always accessed in two BCLK cycles regardless of the setting of these control bits. Table 1.16 shows the software wait and bus cycles.

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

**Table 1.16. Software wait and bus cycles**

| Area | Wait bit | Bus cycle |
|---|---|---|
| SFR | Invalid | 2 BCLK cycles |
| Internal ROM/RAM | 0 | 1 BCLK cycle |
| | 1 | 2 BCLK cycles |

## 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 1.20 shows the protect register. The values in the processor mode register 0 (address $0004_{16}$), processor mode register 1 (address $0005_{16}$), system clock control register 0 (address $0006_{16}$), system clock control register 1 (address $0007_{16}$), Port P9 direction register (address $03F3_{16}$) and VDC control register (address $0018_{16}$)can only be changed when the respective bit in the protect register is set to “1”.

The system clock control registers 0 and 1 write-enable bit (bit 0 at $000A_{16}$) and processor mode register 0 and 1 write-enable bit (bit 1 at $000A_{16}$) do not automatically return to “0” after a value has been written to an address. The program must therefore be written to return these bits to “0”.

Protect register

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

| Symbol | Address | When reset |
|---|---|---|
| PRCR | $000A_{16}$ | $XXXX0000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PRC0 | Enables writing to system clock control registers 0 and 1 (addresses $0006_{16}$ and $0007_{16}$) | 0 : Write-inhibited<br>1 : Write-enabled | O | O |
| PRC1 | Enables writing to processor mode registers 0 and 1 (addresses $0004_{16}$ and $0005_{16}$) | 0 : Write-inhibited<br>1 : Write-enabled | O | O |
| PRC2 | Enables writing to Port P9 direction register (address $03F3_{16}$) and SI/Oi control register (i=3,4) (addresses $0362_{16}$ and $0366_{16}$) (Note) | 0 : Write-inhibited<br>1 : Write-enabled | O | O |
| PRC3 | Enables writing to VDC control register (address $0018_{16}$) | 0 : Write-inhibited<br>1 : Write-enabled | O | O |
| Nothing is assigned.<br>Write "0" when writing to these bits. If read, the value is indeterminate. | | | – | – |

Note: Writing a value to these addresses 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

**Fig. 1.20. Protect register**

# Overview of Interrupts

## Types of Interrupts

Figure 1.21 lists the types of interrupts.

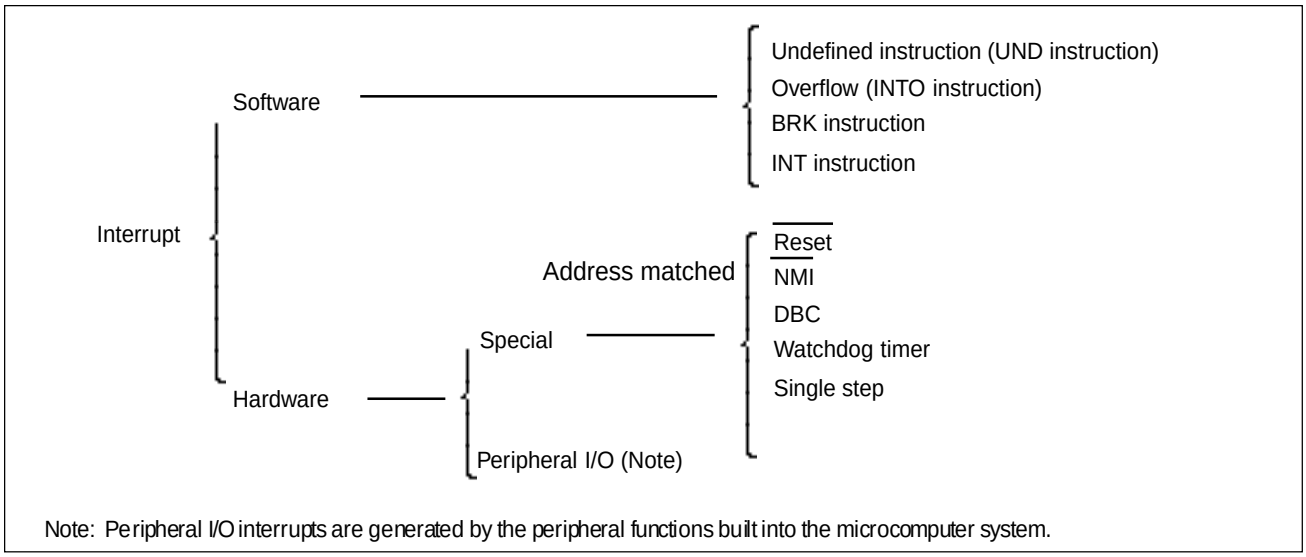



**Figure 1.21. Classification of interrupts**

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

### Software Interrupts

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

**• Undefined instruction interrupt**

An undefined instruction interrupt occurs when executing the UND instruction.

**• Overflow interrupt**

An overflow interrupt occurs when an executing arithmetic instruction overflows. The following instructions will set an O flag when an overflow occurs :

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

**• BRK interrupt**

A BRK interrupt occurs when executing the BRK instruction.

**• INT interrupt**

An INT interrupt occurs when specifying one of the software interrupt numbers 0 through 63 and executing the INT instruction. Software interrupt numbers 0 through 31 are assigned to peripheral I/O interrupts, so executing the INT instruction executes the same interrupt routine as the peripheral I/O interrupt.

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

As far as software interrupt numbers 0 through 31 are concerned, the microcomputer saves the stack pointer assignment flag (U flag) when it accepts an interrupt request. The U flag is set to "0" selecting the interrupt stack pointer then the interrupt sequence is executed. When returning from the interrupt routine, the U flag is returned to its previous state before accepting the interrupt request.

As far as software numbers 32 through 63 are concerned, the stack pointer does not change.

## Hardware Interrupts

Hardware interrupts are classified into two types — special interrupts and peripheral I/O interrupts.

**(1) Special interrupts**

Special interrupts are non-maskable interrupts.

**• Reset**

Reset occurs if an "L" is input to the $\overline{\text{RESET}}$ pin.

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

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

**• DBC interrupt**

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

**• Watchdog timer interrupt**

Generated by the watchdog timer.

**• Single-step interrupt**

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

**• Address match interrupt**

An address match interrupt occurs immediately before the instruction held in the address indicated by the address match interrupt register is executed with the address match interrupt enable bit set to "1". If an address other than the first address of the instruction in the address match interrupt register is set, no address match interrupt occurs.

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

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

**• Bus collision detection interrupt**

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

**• DMA0 interrupt, DMA1 interrupt**

These are interrupts that DMA generates.

**• Key-input interrupt**

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

**• A-D conversion interrupt**

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

**• SIO3, SIO4 interrupt**

These are the interrupts for SIO3, SIO4

**• UART0, UART1, UART2/NACK transmission interrupt**

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

**• UART0, UART1, UART2/ACK reception interrupt**

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

**• Timer A0 interrupt through Timer A4 interrupt**

These are interrupts that Timer A generates

**• Timer B0 interrupt through Timer B5 interrupt**

These are interrupts that Timer B generates.

**• $\overline{\text{INT0}}$ interrupt through $\overline{\text{INT7}}$ interrupt**

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

## Interrupts and Interrupt Vector Tables

If an interrupt request is accepted, program execution branches to the interrupt routine set in the interrupt vector table. Set the first address of the interrupt routine in each vector table. Figure 1.22 shows the format for specifying the address.

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

**Figure 1.22. Format for specifying interrupt vector addresses**

- **Fixed vector tables**

The fixed vector table is a table in which addresses are fixed. The vector tables are located in an area extending from FFFDC16 to FFFFF16. One vector table comprises four bytes. Set the first address of interrupt routine in each vector table. Table 1.17 shows the interrupts assigned to the fixed vector tables and addresses of vector tables.

**Table 1.17. Interrupts assigned to the fixed vector tables and addresses of vector tables**

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

Note: Interrupts used for debugging purposes only.

• **Variable vector tables**

The addresses in the variable vector table can be modified, according to the user's settings. Before enabling interrupts, the user msut load the INTB register with the address of the first entry in the table. The 256-byte area subsequent to the address the INTB indicates becomes the area for the variable vector tables. One vector table comprises four bytes. Set the first address of the interrupt routine in each vector table. Table 1.18 shows the interrupts assigned to the variable vector tables and addresses of vector tables. shows the interrupts assigned to the variable vector tables and addresses of vector tables.

**Table 1.18. Interrupts assigned to the variable vector tables and addresses of vector tables**

| Software interrupt number | Vector table address Address (L) to address (H) | Interrupt source | Remarks |
|---|---|---|---|
| Software interrupt number 0 | +0 to +3 (Note 1) | BRK instruction | Not masked by I flag |
| ≈ | | | ≈ |
| Software interrupt number 4 | +16 to +19 (Note 1) | $\overline{\text{INT3}}$ / SIO4 (Note 3) | |
| Software interrupt number 5 | +20 to +23 (Note 1) | Timer B5 | |
| Software interrupt number 6 | +24 to +27 (Note 1) | Timer B4 | |
| Software interrupt number 7 | +28 to +31 (Note 1) | Timer B3 | |
| Software interrupt number 8 | +32 to +35 (Note 1) | $\overline{\text{INT7}}$ | |
| Software interrupt number 9 | +36 to +39 (Note 1) | $\overline{\text{INT6}}$ | |
| Software interrupt number 10 | +40 to +43 (Note 1) | Bus collision detection | |
| Software interrupt number 11 | +44 to +47 (Note 1) | DMA0 | |
| Software interrupt number 12 | +48 to +51 (Note 1) | DMA1 | |
| Software interrupt number 13 | +52 to +55 (Note 1) | Key input interrupt | |
| Software interrupt number 14 | +56 to +59 (Note 1) | A-D | |
| Software interrupt number 15 | +60 to + 63 (Note 1) | UART2 transmit (Note 2) | |
| Software interrupt number 16 | +64 to +67 (Note 1) | UART2 receive (Note 2) | |
| Software interrupt number 17 | +68 to +71 (Note 1) | UART0 transmit | |
| Software interrupt number 18 | +72 to +75 (Note 1) | UART0 receive | |
| Software interrupt number 19 | +76 to +79 (Note 1) | UART1 transmit | |
| Software interrupt number 20 | +80 to +83 (Note 1) | UART1 receive | |
| Software interrupt number 21 | +84 to +87 (Note 1) | Timer A0 | |
| Software interrupt number 22 | +88 to +91 (Note 1) | Timer A1 | |
| Software interrupt number 23 | +92 to +95 (Note 1) | Timer A2 | |
| Software interrupt number 24 | +96 to +99 (Note 1) | Timer A3/ $\overline{\text{INT4}}$ (Note 3) | |
| Software interrupt number 25 | +100 to +103 (Note 1) | Timer A4 / $\overline{\text{INT5}}$ (Note 3) | |
| Software interrupt number 26 | +104 to +107 (Note 1) | Timer B0 | |
| Software interrupt number 27 | +108 to +111 (Note 1) | Timer B1 | |
| Software interrupt number 28 | +112 to +115 (Note 1) | Timer B2 | |
| Software interrupt number 29 | +116 to +119 (Note 1) | $\overline{\text{INT0}}$ | |
| Software interrupt number 30 | +120 to +123 (Note 1) | $\overline{\text{INT1}}$ | |
| Software interrupt number 31 | +124 to +127 (Note 1) | $\overline{\text{INT2}}$ / SIO3 (Note 3) | |
| Software interrupt number 32 to Software interrupt number 63 | +128 to +131 (Note 1) to +252 to +255 (Note 1) | Software interrupt | Not masked by I flag |

Note 1: Address relative to address in interrupt table register (INTB).
Note 2: When IIC mode is selected, NACK and ACK interrupts are selected.
Note 3: Selected by Interrupt Request Cause Select bit (bits 4, 5, 6, 7 at address 035F16)

## Interrupt Control

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

Enable or disable a maskable interrupt using the interrupt enable flag (I flag), interrupt priority level selection bits, or processor interrupt priority level (IPL). Whether an interrupt request is present or absent is indicated by the interrupt request bit. The interrupt request bit and the interrupt priority level selection bit are located in the interrupt control register of each interrupt. Also, the interrupt enable flag (I flag) and the IPL are located in the CPU flag register (FLG). Figure 1.23 shows the memory map of the interrupt control registers.

Interrupt control register (Note 1)

| Symbol | Address | When reset |
|---|---|---|
| TBiIC(i=3 to 5) | $0045_{16}$ to $0047_{16}$ | XXXX0000 2 |
| BCNIC | $004A_{16}$ | XXXX0000 2 |
| DMiIC(i=0, 1) | $004B_{16}$, $004C_{16}$ | XXXX0000 2 |
| KUPIC | $004D_{16}$ | XXXX0000 2 |
| ADIC | $004E_{16}$ | XXXX0000 2 |
| SiTIC(i=0 to 2) | $0051_{16}$ $0053_{16}$, $004F_{16}$ | XXXX0000 2 |
| SiRIC(i=0 to 2) | $0052_{16}$, $0054_{16}$, $0050_{16}$ | XXXX0000 2 |
| TAiIC(i=0 to 2) | $0055_{16}$ to $0057_{16}$ | XXXX0000 2 |
| TBiIC(i=0 to 2) | $005A_{16}$ to $005C_{16}$ | XXXX0000 2 |

b7 b6 b5 b4 b3 b2 b1 b0

| 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 2) |
| Nothing is assigned.<br>Write "0" when writing to these bits. If read, the value is "0". | | | — | — |

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

| Symbol | Address | When reset |
|---|---|---|
| INTiIC(i=0 to 1) | $005D_{16}$ , $005E_{16}$ | XX000000 2 |
| INT2IC/SI3IC | $005F_{16}$ | XX000000 2 |
| INT31C/SI4IC | $0044_{16}$ | XX000000 2 |
| INT4IC/TA3IC | $0058_{16}$ | XX000000 2 |
| INT5/TA4IC | $0059_{16}$ | XX000000 2 |
| INTiIC(i= 6 to 7) | $0049_{16}$, $0048_{16}$ | XX000000 2 |

b7 b6 b5 b4 b3 b2 b1 b0 (0)

| 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 2) |
| POL | Polarity select bit | 0: Selects falling edge<br>1: Selects rising edge | ○ | ○ |
| Reserved bit | | Always set to "0" | ○ | ○ |
| Nothing is assigned.<br>Write "0" when writing to these bits. If read, the value is "0". | | | — | — |

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

**Figure 1.23. Memory map of the interrupt control registers.**

### Interrupt Enable Flag (I flag)

The interrupt enable flag (I flag) controls the enabling and disabling of maskable interrupts. Setting this flag to “1” enables all maskable interrupts; setting it to “0” disables all maskable interrupts. This flag is set to “0” after reset.

### Interrupt Request Bit

The interrupt request bit is set to "1" by hardware when an interrupt is requested. After the interrupt is accepted and jumps to the corresponding interrupt vector, the request bit is set to "0" by hardware. The interrupt request bit can also be set to "0" by software. (Do not set this bit to "1").

### Interrupt Priority Level Select Bit and Processor Interrupt Priority Level (IPL)

Set the interrupt priority level using the interrupt priority level select bits, which consists of three interrupt control register bits. When an interrupt request occurs, the interrupt priority level is compared with the IPL of the CPU flag register. The interrupt is enabled only when the priority level of the interrupt is higher than the IPL. Therefore, setting the interrupt priority level to “0” disables the interrupt.

Table 1.19 shows the settings of interrupt priority levels and Table 1.20 shows the interrupt levels enabled, according to the contents of the IPL.

The following are conditions under which an interrupt is accepted:
•interrupt enable flag (I flag) = 1
•interrupt request bit = 1 (set by hardware)
•interrupt priority level > IPL

The interrupt enable flag (I flag), the interrupt request bit, the interrupt priority select bit, and the IPL are independent, and they are not affected by one another.

**Table 1.19. Settings of interrupt priority levels**

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

**Table 1.20. Interrupt levels enabled according to the contents of the IPL**

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

### Modifying the interrupt control register

When modifying the interrupt control register, do so at a point that does not generate the interrupt request for that register. If there is a possibility of the interrupt request occurring, access the interrupt control register after the interrupt is disabled. The program examples are described below:

**Example 1:**

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

**Example 2:**

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

**Example 3:**

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

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

When modifying an interrupt control register, it is recommended to use only the instructions: AND, OR, BCLR and BSET. Using the "MOV" or other instruction may cause an interrupt to be missed.

## Interrupt Sequence

The interrupt sequence, described below, is performed over a period from the instant an interrupt is accepted to the instant the interrupt routine is executed.

If an interrupt occurs during execution of an instruction, the processor determines its priority when the execution of the instruction is completed, and transfers control to the interrupt sequence from the next cycle. If an interrupt occurs during execution of either the SMOVB, SMOVF, SSTR or RMPA instruction, the processor temporarily suspends the instruction being executed, and transfers control to the interrupt sequence.

The processor carries out the following in sequence after an interrupt request:

(1) CPU gets the interrupt information (the interrupt number and interrupt request level) by reading address $0000_{16}$.

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

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

(4) Saves the contents of the temporary register (Note) within the CPU in the stack area.

(5) Saves the contents of the program counter (PC) in the stack area.

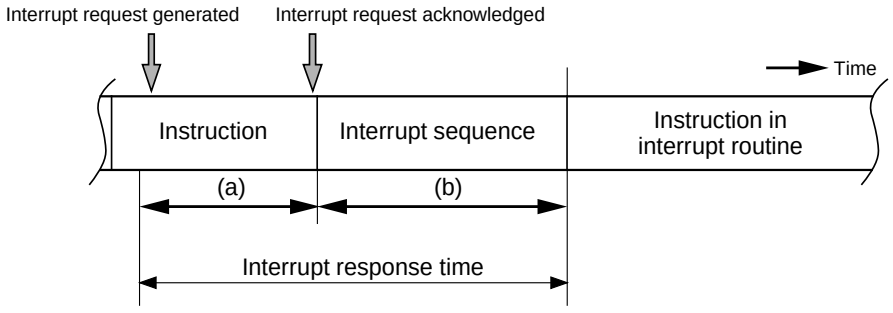
(6) Sets the interrupt priority level of the accepted instruction in the IPL.

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

Note: This register cannot be utilized by the user.

### Interrupt Response Time

'Interrupt response time' is the period between the instant an interrupt occurs and the instant the first instruction within the interrupt routine has been executed. This time comprises the period from the occurrence of an interrupt to the completion of the instruction under execution at that moment (a) and the time required for executing the interrupt sequence (b). Figure 1.24 shows the interrupt response time.




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

**Figure 1.24. Interrupt response time**

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

Time (b) is as shown in Table 1.21.

**Table 1.21. Time required for executing the interrupt sequence**

| Interrupt vector address | Stack pointer (SP) value | 16-bit bus, without wait | 8-bit bus, without wait |
|---|---|---|---|
| Even | Even | 18 cycles (Note 1) | 20 cycles (Note 1) |
| Even | Odd | 19 cycles (Note 1) | 20 cycles (Note 1) |
| Odd (Note 2) | Even | 19 cycles (Note 1) | 20 cycles (Note 1) |
| Odd (Note 2) | Odd | 20 cycles (Note 1) | 20 cycles (Note 1) |

Note 1: Add 2 cycles in the case of a DBC interrupt.
Add 1 cycle in the case of either an address coincidence interrupt or a single-step interrupt.
Note 2: If possible, locate an interrupt vector address in an even address.

Figure 1.25 shows the time required for executing the interrupt sequence

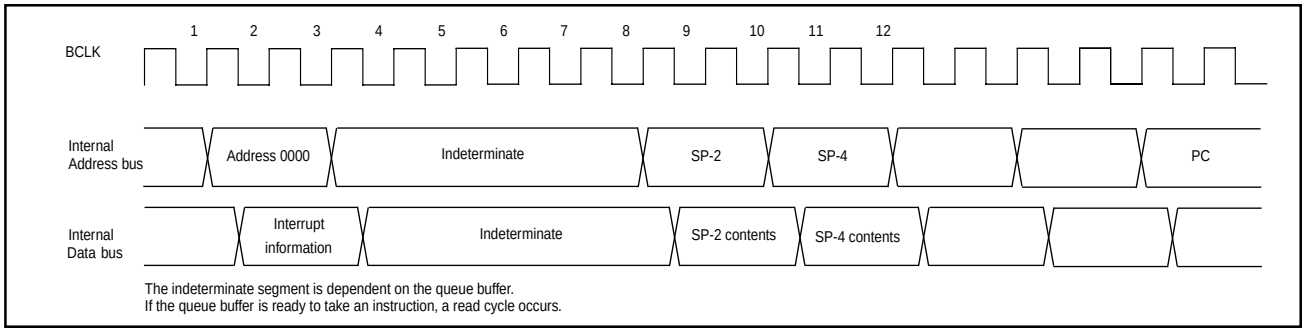



**Fig. 1.25. Time required for executing the interrupt sequence**

### Variation of IPL when Interrupt Request is Accepted

If an interrupt request is accepted, the interrupt priority level of the accepted interrupt is set in the IPL.
If an interrupt request, that does not have an interrupt priority level, is accepted, one of the values shown in Table 1.22 is set in the IPL.

**Table 1.22. Relationship between interrupts without interrupt priority levels and IPL**

| Interrupt sources without priority levels | Value set in the IPL |
|---|---|
| Watchdog timer, $\overline{\text{NMI}}$ | 7 |
| $\overline{\text{RESET}}$ | 0 |
| Other | No change |

### Saving Registers

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

First, the processor saves the four higher-order bits of the program counter, and 4 upper-order bits and 8 lower-order bits of the FLG register, 16 bits in total, in the stack area, then saves 16 lower-order bits of the program counter. Figure 1.26 shows the state of the stack as it was before the acceptance of the interrupt request, and the state the stack after the acceptance of the interrupt request.

Save other necessary registers at the beginning of the interrupt routine using software. Using the PUSHM instruction alone can save all the registers except the stack pointer (SP).

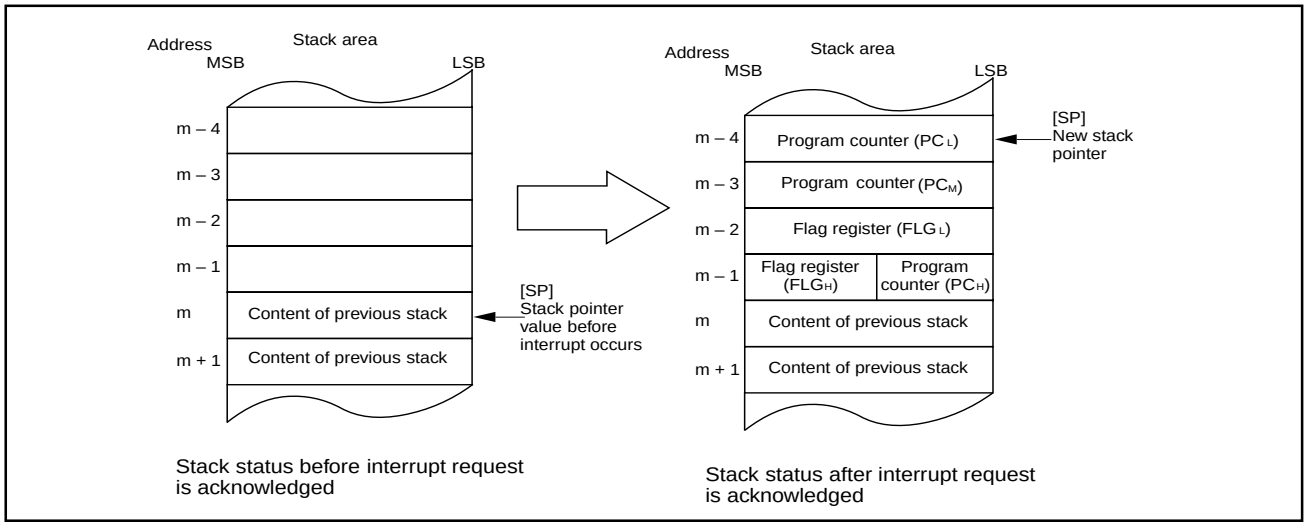
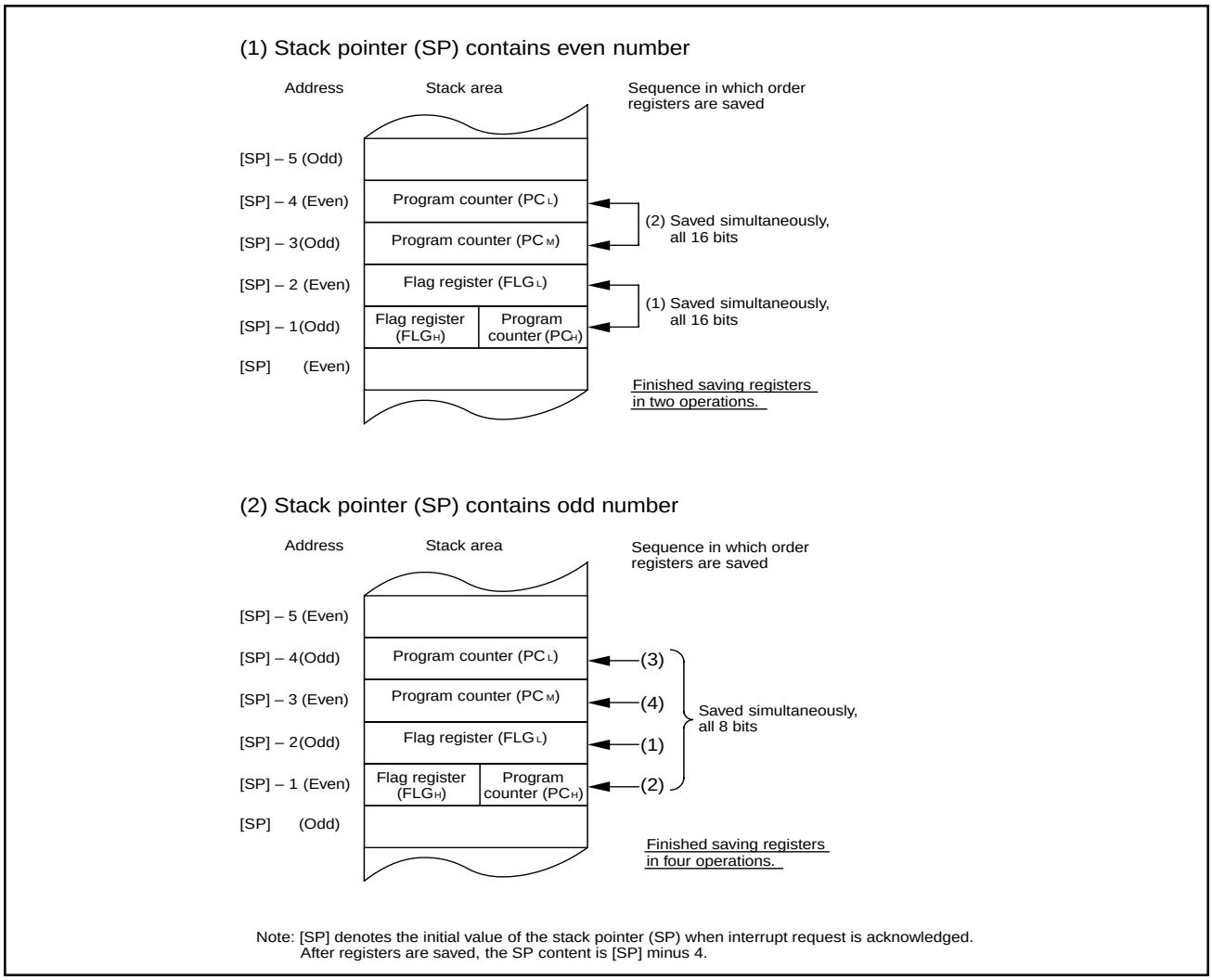



**Figure 1.26. State of stack before and after acceptance of interrupt request**

The operation of saving registers carried out in the interrupt sequence is dependent on whether the content of the stack pointer, at the time of acceptance of an interrupt request, is even or odd. If the content of the stack pointer (Note) is even, the content of the flag register (FLG) and the content of the program counter (PC) are saved, 16 bits at a time. If odd, their contents are saved in two steps, 8 bits at a time. Figure 1.27 shows the operation of the saving registers.

Note: When any INT instruction in software number 32 to 63 is executed, the stack pointer is indicated by the U Flag, otherwise, it is the interrupt stack pointer (ISP)




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

**Figure 1.27. Operation of saving registers**

### Returning from an Interrupt Routine

Executing the REIT instruction at the end of an interrupt routine returns the contents of the flag register (FLG) as it was immediately before the start of interrupt sequence and the contents of the program counter (PC), both of which have been saved in the stack area. Then control returns to the program that was being executed before the acceptance of the interrupt request, so that the suspended process resumes.

Return the other registers saved by software within the interrupt routine using the POPM or similar instruction before executing the REIT instruction.

### Interrupt Priority

If there are two or more interrupt requests occurring at a point in time within a single sampling (checking whether interrupt requests are made), the interrupt assigned a higher priority is accepted.

Assign an arbitrary priority to maskable interrupts (peripheral I/O interrupts) using the interrupt priority level select bits. If the same interrupt priority level is assigned, however, the interrupt assigned a higher hardware priority is accepted.

Priorities of the special interrupts, such as Reset (dealt with as an interrupt assigned the highest priority), watchdog timer interrupt, etc. are regulated by hardware.

Software interrupts are not affected by the interrupt priority. If an instruction is executed, control is distributed to the interrupt routine. Figure 1.28 shows the priorities of hardware interrupts.

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

**Figure 1.28. Hardware interrupts priorities**

### Interrupt resolution circuit

When two or more interrupts are generated simultaneously, this circuit selects the interrupt with the highest priority level. Figure 1.29 shows the circuit that judges the interrupt priority level.

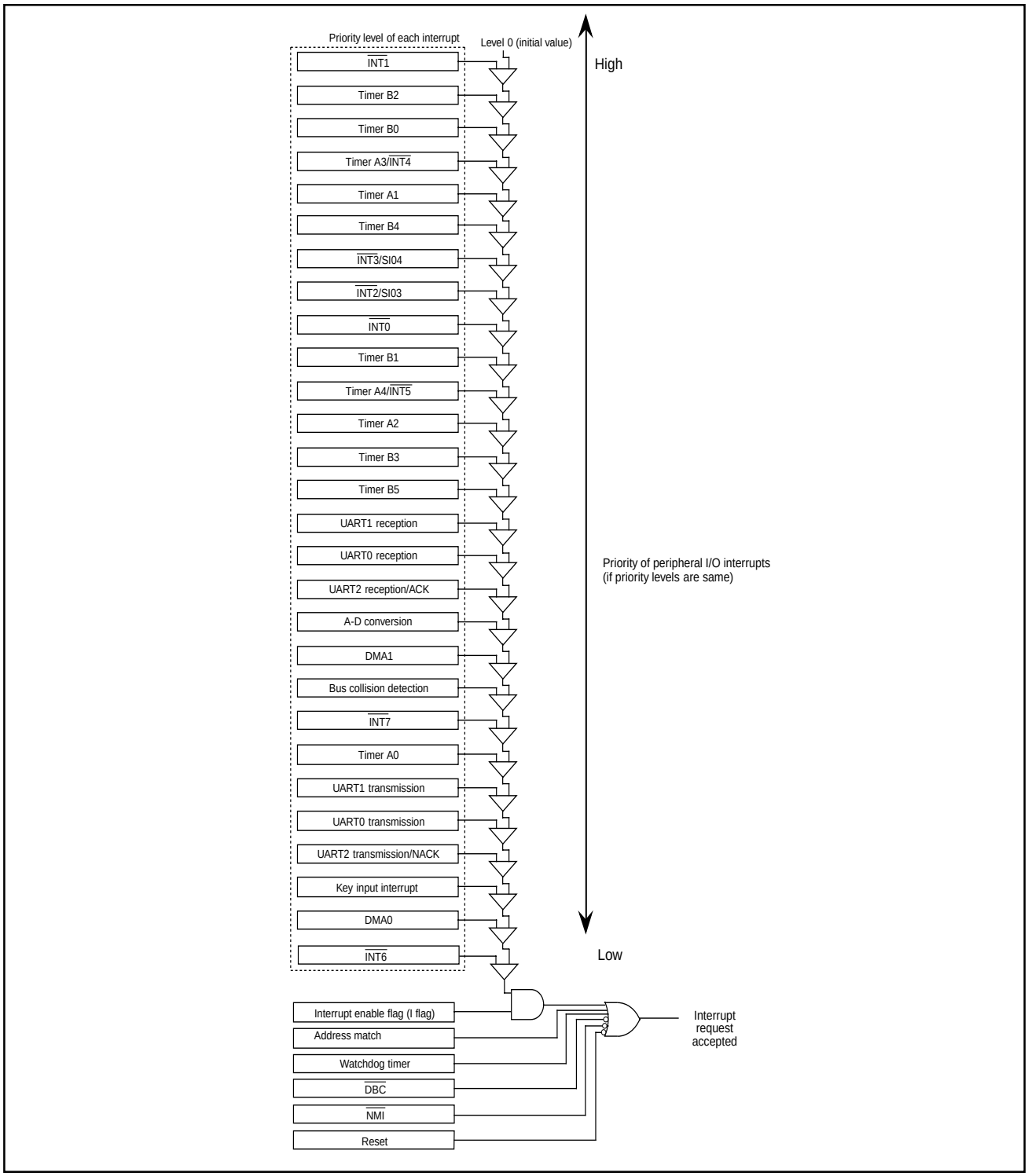



**Figure 1.29. Maskable interrupts priorities (peripheral I/O interrupts)**

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

$\overline{\text{INT}}$0 to $\overline{\text{INT}}$7 are triggered by the edges of external inputs. The edge polarity is selected using the polarity select bit. The interrupt control registers, 0058$_{16}$ is used both as Timer A3 and external interrupt $\overline{\text{INT}}$4 input control register, and 0059$_{16}$ is used both as Timer A4 and as external interrupt $\overline{\text{INT}}$5 input control register. Also, 005F$_{16}$ is used as both SIO3 and external interrupt $\overline{\text{INT}}$2 input control register and 0044$_{16}$ is used as both SIO4 and external interrupt $\overline{\text{INT}}$3 input control register. Use the interrupt request cause select bits - bits 4, 5, 6 and 7 of the interrupt request cause select register 0 (address 035E$_{16}$) - to specify which interrupt request cause to select. When $\overline{\text{INT}}$4 is selected as an interrupt source, the input port for it can be selected by bits 0 and 1 of the interrupt source select register 0 (address 035E$_{16}$). Similarly, when $\overline{\text{INT}}$5 is selected as an interrupt source, the input port for it can be selected by bits 2 and 3 of the interrupt source select register 0 (address 035E$_{16}$). After having set an interrupt request cause and interrupt input ports, be sure to set the corresponding interrupt request bit to "0" before enabling an interrupt.

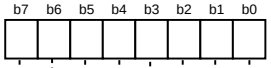
The interrupt control registers - 0058$_{16}$, 0059$_{16}$, 005F$_{16}$, and 0044$_{16}$ - have the polarity-switching bit. Be sure to set this bit to “0” to select a timer or SIO as the interrupt request cause.

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

When $\overline{\text{INT}}$4 input pin select bits = "11", $\overline{\text{INT}}$4 interupt polarity switching bit = "0", and polarity select bit = "1" of the $\overline{\text{INT}}$4 interrupt control register, an interrupt is generated by a rising edge on the input port when the exclusive pin is "H", as shown by "Single edge, Rise" in Figure 1.32. When the exclusive pin is "H", interrupts can only be generated by an active transition on a single edge. The same applies to $\overline{\text{INT}}$5.

Figure 1.30 shows the Interrupt request cause select registers. Figure 1.31 shows the block diagram of $\overline{\text{INT}}$4 and $\overline{\text{INT}}$5.

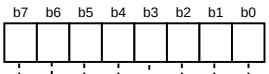
Interrupt request cause select register 0




Symbol: IFSR0 | Address: 035E16 | When reset: 0016

| Bit symbol | Bit name | Fumction | R | W |
|---|---|---|---|---|
| IFSR00 | INT4 input pin select bit | 0 0 : No INT4 input<br>0 1 : P76 input enable<br>1 0 : P77 input enable<br>1 1 : P76, P77 input enable | ○ | ○ |
| IFSR01 | | | ○ | ○ |
| IFSR02 | INT5 input pin select bit | 0 0 : No INT5 input<br>0 1 : P80 input enable<br>1 0 : P81 input enable<br>1 1 : P80, P81 input enable | ○ | ○ |
| IFSR03 | | | ○ | ○ |
| IFSR04 | Interrupt request cause select bit | 0 : SIO3<br>1 : INT2 | ○ | ○ |
| IFSR05 | Interrupt request cause select bit | 0 : SIO4<br>1 : INT3 | ○ | ○ |
| IFSR06 | Interrupt request cause select bit | 0 : Timer A3<br>1 : INT4 | ○ | ○ |
| IFSR07 | Interrupt request cause select bit | 0 : Timer A4<br>1 : INT5 | ○ | ○ |

Interrupt request cause select register 1




Symbol: IFSR1 | Address: 035F16 | When reset: 0016

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| IFSR10 | INT0 interrupt polarity swiching bit | 0 : Single edge<br>1 : Both edges | ○ | ○ |
| IFSR11 | INT1 interrupt polarity swiching bit | 0 : Single edge<br>1 : Both edges | ○ | ○ |
| IFSR12 | INT2 interrupt polarity swiching bit | 0 : Single edge<br>1 : Both edges | ○ | ○ |
| IFSR13 | INT3 interrupt polarity swiching bit | 0 : Single edge<br>1 : Both edges | ○ | ○ |
| IFSR14 | INT4 interrupt polarity swiching bit | 0 : Single edge<br>1 : Both edges | ○ | ○ |
| IFSR15 | INT5 interrupt polarity swiching bit | 0 : Single edge<br>1 : Both edges | ○ | ○ |
| IFSR16 | INT6 interrupt polarity swiching bit | 0 : Single edge<br>1 : Both edges | ○ | ○ |
| IFSR17 | INT7 interrupt polarity swiching bit | 0 : Single edge<br>1 : Both edges | ○ | ○ |

**Figure 1.30. Interrupt request cause select register**

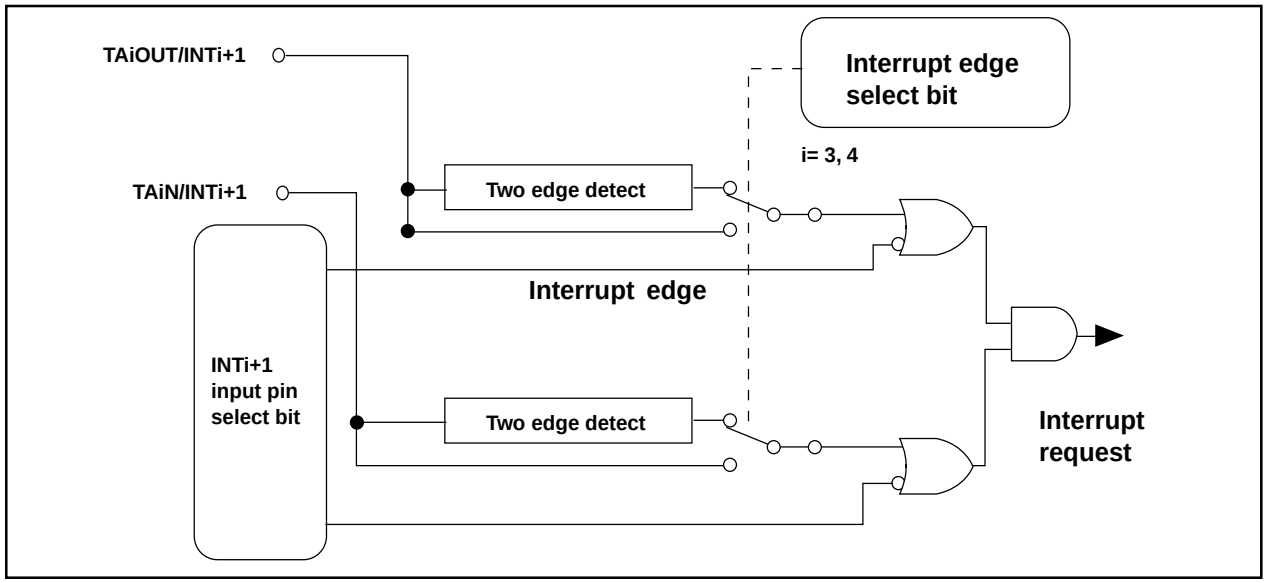



**Figure 1.31. $\overline{\text{INT4}}$ and $\overline{\text{INT5}}$ block diagram**

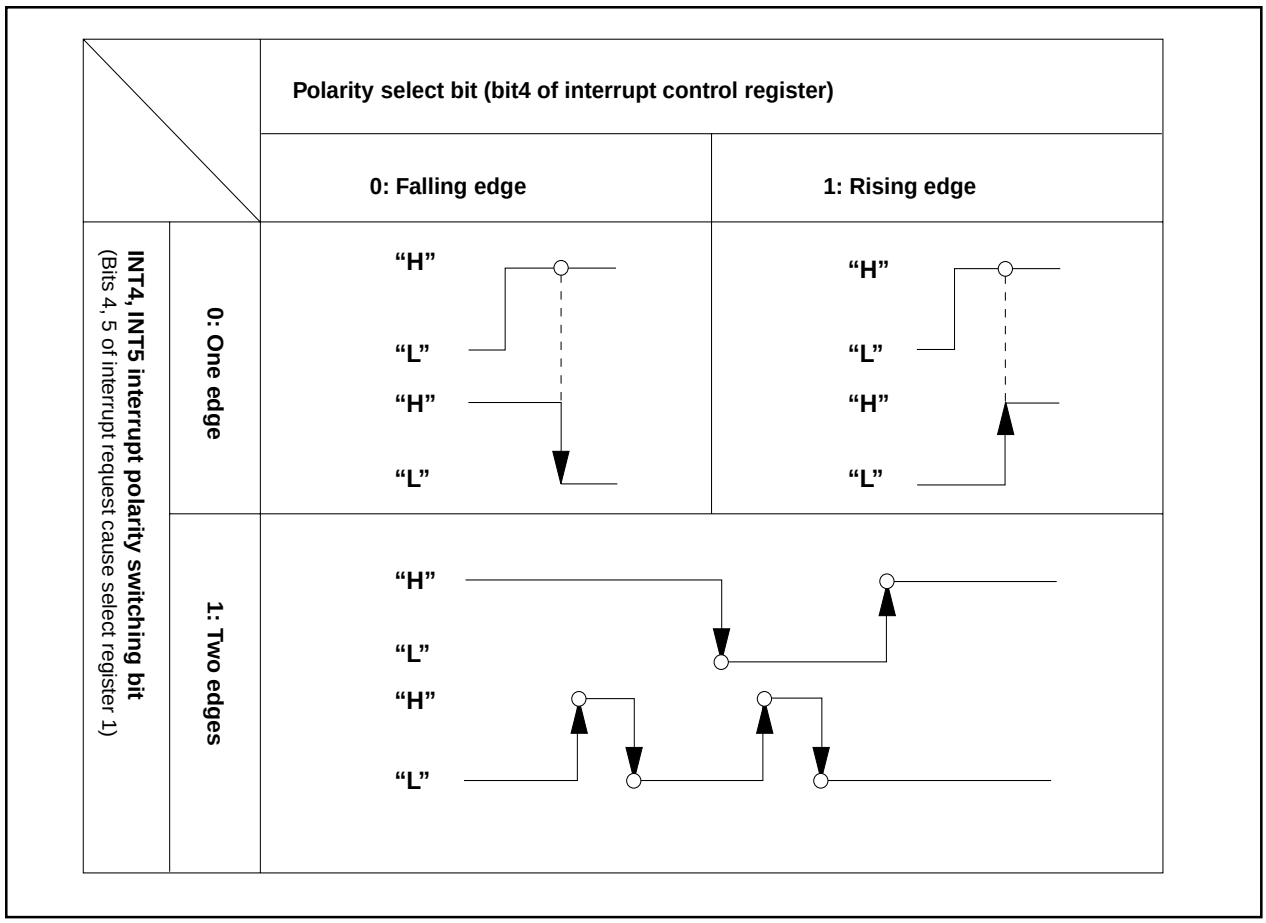



**Figure 1.32. Typical timing of interrupts $\overline{\text{INT4}}$ and $\overline{\text{INT5}}$**

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

An $\overline{\text{NMI}}$ interrupt is generated when the input to the P8$_3$/$\overline{\text{NMI}}$ 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$_3$ register (bit 3 at address 03F0$_{16}$). This pin cannot be used as a normal port input. (See Interrupt Precautions section).

### Key Input Interrupt

All bits of Port 6 can be used as Key Input interrupts. Enable the interrupts using the KUPIC register, then set the direction register of any of P6$_0$ to P6$_7$ bits for input, and a falling edge to that port will generate a key input interrupt. A key input interrupt can also be used as a key-on wakeup function for cancelling the wait mode or stop mode.

Figure 1.33 shows the block diagram of the key input interrupt. Note that if an “L” level is input to any pin that has not been disabled for input, inputs to the other pins are not detected as an interrupt.

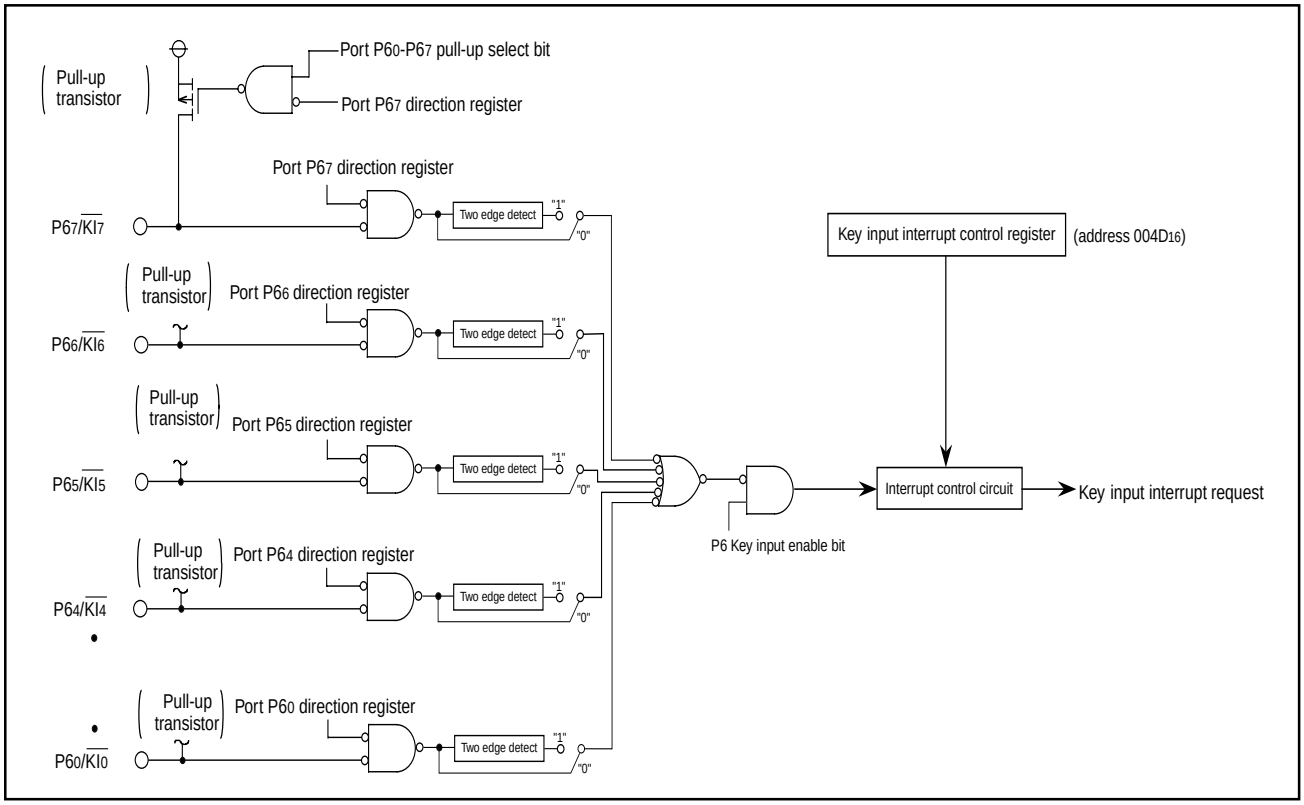
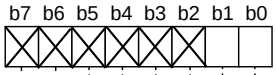



**Figure 1.33. Block diagram of key input interrupt**

Figure 1.34 shows the Key input mode register. With bits 0 and 1 of this register, it is possible to select both edges or the fall edge of the key input for P6. Port P6 is set for pull-up using the pll-up control register.

Key input mode register




| Symbol | Address | When reset |
|---|---|---|
| KUPM | $0126_{16}$ | $XXXXXX00_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| P6KIS | P6 key input select bit (Note) | 0 : Falling edge<br>1 : Two edges | ○ | ○ |
| P6KIE | P6 key input enable bit | 0 : Disable<br>1 : Enable | ○ | ○ |
| Nothing is assigned. Write "0" when writing to these bits. The value is indeterminate when read. | | | — | — |

Note : If this bit is set for “Two edges” when the corresponding port has been specified to have a pull up, the port is automatically pulled high intermittently by the operating subclock.

The pull-up resistance is not connected for pins that are set for output from peripheral functions, regardless of the setting in the pull-up control register

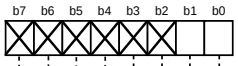
**Fig. 1.34. Key input mode register**

### Address Match Interrupt

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

Figure 1.35 shows the address match interrupt-related registers.

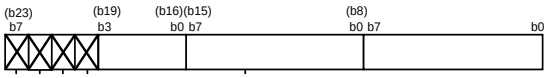
Address match interrupt enable register




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

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| AIER0 | Address match interrupt 0 enable bit | 0 : Interrupt disabled<br>1 : Interrupt enabled | O | O |
| AIER1 | Address match interrupt 1 enable bit | 0 : Interrupt disabled<br>1 : Interrupt enabled | O | O |
| Nothing is assigned.<br>Write 0 when writing to these bits. If read, the value is indeterminate. | | | — | — |

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




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

| Function | Values that can be set | R | W |
|---|---|---|---|
| Address setting register for address match interrupt | $00000_{16}$ to $FFFFF_{16}$ | O | O |
| Nothing is assigned.<br>Write 0 when writing to these bits. If read, the value is indeterminate. | | — | — |

**Figure 1.35. Address match interrupt-related registers.**

## Precautions for Interrupts

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

• When a maskable interrupt occurs, the CPU reads the interrupt information (the interrupt number and interrupt request level)from address 00000$_{16}$ in the interrupt sequence.

The interrupt request bit of the interrupt written in address 00000$_{16}$ will then be set to “0”.
Reading address 00000$_{16}$ by software enables the highest priority interrupt source request bit.
Though the interrupt is generated, the interrupt routine may not be executed.
Do not read address 00000$_{16}$ by software.

### (2) Setting the stack pointer

• The stack pointers immediately after reset are initialized to 0000$_{16}$. The stack pointers must nbe set to valid RAM areas for proper operation. An interrupt occurring immediately after reset will cause a runaway condition.

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

•The $\overline{\text{NMI}}$ interrupt can not be disabled. Be sure to connect $\overline{\text{NMI}}$ pin to Vcc 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.
• Do not reset the CPU with the input to the $\overline{\text{NMI}}$ pin being in the “L” state.
• Do not attempt to go into stop mode with the input to the $\overline{\text{NMI}}$ pin being in the “L” state. With the input to the NMI being in the “L” state, the CM10 is fixed to “0”, so attempting to go into stop mode is ignored.
• Do not attempt to go into wait mode with the input to the $\overline{\text{NMI}}$ pin being in the “L” state. With the input to the $\overline{\text{NMI}}$ pin being in the “L” state, the CPU stops but the oscillation does not stop, so no power is saved. In this instance, the CPU is returned to the normal state by a later interrupt.
• Minimum $\overline{\text{NMI}}$ pulse width is 1 BCLK cycle.

### (4) External interrupts

• A minimum of 250ns pulse width is necessary for the signal input to pins INT$_0$ through $\overline{\text{INT}_7}$ regardless of the CPU operation clock.
• When the polarity of the $\overline{\text{INT}_0}$ to $\overline{\text{INT}_7}$ pins is changed, the interrupt request bit is sometimes set to "1". After changing the polarity, set the interrupt request bit to "0". Figure 1.36 shows the procedure for changing the $\overline{\text{INT}}$ interrupt generate factor.

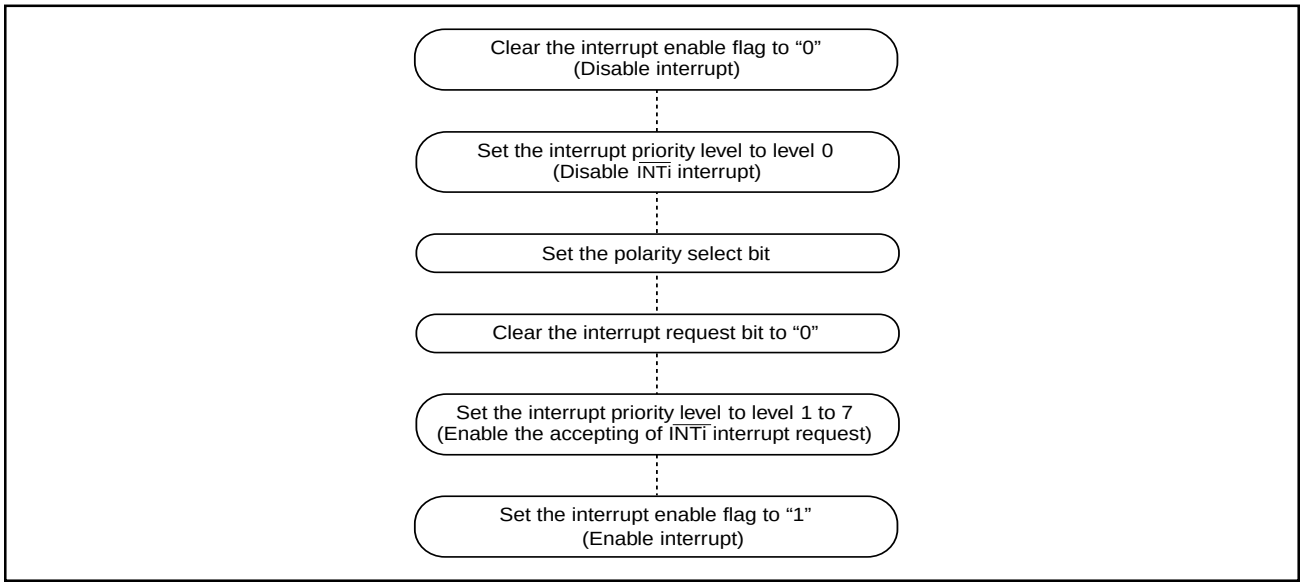



**Figure 1.36. Switching condition of $\overline{\text{INT}}$ interrupt request**

### (5) Rewrite the interrupt control register

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

**Example 1:**

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

**Example 2:**

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

**Example 3:**

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

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

• When modifying an interrupt control register, it is recommended to use only the instructions: AND, OR, BCLR, BSET. Using the "MOV" or other instruction may cause an interrrupt to be missed.

## Watchdog Timer

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

**With XIN chosen for BCLK**

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

**With XCIN chosen for BCLK**

$$\text{Watchdog timer period} = \frac{\text{prescaler dividing ratio (2) X Watchdog timer count (32768)}}{\text{BCLK}}$$

For example, suppose that BCLK runs at 16 MHz and that 16 has been chosen for the dividing ratio of the prescaler, then the Watchdog timer's period becomes approximately 32.8 ms.

The Watchdog timer is initialized by writing to the watchdog timer start register (address 000E16) and when a Watchdog timer interrupt request is generated. The prescaler must be set before initializing the Watchdog timer. Once initialized, the Watchdog timer can only be stopped by a reset. The counter is reset to 7EEE16 by writing any value to the Watchdog timer start register (address 000E16). Figure 1.37 shows the Watchdog timer block diagram. Figure 1.38 shows the Watchdog timer-related registers.

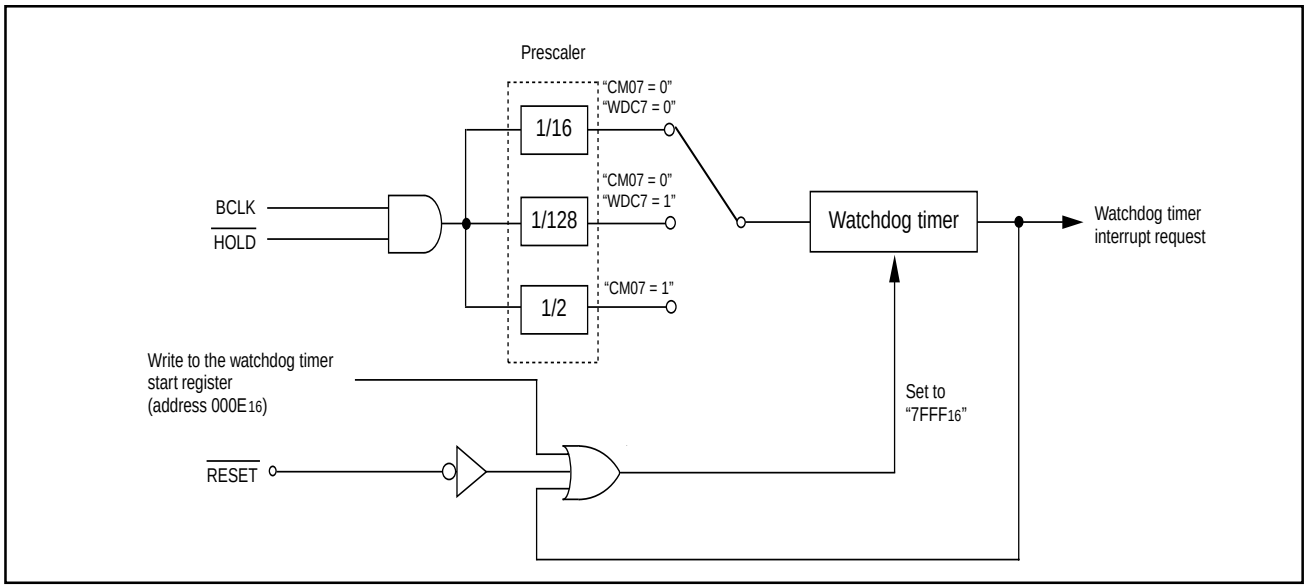



**Fig. 1.37. Block diagram of Watchdog timer**

Watchdog timer control register (Note)

| 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 | ○ | ○ |

Note: Set the desired prescale value before initializing the Watchdog timer.

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 the first write instruction to this register after reset. Writing any value to this register resets the counter to $7FFF_{16}$. | × | ○ |

**Fig. 1.38. Watchdog timer control and start registers**

## DMAC

This microcomputer has two DMAC (direct memory access controller) channels that allow data to be sent to memory without using the CPU. The DMAC shares the same data bus with the CPU. The DMAC uses a high speed cycle-stealing method because it has a higher right to use the bus than the CPU. DMA transfers word (16-bit) or a byte (8-bit) data. Figure 1.39 shows the block diagram of the DMAC. Table 1.23 shows the DMAC specifications. Figures 1.40 to 1.42 show the registers used by the DMAC.

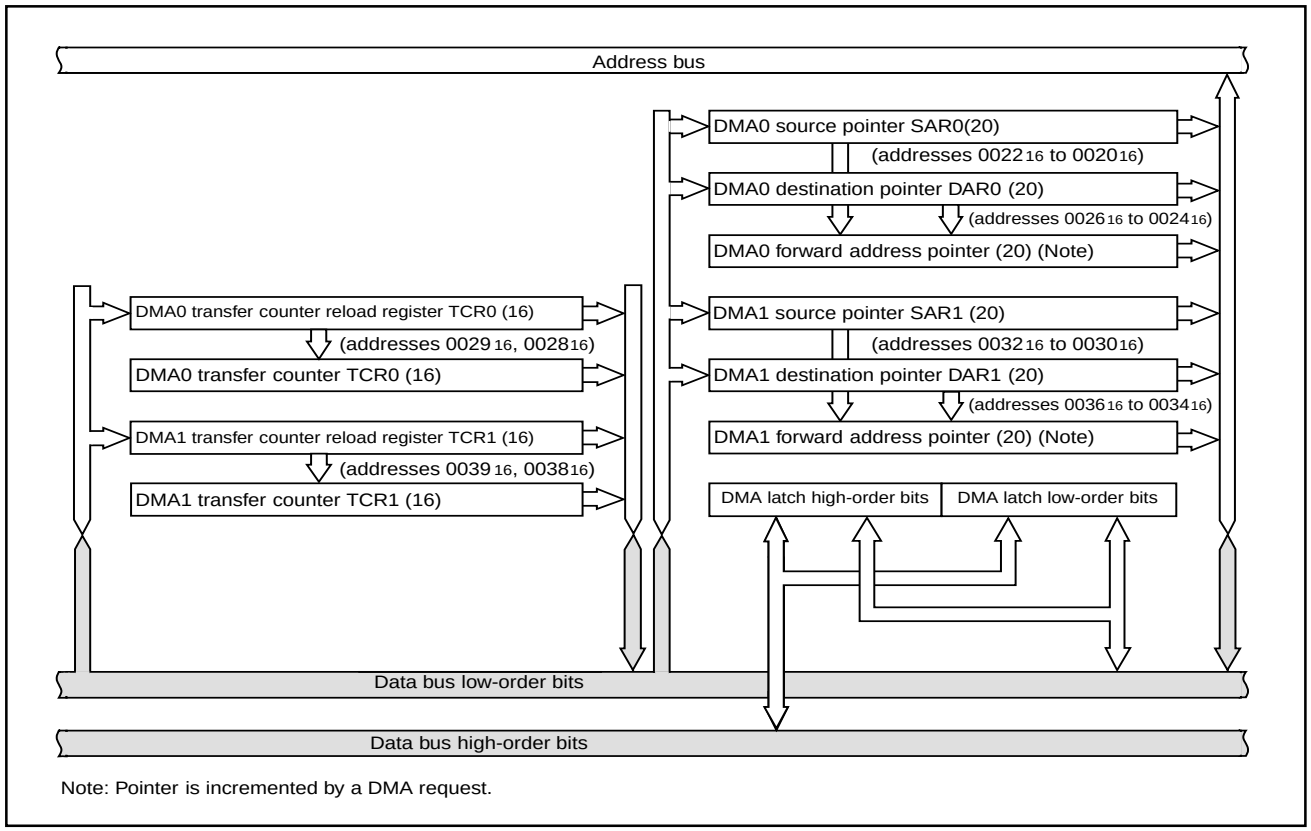



**Figure 1.39. Block diagram of DMAC**

Either a write signal to the software DMA request bit or an interrupt request signal is used as a DMA transfer request signal. The DMA transfers are not affected by the interrupt enable flag (I flag) or by the interrupt priority level and the DMA transfer doesn't affect any interrupt.

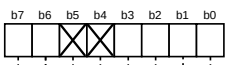
If the DMAC is active (the DMA enable bit is set to 1), data transfer starts every time a DMA transfer request signal occurs. If the cycle of the occurrences of DMA transfer request signals is higher than the DMA transfer cycle, there can be instances in which the number of transfer requests doesn't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

**Table 1.23. DMAC specifications**

| Item | Specification |
|---|---|
| Number of channels | 2 (cycle-stealing method) |
| Transfer memory space | From any address in the 1m byte space to a fixed address<br>From a fixed address to any address in the 1 M byte space<br>From a fixed address to a fixed address<br>DMA-related registers ($0020_{16}$ to $003F_{16}$) cannot be accessed |
| Maximum number of bytes transferred | 128K bytes (with 16-bit transfers) or 64K bytes (with 8-bit transfers) |
| DMA request factors (Note) | Falling edge of $\overline{INT0}$ or $\overline{INT1}$or both edges<br>($\overline{INT0}$ can be selected by DMA0, $\overline{INT1}$ by DMA1)<br>Timer A0 to Timer A4 interrupt requests<br>Timer B0 to Timer B5 interrupt requests<br>UART0 transfer and receive interrupt requests<br>UART1 transfer and receive interrupt requests<br>UART2 transfer and receive interrupt requests<br>Serial I/O 3,4 interrupt requests<br>A-D conversion interrupt requests<br>Software triggers |
| Channel priority | DMA0 has priority if DMA0 and DMA1 requests are generated simultaneously |
| Transfer unit | 8 bit or 16 bit |
| Transfer address direction | Forward/Fixed<br>(Forward direction cannot be specified for both source and destination simultaneously) |
| Transfer mode | Single transfer mode<br>After the transfer counter underflows, the DMA enable becomes 0 and the DMAC becomes inactive.<br>Repeat transfer mode<br>After the transfer counter underflows, the value of the transfer counter reload register is reloaded to the transfer counter. The DMAC remains active unless a 0 is written to the DMA enable bit. |
| DMA interrupt request generation timing | When an underflow occurs in the transfer counter |
| Active | When the DMA enable bit is set to 1 , the DMA is active.<br>When the DMA is active, data transfer starts each time the DMA transfer request signal occurs. |
| Inactive | When the DMA enable bit is set to 0 , the DMAC is inactive.<br>After the transfer counter underflows in single transfer mode. |
| Forward address pointer and reload timing for transfer counter | When the DMAC is enabled, the DMA source pointer is loaded to the DMA forward address pointer. The DMA transfer load pointer is copied to the DMA transfer counter at that time. |
| Writing to register | Registers specified for forward direction transfer are always write enabled. Registers specified for fixed address transfer are write enabled when the DMA enable bit is 0 . |
| Reading the register | Can be read anytime. However, when the DMA enable bit is 1 , reading the register set up as the forward register is the same as reading the value of the forward address pointer. |

Note: DMA transfers do not effect any interrupt and are not affected by the interrupt enable flag (I flag) or by any interrupt priority level.

DMA0 request cause select register




Symbol: DM0SL | Address: 03B8₁₆ | When reset: 00₁₆

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

Note: When the selected functions of the interrupt request are set, a DMA transfer request will occur.

**Fig. 1.40. DMAC register (1)**

DMA0 request cause select register

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

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

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

Note: When the selected functions of the interrupt request are set, a DMA transfer request will occur.

DMAi control register

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

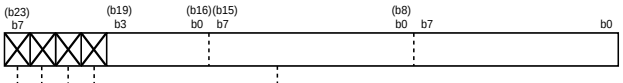
| Symbol | Address | When reset |
|---|---|---|
| DMiCON(i=0,1) | 002C16, 003C16 | XX0000002 |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DMBIT | Transfer unit bit select bit | 0 : 16 bits<br>1 : 8 bits | O | O |
| DMASL | Repeat transfer mode select bit | 0 : Single transfer<br>1 : Repeat transfer | O | O |
| DMAS | DMA request bit (Note 1) | 0 : DMA not requested<br>1 : DMA requested | O | O (Note 2) |
| DMAE | DMA enable bit | 0 : Disabled<br>1 : Enabled | O | O |
| DSD | Source address direction select bit (Note 3) | 0 : Fixed<br>1 : Forward | O | O |
| DAD | Destination address direction select bit (Note 3) | 0 : Fixed<br>1 : Forward | O | O |
| Nothing is assigned.<br>Write "0" when writing to these bits. If read, the value is "0". | | | – | – |

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

**Fig. 1.41. DMAC register (2)**

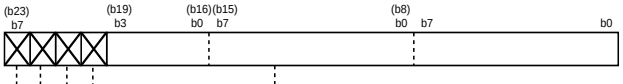
DMAi source pointer (i = 0, 1)




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

| Function | Transfer count specification | R | W |
|---|---|---|---|
| • Source pointer<br>Stores the source address | $00000_{16}$ to FFFFF$_{16}$ | O | O |
| Nothing is assigned.<br>Write "0" when writing to these bits. If read, the value is "0". | | — | — |

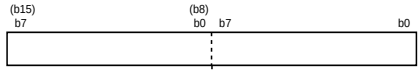
DMAi destination pointer (i = 0, 1)




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

| Function | Transfer count specification | R | W |
|---|---|---|---|
| • Destination pointer<br>Stores the destination address | $00000_{16}$ to FFFFF$_{16}$ | O | O |
| Nothing is assigned.<br>Write "0" when writing to these bits. If read, the value is "0". | | — | — |

DMAi transfer counter (i = 0, 1)




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

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

**Fig. 1.42. DMAC register (3)**

### (1) Transfer cycle

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

(a) Effect of source and destination addresses

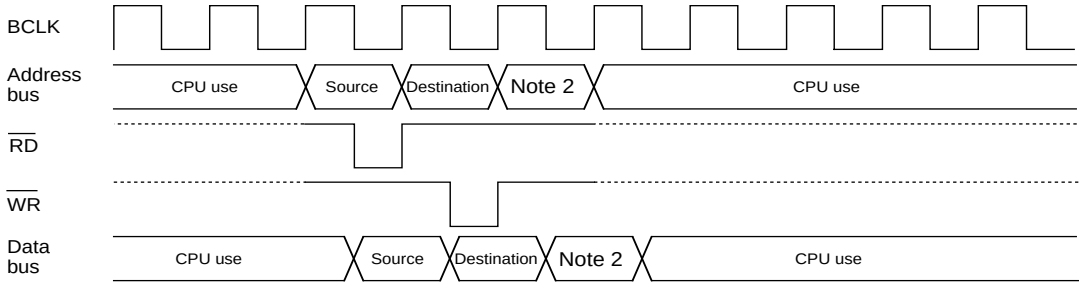
When 16-bit data is transferred on a 16-bit data bus, and the source and destination starts at odd addresses, there is one more source read cycle and destination write cycle than when the source and destination both start at even addresses.

(b) Effect of software wait

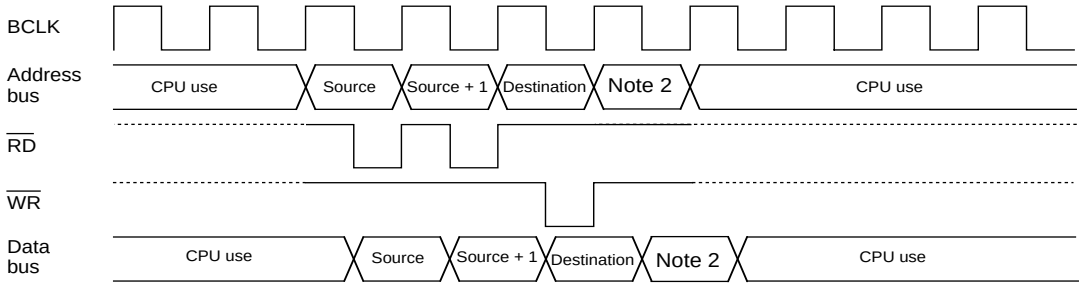
When the SFR area or a memory area with a software wait is accessed, the number of cycles is increased for the wait by 1 bus cycle. The length of the cycle is determined by BCLK.

Figure 1.43 shows the transfer cycles for a source read. For convenience, the destination write cycle is shown as one cycle and the source read cycles for the different conditions are shown. In reality, the destination write cycle is subject to the same conditions as the source read cycle, with the transfer cycle changing accordingly. When calculating the transfer cycle, remember to apply the respective conditions to both the destination write cycle and the source read cycle.

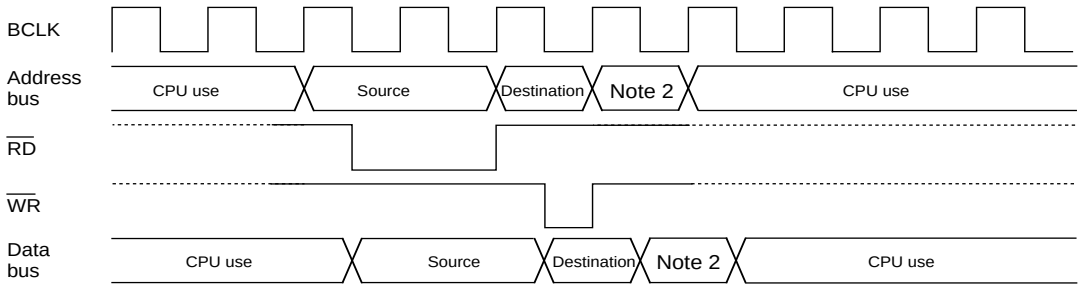
**(1) 8-bit transfers**
**16-bit transfers from even address and the source address is even.**



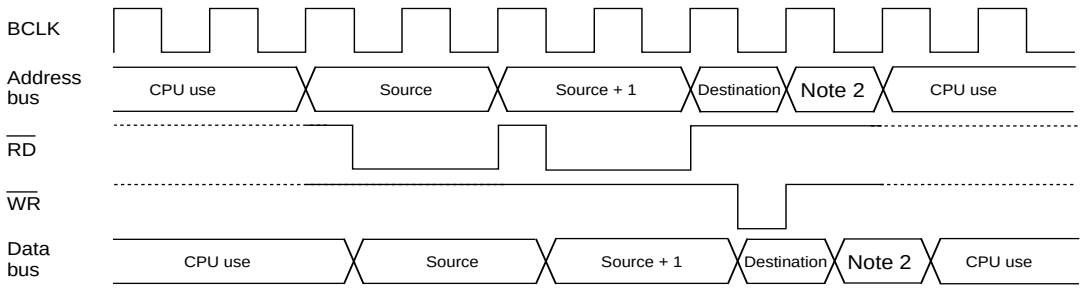

**(2) 16-bit transfers and the source address is odd**




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




**(4) One wait is inserted into the source read under the conditions in (2)**




Note 1: The same timing changes occur with the respective conditions at the destination as at the source.
Note 2: This cycle may be added depending on the instruction queue.

**Fig. 1.43. Example of the transfer cycle for a source read**

### (2) DMAC transfer cycles

Any combination of even or odd transfer read and write addresses is possible. Table 1.24 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 1.24. No. of DMAC transfer cycles**

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

**Coefficient j, k**

| Internal memory | | |
|---|---|---|
| Internal ROM/RAM No Wait | Internal ROM/RAM With Wait | SFR area |
| 1 | 2 | 2 |

### DMA enable bit

Setting the DMA enable bit to "1" makes the DMAC active. The DMAC carries out the following operations at the time data transfer starts immediately after DMAC is turned active.

(1) Reloads the value of one of the source pointer and the destination pointer - the one specified for the forward direction - to the forward direction address pointer.

(2) Reloads the value of the transfer counter reload register to the transfer counter.

Thus overwriting "1" to the DMA enable bit with the DMAC being active carries out the operations given above, so the DMAC operates again from the initial state at the instant "1" is overwritten to the DMA enable bit.

### DMA request bit

The DMA request bit is set by a DMA transfer request signal. This signal is triggered by a factor selected in advance by the DAMi Request Cause select bits.

DMA request factors include the following:

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

For the selection of DMA request factors, see the descriptions of the DMAi register.
The DMA request bit turns to "1" if the DMA transfer request signal occurs regardless of the DMAC's state (regardless of whether the DMA enable bit is set "1" or to "0"). It turns to "0" immediately before data transfer starts.
In addition, it can be set to "0" by use of a program, but cannot be set to "1".

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

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

The timing of changes in the DMA request bit is explained below.

#### (1) Internal factors

Except the DMA request factors triggered by software, the timing for the DMA request bit to turn to "1" due to an internal factor is the same as the timing for the interrupt request bit of the interrupt control register to turn to "1" due to several factors. Turning the DMA request bit to "1" due to an internal factor is timed to be effected immediately before the transfer starts.

### (2) External factors

An external factor is a factor caused to occur by the leading edge of input from the $\overline{\text{INT}}$i pin (i depends on which DMAC channel is used).

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

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

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

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

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

Figure 1.44 shows an example in which DMA transfer is carried out in minimum cycles at the time when DMA transfer request signals due to external factors concurrently occur.

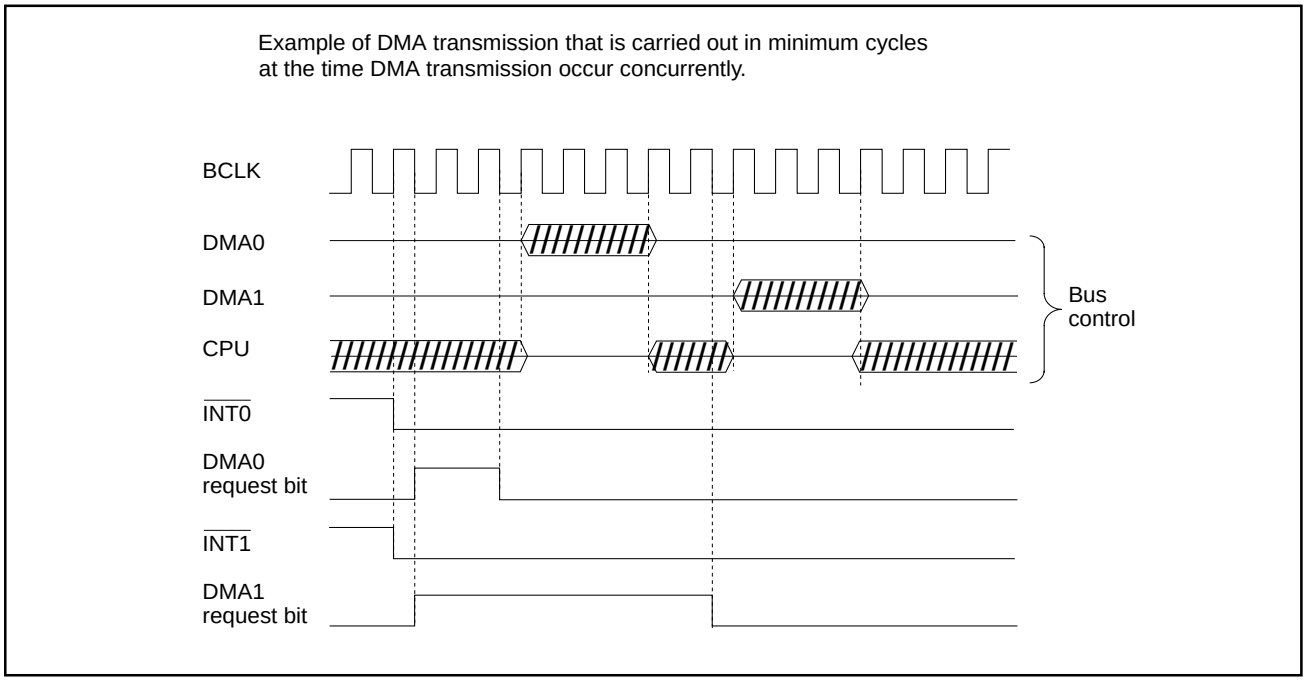
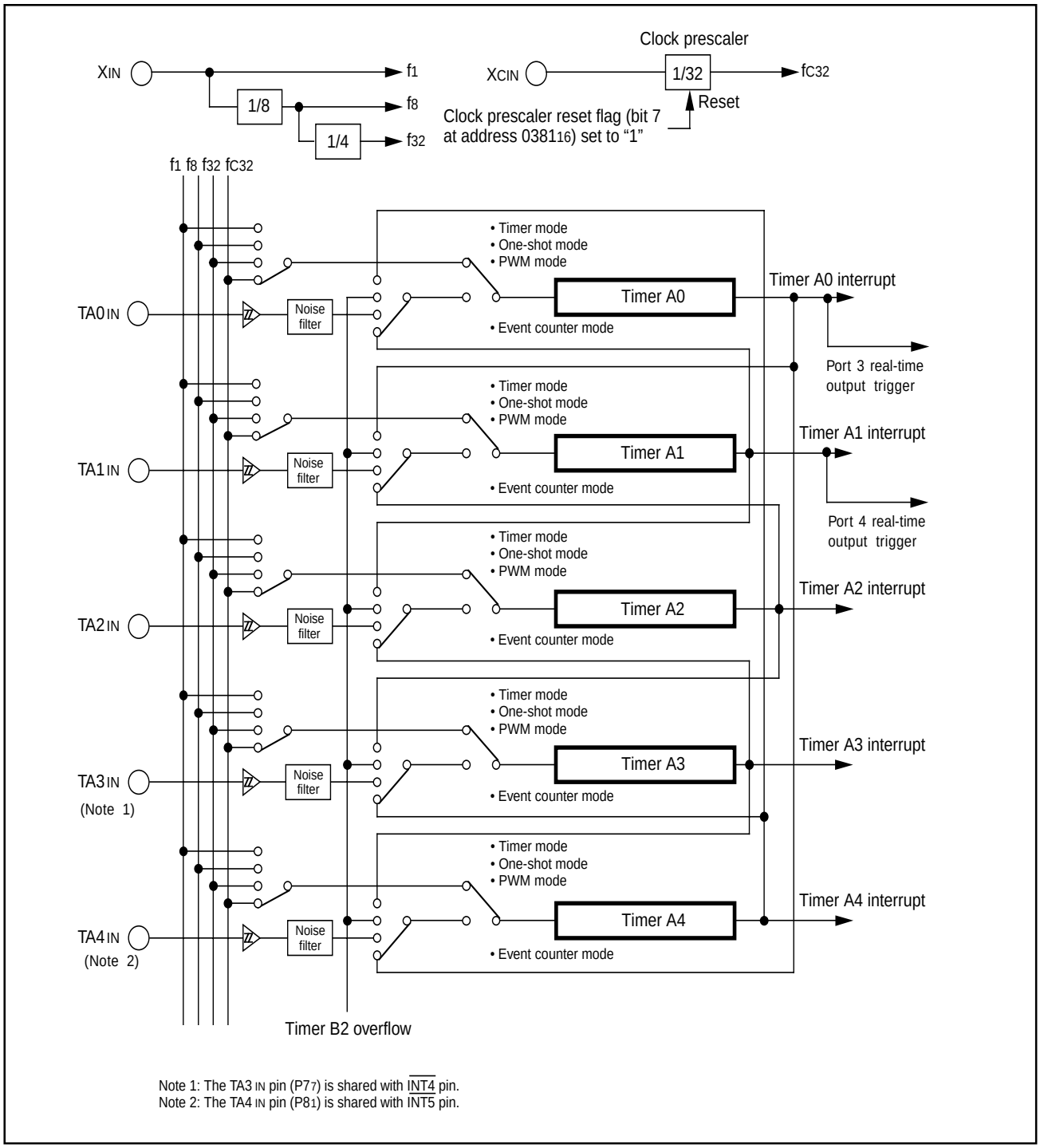



**Fig. 1.44. An example of DMA transfer affected by external factors**

## Timers

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 1.45 and 1.46 show the block diagram of timers.




Note 1: The TA3 IN pin (P77) is shared with $\overline{INT4}$ pin.
Note 2: The TA4 IN pin (P81) is shared with $\overline{INT5}$ pin.

**Figure 1.45. Timer A block diagram**

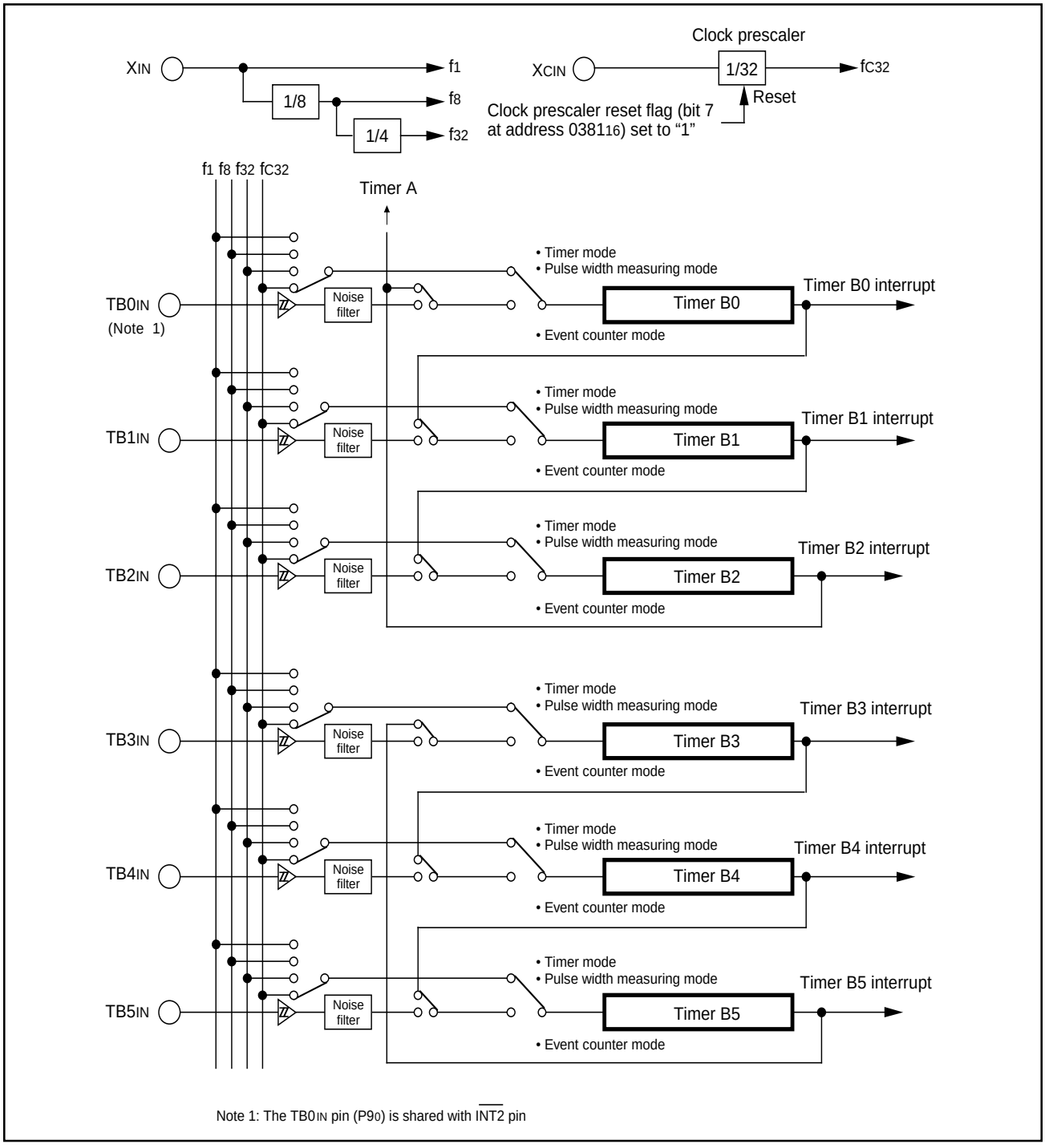



**Figure 1.46. Timer B block diagram**

## Timer A

Figure 1.47 shows the block diagram of Timer A. Figures 1.48 to 1.50 show the Timer A-related registers. Except in event counter mode, Timers A0 through A4 all have the same function. Use the Timer Ai mode register (i = 0 to 4) bits 0 and 1 to choose the desired mode.

Timer A has the four operation modes listed as follows:

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

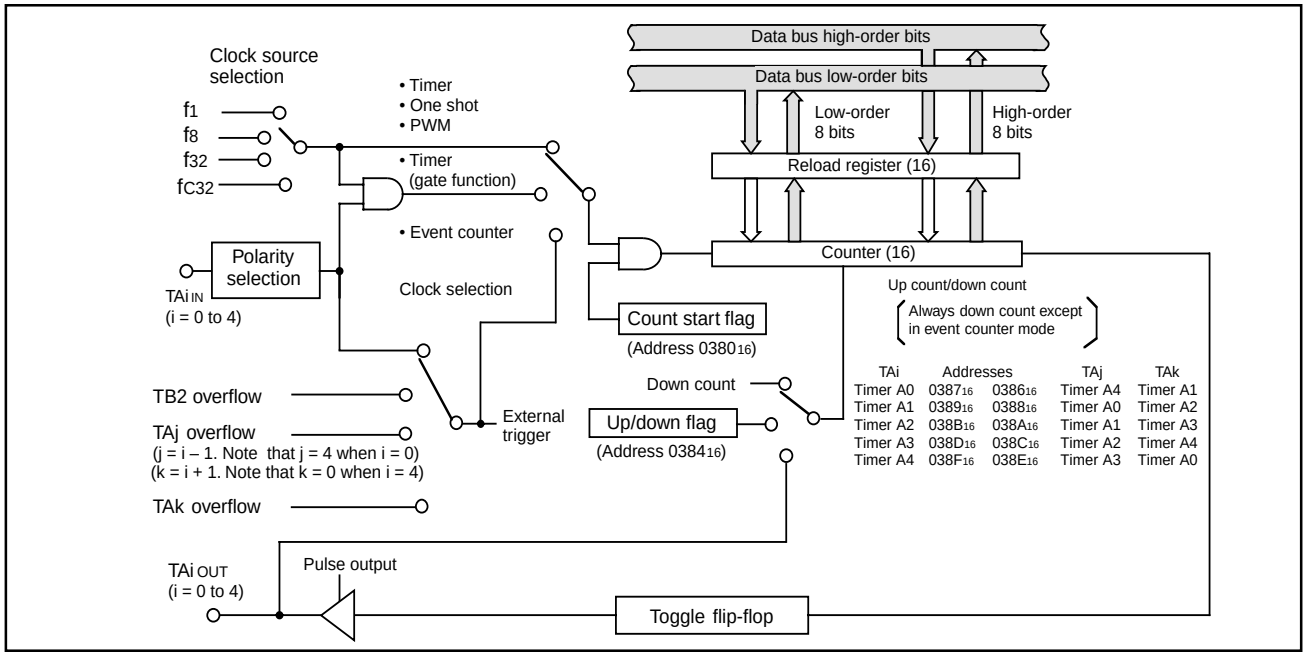



**Fig. 1.47. Block diagram of Timer A**

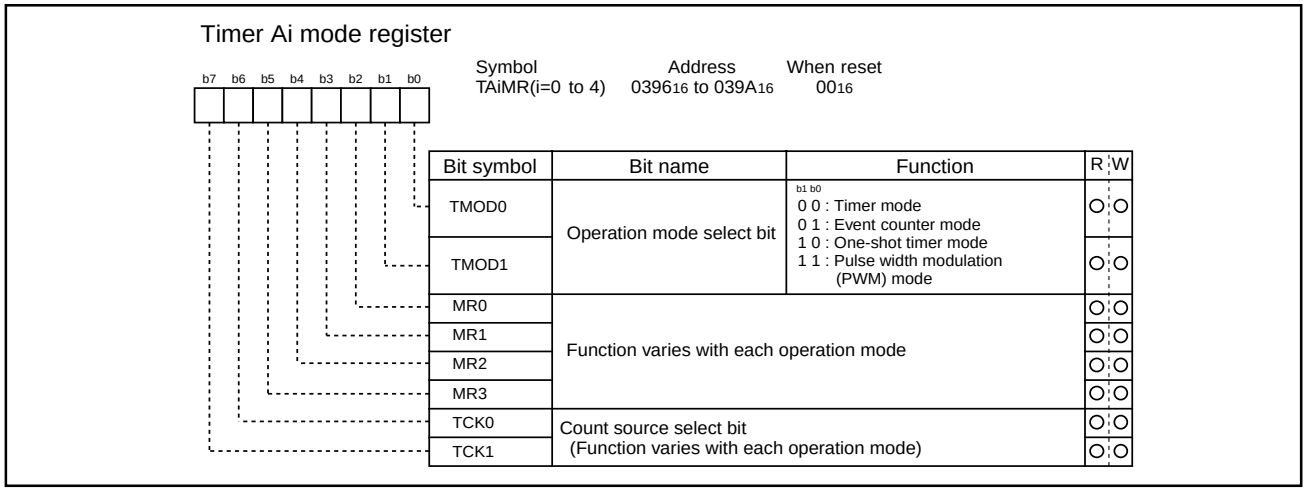
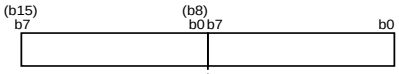



**Fig. 1.48. Timer A-related registers (1)**

Timer Ai register (Note)

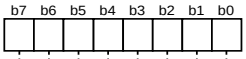



| Symbol | Address | When reset |
|---|---|---|
| TA0 | 038716,038616 | Indeterminate |
| TA1 | 038916,038816 | Indeterminate |
| TA2 | 038B16,038A16 | Indeterminate |
| TA3 | 038D16,038C16 | Indeterminate |
| TA4 | 038F16,038E16 | Indeterminate |

| Function | Values that can be set | R | W |
|---|---|---|---|
| • Timer mode<br>Counts an internal count source | 000016 to FFFF16 | ○ | ○ |
| • Event counter mode<br>Counts pulses from an external source or timer overflow | 000016 to FFFF16 | ○ | ○ |
| • One-shot timer mode<br>Counts a one shot width | 000016 to FFFF16 | × | ○ |
| • Pulse width modulation mode (16-bit PWM)<br>Functions as a 16-bit pulse width modulator | 000016 to FFFE16 | × | ○ |
| • Pulse width modulation mode (8-bit PWM)<br>Timer low-order address functions as an 8-bit prescaler and high-order address functions as an 8-bit pulse width modulator | 0016 to FE16<br>(Both high-order and low-order addresses) | × | ○ |

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

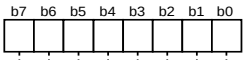
Count start flag




| Symbol | Address | When reset |
|---|---|---|
| TABSR | 038016 | 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 | | ○ | ○ |
| TA2S | Timer A2 count start flag | | ○ | ○ |
| TA3S | Timer A3 count start flag | | ○ | ○ |
| TA4S | Timer A4 count start flag | | ○ | ○ |
| TB0S | Timer B0 count start flag | | ○ | ○ |
| TB1S | Timer B1 count start flag | | ○ | ○ |
| TB2S | Timer B2 count start flag | | ○ | ○ |

Up/down flag




| Symbol | Address | When reset |
|---|---|---|
| UDF | 038416 | 0016 |

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

**Fig. 1.49. Timer A-related registers (2)**

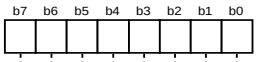
### One-shot start flag

Symbol: ONSF | Address: 038216 | When reset: 00X000002

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TA0OS | Timer A0 one-shot start flag | 1 : Timer start<br>When read, the value is "0" | O | O |
| TA1OS | Timer A1 one-shot start flag | | O | O |
| TA2OS | Timer A2 one-shot start flag | | O | O |
| TA3OS | Timer A3 one-shot start flag | | O | O |
| TA4OS | Timer A4 one-shot start flag | | O | O |
| Nothing is assigned.<br>Write "0" when writing to this bit. When read, the value is indeterminate. | | | — | — |
| TA0TGL | Timer A0 event/trigger select bit | b7 b6<br>0 0 : Input on TA0 IN is selected (Note)<br>0 1 : TB2 overflow is selected<br>1 0 : TA4 overflow is selected<br>1 1 : TA1 overflow is selected | O | O |
| TA0TGH | | | O | O |

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

### Trigger select register

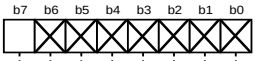



Symbol: TRGSR | Address: 038316 | When reset: 0016

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

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

### Clock prescaler reset flag




Symbol: CPSRF | Address: 038116 | When reset: 0XXXXXXX2

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Nothing is assigned.<br>Write "0" when writing to these bits. When read, the value is indeterminate. | | | — | — |
| CPSR | Clock prescaler reset flag | 0 : No effect<br>1 : Prescaler is reset<br>(When read, the value is "0") | O | O |

**Fig. 1.50. Timer A-related registers (3)**

### (1) Timer mode

In this mode, the timer counts an internally generated count source. (See Table 1.25.) Figure 1.51 shows the Timer Ai mode register in timer mode.

#### Usage Precautions

Reading the Timer Ai register while a count is in progress allows reading, with arbitrary timing, the value of the counter. Reading the Timer Ai register with the reload timing gets "FFFF16". Reading the Timer Ai register after setting a value in theTimer Ai register with a count halted but before the counter starts counting gets a proper value.

**Table 1.25. Timer mode specifications**

| Item | Specification |
|---|---|
| Count source | f1, f8, f32, fc32 |
| Count operation | Count down<br>When the timer underflows, it reloads the reload register contents before counting continues. |
| Divide ratio | 1/(n+1) n: Set value |
| Count start condition | Count start flag is set (=1) |
| Count stop condition | Count start flag is reset (=0) |
| Interrupt request generation timing | When timer underflows |
| TAiIN pin function | Programmable I/O port or gate input |
| TAiOUT pin function | Programmable I/O port or pulse output |
| Read from timer | Count value can be read out by reading Timer Ai register |
| Write to timer | When counting stops<br>When a value is written to Timer Ai register, it is written to both reload register and counter<br>When counting is in progress<br>When a value is written to Timer Ai regiser, 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 timer the timer underflows, the TAiOUT pin s polarity is reversed. |

Timer Ai mode register

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

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

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>0 0 : Timer mode | ○ | ○ |
| TMOD1 | | | ○ | ○ |
| MR0 | Pulse output function select bit | 0 : Pulse is not output (TA$_{iOUT}$ pin is a normal port pin)<br>1 : Pulse is output (Note 1) (TA$_{iOUT}$ pin is a pulse output pin) | ○ | ○ |
| MR1 | Gate function select bit | b4 b3<br>0 X (Note 2): Gate function not available (TA$_{iIN}$ pin is a normal port pin)<br>1 0 : Timer counts only when TA$_{iIN}$ pin is held "L" (Note 3) | ○ | ○ |
| MR2 | | 1 1 : Timer counts only when TA$_{iIN}$ pin is held "H" (Note 3) | ○ | ○ |
| MR3 | 0 (Must always be fixed to "0" in timer mode) | | ○ | ○ |
| TCK0 | Count source select bit | b7 b6<br>0 0 : f$_1$<br>0 1 : f$_8$<br>1 0 : f$_{32}$<br>1 1 : fC$_{32}$ | ○ | ○ |
| TCK1 | | | ○ | ○ |

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

**Fig. 1.51. 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 1.26 lists timer specifications when counting a single-phase external signal. Figure 1.52 shows the Timer Ai mode register in event counter mode. Table 1.27 lists timer specifications when counting a two-phase external signal. Figure 1.53 shows the Timer Ai mode register in event counter mode.

### Usage Precautions

(1) Reading the Timer Ai register while a count is in progress allows reading, with arbitrary timing, the value of the counter. Reading the Timer Ai register with the reload timing gets "FFFF$_{16}$" by underflow or "0000$_{16}$" by overflow. Reading the Timer Ai register after setting a value in the Timer Ai register with a count halted but before the counter starts counting gets a proper value.

(2) When stop counting in free run type, set timer again.

**Table 1.26. Timer specifications in event counter mode (when not processing two-phase pulse signal)**

| Item | Specification |
|---|---|
| Count source | External signals input to TAiIN pin (effective edge can be selected by software)<br>TB2 overflow, TAj overflow |
| Count operation | Up count or down count can be selected by external signal or software.<br>When the timer overflows or underflows, it reloads the reload register contents before counting continues (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 or pulse output, or up/down count select input |
| Read from timer | Count value can be read out by reading Timer Ai register |
| Write to timer | When counting stops<br>When a value is written to Timer Ai register, it is written to both reload register and counter<br>When counting is 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 timer the timer overflows or underflows, the TAiOUT pin s polarity is reversed. |

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

Timer Ai mode register

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

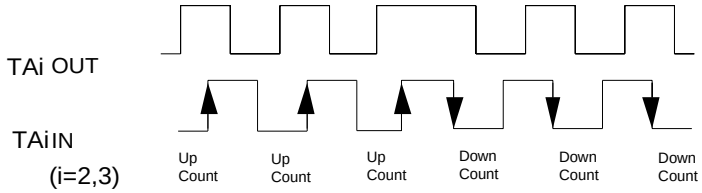
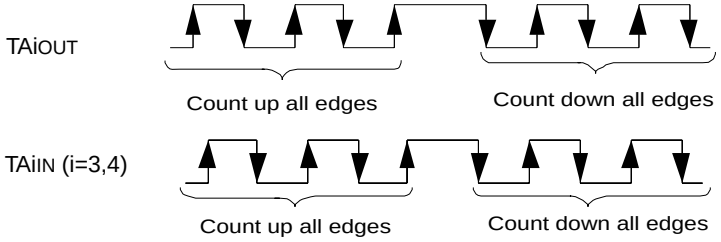
Symbol: TAiMR(i = 0, 1) Address: 039616, 039716 When reset: 0016

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

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

**Fig. 1.52. Timer Ai mode register in event counter mode**

**Table 1.27. Timer specifications in event counter mode (when processing two-phase pulse signal with Timers A2, A3,and A4)**

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

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

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

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

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

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

Note 1: The settings of the corresponding port register and port direction register are invalid.
Note 2: This bit is valid when only counting an external signal.
Note 3: Set the corresponding port direction register to "0".
Note 4: This bit is valid for the timer A3 mode register.
For timer A2 and A4 mode registers, this bit can be "0 "or "1".
Note 5: When performing two-phase pulse signal processing, make sure the two-phase pulse signal processing operation select bit (address 0384 16) is set to "1". Also, always be sure to set the event/trigger select bit (addresses 0382 16 and 038316) to "00".

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

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

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

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

Note1: This bit is valid for Timer A3 mode register.
For Timer A2 and A4 mode registers, this bit can be "0" or "1".
Note 2: When performing two-phase pulse signal processing, make sure the two-phase pulse signal processing operation select bit (address 038416) is set to "1".
Always be sure to set the event/trigger select bit (address 038216 ) to "00".

**Fig. 1.53. Timer Ai mode register in event counter mode**

### (3) One-shot timer mode

In this mode, the timer operates only once as shown in Table 1.28. When a trigger occurs, the timer starts up and continues operating for a given period. Figure 1.54 shows the Timer Ai mode register in one-shot timer mode.

**Usage Precautions**

(1) Setting the count start flag to "0" while a count is in progress causes as follows:
- The counter stops counting and a content of reload register is reloaded.
- The TAiOUT pin outputs "L" level.
- The interrupt request generated and the Timer Ai interrupt request bit goes to "1".

(2) The Timer Ai interrupt request bit goes to "1" if the timer's operation mode is set using any of the following procedures:
- Selecting one-shot timer mode after reset.
- Changing operation mode from timer mode to one-shot timer mode.
- Changing operation mode from event counter mode to one-shot timer mode.

Therefore, to useTimer Ai interrupt (interrupt request bit), set Timer Ai interrupt request bit to "0" after the above listed changes have been made.

**Table 1.28. Timer specifications in one-shot timer mode**

| Item | Specification |
|---|---|
| Count source | f1, f8, f32, fc32 |
| Count operation | 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>Timer overflows<br>One-shot start flag is set (=1) |
| Count stop condition | A new count is reloaded after the count has reached $0000_{16}$<br>The count start flag is reset (=0) |
| Interrupt request generation timing | The count reaches $0000_{16}$ |
| TAiIN pin function | Programmable I/O port or trigger input |
| TAiOUT pin function | Programmable I/O port or pulse output |
| Read from timer | When Timer Ai register is read, it indicates an indeterminate value. |
| Write to timer | When counting stops<br>When a value is written to Timer Ai register, it is written to both reload register and counter<br>When counting is in progress<br>When a value is written to Timer Ai register, it is written to only reload register (Transferred to counter at next reload time). |

Timer Ai mode register

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

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

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>1 0 : One-shot timer mode | ○ | ○ |
| TMOD1 | | | ○ | ○ |
| MR0 | Pulse output function select bit | 0 : Pulse is not output (TAiOUT pin is a normal port pin)<br>1 : Pulse is output (Note 1) (TAiOUT pin is a pulse output pin) | ○ | ○ |
| MR1 | External trigger select bit (Note 2) | 0 : Falling edge of TAiIN pin's input signal (Note 3)<br>1 : Rising edge of TAiIN pin's input signal (Note 3) | ○ | ○ |
| MR2 | Trigger select bit | 0 : One-shot start flag is valid<br>1 : Selected by event/trigger select register | ○ | ○ |
| MR3 | 0 (Must always be "0" in one-shot timer mode) | | ○ | ○ |
| TCK0 | Count source select bit | b7 b6<br>0 0 : f1<br>0 1 : f8<br>1 0 : f32<br>1 1 : fC32 | ○ | ○ |
| TCK1 | | | ○ | ○ |

Note 1: The settings of the corresponding port register and port direction register are invalid.
Note 2: Valid only when the TAiIN pin is selected by the event/trigger select bit (address 038216 and 038316). If timer overflow is selected, this bit can be "1" or "0".
Note 3: Set the corresponding port direction register to "0".

**Fig. 1.54. 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 1.29.) In this mode, the counter functions as either a 16-bit pulse width modulator or an 8-bit pulse width modulator. Figure 1.55 shows theTimer Ai mode register in pulse width modulation mode. Figure 1.56 shows the example of how a 16-bit pulse width modulator operates. Figure 1.57 shows the example of how an 8-bit pulse width modulator operates.

#### Usage Precautions

(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 useTimer 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 theTimer Ai interrupt request bit does not becomes "1".

**Table 1.29. Timer specifications in pulse width modulation mode**

| Item | Specification |
|---|---|
| Count source | f1, f8, f32, fc32 |
| Count operation | Timer counts down (operating as an 8-bit or 16-bit pulse modulator)<br>Timer reloads new count at a rising edge of PWM pulse and continues counting.<br>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 x (m + 1)/fi n: values set Timer Ai s high-order address<br>Cycle time (28-1) x (m+1)/fi m: values set Timer Ai s low-order address |
| Count start condition | External trigger is input<br>Timer overflows<br>Count start flag is set (=1) |
| Count stop condition | Count start flag is reset (=0) |
| Interrupt request generation timing | PWM pulse goes L |
| TAiIN pin function | Programmable I/O port or trigger input |
| TAiOUT pin function | Pulse output |
| Read from timer | When Timer Ai register is read, it indicates an indeterminate value. |
| Write to timer | When counting stops<br>When a value is written to Timer Ai register, it is written to both reload register and counter.<br>When counting is in progress<br>When a value is written to Timer Ai register, it is written to only reload register (Transferred to counter at next reload time). |

Timer Ai mode register

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

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

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

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

**Fig. 1.55. Timer Ai mode register in pulse width modulation mode**

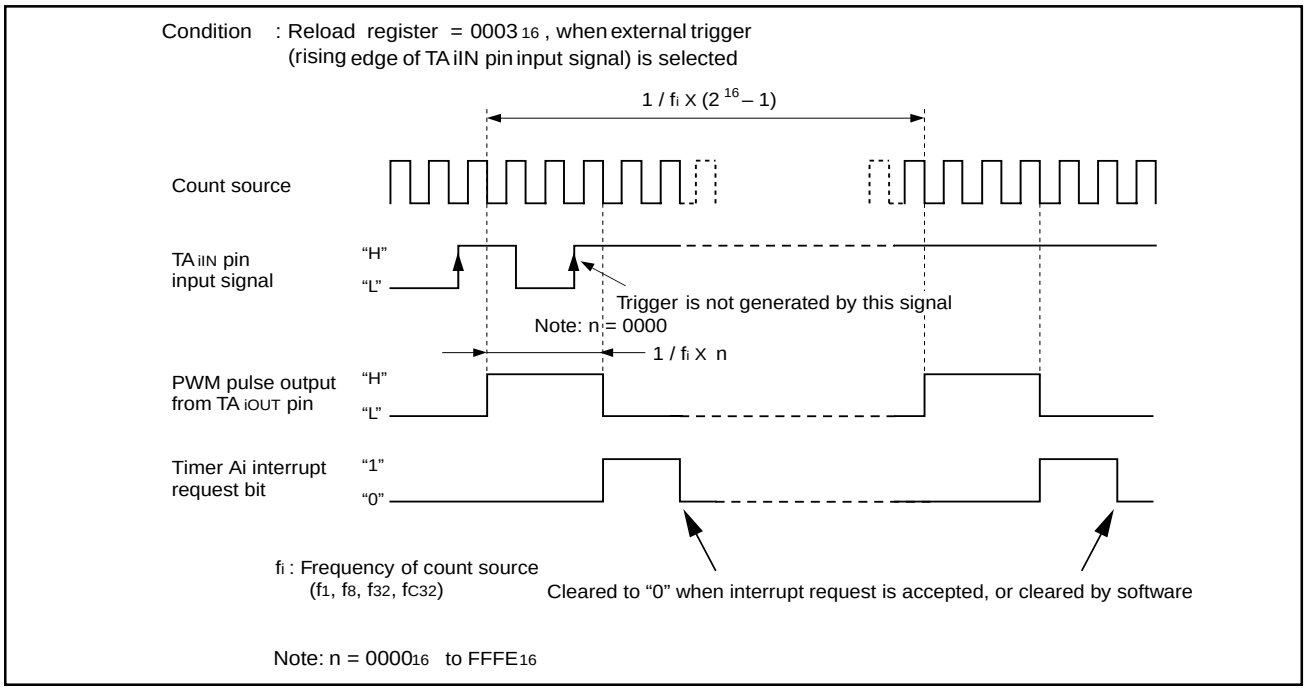



**Fig. 1.56. Example of how a 16-bit pulse width modulator operates**

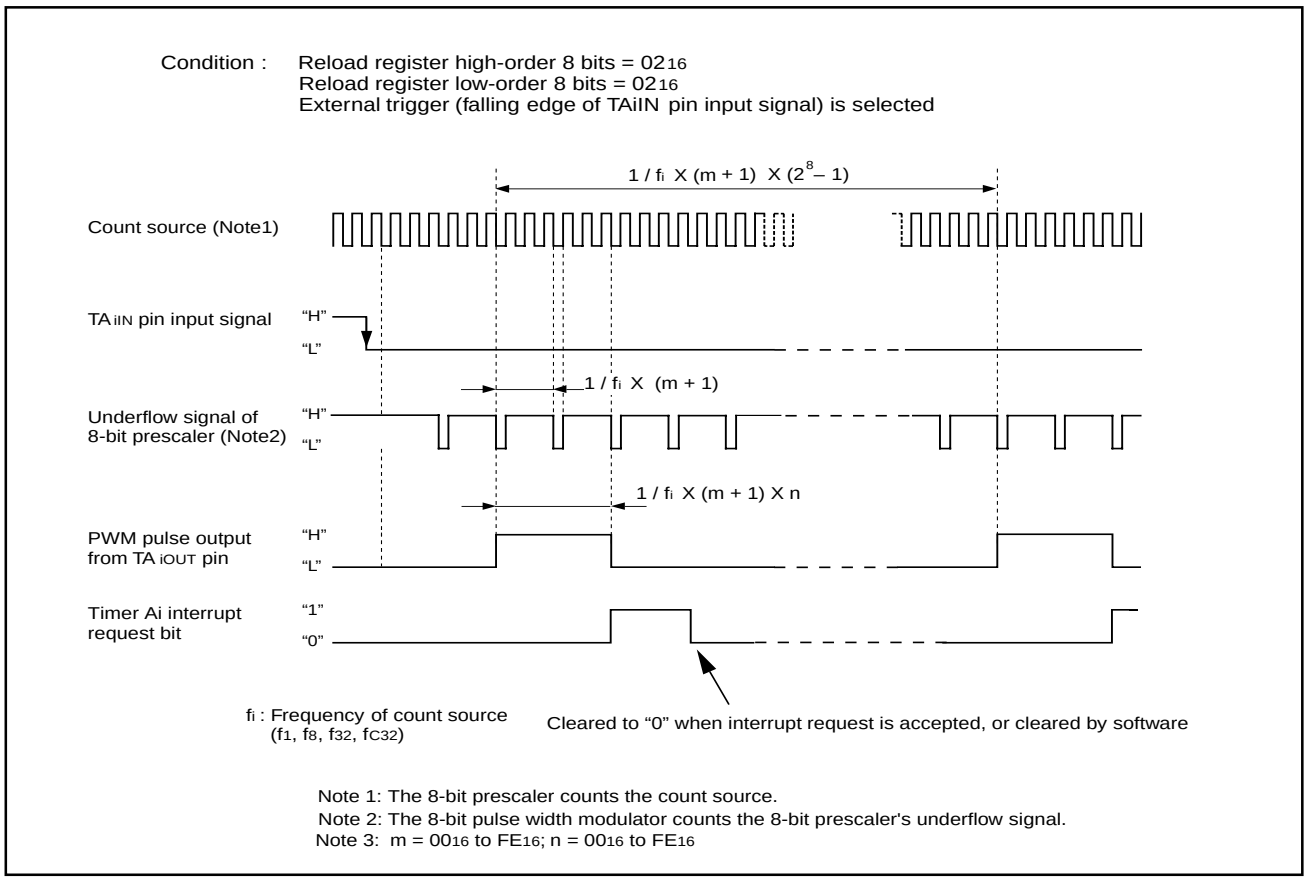



**Fig. 1.57. Example of how an 8-bit pulse width modulator operates**

### (5) Real-time port mode

When Real-time port output is selected, the data previously written to the port Pm latch is clocked into the Real-time port latch each time the corresponding Timer Ai underflows. The Real-time port data is written to the corresponding port Pm register. When the Real-time port mode select bit changes state from "0" to "1", the value of the Real-time port latch becomes "0", which is ouput from the corresponding pin. It is when Timer Ai underflows first that the Real-time port data is ouput. If the Real-time port data is modified when the Real-time port function is enabled, the modified value is output when Timer Ai underflows next time. The port functions as an ordinary port when the Real-time port function is disabled.

Make sure Timer Ai for Real-time port output is set for timer mode, and is set to have "no gate function" using the gate function select bit. Also, before setting the Real-time port mode select bit to "1", temporarily turn off Timer Ai and write its set value to the register. Figure 1.58 shows the block diagram for Real-time port output. Figure 1.59 shows the Real-time control register. Figure 1.60 shows timing in Real-time port output operation.

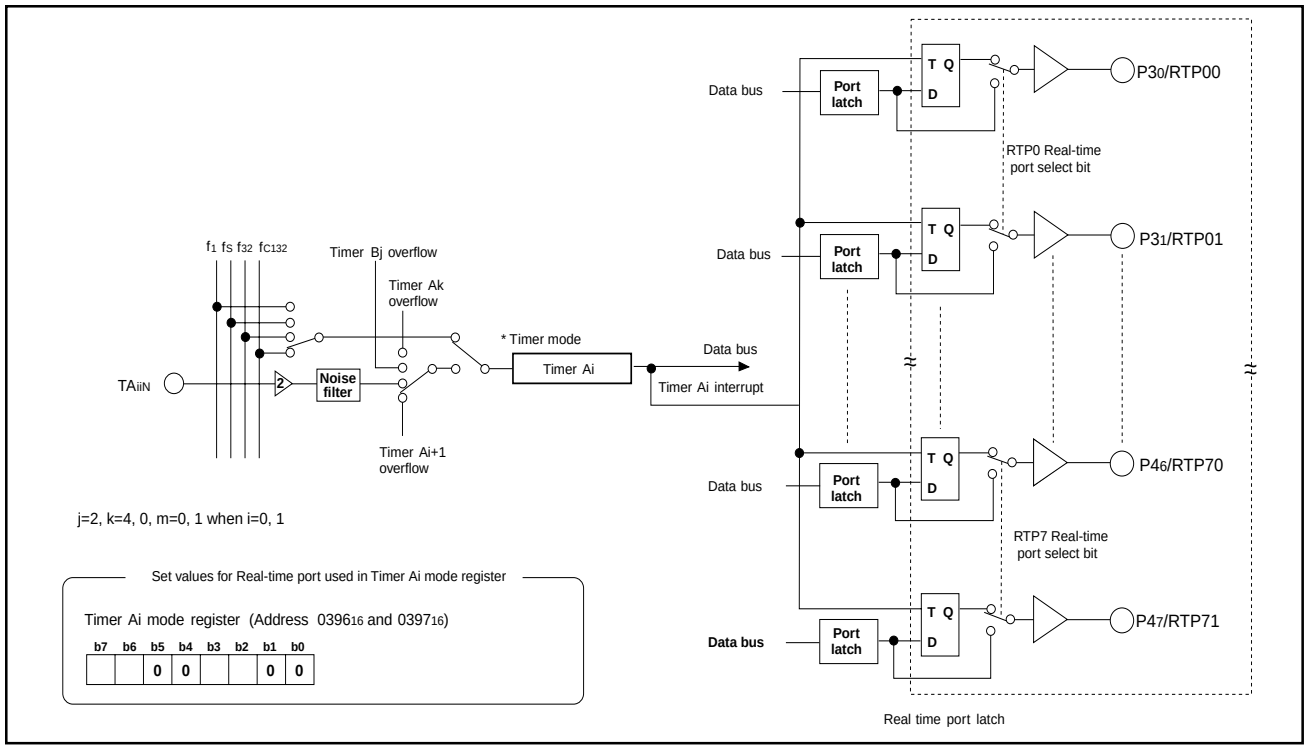



**Fig. 1.58. Block diagram of Real-time port output**

Real-time port control register (Note)

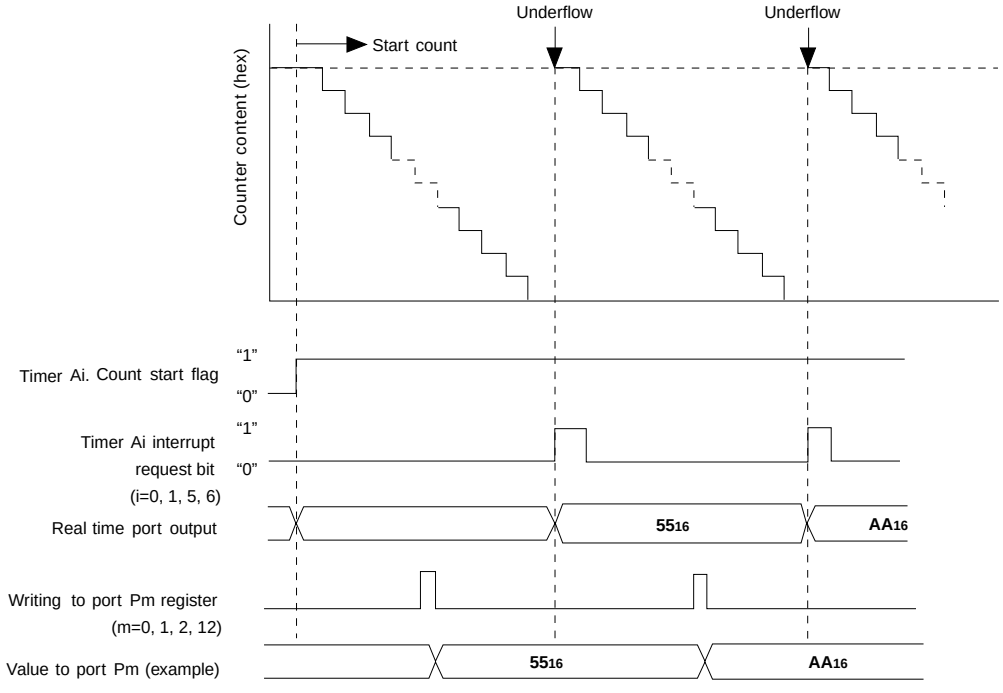
| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Symbol: RTP
Address: 03FF16
When reset: XXXX00002

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| RTP0 | P30, P31 real-time port mode select bit | The corresponding ports of output is controlled<br>0 : Ordinary port output<br>1 : Real-time port output | ○ | ○ |
| RTP1 | P32, P33 real-time port mode select bit | | ○ | ○ |
| RTP2 | P34, P35 real-time port mode select bit | | ○ | ○ |
| RTP3 | P36, P37 real-time port mode select bit | | ○ | ○ |
| RTP4 | P40, P41 real-time port mode select bit | | ○ | ○ |
| RTP5 | P42, P43 real-time port mode select bit | | ○ | ○ |
| RTP6 | P44, P45 real-time port mode select bit | | ○ | ○ |
| RTP7 | P46, P47 real-time port mode select bit | | ○ | ○ |

Note: The corresponding port direction register is invalid

**Figure 1.59. Real-time port control register**




Note : After a reset, the value of the real time port latch is "00".
The value of the real time port latch changes irrespective of the real time port mode select bit as the value of the port Pm register is updated by an underflow of the corresponding timer

**Figure 1.60. Timing in Real-time port output operation**

## Timer B

Figure 1.61shows the block diagram of Timer B. Figures 1.62 and 1.63 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 overlfow.
- •Pulse period/pulse width measuring mode: The timer measures an external signal's pulse period or pulse width.

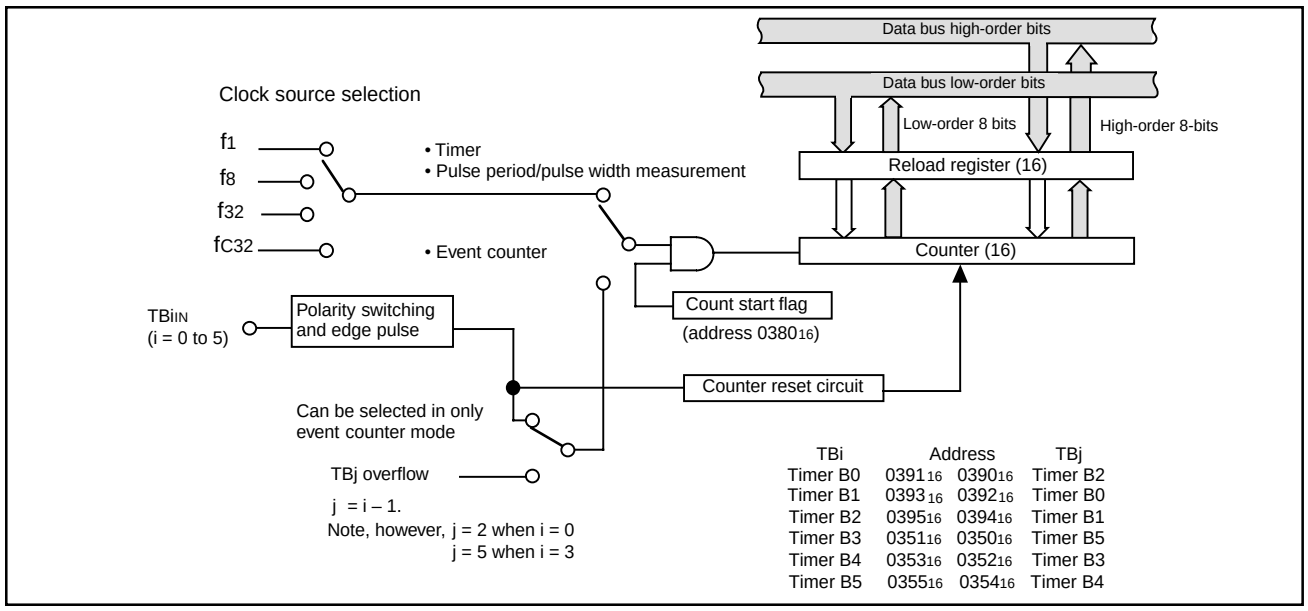



**Fig. 1.61. Block diagram of Timer B**

Timer Bi mode register

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

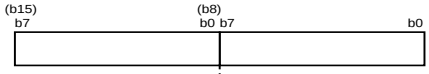
| Symbol | Address | When reset |
|---|---|---|
| TBiMR(i = 0 to 5) | 039B16 to 039D16 | 00XX00002 |
| | 035B16 to 035D16 | 00XX00002 |

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

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

**Fig. 1.62. Timer B-related registers (1)**

### Timer Bi register (Note)

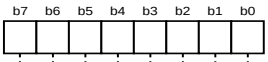



| Symbol | Address | When reset |
|---|---|---|
| TB0 | 0391$_{16}$, 0390$_{16}$ | Indeterminate |
| TB1 | 0393$_{16}$, 0392$_{16}$ | Indeterminate |
| TB2 | 0395$_{16}$, 0394$_{16}$ | Indeterminate |
| TB3 | 0351$_{16}$, 0350$_{16}$ | Indeterminate |
| TB4 | 0353$_{16}$, 0352$_{16}$ | Indeterminate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

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

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

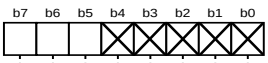
### Count start flag




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

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

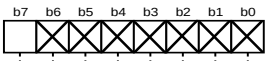
### Timer B3, 4, 5 count start flag




| Symbol | Address | When reset |
|---|---|---|
| TBSR | 0340$_{16}$ | 000XXXXX $_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Nothing is assigned.<br>Write "0" when writing to these bits. When read, the value is "0". | | | — | — |
| TB3S | Timer B3 count start flag | 0 : Stops counting<br>1 : Starts counting | ○ | ○ |
| TB4S | Timer B4 count start flag | | ○ | ○ |
| TB5S | Timer B5 count start flag | | ○ | ○ |

### Clock prescaler reset flag




| Symbol | Address | When reset |
|---|---|---|
| CPSRF | 0381$_{16}$ | 0XXXXXXX$_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Nothing is assigned.<br>Write "0" when writing to these bits. When read, the value is "0". | | | — | — |
| CPSR | Clock prescaler reset flag | 0 : No effect<br>1 : Prescaler is reset<br>(When read, the value is “0”) | ○ | ○ |

**Fig. 1.63. Timer B-related registers (2)**

### (1) Timer mode

In this mode, the timer counts an internally generated count source. (See Table 1.30).

Figure 1.64 shows the Timer Bi mode register in timer mode.

#### Usage Precaution

Reading the Timer Bi register while a count is in progress allows reading , with arbitrary timing, the value of the counter. Reading the Timer Bi register with the reload timing gets "FFFF16". Reading the Timer Bi register after setting a value in the Timer Bi register with a count halted but before the counter starts counting gets a proper value.

**Table 1.30. Timer mode specifications**

| Item | Specification |
|---|---|
| Count source | f1, f8, f32, fc32 |
| Count operation | • Count down<br>• When the timer underflows, it reloads the reload register contents before continuous counting |
| Divide ratio | 1/(n+1) n: Set value |
| Count start condition | Count start flag is set (=1) |
| Count stop condition | Count start is reset (=0) |
| Interrupt request generation timing | When the timer underflows |
| TBiIN pin function | Programmable I/O port or gate input |
| Read from timer | Count value can be read out by reading Timer Bi register |
| Write to timer | • When counting stopped<br>When a value is written to Timer Bi register, it is written to both reload register and counter<br>• When the timer underflows, it reloads the reload register contents before continuous counting<br>When a value is written to Timer Bi register,it is written to only reload register<br>(Transferred to counter at next reload time) |

Timer Bi mode register

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

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

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>0 0 : Timer mode | ○ | ○ |
| TMOD1 | | | ○ | ○ |
| MR0 | Invalid in timer mode<br>Can be “0” or “1” | | ○ | ○ |
| MR1 | | | ○ | ○ |
| MR2 | 0 (Fixed to “0” in timer mode ; i = 0, 3) | | ○ (Note 1) | ○ |
| | Nothing is assigned (i=1, 2, 4, 5)<br>Write "0" when writing to this bit. If read, the value is indeterminate. | | × (Note 2) | × |
| MR3 | Invalid in timer mode.<br>Write "0" when writing to this bit. If read in timer mode, the value is indeterminate. | | ○ | × |
| TCK0 | Count source select bit | b7 b6<br>0 0 : f1<br>0 1 : f8<br>1 0 : f32<br>1 1 : fC32 | ○ | ○ |
| TCK1 | | | ○ | ○ |

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

**Fig. 1.64. 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 1.31). Figure 1.65. shows the Timer Bi mode register in event counter mode.

#### Usage Precaution

Reading the Timer Bi register while a count is in progress allows reading , with arbitrary timing, the value of the counter. Reading the Timer Bi register with the reload timing gets “FFFF16”. Reading the Timer Bi register after setting a value in the Timer Bi register with a count halted but before the counter starts counting gets a proper value

**Table 1.31. 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 or falling edge or both as selected by software |
| Count operation | • Count down<br>• When the timer underflows, it reloads the reload register content before continous counting. |
| Divide ratio | 1/(n + 1) n: Set value |
| Count start condition | Count start flag is set (=i) |
| Count stop condition | Count start flag is reset (=0) |
| Interrupt request generation timing | The timer underflows. |
| TBiIN pin function | Count source input |
| Read from timer | Count value can be read out by reading Timer Bi register |
| Write to timer | • When counting stops<br>When a value is written to Timer Bi register, it is written to both reload register and counter.<br>• When counting is in progress<br>When a value is written to Timer Bi register, it is written to only reload register. (Transferred to counter at next reload time). |

Timer Bi mode register

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

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

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

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

**Fig. 1.65. 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 1.32). Figure 1.66 shows the Timer Bi mode register in pulse period/pulse width measurement mode. Figure 1.67 shows the operation timing when measuring a pulse period. Figure 1.68 shows the operation timing when measuring a pulse width.

#### Usage Precautions

(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.

**Table 1.32. Timer specifications in pulse period/pulse width measurement mode**

| Item | Specification |
|---|---|
| Count source | f1, f8, f32, fc32 |
| 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 . The timer Bi overlfow changes to 0 when the count start flag is 1 and the value is written to another Timer Bi timer mode register). |
| TBiIN pin function | Measurement pulse input |
| Read from timer | When Timer Bi register is read, it indicates the reload register s content (measurement result) (Note 2) |
| Write to timer | Cannot write to timer |

Note 1: An interrupt requst 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.

Timer Bi mode register

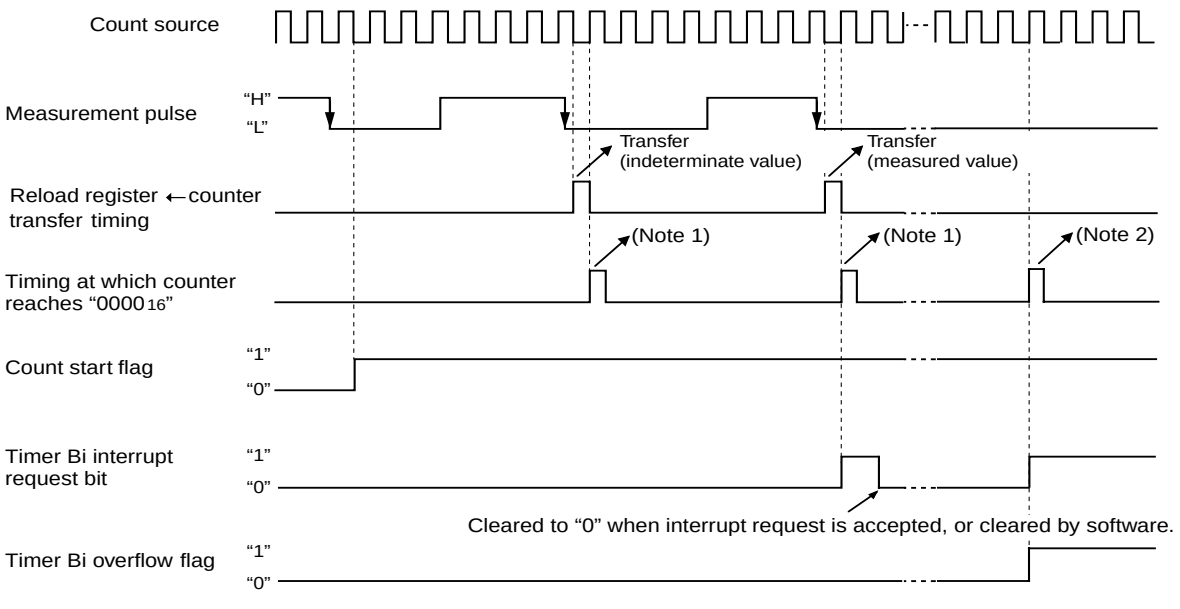
| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | 1 | 0 |

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

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

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

**Fig. 1.66. Timer Bi mode register in pulse period/pulse width measurement mode**




Note 1: Counter is initialized at completion of measurement.
Note 2: Timer has overflowed.

**Fig. 1.67. Operation timing when measuring a pulse period**

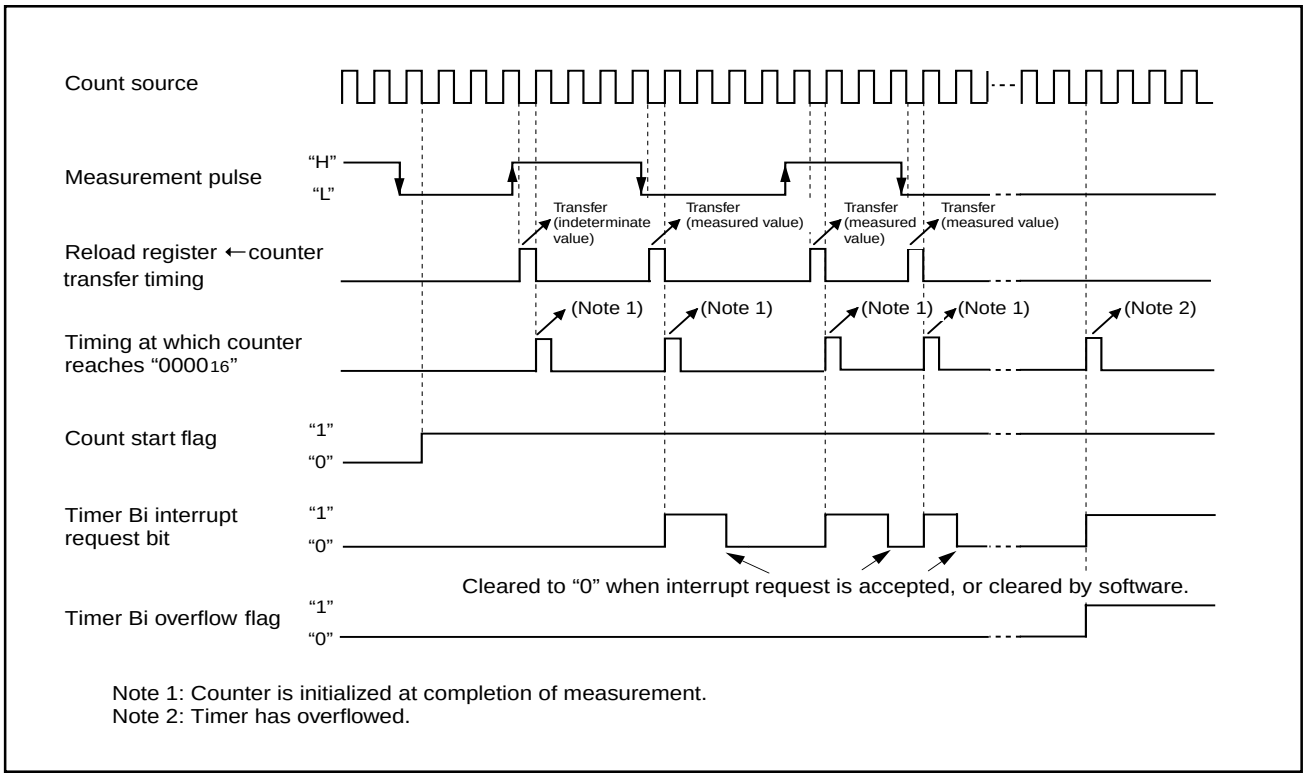



**Fig. 1.68. Operation timing when measuring a pulse width**

## Timer functions for three-phase motor control

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

Figures 1.69 to 1.71 show registers related to timers for three-phase motor control.

Three-phase PWM control register 0

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

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

| Bit symbol | Bit name | Description | R | W |
|---|---|---|---|---|
| INV00 | Effective interrupt output polarity select bit (Note 4) | 0: A Timer B2 interrupt occurs when the timer A1 reload control signal is "1".<br>1: A Timer B2 interrupt occurs when the timer A1 reload control signal is "0".<br>**Effective only in three-phase mode 1** | ○ | ○ |
| INV01 | Effective interrupt output specification bit (Note4) | 0: Not specified.<br>1: Selected by the effective interrupt output polarity selection bit.<br>**Effective only in three-phase mode 1** | ○ | ○ |
| INV02 | Mode select bit (Note 2) | 0: Normal mode<br>1: Three-phase PWM output mode | ○ | ○ |
| INV03 | Output control bit | 0: Output disabled<br>1: Output enabled | ○ | ○ |
| INV04 | Positive and negative phases concurrent L output disable function enable bit | 0: Feature disabled<br>1: Feature enabled | ○ | ○ |
| INV05 | Positive and negative phases concurrent L output detect flag | 0: Not detected yet<br>1: Already detected | ○ | ○ (Note 1) |
| INV06 | Modulation mode select bit (Note 3) | 0: Triangular wave modulation mode<br>1: Sawtooth wave modulation mode | ○ | ○ |
| INV07 | Software trigger bit | 1: Trigger generated<br>The value, when read, is "0". | ○ | ○ |

Note 1: No value other than "0" can be written.
Note 2: Selecting three-phase PWM output mode causes P8$_0$, P8$_1$, and P7$_2$ through P7$_5$ to output U, $\overline{U}$, V, $\overline{V}$, W, and $\overline{W}$, and works the timer for setting short circuit prevention time, the U, V, W phase output control circuits, and the circuit for setting Timer B2 interrupt frequency.
Note 3: **In triangular wave modulation mode:**
The short circuit prevention timer starts in synchronization with the falling edge of Timer Ai output.
The data transfer from the three-phase buffer register to the three-phase output shift register is made only once in synchronization with the transfer trigger signal after writing to the three-phase output buffer register.
**In sawtooth wave modulation mode:**
The short circuit prevention timer starts in synchronization with the falling edge of Timer A output and with the transfer trigger signal
The data transfer from the three-phase output buffer register to the three-phase output shift register is made with respect to every transfer trigger.
Note 4: To write "1" both to bit 0 (INV00) and bit 1 (INV01) of the three-phase PWM control register, set in advance the content of the Timer B2 interrupt occurrences frequency set counter.

Three-phase PWM control register 1

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

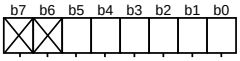
Symbol: INVC1 — Address: 0349$_{16}$ — When reset: 00$_{16}$

| Bit symbol | Bit name | Description | R | W |
|---|---|---|---|---|
| INV10 | Timer Ai start trigger signal select bit | 0: Timer B2 overflow signal<br>1: Timer B2 overflow signal, signal for writing to timer B2 | ○ | ○ |
| INV11 | Timer A1-1, A2-1, A4-1 control bit | 0: Three-phase mode 0<br>1: Three-phase mode 1 | ○ | ○ |
| INV12 | Short circuit timer count source select bit | 0 : Not to be used<br>1 : f1/2 | ○ | ○ |
| Nothing is assigned.<br>Write "0" when writing to this bit. If read, the value is "0". | | | — | — |
| Reserved bit | | Always set to "0" | ○ | ○ |
| Nothing is assigned.<br>Write "0" when writing to these bits. If read, the value is "0". | | | — | — |

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

**Fig. 1.69. Registers related to timers for three-phase motor control**

### Three-phase output buffer register 0

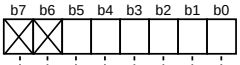



| Symbol | Address | When reset |
|---|---|---|
| IDB0 | 034A16 | 3F16 |

| Bit Symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DU0 | U phase output buffer 0 | Setting in U phase output buffer 0 | ○ | ○ |
| DUB0 | $\overline{U}$ phase output buffer 0 | Setting in $\overline{U}$ phase output buffer 0 | ○ | ○ |
| DV0 | V phase output buffer 0 | Setting in V phase output buffer 0 | ○ | ○ |
| DVB0 | $\overline{V}$ phase output buffer 0 | Setting in $\overline{V}$ phase output buffer 0 | ○ | ○ |
| DW0 | W phase output buffer 0 | Setting in W phase output buffer 0 | ○ | ○ |
| DWB0 | $\overline{W}$ phase output buffer 0 | Setting in $\overline{W}$ phase output buffer 0 | ○ | ○ |
| Nothing is assigned. Write "0" when writing to these bits. If read, the value is "0". | | | — | — |

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

### Three-phase output buffer register 1

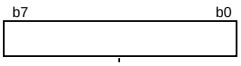



| Symbol | Address | When reset |
|---|---|---|
| IDB1 | 034B16 | 3F16 |

| Bit Symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DU1 | U phase output buffer 1 | Setting in U phase output buffer 1 | ○ | ○ |
| DUB1 | $\overline{U}$ phase output buffer 1 | Setting in $\overline{U}$ phase output buffer 1 | ○ | ○ |
| DV1 | V phase output buffer 1 | Setting in V phase output buffer 1 | ○ | ○ |
| DVB1 | $\overline{V}$ phase output buffer 1 | Setting in $\overline{V}$ phase output buffer 1 | ○ | ○ |
| DW1 | W phase output buffer 1 | Setting in W phase output buffer 1 | ○ | ○ |
| DWB1 | $\overline{W}$ phase output buffer 1 | Setting in $\overline{W}$ phase output buffer 1 | ○ | ○ |
| Nothing is assigned. Write "0" when writing to these bits. If read, the value is "0". | | | — | — |

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

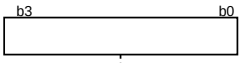
### Dead time timer




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

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

### Timer B2 interrupt occurrences frequency set counter



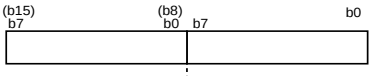

| Symbol | Address | When reset |
|---|---|---|
| ICTB2 | 034D16 | Indeterminate |

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

Note1: In setting 1 to bit 1 (INV01) - the effective interrupt output specification bit - of three-phase PWM control register 0, do not change the B2 interrupt occurrences frequency set counter to deal with the timer function for three-phase motor control.
Note 2: Do not write at the timing of an overflow occurrence in Timer B2

**Fig. 1.70. Registers related to timers for three-phase motor control**

Timer Ai register (Note)

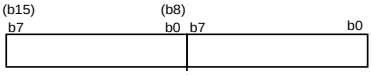



| Symbol | Address | When reset |
|---|---|---|
| TA1 | 038916,038816 | Indeterminate |
| TA2 | 038B16,038A16 | Indeterminate |
| TA4 | 038F16,038E16 | Indeterminate |
| TB2 | 039516,039416 | Indeterminate |

| Function | Values that can be set | R | W |
|---|---|---|---|
| • Timer mode<br>Counts an internal count source | 000016 to FFFF16 | ○ | ○ |
| • One-shot timer mode<br>Counts a one shot width | 000016 to FFFF16 | × | ○ |

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

Timer Ai-1 register (Note)

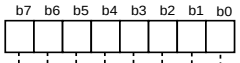



| Symbol | Address | When reset |
|---|---|---|
| TA11 | 034316,034216 | Indeterminate |
| TA21 | 034516,034416 | Indeterminate |
| TA41 | 034716,034616 | Indeterminate |

| Function | Values that can be set | R | W |
|---|---|---|---|
| Counts an internal count source | 000016 to FFFF16 | ○ | ○ |

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

Trigger select register

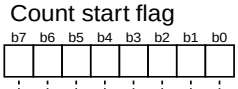



| Symbol | Address | When reset |
|---|---|---|
| TRGSR | 038316 | 0016 |

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

Note: Set the corresponding port direction register to “0”.

Count start flag




| Symbol | Address | When reset |
|---|---|---|
| TABSR | 038016 | 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 | | ○ | ○ |
| TA2S | Timer A2 count start flag | | ○ | ○ |
| TA3S | Timer A3 count start flag | | ○ | ○ |
| TA4S | Timer A4 count start flag | | ○ | ○ |
| TB0S | Timer B0 count start flag | | ○ | ○ |
| TB1S | Timer B1 count start flag | | ○ | ○ |
| TB2S | Timer B2 count start flag | | ○ | ○ |

**Fig. 1.71. Registers related to timers for three-phase motor control**

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

Setting “1” in the mode select bit (bit 2 at 0348$_{16}$) shown in Figure 1.69, causes three-phase waveform mode that uses four Timers A1, A2, A4, and B2 to be selected. As shown in Figure 1.72, set Timers A1, A2, and A4 in one-shot timer mode, set the trigger in Timer B2, and setTimer B2 in timer mode using the respective timer mode registers.

Timer Ai mode register

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

| Symbol | Address | When reset |
|---|---|---|
| TA1MR | 039716 | 0016 |
| TA2MR | 039816 | 0016 |
| TA3MR | 039A16 | 0016 |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>1 0 : One-shot timer mode | ○ | ○ |
| TMOD1 | | | ○ | ○ |
| MR0 | Pulse output function select bit | 0 (Must always be “0” in three-phase PWM output mode) | ○ | ○ |
| MR1 | External trigger select bit | Invalid in three-phase PWM output mode | ○ | ○ |
| MR2 | Trigger select bit | 1 : Selected by event/trigger select register | ○ | ○ |
| MR3 | 0 (Must always be “0” in one-shot timer mode) | | ○ | ○ |
| TCK0 | Count source select bit | b7 b6<br>0 0 : $f_1$<br>0 1 : $f_8$<br>1 0 : $f_{32}$<br>1 1 : $f_{C32}$ | ○ | ○ |
| TCK1 | | | ○ | ○ |

Timer B2 mode register

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

| Symbol | Address | When reset |
|---|---|---|
| TB2MR | 039D16 | 00XX00002 |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>0 0 : Timer mode | ○ | ○ |
| TMOD1 | | | ○ | ○ |
| MR0 | Invalid in timer mode<br>Can be “0” or “1” | | ○ | ○ |
| MR1 | | | ○ | ○ |
| MR2 | 0 (Fixed to “0” in timer mode ; i = 0) | | ○ | ○ |
| MR3 | Invalid in timer mode. | | ○ | × |
| TCK0 | Count source select bit | b7 b6<br>0 0 : $f_1$<br>0 1 : $f_8$<br>1 0 : $f_{32}$<br>1 1 : $f_{C32}$ | ○ | ○ |
| TCK1 | | | ○ | ○ |

**Fig. 1.72. Timer mode registers in three-phase waveform mode**

Figure 1.73 shows the block diagram for three-phase waveform mode. In three-phase waveform mode, the positive-phase waveforms (U phase, V phase, and W phase) and negative waveforms ($\overline{U}$ phase, $\overline{V}$ phase, and $\overline{W}$ phase), six waveforms in total, are output from $P8_0$, $P8_1$, $P7_2$, $P7_3$, $P7_4$, and $P7_5$ as active on the “L” level. Of the timers used in this mode, Timer A4 controls the U phase and $\overline{U}$ phase, timer A1 controls the V phase and $\overline{V}$ phase, and Timer A2 controls the W phase and $\overline{W}$ phase respectively; Timer B2 controls the periods of one-shot pulse output from Timers A4, A1, and A2.

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

To set the dead time, use three 8-bit timers sharing the reload register. 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 timer ($034C_{16}$), the value is written to the reload register shared by the three timers for setting dead time.

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

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

The positive waveforms (U phase, V phase, and W phase) and the negative waveforms ($\overline{U}$ phase, $\overline{V}$ phase, and $\overline{W}$ phase) in three-phase waveform mode are output from respective ports by means of setting “1” in the output control bit (bit 3 at $0348_{16}$). Setting “0” in this bit causes the ports to return to a general purpose I/O port. This bit can be set to “0” by use of the applicable instruction, entering a falling edge in the $\overline{NMI}$ terminal, or by resetting. Also, if “1” is set in the positive and negative phases concurrently, the L output disable function enable bit (bit 4 at $0348_{16}$) causes one of the pairs of U phase and $\overline{U}$ phase, V phase and $\overline{V}$ phase, and W phase and $\overline{W}$ phase to go to “L”. As a result, the port becomes the state set by the port direction register.

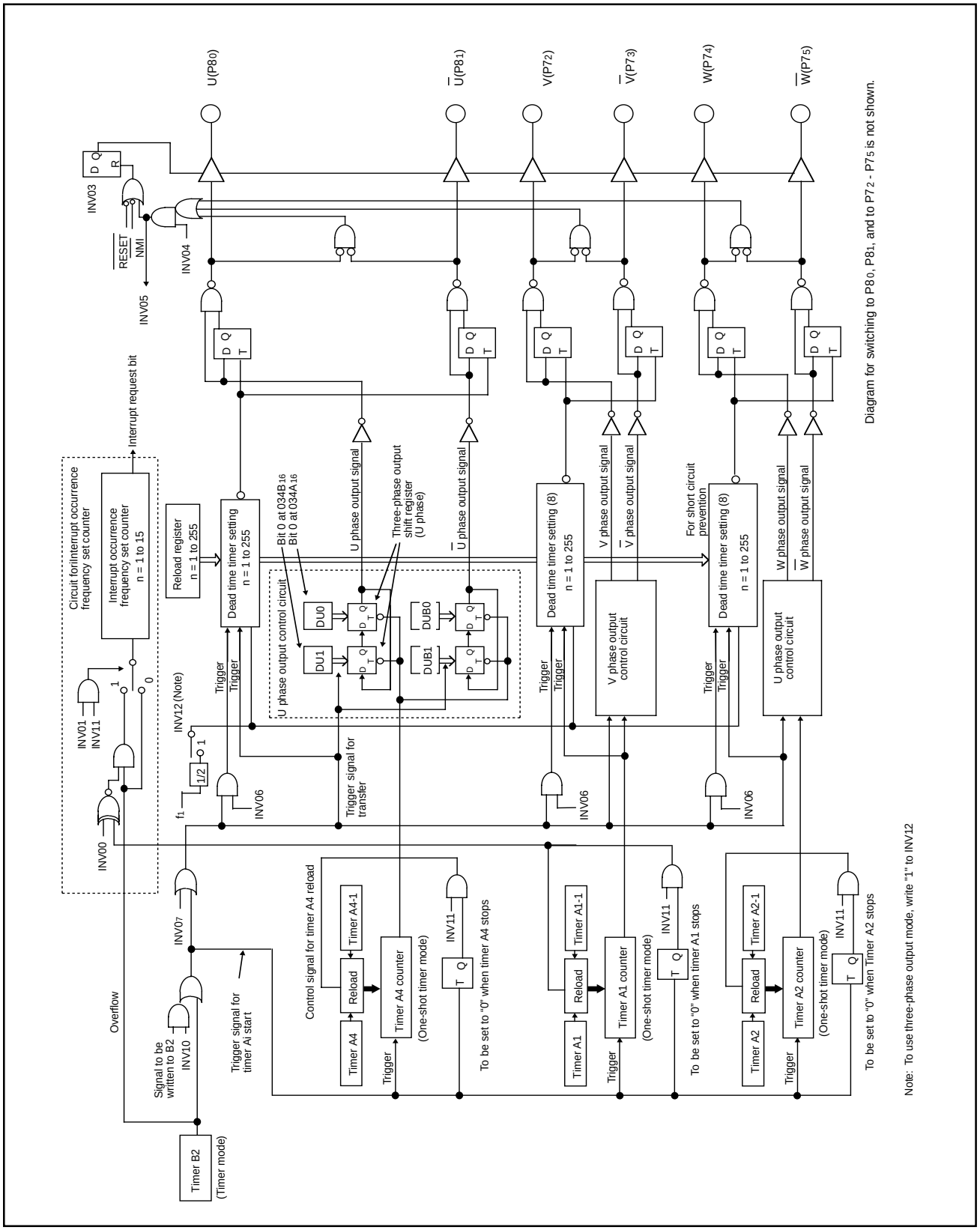



**Fig. 1.73. Block diagram for three-phase waveform mode**

### Delta modulation

To generate a PWM waveform of triangular wave modulation, set “0” in the modulation mode select bit (bit 6 at 0348$_{16}$). Also, set “1” in the Timers A4-1, A1-1, A2-1 control bit (bit 1 at 0349$_{16}$). In this mode, each of Timers A4, A1, and A2 has two timer registers, and alternately reloads the timer register’s content to the counter every time Timer B2 counter’s content becomes 0000$_{16}$. If “1” is set to the effective interrupt output specification bit (bit 1 at 0348$_{16}$), the frequency of interrupt requests that occur every time the Timer B2 counter’s value becomes 0000$_{16}$ can be set by use of the Timer B2 counter (034D$_{16}$) . The frequency of occurrences is dependent on the reload value of Timer B2. The reload value cannot be "0".

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

An example of U phase waveform is shown in Figure 1.74, and the description of waveform output workings is given below. Set “1” in DU0 (bit 0 at 034A$_{16}$). And set “0” in DUB0 (bit 1 at 034A$_{16}$). In addition, set “0” in DU1 (bit 0 at 034B$_{16}$) and set “1” in DUB1 (bit 1 at 034B$_{16}$). Also, set “0” in the effective interrupt output specification bit (bit 1 at 0348$_{16}$) to set a value in the timer B2 interrupt occurrence frequency set counter. By this setting, a Timer B2 interrupt occurs when the Timer B2 counter’s content becomes 0000$_{16}$ as many as (setting) times. Furthermore, set “1” in the effective interrupt output specification bit (bit 1 at 0348$_{16}$), set in the effective interrupt polarity select bit (bit 0 at 0348$_{16}$) and set "1" in the interrupt occurrence frequency set counter (034D16). These settings cause a Timer B2 interrupt to occur every other interval when the U phase output goes to “H”.

When the Timer B2 counter’s content becomes 0000$_{16}$, Timer A4 starts outputting one-shot pulses. In this instance, the content of DU1 (bit 0 at 034B$_{16}$) and that of DU0 (bit 0 at 034A$_{16}$) are set in the three-phase output shift register (U phase), the content of DUB1 (bit 1 at 034B$_{16}$) and that of DUB0 (bit 1 at 034A$_{16}$) are set in the three-phase shift register (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 U terminal (P8$_1$) respectively. When the Timer A4 counter counts the value written to Timer A4 (038F$_{16}$, 038E$_{16}$) and when Timer A4 finishes outputting one-shot pulses, the three-phase shift register’s content is shifted one position, and the value of DU1 and that of DUB1 are output to the U phase output signal and to U phase output signal respectively. At this time, one-shot pulses are output from the timer for setting dead time used for setting the time over which the “L” level of the U phase waveform does not lap over the “L” level of the U phase waveform, which has the opposite phase of the former. The U phase waveform output that started from the “H” level keeps its level until the timer for setting dead time finishes outputting one-shot pulses even though the three-phase output shift register’s content changes from “1” to “0” by the effect of the one-shot pulses. When the timer for setting dead time

finishes outputting one-shot pulses, "0" already shifted in the three-phase shift register goes effective, and the U phase waveform changes to the "L" level. When the Timer B2 counter's content becomes $0000_{16}$, the Timer A4 counter starts counting the value written to Timer A4-1 ($0347_{16}$, $0346_{16}$), and starts outputting one-shot pulses. When Timer A4 finishes outputting one-shot pulses, the three-phase shift register's content is shifted one position, but if the three-phase output shift register's content changes from "0" to "1" as a result of the shift, the output level changes from "L" to "H" without waiting for the timer for setting dead time to finish outputting one-shot pulses. A U phase waveform is generated by these workings repeatedly. With the exception that the three-phase output shift register on the U phase side is used, the workings in generating a U phase waveform, which has the opposite phase of the U phase waveform, are the same as in generating a U phase waveform. In this way, a waveform can be picked up from the applicable terminal in a manner in which the "L" level of the U phase waveform doesn't lap over that of the U phase waveform, which has the opposite phase of the U phase waveform. The width of the "L" level too can be adjusted by varying the values of Timer B2, Timer A4, and Timer A4-1. In dealing with the V and W phases, and V and W phases, the latter are of opposite phase of the former, have the corresponding timers work similarly to dealing with the U and U phases to generate an intended waveform.

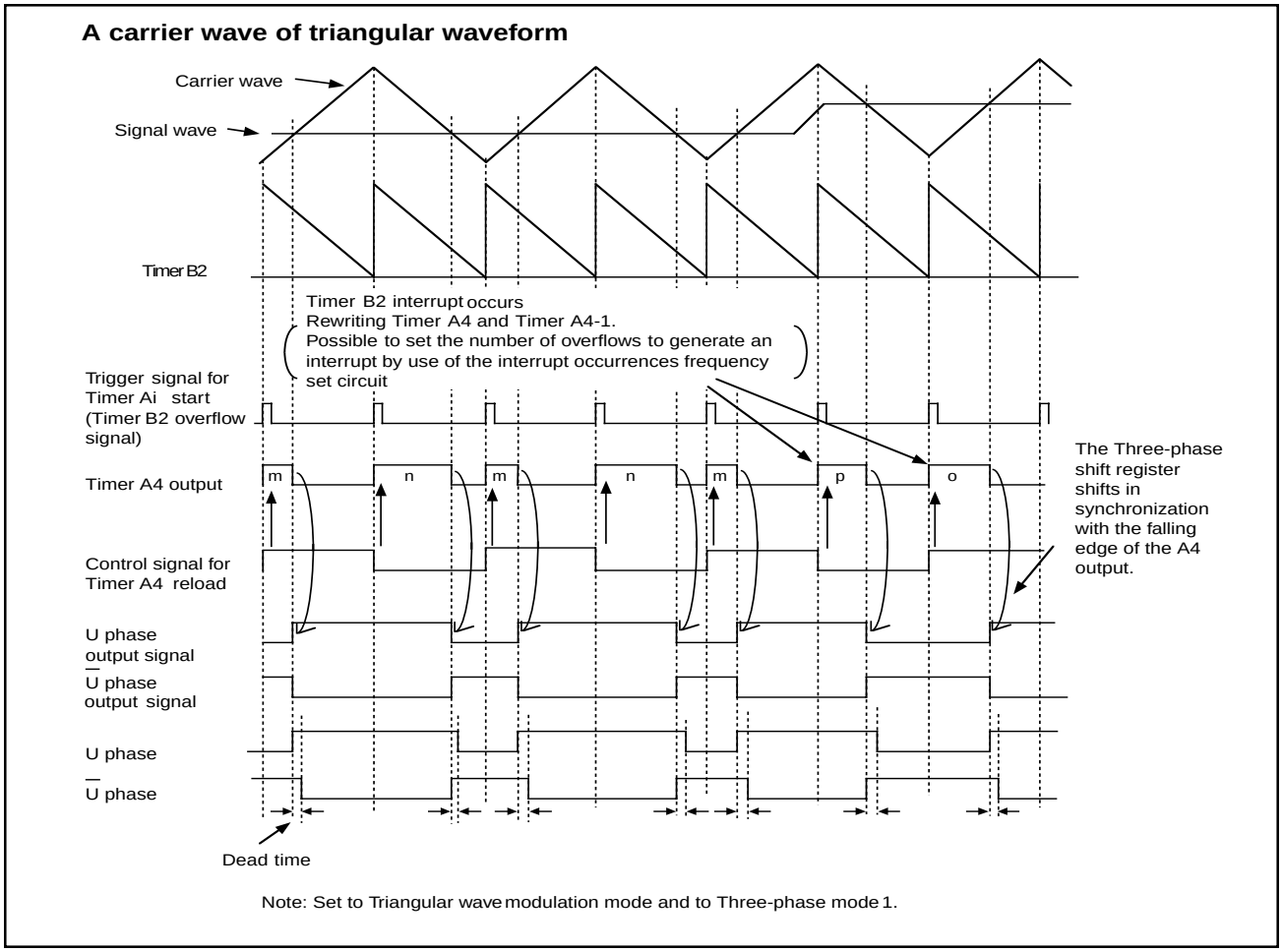



**Fig. 1.74. Timing chart operation (1)**

Assigning certain values to DU0 (bit 0 at 034A16) and DUB0 (bit 1 at 034A16), and to DU1 (bit 0 at 034B16) and DUB1 (bit 1 at 034B16) allows the user to output the waveforms as shown in Figure 1.75, that is, to output the U phase alone, to fix U phase to “H”, to fix the U phase to “H,” or to output the U phase alone.

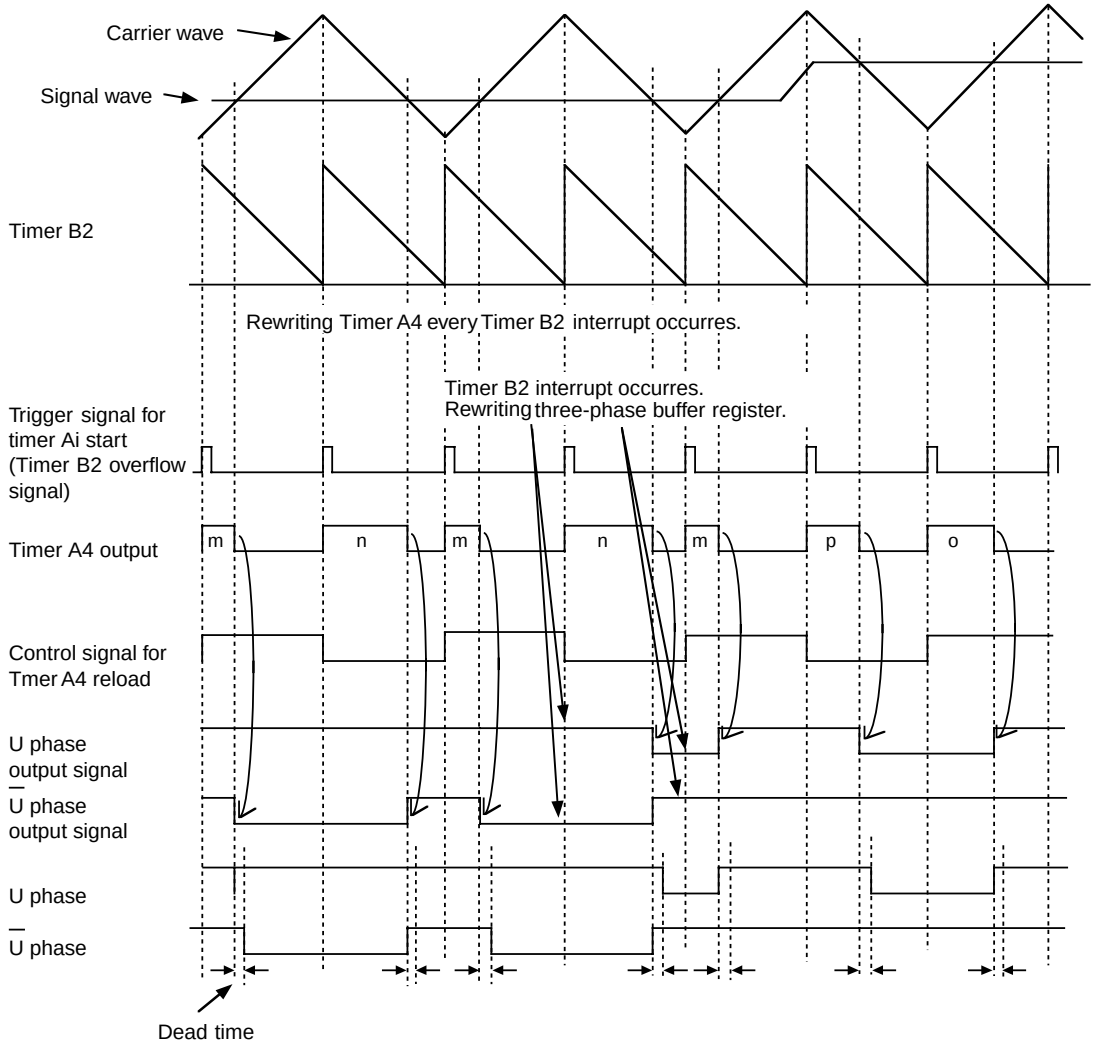



**Fig. 1.75. Timing chart of operation (2)**

### Sawtooth modulation

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

An example of U phase waveform is shown in Figure 1.76, and the description of waveform output workings is given below. Set “1” in DU0 (bit 0 at $034A_{16}$), and set “0” in DUB0 (bit 1 at $034A_{16}$). In addition, set “0” in DU1 (bit 0 at $034A_{16}$) and set “1” in DUB1 (bit 1 at $034A_{16}$).

When the Timer 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 U terminal ($P8_1$) respectively. When the timer A4 counter counts the value written to Timer A4 ($038F_{16}$, $038E_{16}$) and when Timer A4 finishes outputting one-shot pulses, the three-phase output shift register’s content is shifted one position, and the value of DU1 and that of DUB1 are output to the U phase output signal and to the 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 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 (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 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 and Timer A4. In dealing with the V and W phases, and V and W phases, the latter are of opposite phase of the former, have the corresponding timers work similarly to dealing with the U and U phases to generate an intended waveform.

Setting “1” both in DUB0 and in DUB1 provides a means to output the U phase alone and to fix the U phase output to “H” as shown in Figure 1.77.

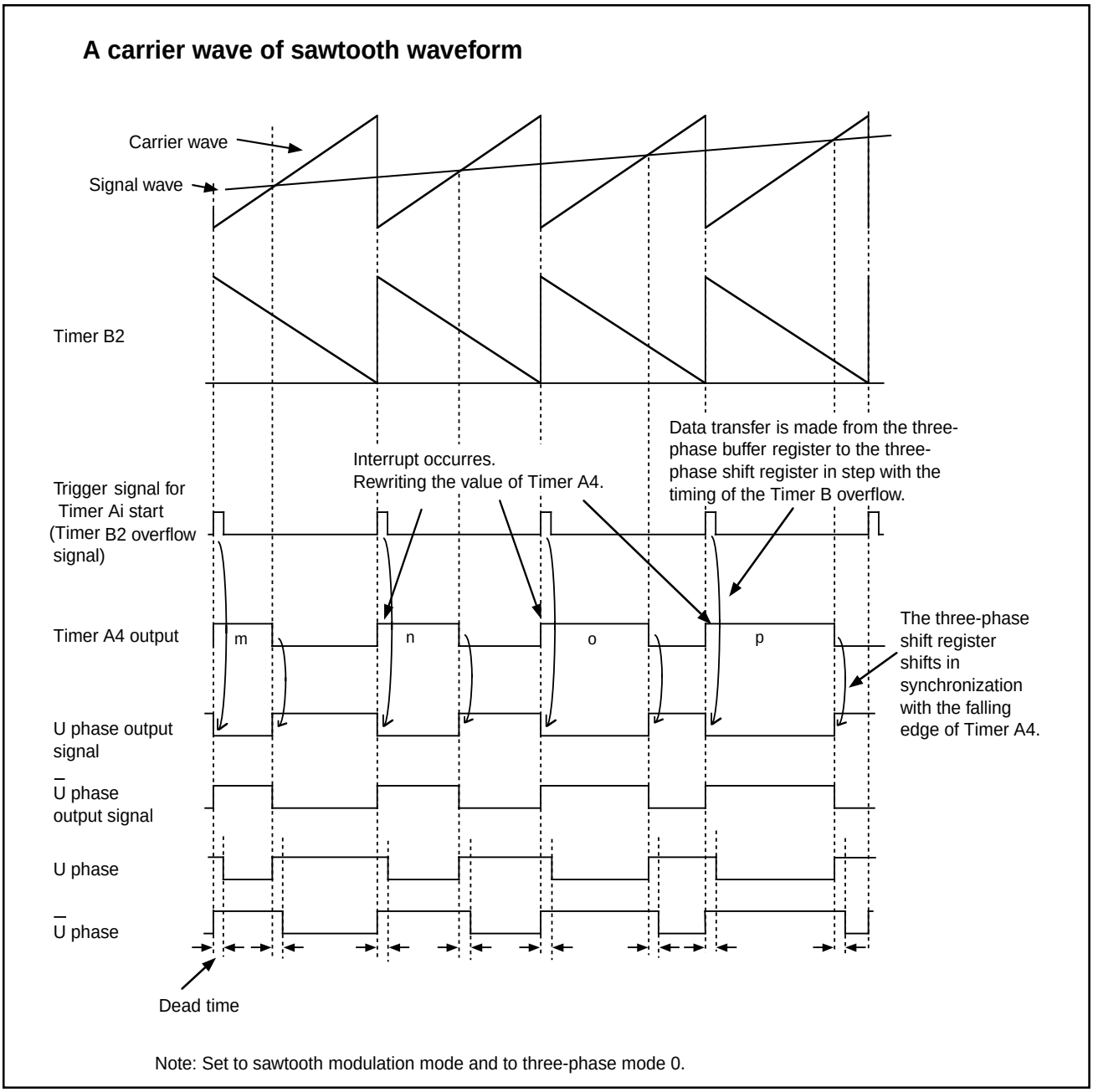
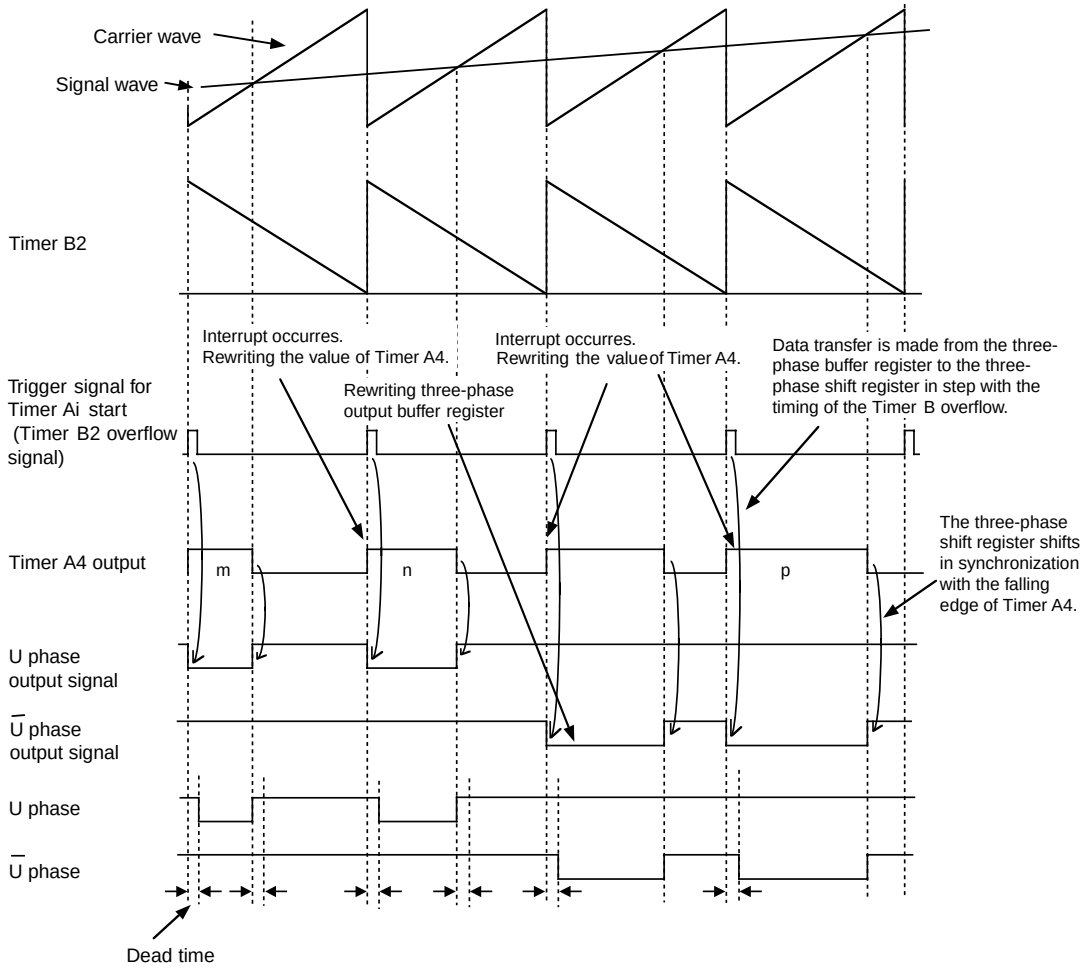



**Fig. 1.76. Timing chart of operation (3)**

**A carrier wave of sawtooth waveform**




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

**Fig. 1.77. Timing chart of operation (4)**

## Serial Communications

The five channels of serial communication that are available are: UART0, UART1, UART2, SI/O3 and SI/O4.

### UART0 to 2

UART0, UART1 and UART2 have exclusive timers to generate the transfer clock, so each operates independently from the others.

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

UART0 through UART2 are almost equal in their functions with minor exceptions. UART2 is compliant with the Subscriber Identity Module (SIM) interface with some extra settings added in clock-asynchronous serial I/O mode. It also has the bus collision detection function that generates an interrupt request if the TxD pin and the RxD pin are different in level. UART2 also provides support for both $I^2C$ and SPI transfer formats.

Figure 1.78 shows the block diagram of UART0, UART1 and UART2. Figures 1.79 and 1.80 show the block diagram of the transmit/receive unit. Figures 1.81 to 1.86 show the registers related to UARTi. Table 1.33 shows the comparison of functions of UART0 through UART2.

**Table 1.33. Comparison of functions of UART0 through UART2**

| Function | UART0 | UART1 | UART2 |
|---|---|---|---|
| CLK polarity selection | Supported (Note 1) | Supported (Note 1) | Supported (Note 1) |
| LSB first/MSB first selection | Supported (Note 1) | Supported (Note 1) | Supported (Note 2) |
| Continuous receive mode selection | Supported (Note 1) | Supported (Note 1) | Supported (Note 1) |
| Transfer clock output from multiple pins selection | Not supported | Supported (Note 1) | Not supported |
| Separate CTS/RTS pins | Supported | Not supported | Not supported |
| Serial data logic switch | Not supported | Not supported | Supported (Note 4) |
| Sleep mode selection | Supported (Note 3) | Supported (Note 3) | Not supported |
| TxD, RxD I/O polarity switch | Not supported | Not supported | Supported |
| TxD, RxD port output format | CMOS or N-channel open drain | CMOS or N-channel open drain | N-channel open drain output (Note 5) |
| Parity error signal output | Not supported | Not supported | Supported (Note 4) |
| Bus collision detection | Not supported | Not supported | Supported |
| $I^2C$ | Not supported | Not supported | Supported |
| SPI | Not supported | Not supported | Supported |

Note 1: Only in Clock Synchronous Serial I/O mode.
Note 2: Only in Clock Synchronous Serial I/O mode and 8-bit UART mode.
Note 3: Only in UART mode.
Note 4: Using SIM interface.
Note 5: Input and output when using SIM interface for UART2 only.

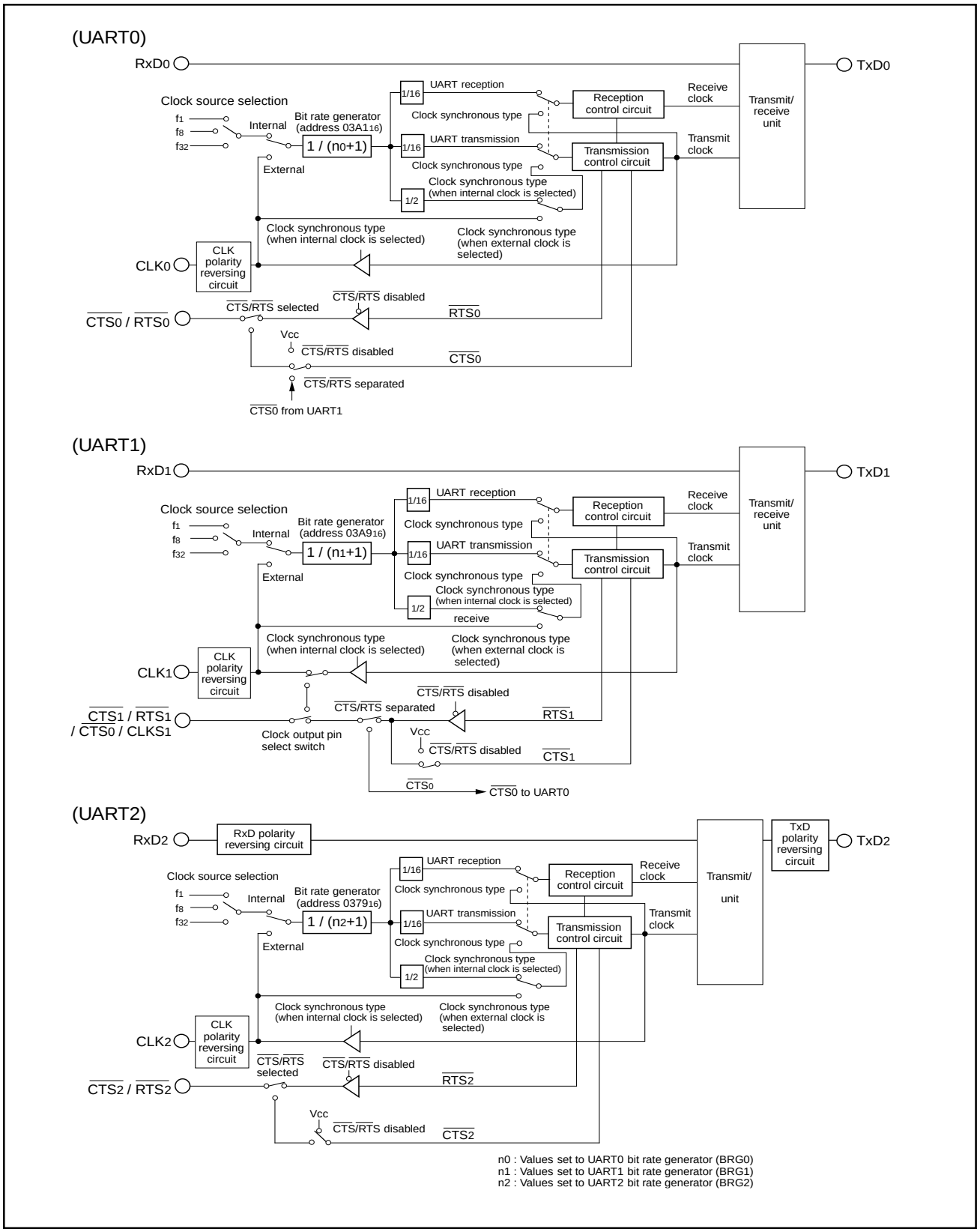



**Fig. 1.78. Block diagram of UARTi (i= 0 to 2)**

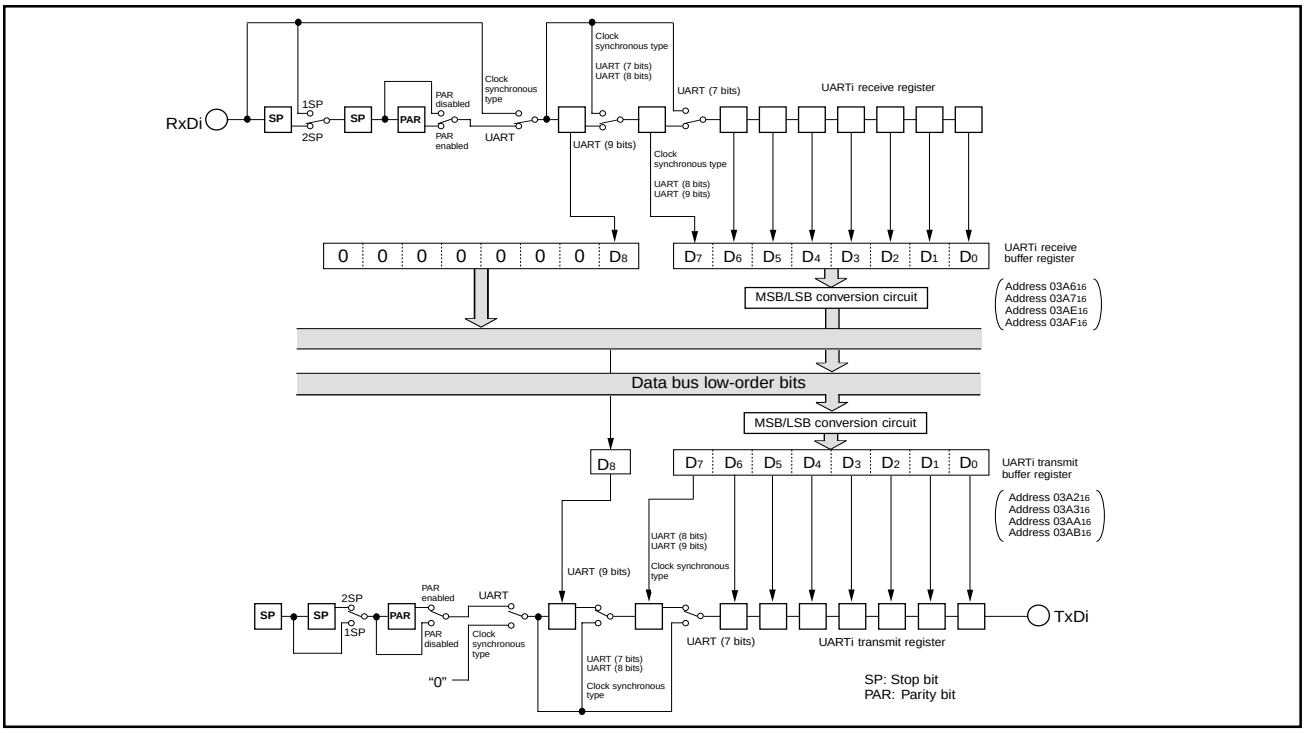



**Fig. 1.79. Block diagram of UARTi (i = 0,1) transmit/receive unit**

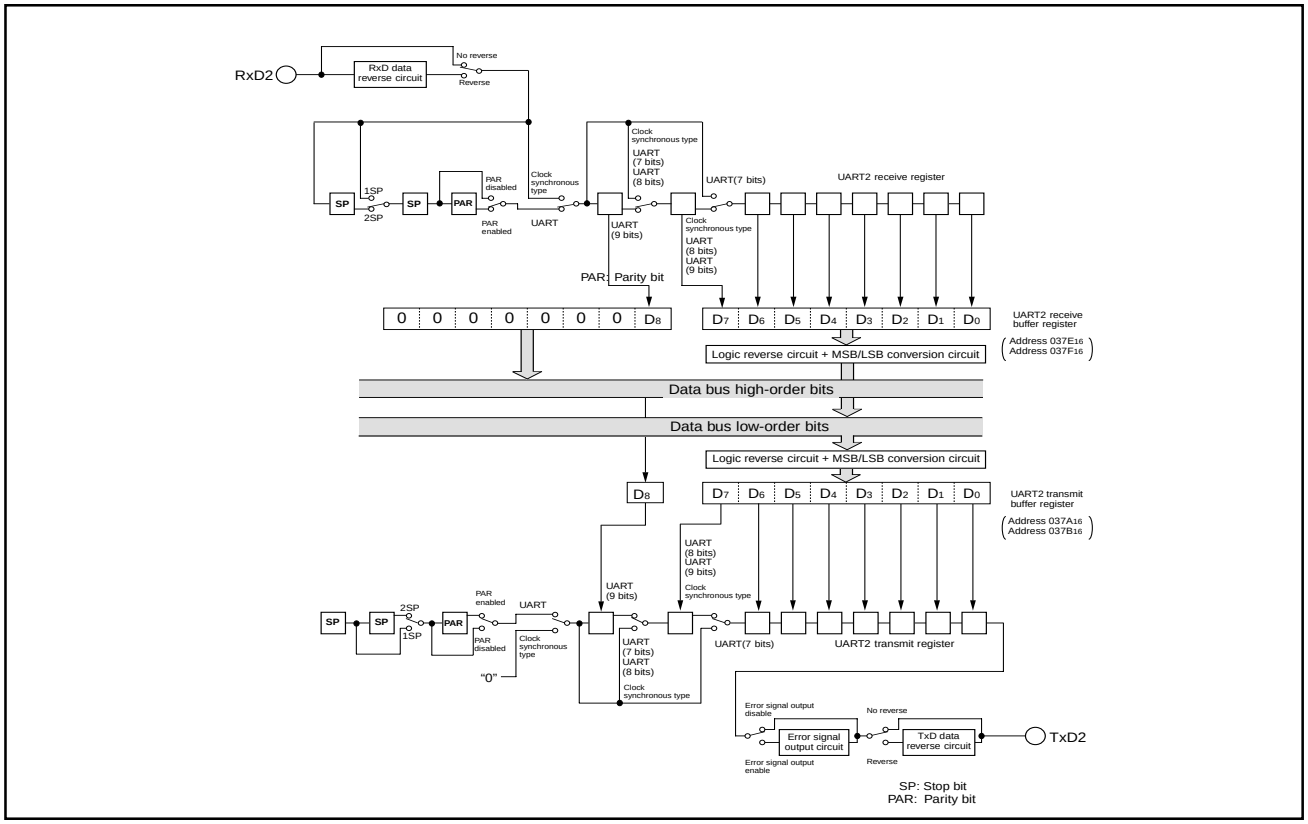
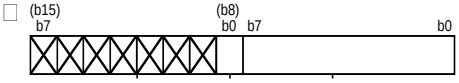



**Fig. 1.80. Block diagram of UART2 transmit/receive unit**

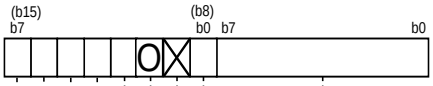
### UARTi transmit buffer register




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

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

### UARTi receive buffer register




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

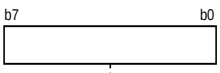
| Bit symbol | Bit name | Function (During clock synchronous serial I/O mode) | Function (During UART mode) | R | W |
|---|---|---|---|---|---|
| — | —— | Receive data | Receive data | O | X |
| Nothing is assigned.<br>Write "0" when writing to this bit. If read, the value is "0". | | | | – | – |
| MDFLT | SPI Mode fault flag (Note 3) | 0 : Not detected<br>1 : Detected | Invalid | O | O |
| ABT | Arbitration lost detecting flag (Note 2) | 0 : Not detected<br>1 : Detected | Invalid | O | O |
| OER | Overrun error flag (Note 1) | 0 : No overrun error<br>1 : Overrun error found | 0 : No overrun error<br>1 : Overrun error found | O | X |
| FER | Framing error flag (Note 1) | Invalid | 0 : No framing error<br>1 : Framing error found | O | X |
| PER | Parity error flag (Note 1) | Invalid | 0 : No parity error<br>1 : Parity error found | O | X |
| SUM | Error sum flag (Note 1) | Invalid | 0 : No error<br>1 : Error found | O | X |

Note 1: Bits 15 through 12 are set to "0" when the serial I/O mode select bit (bits 2 to 0 at addresses $03A0_{16}$ $03A8_{16}$ and $0378_{16}$) are set to "$000_2$" or the receive enable bit is set to "0".
(Bit 15 is set to "0" when bits 14 to 12 all are set to "0".) Bits 14 and 13 are also set to "0" when the lower byte of the UARTi receive buffer register (addresses $03A6_{16}$, $03AE_{16}$ and $037E_{16}$) is read out.

Note 2: Arbitration lost detecting flag is allocated to U2RB and only "0" may be written. Nothing is assigned in bit 11 of U0RB and U1RB. These bits can neither be set or reset. When read, the value of this bit is "0".

Note 3: Mode fault flag is allocated to U2RB only. Nothing is assigned in bit 10 of U0RB and U1RB. The bit is read only. After MDFLT is set, it can be reset only by exiting SPI mdoe.

### UARTi bit rate generator




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

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

**Fig. 1.81. Serial I/O related register (1)**

UARTi transmit/receive mode register

b7 b6 b5 b4 b3 b2 b1 b0

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

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

UART2 transmit/receive mode register

b7 b6 b5 b4 b3 b2 b1 b0

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

| Bit symbol | Bit name | Function (During clock synchronous serial I/O mode) | Function (During UART mode) | R | W |
|---|---|---|---|---|---|
| SMD0 | Serial I/O mode select bit | Must be fixed to 001 | b2 b1 b0<br>1 0 0 : Transfer data 7 bits long<br>1 0 1 : Transfer data 8 bits long<br>1 1 0 : Transfer data 9 bits long<br>0 0 0 : Serial I/O invalid<br>0 1 0 : Inhibited<br>0 1 1 : Inhibited<br>1 1 1 : Inhibited | O | O |
| SMD1 | | b2 b1 b0<br>0 0 0 : Serial I/O invalid<br>0 0 1 : SPI mode (Note)<br>0 1 0 : I$^2$C mode (Note)<br>0 1 1 : Inhibited<br>1 1 1 : Inhibited | | O | O |
| SMD2 | | | | O | O |
| CKDIR | Internal/external clock select bit | 0 : Internal clock<br>1 : External clock | Must always be "0" | O | O |
| STPS | Stop bit length select bit | Invalid | 0 : One stop bit<br>1 : Two stop bits | O | O |
| PRY | Odd/even parity select bit | Invalid | Valid when bit 6 = "1"<br>0 : Odd parity<br>1 : Even parity | O | O |
| PRYE | Parity enable bit | Invalid | 0 : Parity disabled<br>1 : Parity enabled | O | O |
| IOPOL | TxD, RxD I/O polarity reverse bit | 0 : No reverse<br>1 : Reverse<br>Usually set to "0" | 0 : No reverse<br>1 : Reverse<br>Usually set to "0" | O | O |

Note: Bit 2 to bit 0 are set to "010$_2$" when I$^2$C or SPI mode are used.

**Fig. 1.82. Serial I/O related registers (2)**

UARTi transmit/receive control register 0

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

| Bit symbol | Bit name | Function (During clock synchronous serial I/O mode) | Function (During UART mode) | R | W |
|---|---|---|---|---|---|
| CLK0 | BRG count source select bit | b1 b0<br>0 0 : f1 is selected<br>0 1 : f8 is selected<br>1 0 : f32 is selected<br>1 1 : Inhibited | b1 b0<br>0 0 : f1 is selected<br>0 1 : f8 is selected<br>1 0 : f32 is selected<br>1 1 : Inhibited | ○ | ○ |
| CLK1 | | | | ○ | ○ |
| CRS | $\overline{CTS}/\overline{RTS}$ function select bit | Valid when bit 4 = "0"<br>0 : $\overline{CTS}$ function is selected (Note 1)<br>1 : $\overline{RTS}$ function is selected (Note 2) | Valid when bit 4 = "0"<br>0 : $\overline{CTS}$ function is selected (Note 1)<br>1 : $\overline{RTS}$ function is selected (Note 2) | ○ | ○ |
| TXEPT | Transmit register empty flag | 0 : Data present in transmit register (during transmission)<br>1 : No data present in transmit register (transmission completed) | 0 : Data present in transmit register (during transmission)<br>1 : No data present in transmit register (transmission completed) | ○ | × |
| CRD | $\overline{CTS}/\overline{RTS}$ disable bit | 0 : $\overline{CTS}/\overline{RTS}$ function enabled<br>1 : $\overline{CTS}/\overline{RTS}$ function disabled (P60 and P64 function as programmable I/O port) | 0 : $\overline{CTS}/\overline{RTS}$ function enabled<br>1 : $\overline{CTS}/\overline{RTS}$ function disabled (P60 and P64 function as programmable I/O port) | ○ | ○ |
| NCH | Data output select bit | 0 : TXDi pin is CMOS output<br>1 : TXDi pin is N-channel open-drain output | 0: TXDi pin is CMOS output<br>1: TXDi pin is N-channel open-drain output | ○ | ○ |
| CKPOL | CLK polarity select bit | 0 : Transmit data is output at falling edge of transfer clock and receive data is input at rising edge<br>1 : Transmit data is output at rising edge of transfer clock and receive data is input at falling edge | Must always be "0" | ○ | ○ |
| UFORM | Transfer format select bit | 0 : LSB first<br>1 : MSB first | Must always be "0" | ○ | ○ |

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

UART2 transmit/receive control register 0 (Note 3)

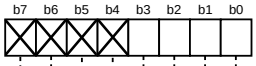
Symbol: U2C0 — Address: 037C$_{16}$ — When reset: 08$_{16}$

| Bit symbol | Bit name | Function (During clock synchronous serial I/O mode) | Function (During UART mode) | R | W |
|---|---|---|---|---|---|
| CLK0 | BRG count source select bit | b1 b0<br>0 0 : f1 is selected<br>0 1 : f8 is selected<br>1 0 : f32 is selected<br>1 1 : Inhibited | b1 b0<br>0 0 : f1 is selected<br>0 1 : f8 is selected<br>1 0 : f32 is selected<br>1 1 : Inhibited | ○ | ○ |
| CLK1 | | | | ○ | ○ |
| CRS | $\overline{CTS}/\overline{RTS}$ function select bit | Valid when bit 4 = "0"<br>0 : $\overline{CTS}$ function is selected (Note 1)<br>1 : $\overline{RTS}$ function is selected (Note 2) | Valid when bit 4 = "0"<br>0 : $\overline{CTS}$ function is selected (Note 1)<br>1 : $\overline{RTS}$ function is selected (Note 2) | ○ | ○ |
| TXEPT | Transmit register empty flag | 0 : Data present in transmit register (during transmission)<br>1 : No data present in transmit register (transmission completed) | 0 : Data present in transmit register (during transmission)<br>1 : No data present in transmit register (transmission completed) | ○ | × |
| CRD | $\overline{CTS}/\overline{RTS}$ disable bit | 0 : $\overline{CTS}/\overline{RTS}$ function enabled<br>1 : $\overline{CTS}/\overline{RTS}$ function disabled (P73 functions programmable I/O port) | 0 : $\overline{CTS}/\overline{RTS}$ function enabled<br>1 : $\overline{CTS}/\overline{RTS}$ function disabled (P73 functions programmable I/O port) | ○ | ○ |
| Nothing is assigned.<br>Write "0" when writing to this bit. If read, the value is "0". | | | | — | — |
| CKPOL | CLK polarity select bit | 0 : Transmit data is output at falling edge of transfer clock and receive data is input at rising edge<br>For SPI, clock is high between transfers<br>1 : Transmit data is output at rising edge of transfer clock and receive data is input at falling edge<br>For SPI, clock is low between transfers | Must always be "0" | ○ | ○ |
| UFORM | Transfer format select bit | 0 : LSB first<br>1 : MSB first | 0 : LSB first<br>1 : MSB first | ○ | ○ |

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

**Fig. 1.83. Serial I/O-related registers (3)**

UARTi transmit/receive control register 1




Symbol: UiC1(i=0,1) | Address: $03A5_{16}$,$03AD_{16}$ | When reset: $02_{16}$

| Bit symbol | Bit name | Function (During clock synchronous serial I/O mode) | Function (During UART mode) | R | W |
|---|---|---|---|---|---|
| TE | Transmit enable bit | 0 : Transmission disabled<br>1 : Transmission enabled | 0 : Transmission disabled<br>1 : Transmission enabled | ○ | ○ |
| TI | Transmit buffer empty flag | 0 : Data present in transmit buffer register<br>1 : No data present in transmit buffer register | 0 : Data present in transmit buffer register<br>1 : No data present in transmit buffer register | ○ | × |
| RE | Receive enable bit | 0 : Reception disabled<br>1 : Reception enabled | 0 : Reception disabled<br>1 : Reception enabled | ○ | ○ |
| RI | Receive complete flag | 0 : No data present in receive buffer register<br>1 : Data present in receive buffer register | 0 : No data present in receive buffer register<br>1 : Data present in receive buffer register | ○ | × |
| Nothing is assigned. Write "0" when writing to these bits. If read, the value is "0". | | | | — | — |

UART2 transmit/receive control register 1

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

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

**Fig. 1.84. Serial I/O-related registers (4)**

UART transmit/receive control register 2

b7 b6 b5 b4 b3 b2 b1 b0

Symbol: UCON | Address: 03B0$_{16}$ | When reset: X0000000$_2$

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

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

UART2 special mode register

b7 b6 b5 b4 b3 b2 b1 b0

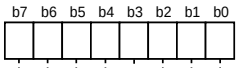
Symbol: U2SMR | Address: 037716 | When reset: 00$_{16}$

| Bit symbol | Bit name | Function (During clock synchronous serial I/O mode) | Function (During UART mode) | R | W |
|---|---|---|---|---|---|
| IICM | I$^2$C mode selection bit | 0 : Normal mode<br>1 : I$^2$C mode | Must always be "0" | ○ | ○ |
| ABC | Arbitration lost detecting flag control bit | 0 : Update per bit<br>1 : Update per byte | Must always be "0" | ○ | ○ |
| BBS | Bus busy flag | 0 : STOP condition detected<br>1 : START condition detected | Must always be "0" | ○ | ○ (Note 1) |
| LSYN | SCLL sync output enable bit | 0 : Disabled<br>1 : Enabled | Must always be "0" | ○ | ○ |
| ABSCS | Bus collision detect sampling clock select bit | Must always be "0" | 0 : Rising edge of transfer clock<br>1 : Underflow signal of timer A0 | ○ | ○ |
| ACSE | Auto clear function select bit of transmit enable bit | Must always be "0" | 0 : No auto clear function<br>1 : Auto clear at occurrence of bus collision | ○ | ○ |
| SSS | Transmit start condition select bit | Must always be "0" | 0 : Ordinary<br>1 : Falling edge of RxD2 | ○ | ○ |
| SDDS | SDA digital delay select bit (Note 2, 3) | 0 : Analog delay output is selected<br>1 : Digital delay output is selected (Must always be "0" when not using I$^2$C mode) | Must always be "0" | ○ | ○ |

Note 1: Nothing but "0" may be written.
Note 2: When not in I$^2$C mode, do not set this bit by writing a "1". during normal mode, fix it to "0". When this bit - "0" , UART2 special mode register 3 (U2SMR3 at address 037516) bits 7 to 5 (DL2 to DL0 = SDA digital delay setup bits) are initialized to "000", with the analog delay circuit selected. Also, when SDDS = "0", do not read or write to U2SMR3 register.
Note 3: When analog delay is selected, only the analog delay value is effective; when digital delay is selected only the digital delay value is effective.

**Fig. 1.85. Serial I/O-related registers (5)**

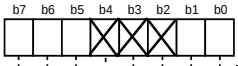
UART2 special mode register 2




Symbol U2SMR2 | Address 037616 | When reset 0016

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| IICM2 | I2C mode selection bit 2 | Refer to Table 1.44 | O | O |
| CSC | Clock-synchronous bit | 0 : Disabled<br>1 : Enabled | O | O |
| SWC | SCL wait output bit | 0 : Disabled<br>1 : Enabled | O | O |
| ASL | SDA output stop bit | 0 : Disabled<br>1 : Enabled | O | O |
| STAC | UART2 initialization bit | 0 : Disabled<br>1 : Enabled | O | O |
| SWC2 | SCL wait output bit 2 | 0: UART2 clock<br>1: 0 output | O | O |
| SDHI | SDA output disable bit | 0: Enabled<br>1: Disabled (high impedance) | O | O |
| SHTC | Start/stop condition control bit | Set this bit to "1" in I2C mode | O | O |

UART2 Special mode register 3 (I2C and SPI bus exclusive use register)




Symbol U2SMR3 | Address 037516 | When reset 0016

| Bit symbol | Bit name | Function during clock synchronous serial I/O mode | Function during UART mode | R | W |
|---|---|---|---|---|---|
| SPIM | SPI mode select bit | 0 : Normal mode<br>1 : SPI mode | Must always be "0" | O | O |
| CPHA | SPI clock-phase select bit | 0 : Data latched on falling clock edge<br>1 : Data latched on rising clock edge | Must always be "0" | O | O |
| Nothing is assigned. Write "0" when writing to these bits. If read, the value is Indeterminate. However, when SDDS = "1", a "0" value is read. (Note 1) | | | | – | – |
| DL0 | SDA digital delay set up bit (Notes 1, 2, 3, 4, 5) | b7 b6 b5<br>0 0 0 : Analog delay is selected<br>0 0 1 : 2 cycle of 1/f(XIN)<br>0 1 0 : 3 cycle of 1/f(XIN)<br>0 1 1 : 4 cycle of 1/f(XIN)<br>1 0 0 : 5 cycle of 1/f(XIN)<br>1 0 1 : 6 cycle of 1/f(XIN)<br>1 1 0 : 7 cycle of 1/f(XIN)<br>1 1 1 : 8 cycle of 1/f(XIN) | Digital delay is selected | O | O |
| DL1 | | | | O | O |
| DL2 | | | | O | O |

Note 1: This bit can be read or written to when UART2 special mode register U2SMR at address 037716 bit 7 (SDDS: SDA digital delay select bit) = "1". When the initial value of UART2 special mode register 3 (U2SMR3 is read after setting SDDS = "1", the value is "0016". When writing to U2SMR3 after setting SDDS = "1", be sure to write 0s to bits 0 - 4. When SDDS = "0", This register cannot be written to; when read, the value is indeterminate.

Note 2: These bits are initialized to "000" when SDDS = "0", with the analog delay circuit selected. After a reset these bits are set to "000", with the analog delay circuit selected. However, because these bits can be read only when SDDS = "1", the value read from these bits when SDDS = "0" is indeterminate.

Note 3: When analog delay is selected, only the analog delay value is effective; when digital delay is selected, only the digital delay value is effective.

Note 4: The amount of delay varies with the load on SCL and SDA pins. Also, when using an external clock, the amount of delay increases by about 100ns. Be sure to take this into account when using this device.

Note 5: Reset values for SPIM and CPHA are not affected by the state of SDDs. Their reset values are always "0".

**Fig. 1.86. Serial I/O-related registers (6)**

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

The clock synchronous serial I/O mode uses a transfer clock to transmit and receive data. Tables 1.34 and 1.35 list the specifications of the clock synchronous serial I/O mode. Figure 1.87 shows the UARTi transmit/receive mode register.

**Table 1.34. Specifications of clock synchrounous serial I/O mode (1)**

| Item | Specification |
|---|---|
| Transfer data format | Transfer data length: 8 bits |
| Transfer clock | When internal clock is selected (bit 3 at addresses $03A0_{16}$, $03A8_{16}$, $0378_{16}$ = 0 ): fi/2 (n+1) (Note 1) fi = f1, f8, f32<br>When external clock is selected (bit 3 at addresses $03A0_{16}$, $03A8_{16}$ $0378_{16}$ = 1 ): Input from CLKi pin |
| Transmission/ reception control | $\overline{CTS}$ function/$\overline{RTS}$ function/$\overline{CTS}$, $\overline{RTS}$ function chosen to be invalid |
| Transmission start condition | To start transmission, the following requirements must be met:<br>-Transmit enable bit (bit 0 at addresses $03A5_{16}$, $03AD_{16}$, $037D_{16}$) = 1<br>-Transmit buffer empty flag (bit 1 at addresses $03A5_{16}$, $03AD_{16}$, $037D_{16}$) = 0<br>-When $\overline{CTS}$ function selected, $\overline{CTS}$ input level = L<br>If external clock is selected, the following requirements must also me be:<br>-CLKi polarity select bit (bit 6 at addresses $03A4_{16}$, $03AC_{16}$, $037C_{16}$,) = 0 : CLKi input level = H<br>-CLKi polarity select bit (bit 6 at addresses $03A4_{16}$, $03AC_{16}$, $037C_{16}$) = 1 : CLKi input level = L |
| Receiving start condition | To start reception, the following requirement must be met:<br>-Receive enable bit (bit 2 at addresses $03A5_{16}$, $03AD_{16}$, $037D_{16}$) = 1<br>-Transmit enable bit (bit 0 at addresses $03A5_{16}$, $03AD_{16}$, $037D_{16}$) = 1<br>-Transmit buffer empty flag (bit 1 at addresses $03A5_{16}$, $03AD_{16}$, $037D_{16}$) = 0<br>If external clock is selected, the following requirements must also be met:<br>-CLKi polarity select bit (bit 6 at addresses $03A4_{16}$, $03AC_{16}$, $037C_{16}$) = 0 : CLKi input level = H<br>-CLKi polarity select bit (bit 6 at addresses $03A4_{16}$, $03AC_{16}$, $037C_{16}$) = 1 : CLKi input level = L |
| Interrupt request generation timing | When transmitting:<br>-Transmit interrupt cause select bit (bits 0, 1 at address $03B0_{16}$, bit 4 at address $037D_{16}$) = 0 : Interrupts requested when data transferred from UARTi transfer buffer register to UARTi transmit register is complete<br>-Transmit interrupt cause select bit (bit 0, 1 at address $03B0_{16}$, bit 4 at address $037D_{16}$) = 1 : Interrupts requested when data transmission from UARTi transfer register is complete<br>When receiving:<br>-Interrupts requested when data transferred from UARTi receive register to UARTi receive buffer register is complete. |
| Error detection | Overrun error (Note 2)<br>This error occurs when the next data are ready before contents of UARTi receive buffer register are read out |

Note 1: n denotes the value $00_{16}$ to $FF_{16}$ that is set to the UART bit rate generator.
Note 2: If an overrun error occurs, the UARTi receive buffer will have the next data written in.
Also, the UART receive interrupt requst bit is not set to 1 .

**Table 1.35. Specifications of clock synchronous serial I/O mode (2)**

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

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

UARTi transmit/receive mode registers

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

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

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

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

UART2 transmit/receive mode register

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

Symbol: U2MR — Address: 037816 — When reset: 00$_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SMD0 | Serial I/O mode select bit | b2 b1 b0<br>0 0 1 : Clock synchronous serial I/O mode | O | O |
| SMD1 | | | O | O |
| SMD2 | | | O | O |
| CKDIR | Internal/external clock select bit | 0 : Internal clock<br>1 : External clock (Note 1) | O | O |
| STPS | Invalid in clock synchronous serial I/O mode | | O | O |
| PRY | | | O | O |
| PRYE | | | O | O |
| IOPOL | TxD, RxD I/O polarity reverse bit (Note) | 0 : No reverse (Note 2)<br>1 : Reverse | O | O |

Note 1: Set the corresponding port direction register to "0".
Note 2: Usually set to "0".

**Fig. 1.87. UARTi transmit/receive mode register in clock synchronous serial I/O mode**

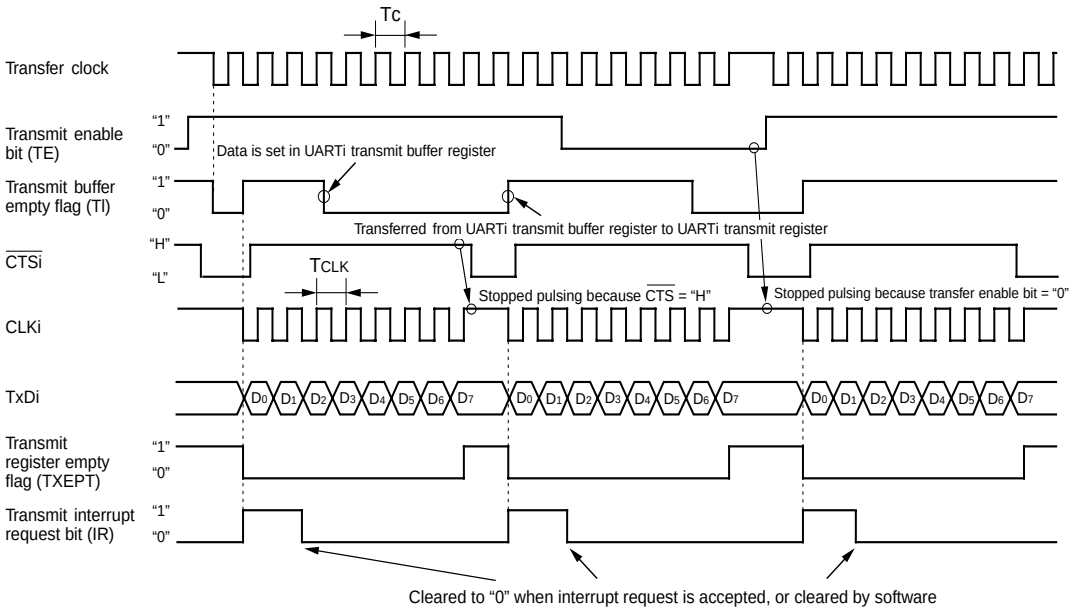
Table 1.36 lists the functions of the input/output pins during clock synchronous serial I/O mode. This table shows the pin functions when the transfer clock output from multiple pins and the separate $\overline{CTS}/\overline{RTS}$ pins functions are not selected. Note that for a period from when the UARTi operation mode is selected to when transfer starts, the TxDi pin outputs a "H". (If the N-channel open-drain is selected, this pin is in floating state).

Fig. 1.88 shows a typical transmit/receive timings in clock synchronous serial I/O mode.

**Table 1.36. Input/output pin functions**

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

**Example of transit timing (when internal clock is selected)**

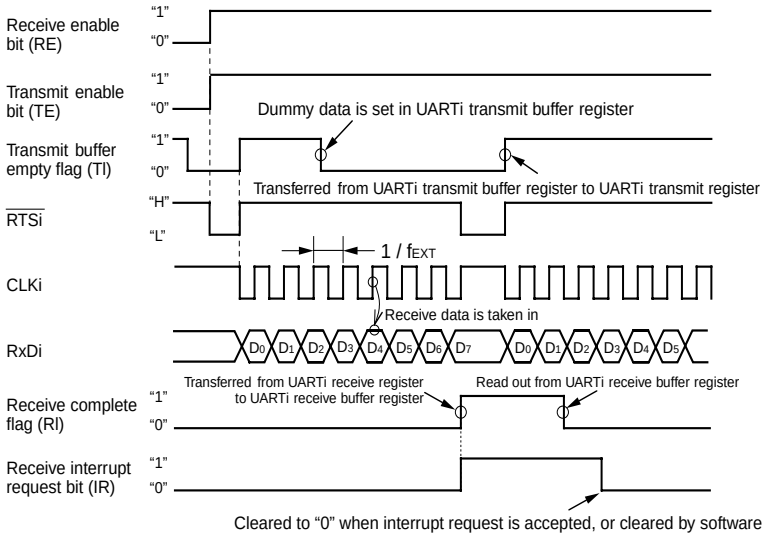



Shown in ( ) are bit symbols.

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

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

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




Shown in ( ) are bit symbols.

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

fEXT: frequency of external clock

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

**Fig. 1.88. Typical transmit/receive timings in clock synchronous serial I/O mode**

### (a) Polarity select function

As shown in Figure 1.89, the CLK polarity select bit (bit 6 at addresses 03A4$_{16}$, 03AC$_{16}$, 037C$_{16}$) allows selection of the polarity of the transfer clock.

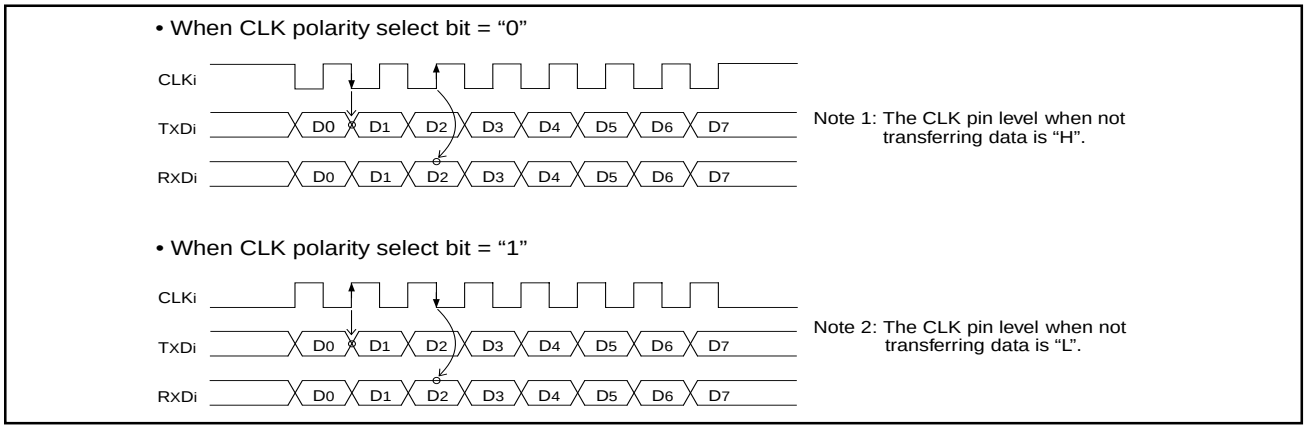



**Fig. 1.89. Polarity of transfer clock**

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

As shown in Figure 1.90, when the transfer format select bit (bit 7 at addresses 03A4$_{16}$, 03AC$_{16}$, 037C$_{16}$) = “0”, the transfer format is “LSB first”; when the bit = “1”, the transfer format is “MSB first”.

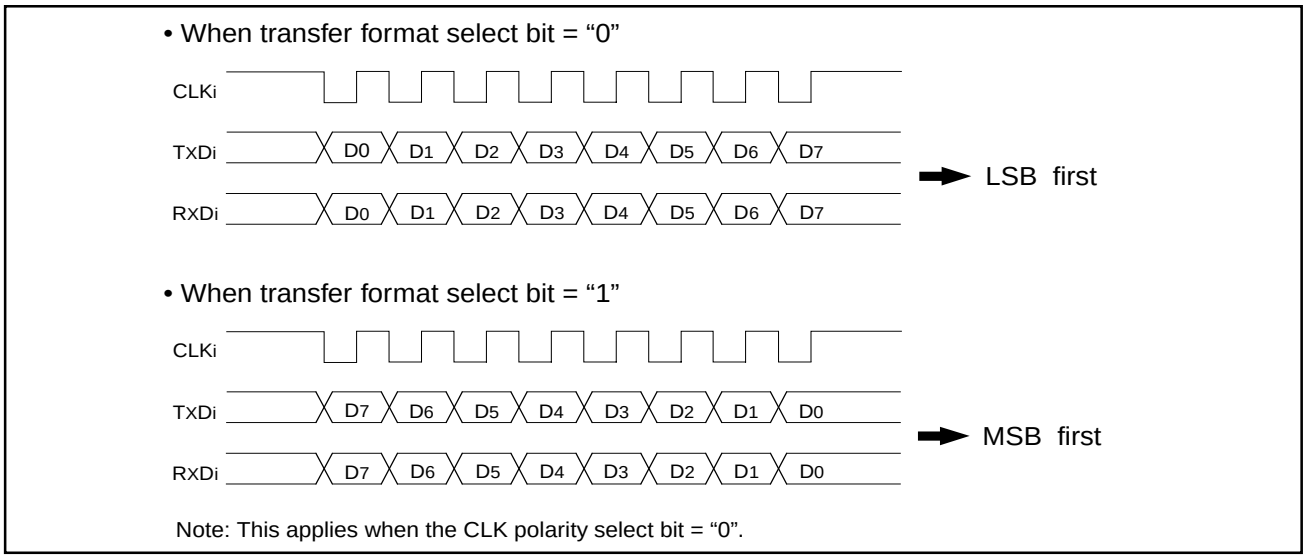



**Fig. 1.90. Transfer format**

### (c) Transfer clock output from multiple pins function (UART1)

This function allows the setting two transfer clock output pins and choosing one of the two to output a clock by using the CLK and CLKS select bit (bits 4 and 5 at address 03B0$_{16}$). (See Figure 1.91). The multiple pins function is valid only when the internal clock is selected for UART1. Note that when this function is selected, UART1 $\overline{CTS}$/$\overline{RTS}$ function cannot be used.

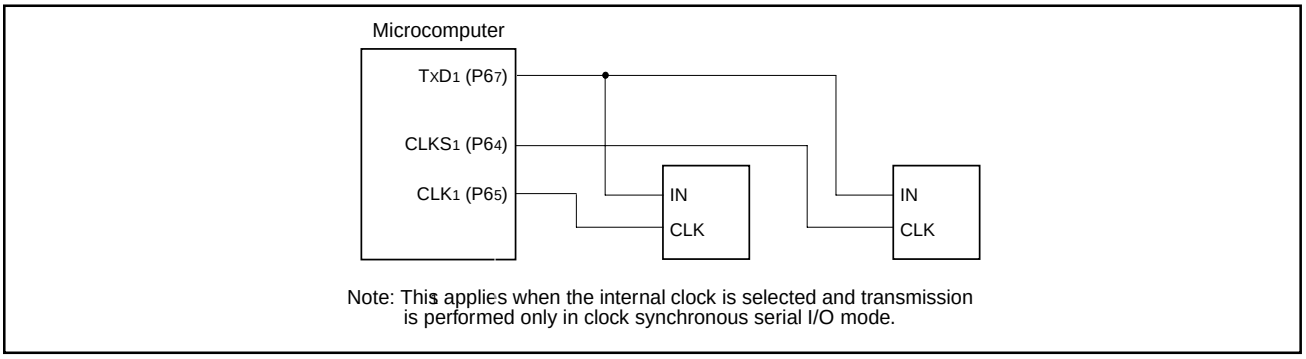



**Fig. 1.91. The transfer clock output from the multiple pins function usage**

### (d) Continuous receive mode

If the continuous receive mode enable bit (bits 2 and 3 at address 03B0$_{16}$, bit 5 at address 037D$_{16}$) is set to "1", the unit is placed in continuous receive mode. When the receive buffer register is read, the unit goes to a receive enable state without having to reset dummy data to the transmit buffer.

### (e) Separate $\overline{CTS}$/$\overline{RTS}$ pins function (UART0)

Refer to the Clock Asynchronous Serial I/O Mode section (Page 1-124) for setting the I/O pin functions. This function is invalid if the transfer clock output from the multiple pin function is selected.

### (f) Serial data logic switch function (UART2)

When the data logic select bit (bit 6 at address 037D$_{16}$) = "1", the data are reversed when writing to the transmit buffer register or reading from the receive buffer register. Figure 1.92 shows an example of the serial data logic switch timing function.

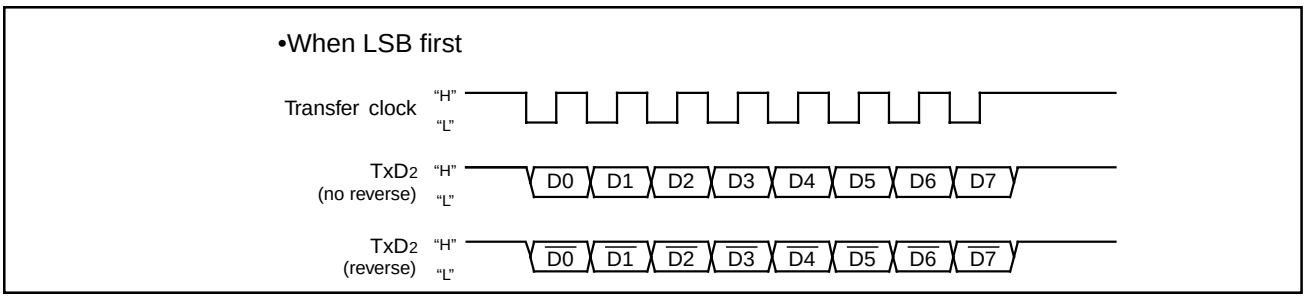



**Fig. 1.92. Serial data logic switch timing**

## (2) Clock Asynchronous Serial I/O (UART) Mode

The UART mode allows transmitting and receiving data after setting the desired transfer rate and transfer data format. Tables 1.37 and 1.38 list the specifications of the UART mode. Figure 1.93 shows the UARTi transmit/receive mode register.

**Table 1.37. Specifications of UART Mode (1)**

| Item | Specification |
|---|---|
| Transfer data format | • Character bit (transfer data): 7 bits, 8 bits, or 9 bits as selected<br>• Start bit: 1 bit<br>• Parity bit: Odd, even, or none as selected<br>• Stop bit: 1 bit or 2 bits as selected |
| Transfer clock | • When internal clock is selected (bit 3 at addresses $03A0_{16}$, $03A8_{16}$, $0378_{16}$ = "0") : fi/16(n+1) (Note 1) fi = $f_1$, $f_8$, $f_{32}$<br>• When external clock is selected (bit 3 at addresses $03A0_{16}$, $03A8_{16}$ ="1") : $f_{EXT}$/16(n+1) (Note 1) (Note 2) |
| Transmit/receive control | • $\overline{CTS}$ function/$\overline{RTS}$ function/$\overline{CTS}$, $\overline{RTS}$ function chosen to be invalid |
| Transmit start condition | • To start transmission, the following requirements must be met:<br>- Transmit enable bit (bit 0 at addresses $03A5_{16}$, $03AD_{16}$, $037D_{16}$) = "1"<br>- Transmit buffer empty flag (bit 1 at addresses $03A5_{16}$, $03AD_{16}$, $037D_{16}$) = "0"<br>- When $\overline{CTS}$ function selected, $\overline{CTS}$ input level = "L" |
| Receive start condition | • To start reception, the following requirements must be met:<br>- Receive enable bit (bit 2 at addresses $03A5_{16}$, $03AD_{16}$, $037D_{16}$) = "1"<br>- Start bit detection |
| Interrupt request generation timing | • When transmitting<br>- Transmit interrupt cause select bits (bits 0,1 at address $03B0_{16}$, bit4 at address $037D_{16}$) = "0": Interrupts requested when data transfer from UARTi transfer buffer register to UARTi transmit register is completed<br>- Transmit interrupt cause select bits (bits 0, 1 at address $03B0_{16}$, bit4 at address $037D_{16}$) = "1": Interrupts requested when data transmission from UARTi transfer register is completed<br>• When receiving<br>- Interrupts requested when data transfer from UARTi receive register to UARTi receive buffer register is completed |
| Error detection | • Overrun error (Note 3)<br>This error occurs when the next character is received before contents of UARTi receive buffer register are read out<br>• Framing error<br>This error occurs when the number of stop bits set is not detected<br>• Parity error<br>This error occurs when if parity is enabled, the number of 1's in parity and character bits does not match the number of 1's set<br>• Error sum flag<br>This flag is set (= 1) when any of the overrun, framing, and parity errors is encountered |

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

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

Note 3: If an overrun error occurs, the UARTi receive buffer will have the last or most recent data written to it. Note also that UARTi receive interrupt requst bit is not set to "1"

**Table 1.38. Specifications of UART Mode (2)**

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

UARTi transmit / receive mode registers

b7 b6 b5 b4 b3 b2 b1 b0

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

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

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

UART2 transmit / receive mode register

b7 b6 b5 b4 b3 b2 b1 b0

Symbol: U2MR | Address: 037816 | When reset: 00$_{16}$

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

Note: Usually set to "0".

**Fig. 1.93. UARTi transmit/receive mode register in UART mode**

Table 1.39lists the functions of the input/output pins during UART mode. This table shows the pin functions when the separate CTS/$\overline{\text{RTS}}$ pins function is not selected. Note that for a period from when the UARTi operation mode is selected to when transfer starts, the TxDi pin outputs a “H”. (If the N-channel open-drain is selected, this pin is in floating state).

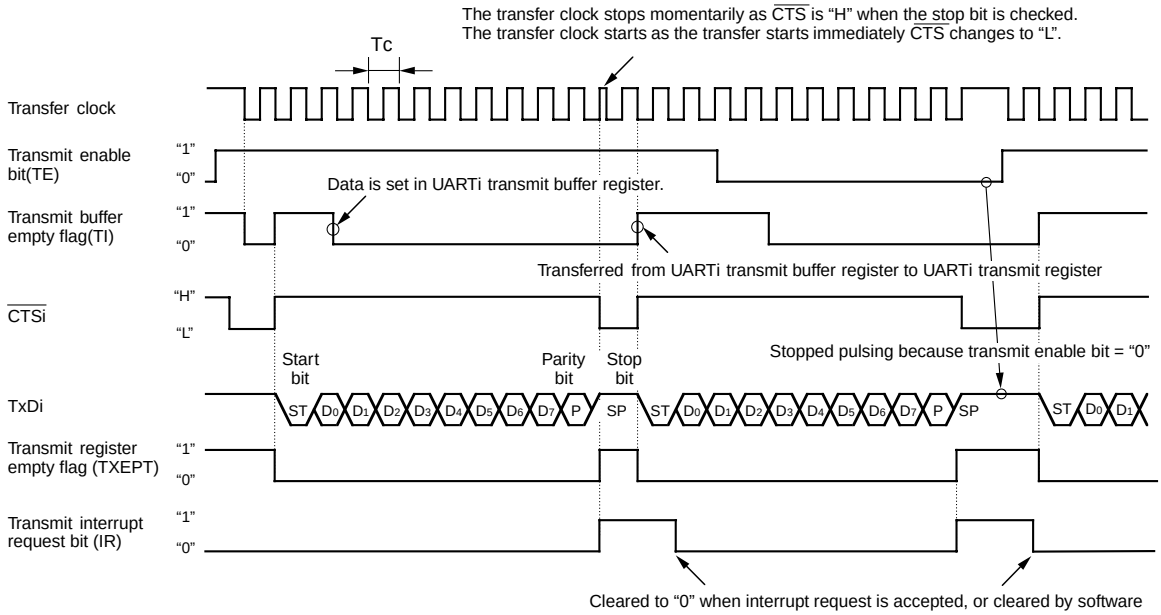
Figures 1.94 and 1.95 show the typical transmit timings in UART mode (UART0, UART1, UART2). Figure 1.96 shows the typical receive timing in UART mode.

**Table 1.39. Input/output pin functions in UART mode**

| Pin name | Function | Method of selection |
|---|---|---|
| TxDi (P6$_3$, P6$_7$, P7$_0$) | Serial data output | |
| RxDi (P6$_2$, P6$_6$, P7$_1$) | Serial data input | Port P6$_2$, P6$_6$ and P7$_1$ direction register (bits 2 and 6 at address 03EE$_{16}$, bit 1 at address 03EF$_{16}$)= “0”<br>(Can be used as an input port when performing transmission only) |
| CLKi (P6$_1$, P6$_5$, P7$_2$) | Programmable I/O port | Internal/external clock select bit (bit 3 at address 03A0$_{16}$, 03A8$_{16}$, 037$_{16}$) = “0” |
| | Transfer clock input | Internal/external clock select bit (bit 3 at address 03A0$_{16}$, 03A8$_{16}$) = “1”<br>Port P6$_1$, P6$_5$ direction register (bits 1 and 5 at address 03EE$_{16}$) = “0” |
| $\overline{\text{CTS}}$i/$\overline{\text{RTS}}$i (P6$_0$, P6$_4$, P7$_3$) | $\overline{\text{CTS}}$ input | $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit (bit 4 at address 03A4$_{16}$, 03AC$_{16}$, 037C$_{16}$) =“0”<br>$\overline{\text{CTS}}$/$\overline{\text{RTS}}$ function select bit (bit 2 at address 03A4$_{16}$, 03AC$_{16}$, 037C$_{16}$) = “0”<br>Port P6$_0$, P6$_4$ and P7$_3$ direction register (bits 0 and 4 at address 03EE$_{16}$, bit 3 at address 03EF$_{16}$) = “0” |
| | $\overline{\text{RTS}}$ output | $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit (bit 4 at address 03A4$_{16}$, 03AC$_{16}$, 037C$_{16}$) = “0”<br>$\overline{\text{CTS}}$/$\overline{\text{RTS}}$ function select bit (bit 2 at address 03A4$_{16}$, 03AC$_{16}$, 037C$_{16}$) = “1” |
| | Programmable I/O port | $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit (bit 4 at address 03A4$_{16}$, 03AC$_{16}$, 037C$_{16}$) = “1” |

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

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

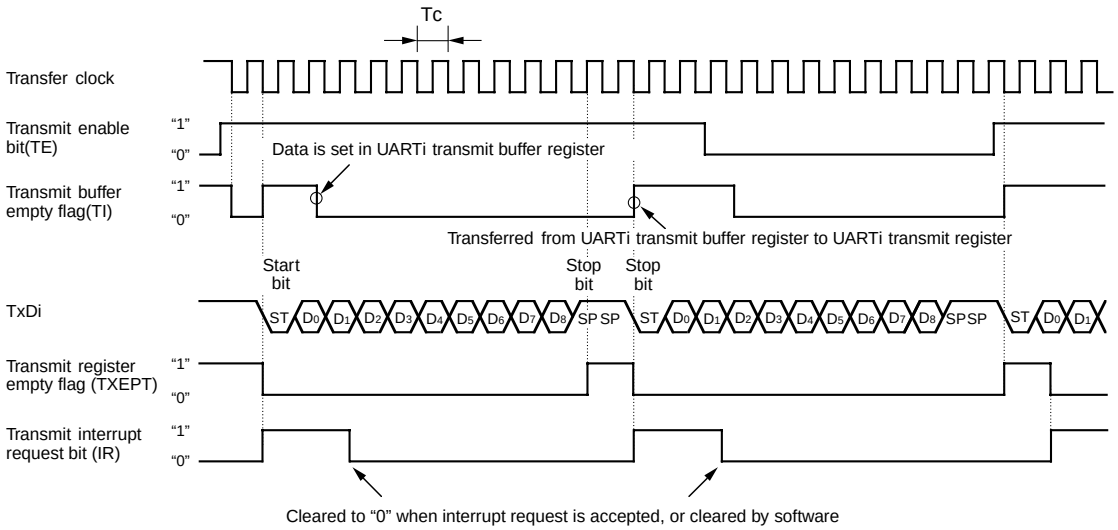



Shown in ( ) are bit symbols.

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

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

**Example of transmit timing when tranfer data is 9 bits long (parity disabled, two-stop bits)**

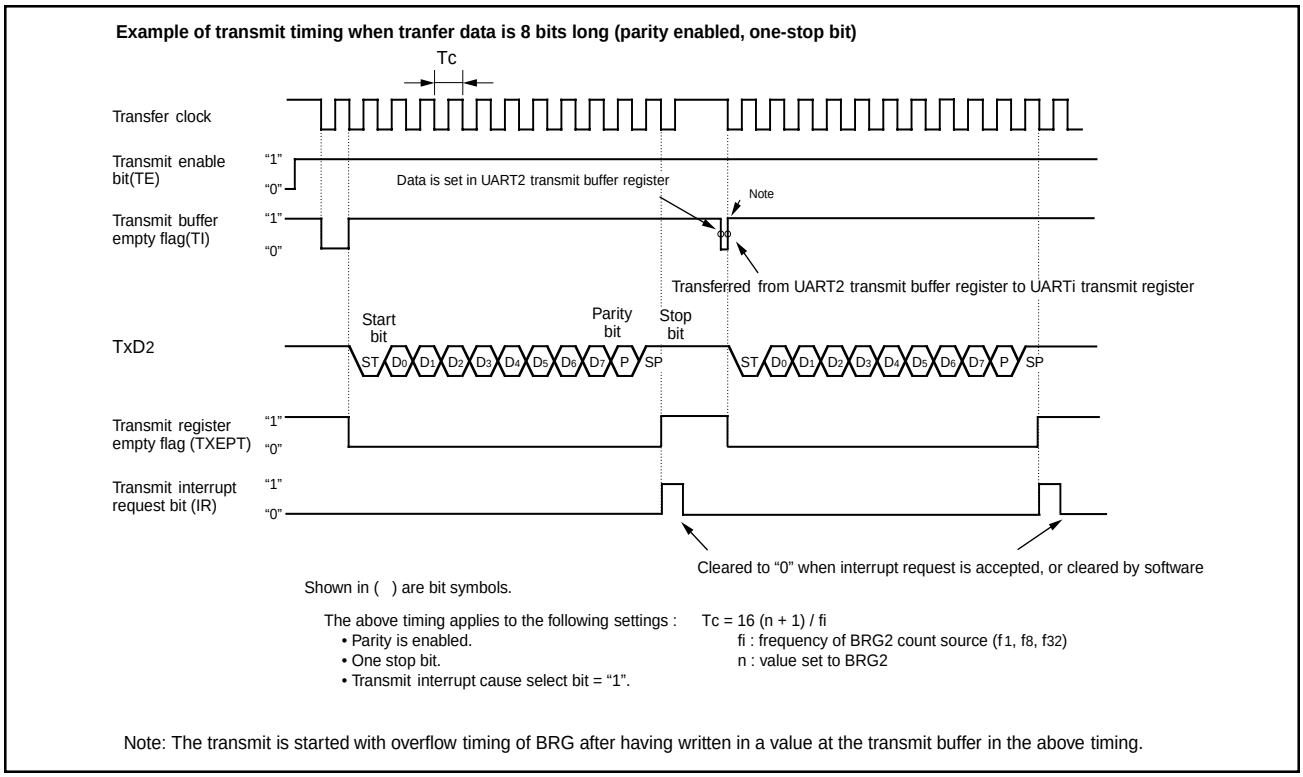



Shown in ( ) are bit symbols.

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

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

**Fig. 1.94. Typical transmit timings in UART mode (UART0, UART1)**




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

**Fig. 1.95. Typical transmit timings in UART mode (UART2)**

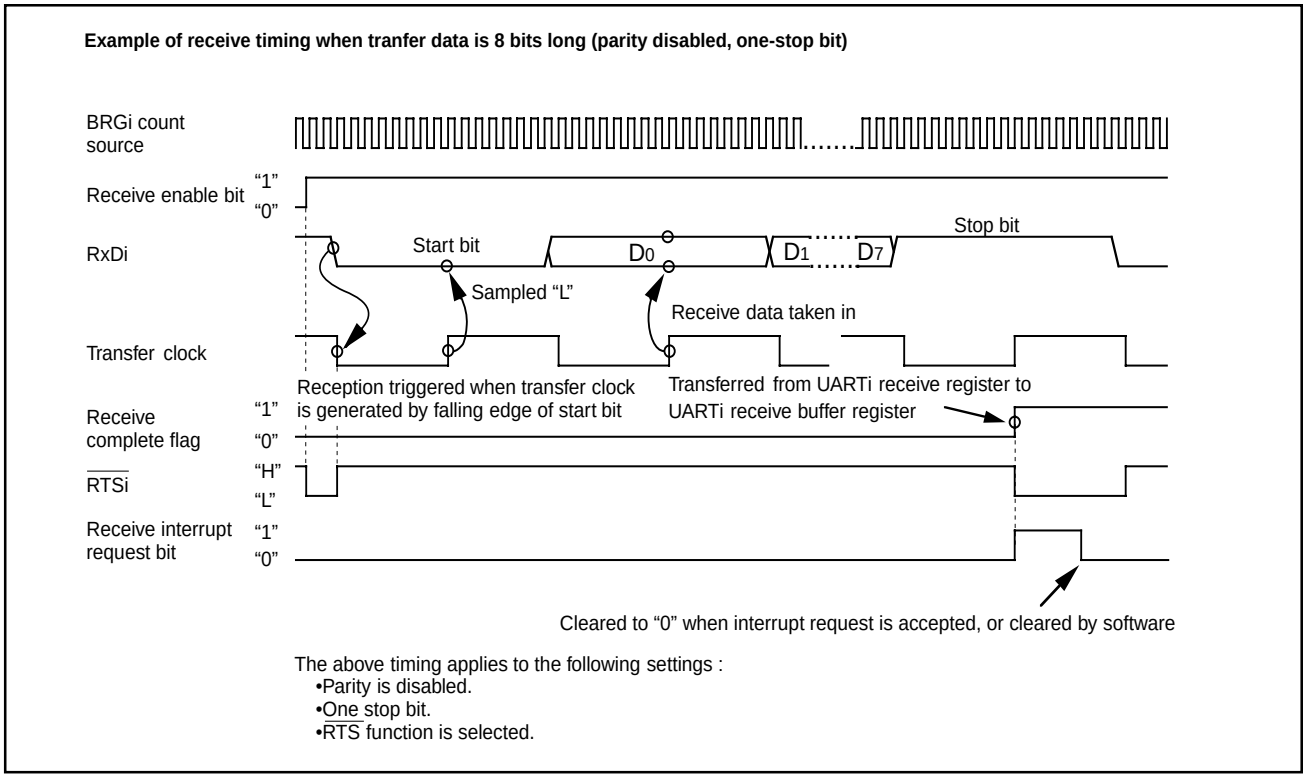
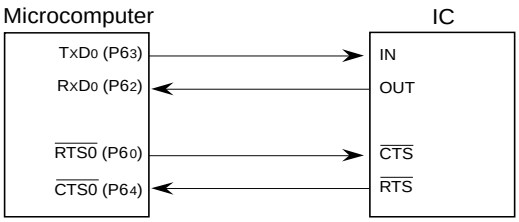



**Fig. 1.96. Typical receive timing in UART mode**

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

Setting the $\overline{CTS}/\overline{RTS}$ separate bit (bit 6 of address $03B0_{16}$) to "1" inputs/outputs the $\overline{CTS}$ signal and $\overline{RTS}$ signal from different pins. Choose which to use, $\overline{CTS}$ or $\overline{RTS}$, by use of the $\overline{CTS}/\overline{RTS}$ function select bit (bit 2 of address $03A4_{16}$). (See Fig. 1.97). This function is effective in UART0 only. With this function chosen, the user cannot use the $\overline{CTS}/\overline{RTS}$ function. Set "0" both to the $\overline{CTS}/\overline{RTS}$ function select bit (bit 2 of address $03AC_{16}$) and to the $\overline{CTS}/\overline{RTS}$ disable bit (bit 4 of address $03AC_{16}$).




Note : The user cannot use $\overline{CTS}$ and $\overline{RTS}$ at the same time.

**Fig. 1.97. The separate $\overline{CTS}/\overline{RTS}$ pins function usage**

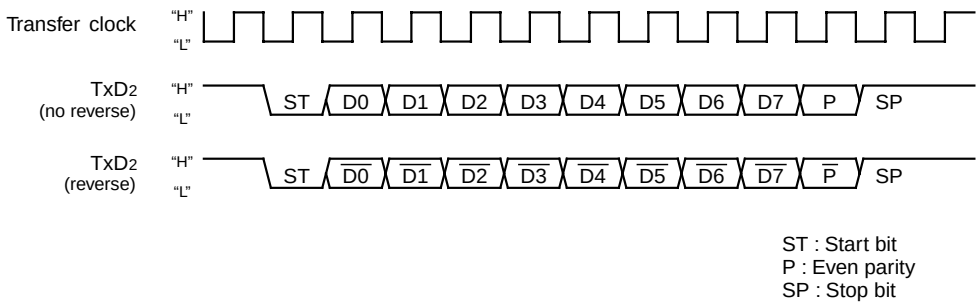
### (b) Sleep mode (UART0, UART1)

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

### (c) Function for switching serial data logic (UART2)

When the data logic select bit (bit 6 of address $037D_{16}$) is assigned 1, data is inverted in writing to the transmission buffer register or reading the reception buffer register. Figure 1.98 shows the example of timing for switching serial data logic.

**Example of timing for switching serial data logic when LSB is first (parity enabled, one-stop bit)**




ST : Start bit
P : Even parity
SP : Stop bit

**Fig. 1.98. Timing for switching serial data logic**

### (d) TxD, RxD I/O polarity reverse function (UART2)

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

### (e) Bus collision detection function (UART2)

This function is to sample the output level of the TxD pin and the input level of the RxD pin at the rising edge of the transfer clock; if their values are different, then an interrupt request occurs. Figure 1.99 shows the example of detection timing of a bus collision (in UART mode).

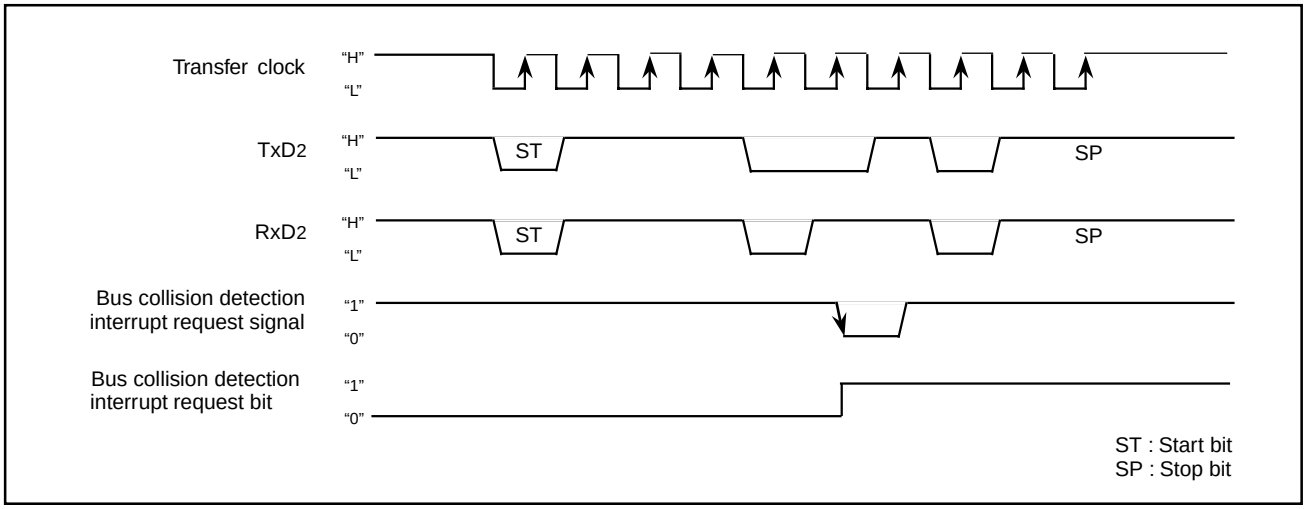



**Fig. 1.99. Detection timing of a bus collision (in UART mode)**

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

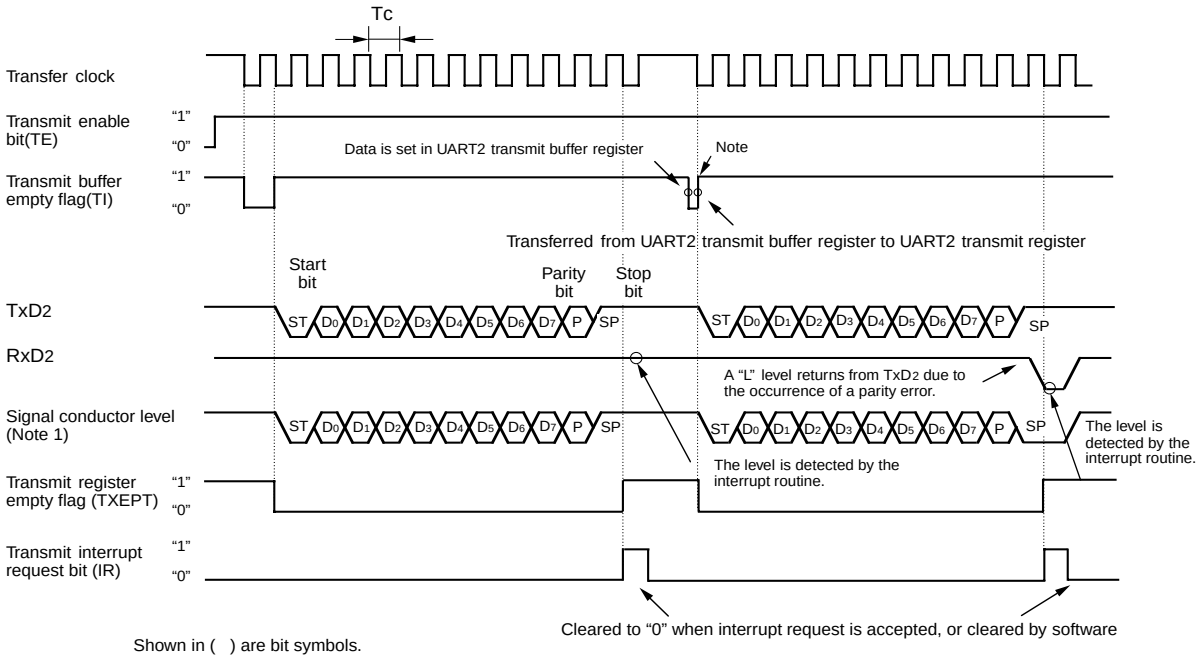
The SIM interface is used for connecting the microcomputer with a memory card or the like; adding some extra settings in UART2 clock-asynchronous serial I/O mode allows the user to effect this function. Table 1.40 shows the specifications of clock-asynchronous serial I/O mode (compliant with the SIM interface). Figure 1.100 shows a typical transmit/receive timing in UART mode (compliant with the SIM interface).

**Table 1.40. Specifications of clock-asynchronous serial I/O mode (compliant with the SIM interface)**

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

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

Note 2: If an overrun error occurs, the UART2 receive buffer will have the last or most recent data written. Note also the UARTi receive interrupt request bit is not set to "1".

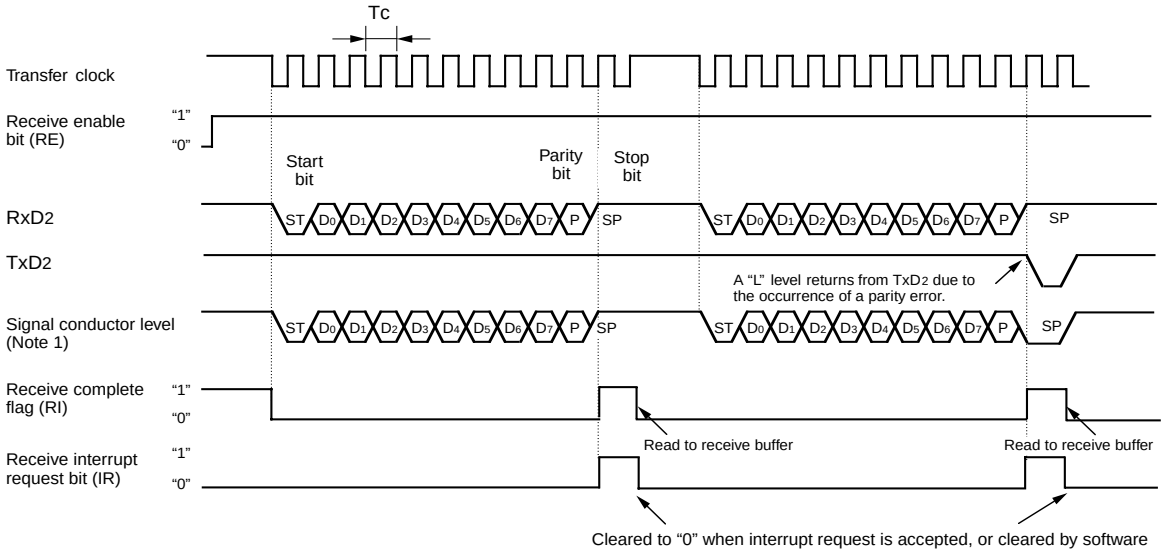



Shown in ( ) are bit symbols.

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

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

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




Shown in ( ) are bit symbols.

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

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

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

**Fig. 1.100. Typical transmit/receive timing in UART mode (compliant with the SIM interface)**

### (a) Function for outputting a parity error signal

With the error signal output enable bit (bit 7 of address 037D16) assigned "1", you can output an "L" level from the TxD2 pin when a parity error is detected. In step with this function, the generation timing of a transmission completion interrupt changes to the detection timing of a parity error signal. Figure 1.101 shows the output timing of the parity error signal.

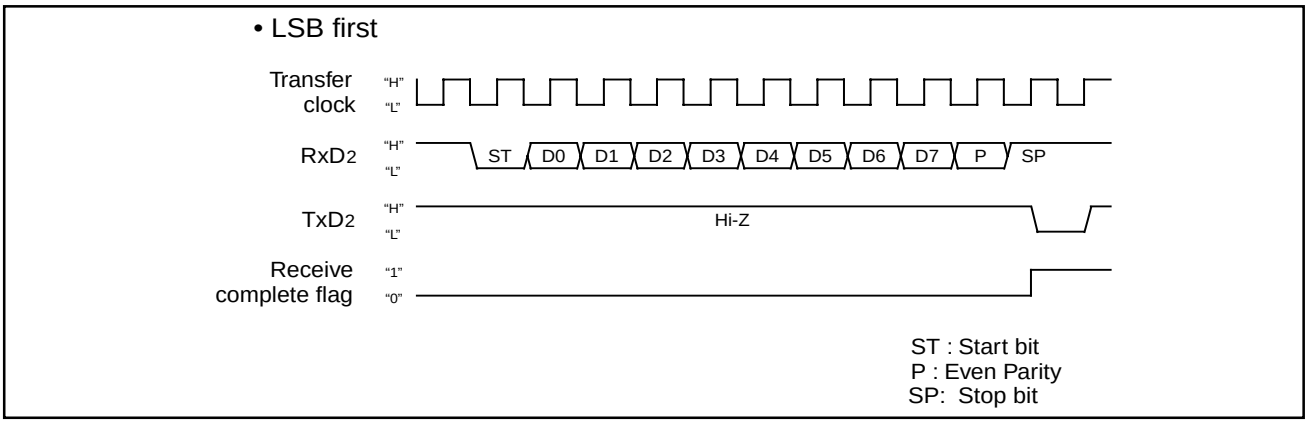



**Fig. 1.101. Output timing of the parity error signal**

### (b) Direct format/inverse format

Connecting the SIM card allows you to switch between direct format and inverse format. If you choose the direct format, D0 data is output from TxD2. If you choose the inverse format, D7 data is inverted and output from TxD2. Figure 1.102 shows the SIM interface format. Figure 1.103 shows the example of connecting the SIM interface. Connect TxD2 and RxD2 and apply pull-up.

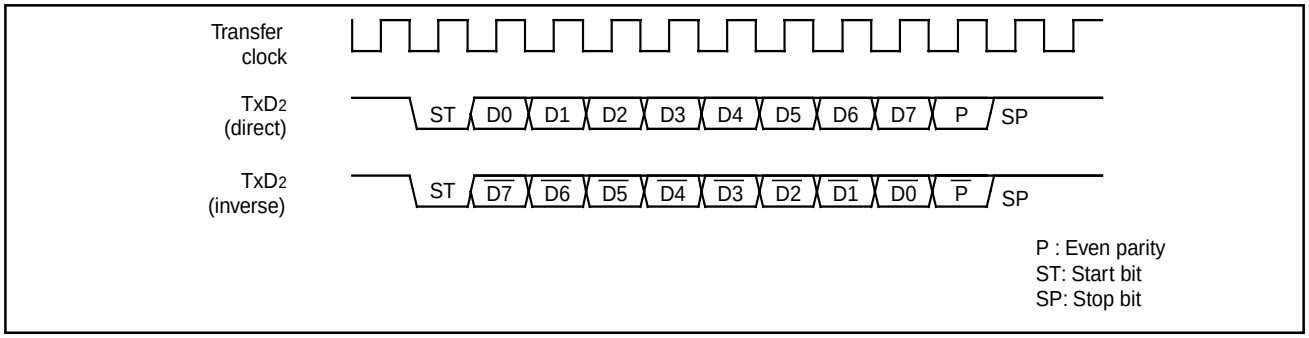



**Fig. 1.102. SIM interface format**

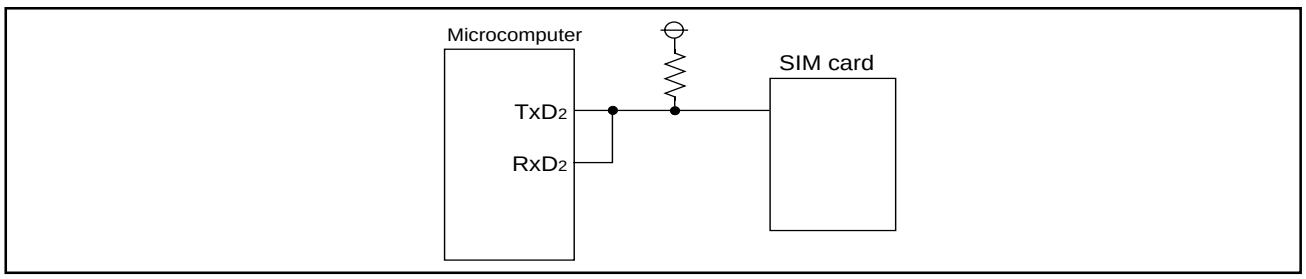



**Fig. 1.103. Connecting the SIM interface**

## UART2 in I2C Mode

The UART2 special mode register (address 037716) is used to control UART2 in various ways. Figure 1.104 shows the UART2 special mode register. Setting the I2C mode select bit (bit 1 of U2SMR) to "1" selects I2C mode.

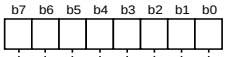
Table 1.41 shows the relation between the I2C mode select bit and respective control workings. Since this function uses clock-synchronous serial I/O mode, set this bit to “0” in UART mode.

**Table 1.41. Features in I2C mode**

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

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

UART2 Special mode register

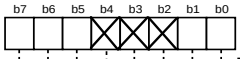



Symbol U2SMR | Address 037716 | When reset 0016

| Bit symbol | Bit name | Function (During clock synchronous serial I/O mode) | Function (During UART mode) | R | W |
|---|---|---|---|---|---|
| IICM | IIC mode selection bit | 0 : Normal mode<br>1 : IIC mode | Must always be "0" | O | O |
| ABC | Arbitration lost detecting flag control bit | 0 : Update per bit<br>1 : Update per byte | Must always be "0" | O | O |
| BBS | Bus busy flag | 0 : STOP condition detected<br>1 : START condition detected | Must always be "0" | O (Note 1) | O |
| LSYN | SCLL sync output enable bit | 0 : Disabled<br>1 : Enabled | Must always be "0" | O | O |
| ABSCS | Bus collision detect sampling clock select bit | Must always be "0" | 0 : Rising edge of transfer clock<br>1 : Underflow signal of timer A0 | O | O |
| ACSE | Auto clear function select bit of transmit enable bit | Must always be "0" | 0 : No auto clear function<br>1 : Auto clear at occurrence of bus collision | O | O |
| SSS | Transmit start condition select bit | Must always be "0" | 0 : Ordinary<br>1 : Falling edge of RxD2 | O | O |
| SDDS | SDA digital delay select bit (Note 2, 3) | 0 : Analog delay output is selected<br>1 : Digital delay output is selected (Must always be "0" when not using I2C mode) | Must always be "0" | O | O |

Note 1: Nothing but "0" may be written.
Note 2: When not in I2C mode, do not set this bit by writing a "1". During normal mode, fix it to "0". When this bit - "0" , UART2 special mode register 3 (U2SMR3 at address 037516) bits 7 to 5 (DL2 to DL0 = SDA digital delay setup bits) are initialized to "000", with the analog delay circuit selected. Also, when SDDS = "0", the U2SMR3 register cannot be read or written to.
Note 3: When analog delay is selected, only the analog delay value is effective; when digital delay is selected only the digital delay value is effective.

UART2 Special mode register 3 (I2C and SPI bus exclusive use register)




Symbol U2SMR3 | Address 037516 | When reset 0016

| Bit symbol | Bit name | Function during clock synchronous serial I/O mode | Function during UART mode | R | W |
|---|---|---|---|---|---|
| SPIM | SPI mode select bit | 0 : Normal mode<br>1 : SPI mode | Must always be "0" | O | O |
| CPHA | SPI clock-phase select bit | 0 : Data latched on falling clock edge<br>1 : Data latched on rising clock edge | Must always be "0" | O | O |
| Nothing is assigned. In an attempt to write to these bits, write "0". When read the value is indeterminate. However, when SDDS = "1", a "0" value is read. (Note 1) | | | | – | – |
| DL0 | SDA digital delay set up bit (Note 1, 2, 3, 4, 5) | b7 b6 b5<br>0 0 0 : Analog delay is selected<br>0 0 1 : 2 cycle of 1/f(XIN)<br>0 1 0 : 3 cycle of 1/f(XIN)<br>0 1 1 : 4 cycle of 1/f(XIN)<br>1 0 0 : 5 cycle of 1/f(XIN)<br>1 0 1 : 6 cycle of 1/f(XIN)<br>1 1 0 : 7 cycle of 1/f(XIN)<br>1 1 1 : 8 cycle of 1/f(XIN) | Digital delay is selected | O | O |
| DL1 | | | | O | O |
| DL2 | | | | O | O |

Note 1: This bit can be read or written to when UART2 special mode register U2SMR at address 037716 bit 7 (SDDS: SDA digital delay select bit) = "1". When the initial value of UART2 special mode register 3 (U2SMR3 is read after setting SDDS = "1", the value is "0016". When writing to U2SMR3 after setting SDDS = "1", be sure to write 0s to bits 0 - 4. When SDDS = "0", This register cannot be written to; when read, the value is indeterminate.
Note 2: These bits are initialized to "000" when SDDS = "0", with the analog delay circuit selected. After a reset these bits are set to "000", with the analog delay circuit selected. However, because these bits can be read only when SDDS = "1", the value read from these bits when SDDS = "0" is indeterminate.
Note 3: When analog delay is selected, only the analog delay value is effective; when digital delay is selected, only the digital delay value is effective.
Note 4: The amount of delay varies with the load on SCL and SDA pins. Also, when using an external clock, the amount of delay increases by about 100ns. Be sure to take this into account when using this device.
Note 5: Reset values for SPIM and CPHA are not affected by the state of SDDs. Their reset values are always "0".

**Fig. 1.104. UART2 special mode register**

Figure 1.105 shows the functional block diagram for I²C mode. Setting "1" in the I²C mode select bit (IICM) causes ports P7₀, P7₁, and P7₂ to work as data transmission-reception terminal SDA, clock input-output terminal SCL, and port P7₂ respectively. A delay circuit is added to the SDA transmission output, so the SDA output changes after SCL fully goes to "L". The SDA digital delay select bit (bit 7 at address 037716) can be used to select between analog delay and digital delay. When digital delay is selected, the amount of delay can be selected in the range of 2 cycles to 8 cycles of f1 using UART2 special mode register 3 (at address 037516). Delay circuit select conditions are shown in Table 1.42.

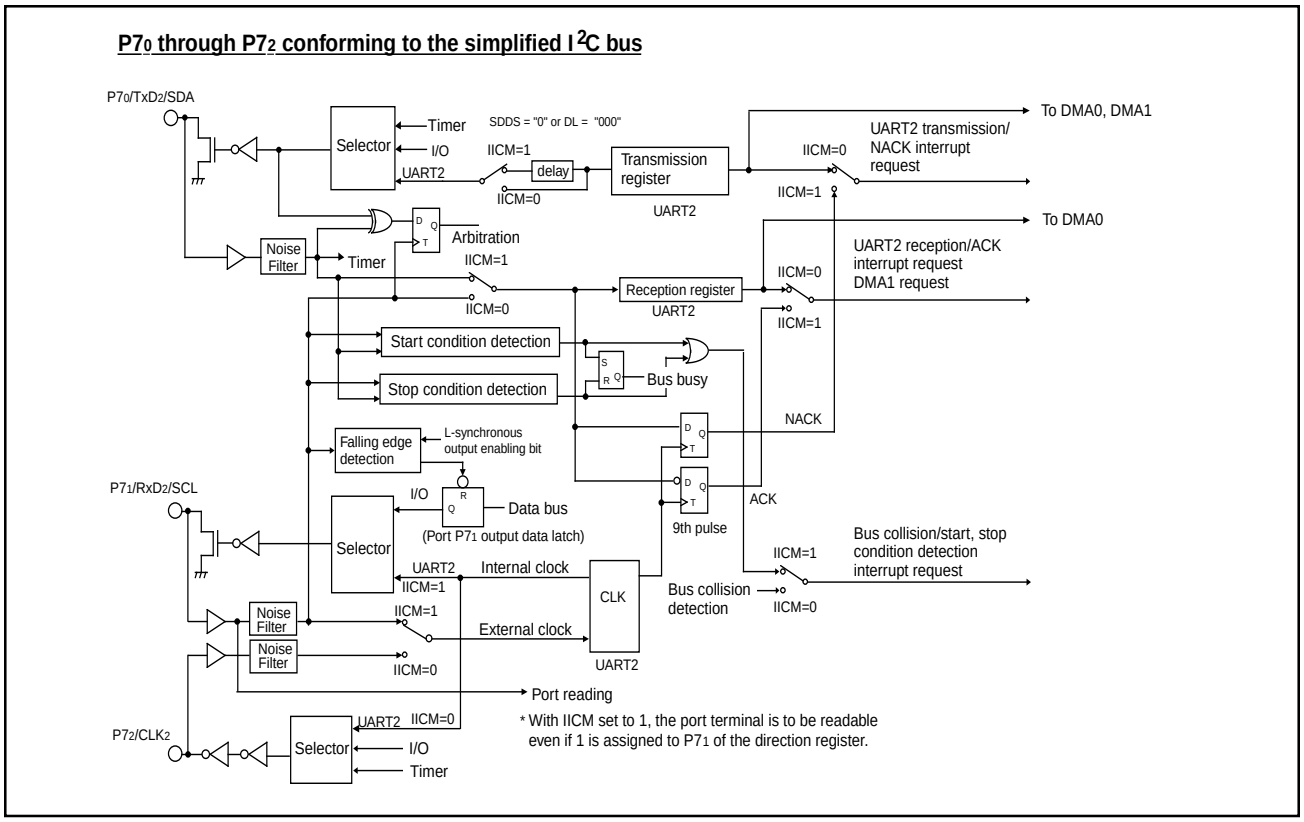



**Fig. 1.105. Functional block diagram for I²C mode**

**Table 1.42. Delay circuit select conditions**

| | Register value | | | Contents |
|---|---|---|---|---|
| | IICM | SDDS | DL | |
| Digital delay is selected | 1 | 1 | 001 to 111 | When digital delay is selected,no analog delay is added. Only digital delay is effective |
| Analog delay is selected | 1 | 1 | 000 | When DL is set ot "000", analog delay is selected no matter what value is set in SDDS. |
| | | 0 | (000) | When SDDS is set to "0", DL is initialized, so that DL = "000". |
| No delay | 0 | 0 | (000) | When IICM = "0", no delay circuit is selected. Always made sure SDDS = "0". |

An attempt to read Port P71 (SCL) gets the pin's level regardless of the content of the port direction register. The initial value of SDA transmission output in this mode goes to the value set in port P70. The interrupt factors of the bus collision detection interrupt, UART2 transmission interrupt, and of UART2 reception interrupt turn to the start/stop condition detection interrupt, acknowledgment non-detection interrupt, and acknowledgment detection interrupt respectively.

The start condition detection interrupt refers to the interrupt that occurs when the falling edge of the SDA terminal (P70) is detected with the SCL terminal (P71) staying “H”. The stop condition detection interrupt refers to the interrupt that occurs when the rising edge of the SDA terminal (P70) is detected with the SCL terminal (P71) staying “H”. The bus busy flag (bit 2 of the UART2 special mode register) is set to “1” by the start condition detection, and set to “0” by the stop condition detection.

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

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

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

Bit 3 of the UART2 special mode register is used as SCL- and L-synchronous output enable bit. Setting this bit to “1” goes the P71 data register to “0” in synchronization with the SCL terminal level going to “L”. Some other functions added are explained here. Figure 1.106 shows their workings.

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

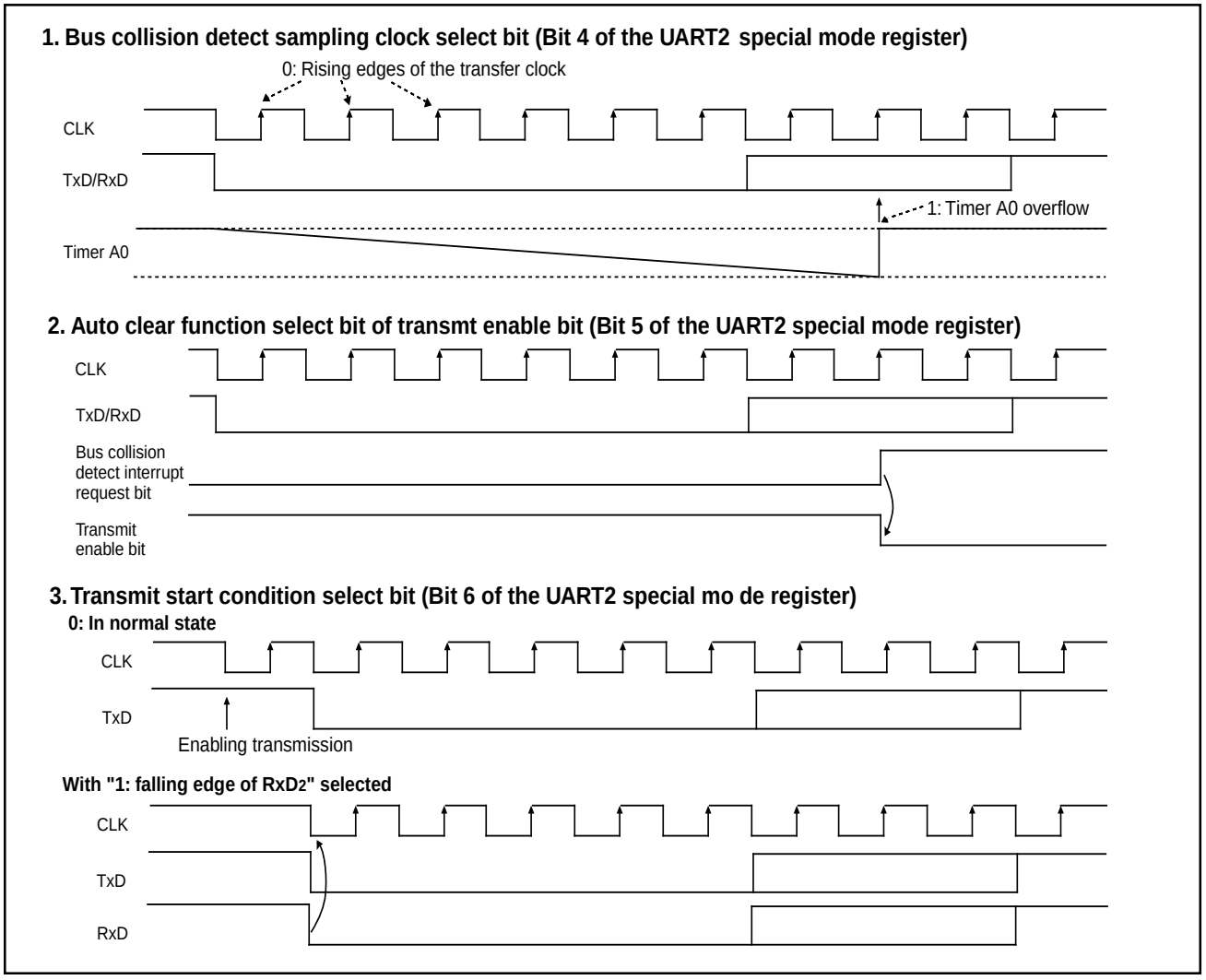



**Fig. 1.106. Some other functions added**

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

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

## UART2 Special Mode Register 2

UART2 special mode register 2 (address 037616) is used to further control UART2 in I2C mode. Figure 1.107 shows the UART2 special mode register 2.

Bit 0 of the UART2 special mode register 2 (address 037616) is used as the I2C mode select bit 2. Table 1.43 shows the types of control to be changed by I2C mode select bit 2 when the I2C mode select bit is set to "1". Figure 1.108 shows the timing characteristics of detecting the start condition and the stop condition. Set the start/stop condition control bit (bit 7 of UART2 special mode register 2) to "1" in I2C mode.

UART2 special mode register 2

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 | Symbol | Address | When reset |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | U2SMR2 | 037616 | 0016 |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| IICM2 | I2C mode selection bit 2 | Refer to Table 1.43 | ○ | ○ |
| CSC | Clock-synchronous bit | 0 : Disabled<br>1 : Enabled | ○ | ○ |
| SWC | SCL wait output bit | 0 : Disabled<br>1 : Enabled | ○ | ○ (Note) |
| ASL | SDA output stop bit | 0 : Disabled<br>1 : Enabled | ○ | ○ |
| STAC | UART2 initialization bit | 0 : Disabled<br>1 : Enabled | ○ | ○ |
| SWC2 | SCL wait output bit 2 | 0: UART2 clock<br>1: 0 output | ○ | ○ |
| SDHI | SDA output disable bit | 0: Enabled<br>1: Disabled (high impedance) | ○ | ○ |
| SHTC | Start/stop condition control bit | Set this bit to "1" in I2C mode (refer to Figure 1.108) | ○ | ○ |

**Fig. 1.107. UART2 special mode register 2**

**Table 1.43. Functions changed by I2C mode select bit 2**

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

| |
|---|
| 3 to 6 cycles < duration for setting-up (Note 2) |
| 3 to 6 cycles < duration for holding (Note 2) |

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

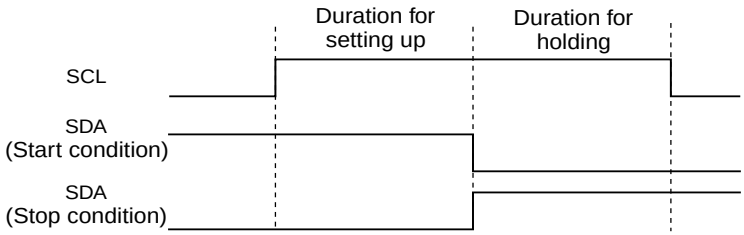



**Fig. 1.108. Timing characteristics of detecting the start condition and the stop condition (Note1)**

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

Bit 1 of the UART2 special mode register 2 (address 037616) is used as the clock synchronization bit. With this bit set to “1” at the time when the internal SCL is set to “H”, the internal SCL turns to “L” if the falling edge is found in the SCL pin; and the baud rate generator reloads the set value, and start counting within the “L” interval. When the internal SCL changes from “L” to “H” with the SCL pin set to “L”, stops counting the baud rate generator, and starts counting it again when the SCL pin turns to “H”. Due to this function, the UART2 transmission-reception clock becomes the logical product of the signal flowing through the internal SCL and that flowing through the SCL pin. This function operates over the period from the moment earlier by a half cycle than falling edge of the UART2 first clock to the rising edge of the ninth bit. To use this function, choose the internal clock for the transfer clock.

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

Bit 4 of the UART2 special mode register 2 (address 037616) is used as the UART2 initialization bit. Setting this bit to “1”, and when the start condition is detected, the microcomputer operates as follows.

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

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

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

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

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

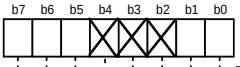
Bit 6 of the UART2 special mode register 2 ($0376_{16}$) is used as the SDA output disable bit. Setting this bit to "1" forces the SDA pin to turn to the high-impedance state. Refrain from changing the value of this bit at the rising edge of the UART2 transfer clock. There can be instances in which arbitration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

## UART2 in SPI mode

The UART2 special mode register 3 (address 037516) is used to activate the SPI mode.

Figure 1.109 shows the UART2 special mode register 3.

UART2 Special mode register 3 ($I^2C$ and SPI bus exclusive use register)




Symbol U2SMR3 | Address 037516 | When reset 0016

| Bit symbol | Bit name | Function during clock synchronous serial I/O mode | Function during UART mode | R | W |
|---|---|---|---|---|---|
| SPIM | SPI mode select bit | 0 : Normal mode<br>1 : SPI mode | Must always be "0' | O | O |
| CPHA | SPI clock-phase select bit | 0 : Data latched on falling clock edge<br>1 : Data latched on rising clock edge | Must always be "0' | O | O |
| Nothing is assigned. In an attempt to write to these bits, write "0". When read the value is indeterminate. However, when SDDS = "1", a "0" value is read. (Note 1) | | | | – | – |
| DL0 | SDA digital delay set up bit (Note 1, 2, 3, 4, 5) | b7 b6 b5<br>0 0 0 : Analog delay is selected<br>0 0 1 : 2 cycle of 1/f(XIN)<br>0 1 0 : 3 cycle of 1/f(XIN)<br>0 1 1 : 4 cycle of 1/f(XIN)<br>1 0 0 : 5 cycle of 1/f(XIN)<br>1 0 1 : 6 cycle of 1/f(XIN)<br>1 1 0 : 7 cycle of 1/f(XIN)<br>1 1 1 : 8 cycle of 1/f(XIN) | Digital delay is selected | O | O |
| DL1 | | | | O | O |
| DL2 | | | | O | O |

Note 1: This bit can be read or written to when UART2 special mode register U2SMR at address 037716 bit 7 (SDDS: SDA digital delay select bit) = "1". When the initial value of UART2 special mode register 3 (U2SMR3 is read after setting SDDS = "1", the value is "0016". When writing to U2SMR3 after setting SDDS = "1", be sure to write 0s to bits 0 - 4. When SDDS = "0", This register cannot be written to; when read, the value is indeterminate.

Note 2: These bits are initialized to "000" when SDDS = "0", with the analog delay circuit selected. After a reset these bits are set to "000", with the analog delay circuit selected. However, because these bits can be read only when SDDS = "1", the value read from these bits when SDDS = "0" is indeterminate.

Note 3: When analog delay is selected, only the analog delay value is effective; when digital delay is selected, only the digital delay value is effective.

Note 4: The amount of delay varies with the load on SCL and SDA pins. Also, when using an external clock, the amount of delay increases by about 100ns. Be sure to take this into account when using this device.

Note 5: Reset values for SPIM and CPHA are not affected by the state of SDDs. Their reset values are always "0".

**Fig. 1.109. UART2 Special mode register 3**

The SPI functionality is an 8 bit, synchronous communication protocol that is user programmable to use one of four different transfer formats. The four transfer formats support the four combinations of clock phase and clock polarity, as the clock relates to the data. Figure 1.110 shows the SPI system level view.

The existing UART2 already provides two of the transfer formats, with the CKPOL control bit. The SPI mode adds the ability to change the phase of the clock, with respect to the transmitted data, in each of the two existing clock polarity formats.

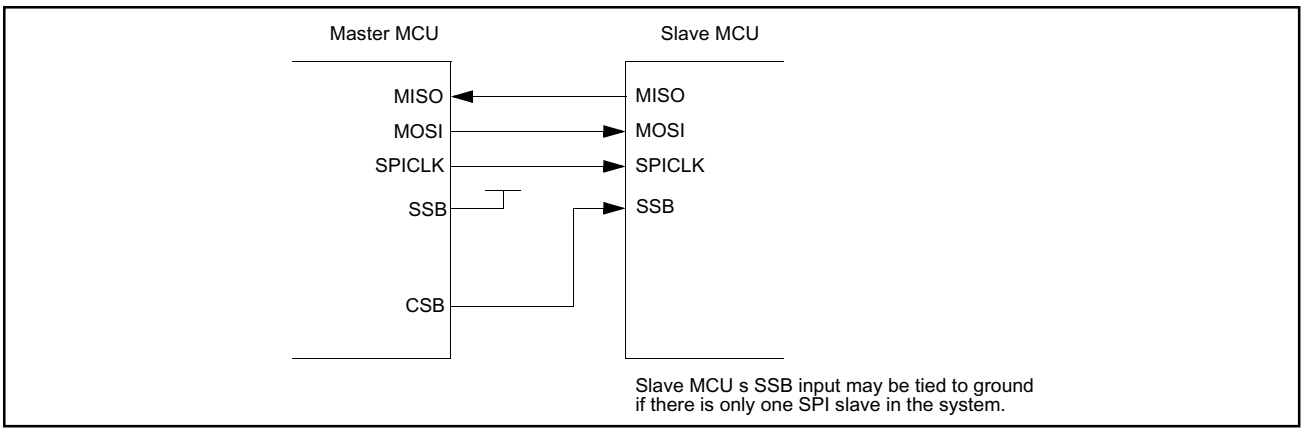



**Fig. 1.110. SPI system level view**

When operated in SPI mode the UART2 package pins provide the alternate SPI functions. MOSI, Master Out Slave In, is multiplexed on pin P7[0]/TxD2. MOSI outputs data when UART2 is a SPI master and inputs data when UART2 is a SPI slave. MISO, Master In Slave Out, is multiplexed on pin P7[1]/RxD2. MISO inputs data when UART2 is a SPI master and outputs data when UART2 is a SPI slave. SPICLK, SPI Clock, is multiplexed on pin P7[2]/CLK2. The SPI clock is input when the SPI is configured as a slave or output when the SPI is configured as a master. SSB, Slave select input, is multiplexed on pin P7[3]/RTSB/CTSB. This pin is used to select the active SPI slave.

The M30222 UART2 can be operated as an SPI master or as an SPI slave. Operation as an SPI Slave or SPI Master is determined by the CKDIR contol bit. While in SPI mode, the TxD and RxD pins act as the SPI MOSI and MISO pins. As implemented on the M30222, the SPI pins MOSI and MISO are open drain.

There are two added control bits and one added status flag. Control bit SPIM is the SPI Mode enable, which enables SPI operation. CPHA is the Clock Phase selection control bit. CPHA is used to chooses the clock to data relationship. Combined with the existing CKPOL control bit, CPHA provides compatibility with all four SPI transmission modes. CPHA is held at “0” when SPIM = “0”. The status flag MDFLT is used to indicate that an SPI mode fault occurred.

Several existing configuration bits are required for SPI operation. SPI Slave / Master mode is controlled by the existing CKDIR control bit. The UART is in SPI master mode when the clock is generated internally and is in SPI slave mode when the clock is generated externally. CKPOL control bit select the

polarity of the transfer clock. The four different combinations of CKPOL and CPHA define the four formats of the SPI communication protocols. While in SPI mode, the CRD control bit enables the $\overline{CTS}$/$\overline{RTS}$ pin to operate as SSB and CRS control bit selects $\overline{CTS}$/$\overline{RTS}$ pin to operate as $\overline{CTS}$/ SSB. Both contorl bits must be properly configured to activate the SSB function. UFORM control bit selects the UART transfer format, MSB or LSB. SPI data is transmitted MSB first.

### SPI operation

Setting SMD[2:0]=010 in U2MR enables either SPI or IIC operation. Operation is undefined if both IICM (U2SMR[0]) and SPI (U2SMR3[0]) control bits are set to logic one while SMD[2:0]=010.

Setting the SPIM control bit puts the UART2 into an SPI compatible mode. This mode is only valid in the clock synchronous configuration and must not be entered when the UART2 is configured for asynchronous operation. The internal / external clock select bit (CKDIR) in U2MR) determines whether UART2 is an SPI master or slave. If internal clock is selected, the UART is an SPI master and if external clock is selected, UART2 is an SPI slave. Figure 1.111 shows the signal wave forms.

Entering SPI mode has the following effects on operation:

(1) An alternate clock to data relationship can be chosen with the CPHA bit (in U2SMR3). This bit can only be set when SPI bit is a “1”. All four SPI clock to data formats are possible by using the CPHA bit together with the CKPOL bit (in U2C0). Figure 1.112 shows the function block diagram of SPI mode.

(2) The RxD pin becomes the MISO pin.

(3) The TxD pin becomes the MOSI pin.

(4) P7[3]/$\overline{CTSB}$/$\overline{RTSB}$ functions as the Slave Select input. This input is active low.

(5) When configured as a master, a Mode Fault will be detected if the Slave Select input goes low. If no port pin is assigned to be a slave select input, then mode fault detection is disabled.

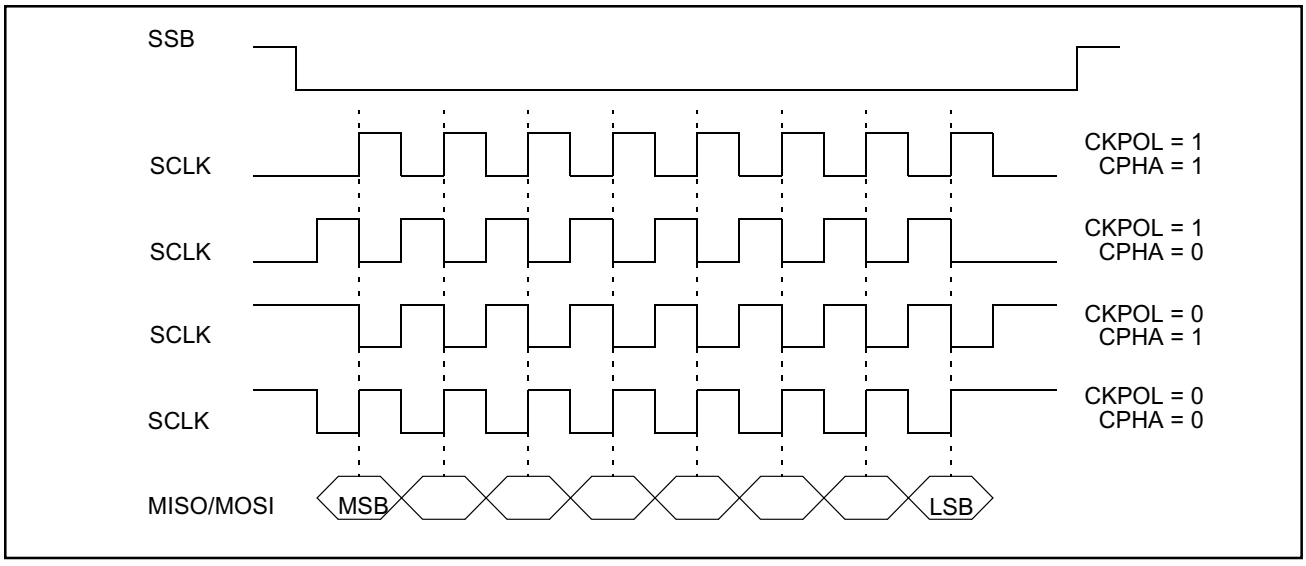



**Fig. 1.111. SPI Transmission formats**

NOTE: To prevent spurious clock transitions, configure the SPI modules as master or slave before enabling them. Enable the master before enabling the slave. Disable the slave before disabling the master.

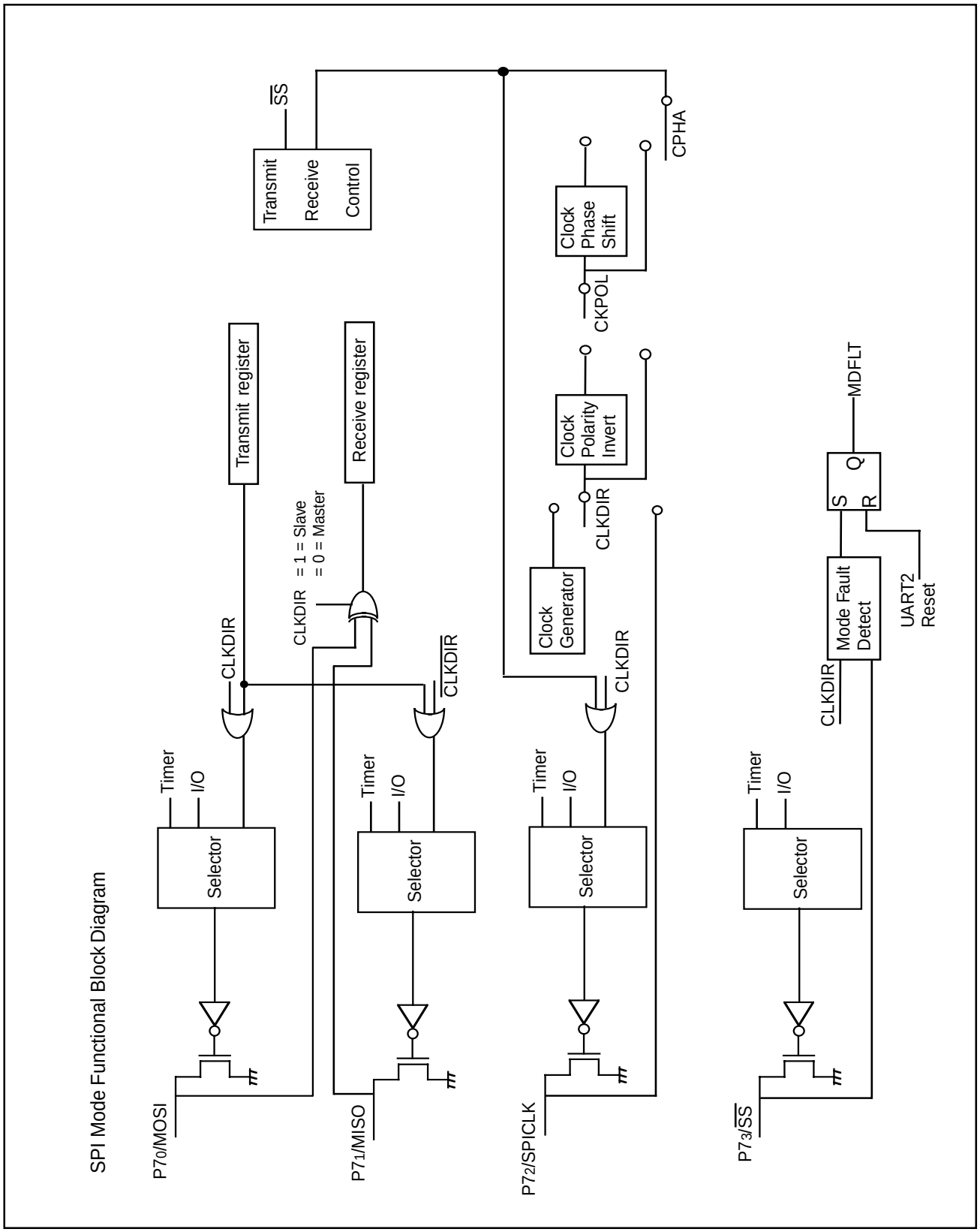



**Fig. 1.112 Functional block diagram of SPI mode**

### Master Mode operation

SPI Master mode is entered by setting both SPI and CKDIR control bits to logic one. In master mode, the UART will generate the clock to be driven on SPICLK. Transmitted data is shifted out on MOSI and and receive data is shifted in on MISO.

The mode fault status flag, MDFLT, is set any time the state of the slave select pin, SS, is inconsistent with an active SPI mode, SPIMSTR or SPISLV. Detection is intended to protect the MCU from damage due to output driver contention.

A mode fault occurs if the SS pin of a slave SPI goes high during a transmissionor if the SS pin of a master SPI goes low at any time A Mode Fault causes the following:

(1) The CLKDIR bit (in U2MR is forced to a “1”. This puts the UART into slave mode so that it is not driving the CLK and MOSI pins.
(2) The UART is inhibited from driving it’s MISO pin.
(3) Mode Fault, MDF, status bit (bit 10 of U2RB) is set. A UART2 receiver interrupt is generated.

Mode Fault detection is disabled when the CTS2 /SSB function is not assigned to a port pin. In this case, Slave Select is internally negated if the UART is configured for SPI Master operation.
A mode fault is cleared by setting the serial I/O mode bits (bits 2 through 0 of U2MR) to “000”. Also, the Receiver Enable bit (RE2 of U2C1) must be cleared. When the Mode Fault is cleared, the UART will return to master mode unless the CRS bit (in U2C0) is explicitly set.

### Slave mode operation

SPI Slave mode is entered by setting the SPI control bit to logic one and the CKDIR control bit to logic zero. Before transmission can start, the SSB pin of the slave SPI must be at logic zero.

When configured as a SPI slave, UART2 does not initiate any serial transfers. All transfers are initiated by an external SPI bus master.

When the CPHA bit is a “1”, serial transfers begin with the falling edge of Slave Select. For CPHA = “0”, serial transfers begin when the CLK leaves it’s idle state (the clock idle state is defined by the CKPOL bit in U2C0). If the UART transmit buffer is empty when a serial transfer starts, the UART will drive the value “80” hexadecimal on it’s MISO pin. The SPI should only write to the transmit buffer when it is empty. If the transmit buffer is written during a serial transfer, the new data will be loaded into the transmit shifter at the end of the current transfer.

The Slave Select function, SSB, is multiplexed on pin P7[3] along with CTSB. When UART2 is configured for SPI operation, the SSB function must be selected by setting the U2C0 CRD bit to logic “0’ to enable CTS/RTS functionality and additionally the U2C0 CRS bit must be set to logic “0” to enable CTS functionality. If the CTS function is not both selected and enabled, the UART2 SPI logic will internally hold the SSB signal to the appropriate level dependant on whether the SPI is configured for master or slave operation.

Slave select has various functions depending on the current state of the SPI. For an SPI configured as a slave, the SSB pin is used to select a slave. For CPHA=0, SSB is also used to indicate the start of a transmission. Since it is used to indicate the start of a transmission, SSB must be toggled high and low between each byte transmitted for the CPHA=0 mode For CPHA=1 format, SSB may be kept asserted low between transmitted bytes. If SSB is asserted while the SPI is configured as a master, a Mode Fault occurs.

## S I/O 3, 4

S I/O 3 and S I/O 4 are exclusive clock-synchronous serial I/Os.

Figure 1.113 shows the S I/Oi block diagram and Figure 1.114 shows the S I/Oi control register. Table 1.44 shows the specifications of S I/Oi.

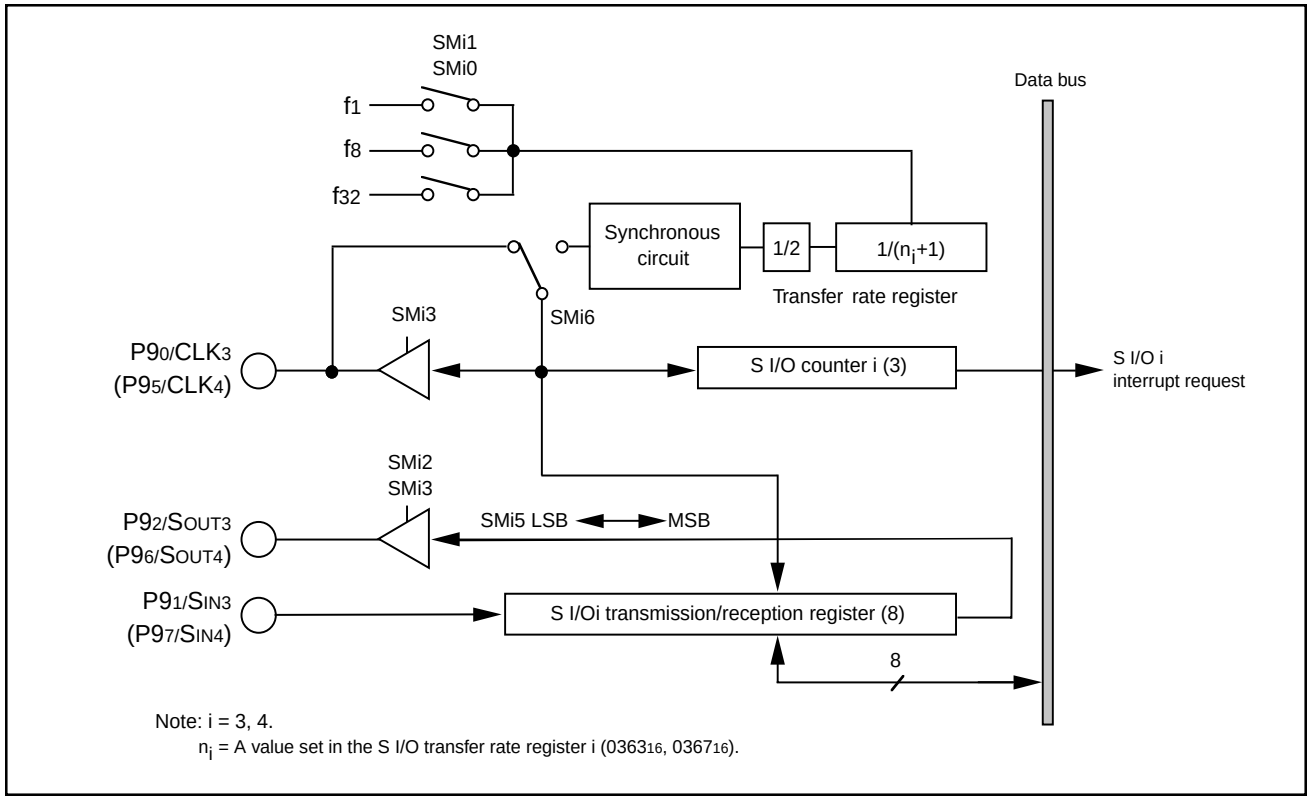
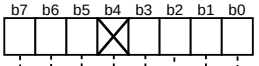



**Fig. 1.113. S I/Oi block diagram (i = 3,4)**

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

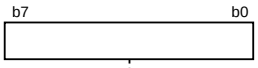



| Symbol | Address | When reset |
|---|---|---|
| SiC | $036216$, $036616$ | $4016$ |

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

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

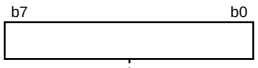
SI/Oi bit rate generator




| Symbol | Address | When reset |
|---|---|---|
| S3BRG | $036316$ | Indeterminate |
| S4BRG | $036716$ | Indeterminate |

| Indeterminate | Values that can be set | R | W |
|---|---|---|---|
| Assuming that set value = n, BRGi divides the count source by n + 1 | $0016$ to $FF16$ | × | ○ |

SI/Oi transmit/receive register




| Symbol | Address | When reset |
|---|---|---|
| S3TRR | $036016$ | Indeterminate |
| S4TRR | $036416$ | Indeterminate |

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

**Fig. 1.114. S I/O 3, 4 related registers**

**Table 1.44. Specifications of S I/O 3, 4**

| Item | Specifications |
|---|---|
| Transfer data format | • Transfer data length: 8 bits |
| Transfer clock | • With the internal clock selected (bit 6 of $0362_{16}$, $0366_{16}$ = "1"): f1/2(ni+1), f8/2(ni+1), f32/2(ni+1) (Note 1)<br>• With the external clock selected (bit 6 of $0362_{16}$, $0366_{16}$ = 0): Input from the CLKi terminal (Note 2) |
| Conditions for transmit/receive start | • To start transmission/reception, the following requirements must be met:<br>- Select the synchronous clock (use bit 6 of $0362_{16}$, $0366_{16}$).<br>Select a frequency dividing ratio if the internal clock has been selected (use bits 0 and 1 of $0362_{16}$, $0366_{16}$).<br>- SOUTi initial value set bit (use bit 7 of $0362_{16}$, $0366_{16}$)= 1.<br>- S I/Oi port select bit (bit 3 of $0362_{16}$, $0366_{16}$) = 1.<br>- Select the transfer direction (use bit 5 of $0362_{16}$, $0366_{16}$)<br>• If an internal clock is selected, set the bit rate generator divisor ($0363_{16}$, $0367_{16}$) [It is not necessry to start transmit/receive. It is only needed for operation as intended]<br>-Write transfer data to SI/Oi transmit/receive register ($0360_{16}$, $0364_{16}$)<br>• To use S I/Oi interrupts, the following requirements must be met:<br>- Clear the SI/Oi interrupt request bit before writing transfer data to the SI/Oi transmit/receive register (bit 3 of $0049_{16}$, $0048_{16}$) = 0. |
| Interrupt request generation timing | • Rising edge of the last transfer clock. (Note 3) |
| Select function | • LSB first or MSB first selection<br>Whether transmission/reception begins with bit 0 (LSB) or bit 7 (MSB) can be selected.<br>• Function for setting an SOUTi initial value selection<br>When using an external clock for the transfer clock, the user can choose the SOUTi pin output level during a non-transfer time. For details on how to set, see Figure 1.112. |
| Precaution | • Unlike UART0–2, SI/Oi (i = 3, 4) is not divided for transfer register and buffer. Therefore, do not write the next transfer data to the SI/Oi transmit/receive register (addresses $0360_{16}$, $0364_{16}$) during a transfer. When the internal clock is selected for the transfer clock, SOUTi holds the last data for a 1/2 transfer clock period after it finished transferring and then goes to a high-impedance state. However, if the transfer data is written to the SI/Oi transmit/receive register (addresses $0360_{16}$, $0364_{16}$) during this time, SOUTi is placed in the high-impedance state immediately upon writing and the data hold time is thereby reduced. |

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

Note 2: With the external clock selected:

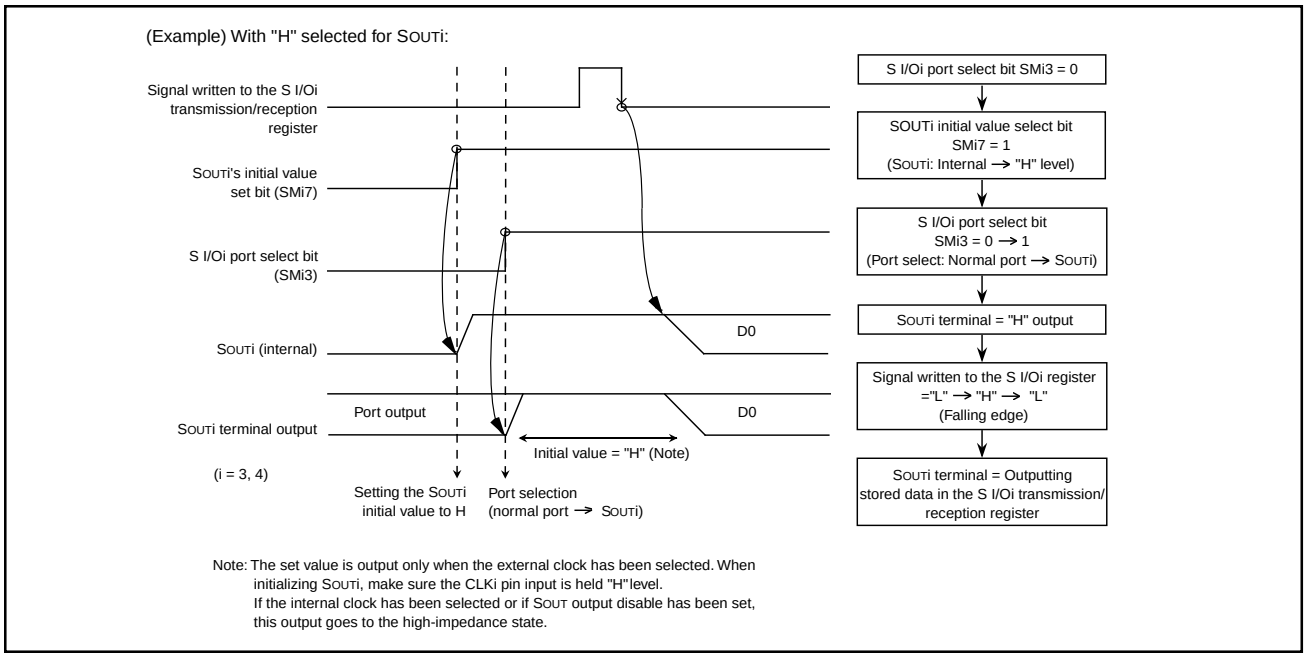
•Before data can be written to the SI/Oi transmit/receive register (addresses $0360_{16}$, $0364_{16}$), the CLKi pin input must be in the low state. Also, before rewriting the SI/Oi Control Register (addresses $0362_{16}$, $0366_{16}$)'s bit 7 (SOUTi initial value set bit), make sure the CLKi pin input is held low.

• The S I/Oi circuit keeps on with the shift operation as long as the synchronous clock is entered in it. Therefore, stop the synchronous clock immediately when count reaches eight. If selected, the internal clock stops automatically clocking the SIO channel.

Note 3: If the internal clock is used for the synchronous clock, the transfer clock signal output, when enabled, stops at the "H" state after transmission is completed.

### Functions for setting an SOUTi initial value

When using an external clock for the tranfer clock, the SOUTi pin output level during a non-transfer time can be set to the high or the low state. Figure 1.115 shows the timing chart for setting an SOUTi initial value.

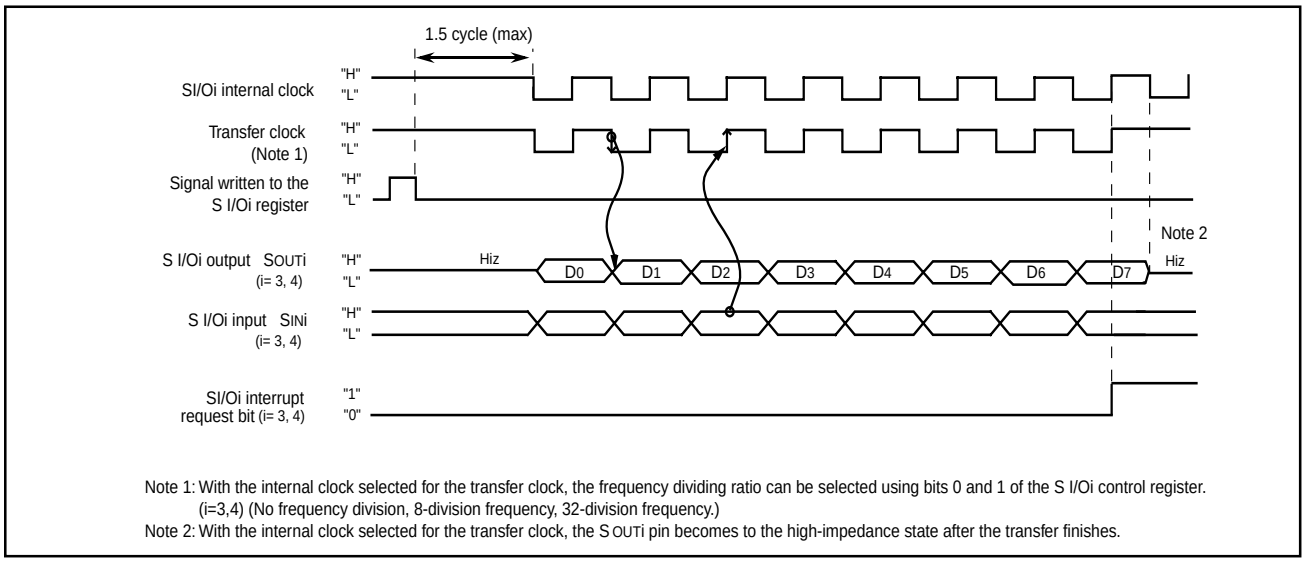



Note: The set value is output only when the external clock has been selected. When initializing SOUTi, make sure the CLKi pin input is held "H" level.
If the internal clock has been selected or if SOUT output disable has been set, this output goes to the high-impedance state.

**Fig. 1.115. Timing chart for setting SOUTi's initial value**

### S I/Oi operation timing

Figure 1.116 shows the S I/Oi operation timing.




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

**Fig. 1.116. S I/Oi operation timing chart**

## LCD Drive Control Circuit

The M30222 group has the built-in Liquid Crystal Display (LCD) drive control circuit consisting of the following.

- LCD display RAM
- Segment output enable register
- LCD mode register
- Voltage multiplier
- Selector
- Timing controller
- Common driver
- Segment driver
- Bias control circuit

A maximum of 40 segment output pins and 4 common output pins can be used. This allows up to 160 LCD pixels to be controlled. If static drive is enabled, up to 14 of the 40 multiplexed segment pins can be assigned to static drive.

The LCD drive control circuit automatically reads the LCD display ram, performs bias and duty ratio control, and displays the data on the LCD panel. The circuit is configured by writing to the LCD mode register, the segment output enable register, the LCD display ram, the LCD frame frequency counter, the LCD expansion register, and the LCD clock divided register. After all these registers are written then the LCD is turned on by setting the LCD enable bit to "1".

The LCDRAM output function allows the LCD segment output pins to be used as a general purpose output pin. This mode is configured by writing to the segment output register to enable the segment function, writing a "00" to the time division select bit and "0" to LCD output enable bit, and setting the LCDRAM output enable bit to "1". The data that is written to the LCDRAM bit 4 or 0 will be output on its corresponding segment pin. Note that in this mode VL3 & VL2 must be connected to VDD and VL1 must be connected to VSS.

Table 1.45 shows the maximum number of display pixels at each duty ratio.
Figure 1.117 shows the block diagram of LCD controller / driver. Figures 1.118 and 1.119 show the LCD-related registers.

**Table 1.45. Maximum number of display pixels at each duty ratio**

| Duty ratio | Maximum number of display pixels (multiplexed) | Static drive |
|---|---|---|
| 2 | 80 dots or 8 segment LCD 10 digits | OFF |
| 3 | 120 dots or 8 segment LCD 15 digits | OFF |
| 4 | 160 dots or 8 segment LCD 20 digits | OFF |

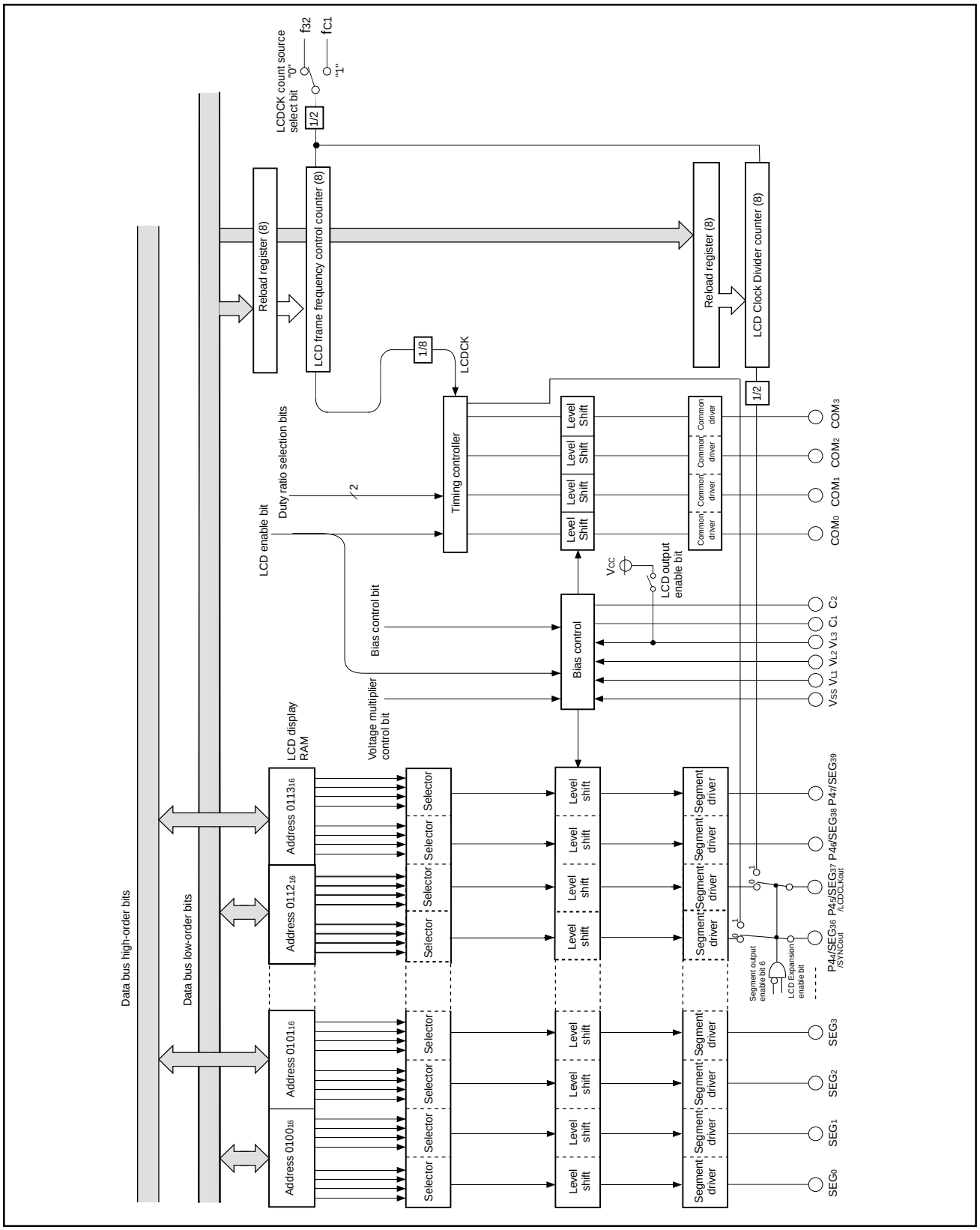



**Fig. 1.117. Block diagram of LCD controller/driver**

LCD mode register

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

Symbol: LCDM — Address: $012016$ — When reset: $0X000000_2$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| LCDT0 | Duty ratio select bit | b1 b0<br>0 0 : Not used<br>0 1 : 2 duty (use COM0, COM1)<br>1 0 : 3 duty (use COM0 - COM2)<br>1 1 : 4 duty (use COM0 - COM3) | ○ | ○ |
| LCDT1 | | | ○ | ○ |
| BIAS | Bias control bit | 0 : 1/3 bias<br>1 : 1/2 bias | ○ | ○ |
| LCDEN | LCD enable bit | 0 : LCD Off<br>1 : LCD On | ○ | ○ |
| PUMP | Voltage multiplier control bit | 0 : Voltage multipler disable<br>1 : Voltage multipler enable (Note 2) | ○ | ○ |
| LRAMOUT | LCDRAM output bit | 0 : LCD waveform output<br>1 : LCD data output | ○ | ○ |
| Nothing is assigned. Write "0" when writing to this bit. When read, the value is indeterminate. | | | – | – |
| LSRC | LCDCK count source select bit (Note 1) | 0 : f32<br>1 : fc1 | ○ | ○ |

Note 1: LCDCK is a clock for an LCD timing controller.
Note 2: When voltage multiplier is enabled, bias control bit must be "0".

Segment output enable register

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

Symbol: SEG — Address: $012216$ — When reset: $0016$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SEG00 | Segment output enable bit 0 | 0 : I/O ports $P3_0$ to $P3_2$<br>1 : Segment output $SEG_{24}$ to $SEG_{26}$ | ○ | ○ |
| SEG01 | Segment output enable bit 1 | 0 : I/O ports $P3_3$<br>1 : Segment output $SEG_{27}$ | ○ | ○ |
| SEG02 | Segment output enable bit 2 | 0 : I/O ports $P3_5$ to $P3_4$<br>1 : Segment output $SEG_{28}$ to $SEG_{29}$ | ○ | ○ |
| SEG03 | Segment output enable bit 3 | 0 : I/O ports $P3_7$ to $P3_6$<br>1 : Segment output $SEG_{30}$ to $SEG_{31}$ | ○ | ○ |
| SEG04 | Segment output enable bit 4 | 0 : I/O ports $P4_0$ to $P4_3$<br>1 : Segment output $SEG_{32}$ to $SEG_{35}$ | ○ | ○ |
| SEG05 | Segment output enable bit 5 | 0 : I/O ports $P4_4$ to $P4_5$<br>1 : Segment output $SEG_{36}$ to $SEG_{37}$ | ○ | ○ |
| SEG06 | Segment output enable bit 6 | 0 : I/O ports $P4_6$ to $P4_7$<br>1 : Segment output $SEG_{38}$ to $SEG_{39}$ | ○ | ○ |
| SEG07 | LCD output enable | 0 : Disable<br>1 : Enable | ○ | ○ |

LCD frame frequency counter

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

Symbol: LCDTIM — Address: $012416$ — When reset: Indeterminate

| Function | Values that can be set | R | W |
|---|---|---|---|
| 8 bit timer | $0016$ to $FF16$ (Note) | X | ○ |

Note: This register must be changed while LCD output enable bit is "0".

**Fig. 1.118. LCD related registers (1)**

LCD expansion register

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

Symbol: LEXP
Address: 013016
When reset: 0016

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| LEXPEN | LCD Expansion enable bit | 0 : Disable<br>1 : Enable | O | O |
| LCKPOL | LCD clock out and polarity select bit | 0 : SOF at rising edge<br>1 : SOF at falling edge | O | O |
| LSYNC | LCD SYNCout initiation bit | 0 : No action<br>1 : Initiate SYNC | O | O |
| Nothing is assigned.<br>These bits can neither be set nor reset. When read, the value is indeterminate. | | | – | – |
| LSTATCNF0 | LCD static drive pin configuration select bits | 0 0 : SEG35 to SEG24 static<br>0 1 : SEG35 to SEG27 static<br>1 0 : SEG35 to SEG32 static<br>1 1 : Do not use | O | O |
| LSTATCNF1 | | | O | O |
| LSTATEN | LCD static drive enable bit | 0 : Disable<br>1 : Enable | O | O |

LCD clock divide counter

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

Symbol: LCDC
Address: 013216
When reset: Indeterminate

| Function | Values that can be set | R | W |
|---|---|---|---|
| 8 bit timer | 0016 to FF16 | X | O |

**Fig. 1.119. LCD expansion related register (2)**

## Voltage Multiplier

The voltage multiplier performs threefold boosting. This circuit inputs a reference voltage for boosting from LCD power input pin $V_{L1}$. (However, when using a 1/2 bias, supply power to the $V_{L1}$ and $V_{L2}$ through an external resistor divider.)

To activate the voltage multiplier, choose the segment/port and duty rate, select bias control, and set up the LCD frame frequency counter and LCDCK count source using the segment enable register and LCD mode register, then enable the LCD output enable bit (bit 7 at address 012216) and set the voltage multiplier control bit (bit 4 at address 012016) to “1” (= voltage multiplier enabled). When voltage is input to the $V_{L1}$ pin during operating the voltage multiplier, voltage that is twice as large as $V_{L1}$ occurs at the $V_{L2}$ pin, and voltage that is three times as large as $V_{L1}$ occurs at the $V_{L3}$ pin.

The voltage multiplier control bit (bit 4 of the address 012016) controls the voltage multiplier. When using the voltage multiplier, apply a voltage equal to or greater than 1.3 V but not exceeding 2.1 V to the $V_{L1}$ pin before enabling the voltage multiplier control bit.

When not using the voltage multiplier, enable the LCD output enable bit and apply an appropriate voltage to the LCD power supply input pins ($V_{L1}$ to $V_{L3}$). When the LCD output enable bit is disabled, the $V_{L3}$ pin is connected to $V_{CC}$ internally.

### Bias Control and Applied Voltage to LCD Power Input Pins

To the LCD power input pins ($V_{L1}$ to $V_{L3}$), apply the voltage shown in Table 1.46 according to the bias value. Select a bias value by the bias control bit (bit 2 of the address $0120_{16}$). Figure 1.120 is an example of circuit at each bias.

**Table 1.46. Bias control and applied voltage to VL1 to VL3**

| Bias value | Voltage value |
|---|---|
| 1/3 bias | VL3 = VLCD<br>VL2 = 2/3 VLCD<br>VL1 = 1/3 VLCD |
| 1/2 bias | VL3 - VLCD<br>VL2 = VL1 = 1/2 VLCD |

Note: VLCD is the maximum value of supplied voltage for the LCD panel.

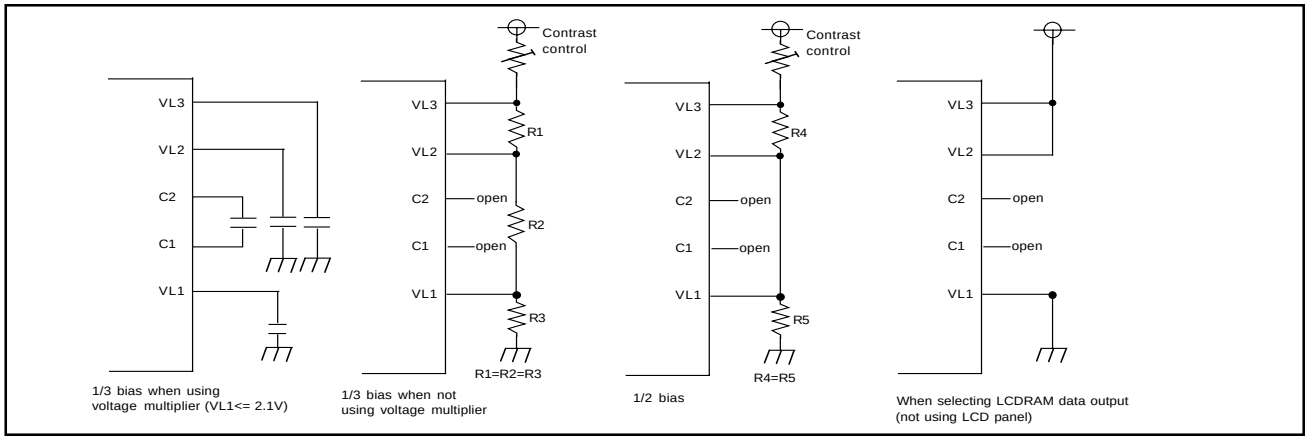



**Fig. 1.120. Example of circuit at each bias**

### Common Pin and Duty Control

The common pins ($COM_0$ to $COM_3$) to be used are determined by the required duty. Table 1.47shows the duty control and the common pins used. Select duty ratio by the duty ratio select bits (bits 0 and 1 of address $0120_{16}$).

**Table 1.47. Duty ratio control and common pins used**

| Duty ratio | Daily ratio select bit | | Common pins used |
|---|---|---|---|
| | Bit 1 | Bit 0 | |
| 2 | 0 | 1 | COM0, COM1 (Note 1) |
| 3 | 1 | 0 | COM0 to COM2 (Note 2) |
| 4 | 1 | 1 | COM0 to COM3 |

Note 1: COM2 and COM3 are open.
Note 2: COM 3 is open.

## LCD Display RAM

Address $0100_{16}$ to $0113_{16}$ is the designated RAM for the LCD display. When “1's" are written to these addresses, the corresponding segments of the LCD display panel are turned on. Table 1.48 shows the LCD display RAM map.

**Table. 1.48. LCD display RAM map**

| COM | 3 | 2 | 1 | 0 | 3 | 2 | 1 | 0 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Bit / Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | R | W |
| $0100_{16}$ | SEG1 | | | | SEG0 | | | | O | O |
| $0101_{16}$ | SEG3 | | | | SEG2 | | | | O | O |
| $0102_{16}$ | SEG5 | | | | SEG4 | | | | O | O |
| $0103_{16}$ | SEG7 | | | | SEG6 | | | | O | O |
| $0104_{16}$ | SEG9 | | | | SEG8 | | | | O | O |
| $0105_{16}$ | SEG11 | | | | SEG10 | | | | O | O |
| $0106_{16}$ | SEG13 | | | | SEG12 | | | | O | O |
| $0107_{16}$ | SEG15 | | | | SEG14 | | | | O | O |
| $0108_{16}$ | SEG17 | | | | SEG16 | | | | O | O |
| $0109_{16}$ | SEG19 | | | | SEG18 | | | | O | O |
| $010A_{16}$ | SEG21 | | | | SEG20 | | | | O | O |
| $010B_{16}$ | SEG23 | | | | SEG22 | | | | O | O |
| $010C_{16}$ | SEF25 | | | | SEG24 | | | | O | O |
| $010D_{16}$ | SEG27 | | | | SEG26 | | | | O | O |
| $010E_{16}$ | SEG29 | | | | SEG28 | | | | O | O |
| $010F_{16}$ | SEG31 | | | | SEG30 | | | | O | O |
| $0110_{16}$ | SEG33 | | | | SEG32 | | | | O | O |
| $0111_{16}$ | SEG35 | | | | SEG34 | | | | O | O |
| $0112_{16}$ | SEG37 | | | | SEG36 | | | | O | O |
| $0113_{16}$ | SEG39 | | | | SEG38 | | | | O | O |

## LCD Drive Timing

The LCDCK timing frequency (LCD drive timing) is generated internally and the frame frequency can be determined with the following equation. The LCDCK count source frequency is $f_{C1}$ (same frequency as $X_{CIN}$) or $f_{32}$ (divide-by-32 of $X_{IN}$ frequency). Figure 1.121 shows the LCD drive waveform (1/2 bias). Figure 1.122 shows the LCD drive waveform (1/3 bias).

$$f(LCDCK) = \frac{\text{(frequency of count source for LCDCK)}}{\text{16 X (LCD frame frequency count value + 1)}}$$

$$\text{Frame frequency} = \frac{f(LCDCK)}{\text{duty ratio}}$$

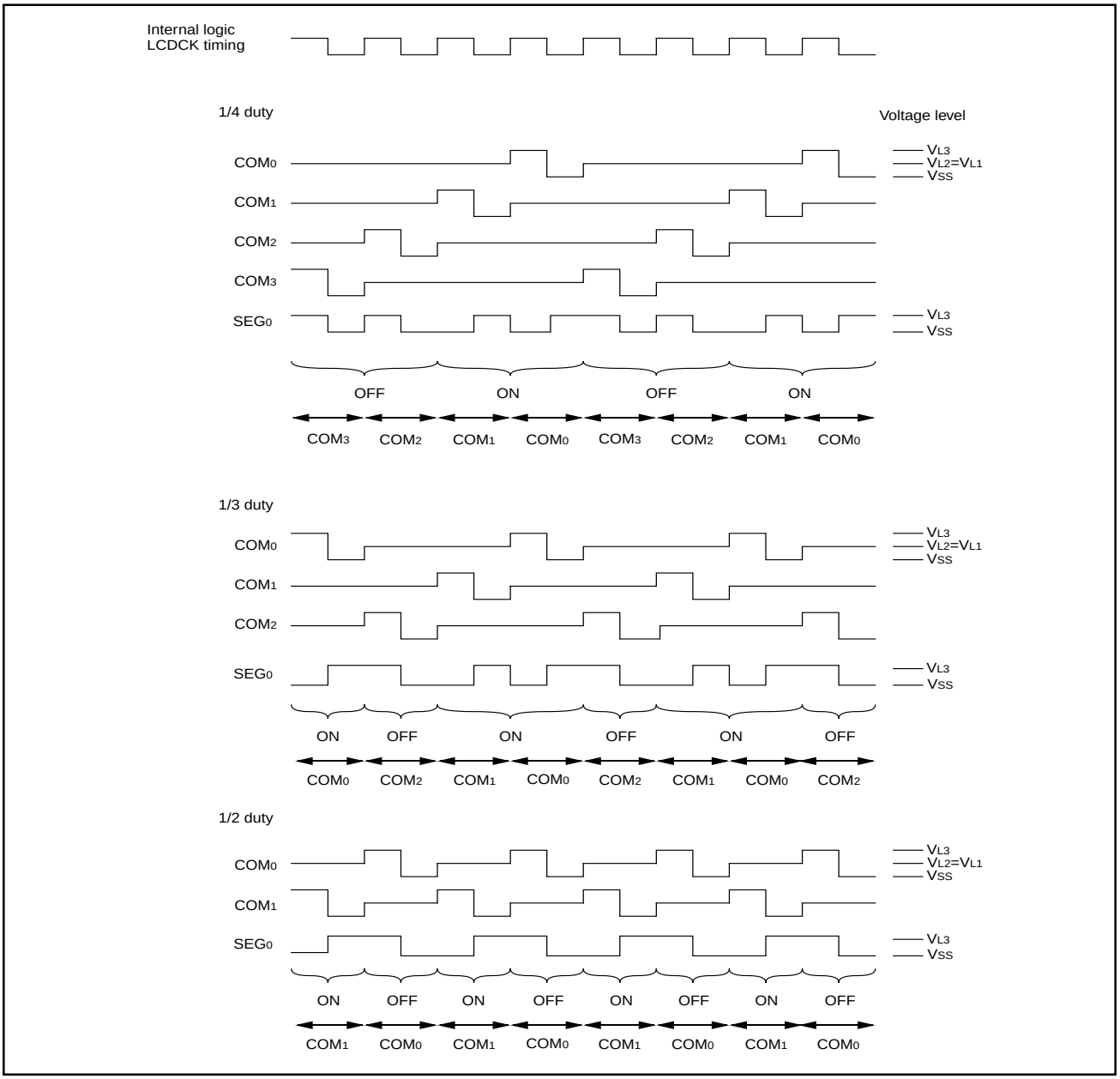



**Fig. 1.121. LCD drive waveform (1/2 bias)**

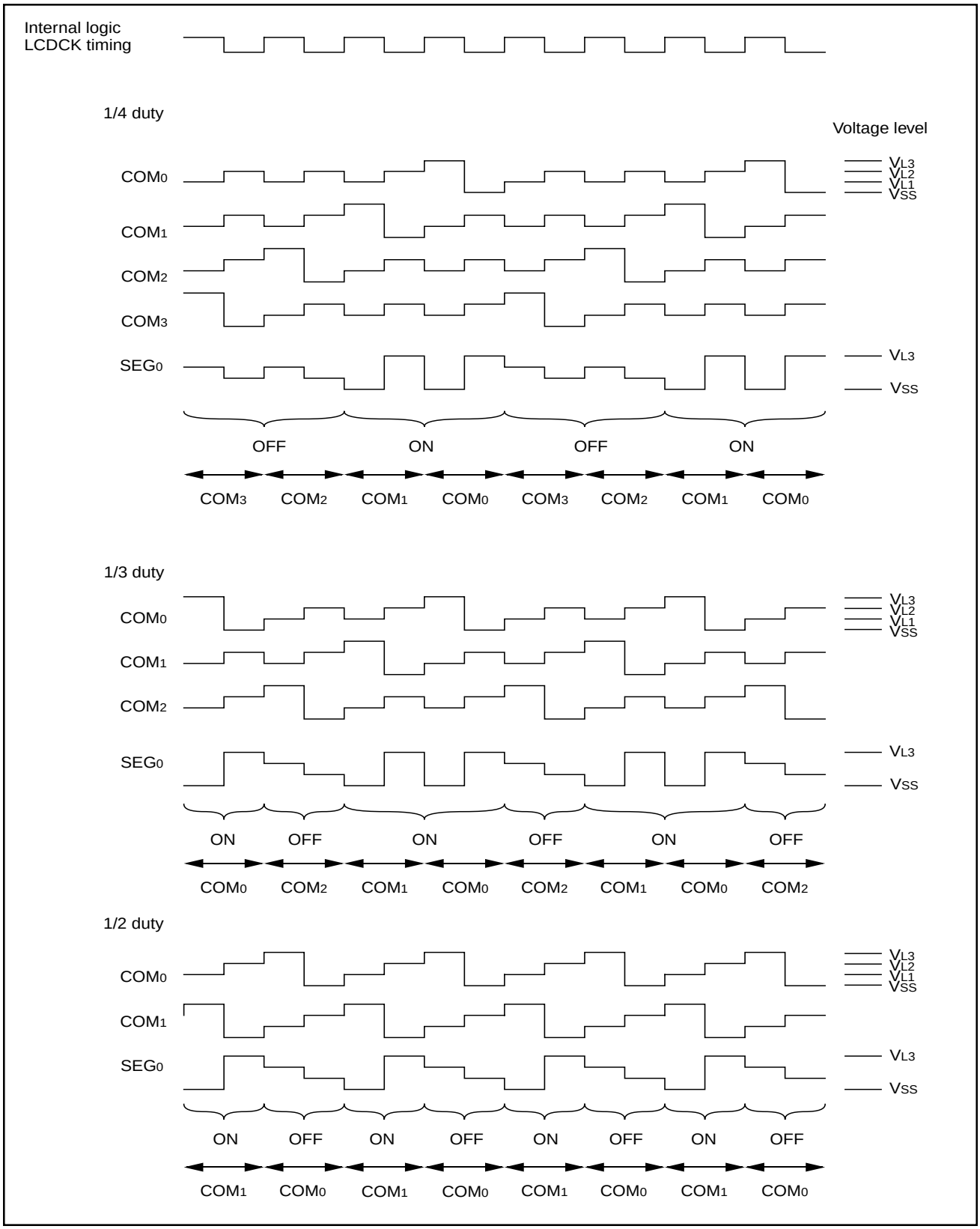



**Fig. 1.122. LCD drive waveform (1/3 bias)**

### LCD Static Drive

When bit 7 of the LCD expansion register is set, then the static drive function is enabled.
The following LCD pins get assigned the static drive function in one of the following groups: SEG35 to SEG24, SEG35 to SEG27, or SEG35 to SEG32. Bits 6 and 5 of the LCD expansion register determine the grouping of the static drive pins. All remaining LCD pins retain their normal LCD functions.

The waveforms that are output from the static drive pins are shown in Figure 1.123. When writing to LCD RAM for static operation, write the data to all four bits assigned to that pin. (Refer to Table 1.50, LCD display RAM map). For example, to set static $SEG_{34}$ ON and $SEG_{35}$ OFF, write $F0_{16}$ to address $0111_{16}$.

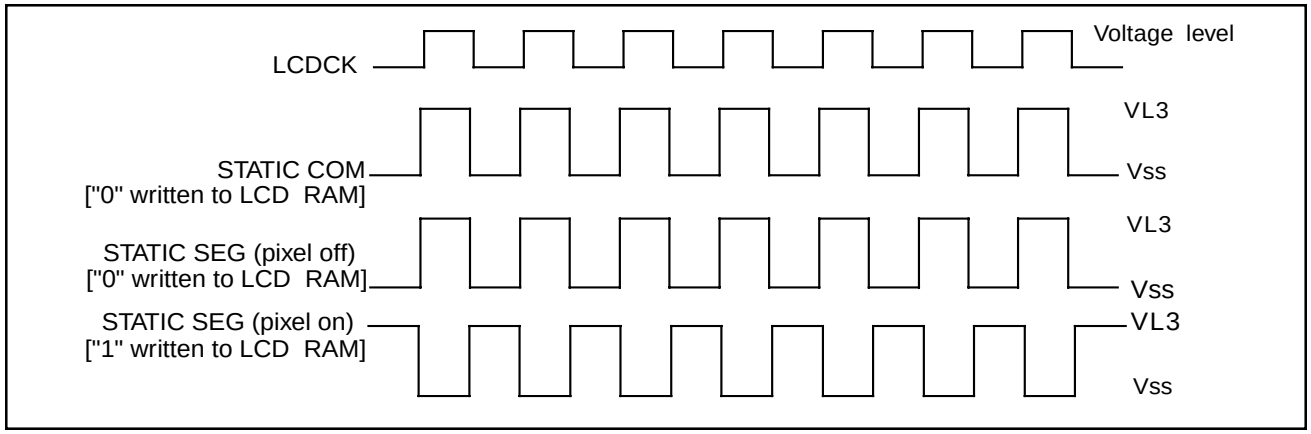



**Fig. 1.123. LCD drive waveform (static drive)**

### LCD Expansion Clock

When bit 7 of the LCD expansion register is set, the LCD expansion clock becomes active. In this mode a clock (LCDCKout) synchronized to the internal LCD clock can be output from the mcu. The frequency of LCDCLKout is set by the LCD clock divided counter and is:

$$f(LCDCKout) = \frac{(\text{frequency of count source for LCDCK})}{2 \times (\text{LCD clock divided counter} + 1)}$$

In addition, a synchronization signal ($\overline{SYNCout}$) is output. When the LCD $\overline{SYNCout}$ initiation bit is set, this signal will go active low at the beginning of the next LCD frame.

Refer to Figures 1.124 and 1.125.

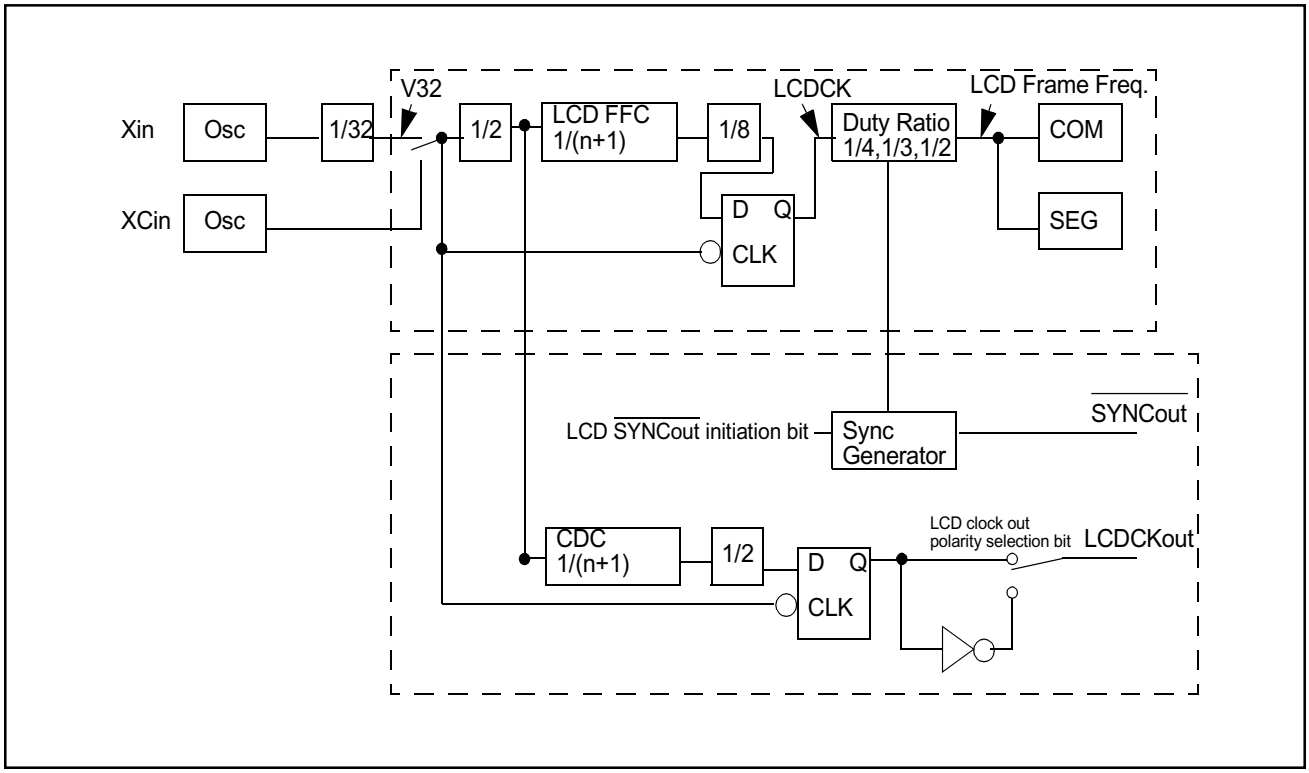



**Fig. 1.124. LCD port expansion block diagram**

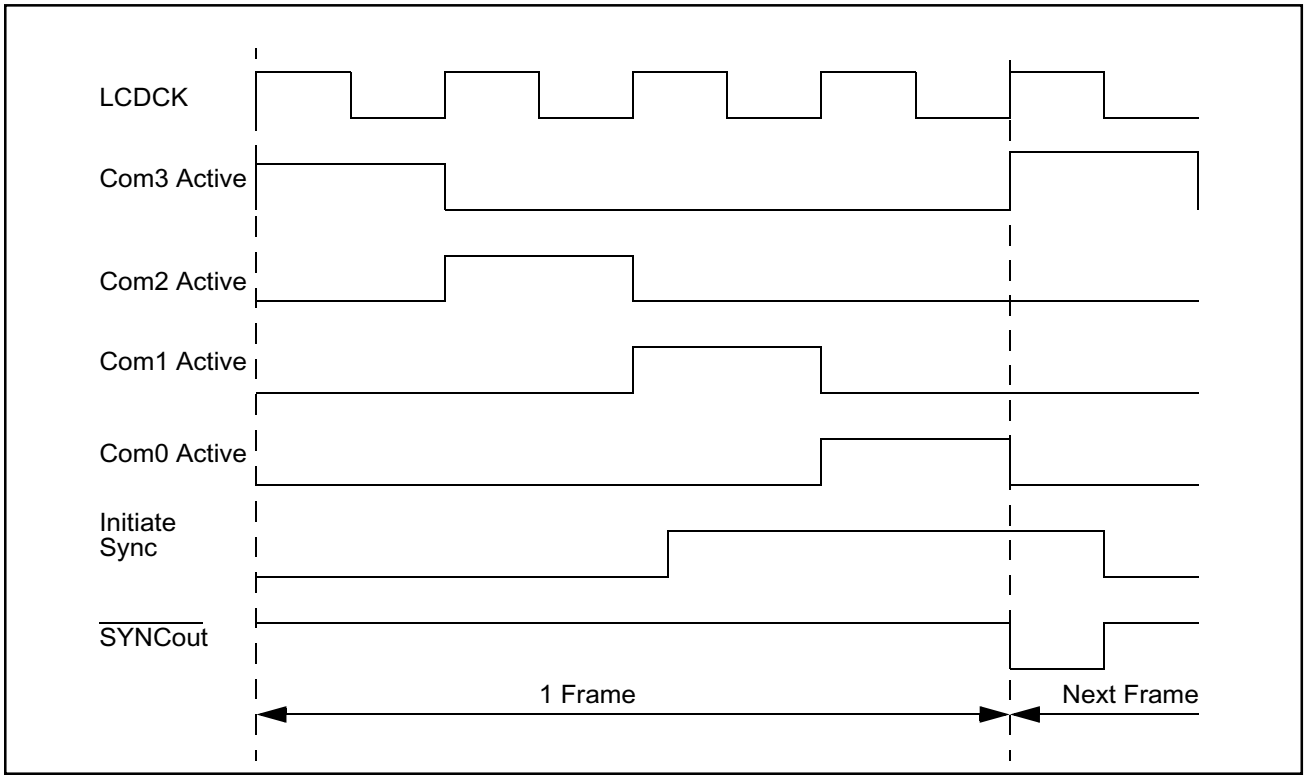



**Fig. 1.125. LCD port expansion timing diagram**

## A-D Converter

The A-D converter consists of one 10-bit successive approximation A-D converter circuit with a capacitive coupling amplifier. Pins $P10_0$ to $P10_7$, $P9_5$, and $P9_6$ also function as the analog signal input pins. The direction registers of these pins for A-D conversion must therefore be set to input. The $V_{REF}$ connect bit (bit 5 at address $03D7_{16}$) can be used to isolate the resistance ladder of the A-D converter from the reference voltage input pin ($V_{REF}$) when the A-D converter is not used. Doing so stops any current flowing into the resistance ladder from $V_{REF}$, reducing the power dissipation. When using the A-D converter, start A-D conversion only after setting bit 5 of $03D7_{16}$ to connect $V_{REF}$.

The result of A-D conversion is stored in the A-D registers of the selected pins. When set to 10-bit precision, the lower 8 bits are stored in the even addresses and the high 2 bits in the odd addresses. When set to 8-bit precision, the low 8 bits are stored in the even addresses.

Table 1.49 shows the performance of the A-D converter. Figure 1.126 shows the block diagram of the A-D converter, and Figures 1.127 and 1.128 show the A-D converter-related registers.

**Table 1.49. Performance of A-D converter**

| Item | Performance |
|---|---|
| Method of A-D conversion | Successive approximation (capacitive coupling amplifier) |
| Analog input voltage (Note 1) | 0V to $AV_{CC}$ ($V_{CC}$) |
| Operating dock $\phi_{AD}$ (Note 2) | $V_{CC}$ = 5V $f_{AD}$/divide-by-2 of $f_{AD}$/divide-by-4 of $f_{AD}$, $f_{AD}=f(X_{IN})$<br>$V_{CC}$ = 3V divide-by-2 of $f_{AD}$/divide-by-4 of $f_{AD}$, $f_{AD}=f(X_{IN})$ |
| Resolution | 8-bit or 10-bit (selectable) |
| Absolute precision | $V_{CC}$ = 5V • Without sample and hold function<br>±3LSB<br>• With sample and hold function (8-bit resolution)<br>±2LSB<br>• With sample and hold function (10-bit resolution)<br>$AN_0$ to $AN_7$ input : ±3LSB<br>ANEX0 and ANEX1 input (including mode in which external operation amp is connected) : ±7LSB<br>$V_{CC}$ = 3V • Without sample and hold function (8-bit resolution)<br>±2LSB to 1MHz |
| Operating modes | One-shot mode, repeat mode, single sweep mode, repeat sweep mode 0, and repeat sweep mode 1 |
| Analog input pins | 8 pins ($AN_0$ to $AN_7$) + 2pins (ANEX0 and ANEX1) |
| A-D conversion start condition | • Software trigger<br>A-D conversion starts when the A-D conversion start flag changes to “1”<br>• External trigger (can be retriggered)<br>A-D conversion starts when the A-D conversion start flag is “1” and the $ADTRG/P9_7$ input changes from “H” to “L” |
| Conversion speed per pin | • Without sample and hold function<br>8-bit resolution: 49 $\phi_{AD}$ cycles, 10-bit resolution: 59 $\phi_{AD}$ cycles<br>• With sample and hold function<br>8-bit resolution: 28 $\phi_{AD}$ cycles, 10-bit resolution: 33 $\phi_{AD}$ cycles |

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

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

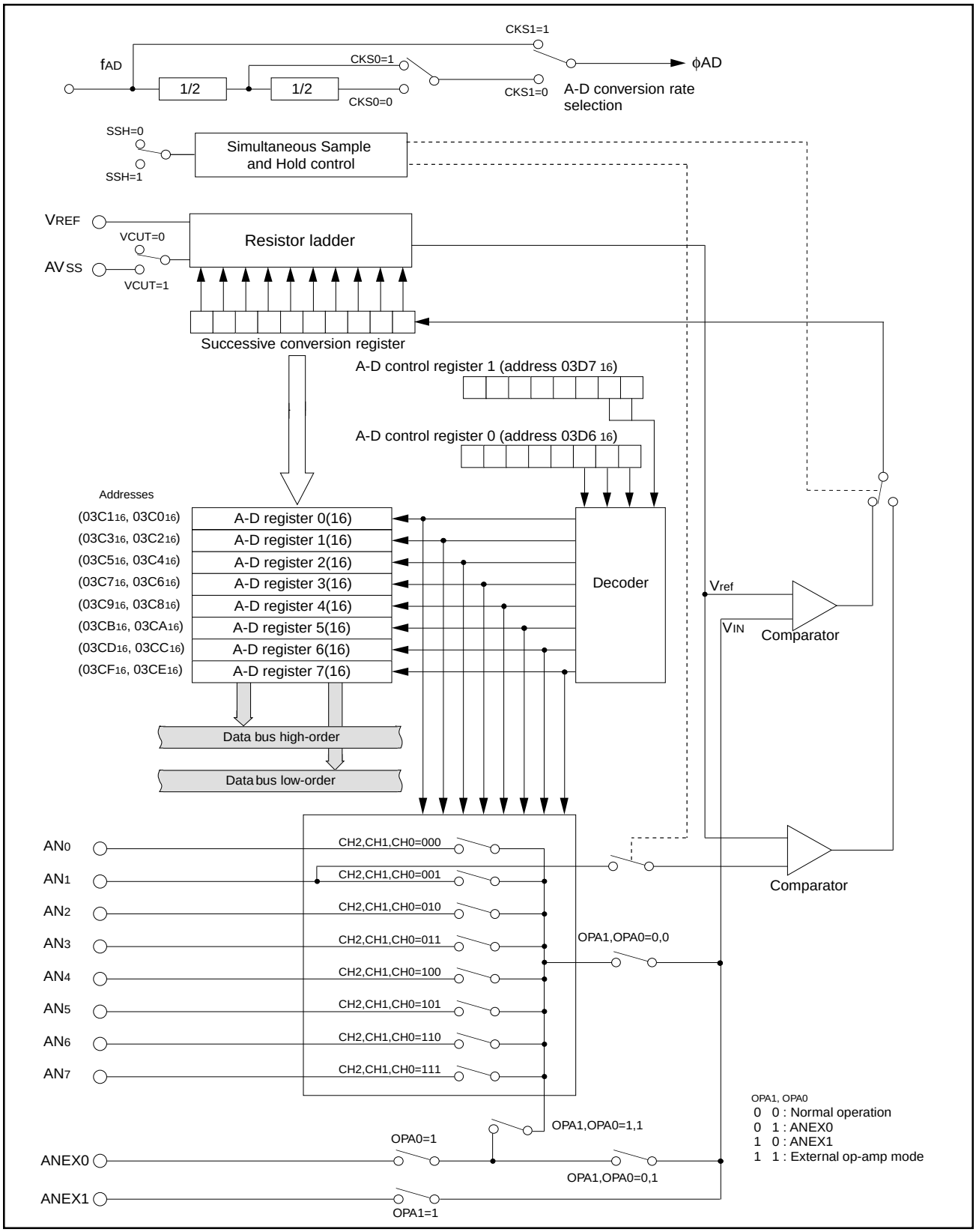
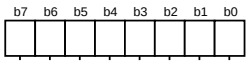



**Fig. 1.126. Block diagram of A-D converter**

A-D control register 0 (Note 1)

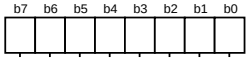



Symbol ADCON0 — Address 03D6$_{16}$ — When reset 00000XXX$_2$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CH0 | Analog input pin select bit | b2 b1 b0<br>0 0 0 : AN$_0$ is selected<br>0 0 1 : AN$_1$ is selected<br>0 1 0 : AN$_2$ is selected<br>0 1 1 : AN$_3$ is selected<br>1 0 0 : AN$_4$ is selected<br>1 0 1 : AN$_5$ is selected<br>1 1 0 : AN$_6$ is selected<br>1 1 1 : AN$_7$ is selected (Note 2) | ○ | ○ |
| CH1 | | | ○ | ○ |
| CH2 | | | ○ | ○ |
| MD0 | A-D operation mode select bit 0 | b4 b3<br>0 0 : One-shot mode<br>0 1 : Repeat mode<br>1 0 : Single sweep mode<br>1 1 : Repeat sweep mode 0<br>Repeat sweep mode 1 (Note 2) | ○ | ○ |
| MD1 | | | ○ | ○ |
| TRG | Trigger select bit | 0 : Software trigger<br>1 : ADTRG trigger | ○ | ○ |
| ADST | A-D conversion start flag | 0 : A-D conversion disabled<br>1 : A-D conversion started | ○ | ○ |
| CKS0 | Frequency select bit 0 | 0 : f$_{AD}$/4 is selected<br>1 : f$_{AD}$/2 is selected | ○ | ○ |

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

A-D control register 1 (Note)




Symbol ADCON1 — Address 03D7$_{16}$ — When reset 00$_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SCAN0 | A-D sweep pin select bit | When single sweep and repeat sweep mode 0 are selected<br>b1 b0<br>0 0 : AN$_0$, AN$_1$ (2 pins)<br>0 1 : AN$_0$ to AN$_3$ (4 pins)<br>1 0 : AN$_0$ to AN$_5$ (6 pins)<br>1 1 : AN$_0$ to AN$_7$ (8 pins) | ○ | ○ |
| SCAN1 | | When repeat sweep mode 1 is selected<br>b1 b0<br>0 0 : AN$_0$ (1 pin)<br>0 1 : AN$_0$, AN$_1$ (2 pins)<br>1 0 : AN$_0$ to AN$_2$ (3 pins)<br>1 1 : AN$_0$ to AN$_3$ (4 pins) | ○ | ○ |
| MD2 | A-D operation mode select bit 1 | 0 : Any mode other than repeat sweep mode 1<br>1 : Repeat sweep mode 1 | ○ | ○ |
| BITS | 8/10-bit mode select bit | 0 : 8-bit mode<br>1 : 10-bit mode | ○ | ○ |
| CKS1 | Frequency select bit 1 | 0 : f$_{AD}$/2 or f$_{AD}$/4 is selected<br>1 : f$_{AD}$ is selected | ○ | ○ |
| VCUT | Vref connect bit | 0 : Vref not connected<br>1 : Vref connected | ○ | ○ |
| OPA0 | External op-amp connection mode bit | b7 b6<br>0 0 : ANEX0 and ANEX1 are not used<br>0 1 : ANEX0 input is A-D converted<br>1 0 : ANEX1 input is A-D converted<br>1 1 : External op-amp connection mode | ○ | ○ |
| OPA1 | | | ○ | ○ |

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

**Fig. 1.127. A-D converter-related registers (1)**

A-D control register 2 (Note)

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

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

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SMP | A-D conversion method select bit | 0 : Without sample and hold<br>1 : With sample and hold | ○ | ○ |
| Reserved bit | | Always set to "0" | ○ | ○ |
| Nothing is assigned. Write "0" when writing to these bits. When read, the value is "0". | | | – | – |
| SSH | Simultaneous sample and hold | 0 : Disabled<br>1 : Enabled | ○ | ○ |

Note 1: If the A-D control register is rewritten during A-D conversion, the conversion result is indeterminate.
Note 2: SSH must be used ony in conjunction with SMP.
Note 3: User must guarantee a conversion of input "1" after a conversion of input "0" in all modes except sweep modes.

A-D register i

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

(b15) b7 (b8) b0 b7 b0

| Function | R | W |
|---|---|---|
| Eight low-order bits of A-D conversion result | ○ | × |
| • During 10-bit mode<br>Two high-order bits of A-D conversion result | ○ | × |
| • During 8-bit mode<br>When read, the content is indeterminate | × | × |
| Nothing is assigned. Write "0" when writing to these bits. When read, the value is "0". | – | – |

**Fig. 1.128. A-D converter-related registers (2)**

### (1) One-shot mode

In one-shot mode, the pin selected using the analog input pin select bit is used for one-shot A-D conversion. Table 1.50 shows the specifications of one-shot mode. Figure 1.129 shows the A-D control register in one-shot mode.

**Table 1.50. One-shot mode specifications**

| Item | Specification |
|---|---|
| Function | The pin selected by the analog input pin select bit is used for one-shot A-D conversion |
| Start condition | Writing 1 to A-D conversion start flag |
| Stop condition | Writing 0 to A-D conversion start flag<br>End of A-D conversion (A-D conversion start flag changes ot 0 , except when external trigger is selected). |
| Interrupt request generation timing | End of A-D conversion |
| Input pin | One of $AN_0$ to $AN_7$ as selected |
| Reading A-D converter results | Read A-D register corresponding to selected pin |

A-D control register 0 (Note 1)

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

Symbol ADCON0 — Address 03D6₁₆ — When reset 00000XXX₂

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

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

A-D control register 1 (Note)

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

Symbol ADCON1 — Address 03D7₁₆ — When reset 00₁₆

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

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

**Figure 1.129. A-D conversion register in one-shot mode**

## (2) Repeat mode

In repeat mode, the pin selected using the analog input pin select bit is used for repeated A-D conversion. Table 1.51 shows the specifications of repeat mode. Figure 1.130 shows the A-D control register in repeat mode.

**Table 1.51. Repeat mode specifications**

| Item | Specification |
|---|---|
| Function | The pin selected by the analog input pin select bit is used for repeated A-D conversion |
| Start condition | Writing 1 to A-D conversion start flag |
| Stop condition | Writing 0 to A-D conversion start flag |
| Interrupt request generation timing | None generated |
| Input pin | One of $AN_0$ to $AN_7$ as selected |
| Reading A-D converter results | Read A-D register corresponding to selected pin |

A-D control register 0 (Note 1)

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

Symbol ADCON0 — Address 03D6₁₆ — When reset 00000XX₂

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

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

A-D control register 1 (Note)

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

Symbol ADCON1 — Address 03D7₁₆ — When reset 00₁₆

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

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

**Fig. 1.130. A-D conversion register in repeat mode**

### (3) Single sweep mode

In single sweep mode, the pins selected using the A-D sweep pin select bit are used for one-by-one A-D conversion. Table 1.52 shows the specifications of single sweep mode. Figure 1.131 shows the A-D control register in single sweep mode.

**Table 1.52. Single sweep mode specifications**

| Item | Specification |
|---|---|
| Function | The pin selected by the A-D sweep pin select bit are used for one-by-one A-D conversion |
| Start condition | Writing 1 to A-D conversion start flag |
| Stop condition | Writing 0 to A-D conversion start flag<br>End of A-D conversion (A-D conversion start flag changes ot 0 , except when external trigger is selected). |
| Interrupt request generation timing | End of A-D conversion |
| Input pin | $AN_0$ and $AN_1$ (2 pins), $AN_0$ and $AN_3$ (4 pins), $AN_0$ and $AN_5$ (6 pins), or $AN_0$ and $AN_1$ (8 pins) |
| Reading A-D converter results | Read A-D register corresponding to selected pin |

A-D control register 0 (Note)

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

Symbol ADCON0 — Address 03D6$_{16}$ — When reset 00000XXX$_2$

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

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

A-D control register 1 (Note 1)

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

Symbol ADCON1 — Address 03D7$_{16}$ — When reset 00$_{16}$

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

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

**Fig. 1.131. A-D conversion register in single sweep mode**

### (4) Repeat sweep mode 0

In repeat sweep mode 0, the pins selected using the A-D sweep pin select bit are used for repeat sweep A-D conversion. Table 1.53 shows the specifications of repeat sweep mode 0. Figure 1.132 shows the A-D control register in repeat sweep mode 0.

**Table 1.53. Repeat sweep mode 0 specifications**

| Item | Specification |
|---|---|
| Function | The pin selected by the A-D sweep pin select bit are used for repeat sweep A-D conversion |
| Start condition | Writing 1 to A-D conversion start flag |
| Stop condition | Writing 0 to A-D conversion start flag |
| Interrupt request generation timing | None generated |
| Input pin | $AN_0$ and $AN_1$ (2 pins), $AN_0$ and $AN_3$ (4 pins), $AN_0$ and $AN_5$ (6 pins), or $AN_0$ and $AN_1$ (8 pins) |
| Reading A-D converter results | Read A-D register corresponding to selected pin (at any time) |

A-D control register 0 (Note)

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

Symbol ADCON0 — Address 03D6₁₆ — When reset 00000XXX₂

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

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

A-D control register 1 (Note 1)

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

Symbol ADCON1 — Address 03D7₁₆ — When reset 00₁₆

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SCAN0 | A-D sweep pin select bit | When single sweep and repeat sweep mode 0 are selected<br>(b1 b0)<br>0 0 : AN0, AN1 (2 pins)<br>0 1 : AN0 to AN3 (4 pins)<br>1 0 : AN0 to AN5 (6 pins)<br>1 1 : AN0 to AN7 (8 pins) | O | O |
| SCAN1 | | | O | O |
| MD2 | A-D operation mode select bit 1 | 0 : Any mode other than repeat sweep mode 1 | O | O |
| BITS | 8/10-bit mode select bit | 0 : 8-bit mode<br>1 : 10-bit mode | O | O |
| CKS1 | Frequency select bit 1 | 0 : fAD/2 or fAD/4 is selected<br>1 : fAD is selected | O | O |
| VCUT | Vref connect bit | 1 : Vref connected | O | O |
| OPA0 | External op-amp connection mode bit (Note 2) | (b7 b6)<br>0 0 : ANEX0 and ANEX1 are not used<br>0 1 : ANEX0 input is A-D converted<br>1 0 : ANEX1 input is A-D converted<br>1 1 : External op-amp connection mode | O | O |
| OPA1 | | | O | O |

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

**Fig. 1.132. A-D conversion register in repeat sweep mode 0**

### (5) Repeat sweep mode 1

In repeat sweep mode 1, all pins are used for A-D conversion with emphasis on the pin or pins selected using the A-D sweep pin select bit. Table 1.54 shows the specifications of repeat sweep mode 1. Figure 1.133 shows the A-D control register in repeat sweep mode 1.

**Table 1.54. Repeat sweep mode 1 specifications**

| Item | Specification |
|---|---|
| Function | All pins perform repeat sweep A-D conversion, with emphasis on the pin or pins selected by the A-D sweep pin select bit.<br>Example: $AN_0$ selected $AN_0 \rightarrow AN_1 \rightarrow AN_0 \rightarrow AN_2 \rightarrow AN_0 \rightarrow AN_3$, etc. |
| Start condition | Writing 1 to A-D conversion start flag |
| Stop condition | Writing 0 to A-D conversion start flag |
| Interrupt request generation timing | None generated |
| Input pin | $AN_0$ (1 pin), $AN_0$ and $AN_1$ (2 pins), $AN_0$ and $AN_2$ (3 pins), $AN_0$ and $AN_3$(4 pins) |
| Reading A-D converter results | Read A-D register corresponding to selected pin (at any time) |

A-D control register 0 (Note)

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

Symbol ADCON0 — Address 03D6$_{16}$ — When reset 00000XXX$_2$

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

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

A-D control register 1 (Note 1)

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

Symbol ADCON1 — Address 03D7$_{16}$ — When reset 00$_{16}$

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

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

**Fig. 1.133. A-D conversion register in repeat sweep mode 1**

### (a) Sample and hold

Sample and hold is selected by setting bit 0 of the A-D control register 2 (address 03D4$_{16}$) to “1”. When sample and hold is selected, the speed of conversion increases. As a result, a 28 $f_{AD}$ cycle is achieved with 8-bit resolution and 33 $f_{AD}$ with 10-bit resolution. Sample and hold can be selected in all modes. However, in all modes, be sure to specify before starting A-D conversion whether sample and hold is to be used.

### (b) Extended analog input pins

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

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

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

### (c) External operation amp connection mode

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

When bit 6 of the A-D control register 1 (address 03D7$_{16}$) is “1” and bit 7 is “1”, input via $AN_0$ to $AN_7$ is output from ANEX0. The input from ANEX1 is converted from analog to digital and the result stored in the corresponding A-D register. The speed of A-D conversion depends on the response of the external operation amp. Do not connect the ANEX0 and ANEX1 pins directly. Figure 1.134 is an example of how to connect the pins in external operation amp mode.

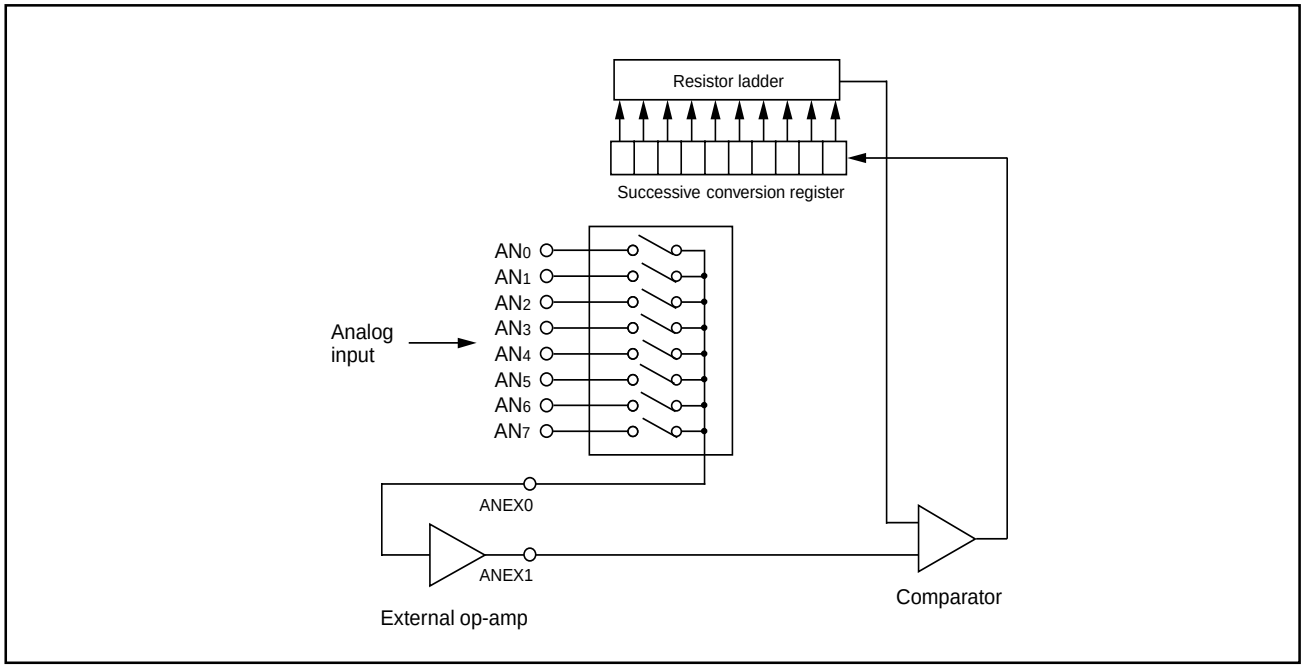



**Fig. 1.134. Example of external op-amp connection mode**

**(6) Operation of Simultaneous Sample and Hold Mode (SSH)**

$AN_1$ is sampled exactly the same time as the sampling of$AN_0$. The actual conversion occurs when a conversion of $AN_1$ is specifically requested. The request can be implicit as a part of a sweep mode or can be explicit by writing an appropriate value to ADCON0. The conversion of input 1 must be completed within 33µs after $AN_0$ conversion is started.

## A-D Usage Precautions

(1) Write to each bit (except bit 6) of A-D control register 0, to each bit of A-D control register 1, and to bit 0 and 7 of A-D control register 2 when A-D conversion is stopped before a trigger occurs.

In particular, when the Vref connection bit is changed from “0” to “1”, start A-D conversion after an elapse of 1 µs or longer.

(2) When changing A-D operation mode, select analog input pin again.

(3) Using one-shot mode or single sweep mode

Read the correspondence A-D register after confirming A-D conversion is finished. (It is known by A-D conversion interrupt request bit.)

(4) Using repeat mode, repeat sweep mode 0 or repeat sweep mode 1

Use the undivided main clock as the internal CPU clock.

(5) Use SSH only in conjunction with SMP at 33µs.

## D-A Converter

This is an 8-bit, R-2R type D-A converter. The microcomputer contains two independent D-A converters of this type. D-A conversion is performed when a value is written to the corresponding D-A register. Bits 0 and 1 (D-A output enable bits) of the D-A control register decide if the result of conversion is to be output. Do not set the target port to output mode if D-A conversion is to be performed.

Output analog voltage (V) is determined by a set value (n : decimal) in the D-A register.

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

$V_{ref}$ : reference voltage

Table 1.55 lists the performance of the D-A converter. Figure 1.135 shows the block diagram of the D-A converter. Figure 1.136 shows the D-A control register. Figure 1.137 shows the D-A converter equivalent circuit.

**Table 1.55. Performance of D-A converter**

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

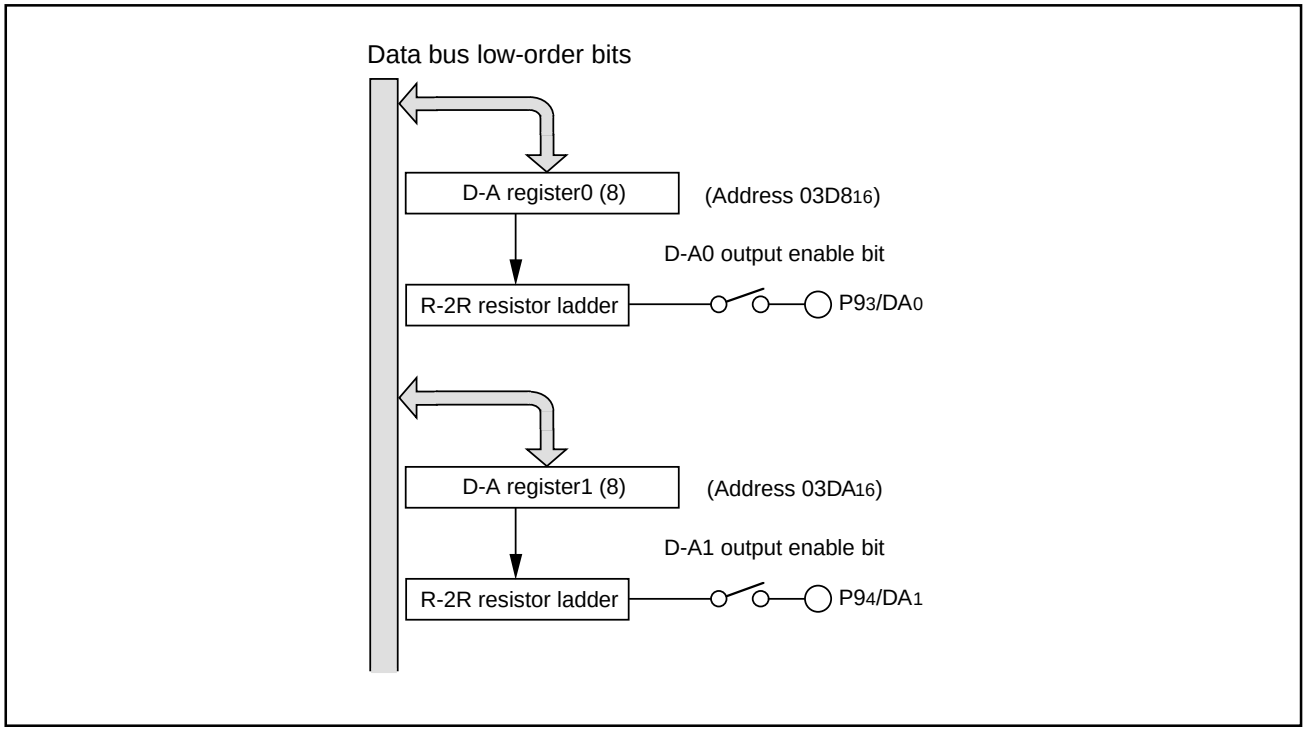



**Fig. 1.135. Block diagram of D-A converter**

D-A control register

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

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

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DA0E | D-A0 output enable bit | 0 : Output disabled<br>1 : Output enabled | O | O |
| DA1E | D-A1 output enable bit | 0 : Output disabled<br>1 : Output enabled | O | O |
| Nothing is assigned.<br>Write "0" when writing to these bits. If read, the value is "0". | | | — | — |

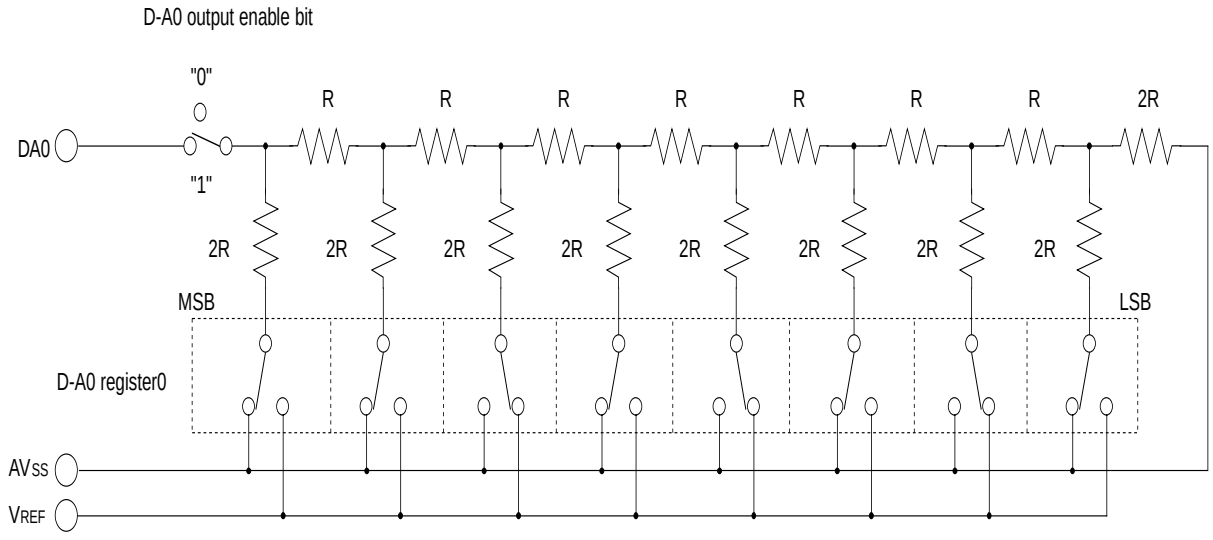
D-A register

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

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

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

**Fig. 1.136. D-A control register**




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

**Fig. 1.137. 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 1.138 shows the block diagram of the CRC circuit. Figure 1.139 shows the CRC-related registers. Figure 1.140 shows the calculation example using the CRC calculation circuit

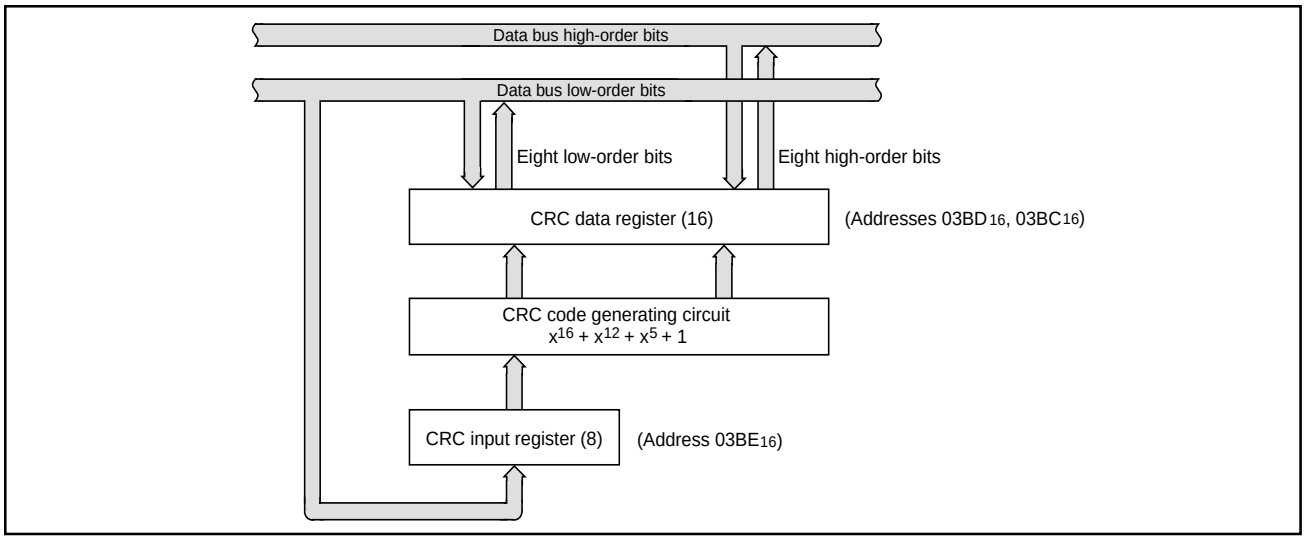



**Fig. 1.138. Block diagram of CRC circuit**

CRC data register

(b15) b7 (b8) b0 b7 b0

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

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

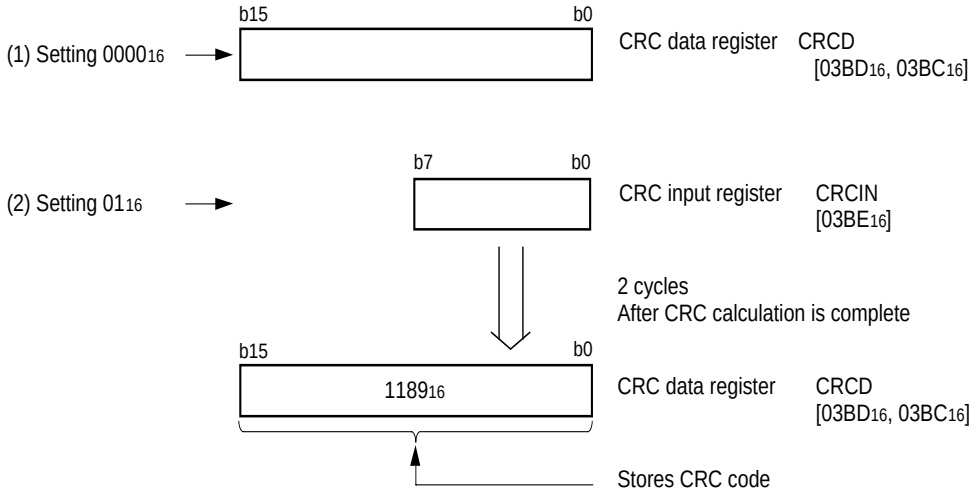
CRC input register

b7 b0

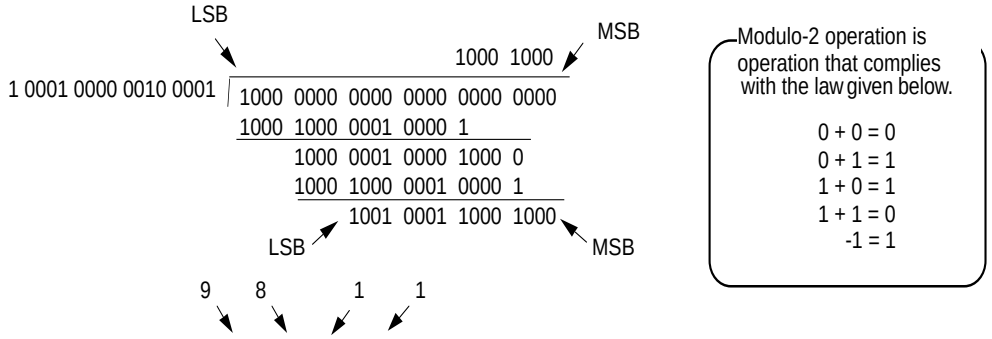
| Symbo | Address | When reset |
|---|---|---|
| CRCIN | 03BE16 | Indeterminate |

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

**Fig. 1.139. CRC-related registers**




The code resulting from sending 01 16 in LSB first mode is (1000 0000). Thus the CRC code in the generating polynomial, ($X^{16} + X^{12} + X^5 + 1$), becomes the remainder resulting from dividing (1000 0000) $X^{16}$ by (1 0001 0000 0010 0001) in conformity with the modulo-2 operation.




Thus the CRC code becomes (1001 0001 1000 1000). Since the operation is in LSB first mode, the (1001 0001 1000 1000) corresponds to 1189 16 in hexadecimal notation. If the CRC operation in MSB first mode is necessary in the CRC operation circuit built in the M16C, switch between the LSB side and the MSB side of the input-holding bits, and carry out the CRC operation. Also switch between the MSB and LSB of the result as stored in CRC data.

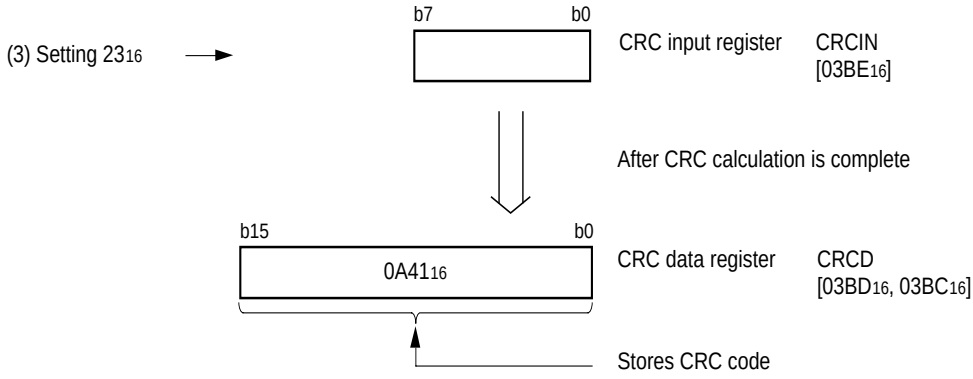



**Fig. 1.140. Calculation example using the CRC calculation circuit**

## Programmable I/O Ports

There are 55 programmable I/O ports: P3 to P10 (excluding P5, $P8_3$ and $P8_7$). 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. $P8_3$ is an input-only port and has no built-in pull-up resistance. $P7_0$ and $P7_1$ do not have pull up resistors.

Figures 1.141 to 1.143 show the programmable I/O ports. Figure 1.144 shows the I/O pins. Each pin functions as a programmable I/O port and as the I/O for the built-in peripheral devices.

To use the pins as the inputs for the built-in peripheral devices, set the direction register of each pin to input mode. When the pins are used as the outputs for the built-in peripheral devices (other than the D-A converter), they function as outputs regardless of the contents of the direction registers. When pins are to be used as the outputs for the D-A converter, do not set the direction registers to output mode. See the descriptions of the respective functions for how to set up the built-in peripheral devices.

### (1) Direction registers

Figure 1.145 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.

### (2) Port registers

Figure 1.146 shows the port registers.

These registers are used to write and read data for input and output to and from an external device. A port register consists of a port latch to hold output data and a circuit to read the status of a pin. Each bit in port registers corresponds one for one to each I/O pin.

### (3) Pull-up control registers

Figure 1.147 shows the pull-up control registers.

The pull-up control register can be set to apply a pull-up resistance to each block of 4 ports. When ports are set to have a pull-up resistance, the pull-up resistance is connected only when the direction register is set for input.

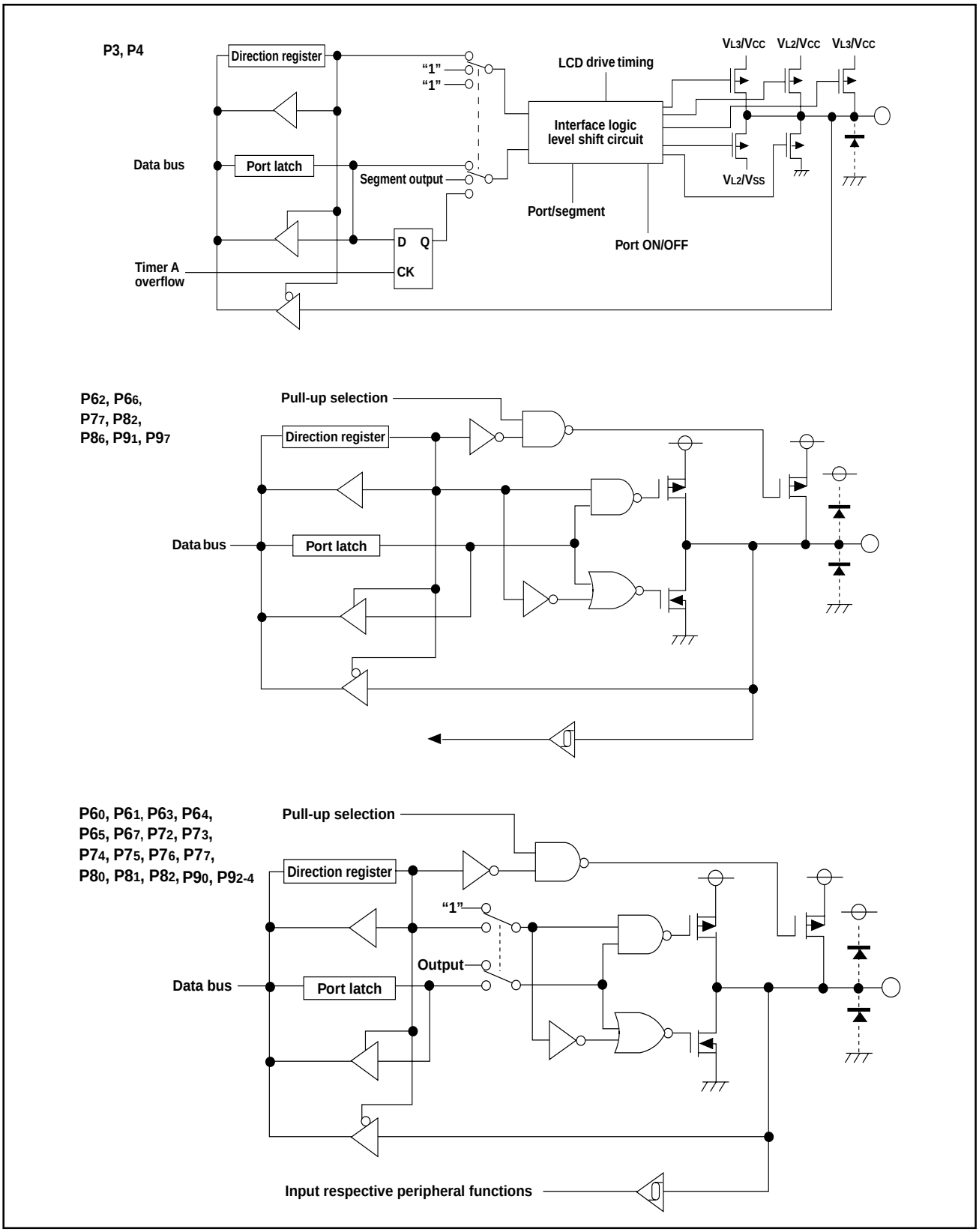



**Fig. 1.141. Programmable I/O ports (1)**

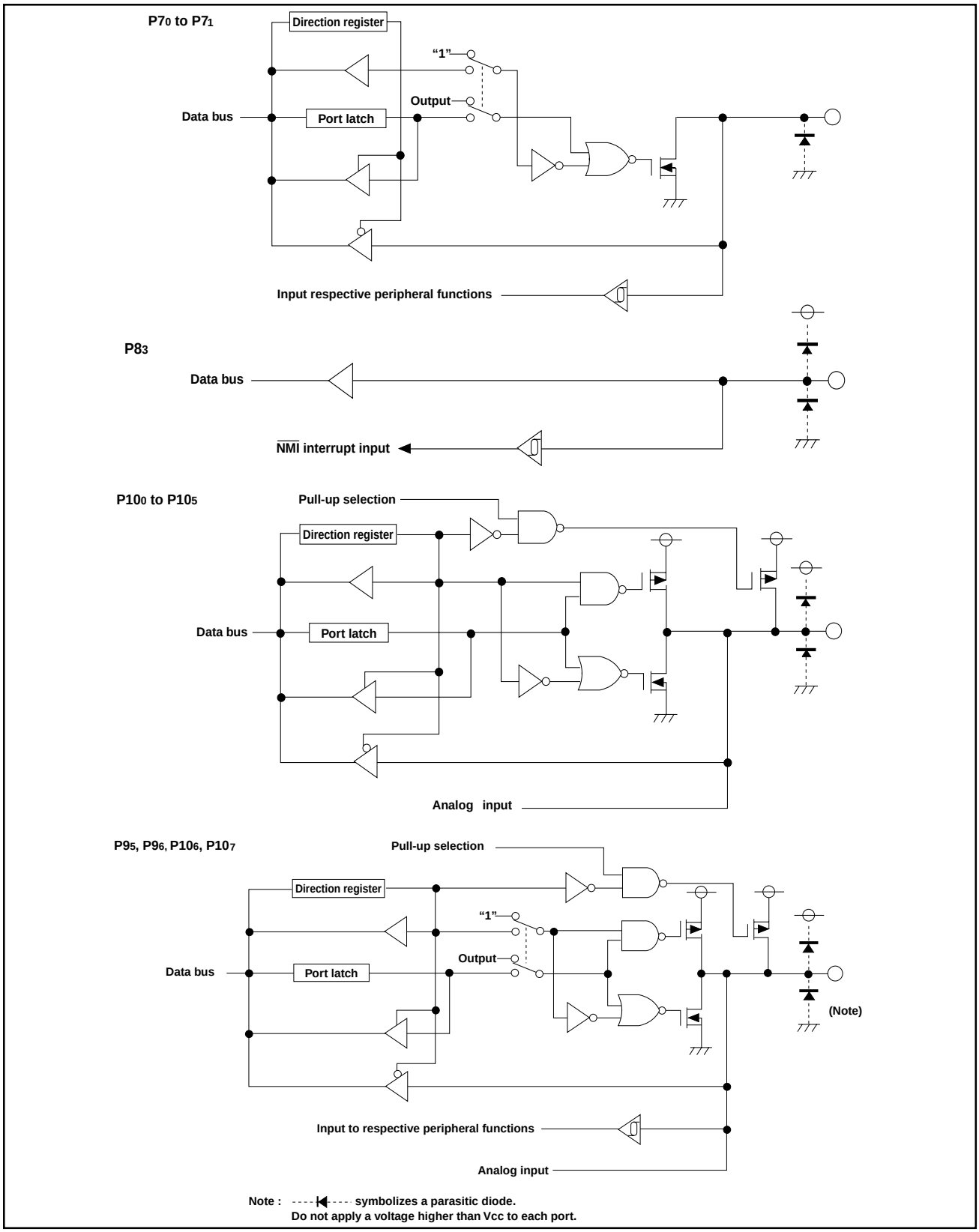
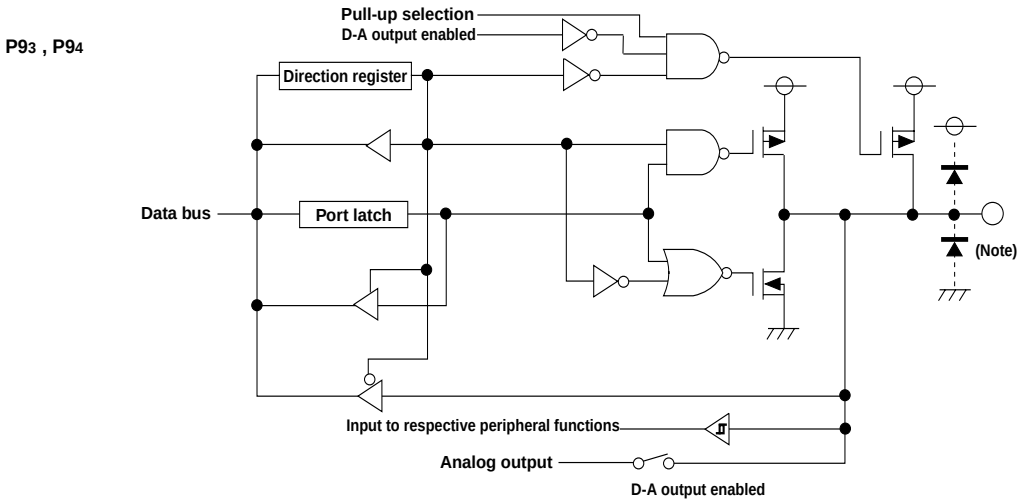



**Fig. 1.142. Programmable I/O ports (2)**



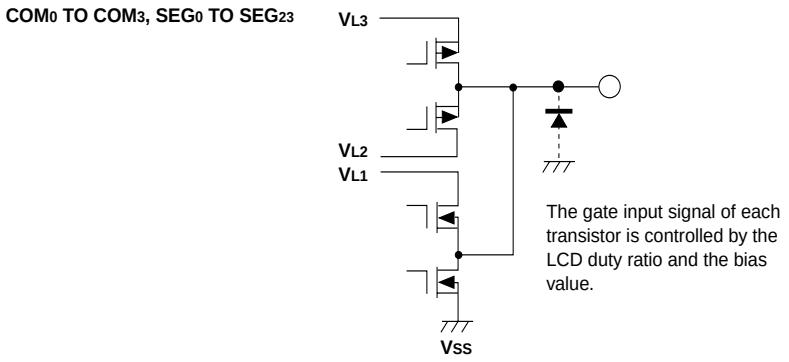




**Fig. 1.143. Programmable I/O ports (3)**

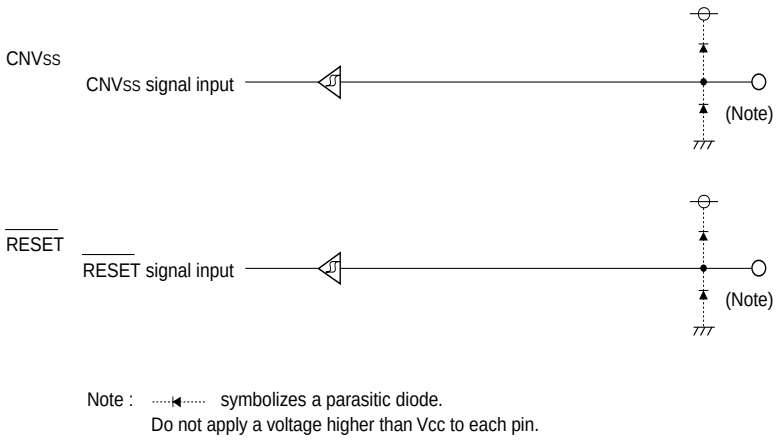



**Fig. 1.144. I/O pins**

Port Pi direction register (Note)

b7 b6 b5 b4 b3 b2 b1 b0

Symbol: PDi (i = 3 to 10, except 5 and 8)

Address: 03E7$_{16}$, 03EA$_{16}$, 03EE$_{16}$, 03EF$_{16}$, 03F3$_{16}$, 03F6$_{16}$

When reset: 00$_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PDi_0 | Port Pi$_0$ direction register | 0 : Input mode (Functions as an input port)<br>1 : Output mode (Functions as an output port)<br>(i = 3, 4, 6, 7, 9, 10) | ○ | ○ |
| PDi_1 | Port Pi$_1$ direction register | | ○ | ○ |
| PDi_2 | Port Pi$_2$ direction register | | ○ | ○ |
| PDi_3 | Port Pi$_3$ direction register | | ○ | ○ |
| PDi_4 | Port Pi$_4$ direction register | | ○ | ○ |
| PDi_5 | Port Pi$_5$ direction register | | ○ | ○ |
| PDi_6 | Port Pi$_6$ direction register | | ○ | ○ |
| PDi_7 | Port Pi$_7$ direction register | | ○ | ○ |

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

Port P8 direction register

b7 b6 b5 b4 b3 b2 b1 b0

Symbol: PD8

Address: 03F2$_{16}$

When reset: 00X00000$_2$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PD8_0 | Port P8$_0$ direction register | 0 : Input mode (Functions as an input port)<br>1 : Output mode (Functions as an output port) | ○ | ○ |
| PD8_1 | Port P8$_1$ direction register | | ○ | ○ |
| PD8_2 | Port P8$_2$ direction register | | ○ | ○ |
| Nothing is assigned. Write "0" when writing to this bit. When read, the value is indeterminate. | | | — | — |
| PD8_4 | Port P8$_4$ direction register | | ○ | ○ |
| PD8_5 | Port P8$_6$ direction register | | ○ | ○ |
| PD8_6 | Port P8$_7$ direction register | | ○ | ○ |
| Nothing is assigned. Write "0" when writing to this bit. When read, the value is indeterminate. | | | — | — |

**Fig. 1.145. Direction register**

Port Pi register

b7 b6 b5 b4 b3 b2 b1 b0

Symbol: Pi (i = 3 to 10, except 5 and 8)

Address: 03E5$_{16}$, 03E8$_{16}$, 03EC$_{16}$, 03ED$_{16}$, 03F1$_{16}$, 03F4$_{16}$

When reset: Indeterminate

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Pi_0 | Port Pi$_0$ register | Data is input and output to and from each pin by reading and writing to and from each corresponding bit<br>0 : "L" level data<br>1 : "H" level data (Note)<br>(i = 3, 4, 6, 7, 9, 10) | ○ | ○ |
| Pi_1 | Port Pi$_1$ register | | ○ | ○ |
| Pi_2 | Port Pi$_2$ register | | ○ | ○ |
| Pi_3 | Port Pi$_3$ register | | ○ | ○ |
| Pi_4 | Port Pi$_4$ register | | ○ | ○ |
| Pi_5 | Port Pi$_5$ register | | ○ | ○ |
| Pi_6 | Port Pi$_6$ register | | ○ | ○ |
| Pi_7 | Port Pi$_7$ register | | ○ | ○ |

Note: Because P7 and P7$_1$ are n-channel open drain ports, the data are high impedance.

Port P8 register

b7 b6 b5 b4 b3 b2 b1 b0

Symbol: P8

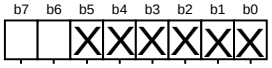
Address: 03F0$_{16}$

When reset: X0000000$_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| P8_0 | Port P8$_0$ register | Data is input and output to and from each pin by reading and writing to and from each corresponding bit except for P8$_3$)<br>0 : "L" level data<br>1 : "H" level data | ○ | ○ |
| P8_1 | Port P8$_1$ register | | ○ | ○ |
| P8_2 | Port P8$_2$ register | | ○ | ○ |
| P8_3 | Port P8$_3$ register | | ○ | ○ |
| P8_4 | Port P8$_4$ register | | ○ | ○ |
| P8_5 | Port P8$_5$ register | | ○ | ○ |
| P8_6 | Port P8$_6$ register | | ○ | ○ |
| Nothing is assigned. Write "0" when writing to this bit. When read, the value is indeterminate. | | | — | — |

**Fig. 1.146. Port register**

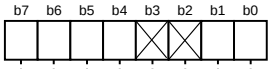
Pull-up control register 0




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

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Nothing is assigned. Write "0" when writing to these bits | | The corresponding port is pulled high with a pull-up resistor<br>0 : Not pulled high<br>1 : Pulled high | O | O |
| PU06 | P30 to P33 pull-up | | O | O |
| PU07 | P34 to P37 pull-up | | O | O |

Pull-up control register 1

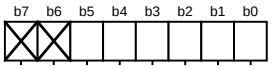



Symbol: PUR1 | Address: 03FD16 | When reset: 0016 (Note 1)

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PU10 | P40 to P43 pull-up | The corresponding port is pulled high with a pull-up resistor<br>0 : Not pulled high<br>1 : Pulled high | O | O |
| PU11 | P44 to P47 pull-up | | O | O |
| Nothing is assigned. Write "0" when writing to these bits. | | | O | O |
| PU14 | P60 to P63 pull-up | | O | O |
| PU15 | P64 to P67 pull-up | | O | O |
| PU16 | P72, P73 pull-up (Note 2) | | O | O |
| PU17 | P74 to P77 pull-up | | O | O |

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

Pull-up control register 2




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

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PU20 | P80 to P82 pull-up | The corresponding port is pulled high with a pull-up resistor<br>0 : Not pulled high<br>1 : Pulled high | O | O |
| PU21 | P84 to P86 pull-up (Except P83) | | O | O |
| PU22 | P90 to P93 pull-up | | O | O |
| PU23 | P94 to P97 pull-up | | O | O |
| PU24 | P100 to P103 pull-up | | O | O |
| PU25 | P104 to P107 pull-up | | O | O |
| Nothing is assigned. Write "0" when writing to these bits. When read, the value is indeterminate. | | | – | – |

**Fig. 1.147. Pull-up register**

## Unused Pins

**Table 1.56. Example of connections for unused pins**

| Pin Name | Connection |
|---|---|
| Ports P3 to P10 (excluding P5, $P8_3$, $P8_7$) | After setting for input mode, enable internal pull-up resistors or connect every pin to Vss cia a resistor. Or, after setitng for output mode, leave these pins open. |
| Xout (Note) | Open |
| $\overline{NMI}$ | Connect to Vcc via pull-up resistor |
| AVcc | Connect Vcc |
| AVss, $V_{REF}$ | Connect to Vss |
| C1, C2 | Open |
| VL2, VL3 | Connect to Vcc |
| VL1 | Connect to Vss |
| CNVss | Connect to Vss via pull-up resistor |
| $COM_0$ to $COM_3$ | Open |
| $SEG_0$ to $SEG_{23}$ | Open |

Note: With external clock input to Xin pin.

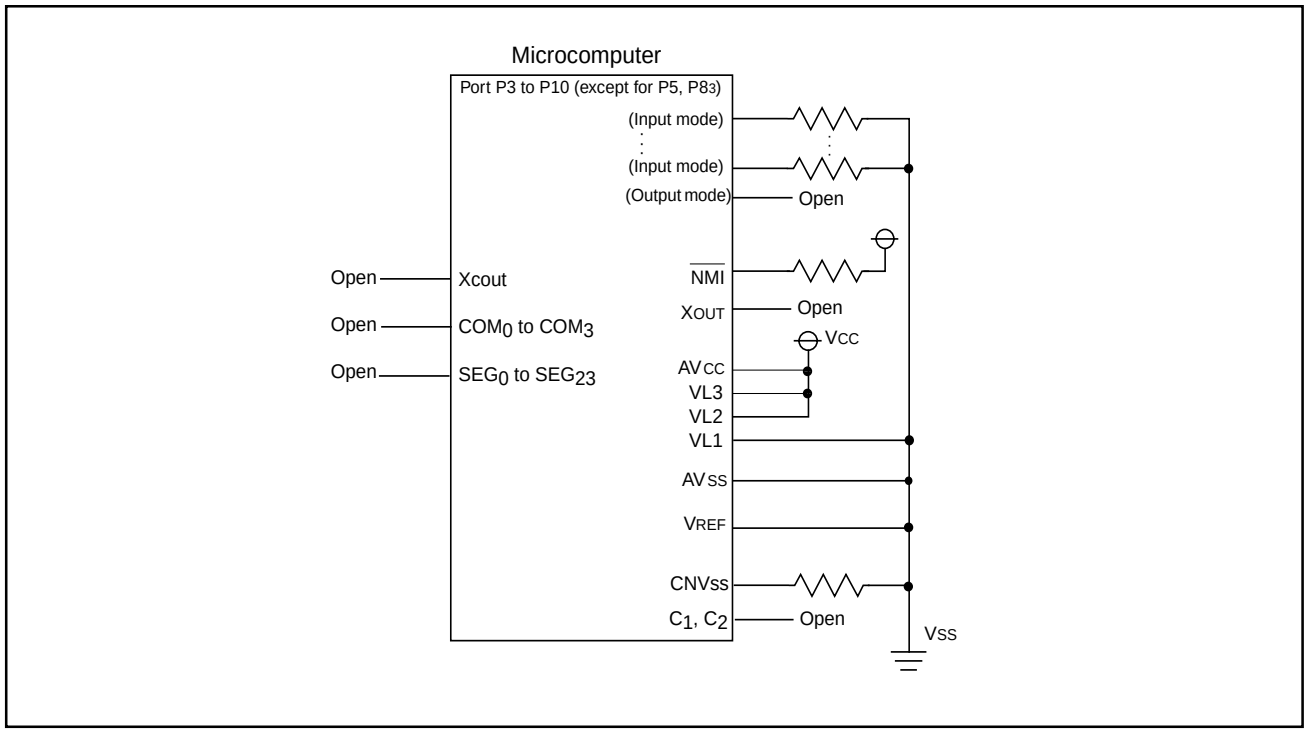



**Fig. 1.148. Example of connections for unused pins**

## Electrical Characteristics

**Table 1.57. Absolute maximum ratings**

| Symbol | Parameter | | Condition | Rated Value | Unit |
|---|---|---|---|---|---|
| Vcc | Supply voltage | | Vcc=AVcc | -0.3 to 6.5 | V |
| AVcc | Analog supply voltage | | Vcc=AVcc | -0.3 to 6.5 | V |
| $V_I$ | Input voltage | $\overline{RESET}$, , $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_6$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, Vref, Xin, CNVss (mask ROM version) | | -0.3 to Vcc + 0.3 | V |
| | | VL1 | | -0.3 to VL2 | V |
| | | VL2 | | VL1 to VL3 | V |
| | | VL3 | | VL2 to 6.5 | V |
| | | $P7_0$, $P7_1$, C1, C2, CNVss (flash memory version) | | -0.3 to 6.5 | V |
| $V_O$ | Output voltage | $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_2$, $P8_4$ to $P8_6$, $P9_0$ to $P9_7$, Xout, | | -0.3 to Vcc + 0.3 | V |
| | | $P3_0$ to $P3_7$, $P4_0$ to $P4_7$ | When output port | -0.3 to Vcc | V |
| | | | When segment output | -0.3 to VL3 | V |
| | | $P7_0$, $P7_1$, CNVss (mask ROM version) | | -0.3 to 6.5 | V |
| Pd | Power dissipation | | Ta=25°C | 300 | mW |
| Topr | Operating ambient temperature | | | -40 to 85 | °C |
| Tstg | Storage temperature | | | -65 to 150 | °C |

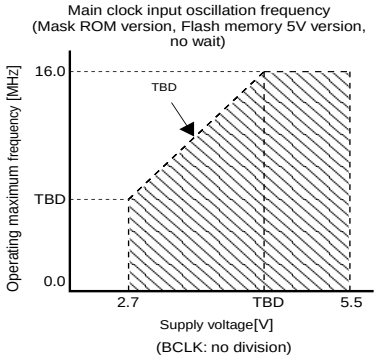
**Table 1.58. Recommended operating conditions (referrenced to Vcc = 2.7 to 5.5V at Ta = -20 to 85°C or -40 to 85°C (Note 3) unless othewise specified)**

| Symbol | Parameter | | | Standard | | | Unit |
|---|---|---|---|---|---|---|---|
| | | | | Min | Typ. | Max | |
| Vcc | Supply voltage | | | 2.7 | 5.0 | 5.5 | V |
| AVcc | Analog supply voltage | | | | Vcc | | V |
| Vss | Supply voltage | | | | 0 | | V |
| Avss | Analog supply voltage | | | | 0 | | V |
| $V_{IH}$ | High Input voltage | Xin, $\overline{RESET}$, CNVss, BYTE, $P3_1$ to $P3_7$, $P4_0$ to $P4_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_6$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$ | | 0.8 Vcc | | Vcc | V |
| | | P70, P71 | | 0.8 Vcc | | 6.5 | V |
| $V_{IL}$ | Low input voltage | Xin, $\overline{RESET}$, CNVss, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P6_0$ to $P6_7$,$P7_0$ to $P7_7$, $P8_0$ to $P8_6$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$ | | 0 | | 0.2 Vcc | V |
| $I_{OH}$ (peak) | High peak output current (Note 2) | $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_2$, $P8_4$ to $P8_6$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$ | | | | -10.0 | mA |
| $I_{OH}$ (avg) | High average output current (Note 1) | $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_2$, $P8_4$ to $P8_6$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$ | | | | -5.0 | mA |
| $I_{OL}$ (peak) | Low peak output current (Note 2) | $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_2$, $P8_4$ to $P8_6$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$ | | | | 10.0 | mA |
| $I_{OL}$ (avg) | Low average output current (Note 1) | $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_2$, $P8_4$ to $P8_6$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$ | | | | 5.0 | mA |
| f(Xin) | Main clock input oscillation frequency (Note 3) | No wait | Vcc=TBD to 5.5V | 0 | | 16 | MHz |
| | | | Vcc=2.7V to TBD | 0 | | TBD | MHz |
| | | With Wait | Vcc=TBD to 5.5V | 0 | | 16 | MHz |
| | | | Vcc=2.7V to TBD | 0 | | TBD | MHz |
| f(Xcin) | Subclock oscillation frequency | | | | 32.768 | 50 | KHz |

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

Note 2: The total $I_{OL}$ (peak) and $I_{OH}$ (peak) fo ports P3, P4, P6, P7, $P8_0$ to $P8_2$, $P8_4$ to $P8_6$, P9 and P10 must be 80 mA max.

Note 3: Relationship between main clock oscillation frequency and supply voltage.



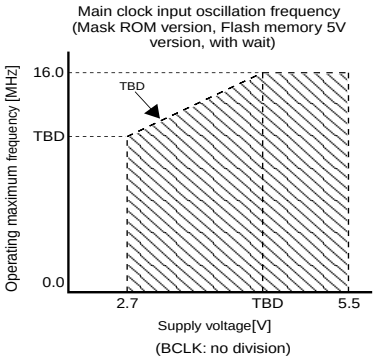



**Table 1.59. Electrical characteristics (referenced to Vcc = 5V, Vss = 0V at Ta = 25°C, f(Xin) = 16MHz unless otherwise specified.**

| Symbol | Parameter | | | Measuring condition | | Standard | | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | Min | Typ. | Max | |
| $V_{OH}$ | High output voltage | $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_2$, $P8_4$ to $P8_6$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$ | | $I_{OH}$=-5mA | | 3.0 | | | V |
| $V_{OH}$ | High output voltage | $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_2$, $P8_4$ to $P8_6$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$ | | $I_{OH}$=-200µA | | 4.7 | | | V |
| $V_{OH}$ | High output voltage | Xout | High power | $I_{OH}$=-1mA | | 3.0 | | | V |
| | | | Low power | $I_{OH}$=-0.5mA | | 3.0 | | | V |
| | High output voltage | Xcout | High power | With no load applied | | | 3.0 | | V |
| | | | Low power | With no load applied | | | 1.6 | | V |
| $V_{OL}$ | Low output voltage | $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_2$, $P8_4$ to $P8_6$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$ | | $I_{OL}$=-5mA | | | | 2.0 | V |
| $V_{OL}$ | Low output voltage | $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_2$, $P8_4$ to $P8_6$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$ | | $I_{OL}$=-200µA | | | | 0.45 | V |
| $V_{OL}$ | Low output voltage | Xout | High power | $I_{OL}$=-1mA | | | | 2.0 | V |
| | | | Low power | $I_{OL}$=-0.5mA | | | | 2.0 | V |
| | Low output voltage | Xcout | High power | With no load applied | | | 0 | | V |
| | | | Low power | With no load applied | | | 0 | | V |
| $V_T$ + - $V_T$- | Hysteresis | TA0in to TA4in, TA0in to TB3in, INT0 to INT5, $AD_{\overline{TRG}}$, CTS0, $\overline{CTS1}$, CLK0, CLK1, TA2out to TA4out, NMI, KI0 to KI4 | | | | 0.2 | | 0.8 | V |
| $V_T$ + - $V_T$- | Hysteresis | $\overline{RESET}$ | | | | 0.2 | | 1.8 | V |
| $I_{IH}$ | High input current | Xin, $\overline{RESET}$, CNVss, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_6$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$ | | $V_I$=5V | | | | 5.0 | µA |
| $I_{IL}$ | Low input current | Xin, $\overline{RESET}$, CNVss, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_6$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$ | | $V_I$=0 | | | | -5.0 | µA |
| $R_{PULLUP}$ | Pull-up resistance | $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_2$, $P8_4$ to $P8_6$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$ | | $V_I$=0V | | 30.0 | 50.0 | 167.0 | kΩ |
| $R_{f(Xin)}$ | Feedback resistance | | Xin | | | | 1.0 | | MΩ |
| $R_{f(Xcin)}$ | Feedback resistence | | Xcin | | | | 6.0 | | MΩ |
| $V_{RAM}$ | RAM retention voltage | | | When clock is stopped | | 2.0 | | | V |
| Icc | Power supply current | | | Mask ROM versions | f(Xin) =16 MHz Square wave, no division | | 30.0 | 50.0 | mA |
| | | | | Flash memory 5V version | f(Xin) =16 MHz Square wave, no division | | 35.0 | 50.0 | mA |
| | | | | Mask ROM versions | f(Xcin) =32 kHz Square wave | | 90.0 | | µA |
| | | | | Flash memory 5V version | f(Xcin) =32 kHz Square wave | | 8.0 | | mA |
| | | | | | f(Xcin) =32 kHz When a WAIT instruction is executed | | 4.0 | | µA |
| | | | | | Ta =25°C when clock is stopped | | | 1.0 | µA |
| | | | | | Ta = 85°C when clock is stopped | | | 20.0 | |

**Table 1.60. A-D conversion characterisitics (referenced to Vcc = AVcc + Vref = 5V, Vss = AVss = 0V at Ta = 25°C, f(Xin) = 16MHz unless otherwise specified)**

| Symbol | Parameter | | Measuring Condition | | Standard | | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | | | | Min. | Typ. | Max. | |
| _ | Resolution | | Vref = Vcc | | | | 10 | Bits |
| _ | Absolute Accuracy | Sample & Hold function not available | Vref = Vcc = 5V | | | | +/-3 | LSB |
| | | Sample & Hold function available (10 bit) | Vref = Vcc = 5V | $AN_0$ to $AN_7$ input | | | +/-3 | LSB |
| | | | | ANEX0, ANEX1 input, external op-amp connection mode | | | +/-7 | LSB |
| | | Sample & Hold function available (8 bit) | Vref = Vcc = 5V | | | | +/-2 | LSB |
| $R_{LADDER}$ | Ladder resistence | | Vref = Vcc | | 10 | | 40 | kΩ |
| $t_{CONV}$ | Conversion time (10 bit) | | | | 3.3 | | | µs |
| $t_{CONV}$ | Converstion time (8 bit) | | | | 2.8 | | | µs |
| $t_{SAMP}$ | Sampling time | | | | 0.3 | | | µs |
| Vref | Reference voltage | | | | 2 | | Vcc | V |
| $V_{IA}$ | Analog input voltage | | | | 0 | | Vref | V |

**Table 1.61. D-A conversion characteristics (referenced to Vcc = 5V, Vss = AVss = 0V, Vref = 5V at Ta = 25°C, f(Xin) = 16MHz unless otherwise specified)**

| Symbol | Parameter | Measuring Condition | Standard | | | Unit |
|---|---|---|---|---|---|---|
| | | | Min. | Typ. | Max. | |
| _ | Resolution | | | | 8 | Bits |
| _ | Absolute Accuracy | | | | 1.0 | % |
| $t_{SU}$ | Settling time | | | | 3 | µs |
| $R_O$ | Output resistance | | 4 | 10 | 20 | kΩ |
| $I_{Vref}$ | Reference power supply input current | (Note) | | | 1.5 | mA |

Note: This applies when using one D-A converter, with the D-A register. The unused D-A converter is set to $00_{16}$ .
The A-D converter s ladder resistance is not included.
When the Vref is disconnected at the A-D control register, IVref is sent.

**Timing requirements (referenced to Vcc = 5V, Vss = 0V at Ta = 25° C unless otherwise stated)**

**Table 1.62 External clock input**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_C$ | External clock input cycle time | 62.5 | | ns |
| $t_W(H)$ | External clock input HIGH pulse width | 25 | | ns |
| $t_W(L)$ | External clock input LOW pulse width | 25 | | ns |
| $t_r$ | External clock rise time | | 15 | ns |
| $t_f$ | External clock fall time | | 15 | ns |

**Table 1.63. 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 1.64. 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 1.65. 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 1.66. 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 1.67. 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 |

**Table 1.68. 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 HIGH pulse width (counted on both edges) | 80 | | ns |

**Table 1.69. 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 |

**Timing requirements (referenced to Vcc = 5V, Vss = 0V at Ta = 25° C unless otherwise stated)**

**Table 1.70. 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 1.71. A-D trigger input**

| 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 1.72 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 |
| $t_h$(D-C) | 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 1.73. External interrupt $\overline{\text{INT}}$i inputs**

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

## Flash Memory

The M30222 (flash memory version) has an internal new DINOR (Divided bit line NOR) flash memory that can be rewritten with a single power source. Three flash memory modes are available that read, program, and erase: parallel I/O and standard serial I/O modes that flash memory manipulates using a programmer, and a CPU rewrite mode that flash memory can manipulate using the Central Processing Unit (CPU). Each mode is detailed in the following pages.

The flash memory is divided into several block as shown in Figure 1.149 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 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 users application system. This boot ROM area can be rewritten in only parallel I/O mode.

### Outline Performance

Table 1.74 shows the outline performance of the M30222 (flash memory version).

**Table 1.74. Outline performance**

| Item | | Performance |
|---|---|---|
| Erase/Write voltage | | 2.7 to 5.5V |
| Flash memory operation mode | | Three modes: CPU rewrite, parallel I/O, standard serial |
| Erase block division | User ROM area | See Fig. 1.149 |
| | Boot ROM area | One division (8 Kbytes) (Note) |
| Program method | | uses word units |
| Erase method | | Collective erase/block erase |
| Program/erase control method | | Program/erase control by software command |
| Number of Commands | | 8 |
| Program/erase count | | 100 |
| ROM code protect | | Parallel I/O and standard serial modes are supported. |

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

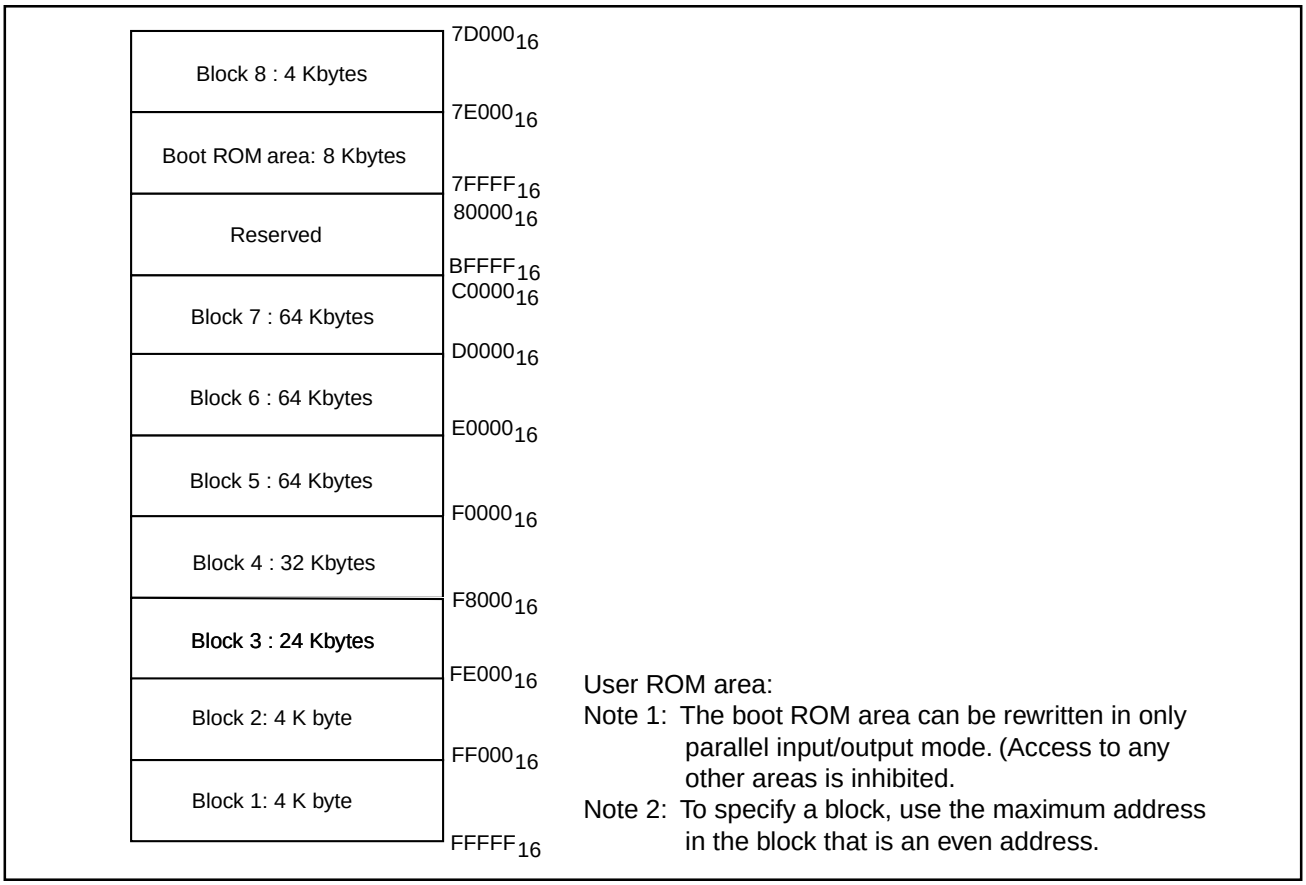



**Figure 1.149. Block diagram of flash memory version**

## 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). Only the user ROM area shown in Figure 1.149 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 RAM memory before it can be executed.

### Microcomputer Mode and Boot Mode

Normal microcomputer mode is entered when the microcomputer is reset with pulling CNVss pin low. In this case, the CPU starts executing the control program in the user ROM area. When the microcomputer is reset and both the CNVss pin and $P7_4$ ($\overline{CE}$) pin are pulled high, the CPU starts operating using the control program in the boot ROM area (program start address is $C000_{16}$, $7D000_{16}$). This mode is called the "boot" mode.

The control program for CPU rewrite mode must be written into the user ROM or boot ROM area beforehand. (If the control program is written into the boot ROM area, standard serial I/O mode becomes unusable.) See Figure 1.149 for details about the boot ROM area.

### Block Address

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

### Outline Performance

In the CPU rewrite mode, the CPU erases, programs, and reads the internal flash memory as instructed by software commands. This rewrite control program must be transferred to internal RAM before it can be executed.

The CPU rewrite mode is accessed by writing "1" for the CPU rewrite mode select bit (bit 1 in address $034B4_{16}$). Software commands are accepted once the mode is accessed.

In the CPU rewrite mode, software commands are used to write and read data into even-numbered addresses ("0" for byte address A0) in 16-bit units. Always write 8-bit software comands 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 1.150 shows the flash memory control register.

Flash memory control register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 | Symbol | Address | When reset |
|---|---|---|---|---|---|---|---|---|---|---|
| X | X | X | | | | | | FMCR | 03B4$_{16}$ | XXXX0001$_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| FMCR0 | RY/$\overline{BY}$ status flag | 0: Busy (being written or erased)<br>1: Ready | O | — |
| FMCR1 | CPU rewrite mode select bit (Note 1) | 0: Normal mode (Software commands invalid)<br>1: CPU rewrite mode (Software commands acceptable) | O | O |
| FMCR2 | CPU rewrite mode entry flag | 0: Normal mode<br>1: CPU rewrite mode | O | — |
| FMCR3 | Flash memory reset bit (Note 2) | 0: Normal operation<br>1: Reset | O | O |
| FMCR4 | Write protect Bit (Note 1) | 0: Write protected<br>1: Write enabled | O | O |
| Nothing is assigned. When write, set "0". When read, values 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. Use the control program in RAM for write to this bit.
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 selection bit is = "1".

**Fig. 1.150 Flash memory control registers**

Bit 0 is the RY/BY status flag used exclusively to read the operating status of the flash memory. During programming and erase operations, it is "0". Otherwise, it is "1".

Bit 1 is the CPU rewrite mode select bit. When this bit is set to "1", the M30222 accesses the CPU rewrite mode. Software commands are accepted once the mode is accessed. In CPU rewrite mode, the CPU becomes unable to access the internal flash memory directly. Therefore, the control program that sets this bit must be executed out of RAM. To set this bit to "1", it is necessary to write "0" and then write "1" in succession. The bit can be set to "0" by only writing a "0".

Bit 2 is the CPU rewrite mode entry flag. This bit can be read to check whether the CPU rewrite mode has been entered or not.

Bit 3 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" for this bit resets the control circuit. To release the reset, it is necessary to set this bit to "0". If the control circuit is reset while erasing is in progress, a 5 ms wait is needed so that the flash memory can restore normal operation. Figure 1.151 shows a flowchart for setting/releasing the CPU rewrite mode.

Bit 4 is the flash memory protect bit. The blocks are write protected when this bit is "0". The write protect is disabled when the bit is "1". The MCU must be in CPU rewrite mode for this bit to have any effect. To set this bit to "1", it is necessary to write "0" and then write "1" in succession.

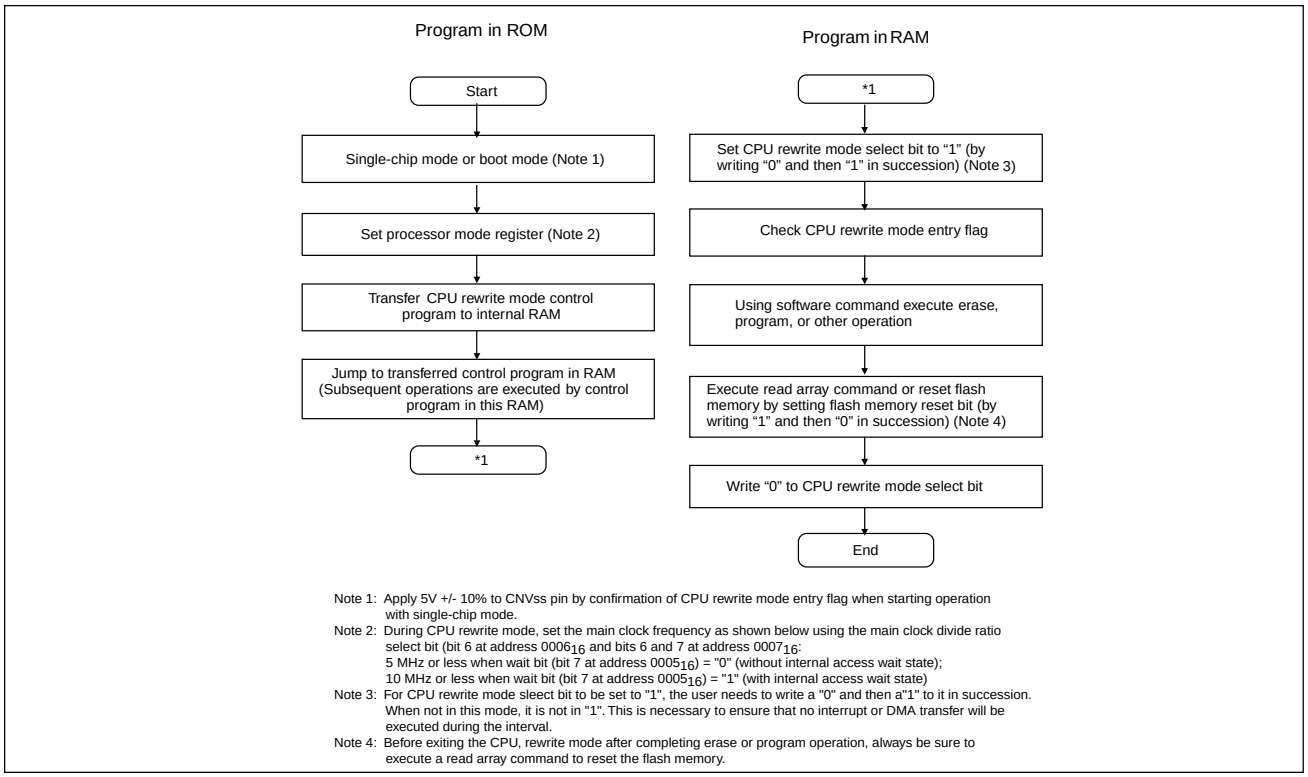



**Figure 1.151. CPU rewrite mode set/reset flowchart**

## 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 divide ratio select bit (bit 6 at address 0006$_{16}$ and bits 6 and 7 at address 0007$_{16}$):

5.0 MHz or less when wait bit (bit 7 at address 0005$_{16}$) = 0 (no wait state)

10.0 MHz or less when wait bit (bit 7 at address 0005$_{16}$) = 1 (one wait state)

**(2) Instruction 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 $\overline{\text{NMI}}$, address match, and watchdog timer interrupts 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.

**(4) Reset**

If the MCU is reset while erasing is in progress, a 5 ms wait is needed so that the flash memory can restore normal operation. Set a 5 ms wait to release the reset operation. Also, when the reset has been released, the program execute start address is automatically set to 07E000$_{16}$, therefore program so that the execute start address of the boot ROM is 07E000$_{16}$.

**Table 1.75. List of software commands (CPU rewrite mode)**

| Commands (Note 1) | Cycle No. | 1st Bus Cycle | | | 2nd Bus Cycle | | |
|---|---|---|---|---|---|---|---|
| | | Mode | Address | Data (D7:D0) | Mode | Address | Data (D7:D0) |
| Read | 1 | Write | X (Note 5) | FFh | | | |
| ID Codes | 2 | Write | X | 90h | Read | IA | ID |
| Status Regiser Read | 2 | Write | X | 70h | Read | X | SRD (Note 2) |
| Status Register Clear | 1 | Write | X | 50h | | | |
| Word Program | 2 | Write | X | 40h | Write | WA (Note 3) | WD (Note 3) |
| Auto Block Erase | 2 | Write | X | 20h | Write | BA (Note 4) | D0h |
| Erase | 2 | Write | X | 20h | Write | X | 20h |
| (Erase all unlocked blocks) | 2 | Write | X | A7h | Write | X | D0h |
| Lock Bit Status Read | 2 | Write | X | 71h | Read | BA (Note 4) | (D6)(Note 6) |
| Lock Bit Program | 2 | Write | X | 77h | Write | BA (Note 4) | D0h |

Note 1: When a software command is input, the high-order byte of data (D15:D8) is ignored.
Note 2: SRD = Status Register Data
Note 3: WA = Write Address, WD = Write Data (16 bits)
Note 4: BA = Block Address (Enter the maximum address of each block that is an even address)
Note 5: X denotes a given even address in the user ROM.
Note 6: Lock bit output on Data bit 6

## Software Commands

Table 1.75 lists the software commands available with the M30222 (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.

### 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 error bits 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.

### Program Command (40$_{16}$)

Program operation starts when the command code "40$_{16}$" is written in the first bus cycle. Then, if the address and data to program are written in the second bus cycle, program operation (data programming and verification) will start.

Whether the write operation is completed can be confirmed by reading the status register or the RY/BY status flag. When the program starts, the read status register mode is accessed automatically and the content of the status register is read into the data bus ($D_0$–$D_7$). The status register bit 7 (SR7) is set to 0 at the same time the write operation starts and is returned to 1 upon completion of the write operation. In this case, the read status register mode remains active until the Read Array command (FF$_{16}$) is written.

The RY/BY status flag is 0 during write operation and 1 when the write operation is completed as is the status register bit 7. At program end, program results can be checked by reading the status register. Figure 1.152 shows an example of a program flowchart.

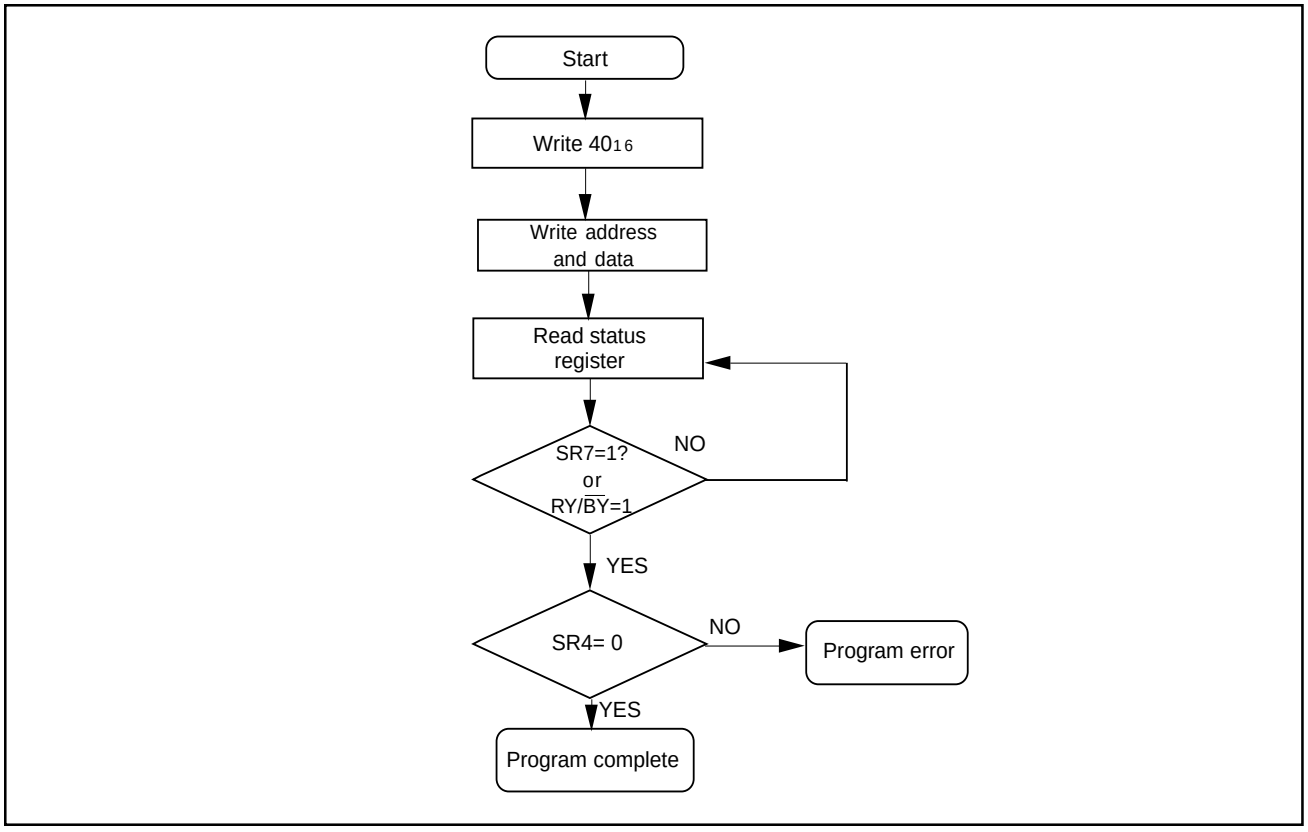



**Fig. 1.152. Program flowchart**

### Erase All Command (2016/2016) (Erases all blocks regardless of lock status)

By writing the command code "2016" in the first bus cycle and the confirmation command code "2016" in the second bus cycle that follows, the system starts erase all blocks (erase and erase verify). Whether the erase all blocks command is terminated can be confirmed by reading the status register or the RY/BY status flag. When the erase all blocks operation starts, the read status register mode is accessed automatically and 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 erase operation starts and is returned to "1" upon completion. The read status register mode remains active until the read array command (FF16) is written. The Boot Block area is not affected by this command.

The RY/BY status flag is "0" during erase operation and "1" when the erase operation is completed as is the status register bit 7.

At erase all blocks end, erase results can be checked by reading the status register. For details, refer to the section where the status register is detailed.

### Block Erase Command (2016/D016)

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

Whether the block erase operation is completed can be confirmed by reading the status register or the RY/BY status flag. At the same time the block 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 clock erase operation starts and is returned to "1" upon completion of the block erase operation. In this case, the read status register mode remains active until the Read Array command (FF16).

The RY/BY status flag is "0" during block erase operation and "1" when the block erase operation is completed as is the status register bit 7. After the block erase operation is completed, the status register can be read out to know the result of the block erase operation. For details, refer to the section where the status register is detailed.

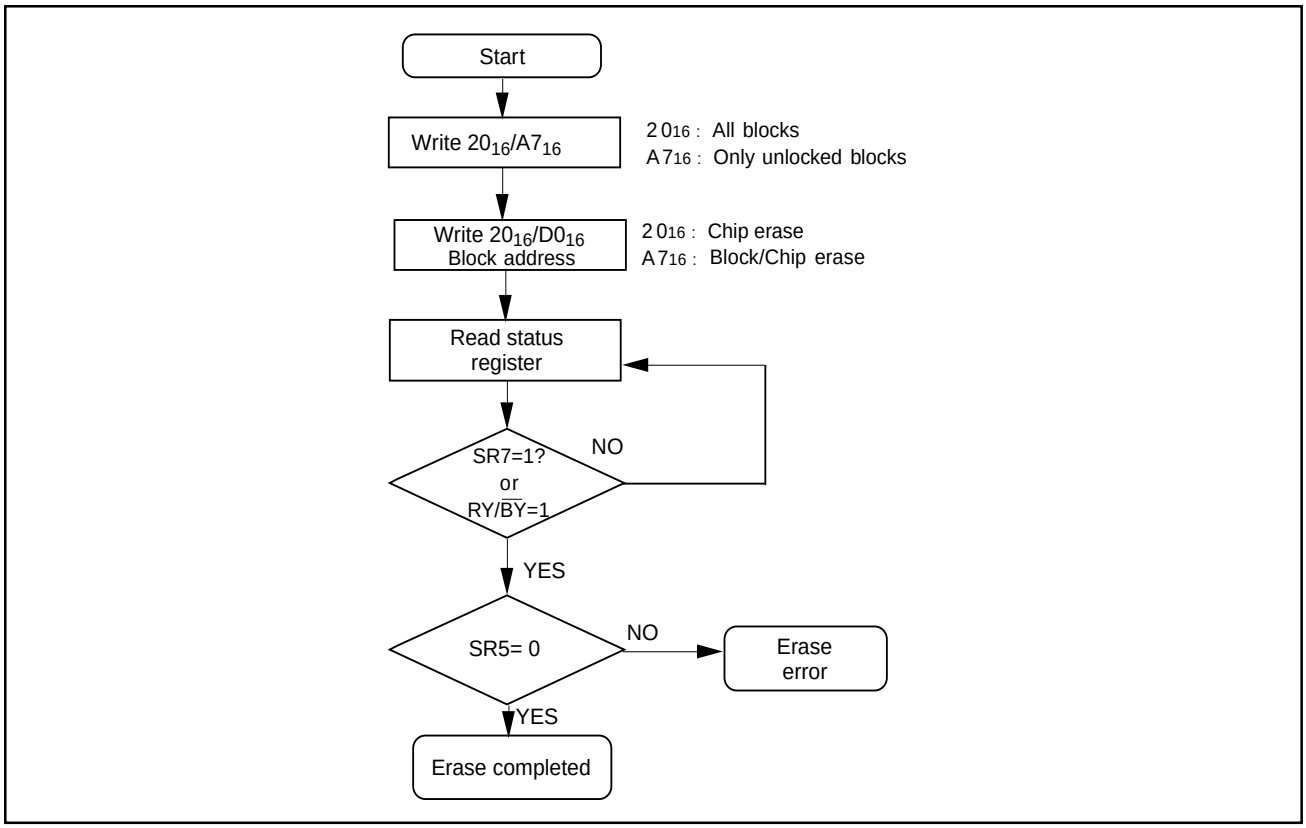



**Fig. 1.153. Erase flowchart**

### Erase All Unlocked Blocks Command (A7₁₆/D0₁₆)

By writing the command code "A7₁₆" in the first bus cycle and the confirmation command code "D0₁₆" in the second bus cycle that follows, the system starts erase all unlocked blocks (erase and erase verify). Whether the erase all unlocked blocks command is terminated can be confirmed by reading the status register or the RY/BY status flag. When the erase all unlocked blocks operation starts, the read status register mode is accessed automatically and 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 erase operation starts and is returned to "1" upon completion of the erase operation. The read status register mode remains active until the Read Array command (FF₁₆) is written. The RY/BY status flag is "0" during erase operation and "1" when the erase operation is completed as is the status register bit 7.

At erase all blocks end, erase results can be checked by reading the status register. For details, refer to the section where the status register is detailed. Figure 1.153 shows an example of a block erase flowchart.

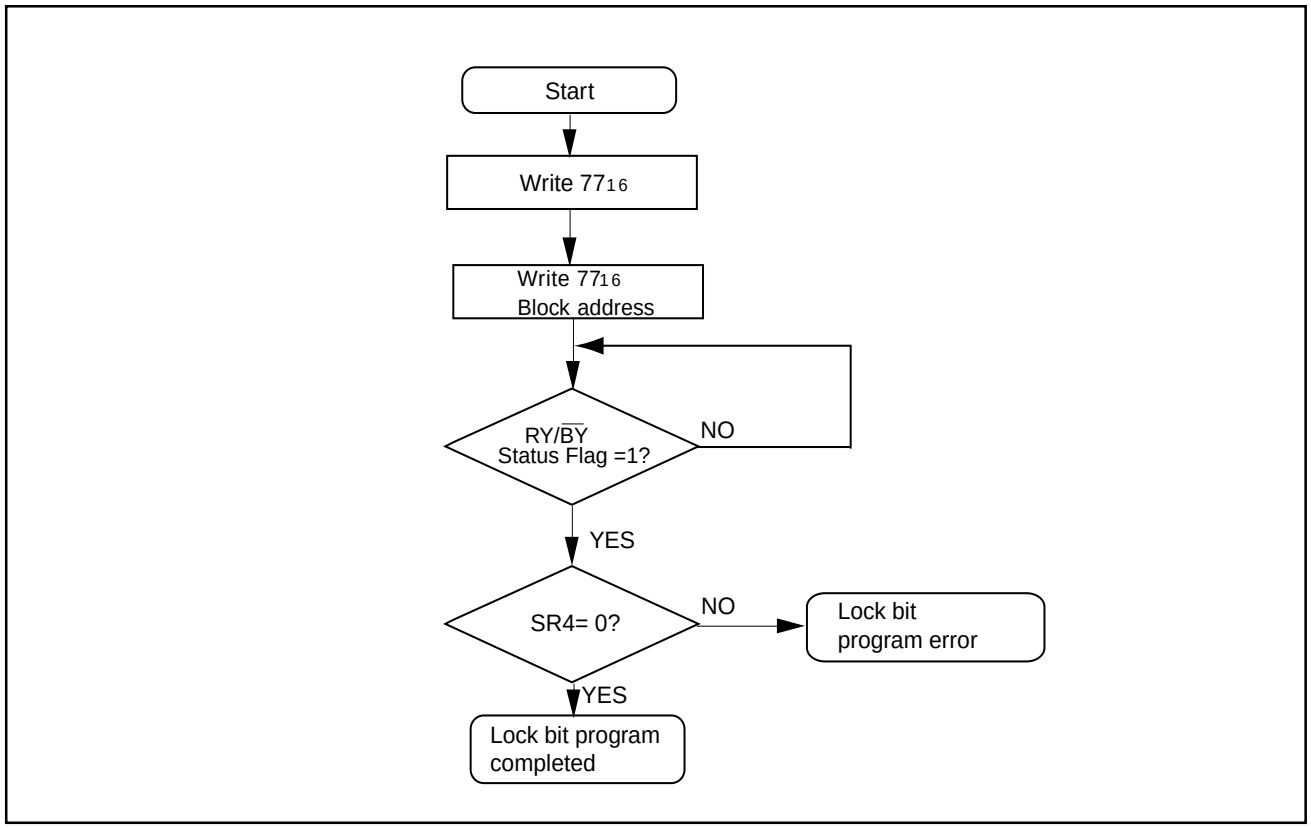



**Fig. 1.154. Lock Bit Program Flowchart**

### Lock Bit Program Command (7716/D016)

This command is available only when the lock bit is enabled. For CPU rewrite mode bit 4 must be a "1". In parallel mode WPB must be a "0". By writing the command code ("7716") in the first bus cycle and the confirmation code ("D016") and block address in the second cycle the lock bit can be programmed for the block specified. The lock bit protects the block against erase during the erase all unlocked blocks command. The status of the lock bit can be read by issuing the Lock bit status read command.

Figure 1.154 is an example of a lockbit program flowchart.

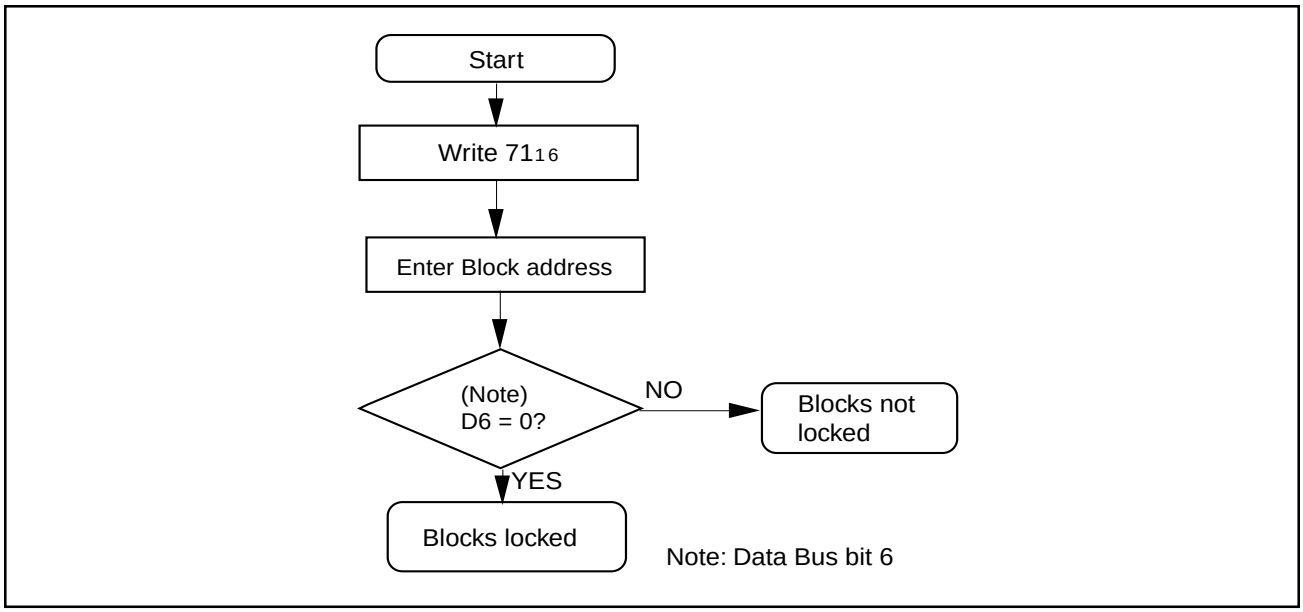



**Fig. 1.155. Lockbit Status Read Flowchart**

### Lock Bit Status Read Command (7116)

This command is available only when the lock bit is enabled. For CPU rewrite mode bit 4 must be a "1". In parallel mode WPB must be a "0". By writing the command code ("7116") in the first bus cycle and reading the Block address and data bit 6 in the second cycle, the value of the lock bit for that block address can be read. A "0" means that the block is locked. A "1" means that the block is unlocked.

Figure 1.155 is an example of a lockbit status read flowchart.

### Data Protect Function (Block Lock)

Each block in Figure 1.156 has a nonvolatile lock bit to specify that the block be protected (locked) against erase or write. The lock bit program command is used to set the lock bit to 0 (locked). The lock bit of against 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 used.

(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 where the lock bit data =1 are not locked, they're enabled for erase/write.

(2) When the lock bit disable bit =1, all blocks are unlocked 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 (unlocked) after erase, so that the lock bit lock is removed.

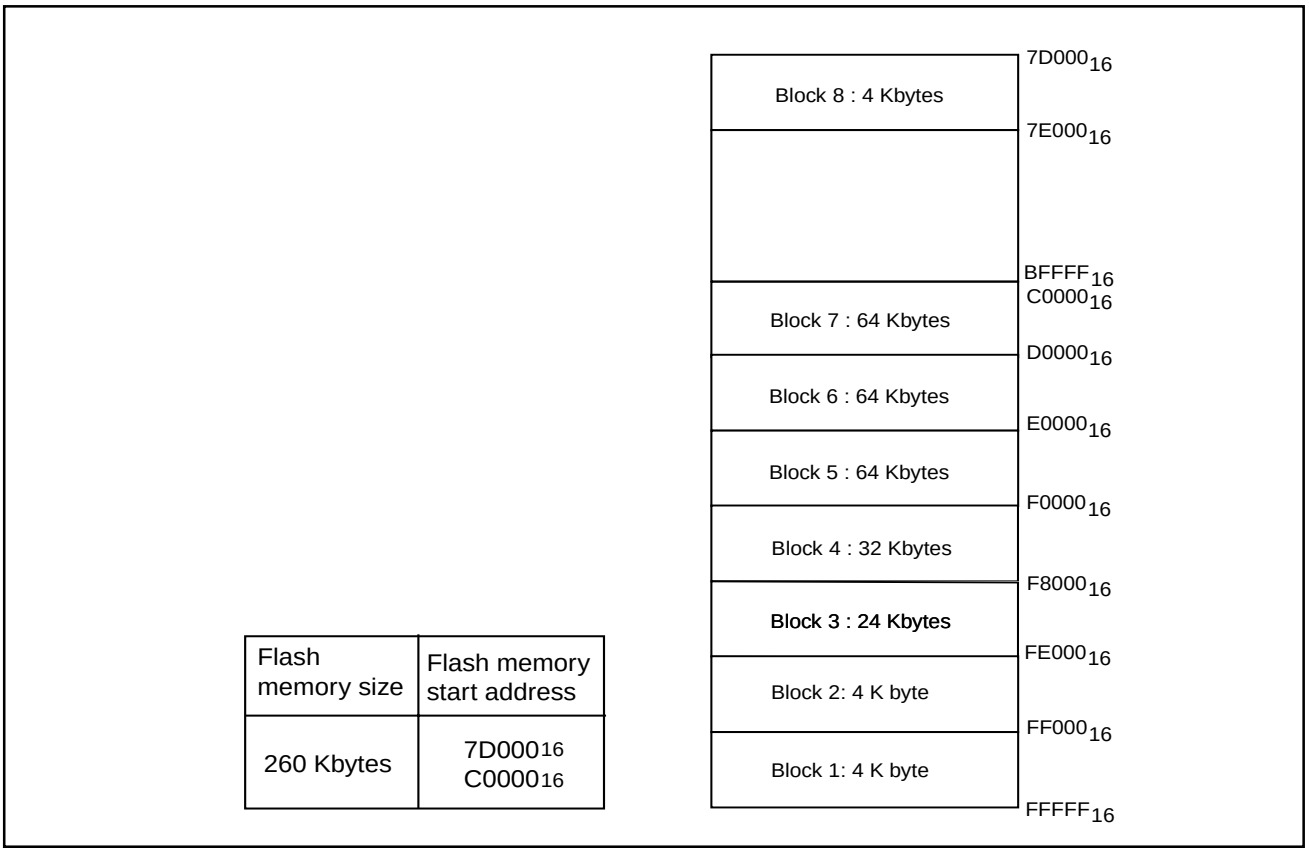



**Fig. 1.156. Block diagram of user area**

## Status Register

The status register shows the operating state of the flash memory and whether erase operations and programs ended successfully or in error. It can be read in the following ways.

(1) By reading an arbitrary address from the user ROM area after writing the read status register command ($70_{16}$)

(2) By reading an arbitrary address from the user ROM area in the period from when the program starts or erase operation starts to when the read array command ($FF_{16}$).

Also, the status register can be cleared by writing the clear status register command ($50_{16}$). After a reset, the status register is set to "$80_{16}$".

**Table 1.76. Definition of each bit in status register**

| Each SRD bit | Status name | Definition | |
|---|---|---|---|
| | | 1 | 0 |
| SR7 (bit 7) | Write state machine (WSM) status | Ready | Busy |
| SR6 (bit 6) | Reserved | _ | _ |
| SR5 (bit 5) | Erase status | Terminated in error | Terminated normally |
| SR4 (bit 4) | Program status | Terminated in error | Terminated normally |
| SR3 (bit 3) | Reserved | _ | _ |
| SR2 (bit 2) | Over write back status | Terminated in error | Terminated normally |
| SR1 (bit 1) | Over erase status | Terminated in error | Terminated normally |
| SR0 (bit 0) | Reserved | _ | _ |

Each bit in this register is explained below.

### Sequencer status (SR7)

After power-on, the sequencer status is set to 1 (ready). The sequencer status indicates the operating status of the device. This status bit is set to 0 (busy) during write or erase operation and is set to 1 upon completion of these operations.

### Erase status (SR5)

The erase status informs the operating status of 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 write operation to 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 the 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 unlocked 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 busy cycle is cancelled.

### Over Write Back Status (SR2)

The over write back status informs the operating status of the write back operation during an erase sequence. When a write back error occurs, SR5 and SR2 are set to "1".
The over write back status is reset to "0" when cleared.

### Over Erase Status (SR1)

The over erase status informs the operating status of the erase operation during an erase sequence. When an erase error occurs, SR5 and SR1 are set to "1". The over erase status is reset to "0" when cleared.

If SR5 or SR4 bits = "1", the program, erase all blocks, and block erase commands are not accepted. Before executing these commands, execute the clear status register command ($50_{16}$) and clear the status register. Bits SR2 and SR1 give more information on the reason for failure.
Also, if any commands are not correct, both SR5 and SR4 are set to 1.

### Full Status Check

By performing full status check, it is possible to know the execution results of erase and program operations. Figure 1.157 shows a full status check flowchart and the action to be taken when each error occurs.

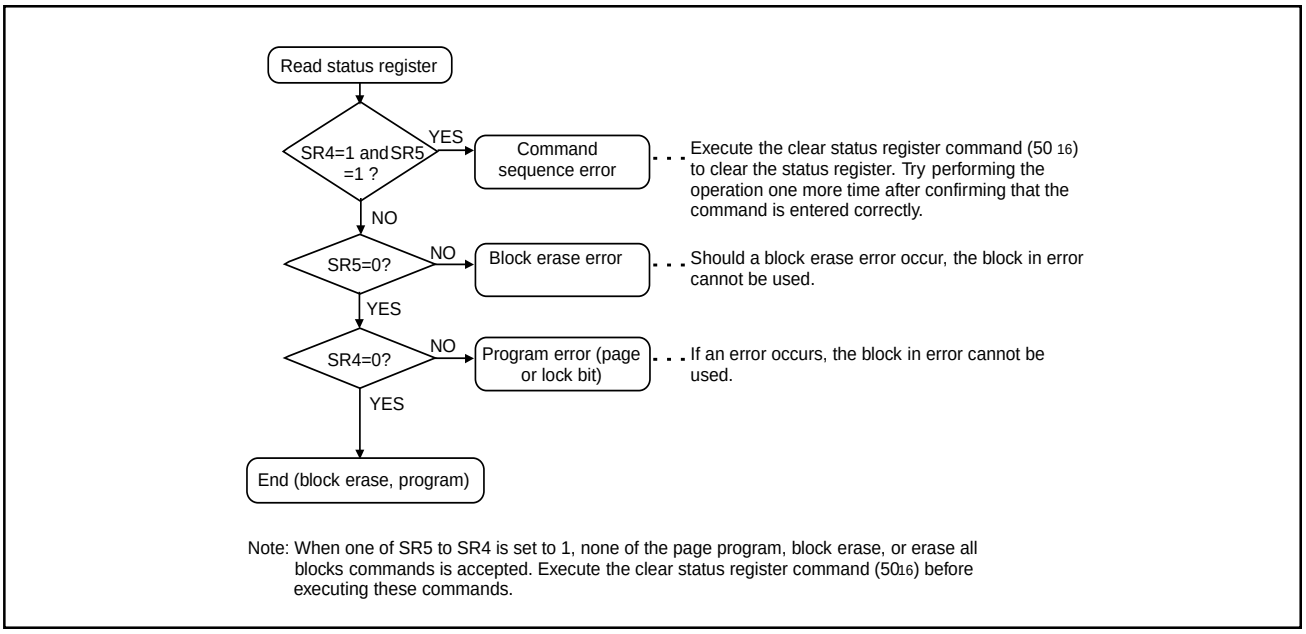



**Fig. 1.157 Full status check flowchart and remedial procedure for errors**

## Functions to Inhibit Rewriting Flash Memory Version

To prevent the contents of the flash memory version from being read out or rewritten easily, the device incorporates a ROM code protect function for use in parallel I/O mode and an ID code check function for use in standard serial I/O mode.

### ROM code protect register

The ROM code protect function prevents reading out or modifying the contents of the flash memory during parallel I/O mode. Figure 1.158 shows the ROM code protect control address ($0FFFFF_{16}$). It is located at the hightest 8 bits of the 32 bit reset vector.

If one of the pair of ROM code protect bits is set to "0", ROM code protect is turned on, so that the contents of the flash memory version are protected against readout and modification. ROM code protect is implemented in two levels. If level 2 is selected, the flash memory is protected even against readout by a shipment inspection LSI tester, etc. When an attempt is made to select both level 1 and level 2, level 2 is selected by default.

If both of the two ROM code protect reset bits are set to "00", ROM code protect is turned off, so that the contents of the flash memory version can be read out or modified. Once ROM code protect is tuned on, the contents of the ROM code protect reset bits cannot be modified in parallel I/O mode. Use the serial I/O or some other mode to rewrite the contents of the ROM code protect reset bits.

ROM code protect control address

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

Symbol: ROMCP
Address: $0FFFFF_{16}$
When reset: $FF_{16}$

| Bit symbol | Bit name | Function |
|---|---|---|
| Reserved bit | | Always set this bit to "1" |
| ROMCP2 | ROM code protect level 2 set bit (Note 1, 2) | b3 b2<br>00: Protect enabled<br>01: Protect enabled<br>10: Protect enabled<br>11: Protect disabled |
| ROMCR | ROM code protect reset bit (Note 3) | b5 b4<br>00: Protect removed<br>01: Protect set bit effective<br>10: Protect set bit effective<br>11: Protect set bit effective |
| ROMCP1 | ROM code protect level 1 set bit (Note 1) | b7 b6<br>00: Protect enabled<br>01: Protect enabled<br>10: Protect enabled<br>11: Protect disabled |

Note 1: When ROM code protect is turned on, the on-chip flash memory is protected against readout or modification in parallel input/output mode.
Note 2: When ROM code protect level 2 is turned on, ROM code readout by a shipment inspection LST tester, etc., is also inhibited.
Note 3: The ROM code protect reset bits can be used to turn off ROM code protect level 1 and ROM code protect level 2. However, because these bits cannot be changed in paralell input/output mode, they need to be rewritten in one of the two other modes.

**Fig. 1.158. ROM code protect control addrsss**

### ID Code Check Function

Use this function in standard serial I/O mode. When the contents of the flash memory are not blank, the ID code sent from the peripheral unit is compared with the ID code written in the flash memory to see if they match: If the ID codes do not match, the commands sent from the peripheral unit are not accepted. The ID code consists of 8-bit data, the areas of which, beginning with the first byte, are $0FFFDF_{16}$, $0FFFE3_{16}$, $0FFFEB_{16}$, $0FFFEF_{16}$, $0FFFF3_{16}$, $0FFFF7_{16}$, and $0FFFFB_{16}$. Write a program which has had the ID code preset at these addresses to the flash memory.

Figure 1.159 shows the storage location for the code addresses.

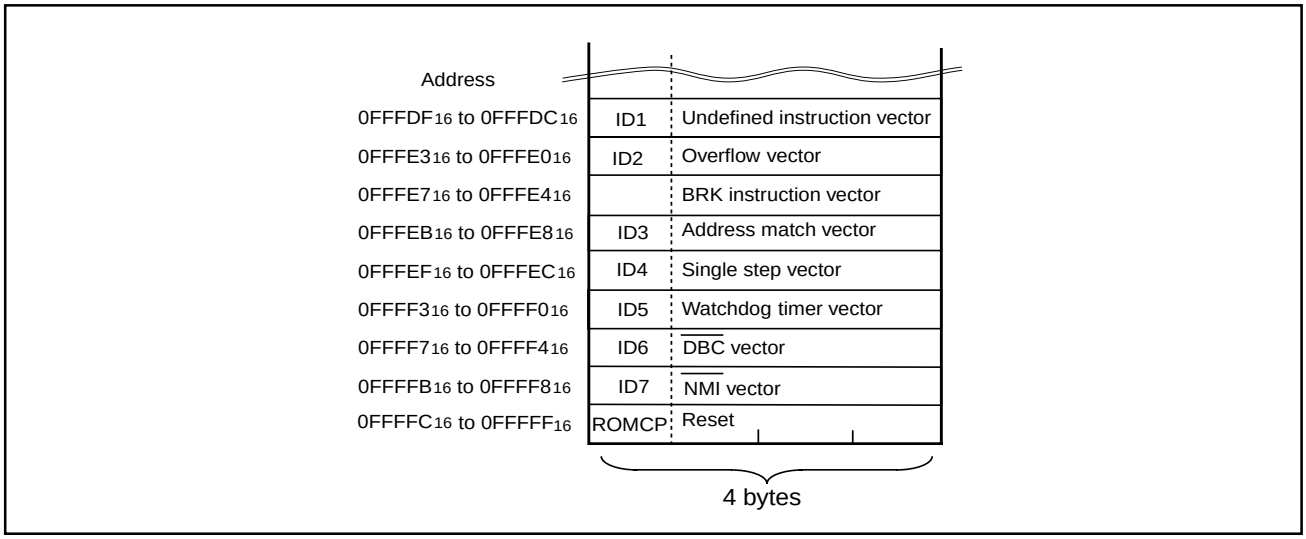



**Fig. 1.159. Code address storage**

## Parallel I/O Mode

The parallel I/O mode inputs and outputs the software commands, addresses and data needed to operate (read, program, erase, etc.) the internal flash memory. Figure 1.160 shows the pin layout for flash paralell mode. Table 1.77 is adescription of pin functions in paralell I/O mode.

Use an exclusive programmer supporting M30222 (flash memory version). Refer to the instruction manual of each programmer make for the detail of use.

### User ROM and Boot ROM Areas

In parallel I/O mode, the user ROM and boot ROM areas shown in Figure 1.149 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 1.149.

The boot ROM area is 8 Kbytes in size. In parallel I/O mode, it is located at addresses $07E000_{16}$ through $07FFF_{16}$. 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.

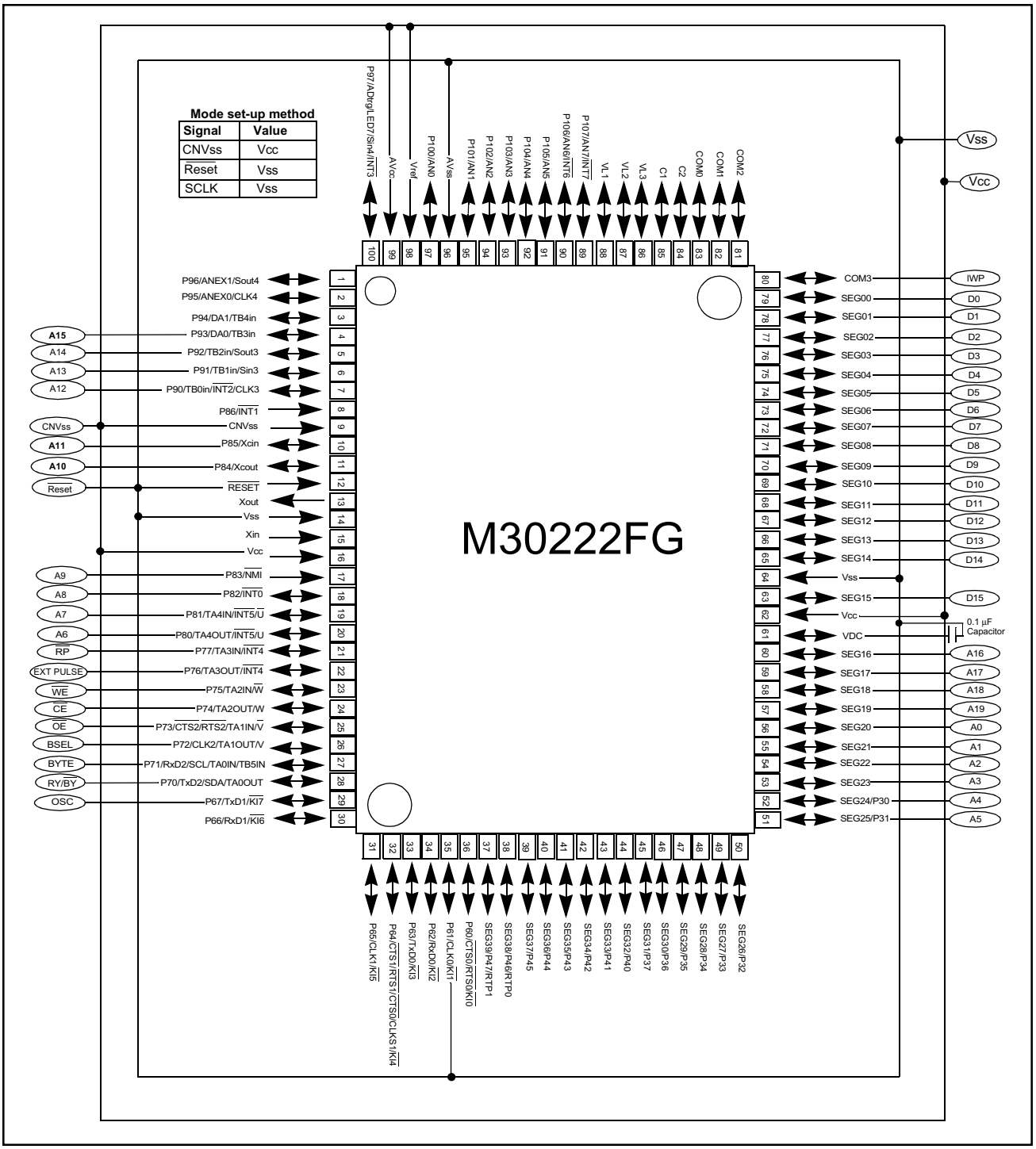



**Fig. 1.160. Pin Connections for Flash Parallel Mode**

**Table 1.77. Description of pin function in parallel I/O mode**

| Pin Name | Signal name | I/O | Function |
|---|---|---|---|
| Vcc, Vss | Power input | I | Apply 2.7-5.5 V to Vcc pin and O V to Vss pin |
| CNVss | CNVss | I | Connect 0.1 μf capacitor from Vcc to Vss |
| $\overline{\text{RESET}}$ | Reset input | I | Connect to Vss |
| Xin/Xout | Clock input | I | No connection required for parallel flash programming |
| AVcc, AVss | Analog power supply | I | Connect AVss to Vss and AVcc to Vcc |
| $P3_0$ | Address Bit 4 | I | Address Bit 4 |
| $P3_1$ | Address Bit 5 | I | Address Bit 5 |
| $P3_2$ to $P3_7$ | Input port P3 | | No connection required for parallel flash programming |
| $P4_0$ to $P4_7$ | Input port P4 | | No connection required for parallel flash programming |
| $P6_0$ | Input port P6 | | No connection required for parallel flash programming |
| $P6_1$ | SCLK | I | Connect to Vss |
| $P6_2$ to $P6_5$ | Input port P6 | | No connection required for parallel flash programming |
| $P6_7$ | OSC | O | Flash oscillator |
| $P7_0$ | RY/$\overline{\text{BY}}$ | O | Ready / Busy signal |
| $P7_1$ | BYTE | I | Byte mode control mode |
| $P7_2$ | BSEL input | I | Boot select mode |
| $P7_3$ | $\overline{\text{OE}}$ input | I | Output enable pin |
| $P7_4$ | $\overline{\text{CE}}$ input | I | Chip enable pin |
| $P7_5$ | $\overline{\text{WE}}$ input | I | Write enable pin |
| $P7_6$ | EXTPULSE | I | External pulse for test modes |
| $P7_7$ | $\overline{\text{RP}}$ | I | Deep power down pin |
| $P8_0$ | Address Bit 6 | I | Address Bit 6 |
| $P8_1$ | Address Bit 7 | I | Address Bit 7 |
| $P8_2$ | Address Bit 8 | I | Address Bit 8 |
| $P8_3$ | Address Bit 9 | I | Address Bit 9 |
| $P8_4$ | Address Bit 10 | I | Address Bit 10 |
| $P8_5$ | Address Bit 11 | I | Address Bit 11 |

| Pin Name | Signal name | I/O | Function |
|---|---|---|---|
| $P9_0$ | Address Bit 12 | I | Address Bit 12 |
| $P9_1$ | Address Bit 13 | I | Address Bit 13 |
| $P9_2$ | Address Bit 14 | I | Address Bit 14 |
| $P9_4$ to $P9_5$ | Input port 9 | | No connection required for parallel flash programming |
| $P9_7$ | Input port 9 | | No connection required for parallel flash programming |
| Vref | A/D Vref voltage | I | Connect A/D reference voltage to Vcc |
| $P10_0$ to $P10_7$ | Input port P10 | | Input H , L or leave open |
| VL1 to VL3 | LCD power supply | I | Connect VL1 to Vss; VL2 and VL3 to Vcc |
| C1 to C2 | LCD condenser | | Leave open |
| COM0 to COM2 | COM ports | | Leave open |
| COM3 | IWP | I | Write protect pin |
| SEG0 to SEG15 | Data Bits 0 - 15 | I/O | Data Bits 0 - 15 |
| SEG16 | Address Bit 16 | I | Address Bit 16 |
| SEG17 | Address Bit 17 | I | Address Bit 17 |
| SEG18 | Address Bit 18 | I | Address Bit 18 |
| SEG19 | Address Bit 19 | I | Address Bit 19 |
| SEG20 | Address Bit 0 | I | Address Bit 0 |
| SEG21 | Address Bit 1 | I | Address Bit 1 |
| SEG22 | Address Bit 2 | I | Address Bit 2 |
| SEG23 | Address Bit 3 | I | Address Bit 3 |

## 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. There are two standard serial I/O modes that require a purpose specific peripheral unit.

- Serial I/O Mode 1 is synchronized
- Serial I/O Mode 2 is as asynchronized

The standard serial I/O mode is different from the parallel I/O mode because it uses the CPU rewrite mode to control flash memory rewrite, rewrite data input and so on. It is started when the reset is released. This is done when the $P5_0$ (CE) pins is “H” level, the $P5_5$ (EPM) pin “L” level and the CNVss pin “H” level. In an ordinary command mode, the CNVss pin is set to “L” level.

This control program is written in the boot ROM area when the product is shipped from Mitsubishi. Please note that the standard serial I/O mode cannot be used if the boot ROM area is rewritten in parallel I/O mode. In standard serial I/O mode, only the user ROM area (see Figure 1.181) can be rewritten. The boot ROM cannot.

Also, a 7-byte ID code is used. When there is data in the flash memory, commands sent from the peripheral unit are not accepted unless the ID code matches.

Figure 1.161 shows 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 and mode 2 according to the level of CLK1 pin when the reset is released.

### Serial I/O Mode 1

To use standard serial I/O mode 1, set the CLK1 pin to “H” level and release the reset. The operation uses the four UART1 pins CLK1, RxD1, TxD1 and RTS1 (BUSY). the CLK1 pin is the transfer clock input pin through which an external transfer clock is input. The TxD1 pin is for CMOS output. the RTS1 (BUSY) pin outputs an “L” level when ready for reception and “H” level when reception starts.

### Serial I/O Mode 2

To use standard serial I/O mode 2, set the CLK1 pin to “L” level and release the reset. The operation uses the two UART1 pins RxD1 and TxD1.

## Overview of standard serial I/O mode 1 (clock synchronous)

In standard serial I/O mode 1, software commands, addresses and data are input and output between the MCU and peripheral units (serial programmer, etc.) using 4-wire clock-synchronous serial I/O (UART1). Standard serial I/O mode 1 is engaged by releasing the reset with the $P5_6$ (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 $CLK_1$ pin, and are then input to the MCU via the $RxD_1$ pin. In transmission, the read data and status are synchronized with the fall of the transfer clock, and output from the $TxD_1$ pin.

The $TxD_1$ pin is for CMOS output. Transfer is in 8-bit units LSB first.

When busy, such as during transmission, reception, erasing or program execution the $RTS_1$ (BUSY) pin is "H" level. Accordingly, always start the next transfer after the $RTS_1$ (BUSY) pin is "L" level.

Also, data and status register 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.

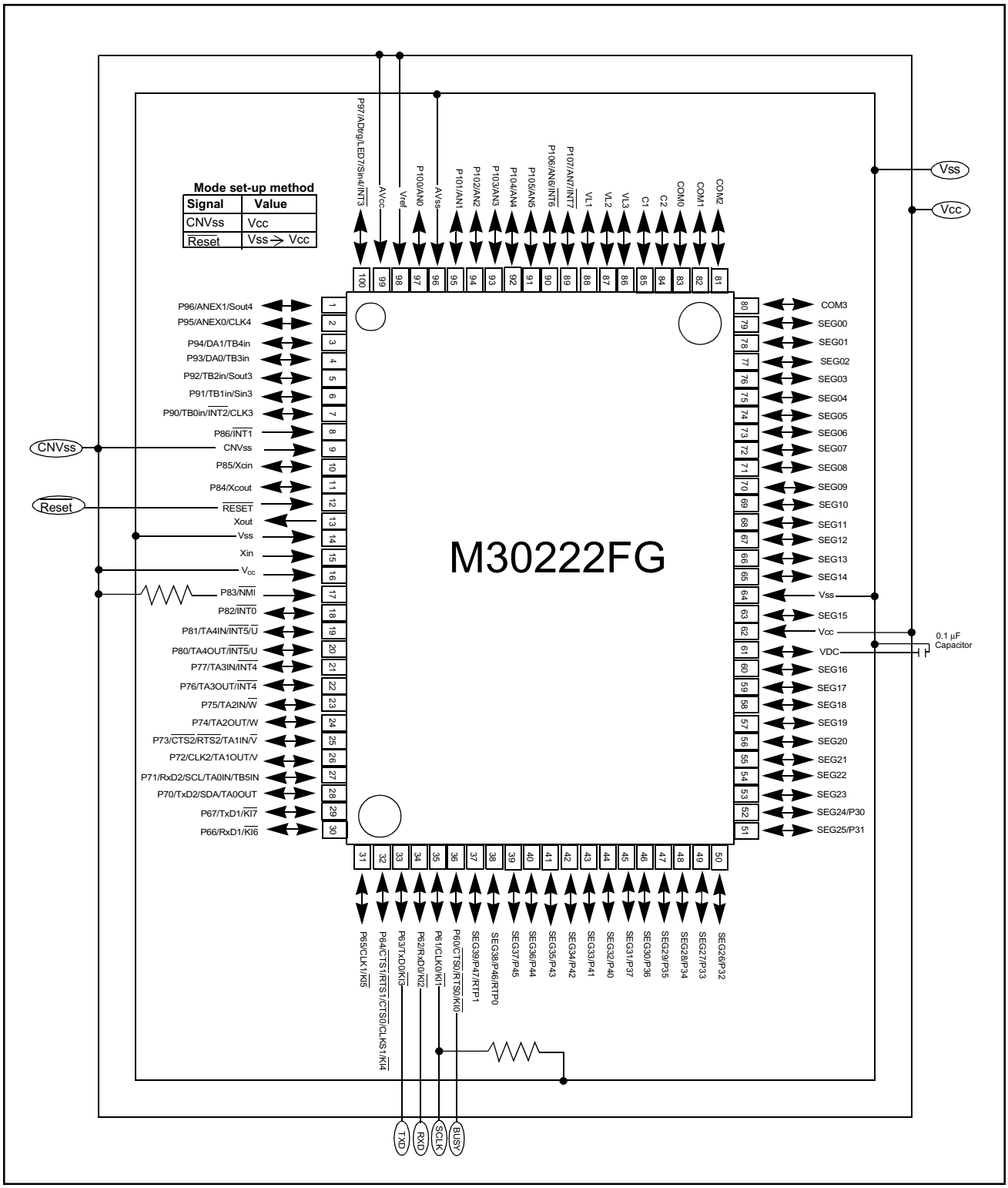



**Fig. 1.161. Pin connections for Flash Serial I/O mode**

**Table 1.78. Pin connections for serial I/O mode**

| Pin Name | Signal name | I/O | Function |
|---|---|---|---|
| Vcc, Vss | Power input | I/O | Apply 2.7 to 5.5 V to Vcc pin and 0 V to the Vss pin |
| CNVss | CNVss | I | Connect to Vcc |
| $\overline{\text{RESET}}$ | Reset input | I | Connect Vss |
| Xin/Xout | 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 and open Xout pin |
| AVcc, AVss | Analog power supply | I | Connect AVss to Vss and AVcc to Vcc |
| $P3_0$ to $P3_7$ | Input port P3 | I/O | Input H , L , or leave open |
| $P4_0$ to $P4_7$ | Input port P4 | I/O | Input H , L , or leave open |
| $P6_0$ | Busy | I/O | Standard serial mode 1: Busy signal output pin<br>Standard serial mode 2: Monitors program operation check. |
| $P6_1$ | SCLK | I/O | Standard serial mode 1: Serial clock input pin<br>Standard serial mode 2: Input L |
| $P6_2$ | RxD input | I/O | Serial data input pin |
| $P6_3$ | TxD output | I/O | Serial data output pin |
| $P6_4$ to $P6_7$ | Input port P6 | I/O | Input H , L , or leave open |
| $P7_0$ to $P7_7$ | Input port P7 | I/O | Input H , L , or leave open |
| $P8_0$ to P86 | Input port P8 | I/O | Input H , L , or leave open |
| $P9_0$ to $P9_7$ | Input port P9 | I/O | Input H , L , or leave open |
| $P10_0$ to $P10_7$ | Input port P10 | I/O | Input H , L , or leave open |
| Vref | A/D Vref voltage | I | Input A/D reference voltage |
| VL1 to VL3 | LCD power supply | I/O | Connect VL1 to Vss; VL2 and VL3 to Vcc when LCD is not used |
| C1 to C2 | LCD condenser | I/O | Connect a condenser between C1 and C2 when using LCD voltage multiplier. Leave open when not used. |
| COM0 to COM3 | COM ports | I/O | Leave open |
| VDC | Voltage Down Converter | | 0.1µF capacitor connect to Vss |

## Software Commands

Table 1.79 lists software commands. In the standard serial I/O mode 1, erase operations, programs and reading are controlled by transferring software commands via the RxD0 pin. Software commands are explained here below.

**Table 1.79. Software commands (Standard serial I/O mode 1)**

| | Control command | | 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 | SRD 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 | ID check function | $F5_{16}$ | Address (low) | Address (middle) | Address (high) | ID size | ID1 | ID7 | Acceptable |
| 12 | Download function | $FA_{16}$ | Size (low) | Size (high) | Check sum | Data input | As required | | 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 | $FD_{16}$ | CRC data (Low) | CRC data (High) | | | | | Not acceptable |
| 16 | Word Read | $FE_{16}$ | Address (low) | Address (middle) | Address (high) | | | | Not acceptable |
| 17 | Word Program | $40_{16}$ | Address (low) | Address (middle) | Address (high) | | | | Not acceptable |
| 18 | Exit | $B9_{16}$ | | | | | | | Acceptable |

Note 1: The shaded areas indicate a transfer from flash MCU to serial programmer. All other data is transferred from programmer to MCU.
Note 2: SRD to Status Register Data. SRD1 refers to Status Register 1 Data.
Note 3: All commands are accepted if the reset vector is blank.

### (1) 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 $A_{23}$ 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 address $A_8$ to $A_{23}$ will be output sequentially from the smallest address first in sync with the rise of the clock.

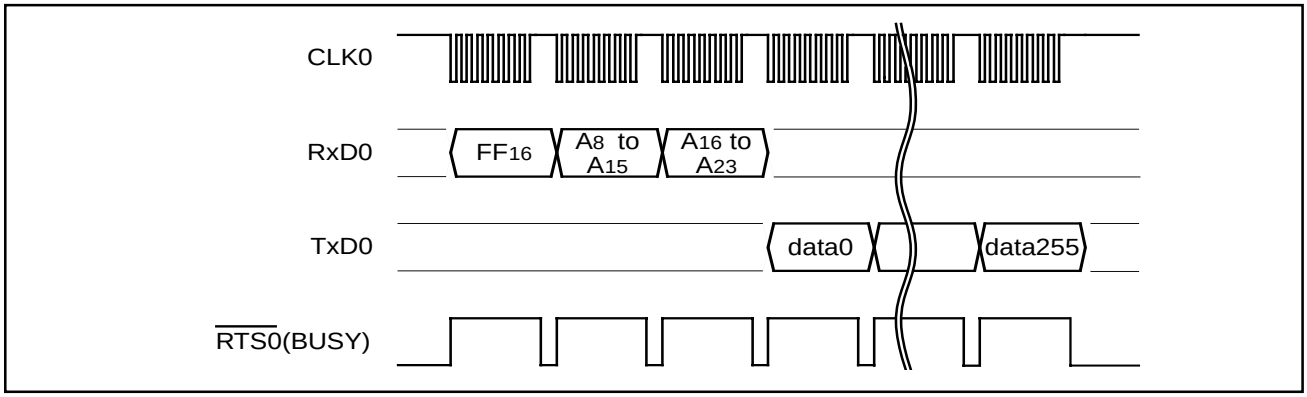



**Fig. 1.162. Timing for page read**

### (2) 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.

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

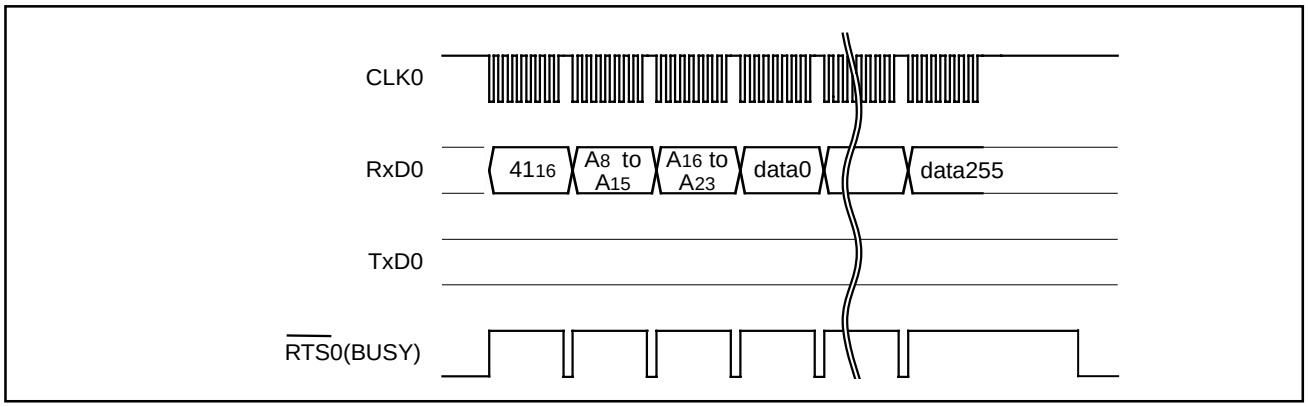



**Fig. 1.163. Timing for the page program**

### (3) 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 A8 to A15 and A16 to A23 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 the specified block for addresses $A_8$ to $A_{23}$.

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

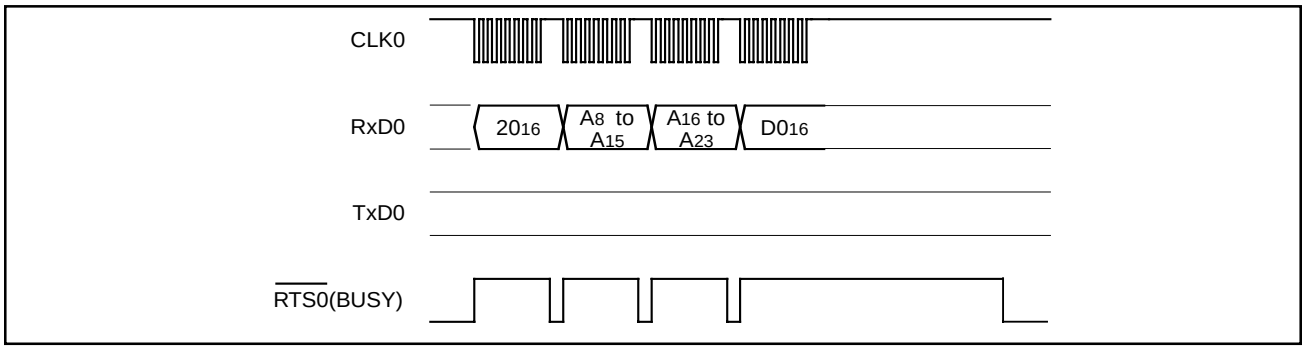



**Fig. 1.164. Timing for block erase**

### (4) Erase All Unlocked Blocks Command

This command erases the content of all blocks. Execute the erase all unlocked blocks command explained below.

(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 erase ends, the RTS0 (BUSY) signal changes from "H" to "L". 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 Data Protection Function section.

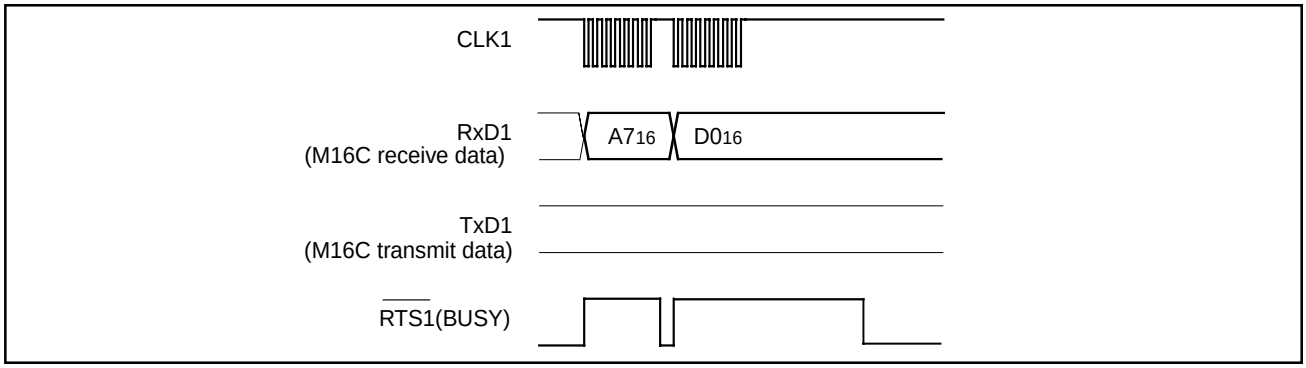



**Fig.1.165. Timing for erasing all unlocked blocks**

### (5) 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.

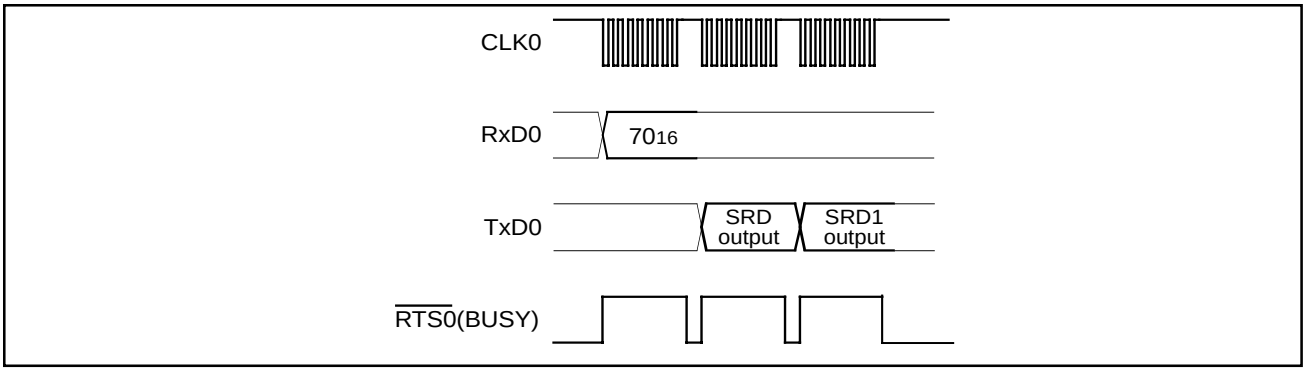



**Fig. 1.166. Timing for reading the status register**

### (6) Clear Status Register Command

This command clears the bits (SR4 - SR5) which are set when the status register operation register operation ends in error. When the "5016" command code is sent with thefirst byte, the aforementioned bits are cleared. When the clear status register operation ends, the RTS0 (Busy) signal changes from the "H" to the "L" level.

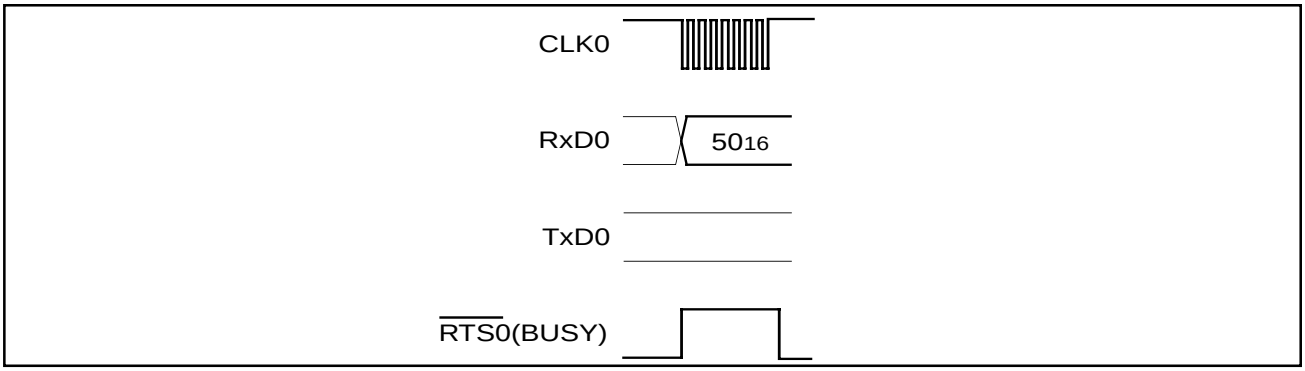



**Fig. 1.167. Timing for clearing the status register**

### (7) Read Lock Bit Status Command

This command reads the lock bit status of the specified block. Execute the lock bit program 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. Write the highest address of the specified block for address $A_8$ to $A_{23}$.

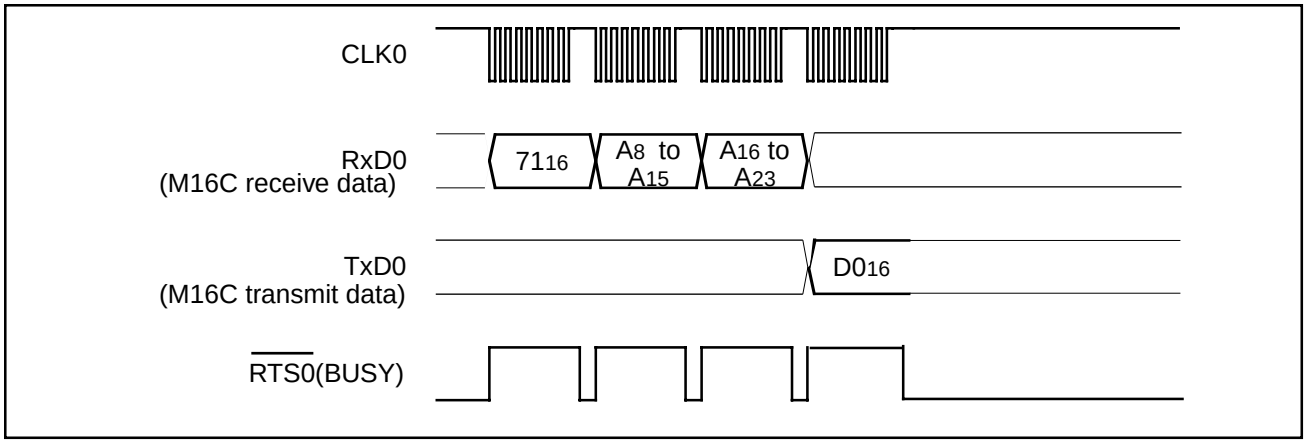



**Fig. 1.168. Timing for reading lock bit status**

### (8) 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.

(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 address $A_8$ to $A_{23}$.

When writing ends, RTS0 (BUSY) signal changes from the "H" to "L". Lock bit status can be read with the read lock bit status command. For information on the lock bit function an reset procedure, see the Data Protection Function section.

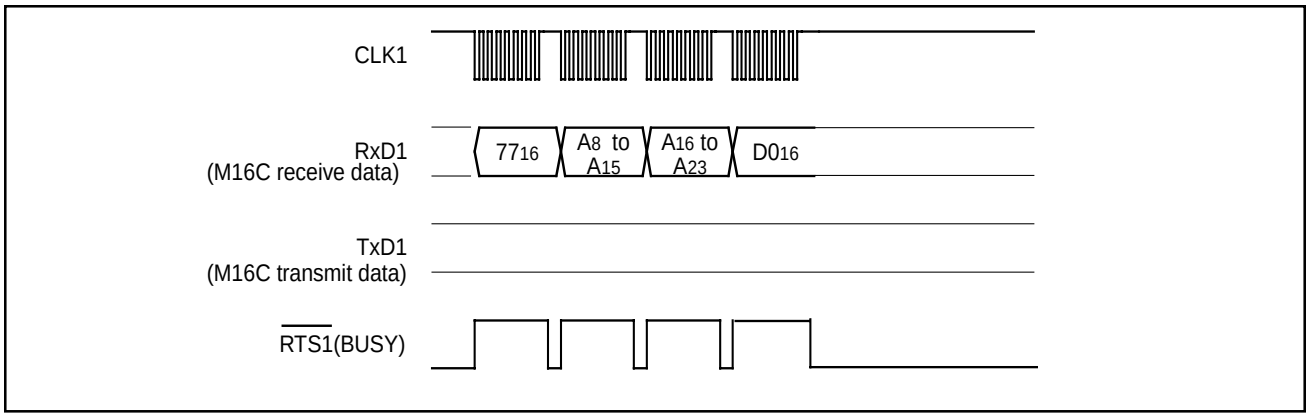



**Fig. 1.169. Timing for the lock bit program**

### (9) 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 "7A16" 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.

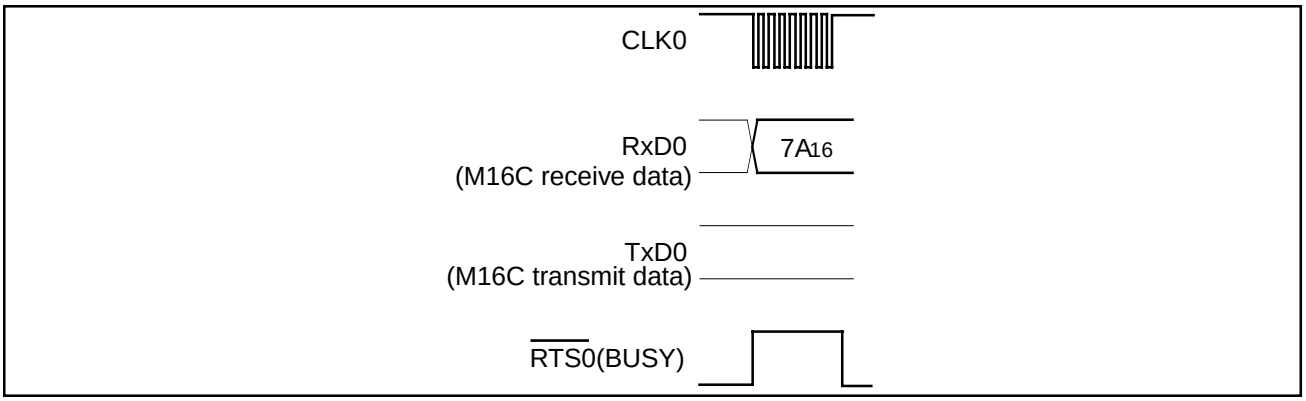



**Fig. 1.170. Timing for enabling the lock bit**

### (10) Lock Bit Disable Command

This command disables the lock bit. The command code "7516" 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.

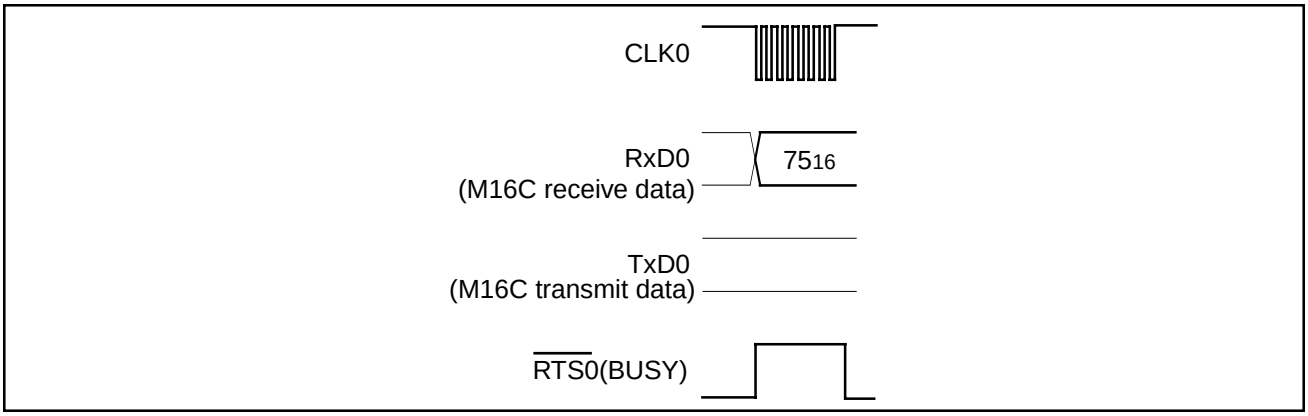



**Fig. 1.171. Timing for disabling the lock bit**

### (11) 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.

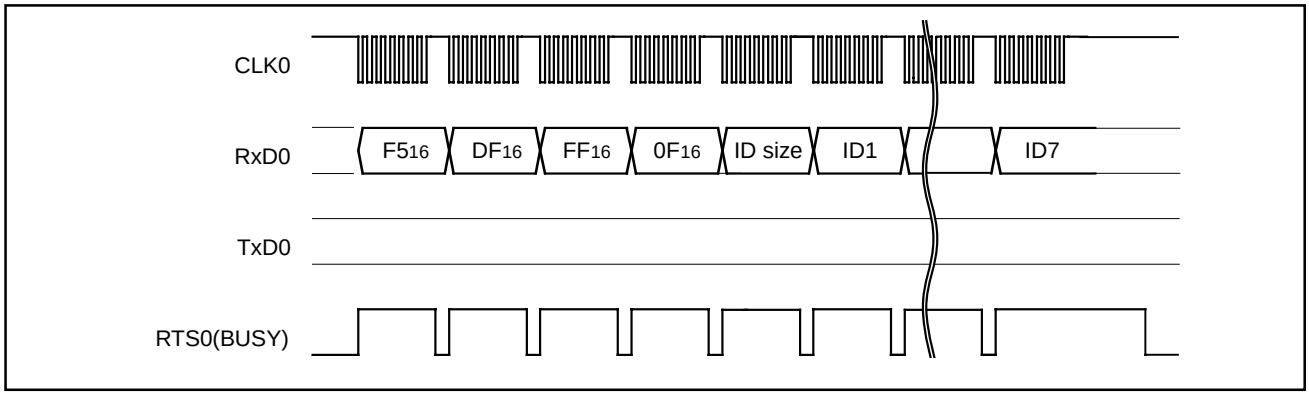



**Fig. 1.172. Timing for the ID check**

### 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 0FFFDF$_{16}$, 0FFFE3$_{16}$, 0FFFEB$_{16}$, 0FFEF$_{16}$, 0FFFF3$_{16}$, 0FFF7$_{16}$, and 0FFFFB$_{16}$. Write a program into the flash memory, which already has the ID code set for these addresses.

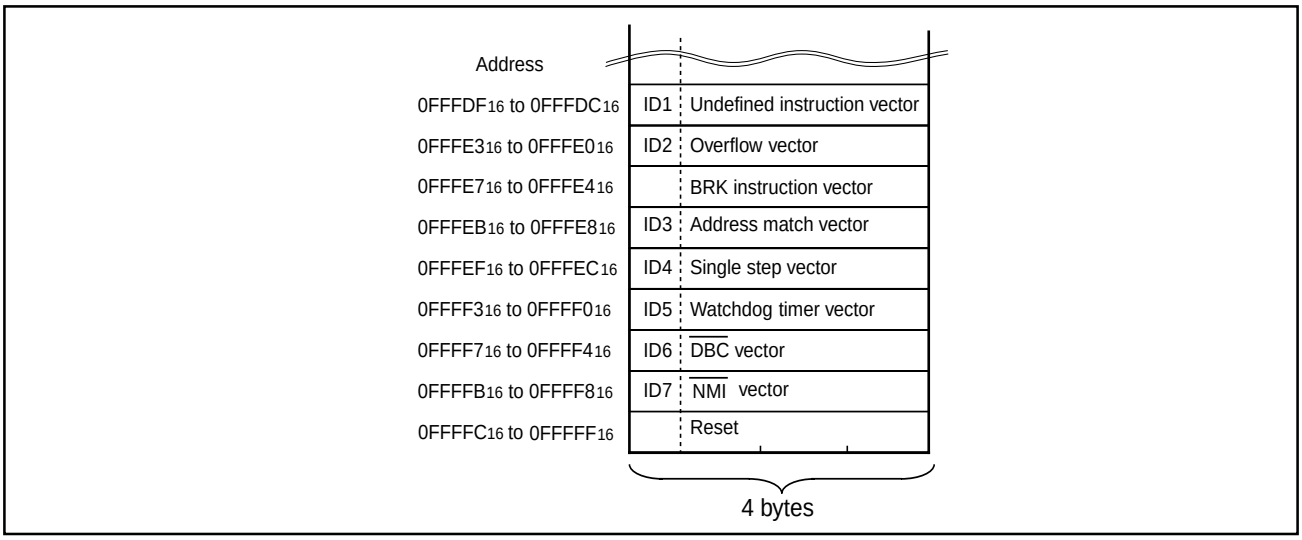



**Fig. 1.173. ID code storage addresses**

### (12) 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.

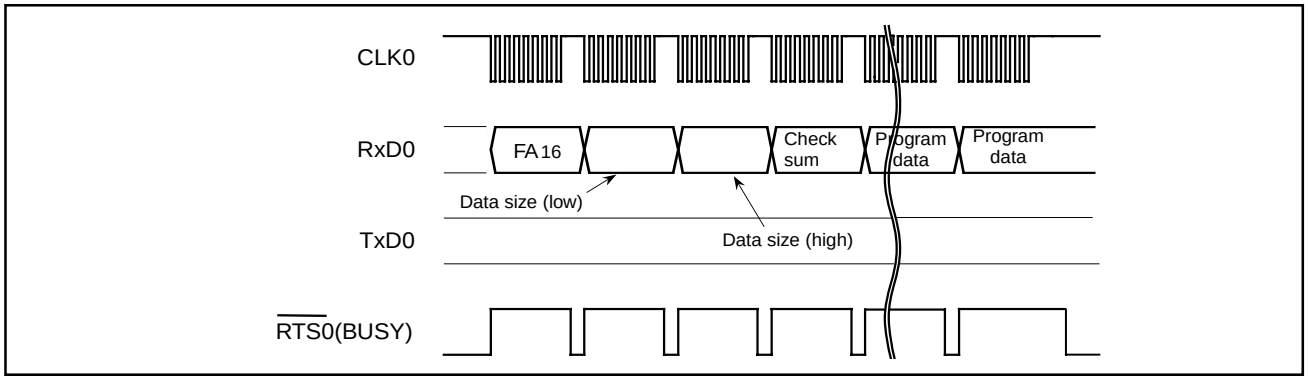



**Fig. 1.174. Timing for download**

### (13) 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.

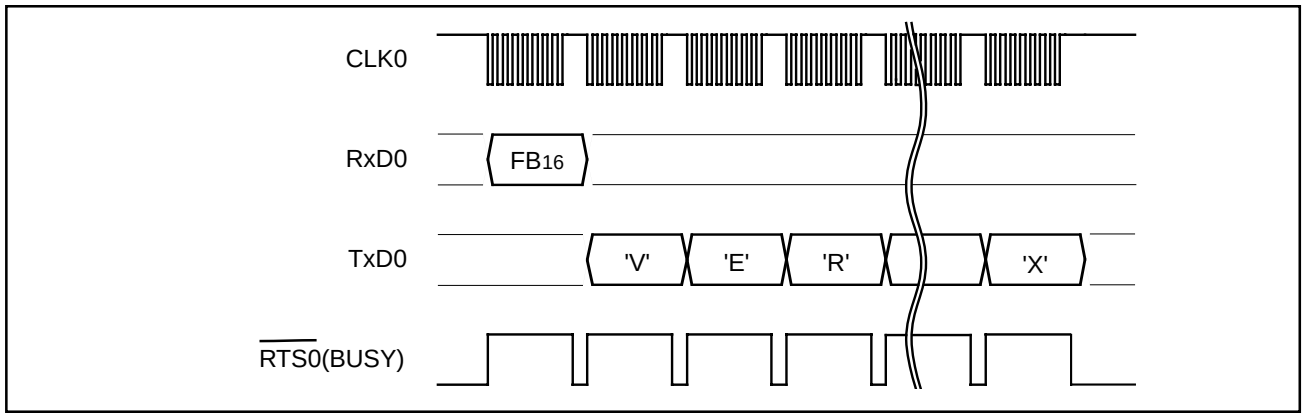



**Fig. 1.175. Timing for version information output**

### (14) 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.

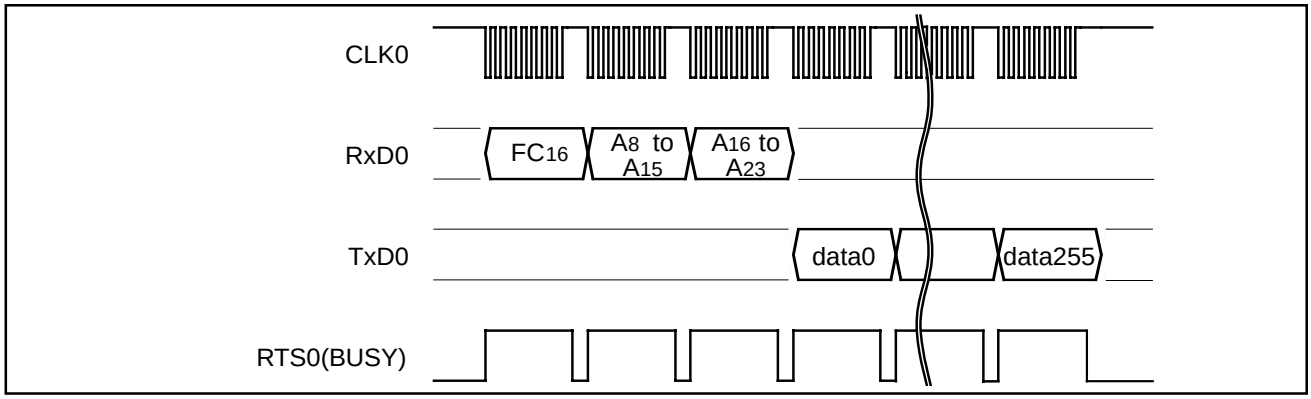



**Fig. 1.176. Timing for boot ROM area ouput**

### (15) Read CRC Data

This command reads the CRC data that confirms that the write data sent with the page program command was successfully received.

(1) Transfer the "$FD_{16}$" command code with the 1st byte.

(2) The CRC data (low) is received with the 2nd byte and the CRC data (high) with the 3rd byte.

To use this command, first execute the command and then initialize the check data. Next, execute the page program command the required number of times. After the read CRC command is executed again, the CRC data for all the read data that was sent with the page program command is read. The CRC data is the result of CRC operation of write data.

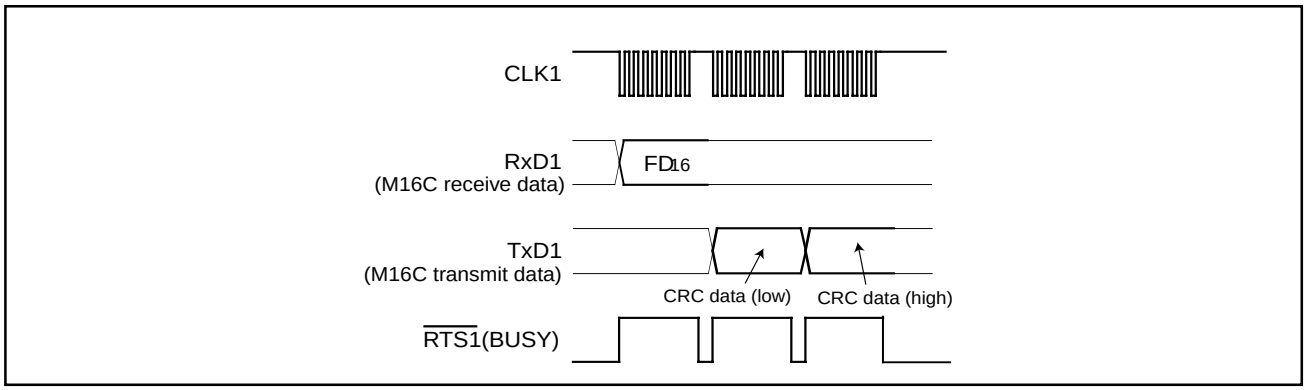



**Fig. 1.177. Timimg for the read CRC data**

### (16) Word Read Program

This command reads the word from the specific address. To execute the word read command:

(1) Transfer the "FE16" command code with the 1st byte.

(2) Transfer the 3 byte address starting with the A0-A8 with the next 3 bytes.

(3) The MCU transfers the byte at the specific address.

(4) The MCU transfers the byte at the specific address +1.

Note: The specified address may be odd or even (A0=0 or 1) and be any value withing the 1M address space.

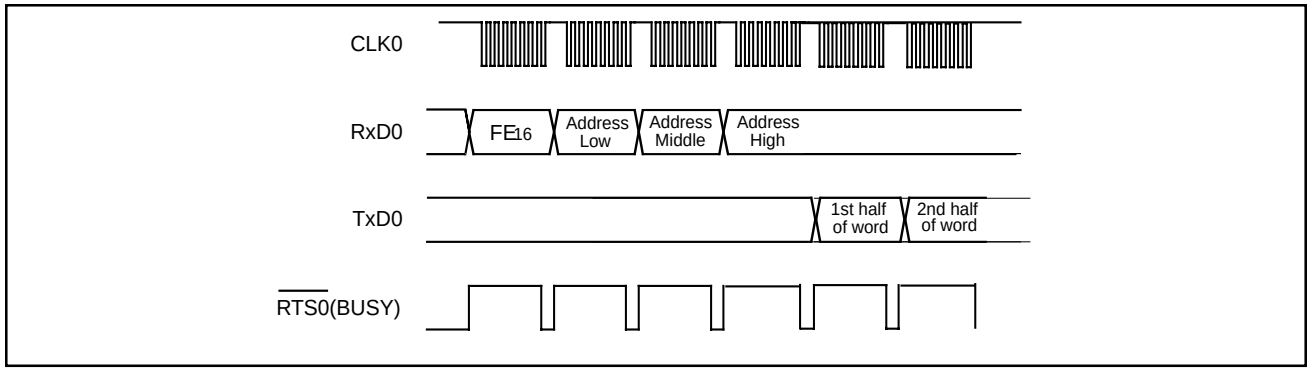



**Fig. 1.178. Timimg for the word read program**

### (17) Word Program

This command writes the word in the flash memory. To execute the word program command:

(1) Transfer the "4016" with the 1st byte.

(2) Transfer the 3 byte address starting with the A0-A8 with the next 3 bytes.

(3) Transfer the 1st half of the word to be written in the lower address (A0=0).

(4) Transfer the 2nd half of the word to be written in the higher address (A0=1).

Note: The specified address must be even (A0=0).

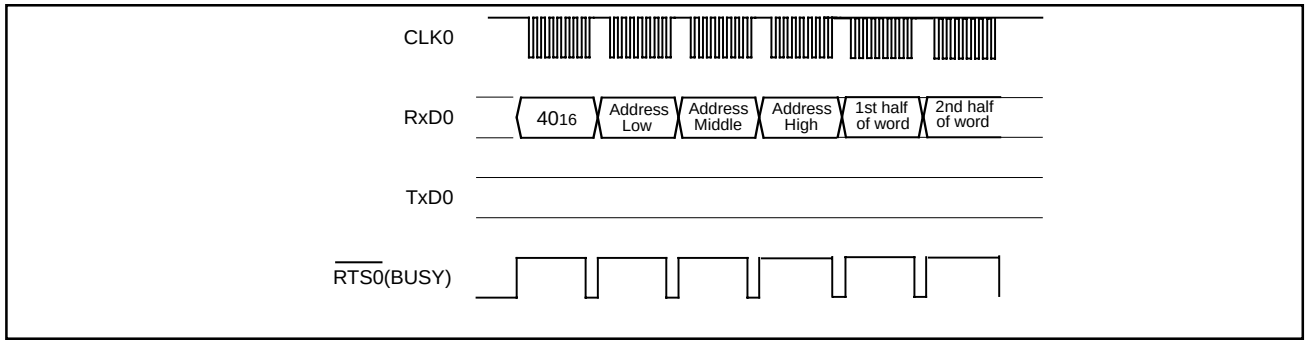



**Figure 1.179. Timing for the word program**

### (18) Exit Command

This command does a software reset by writing a "1" to bit 3 of the Processor Mode register 0. To execute the Exit command:

(1) Transfer "B9$_{16}$" with the 1st byte.

(2) Transfer the confirm command code "D0$_{16}$" in the 2nd byte.

If the CNVss line is low, the MCU will reset in normal mode and begin execution of the user code. If CNVss is high, the MCU will reset back into boot mode and begin execution at 7E000$_{16}$ again.

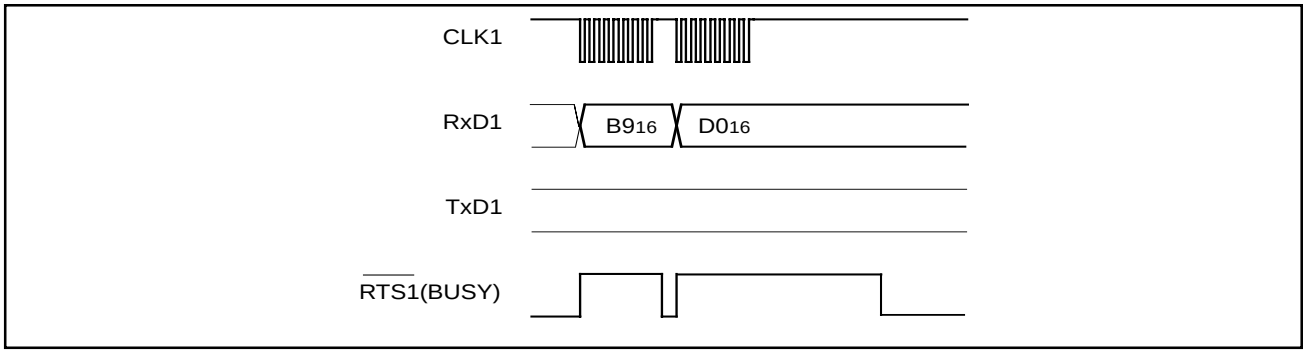



**Figure 1.180.. Timing for the exit command**

### Data Protection (Block Lock)

Each of the blocks in Figure 1.181 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 executing status of the lock bit disable and lock 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 e 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.

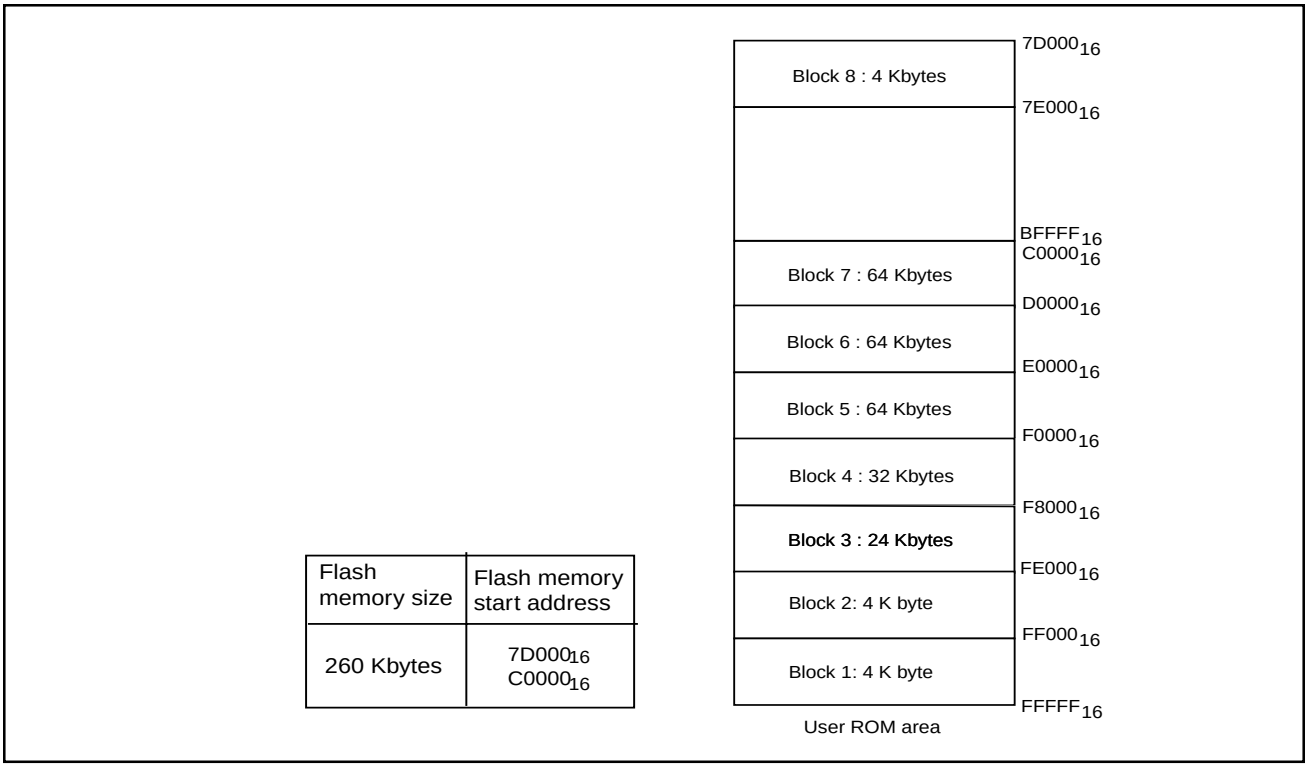



**Fig. 1.181. Block in the user area**

## 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 1.80 gives the definition of each status register bit. After clearing the reset, the status register outputs "$80_{16}$".

**Table 1.80. Status register (SRD)**

| Each SRD bit | Status name | Definition | |
|---|---|---|---|
| | | 1 | 0 |
| SR7 (bit 7) | Write state machine (WSM) status | Ready | Busy |
| SR6 (bit 6) | Reserved | _ | _ |
| SR5 (bit 5) | Erase status | Terminated in error | Terminated normally |
| SR4 (bit 4) | Program status | Terminated in error | Terminated normally |
| SR3 (bit 3) | Reserved | _ | _ |
| SR2 (bit 2) | Reserved | _ | _ |
| SR1 (bit 1) | Reserved | _ | _ |
| SR0 (bit 0) | Reserved | _ | _ |

### Sequencer status (SR7)

After power-on, the sequencer status is set to 1 (ready).

The sequencer status indicates the operating status of the device. This status bit is set to 0 (busy) during write or erase operation and is set to 1 upon completion of these operations.

### Erase Status (SR5)

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

### Program Status (SR4)

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

## 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 1.81 gives the definition of each status register 1 bit. "$00_{16}$" is output when power is turned ON and the flag status is mainteined even after the reset.

**Table 1.81. Status register definitions for SRD1**

| SRD 1 bits | Status Name | Definition | |
|---|---|---|---|
| | | 1 | 0 |
| SR15 (bit 7) | Boot update complete bit | Update completed | Not completed |
| SR14 (bit 6) | Reserved | _ | _ |
| SR13 (bit 5) | Reserved | _ | _ |
| SR12 (bit 4) | Checksum match bit | Match | No match |
| SR11 (bit 3)<br>SR10 (bit 2) | ID check completed bits | 0 0 Not verified<br>0 1 Verification mismatch<br>1 0 Reserved<br>1 1 Verified | |
| SR9 (bit 1) | Data receive time out | Time out | Normal operation |
| SR8 (bit 0) | Reserved | _ | _ |

### Boot Update Completed Bit (SR15)

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

### Check Sum Consistency Bit (SR12)

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

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

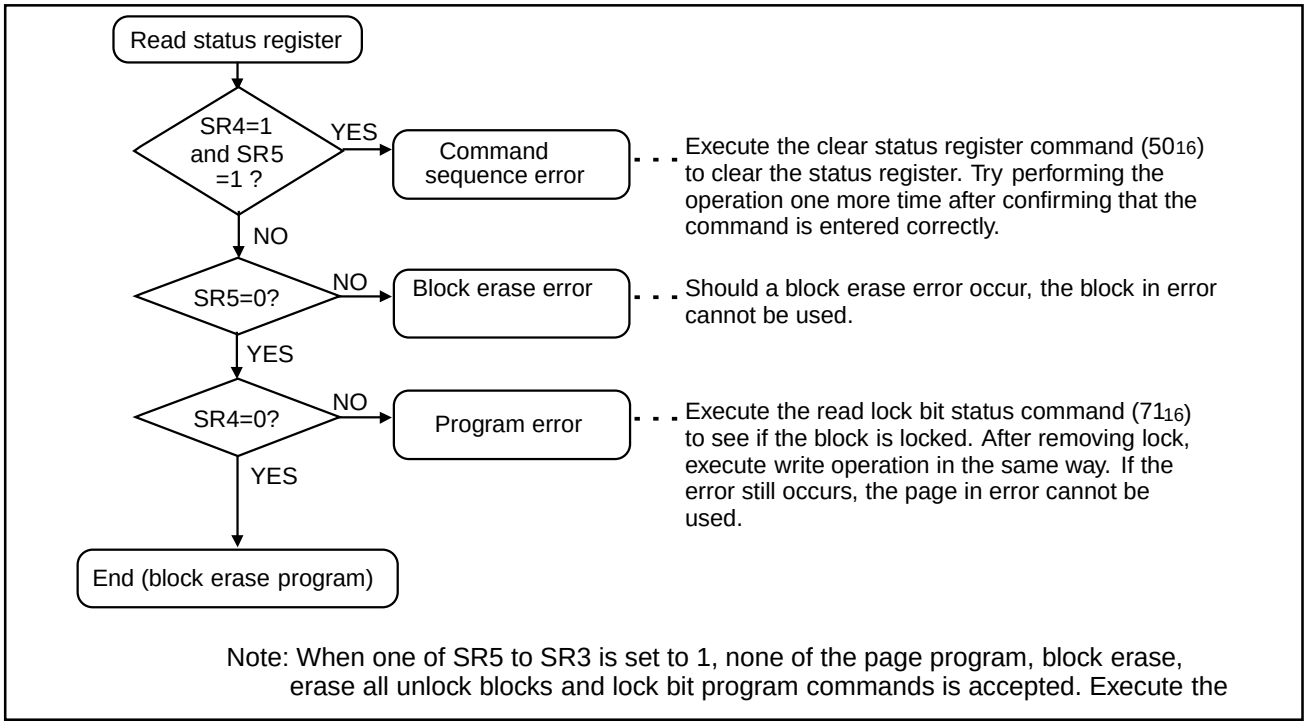
These flags indicate the result of ID checks. Some commands cannot be accepted without an ID check.

### Data Reception Time Out (SR9)

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

### Full Status Check

Results from executed erase and program operations can be known by running a full status check. Figure 1.182 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

**Fig. 1.182. Full status check flowchart and remedial procedure for errors**

## Example Circuit Application for the Standard Serial I/O Modes 1 and 2

Figure 1.183 shows a circuit application for the standard serial I/O modes 1 and 2. Control pins will vary according to programmer, therefore see the peripheral unit manual for more information.

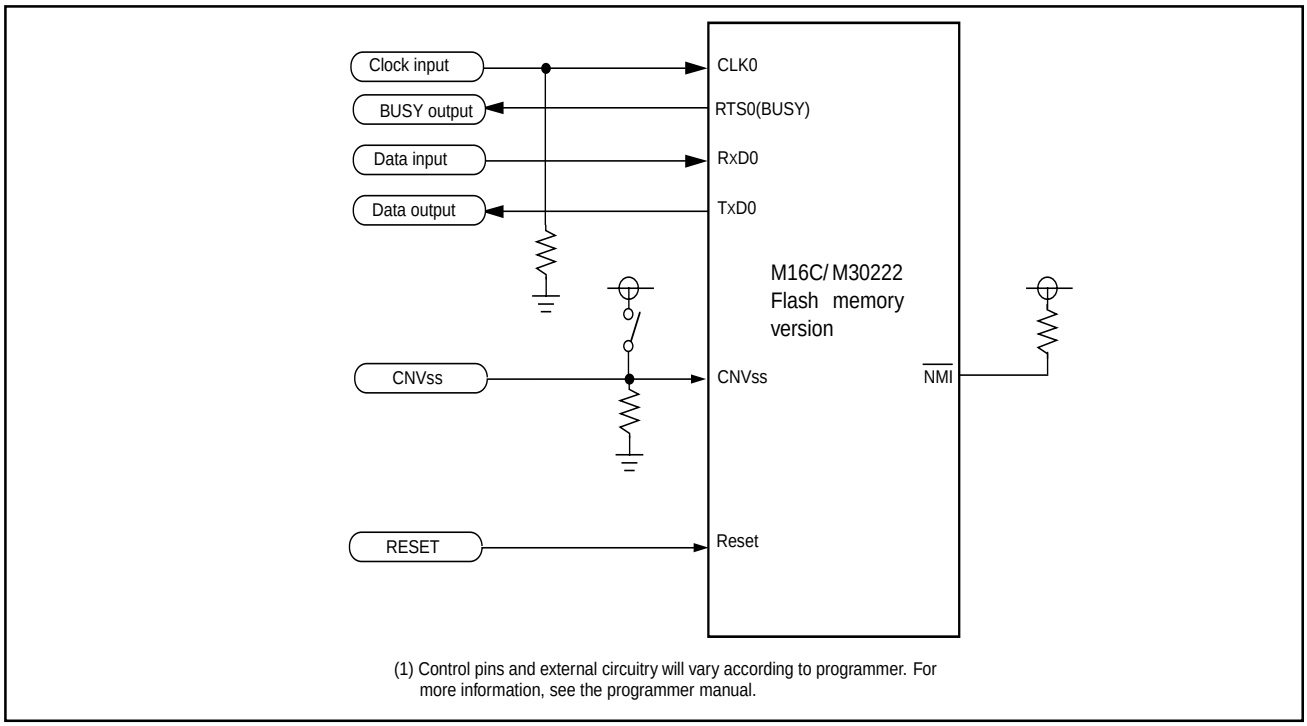



**Fig. 1.183. Example circuit application for serial I/O modes 1 and 2**

## 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 (UART0). Standard serial I/O mode 2 is engaged by releasing the reset with the $P6_1$ ($CLK_0$) pin "L" level.

The $TxD_0$ 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 1.183) are made with a peripheral unit. However, this requires a main clock with a minimum 2 MHz input oscillation frequency. Baud rate can also be changed from 9,600 bps to 19,200, 38,400 or 57,600 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 1.83).

(1) Transmit "$B0_{16}$" from a peripheral unit. If the oscillation frequency input by the main clock is 10 MHz, the MCU with internal flash memory outputs the "$B0_{16}$" check code. If the oscillation frequency is anything other than 10 MHz, the MCU does not output anything.

(2) 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.)

(3) The MCU with internal flash memory outputs the "$B0_{16}$" check code and initial communications end successfully (see Note). 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.

Note: If the peripheral unit cannot receive "$B0_{16}$" successfully, change the oscillation frequency of the main clock.

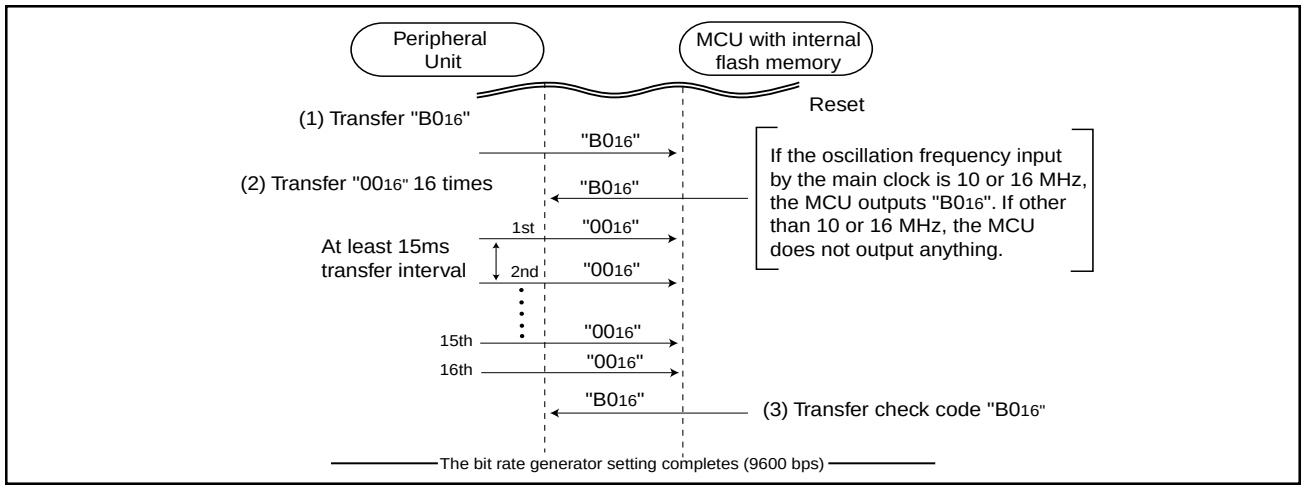



**Figure 1.83. Peripheral unit and initial communication**

## Identifying frequency

When "0016" 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 - 10 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 1.182 gives the operation frequency and the baud rate that can be attained for.

**Table 1.82. Operation frequency and baud rate**

| Operation frequency (MHz) | Baud rate 9600 bps | Baud rate 19200 bps | Baud rate 38400 bps | Baud rate 57600 bps |
|---|---|---|---|---|
| 16 | + | + | + | + |
| 12 | + | + | + | - |
| 11 | + | + | + | - |
| 10 | + | + | - | - |
| 8 | + | + | - | + |
| 7.3728 | + | + | + | + |
| 6 | + | + | + | - |
| 5 | + | + | - | - |
| 4.5 | + | + | - | + |
| 4.194304 | + | + | + | - |
| 4 | + | + | - | - |
| 3.58 | + | + | + | - |
| 3 | + | + | + | - |
| 2 | + | - | - | - |

+ : Communications possible
- : Communications not possible

## Software Commands

Table 1.183 lists software commands. In the standard serial I/O mode 2, erase operations, programs and reading are controlled by transferring software commands via the RxD0 pin. Standard serial I/O mode 2 adds four transmission speed commands - 9,600, 19,200, 38,400 and 57,600 bps - to the software commands of standard serial I/O mode 1. Software commands are explained here below.

**Table 1.183. Software commands (Standard serial I/O mode Page Read Command**

| | Control command | | 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 | SRD 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 | ID check function | $F5_{16}$ | Address (low) | Address (middle) | Address (high) | ID size | ID1 | ID7 | Acceptable |
| 12 | Download function | $FA_{16}$ | Size (low) | Size (high) | Check sum | Data input | As required | | 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 CRC data | $FD_{16}$ | CRC data (Low) | CRC data (High) | | | | | Not acceptable |
| 16 | Word Read | $FE_{16}$ | Address (low) | Address (middle) | Address (high) | | | | Not acceptable |
| 17 | Word Program | $40_{16}$ | Address (low) | Address (middle) | Address (high) | | | | Not acceptable |
| 18 | Exit | $B9_{16}$ | | | | | | | Acceptable |
| 19 | Baud rate 9600 | $B0_{16}$ | **$B0_{16}$** | | | | | | Acceptable |
| 20 | Baud rate 19200 | $B1_{16}$ | **$B1_{16}$** | | | | | | Acceptable |
| 21 | Baud rate 38400 | $B2_{16}$ | **$B2_{16}$** | | | | | | Acceptable |
| 22 | Baud rate 57600 | $B3_{16}$ | **$B3_{16}$** | | | | | | Acceptable |

Note 1: The shaded areas indicate a transfer from flash MCU to serial programmer. All other data is transferred from programmer to MCU.
Note 2: SRD to Status Register Data. SRD1 refers to Status Register 1 Data.
Note 3: All commands are accepted if the reset vector is blank.

### (1) 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.

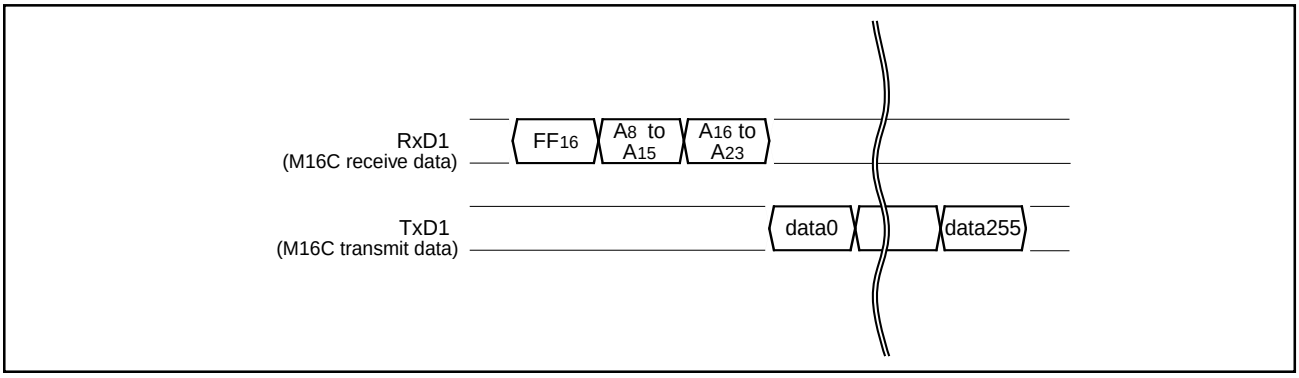



**Figure 1.184. Timing for page read**

### (2) 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 the 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 is known by reading the status register. Each block can be write protected with the lock bit. Additional writing is not allowed with the pages programmed already.

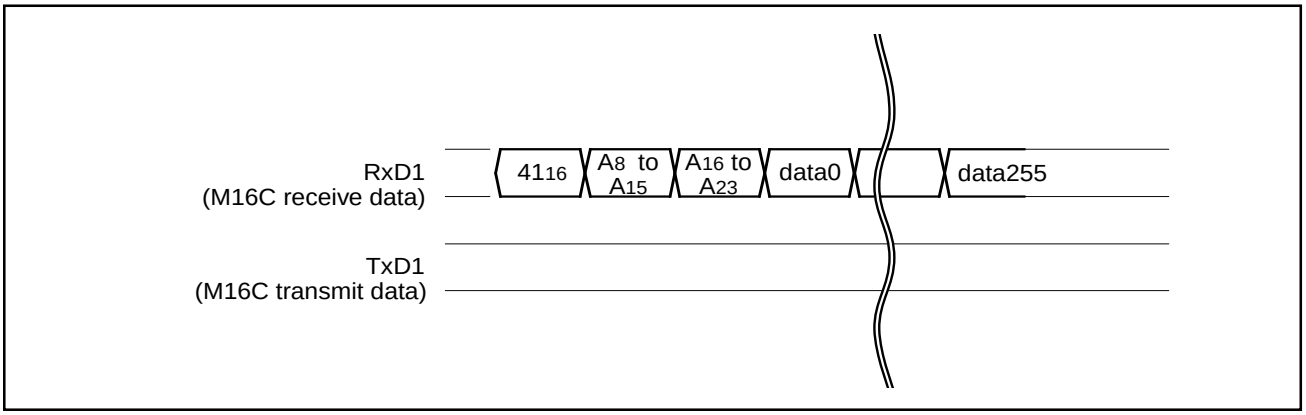



**Figure 1.185. Timing for the page program**

### (3) Block Erase Command

This command erases the data in the specified block. To execute the block erase command :

(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 erase ends, the RTS1 (BUSY) signal changes from the "H" to the "L" level. Afterward, the result of the block erase operation is known by reading the status register. Each block can be erase-protected with the lock bit.

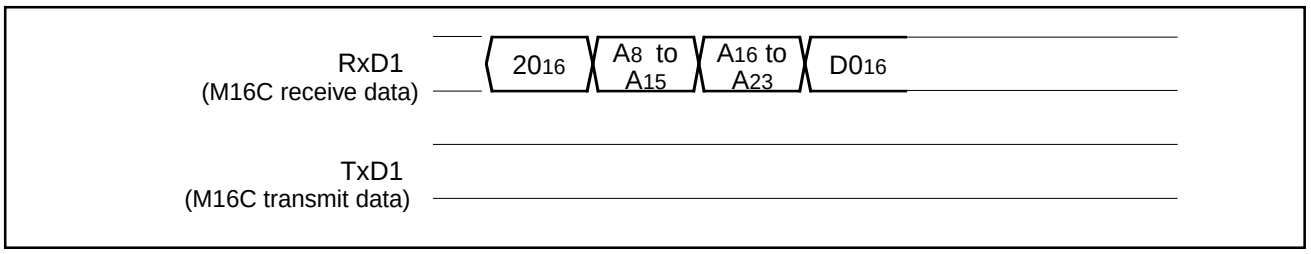



**Figure 1.188. Timing for block erase**

### (4) Erase All Unlocked Blocks Command

This command erases the content of all blocks. Execute the erase all 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 is known be reading the status register. Each block can be erase protected with the lock bit.

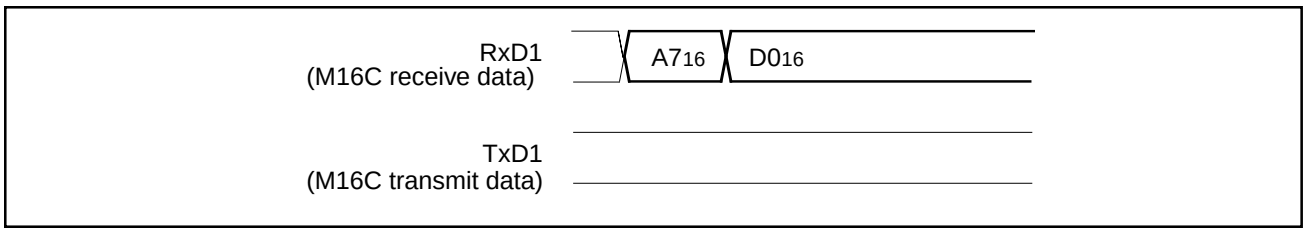



**Figure 1.187. Timing for erasing all unlocked blocks**

### (5) 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.

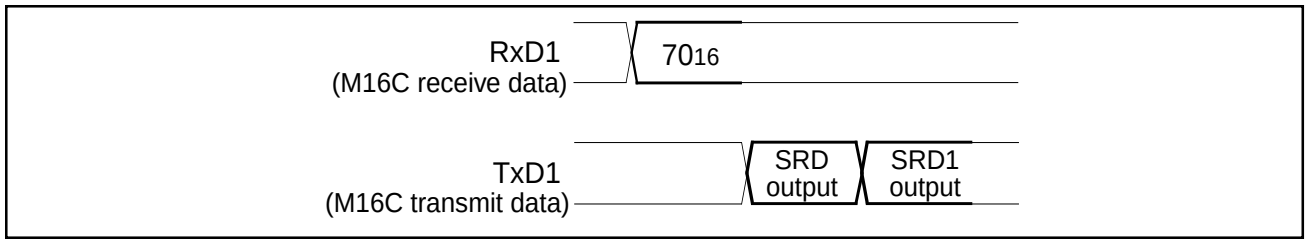



**Figure 1.188. Timing for reading the status register**

### (6) 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, bits SR3-SR5 are cleared.
When the clear status register operation ends, the RTS1 (BUSY) signal changes from the "H" to the "L" level.

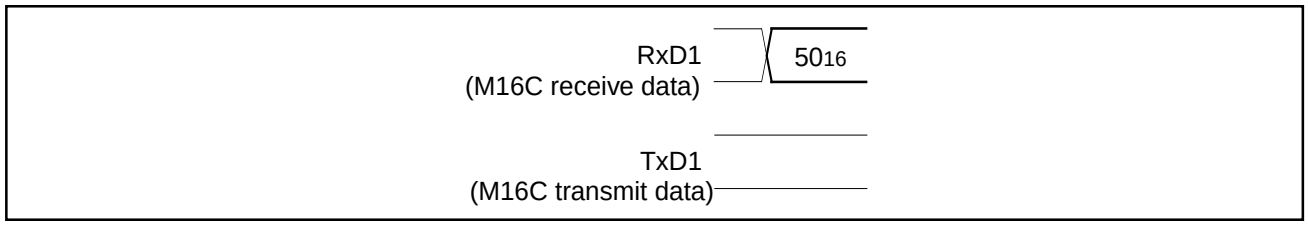



**Figure 1.189. Timing for clearing the status register**

### (7) Read Lock Bit Status Command

This command reads the lock bit status of the specified block. To execute the lock bit status command:
(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. Write the highest address of the specified block for address $A_8$ to $A_{23}$.

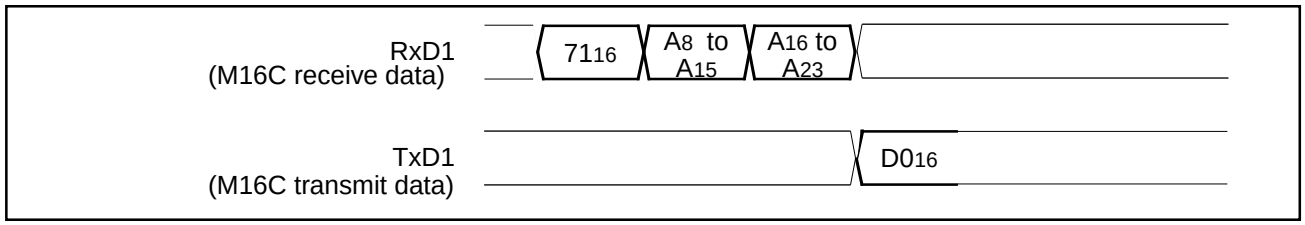



**Figure 1.190. Timing for lock bit status**

### (8) Lock Bit Program Command

This command writes "0" (lock) for the lock bit of the specified block. To execute the lock bit program command :

(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.

(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 address $A_8$ to $A_{23}$.

When writing ends, RTS1 (BUSY) signal changes from the "H" to "L". Lock bit status can be read with the read lock bit status command.

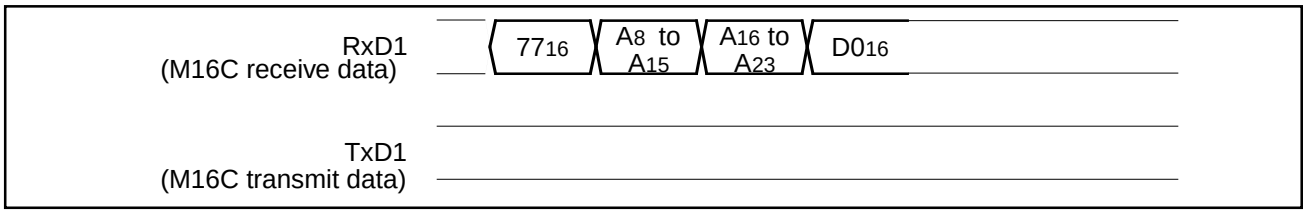



**Figure 1.191. Timing for the lock bit program**

### (9) 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.

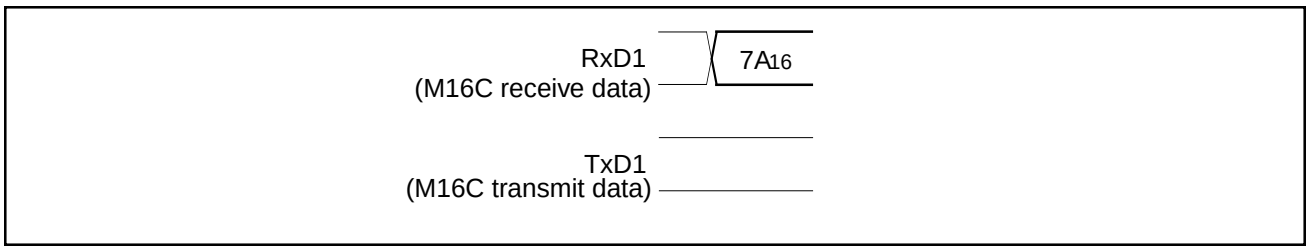



**Figure 1.192. Timing for enabling the lock bit**

### (10) Lock Bit Disable Command

This command disables the lock bit. The command code "7516" 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.

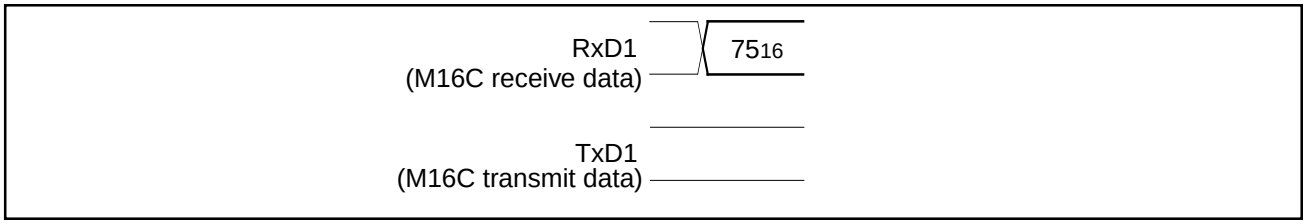



**Figure 1.193. Timing for disabling the lock bit**

### (11) ID Check

This command checks the ID code. To execute the boot ID check command:

(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.

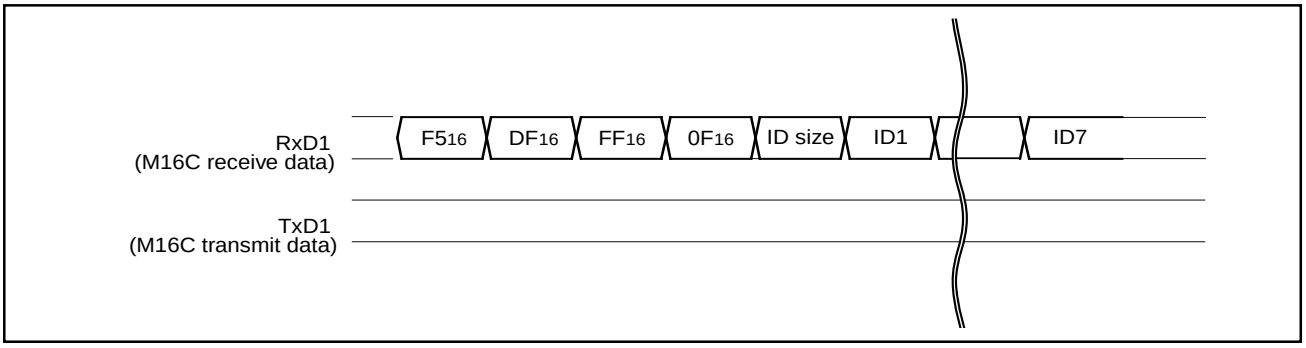



**Figure 1.194. Timing for the ID check**

### 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 0FFFDF$_{16}$, 0FFFE3$_{16}$, 0FFFEB$_{16}$, 0FFFEF$_{16}$, 0FFFF3$_{16}$, 0FFFF7$_{16}$ and 0FFFFB$_{16}$. Write a program into the flash memory, which already has the ID code set for these addresses.

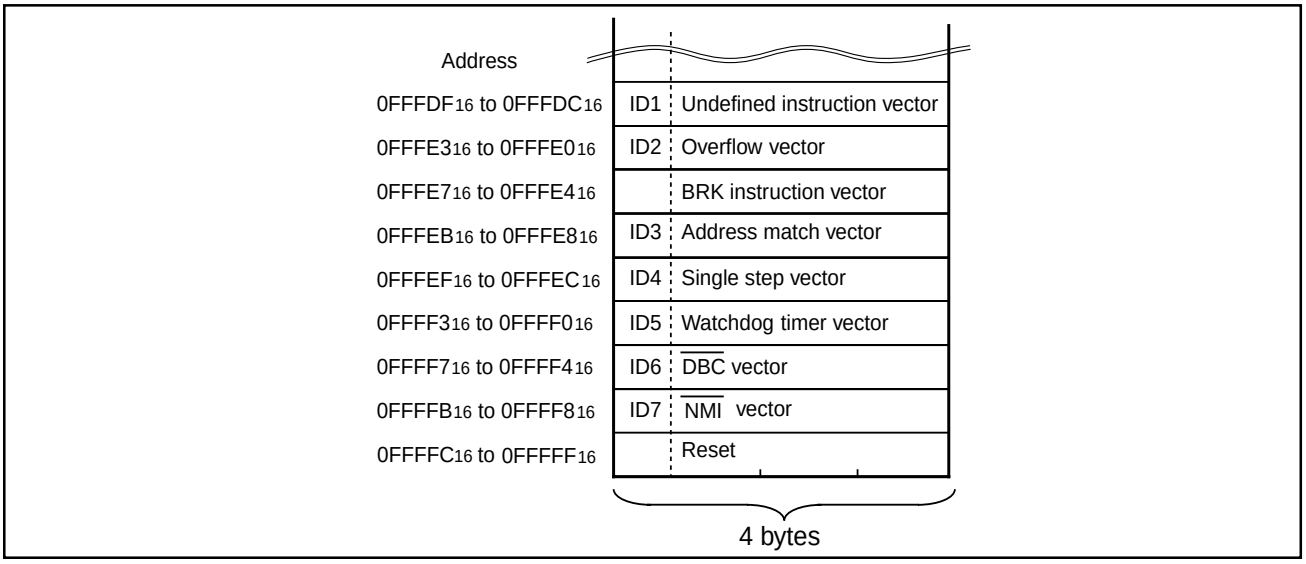



**Figure 1.195. ID code storage addresses**

### (12) Download Command

This command downloads a program to the RAM for execution. To execute the download command:

(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.

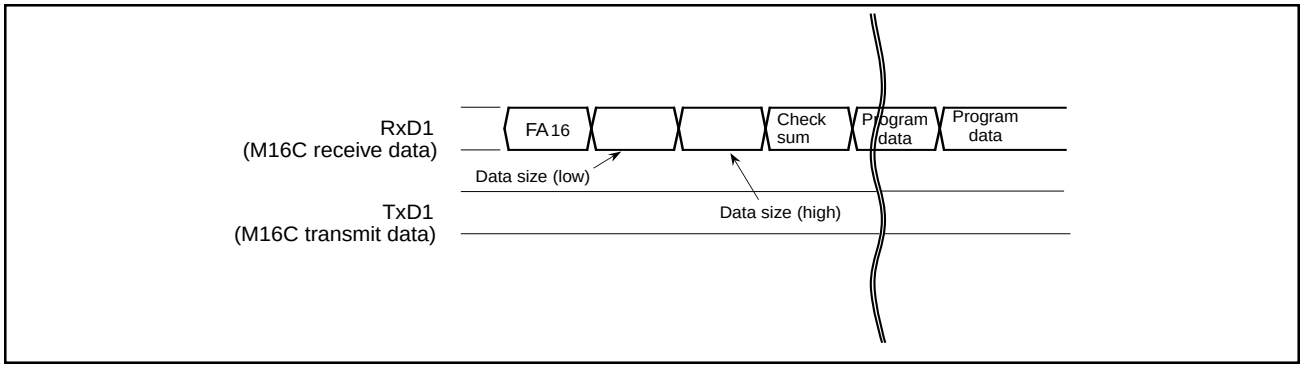



**Figure 1.196. Timing for download**

### (13) Version Information Output Command

This command outputs the version information of the control program stored in the boot area. To execute the version information output command:

(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.

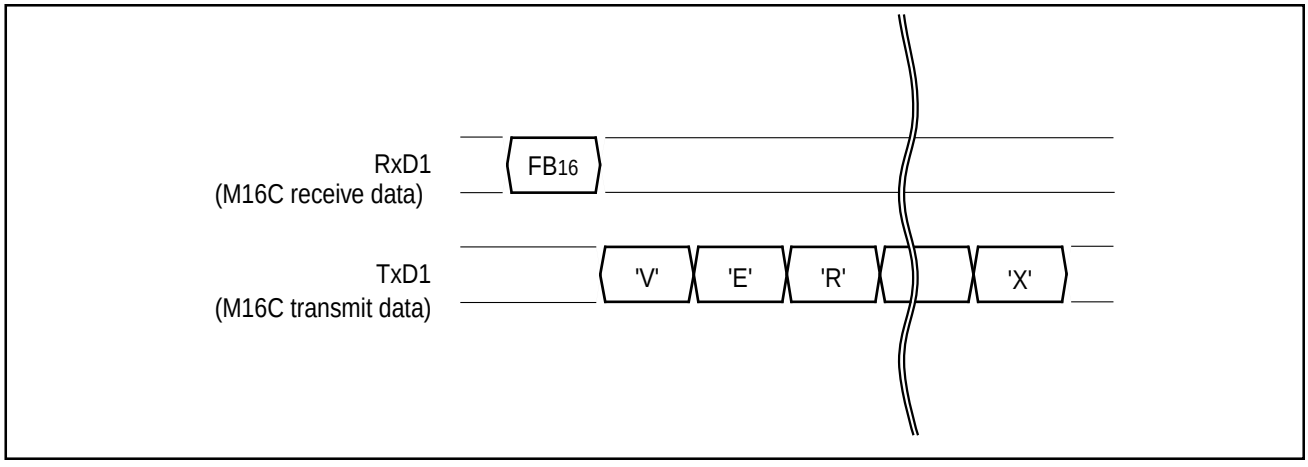



**Figure 1.197. Timing for version information output**

### (14) Boot ROM Area Output Command

This command outputs the control program stored in the boot ROM area in one page blocks (256 bytes). To execute the boot ROM area output command:

(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.

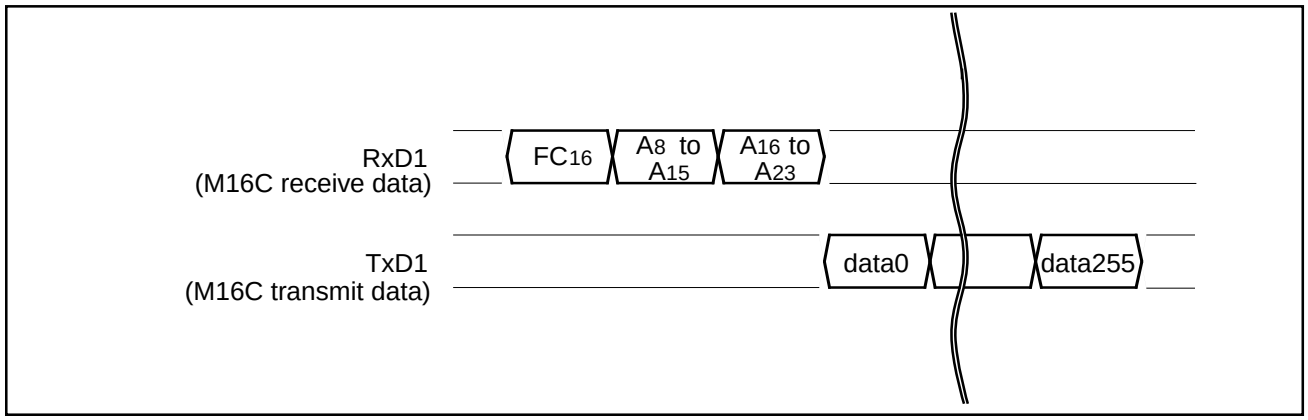



**Figure 1.198. Timing for Boot ROM area output**

### (15) Read CRC Data

This command reads the CRC data that confirms that the write data sent with the page program command was successfully received.

(1) Transfer the "$FD_{16}$" command code with the 1st byte.

(2) The CRC data (low) is received with the 2nd byte and the CRC data (high) with the 3rd byte.

To use this command, first execute the command and then initialize the check data. Next, execute the page program command the required number of times. After the read CRC command is executed again, the CRC data for all the read data that was sent with the page program command is read. The CRC data is the result of CRC operation of write data.

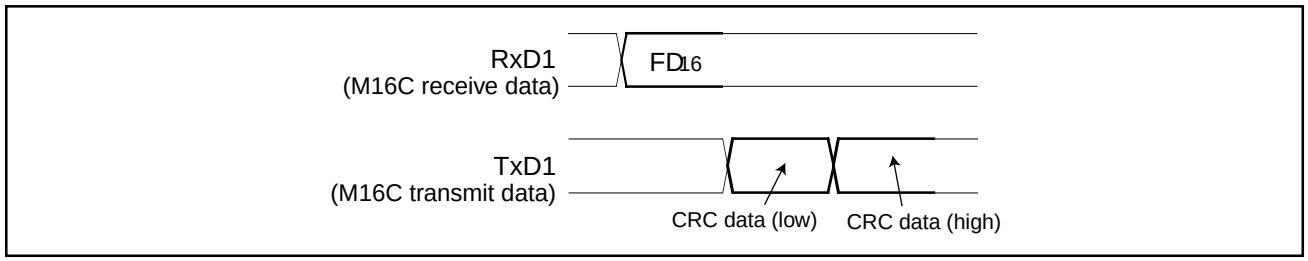



**Figure 1.199. Timing for the read CRC data**

### (16) Word Read Program

This command reads the word from the specific address. To execute the word read command:

(1) Transfer the "FE16" command code with the 1st byte.

(2) Transfer the 3 byte address starting with the A0-A8 with the next 3 bytes.

(3) The MCU transfers the byte at the specific address.

(4) The MCU transfers the byte at the specific address +1.

Note: The specified address may be odd or even (A0=0 or 1) and be any value withing the 1M address space.

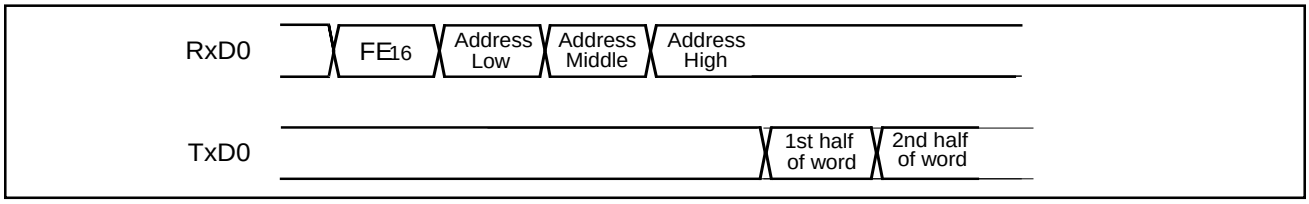



**Figure 1.200. Timing for Word read program**

### (17) Word Program

This command writes the word in the flash memory. To execute the word program command:

(1) Transfer the "4016" with the 1st byte.

(2) Transfer the 3 byte address starting with the A0-A8 with the next 3 bytes.

(3) Transfer the 1st half of the word to be written in the lower address (A0=0).

(4) Transfer the 2nd half of the word to be written in the higher address (A0=1).

Note: The specified address must be even (A0=0).

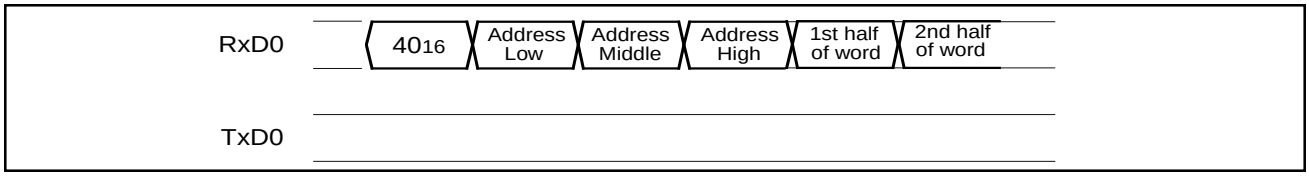



**Figure 1.201. Timing for Word program**

### (18) Exit Command

This command does a software reset by writing a "1" to bit 3 of the Processor Mode register 0. To execute the Exit command:

(1) Transfer "B916" with the 1st byte.

(2) Transfer the confirm command code "D016" in the 2nd byte.

If the CNVss line is low, the MCU will reset in normal mode and begin execution of the user code. If CNVss is high, the MCU will reset back into boot mode and begin execution at 7E00016 again.

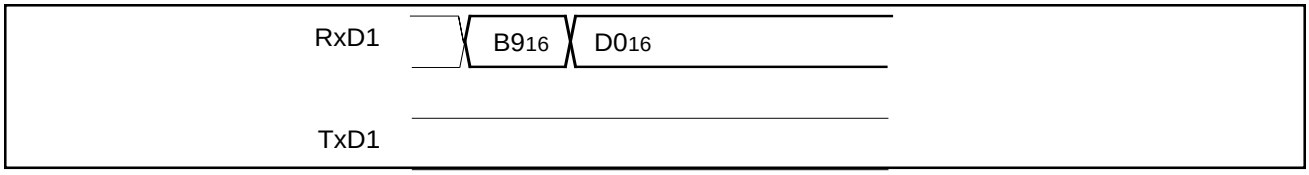



**Figure 1.202. Timing for Exit command**

### (19) Baud Rate 9600

This command changes baud rate to 9,600 bps. To execute:

(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.

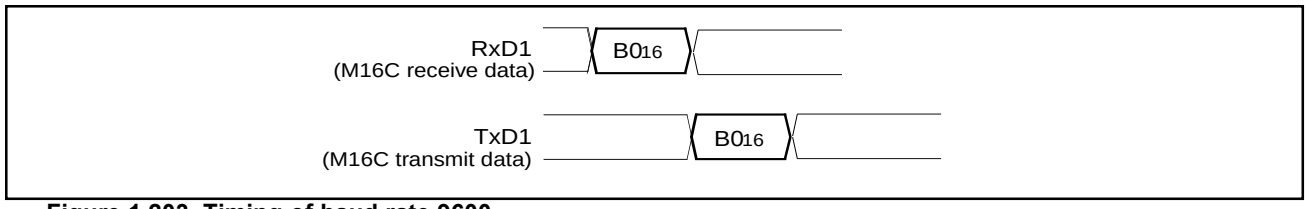



**Figure 1.203. Timing of baud rate 9600**

### (20) 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.

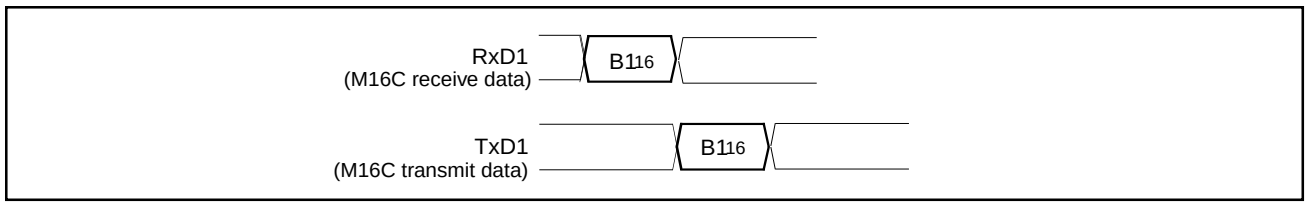



**Figure 1.204. Timing of baud rate 19200**

### (21) 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.

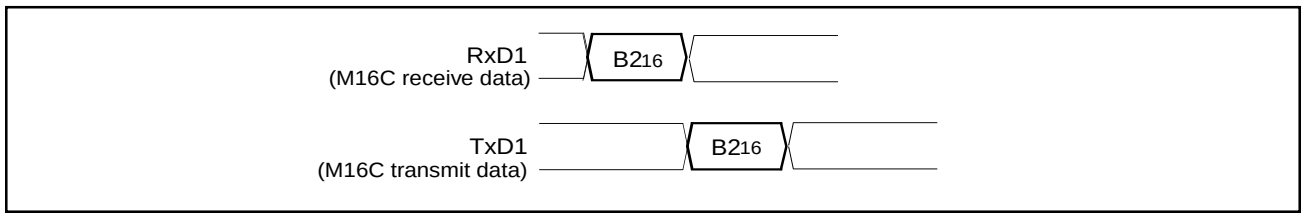



**Figure 1.205. Timing of baud rate 38400**

### (22) 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.

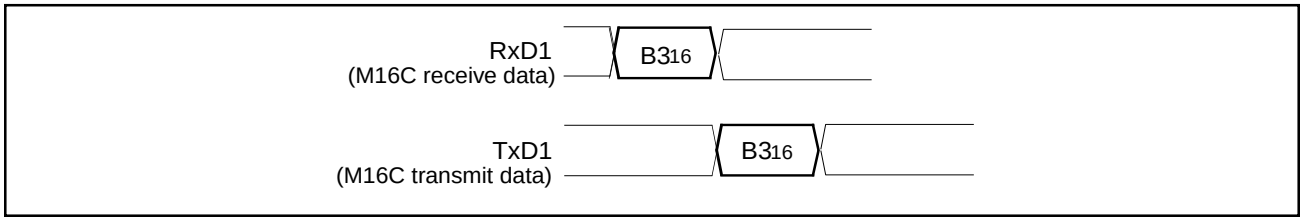



**Figure 1.206. Timing of baud rate 57600**

## Example Circuit Application for the Standard Serial I/O Modes 1 and 2

Figure 1.207 shows a circuit application for the standard serial I/O modes 1 and 2. Control pins will vary according to programmer, therefore see the peripheral unit manual for more information.

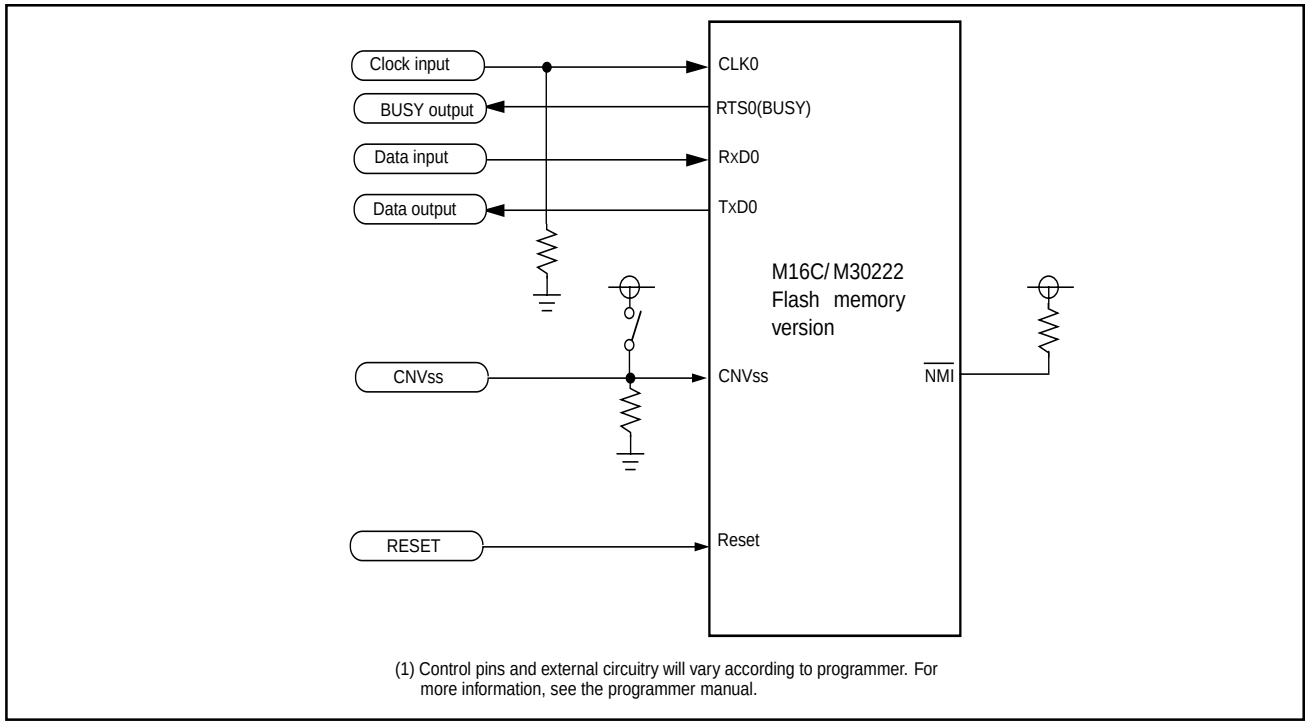



**Fig. 1.207. Example circuit application for serial I/O modes 1 and 2**