Description

The M16C/6NT group of single-chip microcomputers are built using the high-performance silicon gate CMOS process using a M16C/60 Series CPU core and are packaged in a 100-pin plastic molded QFP. These single-chip microcomputers operate using sophisticated instructions featuring a high level of instruction efficiency. With 1M bytes of address space, they are capable of executing instructions at high speed. They also feature a built-in multiplier and DMAC, making them ideal for controlling office, communications, industrial equipment, and other high-speed processing applications.

Being equipped with two CAN (Controller Area Network) modules, the microcomputer is suited to drive automotive and industrial control systems. The CAN modules comply with the 2.0B specification.

Features

• Memory capacity	ROM 128K/256K bytes RAM 5K/10K bytes
• Shortest instruction execution time	62.5 ns (f(X_{IN}) = 16MHz, $^1/_1$ prescaler, without software wait) 100 ns (f(X_{IN}) = 20MHz, $^1/_2$ prescaler, without software wait)
• Supply voltage	4.2 to 5.5V (f(X_{IN}) = 16MHz, $^1/_1$ prescaler, without software wait) 4.2 to 5.5V (f(X_{IN}) = 20MHz, $^1/_2$ prescaler, without software wait)
• Low power consumption	TBD (f(X_{IN}) = 16MHz, $^1/_1$ prescaler, without software wait) TBD (f(X_{IN}) = 20MHz, $^1/_2$ prescaler, without software wait)
• Interrupts	29 internal and 9 external interrupt sources, 4 software interrupt sources; 7 priority levels (including key input interrupt)
• Multifunction 16-bit timer	5 output timers + 6 input timers
• Serial I/O	4 channels (3 for UART or clock synchronous, 1 for clock synchronous)
• DMAC	2 channels (trigger: 23 sources)
• CAN module	2 channels, 2.0B active
• A-D converter	10 bits X 26 analog inputs
• D-A converter	8 bits X 2 analog outputs
• CRC calculation circuit	1 circuit
• Watchdog timer	1 15-bit timer
• Programmable I/O	87 lines
• Input port	1 line ($P8_5$ shared with $\overline{NMI}$ pin)
• Chip select output	4 lines
• Memory expansion	Available (to a maximum of 1M bytes)
• Clock generating circuit	2 built-in clock generation circuits (built-in feedback resistor, and external ceramic or quartz oscillator)

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

Applications

Automotive and industrial control systems

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

Pin Configuration

Figure 1-1 shows the pin configuration (top view).

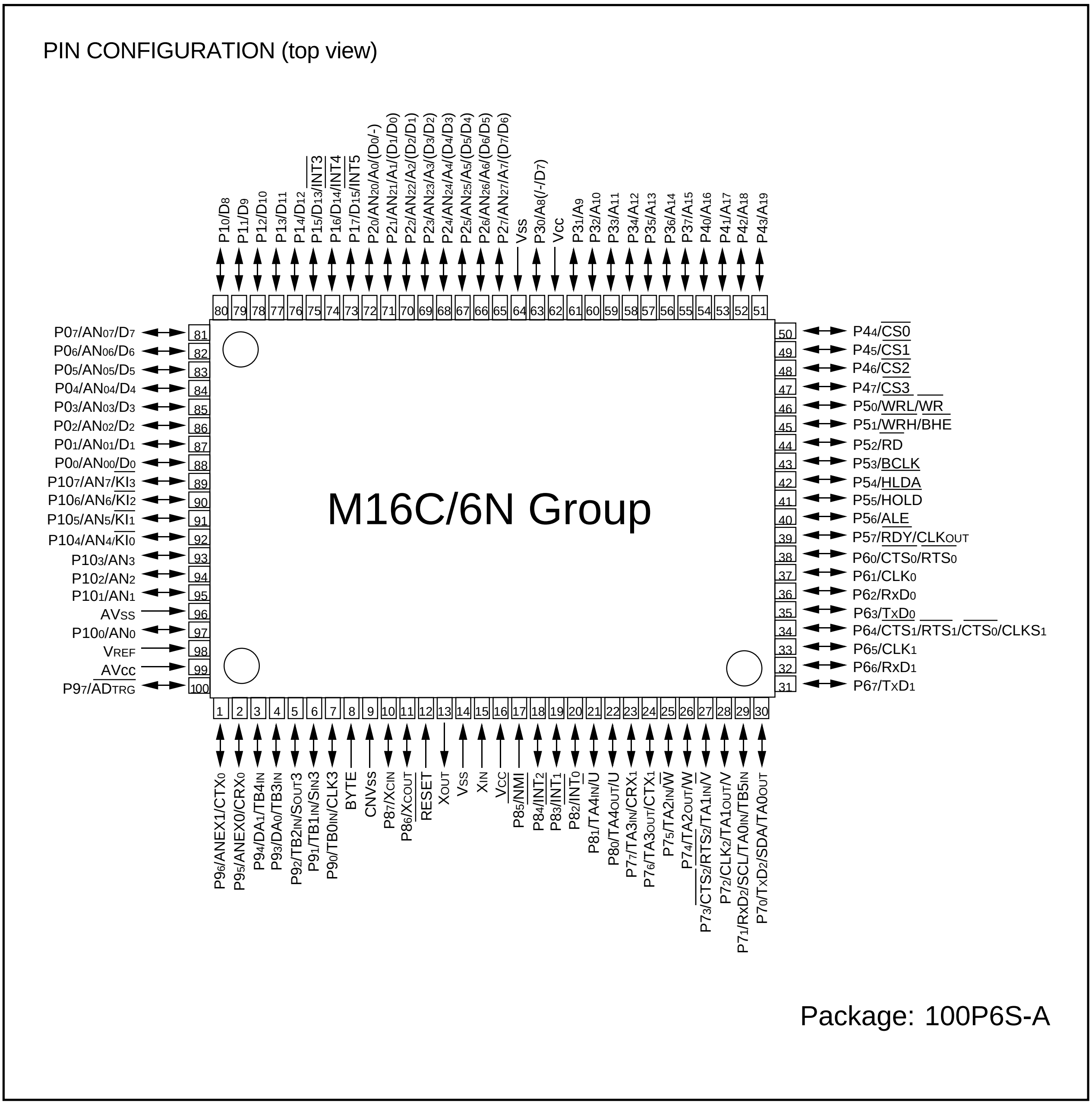
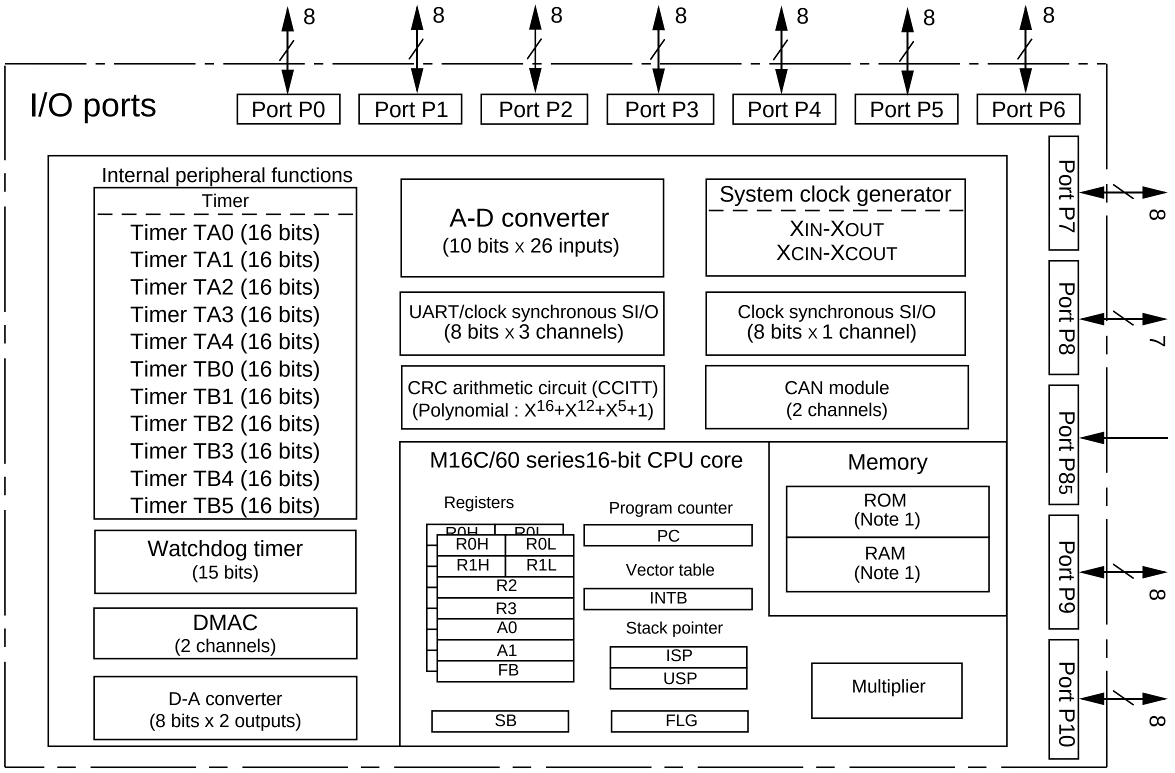


Figure 1-1. Pin configuration (top view)

Block Diagram

Figure 1-2 is a block diagram of the M16C/6N group.

Block diagram of the M16C/6N group



Note 1: Memory sizes depend on MCU type.

Figure 1-2. Block diagram of M16C/6N group

Performance Outline

Table 1-1 is a performance outline of the M16C/6N group.

Table 1-1. Performance outline of M16C/6N group

Item		Performance
Number of basic instructions		91 instructions
Shortest instruction execution time		62.5 ns (f(X_{IN}) = 16MHz, $^1/_1$ prescaler, without software wait) 100ns (f(X_{IN}) = 20MHz, $^1/_2$ prescaler, without software wait)
Memory capacity	ROM	128K to 256K byte
	RAM	5K to 10K byte
I/O ports	P0 to P10 (except P8$_5$)	8 bit x 10, 7 bit x 1
Input port	P8$_5$	1 bit x 1
Multifunction timer	TA0, TA1, TA2, TA3, TA4	16 bit x 5
	TB0, TB1, TB2, TB3, TB4, TB5	16 bit x 6
Serial I/O	UART0, UART1, UART2	(UART or clock synchronous) x 3
	SI/O3	Clock synchronous
A-D converter		10 bits x (8 + 8 + 8 + 2) inputs
D-A converter		8 bits x 2 channels
CRC calculation circuit		CRC-CCITT
DMAC		2 channels (trigger: 23 sources)
CAN module		2 channels, 2.0B active
Watchdog timer		15 bits x 1 (with prescaler)
Interrupt		29 internal and 9 external sources, 4 software sources, 7 priority levels
Clock generating circuit		2 built-in clock generation circuits (built-in feedback resistor, and external ceramic or quartz oscillator)
Supply voltage		4.2 to 5.5V (f(X_{IN}) = 16MHz, $^1/_1$ prescaler, without software wait) 4.2 to 5.5V (f(X_{IN}) = 20MHz, $^1/_2$ prescaler, without software wait)
Power consumption		TBD (f(X_{IN}) = 16MHz, $^1/_1$ prescaler, without software wait) TBD (f(X_{IN}) = 20MHz, $^1/_2$ prescaler, without software wait)
I/O characteristics	I/O withstand voltage	5V
	Output current	5mA
Operating ambient temperature		–40 to 85°C
Device configuration		CMOS high performance silicon gate
Package		100-pin plastic mold QFP

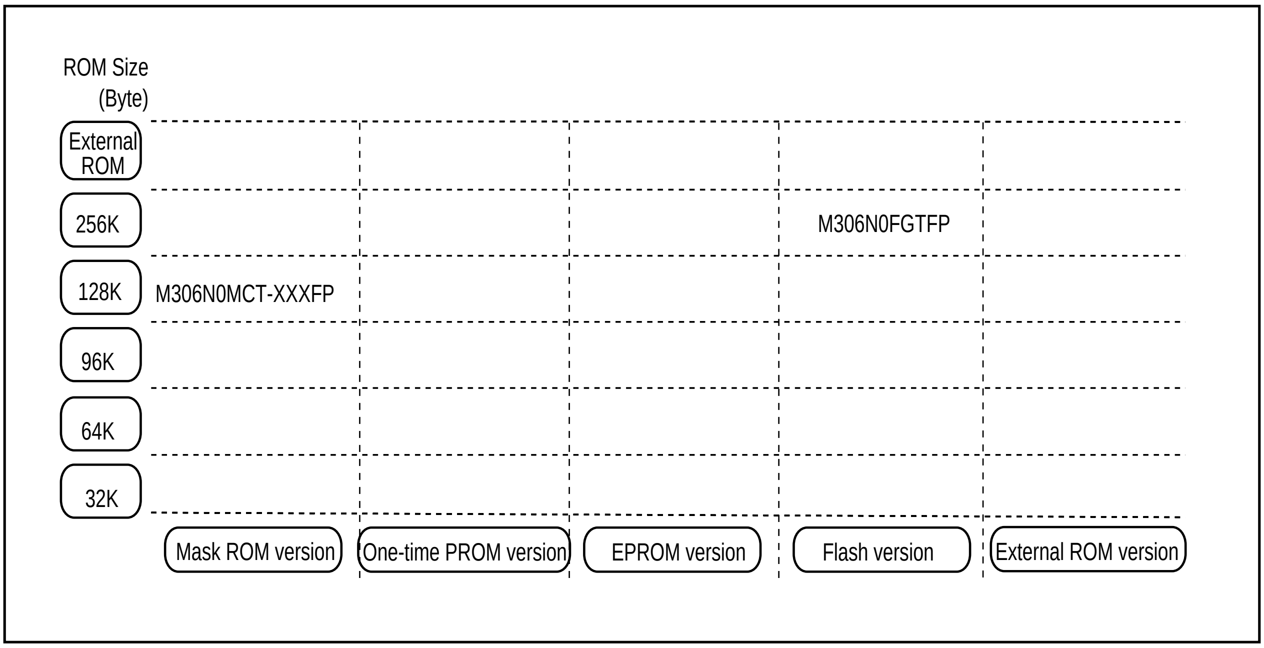
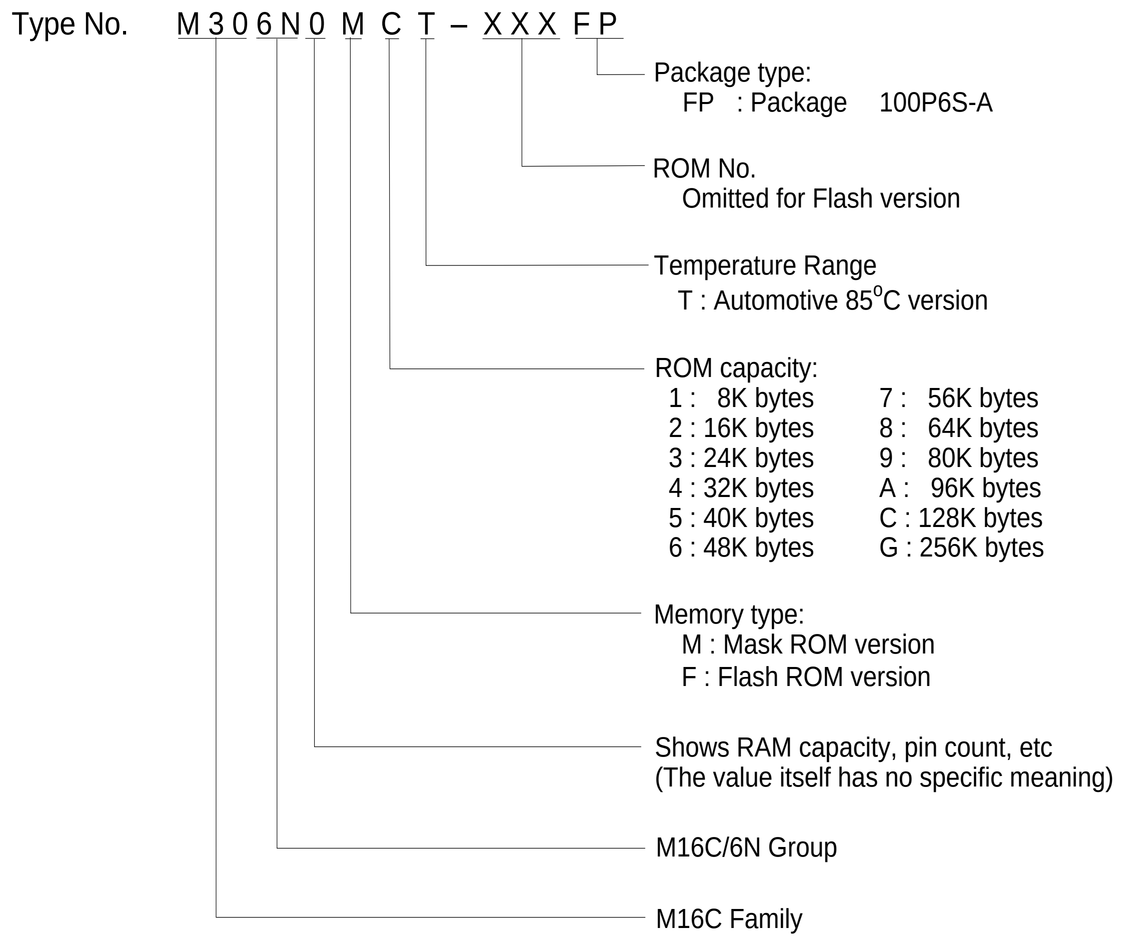


Figure 1-3. ROM expansion

Table 1-2. M16C/6N group

Apr. 1998

Type No	ROM size	RAM size	Package type	Remarks
M306N0MCT-XXXFP	128K byte	5K byte	100P6S-A	Mask ROM version
M306N0FGTFP	256K byte	10K byte	100P6S-A	Flash 5V version



Type No. M 3 0 6 N 0 M C T – X X X F P

Package type:
FP : Package 100P6S-A

ROM No.
Omitted for Flash version

Temperature Range
T : Automotive 85°C version

ROM capacity:

1 : 8K bytes	7 : 56K bytes
2 : 16K bytes	8 : 64K bytes
3 : 24K bytes	9 : 80K bytes
4 : 32K bytes	A : 96K bytes
5 : 40K bytes	C : 128K bytes
6 : 48K bytes	G : 256K bytes

Memory type:
M : Mask ROM version
F : Flash ROM version

Shows RAM capacity, pin count, etc
(The value itself has no specific meaning)

M16C/6N Group

M16C Family

Figure 1-4. Type No., memory size, and package

Table 1-3. Pin Description of M16C/6N group (1)

Pin name	Signal name	I/O type	Function
V_{CC}, V_{SS}	Power supply input		Supply 4.0 to 5.5 V to the V_{CC} pin. Supply 0 V to the V_{SS} pin.
CNV_{SS}	CNV_{SS}	Input	This pin switches between processor modes. Connect it to the V_{SS} pin to operate in single-chip or memory expansion mode. Connect it to the V_{CC} pin to operate in microprocessor mode.
$\overline{RESET}$	Reset input	Input	A “L” on this input resets the microcomputer.
X_{IN} X_{OUT}	Clock input Clock output	Input Output	These pins are provided for the main clock generating circuit.Connect a ceramic resonator or crystal between the X_{IN} and the X_{OUT} pins. To use an externally derived clock, input it to the X_{IN} pin and leave the X_{OUT} pin open.
BYTE	External data bus width select input	Input	This pin selects the width of an external data bus. A 16-bit width is selected when this input is “L”; an 8-bit width is selected when this input is “H”. This input must be fixed to either “H” or “L”. When operating in single-chip mode,connect this pin to V_{SS}.
AV_{CC}	Analog power supply input		This pin is a power supply input for the A-D converter. Connect this pin to V_{CC}.
AV_{SS}	Analog power supply input		This pin is a power supply input for the A-D converter. Connect this pin to V_{SS}.
V_{REF}	Reference voltage input	Input	This pin is a reference voltage input for the A-D converter.
$P0_0$ to $P0_7$	I/O port P0	Input/output	This is an 8-bit CMOS I/O port. It has an input/output port direction register that allows the user to set each pin for input or output individually. When set for input, the user can specify in units of four bits via software whether or not they are tied to a pull-up resistor. Pins in this port also function as A-D converter input pins.
D_0 to D_7		Input/output	When set as a separate bus, these pins input and output data (D_0–D_7).
$P1_0$ to $P1_7$	I/O port P1	Input/output	This is an 8-bit I/O port equivalent to P0. Pins in this port also function as external interrupt pins as selected by software.
D_8 to D_{15}		Input/output	When set as a separate bus, these pins input and output data (D_8–D_{15}).
$P2_0$ to $P2_7$	I/O port P2	Input/output	This is an 8-bit I/O port equivalent to P0. Pins in this port also function as A-D converter input pins.
A_0 to A_7		Output	These pins output 8 low-order address bits (A_0–A_7).
A_0/D_0 to A_7/D_7		Input/output	If the external bus is set as an 8-bit wide multiplexed bus, these pins input and output data (D_0–D_7) and output 8 low-order address bits (A_0–A_7) separated in time by multiplexing.
A_0, A_1/D_0 to A_7/D_6		Output Input/output	If the external bus is set as a 16-bit wide multiplexed bus, these pins input and output data (D_0–D_6) and output address (A_1–A_7) separated in time by multiplexing. They also output address (A_0).
$P3_0$ to $P3_7$	I/O port P3	Input/output	This is an 8-bit I/O port equivalent to P0.
A_8 to A_{15}		Output	These pins output 8 middle-order address bits (A_8–A_{15}).
A_8/D_7, A_9 to A_{15}		Input/output Output	If the external bus is set as a 16-bit wide multiplexed bus, these pins input and output data (D_7) and output address (A_8) separated in time by multiplexing. They also output address (A_9–A_{15}).
$P4_0$ to $P4_7$	I/O port P4	Input/output	This is an 8-bit I/O port equivalent to P0.
$\overline{CS_0}$ to $\overline{CS_3}$, A_{16} to A_{19}		Output Output	These pins output $\overline{CS_0}$–$\overline{CS_3}$ signals and A_{16}–A_{19}. $\overline{CS_0}$–$\overline{CS_3}$ are chip select signals used to specify an access space. A_{16}–A_{19} are 4 high-order address bits.

Table 1-4. Pin Description of M16C/6N group (2)

Pin name	Signal name	I/O type	Function
$P5_0$ to $P5_7$	I/O port P5	Input/output	This is an 8-bit I/O port equivalent to P0. In single-chip mode, $P5_7$ 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 software.
$\overline{\mathrm{WRL}}$ / $\overline{\mathrm{WR}}$, $\overline{\mathrm{WRH}}$ / $\overline{\mathrm{BHE}}$, $\overline{\mathrm{RD}}$, BCLK, $\overline{\mathrm{HLDA}}$, $\overline{\mathrm{HOLD}}$, ALE, $\overline{\mathrm{RDY}}$		Output Output Output Output Output Input Output Input	Output WRL, WRH (WR and BHE), RD, BCLK, HLDA, and ALE signals. WRL and WRH, and BHE and WR can be switched using software control. ■ WRL, WRH, and RD selected With a 16-bit external data bus, data is written to even addresses when the WRL signal is L and to the odd addresses when the WRH signal is L . Data is read when RD is L . ■ WR, BHE, and RD selected Data is written when WR is L . Data is read when RD is L . Odd addresses are accessed when BHE is L . Use this mode when using an 8-bit external data bus. While the input level at the HOLD pin is L , the microcomputer is placed in the hold state. While in the hold state, HLDA outputs a L level. ALE is used to latch the address. While the input level of the RDY pin is L , the microcomputer is in the ready state. BCLK outputs a clock with the same cycle as the internal clock ϕ.
$P6_0$ to $P6_7$	I/O port P6	Input/output	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 software.
$P7_0$ to $P7_7$	I/O port P7	Input/output	This is an 8-bit I/O port equivalent to P0. Pins in this port also function as timer A0 - A3, timer B5, UART2 I/O or CAN1 transmit/receive data pins as selected by software.
$P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P8_5$	I/O port P8 Input port $P8_5$	Input/output Input/output Input/output Input	$P8_0$ to $P8_4$, $P8_6$ and $P8_7$ are I/O ports with the same functions as P0. Using software, they can be made to function as the I/O pins for timer A4 and the input pins for external interrupts. $P8_6$ and $P8_7$ can be set using software to function as the I/O pins for a sub clock generation circuit. In this case, connect a quartz oscillator between $P8_6$ (X_{COUT} pin) and $P8_7$ (X_{CIN} pin). $P8_5$ is an input-only port that also functions for NMI. The NMI interrupt is generated when the input at this pin changes from H to L . The NMI function cannot be cancelled using software. The pull-up cannot be set for this pin.
$P9_0$ to $P9_7$	I/O port P9	Input/output	This is an 8-bit I/O port equivalent to P0. Pins in this port also function as SI/O3 I/O pins, Timer B0 - B4 input pins, D-A converter output pins, A-D converter extended input pins, A-D trigger input pins or CAN0 transmit/receive data pins as selected by software.
$P10_0$ to $P10_7$	I/O port P10	Input/output	This is an 8-bit I/O port equivalent to P0. Pins in this port also function as A-D converter input pins. Furthermore, $P10_4$- $P10_7$ also function as input pins for the key input interrupt function.

Operation of Functional Blocks

The M16C/6N group accommodates several units in a single chip. These units include ROM and RAM to store instructions and data and the central processing unit (CPU) to execute arithmetic/logic operations. Also included are peripheral units such as CAN module, timers, serial I/O, D-A converter, DMAC, CRC calculation circuit, A-D converter, and I/O ports.

Each unit is explained in the following.

Memory

Figure 2-1 depicts the memory map of the M16C/6N group. The address space extends the 1M bytes from address 00000$_{16}$ to FFFFF$_{16}$. ROM is located from FFFFF$_{16}$ down. For example, in the M306N0MCT-XXXFP, there is 128K byte of internal ROM from E0000$_{16}$ to FFFFF$_{16}$. The vector table for fixed interrupts such as the reset and NMI are mapped to FFFDC$_{16}$ to FFFFF$_{16}$. The starting addresses of the interrupt routines are stored here. The address of the vector table for timer interrupts, etc., can be set as desired using the internal register (INTB). See the section on interrupts for details.

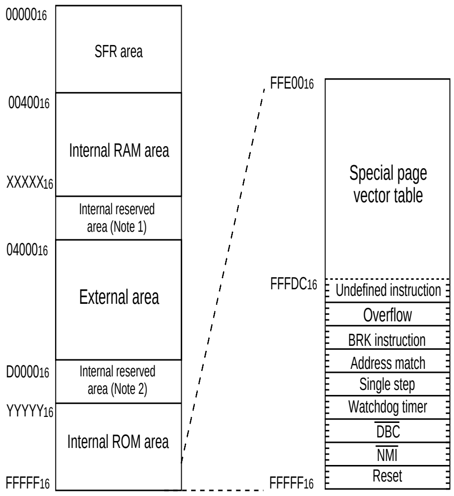
RAM is located from 00400$_{16}$ up. For example, in the M306N0MCT-XXXFP, 5K bytes of internal RAM are mapped to the space from 00400$_{16}$ to 017FF$_{16}$. In addition to storing data, the RAM also stores the stack used when calling subroutines and when interrupts are generated.

The SFR area is mapped to 00000$_{16}$ to 003FF$_{16}$. This area accommodates the control registers for peripheral devices such as I/O ports, A-D converter, serial I/O, CAN controller and timers, etc. Figure 2-2 to 2-9 are locations of peripheral unit control registers. Any part of the SFR area that is not occupied is reserved and cannot be used for other purposes.

The special page vector table is mapped to FFE00$_{16}$ to FFFDB$_{16}$. If the starting addresses of subroutines or the destination addresses of jumps are stored here, subroutine call instructions and jump instructions can be implemented as 2-byte instructions, reducing the number of program steps.

In memory expansion mode and microprocessor mode, a part of the space is reserved and cannot be used. For example, in the M306N0MCT-XXXFP, the following space cannot be used.

- The space between 01800$_{16}$ and 03FFF$_{16}$ (Memory expansion and microprocessor modes)
- The space between D0000$_{16}$ and DFFFF$_{16}$ (Memory expansion mode)



Type No.	Address XXXXX$_{16}$	Address YYYYY$_{16}$
M306N0MC	017FF$_{16}$	E0000$_{16}$
M306N0FG	02BFF$_{16}$	C0000$_{16}$

Note 1: During memory expansion and microprocessor modes, can not be used.
Note 2: In memory expansion mode, can not be used.

Figure 2-1. Memory map

0000_{16}	
0001_{16}	
0002_{16}	
0003_{16}	
0004_{16}	Processor mode register 0 (PM0)
0005_{16}	Processor mode register 1(PM1)
0006_{16}	System clock control register 0 (CM0)
0007_{16}	System clock control register 1 (CM1)
0008_{16}	Chip select control register (CSR)
0009_{16}	Address match interrupt enable register (AIER)
$000A_{16}$	Protect register (PRCR)
$000B_{16}$	
$000C_{16}$	Oscillation stop detection register (CM2)
$000D_{16}$	
$000E_{16}$	Watchdog timer start register (WDTS)
$000F_{16}$	Watchdog timer control register (WDC)
0010_{16}	Address match interrupt register 0 (RMAD0)
0011_{16}	
0012_{16}	
0013_{16}	
0014_{16}	Address match interrupt register 1 (RMAD1)
0015_{16}	
0016_{16}	
0017_{16}	
0018_{16}	
0019_{16}	
$001A_{16}$	
$001B_{16}$	
$001C_{16}$	
$001D_{16}$	
$001E_{16}$	
$001F_{16}$	
0020_{16}	DMA0 source pointer (SAR0)
0021_{16}	
0022_{16}	
0023_{16}	
0024_{16}	DMA0 destination pointer (DAR0)
0025_{16}	
0026_{16}	
0027_{16}	
0028_{16}	DMA0 transfer counter (TCR0)
0029_{16}	
$002A_{16}$	
$002B_{16}$	
$002C_{16}$	DMA0 control register (DM0CON)
$002D_{16}$	
$002E_{16}$	
$002F_{16}$	
0030_{16}	DMA1 source pointer (SAR1)
0031_{16}	
0032_{16}	
0033_{16}	
0034_{16}	DMA1 destination pointer (DAR1)
0035_{16}	
0036_{16}	
0037_{16}	
0038_{16}	DMA1 transfer counter (TCR1)
0039_{16}	
$003A_{16}$	
$003B_{16}$	
$003C_{16}$	DMA1 control register (DM1CON)
$003D_{16}$	
$003E_{16}$	
$003F_{16}$	

0040_{16}	
0041_{16}	CAN0/1 Wake Up interrupt control register (C01WKPIC)
0042_{16}	CAN0 receive successful interrupt control register (C0RECIC)
0043_{16}	CAN0 transmit successful interrupt control register (C0TRMIC)
0044_{16}	INT3 interrupt control register (INT3IC)
0045_{16}	Timer B5 interrupt control register (TB5IC)
0046_{16}	Timer B4 interrupt control register (TB4IC)
0047_{16}	Timer B3 interrupt control register (TB3IC)
0048_{16}	CAN1 receive successful interrupt control register (C1RECIC) INT5 interrupt control register (INT5IC)
0049_{16}	CAN1 transmit successful interrupt control register (C1TRMIC) SIO3 interrupt control register (S3IC) INT4 interrupt control register (INT4IC)
$004A_{16}$	Bus collision detection interrupt control register (BCNIC)
$004B_{16}$	DMA0 interrupt control register (DM0IC)
$004C_{16}$	DMA1 interrupt control register (DM1IC)
$004D_{16}$	CAN0/1 error interrupt control register (C01ERRIC)
$004E_{16}$	A-D conversion interrupt control register (ADIC) Key input interrupt control register (KUPIC)
$004F_{16}$	UART2 transmit interrupt control register (S2TIC)
0050_{16}	UART2 receive interrupt control register (S2RIC)
0051_{16}	UART0 transmit interrupt control register (S0TIC)
0052_{16}	UART0 receive interrupt control register (S0RIC)
0053_{16}	UART1 transmit interrupt control register (S1TIC)
0054_{16}	UART1 receive interrupt control register (S1RIC)
0055_{16}	Timer A0 interrupt control register (TA0IC)
0056_{16}	Timer A1 interrupt control register (TA1IC)
0057_{16}	Timer A2 interrupt control register (TA2IC)
0058_{16}	Timer A3 interrupt control register (TA3IC)
0059_{16}	Timer A4 interrupt control register (TA4IC)
$005A_{16}$	Timer B0 interrupt control register (TB0IC)
$005B_{16}$	Timer B1 interrupt control register (TB1IC)
$005C_{16}$	Timer B2 interrupt control register (TB2IC)
$005D_{16}$	INT0 interrupt control register (INT0IC)
$005E_{16}$	INT1 interrupt control register (INT1IC)
$005F_{16}$	INT2 interrupt control register (INT2IC)
0060_{16}	CAN0 Slot 0: Message Identifier / DLC
0061_{16}	
0062_{16}	
0063_{16}	
0064_{16}	
0065_{16}	
0066_{16}	CAN0 Slot 0: Data Field
0067_{16}	
0068_{16}	
0069_{16}	
$006A_{16}$	
$006B_{16}$	
$006C_{16}$	
$006D_{16}$	
$006E_{16}$	CAN0 Slot 0: Time Stamp
$006F_{16}$	
0070_{16}	CAN0 Slot 1: Message Identifier / DLC
0071_{16}	
0072_{16}	
0073_{16}	
0074_{16}	
0075_{16}	
0076_{16}	CAN0 Slot 1: Data Field
0077_{16}	
0078_{16}	
0079_{16}	
$007A_{16}$	
$007B_{16}$	
$007C_{16}$	
$007D_{16}$	
$007E_{16}$	CAN0 Slot 1: Time Stamp
$007F_{16}$	

Figure 2-2. Location of peripheral unit control registers (1)

0080$_{16}$	CAN0 Slot 2: Message Identifier / DLC	00C0$_{16}$	CAN0 Slot 6: Message Identifier / DLC
0081$_{16}$		00C1$_{16}$	
0082$_{16}$		00C2$_{16}$	
0083$_{16}$		00C3$_{16}$	
0084$_{16}$		00C4$_{16}$	
0085$_{16}$		00C5$_{16}$	
0086$_{16}$	CAN0 Slot 2: Data Field	00C6$_{16}$	CAN0 Slot 6: Data Field
0087$_{16}$		00C7$_{16}$	
0088$_{16}$		00C8$_{16}$	
0089$_{16}$		00C9$_{16}$	
008A$_{16}$		00CA$_{16}$	
008B$_{16}$		00CB$_{16}$	
008C$_{16}$		00CC$_{16}$	
008D$_{16}$		00CD$_{16}$	
008E$_{16}$	CAN0 Slot 2: Time Stamp	00CE$_{16}$	CAN0 Slot 6: Time Stamp
008F$_{16}$		00CF$_{16}$	
0090$_{16}$	CAN0 Slot 3: Message Identifier / DLC	00D0$_{16}$	CAN0 Slot 7: Message Identifier / DLC
0091$_{16}$		00D1$_{16}$	
0092$_{16}$		00D2$_{16}$	
0093$_{16}$		00D3$_{16}$	
0094$_{16}$		00D4$_{16}$	
0095$_{16}$		00D5$_{16}$	
0096$_{16}$	CAN0 Slot 3: Data Field	00D6$_{16}$	CAN0 Slot 7: Data Field
0097$_{16}$		00D7$_{16}$	
0098$_{16}$		00D8$_{16}$	
0099$_{16}$		00D9$_{16}$	
009A$_{16}$		00DA$_{16}$	
009B$_{16}$		00DB$_{16}$	
009C$_{16}$		00DC$_{16}$	
009D$_{16}$		00DD$_{16}$	
009E$_{16}$	CAN0 Slot 3: Time Stamp	00DE$_{16}$	CAN0 Slot 7: Time Stamp
009F$_{16}$		00DF$_{16}$	
00A0$_{16}$	CAN0 Slot 4: Message Identifier / DLC	00E0$_{16}$	CAN0 Slot 8: Message Identifier / DLC
00A1$_{16}$		00E1$_{16}$	
00A2$_{16}$		00E2$_{16}$	
00A3$_{16}$		00E3$_{16}$	
00A4$_{16}$		00E4$_{16}$	
00A5$_{16}$		00E5$_{16}$	
00A6$_{16}$	CAN0 Slot 4: Data Field	00E6$_{16}$	CAN0 Slot 8: Data Field
00A7$_{16}$		00E7$_{16}$	
00A8$_{16}$		00E8$_{16}$	
00A9$_{16}$		00E9$_{16}$	
00AA$_{16}$		00EA$_{16}$	
00AB$_{16}$		00EB$_{16}$	
00AC$_{16}$		00EC$_{16}$	
00AD$_{16}$		00ED$_{16}$	
00AE$_{16}$	CAN0 Slot 4: Time Stamp	00EE$_{16}$	CAN0 Slot 8: Time Stamp
00AF$_{16}$		00EF$_{16}$	
00B0$_{16}$	CAN0 Slot 5: Message Identifier / DLC	00F0$_{16}$	CAN0 Slot 9: Message Identifier / DLC
00B1$_{16}$		00F1$_{16}$	
00B2$_{16}$		00F2$_{16}$	
00B3$_{16}$		00F3$_{16}$	
00B4$_{16}$		00F4$_{16}$	
00B5$_{16}$		00F5$_{16}$	
00B6$_{16}$	CAN0 Slot 5: Data Field	00F6$_{16}$	CAN0 Slot 9: Data Field
00B7$_{16}$		00F7$_{16}$	
00B8$_{16}$		00F8$_{16}$	
00B9$_{16}$		00F9$_{16}$	
00BA$_{16}$		00FA$_{16}$	
00BB$_{16}$		00FB$_{16}$	
00BC$_{16}$		00FC$_{16}$	
00BD$_{16}$		00FD$_{16}$	
00BE$_{16}$	CAN0 Slot 5: Time Stamp	00FE$_{16}$	CAN0 Slot 9: Time Stamp
00BF$_{16}$		00FF$_{16}$	

Figure 2-3. Location of peripheral unit control registers (2)

0100_{16}	CAN0 Slot 10: Message Identifier / DLC	0140_{16}	CAN0 Slot 14: Message Identifier / DLC
0101_{16}		0141_{16}	
0102_{16}		0142_{16}	
0103_{16}		0143_{16}	
0104_{16}		0144_{16}	
0105_{16}		0145_{16}	
0106_{16}	CAN0 Slot 10: Data Field	0146_{16}	CAN0 Slot 14: Data Field
0107_{16}		0147_{16}	
0108_{16}		0148_{16}	
0109_{16}		0149_{16}	
$010A_{16}$		$014A_{16}$	
$010B_{16}$		$014B_{16}$	
$010C_{16}$		$014C_{16}$	
$010D_{16}$		$014D_{16}$	
$010E_{16}$	CAN0 Slot 10: Time Stamp	$014E_{16}$	CAN0 Slot 14: Time Stamp
$010F_{16}$		$014F_{16}$	
0110_{16}	CAN0 Slot 11: Message Identifier / DLC	0150_{16}	CAN0 Slot 15: Message Identifier / DLC
0111_{16}		0151_{16}	
0112_{16}		0152_{16}	
0113_{16}		0153_{16}	
0114_{16}		0154_{16}	
0115_{16}		0155_{16}	
0116_{16}	CAN0 Slot 11: Data Field	0156_{16}	CAN0 Slot 15: Data Field
0117_{16}		0157_{16}	
0118_{16}		0158_{16}	
0119_{16}		0159_{16}	
$011A_{16}$		$015A_{16}$	
$011B_{16}$		$015B_{16}$	
$011C_{16}$		$015C_{16}$	
$011D_{16}$		$015D_{16}$	
$011E_{16}$	CAN0 Slot 11: Time Stamp	$015E_{16}$	CAN0 Slot 15: Time Stamp
$011F_{16}$		$015F_{16}$	
0120_{16}	CAN0 Slot 12: Message Identifier / DLC	0160_{16}	CAN0 global mask (C0GMR)
0121_{16}		0161_{16}	
0122_{16}		0162_{16}	
0123_{16}		0163_{16}	
0124_{16}		0164_{16}	
0125_{16}		0165_{16}	
0126_{16}	CAN0 Slot 12: Data Field	0166_{16}	CAN0 local mask A (C0LMAR)
0127_{16}		0167_{16}	
0128_{16}		0168_{16}	
0129_{16}		0169_{16}	
$012A_{16}$		$016A_{16}$	
$012B_{16}$		$016B_{16}$	
$012C_{16}$		$016C_{16}$	CAN0 local mask B (C0LMBR)
$012D_{16}$		$016D_{16}$	
$012E_{16}$	CAN0 Slot 12: Time Stamp	$016E_{16}$	
$012F_{16}$		$016F_{16}$	
0130_{16}	CAN0 Slot 13: Message Identifier / DLC	0170_{16}	
0131_{16}		0171_{16}	
0132_{16}		0172_{16}	
0133_{16}		0173_{16}	
0134_{16}		0174_{16}	
0135_{16}		0175_{16}	
0136_{16}	CAN0 Slot 13: Data Field	0176_{16}	
0137_{16}		0177_{16}	
0138_{16}			
0139_{16}		$01B9_{16}$	
$013A_{16}$		$01BA_{16}$	
$013B_{16}$		$01BB_{16}$	
$013C_{16}$		$01BC_{16}$	
$013D_{16}$		$01BD_{16}$	
$013E_{16}$	CAN0 Slot 13: Time Stamp	$01BE_{16}$	
$013F_{16}$		$01BF_{16}$	

Figure 2-4. Location of peripheral unit control registers (3)

$01C0_{16}$	Timer B3,4,5 count start flag (TBSR)
$01C1_{16}$	
$01C2_{16}$	Timer A1-1 register (TA11)
$01C3_{16}$	
$01C4_{16}$	Timer A2-1 register (TA21)
$01C5_{16}$	
$01C6_{16}$	Timer A4-1 register (TA41)
$01C7_{16}$	
$01C8_{16}$	Three-phase PWM control register 0 (INVC0)
$01C9_{16}$	Three-phase PWM control register 1 (INVC1)
$01CA_{16}$	Three-phase output buffer register 0 (IDB0)
$01CB_{16}$	Three-phase output buffer register 1 (IDB1)
$01CC_{16}$	Dead time timer (DTT)
$01CD_{16}$	Timer B2 interrupt occurrence frequency set counter (ICTB2)
$01CE_{16}$	
$01CF_{16}$	
$01D0_{16}$	Timer B3 register (TB3)
$01D1_{16}$	
$01D2_{16}$	Timer B4 register (TB4)
$01D3_{16}$	
$01D4_{16}$	Timer B5 register (TB5)
$01D5_{16}$	
$01D6_{16}$	
$01D7_{16}$	
$01D8_{16}$	
$01D9_{16}$	
$01DA_{16}$	
$01DB_{16}$	Timer B3 mode register (TB3MR)
$01DC_{16}$	Timer B4 mode register (TB4MR)
$01DD_{16}$	Timer B5 mode register (TB5MR)
$01DE_{16}$	Interrupt cause select register 0 (IFSR0)
$01DF_{16}$	Interrupt cause select register 1 (IFSR1)
$01E0_{16}$	SI/O3 transmit/receive register (S3TRR)
$01E1_{16}$	
$01E2_{16}$	SI/O3 control register (S3C)
$01E3_{16}$	SI/O3 bit rate generator (S3BRG)
$01E4_{16}$	
$01E5_{16}$	
$01E6_{16}$	
$01E7_{16}$	
$01E8_{16}$	
$01E9_{16}$	
$01EA_{16}$	
$01EB_{16}$	
$01EC_{16}$	
$01ED_{16}$	
$01EE_{16}$	
$01EF_{16}$	
$01F0_{16}$	
$01F1_{16}$	
$01F2_{16}$	
$01F3_{16}$	
$01F4_{16}$	
$01F5_{16}$	
$01F6_{16}$	UART2 special mode register 2 (U2SMR2)
$01F7_{16}$	UART2 special mode register (U2SMR)
$01F8_{16}$	UART2 transmit/receive mode register (U2MR)
$01F9_{16}$	UART2 bit rate generator (U2BRG)
$01FA_{16}$	UART2 transmit buffer register (U2TB)
$01FB_{16}$	
$01FC_{16}$	UART2 transmit/receive mode register 0 (U2C0)
$01FD_{16}$	UART2 transmit/receive mode register 1 (U2C1)
$01FE_{16}$	UART2 receive buffer register (U2RB)
$01FF_{16}$	

0200_{16}	CAN0 message control register 0 (C0MCTL0)
0201_{16}	CAN0 message control register 1 (C0MCTL1)
0202_{16}	CAN0 message control register 2 (C0MCTL2)
0203_{16}	CAN0 message control register 3 (C0MCTL3)
0204_{16}	CAN0 message control register 4 (C0MCTL4)
0205_{16}	CAN0 message control register 5 (C0MCTL5)
0206_{16}	CAN0 message control register 6 (C0MCTL6)
0207_{16}	CAN0 message control register 7 (C0MCTL7)
0208_{16}	CAN0 message control register 8 (C0MCTL8)
0209_{16}	CAN0 message control register 9 (C0MCTL9)
$020A_{16}$	CAN0 message control register 10 (C0MCTL10)
$020B_{16}$	CAN0 message control register 11 (C0MCTL11)
$020C_{16}$	CAN0 message control register 12 (C0MCTL12)
$020D_{16}$	CAN0 message control register 13 (C0MCTL13)
$020E_{16}$	CAN0 message control register 14 (C0MCTL14)
$020F_{16}$	CAN0 message control register 15 (C0MCTL15)
0210_{16}	CAN0 control register (C0CTLR)
0211_{16}	
0212_{16}	CAN0 status register (C0STR)
0213_{16}	
0214_{16}	CAN0 slot status register (C0SSTR)
0215_{16}	
0216_{16}	CAN0 slot interrupt control register (C0SICR)
0217_{16}	
0218_{16}	CAN0 ExtID register (C0IDR)
0219_{16}	
$021A_{16}$	CAN0 configuration register (C0CONR)
$021B_{16}$	
$021C_{16}$	CAN0 REC register (C0RECR)
$021D_{16}$	CAN0 TEC register (C0TECR)
$021E_{16}$	CAN0 time stamp register (C0STR)
$021F_{16}$	
0220_{16}	CAN1 message control register 0 (C1MCTL0)
0221_{16}	CAN1 message control register 1 (C1MCTL1)
0222_{16}	CAN1 message control register 2 (C1MCTL2)
0223_{16}	CAN1 message control register 3 (C1MCTL3)
0224_{16}	CAN1 message control register 4 (C1MCTL4)
0225_{16}	CAN1 message control register 5 (C1MCTL5)
0226_{16}	CAN1 message control register 6 (C1MCTL6)
0227_{16}	CAN1 message control register 7 (C1MCTL7)
0228_{16}	CAN1 message control register 8 (C1MCTL8)
0229_{16}	CAN1 message control register 9 (C1MCTL9)
$022A_{16}$	CAN1 message control register 10 (C1MCTL10)
$022B_{16}$	CAN1 message control register 11 (C1MCTL11)
$022C_{16}$	CAN1 message control register 12 (C1MCTL12)
$022D_{16}$	CAN1 message control register 13 (C1MCTL13)
$022E_{16}$	CAN1 message control register 14 (C1MCTL14)
$022F_{16}$	CAN1 message control register 15 (C1MCTL15)
0230_{16}	CAN1 control register (C1CTLR)
0231_{16}	
0232_{16}	CAN1 status register (C1STR)
0233_{16}	
0234_{16}	CAN1 slot status register (C1SSTR)
0235_{16}	
0236_{16}	CAN1 slot interrupt control register (C1SICR)
0237_{16}	
0238_{16}	CAN1 ExtID register (C1IDR)
0239_{16}	
$023A_{16}$	CAN1 configuration register (C1CONR)
$023B_{16}$	
$023C_{16}$	CAN1 REC register (C1RECR)
$023D_{16}$	CAN1 TEC register (C1TECR)
$023E_{16}$	CAN1 time stamp register (C1STR)
$023F_{16}$	

Figure 2-5. Location of peripheral unit control registers (4)

0240$_{16}$		0280$_{16}$	CAN1 Slot 2: Message Identifier / DLC
0241$_{16}$		0281$_{16}$	
0242$_{16}$	CAN0 acceptance filter support register (C0AFS)	0282$_{16}$	
0243$_{16}$		0283$_{16}$	
0244$_{16}$	CAN1 acceptance filter support register (C1AFS)	0284$_{16}$	
0245$_{16}$		0285$_{16}$	
0246$_{16}$		0286$_{16}$	CAN1 Slot 2: Data Field
0247$_{16}$		0287$_{16}$	
0248$_{16}$		0288$_{16}$	
0249$_{16}$		0289$_{16}$	
024A$_{16}$		028A$_{16}$	
024B$_{16}$		028B$_{16}$	
024C$_{16}$		028C$_{16}$	
024D$_{16}$		028D$_{16}$	
024E$_{16}$		028E$_{16}$	CAN1 Slot 2: Time Stamp
024F$_{16}$		028F$_{16}$	
0250$_{16}$		0290$_{16}$	CAN1 Slot 3: Message Identifier / DLC
0251$_{16}$		0291$_{16}$	
0252$_{16}$		0292$_{16}$	
0253$_{16}$		0293$_{16}$	
0254$_{16}$		0294$_{16}$	
0255$_{16}$		0295$_{16}$	
0256$_{16}$		0296$_{16}$	CAN1 Slot 3: Data Field
0257$_{16}$		0297$_{16}$	
0258$_{16}$		0298$_{16}$	
0259$_{16}$		0299$_{16}$	
025A$_{16}$		029A$_{16}$	
025B$_{16}$		029B$_{16}$	
025C$_{16}$		029C$_{16}$	
025D$_{16}$		029D$_{16}$	
025E$_{16}$	Peripheral function clock select register (PCLKR)	029E$_{16}$	CAN1 Slot 3: Time Stamp
025F$_{16}$	CAN0/1 clock select register (C01CLKR)	029F$_{16}$	
0260$_{16}$	CAN1 Slot 0: Message Identifier / DLC	02A0$_{16}$	CAN1 Slot 4: Message Identifier / DLC
0261$_{16}$		02A1$_{16}$	
0262$_{16}$		02A2$_{16}$	
0263$_{16}$		02A3$_{16}$	
0264$_{16}$		02A4$_{16}$	
0265$_{16}$		02A5$_{16}$	
0266$_{16}$	CAN1 Slot 0: Data Field	02A6$_{16}$	CAN1 Slot 4: Data Field
0267$_{16}$		02A7$_{16}$	
0268$_{16}$		02A8$_{16}$	
0269$_{16}$		02A9$_{16}$	
026A$_{16}$		02AA$_{16}$	
026B$_{16}$		02AB$_{16}$	
026C$_{16}$		02AC$_{16}$	
026D$_{16}$		02AD$_{16}$	
026E$_{16}$	CAN1 Slot 0: Time Stamp	02AE$_{16}$	CAN1 Slot 4: Time Stamp
026F$_{16}$		02AF$_{16}$	
0270$_{16}$	CAN1 Slot 1: Message Identifier / DLC	02B0$_{16}$	CAN1 Slot 5: Message Identifier / DLC
0271$_{16}$		02B1$_{16}$	
0272$_{16}$		02B2$_{16}$	
0273$_{16}$		02B3$_{16}$	
0274$_{16}$		02B4$_{16}$	
0275$_{16}$		02B5$_{16}$	
0276$_{16}$	CAN1 Slot 1: Data Field	02B6$_{16}$	CAN1 Slot 5: Data Field
0277$_{16}$		02B7$_{16}$	
0278$_{16}$		02B8$_{16}$	
0279$_{16}$		02B9$_{16}$	
027A$_{16}$		02BA$_{16}$	
027B$_{16}$		02BB$_{16}$	
027C$_{16}$		02BC$_{16}$	
027D$_{16}$		02BD$_{16}$	
027E$_{16}$	CAN1 Slot 1: Time Stamp	02BE$_{16}$	CAN1 Slot 5: Time Stamp
027F$_{16}$		02BF$_{16}$	

Figure 2-6. Location of peripheral unit control registers (5)

02C0$_{16}$	CAN1 Slot 6: Message Identifier / DLC	0300$_{16}$	CAN1 Slot 10: Message Identifier / DLC
02C1$_{16}$		0301$_{16}$	
02C2$_{16}$		0302$_{16}$	
02C3$_{16}$		0303$_{16}$	
02C4$_{16}$		0304$_{16}$	
02C5$_{16}$		0305$_{16}$	
02C6$_{16}$	CAN1 Slot 6: Data Field	0306$_{16}$	CAN1 Slot 10: Data Field
02C7$_{16}$		0307$_{16}$	
02C8$_{16}$		0308$_{16}$	
02C9$_{16}$		0309$_{16}$	
02CA$_{16}$		030A$_{16}$	
02CB$_{16}$		030B$_{16}$	
02CC$_{16}$		030C$_{16}$	
02CD$_{16}$		030D$_{16}$	
02CE$_{16}$	CAN1 Slot 6: Time Stamp	030E$_{16}$	CAN1 Slot 10: Time Stamp
02CF$_{16}$		030F$_{16}$	
02D0$_{16}$	CAN1 Slot 7: Message Identifier / DLC	0310$_{16}$	CAN1 Slot 11: Message Identifier / DLC
02D1$_{16}$		0311$_{16}$	
02D2$_{16}$		0312$_{16}$	
02D3$_{16}$		0313$_{16}$	
02D4$_{16}$		0314$_{16}$	
02D5$_{16}$		0315$_{16}$	
02D6$_{16}$	CAN1 Slot 7: Data Field	0316$_{16}$	CAN1 Slot 11: Data Field
02D7$_{16}$		0317$_{16}$	
02D8$_{16}$		0318$_{16}$	
02D9$_{16}$		0319$_{16}$	
02DA$_{16}$		031A$_{16}$	
02DB$_{16}$		031B$_{16}$	
02DC$_{16}$		031C$_{16}$	
02DD$_{16}$		031D$_{16}$	
02DE$_{16}$	CAN1 Slot 7: Time Stamp	031E$_{16}$	CAN1 Slot 11: Time Stamp
02DF$_{16}$		031F$_{16}$	
02E0$_{16}$	CAN1 Slot 8: Message Identifier / DLC	0320$_{16}$	CAN1 Slot 12: Message Identifier / DLC
02E1$_{16}$		0321$_{16}$	
02E2$_{16}$		0322$_{16}$	
02E3$_{16}$		0323$_{16}$	
02E4$_{16}$		0324$_{16}$	
02E5$_{16}$		0325$_{16}$	
02E6$_{16}$	CAN1 Slot 8: Data Field	0326$_{16}$	CAN1 Slot 12: Data Field
02E7$_{16}$		0327$_{16}$	
02E8$_{16}$		0328$_{16}$	
02E9$_{16}$		0329$_{16}$	
02EA$_{16}$		032A$_{16}$	
02EB$_{16}$		032B$_{16}$	
02EC$_{16}$		032C$_{16}$	
02ED$_{16}$		032D$_{16}$	
02EE$_{16}$	CAN1 Slot 8: Time Stamp	032E$_{16}$	CAN1 Slot 12: Time Stamp
02EF$_{16}$		032F$_{16}$	
02F0$_{16}$	CAN1 Slot 9: Message Identifier / DLC	0330$_{16}$	CAN1 Slot 13: Message Identifier / DLC
02F1$_{16}$		0331$_{16}$	
02F2$_{16}$		0332$_{16}$	
02F3$_{16}$		0333$_{16}$	
02F4$_{16}$		0334$_{16}$	
02F5$_{16}$		0335$_{16}$	
02F6$_{16}$	CAN1 Slot 9: Data Field	0336$_{16}$	CAN1 Slot 13: Data Field
02F7$_{16}$		0337$_{16}$	
02F8$_{16}$		0338$_{16}$	
02F9$_{16}$		0339$_{16}$	
02FA$_{16}$		033A$_{16}$	
02FB$_{16}$		033B$_{16}$	
02FC$_{16}$		033C$_{16}$	
02FD$_{16}$		033D$_{16}$	
02FE$_{16}$	CAN1 Slot 9: Time Stamp	033E$_{16}$	CAN1 Slot 13: Time Stamp
02FF$_{16}$		033F$_{16}$	

Figure 2-7. Location of peripheral unit control registers (6)

0340$_{16}$	CAN1 Slot 14: Message Identifier / DLC
0341$_{16}$	
0342$_{16}$	
0343$_{16}$	
0344$_{16}$	
0345$_{16}$	
0346$_{16}$	CAN1 Slot 14: Data Field
0347$_{16}$	
0348$_{16}$	
0349$_{16}$	
034A$_{16}$	
034B$_{16}$	
034C$_{16}$	
034D$_{16}$	
034E$_{16}$	CAN1 Slot 14: Time Stamp
034F$_{16}$	
0350$_{16}$	CAN1 Slot 15: Message Identifier / DLC
0351$_{16}$	
0352$_{16}$	
0353$_{16}$	
0354$_{16}$	
0355$_{16}$	
0356$_{16}$	CAN1 Slot 15: Data Field
0357$_{16}$	
0358$_{16}$	
0359$_{16}$	
035A$_{16}$	
035B$_{16}$	
035C$_{16}$	
035D$_{16}$	
035E$_{16}$	CAN1 Slot 15: Time Stamp
035F$_{16}$	
0360$_{16}$	CAN1 global mask (C1GMR)
0361$_{16}$	
0362$_{16}$	
0363$_{16}$	
0364$_{16}$	
0365$_{16}$	
0366$_{16}$	CAN1 local mask A (C1LMAR)
0367$_{16}$	
0368$_{16}$	
0369$_{16}$	
036A$_{16}$	
036B$_{16}$	
036C$_{16}$	CAN1 local mask B (C1LMBR)
036D$_{16}$	
036E$_{16}$	
036F$_{16}$	
0370$_{16}$	
0371$_{16}$	
0372$_{16}$	
0373$_{16}$	
0374$_{16}$	
0375$_{16}$	
0376$_{16}$	
0377$_{16}$	
0378$_{16}$	
0379$_{16}$	
037A$_{16}$	
037B$_{16}$	
037C$_{16}$	
037D$_{16}$	
037E$_{16}$	
037F$_{16}$	

0380$_{16}$	Count start flag (TABSR)
0381$_{16}$	Clock prescaler reset flag (CPSRF)
0382$_{16}$	One-shot start flag (ONSF)
0383$_{16}$	Trigger select register (TRGSR)
0384$_{16}$	Up-down flag (UDF)
0385$_{16}$	
0386$_{16}$	Timer A0 (TA0)
0387$_{16}$	
0388$_{16}$	Timer A1 (TA1)
0389$_{16}$	
038A$_{16}$	Timer A2 (TA2)
038B$_{16}$	
038C$_{16}$	Timer A3 (TA3)
038D$_{16}$	
038E$_{16}$	Timer A4 (TA4)
038F$_{16}$	
0390$_{16}$	Timer B0 (TB0)
0391$_{16}$	
0392$_{16}$	Timer B1 (TB1)
0393$_{16}$	
0394$_{16}$	Timer B2 (TB2)
0395$_{16}$	
0396$_{16}$	Timer A0 mode register (TA0MR)
0397$_{16}$	Timer A1 mode register (TA1MR)
0398$_{16}$	Timer A2 mode register (TA2MR)
0399$_{16}$	Timer A3 mode register (TA3MR)
039A$_{16}$	Timer A4 mode register (TA4MR)
039B$_{16}$	Timer B0 mode register (TB0MR)
039C$_{16}$	Timer B1 mode register (TB1MR)
039D$_{16}$	Timer B2 mode register (TB2MR)
039E$_{16}$	
039F$_{16}$	
03A0$_{16}$	UART0 transmit/receive mode register (U0MR)
03A1$_{16}$	UART0 bit rate generator (U0BRG)
03A2$_{16}$	UART0 transmit buffer register (U0TB)
03A3$_{16}$	
03A4$_{16}$	UART0 transmit/receive control register 0 (U0C0)
03A5$_{16}$	UART0 transmit/receive control register 1 (U0C1)
03A6$_{16}$	UART0 receive buffer register (U0RB)
03A7$_{16}$	
03A8$_{16}$	UART1 transmit/receive mode register (U1MR)
03A9$_{16}$	UART1 bit rate generator (U1BRG)
03AA$_{16}$	UART1 transmit buffer register (U1TB)
03AB$_{16}$	
03AC$_{16}$	UART1 transmit/receive control register 0 (U1C0)
03AD$_{16}$	UART1 transmit/receive control register 1 (U1C1)
03AE$_{16}$	UART1 receive buffer register (U1RB)
03AF$_{16}$	
03B0$_{16}$	UART transmit/receive control register 2 (UCON)
03B1$_{16}$	
03B2$_{16}$	
03B3$_{16}$	
03B4$_{16}$	
03B5$_{16}$	
03B6$_{16}$	Flash memory control register 2 (FMCR2)
03B7$_{16}$	Flash memory control register (FMCR)
03B8$_{16}$	DMA0 cause select register (DM0SL)
03B9$_{16}$	
03BA$_{16}$	DMA1 cause select register (DM1SL)
03BB$_{16}$	
03BC$_{16}$	CRC data register (CRCD)
03BD$_{16}$	
03BE$_{16}$	CRC input register (CRCIN)
03BF$_{16}$	

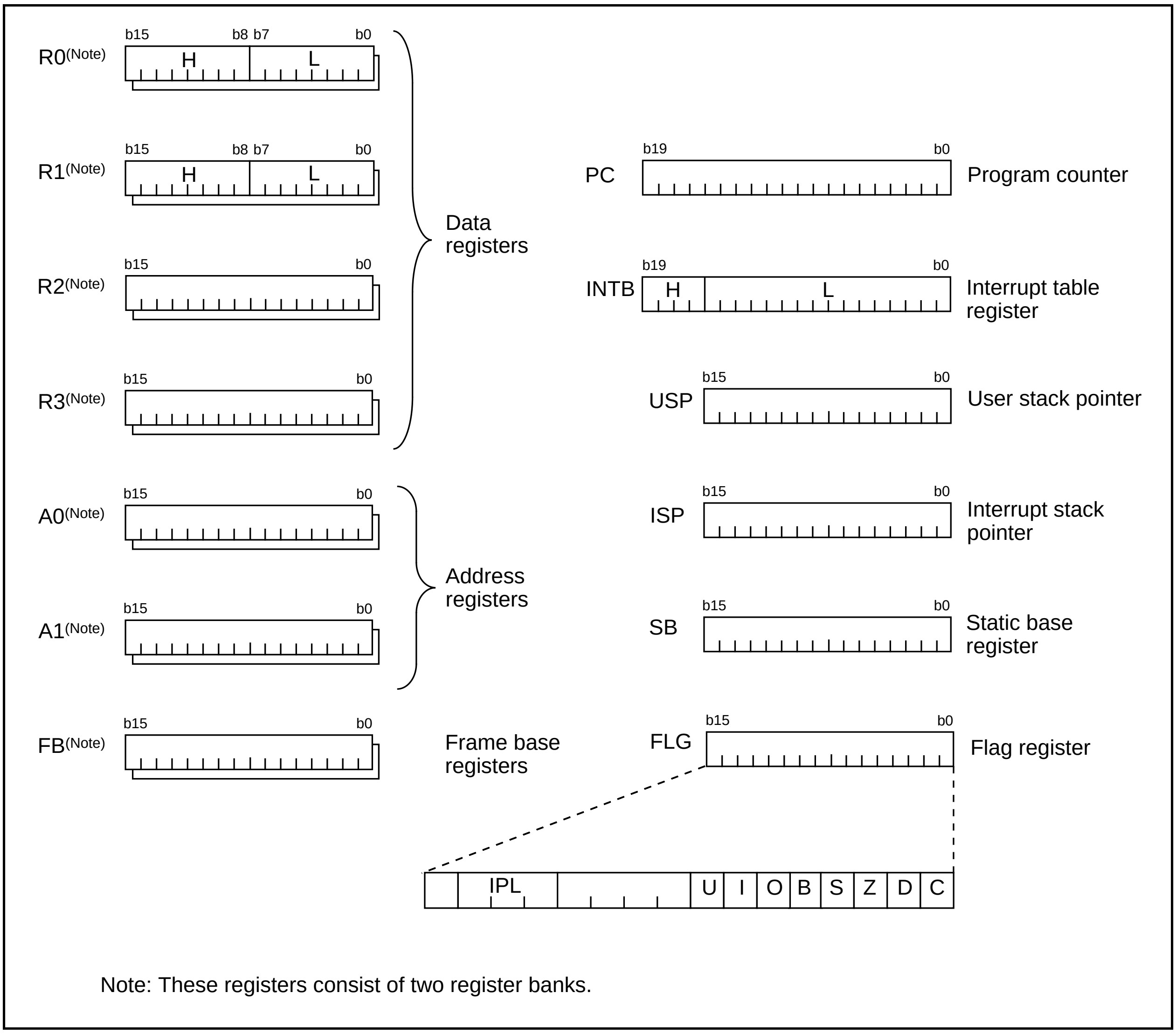
Figure 2-8. Location of peripheral unit control registers (7)

03C0$_{16}$	A-D register 0 (AD0)
03C1$_{16}$	
03C2$_{16}$	A-D register 1 (AD1)
03C3$_{16}$	
03C4$_{16}$	A-D register 2 (AD2)
03C5$_{16}$	
03C6$_{16}$	A-D register 3 (AD3)
03C7$_{16}$	
03C8$_{16}$	A-D register 4 (AD4)
03C9$_{16}$	
03CA$_{16}$	A-D register 5 (AD5)
03CB$_{16}$	
03CC$_{16}$	A-D register 6 (AD6)
03CD$_{16}$	
03CE$_{16}$	A-D register 7 (AD7)
03CF$_{16}$	
03D0$_{16}$	
03D1$_{16}$	
03D2$_{16}$	
03D3$_{16}$	
03D4$_{16}$	A-D control register 2 (ADCON2)
03D5$_{16}$	
03D6$_{16}$	A-D control register 0 (ADCON0)
03D7$_{16}$	A-D control register 1 (ADCON1)
03D8$_{16}$	D-A register 0 (DA0)
03D9$_{16}$	
03DA$_{16}$	D-A register 1 (DA1)
03DB$_{16}$	
03DC$_{16}$	D-A control register (DACON)
03DD$_{16}$	
03DE$_{16}$	
03DF$_{16}$	
03E0$_{16}$	Port P0 (P0)
03E1$_{16}$	Port P1 (P1)
03E2$_{16}$	Port P0 direction register (PD0)
03E3$_{16}$	Port P1 direction register (PD1)
03E4$_{16}$	Port P2 (P2)
03E5$_{16}$	Port P3 (P3)
03E6$_{16}$	Port P2 direction register (PD2)
03E7$_{16}$	Port P3 direction register (PD3)
03E8$_{16}$	Port P4 (P4)
03E9$_{16}$	Port P5 (P5)
03EA$_{16}$	Port P4 direction register (PD4)
03EB$_{16}$	Port P5 direction register (PD5)
03EC$_{16}$	Port P6 (P6)
03ED$_{16}$	Port P7 (P7)
03EE$_{16}$	Port P6 direction register (PD6)
03EF$_{16}$	Port P7 direction register (PD7)
03F0$_{16}$	Port P8 (P8)
03F1$_{16}$	Port P9 (P9)
03F2$_{16}$	Port P8 direction register (PD8)
03F3$_{16}$	Port P9 direction register (PD9)
03F4$_{16}$	Port P10 (P10)
03F5$_{16}$	
03F6$_{16}$	Port P10 direction register (PD10)
03F7$_{16}$	
03F8$_{16}$	
03F9$_{16}$	
03FA$_{16}$	
03FB$_{16}$	
03FC$_{16}$	Pull-up control register 0 (PUR0)
03FD$_{16}$	Pull-up control register 1 (PUR1)
03FE$_{16}$	Pull-up control register 2 (PUR2)
03FF$_{16}$	Port control register (PCR)

Figure 2-9. Location of peripheral unit control registers (8)

Central Processing Unit (CPU)

The CPU has a total of 13 registers shown in Figure 3-1. Seven of these registers (R0, R1, R2, R3, A0, A1, and FB) come in two sets; therefore, these registers have two register banks.



Note: These registers consist of two register banks.

Figure 3-1. Central processing unit register

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

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

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

(2) Address rgisters (A0 and A1)

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

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

(3) Frame base register (FB)

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

(4) Program counter (PC)

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

(5) Interrupt table register (INTB)

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

(6) Stack pointer (USP/ISP)

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

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

(7) Static base register (SB)

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

(8) Flag register (FLG)

The flag register (FLG) is configured with 11 bits, each bit is used as a flag. Figure 3-2 shows the flag register (FLG). The following explains the function of each flag:

- **Bit 0: Carry flag (C flag)**

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

- **Bit 1: Debug flag (D flag)**

  This flag enables a single-step interrupt.

  When this flag is “1”, a single-step interrupt is generated after instruction execution. This flag is cleared to “0” when the interrupt is acknowledged.

- **Bit 2: Zero flag (Z flag)**

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

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

  This flag is set to “1” when an arithmetic operation results in a negative value; otherwise, cleared to “0”.

- **Bit 4: Register bank select flag (B flag)**

  This flag chooses a register bank. Register bank 0 is selected when this flag is “0” ; register bank 1 is selected when this flag is “1”.

- **Bit 5: Overflow flag (O flag)**

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

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

  This flag enables a maskable interrupt.

  An interrupt is disabled when this flag is “0”, and is enabled when this flag is “1”. This flag is cleared to “0” when the interrupt is acknowledged.

- **Bit 7: Stack pointer select flag (U flag)**
  The interrupt stack pointer (ISP) is selected when this flag is “0” ; user stack pointer (USP) is selected when this flag is “1”.
  This flag is cleared to “0” when a hardware interrupt is acknowledged or an INT instruction of software interrupt Nos. 0 to 31 is executed.
- **Bits 8 to 11: Reserved area**
- **Bits 12 to 14: Processor interrupt priority level (IPL)**
  Processor interrupt priority level (IPL) is configured with three bits, for specification of up to eight processor interrupt priority levels from level 0 to level 7.
  If a requested interrupt has a priority greater than the processor interrupt priority level (IPL), the interrupt is enabled.
- **Bit 15: Reserved area**

The C, Z, S, and O flags are changed when instructions are executed. See the software manual for details.

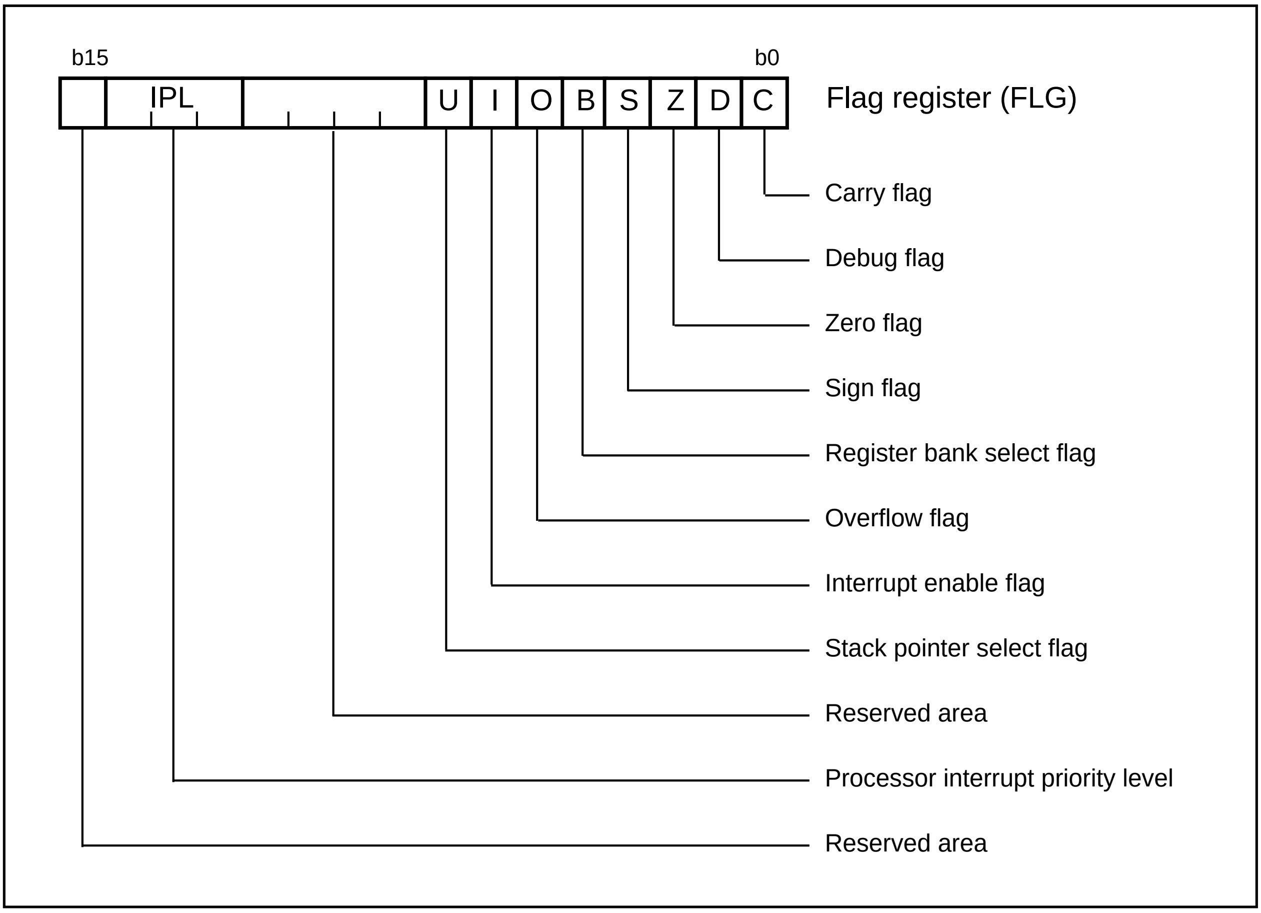


Figure 3-2. Flag register (FLG)

Processor Mode

(1) Processor Mode Types

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

- **Single-chip mode**

  In single-chip mode, only internal memory space (SFR, internal RAM, and internal ROM) can be accessed. Ports P0 to P10 can be used as programmable I/O ports or as I/O ports for the internal peripheral functions.

- **Memory expansion mode**

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

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

- **Microprocessor mode**

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

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

(2) Setting Processor Modes

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

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

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

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

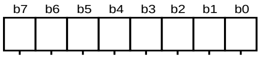
- **Applying V$_{CC}$ to CNV$_{SS}$ pin**

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

Figure 3-3 shows the processor mode register 0 and 1.

Figure 3-4 shows the memory maps applicable for each of the modes when memory area dose not be expanded (normal mode).

Processor mode register 0 (Note 1)

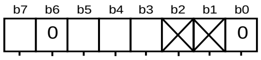


Symbol	Address	When reset
PM0	0004$_{16}$	00$_{16}$ (Note 2)

Bit symbol	Bit name	Function	R	W
PM00	Processor mode bit	b1 b0 0 0: Single-chip mode 0 1: Memory expansion mode 1 0: Inhibited 1 1: Microprocessor mode	○	○
PM01			○	○
PM02	R/W mode select bit	0 : $\overline{RD}$,$\overline{BHE}$,$\overline{WR}$ 1 : $\overline{RD}$,$\overline{WRH}$,$\overline{WRL}$	○	○
PM03	Software reset bit	The device is reset when this bit is set to “1”. The value of this bit is “0” when read.	○	○
PM04	Multiplexed bus space select bit	b5 b4 0 0 : Multiplexed bus is not used 0 1 : Allocated to CS2 space 1 0 : Allocated to CS1 space 1 1 : Allocated to entire space (Note4)	○	○
PM05			○	○
PM06	Port P4$_0$ to P4$_3$ function select bit (Note 3)	0 : Address output 1 : Port function (Address is not output)	○	○
PM07	BCLK output disable bit	0 : BCLK is output 1 : BCLK is not output (Pin is left floating)	○	○

Note 1: Set bit 1 of the protect register (address 000A$_{16}$) to “1” when writing new values to this register.
Note 2: If the VCC voltage is applied to the CNVSS, the value of this register when reset is 03$_{16}$. (PM00 and PM01 both are set to “1”.)
Note 3: Valid in microprocessor and memory expansion modes.
Note 4: If the entire space is of multiplexed bus in memory expansion mode, choose an 8-bit width.The processor operates using the separate bus after reset is revoked, so the entire space multiplexed bus cannot be chosen in microprocessor mode.
The higher-order address becomes a port if the entire space multiplexed bus is chosen, so only 256 bytes can be used in each chip select.

Processor mode register 1 (Note 1)

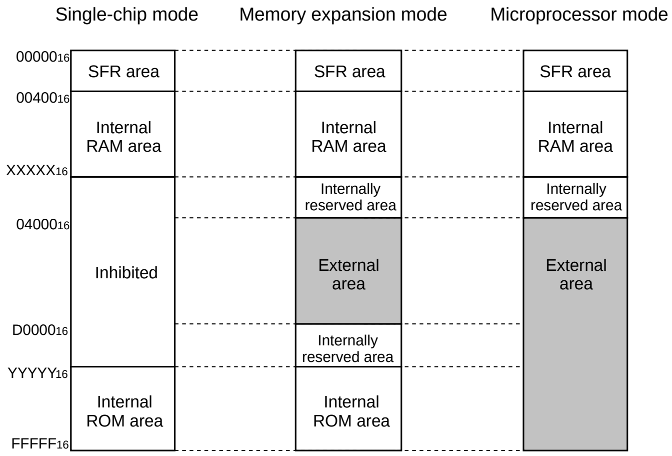


Symbol	Address	When reset
PM1	0005$_{16}$	00000XX0$_2$

Bit symbol	Bit name	Function	R	W
Reserved bit		Must always be set to “0”	○	○
Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminate.			—	—
PM13	Internal reserved area expansion bit (Note 2)	0: The same internal reserved area as that of M16C/60 and M16C/61 group 1: Expands the internal RAM area and internal ROM area to 23 K bytes and to 256K bytes respectively. (Note 2)	○	○
PM14	Memory area expansion bit (Note 3)	b5 b4 0 0 : Normal mode (Do not expand) 0 1 : Inhibited 1 0 : Memory area expansion mode 1 1 1 : Memory area expansion mode 2	○	○
PM15				
Reserved bit		Must always be set to “0”	○	○
PM17	Wait bit	0 : No wait state 1 : Wait state inserted	○	○

Note 1: Set bit 1 of the protect register (address 000A$_{16}$) to “1” when writing new values to this register.
Note 2: Be sure to set this bit to 0 except products whose RAM size and ROM size exceed 15K bytes and 192K bytes respectively.
Set this bit to "1" for M306N0FG.
Specify E0000$_{16}$ or a subsequent address, which becomes an internal ROM area if PM13 is set to “0” at the time reset is revoked, for the reset vector table of user program.
Note 3: With the processor running in memory expansion mode or in microprocessor mode, setting this bit provides the means of expanding the external memory area. (Normal mode: up to 1M byte, expansion mode 1: up to 1.2 M bytes, expansion mode 2: up to 4M bytes)
For details, see “Memory space expansion functions”.

Figure 3-3. Processor mode register 0 and 1



Type No.	Address $XXXXX_{16}$	Address $YYYYY_{16}$
M306N0MC	$017FF_{16}$	$E0000_{16}$
M306N0FG	$02BFF_{16}$	$C0000_{16}$

▨ External area : Accessing this area allows the user to access a device connected externally to the microcomputer.

Figure 3-4. Memory maps in each processor mode

Bus Settings

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

Table 3-1 shows the factors used to change the bus settings.

Table 3-1. Factors for switching bus settings

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

(1) Selecting external address bus width

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

(2) Selecting external data bus width

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

(3) Selecting separate/multiplex bus

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

- **Separate bus**

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

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

- **Multiplex bus**

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

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

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

  Before accessing the multiplex bus, always set the $\overline{CSi}$ wait bit of the chip select control register to “0”.

  In microprocessor mode, multiplexed bus for the entire space cannot be selected.

  In memory expansion mode, when multiplexed bus for the entire space is selected, address bus range is 256 bytes in each chip select.

Table 3-2. Pin functions for each processor mode

Processor mode	Single-chip mode	Memory expansion mode/microprocessor modes				Memory expansion mode
External bus type		Multiplexed bus and separate bus		separate bus		Multiplexed bus (Note 1)
Multiplexed bus space select bit		01 , 10		00		11 (Note 2)
Data bus width BYTE pin level		8 bits = H	16 bits = L	8 bits = H	16 bits = L	8 bita = H
P0$_0$ to P0$_7$	I/O port	Data bus	Data bus	Data bus	Data bus	I/O port
P1$_0$ to P1$_7$	I/O port	I/O port	Data bus	I/O port	Data bus	I/O port
P2$_0$	I/O port	Address bus /data bus(Note 3)	Address bus	Address bus	Address bus	Address bus /data bus
P2$_1$ to P2$_7$	I/O port	Address bus /data bus(Note 3)	Address bus /data bus(Note 3)	Address bus	Address bus	Address bus /data bus
P3$_0$	I/O port	Address bus	Address bus /data bus(Note 3)	Address bus	Address bus	I/O port
P3$_1$ to P3$_7$	I/O port	Address bus	Address bus	Address bus	Address bus	I/O port
P4$_0$ to P4$_3$ Port P4$_0$ to P4$_3$ function select bit = 1	I/O port	I/O port	I/O port	/O port	I/O port	I/O port
P4$_0$ to P4$_3$ Port P4$_0$ to P4$_3$ function select bit = 0	I/O port	Address bus	Address bus	Address bus	Address bus	I/O port
P4$_4$ to P4$_7$	I/O port	$\overline{\text{CS}}$ (chip select) or programmable I/O port (For details, refer to Bus control)				
P5$_0$ to P5$_3$	I/O port	Outputs $\overline{\text{RD}}$, $\overline{\text{WRL}}$, $\overline{\text{WRH}}$, and BCLK or $\overline{\text{RD}}$, $\overline{\text{BHE}}$, $\overline{\text{WR}}$, and BCLK (For details, refer to Bus control)				
P5$_4$	I/O port	$\overline{\text{HLDA}}$	$\overline{\text{HLDA}}$	$\overline{\text{HLDA}}$	$\overline{\text{HLDA}}$	$\overline{\text{HLDA}}$
P5$_5$	I/O port	$\overline{\text{HOLD}}$	$\overline{\text{HOLD}}$	$\overline{\text{HOLD}}$	$\overline{\text{HOLD}}$	$\overline{\text{HOLD}}$
P5$_6$	I/O port	ALE	ALE	ALE	ALE	ALE
P5$_7$	I/O port	$\overline{\text{RDY}}$	$\overline{\text{RDY}}$	$\overline{\text{RDY}}$	$\overline{\text{RDY}}$	$\overline{\text{RDY}}$

Note 1: In memory expansion mode, do not select a 16-bit multiplex bus.
Note 2: In microprocessor mode, multiplexed bus for the entire space cannot be selected.
In memory expansion mode, when multiplexed bus for the entire space is selected, address bus range is 256 bytes in each chip select.
Note 3: Address bus when in separate bus mode.

Bus Control

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

(1) Address bus/data bus

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

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

Both the address and data bus retain their previous states when internal ROM or RAM is accessed. Also, when a change is made from single-chip mode to memory expansion mode, the value of the address bus is undefined until external memory is accessed.

(2) Chip select signal

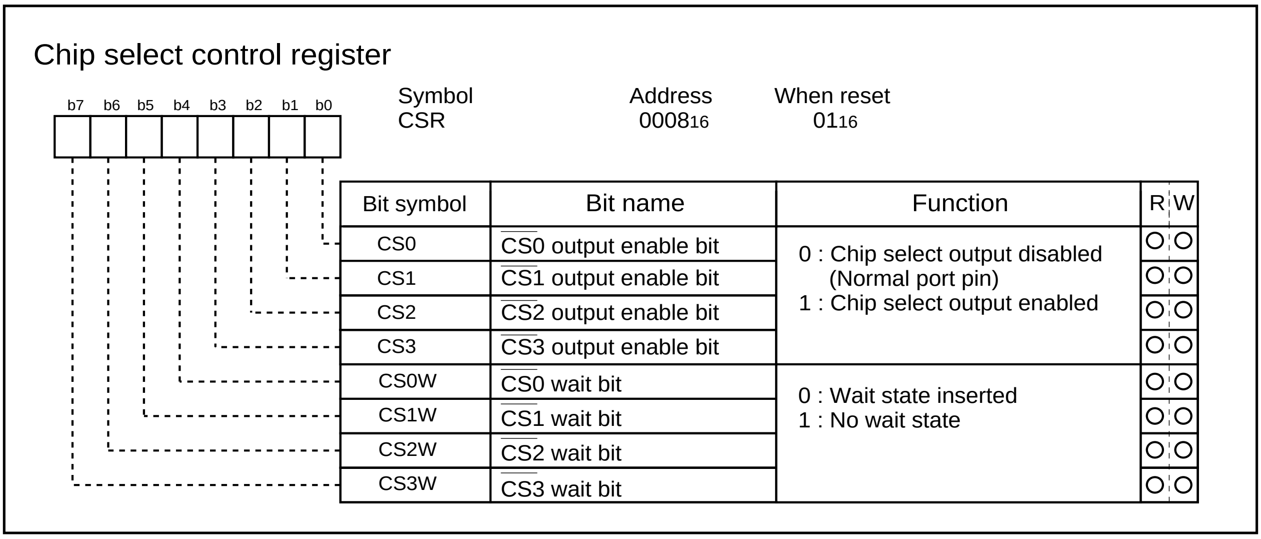
The chip select signal is output using the same pins as $P4_4$ to $P4_7$. Bits 0 to 3 of the chip select control register (address 0008_{16}) set each pin to function as a port or to output the chip select signal. The chip select control register is valid in memory expansion mode and microprocessor mode. In single-chip mode, $P4_4$ to $P4_7$ function as programmable I/O ports regardless of the value in the chip select control register.

In microprocessor mode, only $\overline{CS0}$ outputs the chip select signal after the reset state has been cancelled. $\overline{CS1}$ to $\overline{CS3}$ function as input ports. Figure 3-5 shows the chip select control register.

The chip select signal can be used to split the external area into as many as four blocks. Table 3-3 shows the external memory areas specified using the chip select signal.

Table 3-3. External areas specified by the chip select signals

Chip select	Special address range	
	Memory expansion mode	Microprocessor mode
$\overline{CS0}$	30000_{16} to $CFFFF_{16}$ (640K)	30000_{16} to $FFFFF_{16}$ (832K)
$\overline{CS1}$	28000_{16} to $2FFFF_{16}$ (32K)	28000_{16} to $2FFFF_{16}$ (32K)
$\overline{CS2}$	08000_{16} to $27FFF_{16}$ (128K)	08000_{16} to $27FFF_{16}$ (128K)
$\overline{CS3}$	04000_{16} to $07FFF_{16}$ (16K)	04000_{16} to $07FFF_{16}$ (16K)



Chip select control register

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

Bit symbol	Bit name	Function	R	W
CS0	$\overline{\mathrm{CS0}}$ output enable bit	0 : Chip select output disabled (Normal port pin) 1 : Chip select output enabled	○	○
CS1	$\overline{\mathrm{CS1}}$ output enable bit		○	○
CS2	$\overline{\mathrm{CS2}}$ output enable bit		○	○
CS3	$\overline{\mathrm{CS3}}$ output enable bit		○	○
CS0W	$\overline{\mathrm{CS0}}$ wait bit	0 : Wait state inserted 1 : No wait state	○	○
CS1W	$\overline{\mathrm{CS1}}$ wait bit		○	○
CS2W	$\overline{\mathrm{CS2}}$ wait bit		○	○
CS3W	$\overline{\mathrm{CS3}}$ wait bit		○	○

Figure 3-5. Chip select control register

(3) Read/write signals

With a 16-bit data bus (BYTE pin =“L”), bit 2 of the processor mode register 0 (address 0004_{16}) select the combinations of $\overline{\mathrm{RD}}$, $\overline{\mathrm{BHE}}$, and $\overline{\mathrm{WR}}$ signals or $\overline{\mathrm{RD}}$, $\overline{\mathrm{WRL}}$, and $\overline{\mathrm{WRH}}$ signals. With an 8-bit data bus (BYTE pin = “H”), use the combination of $\overline{\mathrm{RD}}$, $\overline{\mathrm{WR}}$, and $\overline{\mathrm{BHE}}$ signals. (Set bit 2 of the processor mode register 0 (address 0004_{16}) to “0”.) Tables 3-4 and 3-5 show the operation of these signals.

After a reset has been cancelled, the combination of $\overline{\mathrm{RD}}$, $\overline{\mathrm{WR}}$, and $\overline{\mathrm{BHE}}$ signals is automatically selected.

When switching to the $\overline{\mathrm{RD}}$, $\overline{\mathrm{WRL}}$, and $\overline{\mathrm{WRH}}$ combination, do not write to external memory until bit 2 of the processor mode register 0 (address 0004_{16}) has been set (Note).

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

Table 3-4. Operation of $\overline{\mathrm{RD}}$, $\overline{\mathrm{WRL}}$, and $\overline{\mathrm{WRH}}$ signals

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

Table 3-5. Operation of $\overline{\mathrm{RD}}$, $\overline{\mathrm{WR}}$, and $\overline{\mathrm{BHE}}$ signals

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

(4) ALE signal

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

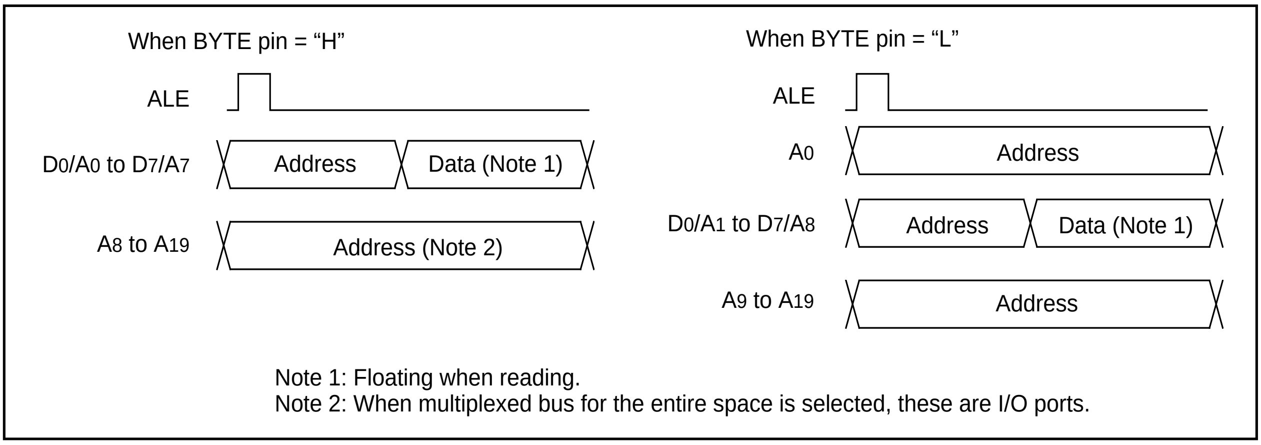


Figure 3-6. ALE sigal and address/data bus

(5) Ready signal

The ready signal facilitates access of external devices that require a long time for access. As shown in Figure 3-7, inputting “L” to the $\overline{\text{RDY}}$ pin at the falling edge of BCLK causes the microcomputer to enter the ready state. Inputting “H” to the $\overline{\text{RDY}}$ pin at the falling edge of BCLK cancels the ready state. Table 3-6 shows the microcomputer status in the ready state. Figure 3-7 shows the example of the $\overline{\text{RD}}$ signal being extended using the $\overline{\text{RDY}}$ signal.

Ready is valid when accessing the external area during the bus cycle in which the software wait is applied.

Table 3-6. Microcomputer status in ready state (Note)

Item	Status
Oscillation	On
R/$\overline{\text{W}}$ signal, address bus, data bus, $\overline{\text{CS}}$ ALE signal, $\overline{\text{HLDA}}$, programmable I/O ports	Maintain status when ready signal received
Internal peripheral circuits	On

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

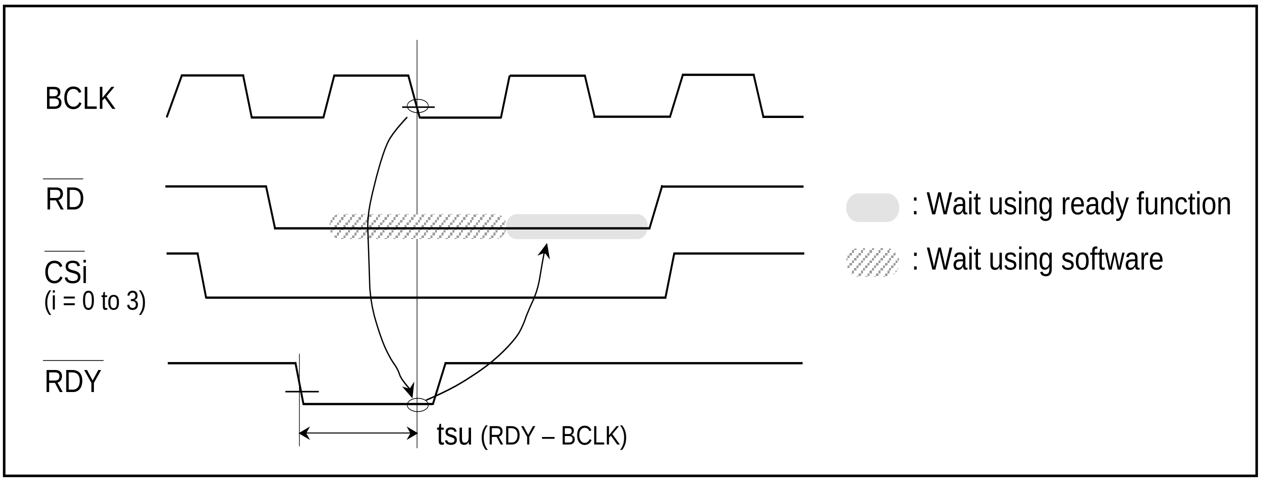


Figure 3-7. Example of RD signal extended by RDY signal

(6) Hold signal

The hold signal is used to transfer the bus privileges from the CPU to the external circuits. Inputting “L” to the $\overline{\text{HOLD}}$ pin places the microcomputer in the hold state at the end of the current bus access. This status is maintained and “L” is output from the $\overline{\text{HLDA}}$ pin as long as “L” is input to the $\overline{\text{HOLD}}$ pin. Table 3-7 shows the microcomputer status in the hold state.

Table 3-7. Microcomputer status in hold state

Item		Status
Oscillation		ON
R/$\overline{\text{W}}$ signal, address bus, data bus, $\overline{\text{CS}}$, $\overline{\text{BHE}}$		Floating
Programmable I/O ports	P0, P1, P2, P3, P4, P5	Floating
	P6, P7, P8, P9, P10	Maintains status when hold signal is received
$\overline{\text{HLDA}}$		Output “L”
Internal peripheral circuits		ON (but watchdog timer stops)
ALE signal		Undefined

(7) BCLK output

The output of the internal clock ϕ can be selected using bit 7 of the processor mode register 0 (address 0004_{16}) (Note). The output is floating when bit 7 is set to “1”.

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

(8) Software wait

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

A software wait is inserted in the internal ROM/RAM area and in the external memory area by setting the wait bit of the processor mode register 1. When set to “0”, each bus cycle is executed in one BCLK cycle. When set to “1”, each bus cycle is executed in two or three BCLK cycles. After the microcomputer has been reset, this bit defaults to “0”. When set to “1”, bits 4 to 7 of the chip select control register are invalid and a wait is applied to all external memory areas (two or three BCLK cycles). However, this is not necessary if the oscillation frequency is less than 3MHz.

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

The SFR area is always accessed in two BCLK cycles regardless of the setting of these control bits. Also, the corresponding bits of the chip select control register must be set to “0” if using the multiplex bus to access the external memory area.

Table 3-8 shows the software wait and bus cycles. Figure 3-8 shows example bus timing when using software waits.

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

Table 3-8. Software waits and bus cycles

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

Note: Always set to "0".

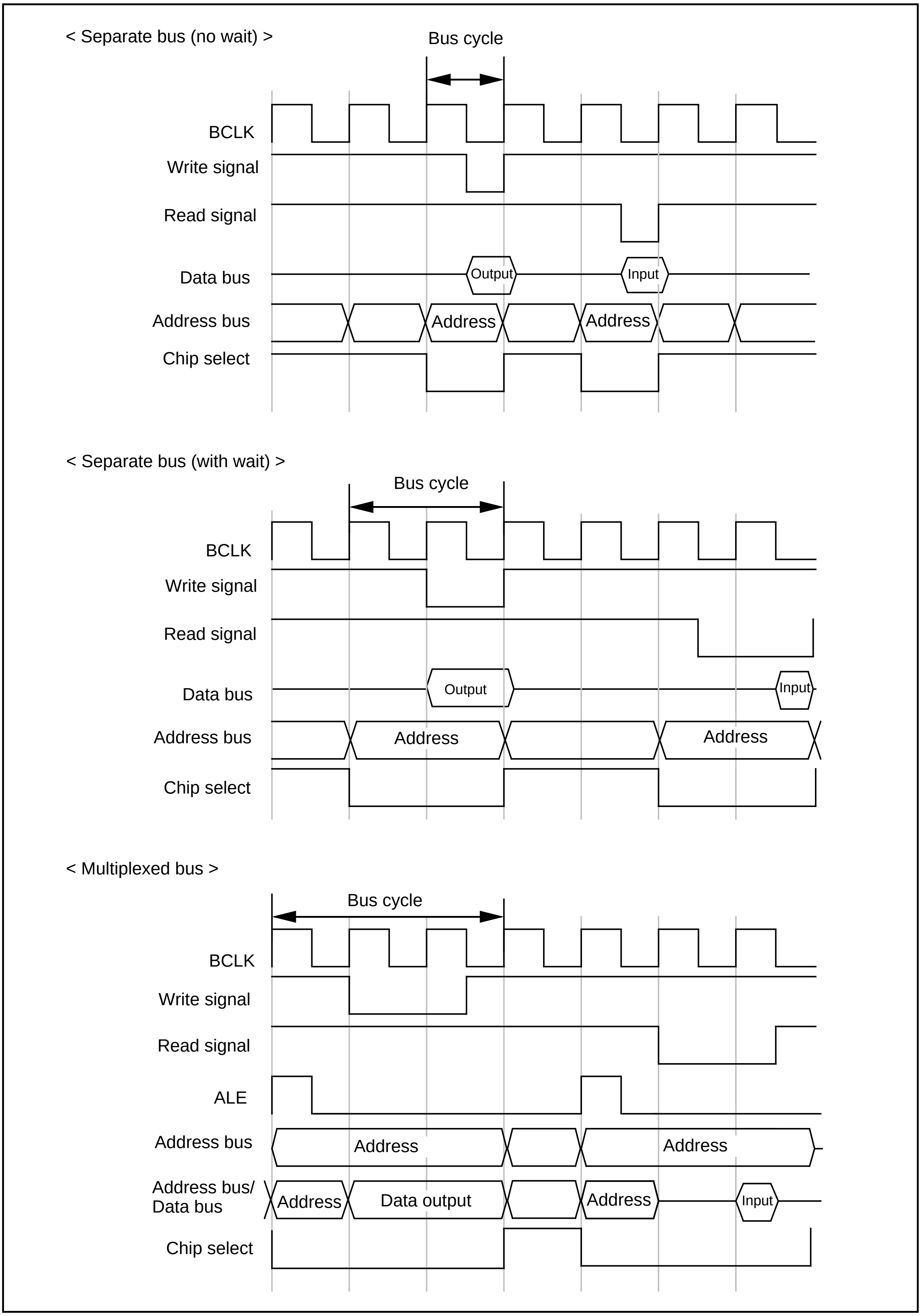


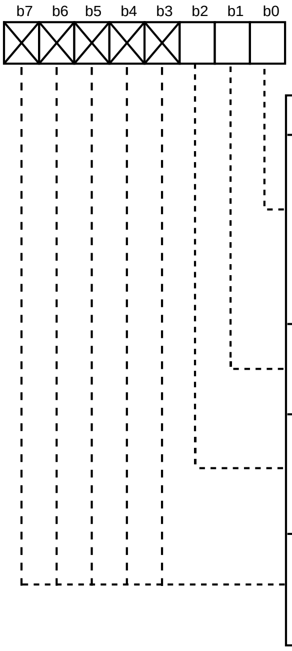
Figure 3-8. Typical bus timings using software wait

Protection

The protection function is provided so that the values in important registers cannot be changed in the event that the program runs out of control. Figure 3-9 shows the protect register. The values in the processor mode register 0 (address 0004_{16}), processor mode register 1 (address 0005_{16}), system clock control register 0 (address 0006_{16}), system clock control register 1 (address 0007_{16}), peripheral function clock select register (address $025E_{16}$), CAN0/1 clock select register (address $025F_{16}$), serial I/O 3 control register ($01E2_{16}$), port P7 direction register (address $03EF_{16}$) and port P9 direction register (address $03F3_{16}$) can only be changed when the respective bit in the protect register is set to “1”. Therefore, important outputs can be allocated to port P7 or port P9.

If, after “1” (write-enabled) has been written to the port P7 or port P9 direction registers write-enable bit (bit 2 at address $000A_{16}$), a value is written to any address, the bit automatically reverts to “0” (write-inhibited). However, the system clock control registers 0 and 1 write-enable bit (bit 0 at $000A_{16}$) and processor mode register 0 and 1 write-enable bit (bit 1 at $000A_{16}$) do not automatically return to “0” after a value has been written to an address. The program must therefore be written to return these bits to “0”.

Protect register



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

Bit symbol	Bit name	Function	R	W
PRC0	Enables writing to the system clock control registers 0 and 1 (addresses 0006_{16} and 0007_{16}), oscillation stop detection register (address $000C_{16}$), peripheral function clock select register (address $025E_{16}$), and CAN0/1 clock select register (address $025F_{16}$)	0 : Write-inhibited 1 : Write-enabled	○	○
PRC1	Enables writing to processor mode registers 0 and 1 (addresses 0004_{16} and 0005_{16})	0 : Write-inhibited 1 : Write-enabled	○	○
PRC2	Enables writing to port P7/9 direction register (address $03F3_{16}$ and $03EF_{16}$) and SI/O3 control register (address $01E2_{16}$) (Note)	0 : Write-inhibited 1 : Write-enabled	○	○
	Nothing is assigned. These bits can neither be set nor reset. When read, their contents are indeterminate.		—	—

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

Figure 3-9. Protect register

Reset

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

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

Figure 4-1 shows the example reset circuit. Figure 4-2 shows the reset sequence.

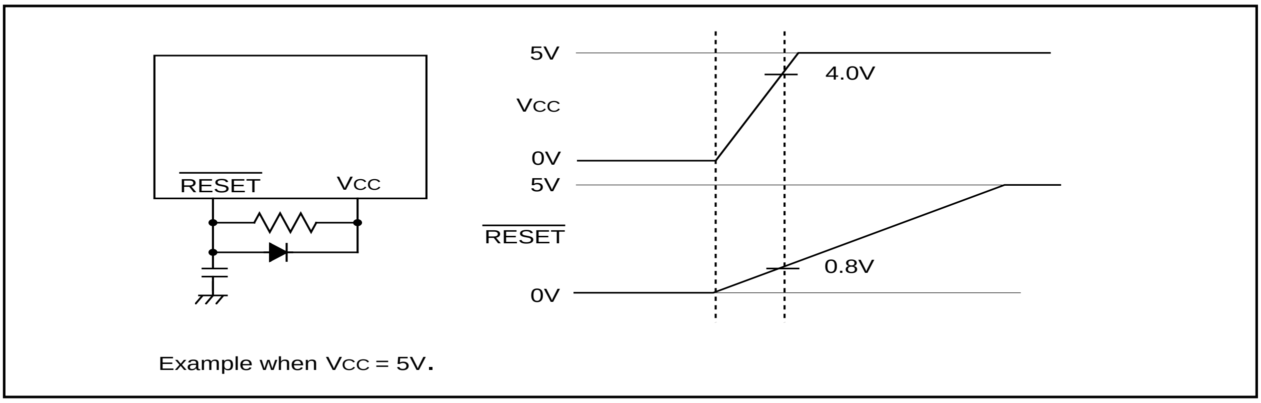


Figure 4-1. Example reset circuit

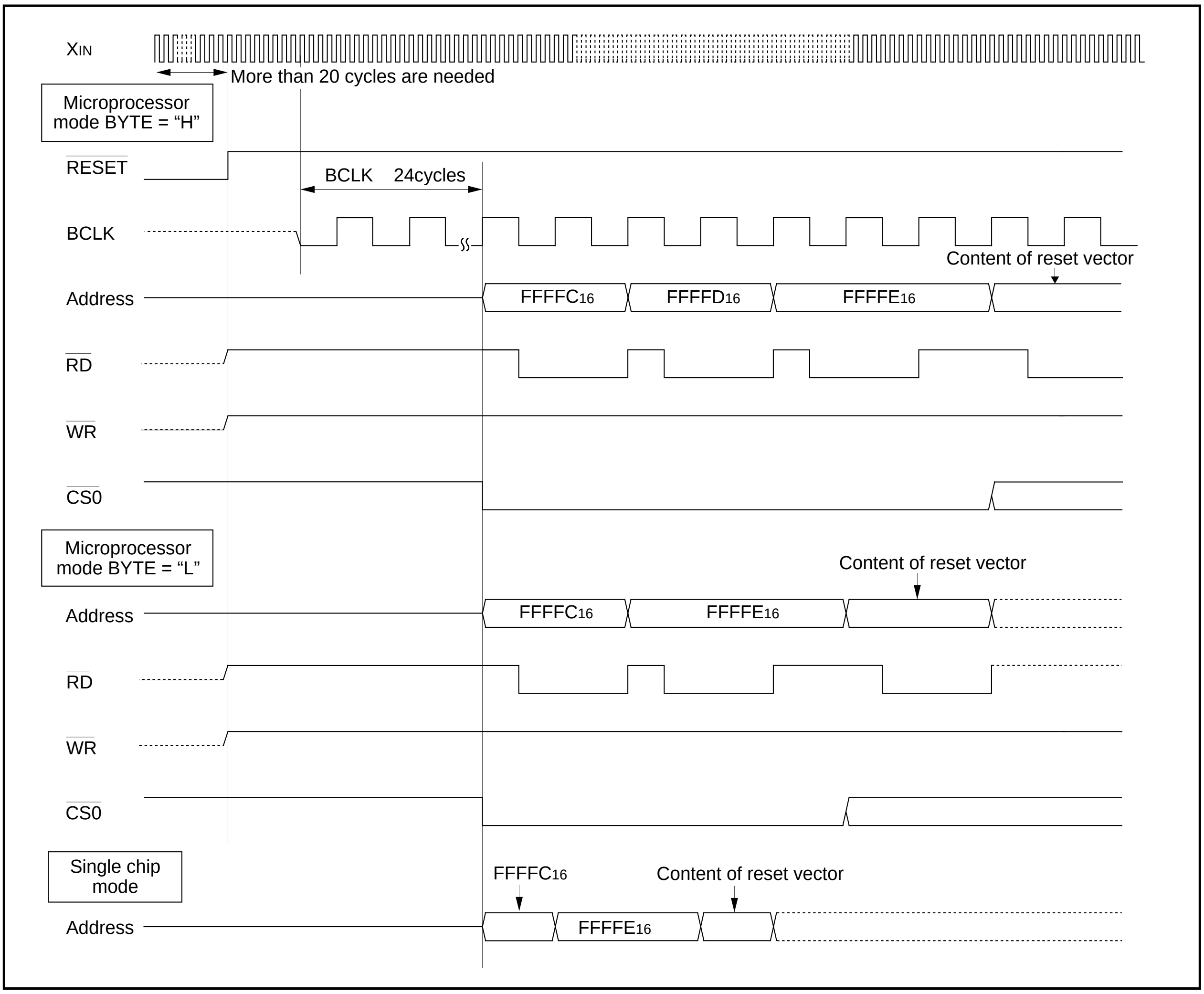


Figure 4-2. Reset sequence

Table 4-1 shows the statuses of the other pins while the $\overline{\text{RESET}}$ pin level is “L”. Figures 4-3, 4-4 and 4-5 show the internal status of the microcomputer immediately after the reset is released.

Table 4-1. Pin status when $\overline{\text{RESET}}$ pin level is “L”

Pin name	Status		
	CNVss = Vss	CNVss = Vcc	
		BYTE = Vss	BYTE = Vcc
P0	Input port (floating)	Data input (floating)	Data input (floating)
P1	Input port (floating)	Data input (floating)	Input port (floating)
P2, P3, P4$_0$ to P4$_3$	Input port (floating)	Address output (undefined)	Address output (undefined)
P4$_4$	Input port (floating)	$\overline{\text{CS0}}$ output (H level is output)	$\overline{\text{CS0}}$ output (H level is output)
P4$_5$ to P4$_7$	Input port (floating)	Input port (floating) (pull-up resistor is on)	Input port (floating) (pull-up resistor is on)
P5$_0$	Input port (floating)	$\overline{\text{WR}}$ output (H level is output)	$\overline{\text{WR}}$ output (H level is output)
P5$_1$	Input port (floating)	$\overline{\text{BHE}}$ output (undefined)	$\overline{\text{BHE}}$ output (undefined)
P5$_2$	Input port (floating)	$\overline{\text{RD}}$ output (H level is output)	$\overline{\text{RD}}$ output (H level is output)
P5$_3$	Input port (floating)	BCLK output	BCLK output
P5$_4$	Input port (floating)	$\overline{\text{HLDA}}$ output (The output value depends on the input to the $\overline{\text{HOLD}}$ pin)	$\overline{\text{HLDA}}$ output (The output value depends on the input to the $\overline{\text{HOLD}}$ pin)
P5$_5$	Input port (floating)	$\overline{\text{HOLD}}$ input (floating)	$\overline{\text{HOLD}}$ input (floating)
P5$_6$	Input port (floating)	ALE output (L level is output)	ALE output (L level is output)
P5$_7$	Input port (floating)	$\overline{\text{RDY}}$ input (floating)	$\overline{\text{RDY}}$ input (floating)
P6, P7, P8$_0$ to P8$_4$, P8$_6$, P8$_7$, P9, P10	Input port (floating)	Input port (floating)	Input port (floating)

(1)	Processor mode register 0 (Note)	(0004_{16})	00_{16}
(2)	Processor mode register 1	(0005_{16})	0 0 0 0 ? x x 0
(3)	System clock control register 0	(0006_{16})	0 1 0 0 1 0 0 0
(4)	System clock control register 1	(0007_{16})	0 0 1 0 0 0 0 0
(5)	Chip select control register	(0008_{16})	0 0 0 0 0 0 0 1
(6)	Address match interrupt enable register	(0009_{16})	x x x x x x 0 0
(7)	Protect register	($000A_{16}$)	x x x x x 0 0 0
(8)	Watchdog timer control register	($000F_{16}$)	0 0 0 ? ? ? ? ?
(9)	Address match interrupt register 0	(0010_{16})	00_{16}
		(0011_{16})	00_{16}
		(0012_{16})	x x x x 0 0 0 0
(10)	Address match interrupt register 1	(0014_{16})	00_{16}
		(0015_{16})	00_{16}
		(0016_{16})	x x x x 0 0 0 0
(11)	DMA0 control register	($002C_{16}$)	0 0 0 0 0 ? 0 0
(12)	DMA1 control register	($003C_{16}$)	0 0 0 0 0 ? 0 0
(13)	CAN0/1 wake up interrupt control register	(0041_{16})	x x x x ? 0 0 0
(14)	CAN0 receive successful interrupt control register	(0042_{16})	x x x x ? 0 0 0
(15)	CAN0 transmit successful interrupt control register	(0043_{16})	x x x x ? 0 0 0
(16)	INT3 interrupt control register	(0044_{16})	x x 0 0 ? 0 0 0
(17)	Timer B5 interrupt control register	(0045_{16})	x x x x ? 0 0 0
(18)	Timer B4 interrupt control register	(0046_{16})	x x x x ? 0 0 0
(19)	Timer B3 interrupt control register	(0047_{16})	x x x x ? 0 0 0
(20)	CAN1 receive successful interrupt control register	(0048_{16})	x x 0 0 ? 0 0 0
(21)	CAN1 transmit successful interrupt control register	(0049_{16})	x x 0 0 ? 0 0 0
(22)	Bus collision detection interrupt control register	($004A_{16}$)	x x x x ? 0 0 0
(23)	DMA0 interrupt control register	($004B_{16}$)	x x x x ? 0 0 0
(24)	DMA1 interrupt control register	($004C_{16}$)	x x x x ? 0 0 0
(25)	CAN0/1 state/error interrupt control register	($004D_{16}$)	x x x x ? 0 0 0
(26)	A-D conversion interrupt control register	($004E_{16}$)	x x x x ? 0 0 0
(27)	UART2 transmit interrupt control register	($004F_{16}$)	x x x x ? 0 0 0
(28)	UART2 receive interrupt control register	(0050_{16})	x x x x ? 0 0 0
(29)	UART0 transmit interrupt control register	(0051_{16})	x x x x ? 0 0 0
(30)	UART0 receive interrupt control register	(0052_{16})	x x x x ? 0 0 0
(31)	UART1 transmit interrupt control register	(0053_{16})	x x x x ? 0 0 0
(32)	UART1 receive interrupt control register	(0054_{16})	x x x x ? 0 0 0
(33)	Timer A0 interrupt control register	(0055_{16})	x x x x ? 0 0 0
(34)	Timer A1 interrupt control register	(0056_{16})	x x x x ? 0 0 0
(35)	Timer A2 interrupt control register	(0057_{16})	x x x x ? 0 0 0
(36)	Timer A3 interrupt control register	(0058_{16})	x x x x ? 0 0 0
(37)	Timer A4 interrupt control register	(0059_{16})	x x x x ? 0 0 0
(38)	Timer B0 interrupt control register	($005A_{16}$)	x x x x ? 0 0 0
(39)	Timer B1 interrupt control register	($005B_{16}$)	x x x x ? 0 0 0
(40)	Timer B2 interrupt control register	($005C_{16}$)	x x x x ? 0 0 0
(41)	INT0 interrupt control register	($005D_{16}$)	x x 0 0 ? 0 0 0
(42)	INT1 interrupt control register	($005E_{16}$)	x x 0 0 ? 0 0 0
(43)	INT2 interrupt control register	($005F_{16}$)	x x 0 0 ? 0 0 0
(44)	Timer B3,4,5 count start flag	($01C0_{16}$)	00_{16}
(45)	Three-phase PWM control register 0	($01C8_{16}$)	00_{16}
(46)	Three-phase PWM control register 1	($01C9_{16}$)	00_{16}
(47)	Three-phase output buffer register 0	($01CA_{16}$)	00_{16}
(48)	Three-phase output buffer register 1	($01CB_{16}$)	00_{16}
(49)	Timer B3 mode register	($01DB_{16}$)	0 0 ? x 0 0 0 0
(50)	Timer B4 mode register	($01DC_{16}$)	0 0 ? x 0 0 0 0
(51)	Timer B5 mode register	($01DD_{16}$)	0 0 ? x 0 0 0 0
(52)	Interrupt cause select register0	($01DE_{16}$)	x x x x x x 0 0
(53)	Interrupt cause select register1	($01DF_{16}$)	00_{16}
(54)	SI/O3 control register	($01E2_{16}$)	40_{16}
(55)	UART2 special mode register	($01F7_{16}$)	00_{16}
(56)	UART2 transmit/receive mode register	($01F8_{16}$)	00_{16}
(57)	UART2 transmit/receive control register 0	($01FC_{16}$)	0 0 0 0 1 0 0 0
(58)	UART2 transmit/receive control register 1	($01FD_{16}$)	0 0 0 0 0 0 1 0

x : Nothing is mapped to this bit
? : Undefined

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

Note: When the VCC level is applied to the CNVSS pin, it is 03_{16} at a reset.

Figure 4-3. Device's internal status after a reset is cleared

(59)CAN0 message control register 0	(0200_{16})...	00_{16}	(84)CAN1 message control register 0	(0220_{16})...	00_{16}
(60)CAN0 message control register 1	(0201_{16})...	00_{16}	(85)CAN1 message control register 1	(0221_{16})...	00_{16}
(61)CAN0 message control register 2	(0202_{16})...	00_{16}	(86)CAN1 message control register 2	(0222_{16})...	00_{16}
(62)CAN0 message control register 3	(0203_{16})...	00_{16}	(87)CAN1 message control register 3	(0223_{16})...	00_{16}
(63)CAN0 message control register 4	(0204_{16})...	00_{16}	(88)CAN1 message control register 4	(0224_{16})...	00_{16}
(64)CAN0 message control register 5	(0205_{16})...	00_{16}	(89)CAN1 message control register 5	(0225_{16})...	00_{16}
(65)CAN0 message control register 6	(0206_{16})...	00_{16}	(90)CAN1 message control register 6	(0226_{16})...	00_{16}
(66)CAN0 message control register 7	(0207_{16})...	00_{16}	(91)CAN1 message control register 7	(0227_{16})...	00_{16}
(67)CAN0 message control register 8	(0208_{16})...	00_{16}	(92)CAN1 message control register 8	(0228_{16})...	00_{16}
(68)CAN0 message control register 9	(0209_{16})...	00_{16}	(93)CAN1 message control register 9	(0229_{16})...	00_{16}
(69)CAN0 message control register 10	($020A_{16}$)...	00_{16}	(94)CAN1 message control register 10	($022A_{16}$)...	00_{16}
(70)CAN0 message control register 11	($020B_{16}$)...	00_{16}	(95)CAN1 message control register 11	($022B_{16}$)...	00_{16}
(71)CAN0 message control register 12	($020C_{16}$)...	00_{16}	(96)CAN1 message control register 12	($022C_{16}$)...	00_{16}
(72)CAN0 message control register 13	($020D_{16}$)...	00_{16}	(97)CAN1 message control register 13	($022D_{16}$)...	00_{16}
(73)CAN0 message control register 14	($020E_{16}$)...	00_{16}	(98)CAN1 message control register 14	($022E_{16}$)...	00_{16}
(74)CAN0 message control register 15	($020F_{16}$)...	00_{16}	(99)CAN1 message control register 15	($022F_{16}$)...	00_{16}
(75)CAN0 control register	(0210_{16})...	0 0 0 0 0 0 0 1	(100)CAN1 control register	(0230_{16})...	0 0 0 0 0 0 0 1
	(0211_{16})...	00_{16}		(0231_{16})...	00_{16}
(76)CAN0 status register	(0212_{16})...	00_{16}	(101)CAN1 status register	(0232_{16})...	00_{16}
	(0213_{16})...	0 0 0 0 0 0 0 1		(0233_{16})...	0 0 0 0 0 0 0 1
(77)CAN0 slot status register	(0214_{16})...	00_{16}	(102)CAN1 slot status register	(0234_{16})...	00_{16}
	(0215_{16})...	00_{16}		(0235_{16})...	00_{16}
(78)CAN0 interrupt control register	(0216_{16})...	00_{16}	(103)CAN1 interrupt control register	(0236_{16})...	00_{16}
	(0217_{16})...	00_{16}		(0237_{16})...	00_{16}
(79)CAN0 ExtID register	(0218_{16})...	00_{16}	(104)CAN1 ExtID register	(0238_{16})...	00_{16}
	(0219_{16})...	00_{16}		(0239_{16})...	00_{16}
(80)CAN0 configuration register	($021A_{16}$)...	XX_{16}	(105)CAN1 configuration register	($023A_{16}$)...	XX_{16}
	($021B_{16}$)...	XX_{16}		($023B_{16}$)...	XX_{16}
(81)CAN0 REC register	($021C_{16}$)...	00_{16}	(106)CAN1 REC register	($023C_{16}$)...	00_{16}
(82)CAN0 TEC register	($021D_{16}$)...	00_{16}	(107)CAN1 TEC register	($023D_{16}$)...	00_{16}
(83)CAN0 time stamp register	($021E_{16}$)...	00_{16}	(108)CAN1 time stamp register	($023E_{16}$)...	00_{16}

x : Nothing is mapped to this bit
? : Undefined

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

Figure 4-4. Device's internal status after a reset is cleared

(109) Peripheral function clock select register	(025E$_{16}$)...	00$_{16}$	(136) A-D control register 0	(03D6$_{16}$)...	0 0 0 0 0 ? ? ?
(110) CAN0/1 clock select register	(025F$_{16}$)...	☒ ☒ ☒ ☒ ☒ ☒ 0 0	(137) A-D control register 1	(03D7$_{16}$)...	00$_{16}$
(111) Count start flag	(0380$_{16}$)...	00$_{16}$	(138) D-A control register	(03DC$_{16}$)...	00$_{16}$
(112) Clock prescaler reset flag	(0381$_{16}$)...	0 ☒ ☒ ☒ ☒ ☒ ☒ ☒	(139) Port P0 direction register	(03E2$_{16}$)...	00$_{16}$
(113) One-shot start flag	(0382$_{16}$)...	0 0 ☒ 0 0 0 0 0	(140) Port P1 direction register	(03E3$_{16}$)...	00$_{16}$
(114) Trigger select flag	(0383$_{16}$)...	00$_{16}$	(141) Port P2 direction register	(03E6$_{16}$)...	00$_{16}$
(115) Up-down flag	(0384$_{16}$)...	00$_{16}$	(142) Port P3 direction register	(03E7$_{16}$)...	00$_{16}$
(116) Timer A0 mode register	(0396$_{16}$)...	00$_{16}$	(143) Port P4 direction register	(03EA$_{16}$)...	00$_{16}$
(117) Timer A1 mode register	(0397$_{16}$)...	00$_{16}$	(144) Port P5 direction register	(03EB$_{16}$)...	00$_{16}$
(118) Timer A2 mode register	(0398$_{16}$)...	00$_{16}$	(145) Port P6 direction register	(03EE$_{16}$)...	00$_{16}$
(119) Timer A3 mode register	(0399$_{16}$)...	00$_{16}$	(146) Port P7 direction register	(03EF$_{16}$)...	00$_{16}$
(120) Timer A4 mode register	(039A$_{16}$)...	00$_{16}$	(147) Port P8 direction register	(03F2$_{16}$)...	0 0 ☒ 0 0 0 0 0
(121) Timer B0 mode register	(039B$_{16}$)...	0 0 ? ☒ 0 0 0 0	(148) Port P9 direction register	(03F3$_{16}$)...	00$_{16}$
(122) Timer B1 mode register	(039C$_{16}$)...	0 0 ? ☒ 0 0 0 0	(149) Port P10 direction register	(03F6$_{16}$)...	00$_{16}$
(123) Timer B2 mode register	(039D$_{16}$)...	0 0 ? ☒ 0 0 0 0	(150) Pull-up control register 0	(03FC$_{16}$)...	00$_{16}$
(124) UART0 transmit/receive mode register	(03A0$_{16}$)...	00$_{16}$	(151) Pull-up control register 1 (Note)	(03FD$_{16}$)...	00$_{16}$
(125) UART0 transmit/receive control register 0	(03A4$_{16}$)...	0 0 0 0 1 0 0 0	(152) Pull-up control register 2	(03FE$_{16}$)...	00$_{16}$
(126) UART0 transmit/receive control register 1	(03A5$_{16}$)...	0 0 0 0 0 0 1 0	(153) Port control register	(03FF$_{16}$)...	00$_{16}$
(127) UART1 transmit/receive mode register	(03A8$_{16}$)...	00$_{16}$	(154) Data registers (R0/R1/R2/R3)		0000$_{16}$
(128) UART1 transmit/receive control register 0	(03AC$_{16}$)...	0 0 0 0 1 0 0 0	(155) Address registers (A0/A1)		0000$_{16}$
(129) UART1 transmit/receive control register 1	(03AD$_{16}$)...	0 0 0 0 0 0 1 0	(156) Frame base register (FB)		00000$_{16}$
(130) UART transmit/receive control register 2	(03B0$_{16}$)...	☒ 0 0 0 0 0 0 0	(157) Interrupt table register (INTB)		0000$_{16}$
(131) Flash memory control register 2	(03B6$_{16}$)...	☒ ☒ ☒ ☒ ☒ 0 ☒ ☒	(158) User stack pointer (USP)		0000$_{16}$
(132) Flash memory control register	(03B7$_{16}$)...	☒ ☒ 0 0 0 0 0 1	(159) Interrupt stack pointer (ISP)		0000$_{16}$
(133) DMA0 cause select register	(03B8$_{16}$)...	00$_{16}$	(160) Static base register (SB)		0000$_{16}$
(134) DMA1 cause select register	(03BA$_{16}$)...	00$_{16}$	(161) Flag register (FLG)		0000$_{16}$
(135) A-D control register 2	(03D4$_{16}$)...	☒ ☒ ☒ ☒ 0 0 0 0			

x : Nothing is mapped to this bit
? : Undefined

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

Note: When the VCC level is applied to the CNVSS pin, it is 02$_{16}$ at a reset.

Figure 4-5. Device's internal status after a reset is cleared

Software Reset

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

Clock Generating Circuit

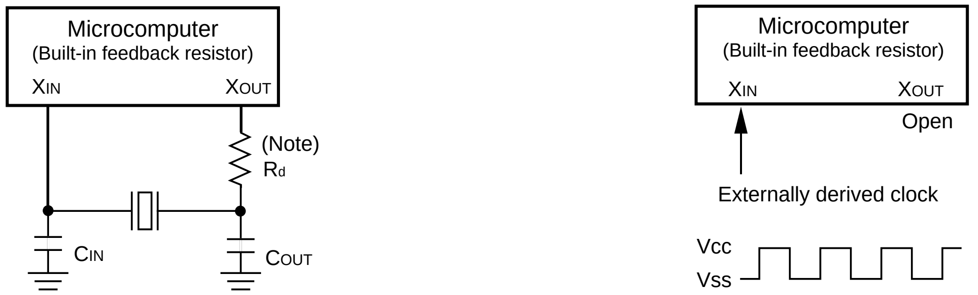
The clock generating circuit contains two oscillator circuits that supply the operating clock sources to the CPU and internal peripheral units.

Table 5-1. Main clock and sub clock generating circuits

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

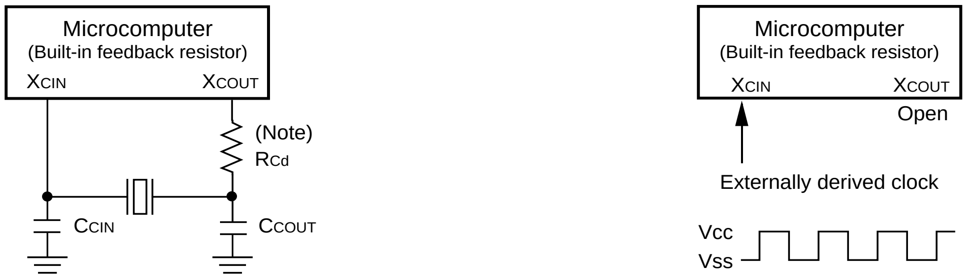
Example of oscillator circuit

Figure 5-1 shows some examples of the main clock circuit, one using an oscillator connected to the circuit, and the other one using an externally derived clock for input. Figure 5-2 shows some examples of sub clock circuits, one using an oscillator connected to the circuit, and the other one using an externally derived clock for input. Circuit constants in Figures 5-1 and 5-2 vary with each oscillator used. Use the values recommended by the manufacturer of your oscillator.



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

Figure 5-1. Examples of main clock



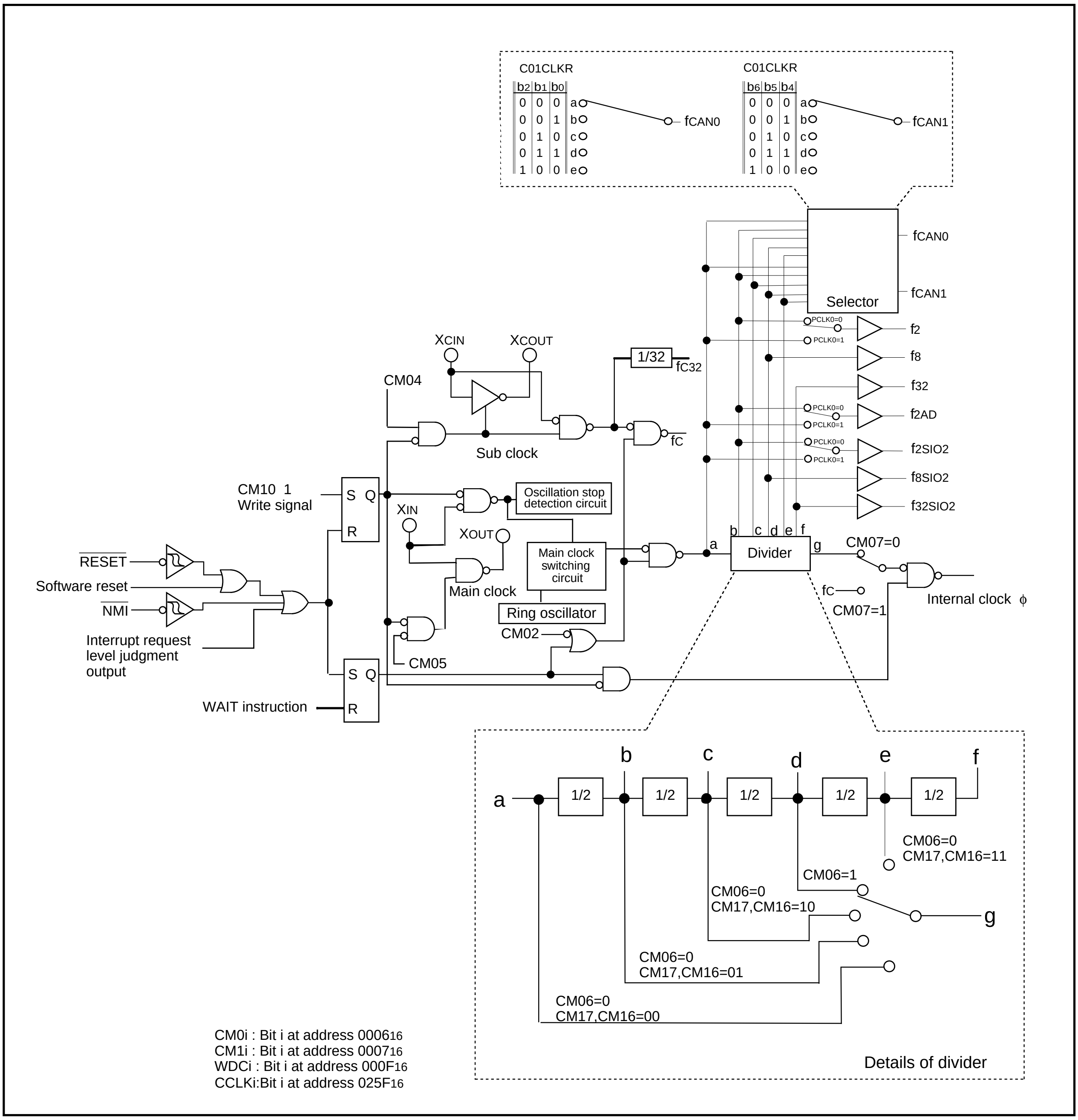
Note: Insert a damping resistor if required. The resistance will vary depending on the oscillator and the oscillation drive capacity setting. Use the value recommended by the maker of the oscillator.
When the oscillation drive capacity is set to low, check that oscillation is stable.

Figure 5-2. Examples of sub clock

A ring oscillator is built in the microcomputer. You can use it, instead of Xin, as a main clock by setup of the bit 1 of the oscillation stop detect register. You can use it when for example at such a wait time as executing confirmation of port value only. At this time, the frequency generated by the ring oscillator is low enough, compared to Xin, to realize a low power consumption.

Clock Control

Figure 5-3 shows the block diagram of the clock generating circuit.



CM0i : Bit i at address 0006_{16}
CM1i : Bit i at address 0007_{16}
WDCi : Bit i at address $000F_{16}$
CCLKi:Bit i at address $025F_{16}$

Figure 5-3. Clock generating circuit

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

(1) Main clock

The main clock is generated by the main clock oscillation circuit. After a reset, the clock is divided by 8 to form the BCLK. The clock can be stopped using the main clock stop bit (bit 5 at address 0006_{16}). Stopping the clock reduces the power consumption.

After the oscillation of the main clock oscillation circuit has stabilized, the drive capacity of the X_{OUT} pin can be reduced using the X_{IN}-X_{OUT} drive capacity select bit (bit 5 at address 0007_{16}). Reducing the drive capacity of the X_{OUT} pin reduces the power consumption. This bit defaults to “1” when shifting to stop mode and after a reset.

You can switch over from the main clock to the ring oscillator by changing the value of the main clock switch bit (bit 5 at address $000C_{16}$).

(2) Sub clock

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

After the oscillation of the sub clock oscillation circuit has stabilized, the drive capacity of the X_{COUT} pin can be reduced using the X_{CIN}-X_{COUT} drive capacity select bit (bit 3 at address 0006_{16}). Reducing the drive capacity of the X_{COUT} pin reduces the power consumption. This bit changes to “1” when shifting to stop mode and at a reset.

(3) BCLK

The BCLK is the clock that drives the CPU and the watchdog timer, i.e. the internal clock ϕ, and is either the main clock or fc or is derived by dividing the main clock by 2, 4, 8, or 16. After a reset the BCLK is derived by dividing the main clock by 8 .

When shifting to stop mode, the main clock division select bit (bit 6 at 0006_{16}) is set to “1”.

(4) Peripheral function clocks

- **f_2, f_8, f_{32}, f_{2SIO2}, f_{8SIO2}, f_{32SIO2}**

The clock for the peripheral devices is derived by dividing the main clock by 2(or no division), 8 or 32. The peripheral function clock is stopped by stopping the main clock or by setting the WAIT peripheral function clock stop bit (bit 2 at 0006_{16}) to “1” and then executing a WAIT instruction.

As to f_2 and f_{2SIO2}, you can select division by 2 or no division by changing the value of the peripheral function clock select register. Select the mode without division only when Xin is 16 MHz or lower.

- **f_{2AD}**

This clock is derived by dividing the main clock by 2(or no division) and is used for A-D conversion. You can select division by 2 or no division by changing the value of the peripheral function clock select register.

- **f_{CAN0} ,f_{CAN1}**

These clocks are derived by dividing the main clock by 1, 2, 4, 8 or 16 and they are used for the corresponding CAN module.

(5) f_{C32}

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

(6) f_C

This clock has the same frequency as the sub clock. It may be selected as the BCLK and for the watchdog timer.

Figure 5-4 shows the system clock control registers 0 and 1.

System clock control register 0 (Note 1)

b7	b6	b5	b4	b3	b2	b1	b0

Symbol	Address	When reset
CM0	0006_{16}	48_{16}

Bit symbol	Bit name	Function	R	W
CM00	Clock output function select bit	b1 b0 0 0 : I/O port P5$_7$ 0 1 : f_C output 1 0 : f_8 output 1 1 : f_{32} output	○	○
CM01			○	○
CM02	WAIT peripheral function clock stop bit	0 : Do not stop f_2, f_8, f_{32} in wait mode 1 : Stop f_2, f_8, f_{32} in wait mode	○	○
CM03	X_{CIN}-X_{COUT} drive capacity select bit (Note 2)	0 : LOW 1 : HIGH	○	○
CM04	Port X$_C$ select bit	0 : I/O port 1 : X_{CIN}-X_{COUT} generation	○	○
CM05	Main clock (X_{IN}-X_{OUT}) stop bit (Note 3, 4 and 5)	0 : On 1 : Off	○	○
CM06	Main clock division select bit 0 (Note 2)	0 : CM16 and CM17 valid 1 : Division by 8 mode	○	○
CM07	System clock select bit (Note 6)	0 : X_{IN}, X_{OUT} 1 : X_{CIN}, X_{COUT}	○	○

Note 1: Set bit 0 of the protect register (address $000A_{16}$) to “1” before writing to this register.
Note 2: Changes to “1” when shifting to stop mode.
Note 3: When entering power saving mode, main clock stops using this bit. When returning from stop mode and operating with X_{IN}, set this bit to “0”. When main clock oscillation is operating by itself, set system clock select bit (CM07) to “1” before setting this bit to “1”.
Note 4: When inputting external clock, only clock oscillation buffer is stopped and clock input is acceptable.
Note 5: If this bit is set to “1”, X_{OUT} turns "H". The built-in feedback resistor remains ON, so X_{IN} turns pulled up to X_{OUT} ("H") via the feedback resistor.
Note 6: Set port Xc select bit (CM04) to “1” before writing to this bit. The both bits can not be written at the same time.

System clock control register 1 (Note 1)

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

Symbol	Address	When reset
CM1	0007_{16}	20_{16}

Bit symbol	Bit name	Function	R	W
CM10	All clock stop control bit (Note 4)	0 : Clock on 1 : All clocks off (stop mode)	○	○
Reserved bit		Always set to “0”	○	○
Reserved bit		Always set to “0”	○	○
Reserved bit		Always set to “0”	○	○
Reserved bit		Always set to “0”	○	○
CM15	X_{IN}-X_{OUT} drive capacity select bit (Note 2)	0 : LOW 1 : HIGH	○	○
CM16	Main clock division select bit 1 (Note 3)	b7 b6 0 0 : No division mode 0 1 : Division by 2 mode 1 0 : Division by 4 mode 1 1 : Division by 16 mode	○	○
CM17				

Note 1: Set bit 0 of the protect register (address $000A_{16}$) to “1” before writing to this register.
Note 2: Changes to “1” when shifting to stop mode.
Note 3: Can be selected when bit 6 of the system clock control register 0 (address 0006_{16}) is “0”. If “1”, division mode is fixed at 8.
Note 4: If this bit is set to “1”, X_{OUT} turns "H", and the built-in feedback resistor turns null.

Figure 5-4. Clock control registers 0 and 1

Figure 5-5 shows the CAN0/1 clock select register and Figure 5-6 shows the peripheral function clock select register.

CAN0/1 clock select register (Note 1, Note 2)

b7 b6 b5 b4 b3 b2 b1 b0

Symbol	Address	When reset
C01CLKR	025F$_{16}$	00$_{16}$

Bit symbol	Bit name	Function	R	W
CCLK0	CAN0 Clock select bit	b2 b1 b0 0 0 0: No division mode 0 0 1: Division by 2 mode 0 1 0: Division by 4 mode 0 1 1: Division by 8 mode 1 0 0: Division by 16 mode	○	○
CCLK1			○	○
CCLK2			○	○
CCLK3	Reserved bit	Always set to "0"	○	○
CCLK4	CAN1 Clock select bit	b6 b5 b4 0 0 0: No division mode 0 0 1: Division by 2 mode 0 1 0: Division by 4 mode 0 1 1: Division by 8 mode 1 0 0: Division by 16 mode	○	○
CCLK5			○	○
CCLK6			○	○
CCLK7	Reserved bit	Always set to "0"	○	○

Note1: Set bit 0 of the protect register (address 000A$_{16}$) to "1" before writing in this register.
Note2: Change the register value only when the CAN module is in Reset/Initialization mode (the bit 0 of the CAN Control Register (address 0210$_{16}$ and 0230$_{16}$) is"1").

Figure 5-5. CAN0/1 clock select register

Peripheral function clock select register (Note 1)

b7 b6 b5 b4 b3 b2 b1 b0

Symbol	Address	When reset
PCLKR	025E$_{16}$	XXXXXX00$_{16}$

Bit symbol	Bit name	Function	R	W
PCLK0	TimerA, TimerB, A-D converter function clock	0: division by 2 mode 1: division by 1 mode (Note 2)	○	○
PCLK1	UART0-2, SIO3 function clock	0: division by 2 mode 1: division by 1 mode (Note 2)	○	○
Nothing is assigned. These bits can neither be set nor reset. When read, their contents are "0".			○	–

Note1: Set bit 0 of the protect register (address 000A$_{16}$) to "1" before writing in this register.
Note 2: Do not set "1" when X$_{IN}$ is more than 16 MHz

Figure 5-6. Peripheral function clock select register

Clock Output

In single-chip mode, the clock output function select bits (bits 0 and 1 at address 0006_{16}) enable f_8, f_{32}, or fc to be output from the $P5_7$/CLK$_{OUT}$ pin. When the WAIT peripheral function clock stop bit (bit 2 at address 0006_{16}) is set to “1”, the output of f_8 and f_{32} stops when a WAIT instruction is executed.

Stop Mode

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

Because the oscillation of BCLK, f_2 to f_{32}, fc, fc_{32}, f_{CAN0}, f_{CAN1} and f_{AD2} stops in stop mode, peripheral functions such as the A-D converter and watchdog timer do not function. However, timer A and timer B operate provided that the event counter mode is set to an external pulse, and UARTi(i = 0 to 2) functions provided an external clock is selected. Table 5-2 shows the status of the ports in stop mode.

Stop mode is cancelled by a hardware reset or interrupt. If an interrupt is to be used to cancel stop mode, that interrupt must first have been enabled.

When shifting to stop mode, the main clock division select bit 0 (bit 6 at 0006_{16}) is set to “1”.

Table 5-2. Port status during stop mode

Pin		Memory expansion mode Microprocessor mode	Single-chip mode
Address bus, data bus, $\overline{CS0}$ to $\overline{CS3}$		Retains status before stop mode	
$\overline{RD}$, $\overline{WR}$, $\overline{BHE}$, $\overline{WRL}$, $\overline{WRH}$		“H”	
$\overline{HLDA}$, BCLK		“H”	
ALE		“H”	
Port		Retains status before stop mode	Retains status before stop mode
CLK$_{OUT}$	When fc selected	Valid only in single-chip mode	“H”
	When f_8, f_{32} selected	Valid only in single-chip mode	Retains status before stop mode

Wait Mode

When a WAIT instruction is executed, the BCLK stops and the microcomputer enters the wait mode. In this mode, oscillation continues but the BCLK and watchdog timer may be stopped under certain conditions. Refer to the section describing the watchdog timer. Writing “1” to the WAIT peripheral function clock stop bit and executing a WAIT instruction stops the clock being supplied to the internal peripheral functions, allowing power consumption to be reduced. Table 5-3 shows the status of the ports in wait mode.

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

Table 5-3. Port status during wait mode

Pin		Memory expansion mode Microprocessor mode	Single-chip mode
Address bus, data bus, $\overline{\text{CS0}}$ to $\overline{\text{CS3}}$		Retains status before wait mode	
$\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{BHE}}$, $\overline{\text{WRL}}$, $\overline{\text{WRH}}$		“H”	
$\overline{\text{HLDA}}$		“H”	
BCLK		“H” (Note)	
ALE		“H”	
Port		Retains status before wait mode	Retains status before wait mode
CLK_{OUT}	When f_C selected	Valid only in single-chip mode	Does not stop
	When f_8, f_{32} selected	Valid only in single-chip mode	Does not stop when the WAIT peripheral function clock stop bit is “0”. When the WAIT peripheral function clock stop bit is “1”, the status immediately prior to entering wait mode is maintained.

Note: BCLK is "H" only when the watchdog timer is stopped. Refer to the watchdog timer section for more information

Status Transition Of BCLK

Power consumption can be reduced and low-voltage operation achieved by changing the count source for BCLK. Table 5-4 shows the operating modes corresponding to the settings of system clock control registers 0 and 1.

After a reset, operation defaults to division by 8 mode. When shifting to stop mode, the main clock division select bit 0 (bit 6 at address 0006_{16}) is set to “1”. The following shows the operational modes of BCLK.

(1) Division by 2 mode

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

(2) Division by 4 mode

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

(3) Division by 8 mode

The main clock is divided by 8 to obtain the BCLK. Note that oscillation of the main clock must have stabilized before transferring from this mode to another mode.

(4) Division by 16 mode

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

(5) No-division mode

The main clock is used as BCLK.

(6) Low-speed mode

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

(7) Low power consumption mode

fC is the BCLK and the main clock is stopped.

(8) Ring oscillator mode

What the ring oscillator generates is the BCLK. You can use it by dividing it by 2, 4, 8 or 16, and also no division is possible.

Table 5-4. Operating modes dictated by settings of system clock control registers 0 and 1

CM17	CM16	CM07	CM06	CM05	CM04	Operating mode of BCLK
0	1	0	0	0	Invalid	Division by 2 mode
1	0	0	0	0	Invalid	Division by 4 mode
Invalid	Invalid	0	1	0	Invalid	Division by 8 mode
1	1	0	0	0	Invalid	Division by 16 mode
0	0	0	0	0	Invalid	No-division mode
Invalid	Invalid	1	Invalid	0	1	Low-speed mode
Invalid	Invalid	1	Invalid	1	1	Low power consumption mode

Power Control

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

Modes

Power control is available in three modes.

(1) Normal operation mode

- **High-speed mode**

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

- **Medium-speed mode**

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

- **Low-speed mode**

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

- **Low power consumption mode**

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

- **Ring oscillator mode**

  The ring oscillator replaces X_{IN}. No-division-, divide-by-2-, 4-, 8- or 16 mode can be selected by changing the values in CM06, CM16 and CM17. The higher the division ratio is, the lower power consumption. The clock driver of X_{IN} can be stopped by changing the value of the main clock stop bit to "0" when the CPU operates using the ring oscillator. Through this the power consumption will be still lower.

(2) Wait mode

The CPU operation is stopped. The oscillator does not stop.

(3) Stop mode

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

Figure 5-7 shows the state transition of power control modes.

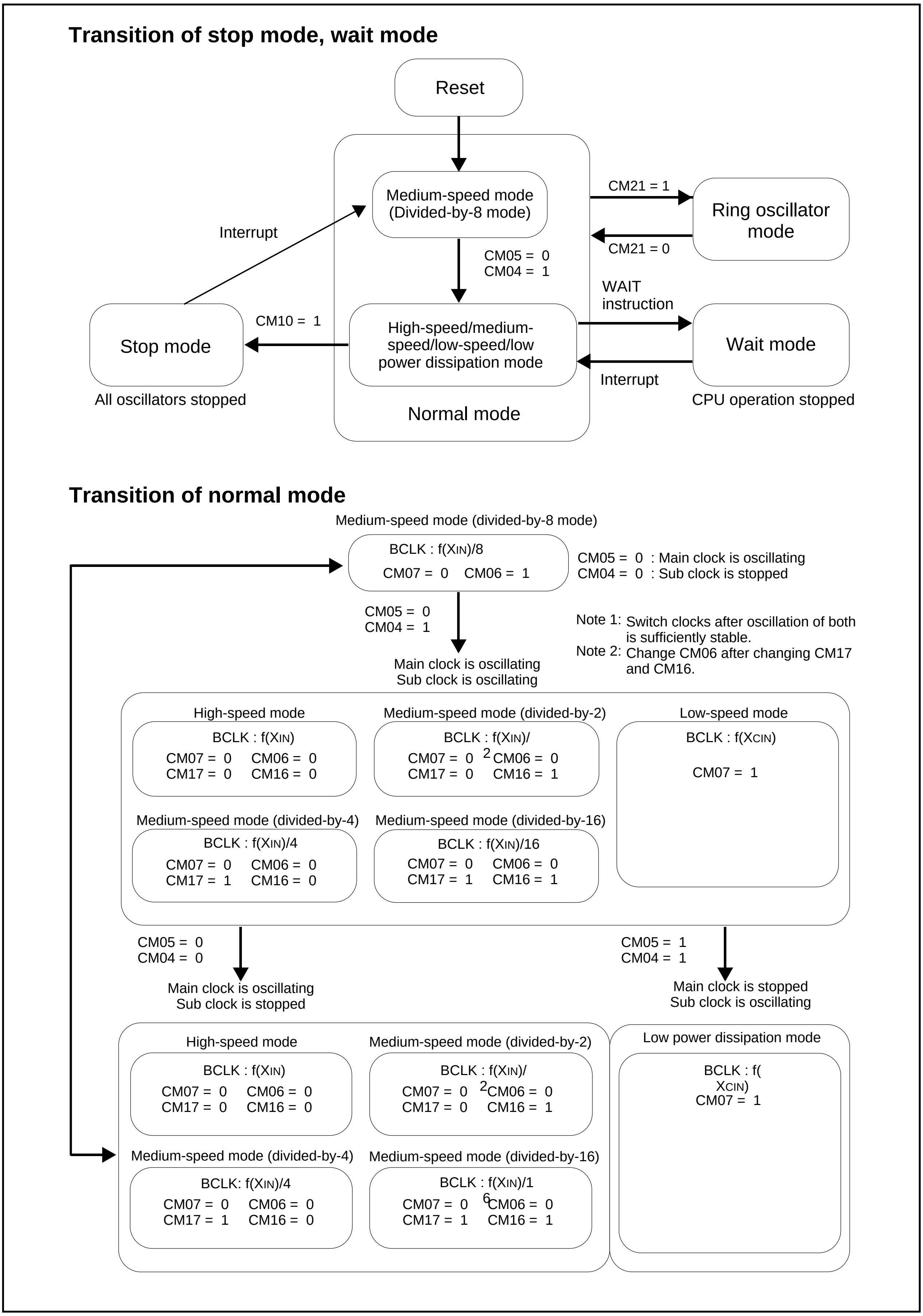


Figure 5-7. State transition diagram of power control mode

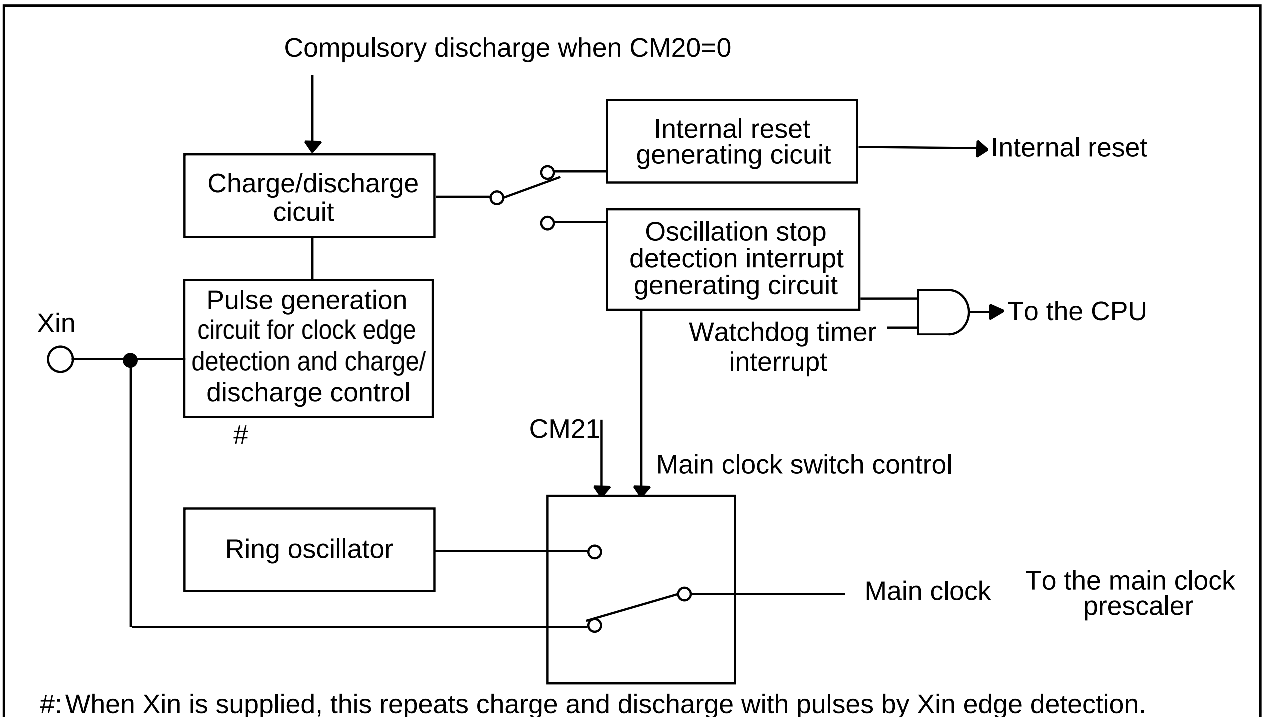
Oscillation Stop Detection Function

This function is for detecting an abnormal stop of the clock which is caused by open- and/or short circuit of the Xin oscillation circuit. When it detects an oscillation stop, it generates either an internal reset or an oscillation stop detection interrupt. The selection depends on the value in the bit 7 of the oscillation stop detection register (000C$_{16}$). When an oscillation stop detection interrupt is generated, the ring oscillator which is built in the microcomputer starts oscillation automatically, which is used as the system clock instead of Xin. Through this an interrupt operation is enabled.

You can set the function to valid/invalid by changing the value in the bit 0 of the oscillation stop detection register. The function is valid when the bit is "1". However, the value of the bit after reset release is "0", so the function is invalid.

Table 5-5. Outline of specification of the oscillation stop detection function

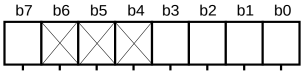
Item	Specification
Clock and Frequency	Xin is 2 Mhz or more.
Condition	The oscillation stop detection bit (bit 0 at 000C$_{16}$) is "1".
Operation when detected an oscillation stop	#Generates an internal reset (when the bit 7 at 000C$_{16}$ is "0") #Generates an oscillation stop detection interrupt (when the bit 7 at 000C$_{16}$ is "1")
In the stop-mode	Write "0" in the oscillation stop detection bit before setup of the stop-mode to set the oscillation stop detection function to "invalid". Write "1" in the bit after stop-mode release.


Compulsory discharge when CM20=0
Internal reset generating cicuit
Internal reset
Charge/discharge cicuit
Oscillation stop detection interrupt generating circuit
To the CPU
Watchdog timer interrupt
Xin
Pulse generation circuit for clock edge detection and charge/ discharge control
#
CM21
Main clock switch control
Ring oscillator
Main clock
To the main clock prescaler

#: When Xin is supplied, this repeats charge and discharge with pulses by Xin edge detection. When Xin is not supplied, this continues charging. When the charge exceeds a certain level, it regards the oscillation as stopped.

Figure 5-8. Structure of the oscillation stop detection circuit

Oscillation stop detection register (Note 1)



Symbol	Address	When reset
CM2	$000C_{16}$	00_{16}

Bit symbol	Bit name	Function	R	W
CM20	Oscillation stop detection bit	0: The function is invalid. 1: The function is valid.	○	○
CM21	Main clock switch bit	0: Select Xin (Ring oscillator is stopped.) 1: Select ring oscillator.	○	○
CM22	Oscillation stop detection status (Note 2)	0: No meaning 1: An oscillation stop is detected.	○	○
CM23	Clock monitor bit (Note 3)	0: Xin is in operation. 1: Xin is stopped.	○	×
CM24	Reserved bit	Always set to "0"	—	—
CM25				
CM26				
CM27	Operation select bit (when an oscillation stop is detected)	0: Internal reset on stop detectio 1: Start ring oscillator	○	○

Note 1: Set bit 0 of the protect register (address $000A_{16}$) to "1" before writing to this register.
Note 2: This bit is valid only in an execution program for the oscillation stop detection interrupt. Use this bit for the purpose of cause judgment(oscillation stop detection- or watchdog timer interrupt) for interrupt execution. You can write in this bit "0" only.
Note 3: This bit is valid only in an execution program for the oscillation stop detection interrupt. Use this bit for the purpose of confirming Xin operation for interrupt execution.

Figure 5-9. Structure of the oscillation stop detection register

Oscillation stop detection bit (CM20)

You can start the oscillation stop detection by setting this bit to "1". The detection is not executed when this bit is set to "0" or in reset status. Be sure to set this bit to "0" before setting for the stop-mode. Set this bit again to "1" after release from stop-mode. This is because it is necessary to cancel the oscillation stop detection function due to a certain period of unstable oscillation after release from stop-mode. Set this bit to "0" also before setting the main clock stop bit (bit 5 at 0006_{16}) to "1".
Do not set this bit to "1" if the frequency of Xin is lower than 2 MHz.

Main clock switch bit (CM21)

You can use the ring oscillator as a system clock by setting this bit to "1". When this bit is "0", the ring oscillator is not in operation. For more explanation, see the section of the clock generating circuit.

Oscillation stop detection status (CM22)

You can see the status of the oscillation stop detection. When this bit is "1", an oscillation stop is detected. For usage of this bit, see the explanation on CM27.

Clock monitor bit (CM23)

You can see the operation status of the Xin clock. When this bit is "1", Xin is operating correctly. You can check the operation status of Xin when an oscillation stop detection interrupt is generated.

Operation select (when an oscillation stop is detected) bit (CM27)

(1) Operation when internal reset is selected (CM27 is set to "0".)

An internal reset is generated when an abnormal stop of Xin is detected. The microcomputer stops in reset status and does not operate further.

Note: Release from this status is only possible through an external reset. However, in case of a defect Xin clock, further operation cannot be compensated.

See Table 5-6 for status of each port after an internal reset is generated.

(2) Operation when oscillation stop detection interrupt is selected (CM27 is set to "1".)

An oscillation stop detection interrupt is generated when an abnormal stop of Xin is detected. The ring oscillator starts operation instead of the Xin clock which stopped abnormally. The operation goes further with the supply from the ring oscillator. For the oscillation stop detection interrupt judgment on the interrupt condition is necessary, because this interrupt shares the vector table with watchdog timer interrupt. Use the oscillation stop detection status (CM22) for the judgment. Figure 5-10 shows the flow of the judgment.

Table 5-6. Port status after an internal reset is generated

Pin name	Pin Status		
	Single-chip mode	Microprocessor mode/Memory expansion mode	
		BYTE = VSS	BYTE = Vcc
P0	Input port (floating)	Data input (floating)	Data input (floating)
P1	Input port (floating)	Data input (floating)	Input port (floating)
P2, P3, $P4_0$ to $P4_3$	Input port (floating)	Address output (undefined)	Address output (undefined)
$P4_4$	Input port (floating)	$\overline{CS0}$ output ("H" level output)	$\overline{CS0}$ output ("H" level output)
$P4_5$ to $P4_7$	Input port (floating)	Input port (floating) (Pull-up resistance is ON.)	Input port (floating) (Pull-up resistance is ON.)
$P5_0$	Input port (floating)	$\overline{WR}$ output ("H" level output)	$\overline{WR}$ output ("H" level output)
$P5_1$	Input port (floating)	$\overline{BHE}$ output (undefined)	$\overline{BHE}$ output (undefined)
$P5_2$	Input port (floating)	$\overline{RD}$ output ("H" level output)	$\overline{RD}$ output ("H" level output)
$P5_3$	Input port (floating)	BCLK output	BCLK output
$P5_4$	Input port (floating)	$\overline{HLDA}$ output (Output value depends on $\overline{HOLD}$ pin input.)	$\overline{HLDA}$ output (Output value depends on $\overline{HOLD}$ pin put.)
$P5_5$	Input port (floating)	$\overline{HOLD}$ input (floating)	$\overline{HOLD}$ input (floating)
$P5_6$	Input port (floating)	ALE output ("L" level output)	ALE output ("L" level output)
$P5_7$	Input port (floating)	$\overline{RDY}$ input (floating)	$\overline{RDY}$ input (floating)
P6, P7, $P8_0$ to $P8_4$ $P8_6$, $P8_7$, P9, P10	Input port (floating)	Input port (floating)	Input port (floating)

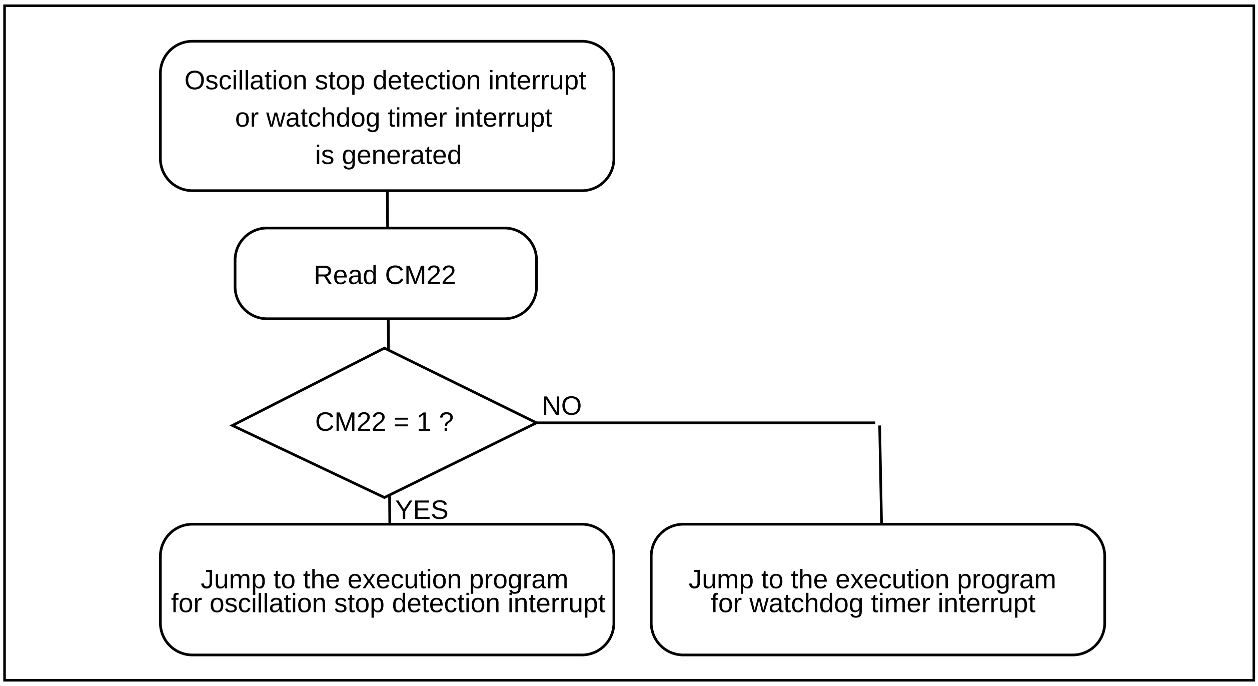
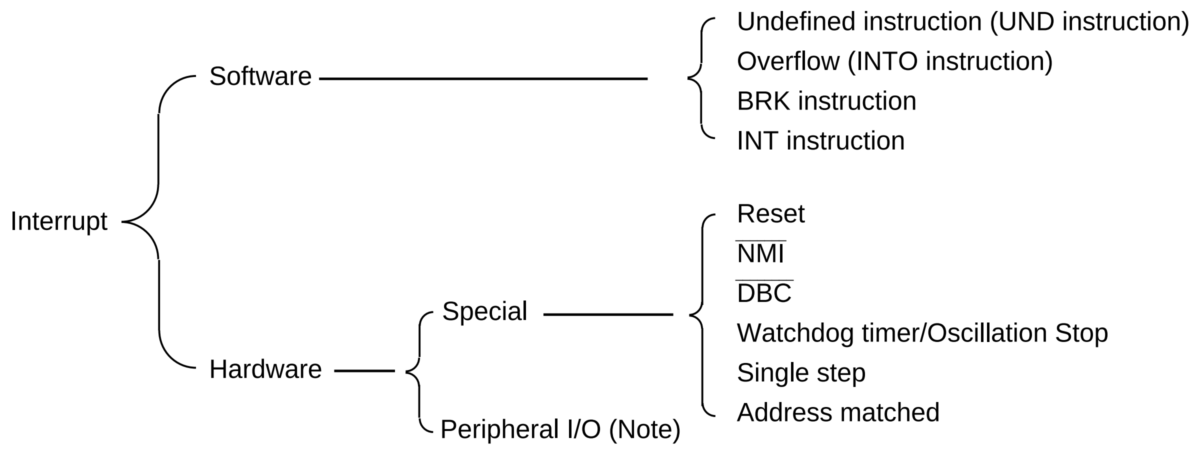

Oscillation stop detection interrupt or watchdog timer interrupt is generated
Read CM22
CM22 = 1 ?
NO
YES
Jump to the execution program for oscillation stop detection interrupt
Jump to the execution program for watchdog timer interrupt

Figure 5-10. Flow of the judgment

Overview of Interrupt

Type of Interrupts

Figure 6-1 lists the types of interrupts.



Note: Peripheral I/O interrupts are generated by the peripheral functions built into the microcomputer system.

Figure 6-1. Classification of interrupts

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

Software Interrupts

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

- **Undefined instruction interrupt**

  An undefined instruction interrupt occurs when executing the UND instruction.

- **Overflow interrupt**

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

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

- **BRK interrupt**

  A BRK interrupt occurs when executing the BRK instruction.

- **INT interrupt**

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

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

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

Hardware Interrupts

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

(1) Special interrupts

Special interrupts are non-maskable interrupts.

- **Reset**
  Reset occurs if an "L" is input to the $\overline{\text{RESET}}$ pin.
- **$\overline{\text{NMI}}$ interrupt**
  An $\overline{\text{NMI}}$ interrupt occurs if an "L" is input to the $\overline{\text{NMI}}$ pin.
- **$\overline{\text{DBC}}$ interrupt**
  This interrupt is exclusively for the debugger, do not use it in other circumstances.
- **Watchdog timer interrupt/Oscillation stop detection interrupt**
  Generated by the watchdog timer or upon oscillation stop detection.
- **Single-step interrupt**
  This interrupt is exclusively for the debugger, do not use it in other circumstances. With the debug flag (D flag) set to "1", a single-step interrupt occurs after one instruction is executed.
- **Address match interrupt**
  An address match interrupt occurs immediately before the instruction held in the address indicated by the address match interrupt register is executed with the address match interrupt enable bit set to "1". If an address other than the first address of the instruction in the address match interrupt register is no address match interrupt occurs. For address match interrupt, see 2. 11 Address match interrupt.

(2) Peripheral I/O interrupts

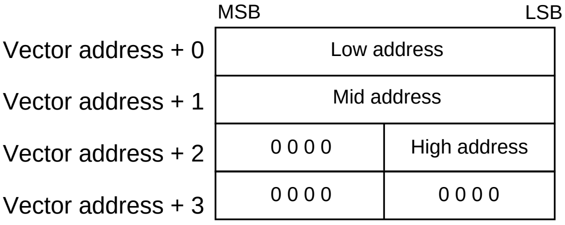
A peripheral I/O interrupt is generated by one of built-in peripheral functions. Built-in peripheral functions are dependent on classes of products, so the interrupt factors too are dependent on classes of products. The interrupt vector table is the same as the one for software interrupt numbers 0 through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

- **Bus collision detection interrupt**
  This is an interrupt that the serial I/O bus collision detection generates.
- **DMA0 interrupt, DMA1 interrupt**
  These are interrupts that DMA generates.
- **Key-input interrupt**
  A key-input interrupt occurs if an "L" is input to the $\overline{\text{KI}}$ pin.
- **A-D conversion interrupt**
  This is an interrupt that the A-D converter generates.
- **UART0, UART1, UART2/NACK, CAN0, CAN1, SI/O3, and SI/O4 transmission interrupt**
  These are interrupts that the serial I/O transmission generates.
- **UART0, UART1, UART2/ACK, CAN0, CAN1, SI/O3, and SI/O4 reception interrupt**
  These are interrupts that the serial I/O reception generates.
- **Timer A0 interrupt through timer A4 interrupt**
  These are interrupts that timer A generates.
- **Timer B0 interrupt through timer B5 interrupt**
  These are interrupts that timer B generates.
- **$\overline{\text{INT0}}$ interrupt through timer $\overline{\text{INT5}}$ interrupt**
  An $\overline{\text{INT}}$ interrupt occurs if either a rising edge or a falling edge or both edges are input to the $\overline{\text{INT}}$ pin.

Interrupts and Interrupt Vector Tables

If an interrupt request is accepted, a program branches to the interrupt routine set in the interrupt vector table. Set the first address of the interrupt routine in each vector table. Figure 6. 2 shows the format for specifying the address.

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



	MSB	LSB
Vector address + 0	Low address	
Vector address + 1	Mid address	
Vector address + 2	0 0 0 0	High address
Vector address + 3	0 0 0 0	0 0 0 0

Figure 6-2. Format for specifying interrupt vector addresses

- **Fixed vector tables**

The fixed vector table is a table in which addresses are fixed. The vector tables are located in an area extending from FFFDC$_{16}$ to FFFFF$_{16}$. One vector table comprises four bytes. Set the first address of interrupt routine in each vector table. Table 6. 1 shows the interrupts assigned to the fixed vector tables and addresses of vector tables.

Table 6-1. Interrupts assigned to the fixed vector tables and addresses of vector tables

Interrupt source	Vector table addresses Address (L) to address (H)	Remarks
Undefined instruction	FFFDC$_{16}$ to FFFDF$_{16}$	Interrupt on UND instruction
Overflow	FFFE0$_{16}$ to FFFE3$_{16}$	Interrupt on INTO instruction
BRK instruction	FFFE4$_{16}$ to FFFE7$_{16}$	If the vector contains FF$_{16}$, program execution starts from the address shown by the vector in the variable vector table
Address match	FFFE8$_{16}$ to FFFEB$_{16}$	There is an address-matching interrupt enable bit
Single step (Note)	FFFEC$_{16}$ to FFFEF$_{16}$	Do not use
Watchdog timer Oscillation stop detection	FFFF0$_{16}$ to FFFF3$_{16}$	
$\overline{\text{DBC}}$ (Note)	FFFF4$_{16}$ to FFFF7$_{16}$	Do not use
NMI	FFFF8$_{16}$ to FFFFB$_{16}$	External interrupt by input to $\overline{\text{NMI}}$ pin
Reset	FFFFC$_{16}$ to FFFFF$_{16}$	

Note: Interrupts used for debugging purposes only.

- **Variable vector tables**

The addresses in the variable vector table can be modified, according to the user's setting. Indicate the first address using the interrupt table register (INTB). The 256-byte area subsequent to the address the INTB indicates becomes the area for the variable vector tables. One vector table comprises four bytes. Set the first address of the interrupt routine in each vector table. Table 6-2 shows the interrupts assigned to the variable vector tables and addresses of vector tables.

Table 6-2. Interrupt assigned to the variable vector tables and addresses of vector tables

Software interrupt number	Vector table address Address (L) to address (H)	Interrupt source	Remarks
Software interrupt number 0	+0 to +3 (Note 1)	BRK instr.	
Software interrupt number 1	+4 to +7 (Note 1)	CAN0,1 Wake Up	
Software interrupt number 2	+8 to +11 (Note 1)	CAN0 reception	
Software interrupt number 3	+12 to +15 (Note 1)	CAN0 transmission	
Software interrupt number 4	+16 to +19 (Note 1)	$\overline{\text{INT3}}$	
Software interrupt number 5	+20 to +23 (Note 1)	Timer B5	
Software interrupt number 6	+24 to +27 (Note 1)	Timer B4	
Software interrupt number 7	+28 to +31 (Note 1)	Timer B3	
Software interrupt number 8	+32 to +35 (Note 1,2)	CAN1 reception, $\overline{\text{INT5}}$	
Software interrupt number 9	+36 to +39 (Note 1,2)	CAN1 transm., $\overline{\text{INT4}}$, S I/O3	
Software interrupt number 10	+40 to +43 (Note 1)	Bus collision detection	
Software interrupt number 11	+44 to +47 (Note 1)	DMA0	
Software interrupt number 12	+48 to +51 (Note 1)	DMA1	
Software interrupt number 13	+52 to +55 (Note 1)	CAN0,1 Error int.	
Software interrupt number 14	+56 to +59 (Note 1,2)	A-D Conv., Key input int.	
Software interrupt number 15	+60 to +63 (Note 1,3)	UART2 transmission	
Software interrupt number 16	+64 to +67 (Note 1,3)	UART2 reception	
Software interrupt number 17	+68 to +71 (Note 1)	UART0 transmission	
Software interrupt number 18	+72 to +75 (Note 1)	UART0 reception	
Software interrupt number 19	+76 to +79 (Note 1)	UART1 transmission	
Software interrupt number 20	+80 to +83 (Note 1)	UART1 reception	
Software interrupt number 21	+84 to +87 (Note 1)	Timer A0	
Software interrupt number 22	+88 to +91 (Note 1)	Timer A1	
Software interrupt number 23	+92 to +95 (Note 1)	Timer A2	
Software interrupt number 24	+96 to +99 (Note 1)	Timer A3	
Software interrupt number 25	+100 to +103 (Note 1)	Timer A4	
Software interrupt number 26	+104 to +107 (Note 1)	Timer B0	
Software interrupt number 27	+108 to +111 (Note 1)	Timer B1	
Software interrupt number 28	+112 to +115 (Note 1)	Timer B2	
Software interrupt number 29	+116 to +119 (Note 1)	$\overline{\text{INT0}}$	
Software interrupt number 30	+120 to +123 (Note 1)	$\overline{\text{INT1}}$	
Software interrupt number 31	+124 to +127 (Note 1)	$\overline{\text{INT2}}$	
Software interrupt number 32 to Software interrupt number 63	+128 to +131 (Note 1) to +252 to +255 (Note 1)	Software interrupt	Cannot be masked I flag

Note 1: Address relative to address in interrupt table register (INTB).
Note 2: It is selected by interrupt request cause select registers (IFSR0/1).
Note 3: When IIC mode is selected, NACK and ACK interrupts are selected.

Interrupt Control

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

Enable or disable a maskable interrupt using the interrupt enable flag (I flag), interrupt priority selection bit, or processor interrupt priority level(IPL). Whethre an interrupt request is present or absent is indicated by the interrupt request bit. The interrupt request bit and the interrupt priority level selection bie are located in the interrupt control register of each interrupt. Also, the interrupt enable flag (I flag) and the IPL are located in the flag register (FLG).

Figure 6-3 shows the memory map of the interrupt control registers.

Interrupt control register (Note 2)

Symbol	Address	When reset
C01WKUPIC,	0041$_{16}$	XXXXX000$_2$
C0RECIC, C0TRMIC	0042$_{16}$, 0043$_{16}$	XXXXX000$_2$
TBiIC(i=3 to 5)	0045$_{16}$ to 0047$_{16}$	XXXXX000$_2$
BCNIC	004A$_{16}$	XXXXX000$_2$
DMiIC(i=0, 1)	004B$_{16}$, 004C$_{16}$	XXXXX000$_2$
C01ERRIC, KUPIC	004D$_{16}$, 004E$_{16}$	XXXXX000$_2$
ADIC	004E$_{16}$	XXXXX000$_2$
SiTIC(i=0 to 2)	0051$_{16}$, 0053$_{16}$, 004F$_{16}$	XXXXX000$_2$
SiRIC(i=0 to 2)	0052$_{16}$, 0054$_{16}$, 0050$_{16}$	XXXXX000$_2$
TAiIC(i=0 to 4)	0055$_{16}$ to 0059$_{16}$	XXXXX000$_2$
TBiIC(i=0 to 2)	005A$_{16}$ to 005C$_{16}$	XXXXX000$_2$

b7 b6 b5 b4 b3 b2 b1 b0
☒ ☒ ☒ ☒ □ □ □ □

Bit symbol	Bit name	Function	R	W
ILVL0	Interrupt priority level select bit	b2 b1 b0 0 0 0 : Level 0 (interrupt disabled) 0 0 1 : Level 1 0 1 0 : Level 2 0 1 1 : Level 3 1 0 0 : Level 4 1 0 1 : Level 5 1 1 0 : Level 6 1 1 1 : Level 7	○	○
ILVL1			○	○
ILVL2			○	○
IR	Interrupt request bit	0 : Interrupt not requested 1 : Interrupt requested	○	○ (Note1)
Nothing is assigned. In an attempt to write to these bits, write 0 . The value, if read, turns out to be 0 .			–	–

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

Symbol		Address	When reset
INTiIC(i=3)		0044$_{16}$	XX00X000$_2$
C1RECIC/INT5IC	(Note 3)	0048$_{16}$	XX00X000$_2$
C1TRMIC/S3IC/INT4IC	(Note 3)	0049$_{16}$	XX00X000$_2$
INTiIC(i=0 to 2)	005D$_{16}$	to 005F$_{16}$	XX00X000$_2$

b7 b6 b5 b4 b3 b2 b1 b0
☒ ☒ 0 □ □ □ □ □

Bit symbol	Bit name	Function	R	W
ILVL0	Interrupt priority level select bit	b2 b1 b0 0 0 0 : Level 0 (interrupt disabled) 0 0 1 : Level 1 0 1 0 : Level 2 0 1 1 : Level 3 1 0 0 : Level 4 1 0 1 : Level 5 1 1 0 : Level 6 1 1 1 : Level 7	○	○
ILVL1			○	○
ILVL2			○	○
IR	Interrupt request bit	0: Interrupt not requested 1: Interrupt requested	○	○ (Note1)
POL	Polarity select bit	0 : Selects falling edge 1 : Selects rising edge	○	○
Reserved bit		Always set to 0	○	○
Nothing is assigned. In an attempt to write to these bits, write 0 . The value, if read, turns out to be 0 .			–	–

Note 1: This bit can only be accessed for reset (= 0), but cannot be accessed for set (= 1).
Note 2: To rewrite the interrupt control register, do so at a point that dose not generate the interrupt request for that register. For details, see the precautions for interrupts.
Note 3: Use IFSR0/ISFR1 (address 1DE/1DF) for interrupt request cause selection.

Figure 6-3. Interrupt control registers

Interrupt Enable Flag (I flag)

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

Interrupt Request Bit

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

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

Set the interrupt priority level using the interrupt priority level select bit, which is one of the component bits of the interrupt control register. When an interrupt request occurs, the interrupt priority level is compared with the IPL. The interrupt is enabled only when the priority level of the interrupt is higher than the IPL. Table 6-3 shows the settings of interrupt priority levels and Table 6-4 shows the interrupt levels enabled, according to the consist of the IPL.

The following are conditions under which an interrupt is accepted.

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

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

Table 6-3. Settings of interrupt priority levels

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

Table 6-3. Interrupt levels enabled according to the contents of the IPL

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

Rewrite the interrupt control register

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

Example 1

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

Example 2

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

Example 3

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

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

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

Instructions: AND, OR, BCLR, BSET

Interrupt Sequence

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

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

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

(1) CPU gets the interrupt information (the interrupt number and interrupt request level) by reading address 00000_{16}.

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

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

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

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

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

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

Note: This register cannot be utilized by the user.

Interrupt Response Time

'Interrupt response time' is the period between the instant an interrupt occurs and the instant the first instruction within the interrupt routine has been executed. This time comprises the period from the occurence of an interrupt to the completion of the instruction under execution at that moment (a) and the time required for executing the interrupt sequence (b). Figure 6-4 shows the interrupt responce time.

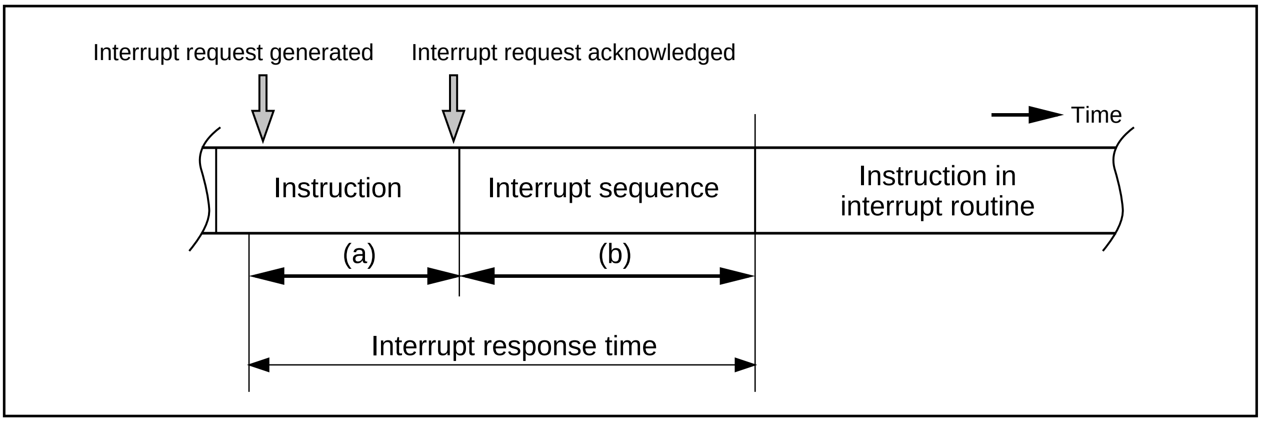

Interrupt request generated
Interrupt request acknowledged
Time
Instruction
Interrupt sequence
Instruction in interrupt routine
(a)
(b)
Interrupt response time

Figure 6-4. Interrupt response time

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

Time (b) is as shown in Table 6-5.

Table 6-5. Time required for executing the interrupt sequence

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

Note 1: Add 2 cycles in the case of a $\overline{\text{DBC}}$ interrupt; add 1 cycle in the case either of an address coincidence interrupt or of a single-step interrupt.

Note 2: Locate an interrupt vector address in an even address, if possible.

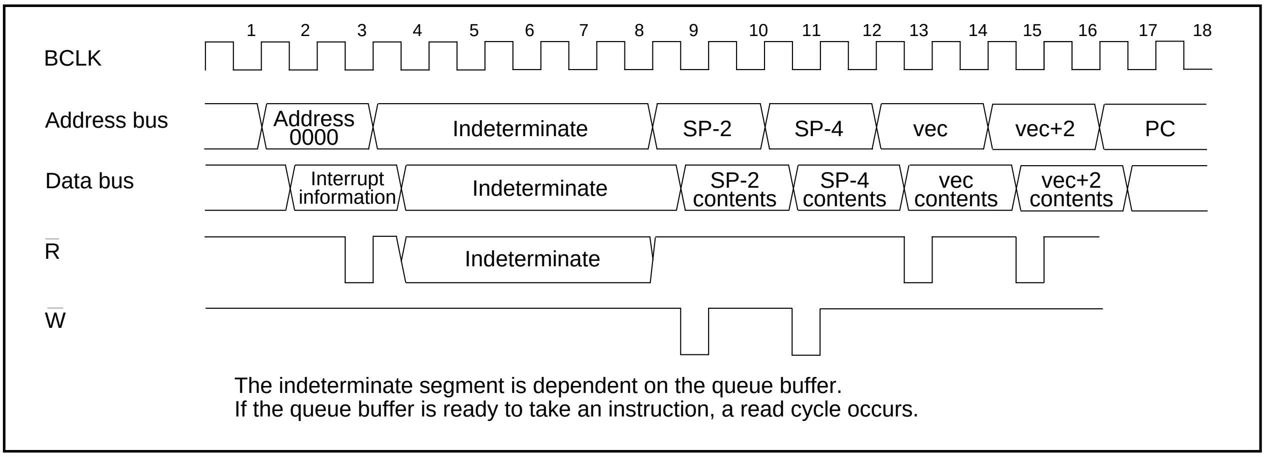


Figure 6-5. Time required for executing the interrupt sequence

Variation of IPL when Interrupt Request is accepted

If an interrupt request is accepted, the interrupt priority level of the accepted interrupt is set in the IPL.

If an interrupt request, that does not have an interrupt priority level, is accepted, one of the values shown in Table 6-6 is set in the IPL.

Table 6-6. Relation between interrupts without interrupt priority levels and IPL

Interrupt sources without priority levels	Value set in the IPL
Watchdog timer, $\overline{\text{NMI}}$	7
Reset	0
Other	Not changed

Saving Registers

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

First, the processor saves the four higher-order bits of the program counter, and 4 upper-order bits and 8 lower-order bits of the FLG register, 16 bits in total, in the stack area, then saves 16 lower-order bits of the program counter. Figure 6-6 shows the state of the stack as it was before the acceptance of the interrupt request, and the state the stack after the acceptance of the interrupt request.

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

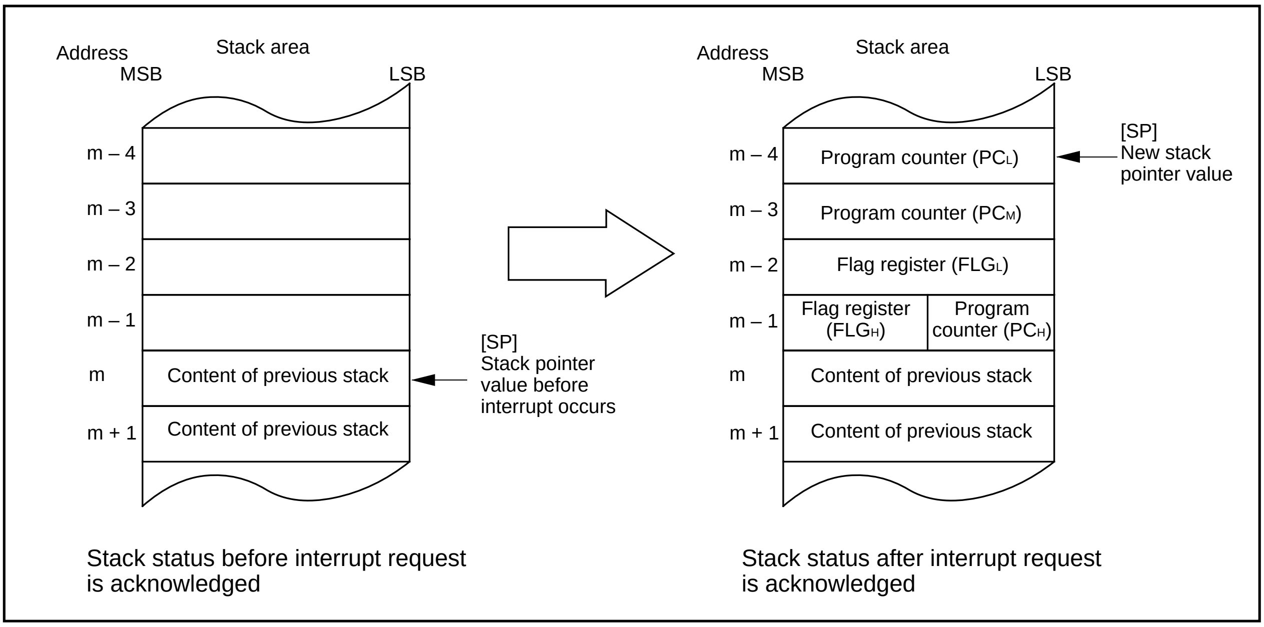
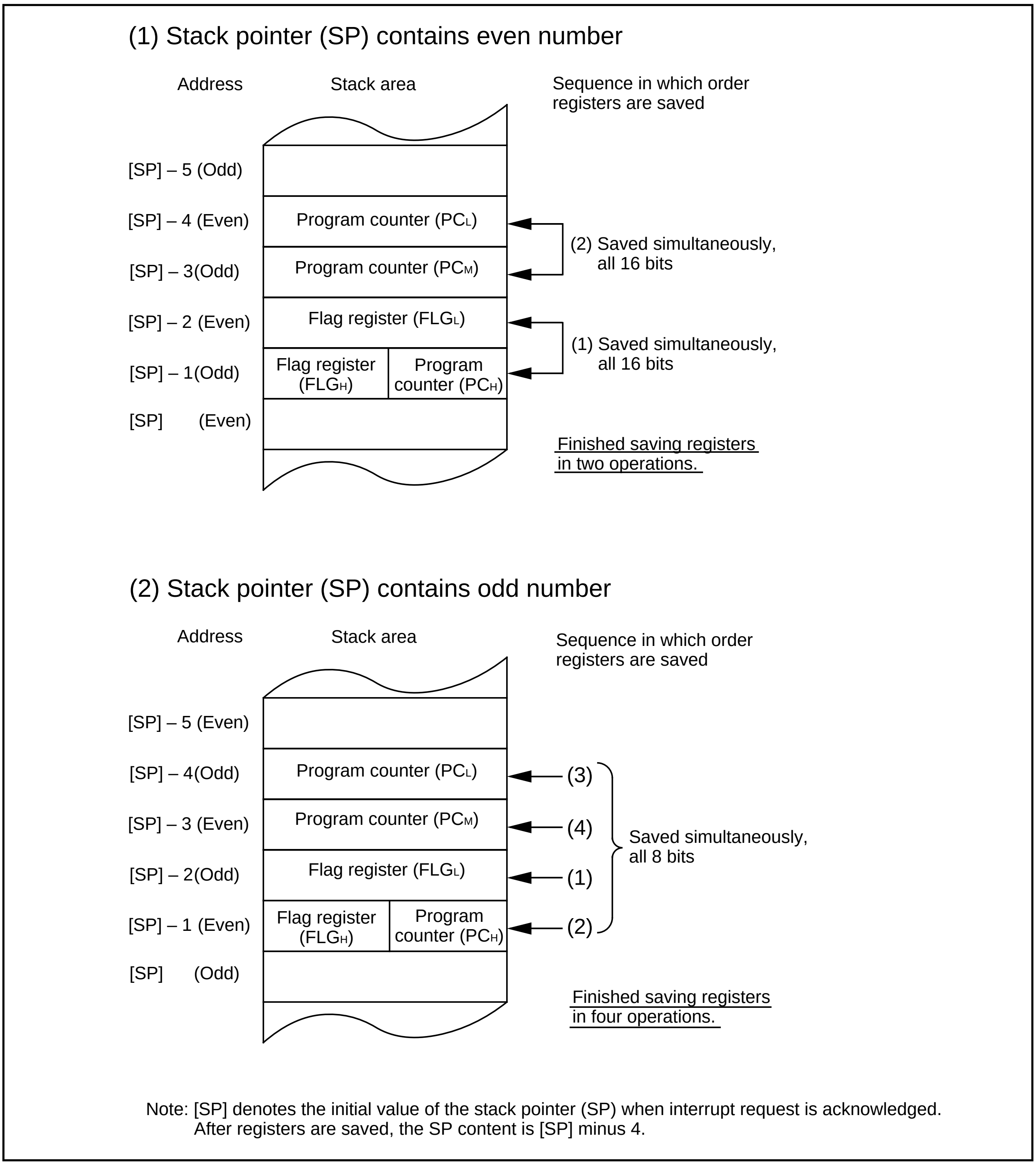


Figure 6-6. State of stack before and after acceptance of interrupt request

The operation of saving registers carried out in the interrupt sequence is dependent whether content of the stack pointer, at the time of acceptance of an interrupt equest, is even or odd. If the counter of the stack pointer (Notze) is even, the counter of the flag register (FLG) and the content of the program counter (PC) are saved, 16 bits at a time. If odd, their contents are saved in two steps, 8 bits at a time. Figure 6-7 shows the operation of the saving registers.

Note: Stack pointer indicated by U flag.



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

Figure 6-7. Operation of saving registers

Returning from an Interrupt Routine

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

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

Interrupt Priority

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

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

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

Figure 6-8 shows the priorities of hardware interrupts.

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

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

Figure 6-8. Hardware interrupts priorities

Interrupt Resolution Circuit

When two or more interupts are generated simultaneously, this circuit selects the interrupt with the highest priority level. Figure 6-9 shows the circuit that judges the interrupt priority level.

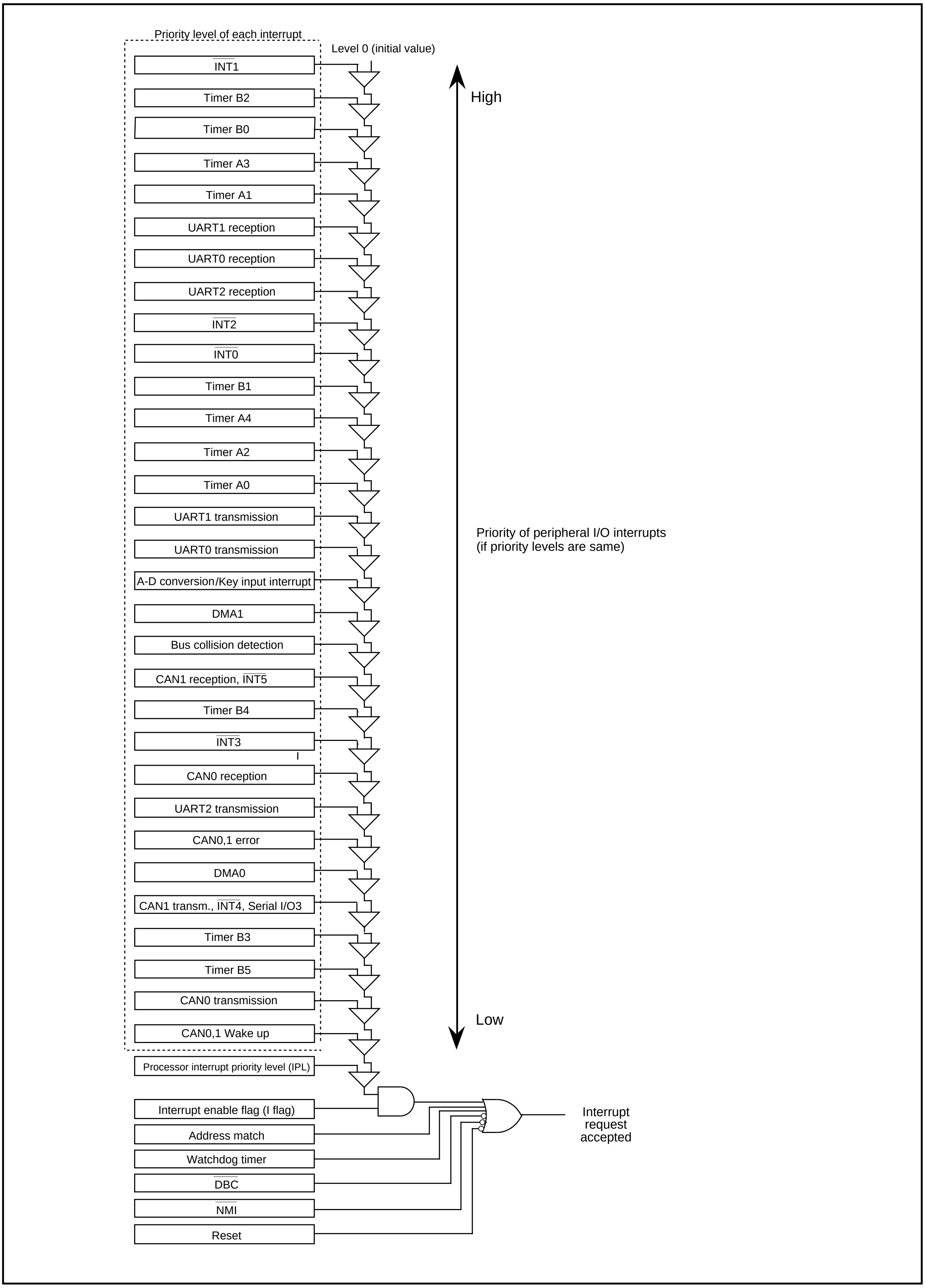


Figure 6-9. Maskable interrupts priorities (peripheral I/O interrupts)

INT Interrupt

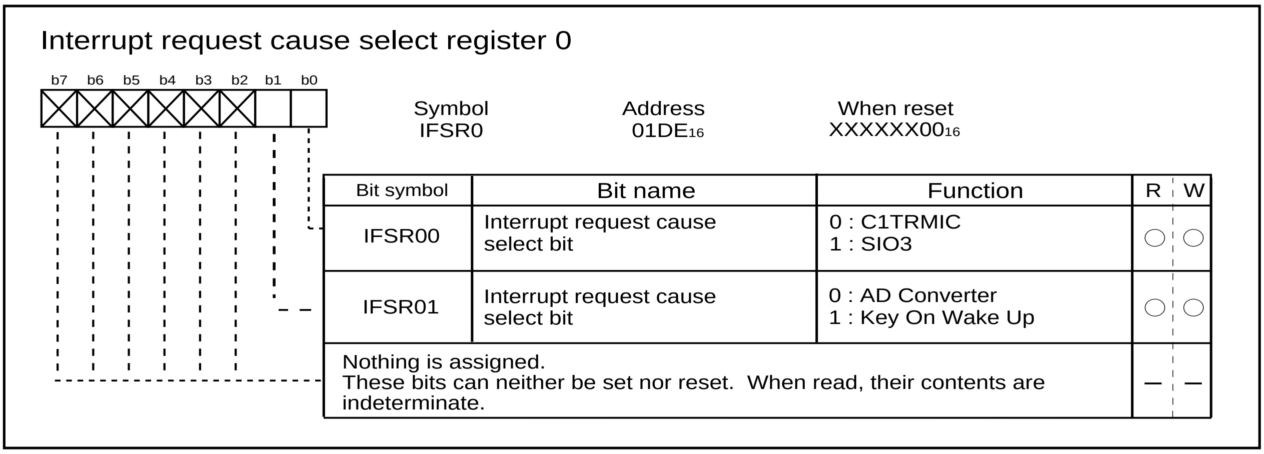
$\overline{\text{INT0}}$ to $\overline{\text{INT5}}$ are triggered by the edges of external inputs. The edge polarity is selected using the polarity select bit.

Of interrupt control registers, 0048_{16} is used both as CAN1 receive and external interrupt $\overline{\text{INT5}}$ input control register, and 0049_{16} is used as serial I/O3, CAN1 transmit and as external interrupt $\overline{\text{INT4}}$ input control register. Use the interrupt request cause select bits - bits 6 and 7 of the interrupt request cause select register ($01DF_{16}$) - to specify which interrupt request cause to select. After having set an interrupt request cause, be sure to clear the corresponding interrupt request bit before enabling an interrupt.

The interrupt control register 0049_{16} has the polarity-switching bit. Be sure to set this bit to "0" when selecting the serial I/O as the interrupt request cause.

As to external interrupt input, an interrupt can be generated both at the rising edge and at the falling edge by setting "1" in the INTi interrupt polarity switching bit of the interrupt request cause select register ($01DF_{16}$). To select both edges, set the polarity switching bit of the correponding interrupt control register to 'falling edge' ("0").

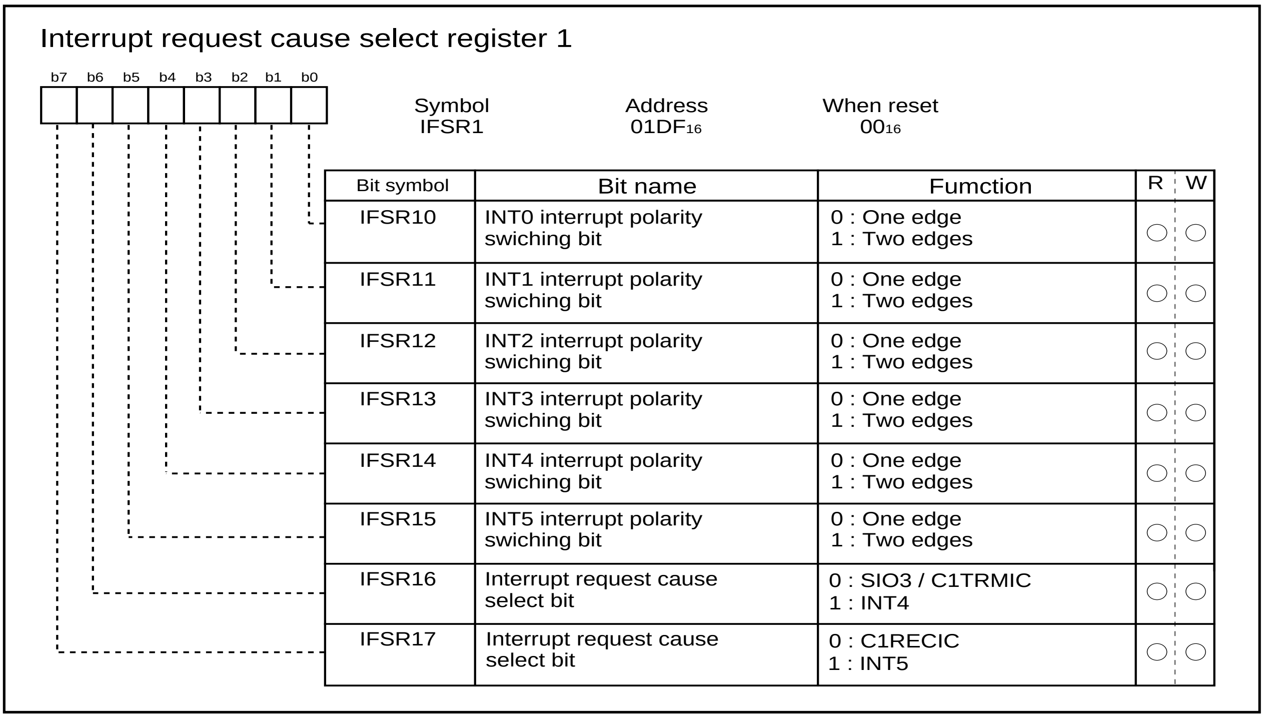
Figures 6-10 and 6-11 show the interrupt request cause select registers 0 and 1.



Symbol	Address	When reset
IFSR0	$01DE_{16}$	$XXXXXX00_{16}$

Bit symbol	Bit name	Function	R	W
IFSR00	Interrupt request cause select bit	0 : C1TRMIC 1 : SIO3	○	○
IFSR01	Interrupt request cause select bit	0 : AD Converter 1 : Key On Wake Up	○	○
Nothing is assigned. These bits can neither be set nor reset. When read, their contents are indeterminate.			—	—

Figure 6-10. Interrupt request cause select register 0



Symbol	Address	When reset
IFSR1	$01DF_{16}$	00_{16}

Bit symbol	Bit name	Fumction	R	W
IFSR10	INT0 interrupt polarity swiching bit	0 : One edge 1 : Two edges	○	○
IFSR11	INT1 interrupt polarity swiching bit	0 : One edge 1 : Two edges	○	○
IFSR12	INT2 interrupt polarity swiching bit	0 : One edge 1 : Two edges	○	○
IFSR13	INT3 interrupt polarity swiching bit	0 : One edge 1 : Two edges	○	○
IFSR14	INT4 interrupt polarity swiching bit	0 : One edge 1 : Two edges	○	○
IFSR15	INT5 interrupt polarity swiching bit	0 : One edge 1 : Two edges	○	○
IFSR16	Interrupt request cause select bit	0 : SIO3 / C1TRMIC 1 : INT4	○	○
IFSR17	Interrupt request cause select bit	0 : C1RECIC 1 : INT5	○	○

Figure 6-11. Interrupt request cause select register 1

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

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

This pin cannot be used as a normal port input.

Key Input Interrupt

If the direction register of any of P10$_4$ to P10$_7$ is set for input and a falling edge is input to that port, a key input interrupt is generated. A key input interrupt can also be used as a key-on wakeup function for cancelling the wait mode or stop mode. However, if you intend to use the key input interrupt, do not use P10$_4$ to P10$_7$ as A-D input ports. Figure 6-12 shows the block diagram of the key input interrupt. Note that if an “L” level is input to any pin that has not been disabled for input, inputs to the other pins are not detected as an interrupt.

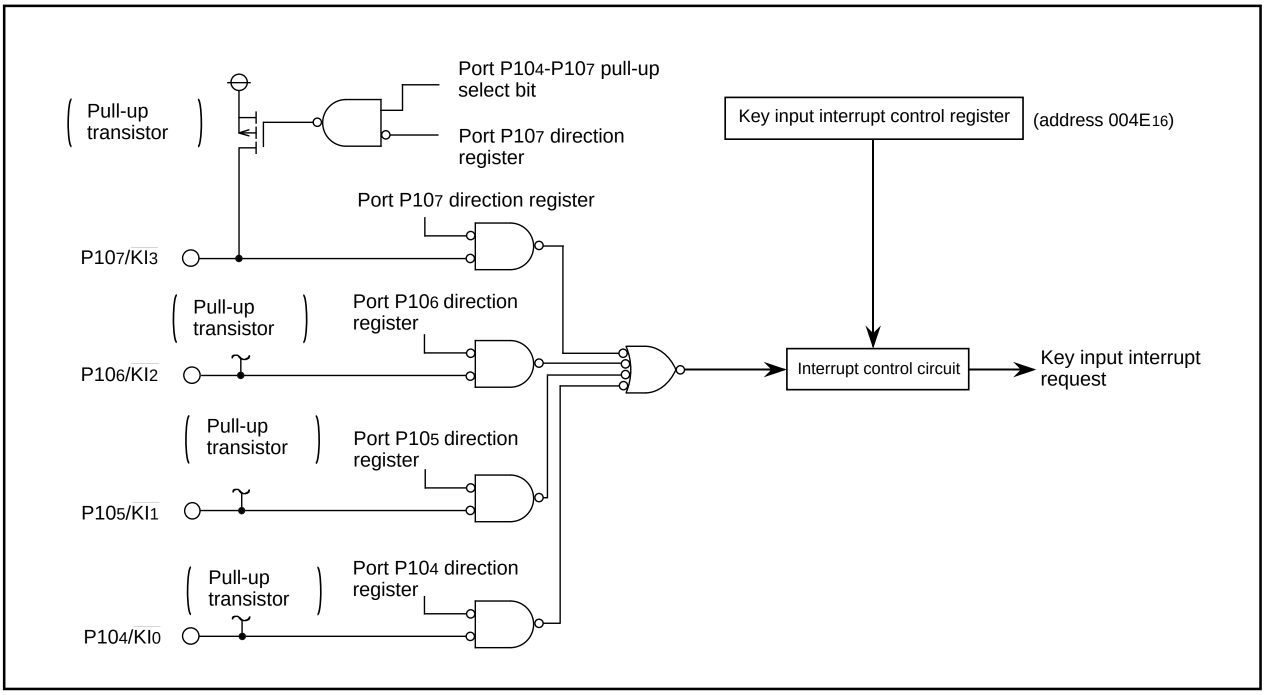
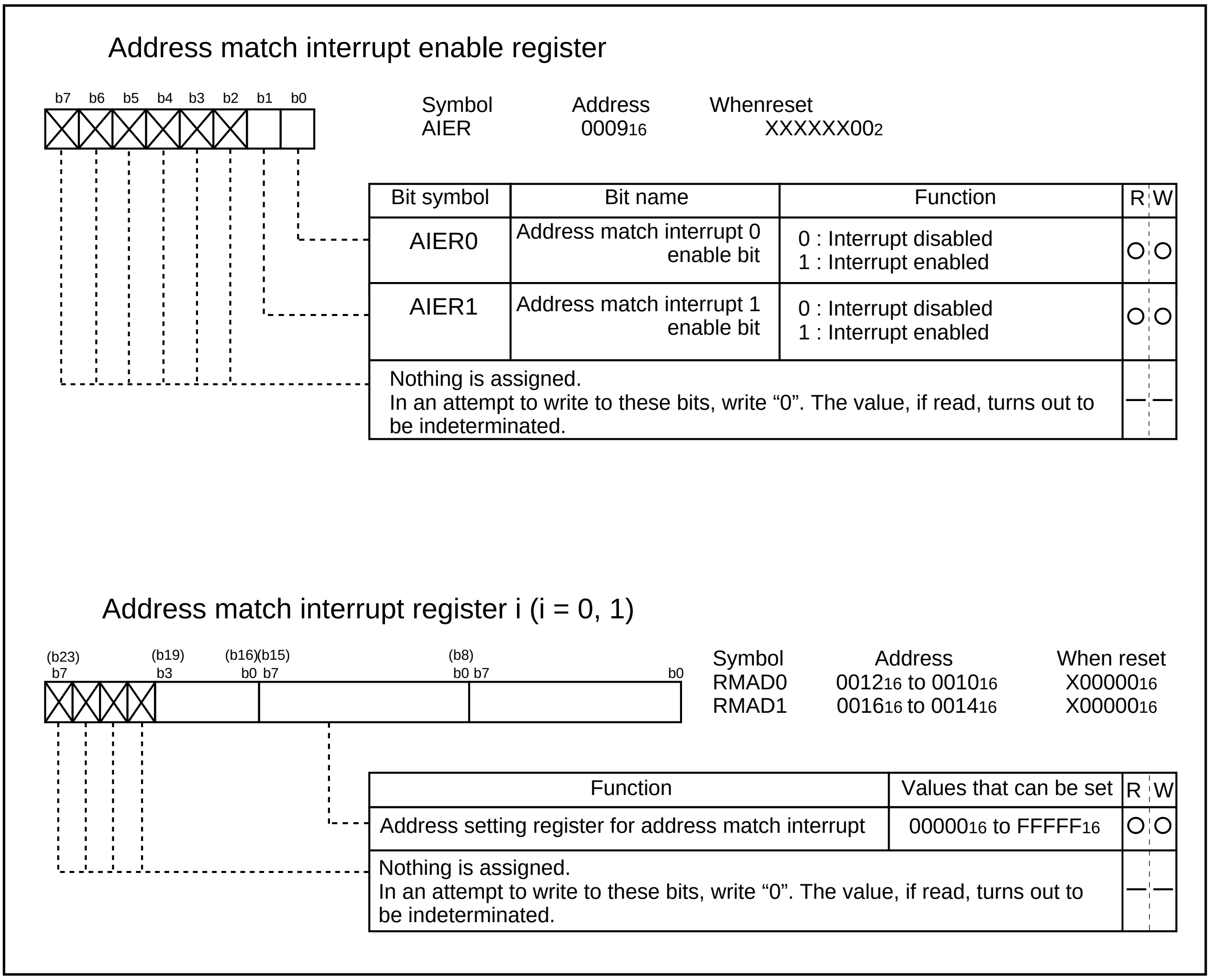

Pull-up transistor
Port P10$_4$-P10$_7$ pull-up select bit
Port P10$_7$ direction register
Key input interrupt control register
(address 004E$_{16}$)
Port P10$_7$ direction register
P10$_7$/$\overline{\text{KI}_3}$
Pull-up transistor
Port P10$_6$ direction register
P10$_6$/$\overline{\text{KI}_2}$
Interrupt control circuit
Key input interrupt request
Pull-up transistor
Port P10$_5$ direction register
P10$_5$/$\overline{\text{KI}_1}$
Pull-up transistor
Port P10$_4$ direction register
P10$_4$/$\overline{\text{KI}_0}$

Figure 6-12. Block diagram of key input interrupt

Address Match Interrupt

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

Figure 6-13 shows the address match interrupt-related registers.



Address match interrupt enable register

Symbol	Address	Whenreset
AIER	0009$_{16}$	XXXXXX00$_2$

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

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

Symbol	Address	When reset
RMAD0	0012$_{16}$ to 0010$_{16}$	X00000$_{16}$
RMAD1	0016$_{16}$ to 0014$_{16}$	X00000$_{16}$

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

Figure 6-13. Address match interrupt-related registers

CAN0/1 Wake Up Interrupt

A CAN Wake Up interrupt is generated after one of the CAN buses becomes active. That means the physical bus turns to a dominant level.

This interrupt can only be used to wake up the CPU from wait mode or stop mode.

The CAN Wake Up interrupt can only be used, if the port(s) are configured as CAN ports. One interrupt signal is generated for both CAN channels.

Please note that the Wake Up message wiil be lost.

Figure 6-8 shows the principle to generate the corresponding interrupt signal.

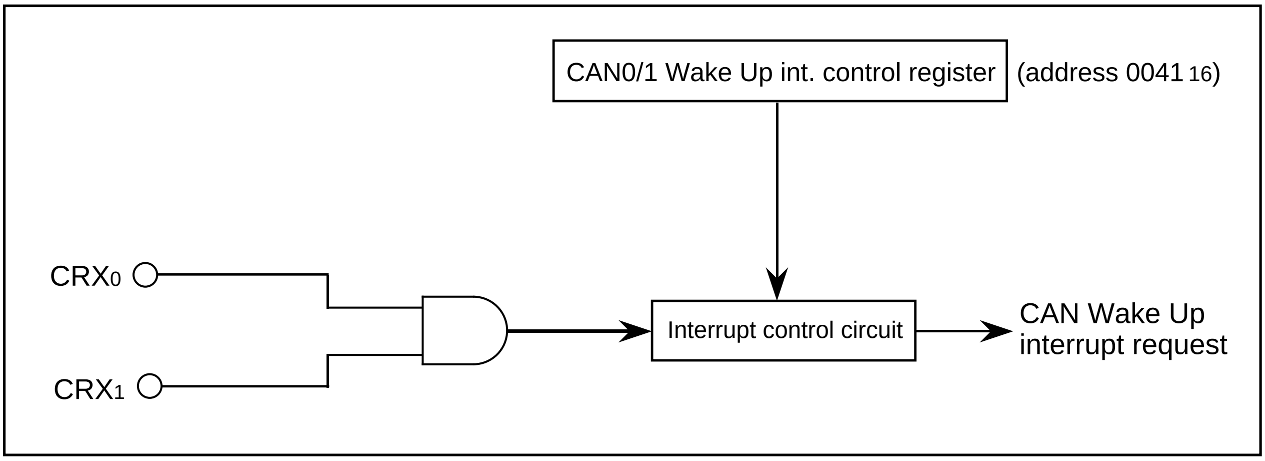

CAN0/1 Wake Up int. control register
(address 0041 16)
CRX0
CRX1
Interrupt control circuit
CAN Wake Up
interrupt request

Figure 6-14. CAN 0/1 Wake Up interrupt

Precautions for Interrupts

(1) Reading address 00000_{16}

- When the maskable interrupt occurs, CPU reads the interrupt information (the interrupt number and interrupt request level) in the interrupt sequence.

The interrupt request bit of the certain interrupt written in address 00000_{16} will then be set to "0".

Reading address 00000_{16} by software sets enabled highest priority interrupt source request bit to "0". Though the interrupt is generated, the interrupt routine may not be executed.

Do not read address 00000_{16} by software.

(2) Setting the stack pointer

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

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

- As for the $\overline{\text{NMI}}$ pin, an interrupt cannot be disabled. Connect it to the Vcc pin via a resistor (pull-up) if unused. Be sure to work on it.
- The $\overline{\text{NMI}}$ pin also serves as $P8_5$, which is exclusively for input. Reading the contents of the P8 register allows reading the pin value. Use the reading of this pin only for establishing the pin level at the time when the $\overline{\text{NMI}}$ interrupt is input.
- Do not reset the CPU with the input to the $\overline{\text{NMI}}$ pin being in the "L" state.
- Do not attempt to go into stop mode with the input to the $\overline{\text{NMI}}$ pin being in the "L" state. With the input to the $\overline{\text{NMI}}$ pin being in the "L" state, the CM10 is fixed to "0", so attempting to go into stop mode is turned down.
- Do not attempt to go into wait mode with the input to the $\overline{\text{NMI}}$ pin being in the "L" state. With the input to the $\overline{\text{NMI}}$ pin being in the "L" state, the CPU stops but the oscillation does not stop, so no power is saved. In this instance, the CPU is returned to the normal state by a later interrupt.
- Signals input to the $\overline{\text{NMI}}$ pin require an "L" level of 1 clock or more, from the operation clock of the CPU.

(4) External interrupt

- Either an "L" level or an "H" level of at least 250 ns width is necessary for the signal input to pins $\overline{\text{INT0}}$ through $\overline{\text{INT5}}$ regardless of the CPU operation clock.
- When the polarity of the $\overline{\text{INT0}}$ to $\overline{\text{INT5}}$ pins is changed, the interrupt request bit is sometimes set to "1". After changing the polarity, set the interrupt request bit to "0". Figure 6-15 shows the procedure for changing the $\overline{\text{INT}}$ interrupt generate factor.

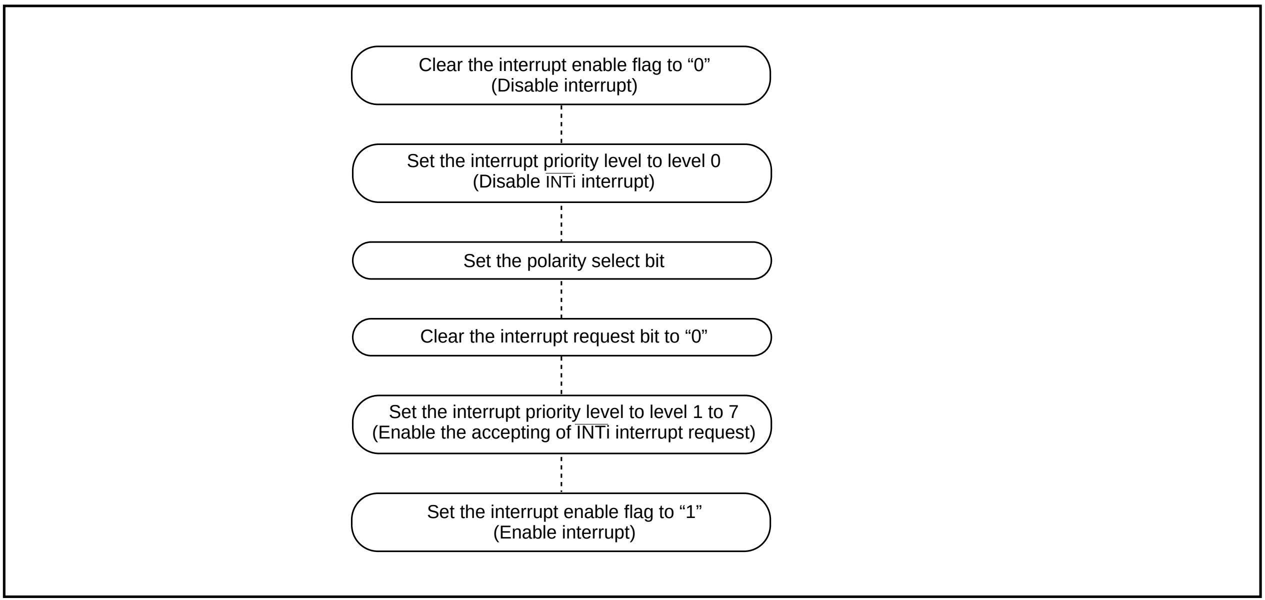

Clear the interrupt enable flag to "0"
(Disable interrupt)
Set the interrupt priority level to level 0
(Disable $\overline{\text{INT}}$i interrupt)
Set the polarity select bit
Clear the interrupt request bit to "0"
Set the interrupt priority level to level 1 to 7
(Enable the accepting of $\overline{\text{INT}}$i interrupt request)
Set the interrupt enable flag to "1"
(Enable interrupt)

Figure 6-15. Switching condition of $\overline{\text{INT}}$ interrupt request

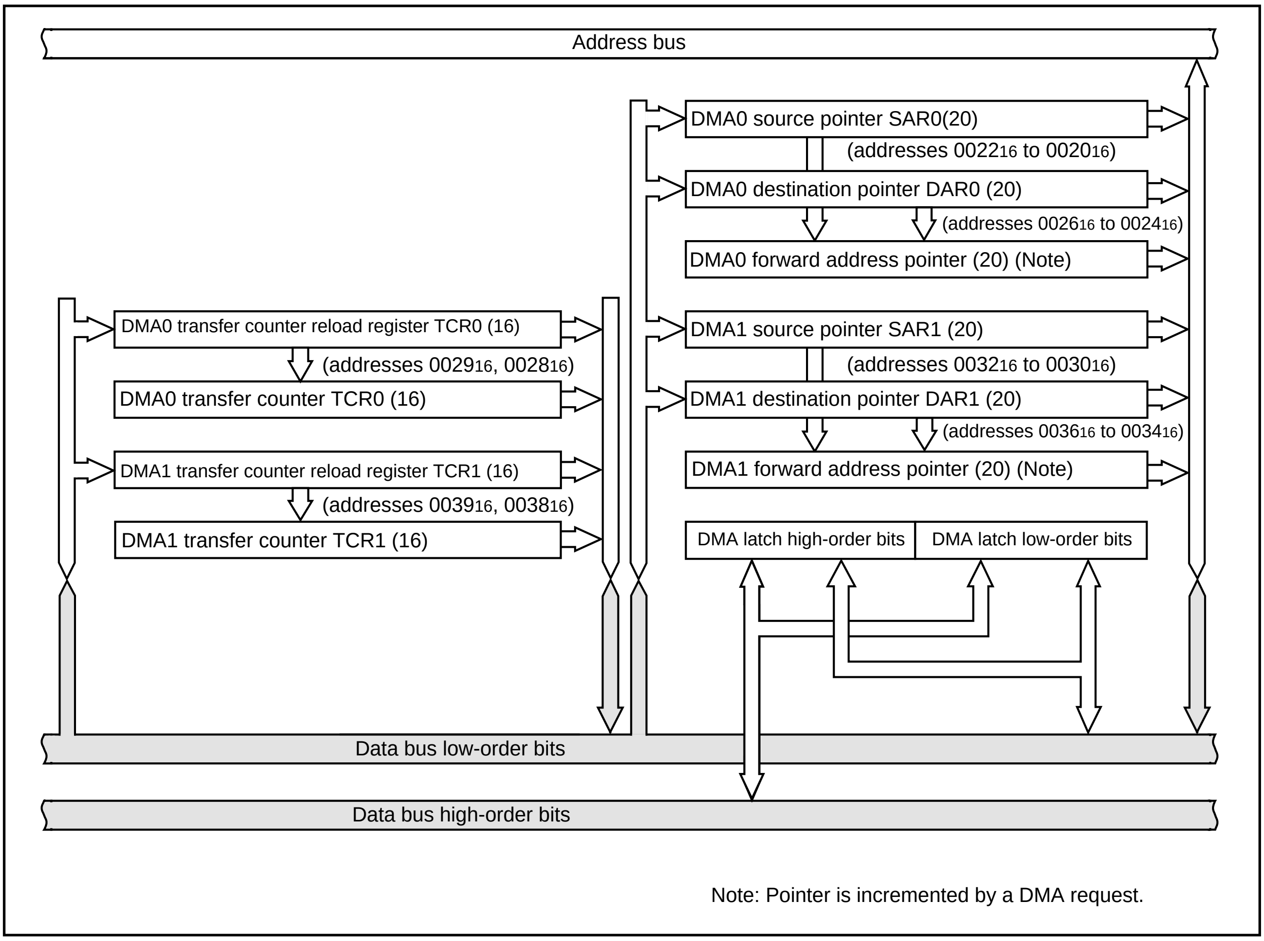
DMAC

This microcomputer has two DMAC (direct memory access controller) channels that allow data to be sent to memory without using the CPU. Table 7-1 shows the DMAC specifications. Figure 7-1 shows the block diagram of the DMAC. Figures 7-2 to 7-4 show the registers used by the DMAC.

Table 7-1. DMAC specifications

Item	Specification
No. of channels	2 (cycle steal method)
Transfer memory space	• From any address in the 1M bytes space to a fixed address • From a fixed address to any address in the 1M bytes space • From a fixed address to a fixed address (Note that DMA-related registers [0020_{16} to $003F_{16}$] cannot be accessed)
Maximum No. of bytes transferred	128K bytes (with 16-bit transfers) or 64K bytes (with 8-bit transfers)
DMA request factors (Note)	Falling edge of $\overline{\text{INT0}}$ or $\overline{\text{INT1}}$ ($\overline{\text{INT0}}$ can be selected by DMA0, $\overline{\text{INT1}}$ byDMA1) or both edge Timer A0 to timer A4 interrupt requests Timer B0 to timer B5 interrupt requests UART0 transmission and reception interrupt requests UART1 transmission and reception interrupt requests UART2 transmission and reception interrupt requests Serial I/O3 interrupt request A-D conversion interrupt requests Software triggers
Channel priority	DMA0 takes precedence if DMA0 and DMA1 requests are generated simultaneously
Transfer unit	8 bits or 16 bits
Transfer address direction	forward/fixed (forward direction cannot be specified for both source and destination simultaneously)
Transfer mode	• Single transfer The DMA enable bit is cleared and transfer ends when an underflow occurs in the transfer counter • Repeat transfer When an underflow occurs in the transfer counter, the value in the transfer counter reload register is reloaded into the transfer counter and the DMA transfer is repeated
DMA interrupt request generation timing	When an underflow occurs in the transfer counter
DMA startup	• Single transfer Transfer starts when the DMA is requested after “1” is written to the DMA enable bit • Repeat transfer Transfer starts when the DMA is requested after “1” is written to the DMA enable bit Transfer starts when the DMA is requested after an underflow occurs in the transfer counter
DMA shutdown	• When “0” is written to the DMA enable bit • When, in single transfer mode, an underflow occurs in the transfer counter
Forward address pointer and reload timing for transfer counter	When DMA transfer starts, the value of whichever of the source or destination pointer that is set up as the forward pointer is reloaded into the forward address pointer. The value in the transfer counter reload register is reloaded into the transfer counter.
Writing to register	Registers specified for forward direction transfer are always write enabled. Registers specified for fixed address transfer are write-enabled when the DMA enable bit is “0”.
Reading the register	Can be read at any time. However, when the DMA enable bit is “1”, reading the register set up as the forward register is the same as reading the value of the forward address pointer.

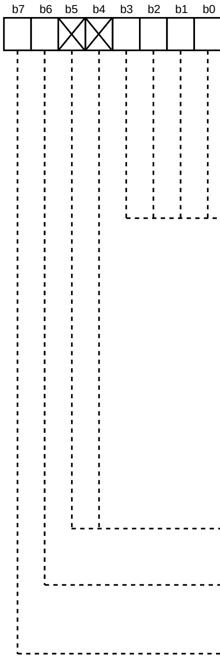
Note: DMA transfer is not effective to any interrupt.


Address bus
DMA0 source pointer SAR0(20)
(addresses 0022_{16} to 0020_{16})
DMA0 destination pointer DAR0 (20)
(addresses 0026_{16} to 0024_{16})
DMA0 forward address pointer (20) (Note)
DMA1 source pointer SAR1 (20)
(addresses 0032_{16} to 0030_{16})
DMA1 destination pointer DAR1 (20)
(addresses 0036_{16} to 0034_{16})
DMA1 forward address pointer (20) (Note)
DMA0 transfer counter reload register TCR0 (16)
(addresses 0029_{16}, 0028_{16})
DMA0 transfer counter TCR0 (16)
DMA1 transfer counter reload register TCR1 (16)
(addresses 0039_{16}, 0038_{16})
DMA1 transfer counter TCR1 (16)
DMA latch high-order bits
DMA latch low-order bits
Data bus low-order bits
Data bus high-order bits

Note: Pointer is incremented by a DMA request.

Figure 7-1. Block diagram of DMAC

DMA0 request cause select register



Symbol DM0SL

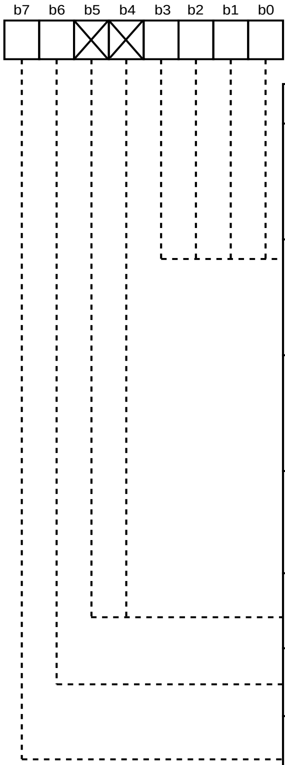
Address $03B8_{16}$

When reset 00_{16}

Bit symbol	Bit name	Function	R	W
DSEL0	DMA request cause select bit	b3 b2 b1 b0 0 0 0 0 : Falling edge of INT0 pin 0 0 0 1 : Software trigger 0 0 1 0 : Timer A0 0 0 1 1 : Timer A1 0 1 0 0 : Timer A2 0 1 0 1 : Timer A3 0 1 1 0 : Timer A4 (DMS=0) /two edges of INT0 pin (DMS=1) 0 1 1 1 : Timer B0 (DMS=0) Timer B3 (DMS=1) 1 0 0 0 : Timer B1 (DMS=0) Timer B4 (DMS=1) 1 0 0 1 : Timer B2 (DMS=0) Timer B5 (DMS=1) 1 0 1 0 : UART0 transmit 1 0 1 1 : UART0 receive 1 1 0 0 : UART2 transmit 1 1 0 1 : UART2 receive 1 1 1 0 : A-D conversion 1 1 1 1 : UART1 transmit	○	○
DSEL1			○	○
DSEL2			○	○
DSEL3			○	○
Nothing is assigned. These bits can neither be set nor reset. When read, the value of these bits is “0”.			–	–
DMS	DMA request cause expansion bit	0 : Normal 1 : Expanded cause	○	○
DSR	Software DMA request bit	If software trigger is selected, a DMA request is generated by setting this bit to “1” (When read, the value of this bit is always “0”)	○	○

Figure 7-2. DMAC register (1)

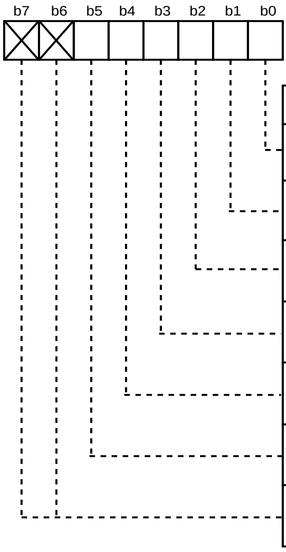
DMA1 request cause select register



Symbol	Address	When reset
DM1SL	03BA$_{16}$	00$_{16}$

Bit symbol	Bit name	Function	R	W
DSEL0	DMA request cause select bit	b3 b2 b1 b0 0 0 0 0 : Falling edge of $\overline{\text{INT0}}$ pin 0 0 0 1 : Software trigger 0 0 1 0 : Timer A0 0 0 1 1 : Timer A1 0 1 0 0 : Timer A2 0 1 0 1 : Timer A3(DMS=0) /serial I/O3 (DMS=1) 0 1 1 0 : Timer A4 (DMS=0) 0 1 1 1 : Timer B0 (DMS=0) /two edges of $\overline{\text{INT1}}$ (DMS=1) 1 0 0 0 : Timer B1 1 0 0 1 : Timer B2 1 0 1 0 : UART0 transmit 1 0 1 1 : UART0 receive 1 1 0 0 : UART2 transmit 1 1 0 1 : UART2 receive 1 1 1 0 : A-D conversion 1 1 1 1 : UART1 receive	○	○
DSEL1			○	○
DSEL2			○	○
DSEL3			○	○
Nothing is assigned. These bits can neither be set nor reset. When read, the value of these bits is 0 .			–	–
DMS	DMA request cause expansion bit	0 : Normal 1 : Expanded cause	○	○
DSR	Software DMA request bit	If software trigger is selected, a DMA request is generated by setting this bit to 1 (When read, the value of this bit is always 0)	○	○

DMAi control register

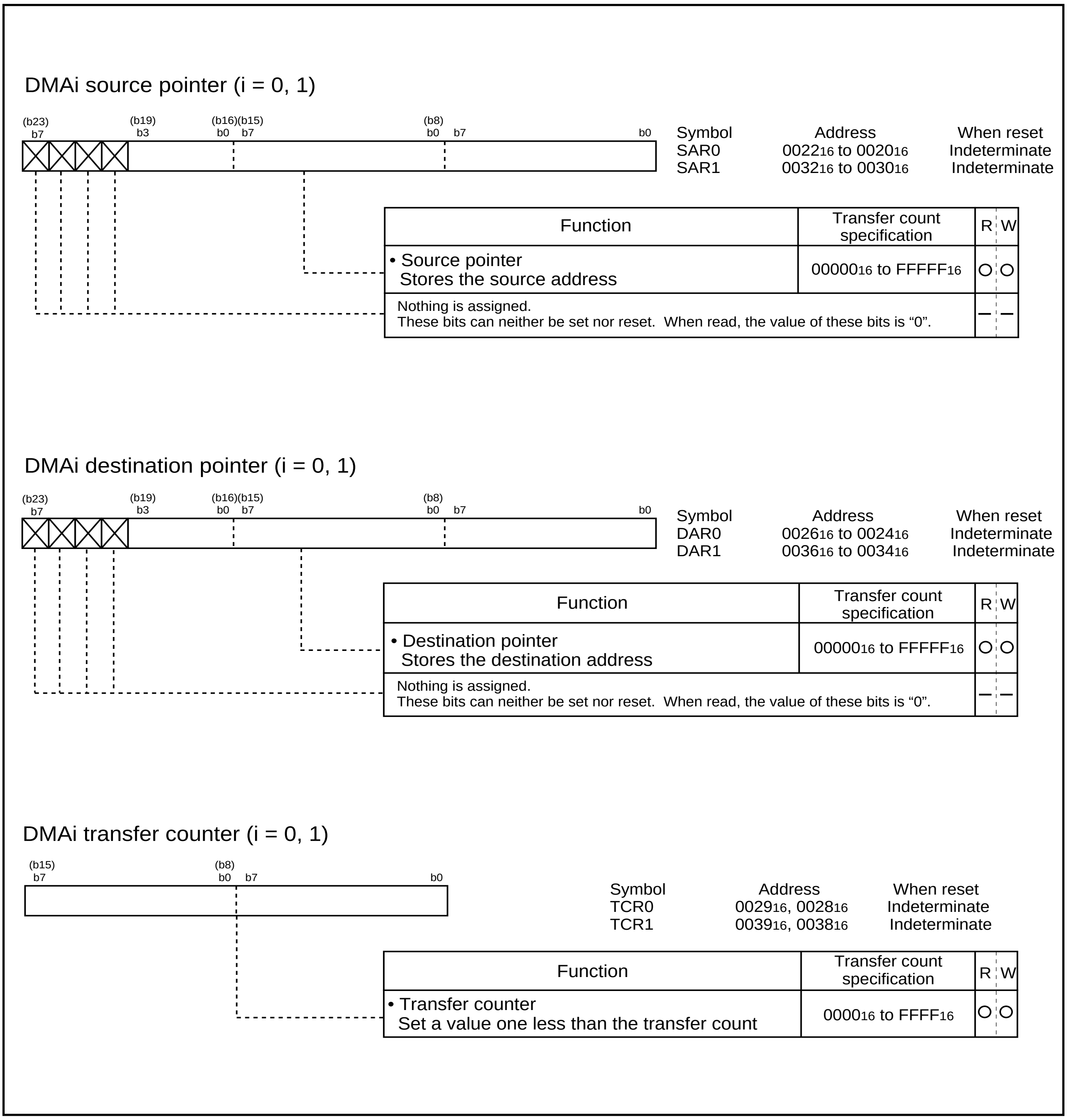


Symbol	Address	When reset
DMiCON(i=0,1)	002C$_{16}$, 003C$_{16}$	00000X00$_2$

Bit symbol	Bit name	Function	R	W
DMBIT	Transfer unit bit select bit	0 : 16 bits 1 : 8 bits	○	○
DMASL	Repeat transfer mode select bit	0 : Single transfer 1 : Repeat transfer	○	○
DMAS	DMA request bit (Note 1)	0 : DMA not requested 1 : DMA requested	○	○ (Note 2)
DMAE	DMA enable bit	0 : Disabled 1 : Enabled	○	○
DSD	Source address direction select bit (Note 3)	0 : Fixed 1 : Forward	○	○
DAD	Destination address direction select bit (Note 3)	0 : Fixed 1 : Forward	○	○
Nothing is assigned. These bits can neither be set nor reset. When read, the value of these bits is 0 .			–	–

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

Figure 7-3. DMAC register (2)



DMAi source pointer (i = 0, 1)

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

Function	Transfer count specification	R	W
• Source pointer Stores the source address	00000_{16} to $FFFFF_{16}$	○	○
Nothing is assigned. These bits can neither be set nor reset. When read, the value of these bits is “0”.		–	–

DMAi destination pointer (i = 0, 1)

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

Function	Transfer count specification	R	W
• Destination pointer Stores the destination address	00000_{16} to $FFFFF_{16}$	○	○
Nothing is assigned. These bits can neither be set nor reset. When read, the value of these bits is “0”.		–	–

DMAi transfer counter (i = 0, 1)

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

Function	Transfer count specification	R	W
• Transfer counter Set a value one less than the transfer count	0000_{16} to $FFFF_{16}$	○	○

Figure 7-4. DMAC register (3)

(1) Transfer cycle

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

(a) Effect of source and destination addresses

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

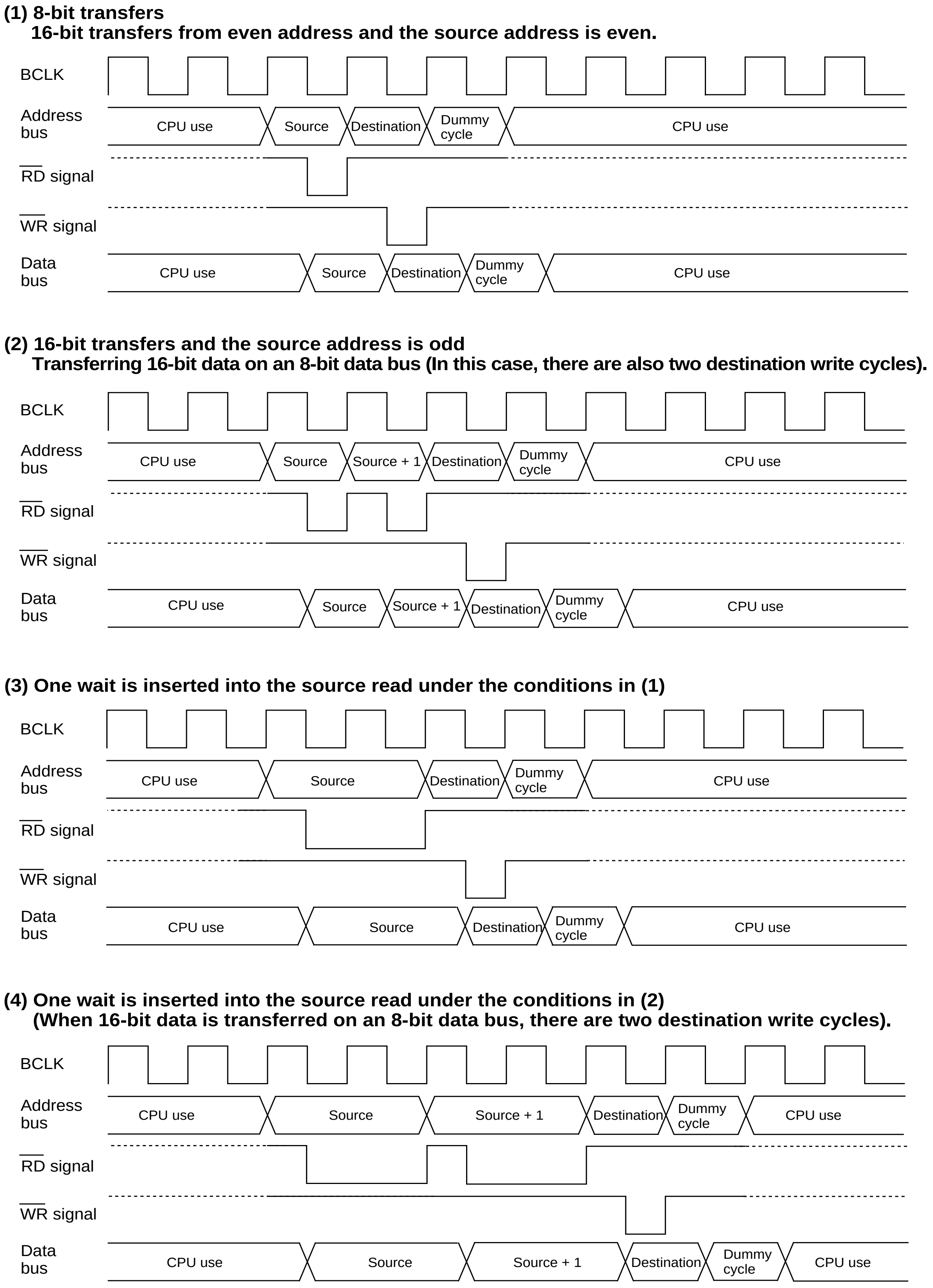
(b) Effect of BYTE pin level

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

(c) Effect of software wait

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

Figure 7-5 shows the example of the transfer cycles for a source read. For convenience, the destination write cycle is shown as one cycle and the source read cycles for the different conditions are shown. In reality, the destination write cycle is subject to the same conditions as the source read cycle, with the transfer cycle changing accordingly. When calculating the transfer cycle, remember to apply the respective conditions to both the destination write cycle and the source read cycle. For example (2) in Figure 7-5, if data is being transferred in 16-bit units on an 8-bit bus, two bus cycles are required for both the source read cycle and the destination write cycle.



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

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

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

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

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

Figure 7-5. Example of the transfer cycles for a source read

(2) DMAC transfer cycles

Any combination of even or odd transfer read and write addresses is possible. Table 7-2 shows the number of DMAC transfer cycles.

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

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

Table 7-2. No. of DMAC transfer cycles

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

Coefficient j, k

Internal memory			External memory		
Internal ROM/RAM No wait	Internal ROM/RAM With wait	SFR area	Separate bus No wait	Separate bus With wait	Multiplex bus
1	2	2	1	2	3

Watchdog Timer

The watchdog timer has the function of detecting when the program is out of control. The watchdog timer is a 15-bit counter which down-counts the clock derived by dividing the internal clock ϕ using the prescaler. A watchdog timer interrupt is generated when an underflow occurs in the watchdog timer. When X$_{IN}$ is selected for the internal clock ϕ, bit 7 of the watchdog timer control register (address 000F$_{16}$) selects the prescaler division ratio (by 16 or by 128). When X$_{CIN}$ is selected as the internal clock ϕ, the prescaler is set for division by 2 regardless of bit 7 of the watchdog timer control register (address 000F$_{16}$). Table 8-1 shows the periodic table for the watchdog timer.

Table 8-1. Watchdog timer periodic table (X$_{IN}$ = 10MHz, X$_{CIN}$ = 32kHz)

CM07	CM06	CM17	CM16	Internal clock ϕ	WDC7	Period
0	0	0	0	10MHz	0	Approx. 52.4ms (Note)
					1	Approx. 419.2ms (Note)
0	0	0	1	5MHz	0	Approx. 104.9ms (Note)
					1	Approx. 838.8ms (Note)
0	0	1	0	2.5MHz	0	Approx. 209.7ms (Note)
					1	Approx. 1.68s (Note)
0	0	1	1	0.625MHz	0	Approx. 838.8ms (Note)
					1	Approx. 6.71s (Note)
0	1	Invalid	Invalid	1.25MHz	0	Approx. 419.2ms (Note)
					1	Approx. 3.35s (Note)
1	Invalid	Invalid	Invalid	32kHz	Invalid	Approx. 2s (Note)

Note: Error is generated by the prescaler.

The watchdog timer is initialized by writing to the watchdog timer start register (address 000E$_{16}$) and when a watchdog timer interrupt request is generated. The prescaler is initialized only when the microcomputer is reset. After a reset is cancelled, the watchdog timer and prescaler are both stopped. The count is started by writing to the watchdog timer start register (address 000E$_{16}$).

Figure 8-1 shows the block diagram of the watchdog timer. Figure 8-2 shows the watchdog timer-related registers.

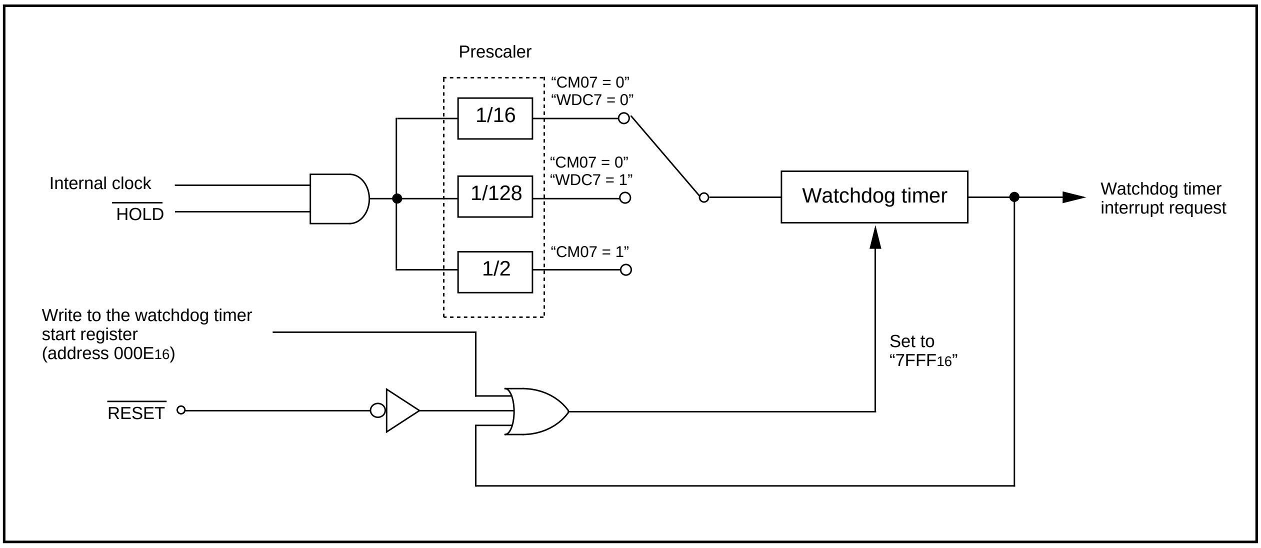


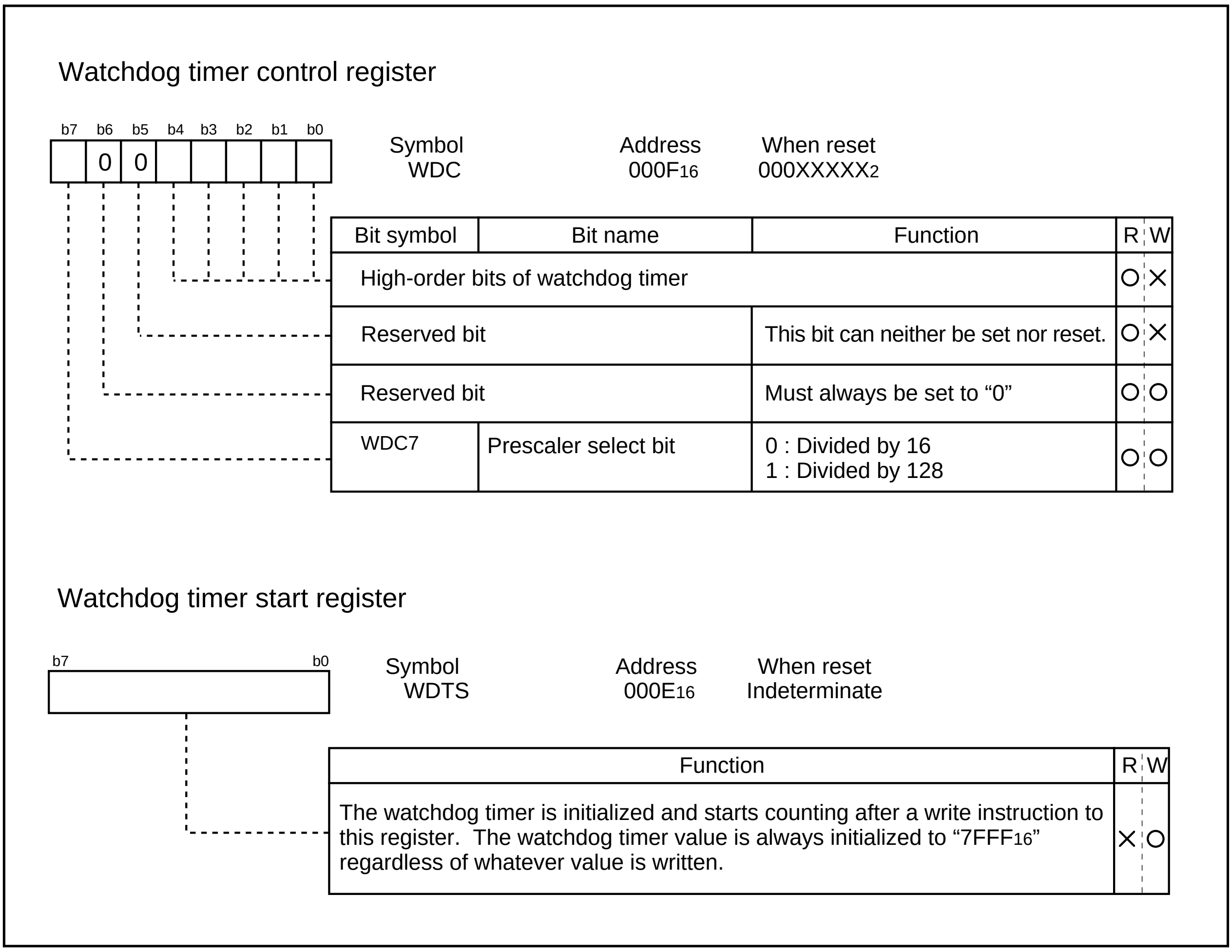
Figure 8-1. Block diagram of watchdog timer

Watchdog Timer during Wait Mode

The watchdog timer is supplied by the internal clock ϕ. If the internal clock ϕ stops, the watchdog timer stops also. When executing a wait instruction, the internal clock ϕ stops if no interrupt request is pending or any interrupt request that is pending is marked (i.e. the interrupts IPL is set to a value not greater than the CPU's IPL).

The internal clock ϕ and the watchdog timer will continue running when at the issuance of the wait instruction any nonmasked interrupt request was pending and the I flag in the flag register was cleared.

The same applies to an internal clock ϕ stopped during wait mode: If during wait mode a disabled but not masked interrupt is requested and the I flag is cleared internal clock ϕ restarts. Though the CPU remains in wait, the watchdog timer recommences activity where it left off and will in time request an interrupt itself.



Watchdog timer control register

Symbol WDC Address 000F$_{16}$ When reset 000XXXXX$_2$

Bit symbol	Bit name	Function	R	W
High-order bits of watchdog timer			○	×
Reserved bit		This bit can neither be set nor reset.	○	×
Reserved bit		Must always be set to “0”	○	○
WDC7	Prescaler select bit	0 : Divided by 16 1 : Divided by 128	○	○

Watchdog timer start register

Symbol WDTS Address 000E$_{16}$ When reset Indeterminate

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

Figure 8-2. Watchdog timer control and start registers

Timer

There are eleven 16-bit timers. These timers can be classified by function into timers A (five) and timers B (six). All these timers function independently. Figures 9-1 and 9-2 show the block diagram of timers.

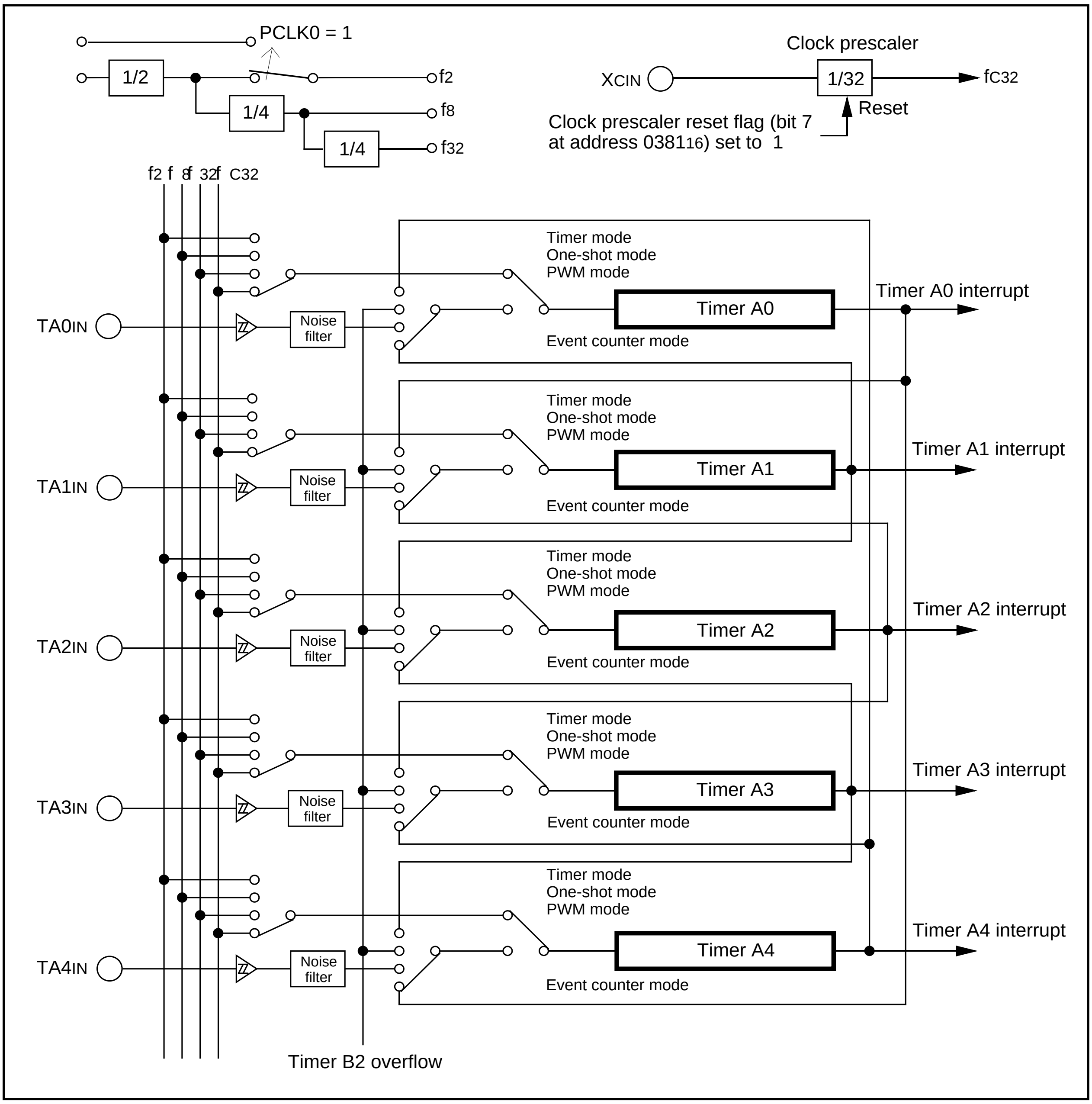


Figure 9-1. Timer A block diagram

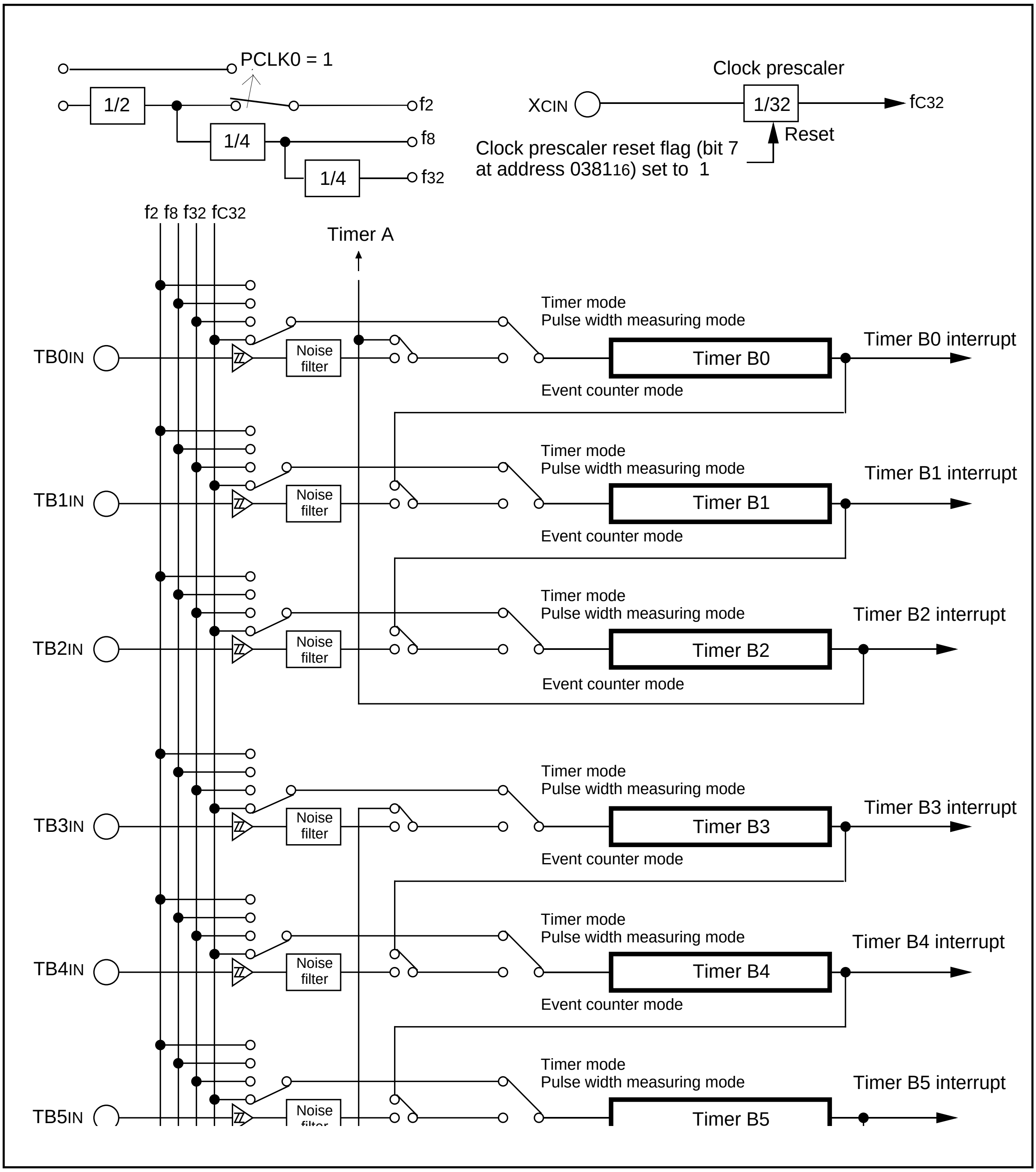


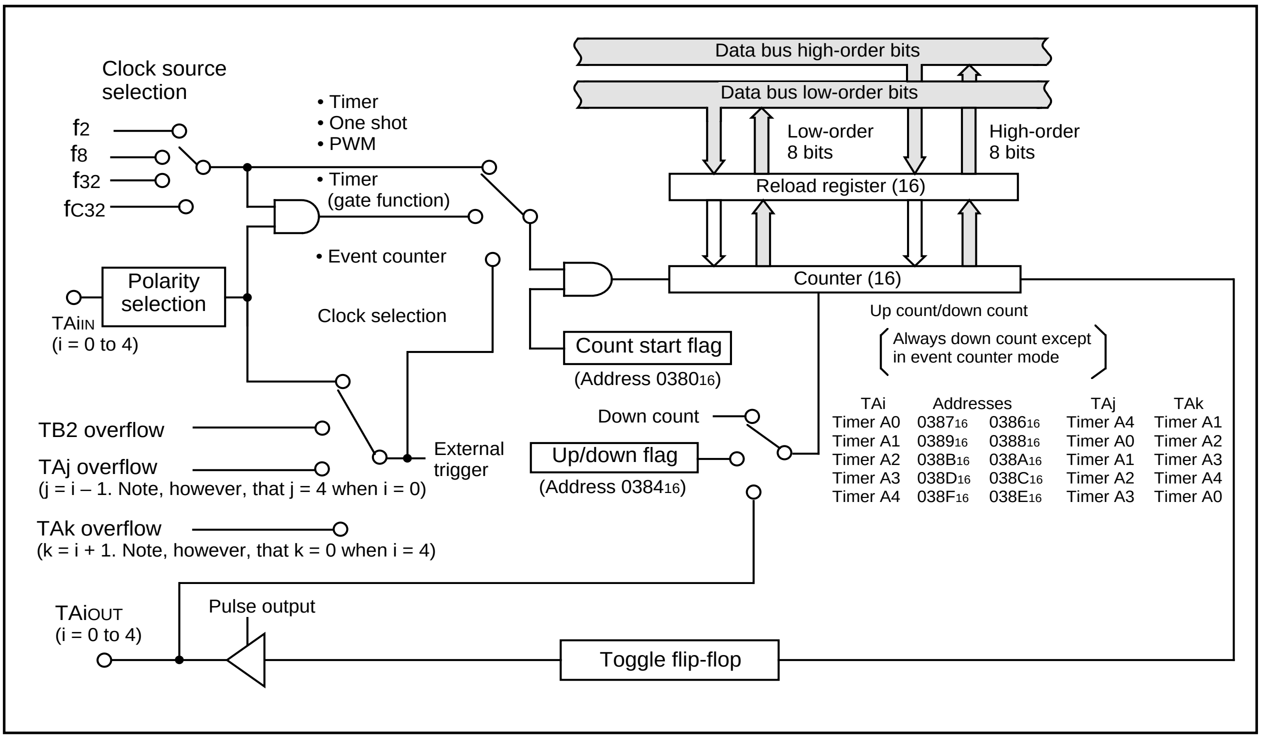
Figure 9-2. Timer B block diagram

Timer A

Figure 9-3 shows the block diagram of timer A. Figures 9-4 to 9-6 show the timer A-related registers. Except in event counter mode, timers A0 through A4 all have the same function. Use the timer Ai mode register (i = 0 to 4) bits 0 and 1 to choose the desired mode.

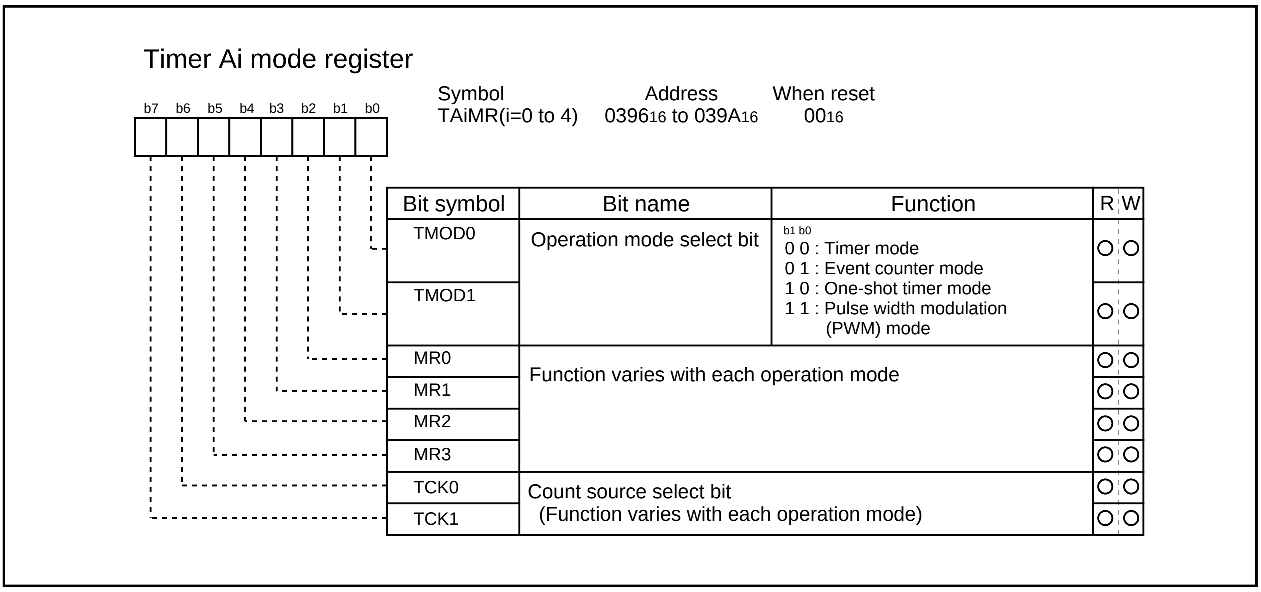
Timer A has the four operation modes listed as follows:

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



TAi	Addresses		TAj	TAk
Timer A0	0387_{16}	0386_{16}	Timer A4	Timer A1
Timer A1	0389_{16}	0388_{16}	Timer A0	Timer A2
Timer A2	$038B_{16}$	$038A_{16}$	Timer A1	Timer A3
Timer A3	$038D_{16}$	$038C_{16}$	Timer A2	Timer A4
Timer A4	$038F_{16}$	$038E_{16}$	Timer A3	Timer A0

Figure 9-3. Block diagram of timer A



Timer Ai mode register

b7 b6 b5 b4 b3 b2 b1 b0

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

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

Figure 9-4. Timer A-related registers (1)

Timer Ai register (Note)

(b15) b7 (b8) b0 b7 b0

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

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

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

Count start flag

b7 b6 b5 b4 b3 b2 b1 b0

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

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

Up/down flag

b7 b6 b5 b4 b3 b2 b1 b0

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

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

Figure 9-5. Timer A-related registers (2)

One-shot start flag

b7 b6 b5 b4 b3 b2 b1 b0

Symbol ONSF Address 0382$_{16}$ When reset 00X00000$_2$

Bit symbol	Bit name	Function	R	W
TA0OS	Timer A0 one-shot start flag	1 : Timer start When read, the value is “0”	○	○
TA1OS	Timer A1 one-shot start flag		○	○
TA2OS	Timer A2 one-shot start flag		○	○
TA3OS	Timer A3 one-shot start flag		○	○
TA4OS	Timer A4 one-shot start flag		○	○
Nothing is assigned. This bit can neither be set nor reset. When read, its content is indeterminate.			—	—
TA0TGL	Timer A0 event/trigger select bit	b7 b6 0 0 : Input on TA0$_{IN}$ is selected (Note) 0 1 : TB2 overflow is selected 1 0 : TA4 overflow is selected 1 1 : TA1 overflow is selected	○	○
TA0TGH			○	○

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

Trigger select register

b7 b6 b5 b4 b3 b2 b1 b0

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

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

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

Clock prescaler reset flag

b7 b6 b5 b4 b3 b2 b1 b0

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

Bit symbol	Bit name	Function	R	W
Nothing is assigned. These bits can neither be set nor reset. When read, their contents are indeterminate.			—	—
CPSR	Clock prescaler reset flag	0 : No effect 1 : Prescaler is reset (When read, the value is “0”)	○	○

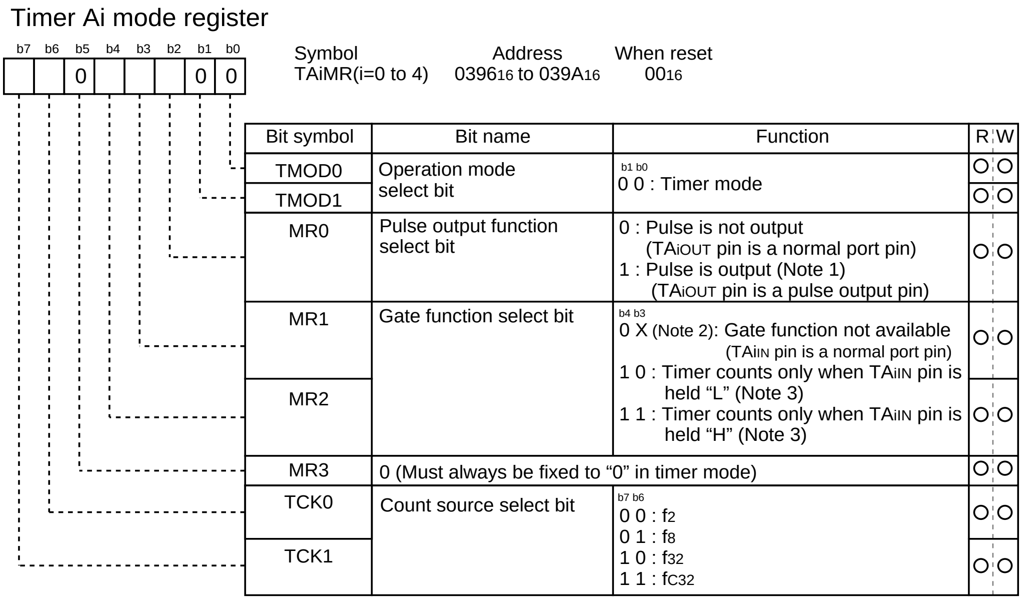
Figure 9-6. Timer A-related registers (3)

(1) Timer mode

In this mode, the timer counts an internally generated count source. (See Table 9-1.) Figure 9-7 shows the timer Ai mode register in timer mode.

Table 9-1. Specifications of timer mode

Item	Specification
Count source	f_2, f_8, f_{32}, f_{C32}
Count operation	• Down count • When the timer underflows, it reloads the reload register contents before continuing counting
Divide ratio	1/(n+1) n : Set value
Count start condition	Count start flag is set (= 1)
Count stop condition	Count start flag is reset (= 0)
Interrupt request generation timing	When the timer underflows
TAiIN pin function	Programmable I/O port or gate input
TAiOUT pin function	Programmable I/O port or pulse output
Read from timer	Count value can be read out by reading timer Ai register
Write to timer	• When counting stopped When a value is written to timer Ai register, it is written to both reload register and counter • When counting in progress When a value is written to timer Ai register, it is written to only reload register (Transferred to counter at next reload time)
Select function	• Gate function Counting can be started and stopped by the TAiIN pin's input signal • Pulse output function Each time the timer underflows, the TAiOUT pin's polarity is reversed



Timer Ai mode register

b7	b6	b5	b4	b3	b2	b1	b0
		0				0	0

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

Bit symbol	Bit name	Function	R	W
TMOD0	Operation mode select bit	b1 b0 0 0 : Timer mode	○	○
TMOD1			○	○
MR0	Pulse output function select bit	0 : Pulse is not output (TAiOUT pin is a normal port pin) 1 : Pulse is output (Note 1) (TAiOUT pin is a pulse output pin)	○	○
MR1	Gate function select bit	b4 b3 0 X (Note 2): Gate function not available (TAiIN pin is a normal port pin) 1 0 : Timer counts only when TAiIN pin is held “L” (Note 3) 1 1 : Timer counts only when TAiIN pin is held “H” (Note 3)	○	○
MR2			○	○
MR3	0 (Must always be fixed to “0” in timer mode)		○	○
TCK0	Count source select bit	b7 b6 0 0 : f_2 0 1 : f_8 1 0 : f_{32} 1 1 : f_{C32}	○	○
TCK1			○	○

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

Figure 9-7. Timer Ai mode register in timer mode

(2) Event counter mode

In this mode, the timer counts an external signal or an internal timer's overflow. Timers A0 and A1 can count a single-phase external signal. Timers A2, A3, and A4 can count a single-phase and a two-phase external signal. Table 9-2 lists timer specifications when counting a single-phase external signal. Figure 9-8 shows the timer Ai mode register in event counter mode.

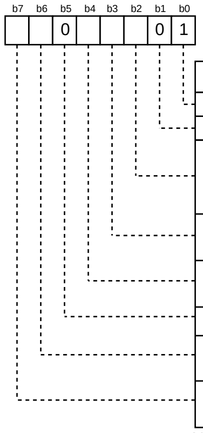
Table 9-2 lists timer specifications when counting a two-phase external signal. Figure 9-9 shows the timer Ai mode register in event counter mode.

Table 9-2. Timer specifications in event counter mode (when not processing two-phase pulse signal)

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

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

Timer Ai mode register



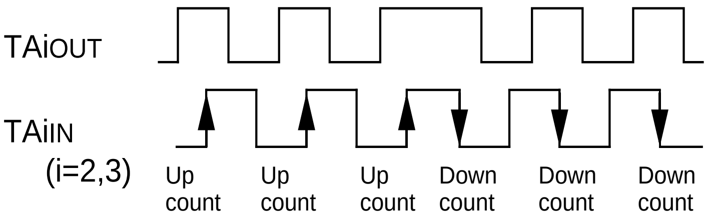
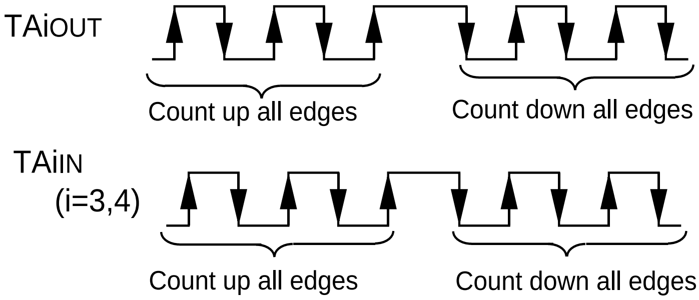
Symbol	Address	When reset
TAiMR(i = 0, 1)	0396_{16}, 0397_{16}	00_{16}

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

Note 1: In event counter mode, the count source is selected by the event / trigger select bit (addresses 0382_{16} and 0383_{16}).
Note 2: The settings of the corresponding port register and port direction register are invalid.
Note 3: Valid only when counting an external signal.
Note 4: When an "L" signal is input to the TAiOUT pin, the downcount is activated. When "H", the upcount is activated. Set the corresponding port direction register to "0".

Figure 9-8. Timer Ai mode register in event counter mode

Table 9-3. Timer specifications in event counter mode (when processing two-phase pulse signal with timers A2, A3, and A4)

Item	Specification
Count source	• Two-phase pulse signals input to TAiIN or TAiOUT pin
Count operation	• Up count or down count can be selected by two-phase pulse signal • When the timer overflows or underflows, the reload register content is reloaded and the timer starts over again (Note)
Divide ratio	1/ (FFFF$_{16}$ - n + 1) for up count 1/ (n + 1) for down count n : Set value
Count start condition	Count start flag is set (= 1)
Count stop condition	Count start flag is reset (= 0)
Interrupt request generation timing	Timer overflows or underflows
TAiIN pin function	Two-phase pulse input
TAiOUT pin function	Two-phase pulse input
Read from timer	Count value can be read out by reading timer A2, A3, or A4 register
Write to timer	• When counting stopped When a value is written to timer A2, A3, or A4 register, it is written to both reload register and counter • When counting in progress When a value is written to timer A2, A3, or A4 register, it is written to only reload register. (Transferred to counter at next reload time.)
Select function	• Normal processing operation The timer counts up rising edges or counts down falling edges on the TAiIN pin when input signal on the TAiOUT pin is “H”  TAiOUT TAiIN (i=2,3) Up count Up count Up count Down count Down count Down count • Multiply-by-4 processing operation If the phase relationship is such that the TAiIN pin goes “H” when the input signal on the TAiOUT pin is “H”, the timer counts up rising and falling edges on the TAiOUT and TAiIN pins. If the phase relationship is such that the TAiIN pin goes “L” when the input signal on the TAiOUT pin is “H”, the timer counts down rising and falling edges on the TAiOUT and TAiIN pins.  TAiOUT Count up all edges Count down all edges TAiIN (i=3,4) Count up all edges Count down all edges

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

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

b7	b6	b5	b4	b3	b2	b1	b0
		0				0	1

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

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

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

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

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

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

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

Note 1: This bit is valid for timer A3 mode register.
For timer A2 and A4 mode registers, this bit can be “0” or “1”.
Note 2: When performing two-phase pulse signal processing, make sure the two-phase pulse signal processing operation select bit (address 0384_{16}) is set to “1”. Also, always be sure to set the event/trigger select bit (addresses 0382_{16} and 0383_{16}) to “00”.

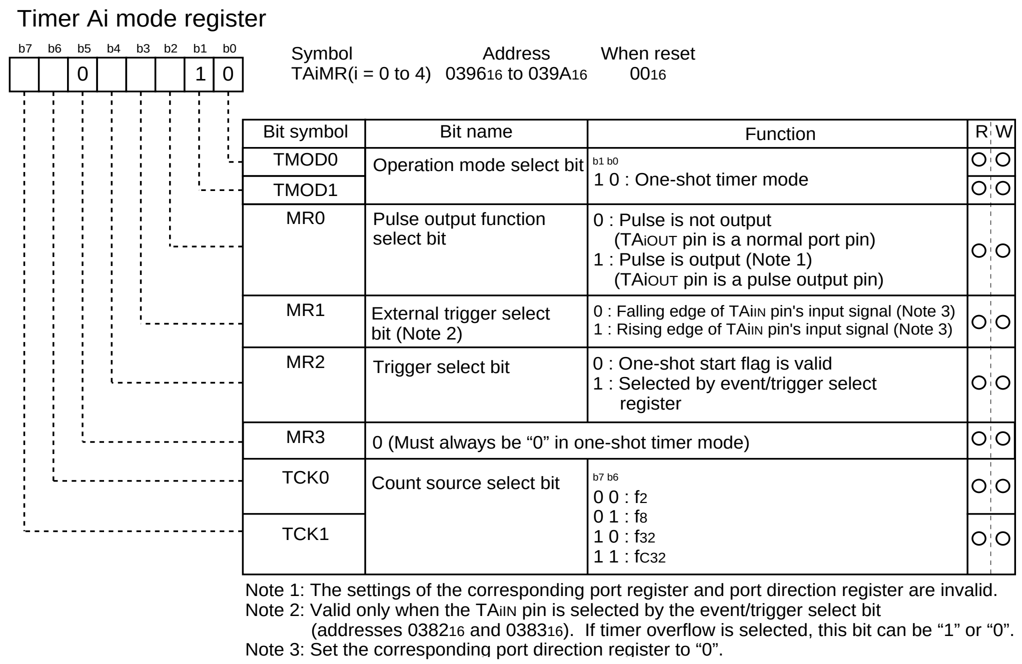
Figure 9-9. Timer Ai mode register in event counter mode

(3) One-shot timer mode

In this mode, the timer operates only once. (See Table 9-4.) When a trigger occurs, the timer starts up and continues operating for a given period. Figure 9-10 shows the timer Ai mode register in one-shot timer mode.

Table 9-4. Timer specifications in one-shot timer mode

Item	Specification
Count source	f$_2$, f$_8$, f$_{32}$, f$_{C32}$
Count operation	• The timer counts down • When the count reaches 0000_{16}, the timer stops counting after reloading a new count • If a trigger occurs when counting, the timer reloads a new count and restarts counting
Divide ratio	1/n n : Set value
Count start condition	• An external trigger is input • The timer overflows • The one-shot start flag is set (= 1)
Count stop condition	• A new count is reloaded after the count has reached 0000_{16} • The count start flag is reset (= 0)
Interrupt request generation timing	The count reaches 0000_{16}
TAi$_{IN}$ pin function	Programmable I/O port or trigger input
TAi$_{OUT}$ pin function	Programmable I/O port or pulse output
Read from timer	When timer Ai register is read, it indicates an indeterminate value
Write to timer	• When counting stopped When a value is written to timer Ai register, it is written to both reload register and counter • When counting in progress When a value is written to timer Ai register, it is written to only reload register (Transferred to counter at next reload time)



Timer Ai mode register

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

Bit symbol	Bit name	Function	R	W
TMOD0	Operation mode select bit	b1 b0 1 0 : One-shot timer mode	○	○
TMOD1			○	○
MR0	Pulse output function select bit	0 : Pulse is not output (TAi$_{OUT}$ pin is a normal port pin) 1 : Pulse is output (Note 1) (TAi$_{OUT}$ pin is a pulse output pin)	○	○
MR1	External trigger select bit (Note 2)	0 : Falling edge of TAi$_{IN}$ pin's input signal (Note 3) 1 : Rising edge of TAi$_{IN}$ pin's input signal (Note 3)	○	○
MR2	Trigger select bit	0 : One-shot start flag is valid 1 : Selected by event/trigger select register	○	○
MR3	0 (Must always be "0" in one-shot timer mode)		○	○
TCK0	Count source select bit	b7 b6 0 0 : f$_2$ 0 1 : f$_8$ 1 0 : f$_{32}$ 1 1 : f$_{C32}$	○	○
TCK1			○	○

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

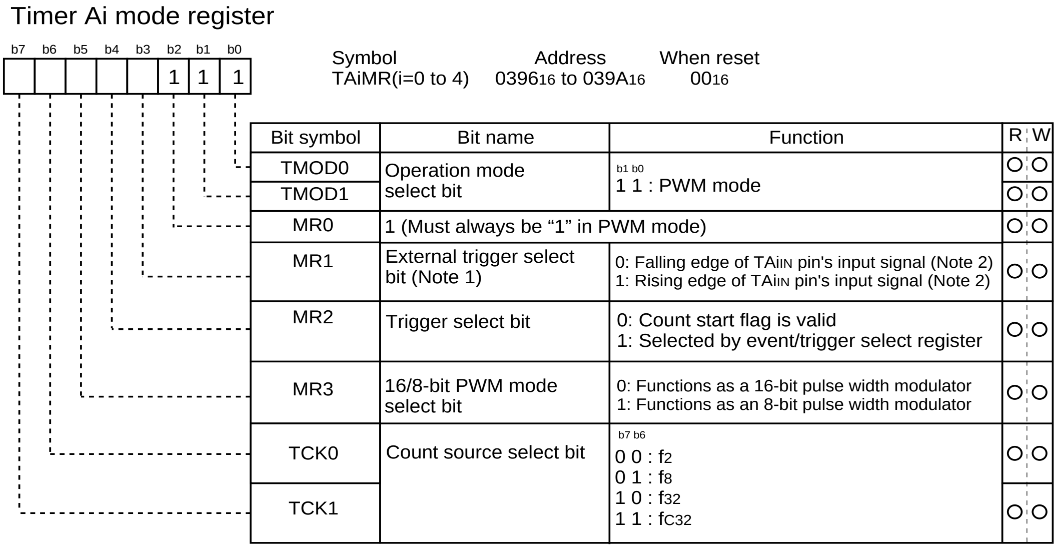
Figure 9-10. Timer Ai mode register in one-shot timer mode

(4) Pulse width modulation (PWM) mode

In this mode, the timer outputs pulses of a given width in succession. (See Table 9-5.) In this mode, the counter functions as either a 16-bit pulse width modulator or an 8-bit pulse width modulator. Figure 9-11 shows the timer Ai mode register in pulse width modulation mode. Figure 9-12 shows the example of how a 16-bit pulse width modulator operates. Figure 9-13 shows the example of how an 8-bit pulse width modulator operates.

Table 9-5. Timer specifications in pulse width modulation mode

Item	Specification
Count source	f2, f8, f32, fC32
Count operation	• The timer counts down (operating as an 8-bit or a 16-bit pulse width modulator) • The timer reloads a new count at a rising edge of PWM pulse and continues counting • The timer is not affected by a trigger that occurs when counting
16-bit PWM	• High level width n / fi n : Set value • Cycle time $(2^{16}-1)$ / fi fixed
8-bit PWM	• High level width n×(m+1) / fi n : values set to timer Ai register's high-order address • Cycle time $(2^{8}-1)$×(m+1) / fi m : values set to timer Ai register's low-order address
Count start condition	• External trigger is input • The timer overflows • The count start flag is set (= 1)
Count stop condition	• The count start flag is reset (= 0)
Interrupt request generation timing	PWM pulse goes "L"
TAiIN pin function	Programmable I/O port or trigger input
TAiOUT pin function	Pulse output
Read from timer	When timer Ai register is read, it indicates an indeterminate value
Write to timer	• When counting stopped When a value is written to timer Ai register, it is written to both reload register and counter • When counting in progress When a value is written to timer Ai register, it is written to only reload register (Transferred to counter at next reload time)

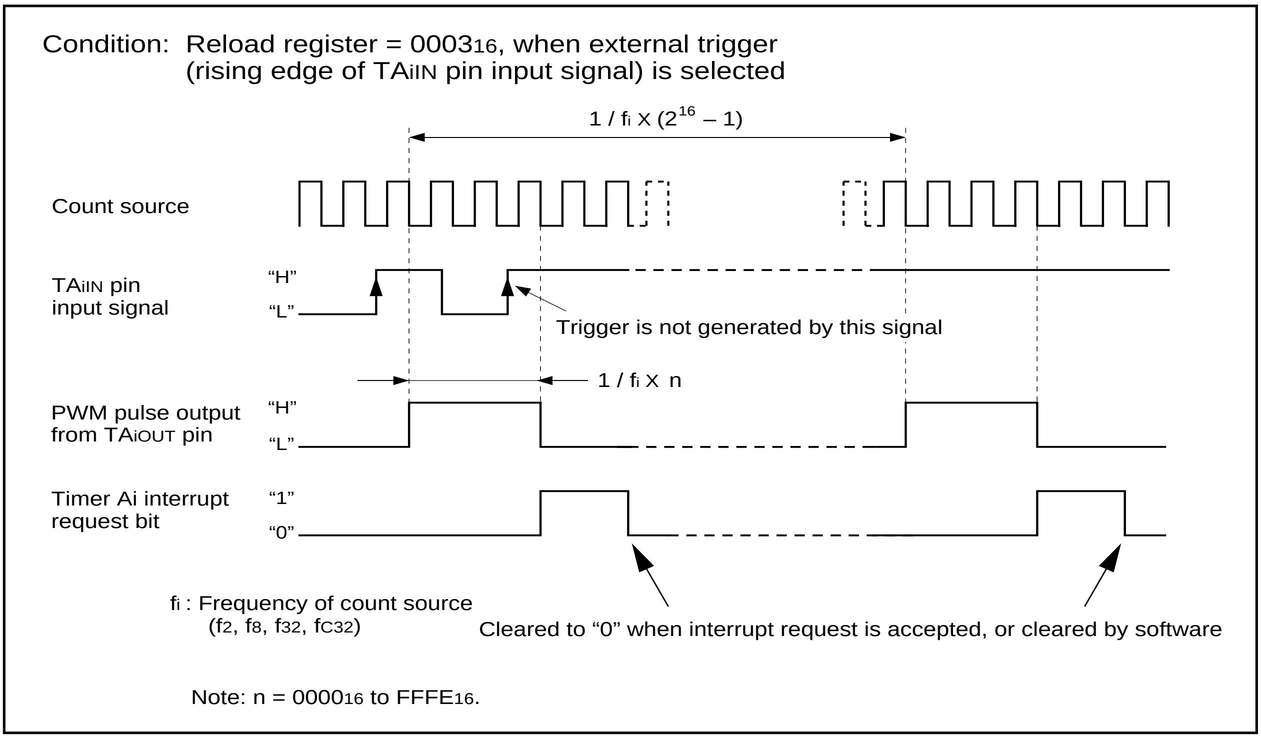


Symbol TAiMR(i=0 to 4) Address 0396_{16} to $039A_{16}$ When reset 00_{16}

Bit symbol	Bit name	Function	R	W
TMOD0	Operation mode select bit	b1 b0 1 1 : PWM mode	○	○
TMOD1			○	○
MR0	1 (Must always be "1" in PWM mode)		○	○
MR1	External trigger select bit (Note 1)	0: Falling edge of TAiIN pin's input signal (Note 2) 1: Rising edge of TAiIN pin's input signal (Note 2)	○	○
MR2	Trigger select bit	0: Count start flag is valid 1: Selected by event/trigger select register	○	○
MR3	16/8-bit PWM mode select bit	0: Functions as a 16-bit pulse width modulator 1: Functions as an 8-bit pulse width modulator	○	○
TCK0	Count source select bit	b7 b6 0 0 : f2 0 1 : f8 1 0 : f32 1 1 : fC32	○	○
TCK1			○	○

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

Figure 9-11. Timer Ai mode register in pulse width modulation mode


Condition: Reload register = 0003$_{16}$, when external trigger (rising edge of TAiIN pin input signal) is selected

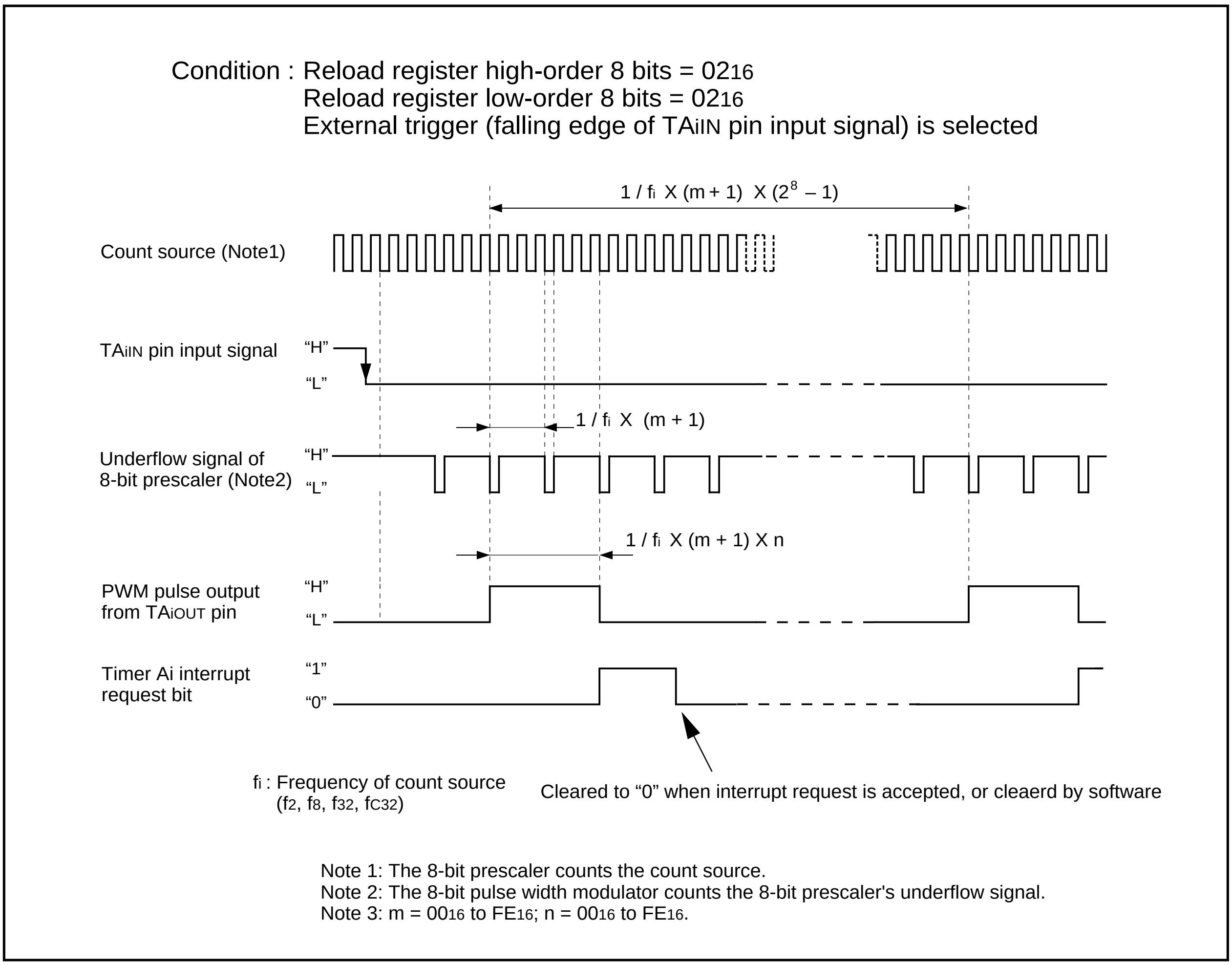
1 / fi X (2^{16} – 1)
Count source
TAiIN pin input signal
"H"
"L"
Trigger is not generated by this signal
1 / fi X n
PWM pulse output from TAiOUT pin
"H"
"L"
Timer Ai interrupt request bit
"1"
"0"

fi : Frequency of count source (f2, f8, f32, fC32)

Cleared to "0" when interrupt request is accepted, or cleared by software

Note: n = 0000$_{16}$ to FFFE$_{16}$.

Figure 9-12. Example of how a 16-bit pulse width modulator operates


Condition : Reload register high-order 8 bits = 02$_{16}$
Reload register low-order 8 bits = 02$_{16}$
External trigger (falling edge of TAiIN pin input signal) is selected

1 / fi X (m + 1) X (2^8 – 1)
Count source (Note1)
TAiIN pin input signal
"H"
"L"
1 / fi X (m + 1)
Underflow signal of 8-bit prescaler (Note2)
"H"
"L"
1 / fi X (m + 1) X n
PWM pulse output from TAiOUT pin
"H"
"L"
Timer Ai interrupt request bit
"1"
"0"

fi : Frequency of count source (f2, f8, f32, fC32)

Cleared to "0" when interrupt request is accepted, or cleaerd by software

Note 1: The 8-bit prescaler counts the count source.
Note 2: The 8-bit pulse width modulator counts the 8-bit prescaler's underflow signal.
Note 3: m = 00$_{16}$ to FE$_{16}$; n = 00$_{16}$ to FE$_{16}$.

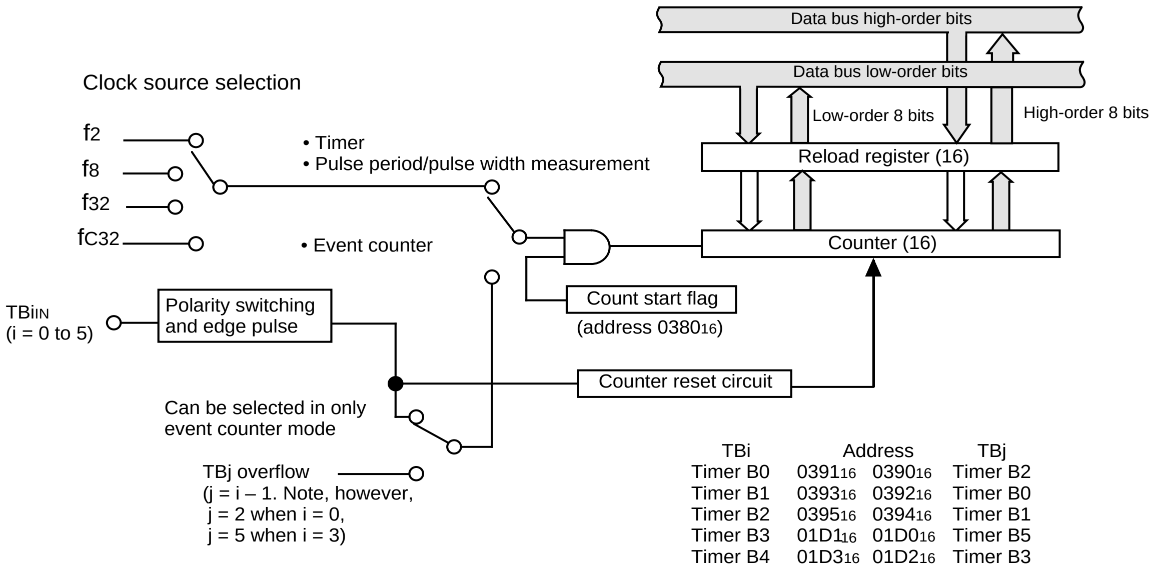
Figure 9-13. Example of how an 8-bit pulse width modulator operates

Timer B

Figure 10-1 shows the block diagram of timer B. Figures 10-2 and 10-3 show the timer B-related registers. Use the timer Bi mode register (i = 0 to 5) bits 0 and 1 to choose the desired mode.

Timer B has three operation modes listed as follows:

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



TBi	Address		TBj
Timer B0	0391_{16}	0390_{16}	Timer B2
Timer B1	0393_{16}	0392_{16}	Timer B0
Timer B2	0395_{16}	0394_{16}	Timer B1
Timer B3	$01D1_{16}$	$01D0_{16}$	Timer B5
Timer B4	$01D3_{16}$	$01D2_{16}$	Timer B3
Timer B5	$01D5_{16}$	$01D4_{16}$	Timer B4

Figure 10-1. Block diagram of timer B

Timer Bi mode register

b7 b6 b5 b4 b3 b2 b1 b0

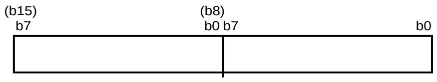
Symbol	Address	When reset
TBiMR(i = 0 to 5)	$039B_{16}$ to $039D_{16}$	$00XX0000_2$
	$01DB_{16}$ to $01DD_{16}$	$00XX0000_2$

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

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

Figure 10-2. Timer B-related registers (1)

Timer Bi register (Note)

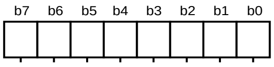


Symbol	Address	When reset
TB0	0391_{16}, 0390_{16}	Indeterminate
TB1	0393_{16}, 0392_{16}	Indeterminate
TB2	0395_{16}, 0394_{16}	Indeterminate
TB3	$01D1_{16}$, $01D0_{16}$	Indeterminate
TB4	$01D3_{16}$, $01D2_{16}$	Indeterminate
TB5	$01D5_{16}$, $01D4_{16}$	Indeterminate

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

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

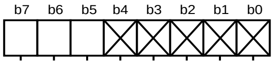
Count start flag



Symbol	Address	When reset
TABSR	0380_{16}	00_{16}

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

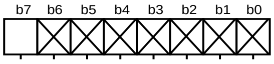
Timer B3, 4, 5 count start flag



Symbol	Address	When reset
TBSR	$01C0_{16}$	00_{16}

Bit symbol	Bit name	Function	R	W
Nothing is assigned. These bits can neither be set nor reset. When read, the value of these bits is “0”.			—	—
TB3S	Timer B3 count start flag	0 : Stops counting 1 : Starts counting	○	○
TB4S	Timer B4 count start flag		○	○
TB5S	Timer B5 count start flag		○	○

Clock prescaler reset flag



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

Bit symbol	Bit name	Function	R	W
Nothing is assigned. These bits can neither be set nor reset. When read, their contents are indeterminate.			—	—
CPSR	Clock prescaler reset flag	0 : No effect 1 : Prescaler is reset (When read, the value is “0”)	○	○

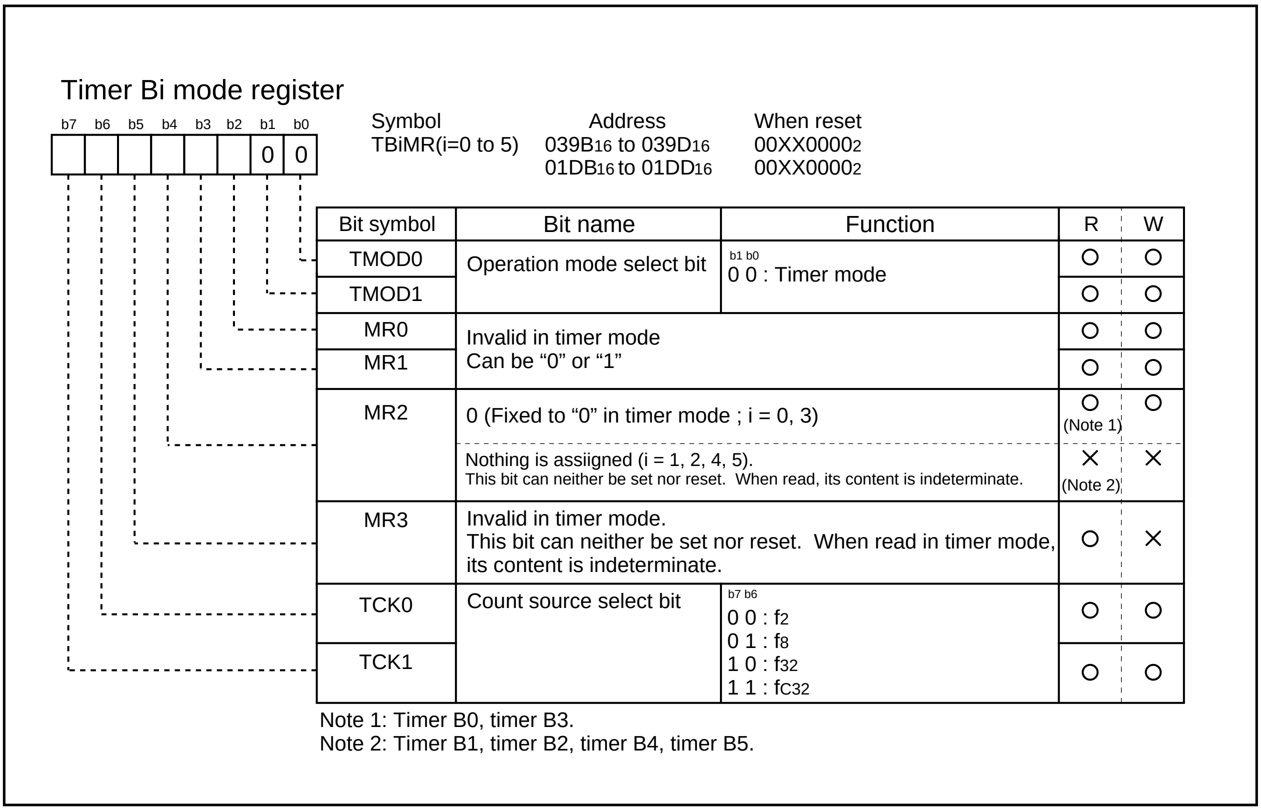
Figure 10-3. Timer B-related registers (2)

(1) Timer mode

In this mode, the timer counts an internally generated count source. (See Table 10-1.) Figure 10-4 shows the timer Bi mode register in timer mode.

Table 10-1. Timer specifications in timer mode

Item	Specification
Count source	f_2, f_8, f_{32}, fC_{32}
Count operation	• Counts down • When the timer underflows, it reloads the reload register contents before continuing counting
Divide ratio	1/(n+1) n : Set value
Count start condition	Count start flag is set (= 1)
Count stop condition	Count start flag is reset (= 0)
Interrupt request generation timing	The timer underflows
TBi_{IN} pin function	Programmable I/O port
Read from timer	Count value is read out by reading timer Bi register
Write to timer	• When counting stopped When a value is written to timer Bi register, it is written to both reload register and counter • When counting in progress When a value is written to timer Bi register, it is written to only reload register (Transferred to counter at next reload time)



Timer Bi mode register

Symbol	Address	When reset
TBiMR(i=0 to 5)	$039B_{16}$ to $039D_{16}$	$00XX0000_2$
	$01DB_{16}$ to $01DD_{16}$	$00XX0000_2$

Bit symbol	Bit name	Function	R	W
TMOD0	Operation mode select bit	b1 b0 0 0 : Timer mode	○	○
TMOD1			○	○
MR0	Invalid in timer mode Can be “0” or “1”		○	○
MR1			○	○
MR2	0 (Fixed to “0” in timer mode ; i = 0, 3)		○ (Note 1)	○
	Nothing is assiigned (i = 1, 2, 4, 5). This bit can neither be set nor reset. When read, its content is indeterminate.		× (Note 2)	×
MR3	Invalid in timer mode. This bit can neither be set nor reset. When read in timer mode, its content is indeterminate.		○	×
TCK0	Count source select bit	b7 b6 0 0 : f_2 0 1 : f_8 1 0 : f_{32} 1 1 : fC_{32}	○	○
TCK1			○	○

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

Figure 10-4. Timer Bi mode register in timer mode

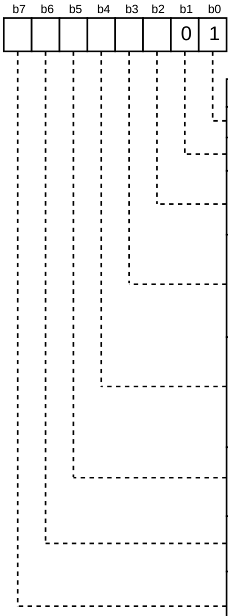
(2) Event counter mode

In this mode, the timer counts an external signal or an internal timer's overflow. (See Table 10-2.) Figure 10-5 shows the timer Bi mode register in event counter mode.

Table 10-2. Timer specifications in event counter mode

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

Timer Bi mode register



Symbol	Address	When reset
TBiMR(i=0 to 5)	039B16 to 039D16 01DB16 to 01DD16	00XX00002 00XX00002

Bit symbol	Bit name	Function	R	W
TMOD0	Operation mode select bit	b1 b0 0 1 : Event counter mode	○	○
TMOD1			○	○
MR0	Count polarity select bit (Note 1)	b3 b2 0 0 : Counts external signal's falling edges 0 1 : Counts external signal's rising edges 1 0 : Counts external signal's falling and rising edges 1 1 : Inhibited	○	○
MR1			○	○
MR2	0 (Fixed to “0” in event counter mode; i = 0, 3)		○ (Note 2)	○
	Nothing is assigned (i = 1, 2, 4, 5). This bit can neither be set nor reset. When read, its content is indeterminate.		× (Note 3)	×
MR3	Invalid in event counter mode. This bit can neither be set nor reset. When read in event counter mode, its content is indeterminate.		○	×
TCK0	Invalid in event counter mode. Can be “0” or “1”.		○	○
TCK1	Event clock select	0 : Input from TBiIN pin (Note 4) 1 : TBj overflow (j = i – 1; however, j = 2 when i = 0, j = 5 when i = 3)	○	○

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

Figure 10-5. Timer Bi mode register in event counter mode

(3) Pulse period/pulse width measurement mode

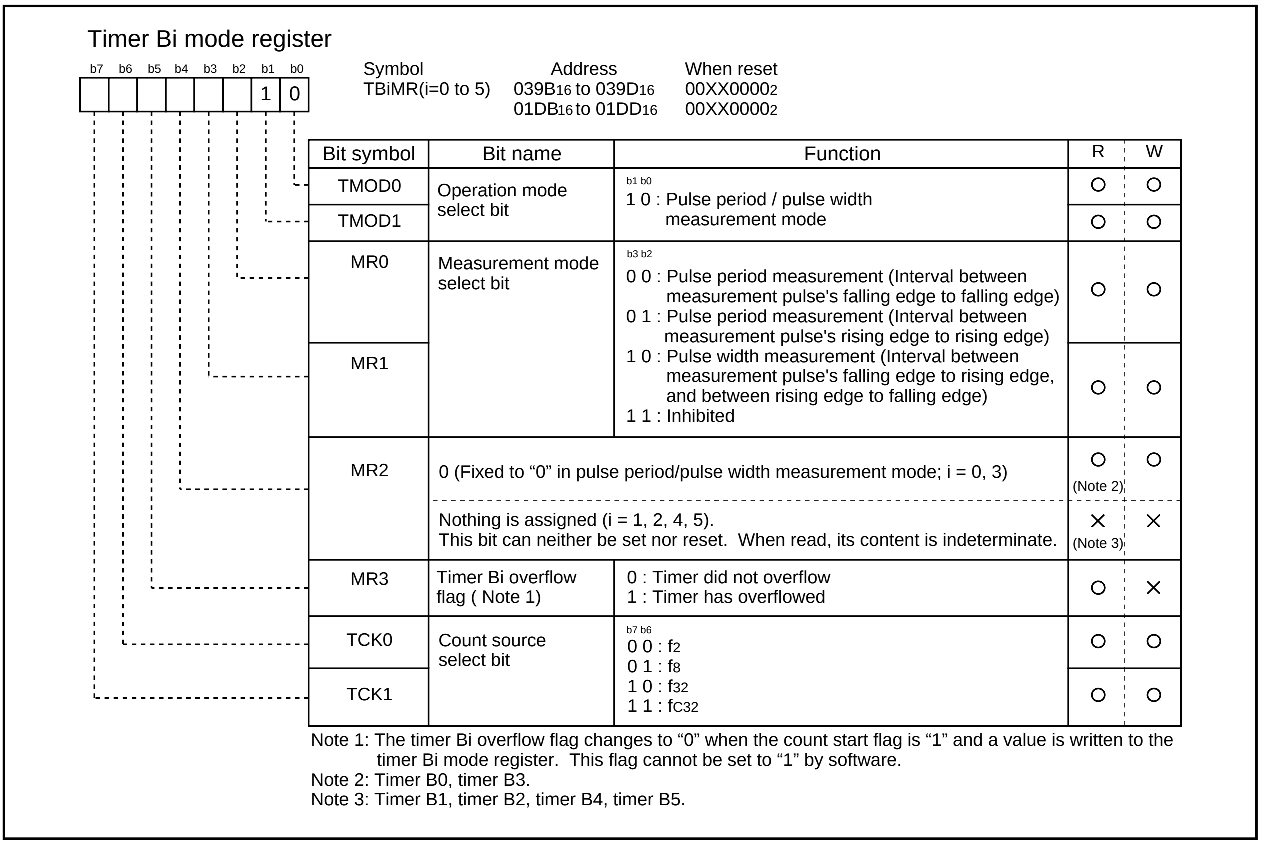
In this mode, the timer measures the pulse period or pulse width of an external signal. (See Table 10-3.) Figure 10-6 shows the timer Bi mode register in pulse period/pulse width measurement mode. Figure 10-7 shows the operation timing when measuring a pulse period. Figure 10-8 shows the operation timing when measuring a pulse width.

Table 10-3. Timer specifications in pulse period/pulse width measurement mode

Item	Specification
Count source	f_2, f_8, f_{32}, f_{C32}
Count operation	• Up count • Counter value “0000$_{16}$” is transferred to reload register at measurement pulse's effective edge and the timer continues counting
Count start condition	Count start flag is set (= 1)
Count stop condition	Count start flag is reset (= 0)
Interrupt request generation timing	• When measurement pulse's effective edge is input (Note 1) • When an overflow occurs. (Simultaneously, the timer Bi overflow flag changes to “1”. The timer Bi overflow flag changes to “0” when the count start flag is “1” and a value is written to the timer Bi mode register.)
TBi$_{IN}$ pin function	Measurement pulse input
Read from timer	When timer Bi register is read, it indicates the reload register’s content (measurement result) (Note 2)
Write to timer	Cannot be written to

Note 1: An interrupt request is not generated when the first effective edge is input after the timer has started counting.

Note 2: The value read out from the timer Bi register is indeterminate until the second effective edge is input after the timer.



Symbol	Address	When reset
TBiMR(i=0 to 5)	039B$_{16}$ to 039D$_{16}$ 01DB$_{16}$ to 01DD$_{16}$	00XX0000$_2$ 00XX0000$_2$

Bit symbol	Bit name	Function	R	W
TMOD0	Operation mode select bit	b1 b0 1 0 : Pulse period / pulse width measurement mode	○	○
TMOD1			○	○
MR0	Measurement mode select bit	b3 b2 0 0 : Pulse period measurement (Interval between measurement pulse's falling edge to falling edge) 0 1 : Pulse period measurement (Interval between measurement pulse's rising edge to rising edge) 1 0 : Pulse width measurement (Interval between measurement pulse's falling edge to rising edge, and between rising edge to falling edge) 1 1 : Inhibited	○	○
MR1			○	○
MR2	0 (Fixed to “0” in pulse period/pulse width measurement mode; i = 0, 3)		○ (Note 2)	○
	Nothing is assigned (i = 1, 2, 4, 5). This bit can neither be set nor reset. When read, its content is indeterminate.		× (Note 3)	×
MR3	Timer Bi overflow flag (Note 1)	0 : Timer did not overflow 1 : Timer has overflowed	○	×
TCK0	Count source select bit	b7 b6 0 0 : f_2 0 1 : f_8 1 0 : f_{32} 1 1 : f_{C32}	○	○
TCK1			○	○

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

Figure 10-6. Timer Bi mode register in pulse period/pulse width measurement mode

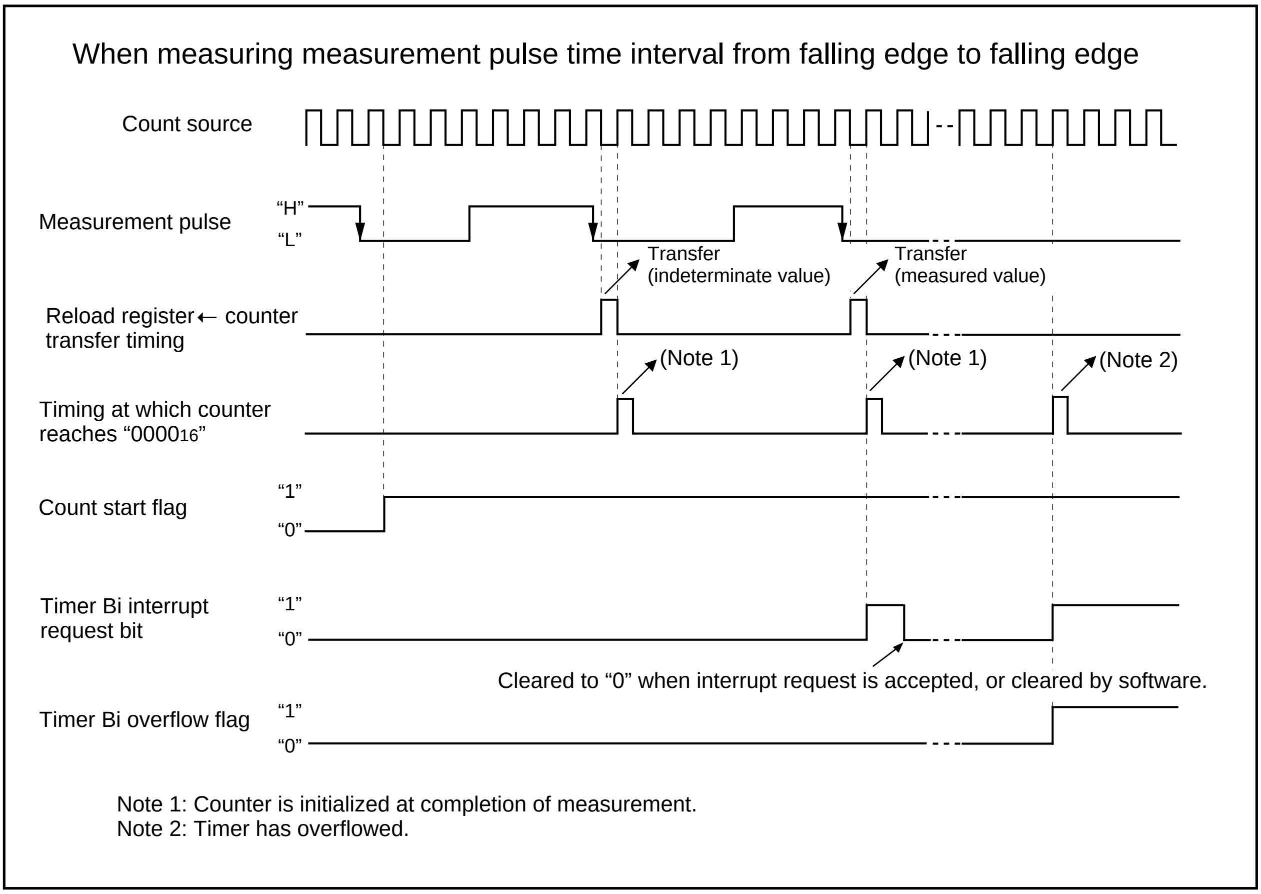

When measuring measurement pulse time interval from falling edge to falling edge
Count source
Measurement pulse
"H"
"L"
Transfer (indeterminate value)
Transfer (measured value)
Reload register ← counter transfer timing
(Note 1)
(Note 1)
(Note 2)
Timing at which counter reaches "0000₁₆"
Count start flag
"1"
"0"
Timer Bi interrupt request bit
"1"
"0"
Cleared to "0" when interrupt request is accepted, or cleared by software.
Timer Bi overflow flag
"1"
"0"
Note 1: Counter is initialized at completion of measurement.
Note 2: Timer has overflowed.

Figure 10-7. Operation timing when measuring a pulse period

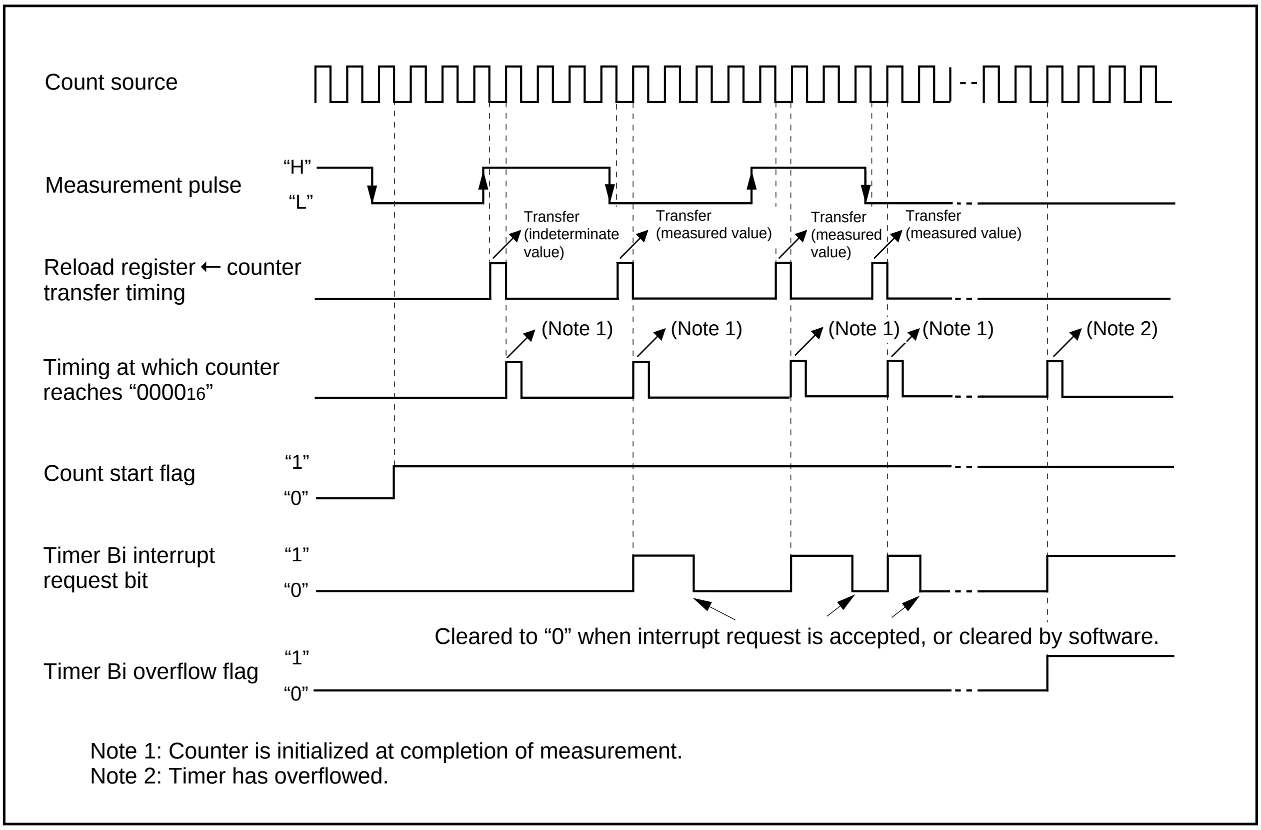

Count source
Measurement pulse
"H"
"L"
Transfer (indeterminate value)
Transfer (measured value)
Transfer (measured value)
Transfer (measured value)
Reload register ← counter transfer timing
(Note 1)
(Note 1)
(Note 1)
(Note 1)
(Note 2)
Timing at which counter reaches "0000₁₆"
Count start flag
"1"
"0"
Timer Bi interrupt request bit
"1"
"0"
Cleared to "0" when interrupt request is accepted, or cleared by software.
Timer Bi overflow flag
"1"
"0"
Note 1: Counter is initialized at completion of measurement.
Note 2: Timer has overflowed.

Figure 10-8. Operation timing when measuring a pulse width

Timers' functions for three-phase motor control

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

Figures 11-1 through 11-3 show registers related to timers for three-phase motor control.

Three-phase PWM control register 0

b7	b6	b5	b4	b3	b2	b1	b0

Symbol	Address	When reset
INVC0	01C8$_{16}$	00$_{16}$

Bit symbol	Bit name	Description	R	W
INV0$_0$	Effective interrupt output polarity select bit	0: A timer B2 interrupt occurs when the timer A1 reload control signal is 0 . 1: A timer B2 interrupt occurs when the timer A1 reload control signal is 1 . **Effective only in three-phase mode 1**	○	○
INV0$_1$	Effective interrupt output specification bit	0: Not specified. 1: Selected by the effective interrupt output polarity selection bit. **Effective only in three-phase mode 1**	○	○
INV0$_2$	Mode select bit (Note 2)	0: Normal mode 1: Three-phase PWM output mode	○	○
INV0$_3$	Output control bit	0: Output disabled 1: Output enabled	○	○
INV0$_4$	Bit to enable the function for concurrent L output disablement of positive and negative phases	0: Feature disabled 1: Feature enabled	○	○
INV0$_5$	Flag to detect concurrent L output of positive and negative phases	0: Not detected yet 1: Already detected	○	○ (Note 1)
INV0$_6$	Modulation mode select bit (Note 3)	0: Triangular wave modulation mode 1: Sawtooth wave modulation mode	○	○
INV0$_7$	Software trigger bit	1: Trigger generated The value, when read, is 0 .	○	○

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

Three-phase PWM control register 1

b7	b6	b5	b4	b3	b2	b1	b0
☒	☒	☒	0	☒			

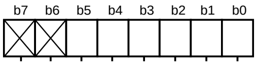
Symbol	Address	When reset
INVC1	01C9$_{16}$	00$_{16}$

Bit symbol	Bit name	Description	R	W
INV1$_0$	Timer Ai start trigger signal select bit	0: Timer B2 overflow signal 1: Timer B2 overflow signal, signal for writing to timer B2	○	○
INV1$_1$	Timer A1-1, A2-1, A4-1 control bit	0: Three-phase mode 0 1: Three-phase mode 1	○	○
INV1$_2$	Short circuit timer count source select bit (Note 1)	0 : inhibited 1 : f$_2$/2	○	○
Noting is assigned. These bits can be set nor reset. When read, their contents are indeterminate.			—	—
Reserved bit		Always set to 0	○	○
Noting is assigned. In an attempt to write to these bits, write 0 . The value, if read, turns out to be 0 .			—	—

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

Figure 11-1. Registers related to timers for three-phase motor control

Three-phase output buffer register 0

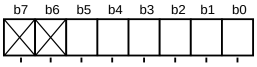


Symbol	Address	When reset
IDB0	$01CA_{16}$	00_{16}

Bit Symbol	Bit name	Function	R	W
DU0	U phase output buffer 0	Setting in U phase output buffer 0	○	○
DUB0	$\overline{\text{U}}$ phase output buffer 0	Setting in $\overline{\text{U}}$ phase output buffer 0	○	○
DV0	V phase output buffer 0	Setting in V phase output buffer 0	○	○
DVB0	$\overline{\text{V}}$ phase output buffer 0	Setting in $\overline{\text{V}}$ phase output buffer 0	○	○
DW0	W phase output buffer 0	Setting in W phase output buffer 0	○	○
DWB0	$\overline{\text{W}}$ phase output buffer 0	Setting in $\overline{\text{W}}$ phase output buffer 0	○	○
Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”.			—	—

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

Three-phase output buffer register 1



Symbol	Address	When reset
IDB1	$01CB_{16}$	00_{16}

Bit Symbol	Bit name	Function	R	W
DU1	U phase output buffer 1	Setting in U phase output buffer 1	○	○
DUB1	$\overline{\text{U}}$ phase output buffer 1	Setting in $\overline{\text{U}}$ phase output buffer 1	○	○
DV1	V phase output buffer 1	Setting in V phase output buffer 1	○	○
DVB1	$\overline{\text{V}}$ phase output buffer 1	Setting in $\overline{\text{V}}$ phase output buffer 1	○	○
DW1	W phase output buffer 1	Setting in W phase output buffer 1	○	○
DWB1	$\overline{\text{W}}$ phase output buffer 1	Setting in $\overline{\text{W}}$ phase output buffer 1	○	○
Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”.			—	—

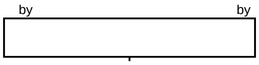
Note: When executing read instruction of this register, the contents of three-phase shift register is read out.

Dead time timer

Symbol	Address	When reset
DOT	$01CC_{16}$	Indeterminate

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

Timer B2 interrupt occurrences frequency set counter



Symbol	Address	When reset
ICTB2	$01CD_{16}$	Indeterminate

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

Note1: In setting 1 to bit 1 (INV01) - the effective interrupt output specification bit - of three-phase PWM control register 0, do not change the B2 interrupt occurrences frequency set counter to deal with the timer function for three-phase motor control.
Note2: Do not write at the timing of an overflow occurrence in timer B2.

Figure 11-2. Registers related to timers for three-phase motor control

Timer Ai register (Note)

(b15) b7 (b8) b0 b7 b0

Symbol	Address	When reset
TA1	0389_{16},0388_{16}	Indeterminate
TA2	$038B_{16}$,$038A_{16}$	Indeterminate
TA4	$038F_{16}$,$038E_{16}$	Indeterminate
TB2	0395_{16},0394_{16}	Indeterminate

Function	Values that can be set	R	W
• Timer mode Counts an internal count source	0000_{16} to $FFFF_{16}$ @	○	○
• One-shot timer mode Counts a one shot width	0000_{16} to $FFFF_{16}$	×	○

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

Timer Ai-1 register (Note)

(b15) b7 (b8) b0 b7 b0

Symbol	Address	When reset
TA11	$01C3_{16}$,$01C2_{16}$	Indeterminate
TA21	$01C5_{16}$,$01C4_{16}$	Indeterminate
TA41	$01C7_{16}$,$01C6_{16}$	Indeterminate

Function	Values that can be set	R	W
Counts an internal count source	0000_{16} to $FFFF_{16}$ @	○	○

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

Trigger select register

b7 b6 b5 b4 b3 b2 b1 b0

Symbol	Address	When reset
TRGSR	0383_{16}	00_{16}

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

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

Count start flag

b7 b6 b5 b4 b3 b2 b1 b0

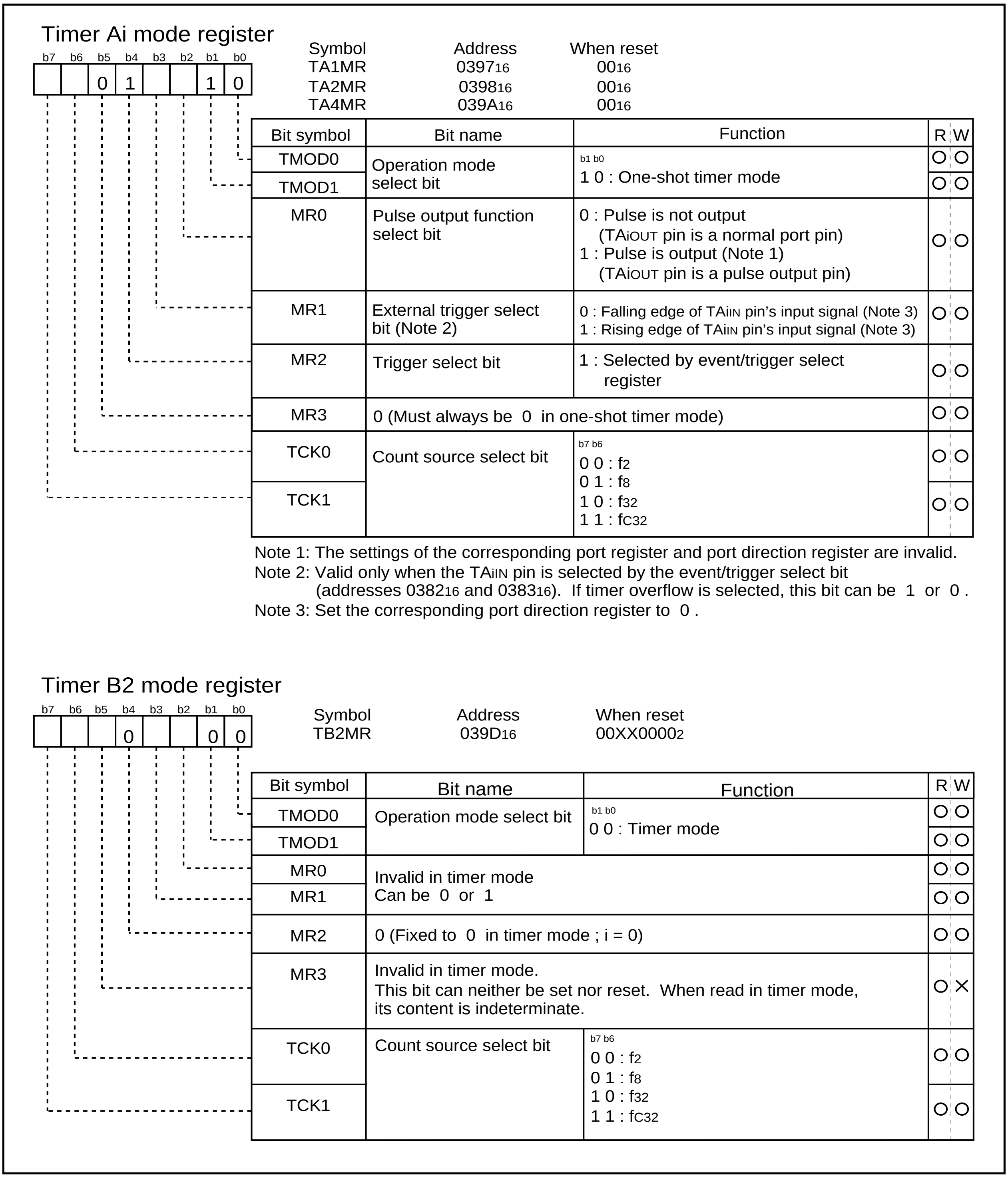
Symbol	Address	When reset
TABSR	0380_{16}	00_{16}

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

Figure 11-3. Registers related to timers for three-phase motor control

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

Setting "1" in the mode select bit - bit 2 of three-phase PWM control register 0 ($01C8_{16}$) shown in Fig. 11-1 - causes three-phase waveform mode that uses four timers A1, A2, A4, and B2 to be selected. As shown in Figure 11-4, set timers A1, A2, and A4 in one-shot timer mode, set the trigger in timer B2, and set timer B2 in timer mode using the respective timer mode registers.



Timer Ai mode register

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

Symbol	Address	When reset
TA1MR	0397_{16}	00_{16}
TA2MR	0398_{16}	00_{16}
TA4MR	$039A_{16}$	00_{16}

Bit symbol	Bit name	Function	R	W
TMOD0	Operation mode select bit	b1 b0 1 0 : One-shot timer mode	○	○
TMOD1			○	○
MR0	Pulse output function select bit	0 : Pulse is not output (TAiOUT pin is a normal port pin) 1 : Pulse is output (Note 1) (TAiOUT pin is a pulse output pin)	○	○
MR1	External trigger select bit (Note 2)	0 : Falling edge of TAiIN pin's input signal (Note 3) 1 : Rising edge of TAiIN pin's input signal (Note 3)	○	○
MR2	Trigger select bit	1 : Selected by event/trigger select register	○	○
MR3	0 (Must always be 0 in one-shot timer mode)		○	○
TCK0	Count source select bit	b7 b6 0 0 : f_2 0 1 : f_8 1 0 : f_{32} 1 1 : f_{C32}	○	○
TCK1			○	○

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

Timer B2 mode register

b7	b6	b5	b4	b3	b2	b1	b0
			0			0	0

Symbol	Address	When reset
TB2MR	$039D_{16}$	$00XX0000_2$

Bit symbol	Bit name	Function	R	W
TMOD0	Operation mode select bit	b1 b0 0 0 : Timer mode	○	○
TMOD1			○	○
MR0	Invalid in timer mode Can be 0 or 1		○	○
MR1			○	○
MR2	0 (Fixed to 0 in timer mode ; i = 0)		○	○
MR3	Invalid in timer mode. This bit can neither be set nor reset. When read in timer mode, its content is indeterminate.		○	×
TCK0	Count source select bit	b7 b6 0 0 : f_2 0 1 : f_8 1 0 : f_{32} 1 1 : f_{C32}	○	○
TCK1			○	○

Figure 11-4. Timer mode registers in three-phase waveform mode

Figure 11-5 shows the block diagram for three-phase waveform mode. In three-phase waveform mode, the positive-phase waveforms (U phase, V phase, and W phase) and negative waveforms ($\overline{U}$ phase, $\overline{V}$ phase, and $\overline{W}$ phase), six waveforms in total, are output from P8$_0$, P8$_1$, P7$_2$, P7$_3$, P7$_4$, and P7$_5$ as active on the “L” level. Of the timers used in this mode, timer A4 controls the U phase and $\overline{U}$ phase, timer A1 controls the V phase and $\overline{V}$ phase, and timer A2 controls the W phase and $\overline{W}$ phase respectively; timer B2 controls the periods of one-shot pulse output from timers A4, A1, and A2.

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

To set short circuit time, use three 8-bit timers sharing the reload register for setting dead time. A value from 1 through 255 can be set as the count of the timer for setting dead time. The timer for setting dead time works as a one-shot timer. If a value is written to the dead timer (01CC$_{16}$), the value is written to the reload register shared by the three timers for setting dead time.

Any of the timers for setting dead time takes the value of the reload register into its counter, if a start trigger comes from its corresponding timer, and performs a down count in line with the clock source selected by the dead time timer count source select bit (bit 2) of three-phase PWM control register 1 (01C9$_{16}$). The timer can receive another trigger again before the workings due to the previous trigger are completed. In this instance, the timer performs a down count from the reload register’s content after its transfer, provoked by the trigger, to the timer for setting dead time.

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

The positive waveforms (U phase, V phase, and W phase) and the negative waveforms ($\overline{U}$ phase, $\overline{V}$ phase, and $\overline{W}$ phase) in three-phase waveform mode are output from respective ports by means of setting “1” in the output control bit (bit 3) of three-phase PWM control register 0 (01C8$_{16}$). Setting “0” in this bit causes the ports to be the state of set by port direction register. This bit can be set to “0” not only by use of the applicable instruction, but by entering a falling edge in the $\overline{\text{NMI}}$ terminal or by resetting. Also, if “1” is set in the positive and negative phases concurrent L output disable function enable bit (bit4) of three-phase PWM control register 0 (01C8$_{16}$) causes one of the pairs of U phase and $\overline{U}$ phase, V phase and $\overline{V}$ phase, and W phase and $\overline{W}$ phase concurrently go to “L”, as a result, the output control bit become the state of set by port direction register.

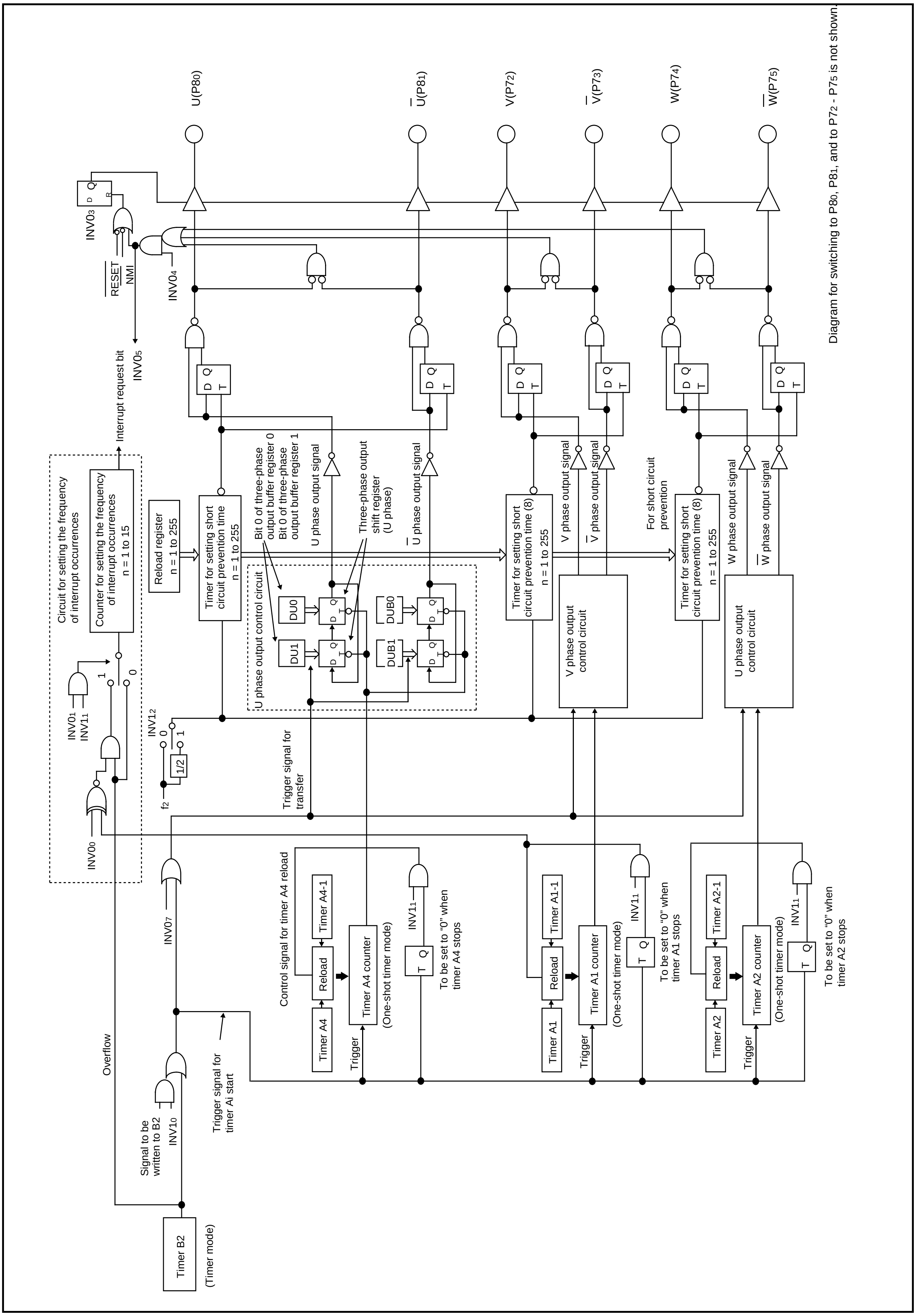

U(P8₀)
U(P8₁)
V(P7₂)
V(P7₃)
W(P7₄)
W(P7₅)
Diagram for switching to P8₀, P8₁, and to P7₂ - P7₅ is not shown.
D Q R
INV0₃
RESET
NMI
INV0₄
INV0₅
Interrupt request bit
Circuit for setting the frequency of interrupt occurrences
Counter for setting the frequency of interrupt occurrences n = 1 to 15
INV0₁
INV1₁
1
0
INV0₀
Reload register n = 1 to 255
Timer for setting short circuit prevention time n = 1 to 255
INV1₂
0
1
1/2
f₂
Bit 0 of three-phase output buffer register 0
Bit 0 of three-phase output buffer register 1
U phase output signal
Three-phase output shift register (U phase)
U phase output signal
U phase output control circuit
DU0
DU1
DUB0
DUB1
D Q T
Trigger signal for transfer
Timer for setting short circuit prevention time (8) n = 1 to 255
V phase output signal
V phase output signal
V phase output control circuit
For short circuit prevention
Timer for setting short circuit prevention time (8) n = 1 to 255
W phase output signal
W phase output signal
U phase output control circuit
INV0₇
Control signal for timer A4 reload
Timer A4-1
Reload
Timer A4
Timer A4 counter
(One-shot timer mode)
Trigger
INV1₁
T Q
To be set to "0" when timer A4 stops
Timer A1-1
Reload
Timer A1
Timer A1 counter
(One-shot timer mode)
Trigger
INV1₁
To be set to "0" when timer A1 stops
Timer A2-1
Reload
Timer A2
Timer A2 counter
(One-shot timer mode)
Trigger
INV1₁
To be set to "0" when timer A2 stops
Overflow
Trigger signal for timer Ai start
Signal to be written to B2
INV1₀
Timer B2
(Timer mode)

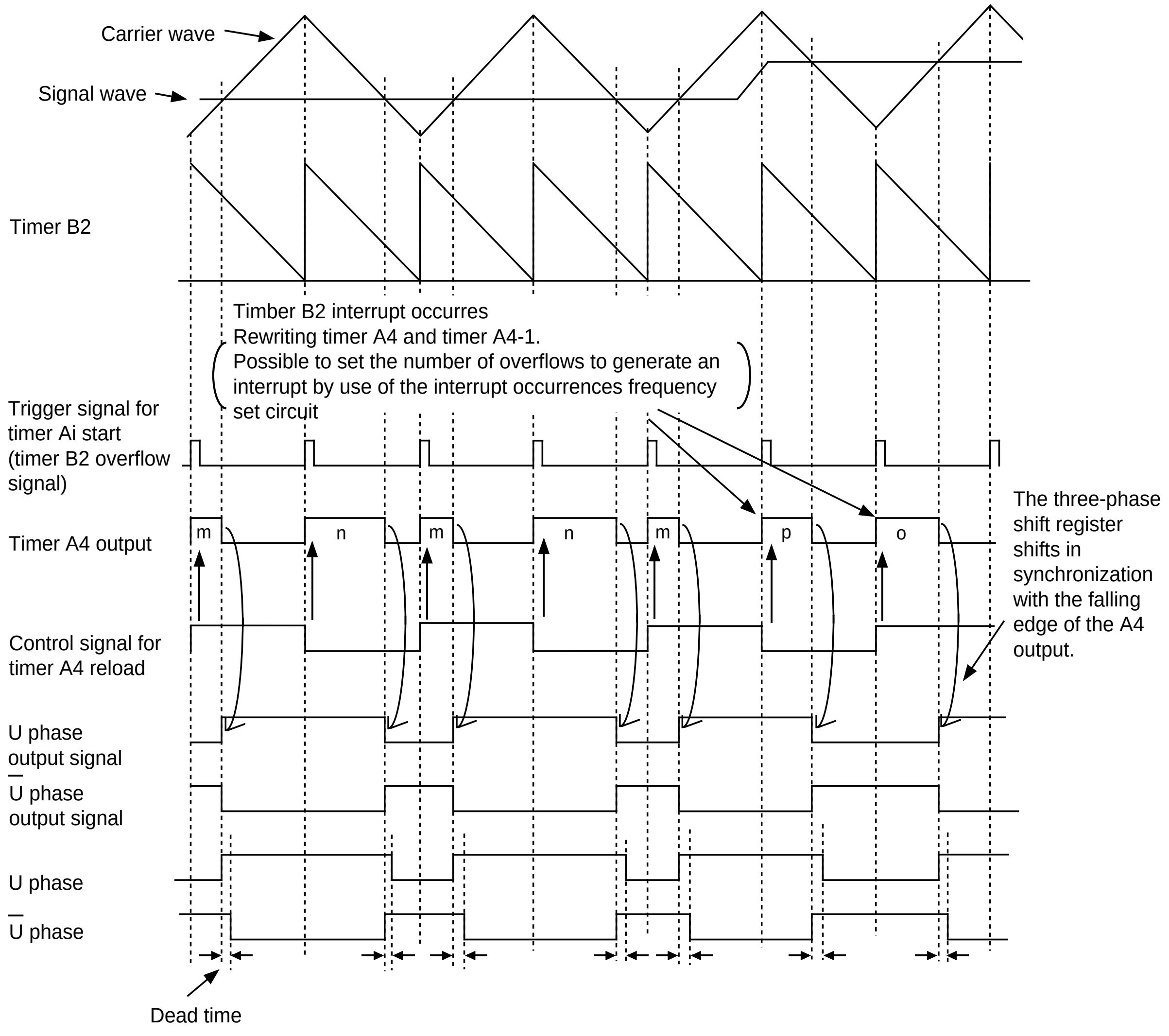
Figure 11-5. Block diagram for three-phase waveform mode

Triangular wave modulation

To generate a PWM waveform of triangular wave modulation, set "0" in the modulation mode select bit (bit 6) of three-phase PWM control register 0 (01C816). Also, set "1" in the timers A4-1, A1-1, A2-1 control bit (bit 1) of three-phase PWM control register 1 (01C916). In this mode, each of timers A4, A1, and A2 has two timer registers, and alternately reloads the timer register's content to the counter every time timer B2 counter's content becomes 000016. If "1" is set to the effective interrupt output specification bit (bit 1) of three-phase PWM control register 0 (01C816), the frequency of interrupt requests that occur every time the timer B2 counter's value becomes 000016 can be set by use of the timer B2 counter (01CD16) for setting the frequency of interrupt occurrences. The frequency of occurrences is given by (setting; setting p 0).
Setting "1" in the effective interrupt output specification bit (bit 1) of three-phase PWM control register 0 provides the means to choose which value of the timer A1 reload control signal to use, "0" or "1", to cause timer B2's interrupt request to occur. To make this selection, use the effective interrupt output polarity selection bit (bit 0) of three-phase PWM control register 0 (01C816).
An example of U phase waveform is shown in Figure 11-6, and the description of waveform output workings is given below. Set "1" in bit 0 (DU0) of three-phase output buffer register 0 (01CA16). and set "0" in bit 1 (DUB0) of the same register. In addition, set "0" in bit 0 (DU1) of three-phase output buffer register 1 (01CB16) and set "1" in bit 1 (DUB1) of the same register. Also, set "0" in the effective interrupt output specification bit (bit 1) of three-phase PWM control register 0 to set a value in the timer B2 interrupt occurence frequency set counter. By this setting, a timer B2 interrupt occurs when the timer B2 counter's content becomes 000016 as many as (setting) times. Furthermore, set "1" in the effective interrupt output specification bit (bit 1) of three-phase PWM control register 0, set in the effective interrupt polarity select bit (bit 0) of three-phase PWM control register 0 and set "1" in the interrupt occurence frequency set counter (01CD16). These settings cause a timer B2 interrupt to occur every other interval when the U phase output goes to "H".
When the timer B2 counter's content becomes 000016, timer A4 starts outputting one-shot pulses. In this instance, the content of the three-phase buffer register DU1 and that of DU0 are set in the three-phase output shift register (U phase), the content of DUB1 and that of DUB0 are set in the three-phase shift register ($\overline{\text{U}}$ phase). After triangular wave modulation mode is selected, however, no setting is made in the shift register even though the timer B2 counter's content becomes 000016.
The value of DU0 and that of DUB0 are output to the U terminal (P80) and to the $\overline{\text{U}}$ terminal (P81) respectively. When the timer A4 counter counts the value written to timer A4 (038F16, 038E16) and when timer A4 finishes outputting one-shot pulses, the three-phase shift register's content is shifted one position, and the value of DU1 and that of DUB1 are output to the U phase output signal and to $\overline{\text{U}}$ phase output signal respectively. At the same time, one-shot pulses are output from the timer for setting dead time used for setting the time over which the "L" level of the U phase waveform doesn't lap over the "L" level of the $\overline{\text{U}}$ phase waveform, which has the opposite phase of the former. The U phase waveform output that started from the "H" level keeps its level until the timer for setting dead time finishes outputting one-shot pulses even though the three-phase output shift register's content changes from "1" to "0" by the effect of the one-shot pulses. When the timer for setting dead time finishes outputting one-shot pulses, "0" already shifted in the three-phase shift register goes effective, and the U phase waveform changes to the "L" level. When the timer B2 counter's content becomes 000016, the timer A4 counter starts counting the value written to timer A4-1 (01C116, 01C016), and starts outputting one-shot pulses. When timer A4 finishes outputting one-shot pulses, the three-phase shift register's content is shifted one position, but if the three-phase output shift register's content changes from "0" to "1" as a result of the shift, the output level changes from "L" to "H" without waiting for the timer for setting dead time to finish outputting one-shot pulses. A U phase waveform

is generated by these workings repeatedly. With the exception that the three-phase output shift register on the U phase side is used, the workings in generating a U phase waveform, which has the opposite phase of the U phase waveform, are the same as in generating a U phase waveform . In this way, a waveform can be picked up from the applicable terminal in a manner in which the L level of the U phase waveform doesn’t lap over that of the U phase waveform, which has the opposite phase of the U phase waveform. The width of the “L” level too can be adjusted by varying the values of timer B2, timer A4, and timer A4-1. In dealing with the V and W phases, and $\overline{V}$ and $\overline{W}$ phases, the latter are of opposite phase of the former, have the corresponding timers work similarly to dealing with the U and $\overline{U}$ phases to generate an intended waveform.

A carrier wave of triangular waveform


Carrier wave
Signal wave
Timer B2
Trigger signal for timer Ai start (timer B2 overflow signal)
Timer A4 output
m
n
m
n
m
p
o
Control signal for timer A4 reload
U phase output signal
$\overline{U}$ phase output signal
U phase
$\overline{U}$ phase
Dead time

Timber B2 interrupt occurres
Rewriting timer A4 and timer A4-1.
Possible to set the number of overflows to generate an interrupt by use of the interrupt occurrences frequency set circuit

The three-phase shift register shifts in synchronization with the falling edge of the A4 output.

Trigger signal: set to triangular wave modulation mode and to three-phase mode 1

Figure 11-6. Timing chart of operation (1)

Assigning certain values to DU0 (bit0) of three-phase output buffer register 0 ($01CA_{16}$) and DUB0 (bit1) of the same register, and to DU1 (bit0) of three-phase output buffer register 1 ($01CB_{16}$) and DUB1 (bit1) of the same register allows you to output the waveforms as shown in the Figure 11-7, that is, to output the U phase alone, to fix $\overline{U}$ phase to "H", to fix the U phase to "H", or to output the $\overline{U}$ phase alone.

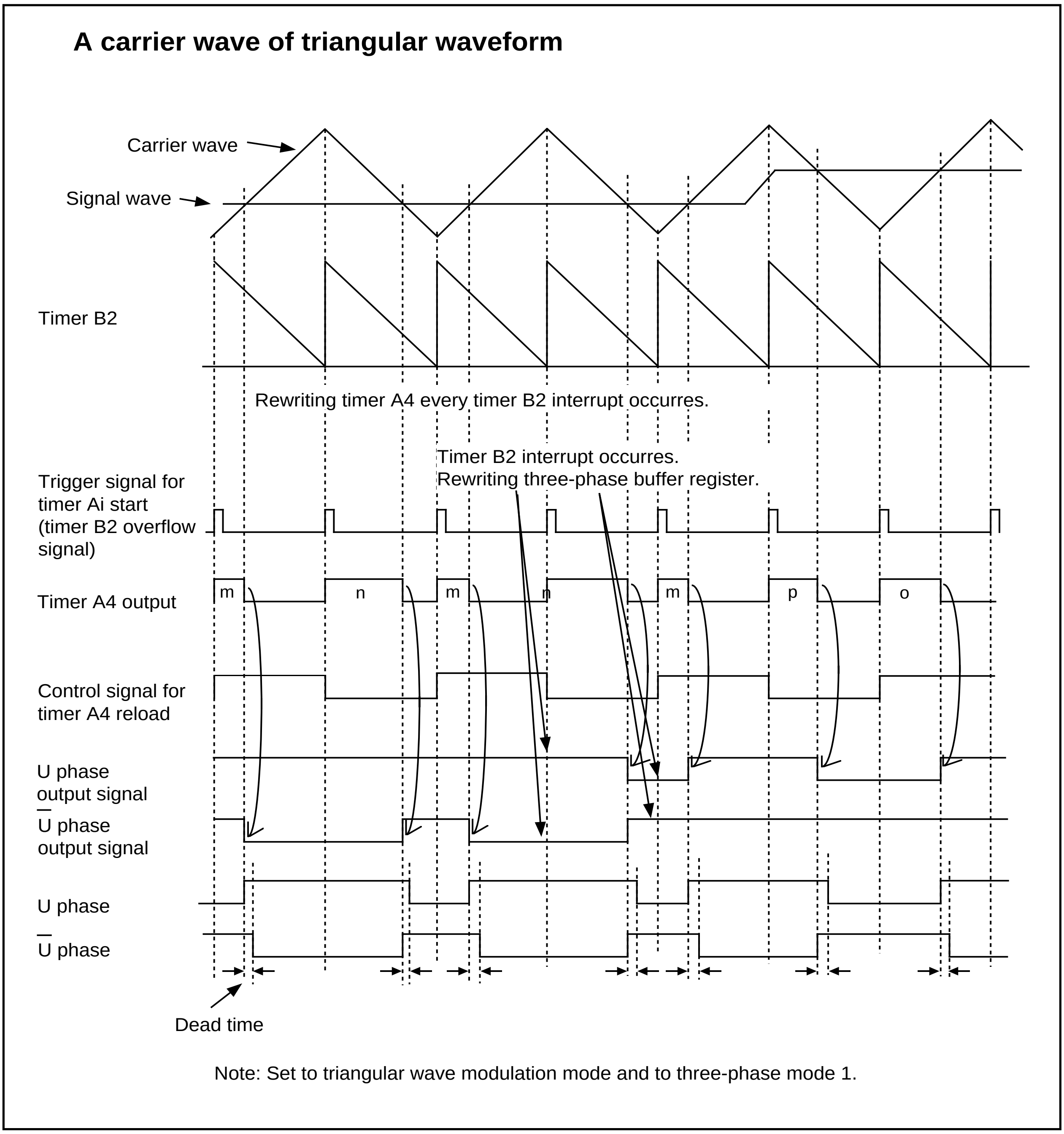

A carrier wave of triangular waveform
Carrier wave
Signal wave
Timer B2
Rewriting timer A4 every timer B2 interrupt occurres.
Timer B2 interrupt occurres.
Rewriting three-phase buffer register.
Trigger signal for timer Ai start (timer B2 overflow signal)
Timer A4 output
m
n
m
n
m
p
o
Control signal for timer A4 reload
U phase output signal
$\overline{U}$ phase output signal
U phase
$\overline{U}$ phase
Dead time
Note: Set to triangular wave modulation mode and to three-phase mode 1.

Figure 11-7. Timing chart of operation (2)

Sawtooth modulation

To generate a PWM waveform of sawtooth wave modulation, set “1” in the modulation mode select bit (bit 6) of three-phase PWM control register 0 (01C8$_{16}$). Also, set “0” in the timers A4, A1, and A2-1 control bit (bit 1) of three-phase PWM control register 1 (01C9$_{16}$). In this mode, the timer registers of timers A4, A1, and of A2 comprise conventional timers A4, A1, and A2 alone, and reload the corresponding timer register's content to the counter every time the timer B2 counter's content becomes 0000$_{16}$. The effective interrupt output specification bit (bit 1) of three-phase PWM control register 0 (01C8$_{16}$) and the effective interrupt output polarity selection bit (bit 0) turn nullified.

An example of U phase waveform is shown in Figure 11-8, and the description of waveform output workings is given below. Set “1” in bit 0 (DU0) of three-phase output buffer register 0 (01CA$_{16}$), and set “0” in bit 1 (DUB0) of the same register. In addition, set “0” in bit 0 (DU1) of three-phase output buffer register 1 (01CA$_{16}$) and set “1” in bit 1 (DUB1) of the same register.

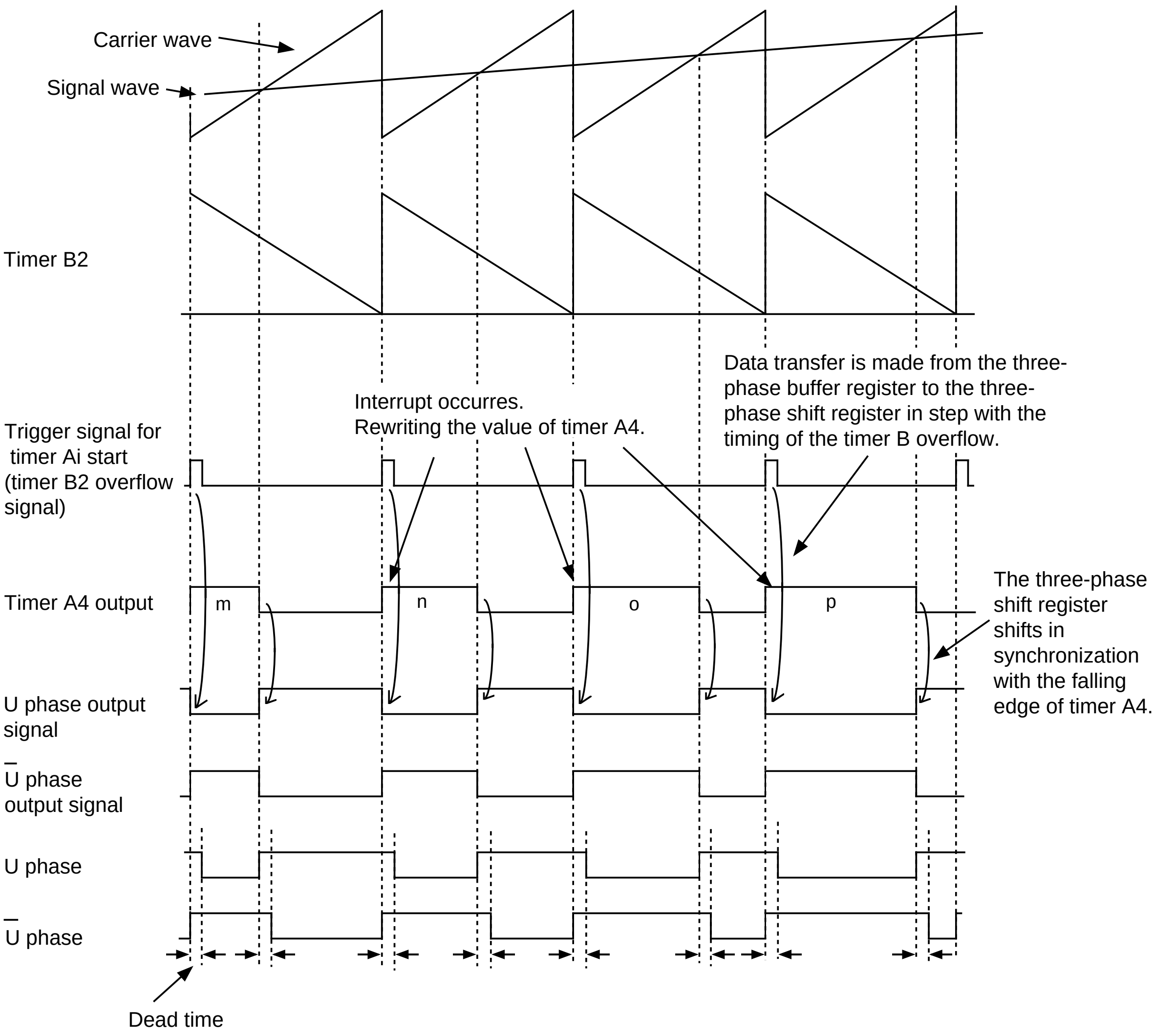
When the timber B2 counter's content becomes 0000$_{16}$, timer B2 generates an interrupt, and timer A4 starts outputting one-shot pulses at the same time. In this instance, the contents of the three-phase buffer registers DU1 and DU0 are set in the three-phase output shift register (U phase), and the contents of DUB1 and DUB0 are set in the three-phase output register (U phase). After this, the three-phase buffer register's content is set in the three-phase shift register every time the timer B2 counter's content becomes 0000$_{16}$.

The value of DU0 and that of DUB0 are output to the U terminal (P8$_0$) and to the $\overline{U}$ terminal (P8$_1$) respectively. When the timer A4 counter counts the value written to timer A4 (038F$_{16}$, 038E$_{16}$) and when timer A4 finishes outputting one-shot pulses, the three-phase output shift register's content is shifted one position, and the value of DU1 and that of DUB1 are output to the U phase output signal and to the $\overline{U}$ output signal respectively. At the same time, one-shot pulses are output from the timer for setting dead time used for setting the time over which the “L” level of the U phase waveform doesn't lap over the “L” level of the $\overline{U}$ phase waveform, which has the opposite phase of the former. The U phase waveform output that started from the “H” level keeps its level until the timer for setting dead time finishes outputting one-shot pulses even though the three-phase output shift register's content changes from “1” to “0 ”by the effect of the one-shot pulses. When the timer for setting dead time finishes outputting one-shot pulses, 0 already shifted in the three-phase shift register goes effective, and the U phase waveform changes to the “L” level. When the timer B2 counter's content becomes 0000$_{16}$, the contents of the three-phase buffer registers DU1 and DU0 are set in the three-phase shift register (U phase), and the contents of DUB1 and DUB0 are set in the three-phase shift register ($\overline{U}$ phase) again.

A U phase waveform is generated by these workings repeatedly. With the exception that the three-phase output shift register on the $\overline{U}$ phase side is used, the workings in generating a $\overline{U}$ phase waveform, which has the opposite phase of the U phase waveform, are the same as in generating a U phase waveform. In this way, a waveform can be picked up from the applicable terminal in a manner in which the “L” level of the U phase waveform doesn't lap over that of the U phase waveform, which has the opposite phase of the U phase waveform. The width of the “L” level too can be adjusted by varying the values of timer B2 and timer A4. In dealing with the V and W phases, and $\overline{V}$ and $\overline{W}$ phases, the latter are of opposite phase of the former, have the corresponding timers work similarly to dealing with the U and $\overline{U}$ phases to generate an intended waveform.

Setting “1” both in bit 1 (DUB0) of three-phase buffer register 0 (01CA$_{16}$) and in bit 1 (DUB1) of three-phase buffer register 1 (01CA$_{16}$) provides a means to output the U phase alone and to fix the $\overline{U}$ phase output to “H” as shown in Figure 11-9.

A carrier wave of sawtooth waveform


Carrier wave
Signal wave
Timer B2
Interrupt occurres.
Rewriting the value of timer A4.
Data transfer is made from the three-phase buffer register to the three-phase shift register in step with the timing of the timer B overflow.
Trigger signal for timer Ai start (timer B2 overflow signal)
Timer A4 output
m
n
o
p
The three-phase shift register shifts in synchronization with the falling edge of timer A4.
U phase output signal
$\overline{U}$ phase output signal
U phase
$\overline{U}$ phase
Dead time

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

Figure 11-8. Timing chart of operation (3)

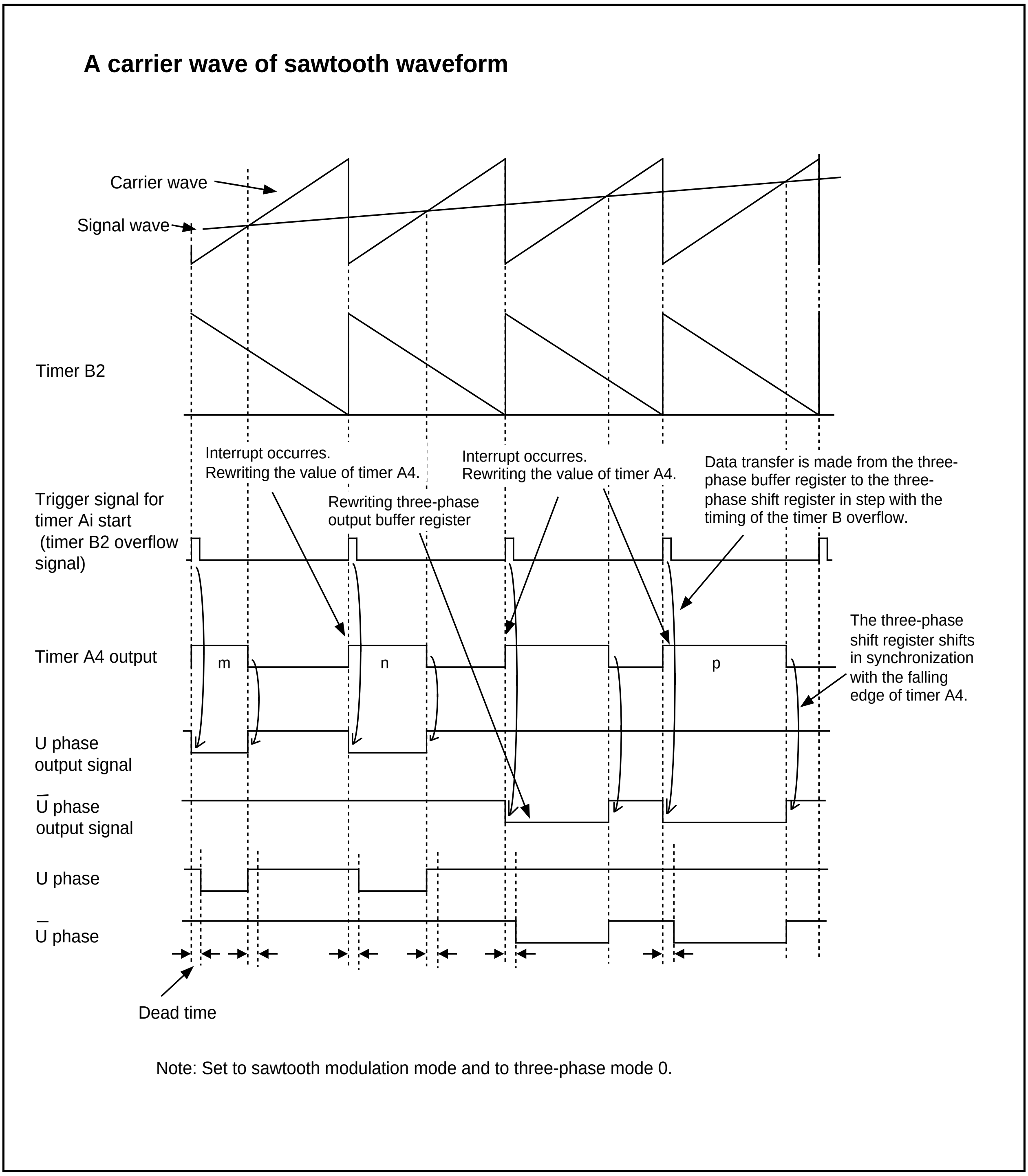

A carrier wave of sawtooth waveform
Carrier wave
Signal wave
Timer B2
Interrupt occurres.
Rewriting the value of timer A4.
Interrupt occurres.
Rewriting the value of timer A4.
Data transfer is made from the three-phase buffer register to the three-phase shift register in step with the timing of the timer B overflow.
Trigger signal for timer Ai start (timer B2 overflow signal)
Rewriting three-phase output buffer register
Timer A4 output
m
n
p
The three-phase shift register shifts in synchronization with the falling edge of timer A4.
U phase output signal
$\overline{U}$ phase output signal
U phase
$\overline{U}$ phase
Dead time
Note: Set to sawtooth modulation mode and to three-phase mode 0.

Figure 11-9. Timing chart of operation (4)

Serial I/O

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

UART0 to 2

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

Figure 12-1 shows the block diagram of UART0, UART1 and UART2. Figures 12-2 and 12-3 show the block diagram of the transmit/receive unit.

UARTi (i = 0 to 2) has two operation modes: a clock synchronous serial I/O mode and a clock asynchronous serial I/O mode (UART mode). The contents of the serial I/O mode select bits (bits 0 to 2 at addresses 03A0$_{16}$, 03A8$_{16}$ and 01F8$_{16}$) determine whether UARTi is used as a clock synchronous serial I/O or as a UART. Although a few functions are different, UART0, UART1 and UART2 have almost the same functions.

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

Table 12-1 shows the comparison of functions of UART0 through UART2, and Figures 12-4 through 12-8 show the registers related to UARTi.

Note: SIM : Subscriber Identity Module

Table 12-1. Comparison of functions of UART0 through UART2

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

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

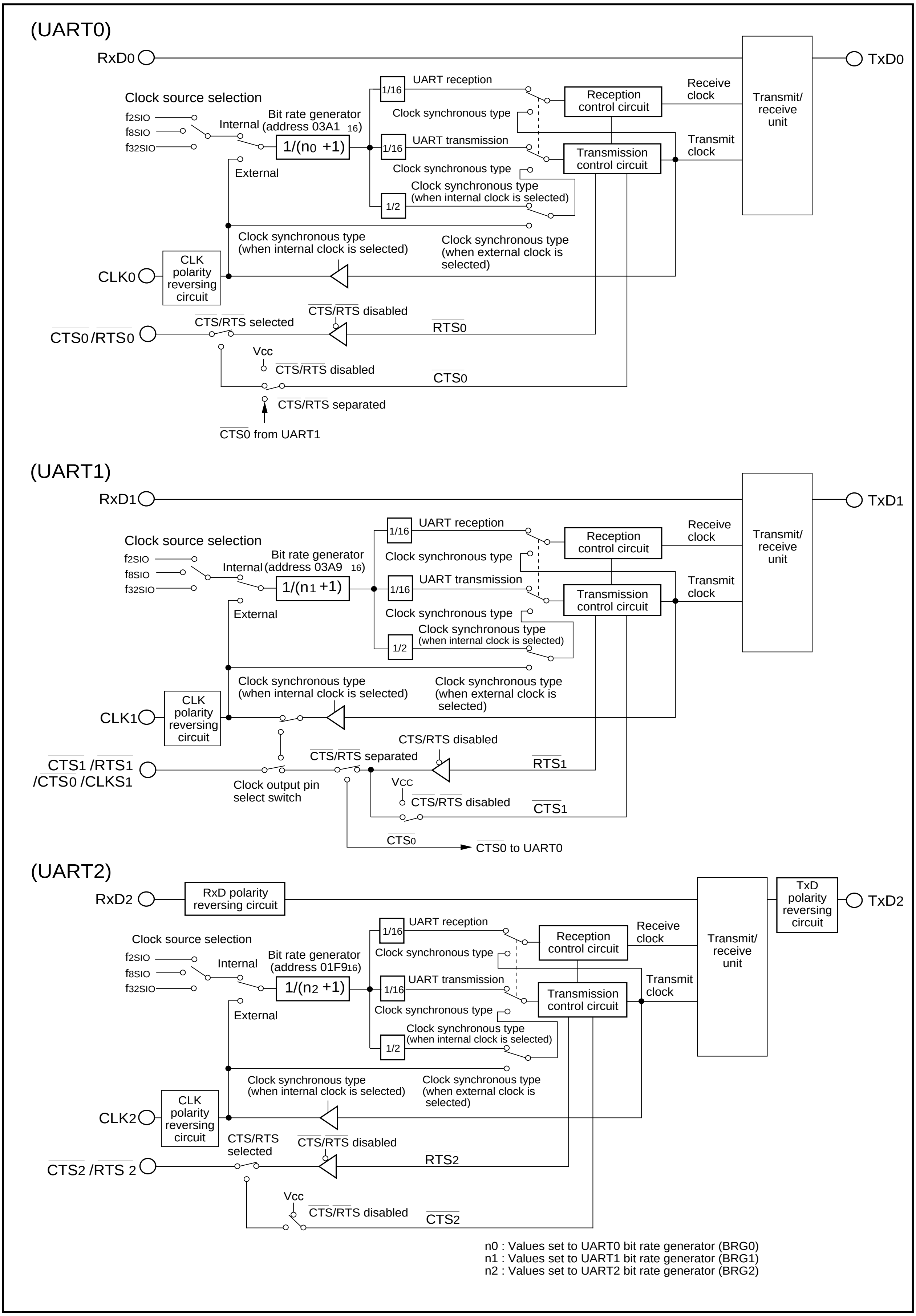

(UART0)
RxD0
TxD0
Clock source selection
f2SIO
f8SIO
f32SIO
Internal
Bit rate generator (address 03A1 16)
1/(n0 +1)
External
1/16
UART reception
Clock synchronous type
1/16
UART transmission
Clock synchronous type
1/2
Clock synchronous type (when internal clock is selected)
Reception control circuit
Transmission control circuit
Receive clock
Transmit clock
Transmit/ receive unit
CLK0
CLK polarity reversing circuit
Clock synchronous type (when internal clock is selected)
Clock synchronous type (when external clock is selected)
CTS0/RTS0
CTS/RTS selected
CTS/RTS disabled
RTS0
Vcc
CTS/RTS disabled
CTS0
CTS/RTS separated
CTS0 from UART1
(UART1)
RxD1
TxD1
Bit rate generator (address 03A9 16)
1/(n1 +1)
CLK1
CTS1 /RTS1 /CTS0 /CLKS1
Clock output pin select switch
CTS/RTS separated
CTS/RTS disabled
RTS1
CTS1
CTS0
CTS0 to UART0
(UART2)
RxD2
RxD polarity reversing circuit
TxD2
TxD polarity reversing circuit
Bit rate generator (address 01F916)
1/(n2 +1)
CLK2
CTS2 /RTS 2
CTS/RTS selected
RTS2
CTS2
n0 : Values set to UART0 bit rate generator (BRG0)
n1 : Values set to UART1 bit rate generator (BRG1)
n2 : Values set to UART2 bit rate generator (BRG2)

Figure 12-1. Block diagram of UARTi (i = 0 to 2)

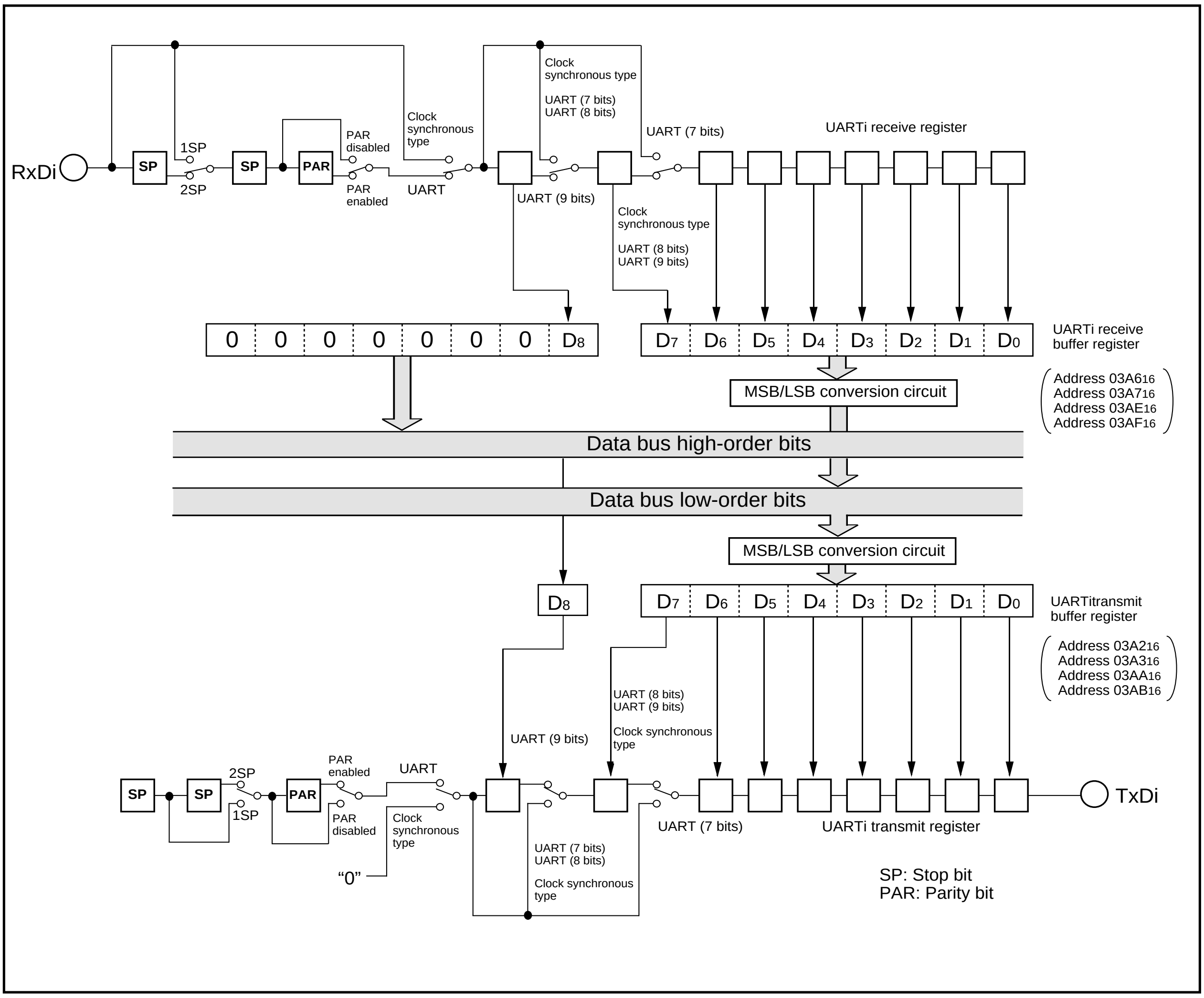


Figure 12-2. Block diagram of UARTi (i = 0, 1) transmit/receive unit

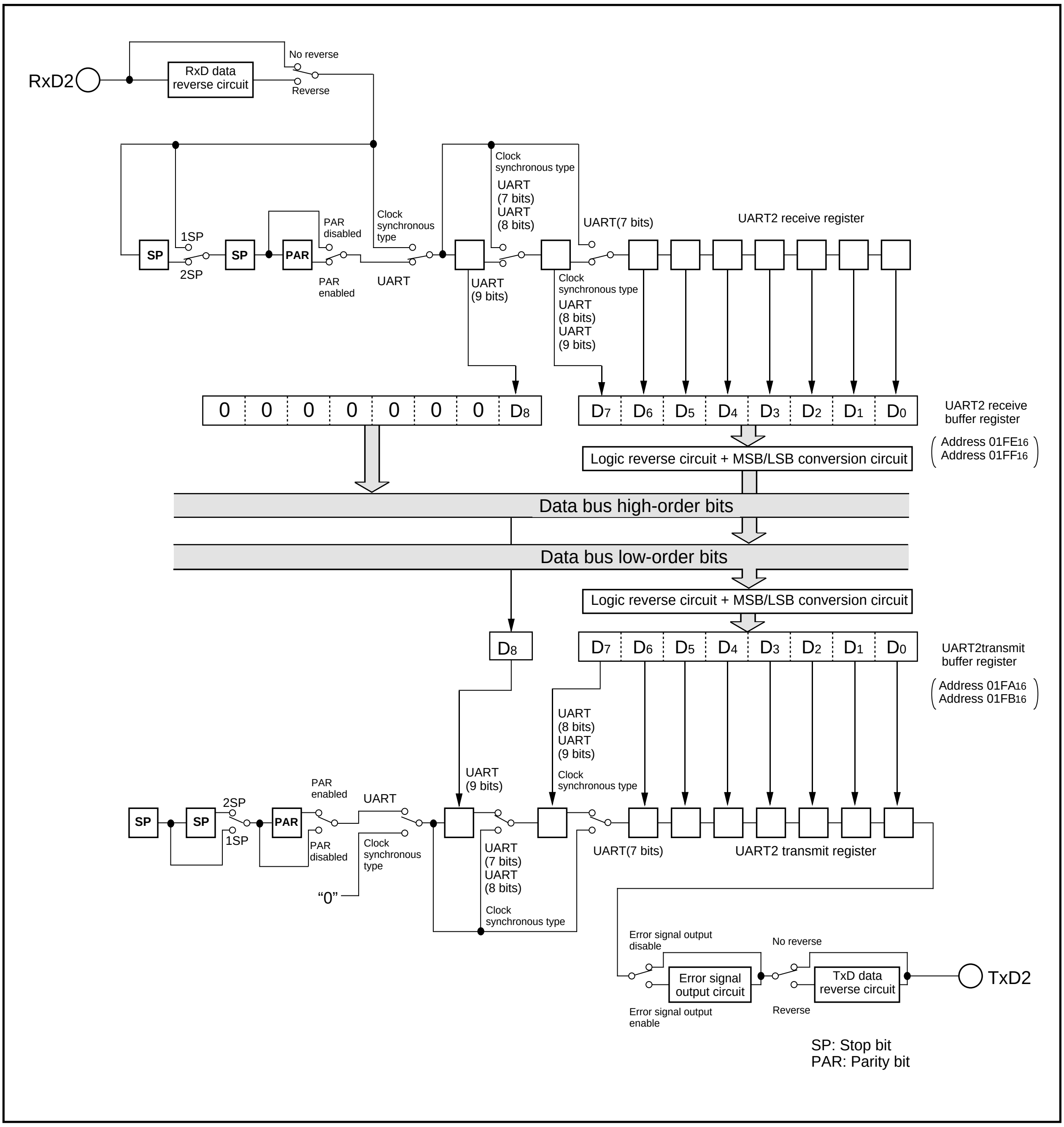
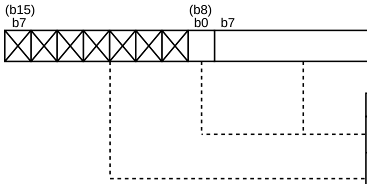


Figure 12-3. Block diagram of UART2 transmit/receive unit

UARTi transmit buffer register

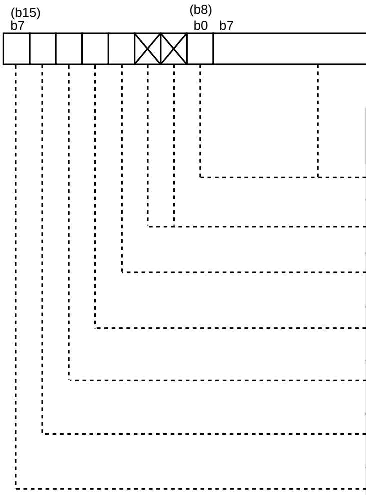


Symbol	Address	When reset
U0TB	03A3$_{16}$, 03A2$_{16}$	Indeterminate
U1TB	03AB$_{16}$, 03AA$_{16}$	Indeterminate
U2TB	01FB$_{16}$, 01FA$_{16}$	Indeterminate

Function	R	W
Transmit data (Note)	×	○
Nothing is assigned. These bits can neither be set nor reset. When read, their contents are indeterminate.	—	—

Note: Bit 8 is set to “1” when I^{2}C mode is used.

UARTi receive buffer register



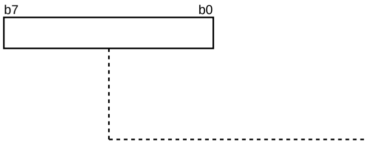
Symbol	Address	When reset
U0RB	03A7$_{16}$, 03A6$_{16}$	Indeterminate
U1RB	03AF$_{16}$, 03AE$_{16}$	Indeterminate
U2RB	01FF$_{16}$, 01FE$_{16}$	Indeterminate

Bit symbol	Bit name	Function (During clock synchronous serial I/O mode)	Function (During UART mode)	R	W
—	——	Receive data	Receive data	○	×
Nothing is assigned. These bits can neither be set nor reset. When read, the value of these bits is “0”.				—	—
ABT	Arbitration lost detecting flag (Note 2)	0 : Not detected 1 : Detected	Invalid	○	○
OER	Overrun error flag (Note 1)	0 : No overrun error 1 : Overrun error found	0 : No overrun error 1 : Overrun error found	○	×
FER	Framing error flag (Note 1)	Invalid	0 : No framing error 1 : Framing error found	○	×
PER	Parity error flag (Note 1)	Invalid	0 : No parity error 1 : Parity error found	○	×
SUM	Error sum flag (Note 1)	Invalid	0 : No error 1 : Error found	○	×

Note 1: Bits 15 through 12 are set to “0” when the serial I/O mode select bit (bits 2 to 0 at addresses 03A0$_{16}$, 03A8$_{16}$ and 01F8$_{16}$) are set to “000$_2$” or the receive enable bit is set to “0”.
(Bit 15 is set to “0” when bits 14 to 12 all are set to “0”.) Bits 14 and 13 are also set to “0” when the lower byte of the UARTi receive buffer register (addresses 03A6$_{16}$, 03AE$_{16}$ and 01FE$_{16}$) is read out.

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

UARTi bit rate generator



Symbol	Address	When reset
U0BRG	03A1$_{16}$	Indeterminate
U1BRG	03A9$_{16}$	Indeterminate
U2BRG	01F9$_{16}$	Indeterminate

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

Figure 12-4. Serial I/O-related registers (1)

UARTi transmit/receive mode register

b7 b6 b5 b4 b3 b2 b1 b0

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

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

UART2 transmit/receive mode register

b7 b6 b5 b4 b3 b2 b1 b0

Symbol	Address	When reset
U2MR	01F8$_{16}$	00$_{16}$

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

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

Figure 12-5. Serial I/O-related registers (2)

UARTi transmit/receive control register 0

b7 b6 b5 b4 b3 b2 b1 b0

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

Bit symbol	Bit name	Function (During clock synchronous serial I/O mode)	Function (During UART mode)	R	W
CLK0	BRG count source select bit	b1 b0 0 0 : f_{2SIO} is selected 0 1 : f_{8SIO} is selected 1 0 : f_{32SIO} is selected 1 1 : Inhibited	b1 b0 0 0 : f_{2SIO} is selected 0 1 : f_{8SIO} is selected 1 0 : f_{32SIO} is selected 1 1 : Inhibited	○	○
CLK1				○	○
CRS	$\overline{CTS}$/$\overline{RTS}$ function select bit	Valid when bit 4 = “0” 0 : $\overline{CTS}$ function is selected (Note 1) 1 : $\overline{RTS}$ function is selected (Note 2)	Valid when bit 4 = “0” 0 : $\overline{CTS}$ function is selected (Note 1) 1 : $\overline{RTS}$ function is selected (Note 2)	○	○
TXEPT	Transmit register empty flag	0 : Data present in transmit register (during transmission) 1 : No data present in transmit register (transmission completed)	0 : Data present in transmit register (during transmission) 1 : No data present in transmit register (transmission completed)	○	×
CRD	$\overline{CTS}$/$\overline{RTS}$ disable bit	0 : $\overline{CTS}$/$\overline{RTS}$ function enabled 1 : CTS/RTS function disabled (P6$_0$ and P6$_4$ function as programmable I/O port)	0 : $\overline{CTS}$/$\overline{RTS}$ function enabled 1 : CTS/RTS function disabled (P6$_0$ and P6$_4$ function as programmable I/O port)	○	○
NCH	Data output select bit	0 : TXDi pin is CMOS output 1 : TXDi pin is N-channel open-drain output	0: TXDi pin is CMOS output 1: TXDi pin is N-channel open-drain output	○	○
CKPOL	CLK polarity select bit	0 : Transmit data is output at falling edge of transfer clock and receive data is input at rising edge 1 : Transmit data is output at rising edge of transfer clock and receive data is input at falling edge	Must always be “0”	○	○
UFORM	Transfer format select bit	0 : LSB first 1 : MSB first	Must always be “0”	○	○

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

UART2 transmit/receive control register 0

b7 b6 b5 b4 b3 b2 b1 b0

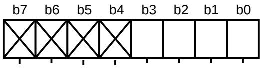
Symbol	Address	When reset
U2C0	01FC$_{16}$	08$_{16}$

Bit symbol	Bit name	Function (During clock synchronous serial I/O mode)	Function (During UART mode)	R	W
CLK0	BRG count source select bit	b1 b0 0 0 : f_{2SIO} is selected 0 1 : f_{8SIO} is selected 1 0 : f_{32SIO} is selected 1 1 : Inhibited	b1 b0 0 0 : f_{2SIO} is selected 0 1 : f_{8SIO} is selected 1 0 : f_{32SIO} is selected 1 1 : Inhibited	○	○
CLK1				○	○
CRS	$\overline{CTS}$/$\overline{RTS}$ function select bit	Valid when bit 4 = “0” 0 : $\overline{CTS}$ function is selected (Note 1) 1 : $\overline{RTS}$ function is selected (Note 2)	Valid when bit 4 = “0” 0 : $\overline{CTS}$ function is selected (Note 1) 1 : $\overline{RTS}$ function is selected (Note 2)	○	○
TXEPT	Transmit register empty flag	0 : Data present in transmit register (during transmission) 1 : No data present in transmit register (transmission completed)	0 : Data present in transmit register (during transmission) 1 : No data present in transmit register (transmission completed)	○	×
CRD	$\overline{CTS}$/$\overline{RTS}$ disable bit	0 : $\overline{CTS}$/$\overline{RTS}$ function enabled 1 : CTS/RTS function disabled (P7$_3$ functions programmable I/O port)	0 : $\overline{CTS}$/$\overline{RTS}$ function enabled 1 : CTS/RTS function disabled (P7$_3$ functions programmable I/O port)	○	○
NCH	Data output select bit	0 : TXD$_2$/RXD$_2$ pin is CMOS output 1 : TXD$_2$/RXD$_2$ pin is N-channel open-drain output (Note 4)	0 : TXD$_2$/RXD$_2$ pin is CMOS output 1 : TXD$_2$/RXD$_2$ pin is N-channel open-drain output	○	○
CKPOL	CLK polarity select bit	0 : Transmit data is output at falling edge of transfer clock and receive data is input at rising edge 1 : Transmit data is output at rising edge of transfer clock and receive data is input at falling edge	Must always be “0”	○	○
UFORM	Transfer format select bit (Note 3)	0 : LSB first 1 : MSB first	0 : LSB first 1 : MSB first	○	○

Note 1: Set the corresponding port direction register to “0”.
Note 2: The settings of the corresponding port register and port direction register are invalid.
Note 3: Only clock synchronous serial I/O mode and 8-bit UART mode are valid.
Note 4: For Flash chips P7$_1$ can only be Nch open drain output!

Figure 12-6. Serial I/O-related registers (3)

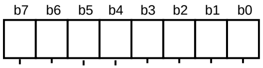
UARTi transmit/receive control register 1



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

Bit symbol	Bit name	Function (During clock synchronous serial I/O mode)	Function (During UART mode)	R	W
TE	Transmit enable bit	0 : Transmission disabled 1 : Transmission enabled	0 : Transmission disabled 1 : Transmission enabled	○	○
TI	Transmit buffer empty flag	0 : Data present in transmit buffer register 1 : No data present in transmit buffer register	0 : Data present in transmit buffer register 1 : No data present in transmit buffer register	○	×
RE	Receive enable bit	0 : Reception disabled 1 : Reception enabled	0 : Reception disabled 1 : Reception enabled	○	○
RI	Receive complete flag	0 : No data present in receive buffer register 1 : Data present in receive buffer register	0 : No data present in receive buffer register 1 : Data present in receive buffer register	○	×
	Nothing is assigned. These bits can neither be set nor reset. When read, the value of these bits is “0”.			—	—

UART2 transmit/receive control register 1



Symbol: U2C1
Address: 01FD$_{16}$
When reset: 02$_{16}$

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

Figure 12-7. Serial I/O-related registers (4)

UART transmit/receive control register 2

b7 b6 b5 b4 b3 b2 b1 b0

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

Bit symbol	Bit name	Function (During clock synchronous serial I/O mode)	Function (During UART mode)	R	W
U0IRS	UART0 transmit interrupt cause select bit	0 : Transmit buffer empty (TI = 1) 1 : Transmission completed (TXEPT = 1)	0 : Transmit buffer empty (TI = 1) 1 : Transmission completed (TXEPT = 1)	○	○
U1IRS	UART1 transmit interrupt cause select bit	0 : Transmit buffer empty (TI = 1) 1 : Transmission completed (TXEPT = 1)	0 : Transmit buffer empty (TI = 1) 1 : Transmission completed (TXEPT = 1)	○	○
U0RRM	UART0 continuous receive mode enable bit	0 : Continuous receive mode disabled 1 : Continuous receive mode enable	Invalid	○	○
U1RRM	UART1 continuous receive mode enable bit	0 : Continuous receive mode disabled 1 : Continuous receive mode enabled	Invalid	○	○
CLKMD0	CLK/CLKS select bit 0	Valid when bit 5 = “1” 0 : Clock output to CLK1 1 : Clock output to CLKS1	Invalid	○	○
CLKMD1	CLK/CLKS select bit 1 (Note)	0 : Normal mode (CLK output is CLK1 only) 1 : Transfer clock output from multiple pins function selected	Must always be “0”	○	○
RCSP	Separate $\overline{\text{CTS}}$/RTS bit	0 : $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ shared pin 1 : CTS/RTS separated	0 : $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ shared pin 1 : $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ separated	○	○
Nothing is assigned. This bit can neither be set nor reset. When read, its content is indeterminate.				—	—

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

UART2 special mode register

b7 b6 b5 b4 b3 b2 b1 b0

Symbol	Address	When reset
U2SMR	01F7$_{16}$	00$_{16}$

Bit symbol	Bit name	Function (During clock synchronous serial I/O mode)	Function (During UART mode)	R	W
IICM	IIC mode selection bit	0 : Normal mode 1 : IIC mode	Must always be “0”	○	○
ABC	Arbitration lost detecting flag control bit	0 : Update per bit 1 : Update per byte	Must always be “0”	○	○
BBS	Bus busy flag	0 : STOP condition detected 1 : START condition detected	Must always be “0”	○	○ (Note)
LSYN	SCLL sync output enable bit	0 : Disabled 1 : Enabled	Must always be “0”	○	○
ABSCS	Bus collision detect sampling clock select bit	Must always be “0”	0 : Rising edge of transfer clock 1 : Underflow signal of timer A0	○	○
ACSE	Auto clear function select bit of transmit enable bit	Must always be “0”	0 : No auto clear function 1 : Auto clear at occurrence of bus collision	○	○
SSS	Transmit start condition select bit	Must always be “0”	0 : Ordinary 1 : Falling edge of RxD2	○	○
Nothing is assigned. This bit can neither be set nor reset. When read, its content is indeterminate.				—	—

Note: Nothing but "0" may be written.

Figure 12-8. Serial I/O-related registers (5)

(1) Clock synchronous serial I/O mode

The clock synchronous serial I/O mode uses a transfer clock to transmit and receive data. Tables 12-2 and 12-3 list the specifications of the clock synchronous serial I/O mode. Figure 12-9 shows the UARTi transmit/receive mode register.

Table 12-2. Specifications of clock synchronous serial I/O mode (1)

Item	Specification
Transfer data format	• Transfer data length: 8 bits
Transfer clock	• When internal clock is selected (bit 3 at addresses $03A0_{16}$, $03A8_{16}$, $01F8_{16}$ = “0”) : fi/ 2(n+1) (Note 1) fi = f_{2SIO}, f_{8SIO}, f_{32SIO} • When external clock is selected (bit 3 at addresses $03A0_{16}$, $03A8_{16}$, $01F8_{16}$ = “1”) : Input from CLKi pin (Note 2)
Transmission/reception control	• $\overline{\text{CTS}}$ function/$\overline{\text{RTS}}$ function/$\overline{\text{CTS}}$, $\overline{\text{RTS}}$ function chosen to be invalid
Transmission start condition	• To start transmission, the following requirements must be met: - Transmit enable bit (bit 0 at addresses $03A5_{16}$, $03AD_{16}$, $01FD_{16}$) = “1” - Transmit buffer empty flag (bit 1 at addresses $03A5_{16}$, $03AD_{16}$, $01FD_{16}$) = “0” - When $\overline{\text{CTS}}$ function selected, $\overline{\text{CTS}}$ input level = “L” • Furthermore, if external clock is selected, the following requirements must also be met: - CLKi polarity select bit (bit 6 at addresses $03A4_{16}$, $03AC_{16}$, $01FC_{16}$) = “0”: CLKi input level = “H” - CLKi polarity select bit (bit 6 at addresses $03A4_{16}$, $03AC_{16}$, $01FC_{16}$) = “1”: CLKi input level = “L”
Reception start condition	• To start reception, the following requirements must be met: - Receive enable bit (bit 2 at addresses $03A5_{16}$, $03AD_{16}$, $01FD_{16}$) = “1” - Transmit enable bit (bit 0 at addresses $03A5_{16}$, $03AD_{16}$, $01FD_{16}$) = “1” - Transmit buffer empty flag (bit 1 at addresses $03A5_{16}$, $03AD_{16}$, $01FD_{16}$) = “0” • Furthermore, if external clock is selected, the following requirements must also be met: - CLKi polarity select bit (bit 6 at addresses $03A4_{16}$, $03AC_{16}$, $01FC_{16}$) = “0”: CLKi input level = “H” - CLKi polarity select bit (bit 6 at addresses $03A4_{16}$, $03AC_{16}$, $01FC_{16}$) = “1”: CLKi input level = “L”
Interrupt request generation timing	• When transmitting - Transmit interrupt cause select bit (bits 0, 1 at address $03B0_{16}$, bit 4 at address $01FD_{16}$) = “0”: Interrupts requested when data transfer from UARTi transfer buffer register to UARTi transmit register is completed - Transmit interrupt cause select bit (bits 0, 1 at address $03B0_{16}$, bit 4 at address $01FD_{16}$) = “1”: Interrupts requested when data transmission from UARTi transfer register is completed • When receiving - Interrupts requested when data transfer from UARTi receive register to UARTi receive buffer register is completed
Error detection	• Overrun error (Note 3) This error occurs when the next data is ready before contents of UARTi receive buffer register are read out

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

Note 2: Maximum 5 Mbps.

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

Table 12-3. Specifications of clock synchronous serial I/O mode (2)

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

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

UARTi transmit/receive mode registers

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

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

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

UART2 transmit/receive mode register

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

Symbol	Address	When reset
U2MR	01F8$_{16}$	00$_{16}$

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

Note: Usually set to “0”.

Figure 12-9. UARTi transmit/receive mode register in clock synchronous serial I/O mode

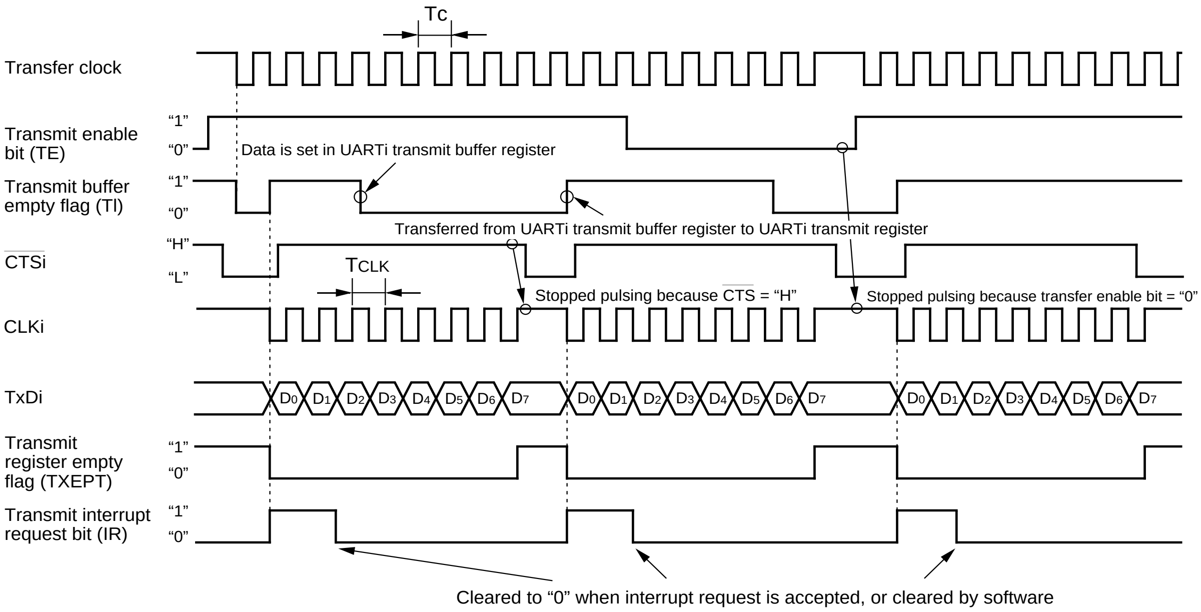
Table 12-4 lists the functions of the input/output pins during clock synchronous serial I/O mode. This table shows the pin functions when the transfer clock output from multiple pins and the separate $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ pins functions are not selected. Note that for a period from when the UARTi operation mode is selected to when transfer starts, the TxDi pin outputs a “H”. (If the N-channel open-drain is selected, this pin is in floating state.)

Table 12-4. Input/output pin functions in clock synchronous serial I/O mode

Pin name	Function	Method of selection
TxDi ($P6_3$, $P6_7$, $P7_0$)	Serial data output	(Outputs dummy data when performing reception only)
RxDi ($P6_2$, $P6_6$, $P7_1$)	Serial data input	Port $P6_2$, $P6_6$ and $P7_1$ direction register (bits 2 and 6 at address 03EE$_{16}$, bit 1 at address 03EF$_{16}$)= “0” (Can be used as an input port when performing transmission only)
CLKi ($P6_1$, $P6_5$, $P7_2$)	Transfer clock output	Internal/external clock select bit (bit 3 at address 03A0$_{16}$, 03A8$_{16}$, 01F8$_{16}$) = “0”
	Transfer clock input	Internal/external clock select bit (bit 3 at address 03A0$_{16}$, 03A8$_{16}$, 01F8$_{16}$) = “1” Port $P6_1$, $P6_5$ and $P7_2$ direction register (bits 1 and 5 at address 03EE$_{16}$, bit 2 at address 03EF$_{16}$) = “0”
$\overline{\text{CTSi}}$/$\overline{\text{RTSi}}$ ($P6_0$, $P6_4$, $P7_3$)	$\overline{\text{CTS}}$ input	$\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit (bit 4 at address 03A4$_{16}$, 03AC$_{16}$, 01FC$_{16}$) =“0” CTS/RTS function select bit (bit 2 at address 03A4$_{16}$, 03AC$_{16}$, 01FC$_{16}$) = “0” Port $P6_0$, $P6_4$ and $P7_3$ direction register (bits 0 and 4 at address 03EE$_{16}$, bit 3 at address 03EF$_{16}$) = “0”
	$\overline{\text{RTS}}$ output	$\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit (bit 4 at address 03A4$_{16}$, 03AC$_{16}$, 01FC$_{16}$) = “0” $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ function select bit (bit 2 at address 03A4$_{16}$, 03AC$_{16}$, 01FC$_{16}$) = “1”
	Programmable I/O port	$\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit (bit 4 at address 03A4$_{16}$, 03AC$_{16}$, 01FC$_{16}$) = “1”

(when transfer clock output from multiple pins and separate CTS/RTS pins functions are not selected)

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


Tc
Transfer clock
Transmit enable bit (TE)
"1"
"0"
Data is set in UARTi transmit buffer register
Transmit buffer empty flag (TI)
"1"
"0"
Transferred from UARTi transmit buffer register to UARTi transmit register
CTSi
"H"
"L"
TCLK
Stopped pulsing because CTS = "H"
Stopped pulsing because transfer enable bit = "0"
CLKi
TxDi
D0 D1 D2 D3 D4 D5 D6 D7
D0 D1 D2 D3 D4 D5 D6 D7
D0 D1 D2 D3 D4 D5 D6 D7
Transmit register empty flag (TXEPT)
"1"
"0"
Transmit interrupt request bit (IR)
"1"
"0"
Cleared to "0" when interrupt request is accepted, or cleared by software

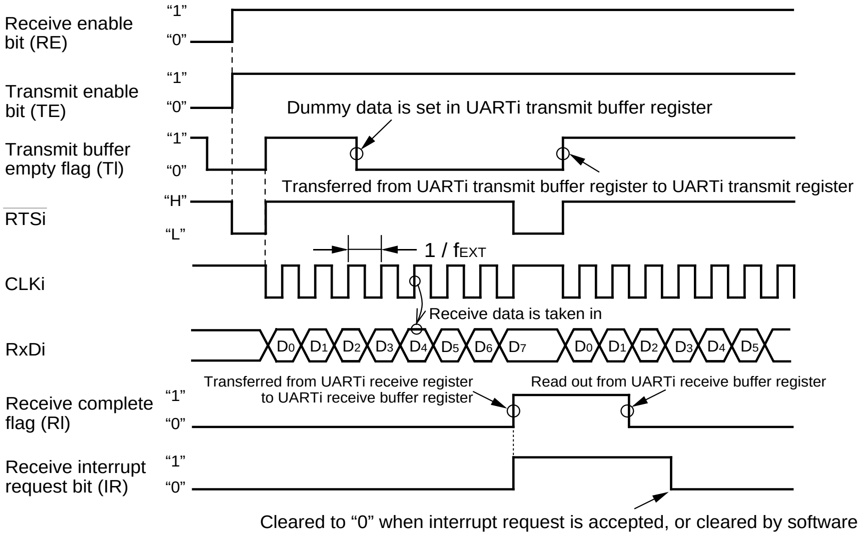
Shown in ( ) are bit symbols.

The above timing applies to the following settings:

- Internal clock is selected.
- CTS function is selected.
- CLK polarity select bit = "0".
- Transmit interrupt cause select bit = "0".

Tc = TCLK = 2(n + 1) / fi
fi: frequency of BRGi count source (f2, f8, f32)
n: value set to BRGi

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


Receive enable bit (RE)
"1"
"0"
Transmit enable bit (TE)
"1"
"0"
Dummy data is set in UARTi transmit buffer register
Transmit buffer empty flag (TI)
"1"
"0"
Transferred from UARTi transmit buffer register to UARTi transmit register
RTSi
"H"
"L"
1 / fEXT
CLKi
Receive data is taken in
RxDi
D0 D1 D2 D3 D4 D5 D6 D7
D0 D1 D2 D3 D4 D5
Transferred from UARTi receive register to UARTi receive buffer register
Read out from UARTi receive buffer register
Receive complete flag (RI)
"1"
"0"
Receive interrupt request bit (IR)
"1"
"0"
Cleared to "0" when interrupt request is accepted, or cleared by software

Shown in ( ) are bit symbols.

The above timing applies to the following settings:

- External clock is selected.
- RTS function is selected.
- CLK polarity select bit = "0".

fEXT: frequency of external clock

Meet the following conditions are met when the CLK input before data reception = "H"

- Transmit enable bit → "1"
- Receive enable bit → "1"
- Dummy data write to UARTi transmit buffer register

Figure 12-10. Typical transmit/receive timings in clock synchronous serial I/O mode

(a) Polarity select function

As shown in Figure 12-11, the CLK polarity select bit (bit 6 at addresses 03A4$_{16}$, 03AC$_{16}$, 01FC$_{16}$) allows selection of the polarity of the transfer clock.

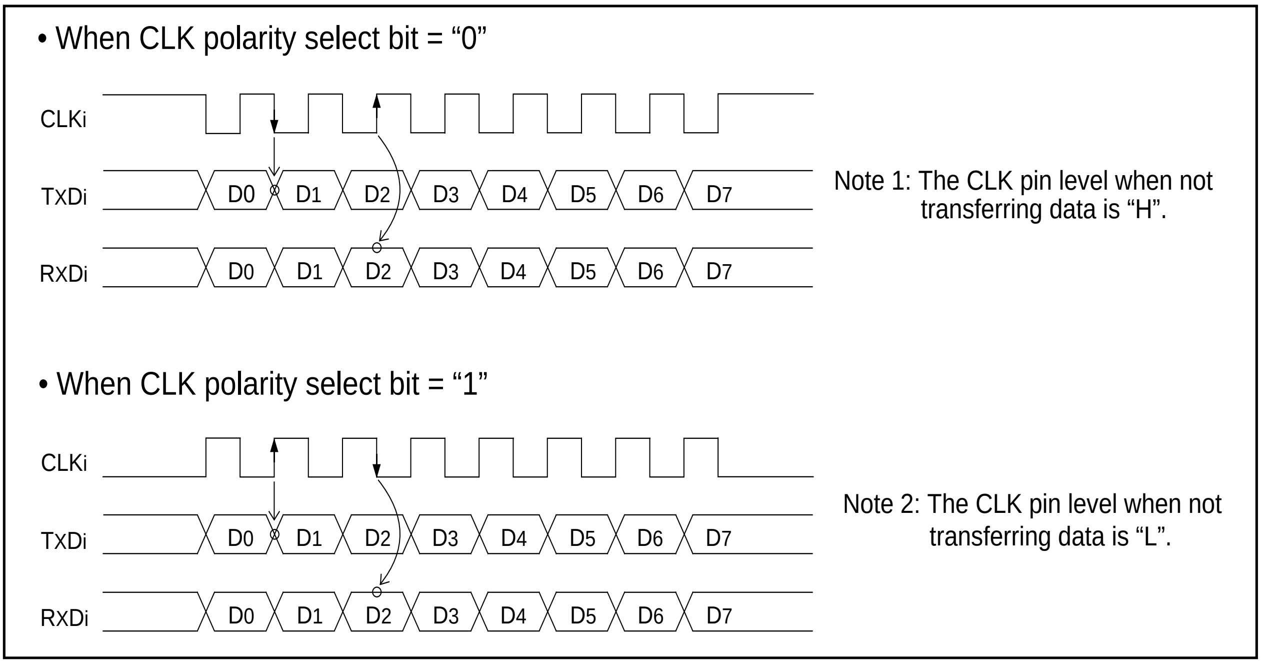

• When CLK polarity select bit = "0"
CLKi
TxDi
RxDi
D0
D1
D2
D3
D4
D5
D6
D7
Note 1: The CLK pin level when not transferring data is "H".
• When CLK polarity select bit = "1"
CLKi
TxDi
RxDi
Note 2: The CLK pin level when not transferring data is "L".

Figure 12-11. Polarity of transfer clock

(b) LSB first/MSB first select function

As shown in Figure 12-12, when the transfer format select bit (bit 7 at addresses 03A4$_{16}$, 03AC$_{16}$, 01FC$_{16}$) = "0", the transfer format is "LSB first"; when the bit = "1", the transfer format is "MSB first".

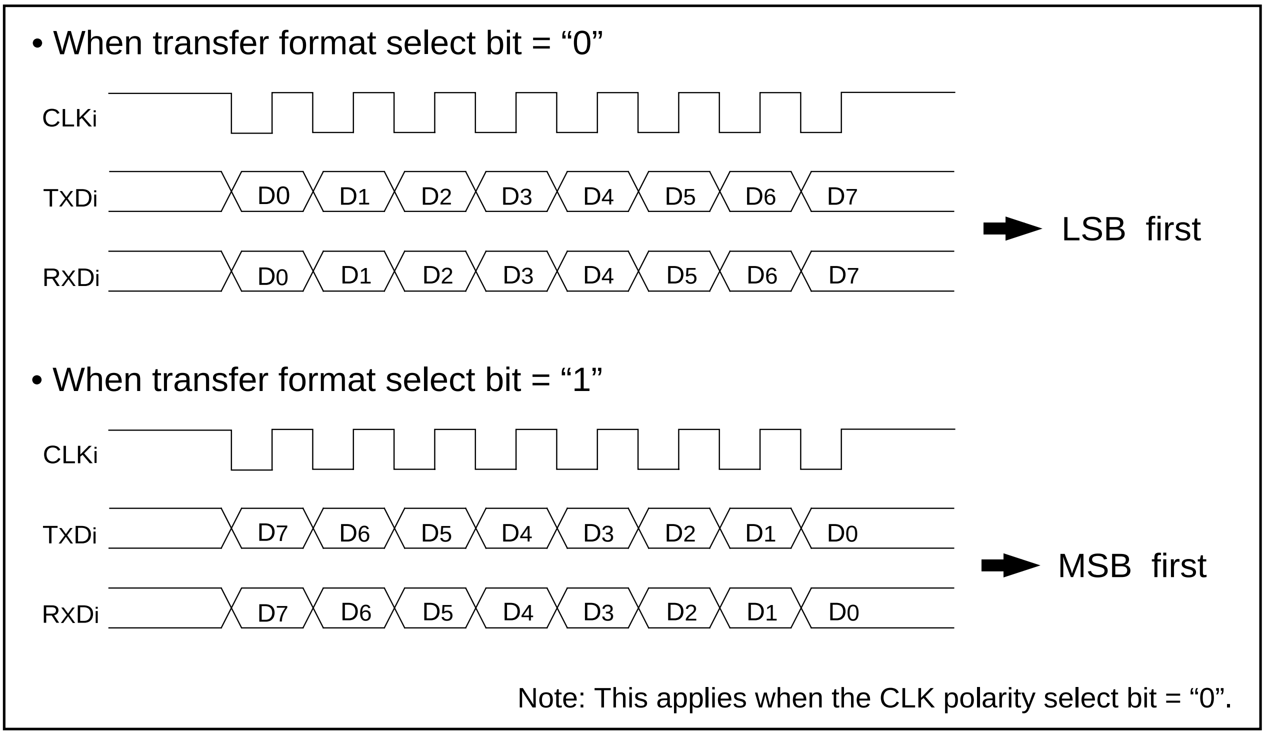
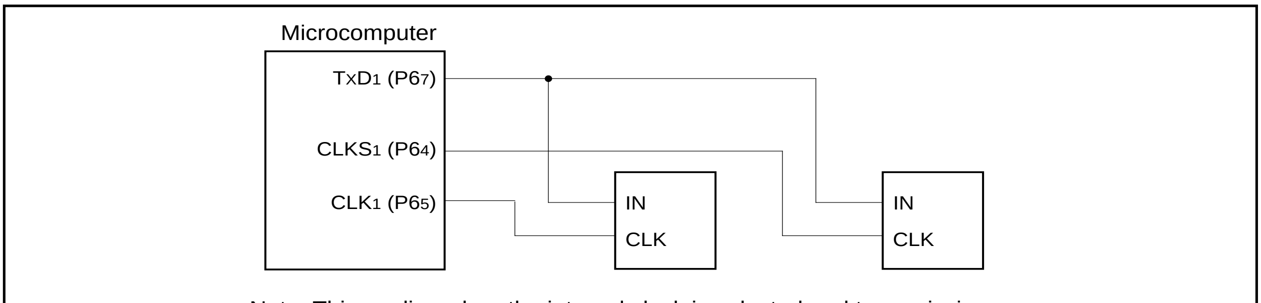

• When transfer format select bit = "0"
CLKi
TxDi
RxDi
D0
D1
D2
D3
D4
D5
D6
D7
LSB first
• When transfer format select bit = "1"
CLKi
TxDi
RxDi
D7
D6
D5
D4
D3
D2
D1
D0
MSB first
Note: This applies when the CLK polarity select bit = "0".

Figure 12-12. Transfer format

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

This function allows the setting two transfer clock output pins and choosing one of the two to output a clock by using the CLK and CLKS select bit (bits 4 and 5 at address $03B0_{16}$). (See Figure 12-13.) The multiple pins function is valid only when the internal clock is selected for UART1. Note that when this function is selected, UART1 $\overline{\text{CTS}}/\overline{\text{RTS}}$ function cannot be used.



Note: This applies when the internal clock is selected and transmission is performed only in clock synchronous serial I/O mode.

Figure 12-13. The transfer clock output from the multiple pins function usage

(d) Continuous receive mode

If the continuous receive mode enable bit (bits 2 and 3 at address $03B0_{16}$, bit 5 at address $01FD_{16}$) is set to “1”, the unit is placed in continuous receive mode. In this mode, when the receive buffer register is read out, the unit simultaneously goes to a receive enable state without having to set dummy data to the transmit buffer register back again.

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

This function works the same way as in the clock asynchronous serial I/O (UART) mode. The method of setting and the input/output pin functions are both the same, so refer to select function in the next section, “(2) Clock asynchronous serial I/O (UART) mode.” Note that this function is invalid if the transfer clock output from the multiple pins function is selected.

(f) Serial data logic switch function (UART2)

When the data logic select bit (bit6 at address $01FD_{16}$) = “1”, and writing to transmit buffer register or reading from receive buffer register, data is reversed. Figure 12-14 shows the example of serial data logic switch timing.

•When LSB first

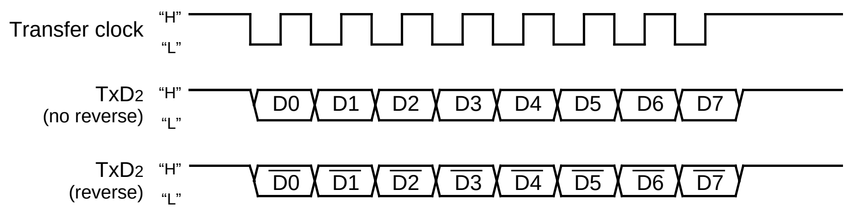


Figure 12-14. Serial data logic switch timing

(2) Clock asynchronous serial I/O (UART) mode

The UART mode allows transmitting and receiving data after setting the desired transfer rate and transfer data format. Tables 13-1 and 13-2 list the specifications of the UART mode. Figure 13-1 shows the UARTi transmit/receive mode register.

Table 13-1. Specifications of UART Mode (1)

Item	Specification
Transfer data format	• Character bit (transfer data): 7 bits, 8 bits, or 9 bits as selected • Start bit: 1 bit • Parity bit: Odd, even, or nothing as selected • Stop bit: 1 bit or 2 bits as selected
Transfer clock	• When internal clock is selected (bit 3 at addresses 03A0$_{16}$, 03A8$_{16}$, 01F8$_{16}$="0") : fi/16(n+1) (Note 1) fi = f$_{2SIO}$, f$_{8SIO}$, f$_{32SIO}$ • When external clock is selected (bit 3 at addresses 03A0$_{16}$, 03A8$_{16}$, 01F8$_{16}$ ="1") : f$_{EXT}$/16(n+1)(Note 1) (Note 2)
Transmission/reception control	• $\overline{\text{CTS}}$ function/$\overline{\text{RTS}}$ function/$\overline{\text{CTS}}$, $\overline{\text{RTS}}$ function chosen to be invalid
Transmission start condition	• To start transmission, the following requirements must be met: - Transmit enable bit (bit 0 at addresses 03A5$_{16}$, 03AD$_{16}$, 01FD$_{16}$) = "1" - Transmit buffer empty flag (bit 1 at addresses 03A5$_{16}$, 03AD$_{16}$, 01FD$_{16}$) = "0" - When $\overline{\text{CTS}}$ function selected, $\overline{\text{CTS}}$ input level = "L"
Reception start condition	• To start reception, the following requirements must be met: - Receive enable bit (bit 2 at addresses 03A5$_{16}$, 03AD$_{16}$, 01FD$_{16}$) = "1" - Start bit detection
Interrupt request generation timing	• When transmitting - Transmit interrupt cause select bits (bits 0,1 at address 03B0$_{16}$, bit4 at address 01FD$_{16}$) = "0": Interrupts requested when data transfer from UARTi transfer buffer register to UARTi transmit register is completed - Transmit interrupt cause select bits (bits 0, 1 at address 03B0$_{16}$, bit4 at address 01FD$_{16}$) = "1": Interrupts requested when data transmission from UARTi transfer register is completed • When receiving - Interrupts requested when data transfer from UARTi receive register to UARTi receive buffer register is completed
Error detection	• Overrun error (Note 3) This error occurs when the next data is ready before contents of UARTi receive buffer register are read out • 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

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

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

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

Table 13-2. Specifications of UART Mode (2)

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

UARTi transmit / receive mode registers

b7 b6 b5 b4 b3 b2 b1 b0

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

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

UART2 transmit / receive mode register

b7 b6 b5 b4 b3 b2 b1 b0

Symbol	Address	When reset
U2MR	01F8$_{16}$	00$_{16}$

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

Note: Usually set to “0”.

Figure 13-1. UARTi transmit/receive mode register in UART mode

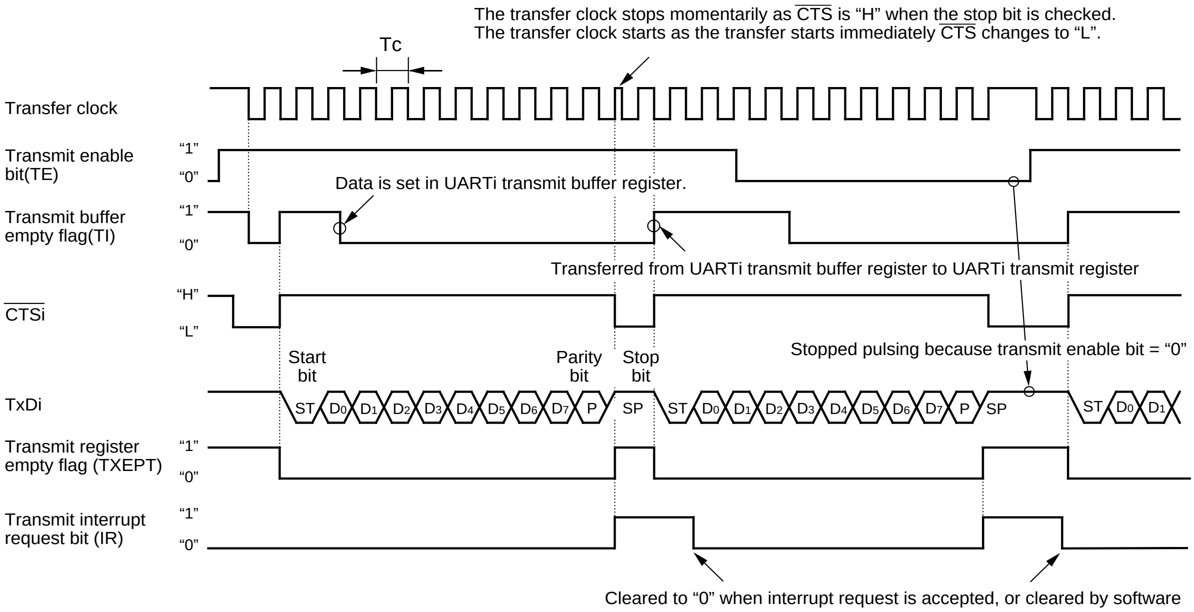
Table 13-3 lists the functions of the input/output pins during UART mode. This table shows the pin functions when the separate $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ pins function is not selected. Note that for a period from when the UARTi operation mode is selected to when transfer starts, the TxDi pin outputs a "H". (If the N-channel open-drain is selected, this pin is in floating state.)

Table 13-3. Input/output pin functions in UART mode

Pin name	Function	Method of selection
TxDi ($P6_3$, $P6_7$, $P7_0$)	Serial data output	
RxDi ($P6_2$, $P6_6$, $P7_1$)	Serial data input	Port $P6_2$, $P6_6$ and $P7_1$ direction register (bits 2 and 6 at address 03EE$_{16}$, bit 1 at address 03EF$_{16}$)= "0" (Can be used as an input port when performing transmission only)
CLKi ($P6_1$, $P6_5$, $P7_2$)	Programmable I/O port	Internal/external clock select bit (bit 3 at address 03A0$_{16}$, 03A8$_{16}$, 01F8$_{16}$) = "0"
	Transfer clock input	Internal/external clock select bit (bit 3 at address 03A0$_{16}$, 03A8$_{16}$, 01F8$_{16}$) = "1" Port $P6_1$, $P6_5$ and $P7_2$ direction register (bits 1 and 5 at address 03EE$_{16}$, bit 2 at address 03EF$_{16}$) = "0"
$\overline{\text{CTSi}}$/$\overline{\text{RTSi}}$ ($P6_0$, $P6_4$, $P7_3$)	$\overline{\text{CTS}}$ input	$\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit (bit 4 at address 03A4$_{16}$, 03AC$_{16}$, 01FC$_{16}$) ="0" $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ function select bit (bit 2 at address 03A4$_{16}$, 03AC$_{16}$, 01FC$_{16}$) = "0" Port $P6_0$, $P6_4$ and $P7_3$ direction register (bits 0 and 4 at address 03EE$_{16}$, bit 3 at address 03EF$_{16}$) = "0"
	$\overline{\text{RTS}}$ output	$\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit (bit 4 at address 03A4$_{16}$, 03AC$_{16}$, 01FC$_{16}$) = "0" $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ function select bit (bit 2 at address 03A4$_{16}$, 03AC$_{16}$, 01FC$_{16}$) = "1"
	Programmable I/O port	$\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit (bit 4 at address 03A4$_{16}$, 03AC$_{16}$, 01FC$_{16}$) = "1"

(when separate CTS/RTS pins function is not selected)

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


The transfer clock stops momentarily as $\overline{CTS}$ is “H” when the stop bit is checked.
The transfer clock starts as the transfer starts immediately $\overline{CTS}$ changes to “L”.

Data is set in UARTi transmit buffer register.

Transferred from UARTi transmit buffer register to UARTi transmit register

Stopped pulsing because transmit enable bit = “0”

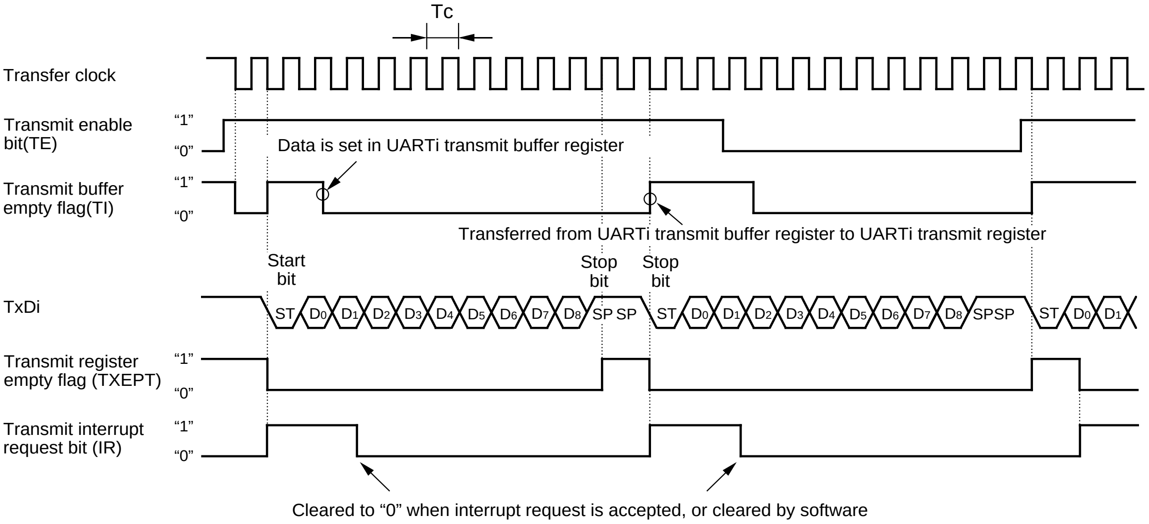
Cleared to “0” when interrupt request is accepted, or cleared by software

Shown in ( ) are bit symbols.

The above timing applies to the following settings :
- Parity is enabled.
- One stop bit.
- $\overline{CTS}$ function is selected.
- Transmit interrupt cause select bit = “1”.

Tc = 16 (n + 1) / fi or 16 (n + 1) / fEXT
fi : frequency of BRGi count source (f2SIO, f8SIO, f32SIO)
fEXT : frequency of BRGi count source (external clock)
n : value set to BRGi

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


Data is set in UARTi transmit buffer register

Transferred from UARTi transmit buffer register to UARTi transmit register

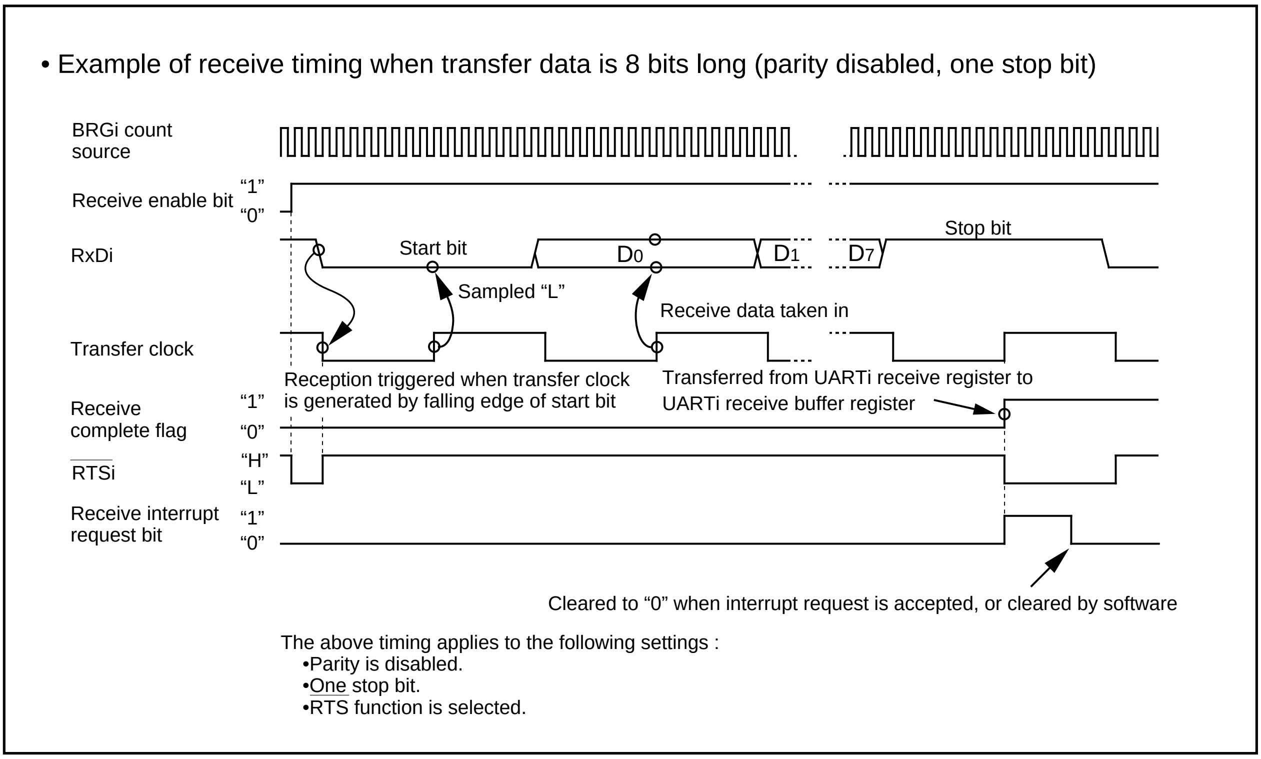
Cleared to “0” when interrupt request is accepted, or cleared by software

Shown in ( ) are bit symbols.

The above timing applies to the following settings :
- Parity is disabled.
- Two stop bits.
- $\overline{CTS}$ function is disabled.
- Transmit interrupt cause select bit = “0”.

Tc = 16 (n + 1) / fi or 16 (n + 1) / fEXT
fi : frequency of BRGi count source (f2SIO, f8SIO, f32SIO)
fEXT : frequency of BRGi count source (external clock)
n : value set to BRGi

Figure 13-2. Typical transmit timings in UART mode


• Example of receive timing when transfer data is 8 bits long (parity disabled, one stop bit)
BRGi count source
Receive enable bit
"1"
"0"
RxDi
Start bit
D0
D1
D7
Stop bit
Sampled "L"
Receive data taken in
Transfer clock
Reception triggered when transfer clock is generated by falling edge of start bit
Transferred from UARTi receive register to UARTi receive buffer register
Receive complete flag
"1"
"0"
RTSi
"H"
"L"
Receive interrupt request bit
"1"
"0"
Cleared to "0" when interrupt request is accepted, or cleared by software

The above timing applies to the following settings :
- Parity is disabled.
- One stop bit.
- RTS function is selected.

Figure 13-3. Typical receive timing in UART mode

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

With the separate $\overline{\text{CTS}}/\overline{\text{RTS}}$ bit (bit 6 at address $03B0_{16}$) is set to "1", the unit outputs/inputs the $\overline{\text{CTS}}$ and $\overline{\text{RTS}}$ signals on different pins. (See Figure 13-4.) This function is valid only for UART0. Note that if this function is selected, the $\overline{\text{CTS}}/\overline{\text{RTS}}$ function for UART1 cannot be used.

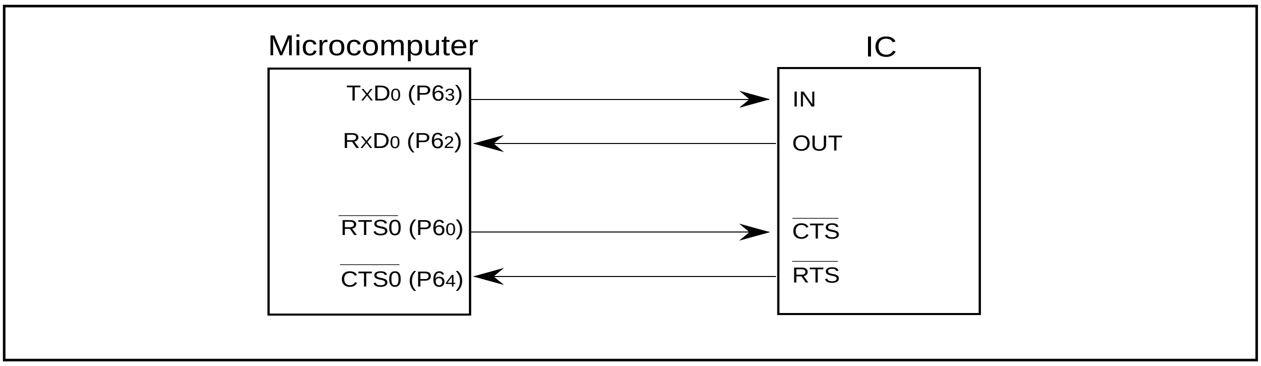


Figure 13-4. The separate CTS/RTS pins function usage

(b) Sleep mode (UART0, UART1)

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

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

When the data logic select bit (bit 6 of address $01FD_{16}$) is assigned 1, data is inverted in writing to the transmission buffer register or reading the reception buffer register. Figure 13-5 shows the example of timing for switching serial data logic.

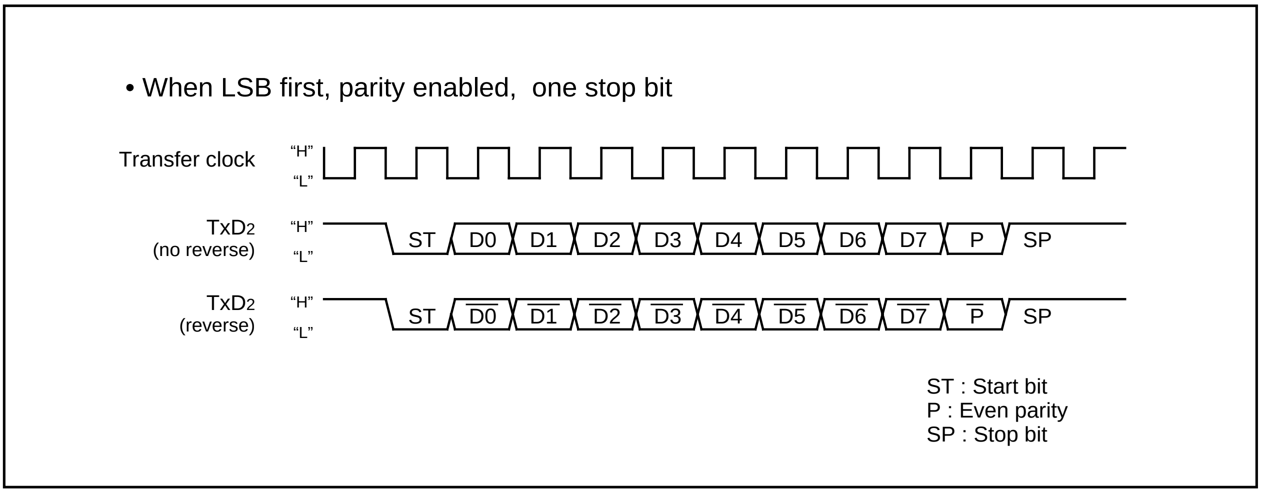


Figure 13-5. Timing for switching serial data logic

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

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

(e) Bus collision detection function (UART2)

This function is to sample the output level of the TxD pin and the input level of the RxD pin at the rising edge of the transfer clock; if their values are different, then an interrupt request occurs. Figure 13-6 shows the example of detection timing of a buss collision (in UART mode).

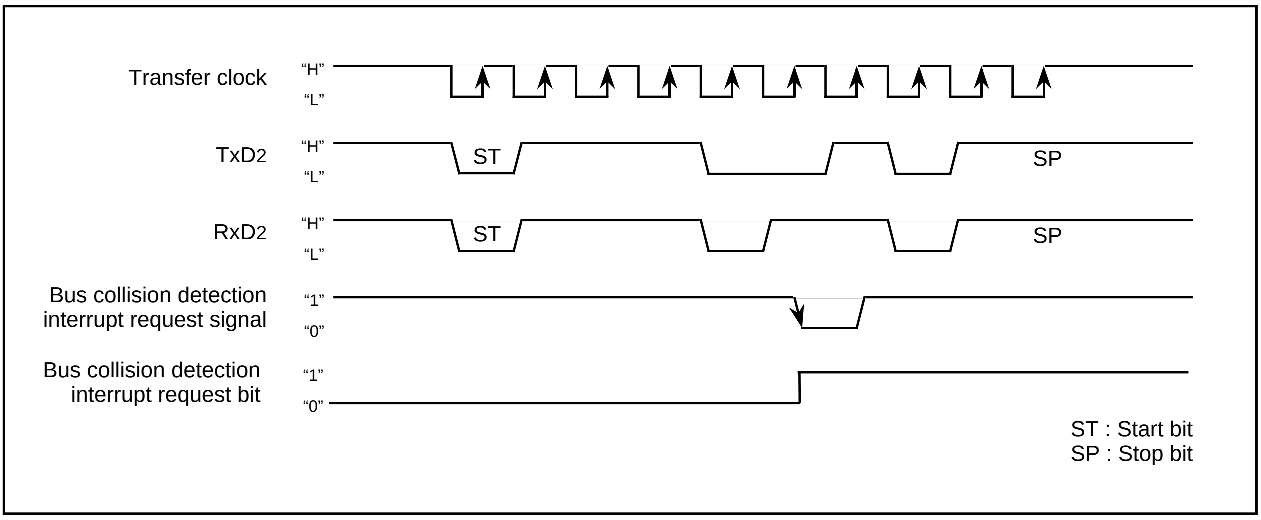

Transfer clock
"H"
"L"
TxD2
RxD2
ST
SP
Bus collision detection interrupt request signal
Bus collision detection interrupt request bit
"1"
"0"
ST : Start bit
SP : Stop bit

Figure 13-6. Detection timing of a bus collision (in UART mode)

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

The SIM interface is used for connecting the microcomputer with a memory card I/C or the like; adding some extra settings in UART2 clock-asynchronous serial I/O mode allows the user to effect this function. Table 13-4 shows the specifications of clock-asynchronous serial I/O mode (compliant with the SIM interface).

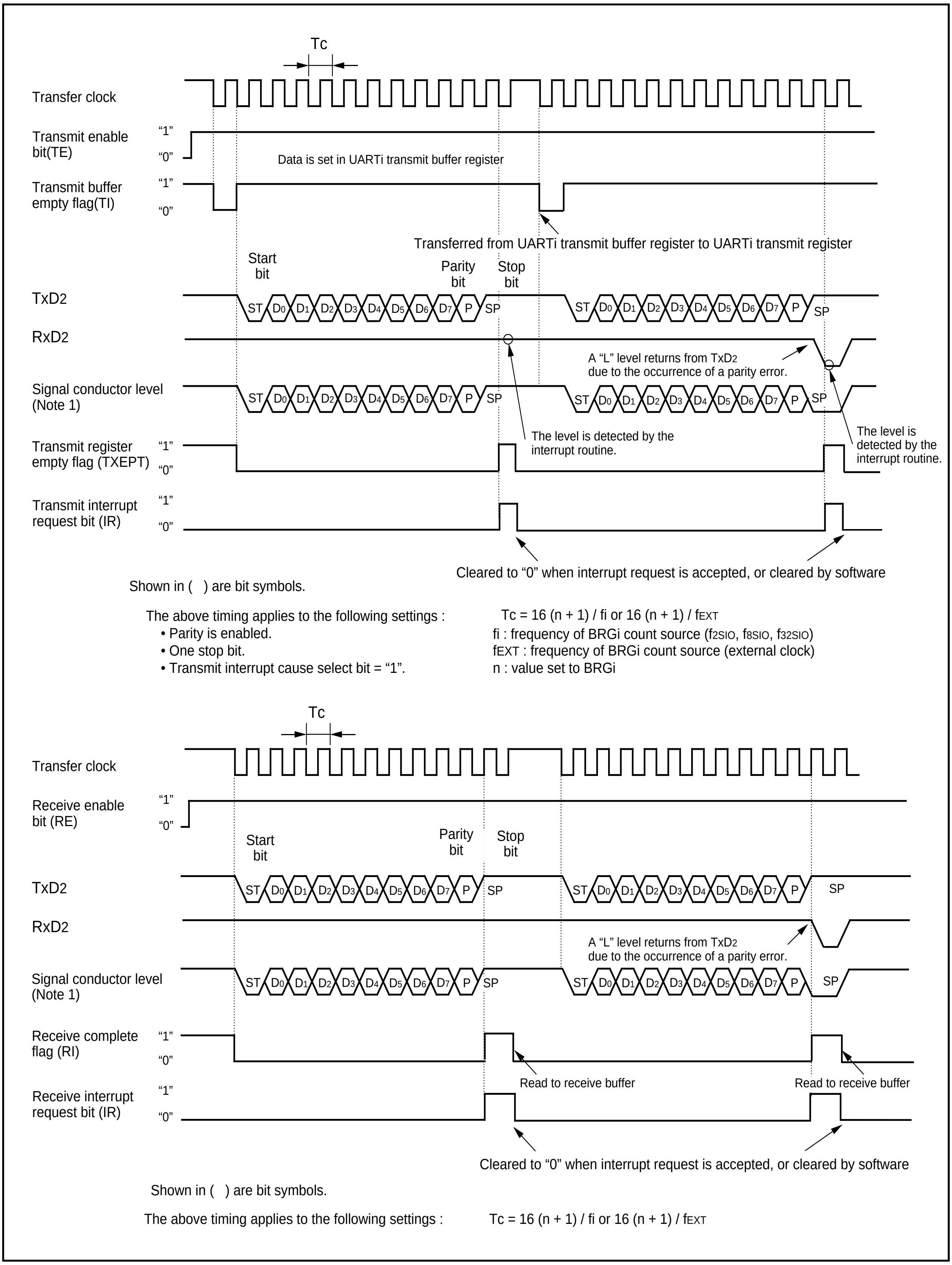
Table 13-4. Specifications of clock-asynchronous serial I/O mode (compliant with the SIM interface)

Item	Specification
Transfer data format	• Transfer data 8-bit UART mode (bit 2 through bit 0 of address $01F8_{16}$ = “101_2”) • One stop bit (bit 4 of address $01F8_{16}$ = “0”) • With the direct format chosen Set parity to “even” (bit 5 and bit 6 of address $01F8_{16}$ = “1” and “1” respectively) Set data logic to “direct” (bit 6 of address $01FD_{16}$ = “0”). Set transfer format to LSB (bit 7 of address $01FC_{16}$ = “0”). • With the inverse format chosen Set parity to “odd” (bit 5 and bit 6 of address $01F8_{16}$ = “0” and “1” respectively) Set data logic to “inverse” (bit 6 of address $01FD_{16}$ = “1”) Set transfer format to MSB (bit 7 of address $01FC_{16}$ = “1”)
Transfer clock	• With the internal clock chosen (bit 3 of address $01F8_{16}$ = “0”) : f_i / 16 (n + 1) (Note 1) : f_i=f_2, f_8, f_{32} • With an external clock chosen (bit 3 of address $01F8_{16}$ = “1”) : f_{EXT} / 16 (n+1) (Note 1) (Note 2)
Transmission / reception control	• Disable the CTS and RTS function (bit 4 of address $01FC_{16}$ = “1”)
Other settings	• The sleep mode select function is not available for UART2 • Set transmission interrupt factor to “transmission completed” (bit 4 of address $01FD_{16}$ = “1”)
Transmission start condition	• To start transmission, the following requirements must be met: - Transmit enable bit (bit 0 of address $01FD_{16}$) = “1” - Transmit buffer empty flag (bit 1 of address $01FD_{16}$) = “0”
Reception start condition	• To start reception, the following requirements must be met: - Reception enable bit (bit 2 of address $01FD_{16}$) = “1” - Detection of a start bit
Interrupt request generation timing	• When transmitting When data transmission from the UART2 transfer register is completed (bit 4 of address $01FD_{16}$ = “1”) • When receiving When data transfer from the UART2 receive register to the UART2 receive buffer register is completed
Error detection	• Overrun error (see the specifications of clock-asynchronous serial I/O) (Note 3) • Framing error (see the specifications of clock-asynchronous serial I/O) • Parity error (see the specifications of clock-asynchronous serial I/O) - On the reception side, an “L” level is output from the TxD_2 pin by use of the parity error signal output function (bit 7 of address $01FD_{16}$ = “1”) when a parity error is detected - On the transmission side, a parity error is detected by the level of input to the RxD_2 pin when a transmission interrupt occurs • The error sum flag (see the specifications of clock-asynchronous serial I/O)

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

Note 2: f_{EXT} is input from the CLK_2 pin.

Note 3: If an overrun error occurs, the UART2 receive buffer will have the next data written in. Note also that the UARTi receive interrupt request bit is not set to “1”.


Tc
Transfer clock
Transmit enable bit(TE)
"1"
"0"
Data is set in UARTi transmit buffer register
Transmit buffer empty flag(TI)
Transferred from UARTi transmit buffer register to UARTi transmit register
Start bit
Parity bit
Stop bit
TxD2
ST D0 D1 D2 D3 D4 D5 D6 D7 P SP
RxD2
A "L" level returns from TxD2 due to the occurrence of a parity error.
Signal conductor level (Note 1)
The level is detected by the interrupt routine.
Transmit register empty flag (TXEPT)
Transmit interrupt request bit (IR)
Cleared to "0" when interrupt request is accepted, or cleared by software
Receive enable bit (RE)
Receive complete flag (RI)
Read to receive buffer
Receive interrupt request bit (IR)

Shown in ( ) are bit symbols.

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

Tc = 16 (n + 1) / fi or 16 (n + 1) / fEXT
fi : frequency of BRGi count source (f2SIO, f8SIO, f32SIO)
fEXT : frequency of BRGi count source (external clock)
n : value set to BRGi

Shown in ( ) are bit symbols.

The above timing applies to the following settings : Tc = 16 (n + 1) / fi or 16 (n + 1) / fEXT

Figure 13-7. Typical transmit/receive timing in UART mode (compliant with the SIM interface)

(a) Function for outputting a parity error signal

With the error signal output enable bit (bit 7 of address $01FD_{16}$) assigned “1”, you can output an “L” level from the TxD_2 pin when a parity error is detected. In step with this function, the generation timing of a transmission completion interrupt changes to the detection timing of a parity error signal. Figure 13-8 shows the output timing of the parity error signal.

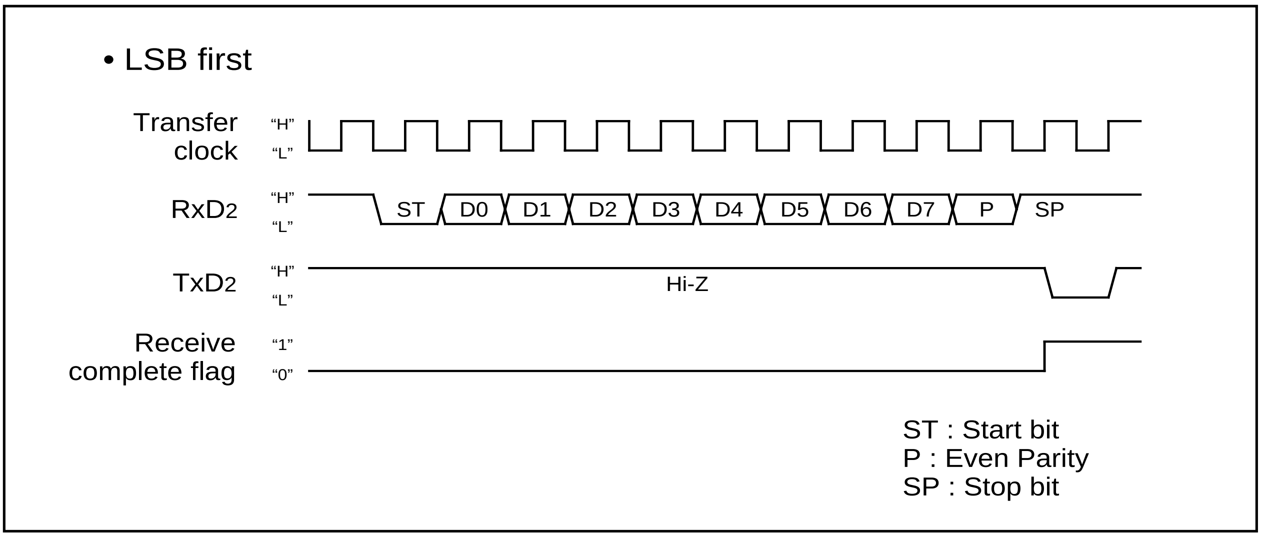


Figure 13-8. Output timing of the parity error signal

(b) Direct format/inverse format

Connecting the SIM card allows you to switch between direct format and inverse format. If you choose the direct format, D_0 data is output from TxD_2. If you choose the inverse format, D_7 data is inverted and output from TxD_2.

Figure 13-9 shows the SIM interface format.

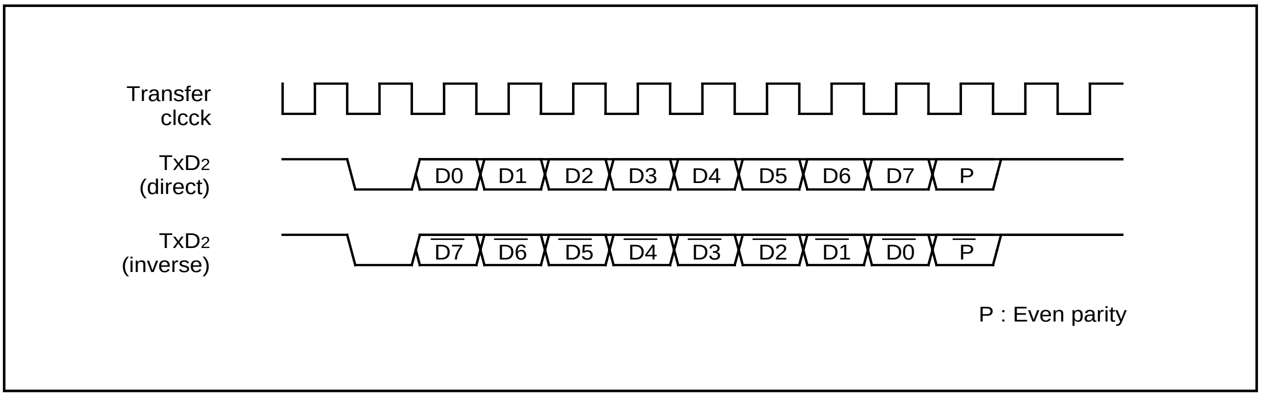


Figure 13-9. SIM interface format

Figure 13-10 shows the example of connecting the SIM interface. Connect TxD2 and RxD2 and apply pull-up.

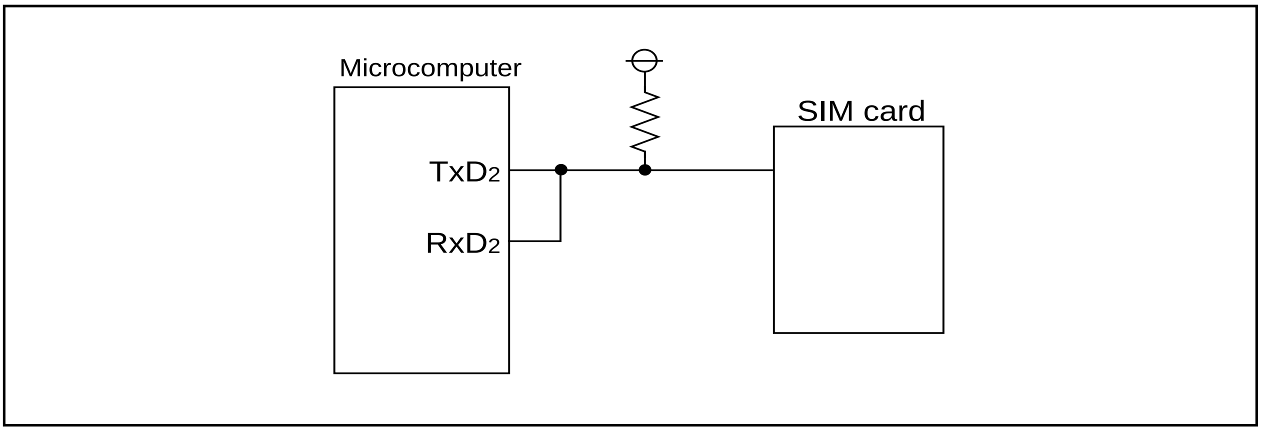
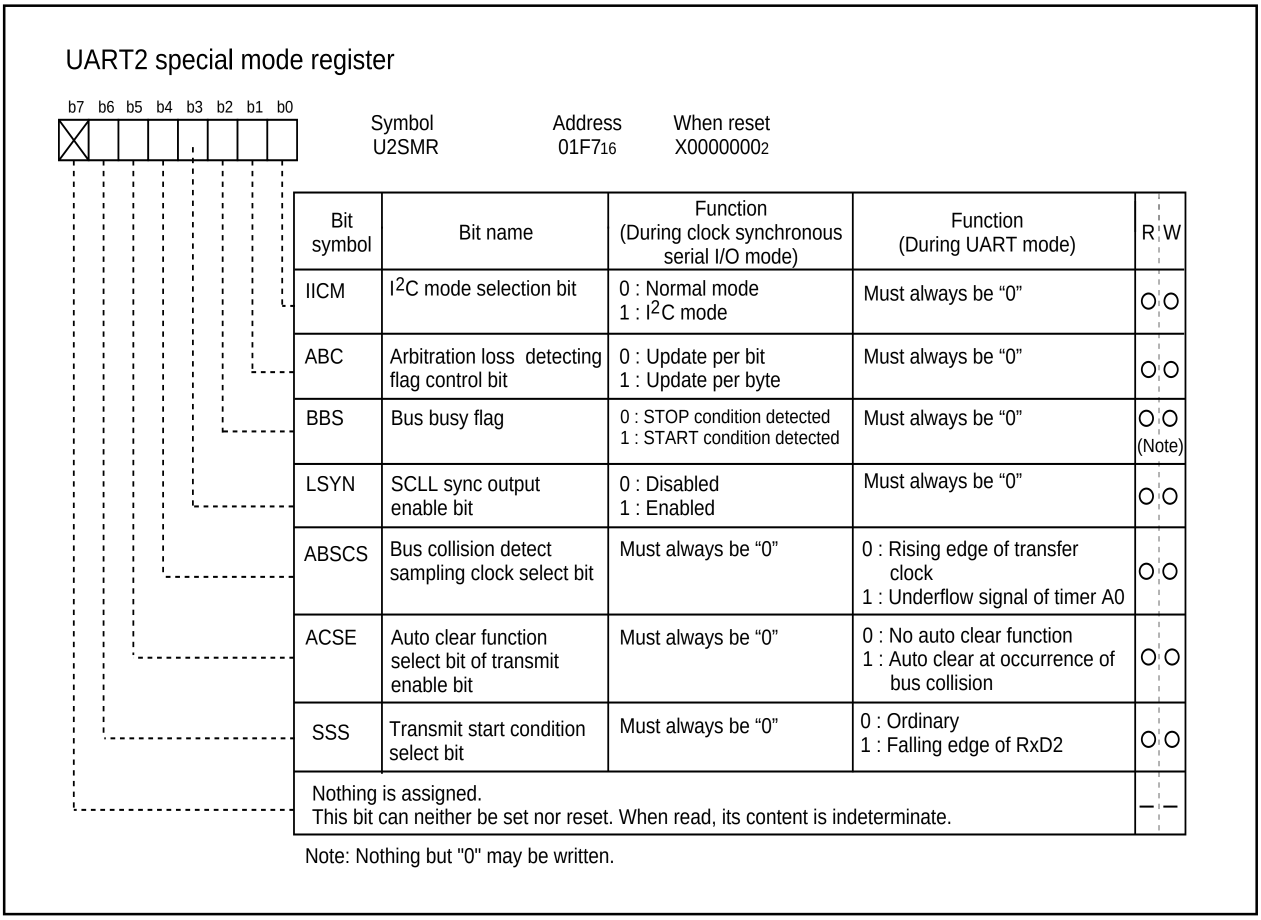


Figure 13-10. Connecting the SIM interface

UART2 Special Mode Register

The UART2 special mode register (address $01F7_{16}$) is used to control UART2 in various ways.

Figure 13-11 shows the special UART2 mode register.



UART2 special mode register

Symbol	Address	When reset
U2SMR	$01F7_{16}$	$X0000000_2$

Bit symbol	Bit name	Function (During clock synchronous serial I/O mode)	Function (During UART mode)	R	W
IICM	I^{2}C mode selection bit	0 : Normal mode 1 : I^{2}C mode	Must always be “0”	○	○
ABC	Arbitration loss detecting flag control bit	0 : Update per bit 1 : Update per byte	Must always be “0”	○	○
BBS	Bus busy flag	0 : STOP condition detected 1 : START condition detected	Must always be “0”	○	○ (Note)
LSYN	SCLL sync output enable bit	0 : Disabled 1 : Enabled	Must always be “0”	○	○
ABSCS	Bus collision detect sampling clock select bit	Must always be “0”	0 : Rising edge of transfer clock 1 : Underflow signal of timer A0	○	○
ACSE	Auto clear function select bit of transmit enable bit	Must always be “0”	0 : No auto clear function 1 : Auto clear at occurrence of bus collision	○	○
SSS	Transmit start condition select bit	Must always be “0”	0 : Ordinary 1 : Falling edge of RxD2	○	○
Nothing is assigned. This bit can neither be set nor reset. When read, its content is indeterminate.				–	–

Note: Nothing but "0" may be written.

Figure 13-11. UART2 special mode register

Table13-5. Features in I^{2}C mode

	Function	Normal mode	I^{2}C mode (Note 1)
1	Factor of interrupt number 10 (Note 2)	Bus collision detection	Start condition detection or stop condition detection
2	Factor of interrupt number 15 (Note 2)	UART2 transmission	No acknowledgment detection (NACK)
3	Factor of interrupt number 16 (Note 2)	UART2 reception	Acknowledgment detection (ACK)
4	UART2 transmission output delay	Not delayed	Delayed
5	P7$_0$ at the time when UART2 is in use	TxD$_2$ (output)	SDA (input/output) (Note 3)
6	P7$_1$ at the time when UART2 is in use	RxD$_2$ (input)	SCL (input/output)
7	P7$_2$ at the time when UART2 is in use	CLK2	P7$_2$
8	DMA1 factor at the time when 1 1 0 1 is assigned to the DMA request factor selection bits	UART2 reception	Acknowledgment detection (ACK)
9	Noise filter width	15ns	50ns
10	Reading P7$_1$	Reading the terminal when 0 is assigned to the direction register	Reading the terminal regardless of the value of the direction register
11	Initial value of UART2 output	H level (when 0 is assigned to the CLK polarity select bit)	The value set in latch P7$_0$ when the port is selected

Note 1: Make the settings given below when I^{2}C mode is in use.
Set 0 1 0 in bits 2, 1, 0 of the UART2 transmission/reception mode register.
Disable the RTS/CTS function. Select TxD$_2$ as Nch. Choose the LSB First function.

Note 2: Follow the steps given below to switch from a factor to another.
1. Disable the interrupt of the corresponding number.
2. Switch from a factor to another.
3. Reset the interrupt request flag of the corresponding number.
4. Set an interrupt level of the corresponding number.

Note 3: Set an initial value of SDA transmission output when serial I/O is invalid.

In the first place, the control bits related to the I^{2}C bus(simplified I^{2}C bus) interface are explained.
Bit 0 of the UART2 special mode register ($01F7_{16}$) is used as the I^{2}C mode selection bit.
Setting “1” in the I^{2}C mode selection bit (bit 0) goes the circuit to achieve the I^{2}C bus interface effective.
Table 13-5 shows the relation between the I^{2}C mode selection bit and respective control workings.
In order to configure P7$_0$ as Nch open drain output, set bit 5 (Nch) in the UART transmit/receive control register 0 (U2C0).
Since this function uses clock-synchronous serial I/O mode, be sure to set this bit to “0” in UART mode.

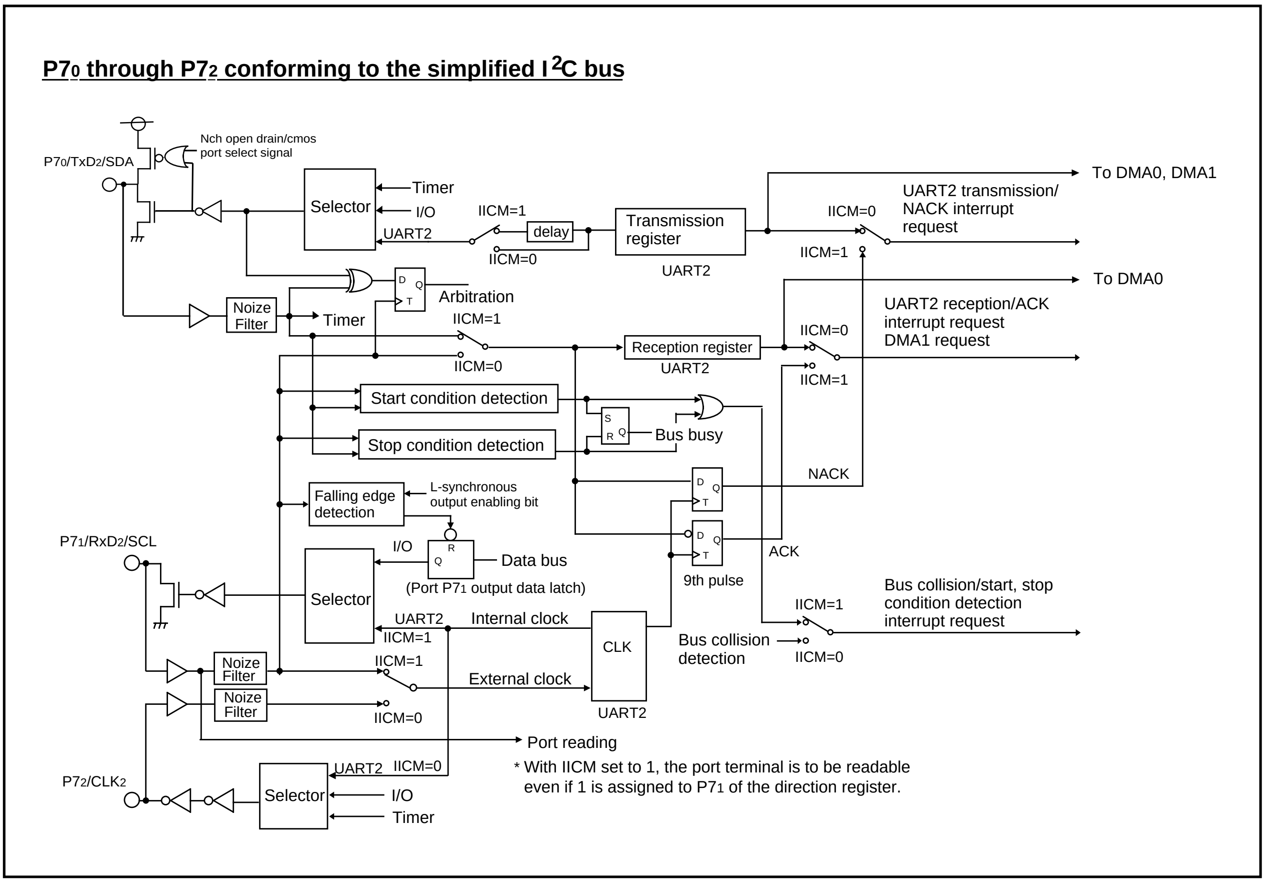


Figure 13-12. Functional block diagram for I^{2}C mode

Figure 13-12 shows the functional block diagram for I^{2}C mode. Setting “1” in the I^{2}C mode selection bit (IICM) causes ports P70, P71, and P72 to work as data transmission-reception terminal SDA, clock input-output terminal SCL, and port P72 respectively. A delay circuit is added to the SDA transmission output, so the SDA output changes after SCL fully goes to L. An attempt to read Port P71 (SCL) results in getting the terminal’s level regardless of the content of the port direction register. The initial value of SDA transmission output in this mode goes to the value set in port P70. The interrupt factors of the bus collision detection interrupt, UART2 transmission interrupt, and of UART2 reception interrupt turn to the start/stop condition detection interrupt, acknowledgment non-detection interrupt, and acknowledgment detection interrupt respectively.
The start condition detection interrupt refers to the interrupt that occurs when the falling edge of the SDA terminal (P70) is detected with the SCL terminal (P71) staying “H”. The stop condition detection interrupt refers to the interrupt that occurs when the rising edge of the SDA terminal (P70) is detected with the SCL terminal (P71) staying “H”. The bus busy flag (bit 2 of the special UART2 mode register) is set to “1” by the

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

Bit 1 of the special UART2 mode register (01F716) is used as the arbitration loss detection flag control bit. Arbitration means the act of detecting the nonconformity between transmission data and SDA terminal data at the timing of the SCL rising edge. This detecting flag is located at bit 3 of the UART2 reception buffer register (01FF16), and “1” is set in this flag when nonconformity is detected. Use the arbitration loss detecting flag control bit to choose which way to use to update the flag, bit by bit or byte by byte. When setting this bit to “1” and updated the flag byte by byte if nonconformity is detected, the arbitration lost detecting flag is set to "1" at the falling edge of the 9th transmission clock.

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

Bit 3 of the special UART2 mode register is used as SCL- and L-synchronous output enabling bit. Setting this bit to “1” resets the P71 data register to “0” in synchronization with the SCL terminal level going to “L”.

Some other functions added are explained here. Figure 13-13 shows their workings.

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

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

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

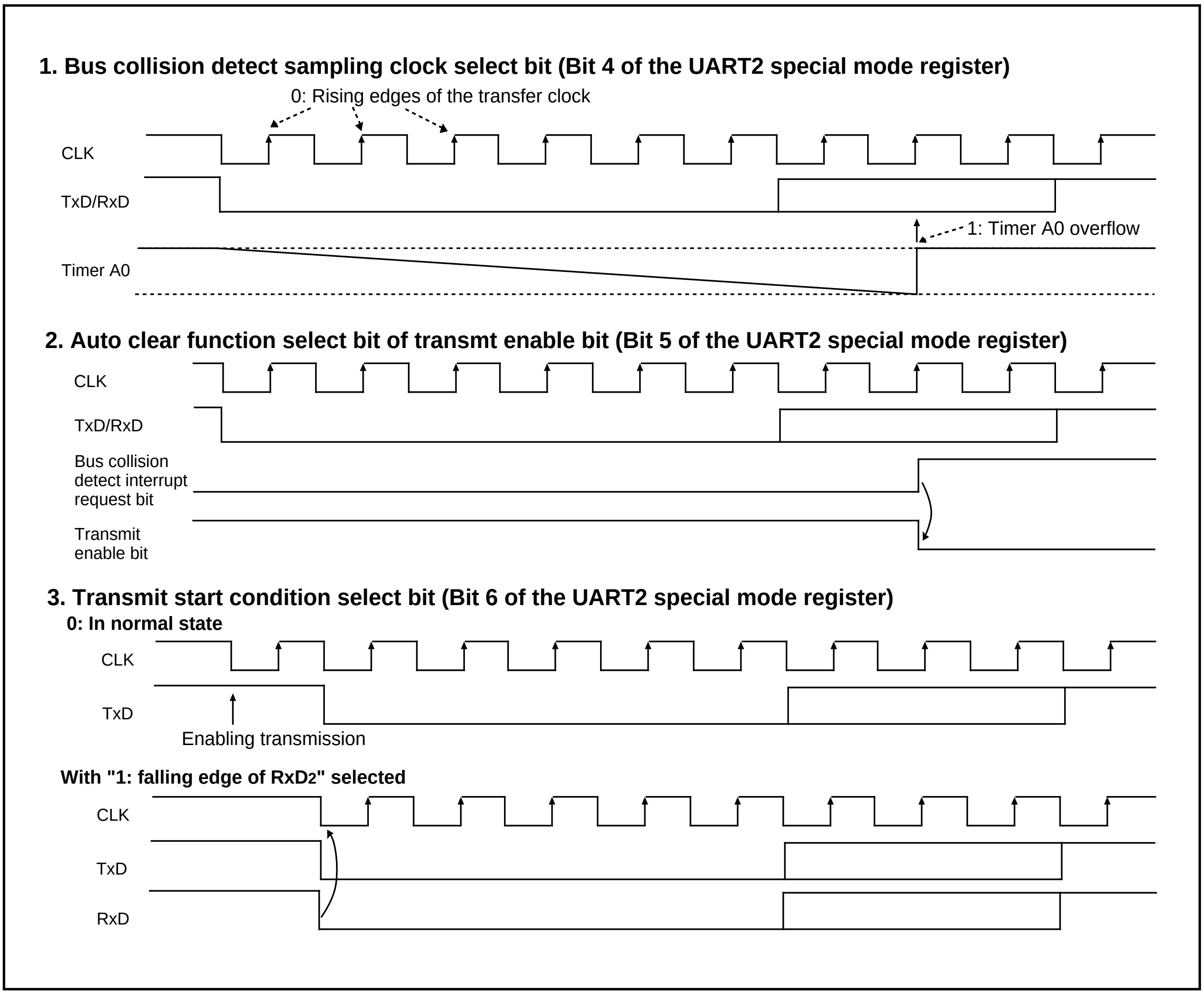
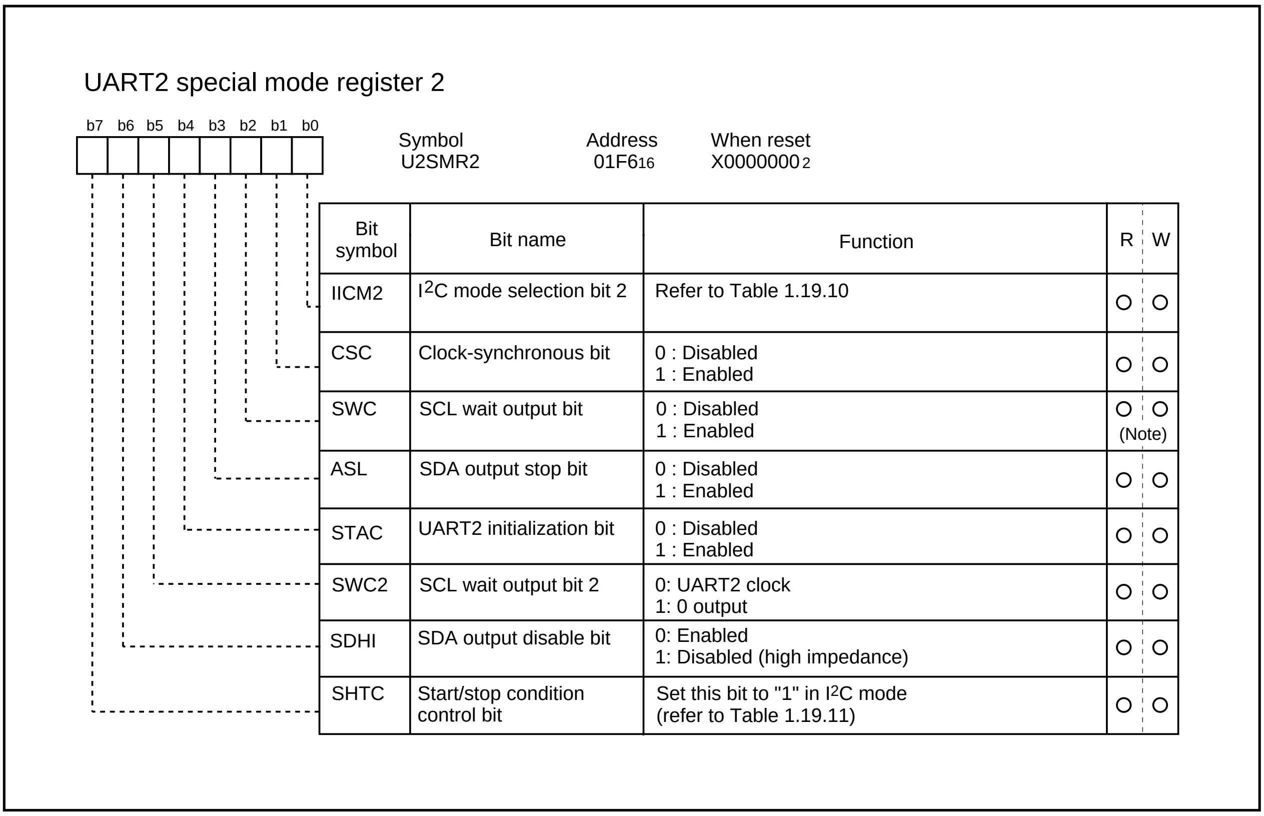

1. Bus collision detect sampling clock select bit (Bit 4 of the UART2 special mode register)
0: Rising edges of the transfer clock
CLK
TxD/RxD
1: Timer A0 overflow
Timer A0
2. Auto clear function select bit of transmt enable bit (Bit 5 of the UART2 special mode register)
CLK
TxD/RxD
Bus collision detect interrupt request bit
Transmit enable bit
3. Transmit start condition select bit (Bit 6 of the UART2 special mode register)
0: In normal state
CLK
TxD
Enabling transmission
With "1: falling edge of RxD2" selected
CLK
TxD
RxD

Figure 13-13. Other functions controlled by UART2 special mode register

UART2 Special Mode Register 2

The UART2 special mode register 2(address $01F6_{16}$) is used to further control UART2 in I^{2}C mode. Figure 13-14 shows the special UART2 mode register.



UART2 special mode register 2

Symbol	Address	When reset
U2SMR2	$01F6_{16}$	$X0000000_2$

Bit symbol	Bit name	Function	R	W
IICM2	I^{2}C mode selection bit 2	Refer to Table 1.19.10	○	○
CSC	Clock-synchronous bit	0 : Disabled 1 : Enabled	○	○
SWC	SCL wait output bit	0 : Disabled 1 : Enabled	○	○ (Note)
ASL	SDA output stop bit	0 : Disabled 1 : Enabled	○	○
STAC	UART2 initialization bit	0 : Disabled 1 : Enabled	○	○
SWC2	SCL wait output bit 2	0: UART2 clock 1: 0 output	○	○
SDHI	SDA output disable bit	0: Enabled 1: Disabled (high impedance)	○	○
SHTC	Start/stop condition control bit	Set this bit to "1" in I^{2}C mode (refer to Table 1.19.11)	○	○

Figure 13-14. UART2 special mode register 2

Bit 0 of the UART2 special mode register 2(address 01F6$_{16}$) is used as the I^{2}C mode selection bit 2. Table 13-6 shows the types of control to be changed by I^{2}C mode selection bit 2 when the I^{2}C mode selection bit is set to "1". Table 13-7 shows the timing characteristics of detecting the start condition and the stop condition. Set the start/stop condition control bit (bit 7 of UART2 special mode register 2) to "1" in I^{2}C mode.

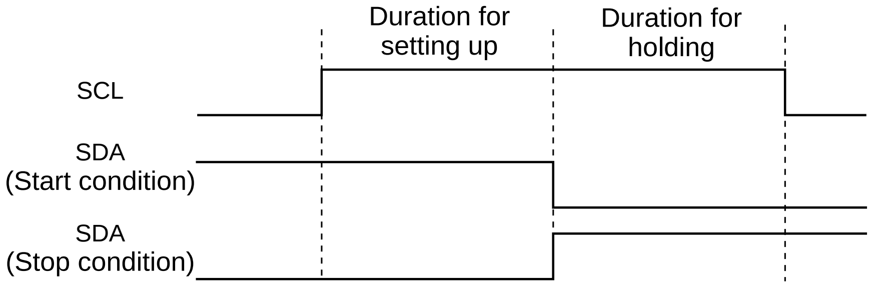
Table 13-6. Functions changed by I^{2}C mode selection bit 2

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

Table 13-7. Timing characteristics of detecting the start condition and the stop condition

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

Note 1 : When the start/stop condition count bit is "1" .
Note 2 : "cycles" is in terms of the input oscillation frequency f(X$_{IN}$) of the main clock.



UART2 Special Mode Register 2

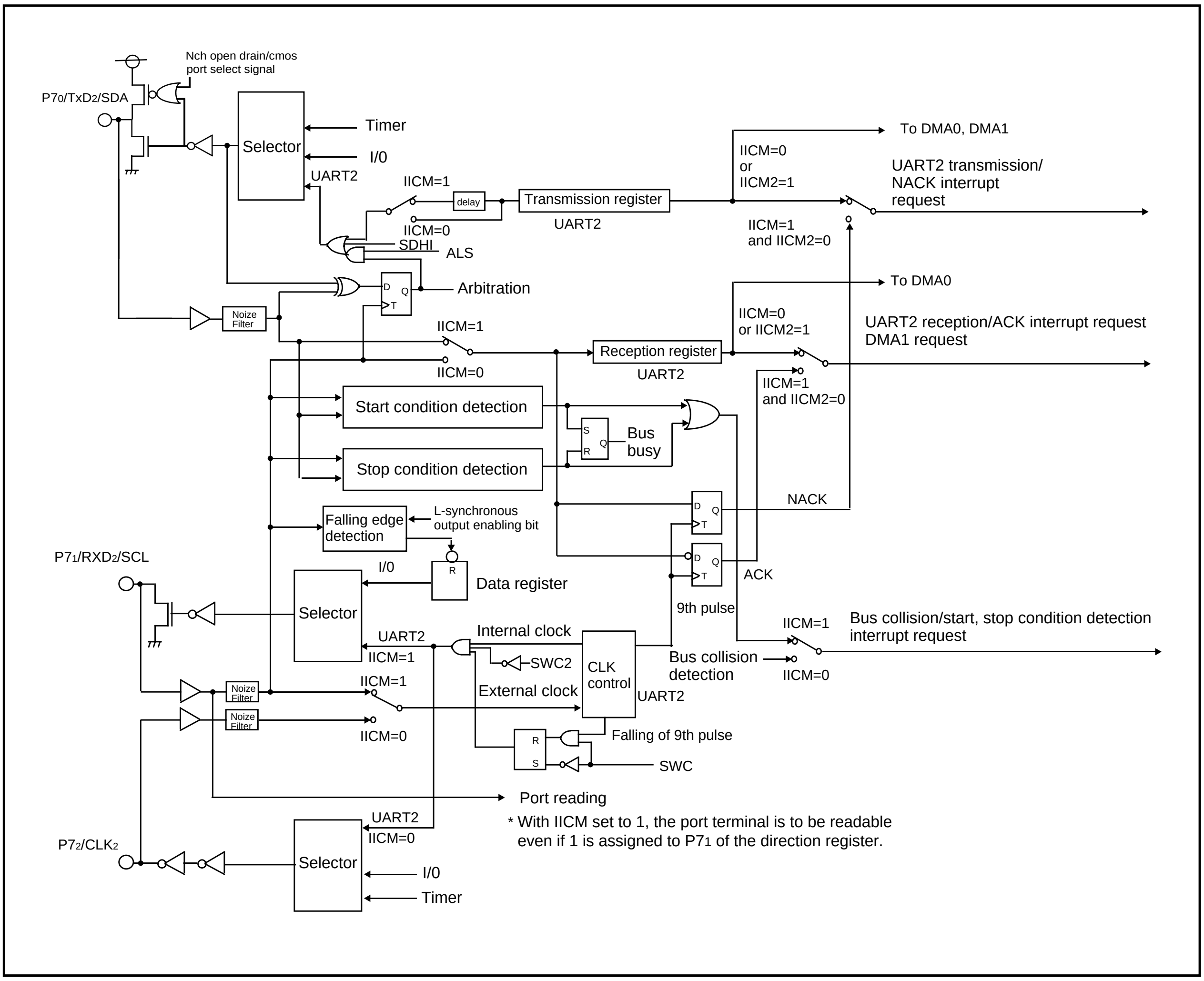


Figure 13-15. Functional block diagram for I²C mode

Functions available in I2C mode are shown in Figure 13-15 — a functional block diagram.

Bit 3 of the UART2 special mode register 2 (address 01F6$_{16}$)is used as the SDA output stop bit. Setting this bit to "1" causes an arbitration loss to occur, and the SDA pin turns to high-impedance state the instant when the arbitration loss detecting flag is set to "1".

Bit 1 of the UART2 special mode register 2 (address 01F6$_{16}$) is used as the clock synchronization bit. With this bit set to "1" at the time when the internal SCL is set to "H", the internal SCL turns to "L" if the falling edge is found in the SCL pin; and the baus rate generator reloads the set value, and start counting within the "L" interval. When the internal SCL changes from "L" to "H" with the SCL pin set to "L", stops counting the baud rate generator, and starts counting it again when the SCL pin turns to "H". Due to this function, the UART2 transmission-reception clock becomes the logical product of the signal flowing through the internal SCL and that flowing through the SCL pin. This function operates over the period from the moment earlier by a half cycle than falling edge of the UART2 first clock to the rising edge of the ninth bit. To use this function, choose the internal clock for the transfer clock.

Bit 2 of the UART2 special mode register 2 (01F6$_{16}$) is used as the SCL wait output bit. Setting this bit to "1" causes the SCL pin to be fixed to "L" at the falling edge of the ninth bit of the clock. Setting this bit to "0" frees the output fixed to "L".

UART2 Special Mode Register 2

Bit 4 of the UART2 special mode register 2(address 01F6$_{16}$) is used as the UART2 initialization bit. Setting this bit to "1", and when the start condition is detected, the microcomputer operates as follows:

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

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

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

Starting to transmit/receive signals to/from UART2 using this function does not change the value of the transmission buffer empty flag. To use this function, choose the external clock for the tansfer clocl.

Bit 5 of the UART2 special mode register 2 (01F6$_{16}$) is used as the SCL pin wait output bit 2. Setting this bit to "1" with the serial I/O specified allows the user to forcibly output an "L" from the SCL pin even if UART2 is in operation. Setting this bit to "0" frees the "L" output from the SCL pin, and the UART2 clock is input/output.

Bit 6 of the UART special mode register 2 (01F6$_{16}$) is used as the SDA output enable bit. Setting this bit to "1" forces the SDA pin to turn to the high-impedance state. Refrain from changing the value of this bit at the rising edge of the UART2 transfer clock. There can be instances in which arbitration loss detecting flag is turned on.

S I/O3

S I/O3 is exclusive clock-synchronous serial I/O.

Figure 14-1 shows the S I/O3 block diagram, and Figure 14-2 shows the S I/O3 control register.

Table 14-1 shows the specifications of S I/O3.

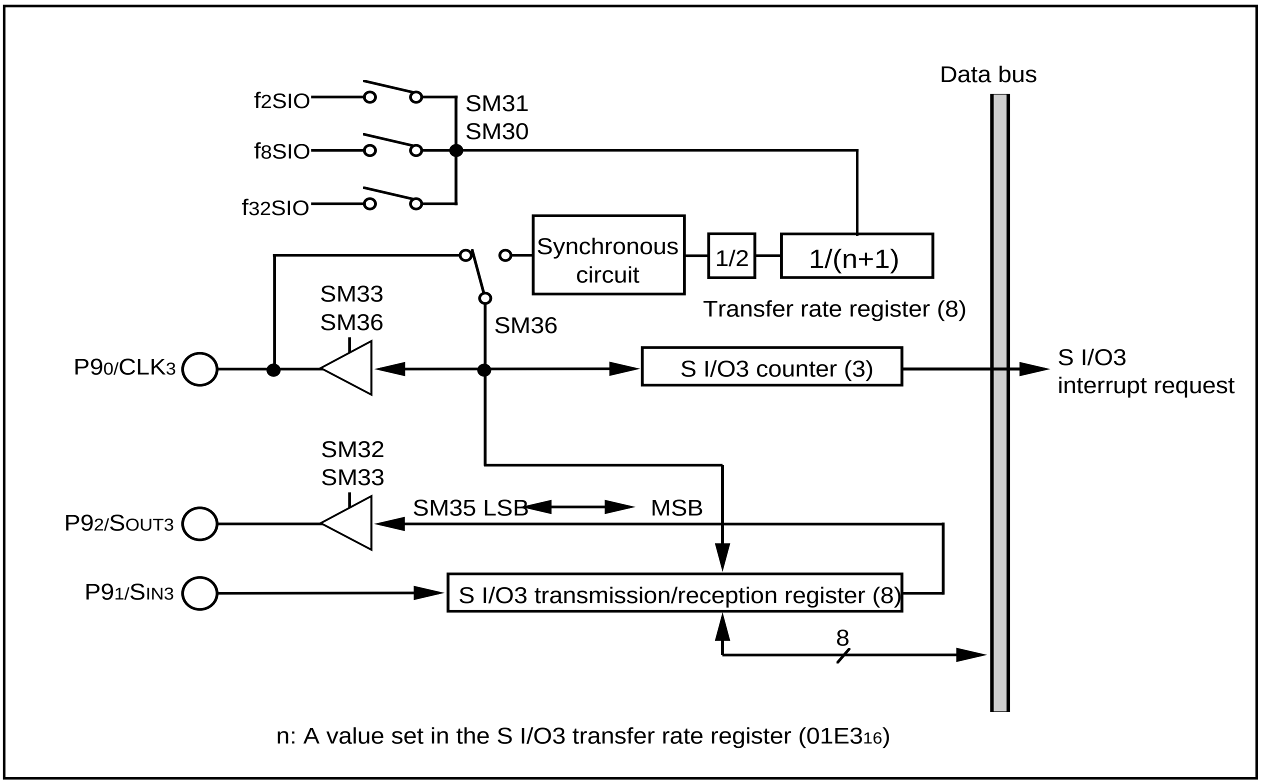


Figure 14-1. S I/O3 block diagram

S I/O3 control register (Note 1)

b7 b6 b5 b4 b3 b2 b1 b0

Symbol S3C | Address $01E2_{16}$ | When reset 40_{16}

output,

Bit symbol	Bit name	Description	R	W
SM30	Internal synchronous clock select bit	b1 b0 0 0 : Selecting f_{2SIO} 0 1 : Selecting f_{8SIO} 1 0 : Selecting f_{32SIO} 1 1 : Not to be used	○	○
SM31			○	○
SM32	SOUT3 high impedance control bit	0 : SOUT3 output 1 : SOUT3 high impedance	○	○
SM33	S I/O3 port select bit (Note 2)	0 : Input-output port 1 : SOUT3 output, CLK function	○	○
Nothing is assigned. This bit can neither be set nor read. When read, the value of this bit is “0”.			—	—
SM35	Transfer direction select bit	0 : LSB first 1 : MSB first	○	○
SM36	Synchronous clock select bit	0 : External clock 1 : Internal clock	○	○
SM37	SOUT3 initial value set bit	Effective when SM33 = 0 0 : L output 1 : H output	○	○

Note 1: Set "1" in bit 2 of the protection register ($000A_{16}$) in advance to write to the S I/O3 control register.

Note 2: When set "0" to SM33 and select input - output port, set "1" to SM36 and select internal clock, or input "H" to $P9_0$ and $P9_5$.

Figure 14-2. S I/O3 control register

Table 14-1. Specifications of S I/O3

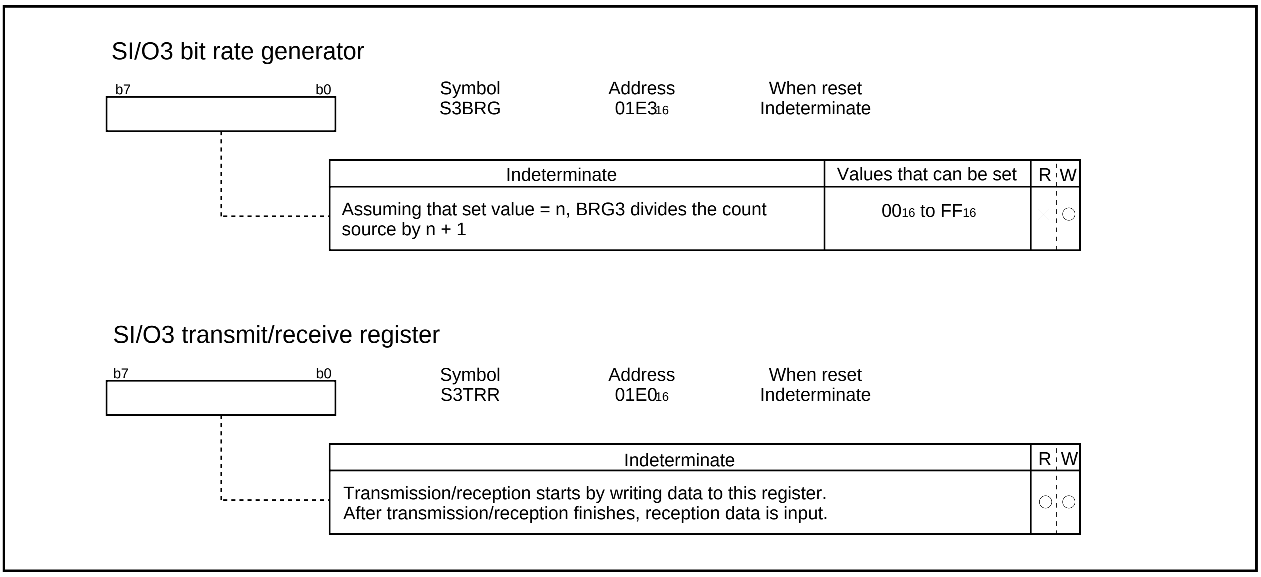
Item	Specifications
Transfer data format	• Transfer data length: 8 bits
Transfer clock	• With the internal clock selected (bit 6 of $01E2_{16}$ = “1”): f2SIO/2(n+1), f8SIO/2(n+1), f32SIO/2(n+1) (Note 1) • With the external clock selected (bit 6 of $01E2_{16}$): Input from the CLK3 terminal (Note 2)
Conditions for transmission/ reception start	• To start transmit/reception, the following requirements must be met: - Select the synchronous clock (use bit 6 of $01E2_{16}$). Select a frequency dividing ratio if the internal clock has been selected (use bits 0 and 1 of $01E2_{16}$). - SOUT3 initial value set bit (use bit 7 of $01E2_{16}$)= 1. - S I/O3 port select bit (bit 3 of $01E2_{16}$) = 1. - Select the transfer direction (use bit 5 of $01E2_{16}$) • To use S I/O3 interrupts, the following requirements must be met: - S I/O3 interrupt request bit (bit 3 of 0049_{16}) = 0.
Interrupt request generation timing	• An interrupt occurs after counting eight transfer clock either in transmitting or receiving data. (Note 3) - In transmitting: At the time data transfer from the S I/O3 transmission/reception register finishes. - In receiving: At the time data reception to the S I/O3 transmission/reception register finishes.
Select function	• LSB first or MSB first selection Whether transmission/reception begins with bit 0 or bit 7 can be selected.

Note 1: n is a value from 00_{16} through FF_{16} set in the S I/O3 transfer rate register.

Note 2: With the external clock selected:

- To write to the S I/O3 transmission/reception register ($01E2_{16}$), enter the “H” level to the CLK3 terminal. Also, to write to the bit 7 (SOUT3 initial value set bit) of SI/O3 control register ($01E2_{16}$), enter the “H” level to the CLK3 terminal.
- The S I/O3 circuit keeps on with the shift operation as long as the synchronous clock is entered in it, so stop the synchronous clock at the instant when it counts to eight. The internal clock, if selected, automatically stops.

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



SI/O3 bit rate generator

Symbol	Address	When reset
S3BRG	$01E3_{16}$	Indeterminate

Indeterminate	Values that can be set	R	W
Assuming that set value = n, BRG3 divides the count source by n + 1	00_{16} to FF_{16}		○

SI/O3 transmit/receive register

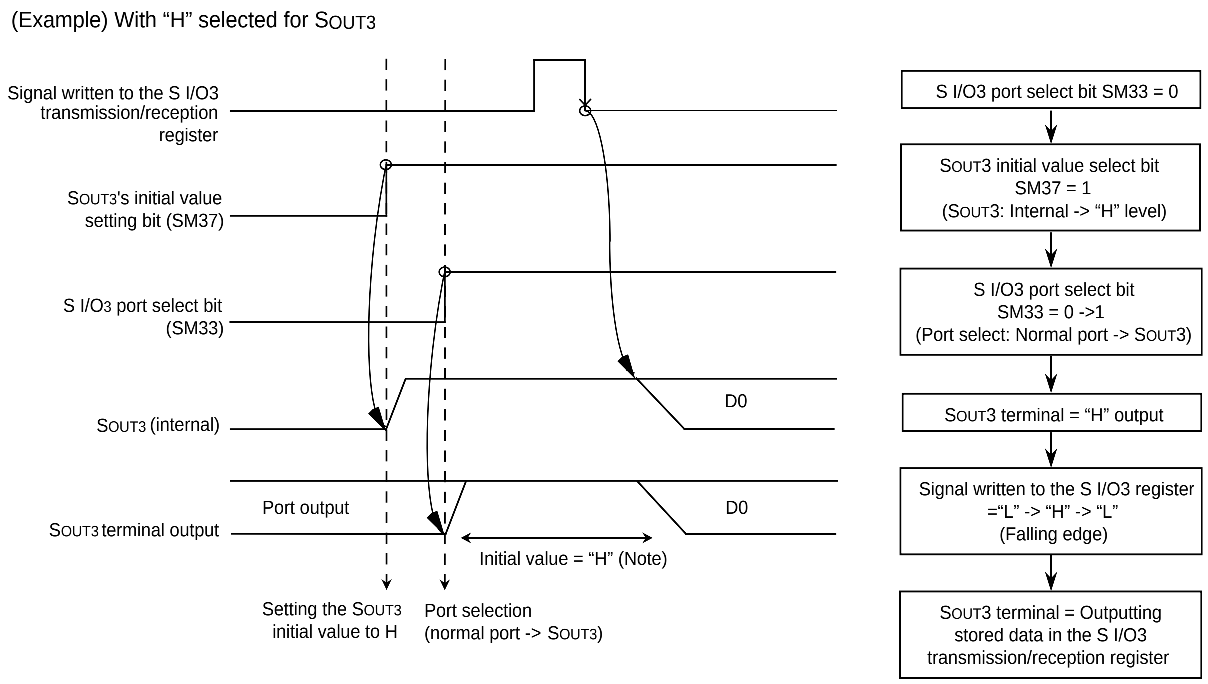
Symbol	Address	When reset
S3TRR	$01E0_{16}$	Indeterminate

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

Figure 14-3. SI/O3 related register

■ Functions for setting an SOUT3 initial value

In carrying out transmission, the output level of the SOUT3 terminal as it is before transmitting 1-bit data can be set either to “H” or to “L”. Figure 14-4 shows the timing chart for setting an SOUT3 initial value and how to set it.

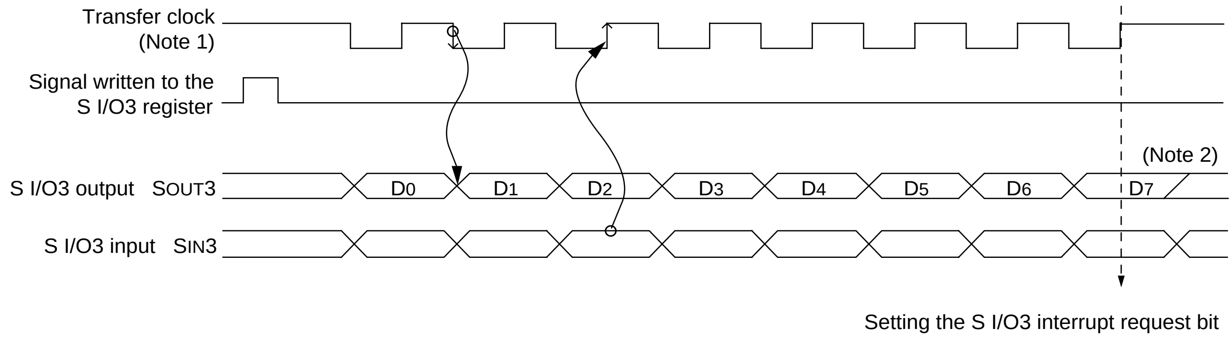

(Example) With “H” selected for SOUT3
Signal written to the S I/O3 transmission/reception register
SOUT3's initial value setting bit (SM37)
S I/O3 port select bit (SM33)
SOUT3 (internal)
SOUT3 terminal output
D0
Port output
D0
Initial value = “H” (Note)
Setting the SOUT3 initial value to H
Port selection (normal port -> SOUT3)
S I/O3 port select bit SM33 = 0
SOUT3 initial value select bit SM37 = 1 (SOUT3: Internal -> “H” level)
S I/O3 port select bit SM33 = 0 ->1 (Port select: Normal port -> SOUT3)
SOUT3 terminal = “H” output
Signal written to the S I/O3 register =“L” -> “H” -> “L” (Falling edge)
SOUT3 terminal = Outputting stored data in the S I/O3 transmission/reception register

Note: The set value is output only when the external clock has been selected.
If the internal clock has been selected or if SOUT high impedance has been set, this output goes to the high-impedance state.

Figure 14-4. Timing chart for setting SOUT3’s initial value and how to set it

■ S I/O3 operation timing

Figure 14-5 shows the S I/O3 operation timing



Note 1: With the internal clock selected for the transfer clock, the frequency dividing ratio can be selected using bits 0 and 1 of the S I/O3 control register. (2-division frequency, 8-division frequency, 32-division frequency)
Note 2: With the internal clock selected for the transfer clock, the SOUT3 terminal becomes to the high-impedance state after the transfer finishes.

Figure 14-5. S I/O3 operation timing chart

A-D Converter

The A-D converter consists of one 10-bit successive approximation A-D converter circuit with a capacitive coupling amplifier. Pins $P0_0$ to $P0_7$, $P2_0$ to $P2_7$, $P10_0$ to $P10_7$, $P9_5$, and $P9_6$ function as the analog signal input pins. The direction registers of these pins for A-D conversion must therefore be set to input. The Vref connect bit (bit 5 at address $03D7_{16}$) can be used to isolate the resistance ladder of the A-D converter from the reference voltage input pin (VREF) when the A-D converter is not used. Doing so stops any current flowing into the resistance ladder from VREF, reducing the power dissipation. When using the A-D converter, start A-D conversion only after setting bit 5 of $03D7_{16}$ to connect VREF.

The result of A-D conversion is stored in the A-D registers of the selected pins. When set to 10-bit precision, the low 8 bits are stored in the even addresses and the high 2 bits in the odd addresses. When set to 8-bit precision, the 8 bits are stored in the even addresses.

Table 15-1 shows the performance of the A-D converter. Figure 15-1 shows the block diagram of the A-D converter, and Figures 15-2 and 15-3 show the A-D converter-related registers.

Table 15-1. 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 condition (Note 2)	VCC = 5V, f_{AD2} divided by 1, 2, or 4, f_{AD2}=f(X_{IN}) divided by 1 or 2
Resolution	8-bit or 10-bit (selectable)
Absolute precision	VCC = 5V • Without sample and hold function ±3LSB • With sample and hold function (8-bit resolution) ±2LSB • With sample and hold function (10-bit resolution) AN_0 to AN_7 input : ±3LSB ANEX0 and ANEX1 input (including mode in which external operation amp is connected) : ±7LSB
Operating modes	One-shot mode, repeat mode, single sweep mode, repeat sweep mode 0, and repeat sweep mode 1
Analog input pins	24 pins (AN_0 to AN_7, $AN0_0$ to $AN0_7$ and $AN2_0$ to $AN2_7$) + 2 pins (ANEX0 and ANEX1)
A-D conversion start condition	• Software trigger A-D conversion starts when the A-D conversion start flag changes to “1” • External trigger (can be retriggered) A-D conversion starts when the A-D conversion start flag is “1” and the $\overline{AD_{TRG}}$/$P9_7$ input changes from “H” to “L”
Conversion speed per pin	• Without sample and hold function 8-bit resolution: 49 ϕ_{AD} cycles, 10-bit resolution: 59 ϕ_{AD} cycles • With sample and hold function 8-bit resolution: 28 ϕ_{AD} cycles, 10-bit resolution: 33 ϕ_{AD} cycles

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

Note 2: Without sample and hold function, set the ϕ_{AD} frequency to 250kHz min.
With the sample and hold function, set the ϕ_{AD} frequency to 1MHz min.
In either case, the ϕAD frequency may not exceed 10 MHz.

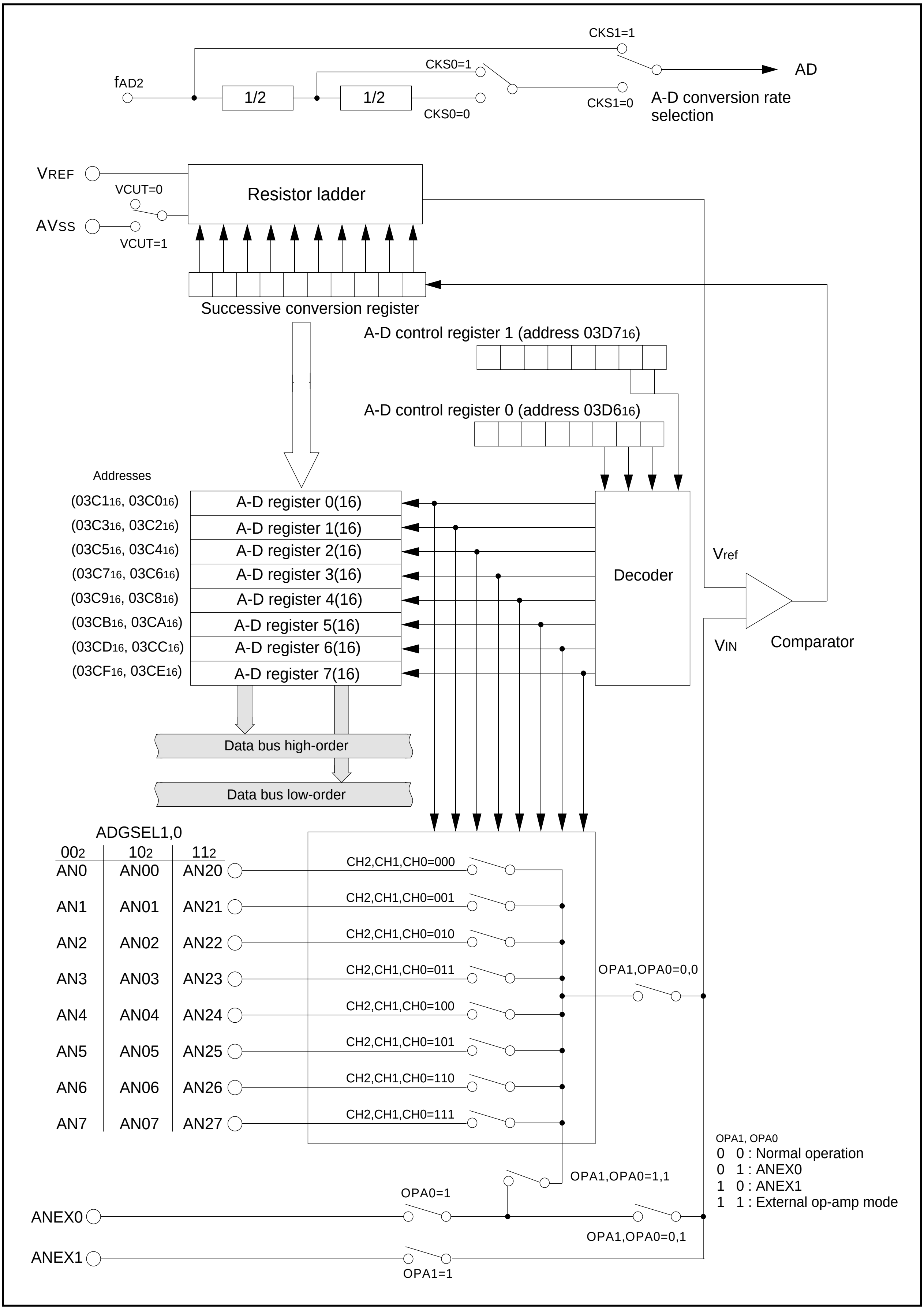


Figure 15-1. Block diagram of A-D converter

A-D control register 0 (Note)

b7 b6 b5 b4 b3 b2 b1 b0

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

Bit symbol	Bit name	Function	R	W
CH0	Analog input pin select bit	b2 b1 b0 0 0 0 : AN_0 is selected 0 0 1 : AN_1 is selected 0 1 0 : AN_2 is selected 0 1 1 : AN_3 is selected 1 0 0 : AN_4 is selected 1 0 1 : AN_5 is selected 1 1 0 : AN_6 is selected 1 1 1 : AN_7 is selected	○	○
CH1			○	○
CH2			○	○
MD0	A-D operation mode select bit 0	b4 b3 0 0 : One-shot mode 0 1 : Repeat mode 1 0 : Single sweep mode 1 1 : Repeat sweep mode 0 Repeat sweep mode 1	○	○
MD1			○	○
TRG	Trigger select bit	0 : Software trigger 1 : ADTRG trigger	○	○
ADST	A-D conversion start flag	0 : A-D conversion disabled 1 : A-D conversion started	○	○
CKS0	Frequency select bit 0	0 : f_{AD2}/4 is selected 1 : f_{AD2}/2 is selected	○	○

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

A-D control register 1 (Note)

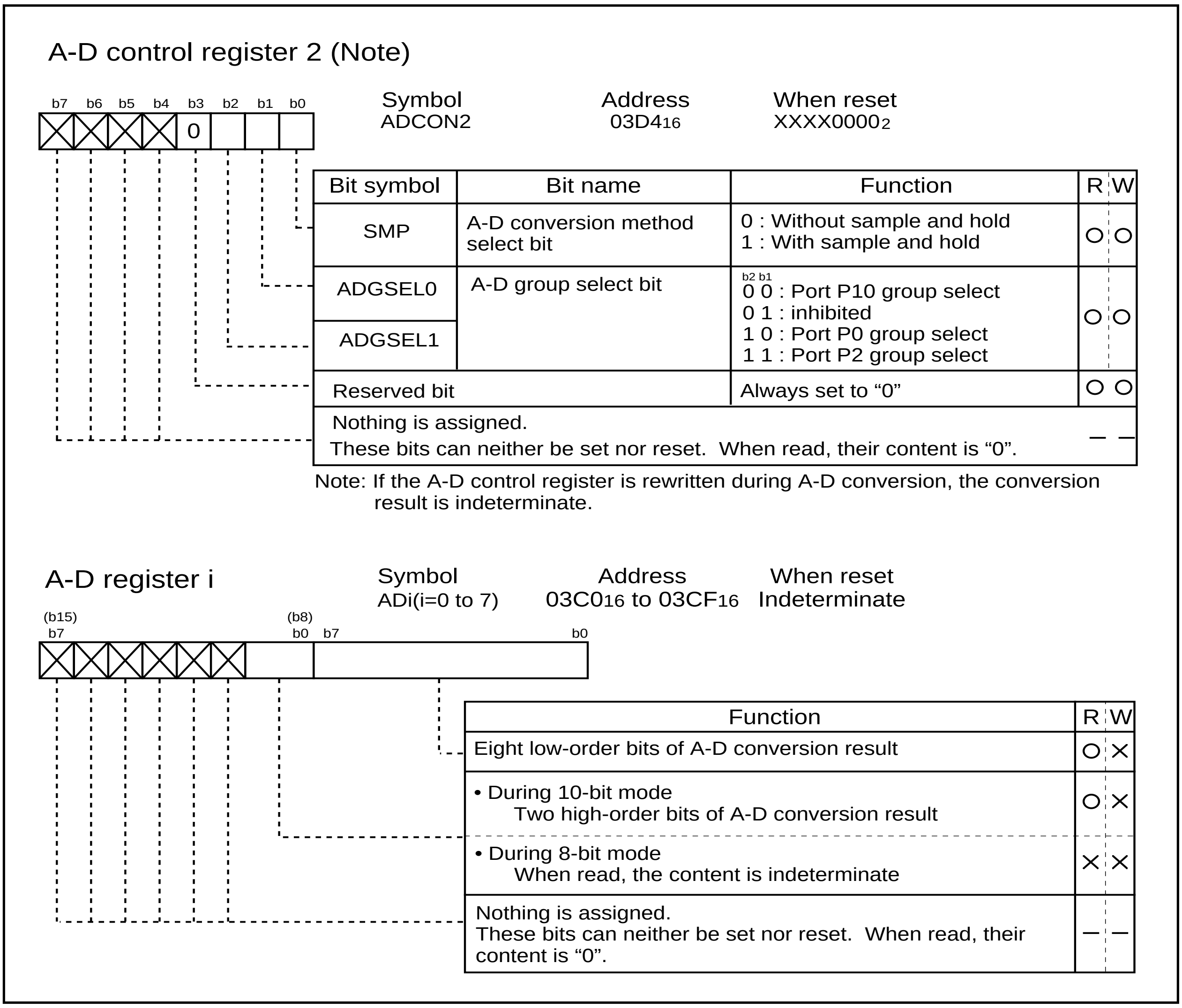
b7 b6 b5 b4 b3 b2 b1 b0

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

Bit symbol	Bit name	Function	R	W
SCAN0	A-D sweep pin select bit	When single sweep and repeat sweep mode 0 are selected 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)	○	○
SCAN1		When repeat sweep mode 1 is selected b1 b0 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)	○	○
MD2	A-D operation mode select bit 1	0 : Any mode other than repeat sweep mode 1 1 : Repeat sweep mode 1	○	○
BITS	8/10-bit mode select bit	0 : 8-bit mode 1 : 10-bit mode	○	○
CKS1	Frequency select bit 1	0 : f_{AD2}/2 or f_{AD2}/4 is selected 1 : f_{AD2} is selected	○	○
VCUT	Vref connect bit	0 : Vref not connected 1 : Vref connected	○	○
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	○	○
OPA1			○	○

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

Figure 15-2. A-D converter-related registers (1)



A-D control register 2 (Note)

Symbol	Address	When reset
ADCON2	03D4$_{16}$	XXXX0000$_2$

Bit symbol	Bit name	Function	R	W
SMP	A-D conversion method select bit	0 : Without sample and hold 1 : With sample and hold	○	○
ADGSEL0	A-D group select bit	b2 b1 0 0 : Port P10 group select 0 1 : inhibited 1 0 : Port P0 group select 1 1 : Port P2 group select	○	○
ADGSEL1				
Reserved bit		Always set to “0”	○	○
Nothing is assigned. These bits can neither be set nor reset. When read, their content is “0”.			—	—

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

A-D register i

Symbol	Address	When reset
ADi(i=0 to 7)	03C0$_{16}$ to 03CF$_{16}$	Indeterminate

Function	R	W
Eight low-order bits of A-D conversion result	○	×
• During 10-bit mode Two high-order bits of A-D conversion result	○	×
• During 8-bit mode When read, the content is indeterminate	×	×
Nothing is assigned. These bits can neither be set nor reset. When read, their content is “0”.	—	—

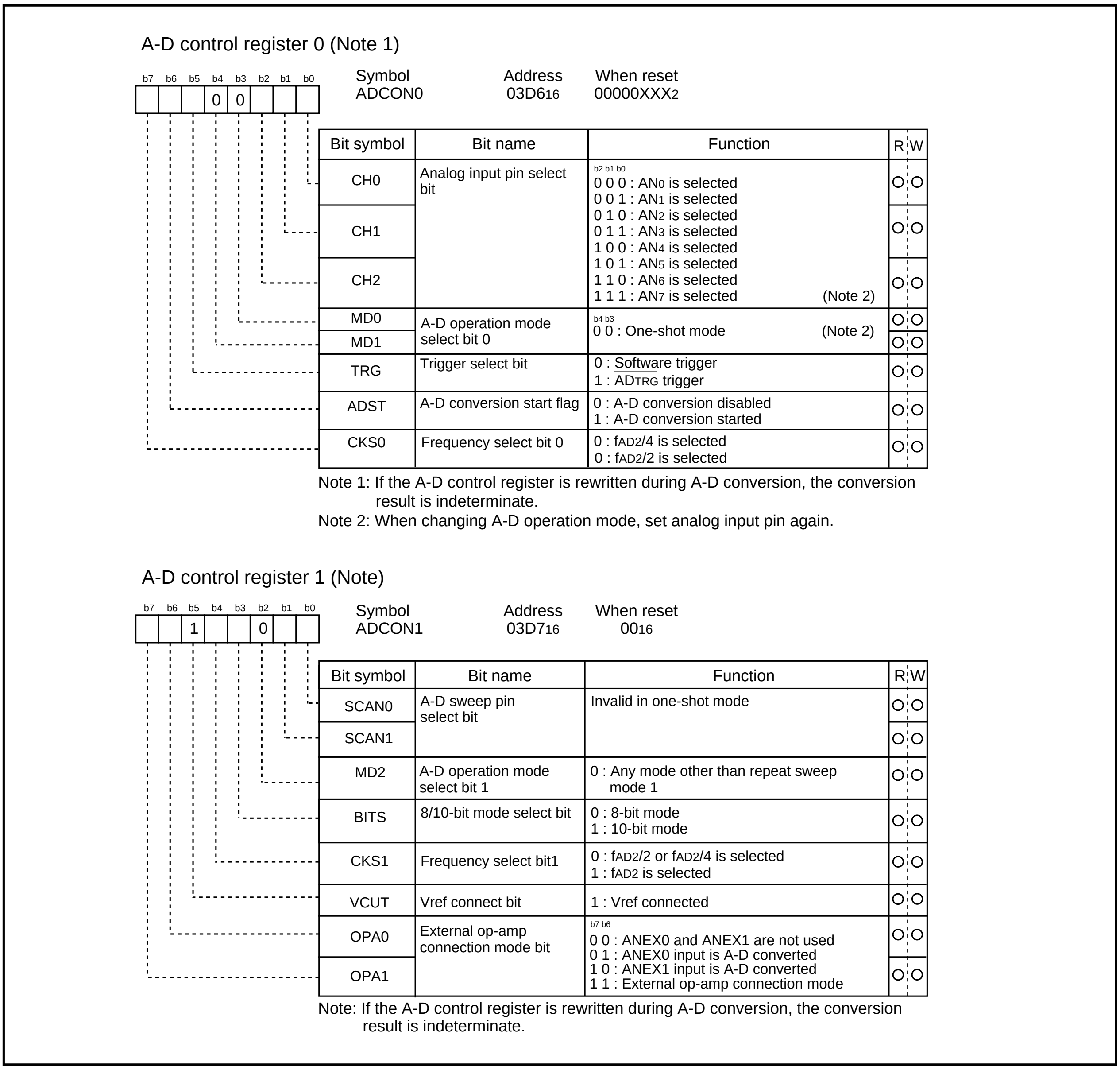
Figure 15-3. A-D converter-related registers (2)

(1) One-shot mode

In one-shot mode, the pin selected using the analog input pin select bit is used for one-shot A-D conversion. Table 15-2 shows the specifications of one-shot mode. Figure 15-4 shows the A-D control register in one-shot mode.

Table 15-2. One-shot mode specifications

Item	Specification
Function	The pin selected by the analog input pin select bit is used for one A-D conversion
Start condition	Writing “1” to A-D conversion start flag
Stop condition	• End of A-D conversion (A-D conversion start flag changes to “0”, except when external trigger is selected) • Writing “0” to A-D conversion start flag
Interrupt request generation timing	End of A-D conversion
Input pin	One of AN0 to AN7, as selected
Reading of result of A-D converter	Read A-D register corresponding to selected pin



A-D control register 0 (Note 1)

Symbol ADCON0 Address 03D6$_{16}$ When reset 00000XXX$_2$

Bit symbol	Bit name	Function	R	W
CH0	Analog input pin select bit	b2 b1 b0 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)	○	○
CH1			○	○
CH2			○	○
MD0	A-D operation mode select bit 0	b4 b3 0 0 : One-shot mode (Note 2)	○	○
MD1			○	○
TRG	Trigger select bit	0 : Software trigger 1 : ADTRG trigger	○	○
ADST	A-D conversion start flag	0 : A-D conversion disabled 1 : A-D conversion started	○	○
CKS0	Frequency select bit 0	0 : fAD2/4 is selected 0 : fAD2/2 is selected	○	○

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

A-D control register 1 (Note)

Symbol ADCON1 Address 03D7$_{16}$ When reset 00$_{16}$

Bit symbol	Bit name	Function	R	W
SCAN0	A-D sweep pin select bit	Invalid in one-shot mode	○	○
SCAN1			○	○
MD2	A-D operation mode select bit 1	0 : Any mode other than repeat sweep mode 1	○	○
BITS	8/10-bit mode select bit	0 : 8-bit mode 1 : 10-bit mode	○	○
CKS1	Frequency select bit1	0 : fAD2/2 or fAD2/4 is selected 1 : fAD2 is selected	○	○
VCUT	Vref connect bit	1 : Vref connected	○	○
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	○	○
OPA1			○	○

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

Figure 15-4. A-D conversion register in one-shot mode

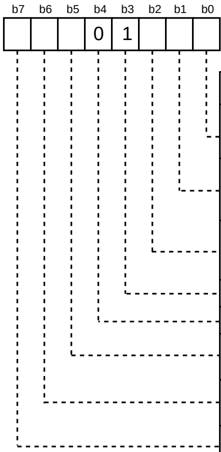
(2) Repeat mode

In repeat mode, the pin selected using the analog input pin select bit is used for repeated A-D conversion. Table 15-3 shows the specifications of repeat mode. Figure 15-5 shows the A-D control register in repeat mode.

Table 15-3. Repeat mode specifications

Item	Specification
Function	The pin selected by the analog input pin select bit is used for repeated A-D conversion
Start condition	Writing “1” to A-D conversion start flag
Stop condition	Writing “0” to A-D conversion start flag
Interrupt request generation timing	None generated
Input pin	One of AN_0 to AN_7, as selected
Reading of result of A-D converter	Read A-D register corresponding to selected pin

A-D control register 0 (Note 1)

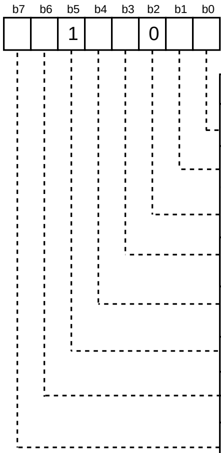


Symbol ADCON0　Address 03D6$_{16}$　When reset 00000XXX$_2$

Bit symbol	Bit name	Function	R	W
CH0	Analog input pin select bit	b2 b1 b0 0 0 0 : AN_0 is selected 0 0 1 : AN_1 is selected 0 1 0 : AN_2 is selected 0 1 1 : AN_3 is selected 1 0 0 : AN_4 is selected 1 0 1 : AN_5 is selected 1 1 0 : AN_6 is selected 1 1 1 : AN_7 is selected (Note 2)	○	○
CH1			○	○
CH2			○	○
MD0	A-D operation mode select bit 0	b4 b3 0 1 : Repeat mode (Note 2)	○	○
MD1			○	○
TRG	Trigger select bit	0 : Software trigger 1 : AD_{TRG} trigger	○	○
ADST	A-D conversion start flag	0 : A-D conversion disabled 1 : A-D conversion started	○	○
CKS0	Frequency select bit 0	0 : f_{AD2}/4 is selected 1 : f_{AD2}/2 is selected	○	○

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

A-D control register 1 (Note)



Symbol ADCON1　Address 03D7$_{16}$　When reset 00$_{16}$

Bit symbol	Bit name	Function	R	W
SCAN0	A-D sweep pin select bit	Invalid in repeat mode	○	○
SCAN1			○	○
MD2	A-D operation mode select bit 1	0 : Any mode other than repeat sweep mode 1	○	○
BITS	8/10-bit mode select bit	0 : 8-bit mode 1 : 10-bit mode	○	○
CKS1	Frequency select bit 1	0 : f_{AD2}/2 or f_{AD2}/4 is selected 1 : f_{AD2} is selected	○	○
VCUT	Vref connect bit	1 : Vref connected	○	○
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	○	○
OPA1			○	○

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

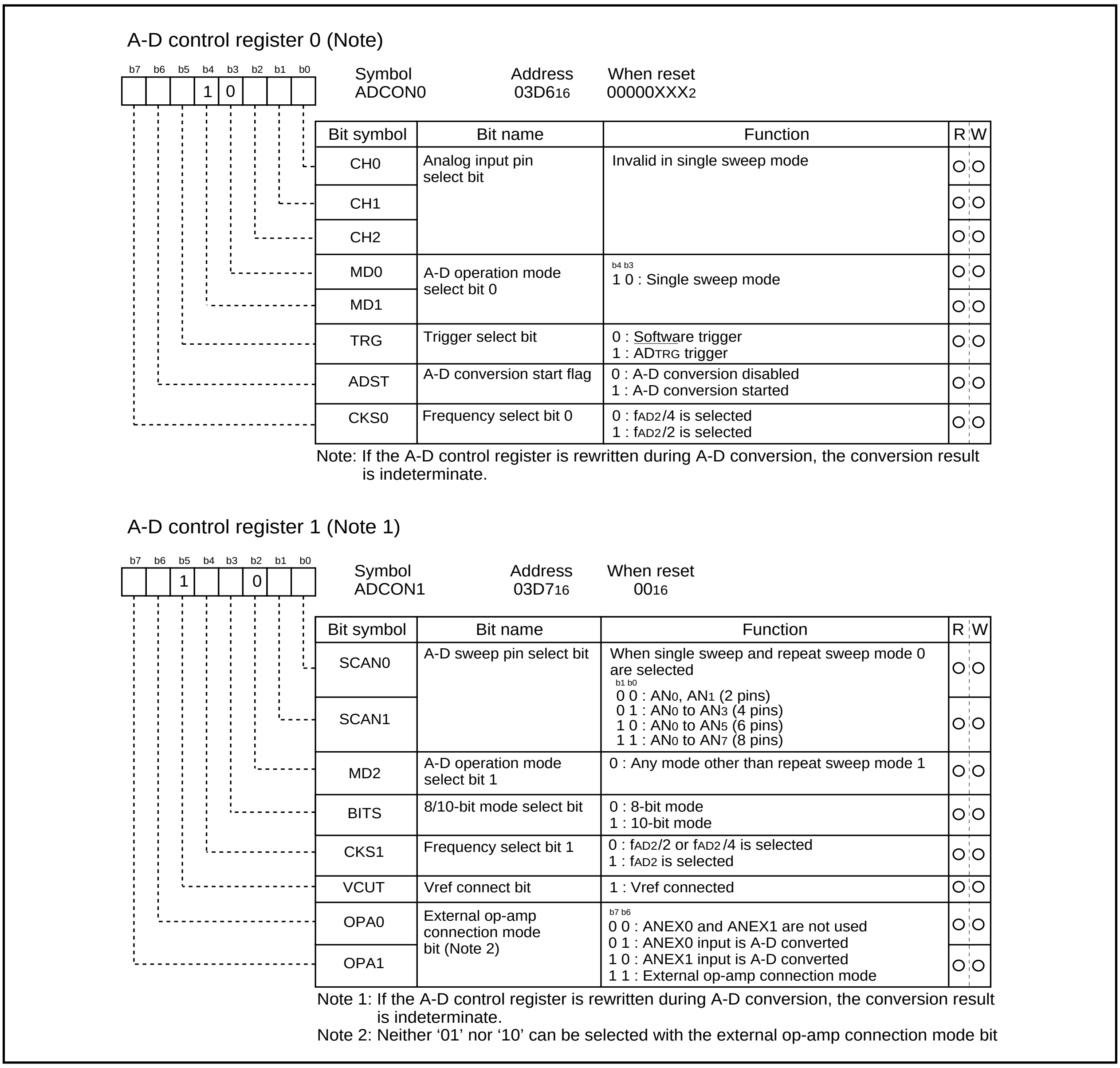
Figure 15-5. A-D conversion register in repeat mode

(3) Single sweep mode

In single sweep mode, the pins selected using the A-D sweep pin select bit are used for one-by-one A-D conversion. Table 15-4 shows the specifications of single sweep mode. Figure 15-6 shows the A-D control register in single sweep mode.

Table 15-4. Single sweep mode specifications

Item	Specification
Function	The pins selected by the A-D sweep pin select bit are used for one-by-one A-D conversion
Start condition	Writing “1” to A-D converter start flag
Stop condition	• End of A-D conversion (A-D conversion start flag changes to “0”, except when external trigger is selected) • Writing “0” to A-D conversion start flag
Interrupt request generation timing	End of A-D conversion
Input pin	AN_0 and AN_1 (2 pins), AN_0 to AN_3 (4 pins), AN_0 to AN_5 (6 pins), or AN_0 to AN_7 (8 pins)
Reading of result of A-D converter	Read A-D register corresponding to selected pin



A-D control register 0 (Note)

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

Bit symbol	Bit name	Function	R	W
CH0	Analog input pin select bit	Invalid in single sweep mode	○	○
CH1			○	○
CH2			○	○
MD0	A-D operation mode select bit 0	b4 b3 1 0 : Single sweep mode	○	○
MD1			○	○
TRG	Trigger select bit	0 : Software trigger 1 : AD_{TRG} trigger	○	○
ADST	A-D conversion start flag	0 : A-D conversion disabled 1 : A-D conversion started	○	○
CKS0	Frequency select bit 0	0 : f_{AD2}/4 is selected 1 : f_{AD2}/2 is selected	○	○

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

A-D control register 1 (Note 1)

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

Bit symbol	Bit name	Function	R	W
SCAN0	A-D sweep pin select bit	When single sweep and repeat sweep mode 0 are selected 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)	○	○
SCAN1			○	○
MD2	A-D operation mode select bit 1	0 : Any mode other than repeat sweep mode 1	○	○
BITS	8/10-bit mode select bit	0 : 8-bit mode 1 : 10-bit mode	○	○
CKS1	Frequency select bit 1	0 : f_{AD2}/2 or f_{AD2}/4 is selected 1 : f_{AD2} is selected	○	○
VCUT	Vref connect bit	1 : Vref connected	○	○
OPA0	External op-amp connection mode bit (Note 2)	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	○	○
OPA1			○	○

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

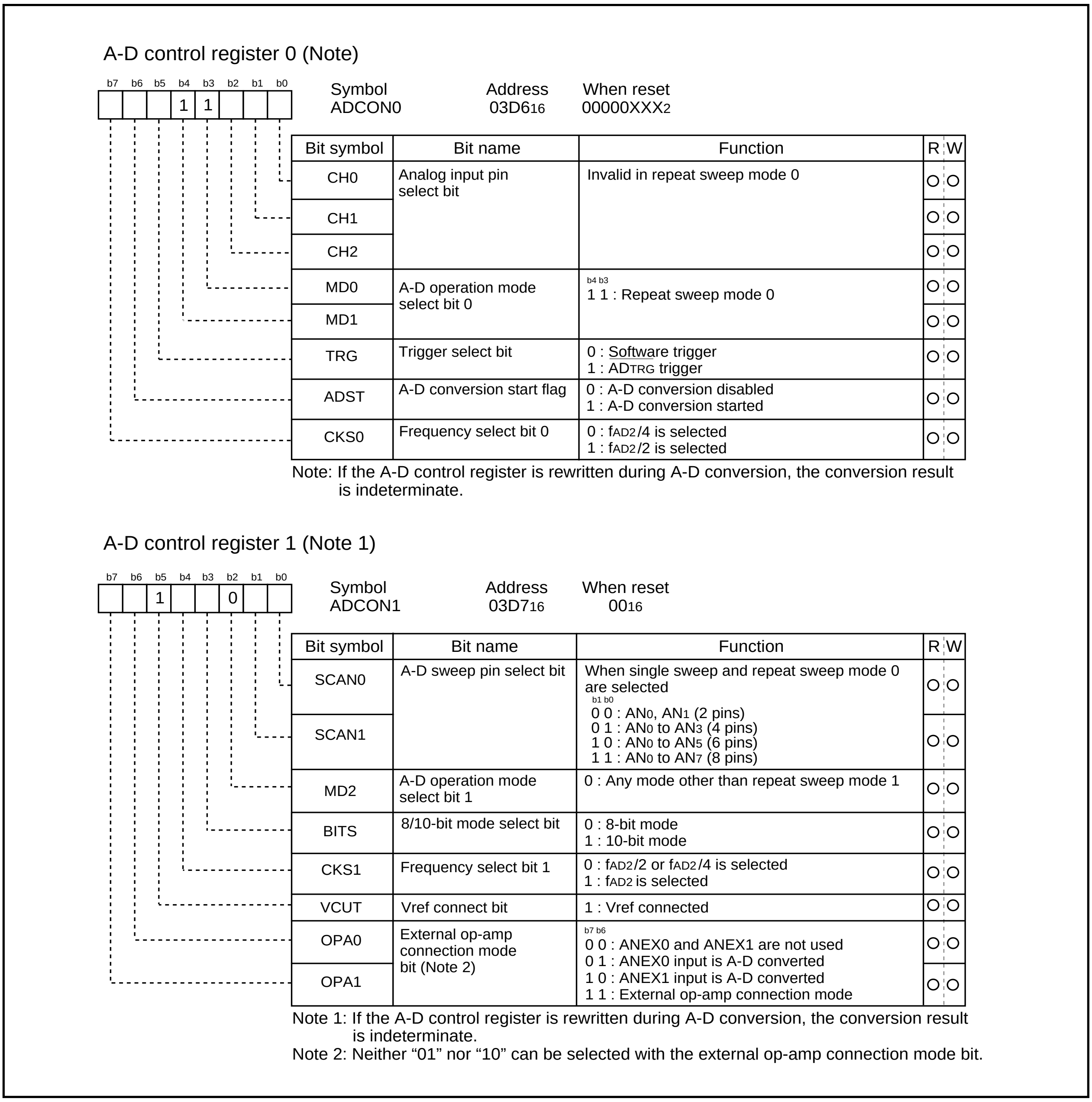
Figure 15-6. A-D conversion register in single sweep mode

(4) Repeat sweep mode 0

In repeat sweep mode 0, the pins selected using the A-D sweep pin select bit are used for repeat sweep A-D conversion. Table 15-5 shows the specifications of repeat sweep mode 0. Figure 15-7 shows the A-D control register in repeat sweep mode 0.

Table 15-5. Repeat sweep mode 0 specifications

Item	Specification
Function	The pins selected by the A-D sweep pin select bit are used for repeat sweep A-D conversion
Start condition	Writing “1” to A-D conversion start flag
Stop condition	Writing “0” to A-D conversion start flag
Interrupt request generation timing	None generated
Input pin	AN_0 and AN_1 (2 pins), AN_0 to AN_3 (4 pins), AN_0 to AN_5 (6 pins), or AN_0 to AN_7 (8 pins)
Reading of result of A-D converter	Read A-D register corresponding to selected pin (at any time)



A-D control register 0 (Note)

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

Bit symbol	Bit name	Function	R	W
CH0	Analog input pin select bit	Invalid in repeat sweep mode 0	○	○
CH1			○	○
CH2			○	○
MD0	A-D operation mode select bit 0	b4 b3 1 1 : Repeat sweep mode 0	○	○
MD1			○	○
TRG	Trigger select bit	0 : Software trigger 1 : AD_{TRG} trigger	○	○
ADST	A-D conversion start flag	0 : A-D conversion disabled 1 : A-D conversion started	○	○
CKS0	Frequency select bit 0	0 : f_{AD2}/4 is selected 1 : f_{AD2}/2 is selected	○	○

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

A-D control register 1 (Note 1)

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

Bit symbol	Bit name	Function	R	W
SCAN0	A-D sweep pin select bit	When single sweep and repeat sweep mode 0 are selected 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)	○	○
SCAN1			○	○
MD2	A-D operation mode select bit 1	0 : Any mode other than repeat sweep mode 1	○	○
BITS	8/10-bit mode select bit	0 : 8-bit mode 1 : 10-bit mode	○	○
CKS1	Frequency select bit 1	0 : f_{AD2}/2 or f_{AD2}/4 is selected 1 : f_{AD2} is selected	○	○
VCUT	Vref connect bit	1 : Vref connected	○	○
OPA0	External op-amp connection mode bit (Note 2)	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	○	○
OPA1			○	○

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

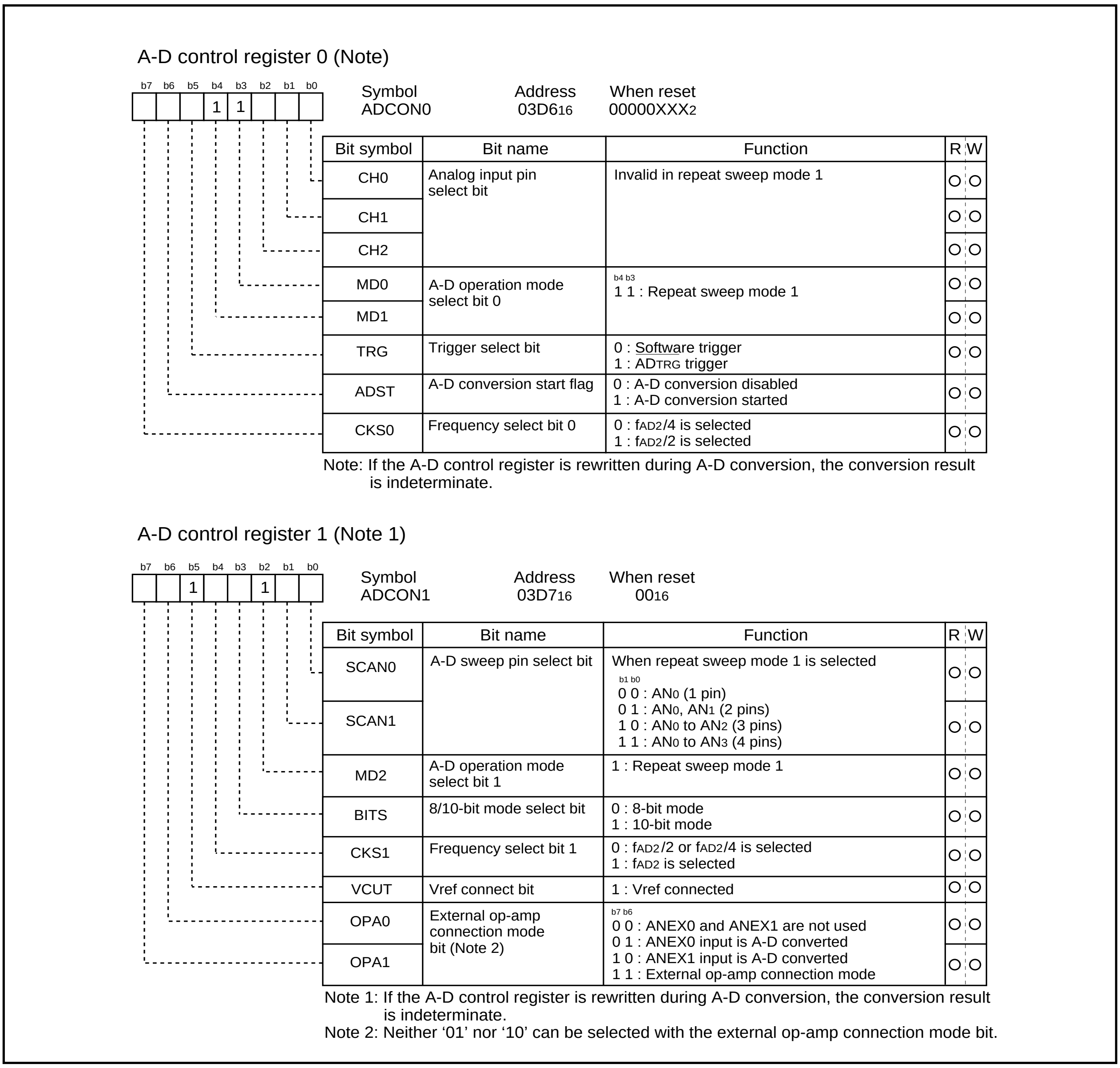
Figure 15-7. A-D conversion register in repeat sweep mode 0

(5) Repeat sweep mode 1

In repeat sweep mode 1, all pins are used for A-D conversion with emphasis on the pin or pins selected using the A-D sweep pin select bit. Table 15-6 shows the specifications of repeat sweep mode 1. Figure 15-8 shows the A-D control register in repeat sweep mode 1.

Table 15-6. Repeat sweep mode 1 specifications

Item	Specification
Function	All pins perform repeat sweep A-D conversion, with emphasis on the pin or pins selected by the A-D sweep pin select bit Example : AN_0 selected AN_0 – AN_1 – AN_0 – AN_2 – AN_0 – AN_3, etc
Start condition	Writing “1” to A-D conversion start flag
Stop condition	Writing “0” to A-D conversion start flag
Interrupt request generation timing	None generated
Input pin	AN_0 (1 pin), AN_0 and AN_1 (2 pins), AN_0 to AN_2 (3 pins), AN_0 to AN_3 (4 pins)
Reading of result of A-D converter	Read A-D register corresponding to selected pin (at any time)



A-D control register 0 (Note)

b7	b6	b5	b4	b3	b2	b1	b0
			1	1			

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

Bit symbol	Bit name	Function	R	W
CH0	Analog input pin select bit	Invalid in repeat sweep mode 1	○	○
CH1			○	○
CH2			○	○
MD0	A-D operation mode select bit 0	b4 b3 1 1 : Repeat sweep mode 1	○	○
MD1			○	○
TRG	Trigger select bit	0 : Software trigger 1 : ADTRG trigger	○	○
ADST	A-D conversion start flag	0 : A-D conversion disabled 1 : A-D conversion started	○	○
CKS0	Frequency select bit 0	0 : $f_{AD2}/4$ is selected 1 : $f_{AD2}/2$ is selected	○	○

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

A-D control register 1 (Note 1)

b7	b6	b5	b4	b3	b2	b1	b0
		1			1		

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

Bit symbol	Bit name	Function	R	W
SCAN0	A-D sweep pin select bit	When repeat sweep mode 1 is selected b1 b0 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)	○	○
SCAN1			○	○
MD2	A-D operation mode select bit 1	1 : Repeat sweep mode 1	○	○
BITS	8/10-bit mode select bit	0 : 8-bit mode 1 : 10-bit mode	○	○
CKS1	Frequency select bit 1	0 : $f_{AD2}/2$ or $f_{AD2}/4$ is selected 1 : f_{AD2} is selected	○	○
VCUT	Vref connect bit	1 : Vref connected	○	○
OPA0	External op-amp connection mode bit (Note 2)	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	○	○
OPA1			○	○

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

Figure 15-8. A-D conversion register in repeat sweep mode 1

(a) Sample and hold

Sample and hold is selected by setting bit 0 of the A-D control register 2 (address 03D4$_{16}$) to “1”. When sample and hold is selected, the rate of conversion of each pin increases. As a result, a 28 f AD cycle is achieved with 8-bit resolution and 33 f AD with 10-bit resolution. Sample and hold can be selected in all modes. However, in all modes, be sure to specify before starting A-D conversion whether sample and hold is to be used.

(b) Extended analog input pins

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

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

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

(c) External operation amp connection mode

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

When bit 6 of the A-D control register 1 (address 03D7$_{16}$) is “1” and bit 7 is “1”, input via AN$_0$ to AN$_7$ is output from ANEX0. The input from ANEX1 is converted from analog to digital and the result stored in the corresponding A-D register. The speed of A-D conversion depends on the response of the external operation amp. Do not connect the ANEX0 and ANEX1 pins directly. Figure 15-9 is an example of how to connect the pins in external operation amp mode.

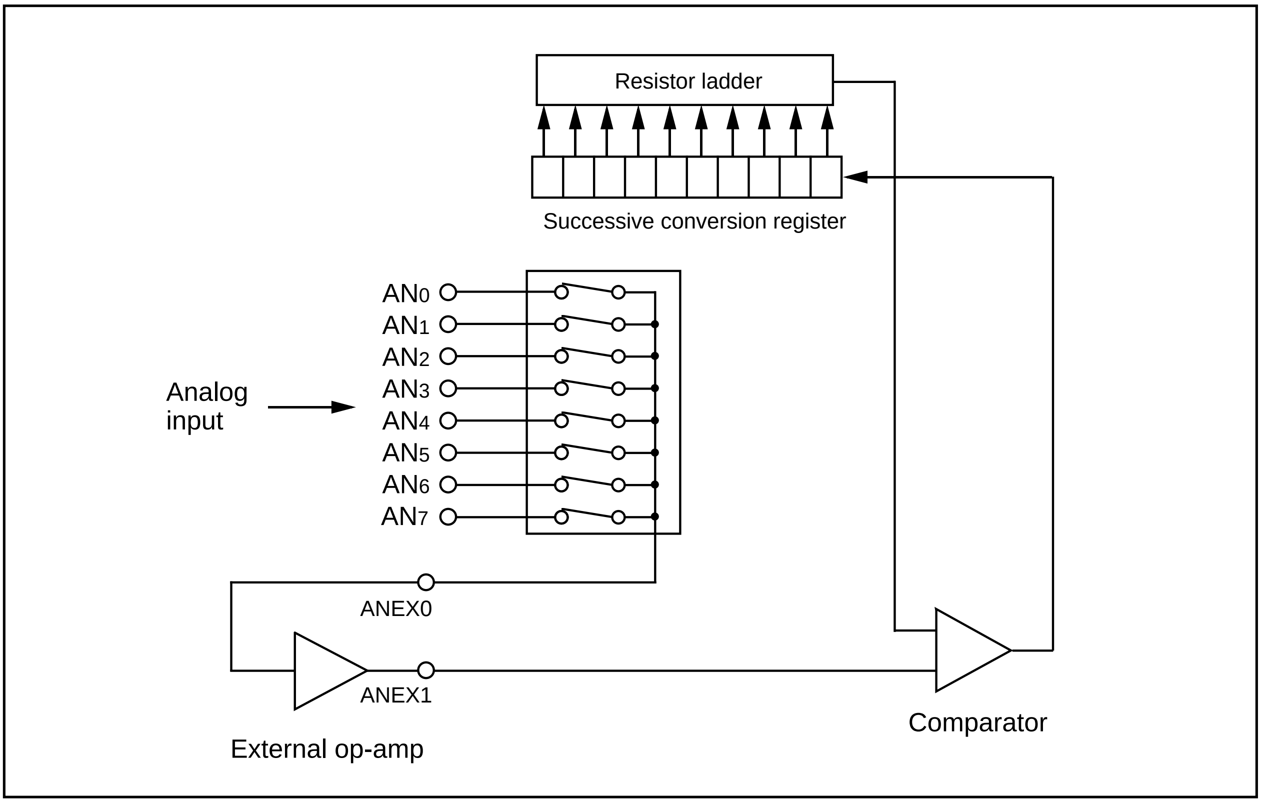


Figure 15-9. Example of external op-amp connection mode

D-A Converter

This is an 8-bit, R-2R type D-A converter. The microcomputer contains two independent D-A converters of this type.

D-A conversion is performed when a value is written to the corresponding D-A register. Bits 0 and 1 (D-A output enable bits) of the D-A control register decide if the result of conversion is to be output. Do not set the target port to output mode if D-A conversion is to be performed.

Output analog voltage (V) is determined by a set value (n : decimal) in the D-A register.

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

V_{REF} : reference voltage

Table 16-1 lists the performance of the D-A converter. Figure 16-1 shows the block diagram of the D-A converter. Figure 16-2 shows the D-A control register.

Table 16-1. Performance of D-A converter

Item	Performance
Conversion method	R-2R method
Resolution	8 bits
Analog output pin	2 channels

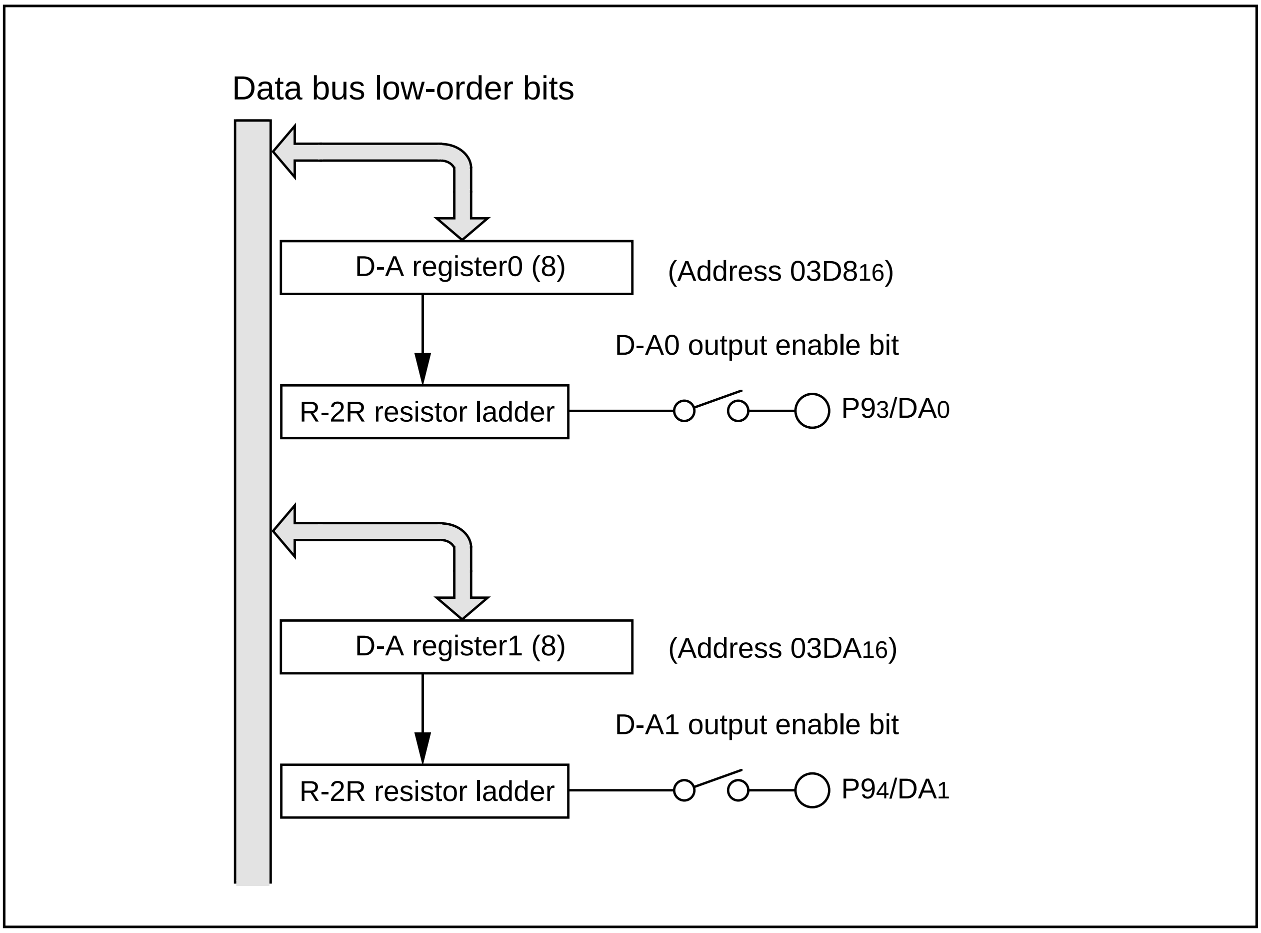


Figure 16-1. Block diagram of D-A converter

D-A control register

b7 b6 b5 b4 b3 b2 b1 b0

Symbol	Address	When reset
DACON	03DC16	0016

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

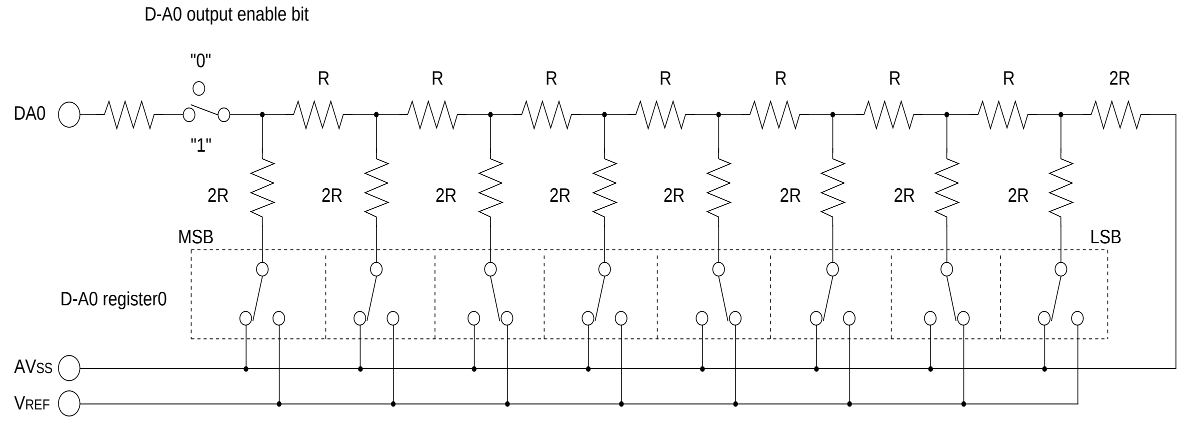
D-A register

b7 b0

Symbol	Address	When reset
DAi (i = 0,1)	03D816, 03DA16	Indeterminate

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

Figure 16-2. D-A control register



Note 1: The above diagram shows an instance in which the D-A register is assigned 2A16.
Note 2: The same circuit as this is also used for D-A1.
Note 3: To reduce the current consumption when the D-A converter is not used, set the D-A output enable bit to 0 and set the D-A register to 0016 so that no current flows in the resistors Rs and 2Rs.

Figure 16-3. D-A converter equivalent circuit

CRC Calculation Circuit

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

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

Figure 17-1 shows the block diagram of the CRC circuit. Figure 17-2 shows the CRC-related registers.

Figure 17-3 shows the calculation example using the CRC calculation circuit.

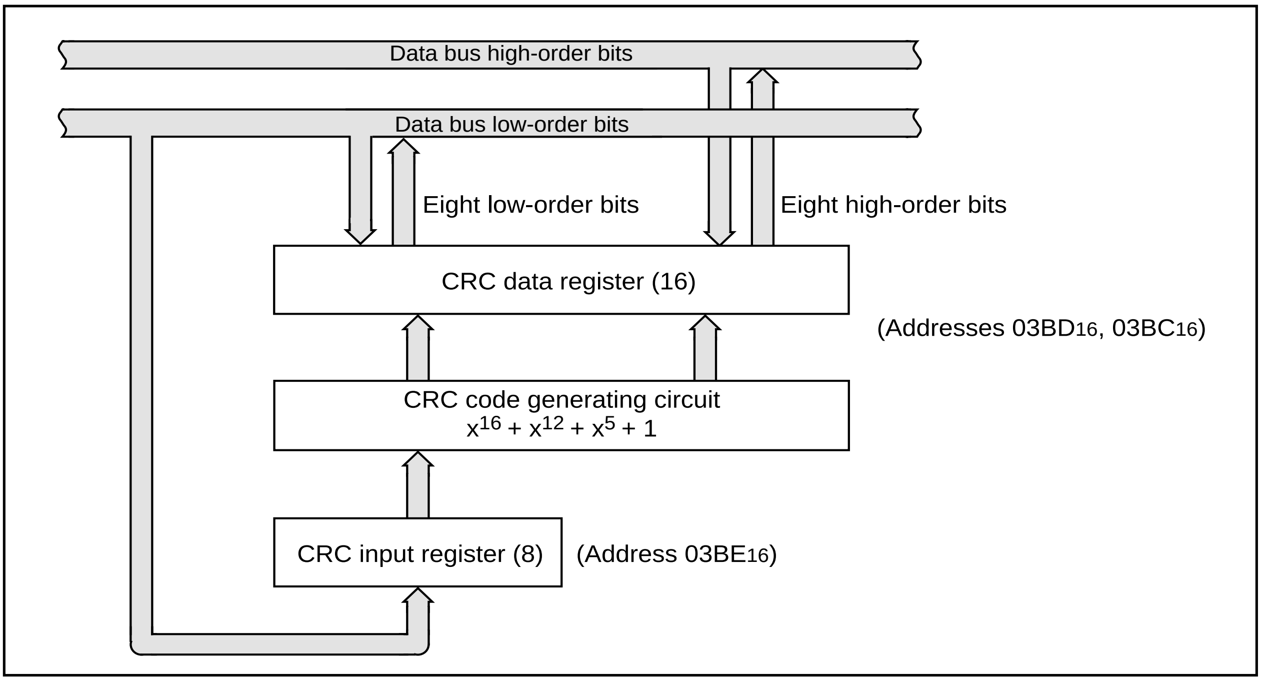
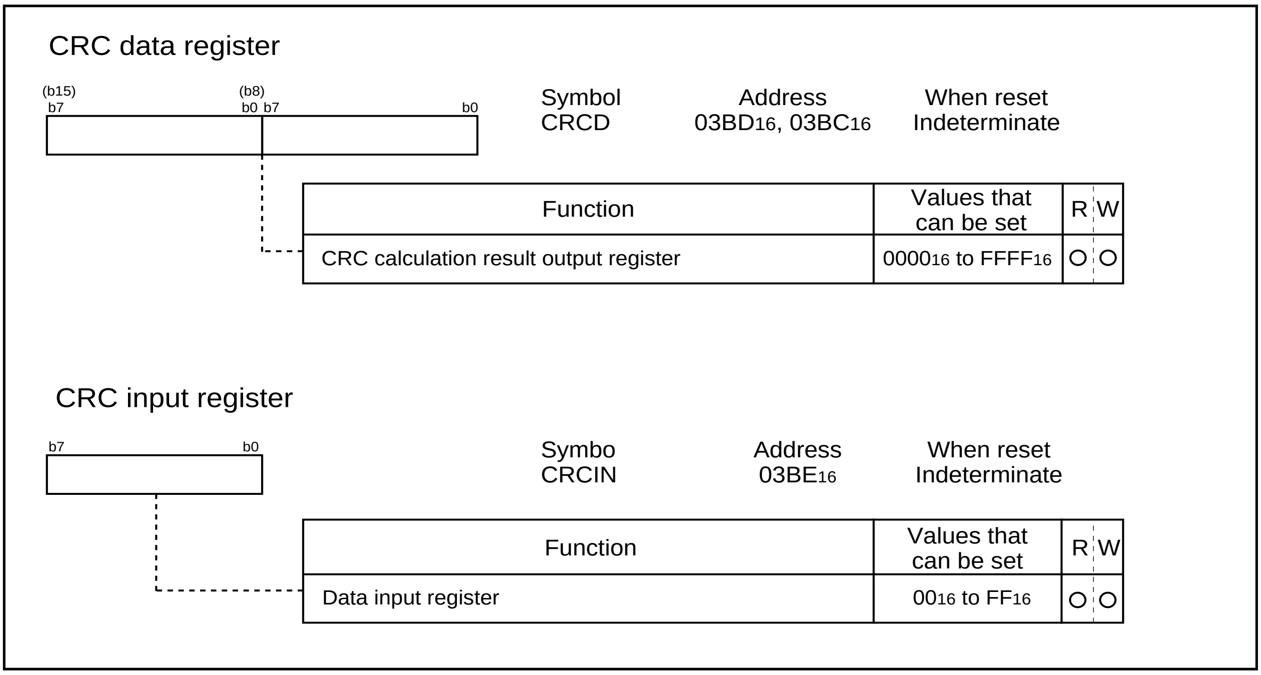


Figure 17-1. Block diagram of CRC circuit



CRC data register

Symbol	Address	When reset
CRCD	$03BD_{16}$, $03BC_{16}$	Indeterminate

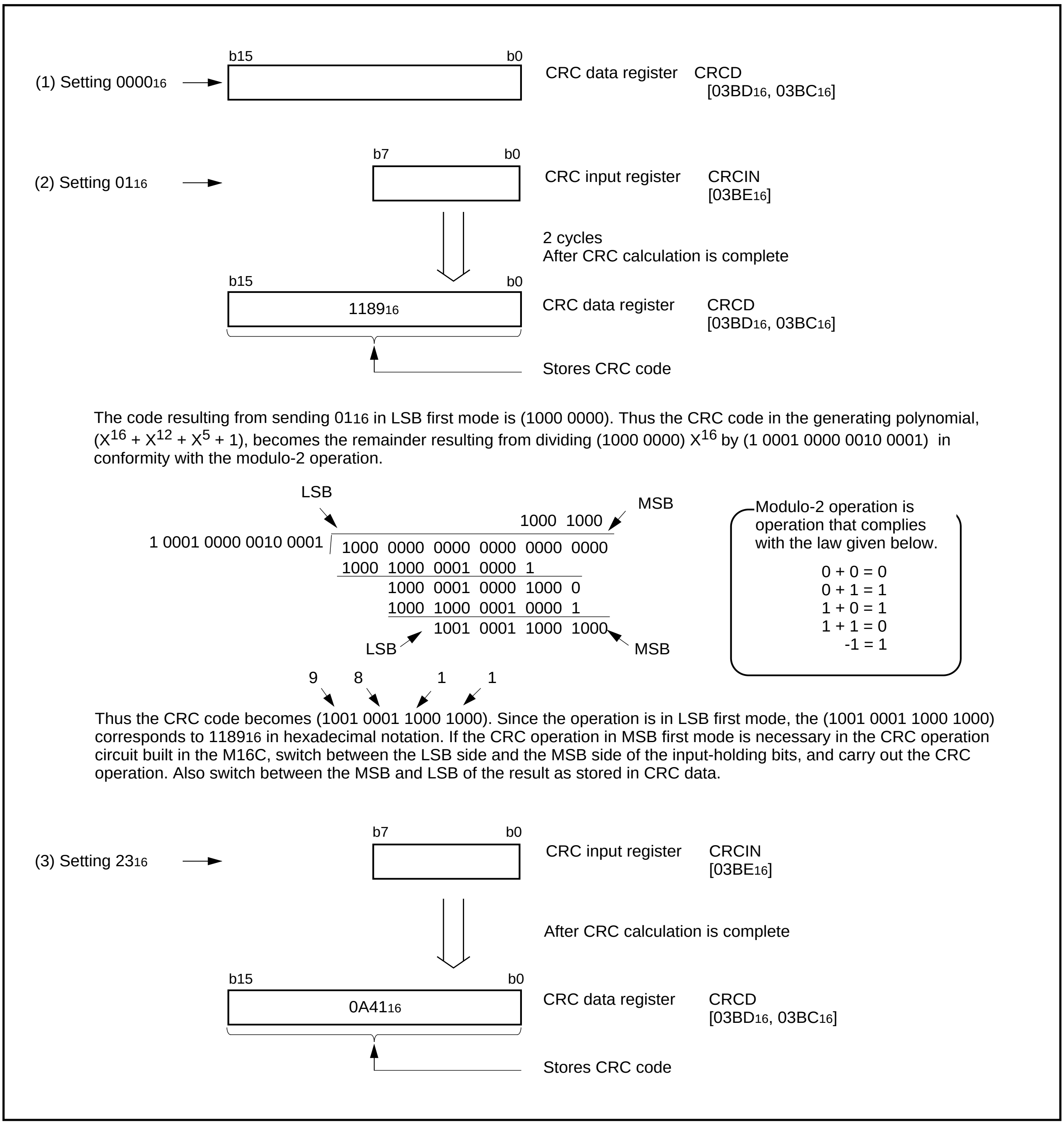
Function	Values that can be set	R	W
CRC calculation result output register	0000_{16} to $FFFF_{16}$	○	○

CRC input register

Symbo	Address	When reset
CRCIN	$03BE_{16}$	Indeterminate

Function	Values that can be set	R	W
Data input register	00_{16} to FF_{16}	○	○

Figure 17-2. CRC-related registers



The code resulting from sending 01₁₆ in LSB first mode is (1000 0000). Thus the CRC code in the generating polynomial, ($X^{16} + X^{12} + X^5 + 1$), becomes the remainder resulting from dividing (1000 0000) X^{16} by (1 0001 0000 0010 0001) in conformity with the modulo-2 operation.

Thus the CRC code becomes (1001 0001 1000 1000). Since the operation is in LSB first mode, the (1001 0001 1000 1000) corresponds to 1189₁₆ in hexadecimal notation. If the CRC operation in MSB first mode is necessary in the CRC operation circuit built in the M16C, switch between the LSB side and the MSB side of the input-holding bits, and carry out the CRC operation. Also switch between the MSB and LSB of the result as stored in CRC data.

Figure 17-3. Calculation example using the CRC calculation circuit

CAN Module 0/1

The CAN module provides the CAN (Controller Area Network) serial-bus data communication interface. This asynchronous communication protocol is used in distributed systems, such as automotive and industrial control systems and where high-speed processing and data exchange are required with a very high level of security. This module supports data transfer rates up to 1 Mbps.

According to the BOSCH 2.0B CAN protocol specification, the CAN module can handle and process both the standard and extended identifier message formats. For more details, refer to the BOSCH CAN Specification 2.0B, hereinafter referred to as CAN specification.

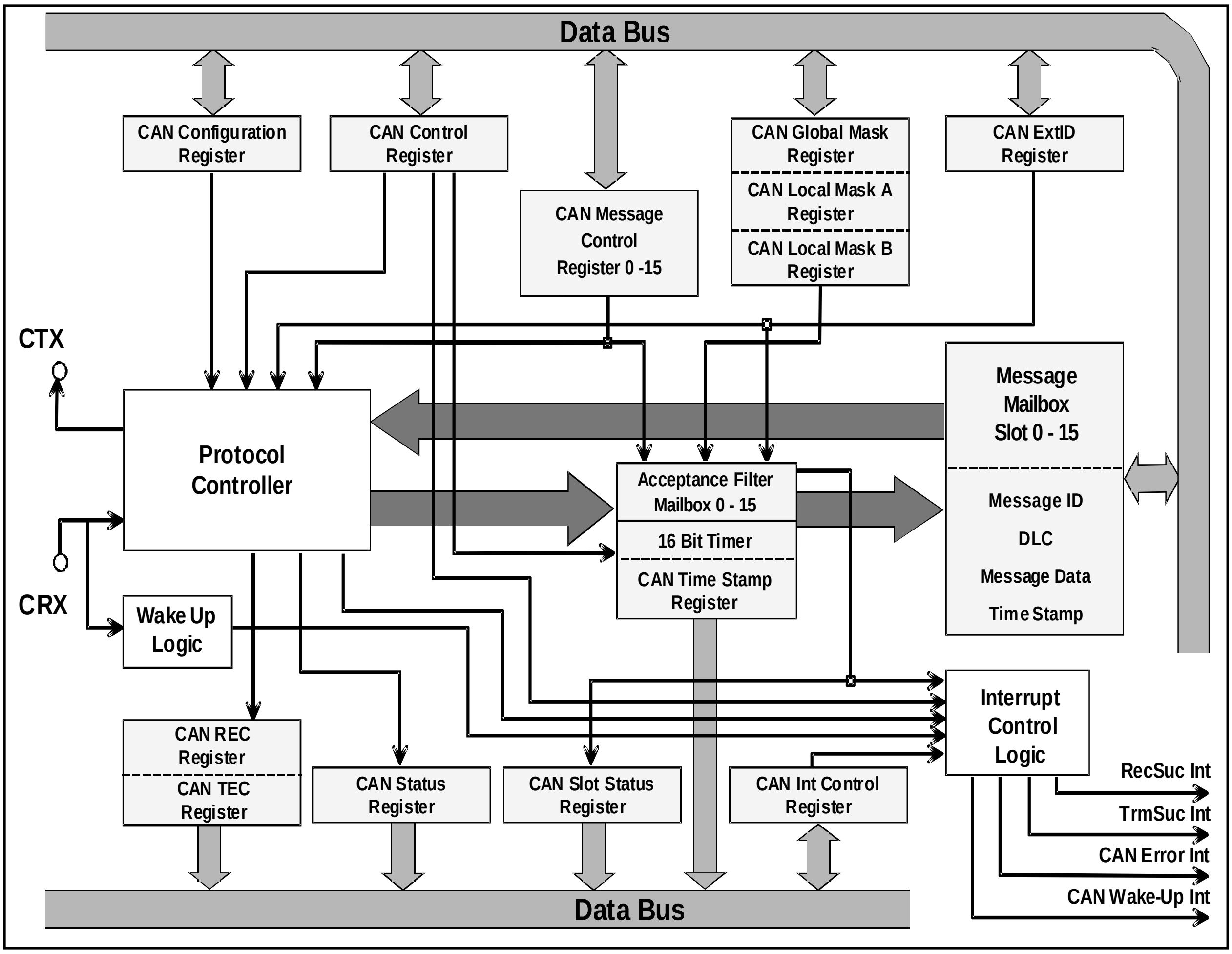


Figure 18-1. Block diagram of one CAN module

Figure 18-1 shows a block diagram of the M16C CAN module. The main functional blocks in this description are:

Protocol Controller: This controller handles the bus arbitration and the CAN serial communication protocol message transmission and reception services, i.e. bit stuffing, CRC, error status etc.

Message Mailbox: This memory block consists of several message slots which can be configured to act either as a transmit- or receive message box. Each slot consists of a relevant identifier, data length code, a data field (8 bytes) and a communication time stamp. This message slot time stamp value corresponds to the instant of time (event) when the Protocol Controller indicates a successful CAN message reception.

Acceptance Filter: This block performs the comparison between the identifier of the received message and the key identifier of all receive slots. For this acceptance filter, users can define the content of special mask registers to filter a range of identifier for the corresponding message slot.

16 bit Timer: This 16 bit timer is used for a time stamp function. The timer provides the counter status which will be stored together with the received message in the message mailbox.

Wake Up Logic:	The MCU can be set to stop- or wait mode to reduce power consumption. This module provides the possibility to wake up the MCU from sleep mode via the CAN bus (refer to section CAN wake up interrupt).
Interrupt Generation:	The CAN module signals the CPU different events via 6 interrupts.

Four interrupt channels are used for successful CAN message transmission and reception indication, i.e. 'message receive successful' interrupt (C0RECIC/C1RECIC) and 'message transmit successful' interrupt (C0TRMIC/ C1TRMIC).

One interrupt signals if the CAN module enters an error operating state (C01ERRIC), i.e. 'error passive', 'bus off' and if any CAN bus error occurred in the communication process. The CAN bus error interrupt generation can be individually disabled in the CAN Control Register.

The wake up case will also be flagged to the CPU by an additional interrupt line (C01WKPIC).

Interrupts

- **6 Interrupts**

CAN0- Successful Transmission Interrupt

CAN0- Successful Reception Interrupt

CAN1- Successful Transmission Interrupt

CAN1- Successful Reception Interrupt

CAN0/1- Error Interrupt

- Error Passive State
- Error BusOff State
- Bus Error (this feature could be disabled separately)

CAN0/1- Wake Up Interrupt

When the CPU detects an Successful Transmission/Reception Interrupt, the CAN Status Register must be read to determine which Mailbox has issued the interrupt.

Memory Map of the CAN0/1 Special Function Registers

This memory map is valid for both CAN channels (CAN0 and CAN1)

- **CAN Mailboxes**

- 16 message slots (each mailbox comprises 16 bytes (8 words))
- fixed mailbox-organization
- 'Basic CAN'-feature is composed of two regular CAN slots (#14/15) - This feature is implemented as an option.

- **CAN Mask Registers**

- 3 masks for the acceptance filter (refer to section 'Mask Register and Acceptance Filter') (each mask comprises 6 bytes)

- **CAN SFR Registers**

- 9 CAN Special Function Registers

Control Register	(16 Bits): controls the CAN module.
Status Register	(16 Bits): displays the status of the CAN module.
Slot Status Register	(16 Bits): for each slot, the current content status is monitored.
Interrupt Control Register	(16 Bits): for each slot, the interrupts can be disabled.
Extended ID Register	(16 Bits): distinguishes between ExtendedID and StandardID mailboxes.
Configuration Register	(16 Bits): configuration of the bus timing
REC Register	(8 Bits) : receive error counter of the CAN module
TEC Register	(8 Bits) : transmit error counter of the CAN module
Time Stamp Register	(16 Bits): time stamp counter

Memory Map of a Message Object

Address	Content	
	Byte Order (8 Bits)	Word Order (16 Bits)
0060_{16} + x•16 + 0	StdID [10 to 6]	StdID [5 to 0]
0060_{16} + x•16 + 1	StdID [5 to 0]	StdID [10 to 6]
0060_{16} + x•16 + 2	ExtID [17 to 14]	ExtID [13 to 6]
0060_{16} + x•16 + 3	ExtID [13 to 6]	ExtID [17 to 14]
0060_{16} + x•16 + 4	ExtID [5 to 0]	DLC
0060_{16} + x•16 + 5	DLC	ExtID [5 to 0]
0060_{16} + x•16 + 6	Data Byte 0	Data Byte 1
0060_{16} + x•16 + 7	Data Byte 1	Data Byte 0
...	...	...
0060_{16} + x•16 + 13	Data Byte 7	Data Byte 6
0060_{16} + x•16 + 14	Time Stamp Upper Byte	Time Stamp Lower Byte
0060_{16} + x•16 + 15	Time Stamp Lower Byte	Time Stamp Upper Byte

Note: x: Number of message slot (x = 0 to 15)

Table 18-1. Message object overview (CAN0)

Address	Content	
	Byte Order (8 Bits)	Word Order (16 Bits)
0260_{16} + x•16 + 0	StdID [10 to 6]	StdID [5 to 0]
0260_{16} + x•16 + 1	StdID [5 to 0]	StdID [10 to 6]
0260_{16} + x•16 + 2	ExtID [17 to 14]	ExtID [13 to 6]
0260_{16} + x•16 + 3	ExtID [13 to 6]	ExtID [17 to 14]
0260_{16} + x•16 + 4	ExtID [5 to 0]	DLC
0260_{16} + x•16 + 5	DLC	ExtID [5 to 0]
0260_{16} + x•16 + 6	Data Byte 0	Data Byte 1
0260_{16} + x•16 + 7	Data Byte 1	Data Byte 0
...	...	...
0260_{16} + x•16 + 13	Data Byte 7	Data Byte 6
0260_{16} + x•16 + 14	Time Stamp Upper Byte	Time Stamp Lower Byte
0260_{16} + x•16 + 15	Time Stamp Lower Byte	Time Stamp Upper Byte

Note: x: Number of message slot (x = 0 to 15)

Table 18-2. Message object overview (CAN1)

To access the message memory, either linear address order (byte access) or crossed address order (word access), which supports word access especially for even addresses, can be selected. The location of the message object bytes depends on the Message Order control bit (MsgOrder), which selects byte- or word address order. Refer also to section 'CAN Control Register'.

CAN Message Objects

Data can be written in the grey shaded bits in the identifier bytes. But in the case of read-process, the value of these bits will be set to "0" if a successful receive process is performed for the corresponding slot. In case of no message storage by the CAN module, these bits contain their previous values (written by the CPU).
Please note the meaning of byte order in the message object. This order corresponds directly with the data field on the bus in chronological order (see Figures. 18-2/18-3 CAN data frames).

Bit 7							Bit 0	
▨	▨	▨	SID$_{10}$	SID$_{9}$	SID$_{8}$	SID$_{7}$	SID$_{6}$	StdID [10 to 6]
▨	▨	SID$_{5}$	SID$_{4}$	SID$_{3}$	SID$_{2}$	SID$_{1}$	SID$_{0}$	StdID [5 to 0]
▨	▨	▨	▨	EID$_{17}$	EID$_{16}$	EID$_{15}$	EID$_{14}$	ExtID [17 to 14]
EID$_{13}$	EID$_{12}$	EID$_{11}$	EID$_{10}$	EID$_{9}$	EID$_{8}$	EID$_{7}$	EID$_{6}$	ExtID [13 to 6]
▨	▨	EID$_{5}$	EID$_{4}$	EID$_{3}$	EID$_{2}$	EID$_{1}$	EID$_{0}$	ExtID [5 to 0]
▨	▨	▨	▨	DLC$_{3}$	DLC$_{2}$	DLC$_{1}$	DLC$_{0}$	DLC
Data Byte 0								
Data Byte 1								
...								
Data Byte 7								
Time Stamp [15 to 8]								Time Stamp Upper Byte
Time Stamp [7 to 0]								Time Stamp Lower Byte

CAN Data Frame:

StdID[10 to 6]	StdID[5 to 0]	ExtID[17 to 14]	ExtID[13 to 6]	ExtID[5 to 0]	DLC[3 to 0]	Data 0[7 to 0]	Data 1[7 to 0]		Data 7[7 to 0]

Figure 18-2. Bit organization of the message objects for byte access

Bit 15							Bit 8	Bit 7							Bit 0
▨	▨	▨	SID$_{10}$	SID$_{9}$	SID$_{8}$	SID$_{7}$	SID$_{6}$	▨	▨	SID$_{5}$	SID$_{4}$	SID$_{3}$	SID$_{2}$	SID$_{1}$	SID$_{0}$
▨	▨	▨	▨	EID$_{17}$	EID$_{16}$	EID$_{15}$	EID$_{14}$	EID$_{13}$	EID$_{12}$	EID$_{11}$	EID$_{10}$	EID$_{9}$	EID$_{8}$	EID$_{7}$	EID$_{6}$
▨	▨	EID$_{5}$	EID$_{4}$	EID$_{3}$	EID$_{2}$	EID$_{1}$	EID$_{0}$	▨	▨	▨	▨	DLC$_{3}$	DLC$_{2}$	DLC$_{1}$	DLC$_{0}$
Data Byte 0								Data Byte 1							
Data Byte 2								Data Byte 3							
Data Byte 4								Data Byte 5							
Data Byte 6								Data Byte 7							
Time Stamp [15 to 8]								Time Stamp [7 to 0]							

CAN Data Frame:

StdID[10 to 6]	StdID[5 to 0]	ExtID[17 to 14]	ExtID[13 to 6]	ExtID[5 to 0]	DLC[3 to 0]	Data 0[7 to 0]	Data 1[7 to 0]		Data 7[7 to 0]

Figure 18-3. Bit organization of the message objects for word access

CAN Configuration Register

A programmable clock prescaler is used to derive the CAN module basic clock from the clock frequency f(CAN0/1)/2. Bit 0 to bit 3 of the CAN configuration register represent the prescaler, allowing a division ratio of 1 to 1/16 to be selected. So the CAN module basic clock frequency fCANB can be calculated as follows:

$$f_{CANB} = \frac{f(CAN0/1)}{2 \text{ x } (BRP + 1)}$$

where BRP is the value of the prescaler (selectable from 0 to 15). The effective baud rate of the CAN bus communication depends on the CAN bus timing control parameters and will be explained below.

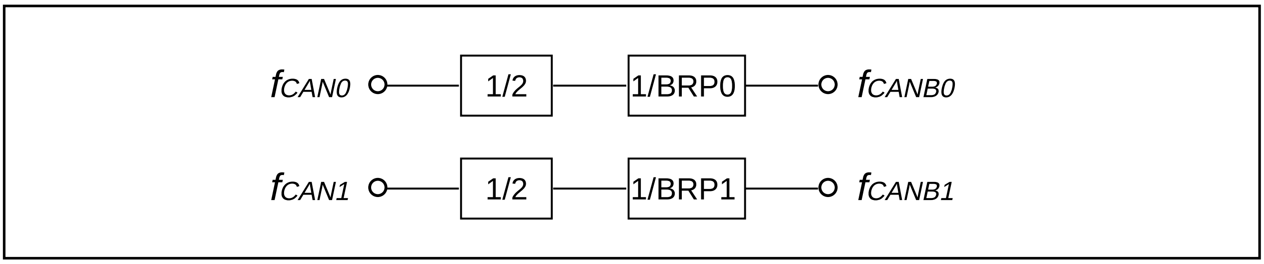
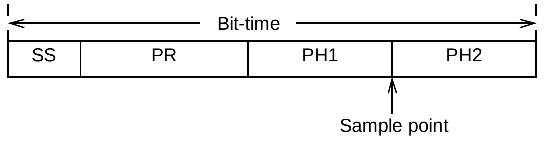


Figure 18-4. Generation of CAN basic clock frequency

CAN bus timing control

Each bit-time consists of four different segments:



Synchronization segment (SS),
Propagation time segment (PR),
Phase buffer segment 1 (PH1) and
Phase buffer segment 2 (PH2).

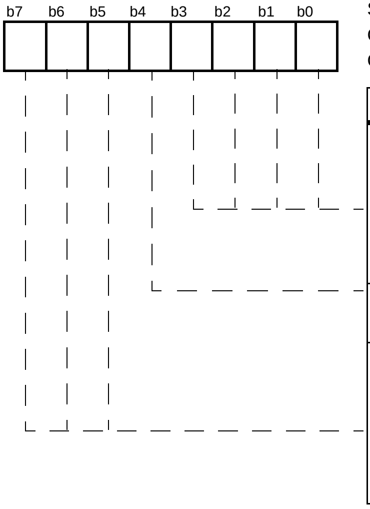
Figure 18-5. Bit timing

The first segment (SS) is fixed to one Time Quantum, the segments PR, PH1 and PH2 can be programmed from 1 to 8 Time Quanta by the CAN configuration register. The whole bit-time has to consist of minimum 8 and maximum 25 Time Quanta. The duration of one Time Quantum is the cycle time of fCANB.

$$Baudrate = \frac{f(CAN0/1)}{2 \text{ x } (BRP + 1) \text{ x } Num(quanta)}$$

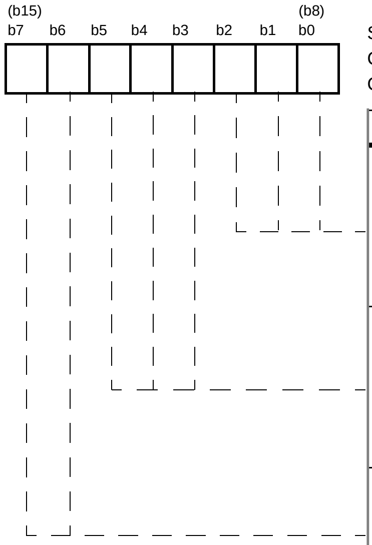
For example: assuming $f(X_{IN})$=16MHz and BRP=0, one Time Quantum will be 125ns long. This allows a maximum transmission rate of 1Mbps (assuming 8 Time Quanta per bit-time).

CAN Configuration Registers



Symbol	Address	When reset
C0CONR	021A$_{16}$	Indeterminate
C1CONR	023A$_{16}$	Indeterminate

Bit symbol	Bit name	Function	R	W
BRP	Prescaler Divider	Prescaler division ratio selection b3 b2 b1 b0 0 0 0 0 : 1 0 0 0 1 : 1/2 0 0 1 0 : 1/3 ... 1 1 1 1 : 1/16	O	O
SAM	Sampling Control Bit	0: One sample per bit 1: Three samples per bit	O	O
PR	Propagation Time	Duration Control Bits b7 b6 b5 0 0 0 : One Time Quantum 0 0 1 : Two Time Quanta ... 1 1 0 : Seven Time Quanta 1 1 1 : Eight Time Quanta	O	O



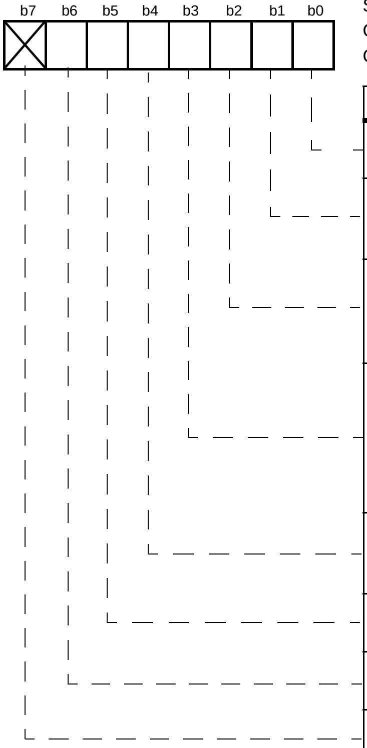
Symbol	Address	When reset
C0CONR	021B$_{16}$	Indeterminate
C1CONR	023B$_{16}$	Indeterminate

Bit symbol	Bit name	Function	R	W
PH1	Phase Buffer Segment 1	Duration Control Bits b2 b1b0 0 0 0 : One Time Quantum 0 0 1 : Two Time Quanta ... 1 1 0 : Seven Time Quanta 1 1 1 : Eight Time Quanta	O	O
PH2	Phase Buffer Segment 2	Duration Control Bits b5 b4 b3 0 0 0 : One Time Quantum 0 0 1 : Two Time Quanta ... 1 1 0 : Seven Time Quanta 1 1 1 : Eight Time Q uanta	O	O
SJW	Synchronization Jump Width	Control Bits b7 b6 0 0 : One Time Quantum 0 1 : Two Time Quanta 1 0 : Three Time Quanta 1 1 : Four Time Quanta	O	O

Figure 18-6. Description of CAN configuration register (Settings for CAN bus timing)

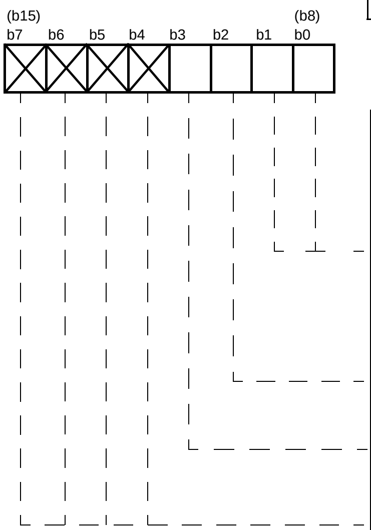
CAN Control Register

CAN Control Register



Symbol	Address	When reset
C0CTLR	0210_{16}	01_{16}
C1CTLR	0230_{16}	01_{16}

Bit symbol	Bit name	Function	R	W
Reset	CAN Module Reset	0: Operation Mode 1: Reset / Initialization Mode	O	O
LoopBack	Loop Back Mode for CAN Module	0: Normal Operation Mode 1: Loop Back Mode (read back and store the transmitted message)	O	O
MsgOrder	Message Order	0: The address order is adapted to word access (16Bit) for the message objects and also the mask memory. 1: The address order is byte linear(8Bit access).	O	O
BasicCAN	Basic CAN Feature	0: Normal Operation Mode: Slot #14/15 receive messages according to the 'first fit' system. 1: The CAN module switches between the two active receive-slots #14 and #15 in case of successive messages fitting into both slots.	O	O
BusErrEn	Bus Error Enable	0: Bus-errors will not be visible for ICU/CPU. 1: Bus-errors will be flagged for the ICU via the CAN Error Interrupt signal.	O	O
Sleep	Local Sleep Mode for CAN Module	0: Operation Mode 1: Clock of the CAN module will be stopped.	O	O
PortEn	CAN Port Enable	0: Port serves as I/O port. 1: Port serves as CAN terminal (CRX/CTX).	O	O
Nothing is assigned. It is not allowed to write '1' to this bit location. When read, its content is indeterminate.			-	-



Symbol	Address	When reset
C0CTLR	0211_{16}	00_{16}
C1CTLR	0231_{16}	00_{16}

Bit symbol	Bit name	Function	R	W
TSPreScale Bit1, Bit0	TimeStamp (TS) Prescaler	Prescaler value for TS counting source (basic source is the CAN bit-clock.) b1 b0 0 0: Bit-clock 0 1: (Bit-clock)/2 1 0: (Bit-clock)/4 1 1: (Bit-clock)/8	O	O
TSReset	TimeStamp (TS) Reset	0: TS counter counts bit cycles (s. the divider stages is activated by TSPreScale setting.) 1: Reset (Counter value is cleared to $0000.)	O	O (Note)
RetBusOff	Return from 'Error BusOff' State	0: Normal Operation Mode 1: Reset only for the CAN module to return from 'Error BusOff' state	O	O (Note)
Nothing is assigned. It is not allowed to write '1' to these bit locations. When read, their contents are indeterminate.			-	-

*) Note: If the TS Reset/RetBusOff (write '1') is activated, the control bit will be cleared automatically by the CAN module after performing the reset for the TS counter or the return from 'BusOff' state.

Figure 18-7. Structure of CAN control register

After leaving the MCU 'Reset'-state, the CAN module starts in 'Reset/Initialization' mode. All module setup parameters should be written in the relevant registers to enable the CAN module to take part in the CAN bus communication with the correct transfer rate , bit timing etc. (CAN Configuration Register). After finishing the initialization stage, the 'Reset' bit (CAN Control Register) has to be cleared by the user and the CAN module will start the bus participation according to the CAN specification.

In order to change the existing setting of the protocol configuration, activate 'Reset/Initialization' mode also during normal operation. In this case, the CAN module will leave the CAN bus communication in conformity with the protocol. This means, a just started transmit process has to be finished before entering the 'reset' state.

In case the protocol engine enters the 'Error BusOff' state, the system can be restarted in 'Error Active' mode by setting the 'RetBusOff' bit in the CAN Control Register. This 'reset' for the 'protocol controller' has no effect on the CAN-Interface configuration. The entire slot-configuration, slot contents and all SFR settings will be kept without changes.

Basic CAN Feature

Some applications for the CAN network operate with more than 16 message types (identifiers), so an original CAN approach (one slot corresponds to one message type) is not a feasible way for these systems. The first approach to give system support for these applications is the sophisticated mask concept implemented in this CAN module (refer to section 'Mask Register and Acceptance Filter').

In case there is the requirement to receive most or all messages from the CAN bus (performing further acceptance filter by software), the CAN module provides a special slot configuration to support this kind of system solution. In the normal operation mode, the received message is stored in the first fitting message slot. The slots under consideration for this decision will be determined in the acceptance filter phase. In this case many messages will be received by one slot, the CPU is heavily loaded to serve this slot without loosing a message because of 'overwriting' (receiving the next fitting message).

By activating the 'Basic CAN' feature, the slot scheduling in 'receive' case changes for slot #14 and #15. Received messages are stored alternately in these two message boxes.

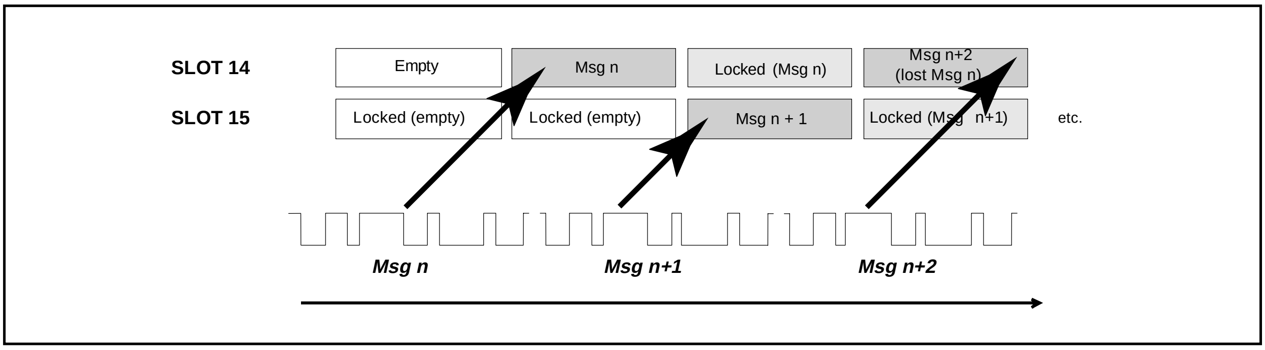


Figure 18-8. Receive slot scheduling for implementing the 'Basic CAN' feature

The CAN module uses two different slot addresses to build the Basic CAN feature (no shadow buffer concept). The 'lock/unlock' function will be controlled exclusively by the CAN module without any influence of the CPU. There is no 'message protection mechanism' implemented, so the message n+2 will overwrite the content of message n (Figure 18-7).

The following restrictions have to be kept in case of using the 'Basic CAN' feature:

- The module configuration ('Basic CAN' ON/OFF) should be selected before leaving the 'Reset/Initialization' state. The CAN module will store the first fitting message into slot #14 (in case the filtering failed for all preceding slots). In case the 'Basic CAN' feature will be enabled or disabled 'on the fly', the slot where the first message will be stored is undetermined.
- The CAN module never checks, whether or not the received message will be accepted by slot #15 when slot #14 is locked (last message is stored in slot #14). Therefore it is recommended to use the same identifier for the message slots #14 and #15 (building the 'Basic CAN' channel) and the same mask values for both local masks. Otherwise a received message might be dropped by the CAN module, although slot #14 could accept this message.

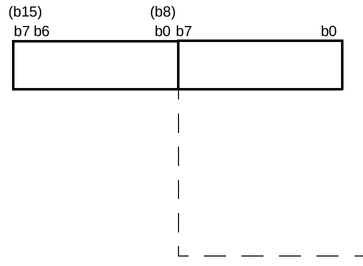
In the 'Basic CAN' mode, two exceptions regarding the message control register and the general configuration concept exist:

1. CAN Frame Type Tolerance
In the 'normal operation mode', decide if an activated slot should handle 'data' or 'remote' frames. It is not possible to receive 'data' frames and 'remote' frames in the same slot without a reconfiguration process by the CPU.
In case of the operation with a 'Basic CAN' channel, this behavior is not tolerable, because both frame types must be handled without CPU interaction.
Therefore, the 'Basic CAN' feature enables message slots #14 and #15 to receive both types of CAN frames, 'data'- and 'remote' frames.

2. CAN Frame Type Indication
In the 'normal operation mode', the Extended ID register (C0IDR/C1IDR) dictates the type of frames, i.e. Extended or Standard, which can be handled by the message box. As described in the upper section for the 'Basic CAN' slots, it is possible to receive both frame types irrespective of the slot configuration. Therefore, the CAN module provides the frame type information in the corresponding message control register.
Because the 'RemActive' information is not needed for the 'Basic CAN' slot function, the frame type information is mapped to this location (refer to section 'CAN Message Control Register'). The content of this bit corresponds to the frame type stored last in this slot location .

CAN Extended ID Register

CAN ExtID Register



Symbol	Address	When reset
C0IDR	0219_{16}, 0218_{16}	0000_{16}
C1IDR	0239_{16}, 0238_{16}	0000_{16}

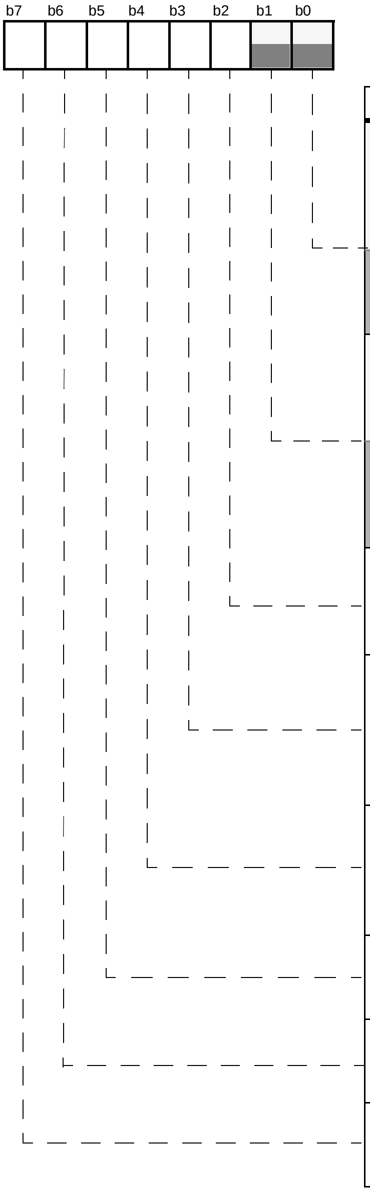
Function	Values that can be set for each bit	R	W
ExtID Bits Each bit corresponds with the appropriate message mailbox. ID-format assignment for each message mailbox	0: Mailbox handles standard identifiers. 1: Mailbox handles extended identifiers.	O	O

Note: These bits can not be set in Reset/Initialization Mode.

Figure 18-9. Structure of CAN extended ID register

CAN Message Control Register

CAN Message Control Register i



Symbol	Address	When reset
C0MCTLi	0200_{16} to $020F_{16}$	00_{16}
C1MCTLi	0220_{16} to $022F_{16}$	00_{16}
(i = 0 to 15)		

Bit symbol	Bit name	Function	R	W
NewData	New Data	- Receive Mailbox 0: The content of the message slot is read or still under processing by the CPU. 1: The CAN module has stored new data in the corresponding mailbox.	O	O (Note)
SentData	Sent Data	- Transmit Mailbox 0: Transmission is not started/finished yet. 1: Frame is transmitted successfully.		
InvalData	Invalid Data	- Receive Mailbox 0: The content of the message slot is valid. 1: Message slot contains invalid data (update of the message content is in progress).	O	O (Note)
TrmActive	Transmission Active	- Transmit Mailbox 0: Slot is waiting for bus free and passing internal transmit arbitration. 1: Transmit process for this message is active.		
MsgLost	Message Lost	- Receive Mailbox 0: No message was overwritten in this slot. 1: This mailbox already contained a message, but it was overwritten by a newer one.	O	O (Note)
RemActive	Remote Active	0: The module handles data frames; transmit/ receive case depends on the slot configuration (BasicCAN mode: data frame is stored). 1: Remote part of the auto-switch modes (Transmit Remote Frame / Receive Remote Frame) is active (BasicCAN mode: remote frame is stored).	O	O (Note)
RspLock	Response Locked	- Transmission Remote Mailbox 0: After a remote frame is received, it will be answered automatically. 1: After a remote frame is received, no transmission will be started as long as this bit is set to 1.	O	O
Remote	Remote Mailbox	0: 1: Mailbox is Remote Mailbox (s. also table 18-3)	O	O
RecReq	Receive Mailbox	0: 1: Mailbox is Receive Mailbox (s. also table 18-3)	O	O
TrmReq	Transmit Mailbox	0: 1: Mailbox is Transmission Mailbox (s. also table 18-3)	O	O

Note: The write access on CPU side is limited to 'write only 0'. If the CPU tries to set these bits to 1, it will not have any influence to the content of these bits. In this case, the values of these bits are forced to their 'status meaning' defined by the current state of the CAN module.

Figure 18-10. Structure of CAN message control register

Reception- and Transmission Modes

TrmReq	RecReq	Remote	RspLock	Description
0	0	-	-	Configuration Mode CPU could configure new transfer mode for this mailbox.
0	1	0	-	Mailbox is configured as a **Receivebox for Dataframes**.
1	0	1	0	1. Step: Mailbox **transmits Remoteframe** (RemAct-Bit is 1) 2. Step: Mailbox becomes a **Receivebox for Dataframes** - RemActive-Bit is set to 0. **Exception:** When the matching dataframe is already detected on the busline before the remoteframe can be sent, the mailbox becomes immediately a **Receivebox for Dataframes**.
1	0	0	0	Mailbox is configured as a **Transmissionbox for Dataframes**.
0	1	1	1/0	1. Step: Mailbox **receives a Remoteframe** (RemAct-Bit is 1). 2. Step: Mailbox becomes a **Transmissionbox for Dataframes** - RemActive-Bit will be set to 0. **Remark:** As long as RspLock=1, no transmission can be started. This means that Remoteframes are not answered automatically.

Table 18-3. Table of all reception- and transmission modes

Notes - Reception Mode

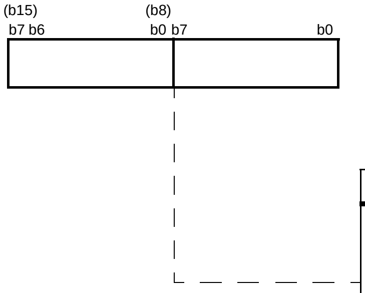
- A received message, which fulfills the comparison conditions of several mailboxes, will be stored in the first suitable mailbox starting with the Message Mailbox Slot0 (special case for the 'Basic CAN' feature). This means the message will be stored only one time.
- When the CAN module transmits a message, the CAN module receives its own message. However, the CAN module does not store that message in the normal operation mode, even if there is a receive box with a fitting identifier. In case the CAN module operates in the 'loop back' mode (CAN Control Register), the transmitted message is stored in a prepared mailbox (receivebox with corresponding identifier).

Notes - Transmission Mode

- Overwrite Procedure of an activated Transmission Mailbox
- In order to activate a transmission mailbox, set the configuration bits according to table 18-3.
- In order to overwrite the content of a transmission mailbox, deactivate the transmission mailbox. This means, the CPU must clear the TrmReq-Bit (together with the RecReq-Bit!).
- The CPU has to read the TrmActive-Bit to check its current status. When the TrmActive-Bit is '0', the abort request is successful and the CPU can overwrite the data of the transmission mailbox.
- After this check, the CPU has the possibility to determine whether the message is transmitted or not. The abort request by CPU side is executed (successful), in case the SentData-Bit is not set. Otherwise the message is transmitted successfully in spite of the abort request.

CAN Slot Interrupt Control Register

CAN Slot Interrupt Control Register



Symbol	Address	When reset
C0SICR	0217_{16}, 0216_{16}	0000_{16}
C1SICR	0237_{16}, 0236_{16}	0000_{16}

Function	Values that can be set for each bit	R	W
Interrupt Enable Bits Each bit corresponds with the appropriate message mailbox. The transfer interrupts ('Successful Reception' / 'Successful Transmission') for each message mailbox can be enabled and disabled.	0: After a successful transmission or reception operation no interrupt request bit is set. 1: After a successful transmission or reception operation the interrupt request bit in the corresponding MCU interrupt control register is set.	O	O

Note: These bits can not be set in Reset/Initialization Mode.

Figure 18-11. Structure of CAN slot interrupt control register

Mask Register and Acceptance Filter

For the acceptance filter, three 29-Bit mask registers are provided. One global mask is assigned to the mailboxes 0 to 13 and two local masks belong to mailbox 14 and 15 respectively. If the mailbox is configured as a receive slot, the Standard ID and the Extended ID of the message object act as the local ID mask.

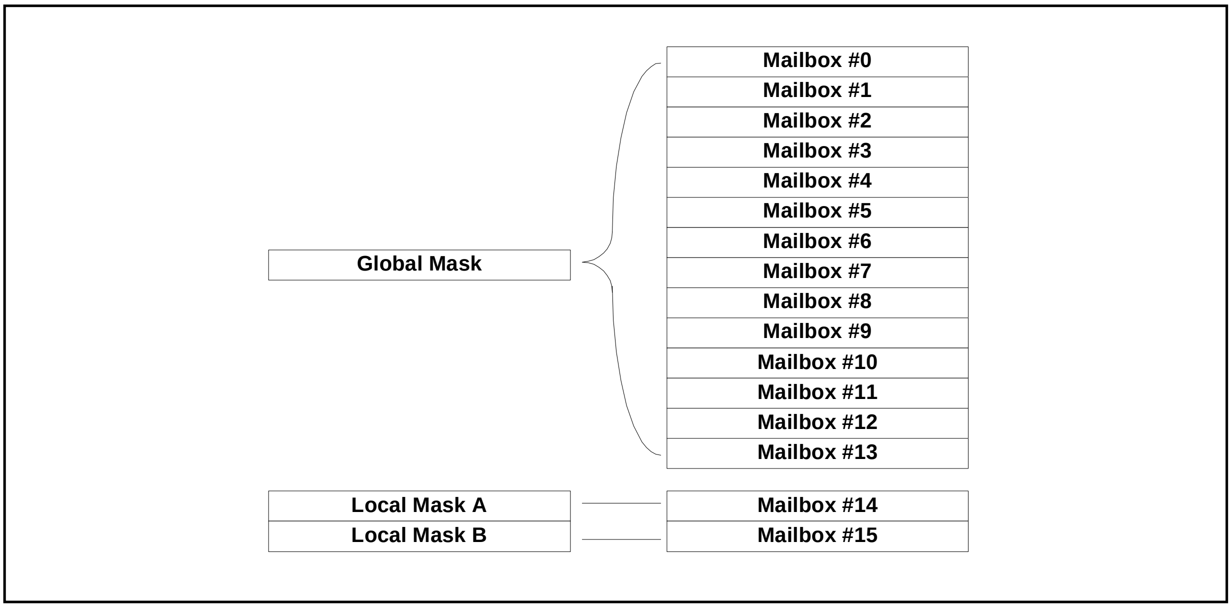
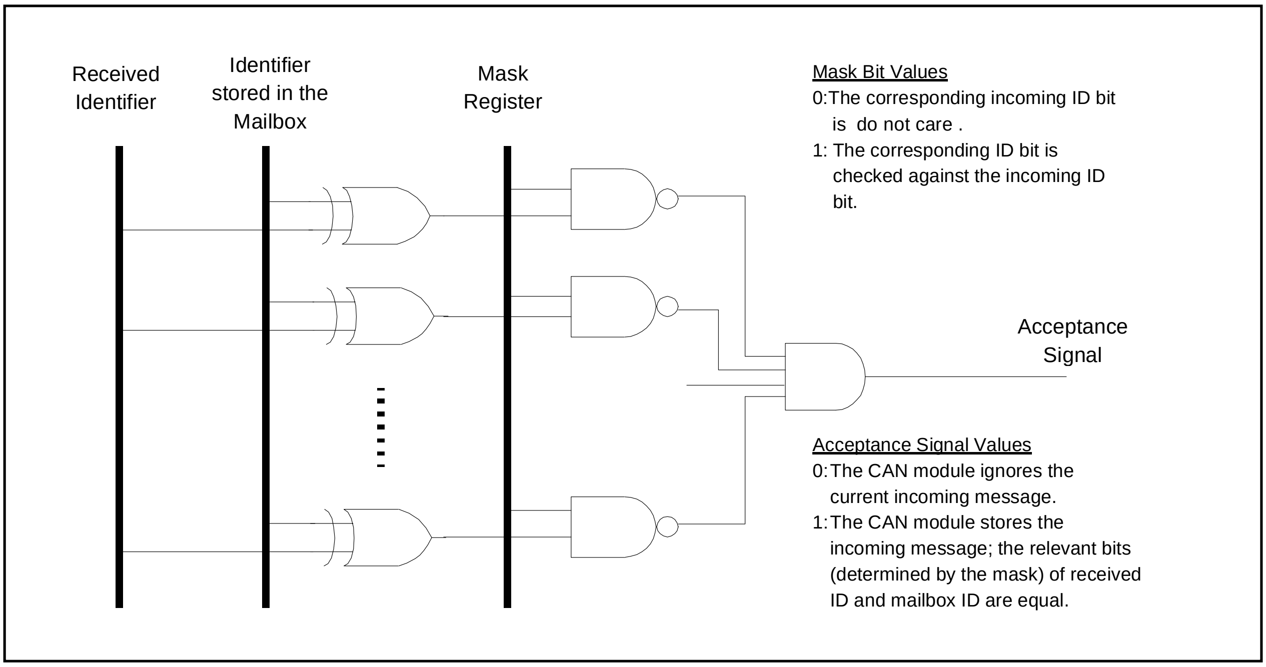


Figure 18-12. Mask assignment

The mask registers provide the possibility to filter a range of identifier. They can mask the identifier by setting each bit to '0'. The acceptance filter can be performed either for 29 or for 11 bit identifier length, determined by the Extended ID register setting for the corresponding mailbox. The mailbox itself contains the identifier for the filtering process. Together with the relevant mask, the filtering is performed as shown in the figure below.



Mask Bit Values
0:The corresponding incoming ID bit is do not care .
1: The corresponding ID bit is checked against the incoming ID bit.

Acceptance Signal Values
0:The CAN module ignores the current incoming message.
1:The CAN module stores the incoming message; the relevant bits (determined by the mask) of received ID and mailbox ID are equal.

Figure 18-13. Structure of acceptance filter

Figure 18-13 and 18-14 show the memory location of these three filter masks and their bitmap. The structure of the bit organization is adapted to the identifier format in every message slot (refer to Figure 18-2/18-3).
After MCU reset condition, the content of the mask registers is undefined.

Bit 7							Bit 0	Addresses CAN0	CAN1	
			SID_{10}	SID_9	SID_8	SID_7	SID_6	**0160_{16}**	**0360_{16}**	Global Mask
		SID_5	SID_4	SID_3	SID_2	SID_1	SID_0	**0161_{16}**	**0361_{16}**	
				EID_{17}	EID_{16}	EID_{15}	EID_{14}	**0162_{16}**	**0362_{16}**	
EID_{13}	EID_{12}	EID_{11}	EID_{10}	EID_9	EID_8	EID_7	EID_6	**0163_{16}**	**0363_{16}**	
		EID_5	EID_4	EID_3	EID_2	EID_1	EID_0	**0164_{16}**	**0364_{16}**	
			SID_{10}	SID_9	SID_8	SID_7	SID_6	**0166_{16}**	**0366_{16}**	Local Mask A
		SID_5	SID_4	SID_3	SID_2	SID_1	SID_0	**0167_{16}**	**0367_{16}**	
				EID_{17}	EID_{16}	EID_{15}	EID_{14}	**0168_{16}**	**0368_{16}**	
EID_{13}	EID_{12}	EID_{11}	EID_{10}	EID_9	EID_8	EID_7	EID_6	**0169_{16}**	**0369_{16}**	
		EID_5	EID_4	EID_3	EID_2	EID_1	EID_0	**$016A_{16}$**	**$036A_{16}$**	
			SID_{10}	SID_9	SID_8	SID_7	SID_6	**$016C_{16}$**	**$036C_{16}$**	Local Mask B
		SID_5	SID_4	SID_3	SID_2	SID_1	SID_0	**$016D_{16}$**	**$036D_{16}$**	
				EID_{17}	EID_{16}	EID_{15}	EID_{14}	**$016E_{16}$**	**$036E_{16}$**	
EID_{13}	EID_{12}	EID_{11}	EID_{10}	EID_9	EID_8	EID_7	EID_6	**$016F_{16}$**	**$036F_{16}$**	
		EID_5	EID_4	EID_3	EID_2	EID_1	EID_0	**0170_{16}**	**0370_{16}**	

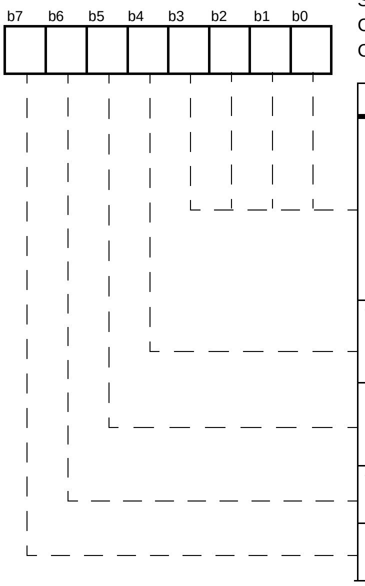
Figure 18-14. Acceptance filter masks (byte address order)

	n+1								n								Addresses
	Bit 15							Bit 8	Bit 7							Bit 0	CAN0 / CAN1
Global Mask				SID_{10}	SID_9	SID_8	SID_7	SID_6			SID_5	SID_4	SID_3	SID_2	SID_1	SID_0	**$0160_{16}/0360_{16}$**
					EID_{17}	EID_{16}	EID_{15}	EID_{14}	EID_{13}	EID_{12}	EID_{11}	EID_{10}	EID_9	EID_8	EID_7	EID_6	**$0162_{16}/0362_{16}$**
			EID_5	EID_4	EID_3	EID_2	EID_1	EID_0									**$0164_{16}/0364_{16}$**
Local Mask A				SID_{10}	SID_9	SID_8	SID_7	SID_6			SID_5	SID_4	SID_3	SID_2	SID_1	SID_0	**$0166_{16}/0366_{16}$**
					EID_{17}	EID_{16}	EID_{15}	EID_{14}	EID_{13}	EID_{12}	EID_{11}	EID_{10}	EID_9	EID_8	EID_7	EID_6	**$0168_{16}/0368_{16}$**
			EID_5	EID_4	EID_3	EID_2	EID_1	EID_0									**$016A_{16}/036A_{16}$**
Local Mask B				SID_{10}	SID_9	SID_8	SID_7	SID_6			SID_5	SID_4	SID_3	SID_2	SID_1	SID_0	**$016C_{16}/036C_{16}$**
					EID_{17}	EID_{16}	EID_{15}	EID_{14}	EID_{13}	EID_{12}	EID_{11}	EID_{10}	EID_9	EID_8	EID_7	EID_6	**$016E_{16}/036E_{16}$**
			EID_5	EID_4	EID_3	EID_2	EID_1	EID_0									**$0170_{16}/0370_{16}$**

Figure 18-15. Acceptance filter masks (word address order)

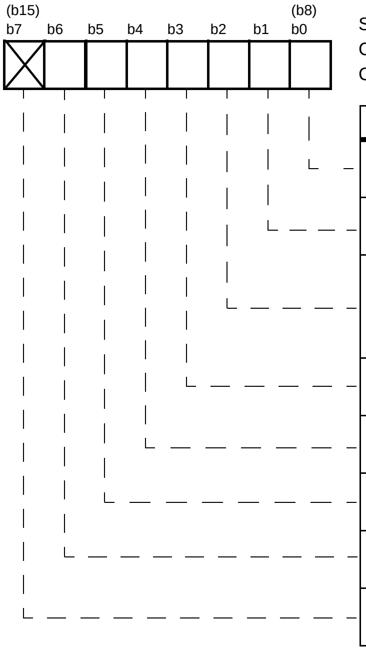
CAN Status Register

CAN Status Register



Symbol	Address	When reset
C0STR	0212$_{16}$	00$_{16}$
C1STR	0232$_{16}$	00$_{16}$

Bit symbol	Bit name	Function	R	W
MBox Bit3..Bit0	Mailbox Number	Number of the mailbox which transmitted/ received a message successfully b3 b2 b1 b0 0 0 0 0 : Mailbox 0 0 0 0 1 : Mailbox 1 ... 1 1 1 1 : Mailbox 15	O	-
TrmSucc	Transmission Successful	0: No [successful] transmission 1: CAN module transmitted a message successfully.	O	-
RecSucc	Receive Successful	0: No [successful] reception 1: CAN module received a message successfully.	O	-
TrmState	Transmitter	0: CAN module is idle or receiver. 1: CAN module is transmitter.	O	-
RecState	Receiver	0: CAN module is idle or transmitter. 1: CAN module is receiver.	O	-



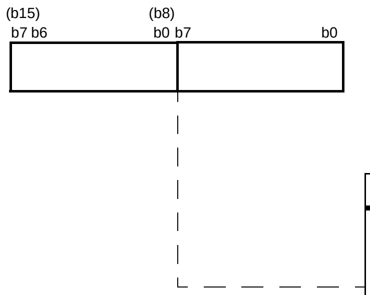
Symbol	Address	When reset
C0STR	0213$_{16}$	01$_{16}$
C1STR	0233$_{16}$	01$_{16}$

Bit symbol	Bit name	Function	R	W
Reset	Reset Acknowledge Bit	0: Operation Mode 1: Reset	O	-
LoopBack	Loop Back Acknowledge	0: Normal Operation Mode 1: Loop Back Mode	O	-
MsgOrder	Message Order	0: The address order for the message objects is adapted to word access. 1: The address order is byte linear (8 Bit access only).	O	-
BasicCAN	BasicCAN Feature	0: Normal Operation Mode 1: BasicCAN feature (slot #14/15)	O	-
BusError	Bus Error	0: No error occurred. 1: CAN Bus-Error occurred.	O	-
ErrPass	CAN module is in Error Passive state	0: CAN module is not Error Passive and Bus Off. 1: CAN module is Error Passive or Bus Off.	O	-
BusOff	CAN module is in Error BusOff state	0: CAN module is not Bus Off. 1: CAN module is Bus Off.	O	-
Nothing is assigned. It is not allowed to write ’1’ to this bit location. When read, their contents are indeterminate.			-	-

Figure 18-16. Structure of CAN status register

CAN Slot Status Register

CAN Slot Status Register



Symbol	Address	When reset
C0SSTR	0215_{16}, 0214_{16}	0000_{16}
C1SSTR	0235_{16}, 0234_{16}	0000_{16}

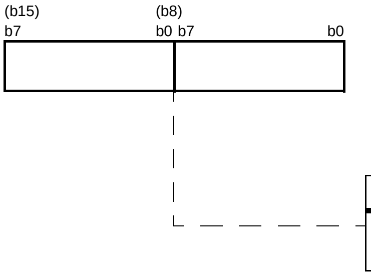
Function	Values that can be set for each bit	R	W
Slot Status Bits Each bit corresponds with the appropriate message mailbox.	0: - Receive Mailbox Slot contains no unread message. - Transmit Mailbox Message is not sent yet 1: - Receive Mailbox Slot contains unread message. - Transmit Mailbox Message was sent successfully.	O	-

Figure 18-17. Structure of CAN slot status register

CAN Time Stamp Register

- The CPU can read out the content of the 16 bit timer responsible for the time stamp function of the CAN module.

CAN Time Stamp Register



Symbol	Address	When reset
C0TSR	$021F_{16}$, $021E_{16}$	0000_{16}
C1TSR	$023F_{16}$, $023E_{16}$	0000_{16}

Function	Values that can be read	R	W
Time Stamp (16 Bit) Counts the CAN bus bit cycles	0000_{16} to $FFFF_{16}$	O	-

Figure 18-18. CAN time stamp register

The basic clock for this timer is the bit clock derived from the CAN bus bit timing. The content of the timer is increased by one when received or transmitted frame bits. When the CAN bus is idle, the timer is increased by the nominal bit rate, which is defined in the CAN Configuration Register.

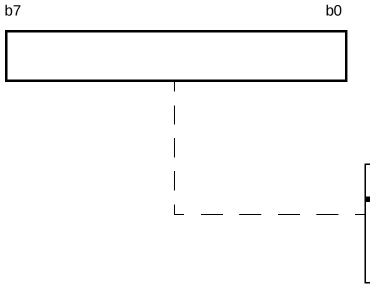
By help of an additional prescaler structure, the basic clock can be divided by the scale factor 1/1, 1/2, 1/4 or 1/8 (refer to the description of the CAN Control Register).

For the 'time stamp' function the content of the counter is captured after the current message on the bus is declared to be valid. This decision is made in conformity to the definition of a 'successful receive process' based on the CAN specification. This 'time stamp' is stored in the message buffer which corresponds to the successful receive process.

CAN REC- and CAN TEC-Register

- The REC- and TEC-Register can be used for the analysis of the CAN bus transmit and receive error occurrences.

CAN REC Register

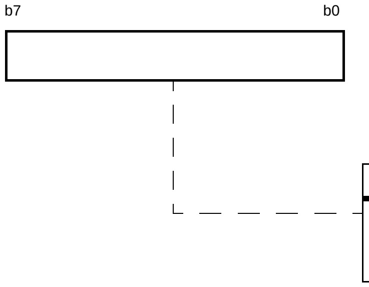


Symbol	Address	When reset
C0RECR	$021C_{16}$	00_{16}
C1RECR	$023C_{16}$	00_{16}

Function	Values that can be read	R	W
Receive Error Counter Increment and decrement according to the CAN specification	00_{16} to FF_{16}	O	-

Figure 18-19. Structure of CAN REC register (Receive error counter)

CAN TEC Register



Symbol	Address	When reset
C0TECR	$021D_{16}$	00_{16}
C1TECR	$023D_{16}$	00_{16}

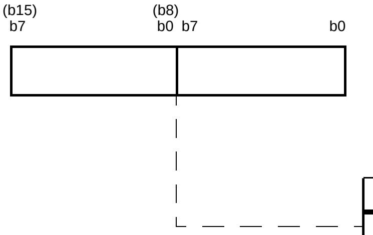
Function	Values that can be read	R	W
Transmit Error Counter Increment and decrement according to the CAN specification	00_{16} to FF_{16}	O	-

Figure 18-20. Structure of CAN TEC register (Transmit error counter)

Acceptance Filter Support Register

- The Acceptance filter Support Register can be used for the implementation of efficient acceptance filter rooutines.

Acceptance Filter Support Register



Symbol	Address	When reset
C0AFS	0243_{16}, 0242_{16}	indeterminate
C1AFS	0245_{16}, 0244_{16}	indeterminate

Function	Values that can be set	R	W
At write each bit corresponds to an standard identifier bit of a received message object. At read a modified form of the standard identifier is obtained.	Standard identifier in word- or byte order can be set.	O	O

Figure 18-21. Acceptance Filter Support Register

When writing the first two bytes of a received message object to the Acceptance Filter Support Register, the bits are modified as illustrated in Figure 18-20. Therefore, when read, the obtained value can be used for an efficient software acceptance filtering of the most recently written standard identifier. The message order that the Acceptance Filter Support Register expects is the message order of the according CAN module.

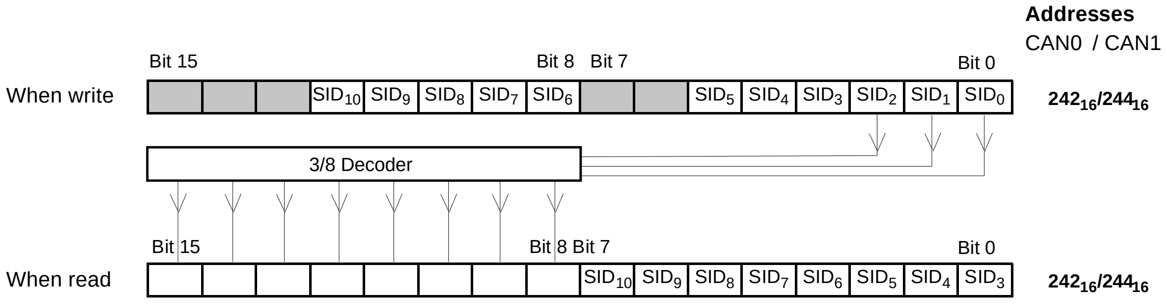


Figure 18-22. Write/read of Acceptance Filter Support Register (word order)

Programmable I/O Ports

There are 87 programmable I/O ports: P0 to P10 (excluding P8$_5$). Each port can be set independently for input or output using the direction register. A pull-up resistance for each block of 4 ports can be set. P7$_1$ and P9$_1$ are Nch open drain ports and have no built-in pull-up resistance. P8$_5$ is an input-only port and has no built-in pull-up resistance.
Figures 19-1 and 19-3 show the programmable I/O ports.
Each pin functions as a programmable I/O port and as the I/O for the built-in peripheral devices.
To use the pins as the inputs for the built-in peripheral devices, set the direction register of each pin to input mode. When the pins are used as the outputs for the built-in peripheral devices (other than the D-A converter), they function as outputs regardless of the contents of the direction registers. See the descriptions of the respective functions for how to set up the built-in peripheral devices.

(1) Direction registers

Figure 19-4 shows the direction registers.
These registers are used to choose the direction of the programmable I/O ports. Each bit in these registers corresponds one for one to each I/O pin.
Note: There is no direction register bit for P8$_5$.

(2) Port registers

Figure 19-5 shows the port registers.
These registers are used to write and read data for input and output to and from an external device. A port register consists of a port latch to hold output data and a circuit to read the status of a pin. Each bit in port registers corresponds one for one to each I/O pin.

(3) Pull-up control registers

Figure 19-6 shows the pull-up control registers.
The pull-up control register can be set to apply a pull-up resistance to each block of 4 ports. When ports are set to have a pull-up resistance, the pull-up resistance is connected only when the direction register is set for input.
However, in memory expansion mode and microprocessor mode, P0 to P5 operate as the bus and the pull-up control register setting is invalid.

(4) Port control register

Figure 19-7 shows the port control register.
The bit 0 of port control register is used to read port P1 as follows:

0 : When port P1 is input port, port input level is read.
When port P1 is output port , the contents of port P1 register is read.
1 : The contents of port P1 register is read though port P1 is input/output port.

This register is valid in the following:

- External bus width is 8 bits in microprocessor mode or memory expansion mode.
- Port P1 can be used as a port in multiplexed bus for the entire space.

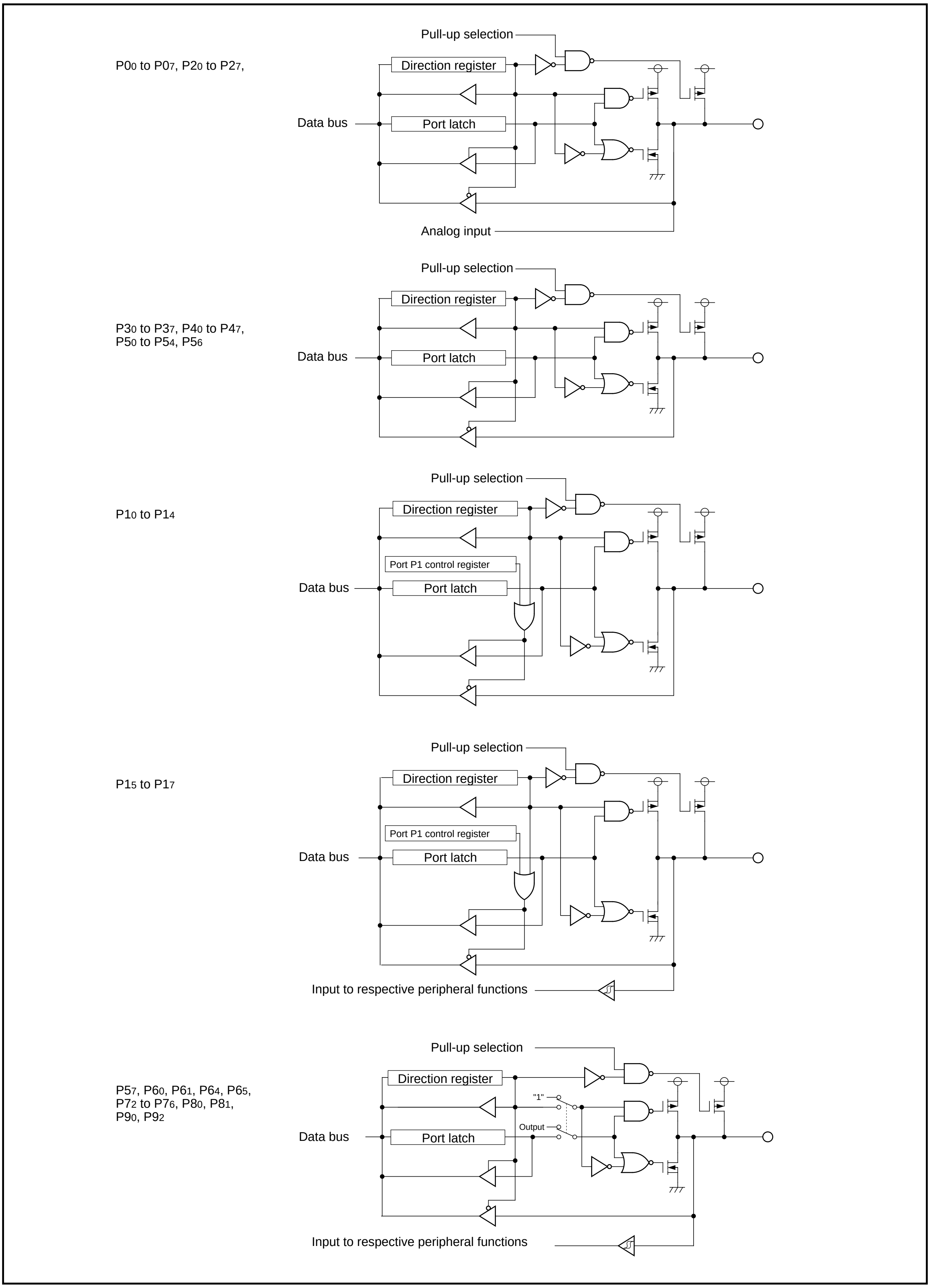
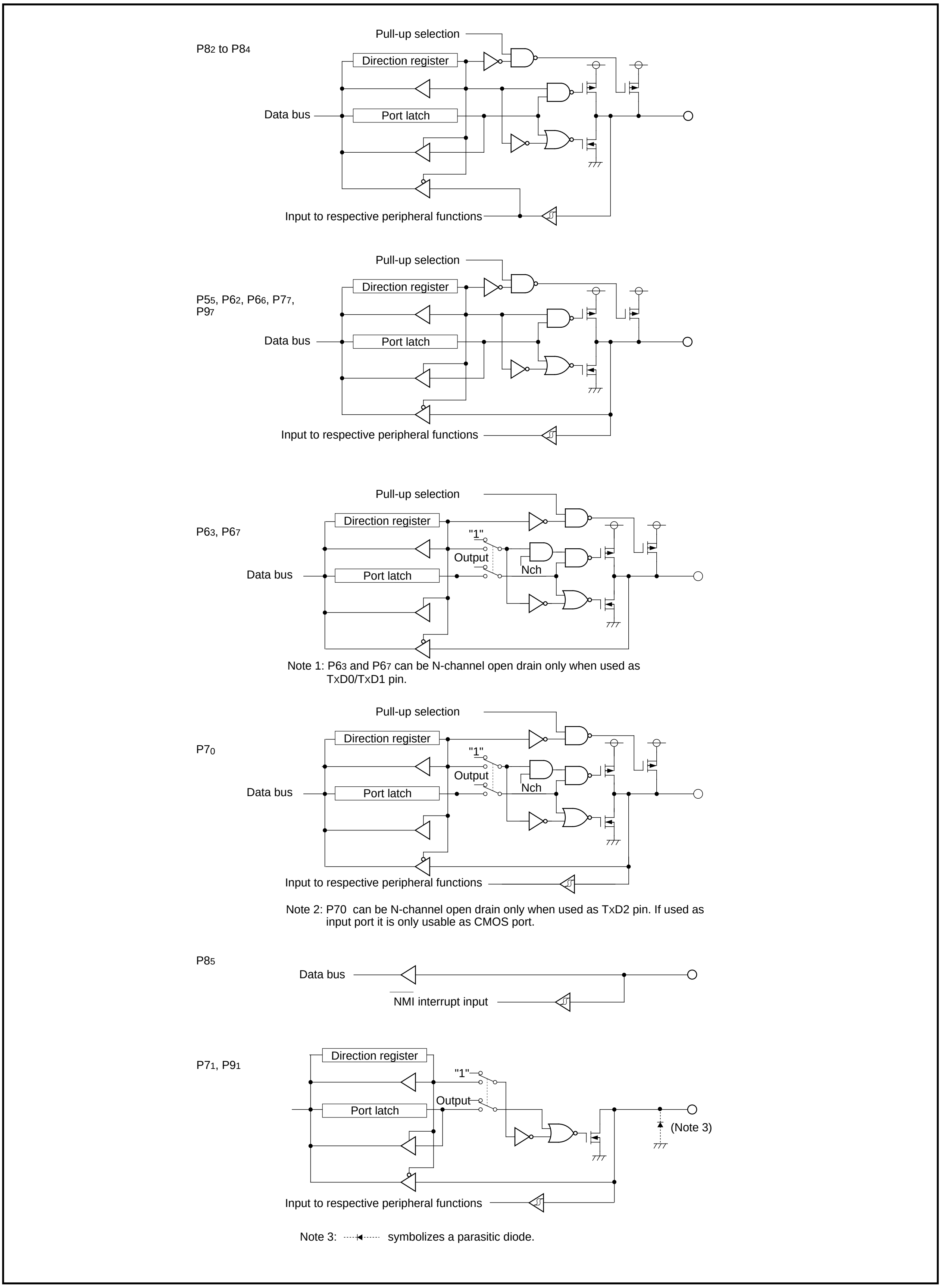


Figure 19-1. Programmable I/O ports (1)



Note 1: $P6_3$ and $P6_7$ can be N-channel open drain only when used as TxD0/TxD1 pin.

Note 2: P70 can be N-channel open drain only when used as TxD2 pin. If used as input port it is only usable as CMOS port.

Note 3: ····|◂···· symbolizes a parasitic diode.

Figure 19-2. Programmable I/O ports (2)

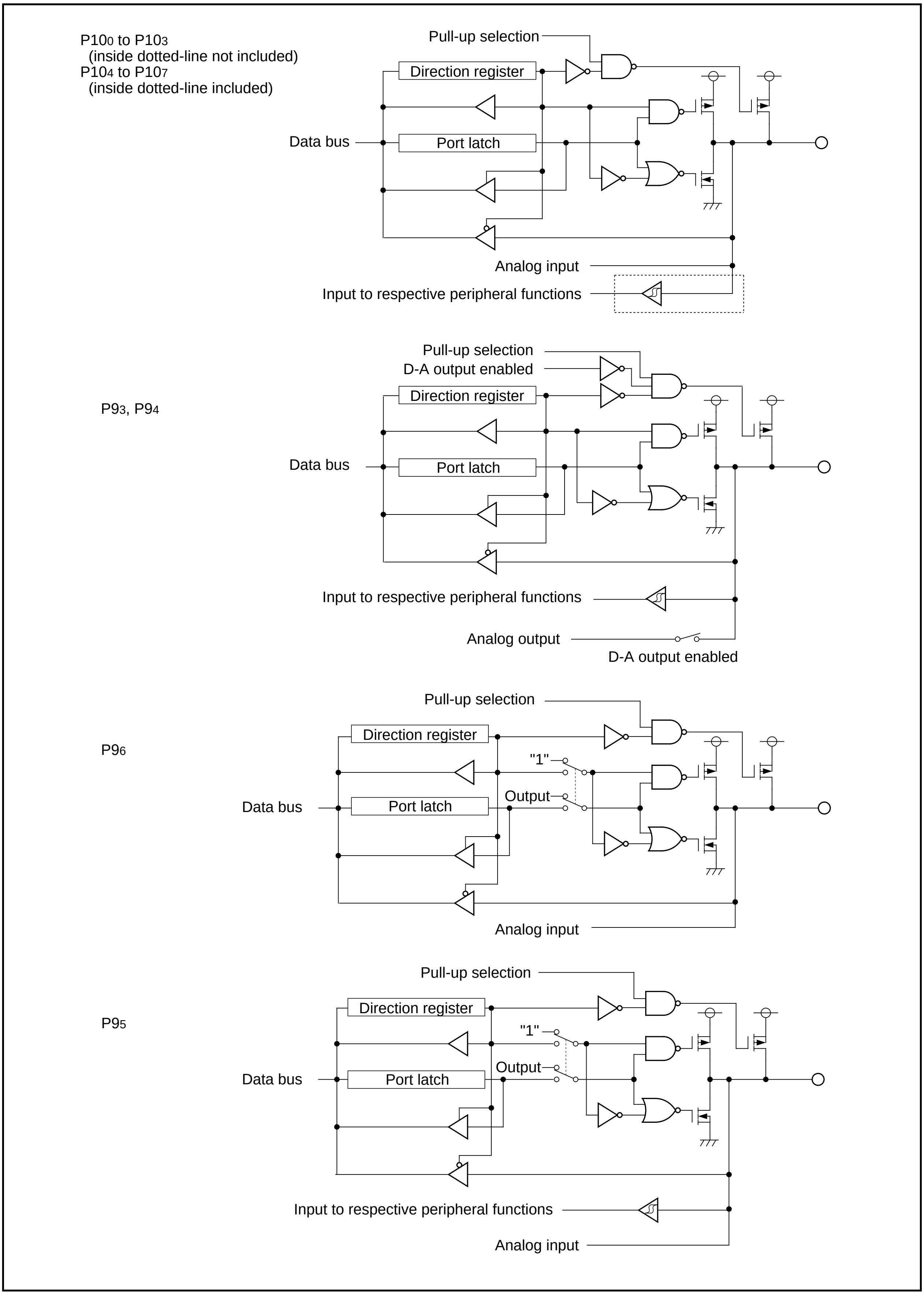


Figure 19-3. Programmable I/O ports (3)

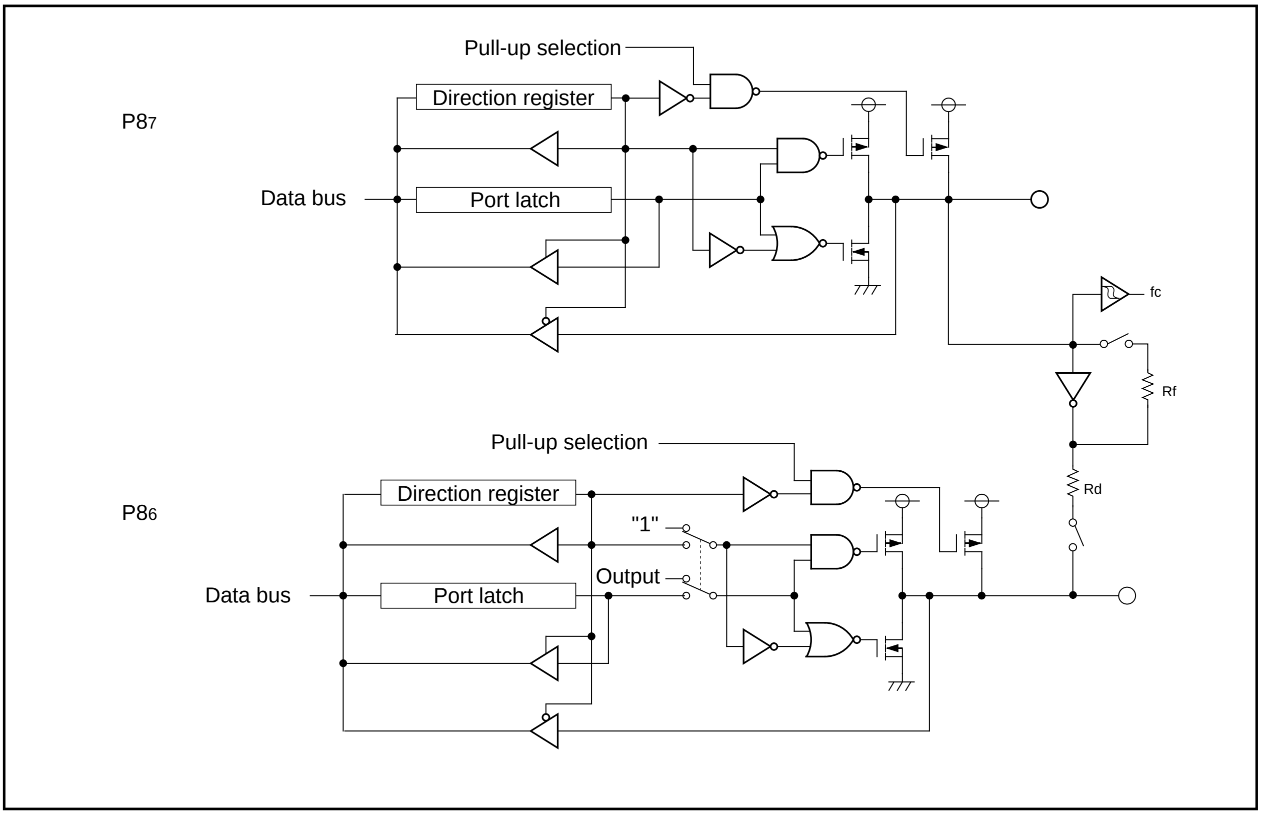


Figure 19-4. Programmable I/O ports (4)

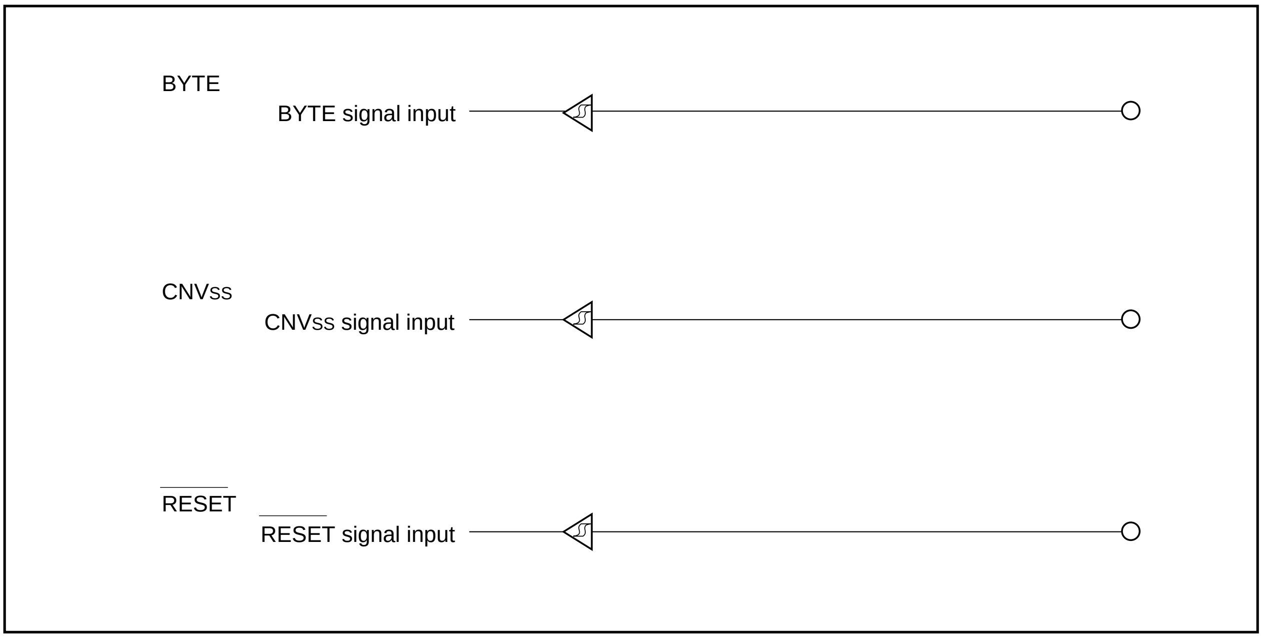

BYTE
BYTE signal input
CNVSS
CNVSS signal input
$\overline{\text{RESET}}$
$\overline{\text{RESET}}$ signal input

Figure 19-5. I/O pins

Port Pi direction register (Note)

b7 b6 b5 b4 b3 b2 b1 b0

Symbol	Address	When reset
PDi (i = 0 to 10, except 8)	$03E2_{16}$, $03E3_{16}$, $03E6_{16}$, $03E7_{16}$, $03EA_{16}$ $03EB_{16}$, $03EE_{16}$, $03EF_{16}$, $03F3_{16}$, $03F6_{16}$	00_{16}

Bit symbol	Bit name	Function	R	W
PDi_0	Port Pi_0 direction register	0 : Input mode (Functions as an input port) 1 : Output mode (Functions as an output port) (i = 0 to 10 except 8)	○	○
PDi_1	Port Pi_1 direction register		○	○
PDi_2	Port Pi_2 direction register		○	○
PDi_3	Port Pi_3 direction register		○	○
PDi_4	Port Pi_4 direction register		○	○
PDi_5	Port Pi_5 direction register		○	○
PDi_6	Port Pi_6 direction register		○	○
PDi_7	Port Pi_7 direction register		○	○

Note: Set bit 2 of protect register (address $000A_{16}$) to “1” before rewriting to the port P7 and P9 direction register.

Port P8 direction register

b7 b6 b5 b4 b3 b2 b1 b0

Symbol	Address	When reset
PD8	$03F2_{16}$	$00X00000_2$

Bit symbol	Bit name	Function	R	W
PD8_0	Port $P8_0$ direction register	0 : Input mode (Functions as an input port) 1 : Output mode (Functions as an output port)	○	○
PD8_1	Port $P8_1$ direction register		○	○
PD8_2	Port $P8_2$ direction register		○	○
PD8_3	Port $P8_3$ direction register		○	○
PD8_4	Port $P8_4$ direction register		○	○
Nothing is assigned. This bit can either be set nor reset. When read, its content is indeterminate.			—	—
PD8_6	Port $P8_6$ direction register	0 : Input mode (Functions as an input port) 1 : Output mode (Functions as an output port)	○	○
PD8_7	Port $P8_7$ direction register		○	○

Figure 19-6. Direction register

Port Pi register

b7 b6 b5 b4 b3 b2 b1 b0

Symbol	Address	When reset
Pi (i = 0 to 10, except 8)	$03E0_{16}$, $03E1_{16}$, $03E4_{16}$, $03E5_{16}$, $03E8_{16}$	Indeterminate
	$03E9_{16}$, $03EC_{16}$, $03ED_{16}$, $03F1_{16}$, $03F4_{16}$	Indeterminate

Bit symbol	Bit name	Function	R	W
Pi_0	Port Pi$_0$ register	Data is input and output to and from each pin by reading and writing to and from each corresponding bit 0 : L level data 1 : H level data (i = 0 to 10 except 8)	○	○
Pi_1	Port Pi$_1$ register		○	○
Pi_2	Port Pi$_2$ register		○	○
Pi_3	Port Pi$_3$ register		○	○
Pi_4	Port Pi$_4$ register		○	○
Pi_5	Port Pi$_5$ register		○	○
Pi_6	Port Pi$_6$ register		○	○
Pi_7	Port Pi$_7$ register		○	○

Port P8 register

b7 b6 b5 b4 b3 b2 b1 b0

Symbol	Address	When reset
P8	$03F0_{16}$	Indeterminate

Bit symbol	Bit name	Function	R	W
P8_0	Port P8$_0$ register	Data is input and output to and from each pin by reading and writing to and from each corresponding bit (except for P8$_5$) 0 : L level data 1 : H level data	○	○
P8_1	Port P8$_1$ register		○	○
P8_2	Port P8$_2$ register		○	○
P8_3	Port P8$_3$ register		○	○
P8_4	Port P8$_4$ register		○	○
P8_5	Port P8$_5$ register		○	×
P8_6	Port P8$_6$ register		○	○
P8_7	Port P8$_7$ register		○	○

Figure 19-7. Port register

Pull-up control register 0

b7 b6 b5 b4 b3 b2 b1 b0

Symbol	Address	When reset
PUR0	$03FC_{16}$	00_{16}

Bit symbol	Bit name	Function	R	W
PU00	$P0_0$ to $P0_3$ pull-up	The corresponding port is pulled high with a pull-up resistor 0 : Not pulled high 1 : Pulled high	○	○
PU01	$P0_4$ to $P0_7$ pull-up		○	○
PU02	$P1_0$ to $P1_3$ pull-up		○	○
PU03	$P1_4$ to $P1_7$ pull-up		○	○
PU04	$P2_0$ to $P2_3$ pull-up		○	○
PU05	$P2_4$ to $P2_7$ pull-up		○	○
PU06	$P3_0$ to $P3_3$ pull-up		○	○
PU07	$P3_4$ to $P3_7$ pull-up		○	○

Pull-up control register 1

b7 b6 b5 b4 b3 b2 b1 b0

Symbol	Address	When reset
PUR1	$03FD_{16}$	00_{16} (Note 2)

Bit symbol	Bit name	Function	R	W
PU10	$P4_0$ to $P4_3$ pull-up	The corresponding port is pulled high with a pull-up resistor 0 : Not pulled high 1 : Pulled high	○	○
PU11	$P4_4$ to $P4_7$ pull-up		○	○
PU12	$P5_0$ to $P5_3$ pull-up		○	○
PU13	$P5_4$ to $P5_7$ pull-up		○	○
PU14	$P6_0$ to $P6_3$ pull-up		○	○
PU15	$P6_4$ to $P6_7$ pull-up		○	○
PU16	$P7_0$ to $P7_3$ pull-up (Note 1)		○	○
PU17	$P7_4$ to $P7_7$ pull-up		○	○

Note 1: Since $P7_0$ is N-channel open drain port, pull-up is not available for it.
Note 2: When the V_{CC} level is being impressed to the CNV_{SS} terminal, this register becomes to 02_{16} when reset (PU11 becomes to 1).

Pull-up control register 2

b7 b6 b5 b4 b3 b2 b1 b0

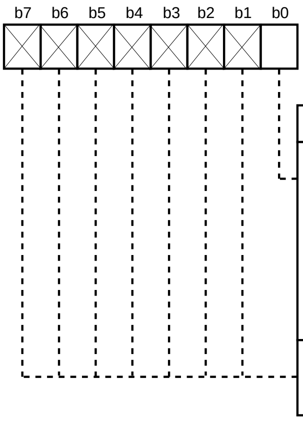
Symbol	Address	When reset
PUR2	$03FE_{16}$	00_{16}

Bit symbol	Bit name	Function	R	W
PU20	$P8_0$ to $P8_3$ pull-up	The corresponding port is pulled high with a pull-up resistor 0 : Not pulled high 1 : Pulled high	○	○
PU21	$P8_4$ to $P8_7$ pull-up (Except $P8_5$)		○	○
PU22	$P9_0$ to $P9_3$ pull-up (Note 3)		○	○
PU23	$P9_4$ to $P9_7$ pull-up		○	○
PU24	$P10_0$ to $P10_3$ pull-up		○	○
PU25	$P10_4$ to $P10_7$ pull-up		○	○
Nothing is assigned. Theses bits can neither be set nor reset. When read, their contents are 0 .			—	—

Note 3: Since $P9_1$ is N-channel open drain port, pull-up is not available for it.

Figure 19-8. Pull-up control register

Port control register



Symbpl	Address	When reset
PCR	03FF$_{16}$	00$_{16}$

Bit symbol	Bit name	Function	R	W
PCR0	Port P1 control register	0 : When input port, read port input level. When output port, read the contents of port P1 register. 1 : Read the contents of port P1 register though input/output port.	○	○
Nothing is assigned. Theses bits can neither be set nor reset. When read, their contents are 0 .			—	—

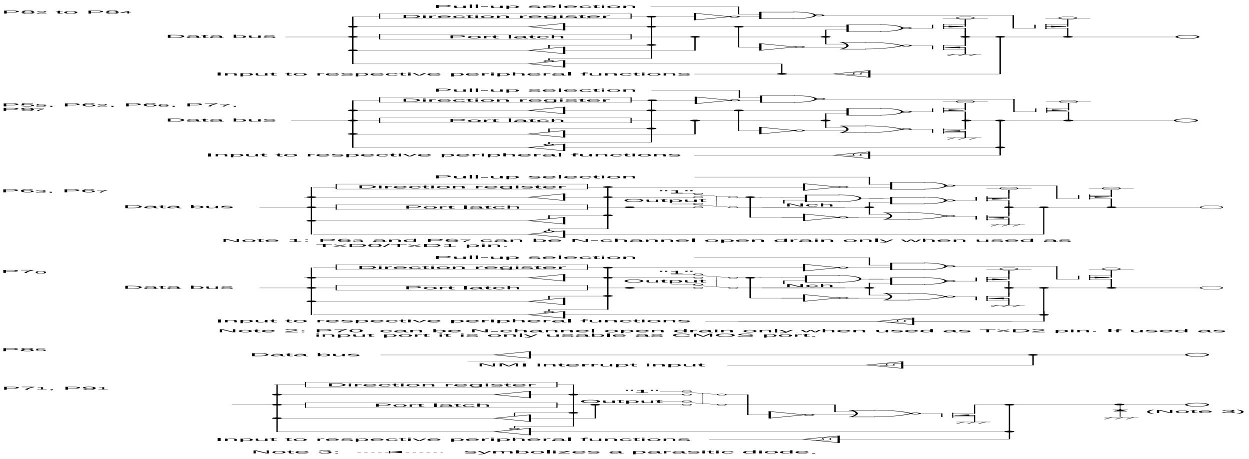
Figure 19-9. Port control register

Table 19-1. Example connection of unused pins in single-chip mode

Pin name	Connection
Ports P0 to P10 (excluding P8₅)	After setting for input mode, connect every pin to VSS or VCC via a resistor; or after setting for output mode, leave these pins open.
XOUT (Note)	Open
$\overline{\text{NMI}}$	Connect via resistor to VCC (pull-up)
AVCC	Connect to VCC
AVSS, VREF, BYTE	Connect to VSS

Note: With external clock input to XIN pin.

Table 19-2. Example connection of unused pins in memory expansion mode and microprocessor mode



Note 1: P6₃ and P6₇ can be N-channel open drain only when used as TxD0/TxD1 pin.

Note 2: P70 can be N-channel open drain only when used as TxD2 pin. If used as input port it is only usable as CMOS port.

Note 3: symbolizes a parasitic diode.

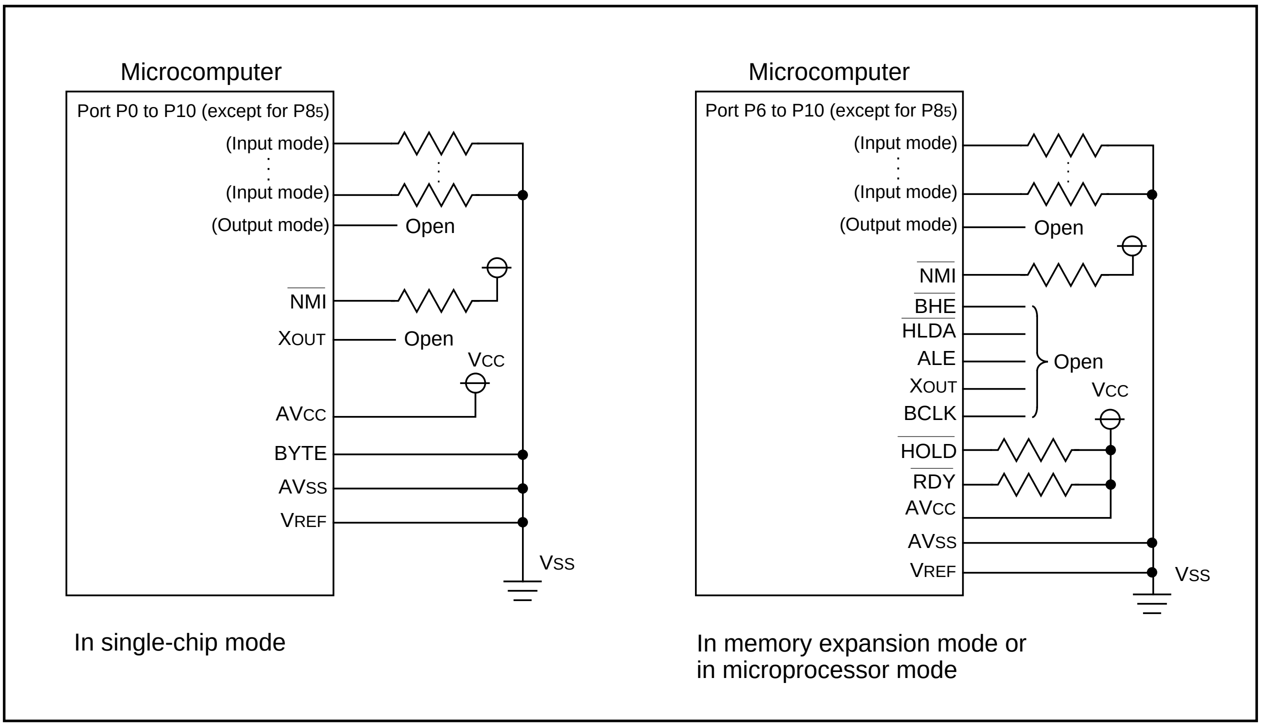


Figure 19-10. Example connection of unused pins

Usage Precaution

Timer A (timer mode)

(1) Reading the timer Ai register while a count is in progress allows reading, with arbitrary timing, the value of the counter. Reading the timer Ai register with the reload timing gets “$FFFF_{16}$”. Reading the timer Ai register after setting a value in the timer Ai register with a count halted but before the counter starts counting gets a proper value.

Timer A (event counter mode)

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

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

Timer A (one-shot timer mode)

(1) Setting the count start flag to “0” while a count is in progress causes as follows:

- The counter stops counting and a content of reload register is reloaded.
- The TAi_{OUT} pin outputs “L” level.
- The interrupt request generated and the timer Ai interrupt request bit goes to “1”.

(2) The timer Ai interrupt request bit goes to “1” if the timer's operation mode is set using any of the following procedures:

- Selecting one-shot timer mode after reset.
- Changing operation mode from timer mode to one-shot timer mode.
- Changing operation mode from event counter mode to one-shot timer mode.

Therefore, to use timer Ai interrupt (interrupt request bit), set timer Ai interrupt request bit to “0” after the above listed changes have been made.

Timer A (pulse width modulation mode)

(1) The timer Ai interrupt request bit becomes “1” if setting operation mode of the timer in compliance with any of the following procedures:

- Selecting PWM mode after reset.
- Changing operation mode from timer mode to PWM mode.
- Changing operation mode from event counter mode to PWM mode.

Therefore, to use timer Ai interrupt (interrupt request bit), set timer Ai interrupt request bit to “0” after the above listed changes have been made.

(2) Setting the count start flag to “0” while PWM pulses are being output causes the counter to stop counting. If the TAi_{OUT} pin is outputting an “H” level in this instance, the output level goes to “L”, and the timer Ai interrupt request bit goes to “1”. If the TAi_{OUT} pin is outputting an “L” level in this instance, the level does not change, and the timer Ai interrupt request bit does not becomes “1”.

Timer B (timer mode, event counter mode)

(1) Reading the timer Bi register while a count is in progress allows reading , with arbitrary timing, the value of the counter. Reading the timer Bi register with the reload timing gets “$FFFF_{16}$”. Reading the timer Bi register after setting a value in the timer Bi register with a count halted but before the counter starts counting gets a proper value.

Timer B (pulse period/pulse width measurement mode)

(1) If changing the measurement mode select bit is set after a count is started, the timer Bi interrupt request bit goes to “1”.

(2) When the first effective edge is input after a count is started, an indeterminate value is transferred to the reload register. At this time, timer Bi interrupt request is not generated.

A-D Converter

(1) Write to each bit (except bit 6) of A-D control register 0, to each bit of A-D control register 1, and to bit 0 of A-D control register 2 when A-D conversion is stopped (before a trigger occurs).
In particular, when the Vref connection bit is changed from “0” to “1”, start A-D conversion after an elapse of 1 µs or longer.

(2) When changing A-D operation mode, select analog input pin again.

(3) Using one-shot mode or single sweep mode
Read the correspondence A-D register after confirming A-D conversion is finished. (It is known by A-D conversion interrupt request bit.)

(4) Using repeat mode, repeat sweep mode 0 or repeat sweep mode 1
Use the undivided main clock as internal CPU clock.

Stop Mode and Wait Mode

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

Interrupts

(1) Reading address 00000_{16}

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

(2) Setting the stack pointer

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

(3) The NMI interrupt

- As for the NMI interrupt pin, an interrupt cannot be prohibited. Connect it to the V_{CC} pin if unused. Be sure to work on it.
- Do not get either into stop mode or into wait mode with the NMI pin set to “L”.

External ROM version

The external ROM version is operated only in microprocessor mode, so be sure to perform the following:

- Connect CNVss pin to Vcc.
- Fix the processor mode bit to “11_2”

Table 21-1. Absolute maximum ratings

Symbol	Parameter		Condition	Rated value	Unit
V_{CC}	Supply voltage		$V_{CC}=AV_{CC}$	-0.3 to 6.5	V
AV_{CC}	Analog supply voltage		$V_{CC}=AV_{CC}$	-0.3 to 6.5	V
V_I	Input voltage	$\overline{RESET}$, CNV_{SS}, BYTE, $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, V_{REF}, X_{IN}		-0.3 to Vcc+0.3	V
		$P7_1$, $P9_1$		-0.3 to 6.5	V
V_O	Output voltage	$P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$,$P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$,$P7_2$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, X_{OUT}		-0.3 to Vcc+0.3	V
		$P7_0$, $P7_1$,		-0.3 to 6.5	V
P_d	Power dissipation		Ta=25 °C	700	mW
T_{opr}	Operating ambient temperature			-40 to 85 (Note 1)	°C
T_{stg}	Storage temperature			-65 to 150	°C

Note 1: Specify a product of -40 to 85°C to use it.

Table 21-2. Recommended operating conditions (referenced to Vcc = 4.5 V to 5.5V at Ta = -40 to 85 °C (Note 3) unless otherwise specified)

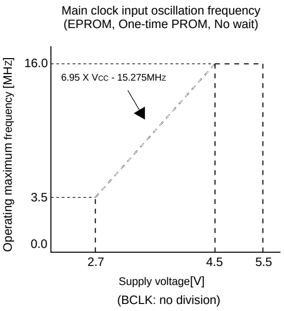
Symbol	Parameter				Standard			Unit
					Min	Typ.	Max.	
Vcc	Supply voltage				4.2	5.0	5.5	V
AVcc	Analog supply voltage					Vcc		V
Vss	Supply voltage					0		V
AVss	Analog supply voltage					0		V
V_{IH}	HIGH input voltage	$P3_1$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$, 0, $P7_2$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$, $P9_2$ to $P9_7$, $P10_0$ to $P10_7$, X_{IN}, RESET, CNV_{SS}, BYTE			0.8Vcc		Vcc	V
		$P9_1$, $P7_1$			0.8Vcc		6.5	V
		$P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ (during single-chip mode)			0.8Vcc		Vcc	V
		$P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ (data input function during memory expansion and microprocessor modes)			0.5Vcc		Vcc	V
V_{IL}	LOW input voltage	$P3_1$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, X_{IN}, RESET, CNV_{SS}, BYTE			0		0.2Vcc	V
		$P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ (during single-chip mode)			0		0.2Vcc	V
		$P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ (data input function during memory expansion and microprocessor modes)			0		0.16Vcc	V
$I_{OH\ (peak)}$	HIGH 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 $P6_7$,$P7_0$,$P7_2$ to $P7_7$, $P8_0$ to $P8_4$,$P8_6$,$P8_7$,$P9_0$,$P9_2$ to $P9_7$,$P10_0$ to $P10_7$					-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 $P10_7$					-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 $P10_7$					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 $P10_7$					5.0	mA
f (X_{IN})	Main clock input oscillation frequency	No wait	Mask ROM, Flash	Vcc=4.2V to 5.5V	0		16	MHz
				Vcc=4.2V to 5.5V	0		20	MHz
		With wait	Mask ROM, Flash	Vcc=4.2V to 5.5V	0		16	MHz
				Vcc=4.2V to 5.5V	0		20	MHz
f (X_{CIN})	Subclock oscillation frequency					32.768	50	kHz

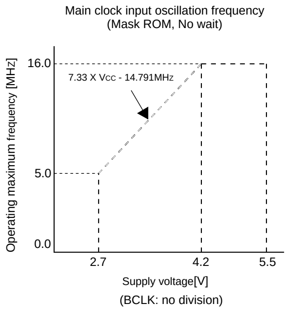
Note 1: The mean output current is the mean value within 100ms.

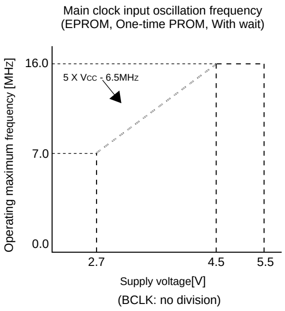
Note 2: The total I_{OL} (peak) for ports P0, P1, P2, $P8_6$, $P8_7$, P9, and P10 must be 80mA max. The total I_{OH} (peak) for ports P0, P1, P2, $P8_6$, $P8_7$, P9, and P10 must be 80mA max. The total I_{OL} (peak) for ports P3, P4, P5, P6, P7, and $P8_0$ to $P8_4$ must be 80mA max. The total I_{OH} (peak) for ports P3, P4, P5, P6, $P7_2$ to $P7_7$, and $P8_0$ to $P8_4$ must be 80mA max.

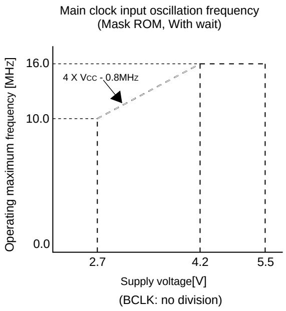
Note 3: Specify a product of -40 to 85°C to use it.

Note 4: Relationship between main clock oscillation frequency and supply voltage.









Note 5: Execute case without wait, program/erase of flash memory by Vcc = 4.2V to 5.5V and f(BCLK) ≤ 6.25 MHz. Execute case with wait, program/erase of flash memory by Vcc = 4.2V to 5.5V and f(BCLK) ≤ 12.5 MHz.

Table 21-3. Electrical characteristics (referenced to Vcc = AVcc = VREF = 5V, Vss = AVss = 0 V at Ta = 25 °C, f(XIN) = 16MHz unless otherwise specified)

Symbol	Parameter			Measuring condition		Standard			Unit
						Min	Typ.	Max.	
V_{OH}	HIGH output voltage	$P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$, $P7_2$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$, $P9_2$ to $P9_7$, $P10_0$ to $P10_7$		I_{OH}=-5mA		3.0			V
V_{OH}	HIGH output voltage	$P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$, $P7_2$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$, $P9_2$ to $P9_7$, $P10_0$ to $P10_7$		I_{OH}=-200μA		4.7			V
V_{OH}	HIGH output voltage	X_{OUT}	HIGHPOWER	I_{OH}=-1mA		3.0			V
			LOWPOWER	I_{OH}=-0.5mA		3.0			
	HIGH output voltage	X_{COUT}	HIGHPOWER	With no load applied			3.0		V
			LOWPOWER	With no load applied			1.6		
V_{OL}	LOW output voltage	$P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$		I_{OL}=5mA				2.0	V
V_{OL}	LOW output voltage	$P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$		I_{OL}=200μA				0.45	V
V_{OL}	LOW output voltage	X_{OUT}	HIGHPOWER	I_{OL}=1mA				2.0	V
			LOWPOWER	I_{OL}=0.5mA				2.0	
	LOW output voltage	X_{COUT}	HIGHPOWER	With no load applied			0		V
			LOWPOWER	With no load applied			0		
V_{T+} -V_{T-}	Hysteresis	$\overline{HOLD}$, $\overline{RDY}$, $TA0_{IN}$ to $TA4_{IN}$, $TB0_{IN}$ to $TB2_{IN}$, $\overline{INT_0}$ to $\overline{INT_5}$, $\overline{AD_{TRG}}$, $\overline{CTS_0}$, $\overline{CTS_1}$, CLK_0, CLK_1, $TA2_{OUT}$ to $TA4_{OUT}$, $\overline{NMI}$, $\overline{KI_0}$ to $\overline{KI_3}$				0.2		0.8	V
V_{T+} -V_{T-}	Hysteresis	$\overline{RESET}$				0.2		1.8	V
I_{IH}	HIGH input current	$P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, X_{IN}, $\overline{RESET}$, CNVss, BYTE		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}, $\overline{RESET}$, CNVss, BYTE		V_I=0V				-5.0	μA
R_{PULLUP}	Pull-up resistance	$P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$		V_I=0V		30.0	50.0	167.0	kΩ
R_{fXIN}	Feedback resistance X_{IN}						1.0		MΩ
R_{fCXIN}	Feedback resistance X_{CIN}						6.0		MΩ
V_{RAM}	RAM retention voltage			When clock is stopped		2.0			V
Icc	Power supply current			In single-chip mode, the output pins are open and other pins are Vss	f(X_{IN})=16MHz Square wave, no division		50.0	80.0	mA
					Mask ROM version f(X_{CIN})=32kHz Square wave		200.0		μA
					Flash memory 5V version f(X_{CIN})=32kHz Square wave		8.0		mA
					f(X_{CIN})=32kHz Square wave When a WAIT instruction is executed Timer A operates with fc32		4.0		μA
					Ring oscillation		9.0		mA
					Ta=25˚C when clock is stopped			1.0	μA
					Ta=85˚C when clock is stopped			20.0	

Vcc = 5 V

Table 21-4. A-D conversion characteristics (referenced to Vcc = AVcc = VREF = 5V, Vss = AVss = 0 V at Ta = 25 °C, f(XIN) = 16MHz unless otherwise specified)

Symbol	Parameter		Measuring condition		Standard Min.	Standard Typ.	Standard Max.	Unit
-	Resolution		V_{REF} = V_{CC}				10	Bits
-	Absolute accuracy	Sample & hold function disabled	V_{REF} = V_{CC} = 5V				±3	LSB
		Sample & hold function enabled10bit)	V_{REF} =V_{CC} = 5V	AN_0 to AN_7 input AN_{00} to AN_{07} input AN_{20} to AN_{27} input			±3	LSB
				ANEX0, ANEX1 input, External op-amp connection mode			±7	LSB
		Sample & hold function enabled8bit)	V_{REF} = V_{CC} = 5V				±2	LSB
R_{LADDER}	Ladder resistance		V_{REF} = V_{CC}		10		40	kΩ
t_{CONV}	Conversion time (10bit) (Note 1)				33			cycles
t_{CONV}	Conversion time (8bit) (Note 1)				28			cycles
t_{SAMP}	Sampling time				0.3			µs
V_{REF}	Reference voltage				2		V_{CC}	V
V_{IA}	Analog input voltage				0		V_{REF}	V

Note 1: The conversion times are given in cycles of fAD. fAD is derived from f(Xin) divided by 1, 2, 4, or 8 and may not exceed 10 MHz. Minimal conversion times are achieved with an f(Xin) of 10 MHz or 20 MHz.

Table 21-5. D-A conversion characteristics (referenced to Vcc = 5 V, VREF = 5V, Vss = AVss = 0 V at Ta = 25 °C, f(XIN) = 16MHz unless otherwise specified)

Symbol	Parameter	Measuring condition	Standard Min.	Standard Typ.	Standard Max.	Unit
−	Resolution				8	Bits
−	Absolute accuracy				1.0	%
t_{su}	Setup time				3	µs
R_O	Output resistance		4	10	20	kΩ
I_{VREF}	Reference power supply input current	(Note)			1.5	mA

Note: This applies when using one D-A converter, with the D-A register for the unused D-A converter set to "00_{16}". The A-D converter's ladder resistance is not included. Also, when the V_{REF} is unconnected at the A-D control register, I_{VREF} is sent.

Vcc = 5 V

Timing requirements (referenced to Vcc = 5 V, Vss = 0 V at Ta = 25 °C unless otherwise specified)

Table 21-6. External clock input

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t_c	External clock input cycle time	62.5		ns
$t_{w(H)}$	External clock input HIGH pulse width	25		ns
$t_{w(L)}$	External clock input LOW pulse width	25		ns
t_r	External clock rise time		15	ns
t_f	External clock fall time		15	ns

Table 21-7. Memory expansion- and microprocessor modes

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{ac1(RD-DB)}$	Data input access time (no wait)		(Note)	ns
$t_{ac2(RD-DB)}$	Data input access time (with wait)		(Note)	ns
$t_{ac3(RD-DB)}$	Data input access time (when accessing multiplex bus area)		(Note)	ns
$t_{su(DB-RD)}$	Data input setup time	40		ns
$t_{su(RDY-BCLK)}$	$\overline{\text{RDY}}$ input setup time	30		ns
$t_{su(HOLD-BCLK)}$	$\overline{\text{HOLD}}$ input setup time	40		ns
$t_{h(RD-DB)}$	Data input hold time	0		ns
$t_{h(BCLK-RDY)}$	$\overline{\text{RDY}}$ input hold time	0		ns
$t_{h(BCLK-HOLD)}$	$\overline{\text{HOLD}}$ input hold time	0		ns
$t_{d(BCLK-HLDA)}$	$\overline{\text{HLDA}}$ output delay time		40	ns

Note: Calculated according to the BCLK frequency as follows:

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

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

$$t_{ac3}(RD-DB) = \frac{3 \times 10^9}{f(BCLK) \times 2} - 45 \quad [ns]$$

Vcc = 5 V

Timing requirements (referenced to Vcc = 5 V, Vss = 0 V at Ta = 25 °C unless otherwise specified)

Table 21-8. 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 21-9. 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 21-10. Timer A input (gating input in 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 21-11. Timer A input (external trigger input in one-shot timer 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 21-12. Timer A input (up/down input in event counter mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(UP)}$	TAi$_{OUT}$ input cycle time	2000		ns
$t_{w(UPH)}$	TAi$_{OUT}$ input HIGH pulse width	1000		ns
$t_{w(UPL)}$	TAi$_{OUT}$ input LOW pulse width	1000		ns
$t_{su(UP-T_{IN})}$	TAi$_{OUT}$ input setup time	400		ns
$t_{h(T_{IN}-UP)}$	TAi$_{OUT}$ input hold time	400		ns

Vcc = 5 V

Timing requirements (referenced to Vcc = 5 V, Vss = 0 V at Ta = 25 °C unless otherwise specified)

Table 21-13. Timer B input (counter input in event counter mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(TB)}$	TBi$_{IN}$ input cycle time (counted on one edge)	100		ns
$t_{w(TBH)}$	TBi$_{IN}$ input HIGH pulse width (counted on one edge)	40		ns
$t_{w(TBL)}$	TBi$_{IN}$ input LOW pulse width (counted on one edge)	40		ns
$t_{c(TB)}$	TBi$_{IN}$ input cycle time (counted on both edges)	200		ns
$t_{w(TBH)}$	TBi$_{IN}$ input HIGH pulse width (counted on both edges)	80		ns
$t_{w(TBL)}$	TBi$_{IN}$ input LOW pulse width (counted on both edges)	80		ns

Table 21-14. Timer B input (pulse period measurement mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(TB)}$	TBi$_{IN}$ input cycle time	400		ns
$t_{w(TBH)}$	TBi$_{IN}$ input HIGH pulse width	200		ns
$t_{w(TBL)}$	TBi$_{IN}$ input LOW pulse width	200		ns

Table 21-15. Timer B input (pulse width measurement mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(TB)}$	TBi$_{IN}$ input cycle time	400		ns
$t_{w(TBH)}$	TBi$_{IN}$ input HIGH pulse width	200		ns
$t_{w(TBL)}$	TBi$_{IN}$ input LOW pulse width	200		ns

Table 21-16. A-D trigger input

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(AD)}$	$\overline{AD_{TRG}}$ input cycle time (trigger able minimum)	1000		ns
$t_{w(ADL)}$	$\overline{AD_{TRG}}$ input LOW pulse width	125		ns

Table 21-17. 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 21-18. External interrupt $\overline{INTi}$ inputs

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

Vcc = 5 V

Switching characteristics (referenced to Vcc = 5 V, Vss = 0 V at Ta = 25 °C, CM15 ="1" unless otherwise specified)

Table 21-19. Memory expansion mode and microprocessor mode (no wait)

Symbol	Parameter	Measuring condition	Standard		Unit
			Min.	Max.	
$t_{d(BCLK-AD)}$	Address output delay time	Figure 1.26.1		25	ns
$t_{h(BCLK-AD)}$	Address output hold time (BCLK standard)		4		ns
$t_{h(RD-AD)}$	Address output hold time (RD standard)		0		ns
$t_{h(WR-AD)}$	Address output hold time (WR standard)		0		ns
$t_{d(BCLK-CS)}$	Chip select output delay time			25	ns
$t_{h(BCLK-CS)}$	Chip select output hold time (BCLK standard)		4		ns
$t_{d(BCLK-ALE)}$	ALE signal output delay time			25	ns
$t_{h(BCLK-ALE)}$	ALE signal output hold time		– 4		ns
$t_{d(BCLK-RD)}$	RD signal output delay time			25	ns
$t_{h(BCLK-RD)}$	RD signal output hold time		0		ns
$t_{d(BCLK-WR)}$	WR signal output delay time			25	ns
$t_{h(BCLK-WR)}$	WR signal output hold time		0		ns
$t_{d(BCLK-DB)}$	Data output delay time (BCLK standard)			40	ns
$t_{h(BCLK-DB)}$	Data output hold time (BCLK standard)		4		ns
$t_{d(DB-WR)}$	Data output delay time (WR standard)		(Note1)		ns
$t_{h(WR-DB)}$	Data output hold time (WR standard)(Note2)		0		ns

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

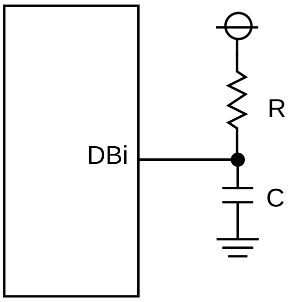
$$td(DB-WR) = \frac{10^{9}}{f(BCLK) \text{ X } 2} - 40 \quad [ns]$$

Note 2: This is standard value shows the timing when the output is off, and doesn't show hold time of data bus.
Hold time of data bus is different by capacitor volume and pull-up (pull-down) resistance value.
Hold time of data bus is expressed in

$$t = -CR \text{ X } \ln(1 - V_{OL} / V_{CC})$$

by a circuit of the right figure.
For example, when $V_{OL} = 0.2V_{CC}$, C = 30pF, R = 1kΩ, hold time of output “L” level is

$$t = -30pF \text{ X } 1k\Omega \text{ X } \ln(1 - 0.2V_{CC} / V_{CC}) = 6.7ns.$$



Vcc = 5 V

Switching characteristics (referenced to Vcc = 5 V, Vss = 0 V at Ta = -25 °C, CM15 ="1" unless otherwise specified)

Table 21-20. Memory expansion mode and microprocessor mode (with wait, accessing external memory)

Symbol	Parameter	Measuring condition	Standard		Unit
			Min.	Max.	
$t_{d(BCLK-AD)}$	Address output delay time	Figure 1.26.1		25	ns
$t_{h(BCLK-AD)}$	Address output hold time (BCLK standard)		4		ns
$t_{h(RD-AD)}$	Address output hold time (RD standard)		0		ns
$t_{h(WR-AD)}$	Address output hold time (WR standard)		0		ns
$t_{d(BCLK-CS)}$	Chip select output delay time			25	ns
$t_{h(BCLK-CS)}$	Chip select output hold time (BCLK standard)		4		ns
$t_{d(BCLK-ALE)}$	ALE signal output delay time			25	ns
$t_{h(BCLK-ALE)}$	ALE signal output hold time		– 4		ns
$t_{d(BCLK-RD)}$	RD signal output delay time			25	ns
$t_{h(BCLK-RD)}$	RD signal output hold time		0		ns
$t_{d(BCLK-WR)}$	WR signal output delay time			25	ns
$t_{h(BCLK-WR)}$	WR signal output hold time		0		ns
$t_{d(BCLK-DB)}$	Data output delay time (BCLK standard)			40	ns
$t_{h(BCLK-DB)}$	Data output hold time (BCLK standard)		4		ns
$t_{d(DB-WR)}$	Data output delay time (WR standard)		(Note1)		ns
$t_{h(WR-DB)}$	Data output hold time (WR standard)(Note2)		0		ns

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

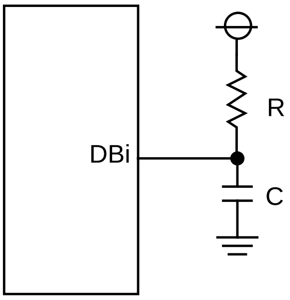
$$td(DB – WR) = \frac{10^{9}}{f(BCLK)} – 40 \quad [ns]$$

Note 2: This is standard value shows the timing when the output is off, and doesn't show hold time of data bus.
Hold time of data bus is different by capacitor volume and pull-up (pull-down) resistance value.
Hold time of data bus is expressed in

$$t = –CR \times \ln (1 – V_{OL} / V_{CC})$$

by a circuit of the right figure.
For example, when $V_{OL} = 0.2V_{CC}$, C = 30pF, R = 1kΩ, hold time of output “L” level is

$$t = – 30pF \times 1k\Omega \times \ln (1 – 0.2V_{CC} / V_{CC}) = 6.7ns.$$



Vcc = 5 V

Switching characteristics (referenced to Vcc = 5 V, Vss = 0 V at Ta = -25 °C, CM15 ="1" unless otherwise specified)

Table 21-21. Memory expansion mode and microprocessor mode (with wait, accessing external memory, multiplex bus area selected)

Symbol	Parameter	Measuring condition	Standard		Unit
			Min.	Max.	
$t_{d(BCLK-AD)}$	Address output delay time	Figure 1.26.1		25	ns
$t_{h(BCLK-AD)}$	Address output hold time (BCLK standard)		4		ns
$t_{h(RD-AD)}$	Address output hold time (RD standard)		(Note)		ns
$t_{h(WR-AD)}$	Address output hold time (WR standard)		(Note)		ns
$t_{d(BCLK-CS)}$	Chip select output delay time			25	ns
$t_{h(BCLK-CS)}$	Chip select output hold time (BCLK standard)		4		ns
$t_{h(RD-CS)}$	Chip select output hold time (RD standard)		(Note)		ns
$t_{h(WR-CS)}$	Chip select output hold time (WR standard)		(Note)		ns
$t_{d(BCLK-RD)}$	RD signal output delay time			25	ns
$t_{h(BCLK-RD)}$	RD signal output hold time		0		ns
$t_{d(BCLK-WR)}$	WR signal output delay time			25	ns
$t_{h(BCLK-WR)}$	WR signal output hold time		0		ns
$t_{d(BCLK-DB)}$	Data output delay time (BCLK standard)			40	ns
$t_{h(BCLK-DB)}$	Data output hold time (BCLK standard)		4		ns
$t_{d(DB-WR)}$	Data output delay time (WR standard)		(Note)		ns
$t_{h(WR-DB)}$	Data output hold time (WR standard)		(Note)		ns
$t_{d(BCLK-ALE)}$	ALE signal output delay time (BCLK standard)			25	ns
$t_{h(BCLK-ALE)}$	ALE signal output hold time (BCLK standard)		– 4		ns
$t_{d(AD-ALE)}$	ALE signal output delay time (Address standard)		(Note)		ns
$t_{h(ALE-AD)}$	ALE signal output hold time (Adderss standard)		50		ns
$t_{d(AD-RD)}$	Post-address RD signal output delay time		0		ns
$t_{d(AD-WR)}$	Post-address WR signal output delay time		0		ns
$t_{dZ(RD-AD)}$	Address output floating start time			8	ns

Note: Calculated according to the BCLK frequency as follows:

$$th(RD-AD) = \frac{10^9}{f(BCLK) \times 2} \quad [ns]$$

$$th(WR-AD) = \frac{10^9}{f(BCLK) \times 2} \quad [ns]$$

$$th(RD-CS) = \frac{10^9}{f(BCLK) \times 2} \quad [ns]$$

$$th(WR-CS) = \frac{10^9}{f(BCLK) \times 2} \quad [ns]$$

$$td(DB-WR) = \frac{10^9 \times 3}{f(BCLK) \times 2} - 40 \quad [ns]$$

$$th(WR-DB) = \frac{10^9}{f(BCLK) \times 2} \quad [ns]$$

$$td(AD-ALE) = \frac{10^9}{f(BCLK) \times 2} - 25 \quad [ns]$$

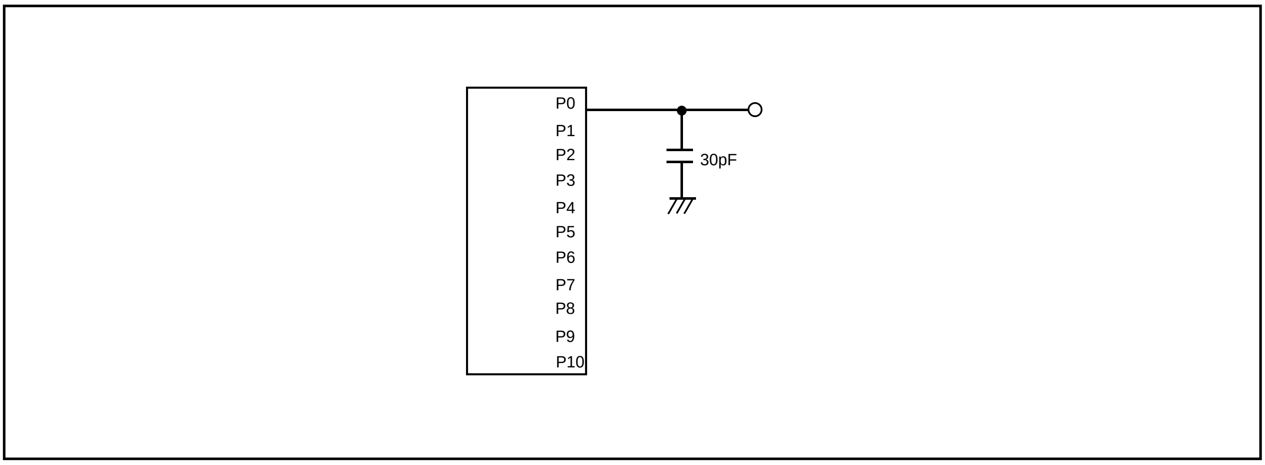


Figure 21-1. Port P0 to P10 measurement circuit

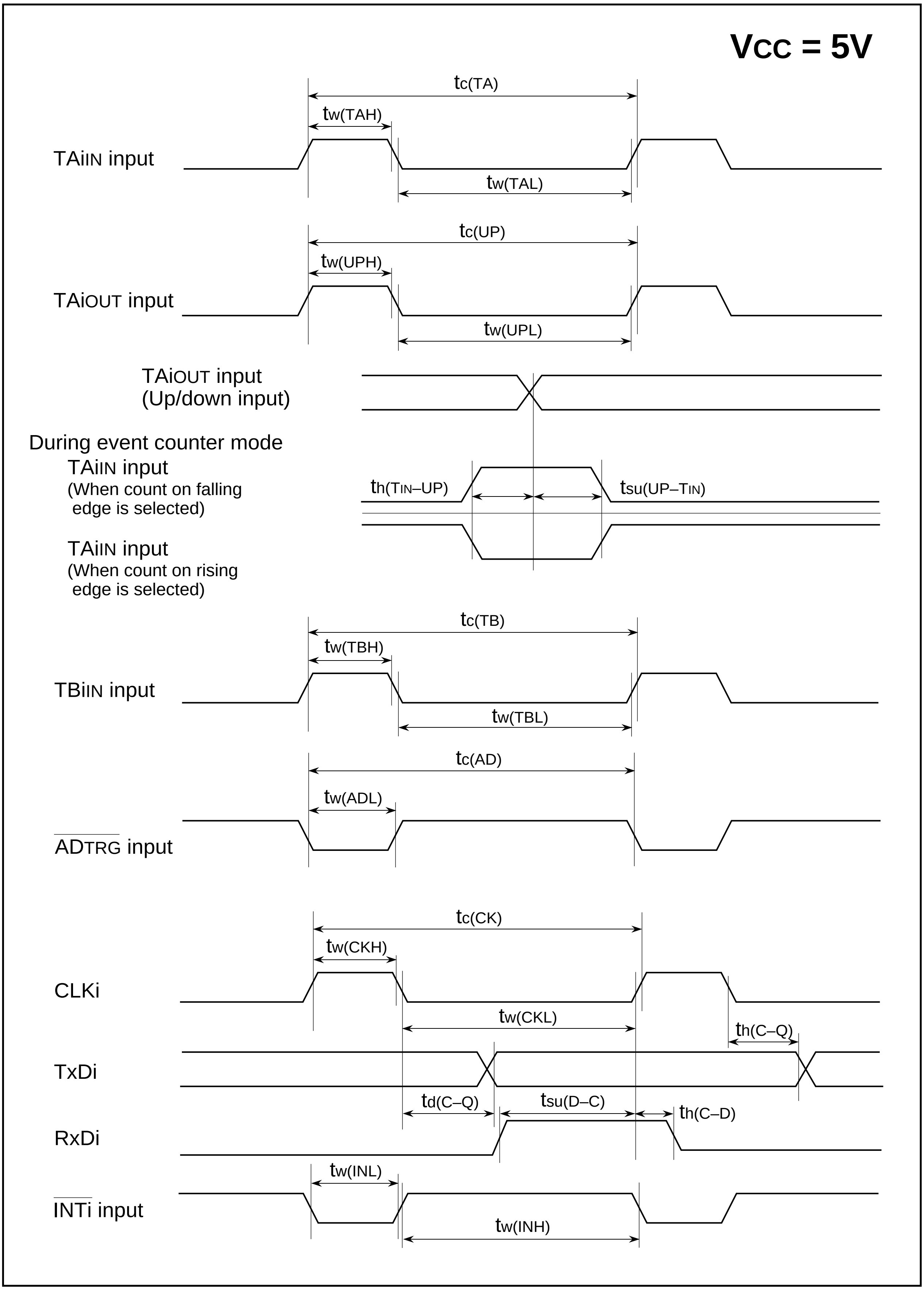
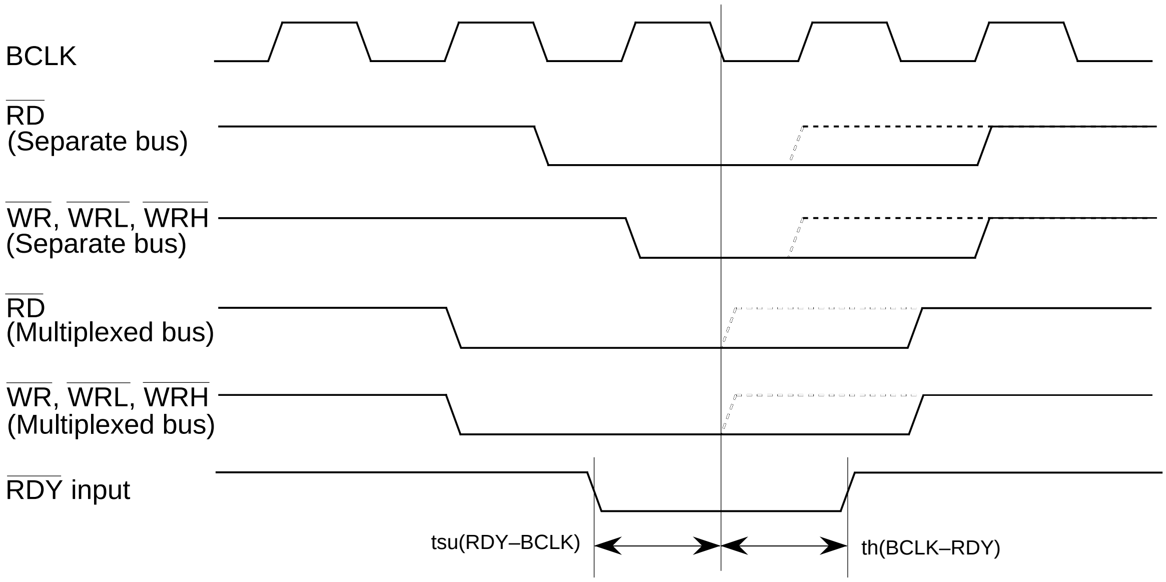


Figure 21-2. Vcc = 5V timing diagram

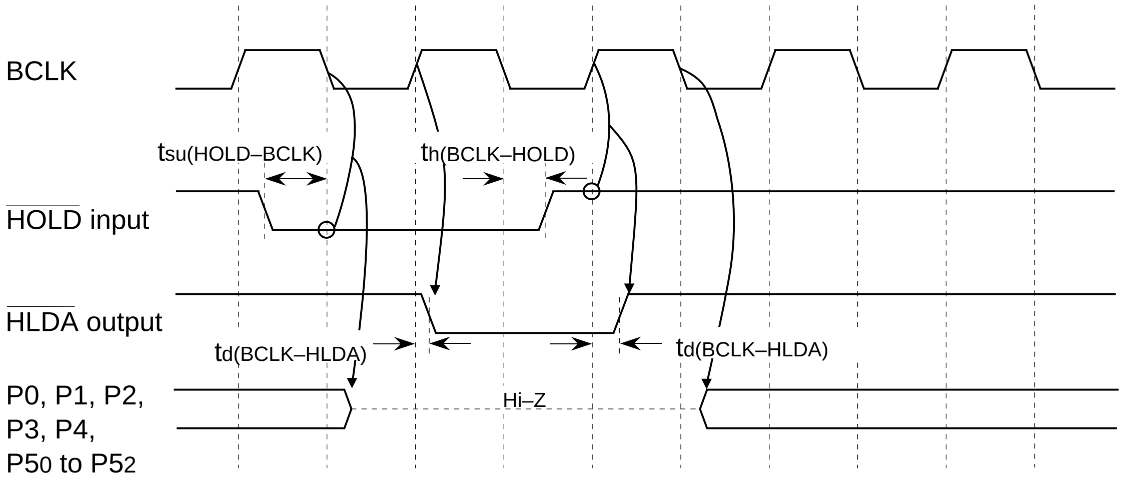
Vcc = 5V

Memory Expansion Mode and Microprocessor Mode

(Valid only with wait)



(Valid with or without wait)



Note: The above pins are set to high-impedance regardless of the input level of the BYTE pin and bit (PM06) of processor mode register 0 selects the function of ports P40 to P43.

Measuring conditions :

- V_{CC}=5V
- Input timing voltage : Determined with V_{IL}=1.0V, V_{IH}=4.0V
- Output timing voltage : Determined with V_{OL}=2.5V, V_{OH}=2.5V

Figure 21-3. Vcc = 5V timing diagram

Memory Expansion Mode and Microprocessor Mode
(With no wait)

Vcc = 5V

Read timing

Write timing

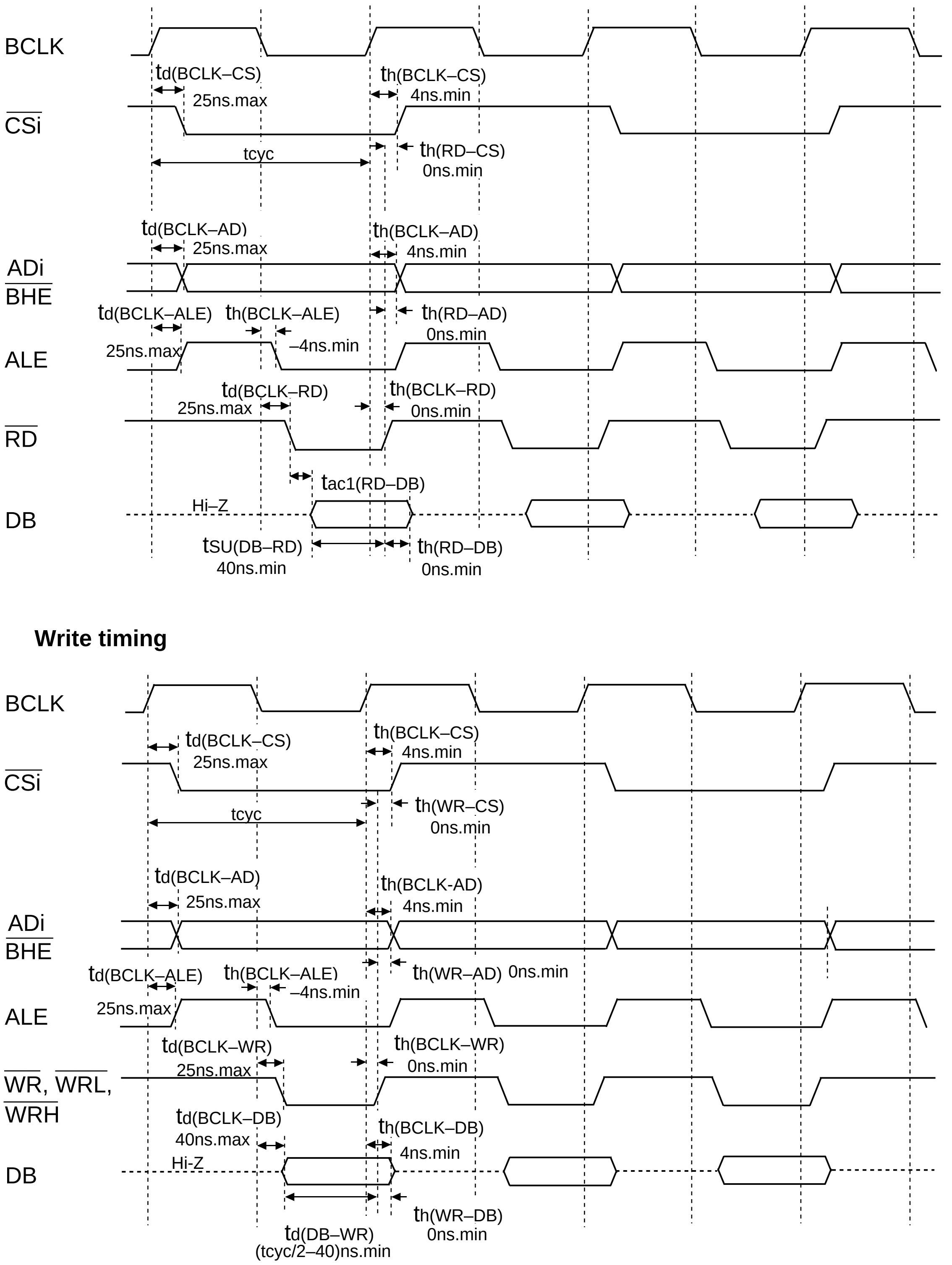
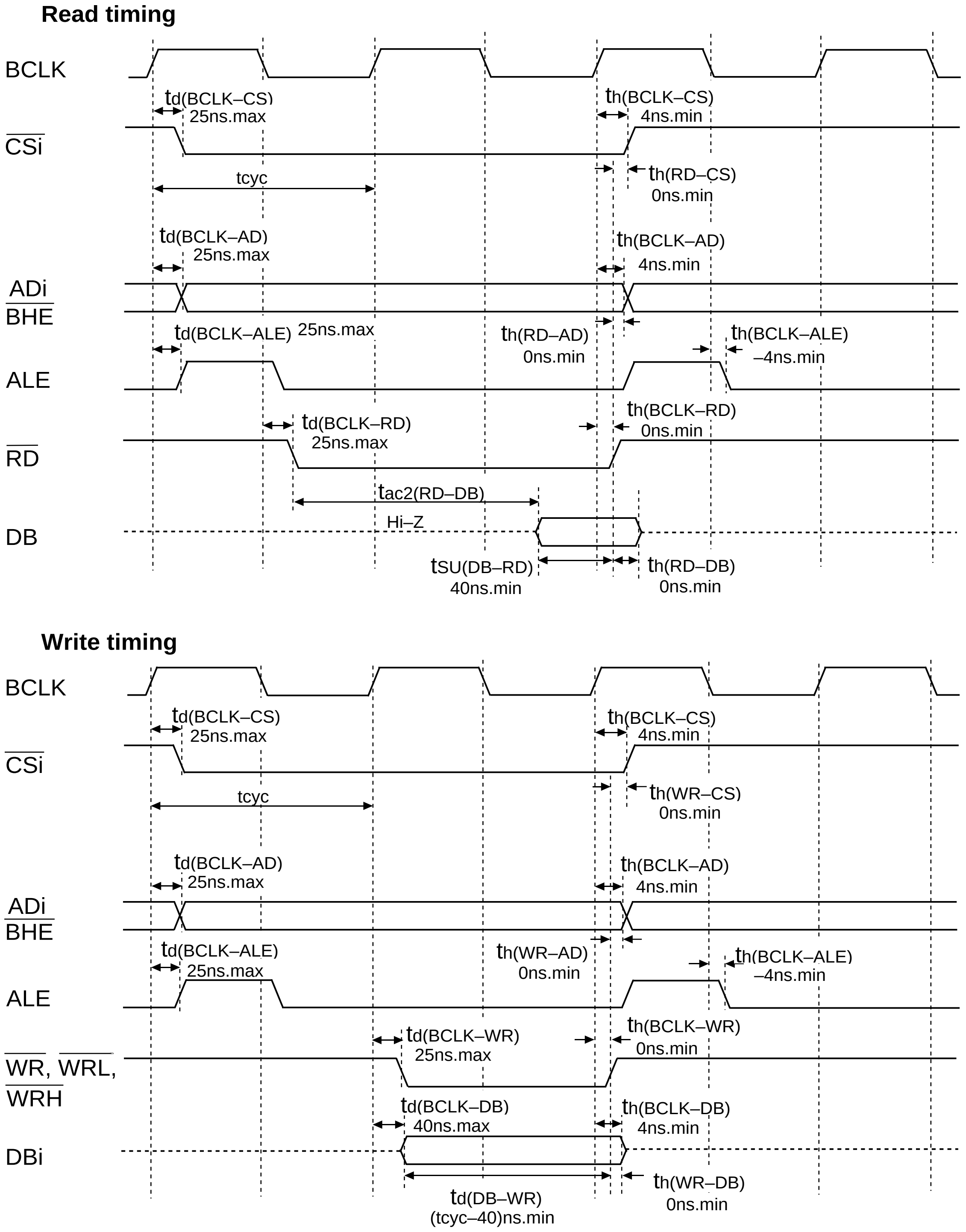


Figure 21-4. Vcc = 5V timing diagram

Memory Expansion Mode and Microprocessor Mode

(When accessing external memory area with wait)

Vcc = 5V

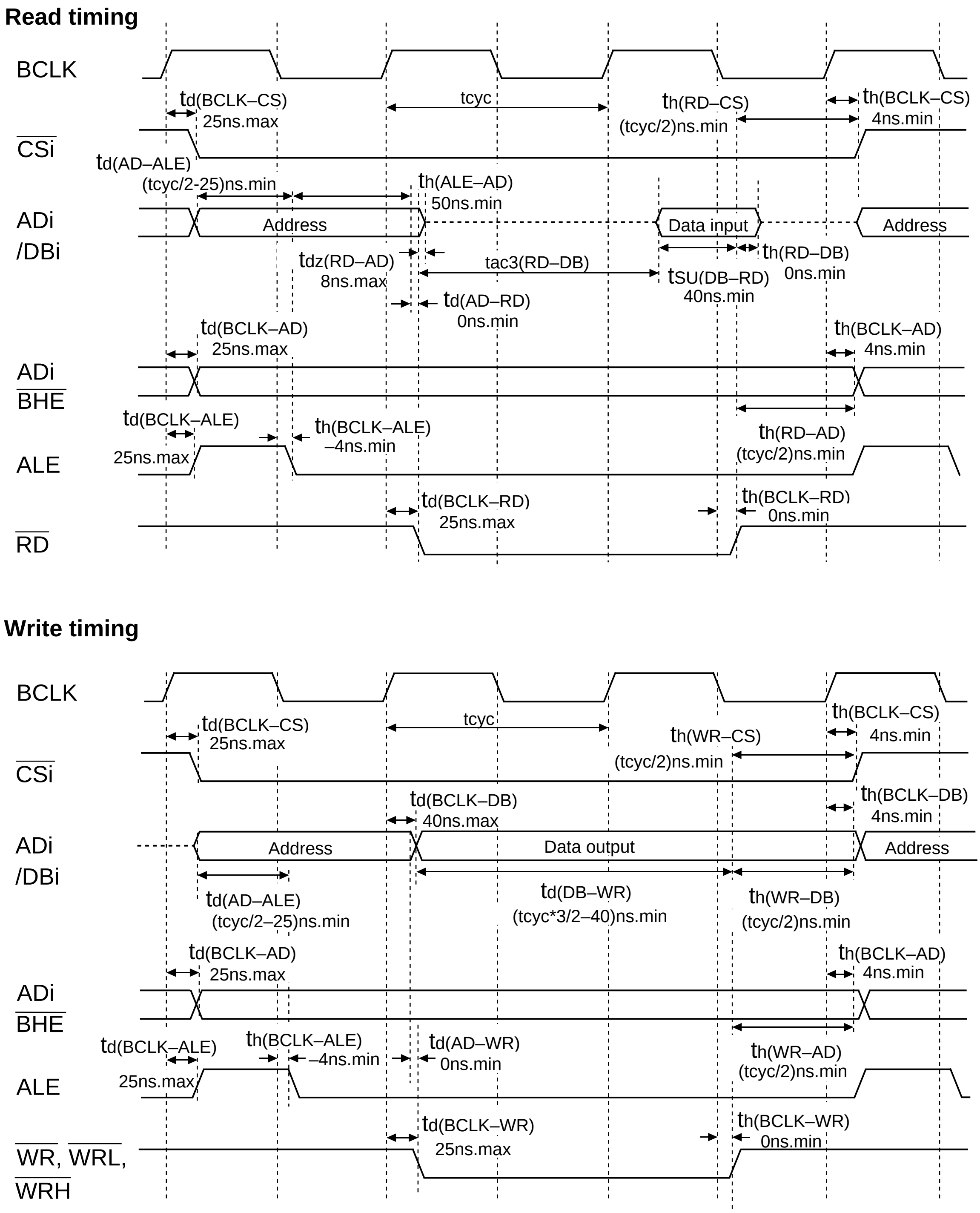


Measuring conditions :
- VCC=5V
- Input timing voltage : Determined with: VIL=0.8V, VIH=2.5V
- Output timing voltage : Determined with: VOL=0.8V, VOH=2.0V

Figure 21-5. Vcc = 5V timing diagram

Memory Expansion Mode and Microprocessor Mode **VCC = 5V**

(When accessing external memory area with wait, and select multiplexed bus)



Measuring conditions :
- VCC=5V
- Input timing voltage : Determined with VIL=0.8V, VIH=2.5V
- Output timing voltage : Determined with VOL=0.8V, VOH=2.0V

Figure 21-6. Vcc = 5V timing diagram

Outline Performance

Table 22-1 shows the outline performance of the M16C/6N (with on-chip flash memory).

Table 22-1. Outline performance of the M16C/6N (with on-chip flash memory)

Item		Performance
Power supply voltage		5V version: 4.5 to 5.5 V (f(XIN)=16MHz, without wait, 4.2 to 5.5V)
Program/erase voltage		5V version: 4.5 to 5.5 V (f(XIN)=12.5MHz, with one wait)
Flash memory mode		Three modes (parallel I/O, standard serial I/O, CPU rewrite)
Erase block division	User ROM area	See Figure 22-3.
	Boot ROM area	One division (8 Kbytes) (Note 1)
Program method		In units of pages (in units of 256 bytes)
Erase method		Collective erase/block erase
Program/erase control method		Program/erase control by software command
Protect method		Protected for each block by lock bit
Number of commands		8 commands
Program/erase count		100 times
ROM code protect		Parallel I/O and standard serial modes are supported.

Note 1: The boot ROM area contains a standard serial I/O mode control program which is stored in it when shipped from the factory. This area can be erased and programmed in only parallel I/O mode.

Table 22-2. Power supply current (typ.) of the M16C/6N (flash memory version)

Parameter	Measuring condition	Standard (Typ.)			Remark
		Read	Program	Erase	
5 V power supply current (5 V version)	f(Xin)=16 MHz, without wait, no division	35 mA	28 mA	25 mA	Division by 4 in program/erase

The following shows Mitsubishi plans to develop a line of M16C/6N products (with on-chip flash memory).

(1) ROM size

(2) Package 100P6S-A ... Plastic molded QFP

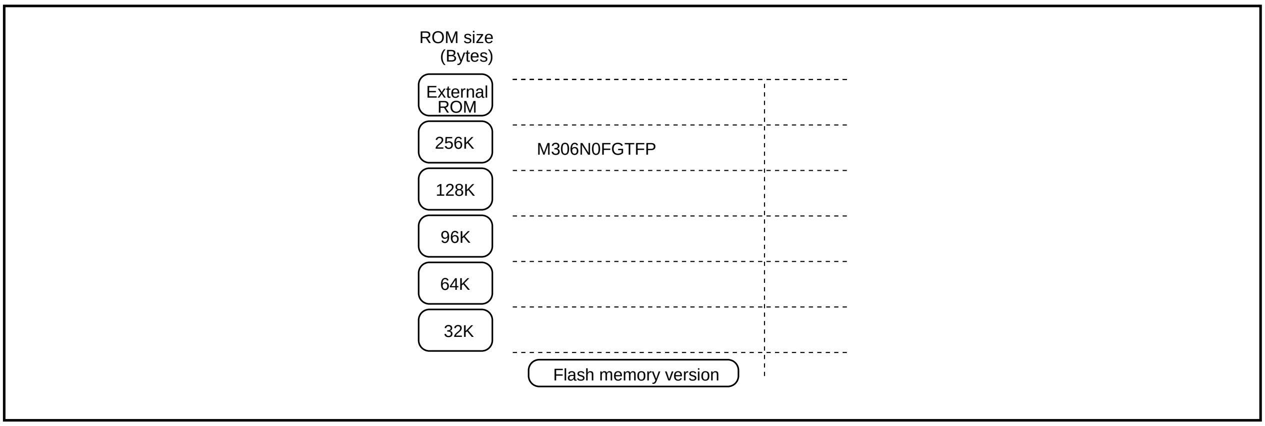


Figure 22-1. ROM expansion

The following lists the M16C/6N products to be supported in the future.

Table 22-3. Product list

Parameter	Measuring condition	Standard (Typ.)			Remark
		Read	Program	Erase	
5 V power supply current (5 V version)	f(Xin)=16 MHz, without wait, no division	35 mA	28 mA	25 mA	Division by 4 in program/erase

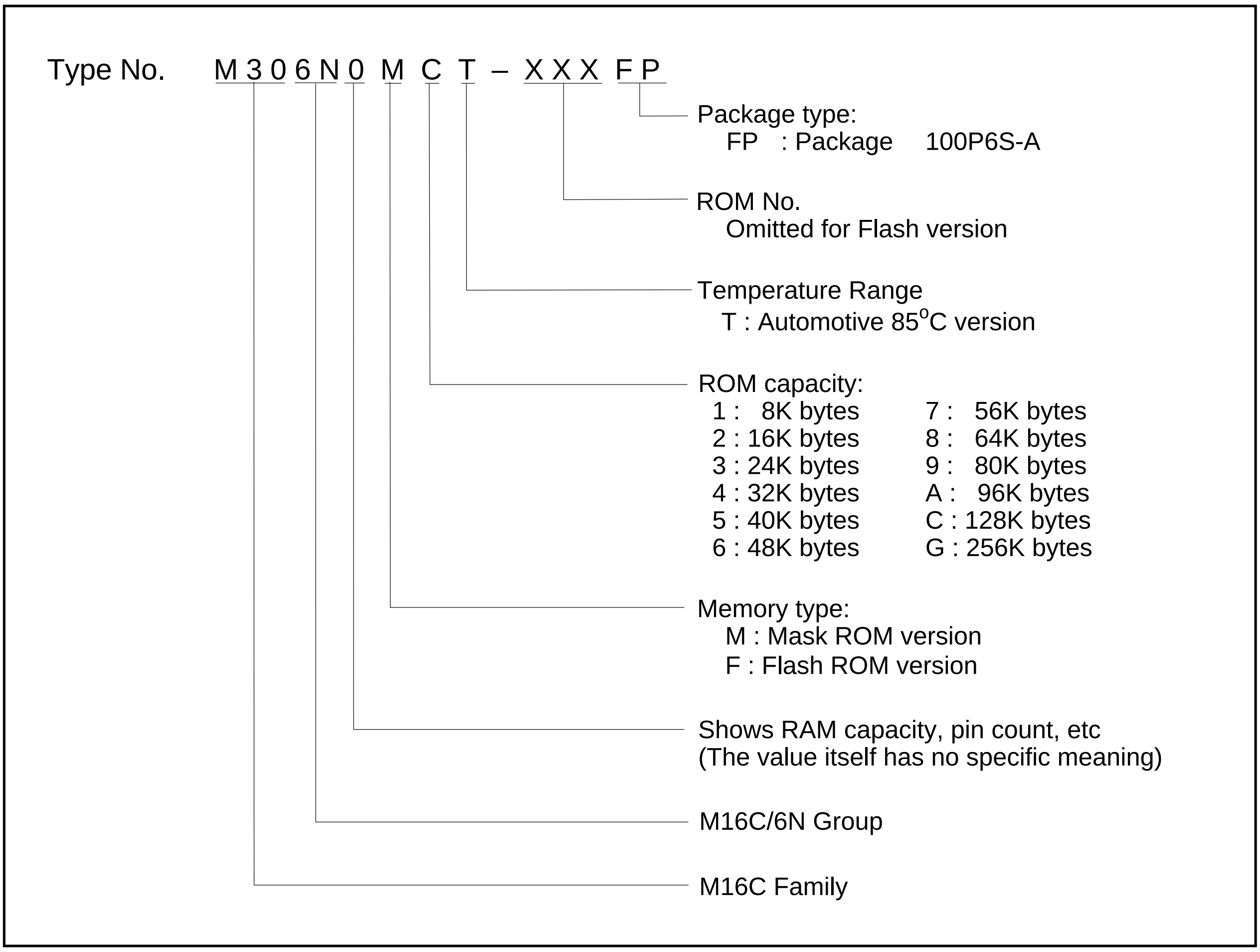
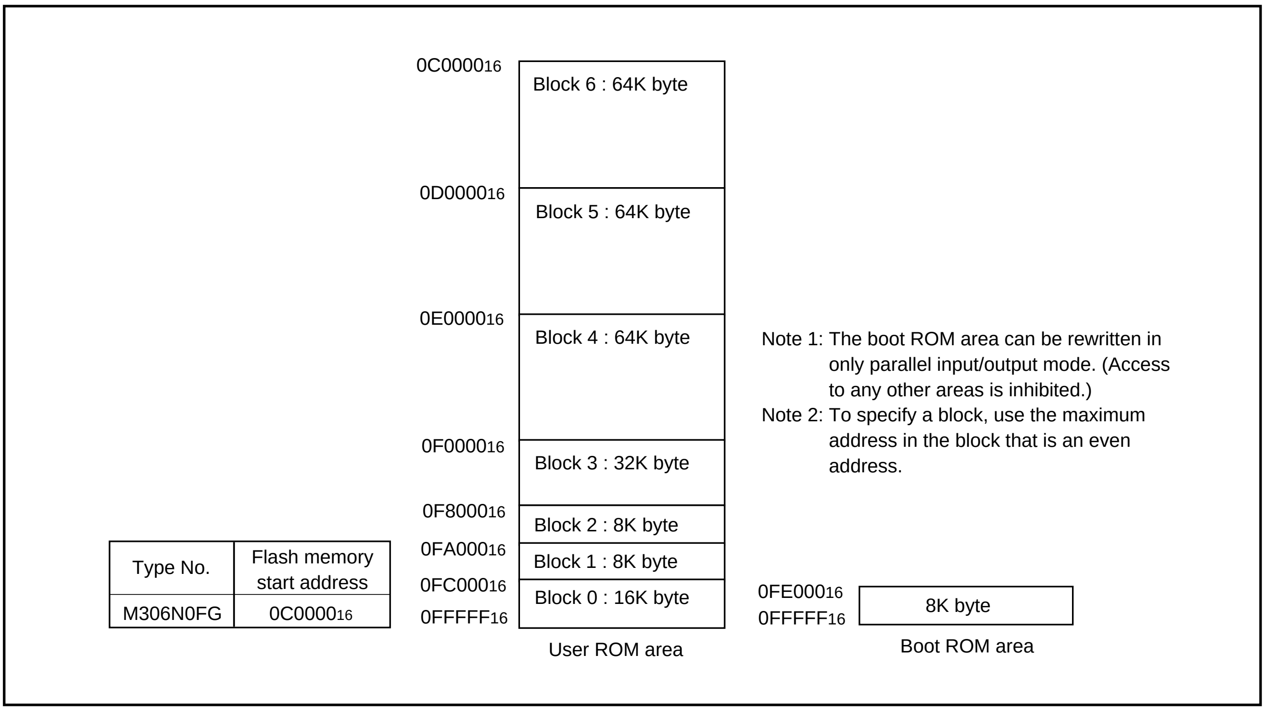


Figure 22-2. Type names, memory sizes and package

Flash Memory Modes

The M16C/6N (with on-chip flash memory) contains the DINOR (Divided bit line NOR) type of flash memory that can be rewritten with a single voltage of 5 V or 3.3 V. For this flash memory, three flash memory modes are available in which to read, program and erase: parallel I/O and standard serial I/O modes in which the flash memory can be manipulated using a programmer and a CPU rewrite mode in which the flash memory can be manipulated by the Central Processing Unit (CPU). Each mode is detailed in the pages to follow. The flash memory is divided into several blocks as shown in Figure 22-3, so that memory can be erased one block at a time. Each block has a lock bit to enable or disable execution of an erase or program operation, allowing for data in each block to be protected.

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



Type No.	Flash memory start address
M306N0FG	0C0000$_{16}$

Note 1: The boot ROM area can be rewritten in only parallel input/output mode. (Access to any other areas is inhibited.)
Note 2: To specify a block, use the maximum address in the block that is an even address.

Figure 22-3. Block diagram of on-chip flash memory

CPU Rewrite Mode

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

In CPU rewrite mode, it is possible to write only in the user ROM area in Figure 22-3; in the boot ROM area not possible. Make sure the program and block erase commands are issued only for the user ROM area and each block area.

The control program for CPU rewrite mode can be stored in either user ROM- or boot ROM area. In CPU rewrite mode, since the flash memory cannot be accessed for read by the CPU, use the rewrite control program except in the internal flash memory.

Boot Mode

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

See Figure 23-3 for details about the boot ROM area.

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

When the microcomputer is reset by pulling the $P5_5$(EPM) pin low, the CNVss pin high, and the $P5_0$(CE) pin high, the CPU start operating, using the control program in the boot ROM area. This mode is called the "boot" mode. The control program in the boot ROM area can also be used to rewrite the user ROM area.

Block Address

Block addresses refer to the maximum even address of each block. These addresses are used in the block erase command, erase all unlock blocks command, lock bit program command and read lock status command.

Outline Performance (CPU Rewrite Mode)

The CPU rewrite mode can be executed in single-chip mode, memory expansion mode and boot mode, allowing for only the user ROM area to be rewritten.

In CPU rewrite mode, the on-chip flash memory is operated on for erase, program or read operation by the CPU by writing a software command. Note that in this case the control program may not be located in the internal flash memory. For example, in single-chip mode, transferred into internal RAM.

When the CPU rewrite mode select bit (bit 1 at address $03B7_{16}$) is set to 1, transition to CPU rewrite mode occurs and software commands can be accepted.

In CPU rewrite mode, all software commands and data are written into and read from even addresses (address A0 of byte address = 0) 16 bits at a time. Therefore, make sure 8-bit software commands are always written into even addresses. Data at odd addresses have no effect.

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

Figure 23-1 shows the flash memory control register. Bit 0 is the RY/$\overline{\text{BY}}$ status flag, a read-only bit indicating the operating status of the flash memory. This flag is 0 (busy) during auto write and auto erase operation; otherwise, it is 1 (ready). (Its function is equivalent to that of the RY/$\overline{\text{BY}}$ pin in parallel I/O mode.) Bit 1 is the CPU rewrite mode select bit. The CPU rewrite mode is entered by setting this bit to 1, so that software commands become acceptable. In CPU rewrite mode, the CPU becomes unable to access the on-chip flash memory directly. Therefore, use the control program except in the internal flash memory to set this bit to 0. For this bit to be set to 1, the user needs to write a 0 and then a 1 in it in succession. The bit can be set to 0 by writing a 0 only.

Bit 2 is a lock bit disable bit. By setting this bit to 1, it is possible to disable erase and write protect (block lock) effectuated by the lock bit data. (This function is equivalent to that of the WP pin in parallel I/O mode.) The lock bit disable bit only disables the function of the lock bit and cannot set the lock bit itself. However, if an erase operation is performed when this bit is 1, the lock bit data that is 0 (locked) is set to 1 (unlocked) after erasure. For this bit to be set to 1, it is necessary to write a 0 and then a 1 in it in succession when the CPU rewrite mode select bit is 1. This bit can be manipulated only when the CPU rewrite mode select bit is 1.

Bit 3 is a flash memory reset bit, provided to reset the control circuit of the on-chip flash memory. This bit is used when exiting CPU rewrite mode and when flash memory access has failed. If this bit is set to 1 when the CPU rewrite mode select bit is 1, the flash memory is reset. To deassert this reset, the bit needs to be cleared to 0 after being set to 1.

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

Figure 23-2 shows a flowchart to set and reset the CPU rewrite mode. Always be sure to follow this flowchart.

Flash memory control register

b7	b6	b5	b4	b3	b2	b1	b0
☒	☒		0				

Symbol	Address	When reset
FMCR	03B7$_{16}$	XX000001$_2$

Bit symbol	Bit name	Function	R	W
FMCR0	RY/$\overline{\text{BY}}$ status flag	0: Busy (being written or erased) 1: Ready	○	×
FMCR1	CPU rewrite mode select bit (Note 2)	0: Normal mode (Software commands invalid) 1: CPU rewrite mode (Software commands acceptable)	○	○
FMCR2	Lock bit disable bit (Note 3)	0: Block lock by lock bit data is enabled 1: Block lock by lock bit data is disabled	○	○
FMCR3	Flash memory reset bit (Note 3)	0: Normal operation 1: Reset	○	○
This bit must always be set to 0.			○	○
FMCR5	User ROM area select bit (Note 5) (Effective in only boot mode)	0: Boot ROM area is accessed 1: User ROM area is accessed	○	○
Nothing is assigned. When write, set "0". When read, values are indeterminate.			—	—

Note 1: The value of the flash memory control register after a reset is "--000001".

Note 2: For this bit to be set to 1, the user needs to write a 0 and then a 1 to it in succession. Use the control program except in the internal flash memory for write to this bit.

Note 3: For this bit to be set to 1, the user needs to write a 0 and then a 1 to it in succession when the CPU rewrite mode select bit = 1.

Note 4: Effective only when the CPU rewrite mode select bit = 1. Set this bit to 0 subsequently after setting it to 1 (reset).

Note 5: Use the control program except in the internal flash memory for write tothis bit.

Figure 23-1. Flash memory control register

Flash memory control register 2

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

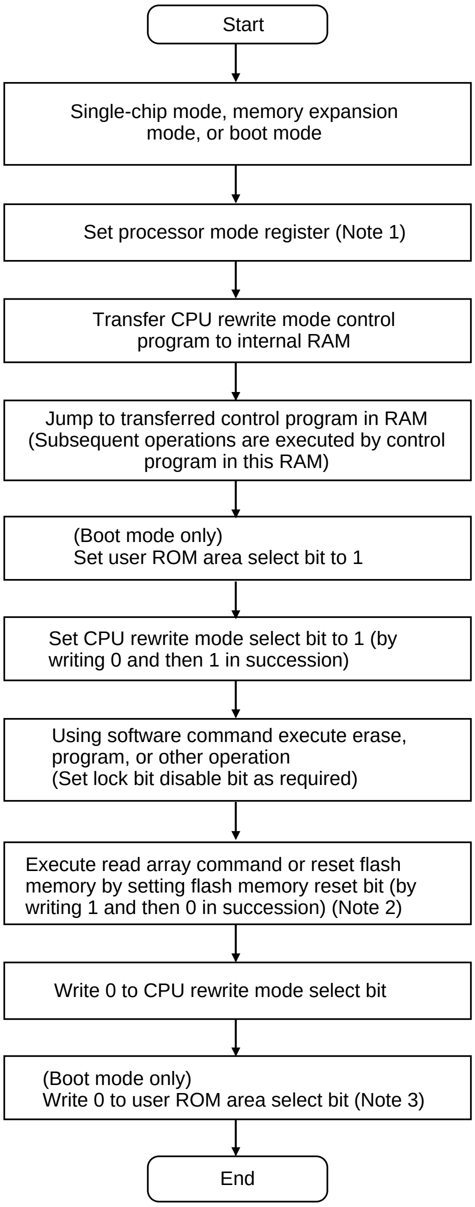
Symbol	Address	When reset
FMCR2	03B6$_{16}$	XX000001$_2$

Bit symbol	Bit name	Function	R	W
Reserved bits		Must always be set to "0".	—	○
FMCR22	Flash memory power supply-OFF bit (Note)	0: Flash memory power supply is connected. 1: Flash memory power supply-OFF	○	○
Reserved bits		Must always be set to "0".	—	○

Note: For this bit to be set to "1", the user needs to write a 0"" and then a 1 to it in succession. When this procedure is not taken, it is not enacted in "1". This is necessary to ensure that no interrupt or DMA transfer will be executed during the interval. During parallel I/O mode, programming, erase, or read of flash memory is not controlled by this bit, only by external pins.

Figure 23-2. Flash memory control register 2

Set the bit 2 of FMCR2 (address $03B6_{16}$) in order to reduce power consumption.Although setting this bit to "1" helps to reduce the device's power consumption, programs cannot be read from the internal flash memory. Make sure the operation to saet this bit to "1" and other operations to be performed while this bit remains "1" are executed in areas outside flash memory.


Start
Single-chip mode, memory expansion mode, or boot mode
Set processor mode register (Note 1)
Transfer CPU rewrite mode control program to internal RAM
Jump to transferred control program in RAM (Subsequent operations are executed by control program in this RAM)
(Boot mode only) Set user ROM area select bit to 1
Set CPU rewrite mode select bit to 1 (by writing 0 and then 1 in succession)
Using software command execute erase, program, or other operation (Set lock bit disable bit as required)
Execute read array command or reset flash memory by setting flash memory reset bit (by writing 1 and then 0 in succession) (Note 2)
Write 0 to CPU rewrite mode select bit
(Boot mode only) Write 0 to user ROM area select bit (Note 3)
End

Note 1: Set bit 7 (internal ROM access wait bit) of the processor mode register 1 (address 0005_{16}) to 1 (1 wait state).
Note 2: Before exiting the CPU rewrite mode after completing erase or program operation, always be sure to execute a read array command or reset the flash memory.
Note 3: 1 can be set. However, when this bit is 1, user ROM area is accessed.

Figure 23-3 CPU rewrite mode set/reset flowchart

Precautions on CPU Rewrite Mode

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

(1) Operation speed

When in CPU rewrite mode, set the main clock frequency as shown below, using the main clock divide ratio select bit (bit 6 at address 0006_{16} and bits 6 and 7 at address 0007_{16}):
6.25 MHz or less when wait bit (bit 7 at address 0005_{16}) is 0 (without internal access wait state)
12.5 MHz or less when wait bit (bit 7 at address 0005_{16}) is 1 (with internal access wait state)

(2) Instructions inhibited

The instructions listed below cannot be used when in CPU rewrite mode, because they refer to the internal data of the flash memory:
UND instruction, INTO instruction, JMPS instruction, JSRS instruction and BRK instruction (when using fixed vector table only)

(3) Interrupts inhibited

the NMI interrupt and address match interrupt cannot be used in CPU rewrite mode because they refer to the internal flash memory. If interrupts have their in the INTB register, they can be used by transferring the vector into the RAM area. The WDT interrupt can be used because the operation mode is forcibly changed to normal mode when the interrupt is generated. Since the rewrite operation is halted when the WDT interrupt occurs, the erase/program operation needs to be performed over again.

(4) Internal reserved expansion bit (bit 3 at address 0005_{16})

The reserved area of the internal memory can be changed by using the internal reserved expansion bit (bit 3 at address 0005_{16}). However, if the CPU rewrite mode select bit (bit 1 at address $03B7_{16}$) is set to 1, the internal reserved expansion bit (bit 3 at address 0005_{16}) is also set to 1 automatically. Similarly, if the CPU rewrite mode select bit (bit 1 at address $03B7_{16}$) is set to 0, the internal reserved bit (bit 3 at address 0005_{16}) also is set to 0 automatically.

(5) Reset

Reset input is always accepted. After a reset, the address $0C0000_{16}$ through $0CFFFF_{16}$ are made a reserved area and cannot be accessed. Therefore, if your product has this area in the user ROM area, do not write any address of this area into the reset vector. This area is made accessible by changing the internal reserved expansion bit (bit 3 at address 0005_{16}) in a program.

Software Commands

Table 23-1 lists the software commands available with the M16C/6NT (with on-chip flash memory). After setting the CPU rewrite mode select bit to 1, write a software command to specify an erase or program operation. Note that when entering a software command, the upper byte (D_8 to D_{15}) is ignored.

The content of each software command is explained below.

Table 23-1. List of software commands (CPU rewrite mode)

Command	First bus cycle			Second bus cycle			Third bus cycle		
	Mode	Address	Data (D_0 to D_7)	Mode	Address	Data (D_0 to D_7)	Mode	Address	Data (D_0 to D_7)
Read array	Write	X (Note 6)	FF_{16}						
Read status register	Write	X	70_{16}	Read	X	SRD (Note 2)			
Clear status register	Write	X	50_{16}						
Page program (Note 3)	Write	X	41_{16}	Write	WA0 (Note 3)	WD0 (Note 3)	Write	WA1	WD1
Block erase	Write	X	20_{16}	Write	BA (Note 4)	$D0_{16}$			
Erase all unlock block	Write	X	$A7_{16}$	Write	X	$D0_{16}$			
Lock bit program	Write	X	77_{16}	Write	BA	$D0_{16}$			
Read lock bit status	Write	X	71_{16}	Read	BA	D_6 (Note 5)			

Note 1: When a software command is input, the high-order byte of data (D_8 to D_{15}) is ignored.
Note 2: SRD = Status Register Data
Note 3: WA = Write Address, WD = Write Data
WA and WD must be set sequentially from 00_{16} to FE_{16} (byte address; however, an even address). The page size is 256 bytes.
Note 4: BA = Block Address (Enter the maximum address of each block that is an even address.)
Note 5: D_6 corresponds to the block lock status. Block not locked when D_6 = 1, block locked when D_6 = 0.
Note 6: X denotes a given address in the user ROM area (that is an even address).

Read Array Command (FF_{16})

The read array mode is entered by writing the command code "FF_{16}" in the first bus cycle. When an even address to read is input in one of the bus cycles that follow, the content of the specified address is read out at the data bus (D_0 to D_{15}), 16 bits at a time.

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

Read Status Register Command (70_{16})

When the command code "70_{16}" is written in the first bus cycle, the content of the status register is read out at the data bus (D_0 to D_7) by a read in the second bus cycle.

The status register is explained in the next section.

Clear Status Register Command (50_{16})

This command is used to clear the bits SR3 to 5 of the status register after they are set. These bits indicate that operation has ended in an error. To use this command, write the command code "50_{16}" in the first bus cycle.

Page Program Command (41_{16})

Page program allows for high-speed programming in units of 256 bytes. Page program operation starts when the command code "41_{16}" is written in the first bus cycle. In the second cycle through the 129th bus cycle, the write data are sequentially written 16 bits at a time. At this time, the addresses A_0 to A_7 need to be increased by 2 from "00_{16}" to "FE_{16}". When the system finishes loading the data it starts an auto write operation (data program and verify operation).

Whether or not the auto write operation is completed can be confirmed by reading the status register or the flash memory control register. At the same time the auto write operation starts, the read status register mode is automatically entered, so the content of the status register can be read out. The status register bit 7 (SR7) is set to 0 at the same time the auto write operation starts and is returned to 1 upon completion of the auto write operation. In this case, the read status register mode remains active until the Read Array command (FF_{16}) or Read Lock Bit Status command (71_{16}) is written or the flash memory is reset using its reset bit.

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

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

Figure 23-3 shows an example of a page program flowchart.

Each block of the flash memory can be write protected by using a lock bit. For details, refer to the section where the data protect function is detailed.

Additional writes in the already programmed pages are prohibited.

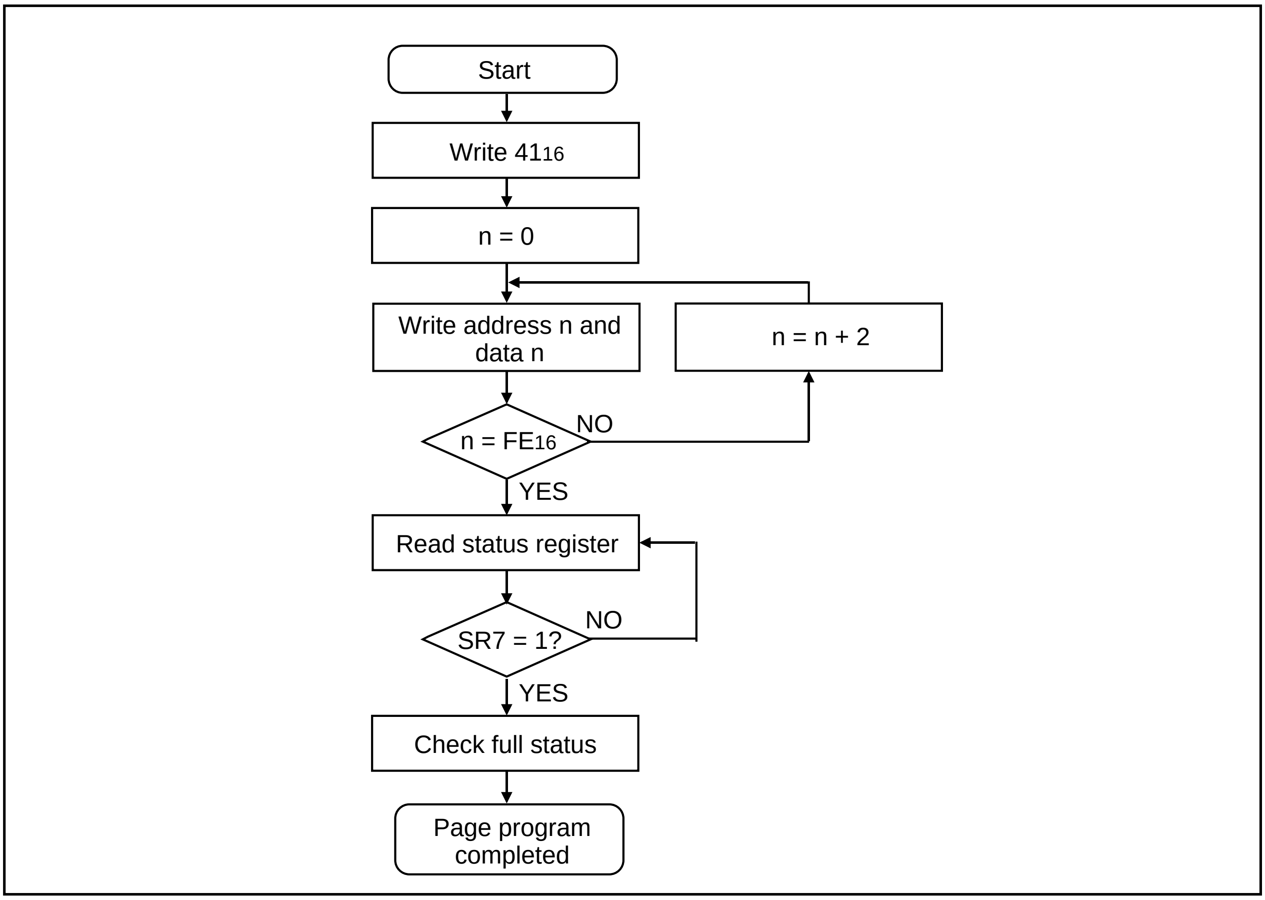


Figure 23-4. Page program flowchart

Block Erase Command (41_{16})

By writing the command code "20_{16}" in the first bus cycle, the confirmation command code "$D0_{16}$" and the block address of a flash memory block in the second bus cycle that follows, the system initiates an auto erase (erase and erase verify) operation.

Whether or not the auto erase operation is completed can be confirmed by reading the status register or the flash memory control register. At the same time the auto erase operation starts, the read status register mode is automatically entered, so the content of the status register can be read out. The status register bit 7 (SR7) is set to 0 at the same time the auto erase operation starts and is returned to 1 upon completion of the erase operation. In this case, the read status register mode remains active until the Read Array command (FF_{16}) or Read Lock Bit Status command (71_{16}) is written or the flash memory is reset using its reset bit.

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

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

Figure 23-4 shows an example of a block erase flowchart.

Each block of the flash memory can be protected against erasure by using a lock bit. For details, refer to the section where the data protect function is detailed.

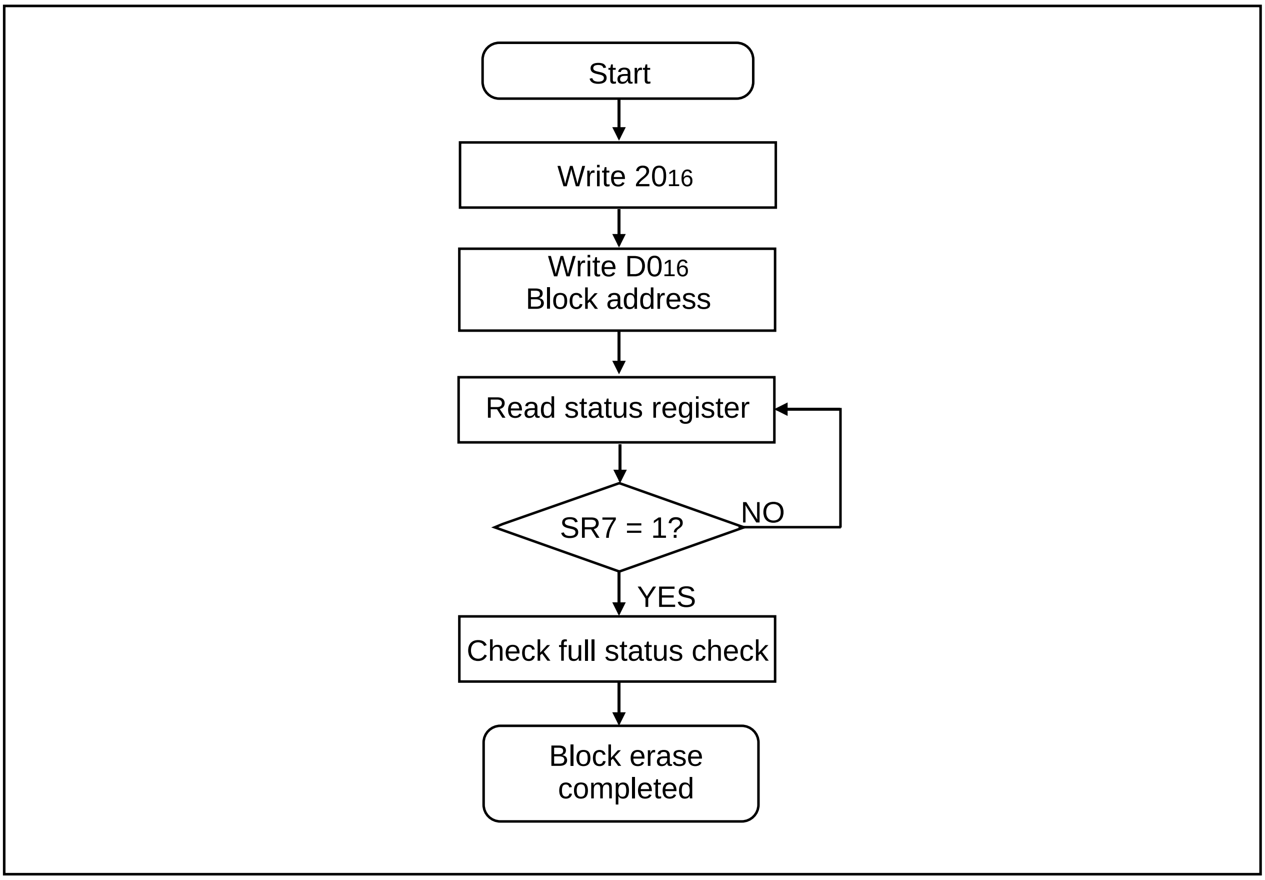


Figure 23-5. Block erase flowchart

Erase All Unlock Blocks Command (A7$_{16}$/D0$_{16}$)

By writing the command code "A7$_{16}$" in the first bus cycle and the confirmation command code "D0$_{16}$" in the second bus cycle that follows, the system starts erasing blocks successively.

Whether or not the erase all unlock blocks command is terminated can be confirmed by reading the status register or the flash memory control register, in the same way as for block erase. Also, the status register can be read out to know the result of the auto erase operation.

When the lock bit disable bit of the flash memory control register is 1, all blocks are erased no matter how the lock bit is set. On the other hand, when the lock bit disable bit is 0, the function of the lock bit is effective and only nonlocked blocks (when lock bit data is 1) are erased.

Lock Bit Program Command (77$_{16}$/D0$_{16}$)

By writing the command code "77$_{16}$" in the first bus cycle, the confirmation command code "D0$_{16}$" and the block address of a flash memory block in the second bus cycle that follows, the system sets the lock bit for the specified block to 0 (locked).

Figure 23-5 shows an example of a lock bit program flowchart. The status of the lock bit (lock bit data) can be read out by a read lock bit status command.

Whether or not the lock bit program command is terminated can be confirmed by reading the status register or the flash memory control register, in the same way as for page program.

For details about the function of the lock bit and how to reset the lock bit, refer to the section where the data protect function is detailed.

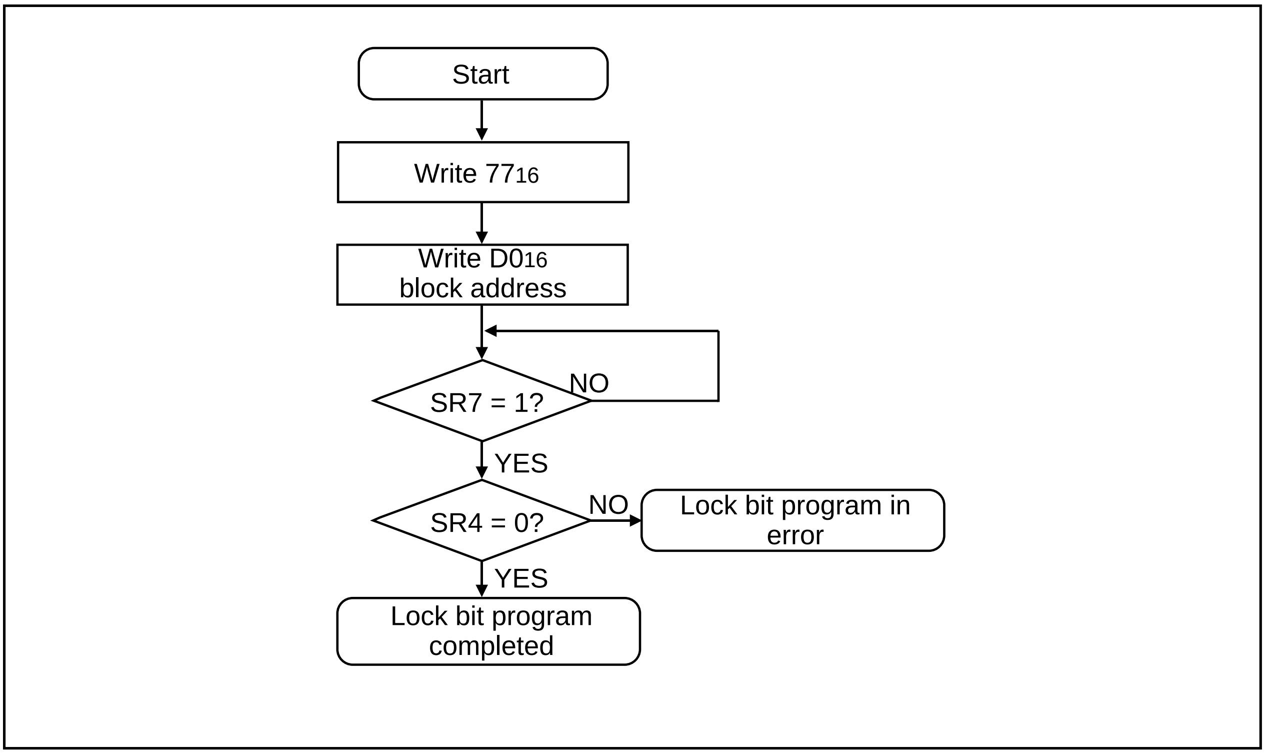


Figure 23-6. Lock bit program flowchart

Read Lock Bit Status Command (71_{16})

By writing the command code "71_{16}" in the first bus cycle and then the block address of a flash memory block in the second bus cycle that follows, the system reads out the status of the lock bit of the specified block on to the data (D6).

Figure 23-6 shows an example of a read lock bit program flowchart.

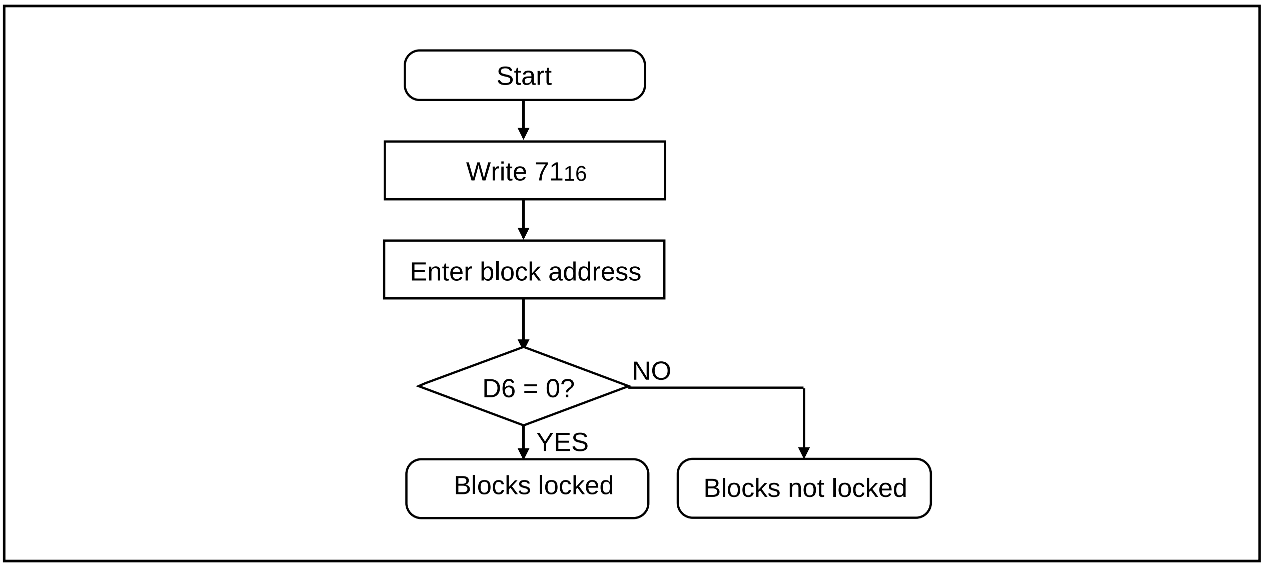


Figure 23-7. Read lock bit status flowchart

Data Protect Function (Block Lock)

Each block in figure 23-1 has a nonvolatile lock bit to specify that the block should be protected (locked) against erase/write. The lock bit program command is used to set the lock bit to 0 (locked). The lock bit of each block can be read out using the read lock bit status command.

Whether block lock is enabled or disabled is determined by the status of the lock bit and how the flash memory control register's lock bit disable bit is set.

(1) When the lock bit disable bit is 0, a specified block can be locked or unlocked by the lock bit status (lock bit data). Blocks whose lock bit data are 0 are locked, so they are disabled against erase/write. On the other hand, the blocks whose lock bit data are 1 are not locked, so they are enabled for erase/write.

(2) When the lock bit disable bit is 1, all blocks are nonlocked regardless of the lock bit data, so they are enabled for erase/write. In this case, the lock bit data that are 0 (locked) are set to 1 (nonlocked) after erasure, so the lock bit-actuated lock is removed.

Status Register

The status register indicates the operating status of the flash memory and whether an erase- or a program operation has terminated normally or in error. The content of this register can be read out only by writing the read status register command (70_{16}). Table 23-2 details the status register.

The status register is cleared by writing the Clear Status Register command (50_{16}).

After a reset, the status register is set to "80_{16}".

Each bit in this register is explained below.

Write state machine (WSM) status (SR7)

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

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

Erase status (SR5)

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

The erase status is reset to 0 when cleared.

Program status (SR4)

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

The program status is reset to 0 when cleared.

When an erase command is in error (which occurs if the command entered after the block erase command (20_{16}) is not the confirmation command ($D0_{16}$)), both the program status and erase status (SR5) are set to 1.

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

Also, in one of the following cases, both SR4 and SR5 are set to 1 (command sequence error):

(1) When the valid command is not entered correctly.

(2) When the data entered in the second bus cycle of the lock bit program (77_{16}/$D0_{16}$), block erase (20_{16}/$D0_{16}$), or erase all unlock blocks ($A7_{16}$/$D0_{16}$) is not the $D0_{16}$ or FF_{16}. However, if FF_{16} is entered, read array is assumed and the command that has been set up in the first bus cycle is cancelled.

Block status after program

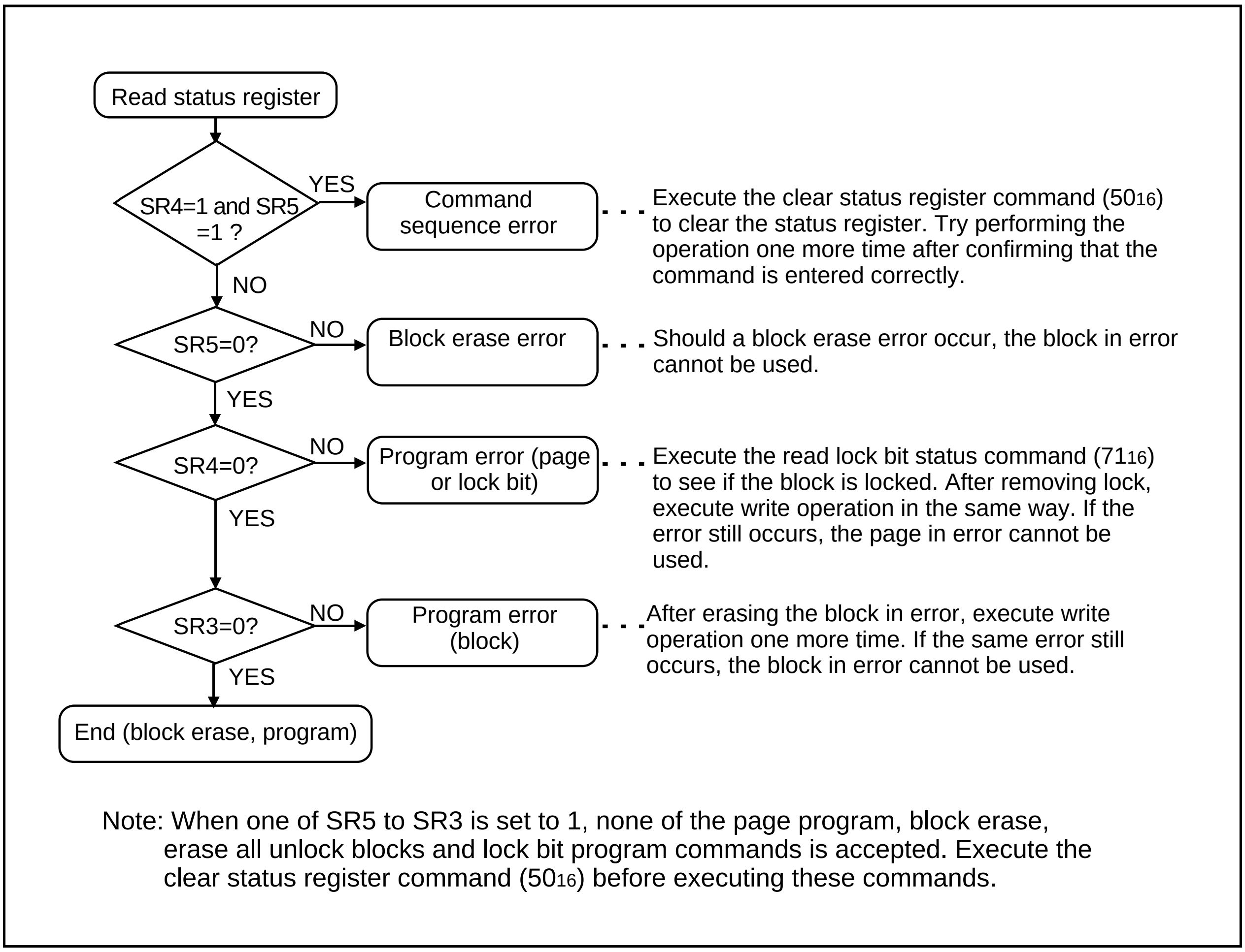
If excessive data are written (phenomenon whereby the memory cell becomes depressed which results in data not being read correctly), "1" is set for the program status after-program at the end of the page write operation. In other words, when writing ends successfully, "80_{16}" is output; when writing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

Table 23-2. Definition of each bit in status register

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

Full Status Check

By performing full status check, it is possible to know the execution results of erase- and program operations. Figure 23-7 shows a full status check flowchart and the action to take when each error occurs.


Read status register
SR4=1 and SR5 =1 ?
YES
Command sequence error
NO
SR5=0?
NO
Block erase error
YES
SR4=0?
NO
Program error (page or lock bit)
YES
SR3=0?
NO
Program error (block)
YES
End (block erase, program)
Execute the clear status register command (50_{16}) to clear the status register. Try performing the operation one more time after confirming that the command is entered correctly.
Should a block erase error occur, the block in error cannot be used.
Execute the read lock bit status command (71_{16}) to see if the block is locked. After removing lock, execute write operation in the same way. If the error still occurs, the page in error cannot be used.
After erasing the block in error, execute write operation one more time. If the same error still occurs, the block in error cannot be used.

Note: When one of SR5 to SR3 is set to 1, none of the page program, block erase, erase all unlock blocks and lock bit program commands is accepted. Execute the clear status register command (50_{16}) before executing these commands.

Figure 23-8. Full status check flowchart and reemedial procedure for errors

Functions To Inhibit Rewriting On-chip Flash Memory

To prevent the contents of the on-chip flash memory from being read out or rewritten easily, the device incorporates a ROM code protect function for use in parallel I/O mode and an ID code check function for use in standard serial I/O mode.

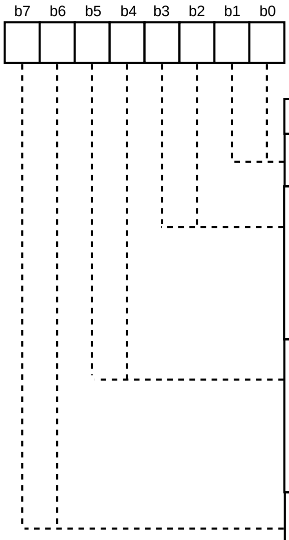
ROM Code Protect Function

The ROM code protect function reading out or modifying the contents of the on-chip flash memory by using the ROM code protect address ($0FFFFF_{16}$) when in parallel I/O mode. Figure 23-8 shows the ROM code protect control address ($0FFFFF_{16}$). (This address exists in the user ROM area.)

If one of the pair of ROM code protect bits is set to 0, ROM code protect is turned on, so that the contents of the on-chip flash memory are protected against readout and modification. ROM code protect is implemented in two levels. If level 2 is selected, the flash memory is protected even against readout by a shipment inspection LSI tester, etc. When an attempt is made to select both level 1 and level 2, level 2 is selected by default.

If both of the two ROM code protect reset bits are set to "00", ROM code protect is turned off, so that the contents of the on-chip flash memory can be read out or modified. Once ROM code protect is turned on, the contents of the ROM code protect reset bits cannot be modified in parallel I/O mode. Use the serial I/O- or some other mode to rewrite the contents of the ROM code protect reset bits.

ROM code protect control address



Symbol	Address	When reset
ROMCP	$0FFFFF_{16}$	FF_{16}

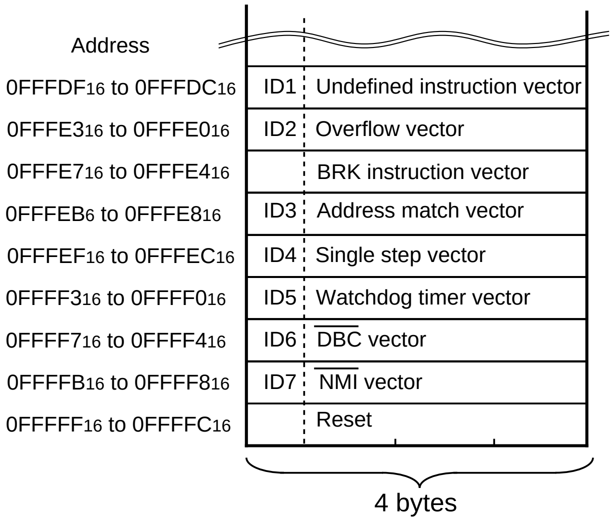
Bit symbol	Bit name	Function
Reserved bit		Always set this bit to 1.
ROMCP2	ROM code protect level 2 set bit (Note 1, 2)	b3 b2 00: Protect enabled 01: Protect enabled 10: Protect enabled 11: Protect disabled
ROMCR	ROM code protect reset bit (Note 3)	b5 b4 00: Protect removed 01: Protect set bit effective 10: Protect set bit effective 11: Protect set bit effective
ROMCP1	ROM code protect level 1 set bit (Note 1)	b7 b6 00: Protect enabled 01: Protect enabled 10: Protect enabled 11: Protect disabled

Note 1: When ROM code protect is turned on, the on-chip flash memory is protected against readout or modification in parallel input/output mode.
Note 2: When ROM code protect level 2 is turned on, ROM code readout by a shipment inspection LSI tester, etc. also is inhibited.
Note 3: The ROM code protect reset bits can be used to turn off ROM code protect level 1 and ROM code protect level 2. However, since these bits cannot be changed in parallel input/output mode, they need to be rewritten in serial input/output or some other mode.

Figure 23-9. ROM code protect control address

ID Code Check Function

Use this function in standard serial I/O mode. When the contents of the flash memory is not blank, the ID code sent from the serial programmer is compared with the ID code written in the flash memory to see if they match. If the ID codes do not match, the commands sent from the serial programmer are not accepted. The ID code consists of 8-bit data, the area of which, beginning with the first byte, are 0FFFDF$_{16}$, 0FFFE3$_{16}$, 0FFFEB$_{16}$, 0FFFEF$_{16}$, 0FFFF3$_{16}$, 0FFFF7$_{16}$ and 0FFFFB$_{16}$. Write a program which has the ID code preset at these addresses to the flash memory.



Address		
0FFFDF$_{16}$ to 0FFFDC$_{16}$	ID1	Undefined instruction vector
0FFFE3$_{16}$ to 0FFFE0$_{16}$	ID2	Overflow vector
0FFFE7$_{16}$ to 0FFFE4$_{16}$		BRK instruction vector
0FFFEB$_{6}$ to 0FFFE8$_{16}$	ID3	Address match vector
0FFFEF$_{16}$ to 0FFFEC$_{16}$	ID4	Single step vector
0FFFF3$_{16}$ to 0FFFF0$_{16}$	ID5	Watchdog timer vector
0FFFF7$_{16}$ to 0FFFF4$_{16}$	ID6	$\overline{\text{DBC}}$ vector
0FFFFB$_{16}$ to 0FFFF8$_{16}$	ID7	$\overline{\text{NMI}}$ vector
0FFFFF$_{16}$ to 0FFFFC$_{16}$		Reset

Figure 23-10. ID code store addresses

Description of Pin Function (Flash Memory Parallel I/O Mode)

Pin name	Signal name	I/O	Function
V_{CC},V_{SS}	Power supply input		Apply 3.3 ± 0.3 V to the Vcc pin (for both 5V and 3.3V versions) and 0 V to the Vss pin.
CNV_{SS}	CNV_{SS}	I	Connect this pin to V_{CC}.
$\overline{RESET}$	Reset input	I	Reset input pin. When reset is held low, more than 20 cycles of clock are required at the X_{IN} pin.
X_{IN}	Clock input	I	Connect a ceramic or crystal resonator between the X_{IN} and X_{OUT} pins. When entering an externally derived clock, enter it from X_{IN} and leave X_{OUT} open.
X_{OUT}	Clock output	O	
BYTE	BYTE	I	Connect this pin to Vcc or Vss.
AV_{CC}, AV_{SS}	Analog power supply input		Connect AVSS to Vss and AVcc to Vcc, respectively.
V_{REF}	Reference voltage input	O	Enter the reference voltage for AD from this pin.
$P0_0$ to $P0_7$	Data I/O D_0 to D_7	I/O	These are data D_0–D_7 input/output pins.
$P1_0$ to $P1_7$	Data I/O D_8 to D_{15}	I/O	These are data D_8–D_{15} input/output pins.
$P2_0$ to $P2_7$	Address input A_0 to A_6	I	These are address A_0–A_6 (word address) input pins. Address Ai in parallel input/output mode is equivalent to Ai + 1 in microcomputer mode.
$P3_0$ to $P3_7$	Address input A_7 to A_{14}	I	These are address A_7–A_{14} (word address) input pins.
$P4_0$ to $P4_2$	Address input A_{15} to A_{17}	I	These are address A_{15}–A_{17} (word address) input pins.
$P4_3$ to $P4_7$	Input port P4	I	Enter high signals to these pins.
$P5_0$	$\overline{CE}$ input	I	This is a $\overline{CE}$ input pin.
$P5_1$	$\overline{OE}$ input	I	This is a $\overline{OE}$ input pin.
$P5_2$	$\overline{WE}$ input	I	This is a $\overline{WE}$ input pin.
$P5_3$	$\overline{WP}$ input	I	This is a $\overline{WP}$ input pin.
$P5_4$	BSEL input	I	This is a BSEL input pin.
$P5_5$	EPM input	I	Enter a low signal to this pin.
$P5_6$ to $P5_7$	Input port P5	I	Enter high signals to these pins.
$P6_0$ to $P6_7$	Input port P6	I	Enter high signals to these pins.
$P7_0$ to $P7_7$	Input port P7	I	Enter high signals to these pins.
$P8_0$ to $P8_1$	Input port P8	I	Enter high signals to these pins.
$P8_2$ to $P8_4$	Input port P8	I	Enter low signals to these pins.
$P8_5$	$\overline{RP}$ input	I	This is a $\overline{RP}$ input pin.
$P8_6$ to $P8_7$	Input port P8	I	Enter high signals to these pins.
$P9_0$	Input port P9	I	Enter high or low signals to these pins or leave these pins open.
$P9_1$	RY/$\overline{BY}$ output	O	This is a RY/$\overline{BY}$ output pin.
$P9_2$ to $P9_7$	Input port P9	I	Enter high or low signals to these pins or leave these pins open.
$P10_0$ to $P10_7$	Input port P10	I	Enter high or low signals to these pins or leave these pins open.

Parallel I/O Mode

The parallel I/O mode is entered by making connections shown in Figure 24-2 and then turning the Vcc power supply (3.3 V) on. In this mode, the M16C/6N (with on-chip flash memory) operates in a manner similar to the DINOR flash memory M5M29FB800 by Mitsubishi. Note, however, that there are some differences in regard to the functions not available with the microcomputer and matters related to memory size, as shown in Table 24-1.

Only in parallel I/O mode, the M16C/6N (with on-chip flash memory), either 5V- or 3.3V version, needs to be operated with a supply voltage of 3.3 V ± 0.3 V. Table 24-2 shows pin relationship between the M16C/6N and M5M29FB800 in parallel I/O mode.

Table 24-1. Differences from the M5M29FB800

Functions not available with microcomputer	Differences in matters related to memory capacity
• Device ID code readout • Suspend/resume functions • Sleep function • Additional write function	• Flash memory capacity • Block arrangement (See Figure CC-1)
	Functions only available with microcomputer
	• Boot ROM area selection

Note: Do not apply V_{HH} (12 V) to the A9 and RP pins.

Table 24-2. Pin relationship in parallel I/O mode

	M16C/62(on-chip flash memory)	M5M29FB800
V_{CC}	V_{CC}	V_{CC}
V_{SS}	V_{SS}	V_{SS}
Address input	$P2_1$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$, $P4_1$, $P4_2$	A_0 to A_{16}
Data I/O	$P0_0$ to $P0_7$, $P1_0$ to $P1_7$	D_0 to D_{15}
$\overline{OE}$ input	$P5_1$	$\overline{OE}$
$\overline{CE}$ input	$P5_0$	$\overline{CE}$
$\overline{WP}$ input	$P5_3$	$\overline{WP}$
$\overline{WE}$ input	$P5_2$	$\overline{WE}$
$\overline{BYTE}$ input	BYTE	$\overline{BYTE}$
$\overline{RP}$ input	$\overline{NMI}$	$\overline{RP}$
RY/$\overline{BY}$ output	$P7_0$	RY/$\overline{BY}$
BSEL input (Note)	$P5_4$	—

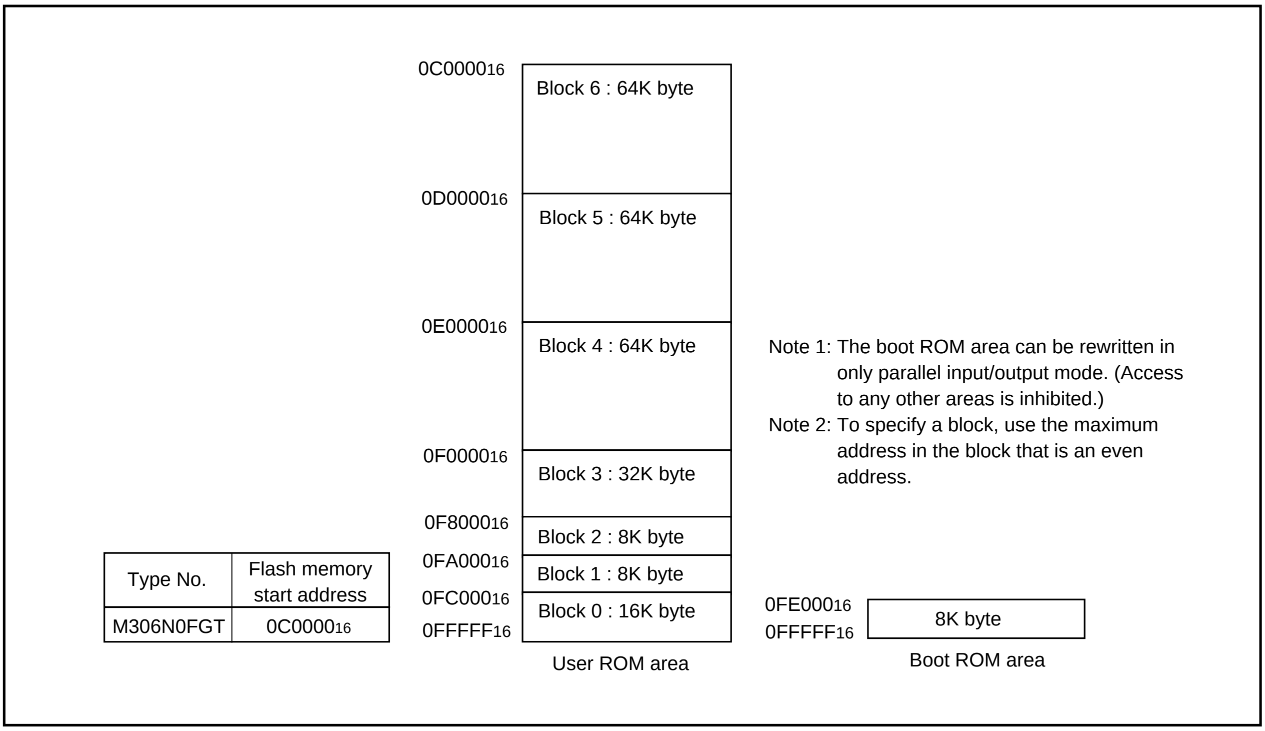
Note: BSEL is used to choose between the user ROM and boot ROM areas and has no equivalent pin in the M5M29FB800.

Address

The M16C/6N (with on-chip flash memory) has word- and byte modes which are switched over by the BYTE pin.

When the BYTE pin is high, the 16-bit data bus is selected and the memory is accessed in 16 bits. In this case, addresses must always be specified by an even address. When the BYTE pin is low, the 8-bit data bus is selected and the memory is accessed in 8 bits.

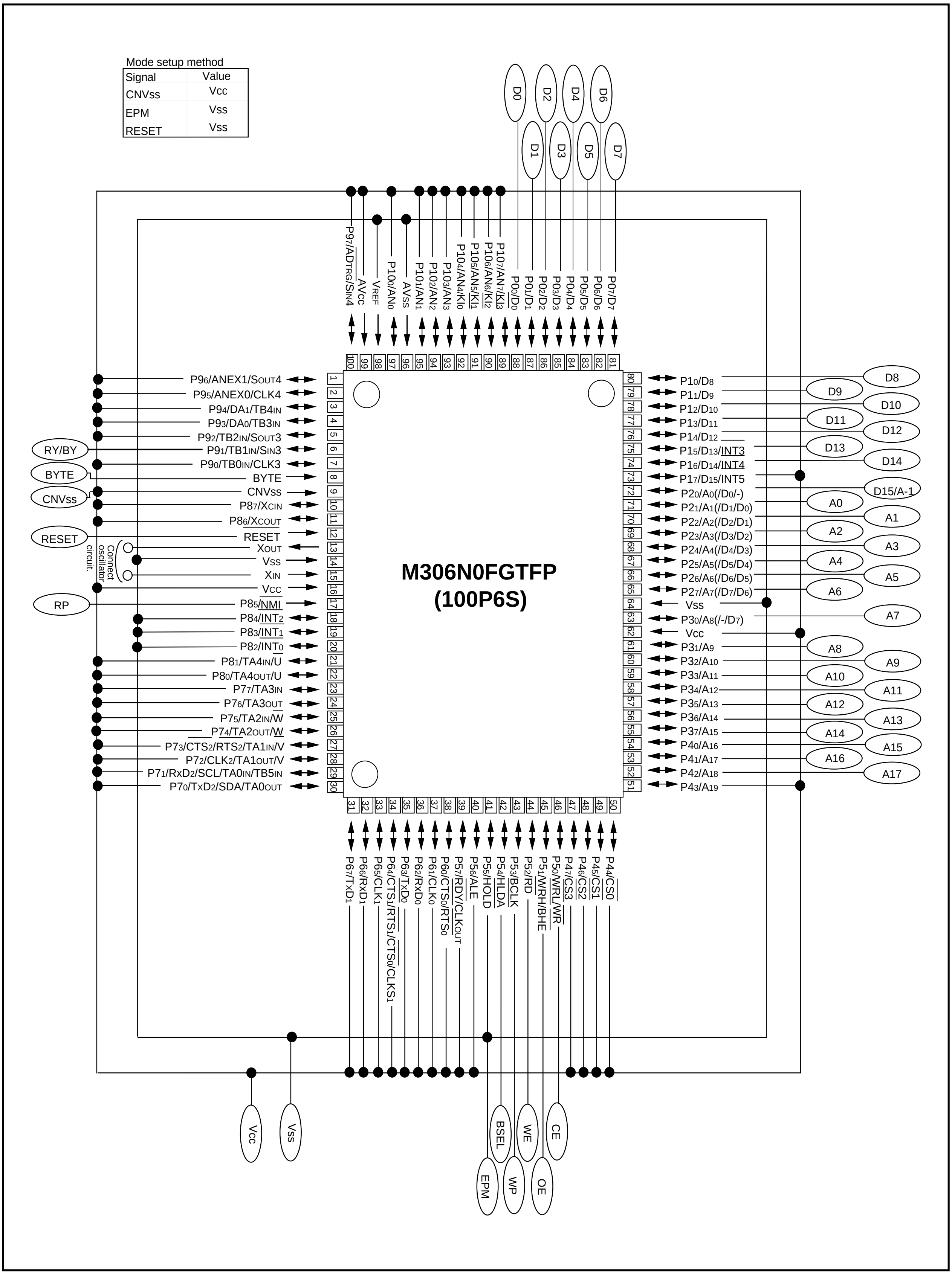
The user ROM is divided into blocks as shown in Figure 24-1. The block address referred to in this manual is the maximum even address of each block.



Type No.	Flash memory start address
M306N0FGT	0C0000$_{16}$

Note 1: The boot ROM area can be rewritten in only parallel input/output mode. (Access to any other areas is inhibited.)
Note 2: To specify a block, use the maximum address in the block that is an even address.

Figure 24-1. Block diagram of on-chip flash memory



Mode setup method

Signal	Value
CNVss	Vcc
EPM	Vss
RESET	Vss

Figure 24-2. Pin connection diagram in parallel I/O mode

User ROM and Boot ROM Areas

In parallel I/O mode, the user ROM- and boot ROM areas shown in Figure 24-1 can be rewritten.

BSEL pin is used to select between these two areas. The user ROM area is selected by pulling the BSEL input low; the boot ROM area is selected by driving the BSEL input high. Both areas of flash memory can be operated on in the same way.

Program- and block erase operations can be performed in the user ROM area. The user ROM area and its blocks are shown in Figure 24-1.

The boot ROM area is 8 Kbytes in size. In parallel I/O mode, it is located at address $0FE000_{16}$ through $0FFFFF_{16}$. Make sure program- and block erase operations are always performed within this address range. (Access to any location outside this address range is prohibited.)

In the boot ROM area, an erase block operation is applied to only one 8 Kbyte block. The boot ROM area has a standard serial I/O mode control program stored in it when shipped from the Mitsubishi factory. Therefore, if the device is going to be used in standard serial I/O mode, do not rewrite the boot ROM area.

Functional Outline (Parallel I/O Mode)

In parallel I/O mode, bus operation modes — Read, Output Disable, Standby, Write and Deep Power Down — are selected by the status of the $\overline{CE}$-, $\overline{OE}$-, $\overline{WE}$- and $\overline{RP}$ input pins.

The contents of erase-, program- and other operations are selected by writing a software command. The data, status register, etc. in memory can be read out only by a read after software command input.

Program- and erase operations are controlled using software commands.

Table 24-3. Relationship between control signals and bus operation modes

Mode \ Pin name		$\overline{CE}$	$\overline{OE}$	$\overline{WE}$	$\overline{RP}$	D_0 to D_{15}
Read	Array	V_{IL}	V_{IL}	V_{IH}	V_{IH}	Data output
	Status register	V_{IL}	V_{IL}	V_{IH}	V_{IH}	Status register data output
	Lock bit status	V_{IL}	V_{IL}	V_{IH}	V_{IH}	Lock bit data (D_6) output
Output disabled		V_{IL}	V_{IH}	V_{IH}	V_{IH}	Hi-Z
Stand by		V_{IH}	X	X	V_{IH}	Hi-Z
Write	Program	V_{IL}	V_{IH}	V_{IL}	V_{IH}	Command/data input
	Erase	V_{IL}	V_{IH}	V_{IL}	V_{IH}	Command input
	Other	V_{IL}	V_{IH}	V_{IL}	V_{IH}	Command input
Deep power down		X	X	X	V_{IL}	Hi-Z

Note: X can be V_{IL} or V_{IH}.

Bus Operation Modes

Read

The Read mode is entered by pulling the $\overline{OE}$ pin low when the $\overline{CE}$ pin is low and the $\overline{WE}$- as well as $\overline{RP}$ pins are high. There are three read modes: Array, Status Register and Lock Bit Status which are selected by software command input. In Read mode, the data corresponding to each software command entered are output from the data I/O pins D_0 – D_{15}. The Read Array mode is automatically selected when the device is powered on after it exits Deep Power Down mode.

Output Disable

The Output Disable mode is entered by pulling the $\overline{CE}$ pin low and the $\overline{WE}$-,$\overline{OE}$- and $\overline{RP}$ pins high. Also, the data I/O pins are placed in the high-impedance state.

Standby

The Standby mode is entered by driving the $\overline{CE}$ pin high when the $\overline{RP}$ pin is high. Also, the data I/O pins are placed in the high-impedance state. However, if the $\overline{CE}$ pin is set high during erase- or program operation, the internal control circuit does not halt immediately and normal power consumption is required until the operation under way is completed.

Write

The Write mode is entered by pulling the $\overline{WE}$ pin low when the $\overline{CE}$ pin is low and the $\overline{OE}$- as well as $\overline{RP}$ pins are high. In this mode, the device accepts the software commands or write data entered from the data I/O pins. A program-, erase- or some other operation is initiated depending on the content of the software command entered here. The input data such as addresses and software command are latched at the rising edge of $\overline{WE}$ or $\overline{CE}$ whichever occurs earlier.

Deep Power Down

The Deep Power Down is entered by pulling the $\overline{RP}$ pin low. Also, the data I/O pins are placed in the high-impedance state. When the device is freed from Deep Power Down mode, the Read Array mode is selected and the content of the status register is set to "80_{16}". If the $\overline{RP}$ pin is pulled low during erase- or program operation, the operation under way is cancelled and the data in the relevant block becomes invalid.

Software Commands

Table 24-4 lists the software commands available with the M16C/6NT (with on-chip flash memory). By entering a software command from the data I/O pins (D_0 – D_7) in Write mode, specify the content of the operation, such as erase- or program operation, to be performed. When entering a software command, the upper byte (D_8 – D_{15}) is ignored.

Table 24-4. Software command list (parallel I/O mode)

Command	First bus cycle			Second bus cycle			Third bus cycle		
	Mode	Address	Data (D_0 to D_7)	Mode	Address	Data (D_0 to D_7)	Mode	Address	Data (D_0 to D_7)
Read array	Write	X (Note 6)	FF_{16}						
Read status register	Write	X	70_{16}	Read	X	SRD (Note 2)			
Clear status register	Write	X	50_{16}						
Page program (Note 3)	Write	X	41_{16}	Write	WA0 (Note 3)	WD0 (Note 3)	Write	WA1	WD1
Block erase	Write	X	20_{16}	Write	BA (Note 4)	$D0_{16}$			
Erase all unlock block	Write	X	$A7_{16}$	Write	X	$D0_{16}$			
Lock bit program	Write	X	77_{16}	Write	BA	$D0_{16}$			
Read lock bit status	Write	X	71_{16}	Read	BA	D_6 (Note 5)			

Note 1: When a software command is input, the upper byte of data (D_8 to D_{15}) is ignored.
Note 2: SRD = Status Register Data
Note 3: WA = Write Address, WD = Write Data
WA and WD must be set sequentially from 00_{16} to FE_{16} (an even address). The page size is 256 bytes.
Note 4: BA = Block Address (Enter the maximum address of each block that is an even address.)
Note 5: D_6 corresponds to the block lock status. Block not locked when D_6 = 1, block locked when D_6 = 0.

Read Array Command (FF_{16})

The Read Array mode is entered by writing the command code "FF_{16}" in the first bus cycle. When an address to read is input in one of the bus cycles that follow, the content of the specified address is output from the data bus (D_0 – D_{15}). The address entered here must be an even address when the BYTE pin is high (16-bit mode).

The Read Array mode is retained intact until another command is written.

The Read Array mode is also selected automatically when the device is powered on and after it exits Deep Power Down mode.

Read Status Register Command (70_{16})

When the command code "70_{16}" is written in the first bus cycle, the content of the status register is output from the data bus **(D_0 – D_7) by a read in the second bus cycle. Since the content of the status register is updated at the falling edge of $\overline{OE}$ or $\overline{CE}$, the $\overline{OE}$- or $\overline{CE}$ signal must be asserted each time the status is read. The status register is explained in the next section.**

Clear Status Register Command (50_{16})

This command is used to clear the bits SR3 to 5 of the status register after they are set. These bits indicate that operation has ended in an error. To use this command, write the command code "50_{16}" in the first bus cycle.

Page Program Command (41_{16})

Page programming enables high-speed programming in blocks of 256 bytes. The page programming operation is started when the "41_{16}" command code is written for the first bus cycle. Write data are then written sequentially from the second bus cycle to the 129th bus cycle. In this case, when the byte pin is "H" level, the address must be odd and increased by two from "00_{16}" to "FF_{16}". When the byte pin is "L" level, the address must increase from "00_{16}" to "FF_{16}". When data loading ends, the auto write (data program and verify) operation starts.

Auto Write end can be verified by reading the status register or the status of the RY/$\overline{BY}$ signal. At the start of the auto write operation, the read status register mode is automatically engaged, so the contents of the status register can be read from the data I/O pins (D_0 – D_7). Status register bit 7 (SR7) becomes "0" when the auto write operation starts and returns to "1" when it ends. In this way, the read status register mode is maintained until the next read array command (FF_{16}) or read lock bit status command (71_{16}) is written.

Similar to the status register bit 7, the RY/$\overline{BY}$ pin is "L" level during the auto write period and becomes "H" level when auto write ends.

After the auto write operation ends, the result of the operation can be known by reading the status register. For more information, see the section on the status register.

Figure 24-3 shows a flowchart of the page program. For the operation timing of the page program, see the time chart in the section on electric characteristics.

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

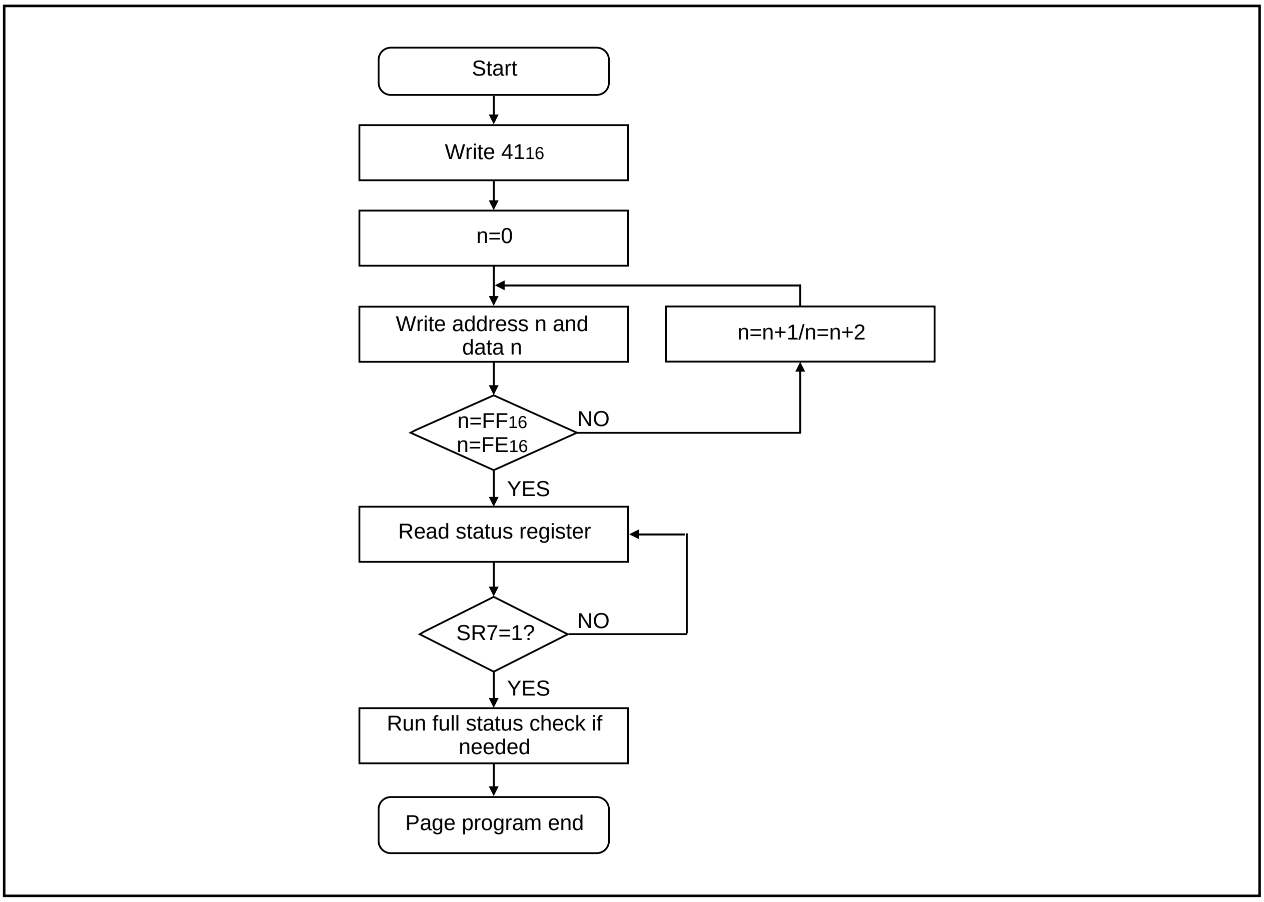


Figure 24-3. Page program flowchart

Block Erase Command (20_{16}/$D0_{16}$)

Writing the "20_{16}" command code for the first bus cycle and, after that, the "$D0_{16}$" verify command code and the block address of a block for the second bus cycle starts the auto erase (erase and erase verify) operation for the specified block.

Auto erase end can verified by reading the status register or the status of the RY/$\overline{\text{BY}}$ signal. At the start of the auto erase operation, the read status register mode is automatically engaged, so the contents of the status register can be read from the data I/O pins (D_0 – D_7). Status register bit 7 (SR7) becomes "0" when the auto erase operation starts and returns to "1" when it ends. In this way, the read status register mode is maintained until the next read array command (FF_{16}) or read lock bit status command (71_{16}) is written. Similar to the status register bit 7, the RY/$\overline{\text{BY}}$ pin is "L" level during auto erase operations and becomes "H" level when auto erase ends.

After the block erase operation ends, the result of the operation can be known by reading the status register. For more information, see the section on the status register.

Figure 24-4 shows a flowchart of block erasing. For the block erase operation timing, see time chart in the section on electric characteristics.

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

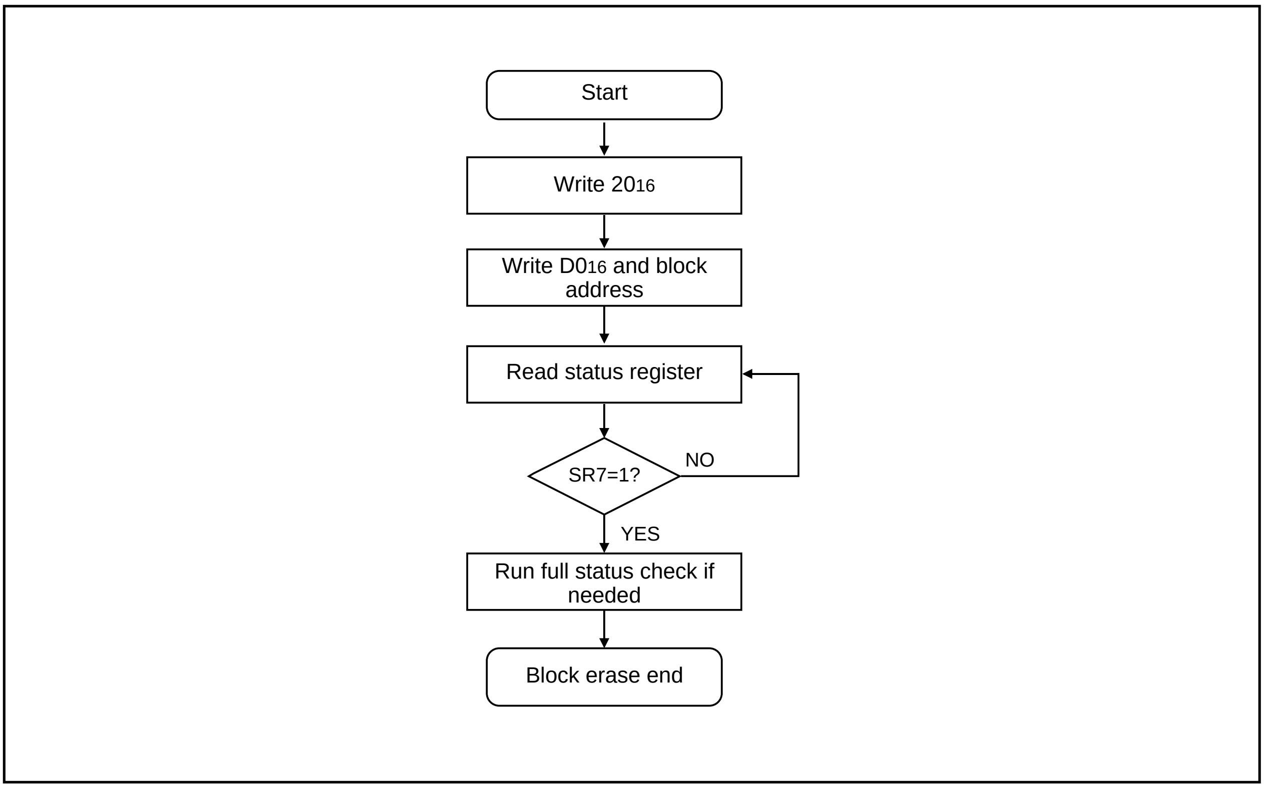


Figure 24-4. Block erase flowchart

Erase All Unlocked Blocks Command (A7$_{16}$/D0$_{16}$)

Writing the "A7$_{16}$" command code for the first bus cycle and the "D0$_{16}$" verify command code for the second bus cycle continuously executes the block erase operation for all the blocks. In this case, it is not necessary to specify an address in the second bus cycle.

Even after the erase all unlock blocks operation ends, as with block erase, the end of the operation can be verified by reading the status register or the status of the RY/$\overline{\text{BY}}$ signal. Also, the result of the erase operation can be known by reading the status register.

When the $\overline{\text{WP}}$ pin is "H" level, all blocks are erased regardless of lock bit status. When the $\overline{\text{WP}}$ pin is "L" level, The lock pin is enabled and only unlocked blocks (lock bit data are "1") are erased.

Lock Bit Program Command (77$_{16}$/D0$_{16}$)

Writing the "77$_{16}$" command code for the first bus cycle and, after that, the "D0$_{16}$" verify command code as well as the block address of a block for the second bus cycle writes "0" (lock) for the lock bit of the specified block.

Figure 24-5 shows an example of flowchart of the lock bit program. The lock bit status (lock bit data) can be read with the read lock bit status command.

As with the page program, the end of the lock bit program auto write operation can be verified by reading the status register or the status of the RY/$\overline{\text{BY}}$ signal.

For information on the lock bit function, reset procedure and so on, see the section on the data protection function.

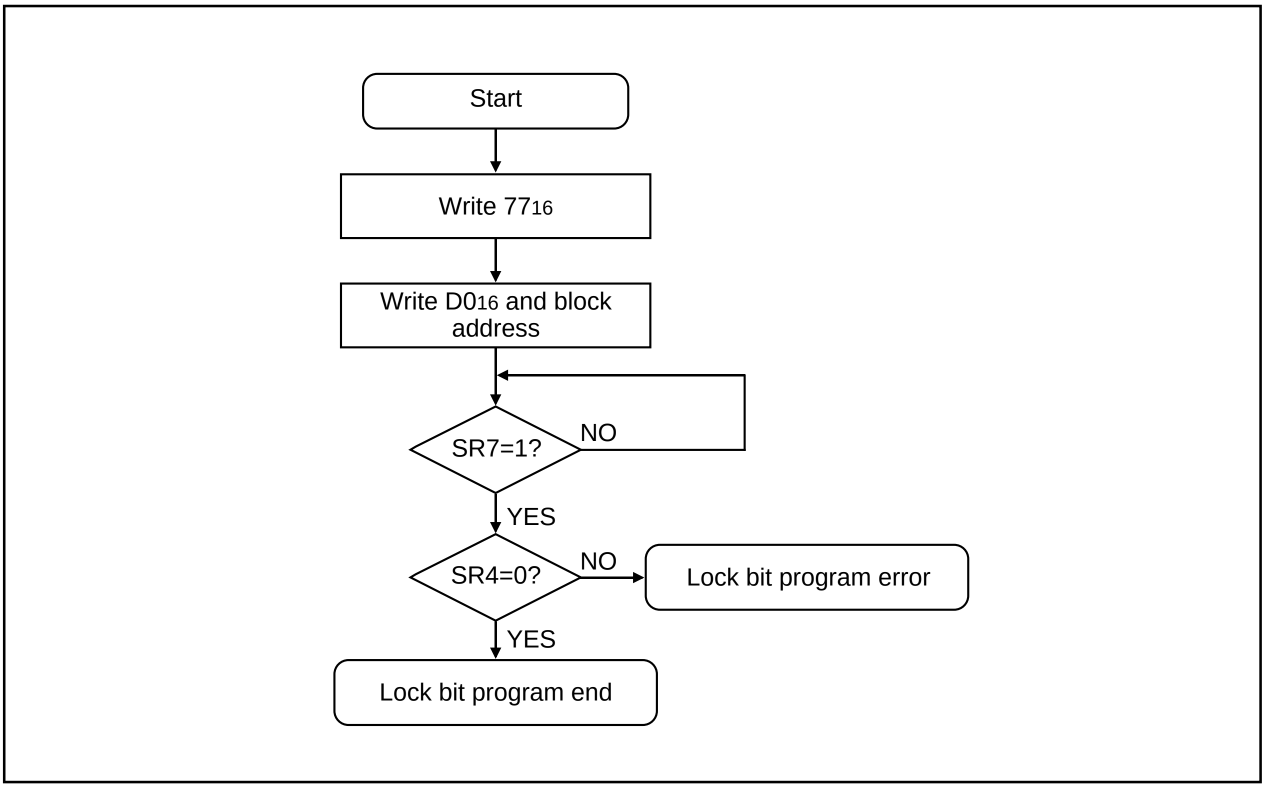
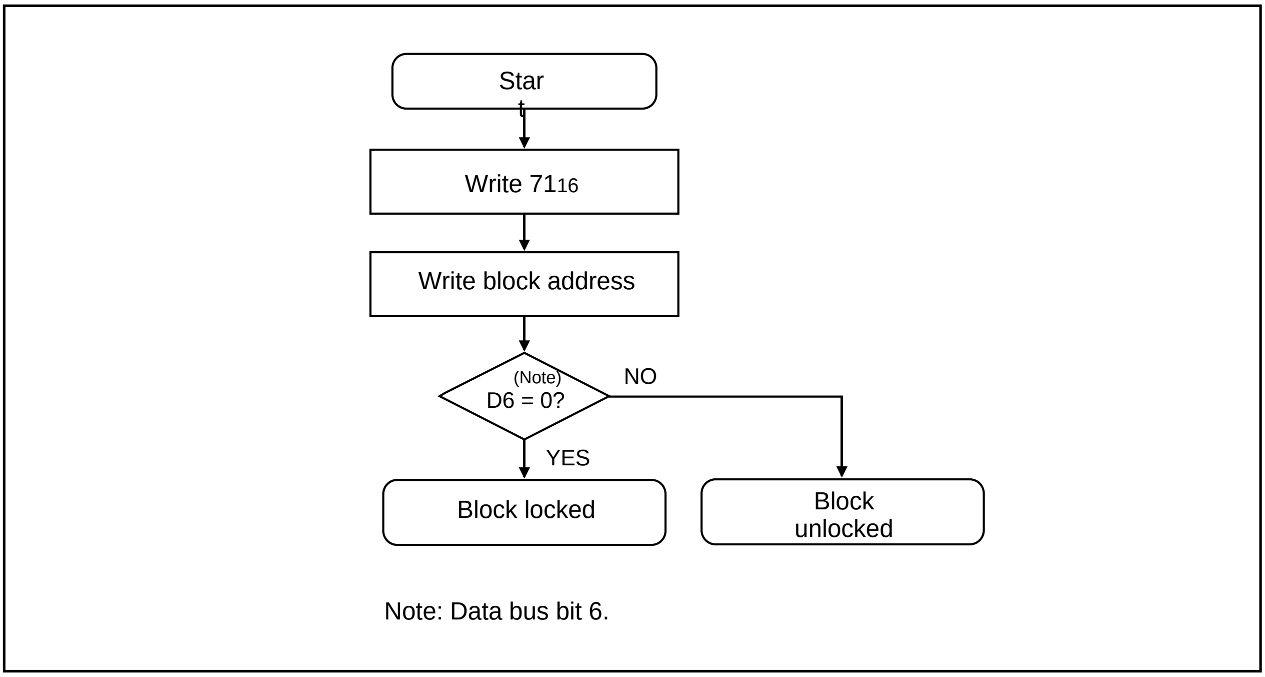


Figure 24-5. Lock bit program flowchart

Read Lock Bit Status Command (71_{16})

After the "71_{16}" command code is written for the first bus cycle and the address block of a given block is specified in the second bus cycle, the lock status of the specified block is output as data I/O pin bit 6 (D6). Figure 24-6 shows an example of flowchart of the read lock bit status.



Note: Data bus bit 6.

Figure 24-5. Lock bit program flowchart

Data Protection Function (Block Lock)

Each of the blocks in Figure 24-1 has a nonvolatile lock bit that specifies protection (block lock) against erasing/writing. A block is locked (writing "0" for the lock bit) with the lock bit program command. Also, the lock bit of any block can be read with the read lock bit status command.

Block lock enable/disable is determined by the status of the lock bit itself and the status of the $\overline{\text{RP}}$- and $\overline{\text{WP}}$ pins. This relationship is given in Table 24-5.

(1) When the $\overline{\text{RP}}$ pin is "L" level, the deep power down mode is engaged and all blocks are locked.

(2) When the $\overline{\text{RP}}$ pin is "H" level and the $\overline{\text{WP}}$ pin "L" level, the specified block can be locked/unlocked using the lock bit (lock bit data). Blocks with "0" lock bit data are locked and cannot be erased or written in. On the other hand, blocks with "1" lock bit data are unlocked and can be erased or written in.

(3) When the $\overline{\text{RP}}$ pin and the $\overline{\text{WP}}$ pin are both "H" level, all blocks are unlocked regardless of lock bit data status and can be erased or written in. In this case, lock bit data that were "0" before the block was erased are set to "1" (unlocked) after erasing, therefore the block is actually unlocked with the lock bit.

Table 24-5. Block lock conditions

$\overline{\text{RP}}$	$\overline{\text{WP}}$ (Note 1)	Lock bit (Internal)	Block lock
V_{IL}	X	X	Locks all blocks (Deep power down mode)
V_{IH}	V_{IL}	0	Locks block using lock bit data
V_{IH}	V_{IL}	1	Unlocks block using lock bit data
V_{IH}	V_{IH}	X	Unlocks all blocks (Note 2)

Note 1: During read/write operations or when the write state machine (WSM) status is busy (SR7 = "0"), do not switch $\overline{\text{WP}}$ pin state.

Note 2: In this case, the lock bit is set to "1" after the block is erased.

Status Register

The status register indicates status such as whether an erase operation or a program ended successfully or in error. It can be read under the following conditions.

(1) In the read array mode or lock bit status mode, when the read status register command (70_{16}) is written and the block address is subsequently read.

(2) In the period from when the page program auto write or auto erase starts to when the read array command (FF_{16}) or the read lock bit status command (71_{16}) is input.

The status register is cleared in the following situations.

(1) When the clear status register command (50_{16}) is written

(2) When in the deep power down mode

(3) When power is turned off

Table 24-6 gives the definition of each status register bit. When power is turned on or returning from the deep power down mode, the status register outputs "80_{16}".

Table 24-6. Status register

Symbol	Status	Definition	
		"1"	"0"
SR7 (D_7)	Write state machine (WSM) status	Ready	Busy
SR6 (D_6)	Reserved	—	—
SR5 (D_5)	Erase status	Ended in error	Ended successfully
SR4 (D_4)	Program status	Ended in error	Ended successfully
SR3 (D_3)	Program status after-program	Ended in error	Ended successfully
SR2 (D_2)	Reserved	—	—
SR1 (D_1)	Reserved	—	—
SR0 (D_0)	Reserved	—	—

Write State Machine (WSM) Status (SR7)

The write state machine (WSM) status indicates the operating status of the flash memory. When power is turned on or returning from deep power down mode, it is set to "1". This bit is "0" (busy) during the auto write- or erase operation and becomes "1" when the operation ends.

Erase Status (SR5)

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

Program Status (SR4)

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

Block Status After Program (SR3)

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

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

Also in the following cases, both SR4 and SR5 are set to "1" (command sequence error).

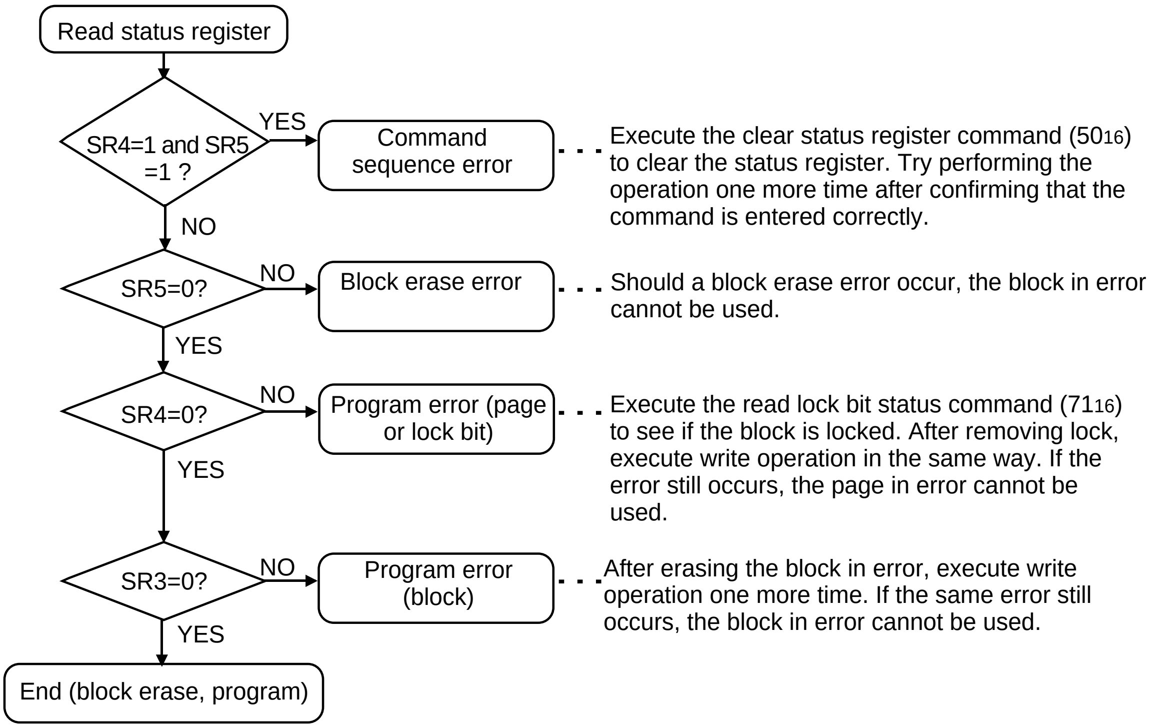
(1) If data other than "$D0_{16}$" or "FF_{16}" are input for the second bus cycle data of the lock bit program command (77_{16}/$D0_{16}$).

(2) If data other than "$D0_{16}$" or "FF_{16}" are input for the second bus cycle data of the block erase command (20_{16}/$D0_{16}$).

(3) If data other than "$D0_{16}$" or "FF_{16}" are input for the second bus cycle data of the erase all unlocked blocks command ($A7_{16}$/$D0_{16}$). However, inputting "FF_{16}" engages the read array mode and cancels the setup command in the first bus cycle.

Full Status Check

Results of executed erase- and program operations can be known by running a full status check. Figure 24-7 shows a flowchart of the full status check and explains how to remedy errors which may occur.



. . . Execute the clear status register command (50_{16}) to clear the status register. Try performing the operation one more time after confirming that the command is entered correctly.

. . . Should a block erase error occur, the block in error cannot be used.

. . . Execute the read lock bit status command (71_{16}) to see if the block is locked. After removing lock, execute write operation in the same way. If the error still occurs, the page in error cannot be used.

. . . After erasing the block in error, execute write operation one more time. If the same error still occurs, the block in error cannot be used.

Note: When one of SR5 to SR3 is set to 1, none of the page program, block erase, erase all unlock blocks and lock bit program commands is accepted. Execute the clear status register command (50_{16}) before executing these commands.

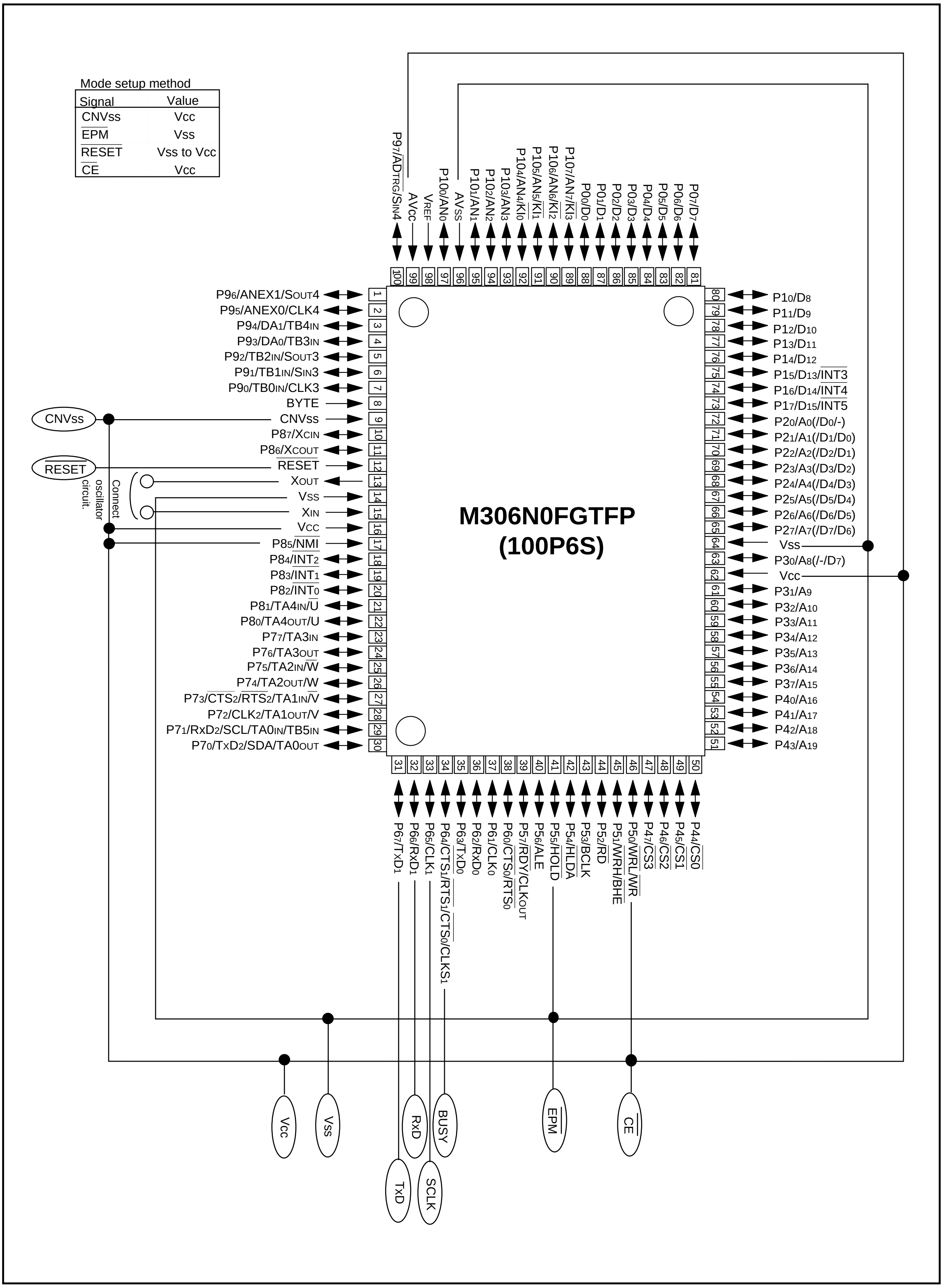
Figure 24-7. Full status flowchart and remedial procedure for errors

Ready/Busy (RY/$\overline{\text{BY}}$) pin

The RY/$\overline{\text{BY}}$ pin is an output pin which, like the write state machine (WSM) status (SR7), indicates the operating status of the flash memory. It is "L" level during the auto write- or the auto erase operation and becomes to the high-impedance state (ready state) when the operation ends. The RY/$\overline{\text{BY}}$ pin requires an external pull-up.

Pin functions (flash memory standard serial I/O mode)

Pin	DescriptionName	I/O	
VCC,VSS	Power input		Apply program/erase protection voltage to Vcc pin and 0 V to Vss pin.
CNVSS	CNVSS	I	Connect to Vcc pin.
$\overline{\text{RESET}}$	Reset input	I	Reset input pin. While reset is "L" level, a 20 cycle or longer clock must be input to XIN pin.
XIN	Clock input	I	Connect a ceramic resonator or crystal oscillator between XIN and XOUT pins. To input an externally generated clock, input it to XIN pin and open XOUT pin.
XOUT	Clock output	O	
BYTE	BYTE	I	Connect this pin to Vcc or Vss.
AVCC, AVSS	Analog power supply input		Connect AVSS to Vss and AVcc to Vcc, respectively.
VREF	Reference voltage input	O	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 P54, P56, P57	Input port P5	I	Input "H" or "L" level signal or open.
P50	CE input	I	Input "H" level signal.
P55	EPM input	I	Input "L" level signal.
P60 to P63	Input port P6	I	Input "H" or "L" level signal or open.
P64	BUSY output	O	BUSY signal output pin
P65	SCLK input	I	Serial clock input pin
P66	RxD input	I	Serial data input pin
P67	TxD output	O	Serial data output pin
P70 to P77	Input port P7	I	Input "H" or "L" level signal or open.
P80 to P87	Input port P8	I	Input "H" or "L" level signal or open.
P9 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.


Mode setup method

Signal	Value
CNVss	Vcc
EPM	Vss
RESET	Vss to Vcc
CE	Vcc

Figure 25-1. Pin connections for serial I/O mode

Standard Serial I/O Mode

The standard serial I/O mode serially inputs and outputs the software commands, addresses and data necessary for operating (read, program, erase, etc.) the internal flash memory. It uses a purpose-specific serial programmer.
The standard serial I/O mode differs from the parallel I/O mode in that the CPU controls operations like rewriting (uses the CPU rewrite mode) in the flash memory or serial input for rewriting data. The standard serial mode is started by clearing the reset with an "H" level signal at the $P5_0$ (CE) pin, an "L" signal at the $P5_5$ (EPM) pin and an "H" level at the CNVss pin. (For the normal microprocessor mode, set CNVss to "L".)
This control program is written in the boot ROM area when shipped from Mitsubishi Electric. Therefore, if the boot ROM area is rewritten in the parallel I/O mode, the standard serial I/O mode cannot be used.
Figure 25-1 shows the pin connections for the standard serial I/O mode. Serial data I/O uses four UART1 pins: CLK1, RXD1, TXD1 and RTS1 (BUSY).
The CLK1 pin is the transfer clock input pin and it transfers the external transfer clock. The TXD1 pin outputs the CMOS signal. The RTS1 (BUSY) pin outputs an "L" level when reception setup ends and an "H" level when the reception operation starts. Transmission- and reception data are transferred serially in 8-byte blocks.
In the standard serial I/O mode, only the user ROM area shown in 24-1 can be rewritten, the boot ROM area cannot.
The standard serial I/O mode has a 7-byte ID code. When the flash memory is not blank and the ID code does not match the content of the flash memory, the command sent from the programmer is not accepted.

Function Overview (Standard Serial I/O Mode)

In the standard serial I/O mode, software commands, addresses and data are input and output between the flash memory and an external device (serial programmer, etc.) using a 4-wire clock-synchronized serial I/O (UART1). In reception, the software commands, addresses and program data are synchronized with the rise of the transfer clock input to the CLK1 pin and input into the flash memory via the RXD1 pin. In transmission, the read data and status are synchronized with the fall of the transfer clock and output to the outside from the TXD1 pin.
The TXD1 pin is CMOS output. Transmission is in 8-bit blocks and LSB first.
When busy, either during transmission or reception, or while executing an erase operation or program the RTS1 (BUSY) pin is "H" level. Accordingly, do not start the next transmission until the RTS1 (BUSY) pin is "L" level.
Also, data in memory and the status register can be read after inputting a software command. It is possible to check flash memory operating status or whether a program- or erase operation ended successfully or in error by reading the status register.
Software commands and the status register are explained here following.

Software Commands

Table 25-1 lists software commands. In the standard serial I/O mode, erase operation, programs and reading are controlled by transferring software commands via the RxD pin. Software commands for the serial I/O mode are basically the same as those for the parallel I/O mode, but there are six additional commands to make up for WP pin functions used in the parallel I/O mode. These commands are lock bit disable, lock bit enable, ID check, download, version information output and boot area output.

Table 25-1. Software commands (standard serial I/O mode)

	Control command		2nd byte	3rd byte	4th byte	5th byte	6th byte		When ID is not verificate
1	Page read	FF$_{16}$	Address (middle)	Address (high)	Data output	Data output	Data output	Data output to 259th byte	Not acceptable
2	Page program	41$_{16}$	Address (middle)	Address (high)	Data input	Data input	Data input	Data input to 259th byte	Not acceptable
3	Block erase	20$_{16}$	Address (middle)	Address (high)	D0$_{16}$				Not acceptable
4	Erase all unlocked blocks	A7$_{16}$	D0$_{16}$						Not acceptable
5	Read status register	70$_{16}$	SRD output	SRD1 output					Acceptable
6	Clear status register	50$_{16}$							Not acceptable
7	Read lockbit status	71$_{16}$	Address (middle)	Address (high)	Lock bit data output				Not acceptable
8	Lockbit program	77$_{16}$	Address (middle)	Address (high)	D0$_{16}$				Not acceptable
9	Lockbit enable	7A$_{16}$							Not acceptable
10	Lockbit disable	75$_{16}$							Not acceptable
11	ID check function	F5$_{16}$	Address (low)	Address (middle)	Address (high)	ID size	ID1	To ID7	Acceptable
12	Download function	FA$_{16}$	Size (low)	Size (high)	Check-sum	Data input	To required number of times		Not acceptable
13	Version data output function	FB$_{16}$	Version data output	Version data output	Version data output	Version data output	Version data output	Version data output to 9th byte	Acceptable
14	Boot area output function	FC$_{16}$	Address (middle)	Address (high)	Data output	Data output	Data output	Data output to 259th byte	Not acceptable

- Shading indicates transfer from flash memory microcomputer to serial programmer. All other data is transferred from the serial programmer to the flash memory microcomputer.
- SRD refers to status register data. SRD1 refers to status register 1 data.
- All commands can be accepted when the flash memory is totally blank.

Read Array Command

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

(1) Send the "FF_{16}" command code in the first byte of the transmission.

(2) Send addresses A_8 to A_{15} and A_{16} to A_{23} in the second and the third bytes of the transmission respectively.

(3) From the fourth byte onward, data (D0 to D7) for the page (256 bytes) specified with addresses A_8 to A_{23} will be output sequentially from the smaller address first in synchronization with the rise of the clock.

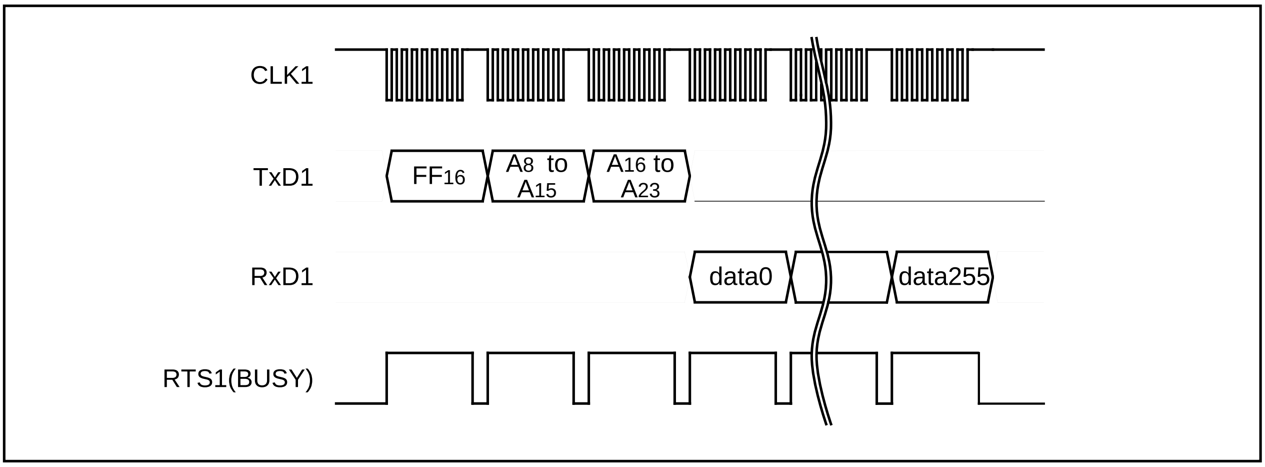


Figure 25-2. Timing for reading array

Read Status Register Command

This command reads status information. When the "70_{16}" command code is sent in the first byte of the transmission, the contents of the status register (SRD) specified in the second byte of the transmission and the contents of status register 1 (SRD1) specified in the third byte of the transmission are read.

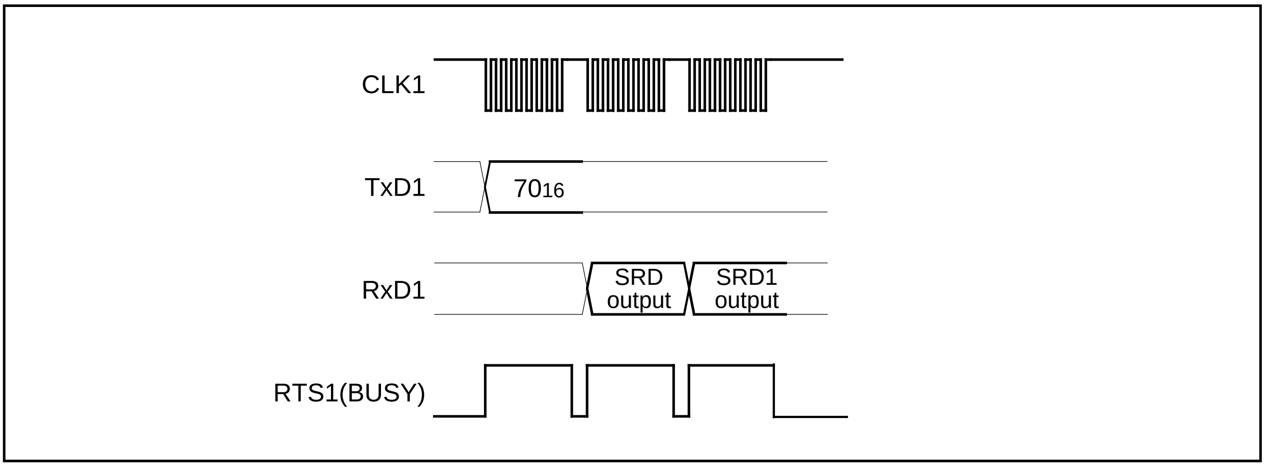


Figure 25-3. Timing for reading the status register

Clear Status Register Command

This command clears the bits (SR3 – SR5) which are set when the status register operation ends in error. When the "50_{16}" command code is sent in the first byte of the transmission, the abovementioned bits are cleared. When the clear status register operation ends, the RTS1 (BUSY) signal changes from the "H" to the "L" level.

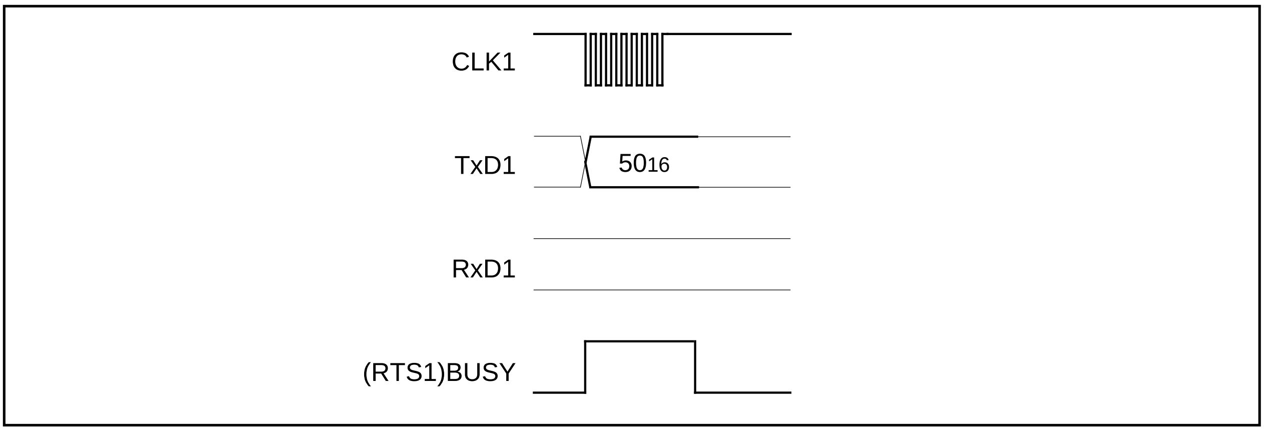


Figure 25-4. Timing for clearing the status register

Page Program Command

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

(1) Send the "41_{16}" command code in the first byte of the transmission.

(2) Send addresses A_8 to A_{15} and A_{16} to A_{23} in the second and the third bytes of the transmission respectively.

(3) From the fourth byte onward, as write data (D0 – D7) for the page (256 bytes) specified with addresses A_8 to A_{23} are input sequentially from the smallest address first, that page is automatically written.

When reception setup for the next 256 bytes ends, the RTS1 (BUSY) signal changes from the "H" to the "L" level. The result of the page program can be known by reading the status register. For more information, see the section on the status register.

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

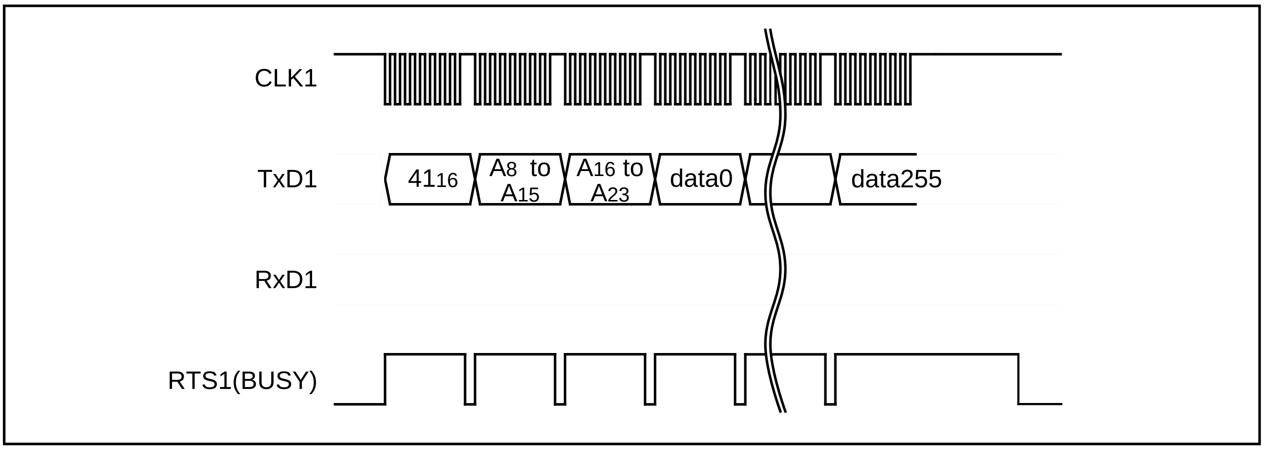


Figure 25-5. Timing for the page program

Block Erase Command

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

(1) Send the "20_{16}" command code in the first byte of the transmission.

(2) Send addresses A_8 to A_{15} and A_{16} to A_{23} in the second and the third bytes of the transmission respectively.

(3) Send the verify command code "D016" in the fourth byte of the transmission. With the verify command code, the erase operation will start for the specified block in the flash memory. Write the highest address of the specified block for addresses A_{16} to A_{23}.

When the block ease ends, the RTS1 (BUSY) signal changes from the "H" to the "L" level. After the block erase ends, the result of the block erase operation can be known by reading the status register. For more information, see the section on the status register.

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

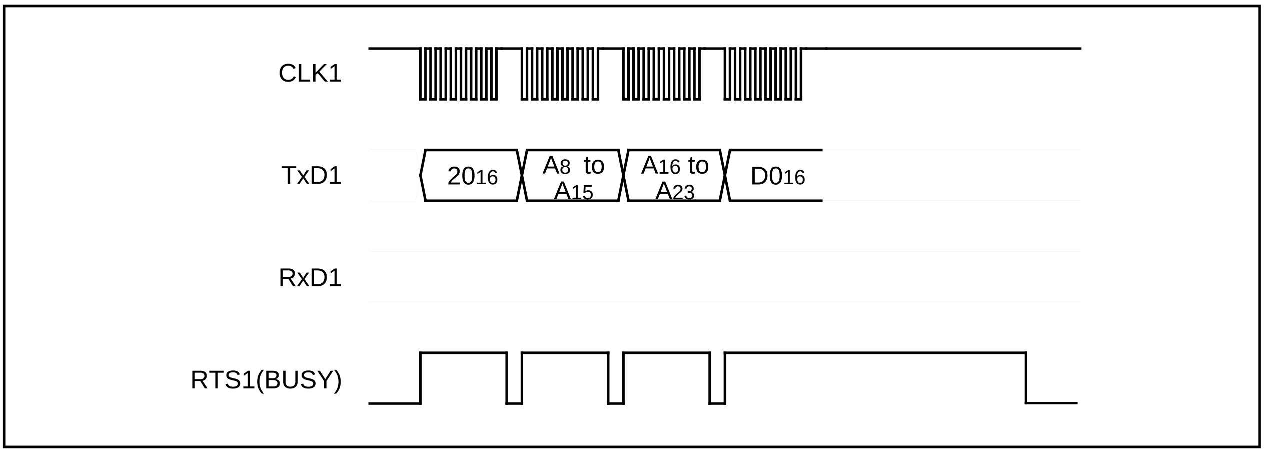


Figure 25-6. Timing for block erase

Erase All Unlocked Blocks Command

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

(1) Send the "A7$_{16}$" command code in the first byte of the transmission.

(2) Send verify command code "D0$_{16}$" in the second byte of the transmission. With the verify command code, the erase operation will start and continue for all the block in the flash memory.

When the block erase ends, the RTS1 (BUSY) signal changes from the "H" to the "L" level. The result of the erase operation can be known by reading the status register. Each block can be erase-protected with the lock bit. For more information, see the section on the data protection function.

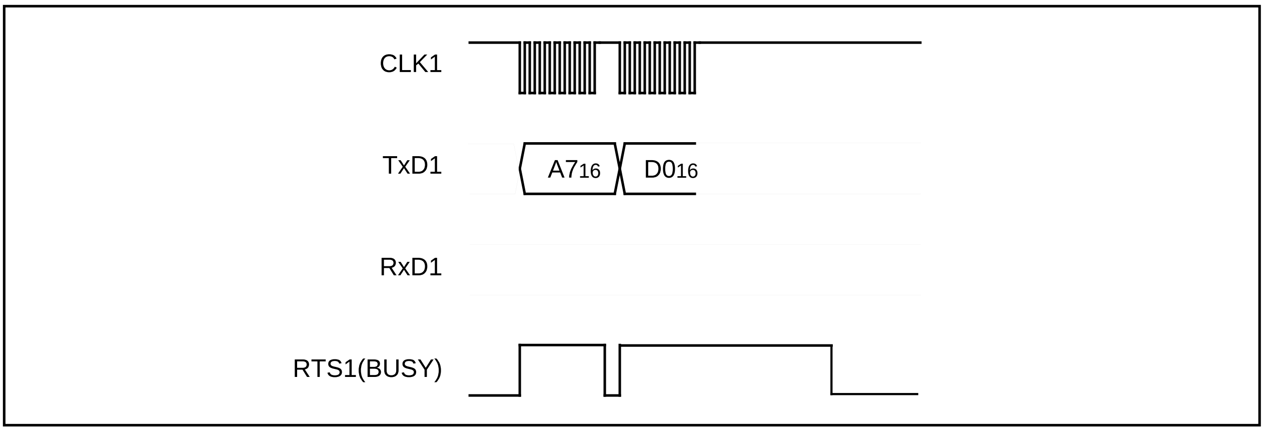


Figure 25-7. Timing for erasing all unlocked blocks

Lock Bit Program Command

This command writes "0" (lock) for the lock bit of the specified block. Execute the lock bit program command as explained below.

(1) Send the "77$_{16}$" command code in the first byte of the transmission.

(2) Send addresses A_8 to A_{15} and A_{16} to A_{23} in the second and the third bytes of the transmission respectively.

(3) Send verify command code "D0$_{16}$" in the fourth byte of the transmission. With the verify command code, "0" is written for the lock bit of the specified block. Write the highest address of the specified block for the addresses A_8 to A_{23} .

When the writing ends, the RTS1 (BUSY) signal changes from the "H" to the "L" level. Lock bit status can be read with the read lock bit status command. For information on the lock bit function, see the section on the data protection function.

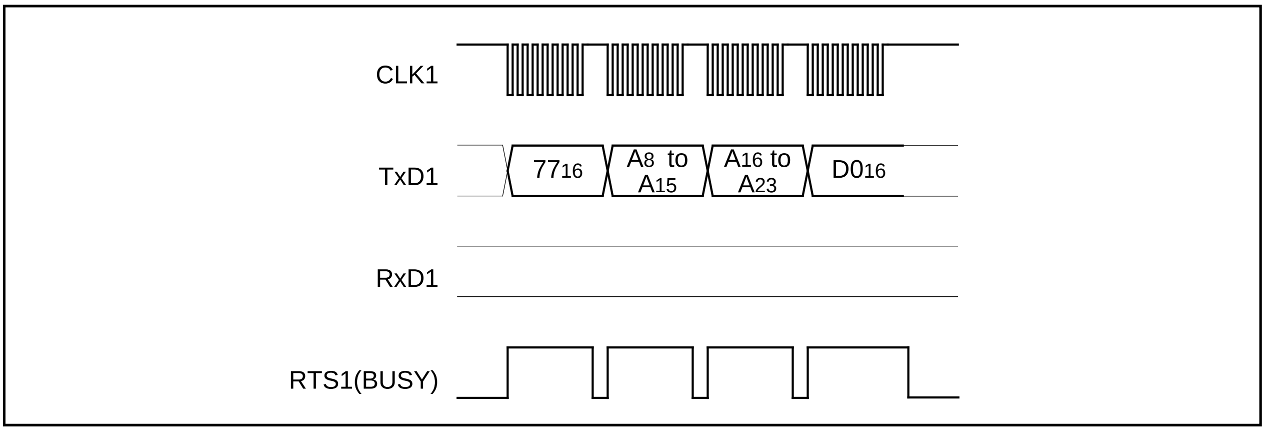


Figure 25-8. Timing for lock bit program

Read Lock Bit Status Command

This command reads the lock bit status of the specified block. Execute the read lock bit status command as explained below.

(1) Send the "71_{16}" command code in the first byte of the transmission.

(2) Send addresses A_8 to A_{15} and A_{16} to A_{23} in the second and the third bytes of the transmission respectively.

(3) The lock bit data of the specified block are output in the fourth byte of the transmission. Write the highest address of the specified block for the addresses A_8 to A_{23} .

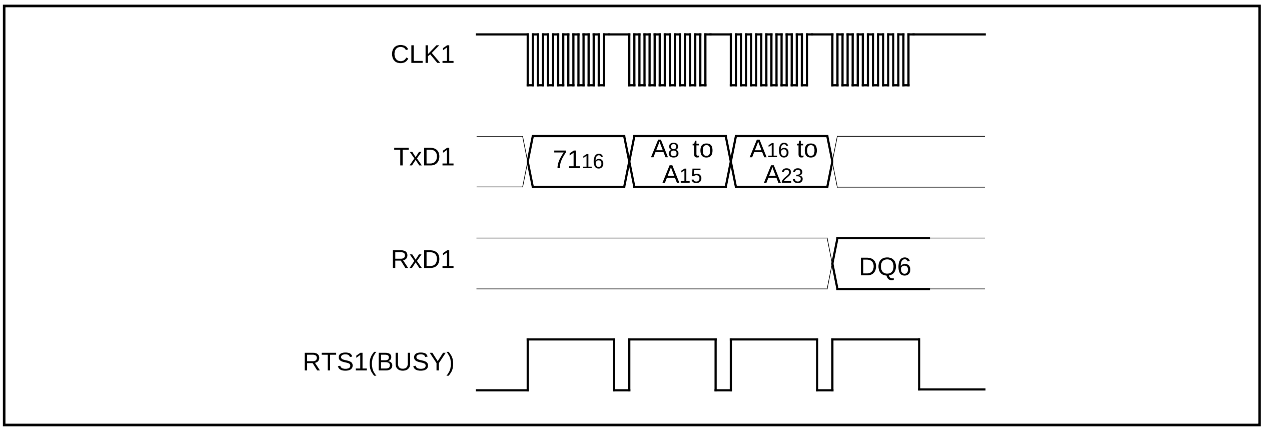


Figure 25-9. Timing for reading lock bit status

Lock Bit Enable Command

This command enables the lock bit in blocks whose bit has been disabled with the lock bit disable command. It functions in the same way as the $\overline{\text{WP}}$ pin in the parallel I/O mode. The command code "$7A_{16}$" is sent in the first byte of the serial transmission. This command enables the lock bit function only; it does not set the lock bit itself.

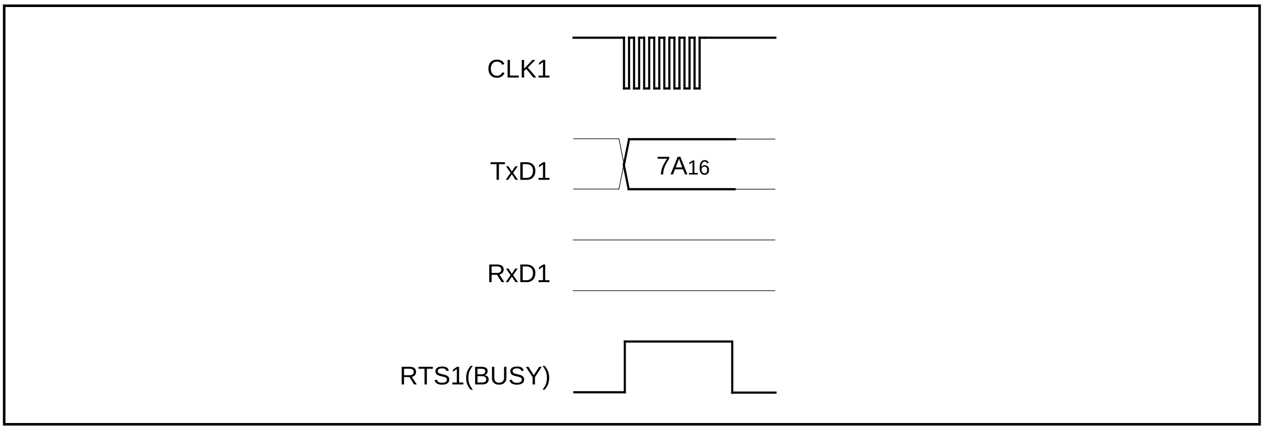


Figure 25-10. Timing for enabling the lock bit

Lock Bit Disable Command

This command disables the lock bit. It functions in the same way as the $\overline{WP}$ pin in the parallel I/O mode. The command code "7A$_{16}$" is sent in the first byte of the serial transmission. This command enables the lock bit function only; it does not set the lock bit itself. however, if an erase command is executed after executing the lock bit disable command, "0" (locked) lock bit data will be set to "1" (unlocked) after the erase operation ends. In any case, after the reset is cancelled, the lock bit is enabled.

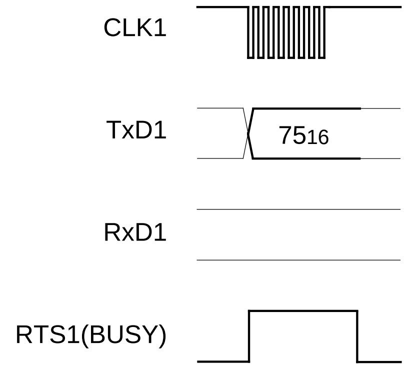


Figure 25-11. Timing for disabling the lock bit

Download Command

This command downloads a program into the RAM for execution. Execute the download command as explained below.

(1) Send the "FA$_{16}$" command code in the first byte of the transmission.

(2) Send the program size in the second and the third bytes of the transmission.

(3) Send the check sum in the fourth byte of the transmission. The check sum is added to all the data in the fifth byte onward.

(4) The program to execute is sent in the fifth byte onward.

When all the data are transmitted, if the check sum matches, the download program is executed.
The size of the program may vary according to the internal RAM.

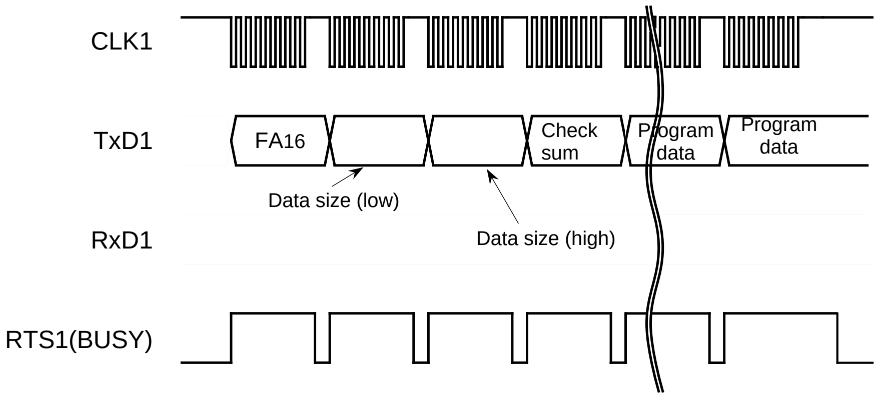


Figure 25-12. Timing for download

Version Information Output Command

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

(1) Send the "FB$_{16}$" command code in the first byte of the transmission.

(2) The version information will be output from the second byte onward. These data are composed of 8 ASCII code characters.

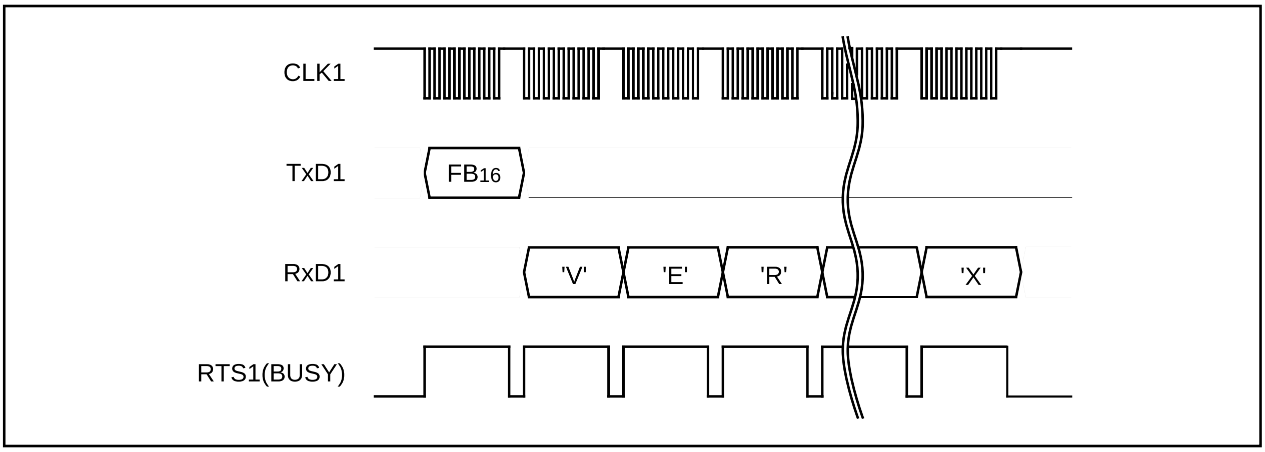


Figure 25-13. Timing for version information output

Boot Area Output Command

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

(1) Send the "FC$_{16}$" command code in the first byte of the transmission.

(2) Send addresses A_8 to A_{15} and A_{16} to A_{23} in the second and the third bytes of the transmission respectively.

(3) From the fourth byte onward, data (D0 – D7) for the page (256 bytes) specified with addresses A_8 to A_{23} will be output sequentially from the smallest address first, in synchronization with the rise of the clock.

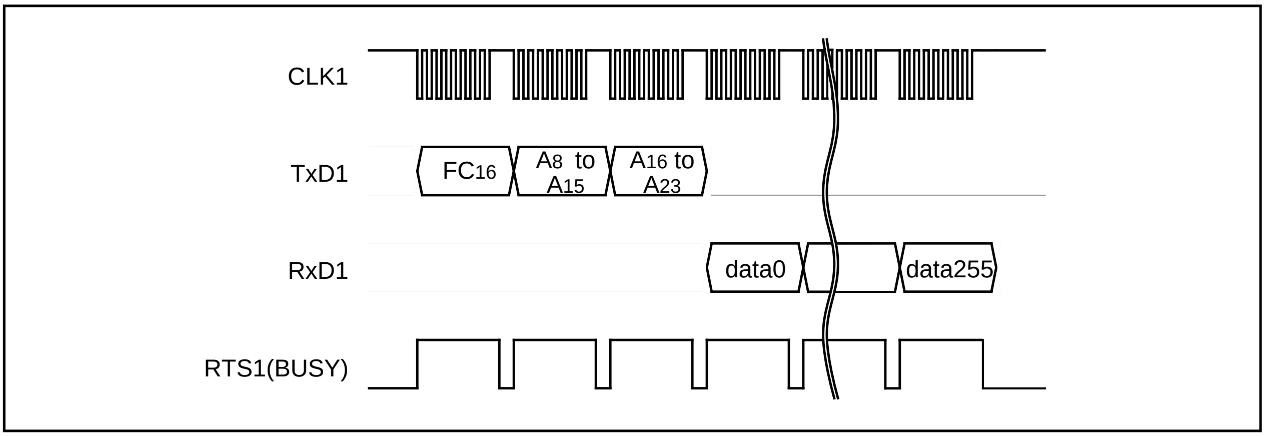


Figure 25-14. Timing for boot area output

ID Check

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

(1) Send the "$F5_{16}$" command code in the first byte of the transmission.

(2) Send addresses A_0 to A_7, A_8 to A_{15} and A_{16} to A_{23} of the first byte of the ID code in the second, the third and the fourth bytes of the transmission respectively.

(3) Send the number of data sets of the ID code in the fifth byte.

(4) The ID code is sent in the sixth byte onward, starting with the first byte of the code.

When all the data are transmitted, if the check sum matches, the download program is executed. The size of the program may vary according to the internal RAM.

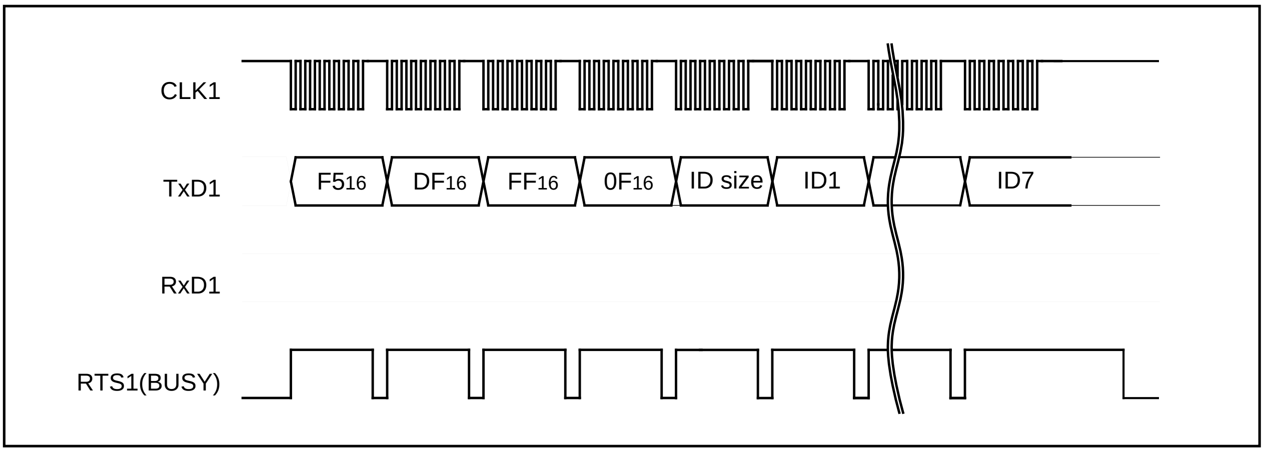
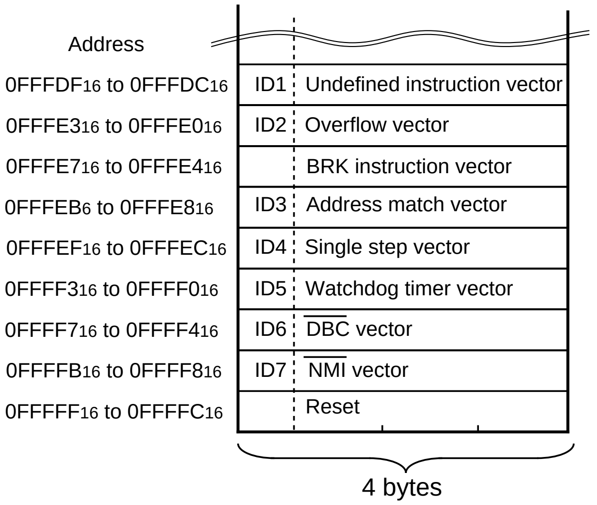


Figure 25-15. Timing for ID check

ID Code

When the flash memory is not blank, the ID code sent from the serial programmer and the ID code written in the flash memory are compared to see if they match. If the codes do not match, the command sent from the serial programmer is not accepted. An ID code contains 8 bits of data. Its area is, from the first byte, $0FFFDF_{16}$, $0FFFE3_{16}$, $0FFFEB_{16}$, $0FFFEF_{16}$, $0FFFF3_{16}$, $0FFFF7_{16}$ and $0FFFFB_{16}$. Write a program into the flash memory, which already has the ID code set for these addresses.



Address		
$0FFFDF_{16}$ to $0FFFDC_{16}$	ID1	Undefined instruction vector
$0FFFE3_{16}$ to $0FFFE0_{16}$	ID2	Overflow vector
$0FFFE7_{16}$ to $0FFFE4_{16}$		BRK instruction vector
$0FFFEB_6$ to $0FFFE8_{16}$	ID3	Address match vector
$0FFFEF_{16}$ to $0FFFEC_{16}$	ID4	Single step vector
$0FFFF3_{16}$ to $0FFFF0_{16}$	ID5	Watchdog timer vector
$0FFFF7_{16}$ to $0FFFF4_{16}$	ID6	$\overline{\text{DBC}}$ vector
$0FFFFB_{16}$ to $0FFFF8_{16}$	ID7	$\overline{\text{NMI}}$ vector
$0FFFFF_{16}$ to $0FFFFC_{16}$		Reset

Figure 25-16. ID code storage addresses

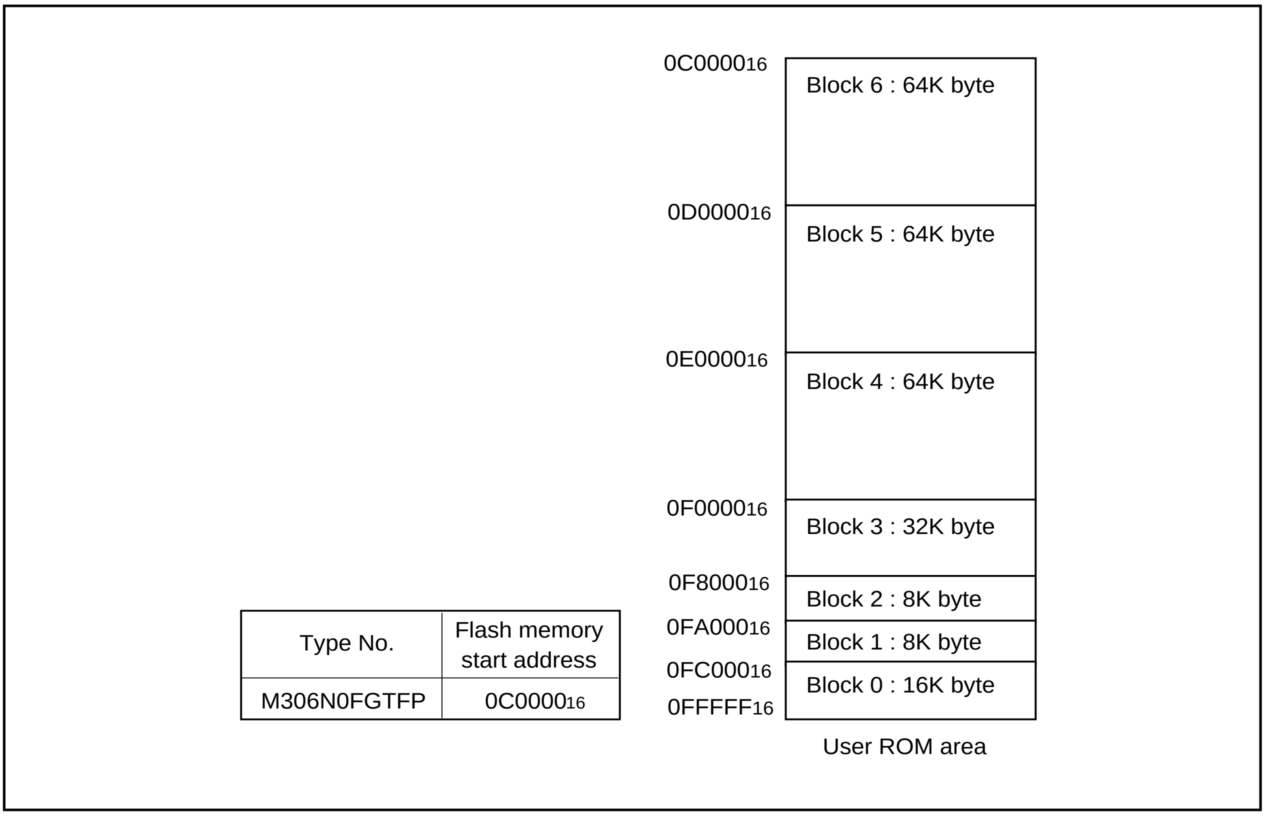
Data Protection (Block Lock)

Each of the blocks in Figure 24-1 has a nonvolatile lock bit that specifies protection (block lock) against erasing/writing. A block is locked (writing "0" for the lock bit) with the lock bit program command. Also, the lock bit of any block can be read with the read lock bit status command.

Block lock disable/enable is determined by the status of the lock bit itself and execution status of the lock bit disable- and lock enable bit commands.

(1) After the reset is cancelled and the lock bit enable command executed, the specified block can be locked/unlocked using the lock bit (lock bit data). Blocks with "0" lock bit data are locked and cannot be erased or written in. On the other hand, blocks with "1" lock bit data are unlocked and can be erased or written in.

(2) After the lock bit enable command is executed, all the blocks are unlocked regardless of lock bit data status and can be erased or written in. In this case, lock bit data that were "0" before the block was erased will be set to "1" (unlocked) after erasing, therefore the block is actually unlocked with the lock bit.



Type No.	Flash memory start address
M306N0FGTFP	$0C0000_{16}$

Figure 25-17. Blocks in the user area

Status Register (SRD)

The status register indicates operating status of the flash memory and status such as whether an erase operation or a program ended successfully or in error. It can be read by writing the read status register command (70_{16}). Also, the status register is cleared by writing the clear status register command (50_{16}). Table 25-2 gives the definition of the status register bits. After clearing the reset, the status register outputs "80_{16}".

Table 25-2. Status register (SRD)

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

Write State Machine (WSM) Status (SR7)

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

Erase Status (SR5)

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

Program Status (SR4)

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

Program Status After Program (SR3)

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

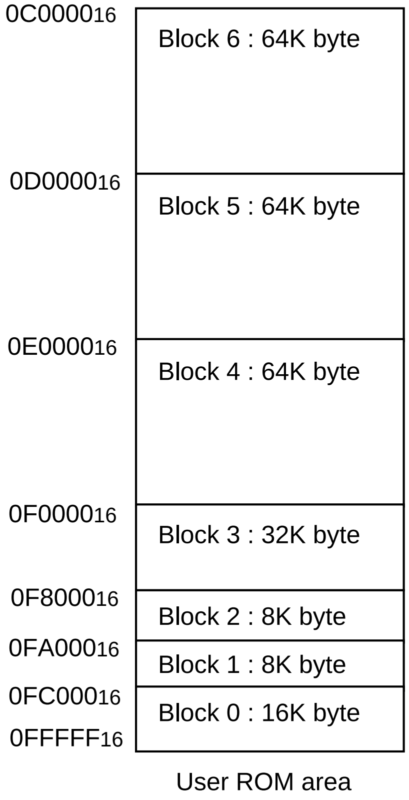
If "1" is written for any of SR5-, SR4- or SR3 bits, the page program-, block erase-, erase all unlocked blocks- and lock bit program commands are not accepted. Before executing these commands, execute the clear status register command (50_{16}) and clear the status register.

Also in the following cases, both SR4 and SR5 are set to "1" (command sequence error).

(1) If data other than "$D0_{16}$" or "FF_{16}" are input for the second bus cycle data of the lock bit program command (77_{16}/$D0_{16}$).

(2) If data other than "$D0_{16}$" or "FF_{16}" are input for the second bus cycle data of the block erase command (20_{16}/$D0_{16}$).

(3) If data other than "$D0_{16}$" or "FF_{16}" are input for the second bus cycle data of the erase all unlocked blocks command ($A7_{16}$/$D0_{16}$). However, inputting "FF_{16}" engages the read array mode and cancels the setup command in the first bus cycle.



Type No.	Flash memory start address
M306N0FGTFP	$0C0000_{16}$

Figure 25-17. Blocks in the user area

Status Register 1 (SRD1)

The status register 1 indicates status of serial communication, results of ID checks and those of check sum comparisons. It can be read after the SRD by writing the read status register command (70_{16}). Also, status register 1 is cleared by writing the clear status register command (50_{16}).

Table 25-4 gives the definition of each status register bit. "00_{16}" is output when power is turned on and the flag status is maintained even after the reset.

Table 25-3. Status register 1 (SRD 1)

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

Block Update Completed Bit (SR15)

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

Check Sum Consistency Bit (SR12)

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

ID Check Completed Bits (SR11 and SR 10)

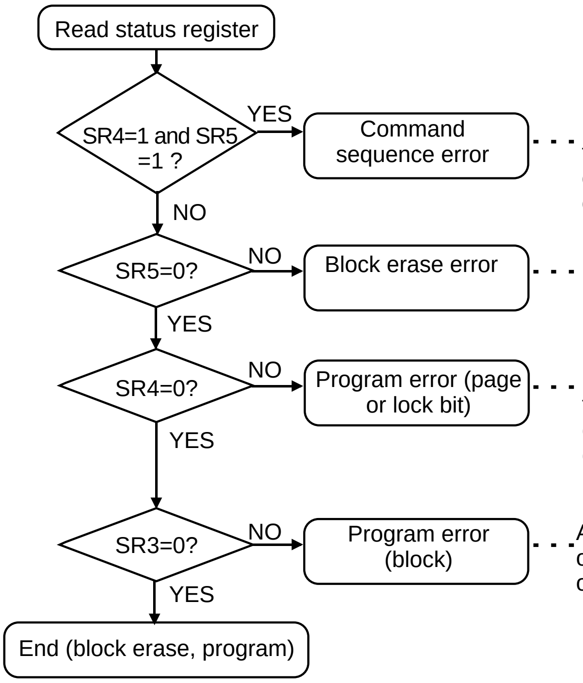
These flags indicate the result of ID checks. Some commands cannot be accepted without an ID check.

Data Reception Time Out (SR9)

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

Full Status Check

Results of executed erase- and program operations can be known by running a full status check. Figure 25-18 shows a flowchart of the full status check and explains how to remedy errors which may occur.



. . . Execute the clear status register command (50_{16}) to clear the status register. Try performing the operation one more time after confirming that the command is entered correctly.

. . . Should a block erase error occur, the block in error cannot be used.

. . . Execute the read lock bit status command (71_{16}) to see if the block is locked. After removing lock, execute write operation in the same way. If the error still occurs, the page in error cannot be used.

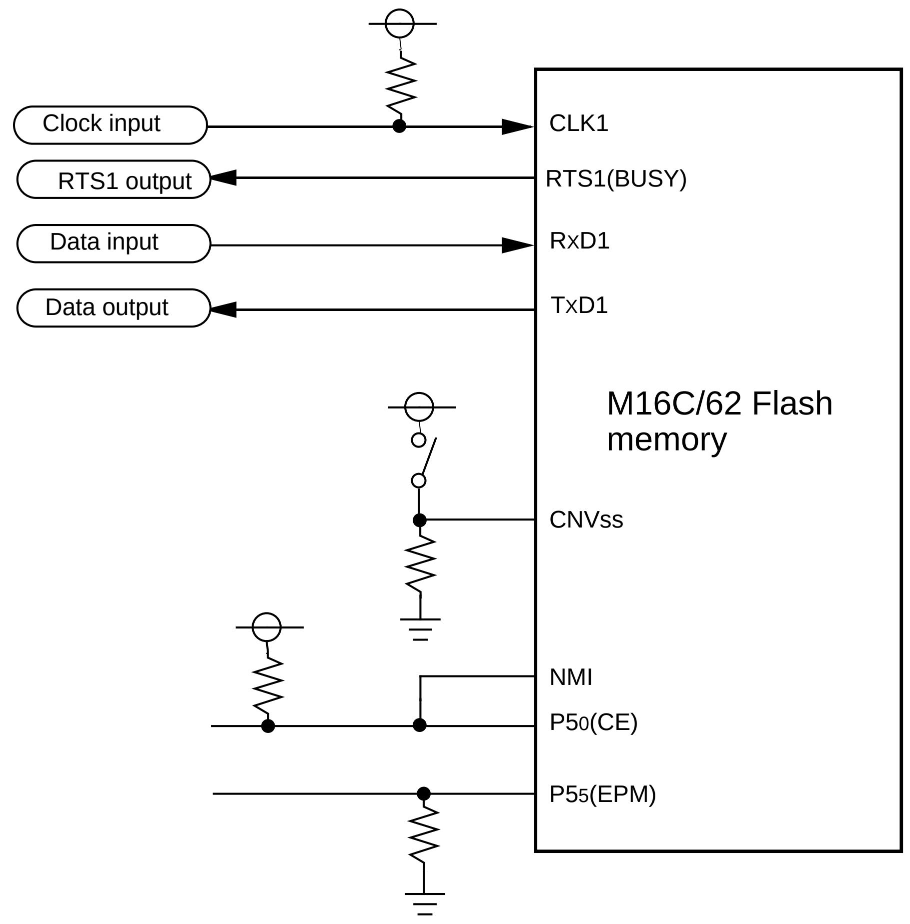
. . . After erasing the block in error, execute write operation one more time. If the same error still occurs, the block in error cannot be used.

Note: When one of SR5 to SR3 is set to 1, none of the page program, block erase, erase all unlock blocks and lock bit program commands is accepted. Execute the clear status register command (50_{16}) before executing these commands.

Figure 25-18. Full status check flowchart and remedial procedure for errors

Example Circuit Application for The Standard Serial I/O Mode

The figure blow shows a circuit application for the standard serial I/O mode. Control pins may vary according to programmer, therefore see the programmer manual for more information.



(1) Control pins and external circuitry will vary according to programmer. For more information, see the programmer manual.
(2) In this example, the microprocessor mode and standard serial I/O mode are switched via a switch.

Figure 25-19. Example circuit application for the standard serial I/O mode

MITSUBISHI SEMICONDUCTORS
M16C/6N Group DATA SHEET REV.B

Second Edition March1999

Editioned by
Committee of editing of Mitsubishi Semiconductor DATA SHEET

Published by
Mitsubishi Electric Corp., Kitaitami Works