

Table of contents

1. DESCRIPTION	1
1.1 Features	1
1.2 Applications	1
1.3 Pin Configuration	3
1.4 Performance Outline	4
1.5 Block Diagram	6
2. OPERATION OF FUNCTIONAL BLOCKS	10
2.1 Memory	10
2.2 Central Processing Unit (CPU)	11
2.3 Reset	13
2.4 Processor Mode	23
2.5 Clock Generating Circuit	39
2.6 Protection	56
2.7 Interrupt	57
2.8 Watchdog Timer	75
2.9 DMAC	77
2.10 Timer	87
2.11 Serial I/O	108
2.12 A-D Converter	158
2.13 CRC Calculation Circuit	175
2.14 Expansion Function	177
2.15 Programmable I/O Ports	237
3. ELECTRICAL CHARACTERISTICS	249
4. FLASH MEMORY VERSION	275
4.1 Flash Memory Performance	275
4.2 Memory Map	277
4.3 Software Commands	289
5. PACKAGE OUTLINE	304
6. USAGE NOTES	305
7. DIFFERENCES BETWEEN M306H5 AND M306H3	323

1.3 Pin Configuration

Figures 1.3.1 shows the pin configuration (top view).

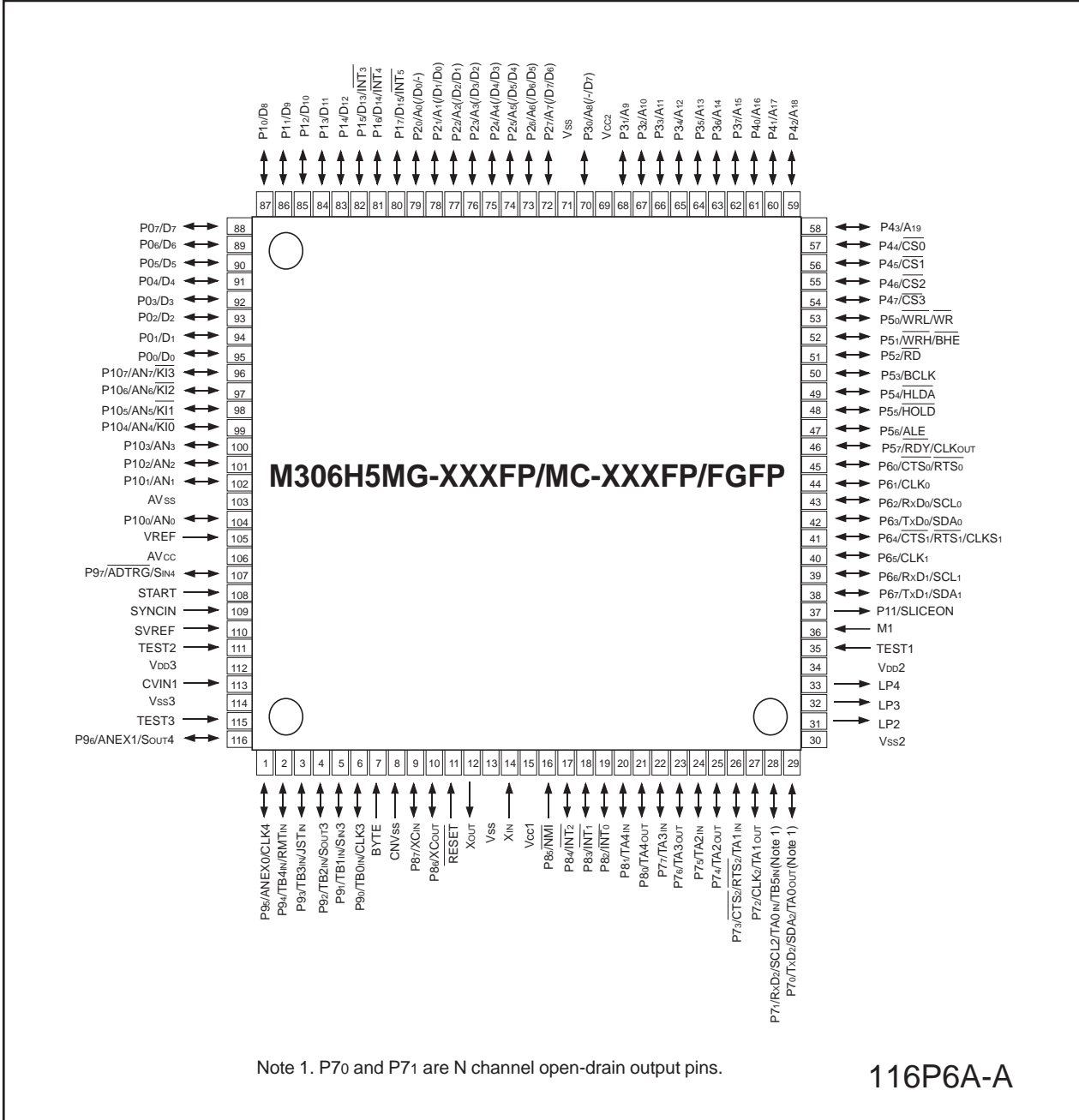


Figure 1.3.1 Pin configuration (top view)

1.4 Performance Outline

Table 1.4.1 is a performance outline.

Table 1.4.1 Performance outline

Item		Performance
Number of basic instructions		91 instructions
Shortest instruction execution time		62.5 ns ($f(X_{IN})=16\text{MHz}$, $V_{CC}=4.5\text{V}$ to 5.5V)
Memory capacity	ROM	Refer to the Product table (Fig. 1.4.2)
	RAM	Refer to the Product table (Fig. 1.4.2)
I/O port	P0 to P5, P86 to P87, P9 to P10	8 bits x 8, 2 bits x 1 : V_{CC2} system
	P6 to P7, P80 to P84	8 bits x 2, 5 bits x 1 : V_{CC1} system
Input port	P85	1 bit x 1 (NMI pin V_{CC2} level judgment) : V_{CC2} system
Output	P11	1 bit x 1
Multifunction timer		16 bits x 5 channels (TA0, TA1, TA2, TA3, TA4) 16 bits x 6 channels (TB0, TB1, TB2, TB3, TB4, TB5)
Serial I/O		3 channels (UART0, UART1, UART2) UART, clock synchronous, I ² C bus (option, Note 1), or IEBus (option, Note 2) 2 channels (SI/O3, SI/O4) Clock synchronous
A-D converter		8 bits x (8 + 2) channels
DMAC		2 channels (trigger: 24 sources)
CRC calculation circuit		CRC-CCITT
Watchdog timer		15 bits x 1 (with prescaler)
Interrupt		25 internal and 8 external sources, 4 software sources, 7 levels
Clock generation circuit		2 circuits <ul style="list-style-type: none"> • Main clock • Sub-clock (These circuits contain a built-in feedback resistor for external ceramic or crystal oscillator)
Power supply voltage		$V_{CC1}=3.00\text{ V}$ to V_{CC2} , $V_{CC2}=4.5\text{ V}$ to 5.5 V (at $f(X_{IN})=16\text{MHz}$) $V_{CC1}=3.00\text{ V}$ to V_{CC2} , $V_{CC2}=4.00\text{ V}$ to 5.5 V (at $f(X_{IN})=16\text{MHz}$) (Note 3) $V_{CC1}=2.90\text{ V}$ to V_{CC2} , $V_{CC2}=2.90\text{ V}$ to 5.5 V (at $f(X_{IN})=16\text{MHz}$, at divide-by-8 or 16) (Note 3) $V_{CC1}=2.0\text{ V}$ to V_{CC2} , $V_{CC2}=2.0\text{ V}$ to 5.5 V (at $f(X_{CIN})=32\text{kHz}$, only low-power consumption mode) (Note 3) (Note 4)
Flash memory	Program/erase voltage	$5.0\text{ V} \pm 0.25\text{ V}$
	Number of program/erase	100 times
Device configuration		CMOS high performance silicon gate
Package		116-pin plastic mold QFP
Data acquisition	Slice RAM	864 bytes (48 X 18 X 8-bit)
	Data acquisition circuit	Corresponds to PDC, VPS, EPG-J, XDS and WSS

Notes:

1. I²C bus is a registered trademark of Koninklijke Philips Electronics N.V.
If you desire this option, please so specify.
2. IEBus is a registered trademark of NEC Electronics Corporation.
3. If the V_{CC2} supply voltage is less than 4.50 V, the A-D converter, data slicer cannot be used.
4. If the V_{CC2} supply voltage is less than 2.60 V, be aware that only the CPU, RAM, clock timer, interrupt, and Input/Output ports can be used. Other control circuits (e.g., timers A and B, serial I/O, UART) cannot be used.

Figure 1.4.2 Product table

Type No.	ROM capacity	RAM capacity	Package type	Remarks
M306H5MG-XXXFP	256K bytes	8K bytes	116P6A-A	Mask ROM version
M306H5MC-XXXFP	128K bytes	5K bytes		
M306H5FGFP	256K bytes	8K bytes		Flash Memory version

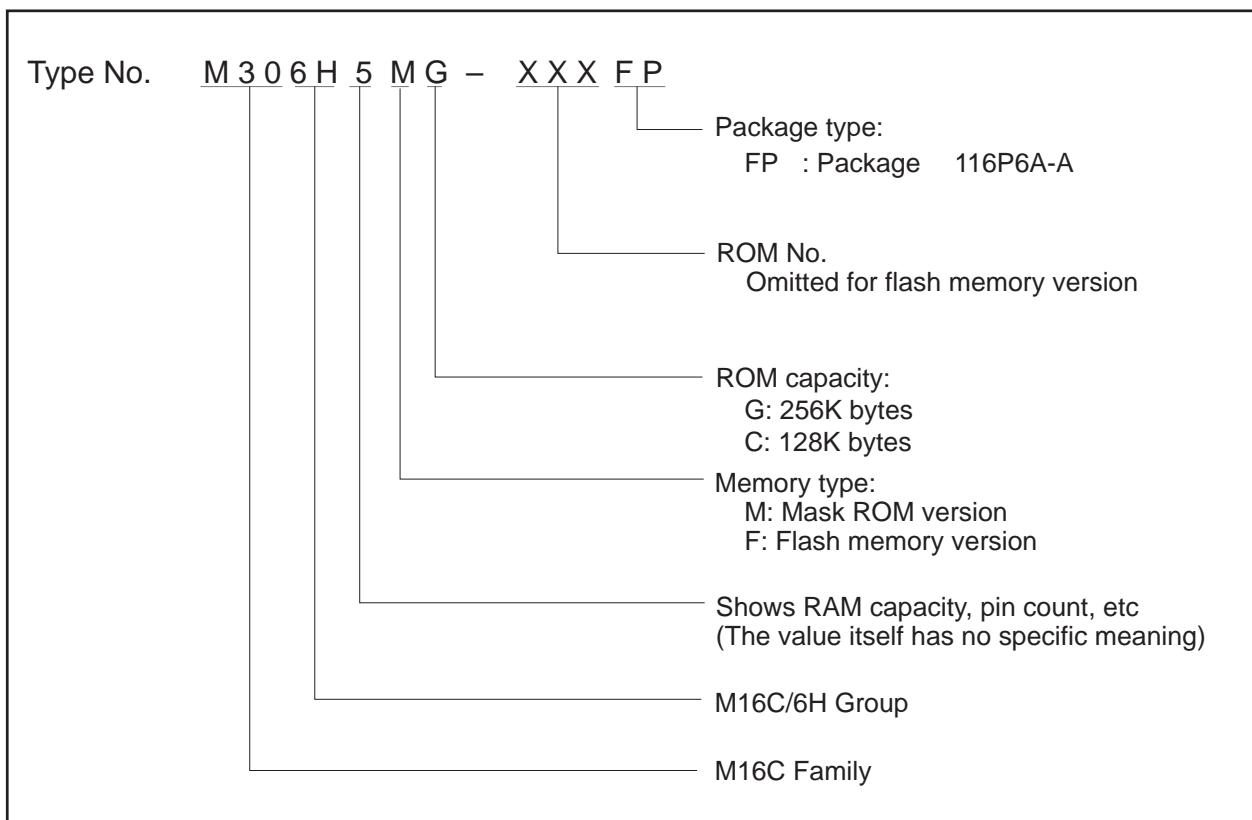


Figure 1.4.1 Type No, Memory Size, and Package

1.5 Block Diagram

Figure 1.5.1 is a block diagram.

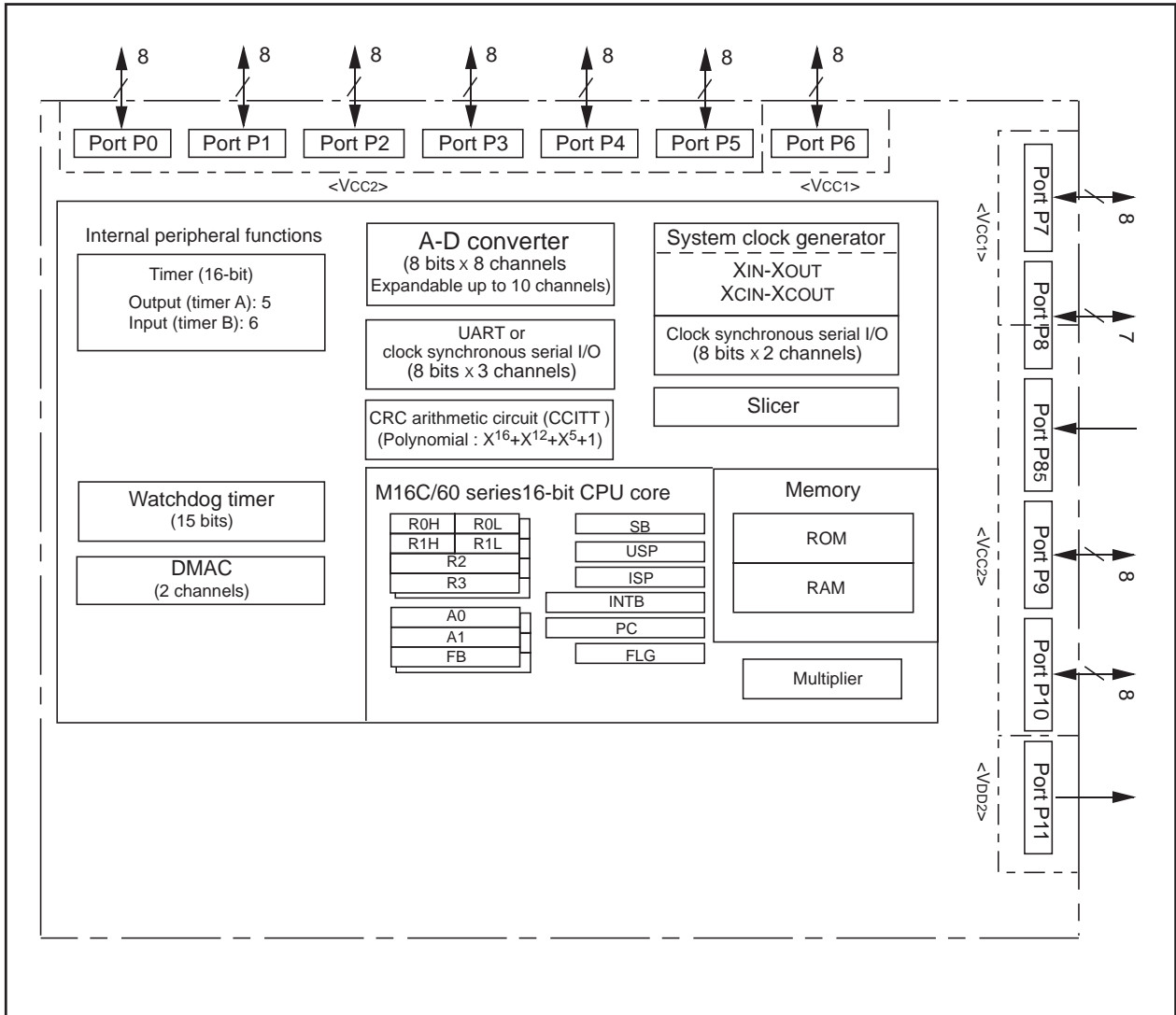


Figure 1.5.1 Block diagram

Table 1.5.1 Pin Description

Pin name	Signal name	I/O type	Power supply	Function
Vcc1, Vcc2, Vss	Power supply input			Apply 2.00 V to 5.5 V to the Vcc1 and Vcc2 pins. Apply 0 V to the Vss pin. Input condition of Vcc1 and Vcc2 are $V_{cc1} \leq V_{cc2}$. (Note 1)
CNVss	CNVss	Input	VCC2	This pin switches between processor modes. Connect this pin to Vss pin when after a reset you want to start operation in single-chip mode (memory expansion mode) or the VCC pin when starting operation in microprocessor mode.
RESET	Reset input	Input	VCC2	"L" on this input resets the microcomputer.
XIN XOUT	Clock input Clock output	Input Output	VCC2	These pins are provided for the main clock generating circuit input/output. Connect a ceramic resonator or crystal between the XIN and the XOUT pins. To use an externally derived clock, input it to the XIN pin and leave the XOUT pin open.
BYTE	External data bus width select input	Input	VCC2	This pin selects the width of an external data bus. A 16-bit width is selected when this input is "L"; an 8-bit width is selected when this input is "H". This input must be fixed to either "H" or "L". Connect this pin to the Vss when single-chip mode.
AVCC	Analog power supply input			This pin is a power supply input for the A-D converter. Connect this pin to VCC.
AVSS	Analog power supply input			This pin is a power supply input for the 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.
P00 to P07	I/O port P0	Input/output	VCC2	This is an 8-bit CMOS I/O port. This port has an input/output select direction register, allowing each pin in that port to be directed for input or output individually. If any port is set for input, selection can be made for it in a program whether or not to have a pull-up resistor in 4 bit units. This selection is unavailable in memory extension and microprocessor modes.
D0 to D7		Input/output		When set as a separate bus, these pins input and output data (D0 to D7).
P10 to P17	I/O port P1	Input/output	VCC2	This is an 8-bit I/O port equivalent to P0. Pins in this port also function as INT interrupt input pins as selected by software.
D8 to D15		Input/output		When set as a separate bus, these pins input and output data (D8 to D15).
P20 to P27	I/O port P2	Input/output	VCC2	This is an 8-bit I/O port equivalent to P0.
A0 to A7		Output		These pins output 8 low-order address bits (A0 to A7).
A0/D0 to A7/D7		Input/output		If the external bus is set as an 8-bit wide multiplexed bus, these pins input and output data (D0 to D7) and output 8 low-order address bits (A0 to A7) separated in time by multiplexing.
A0 A1/D0 to A7/D6		Output Input/output		If the external bus is set as a 16-bit wide multiplexed bus, these pins input and output data (D0 to D6) and output address (A1 to A7) separated in time by multiplexing. They also output address (A0).
P30 to P37	I/O port P3	Input/output	VCC2	This is an 8-bit I/O port equivalent to P0.
A8 to A15		Output		These pins output 8 middle-order address bits (A8 to A15).
A8/D7, A9 to A15		Input/output Output		If the external bus is set as a 16-bit wide multiplexed bus, these pins input and output data (D7) and output address (A8) separated in time by multiplexing. They also output address (A9 to A15).
P40 to P47	I/O port P4	Input/output	VCC2	This is an 8-bit I/O port equivalent to P0.
CS0 to CS3, A16 to A19		Output Output		These pins output CS0 to CS3 signals and A16 to A19. CS0 to CS3 are chip select signals used to specify an access space. A16 to A19 are 4 high-order address bits.

Note 1: In this datasheet, hereafter, VCC refers to VCC2 unless otherwise noted.

Table 1.5.2 Pin Description

Pin name	Signal name	I/O type	Power supply	Function
P50 to P57	I/O port P5	Input/output	VCC2	<p>This is an 8-bit I/O port equivalent to P0. In single-chip mode, P57 in this port outputs a divide-by-8 or divide-by-32 clock of X_{IN} or a clock of the same frequency as X_{CIN} as selected by program.</p> <p>Output \overline{WRL}, \overline{WRH} (\overline{WR} and \overline{BHE}), \overline{RD}, \overline{BCLK}, \overline{HLDA}, and \overline{ALE} signals. \overline{WRL} and \overline{WRH}, and \overline{BHE} and \overline{WR} can be switched using program.</p> <p>■ \overline{WRL}, \overline{WRH}, and \overline{RD} selected With a 16-bit external data bus, data is written to even addresses when the \overline{WRL} signal is "L" and to the odd addresses when the \overline{WRH} signal is "L". Data is read when \overline{RD} is "L".</p> <p>■ \overline{WR}, \overline{BHE}, and \overline{RD} selected Data is written when \overline{WR} is "L". Data is read when \overline{RD} is "L". Odd addresses are accessed when \overline{BHE} is "L". Use this mode when using an 8-bit external data bus.</p> <p>While the input level at the \overline{HOLD} pin is "L", the microcomputer is placed in the hold state. While in the hold state, \overline{HLDA} outputs a "L" level. \overline{ALE} is used to latch the address. While the input level of the \overline{RDY} pin is "L", the microcomputer is in the wait state.</p>
\overline{WRL} / \overline{WR} , \overline{WRH} / \overline{BHE} , \overline{RD} , \overline{BCLK} , \overline{HLDA} , \overline{HOLD} , \overline{ALE} , \overline{RDY}	Output Output Output Output Input Output Input			
P60 to P67	I/O port P6	Input/output	VCC1	This is an 8-bit I/O port equivalent to P0. Pins in this port also function as UART0 and UART1 I/O pins as selected by program.
P70 to P77	I/O port P7	Input/output	VCC1	This is an 8-bit I/O port equivalent to P0 (P70 and P71 are N channel open-drain output). This port can function as input/output pins for timers A0 to A3 when so selected in a program. Furthermore, P70 to P73, and P71 can also function as input/output pins for UART2, an input pin for timer B5, respectively.
P80 to P84, P86, P87, P85	I/O port P80 to P84 I/O port P86 I/O port P87 I/O port P85	Input/output Input/output Input/output Input	VCC1 (P80 to P84) VCC2 (P85 to P87)	<p>P80 to P84, P86, and P87 are I/O ports with the same functions as P0. When so selected in a program, P80 to P81 and P82 to P84 can function as input/output pins for timer A4 and \overline{INT} interrupt input pins, respectively. P86 and P87, when so selected in a program, both can function as input/output pins for the subclock oscillator circuit. In that case, connect a crystal resonator between P86 (\overline{XCOUT} pin) and P87 (\overline{XCIN} pin).</p> <p>P85 is an input-only port shared with \overline{NMI}. An \overline{NMI} interrupt is generated when input on this pin changes state from high to low. The \overline{NMI} function cannot be disabled in a program. A pull-up cannot be set for this pin.</p>
P90 to P97	I/O port P9	Input/output	VCC2	This is an 8-bit I/O port equivalent to P0. Pins in this port also function as SI/O3, 4 I/O pins, Timer B0 to B4 input pins, A-D converter extended input pins, A-D trigger input pins, or remote control input pins as selected by program.
P100 to P107	I/O port P10	Input/output	VCC2	This is an 8-bit I/O port equivalent to P0. Pins in this port also function as A-D converter input pins as selected by program. Furthermore, P104 to P107 also function as input pins for the key input interrupt function.
P11	Output port P11	Output	VDD2	This is a 1-bit output-only port. Pins in this port also function as SLICEON output pins as selected by program.

Table 1.5.3 Pin Description

Pin name	Signal name	I/O type	Power supply	Function
VDD2, VSS2	Power supply input			Analog power supply pin. Apply the same potential as VCC2 to the VDD2 pin. Apply 0 V to the VSS2 pin.
VDD3, VSS3	Power supply input			Analog power supply pin. Apply the same potential as VCC2 to the VDD3 pin. Apply 0 V to the VSS3 pin.
SVREF	Synchronous slice level input	Input	VCC2	When slice the vertical synchronous signal, input slice level.
CVIN1	Composite video signal input 1	Input	VCC2	This pin inputs the external composite video signal. Data-acquisition slices this signal internally by setting.
SYNCIN	Composite video signal input 2	Input	VCC2	This pin inputs the external composite video signal. Sync.-separate circuit divides this signal internally.
START	Oscillation selection input	Input	VCC2	This pin selects the oscillation circuit. XIN-XOUT circuit is selected when this pin is "H"; XCIN-XCOUT circuit is selected when this pin is "L".
LP2	Filter output 1	Output	VDD2	This is a filter output pin 1 (for fsc).
LP3	Filter output 2	Output	VDD2	This is a filter output pin 2 (for VPS).
LP4	Filter output 3	Output	VDD2	This is a filter output pin 3 (for PDC).
TEST3	VCC1 Power supply input select	Input	VCC2	Normally, please input "L" level. When VCC1 power supply is off, please input "H" level.
M1	Mode selection input (M1 input)	Input	VDD2	In the flash memory version, connect this pin to the VDD2 when use microprocessor mode or memory expansion mode. Connect it to the VSS when use standard serial I/O mode (single-chip mode). In the mask ROM version, connect this pin to the VSS or the VDD2.
TEST1	Test input	Input		This is a test pin. Connect a capacitor.
TEST2	Test input	Input		This is a test pin. Connect this pin to the VSS.

2. OPERATION OF FUNCTIONAL BLOCKS

2.1 Memory

Figure 2.1.1 is a memory map of M306H5/MG-XXXFP/MC-XXXFP/FCFP. The address space extends the 1M bytes from address 00000₁₆ to FFFFF₁₆.

The internal ROM is allocated in a lower address direction beginning with address FFFFF₁₆. An internal ROM of M306H5MC-XXXFP, for instance, is allocated to the addresses from E0000₁₆ to FFFFF₁₆.

The fixed interrupt vector table is allocated to the addresses from FFFDC₁₆ to FFFFF₁₆. Therefore, store the start address of each interrupt routine here.

The internal RAM is allocated in an upper address direction beginning with address 00400₁₆. An internal RAM of M306H5MC-XXXFP, for instance, is allocated to the addresses from 00400₁₆ to 017FF₁₆/023FF₁₆. In addition to storing data, the internal RAM also stores the stack used when calling subroutines and when interrupts are generated.

The SFR is allocated to the addresses from 00000₁₆ to 003FF₁₆. Peripheral function control registers are located here. Of the SFR, any area which has no functions allocated is reserved for future use and cannot be used by users.

The special page vector table is allocated to the addresses from FFE00₁₆ to FFFDB₁₆. This vector is used by the JMPS or JSRS instruction. For details, refer to the “M16C/60 and M16C/20 Series Software Manual.”

In memory expansion and microprocessor modes, some areas are reserved for future use and cannot be used by users.

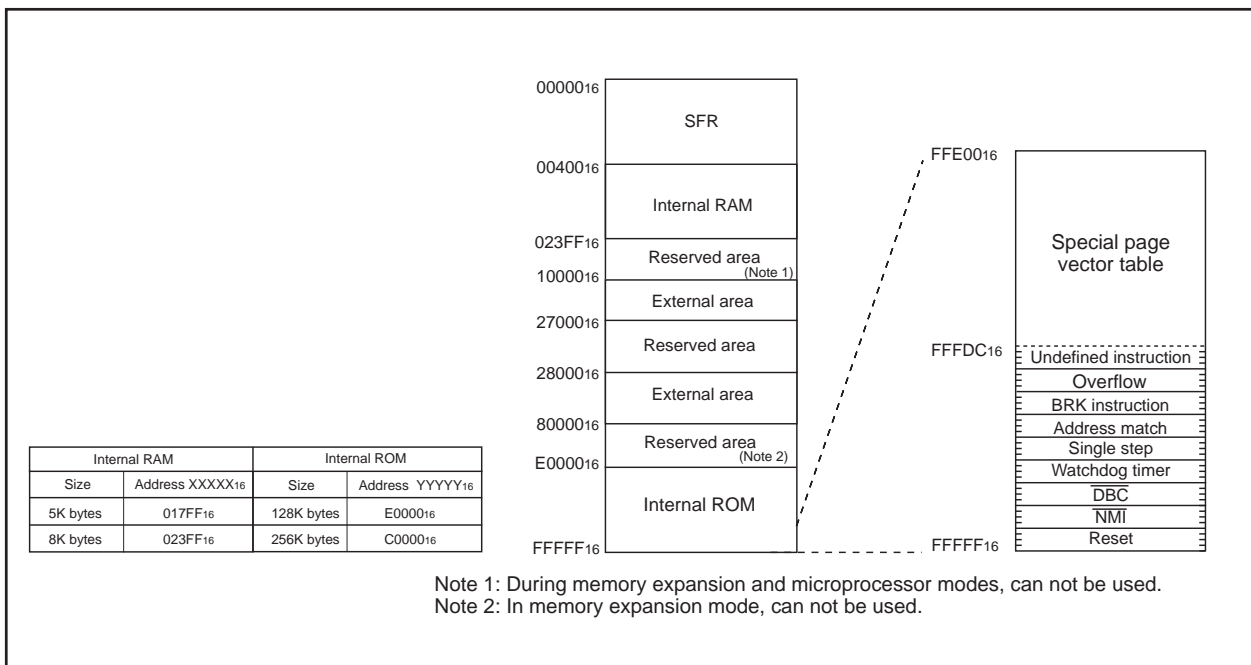


Figure 2.1.1. Memory Map

2.2 Central Processing Unit (CPU)

Figure 2.2.1 shows the CPU registers. The CPU has 13 registers. Of these, R0, R1, R2, R3, A0, A1 and FB comprise a register bank. There are two register banks.

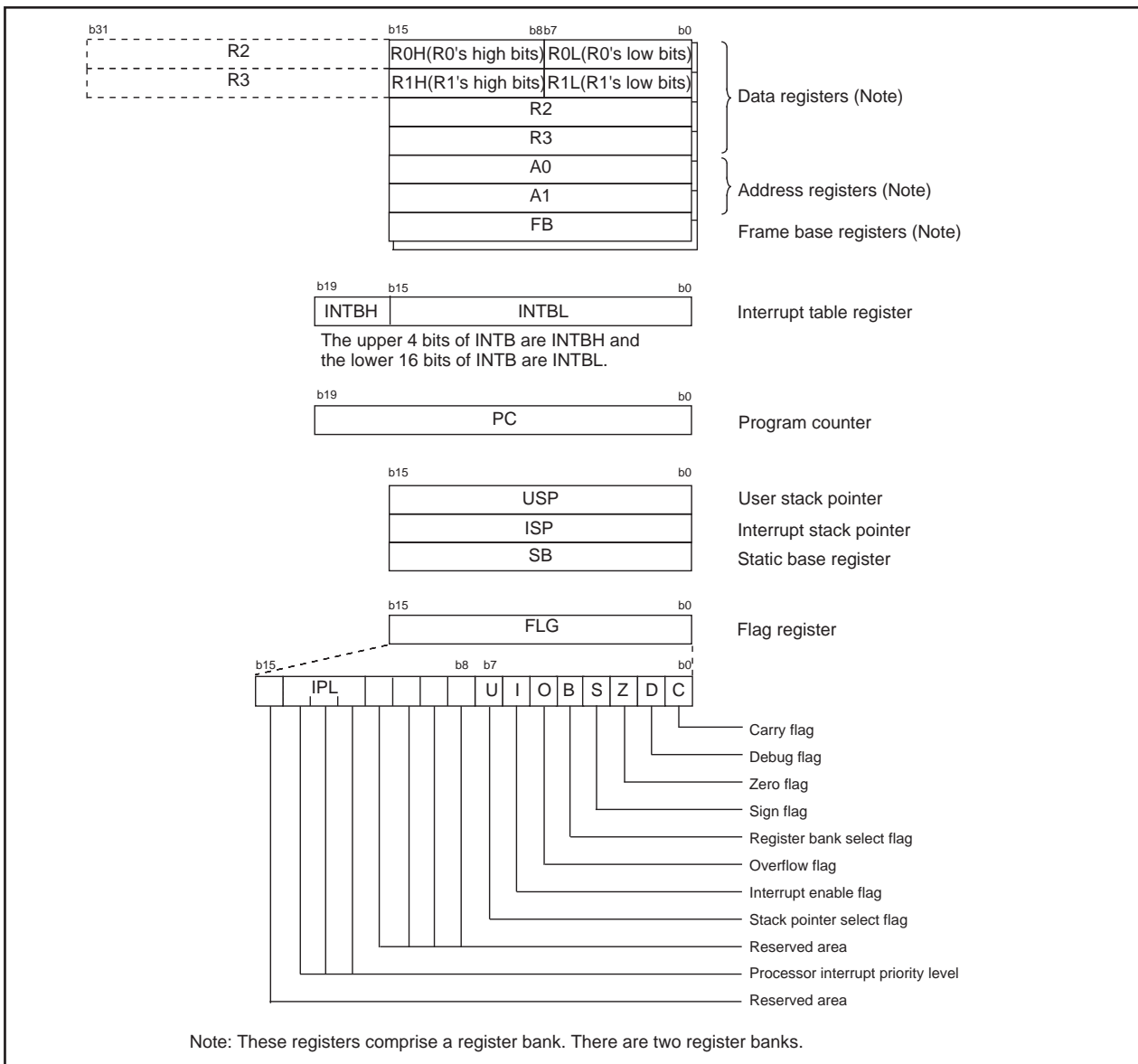


Figure 2.2.1. Central Processing Unit Register

(1) Data Registers (R0, R1, R2 and R3)

The R0 register consists of 16 bits, and is used mainly for transfers and arithmetic/logic operations. R1 to R3 are the same as R0.

The R0 register can be separated between high (R0H) and low (R0L) for use as two 8-bit data registers. R1H and R1L are the same as R0H and R0L. Conversely, R2 and R0 can be combined for use as a 32-bit data register (R2R0). R3R1 is the same as R2R0.

(2) Address Registers (A0 and A1)

The register A0 consists of 16 bits, and is used for address register indirect addressing and address register relative addressing. They also are used for transfers and logic/logic operations. A1 is the same as A0.

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

(3) Frame Base Register (FB)

FB is configured with 16 bits, and is used for FB relative addressing.

(4) Interrupt Table Register (INTB)

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

(5) Program Counter (PC)

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

(6) User Stack Pointer (USP) and Interrupt Stack Pointer (ISP)

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

Your desired type of stack pointer (USP or ISP) can be selected by the U flag of FLG.

(7) Static Base Register (SB)

SB is configured with 16 bits, and is used for SB relative addressing.

(8) Flag Register (FLG)

FLG consists of 11 bits, indicating the CPU status.

• Carry Flag (C Flag)

This flag retains a carry, borrow, or shift-out bit that has occurred in the arithmetic/logic unit.

• Debug Flag (D Flag)

The D flag is used exclusively for debugging purpose. During normal use, it must be set to "0".

• Zero Flag (Z Flag)

This flag is set to "1" when an arithmetic operation resulted in 0; otherwise, it is "0".

• Sign Flag (S Flag)

This flag is set to "1" when an arithmetic operation resulted in a negative value; otherwise, it is "0".

• Register Bank Select Flag (B Flag)

Register bank 0 is selected when this flag is "0"; register bank 1 is selected when this flag is "1".

• Overflow Flag (O Flag)

This flag is set to "1" when the operation resulted in an overflow; otherwise, it is "0".

• Interrupt Enable Flag (I Flag)

This flag enables a maskable interrupt.

Maskable interrupts are disabled when the I flag is "0", and are enabled when the I flag is "1". The I flag is cleared to "0" when the interrupt request is accepted.

• Stack Pointer Select Flag (U Flag)

ISP is selected when the U flag is "0"; USP is selected when the U flag is "1".

The U flag is cleared to "0" when a hardware interrupt request is accepted or an INT instruction for software interrupt Nos. 0 to 31 is executed.

• Processor Interrupt Priority Level (IPL)

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 IPL, the interrupt is enabled.

• Reserved Area

When write to this bit, write "0". When read, its content is indeterminate.

2.3 Reset

There are three types of resets: a hardware reset, a software reset, and an watchdog timer reset.

2.3.1 Hardware Reset

A reset is applied using the $\overline{\text{RESET}}$ pin. When an “L” signal is applied to the $\overline{\text{RESET}}$ pin while the power supply voltage is within the recommended operating condition, the pins are initialized (see Table 2.3.1). The oscillation circuit is initialized and the main clock starts oscillating. When the input level at the $\overline{\text{RESET}}$ pin is released from “L” to “H”, the CPU and SFR are initialized, and the program is executed starting from the address indicated by the reset vector. The internal RAM is not initialized. If the $\overline{\text{RESET}}$ pin is pulled “L” while writing to the internal RAM, the internal RAM becomes indeterminate.

Figure 2.3.1 shows the example reset circuit. Figure 2.3.2 shows the reset sequence. Table 2.3.1 shows the statuses of the other pins while the $\overline{\text{RESET}}$ pin is “L”. Figure 2.3.3 shows the CPU register status after reset. Refer to “SFR” for SFR status after reset.

1. When the power supply is stable

- When START pin = “H”
 - (1) Apply an “L” signal to the $\overline{\text{RESET}}$ pin.
 - (2) Apply a clock for 20 cycles or more to the XIN pin.
 - (3) Apply an “H” signal to the $\overline{\text{RESET}}$ pin.
- When START pin = “L”
 - (1) Apply an “L” signal to the $\overline{\text{RESET}}$ pin.
 - (2) Apply a clock for 20 cycles or more to the XCIN pin.
 - (3) Apply an “H” signal to the $\overline{\text{RESET}}$ pin.

2. Power on

- When START pin = “H”
 - (1) Apply an “L” signal to the $\overline{\text{RESET}}$ pin.
 - (2) Let the power supply voltage increase until it meets the recommended operating condition.
 - (3) Wait $t_d(\text{P-R})$ or more until the internal power supply is stabilized.
 - (4) Apply a clock for 20 cycles or more to the XIN pin.
 - (5) Apply an “H” signal to the $\overline{\text{RESET}}$ pin.
- When START pin = “L”
 - (1) Apply an “L” signal to the $\overline{\text{RESET}}$ pin.
 - (2) Let the power supply voltage increase until it meets the recommended operating condition.
 - (3) Wait $t_d(\text{P-R})$ or more until the internal power supply is stabilized.
 - (4) Apply a clock for 20 cycles or more to the XCIN pin.
 - (5) Apply an “H” signal to the $\overline{\text{RESET}}$ pin.

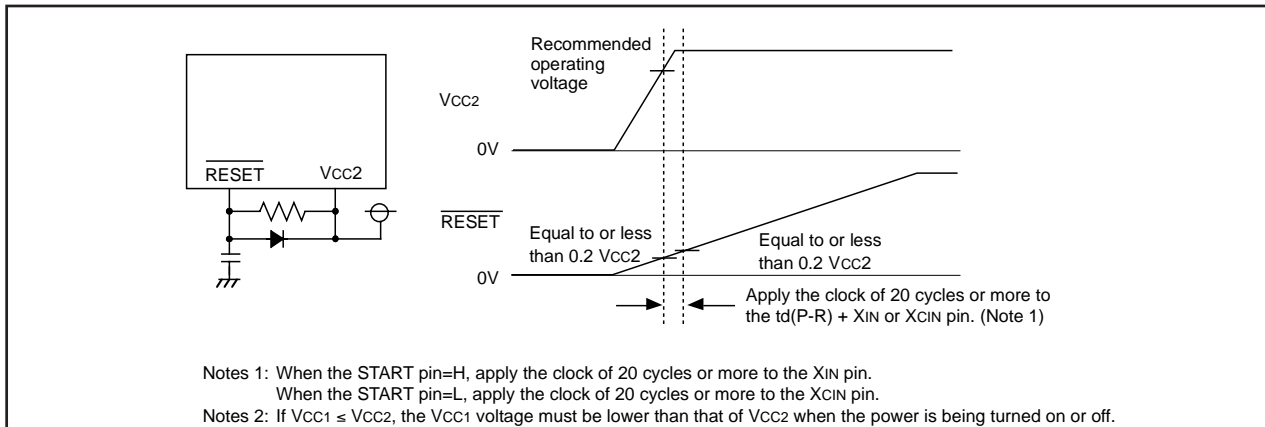


Figure 2.3.1. Example Reset Circuit

2.3.2 Software Reset

When the PM03 bit in the PM0 register is set to “1” (microcomputer reset), the microcomputer has its pins, CPU, and SFR initialized. Then the program is executed starting from the address indicated by the reset vector.

Select the main clock for the CPU clock source, and set the PM03 bit to “1” with main clock oscillation satisfactorily stable.

At software reset, some SFR’s are not initialized. Refer to “SFR”. Also, since the PM01 to PM00 bits in the PM0 register are not initialized, the processor mode remains unchanged.

2.3.3 Watchdog Timer Reset

Where the PM12 bit in the PM1 register is “1” (reset when watchdog timer underflows), the microcomputer initializes its pins, CPU and SFR if the watchdog timer underflows. Then the program is executed starting from the address indicated by the reset vector.

At watchdog timer reset, some SFR’s are not initialized. Refer to “SFR”. Also, since the PM01 to PM00 bits in the PM0 register are not initialized, the processor mode remains unchanged.

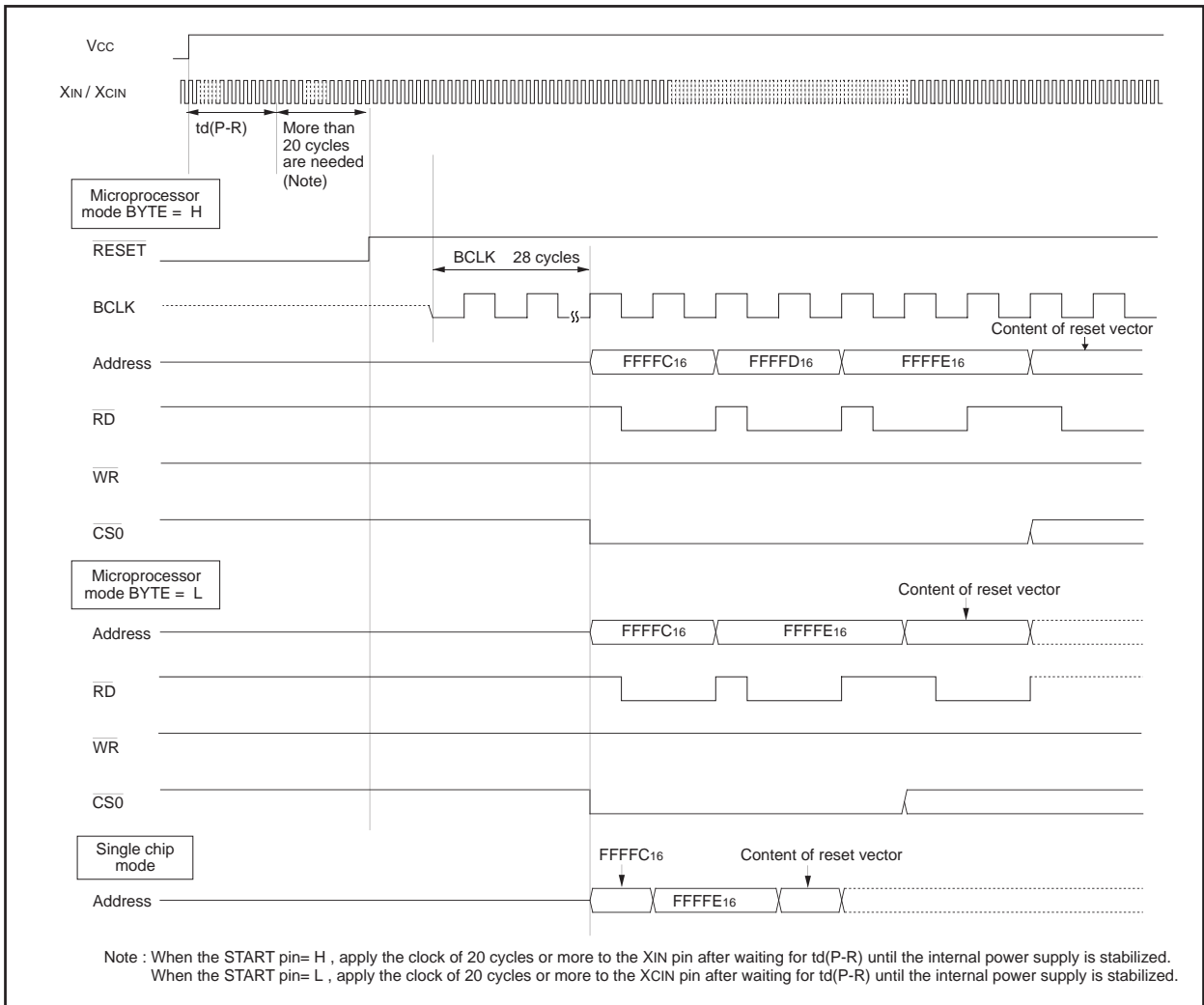


Figure 2.3.2. Reset Sequence

Table 2.3.1. Pin Status When RESET Pin Level is “L”

Pin name	Status		
	CNVss = Vss	CNVss = Vcc (Note)	
		BYTE = Vss	BYTE = Vcc
P0	Input port	Data input	Data input
P1	Input port	Data input	Input port
P2, P3, P40 to P43	Input port	Address output (undefined)	Address output (undefined)
P44	Input port	$\overline{CS0}$ output (“H” is output)	$\overline{CS0}$ output (“H” is output)
P45 to P47	Input port	Input port (Pulled high)	Input port (Pulled high)
P50	Input port	\overline{WR} output (“H” is output)	\overline{WR} output (“H” is output)
P51	Input port	\overline{BHE} output (undefined)	\overline{BHE} output (undefined)
P52	Input port	\overline{RD} output (“H” is output)	\overline{RD} output (“H” is output)
P53	Input port	BCLK output	BCLK output
P54	Input port	\overline{HLDA} output (The output value depends on the input to the HOLD pin)	\overline{HLDA} output (The output value depends on the input to the HOLD pin)
P55	Input port	\overline{HOLD} input	\overline{HOLD} input
P56	Input port	ALE output (“L” is output)	ALE output (“L” is output)
P57	Input port	\overline{RDY} input	\overline{RDY} input
P6, P7, P80 to P84, P86, P87, P9, P10	Input port	Input port	Input port
P11	Output port	Output port	Output port

Note : Connect the M1 pin to the VDD2 in the flash memory version of microcomputer.
 This is the state after internal power supply voltage is stabilized after a power supply voltage.
 It is undefined until internal power supply voltage is stabilized.

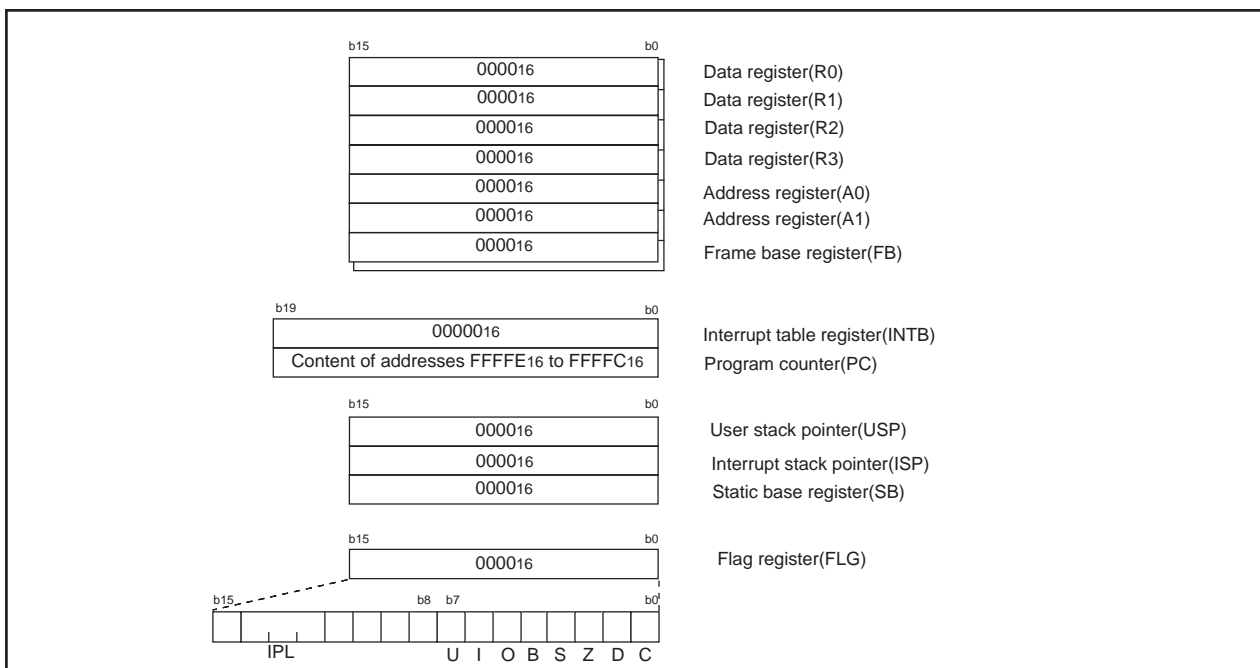


Figure 2.3.3. CPU Register Status After Rreset

2.3.4 SFR

Address	Register	Symbol	After reset
0000 ₁₆			
0001 ₁₆			
0002 ₁₆			
0003 ₁₆			
0004 ₁₆	Processor mode register 0 (Note 2)	PM0	00000000 ₂ (the CNVss pin is "L") 00000011 ₂ (the CNVss pin is "H" (Note 5))
0005 ₁₆	Processor mode register 1	PM1	00001000 ₂
0006 ₁₆	System clock control register 0	CM0	01001000 ₂ (the START pin is "H" (Note 4))
0007 ₁₆	System clock control register 1	CM1	00100000 ₂
0008 ₁₆	Chip select control register	CSR	00000001 ₂
0009 ₁₆	Address match interrupt enable register	AIER	XXXXXXXX00 ₂
000A ₁₆	Protect register	PRCR	XX000000 ₂
000B ₁₆			
000C ₁₆			
000D ₁₆			
000E ₁₆	Watchdog timer start register	WDTS	XX ₁₆
000F ₁₆	Watchdog timer control register	WDC	00XXXXXXXX ₂ (Note 3)
0010 ₁₆	Address match interrupt register 0	RMAD0	00 ₁₆
0011 ₁₆			00 ₁₆
0012 ₁₆			X0 ₁₆
0013 ₁₆			
0014 ₁₆	Address match interrupt register 1	RMAD1	00 ₁₆
0015 ₁₆			00 ₁₆
0016 ₁₆			X0 ₁₆
0017 ₁₆			
0018 ₁₆			
0019 ₁₆			
001A ₁₆			
001B ₁₆	Chip select expansion control register	CSE	00 ₁₆
001C ₁₆			
001D ₁₆			
001E ₁₆	Processor mode register 2	PM2	XXX00000 ₂
001F ₁₆			
0020 ₁₆	DMA0 source pointer	SAR0	XX ₁₆
0021 ₁₆			XX ₁₆
0022 ₁₆			XX ₁₆
0023 ₁₆			
0024 ₁₆	DMA0 destination pointer	DAR0	XX ₁₆
0025 ₁₆			XX ₁₆
0026 ₁₆			XX ₁₆
0027 ₁₆			
0028 ₁₆	DMA0 transfer counter	TCR0	XX ₁₆
0029 ₁₆			XX ₁₆
002A ₁₆			
002B ₁₆			
002C ₁₆	DMA0 control register	DM0CON	00000X00 ₂
002D ₁₆			
002E ₁₆			
002F ₁₆			
0030 ₁₆	DMA1 source pointer	SAR1	XX ₁₆
0031 ₁₆			XX ₁₆
0032 ₁₆			XX ₁₆
0033 ₁₆			
0034 ₁₆	DMA1 destination pointer	DAR1	XX ₁₆
0035 ₁₆			XX ₁₆
0036 ₁₆			XX ₁₆
0037 ₁₆			
0038 ₁₆	DMA1 transfer counter	TCR1	XX ₁₆
0039 ₁₆			XX ₁₆
003A ₁₆			
003B ₁₆			
003C ₁₆	DMA1 control register	DM1CON	00000X00 ₂
003D ₁₆			
003E ₁₆			
003F ₁₆			

Note 1: The blank areas are reserved and cannot be accessed by users.

Note 2: The PM00 and PM01 bits do not change at software reset, watchdog timer reset and oscillation stop detection reset.

Note 3: The WDC5 bit is "0" (cold start) immediately after power-on. It can only be set to "1" in a program.

Note 4: 01111000₂ when the START pin is "L."

Note 5: The CNVss pin and the M1 pin are "H" in the flash memory version.

X : Undefined

Address	Register	Symbol	After reset
0040 ₁₆			
0041 ₁₆			
0042 ₁₆			
0043 ₁₆			
0044 ₁₆	INT3 interrupt control register	INT3IC	XX00X0002
0045 ₁₆	Timer B5/SLICE ON interrupt control register	TB5IC	XXXXX0002
0046 ₁₆	Timer B4/Remote control interrupt control register, UART1 BUS collision detection interrupt control register	TB4IC, U1BCNIC	XXXXX0002
0047 ₁₆	Timer B3/HINT interrupt control register, UART0 BUS collision detection interrupt control register	TB3IC, U0BCNIC	XXXXX0002
0048 ₁₆	SI/O4 interrupt control register (S4IC), INT5 interrupt control register	S4IC, INT5IC	XX00X0002
0049 ₁₆	SI/O3 interrupt control register, INT4 interrupt control register	S3IC, INT4IC	XX00X0002
004A ₁₆	UART2 Bus collision detection interrupt control register	BCNIC	XXXXX0002
004B ₁₆	DMA0 interrupt control register	DM0IC	XXXXX0002
004C ₁₆	DMA1 interrupt control register	DM1IC	XXXXX0002
004D ₁₆	Key input interrupt control register	KUPIC	XXXXX0002
004E ₁₆	A-D conversion interrupt control register	ADIC	XXXXX0002
004F ₁₆	UART2 transmit interrupt control register	S2TIC	XXXXX0002
0050 ₁₆	UART2 receive interrupt control register	S2RIC	XXXXX0002
0051 ₁₆	UART0 transmit interrupt control register	S0TIC	XXXXX0002
0052 ₁₆	UART0 receive interrupt control register	S0RIC	XXXXX0002
0053 ₁₆	UART1 transmit interrupt control register	S1TIC	XXXXX0002
0054 ₁₆	UART1 receive interrupt control register	S1RIC	XXXXX0002
0055 ₁₆	Timer A0 interrupt control register	TA0IC	XXXXX0002
0056 ₁₆	Timer A1 interrupt control register	TA1IC	XXXXX0002
0057 ₁₆	Timer A2 interrupt control register	TA2IC	XXXXX0002
0058 ₁₆	Timer A3 interrupt control register	TA3IC	XXXXX0002
0059 ₁₆	Timer A4 interrupt control register	TA4IC	XXXXX0002
005A ₁₆	Timer B0 interrupt control register	TB0IC	XXXXX0002
005B ₁₆	Timer B1 interrupt control register	TB1IC	XXXXX0002
005C ₁₆	Timer B2/Clock timer interrupt control register	TB2IC	XXXXX0002
005D ₁₆	INT0 interrupt control register	INT0IC	XX00X0002
005E ₁₆	INT1 interrupt control register	INT1IC	XX00X0002
005F ₁₆	INT2 interrupt control register	INT2IC	XX00X0002
0060 ₁₆			
0061 ₁₆			
0062 ₁₆			
0063 ₁₆			
0064 ₁₆			
0065 ₁₆			
0066 ₁₆			
0067 ₁₆			
0068 ₁₆			
0069 ₁₆			
006A ₁₆			
006B ₁₆			
006C ₁₆			
006D ₁₆			
006E ₁₆			
006F ₁₆			
0070 ₁₆			
0071 ₁₆			
0072 ₁₆			
0073 ₁₆			
0074 ₁₆			
0075 ₁₆			
0076 ₁₆			
0077 ₁₆			
0078 ₁₆			
0079 ₁₆			
007A ₁₆			
007B ₁₆			
007C ₁₆			
007D ₁₆			
007E ₁₆			
007F ₁₆			

Note :The blank areas are reserved and cannot be accessed by users.

X : Undefined

Address	Register	Symbol	After reset
0080 ₁₆			
0081 ₁₆			
0082 ₁₆			
0083 ₁₆			
0084 ₁₆			
0085 ₁₆			
0086 ₁₆			
≈			≈
01B0 ₁₆			
01B1 ₁₆			
01B2 ₁₆			
01B3 ₁₆			
01B4 ₁₆			
01B5 ₁₆	Flash memory control register 1 (Note 2)	FMR1	0X00XX0X2
01B6 ₁₆			
01B7 ₁₆	Flash memory control register 0 (Note 2)	FMR0	XX0000012
01B8 ₁₆	Address match interrupt register 2	RMAD2	0016
01B9 ₁₆			0016
01BA ₁₆			X016
01BB ₁₆	Address match interrupt enable register 2	AIER2	XXXXXX002
01BC ₁₆	Address match interrupt register 3	RMAD3	0016
01BD ₁₆			0016
01BE ₁₆			X016
01BF ₁₆			
≈			≈
020E ₁₆	Slice RAM address control register	SA	0016
020F ₁₆			
0210 ₁₆	Slice RAM data control register	SD	0016
0211 ₁₆			
0212 ₁₆	Address control register for CRC registers	CA	0016
0213 ₁₆			
0214 ₁₆	Data control register for CRC registers	CD	0016
0215 ₁₆			
0216 ₁₆	Address control register for extended registers	DA	0016
0217 ₁₆			
0218 ₁₆	Data control register for extended registers	DD	0016
0219 ₁₆			
021A ₁₆	Humming 8/4 register	HM8	0016
021B ₁₆			
021C ₁₆	Humming 24/18 register 0	HM0	0016
021D ₁₆			
021E ₁₆	Humming 24/18 register 1	HM1	0016
021F ₁₆			
0250 ₁₆			
≈			≈
0259 ₁₆			
025A ₁₆			
025B ₁₆			
025C ₁₆			
025D ₁₆			
025E ₁₆	Peripheral clock select register	PCLKR	000000112
025F ₁₆			
≈			≈
0330 ₁₆			
0331 ₁₆			
0332 ₁₆			
0333 ₁₆			
0334 ₁₆			
0335 ₁₆			
0336 ₁₆			
0337 ₁₆			
0338 ₁₆			
0339 ₁₆			
033A ₁₆			
033B ₁₆			
033C ₁₆			
033D ₁₆			
033E ₁₆			
033F ₁₆			

Note 1: The blank areas are reserved and cannot be accessed by users.

Note 2: This register is included in the flash memory version.

X : Undefined

Address	Register	Symbol	After reset
0340 ₁₆	Timer B3, 4, 5 count start flag	TBSR	000XXXXX ₂
0341 ₁₆			
0342 ₁₆			
0343 ₁₆			
0344 ₁₆			
0345 ₁₆			
0346 ₁₆			
0347 ₁₆			
0348 ₁₆			
0349 ₁₆			
034A ₁₆			
034B ₁₆			
034C ₁₆			
034D ₁₆			
034E ₁₆			
034F ₁₆			
0350 ₁₆	Timer B3 register	TB3	XX ₁₆
0351 ₁₆			XX ₁₆
0352 ₁₆	Timer B4 register	TB4	XX ₁₆
0353 ₁₆			XX ₁₆
0354 ₁₆	Timer B5 register	TB5	XX ₁₆
0355 ₁₆			XX ₁₆
0356 ₁₆			
0357 ₁₆			
0358 ₁₆			
0359 ₁₆			
035A ₁₆			
035B ₁₆	Timer B3 mode register	TB3MR	00XX0000 ₂
035C ₁₆	Timer B4 mode register	TB4MR	00XX0000 ₂
035D ₁₆	Timer B5 mode register	TB5MR	00XX0000 ₂
035E ₁₆	Interrupt cause select register 2	IFSR2A	00XXXXXX ₂
035F ₁₆	Interrupt cause select register	IFSR	00 ₁₆
0360 ₁₆	SI/O3 transmit/receive register	S3TRR	XX ₁₆
0361 ₁₆			
0362 ₁₆	SI/O3 control register	S3C	01000000 ₂
0363 ₁₆	SI/O3 bit rate generator	S3BRG	XX ₁₆
0364 ₁₆	SI/O4 transmit/receive register	S4TRR	XX ₁₆
0365 ₁₆			
0366 ₁₆	SI/O4 control register	S4C	01000000 ₂
0367 ₁₆	SI/O4 bit rate generator	S4BRG	XX ₁₆
0368 ₁₆			
0369 ₁₆			
036A ₁₆			
036B ₁₆			
036C ₁₆	UART0 special mode register 4	U0SMR4	00 ₁₆
036D ₁₆	UART0 special mode register 3	U0SMR3	000X0X0X ₂
036E ₁₆	UART0 special mode register 2	U0SMR2	X0000000 ₂
036F ₁₆	UART0 special mode register	U0SMR	X0000000 ₂
0370 ₁₆	UART1 special mode register 4	U1SMR4	00 ₁₆
0371 ₁₆	UART1 special mode register 3	U1SMR3	000X0X0X ₂
0372 ₁₆	UART1 special mode register 2	U1SMR2	X0000000 ₂
0373 ₁₆	UART1 special mode register	U1SMR	X0000000 ₂
0374 ₁₆	UART2 special mode register 4	U2SMR4	00 ₁₆
0375 ₁₆	UART2 special mode register 3	U2SMR3	000X0X0X ₂
0376 ₁₆	UART2 special mode register 2	U2SMR2	X0000000 ₂
0377 ₁₆	UART2 special mode register	U2SMR	X0000000 ₂
0378 ₁₆	UART2 transmit/receive mode register	U2MR	00 ₁₆
0379 ₁₆	UART2 bit rate generator	U2BRG	XX ₁₆
037A ₁₆	UART2 transmit buffer register	U2TB	XXXXXXXX ₂
037B ₁₆			XXXXXXXX ₂
037C ₁₆	UART2 transmit/receive control register 0	U2C0	00001000 ₂
037D ₁₆	UART2 transmit/receive control register 1	U2C1	00000010 ₂
037E ₁₆	UART2 receive buffer register	U2RB	XXXXXXXX ₂
037F ₁₆			XXXXXXXX ₂

Note : The blank areas are reserved and cannot be accessed by users.

X : Undefined

Address	Register	Symbol	After reset
0380 ₁₆	Count start flag	TABSR	0016
0381 ₁₆	Clock prescaler reset flag	CPSRF	0XXXXXXXX2
0382 ₁₆	One-shot start flag	ONSF	0016
0383 ₁₆	Trigger select register	TRGSR	0016
0384 ₁₆	Up-down flag	UDF	0016
0385 ₁₆			
0386 ₁₆ 0387 ₁₆	Timer A0 register	TA0	XX16 XX16
0388 ₁₆ 0389 ₁₆	Timer A1 register	TA1	XX16 XX16
038A ₁₆ 038B ₁₆	Timer A2 register	TA2	XX16 XX16
038C ₁₆ 038D ₁₆	Timer A3 register	TA3	XX16 XX16
038E ₁₆ 038F ₁₆	Timer A4 register	TA4	XX16 XX16
0390 ₁₆ 0391 ₁₆	Timer B0 register	TB0	XX16 XX16
0392 ₁₆ 0393 ₁₆	Timer B1 register	TB1	XX16 XX16
0394 ₁₆ 0395 ₁₆	Timer B2 register	TB2	XX16 XX16
0396 ₁₆	Timer A0 mode register	TA0MR	0016
0397 ₁₆	Timer A1 mode register	TA1MR	0016
0398 ₁₆	Timer A2 mode register	TA2MR	0016
0399 ₁₆	Timer A3 mode register	TA3MR	0016
039A ₁₆	Timer A4 mode register	TA4MR	0016
039B ₁₆	Timer B0 mode register	TB0MR	00XX00002
039C ₁₆	Timer B1 mode register	TB1MR	00XX00002
039D ₁₆	Timer B2 mode register	TB2MR	00XX00002
039E ₁₆			
039F ₁₆			
03A0 ₁₆	UART0 transmit/receive mode register	U0MR	0016
03A1 ₁₆	UART0 bit rate generator	U0BRG	XX16
03A2 ₁₆ 03A3 ₁₆	UART0 transmit buffer register	U0TB	XXXXXXXXX2 XXXXXXXXX2
03A4 ₁₆	UART0 transmit/receive control register 0	U0C0	000010002
03A5 ₁₆	UART0 transmit/receive control register 1	U0C1	000000102
03A6 ₁₆ 03A7 ₁₆	UART0 receive buffer register	U0RB	XXXXXXXXX2 XXXXXXXXX2
03A8 ₁₆	UART1 transmit/receive mode register	U1MR	0016
03A9 ₁₆	UART1 bit rate generator	U1BRG	XX16
03AA ₁₆ 03AB ₁₆	UART1 transmit buffer register	U1TB	XXXXXXXXX2 XXXXXXXXX2
03AC ₁₆	UART1 transmit/receive control register 0	U1C0	000010002
03AD ₁₆	UART1 transmit/receive control register 1	U1C1	000000102
03AE ₁₆ 03AF ₁₆	UART1 receive buffer register	U1RB	XXXXXXXXX2 XXXXXXXXX2
03B0 ₁₆	UART transmit/receive control register 2	UCON	X00000002
03B1 ₁₆			
03B2 ₁₆			
03B3 ₁₆			
03B4 ₁₆			
03B5 ₁₆			
03B6 ₁₆			
03B7 ₁₆			
03B8 ₁₆	DMA0 request cause select register	DM0SL	0016
03B9 ₁₆			
03BA ₁₆	DMA1 request cause select register	DM1SL	0016
03BB ₁₆			
03BC ₁₆ 03BD ₁₆	CRC data register	CRCD	XX16 XX16
03BE ₁₆	CRC input register	CRCIN	XX16
03BF ₁₆			

Note : The blank areas are reserved and cannot be accessed by users.
X : Undefined

Address	Register	Symbol	After reset
03C0 ₁₆	A-D register 0	AD0	XXXXXXXX ₂
03C1 ₁₆			
03C2 ₁₆	A-D register 1	AD1	XXXXXXXX ₂
03C3 ₁₆			
03C4 ₁₆	A-D register 2	AD2	XXXXXXXX ₂
03C5 ₁₆			
03C6 ₁₆	A-D register 3	AD3	XXXXXXXX ₂
03C7 ₁₆			
03C8 ₁₆	A-D register 4	AD4	XXXXXXXX ₂
03C9 ₁₆			
03CA ₁₆	A-D register 5	AD5	XXXXXXXX ₂
03CB ₁₆			
03CC ₁₆	A-D register 6	AD6	XXXXXXXX ₂
03CD ₁₆			
03CE ₁₆	A-D register 7	AD7	XXXXXXXX ₂
03CF ₁₆			
03D0 ₁₆			
03D1 ₁₆			
03D2 ₁₆			
03D3 ₁₆			
03D4 ₁₆	A-D control register 2	ADCON2	0016
03D5 ₁₆			
03D6 ₁₆	A-D control register 0	ADCON0	0000XXX ₂
03D7 ₁₆	A-D control register 1	ADCON1	0016
03D8 ₁₆			
03D9 ₁₆			
03DA ₁₆			
03DB ₁₆			
03DC ₁₆			
03DD ₁₆			
03DE ₁₆			
03DF ₁₆			
03E0 ₁₆	Port P0 register	P0	XX ₁₆
03E1 ₁₆	Port P1 register	P1	XX ₁₆
03E2 ₁₆	Port P0 direction register	PD0	0016
03E3 ₁₆	Port P1 direction register	PD1	0016
03E4 ₁₆	Port P2 register	P2	XX ₁₆
03E5 ₁₆	Port P3 register	P3	XX ₁₆
03E6 ₁₆	Port P2 direction register	PD2	0016
03E7 ₁₆	Port P3 direction register	PD3	0016
03E8 ₁₆	Port P4 register	P4	XX ₁₆
03E9 ₁₆	Port P5 register	P5	XX ₁₆
03EA ₁₆	Port P4 direction register	PD4	0016
03EB ₁₆	Port P5 direction register	PD5	0016
03EC ₁₆	Port P6 register	P6	XX ₁₆
03ED ₁₆	Port P7 register	P7	XX ₁₆
03EE ₁₆	Port P6 direction register	PD6	0016
03EF ₁₆	Port P7 direction register	PD7	0016
03F0 ₁₆	Port P8 register	P8	XX ₁₆
03F1 ₁₆	Port P9 register	P9	XX ₁₆
03F2 ₁₆	Port P8 direction register	PD8	00X00000 ₂
03F3 ₁₆	Port P9 direction register	PD9	0016
03F4 ₁₆	Port P10 register	P10	XX ₁₆
03F5 ₁₆			
03F6 ₁₆	Port P10 direction register	PD10	0016
03F7 ₁₆			
03F8 ₁₆			
03F9 ₁₆			
03FA ₁₆			
03FB ₁₆			
03FC ₁₆	Pull-up control register 0	PUR0	0016
03FD ₁₆	Pull-up control register 1	PUR1	00000000 ₂ 00000010 ₂ (Note 2)
03FE ₁₆	Pull-up control register 2	PUR2	0016
03FF ₁₆	Port control register	PCR	0016

Note 1: The blank areas are reserved and cannot be accessed by users.

Note 2: At hardware reset, the register is as follows:

- "00000000₂" where "L" is inputted to the CNV_{ss} pin
- "00000010₂" where "H" is inputted to the CNV_{ss} pin and the M1 pin (flash memory version of microcomputer)
- "00000010₂" where "H" is inputted to the CNV_{ss} pin (mask ROM version).

At software reset, watchdog timer reset and oscillation stop detection reset, the register is as follows:

- "00000000₂" where the PM01 to PM00 bits in the PM0 register are "00₂" (single-chip mode)
- "00000010₂" where the PM01 to PM00 bits in the PM0 register are "01₂" (memory expansion mode) or "11₂" (microprocessor mode)

X : Undefined

2.4 Processor Mode

(1) Types of Processor Mode

Three processor modes are available to choose from: single-chip mode, memory expansion mode, and microprocessor mode. Table 2.4.1 shows the features of these processor modes.

Table 2.4.1. Features of Processor Modes

Processor modes	Access space	Pins which are assigned I/O ports
Single-chip mode	SFR, internal RAM, internal ROM	All pins are I/O ports or peripheral function I/O pins
Memory expansion mode	SFR, internal RAM, internal ROM, external area (Note)	Some pins serve as bus control pins (Note)
Microprocessor mode	SFR, internal RAM, external area (Note)	Some pins serve as bus control pins (Note)

Note : Refer to "Bus".

(2) Setting Processor Modes

Processor mode is set by using the CNVss pin and the PM01 to PM00 bits in the PM0 register.

Table 2.4.2 shows the processor mode after hardware reset. Table 2.4.3 shows the PM01 to PM00 bit set values and processor modes.

In the flash memory version, after hardware reset, apply the CNVss pin and the M1 pin to Vcc when use microprocessor mode. In the mask ROM version, after hardware reset, apply the CNVss pin to Vcc when use microprocessor mode.

Table 2.4.2. Processor Mode After Hardware Reset

CNVSS pin input level	Processor mode
Vss	Single-chip mode
Vcc (Note 1, Note 2)	Microprocessor mode

Note 1: If the microcomputer is reset in hardware by applying Vcc to the CNVss pin and the M1 pin in the flash memory version (by applying Vcc to the CNVss pin in the mask ROM version) the internal ROM cannot be accessed regardless of PM10 to PM00 bits.

Note 2: The multiplexed bus cannot be assigned to the entire \overline{CS} space.

Table 2.4.3. PM01 to PM00 Bits Set Values and Processor Modes

PM01 to PM00 bits	Processor modes
002	Single-chip mode
012	Memory expansion mode
102	Must not be set
112	Microprocessor mode

Rewriting the PM01 to PM00 bits places the microcomputer in the corresponding processor mode regardless of whether the input level on the CNVss pin is "H" or "L". Note, however, that the PM01 to PM00 bits cannot be rewritten to "012" (memory expansion mode) or "112" (microprocessor mode) at the same time the PM07 to PM02 bits are rewritten. Note also that these bits cannot be rewritten to enter microprocessor mode in the internal ROM, nor can they be rewritten to exit microprocessor mode in areas overlapping the internal ROM.

If the microcomputer is reset in hardware by applying Vcc to the CNVss pin and the M1 pin in the flash memory version (by applying Vcc to the CNVss pin in the mask ROM version), the internal ROM cannot be accessed regardless of PM01 to PM00 bits.

Figures 2.4.1 and 2.4.2 show the registers associated with processor modes. Figure 2.4.3 show the memory map in single chip mode.

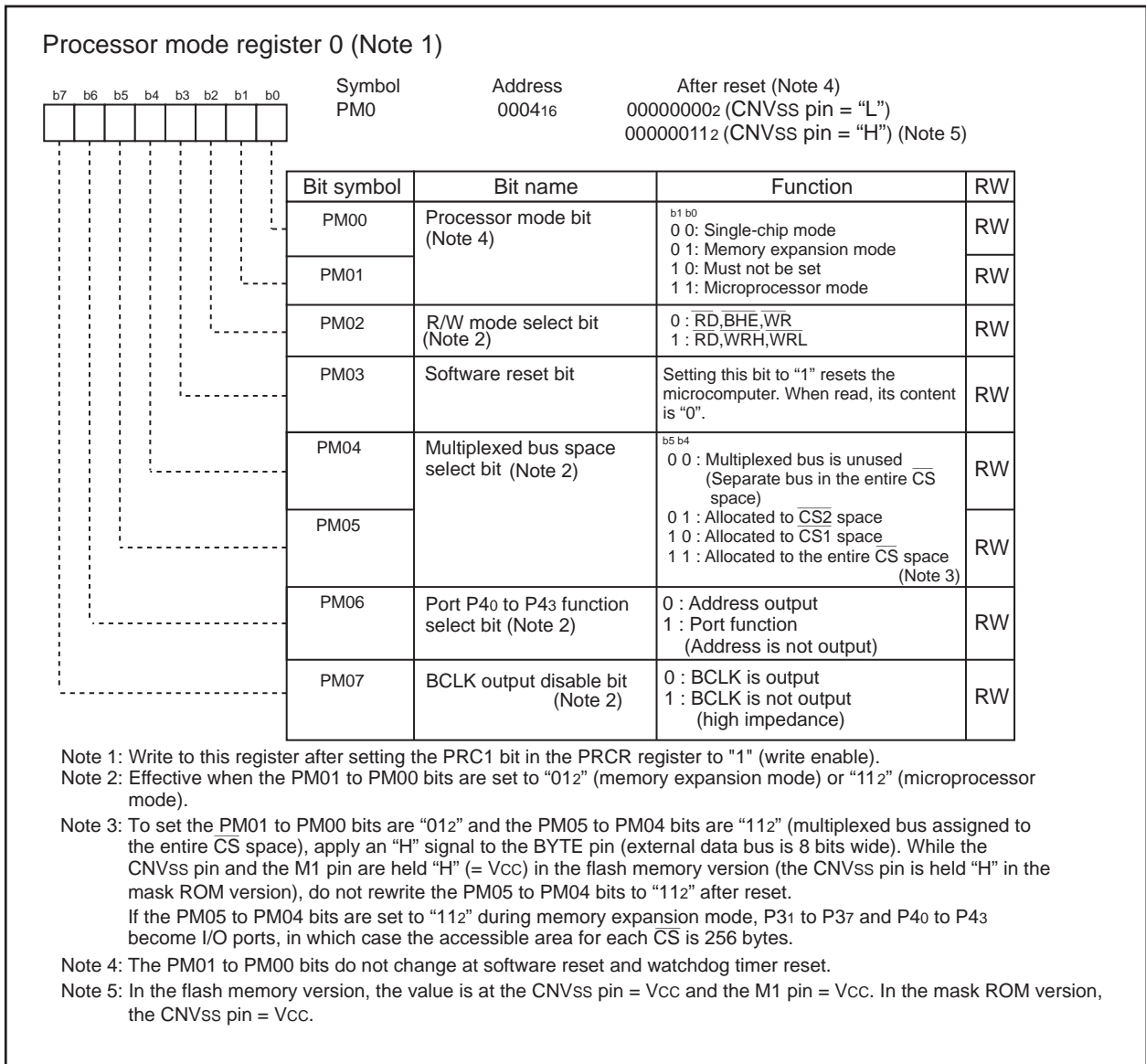


Figure 2.4.1. PM0 Register

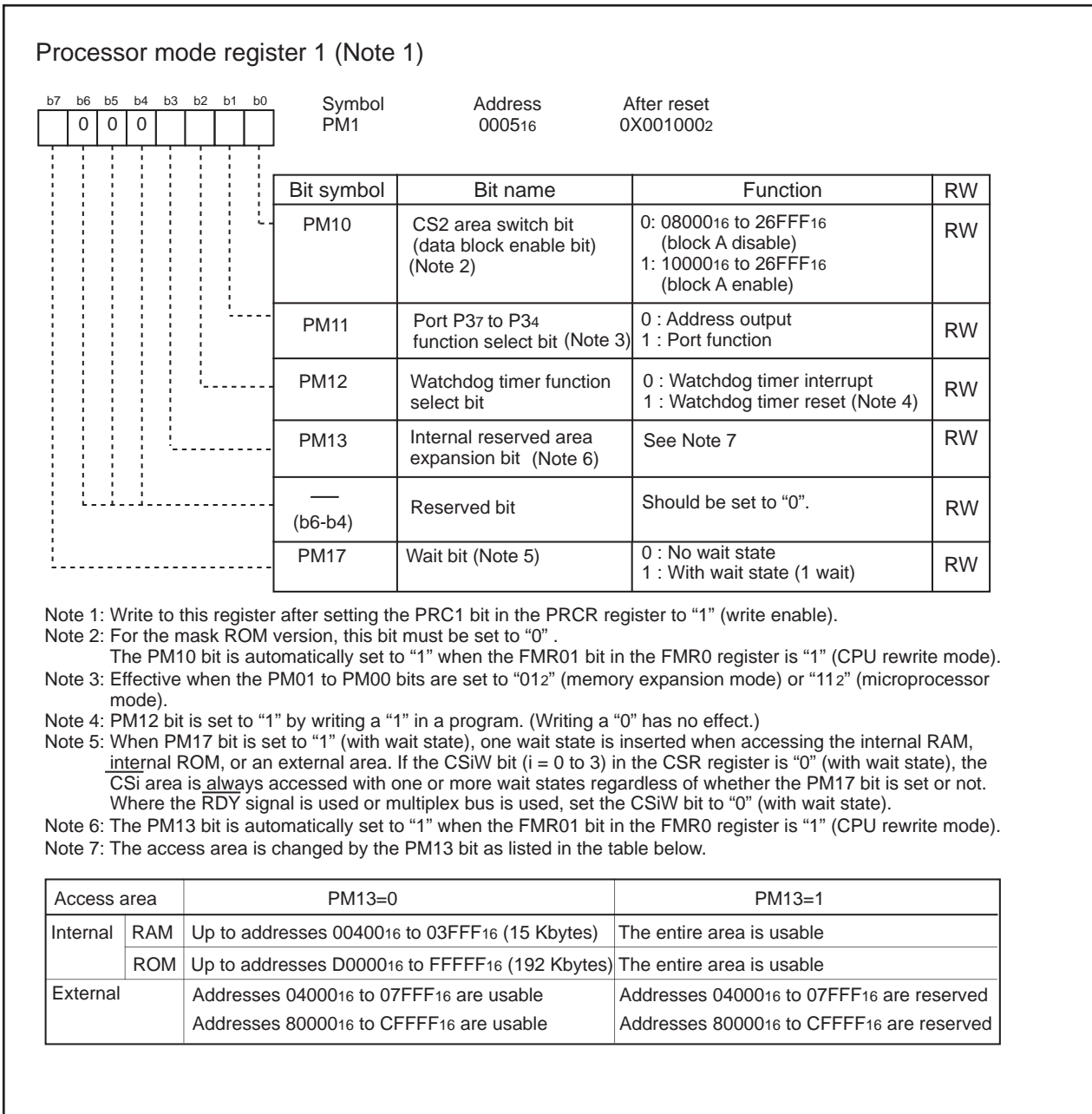


Figure 2.4.2. PM1 Register

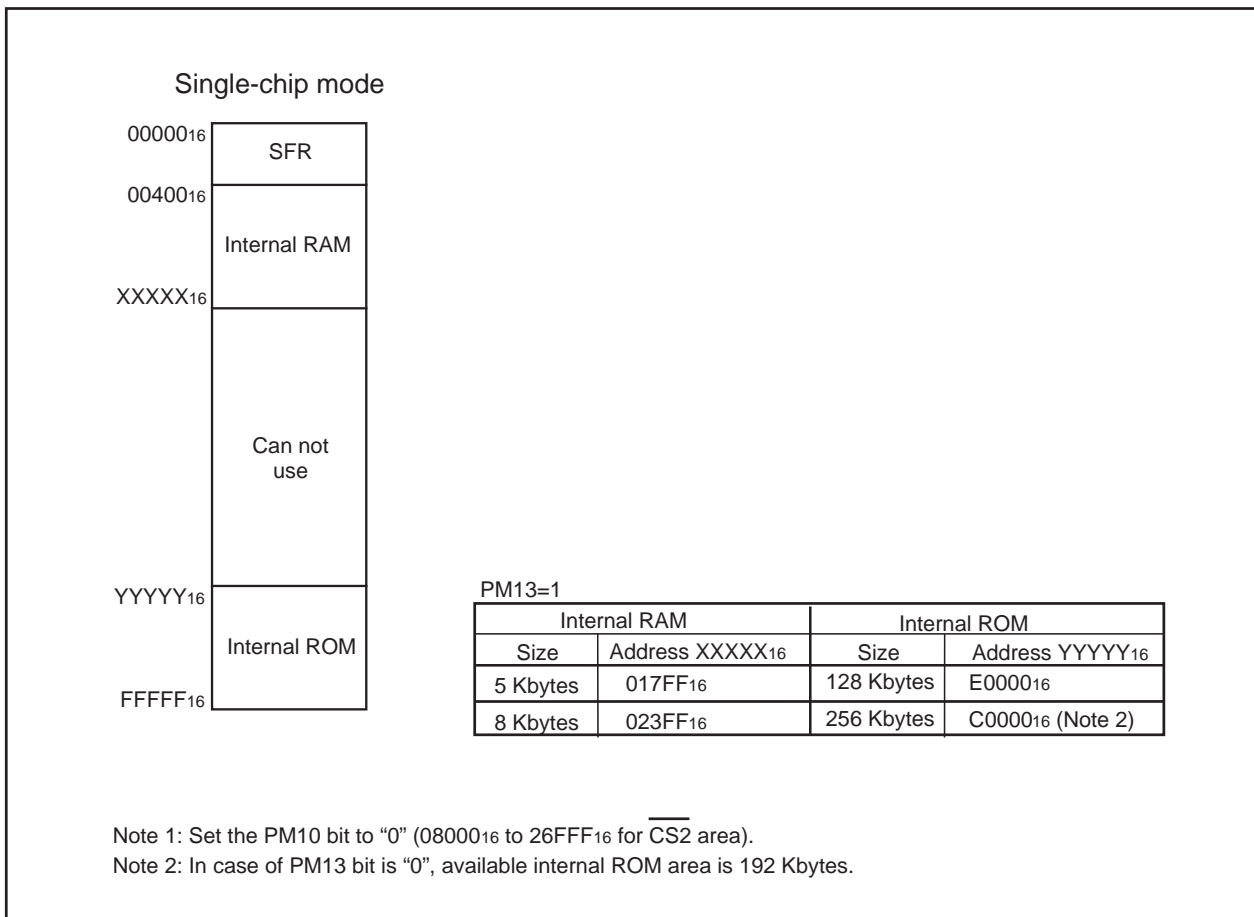


Figure 2.4.3. Memory Map in Single Chip Mode

2.4.1 Bus

During memory expansion or microprocessor mode, some pins serve as the bus control pins to perform data input/output to and from external devices. These bus control pins include A0 to A19, D0 to D15, $\overline{CS0}$ to $\overline{CS3}$, \overline{RD} , $\overline{WRL}/\overline{WR}$, $\overline{WRH}/\overline{BHE}$, \overline{ALE} , \overline{RDY} , \overline{HOLD} , \overline{HLDA} and \overline{BCLK} .

Bus Mode

The bus mode, either multiplexed or separate, can be selected using the PM05 to PM04 bits in the PM0 register.

Separate Bus

In this bus mode, data and address are separate.

Multiplexed Bus

In this bus mode, data and address are multiplexed.

- **When the input level on BYTE pin is high (8-bit data bus)**

D0 to D7 and A0 to A7 are multiplexed.

- **When the input level on BYTE pin is low (16-bit data bus)**

D0 to D7 and A1 to A8 are multiplexed. D8 to D15 are not multiplexed. Do not use D8 to D15.

External devices connecting to a multiplexed bus are allocated to only the even addresses of the microcomputer. Odd addresses cannot be accessed.

2.4.2 Bus Control

The following describes the signals needed for accessing external devices and the functionality of software wait.

(1) Address Bus

The address bus consists of 20 lines, A0 to A19. The address bus width can be chosen to be 12, 16 or 20 bits by using the PM06 bit in the PM0 register and the PM11 bit in the PM1 register. Table 2.4.4 shows the PM06 and PM11 bit set values and address bus widths.

Table 2.4.4. PM06 and PM11 Bits Set Value and Address Bus Width

Set value(Note)	Pin function	Address bus wide
PM11=1	P34 to P37	12 bits
PM06=1	P40 to P43	
PM11=0	A12 to A15	16 bits
PM06=1	P40 to P43	
PM11=0	A12 to A15	20 bits
PM06=0	A16 to A19	

Note 1: No values other than those shown above can be set.

When processor mode is changed from single-chip mode to memory extension mode, the address bus is indeterminate until any external area is accessed.

(2) Data Bus

When input on the BYTE pin is high(data bus is 8 bits wide), 8 lines D0 to D7 comprise the data bus; when input on the BYTE pin is low(data bus is 16 bits wide), 16 lines D0 to D15 comprise the data bus. Do not change the input level on the BYTE pin while in operation.

(3) Chip Select Signal

The chip select (hereafter referred to as the \overline{CS}_i) signals are output from the \overline{CS}_i ($i = 0$ to 3) pins. These pins can be chosen to function as I/O ports or as \overline{CS} by using the CSi bit in the CSR register. Figure 2.4.4 shows the CSR register.

During 1 Mbyte mode, the external area can be separated into up to 4 by the \overline{CS}_i signal which is output from the \overline{CS}_i pin. Figure 2.4.5 shows the example of address bus and \overline{CS}_i signal output in 1 Mbyte mode. Figure 2.4.6 to 2.4.7 show \overline{CS} area in 1 Mbyte mode.

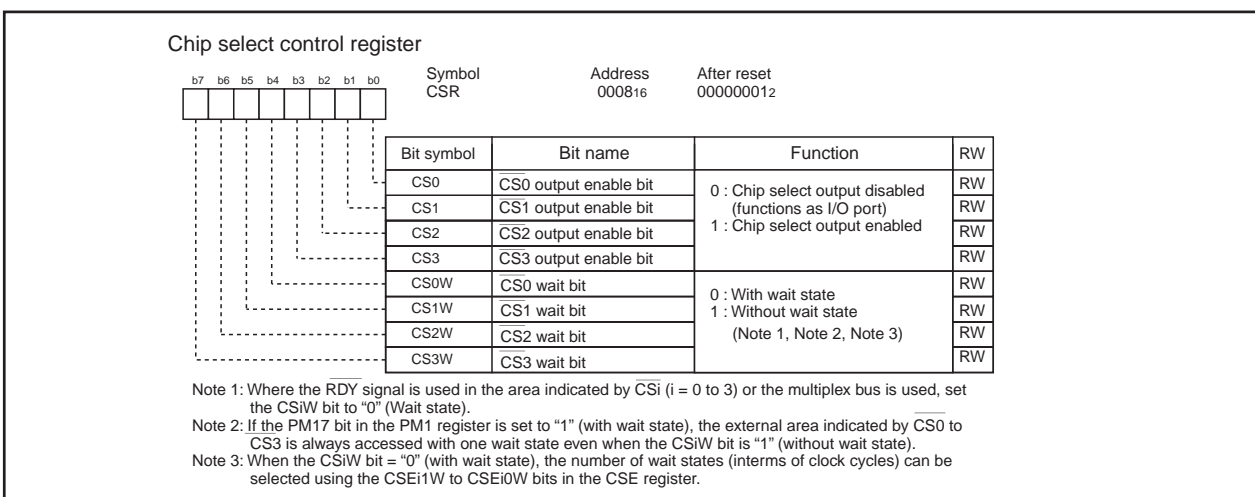
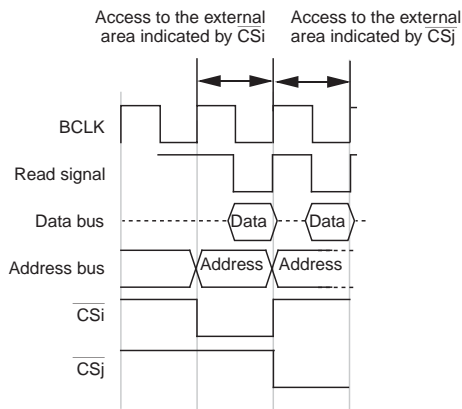


Figure 2.4.4. CSR Register

Example 1

To access the external area indicated by \overline{CS}_j in the next cycle after accessing the external area indicated by \overline{CS}_i

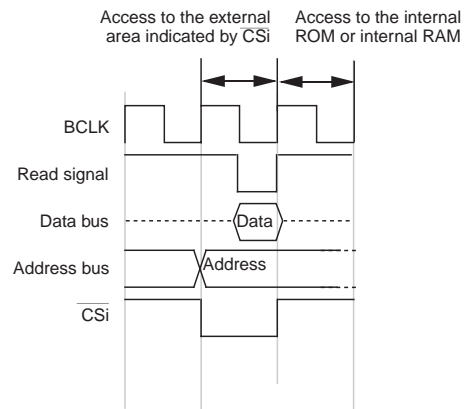
The address bus and the chip select signal both change state between these two cycles.



Example 2

To access the internal ROM or internal RAM in the next cycle after accessing the external area indicated by \overline{CS}_i

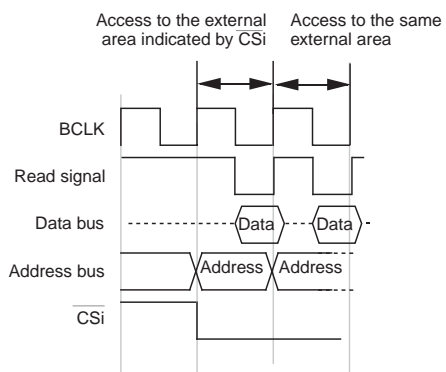
The chip select signal changes state but the address bus does not change state



Example 3

To access the external area indicated by \overline{CS}_i in the next cycle after accessing the external area indicated by the same \overline{CS}_i

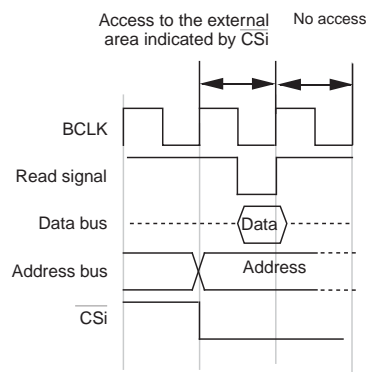
The address bus changes state but the chip select signal does not change state



Example 4

Not to access any area (nor instruction prefetch generated) in the next cycle after accessing the external area indicated by \overline{CS}_i

Neither the address bus nor the chip select signal changes state between these two cycles



Note : These examples show the address bus and chip select signal when accessing areas in two successive cycles. The chip select bus cycle may be extended more than two cycles depending on a combination of these examples.

Shown above is the case where separate bus is selected and the area is accessed for read without wait states. $i = 0$ to 3 , $j = 0$ to 3 (not including i , however)

Figure 2.4.5. Example of Address Bus and \overline{CS}_i Signal Output in 1 Mbyte Mode

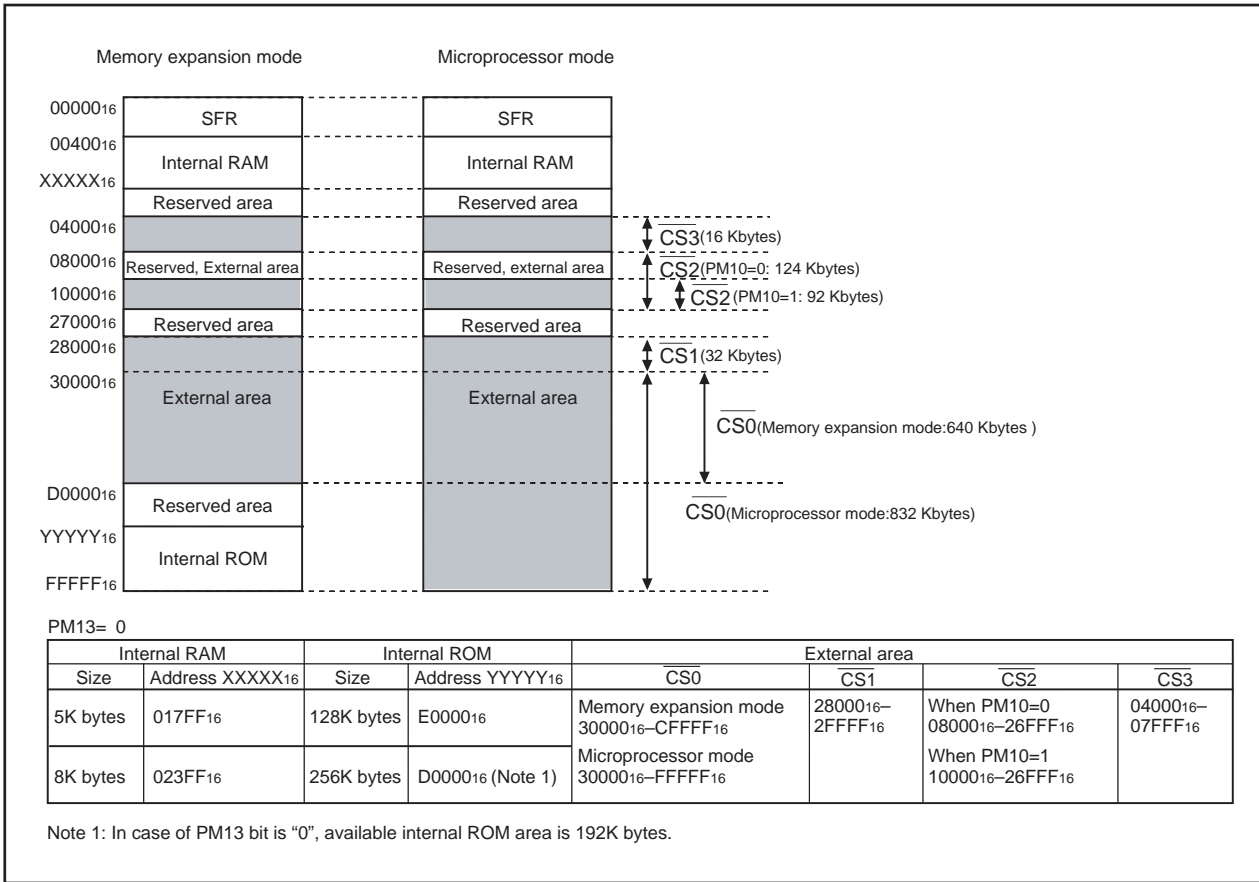


Figure 2.4.6. \overline{CS} Area in 1 Mbyte Mode (PM13=0)

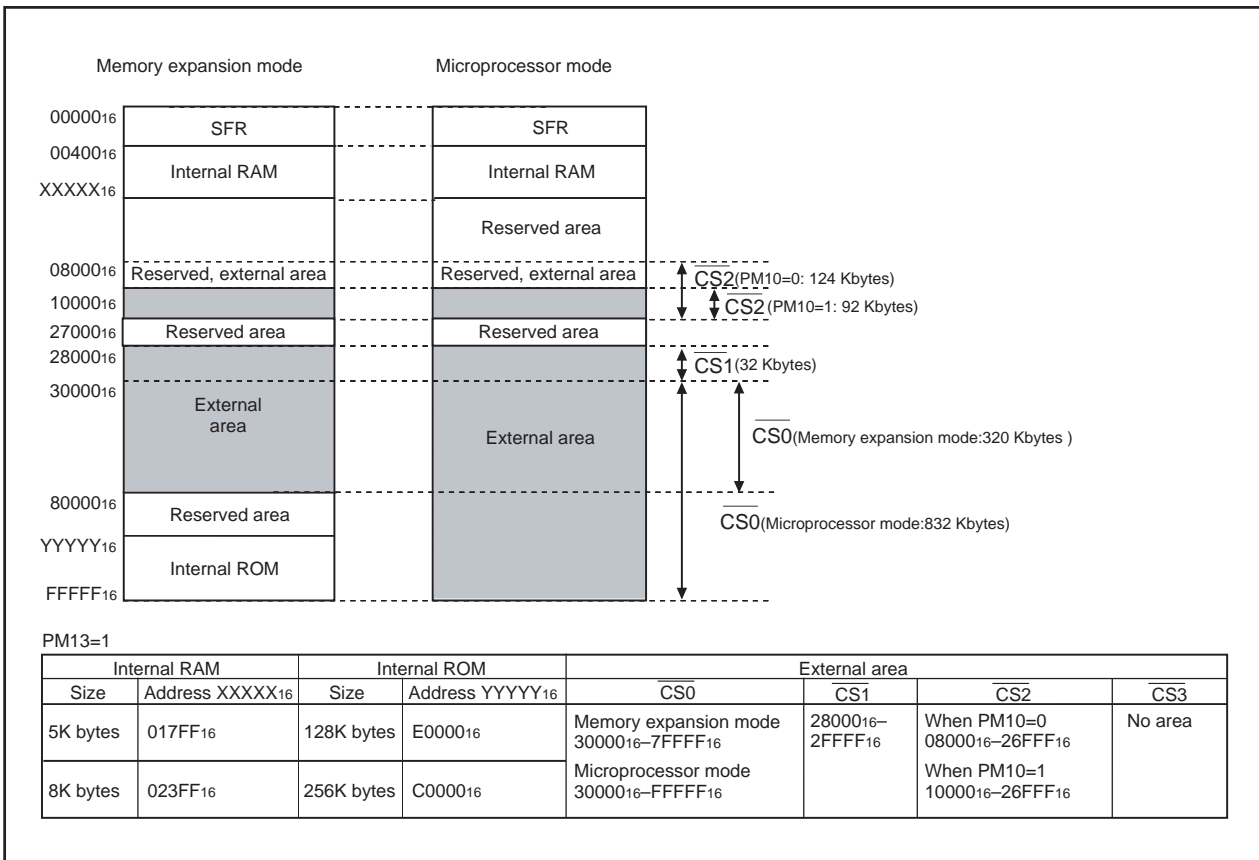


Figure 2.4.7. \overline{CS} Area in 1 Mbyte Mode (PM13=1)

(4) Read and Write Signals

When the data bus is 16 bits wide, the read and write signals can be chosen to be a combination of \overline{RD} , \overline{BHE} and \overline{WR} or a combination of \overline{RD} , \overline{WRL} and \overline{WRH} by using the PM02 bit in the PM0 register. When the data bus is 8 bits wide, use a combination of \overline{RD} , \overline{WR} and \overline{BHE} .

Table 2.4.5 shows the operation of \overline{RD} , \overline{WRL} , and \overline{WRH} signals. Table 2.4.6 shows the operation of operation of \overline{RD} , \overline{WR} , and \overline{BHE} signals.

Table 2.4.5. Operation of \overline{RD} , \overline{WRL} and \overline{WRH} Signals

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

Table 2.4.6. Operation of \overline{RD} , \overline{WR} and \overline{BHE} Signals

Data bus width	\overline{RD}	\overline{WR}	\overline{BHE}	A0	Status of external data bus
16-bit (BYTE pin input = "L")	H	L	L	H	Write 1 byte of data to an odd address
	L	H	L	H	Read 1 byte of data from an odd address
	H	L	H	L	Write 1 byte of data to an even address
	L	H	H	L	Read 1 byte of data from an even address
	H	L	L	L	Write data to both even and odd addresses
	L	H	L	L	Read data from both even and odd addresses
8-bit (BYTE pin input = "H")	H	L	— (Note)	H or L	Write 1 byte of data
	L	H	— (Note)	H or L	Read 1 byte of data

Note : Do not use.

(5) ALE Signal

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

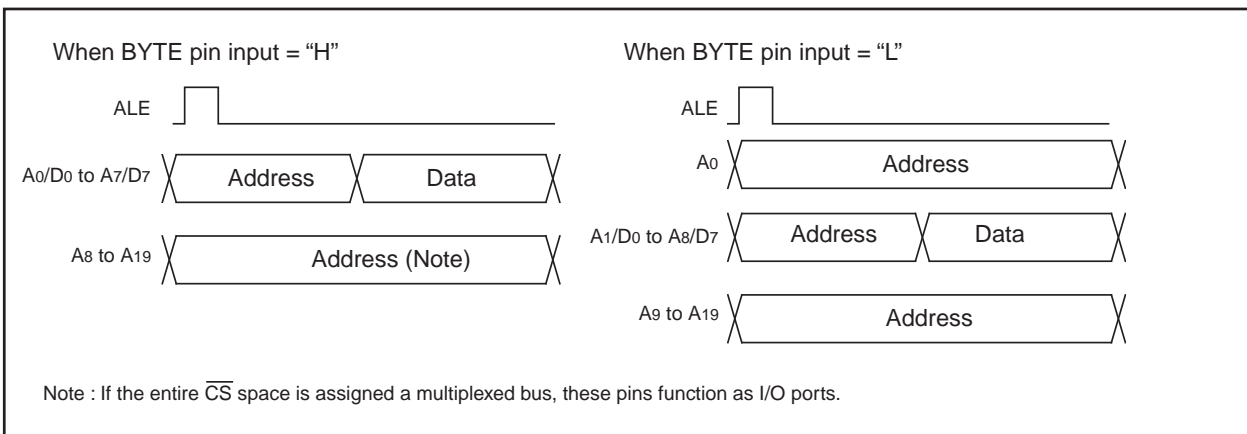


Figure 2.4.8. ALE Signal, Address Bus, Data Bus

(6) The $\overline{\text{RDY}}$ Signal

This signal is provided for accessing external devices which need to be accessed at low speed. If input on the $\overline{\text{RDY}}$ pin is asserted low at the last falling edge of BCLK of the bus cycle, one wait state is inserted in the bus cycle. While in a wait state, the following signals retain the state in which they were when the $\overline{\text{RDY}}$ signal was acknowledged.

A0 to A19, D0 to D15, $\overline{\text{CS}}_0$ to $\overline{\text{CS}}_3$, $\overline{\text{RD}}$, $\overline{\text{WRL}}$, $\overline{\text{WRH}}$, $\overline{\text{WR}}$, $\overline{\text{BHE}}$, $\overline{\text{ALE}}$, $\overline{\text{HLDA}}$

Then, when the input on the $\overline{\text{RDY}}$ pin is detected high at the falling edge of BCLK, the remaining bus cycle is executed. Figure 2.4.9 shows example in which the wait state was inserted into the read cycle by the $\overline{\text{RDY}}$ signal. To use the $\overline{\text{RDY}}$ signal, set the corresponding bit ($\text{CS}3\text{W}$ to $\text{CS}0\text{W}$ bits) in the CSR register to "0" (with wait state). When not using the $\overline{\text{RDY}}$ signal, process the $\overline{\text{RDY}}$ pin as an unused pin.

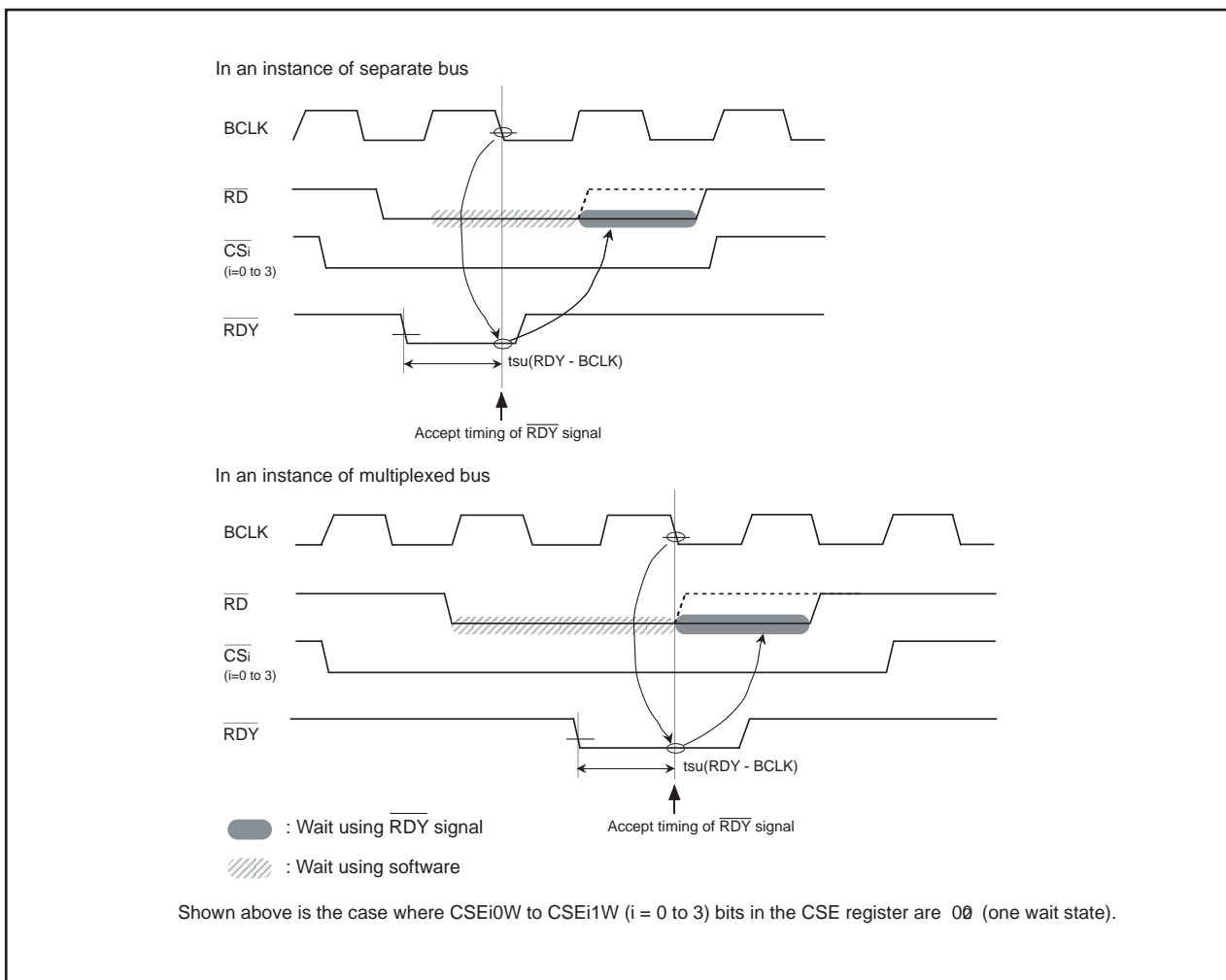


Figure 2.4.9. Example in which Wait State was Inserted into Read Cycle by $\overline{\text{RDY}}$ Signal

(7) Hold Signal

This signal is used to transfer control of the bus from the CPU or DMAC to an external circuit. When the input on $\overline{\text{HOLD}}$ pin is pulled low, the microcomputer is placed in a hold state after the bus access then in process finishes. The microcomputer remains in the hold state while the $\overline{\text{HOLD}}$ pin is held low, during which time the $\overline{\text{HLDA}}$ pin outputs a low-level signal.

Table 2.4.7 shows the microcomputer status in the hold state.

Bus-using priorities are given to $\overline{\text{HOLD}}$, DMAC, and CPU in order of decreasing precedence. However, if the CPU is accessing an odd address in word units, the DMAC cannot gain control of the bus during two separate accesses.

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

Figure 2.4.10. Bus-using Priorities

Table 2.4.7. Microcomputer Status in Hold State

Item	Status	
BCLK	Output	
A ₀ to A ₁₉ , D ₀ to D ₁₅ , $\overline{\text{CS0}}$ to $\overline{\text{CS3}}$, $\overline{\text{RD}}$, $\overline{\text{WRL}}$, $\overline{\text{WRH}}$, $\overline{\text{WR}}$, BHE	High-impedance	
I/O ports	P0, P1, P3, P4(Note 1)	High-impedance
	P6 to P10	Maintains status when hold signal is received
$\overline{\text{HLDA}}$	Output "L"	
Internal peripheral circuits	ON (but watchdog timer stops)	
ALE signal	Undefined	

Note 1: When I/O port function is selected.

(8) BCLK Output

If the PM07 bit in the PM0 register is set to "0" (output enable), a clock with the same frequency as that of the CPU clock is output as BCLK from the BCLK pin. Refer to "CPU clock and peripheral function clock".

Table 2.4.8. Pin Functions for Each Processor Mode

Processor mode	Memory expansion mode or microprocessor mode				Memory expansion mode
PM05–PM04 bits	002(separate bus)		012(CS2 is for multiplexed bus and others are for separate bus) 102(CS1 is for multiplexed bus and others are for separate bus)		112(multiplexed bus for the entire space) (Note 1)
Data bus width BYTE pin	8 bits “H”	16 bits “L”	8 bits “H”	16 bits “L”	8 bits “H”
P00 to P07	D0 to D7	D0 to D7	D0 to D7(Note 4)	D0 to D7(Note 4)	I/O ports
P10 to P17	I/O ports	D8 to D15	I/O ports	D8 to D15(Note 4)	I/O ports
P20	A0	A0	A0/D0(Note 2)	A0	A0/D0
P21 to P27	A1 to A7	A1 to A7	A1 to A7/D1 to D7 (Note 2)	A1 to A7/D0 to D6 (Note 2)	A1 to A7/D1 to D7
P30	A8	A8	A8	A8/D7(Note 2)	A8
P31 to P33	A9 to A11				I/O ports
P34 to P37	PM11=0	A12 to A15			I/O ports
	PM11=1	I/O ports			
P40 to P43	PM06=0	A16 to A19			I/O ports
	PM06=1	I/O ports			
P44	CS0=0	I/O ports			
	CS0=1	$\overline{\text{CS0}}$			
P45	CS1=0	I/O ports			
	CS1=1	$\overline{\text{CS1}}$			
P46	CS2=0	I/O ports			
	CS2=1	$\overline{\text{CS2}}$			
P47	CS3=0	I/O ports			
	CS3=1	$\overline{\text{CS3}}$			
P50	PM02=0	WR			
	PM02=1	— (Note 3)	$\overline{\text{WRL}}$	— (Note 3)	$\overline{\text{WRL}}$
P51	PM02=0	$\overline{\text{BHE}}$			
	PM02=1	— (Note 3)	$\overline{\text{WRH}}$	— (Note 3)	$\overline{\text{WRH}}$
P52	RD				
P53	BCLK				
P54	HLDA				
P55	$\overline{\text{HOLD}}$				
P56	ALE				
P57	$\overline{\text{RDY}}$				

I/O ports: Function as I/O ports or peripheral function I/O pins.

Note 1: To set the PM01 to PM00 bits are set to “012” and the PM05 to PM04 bits are set to “112” (multiplexed bus assigned to the entire $\overline{\text{CS}}$ space), apply “H” to the BYTE pin (external data bus 8 bits wide). While the CNVSS pin and the M1 pin are held “H” (= VCC) in the flash memory version (the CNVSS pin is held “H” in the mask ROM version), do not rewrite the PM05 to PM04 bits to “112” after reset. If the PM05 to PM04 bits are set to “112” during memory expansion mode, P31 to P37 and P40 to P43 become I/O ports, in which case the accessible area for each CS is 256 bytes.

Note 2: In separate bus mode, these pins serve as the address bus.

Note 3: If the data bus is 8 bits wide, make sure the PM02 bit is set to “0” ($\overline{\text{RD}}$, $\overline{\text{BHE}}$, $\overline{\text{WR}}$).

Note 4: When accessing the area that uses a multiplexed bus, these pins output an indeterminate value during a write.

(9) External Bus Status When Internal Area Accessed

Table 2.4.9 shows the external bus status when the internal area is accessed.

Table 2.4.9. External Bus Status When Internal Area Accessed

Item		SFR accessed	Internal ROM, RAM accessed
A0 to A19		Address output	Maintain status before accessed address of external area or SFR
D0 to D15	When read	High-impedance	High-impedance
	When write	Output data	Undefined
\overline{RD} , \overline{WR} , \overline{WRL} , \overline{WRH}		\overline{RD} , \overline{WR} , \overline{WRL} , \overline{WRH} output	Output "H"
\overline{BHE}		\overline{BHE} output	Maintain status before accessed status of external area or SFR
$\overline{CS0}$ to $\overline{CS3}$		Output "H"	Output "H"
ALE		Output "L"	Output "L"

(10) Software Wait

Software wait states can be inserted by using the PM17 bit in the PM1 register, the CS0W to CS3W bits in the CSR register, and the CSE register. The SFR area is unaffected by these control bits. This area is always accessed in 2 BCLK.

To use the \overline{RDY} signal, set the corresponding CS3W to CS0W bit to "0"(with wait state). Figure 2.4.11 shows the CSE register. Table 2.4.10 shows the software wait related bits and bus cycles. Figure 2.4.12 and 2.4.13 show the typical bus timings using software wait.

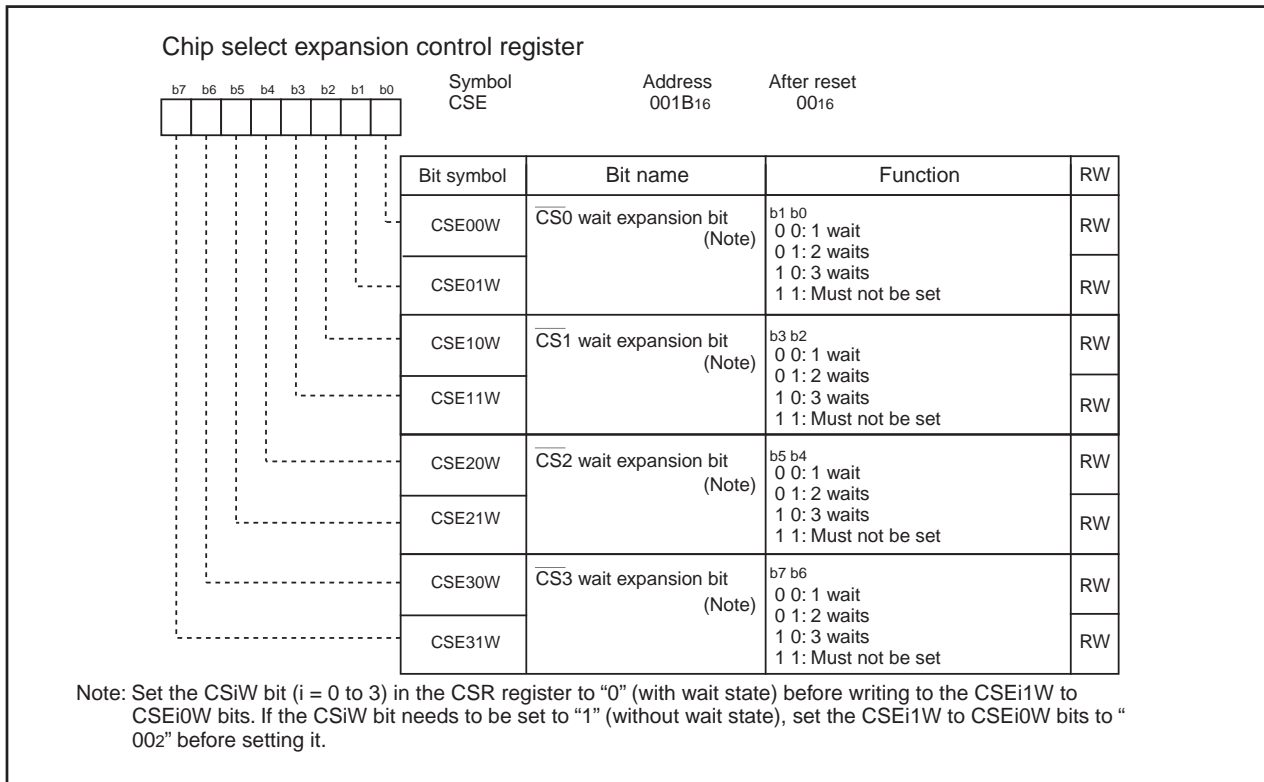


Figure 2.4.11. CSE Register

Table 2.4.10. Bit and Bus Cycle Related to Software Wait

Area	Bus mode	PM1 register PM17 bit	CSR register CS3W bit (Note 1) CS2W bit (Note 1) CS1W bit (Note 1) CS0W bit (Note 1)	CSE register CSE31W to CSE30W bit CSE21W to CSE20W bit CSE11W to CSE10W bit CSE01W to CSE00W bit	Software wait	Bus cycle
SFR	---	---	---	---	---	2 BCLK cycle
Internal RAM, ROM	---	0	---	---	No wait	1 BCLK cycle (Note 3)
	---	1	---	---	1 wait	2 BCLK cycles
External area	Separate bus	0	1	002	No wait	1 BCLK cycle (read) 2 BCLK cycles (write)
		---	0	002	1 wait	2 BCLK cycles (Note 3)
		---	0	012	2 waits	3 BCLK cycles
		---	0	102	3 waits	4 BCLK cycles
		1	1	002	1 wait	2 BCLK cycles
	Multiplexed bus (Note 2)	---	0	002	1 wait	3 BCLK cycles
		---	0	012	2 waits	3 BCLK cycles
		---	0	102	3 waits	4 BCLK cycles
		1	0	002	1 wait	3 BCLK cycles

Note 1: To use the \overline{RDY} signal, set this bit to "0" (with wait state).

Note 2: To access in multiplexed bus mode, set the corresponding bit of CS0W to CS3W to "0" (with wait state).

Note 3: After reset, the PM17 bit is set to "0" (without wait state), all of the CS0W to CS3W bits are set to "0" (with wait state), and the CSE register is set to "0016" (one wait state for CS0 to CS3). Therefore, the internal RAM and internal ROM are accessed with no wait states, and all external areas are accessed with one wait state.

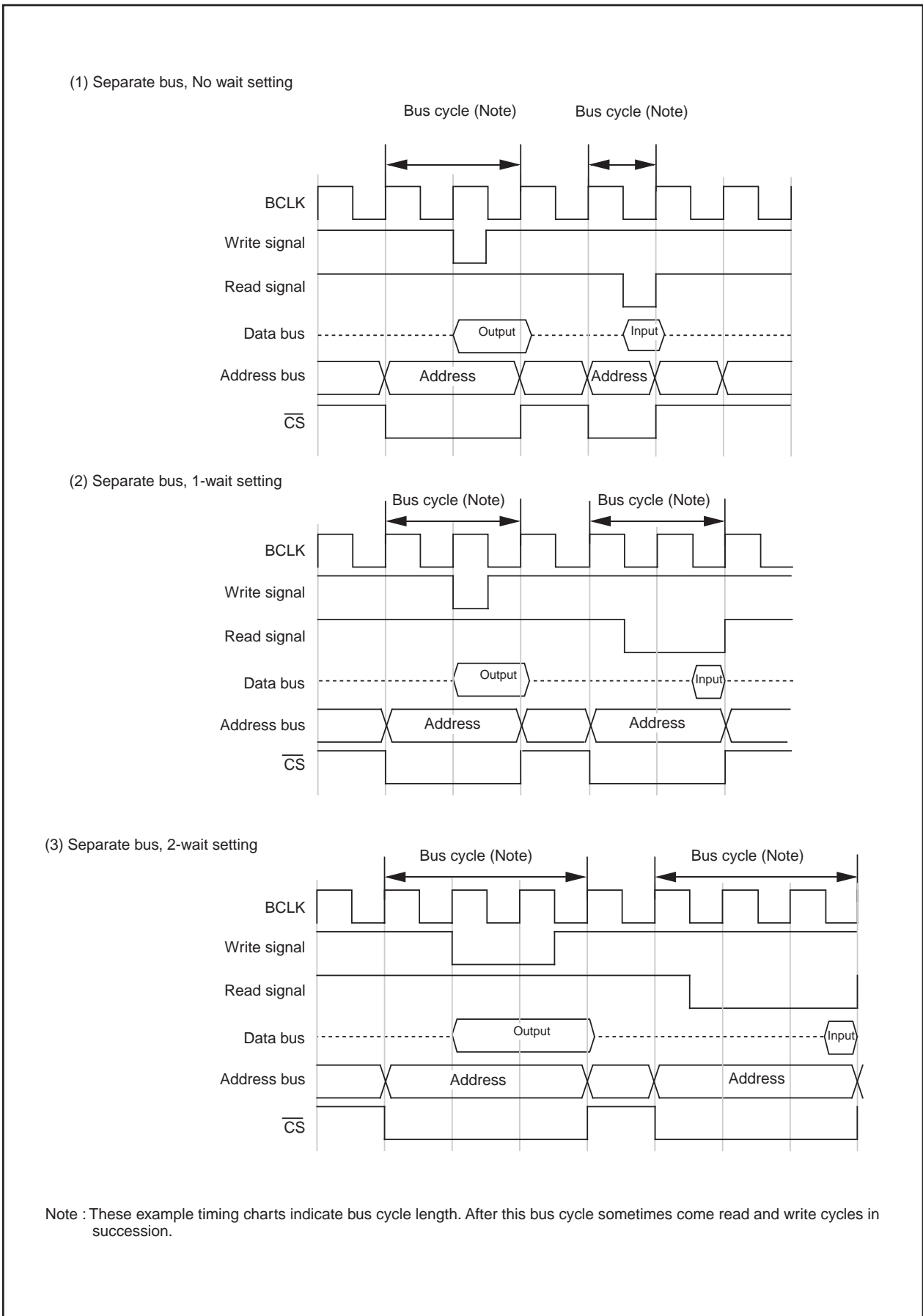


Figure 2.4.12. Typical Bus Timings Using Software Wait (1)

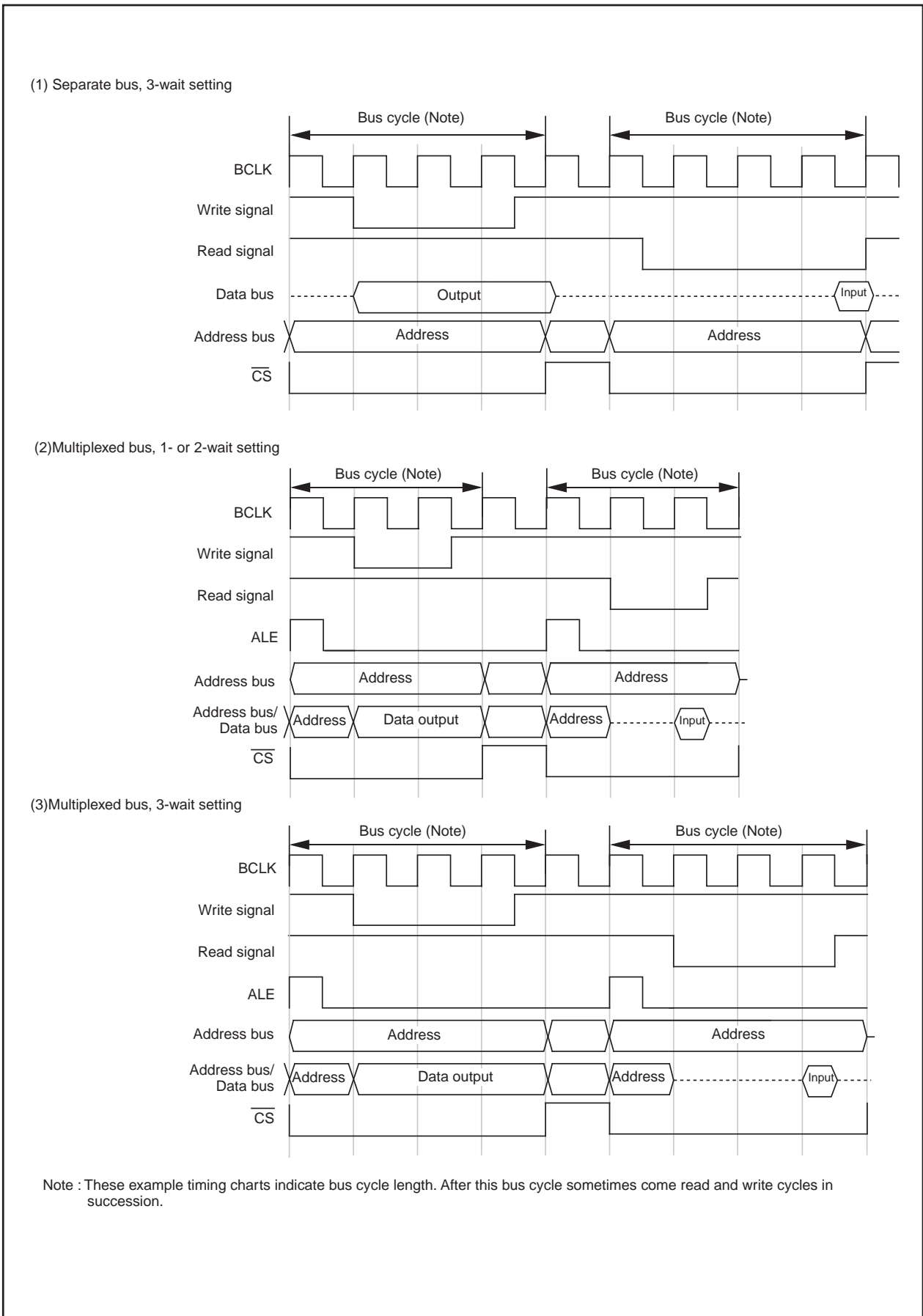


Figure 2.4.13. Typical Bus Timings Using Software Wait (2)

2.5 Clock Generation Circuit

The clock generation circuit contains two oscillator circuits as follows:

- (1) Main clock oscillation circuit
- (2) Sub clock oscillation circuit

Table 2.5.1 lists the clock generation circuit specifications. Figure 2.5.1 shows the clock generation circuit. Figures 2.5.2 to 2.5.4 show the clock-related registers.

Table 2.5.1. Clock Generation Circuit Specifications

Item	Main clock oscillation circuit	Sub clock oscillation circuit
Use of clock	<ul style="list-style-type: none"> • CPU clock source • Peripheral function clock source 	<ul style="list-style-type: none"> • CPU clock source • Timer A, B's clock source
Clock frequency	0 to 16 MHz (Note 3)	32.768 kHz
Usable oscillator	<ul style="list-style-type: none"> • Ceramic oscillator • Crystal oscillator (Note 2) 	<ul style="list-style-type: none"> • Crystal oscillator
Pins to connect oscillator	XIN, XOUT	XCIN, XCOUT
Oscillation stop, restart function	Presence	Presence
Oscillator status after reset (Note1)	Oscillating	Stopped
Other	Externally derived clock can be input	

Note 1. The state that the START pin is held "H" after reset is shown.

The state that the START pin is held "L" after reset is following.

Main clock oscillation circuit: Stopped

Sub clock oscillation circuit: Oscillating

Note 2. If you use "2.14 Expansion Function (Data acquisition)", be sure to connect a crystal oscillator between the XIN and XOUT pins.

Note 3. If you use "2.14 Expansion Function (Data acquisition)", connect a crystal of 10MHz, 12MHz, 14MHz, or 16MHz.

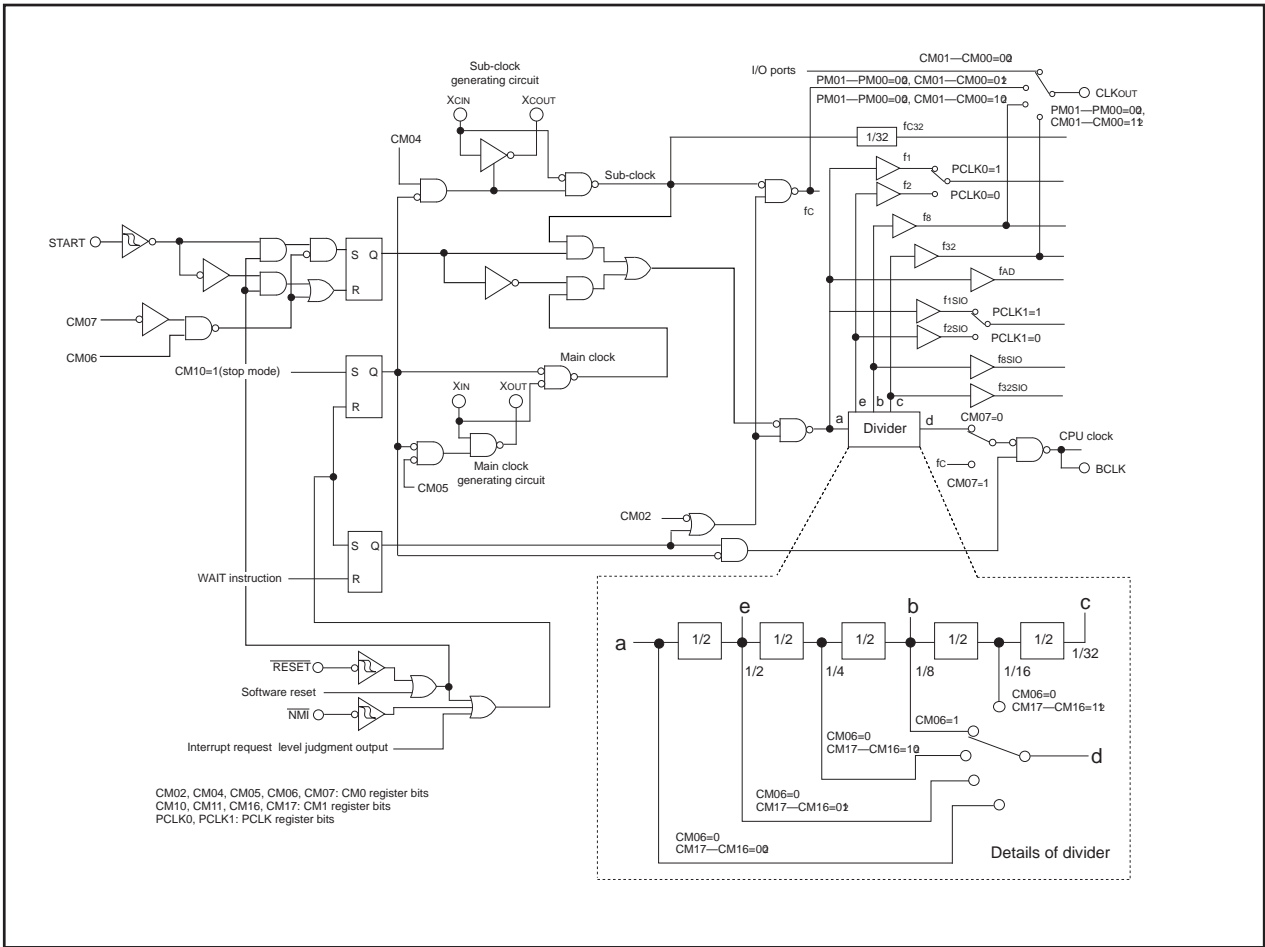


Figure 2.5.1. Clock Generation Circuit

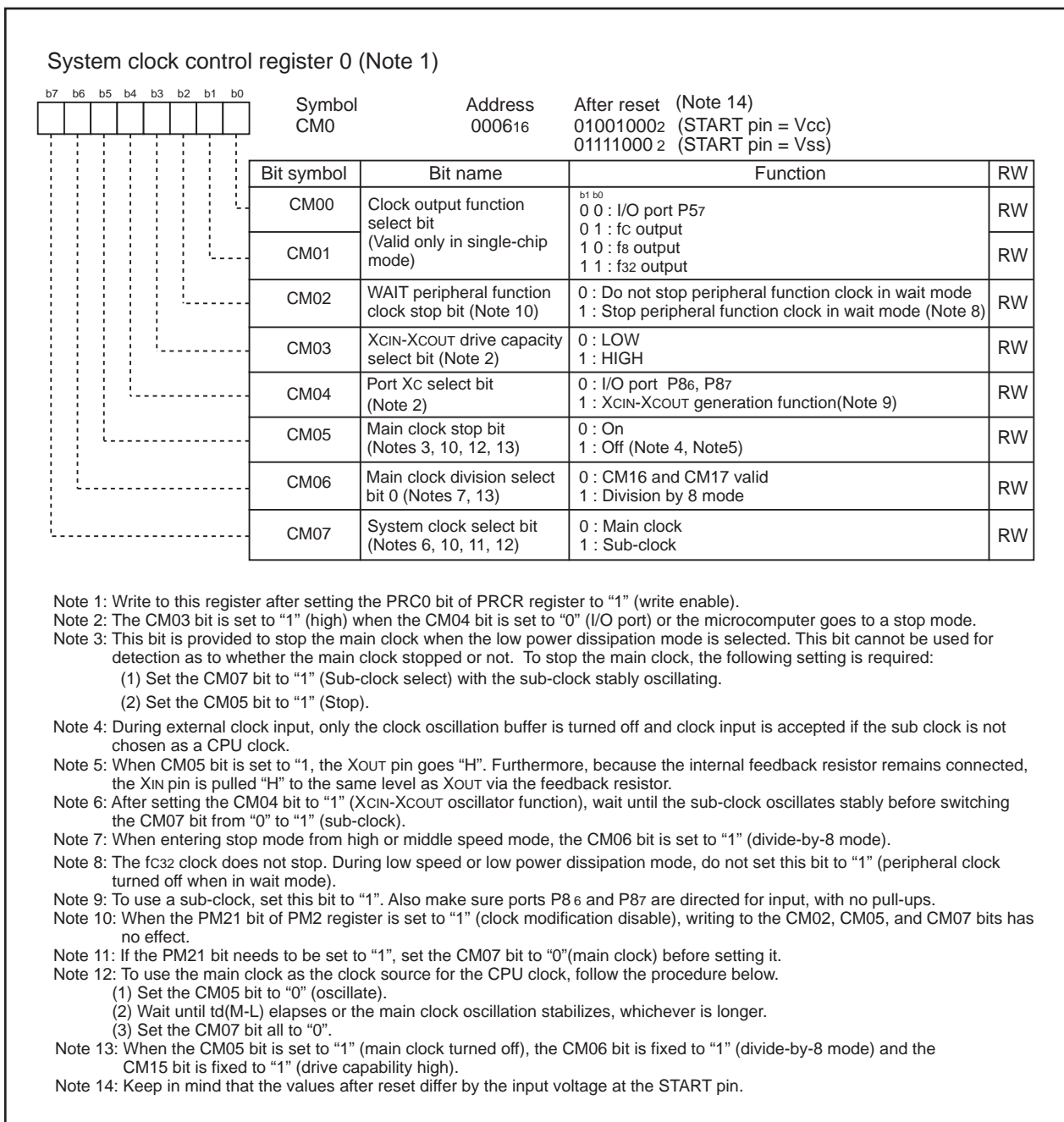


Figure 2.5.2. CM0 Register

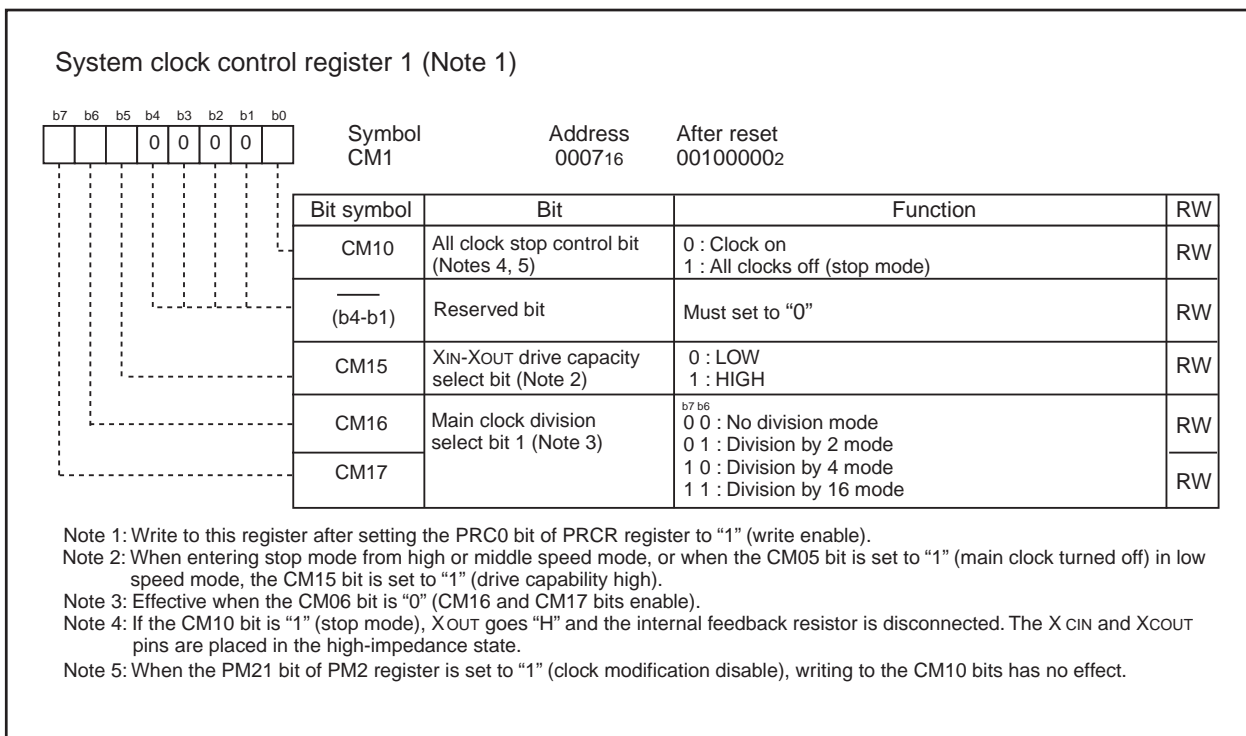
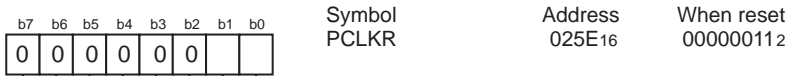


Figure 2.5.3. CM1 Register

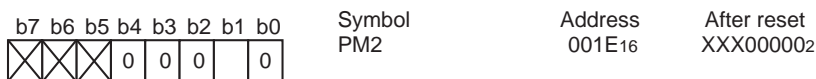
Peripheral clock select register (Note)



Bit symbol	Bit name	Function	RW
PCLK0	Timers A, B clock select bit (Clock source for the timers A and B)	0 : f2 1 : f1	RW
PCLK1	SI/O clock select bit (Clock source for UART0 to UART2, SI/O3, SI/O4)	0 : f2SIO 1 : f1SIO	RW
— (b7-b2)	Reserved bit	Must set to "0"	RW

Note: Write to this register after setting the PRC0 bit of PRCR register to "1" (write enable).

Processor mode register 2 (Note 1)



Bit symbol	Bit name	Function	RW
— (b0)	Reserved bit	Must set to "0"	RW
PM21	System clock protective bit (Note 2, Note 3)	0 : Clock is protected by PRCR register 1 : Clock modification disabled	RW
— (b4-b2)	Reserved bit	Must set to "0"	RW
— (b7-b5)	Nothing is assigned. When write, set to "0". When read, its content is indeterminate.		—

Note 1: Write to this register after setting the PRC1 bit of PRCR register to "1" (write enable).

Note 2: Once this bit is set to "1", it cannot be cleared to "0" in a program.

Note 3: If the PM21 bit is set to "1," writing to the following bits has no effect.

- CM02 bit of CM0 register
- CM05 bit of CM0 register (main clock is not halted)
- CM07 bit of CM0 register (CPU clock source does not change)
- CM10 bit of CM1 register (stop mode is not entered)

Figure 2.5.4. PCLKR Register and PM2 Register

2.5.1 Oscillator Circuit

The following describes the clocks generated by the clock generation circuit.

Two oscillation circuits are built in the clock generating circuit, and a main clock or a sub clock can be chosen as a CPU clock by setup of the START pin after reset.

(1) Main Clock

This clock is used as the clock source for the CPU and peripheral function clocks. The main clock oscillator circuit is configured by connecting a resonator between the XIN and XOUT pins. The main clock oscillator circuit contains a feedback resistor, which is disconnected from the oscillator circuit during stop mode in order to reduce the amount of power consumed in the chip. The main clock oscillator circuit may also be configured by feeding an externally generated clock to the XIN pin. Figure 2.5.5 shows the examples of main clock connection circuit.

When the level on the START pin is "H", the main clock divided by 8 is selected for the CPU clock (Sub clock turned off) after reset.

The power consumption in the chip can be reduced by setting the CM05 bit of CM0 register to "1" (main clock oscillator circuit turned off) after switching the clock source for the CPU clock to a sub clock. In this case, XOUT goes "H". Furthermore, because the internal feedback resistor remains on, XIN is pulled "H" to XOUT via the feedback resistor. Note that if an externally generated clock is fed into the XIN pin, the main clock cannot be turned off by setting the CM05 bit to "1" without selecting sub clock for the CPU clock. If necessary, use an external circuit to turn off the clock.

During stop mode, all clocks including the main clock are turned off. Refer to "power control".

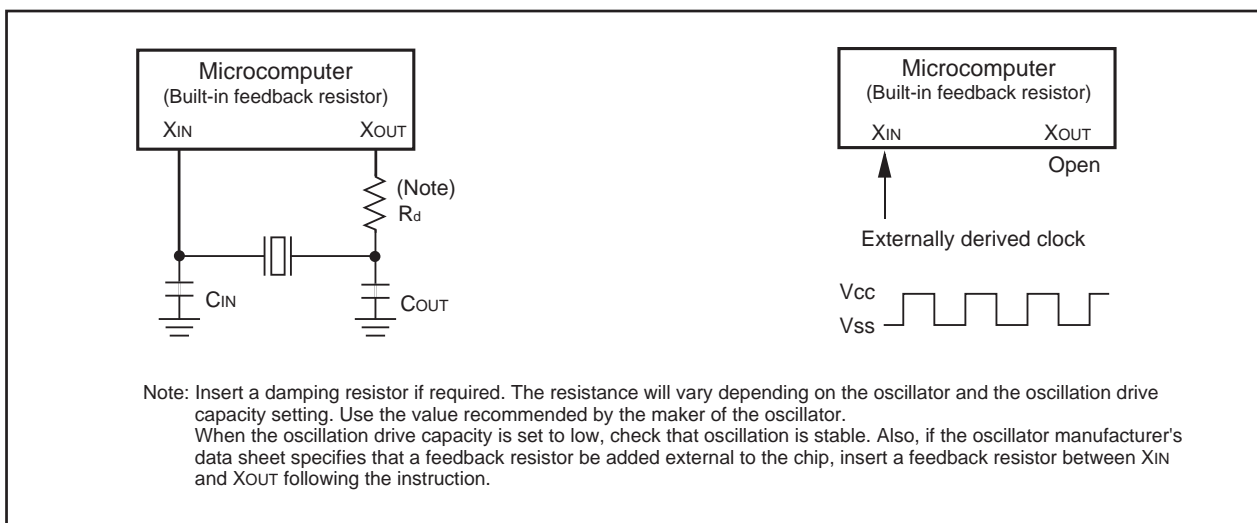


Figure 2.5.5. Examples of Main Clock Connection Circuit

(2) Sub Clock

The sub clock is generated by the sub clock oscillation circuit. This clock is used as the clock source for the CPU clock, as well as the timer A and timer B count sources. In addition, an fc clock with the same frequency as that of the sub clock can be output from the CLKOUT pin.

The sub clock oscillator circuit is configured by connecting a crystal resonator between the XCIN and XCOU pins. The sub clock oscillator circuit contains a feedback resistor, which is disconnected from the oscillator circuit during stop mode in order to reduce the amount of power consumed in the chip. The sub clock oscillator circuit may also be configured by feeding an externally generated clock to the XCIN pin. Figure 2.5.6 shows the examples of sub clock connection circuit.

When the level on the START pin is "H", the sub clock is turned off after reset. At this time, the feedback resistor is disconnected from the oscillator circuit.

To use the sub clock for the CPU clock, set the CM07 bit of CM0 register to "1" (sub clock) after the sub clock becomes oscillating stably.

When a START pin is "L", the sub clock (XCIN) divided by 8 becomes the CPU clock after reset (the main clock stops). When you use a main clock after this, please shift according to the procedure shown in Fig. 2.5.7.

During stop mode, all clocks including the sub clock are turned off. Refer to "power control".

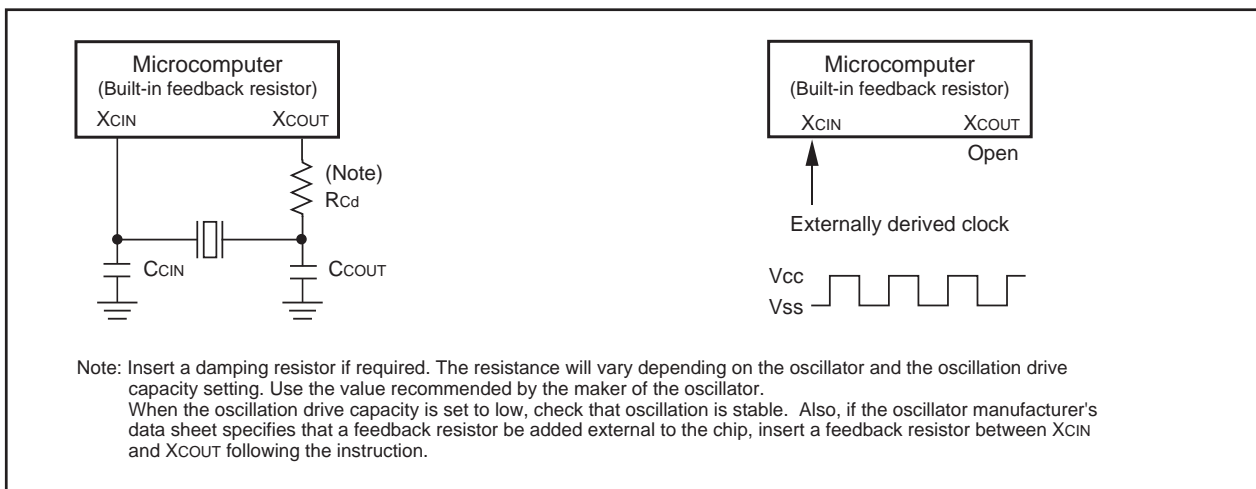


Figure 2.5.6. Examples of Sub Clock Connection Circuit

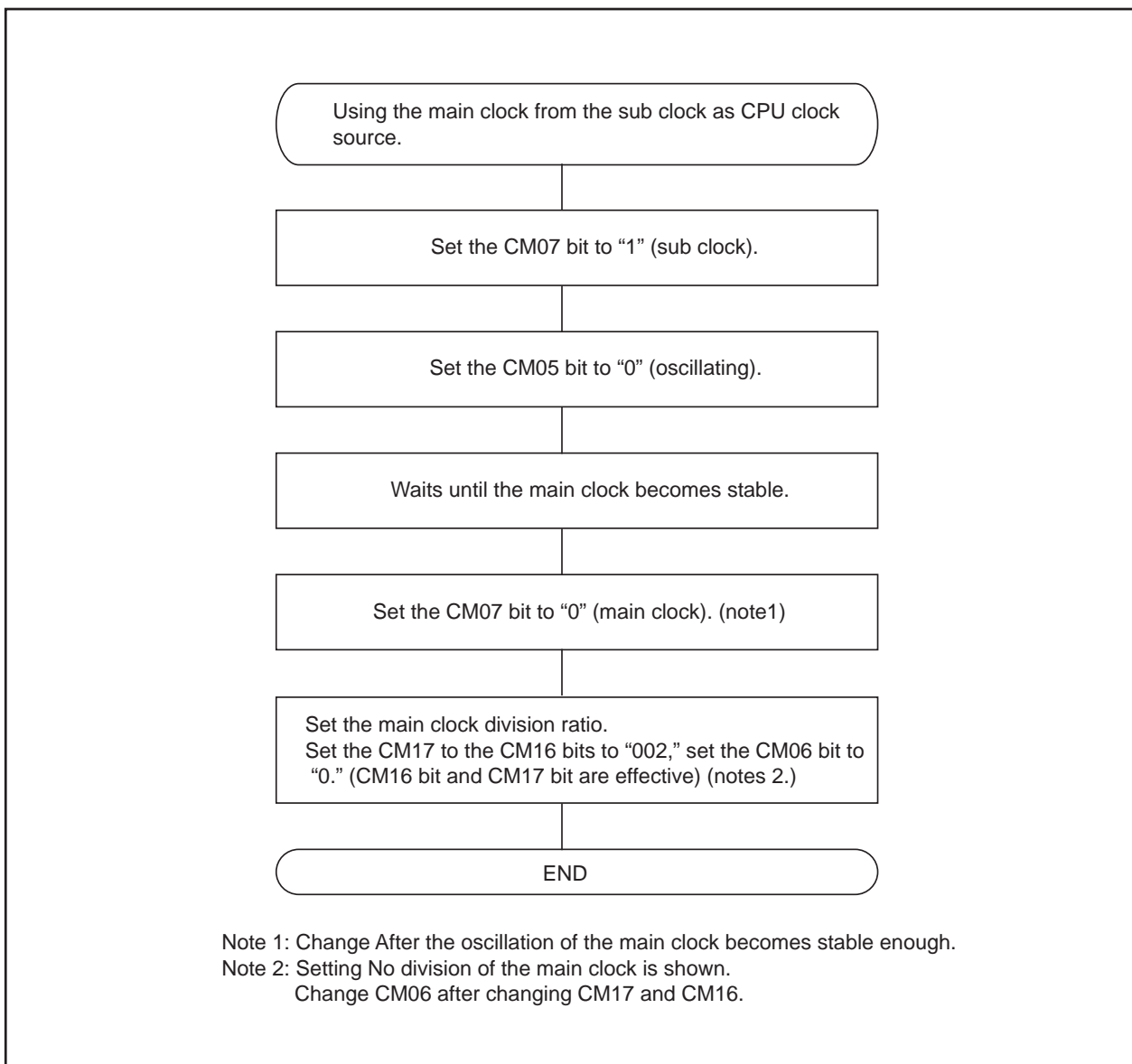


Figure 2.5.7. Procedure to Use the Main Clock from the Sub Clock as CPU Clock Source

2.5.2 CPU Clock and Peripheral Function Clock

Two type clocks: CPU clock to operate the CPU and peripheral function clocks to operate the peripheral functions.

(1) CPU Clock and BCLK

These are operating clocks for the CPU and watchdog timer.

The clock source for the CPU clock can be chosen to be the main clock or sub clock.

If the main clock is selected as the clock source for the CPU clock, the selected clock source can be divided by 1 (undivided), 2, 4, 8 or 16 to produce the CPU clock. Use the CM06 bit in CM0 register and the CM17 to CM16 bits in CM1 register to select the divide-by-n value.

When the level on the START pin is "H", the main clock divided by 8 provides the CPU clock after reset. When the level on the START pin is "L", the sub clock of frequency divided by 8 provides the CPU clock after reset.

At this time, the CM04 bit and the CM05 bit of CM0 register become "1" .

During memory expansion or microprocessor mode, a BCLK signal with the same frequency as the CPU clock can be output from the BCLK pin by setting the PM07 bit of PM0 register to "0" (output enabled).

Note that when entering stop mode from high or middle speed mode, or when the CM05 bit of CM0 register is set to "1" (main clock turned off) in low-speed mode, the CM06 bit of CM0 register is set to "1" (divide-by-8 mode).

(2) Peripheral Function Clock(f_1 , f_2 , f_8 , f_{32} , f_{1SIO} , f_{2SIO} , f_{8SIO} , f_{32SIO} , f_{AD} , f_{C32})

These are operating clocks for the peripheral functions.

Of these, f_i ($i = 1, 2, 8, 32$) and f_{iSIO} are derived from the main clock by dividing them by i . The clock f_i is used for timers A and B, and f_{iSIO} is used for serial I/O. The f_8 and f_{32} clocks can be output from the CLKOUT pin.

The f_{AD} clock is produced from the main clock, and is used for the A-D converter.

When the WAIT instruction is executed after setting the CM02 bit of CM0 register to "1" (peripheral function clock turned off during wait mode), or when the microcomputer is in low power dissipation mode, the f_i , f_{iSIO} and f_{AD} clocks are turned off.

The f_{C32} clock is produced from the sub clock, and is used for timers A and B. This clock can be used when the sub clock is on.

Clock Output Function

During single-chip mode, the f_8 , f_{32} or f_C clock can be output from the CLKOUT pin. Use the CM01 to CM00 bits of CM0 register to select.

2.5.3 Power Control

There are three power control modes. For convenience' sake, all modes other than wait and stop modes are referred to as normal operation mode here.

(1) Normal Operation Mode

Normal operation mode is further classified into four modes.

In normal operation mode, because the CPU clock and the peripheral function clocks both are on, the CPU and the peripheral functions are operating. Power control is exercised by controlling the CPU clock frequency. The higher the CPU clock frequency, the greater the processing capability. The lower the CPU clock frequency, the smaller the power consumption in the chip. If the unnecessary oscillator circuits are turned off, the power consumption is further reduced.

Before the clock sources for the CPU clock can be switched over, the new clock source to which switched must be oscillating stably. If the new clock source is the main clock or sub clock, allow a sufficient wait time in a program until it becomes oscillating stably.

- **High-speed Mode**

The main clock divided by 1 provides the CPU clock. If the sub clock is on, fc32 can be used as the count source for timers A and B.

- **Medium-speed Mode**

The main clock divided by 2, 4, 8 or 16 provides the CPU clock. If the sub clock is on, fc32 can be used as the count source for timers A and B.

- **Low-speed Mode**

The sub clock provides the CPU clock. The main clock is used as the clock source for the peripheral function clock.

The fc32 clock can be used as the count source for timers A and B.

- **Low Power Dissipation Mode**

In this mode, the main clock is turned off after being placed in low speed mode. The sub clock provides the CPU clock. The fc32 clock can be used as the count source for timers A and B.

Simultaneously when this mode is selected, the CM06 bit of CM0 register becomes "1" (divided by 8 mode). In the low power dissipation mode, do not change the CM06 bit. Consequently, the medium speed (divided by 8) mode is to be selected when the main clock is operated next.

(2) Wait Mode

In wait mode, the CPU clock is turned off, so are the CPU (because operated by the CPU clock) and the watchdog timer. Because the main clock and sub clock, are on, the peripheral functions using these clocks keep operating.

- **Peripheral Function Clock Stop Function**

If the CM02 bit is "1" (peripheral function clocks turned off during wait mode), the f1, f2, f8, f32, f1SIO, f8SIO, f32SIO and fAD clocks are turned off when in wait mode, with the power consumption reduced that much. However, fC32 remains on.

- **Entering Wait Mode**

The microcomputer is placed into wait mode by executing the WAIT instruction.

- **Pin Status During Wait Mode**

Table 2.5.2 lists pin status during wait mode

- **Exiting Wait Mode**

The microcomputer is moved out of wait mode by a hardware reset, $\overline{\text{NMI}}$ interrupt or peripheral function interrupt.

If the microcomputer is to be moved out of exit wait mode by a hardware reset or $\overline{\text{NMI}}$ interrupt, set the peripheral function interrupt priority ILVL2 to ILVL0 bits to "0002" (interrupts disabled) before executing the WAIT instruction.

The peripheral function interrupts are affected by the CM02 bit. If CM02 bit is "0" (peripheral function clocks not turned off during wait mode), all peripheral function interrupts can be used to exit wait mode. If CM02 bit is "1" (peripheral function clocks turned off during wait mode), the peripheral functions using the peripheral function clocks stop operating, so that only the peripheral functions clocked by external signals can be used to exit wait mode.

Table 2.5.2. Pin Status During Wait Mode

Pin		Memory expansion mode Microprocessor mode	Single-chip mode
A ₀ to A ₁₉ , D ₀ to D ₁₅ , \overline{CS}_0 to \overline{CS}_3 , \overline{BHE}		Retains status before wait mode	/
RD, WR, WRL, WRH		"H"	
HLDA, BCLK		"H"	
ALE		"H"	
I/O ports		Retains status before wait mode	
CLKOUT	When f _c selected		Does not stop
	When f ₈ , f ₃₂ selected		Does not stop when the CM02 bit is "0". When the CM02 bit is "1", the status immediately prior to entering wait mode is maintained.

Table 2.5.3. Interrupts to Exit Wait Mode

Interrupt	CM02=0	CM02=1
NMI interrupt	Can be used	Can be used
Serial I/O interrupt	Can be used when operating with internal or external clock	Can be used when operating with external clock
key input interrupt	Can be used	Can be used
A-D conversion interrupt	Can be used in one-shot mode or single sweep mode	— (Do not use)
Timer A interrupt Timer B interrupt	Can be used in all modes	Can be used in event counter mode or when the count source is f _{C32}
INT interrupt	Can be used	Can be used

Table 2.5.3 lists the interrupts to exit wait mode.

If the microcomputer is to be moved out of wait mode by a peripheral function interrupt, set up the following before executing the WAIT instruction.

1. In the ILVL2 to ILVL0 bits of interrupt control register, set the interrupt priority level of the peripheral function interrupt to be used to exit wait mode.

Also, for all of the peripheral function interrupts not used to exit wait mode, set the ILVL2 to ILVL0 bits to "0002" (interrupt disable).

2. Set the I flag to "1".
3. Enable the peripheral function whose interrupt is to be used to exit wait mode.

In this case, when an interrupt request is generated and the CPU clock is thereby turned on, an interrupt routine is executed.

The CPU clock turned on when exiting wait mode by a peripheral function interrupt is the same CPU clock that was on when the WAIT instruction was executed.

(3) Stop Mode

In stop mode, all oscillator circuits are turned off, so are the CPU clock and the peripheral function clocks. Therefore, the CPU and the peripheral functions clocked by these clocks stop operating. The least amount of power is consumed in this mode. If the voltage applied to Vcc pins is V_{RAM} or more, the internal RAM is retained.

However, the peripheral functions clocked by external signals keep operating. The following interrupts can be used to exit stop mode.

- $\overline{\text{NMI}}$ interrupt
- Key interrupt
- $\overline{\text{INT}}$ interrupt
- Timer A, Timer B interrupt (when counting external pulses in event counter mode)
- Serial I/O interrupt (when external clock is selected)

The internal oscillator circuit of expansion function (Data acquisition / humming function) stops oscillation when expansion register XTAL_VCO, PDC_VCO_ON, VPS_VCO_ON = "L".

• Entering Stop Mode

The microcomputer is placed into stop mode by setting the CM10 bit of CM1 register to "1" (all clocks turned off). At the same time, the CM06 bit of CM0 register is set to "1" (divide-by-8 mode) and the CM15 bit of CM1 register is set to "1" (main clock oscillator circuit drive capability high).

• Pin Status in Stop Mode

Table 2.5.4 lists pin status during stop mode

• Exiting Stop Mode

The microcomputer is moved out of stop mode by a hardware reset, $\overline{\text{NMI}}$ interrupt or peripheral function interrupt.

If the microcomputer is to be moved out of stop mode by a hardware reset or $\overline{\text{NMI}}$ interrupt, set the peripheral function interrupt priority ILVL2 to ILVL0 bits to "0002" (interrupts disable) before setting the CM10 bit to "1".

If the microcomputer is to be moved out of stop mode by a peripheral function interrupt, set up the following before setting the CM10 bit to "1".

1. In the ILVL2 to ILVL0 bits of interrupt control register, set the interrupt priority level of the peripheral function interrupt to be used to exit stop mode.

Also, for all of the peripheral function interrupts not used to exit stop mode, set the ILVL2 to ILVL0 bits to "0002".

2. Set the I flag to "1".

3. Enable the peripheral function whose interrupt is to be used to exit stop mode.

In this case, when an interrupt request is generated and the CPU clock is thereby turned on, an interrupt service routine is executed.

Which CPU clock will be used after exiting stop mode by a peripheral function or $\overline{\text{NMI}}$ interrupt is determined by the CPU clock that was on when the microcomputer was placed into stop mode as follows:

If the CPU clock before entering stop mode was derived from the sub clock: sub clock

If the CPU clock before entering stop mode was derived from the main clock: main clock divide-by-8

Table 2.5.4. Pin Status in Stop Mode

Pin		Memory expansion mode Microprocessor mode	Single-chip mode
A ₀ to A ₁₉ , D ₀ to D ₁₅ , $\overline{\text{CS}}_0$ to $\overline{\text{CS}}_3$, $\overline{\text{BHE}}$		Retains status before stop mode	
RD, WR, $\overline{\text{WRL}}$, $\overline{\text{WRH}}$		"H"	
$\overline{\text{HLDA}}$, BCLK		"H"	
ALE		"H"	
I/O ports		Retains status before stop mode	Retains status before stop mode
CLKOUT	When fc selected		"H"
	When f8, f32 selected		Retains status before stop mode

Figure 2.5.8 shows the state transition from normal operation mode to stop mode and wait mode. Figure 2.5.9 shows the state transition in normal operation mode.

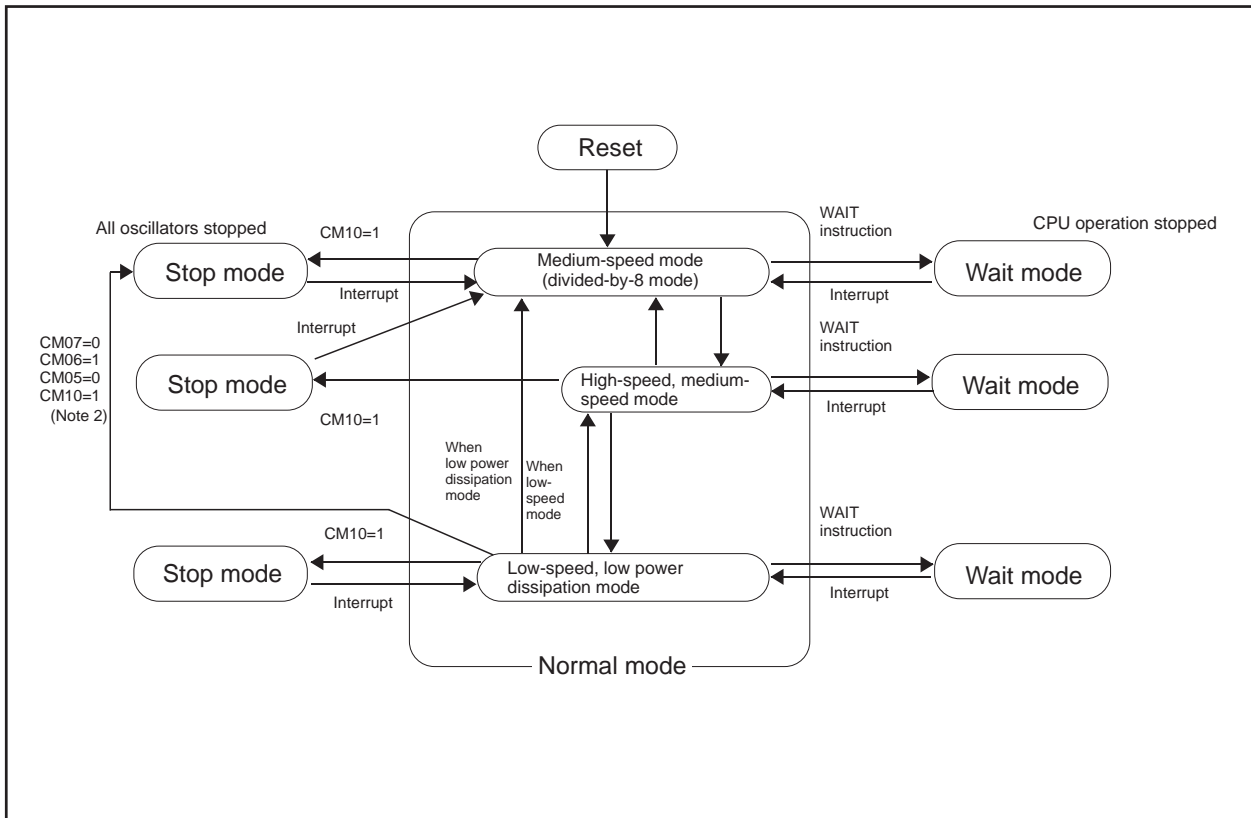


Figure 2.5.8. State Transition to Stop Mode and Wait Mode

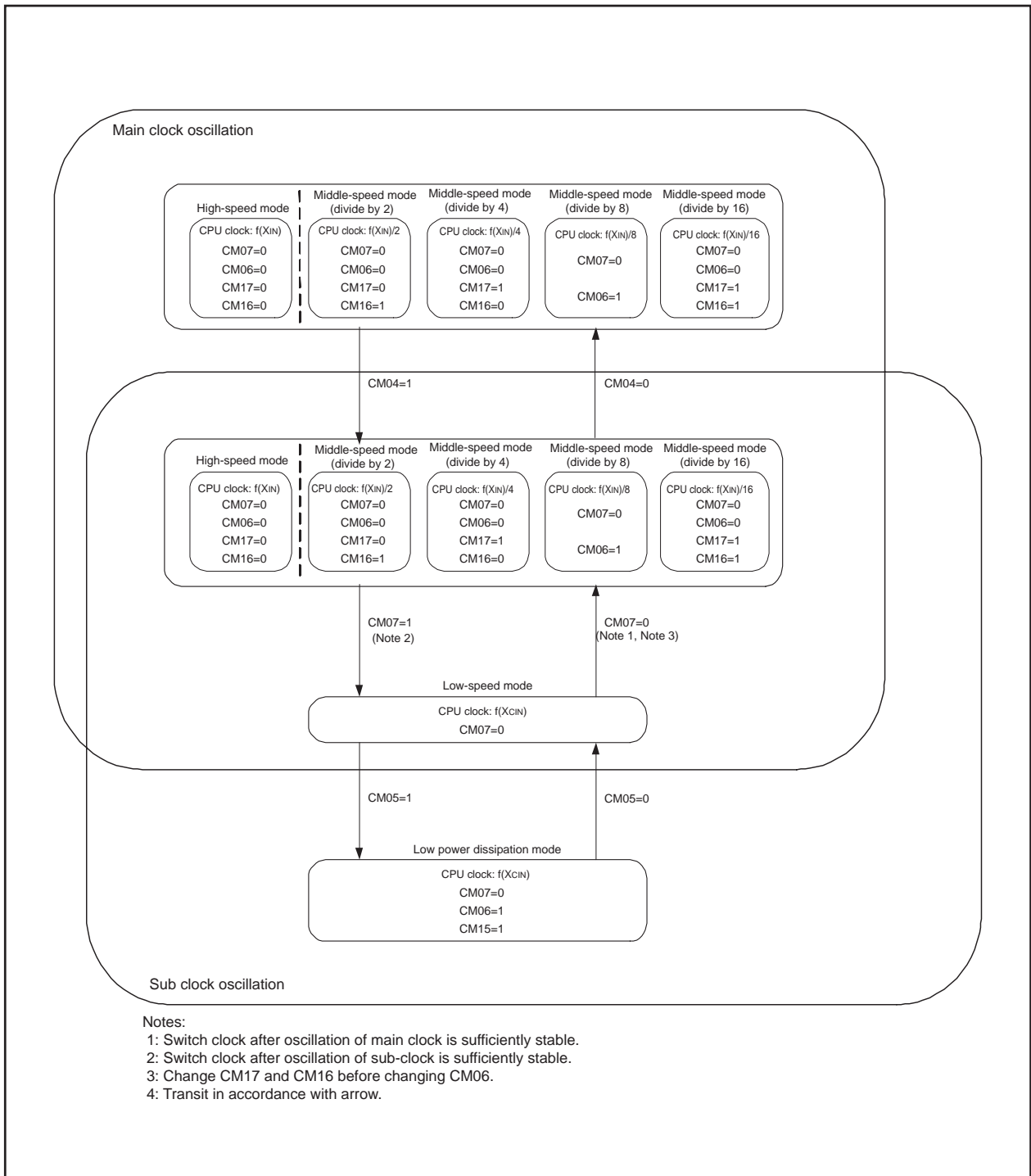


Figure 2.5.9. State Transition in Normal Mode

2.5.4 System Clock Protective Function

When the main clock is selected for the CPU clock source, this function disables the clock against modifications in order to prevent the CPU clock from becoming halted by run-away.

If the PM21 bit of PM2 register is set to "1" (clock modification disabled), the following bits are protected against writes:

- CM02, CM05, and CM07 bits in CM0 register
- CM10, CM11 bits in CM1 register

Before the system clock protective function can be used, the following register settings must be made while the CM05 bit of CM0 register is "0" (main clock oscillating) and CM07 bit is "0" (main clock selected for the CPU clock source):

- (1) Set the PRC1 bit of PRCR register to "1" (enable writes to PM2 register).
- (2) Set the PM21 bit of PM2 register to "1" (disable clock modification).
- (3) Set the PRC1 bit of PRCR register to "0" (disable writes to PM2 register).

Do not execute the WAIT instruction when the PM21 bit is "1".

2.6 Protection

In the event that a program runs out of control, this function protects the important registers so that they will not be rewritten easily. Figure 2.6.1 shows the PRCR register. The following lists the registers protected by the PRCR register.

- Registers protected by PRC0 bit: CM0, CM1 and PCLKR registers
- Registers protected by PRC1 bit: PM0, PM1 and PM2 registers
- Registers protected by PRC2 bit: PD9, S3C and S4C registers

Set the PRC2 bit to “1” (write enabled) and then write to any address, and the PRC2 bit will be cleared to “0” (write protected). The registers protected by the PRC2 bit should be changed in the next instruction after setting the PRC2 bit to “1”. Make sure no interrupts or DMA transfers will occur between the instruction in which the PRC2 bit is set to “1” and the next instruction. The PRC0 and PRC1 bits are not automatically cleared to “0” by writing to any address. They can only be cleared in a program.

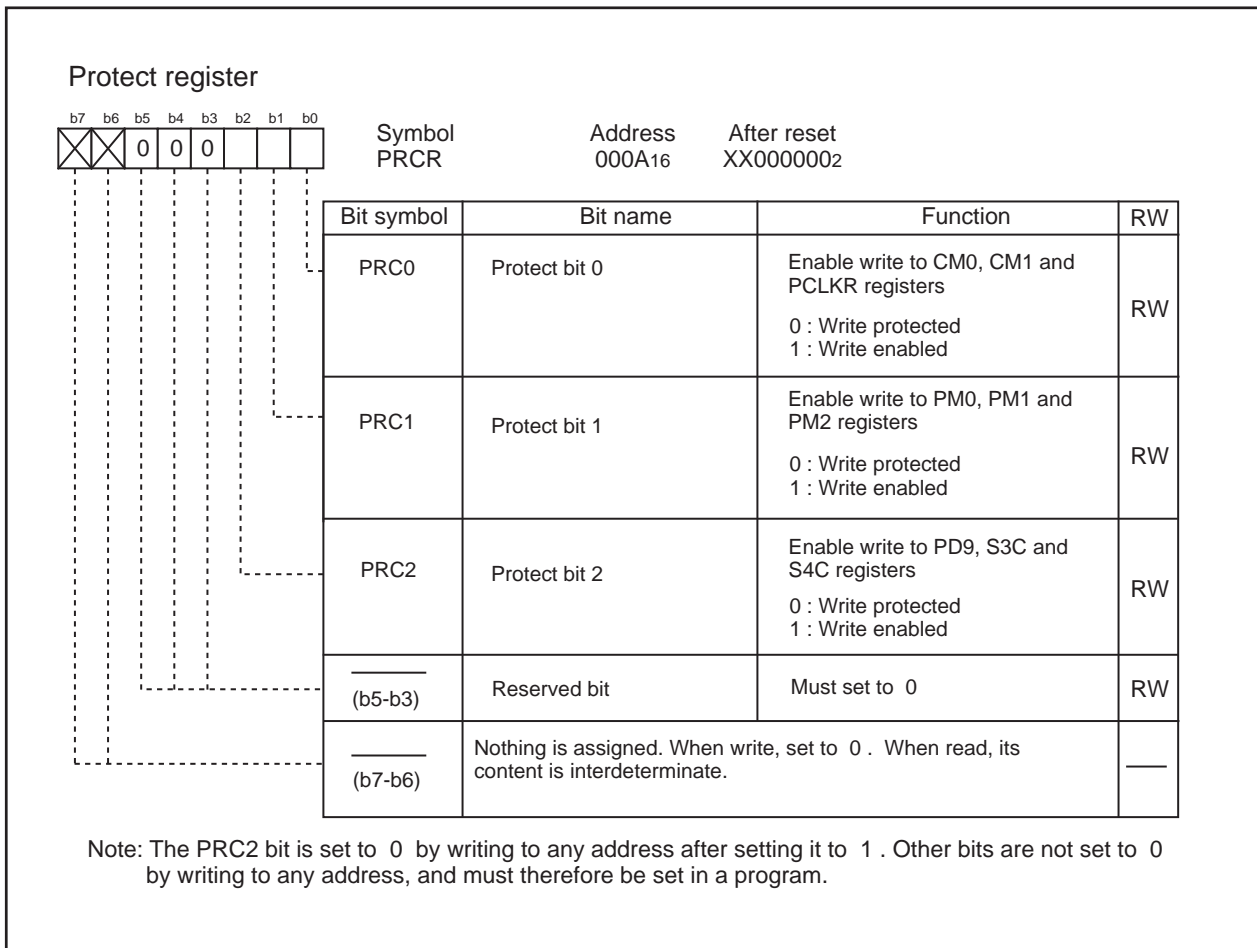


Figure 2.6.1. PRCR Register

2.7 Interrupts

2.7.1 Type of Interrupts

Figure 2.7.1 shows types of interrupts.

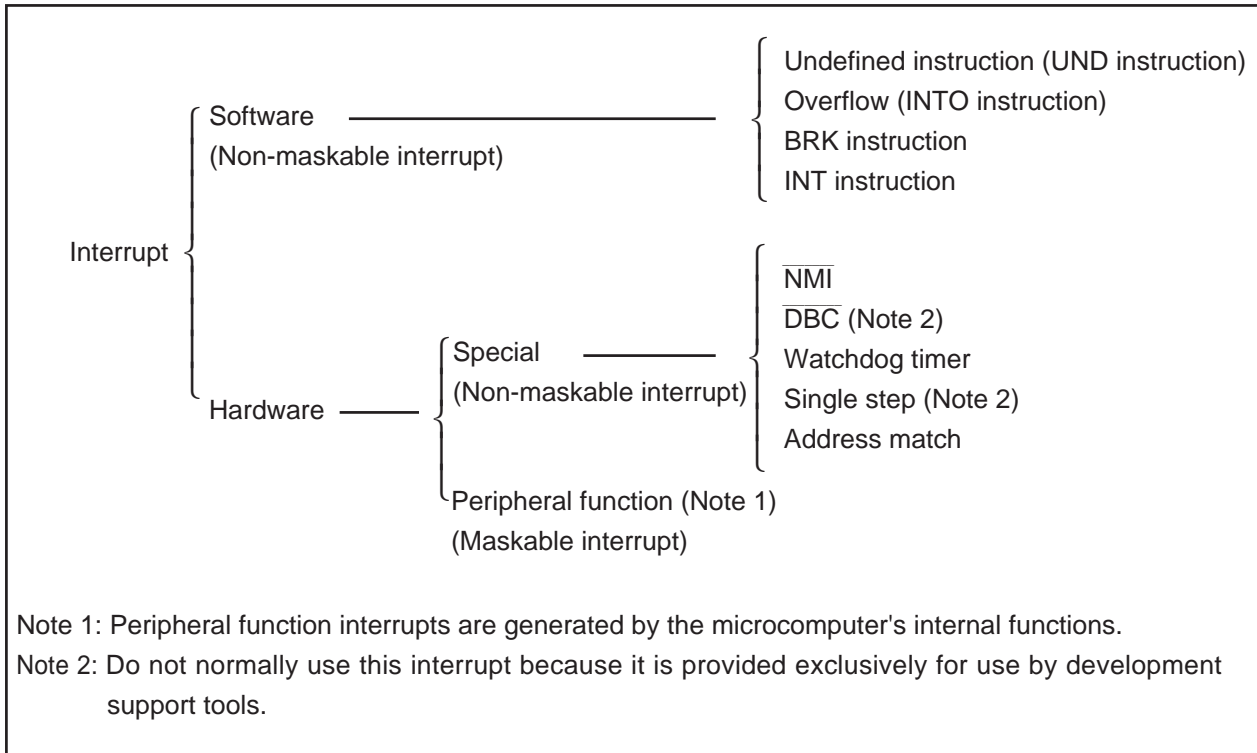


Figure 2.7.1. Interrupts

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

2.7.2 Software Interrupts

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

- **Undefined Instruction Interrupt**

An undefined instruction interrupt occurs when executing the UND instruction.

- **Overflow Interrupt**

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

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

- **BRK Interrupt**

A BRK interrupt occurs when executing the BRK instruction.

- **INT Instruction Interrupt**

An INT instruction interrupt occurs when executing the INT instruction. Software interrupt Nos. 0 to 63 can be specified for the INT instruction. Because software interrupt Nos. 4 to 31 are assigned to peripheral function interrupts, the same interrupt routine as for peripheral function interrupts can be executed by executing the INT instruction.

In software interrupt Nos. 0 to 31, the U flag is saved to the stack during instruction execution and is cleared to "0" (ISP selected) before executing an interrupt sequence. The U flag is restored from the stack when returning from the interrupt routine. In software interrupt Nos. 32 to 63, the U flag does not change state during instruction execution, and the SP then selected is used.

2.7.3 Hardware Interrupts

Hardware interrupts are classified into two types — special interrupts and peripheral function interrupts.

(1) Special Interrupts

Special interrupts are non-maskable interrupts.

- **NMI Interrupt**

An $\overline{\text{NMI}}$ interrupt is generated when input on the $\overline{\text{NMI}}$ pin changes state from high to low. For details about the $\overline{\text{NMI}}$ interrupt, refer to the section "NMI interrupt".

- **DBC Interrupt**

Do not normally use this interrupt because it is provided exclusively for use by development support tools.

- **Watchdog Timer Interrupt**

Generated by the watchdog timer. Once a watchdog timer interrupt is generated, be sure to initialize the watchdog timer. For details about the watchdog timer, refer to the section "watchdog timer".

- **Single-step Interrupt**

Do not normally use this interrupt because it is provided exclusively for use by development support tools.

- **Address Match Interrupt**

An address match interrupt is generated immediately before executing the instruction at the address indicated by the RMAD0 to RMAD3 register that corresponds to one of the AIER register's AIER0 or AIER1 bit or the AIER2 register's AIER20 or AIER21 bit which is "1" (address match interrupt enabled). For details about the address match interrupt, refer to the section "address match interrupt".

(2) Peripheral Function Interrupts

Peripheral function interrupts are maskable interrupts and generated by the microcomputer's internal functions. The interrupt sources for peripheral function interrupts are listed in Table 2.7.2. For details about the peripheral functions, refer to the description of each peripheral function in this manual.

2.7.4 Interrupts and Interrupt Vector

One interrupt vector consists of 4 bytes. Set the start address of each interrupt routine in the respective interrupt vectors. When an interrupt request is accepted, the CPU branches to the address set in the corresponding interrupt vector. Figure 2.7.2 shows the interrupt vector.

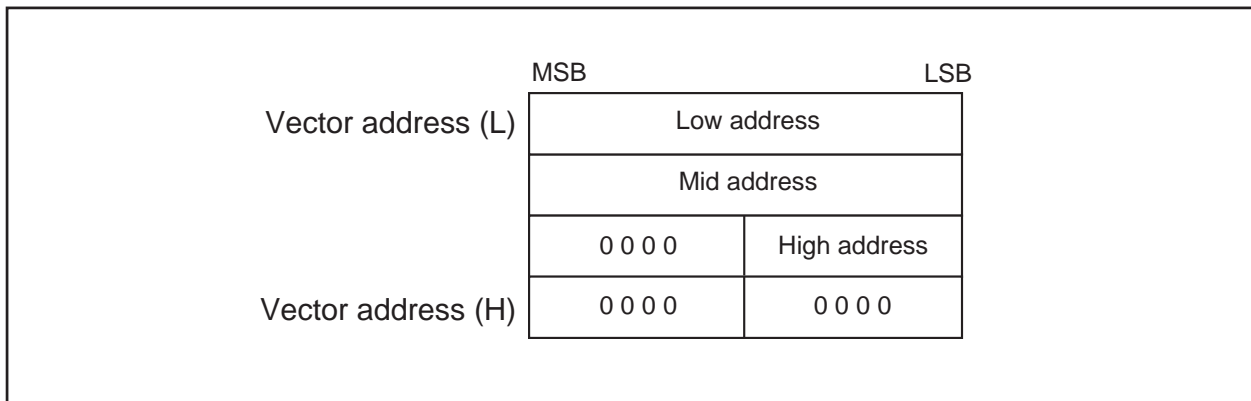


Figure 2.7.2. Interrupt Vector

• Fixed Vector Tables

The fixed vector tables are allocated to the addresses from FFFDC₁₆ to FFFFF₁₆. Table 2.7.1 lists the fixed vector tables. In the flash memory version of microcomputer, the vector addresses (H) of fixed vectors are used by the ID code check function. For details, refer to the section "flash memory rewrite disabling function".

Table 2.7.1. Fixed Vector Tables

Interrupt source	Vector table addresses Address (L) to address (H)	Remarks	Reference
Undefined instruction	FFFD _{C16} to FFFD _{F16}	Interrupt on UND instruction	M16C/60, M16C/20 series software manual
Overflow	FFFE ₀₁₆ to FFFE ₃₁₆	Interrupt on INTO instruction	
BRK instruction	FFFE ₄₁₆ to FFFE ₇₁₆	If the contents of address FFFE ₇₁₆ is FF ₁₆ , program execution starts from the address shown by the vector in the relocatable vector table.	
Address match	FFFE ₈₁₆ to FFFE _{B16}		Address match interrupt
Single step (Note)	FFFE _{C16} to FFFE _{F16}		
Watchdog timer	FFFF ₀₁₆ to FFFF ₃₁₆		Watchdog timer
$\overline{\text{DBC}}$ (Note)	FFFF ₄₁₆ to FFFF ₇₁₆		
NMI	FFFF ₈₁₆ to FFFF _{B16}		NMI interrupt
Reset	FFFF _{C16} to FFFF _{F16}		Reset

Note: Do not normally use this interrupt because it is provided exclusively for use by development support tools.

• Relocatable Vector Tables

The 256 bytes beginning with the start address set in the INTB register comprise a relocatable vector table area. Table 2.7.2 lists the relocatable vector tables. Setting an even address in the INTB register results in the interrupt sequence being executed faster than in the case of odd addresses.

Table 2.7.2. Relocatable Vector Tables

Interrupt source	Vector address (Note 1) Address (L) to address (H)	Software interrupt number	Reference
BRK instruction (Note 5)	+0 to +3 (0000 ₁₆ to 0003 ₁₆)	0	M16C/60, M16C/20 series software manual
———— (Reserved)		1 to 3	
$\overline{\text{INT}}_3$	+16 to +19 (0010 ₁₆ to 0013 ₁₆)	4	INT interrupt
Timer B5/SLICE ON (Note 7)	+20 to +23 (0014 ₁₆ to 0017 ₁₆)	5	Timer
Timer B4/Remote control, UART1 bus collision detect (Note 4/Note 6/Note 7)	+24 to +27 (0018 ₁₆ to 001B ₁₆)	6	Timer Serial I/O
Timer B3/HINT, UART0 bus collision detect (Note 4/Note 6/Note 7)	+28 to +31 (001C ₁₆ to 001F ₁₆)	7	
SI/O4, $\overline{\text{INT}}_5$ (Note 2)	+32 to +35 (0020 ₁₆ to 0023 ₁₆)	8	$\overline{\text{INT}}$ interrupt Serial I/O
SI/O3, $\overline{\text{INT}}_4$ (Note 2)	+36 to +39 (0024 ₁₆ to 0027 ₁₆)	9	
UART 2 bus collision detection	+40 to +43 (0028 ₁₆ to 002B ₁₆)	10	Serial I/O
DMA0	+44 to +47 (002C ₁₆ to 002F ₁₆)	11	DMAC
DMA1	+48 to +51 (0030 ₁₆ to 0033 ₁₆)	12	
Key input interrupt	+52 to +55 (0034 ₁₆ to 0037 ₁₆)	13	Key input interrupt
A-D	+56 to +59 (0038 ₁₆ to 003B ₁₆)	14	A-D convertor
UART2 transmit, NACK2 (Note 3)	+60 to +63 (003C ₁₆ to 003F ₁₆)	15	Serial I/O
UART2 receive, ACK2 (Note 3)	+64 to +67 (0040 ₁₆ to 0043 ₁₆)	16	
UART0 transmit, NACK0 (Note 3)	+68 to +71 (0044 ₁₆ to 0047 ₁₆)	17	
UART0 receive, ACK0 (Note 3)	+72 to +75 (0048 ₁₆ to 004B ₁₆)	18	
UART1 transmit, NACK1 (Note 3)	+76 to +79 (004C ₁₆ to 004F ₁₆)	19	
UART1 receive, ACK1 (Note 3)	+80 to +83 (0050 ₁₆ to 0053 ₁₆)	20	
Timer A0	+84 to +87 (0054 ₁₆ to 0057 ₁₆)	21	Timer
Timer A1	+88 to +91 (0058 ₁₆ to 005B ₁₆)	22	
Timer A2	+92 to +95 (005C ₁₆ to 005F ₁₆)	23	
Timer A3	+96 to +99 (0060 ₁₆ to 0063 ₁₆)	24	
Timer A4	+100 to +103 (0064 ₁₆ to 0067 ₁₆)	25	
Timer B0	+104 to +107 (0068 ₁₆ to 006B ₁₆)	26	
Timer B1	+108 to +111 (006C ₁₆ to 006F ₁₆)	27	
Timer B2	+112 to +115 (0070 ₁₆ to 0073 ₁₆)	28	$\overline{\text{INT}}$ interrupt
$\overline{\text{INT}}_0$	+116 to +119 (0074 ₁₆ to 0077 ₁₆)	29	
$\overline{\text{INT}}_1$	+120 to +123 (0078 ₁₆ to 007B ₁₆)	30	
$\overline{\text{INT}}_2$	+124 to +127 (007C ₁₆ to 007F ₁₆)	31	M16C/60, M16C/20 series software manual
Software interrupt (Note 5)	+128 to +131 (0080 ₁₆ to 0083 ₁₆) to +252 to +255 (00FC ₁₆ to 00FF ₁₆)	32 to 63	

Note 1: Address relative to address in INTB.

Note 2: Use the IFSR register's IFSR6 and IFSR7 bits to select.

Note 3: During I²C mode, NACK and ACK interrupts comprise the interrupt source.

Note 4: Use the IFSR2A register's IFSR26 and IFSR27 bits to select.

Note 5: These interrupts cannot be disabled using the I flag.

Note 6: Bus collision detection : During IE mode, this bus collision detection constitutes the cause of an interrupt.

During I²C mode, however, a start condition or a stop condition detection constitutes the cause of an interrupt.

Note 7: When use SLICEON, remote control, and HINT interruption, refer to address 36₁₆ expansion register of "2.14 Expansion Function."

2.7.5 Interrupt Control

The following describes how to enable/disable the maskable interrupts, and how to set the priority in which order they are accepted. What is explained here does not apply to nonmaskable interrupts.

Use the FLG register's I flag, IPL, and each interrupt control register's ILVL2 to ILVL0 bits to enable/disable the maskable interrupts. Whether an interrupt is requested is indicated by the IR bit in each interrupt control register.

Figure 2.7.3 shows the interrupt control registers.

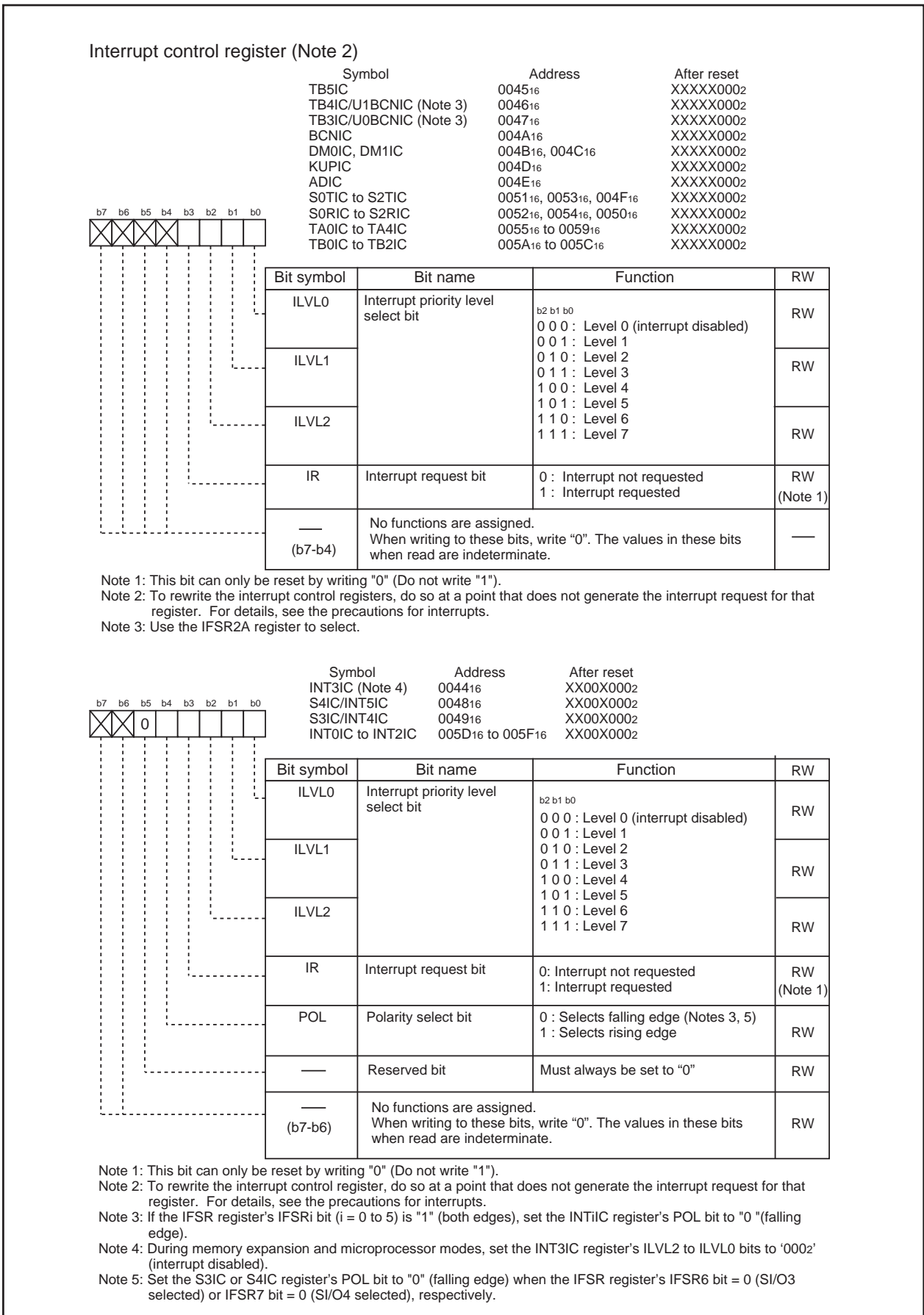


Figure 2.7.3. Interrupt Control Registers

I Flag

The I flag enables or disables the maskable interrupt. Setting the I flag to “1” (= enabled) enables the maskable interrupt. Setting the I flag to “0” (= disabled) disables all maskable interrupts.

IR Bit

The IR bit is set to “1” (= interrupt requested) when an interrupt request is generated. Then, when the interrupt request is accepted and the CPU branches to the corresponding interrupt vector, the IR bit is cleared to “0” (= interrupt not requested).

The IR bit can be cleared to “0” in a program. Note that do not write “1” to this bit.

ILVL2 to ILVL0 Bits and IPL

Interrupt priority levels can be set using the ILVL2 to ILVL0 bits.

Table 2.7.3 shows the settings of interrupt priority levels and Table 2.7.4 shows the interrupt priority levels enabled by the IPL.

The following are conditions under which an interrupt is accepted:

- I flag = “1”
- IR bit = “1”
- interrupt priority level > IPL

The I flag, IR bit, ILVL2 to ILVL0 bits and IPL are independent of each other. In no case do they affect one another.

Table 2.7.3. Settings of Interrupt Priority Levels


ILVL2 to ILVL0 bits	Interrupt priority level	Priority order
0002	Level 0 (interrupt disabled)	————
0012	Level 1	Low  High
0102	Level 2	
0112	Level 3	
1002	Level 4	
1012	Level 5	
1102	Level 6	
1112	Level 7	

Table 2.7.4. Interrupt Priority Levels Enabled by IPL

IPL	Enabled interrupt priority levels
0002	Interrupt levels 1 and above are enabled
0012	Interrupt levels 2 and above are enabled
0102	Interrupt levels 3 and above are enabled
0112	Interrupt levels 4 and above are enabled
1002	Interrupt levels 5 and above are enabled
1012	Interrupt levels 6 and above are enabled
1102	Interrupt levels 7 and above are enabled
1112	All maskable interrupts are disabled

2.7.6 Interrupt Sequence

An interrupt sequence — what are performed over a period from the instant an interrupt is accepted to the instant the interrupt routine is executed — is described here.

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

The CPU behavior during the interrupt sequence is described below. Figure 2.7.4 shows time required for executing the interrupt sequence.

- (1) The CPU gets interrupt information (interrupt number and interrupt request priority level) by reading the address 000016. Then it clears the IR bit for the corresponding interrupt to “0” (interrupt not requested).
- (2) The FLG register immediately before entering the interrupt sequence is saved to the CPU’s internal temporary register^(Note 1).
- (3) The I, D and U flags in the FLG register become as follows:
 The I flag is cleared to “0” (interrupts disabled).
 The D flag is cleared to “0” (single-step interrupt disabled).
 The U flag is cleared to “0” (ISP selected).
 However, the U flag does not change state if an INT instruction for software interrupt Nos. 32 to 63 is executed.
- (4) The CPU’s internal temporary register^(Note 1) is saved to the stack.
- (5) The PC is saved to the stack.
- (6) The interrupt priority level of the accepted interrupt is set in the IPL.
- (7) The start address of the relevant interrupt routine set in the interrupt vector is stored in the PC.

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

Note: This register cannot be used by user.

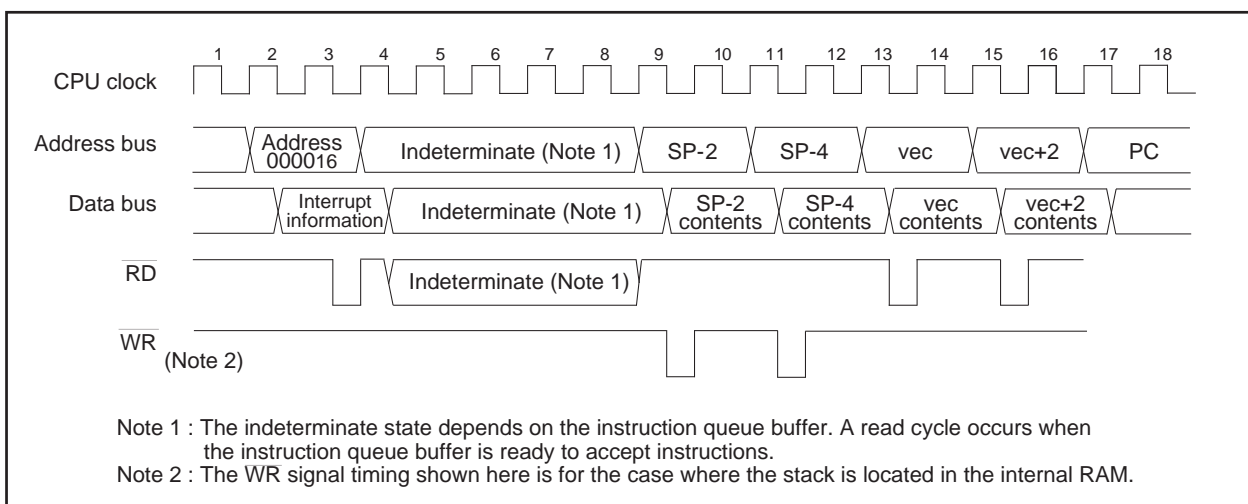


Figure 2.7.4. Time Required for Executing Interrupt Sequence

Interrupt Response Time

Figure 2.7.5 shows the interrupt response time. The interrupt response or interrupt acknowledge time denotes a time from when an interrupt request is generated till when the first instruction in the interrupt routine is executed. Specifically, it consists of a time from when an interrupt request is generated till when the instruction then executing is completed ((a) in Figure 2.7.5) and a time during which the interrupt sequence is executed ((b) in Figure 2.7.5).

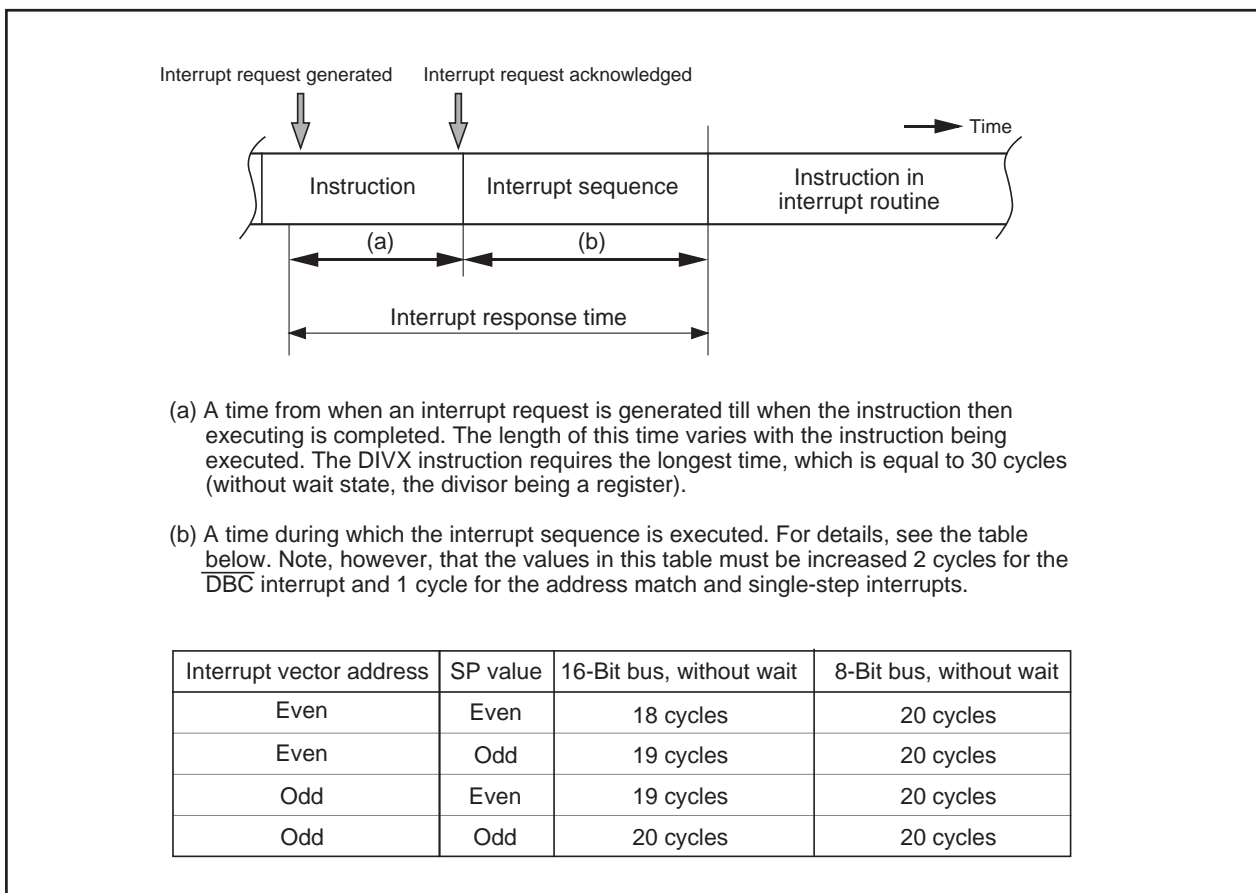


Figure 2.7.5. Interrupt response time

Variation of IPL when Interrupt Request is Accepted

When a maskable interrupt request is accepted, the interrupt priority level of the accepted interrupt is set in the IPL.

When a software interrupt or special interrupt request is accepted, one of the interrupt priority levels listed in Table 2.7.5 is set in the IPL. Shown in Table 2.7.5 are the IPL values of software and special interrupts when they are accepted.

Table 2.7.5. IPL Level That is Set to IPL When A Software or Special Interrupt Is Accepted

Interrupt sources	Level that is set to IPL
Watchdog timer, $\overline{\text{NMI}}$	7
Software, address match, $\overline{\text{DBC}}$, single-step	Not changed

Saving Registers

In the interrupt sequence, the FLG register and PC are saved to the stack.

At this time, the 4 high-order bits of the PC and the 4 high-order (IPL) and 8 low-order bits of the FLG register, 16 bits in total, are saved to the stack first. Next, the 16 low-order bits of the PC are saved. Figure 2.7.6 shows the stack status before and after an interrupt request is accepted.

The other necessary registers must be saved in a program at the beginning of the interrupt routine. Use the PUSHM instruction, and all registers except SP can be saved with a single instruction.

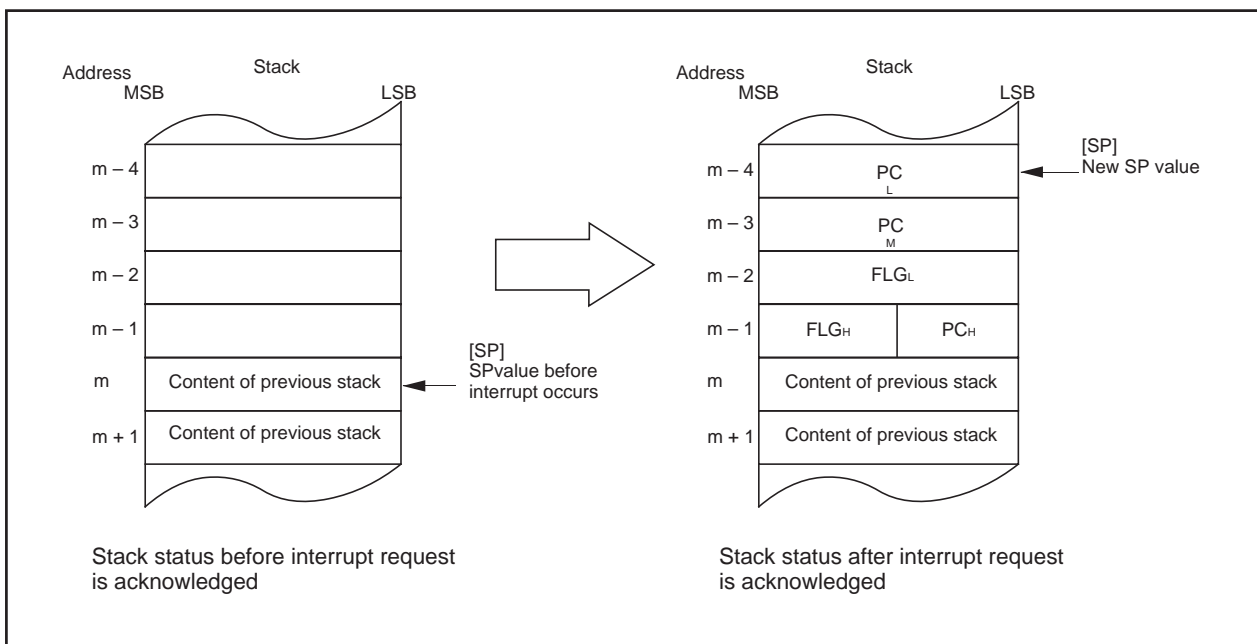


Figure 2.7.6. Stack Status Before and After Acceptance of Interrupt Request

The operation of saving registers carried out in the interrupt sequence is dependent on whether the SP^(Note), at the time of acceptance of an interrupt request, is even or odd. If the stack pointer ^(Note) is even, the FLG register and the PC are saved, 16 bits at a time. If odd, they are saved in two steps, 8 bits at a time. Figure 2.7.7 shows the operation of the saving registers.

Note: When any INT instruction in software numbers 32 to 63 has been executed, this is the SP indicated by the U flag. Otherwise, it is the ISP.

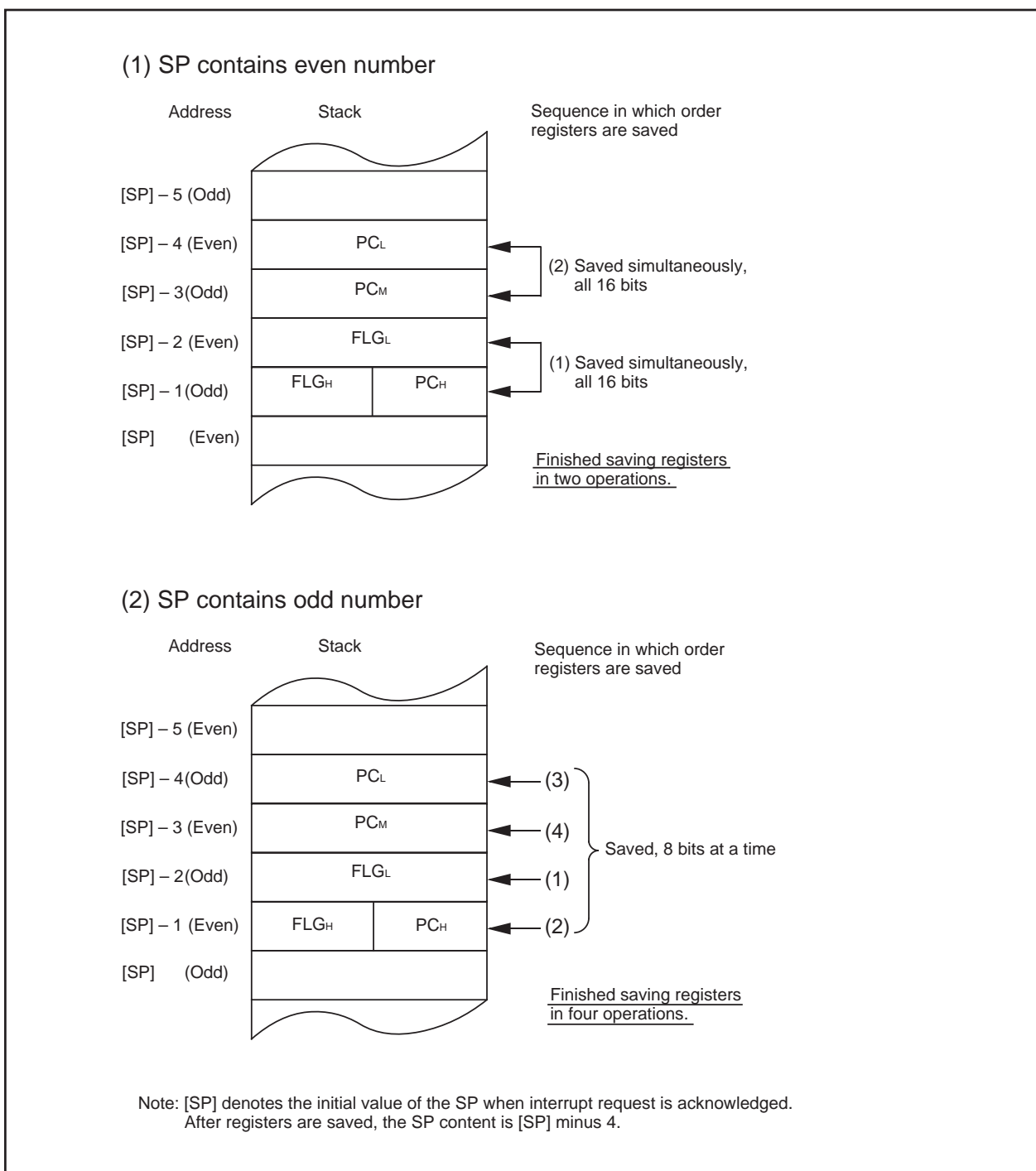


Figure 2.7.7. Operation of Saving Register

Returning from an Interrupt Routine

The FLG register and PC in the state in which they were immediately before entering the interrupt sequence are restored from the stack by executing the REIT instruction at the end of the interrupt routine. Thereafter the CPU returns to the program which was being executed before accepting the interrupt request.

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

Interrupt Priority

If two or more interrupt requests are generated while executing one instruction, the interrupt request that has the highest priority is accepted.

For maskable interrupts (peripheral functions), any desired priority level can be selected using the ILVL2 to ILVL0 bits. However, if two or more maskable interrupts have the same priority level, their interrupt priority is resolved by hardware, with the highest priority interrupt accepted.

The watchdog timer and other special interrupts have their priority levels set in hardware. Figure 2.7.8 shows the priorities of hardware interrupts.

Software interrupts are not affected by the interrupt priority. If an instruction is executed, control branches invariably to the interrupt routine.

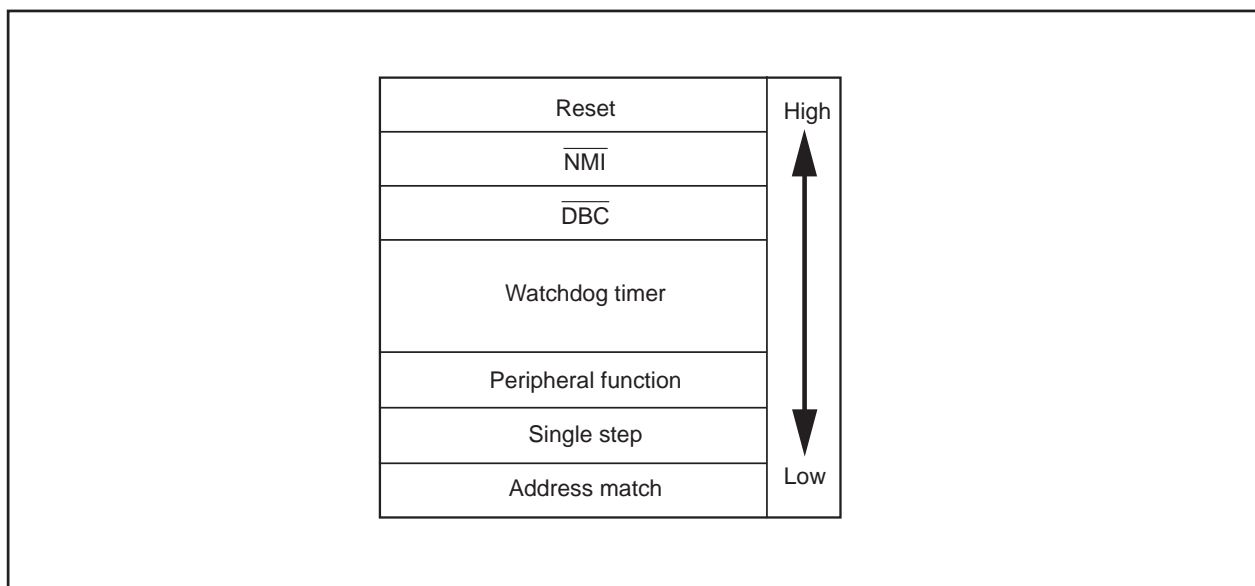


Figure 2.7.8. Hardware Interrupt Priority

Interrupt Priority Resolution Circuit

The interrupt priority resolution circuit is used to select the interrupt with the highest priority among those requested.

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

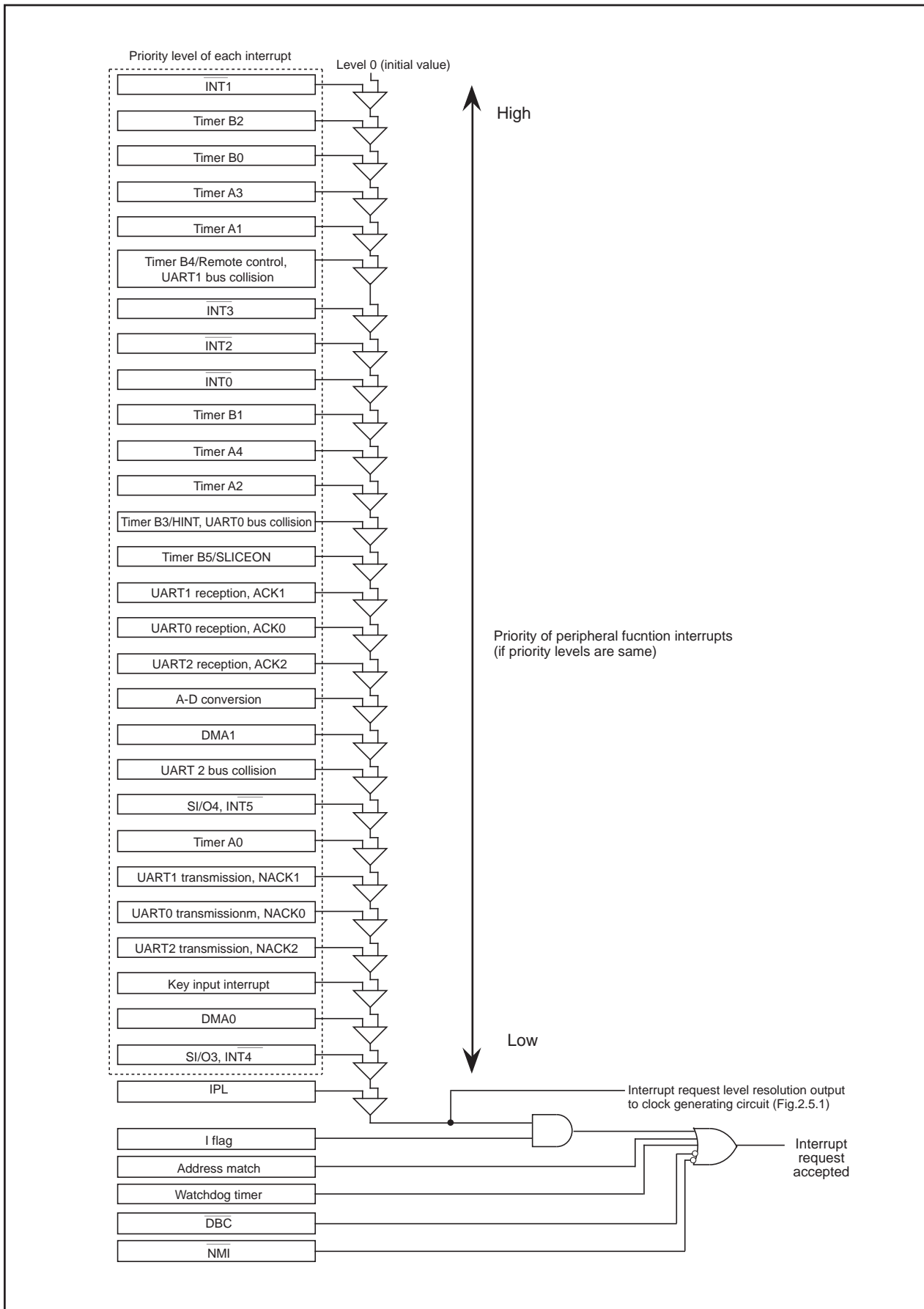


Figure 2.7.9. Interrupts Priority Select Circuit

2.7.7 INT Interrupt

INT_i interrupt (i=0 to 5) is triggered by the edges of external inputs. The edge polarity is selected using the IFSR register's IFSR_i bit.

INT₄ and INT₅ share the interrupt vector and interrupt control register with SI/O₃ and SI/O₄, respectively. To use the INT₄ interrupt, set the IFSR register's IFSR₆ bit to "1" (= INT₄). To use the INT₅ interrupt, set the IFSR register's IFSR₇ bit to "1" (= INT₅).

After modifying the IFSR₆ or IFSR₇ bit, clear the corresponding IR bit to "0" (= interrupt not requested) before enabling the interrupt.

Figure 2.7.10 shows the IFSR and IFSR_{2A} registers.

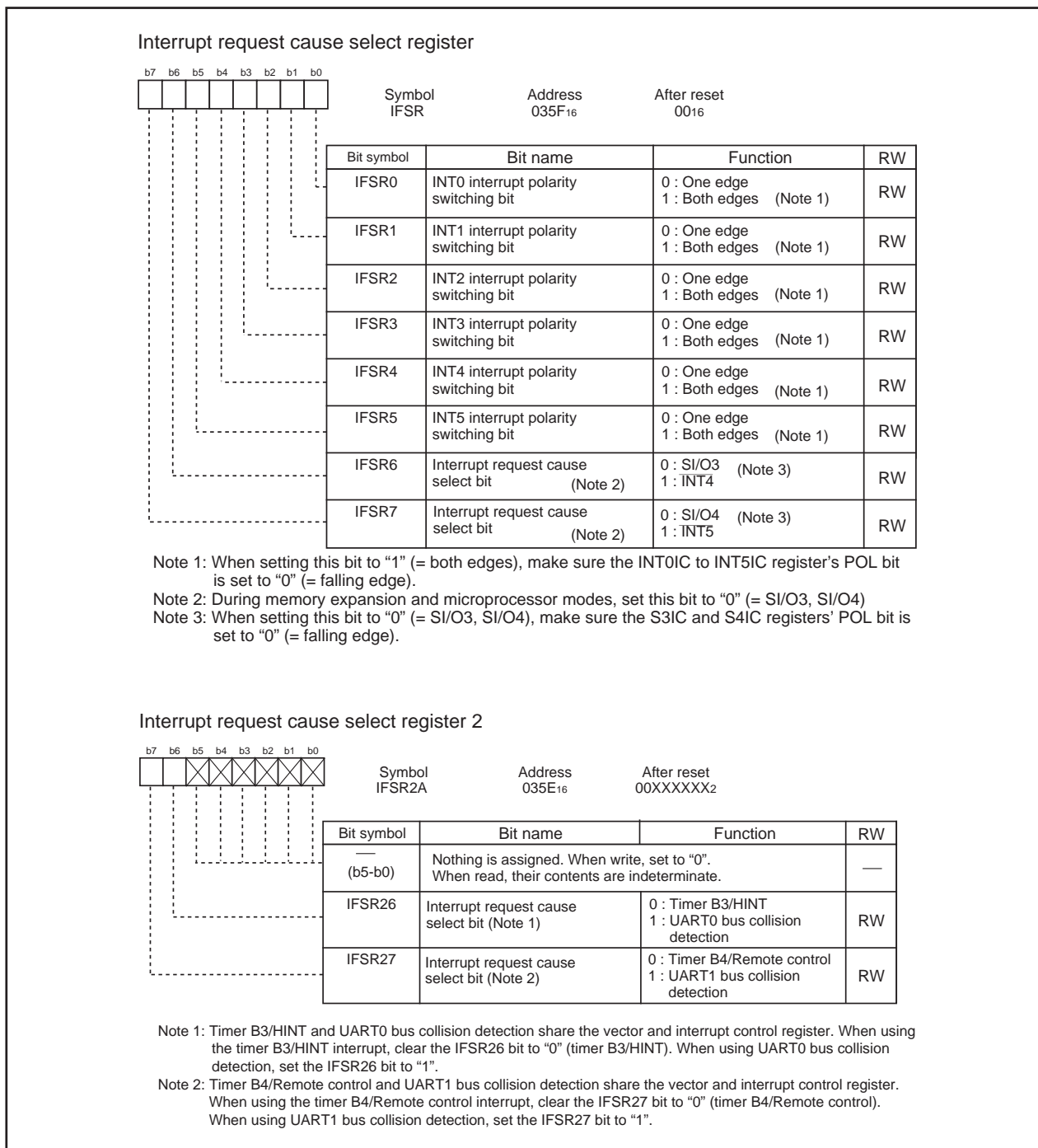


Figure 2.7.10. IFSR Register and IFSR_{2A} Register

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

An $\overline{\text{NMI}}$ interrupt is generated when input on the $\overline{\text{NMI}}$ pin changes state from high to low. The $\overline{\text{NMI}}$ interrupt is a non-maskable interrupt.

The input level of this $\overline{\text{NMI}}$ interrupt input pin can be read by accessing the P8 register's P8_5 bit.

This pin cannot be used as an input port.

2.7.9 Key Input Interrupt

Of P104 to P107, a key input interrupt is generated when input on any of the P104 to P107 pins which has had the PD10 register's PD10_4 to PD10_7 bits set to "0" (= input) goes low. Key input interrupts can be used as a key-on wakeup function, the function which gets the microcomputer out of wait or stop mode. However, if you intend to use the key input interrupt, do not use P104 to P107 as analog input ports. Figure 2.7.11 shows the block diagram of the key input interrupt. Note, however, that while input on any pin which has had the PD10_4 to PD10_7 bits set to "0" (= input mode) is pulled low, inputs on all other pins of the port are not detected as interrupts.

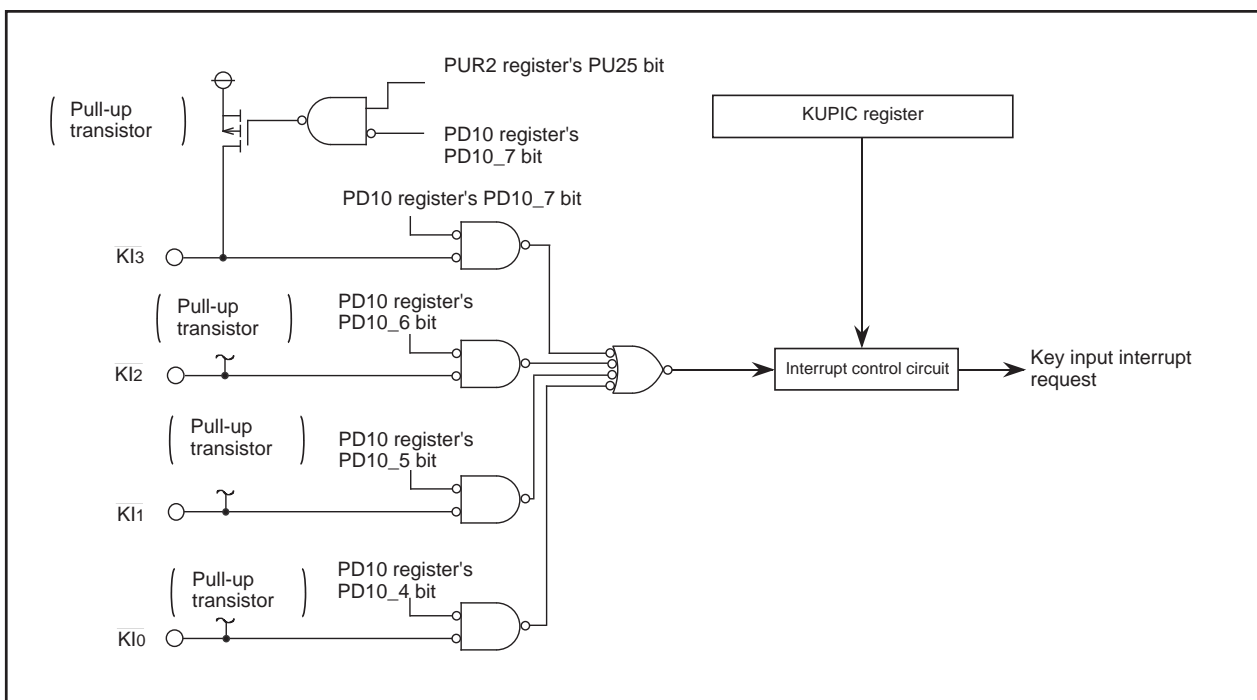


Figure 2.7.11. Key Input Interrupt

2.7.10 Address Match Interrupt

An address match interrupt request is generated immediately before executing the instruction at the address indicated by the RMAD_i register (i=0 to 3). Set the start address of any instruction in the RMAD_i register. Use the AIER register's AIER0 and AIER1 bits and the AIER2 register's AIER20 and AIER21 bits to enable or disable the interrupt. Note that the address match interrupt is unaffected by the I flag and IPL. For address match interrupts, the value of the PC that is saved to the stack area varies depending on the instruction being executed (refer to "Saving Registers").

(The value of the PC that is saved to the stack area is not the correct return address.) Therefore, follow one of the methods described below to return from the address match interrupt.

- Rewrite the content of the stack and then use the REIT instruction to return.
- Restore the stack to its previous state before the interrupt request was accepted by using the POP or similar other instruction and then use a jump instruction to return.

Table 2.7.6 shows the value of the PC that is saved to the stack area when an address match interrupt request is accepted.

Note that when using the external bus in 8 bits width, no address match interrupts can be used for external areas.

Figure 2.7.13 shows the AIER, AIER2, and RMAD0 to RMAD3 registers.

Table 2.7.6. Instruction Just Before Execution and Address Stored in Stack When There Occurs Interrupts

Instruction at the address indicated by the RMAD _i register	Value of the PC that is saved to the stack area
<ul style="list-style-type: none"> • 16-bit op-code instruction • Instruction shown below among 8-bit operation code instructions <pre> ADD.B:S #IMM8,dest SUB.B:S #IMM8,dest AND.B:S #IMM8,dest OR.B:S #IMM8,dest MOV.B:S #IMM8,dest STZ.B:S #IMM8,dest STNZ.B:S #IMM8,dest STZX.B:S #IMM81,#IMM82,dest CMP.B:S #IMM8,dest PUSHM src POPM dest JMPS #IMM8 JSRS #IMM8 MOV.B:S #IMM,dest (However, dest=A0 or A1) </pre>	The address indicated by the RMAD _i register +2
Instructions other than the above	The address indicated by the RMAD _i register +1

Value of the PC that is saved to the stack area : Refer to "Saving Registers".

Table 2.7.7. Relationship Between Address Match Interrupt Sources and Associated Registers

Address match interrupt sources	Address match interrupt enable bit	Address match interrupt register
Address match interrupt 0	AIER0	RMAD0
Address match interrupt 1	AIER1	RMAD1
Address match interrupt 2	AIER20	RMAD2
Address match interrupt 3	AIER21	RMAD3

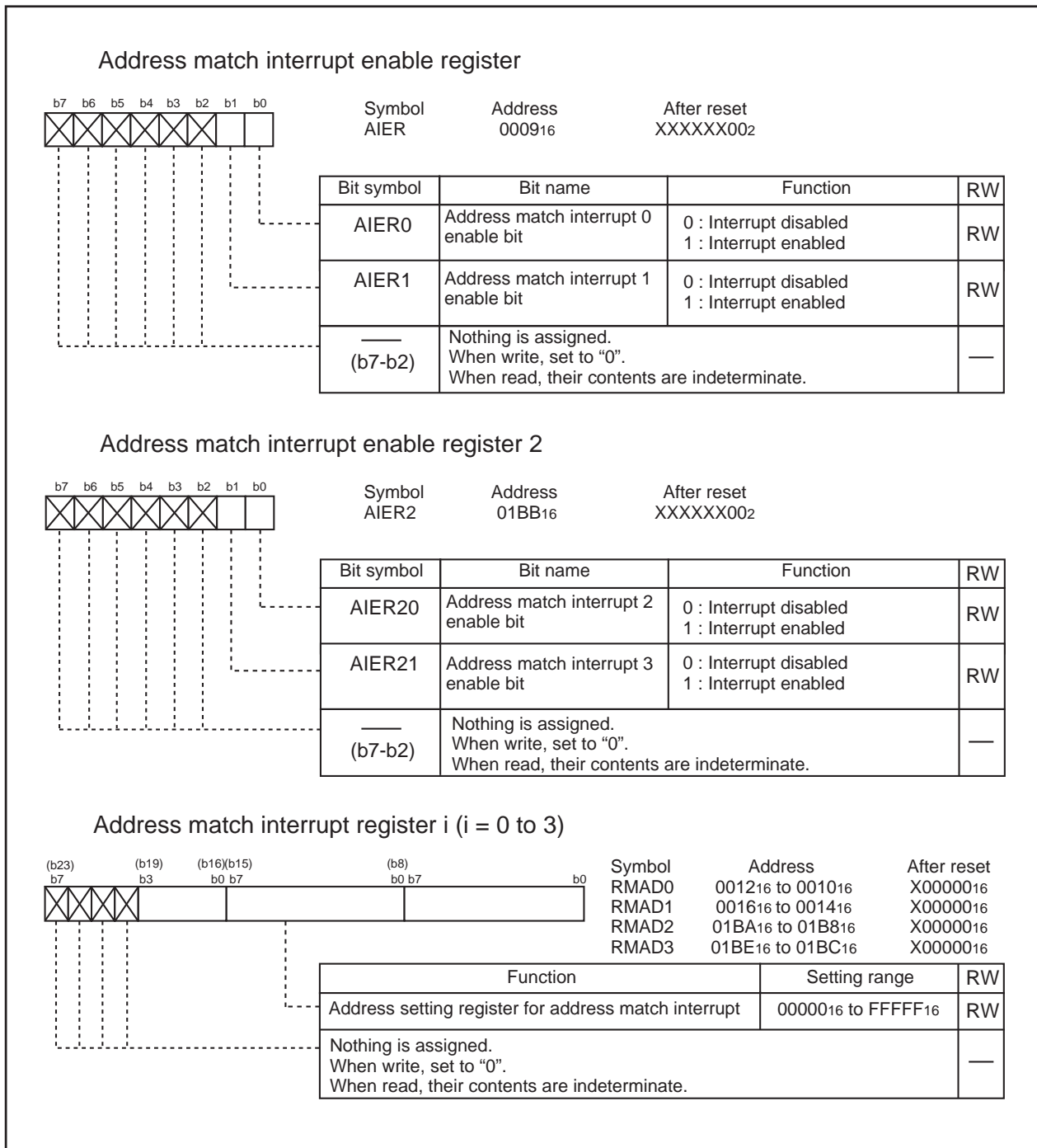


Figure 2.7.12. AIER Register, AIER2 Register and RMAD0 to RMAD3 Registers

2.8 Watchdog Timer

The watchdog timer is the function of detecting when the program is out of control. Therefore, we recommend using the watchdog timer to improve reliability of a system. The watchdog timer contains a 15-bit counter which counts down the clock derived by dividing the CPU clock using the prescaler. Whether to generate a watchdog timer interrupt request or apply a watchdog timer reset as an operation to be performed when the watchdog timer underflows after reaching the terminal count can be selected using the PM12 bit of PM1 register. The PM12 bit can only be set to "1" (reset). Once this bit is set to "1", it cannot be set to "0" (watchdog timer interrupt) in a program.

Refer to "Watchdog Timer Reset" for the details of watchdog timer reset.

When the main clock is selected for CPU clock, the divide-by-N value for the prescaler can be chosen to be 16 or 128 using the WDC7 bit of WDC register. If a sub-clock is selected for CPU clock, the divide-by-N value for the prescaler is always 2 no matter how the WDC7 bit is set. The period of watchdog timer can be calculated as given below. The period of watchdog timer is, however, subject to an error due to the prescaler.

With main clock chosen for CPU clock

$$\text{Watchdog timer period} = \frac{\text{Prescaler dividing (16 or 128)} \times \text{Watchdog timer count (32768)}}{\text{CPU clock}}$$

With sub-clock chosen for CPU clock

$$\text{Watchdog timer period} = \frac{\text{Prescaler dividing (2)} \times \text{Watchdog timer count (32768)}}{\text{CPU clock}}$$

For example, when CPU clock = 10 MHz and the divide-by-N value for the prescaler= 16, the watchdog timer period is approx. 52.4 ms.

The watchdog timer is initialized by writing to the WDTS register. The prescaler is initialized after reset. Note that the watchdog timer and the prescaler both are inactive after reset, so that the watchdog timer is activated to start counting by writing to the WDTS register.

In stop mode, wait mode and hold state, the watchdog timer and prescaler are stopped. Counting is resumed from the held value when the modes or state are released.

Figure 2.8.1 shows the block diagram of the watchdog timer. Figure 2.8.2 shows the watchdog timer-related registers.

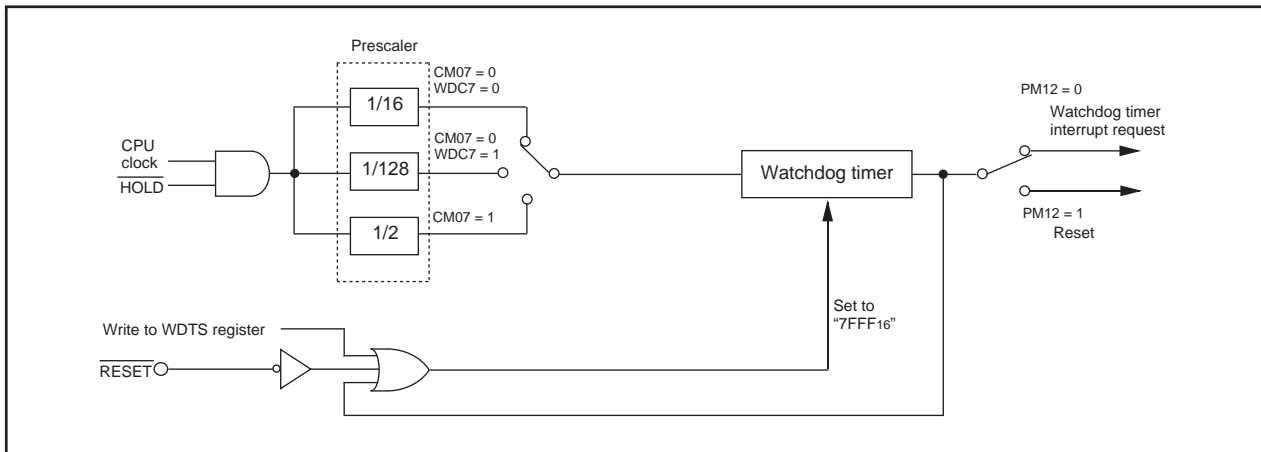


Figure 2.8.1. Watchdog Timer Block Diagram

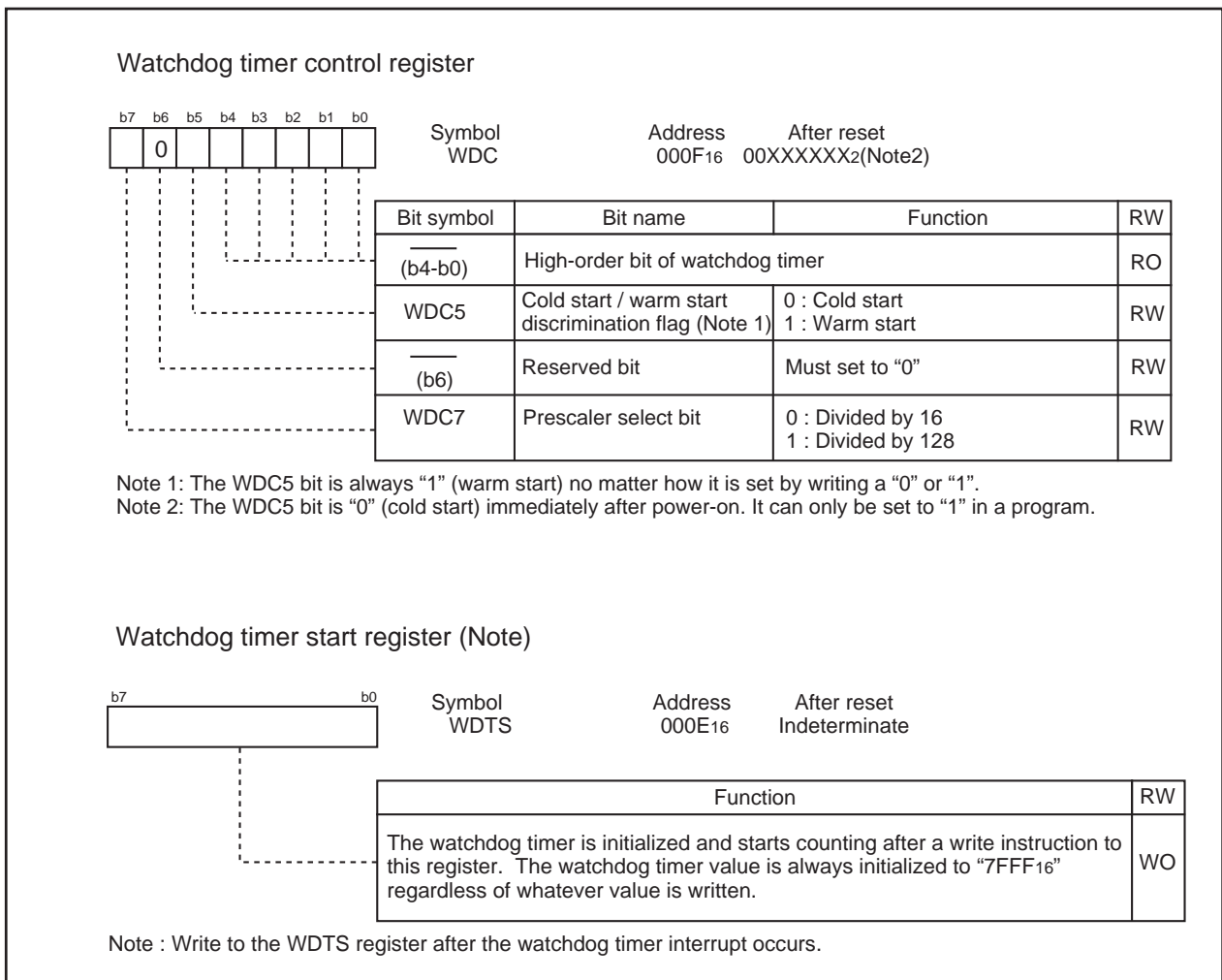


Figure 2.8.2. WDC Register and WDTS Register

2.9 DMAC

The DMAC (Direct Memory Access Controller) allows data to be transferred without the CPU intervention. Two DMAC channels are included. Each time a DMA request occurs, the DMAC transfers one (8 or 16-bit) data from the source address to the destination address. The DMAC uses the same data bus as used by the CPU. Because the DMAC has higher priority of bus control than the CPU and because it makes use of a cycle steal method, it can transfer one word (16 bits) or one byte (8 bits) of data within a very short time after a DMA request is generated. Figure 2.9.1 shows the block diagram of the DMAC. Table 2.9.1 shows the DMAC specifications. Figures 2.9.2 to 2.9.4 show the DMAC-related registers.

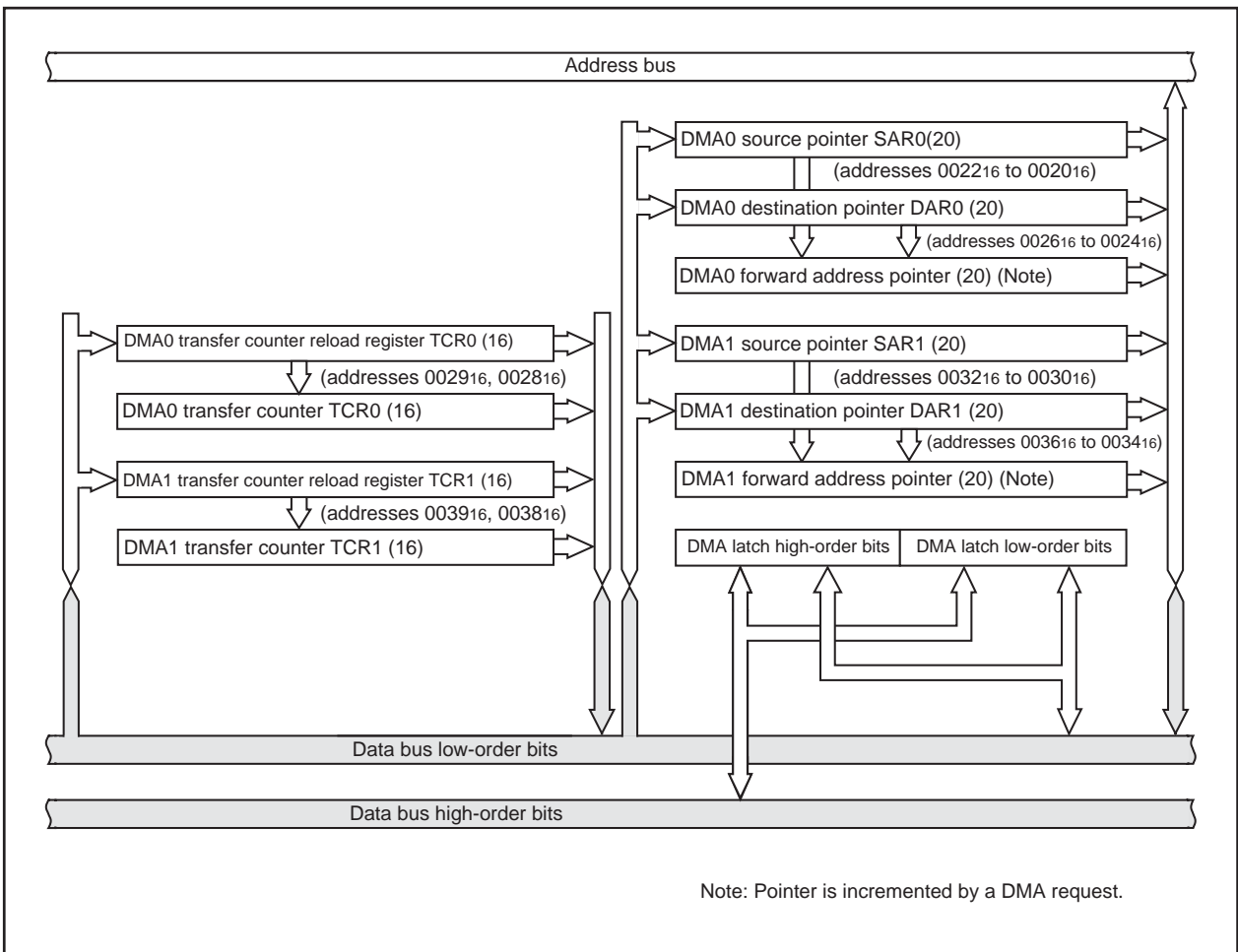


Figure 2.9.1. DMAC Block Diagram

A DMA request is generated by a write to the DMiSL register (i = 0 to 1)'s DSR bit, as well as by an interrupt request which is generated by any function specified by the DMiSL register's DMS and DSEL3 to DSEL0 bits. However, unlike in the case of interrupt requests, DMA requests are not affected by the I flag and the interrupt control register, so that even when interrupt requests are disabled and no interrupt request can be accepted, DMA requests are always accepted. Furthermore, because the DMAC does not affect interrupts, the interrupt control register's IR bit does not change state due to a DMA transfer.

A data transfer is initiated each time a DMA request is generated when the DMiCON register's DMAE bit = "1" (DMA enabled). However, if the cycle in which a DMA request is generated is faster than the DMA transfer cycle, the number of transfer requests generated and the number of times data is transferred may not match. For details, refer to "DMA Requests".

Table 2.9.1. DMAC Specifications

Item	Specification	
No. of channels	2 (cycle steal method)	
Transfer memory space	<ul style="list-style-type: none"> • From any address in the 1M bytes space to a fixed address • From a fixed address to any address in the 1M bytes space • From a fixed address to a fixed address 	
Maximum No. of bytes transferred	128K bytes (with 16-bit transfers) or 64K bytes (with 8-bit transfers)	
DMA request factors (Note 1, Note 2)	Falling edge of $\overline{INT0}$ or $\overline{INT1}$ Both edge of $\overline{INT0}$ or $\overline{INT1}$ Timer A0 to timer A4 interrupt requests Timer B0 to timer B5 interrupt requests UART0 transfer, UART0 reception interrupt requests UART1 transfer, UART1 reception interrupt requests UART2 transfer, UART2 reception interrupt requests SI/O3, SI/O4 interrupt requests A-D conversion interrupt requests Software triggers	
Channel priority	DMA0 > DMA1 (DMA0 takes precedence)	
Transfer unit	8 bits or 16 bits	
Transfer address direction	forward or fixed (The source and destination addresses cannot both be in the forward direction.)	
Transfer mode	•Single transfer	Transfer is completed when the DMA _i transfer counter (i = 0–1) underflows after reaching the terminal count.
	•Repeat transfer	When the DMA _i transfer counter underflows, it is reloaded with the value of the DMA _i transfer counter reload register and a DMA transfer is continued with it.
DMA interrupt request generation timing	When the DMA _i transfer counter underflowed	
DMA startup	Data transfer is initiated each time a DMA request is generated when the DMA _i CON register's DMAE bit = "1" (enabled).	
DMA shutdown	•Single transfer	<ul style="list-style-type: none"> • When the DMAE bit is set to "0" (disabled) • After the DMA_i transfer counter underflows
	•Repeat transfer	When the DMAE bit is set to "0" (disabled)
Reload timing for forward address pointer and transfer counter	When a data transfer is started after setting the DMAE bit to "1" (enabled), the forward address pointer is reloaded with the value of the SAR _i or the DAR _i pointer whichever is specified to be in the forward direction and the DMA _i transfer counter is reloaded with the value of the DMA _i transfer counter reload register.	

Notes:

1. DMA transfer is not effective to any interrupt. DMA transfer is affected neither by the I flag nor by the interrupt control register.
2. The selectable causes of DMA requests differ with each channel.
3. Make sure that no DMAC-related registers (addresses 0020₁₆ to 003F₁₆) are accessed by the DMAC.

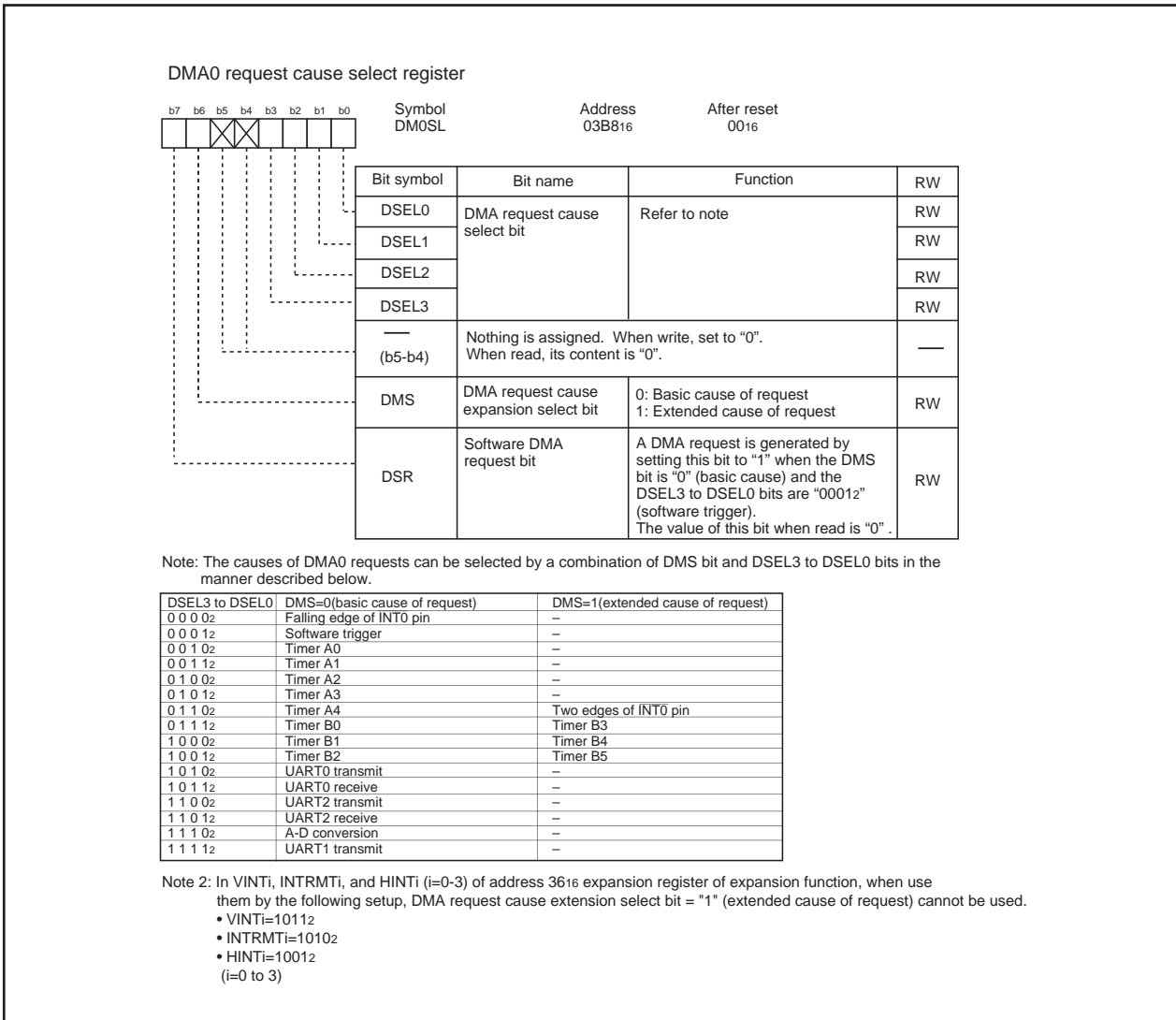


Figure 2.9.2. DM0SL Register

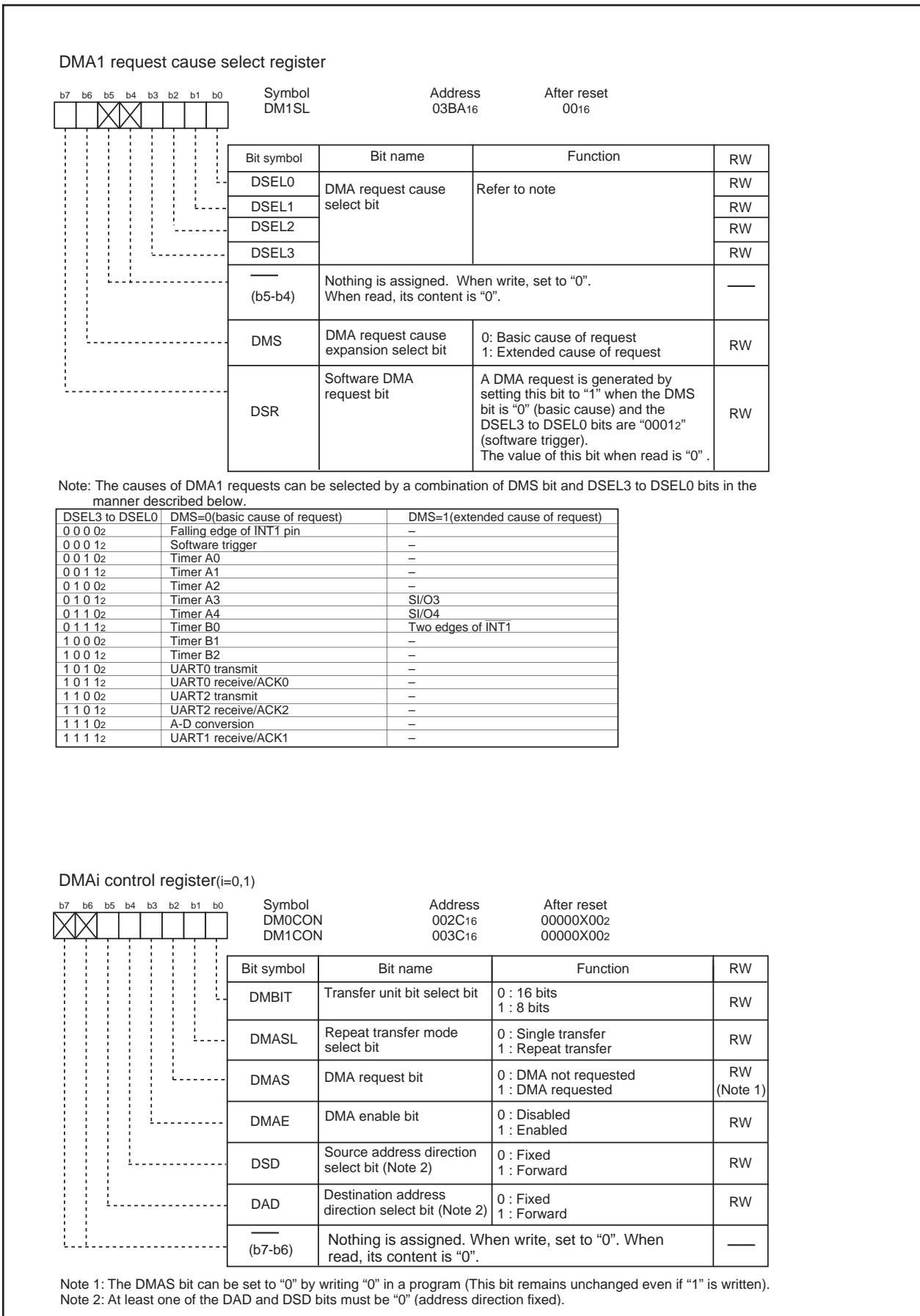


Figure 2.9.3. DM1SL Register, DM0CON Register, and DM1CON Registers

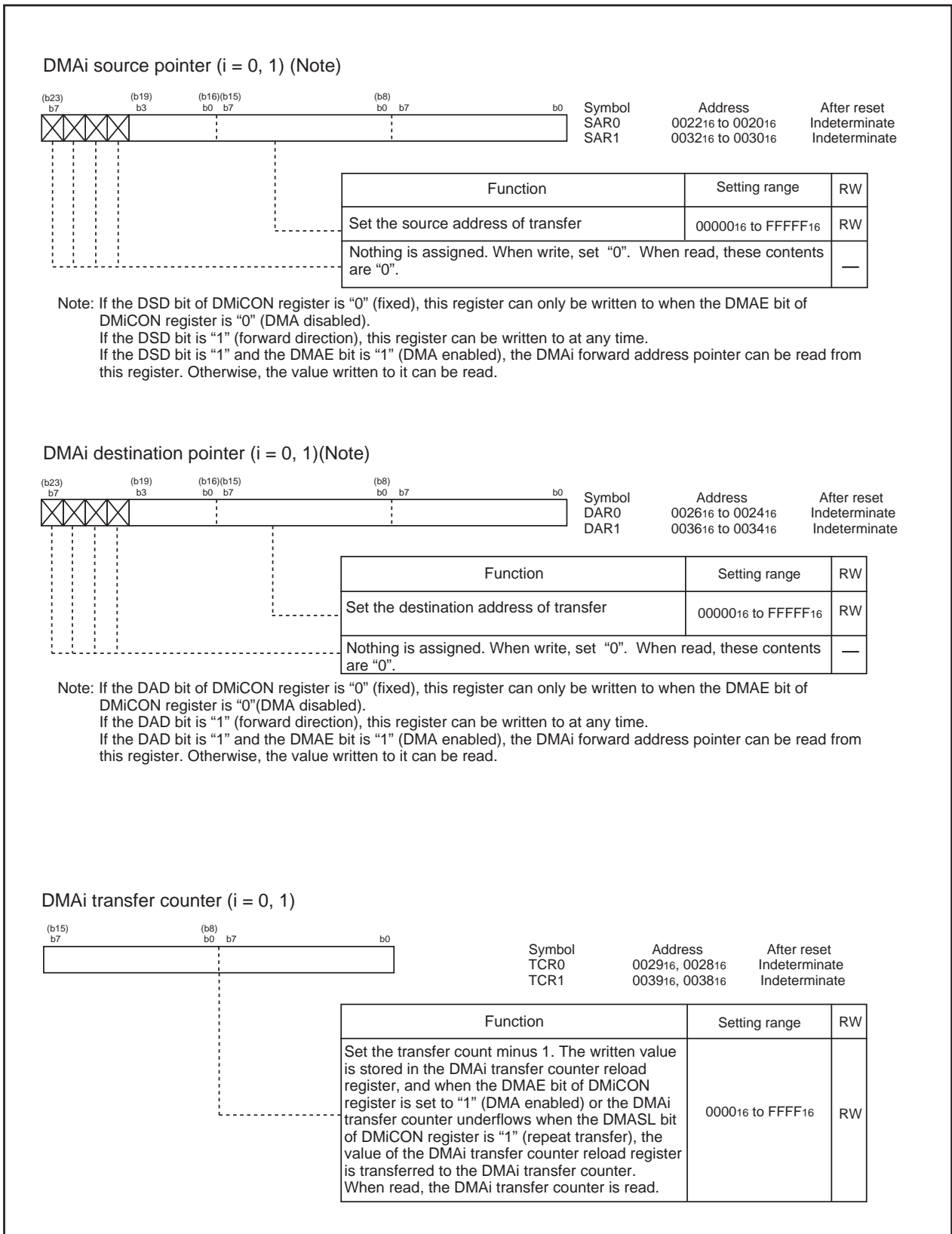


Figure 2.9.4. SAR0, SAR1, DAR0, DAR1, TCR0, and TCR1 Registers

2.9.1 Transfer Cycles

The transfer cycle consists of a memory or SFR read (source read) bus cycle and a write (destination write) bus cycle. The number of read and write bus cycles is affected by the source and destination addresses of transfer. During memory extension and microprocessor modes, it is also affected by the BYTE pin level. Furthermore, the bus cycle itself is extended by a software wait or $\overline{\text{RDY}}$ signal.

(a) Effect of Source and Destination Addresses

If the transfer unit and data bus both are 16 bits and the source address of transfer begins with an odd address, the source read cycle consists of one more bus cycle than when the source address of transfer begins with an even address.

Similarly, if the transfer unit and data bus both are 16 bits and the destination address of transfer begins with an odd address, the destination write cycle consists of one more bus cycle than when the destination address of transfer begins with an even address.

(b) Effect of BYTE Pin Level

During memory extension and microprocessor modes, if 16 bits of data are to be transferred on an 8-bit data bus (input on the BYTE pin = high), the operation is accomplished by transferring 8 bits of data twice. Therefore, this operation requires two bus cycles to read data and two bus cycles to write data. Furthermore, if the DMAC is to access the internal area (internal ROM, internal RAM, or SFR), unlike in the case of the CPU, the DMAC does it through the data bus width selected by the BYTE pin.

(c) Effect of Software Wait

For memory or SFR accesses in which one or more software wait states are inserted, the number of bus cycles required for that access increases by an amount equal to software wait states.

(d) Effect of $\overline{\text{RDY}}$ Signal

During memory extension and microprocessor modes, DMA transfers to and from an external area are affected by the $\overline{\text{RDY}}$ signal. Refer to “ $\overline{\text{RDY}}$ signal”.

Figure 2.9.5 shows the example of the 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 transfer cycles, take into consideration each condition for the source read and the destination write cycle, respectively. For example, when data is transferred in 16 bit units using an 8-bit bus ((2) in Figure 2.9.5), two source read bus cycles and two destination write bus cycles are required.

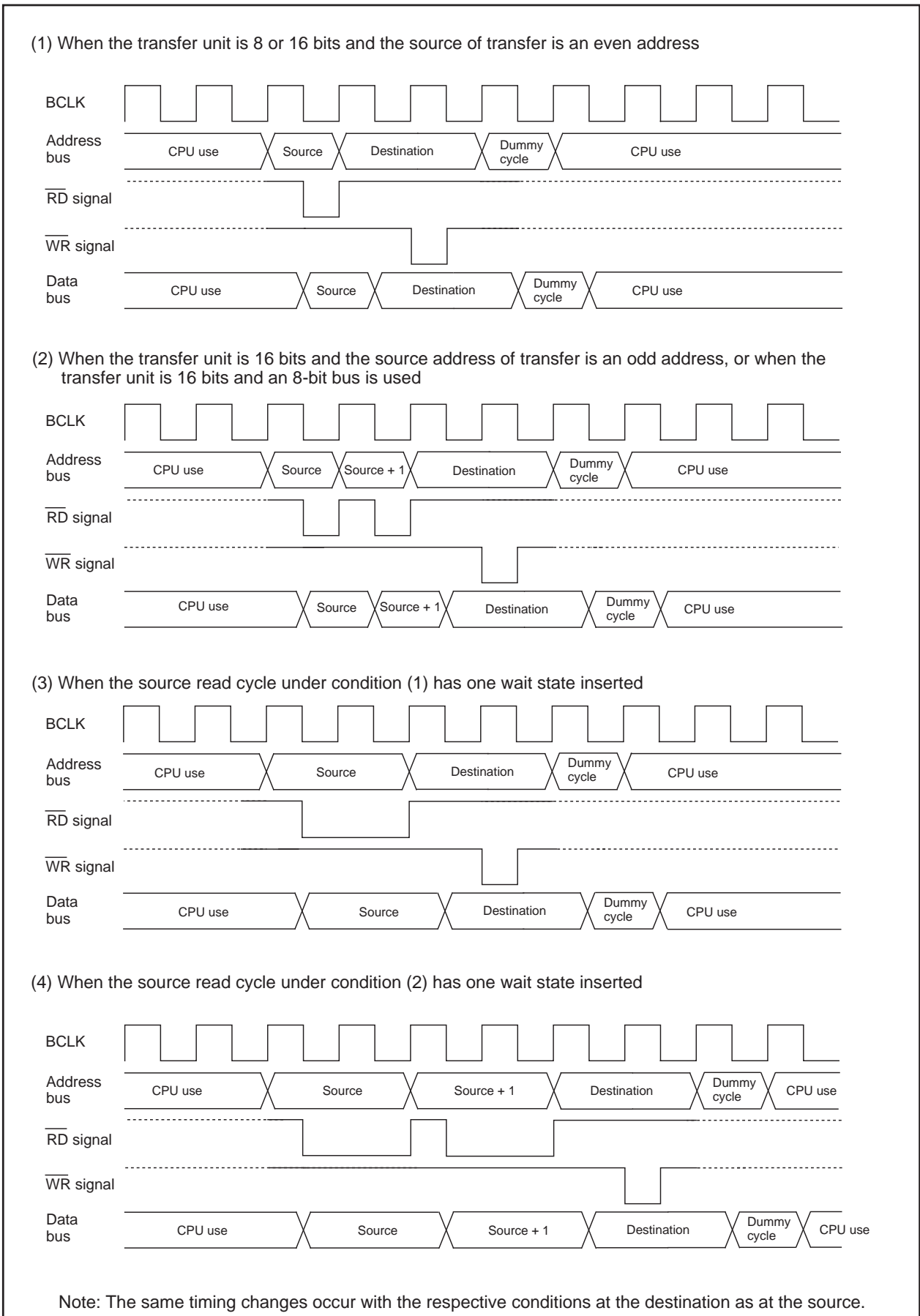


Figure 2.9.5. Transfer Cycles for Source Read

2.9.2 Number of DMA Transfer Cycles

Any combination of even or odd transfer read and write addresses is possible. Table 2.9.2 shows the number of DMA transfer cycles. Table 2.9.3 shows the Coefficient j, k.

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

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

Table 2.9.2. Number of DMA Transfer Cycles

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

Table 2.9.3. Coefficient j, k

	Internal area			External area						
	Internal ROM, RAM		SFR	Separate bus				Multiplex bus		
	No wait	With wait		No wait	With wait ¹			With wait ¹		
					1 wait	2 waits	3 waits	1wait	2 waits	3 waits
j	1	2	2	1	2	3	4	3	3	4
k	1	2	2	2	2	3	4	3	3	4

Notes:

1. Depends on the set value of CSE register.

2.9.3 DMA Enable

When a data transfer starts after setting the DMAE bit in DMiCON register ($i = 0, 1$) to “1” (enabled), the DMAC operates as follows:

- (1) Reload the forward address pointer with the SARi register value when the DSD bit in DMiCON register is “1” (forward) or the DARi register value when the DAD bit of DMiCON register is “1” (forward).
- (2) Reload the DMAi transfer counter with the DMAi transfer counter reload register value.

If the DMAE bit is set to “1” again while it remains set, the DMAC performs the above operation. However, if a DMA request may occur simultaneously when the DMAE bit is being written, follow the steps below.

Step 1: Write “1” to the DMAE bit and DMAS bit in DMiCON register simultaneously.

Step 2: Make sure that the DMAi is in an initial state as described above (1) and (2) in a program.

If the DMAi is not in an initial state, the above steps should be repeated.

2.9.4 DMA Request

The DMAC can generate a DMA request as triggered by the cause of request that is selected with the DMS and DSEL3 to DSEL0 bits of DMiSL register ($i = 0, 1$) on either channel. Table 2.9.4 shows the timing at which the DMAS bit changes state.

Whenever a DMA request is generated, the DMAS bit is set to “1” (DMA requested) regardless of whether or not the DMAE bit is set. If the DMAE bit was set to “1” (enabled) when this occurred, the DMAS bit is set to “0” (DMA not requested) immediately before a data transfer starts. This bit cannot be set to “1” in a program (it can only be set to “0”).

The DMAS bit may be set to “1” when the DMS or the DSEL3 to DSEL0 bits change state. Therefore, always be sure to set the DMAS bit to “0” after changing the DMS or the DSEL3 to DSEL0 bits.

Because if the DMAE bit is “1”, a data transfer starts immediately after a DMA request is generated, the DMAS bit in almost all cases is “0” when read in a program. Read the DMAE bit to determine whether the DMAC is enabled.

Table 2.9.4. Timing at Which the DMAS Bit Changes State

DMA factor	DMAS bit of the DMiCON register	
	Timing at which the bit is set to “1”	Timing at which the bit is set to “0”
Software trigger	When the DSR bit of DMiSL register is set to “1”	<ul style="list-style-type: none"> • Immediately before a data transfer starts • When set by writing “0” in a program
Peripheral function	When the interrupt control register for the peripheral function that is selected by the DSEL3 to DSEL0 and DMS bits of DMiSL register has its IR bit set to “1”	

2.9.5 Channel Priority and DMA Transfer Timing

If both DMA0 and DMA1 are enabled and DMA transfer request signals from DMA0 and DMA1 are detected active in the same sampling period (one period from a falling edge to the next falling edge of BCLK), the DMAS bit on each channel is set to "1" (DMA requested) at the same time. In this case, the DMA requests are arbitrated according to the channel priority, DMA0 > DMA1. The following describes DMAC operation when DMA0 and DMA1 requests are detected active in the same sampling period. Figure 2.9.6 shows an example of DMA transfer effected by external factors.

DMA0 request having priority is received first to start a transfer when a DMA0 request and DMA1 request are generated simultaneously. After one DMA0 transfer is completed, a bus arbitration is returned to the CPU. When the CPU has completed one bus access, a DMA1 transfer starts. After one DMA1 transfer is completed, the bus arbitration is again returned to the CPU.

In addition, DMA requests cannot be counted up since each channel has one DMAS bit. Therefore, when DMA requests, as DMA1 in Figure 2.9.6, occurs more than one time, the DMAS bit is set to "0" as soon as getting the bus arbitration. The bus arbitration is returned to the CPU when one transfer is completed. Refer to "(7) Hold Signal in 2.4.2 Bus Control" for details about bus arbitration between the CPU and DMA.

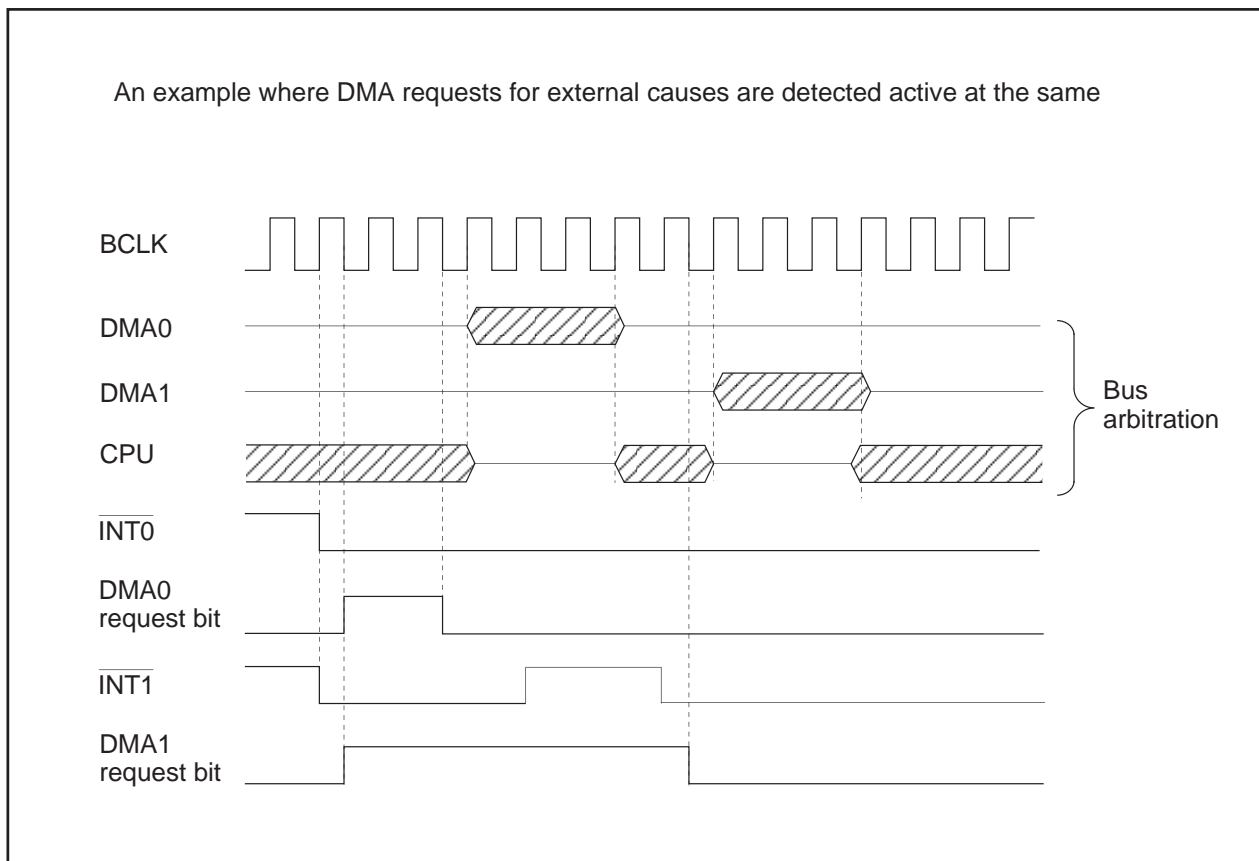


Figure 2.9.6. DMA Transfer by External Factors

2.10 Timers

Eleven 16-bit timers, each capable of operating independently of the others, can be classified by function as either timer A (five) and timer B (six). The count source for each timer acts as a clock, to control such timer operations as counting, reloading, etc. Figures 2.10.1 and 2.10.2 show block diagrams of timer A and timer B configuration, respectively.

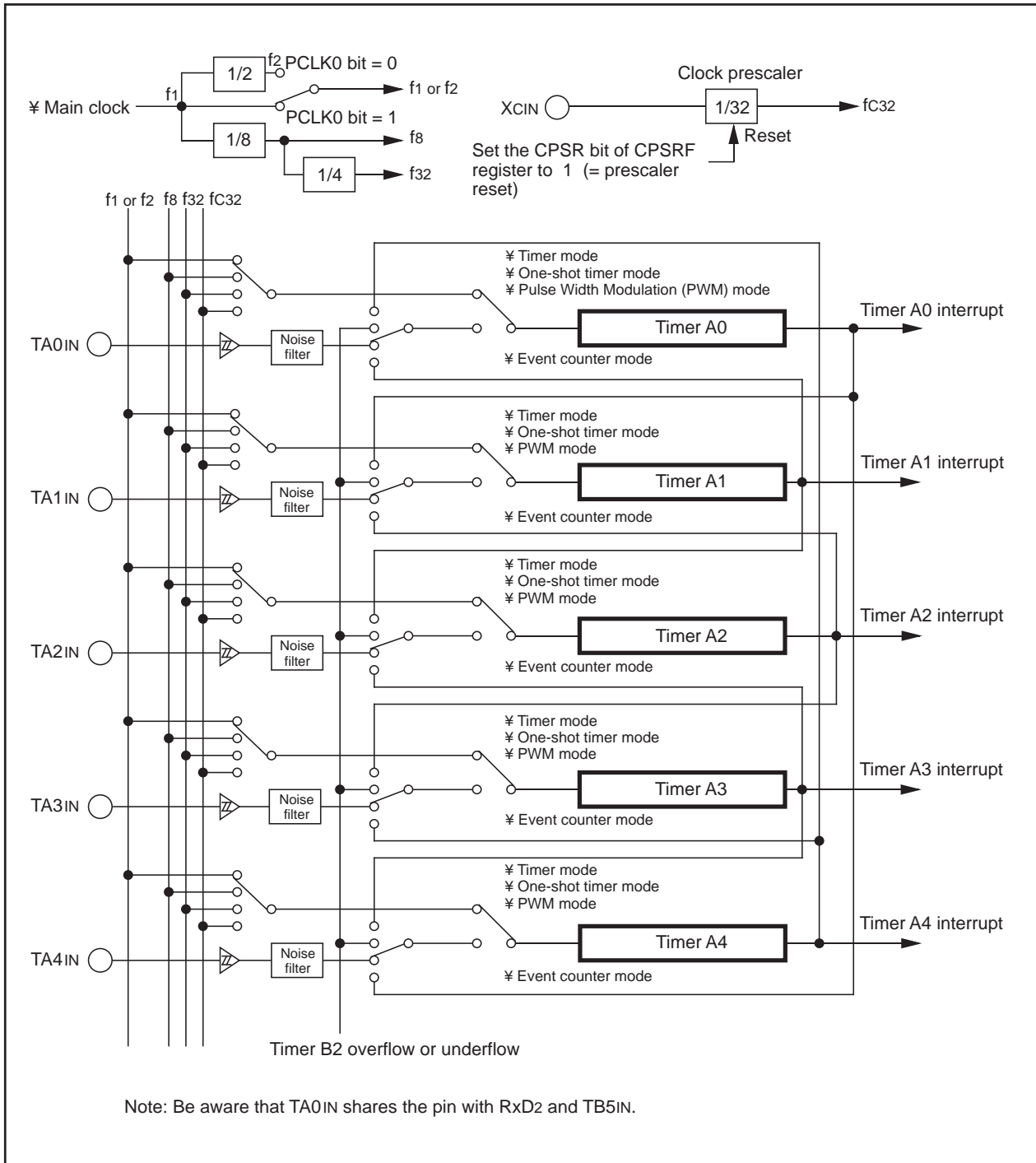


Figure 2.10.1. Timer A Configuration

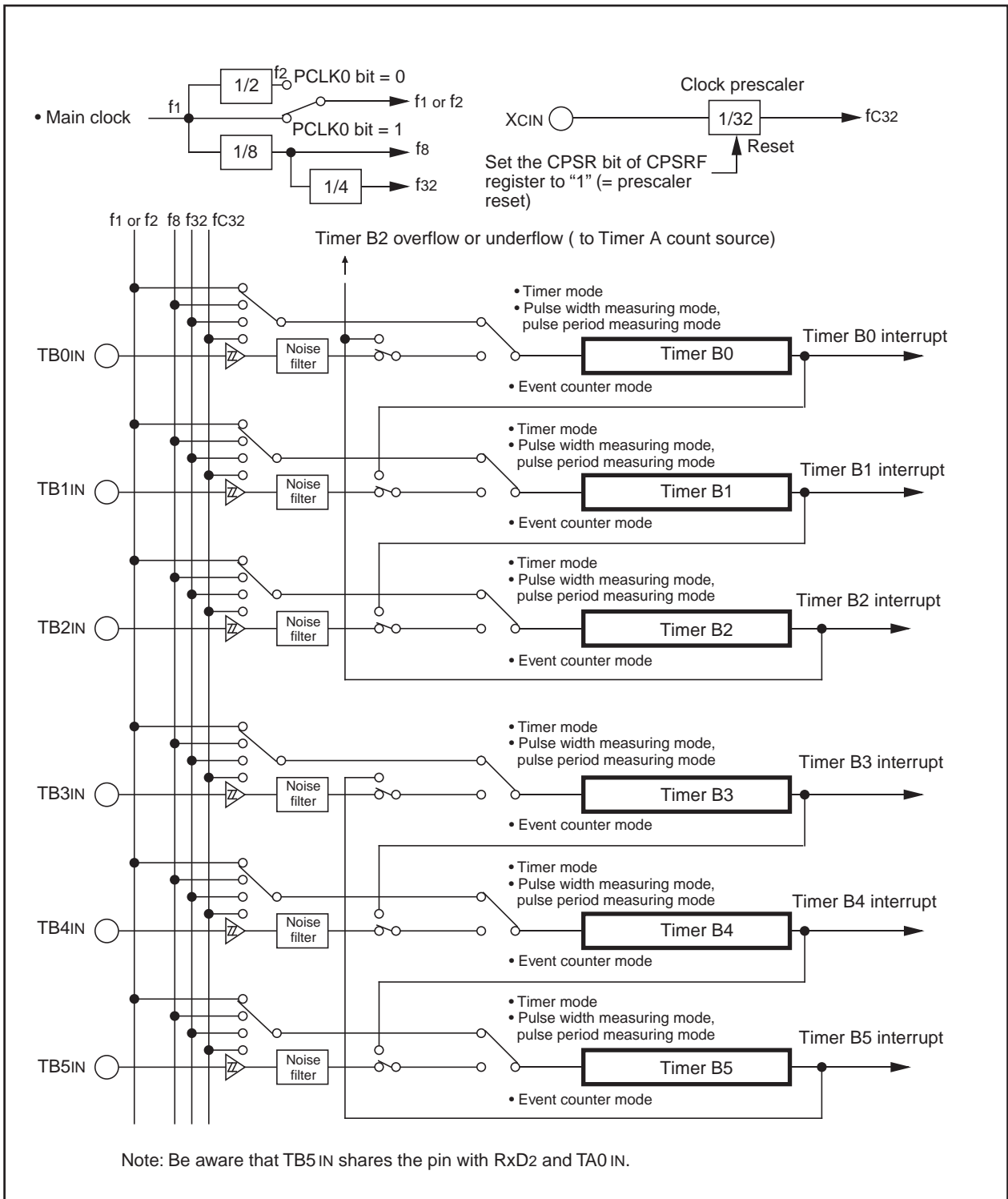


Figure 2.10.2. Timer B Configuration

2.10.1 Timer A

Figure 2.10.3 shows a block diagram of the timer A. Figures 2.10.4 to 2.10.6 show registers related to the timer A.

The timer A supports the following four modes. Except in event counter mode, timers A0 to A4 all have the same function. Use the TMOD1 to TMOD0 bits of TAI_{MR} register (i = 0 to 4) to select the desired mode.

- Timer mode: The timer counts an internal count source.
- Event counter mode: The timer counts pulses from an external device or overflows and underflows of other timers.
- One-shot timer mode: The timer outputs a pulse only once before it reaches the minimum count “0000₁₆.”
- Pulse width modulation (PWM) mode: The timer outputs pulses in a given width successively.

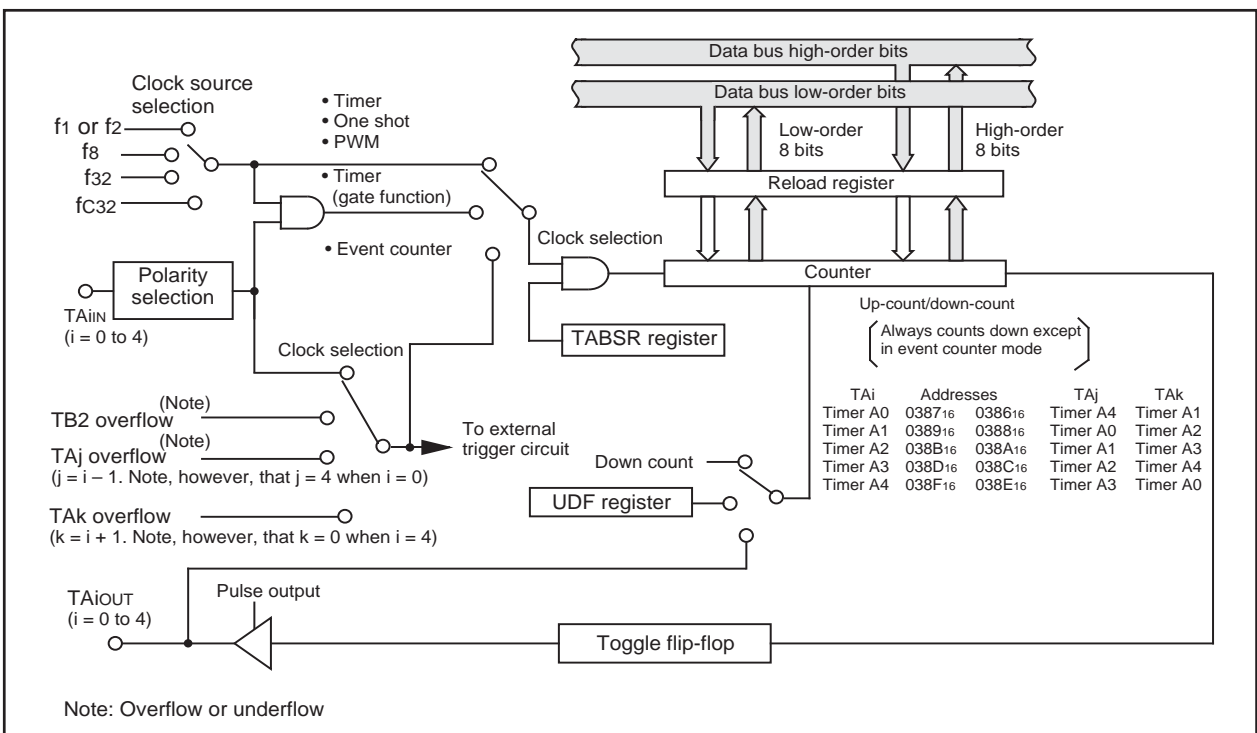


Figure 2.10.3. Timer A Block Diagram

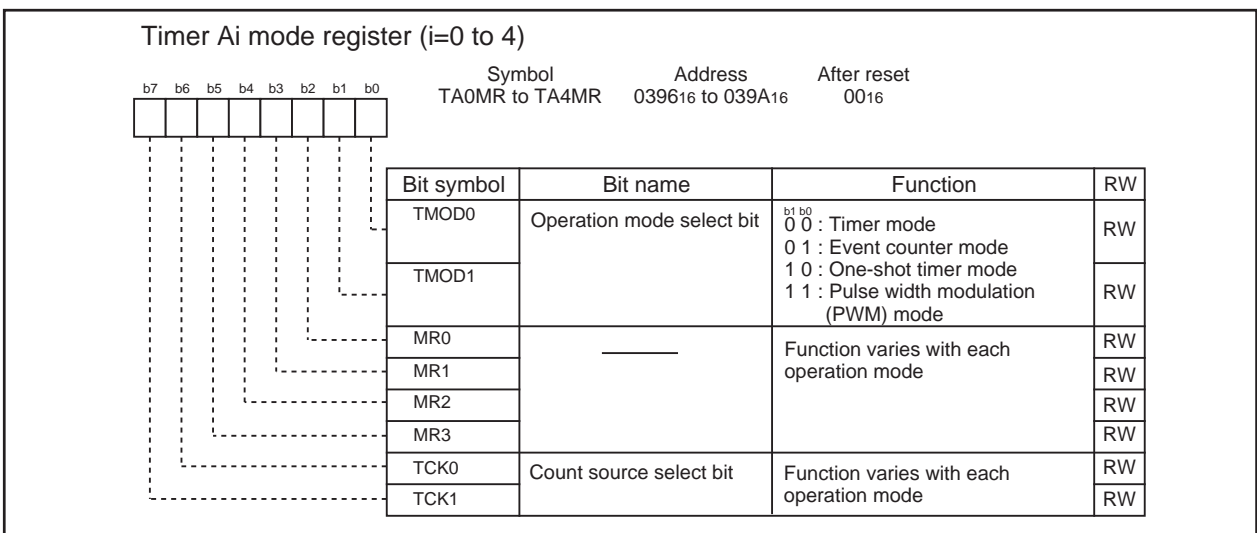


Figure 2.10.4. TA0MR to TA4MR Registers

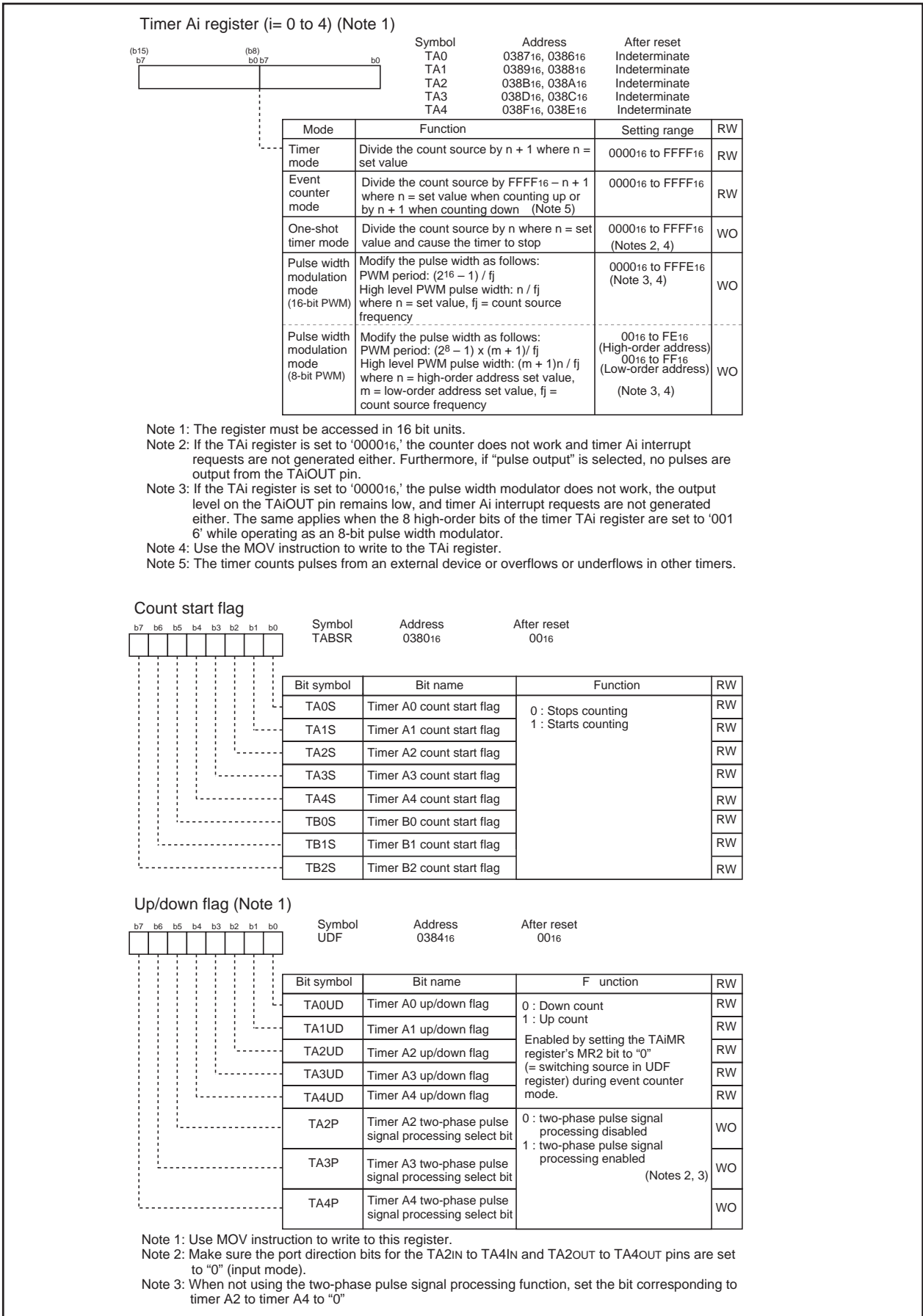


Figure 2.10.5. TA0 to TA4 Registers, TABSR Register, and UDF Register

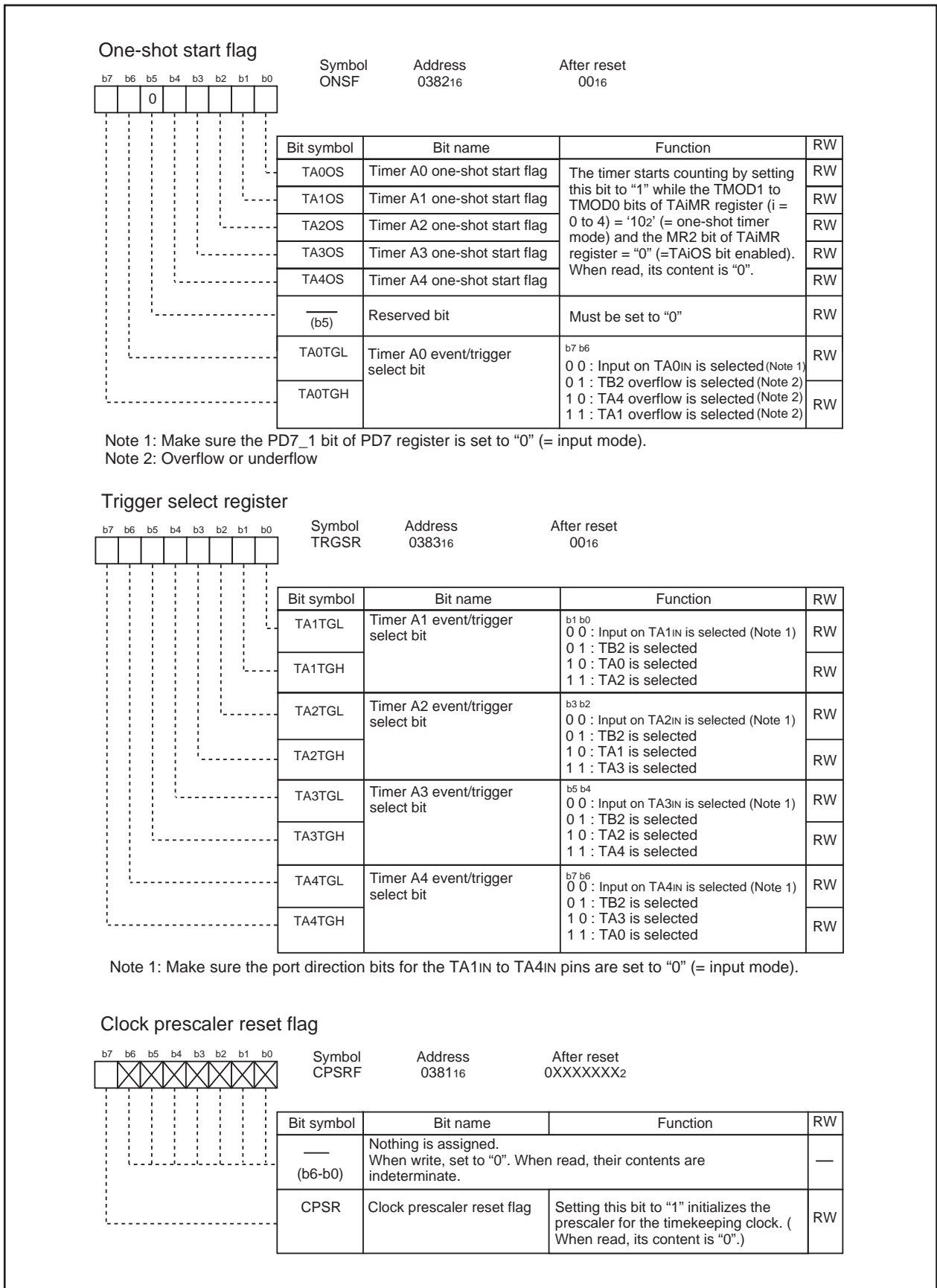


Figure 2.10.6. ONSF Register, TRGSR Register, and CPSRF Register

(1) Timer Mode

In timer mode, the timer counts a count source generated internally (see Table 2.10.1). Figure 2.10.7 shows TAI_iMR register in timer mode.

Table 2.10.1. Specifications in Timer Mode

Item	Specification
Count source	f ₁ , f ₂ , f ₈ , f ₃₂ , f _{C32}
Count operation	<ul style="list-style-type: none"> Down-count When the timer underflows, it reloads the reload register contents and continues counting
Divide ratio	1/(n+1) n: set value of TAI register (i= 0 to 4) 0000 ₁₆ to FFFF ₁₆
Count start condition	Set TAI _S bit of TABSR register to "1" (= start counting)
Count stop condition	Set TAI _S bit to "0" (= stop counting)
Interrupt request generation timing	Timer underflow
TAI _i N pin function	I/O port or gate input
TAI _i OUT pin function	I/O port or pulse output
Read from timer	Count value can be read by reading TAI register
Write to timer	<ul style="list-style-type: none"> When not counting and until the 1st count source is input after counting start Value written to TAI register is written to both reload register and counter When counting (after 1st count source input) Value written to TAI register is written to only reload register (Transferred to counter when reloaded next)
Select function	<ul style="list-style-type: none"> Gate function Counting can be started and stopped by an input signal to TAI_iN pin Pulse output function Whenever the timer underflows, the output polarity of TAI_iOUT pin is inverted. When not counting, the pin outputs a low.

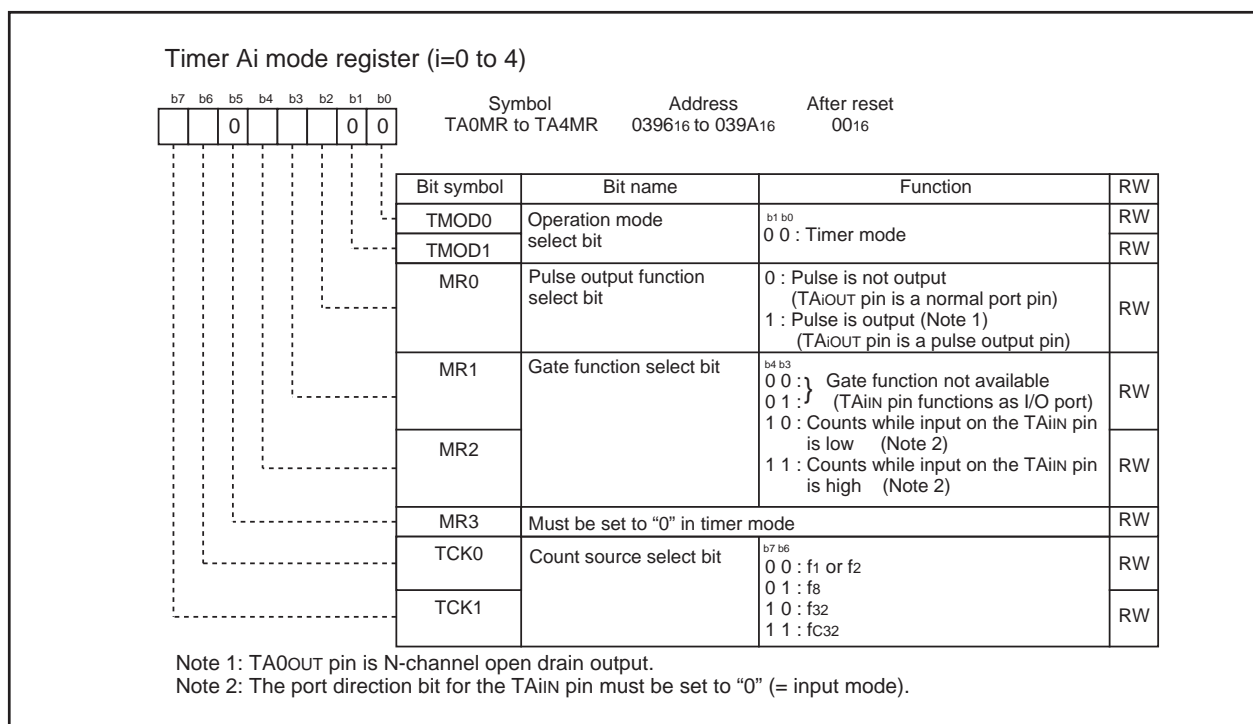


Figure 2.10.7. Timer Ai Mode Register in Timer Mode

(2) Event Counter Mode

In event counter mode, the timer counts pulses from an external device or overflows and underflows of other timers. Timers A2, A3 and A4 can count two-phase external signals. Table 2.10.2 lists specifications in event counter mode (when not processing two-phase pulse signal). Table 2.10.3 lists specifications in event counter mode (when processing two-phase pulse signal with the timers A2, A3 and A4). Figure 2.10.8 shows TAI_iMR register in event counter mode (when not processing two-phase pulse signal). Figure 2.10.9 shows TA2MR to TA4MR registers in event counter mode (when processing two-phase pulse signal with the timers A2, A3 and A4).

Table 2.10.2. Specifications in Event Counter Mode (when not processing two-phase pulse signal)

Item	Specification
Count source	<ul style="list-style-type: none"> External signals input to TAI_in pin (i=0 to 4) (effective edge can be selected in program) Timer B2 overflows or underflows, timer A_j (j=i-1, except j=4 if i=0) overflows or underflows, timer A_k (k=i+1, except k=0 if i=4) overflows or underflows
Count operation	<ul style="list-style-type: none"> Up-count or down-count can be selected by external signal or program When the timer overflows or underflows, it reloads the reload register contents and continues counting. When operating in free-running mode, the timer continues counting without reloading.
Divided ratio	1/ (FFFF ₁₆ - n + 1) for up-count 1/ (n + 1) for down-count n : set value of TAI register 0000 ₁₆ to FFFF ₁₆
Count start condition	Set TAI _i S bit of TABSR register to "1" (= start counting)
Count stop condition	Set TAI _i S bit to "0" (= stop counting)
Interrupt request generation timing	Timer overflow or underflow
TAI _i n pin function	I/O port or count source input
TAI _i OUT pin function	I/O port, pulse output, or up/down-count select input
Read from timer	Count value can be read by reading TAI register
Write to timer	<ul style="list-style-type: none"> When not counting and until the 1st count source is input after counting start Value written to TAI register is written to both reload register and counter When counting (after 1st count source input) Value written to TAI register is written to only reload register (Transferred to counter when reloaded next)
Select function	<ul style="list-style-type: none"> Free-run count function Even when the timer overflows or underflows, the reload register content is not reloaded to it Pulse output function Whenever the timer overflows or underflows, the output polarity of TAI_iOUT pin is inverted . When not counting, the pin outputs a low.

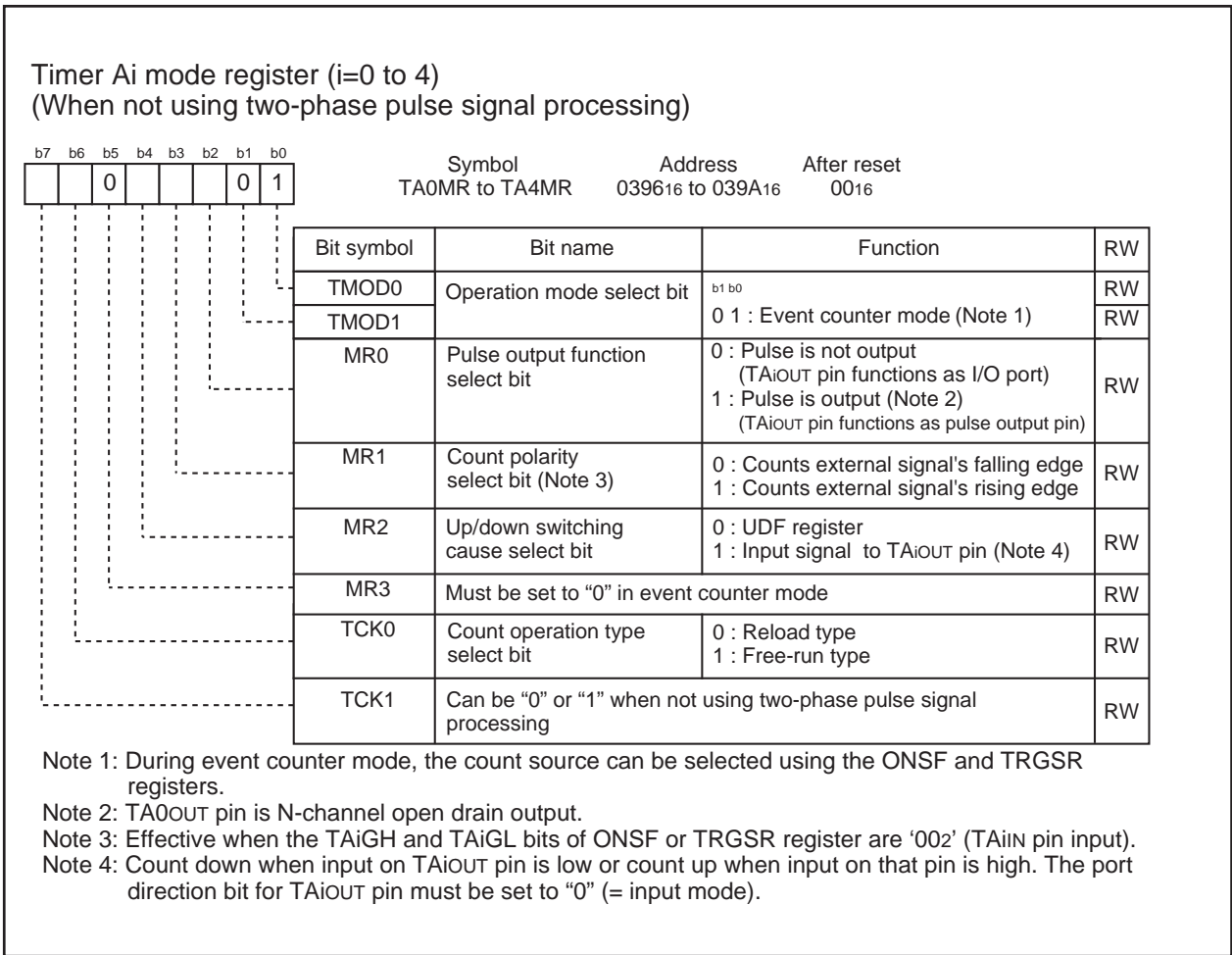
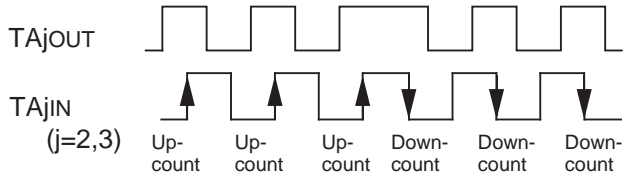
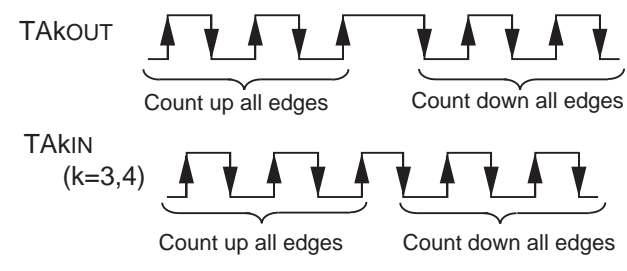


Figure 2.10.8. TAiMR Register in Event Counter Mode (when not using two-phase pulse signal processing)

Table 2.10.3. Specifications in Event Counter Mode (when processing two-phase pulse signal with timers A2, A3 and A4)

Item	Specification
Count source	<ul style="list-style-type: none"> Two-phase pulse signals input to TAIIN or TAIOUT pins (i = 2 to 4)
Count operation	<ul style="list-style-type: none"> Up-count or down-count can be selected by two-phase pulse signal When the timer overflows or underflows, it reloads the reload register contents and continues counting. When operating in free-running mode, the timer continues counting without reloading.
Divide ratio	$1 / (FFFF_{16} - n + 1)$ for up-count $1 / (n + 1)$ for down-count n : set value of TAI register 0000_{16} to $FFFF_{16}$
Count start condition	Set TAI _S bit of TABSR register to "1" (= start counting)
Count stop condition	Set TAI _S bit to "0" (= stop counting)
Interrupt request generation timing	Timer overflow or underflow
TAIIN pin function	Two-phase pulse input
TAIOUT pin function	Two-phase pulse input
Read from timer	Count value can be read by reading timer A2, A3 or A4 register
Write to timer	<ul style="list-style-type: none"> When not counting and until the 1st count source is input after counting start Value written to TAI register is written to both reload register and counter When counting (after 1st count source input) Value written to TAI register is written to reload register (Transferred to counter when reloaded next)
Select function (Note)	<ul style="list-style-type: none"> Normal processing operation (timer A2 and timer A3) The timer counts up rising edges or counts down falling edges on TAJIN pin when input signals on TAJOUT pin is "H".  <ul style="list-style-type: none"> Multiply-by-4 processing operation (timer A3 and timer A4) If the phase relationship is such that TAKIN(k=3, 4) pin goes "H" when the input signal on TAKOUT pin is "H", the timer counts up rising and falling edges on TAKOUT and TAKIN pins. If the phase relationship is such that TAKIN pin goes "L" when the input signal on TAKOUT pin is "H", the timer counts down rising and falling edges on TAKOUT and TAKIN pins. 

Notes:

- Only timer A3 is selectable. Timer A2 is fixed to normal processing operation, and timer A4 is fixed to multiply-by-4 processing operation.

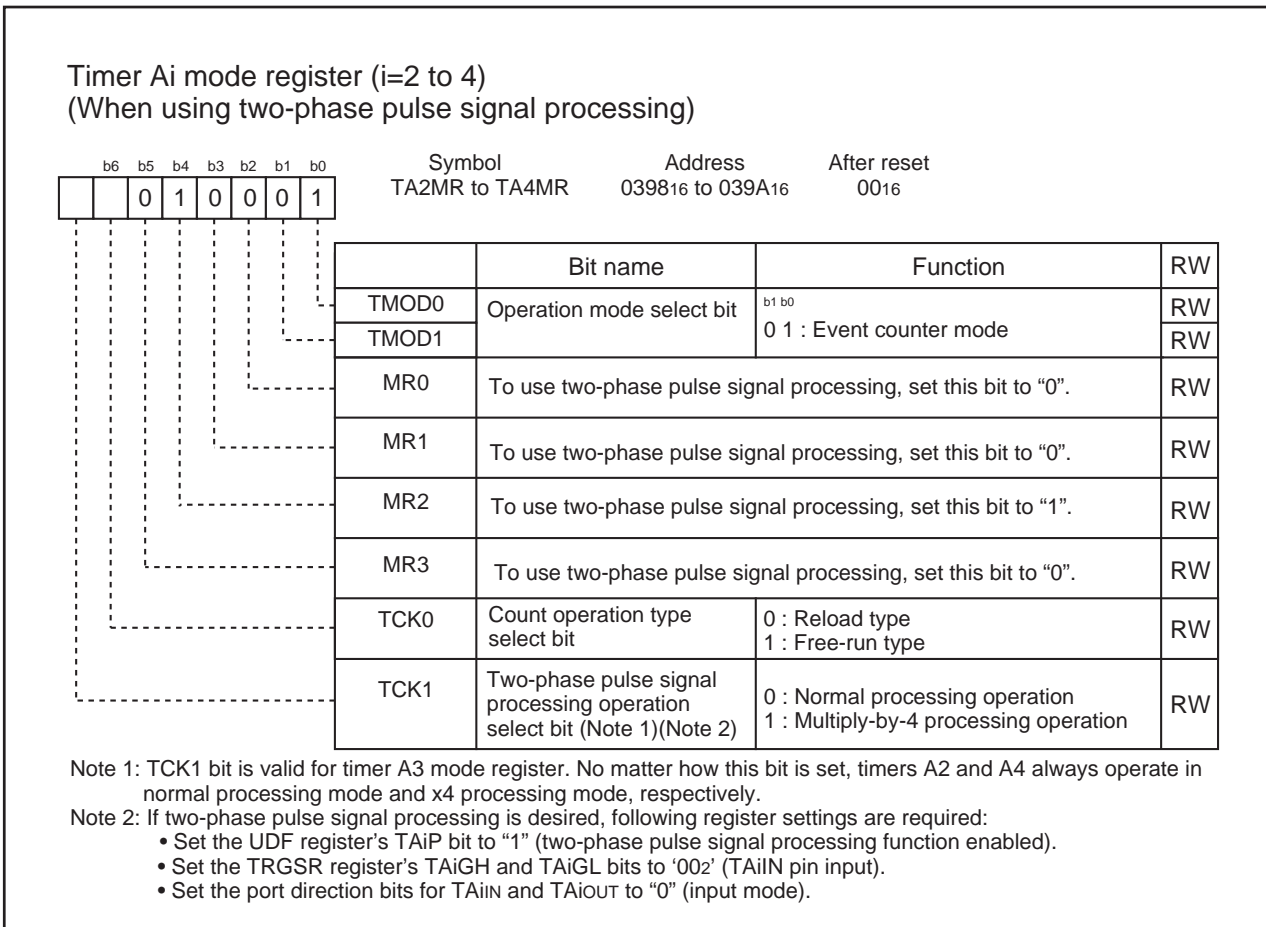


Figure 2.10.9. TA2MR to TA4MR Registers in Event Counter Mode (when using two-phase pulse signal processing with timer A2, A3 or A4)

(3) One-shot Timer Mode

In one-shot timer mode, the timer is activated only once by one trigger. (See Table 2.10.4.) When the trigger occurs, the timer starts up and continues operating for a given period. Figure 2.10.10 shows the TAI_{MR} register in one-shot timer mode.

Table 2.10.4. Specifications in One-shot Timer Mode

Item	Specification
Count source	f1, f2, f8, f32, fC32
Count operation	<ul style="list-style-type: none"> Down-count When the counter reaches 0000₁₆, it stops counting after reloading a new value If a trigger occurs when counting, the timer reloads a new count and restarts counting
Divide ratio	$1/n$ n : set value of TAI register 0000 ₁₆ to FFFF ₁₆ However, the counter does not work if the divide-by-n value is set to 0000 ₁₆ .
Count start condition	TAI _S bit of TABSR register = "1" (start counting) and one of the following triggers occurs. <ul style="list-style-type: none"> External trigger input from the TAI_{IN} pin Timer B2 overflow or underflow, timer A_j (j=i-1, except j=4 if i=0) overflow or underflow, timer A_k (k=i+1, except k=0 if i=4) overflow or underflow The TAI_{OS} bit of ONSF register is set to "1" (= timer starts)
Count stop condition	<ul style="list-style-type: none"> When the counter is reloaded after reaching "0000₁₆" TAI_S bit is set to "0" (= stop counting)
Interrupt request generation timing	When the counter reaches "0000 ₁₆ "
TAI _{IN} pin function	I/O port or trigger input
TAI _{OUT} pin function	I/O port or pulse output
Read from timer	An indeterminate value is read by reading TAI register
Write to timer	<ul style="list-style-type: none"> When not counting and until the 1st count source is input after counting start Value written to TAI register is written to both reload register and counter When counting (after 1st count source input) Value written to TAI register is written to only reload register (Transferred to counter when reloaded next)
Select function	<ul style="list-style-type: none"> Pulse output function The timer outputs a low when not counting and a high when counting.

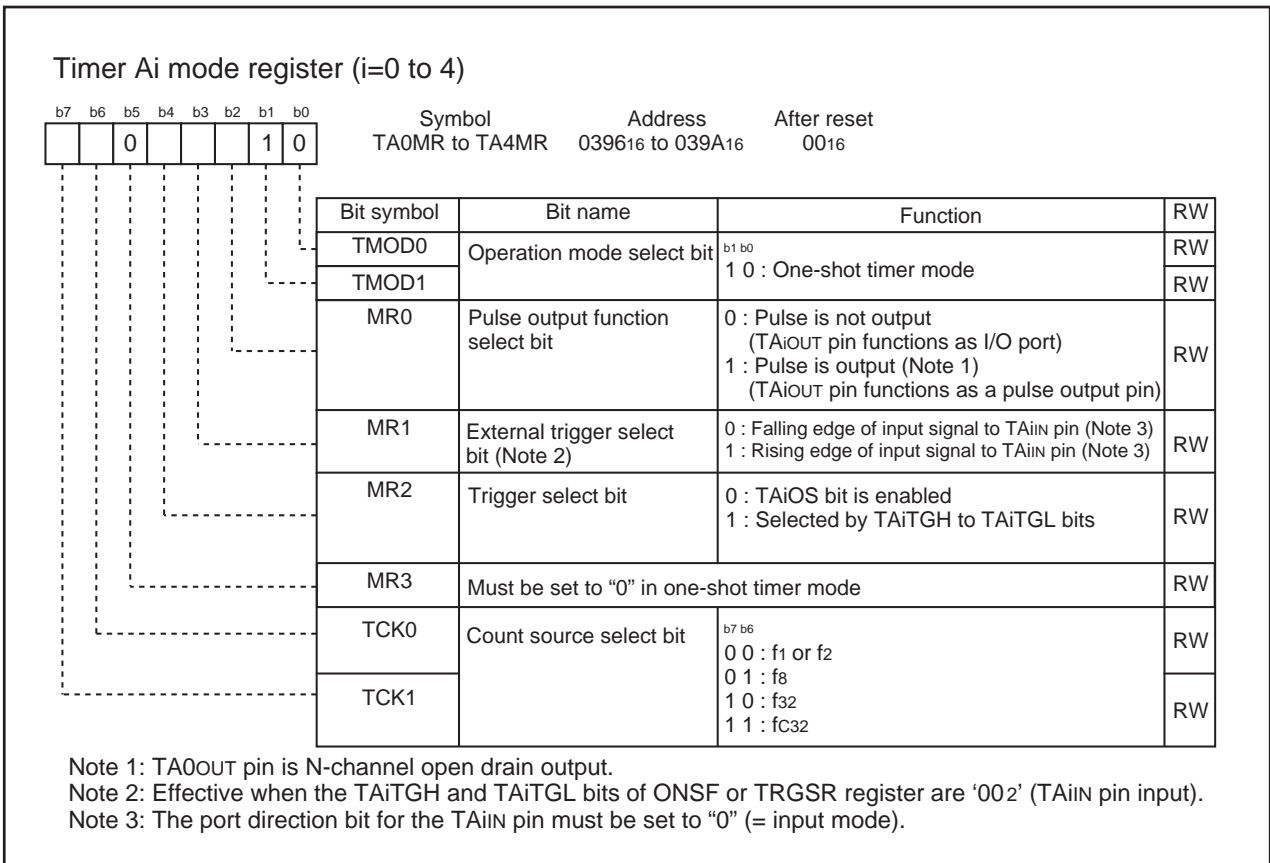


Figure 2.10.10. TAI_iMR Register in One-shot Timer Mode

(4) Pulse Width Modulation (PWM) Mode

In PWM mode, the timer outputs pulses of a given width in succession (see Table 2.10.5). The counter functions as either 16-bit pulse width modulator or 8-bit pulse width modulator. Figure 2.10.11 shows TAI_{MR} register in pulse width modulation mode. Figures 2.10.12 and 2.10.13 show examples of how a 16-bit pulse width modulator operates and how an 8-bit pulse width modulator operates.

Table 2.10.5. Specifications in PWM Mode

Item	Specification
Count source	f ₁ , f ₂ , f ₈ , f ₃₂ , f _{C32}
Count operation	<ul style="list-style-type: none"> • Down-count (operating as an 8-bit or a 16-bit pulse width modulator) • The timer reloads a new value at a rising edge of PWM pulse and continues counting • The timer is not affected by a trigger that occurs during counting
16-bit PWM	<ul style="list-style-type: none"> • High level width n / f_j n : set value of TAI register (i=0 to 4) • Cycle time $(2^{16}-1) / f_j$ fixed f_j: count source frequency (f₁, f₂, f₈, f₃₂, f_{C32})
8-bit PWM	<ul style="list-style-type: none"> • High level width $n \times (m+1) / f_j$ n : set value of TAI register high-order address • Cycle time $(2^8-1) \times (m+1) / f_j$ m : set value of TAI register low-order address
Count start condition	<ul style="list-style-type: none"> • TAI_S bit of TABSR register is set to "1" (= start counting) • The TAI_S bit = 1 and external trigger input from the TAI_{IN} pin • The TAI_S bit = 1 and one of the following external triggers occurs • Timer B2 overflow or underflow, timer A_j (j=i-1, except j=4 if i=0) overflow or underflow, timer A_k (k=i+1, except k=0 if i=4) overflow or underflow
Count stop condition	TAI _S bit is set to "0" (= stop counting)
Interrupt request generation timing	PWM pulse goes "L"
TAI _{IN} pin function	I/O port or trigger input
TAI _{OUT} pin function	Pulse output
Read from timer	An indeterminate value is read by reading TAI register
Write to timer	<ul style="list-style-type: none"> • When not counting and until the 1st count source is input after counting start Value written to TAI register is written to both reload register and counter • When counting (after 1st count source input) Value written to TAI register is written to only reload register (Transferred to counter when reloaded next)

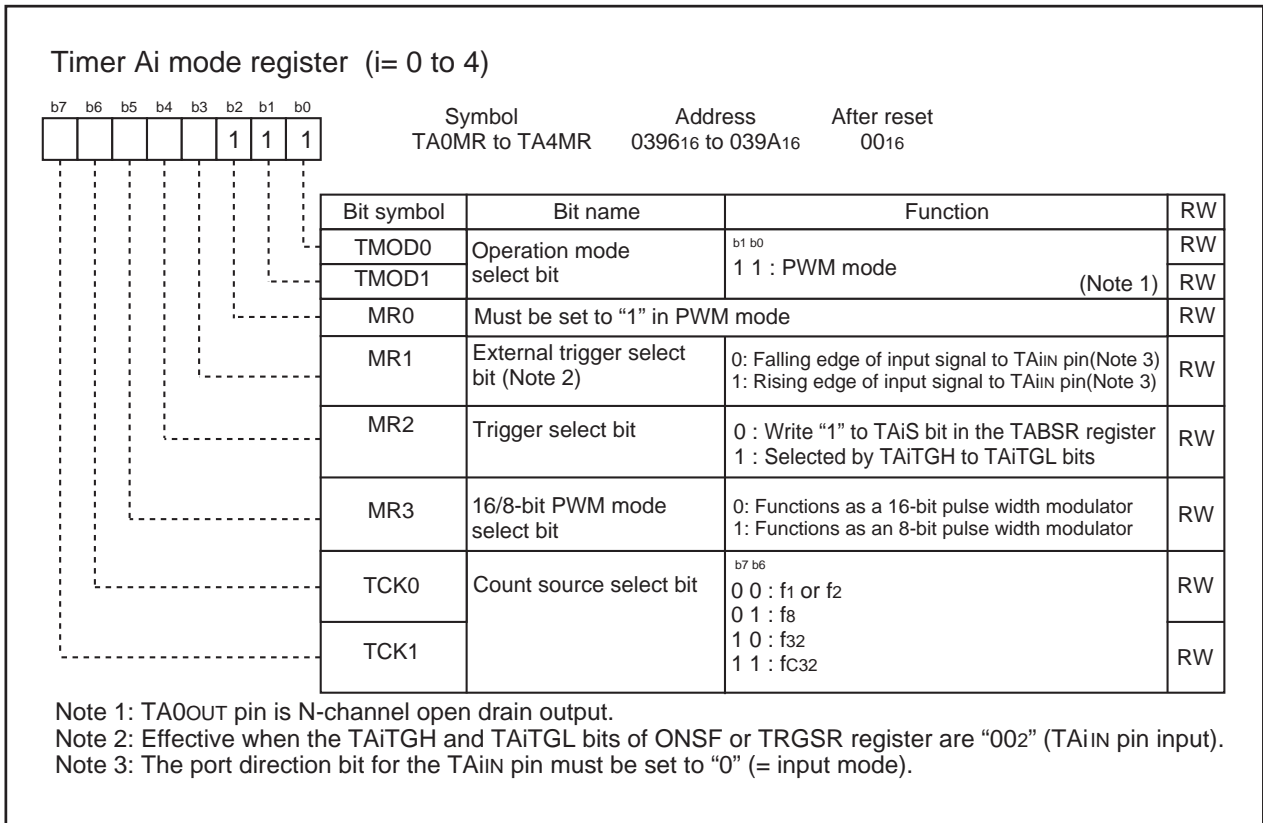


Figure 2.10.11. TAIiMR Register in PWM Mode

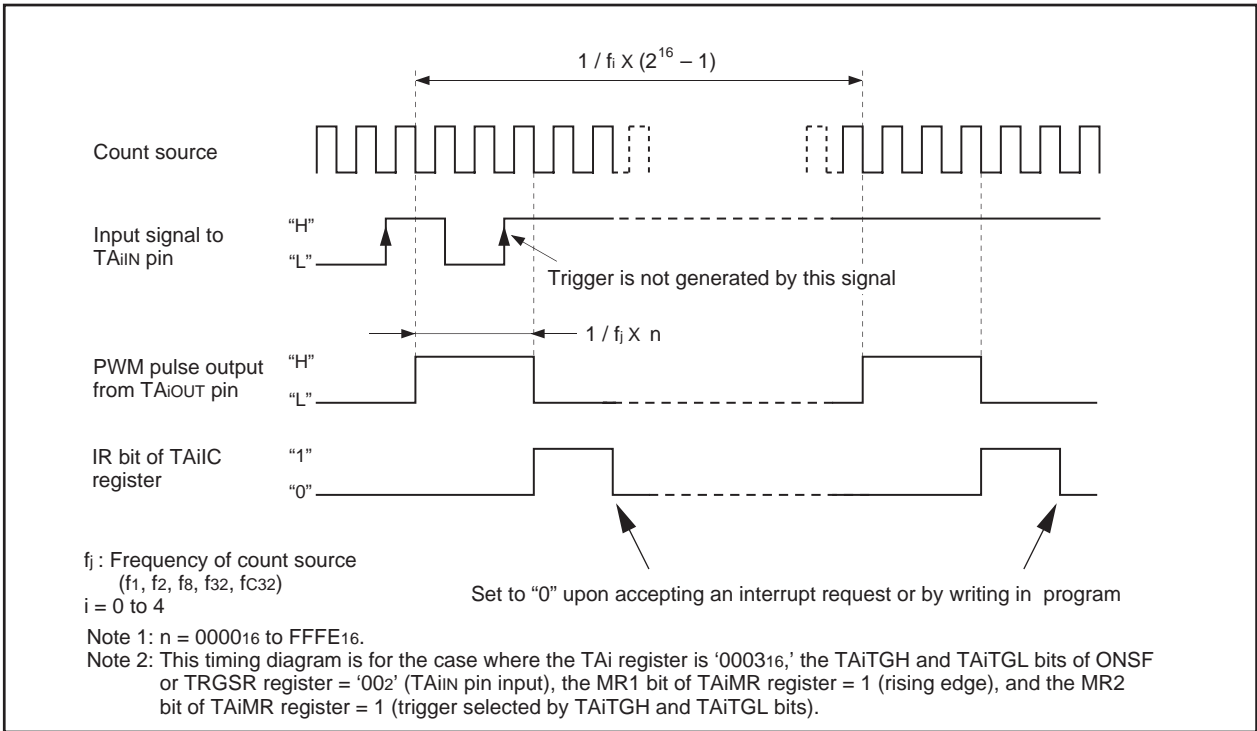


Figure 2.10.12. Example of 16-bit Pulse Width Modulator Operation

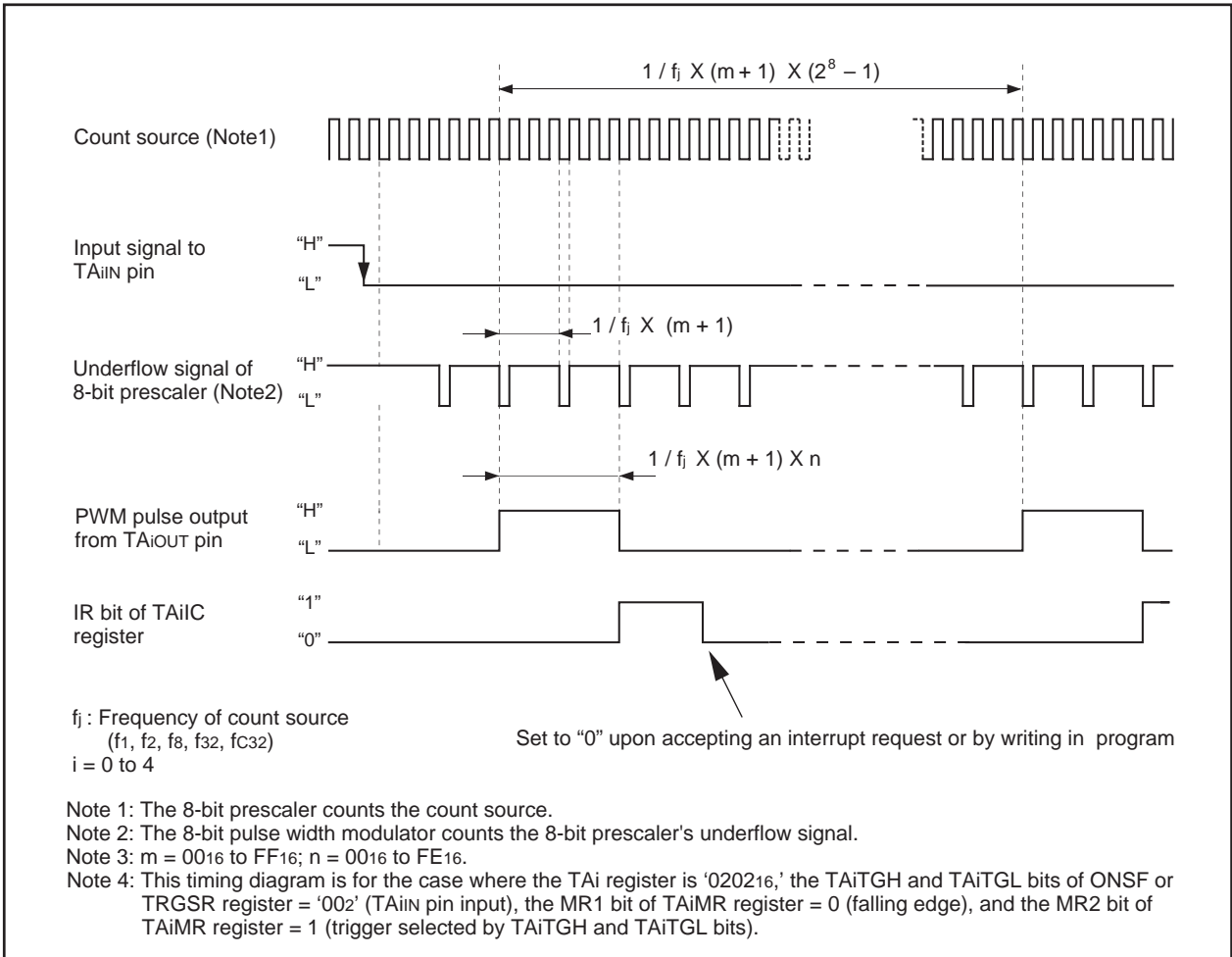


Figure 2.10.13. Example of 8-bit Pulse Width Modulator Operation

2.10.2 Timer B

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

Timer B supports the following three modes. Use the TMOD1 and TMOD0 bits of TBiMR register (i = 0 to 5) to select the desired mode.

- Timer mode: The timer counts an internal count source.
- Event counter mode: The timer counts pulses from an external device or overflows or underflows of other timers.
- Pulse period/pulse width measuring mode: The timer measures an external signal's pulse period or pulse width.

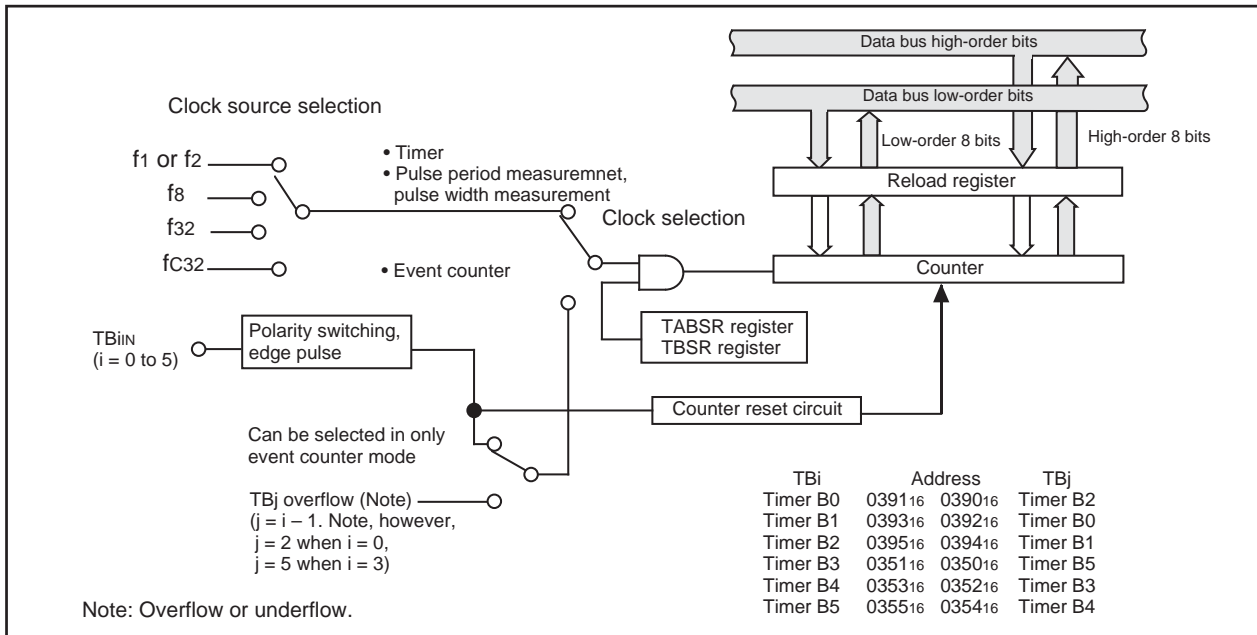


Figure 2.10.14. Timer B Block Diagram

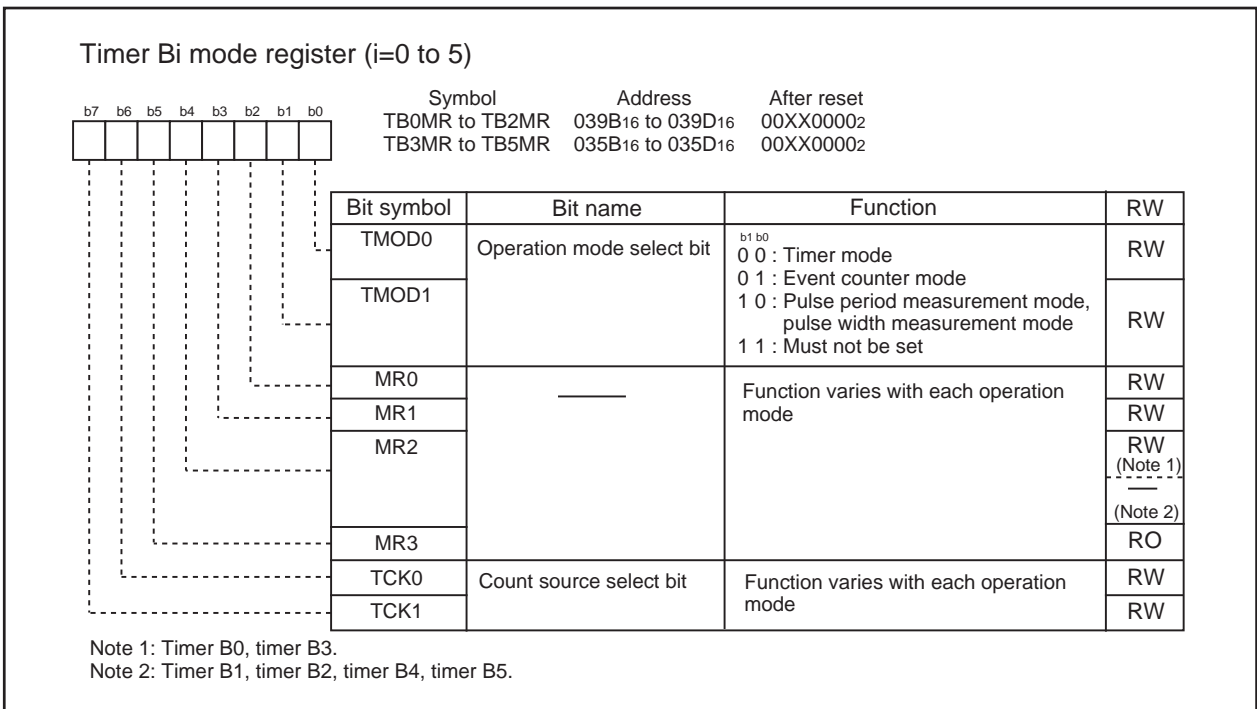


Figure 2.10.15. TB0MR to TB5MR Registers

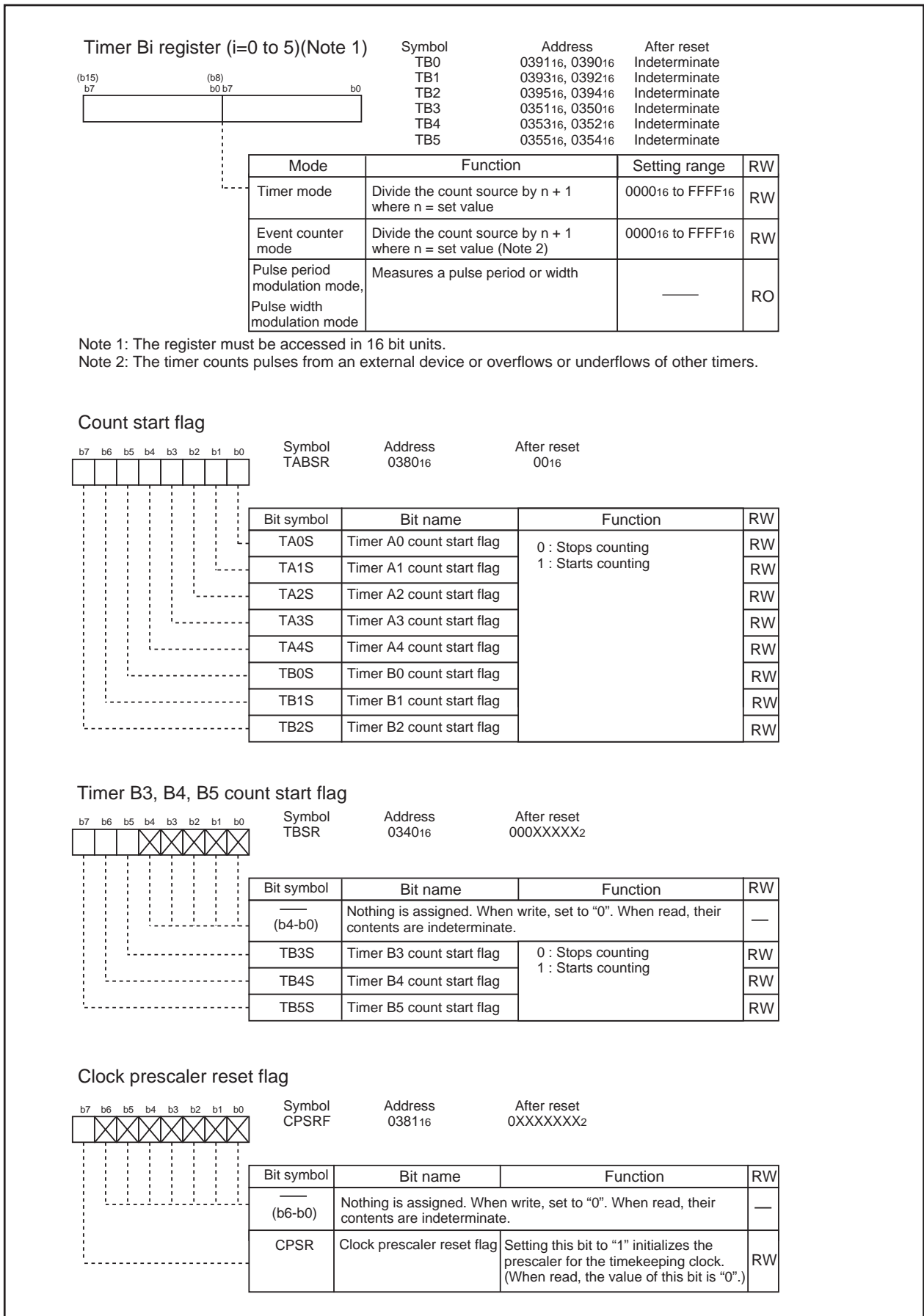


Figure 2.10.16. TB0 to TB5 Registers, TABSR Register, TBSR Register, CPSRF Register

(1) Timer Mode

In timer mode, the timer counts a count source generated internally (see Table 2.10.6). Figure 2.10.17 shows TBiMR register in timer mode.

Table 2.10.6. Specifications in Timer Mode

Item	Specification
Count source	f1, f2, f8, f32, fc32
Count operation	<ul style="list-style-type: none"> Down-count When the timer underflows, it reloads the reload register contents and continues counting
Divide ratio	1/(n+1) n: set value of TBi register (i= 0 to 5) 0000 ₁₆ to FFFF ₁₆
Count start condition	Set TBiS bit ^(Note) to "1" (= start counting)
Count stop condition	Set TBiS bit to "0" (= stop counting)
Interrupt request generation timing	Timer underflow
TBiIN pin function	I/O port
Read from timer	Count value can be read by reading TBi register
Write to timer	<ul style="list-style-type: none"> When not counting and until the 1st count source is input after counting start Value written to TBi register is written to both reload register and counter When counting (after 1st count source input) Value written to TBi register is written to only reload register (Transferred to counter when reloaded next)

Note : The TB0S to TB2S bits are assigned to the TABSR register bit 5 to bit 7, and the TB3S to TB5S bits are assigned to the TBSR register bit 5 to bit 7.

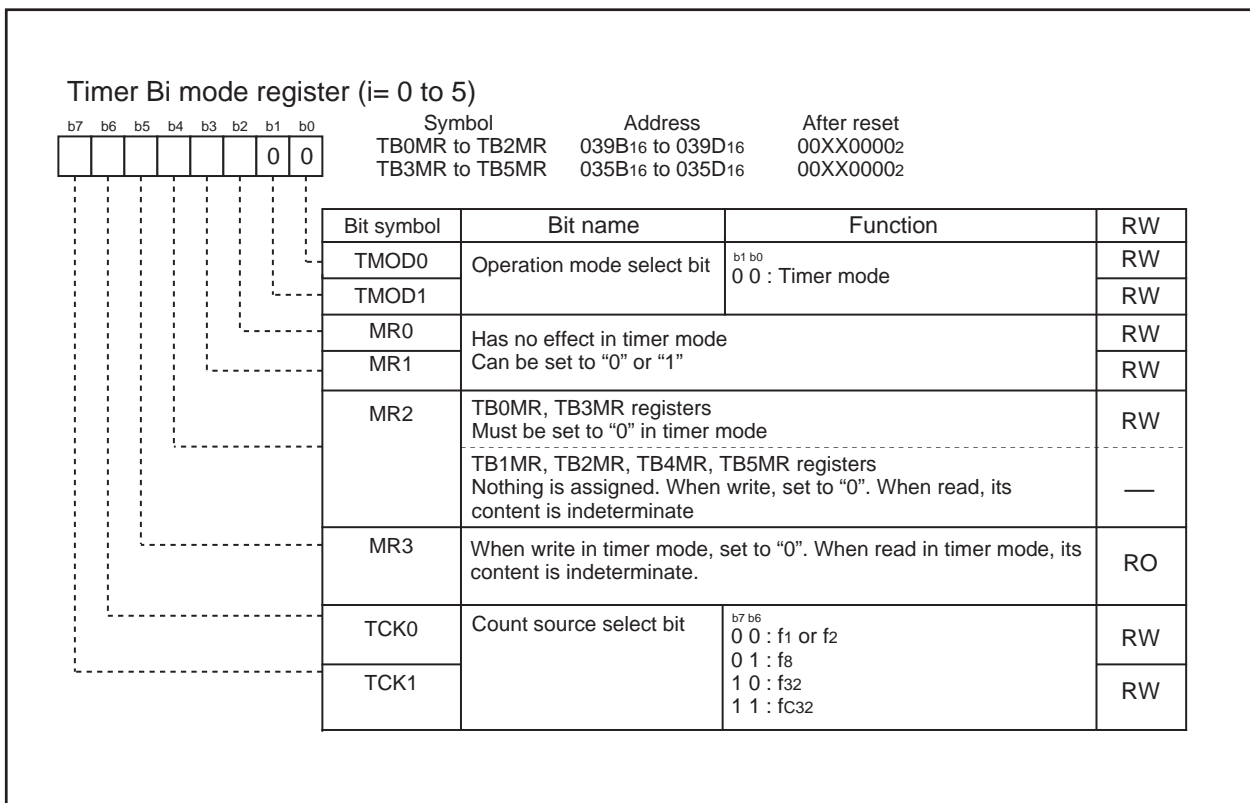


Figure 2.10.17. TBiMR Register in Timer Mode

(2) Event Counter Mode

In event counter mode, the timer counts pulses from an external device or overflows and underflows of other timers (see Table 2.10.7) . Figure 2.10.20 shows TBiMR register in event counter mode.

Table 2.10.7. Specifications in Event Counter Mode

Item	Specification
Count source	<ul style="list-style-type: none"> External signals input to TBIIN pin (i=0 to 5) (effective edge can be selected in program) Timer Bj overflow or underflow (j=i-1, except j=2 if i=0, j=5 if i=3)
Count operation	<ul style="list-style-type: none"> Down-count When the timer underflows, it reloads the reload register contents and continues counting
Divide ratio	1/(n+1) n: set value of TBi register 0000 ₁₆ to FFFF ₁₆
Count start condition	Set TBiS bit ¹ to "1" (= start counting)
Count stop condition	Set TBiS bit to "0" (= stop counting)
Interrupt request generation timing	Timer underflow
TBIIN pin function	Count source input
Read from timer	Count value can be read by reading TBi register
Write to timer	<ul style="list-style-type: none"> When not counting and until the 1st count source is input after counting start Value written to TBi register is written to both reload register and counter When counting (after 1st count source input) Value written to TBi register is written to only reload register (Transferred to counter when reloaded next)

Notes:

- The TB0S to TB2S bits are assigned to the TABSR register bit 5 to bit 7, and the TB3S to TB5S bits are assigned to the TBSR register bit 5 to bit 7.

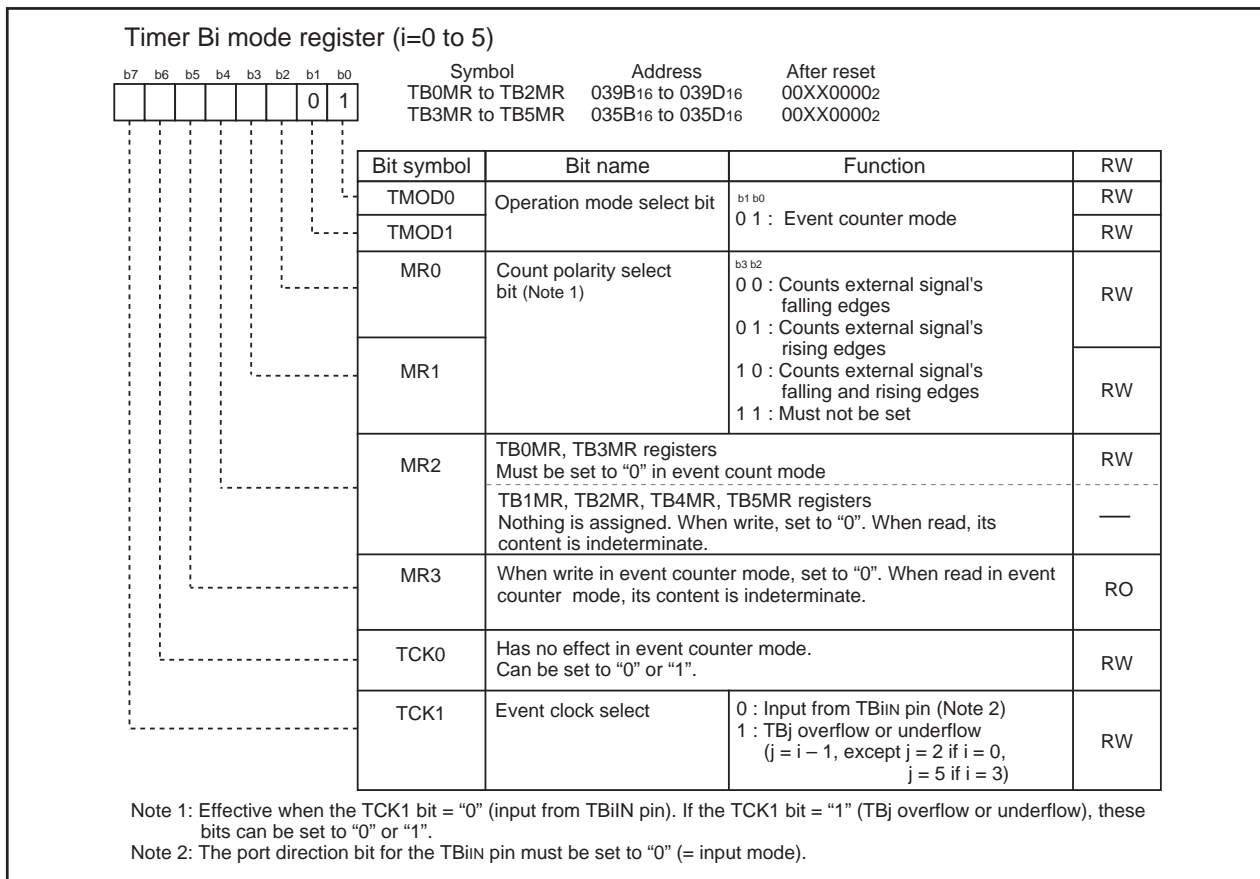


Figure 2.10.20. TBiMR Register in Event Counter Mode

(3) Pulse Period and Pulse Width Measurement Mode

In pulse period and pulse width measurement mode, the timer measures pulse period¹ or pulse width of an external signal (see Table 2.10.8). Figure 2.10.21 shows TBiMR register in pulse period and pulse width measurement mode. Figure 2.10.22 shows the operation timing when measuring a pulse period. Figure 2.10.23 shows the operation timing when measuring a pulse width.

Table 2.10.8. Specifications in Pulse Period and Pulse Width Measurement Mode

Item	Specification
Count source	f1, f2, f8, f32, fc32
Count operation	<ul style="list-style-type: none"> Up-count Counter value is transferred to reload register at an effective edge of measurement pulse. The counter value is set to "000016" to continue counting.
Count start condition	Set TBiS (i=0 to 5) bit ³ to "1" (= start counting)
Count stop condition	Set TBiS bit to "0" (= stop counting)
Interrupt request generation timing	<ul style="list-style-type: none"> When an effective edge of measurement pulse is input¹ Timer overflow. When an overflow occurs, MR3 bit of TBiMR register is set to "1" (overflowed) simultaneously. MR3 bit is cleared to "0" (no overflow) by writing to TBiMR register at the next count timing or later after MR3 bit was set to "1". At this time, make sure TBiS bit is set to "1" (start counting).
TBiIN pin function	Measurement pulse input
Read from timer	Contents of the reload register (measurement result) can be read by reading TBi register ²
Write to timer	Value written to TBi register is written to neither reload register nor counter

Notes:

- Interrupt request is not generated when the first effective edge is input after the timer started counting.
- Value read from TBi register is indeterminate until the second valid edge is input after the timer starts counting.
- The TB0S to TB2S bits are assigned to the TABSR register bit 5 to bit 7, and the TB3S to TB5S bits are assigned to the TBSR register bit 5 to bit 7.

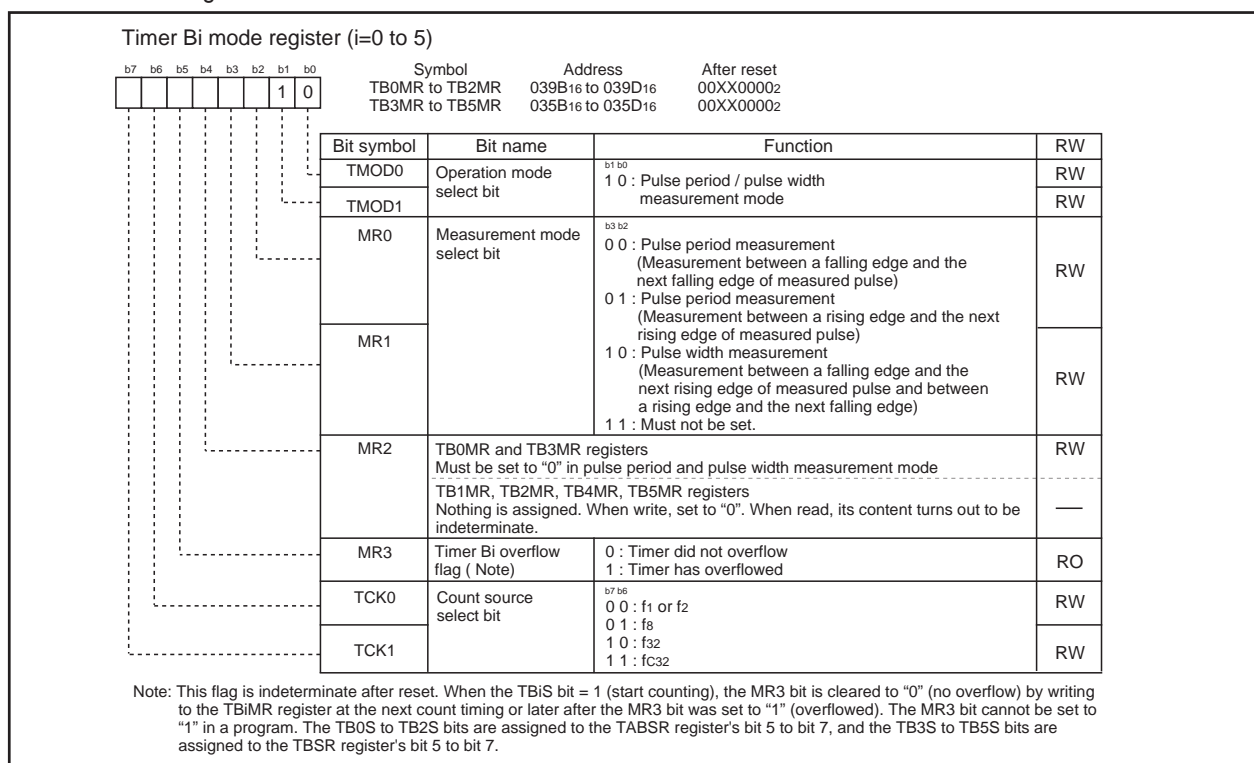


Figure 2.10.21. TBiMR Register in Pulse Period and Pulse Width Measurement Mode

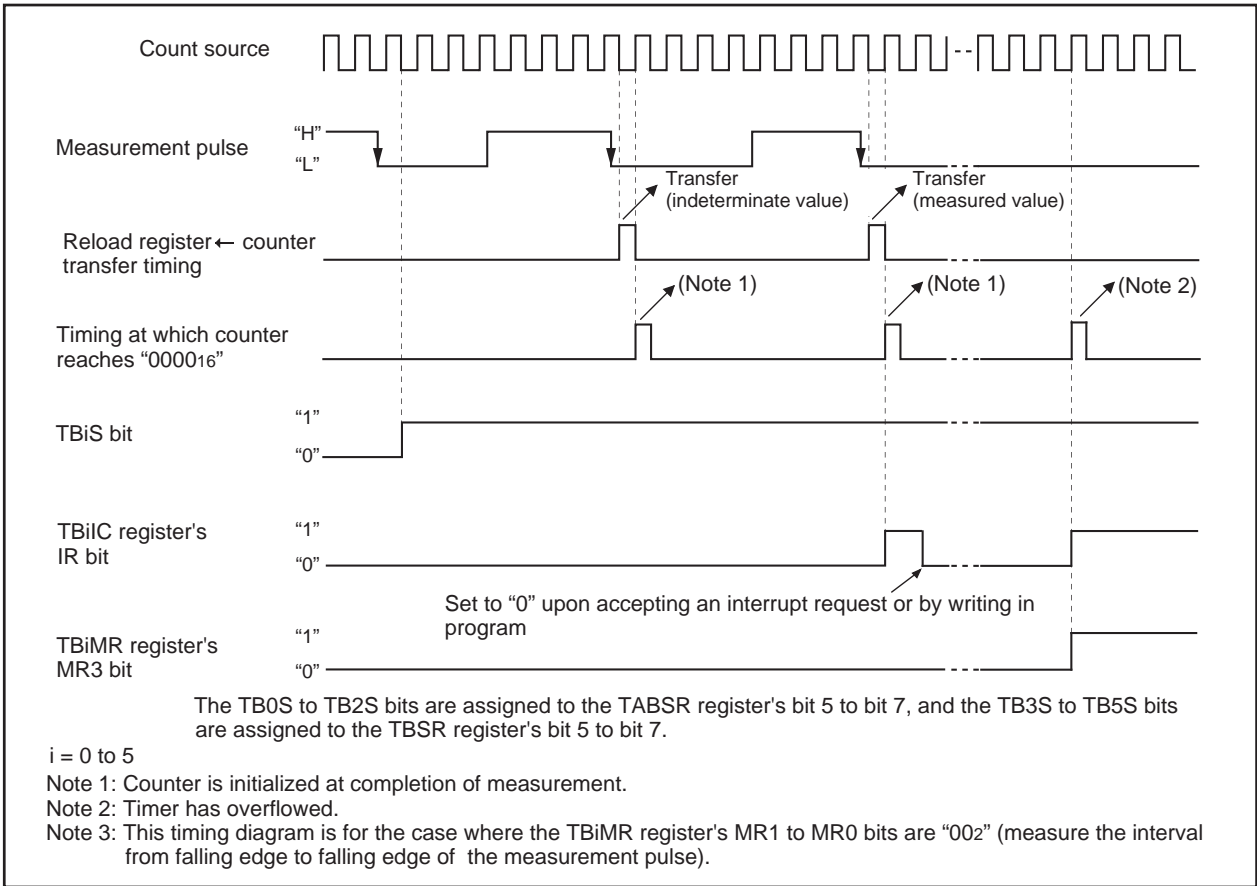


Figure 2.10.22. Operation timing when measuring a pulse period

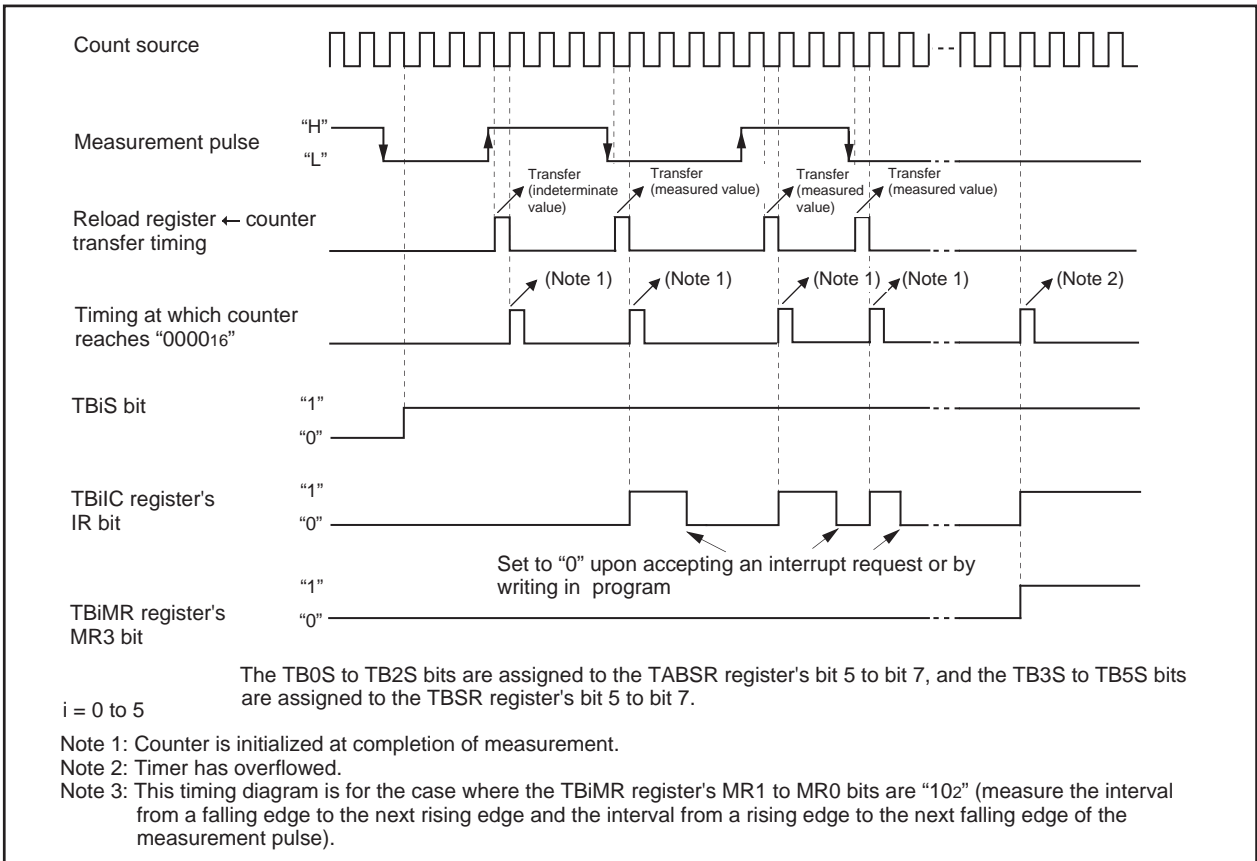


Figure 2.10.23. Operation timing when measuring a pulse width

2.11 Serial I/O

Serial I/O is configured with five channels: UART0 to UART2, SI/O3 and SI/O4.

2.11.1 UARTi (i=0 to 2)

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

Figure 2.11.1 shows the block diagram of UARTi. Figures 2.11.2 shows the block diagram of the UARTi transmit/receive.

UARTi has the following modes:

- Clock synchronous serial I/O mode
- Clock asynchronous serial I/O mode (UART mode).
- Special mode 1 (I²C mode)
- Special mode 2
- Special mode 3 (Bus collision detection function, IE mode) : UART0, UART1
- Special mode 4 (SIM mode) : UART2

Figures 2.11.3 to 2.11.8 show the UARTi-related registers.

Refer to tables listing each mode for register setting.

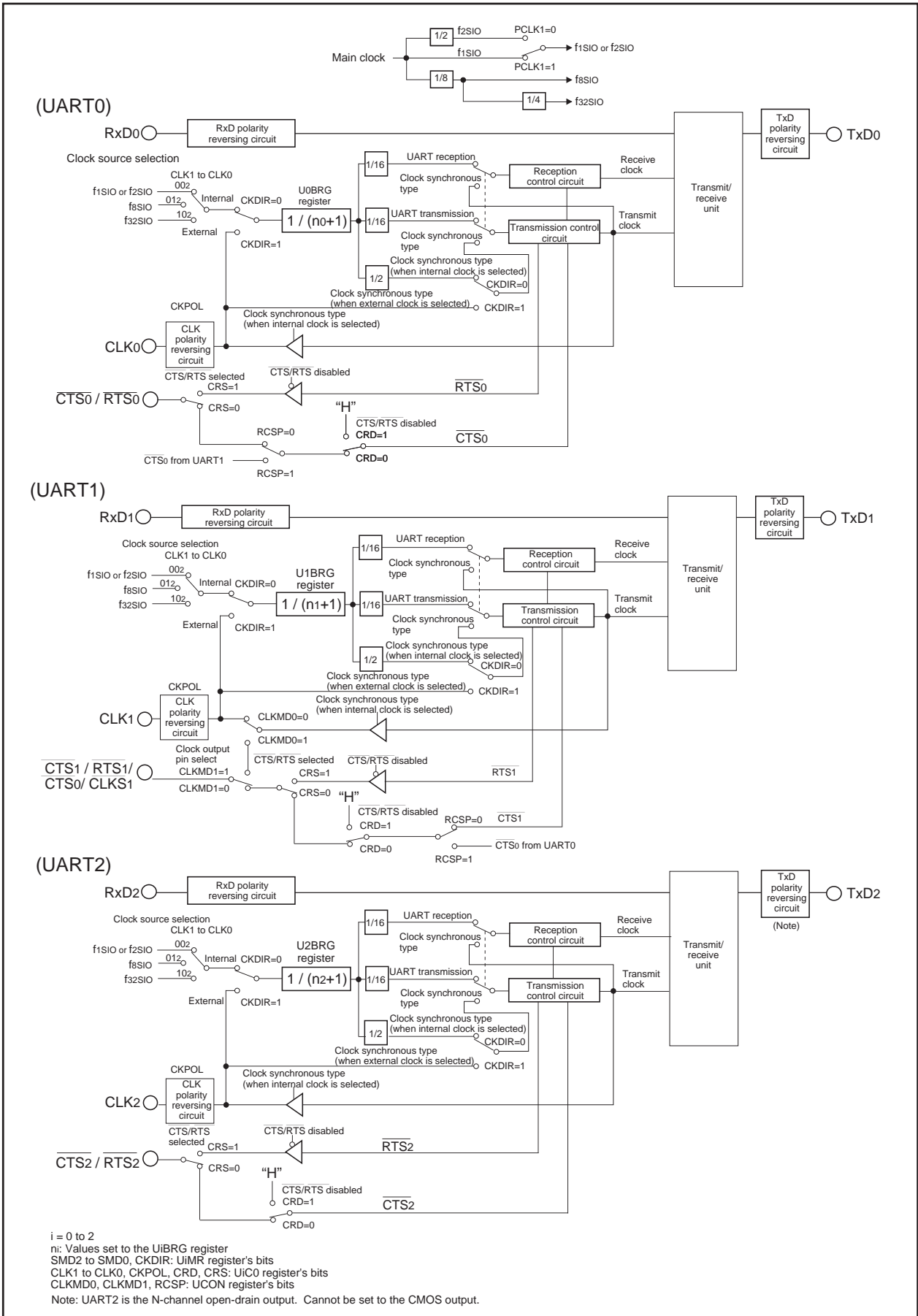


Figure 2.11.1. UARTi Block Diagram

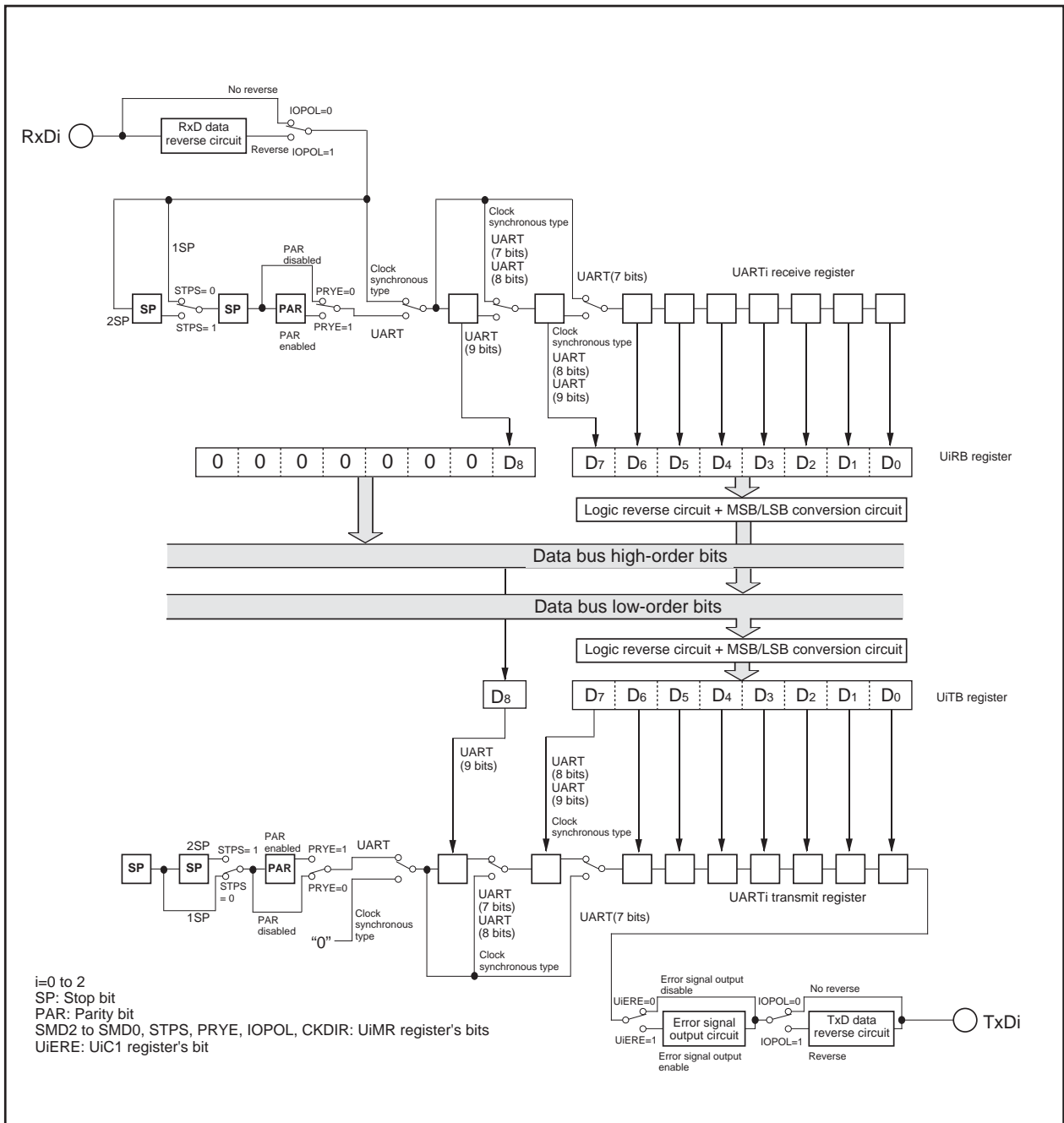


Figure 2.11.2. UARTi Transmit/Receive Unit

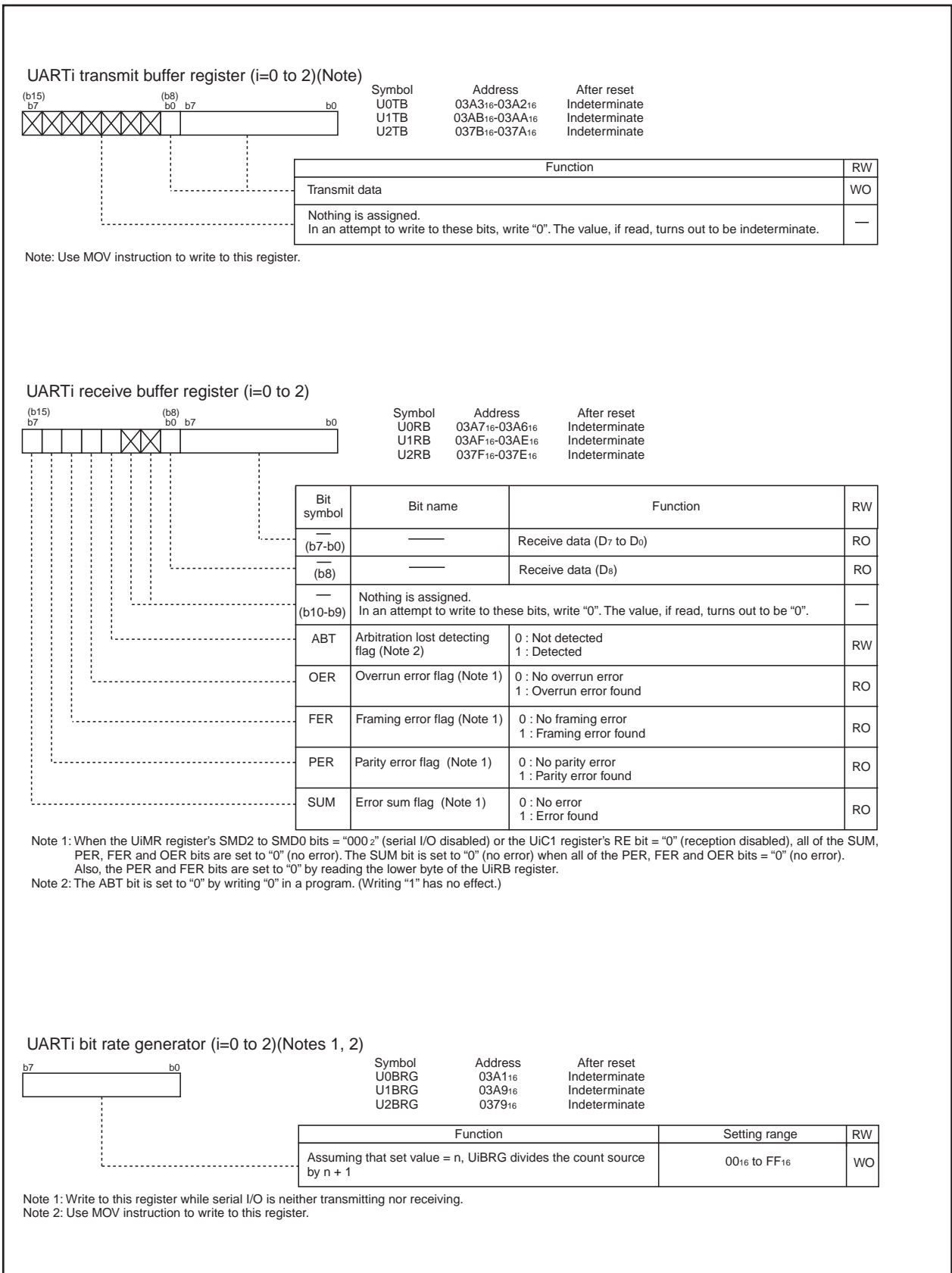


Figure 2.11.3. U0TB to U2TB Register, U0RB to U2RB Register, and U0BRG to U2BRG Register

UARTi transmit/receive mode register (i=0 to 2)

b7	b6	b5	b4	b3	b2	b1	b0	Symbol	Address	After reset
								U0MR to U2MR	03A0 ₁₆ , 03A8 ₁₆ , 0378 ₁₆	00 ₁₆

Bit symbol	Bit name	Function	RW
SMD0	Serial I/O mode select bit (Note 2)	^{b2 b1 b0} 0 0 0 : Serial I/O disabled 0 0 1 : Clock synchronous serial I/O mode 0 1 0 : I ² C mode (Note 3) 1 0 0 : UART mode transfer data 7 bits long 1 0 1 : UART mode transfer data 8 bits long 1 1 0 : UART mode transfer data 9 bits long Must not be set except above	RW
SMD1			RW
SMD2			RW
CKDIR	Internal/external clock select bit	0 : Internal clock 1 : External clock (Note 1)	RW
STPS	Stop bit length select bit	0 : One stop bit 1 : Two stop bits	RW
PRY	Odd/even parity select bit	Effective when PRYE = 1 0 : Odd parity 1 : Even parity	RW
PRYE	Parity enable bit	0 : Parity disabled 1 : Parity enabled	RW
IOPOL	TxD, RxD I/O polarity reverse bit	0 : No reverse 1 : Reverse	RW

Note 1: Set the corresponding port direction bit for each CLKi pin to "0" (input mode).
 Note 2: To receive data, set the corresponding port direction bit for each RxDi pin to "0" (input mode).
 Note 3: Set the corresponding port direction bit for SCL and SDA pins to "0" (input mode).

UARTi transmit/receive control register 0 (i=0 to 2)

b7	b6	b5	b4	b3	b2	b1	b0	Symbol	Address	After reset
								U0C0 to U2C0	03A4 ₁₆ , 03AC ₁₆ , 037C ₁₆	00001000 ₂

Bit symbol	Bit name	Function	RW
CLK0	BRG count source select bit	^{b1 b0} 0 0 : f _{1SIO} or f _{2SIO} is selected 0 1 : f _{8SIO} is selected 1 0 : f _{32SIO} is selected 1 1 : Must not be set	RW
CLK1			RW
CRS	CTS/RTS function select bit (Note 4)	Effective when CRD = 0 0 : CTS function is selected (Note 1) 1 : RTS function is selected	RW
TXEPT	Transmit register empty flag	0 : Data present in transmit register (during transmission) 1 : No data present in transmit register (transmission completed)	RO
CRD	CTS/RTS disable bit	0 : CTS/RTS function enabled 1 : CTS/RTS function disabled (P6 ₀ , P6 ₄ and P7 ₃ can be used as I/O ports)	RW
NCH	Data output select bit (Note 2)	0 : TxDi/SDAi and SCLi pins are CMOS output 1 : TxDi/SDAi and SCLi pins are N-channel open-drain output	RW
CKPOL	CLK polarity select bit	0 : Transmit data is output at falling edge of transfer clock and receive data is input at rising edge 1 : Transmit data is output at rising edge of transfer clock and receive data is input at falling edge	RW
UFORM	Transfer format select bit (Note 3)	0 : LSB first 1 : MSB first	RW

Note 1: Set the corresponding port direction bit for each CTSi pin to "0" (input mode).
 Note 2: TxD₂/SDA₂ and SCL₂ are N-channel open-drain output. Cannot be set to the CMOS output. Set the NCH bit of the U2C0 register to "0".
 Note 3: Effective for clock synchronous serial I/O mode and UART mode transfer data 8 bits long.
 Note 4: CTS₁/RTS₁ can be used when the UCON register's CLKMD1 bit = "0" (only CLK₁ output) and the UCON register's RCSP bit = "0" (CTS₀/RTS₀ not separated).

Figure 2.11.4. U0MR to U2MR Register and U0C0 to U2C0 Register

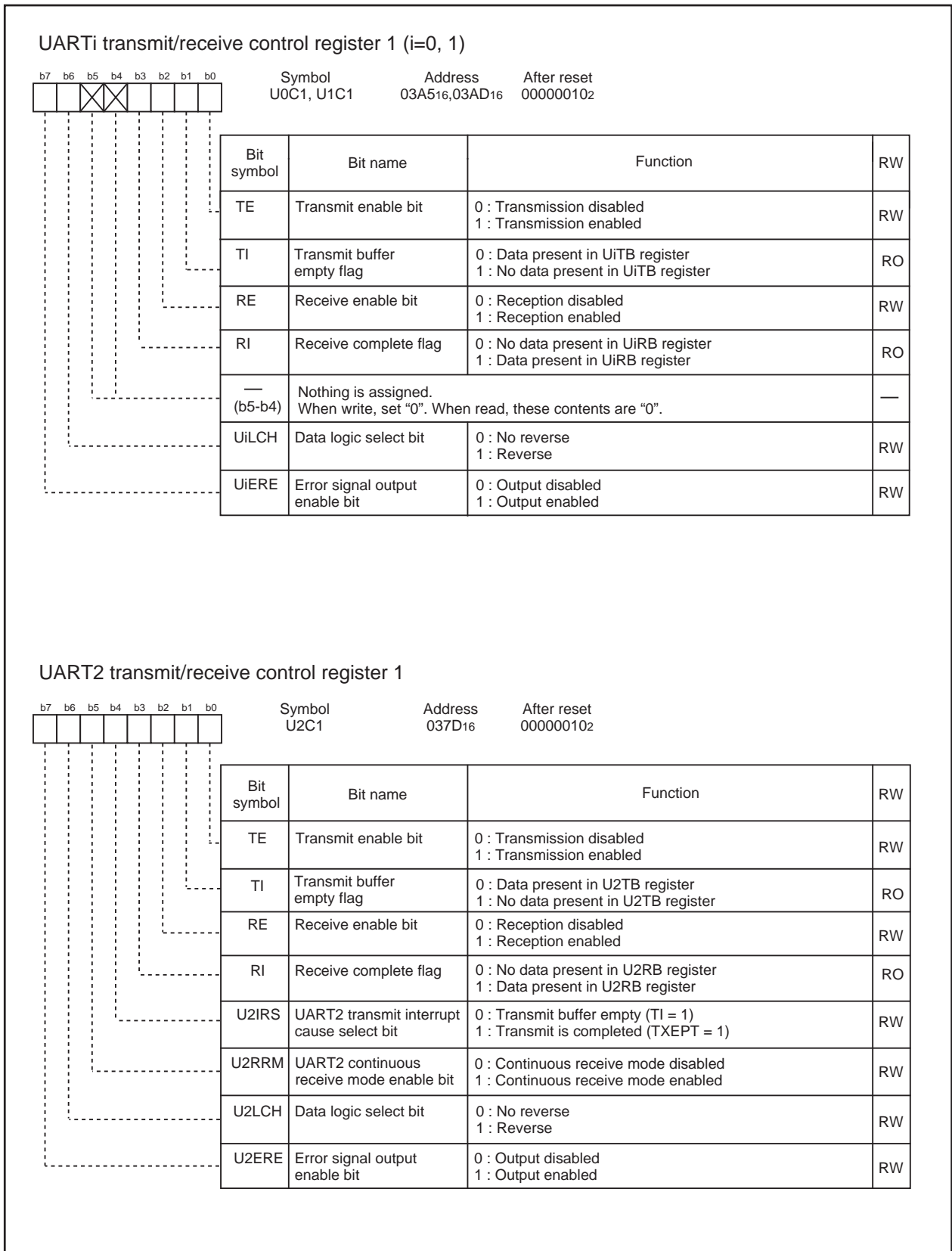


Figure 2.11.5. U0C1 to U2C1 Registers

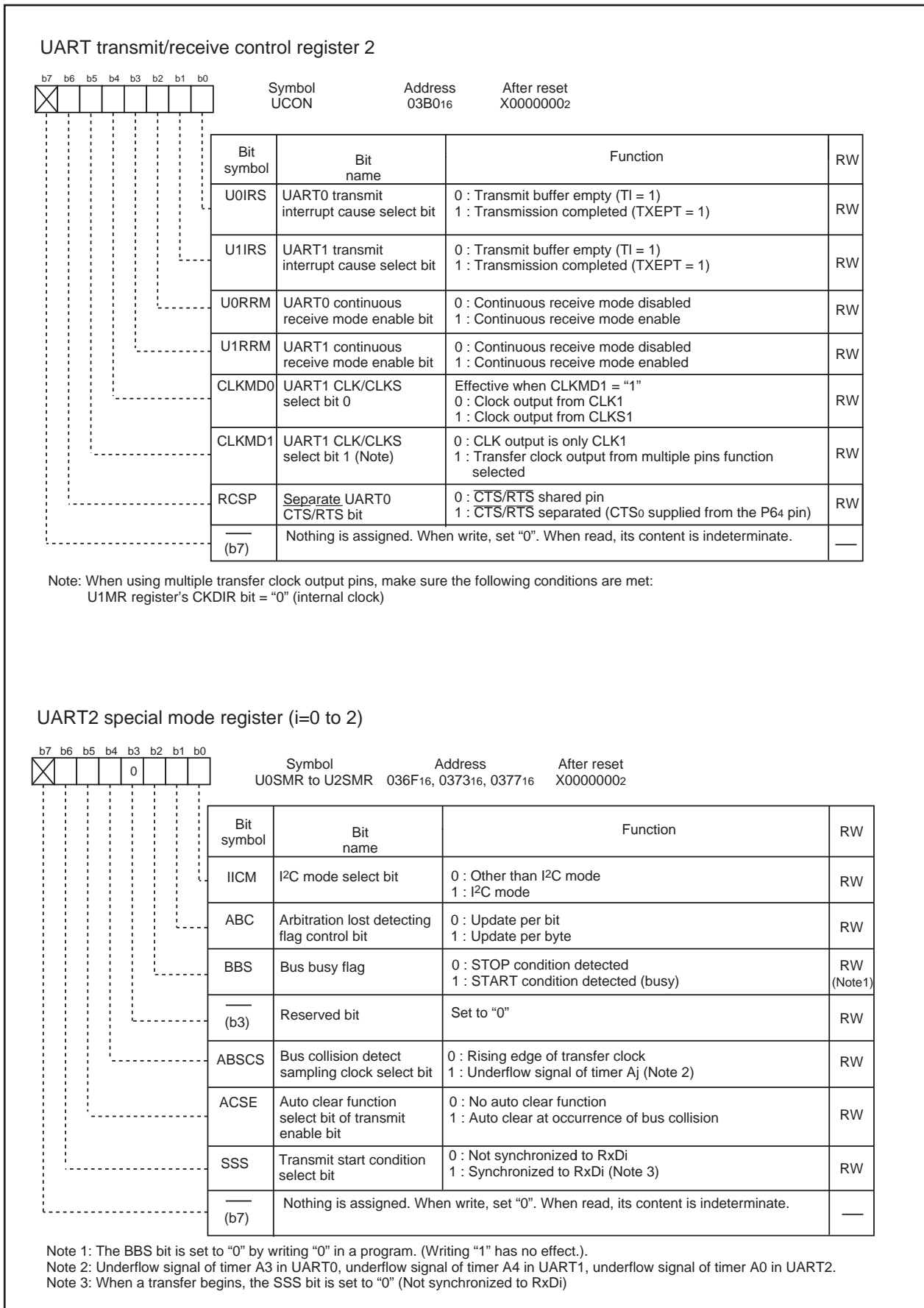


Figure 2.11.6. UCON Register and U0SMR to U2SMR Registers

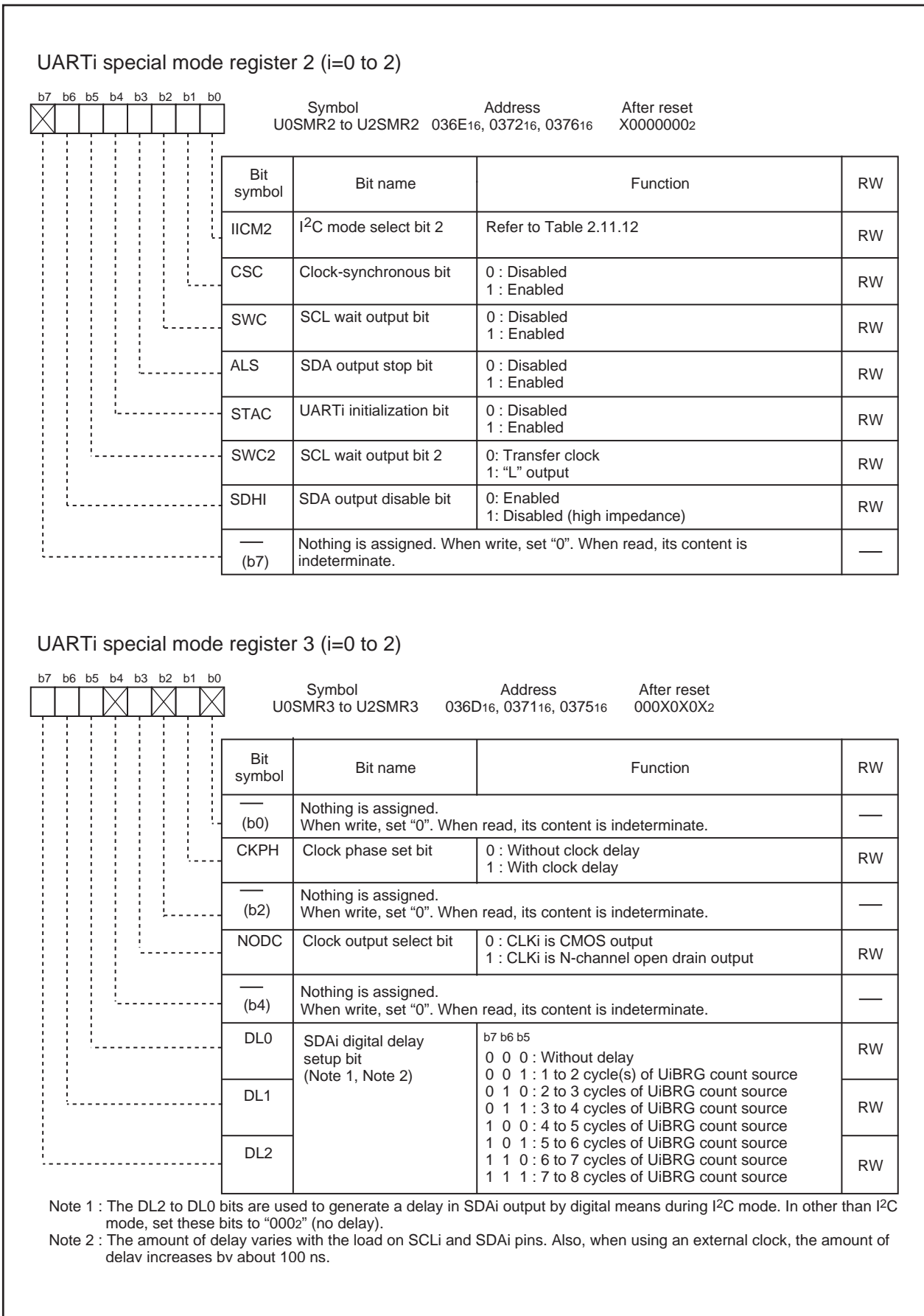


Figure 2.11.7. U0SMR2 to U2SMR2 Registers and U0SMR3 to U2SMR3 Registers

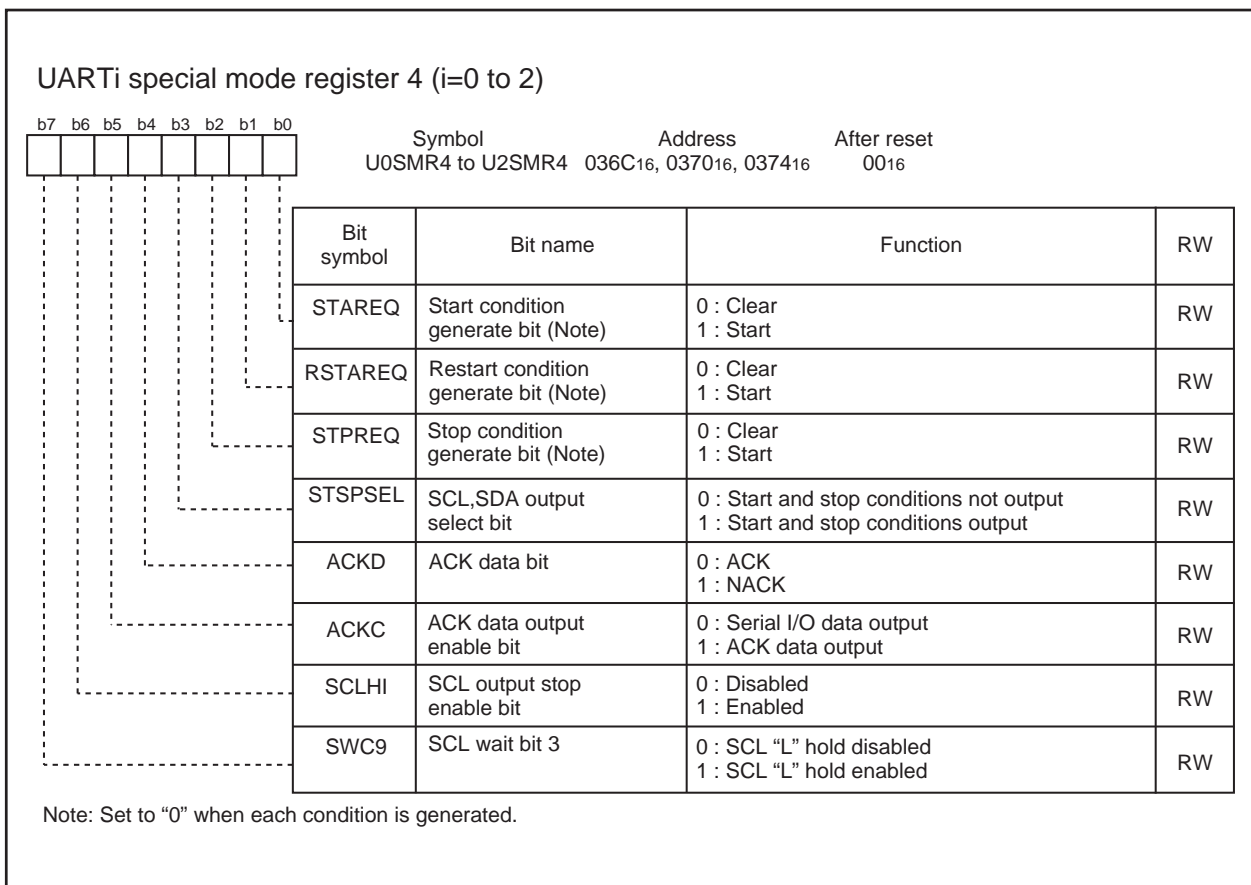


Figure 2.11.8. U0SMR4 to U2SMR4 Registers

2.11.2 Clock Synchronous serial I/O Mode

The clock synchronous serial I/O mode uses a transfer clock to transmit and receive data. Table 2.11.1 lists the specifications of the clock synchronous serial I/O mode. Table 2.11.2 lists the registers used in clock synchronous serial I/O mode and the register values set.

Table 2.11.1. Clock Synchronous Serial I/O Mode Specifications

Item	Specification
Transfer data format	<ul style="list-style-type: none"> Transfer data length: 8 bits
Transfer clock	<ul style="list-style-type: none"> UiMR(i=0 to 2) register's CKDIR bit = "0" (internal clock) : $f_j / 2(n+1)$ $f_j = f_{1SIO}, f_{2SIO}, f_{8SIO}, f_{32SIO}$. n: Setting value of UiBRG register 0016 to FF16 CKDIR bit = "1" (external clock) : Input from CLKi pin
Transmission, reception control	<ul style="list-style-type: none"> Selectable from CTS function, RTS function or CTS/RTS function disable
Transmission start condition	<ul style="list-style-type: none"> Before transmission can start, the following requirements must be met (Note 1) <ul style="list-style-type: none"> The TE bit of UiC1 register= 1 (transmission enabled) The TI bit of UiC1 register = 0 (data present in UiTB register) If CTS function is selected, input on the CTSi pin = "L"
Reception start condition	<ul style="list-style-type: none"> Before reception can start, the following requirements must be met (Note 1) <ul style="list-style-type: none"> The RE bit of UiC1 register= 1 (reception enabled) The TE bit of UiC1 register= 1 (transmission enabled) The TI bit of UiC1 register= 0 (data present in the UiTB register)
Interrupt request generation timing	<ul style="list-style-type: none"> For transmission, one of the following conditions can be selected <ul style="list-style-type: none"> The UiIRS bit (Note 3) = 0 (transmit buffer empty): when transferring data from the UiTB register to the UARTi transmit register (at start of transmission) The UiIRS bit =1 (transfer completed): when the serial I/O finished sending data from the UARTi transmit register For reception <ul style="list-style-type: none"> When transferring data from the UARTi receive register to the UiRB register (at completion of reception)
Error detection	<ul style="list-style-type: none"> Overrun error (Note 2) <ul style="list-style-type: none"> This error occurs if the serial I/O started receiving the next data before reading the UiRB register and received the 7th bit of the next data
Select function	<ul style="list-style-type: none"> CLK polarity selection <ul style="list-style-type: none"> Transfer data input/output can be chosen to occur synchronously with the rising or the falling edge of the transfer clock LSB first, MSB first selection <ul style="list-style-type: none"> Whether to start sending/receiving data beginning with bit 0 or beginning with bit 7 can be selected Continuous receive mode selection <ul style="list-style-type: none"> Reception is enabled immediately by reading the UiRB register Switching serial data logic <ul style="list-style-type: none"> This function reverses the logic value of the transmit/receive data Transfer clock output from multiple pins selection (UART1) <ul style="list-style-type: none"> The output pin can be selected in a program from two UART1 transfer clock pins that have been set Separate CTS/RTS pins (UART0) <ul style="list-style-type: none"> CTS0 and RTS0 are input/output from separate pins

Note 1: When an external clock is selected, the conditions must be met while if the UiC0 register's CKPOL bit = "0" (transmit data output at the falling edge and the receive data taken in at the rising edge of the transfer clock), the external clock is in the high state; if the UiC0 register's CKPOL bit = "1" (transmit data output at the rising edge and the receive data taken in at the falling edge of the transfer clock), the external clock is in the low state.

Note 2: If an overrun error occurs, the value of UiRB register will be indeterminate. The IR bit of SiRIC register does not change.

Note 3: The U0IRS and U1IRS bits respectively are the UCON register bits 0 and 1; the U2IRS bit is the U2C1 register bit 4.

Table 2. 11. 2. Registers to Be Used and Settings in Clock Synchronous Serial I/O Mode

Register	Bit	Function
UiTB(Note3)	0 to 7	Set transmission data
UiRB(Note3)	0 to 7	Reception data can be read
	OER	Overrun error flag
UiBRG	0 to 7	Set a transfer rate
UiMR(Note3)	SMD2 to SMD0	Set to "0012"
	CKDIR	Select the internal clock or external clock
	IOPOL	Set to "0"
UiC0	CLK1 to CLK0	Select the count source for the UiBRG register
	CRS	Select CTS or RTS to use
	TXEPT	Transmit register empty flag
	CRD	Enable or disable the CTS or RTS function
	NCH	Select TxDi pin output mode (Note 2)
	CKPOL	Select the transfer clock polarity
	UFORM	Select the LSB first or MSB first
UiC1	TE	Set this bit to "1" to enable transmission/reception
	TI	Transmit buffer empty flag
	RE	Set this bit to "1" to enable reception
	RI	Reception complete flag
	U2IRS (Note 1)	Select the source of UART2 transmit interrupt
	U2RRM (Note 1)	Set this bit to "1" to use continuous receive mode
	UiLCH	Set this bit to "1" to use inverted data logic
	UiERE	Set to "0"
UiSMR	0 to 7	Set to "0"
UiSMR2	0 to 7	Set to "0"
UiSMR3	0 to 2	Set to "0"
	NODC	Select clock output mode
	4 to 7	Set to "0"
UiSMR4	0 to 7	Set to "0"
UCON	U0IRS, U1IRS	Select the source of UART0/UART1 transmit interrupt
	U0RRM, U1RRM	Set this bit to "1" to use continuous receive mode
	CLKMD0	Select the transfer clock output pin when CLKMD1 = 1
	CLKMD1	Set this bit to "1" to output UART1 transfer clock from two pins
	RCSP	Set this bit to "1" to accept as input the UART0 CTS ₀ signal from the P64 pin
	7	Set to "0"

Note 1: Set the U0C1 and U1C1 register bit 4 and bit 5 to "0". The U0IRS, U1IRS, U0RRM and U1RRM bits are in the UCON register.

Note 2: TxD2 pin is N channel open-drain output. Set the U2C0 register's NCH bit to "0".

Note 3: Not all register bits are described above. Set those bits to "0" when writing to the registers in clock synchronous serial I/O mode.

i=0 to 2

Table 2.11.3 lists the functions of the input/output pins during clock synchronous serial I/O mode. Table 2.11.3 shows pin functions for the case where the multiple transfer clock output pin select function is deselected. Table 2.11.4 lists the P64 pin functions during clock synchronous serial I/O mode. Note that for a period from when the UART_i operation mode is selected to when transfer starts, the TxDi pin outputs an “H”. (If the N-channel open-drain output is selected, this pin is in a high-impedance state.)

Table 2.11.3. Pin Functions (When Not Select Multiple Transfer Clock Output Pin Function)

Pin name	Function	Method of selection
TxD _i (i = 0 to 2) (P63, P67, P70)	Serial data output	(Outputs dummy data when performing reception only)
RxD _i (P62, P66, P71)	Serial data input	PD6 register's PD6_2 bit=0, PD6_6 bit=0, PD7 register's PD7_1 bit=0 (Can be used as an input port when performing transmission only)
CLK _i (P61, P65, P72)	Transfer clock output	UiMR register's CKDIR bit=0
	Transfer clock input	UiMR register's CKDIR bit=1 PD6 register's PD6_1 bit=0, PD6_5 bit=0, PD7 register's PD7_2 bit=0
CTS _i /RTS _i (P60, P64, P73)	CTS input	UiC0 register's CRD bit=0 UiC0 register's CRS bit=0 PD6 register's PD6_0 bit=0, PD6_4 bit=0, PD7 register's PD7_3 bit=0
	RTS output	UiC0 register's CRD bit=0 UiC0 register's CRS bit=1
	I/O port	UiC0 register's CRD bit=1

Table 2.11.4. P64 Pin Functions

Pin function	Bit set value					
	U1C0 register		UCON register			PD6 register
	CRD	CRS	RCSP	CLKMD1	CLKMD0	PD6_4
P64	1	—	0	0	—	Input: 0, Output: 1
CTS ₁	0	0	0	0	—	0
RTS ₁	0	1	0	0	—	—
CTS ₀ (Note1)	0	0	1	0	—	0
CLKS ₁	—	—	—	1(Note 2)	1	—

Note 1: In addition to this, set the U0C0 register's CRD bit to “0” (CTS₀/RTS₀ enabled) and the U0C0 register's CRS bit to “1” (RTS₀ selected).

Note 2: When the CLKMD1 bit = 1 and the CLKMD0 bit = 0, the following logic levels are output:

- High if the U1C0 register's CLKPOL bit = 0
- Low if the U1C0 register's CLKPOL bit = 1

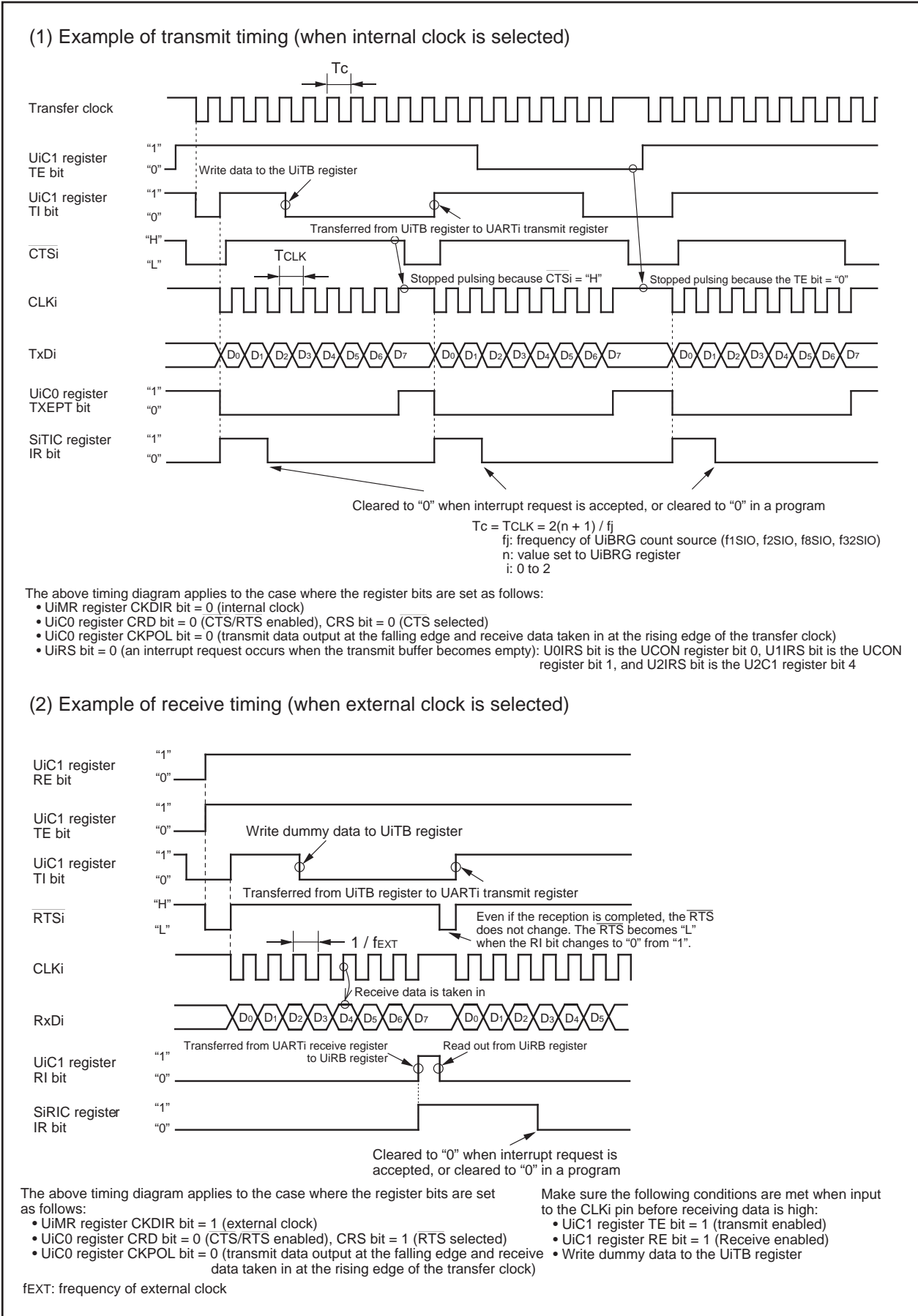


Figure 2.11.9. Transmit and Receive Operation

(a) CLK Polarity Select Function

Use the UiC0 register (i = 0 to 2)'s CKPOL bit to select the transfer clock polarity. Figure 2.11.10 shows the polarity of the transfer clock.

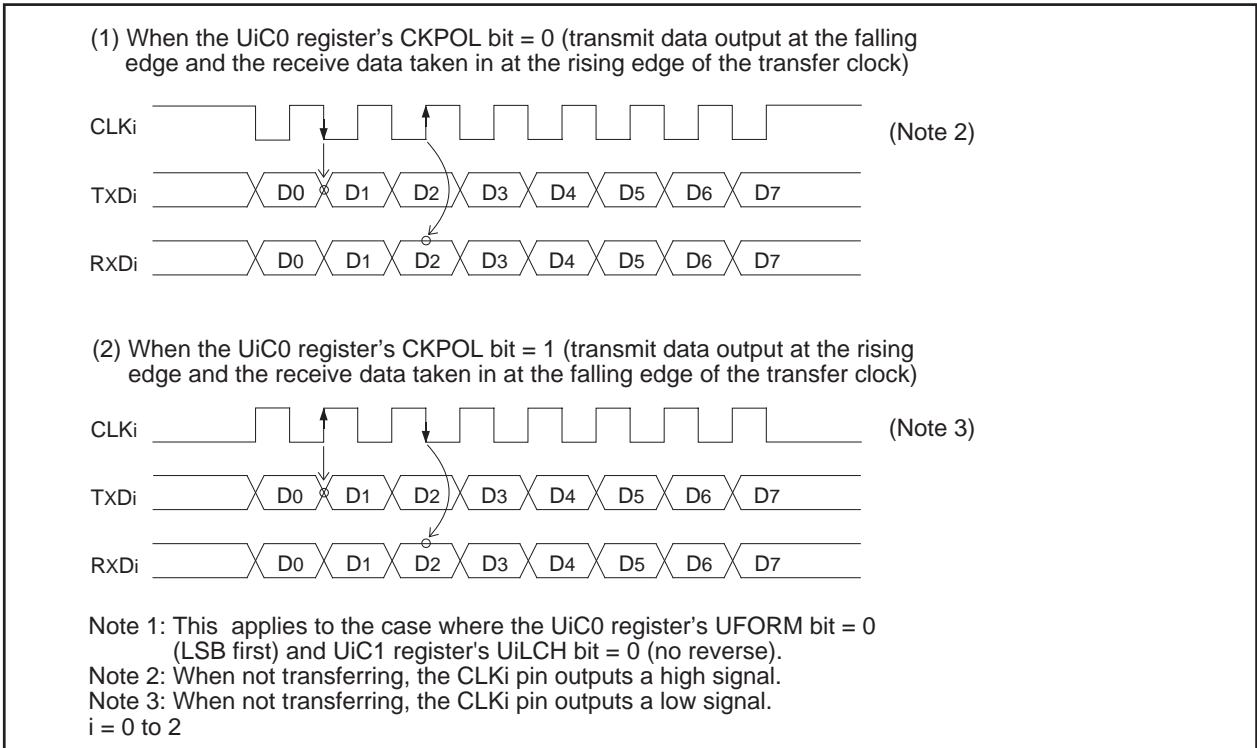


Figure 2.11.10. Transfer Clock Polarity

(b) LSB First/MSB First Select Function

Use the UiC0 register (i = 0 to 2)'s UFORM bit to select the transfer format. Figure 2.11.11 shows the transfer format.

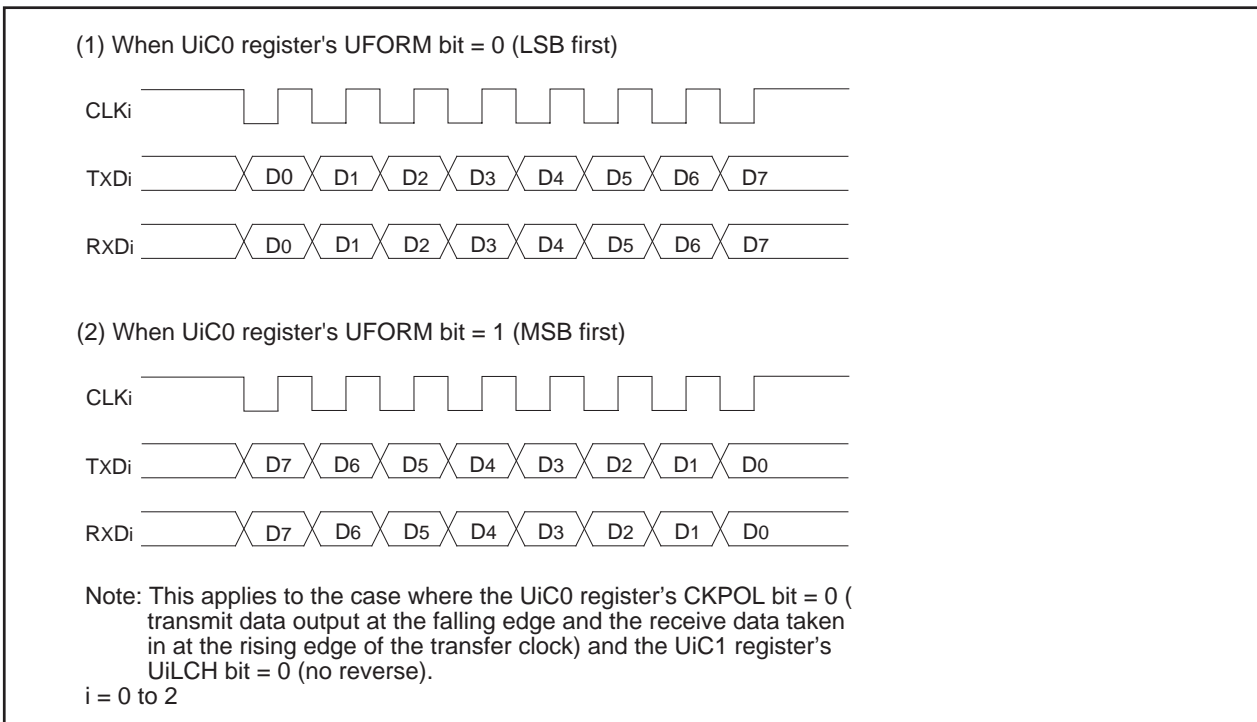


Figure 2.11.11. Transfer Format

(c) Continuous Receive Mode

When the UiRRM bit ($i = 0$ to 2) = 1 (continuous receive mode), the UiC1 register's TI bit is set to "0" (data present in the UiTB register) by reading the UiRB register. In this case, i.e., UiRRM bit = 1, do not write dummy data to the UiTB register in a program. The U0RRM and U1RRM bits are the UCON register bit 2 and bit 3, respectively, and the U2RRM bit is the U2C1 register bit 5.

(d) Serial Data Logic Switching Function

When the UiC1 register ($i = 0$ to 2)'s UiLCH bit = 1 (reverse), the data written to the UiTB register has its logic reversed before being transmitted. Similarly, the received data has its logic reversed when read from the UiRB register. Figure 2.11.12 shows serial data logic.

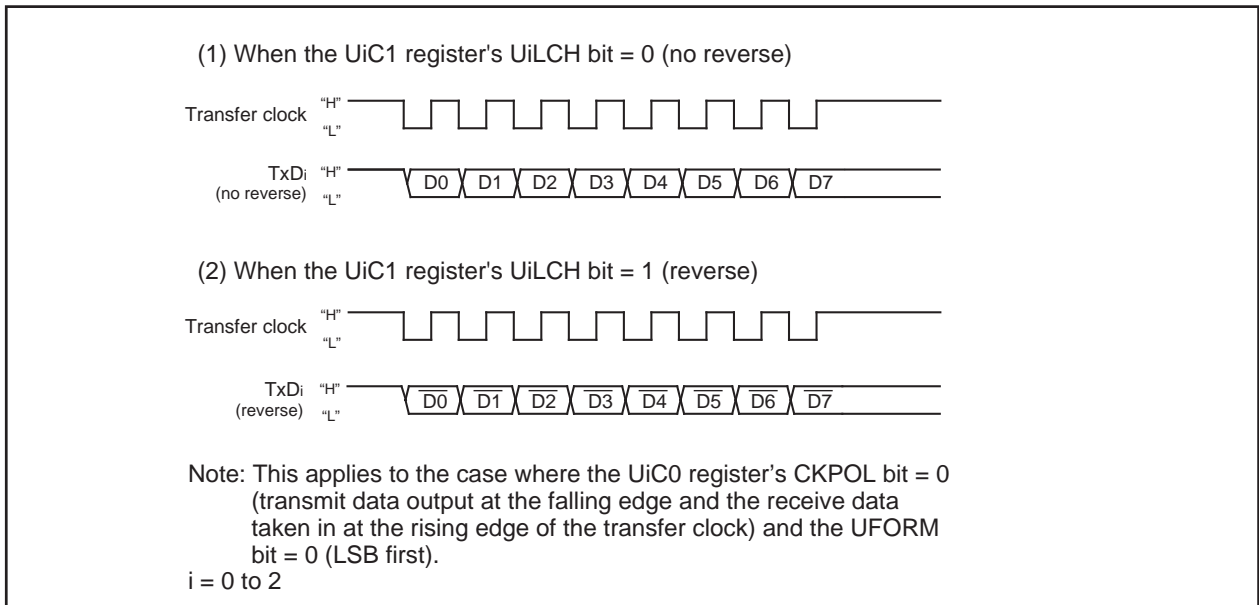


Figure 2.11.12. Serial Data Logic Switching

(e) Transfer Clock Output From Multiple Pins (UART1)

Use the UCON register's CLKMD1 to CLKMD0 bits to select one of the two transfer clock output pins. (See Figure 2.11.13.) This function can be used when the selected transfer clock for UART1 is an internal clock.

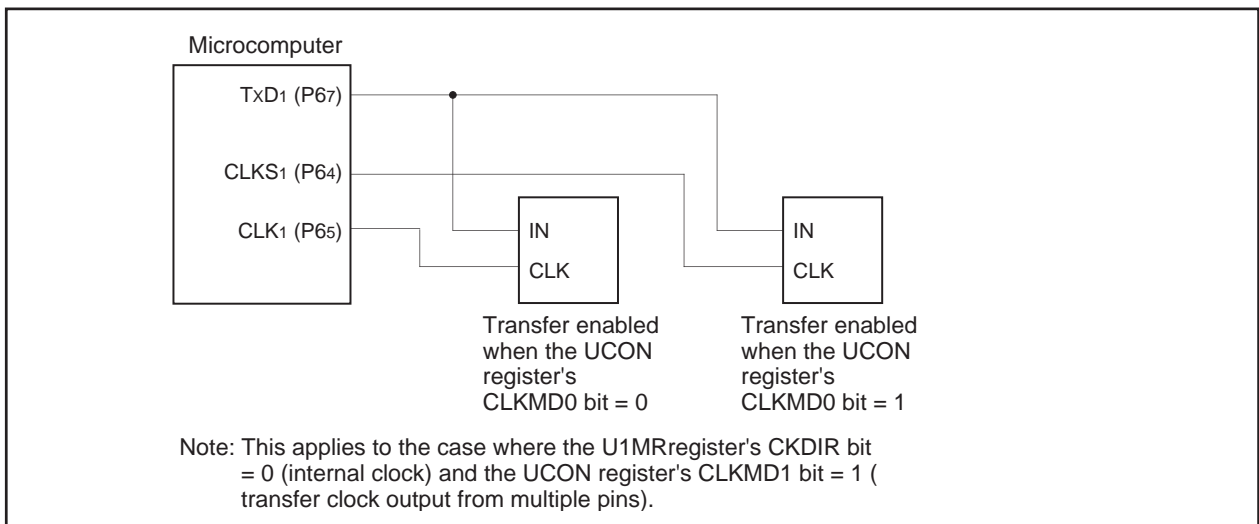


Figure 2.11.13. Transfer Clock Output From Multiple Pins

(f) $\overline{\text{CTS}}/\overline{\text{RTS}}$ Separate Function (UART0)

This function separates $\overline{\text{CTS}}_0/\overline{\text{RTS}}_0$, outputs $\overline{\text{RTS}}_0$ from the P60 pin, and accepts as input the $\overline{\text{CTS}}_0$ from the P64 pin. To use this function, set the register bits as shown below.

- U0C0 register's CRD bit = 0 (enables UART0 $\overline{\text{CTS}}/\overline{\text{RTS}}$)
- U0C0 register's CRS bit = 1 (outputs UART0 $\overline{\text{RTS}}$)
- U1C0 register's CRD bit = 0 (enables UART1 $\overline{\text{CTS}}/\overline{\text{RTS}}$)
- U1C0 register's CRS bit = 0 (inputs UART1 $\overline{\text{CTS}}$)
- UCON register's RCSP bit = 1 (inputs $\overline{\text{CTS}}_0$ from the P64 pin)
- UCON register's CLKMD1 bit = 0 (CLKS1 not used)

Note that when using the $\overline{\text{CTS}}/\overline{\text{RTS}}$ separate function, UART1 $\overline{\text{CTS}}/\overline{\text{RTS}}$ function cannot be used.

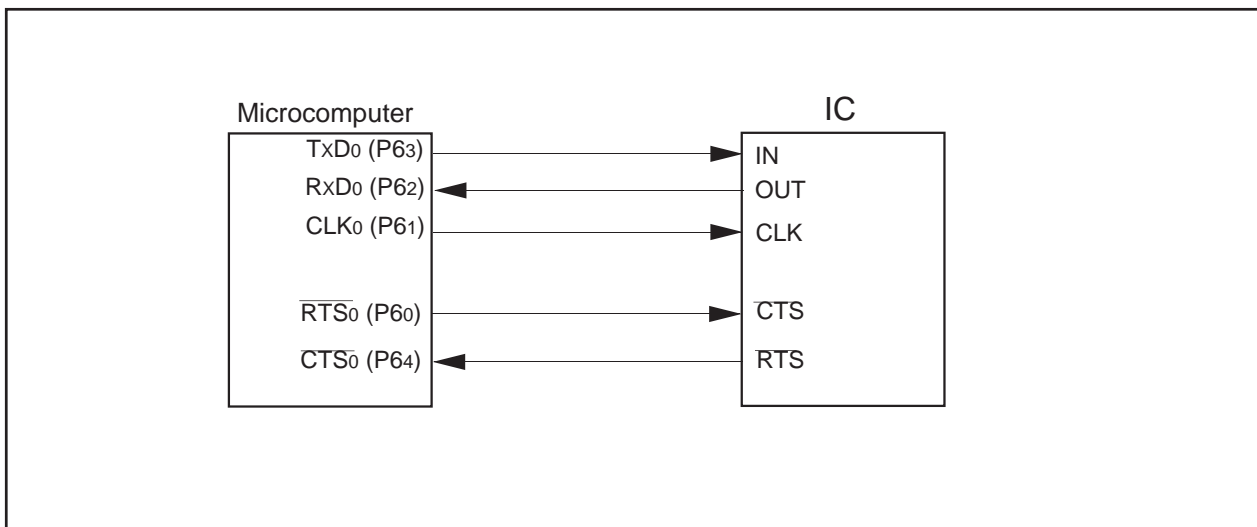


Figure 2.11.14. $\overline{\text{CTS}}/\overline{\text{RTS}}$ Separat Function

2.11.3 Clock Asynchronous Serial I/O (UART) Mode

The UART mode allows transmitting and receiving data after setting the desired transfer rate and transfer data format. Tables 2.11.5 lists the specifications of the UART mode.

Table 2.11.5. UART Mode Specifications

Item	Specification
Transfer data format	<ul style="list-style-type: none"> Character bit (transfer data): Selectable from 7, 8 or 9 bits Start bit: 1 bit Parity bit: Selectable from odd, even, or none Stop bit: Selectable from 1 or 2 bits
Transfer clock	<ul style="list-style-type: none"> UiMR(i=0 to 2) register's CKDIR bit = 0 (internal clock) : $f_j / 16(n+1)$ $f_j = f_{1SIO}, f_{2SIO}, f_{8SIO}, f_{32SIO}$. n: Setting value of UiBRG register 00₁₆ to FF₁₆ CKDIR bit = "1" (external clock) : $f_{EXT}/16(n+1)$ f_{EXT}: Input from CLKi pin. n: Setting value of UiBRG register 00₁₆ to FF₁₆
Transmission, reception control	<ul style="list-style-type: none"> Selectable from \overline{CTS} function, \overline{RTS} function or $\overline{CTS}/\overline{RTS}$ function disable
Transmission start condition	<ul style="list-style-type: none"> Before transmission can start, the following requirements must be met <ul style="list-style-type: none"> The TE bit of UiC1 register= 1 (transmission enabled) The TI bit of UiC1 register = 0 (data present in UiTB register) If \overline{CTS} function is selected, input on the \overline{CTS}_i pin = "L"
Reception start condition	<ul style="list-style-type: none"> Before reception can start, the following requirements must be met <ul style="list-style-type: none"> The RE bit of UiC1 register= 1 (reception enabled) Start bit detection
Interrupt request generation timing	<ul style="list-style-type: none"> For transmission, one of the following conditions can be selected <ul style="list-style-type: none"> The UiIRS bit (Note 2) = 0 (transmit buffer empty): when transferring data from the UiTB register to the UARTi transmit register (at start of transmission) The UiIRS bit =1 (transfer completed): when the serial I/O finished sending data from the UARTi transmit register For reception <ul style="list-style-type: none"> When transferring data from the UARTi receive register to the UiRB register (at completion of reception)
Error detection	<ul style="list-style-type: none"> Overrun error (Note 1) This error occurs if the serial I/O started receiving the next data before reading the UiRB register and received the bit one before the last stop bit of the next data Framing error This error occurs when the number of stop bits set is not detected Parity error This error occurs when if parity is enabled, the number of 1's in parity and character bits does not match the number of 1's set Error sum flag This flag is set (= 1) when any of the overrun, framing, and parity errors is encountered
Select function	<ul style="list-style-type: none"> LSB first, MSB first selection Whether to start sending/receiving data beginning with bit 0 or beginning with bit 7 can be selected Serial data logic switch This function reverses the logic of the transmit/receive data. The start and stop bits are not reversed. TxD, RxD I/O polarity switch This function reverses the polarities of the TxD pin output and RxD pin input. The logic levels of all I/O data is reversed. Separate $\overline{CTS}/\overline{RTS}$ pins (UART0) \overline{CTS}_0 and \overline{RTS}_0 are input/output from separate pins

Note 1: If an overrun error occurs, the value of UiRB register will be indeterminate. The IR bit of SiRIC register does not change.

Note 2: The U0IRS and U1IRS bits respectively are the UCON register bits 0 and 1; the U2IRS bit is the U2C1 register bit 4.

Table 2. 11. 6. Registers to Be Used and Settings in UART Mode

Register	Bit	Function
UiTB	0 to 8	Set transmission data (Note 1)
UiRB	0 to 8	Reception data can be read (Note 1)
	OER,FER,PER,SUM	Error flag
UiBRG	0 to 7	Set a transfer rate
UiMR	SMD2 to SMD0	Set these bits to '1002' when transfer data is 7 bits long Set these bits to '1012' when transfer data is 8 bits long Set these bits to '1102' when transfer data is 9 bits long
	CKDIR	Select the internal clock or external clock
	STPS	Select the stop bit
	PRY, PRYE	Select whether parity is included and whether odd or even
	IOPOL	Select the TxD/RxD input/output polarity
UiC0	CLK0, CLK1	Select the count source for the UiBRG register
	CRS	Select CTS or RTS to use
	TXEPT	Transmit register empty flag
	CRD	Enable or disable the CTS or RTS function
	NCH	Select TxDi pin output mode (Note 3)
	CKPOL	Set to "0"
	UFORM	LSB first or MSB first can be selected when transfer data is 8 bits long. Set this bit to "0" when transfer data is 7 or 9 bits long.
UiC1	TE	Set this bit to "1" to enable transmission
	TI	Transmit buffer empty flag
	RE	Set this bit to "1" to enable reception
	RI	Reception complete flag
	U2IRS (Note 2)	Select the source of UART2 transmit interrupt
	U2RRM (Note 2)	Set to "0"
	UiLCH	Set this bit to "1" to use inverted data logic
	UiERE	Set to "0"
UiSMR	0 to 7	Set to "0"
UiSMR2	0 to 7	Set to "0"
UiSMR3	0 to 7	Set to "0"
UiSMR4	0 to 7	Set to "0"
UCON	U0IRS, U1IRS	Select the source of UART0/UART1 transmit interrupt
	U0RRM, U1RRM	Set to "0"
	CLKMD0	Invalid because CLKMD1 = 0
	CLKMD1	Set to "0"
	RCSP	Set this bit to "1" to accept as input the UART0 CTS ₀ signal from the P64 pin
	7	Set to "0"

Note 1: The bits used for transmit/receive data are as follows: Bit 0 to bit 6 when transfer data is 7 bits long; bit 0 to bit 7 when transfer data is 8 bits long; bit 0 to bit 8 when transfer data is 9 bits long.

Note 2: Set the U0C1 and U1C1 registers bit 4 to bit 5 to "0". The U0IRS, U1IRS, U0RRM and U1RRM bits are included in the UCON register.

Note 3: TxD2 pin is N channel open-drain output. Set the U2C0 register's NCH bit to "0".

i=0 to 2

Table 2.11.7 lists the functions of the input/output pins during UART mode. Table 2.11.8 lists the P64 pin functions during UART mode. Note that for a period from when the UARTi operation mode is selected to when transfer starts, the TxDi pin outputs an “H”. (If the N-channel open-drain output is selected, this pin is in a high-impedance state.)

Table 2.11.7. I/O Pin Functions

Pin name	Function	Method of selection
TxDi (i = 0 to 2) (P63, P67, P70)	Serial data output	(Outputs dummy data when performing reception only)
RxDi (P62, P66, P71)	Serial data input	PD6 register's PD6_2 bit=0, PD6_6 bit=0, PD7 register's PD7_1 bit=0 (Can be used as an input port when performing transmission only)
CLKi (P61, P65, P72)	Input/output port	UiMR register's CKDIR bit=0
	Transfer clock input	UiMR register's CKDIR bit=1 PD6 register's PD6_1 bit=0, PD6_5 bit=0, PD7 register's PD7_2 bit=0
$\overline{\text{CTS}}/\overline{\text{RTS}}_i$ (P60, P64, P73)	$\overline{\text{CTS}}$ input	UiC0 register's CRD bit=0 UiC0 register's CRS bit=0 PD6 register's PD6_0 bit=0, PD6_4 bit=0, PD7 register's PD7_3 bit=0
	RTS output	UiC0 register's CRD bit=0 UiC0 register's CRS bit=1
	Input/output port	UiC0 register's CRD bit=1

Table 2.11.8. P64 Pin Functions

Pin function	Bit set value				
	U1C0 register		UCON register		PD6 register
	CRD	CRS	RCSP	CLKMD1	PD6_4
P64	1	—	0	0	Input: 0, Output: 1
$\overline{\text{CTS}}_1$	0	0	0	0	0
$\overline{\text{RTS}}_1$	0	1	0	0	—
$\overline{\text{CTS}}_0$ (Note)	0	0	1	0	0

Note: In addition to this, set the U0C0 register's CRD bit to “0” ($\overline{\text{CTS}}_0/\overline{\text{RTS}}_0$ enabled) and the U0C0 register's CRS bit to “1” ($\overline{\text{RTS}}_0$ selected).

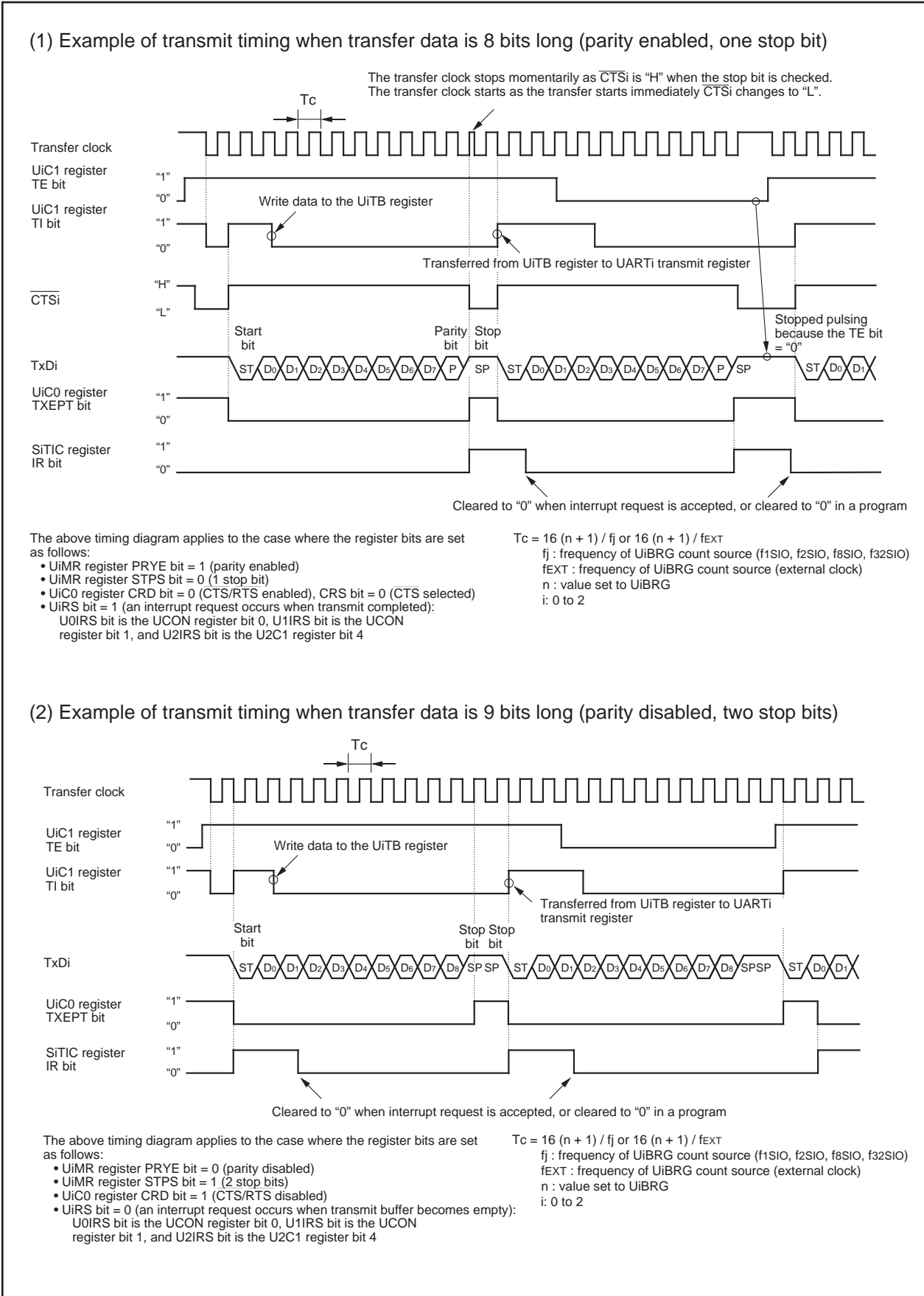


Figure 2.11.15. Transmit Operation

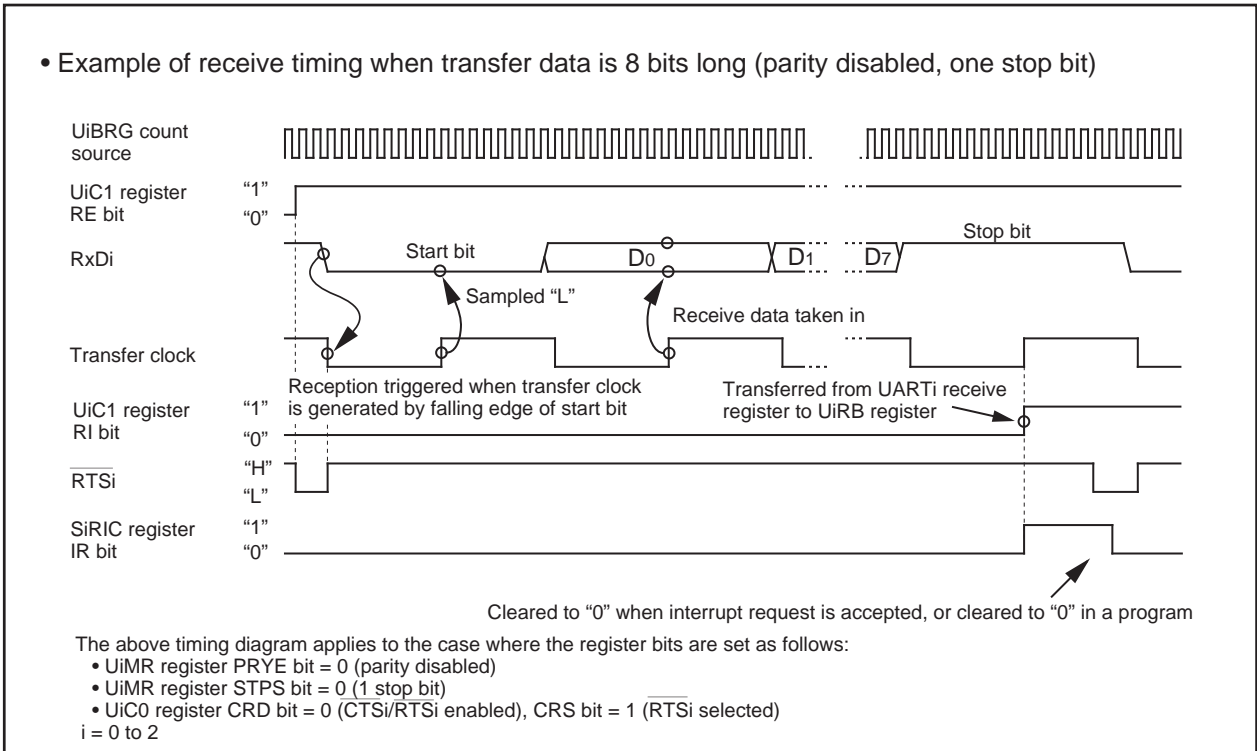


Figure 2.11.16. Receive Operation

(a) LSB First/MSB First Select Function

As shown in Figure 2.11.17, use the UiC0 register's UFORM bit to select the transfer format. This function is valid when transfer data is 8 bits long.

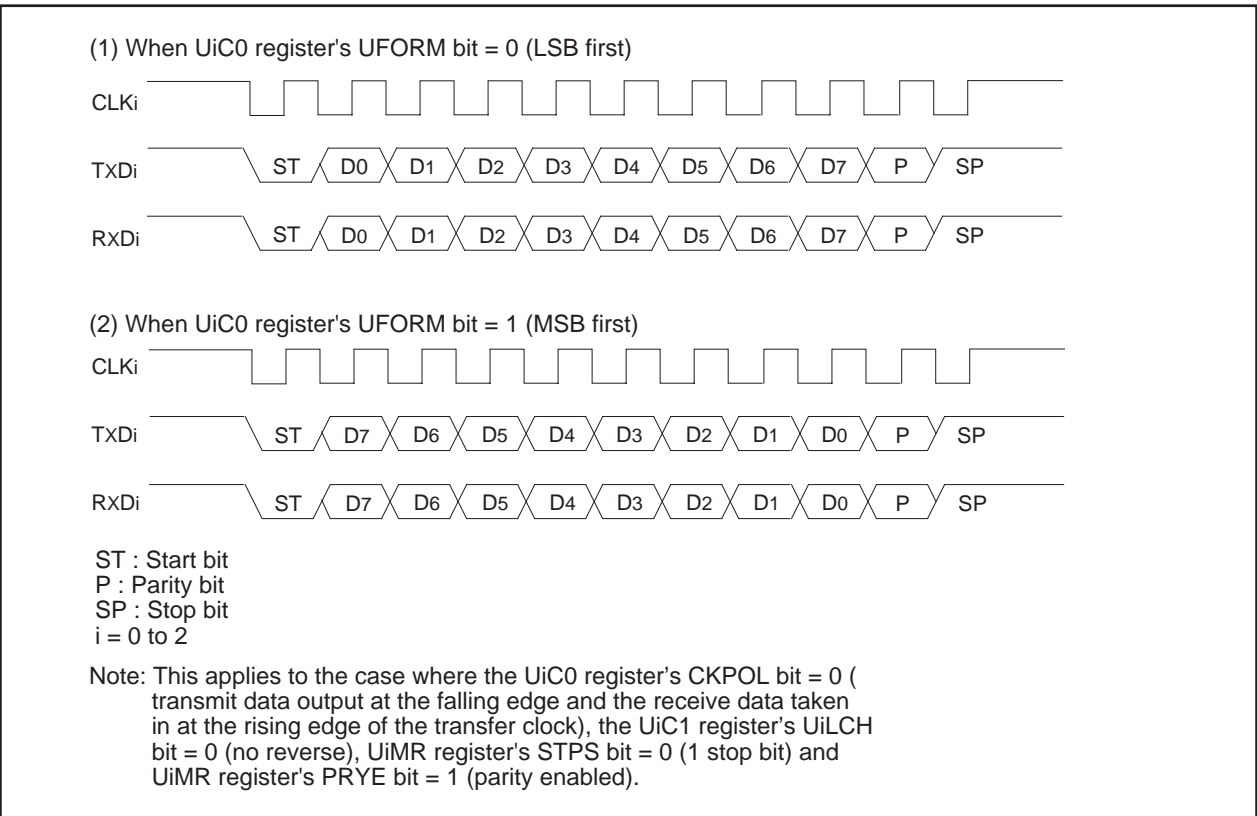


Figure 2.11.17. Transfer Format

(b) Serial Data Logic Switching Function

The data written to the UiTB register has its logic reversed before being transmitted. Similarly, the received data has its logic reversed when read from the UiRB register. Figure 2.11.18 shows serial data logic.

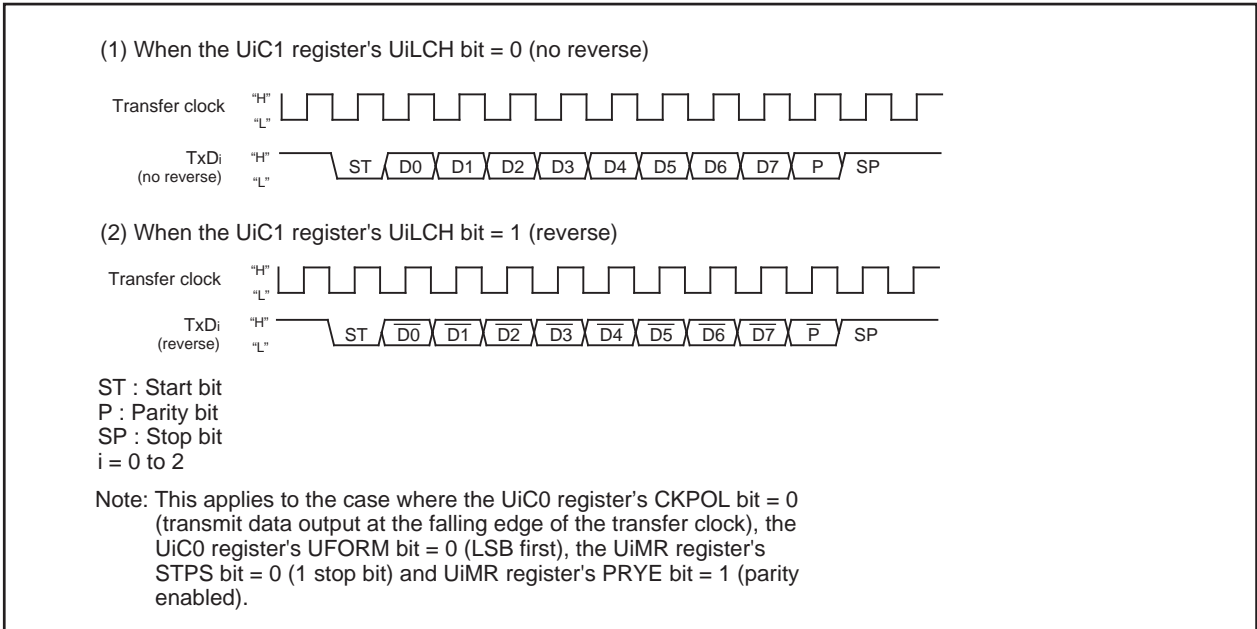


Figure 2.11.18. Serial Data Logic Switching

(c) TxD and RxD I/O Polarity Inverse Function

This function inverts the polarities of the TxDi pin output and RxDi pin input. The logic levels of all input/output data (including the start, stop and parity bits) are inverted. Figure 2.11.19 shows the TxD pin output and RxD pin input polarity inverse.

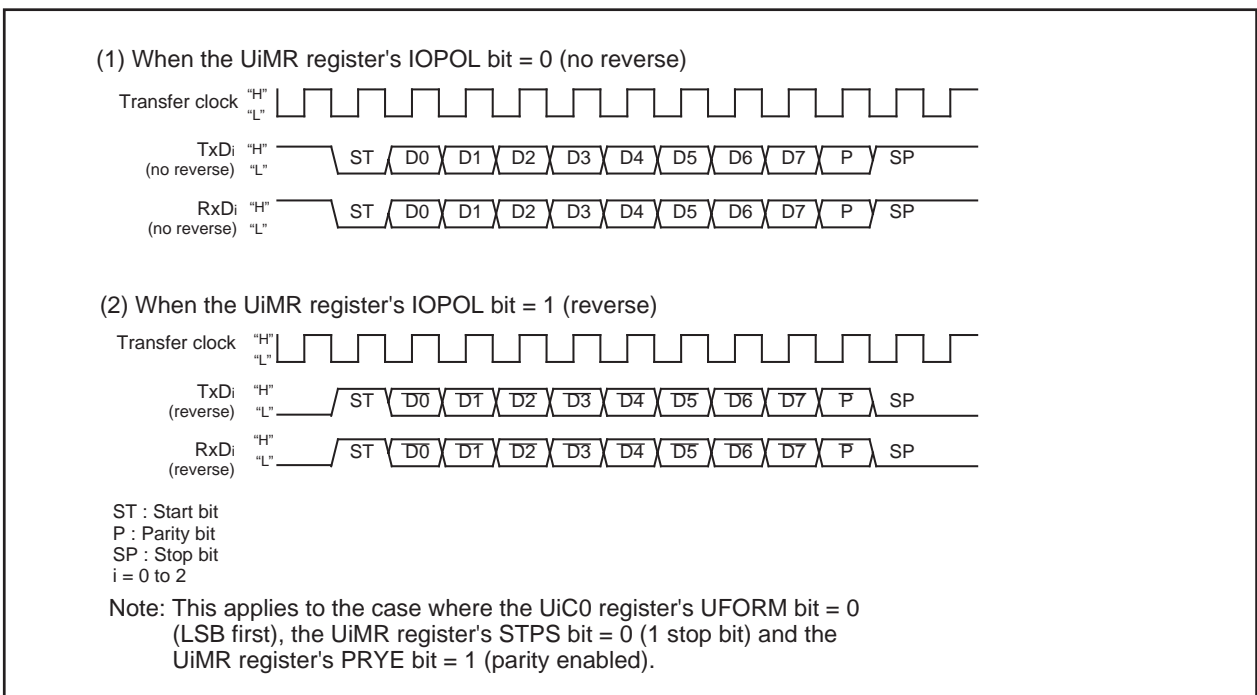


Figure 2.11.19. TxD and RxD I/O Polarity Inverse

(d) $\overline{\text{CTS}}/\overline{\text{RTS}}$ Separate Function (UART0)

This function separates $\overline{\text{CTS}}_0/\overline{\text{RTS}}_0$, outputs $\overline{\text{RTS}}_0$ from the P60 pin, and accepts as input the $\overline{\text{CTS}}_0$ from the P64 pin. To use this function, set the register bits as shown below.

- U0C0 register's CRD bit = 0 (enables UART0 $\overline{\text{CTS}}/\overline{\text{RTS}}$)
- U0C0 register's CRS bit = 1 (outputs UART0 $\overline{\text{RTS}}$)
- U1C0 register's CRD bit = 0 (enables UART1 $\overline{\text{CTS}}/\overline{\text{RTS}}$)
- U1C0 register's CRS bit = 0 (inputs UART1 $\overline{\text{CTS}}$)
- UCON register's RCSP bit = 1 (inputs $\overline{\text{CTS}}_0$ from the P64 pin)
- UCON register's CLKMD1 bit = 0 (CLKS1 not used)

Note that when using the $\overline{\text{CTS}}/\overline{\text{RTS}}$ separate function, UART1 $\overline{\text{CTS}}/\overline{\text{RTS}}$ function cannot be used.

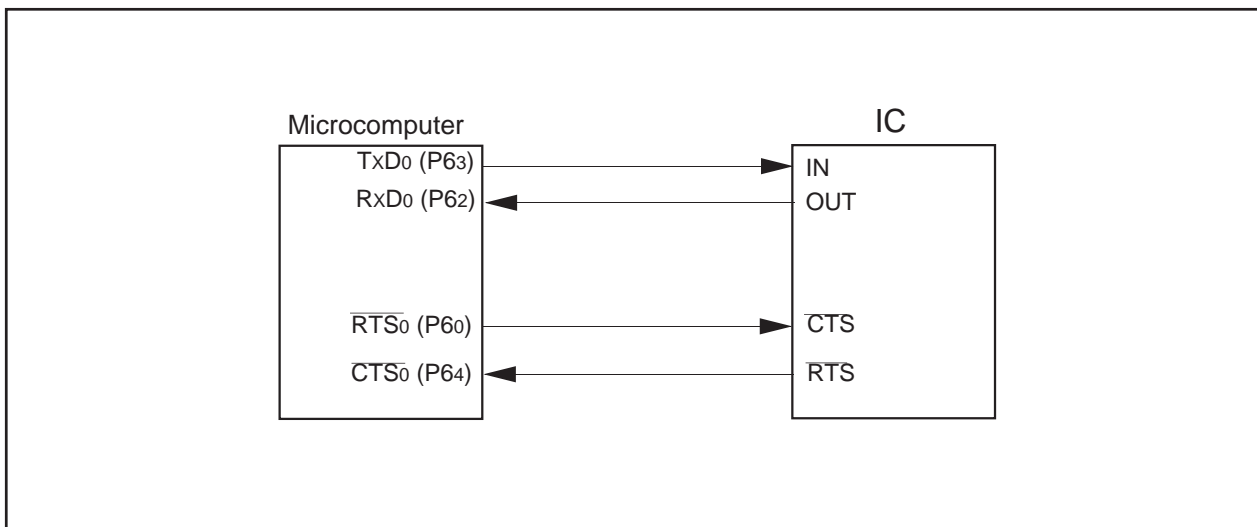


Figure 2.11.20. $\overline{\text{CTS}}/\overline{\text{RTS}}$ Separate Function

2.11.4 Special Mode 1 (I²C mode)

I²C mode is provided for use as a simplified I²C interface compatible mode. Table 2.11.9 lists the specifications of the I²C mode. Table 2.11.10 to 2.11.11 lists the registers used in the I²C mode and the register values set, Table 2.11.12 lists the I²C mode functions. Figure 2.11.21 shows the block diagram for I²C mode. Figure 2.11.22 shows SCLi timing.

As shown in Table 2.11.12, the microcomputer is placed in I²C mode by setting the SMD2 to SMD0 bits to '0102' and the IICM bit to "1". Because SDAi transmit output has a delay circuit attached, SDAi output does not change state until SCLi goes low and remains stably low.

Table 2.11.9. I²C Mode Specifications

Item	Specification
Transfer data format	<ul style="list-style-type: none"> Transfer data length: 8 bits
Transfer clock	<ul style="list-style-type: none"> During master <ul style="list-style-type: none"> UiMR(i=0 to 2) register's CKDIR bit = "0" (internal clock) : $f_j / 2(n+1)$ $f_j = f_{1SIO}, f_{2SIO}, f_{8SIO}, f_{32SIO}$. n: Setting value of UiBRG register 0016 to FF16 During slave <ul style="list-style-type: none"> CKDIR bit = "1" (external clock) : Input from SCLi pin
Transmission start condition	<ul style="list-style-type: none"> Before transmission can start, the following requirements must be met (Note 1) <ul style="list-style-type: none"> The TE bit of UiC1 register= 1 (transmission enabled) The TI bit of UiC1 register = 0 (data present in UiTB register)
Reception start condition	<ul style="list-style-type: none"> Before reception can start, the following requirements must be met (Note 1) <ul style="list-style-type: none"> The RE bit of UiC1 register= 1 (reception enabled) The TE bit of UiC1 register= 1 (transmission enabled) The TI bit of UiC1 register= 0 (data present in the UiTB register)
Interrupt request generation timing	When start or stop condition is detected, acknowledge undetected, and acknowledge detected
Error detection	<ul style="list-style-type: none"> Overflow error (Note 2) <ul style="list-style-type: none"> This error occurs if the serial I/O started receiving the next data before reading the UiRB register and received the 8th bit of the next data
Select function	<ul style="list-style-type: none"> Arbitration lost <ul style="list-style-type: none"> Timing at which the UiRB register's ABT bit is updated can be selected SDAi digital delay <ul style="list-style-type: none"> No digital delay or a delay of 2 to 8 UiBRG count source clock cycles selectable Clock phase setting <ul style="list-style-type: none"> With or without clock delay selectable

Note 1: When an external clock is selected, the conditions must be met while the external clock is in the high state.

Note 2: If an overrun error occurs, the value of UiRB register will be indeterminate. The IR bit of SiRIC register does not change.

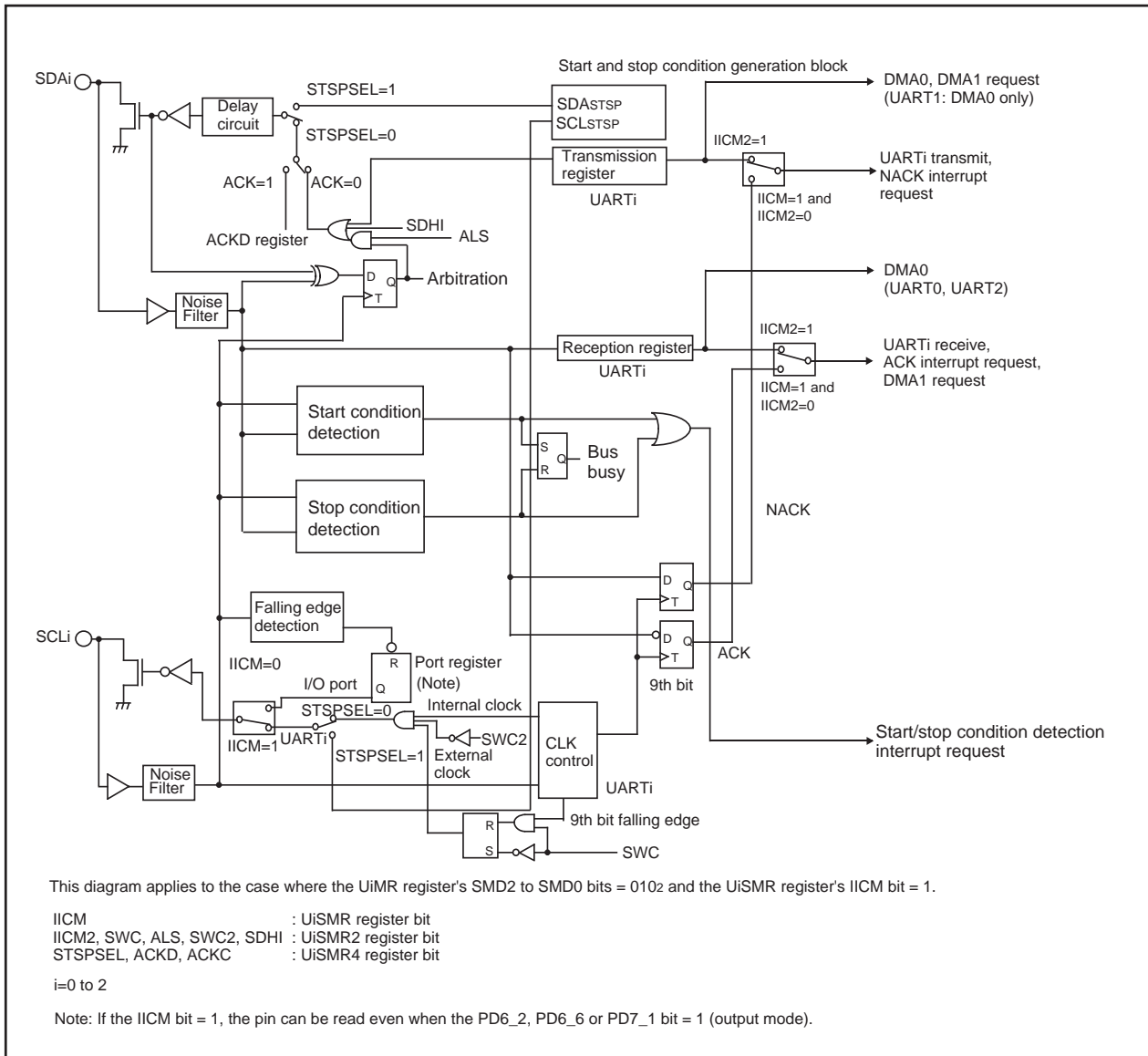


Figure 2.11.21. I²C Mode Block Diagram

Table 2.11.10. Registers to Be Used and Settings in I²C Mode (1) (Continued)

Register	Bit	Function	
		Master	Slave
UiTB (Note 3)	0 to 7	Set transmission data	Set transmission data
UiRB (Note 3)	0 to 7	Reception data can be read	Reception data can be read
	8	ACK or NACK is set in this bit	ACK or NACK is set in this bit
	ABT	Arbitration lost detection flag	Invalid
	OER	Overrun error flag	Overrun error flag
UiBRG	0 to 7	Set a transfer rate	Invalid
UiMR (Note 3)	SMD2 to SMD0	Set to '0102'	Set to '0102'
	CKDIR	Set to "0"	Set to "1"
	IOPOL	Set to "0"	Set to "0"
UiC0	CLK1, CLK0	Select the count source for the UiBRG register	Invalid
	CRS	Invalid because CRD = 1	Invalid because CRD = 1
	TXEPT	Transmit buffer empty flag	Transmit buffer empty flag
	CRD	Set to "1"	Set to "1"
	NCH	Set to "1" (Note 2)	Set to "1" (Note 2)
	CKPOL	Set to "0"	Set to "0"
	UFORM	Set to "1"	Set to "1"
UiC1	TE	Set this bit to "1" to enable transmission	Set this bit to "1" to enable transmission
	TI	Transmit buffer empty flag	Transmit buffer empty flag
	RE	Set this bit to "1" to enable reception	Set this bit to "1" to enable reception
	RI	Reception complete flag	Reception complete flag
	U2IRS (Note 1)	Invalid	Invalid
	U2RRM (Note 1), UiLCH, UiERE	Set to "0"	Set to "0"
UiSMR	IICM	Set to "1"	Set to "1"
	ABC	Select the timing at which arbitration-lost is detected	Invalid
	BBS	Bus busy flag	Bus busy flag
	3 to 7	Set to "0"	Set to "0"
UiSMR2	IICM2	Refer to Table 2.11.12	Refer to Table 2.11.12
	CSC	Set this bit to "1" to enable clock synchronization	Set to "0"
	SWC	Set this bit to "1" to have SCLi output fixed to "L" at the falling edge of the 9th bit of clock	Set this bit to "1" to have SCLi output fixed to "L" at the falling edge of the 9th bit of clock
	ALS	Set this bit to "1" to have SDAi output stopped when arbitration-lost is detected	Set to "0"
	STAC	Set to "0"	Set this bit to "1" to initialize UARTi at start condition detection
	SWC2	Set this bit to "1" to have SCLi output forcibly pulled low	Set this bit to "1" to have SCLi output forcibly pulled low
	SDHI	Set this bit to "1" to disable SDAi output	Set this bit to "1" to disable SDAi output
UiSMR3	7	Set to "0"	Set to "0"
	0, 2, 4 and NODC	Set to "0"	Set to "0"
	CKPH	Refer to Table 2.11.12	Refer to Table 2.11.12
	DL2 to DL0	Set the amount of SDAi digital delay	Set the amount of SDAi digital delay

i=0 to 2

Notes:

1. Set the U0C1 and U1C1 register bit 4 and bit 5 to "0". The U0IRS, U1IRS, U0RRM and U1RRM bits are in the UCON register.
2. TxD2 pin is N channel open-drain output. Set the NCH bit in the U2C0 register to "0".
3. Not all register bits are described above. Set those bits to "0" when writing to the registers in I²C mode.

Table 2.11.11. Registers to Be Used and Settings in I²C Mode (2) (Continued)

Register	Bit	Function	
		Master	Slave
UiSMR4	STAREQ	Set this bit to "1" to generate start condition	Set to "0"
	RSTAREQ	Set this bit to "1" to generate restart condition	Set to "0"
	STPREQ	Set this bit to "1" to generate stop condition	Set to "0"
	STSPSEL	Set this bit to "1" to output each condition	Set to "0"
	ACKD	Select ACK or NACK	Select ACK or NACK
	ACKC	Set this bit to "1" to output ACK data	Set this bit to "1" to output ACK data
	SCLHI	Set this bit to "1" to have SCLi output stopped when stop condition is detected	Set to "0"
	SWC9	Set to "0"	Set this bit to "1" to set the SCLi to "L" hold at the next falling edge of the 9th bit of clock
IFSR2A	IFSR26, ISFR27	Set to "1"	Set to "1"
UCON	U0IRS, U1IRS	Invalid	Invalid
	2 to 7	Set to "0"	Set to "0"

i=0 to 2

Table 2.11.12. I²C Mode Functions

Function	Clock synchronous serial I/O mode (SMD2 to SMD0 = 0012, IICM = 0)	I ² C mode (SMD2 to SMD0 = 0102, IICM = 1)			
		IICM2 = 0 (NACK/ACK interrupt)		IICM2 = 1 (UART transmit/ receive interrupt)	
		CKPH = 0 (No clock delay)	CKPH = 1 (Clock delay)	CKPH = 0 (No clock delay)	CKPH = 1 (Clock delay)
Factor of interrupt number 6, 7 and 10 (Note 1, 5, 7)	————	Start condition detection or stop condition detection (Refer to “Table 2.11.13. STSPSEL Bit Functions”)			
Factor of interrupt number 15, 17 and 19 (Note 1, 6)	UARTi transmission Transmission started or completed (selected by UiIRS)	No acknowledgment detection (NACK) Rising edge of SCLi 9th bit	UARTi transmission Rising edge of SCLi 9th bit	UARTi transmission Falling edge of SCLi next to the 9th bit	
Factor of interrupt number 16, 18 and 20 (Note 1, 6)	UARTi reception When 8th bit received CKPOL = 0 (rising edge) CKPOL = 1 (falling edge)	Acknowledgment detection (ACK) Rising edge of SCLi 9th bit	UARTi reception Falling edge of SCLi 9th bit		
Timing for transferring data from the UART reception shift register to the UiRB register	CKPOL = 0 (rising edge) CKPOL = 1 (falling edge)	Rising edge of SCLi 9th bit	Falling edge of SCLi 9th bit	Falling and rising edges of SCLi 9th bit	
UARTi transmission output delay	Not delayed	Delayed			
Functions of P63, P67 and P70 pins	TxDi output	SDAi input/output			
Functions of P62, P66 and P71 pins	RxDi input	SCLi input/output			
Functions of P61, P65 and P72 pins	CLKi input or output selected	———— (Cannot be used in I ² C mode)			
Noise filter width	15ns	200ns			
Read RxDi and SCLi pin levels	Possible when the corresponding port direction bit = 0	Always possible no matter how the corresponding port direction bit is set			
Initial value of TxDi and SDAi outputs	CKPOL = 0 (H) CKPOL = 1 (L)	The value set in the port register before setting I ² C mode (Note 2)			
Initial and end values of SCLi	————	H	L	H	L
DMA1 factor (Refer to Fig 2.11.22)	UARTi reception	Acknowledgment detection (ACK)	UARTi reception Falling edge of SCLi 9th bit		
Store received data	1st to 8th bits are stored in UiRB register bit 0 to bit 7	1st to 8th bits are stored in UiRB register bit 7 to bit 0	1st to 7th bits are stored in UiRB register bit 6 to bit 0, with 8th bit stored in UiRB register bit 8		
			1st to 8th bits are stored in UiRB register bit 7 to bit 0 (Note 3)		
Read received data	UiRB register status is read directly as is	Read UiRB register Bit 6 to bit 0 as bit 7 to bit 1, and bit 8 as bit 0 (Note 4)			

i = 0 to 2

Note 1: If the source or cause of any interrupt is changed, the IR bit in the interrupt control register for the changed interrupt may inadvertently be set to “1” (interrupt requested). (Refer to “precautions for interrupts” of the Usage Notes Reference Book.) If one of the bits shown below is changed, the interrupt source, the interrupt timing, etc. change. Therefore, always be sure to clear the IR bit to “0” (interrupt not requested) after changing those bits.
SMD2 to SMD0 bits in the UiMR register, IICM bit in the UiSMR register, IICM2 bit in the UiSMR2 register, CKPH bit in the UiSMR3 register

Note 2: Set the initial value of SDAi output while the UiMR register’s SMD2 to SMD0 bits = ‘0002’ (serial I/O disabled).

Note 3: Second data transfer to UiRB register (Rising edge of SCLi 9th bit)

Note 4: First data transfer to UiRB register (Falling edge of SCLi 9th bit)

Note 5: Refer to “Figure 2.11.24. STSPSEL Bit Functions”.

Note 6: Refer to “Figure 2.11.22. Transfer to UiRB Register and Interrupt Timing”

Note 7: When using UART0, be sure to set the IFSR26 bit in the IFSR2A register to “1” (cause of interrupt: UART0 bus collision).

When using UART1, be sure to set the IFSR27 bit in the IFSR2A register to “1” (cause of interrupt: UART1 bus collision).

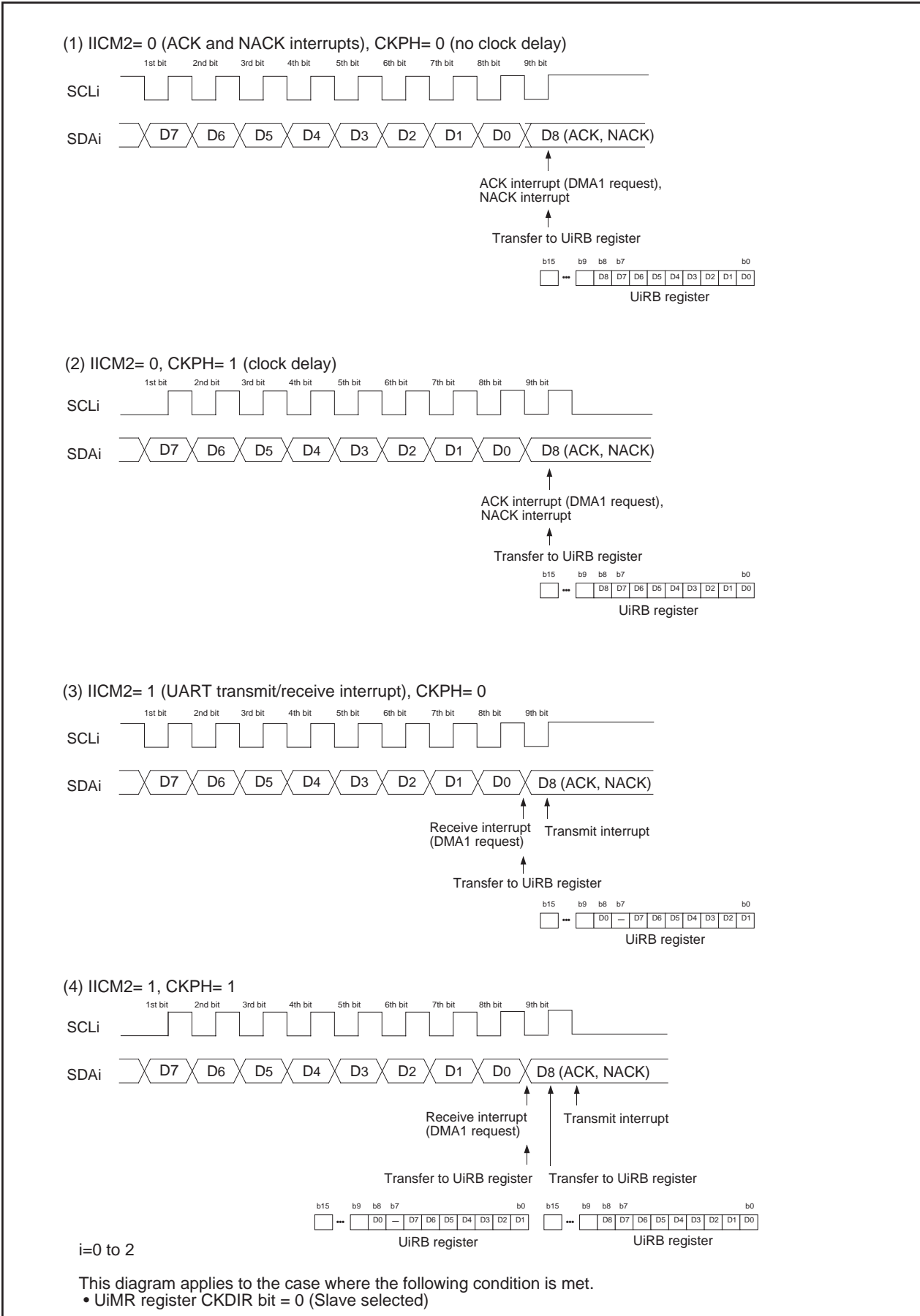


Figure 2.11.22. Transfer to UiRB Register and Interrupt Timing

• Detection of Start and Stop Condition

Whether a start or a stop condition has been detected is determined.

A start condition-detected interrupt request is generated when the SDA_i pin changes state from high to low while the SCL_i pin is in the high state. A stop condition-detected interrupt request is generated when the SDA_i pin changes state from low to high while the SCL_i pin is in the high state.

Because the start and stop condition-detected interrupts share the interrupt control register and vector, check the UiSMR register's BBS bit to determine which interrupt source is requesting the interrupt.

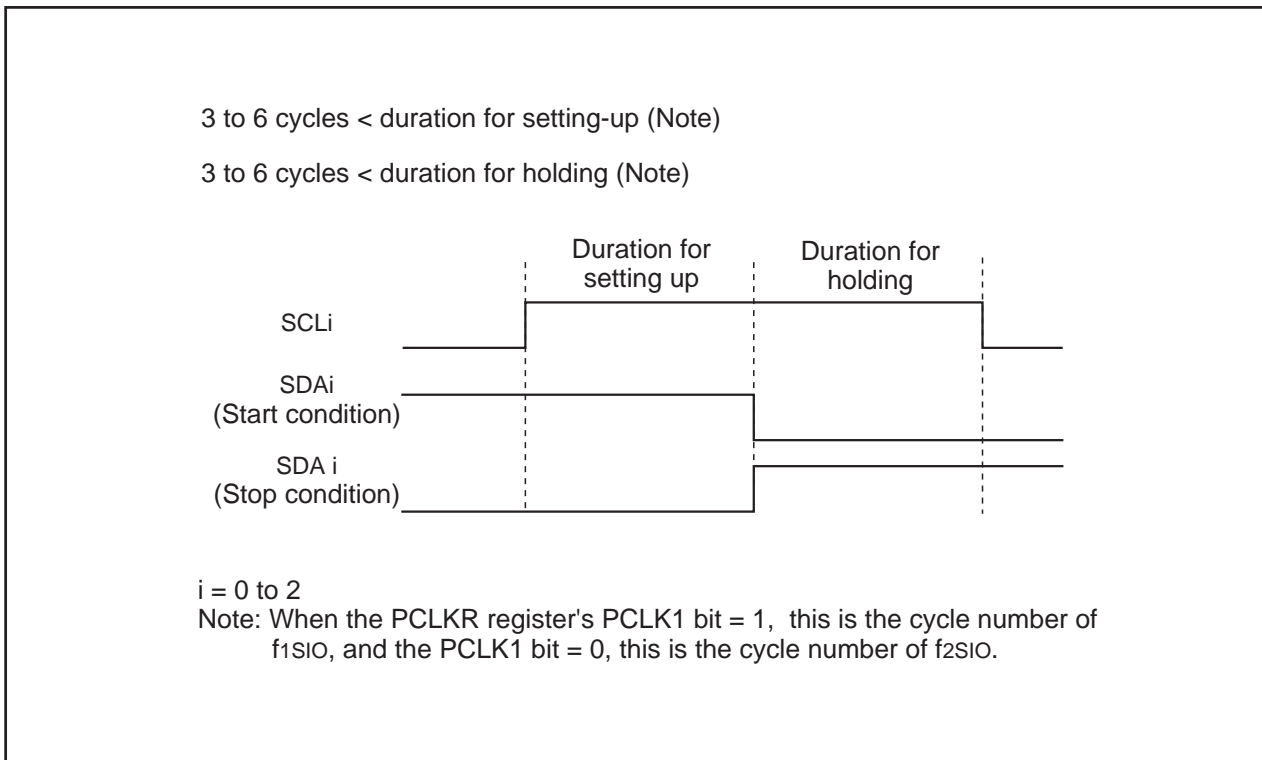


Figure 2.11.23. Detection of Start and Stop Condition

• Output of Start and Stop Condition

A start condition is generated by setting the UiSMR4 register ($i = 0$ to 2)'s STAREQ bit to "1" (start).

A restart condition is generated by setting the UiSMR4 register's RSTAREQ bit to "1" (start).

A stop condition is generated by setting the UiSMR4 register's STPREQ bit to "1" (start).

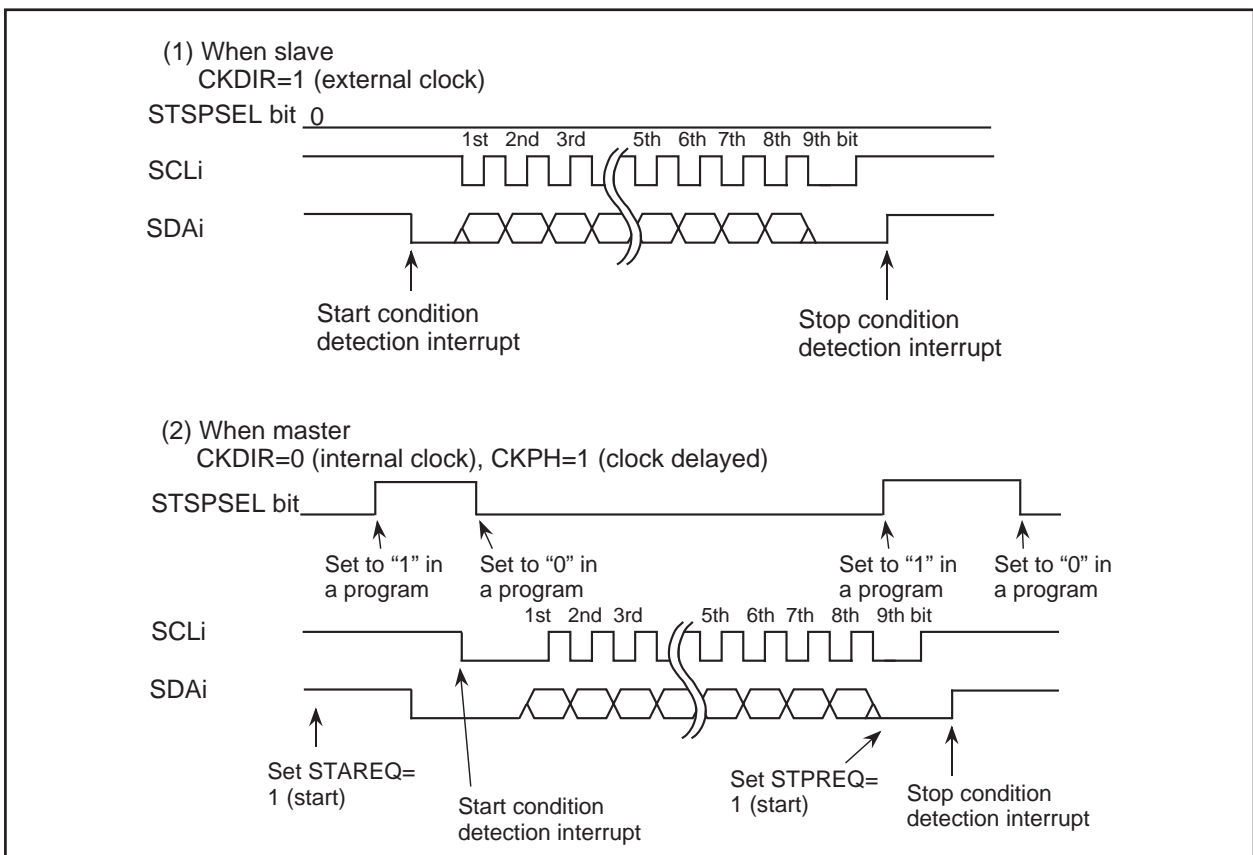
The output procedure is described below.

- (1) Set the STAREQ bit, RSTAREQ bit or STPREQ bit to "1" (start).
- (2) Set the STSPSEL bit in the UiSMR4 register to "1" (output).

The function of the STSPSEL bit is shown in Table 2.11.13 and Figure 2.11.24.

Table 2.11.13. STSPSEL Bit Functions

Function	STSPSEL = 0	STSPSEL = 1
Output of SCLi and SDAi pins	Output of transfer clock and data Output of start/stop condition is accomplished by a program using ports (not automatically generated in hardware)	Output of a start/stop condition according to the STAREQ, RSTAREQ and STPREQ bit
Star/stop condition interrupt request generation timing	Start/stop condition detection	Finish generating start/stop condition

**Figure 2.11.24. STSPSEL Bit Functions**

- **Arbitration**

Unmatching of the transmit data and SDAi pin input data is checked synchronously with the rising edge of SCLi. Use the UiSMR register's ABC bit to select the timing at which the UiRB register's ABT bit is updated. If the ABC bit = 0 (updated bitwise), the ABT bit is set to "1" at the same time unmatching is detected during check, and is cleared to "0" when not detected. In cases when the ABC bit is set to "1", if unmatching is detected even once during check, the ABT bit is set to "1" (unmatching detected) at the falling edge of the clock pulse of 9th bit. If the ABT bit needs to be updated bitwise, clear the ABT bit to "0" (undetected) after detecting acknowledge in the first byte, before transferring the next byte.

Setting the UiSMR2 register's ALS bit to "1" (SDA output stop enabled) causes arbitration-lost to occur, in which case the SDAi pin is placed in the high-impedance state at the same time the ABT bit is set to "1" (unmatching detected).

• Transfer Clock

Data is transmitted/received using a transfer clock like the one shown in Figure 2.11.24.

The UiSMR2 register's CSC bit is used to synchronize the internally generated clock (internal SCLi) and an external clock supplied to the SCLi pin. In cases when the CSC bit is set to "1" (clock synchronization enabled), if a falling edge on the SCLi pin is detected while the internal SCLi is high, the internal SCLi goes low, at which time the UiBRG register value is reloaded with and starts counting in the low-level interval. If the internal SCLi changes state from low to high while the SCLi pin is low, counting stops, and when the SCLi pin goes high, counting restarts.

In this way, the UARTi transfer clock is comprised of the logical product of the internal SCLi and SCLi pin signal. The transfer clock works from a half period before the falling edge of the internal SCLi 1st bit to the rising edge of the 9th bit. To use this function, select an internal clock for the transfer clock. The UiSMR2 register's SWC bit allows to select whether the SCLi pin should be fixed to or freed from low-level output at the falling edge of the 9th clock pulse.

If the UiSMR4 register's SCLHI bit is set to "1" (enabled), SCLi output is turned off (placed in the high-impedance state) when a stop condition is detected.

Setting the UiSMR2 register's SWC2 bit = 1 (0 output) makes it possible to forcibly output a low-level signal from the SCLi pin even while sending or receiving data. Clearing the SWC2 bit to "0" (transfer clock) allows the transfer clock to be output from or supplied to the SCLi pin, instead of outputting a low-level signal.

If the UiSMR4 register's SWC9 bit is set to "1" (SCL hold low enabled) when the UiSMR3 register's CKPH bit = 1, the SCLi pin is fixed to low-level output at the falling edge of the clock pulse next to the ninth. Setting the SWC9 bit = 0 (SCL hold low disabled) frees the SCLi pin from low-level output.

• SDA Output

The data written to the UiTB register bit 7 to bit 0 (D7 to D0) is sequentially output beginning with D7. The ninth bit (D8) is ACK or NACK.

The initial value of SDAi transmit output can only be set when IICM = 1 (I²C mode) and the UiMR register's SMD2 to SMD0 bits = '0002' (serial I/O disabled).

The UiSMR3 register's DL2 to DL0 bits allow to add no delays or a delay of 2 to 8 UiBRG count source clock cycles to SDAi output.

Setting the UiSMR2 register's SDHI bit = 1 (SDA output disabled) forcibly places the SDAi pin in the high-impedance state. Do not write to the SDHI bit synchronously with the rising edge of the UARTi transfer clock. This is because the ABT bit may inadvertently be set to "1" (detected).

• SDA Input

When the IICM2 bit = 0, the 1st to 8th bits (D7 to D0) of received data are stored in the UiRB register bit 7 to bit 0. The 9th bit (D8) is ACK or NACK.

When the IICM2 bit = 1, the 1st to 7th bits (D7 to D1) of received data are stored in the UiRB register bit 6 to bit 0 and the 8th bit (D0) is stored in the UiRB register bit 8. Even when the IICM2 bit = 1, providing the CKPH bit = 1, the same data as when the IICM2 bit = 0 can be read out by reading the UiRB register after the rising edge of the corresponding clock pulse of 9th bit.

• ACK and NACK

If the STSPSEL bit in the UiSMR4 register is set to “0” (start and stop conditions not generated) and the ACKC bit in the UiSMR4 register is set to “1” (ACK data output), the value of the ACKD bit in the UiSMR4 register is output from the SDAi pin.

If the IICM2 bit = 0, a NACK interrupt request is generated if the SDAi pin remains high at the rising edge of the 9th bit of transmit clock pulse. An ACK interrupt request is generated if the SDAi pin is low at the rising edge of the 9th bit of transmit clock pulse.

If ACKi is selected for the cause of DMA1 request, a DMA transfer can be activated by detection of an acknowledge.

• Initialization of Transmission/Reception

If a start condition is detected while the STAC bit = 1 (UARTi initialization enabled), the serial I/O operates as described below.

- The transmit shift register is initialized, and the content of the UiTB register is transferred to the transmit shift register. In this way, the serial I/O starts sending data synchronously with the next clock pulse applied. However, the UARTi output value does not change state and remains the same as when a start condition was detected until the first bit of data is output synchronously with the input clock.
- The receive shift register is initialized, and the serial I/O starts receiving data synchronously with the next clock pulse applied.
- The SWC bit is set to “1” (SCL wait output enabled). Consequently, the SCLi pin is pulled low at the falling edge of the ninth clock pulse.

Note that when UARTi transmission/reception is started using this function, the TI does not change state. Note also that when using this function, the selected transfer clock should be an external clock.

2.11.5 Special Mode 2

Multiple slaves can be serially communicated from one master. Transfer clock polarity and phase are selectable. Table 2.11.14 lists the specifications of Special Mode 2. Table 2.11.15 lists the registers used in Special Mode 2 and the register values set. Figure 2.11.25 shows communication control example for Special Mode 2.

Table 2.11.14. Special Mode 2 Specifications

Item	Specification
Transfer data format	<ul style="list-style-type: none"> Transfer data length: 8 bits
Transfer clock	<ul style="list-style-type: none"> Master mode UiMR(i=0 to 2) register's CKDIR bit = "0" (internal clock) : $f_j / 2^{(n+1)}$ $f_j = f_{1SIO}, f_{2SIO}, f_{8SIO}, f_{32SIO}$. n: Setting value of UiBRG register 0016 to FF16 Slave mode CKDIR bit = "1" (external clock selected) : Input from CLKi pin
Transmit/receive control	Controlled by input/output ports
Transmission start condition	<ul style="list-style-type: none"> Before transmission can start, the following requirements must be met (Note 1) <ul style="list-style-type: none"> The TE bit of UiC1 register= 1 (transmission enabled) The TI bit of UiC1 register = 0 (data present in UiTB register)
Reception start condition	<ul style="list-style-type: none"> Before reception can start, the following requirements must be met (Note 1) <ul style="list-style-type: none"> The RE bit of UiC1 register= 1 (reception enabled) The TE bit of UiC1 register= 1 (transmission enabled) The TI bit of UiC1 register= 0 (data present in the UiTB register)
Interrupt request generation timing	<ul style="list-style-type: none"> For transmission, one of the following conditions can be selected <ul style="list-style-type: none"> The UiIRS bit of UiC1 register = 0 (transmit buffer empty): when transferring data from the UiTB register to the UARTi transmit register (at start of transmission) The UiIRS bit =1 (transfer completed): when the serial I/O finished sending data from the UARTi transmit register For reception When transferring data from the UARTi receive register to the UiRB register (at completion of reception)
Error detection	<ul style="list-style-type: none"> Overrun error (Note 2) This error occurs if the serial I/O started receiving the next data before reading the UiRB register and received the 7th bit of the next data
Select function	<ul style="list-style-type: none"> Clock phase setting Selectable from four combinations of transfer clock polarities and phases

Note 1: When an external clock is selected, the conditions must be met while if the UiC0 register's CKPOL bit = "0" (transmit data output at the falling edge and the receive data taken in at the rising edge of the transfer clock), the external clock is in the high state; if the UiC0 register's CKPOL bit = "1" (transmit data output at the rising edge and the receive data taken in at the falling edge of the transfer clock), the external clock is in the low state.

Note 2: If an overrun error occurs, the value of UiRB register will be indeterminate. The IR bit of SiRIC register does not change.

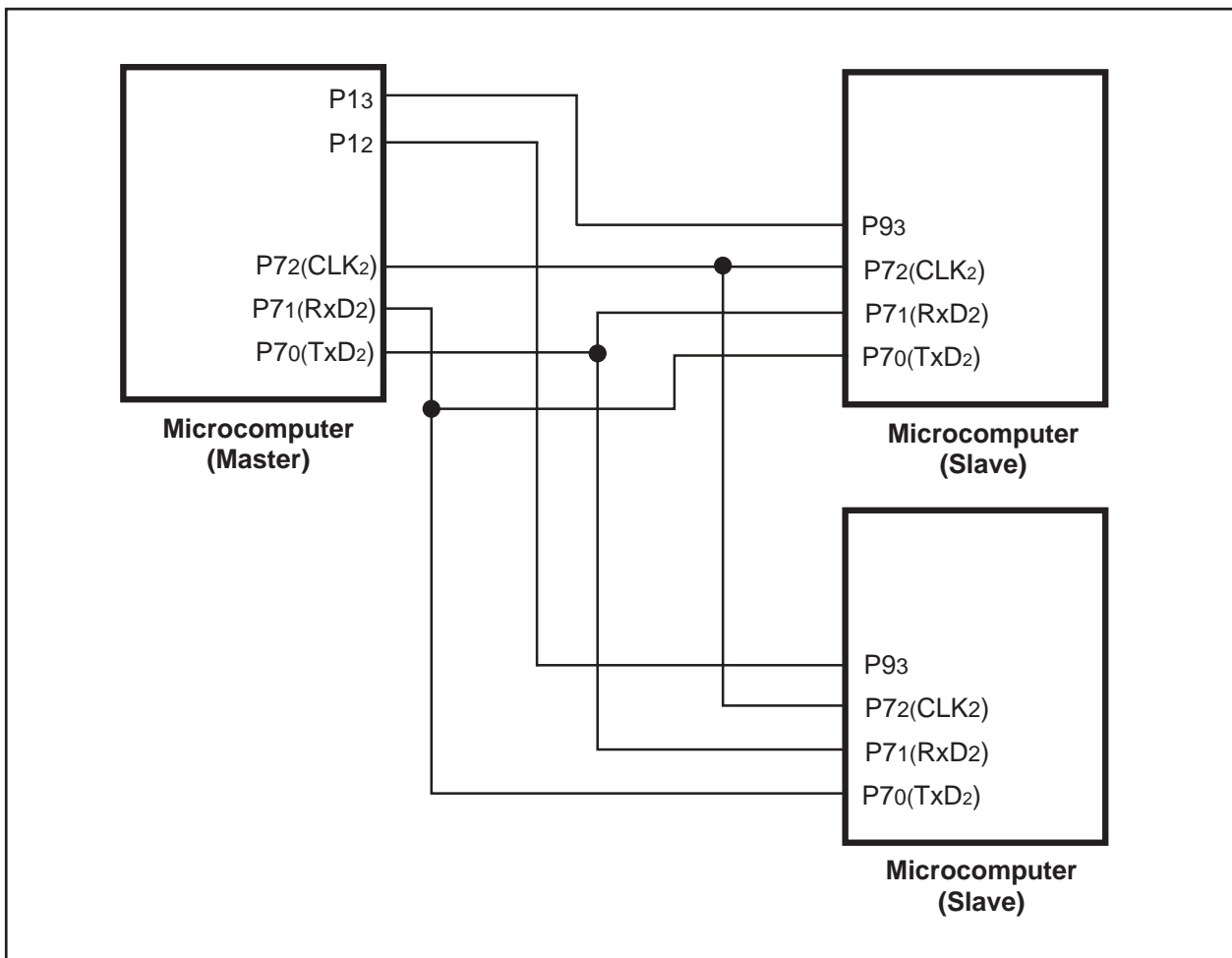


Figure 2.11.25. Serial Bus Communication Control Example (UART2)

Table 2.11.15. Registers to Be Used and Settings in Special Mode 2

Register	Bit	Function
UiTB(Note3)	0 to 7	Set transmission data
UiRB(Note3)	0 to 7	Reception data can be read
	OER	Overrun error flag
UiBRG	0 to 7	Set a transfer rate
UiMR(Note3)	SMD2 to SMD0	Set to '0012'
	CKDIR	Set this bit to "0" for master mode or "1" for slave mode
	IOPOL	Set to "0"
UiC0	CLK1, CLK0	Select the count source for the UiBRG register
	CRS	Invalid because CRD = 1
	TXEPT	Transmit register empty flag
	CRD	Set to "1"
	NCH	Select TxDi pin output format(Note 2)
	CKPOL	Clock phases can be set in combination with the UiSMR3 register's CKPH bit
	UFORM	Set to "0"
UiC1	TE	Set this bit to "1" to enable transmission
	TI	Transmit buffer empty flag
	RE	Set this bit to "1" to enable reception
	RI	Reception complete flag
	U2IRS (Note 1)	Select UART2 transmit interrupt cause
	U2RRM(Note 1), U2LCH, UiERE	Set to "0"
UiSMR	0 to 7	Set to "0"
UiSMR2	0 to 7	Set to "0"
UiSMR3	CKPH	Clock phases can be set in combination with the UiC0 register's CKPOL bit
	NODC	Set to "0"
	0, 2, 4 to 7	Set to "0"
UiSMR4	0 to 7	Set to "0"
UCON	U0IRS, U1IRS	Select UART0 and UART1 transmit interrupt cause
	U0RRM, U1RRM	Set to "0"
	CLKMD0	Invalid because CLKMD1 = 0
	CLKMD1, RCSP, 7	Set to "0"

Note 1: Set the U0C0 and U1C1 register bit 4 and bit 5 to "0". The U0IRS, U1IRS, U0RRM and U1RRM bits are in the UCON register.

Note 2: TxD2 pin is N channel open-drain output. Set the U2C0 register's NCH bit to "0".

Note 3: Not all register bits are described above. Set those bits to "0" when writing to the registers in Special Mode 2.

i = 0 to 2

- **Clock Phase Setting Function**

One of four combinations of transfer clock phases and polarities can be selected using the UiSMR3 register's CKPH bit and the UiC0 register's CKPOL bit.

Make sure the transfer clock polarity and phase are the same for the master and slaves to be communicated.

(a) Master (Internal Clock)

Figure 2.11.26 shows the transmission and reception timing in master (internal clock).

(b) Slave (External Clock)

Figure 2.11.27 shows the transmission and reception timing (CKPH=0) in slave (external clock) while Figure 2.11.28 shows the transmission and reception timing (CKPH=1) in slave (external clock).

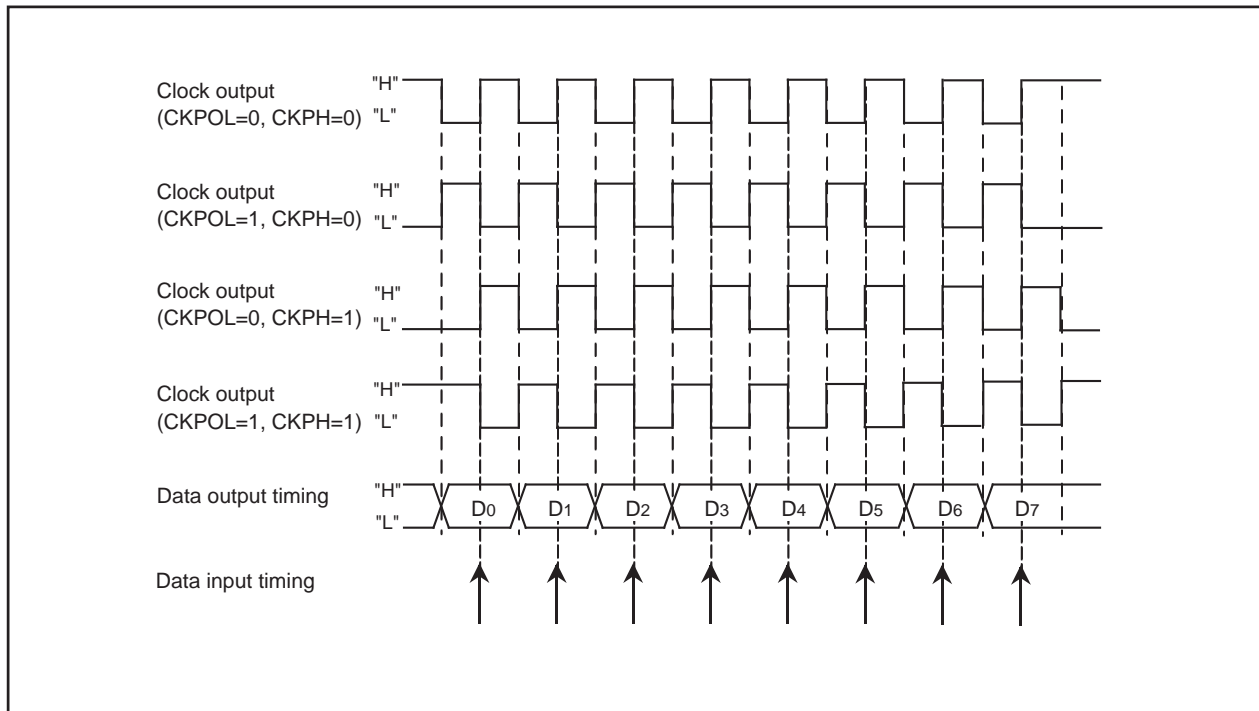


Figure 2.11.26. Transmission and Reception Timing in Master Mode (Internal Clock)

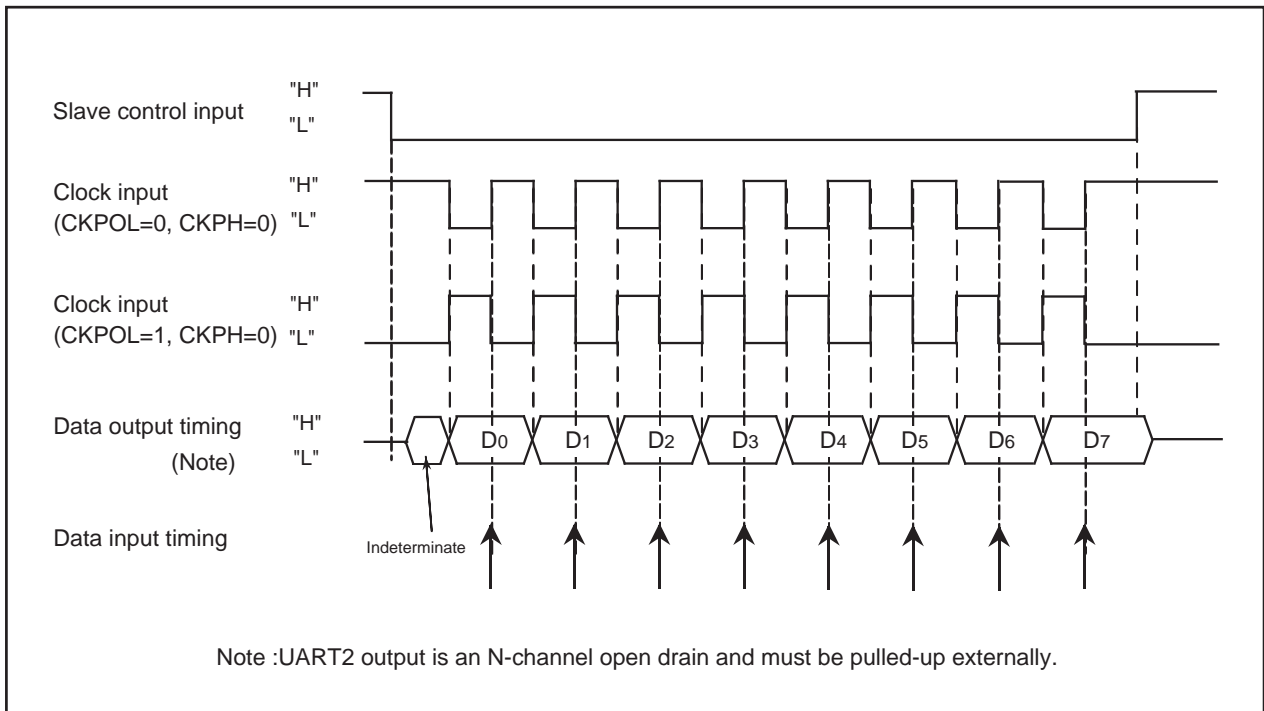


Figure 2.11.27. Transmission and Reception Timing (CKPH=0) in Slave Mode (External Clock)

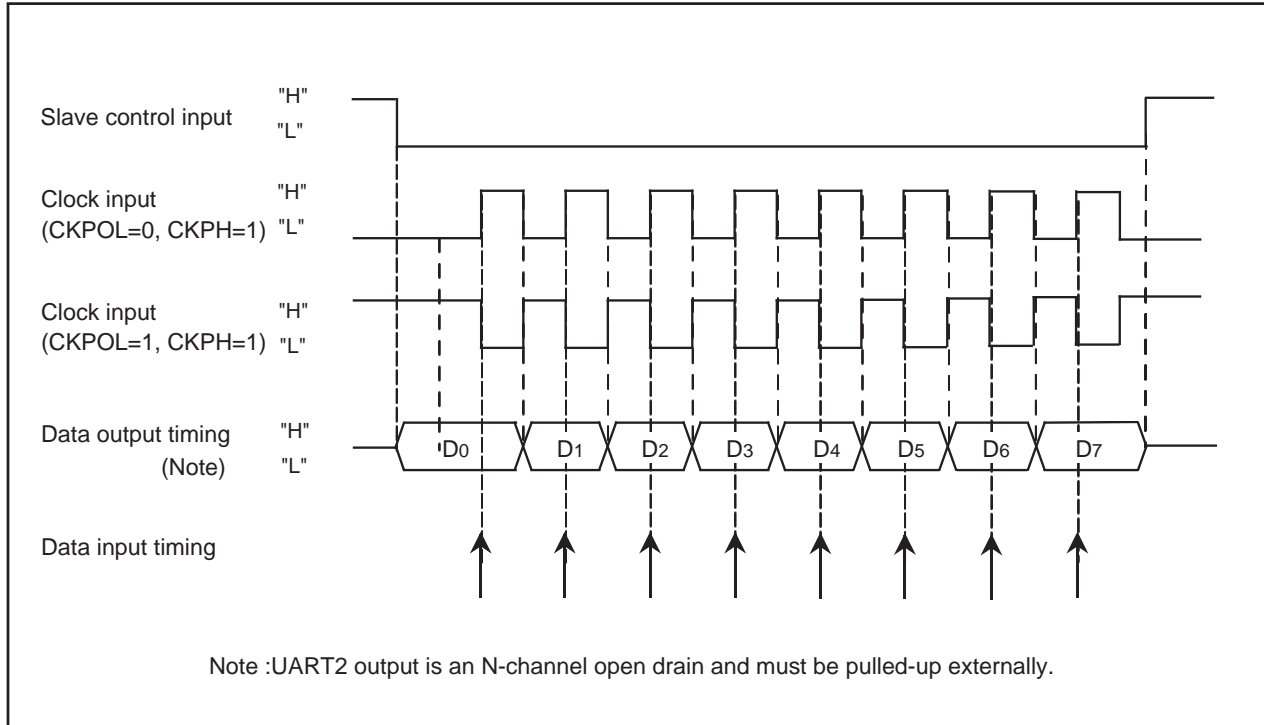


Figure 2.11.28. Transmission and Reception Timing (CKPH=1) in Slave Mode (External Clock)

2.11.6 Special Mode 3 (IE mode)

In this mode, one bit of IEBus is approximated with one byte of UART mode waveform.

Table 2.11.16 lists the registers used in IE mode and the register values set. Figure 2.11.29 shows the functions of bus collision detect function related bits.

If the TxDi pin (i = 0 to 2) output level and RxDi pin input level do not match, a UARTi bus collision detect interrupt request is generated.

Use the IFSR2A register's IFSR26 and IFSR27 bits to enable the UART0/UART1 bus collision detect function.

Table 2.11.16. Registers to Be Used and Settings in IE Mode

Register	Bit	Function
UiTB	0 to 8	Set transmission data
UiRB(Note3)	0 to 8	Reception data can be read
	OER,FER,PER,SUM	Error flag
UiBRG	0 to 7	Set a transfer rate
UiMR	SMD2 to SMD0	Set to '1102'
	CKDIR	Select the internal clock or external clock
	STPS	Set to "0"
	PRY	Invalid because PRYE=0
	PRYE	Set to "0"
	IOPOL	Select the TxD/RxD input/output polarity
UiC0	CLK1, CLK0	Select the count source for the UiBRG register
	CRS	Invalid because CRD=1
	TXEPT	Transmit register empty flag
	CRD	Set to "1"
	NCH	Select TxDi pin output mode (Note 2)
	CKPOL	Set to "0"
	UFORM	Set to "0"
UiC1	TE	Set this bit to "1" to enable transmission
	TI	Transmit buffer empty flag
	RE	Set this bit to "1" to enable reception
	RI	Reception complete flag
	U2IRS (Note 1)	Select the source of UART2 transmit interrupt
	UiRRM (Note 1), UiLCH, UiERE	Set to "0"
UiSMR	0 to 3, 7	Set to "0"
	ABSCS	Select the sampling timing at which to detect a bus collision
	ACSE	Set this bit to "1" to use the auto clear function of transmit enable bit
	SSS	Select the transmit start condition
UiSMR2	0 to 7	Set to "0"
UiSMR3	0 to 7	Set to "0"
UiSMR4	0 to 7	Set to "0"
IFSR2A	IFSR26, IFSR27	Set to "1"
UCON	U0IRS, U1IRS	Select the source of UART0/UART1 transmit interrupt
	U0RRM, U1RRM	Set to "0"
	CLKMD0	Invalid because CLKMD1 = 0
	CLKMD1,RCSP,7	Set to "0"

Note 1: Set the U0C0 and U1C1 registers bit 4 and bit 5 to "0". The U0IRS, U1IRS, U0RRM and U1RRM bits are in the UCON register.

Note 2: TxD2 pin is N channel open-drain output. Set the U2C0 register's NCH bit to "0".

Note 3: Not all register bits are described above. Set those bits to "0" when writing to the registers in IEmode. i= 0 to 2

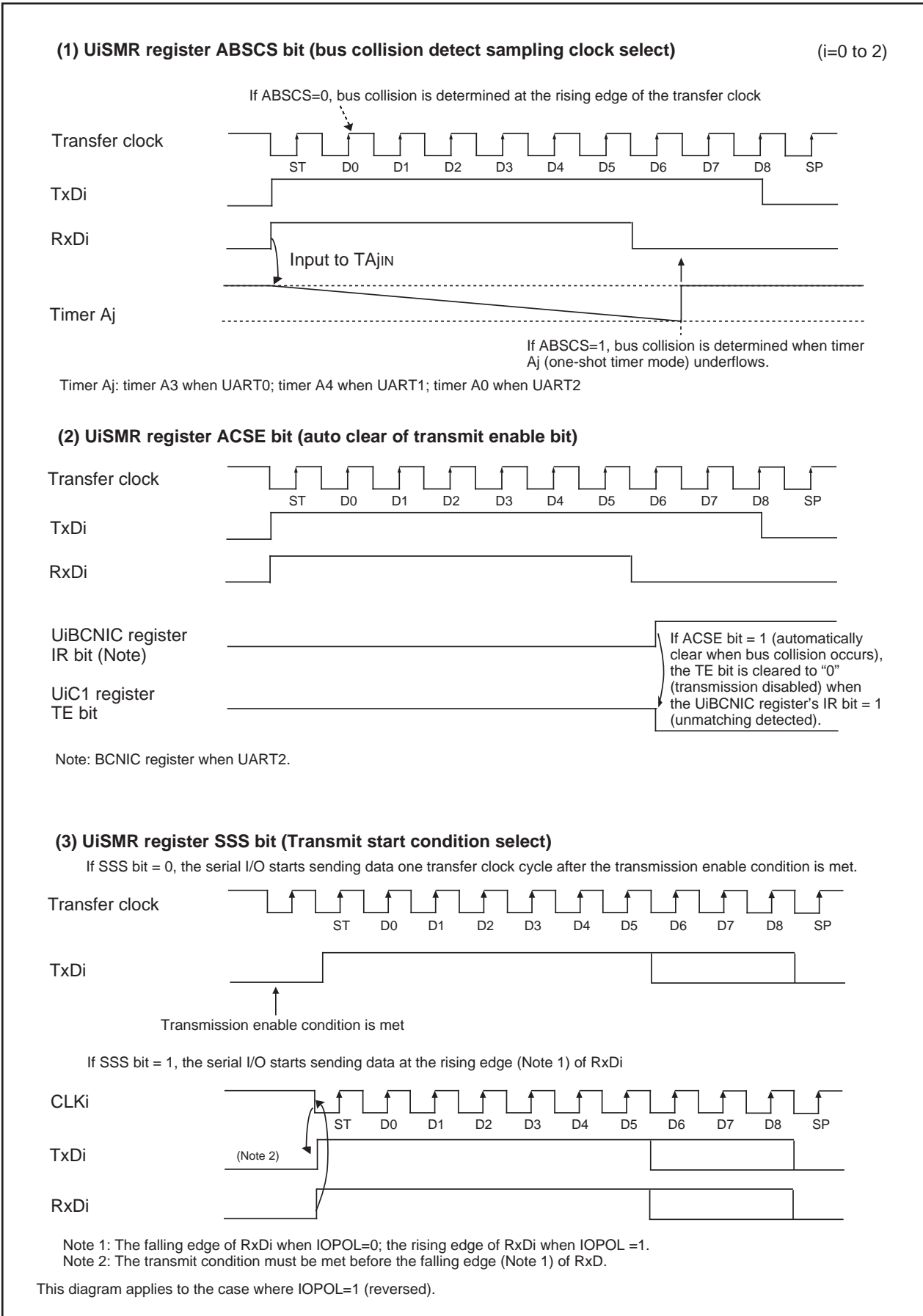


Figure 2.11.29. Bus Collision Detect Function-Related Bits

2.11.7 Special Mode 4 (SIM Mode) (UART2)

Based on UART mode, this is an SIM interface compatible mode. Direct and inverse formats can be implemented, and this mode allows to output a low from the TxD2 pin when a parity error is detected.

Tables 2.11.17 lists the specifications of SIM mode. Table 2.11.18 lists the registers used in the SIM mode and the register values set.

Table 2.11.17. SIM Mode Specifications

Item	Specification
Transfer data format	<ul style="list-style-type: none"> • Direct format • Inverse format
Transfer clock	<ul style="list-style-type: none"> • U2MR register's CKDIR bit = "0" (internal clock) : $f_i / 16(n+1)$ $f_i = f_{1SIO}, f_{2SIO}, f_{8SIO}, f_{32SIO}$. n: Setting value of U2BRG register 00₁₆ to FF₁₆ • CKDIR bit = "1" (external clock) : $f_{EXT} / 16(n+1)$ f_{EXT}: Input from CLK2 pin. n: Setting value of U2BRG register 00₁₆ to FF₁₆
Transmission start condition	<ul style="list-style-type: none"> • Before transmission can start, the following requirements must be met <ul style="list-style-type: none"> – The TE bit of U2C1 register= 1 (transmission enabled) – The TI bit of U2C1 register = 0 (data present in U2TB register)
Reception start condition	<ul style="list-style-type: none"> • Before reception can start, the following requirements must be met <ul style="list-style-type: none"> – The RE bit of U2C1 register= 1 (reception enabled) – Start bit detection
Interrupt request generation timing (Note 2)	<ul style="list-style-type: none"> • For transmission When the serial I/O finished sending data from the U2TB transfer register (U2IRS bit =1) • For reception When transferring data from the UART2 receive register to the U2RB register (at completion of reception)
Error detection	<ul style="list-style-type: none"> • Overrun error (Note 1) This error occurs if the serial I/O started receiving the next data before reading the U2RB register and received the bit one before the last stop bit of the next data • Framing error This error occurs when the number of stop bits set is not detected • Parity error During reception, if a parity error is detected, parity error signal is output from the TxD2 pin. During transmission, a parity error is detected by the level of input to the RxD2 pin when a transmission interrupt occurs • Error sum flag This flag is set (= 1) when any of the overrun, framing, and parity errors is encountered

Note 1: If an overrun error occurs, the value of U2RB register will be indeterminate. The IR bit of S2RIC register does not change.

Note 2: A transmit interrupt request is generated by setting the U2C1 register U2IRS bit to "1" (transmission complete) and U2ERE bit to "1" (error signal output) after reset. Therefore, when using SIM mode, be sure to clear the IR bit to "0" (no interrupt request) after setting these bits.

Table 2.11.18. Registers to Be Used and Settings in SIM Mode

Register	Bit	Function
U2TB(Note)	0 to 7	Set transmission data
U2RB(Note)	0 to 7	Reception data can be read
	OER,FER,PER,SUM	Error flag
U2BRG	0 to 7	Set a transfer rate
U2MR	SMD2 to SMD0	Set to '1012'
	CKDIR	Select the internal clock or external clock
	STPS	Set to "0"
	PRY	Set this bit to "1" for direct format or "0" for inverse format
	PRYE	Set to "1"
	IOPOL	Set to "0"
U2C0	CLK1, CLK0	Select the count source for the U2BRG register
	CRS	Invalid because CRD=1
	TXEPT	Transmit register empty flag
	CRD	Set to "1"
	NCH	Set to "0"
	CKPOL	Set to "0"
	UFORM	Set this bit to "0" for direct format or "1" for inverse format
U2C1	TE	Set this bit to "1" to enable transmission
	TI	Transmit buffer empty flag
	RE	Set this bit to "1" to enable reception
	RI	Reception complete flag
	U2IRS	Set to "1"
	U2RRM	Set to "0"
	U2LCH	Set this bit to "0" for direct format or "1" for inverse format
	U2ERE	Set to "1"
U2SMR(Note)	0 to 3	Set to "0"
U2SMR2	0 to 7	Set to "0"
U2SMR3	0 to 7	Set to "0"
U2SMR4	0 to 7	Set to "0"

Note: Not all register bits are described above. Set those bits to "0" when writing to the registers in SIM mode.

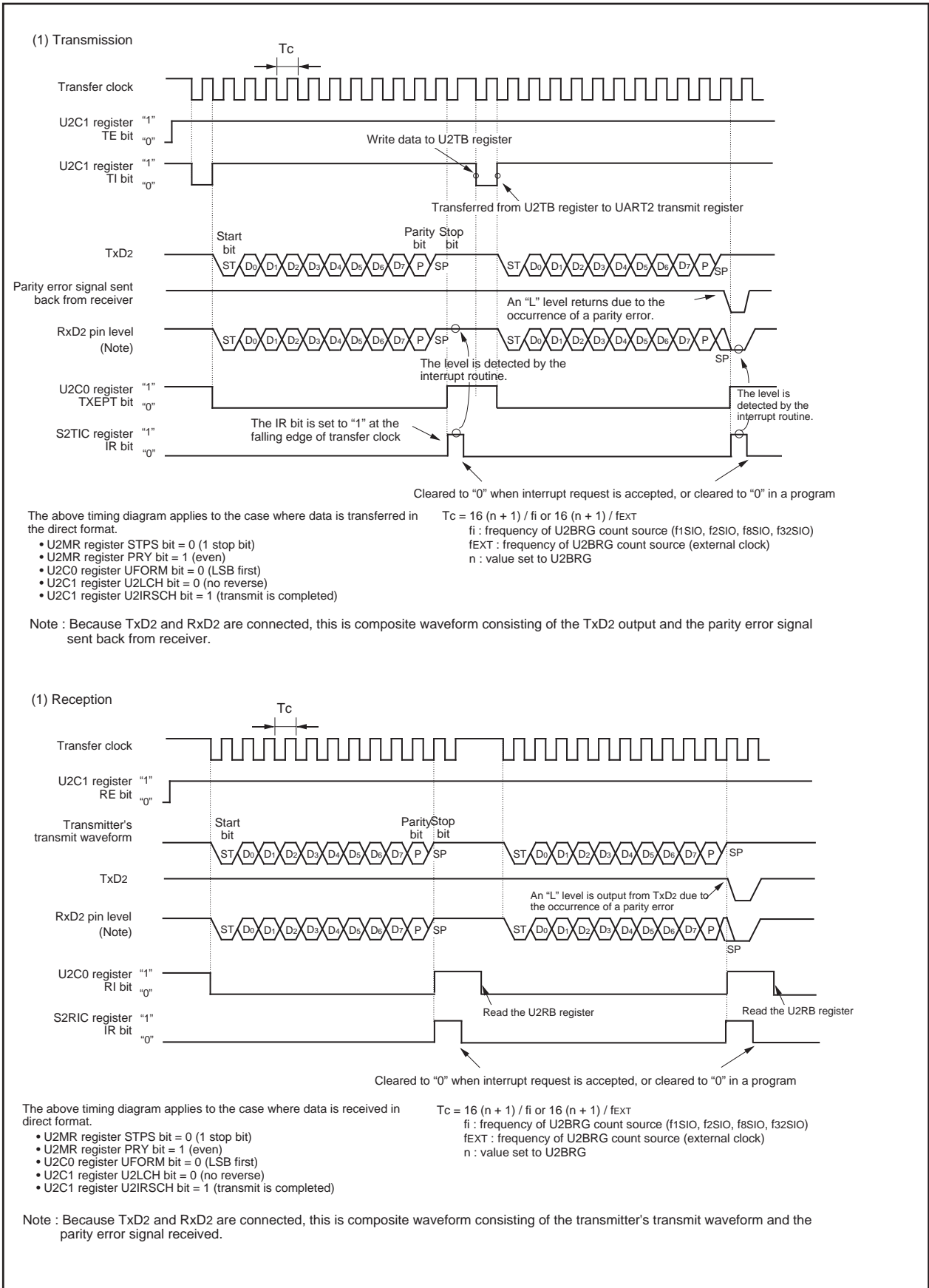


Figure 2.11.30. Transmit and Receive Timing in SIM Mode

Figure 2.11.31 shows the example of connecting the SIM interface. Connect TxD2 and RxD2 and apply pull-up.

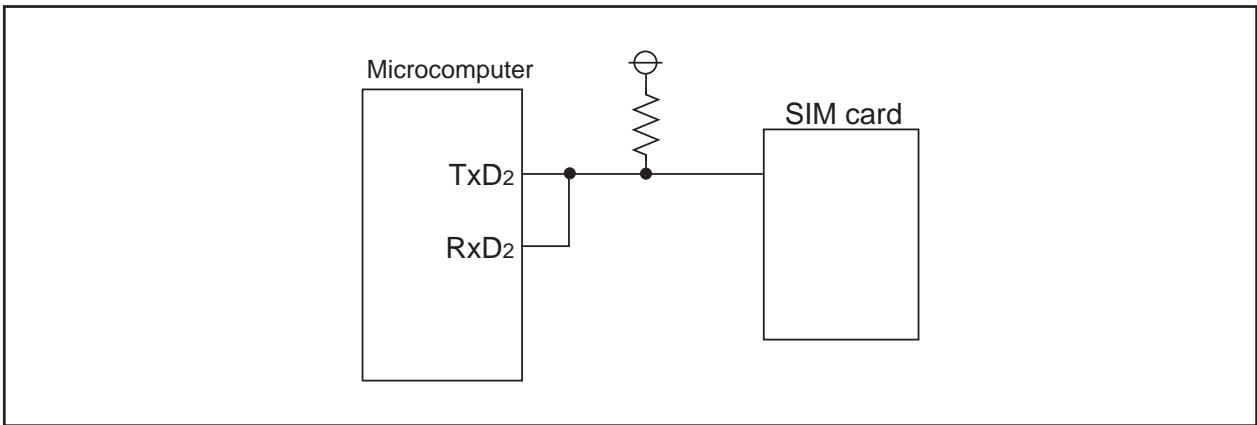


Figure 2.11.31. SIM Interface Connection

(a) Parity Error Signal Output

The parity error signal is enabled by setting the U2C1 register's U2ERE bit to "1".

- When receiving

The parity error signal is output when a parity error is detected while receiving data. This is achieved by pulling the TxD2 output low with the timing shown in Figure 2.11.32. If the U2RB register is read while outputting a parity error signal, the PER bit is cleared to "0" and at the same time the TxD2 output is returned high.

- When transmitting

A transmission-finished interrupt request is generated at the falling edge of the transfer clock pulse that immediately follows the stop bit. Therefore, whether a parity signal has been returned can be determined by reading the port that shares the RxD2 pin in a transmission-finished interrupt service routine.

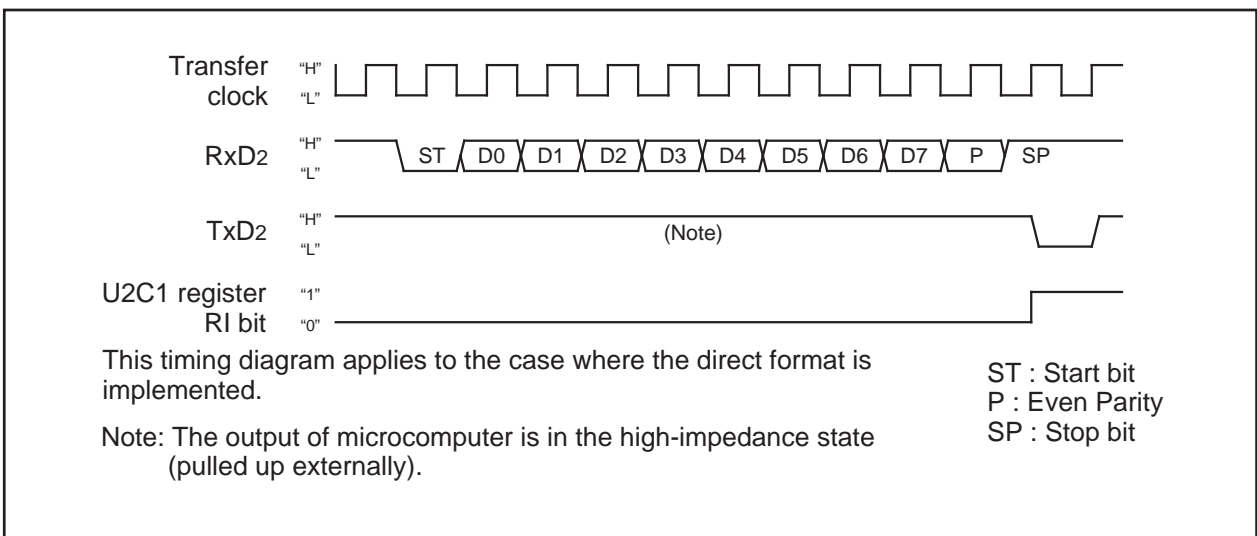


Figure 2.11.32. Parity Error Signal Output Timing

(b) Format

- Direct Format

Set the U2MR register's PRY bit to "1", U2C0 register's UFORM bit to "0" and U2C1 register's U2LCH bit to "0".

- Inverse Format

Set the PRY bit to "0", UFORM bit to "1" and U2LCH bit to "1".

Figure 2.11.33 shows the SIM interface format.

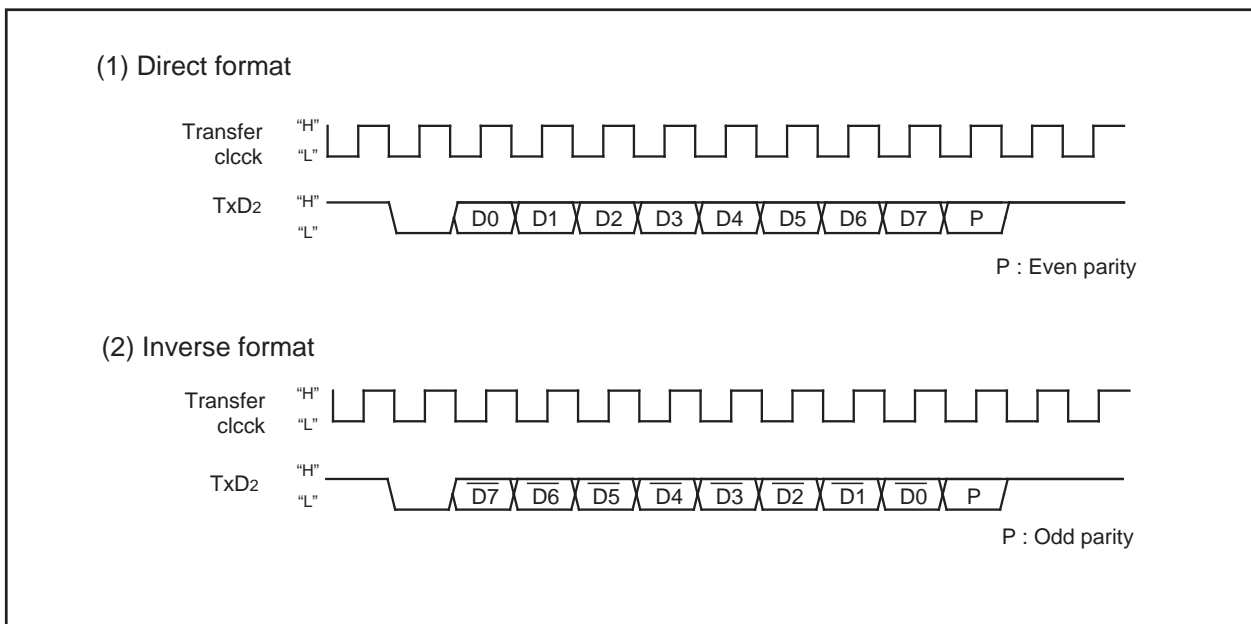


Figure 2.11.33. SIM Interface Format

2.11.8 SI/O3 and SI/O4

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

Figure 2.11.34 shows the block diagram of SI/O3 and SI/O4, and Figure 2.11.35 shows the SI/O3 and SI/O4-related registers.

Table 2.11.19 shows the specifications of SI/O3 and SI/O4.

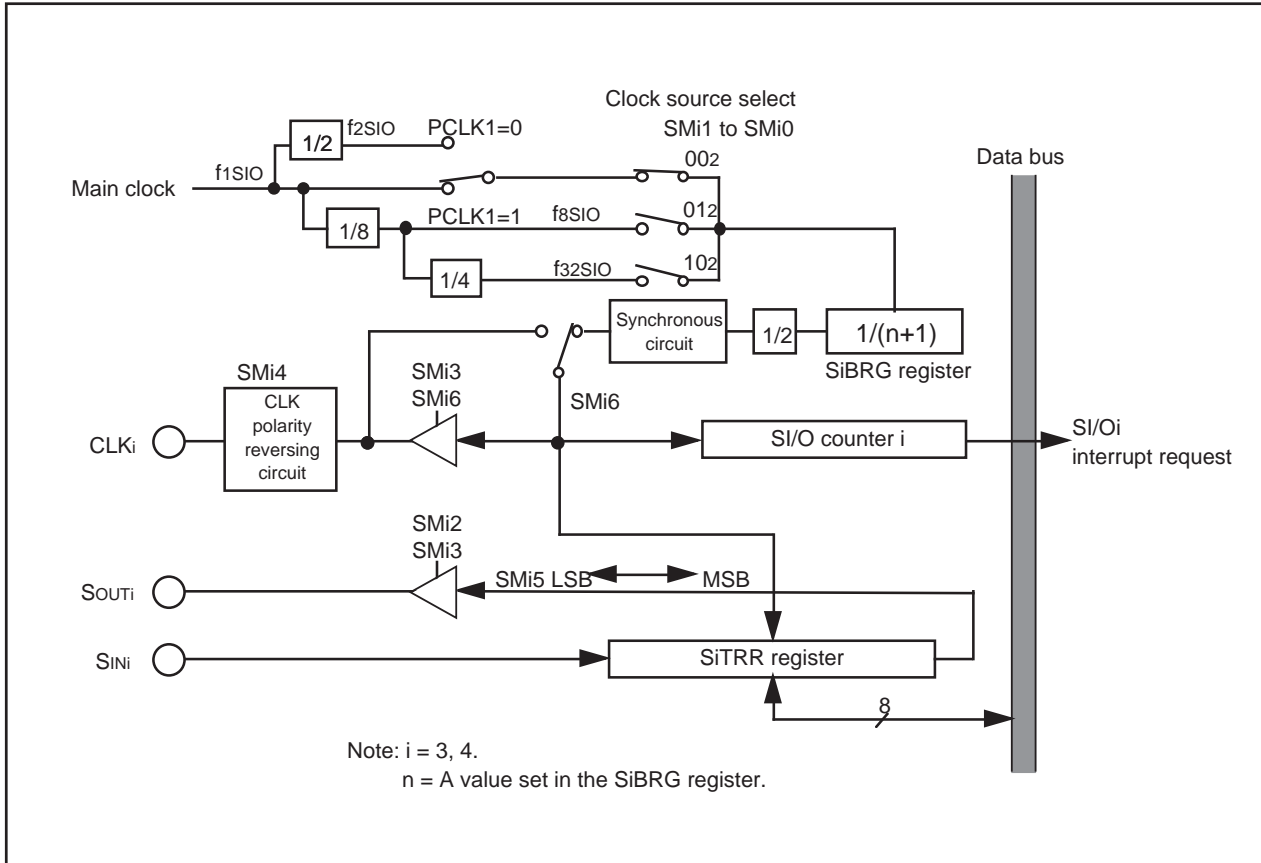


Figure 2.11.34. SI/O3 and SI/O4 Block Diagram

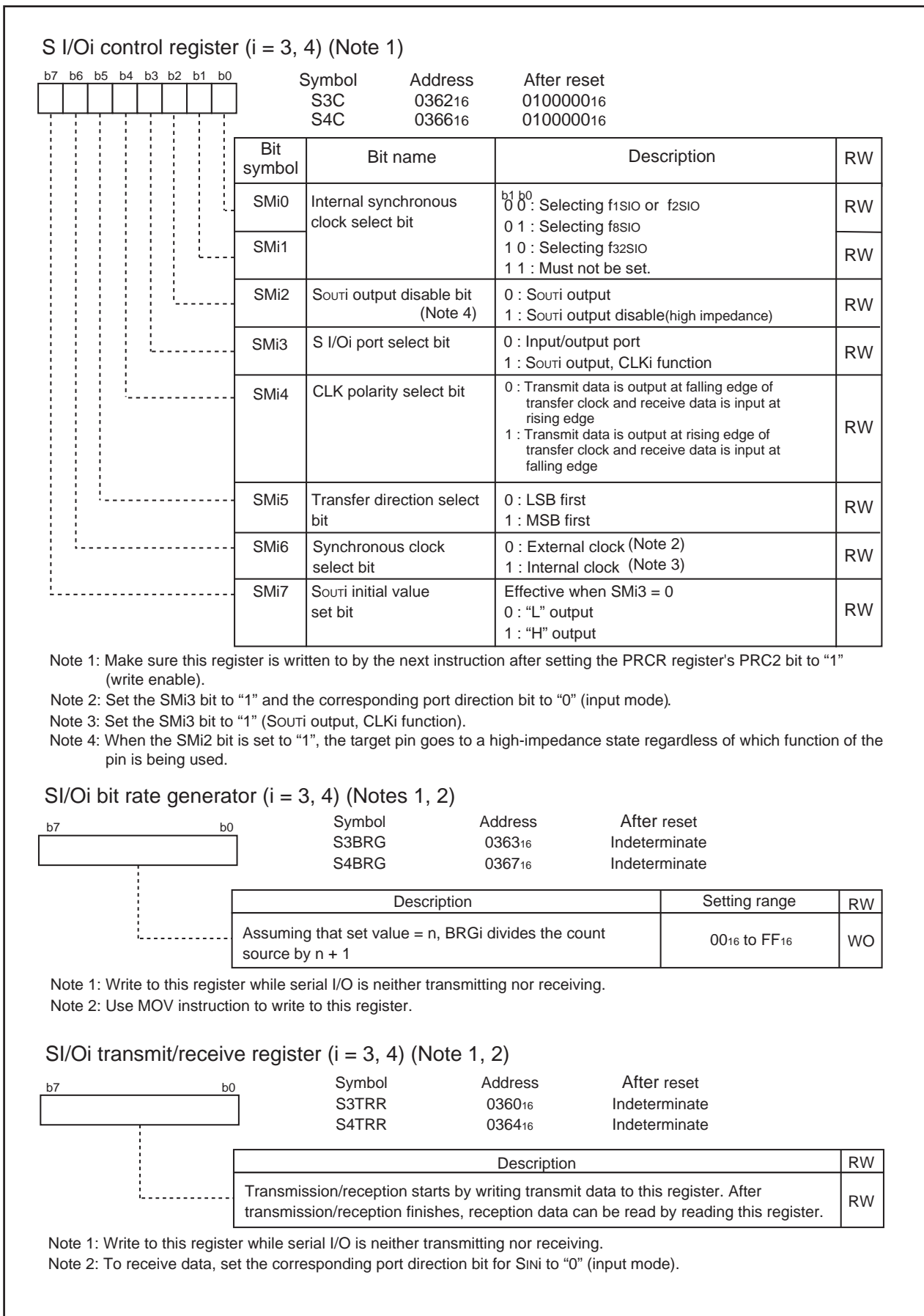


Figure 2.11.35. S3C and S4C Registers, S3BRG and S4BRG Registers, and S3TRR and S4TRR Registers

Table 2.11.19. SI/O3 and SI/O4 Specifications

Item	Specification
Transfer data format	<ul style="list-style-type: none"> Transfer data length: 8 bits
Transfer clock	<ul style="list-style-type: none"> SiC (i=3, 4) register's SMi6 bit = "1" (internal clock) : $f_j / 2^{(n+1)}$ $f_j = f_{1SIO}, f_{8SIO}, f_{32SIO}$. n=Setting value of SiBRG register 0016 to FF16. SMi6 bit = "0" (external clock) : Input from CLKi pin (Note 1)
Transmission/reception start condition	<ul style="list-style-type: none"> Before transmission/reception can start, the following requirements must be met Write transmit data to the SiTRR register (Notes 2, 3)
Interrupt request generation timing	<ul style="list-style-type: none"> When SiC register's SMi4 bit = 0 The rising edge of the last transfer clock pulse (Note 4) When SMi4 = 1 The falling edge of the last transfer clock pulse (Note 4)
CLKi pin function	I/O port, transfer clock input, transfer clock output
SOUTi pin function	I/O port, transmit data output, high-impedance
SINi pin function	I/O port, receive data input
Select function	<ul style="list-style-type: none"> LSB first or MSB first selection Whether to start sending/receiving data beginning with bit 0 or beginning with bit 7 can be selected Function for setting an SOUTi initial value set function When the SiC register's SMi6 bit = 0 (external clock), the SOUTi pin output level while not transmitting can be selected. CLK polarity selection Whether transmit data is output/input timing at the rising edge or falling edge of transfer clock can be selected.

Note 1: To set the SiC register's SMi6 bit to "0" (external clock), follow the procedure described below.

- If the SiC register's SMi4 bit = 0, write transmit data to the SiTRR register while input on the CLKi pin is high. The same applies when rewriting the SiC register's SMi7 bit.
- If the SMi4 bit = 1, write transmit data to the SiTRR register while input on the CLKi pin is low. The same applies when rewriting the SMi7 bit.
- Because shift operation continues as long as the transfer clock is supplied to the SI/Oi circuit, stop the transfer clock after supplying eight pulses. If the SMi6 bit = 1 (internal clock), the transfer clock automatically stops.

Note 2: Unlike UART0 to UART2, SI/Oi (i = 3 to 4) is not separated between the transfer register and buffer. Therefore, do not write the next transmit data to the SiTRR register during transmission.

Note 3: When the SiC register's SMi6 bit = 1 (internal clock), SOUTi retains the last data for a 1/2 transfer clock period after completion of transfer and, thereafter, goes to a high-impedance state. However, if transmit data is written to the SiTRR register during this period, SOUTi immediately goes to a high-impedance state, with the data hold time thereby reduced.

Note 4: When the SiC register's SMi6 bit = 1 (internal clock), the transfer clock stops in the high state if the SMi4 bit = 0, or stops in the low state if the SMi4 bit = 1.

(a) SI/Oi Operation Timing

Figure 2.11.36 shows the SI/Oi operation timing

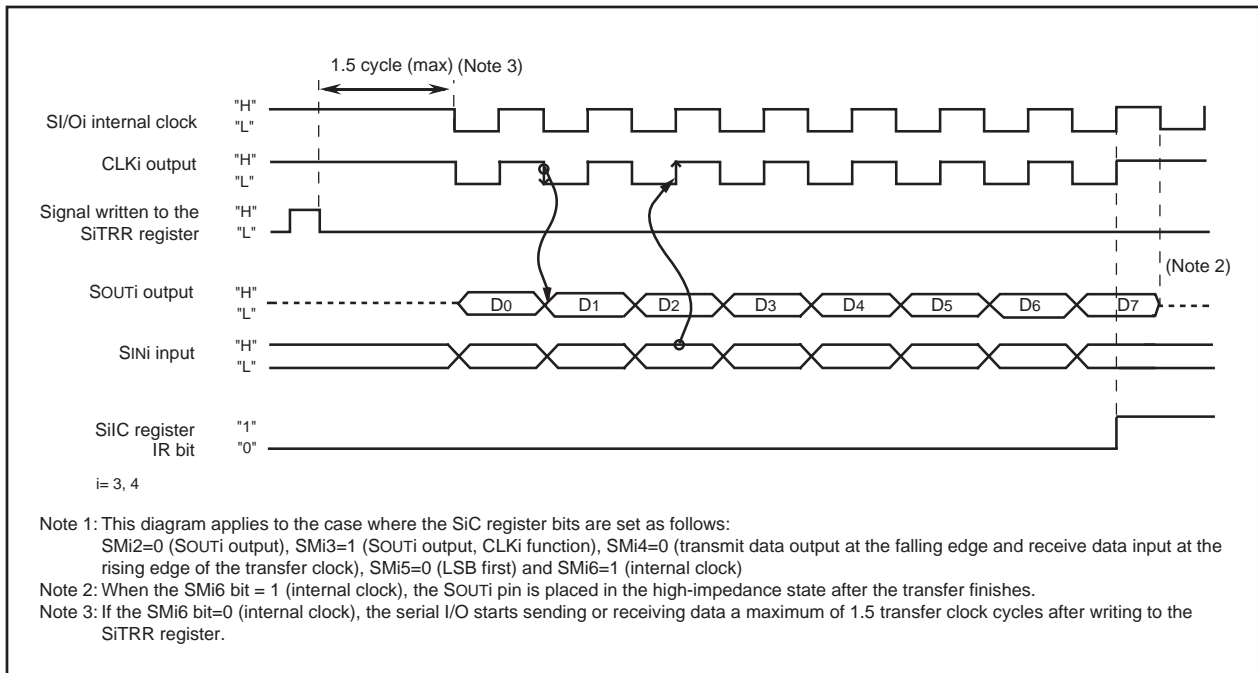


Figure 2.11.36. SI/Oi Operation Timing

(b) CLK Polarity Selection

The SiC register's SMi4 bit allows selection of the polarity of the transfer clock. Figure 2.11.37 shows the polarity of the transfer clock.

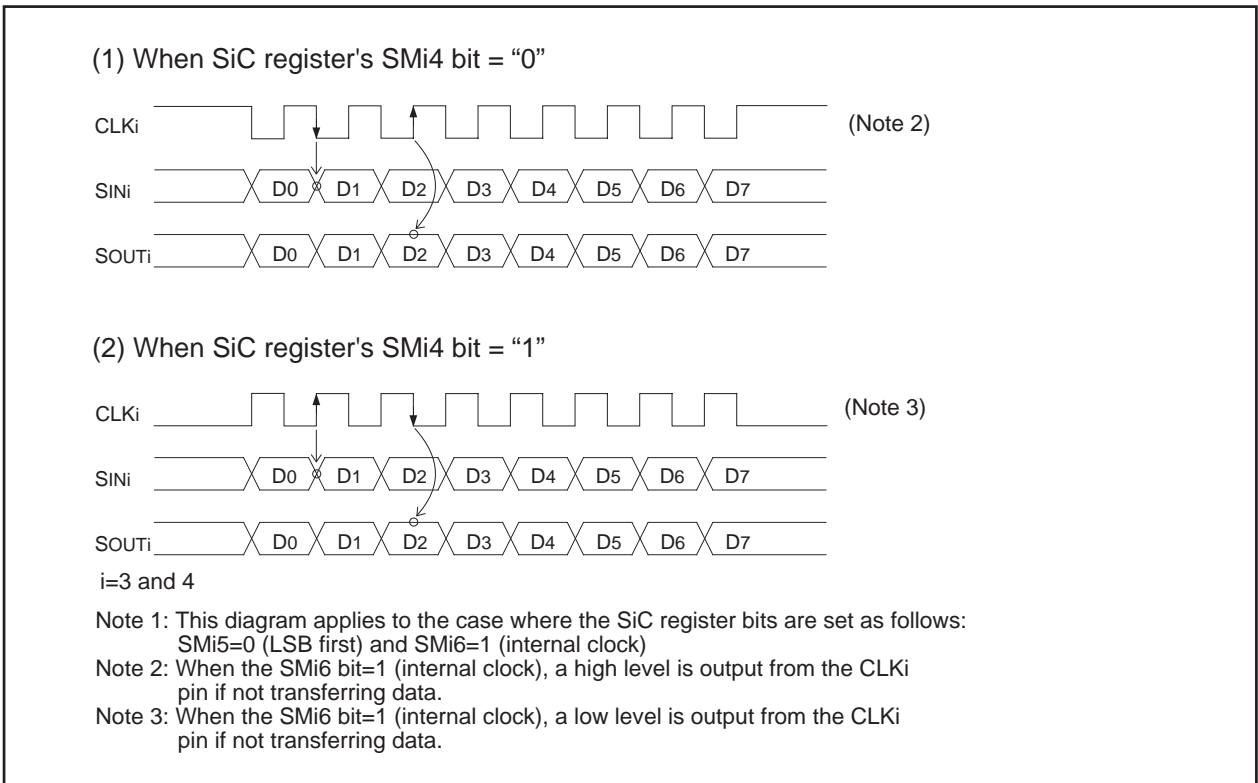


Figure 2.11.37. Polarity of Transfer Clock

(c) Functions for Setting an Sout_i Initial Value

If the SiC register's SMI₆ bit = 0 (external clock), the SOUT_i pin output can be fixed high or low when not transferring. Figure 2.11.38 shows the timing chart for setting an SOUT_i initial value and how to set it.

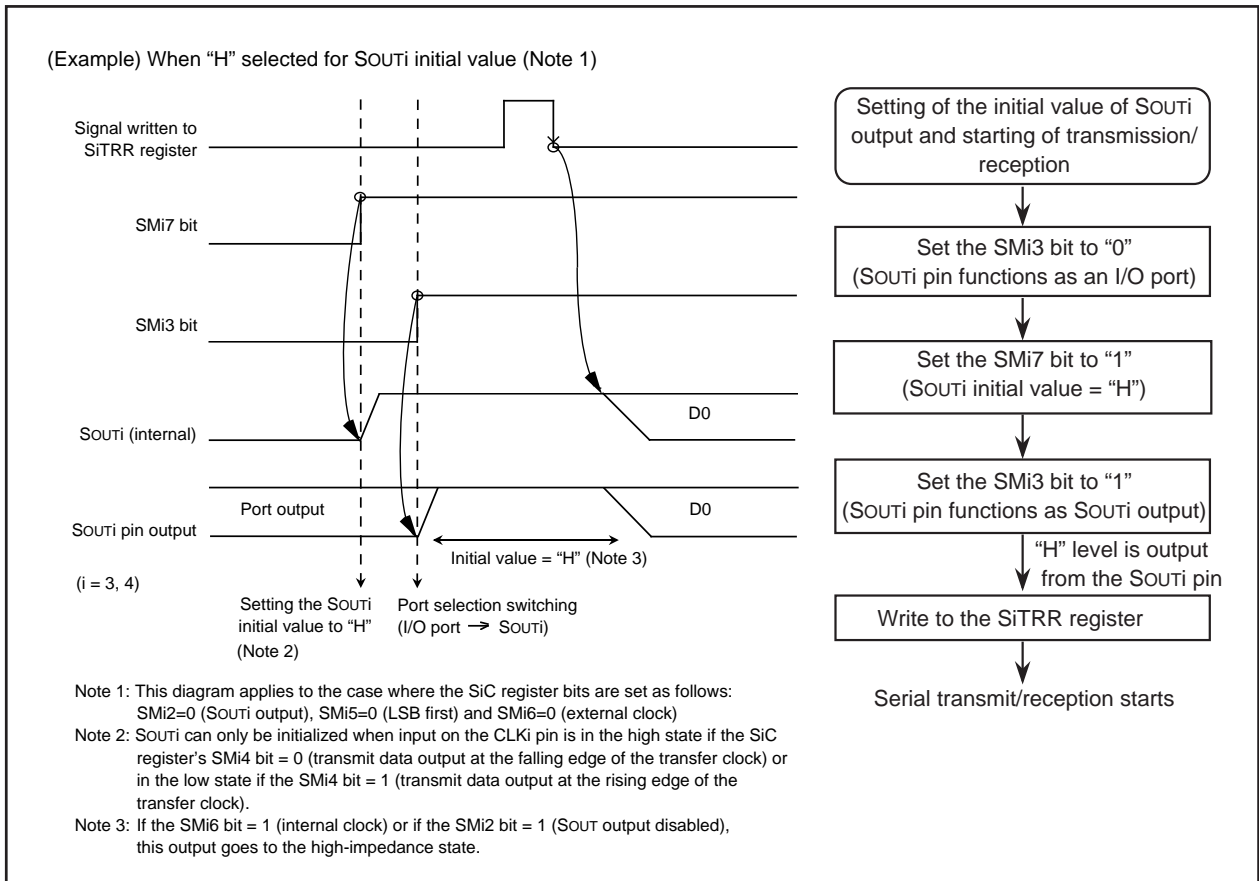


Figure 2.11.38. Sout_i's Initial Value Setting

2.12 A-D Converter

The microcomputer contains one A-D converter circuit based on 8-bit successive approximation method configured with a capacitive-coupling amplifier. The analog inputs share the pins with P100 to P107, P95 and P96. Similarly, $\overline{\text{ADTRG}}$ input shares the pin with P97. Therefore, when using these inputs, make sure the corresponding port direction bits are set to "0" (= input mode).

When not using the A-D converter, set the VCUT bit to "0" (= Vref unconnected), so that no current will flow from the VREF pin into the resistor ladder, helping to reduce the power consumption of the chip.

The A-D conversion result is stored in the ADi register bits for ANi pins (i = 0 to 7).

Table 2.12.1 shows the performance of the A-D converter. Figure 2.12.1 shows the block diagram of the A-D converter, and Figures 2.12.2 and 2.12.3 show the A-D converter-related registers.

Table 2.12.1. Performance of A-D Converter

Item	Performance
Method of A-D conversion	Successive approximation (capacitive coupling amplifier)
Analog input voltage (Note 1)	0V to AVCC (VCC)
Operating clock ϕ_{AD} (Note 2)	fAD/divide-by-2 of fAD/divide-by-3 of fAD/divide-by-4 of fAD/divide-by-6 of fAD/divide-by-12 of fAD
Resolution	8-bit
Integral nonlinearity error	When AVCC = VREF = 5V <ul style="list-style-type: none"> • With 8-bit resolution: $\pm 3\text{LSB}$ <ul style="list-style-type: none"> - ANEX0 and ANEX1 input (including mode in which external operation amp is connected) : $\pm 4\text{LSB}$
Operating modes	One-shot mode, repeat mode, single sweep mode, repeat sweep mode 0, and repeat sweep mode 1
Analog input pins	8 pins (AN0 to AN7) + 2 pins (ANEX0 and ANEX1)
A-D conversion start condition	<ul style="list-style-type: none"> • Software trigger <ul style="list-style-type: none"> The ADCON0 register's ADST bit is set to "1" (A-D conversion starts) • External trigger (retriggerable) <ul style="list-style-type: none"> Input on the $\overline{\text{ADTRG}}$ pin changes state from high to low after the ADST bit is set to "1" (A-D conversion starts)
Conversion speed per pin	<ul style="list-style-type: none"> • Without sample and hold function <ul style="list-style-type: none"> 8-bit resolution: 49 ϕ_{AD} cycles • With sample and hold function <ul style="list-style-type: none"> 8-bit resolution: 28 ϕ_{AD} cycles

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

Note 2: The fAD frequency must be 10 MHz or less.

Without sample-and-hold function, limit the fAD frequency to 250kHz or more.

With the sample and hold function, limit the fAD frequency to 1MHz or more.

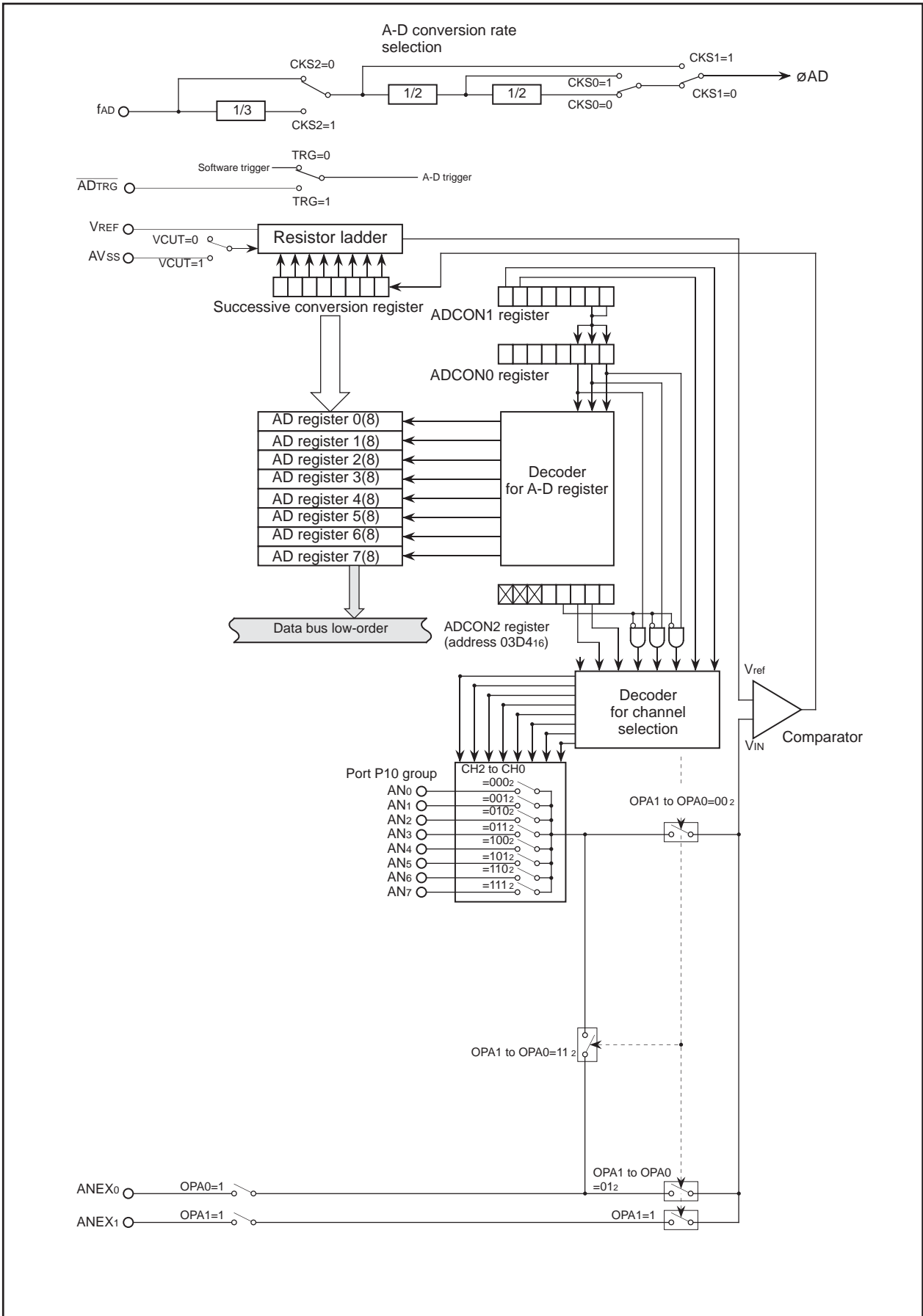


Figure 2.12.1. A-D Converter Block Diagram

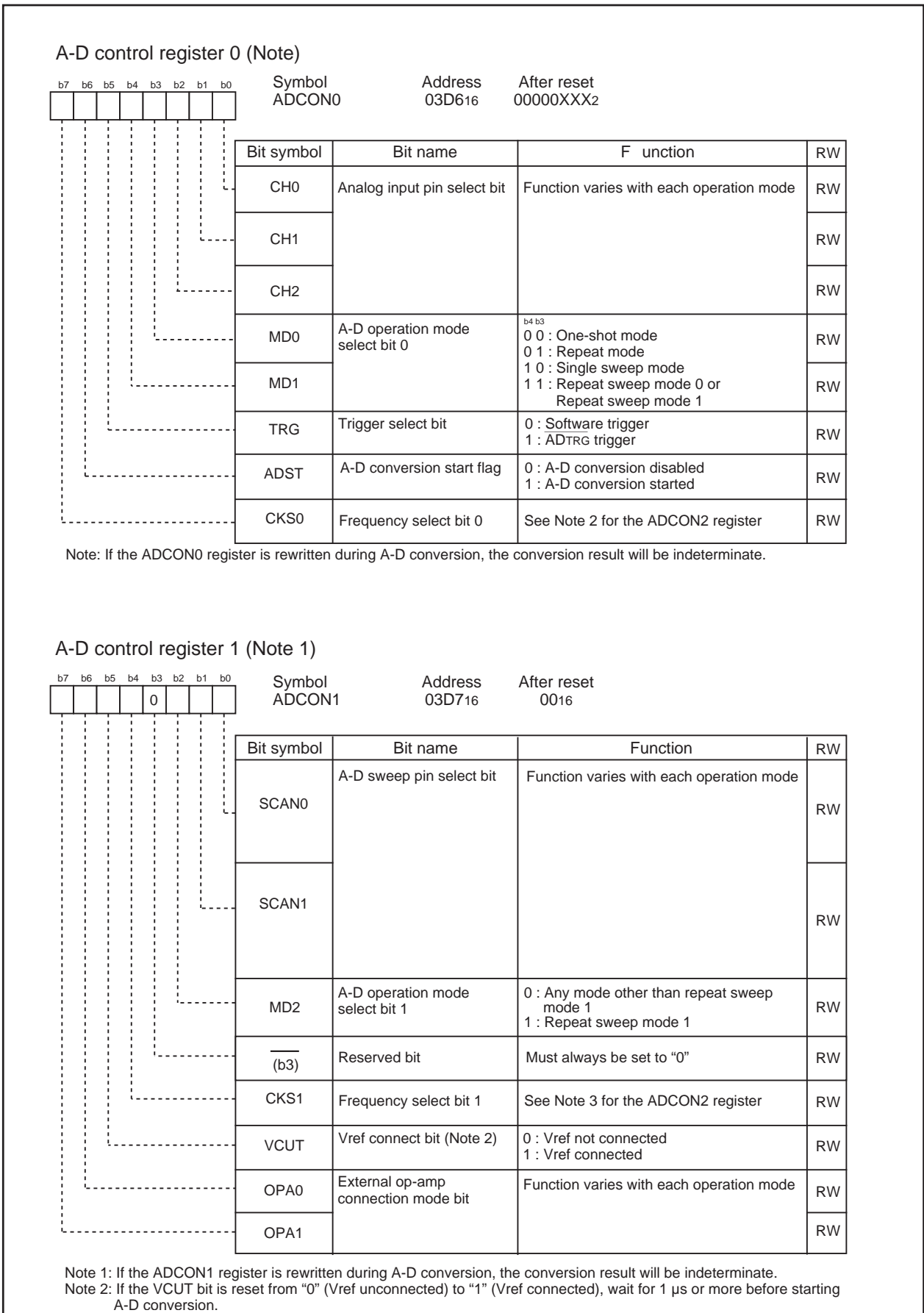


Figure 2.12.2. ADCON0 to ADCON1 Registers

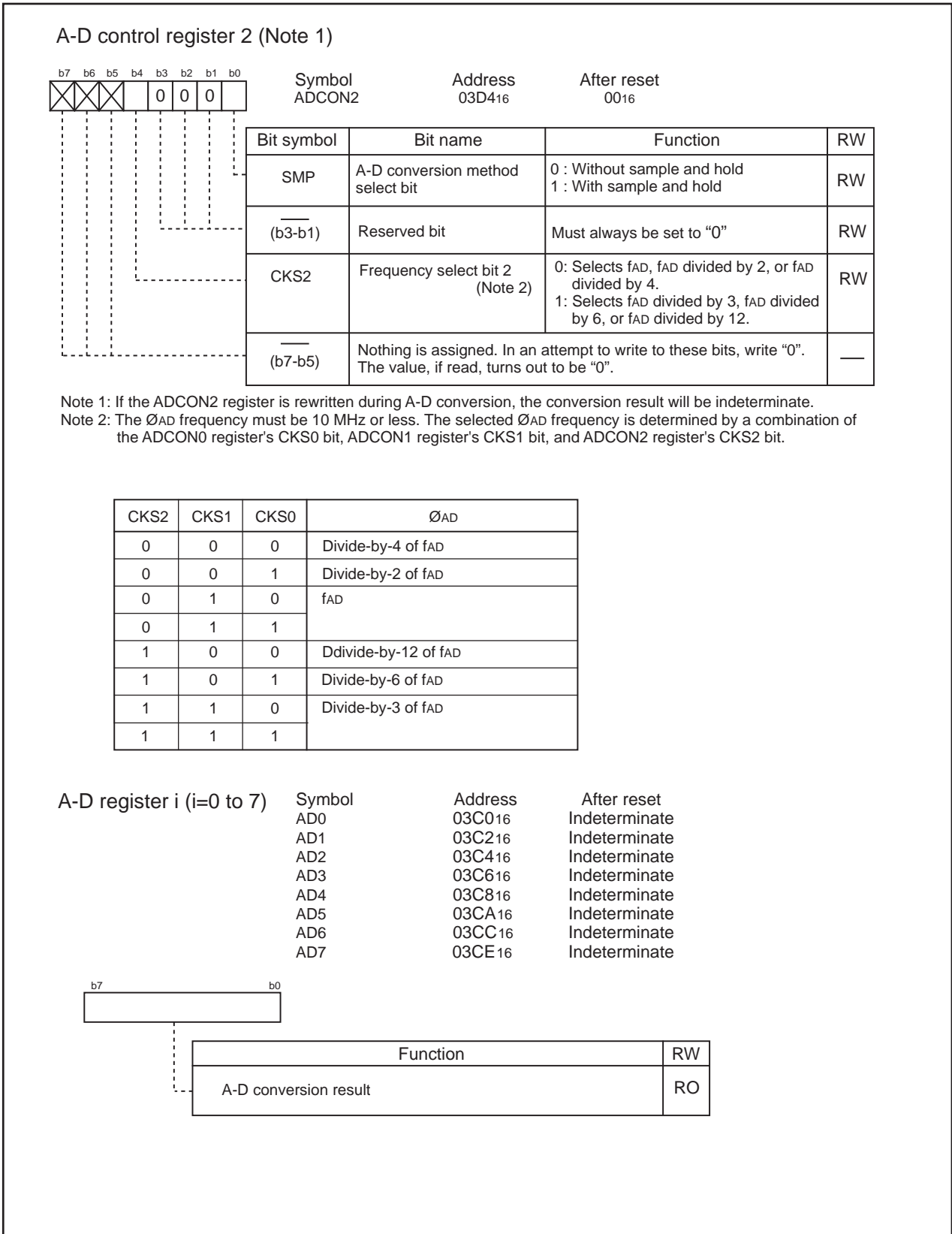


Figure 2.12.3. ADCON2 Register, and AD0 to AD7 Registers

(1) One-shot Mode

In this mode, the input voltage on one selected pin is A-D converted once. Table 2.12.2 shows the specifications of one-shot mode. Figure 2.12.4 shows the ADCON0 to ADCON1 registers in one-shot mode.

Table 2.12.2. One-shot Mode Specifications

Item	Specification
Function	The input voltage on one pin selected by the ADCON0 register's CH2 to CH0 bits and the ADCON1 register's OPA1 to OPA0 bits is A-D converted once.
A-D conversion start condition	<ul style="list-style-type: none"> • When the ADCON0 register's TRG bit is "0" (software trigger) The ADCON0 register's ADST bit is set to "1" (A-D conversion starts) • When the TRG bit is "1" ($\overline{\text{ADTRG}}$ trigger) Input on the $\overline{\text{ADTRG}}$ pin changes state from high to low after the ADST bit is set to "1" (A-D conversion starts)
A-D conversion stop condition	<ul style="list-style-type: none"> • Completion of A-D conversion (If a software trigger is selected, the ADST bit is cleared to "0" (A-D conversion halted).) • Set the ADST bit to "0"
Interrupt request generation timing	Completion of A-D conversion
Analog input pin	Select one pin from AN0 to AN7, ANEX0 to ANEX1
Reading of result of A-D converter	Read one of the AD0 to AD7 registers that corresponds to the selected pin

A-D control register 0 (Note 1)

b7	b6	b5	b4	b3	b2	b1	b0	Symbol ADCON0	Address 03D616	After reset 00000XXX2
			0	0						

Bit symbol	Bit name	Function	RW
CH0	Analog input pin select bit	b2 b1 b0 0 0 0 : AN0 is selected 0 0 1 : AN1 is selected 0 1 0 : AN2 is selected 0 1 1 : AN3 is selected 1 0 0 : AN4 is selected 1 0 1 : AN5 is selected 1 1 0 : AN6 is selected 1 1 1 : AN7 is selected (Note 2)	RW
CH1			RW
CH2			RW
MD0	A-D operation mode select bit 0	b4 b3 0 0 : One-shot mode (Note 2)	RW
MD1			RW
TRG	Trigger select bit	0 : Software trigger 1 : ADTRG trigger	RW
ADST	A-D conversion start flag	0 : A-D conversion disabled 1 : A-D conversion started	RW
CKS0	Frequency select bit 0	See Note 2 for the ADCON2 register	RW

Note 1: If the ADCON0 register is rewritten during A-D conversion, the conversion result will be indeterminate.
 Note 2: After rewriting the MD1 to MD0 bits, set the CH2 to CH0 bits over again using another instruction.

A-D control register 1 (Note)

b7	b6	b5	b4	b3	b2	b1	b0	Symbol ADCON1	Address 03D716	After reset 0016
		1		0	0					

Bit symbol	Bit name	Function	RW
SCAN0	A-D sweep pin select bit	Invalid in one-shot mode	RW
SCAN1			RW
MD2	A-D operation mode select bit 1	Set to "0" when one-shot mode is selected	RW
(b3)	Reserved bit	Must always be set to "0"	RW
CKS1	Frequency select bit1	See Note 2 for the ADCON2 register	RW
VCUT	Vref connect bit (Note 2)	1 : Vref connected	RW
OPA0	External op-amp connection mode bit	b7 b6 0 0 : ANEX0 and ANEX1 are not used 0 1 : ANEX0 input is A-D converted 1 0 : ANEX1 input is A-D converted 1 1 : External op-amp connection mode	RW
OPA1			RW

Note 1: If the ADCON1 register is rewritten during A-D conversion, the conversion result will be indeterminate.
 Note 2: If the VCUT bit is reset from "0" (Vref unconnected) to "1" (Vref connected), wait for 1 μs or more before starting A-D conversion.

Figure 2.12.4. ADCON0 Register and ADCON1 Register (One-shot Mode)

(2) Repeat mode

In this mode, the input voltage on one selected pin is A-D converted repeatedly. Table 2.12.3 shows the specifications of repeat mode. Figure 2.12.5 shows the ADCON0 to ADCON1 registers in repeat mode.

Table 2.12.3. Repeat Mode Specifications

Item	Specification
Function	The input voltage on one pin selected by the ADCON0 register's CH2 to CH0 bits and the ADCON1 register's OPA1 to OPA0 bits is A-D converted repeatedly.
A-D conversion start condition	<ul style="list-style-type: none"> • When the ADCON0 register's TRG bit is "0" (software trigger) The ADCON0 register's ADST bit is set to "1" (A-D conversion starts) • When the TRG bit is "1" ($\overline{\text{ADTRG}}$ trigger) Input on the $\overline{\text{ADTRG}}$ pin changes state from high to low after the ADST bit is set to "1" (A-D conversion starts)
A-D conversion stop condition	Set the ADST bit to "0" (A-D conversion halted)
Interrupt request generation timing	None generated
Analog input pin	Select one pin from AN0 to AN7, ANEX0 to ANEX1
Reading of result of A-D converter	Read one of the AD0 to AD7 registers that corresponds to the selected pin

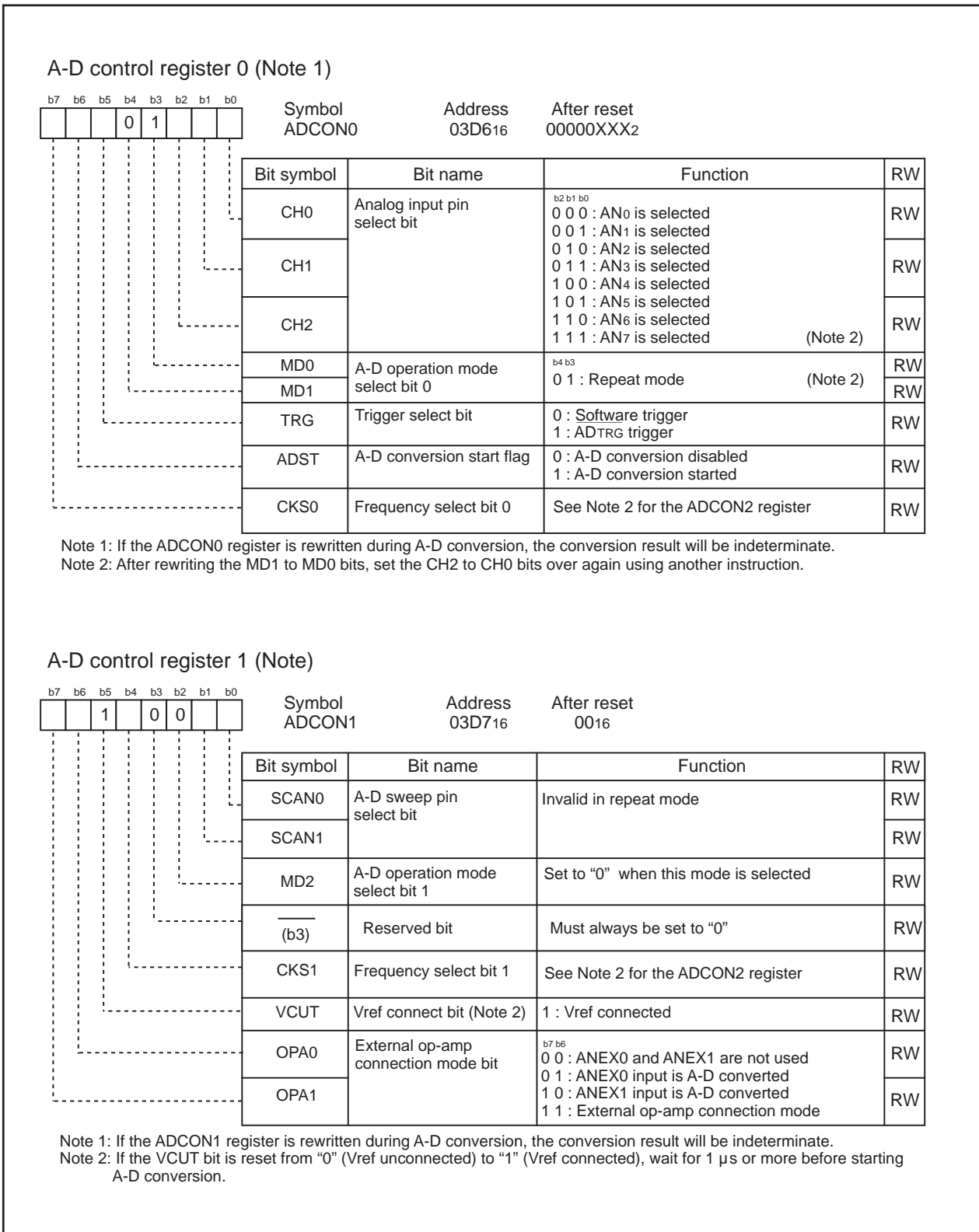


Figure 2.12.5. ADCON0 Register and ADCON1 Register (Repeat Mode)

(3) Single Sweep Mode

In this mode, the input voltages on selected pins are A-D converted, one pin at a time. Table 2.12.4 shows the specifications of single sweep mode. Figure 2.12.6 shows the ADCON0 to ADCON1 registers in single sweep mode.

Table 2.12.4. Single Sweep Mode Specifications

Item	Specification
Function	The input voltages on pins selected by the ADCON1 register's SCAN1 to SCAN0 bits are A-D converted, one pin at a time.
A-D conversion start condition	<ul style="list-style-type: none"> • When the ADCON0 register's TRG bit is "0" (software trigger) The ADCON0 register's ADST bit is set to "1" (A-D conversion starts) • When the TRG bit is "1" ($\overline{\text{ADTRG}}$ trigger) Input on the $\overline{\text{ADTRG}}$ pin changes state from high to low after the ADST bit is set to "1" (A-D conversion starts)
A-D conversion stop condition	<ul style="list-style-type: none"> • Completion of A-D conversion (If a software trigger is selected, the ADST bit is cleared to "0" (A-D conversion halted).) • Set the ADST bit to "0"
Interrupt request generation timing	Completion of A-D conversion
Analog input pin	Select from AN0 to AN1 (2 pins), AN0 to AN3 (4 pins), AN0 to AN5 (6 pins), AN0 to AN7 (8 pins)
Reading of result of A-D converter	Read one of the AD0 to AD7 registers that corresponds to the selected pin

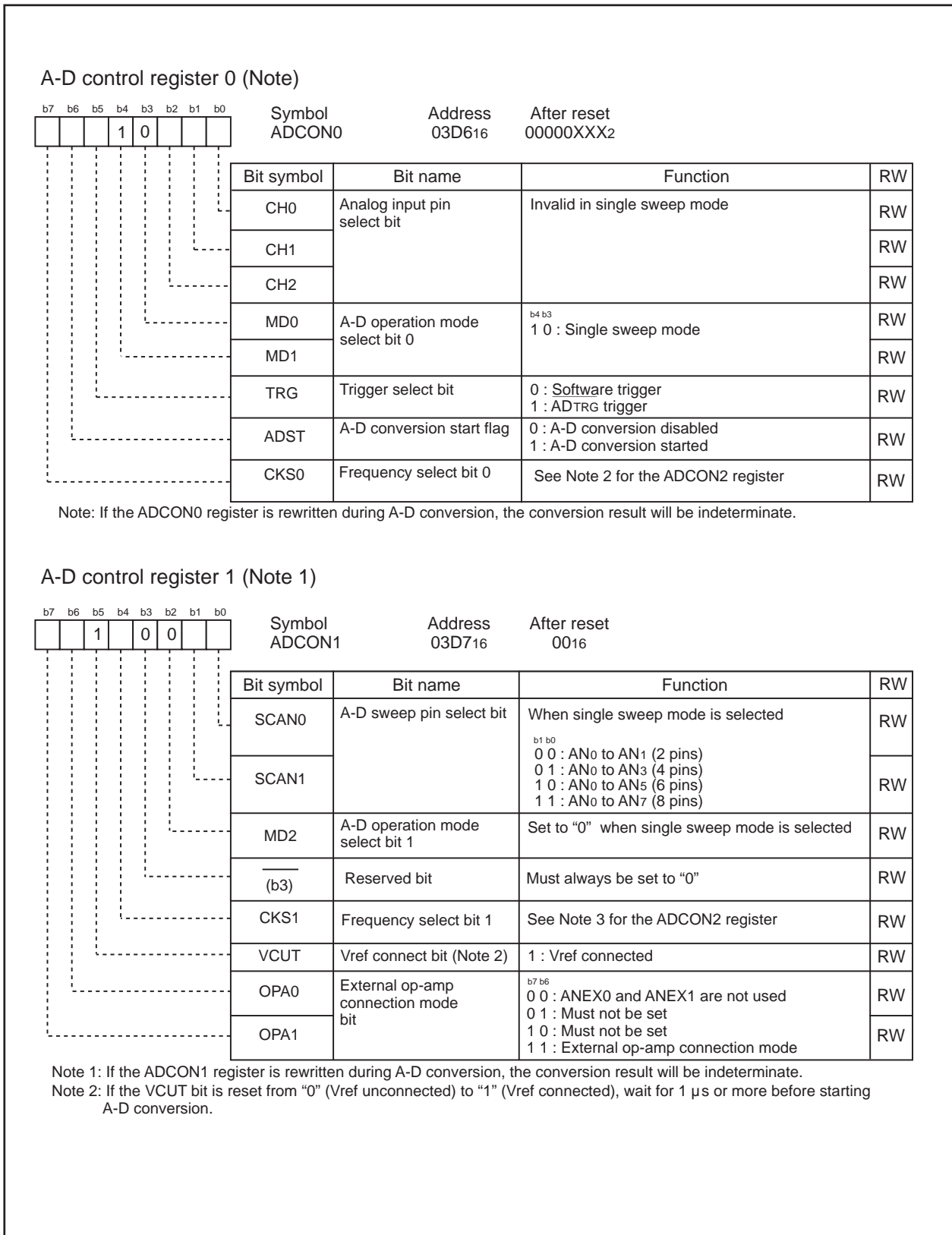


Figure 2.12.6. ADCON0 Register and ADCON1 Register (Single Sweep Mode)

(4) Repeat Sweep Mode 0

In this mode, the input voltages on selected pins are A-D converted repeatedly. Table 2.12.5 shows the specifications of repeat sweep mode 0. Figure 2.12.7 shows the ADCON0 to ADCON1 registers in repeat sweep mode 0.

Table 2.12.5. Repeat Sweep Mode 0 Specifications

Item	Specification
Function	The input voltages on pins selected by the ADCON1 register's SCAN1 to SCAN0 bits are A-D converted repeatedly.
A-D conversion start condition	<ul style="list-style-type: none"> • When the ADCON0 register's TRG bit is "0" (software trigger) The ADCON0 register's ADST bit is set to "1" (A-D conversion starts) • When the TRG bit is "1" ($\overline{\text{ADTRG}}$ trigger) Input on the $\overline{\text{ADTRG}}$ pin changes state from high to low after the ADST bit is set to "1" (A-D conversion starts)
A-D conversion stop condition	Set the ADST bit to "0" (A-D conversion halted)
Interrupt request generation timing	None generated
Analog input pin	Select from AN0 to AN1 (2 pins), AN0 to AN3 (4 pins), AN0 to AN5 (6 pins), AN0 to AN7 (8 pins)
Reading of result of A-D converter	Read one of the AD0 to AD7 registers that corresponds to the selected pin

A-D control register 0 (Note)

b7	b6	b5	b4	b3	b2	b1	b0	Symbol ADCON0	Address 03D6 ₁₆	After reset 00000XXX ₂	
			1	1							
								Bit symbol	Bit name	Function	RW
								CH0	Analog input pin select bit	Invalid in repeat sweep mode 0	RW
							CH1	RW			
							CH2	RW			
								MD0	A-D operation mode select bit 0	b4 b3 1 1 : Repeat sweep mode 0 or Repeat sweep mode 1	RW
							MD1	RW			
								TRG	Trigger select bit	0 : Software trigger 1 : ADTRG trigger	RW
								ADST	A-D conversion start flag	0 : A-D conversion disabled 1 : A-D conversion started	RW
								CKS0	Frequency select bit 0	See Note 2 for the ADCON2 register	RW

Note: If the ADCON0 register is rewritten during A-D conversion, the conversion result will be indeterminate.

A-D control register 1 (Note 1)

b7	b6	b5	b4	b3	b2	b1	b0	Symbol ADCON1	Address 03D7 ₁₆	After reset 00 ₁₆	
		1		0	0						
								Bit symbol	Bit name	Function	RW
								SCAN0	A-D sweep pin select bit	When repeat sweep mode 0 is selected	RW
							SCAN1	b1 b0 0 0 : AN ₀ , AN ₁ (2 pins) 0 1 : AN ₀ to AN ₃ (4 pins) 1 0 : AN ₀ to AN ₅ (6 pins) 1 1 : AN ₀ to AN ₇ (8 pins)			RW
								MD2	A-D operation mode select bit 1	Set to "0" when repeat sweep mode 0 is selected	RW
								(b3)	Reserved bit	Must always be set to "0"	RW
								CKS1	Frequency select bit 1	See Note 2 for the ADCON2 register	RW
								VCUT	Vref connect bit (Note 2)	1 : Vref connected	RW
								OPA0	External op-amp connection mode bit	b7 b6 0 0 : ANEX0 and ANEX1 are not used 0 1 : Must not be set 1 0 : Must not be set 1 1 : External op-amp connection mode	RW
							OPA1	RW			

Note 1: If the ADCON1 register is rewritten during A-D conversion, the conversion result will be indeterminate.

Note 2: If the VCUT bit is reset from "0" (Vref unconnected) to "1" (Vref connected), wait for 1 μs or more before starting A-D conversion.

Figure 2.12.7. ADCON0 Register and ADCON1 Registers (Repeat Sweep Mode 0)

(5) Repeat Sweep Mode 1

In this mode, the input voltages on all pins are A-D converted repeatedly, with priority given to the selected pins. Table 2.12.6 shows the specifications of repeat sweep mode 1. Figure 2.12.8 shows the ADCON0 to ADCON1 registers in repeat sweep mode 1.

Table 2.12.6. Repeat Sweep Mode 1 Specifications

Item	Specification
Function	The input voltages on all selected pins are A-D converted repeatedly, with priority given to pins selected by the ADCON1 register's SCAN1 to SCAN0 bits. Example : If AN0 selected, input voltages are A-D converted in order of AN0 → AN1 → AN0 → AN2 → AN0 → AN3, and so on.
A-D conversion start condition	<ul style="list-style-type: none"> • When the ADCON0 register's TRG bit is "0" (software trigger) The ADCON0 register's ADST bit is set to "1" (A-D conversion starts) • When the TRG bit is "1" ($\overline{\text{ADTRG}}$ trigger) Input on the $\overline{\text{ADTRG}}$ pin changes state from high to low after the ADST bit is set to "1" (A-D conversion starts)
A-D conversion stop condition	Set the ADST bit to "0" (A-D conversion halted)
Interrupt request generation timing	None generated
Analog input pins to be given priority when A-D converted	Select from AN0 (1 pins), AN0 to AN1 (2 pins), AN0 to AN2 (3 pins), AN0 to AN3 (4 pins)
Reading of result of A-D converter	Read one of the AD0 to AD7 registers that corresponds to the selected pin

A-D control register 0 (Note)

b7	b6	b5	b4	b3	b2	b1	b0	Symbol ADCON0	Address 03D6 ₁₆	After reset 0000XXX ₂	
			1	1							
								Bit symbol	Bit name	Function	RW
								CH0	Analog input pin select bit	Invalid in repeat sweep mode 1	RW
								CH1			RW
								CH2			RW
								MD0	A-D operation mode select bit 0	b ₄ b ₃ 1 1 : Repeat sweep mode 0 or Repeat sweep mode 1	RW
								MD1			RW
								TRG	Trigger select bit	0 : Software trigger 1 : ADTRG trigger	RW
								ADST	A-D conversion start flag	0 : A-D conversion disabled 1 : A-D conversion started	RW
								CKS0	Frequency select bit 0	See Note 2 for the ADCON2 register	RW

Note: If the ADCON0 register is rewritten during A-D conversion, the conversion result will be indeterminate.

A-D control register 1 (Note 1)

b7	b6	b5	b4	b3	b2	b1	b0	Symbol ADCON1	Address 03D7 ₁₆	After reset 00 ₁₆	
		1		0	1						
								Bit symbol	Bit name	Function	RW
								SCAN0	A-D sweep pin select bit	When repeat sweep mode 1 is selected b ₁ b ₀ 0 0 : AN ₀ (1 pin) 0 1 : AN ₀ , AN ₁ (2 pins) 1 0 : AN ₀ to AN ₂ (3 pins) 1 1 : AN ₀ to AN ₃ (4 pins)	RW
								SCAN1			RW
								MD2	A-D operation mode select bit 1	Set to "1" when repeat sweep mode 1 is selected	RW
								(b3)	Reserved bit	Must always be set to "0"	RW
								CKS1	Frequency select bit 1	See Note 2 for the ADCON2 register	RW
								VCUT	Vref connect bit (Note 2)	1 : Vref connected	RW
								OPA0	External op-amp connection mode bit	b ₇ b ₆ 0 0 : ANEX0 and ANEX1 are not used 0 1 : Must not be set 1 0 : Must not be set 1 1 : External op-amp connection mode	RW
								OPA1			RW

Note 1: If the ADCON1 register is rewritten during A-D conversion, the conversion result will be indeterminate.

Note 2: If the VCUT bit is reset from "0" (Vref unconnected) to "1" (Vref connected), wait for 1 μs or more before starting A-D conversion.

Figure 2.12.8. ADCON0 Register and ADCON1 Register (Repeat Sweep Mode 1)

(a) Sample and Hold

If the ADCON2 register's SMP bit is set to "1" (with sample-and-hold), the conversion speed per pin is increased to $28 \cdot \varnothing_{AD}$ cycles for 8-bit resolution. Sample-and-hold is effective in all operation modes. Select whether or not to use the sample-and-hold function before starting A-D conversion.

(b) Extended Analog Input Pins

In one-shot and repeat modes, the ANEX0 and ANEX1 pins can be used as analog input pins. Use the ADCON1 register's OPA1 to OPA0 bits to select whether or not use ANEX0 and ANEX1.

The A-D conversion results of ANEX0 and ANEX1 inputs are stored in the AD0 and AD1 registers, respectively.

(c) External Operation Amp Connection Mode

Multiple analog inputs can be amplified using a single external op-amp via the ANEX0 and ANEX1 pins. Set the ADCON1 register's OPA1 OPA0 bits to '112' (external op-amp connection mode). The inputs from AN_i (i = 0 to 7) are output from the ANEX0 pin. Amplify this output with an external op-amp before sending it back to the ANEX1 pin. The A-D conversion result is stored in the corresponding AD_i register. The A-D conversion speed depends on the response characteristics of the external op-amp. Note that the ANEX0 and ANEX1 pins cannot be directly connected to each other. Figure 2.12.9 is an example of how to connect the pins in external operation amp.

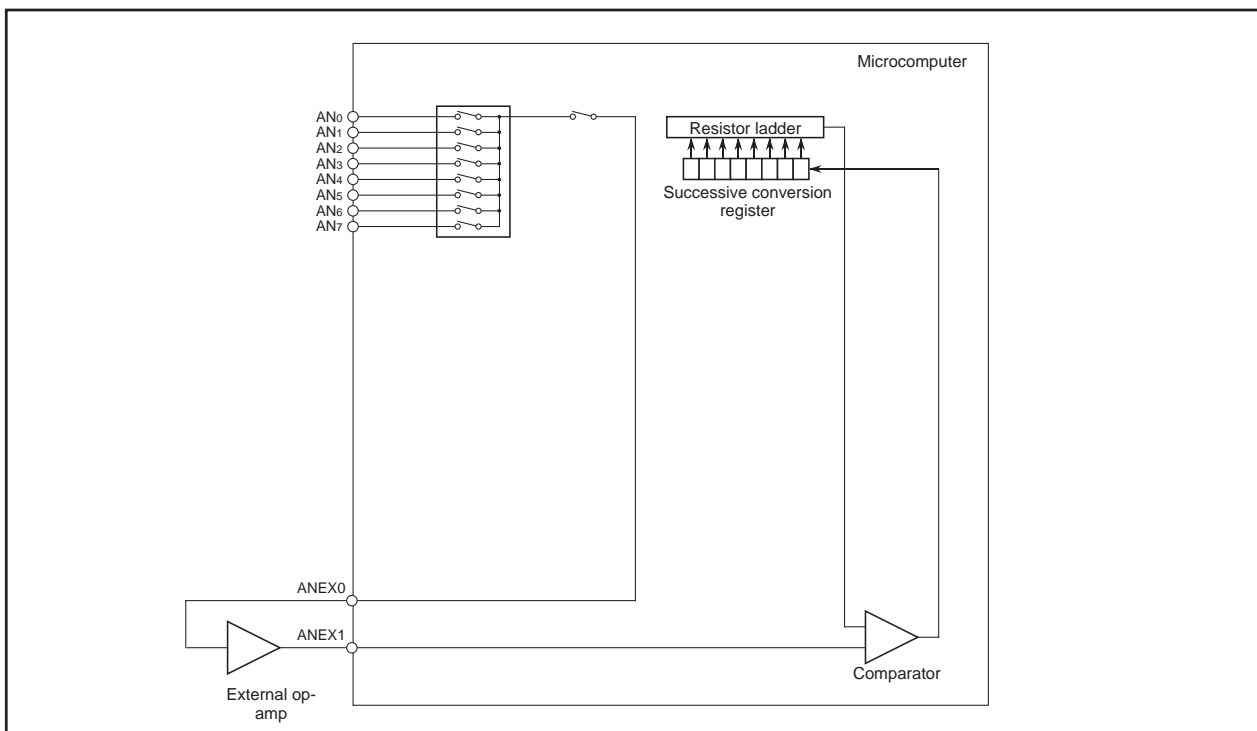


Figure 2.12.9. External Op-amp Connection

(d) Current Consumption Reducing Function

When not using the A-D converter, its resistor ladder and reference voltage input pin (VREF) can be separated using the ADCON1 register's VCUT bit. When separated, no current will flow from the VREF pin into the resistor ladder, helping to reduce the power consumption of the chip.

To use the A-D converter, set the VCUT bit to "1" (VREF connected) and then set the ADCON0 register's ADST bit to "1" (A-D conversion start). The VCUT and ADST bits cannot be set to "1" at the same time. Nor can the VCUT bit be set to "0" (VREF unconnected) during A-D conversion.

(e) Analog Input Pin and External Sensor Equivalent Circuit Example

Figure 2.12.10 shows analog input pin and external sensor equivalent circuit example.

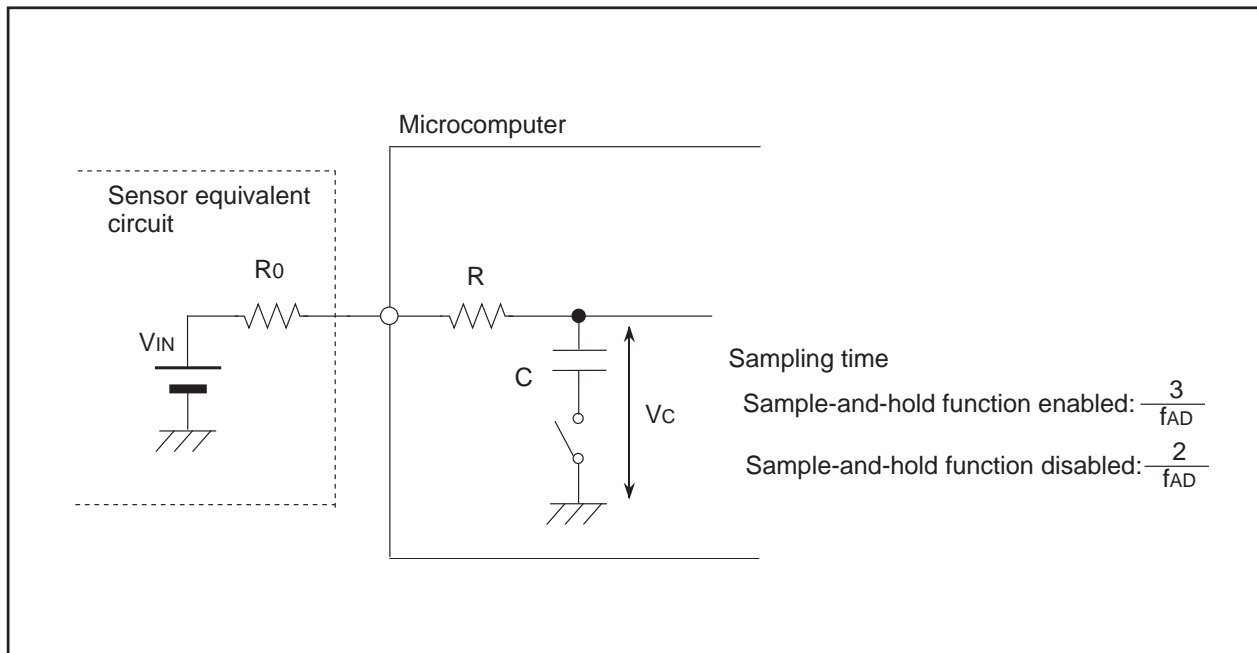
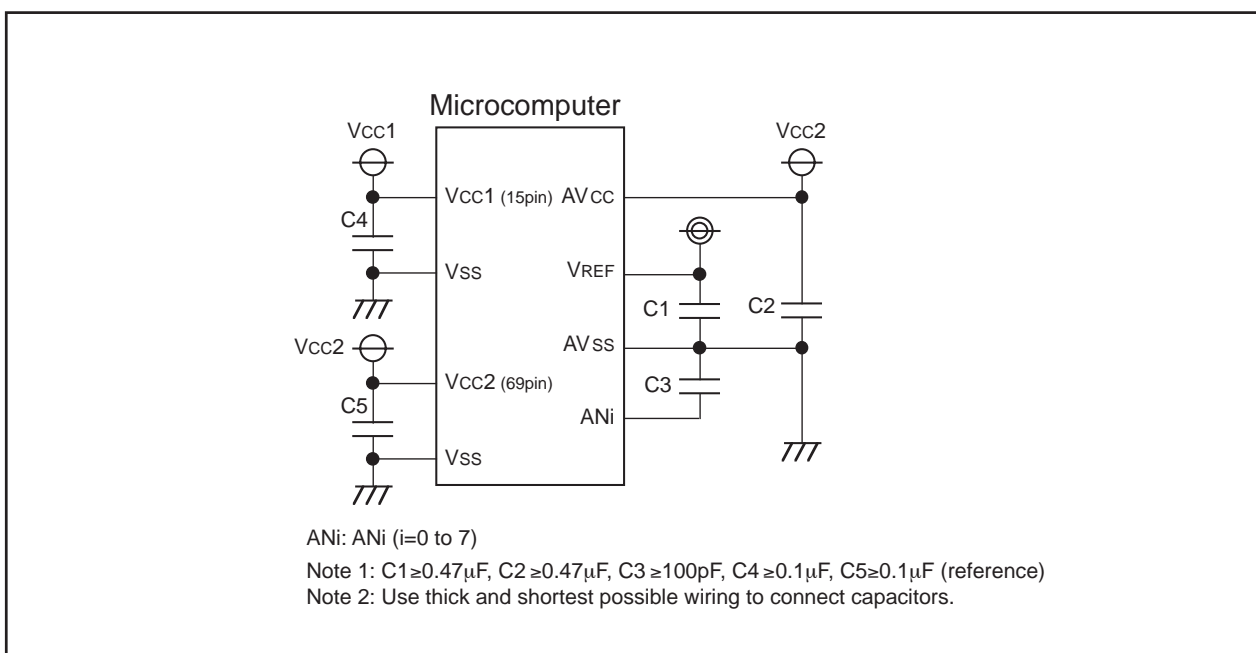


Figure 2.12.10. Analog Input Pin and External Sensor Equivalent Circuit

(f) Caution of Using A-D Converter

- (1) Make sure the port direction bits for those pins that are used as analog inputs are set to "0" (input mode). Also, if the ADCON0 register's TGR bit = 1 (external trigger), make sure the port direction bit for the $\overline{\text{ADTRG}}$ pin is set to "0" (input mode).
- (2) When using key input interrupts, do not use any of the four AN4 to AN7 pins as analog inputs. (A key input interrupt request is generated when the A-D input voltage goes low.)
- (3) To prevent noise-induced device malfunction or latchup, as well as to reduce conversion errors, insert capacitors between the AVCC, VREF, and analog input pins (ANi (i=0 to 7)) each and the AVSS pin. Similarly, insert a capacitor between the VCC pin and the VSS pin. Figure 2.12.11 is an example connection of each pin.
- (4) If the CPU reads the ADi register (i = 0 to 7) at the same time the conversion result is stored in the ADi register after completion of A-D conversion, an incorrect value may be stored in the ADi register. This problem occurs when a divide-by-n clock derived from the main clock or a subclock is selected for CPU clock.
 - When operating in one-shot or single-sweep mode
Check to see that A-D conversion is completed before reading the target ADi register. (Check the IR bit in the ADIC register to see if A-D conversion is completed.)
 - When operating in repeat mode or repeat sweep mode 0 or 1
Use the main clock for CPU clock directly without dividing it.
- (5) If A-D conversion is forcibly terminated while in progress by setting the ADCON0 register's ADST bit to "0" (A-D conversion halted), the conversion result of the A-D converter is indeterminate. The contents of ADi registers irrelevant to A-D conversion may also become indeterminate. If while A-D conversion is underway the ADST bit is cleared to "0" in a program, ignore the values of all ADi registers.

**Figure 2.12.11. Vcc, Vss, AVcc, AVss, VREF and ANi Connection**

2.13 CRC Calculation

The Cyclic Redundancy Check (CRC) operation 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 consists of 16 bits which are generated for each data block in given length, separated in 8 bit units. After the initial value is set in the CRCD register, the CRC code is set in that register each time one byte of data is written to the CRCIN register. CRC code generation for one-byte data is finished in two cycles.

Figure 2.13.1 shows the block diagram of the CRC circuit. Figure 2.13.2 shows the CRC-related registers. Figure 2.13.3 shows the calculation example using the CRC operation.

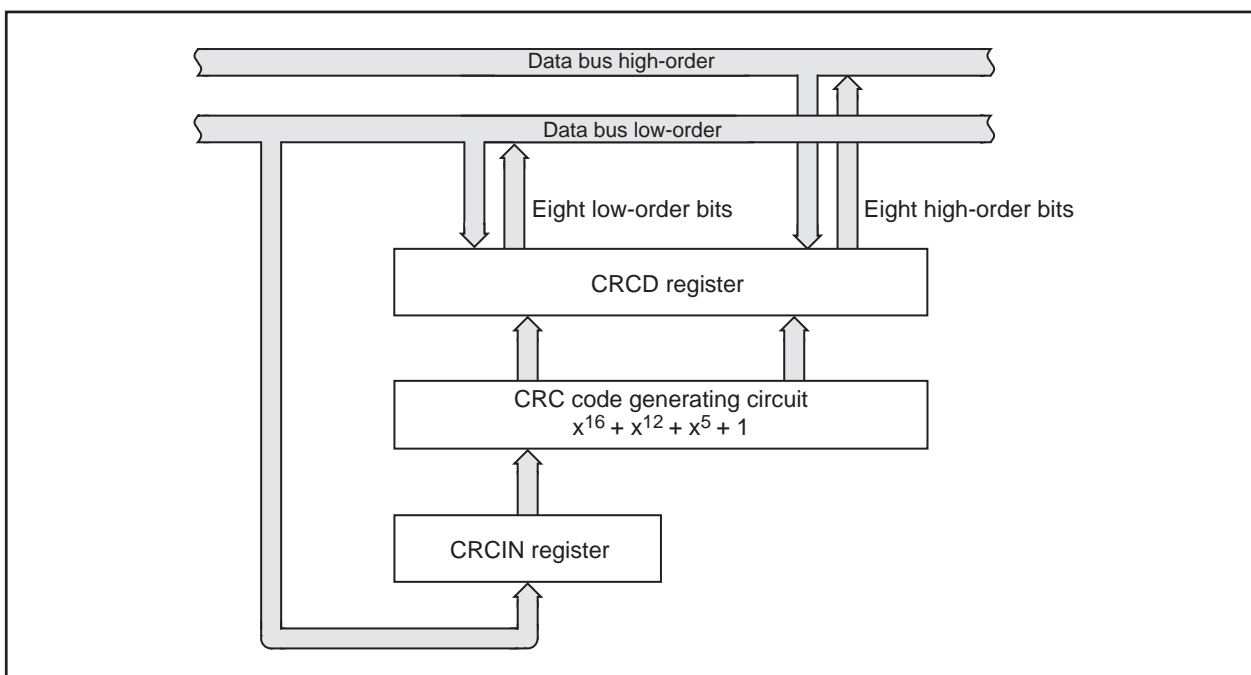


Figure 2.13.1. CRC Circuit Block Diagram

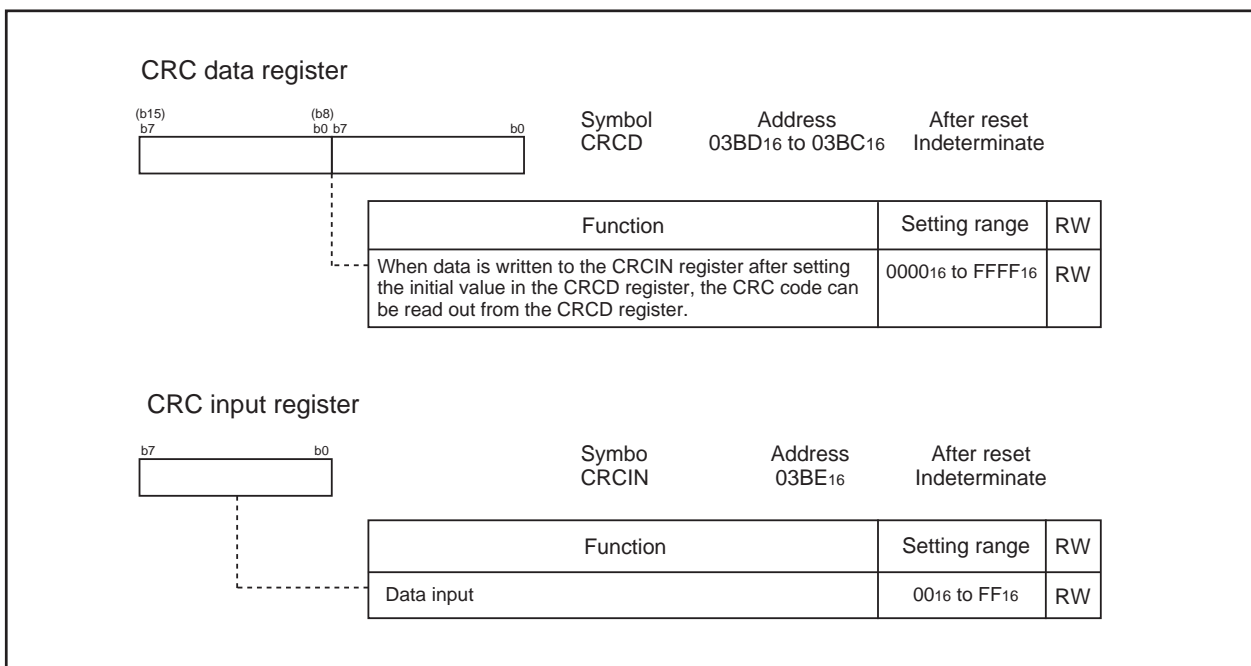


Figure 2.13.2. CRCD Register and CRCIN Register

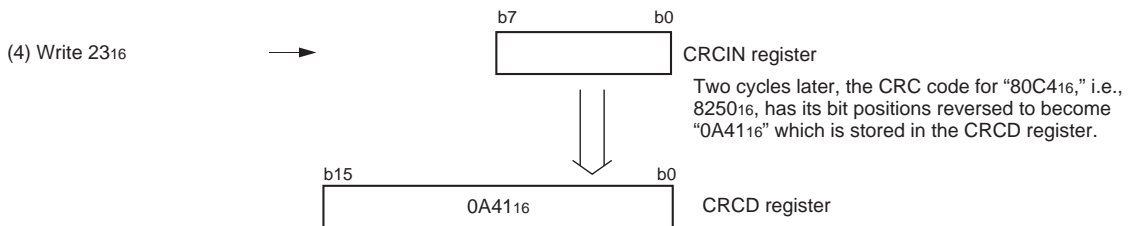
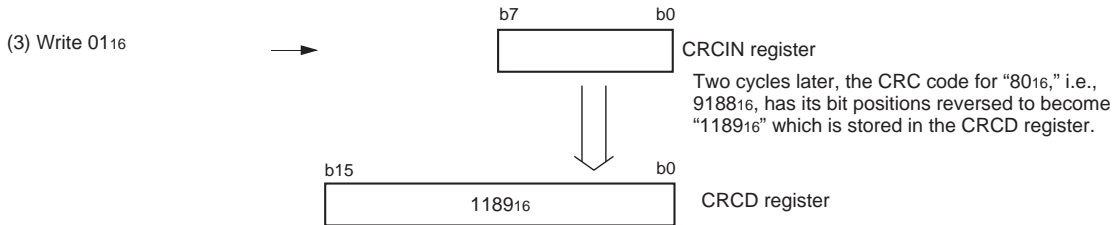
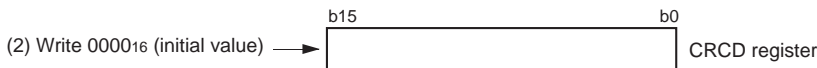
Setup procedure and CRC operation when generating CRC code "80C416"

(a) CRC operation performed by the M16C

CRC code: Remainder of a division in which the value written to the CRCIN register with its bit positions reversed is divided by the generator polynomial
 Generator polynomial: $X^{16} + X^{12} + X^5 + 1$ (1 0001 0000 0010 0001₂)

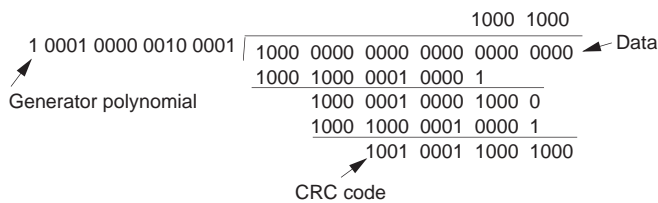
(b) Setting procedure

(1) Reverse the bit positions of the value "80C416" bitwise in a program.
 "8016" → "0116", "C416" → "2316"



(c) Details of CRC operation

In the case of (3) above, the value written to the CRCIN register "01₁₆ (00000001₂)" has its bit positions reversed to become "10000000₂." The value "1000 0000 0000 0000 0000 0000₂" derived from that by adding 16 digits and the CRCD register's initial value "000016" are added, the result of which is divided by the generator polynomial using modulo-2 arithmetic.



Modulo-2 operation is operation that complies with the law given below.

0 + 0 = 0
 0 + 1 = 1
 1 + 0 = 1
 1 + 1 = 0
 -1 = 1

The value "0001 0001 1000 1001₂ (118916)" derived from the remainder "1001 0001 1000 1000₂ (918816)" by reversing its bit positions may be read from the CRCD register.

If operation (4) above is performed subsequently, the value written to the CRCIN register "23₁₆ (00100011₂)" has its bit positions reversed to become "11000100₂. The value "1100 0100 0000 0000 0000 0000₂" derived from that by adding 16 digits and the remainder in (3) "1001 0001 1000 1000₂" which is left in the CRCD register are added, the result of which is divided by the generator polynomial using modulo-2 arithmetic. The value "0000 1010 0100 0001₂ (0A4116)" derived from the remainder by reversing its bit positions may be read from the CRCD register.

Figure 2.13.3. CRC Calculation

2.14 Expansion Function

2.14.1 Expansion function description

Expansion function consists of CRC operation function, data slice function and humming decoder function. Each function is controlled by expansion memories.

(1) CRC operation function

It performs error detection of a code, and error correction.

(2) Data slice function

It performs data acquisition to get such format data as below.

Hardware : TELETEXT, PDC, VPS, VBI, EPG-J, XDS and WSS

Software : CCD, WSS and VBI-ID

(3) Humming decoder function

It performs 8/4 humming and 24/18 humming

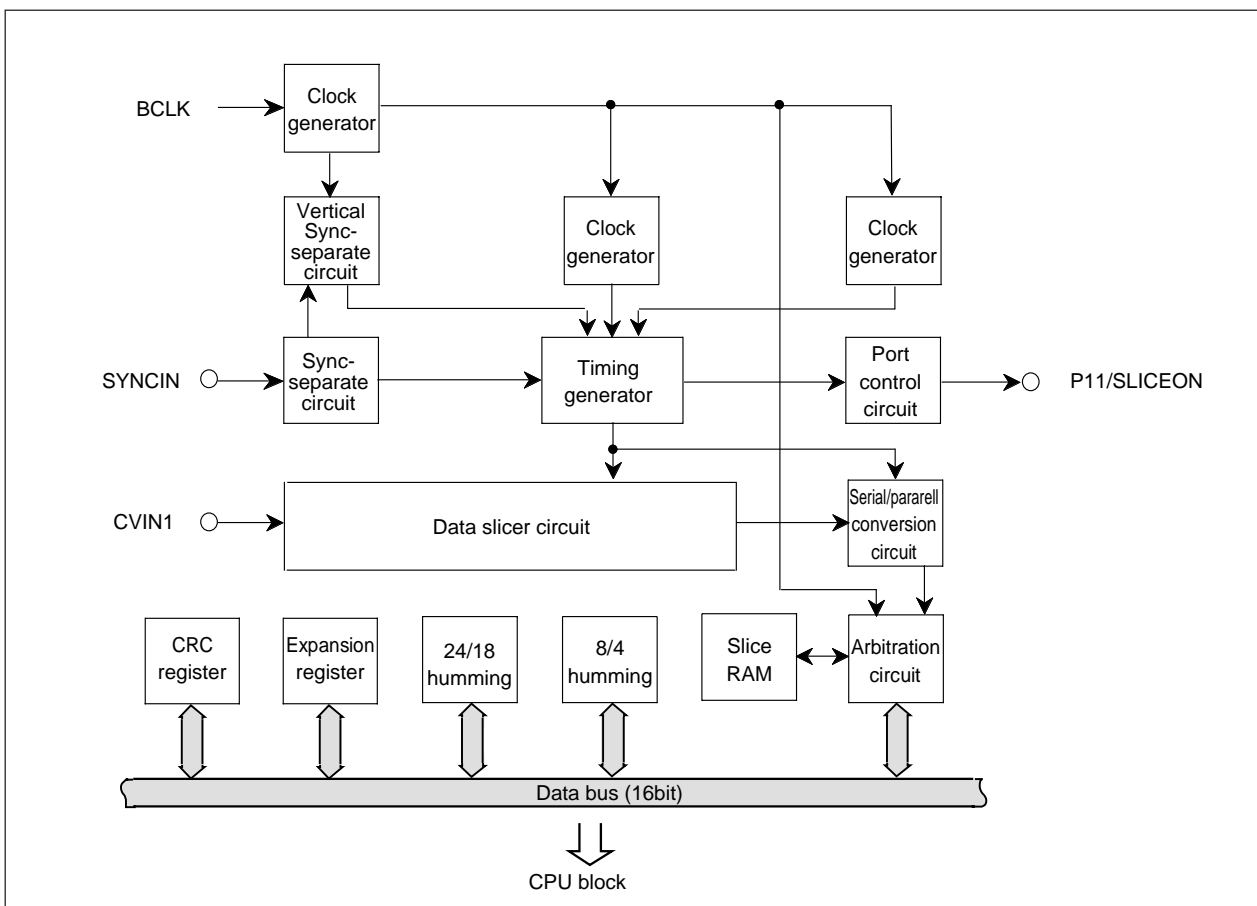


Figure 2.14.1 Block diagram of expansion function

2.14.2 Expansion memory

Expansion function memory is divided by 3 patterns ; Slice RAM, CRC registers and expansion registers (Humming decoder operates by the register placed on SFR). Data writing and read out to the Slice RAM, CRC registers and the expansion registers are carried out per 16 bit unit by the data setting register (addresses 020E₁₆, 0210₁₆, 0212₁₆, 0214₁₆, 0216₁₆ and 0218₁₆) placed on SFR.

Contents of each memory and data setting register are shown in Table 2.14.1.

Table 2.14.1 Expansion memory composition

Expansion memory	Contents	Data setting register
Slice RAM	This register holds acquired data.	Slice RAM address control register (020E ₁₆) Slice RAM data control register (0210 ₁₆)
CRC register	This register controls a set up generation polynomial and code data.	CRC register address control register (0212 ₁₆) CRC register data control register (0214 ₁₆)
Expansion register	This register performs data slicer control and VBI encoder control.	Expansion register address control register (0216 ₁₆) Expansion register data control register (0218 ₁₆)

2.14.3 Slice RAM

Slice RAM stores 18-line slice data. There are several types of Slice data : PDC, VPS, VBI, XDS, WSS, etc. All data are stored to addresses which corresponds to slice line (ex. 22 line' data is stored to addresses 200₁₆ to 217₁₆). 24 addresses (SR00x to SR17x) are prepared for 1 line, slice data is stored in order from LSB side. Then, slice data type and field information are stored to the top address of each line.

Slice RAM composition is shown in Table 2.14.2.

Table 2.14.2 Slice RAM composition

Slice RAM addresses (SA9 to SA0)	SD15	SD14	SD13	SD12	SD11	SD10	SD9	SD8	SD7	SD6	SD5	SD4	SD3	SD2	SD1	SD0	Remarks	
000 ₁₆	SR00F	SR00E	SR00D	SR00C	SR00B	SR00A	SR009	SR008	SR007	SR006	SR005	SR004	SR003	SR002	SR001	SR000	6th line or 318th line slice data	
001 ₁₆	SR01F	SR01E	SR01D	SR01C	SR01B	SR01A	SR019	SR018	SR017	SR016	SR015	SR014	SR013	SR012	SR011	SR010		
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮		
016 ₁₆	SR16F	SR16E	SR16D	SR16C	SR16B	SR16A	SR169	SR168	SR167	SR166	SR165	SR164	SR163	SR162	SR161	SR160		
017 ₁₆	SR17F	SR17E	SR17D	SR17C	SR17B	SR17A	SR179	SR178	SR177	SR176	SR175	SR174	SR173	SR172	SR171	SR170		
018 ₁₆	Unused area																	
⋮																		
01F ₁₆																		
020 ₁₆	SR00F	SR00E	SR00D	SR00C	SR00B	SR00A	SR009	SR008	SR007	SR006	SR005	SR004	SR003	SR002	SR001	SR000	7th line or 319th line slice data	
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮		
037 ₁₆	SR17F	SR17E	SR17D	SR17C	SR17B	SR17A	SR179	SR178	SR177	SR176	SR175	SR174	SR173	SR172	SR171	SR170		
040 ₁₆	⋮																8th line to 21th line or 320th line to 333 line slice data	
⋮																		
1F7 ₁₆																		
200 ₁₆	SR00F	SR00E	SR00D	SR00C	SR00B	SR00A	SR009	SR008	SR007	SR006	SR005	SR004	SR003	SR002	SR001	SR000	22th line or 334th line slice data	
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮		
217 ₁₆	SR17F	SR17E	SR17D	SR17C	SR17B	SR17A	SR179	SR178	SR177	SR176	SR175	SR174	SR173	SR172	SR171	SR170		
220 ₁₆	SR00F	SR00E	SR00D	SR00C	SR00B	SR00A	SR009	SR008	SR007	SR006	SR005	SR004	SR003	SR002	SR001	SR000	23th line or 335th line slice data	
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮		
237 ₁₆	SR17F	SR17E	SR17D	SR17C	SR17B	SR17A	SR179	SR178	SR177	SR176	SR175	SR174	SR173	SR172	SR171	SR170		

For accessing to Slice RAM data, set accessing address (SA9 to SA0) (shown in Table 2.14.2) to Slice RAM address control register (address 020E₁₆). Then read out data from Slice RAM data control register (address 0210₁₆). When end the data reading, Slice RAM address control register increments address automatically. Then, next address data reading is possible. Do not access to unused area of each character codes. Must set address to each line because unused area has no address' automatically increment.

Slice RAM bit composition is shown in Figure 2.14.2, Slice RAM access registers are shown in Figure 2.14.3 and Slice RAM access block diagram is shown in Figure 2.14.4.

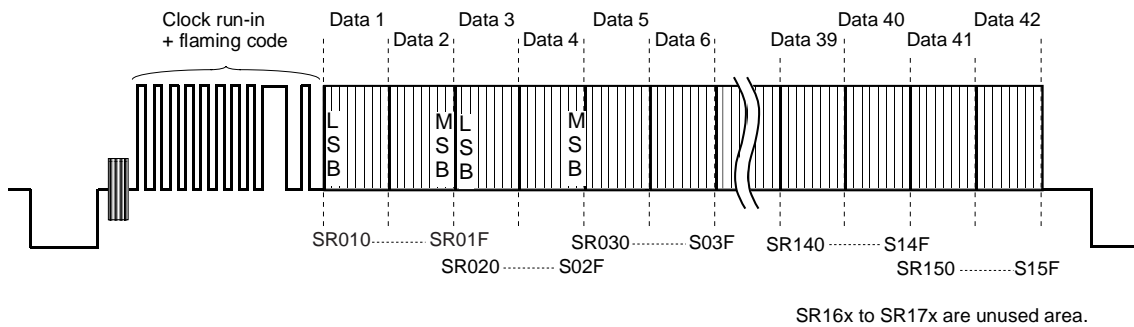
The each head address of the address is corresponded to slice line following slice information.

	SR00F to SR004	SR003	SR002	SR001	SR000
Line register 3	0	field * (Note)	0	1	1
Line register 2	0	field * (Note)	0	1	0
Line register 1	0	field * (Note)	0	0	1
Other	0	0	0	0	0

Note : * the first field : 1
the second field : 0

(1) PDC

In case of the PDC data, 16 bits (2 data) are stored for the 1 address from the LSB side.

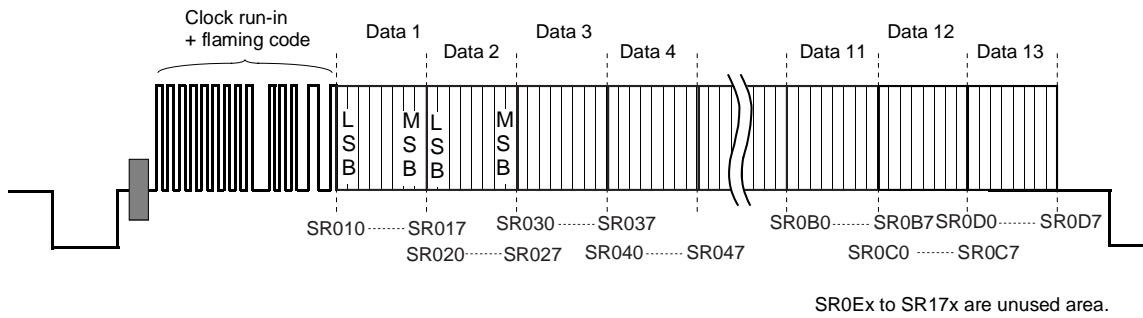


(2) VPS

In case of the VPS data or the VBI data, 8 bits (a data) are stored for an address from the LSB side.

Low-order 8 bits hold the slice data. And, high-order 8 bits hold warning bit, when the send data is not recognized as bi-phase type.

The case of bi-phase data = "1,0" or "0,1" (the bi-phase type) becomes "0" for this warning bit, and it becomes "1" in bi-phase data = "0,0" or "1,1" (it is not the bi-phase type). (For example, bi-phase data of SR011 is "0,0" or "1,1", "1" is set to SR019.)



(3) EPG-J

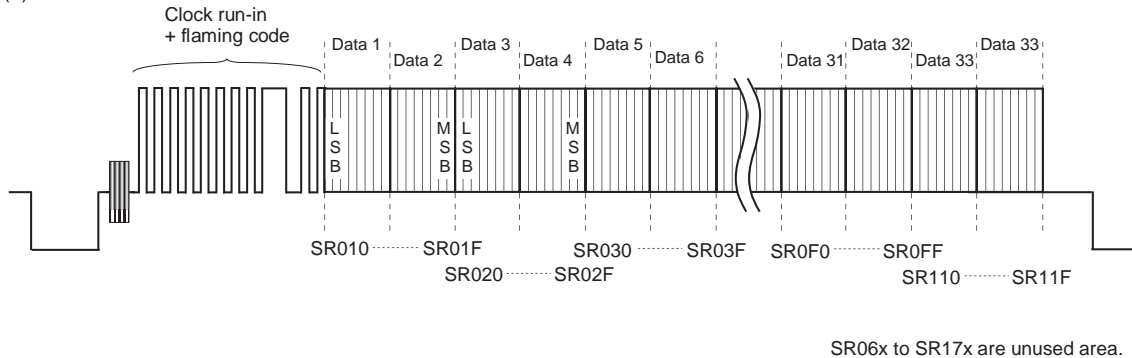


Figure 2.14.2 Slice RAM bit composition

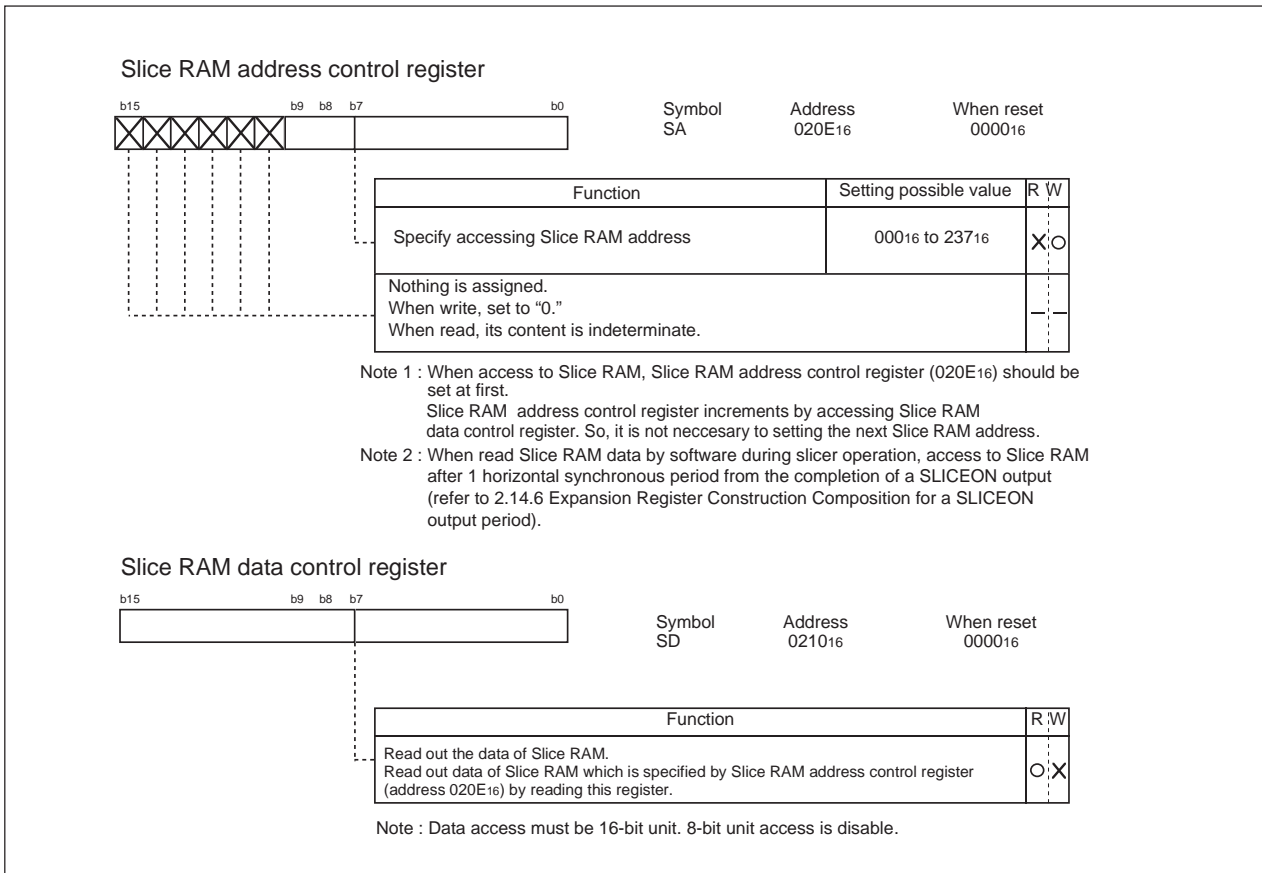


Figure 2.14.3 Slice RAM access registers

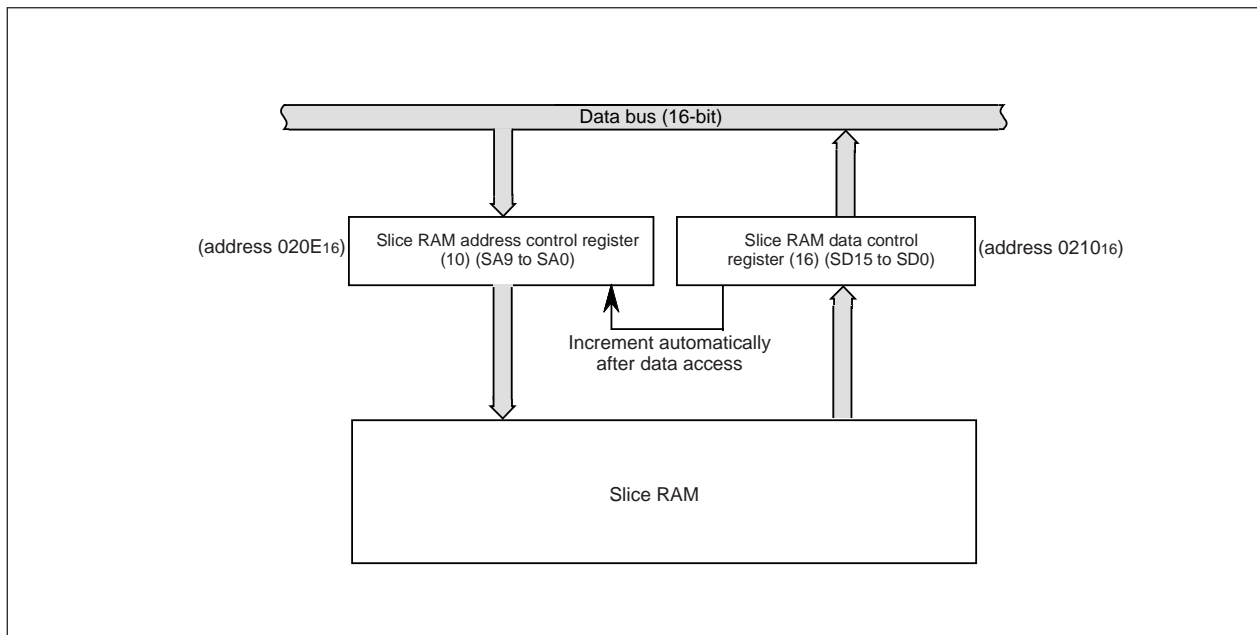


Figure 2.14.4 Slice RAM access block diagram

2.14.4 CRC Operation Circuit (EPG-J)

CRC operation circuit (EPG-J) is a circuit for performing error detection and error correction by the 272-190 shortening difference set cyclic code which is a coding system in a data multiplex broadcast. CRC register consists of registers shown in Figure 2.14.5. CRC register can perform error detection and error correction by majority logic by setting up a generator polinomial, code data, etc. CRC register composition is shown in Table 2.14.3.

Table 2.14.3 CRC register composition

CA3 to CA0	CD15	CD14	CD13	CD12	CD11	CD10	CD9	CD8	CD7	CD6	CD5	CD4	CD3	CD2	CD1	CD0	Remarks
0016	DAOUT15	DAOUT14	DAOUT13	DAOUT12	DAOUT11	DAOUT10	DAOUT9	DAOUT8	DAOUT7	DAOUT6	DAOUT5	DAOUT4	DAOUT3	DAOUT2	DAOUT1	DAOUT0	
0116						CRC_ERR10	CRC_ERR09	CRC_ERR08	CRC_ERR07	CRC_ERR06	CRC_ERR05	CRC_ERR04	CRC_ERR03	CRC_ERR02	CRC_ERR01	CRC_ERR00	
0216	CRC_66	CRC_67	CRC_68	CRC_69	CRC_70	CRC_71	CRC_72	CRC_73	CRC_74	CRC_75	CRC_76	CRC_77	CRC_78	CRC_79	CRC_80	CRC_81	
0316	CRC_50	CRC_51	CRC_52	CRC_53	CRC_54	CRC_55	CRC_56	CRC_57	CRC_58	CRC_59	CRC_60	CRC_61	CRC_62	CRC_63	CRC_64	CRC_65	
0416	CRC_34	CRC_35	CRC_36	CRC_37	CRC_38	CRC_39	CRC_40	CRC_41	CRC_42	CRC_43	CRC_44	CRC_45	CRC_46	CRC_47	CRC_48	CRC_49	
0516	CRC_18	CRC_19	CRC_20	CRC_21	CRC_22	CRC_23	CRC_24	CRC_25	CRC_26	CRC_27	CRC_28	CRC_29	CRC_30	CRC_31	CRC_32	CRC_33	
0616	CRC_02	CRC_03	CRC_04	CRC_05	CRC_06	CRC_07	CRC_08	CRC_09	CRC_10	CRC_11	CRC_12	CRC_13	CRC_14	CRC_15	CRC_16	CRC_17	
0716															CRC_00	CRC_01	
0816																	
0916																	
0A16																	
0B16																	
0C16																	
0D16																	

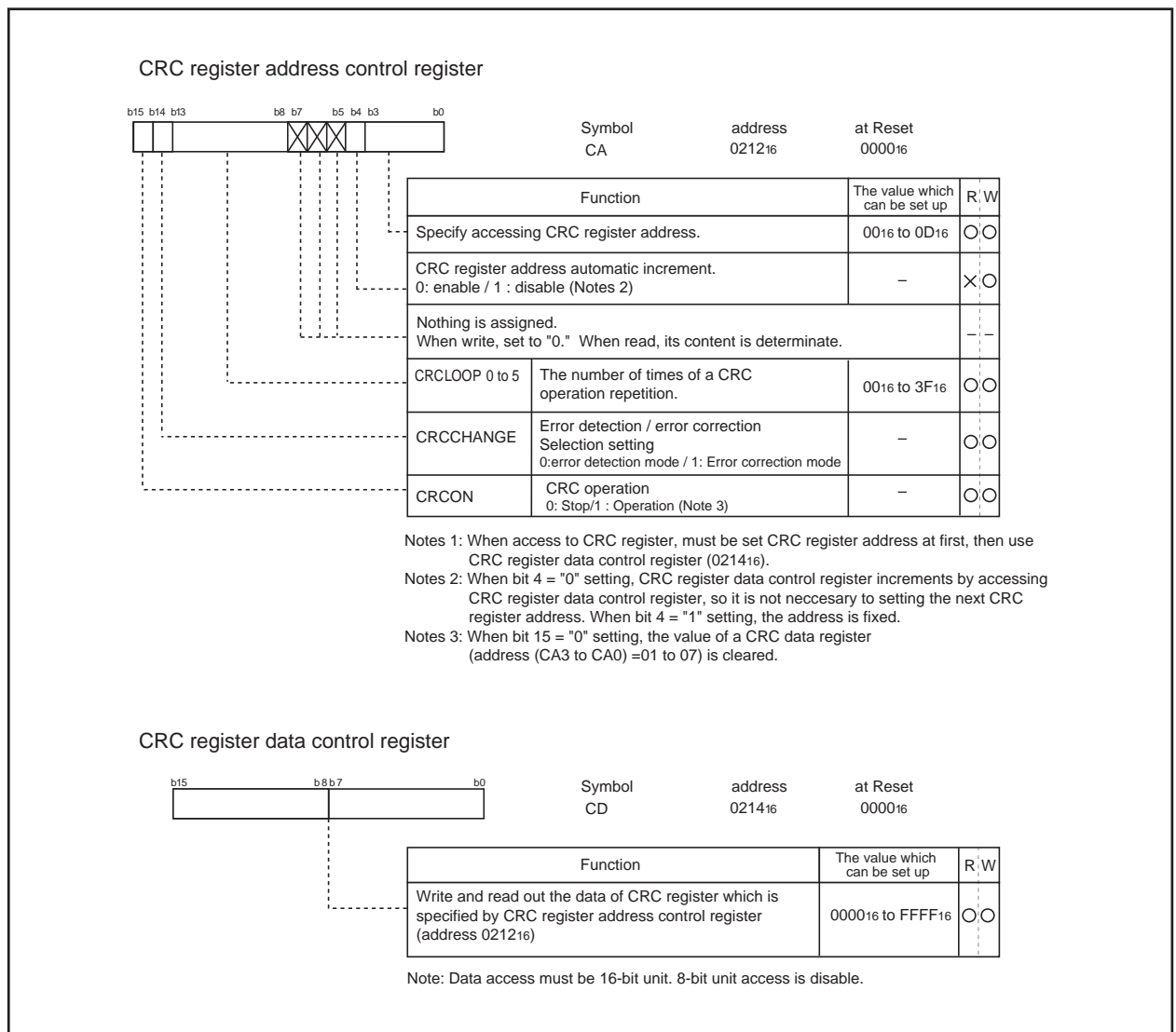


Figure 2.14.5 Composition of CRC register access related register

For accessing to CRC register data, set accessing address (CA3 to CA0) (shown in Table 2.14.3) to CRC register address control register (address 0212₁₆). Then write data (CD15 to CD0) by CRC register data control register (address 0214₁₆). When end the data accessing, CRC register address control register increments address automatically. Then, next address data writing is possible. CRC register access registers are shown in Figure 2.14.5, CRC register access block diagram is shown in Figure 2.14.6. The operation example of CRC operation circuit is shown in Figure 2.14.7. The example of program is shown in Figure 2.14.8, and CRC register bit compositions are shown in p186 to 194.

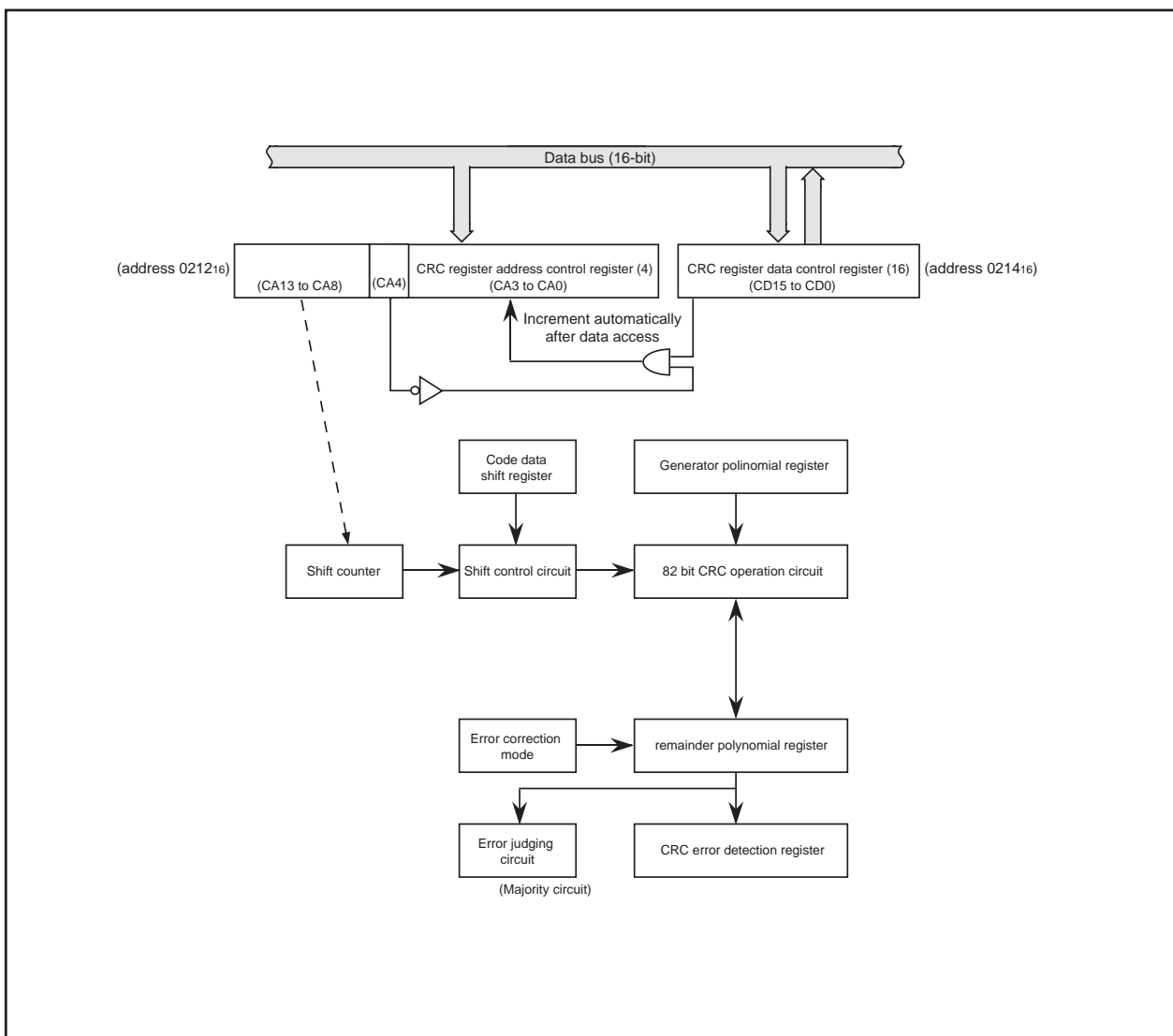


Figure 2.14.6 Access block diagram for CRC registers

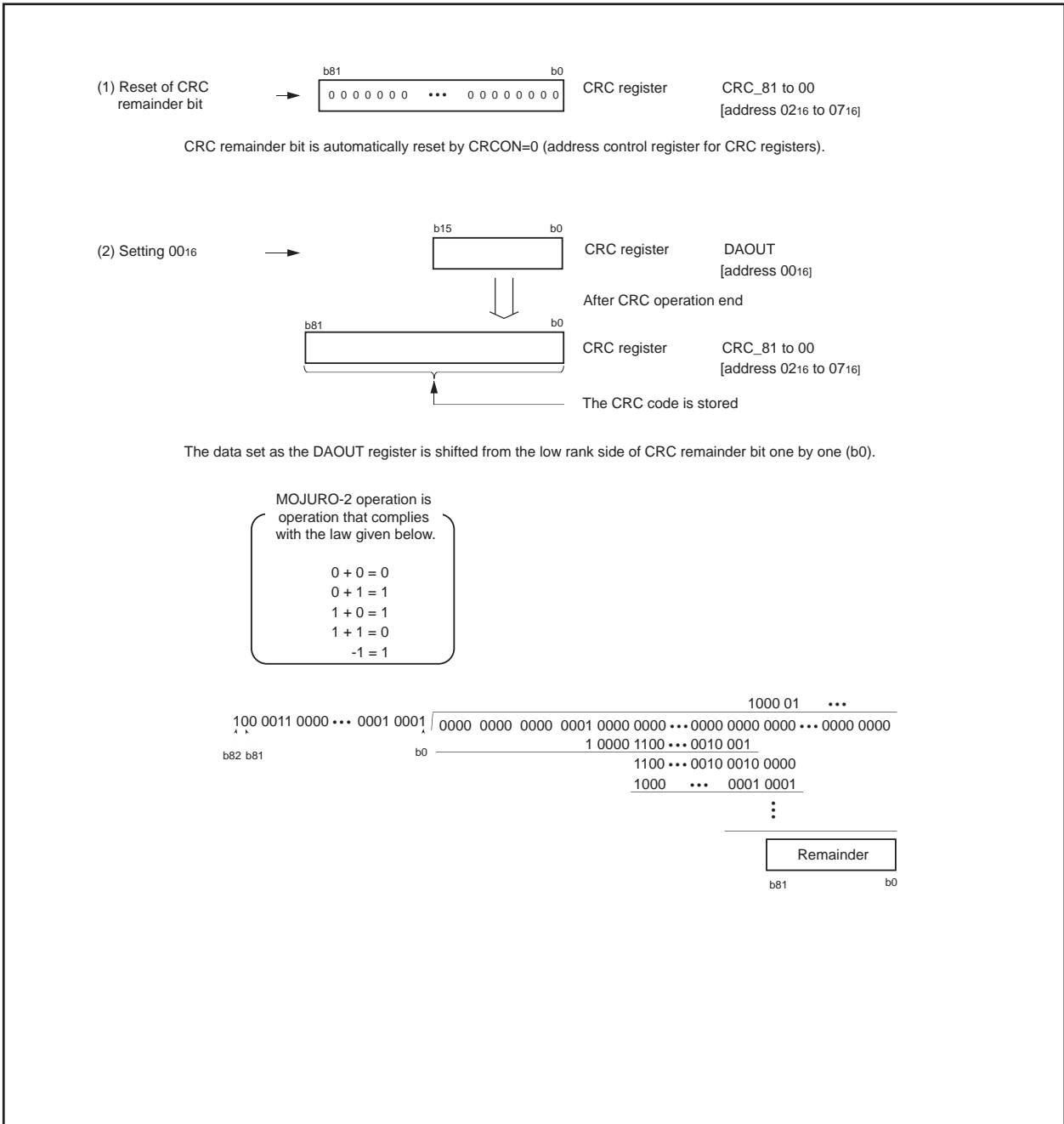


Figure 2.14.7 Example of operation of CRC operation circuit


```

-----
;
; Equations (Constant definition)
-----
_CRC_ADRS      .equ      00212h      ; SFR address of CRC register address control register
_CRC_DATA     .equ      00214h      ; SFR address of CRC register data control register
SLICE_WORD_NUM .equ      17         ; Code data length (in units of word)

-----
; Macro definition
-----
_wait .macro
      nop
      nop
      nop
.endm

-----
; CRC operation routine
-----
----- Writing of code data -----
mov.w #0000H      , _CRC_ADRS      ; Initialization of CRC register address control register
mov.w #9010H      , _CRC_ADRS      ; Set up of CRCON=1, CRCCHANGE=0, CRCLOOP=10H, Increment=ON, and CRC address=00H.
mov.w #0000H      , A0             ; Initialization of a loop variable (A0)

L18:
  cmp.w #SLICE_WORD_NUM*2      , A0      ; Branch label
  jgeu L20                      ; Comparison of the loop variable
  ; Go to L20 if writing code data is finished.
  lde.w _CrcCodeData[A0]      , _CRC_DATA ; Writing code data to the code data shift register.
  add.w #0002H              , A0        ; Increment of the address storing code data.
  jmp L18                    ; Return to the head of this loop.

L20:
  ; Branch label
  ; Dummy shift
  ; After finishing writing 272-bit code data,
  ; shift a bit for dummy surely in error correction mode.
  ; Specifying 1-bit is set up by CRCLOOP=01H.
  mov.w #8100H      , _CRC_ADRS      ; Set up of CRCON=1, CRCCHANGE=0, CRCLOOP=10H, Increment=OFF, and CRC address=00H.
  _wait
  mov.w #0000H      , _CRC_DATA      ; Writing data to the code data shift register for dummy shift.

----- Error detection -----
; Since the address automatic increment in dummy shift (Increment=OFF), set CRC address=01H here.
; When accessing other CRC registers, the processing shown in the following two lines is necessary.
  mov.w #9001H      , _CRC_ADRS      ; Set up of CRCON=1, CRCCHANGE=0, CRCLOOP=10H, Increment=OFF and CRC address=01H.
  _wait
  ; Wait
  mov.w _CRC_DATA   , R0             ; Read of CRC error detection register.
  cmp.w #0000H      , R0             ; Judgement of CRC error.
  jeq L16           ; In the case of R0=0, branch to L16 since CRC error has not occurred (CRC error correction is skipped).

----- Error correction -----
  mov.w #0D010H     , _CRC_ADRS      ; Set up of CRCON=1, CRCCHANGE=1, CRCLOOP=10H, Increment=ON and CRC address=00H.
  _wait
  mov.w #0000H      , A0             ; Initialization of a loop variable (A0)

L22:
  cmp.w #SLICE_WORD_NUM      , A0      ; Branch label
  ; Comparison of the loop variable
  jgeu L24                      ; Go to L24 if correction of error data is finished.
  lde.w _CrcCodeData[A0]      , _CRC_DATA ; Writing code data to the code data shift register.
  jsr _waitlong              ; Wait for finish of error correction.
  mov.w _CRC_DATA   , _CrcCodeData[A0] ; Read of error correction data in the address storing code data.
  add.w #0002H              , A0        ; Increment of the address storing code data.
  jmp L22                    ; Return to the head of this loop.

L24:
  ; Branch label

----- The check of error correction data -----
  mov.w #8111H      , _CRC_ADRS      ; Set up of CRCON=1, CRCCHANGE=0, CRCLOOP=10H, Increment=ON and CRC address=00H
  _wait
  mov.w _CRC_DATA   , R0             ; Error check after error correction. R0=000H if correction is performed.

L16:
  ; Error check after error correction. R0=000H if correction is performed.

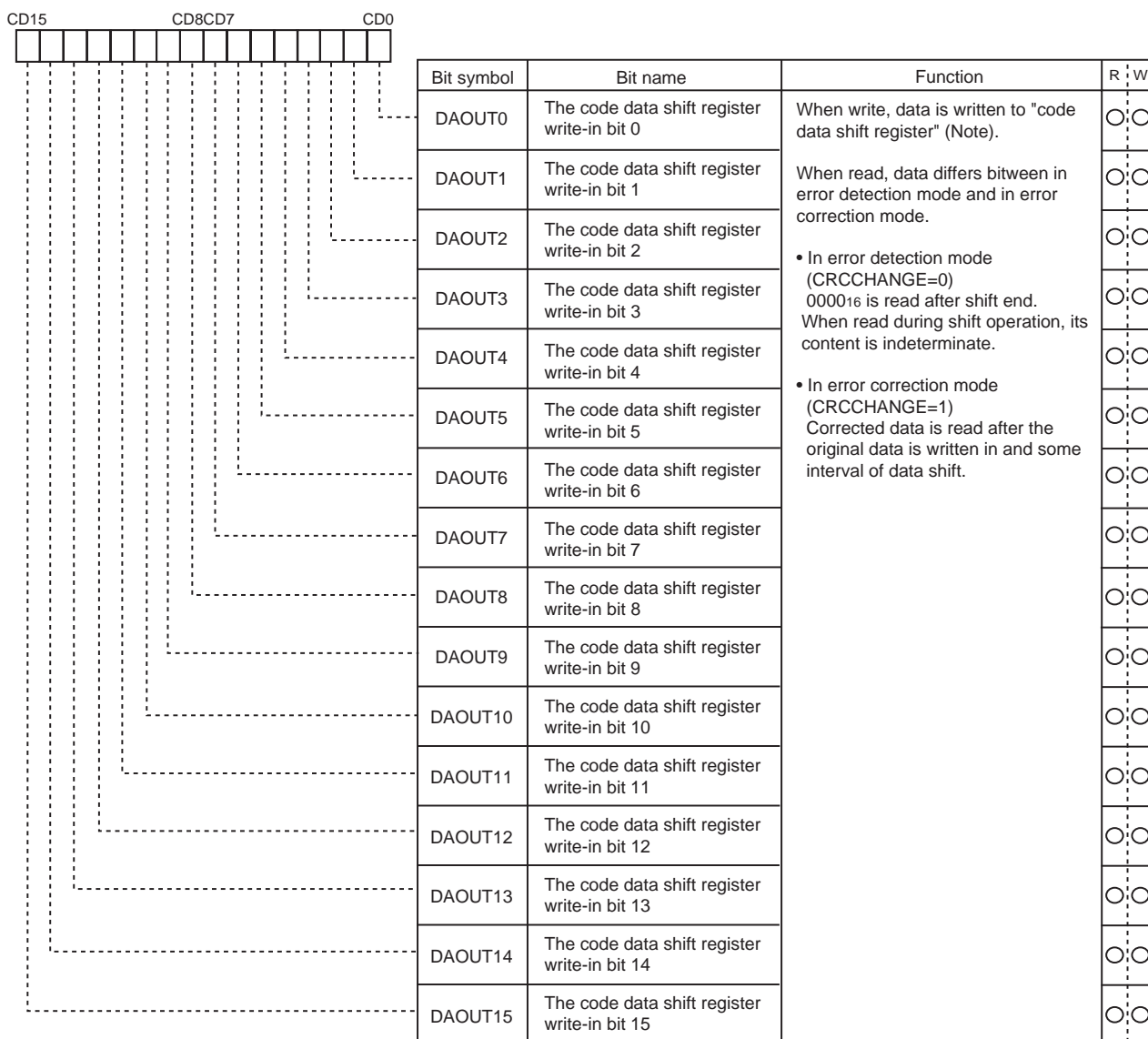
-----
; The function sample for weight for error correction
-----
      .align
      .glob _waitlong
_waitlong:
      rts
; Function label

```

Figure 2.14.8 Example of program

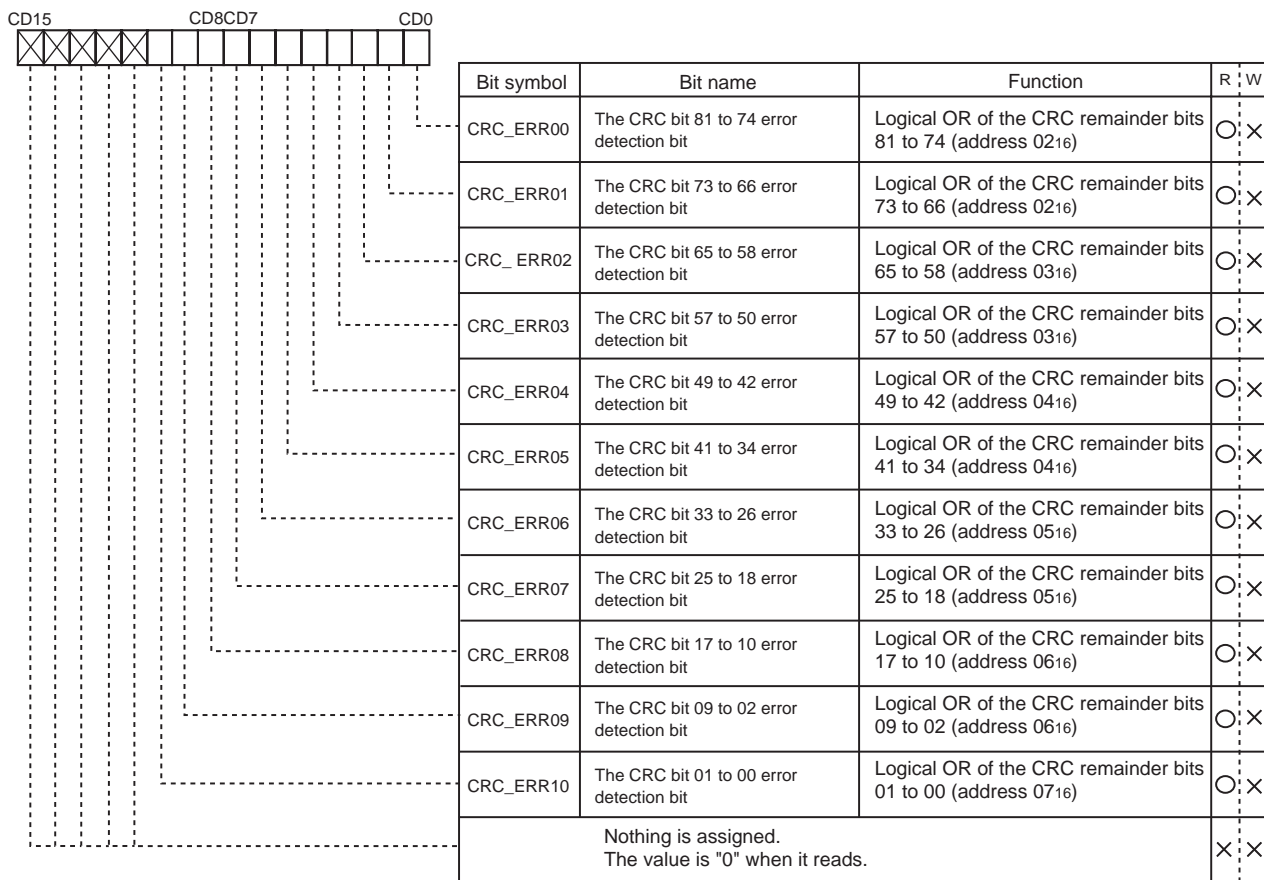
Bit composition of a CRC register

(1) Address 00₁₆ (=CA3 to 0)

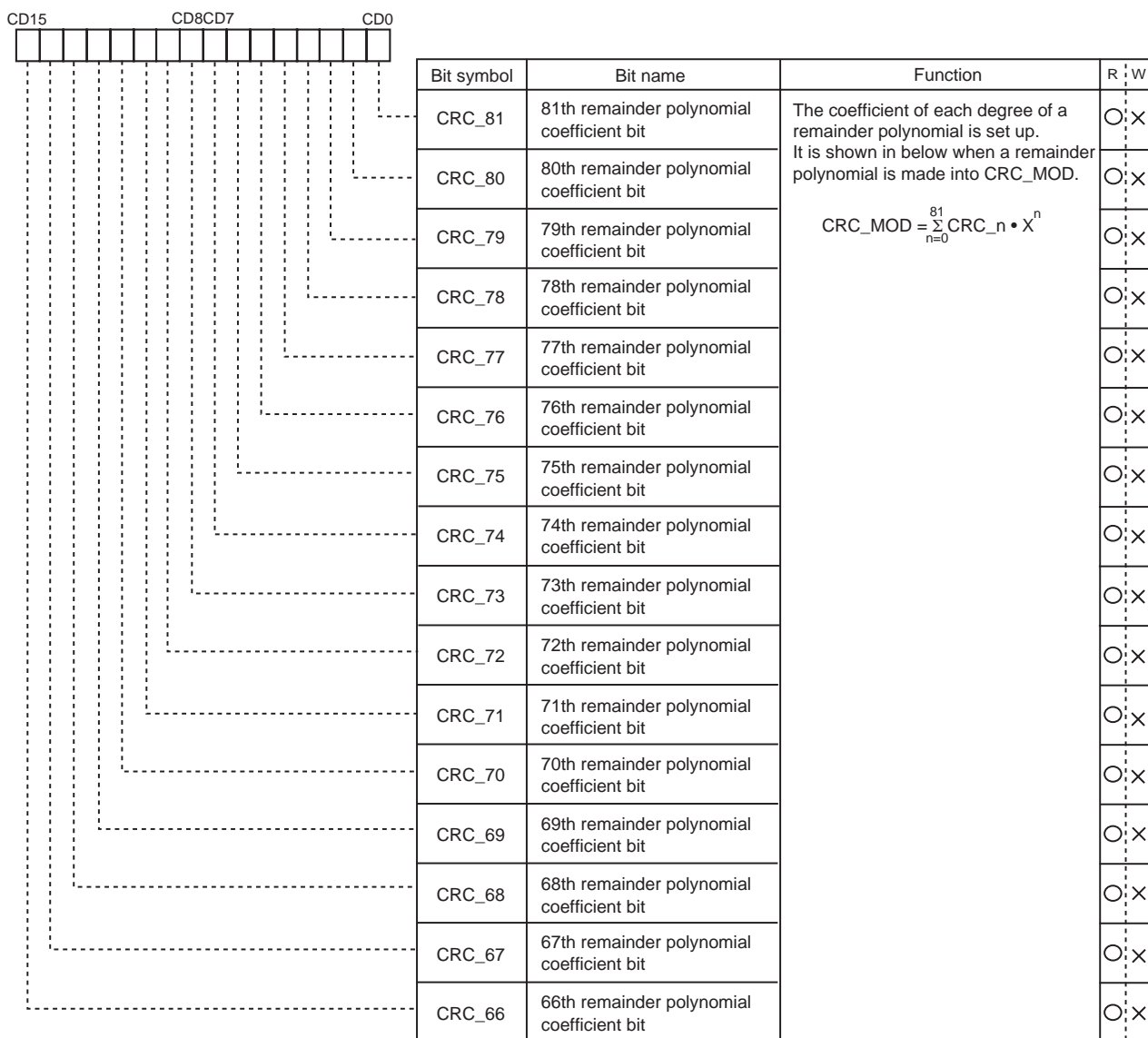


Note: Refer to Figure 2.14.16 Access block diagram for CRC registers.

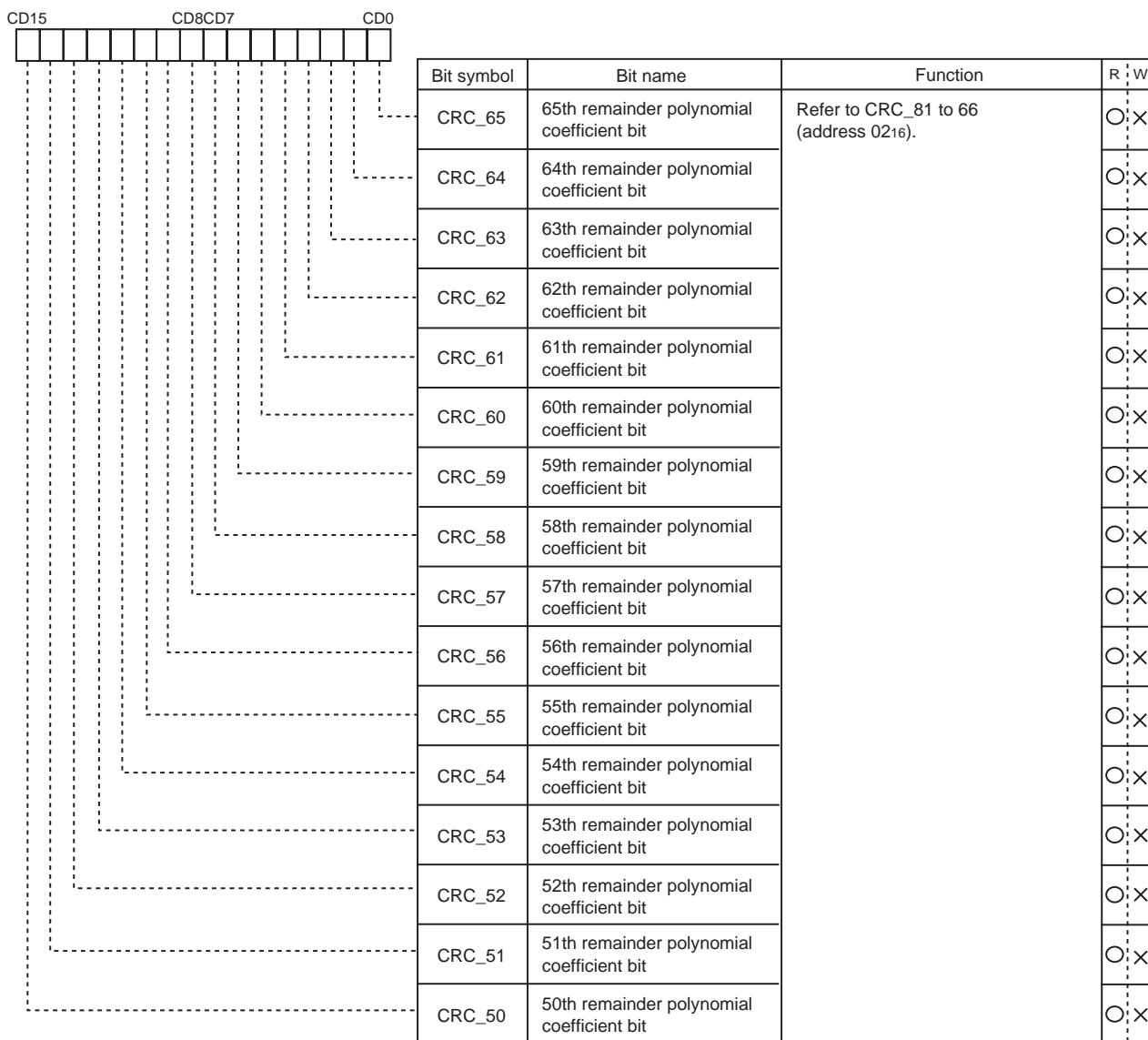
(2) Address 01₁₆ (=CA3 to 0)



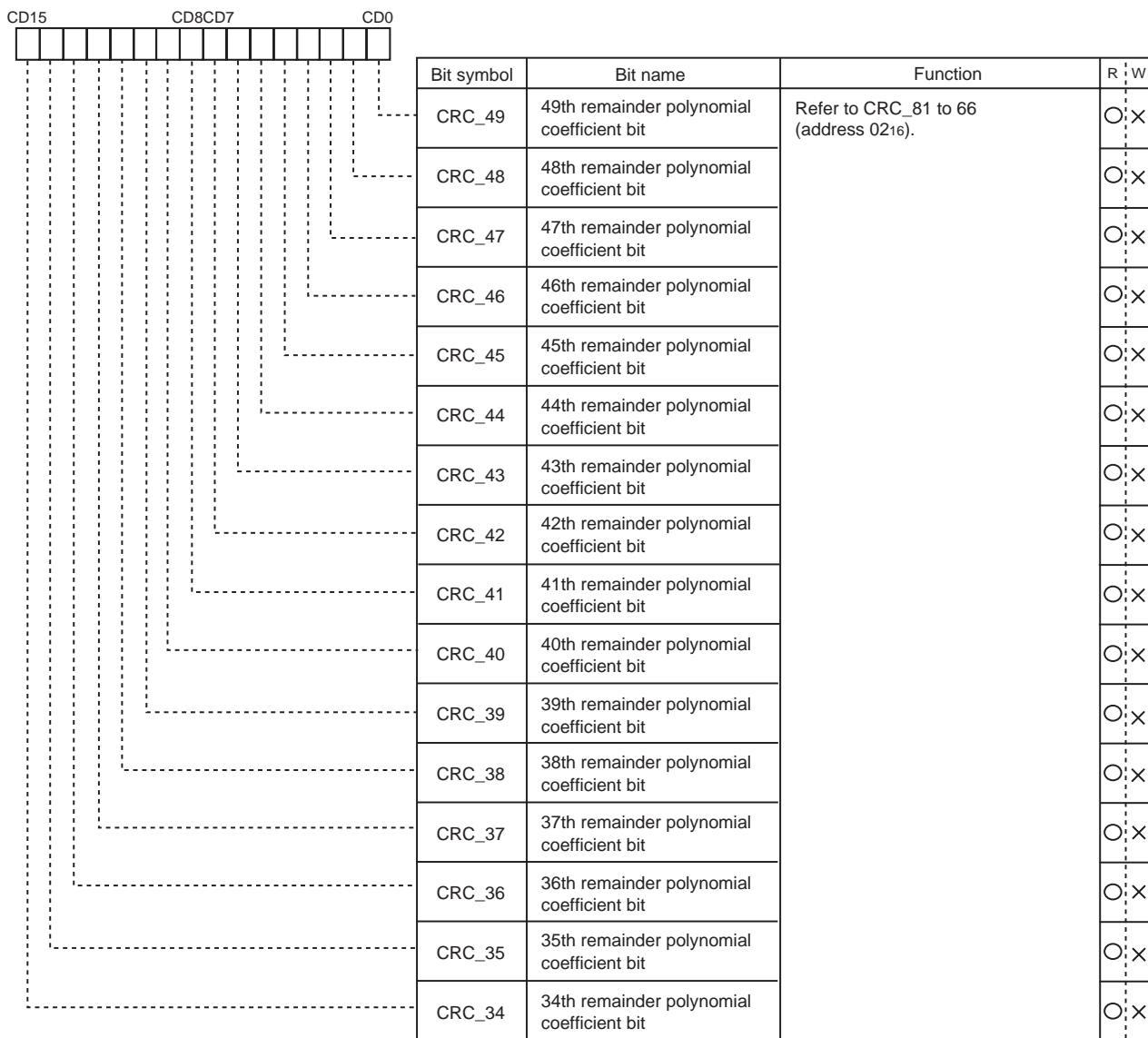
(3) Address 0216 (=CA3 to 0)



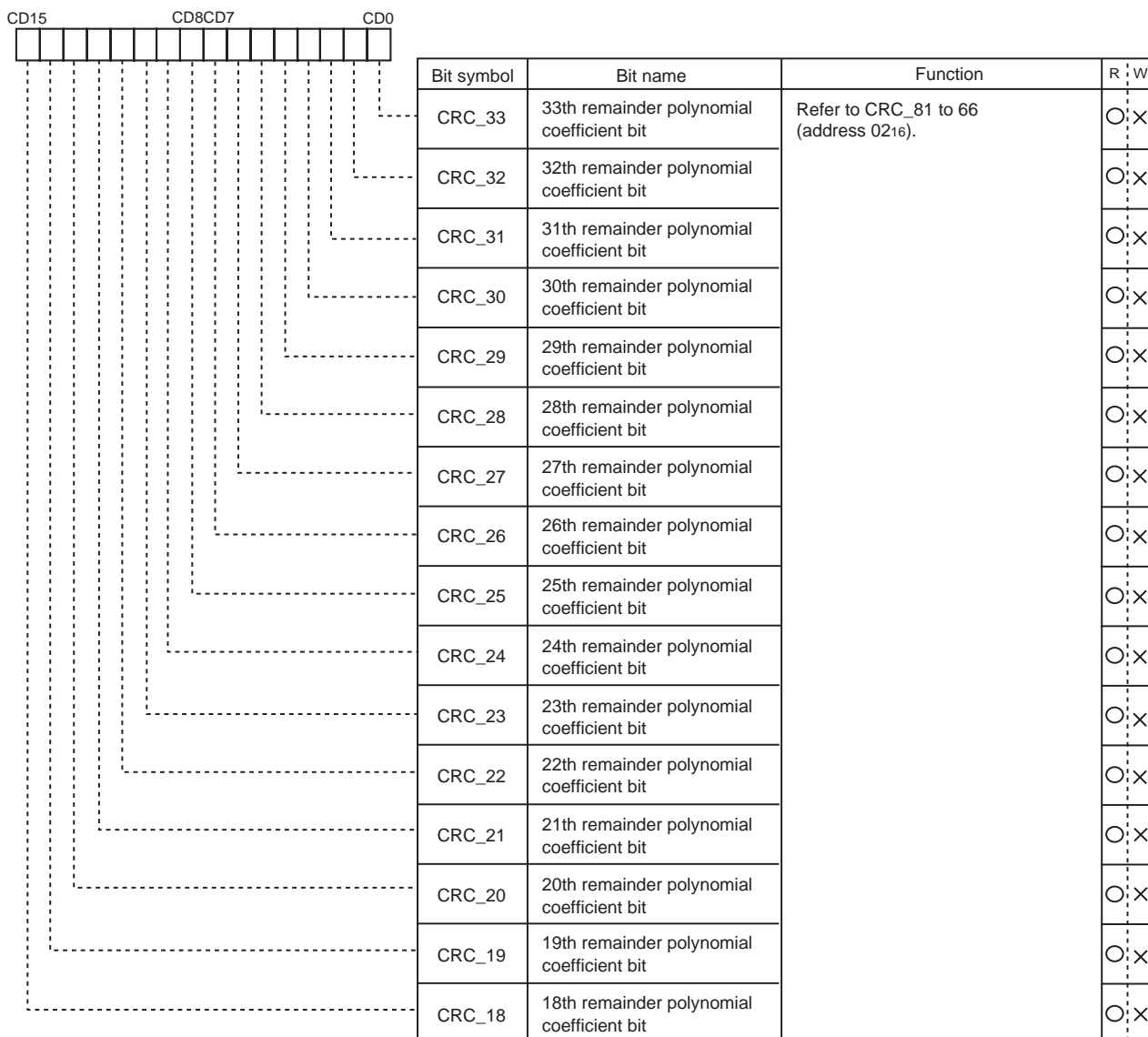
(4) Address 0316 (=CA3 to 0)



(5) Address 0416 (=CA3 to 0)



(6) Address 0516 (=CA3 to 0)



(7) Address 06₁₆ (=CA3 to 0)



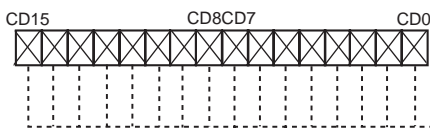
Bit symbol	Bit name	Function	R	W
CRC_17	17th remainder polynomial coefficient bit	Refer to CRC_81 to 66 (address 02 ₁₆).	0	X
CRC_16	16th remainder polynomial coefficient bit		0	X
CRC_15	15th remainder polynomial coefficient bit		0	X
CRC_14	14th remainder polynomial coefficient bit		0	X
CRC_13	13th remainder polynomial coefficient bit		0	X
CRC_12	12th remainder polynomial coefficient bit		0	X
CRC_11	11th remainder polynomial coefficient bit		0	X
CRC_10	10th remainder polynomial coefficient bit		0	X
CRC_09	09th remainder polynomial coefficient bit		0	X
CRC_08	08th remainder polynomial coefficient bit		0	X
CRC_07	07th remainder polynomial coefficient bit		0	X
CRC_06	06th remainder polynomial coefficient bit		0	X
CRC_05	05th remainder polynomial coefficient bit		0	X
CRC_04	04th remainder polynomial coefficient bit		0	X
CRC_03	03rd remainder polynomial coefficient bit	0	X	
CRC_02	02nd remainder polynomial coefficient bit	0	X	

(8) Address 07₁₆ (=CA3 to 0)



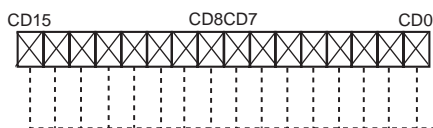
Bit symbol	Bit name	Function	R	W
CRC_01	01st remainder polynomial coefficient bit	Refer to CRC_81 to 66 (address 02 ₁₆).	0	X
CRC_00	00th remainder polynomial coefficient bit		0	X
Nothing is assigned. The value is "0" when it reads.			X	X

(9) Address 08₁₆ (=CA3 to 0)



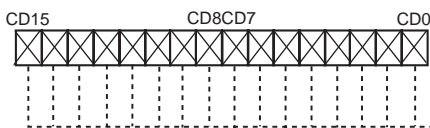
Bit symbol	Bit name	Function	R	W
Nothing is assigned. The value is unfixed when it reads.			X	X

(10) Address 09₁₆ (=CA3 to 0)



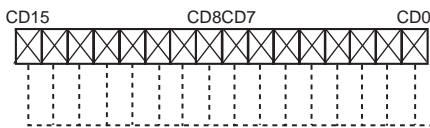
Bit symbol	Bit name	Function	R	W
Nothing is assigned. The value is unfixed when it reads.			X	X

(11) Address 0A₁₆ (=CA3 to 0)



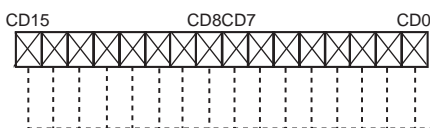
Bit symbol	Bit name	Function	R	W
Nothing is assigned. The value is unfixed when it reads.			X	X

(12) Address 0B₁₆ (=CA3 to 0)



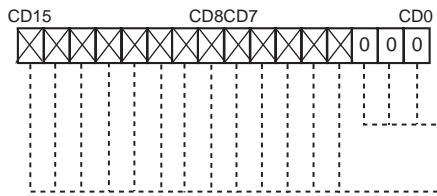
Bit symbol	Bit name	Function	R	W
Nothing is assigned. The value is unfixed when it reads.			X	X

(13) Address 0C₁₆ (=CA3 to 0)



Bit symbol	Bit name	Function	R	W
Nothing is assigned. The value is unfixed when it reads.			X	X

(14) Address 0D16 (=CA3 to 0)



Bit symbol	Bit name	Function	R	W
	Reserved bit	Must set to "0."	○	○
	Nothing is assigned. The value is unfixed when it reads.		×	×

2.14.5 Expansion Register

Control Data slice function. Expansion register composition is shown in Table 2.14.4.

Table 2.14.4 Expansion register composition

DA5 to DA0	DD15	DD14	DD13	DD12	DD11	DD10	DD9	DD8	DD7	DD6	DD5	DD4	DD3	DD2	DD1	DD0	Remarks
00e	LN5_EV0	LN4_EV0	LN3_EV0	LN2_EV0	LN1_EV0	LN0_EV0	LN5_EV1	LN8_EV0	LN7_EV0	LN6_EV0	LN5_EV0	LN4_EV0	LN3_EV0	LN2_EV0	LN1_EV0	LN0_EV0	Line register
01e	LN5_EV1	LN4_EV1	LN3_EV1	LN2_EV1	LN1_EV1	LN0_EV1	LN8_EV1	LN7_EV1	LN6_EV1	LN5_EV1	LN4_EV1	LN3_EV1	LN2_EV1	LN1_EV1	LN0_EV1		
02e	LN7_EV0	LN6_EV0	LN5_EV0	LN4_EV0	LN3_EV0	LN2_EV0	LN1_EV0	LN8_OD0	LN7_OD0	LN6_OD0	LN5_OD0	LN4_OD0	LN3_OD0	LN2_OD0	LN1_OD0	LN0_OD0	
03e	LN7_EV1	LN6_EV1	LN5_EV1	LN4_EV1	LN3_EV1	LN2_EV1	LN1_EV1	LN8_OD1	LN7_OD1	LN6_OD1	LN5_OD1	LN4_OD1	LN3_OD1	LN2_OD1	LN1_OD1	LN0_OD1	
04e	LN5_OD0	LN4_OD0	LN3_OD0	LN2_OD0	LN1_OD0	LN0_OD0	LN8_OD0	LN7_OD0	LN6_OD0	LN5_OD0	LN4_OD0	LN3_OD0	LN2_OD0	LN1_OD0	LN0_OD0		
05e	LN5_OD1	LN4_OD1	LN3_OD1	LN2_OD1	LN1_OD1	LN0_OD1	LN8_OD1	LN7_OD1	LN6_OD1	LN5_OD1	LN4_OD1	LN3_OD1	LN2_OD1	LN1_OD1	LN0_OD1		
06e	DIVS1	DIVS0	SELVCO	FLC12	FLC11	FLC10	FLC9	FLC8	FLC7	FLC6	FLC5	FLC4	FLC3	FLC2	FLC1	FLC0	Status register 1
07e	FLC15	FLC14	CHK_FLG13	CHK_FLG12	CHK_FLG11	CHK_FLG10	CHK_FLG9	CHK_FLG8	CHK_FLG7	CHK_FLG6	CHK_FLG5	CHK_FLG4	CHK_FLG3	CHK_FLG2	CHK_FLG1	CHK_FLG0	
08e	CHK_FLG15	CHK_FLG14	CHK_FLG13	CHK_FLG12	CHK_FLG11	SLS_VL	GET_HP1	GET_HP0	SELVCO	SELVCO	SELVCO	SELVCO	SELVCO	SELVCO	SELVCO	SELVCO	
09e	GETPEEK3	GETPEEK2	GETPEEK1	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	
0Ae	GETPEEK3	GETPEEK2	GETPEEK1	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	
0Be	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	
0Ce	DIVS1	DIVS0	SELVCO	FLC12	FLC11	FLC10	FLC9	FLC8	FLC7	FLC6	FLC5	FLC4	FLC3	FLC2	FLC1	FLC0	Status register 2
0De	FLC15	FLC14	FLC13	FLC12	FLC11	FLC10	FLC9	FLC8	FLC7	FLC6	FLC5	FLC4	FLC3	FLC2	FLC1	FLC0	
0Ee	CHK_FLG15	CHK_FLG14	CHK_FLG13	CHK_FLG12	CHK_FLG11	CHK_FLG10	CHK_FLG9	CHK_FLG8	CHK_FLG7	CHK_FLG6	CHK_FLG5	CHK_FLG4	CHK_FLG3	CHK_FLG2	CHK_FLG1	CHK_FLG0	
0Fe	CHK_FLG15	CHK_FLG14	CHK_FLG13	CHK_FLG12	CHK_FLG11	SLS_VL	GET_HP1	GET_HP0	SELVCO	SELVCO	SELVCO	SELVCO	SELVCO	SELVCO	SELVCO	SELVCO	
10e	GETPEEK3	GETPEEK2	GETPEEK1	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	
11e	GETPEEK3	GETPEEK2	GETPEEK1	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	
12e	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	
13e	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	
14e	DIVS1	DIVS0	SELVCO	FLC12	FLC11	FLC10	FLC9	FLC8	FLC7	FLC6	FLC5	FLC4	FLC3	FLC2	FLC1	FLC0	Status register 3
15e	FLC15	FLC14	FLC13	FLC12	FLC11	FLC10	FLC9	FLC8	FLC7	FLC6	FLC5	FLC4	FLC3	FLC2	FLC1	FLC0	
16e	CHK_FLG15	CHK_FLG14	CHK_FLG13	CHK_FLG12	CHK_FLG11	CHK_FLG10	CHK_FLG9	CHK_FLG8	CHK_FLG7	CHK_FLG6	CHK_FLG5	CHK_FLG4	CHK_FLG3	CHK_FLG2	CHK_FLG1	CHK_FLG0	
17e	CHK_FLG15	CHK_FLG14	CHK_FLG13	CHK_FLG12	CHK_FLG11	SLS_VL	GET_HP1	GET_HP0	SELVCO	SELVCO	SELVCO	SELVCO	SELVCO	SELVCO	SELVCO	SELVCO	
18e	GETPEEK3	GETPEEK2	GETPEEK1	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	
19e	GETPEEK3	GETPEEK2	GETPEEK1	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	GETPEEK0	
1Ae	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	
1Be	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	FRAM	
1Ce	ADSTART	VREF1															
1De																	
1Ee																	
1Fe																	
20e		MPAL	NXP														
21e																	
22e	HMB4SEL																
23e																	
24e																	
25e																	
26e	DIVV_CK7	DIVV_CK6	DIVV_CK5	DIVV_CK4	DIVV_CK3	DIVV_CK2	DIVV_CK1	DIVV_CK0	DIVV_CK7	DIVV_CK6	DIVV_CK5	DIVV_CK4	DIVV_CK3	DIVV_CK2	DIVV_CK1	DIVV_CK0	
27e																	
28e																	
29e																	
2Ae																	
2Be																	
2Ce																	
2De																	
2Ee																	
2Fe																	
30e																	
31e																	
32e																	
33e																	
34e																	
35e																	
36e																	
37e																	
38e																	
39e																	
3Ae																	
3Be																	
3Ce																	
3De																	
3Ee																	
3Fe																	

For accessing to expansion register data, set accessing address (DA5 to DA0) (shown in Table 2.14.4) to expansion register address control register (address 0216₁₆). Then write data (DD15 to DD0) to expansion register data control register (address 0218₁₆). When end the data accessing, expansion register address control register increments address automatically. Then, next address data writing is possible.

Expansion register access registers are shown in Figure 2.14.9, expansion register access block diagram is shown in Figure 2.14.10, and expansion register bit compositions are shown in p197 to 227.

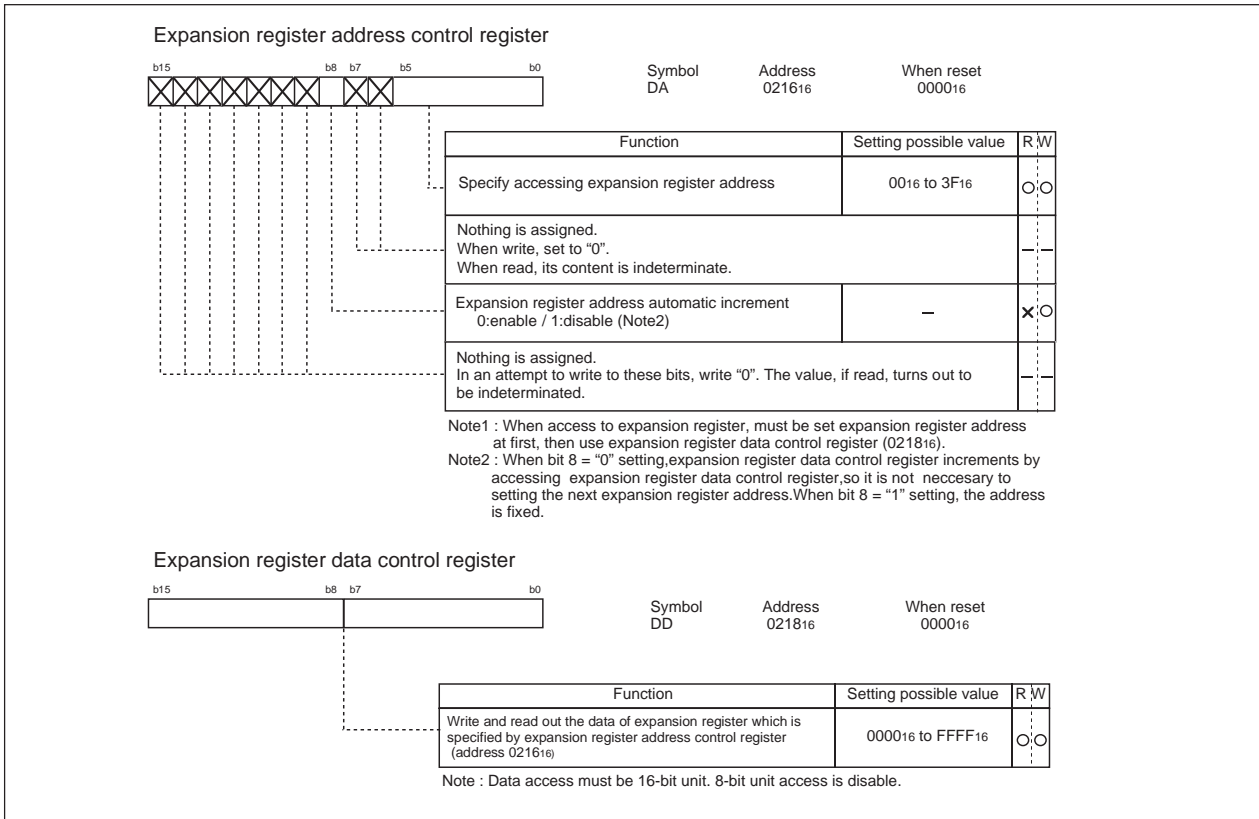


Figure 2.14.9 Expansion register access registers composition

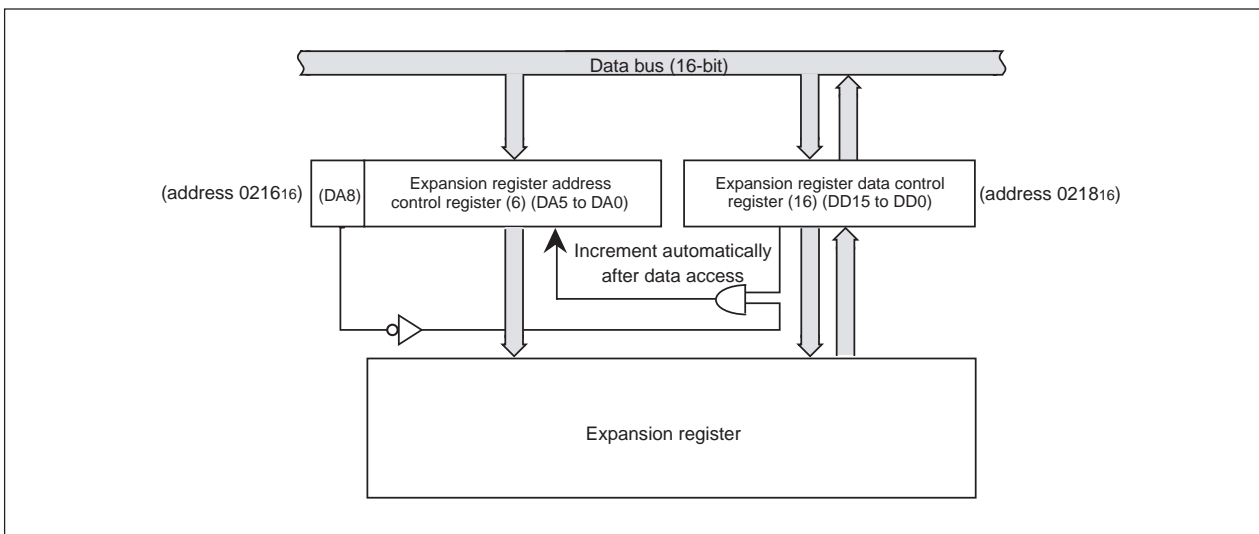
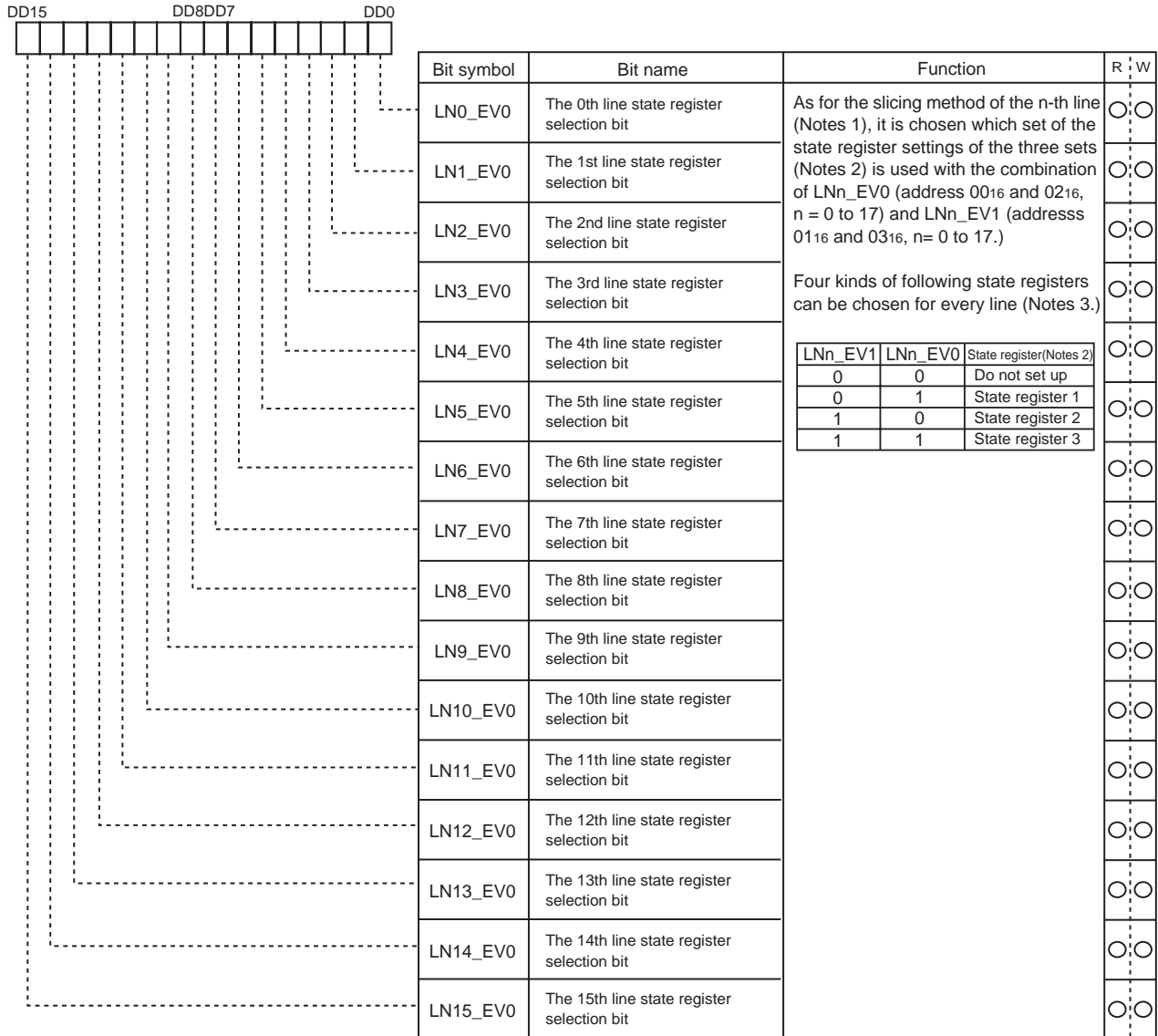


Figure 2.14.10 Expansion register access block diagram

Bit composition of an expansion register

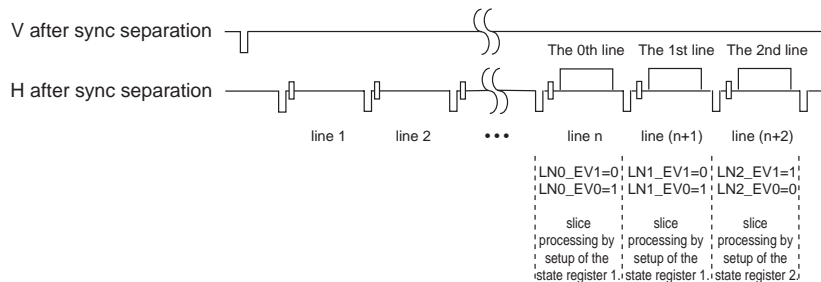
(1) Address 00₁₆ (=DA5 to 0)



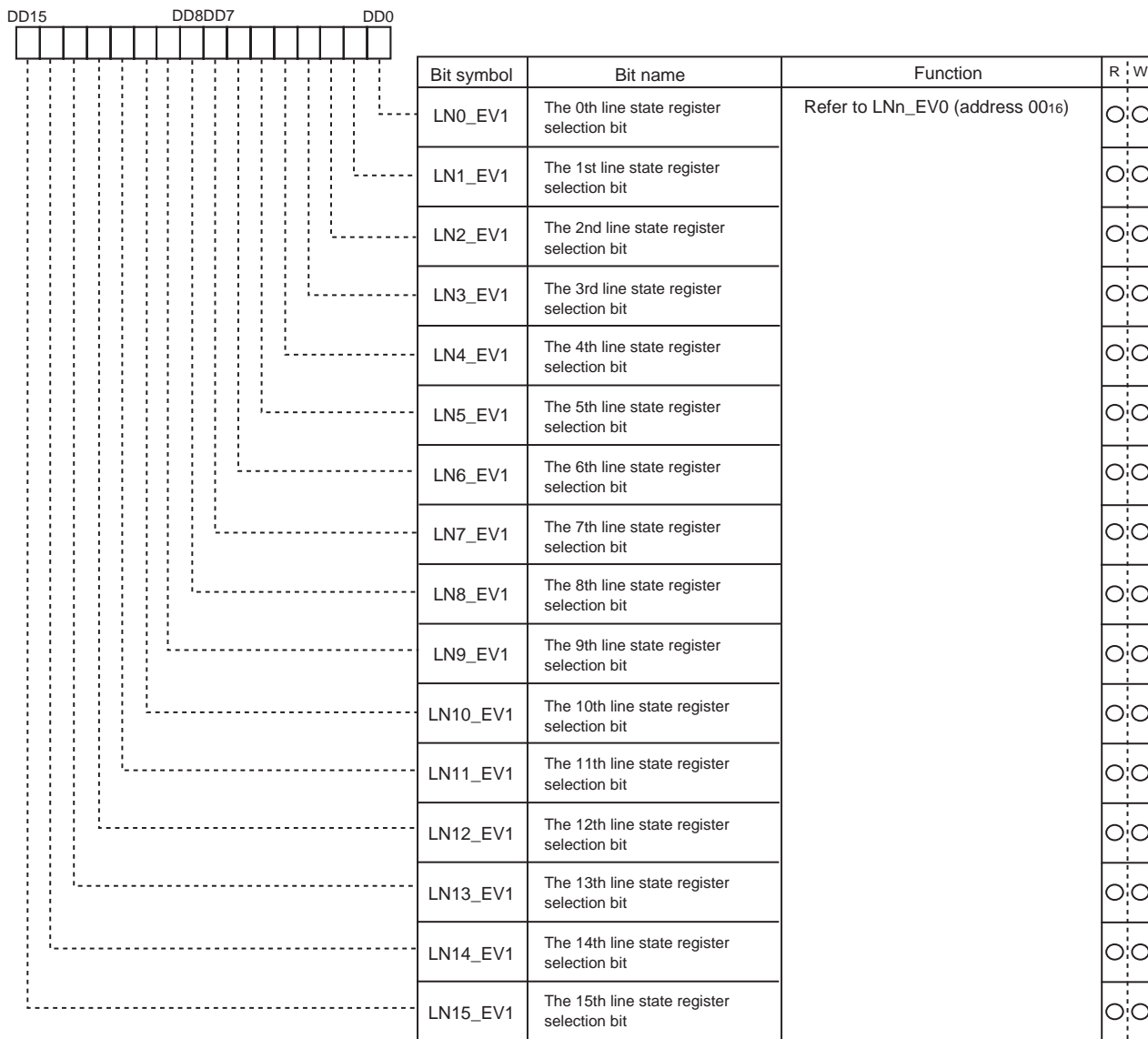
Notes 1. The n-th line: The number of lines after a slice start.
Please refer to the supplement (3) of 2.14.6 expansion register composition (P229) for details.

Notes 2. 06h to 0Ch address: State register 1
0Dh to 13h address: State register 2
14h to 1Ah address: State register 3

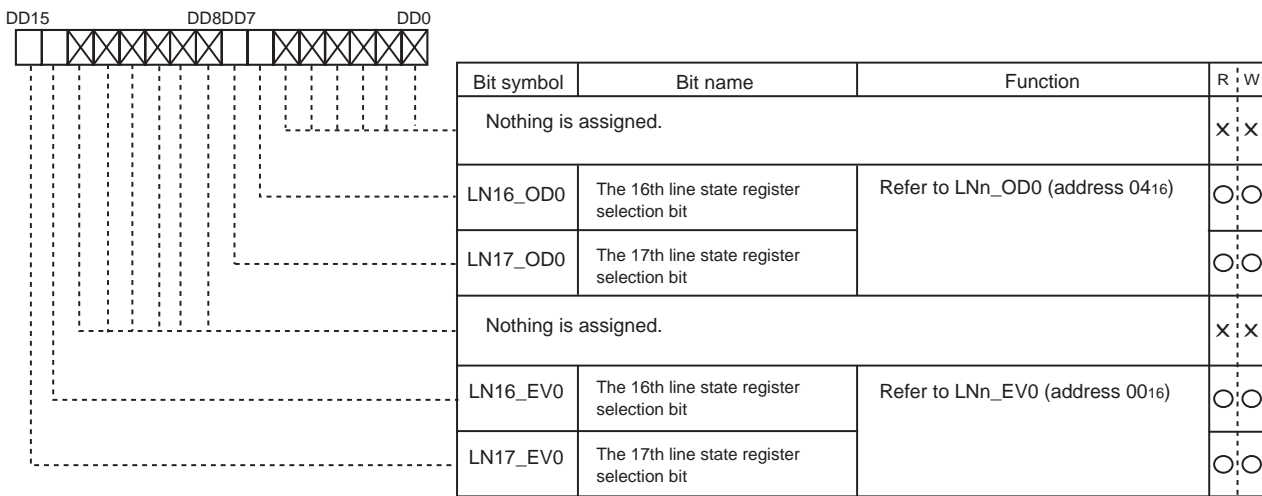
Notes 3. The example of a setting.



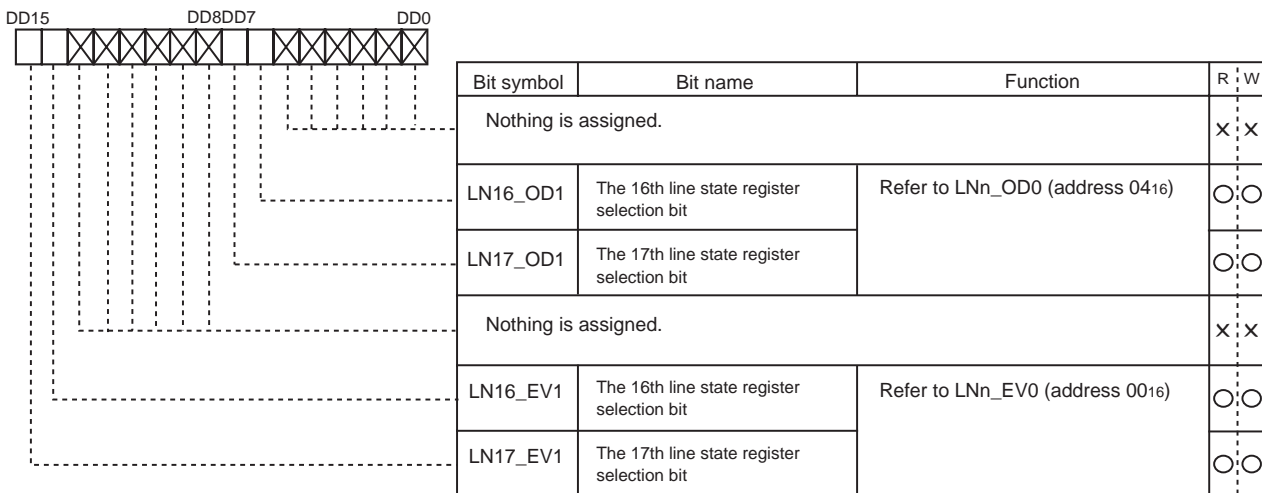
(2) Address 0116 (=DA5 to 0)



(3) Address 0216 (=DA5 to 0)



(4) Address 0316 (=DA5 to 0)



(5) Address 0416 (=DA5 to 0)

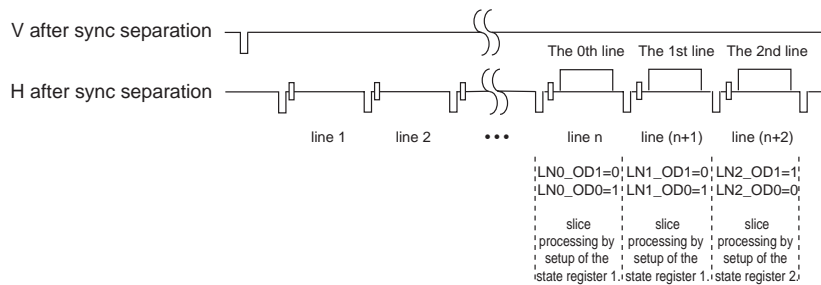
DD15	DD8DD7	DD0				
			Bit symbol	Bit name	Function	R : W
			LN0_OD0	The 0th line state register selection bit	As for the slicing method of the n-th line (Notes 1), it is chosen which set of the state register settings of the three sets (Notes 2) is used with the combination of LNn_OD0 (address 0416 and 0216, n = 0 to 17) and LNn_OD1 (addresses 0516 and 0316, n= 0 to 17.) Four kinds of following state registers can be chosen for every line. (Notes 3)	○ : ○
			LN1_OD0	The 1st line state register selection bit		○ : ○
			LN2_OD0	The 2nd line state register selection bit		○ : ○
			LN3_OD0	The 3rd line state register selection bit		○ : ○
			LN4_OD0	The 4th line state register selection bit		○ : ○
			LN5_OD0	The 5th line state register selection bit		○ : ○
			LN6_OD0	The 6th line state register selection bit		○ : ○
			LN7_OD0	The 7th line state register selection bit		○ : ○
			LN8_OD0	The 8th line state register selection bit		○ : ○
			LN9_OD0	The 9th line state register selection bit		○ : ○
			LN10_OD0	The 10th line state register selection bit		○ : ○
			LN11_OD0	The 11th line state register selection bit		○ : ○
			LN12_OD0	The 12th line state register selection bit		○ : ○
			LN13_OD0	The 13th line state register selection bit		○ : ○
			LN14_OD0	The 14th line state register selection bit		○ : ○
			LN15_OD0	The 15th line state register selection bit		○ : ○

LNn_EV1	LNn_EV0	State register(Notes 2)
0	0	Do not set up
0	1	State register 1
1	0	State register 2
1	1	State register 3

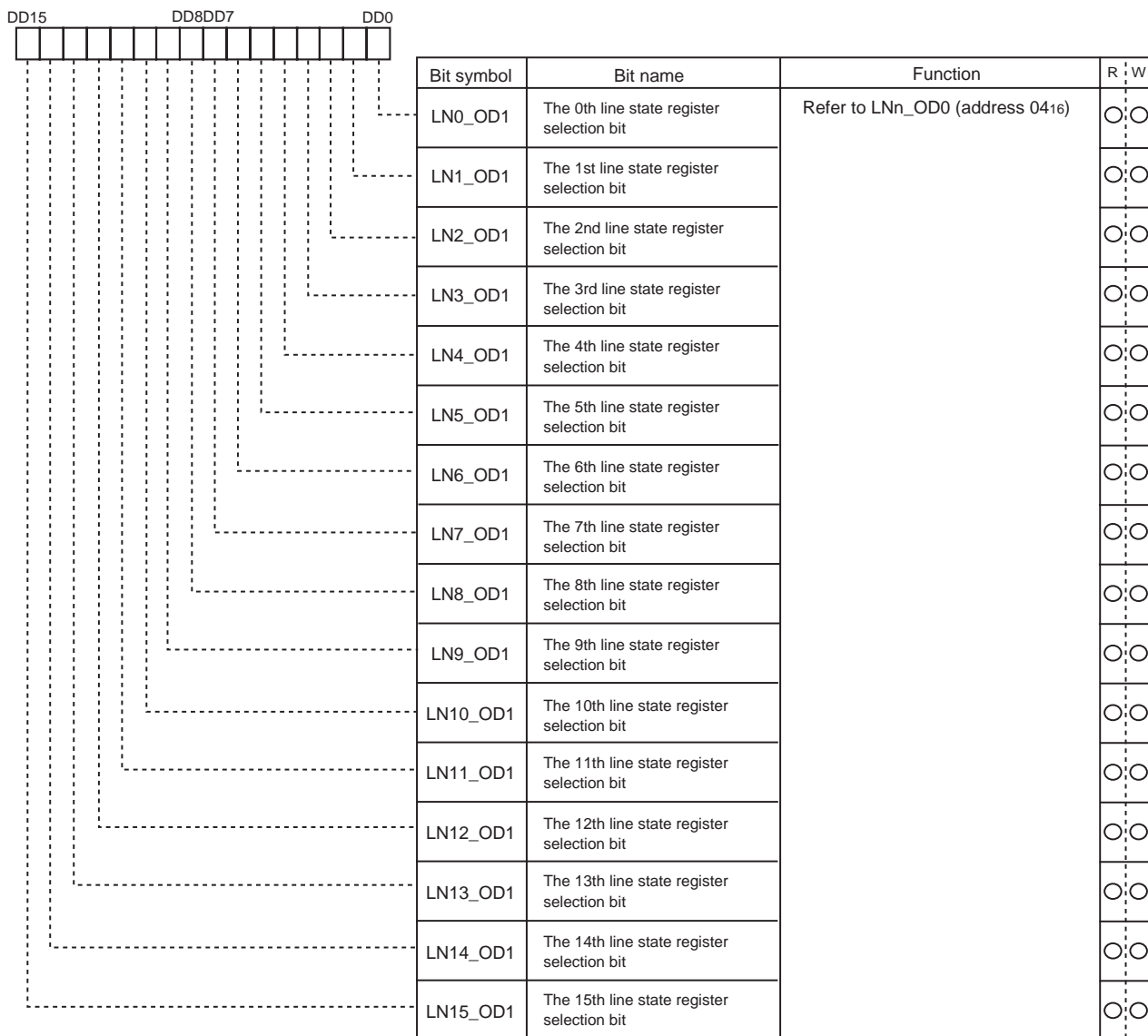
Notes 1. The n-th line: The number of lines after a slice start.
Please refer to the supplement (3) of 2.14.6 expansion register composition, and (P229) for details.

Notes 2. 06h to 0Ch address: State register 1
0Dh to 13h address: State register 2
14h to 1Ah address: State register 3

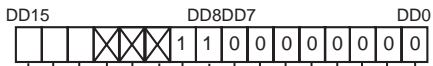
Notes 3. The example of a setting.



(6) Address 0516 (=DA5 to 0)



(7) Address 06₁₆, 0D₁₆, 14₁₆ (=DA₅ to 0)



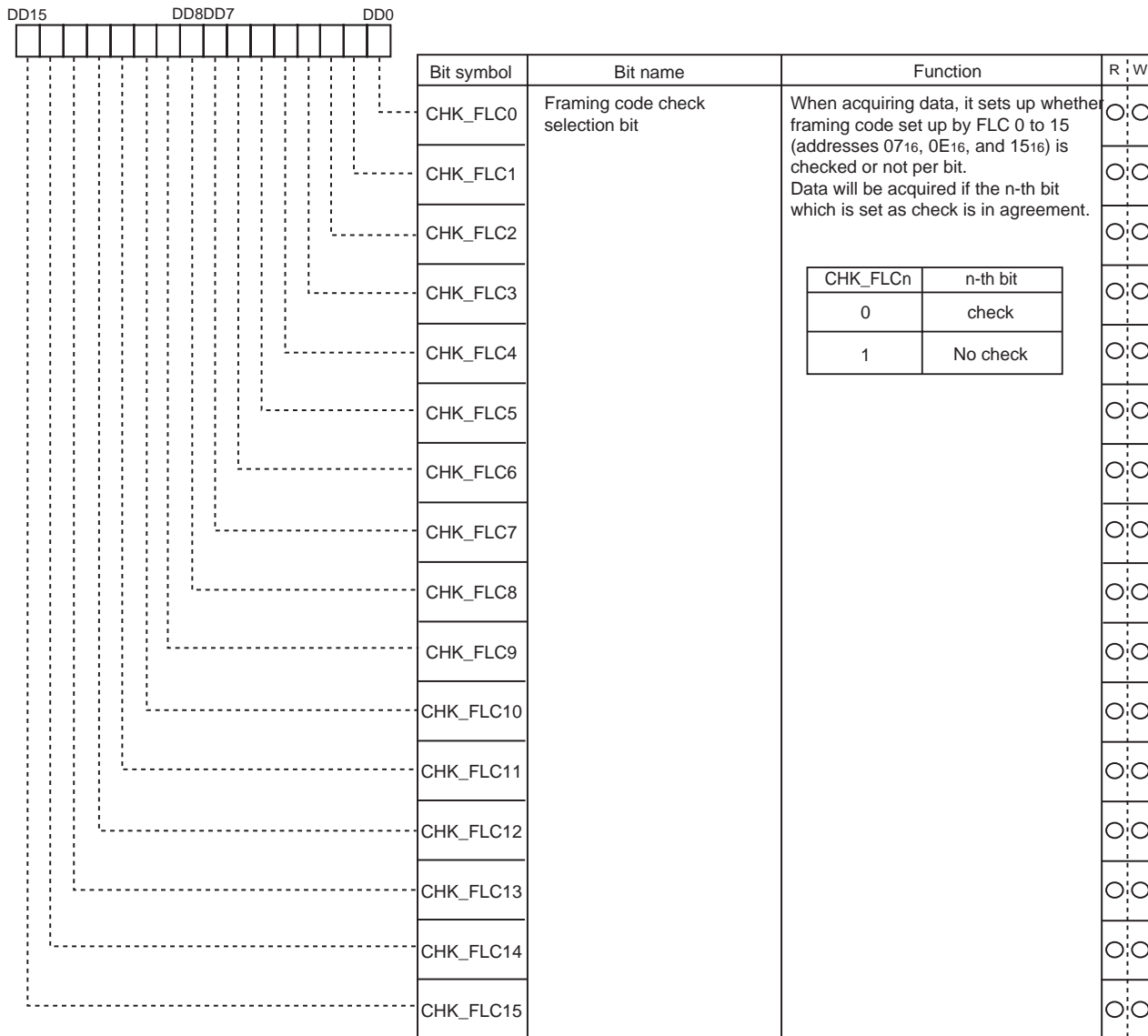
Bit symbol	Bit name	Function	R	W
Reserved bits		Must set to "0."	X	0
Reserved bits		Must set to "1."	X	0
Nothing is assigned.			X	X
SELVCO	The PLL selection bit for slice	0 PDC	0	0
		1 VPS		
DIVS0	The clock division bit for slice	DIVS1	0	0
		DIVS0		
DIVS1	The clock division bit for slice	0 0	0	0
		0 1		
		1 0		
		1 1		

(8) Address 07₁₆, 0E₁₆, 15₁₆ (=DA₅ to 0)



Bit symbol	Bit name	Function	R	W
FLC0	Framing code selection bit	Framing code is set up 16 bits are checked at maximum. However, the bit of CHK_FLCn (addresses 08 ₁₆ , 0F ₁₆ and 16 ₁₆) = "1" is not checked.	0	0
FLC1			0	0
FLC2			0	0
FLC3			0	0
FLC4			0	0
FLC5			0	0
FLC6			0	0
FLC7			0	0
FLC8			0	0
FLC9			0	0
FLC10			0	0
FLC11			0	0
FLC12			0	0
FLC13			0	0
FLC14			0	0
FLC15			0	0

(9) Address 08₁₆, 0F₁₆, 16₁₆ (=DA₅ to 0)



(10) Address 0916, 1016, 1716 (=DA5 to 0)



Bit symbol	Bit name	Function	R	W															
SEK10	Data slicer control bit 1	<table border="1"> <tr><td>SEK11</td><td>SEK10</td><td>N</td></tr> <tr><td>0</td><td>0</td><td>5</td></tr> <tr><td>0</td><td>1</td><td>4</td></tr> <tr><td>1</td><td>0</td><td>3</td></tr> <tr><td>1</td><td>1</td><td>With no differentiation (Note 1)</td></tr> </table>	SEK11	SEK10	N	0	0	5	0	1	4	1	0	3	1	1	With no differentiation (Note 1)	○	○
SEK11		SEK10	N																
0	0	5																	
0	1	4																	
1	0	3																	
1	1	With no differentiation (Note 1)																	
SEK11	N-times the digital value after SEK17.6.																		
SEK12	Data slicer control bit 2	<table border="1"> <tr><td>SEK13</td><td>SEK12</td><td>N</td></tr> <tr><td>0</td><td>0</td><td>4</td></tr> <tr><td>0</td><td>1</td><td>3</td></tr> <tr><td>1</td><td>0</td><td>1</td></tr> <tr><td>1</td><td>1</td><td>No differentiation</td></tr> </table>	SEK13	SEK12	N	0	0	4	0	1	3	1	0	1	1	1	No differentiation	○	○
SEK13		SEK12	N																
0	0	4																	
0	1	3																	
1	0	1																	
1	1	No differentiation																	
SEK13	It differentiates from the digitized data in front of N/8 cycles (clock run-in cycle) to the digital value after SEK10 and 1.																		
SEK14	Data slicer control bit 3	<table border="1"> <tr><td>SEK15</td><td>SEK14</td><td>N</td></tr> <tr><td>0</td><td>0</td><td>4</td></tr> <tr><td>0</td><td>1</td><td>3</td></tr> <tr><td>1</td><td>0</td><td>1</td></tr> <tr><td>1</td><td>1</td><td>No differentiation</td></tr> </table>	SEK15	SEK14	N	0	0	4	0	1	3	1	0	1	1	1	No differentiation	○	○
SEK15		SEK14	N																
0	0	4																	
0	1	3																	
1	0	1																	
1	1	No differentiation																	
SEK15	It differentiates from the digitized data in front of N/8 cycles (clock run-in cycle) to the digital value after SEK13 and 2.																		
SEK16	Data slicer control bit 4	<table border="1"> <tr><td>SEK17</td><td>SEK16</td><td>N</td></tr> <tr><td>0</td><td>0</td><td>4</td></tr> <tr><td>0</td><td>1</td><td>3</td></tr> <tr><td>1</td><td>0</td><td>5</td></tr> <tr><td>1</td><td>1</td><td>1</td></tr> </table>	SEK17	SEK16	N	0	0	4	0	1	3	1	0	5	1	1	1	○	○
SEK17		SEK16	N																
0	0	4																	
0	1	3																	
1	0	5																	
1	1	1																	
SEK17	A digital value is averaged after AD for N clock.																		
Nothing is assigned.			X	X															
Reserved bit		Must set to "1."	X	○															
SLSLVL	Slice level measurement period selection bit	<table border="1"> <tr><td>0</td><td>2 cycles of Clock run-in</td></tr> <tr><td>1</td><td>4 cycles of Clock run-in</td></tr> </table>	0	2 cycles of Clock run-in	1	4 cycles of Clock run-in	X	○											
0	2 cycles of Clock run-in																		
1	4 cycles of Clock run-in																		
BIFON	Data format selection bit	<table border="1"> <tr><td>0</td><td>Non Return Zero</td></tr> <tr><td>1</td><td>Bi-phase type</td></tr> </table>	0	Non Return Zero	1	Bi-phase type	○	○											
0	Non Return Zero																		
1	Bi-phase type																		
Reserved bit		Must set to "0."	X	○															
Reserved bits		Must set to "1."	X	○															
Reserved bit		Must set to "0."	X	○															

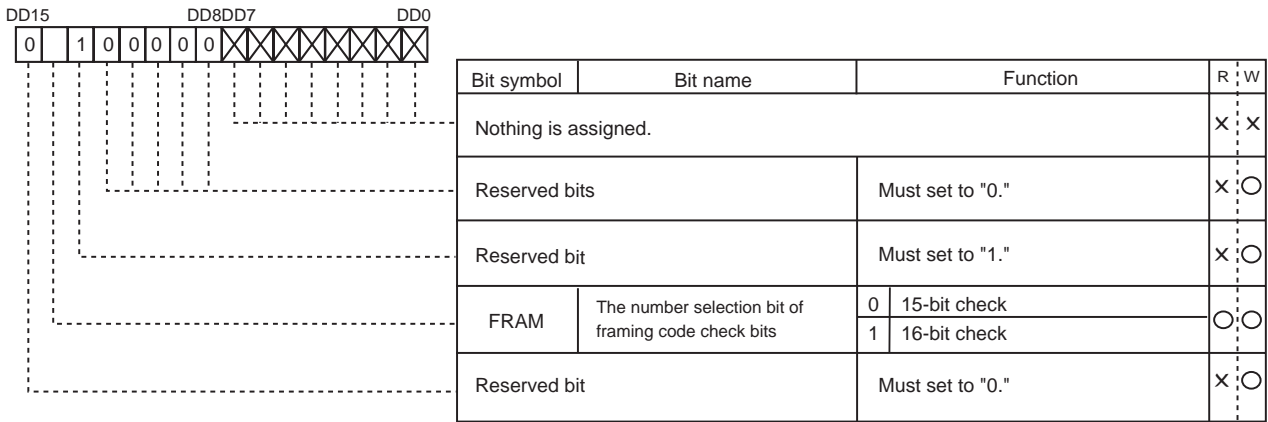
Note 1. Multiplying factor set up by SEK16 and SEK17.
 However, do not set it with (SEK17, SEK16) = (1, 1).

(11) Address 0A16, 1116, 1816 (=DA5 to 0)

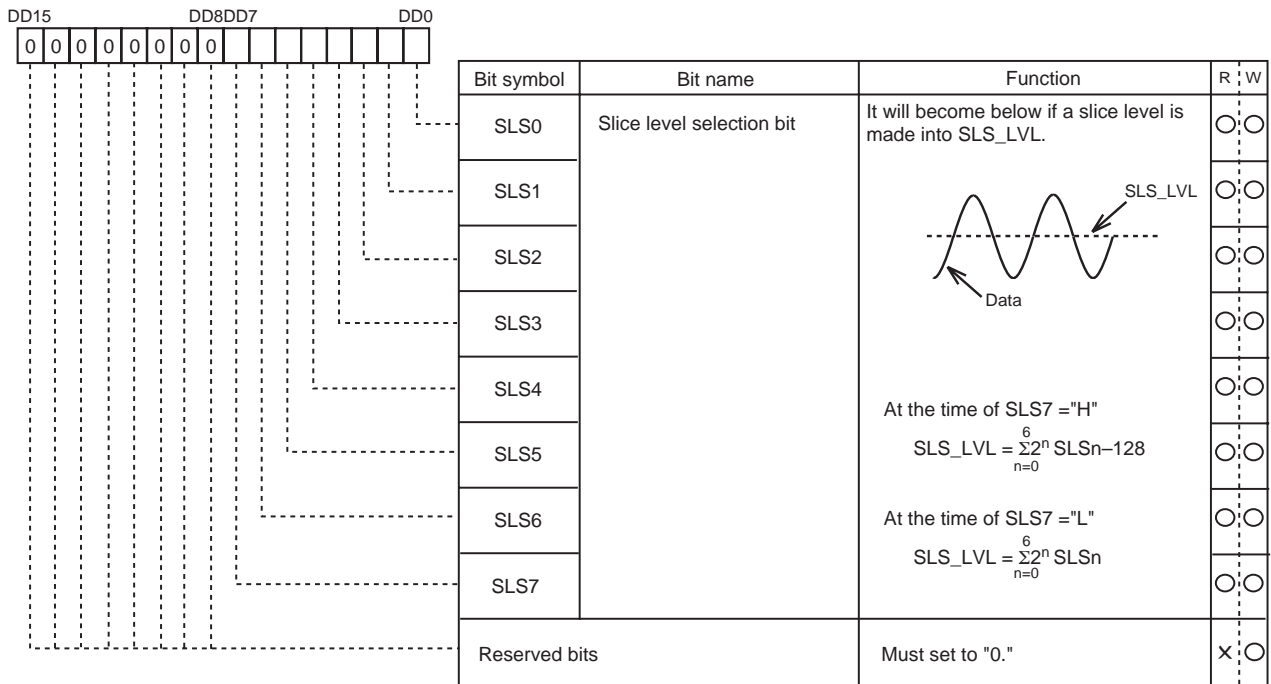


Bit symbol	Bit name	Function	R	W															
SLS_HP0	Slice check start position selection bit	It will become below if data slice start position is made into SLS_HS. $SLS_HS = T2 \times \sum_{n=0}^7 SLS_HPn$ T2 : Clock run-in cycle /2 The position where framing code begins to be checked is set up. Setup in a 1-bit unit is possible.	○	○															
SLS_HP1			○	○															
SLS_HP2			○	○															
SLS_HP3			○	○															
SLS_HP4			○	○															
SLS_HP5			○	○															
SLS_HP6			○	○															
SLS_HP7			○	○															
GET_HP0	Phase fine-tuning bit	Slice data 0/1 judging clock is tuned finely.	○	○															
GET_HP1			○	○															
Reserved bit		Must set to "1."	x	○															
Reserved bit		Must set to "0."	x	○															
GETPEEK0	Peak detection period selection bit 0	<table border="1"> <thead> <tr> <th>GETPEEK1</th> <th>GETPEEK0</th> <th>Clock run-in period</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>2</td> </tr> <tr> <td>0</td> <td>1</td> <td>3</td> </tr> <tr> <td>1</td> <td>0</td> <td>6</td> </tr> <tr> <td>1</td> <td>1</td> <td>8</td> </tr> </tbody> </table>	GETPEEK1	GETPEEK0	Clock run-in period	0	0	2	0	1	3	1	0	6	1	1	8	○	○
GETPEEK1	GETPEEK0	Clock run-in period																	
0	0	2																	
0	1	3																	
1	0	6																	
1	1	8																	
GETPEEK1	Peak detection period selection bit 1		○	○															
GETPEEK2	Peak detection period selection bit 2	0	With clock compensation		○														
		1	With no clock compensation																
GETPEEK3	Peak detection period selection bit 3	0	Only a mountain is detected.		○														
		1	A mountain and a valley are detected.																

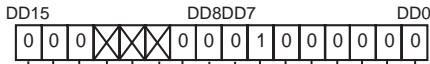
(12) Address 0B₁₆, 12₁₆, 19₁₆ (=DA₅ to 0)



(13) Address 0C₁₆, 13₁₆, 1A₁₆ (=DA₅ to 0)

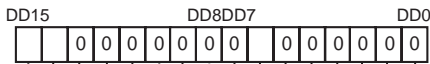


(14) Address 1B₁₆ (=DA5 to 0)



Bit symbol	Bit name	Function	R	W
Reserved bits		Must set to "0."	X	○
Reserved bits		Must set to "1."	X	○
Reserved bits		Must set to "0."	X	○
Nothing is assigned.			X	X
Reserved bits		Must set to "0."	X	○

(15) Address 1C₁₆ (=DA5 to 0)



Bit symbol	Bit name	Function	R	W
Reserved bits		Must set to "0."	X	○
SELSEP0	H•V input selection bit	0 Separated H•V is used. 1 H•V of an external input is used.	○	○
Reserved bits		Must set to "0."	X	○
VREF1	Horizontal synchronous signal slice level source select bit	0 Input to SVREF pin externally. 1 Generated internally (SVREF input is unnecessary).	○	○
ADSTART	A/D conversion completion bit	0 Conversion completion 1 Under conversion	○	○

(16) Address 1D₁₆ (=DA5 to 0)



Bit symbol	Bit name	Function	R	W	
Nothing is assigned.			X	X	
XTAL_VCO	Synchronous clock oscillation selection bit	0	Clock for insides stop		
		1	Clock for insides oscillation		
Reserved bit			X	0	
PDC_VCO_ON	PDC clock oscillation selection bit	0	PDC clock stop		
		1	PDC clock oscillation		
PDC_VCO_R0	PDC clock oscillation change bit	PDC_VCO_R1		X	
PDC_VCO_R1		0	0		Select PDC clock
		1	0		Select EPG-J clock
	0	1	Do not set up		
	1	1	Do not set up		
VPS_VCO_ON	VPS clock oscillation selection bit	0	VPS clock stop		
		1	VPS clock oscillation		
Reserved bits			X	0	
Nothing is assigned.			X	X	

(17) Address 1E₁₆ (=DA5 to 0)



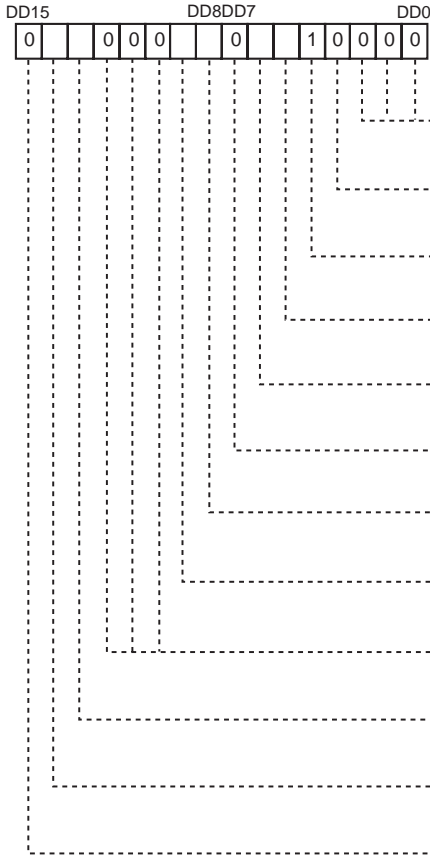
Bit symbol	Bit name	Function	R	W
Reserved bits			X	0
Nothing is assigned.			X	X

(18) Address 1F₁₆ (=DA5 to 0)



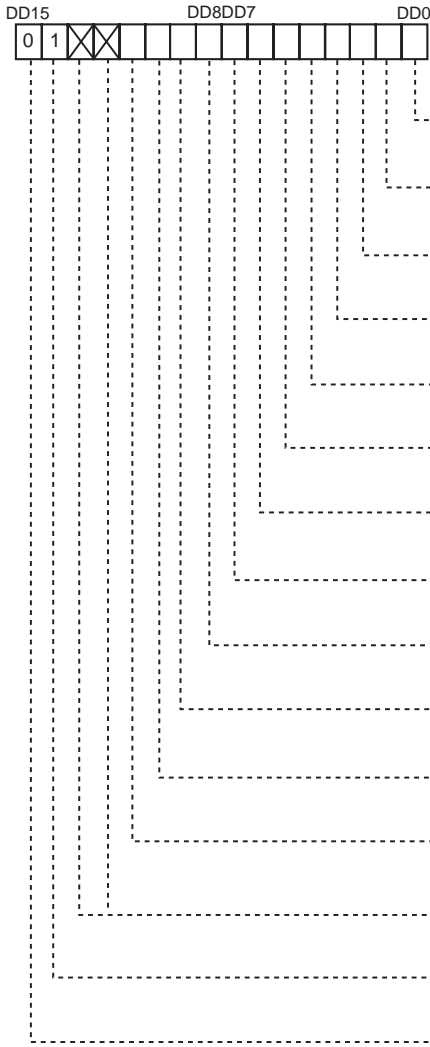
Bit symbol	Bit name	Function	R	W
Nothing is assigned.			X	X
FLD1V	Field state flag	0	Even field	
		1	Odd field	
Reserved bits			X	0
MACRO_ON	Synchronized signal seaech flag	0	normal	
		1	unusual	
Nothing is assigned.			X	X

(19) Address 2016 (=DA5 to 0)



Bit symbol	Bit name	Function	R	W															
Reserved bits		Must set to "0."	X	0															
SELXT0	Synchronous (fsc) clock phase adjustment control bit	Set up (SELXT1, SELXT0) = (1, 0)	0	0															
SELXT1			0	0															
SELXT2	Synchronous (fsc) clock division control bit (Note1)	0 Divided by 32 1 Setup divided value (refer to address 2116 DIV_FSC)	0	0															
			1	0															
SEPV0	Vertical synchronous separation standard selection bit	0 Detected in L period of 15μs/22μs. 1 Detected in L period of 22μs.	0	0															
			1	0															
Reserved bit		Must set to "0."	X	0															
NORMAL	Framing code check control bit	0 Check (Data is acquired if Framing code is in agreement). 1 No check (All data is acquired).	0	0															
			1	0															
LEVELA	Synchronous signal slice potential generating control bit	0 Synchronous signal slice potential generating circuit OFF 1 Synchronous signal slice potential generating circuit ON	0	0															
			1	0															
Reserved bits		Must set to "0."	X	0															
NXP	Broadcast method selection bit	<table border="1"> <thead> <tr> <th>NXP</th> <th>MPAL</th> <th>Broadcast method</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>NTSC</td> </tr> <tr> <td>0</td> <td>1</td> <td>M-PAL</td> </tr> <tr> <td>1</td> <td>0</td> <td>PAL</td> </tr> <tr> <td>1</td> <td>1</td> <td>Do not set up</td> </tr> </tbody> </table>	NXP	MPAL	Broadcast method	0	0	NTSC	0	1	M-PAL	1	0	PAL	1	1	Do not set up	0	0
			NXP	MPAL	Broadcast method														
			0	0	NTSC														
			0	1	M-PAL														
1	0	PAL																	
1	1	Do not set up																	
MPAL			0	0															
			1	0															
			1	0															
Reserved bit		Must set to "0."	X	0															

(20) Address 2116 (=DA5 to 0)



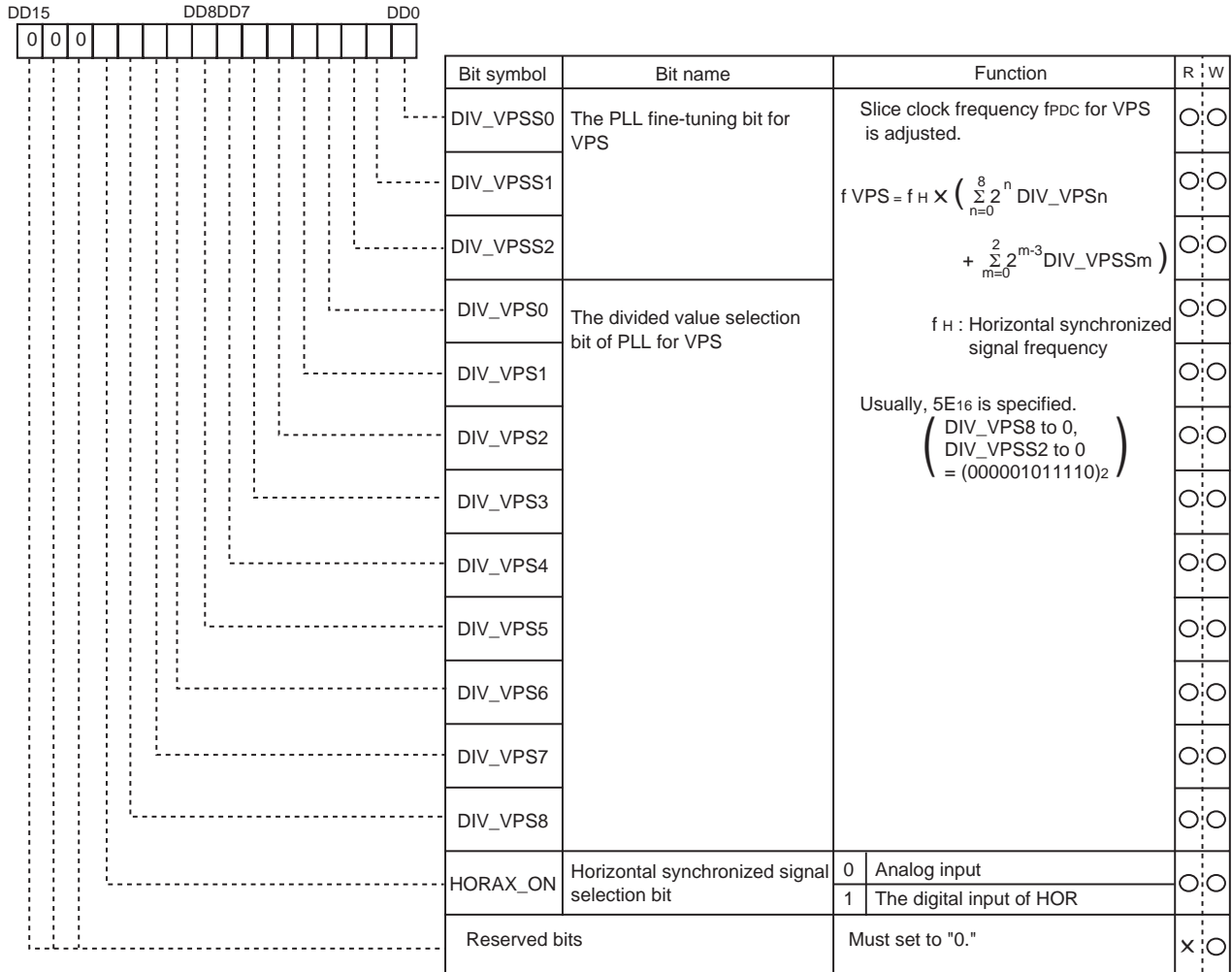
Bit symbol	Bit name	Function	R	W
DIV_FSC0	The divided value selection bit of PLL for fsc	The divided clock frequency fsc is adjusted to the phase comparison with a main clock. $f_{fsc} = f_{P1} \times \sum_{n=0}^7 2^n \text{DIV_FSCn}$ f P1 : Divided main clock frequency	○	○
DIV_FSC1			○	○
DIV_FSC2			○	○
DIV_FSC3			○	○
DIV_FSC4			○	○
DIV_FSC5			○	○
DIV_FSC6			○	○
DIV_FSC7			○	○
DIVF_CK0	The main clock deviation value selection bit for phase comparison	Using for the phase comparison with fsc PLL, the divided main clock frequency fP1 is adjusted. $f(\text{BCLK}) = f_{P1} \times \sum_{n=0}^3 2^n \text{DIVF_CKn}$ Set the following DIVF_CK3 to 0 = (0101) ₂ (at f(BCLK) = 10MHz) DIVF_CK3 to 0 = (1000) ₂ (at f(BCLK) = 16MHz)	○	○
DIVF_CK1			○	○
DIVF_CK2			○	○
DIVF_CK3			○	○
Nothing is assigned.			X	X
Reserved bit		Must set to "1."	X	○
Reserved bit		Must set to "0."	X	○

(21) Address 2216 (=DA5 to 0)

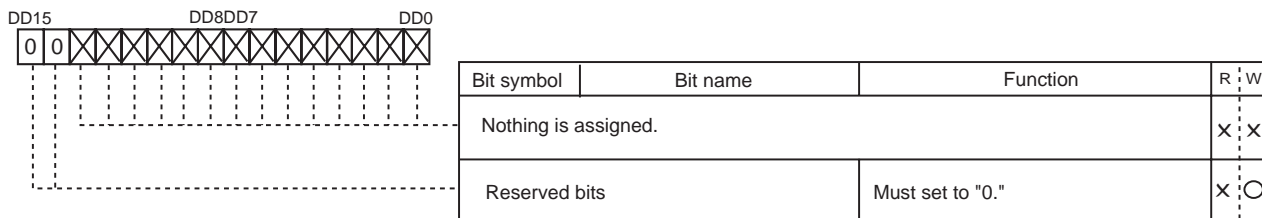


Bit symbol	Bit name	Function	R	W
DIV_PDCS0	The PLL fine-tuning bit for PDC	Slice clock frequency f_{PDC} for PDC is adjusted. $f_{PDC} = f_H \times \left(\sum_{n=0}^8 2^n \text{DIV_PDCn} + \sum_{m=0}^2 2^{m-3} \text{DIV_PDCSm} \right)$	○	○
DIV_PDCS1			○	○
DIV_PDCS2			○	○
DIV_PDC0	The divided value selection bit of PLL for PDC f_H : Horizontal synchronized signal frequency When select synchronization with main clock, set these bits as follows. • When teletext (PDC) data is acquired (DIV_PDC8 to 0, DIV_PDCS2 to 0) = (0000010001) ₂ • When EPG-J is acquired (DIV_PDC8 to 0, DIV_PDCS2 to 0) = (00000101000) ₂	○	○	
DIV_PDC1		○	○	
DIV_PDC2		○	○	
DIV_PDC3		○	○	
DIV_PDC4		○	○	
DIV_PDC5		○	○	
DIV_PDC6		○	○	
DIV_PDC7		○	○	
DIV_PDC8		○	○	
Nothing is assigned.			X	X
HM84SEL	8/4 humming polarity selection bit	0	Normal	
		1	The 4-bit data of 8/4 humming is reversal-outputted.	

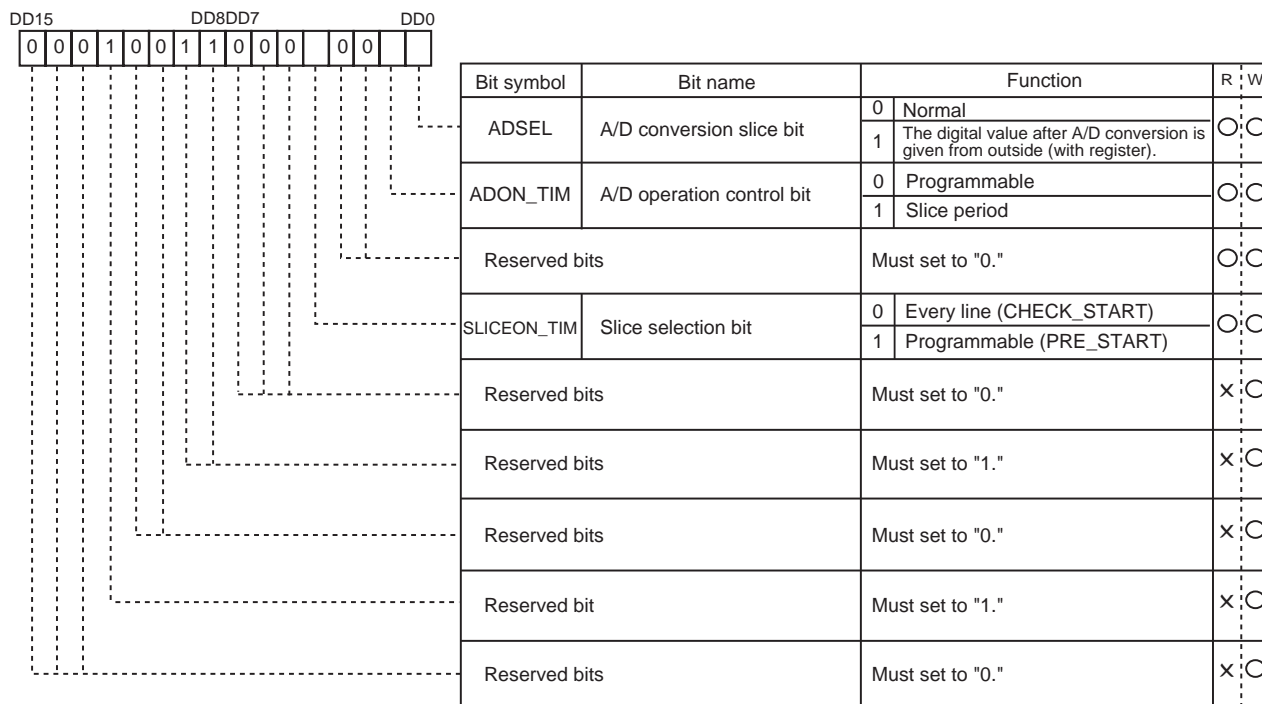
(22) Address 2316 (=DA5 to 0)



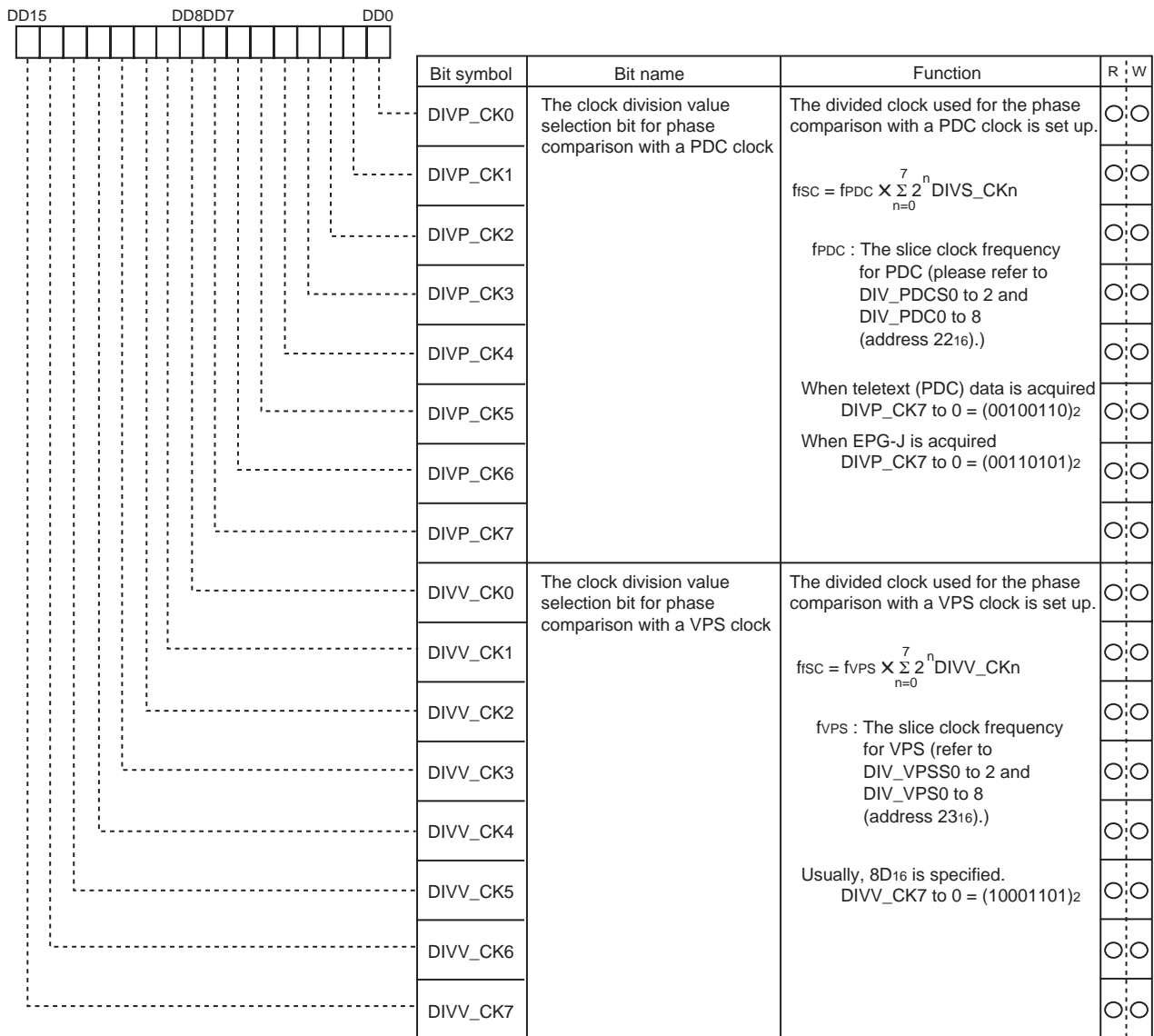
(23) Address 2416 (=DA5 to 0)



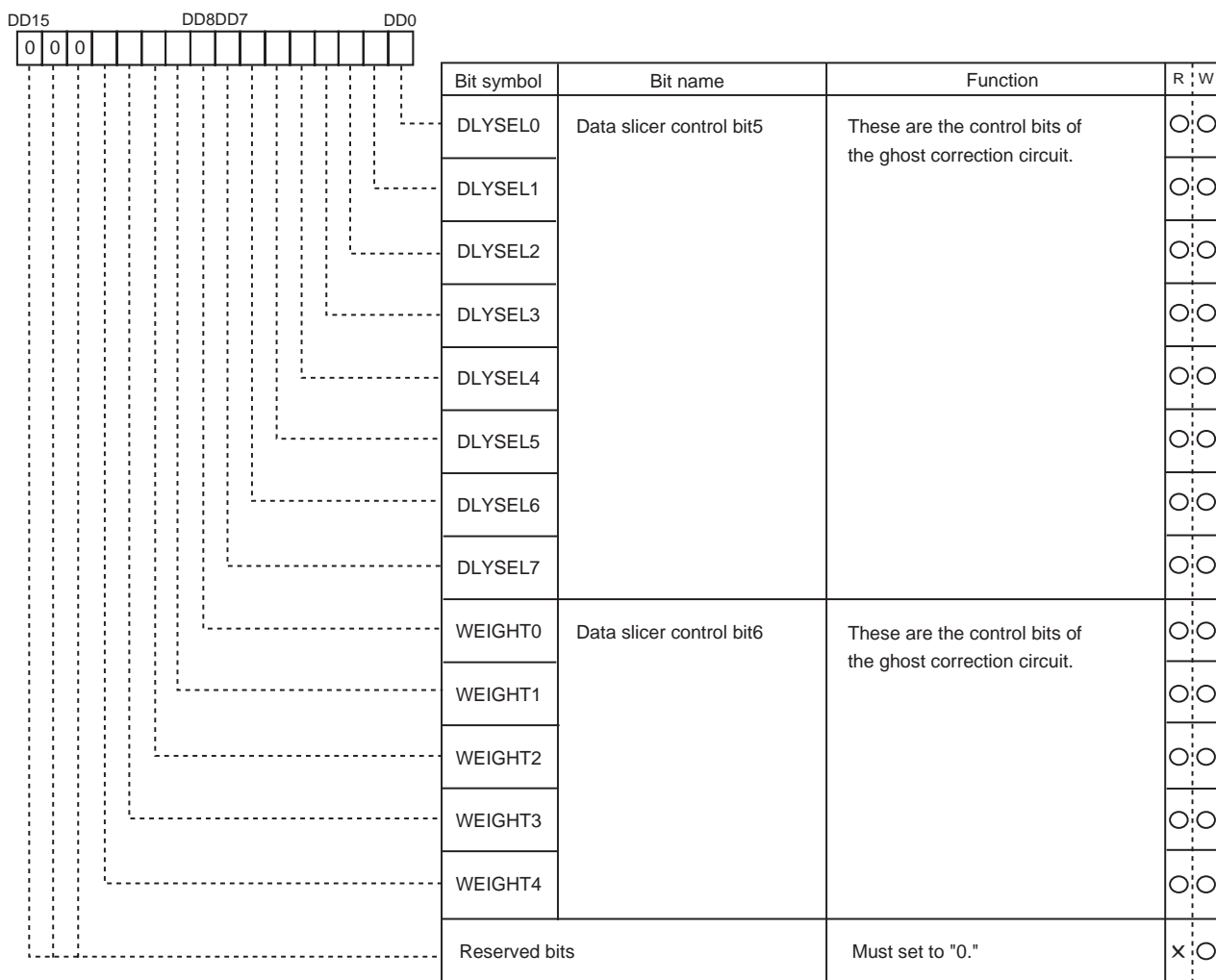
(24) Address 2516 (=DA5 to 0)



(25) Address 26₁₆ (=DA5 to 0)



(26) Address 27₁₆ (=DA5 to 0)

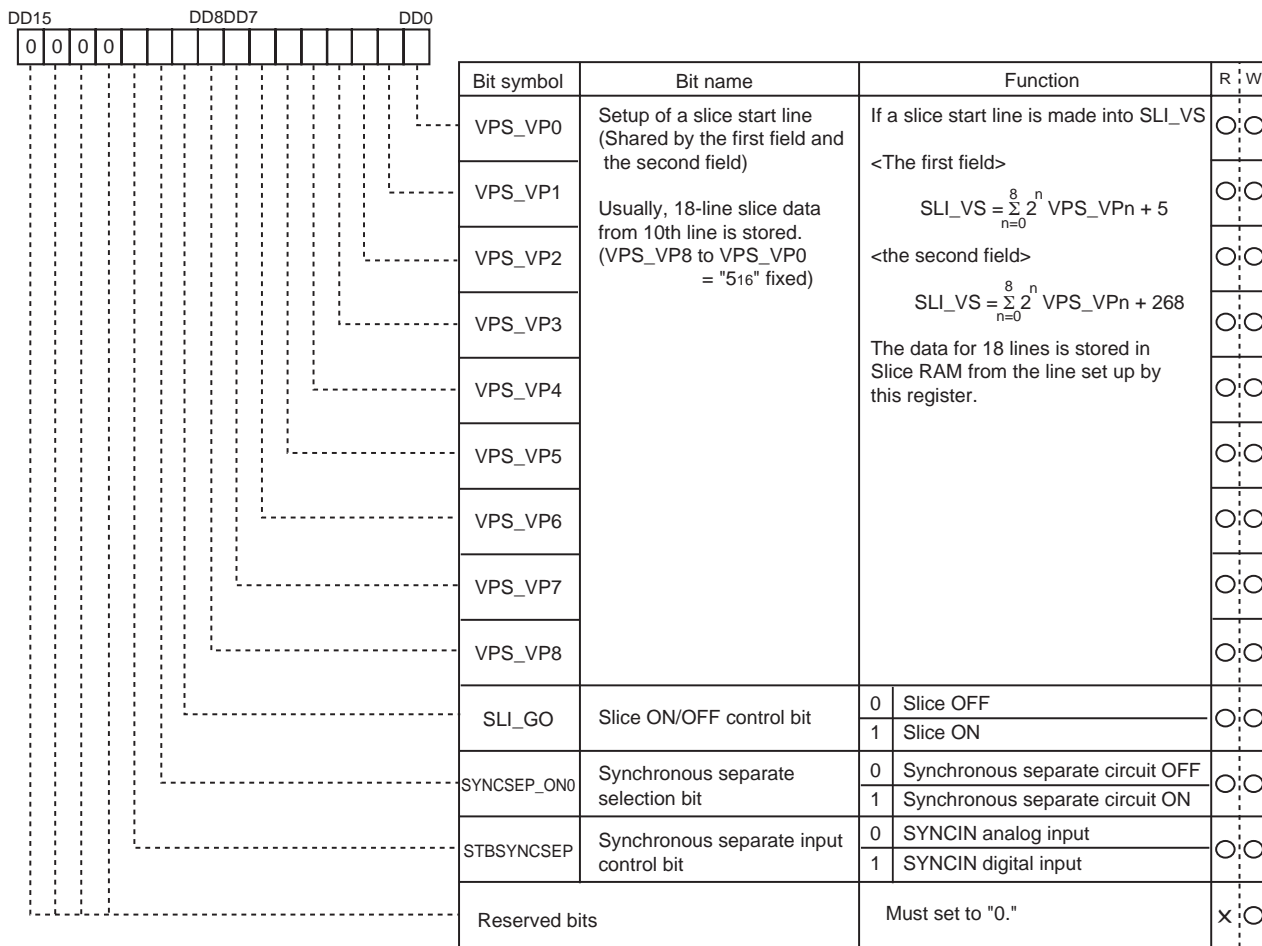


(27) Address 28₁₆ (=DA5 to 0)

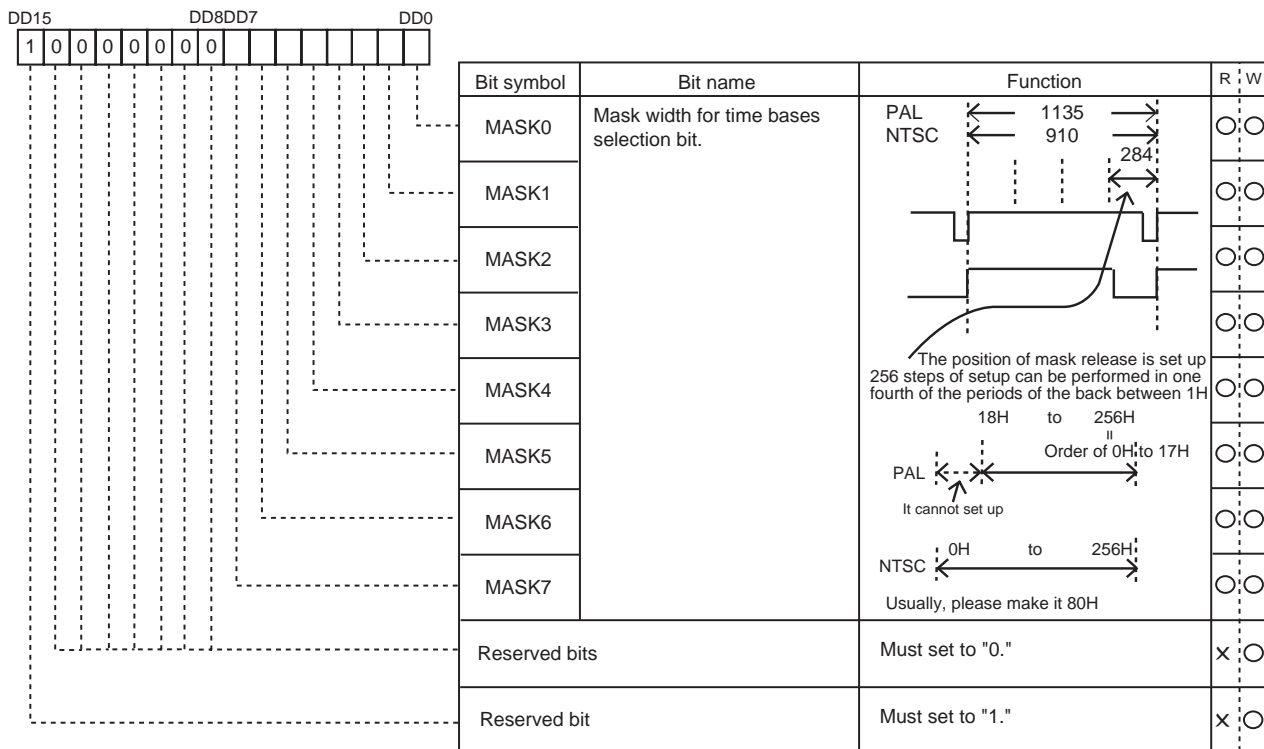


Bit symbol	Bit name	Function	R	W																																				
ADLAT	Data acquisition selection bit	0 Acquisition of slice data	○	○																																				
		1 Acquisition of A/D data																																						
START	Slice data selection bit	Buffer memory 0: 1: Offset to a start (8 bits) Slice level (8 bits)	○	○																																				
Reserved bit		Must set to "0."	x	○																																				
6BITOFF	A/D lower bit selection bit	0 Normal	○	○																																				
		1 Stop by 6th bit of A/D																																						
Reserved bits		Must set to "0."	x	○																																				
SYNLVL0	Synchronous signal slice level control bit	<table border="1"> <thead> <tr> <th>SYNLVL2</th> <th>SYNLVL1</th> <th>SYNLVL0</th> <th>Slice level</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>approx.1.10V±0.10V</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>approx.1.15V±0.10V</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>approx.1.20V±0.10V</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>approx.1.25V±0.10V</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>approx.1.30V±0.10V</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>approx.1.35V±0.10V</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>approx.1.40V±0.10V</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>approx.1.45V±0.10V</td> </tr> </tbody> </table>	SYNLVL2	SYNLVL1	SYNLVL0	Slice level	0	0	0	approx.1.10V±0.10V	0	0	1	approx.1.15V±0.10V	0	1	0	approx.1.20V±0.10V	0	1	1	approx.1.25V±0.10V	1	0	0	approx.1.30V±0.10V	1	0	1	approx.1.35V±0.10V	1	1	0	approx.1.40V±0.10V	1	1	1	approx.1.45V±0.10V	○	○
SYNLVL2		SYNLVL1	SYNLVL0	Slice level																																				
0		0	0	approx.1.10V±0.10V																																				
0		0	1	approx.1.15V±0.10V																																				
0	1	0	approx.1.20V±0.10V																																					
0	1	1	approx.1.25V±0.10V																																					
1	0	0	approx.1.30V±0.10V																																					
1	0	1	approx.1.35V±0.10V																																					
1	1	0	approx.1.40V±0.10V																																					
1	1	1	approx.1.45V±0.10V																																					
SYNLVL1																																								
SYNLVL2																																								
ADON	Data slicer control bit	0 Data slicer OFF. (The amplifier for slicer is also turned off). 1 Data slicer ON (see INTAD and the INTDA about the amplifier for slicer)	○	○																																				
INTAD	The amplifier control bit for data slicers	0 Always data slicer ON. 1 On 3 to 23 lines and 315 to 335 line amplifier ON. On other line amplifier OFF	○	○																																				
INTDA	The rudder resistance control bit for data slicers	0 Always ladder resistance for data slicer ON. 1 On 3 to 23 lines and 315 to 335 line Ladder resistance ON. On other line Ladder resistance OFF	○	○																																				
Reserved bits		Must set to "0."	x	○																																				

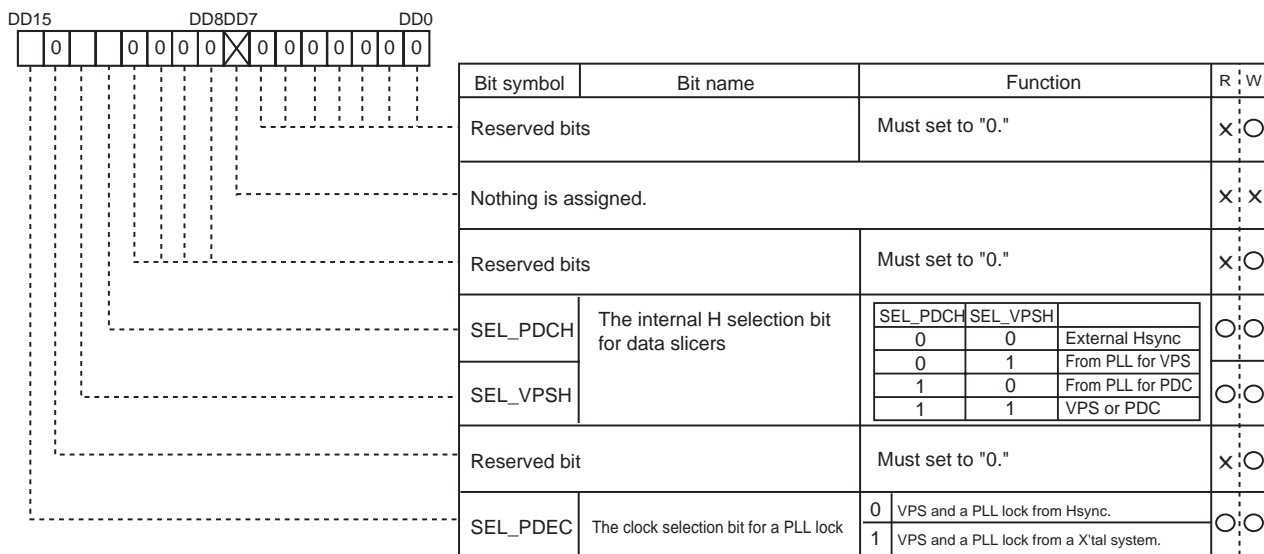
(28) Address 2916 (=DA5 to 0)



(29) Address 2A₁₆ (=DA5 to 0)



(30) Address 2B₁₆ (=DA5 to 0)



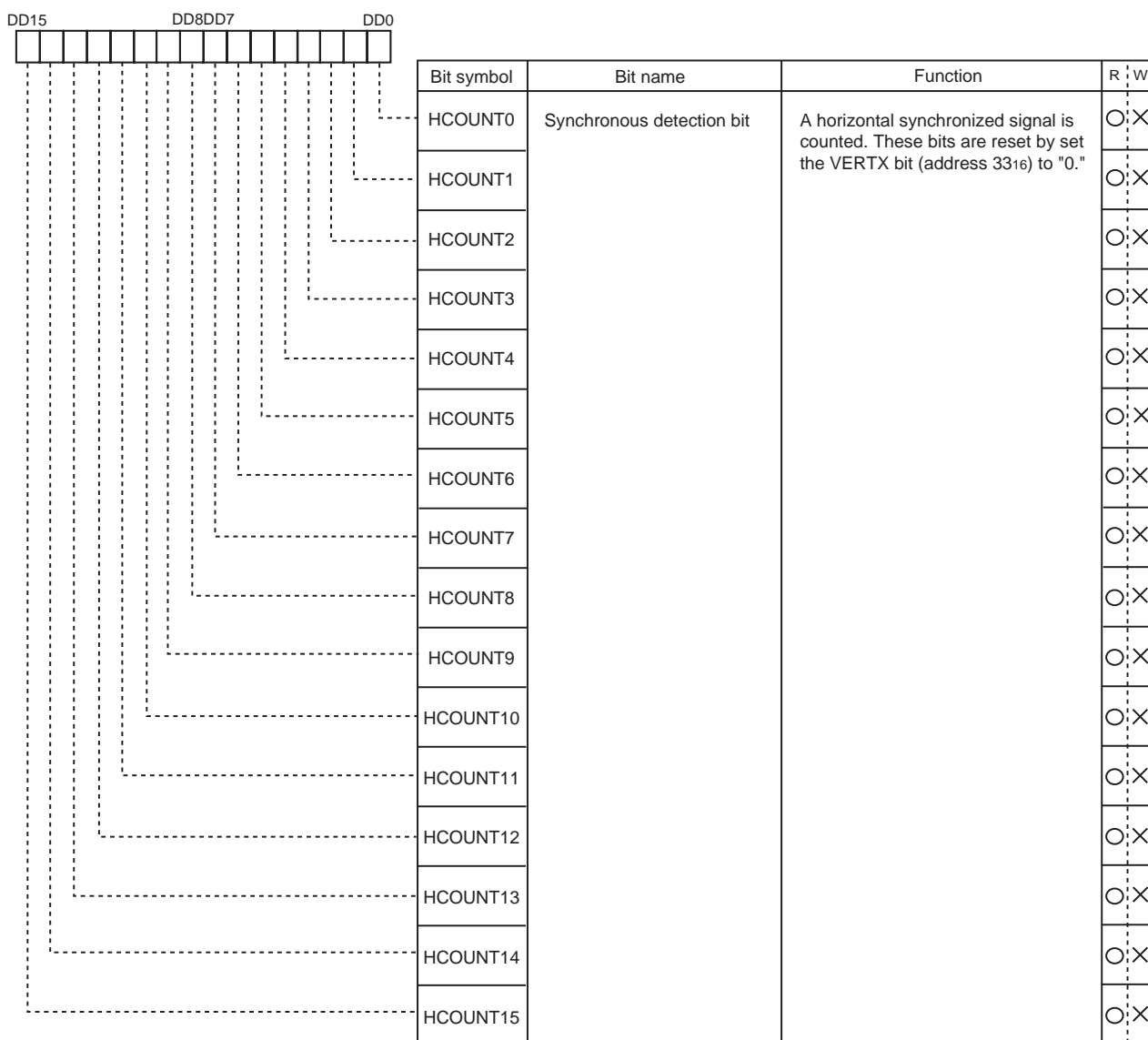
(31) Address 2C₁₆ (=DA5 to 0)

Bit symbol	Bit name	Function	R/W
DD15	1	Slice A/D ON period selection bit 	○/○
	0		○/○
	0		○/○
	0		○/○
	0		○/○
	0		○/○
	0		○/○
	0		○/○
	0		○/○
	0		○/○
		Reserved bits	X/○
		Reserved bit	X/○

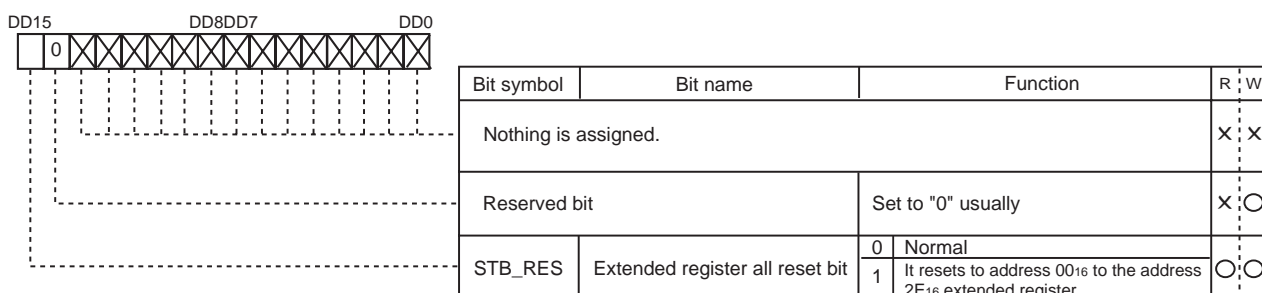
(32) Address 2D₁₆ (=DA5 to 0)

Bit symbol	Bit name	Function	R/W
DD15	0	Slice-ON period selection bit Refer to PLSPOS0 to 8 (Address 2C ₁₆)	○/○
	0		○/○
	0		○/○
	0		○/○
	0		○/○
	0		○/○
	0		○/○
	0		○/○
	0		○/○
		Reserved bits	X/○

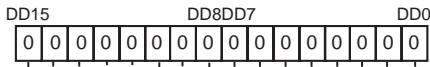
(33) Address 2E₁₆ (=DA5 to 0)



(34) Address 2F₁₆ (=DA5 to 0)

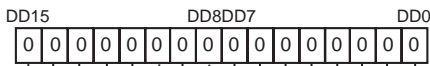


(35) Address 30₁₆ (=DA5 to 0)



Bit symbol	Bit name	Function	R	W
	Reserved bit	Set to "0" usually	x	0

(36) Address 31₁₆ (=DA5 to 0)



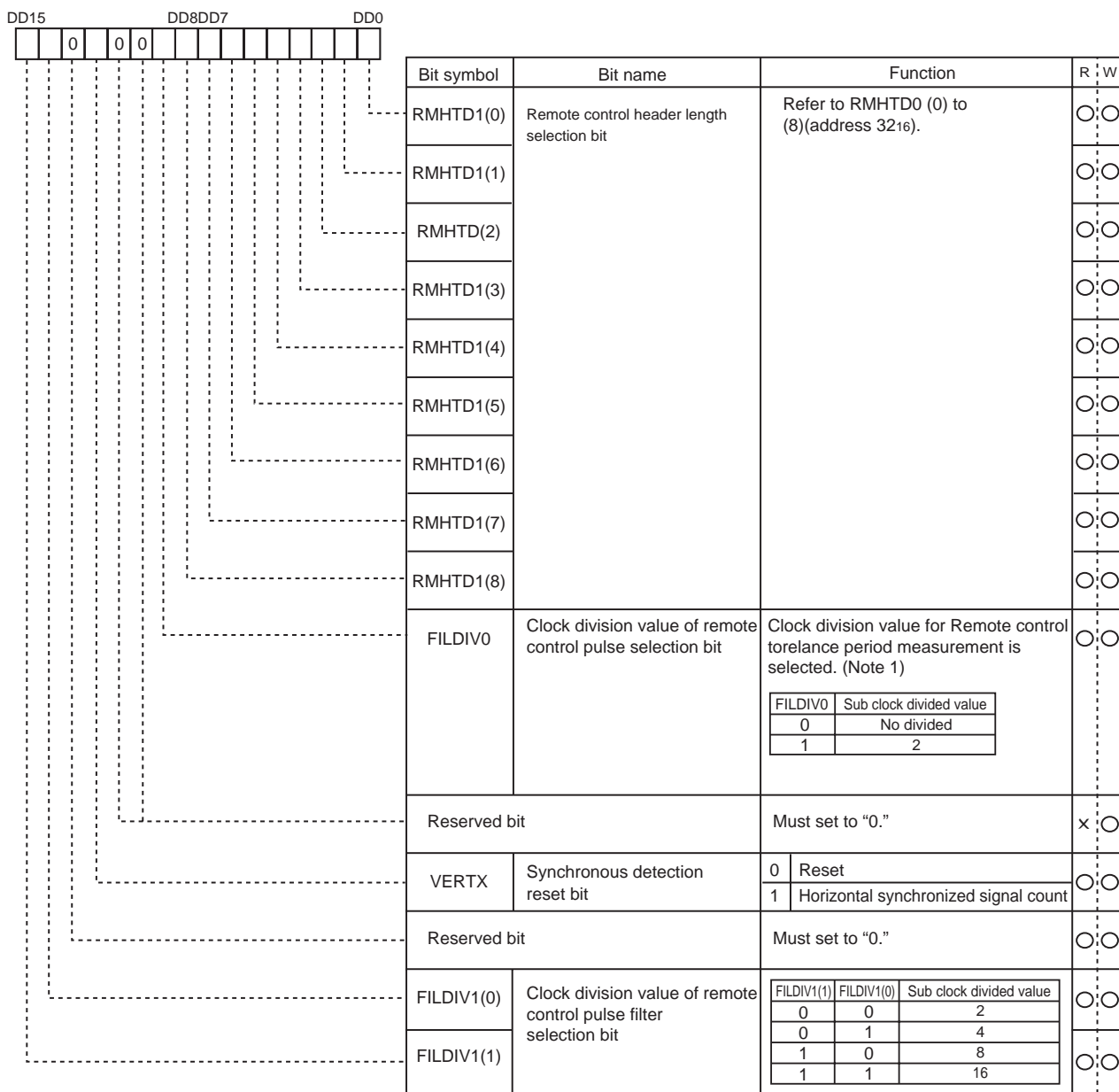
Bit symbol	Bit name	Function	R	W
	Reserved bit	Set to "0" usually	x	0

(37) Address 32₁₆ (=DA5 to 0)



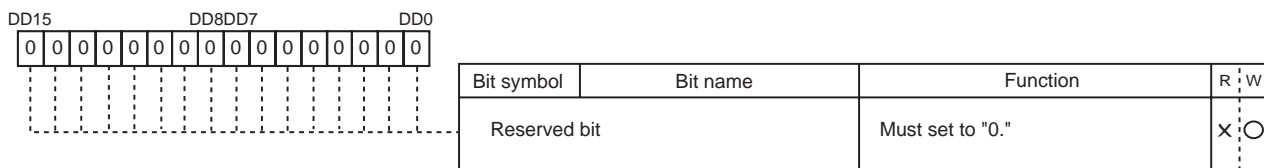
Bit symbol	Bit name	Function	R	W															
RMHTD0(0)	Remote control header length selection bit	In order to detect a remote control pulse in standby mode, the header length to the oscillation for clocks (address 32 ₁₆) is chosen. $A = TXCIN \times \sum_{n=0}^8 2^n RMHTD0(n)$ $C = TXCIN \times \sum_{n=0}^8 2^n RMHTD1(n)$ $B = TXCIN \times FILDIV0 \times \sum_{n=0}^5 YUKOU0(n)$ $D = TXCIN \times FILDIV0 \times \sum_{n=0}^5 YUKOU1(n)$ TXCIN : XCIN pin input cycle Division value set by FILDIV0 (bit 10 and 9 of address 33 ₁₆)	0	0															
RMHTD0(1)			0	0															
RMHTD0(2)			0	0															
RMHTD0(3)			0	0															
RMHTD0(4)			0	0															
RMHTD0(5)			0	0															
RMHTD0(6)			0	0															
RMHTD0(7)			0	0															
RMHTD0(8)			0	0															
JSTCKDIV0	Clock division value of JUST CLOCK filter selection bit.	<table border="1"> <thead> <tr> <th>JSTCKDIV1</th> <th>JSTCKDIV0</th> <th>Main clock divided value</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>32 divided</td> </tr> <tr> <td>0</td> <td>1</td> <td>64 divided</td> </tr> <tr> <td>1</td> <td>0</td> <td>128 divided</td> </tr> <tr> <td>1</td> <td>1</td> <td>256 divided</td> </tr> </tbody> </table>	JSTCKDIV1	JSTCKDIV0	Main clock divided value	0	0	32 divided	0	1	64 divided	1	0	128 divided	1	1	256 divided	0	0
JSTCKDIV1			JSTCKDIV0	Main clock divided value															
0	0	32 divided																	
0	1	64 divided																	
1	0	128 divided																	
1	1	256 divided																	
JSTCKDIV1	0	0																	
JSTCKON	ON/OFF of JUST CLOCK filter selection bit.	0 Filter OFF 1 Filter ON	0	0															
	Nothing is assigned.		x	x															
RMTSEL	Remote control header polarity selection bit	0 No reverse 1 reverse	0	0															

(38) Address 33₁₆ (=DA5 to 0)



Note 1. Refer to RMHTD0 (0) to (8) (address 32₁₆)

(39) Address 34₁₆ (=DA5 to 0)



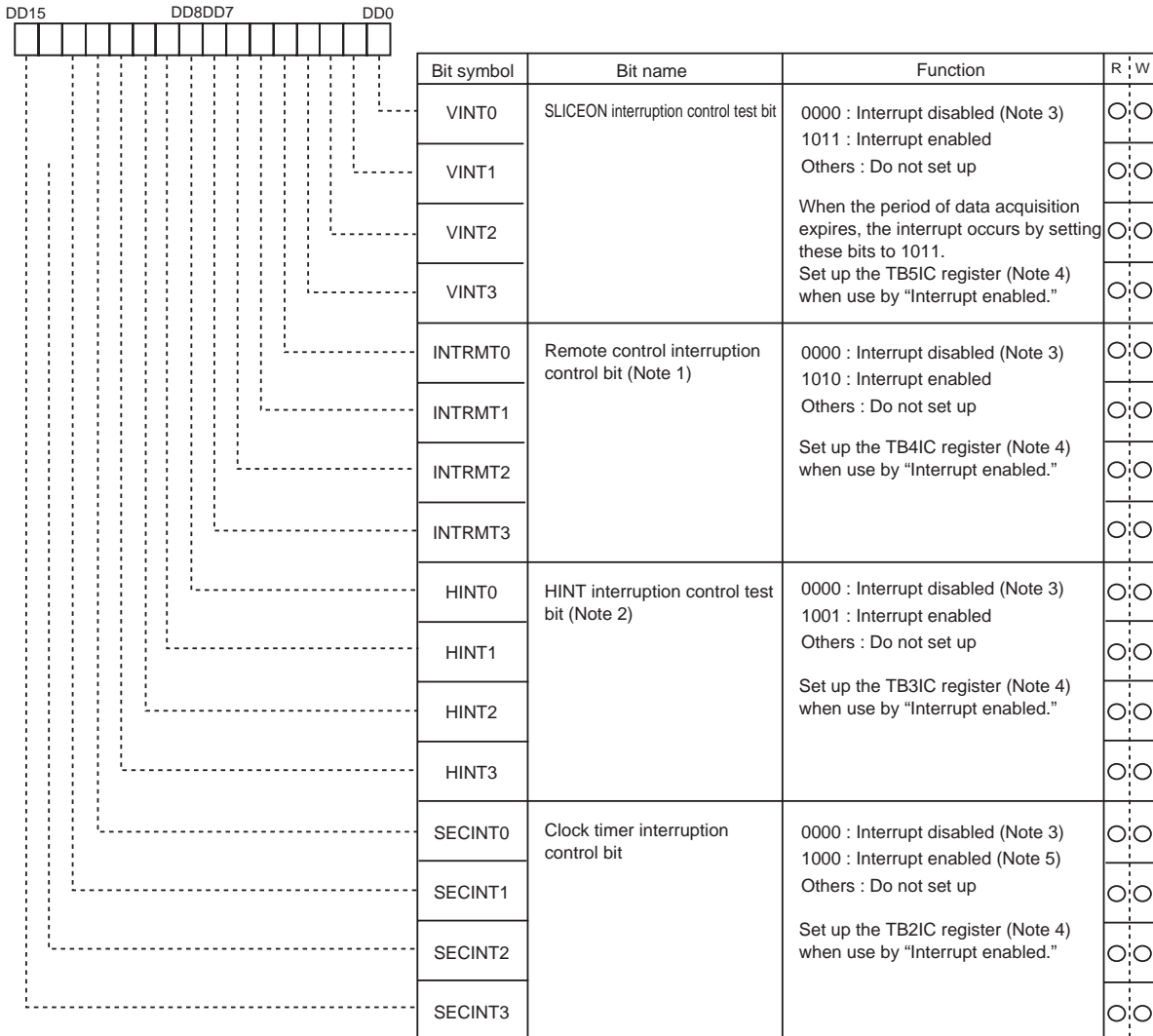
(40) Address 3516 (=DA5 to 0)



Bit symbol	Bit name	Function	R	W															
HINT_LINE0	H_INT interruption position selection bit	A period after V is inputted until H_INT rises is counted. 	○	○															
HINT_LINE1			○	○															
HINT_LINE2			○	○															
HINT_LINE3			○	○															
HINT_LINE4			○	○															
HINT_LINE5			○	○															
HINT_LINE6			○	○															
HINT_LINE7			○	○															
HINT_LINE8			○	○															
PTC8	Port P11 output control bit	<table border="1"> <tr> <th>PTC8</th> <th>PTD8</th> <th></th> </tr> <tr> <td>0</td> <td>0</td> <td>Fixed to "L"</td> </tr> <tr> <td>0</td> <td>1</td> <td>Fixed to "H"</td> </tr> <tr> <td>1</td> <td>0</td> <td>reverse (Note 1)</td> </tr> <tr> <td>1</td> <td>1</td> <td>No reverse (Note 1)</td> </tr> </table>	PTC8	PTD8		0	0	Fixed to "L"	0	1	Fixed to "H"	1	0	reverse (Note 1)	1	1	No reverse (Note 1)	○	○
PTC8		PTD8																	
0	0	Fixed to "L"																	
0	1	Fixed to "H"																	
1	0	reverse (Note 1)																	
1	1	No reverse (Note 1)																	
PTD8			○	○															
Reserved bit		Must set to "0."	X	○															
EXAOFF	P11 output signal selection bit (Note 2)	0	SLICEON signal																
		1	H_INT signal (Note 3)																

- Note 1. Signal selected by the EXAOFF bit is output.
- Note 2. For PTC8 = "1" setting.
- Note 3. Refer to HINT_LINE_n.

(41) Address 3616 (=DA5 to 0)



Note 1. Refer to 2.14.6 Expansion Register Construction Composition.

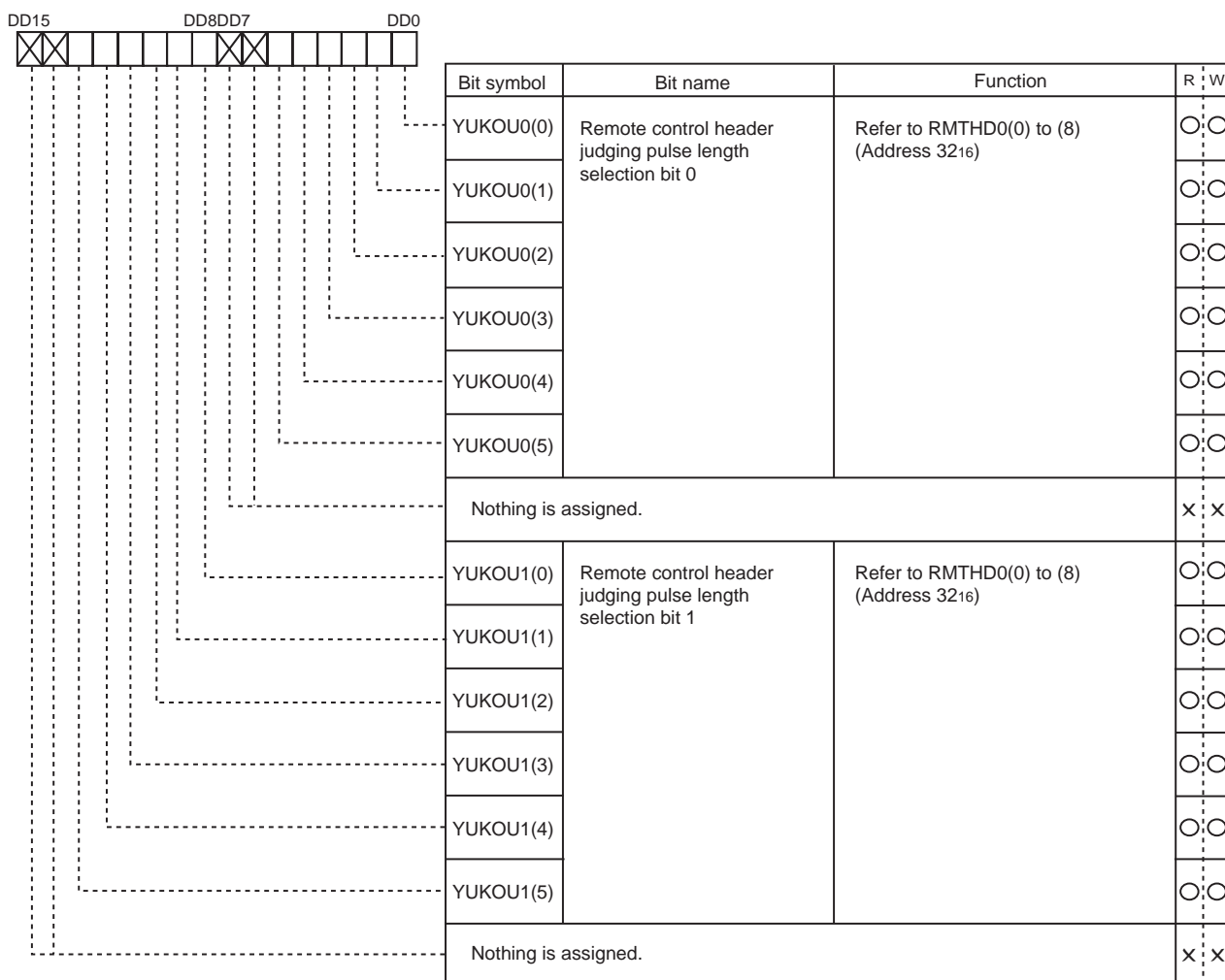
Note 2. Refer to the function of HINT_LINEn (Address 3516.)

Note 3. Set these bits to 0000 when use the interrupt of Timer B3, Timer B4, or Timer B5.

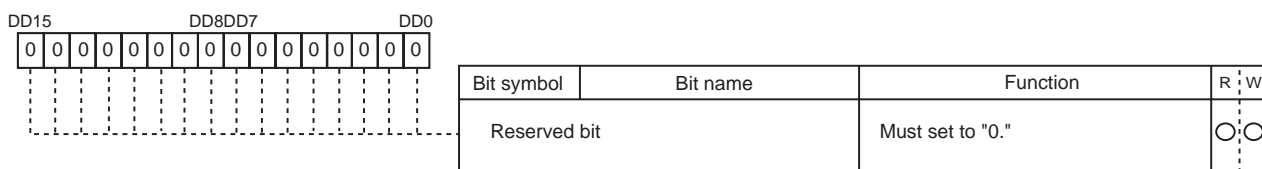
Note 4. Refer to Figure 2.7.3 Interrupt Control Registers.

Note 5. When the second counter (Address 3916) is changed, an interrupt is generated every 1 second.

(42) Address 37₁₆ (=DA5 to 0)



(43) Address 38₁₆ (=DA5 to 0)



(44) Address 39₁₆ (=DA5 to 0)



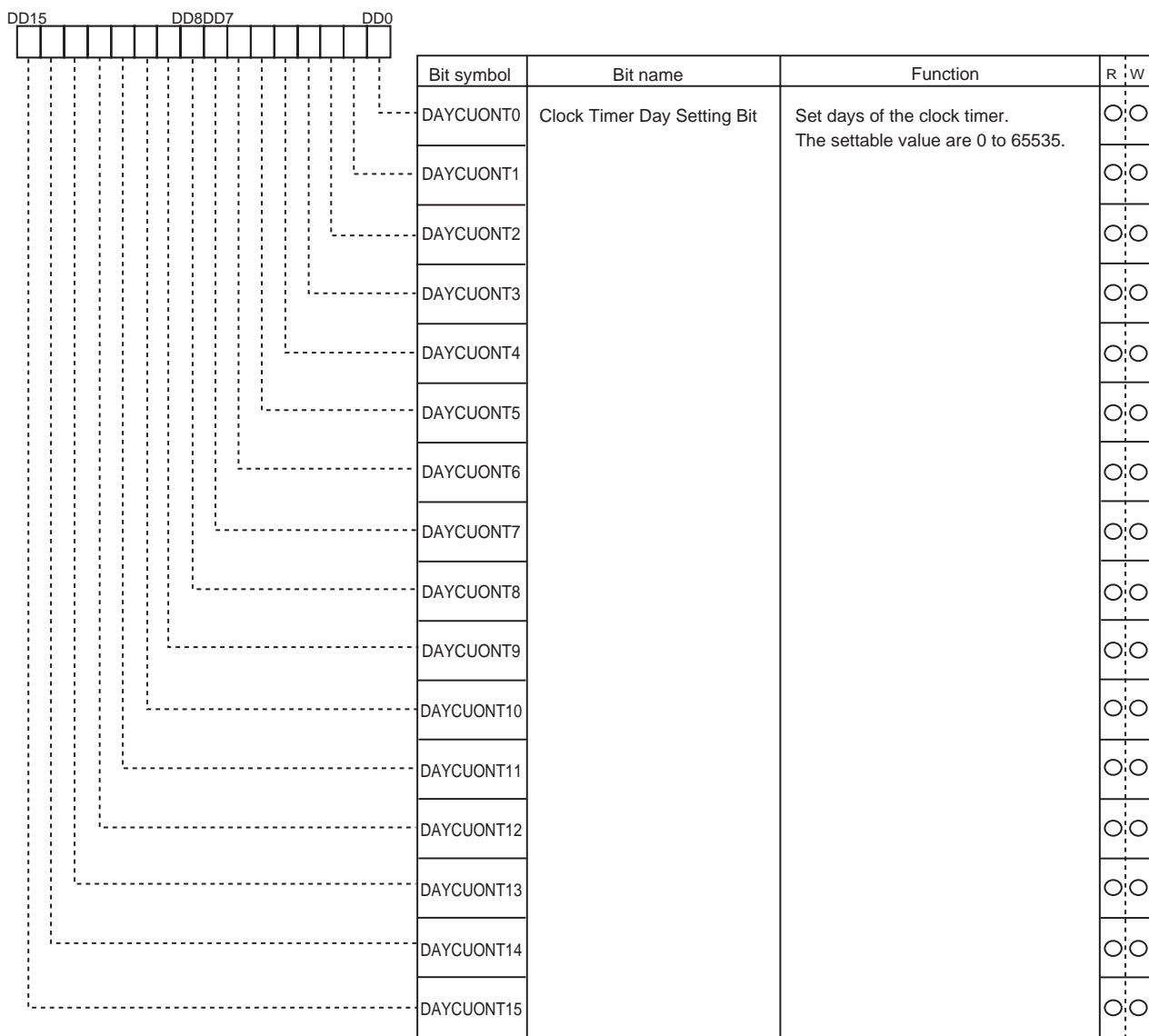
Bit symbol	Bit name	Function	R	W
SECOUT0	Clock Timer Second Setting Bit	Set seconds (0 to 59 seconds) of clock timer. The settable values are 0 to 59.	○	○
SECOUT1			○	○
SECOUT2			○	○
SECOUT3			○	○
SECOUT4			○	○
SECOUT5			○	○
Nothing is assigned.			x	x
RTCON	Clock Timer Operation Selection Bit	0	○	○
		1	○	○
Nothing is assigned.			x	x
SECJUST	Second Just Setting Bit	When writing "1", less than second of the clock timer is reset. When reading, the value is "0".	x	○

(45) Address 3A₁₆ (=DA5 to 0)

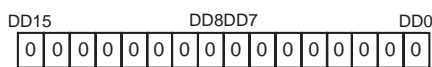


Bit symbol	Bit name	Function	R	W
MINOUT0	Clock Timer Minute Setting Bit	Set hours and minutes of the clock timer by the minute. The settable values are 0 to 1439 (00:00 to 23:59)	○	○
MINOUT1			○	○
MINOUT2			○	○
MINOUT3			○	○
MINOUT4			○	○
MINOUT5			○	○
MINOUT6			○	○
MINOUT7			○	○
MINOUT8			○	○
MINOUT9			○	○
MINOUT10			○	○
Nothing is assigned.			x	x

(46) Address 3B₁₆ (=DA5 to 0)

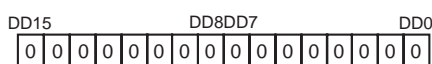


(47) Address 3C₁₆ (=DA5 to 0)



Bit symbol	Bit name	Function	R	W
	Reserved bit	Must set to "0."	○	○

(48) Address 3D₁₆ (=DA5 to 0)



Bit symbol	Bit name	Function	R	W
	Reserved bits	Must set to "0."	○	○

(49) Address 3E₁₆ (=DA5 to 0)



Bit symbol	Bit name	Function	R	W
	Nothing is assigned.		X	X
	Reserved bits	Must set to "0."	X	○

(50) Address 3F₁₆ (=DA5 to 0)



Bit symbol	Bit name	Function	R	W
	Nothing is assigned.		X	X
	Reserved bit	Must set to "0."	X	○

2.14.6 Expansion Register Construction Composition

(1) Acquisition timing

The SLICEON signal is output in the acquisition possible period.

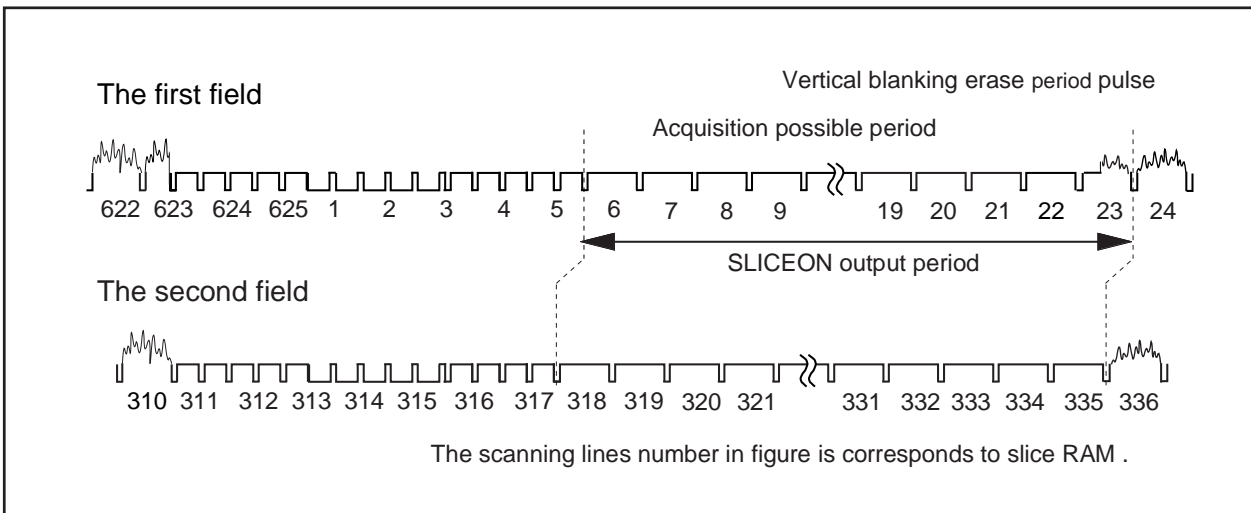


Figure 2.14.11 Acquisition timing

(2) Synchronized signal detection circuit

The number of pulses of the horizontal synchronized signal of a compound video signal is counted during a fixed period. The horizontal synchronous number of pulses can always be read from an expansion register.

A block diagram is shown in Figure. 2.14.12.

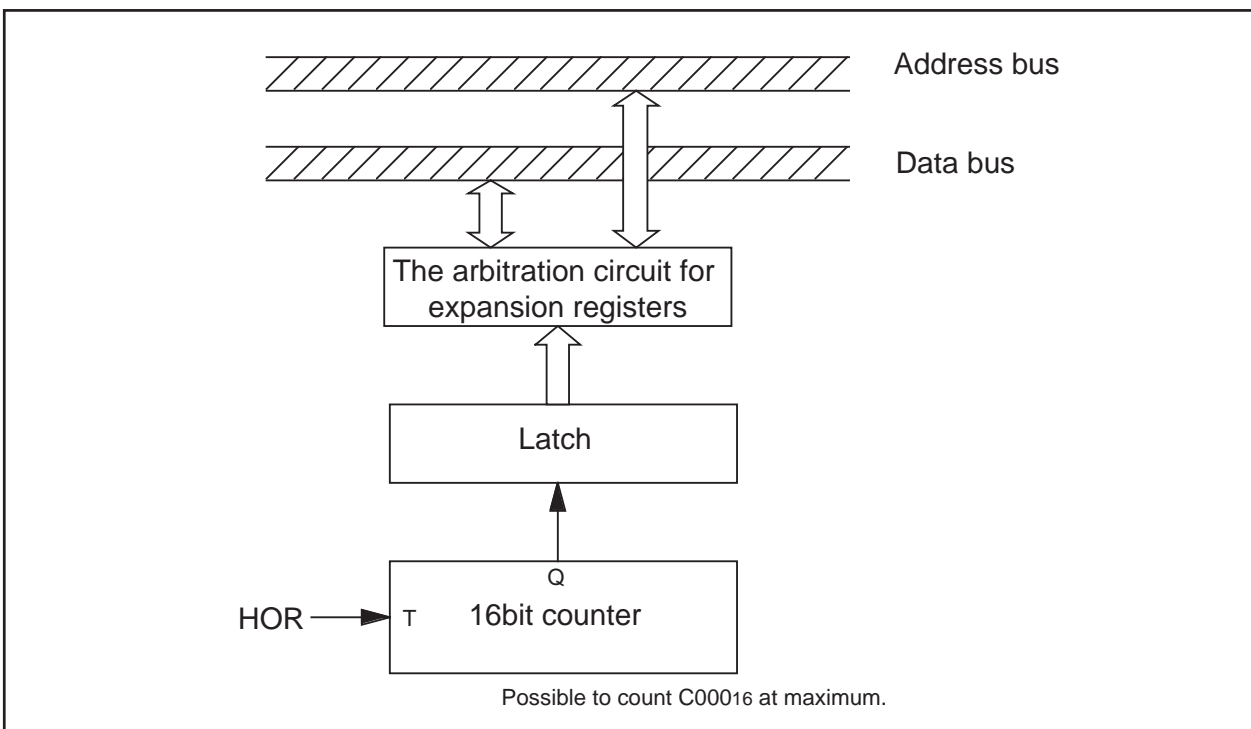


Figure 2.14.12 Block diagram of Synchronized detection circuit

(3) Register related to Slicer

The relation between V, H signal, and the register related to slicer is shown in Figure. 2.14.13 and Figure. 2.14.14.

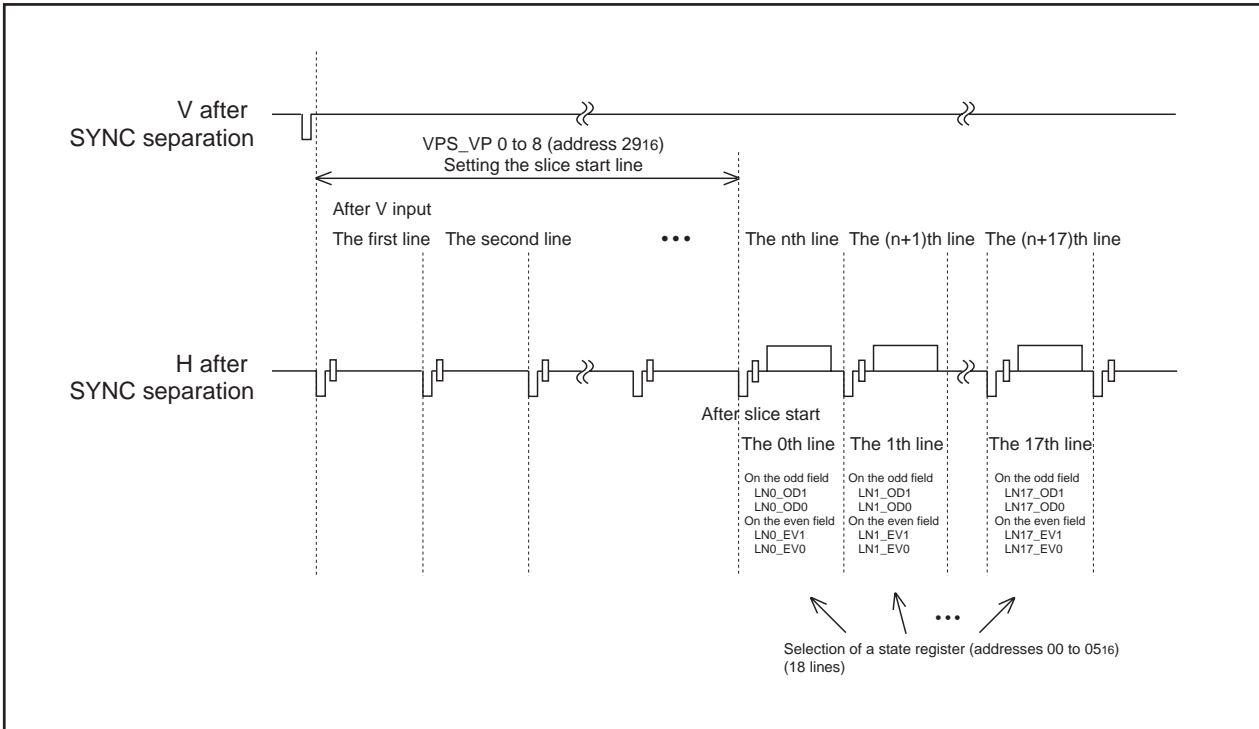


Figure 2.14.13 Register related to slicer (1)

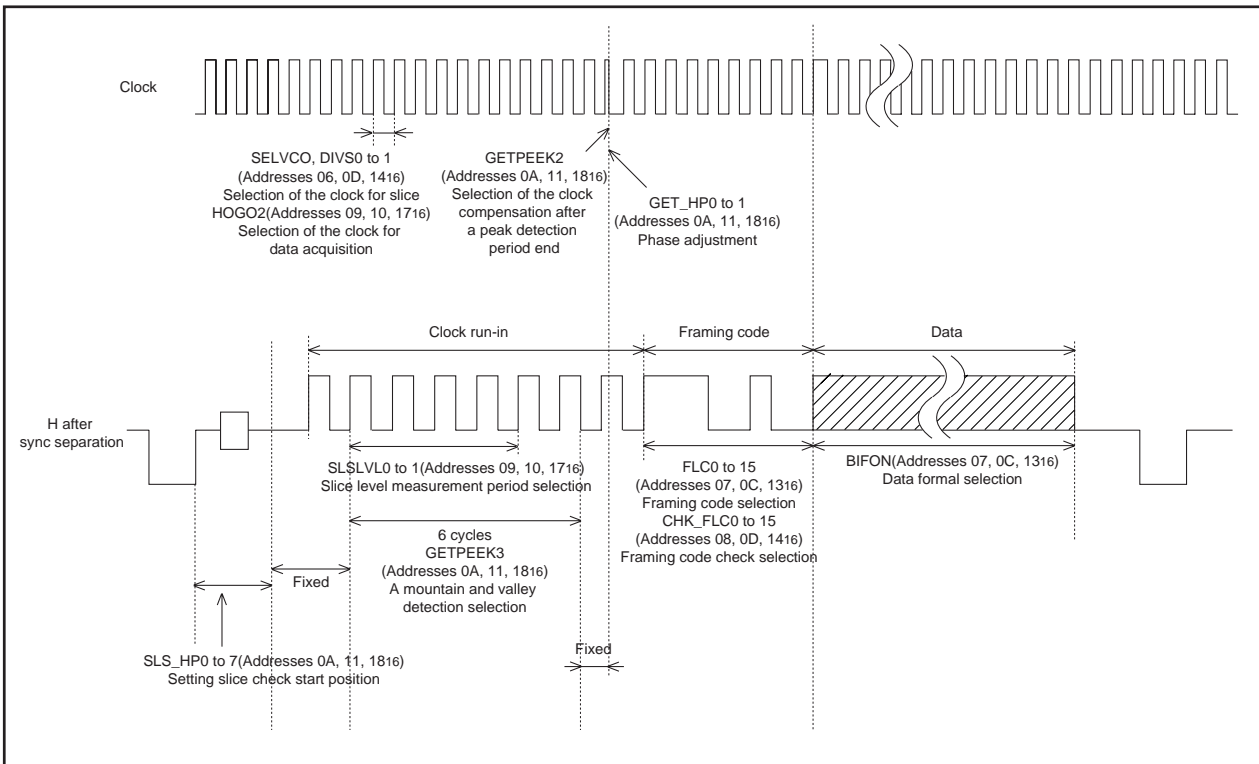


Figure 2.14.14 Register related to slicer (2)

(4) Remote control pattern recognition

Pattern matching of remote control is performed using a sub clock oscillation. Remote control input is input from RMTIN terminal. Interruption is generated when pattern matching is in agreement.

The example of a waveform of pattern matching is shown in Figure.2.14.15.

The flow of pattern matching is shown in Figure.2.14.16.

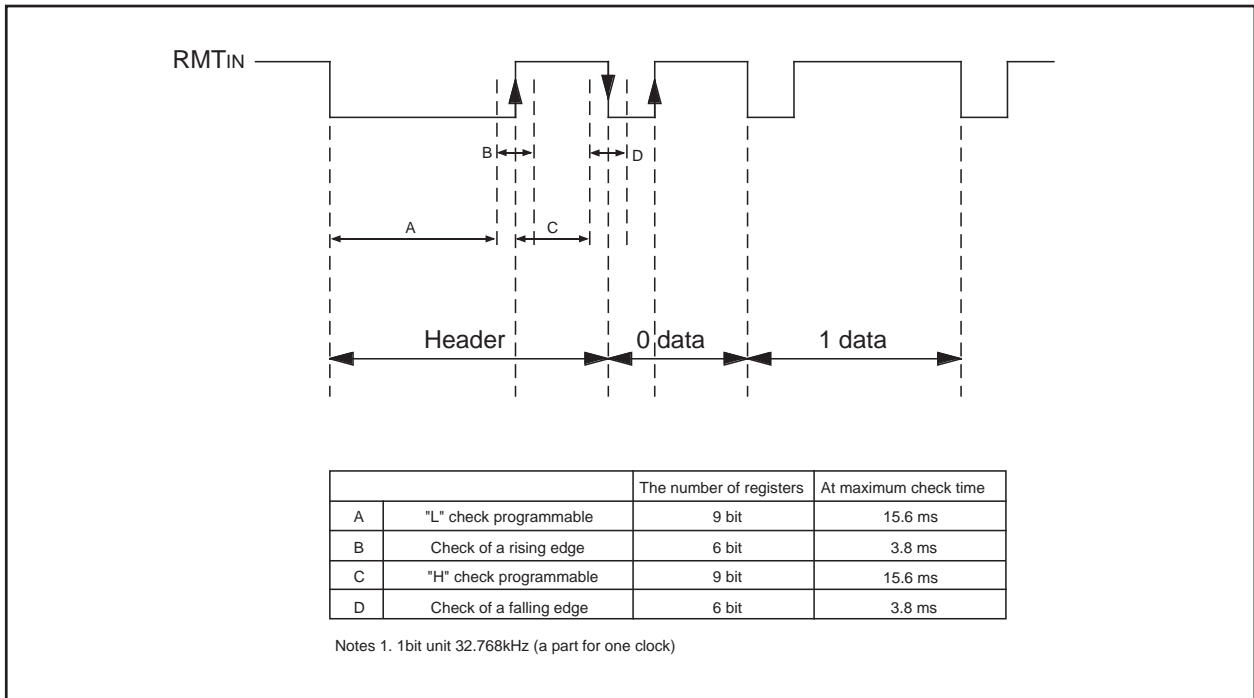


Figure 2.14.15 Example of waveform of pattern matching

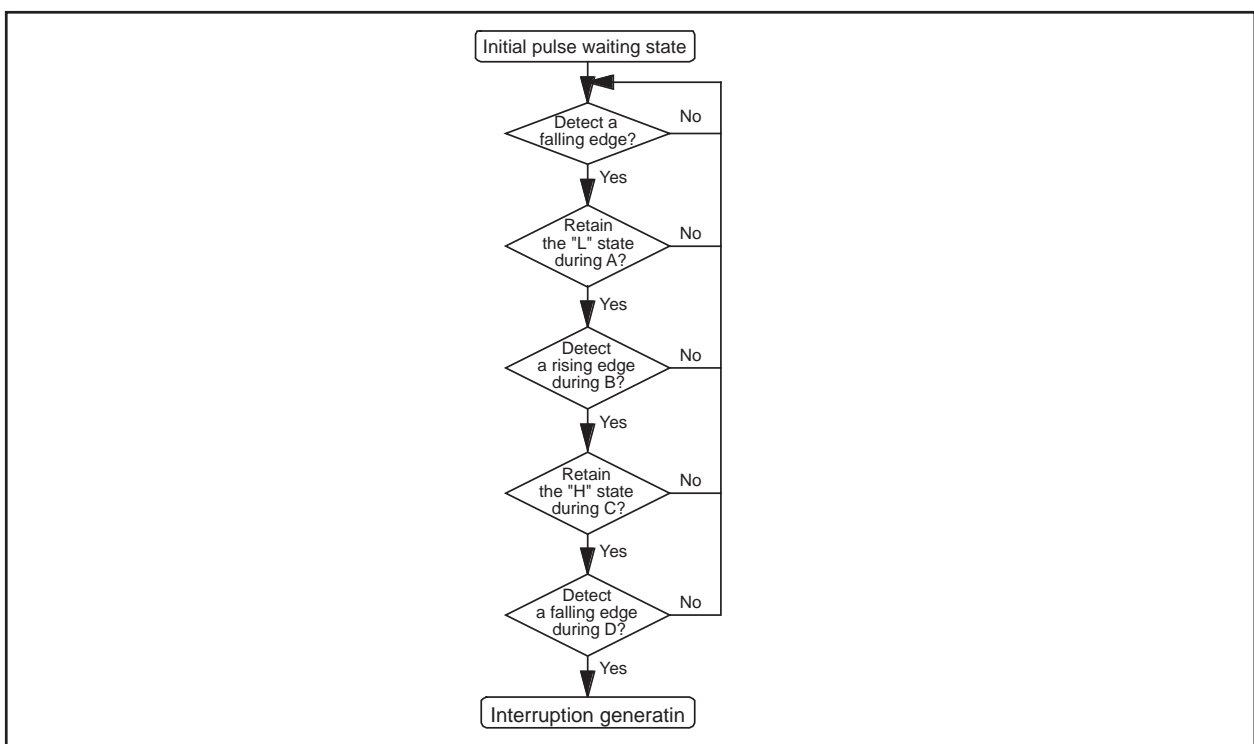


Figure 2.14.16 Flow of pattern matching

2.14.7 8/4 Humming Decoder

8/4 humming decoder operates only by written the data which is 8/4 humming- encoded to 8/4 humming register (address 021A16). 8/4 humming register consists of 16 bits, can decode two data at once. Can obtain the decoded result by reading 8/4 humming register, and the decoded value and error information are output. Corrects and outputs the decoded value for single error, and outputs only error information for double error. Decoded result is shown in Figure 2.14.17 and humming 8/4 register composition is shown in Figure 2.14.18.

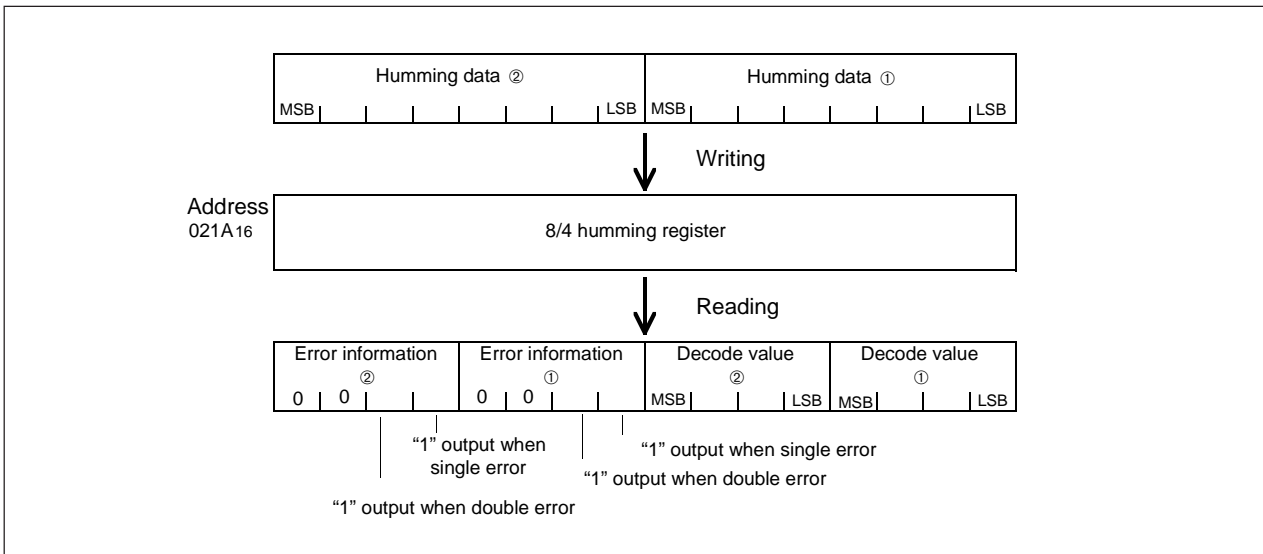


Figure 2.14.17 Decoded result

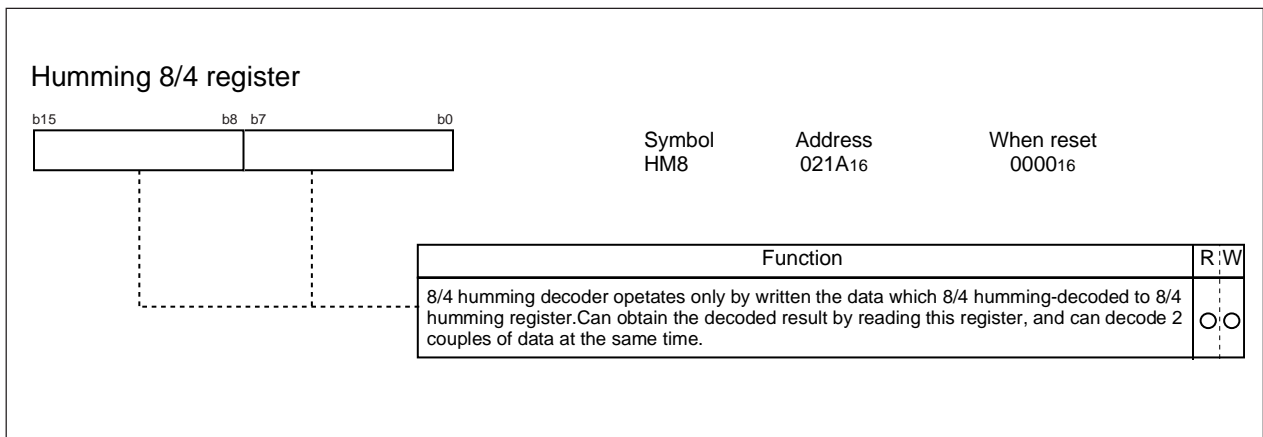


Figure 2.14.18 Humming 8/4 register composition

2.14.8 24/18Humming Decoder

24/18 humming decoder operates only by written the data which is 24/18 humming-encoded to 24/18 humming register 0 (address 021C16) and 1 (address 021E16). Can obtain the decoded result by reading the same 24/18 humming register, and the decoded value and error information are output. Decoded result is shown in Figure 2.14.19 and humming 24/18 register composition is shown in Figure 2.14.20.

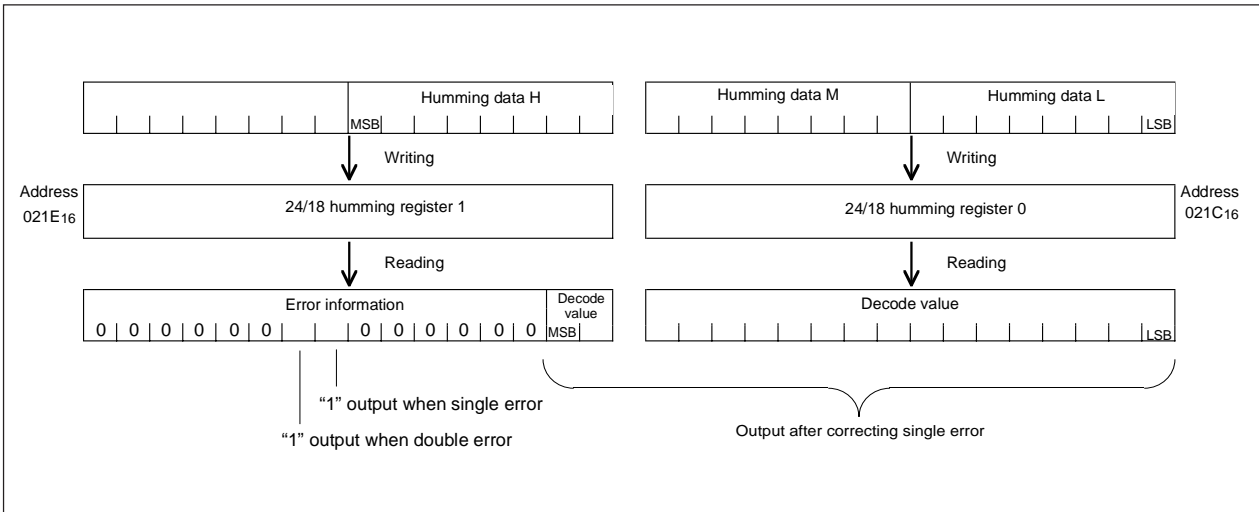


Figure 2.14.19 Decoded result

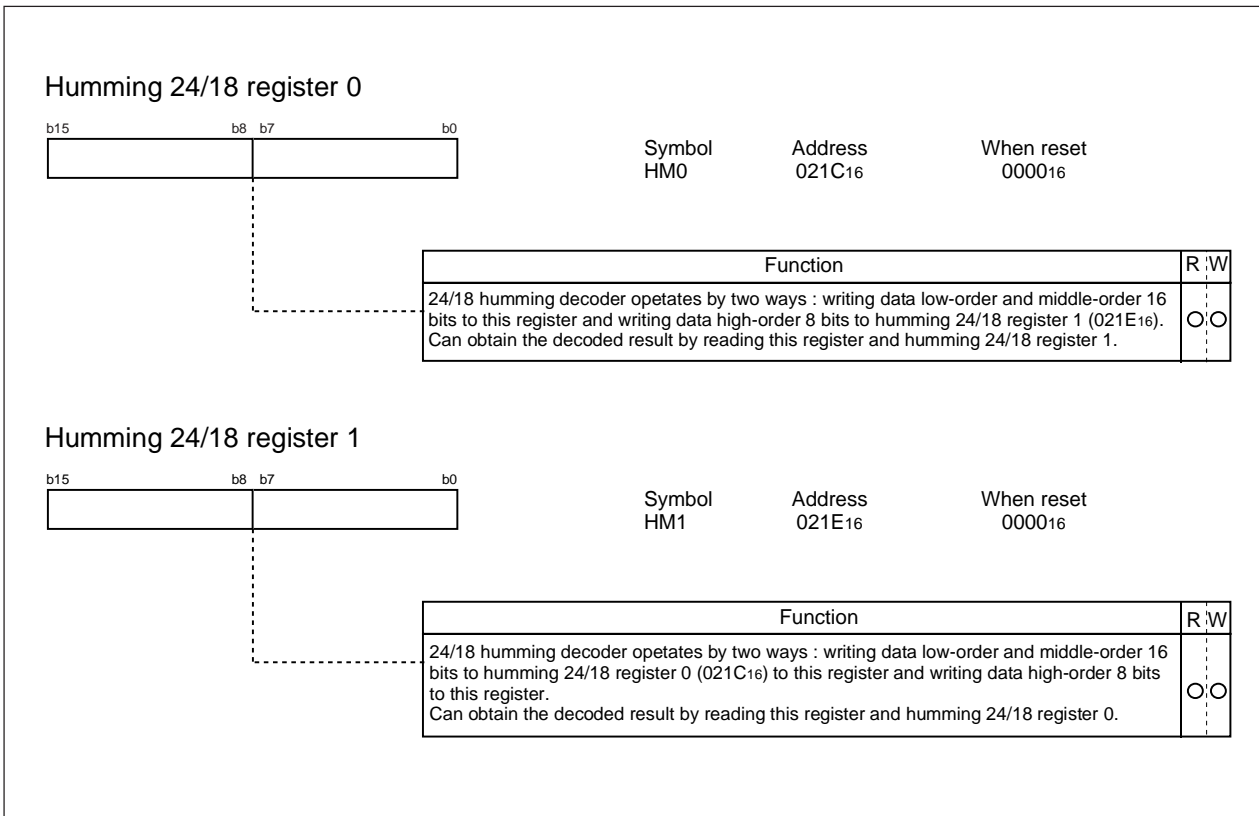


Figure 2.14.20 Humming 24/18 register composition

Continuous error correction

When uses humming 8/4 (address 021A₁₆) at the same time as humming 24/18, can do the continuous error correction.

Continuous error correction sequence is shown in Figure 2.14.21.

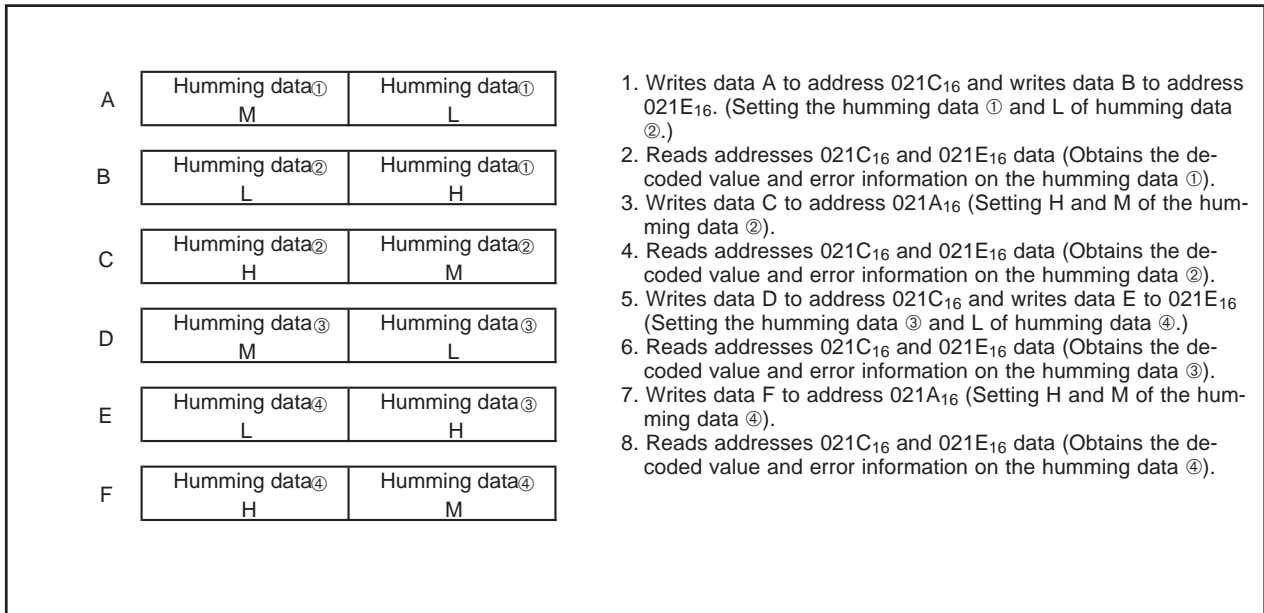


Figure 2.14.21 Continuous error correction sequence

Then, because using a part of circuit of humming 8/4 about this operation, cannot use this operation at the same time.

When using the humming circuit, do the decoded result reading operation at once after the setting data of humming. And do not access other memories (Including the humming circuit) before reading of the decoded result.

2.14.9 I/O Composition of pins for Expansion Function

Figure 2.14.22 and figure 2.14.23 show pins for expansion function.

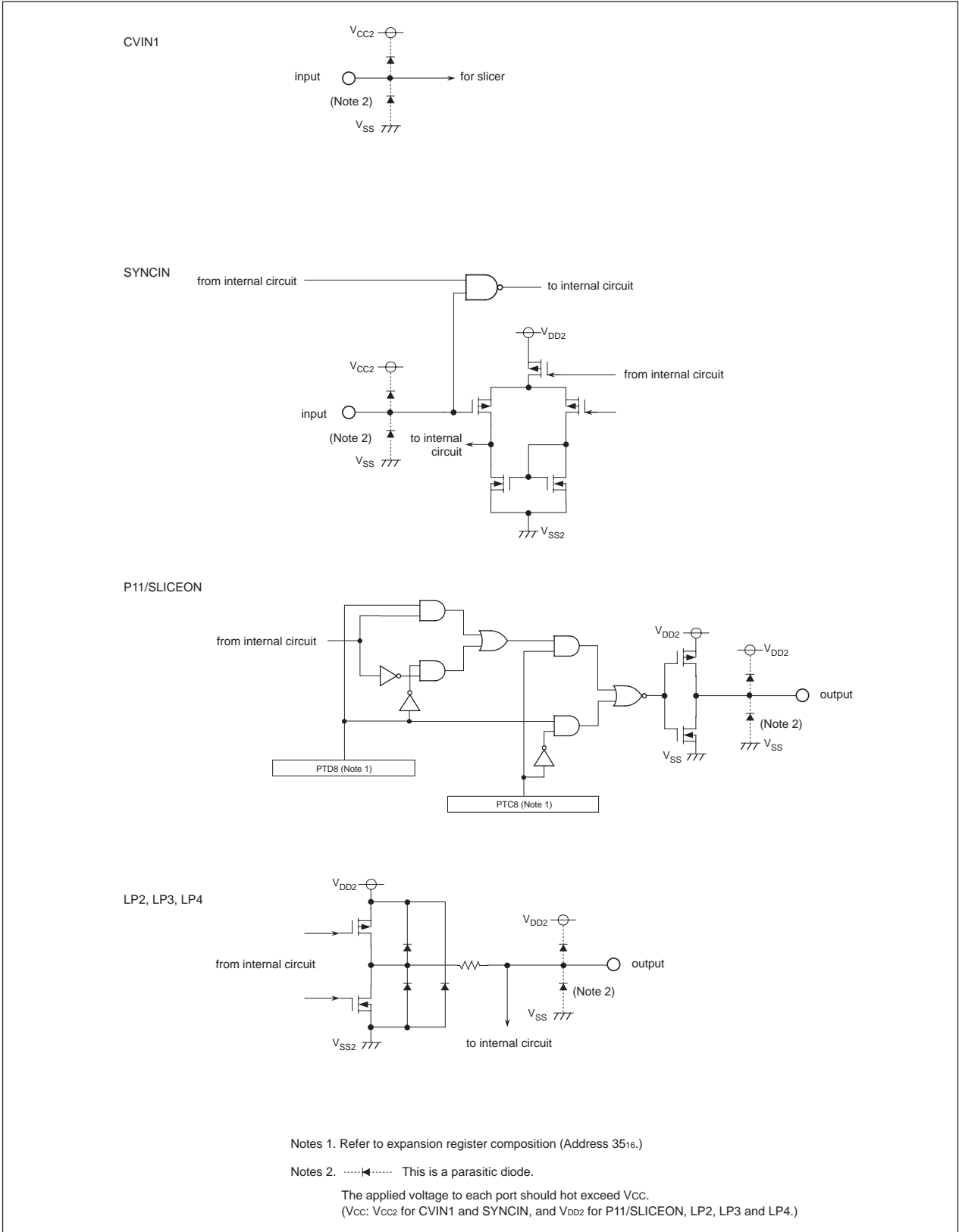


Figure 2.14.22 Pins for expansion function (1)

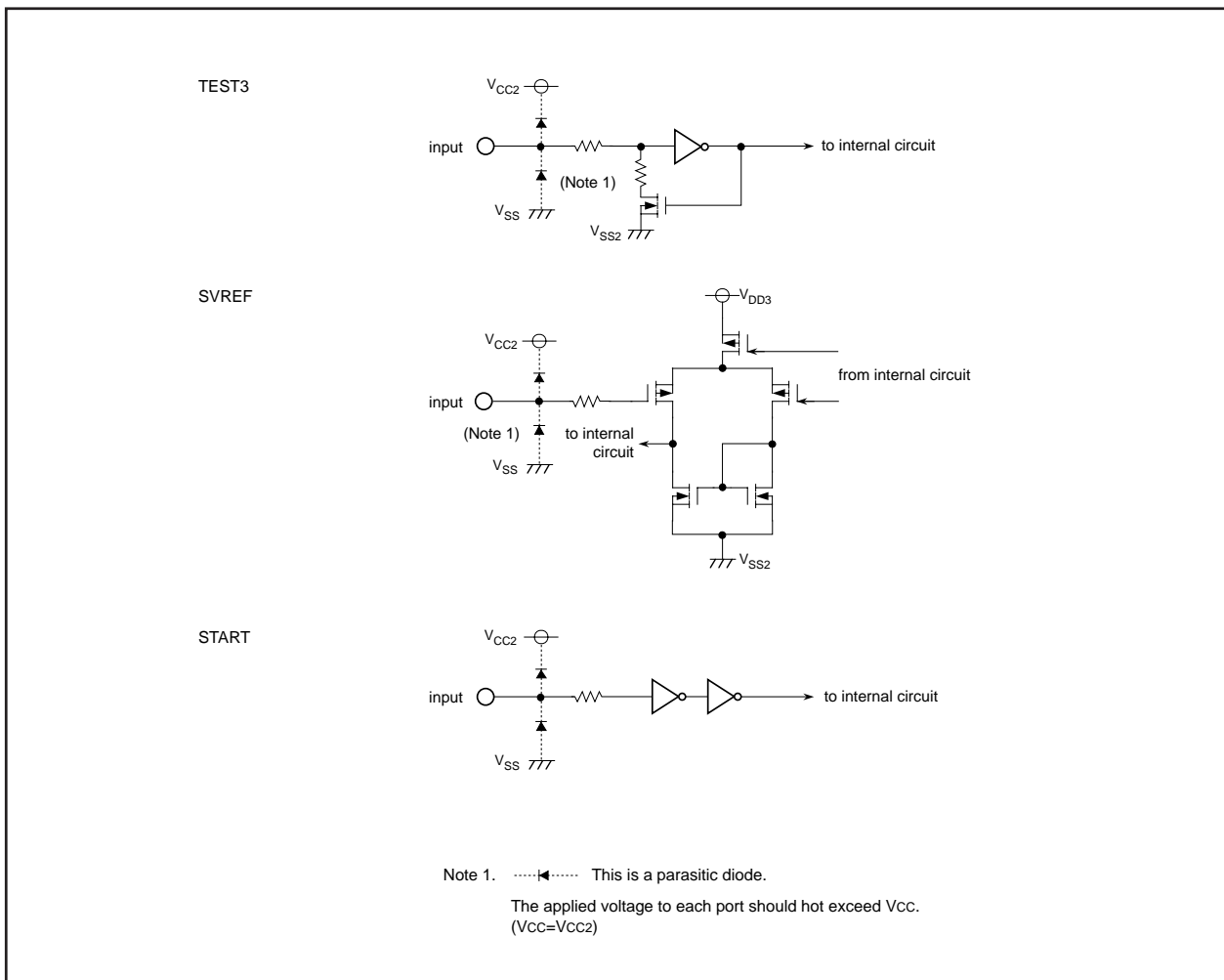


Figure 2.14.23 Pins for expansion function (2)

2.15 Programmable I/O Ports

The programmable input/output ports (hereafter referred to simply as “I/O ports”) consist of 87 lines P0 to P10 (except P85). Each port can be set for input or output every line by using a direction register, and can also be chosen to be or not be pulled high every 4 lines. P85 is an input-only port and does not have a pull-up resistor. Port P85 shares the pin with $\overline{\text{NMI}}$, so that the $\overline{\text{NMI}}$ input level can be read from the P8 register P8_5 bit.

Figures 2.15.1 to 2.15.5 show the I/O ports. Figure 2.15.6 shows the I/O pins.

Each pin functions as an I/O port, a peripheral function input/output, or a bus control pin.

For details on how to set peripheral functions, refer to each functional description in this manual. If any pin is used as a peripheral function input, set the direction bit for that pin to “0” (input mode). Any pin used as an output pin for peripheral functions is directed for output no matter how the corresponding direction bit is set.

When using any pin as a bus control pin, refer to “Bus Control.”

(1) Port Pi Direction Register (PDi Register, i = 0 to 10)

Figure 2.15.7 shows the direction registers.

This register selects whether the I/O port is to be used for input or output. The bits in this register correspond one for one to each port.

During memory extension and microprocessor modes, the PDi registers for the pins functioning as bus control pins (A0 to A19, D0 to D15, $\overline{\text{CS0}}$ to $\overline{\text{CS3}}$, $\overline{\text{RD}}$, $\overline{\text{WRL/WR}}$, $\overline{\text{WRH/BHE}}$, ALE, RDY, HOLD, HLDA, and BCLK) cannot be modified.

No direction register bit for P85 is available.

(2) Port Pi Register (Pi Register, i = 0 to 10)

Figure 2.15.8 show the Pi registers.

Data input/output to and from external devices are accomplished by reading and writing to the Pi register. The Pi register consists of a port latch to hold the output data and a circuit to read the pin status. For ports set for input mode, the input level of the pin can be read by reading the corresponding Pi register, and data can be written to the port latch by writing to the Pi register.

For ports set for output mode, the port latch can be read by reading the corresponding Pi register, and data can be written to the port latch by writing to the Pi register. The data written to the port latch is output from the pin. The bits in the Pi register correspond one for one to each port.

During memory extension and microprocessor modes, the PDi registers for the pins functioning as bus control pins (A0 to A19, D0 to D15, $\overline{\text{CS0}}$ to $\overline{\text{CS3}}$, $\overline{\text{RD}}$, $\overline{\text{WRL/WR}}$, $\overline{\text{WRH/BHE}}$, ALE, RDY, HOLD, HLDA, and BCLK) cannot be modified.

(3) Pull-up Control Register 0 to Pull-up Control Register 2 (PUR0 to PUR2 Registers)

Figure 2.15.9 shows the PUR0 to PUR2 registers.

The PUR0 to PUR2 register bits can be used to select whether or not to pull the corresponding port high in 4 bit units. The port chosen to be pulled high has a pull-up resistor connected to it when the direction bit is set for input mode.

However, the pull-up control register has no effect on P0 to P3, P40 to P43, and P5 during memory extension and microprocessor modes. Although the register contents can be modified, no pull-up resistors are connected.

(4) Port Control Register

Figure 2.15.10 shows the port control register.

When the P1 register is read after setting the PCR register’s PCR0 bit to “1”, the corresponding port latch can be read no matter how the PD1 register is set.

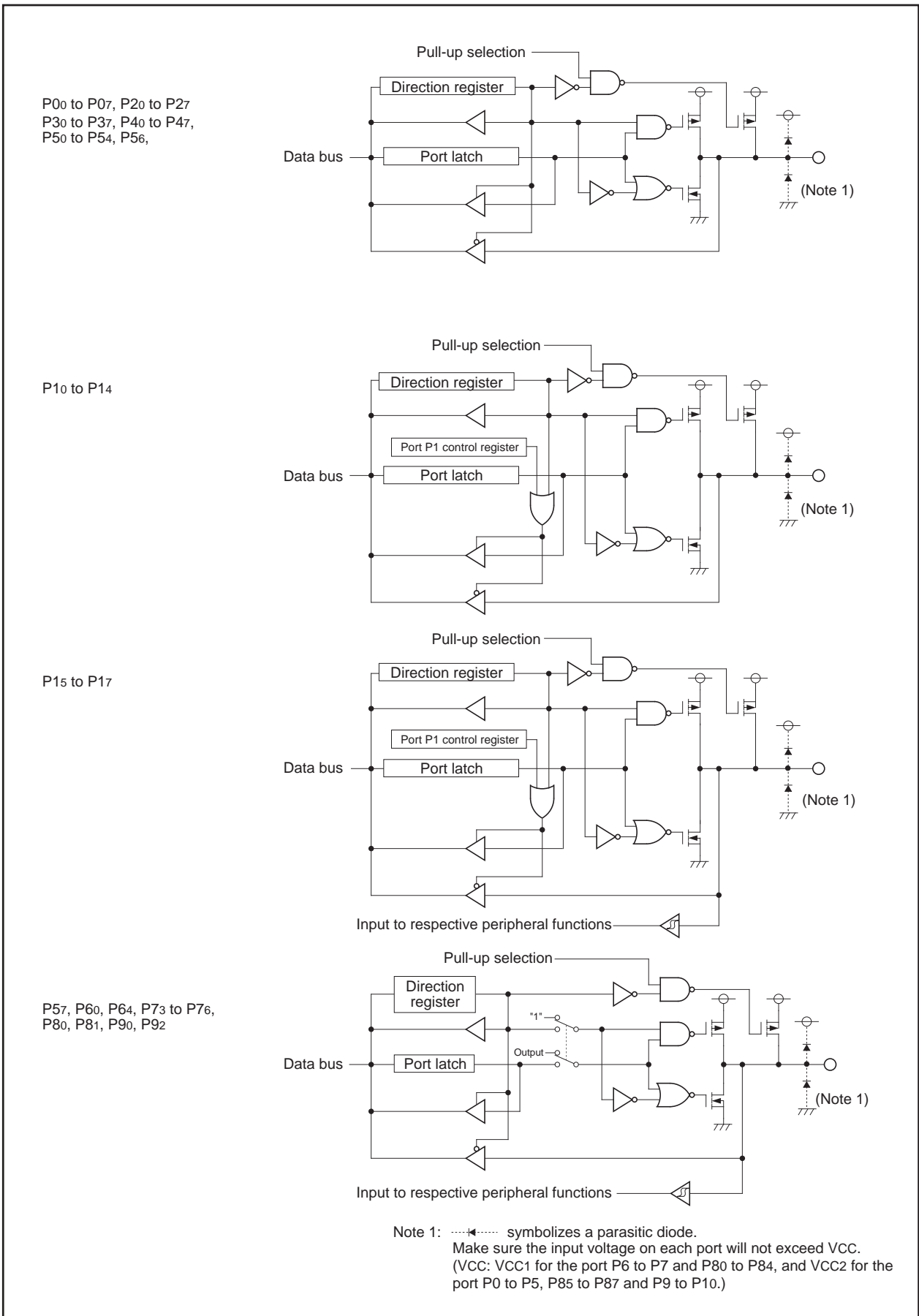


Figure 2.15.1. I/O Ports (1)

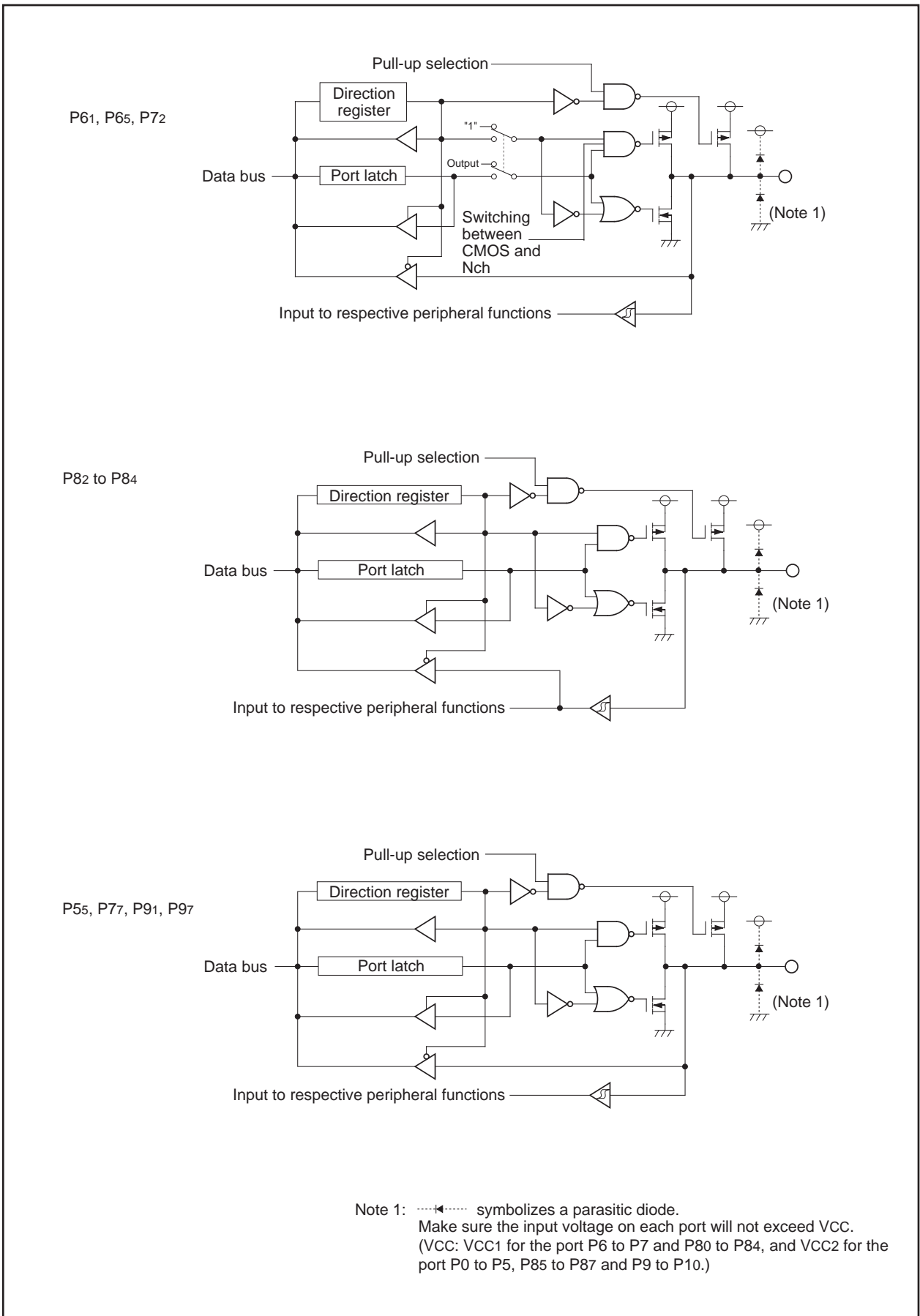


Figure 2.15.2. I/O Ports (2)

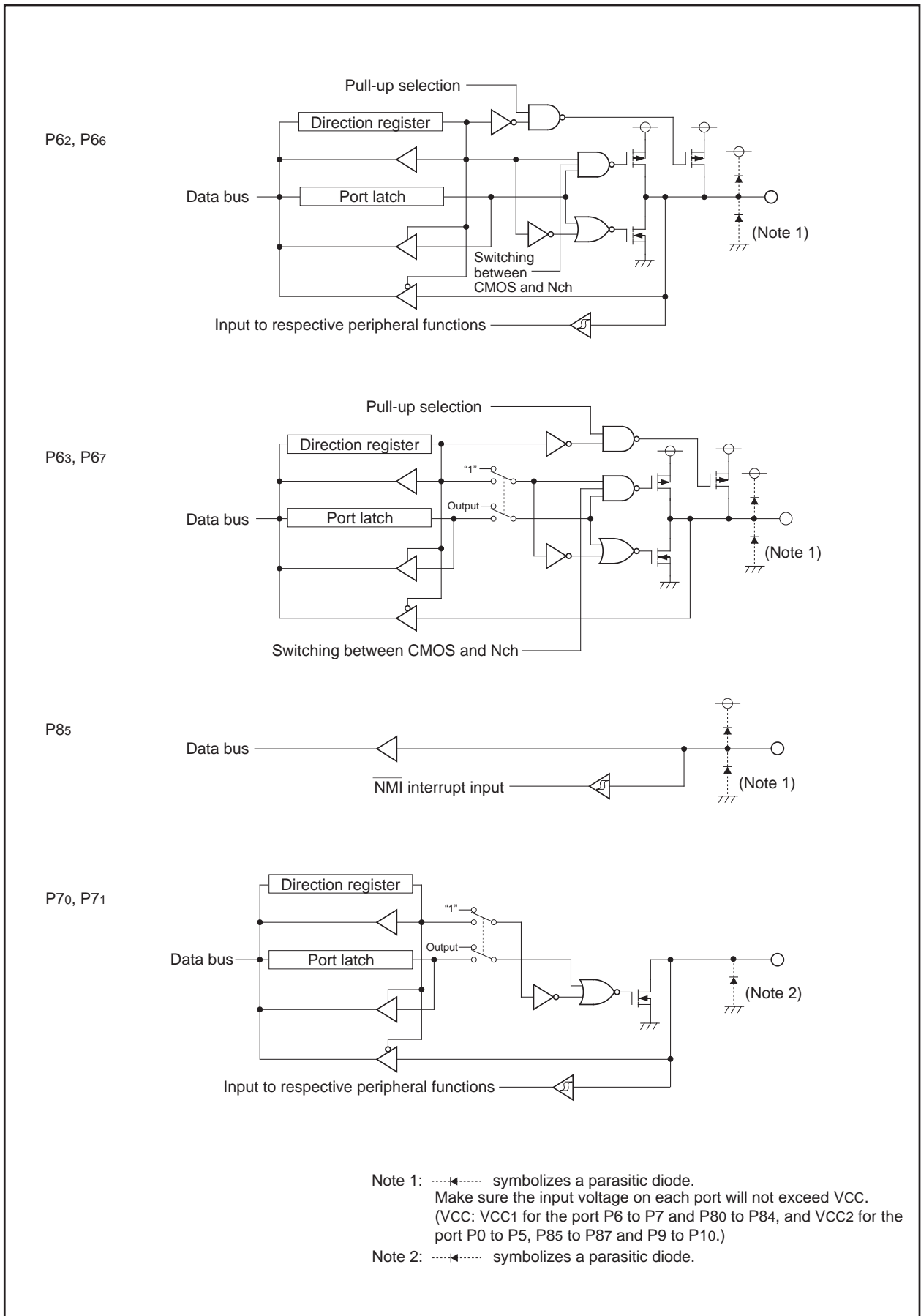


Figure 2.15.3. I/O Ports (3)

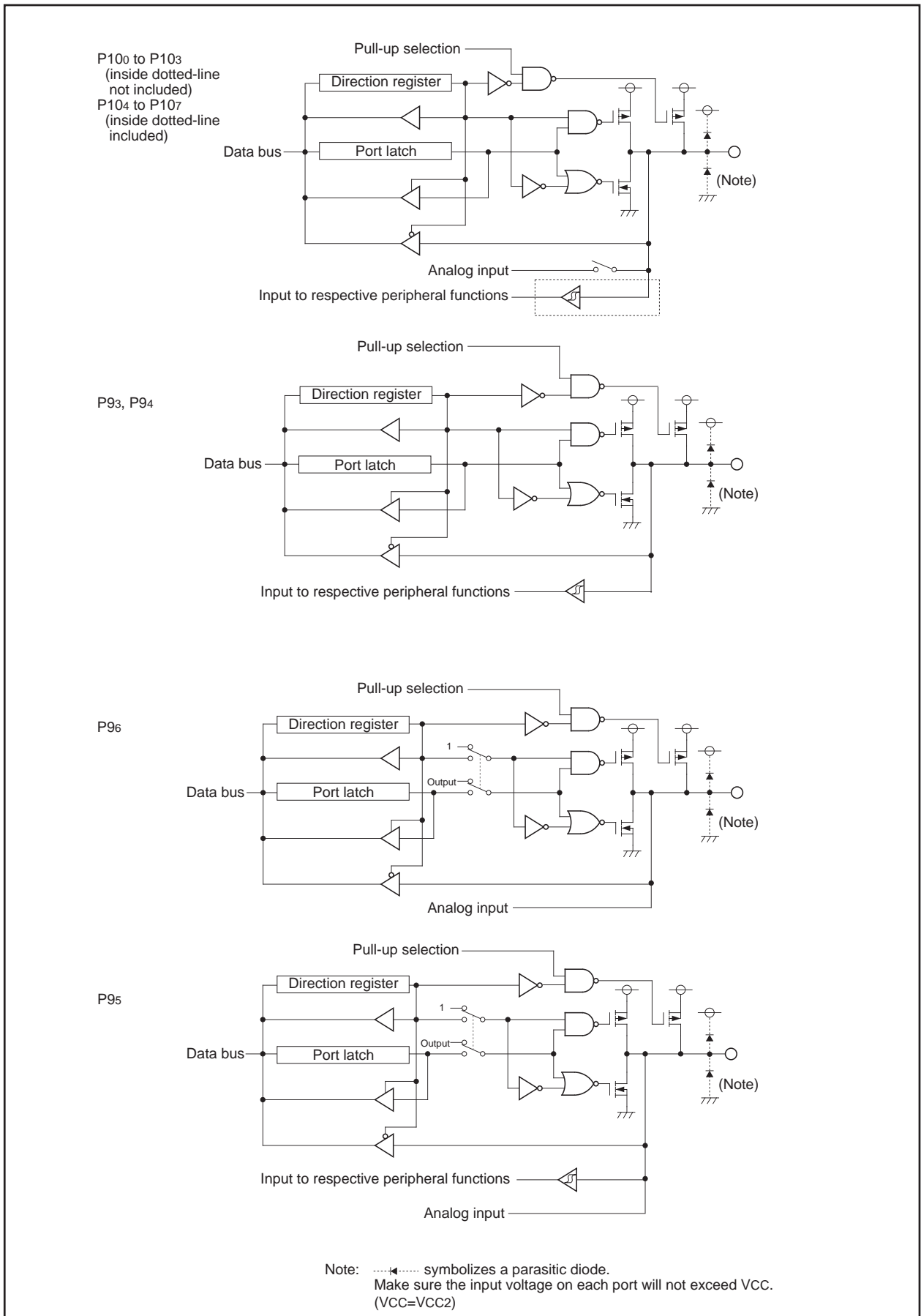


Figure 2.15.4. I/O Ports (4)

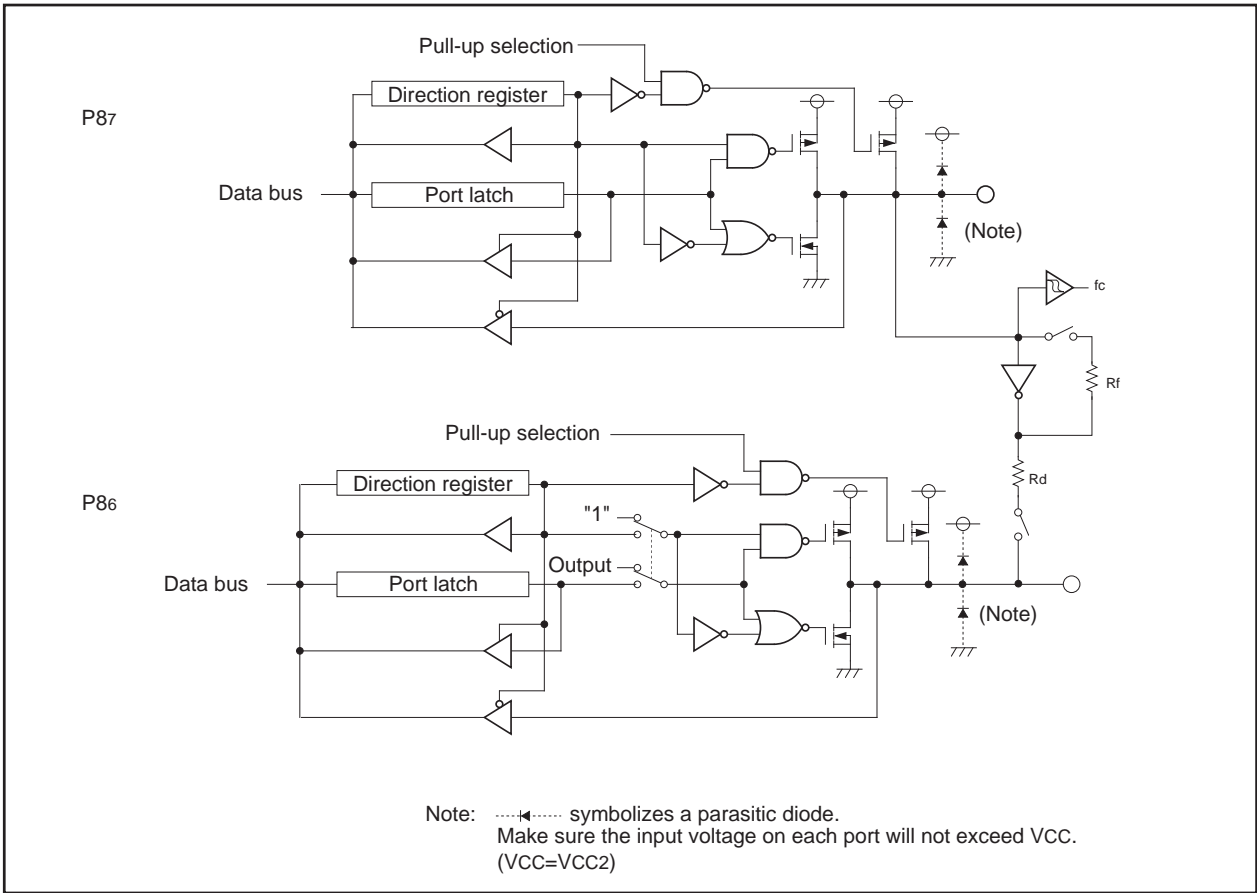


Figure 2.15.5. I/O Ports (5)

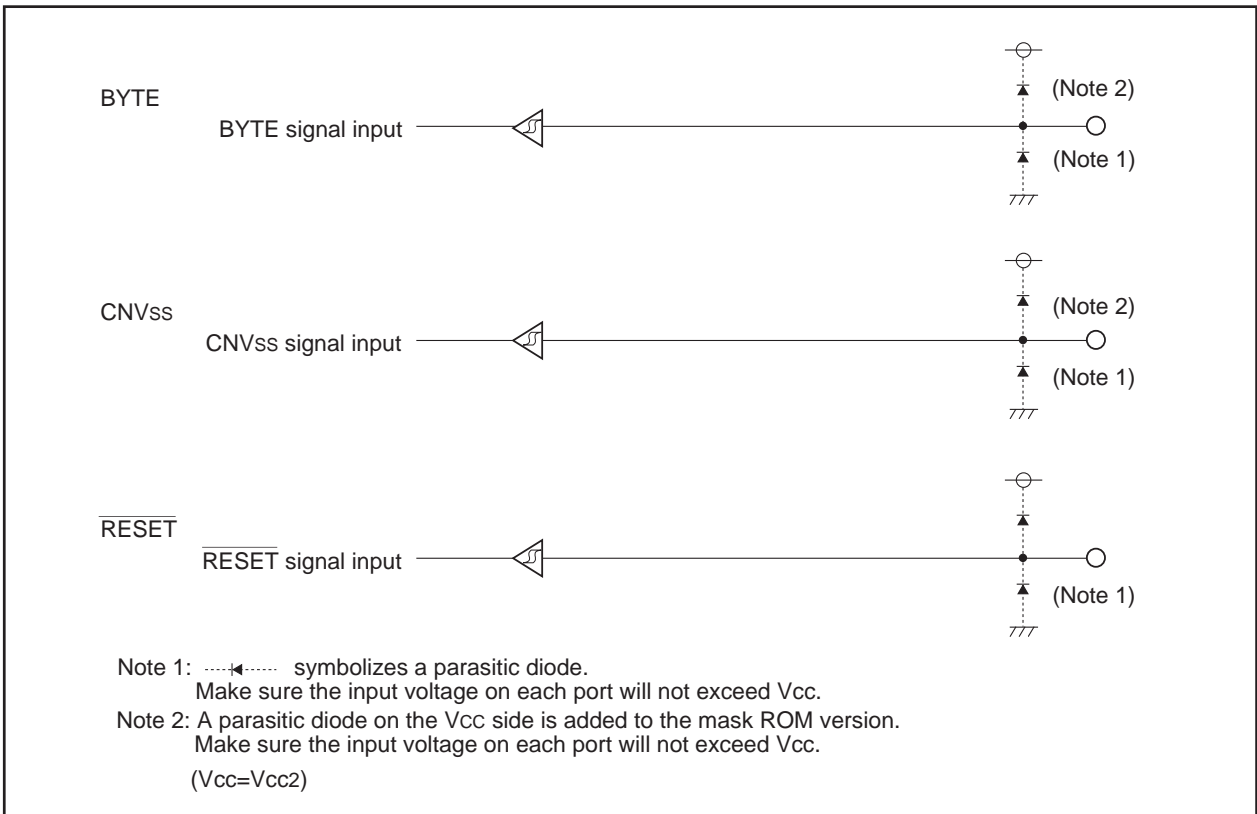


Figure 2.15.6. I/O Pins

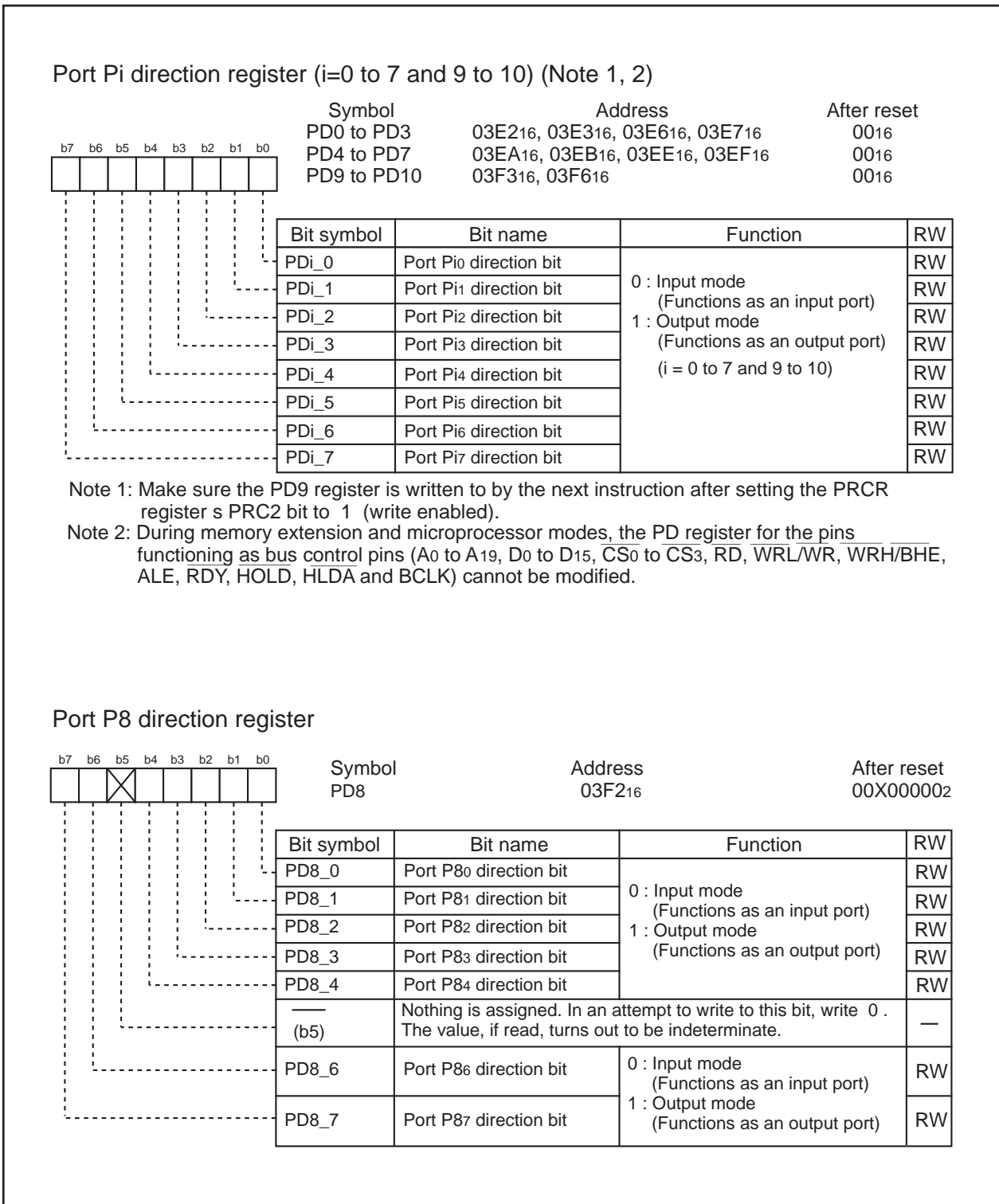


Figure 2.15.7. PD0 to PD10 Registers

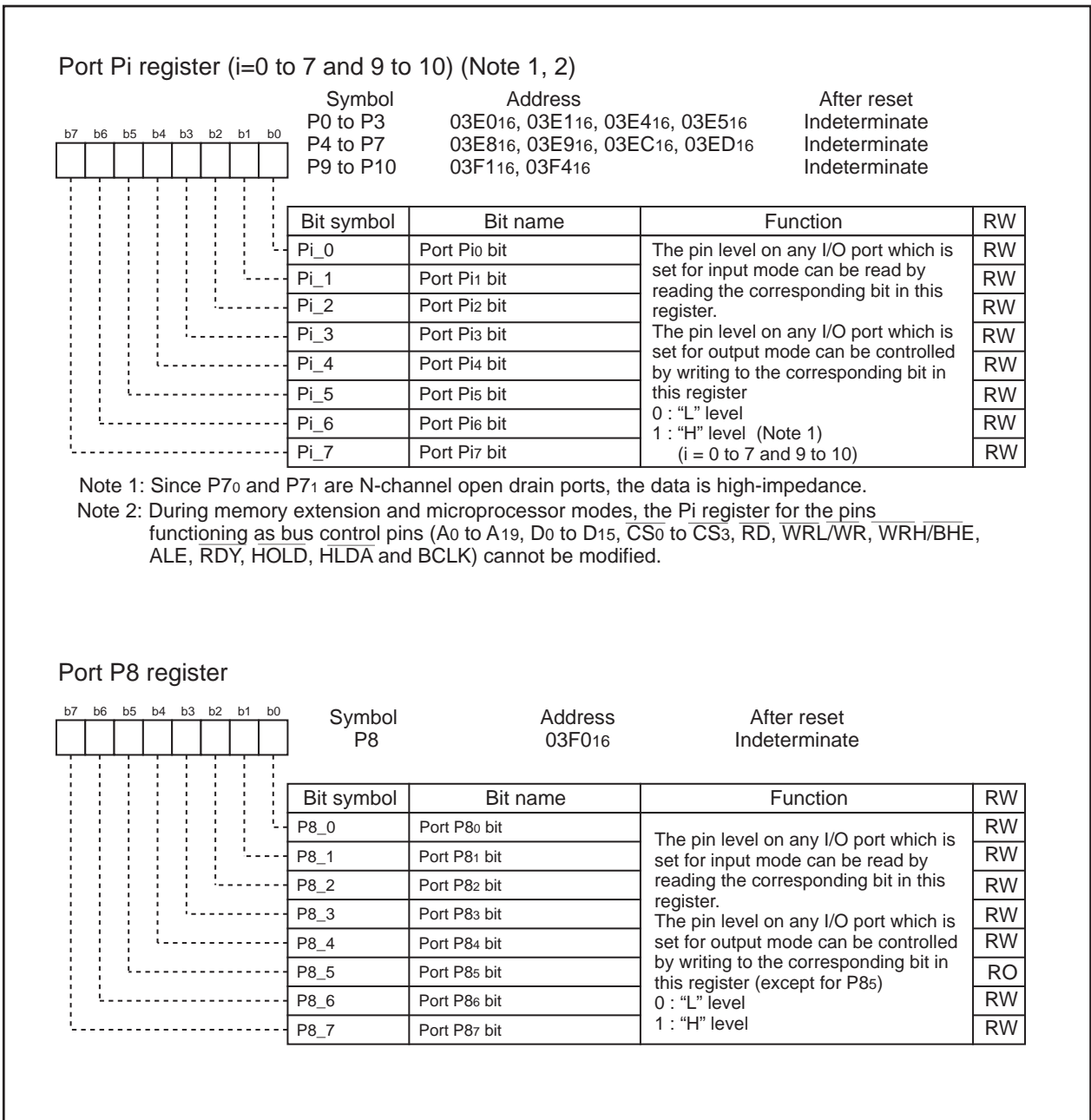


Figure 2.15.8. P0 to P10 Registers

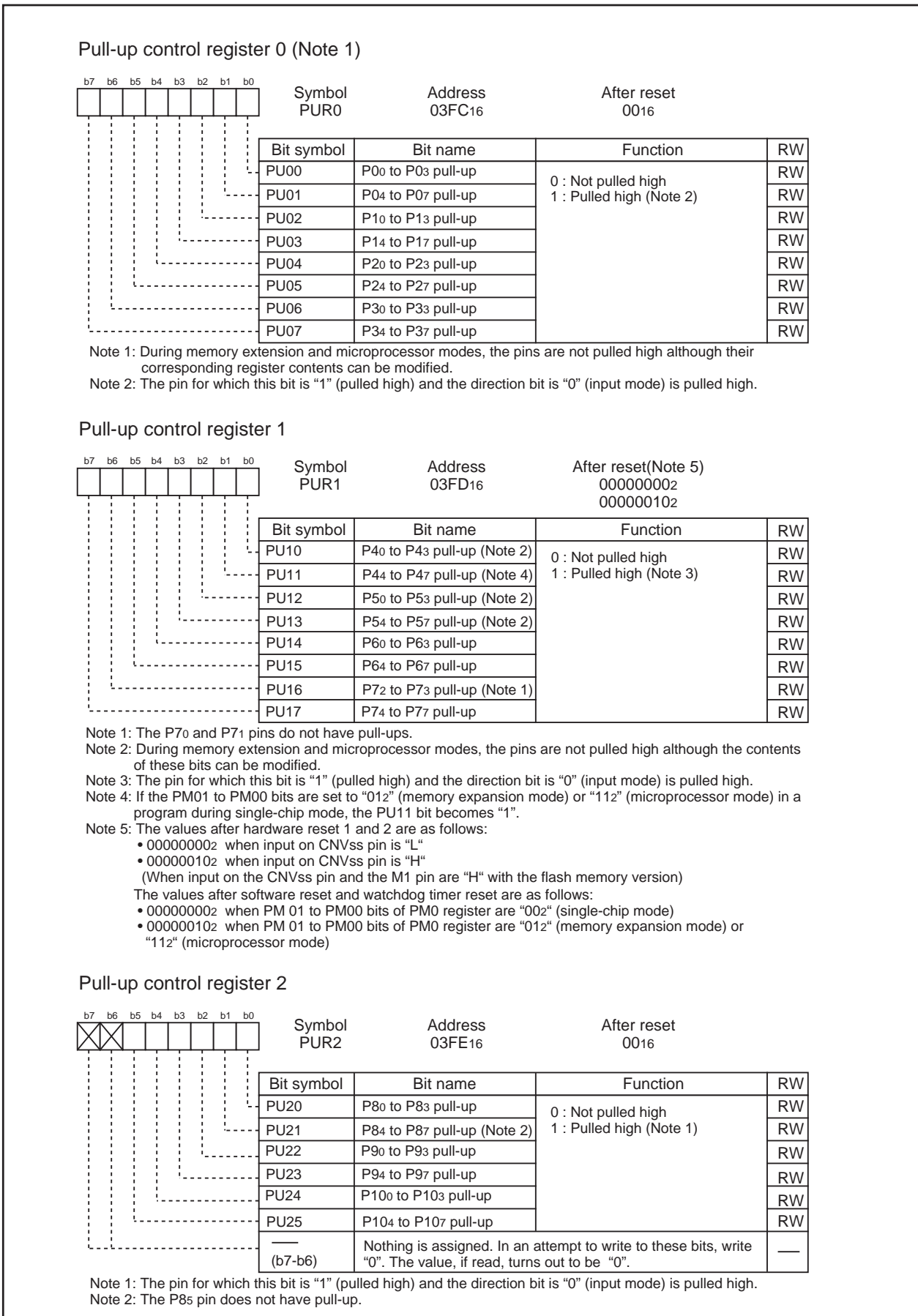


Figure 2.15.9. PUR0 to PUR2 Registers

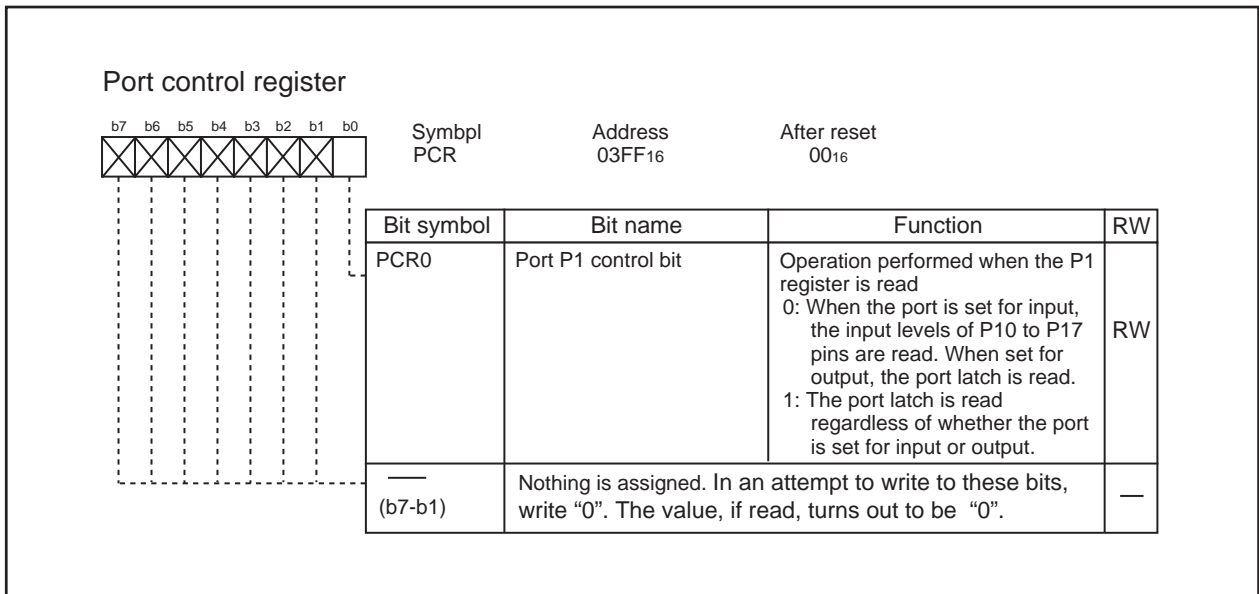


Figure 2.15.10. PCR Register

Table 2.15.1. Unassigned Pin Handling in Single-chip Mode

Pin name	Connection
Ports P0 to P7, P80 to P84, P86 to P87, P9 to P10	After setting for input mode, connect every pin to Vss via a resistor(pull-down); or after setting for output mode, leave these pins open. (Note 1, 2 ,3)
XOUT (Note 4)	Open
$\overline{\text{NMI}}$ (P85)	Connect via resistor to Vcc (pull-up)
AVcc	Connect to Vcc
AVss, VREF, BYTE	Connect to Vss

Note 1: When setting the port for output mode and leave it open, be aware that the port remains in input mode until it is switched to output mode in a program after reset. For this reason, the voltage level on the pin becomes indeterminate, causing the power supply current to increase while the port remains in input mode.

Furthermore, by considering a possibility that the contents of the direction registers could be changed by noise or noise-induced runaway, it is recommended that the contents of the direction registers be periodically reset in software, for the increased reliability of the program.

Note 2: Make sure the unused pins are processed with the shortest possible wiring from the microcomputer pins (within 2 cm).

Note 3: When the ports P70 and P71 are set for output mode, make sure a low-level signal is output from the pins. The ports P70 and P71 are N-channel open-drain outputs.

Note 4: With external clock input to XIN pin.

Table 2.15.2. Unassigned Pin Handling in Memory Expansion Mode and Microprocessor Mode

Pin name	Connection
Ports P0 to P7, P80 to P84, P86 to P87, P9 to P10	After setting for input mode, connect every pin to Vss via a resistor (pull-down); or after setting for output mode, leave these pins open. (Note 1, 2, 3, 4)
P45 / $\overline{\text{CS1}}$ to P47 / $\overline{\text{CS3}}$	Connect to Vcc via a resistor (pulled high) by setting the PD4 register's corresponding direction bit for $\overline{\text{CSi}}$ (i=1 to 3) to "0" (input mode) and the CSR register's $\overline{\text{CSi}}$ bit to "0" (chip select disabled).
$\overline{\text{BHE}}$, ALE, $\overline{\text{HLDA}}$, XOUT (Note 5), BCLK (Note 6)	Open
$\overline{\text{HOLD}}$, RDY, $\overline{\text{NMI}}$ (P85)	Connect via resistor to Vcc (pull-up)
AVcc	Connect to Vcc
AVss, VREF	Connect to Vss

Note 1: When setting the port for output mode and leave it open, be aware that the port remains in input mode until it is switched to output mode in a program after reset. For this reason, the voltage level on the pin becomes indeterminate, causing the power supply current to increase while the port remains in input mode.

Furthermore, by considering a possibility that the contents of the direction registers could be changed by noise or noise-induced runaway, it is recommended that the contents of the direction registers be periodically reset in software, for the increased reliability of the program.

Note 2: Make sure the unused pins are processed with the shortest possible wiring from the microcomputer pins (within 2 cm).

Note 3: If the CNVss pin has the Vss level applied to it, these pins are set for input ports until the processor mode is switched over in a program after reset. For this reason, the voltage levels on these pins become indeterminate, causing the power supply current to increase while they remain set for input ports.

Note 4: When the ports P70 and P71 are set for output mode, make sure a low-level signal is output from the pins. The ports P70 and P71 are N-channel open-drain outputs.

Note 5: With external clock input to XIN pin.

Note 6: If the PM07 bit in the PM0 register is set to "1" (BCLK not output), connect this pin to Vcc via a resistor (pulled high).

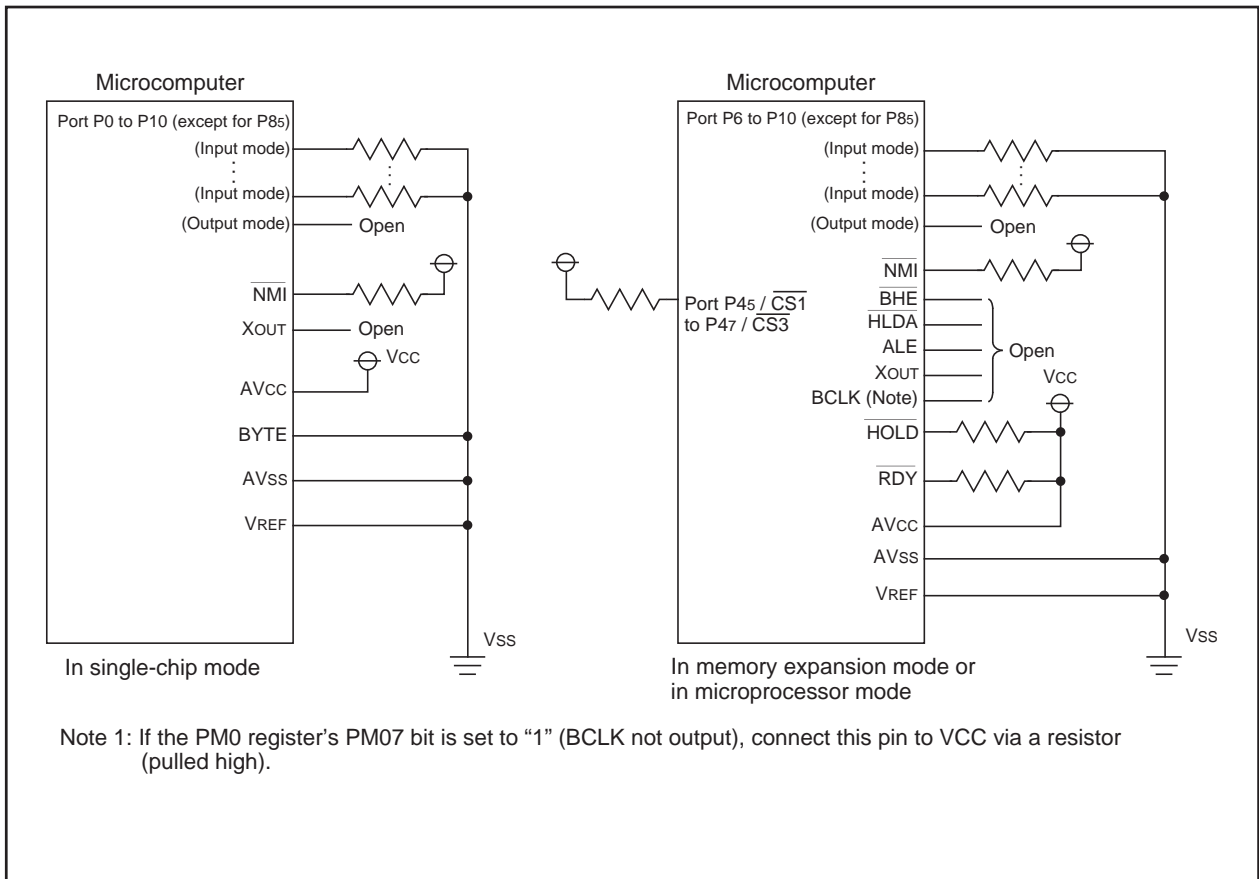


Figure 2.15.11. Unassigned Pins Handling

3. Electrical Characteristics

Table 3.1. Absolute Maximum Ratings

Symbol	Parameter		Condition	Rated value	Unit
V _{cc1} , V _{cc2}	Supply voltage		V _{cc2} =AV _{cc}	-0.3 to 6.0	V
V _{cc1}	Supply voltage		V _{cc1}	-0.3 to V _{cc2}	V
AV _{cc}	Analog supply voltage		V _{cc2} =AV _{cc}	-0.3 to 6.0	V
V _{DD2} , V _{DD3}	Analog supply voltage		V _{cc2} =V _{DD2} =V _{DD3}	-0.3 to 6.0	V
V _I	Input voltage	RESET, CNV _{ss} , BYTE, P0 ₀ to P0 ₇ , P1 ₀ to P1 ₇ , P2 ₀ to P2 ₇ , P3 ₀ to P3 ₇ , P4 ₀ to P4 ₇ , P5 ₀ to P5 ₇ , P8 ₅ to P8 ₇ , P9 ₀ to P9 ₇ , P10 ₀ to P10 ₇ , VREF, XIN, M1, START		-0.3 to V _{cc2} + 0.3	V
		P6 ₀ to P6 ₇ , P7 ₀ to P7 ₇ , P8 ₀ to P8 ₄		-0.3 to V _{cc1} + 0.3	V
		P7 ₀ , P7 ₁		-0.3 to 6.0	V
V _O	Output voltage	P0 ₀ to P0 ₇ , P1 ₀ to P1 ₇ , P2 ₀ to P2 ₇ , P3 ₀ to P3 ₇ , P4 ₀ to P4 ₇ , P5 ₀ to P5 ₇ , P8 ₆ , P8 ₇ , P9 ₀ to P9 ₇ , P10 ₀ to P10 ₇ , P11 XOUT		-0.3 to V _{cc2} + 0.3	V
		P6 ₀ to P6 ₇ , P7 ₀ to P7 ₇ , P8 ₀ to P8 ₄		-0.3 to V _{cc1} + 0.3	V
		P7 ₀ , P7 ₁		-0.3 to 6.0	V
P _d	Power dissipation		T _{opr} =25 °C	550	mW
T _{opr}	Operating ambient temperature			-20 to 70	°C
T _{stg}	Storage temperature			-20 to 125	°C

Note: Following setting is required: V_{cc1} ≤ V_{cc2}

Table 3.2. Recommended Operating Conditions (Note 1)

Symbol	Parameter		Standard			Unit
			Min.	Typ.	Max.	
V _{CC1} , V _{CC2}	Supply voltage (V _{CC1} ≤ V _{CC2})		2.0	5.0	5.5	V
AV _{CC}	Analog supply voltage			V _{CC2}		V
V _{DD2} , V _{DD3}	Analog supply voltage			V _{CC2}		V
V _{SS}	Supply voltage			0		V
AV _{SS}	Analog supply voltage			0		V
V _{IH}	HIGH input voltage	P3 ₁ to P3 ₇ , P4 ₀ to P4 ₇ , P5 ₀ to P5 ₇	0.8V _{CC2}		V _{CC2}	V
		P0 ₀ to P0 ₇ , P1 ₀ to P1 ₇ , P2 ₀ to P2 ₇ , P3 ₀ (during single-chip mode)	0.8V _{CC2}		V _{CC2}	V
		P0 ₀ to P0 ₇ , P1 ₀ to P1 ₇ , P2 ₀ to P2 ₇ , P3 ₀ (data input during memory expansion and microprocessor modes)	0.5V _{CC2}		V _{CC2}	V
		P6 ₀ to P6 ₇ , P7 ₂ to P7 ₇ , P8 ₀ to P8 ₄	0.8V _{CC1}		V _{CC1}	V
		P8 ₅ to P8 ₇ , P9 ₀ to P9 ₇ , P10 ₀ to P10 ₇ , X _{IN} , RESET, CNV _{SS} , BYTE, M1, START, TEST3	0.8V _{CC2}		V _{CC2}	V
		P7 ₀ , P7 ₁	0.8V _{CC}		5.75	V
V _{IL}	LOW input voltage	P3 ₁ to P3 ₇ , P4 ₀ to P4 ₇ , P5 ₀ to P5 ₇	0		0.2V _{CC2}	V
		P0 ₀ to P0 ₇ , P1 ₀ to P1 ₇ , P2 ₀ to P2 ₇ , P3 ₀ (during single-chip mode)	0		0.2V _{CC2}	V
		P0 ₀ to P0 ₇ , P1 ₀ to P1 ₇ , P2 ₀ to P2 ₇ , P3 ₀ (data input during memory expansion and microprocessor modes)	0		0.16V _{CC2}	V
		P6 ₀ to P6 ₇ , P7 ₀ to P7 ₇ , P8 ₀ to P8 ₄	0		0.2V _{CC1}	V
		P8 ₅ to P8 ₇ , P9 ₀ to P9 ₇ , P10 ₀ to P10 ₇ , X _{IN} , RESET, CNV _{SS} , BYTE, M1, START, TEST3	0		0.2V _{CC2}	V
V _{CVIN}	Composite video input voltage CVIN, SYNCIN			2V P-P		V
I _{OH} (peak)	HIGH peak output current (Note2, Note3)	P0 ₀ to P0 ₇ , P1 ₀ to P1 ₇ , P2 ₀ to P2 ₇ , P3 ₀ to P3 ₇ , P4 ₀ to P4 ₇ , P5 ₀ to P5 ₇ , P6 ₀ to P6 ₇ , P7 ₂ to P7 ₇ , P8 ₀ to P8 ₄ , P8 ₆ , P8 ₇ , P9 ₀ to P9 ₇ , P10 ₀ to P10 ₇ , P11			-10.0	mA
I _{OH} (avg)	HIGH average output current	P0 ₀ to P0 ₇ , P1 ₀ to P1 ₇ , P2 ₀ to P2 ₇ , P3 ₀ to P3 ₇ , P4 ₀ to P4 ₇ , P5 ₀ to P5 ₇ , P6 ₀ to P6 ₇ , P7 ₂ to P7 ₇ , P8 ₀ to P8 ₄ , P8 ₆ , P8 ₇ , P9 ₀ to P9 ₇ , P10 ₀ to P10 ₇ , P11			-5.0	mA
I _{OL} (peak)	LOW peak output current	P0 ₀ to P0 ₇ , P1 ₀ to P1 ₇ , P2 ₀ to P2 ₇ , P3 ₀ to P3 ₇ , P4 ₀ to P4 ₇ , P5 ₀ to P5 ₇ , P6 ₀ to P6 ₇ , P7 ₀ to P7 ₇ , P8 ₀ to P8 ₄ , P8 ₆ , P8 ₇ , P9 ₀ to P9 ₇ , P10 ₀ to P10 ₇ , P11			10.0	mA
I _{OL} (avg)	LOW average output current	P0 ₀ to P0 ₇ , P1 ₀ to P1 ₇ , P2 ₀ to P2 ₇ , P3 ₀ to P3 ₇ , P4 ₀ to P4 ₇ , P5 ₀ to P5 ₇ , P6 ₀ to P6 ₇ , P7 ₀ to P7 ₇ , P8 ₀ to P8 ₄ , P8 ₆ , P8 ₇ , P9 ₀ to P9 ₇ , P10 ₀ to P10 ₇ , P11			5.0	mA
f (X _{IN})	Main clock input oscillation frequency (Note 4)	V _{CC2} = 2.9 to 5.5V	0		16	MHz
f (X _{CIN})	Sub-clock oscillation frequency	V _{CC2} = 2.0 to 5.5V (Note 5)		32.768	50	kHz
f (BCLK)	CPU operation clock		0		16	MHz

Note 1: Referenced to V_{CC} = V_{CC1} = V_{CC2} = 2.0 to 5.5V at T_{opr} = -20 to 70 °C unless otherwise specified.

When operating in microprocessor and memory expansion mode, use this device under the conditions of V_{CC} = V_{CC1} = V_{CC2} = 4.5 to 5.5V at T_{opr} = -20 to 70 °C

(If V_{CC1} and V_{CC2} are less than 4.0V, it cannot be used.)

Note 2: The mean output current is the mean value within 100ms.

Note 3: The total I_{OL} (peak) for ports P0, P1, P2, P3, P4, P5, P8₆, P8₇, P9, P10 and P11 must be 80mA max. The total I_{OL} (peak) for ports P6, P7 and P8₀ to P8₄ must be 80mA max. The total I_{OH} (peak) for ports P0, P1, and P2 must be -40mA max. The total I_{OH} (peak) for ports P3, P4 and P5 must be -40mA max. The total I_{OH} (peak) for ports P6, P7, and P8₀ to P8₄ must be -40mA max. The total I_{OH} (peak) for ports P8₆, P8₇, P9, P10 and P11 must be -40mA max.

Note 4: Use the V_{CC1} and V_{CC2} power supply voltage on the following conditions.

- V_{CC1} = 3.00V to V_{CC2}, V_{CC2} = 4.00V to 5.5V (at f(X_{IN}) = 16MHz)
- V_{CC1} = 2.90V to V_{CC2}, V_{CC2} = 2.90V to 5.5V (at f(X_{IN}) = 16MHz, at divide-by-8 or 16)

Note 5: Use in low power dissipation mode. When operating on low voltage (V_{CC} = 3.0V), only single-chip mode can be used.

If the V_{CC2} supply voltage is less than 2.6 V, be aware that only the CPU, RAM, clock timer, interrupt, and Input/Output ports can be used. Other control circuits (e.g., timers A and B, serial I/O, UART) cannot be used.

Table 3.3. A-D Conversion Characteristics (Note 1)

Symbol	Parameter	Measuring condition	Standard			Unit	
			Min.	Typ.	Max.		
—	Resolution	$V_{REF} = V_{CC}$			8	Bits	
—	Absolute accuracy	$V_{REF} = V_{CC} = 5V$	AN0 to AN7 input			± 3	LSB
			ANEX0, ANEX1 input External operation amp			± 4	LSB
RLADDER	Ladder resistance	$V_{REF} = V_{CC}$	10		40	k Ω	
tCONV	Conversion time(8bit), Sample & hold function available	$V_{REF} = V_{CC} = 5V, \phi_{AD} = 10MHz$	2.8			μs	
tsAMP	Sampling time		0.3			μs	
VREF	Reference voltage		4.5		V_{CC}	V	
VIA	Analog input voltage		0		V_{REF}	V	

Note 1: Referenced to $V_{CC2} = AV_{CC} = V_{REF} = 4.5$ to 5.5 V, $V_{SS} = AV_{SS} = 0V$ at $T_{opr} = -20$ to 70 °C unless otherwise specified.

Note 2: AD operation clock frequency (ϕ_{AD} frequency) must be 10 MHz or less.

Note 3: A case without sample & hold function turn ϕ_{AD} frequency into 250 kHz or more.

A case with sample & hold function turn ϕ_{AD} frequency into 1 MHz or more.

Table 3.4. Flash Memory Version Electrical Characteristics (Note 1)

Symbol	Parameter	Measuring condition	Standard			Unit
			Min.	Typ.	Max	
—	Word program time			30	200	μs
—	Block erase time			1	4	s
—	Lock bit program time			30	200	μs
tps	Flash memory circuit stabilization wait time				15	μs

Note 1: Referenced to $V_{CC2} = 4.75$ to 5.25 V at $T_{opr} = 0$ to 60 °C unless otherwise specified.

Note 2: n denotes the number of block erases.

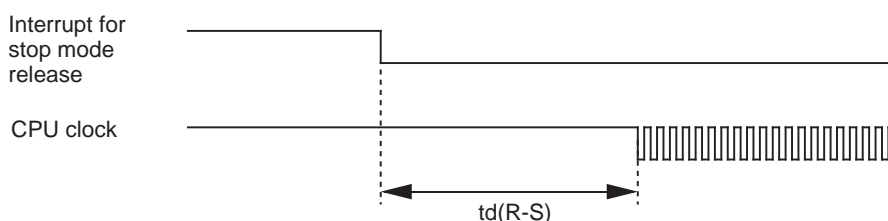
Table 3.5. Flash Memory Version Program/Erase Voltage and Read Operation Voltage Characteristics (at $T_{opr} = 0$ to 60 °C)

Flash program, erase voltage	Flash read operation voltage
$V_{CC2} = 5.0 \pm 0.25$ V	$V_{CC2} = 2.0$ to 5.5 V

Table 3.6. Power Supply Circuit Timing Characteristics

Symbol	Parameter	Measuring condition	Standard			Unit
			Min.	Typ.	ax.	
td(P-R)	Time for internal power supply stabilization during powering-on	$V_{CC} = 5.0V$			2	ms
td(R-S)	STOP release time				150	μs
td(W-S)	Low power dissipation mode wait mode release time				150	μs
td(M-L)	Time for internal power supply stabilization when main clock oscillation starts (Note)				50	μs

Note : At XIN-XOUT generation.



$$V_{CC1} = V_{CC2} = 5V$$

Table 3.7. Electrical Characteristics (1) (Note 1)

Symbol	Parameter		Measuring condition	Standard			Unit	
				Min	Typ.	Max.		
V _{OH}	HIGH output voltage	P0 ₀ to P0 ₇ , P1 ₀ to P1 ₇ , P2 ₀ to P2 ₇ , P3 ₀ to P3 ₇ , P4 ₀ to P4 ₇ , P5 ₀ to P5 ₇ , P8 ₆ , P8 ₇ , P10 ₀ to P10 ₇ , P11	I _{OH} =-5mA	V _{CC2} -2.0		V _{CC2}	V	
		P6 ₀ to P6 ₇ , P7 ₂ to P7 ₇ , P8 ₀ to P8 ₄	I _{OH} =-5mA	V _{CC1} -2.0		V _{CC1}	V	
V _{OH}	HIGH output voltage	P0 ₀ to P0 ₇ , P1 ₀ to P1 ₇ , P2 ₀ to P2 ₇ , P3 ₀ to P3 ₇ , P4 ₀ to P4 ₇ , P5 ₀ to P5 ₇ , P8 ₆ , P8 ₇ , P9 ₀ to P9 ₇ , P10 ₀ to P10 ₇ , P11	I _{OH} =-200μA	V _{CC2} -0.3		V _{CC2}	V	
		P6 ₀ to P6 ₇ , P7 ₂ to P7 ₇ , P8 ₀ to P8 ₄	I _{OH} =-200μA	V _{CC1} -0.3		V _{CC1}	V	
V _{OH}	HIGH output voltage	LP2 to LP4	V _{CC} =4.5V, I _{OH} =-0.05mA	3.75			V	
V _{OH}	HIGH output voltage	X _{OUT}	HIGHPOWER	I _{OH} =-1mA	V _{CC2} -2.0		V _{CC2}	V
			LOWPOWER	I _{OH} =-0.5mA	V _{CC2} -2.0		V _{CC2}	V
	HIGH output voltage	X _{COU} T	HIGHPOWER	With no load applied		2.5		V
			LOWPOWER	With no load applied		1.6		V
V _{OL}	LOW output voltage	P0 ₀ to P0 ₇ , P1 ₀ to P1 ₇ , P2 ₀ to P2 ₇ , P3 ₀ to P3 ₇ , P4 ₀ to P4 ₇ , P5 ₀ to P5 ₇ , P8 ₆ , P8 ₇ , P9 ₀ to P9 ₇ , P10 ₀ to P10 ₇ , P11	I _{OL} =5mA			2.0	V	
		P6 ₀ to P6 ₇ , P7 ₀ to P7 ₇ , P8 ₀ to P8 ₄	I _{OL} =5mA			2.0	V	
V _{OL}	LOW output voltage	P0 ₀ to P0 ₇ , P1 ₀ to P1 ₇ , P2 ₀ to P2 ₇ , P3 ₀ to P3 ₇ , P4 ₀ to P4 ₇ , P5 ₀ to P5 ₇ , P8 ₆ , P8 ₇ , P9 ₀ to P9 ₇ , P10 ₀ to P10 ₇ , P11	I _{OL} =200μA			0.45	V	
		P6 ₀ to P6 ₇ , P7 ₀ to P7 ₇ , P8 ₀ to P8 ₄	I _{OL} =200μA			0.45	V	
V _{OL}	LOW output voltage	LP2 to LP4	V _{CC} =4.5V, I _{OL} =0.05mA			0.4	V	
V _{OL}	LOW output voltage	X _{OUT}	HIGHPOWER	I _{OL} =1mA		2.0	V	
			LOWPOWER	I _{OL} =0.5mA		2.0	V	
	LOW output voltage	X _{COU} T	HIGHPOWER	With no load applied		0		V
			LOWPOWER	With no load applied		0		V
V _{T+} -V _{T-}	Hysteresis	HOLD, RDY, TA0 _{IN} to TA4 _{IN} , TB0 _{IN} to TB5 _{IN} , INT0 to INT5, NMI, ADTRG, CTS0 to CTS2, SCL, SDA, CLK0 to CLK4, TA2 _{OUT} to TA4 _{OUT} , K10 to K13, RxD0 to RxD2, S1N3, S1N4		0.2		1.0	V	
V _{T+} -V _{T-}	Hysteresis	RESET		0.2		2.2	V	
I _{IH}	HIGH input current	P0 ₀ to P0 ₇ , P1 ₀ to P1 ₇ , P2 ₀ to P2 ₇ , P3 ₀ to P3 ₇ , P4 ₀ to P4 ₇ , P5 ₀ to P5 ₇ , P6 ₀ to P6 ₇ , P7 ₀ to P7 ₇ , P8 ₀ to P8 ₇ , P9 ₀ to P9 ₇ , P10 ₀ to P10 ₇ , X _{IN} , RESET, CNV _{SS} , BYTE, M1, START	V _I =5V			5.0	μA	
I _{IL}	LOW input current	P0 ₀ to P0 ₇ , P1 ₀ to P1 ₇ , P2 ₀ to P2 ₇ , P3 ₀ to P3 ₇ , P4 ₀ to P4 ₇ , P5 ₀ to P5 ₇ , P6 ₀ to P6 ₇ , P7 ₀ to P7 ₇ , P8 ₀ to P8 ₇ , P9 ₀ to P9 ₇ , P10 ₀ to P10 ₇ , X _{IN} , RESET, CNV _{SS} , BYTE, M1, START	V _I =0V			-5.0	μA	
R _{PULLUP}	Pull-up resistance	P0 ₀ to P0 ₇ , P1 ₀ to P1 ₇ , P2 ₀ to P2 ₇ , P3 ₀ to P3 ₇ , P4 ₀ to P4 ₇ , P5 ₀ to P5 ₇ , P6 ₀ to P6 ₇ , P7 ₂ to P7 ₇ , P8 ₀ to P8 ₄ , P8 ₆ , P8 ₇ , P9 ₀ to P9 ₇ , P10 ₀ to P10 ₇	V _I =0V	30	50	170	kΩ	
R _{FXIN}	Feedback resistance	X _{IN}			1.5		MΩ	
R _{FXCIN}	Feedback resistance	X _{CIN}			15		MΩ	
V _{RAM}	RAM retention voltage		Stop mode	2.0			V	
V _{SYNCIN}	Sync voltage amplitude			0.3	0.6	1.2	V	
V _{dat(text)}	Teletext data voltage amplitude			0.6	0.9	1.4	V	
f _H	Horizontal synchronous signal frequency			14.6	15.625	17.0	kHz	

Note 1: Referenced to V_{CC}=V_{CC1}=V_{CC2}=4.50 to 5.50 V, V_{SS}=0V at T_{opr} = -20 to 70 °C, f(BCLK)=16MHz unless otherwise specified.

$$V_{CC1} = V_{CC2} = 3V$$

Table 3.8. Electrical Characteristics (2) (Note)

Symbol	Parameter		Measuring condition	Standard			Unit
				Min.	Typ.		
V _{OH}	HIGH output voltage	P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P72 to P77, P80 to P84, P86, P87, P90 to P97, P100 to P107, P11	I _{OH} =-1mA	V _{CC} -0.5		V _{CC}	V
V _{OH}	HIGH output voltage	X _{OUT}	HIGHPOWER	I _{OH} =-0.1mA	V _{CC2} -0.5	V _{CC2}	V
			LOWPOWER	I _{OH} =-50μA	V _{CC2} -0.5	V _{CC2}	V
	HIGH output voltage	X _{COUT}	HIGHPOWER	With no load applied		2.5	V
			LOWPOWER	With no load applied		1.6	V
V _{OL}	LOW output voltage	P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P70 to P77, P80 to P84, P86, P87, P90 to P97, P100 to P107, P11	I _{OL} =1mA			0.5	V
V _{OL}	LOW output voltage	X _{OUT}	HIGHPOWER	I _{OL} =-0.1mA		0.5	V
			LOWPOWER	I _{OL} =-50μA		0.5	V
	LOW output voltage	X _{COUT}	HIGHPOWER	With no load applied		0	V
			LOWPOWER	With no load applied		0	V
V _{T+} -V _{T-}	Hysteresis	TA0IN to TA4IN, TB0IN to TB5IN, INT0 to INT5, NMI, CLK0 to CLK4, TA2OUT to TA4OUT, K10 to K15		0.2		0.8	V
V _{T+} -V _{T-}	Hysteresis	RESET		0.2	(0.7)	1.8	V
I _{IH}	HIGH input current	P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P70 to P77, P80 to P87, P90 to P97, P100 to P107, X _{IN} , RESET, CNVss, BYTE, M1, START	V _I =3V			4.0	μA
I _{IL}	LOW input current	P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P70 to P77, P80 to P87, P90 to P97, P100 to P107, X _{IN} , RESET, CNVss, BYTE, M1, START	V _I =0V			-4.0	μA
R _{PULLUP}	Pull-up resistance	P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P72 to P77, P80 to P84, P86, P87, P90 to P97, P100 to P107,	V _I =0V	50	100	500	kΩ
R _{I_{XIN}}	Feedback resistance	X _{IN}			3.0		MΩ
R _{I_XCIN}	Feedback resistance	X _{CIN}			25		MΩ

Note : Referenced to V_{CC}=V_{CC1}=V_{CC2}=3.0V, V_{SS}=0V at T_{opr} = -20 to 70 °C, f(X_{CIN})=32kHz unless otherwise specified.
Use in single-chip mode and low power dissipation mode.

Table 3.9. Electrical Characteristics (2) (Note 1)

Symbol	Parameter		Measuring condition		Standard			Unit
					Min.	Typ.	Max.	
I _{cc}	Power supply current	In single-chip mode, the output pins are open and other pins are V _{SS}	Mask ROM	f(BCLK)=16MHz, V _{CC} =5.0V		50	100	mA
			Flash memory	f(BCLK)=16MHz, V _{CC} =5.0V		50	100	mA
			Flash memory Program	f(BCLK)=16MHz, V _{CC} =5.0V		15		mA
			Flash memory Erase	f(BCLK)=16MHz, V _{CC} =5.0V		25		mA
			Mask ROM	f(XCIN)=32kHz, Low power dissipation mode, ROM(Note 3), (Note4) V _{CC} =5.0V		25		μA
			Flash memory	f(BCLK)=32kHz, Low power dissipation mode, RAM(Note 3), (Note4) V _{CC} =5.0V		25		μA
				f(BCLK)=32kHz, Low power dissipation mode, Flash memory(Note 3), (Note4) V _{CC} =5.0V		420		μA
			Mask ROM Flash memory	f(BCLK)=32kHz, Wait mode (Note 2), (Note4) Oscillation capacity High		7.5		μA
				f(BCLK)=32kHz, Wait mode(Note 2), (Note4) Oscillation capacity Low V _{CC} =5.0V		5.0	10.0	μA
				f(BCLK)=32kHz, Wait mode (Note 2), (Note4) Oscillation capacity High V _{CC} =3.0V		6.0		μA
f(BCLK)=32kHz, Wait mode(Note 2), (Note4) Oscillation capacity Low V _{CC} =3.0V		2.0		8.0	μA			
	Stop mode, (Note4) T _{opr} =25°C V _{CC} =5.0V		0.8	5.0	μA			

Note 1: Referenced to V_{CC1}=V_{CC2}= 5V, V_{SS}=0V at T_{opr} = 25 °C, f(BCLK)=16MHz unless otherwise specified.

Note 2: With one timer operated using fc32. (Slicer operation OFF)

Note 3: This indicates the memory in which the program to be executed exists.

Note 4: • All of V_{DD2} and V_{DD3} are at the same potential level as V_{CC2}.

- Extension registers (addresses 0016 through 3F16) are set to the initial state.
- Inputs to the SYNCIN and CVIN1 pins are disabled.
- For current consumption reducing, set the level of V_{SS} or V_{CC} to the ports used in input mode.

Table 3.10 Video signal input conditions (Note 1)

Symbol	Parameter	Measuring condition	Standard			Unit
			Min	Typ.	Max.	
V _{IN-cu}	Composite video signal input clamp voltage	Sync-chip voltage		1.0		V

Note 1: Referenced to V_{CC2} = 5.0 V at T_{opr} = -20 to 70 °C unless otherwise specified.

$$V_{CC1} = V_{CC2} = 5V$$

Timing Requirements

($V_{CC1} = V_{CC2} = 5V$, $V_{SS} = 0V$, at $T_{opr} = -20$ to $70^{\circ}C$ unless otherwise specified)

Table 3.11. External Clock Input (XIN 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	30		ns
t _{w(L)}	External clock input LOW pulse width	30		ns
t _r	External clock rise time		15	ns
t _f	External clock fall time		15	ns

Table 3.12. Memory Expansion Mode and Microprocessor Mode

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t _{ac1(RD-DB)}	Data input access time (for setting with no wait)		(Note 1)	ns
t _{ac2(RD-DB)}	Data input access time (for setting with wait)		(Note 2)	ns
t _{ac3(RD-DB)}	Data input access time (when accessing multiplex bus area)		(Note 3)	ns
t _{su(DB-RD)}	Data input setup time	40		ns
t _{su(RDY-BCLK)}	RDY input setup time	30		ns
t _{su(HOLD-BCLK)}	HOLD input setup time	40		ns
t _{h(RD-DB)}	Data input hold time	0		ns
t _{h(BCLK-RDY)}	RDY input hold time	0		ns
t _{h(BCLK-HOLD)}	HOLD input hold time	0		ns
t _{d(BCLK-HLDA)}	HLDA output delay time		40	ns

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

$$\frac{0.5 \times 10^9}{f(\text{BCLK})} - 45 \quad [\text{ns}]$$

Note 2: Calculated according to the BCLK frequency as follows:

$$\frac{(n-0.5) \times 10^9}{f(\text{BCLK})} - 45 \quad [\text{ns}] \quad n \text{ is "2" for 1-wait setting, "3" for 2-wait setting and "4" for 3-wait setting.}$$

Note 3: Calculated according to the BCLK frequency as follows:

$$\frac{(n-0.5) \times 10^9}{f(\text{BCLK})} - 45 \quad [\text{ns}] \quad n \text{ is "2" for 2-wait setting, "3" for 3-wait setting.}$$

Table 3.13. Remote Control Pulse Input

Symbol	Parameter	Standard		Unit
		Min.	Max.	
T _{w(RMTH)}	RMTIN input HIGH pulse width	61		μs
T _{w(RMTL)}	RMTIN input LOW pulse width	61		μs

Table 3.14. JUST CLOCK Input

Symbol	Parameter	Standard		Unit
		Min.	Max.	
T _{w(JSTH)}	JSTIN input HIGH pulse width	61		μs
T _{w(JSTL)}	JSTIN input LOW pulse width	61		μs

$$V_{CC1} = V_{CC2} = 5V$$

Timing Requirements

($V_{CC1} = V_{CC2} = 5V$, $V_{SS} = 0V$, at $T_{opr} = -20$ to $70^{\circ}C$ unless otherwise specified)

Table 3.15. Timer A Input (Counter Input in Event Counter Mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(TA)}$	TAi IN input cycle time	100		ns
$t_{w(TAH)}$	TAi IN input HIGH pulse width	40		ns
$t_{w(TAL)}$	TAi IN input LOW pulse width	40		ns

Table 3.16. Timer A Input (Gating Input in Timer Mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(TA)}$	TAi IN input cycle time	400		ns
$t_{w(TAH)}$	TAi IN input HIGH pulse width	200		ns
$t_{w(TAL)}$	TAi IN input LOW pulse width	200		ns

Table 3.17. Timer A Input (External Trigger Input in One-shot Timer Mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(TA)}$	TAi IN input cycle time	200		ns
$t_{w(TAH)}$	TAi IN input HIGH pulse width	100		ns
$t_{w(TAL)}$	TAi IN input LOW pulse width	100		ns

Table 3.18. Timer A Input (External Trigger Input in Pulse Width Modulation Mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{w(TAH)}$	TAi IN input HIGH pulse width	100		ns
$t_{w(TAL)}$	TAi IN input LOW pulse width	100		ns

Table 3.19. Timer A Input (Counter Increment/decrement Input in Event Counter Mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(UP)}$	TAi OUT input cycle time	2000		ns
$t_{w(UPH)}$	TAi OUT input HIGH pulse width	1000		ns
$t_{w(UPL)}$	TAi OUT input LOW pulse width	1000		ns
$t_{su(UP-TIN)}$	TAi OUT input setup time	400		ns
$t_h(TIN-UP)$	TAi OUT input hold time	400		ns

$$V_{CC1} = V_{CC2} = 5V$$

Timing Requirements

($V_{CC1} = V_{CC2} = 5V$, $V_{SS} = 0V$, at $T_{opr} = -20$ to $70^{\circ}C$ unless otherwise specified)

Table 3.20. Timer B Input (Counter Input in Event Counter Mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(TB)}$	TBiIN input cycle time (counted on one edge)	100		ns
$t_{w(TBH)}$	TBiIN input HIGH pulse width (counted on one edge)	40		ns
$t_{w(TBL)}$	TBiIN input LOW pulse width (counted on one edge)	40		ns
$t_{c(TB)}$	TBiIN input cycle time (counted on both edges)	200		ns
$t_{w(TBH)}$	TBiIN input HIGH pulse width (counted on both edges)	80		ns
$t_{w(TBL)}$	TBiIN input LOW pulse width (counted on both edges)	80		ns

Table 3.21. Timer B Input (Pulse Period Measurement Mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(TB)}$	TBiIN input cycle time	400		ns
$t_{w(TBH)}$	TBiIN input HIGH pulse width	200		ns
$t_{w(TBL)}$	TBiIN input LOW pulse width	200		ns

Table 3.22. 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 3.23. A-D Trigger Input

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(AD)}$	ADTRG input cycle time (trigger able minimum)	1000		ns
$t_{w(ADL)}$	ADTRG input LOW pulse width	125		ns

Table 3.24. Serial I/O

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(CK)}$	CLKi input cycle time	200		ns
$t_{w(CKH)}$	CLKi input HIGH pulse width	100		ns
$t_{w(CKL)}$	CLKi input LOW pulse width	100		ns
$t_{d(C-Q)}$	TxDi output delay time		80	ns
$t_{h(C-Q)}$	TxDi hold time	0		ns
$t_{su(D-C)}$	RxDi input setup time	30		ns
$t_{h(C-D)}$	RxDi input hold time	90		ns

Table 3.25. External Interrupt \overline{INTi} Input

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

$$V_{CC1} = V_{CC2} = 3V$$

Timing Requirements

($V_{CC1} = V_{CC2} = 3V$, $V_{SS} = 0V$, at $T_{opr} = -20$ to $70^{\circ}C$ unless otherwise specified)

Table 3.26. External Clock Input (XIN input)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t_c	External clock input cycle time	100		ns
$t_w(H)$	External clock input HIGH pulse width	40		ns
$t_w(L)$	External clock input LOW pulse width	40		ns
t_r	External clock rise time		18	ns
t_f	External clock fall time		18	ns

Table 3.27. Memory Expansion Mode and Microprocessor Mode

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{ac1}(RD-DB)$	Data input access time (for setting with no wait)		(Note 1)	ns
$t_{ac2}(RD-DB)$	Data input access time (for setting with wait)		(Note 2)	ns
$t_{ac3}(RD-DB)$	Data input access time (when accessing multiplex bus area)		(Note 3)	ns
$t_{su}(DB-RD)$	Data input setup time	50		ns
$t_{su}(RDY-BCLK)$	RDY input setup time	40		ns
$t_{su}(HOLD-BCLK)$	HOLD input setup time	50		ns
$t_h(RD-DB)$	Data input hold time	0		ns
$t_h(BCLK-RDY)$	RDY input hold time	0		ns
$t_h(BCLK-HOLD)$	HOLD input hold time	0		ns
$t_d(BCLK-HLDA)$	HLDA output delay time		40	ns

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

$$\frac{0.5 \times 10^9}{f(BCLK)} - 60 \quad [ns]$$

Note 2: Calculated according to the BCLK frequency as follows:

$$\frac{(n-0.5) \times 10^9}{f(BCLK)} - 60 \quad [ns] \quad n \text{ is "2" for 1-wait setting, "3" for 2-wait setting and "4" for 3-wait setting.}$$

Note 3: Calculated according to the BCLK frequency as follows:

$$\frac{(n-0.5) \times 10^9}{f(BCLK)} - 60 \quad [ns] \quad n \text{ is "2" for 2-wait setting, "3" for 3-wait setting.}$$

$$V_{CC1} = V_{CC2} = 3V$$

Timing Requirements

($V_{CC1} = V_{CC2} = 3V$, $V_{SS} = 0V$, at $T_{opr} = -20$ to $70^{\circ}C$ unless otherwise specified)

Table 3.28. Timer A Input (Counter Input in Event Counter Mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(TA)}$	TAiIN input cycle time	150		ns
$t_{w(TAH)}$	TAiIN input HIGH pulse width	60		ns
$t_{w(TAL)}$	TAiIN input LOW pulse width	60		ns

Table 3.29. Timer A Input (Gating Input in Timer Mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(TA)}$	TAiIN input cycle time	600		ns
$t_{w(TAH)}$	TAiIN input HIGH pulse width	300		ns
$t_{w(TAL)}$	TAiIN input LOW pulse width	300		ns

Table 3.30. Timer A Input (External Trigger Input in One-shot Timer Mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(TA)}$	TAiIN input cycle time	300		ns
$t_{w(TAH)}$	TAiIN input HIGH pulse width	150		ns
$t_{w(TAL)}$	TAiIN input LOW pulse width	150		ns

Table 3.31. Timer A Input (External Trigger Input in Pulse Width Modulation Mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{w(TAH)}$	TAiIN input HIGH pulse width	150		ns
$t_{w(TAL)}$	TAiIN input LOW pulse width	150		ns

Table 3.32. Timer A Input (Counter Increment/decrement Input in Event Counter Mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(UP)}$	TAiOUT input cycle time	3000		ns
$t_{w(UPH)}$	TAiOUT input HIGH pulse width	1500		ns
$t_{w(UPL)}$	TAiOUT input LOW pulse width	1500		ns
$t_{su(UP-TIN)}$	TAiOUT input setup time	600		ns
$t_{h(TIN-UP)}$	TAiOUT input hold time	600		ns

$$V_{CC1} = V_{CC2} = 3V$$

Timing Requirements

($V_{CC1} = V_{CC2} = 3V$, $V_{SS} = 0V$, at $T_{opr} = -20$ to $70^{\circ}C$ unless otherwise specified)

Table 3.33. Timer B Input (Counter Input in Event Counter Mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_c(TB)$	TBiIN input cycle time (counted on one edge)	150		ns
$t_w(TBH)$	TBiIN input HIGH pulse width (counted on one edge)	60		ns
$t_w(TBL)$	TBiIN input LOW pulse width (counted on one edge)	60		ns
$t_c(TB)$	TBiIN input cycle time (counted on both edges)	300		ns
$t_w(TBH)$	TBiIN input HIGH pulse width (counted on both edges)	120		ns
$t_w(TBL)$	TBiIN input LOW pulse width (counted on both edges)	120		ns

Table 3.34. Timer B Input (Pulse Period Measurement Mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_c(TB)$	TBiIN input cycle time	600		ns
$t_w(TBH)$	TBiIN input HIGH pulse width	300		ns
$t_w(TBL)$	TBiIN input LOW pulse width	300		ns

Table 3.35. Timer B Input (Pulse Width Measurement Mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_c(TB)$	TBiIN input cycle time	600		ns
$t_w(TBH)$	TBiIN input HIGH pulse width	300		ns
$t_w(TBL)$	TBiIN input LOW pulse width	300		ns

Table 3.36. Serial I/O

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_c(CK)$	CLKi input cycle time	300		ns
$t_w(CKH)$	CLKi input HIGH pulse width	150		ns
$t_w(CKL)$	CLKi input LOW pulse width	150		ns
$t_d(C-Q)$	TxDi output delay time		160	ns
$t_h(C-Q)$	TxDi hold time	0		ns
$t_{su}(D-C)$	RxDi input setup time	70		ns
$t_h(C-D)$	RxDi input hold time	90		ns

Table 3.37. External Interrupt \overline{INTi} Input

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_w(INH)$	\overline{INTi} input HIGH pulse width	380		ns
$t_w(INL)$	\overline{INTi} input LOW pulse width	380		ns

$$V_{CC1} = V_{CC2} = 5V$$

Switching Characteristics

($V_{CC1} = V_{CC2} = 5V$, $V_{SS} = 0V$, at $T_{opr} = -20$ to $70^{\circ}C$ unless otherwise specified)

Table 3.38. Memory Expansion and Microprocessor Modes (for setting with no wait)

Symbol	Parameter	Measuring condition	Standard		Unit
			Min.	Max.	
$t_{d(BCLK-AD)}$	Address output delay time	Figure 3.1		28	ns
$t_{h(BCLK-AD)}$	Address output hold time (refers to BCLK)		4		ns
$t_{h(RD-AD)}$	Address output hold time (refers to RD)		0		ns
$t_{h(WR-AD)}$	Address output hold time (refers to WR)		(Note 2)		ns
$t_{d(BCLK-CS)}$	Chip select output delay time			28	ns
$t_{h(BCLK-CS)}$	Chip select output hold time (refers to BCLK)		4		ns
$t_{d(BCLK-ALE)}$	ALE signal output delay time			28	ns
$t_{h(BCLK-ALE)}$	ALE signal output hold time		-4		ns
$t_{d(BCLK-RD)}$	RD signal output delay time			28	ns
$t_{h(BCLK-RD)}$	RD signal output hold time		0		ns
$t_{d(BCLK-WR)}$	WR signal output delay time			28	ns
$t_{h(BCLK-WR)}$	WR signal output hold time		0		ns
$t_{d(BCLK-DB)}$	Data output delay time (refers to BCLK)			40	ns
$t_{h(BCLK-DB)}$	Data output hold time (refers to BCLK)(Note 3)		4		ns
$t_{d(DB-WR)}$	Data output delay time (refers to WR)		(Note 1)		ns
$t_{h(WR-DB)}$	Data output hold time (refers to WR)(Note 3)		(Note 2)		ns

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

$$\frac{0.5 \times 10^9}{f(BCLK)} - 40 \quad [ns]$$

Note 2: Calculated according to the BCLK frequency as follows:

$$\frac{0.5 \times 10^9}{f(BCLK)} - 10 \quad [ns]$$

Note 3: This standard value shows the timing when the output is off, and does not show hold time of data bus.

Hold time of data bus varies with capacitor volume and pull-up (pull-down) resistance value.

Hold time of data bus is expressed in $t = -CR \times \ln(1 - V_{OL} / V_{CC2})$

by a circuit of the right figure.

For example, when $V_{OL} = 0.2V_{CC2}$, $C = 30pF$, $R = 1k\Omega$, hold time of output "L" level is

$$t = -30pF \times 1k\Omega \times \ln(1 - 0.2V_{CC2} / V_{CC2}) = 6.7ns.$$

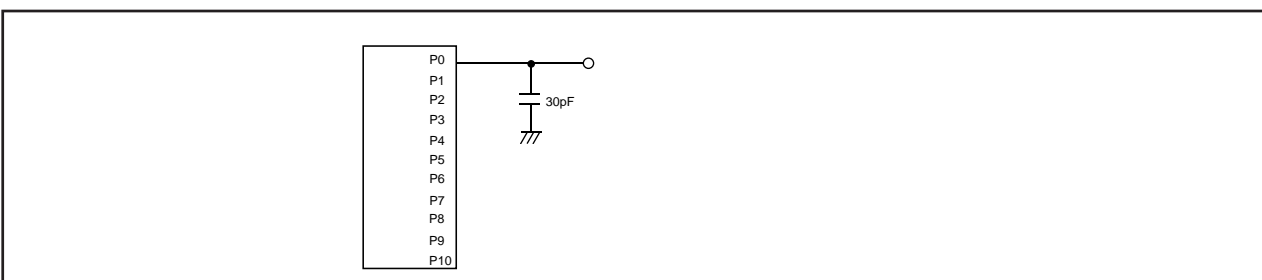
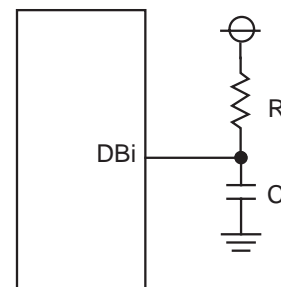


Figure 3.1. Ports P0 to P10 Measurement Circuit

$$V_{CC1} = V_{CC2} = 5V$$

Switching Characteristics

($V_{CC1} = V_{CC2} = 5V$, $V_{SS} = 0V$, at $T_{opr} = -20$ to $70^{\circ}C$ unless otherwise specified)

Table 3.39. Memory Expansion and Microprocessor Modes
(for 1- to 3-wait setting and external area access)

Symbol	Parameter	Measuring condition	Standard		Unit
			Min.	Max.	
$t_{d(BCLK-AD)}$	Address output delay time	Figure 3.1		28	ns
$t_{h(BCLK-AD)}$	Address output hold time (refers to BCLK)		4		ns
$t_{h(RD-AD)}$	Address output hold time (refers to RD)		0		ns
$t_{h(WR-AD)}$	Address output hold time (refers to WR)		(Note 2)		ns
$t_{d(BCLK-CS)}$	Chip select output delay time			28	ns
$t_{h(BCLK-CS)}$	Chip select output hold time (refers to BCLK)		4		ns
$t_{d(BCLK-ALE)}$	ALE signal output delay time			28	ns
$t_{h(BCLK-ALE)}$	ALE signal output hold time		-4		ns
$t_{d(BCLK-RD)}$	RD signal output delay time			28	ns
$t_{h(BCLK-RD)}$	RD signal output hold time		0		ns
$t_{d(BCLK-WR)}$	WR signal output delay time			28	ns
$t_{h(BCLK-WR)}$	WR signal output hold time		0		ns
$t_{d(BCLK-DB)}$	Data output delay time (refers to BCLK)			40	ns
$t_{h(BCLK-DB)}$	Data output hold time (refers to BCLK)(Note 3)		4		ns
$t_{d(DB-WR)}$	Data output delay time (refers to WR)		(Note 1)		ns
$t_{h(WR-DB)}$	Data output hold time (refers to WR)(Note 3)		(Note 2)		ns

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

$$\frac{(n-0.5) \times 10^9}{f(\text{BCLK})} - 40 \quad [\text{ns}]$$

n is "1" for 1-wait setting, "2" for 2-wait setting and "3" for 3-wait setting.
When n = "1", f(BCLK) is 12.5 MHz or less.

Note 2: Calculated according to the BCLK frequency as follows:

$$\frac{0.5 \times 10^9}{f(\text{BCLK})} - 10 \quad [\text{ns}]$$

Note 3: This standard value shows the timing when the output is off, and does not show hold time of data bus.

Hold time of data bus varies with capacitor volume and pull-up (pull-down) resistance value.

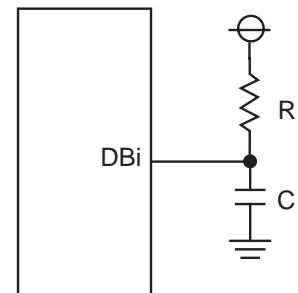
Hold time of data bus is expressed in

$$t = -CR \times \ln(1 - V_{OL} / V_{CC2})$$

by a circuit of the right figure.

For example, when $V_{OL} = 0.2V_{CC2}$, $C = 30\text{pF}$, $R = 1\text{k}\Omega$, hold time of output "L" level is

$$t = -30\text{pF} \times 1\text{k}\Omega \times \ln(1 - 0.2V_{CC2} / V_{CC2}) \\ = 6.7\text{ns}.$$



$$V_{CC1} = V_{CC2} = 5V$$

Switching Characteristics

($V_{CC1} = V_{CC2} = 5V$, $V_{SS} = 0V$, at $T_{opr} = -20$ to $70^{\circ}C$ unless otherwise specified)

Table 3.40. Memory Expansion and Microprocessor Modes

(for 2- to 3-wait setting, external area access and multiplex bus selection)

Symbol	Parameter	Measuring condition	Standard		Unit
			Min.	Max.	
$t_{d(BCLK-AD)}$	Address output delay time	Figure 3.1		28	ns
$t_{h(BCLK-AD)}$	Address output hold time (refers to BCLK)		4		ns
$t_{h(RD-AD)}$	Address output hold time (refers to RD)		(Note 1)		ns
$t_{h(WR-AD)}$	Address output hold time (refers to WR)		(Note 1)		ns
$t_{d(BCLK-CS)}$	Chip select output delay time			28	ns
$t_{h(BCLK-CS)}$	Chip select output hold time (refers to BCLK)		4		ns
$t_{h(RD-CS)}$	Chip select output hold time (refers to RD)		(Note 1)		ns
$t_{h(WR-CS)}$	Chip select output hold time (refers to WR)		(Note 1)		ns
$t_{d(BCLK-RD)}$	RD signal output delay time			28	ns
$t_{h(BCLK-RD)}$	RD signal output hold time		0		ns
$t_{d(BCLK-WR)}$	WR signal output delay time			28	ns
$t_{h(BCLK-WR)}$	WR signal output hold time		0		ns
$t_{d(BCLK-DB)}$	Data output delay time (refers to BCLK)			40	ns
$t_{h(BCLK-DB)}$	Data output hold time (refers to BCLK)		4		ns
$t_{d(DB-WR)}$	Data output delay time (refers to WR)		(Note 2)		ns
$t_{h(WR-DB)}$	Data output hold time (refers to WR)		(Note 1)		ns
$t_{d(BCLK-ALE)}$	ALE signal output delay time (refers to BCLK)			28	ns
$t_{h(BCLK-ALE)}$	ALE signal output hold time (refers to BCLK)		-4		ns
$t_{d(AD-ALE)}$	ALE signal output delay time (refers to Address)		(Note 3)		ns
$t_{h(ALE-AD)}$	ALE signal output hold time (refers to Address)		(Note 4)		ns
$t_{d(AD-RD)}$	RD signal output delay from the end of Address		0		ns
$t_{d(AD-WR)}$	WR signal output delay from the end of Address		0		ns
$t_{dZ(RD-AD)}$	Address output floating start time			8	ns

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

$$\frac{0.5 \times 10^9}{f(\text{BCLK})} - 10 \quad [\text{ns}]$$

Note 2: Calculated according to the BCLK frequency as follows:

$$\frac{(n-0.5) \times 10^9}{f(\text{BCLK})} - 40 \quad [\text{ns}] \quad n \text{ is "2" for 2-wait setting, "3" for 3-wait setting.}$$

Note 3: Calculated according to the BCLK frequency as follows:

$$\frac{0.5 \times 10^9}{f(\text{BCLK})} - 25 \quad [\text{ns}]$$

Note 4: Calculated according to the BCLK frequency as follows:

$$\frac{0.5 \times 10^9}{f(\text{BCLK})} - 15 \quad [\text{ns}]$$

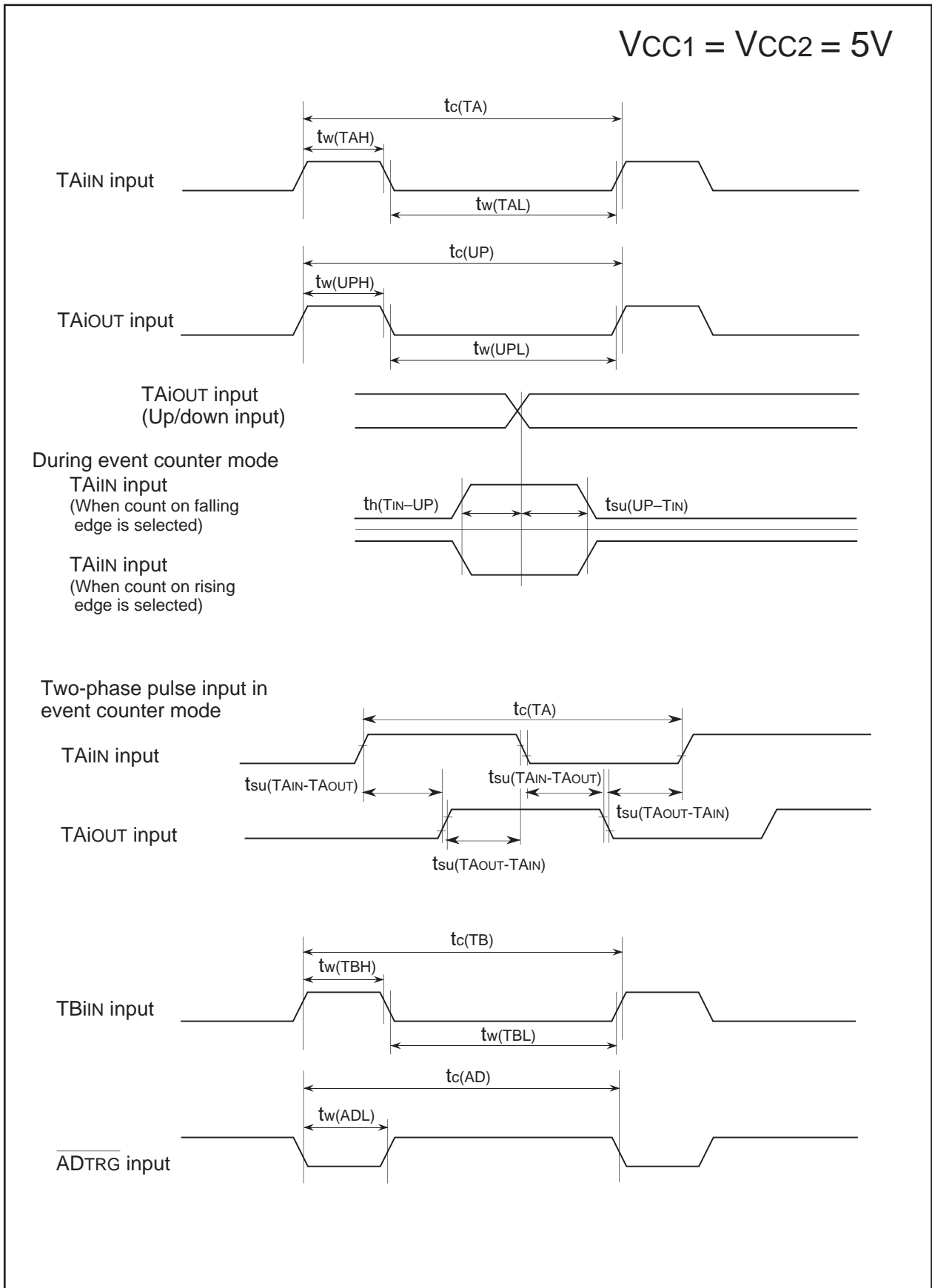


Figure 3.2. Timing Diagram (1)

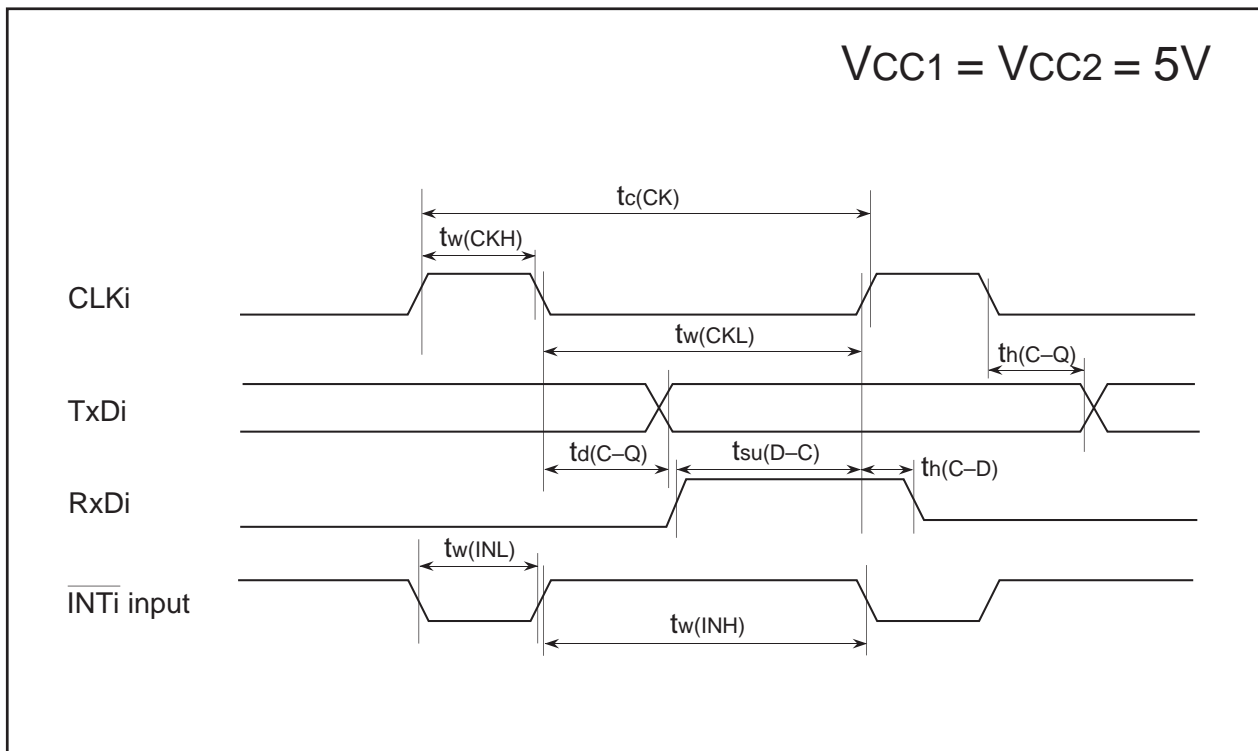


Figure 3.3. Timing Diagram (2)

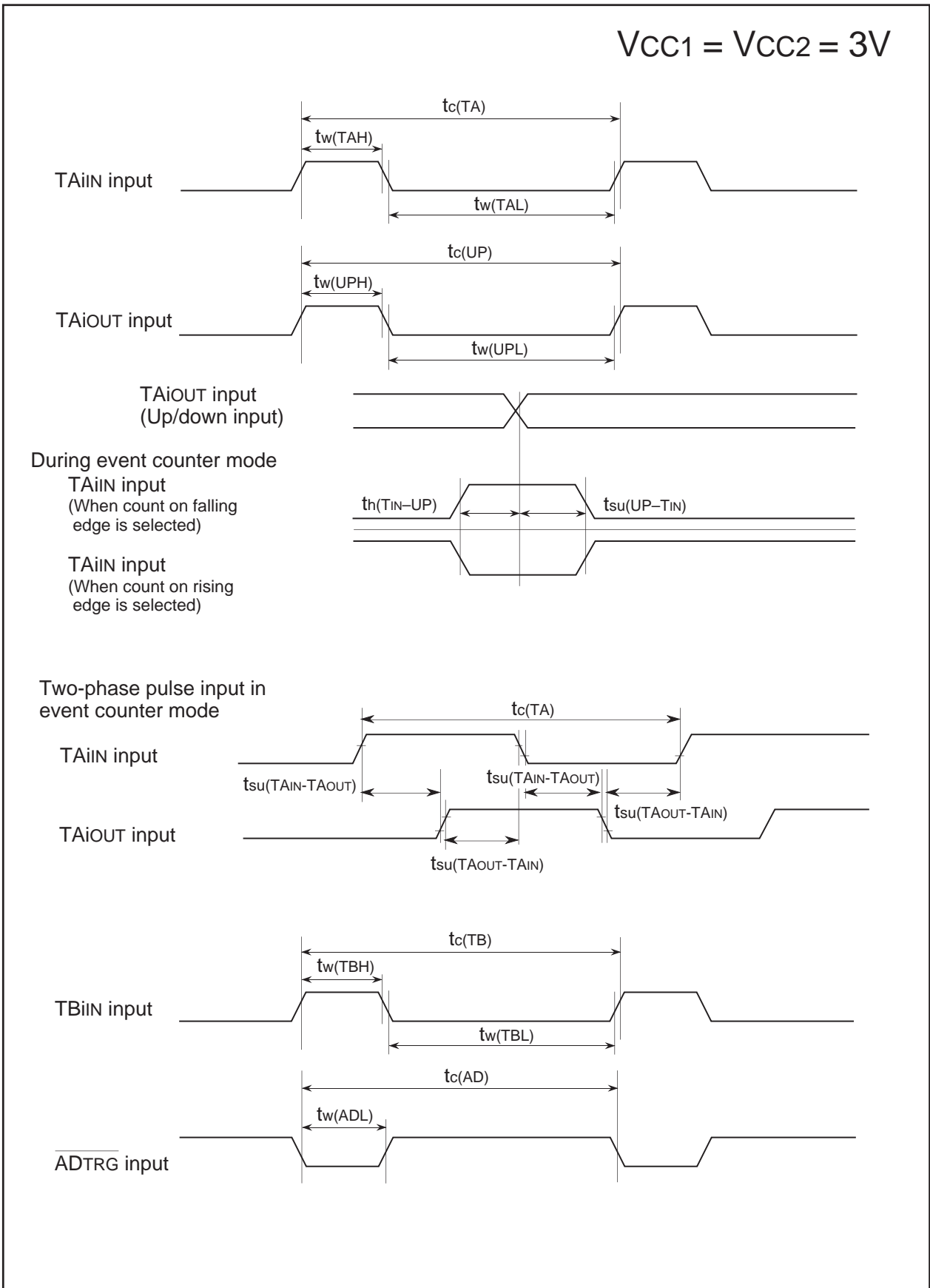


Figure 3.4. Timing Diagram (3)

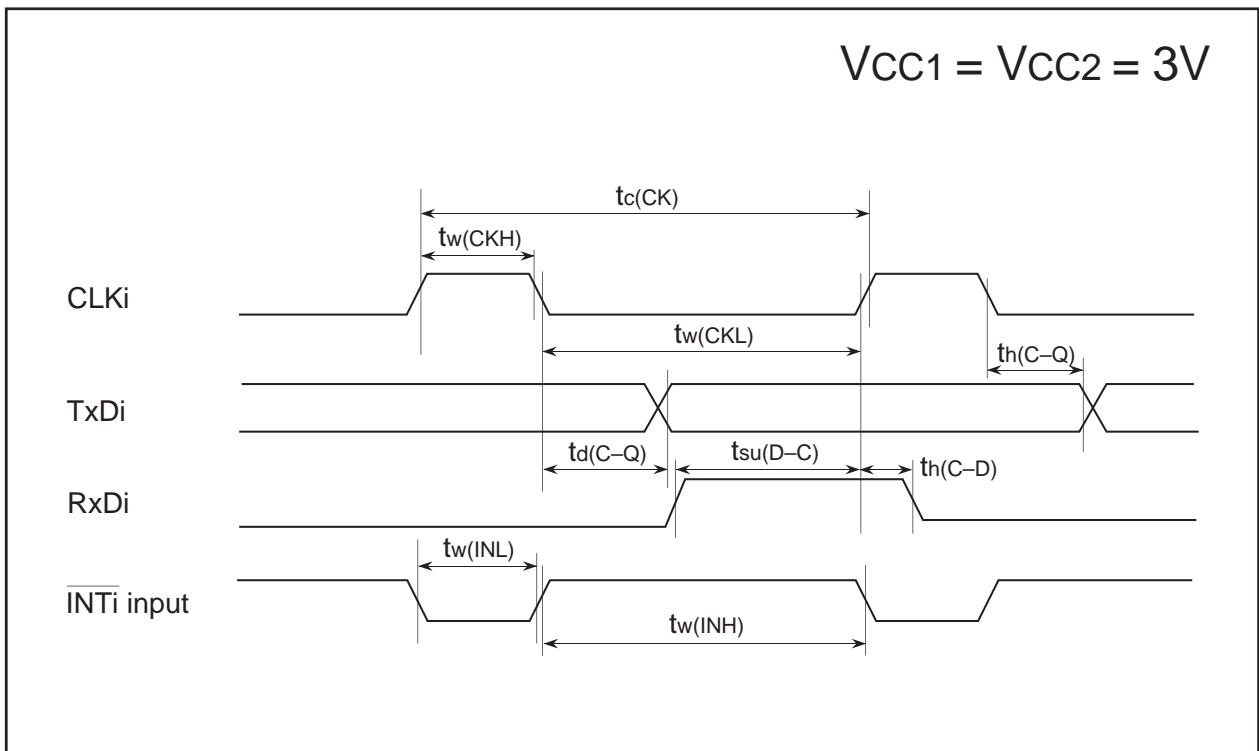


Figure 3.5. Timing Diagram (4)

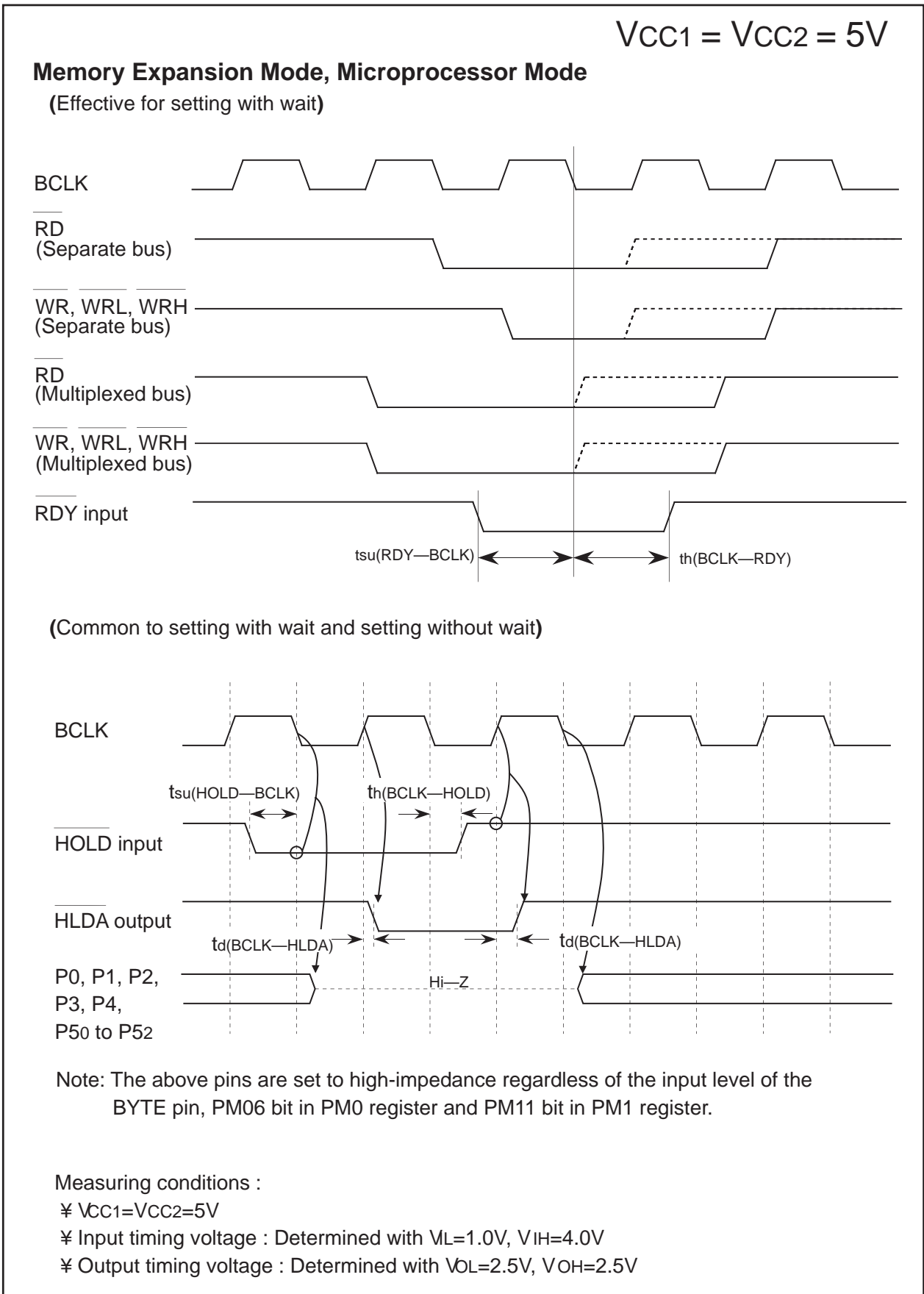
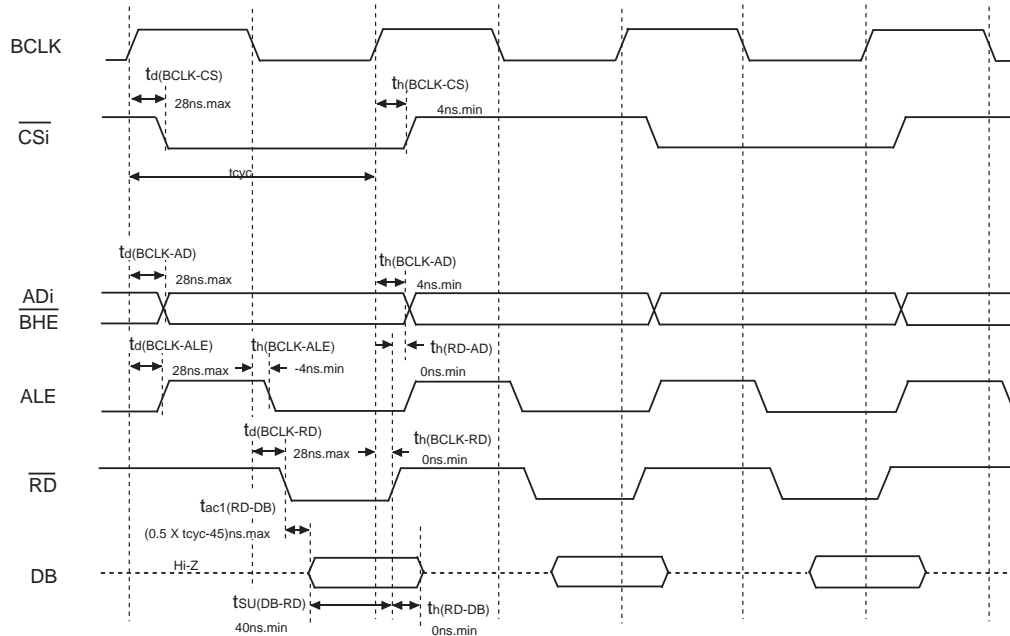


Figure 3.6. Timing Diagram (5)

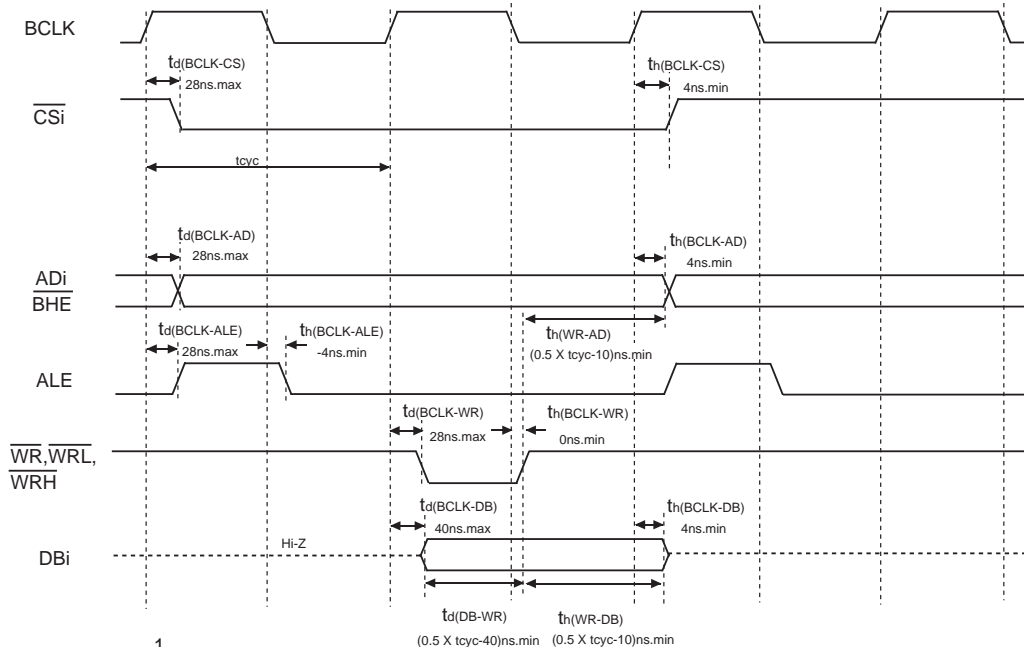
VCC1 = VCC2 = 5V

Memory Expansion Mode, Microprocessor Mode
(For setting with no wait)

Read timing



Write timing



$$t_{cyc} = \frac{1}{f(\text{BCLK})}$$

Measuring conditions

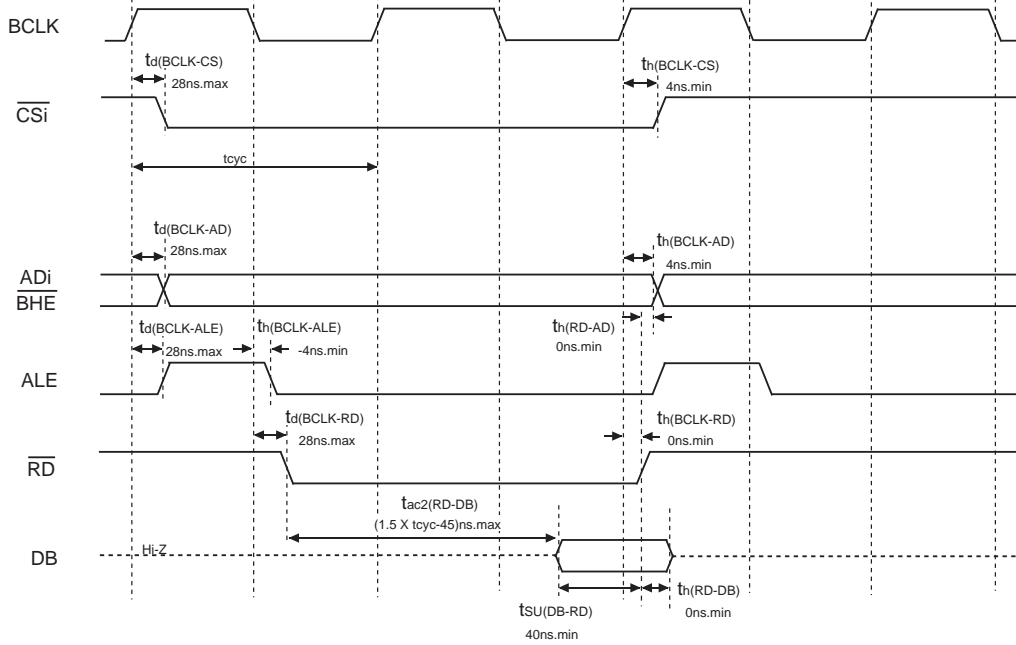
- VCC1=VCC2=5V
- Input timing voltage : VIL=0.8V, VIH=2.0V
- Output timing voltage : VOL=0.4V, VOH=2.4V

Figure 3.7. Timing Diagram (6)

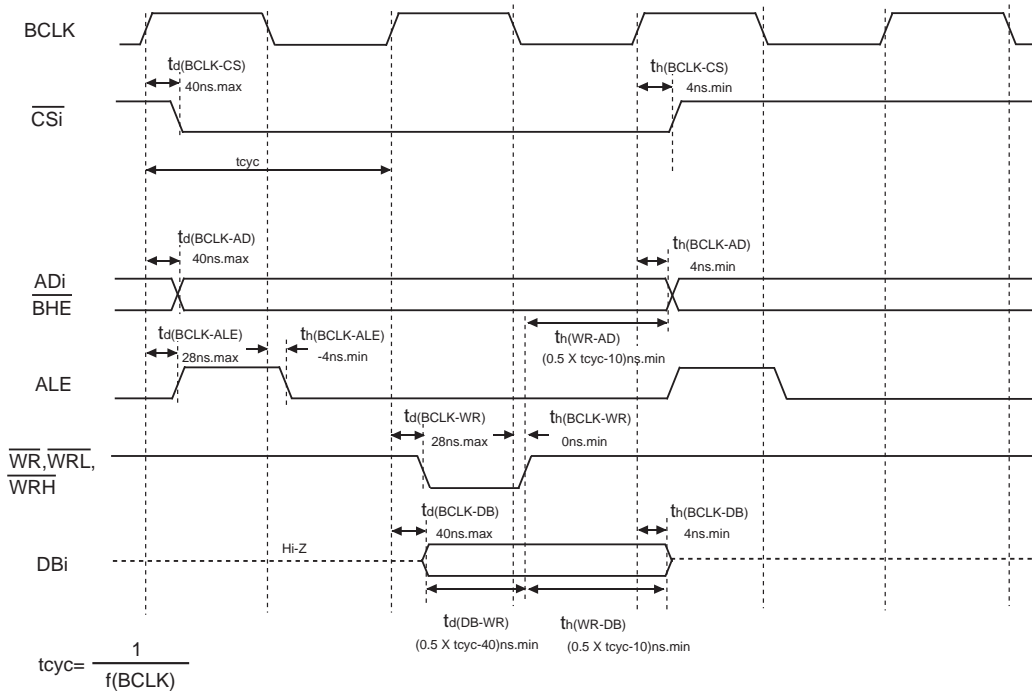
VCC1 = VCC2 = 5V

Memory Expansion Mode, Microprocessor Mode
(for 1-wait setting and external area access)

Read timing



Write timing



Measuring conditions

- VCC1=VCC2=5V
- Input timing voltage : VIL=0.8V, VIH=2.0V
- Output timing voltage : VOL=0.4V, VOH=2.4V

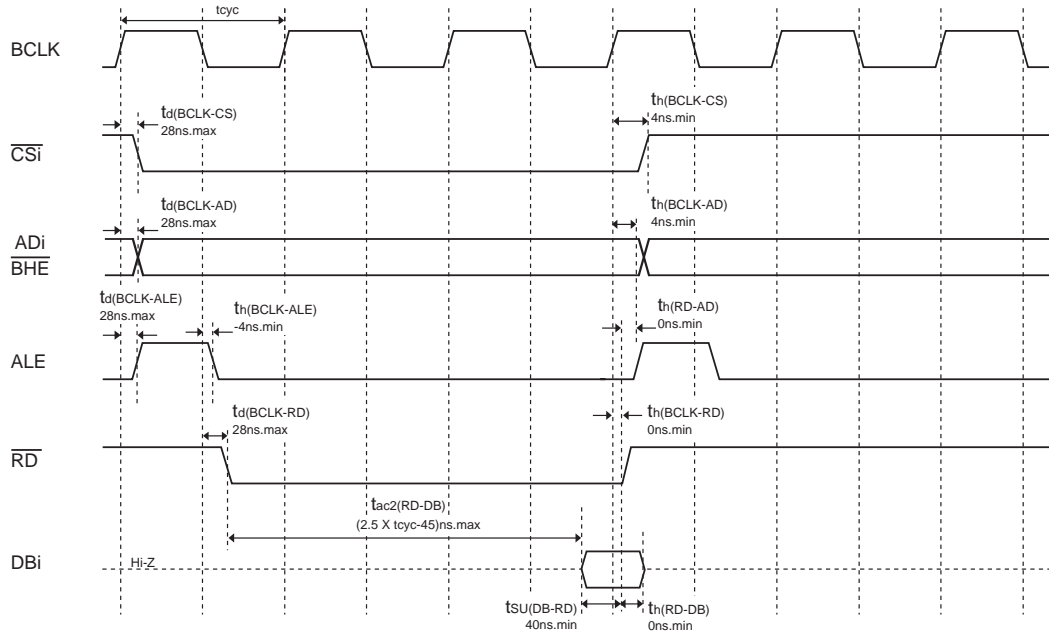
Figure 3.8. Timing Diagram (7)

VCC1 = VCC2 = 5V

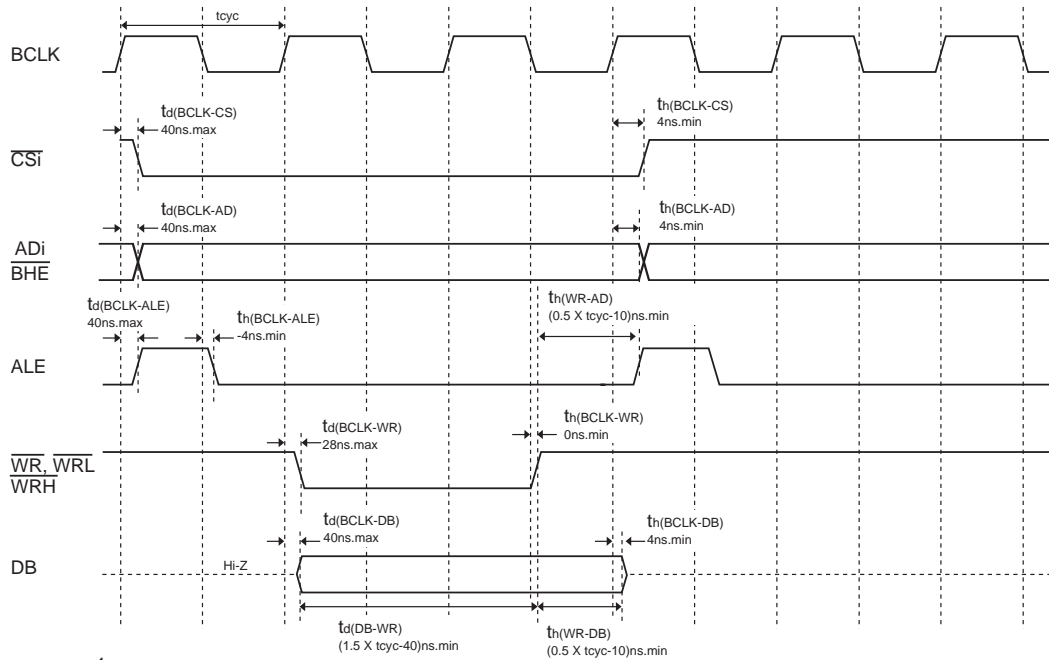
Memory Expansion Mode, Microprocessor Mode

(for 2-wait setting and external area access)

Read timing



Write timing



$$t_{cy} = \frac{1}{f(\text{BCLK})}$$

Measuring conditions

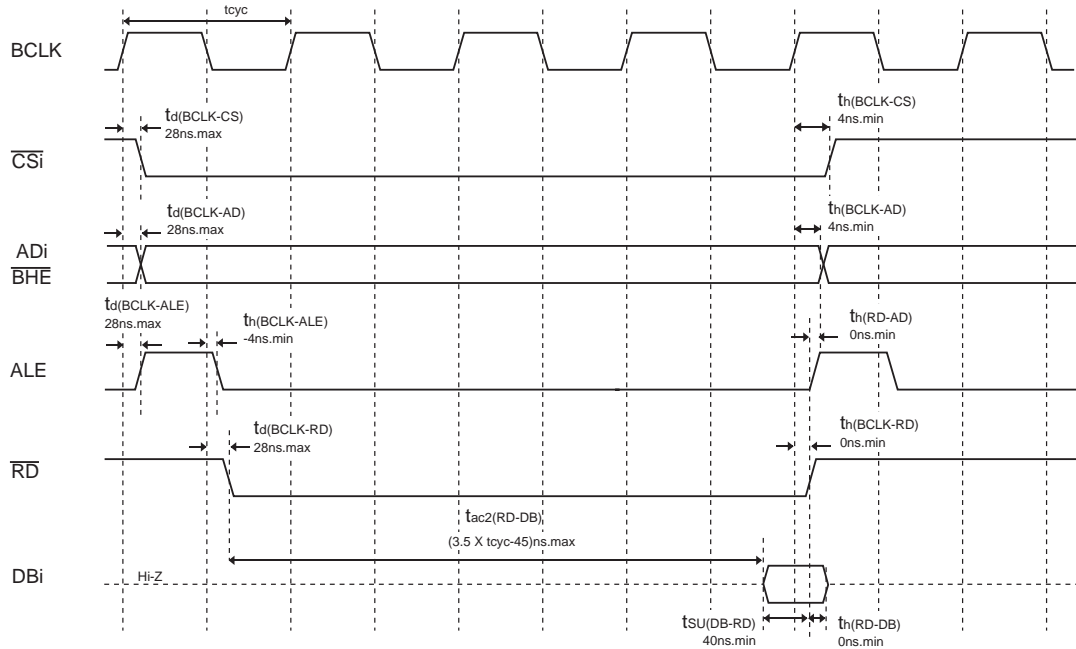
- VCC1=VCC2=5V
- Input timing voltage : VIL=0.8V, VIH=2.0V
- Output timing voltage : VOL=0.4V, VOH=2.4V

Figure 3.9. Timing Diagram (8)

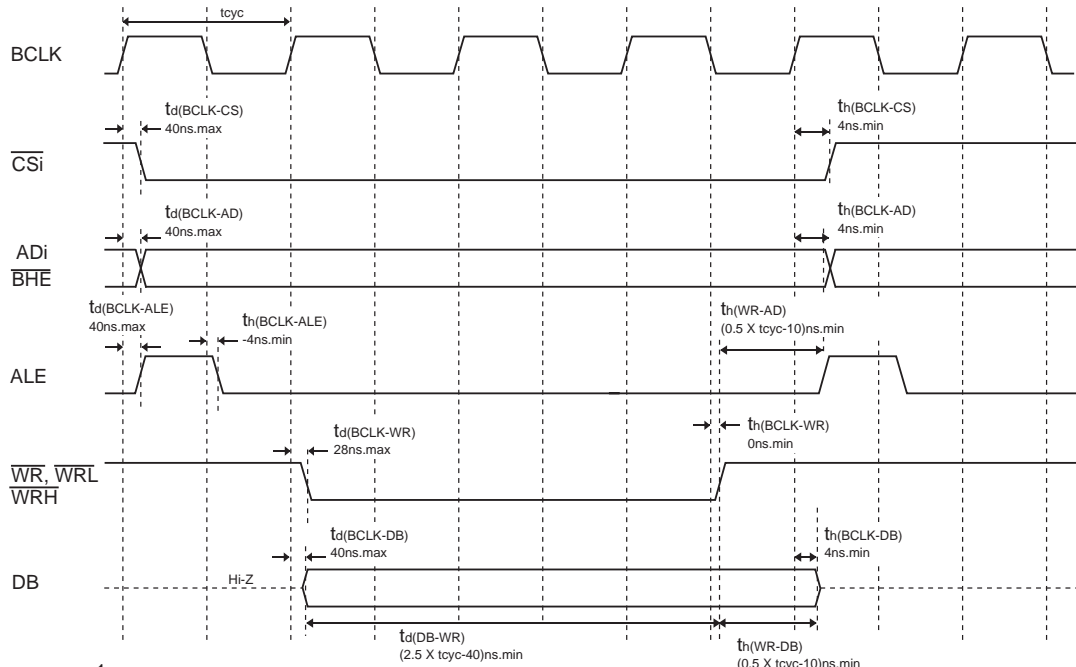
VCC1 = VCC2 = 5V

Memory Expansion Mode, Microprocessor Mode
(for 3-wait setting and external area access)

Read timing



Write timing



$$t_{cyc} = \frac{1}{f(BCLK)}$$

Measuring conditions

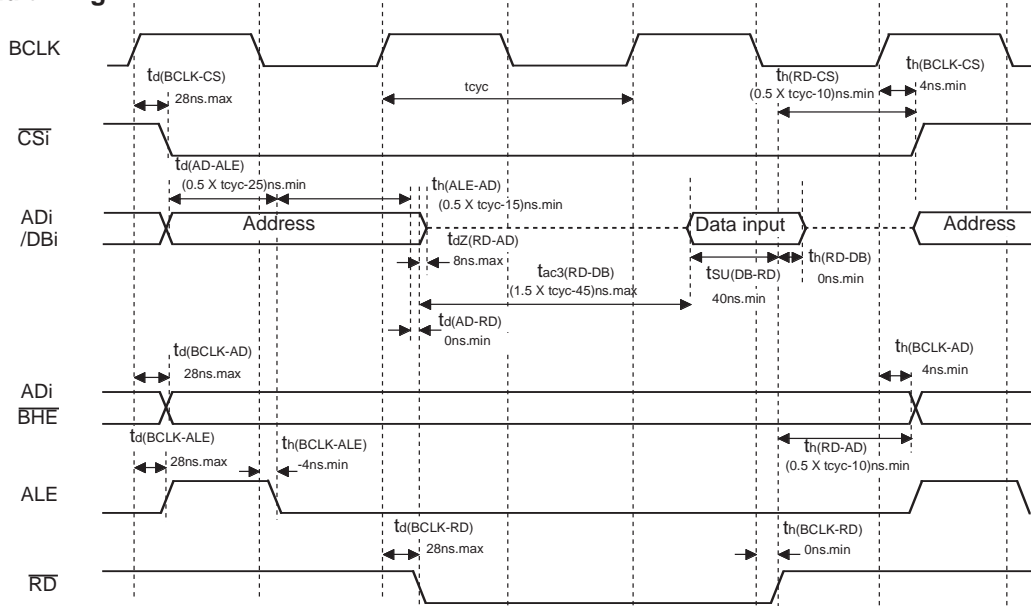
- VCC1=VCC2=5V
- Input timing voltage : VIL=0.8V, VIH=2.0V
- Output timing voltage : VOL=0.4V, VOH=2.4V

Figure 3.10. Timing Diagram (9)

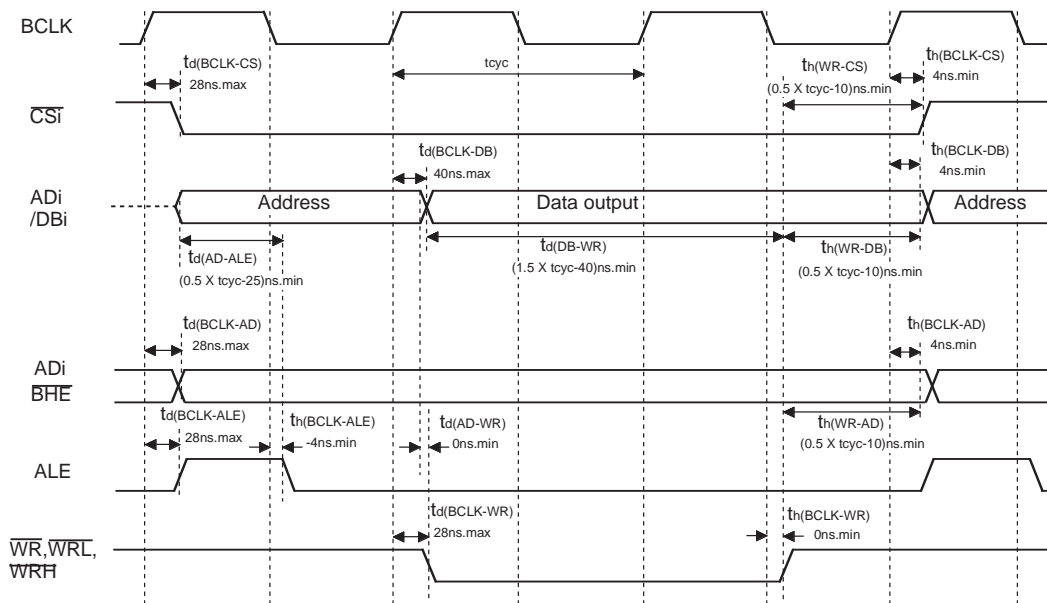
VCC1 = VCC2 = 5V

Memory Expansion Mode, Microprocessor Mode
 (For 1- or 2-wait setting, external area access and multiplex bus selection)

Read timing



Write timing



$$t_{\text{cyc}} = \frac{1}{f(\text{BCLK})}$$

Measuring conditions

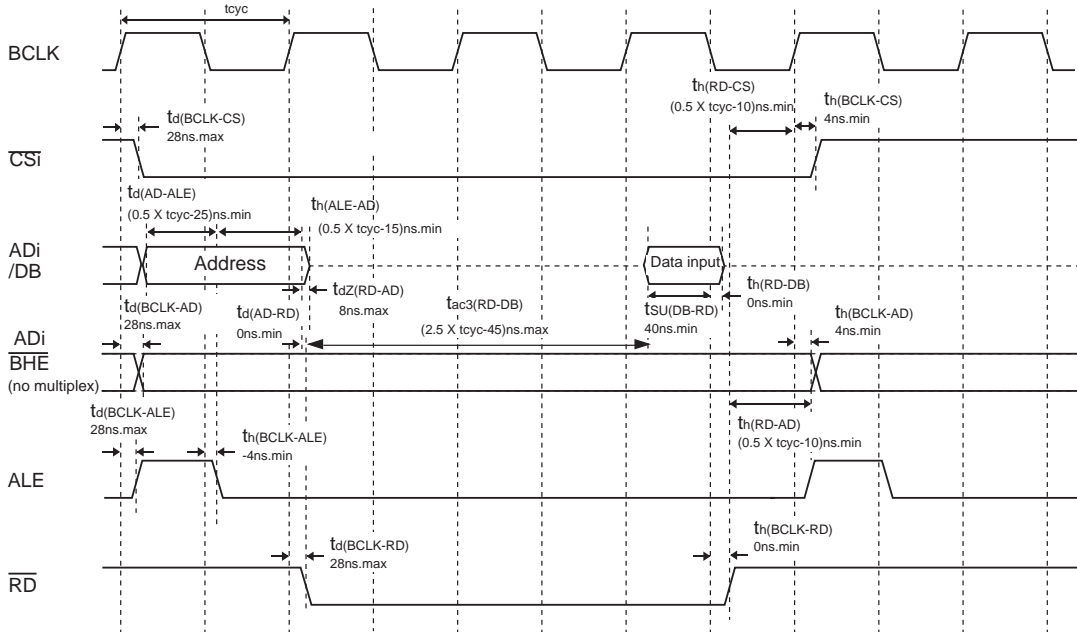
- VCC1=VCC2=5V
- Input timing voltage : VIL=0.8V, VIH=2.0V
- Output timing voltage : VOL=0.4V, VOH=2.4V

Figure 3.11. Timing Diagram (10)

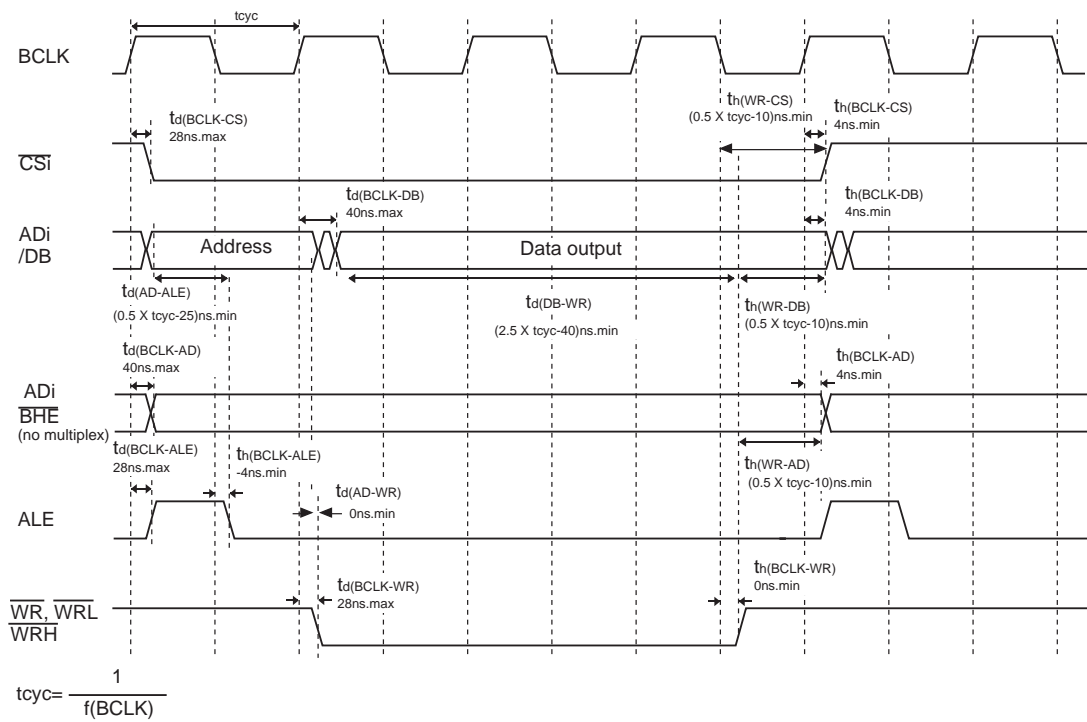
VCC1 = VCC2 = 5V

Memory Expansion Mode, Microprocessor Mode
 (For 3-wait setting, external area access and multiplex bus selection)

Read timing



Write timing



Measuring conditions
 • VCC1=VCC2=5V
 • Input timing voltage : VIL=0.8V, VIH=2.0V
 • Output timing voltage : VOL=0.4V, VOH=2.4V

Figure 3.12. Timing Diagram (11)

4 Flash Memory Version

4.1 Flash Memory Performance

The flash memory version is functionally the same as the mask ROM version except that it internally contains flash memory.

The flash memory version has three modes—CPU rewrite, standard serial input/output, and parallel input/output modes—in which its internal flash memory can be operated on.

Table 4.1.1 shows the outline performance of flash memory version (see Table 1.4.1 for the items not listed in Table 4.1.1.).

Table 4.1.1. Flash Memory Version Specifications

Item		Specification
Flash memory operating mode		3 modes (CPU rewrite, standard serial I/O, parallel I/O)
Erase block	User ROM area	See Figure 4.2.1
	Boot ROM area	1 block (4 Kbytes) (Note 1)
Method for program		In units of word
Method for erasure		Block erase
Program, erase control method		Program and erase controlled by software command
Protect method		Protected for each block by lock bit
Number of commands		7 commands
Number of program and erasure		100 times
Data Retention		10 years
ROM code protection		Parallel I/O and standard serial I/O modes are supported.

Note 1: The boot ROM area contains a standard serial I/O mode rewrite control program which is stored in it when shipped from the factory. This area can only be rewritten in parallel input/output mode.

Table 4.1.2. Flash Memory Rewrite Modes Overview

Flash memory rewrite mode	CPU rewrite mode (Note 1)	Standard serial I/O mode	Parallel I/O mode
Function	The user ROM area is rewritten by executing software commands from the CPU. EW0 mode: Can be rewritten in any area other than the flash memory (Note 2) EW1 mode: Can be rewritten in the flash memory	The user ROM area is rewritten by using a dedicated serial programmer. Standard serial I/O mode 1: Clock sync serial I/O Standard serial I/O mode 2: UART	The boot ROM and user ROM areas are rewritten by using a dedicated parallel programmer.
Areas which can be rewritten	User ROM area	User ROM area	User ROM area Boot ROM area
Operation mode	Single chip mode Boot mode (EW0 mode)	Boot mode	Parallel I/O mode
ROM programmer	None	Serial programmer	Parallel programmer

Note 1: The PM13 bit remains set to "1" while the FMR0 register FMR01 bit = 1 (CPU rewrite mode enabled). The PM13 bit is reverted to its original value by clearing the FMR01 bit to "0" (CPU rewrite mode disabled). However, if the PM13 bit is changed during CPU rewrite mode, its changed value is not reflected until after the FMR01 bit is cleared to "0".

Note 2: When in CPU rewrite mode, the PM10 and PM13 bits in the PM1 register are set to "1". The rewrite control program can only be executed in the internal RAM.

4.2 Memory Map

The ROM in the flash memory version is separated between a user ROM area and a boot ROM area. Figure 4.2.1 shows the block diagram of flash memory.

The user ROM area is divided into several blocks, each of which can individually be protected (locked) against programming or erasure. The user ROM area can be rewritten in all of CPU rewrite, standard serial input/output, and parallel input/output modes.

The boot ROM area is located at addresses that overlap the user ROM area, and can only be rewritten in parallel input/output mode. After a hardware reset that is performed by applying a high-level signal to the CNVss and P50 pins and a low-level signal to the M1 pin, the program in the boot ROM area is executed. After a hardware reset that is performed by applying a low-level signal to the CNVss pin, the program in the user ROM area is executed (but the boot ROM area cannot be read).

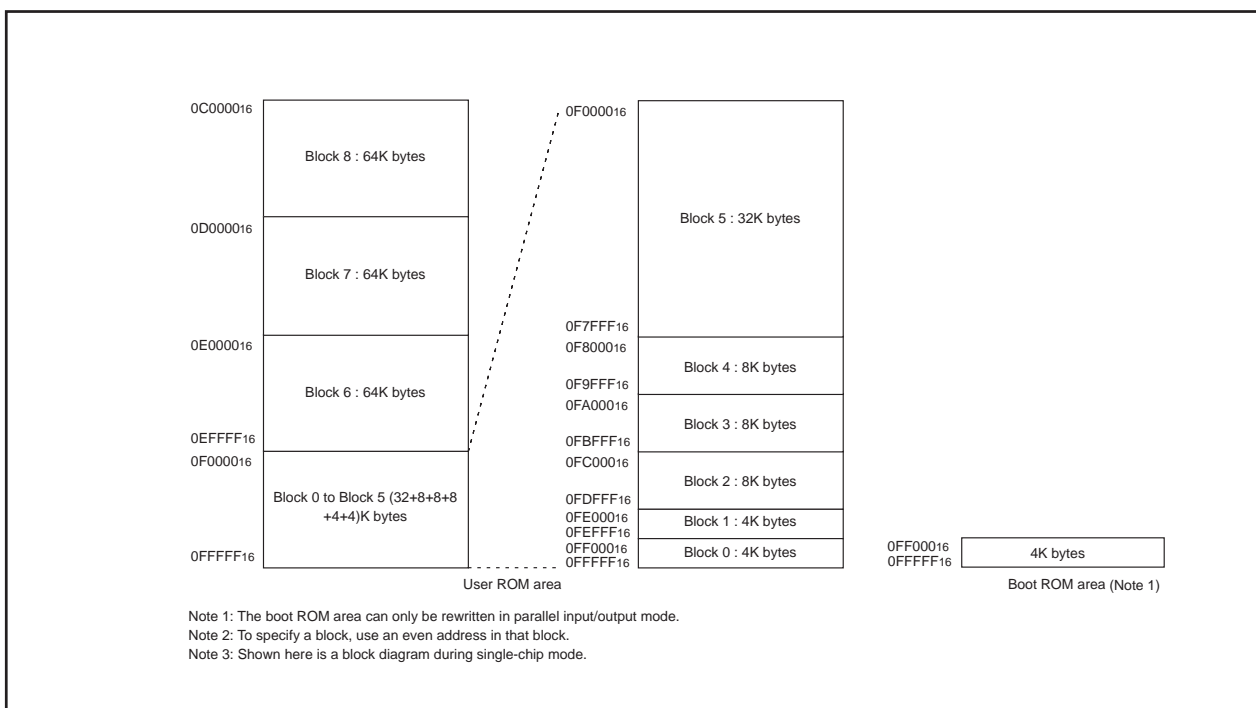


Figure 4.2.1. Flash Memory Block Diagram

Boot Mode

After a hardware reset which is performed by applying a low-level signal to the M1 pin and a high-level signal to the CNVss and P50 pins, the microcomputer is placed in boot mode, thereby executing the program in the boot ROM area.

During boot mode, the boot ROM and user ROM areas are switched over by the FMR05 bit in the FMR0 register.

The boot ROM area contains a standard serial input/output mode based rewrite control program which was stored in it when shipped from the factory.

The boot ROM area can be rewritten in parallel input/output mode. Prepare an EW0 mode based rewrite control program and write it in the boot ROM area, and the flash memory can be rewritten as suitable for the system.

Functions To Prevent Flash Memory from Rewriting

To prevent the flash memory from being read or rewritten easily, parallel input/output mode has a ROM code protect and standard serial input/output mode has an ID code check function.

- **ROM Code Protect Function**

The ROM code protect function inhibits the flash memory from being read or rewritten during parallel input/output mode. Figure 4.2.2 shows the ROMCP register.

The ROMCP register is located in the user ROM area. The ROMCP1 bit consists of two bits. The ROM code protect function is enabled by clearing one or both of two ROMCP1 bits to "0" when the ROMCR bits are not '002,' with the flash memory thereby protected against reading or rewriting. Conversely, when the ROMCR bits are '002' (ROM code protect removed), the flash memory can be read or rewritten. Once the ROM code protect function is enabled, the ROMCR bits cannot be changed during parallel input/output mode. Therefore, use standard serial input/output or other modes to rewrite the flash memory.

- **ID Code Check Function**

Use this function in standard serial input/output mode. Unless the flash memory is blank, the ID codes sent from the programmer and the ID codes written in the flash memory are compared to see if they match. If the ID codes do not match, the commands sent from the programmer are not accepted. The ID code consists of 8-bit data, the areas of which, beginning with the first byte, are 0FFFDF₁₆, 0FFFE3₁₆, 0FFFEB₁₆, 0FFFEF₁₆, 0FFFF3₁₆, 0FFFF7₁₆, and 0FFFFB₁₆. Prepare a program in which the ID codes are preset at these addresses and write it in the flash memory.

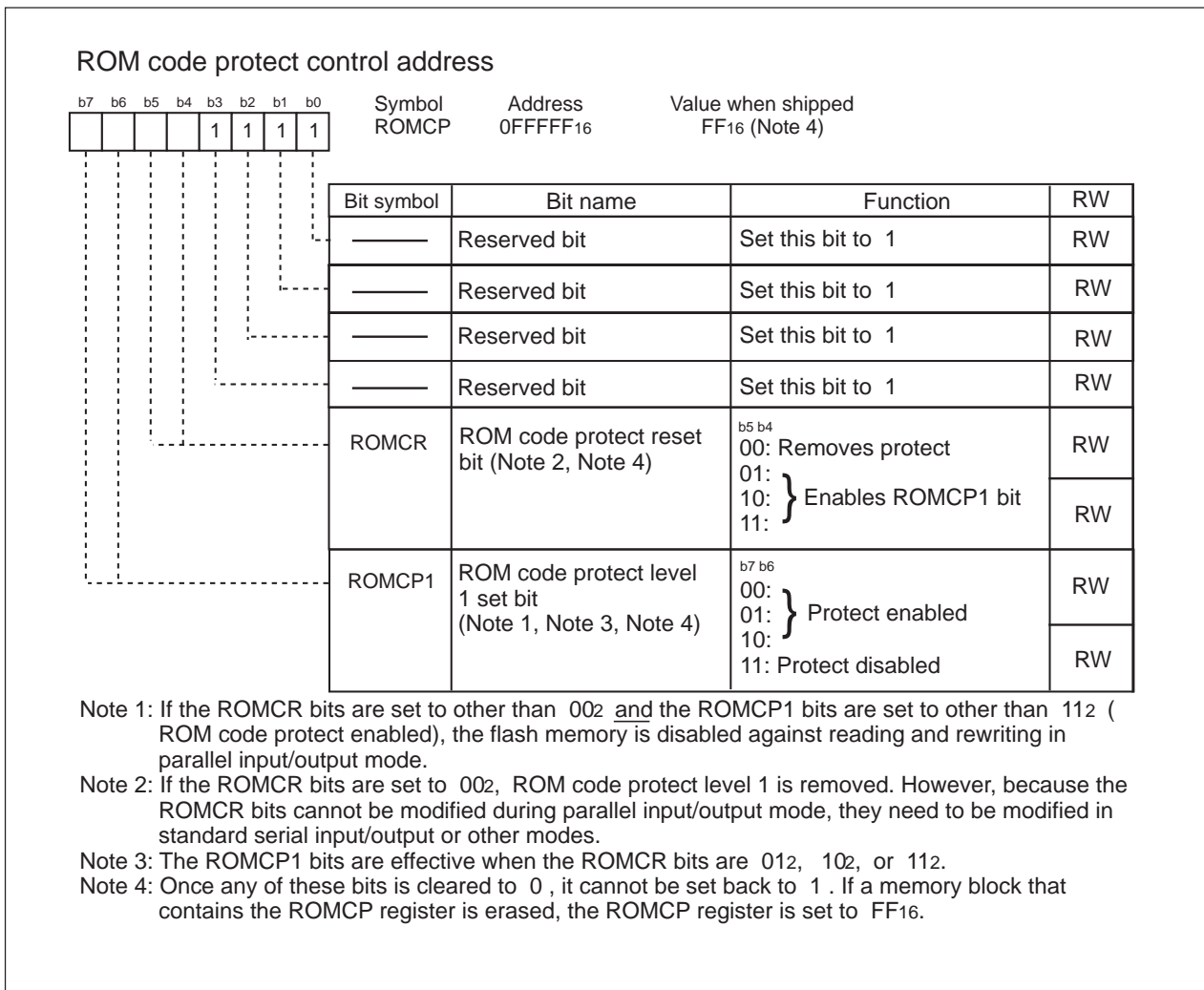


Figure 4.2.2. ROMCP Register

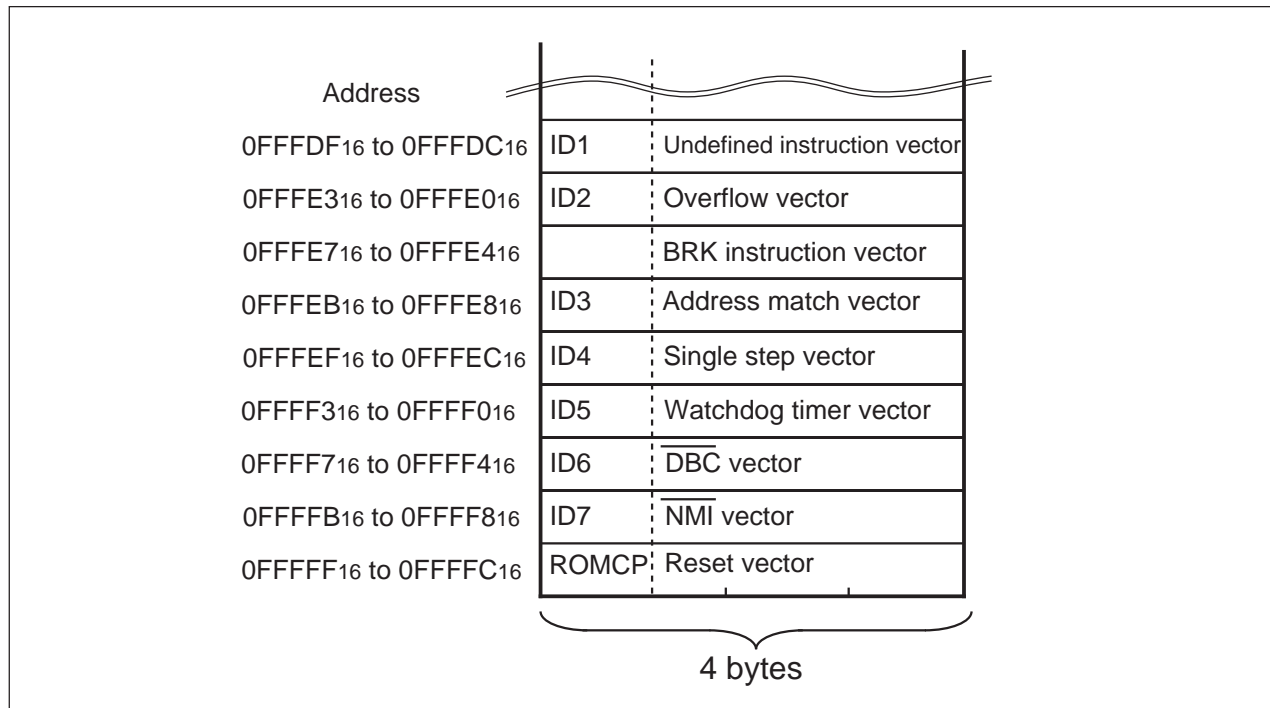


Figure 4.2.3. Address for ID Code Stored

CPU Rewrite Mode

In CPU rewrite mode, the user ROM area can be rewritten by executing software commands from the CPU. Therefore, the user ROM area can be rewritten directly while the microcomputer is mounted on-board without having to use a ROM programmer, etc.

In CPU rewrite mode, only the user ROM area shown in Figure 4.2.1 can be rewritten and the boot ROM area cannot be rewritten. Make sure the Program and the Block Erase commands are executed only on each block in the user ROM area.

During CPU rewrite mode, the user ROM area be operated on in either Erase Write 0 (EW0) mode or Erase Write 1 (EW1) mode. Table 4.2.1 lists the differences between Erase Write 0 (EW0) and Erase Write 1 (EW1) modes.

Table 4.2.1. EW0 Mode and EW1 Mode

Item	EW0 mode	EW1 mode
Operation mode	<ul style="list-style-type: none"> • Single chip mode • Boot mode 	Single chip mode
Areas in which a rewrite control program can be located	<ul style="list-style-type: none"> • User ROM area • Boot ROM area 	User ROM area
Areas in which a rewrite control program can be executed	Must be transferred to any area other than the flash memory (RAM) before being executed (Note 2)	Can be executed directly in the user ROM area
Areas which can be rewritten	User ROM area	User ROM area However, this does not include the area in which a rewrite control program exists
Software command limitations	None	<ul style="list-style-type: none"> • Program, Block Erase command Cannot be executed on any block in which a rewrite control program exists • Read Status Register command Cannot be executed
Modes after Program or Erase	Read Status Register mode	Read Array mode
CPU status during Auto Write and Auto Erase	Operating	Hold state (I/O ports retain the state in which they were before the command was executed) ^(Note 1)
Flash memory status detection	<ul style="list-style-type: none"> • Read the FMR0 register's FMR00, FMR06, and FMR07 bits in a program • Execute the Read Status Register command to read the status register's SR7, SR5, and SR4 flags. 	Read the FMR0 register's FMR00, FMR06, and FMR07 bits in a program

Note 1: Make sure no interrupts (except NMI and watchdog timer interrupts) and DMA transfers will occur.

Note 2: When in CPU rewrite mode, the PM10 and PM13 bits in the PM1 register are set to "1". The rewrite control program can only be executed in the internal RAM.

- **EW0 Mode**

The microcomputer is placed in CPU rewrite mode by setting the FMR0 register's FMR01 bit to "1" (CPU rewrite mode enabled), ready to accept commands. In this case, because the FMR1 register's FMR11 bit = 0, EW0 mode is selected. The FMR01 bit can be set to "1" by writing "0" and then "1" in succession. Use software commands to control program and erase operations. Read the FMR0 register or status register to check the status of program or erase operation at completion.

- **EW1 Mode**

EW1 mode is selected by setting FMR11 bit to "1" (by writing "0" and then "1" in succession) after setting the FMR01 bit to "1" (by writing "0" and then "1" in succession).

Read the FMR0 register to check the status of program or erase operation at completion. The status register cannot be read during EW1 mode.

Figure 4.2.4 shows the FMR0 and FMR1 registers.

FMR00 Bit

This bit indicates the operating status of the flash memory. The bit is “0” when the Program, Erase, or Lock Bit program is running; otherwise, the bit is “1”.

FMR01 Bit

The microcomputer is made ready to accept commands by setting the FMR01 bit to “1” (CPU rewrite mode). During boot mode, make sure the FMR05 bit also is “1” (user ROM area access).

FMR02 Bit

The lock bit set for each block can be disabled by setting the FMR02 bit to “1” (lock bit disabled). (Refer to the description of the data protect function.) The lock bits set are enabled by setting the FMR02 bit to “0”. The FMR02 bit only disables the lock bit function and does not modify the lock bit data (lock bit status flag). However, if the Erase command is executed while the FMR02 bit is set to “1”, the lock bit data changes state from “0” (locked) to “1” (unlocked) after Erase is completed.

FMSTP Bit

This bit is provided for initializing the flash memory control circuits, as well as for reducing the amount of current consumed in the flash memory. The internal flash memory is disabled against access by setting the FMSTP bit to “1”. Therefore, make sure the FMSTP bit is modified in other than the flash memory. In the following cases, set the FMSTP bit to “1”:

- When flash memory access resulted in an error while erasing or programming in EW0 mode (FMR00 bit not reset to “1” (ready))
- When entering low power mode

Figure 4.2.7 shows a flow chart to be followed before and after entering low power mode.

Note that when going to stop or wait mode, the FMR0 register does not need to be set because the power for the internal flash memory is automatically turned off and is turned back on again after returning from stop or wait mode.

FMR05 Bit

This bit switches between the boot ROM and user ROM areas during boot mode. Set this bit to “0” when accessing the boot ROM area (for read) or “1” (user ROM access) when accessing the user ROM area (for read, write, or erase).

FMR06 Bit

This is a read-only bit indicating the status of auto program operation. The bit is set to “1” when a program error occurs; otherwise, it is cleared to “0”. For details, refer to the description of the full status check.

FMR07 Bit

This is a read-only bit indicating the status of auto erase operation. The bit is set to “1” when an erase error occurs; otherwise, it is cleared to “0”. For details, refer to the description of the full status check.

Figure 4.2.5 and 4.2.6 show the setting and resetting of EW0 mode and EW1 mode, respectively.

FMR11 Bit

Setting this bit to “1” places the microcomputer in EW1 mode.

FMR16 Bit

This is a read-only bit indicating the execution result of the Read Lock Bit Status command.

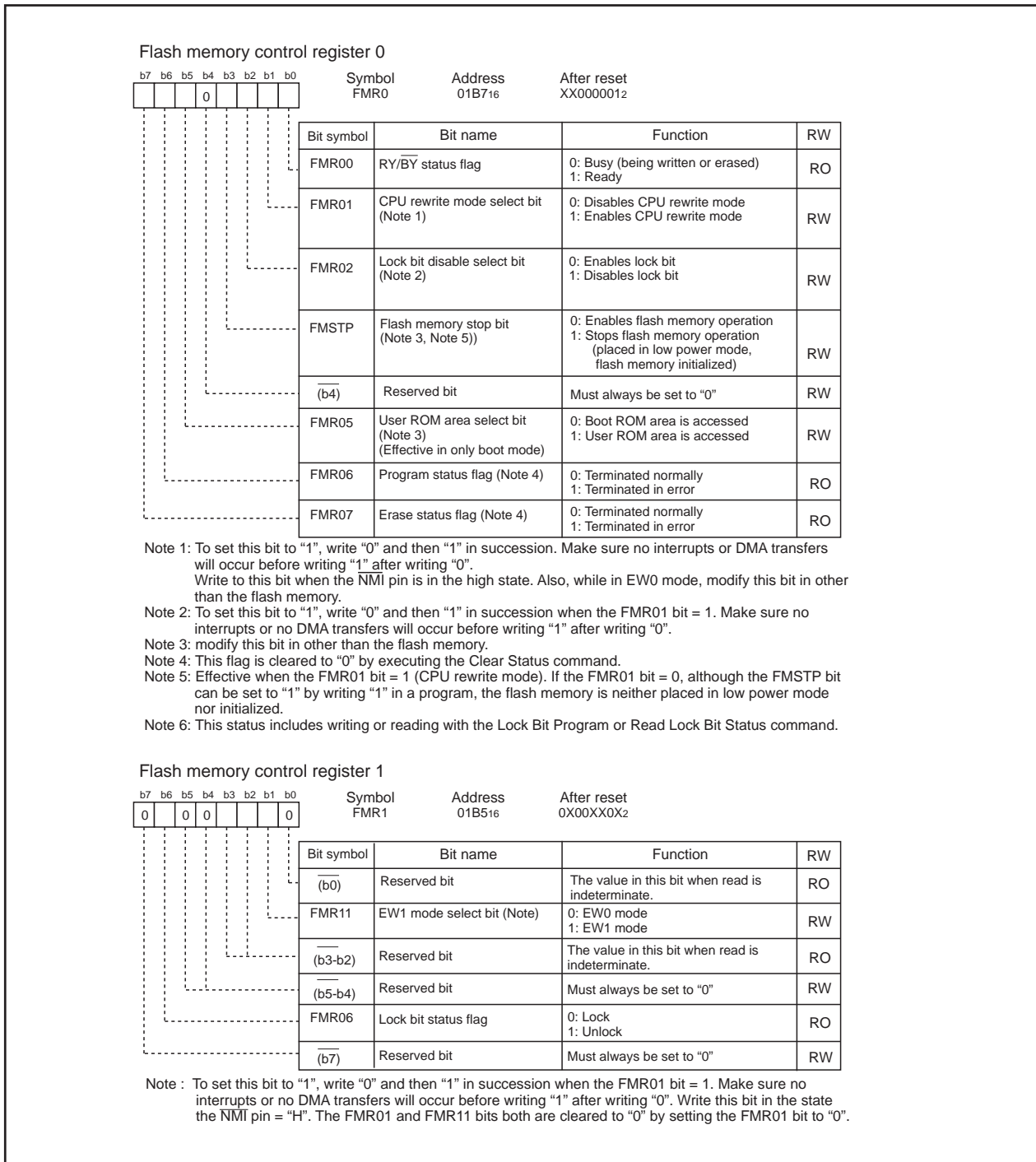


Figure 4.2.4. FIDR Register and FMR0 and FMR1 Registers

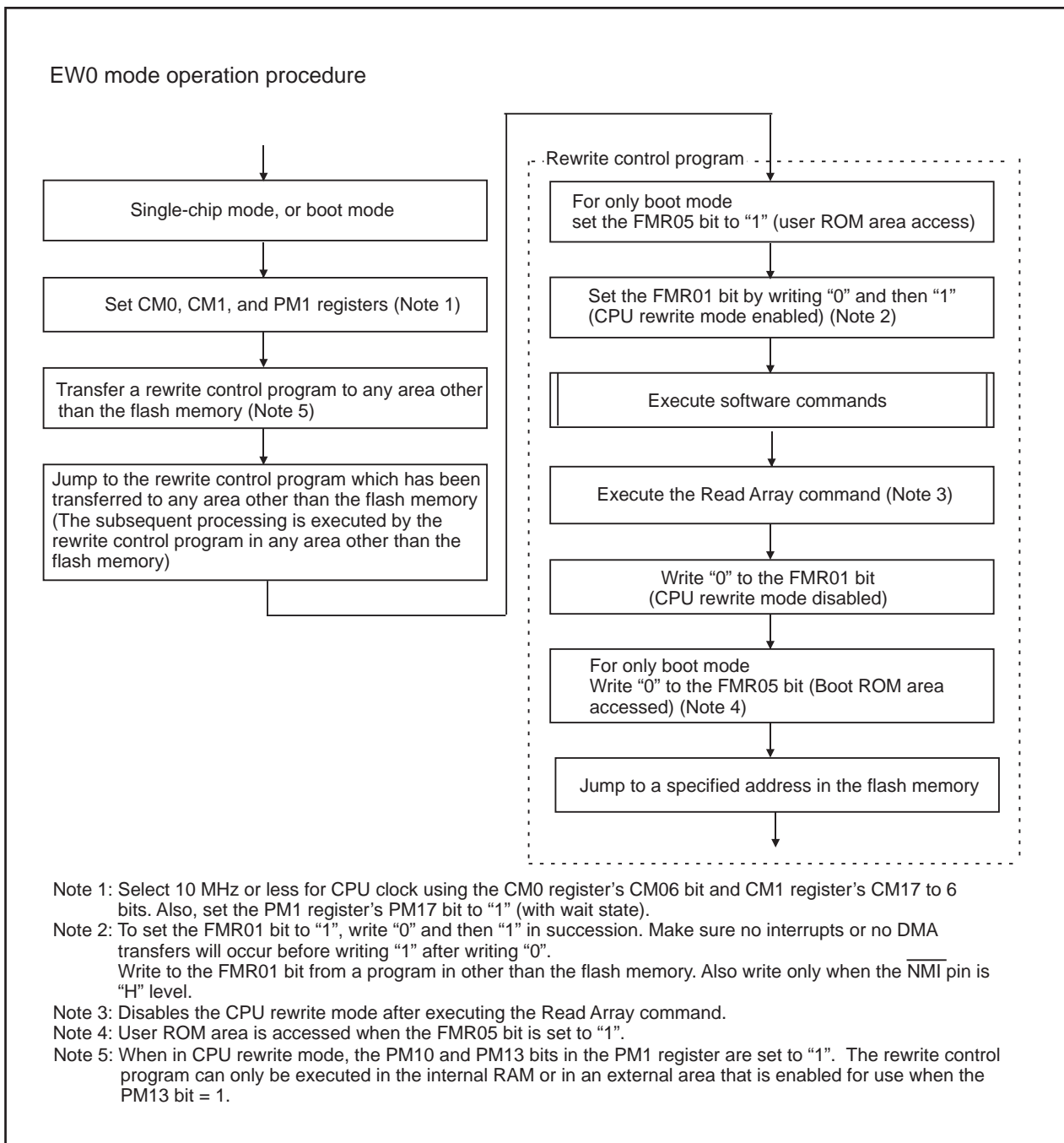


Figure 4.2.5. Setting and Resetting of EW0 Mode

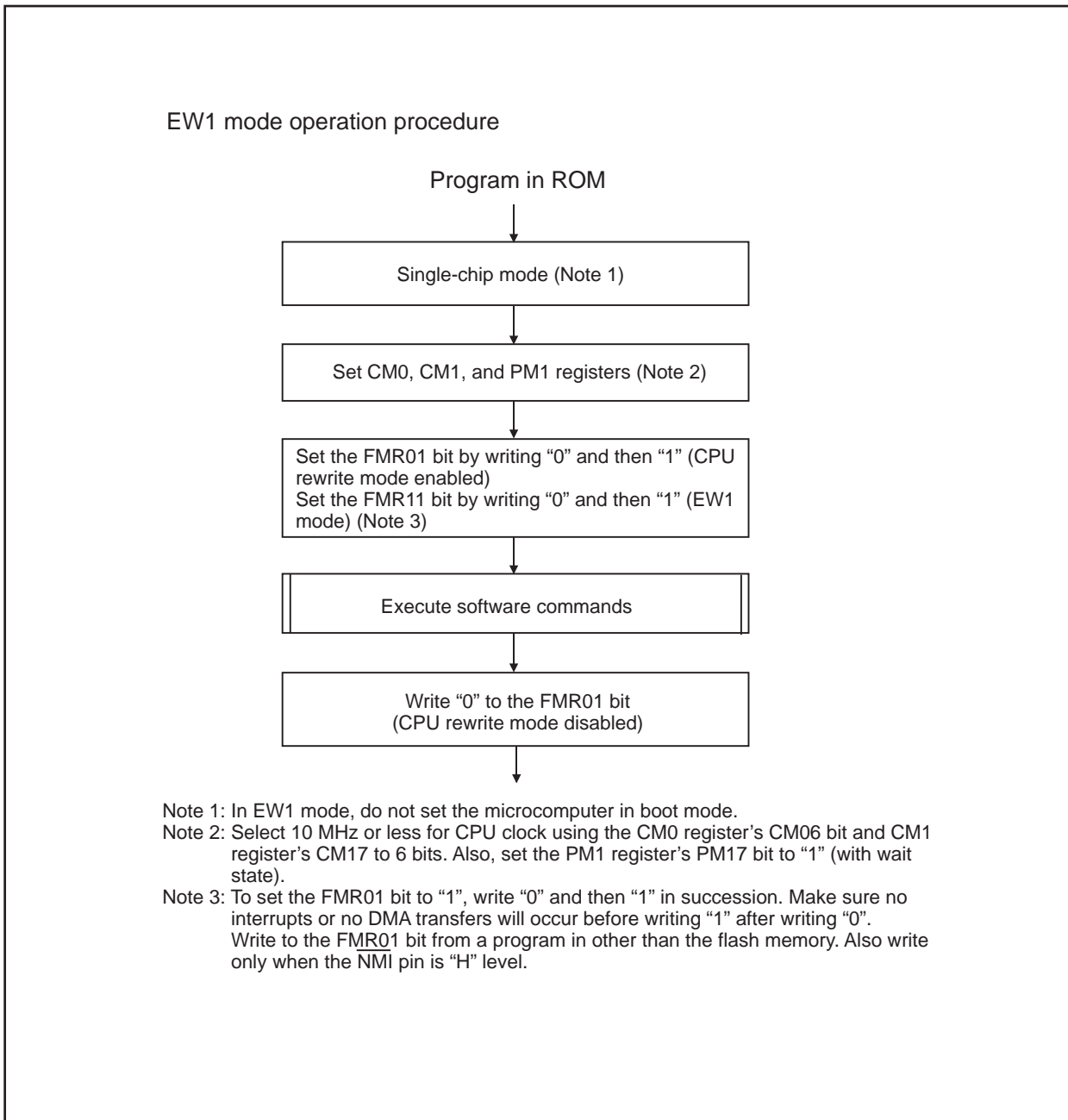


Figure 4.2.6. Setting and Resetting of EW1 Mode

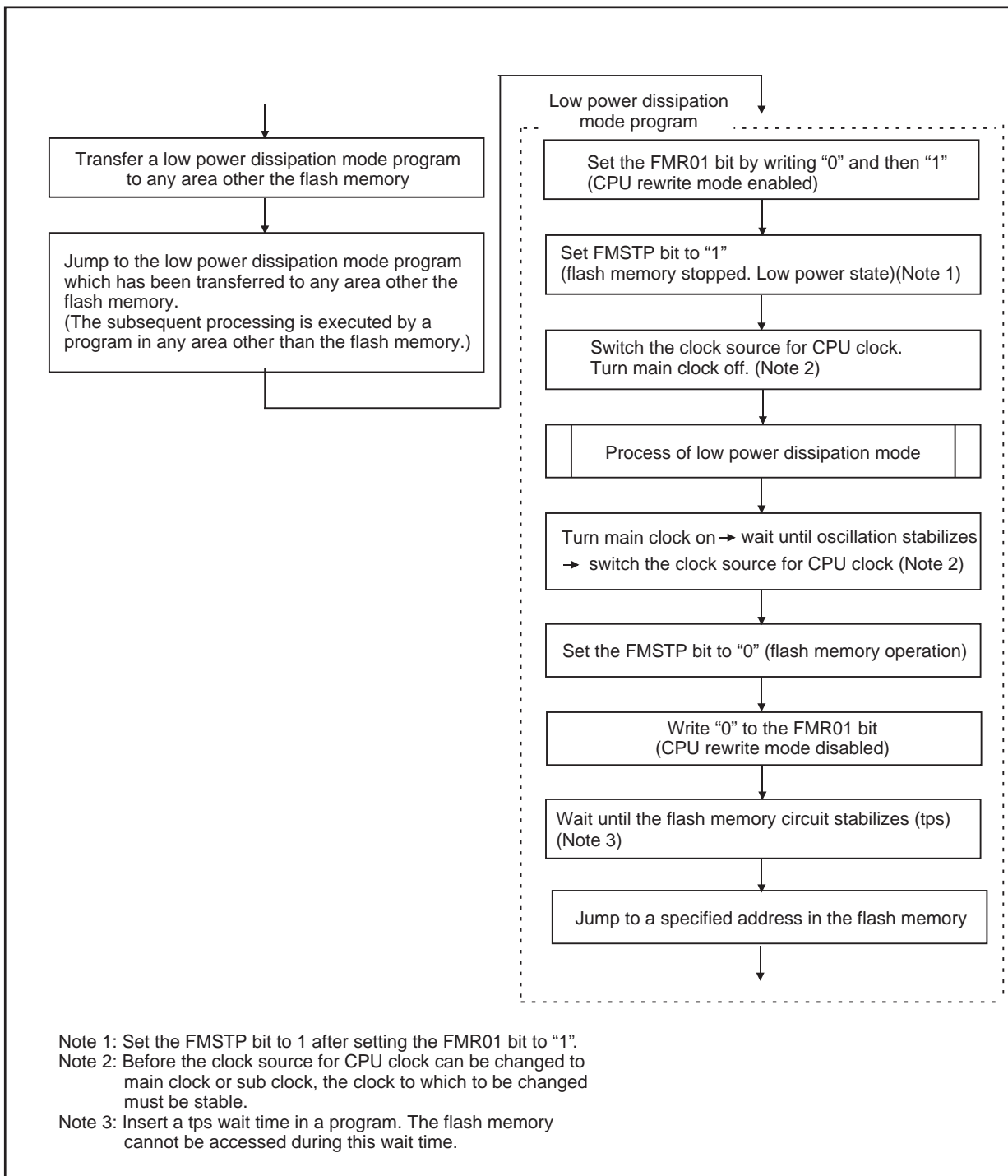


Figure 4.2.7. Processing Before and After Low Power Dissipation Mode

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

Before entering CPU rewrite mode (EW0 or EW1 mode), select 10 MHz or less for BCLK using the CM06 bit in the CM0 register and the CM17 to CM16 bits in the CM1 register. Also, set the PM17 bit in the PM1 register to "1" (with wait state).

(2) Instructions to Prevent from Using

The following instructions cannot be used in EW0 mode because the flash memory's internal data is referenced: UND instruction, INTO instruction, JMPS instruction, JSRS instruction, and BRK instruction

(3) Interrupts

EW0 Mode

- Any interrupt which has a vector in the variable vector table can be used providing that its vector is transferred into the RAM area.
- The $\overline{\text{NMI}}$ and watchdog timer interrupts can be used because the FMR0 register and FMR1 register are initialized when one of those interrupts occurs. The jump addresses for those interrupt service routines should be set in the fixed vector table.

Because the rewrite operation is halted when a $\overline{\text{NMI}}$ or watchdog timer interrupt occurs, the rewrite program must be executed again after exiting the interrupt service routine.

- The address match interrupt cannot be used because the flash memory's internal data is referenced.

EW1 Mode

- Make sure that any interrupt which has a vector in the variable vector table or address match interrupt will not be accepted during the auto program or auto erase period.
- Avoid using watchdog timer interrupts.
- The $\overline{\text{NMI}}$ interrupt can be used because the FMR0 register and FMR1 register are initialized when this interrupt occurs. The jump address for the interrupt service routine should be set in the fixed vector table.

Because the rewrite operation is halted when a $\overline{\text{NMI}}$ interrupt occurs, the rewrite program must be executed again after exiting the interrupt service routine.

(4) How to Access

To set the FMR01, FMR02, or FMR11 bit to "1", write "0" and then "1" in succession. This is necessary to ensure that no interrupts or DMA transfers will occur before writing "1" after writing "0". Also only when $\overline{\text{NMI}}$ pin is "H" level.

(5) Writing in the User ROM Space

EW0 Mode

- If the power supply voltage drops while rewriting any block in which the rewrite control program is stored, a problem may occur that the rewrite control program is not correctly rewritten and, consequently, the flash memory becomes unable to be rewritten thereafter. In this case, standard serial I/O or parallel I/O mode should be used.

EW1 Mode

- Avoid rewriting any block in which the rewrite control program is stored.

(6) DMA Transfer

In EW1 mode, make sure that no DMA transfers will occur while the FMR0 register's FMR00 bit = 0 (during the auto program or auto erase period).

(7) Writing Command and Data

Write the command code and data at even addresses.

(8) Wait Mode

When shifting to wait mode, set the FMR01 bit to "0" (CPU rewrite mode disabled) before executing the WAIT instruction.

(9) Stop Mode

When shifting to stop mode, the following settings are required:

- Set the FMR01 bit to "0" (CPU rewrite mode disabled) and disable DMA transfers before setting the CM10 bit to "1" (stop mode).
- Execute the JMP.B instruction subsequent to the instruction which sets the CM10 bit to "1" (stop mode)

```
Example program  BSET      0, CM1      ; Stop mode
                  JMP.B    L1
```

L1:

Program after returning from stop mode

(10) Low Power Dissipation Mode

If the CM05 bit is set to "1" (main clock stop), the following commands must not be executed.

- Program
- Block erase
- Lock bit program

4.3 Software Commands

Software commands are described below. The command code and data must be read and written in 16-bit units, to and from even addresses in the user ROM area. When writing command code, the 8 high-order bits (D₁₁–D₈) are ignored.

Table 4.3.1. Software Commands

Command	First bus cycle			Second bus cycle		
	Mode	Address	Data (D ₀ to D ₇)	Mode	Address	Data (D ₀ to D ₇)
Read array	Write	X	xxFF ₁₆			
Read status register	Write	X	xx70 ₁₆	Read	X	SRD
Clear status register	Write	X	xx50 ₁₆			
Program	Write	WA	xx40 ₁₆	Write	WA	WD
Block erase	Write	X	xx20 ₁₆	Write	BA	xxD0 ₁₆
Lock bit program	Write	BA	xx77 ₁₆	Write	BA	xxD0 ₁₆
Read lock bit status	Write	X	xx71 ₁₆	Write	BA	xxD0 ₁₆

SRD: Status register data (D₇ to D₀)

WA: Write address (Make sure the address value specified in the the first bus cycle is the same even address as the write address specified in the second bus cycle.)

WD: Write data (16 bits)

BA: Uppermost block address (even address, however)

X: Any even address in the user ROM area

x: High-order 8 bits of command code (ignored)

Read Array Command (FF₁₆)

This command reads the flash memory.

Writing 'xxFF₁₆' in the first bus cycle places the microcomputer in read array mode. Enter the read address in the next or subsequent bus cycles, and the content of the specified address can be read in 16-bit units.

Because the microcomputer remains in read array mode until another command is written, the contents of multiple addresses can be read in succession.

Read Status Register Command (70₁₆)

This command reads the status register.

Write 'xx70₁₆' in the first bus cycle, and the status register can be read in the second bus cycle. (Refer to "Status Register.") When reading the status register too, specify an even address in the user ROM area.

Do not execute this command in EW1 mode.

Clear Status Register Command

This command clears the status register to "0".

Write 'xx5016' in the first bus cycle, and the FMR06 to FMR07 bits in the FMR0 register and SR4 to SR5 in the status register will be cleared to "0".

Program Command

This command writes data to the flash memory in 1 word (2 byte) units.

Write 'xx4016' in the first bus cycle and write data to the write address in the second bus cycle, and an auto program operation (data program and verify) will start. Make sure the address value specified in the first bus cycle is the same even address as the write address specified in the second bus cycle.

Check the FMR00 bit in the FMR0 register to see if auto programming has finished. The FMR00 bit is "0" during auto programming and set to "1" when auto programming is completed.

Check the FMR06 bit in the FMR0 register after auto programming has finished, and the result of auto programming can be known. (Refer to "Full Status Check.")

Each block can be protected against programming by a lock bit. (Refer to "Data Protect Function.")

Be careful not to write over the already programmed addresses.

In EW1 mode, do not execute this command on any address at which the rewrite control program is located.

In EW0 mode, the microcomputer goes to read status register mode at the same time auto programming starts, making it possible to read the status register. The status register bit 7 (SR7) is cleared to "0" at the same time auto programming starts, and set back to "1" when auto programming finishes. In this case, the microcomputer remains in read status register mode until a read command is written next. The result of auto programming can be known by reading the status register after auto programming has finished.

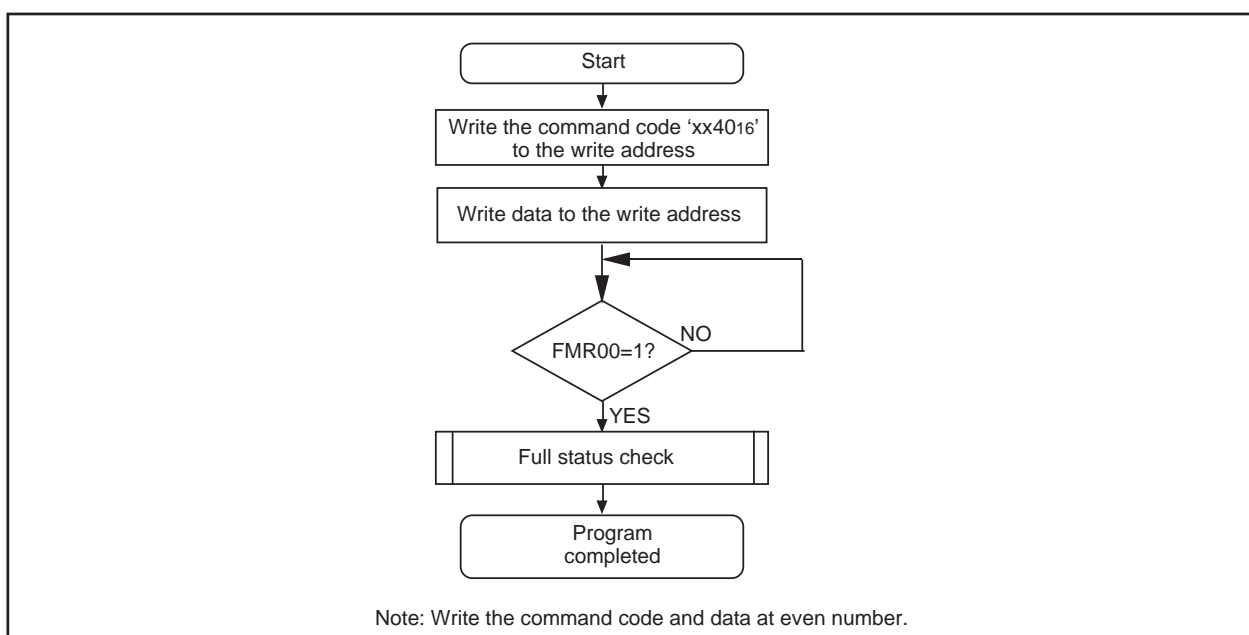


Figure 4.3.1. Program Command

Block Erase

Write 'xx2016' in the first bus cycle and write 'xxD016' to the uppermost address of a block (even address, however) in the second bus cycle, and an auto erase operation (erase and verify) will start. Check the FMR0 register's FMR00 bit to see if auto erasing has finished.

The FMR00 bit is "0" during auto erasing and set to "1" when auto erasing is completed.

Check the FMR0 register's FMR07 bit after auto erasing has finished, and the result of auto erasing can be known. (Refer to "Full Status Check.")

Figure 4.3.2 shows an example of a block erase flowchart.

Each block can be protected against erasing by a lock bit. (Refer to "Data Protect Function.")

In EW1 mode, do not execute this command on any address at which the rewrite control program is located.

In EW0 mode, the microcomputer goes to read status register mode at the same time auto erasing starts, making it possible to read the status register. The status register bit 7 (SR7) is cleared to "0" at the same time auto erasing starts, and set back to "1" when auto erasing finishes. In this case, the microcomputer remains in read status register mode until the Read Array or Read Lock Bit Status command is written next.

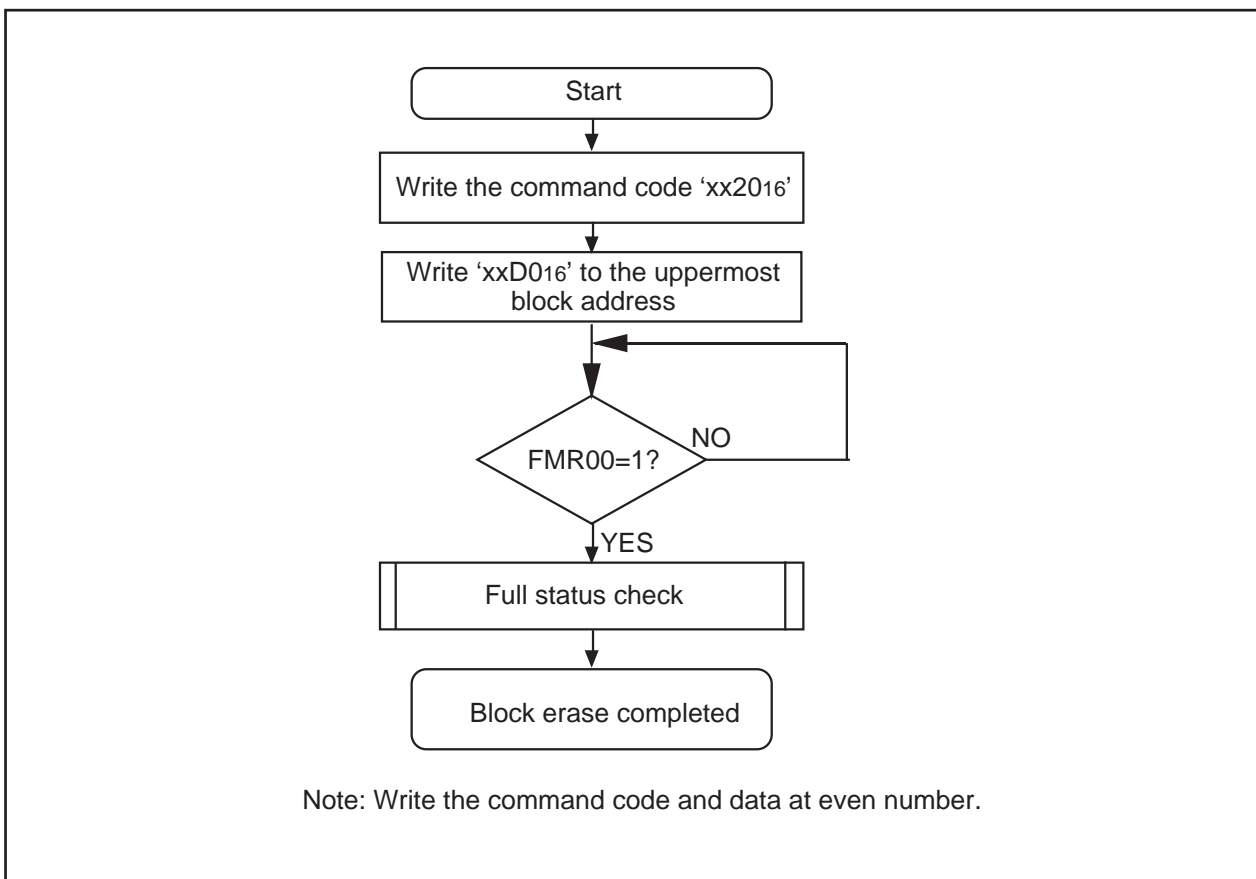


Figure 4.3.2. Block Erase Command

Lock Bit Program Command

This command sets the lock bit for a specified block to “0” (locked).

Write ‘xx7716’ in the first bus cycle and write ‘xxD016’ to the uppermost address of a block (even address, however) in the second bus cycle, and the lock bit for the specified block is cleared to “0”. Make sure the address value specified in the first bus cycle is the same uppermost block address that is specified in the second bus cycle.

Figure 4.3.3 shows an example of a lock bit program flowchart. The lock bit status (lock bit data) can be read using the Read Lock Bit Status command.

Check the FMR0 register’s FMR00 bit to see if writing has finished.

For details about the lock bit function, and on how to set the lock bit to “1”, refer to “Data Protect Function.”

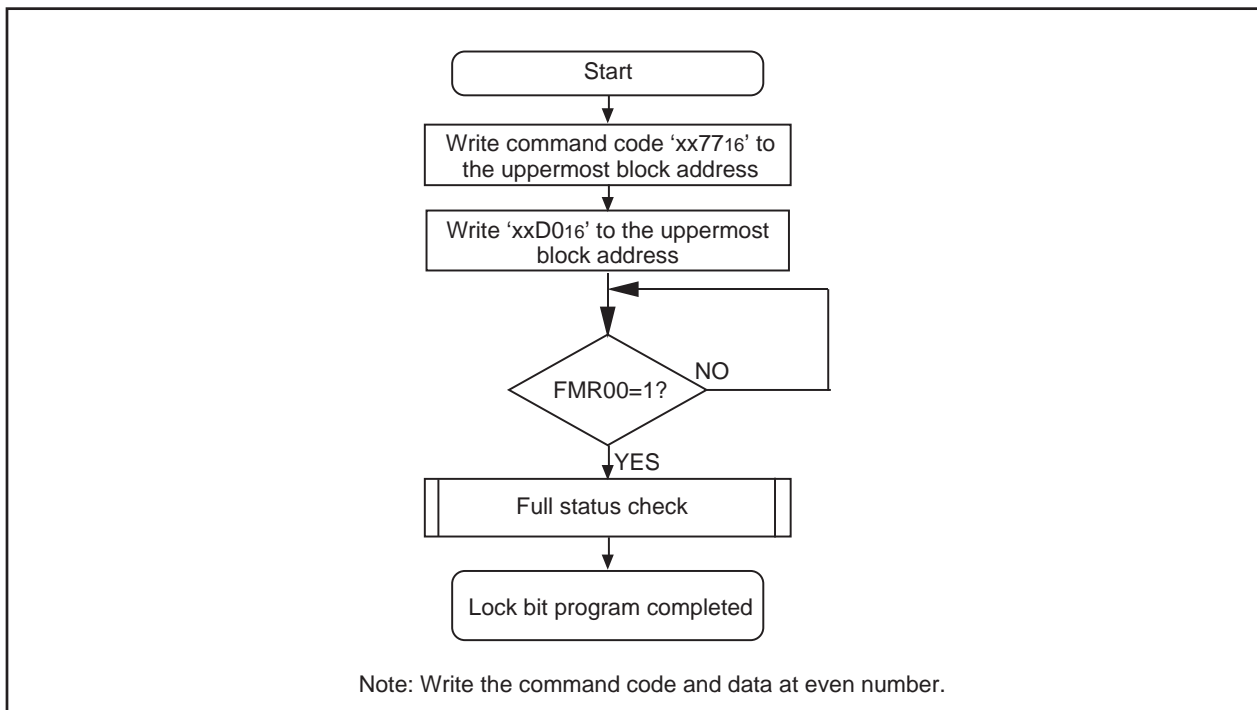


Figure 4.3.3. Lock Bit Program Command

Read Lock Bit Status Command (7116)

This command reads the lock bit status of a specified block.

Write 'xx7116' in the first bus cycle and write 'xxD016' to the uppermost address of a block (even address, however) in the second bus cycle, and the lock bit status of the specified block is stored in the FMR1 register's FMR16 bit. Read the FMR16 bit after the FMR0 register's FMR00 bit is set to "1" (ready).

Figure 4.3.4 shows an example of a read lock bit status flowchart.

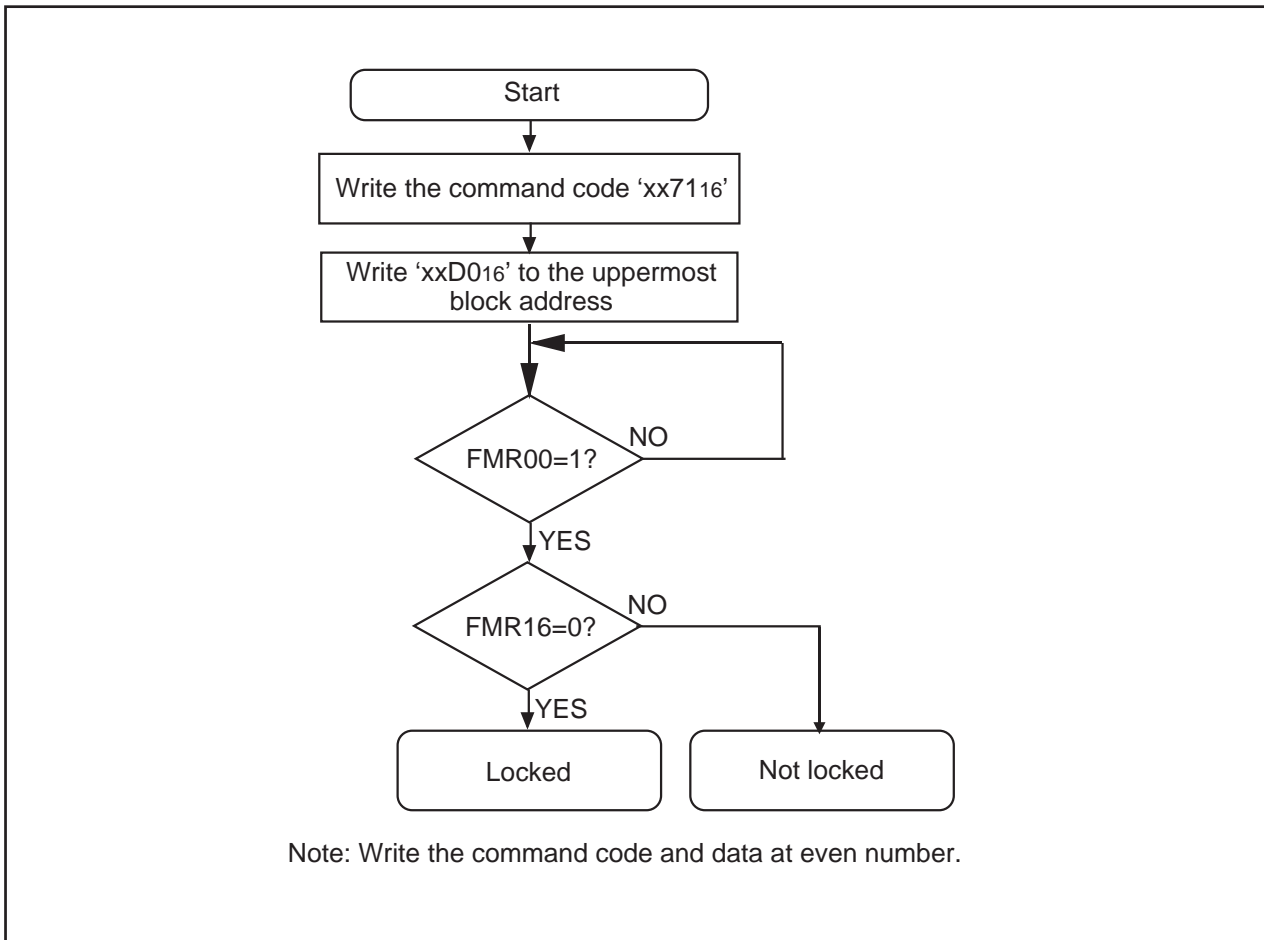


Figure 4.3.4. Read Lock Bit Status Command

Data Protect Function

Each block in the flash memory has a nonvolatile lock bit. The lock bit is effective when the FMR02 bit = 0 (lock bit enabled). The lock bit allows each block to be individually protected (locked) against programming and erasure. This helps to prevent data from inadvertently written to or erased from the flash memory. The following shows the relationship between the lock bit and the block status.

- When the lock bit = 0, the block is locked (protected against programming and erasure).
- When the lock bit = 1, the block is not locked (can be programmed or erased).

The lock bit is cleared to “0” (locked) by executing the Lock Bit Program command, and is set to “1” (unlocked) by erasing the block. The lock bit cannot be set to “1” by a command.

The lock bit status can be read using the Read Lock Bit Status command

The lock bit function is disabled by setting the FMR02 bit to “1”, with all blocks placed in an unlocked state. (The lock bit data itself does not change state.) Setting the FMR02 bit to “0” enables the lock bit function (lock bit data retained).

If the Block Erase command is executed while the FMR02 bit = 1, the target block or all blocks are erased irrespective of how the lock bit is set. The lock bit for each block is set to “1” after completion of erasure. For details about the commands, refer to “Software Commands.”

Status Register

The status register indicates the operating status of the flash memory and whether an erase or programming operation terminated normally or in error. The status of the status register can be known by reading the FMR0 register’s FMR00, FMR06, and FMR07 bits.

Table 4.3.2 shows the status register.

In EW0 mode, the status register can be read in the following cases:

- (1) When a given even address in the user ROM area is read after writing the Read Status Register command
- (2) When a given even address in the user ROM area is read after executing the Program, Block Erase, or Lock Bit Program command but before executing the Read Array command.

Sequencer Status (SR7 and FMR00 Bits)

The sequence status indicates the operating status of the flash memory. SR7 = 0 (busy) during auto programming, auto erase, and lock bit write, and is set to “1” (ready) at the same time the operation finishes.

Erase Status (SR5 and FMR07 Bits)

Refer to “Full Status Check.”

Program Status (SR4 and FMR06 Bits)

Refer to “Full Status Check.”

Table 4.3.2. Status Register

Status register bit	FMR0 register bit	Status name	Contents		Value after reset
			"0"	"1"	
SR7 (D7)	FMR00	Sequencer status	Busy	Ready	1
SR6 (D6)	—	Reserved	-	-	—
SR5 (D5)	FMR07	Erase status	Terminated normally	Terminated in error	0
SR4 (D4)	FMR06	Program status	Terminated normally	Terminated in error	0
SR3 (D3)	—	Reserved	-	-	—
SR2 (D2)	—	Reserved	-	-	—
SR1 (D1)	—	Reserved	-	-	—
SR0 (D0)	—	Reserved	-	-	—

- D0 to D7: Indicates the data bus which is read out when the Read Status Register command is executed.
- The FMR07 bit (SR5) and FMR06 bit (SR4) are cleared to "0" by executing the Clear Status Register command.
- When the FMR07 bit (SR5) or FMR06 bit (SR4) = 1, the Program, Block Erase, and Lock Bit Program commands are not accepted.

Full Status Check

When an error occurs, the FMR0 register's FMR06 to FMR07 bits are set to "1", indicating occurrence of each specific error. Therefore, execution results can be verified by checking these status bits (full status check). Table 4.3.3 lists errors and FMR0 register status. Figure 4.3.5 shows a full status check flowchart and the action to be taken when each error occurs.

Table 4.3.3. Errors and FMR0 Register Status

FRM00 register (status register) status		Error	Error occurrence condition
FMR07 (SR5)	FMR06 (SR4)		
1	1	Command sequence error	<ul style="list-style-type: none"> • When any command is not written correctly • When invalid data was written other than those that can be written in the second bus cycle of the Lock Bit Program or Block Erase command (i.e., other than 'xD016' or 'xFF16') (Note 1)
1	0	Erase error	<ul style="list-style-type: none"> • When the Block Erase command was executed on locked blocks (Note 2) • When the Block Erase command was executed on unlocked blocks but the blocks were not automatically erased correctly
0	1	Program error	<ul style="list-style-type: none"> • When the Block Erase command was executed on locked blocks (Note 2) • When the Program command was executed on unlocked blocks but the blocks were not automatically programmed correctly. • When the Lock Bit Program command was executed but not programmed correctly

Note 1: If "xFF16" is written by the 2nd bus cycle of these commands, it will become lead array mode and the command code written by the 1st bus cycle will become invalid simultaneously.

Note 2: When FMR02 bit is "1" (lock bit is invalid), an error is not generated on these conditions.

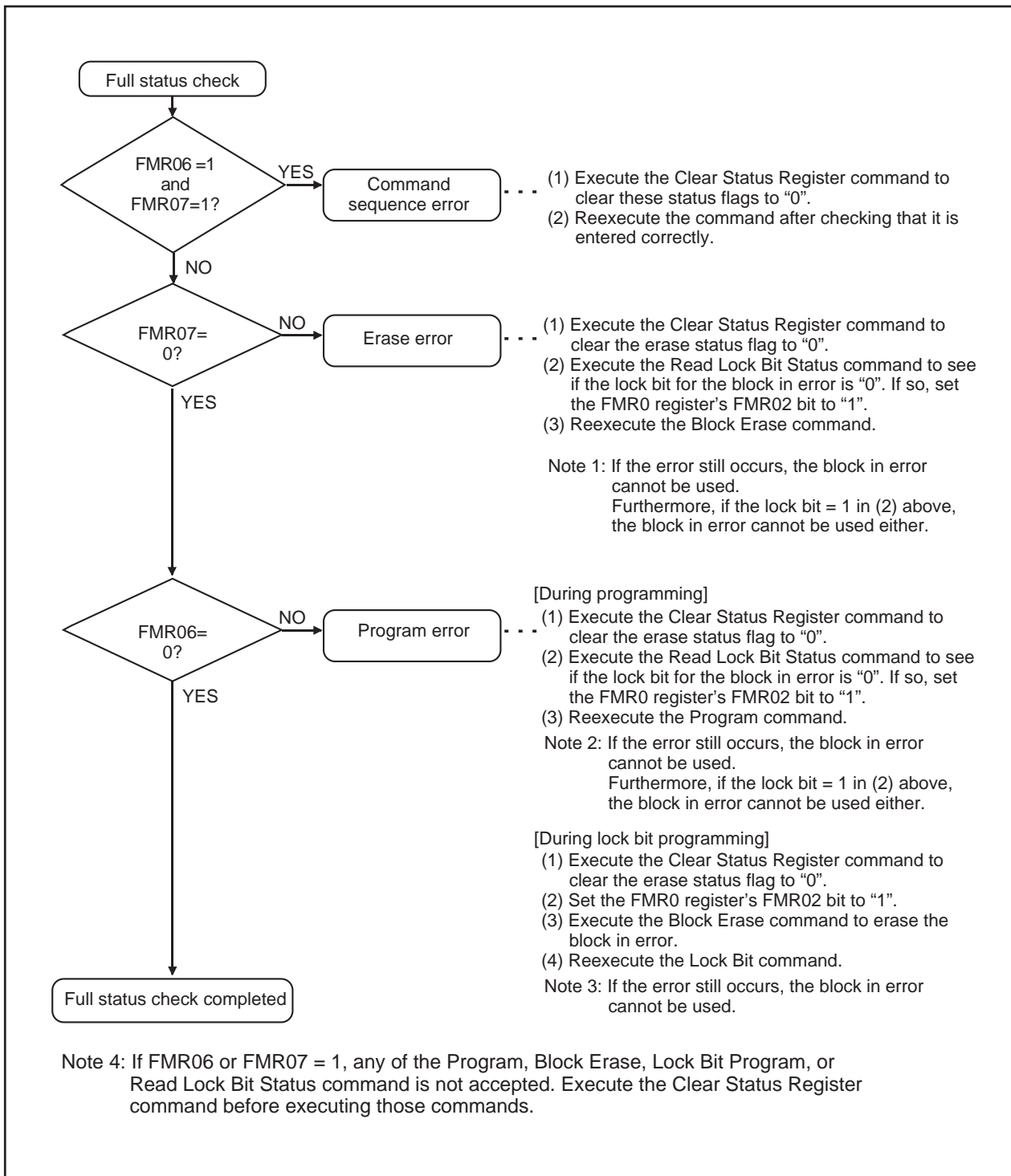


Figure 4.3.5. Full Status Check and Handling Procedure for Each Error

Standard Serial I/O Mode

In standard serial input/output mode, the user ROM area can be rewritten while the microcomputer is mounted on-board by using a serial programmer suitable for M306H5FGFP. For more information about serial programmers, contact the manufacturer of your serial programmer. For details on how to use, refer to the user's manual included with your serial programmer.

Table 4.3.4 lists pin functions (flash memory standard serial input/output mode). Figures 4.3.7 show pin connections for serial input/output mode.

ID Code Check Function

This function determines whether the ID codes sent from the serial programmer and those written in the flash memory match. (Refer to the description of the functions to inhibit rewriting flash memory version.)

Table 4.3.4. Pin Functions (Flash Memory Standard Serial I/O Mode)

Pin	Name	I/O	Description
VCC1, VCC2, VSS	Power input		Apply 4.75 V to 5.25 V to Vcc2 pin, and Vcc1 ($V_{cc1} \leq V_{cc2}$) to Vcc1 pin.
CNVss	CNVss	I	Connect to Vcc2 pin.
RESET	Reset input	I	Reset input pin. While RESET pin is "L" level, input a 20 cycle or longer clock to XIN pin.
M1	Mode select	I	Connect to Vss pin.
START	Oscillation selection input	I	Connect to Vcc2 pin.
XIN	Clock input	I	Connect a ceramic resonator or crystal oscillator between XIN and XOUT pins. To input an externally generated clock, input it to XIN pin and open XOUT pin.
XOUT	Clock output	O	
BYTE	BYTE	I	Connect this pin to Vcc or Vss.
AVcc, AVss	Analog power supply input		Connect AVss to Vss and AVcc to Vcc2, respectively Apply Vcc2 to AVcc pin and 0V to AVSS pin..
VREF	Reference voltage input	I	Enter the reference voltage for AD from this pin.
P00 to P07	Input port P0	I	Input "H" or "L" level signal or open.
P10 to P17	Input port P1	I	Input "H" or "L" level signal or open.
P20 to P27	Input port P2	I	Input "H" or "L" level signal or open.
P30 to P37	Input port P3	I	Input "H" or "L" level signal or open.
P40 to P47	Input port P4	I	Input "H" or "L" level signal or open.
P51 to P57	Input port P5	I	Input "H" or "L" level signal or open.
P50	CE input	I	Input "H" level signal.
P60 to P63	Input port P6	I	Input "H" or "L" level signal or open.
P64/RTS1	BUSY output	O	Standard serial I/O mode 1: BUSY signal output pin Standard serial I/O mode 2: Monitors the boot program operation check signal output pin.
P65/CLK1	SCLK input	I	Standard serial I/O mode 1: Serial clock input pin Standard serial I/O mode 2: Input "L".
P66/RXD1	RxD input	I	Serial data input pin
P67/TXD1	TxD output	O	Serial data output pin (Note 1)
P70 to P77	Input port P7	I	Input "H" or "L" level signal or open.
P80 to P84, P86, P87	Input port P8	I	Input "H" or "L" level signal or open.
P85/NM1	NMI input	I	Connect this pin to Vcc2.
P90 to P97	Input port P9	I	Input "H" or "L" level signal or open.
P100 to P107	Input port P10	I	Input "H" or "L" level signal or open.
P11	Output port P11	O	Open
VDD2, Vss2	Power input		Connect VDD2 pin to Vcc2 and connect VSS2 pin to Vss. Apply Vcc2 to VDD2 pin and 0V to VSS2 pin.
VDD3, Vss3	Power input		Connect VDD3 pin to Vcc2 and connect VSS3 pin to Vss. Apply Vcc2 to VDD3 pin and 0V to VSS3 pin.
LP2 to LP4	Filter output	O	Open
TEST3	Vcc1 Power supply switching	I	Input "L" level signal.
CVIN1, SYNCIN	Compound video input	I	Input "H" or "L" level signal or open.
SVREF	Synchronous slice level input	I	A slice potential input pin in slicing a synchronized signal.

Note 1: When using standard serial input/output mode 1, the TxD pin must be held high while the RESET pin is pulled low. Therefore, connect this pin to VCC1 via a resistor. Because this pin is directed for data output after reset, adjust the pull-up resistance value in the system so that data transfers will not be affected.

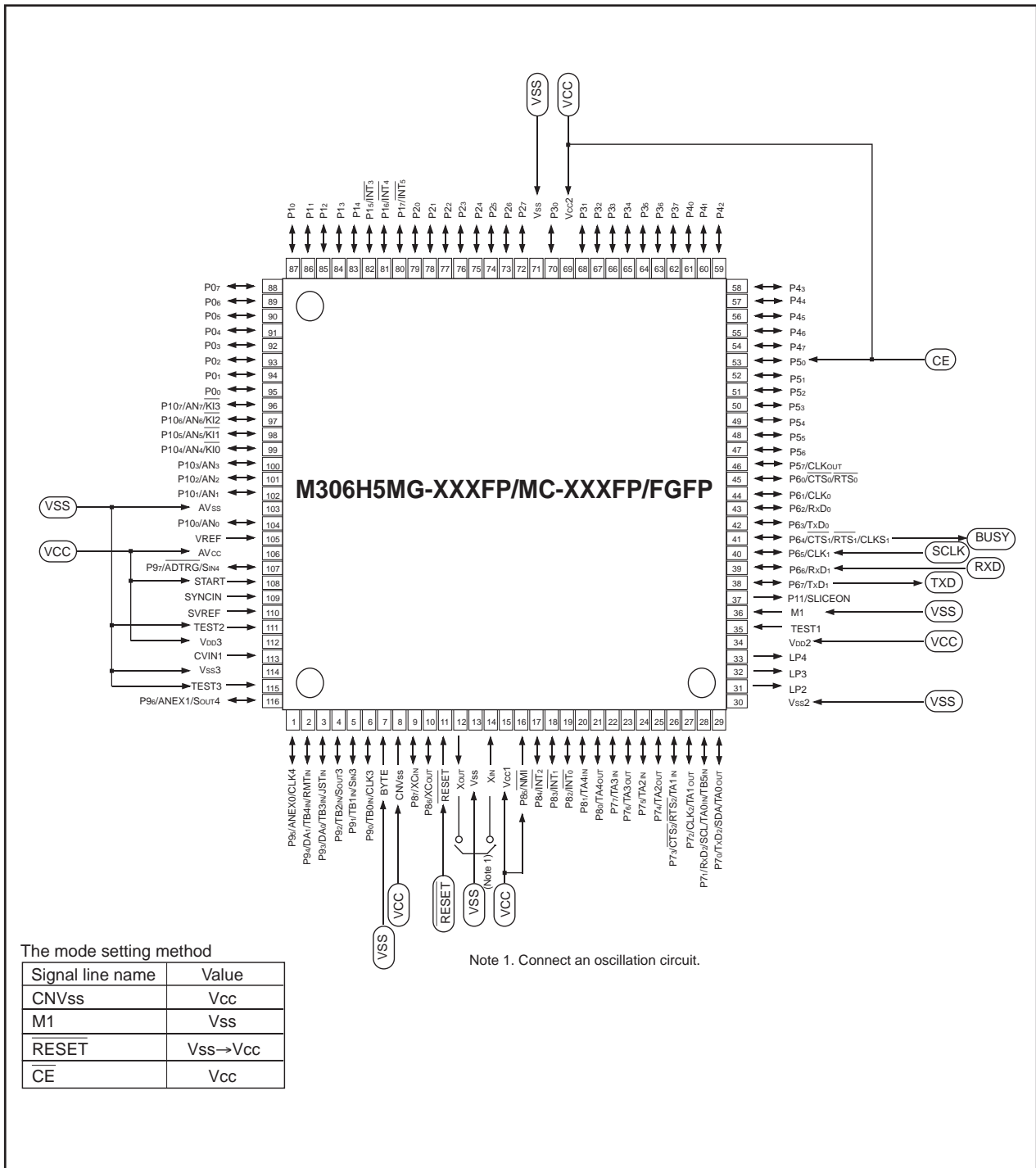


Figure 4.3.6. Pin Connections for Serial I/O Mode

Example of Circuit Application in the Standard Serial I/O Mode

Figure 4.3.7 and 4.3.8 show example of circuit application in standard serial I/O mode 1 and mode 2, respectively. Refer to the user's manual for serial writer to handle pins controlled by a serial writer.

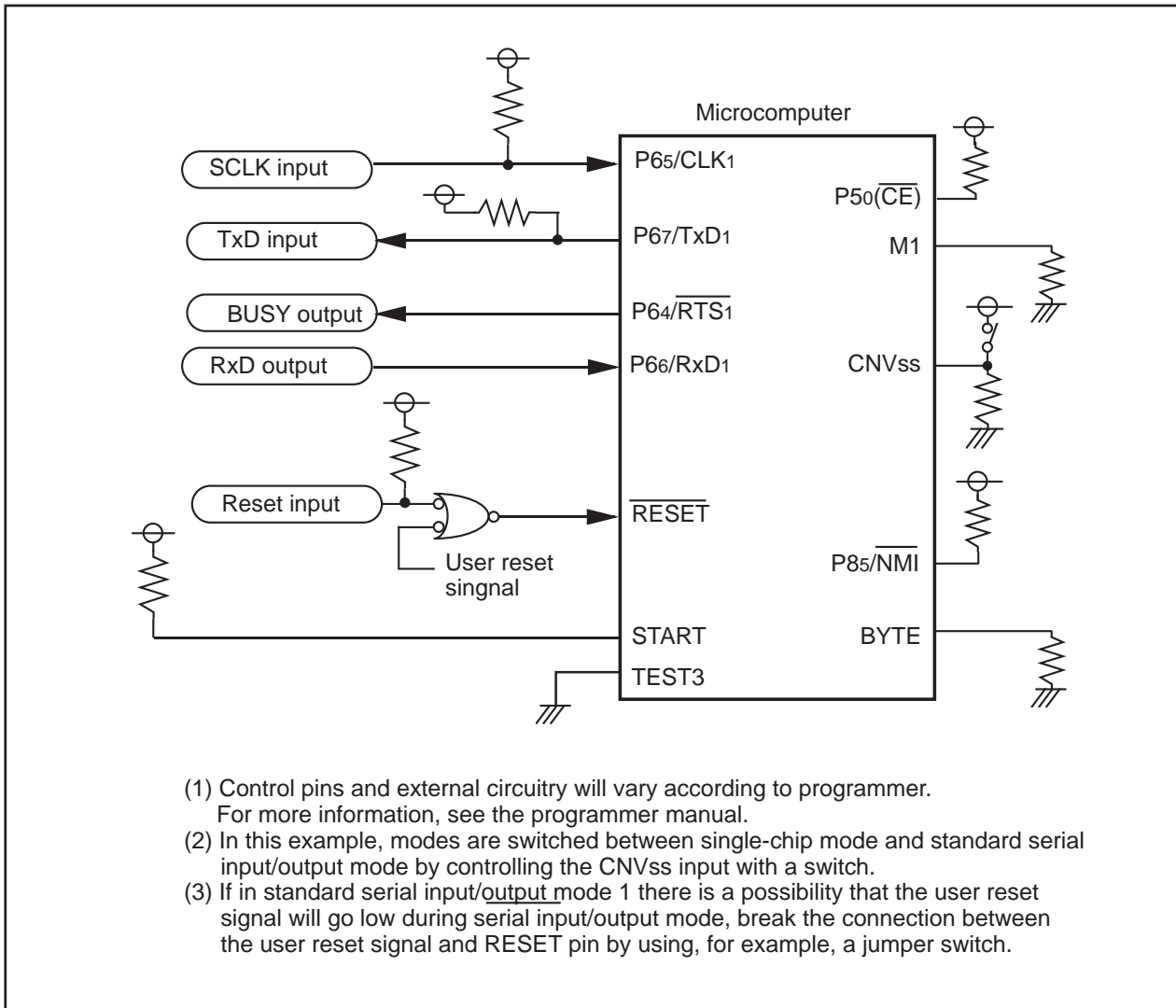


Figure 4.3.7. Circuit Application in Standard Serial I/O Mode 1

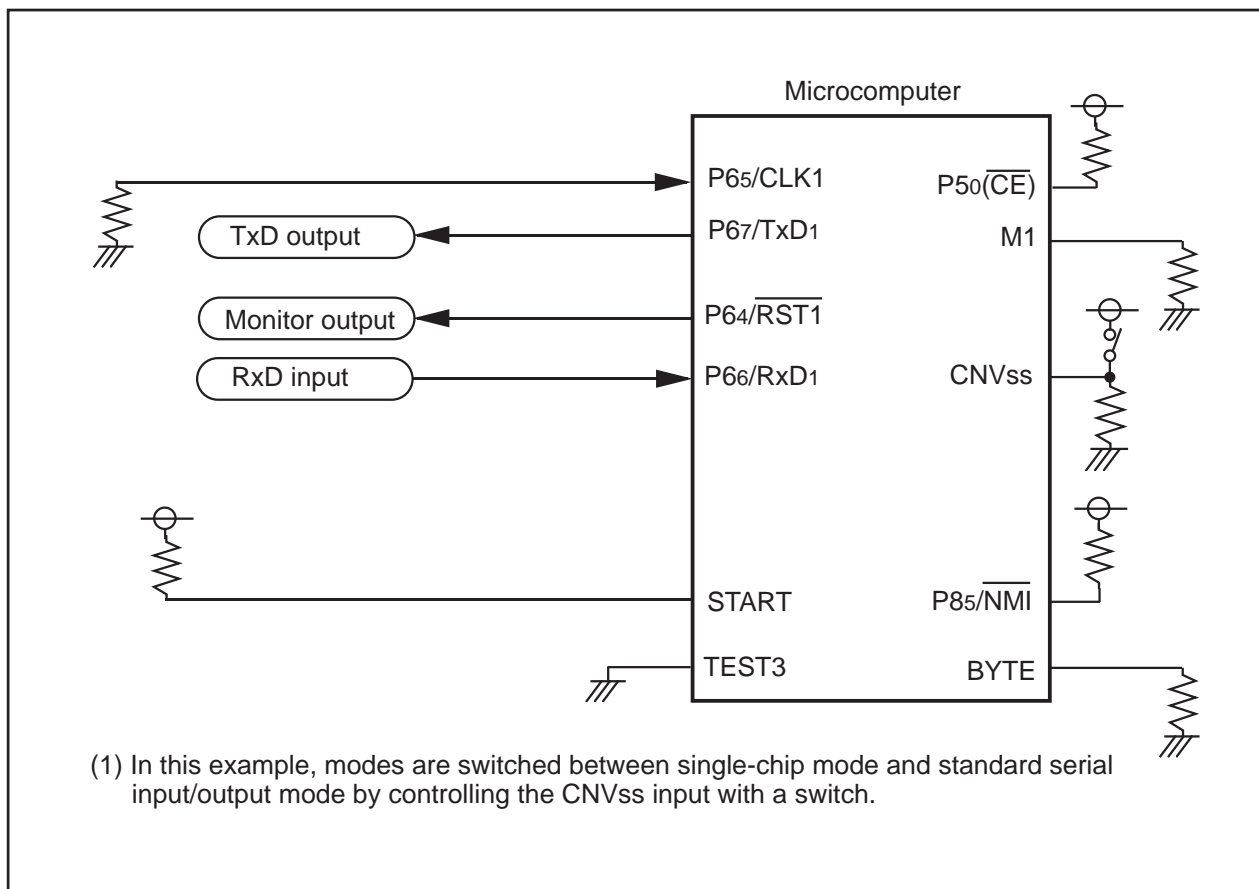


Figure 4.3.8. Circuit Application in Standard Serial I/o Mode 2

Parallel I/O Mode

In parallel input/output mode, the user ROM and boot ROM areas can be rewritten by using a parallel programmer suitable for the M16C/62P group. For more information about parallel programmers, contact the manufacturer of your parallel programmer. For details on how to use, refer to the user's manual included with your parallel programmer.

User ROM and Boot ROM Areas

In the boot ROM area, an erase block operation is applied to only one 4 Kbyte block. The boot ROM area contains a standard serial input/output mode based rewrite control program which was written in it when shipped from the factory. Therefore, when using a serial programmer, be careful not to rewrite the boot ROM area.

When in parallel output mode, the boot ROM area is located at addresses 0FF000₁₆ to 0FFFFFF₁₆. When rewriting the boot ROM area, make sure that only this address range is rewritten. (Do not access other than the addresses 0FF000₁₆ to 0FFFFFF₁₆.)

ROM Code Protect Function

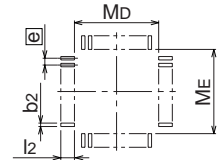
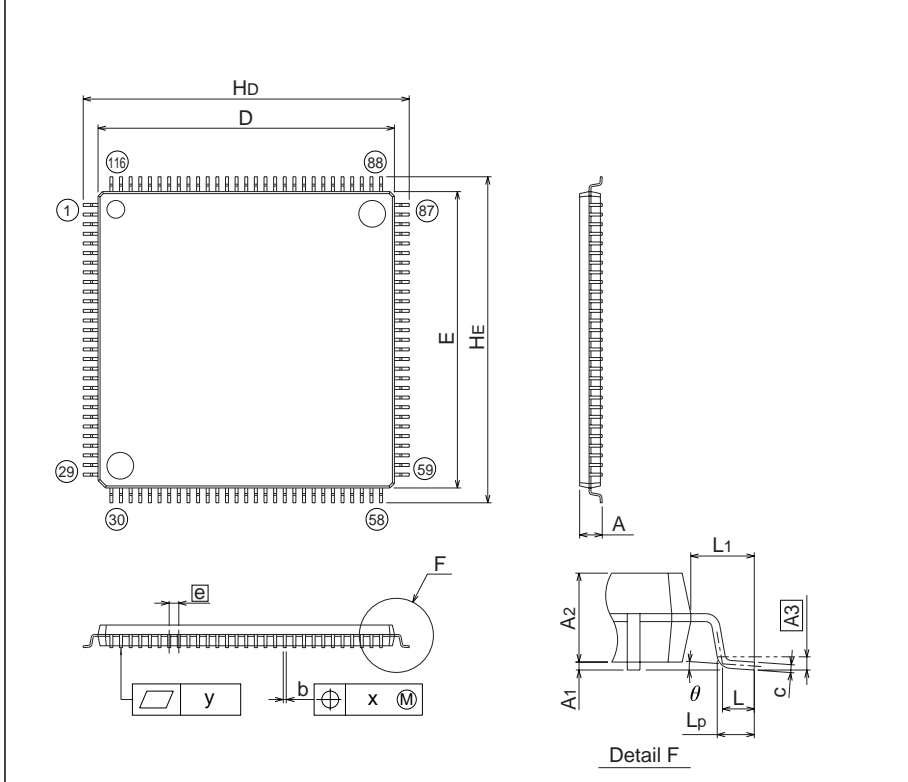
The ROM code protect function inhibits the flash memory from being read or rewritten. (Refer to the description of the functions to inhibit rewriting flash memory version.)

5. PACKAGE OUTLINE

116P6A-A (MMP)

Plastic 116pin 20X20mm body LQFP

EIAJ Package Code	JEDEC Code	Weight(g)	Lead Material
LQFP116-P-2020-0.65	-		Cu Alloy



Recommended Mount Pad

Symbol	Dimension in Millimeters		
	Min	Nom	Max
A	-	-	1.7
A1	0.05	0.125	0.2
A2	-	1.4	-
b	0.17	0.22	0.27
c	0.105	0.125	0.175
D	19.9	20.0	20.1
E	19.9	20.0	20.1
e	-	0.65	-
Hd	21.8	22.0	22.2
HE	21.8	22.0	22.2
L	0.35	0.5	0.65
L1	-	1.0	-
Lp	0.45	0.6	0.75
A3	-	0.25	-
x	-	-	0.13
y	-	-	0.1
theta	0°	-	8°
b2	-	0.225	-
l2	0.95	-	-
MD	-	20.4	-
ME	-	20.4	-

6. USEGE NOTES

Precautions for External Bus

1. In the mask ROM version, connect the CNVss pin to the VCC2 when use microprocessor mode or memory expansion mode.
In the flash memory version, connect the CNVss pin and the M1 pin to the VCC2.
2. In the mask ROM version, contents of internal ROM cannot be read out when resetting the CNVss pin with "H" input. In the flash memory version, contents of internal ROM cannot be read out when resetting the CNVss pin and the M1 pin with "H" input.

Precautions for Power Control

1. When exiting stop mode by hardware reset, set $\overline{\text{RESET}}$ pin to "L" until a main clock oscillation is stabilized.
2. Insert more than four NOP instructions after an WAIT instruction or a instruction to set the CM10 bit of CM1 register to "1". When shifting to wait mode or stop mode, an instruction queue reads ahead to the next instruction to halt a program by an WAIT instruction and an instruction to set the CM10 bit to "1" (all clocks stopped). The next instruction may be executed before entering wait mode or stop mode, depending on a combination of instruction and an execution timing.
3. Wait until the $t_{d(M-L)}$ elapses or main clock oscillation stabilization time, whichever is longer, before switching the clock source for CPU clock to the main clock.
Similarly, wait until the sub clock oscillates stably before switching the clock source for CPU clock to the sub clock.
4. Suggestions to reduce power consumption

(a) Ports

The processor retains the state of each I/O port even when it goes to wait mode or to stop mode. A current flows in active I/O ports. A pass current flows in input ports that high-impedance state. When entering wait mode or stop mode, set non-used ports to input and stabilize the potential.

(b) A-D converter

When A-D conversion is not performed, set the VCUT bit of ADiCON1 register to "0" (no VREF connection). When A-D conversion is performed, start the A-D conversion at least 1 μs or longer after setting the VCUT bit to "1" (VREF connection).

(c) Stopping peripheral functions

Use the CM0 register CM02 bit to stop the unnecessary peripheral functions during wait mode.

However, because the peripheral function clock (fc32) generated from the sub-clock does not stop, this measure is not conducive to reducing the power consumption of the chip. If low speed mode or low power dissipation mode is to be changed to wait mode, set the CM02 bit to "0" (do not peripheral function clock stopped when in wait mode), before changing wait mode.

(d) Switching the oscillation-driving capacity

Set the driving capacity to "LOW" when oscillation is stable.

(e) External clock

When using an external clock input for the CPU clock, set the CM0 register CM05 bit to "1" (stop). Setting the CM05 bit to "1" disables the XOUT pin from functioning, which helps to reduce the amount of current drawn in the chip. (When using an external clock input, note that the clock remains fed into the chip regardless of how the CM05 bit is set.)

Precautions for Protect

Set the PRC2 bit to "1" (write enabled) and then write to any address, and the PRC2 bit will be cleared to "0" (write protected). The registers protected by the PRC2 bit should be changed in the next instruction after setting the PRC2 bit to "1". Make sure no interrupts or DMA transfers will occur between the instruction in which the PRC2 bit is set to "1" and the next instruction.

Precautions for Interrupts

Reading address 00000₁₆

Do not read the address 00000₁₆ in a program. When a maskable interrupt request is accepted, the CPU reads interrupt information (interrupt number and interrupt request priority level) from the address 00000₁₆ during the interrupt sequence. At this time, the IR bit for the accepted interrupt is cleared to "0".

If the address 00000₁₆ is read in a program, the IR bit for the interrupt which has the highest priority among the enabled interrupts is cleared to "0". This causes a problem that the interrupt is canceled, or an unexpected interrupt request is generated.

Setting the SP

Set any value in the SP(USP, ISP) before accepting an interrupt. The SP(USP, ISP) is cleared to '0000₁₆' after reset. Therefore, if an interrupt is accepted before setting any value in the SP(USP, ISP), the program may go out of control.

Especially when using $\overline{\text{NMI}}$ interrupt, set a value in the ISP at the beginning of the program. For the first and only the first instruction after reset, all interrupts including $\overline{\text{NMI}}$ interrupt are disabled.

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

1. The $\overline{\text{NMI}}$ interrupt cannot be disabled. If this interrupt is unused, connect the $\overline{\text{NMI}}$ pin to Vcc via a resistor (pull-up).
2. The input level of the $\overline{\text{NMI}}$ pin can be read by accessing the P8 register's P8_5 bit. Note that the P8_5 bit can only be read when determining the pin level in $\overline{\text{NMI}}$ interrupt routine.
3. Stop mode cannot be entered into while input on the $\overline{\text{NMI}}$ pin is low. This is because while input on the $\overline{\text{NMI}}$ pin is low the CM1 register's CM10 bit is fixed to "0".
4. Do not go to wait mode while input on the $\overline{\text{NMI}}$ pin is low. This is because when input on the $\overline{\text{NMI}}$ pin goes low, the CPU stops but CPU clock remains active; therefore, the current consumption in the chip does not drop. In this case, normal condition is restored by an interrupt generated thereafter.
5. The low and high level durations of the input signal to the $\overline{\text{NMI}}$ pin must each be 2 CPU clock cycles + 300 ns or more.

Changing the Interrupt Generate Factor

If the interrupt generate factor is changed, the IR bit in the interrupt control register for the changed interrupt may inadvertently be set to “1” (interrupt requested). If you changed the interrupt generate factor for an interrupt that needs to be used, be sure to clear the IR bit for that interrupt to “0” (interrupt not requested).

“Changing the interrupt generate factor” referred to here means any act of changing the source, polarity or timing of the interrupt assigned to each software interrupt number. Therefore, if a mode change of any peripheral function involves changing the generate factor, polarity or timing of an interrupt, be sure to clear the IR bit for that interrupt to “0” (interrupt not requested) after making such changes. Refer to the description of each peripheral function for details about the interrupts from peripheral functions.

Figure 6.1 shows the procedure for changing the interrupt generate factor.

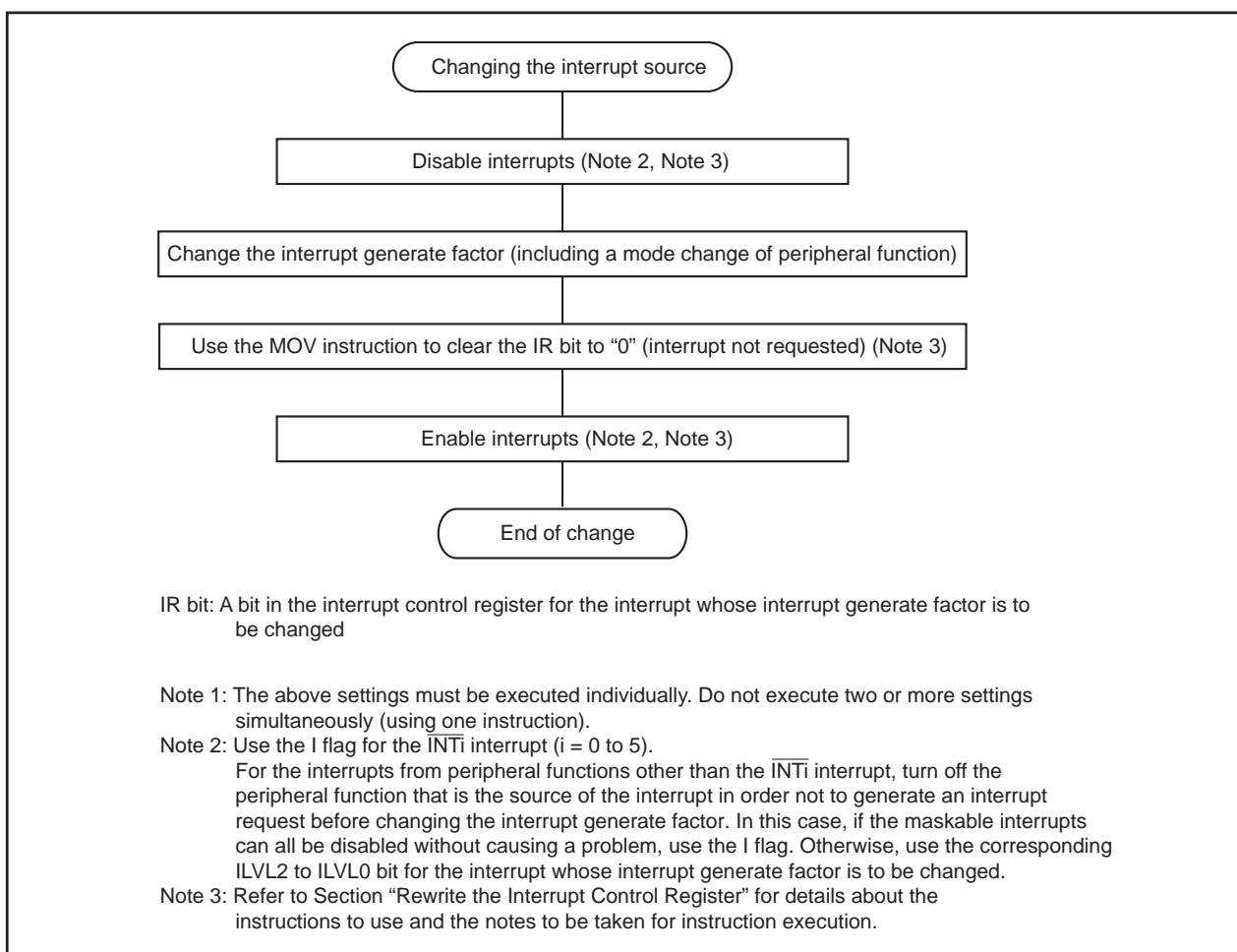


Figure 6.1. Procedure for Changing the Interrupt Generate Factor

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

1. Either an “L” level of at least $t_{w(\text{INL})}$ or an “H” level of at least $t_{w(\text{INH})}$ width is necessary for the signal input to pins INT0 through INT5 regardless of the CPU operation clock.
2. If the POL bit in the INT0IC to INT5IC registers or the IFSR7 to IFSR0 bits in the IFSR register are changed, the IR bit may inadvertently set to 1 (interrupt requested). Be sure to clear the IR bit to 0 (interrupt not requested) after changing any of those register bits.

Rewrite the Interrupt Control Register

- (1) The interrupt control register for any interrupt should be modified in places where no requests for that interrupt may occur. Otherwise, disable the interrupt before rewriting the interrupt control register.
- (2) To rewrite the interrupt control register for any interrupt after disabling that interrupt, be careful with the instruction to be used.

Changing any bit other than the IR bit

If while executing an instruction, a request for an interrupt controlled by the register being modified occurs, the IR bit in the register may not be set to "1" (interrupt requested), with the result that the interrupt request is ignored. If such a situation presents a problem, use the instructions shown below to modify the register.

Usable instructions: AND, OR, BCLR, BSET

Changing the IR bit

Depending on the instruction used, the IR bit may not always be cleared to "0" (interrupt not requested). Therefore, be sure to use the MOV instruction to clear the IR bit.

- (3) When using the I flag to disable an interrupt, refer to the sample program fragments shown below as you set the I flag. (Refer to (2) for details about rewrite the interrupt control registers in the sample program fragments.)

Examples 1 through 3 show how to prevent the I flag from being set to "1" (interrupts enabled) before the interrupt control register is rewritten, owing to the effects of the internal bus and the instruction queue buffer.

Example 1: Using the NOP instruction to keep the program waiting until the interrupt control register is modified

```
INT_SWITCH1:
  FCLR    I           ; Disable interrupts.
  AND.B   #00h, 0055h ; Set the TA0IC register to "0016".
  NOP
  NOP
  FSET    I           ; Enable interrupts.
```

The number of NOP instruction is as follows.

PM20=1(1 wait) : 2, PM20=0(2 wait) : 3, when using HOLD function : 4.

Example 2: Using the dummy read to keep the FSET instruction waiting

```
INT_SWITCH2:
  FCLR    I           ; Disable interrupts.
  AND.B   #00h, 0055h ; Set the TA0IC register to "0016".
  MOV.W   MEM, R0     ; Dummy read.
  FSET    I           ; Enable interrupts.
```

Example 3: Using the POPC instruction to changing the I flag

```
INT_SWITCH3:
  PUSHC   FLG
  FCLR    I           ; Disable interrupts.
  AND.B   #00h, 0055h ; Set the TA0IC register to "0016".
  POPC    FLG         ; Enable interrupts.
```

Watchdog Timer Interrupt

Initialize the watchdog timer after the watchdog timer interrupt occurs.

Precautions for DMAC**Write to DMAE Bit in DMiCON Register**

When both of the conditions below are met, follow the steps below.

Conditions

- The DMAE bit is set to "1" again while it remains set (DMAi is in an active state).
- A DMA request may occur simultaneously when the DMAE bit is being written.

Step 1: Write "1" to the DMAE bit and DMAS bit in DMiCON register simultaneously^(*1).

Step 2: Make sure that the DMAi is in an initial state^(*2) in a program.

If the DMAi is not in an initial state, the above steps should be repeated.

Notes:

*1. The DMAS bit remains unchanged even if "1" is written. However, if "0" is written to this bit, it is set to "0" (DMA not requested). In order to prevent the DMAS bit from being modified to "0", "1" should be written to the DMAS bit when "1" is written to the DMAE bit. In this way the state of the DMAS bit immediately before being written can be maintained.

Similarly, when writing to the DMAE bit with a read-modify-write instruction, "1" should be written to the DMAS bit in order to maintain a DMA request which is generated during execution.

*2. Read the TCRi register to verify whether the DMAi is in an initial state. If the read value is equal to a value which was written to the TCRi register before DMA transfer start, the DMAi is in an initial state. (If a DMA request occurs after writing to the DMAE bit, the value written to the TCRi register - 1.) If the read value is a value in the middle of transfer, the DMAi is not in an initial state.

Precautions for Timers

Timer A

(a) Timer A (Timer Mode)

1. The timer remains idle after reset. Set the mode, count source, counter value, etc. using the TAI_{MR} (i = 0 to 4) register and the TAI register before setting the TAI_S bit in the TABSR register to "1" (count starts).

Always make sure the TAI_{MR} register is modified while the TAI_S bit remains "0" (count stops) regardless whether after reset or not.

2. While counting is in progress, the counter value can be read out at any time by reading the TAI register. However, if the counter is read at the same time it is reloaded, the value "FFFF₁₆" is read. Also, if the counter is read before it starts counting after a value is set in the TAI register while not counting, the set value is read.

(b) Timer A (Event Counter Mode)

1. The timer remains idle after reset. Set the mode, count source, counter value, etc. using the TAI_{MR} (i = 0 to 4) register, the TAI register, the UDF register, the ONSF register TAZIE, TA0TGL and TA0TGH bits and the TRGSR register before setting the TAI_S bit in the TABSR register to "1" (count starts).

Always make sure the TAI_{MR} register, the UDF register, the ONSF register TAZIE, TA0TGL and TA0TGH bits and the TRGSR register are modified while the TAI_S bit remains "0" (count stops) regardless whether after reset or not.

2. While counting is in progress, the counter value can be read out at any time by reading the TAI register. However, "FFFF₁₆" can be read in underflow, while reloading, and "0000₁₆" in overflow. When setting TAI register to a value during a counter stop, the setting value can be read before a counter starts counting.

(c) Timer A (One-shot Timer Mode)

1. The timer remains idle after reset. Set the mode, count source, counter value, etc. using the TAI_{MR} (i = 0 to 4) register, the TAI register, the ONSF register TA0TGL and TA0TGH bits and the TRGSR register before setting the TAI_S bit in the TABSR register to "1" (count starts).

Always make sure the TAI_{MR} register, the ONSF register TA0TGL and TA0TGH bits and the TRGSR register are modified while the TAI_S bit remains "0" (count stops) regardless whether after reset or not.

2. When setting TAI_S bit to "0" (count stop), the followings occur:
 - A counter stops counting and a content of reload register is reloaded.
 - TAI_{OUT} pin outputs "L".
 - After one cycle of the CPU clock, the IR bit of TAI_{IC} register is set to "1" (interrupt request).

3. Output in one-shot timer mode synchronizes with a count source internally generated. When an external trigger has been selected, one-cycle delay of a count source as maximum occurs between a trigger input to TAIIN pin and output in one-shot timer mode.
4. The IR bit is set to "1" when timer operation mode is set with any of the following procedures:
 - Select one-shot timer mode after reset.
 - Change an operation mode from timer mode to one-shot timer mode.
 - Change an operation mode from event counter mode to one-shot timer mode.To use the timer Ai interrupt (the IR bit), set the IR bit to "0" after the changes listed above have been made.
5. When a trigger occurs, while counting, a counter reloads the reload register to continue counting after generating a re-trigger and counting down once. To generate a trigger while counting, generate a second trigger between occurring the previous trigger and operating longer than one cycle of a timer count source.

(d) Timer A (Pulse Width Modulation Mode)

1. The timer remains idle after reset. Set the mode, count source, counter value, etc. using the TAIiMR (i = 0 to 4) register, the TAIi register, the ONSF register TA0TGL and TA0TGH bits and the TRGSR register before setting the TAIiS bit in the TABSR register to "1" (count starts).
Always make sure the TAIiMR register, the ONSF register TA0TGL and TA0TGH bits and the TRGSR register are modified while the TAIiS bit remains "0" (count stops) regardless whether after reset or not.
2. The IR bit is set to "1" when setting a timer operation mode with any of the following procedures:
 - Select the PWM mode after reset.
 - Change an operation mode from timer mode to PWM mode.
 - Change an operation mode from event counter mode to PWM mode.To use the timer Ai interrupt (interrupt request bit), set the IR bit to "0" by program after the above listed changes have been made.
3. When setting TAIiS register to "0" (count stop) during PWM pulse output, the following action occurs:
 - Stop counting.
 - When TAIiOUT pin is output "H", output level is set to "L" and the IR bit is set to "1".
 - When TAIiOUT pin is output "L", both output level and the IR bit remains unchanged.

Timer B

(a) Timer B (Timer Mode)

1. The timer remains idle after reset. Set the mode, count source, counter value, etc. using the TBIiMR (i = 0 to 5) register and TBIi register before setting the TBIiS bit in the TABSR or the TBSR register to "1" (count starts).
Always make sure the TBIiMR register is modified while the TBIiS bit remains "0" (count stops) regardless whether after reset or not.

2. A value of a counter, while counting, can be read in TBi register at any time. "FFFF16" is read while reloading. Setting value is read between setting values in TBi register at count stop and starting a counter.

(b) Timer B (Event Counter Mode)

1. The timer remains idle after reset. Set the mode, count source, counter value, etc. using the TBiMR (i = 0 to 5) register and TBi register before setting the TBiS bit in the TABSR or the TBSR register to "1" (count starts).

Always make sure the TBiMR register is modified while the TBiS bit remains "0" (count stops) regardless whether after reset or not.

2. The counter value can be read out on-the-fly at any time by reading the TBi register. However, if this register is read at the same time the counter is reloaded, the read value is always "FFFF16." If the TBi register is read after setting a value in it while not counting but before the counter starts counting, the read value is the one that has been set in the register.

(c) Timer B (Pulse Period/pulse Width Measurement Mode)

1. The timer remains idle after reset. Set the mode, count source, etc. using the TBiMR (i = 0 to 5) register before setting the TBiS bit in the TABSR or the TBSR register to "1" (count starts).

Always make sure the TBiMR register is modified while the TBiS bit remains "0" (count stops) regardless whether after reset or not. To clear the MR3 bit to "0" by writing to the TBiMR register while the TBiS bit = "1" (count starts), be sure to write the same value as previously written to the TMOD0, TMOD1, MR0, MR1, TCK0 and TCK1 bits and a 0 to the MR2 bit.

2. The IR bit of TBiIC register (i=0 to 5) goes to "1" (interrupt request), when an effective edge of a measurement pulse is input or timer Bi is overflowed. The factor of interrupt request can be determined by use of the MR3 bit of TBiMR register within the interrupt routine.

3. If the source of interrupt cannot be identified by the MR3 bit such as when the measurement pulse input and a timer overflow occur at the same time, use another timer to count the number of times timer B has overflowed.

4. To set the MR3 bit to "0" (no overflow), set TBiMR register with setting the TBiS bit to "1" and counting the next count source after setting the MR3 bit to "1" (overflow).

5. Use the IR bit of TBiIC register to detect only overflows. Use the MR3 bit only to determine the interrupt factor within the interrupt routine.

6. When a count is started and the first effective edge is input, an indeterminate value is transferred to the reload register. At this time, timer Bi interrupt request is not generated.

7. A value of the counter is indeterminate at the beginning of a count. MR3 may be set to "1" and timer Bi interrupt request may be generated between a count start and an effective edge input.

8. For pulse width measurement, pulse widths are successively measured. Use program to check whether the measurement result is an "H" level width or an "L" level width.

Precautions for Serial I/O (Clock-synchronous Serial I/O)

Transmission/reception

With an external clock selected, and choosing the $\overline{\text{RTS}}$ function, the output level of the $\overline{\text{RTSi}}$ pin goes to “L” when the data-receivable status becomes ready, which informs the transmission side that the reception has become ready. The output level of the RTSi pin goes to “H” when reception starts. So if the $\overline{\text{RTSi}}$ pin is connected to the $\overline{\text{CTS}}$ pin on the transmission side, the circuit can transmission and reception data with consistent timing. With the internal clock, the $\overline{\text{RTS}}$ function has no effect.

Transmission

When an external clock is selected, the conditions must be met while if the UiC0 register's CKPOL bit = “0” (transmit data output at the falling edge and the receive data taken in at the rising edge of the transfer clock), the external clock is in the high state; if the UiC0 register's CKPOL bit = “1” (transmit data output at the rising edge and the receive data taken in at the falling edge of the transfer clock), the external clock is in the low state.

- The TE bit of UiC1 register= “1” (transmission enabled)
- The TI bit of UiC1 register = “0” (data present in UiTB register)
- If CTS function is selected, input on the CTSi pin = “L”

Reception

1. In operating the clock-synchronous serial I/O, operating a transmitter generates a shift clock. Fix settings for transmission even when using the device only for reception. Dummy data is output to the outside from the TxDi pin when receiving data.
2. When an internal clock is selected, set the UiC1 register (i = 0 to 2)'s TE bit to 1 (transmission enabled) and write dummy data to the UiTB register, and the shift clock will thereby be generated. When an external clock is selected, set the UiC1 register (i = 0 to 2)'s TE bit to 1 and write dummy data to the UiTB register, and the shift clock will be generated when the external clock is fed to the CLKi input pin.
3. When successively receiving data, if all bits of the next receive data are prepared in the UARTi receive register while the UiC1 register (i = 0 to 2)'s RE bit = “1” (data present in the UiRB register), an overrun error occurs and the UiRB register OER bit is set to “1” (overrun error occurred). In this case, because the content of the UiRB register is indeterminate, a corrective measure must be taken by programs on the transmit and receive sides so that the valid data before the overrun error occurred will be retransmitted. Note that when an overrun error occurred, the SiRIC register IR bit does not change state.
4. To receive data in succession, set dummy data in the lower-order byte of the UiTB register every time reception is made.
5. When an external clock is selected, the conditions must be met while if the CKPOL bit = “0”, the external clock is in the high state; if the CKPOL bit = “1”, the external clock is in the low state.
 - The RE bit of UiC1 register= “1” (reception enabled)
 - The TE bit of UiC1 register= “1” (transmission enabled)
 - The TI bit of UiC1 register= “0” (data present in the UiTB register)

Precautions for Serial I/O (UART Mode)

Special Mode 4 (SIM Mode)

A transmit interrupt request is generated by setting the U2C1 register U2IRS bit to “1” (transmission complete) and U2ERE bit to “1” (error signal output) after reset. Therefore, when using SIM mode, be sure to clear the IR bit to “0” (no interrupt request) after setting these bits.

Precautions for A-D Converter

1. Set ADCON0 (except bit 6), ADCON1 and ADCON2 registers when A-D conversion is stopped (before a trigger occurs).
2. When the VCUT bit of ADCON1 register is changed from “0” (Vref not connected) to “1” (Vref connected), start A-D conversion after passing 1 μ s or longer.
3. To prevent noise-induced device malfunction or latchup, as well as to reduce conversion errors, insert capacitors between the AVCC, VREF, and analog input pins (AN_i(i=0 to 7)) each and the AVSS pin. Similarly, insert a capacitor between the VCC pin and the VSS pin. Figure 6.2 is an example connection of each pin.
4. Make sure the port direction bits for those pins that are used as analog inputs are set to “0” (input mode). Also, if the ADCON0 register's TGR bit = 1 (external trigger), make sure the port direction bit for the ADTRG pin is set to “0” (input mode).
5. When using key input interrupts, do not use any of the four AN4 to AN7 pins as analog inputs. (A key input interrupt request is generated when the A-D input voltage goes low.)
6. The ϕ_{AD} frequency must be 10 MHz or less. Without sample-and-hold function, limit the ϕ_{AD} frequency to 250kHz or more. With the sample and hold function, limit the ϕ_{AD} frequency to 1MHz or more.
7. When changing an A-D operation mode, select analog input pin again in the CH2 to CH0 bits of ADCON0 register and the SCAN1 to SCAN0 bits of ADCON1 register.

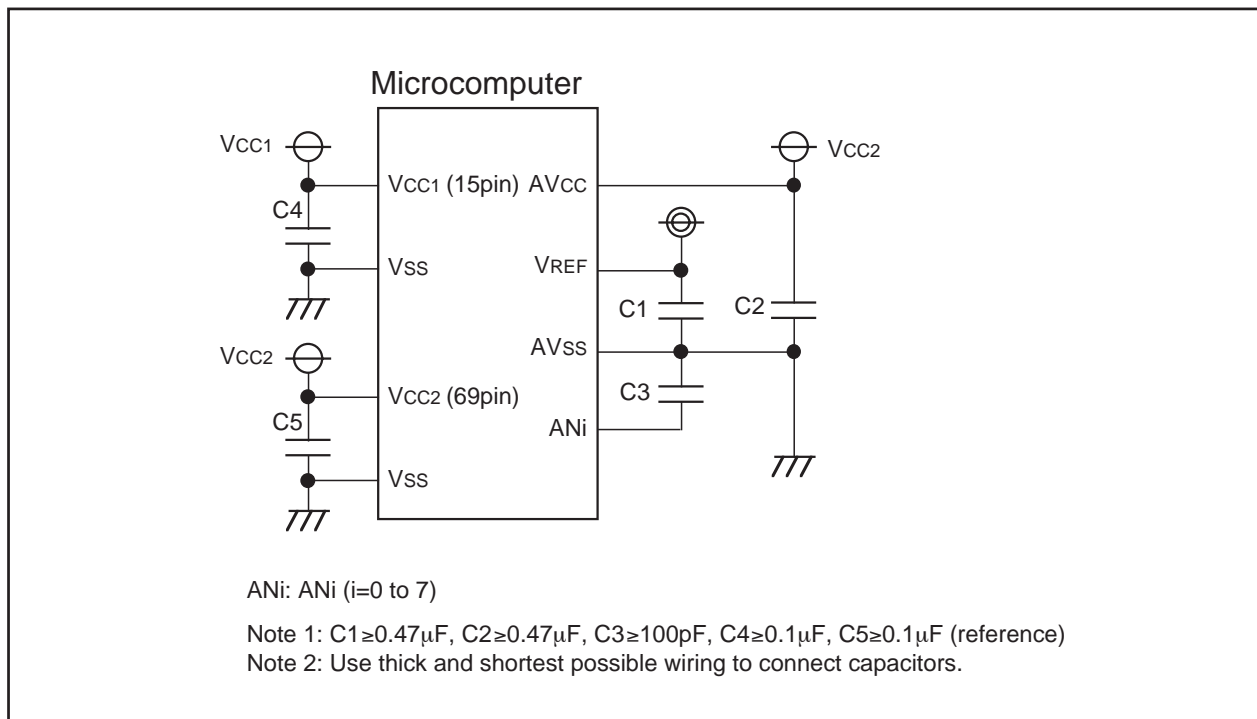


Figure 6.2. Use of capacitors to reduce noise

8. If the CPU reads the ADi register (i = 0 to 7) at the same time the conversion result is stored in the ADi register after completion of A-D conversion, an incorrect value may be stored in the ADi register. This problem occurs when a divide-by-n clock derived from the main clock or a subclock is selected for CPU clock.
 - When operating in one-shot or single-sweep mode

Check to see that A-D conversion is completed before reading the target ADi register. (Check the ADIC register's IR bit to see if A-D conversion is completed.)
 - When operating in repeat mode or repeat sweep mode 0 or 1

Use the main clock for CPU clock directly without dividing it.

9. If A-D conversion is forcibly terminated while in progress by setting the ADCON0 register's ADST bit to "0" (A-D conversion halted), the conversion result of the A-D converter is indeterminate. The contents of ADi registers irrelevant to A-D conversion may also become indeterminate. If while A-D conversion is underway the ADST bit is cleared to "0" in a program, ignore the values of all ADi registers.

Precautions for Programmable I/O Ports

1. Setting the SM32 bit in the S3C register to “1” causes the P92 pin to go to a high-impedance state. Similarly, setting the SM42 bit in the S4C register to “1” causes the P96 pin to go to a high-impedance state.
2. The input threshold voltage of pins differs between programmable input/output ports and peripheral functions.
Therefore, if any pin is shared by a programmable input/output port and a peripheral function and the input level at this pin is outside the range of recommended operating conditions V_{IH} and V_{IL} (neither “high” nor “low”), the input level may be determined differently depending on which side—the programmable input/output port or the peripheral function—is currently selected.

Electric Characteristic Differences Between Mask ROM and Flash Memory Version Microcomputers

Flash memory version and mask ROM version may have different characteristics, operating margin, noise tolerated dose, noise width dose in electrical characteristics due to internal ROM, different layout pattern, etc. When switching to the mask ROM version, conduct equivalent tests as system evaluation tests conducted in the flash memory version.

Precautions for Flash Memory Version

Precautions for Functions to Inhibit Rewriting Flash Memory Rewrite

ID codes are stored in addresses 0FFFDF₁₆, 0FFFE3₁₆, 0FFFEB₁₆, 0FFFEF₁₆, 0FFFF3₁₆, 0FFFF7₁₆, and 0FFFFB₁₆. If wrong data are written to these addresses, the flash memory cannot be read or written in standard serial I/O mode.

The ROMCP register is mapped in address 0FFFFFF₁₆. If wrong data is written to this address, the flash memory cannot be read or written in parallel I/O mode.

In the flash memory version of microcomputer, these addresses are allocated to the vector addresses (H) of fixed vectors.

Precautions for Stop mode

When shifting to stop mode, the following settings are required:

- Set the FMR01 bit to “0” (CPU rewrite mode disabled) and disable DMA transfers before setting the CM10 bit to “1” (stop mode).
- Execute the JMP.B instruction subsequent to the instruction which sets the CM10 bit to “1” (stop mode)

```
Example program  BSET      0, CM1      ; Stop mode
                  JMP.B     L1
```

L1:

Program after returning from stop mode

Precautions for Wait mode

When shifting to wait mode, set the FMR01 bit to "0" (CPU rewrite mode disabled) before executing the WAIT instruction.

Precautions for Low power dissipation mode

If the CM05 bit is set to "1" (main clock stop), the following commands must not be executed.

- Program
- Block erase
- Lock bit program

Writing command and data

Write the command code and data at even addresses.

Precautions for Program Command

Write 'xx4016' in the first bus cycle and write data to the write address in the second bus cycle, and an auto program operation (data program and verify) will start. Make sure the address value specified in the first bus cycle is the same even address as the write address specified in the second bus cycle.

Precautions for Lock Bit Program Command

Write 'xx7716' in the first bus cycle and write 'xxD016' to the uppermost address of a block (even address, however) in the second bus cycle, and the lock bit for the specified block is cleared to "0". Make sure the address value specified in the first bus cycle is the same uppermost block address that is specified in the second bus cycle.

Operation speed

Before entering CPU rewrite mode (EW0 or EW1 mode), select 10 MHz or less for CPU clock using the CM0 register's CM06 bit and CM1 register's CM17–6 bits. Also, set the PM1 register's PM17 bit to 1 (with wait state).

Instructions inhibited against use

The following instructions cannot be used in EW0 mode because the flash memory's internal data is referenced: UND instruction, INTO instruction, JMPS instruction, JSRS instruction, and BRK instruction

Interrupts

EW0 Mode

- Any interrupt which has a vector in the variable vector table can be used providing that its vector is transferred into the RAM area.
- The $\overline{\text{NMI}}$ and watchdog timer interrupts can be used because the FMR0 register and FMR1 register are initialized when one of those interrupts occurs. The jump addresses for those interrupt service routines should be set in the fixed vector table.

Because the rewrite operation is halted when a $\overline{\text{NMI}}$ or watchdog timer interrupt occurs, the rewrite program must be executed again after exiting the interrupt service routine.

- The address match interrupt cannot be used because the flash memory's internal data is referenced.

EW1 Mode

- Make sure that any interrupt which has a vector in the variable vector table or address match interrupt will not be accepted during the auto program or auto erase period.
- Avoid using watchdog timer interrupts.
- The $\overline{\text{NMI}}$ interrupt can be used because the FMR0 register and FMR1 register are initialized when this interrupt occurs. The jump address for the interrupt service routine should be set in the fixed vector table.

Because the rewrite operation is halted when a $\overline{\text{NMI}}$ interrupt occurs, the rewrite program must be executed again after exiting the interrupt service routine.

How to access

To set the FMR01, FMR02, or FMR11 bit to "1", write "0" and then "1" in succession. This is necessary to ensure that no interrupts or DMA transfers will occur before writing "1" after writing "0". Also only when $\overline{\text{NMI}}$ pin is "H" level.

Writing in the user ROM area

EW0 Mode

- If the power supply voltage drops while rewriting any block in which the rewrite control program is stored, a problem may occur that the rewrite control program is not correctly rewritten and, consequently, the flash memory becomes unable to be rewritten thereafter. In this case, standard serial I/O or parallel I/O mode should be used.

EW1 Mode

- Avoid rewriting any block in which the rewrite control program is stored.

DMA transfer

In EW1 mode, make sure that no DMA transfers will occur while the FMR0 register's FMR00 bit = 0 (during the auto program or auto erase period).

Regarding Programming/Erase Times and Execution Time

As the number of programming/erase times increases, so does the execution time for software commands (Program, Block Erase, and Lock Bit Program). Especially when the number of programming/erase times exceeds 1,000, the software command execution time is noticeably extended. Therefore, the software command wait time that is set must be greater than the maximum rated value of electrical characteristics.

The software commands are aborted by hardware reset 1, hardware reset 2, $\overline{\text{NMI}}$ interrupt, and watchdog timer interrupt. If a software command is aborted by such reset or interrupt, the block that was in process must be erased before reexecuting the aborted command.

Other Notes

When the power is being turned on or off

Start VCC1, VCC2, VDD2, VDD3 and AVCC simultaneously.

While this device is operating, set these pins to the same electric potential.

Also, turn off VCC1, VCC2, VDD2, VDD3, and AVCC simultaneously when the power supply is being turned off.

When using $VCC1 < VCC2$, ensure voltage of VCC1 will not exceed voltage of VCC2 while the power is being turned on or off.

Execute in the following procedure when VCC1 is turned off (VCC2 voltage is supplied).

Procedure of Vcc1 Off (Note 1)

- ① Disable an interrupt which uses pins related to VCC1.
- ② Stop peripheral functions related to VCC1 (Note 2).
- ③ Set pins related to VCC1 to input mode (Note 3).
- ④ A low-level signal "L" is applied to the TEST3 pin (115 pins) from a high-level signal "H".
- ⑤ Turn off VCC1.

Note 1: Refer to the following "Additions" for details of procedures ① to ⑤.

Note 2: Only when the input from pins related to VCC1 is used. Refer to the following "Additions" for details.

Note 3: If the amount of power consumption is not a problem for a system when the above procedure ④ is executed, it is also possible to execute this procedure after the procedure ④.

Procedure of Vcc1 ON

- ① Turn on VCC1.
- ② VCCOFF pin (91-pin) is switched from "H" to "L".
- ③ Set pins VCC1, Peripheral function and Interrupt.

<Additions>

① Disable an affected interrupt by pins related to VCC1.

Disable an affected interrupt by pins related to VCC1 by setting the interrupt priority level selection and the interrupt request bits in the the following interrupt control register to "0". The interrupt that pins as to VCC1 influences is prohibited by setting the interrupt priority level selection bit and the interrupt request bit of the following interrupt control register to "0".

In the transitional state when changing the power supply voltage including being turned on or off, ensure each voltage of VCC1, VDD2, and VDD3 will not exceed voltage of VCC2.

TA0IC to TA4IC (timer A interrupt control register)

INT0 to INT2IC (external interrupt control register)

S0RIC to S2RIC (UART receive interrupt control register)

Even if other interrupts are disabled without any problem in software, clear the I flag and it is also possible to execute the above interrupt disable process after the procedure ④.

② Stop peripheral functions related to VCC1

Stop the function when pins related to VCC1 input affect.

When pins related to VCC1 input affect as follows:

- When operating in timer A (TA0 to TA4) and the event count mode
- When the gate input function is used in the event count mode, the one-shot timer, and PWM mode. (When the MR2 bit in the timer A mode registers TA0MR to TA4MR are set to "1")
- When UART to UART2 reception are set

Set the following in these cases.

- Timer A
 - Set the timer count start flags of timers A0 to A4 (TA0S to TA4S bits in the TABSR register) to "0".
- UART reception
 - Set the RE and TE bits in the U0C1 to U2C1 registers to "0."

Precautions when sub clock starts

When a low-level signal "L" is applied to the START pin and a reset is deasserted, a sub clock divided-by-8 becomes a CPU clock.

When using in this condition, set the CM07 bit in the CM0 register to "1" and switch the CPU clock to sub clock (no division).

Power supply noise and latch-up

In order to avoid power supply noise and latch-up, connect a bypass capacitor (more than 0.1 μ F) directly between the VCC pin and VSS pin, VDD2 pin and VSS2 pin, VDD3 pin and VSS3 pin, AVCC pin and AVSS pin using a heavy wire.

Please note that neither the over shot nor the shot under are generated in the pulse shape of pin (The voltage that exceeds the absolute maximum rating is not impressed) for the device characteristic deterioration prevention that accompanies the microcomputer malfunction and the latch-up to pin by the outpatient noise element.

And, connect VSS (GND) to the TEST1 pin (35 pin) via the capacitor (more than 0.1 μ F).

When oscillation circuit stop for data slicer

Expansion register XTAL_VCO, PDC_VCO_ON, VPS_VCO_ON is set at "L", when the data slicer is not used, and the oscillation is stopped. When starting oscillation again, set data at the following order.

- (a) Set expansion register XTAL_VCO = "H."
- (b) Set expansion register PDC_VCO_ON, VPS_VCO_ON = "H."
- (c) 60 ms or more is a waiting state (stability period of internal oscillation circuit + data slice preparation).

* To operate slice RAM, set expansion register XTAL_VCO = "H."

Access the memories after waiting for 20 ms certainly when resuming synchronous oscillation from the off state.

When operation start from stand-by mode (clock is stopped)

Set up an extended register as follows in standby mode.

- (a) Set extended register XTAL_VCO, PDC_VCO_ON, and VPS_VCO_ON as "L."

When you return to an oscillation state from a clock oscillation stop, set up as the notes of the oscillation circuit stop for data slicers.

Notes on operating with a low supply voltage (Vcc = 2.0 V to 5.5 V, f(XcIN) = 32 kHz)

When in single-chip mode, this product can operate with a low supply voltage only during low power dissipation mode. Before operating with a low supply voltage, always be sure to set the relevant register bits to select low power dissipation mode (BCLK : f(XcIN), main clock XIN : stop, subclock XCIN : oscillating). Then reduce the power supply voltage Vcc to 3.0 V.

Also, when returning to normal operation, raise the power supply voltage to 5.0V while in low power consumption mode before entering normal operation mode.

When moving from any operation mode to another, make sure a state transition occurs according to the state transition diagram (Figure 2.5.9) in Section 2.5.3, "Power control."

The status of the power supply voltage Vcc during operation mode transition is shown in Figure 6.3 below.

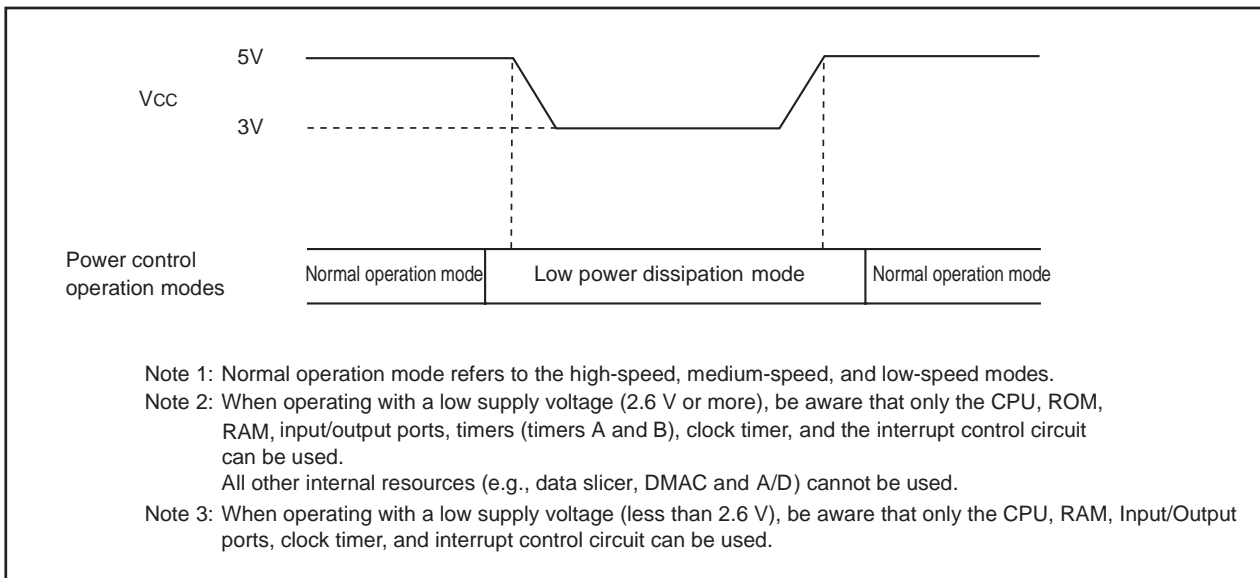


Figure 6.3 Status of the power supply voltage Vcc during operation mode transition

Serial I/O (RxDi input setup time)

For the RxDi input setup time, refer to the rated values shown below, as well as Electrical Characteristics Table 3.23, "Serial I/O."

Table6.1. Serial I/O (Vcc=5V)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t _{su(D-C)}	RxDi input setup time	70		ns

Note: Refer to "Table 3.23. Serial I/O of the Electrical Characteristics."

Precautions for LP2, LP3 and LP4 pins

Connect capacitors to LP2, LP3 and LP4 as shown in Figure 6.4.

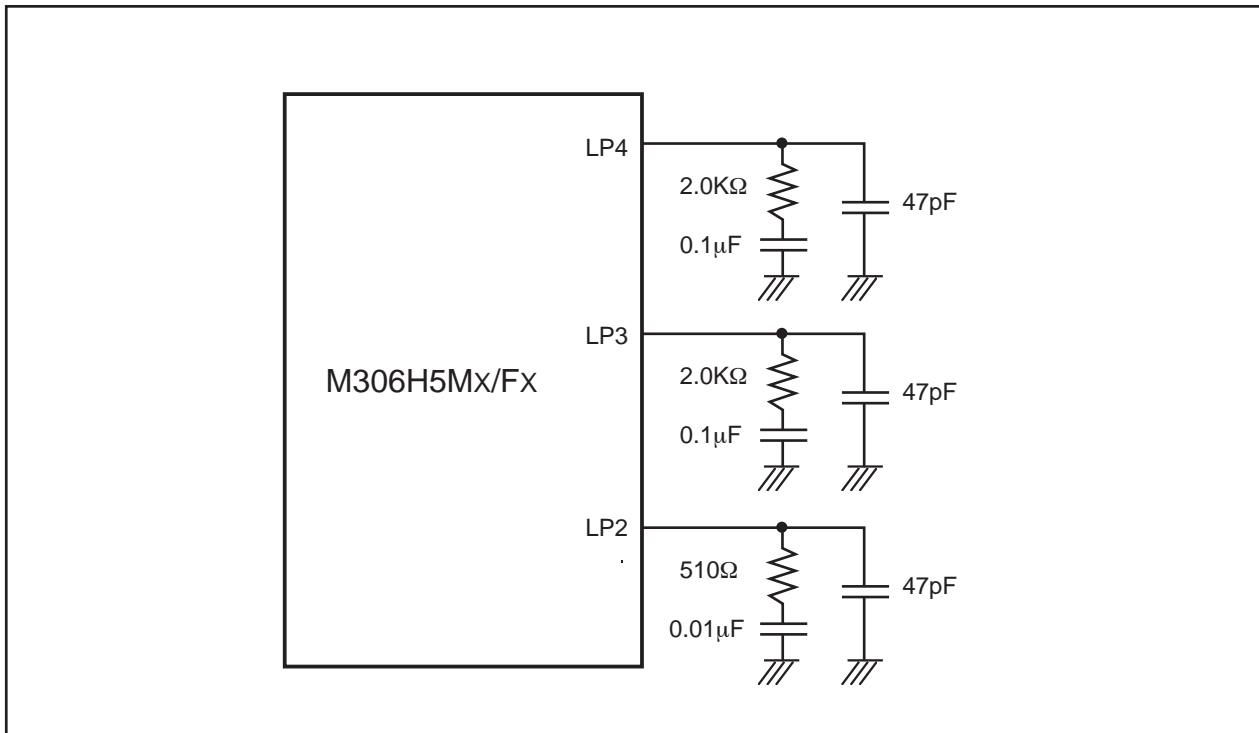


Figure 6.4 Use of capacitors to reduce noise

7. Differences Between M306H5 and M306H3

Differences Between M306H5 and M306H3: Pin connect (Note 1)

Item	M306H5	M306H3
Minimum instruction execution time	62.5 nsec (f(XIN) = 16 MHz)	100 nsec (f(XIN) = 10 MHz)
Power supply voltage	Vcc1 = 3.0 V to Vcc2, Vcc2 = 4.5 V to 5.5V (at f(XIN) = 16 MHz) Vcc1 = 3.0 V to Vcc2, Vcc2 = 4.0 V to 5.5V (at f(XIN) = 16 MHz, except for A-D converter and data slicer.) Vcc1 = 2.9 V to Vcc2, Vcc2 = 2.9 V to 5.5V (at f(XIN) = 16 MHz, at divide-by-8) Vcc1 = 2.0 V to Vcc2, Vcc2 = 2.0 V to 5.5V (at f(XCIN) = 32 kHz, during low power dissipation mode)	Vcc = 4.75 V to 5.25 V (at f(XIN) = 10 MHz) Vcc = 2.6 V to 5.25 V (at f(XCIN) = 32 kHz)
15-pin, 69-pin	Vcc1 pin (15-pin), Vcc2 pin (69-pin) It is possible to connect a different power supply to Vcc1 and Vcc2 (Vcc1 ≤ Vcc2).	Vcc pin (15-pin, 69-pin) • 15-pin and 69-pin are connected at the same potential level.
115-pin	TEST3 • Vcc1 power supply input switching pin	FSCIN • fsc input pin for synchronous signal generation
Pin power supply	Vcc1 pin : P6, P7, P8 ₀ to P8 ₄ Vcc2 pin : P0 to P5, P8 ₅ to P8 ₇ , P9, P10 VDD2 pin : P11 Input from Vcc1 pin is controllable by pin level	Power supply voltage (Vcc) of Port (P0 to P10, P11) are common.
User ROM blocks	9 blocks : 4 Kbytes X 2, 8 Kbytes X 3, 32 Kbytes X 1, 64 Kbytes X 3	7 blocks : 4 Kbytes X 2, 8 Kbytes X 3, 32 Kbytes X 1, 64 Kbytes X 1
Remote control header detection function	Enable to set a rising period, falling period, and permissible period. (Enable to set L-period and H-period of permissible period respectively.)	Enable to set a rising period, falling period, and permissible period. (Permissible period is common to L-period and H-period.)
Remote control input filter function	Have (Noise Cancel width : approx. 2μ sec (Max.))	None
CRC calculation circuit for EPG-J	Generator polynomial is fixed	Generator polynomial is variable (register)
Ghost correction circuit	Built-in	None
Flash Memory Version Software command	7 commands • Read array • Read status register • Clear status register • Program • Block erase • Lock bit program • Read lock bit status	8 commands • Read array • Read status register • Clear status register • Program • Block erase • Erase all unlocked block • Lock bit program • Read lock bit status
Synchronous signal slice potential generation circuit	Built-in	None
Clock timer	Have	None
Slice beginning condition	As for the slice beginning condition, either after slice check beginning period passes or after standing up of clock run-in after the slice check beginning period passes is possible.	After slice check beginning period passes
Synchronous signal input	SYNIN input	SYNIN input or external H-V input

Note 1 : For details, refer to Datasheet

REVISION HISTORY

M306H5MG-XXXFP/MC-XXXFP/FGFP

Rev.	Date	Description	
		Page	Summary
1.00	Jan 19, 2005	–	First edition issued
1.20	Dec 13, 2005	p.319 p.320 p.323	"Procedure of VCC1 ON" is added. "Power supply noise and latch-up" is changed. "Differences Between M306H5 and M306H3" is changed.