To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: http://www.renesas.com

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: “Standard”, “High Quality”, and “Specific”. The recommended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as “Specific” without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as “Specific” or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is “Standard” unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

   “Standard”: Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.

   “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.

   “Specific”: Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

To all our customers

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

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

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

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

Description

The M30220 group of single-chip microcomputers are built using the high-performance silicon gate CMOS process using a M16C/60 Series CPU core. The M30220 group has LCD controller/driver. M30220 group is packaged in a 144-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.

Features

• Basic machine instructions	Compatible with the M16C/60 series
• Memory capacity	See figure memory expansion
• Shortest instruction execution time	100ns (f(X_{IN})=10MHz)
• Supply voltage	4.0V to 5.5V (f(X_{IN})=10MHz) 2.7V to 5.5V (f(X_{IN})=7MHz with software one-wait)
• Interrupts	26 internal and 8 external interrupt sources, 4 software, 7 levels (including key input interrupt)
• Multifunction 16-bit timer	Timer A (output) x 8, timer B (input) x 6
• Real time port outputs	8 bits X 4 lines
• Serial I/O	3 channel for UART or clock synchronous
• DMAC	2 channels (trigger: 26 sources)
• A-D converter	10 bits X 8 channels
• D-A converter	8 bits X 3 channels
• Watchdog timer	1 line
• Programmable I/O	104 lines (32 lines are shared with LCD outputs)
• Output port	16 lines (shared with LCD outputs)
• Input port	1 line ($P7_7$, shared with $\overline{NMI}$ pin)
• LCD drive control circuit	1/2, 1/3 bias 2, 3 and 4 time sharing 4 common outputs 48 segment outputs built-in Charge-pump
• Key input interrupt	20 lines
• Clock generating circuit	2 built-in clock generation circuits (built-in feedback resistor, and external ceramic or quartz oscillator)

Applications

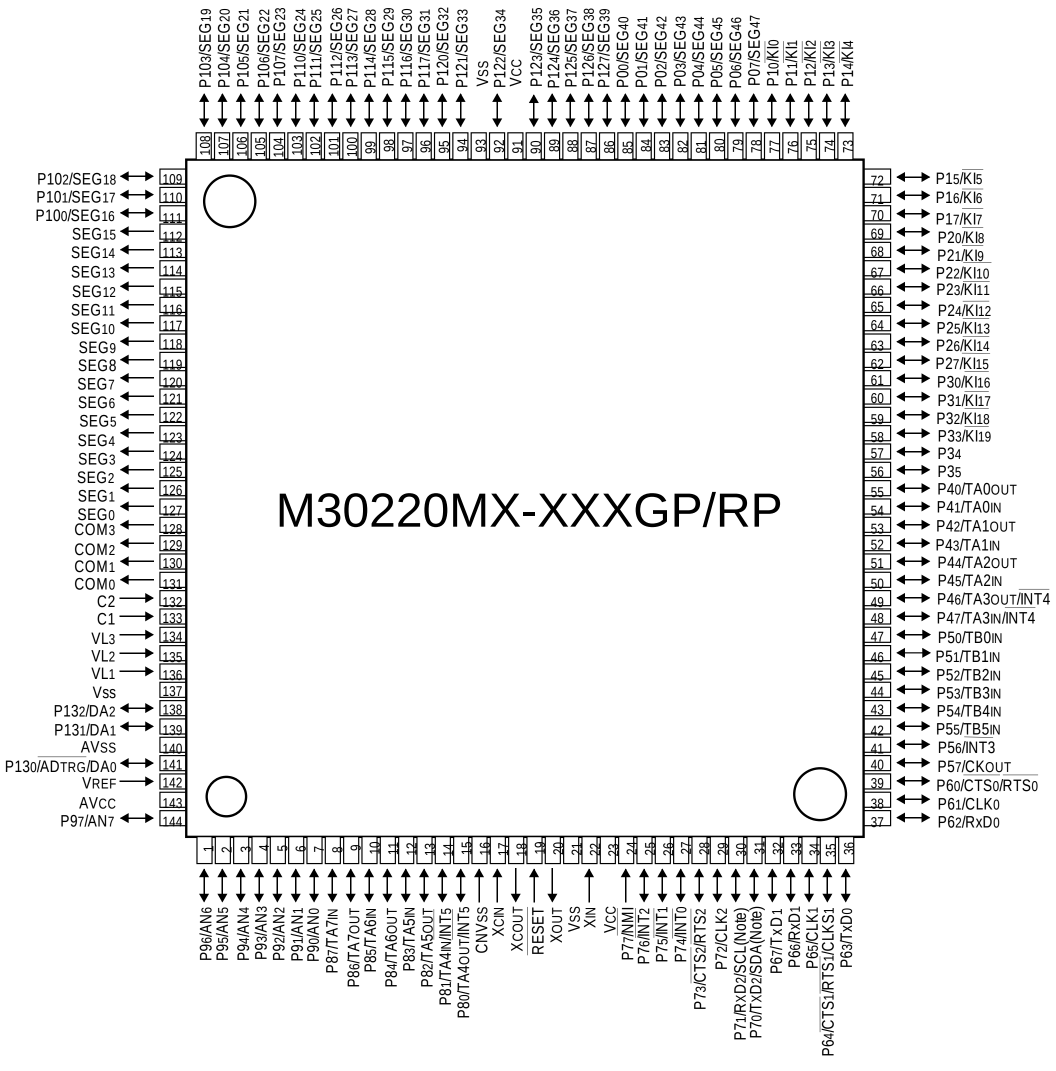
Camera, Home appliances, Portable equipment, Drop meter, Audio, Office equipment, etc.

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

Pin Configuration

Figure 1.1.1 shows the pin configurations (top view).

PIN CONFIGURATION (top view)



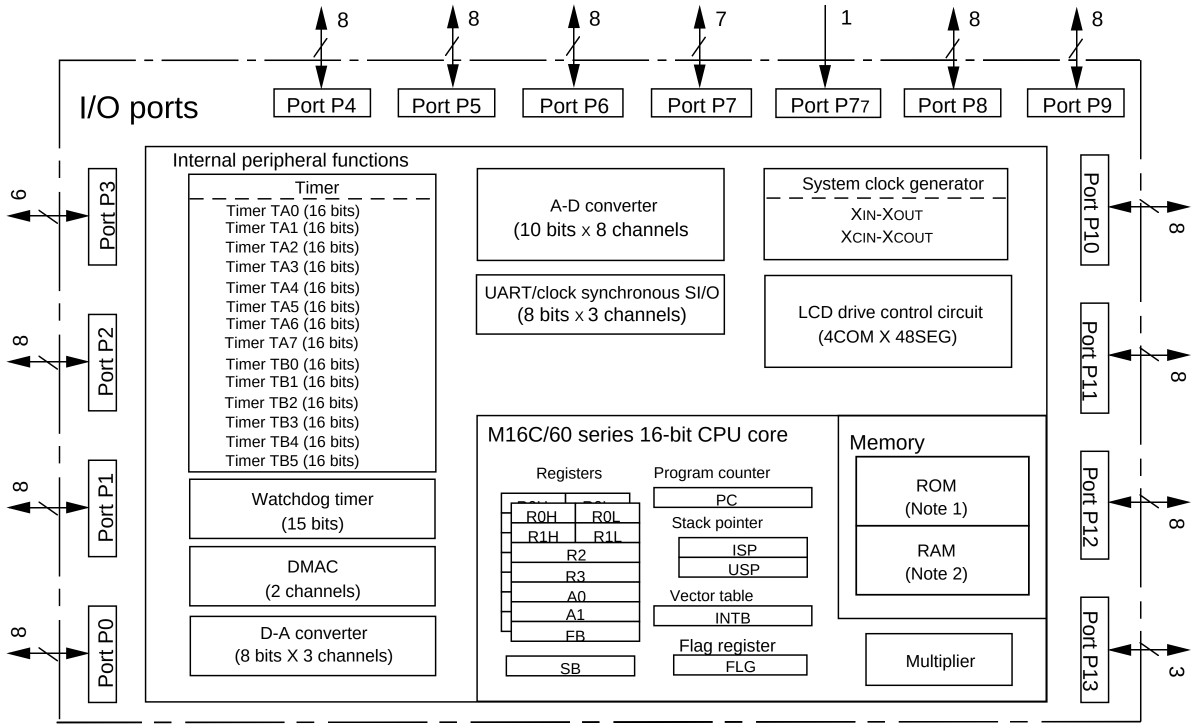
Note: P70 and P71 are N channel open-drain output pin.

Package: 144P6Q-A, 144PFB-A

Figure 1.1.1. Pin configuration (top view)

Block Diagram

Figure 1.1.2 is a block diagram of the M30220 group.



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

Figure 1.1.2. Block diagram of M30220 group

Performance Outline

Table 1.1.1 is performance outline of M30220 group.

Table 1.1.1. Performance outline of M30220 group

Item			Performance
Number of basic instructions			91 instructions
Shortest instruction execution time			100ns (f(X_{IN})=10MHz
Memory capacity	ROM		96 Kbytes
	RAM		6 Kbytes
I/O port	P0 to P13 (except $P7_7$)		8 bits x 11, 3 bits x 1, 6 bits x 1, 7 bits x 1
Input port	$P7_7$		1 bit x 1
Output port	SEG_0 to SEG_{15}		2 bits x 8
Multifunction timer	TA0 to TA7		16 bits x 8
	TB0 to TB5		16 bits x 6
Real time port outputs			8 bits x 4 lines
Serial I/O	UART0 to UART2		(UART or clock synchronous) x 3
A-D converter			10 bits x 8 channels
D-A converter			8 bits x 3 channels
DMAC			2 channel(trigger:26 sources)
LCD	COM_0 to COM_3		4 lines
	SEG_0 to SEG_{47}		48 lines (32 lines are shared with I/O ports)
Watchdog timer			15 bits x 1 (with prescaler)
Interrupt			26 internal and 8 external sources, 4 software sources
Clock generating circuit			2 built-in clock generation circuits (built-in feedback resistor, and external ceramic or quartz oscillator)
Supply voltage			4.0V to 5.5V (f(X_{IN})=10MHz)
			2.7V to 5.5V (f(X_{IN})=7MHz with software one-wait)
Power consumption			18mW (V_{CC}=3V, f(X_{IN})=7MHz with software one-wait)
I/O characteristics	I/O withstand voltage (P0 to P13)		5 V
	Output current	P1 to P9,P13	5 mA
		P0, P10 to P12	0.1mA("H" output), 2.5mA("L" output)
Device configuration			CMOS high-performance silicon gate
Package			144-pin plastic mold QFP

Mitsubishi plans to release the following products in the M30220 group:

(1) Support for mask ROM version, flash memory version

(2) ROM capacity

(3) Package

144P6Q-A : Plastic molded QFP (mask ROM and flash memory versions)

144PFB-A : Plastic molded QFP(mask ROM and flash memory versions)

Figure 1.1.3 shows the ROM expansion and figure 1.1.4 shows the Type No., memory size, and package.

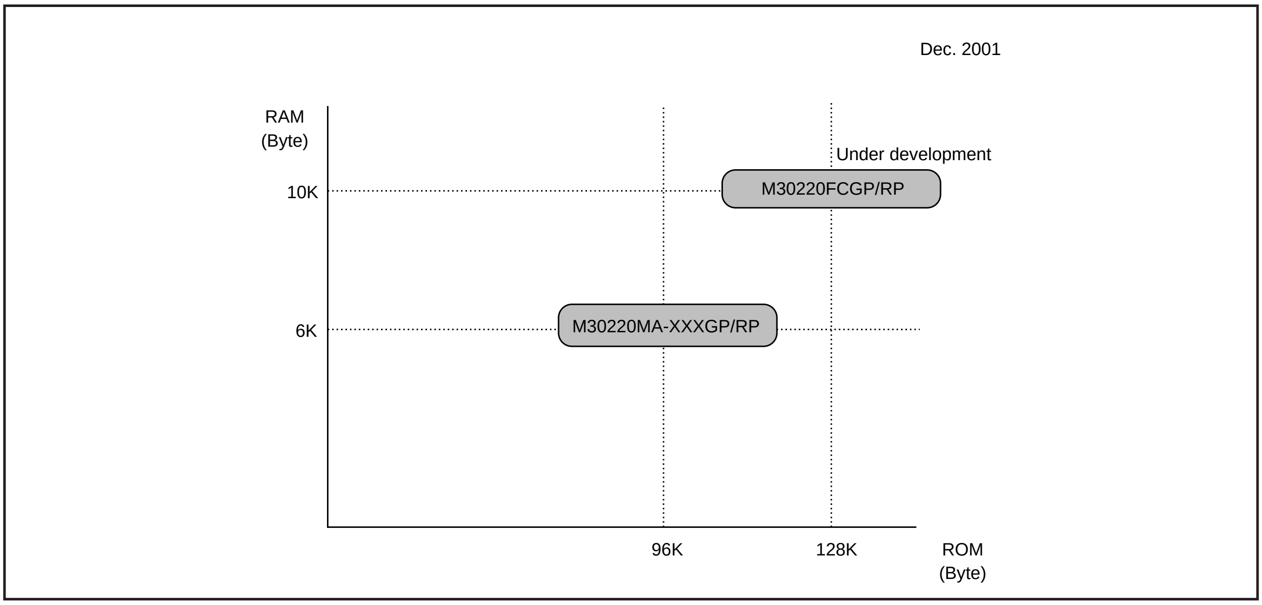


Figure 1.1.3. Memory expansion

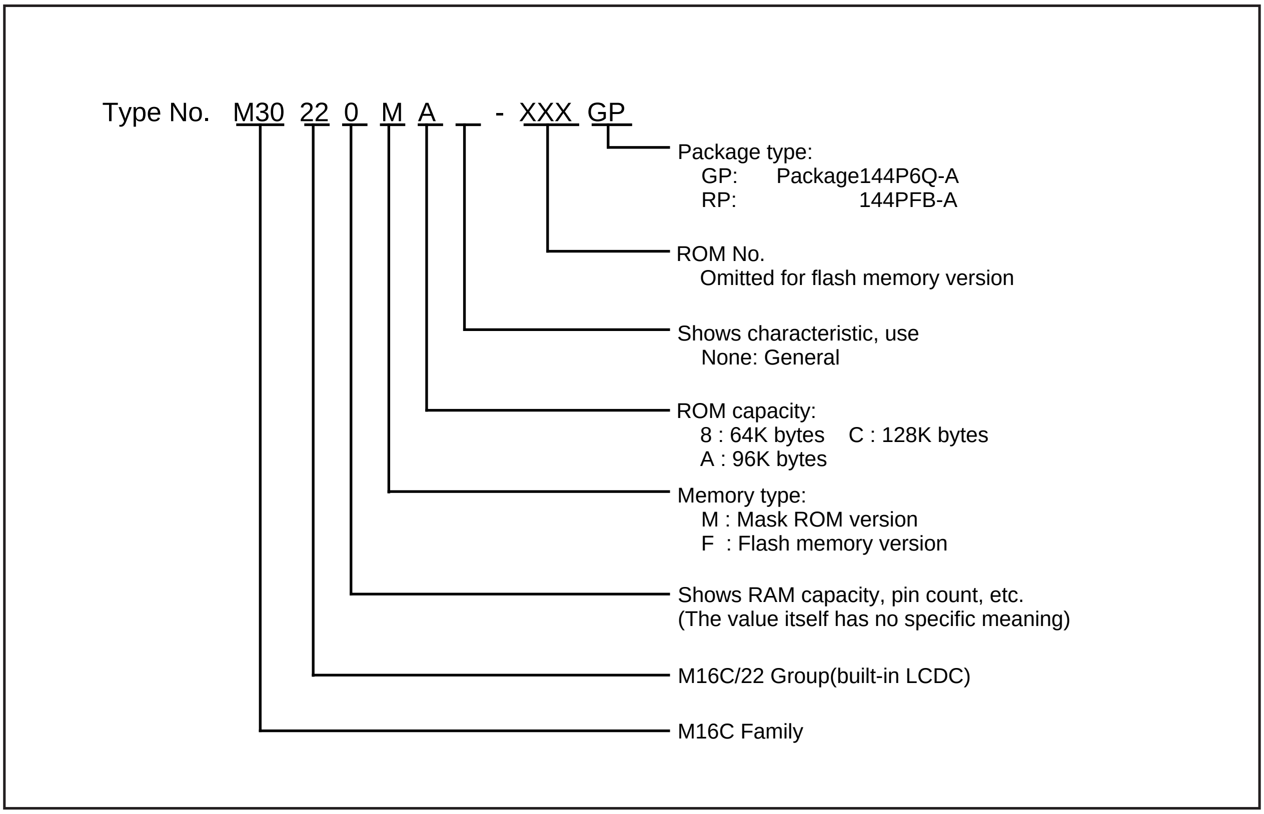

Type No. M30 22 0 M A - XXX GP
Package type:
GP: Package144P6Q-A
RP: 144PFB-A
ROM No.
Omitted for flash memory version
Shows characteristic, use
None: General
ROM capacity:
8 : 64K bytes C : 128K bytes
A : 96K bytes
Memory type:
M : Mask ROM version
F : Flash memory version
Shows RAM capacity, pin count, etc.
(The value itself has no specific meaning)
M16C/22 Group(built-in LCDC)
M16C Family

Figure 1.1.4. Type No., memory size, and package

Pin Description

Pin name	Signal name	I/O	Function
V_{CC}, V_{SS}	Power supply input		Supply 2.7 to 5.5 V to the V_{CC} pin. Supply 0 V to the V_{SS} pin.
CNV_{SS}	CNV_{SS}	I	Connect it to the V_{SS} pin via resistor.
$\overline{RESET}$	Reset input	I	A “L” on this input resets the microcomputer.
X_{IN} X_{OUT}	Clock input Clock output	I O	These pins are provided for the main clock generating circuit. Connect a ceramic resonator or crystal between the X_{IN} and the X_{OUT} pins. To use an externally derived clock, input it to the X_{IN} pin and leave the X_{OUT} pin open.
X_{CIN} X_{COUT}	Clock input Clock output	I O	These pins are provided for the sub clock generating circuit. Connect a ceramic resonator or crystal between the X_{CIN} and the X_{COUT} pins. To use an externally derived clock, input it to the X_{CIN} pin and leave the X_{COUT} pin open.
AV_{CC}	Analog power supply input		This pin is a power supply input for the A-D converter. Connect it to V_{CC}.
AV_{SS}	Analog power supply input		This pin is a power supply input for the A-D converter. Connect it to V_{SS}.
V_{REF}	Reference voltage input	I	This pin is a reference voltage input for the A-D converter.
$P0_0$ to $P0_7$	I/O port P0	I/O	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 use as LCD segment output and real time port output.
$P1_0$ to $P1_7$	I/O port P1	I/O	This is an 8-bit I/O port equivalent to P0. Pins in this port also function as input pins for the key input interrupt function and real time port output.
$P2_0$ to $P2_7$	I/O port P2	I/O	This is an 8-bit I/O port equivalent to P0. Pins in this port also function as input pins for the key input interrupt function and real time port output.
$P3_0$ to $P3_5$	I/O port P3	I/O	This is a 6-bit I/O port equivalent to P0. $P3_0$ to $P3_3$ also function as input pins for the key input interrupt function.
$P4_0$ to $P4_7$	I/O port P4	I/O	This is a 8-bit I/O port equivalent to P0. Pins in this port also function as timer A0 to A3 I/O pins, $\overline{INT_4}$ input pin as selected by software.
$P5_0$ to $P5_7$	I/O port P5	I/O	This is a 8-bit I/O port equivalent to P0. Pins in this port also function as timer B0 to B5 and $\overline{INT_3}$ input pins, CK_{OUT} output pin as selected by software.
$P6_0$ to $P6_7$	I/O port P6	I/O	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.

Pin Description

Pin name	Signal name	I/O	Function
$P7_0$ to $P7_6$ $P7_7$	I/O port P7	I/O I	$P7_0$ to $P7_6$ are I/O ports equivalent to P0 ($P7_0$ and $P7_1$ are N channel open-drain output). Pins in this port also function as UART2 I/O pin, $\overline{INT_0}$ to $\overline{INT_2}$ input pins as selected by software. $P7_7$ is an input-only port that also functions for $\overline{NMI}$.
$P8_0$ to $P8_7$	I/O port P8	I/O	This is a 8-bit I/O port equivalent to P0. Pins in this port also function as timer A4 to A7 I/O pins, $\overline{INT_5}$ input pin as selected by software.
$P9_0$ to $P9_7$	I/O port P9	I/O	This is an 8-bit I/O port equivalent to P0. Pins in this port also function as A-D converter analog input pins as selected by software.
$P10_0$ to $P10_7$	I/O port P10	I/O	This is an 8-bit I/O port equivalent to P0. Pins in this port also function as SEG output for LCD as selected by software.
$P11_0$ to $P11_7$	I/O port P11	I/O	This is an 8-bit I/O port equivalent to P0. Pins in this port also function as SEG output for LCD as selected by software.
$P12_0$ to $P12_7$	I/O port P12	I/O	This is an 8-bit I/O port equivalent to P0. Pins in this port also function as SEG output for LCD and real time port output.
$P13_0$ to $P13_2$	I/O port P13	I/O	This is an 3-bit I/O port equivalent to P0. Pins in this port also function as D-A converter analog output pins or start trigger for A-D input pins.
SEG_0 to SEG_{15}	Segment output	O	Pins in this port function as SEG output for LCD drive circuit.
COM_0 to COM_3	Common output	O	Pins in this port function as common output for LCD drive circuit.
VL_1 to VL_3	Power supply input for LCD		Power supply input for LCD drive circuit.
C_1, C_2	Step-up condenser connect port		Pins in this port function as external pin for LCD step-up condenser. Connect a condenser between C_1 and C_2.

Operation of Functional Blocks

The M30220 group accommodates certain units in a single chip. These units include ROM and RAM to store instructions and data and the central processing unit (CPU) to execute arithmetic/logic operations. Also included are peripheral units such as timers, real time port, serial I/O, LCD drive control circuit, D-A converter, A-D converter, DMAC and I/O ports.

The following explains each unit.

Memory

Figure 1.4.1 is a memory map of the M30220 group. The address space extends the 1M bytes from address 00000$_{16}$ to FFFFF$_{16}$. From FFFFF$_{16}$ down is ROM. For example, in the M30220MA-XXXGP, there is 96K bytes of internal ROM from E8000$_{16}$ to FFFFF$_{16}$. The vector table for fixed interrupts such as the reset and $\overline{\mathrm{NMI}}$ are mapped to FFFDC$_{16}$ to FFFFF$_{16}$. The starting address of the interrupt routine is stored here. The address of the vector table for timer interrupts, etc., can be set as desired using the internal register (INTB). See the section on interrupts for details.

From 00400$_{16}$ up is RAM. For example, in the M30220MA-XXXGP, 6K bytes of internal RAM is mapped to the space from 00400$_{16}$ to 01BFF$_{16}$. 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, timers, and LCD, etc. Figures 1.7.1 to 1.7.3 are location of peripheral unit control registers. Any part of the SFR area that is not occupied is reserved and cannot be used for other purposes.

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

RAM size	Address XXXXX$_{16}$
4K bytes	013FF$_{16}$
6K bytes	01BFF$_{16}$
10K bytes	02BFF$_{16}$

ROM size	Address YYYYY$_{16}$
64K bytes	F0000$_{16}$
96K bytes	E8000$_{16}$
128K bytes	E0000$_{16}$

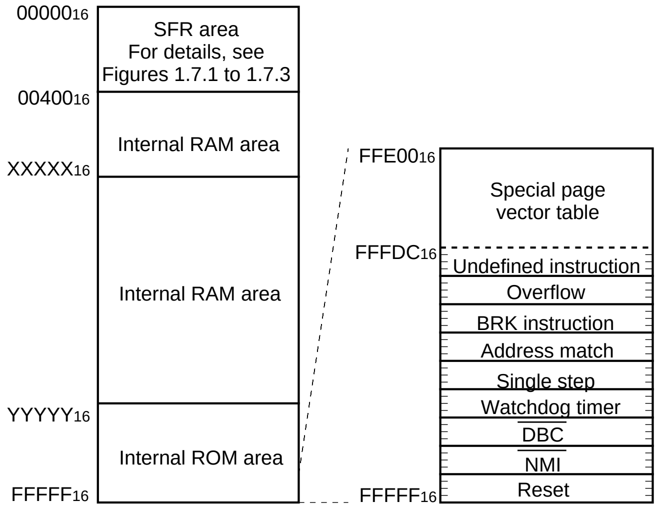


Figure 1.4.1. Memory map

Central Processing Unit (CPU)

The CPU has a total of 13 registers shown in Figure 1.5.1. Seven of these registers (R0, R1, R2, R3, A0, A1, and FB) come in two sets; therefore, these have two register banks.

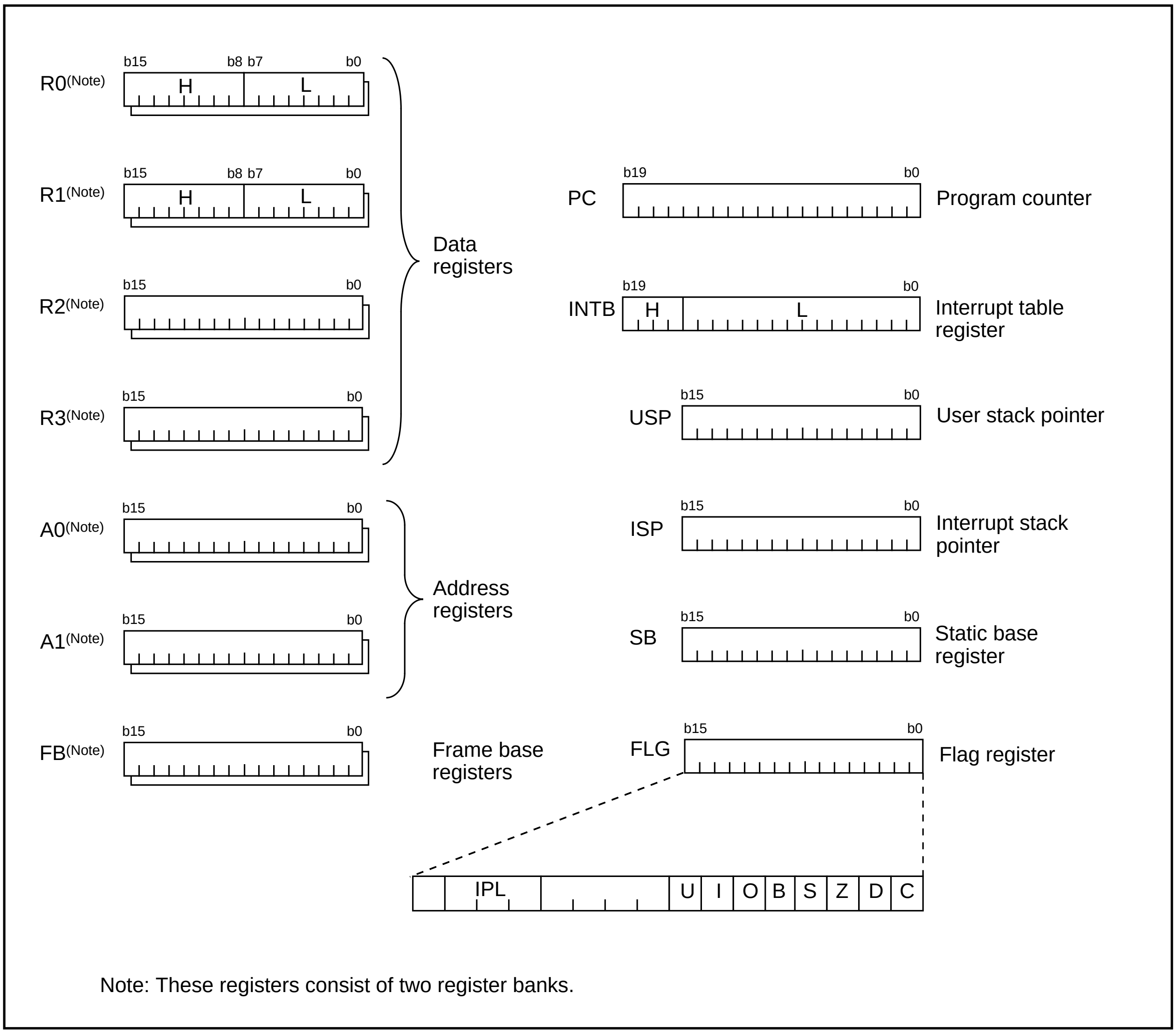


Figure 1.5.1. Central processing unit register

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

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

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

(2) Address registers (A0 and A1)

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

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

(3) Frame base register (FB)

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

(4) Program counter (PC)

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

(5) Interrupt table register (INTB)

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

(6) Stack pointer (USP/ISP)

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

(7) Static base register (SB)

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

(8) Flag register (FLG)

Flag register (FLG) is configured with 11 bits, each bit is used as a flag. Figure 1.5.2 shows the flag register (FLG). The following explains the function of each flag:

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

- **Bit 7: Stack pointer select flag (U flag)**

  Interrupt stack pointer (ISP) is selected when this flag is “0” ; user stack pointer (USP) is selected when this flag is “1”.

  This flag is cleared to “0” when a hardware interrupt is acknowledged or an INT instruction of software interrupt Nos. 0 to 31 is executed.
- **Bits 8 to 11: Reserved area**
- **Bits 12 to 14: Processor interrupt priority level (IPL)**

  Processor interrupt priority level (IPL) is configured with three bits, for specification of up to eight processor interrupt priority levels from level 0 to level 7.

  If a requested interrupt has priority greater than the processor interrupt priority level (IPL), the interrupt is enabled.
- **Bit 15: Reserved area**

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

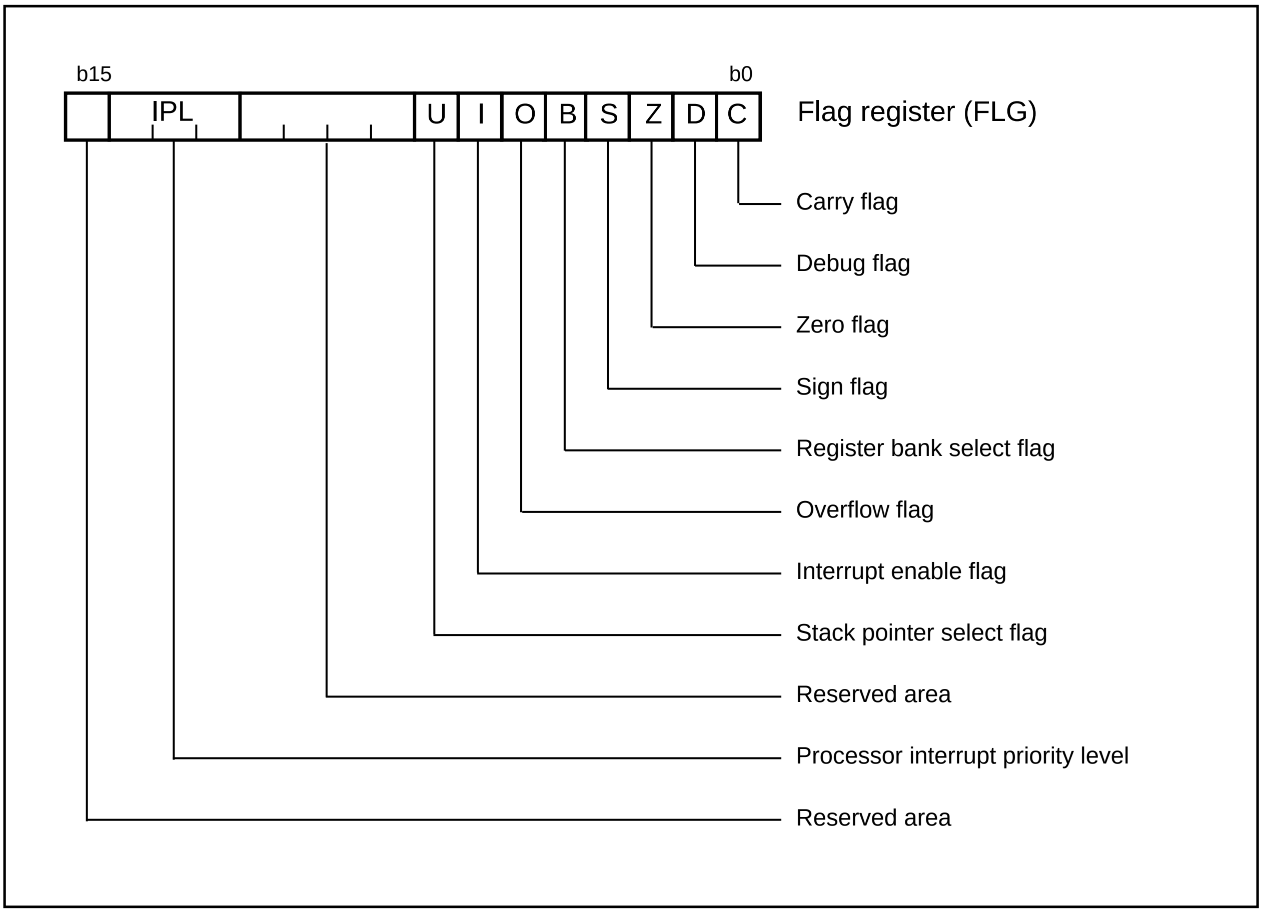


Figure 1.5.2. Flag register (FLG)

Reset

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

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

Figure 1.6.1 shows the example reset circuit. Figure 1.6.2 shows the reset sequence.

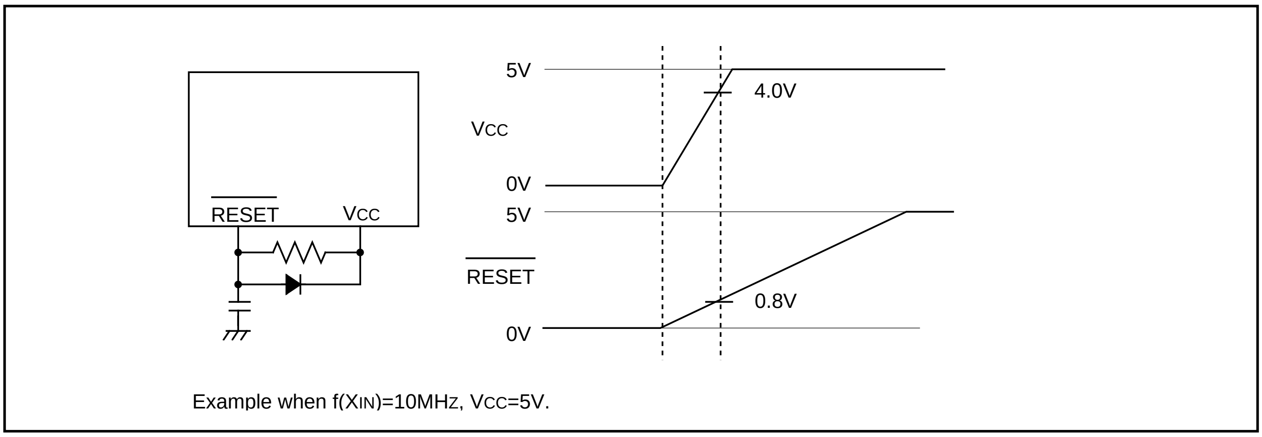


Figure 1.6.1. Example reset circuit

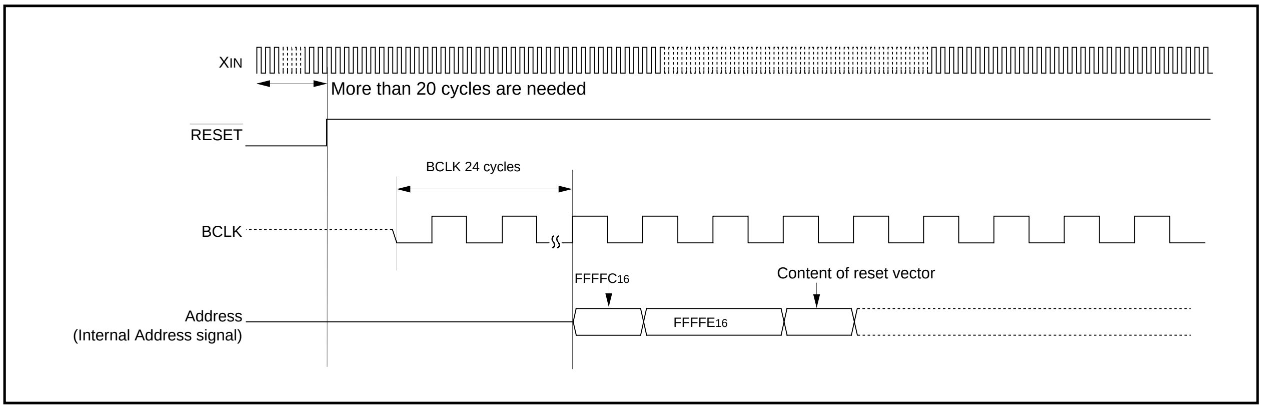


Figure 1.6.2. Reset sequence

Table 1.6.1 shows the statuses of the other pins while the $\overline{\text{RESET}}$ pin level is “L”. Figures 1.6.3 and 1.6.4 show the internal status of the microcomputer immediately after the reset is cancelled.

Table 1.6.1. Pin status when $\overline{\text{RESET}}$ pin level is “L”

Pin name	Status
P0, P10 to P12	Input port(with a pull up resistor)
P1 to P9, P13	Input port (floating)
SEG$_0$ to SEG$_{15}$	“H” level is output
COM$_0$ to COM$_3$	“H” level is output

(1)Processor mode register 0	(0004$_{16}$)•••	x	x	x	x	0	0	0	0
(2)Processor mode register 1	(0005$_{16}$)•••	0	x	x	x	x	x	0	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)Address match interrupt enable register	(0009$_{16}$)•••	x	x	x	x	x	x	0	0
(6)Protect register	(000A$_{16}$)•••	x	x	x	x	x	x	0	0
(7)Watchdog timer control register	(000F$_{16}$)•••	0	0	0	?	?	?	?	?
(8)Address match interrupt register 0	(0010$_{16}$)•••	00$_{16}$							
	(0011$_{16}$)•••	00$_{16}$							
	(0012$_{16}$)•••	x	x	x	x	0	0	0	0
(9)Address match interrupt register 1	(0014$_{16}$)•••	00$_{16}$							
	(0015$_{16}$)•••	00$_{16}$							
	(0016$_{16}$)•••	x	x	x	x	0	0	0	0
(10)DMA0 control register	(002C$_{16}$)•••	0	0	0	0	0	?	0	0
(11)DMA1 control register	(003C$_{16}$)•••	0	0	0	0	0	?	0	0
(12)$\overline{\text{INT3}}$ interrupt control register	(0044$_{16}$)•••	x	x	0	0	?	0	0	0
(13)Timer B5 interrupt control register	(0045$_{16}$)•••	x	x	x	x	?	0	0	0
(14)Timer B4 interrupt control register	(0046$_{16}$)•••	x	x	x	x	?	0	0	0
(15)Timer B3 interrupt control register	(0047$_{16}$)•••	x	x	x	x	?	0	0	0
(16)Timer A7 interrupt control register	(0048$_{16}$)•••	x	x	x	x	?	0	0	0
(17)Timer A6 interrupt control register	(0049$_{16}$)•••	x	x	x	x	?	0	0	0
(18)Timer A5 interrupt control register	(004A$_{16}$)•••	x	x	x	x	?	0	0	0
(19)DMA0 interrupt control register	(004B$_{16}$)•••	x	x	x	x	?	0	0	0
(20)DMA1 interrupt control register	(004C$_{16}$)•••	x	x	x	x	?	0	0	0
(21)Key input interrupt control register	(004D$_{16}$)•••	x	x	x	x	?	0	0	0
(22)A-D conversion interrupt control register	(004E$_{16}$)•••	x	x	x	x	?	0	0	0
(23)UART2 transmit interrupt control register	(004F$_{16}$)•••	x	x	x	x	?	0	0	0
(24)UART2 receive interrupt control register	(0050$_{16}$)•••	x	x	x	x	?	0	0	0
(25)UART0 transmit interrupt control register	(0051$_{16}$)•••	x	x	x	x	?	0	0	0
(26)UART0 receive interrupt control register	(0052$_{16}$)•••	x	x	x	x	?	0	0	0
(27)UART1 transmit interrupt control register	(0053$_{16}$)•••	x	x	x	x	?	0	0	0
(28)UART1 receive interrupt control register	(0054$_{16}$)•••	x	x	x	x	?	0	0	0
(29)Timer A0 interrupt control register	(0055$_{16}$)•••	x	x	x	x	?	0	0	0
(30)Timer A1 interrupt control register	(0056$_{16}$)•••	x	x	x	x	?	0	0	0
(31)Timer A2 interrupt control register	(0057$_{16}$)•••	x	x	x	x	?	0	0	0
(32)Timer A3 / $\overline{\text{INT4}}$ interrupt control register	(0058$_{16}$)•••	x	x	0	0	?	0	0	0
(33)Timer A4 / $\overline{\text{INT5}}$ interrupt control register	(0059$_{16}$)•••	x	x	0	0	?	0	0	0
(34)Timer B0 interrupt control register	(005A$_{16}$)•••	x	x	x	x	?	0	0	0
(35)Timer B1 interrupt control register	(005B$_{16}$)•••	x	x	x	x	?	0	0	0
(36)Timer B2 interrupt control register	(005C$_{16}$)•••	x	x	x	x	?	0	0	0
(37)$\overline{\text{INT0}}$ interrupt control register	(005D$_{16}$)•••	x	x	0	0	?	0	0	0
(38)$\overline{\text{INT1}}$ interrupt control register	(005E$_{16}$)•••	x	x	0	0	?	0	0	0
(39)$\overline{\text{INT2}}$ interrupt control register	(005F$_{16}$)•••	x	x	0	0	?	0	0	0
(40)LCD mode register	(0120$_{16}$)•••	0	x	0	0	0	0	0	0
(41)Segment output enable register	(0122$_{16}$)•••	0	0	0	0	0	0	0	0
(42)Key input mode register	(0126$_{16}$)•••	0	1	1	0	0	0	0	0
(43)Count start flag 1	(0340$_{16}$)•••	0	0	0	x	x	0	0	0
(44)One-shot start flag 1	(0342$_{16}$)•••	0	0	x	x	x	0	0	0
(45)Trigger select flag 1	(0343$_{16}$)•••	x	x	x	x	0	0	0	0
(46)Up-down flag 1	(0344$_{16}$)•••	x	x	0	x	x	0	0	0
(47)Timer A5 mode register	(0356$_{16}$)•••	00$_{16}$							
(48)Timer A6 mode register	(0357$_{16}$)•••	00$_{16}$							
(49)Timer A7 mode register	(0358$_{16}$)•••	00$_{16}$							
(50)Timer B3 mode register	(035B$_{16}$)•••	0	0	?	x	0	0	0	0
(51)Timer B4 mode register	(035C$_{16}$)•••	0	0	?	x	0	0	0	0
(52)Timer B5 mode register	(035D$_{16}$)•••	0	0	?	x	0	0	0	0
(53)Interrupt cause select register 0	(035E$_{16}$)•••	x	0	0	0	0	0	0	0
(54)Interrupt cause select register 1	(035F$_{16}$)•••	00$_{16}$							
(55)Clock division counter control register	(0360$_{16}$)•••	0	x	x	x	x	x	x	x
(56)UART2 special mode register 2	(0376$_{16}$)•••	00$_{16}$							
(57)UART2 special mode register	(0377$_{16}$)•••	00$_{16}$							
(58)UART2 transmit/receive mode register	(0378$_{16}$)•••	00$_{16}$							

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

x : Nothing is mapped to this bit
? : Undefined

Figure 1.6.3. Device's internal status after a reset is cleared

(59)UART2 transmit/receive control register 0	($037C_{16}$)· · ·	0 0 0 0 1 0 0 0
(60)UART2 transmit/receive control register 1	($037D_{16}$)· · ·	0 0 0 0 0 0 1 0
(61)Count start flag 0	(0380_{16})· · ·	00_{16}
(62) Clock prescaler reset flag	(0381_{16})· · ·	0 ☒ ☒ ☒ ☒ ☒ ☒ ☒
(63)One-shot start flag 0	(0382_{16})· · ·	0 0 ☒ 0 0 0 0 0
(64)Trigger select flag 0	(0383_{16})· · ·	00_{16}
(65)Up-down flag 0	(0384_{16})· · ·	00_{16}
(66)Timer A0 mode register	(0396_{16})· · ·	00_{16}
(67)Timer A1 mode register	(0397_{16})· · ·	00_{16}
(68)Timer A2 mode register	(0398_{16})· · ·	00_{16}
(69)Timer A3 mode register	(0399_{16})· · ·	00_{16}
(70)Timer A4 mode register	($039A_{16}$)· · ·	00_{16}
(71)Timer B0 mode register	($039B_{16}$)· · ·	0 0 ? ☒ 0 0 0 0
(72)Timer B1 mode register	($039C_{16}$)· · ·	0 0 ? ☒ 0 0 0 0
(73)Timer B2 mode register	($039D_{16}$)· · ·	0 0 ? ☒ 0 0 0 0
(74)UART0 transmit/receive mode register	($03A0_{16}$)· · ·	00_{16}
(75)UART0 transmit/receive control register 0	($03A4_{16}$)· · ·	0 0 0 0 1 0 0 0
(76)UART0 transmit/receive control register 1	($03A5_{16}$)· · ·	0 0 0 0 0 0 1 0
(77)UART1 transmit/receive mode register	($03A8_{16}$)· · ·	00_{16}
(78)UART1 transmit/receive control register 0	($03AC_{16}$)· · ·	0 0 0 0 1 0 0 0
(79)UART1 transmit/receive control register 1	($03AD_{16}$)· · ·	0 0 0 0 0 0 1 0
(80)UART transmit/receive control register 2	($03B0_{16}$)· · ·	☒ 0 0 0 0 0 0 0
(81)Flash memory control register (Note)	($03B4_{16}$)· · ·	☒ ☒ ☒ ☒ 0 ☒ 0 1
(82)DMA0 cause select register	($03B8_{16}$)· · ·	00_{16}
(83)DMA1 cause select register	($03BA_{16}$)· · ·	00_{16}
(84) A-D control register 2	($03D4_{16}$)· · ·	☒ ☒ ☒ ☒ 0 0 0 0

(85)A-D control register 0	($03D6_{16}$)· · ·	☒ ☒ 0 0 0 ? ? ?
(86)A-D control register 1	($03D7_{16}$)· · ·	00_{16}
(87)D-A control register	($03DC_{16}$)· · ·	☒ ☒ ☒ ☒ ☒ 0 0 0
(88)Port P0 direction register	($03E2_{16}$)· · ·	00_{16}
(89)Port P1 direction register	($03E3_{16}$)· · ·	00_{16}
(90)Port P2 direction register	($03E6_{16}$)· · ·	00_{16}
(91)Port P3 direction register	($03E7_{16}$)· · ·	☒ ☒ 0 0 0 0 0 0
(92)Port P4 direction register	($03EA_{16}$)· · ·	00_{16}
(93)Port P5 direction register	($03EB_{16}$)· · ·	00_{16}
(94)Port P6 direction register	($03EE_{16}$)· · ·	00_{16}
(95)Port P7 direction register	($03EF_{16}$)· · ·	☒ 0 0 0 0 0 0 0
(96)Port P8 direction register	($03F2_{16}$)· · ·	00_{16}
(97)Port P9 direction register	($03F3_{16}$)· · ·	00_{16}
(98)Port P10 direction register	($03F6_{16}$)· · ·	00_{16}
(99)Port P11 direction register	($03F7_{16}$)· · ·	00_{16}
(100)Port P12 direction register	($03FA_{16}$)· · ·	00_{16}
(101)Port P13 direction register	($03FB_{16}$)· · ·	☒ ☒ ☒ ☒ ☒ 0 0 0
(102)Pull-up control register 0	($03FC_{16}$)· · ·	0 0 0 0 0 0 1 1
(103)Pull-up control register 1	($03FD_{16}$)· · ·	00_{16}
(104)Pull-up control register 2	($03FE_{16}$)· · ·	1 1 1 1 0 0 0 0
(105)Real time port control register	($03FF_{16}$)· · ·	00_{16}
(106)Data registers (R0/R1/R2/R3)	· · ·	000_{16}
(107)Address registers (A0/A1)	· · ·	000_{16}
(108)Frame base register (FB)	· · ·	000_{16}
(109)Interrupt table register (INTB)	· · ·	0000_{16}
(110)User stack pointer (USP)	· · ·	000_{16}
(111)Interrupt stack pointer (ISP)	· · ·	000_{16}
(112)Static base register (SB)	· · ·	000_{16}
(113)Flag register (FLG)	· · ·	000_{16}

x : Nothing is mapped to this bit
? : Undefined

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

Note : This register is only exist in flash memory version.

Figure 1.6.4. Device's internal status after a reset is cleared

Address	Register
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}	
0009_{16}	Address match interrupt enable register (AIER)
$000A_{16}$	Protect register (PRCR)
$000B_{16}$	
$000C_{16}$	
$000D_{16}$	
$000E_{16}$	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)
$003D_{16}$	
$003E_{16}$	
$003F_{16}$	

Address	Register
0040_{16}	
0041_{16}	
0042_{16}	
0043_{16}	
0044_{16}	$\overline{\text{INT3}}$ 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}	Timer A7 interrupt control register (TA7IC)
0049_{16}	Timer A6 interrupt control register (TA6IC)
$004A_{16}$	Timer A5 interrupt control register (TA5IC) Bus collision detection interrupt control register (BCNIC)
$004B_{16}$	DMA0 interrupt control register (DM0IC)
$004C_{16}$	DMA1 interrupt control register (DM1IC)
$004D_{16}$	Key input interrupt control register (KUPIC)
$004E_{16}$	A-D conversion interrupt control register (ADIC)
$004F_{16}$	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) INT4 interrupt control register (INT4IC)
0059_{16}	Timer A4 interrupt control register (TA4IC) INT5 interrupt control register (INT5IC)
$005A_{16}$	Timer B0 interrupt control register (TB0IC)
$005B_{16}$	Timer B1 interrupt control register (TB1IC)
$005C_{16}$	Timer B2 interrupt control register (TB2IC)
$005D_{16}$	$\overline{\text{INT0}}$ interrupt control register (INT0IC)
$005E_{16}$	$\overline{\text{INT1}}$ interrupt control register (INT1IC)
$005F_{16}$	$\overline{\text{INT2}}$ interrupt control register (INT2IC)

Address	Register
0100_{16}	LCD RAM0(LRAM0)
0101_{16}	LCD RAM1(LRAM1)
0102_{16}	LCD RAM2(LRAM2)
0103_{16}	LCD RAM3(LRAM3)
0104_{16}	LCD RAM4(LRAM4)
0105_{16}	LCD RAM5(LRAM5)
0106_{16}	LCD RAM6(LRAM6)
0107_{16}	LCD RAM7(LRAM7)
0108_{16}	LCD RAM8(LRAM8)
0109_{16}	LCD RAM9(LRAM9)
$010A_{16}$	LCD RAM10(LRAM10)
$010B_{16}$	LCD RAM11(LRAM11)
$010C_{16}$	LCD RAM12(LRAM12)
$010D_{16}$	LCD RAM13(LRAM13)
$010E_{16}$	LCD RAM14(LRAM14)
$010F_{16}$	LCD RAM15(LRAM15)
0110_{16}	LCD RAM16(LRAM16)
0111_{16}	LCD RAM17(LRAM17)
0112_{16}	LCD RAM18(LRAM18)
0113_{16}	LCD RAM19(LRAM19)
0114_{16}	LCD RAM20(LRAM20)
0115_{16}	LCD RAM21(LRAM21)
0116_{16}	LCD RAM22(LRAM22)
0117_{16}	LCD RAM23(LRAM23)

Address	Register
0120_{16}	LCD mode register (LCDM)
0121_{16}	
0122_{16}	Segment output enable register (SEG)
0123_{16}	
0124_{16}	LCD frame frequency counter (LCDTIM)
0125_{16}	
0126_{16}	Key input mode register (KUPM)

Note : Locations in the SFR area where nothing is allocated are reserved areas. Do not access these areas for read or write.

Figure 1.7.1. Location of peripheral unit control registers (1)

0340$_{16}$	Count start flag 1 (TABSR1)	0380$_{16}$	Count start flag 0 (TABSR0)
0341$_{16}$		0381$_{16}$	Clock prescaler reset flag (CPSRF)
0342$_{16}$	One-shot start flag 1 (ONSF1)	0382$_{16}$	One-shot start flag 0 (ONSF0)
0343$_{16}$	Trigger select register 1 (TRGSR1)	0383$_{16}$	Trigger select register 0 (TRGSR0)
0344$_{16}$	Up-down flag 1(UDF1)	0384$_{16}$	Up-down flag 0 (UDF0)
0345$_{16}$		0385$_{16}$	
0346$_{16}$	Timer A5 register (TA5)	0386$_{16}$	Timer A0 register (TA0)
0347$_{16}$		0387$_{16}$	
0348$_{16}$	Timer A6 register (TA6)	0388$_{16}$	Timer A1 register (TA1)
0349$_{16}$		0389$_{16}$	
034A$_{16}$	Timer A7 register (TA7)	038A$_{16}$	Timer A2 register (TA2)
034B$_{16}$		038B$_{16}$	
034C$_{16}$		038C$_{16}$	Timer A3 register (TA3)
034D$_{16}$		038D$_{16}$	
034E$_{16}$		038E$_{16}$	Timer A4 register (TA4)
034F$_{16}$		038F$_{16}$	
0350$_{16}$	Timer B3 register (TB3)	0390$_{16}$	Timer B0 register (TB0)
0351$_{16}$		0391$_{16}$	
0352$_{16}$	Timer B4 register (TB4)	0392$_{16}$	Timer B1 register (TB1)
0353$_{16}$		0393$_{16}$	
0354$_{16}$	Timer B5 register (TB5)	0394$_{16}$	Timer B2 register (TB2)
0355$_{16}$		0395$_{16}$	
0356$_{16}$	Timer A5 mode register (TA5MR)	0396$_{16}$	Timer A0 mode register (TA0MR)
0357$_{16}$	Timer A6 mode register (TA6MR)	0397$_{16}$	Timer A1 mode register (TA1MR)
0358$_{16}$	Timer A7 mode register (TA7MR)	0398$_{16}$	Timer A2 mode register (TA2MR)
0359$_{16}$		0399$_{16}$	Timer A3 mode register (TA3MR)
035A$_{16}$		039A$_{16}$	Timer A4 mode register (TA4MR)
035B$_{16}$	Timer B3 mode register (TB3MR)	039B$_{16}$	Timer B0 mode register (TB0MR)
035C$_{16}$	Timer B4 mode register (TB4MR)	039C$_{16}$	Timer B1 mode register (TB1MR)
035D$_{16}$	Timer B5 mode register(TB5MR)	039D$_{16}$	Timer B2 mode register (TB2MR)
035E$_{16}$	Interrupt cause select register 0 (IFSR0)	039E$_{16}$	
035F$_{16}$	Interrupt cause select register 1 (IFSR1)	039F$_{16}$	
0360$_{16}$	Clock division counter control register (CDCC)	03A0$_{16}$	UART0 transmit/receive mode register (U0MR)
0361$_{16}$		03A1$_{16}$	UART0 bit rate generator (U0BRG)
0362$_{16}$		03A2$_{16}$	UART0 transmit buffer register (U0TB)
0363$_{16}$		03A3$_{16}$	
0364$_{16}$		03A4$_{16}$	UART0 transmit/receive control register 0 (U0C0)
0365$_{16}$		03A5$_{16}$	UART0 transmit/receive control register 1 (U0C1)
0366$_{16}$		03A6$_{16}$	UART0 receive buffer register (U0RB)
0367$_{16}$		03A7$_{16}$	
0368$_{16}$		03A8$_{16}$	UART1 transmit/receive mode register (U1MR)
0369$_{16}$		03A9$_{16}$	UART1 bit rate generator (U1BRG)
036A$_{16}$		03AA$_{16}$	UART1 transmit buffer register (U1TB)
036B$_{16}$		03AB$_{16}$	
036C$_{16}$		03AC$_{16}$	UART1 transmit/receive control register 0 (U1C0)
036D$_{16}$		03AD$_{16}$	UART1 transmit/receive control register 1 (U1C1)
036E$_{16}$	Clock division counter (CDC)	03AE$_{16}$	UART1 receive buffer register (U1RB)
036F$_{16}$		03AF$_{16}$	
0370$_{16}$		03B0$_{16}$	UART transmit/receive control register 2 (UCON)
0371$_{16}$		03B1$_{16}$	
0372$_{16}$		03B2$_{16}$	
0373$_{16}$		03B3$_{16}$	
0374$_{16}$		03B4$_{16}$	Flash memory control register (FMCR)(Note 1)
0375$_{16}$		03B5$_{16}$	
0376$_{16}$	UART2 special mode register 2(U2SMR2)	03B6$_{16}$	
0377$_{16}$	UART2 special mode register (U2SMR)	03B7$_{16}$	
0378$_{16}$	UART2 transmit/receive mode register (U2MR)	03B8$_{16}$	DMA0 request cause select register (DM0SL)
0379$_{16}$	UART2 bit rate generator (U2BRG)	03B9$_{16}$	
037A$_{16}$	UART2 transmit buffer register (U2TB)	03BA$_{16}$	DMA1 request cause select register (DM1SL)
037B$_{16}$		03BB$_{16}$	
037C$_{16}$	UART2 transmit/receive control register 0 (U2C0)	03BC$_{16}$	
037D$_{16}$	UART2 transmit/receive control register 1 (U2C1)	03BD$_{16}$	
037E$_{16}$	UART2 receive buffer register (U2RB)	03BE$_{16}$	
037F$_{16}$		03BF$_{16}$	

Note 1: This register is only exist in flash memory version.
Note 2: Locations in the SFR area where nothing is allocated are reserved areas. Do not access these areas for read or write.

Figure 1.7.2. Location of peripheral unit control registers (2)

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$_{16}$	D-A register 2 (DA2)
03DF$_{16}$	
03E0$_{16}$	Port P0 register (P0)
03E1$_{16}$	Port P1 register (P1)
03E2$_{16}$	Port P0 direction register (PD0)
03E3$_{16}$	Port P1 direction register (PD1)
03E4$_{16}$	Port P2 register (P2)
03E5$_{16}$	Port P3 register (P3)
03E6$_{16}$	Port P2 direction register (PD2)
03E7$_{16}$	Port P3 direction register (PD3)
03E8$_{16}$	Port P4 register (P4)
03E9$_{16}$	Port P5 register (P5)
03EA$_{16}$	Port P4 direction register (PD4)
03EB$_{16}$	Port P5 direction register (PD5)
03EC$_{16}$	Port P6 register (P6)
03ED$_{16}$	Port P7 register (P7)
03EE$_{16}$	Port P6 direction register (PD6)
03EF$_{16}$	Port P7 direction register (PD7)
03F0$_{16}$	Port P8 register (P8)
03F1$_{16}$	Port P9 register (P9)
03F2$_{16}$	Port P8 direction register (PD8)
03F3$_{16}$	Port P9 direction register (PD9)
03F4$_{16}$	Port P10 register (P10)
03F5$_{16}$	Port P11 register (P11)
03F6$_{16}$	Port P10 direction register (PD10)
03F7$_{16}$	Port P11 direction register (PD11)
03F8$_{16}$	Port P12 register (P12)
03F9$_{16}$	Port P13 register (P13)
03FA$_{16}$	Port P12 direction register (PD12)
03FB$_{16}$	Port P13 direction register (PD13)
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$_{16}$	Real time port control register (RTP)

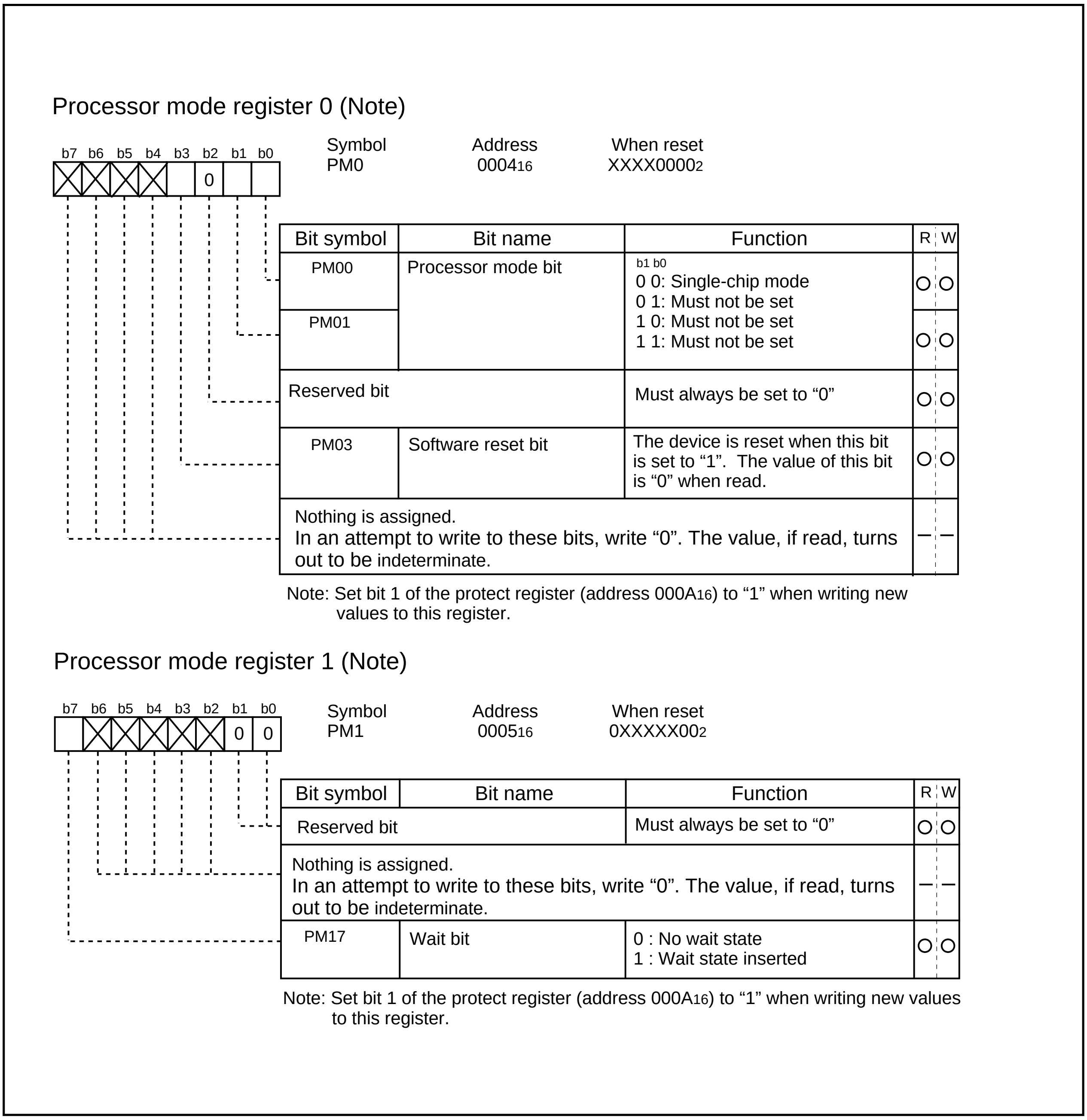
Note : Locations in the SFR area where nothing is allocated are reserved areas.
Do not access these areas for read or write.

Figure 1.7.3. Location of peripheral unit control registers (3)

Software Reset

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

Figure 1.8.1 shows the processor mode register 0 and 1.



Processor mode register 0 (Note)

b7	b6	b5	b4	b3	b2	b1	b0
☒	☒	☒	☒		0		

Symbol	Address	When reset
PM0	0004_{16}	$XXXX0000_2$

Bit symbol	Bit name	Function	R	W
PM00	Processor mode bit	b1 b0 0 0: Single-chip mode 0 1: Must not be set 1 0: Must not be set 1 1: Must not be set	○	○
PM01			○	○
Reserved bit		Must always be set to “0”	○	○
PM03	Software reset bit	The device is reset when this bit is set to “1”. The value of this bit is “0” when read.	○	○
Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminate.			–	–

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

Processor mode register 1 (Note)

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

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

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.			–	–
PM17	Wait bit	0 : No wait state 1 : Wait state inserted	○	○

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

Figure 1.8.1. Processor mode register 0 and 1

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)

A software wait is inserted in the internal ROM/RAM area. When set to “0”, each bus cycle is executed in one BCLK cycle. When set to “1”, each bus cycle is executed in two BCLK cycles. After the microcomputer has been reset, this bit defaults to “0”. Set this bit after referring to the recommended operating conditions (main clock input oscillation frequency) of the electric characteristics.

The SFR area is always accessed in two BCLK cycles regardless of the setting of this control bit.

Table 1.8.1 shows the software waits and bus cycles.

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

Table 1.8.1. Software waits and bus cycles

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

Clock Generating Circuit

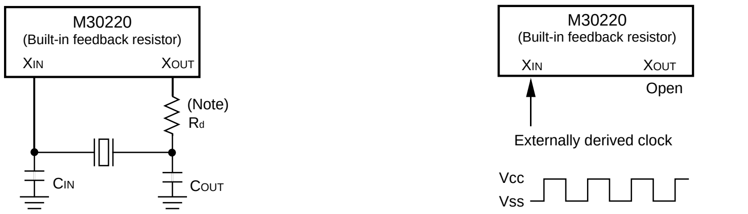
The clock generating circuit contains two oscillator circuits that supply the operating clock sources to the CPU and internal peripheral units.

Table 1.9.1. 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 • Intermittent pullup operation clock source of key input • LCD operation 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	

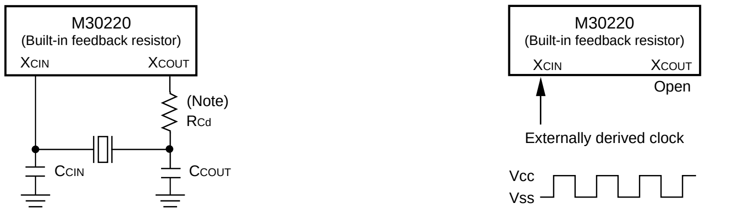
Example of oscillator circuit

Figure 1.9.1 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 1.9.2 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 1.9.1 and 1.9.2 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. Also, if the oscillator manufacturer's data sheet specifies that a feedback resistor be added external to the chip, insert a feedback resistor between XIN and XOUT following the instruction.

Figure 1.9.1. 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. Also, if the oscillator manufacturer's data sheet specifies that a feedback resistor be added external to the chip, insert a feedback resistor between XCIN and XCOUT following the instruction.

Figure 1.9.2. Examples of sub-clock

Clock Control

Figure 1.9.3 shows the block diagram of the clock generating circuit.

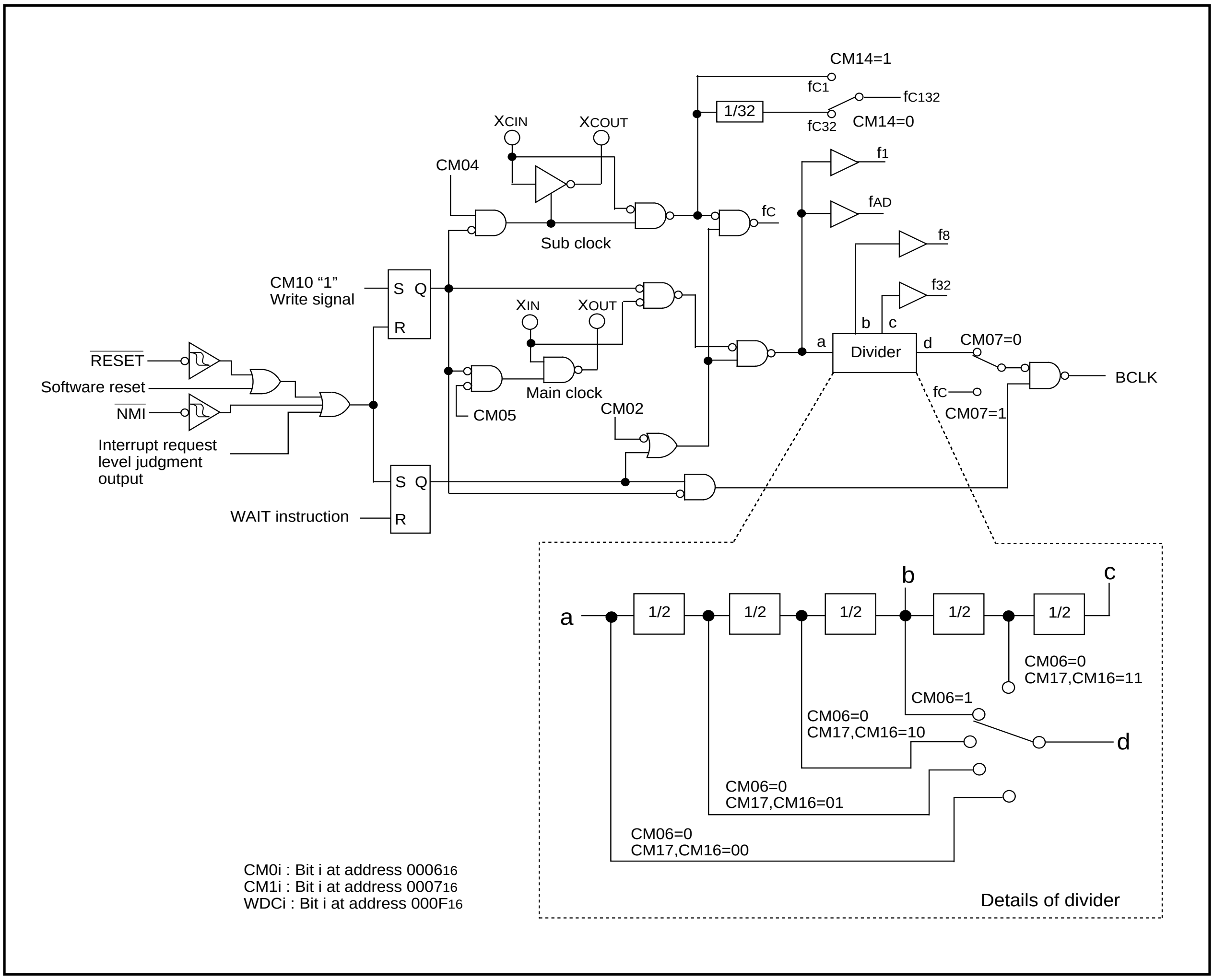


Figure 1.9.3. Clock generating circuit

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

(1) Main clock

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

(2) Sub-clock

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

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

(3) BCLK

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

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

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

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

(5) f_{C132}

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

(6) f_C

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

Figure 1.9.4 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_{C1} output 1 0 : f_1 output 1 1 : Clock divide counter output	○	○
CM01			○	○
CM02	WAIT peripheral function clock stop bit	0 : Do not stop peripheral function clock in wait mode 1 : Stop peripheral function clock in wait mode (Note 8)	○	○
CM03	X_{CIN}-X_{COUT} drive capacity select bit (Note 2)	0 : LOW 1 : HIGH	○	○
CM04	Sub clock (X_{CIN}-X_{COUT}) oscillation enable bit	0 : Off 1 : On	○	○
CM05	Main clock (X_{IN}-X_{OUT}) stop bit (Note 3, 4, 5)	0 : On 1 : Off	○	○
CM06	Main clock division select bit 0 (Note 7)	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 and at a reset.
Note 3: When entering power saving mode, main clock stops using this bit. When returning from stop mode and operating with 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 being connected, so X_{IN} turns pulled up to X_{OUT} ("H") via the feedback resistor.
Note 6: Sub clock (X_{CIN}-X_{COUT}) oscillation enable bit (CM04) to "1" and stabilize the sub-clock oscillating before setting to this bit from "0" to "1". Do not write to both bits at the same time. And also, set the main clock stop bit (CM05) to "0" and stabilize the main clock oscillating before setting this bit from "1" to "0".
Note 7: This bit changes to "1" when shifting from high-speed/medium-speed mode to stop mode and at a reset. When shifting from low-speed/low power dissipation mode to stop mode, the value before stop mode is retained.
Note 8: f_{C132}, f_{C1}, f_{C32} is not included. Do not set to "1" when using low-speed or low power dissipation mode.

System clock control register 1 (Note 1)

b7	b6	b5	b4	b3	b2	b1	b0
				0	0	0	

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

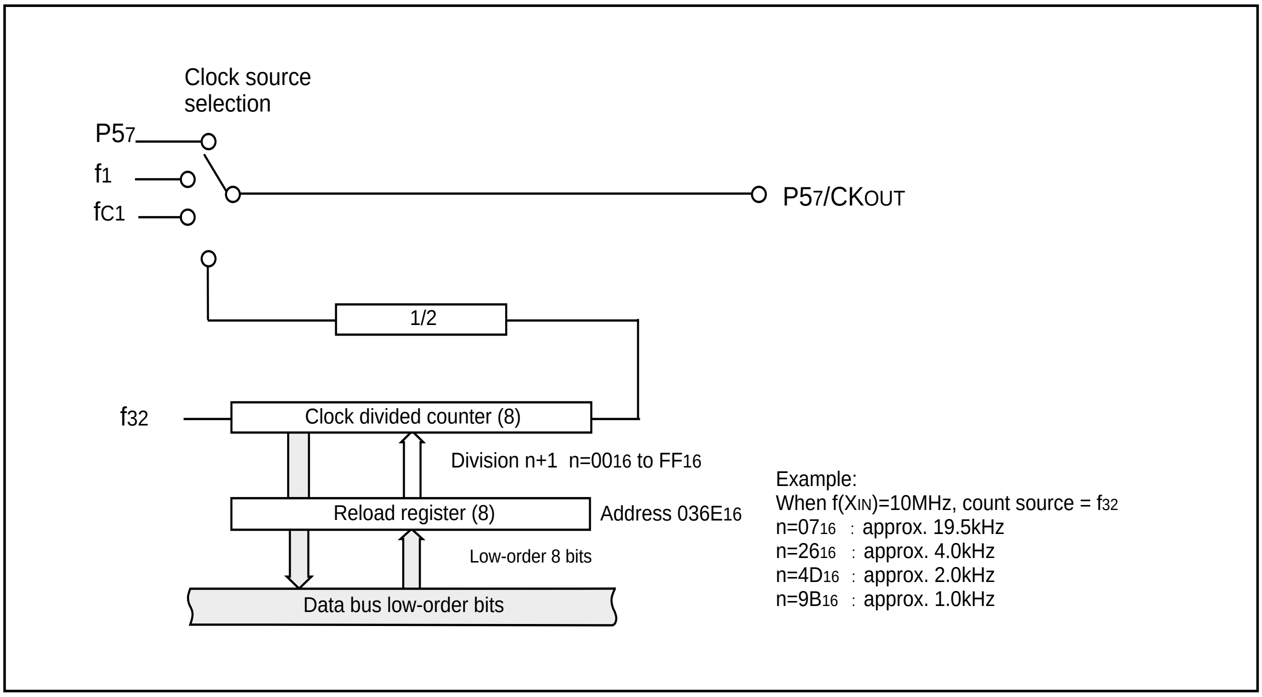
Bit symbol	Bit name	Function	R	W
CM10	All clock stop control bit (Note 4)	0 : Clock on 1 : All clocks off (stop mode)	○	○
Reserved bit		Must always be set to "0"	○	○
Reserved bit		Must always be set to "0"	○	○
Reserved bit		Must always be set to "0"	○	○
CM14	f_{C132} clock select bit	0 : f_{C32} 1 : f_{C1}	○	○
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: This bit changes to "1" when shifting from high-speed/medium-speed mode to stop mode and at a reset. When shifting from low-speed/low power dissipation mode to stop mode, the value before stop mode is retained.
Note 3: Can be selected when bit 6 of the system clock control register 0 (address 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 is cut off. X_{CIN} and X_{COUT} turn high-impedance state.

Figure 1.9.4. System clock control registers 0 and 1

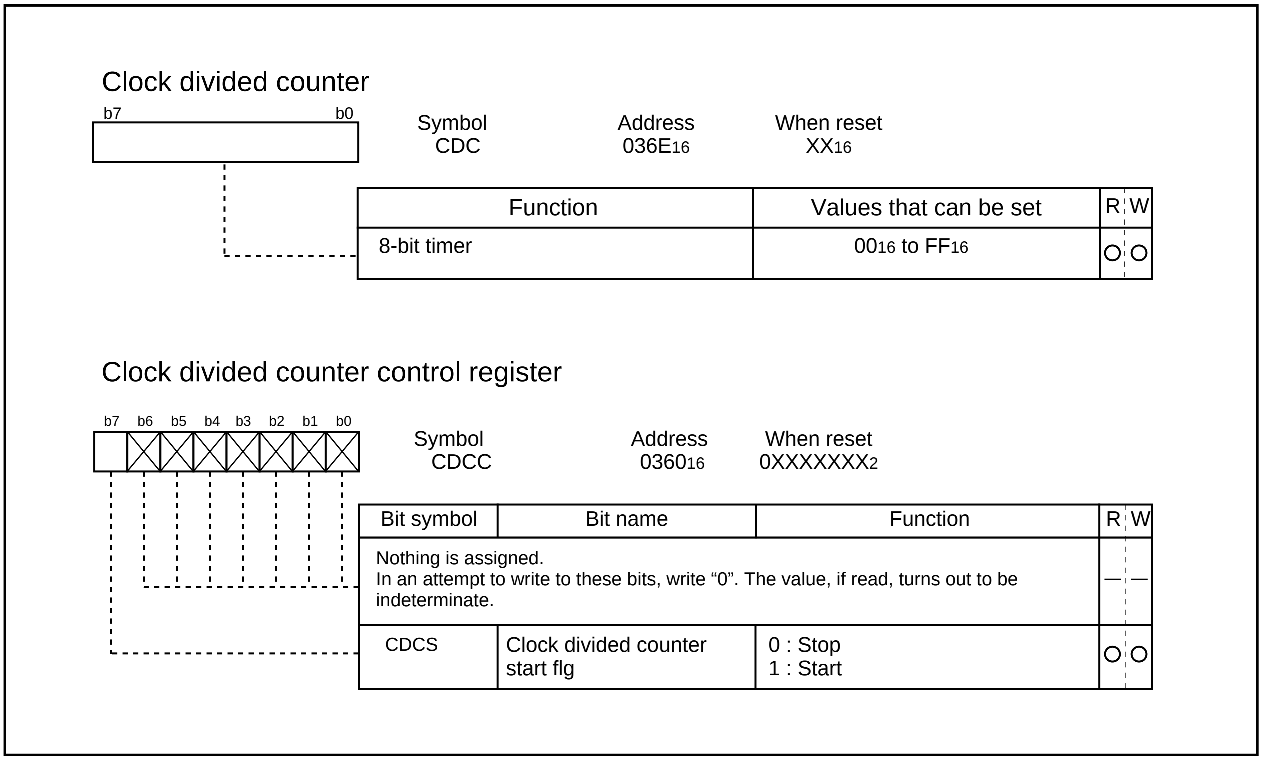
Clock Output

The clock output function select bit allows you to choose the clock from f_1, f_{C1}, or a divide-by-n clock that is output from the $P5_7/CK_{OUT}$ pin. The clock divide counter is an 8-bit counter whose count source is f_{32}, and its divide ratio can be set in the range of 00_{16} to FF_{16}. Also, the clock divided counter can be controlled for start or stop by the clock divide counter start flag. Figure 1.9.5 shows a block diagram of clock output. Figure 1.9.6 shows a clock divided counter related register.



Example:
When $f(X_{IN})$=10MHz, count source = f_{32}
n=07_{16} : approx. 19.5kHz
n=26_{16} : approx. 4.0kHz
n=$4D_{16}$: approx. 2.0kHz
n=$9B_{16}$: approx. 1.0kHz

Figure 1.9.5. Block diagram of clock output



Clock divided counter

Symbol CDC
Address $036E_{16}$
When reset XX_{16}

Function	Values that can be set	R	W
8-bit timer	00_{16} to FF_{16}	○	○

Clock divided counter control register

Symbol CDCC
Address 0360_{16}
When reset $0XXXXXXX_2$

Bit symbol	Bit name	Function	R	W
Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminate.			—	—
CDCS	Clock divided counter start flg	0 : Stop 1 : Start	○	○

Figure 1.9.6. Clock divided counter related register

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 , BCLK, f_1 to f_{32}, f_C, f_{C132}, f_{C1}, f_{C32} and f_{AD} stops in stop mode, peripheral functions such as the A-D converter and watchdog timer do not function. However, timer A and timer B operate provided that the event counter mode is set to an external pulse, and UART0 to UART2 functions provided an external clock is selected. Table 1.9.2 shows the status of the ports in stop mode.

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

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

Table 1.9.2. Port status during stop mode

Pin		Status
Port		Retains status before stop mode
CK_{OUT}	When f_{C1} selected	“H”
	When f_1, clock devided counter output selected	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 stop. Writing “1” to the WAIT peripheral function clock stop bit and executing a WAIT instruction stops the clock being supplied to the internal peripheral functions, allowing power dissipation to be reduced. However, peripheral function clock f_{C132}, f_{C1}, and f_{C32} do not stop so that the peripherals using f_{C132}, f_{C1}, and f_{C32} do not contribute to the power saving. When the MCU running in low-speed or low power dissipation mode, do not enter WAIT mode with this bit set to “1”. Table 1.9.3 shows the status of the ports in wait mode.

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

Table 1.9.3. Port status during wait mode

Pin		Status
Port		Retains status before wait mode
CK_{OUT}	When f_{C1} selected	Does not stop
	When f_1, clock devided counter output selected	Retains status before stop mode Does not stop when the WAIT peripheral function clock stop bit is “0”. When the WAIT peripheral function clock stop bit is “1”, the status immediately prior to entering wait mode is main-tained.

Status Transition Of BCLK

Power dissipation can be reduced and low-voltage operation achieved by changing the count source for BCLK. Table 1.9.4 shows the operating modes corresponding to the settings of system clock control registers 0 and 1.

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

The following shows the operational modes of BCLK.

(1) Division by 2 mode

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

(2) Division by 4 mode

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

(3) Division by 8 mode

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

(4) Division by 16 mode

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

(5) No-division mode

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

(6) Low-speed mode

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

(7) Low power dissipation mode

f_C is the BCLK and the main clock is stopped.

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

Table 1.9.4. Operating modes dictated by settings of system clock control registers 0 and 1

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

Power control

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

Modes

Power control is available in three modes.

(a) Normal operation mode

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

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

- **Low-speed mode**
  fC becomes the BCLK. The CPU operates according to the fc clock. The fc clock is supplied by the secondary clock. Each peripheral function operates according to its assigned clock.

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

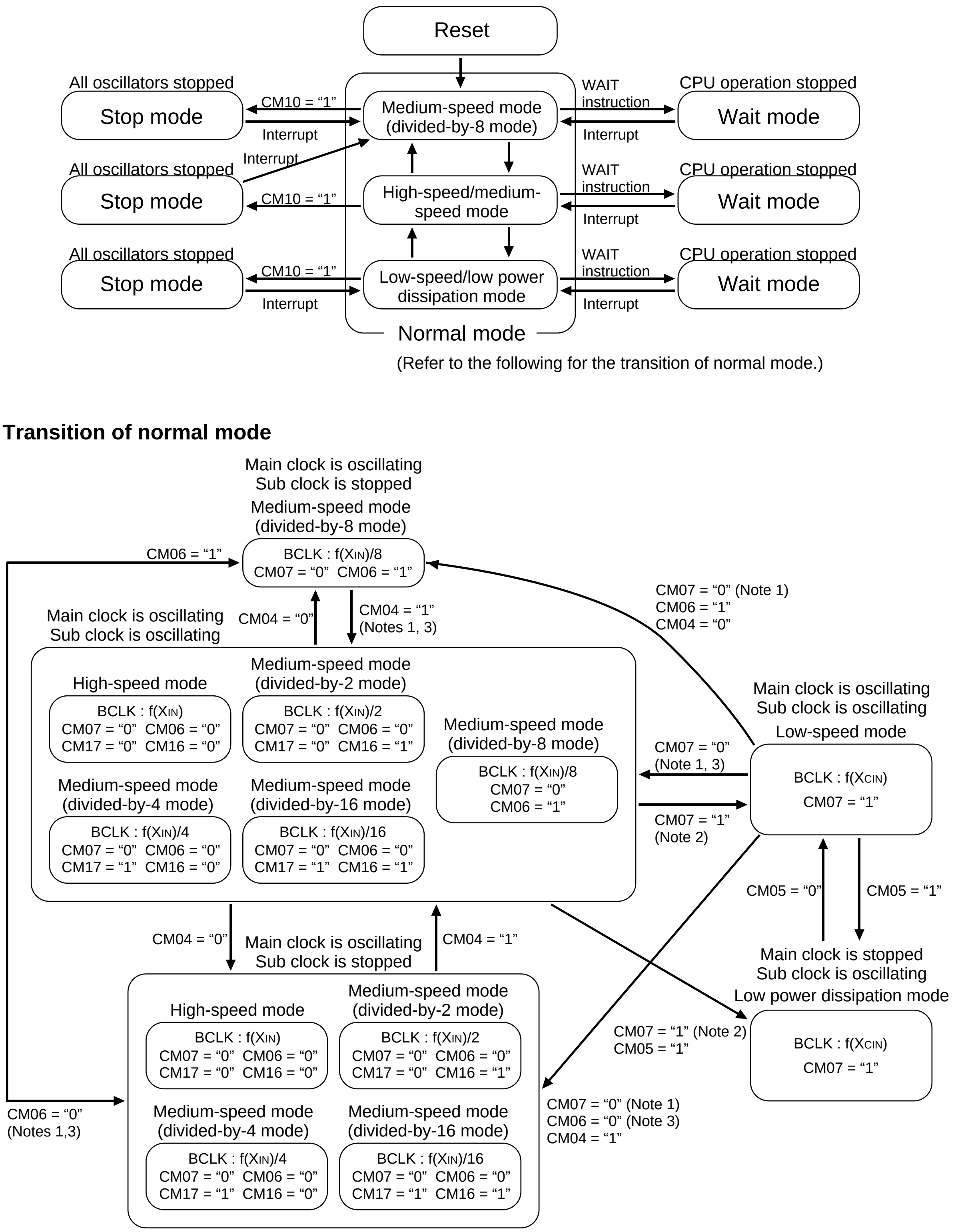
(b) Wait mode

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

(c) Stop mode

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

Figure 1.9.7 is the state transition diagram of the above modes.



Transition of stop mode, wait mode

Transition of normal mode

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

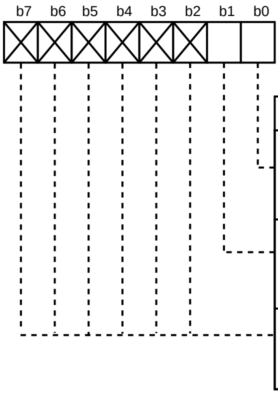
Figure 1.9.7. State transition diagram of Power control mode

Protection

The protection function is provided so that the values in important registers cannot be changed in the event that the program runs out of control. Figure 1.9.8 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$_{16}$) can only be changed when the respective bit in the protect register is set to “1”.

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

Protect register



Symbol	Address	When reset
PRCR	000A$_{16}$	XXXXXX00$_2$

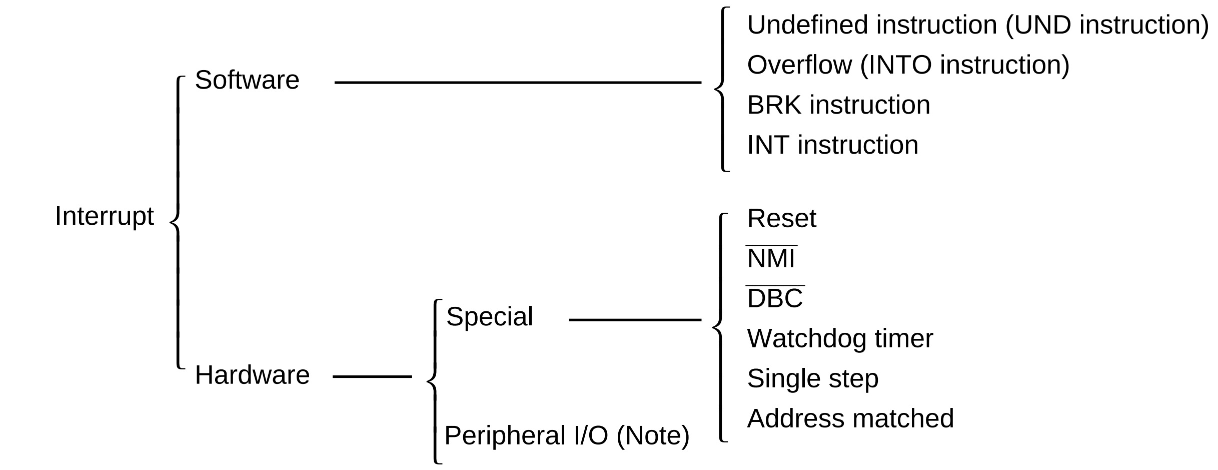
Bit symbol	Bit name	Function	R	W
PRC0	Enables writing to system clock control registers 0 and 1 (addresses 0006$_{16}$ and 0007$_{16}$)	0 : Write-inhibited 1 : Write-enabled	○	○
PRC1	Enables writing to processor mode registers 0 and 1 (addresses 0004$_{16}$ and 0005$_{16}$)	0 : Write-inhibited 1 : Write-enabled	○	○
Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminate.			—	—

Figure 1.9.8. Protect register

Overview of Interrupt

Type of Interrupts

Figure 1.10.1 lists the types of interrupts.



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

Figure 1.10.1. 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 specifying one of software interrupt numbers 0 through 63 and executing the INT instruction. Software interrupt numbers 0 through 31 are assigned to peripheral I/O interrupts, so executing the INT instruction allows executing the same interrupt routine that a peripheral I/O interrupt does.
  The stack pointer (SP) used for the INT interrupt is dependent on which software interrupt number is involved.
  So far as software interrupt numbers 0 through 31 are concerned, the microcomputer saves the stack pointer assignment flag (U flag) when it accepts an interrupt request. If change the U flag to “0” and select the interrupt stack pointer (ISP), and then execute an interrupt sequence. When returning from the interrupt routine, the U flag is returned to the state it was before the acceptance of interrupt request. So far as software numbers 32 through 63 are concerned, the stack pointer does not make a shift.

Hardware Interrupts

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

(1) Special interrupts

Special interrupts are non-maskable interrupts.

- **Reset**

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

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

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

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

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

- **Watchdog timer interrupt**

  Generated by the watchdog timer.

- **Single-step interrupt**

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

- **Address match interrupt**

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

(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 either a falling edge or a both edge 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 transmission interrupt**

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

- **UART0, UART1, UART2 reception interrupt**

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

- **Timer A0 interrupt through timer A7 interrupt**

  These are interrupts that timer A generates

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

  These are interrupts that timer B generates.

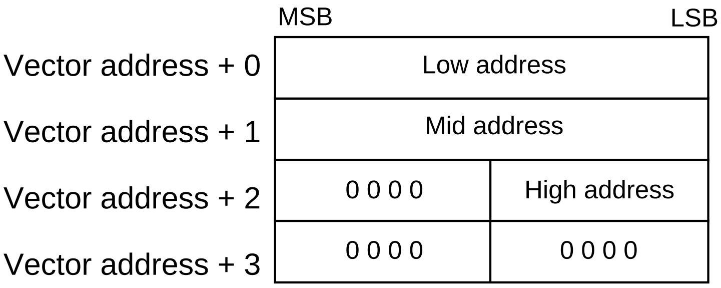
- **$\overline{\text{INT0}}$ interrupt through $\overline{\text{INT5}}$ interrupt**

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

Interrupts and Interrupt Vector Tables

If an interrupt request is accepted, a program branches to the interrupt routine set in the interrupt vector table. Set the first address of the interrupt routine in each vector table. Figure 1.10.2 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 1.10.2. 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 1.10.1 shows the interrupts assigned to the fixed vector tables and addresses of vector tables.

Table 1.10.1. 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	FFFF0$_{16}$ to FFFF3$_{16}$	
$\overline{\text{DBC}}$ (Note)	FFFF4$_{16}$ to FFFF7$_{16}$	Do not use
$\overline{\text{NMI}}$	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 settings. Indicate the first address using the interrupt table register (INTB). The 256-byte area subsequent to the address the INTB indicates becomes the area for the variable vector tables. One vector table comprises four bytes. Set the first address of the interrupt routine in each vector table. Table 1.10.2 shows the interrupts assigned to the variable vector tables and addresses of vector tables.

Table 1.10.2. Interrupts assigned to the variable vector tables and addresses of vector tables

Software interrupt number	Vector table address Address (L) to address (H)	Interrupt source	Remarks
Software interrupt number 0	+0 to +3 (Note 1)	BRK instruction	Cannot be masked I flag
Software interrupt number 4	+16 to +19 (Note 1)	$\overline{\text{INT3}}$	
Software interrupt number 5	+20 to +23 (Note 1)	Timer B5	
Software interrupt number 6	+24 to +27 (Note 1)	Timer B4	
Software interrupt number 7	+28 to +31 (Note 1)	Timer B3	
Software interrupt number 8	+32 to +35 (Note 1)	Timer A7	
Software interrupt number 9	+36 to +39 (Note 1)	Timer A6	
Software interrupt number 10	+40 to +43 (Note 1)	Timer A5/Bus collision detection (Note 2)	
Software interrupt number 11	+44 to +47 (Note 1)	DMA0	
Software interrupt number 12	+48 to +51 (Note 1)	DMA1	
Software interrupt number 13	+52 to +55 (Note 1)	Key input interrupt	
Software interrupt number 14	+56 to +59 (Note 1)	A-D	
Software interrupt number 15	+60 to +63 (Note 1)	UART2 transmit / NACK (Note 3)	
Software interrupt number 16	+64 to +67 (Note 1)	UART2 receive / ACK (Note 3)	
Software interrupt number 17	+68 to +71 (Note 1)	UART0 transmit	
Software interrupt number 18	+72 to +75 (Note 1)	UART0 receive	
Software interrupt number 19	+76 to +79 (Note 1)	UART1 transmit	
Software interrupt number 20	+80 to +83 (Note 1)	UART1 receive	
Software interrupt number 21	+84 to +87 (Note 1)	Timer A0	
Software interrupt number 22	+88 to +91 (Note 1)	Timer A1	
Software interrupt number 23	+92 to +95 (Note 1)	Timer A2	
Software interrupt number 24	+96 to +99 (Note 1)	Timer A3/$\overline{\text{INT4}}$ (Note 4)	
Software interrupt number 25	+100 to +103 (Note 1)	Timer A4/$\overline{\text{INT5}}$ (Note 4)	
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 bit (bit 4 in address $035E_{16}$).
Note 3: When IIC mode is selected, NACK and ACK interrupts are selected.
Note 4: It is selected by interrupt request cause select bit (bit 6, 7 in address $035F_{16}$).

Interrupt Control

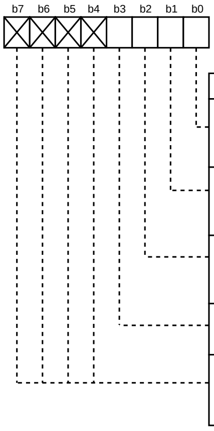
Descriptions are given here regarding how to enable or disable maskable interrupts and how to set the priority to be accepted. What is described here does not apply to non-maskable interrupts.

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

Figure 1.10.3 shows the memory map of the interrupt control registers.

Interrupt control register (Note2)

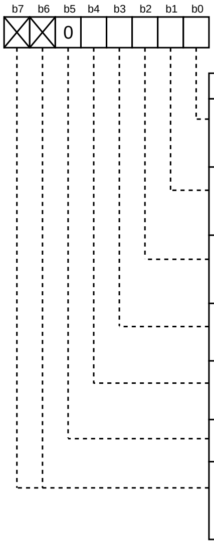
Symbol	Address	When reset
TBiIC(i=3 to 5)	0045_{16} to 0047_{16}	$XXXXX000_2$
TAiIC(i=6, 7)	0048_{16}, 0049_{16}	$XXXXX000_2$
TA5IC/BCNIC	$004A_{16}$	$XXXXX000_2$
DMiIC(i=0, 1)	$004B_{16}$, $004C_{16}$	$XXXXX000_2$
KUPIC	$004D_{16}$	$XXXXX000_2$
ADIC	$004E_{16}$	$XXXXX000_2$
SiTIC(i=0 to 2)	0051_{16}, 0053_{16}, $004F_{16}$	$XXXXX000_2$
SiRIC(i=0 to 2)	0052_{16}, 0054_{16}, 0050_{16}	$XXXXX000_2$
TAiIC(i=0 to 2)	0055_{16} to 0057_{16}	$XXXXX000_2$
TBiIC(i=0 to 2)	$005A_{16}$ to $005C_{16}$	$XXXXX000_2$



Bit symbol	Bit name	Function	R	W
ILVL0	Interrupt priority level select bit	b2 b1 b0 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	○	○ (Note 1)
Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminate.			–	–

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=0 to 2)	$005D_{16}$ to $005F_{16}$	$XX00X000_2$
(i=3)	0044_{16}	$XX00X000_2$
TAiIC/INTjIC(i=3, 4)	0058_{16}, 0059_{16}	$XX00X000_2$
(j=4, 5)	0058_{16}, 0059_{16}	$XX00X000_2$



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	○	○ (Note 1)
POL	Polarity select bit	0 : Selects falling edge 1 : Selects rising edge	○	○
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.			–	–

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

Figure 1.10.3. 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 it to “0” disables all maskable interrupts. This flag is set to “0” after reset.

Interrupt Request Bit

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

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

Set the interrupt priority level using the interrupt priority level select bit, which is one of the component bits of the interrupt control register. When an interrupt request occurs, the interrupt priority level is compared with the IPL. The interrupt is enabled only when the priority level of the interrupt is higher than the IPL. Therefore, setting the interrupt priority level to “0” disables the interrupt.
Table 1.10.3 shows the settings of interrupt priority levels and Table 1.10.4 shows the interrupt levels enabled, according to the contents of the IPL.

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

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

Table 1.10.3. 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 1.10.4. 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 are inserted before FSET I in Examples 1 and 2 is to prevent the interrupt enable flag I from being set before the interrupt control register is rewritten due to effects of the instruction queue.

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

Instructions : AND, OR, BCLR, BSET

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 0000_{16}. After this, the corresponding interrupt request bit becomes “0”.

(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.

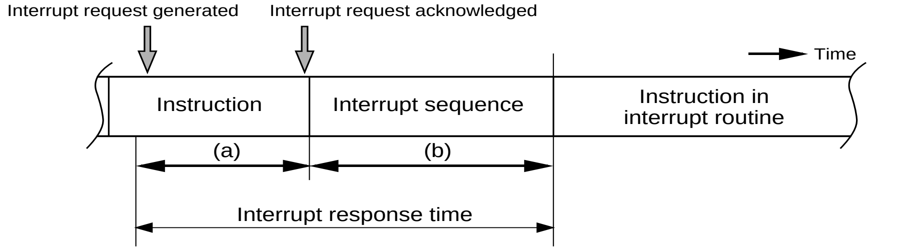
(6) Sets the interrupt priority level of the accepted instruction in the IPL.

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 occurrence of an interrupt to the completion of the instruction under execution at that moment (a) and the time required for executing the interrupt sequence (b). Figure 1.10.4 shows the interrupt response time.


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

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

Figure 1.10.4. Interrupt response time

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

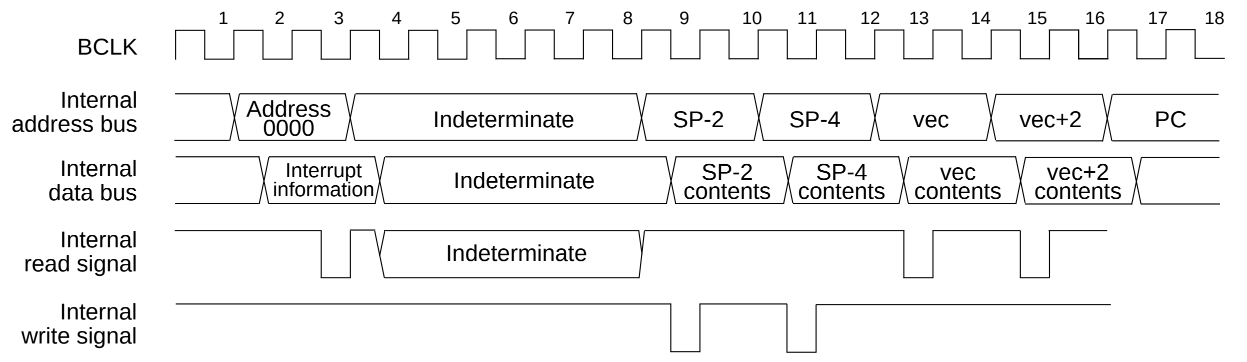
Time (b) is as shown in Table 1.10.5.

Table 1.10.5. 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.



The indeterminate segment is dependent on the queue buffer.
If the queue buffer is ready to take an instruction, a read cycle occurs.

Figure 1.10.5. 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 1.10.6 is set in the IPL.

Table 1.10.6. Relationship between interrupts without interrupt priority levels and IPL

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

Saving Registers

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

First, the processor saves the four higher-order bits of the program counter, and 4 upper-order bits and 8 lower-order bits of the FLG register, 16 bits in total, in the stack area, then saves 16 lower-order bits of the program counter. Figure 1.10.6 shows the state of the stack as it was before the acceptance of the interrupt request, and the state the stack after the acceptance of the interrupt request.

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

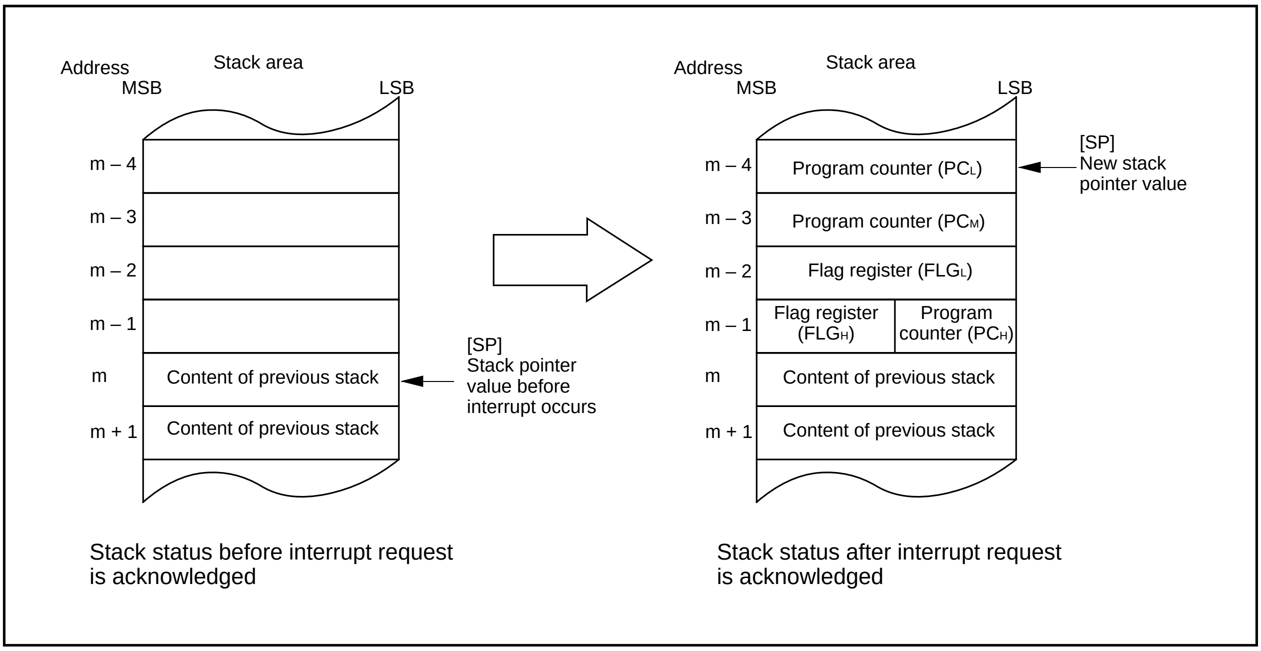
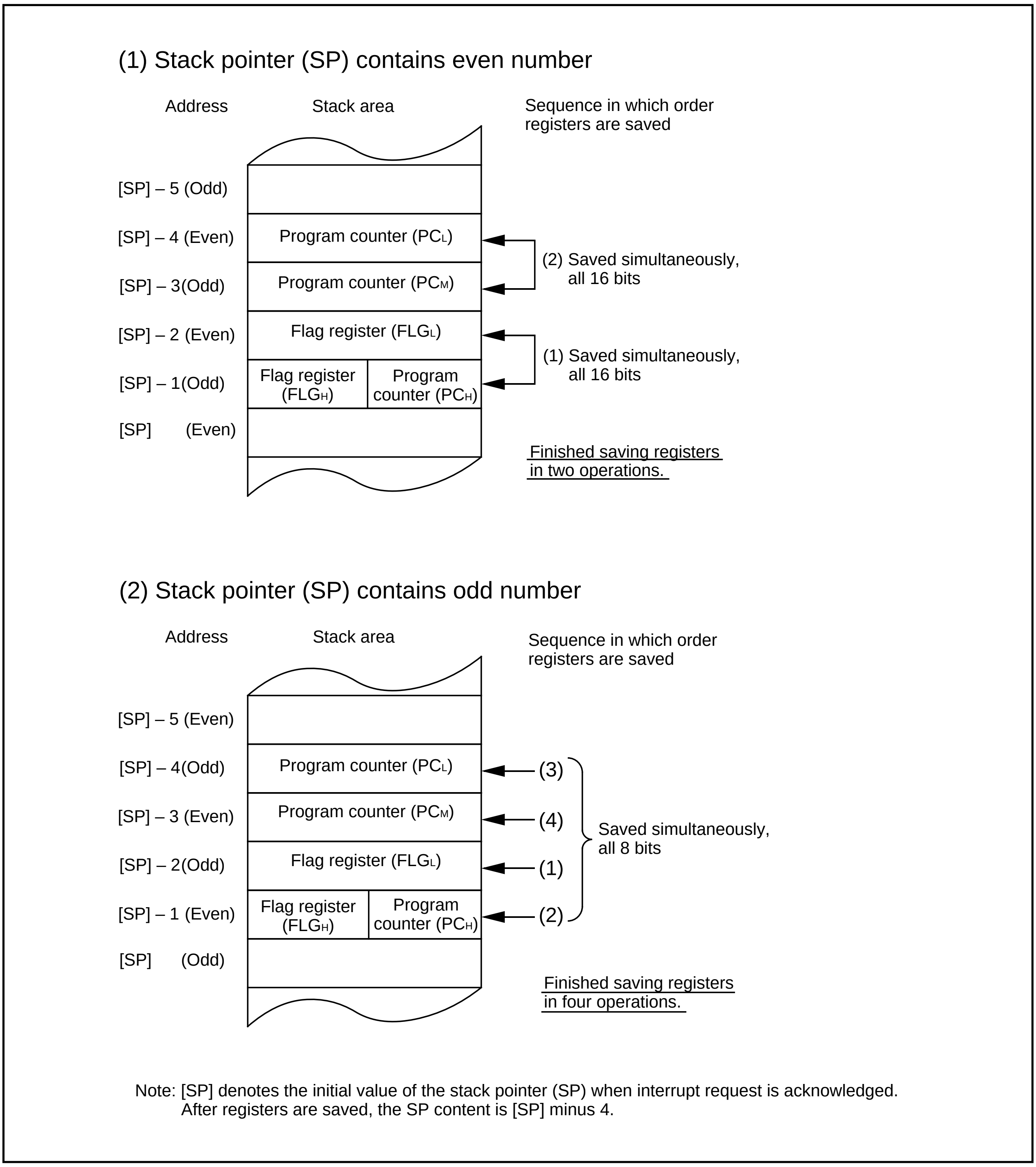


Figure 1.10.6. State of stack before and after acceptance of interrupt request

The operation of saving registers carried out in the interrupt sequence is dependent on whether the content of the stack pointer, at the time of acceptance of an interrupt request, is even or odd. If the content of the stack pointer (Note) is even, the content of the flag register (FLG) and the content of the program counter (PC) are saved, 16 bits at a time. If odd, their contents are saved in two steps, 8 bits at a time. Figure 1.10.7 shows the operation of the saving registers.

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


(1) Stack pointer (SP) contains even number

(2) Stack pointer (SP) contains odd number

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

Figure 1.10.7. 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 suspended process resumes.

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

Interrupt Priority

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

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

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

Figure 1.10.8 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 1.10.8. Hardware interrupts priorities

Interrupt resolution circuit

When two or more interrupts are generated simultaneously, this circuit selects the interrupt with the highest priority level. Figure 1.10.9 shows the circuit that judges the interrupt priority level.

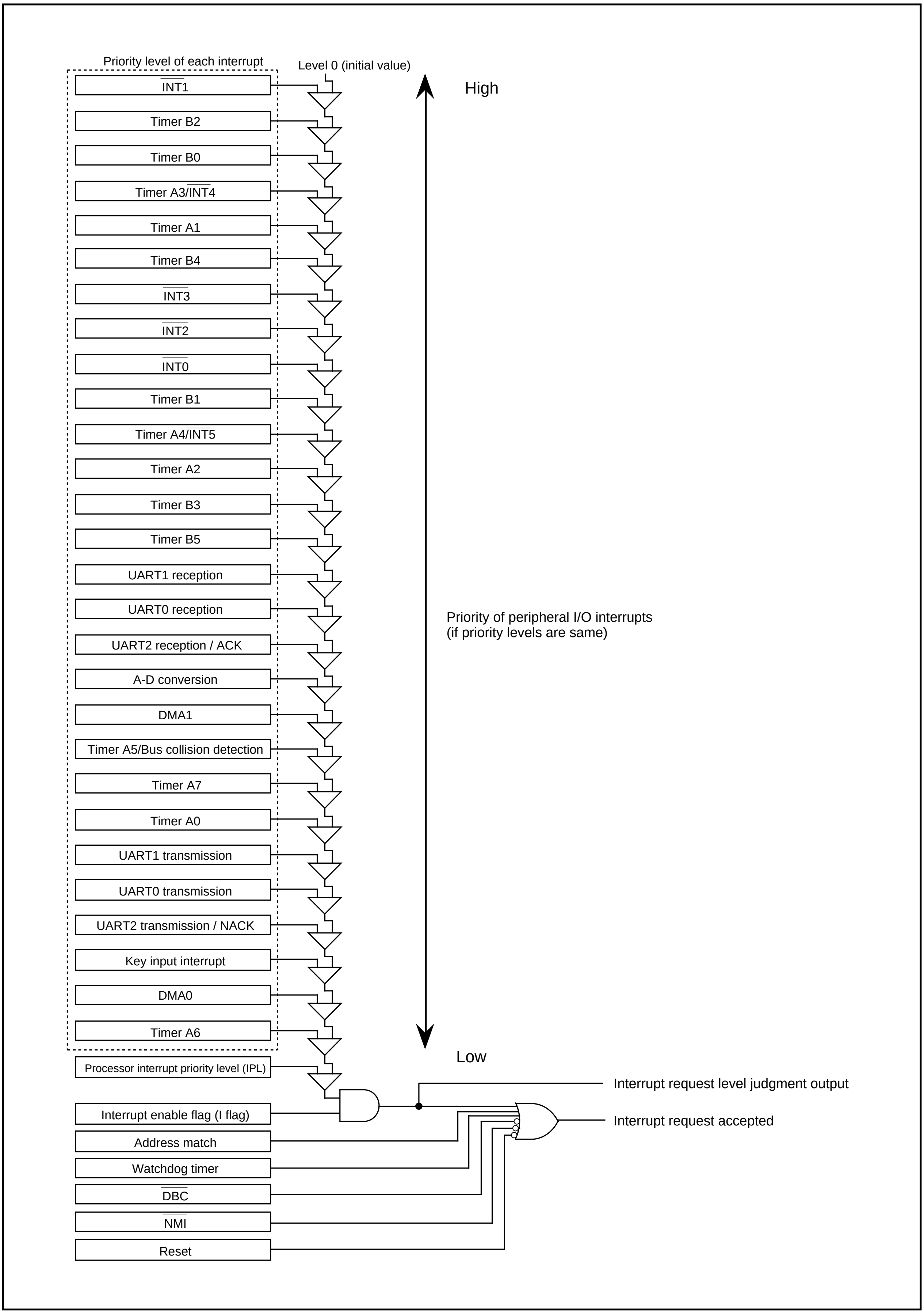


Figure 1.10.9. Maskable interrupts priorities (peripheral I/O interrupts)

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

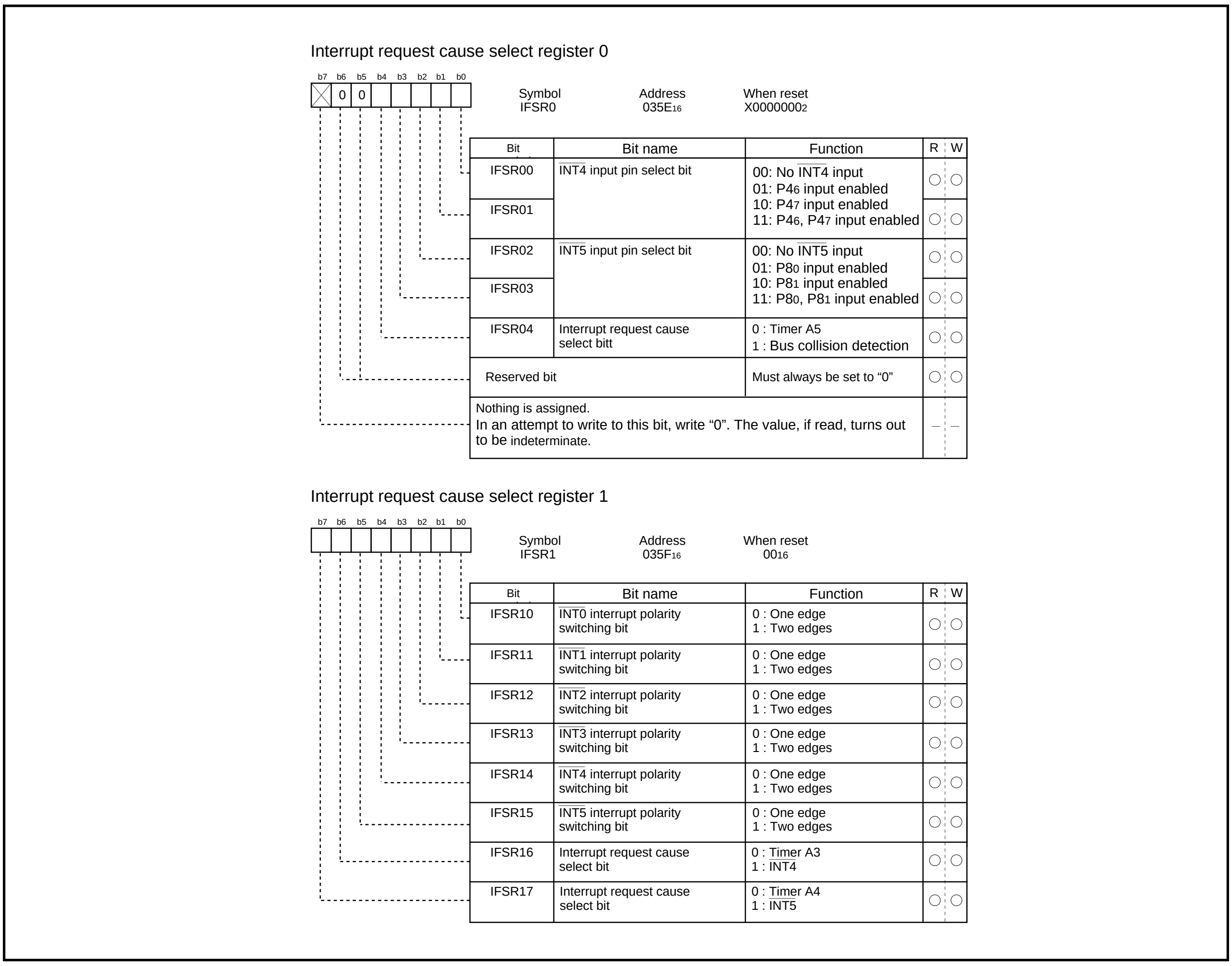
$\overline{\text{INT0}}$ to $\overline{\text{INT5}}$ are triggered by the edges of external inputs. The edge polarity is selected using the polarity select bit. Of interrupt control registers, 0058$_{16}$ is used both as timer A3 and external interrupt $\overline{\text{INT4}}$ input control register, and 0059$_{16}$ is used both as timer A4 and as external interrupt $\overline{\text{INT5}}$ input control register. Use the interrupt request cause select bits - bits 6 and 7 of the interrupt request cause select register 1 (address 035F$_{16}$) - to specify which interrupt request cause to select. When $\overline{\text{INT4}}$ is selected as an interrupt source, the input port for it can be selected by bits 0 and 1 of the interrupt source select register 0 (address 035E$_{16}$). Similarly, when $\overline{\text{INT5}}$ is selected as an interrupt source, the input port for it can be selected by bits 2 and 3 of the interrupt source select register 0 (address 035E$_{16}$). After having set an interrupt request cause and interrupt input ports, be sure to set the corresponding interrupt request bit to “0” before enabling an interrupt.

Either of the interrupt control registers - 0058$_{16}$, 0059$_{16}$ - has the polarity-switching bit. Be sure to set this bit to “0” to select an timer as the interrupt request cause.

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

When $\overline{\text{INT4}}$ input pin select bits = “11”, $\overline{\text{INT4}}$ interrupt polarity switching bit = “0”, and polarity select bit = “1” of the $\overline{\text{INT4}}$ interrupt control register, an interrupt is generated by a rising edge on the input port when the exclusive pin is “H”, as shown by “Single edge, Rise” in Figure 1.10.12. When the exclusive pin is “H”, interrupts can only be generated by an active transition on a single edge. The same applies to $\overline{\text{INT5}}$.

Figure 1.10.10 shows the interrupt request cause select register.



Interrupt request cause select register 0

Symbol IFSR0 | Address 035E$_{16}$ | When reset X0000000$_2$

Bit	Bit name	Function	R	W
IFSR00	$\overline{\text{INT4}}$ input pin select bit	00: No $\overline{\text{INT4}}$ input 01: P4$_6$ input enabled 10: P4$_7$ input enabled 11: P4$_6$, P4$_7$ input enabled	○	○
IFSR01			○	○
IFSR02	$\overline{\text{INT5}}$ input pin select bit	00: No $\overline{\text{INT5}}$ input 01: P8$_0$ input enabled 10: P8$_1$ input enabled 11: P8$_0$, P8$_1$ input enabled	○	○
IFSR03			○	○
IFSR04	Interrupt request cause select bitt	0 : Timer A5 1 : Bus collision detection	○	○
Reserved bit		Must always be set to “0”	○	○
Nothing is assigned. In an attempt to write to this bit, write “0”. The value, if read, turns out to be indeterminate.			–	–

Interrupt request cause select register 1

Symbol IFSR1 | Address 035F$_{16}$ | When reset 00$_{16}$

Bit	Bit name	Function	R	W
IFSR10	$\overline{\text{INT0}}$ interrupt polarity switching bit	0 : One edge 1 : Two edges	○	○
IFSR11	$\overline{\text{INT1}}$ interrupt polarity switching bit	0 : One edge 1 : Two edges	○	○
IFSR12	$\overline{\text{INT2}}$ interrupt polarity switching bit	0 : One edge 1 : Two edges	○	○
IFSR13	$\overline{\text{INT3}}$ interrupt polarity switching bit	0 : One edge 1 : Two edges	○	○
IFSR14	$\overline{\text{INT4}}$ interrupt polarity switching bit	0 : One edge 1 : Two edges	○	○
IFSR15	$\overline{\text{INT5}}$ interrupt polarity switching bit	0 : One edge 1 : Two edges	○	○
IFSR16	Interrupt request cause select bit	0 : Timer A3 1 : $\overline{\text{INT4}}$	○	○
IFSR17	Interrupt request cause select bit	0 : Timer A4 1 : $\overline{\text{INT5}}$	○	○

Figure 1.10.10. Interrupt request cause select registers 0, 1

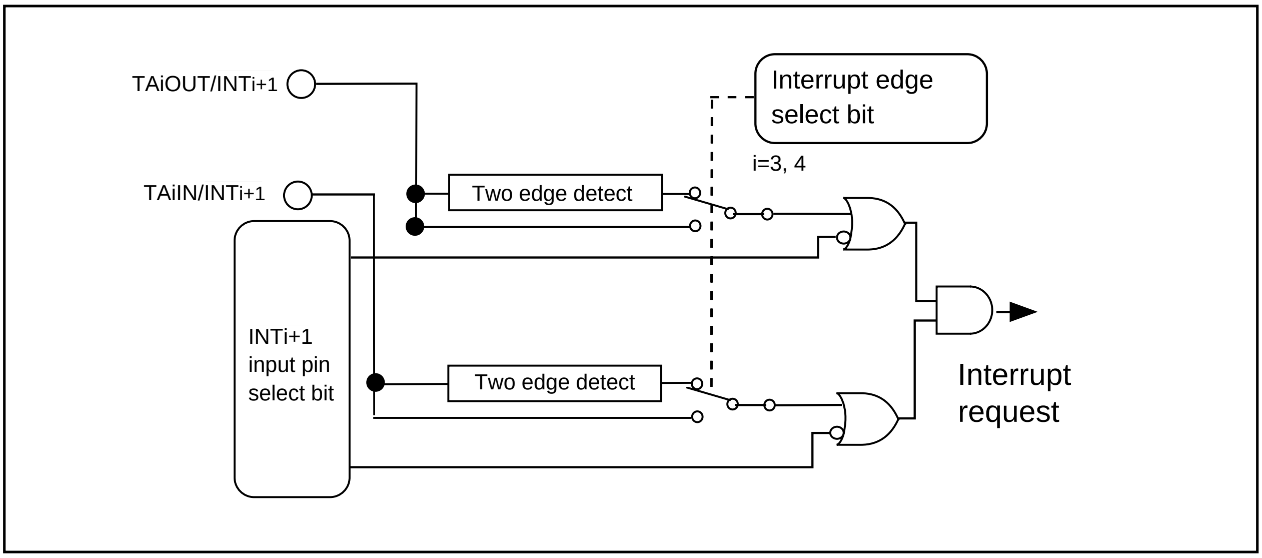


Figure 1.10.11. Constitution of $\overline{\text{INT4}}$ and $\overline{\text{INT5}}$

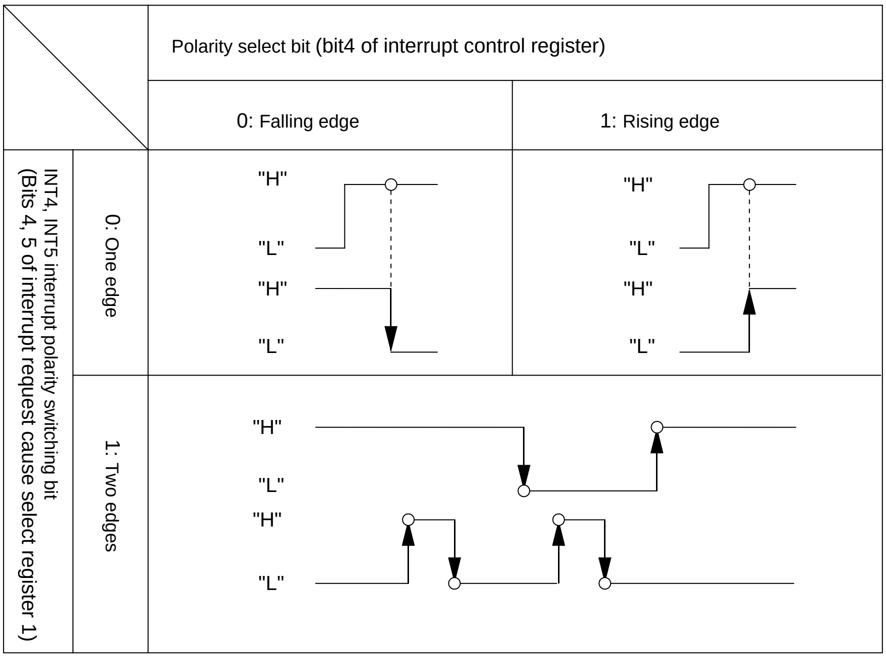

Polarity select bit (bit4 of interrupt control register)
0: Falling edge
1: Rising edge
INT4, INT5 interrupt polarity switching bit (Bits 4, 5 of interrupt request cause select register 1)
0: One edge
1: Two edges
"H"
"L"
"H"
"L"

Figure 1.10.12. Typical timings in two input interrupt of $\overline{\text{INT4}}$ and $\overline{\text{INT5}}$ selected

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

An $\overline{\text{NMI}}$ interrupt is generated when the input to the $P7_7$/$\overline{\text{NMI}}$ 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 $P7_7$ register (bit 7 at address $03ED_{16}$).

This pin cannot be used as a normal port input.

Key Input Interrupt

A key input interrupt request is generated when an active edge selected by the key input mode register's P1, P2 key input select bits occurs on one of input ports $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, or $P3_0$ to $P3_3$ whose direction register is set for input and which has been enabled for key input by the key input enable bit. For $P3_0$ to $P3_3$, key input interrupt requests are always generated by a falling edge.

A key input interrupt can also be used as a key-on wakeup function for cancelling the wait mode or stop mode. When using an oscillator connected between X_{CIN}—X_{COUT} and the corresponding port has been set to have a pullup, if the P1, P2 key input select bits (bits 0, 2 at address 0126_{16}) are set for "Two edges" and the P1, P2 key input enable bits (bits 1, 3 at address 0126_{16}) are "Enabled", pullups on $P1_0$ to $P1_7$ and $P2_0$ to $P2_7$ are automatically turned on and the port is pulled "H" for only a period of about 244 us (Note) at intervals of approximately 7.8 ms (Note), as shown in Figure 1.10.15. And if the key input enable bit (bit 1, 3 and 4 at the address 0126_{16}) is set to "enable", sometimes the interrupt request bit may be set to "1", therefore set the interrupt request bit to "0" with a program.

Figure 1.10.13 shows a block diagram for key input interrupts. Note that when a "L" signal is applied to any pin which has had its key input enable bit set to "0" and is not processed for input inhibition, input to other pins are not detected as an interrupt. The f_{C32} is affected by a clock prescaler reset flag.

Note : X_{CIN} = 32.768kHz

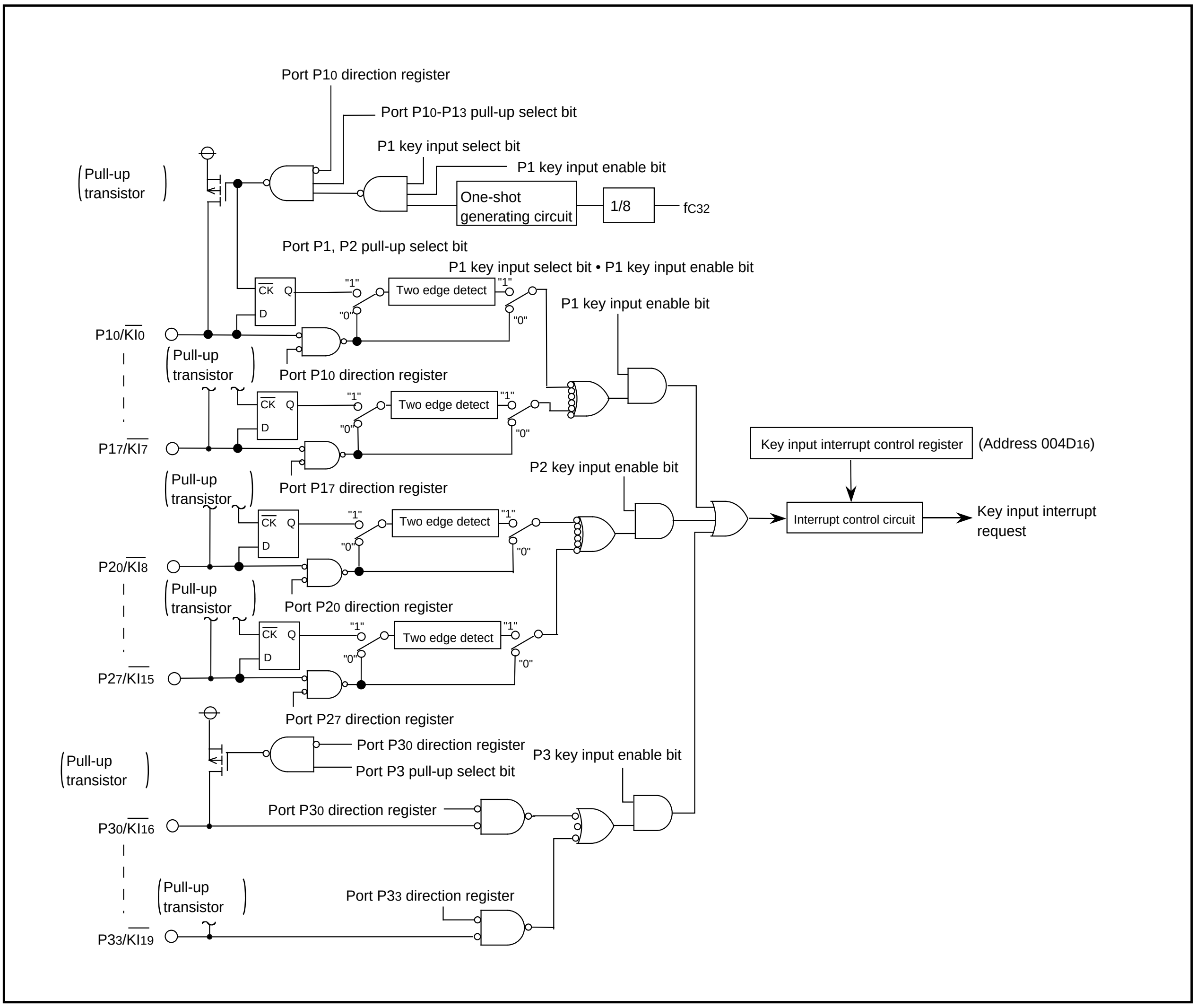
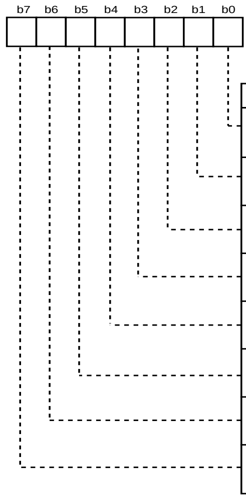

Port P10 direction register
Port P10-P13 pull-up select bit
P1 key input select bit
P1 key input enable bit
Pull-up transistor
One-shot generating circuit
1/8
fC32
Port P1, P2 pull-up select bit
P1 key input select bit • P1 key input enable bit
CK Q
D
"1"
"0"
Two edge detect
P1 key input enable bit
P10/KI0
P17/KI7
P20/KI8
P27/KI15
P30/KI16
P33/KI19
Port P10 direction register
Port P17 direction register
Port P20 direction register
Port P27 direction register
P2 key input enable bit
Key input interrupt control register
(Address 004D16)
Interrupt control circuit
Key input interrupt request
Port P30 direction register
Port P3 pull-up select bit
P3 key input enable bit
Port P33 direction register

Figure 1.10.13. Block diagram of key input interrupt

Key input mode register



Symbol	Address	When reset
KUPM	0126_{16}	01100000_2

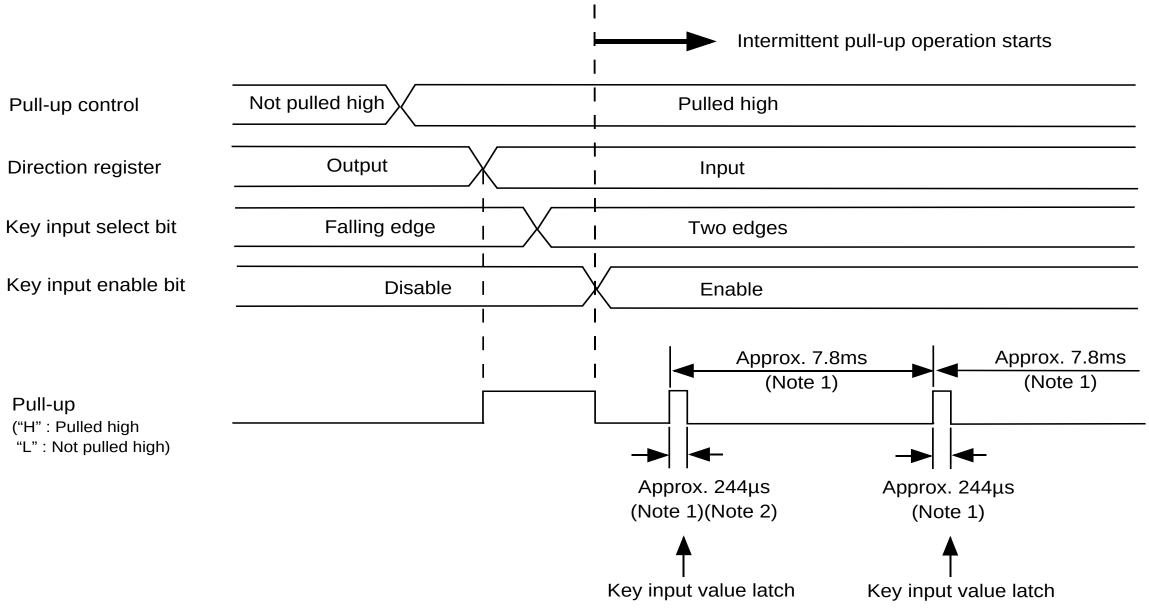
Bit	Bit name	Function	R	W
P1KIS	P1 key input select bit (Note 1)	0 : Falling edge 1 : Two edges (Note 2)	○	○
P1KIE	P1 key input enable bit	0 : Disable 1 : Enable	○	○
P2KIS	P2 key input select bit (Note 1)	0 : Falling edge 1 : Two edges (Note 2)	○	○
P2KIE	P2 key input enable bit	0 : Disable 1 : Enable	○	○
P3KIE	P3 key input enable bit	0 : Disable 1 : Enable	○	○
PUP12L	$P12_0$ to $P12_3$ pull-up (Note 3)	The corresponding port is pulled high with a pull-up resistor 0 : Not pulled high 1 : Pulled high	○	○
PUP12H	$P12_4$ to $P12_7$ pull-up (Note 3)		○	○
PUP13	$P13_0$ to $P13_2$ pull-up (Note 3)		○	○

Note 1 : If this bit is set for “Two edges” when the corresponding port has been specified to have a pullup, the port is automatically pulled high intermittently. Operating sub-clock.

Note 2 : When this bit is set for “Two edges” and the input from either of the corresponding pin is “L”, if the pullup control register 0 of the corresponding port (bit 2 to 5 at the address $03FC_{16}$) is changed, there may be the thing that the key input interruption request is set to “1”.

Note 3 : The pull-up resistance is not connected for pins that are set for output from peripheral functions, regardless of the setting in the pull-up control register.

Figure 1.10.14. Key input mode register



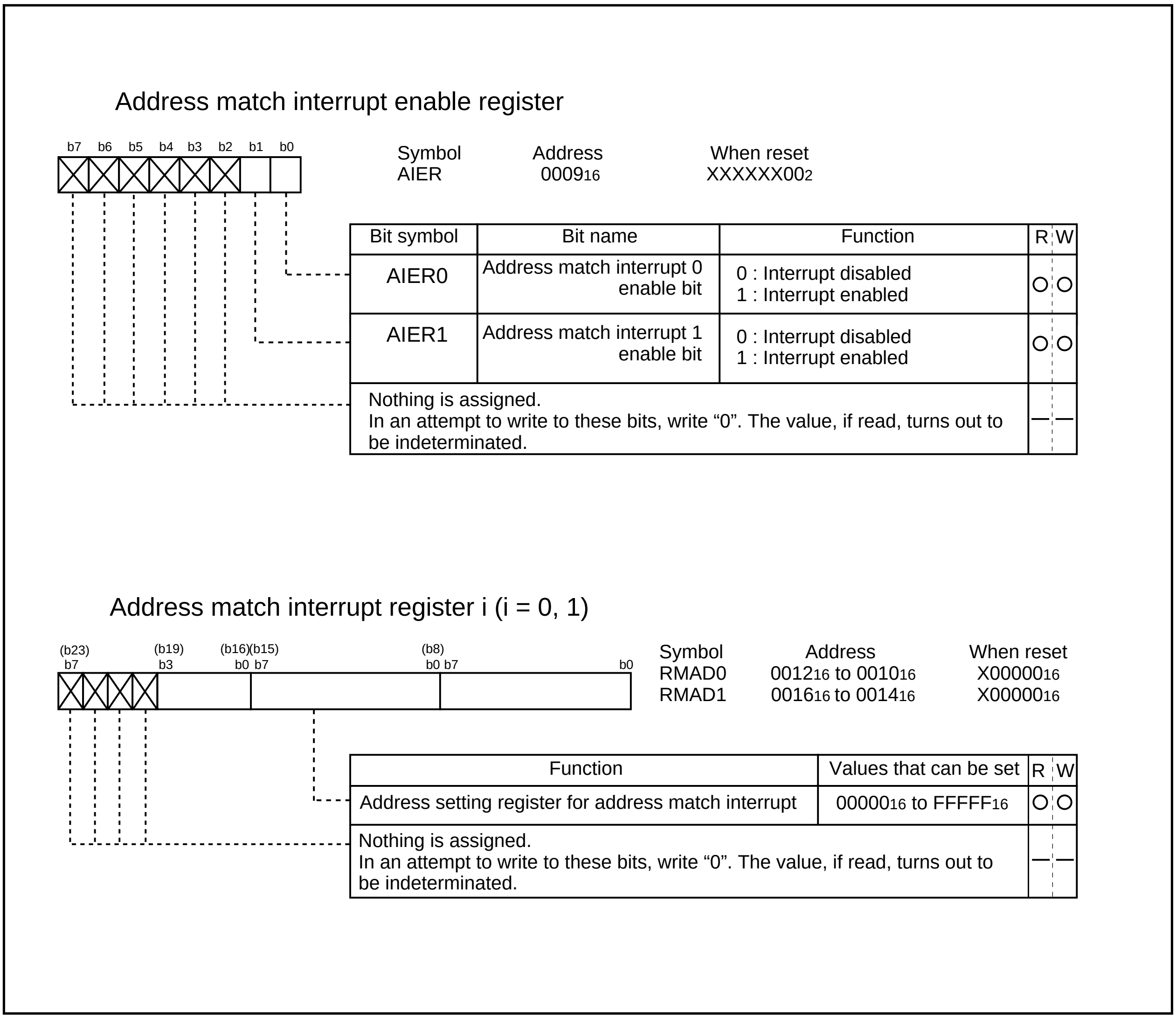
Note 1 : X_{CIN} = 32.768kHz

Note 2 : There may be the thing that the key input interrupt request bit is set to "1" when input "L" in the first key input value latch timing.

Figure 1.10.15. Intermittent pull-up operation

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 1.10.16 shows the address match interrupt-related registers.



Address match interrupt enable register

Symbol	Address	When reset
AIER	0009$_{16}$	XXXXXX00$_2$

Bit symbol	Bit name	Function	R	W
AIER0	Address match interrupt 0 enable bit	0 : Interrupt disabled 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 1.10.16. Address match interrupt-related registers

Precautions for Interrupts

(1) Reading address 00000_{16}

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

(2) Setting the stack pointer

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

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

- The $\overline{\text{NMI}}$ interrupt can not be disabled. Be sure to connect $\overline{\text{NMI}}$ pin to Vcc via a resistor (pull-up) if unused. Be sure to work on it.
- The $\overline{\text{NMI}}$ pin also serves as $P7_7$, which is exclusively input. Reading the contents of the P7 register allows reading the pin value. Use the reading of this pin only for establishing the pin level at the time when the $\overline{\text{NMI}}$ interrupt is input.
- Do not reset the CPU with the input to the $\overline{\text{NMI}}$ pin being in the “L” state.
- Do not attempt to go into stop mode with the input to the $\overline{\text{NMI}}$ pin being in the “L” state. With the input to the $\overline{\text{NMI}}$ being in the “L” state, the CM10 is fixed to “0”, so attempting to go into stop mode is turned down.
- Do not attempt to go into wait mode with the input to the $\overline{\text{NMI}}$ pin being in the “L” state. With the input to the $\overline{\text{NMI}}$ pin being in the “L” state, the CPU stops but the oscillation does not stop, so no power is saved. In this instance, the CPU is returned to the normal state by a later interrupt.
- Signals input to the $\overline{\text{NMI}}$ pin require an “L” level of 1 clock or more, from the operation clock of the CPU.

(4) External interrupt

- Either an “L” level or an “H” level of at least 250 ns width is necessary for the signal input to pins $\overline{\text{INT}_0}$ through $\overline{\text{INT}_5}$ regardless of the CPU operation clock.
- When the polarity of the $\overline{\text{INT}_0}$ to $\overline{\text{INT}_5}$ pins is changed, the interrupt request bit is sometimes set to "1". After changing the polarity, set the interrupt request bit to "0". Figure 1.10.17 shows the procedure for changing the $\overline{\text{INT}}$ interrupt generate factor.

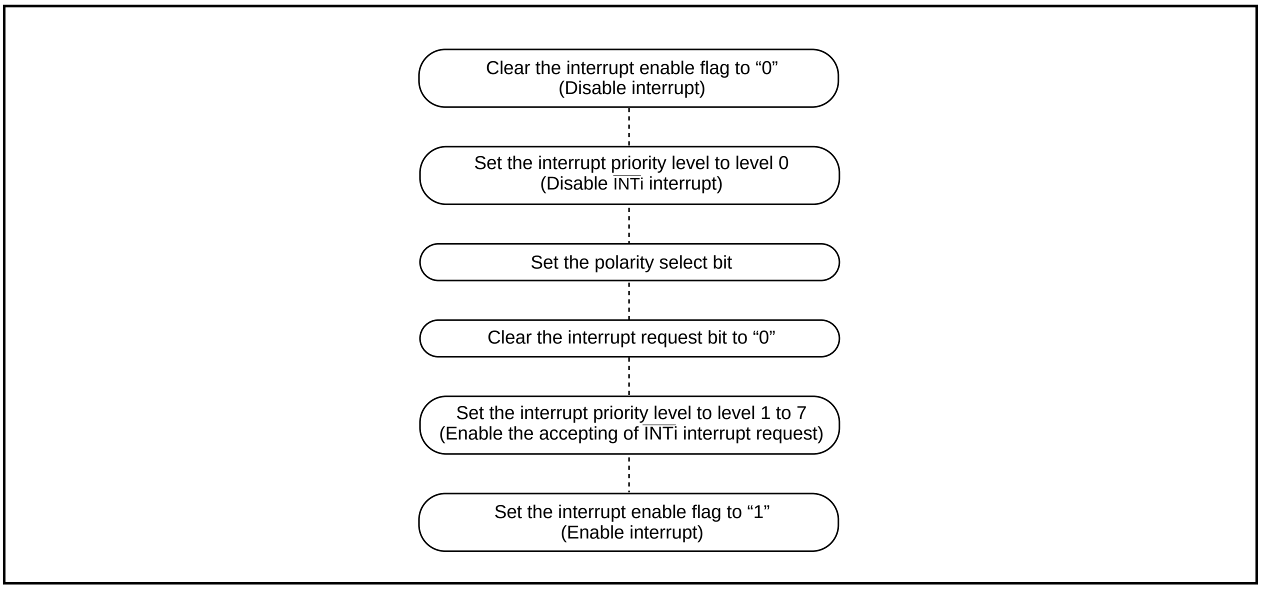

Clear the interrupt enable flag to “0”
(Disable interrupt)
Set the interrupt priority level to level 0
(Disable $\overline{\text{INT}}$i interrupt)
Set the polarity select bit
Clear the interrupt request bit to “0”
Set the interrupt priority level to level 1 to 7
(Enable the accepting of $\overline{\text{INT}}$i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

Figure 1.10.17. Switching condition of $\overline{\text{INT}}$ interrupt request

(5) Rewrite the interrupt control register

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

Example 1:

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

Example 2:

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

Example 3:

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

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

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

Watchdog Timer

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

With X_{IN} chosen for BCLK

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

With X_{CIN} chosen for BCLK

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

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

The watchdog timer is initialized by writing to the watchdog timer start register (address $000E_{16}$) and when a watchdog timer interrupt request is generated. The prescaler is initialized only when the microcomputer is reset. After a reset is cancelled, the watchdog timer and prescaler are both stopped. The count is started by writing to the watchdog timer start register (address $000E_{16}$). In stop mode and wait mode, the watchdog timer and prescaler are stopped. Counting is resumed from the held value when the modes or state are released.
Figure 1.11.1 shows the block diagram of the watchdog timer. Figure 1.11.2 shows the watchdog timer-related registers.

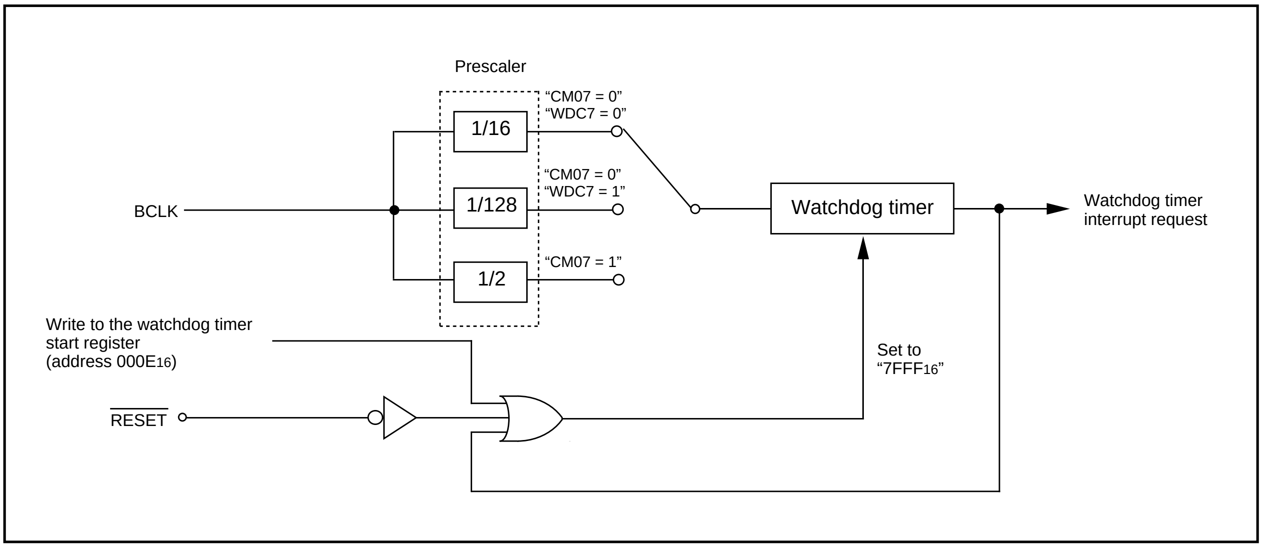
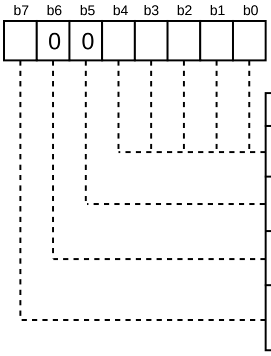


Figure 1.11.1. Block diagram of watchdog timer

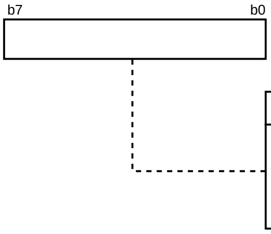
Watchdog timer control register



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

Bit symbol	Bit name	Function	R	W
	High-order bit of watchdog timer		○	×
	Reserved bit	Must always be set to “0”	○	○
	Reserved bit	Must always be set to “0”	○	○
WDC7	Prescaler select bit	0 : Divided by 16 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 1.11.2. Watchdog timer control and start registers

DMAC

This microcomputer has two DMAC (direct memory access controller) channels that allow data to be sent to memory without using the CPU. DMAC shares the same data bus with the CPU. The DMAC is given a higher right of using the bus than the CPU, which leads to working the cycle stealing method. On this account, the operation from the occurrence of DMA transfer request signal to the completion of 1-word (16-bit) or 1-byte (8-bit) data transfer can be performed at high speed. Figure 1.12.1 shows the block diagram of the DMAC. Table 1.12.1 shows the DMAC specifications. Figures 1.12.2 to 1.12.4 show the registers used by the DMAC.

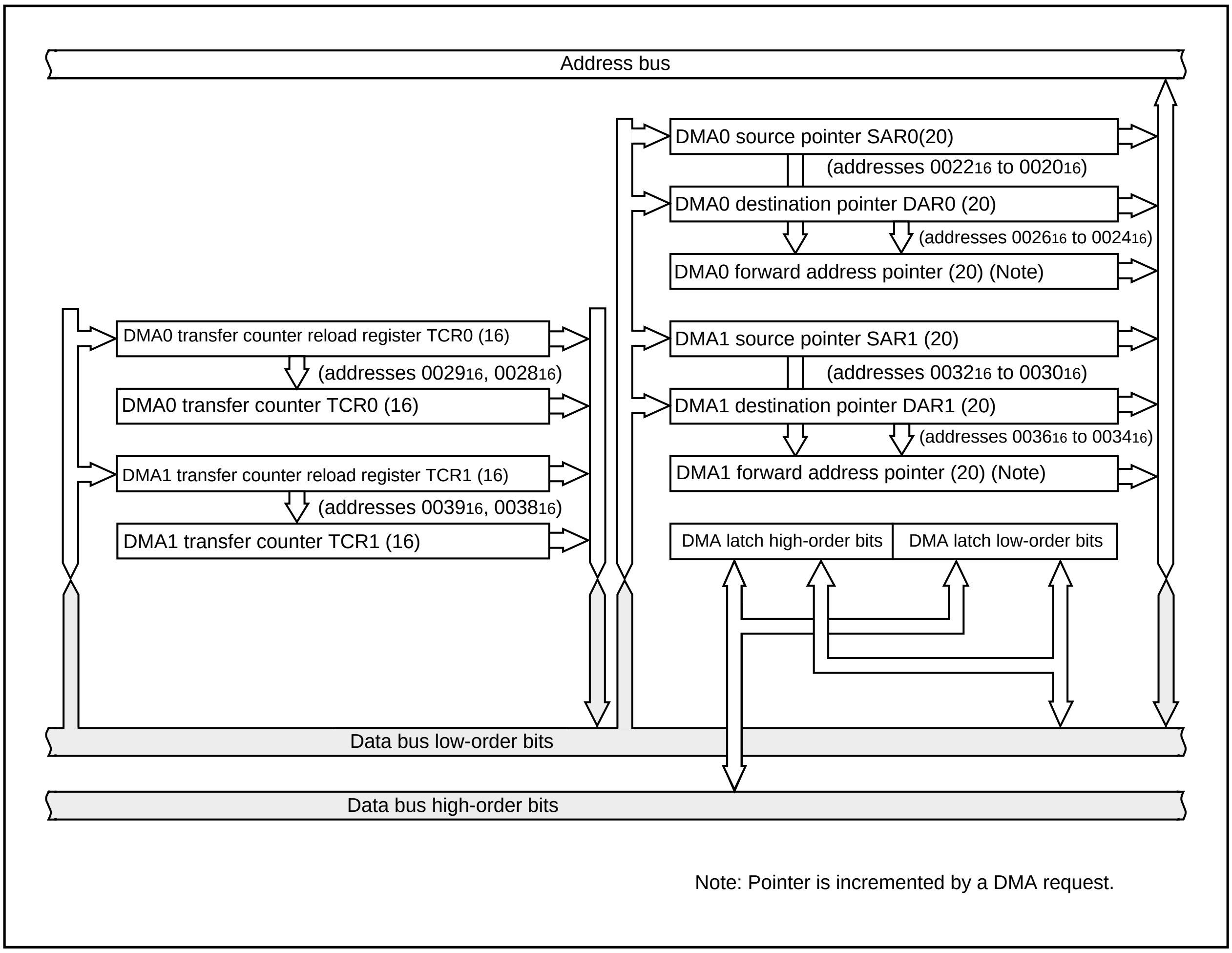

Address bus
DMA0 source pointer SAR0(20)
(addresses 0022_{16} to 0020_{16})
DMA0 destination pointer DAR0 (20)
(addresses 0026_{16} to 0024_{16})
DMA0 forward address pointer (20) (Note)
DMA1 source pointer SAR1 (20)
(addresses 0032_{16} to 0030_{16})
DMA1 destination pointer DAR1 (20)
(addresses 0036_{16} 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 1.12.1. Block diagram of DMAC

Either a write signal to the software DMA request bit or an interrupt request signal is used as a DMA transfer request signal. But the DMA transfer is affected neither by the interrupt enable flag (I flag) nor by the interrupt priority level. The DMA transfer doesn't affect any interrupts either.

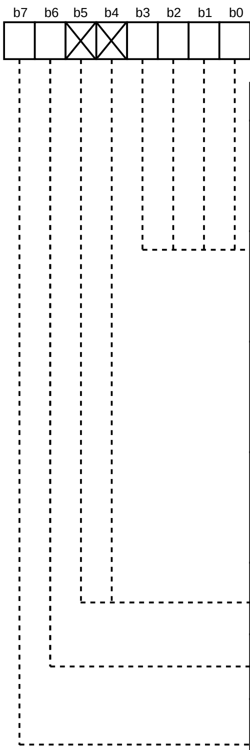
If the DMAC is active (the DMA enable bit is set to 1), data transfer starts every time a DMA transfer request signal occurs. If the cycle of the occurrences of DMA transfer request signals is higher than the DMA transfer cycle, there can be instances in which the number of transfer requests doesn't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

Table 1.12.1. 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 $\overline{\text{INT1}}$ or both edge Timer A0 to timer A7 interrupt requests Timer B0 to timer B5 interrupt requests UART0 transfer and reception interrupt requests UART1 transfer and reception interrupt requests UART2 transfer and reception interrupt requests 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 mode After the transfer counter underflows, the DMA enable bit turns to "0", and the DMAC turns inactive • Repeat transfer mode After the transfer counter underflows, the value of the transfer counter reload register is reloaded to the transfer counter. The DMAC remains active unless a "0" is written to the DMA enable bit.
DMA interrupt request generation timing	When an underflow occurs in the transfer counter
Active	When the DMA enable bit is set to "1", the DMAC is active. When the DMAC is active, data transfer starts every time a DMA transfer request signal occurs.
Inactive	• When the DMA enable bit is set to "0", the DMAC is inactive. • After the transfer counter underflows in single transfer mode
Reload timing for forward address pointer and transfer counter	At the time of starting data transfer immediately after turning the DMAC active, the value of one of source pointer and destination pointer - the one specified for the forward direction - is reloaded to the forward direction address pointer, and the value of the transfer counter reload register is reloaded to the transfer counter.
Writing to register	Registers specified for forward direction transfer are always write enabled. 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. DMA transfer is affected neither by the interrupt enable flag (I flag) nor by the interrupt priority level.

DMA0 request cause select register

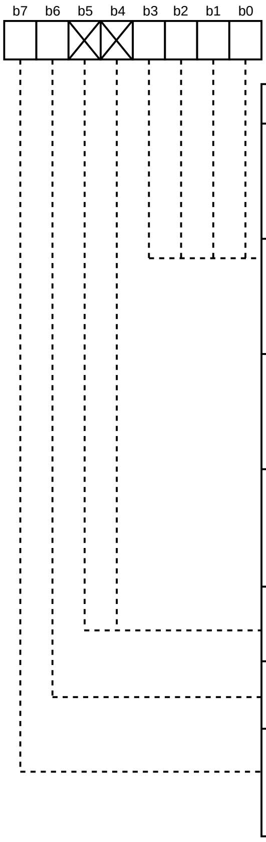


Symbol	Address	When reset
DM0SL	$03B8_{16}$	00_{16}

Bit symbol	Bit name	Function	R	W
DSEL0	DMA request cause select bit	b3 b2 b1 b0 0 0 0 0 : Falling edge of $\overline{\mathrm{INT0}}$ 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 $\overline{\mathrm{INT0}}$ 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. In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”.			–	–
DMS	DMA request cause expansion select 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 1.12.2. DMAC register (1)

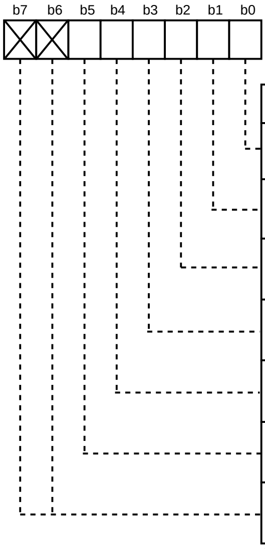
DMA1 request cause select register



Symbol DM1SL Address 03BA$_{16}$ 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 $\overline{INT1}$ 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(DMS=0) /timer A5(DMS=1) 0 1 0 1 : Timer A3(DMS=0) /timer A6 (DMS=1) 0 1 1 0 : Timer A4 (DMS=0) /timer A7 (DMS=1) 0 1 1 1 : Timer B0 (DMS=0) /two edges of $\overline{INT1}$ (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. In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”.			–	–
DMS	DMA request cause expansion select 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



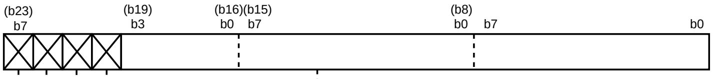
Symbol DMiCON(i=0,1) Address 002C$_{16}$, 003C$_{16}$ When reset 00000X00$_2$

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. In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”.			–	–

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

Figure 1.12.3. DMAC register (2)

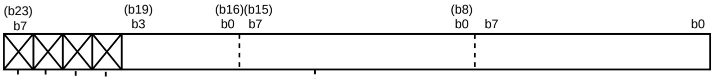
DMAi source pointer (i = 0, 1)



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

Function	Transfer address specification	R	W
• Source pointer Stores the source address	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 “0”.		–	–

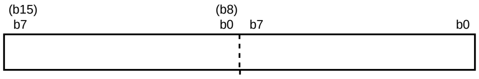
DMAi destination pointer (i = 0, 1)



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

Function	Transfer address specification	R	W
• Destination pointer Stores the destination address	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 “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 1.12.4. 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. Also, the bus cycle itself is longer when software waits are inserted.

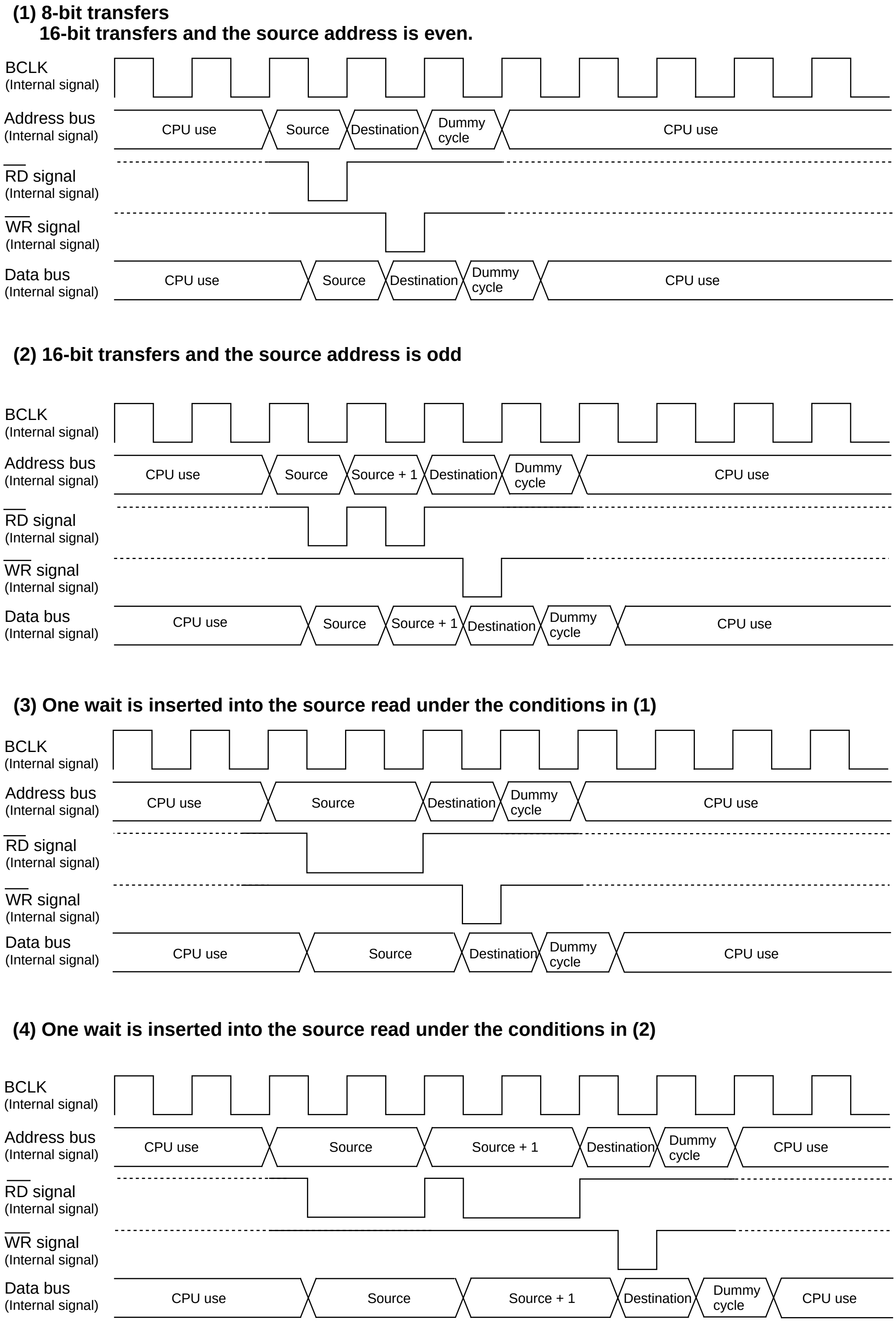
(a) Effect of source and destination addresses

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 software wait

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

Figure 1.12.5 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.



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

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

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

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

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

Figure 1.12.5. 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 1.12.2 shows the number of DMAC transfer cycles.

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

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

Table 1.12.2. No. of DMAC transfer cycles

Transfer unit	Access address	No. of read cycles	No. of read cycles
8-bit transfers (DMBIT= “1”)	Even	1	1
	Odd	1	1
16-bit transfers (DMBIT= “0”)	Even	1	1
	Odd	2	2

Coefficient j, k

Internal memory		
Internal ROM/RAM No wait	Internal ROM/RAM With wait	SFR area
1	2	2

DMA enable bit

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

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

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

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

DMA request bit

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

DMA request factors include the following.

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

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

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

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

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

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

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

(1) Internal factors

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

Turning the DMA request bit to "0" due to an internal factor is timed to be effected immediately before the transfer starts.

(2) External factors

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

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

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

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

(3) The priorities of channels and DMA transfer timing

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

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

Figure 1.12.6 An example of DMA transfer effected by external factors.

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

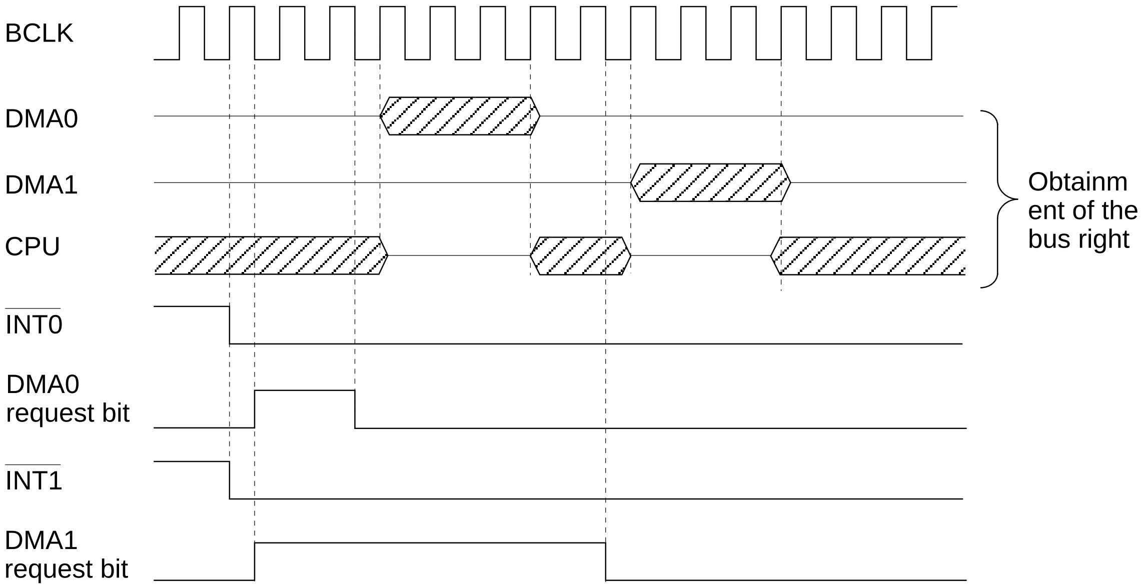
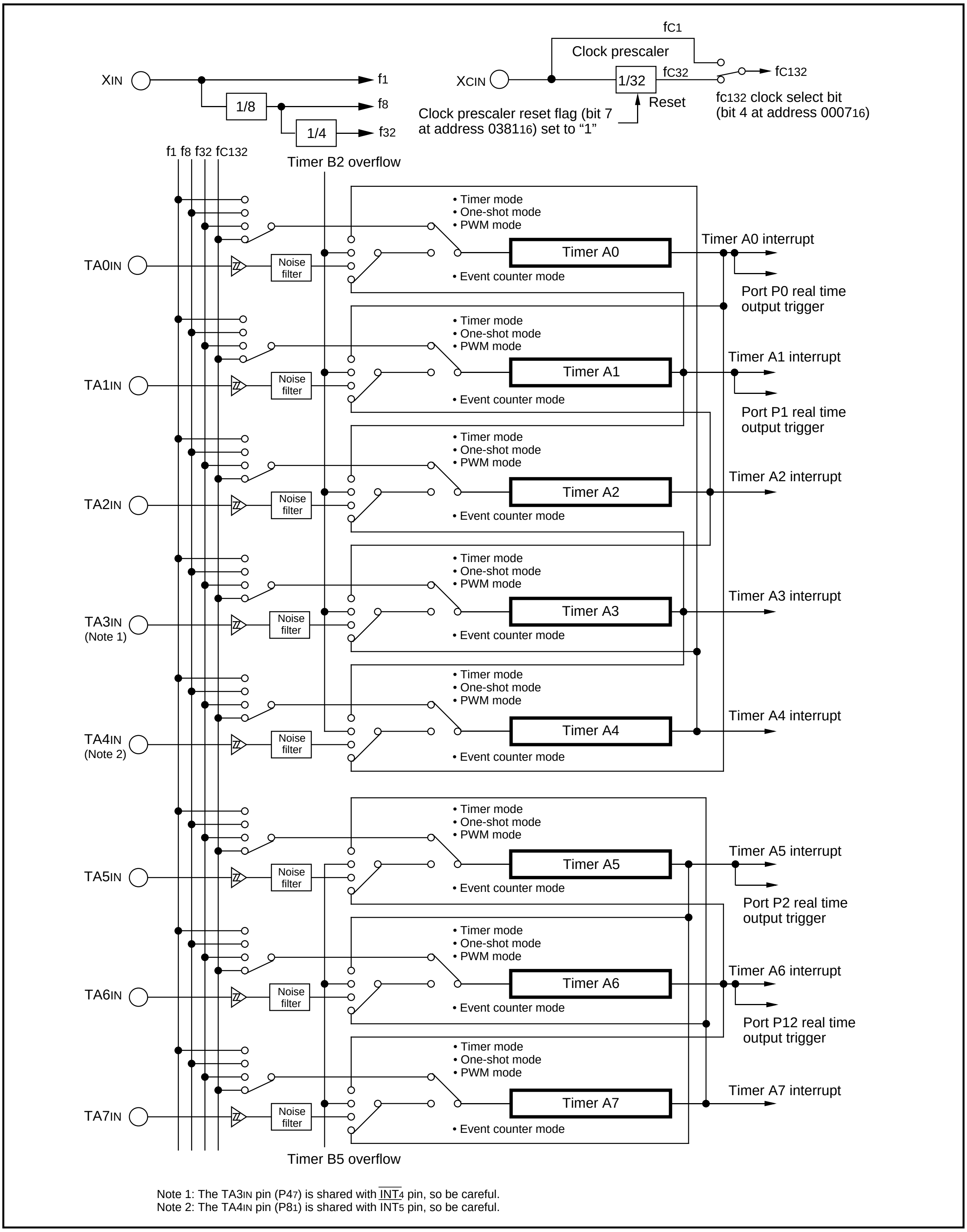


Figure 1.12.6. An example of DMA transfer effected by external factors

Timer

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



Note 1: The TA3IN pin (P47) is shared with $\overline{\text{INT4}}$ pin, so be careful.
Note 2: The TA4IN pin (P81) is shared with INT5 pin, so be careful.

Figure 1.13.1. Timer A block diagram

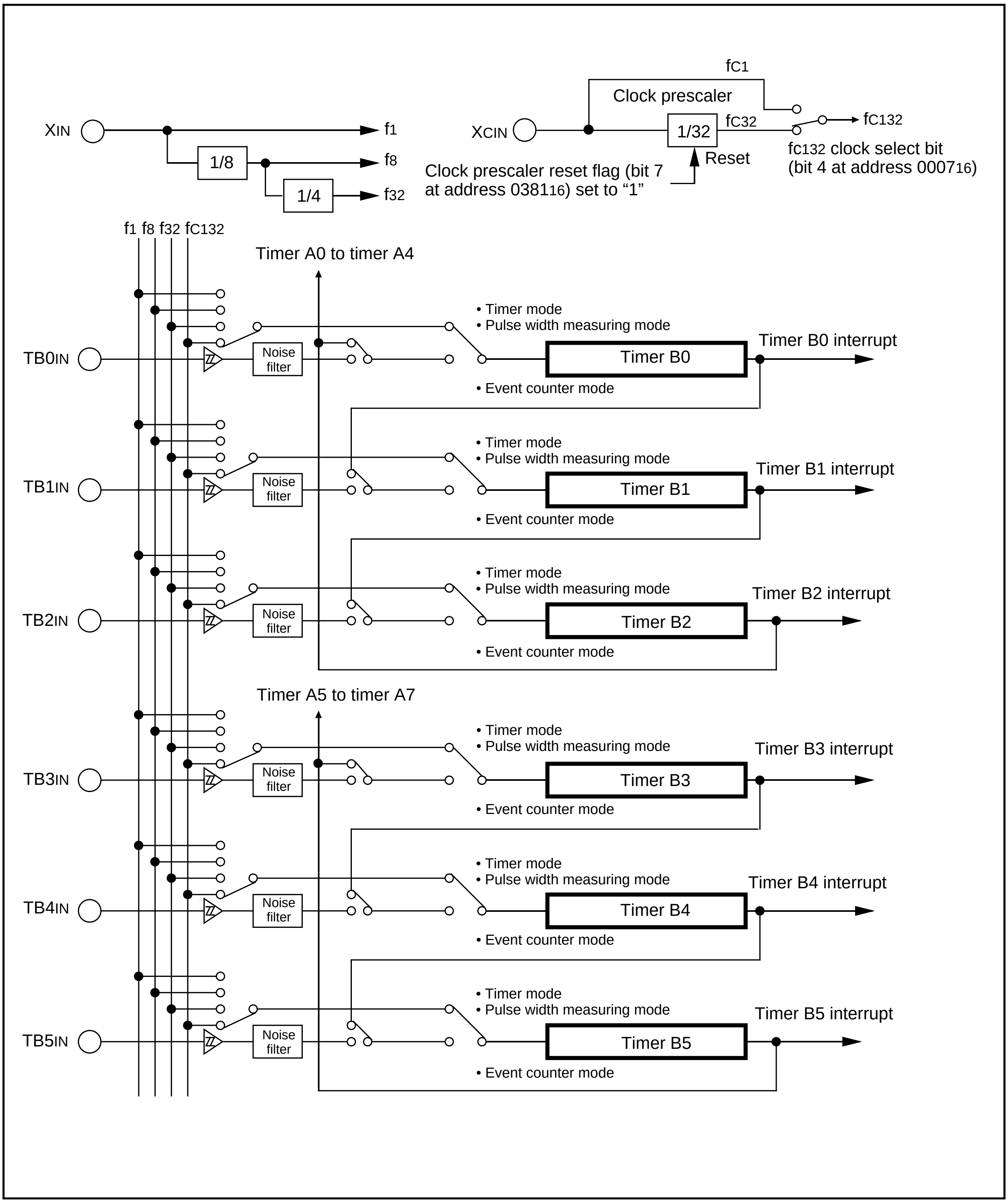


Figure 1.13.2. Timer B block diagram

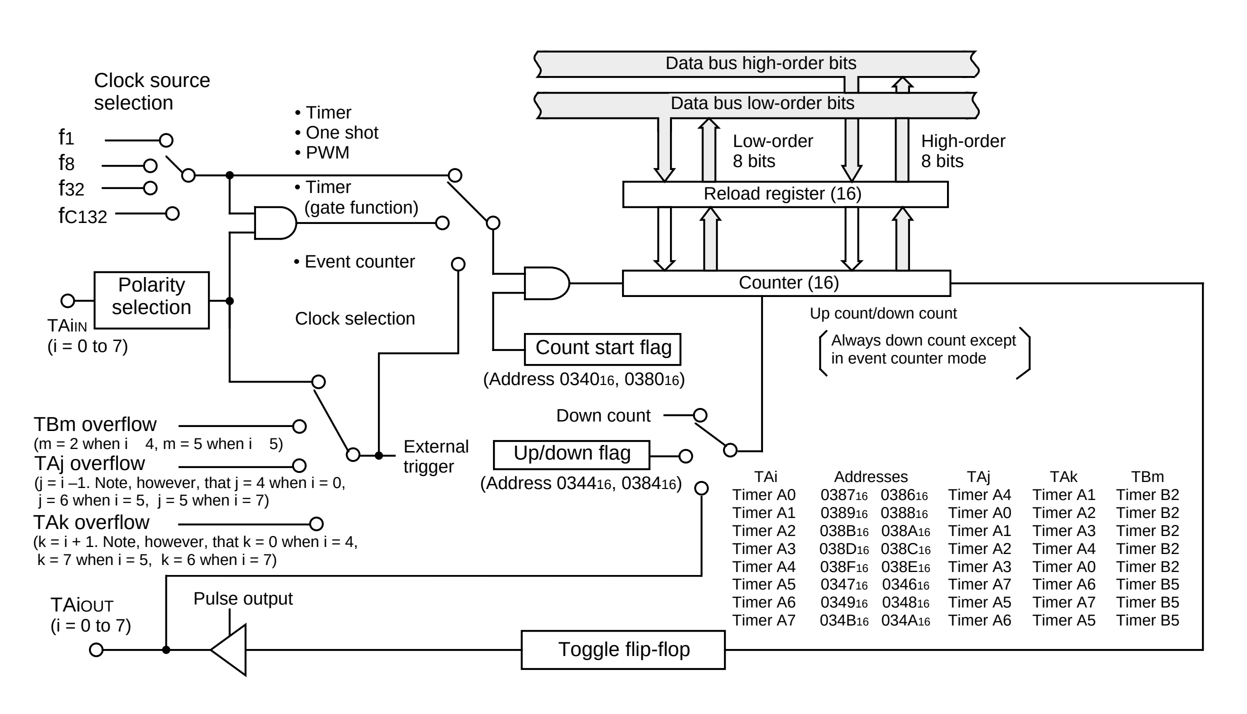
Timer A

Figure 1.13.3 shows the block diagram of timer A. Figures 1.13.4 to 1.13.8 show the timer A-related registers.

Use the timer Ai mode register (i = 0 to 7) bits 0 and 1 to choose the desired mode.

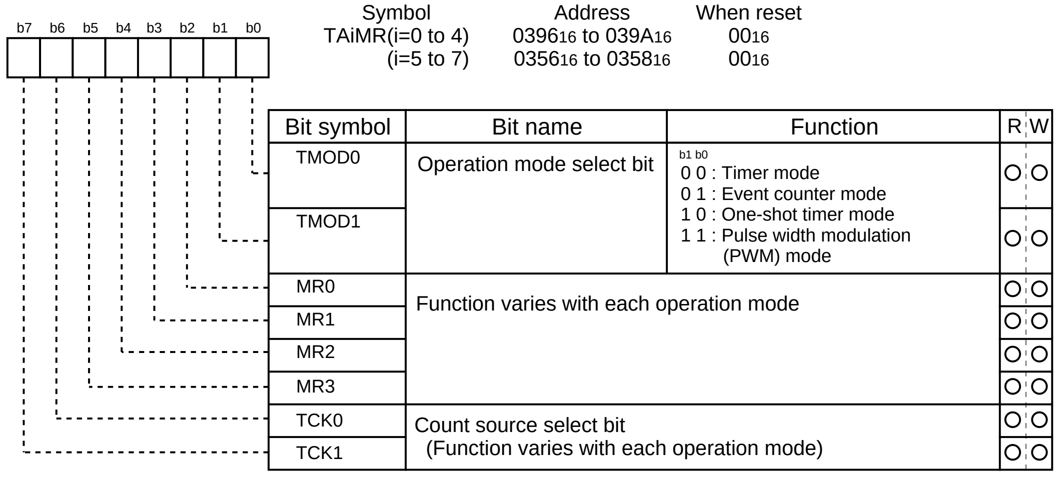
Timer A has the four operation modes listed as follows:

- 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	TBm
Timer A0	0387_{16}	0386_{16}	Timer A4	Timer A1	Timer B2
Timer A1	0389_{16}	0388_{16}	Timer A0	Timer A2	Timer B2
Timer A2	$038B_{16}$	$038A_{16}$	Timer A1	Timer A3	Timer B2
Timer A3	$038D_{16}$	$038C_{16}$	Timer A2	Timer A4	Timer B2
Timer A4	$038F_{16}$	$038E_{16}$	Timer A3	Timer A0	Timer B2
Timer A5	0347_{16}	0346_{16}	Timer A7	Timer A6	Timer B5
Timer A6	0349_{16}	0348_{16}	Timer A5	Timer A7	Timer B5
Timer A7	$034B_{16}$	$034A_{16}$	Timer A6	Timer A5	Timer B5

Figure 1.13.3. Block diagram of timer A



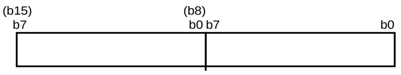
Timer Ai mode register

Symbol	Address	When reset
TAiMR(i=0 to 4)	0396_{16} to $039A_{16}$	00_{16}
(i=5 to 7)	0356_{16} to 0358_{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 1.13.4. Timer A-related registers (1)

Timer Ai register (Note 1)

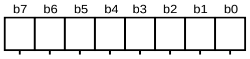


Symbol	Address	When reset
TA0	0387$_{16}$,0386$_{16}$	Indeterminate
TA1	0389$_{16}$,0388$_{16}$	Indeterminate
TA2	038B$_{16}$,038A$_{16}$	Indeterminate
TA3	038D$_{16}$,038C$_{16}$	Indeterminate
TA4	038F$_{16}$,038E$_{16}$	Indeterminate
TA5	0347$_{16}$,0346$_{16}$	Indeterminate
TA6	0349$_{16}$,0348$_{16}$	Indeterminate
TA7	034B$_{16}$,034A$_{16}$	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$_{16}$ (Note 2, Note 4)	×	○
• Pulse width modulation mode (16-bit PWM) Functions as a 16-bit pulse width modulator	0000$_{16}$ to FFFE$_{16}$ (Note 3, Note 4)	×	○
• 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$_{16}$ (High-order address) 00$_{16}$ to FF$_{16}$ (Low-order address) (Note 3, Note 4)	×	○

Note 1: Read and write data in 16-bit units.
Note 2: When the timer Ai register is set to “0000$_{16}$”, the counter does not operate and the timer Ai interrupt request is not generated. When the pulse is set to output, the pulse does not output from the TAiOUT pin.
Note 3: When the timer Ai register is set to “0000$_{16}$”, the pulse width modulator does not operate and the output level of the TAiOUT pin remains “L” level, therefore the timer Ai interrupt request is not generated. This also occurs in the 8-bit pulse width modulator mode when the significant 8 high-order bits in the timer Ai register are set to “00$_{16}$”.
Note 4: Use MOV instruction to write to this register.

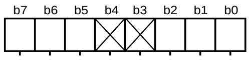
Count start flag 0



Symbol	Address	When reset
TABSR0	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		○	○

Count start flag 1



Symbol	Address	When reset
TABSR1	0340$_{16}$	000XX000$_2$

Bit symbol	Bit name	Function	R	W
TA5S	Timer A5 count start flag	0 : Stops counting 1 : Starts counting	○	○
TA6S	Timer A6 count start flag		○	○
TA7S	Timer A7 count start flag		○	○
Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminate.			—	—
TB3S	Timer B3 count start flag	0 : Stops counting 1 : Starts counting	○	○
TB4S	Timer B4 count start flag		○	○
TB5S	Timer B5 count start flag		○	○

Figure 1.13.5. Timer A-related registers (2)

Up/down flag 0 (Note)

b7 b6 b5 b4 b3 b2 b1 b0

Symbol	Address	When reset
UDF0	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		×	○

Note : Use MOV instruction to write to this register.

Up/down flag 1 (Note)

b7 b6 b5 b4 b3 b2 b1 b0

Symbol	Address	When reset
UDF1	0344_{16}	$XX0XX000_2$

Bit symbol	Bit name	Function	R	W
TA5UD	Timer A5 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	○	○
TA6UD	Timer A6 up/down flag		○	○
TA7UD	Timer A7 up/down flag		○	○
Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminate.			—	—
TA7P	Timer A7 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”	×	○
Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminate.			—	—

Note : Use MOV instruction to write to this register.

One-shot start flag 0

b7 b6 b5 b4 b3 b2 b1 b0

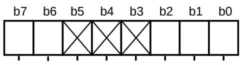
Symbol	Address	When reset
ONSF0	0382_{16}	$00X00000_2$

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. In an attempt to write to this bit, write “0”. The value, if read, turns out to be indeterminate.			—	—
TA0TGL	Timer A0 event/trigger select bit	b7 b6 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”.

Figure 1.13.6. Timer A-related registers (3)

One-shot start flag 1

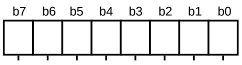


Symbol	Address	When reset
ONSF1	0342_{16}	$00XXX000_2$

Bit symbol	Bit name	Function	R	W
TA5OS	Timer A5 one-shot start flag	1 : Timer start When read, the value is “0”	○	○
TA6OS	Timer A6 one-shot start flag		○	○
TA7OS	Timer A7 one-shot start flag		○	○
Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminate.			—	—
TA5TGL	Timer A5 event/trigger select bit	b7 b6 0 0 : Input on $TA5_{IN}$ is selected (Note) 0 1 : TB5 overflow is selected 1 0 : TA6 overflow is selected 1 1 : TA7 overflow is selected	○	○
TA5TGH			○	○

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

Trigger select register 0

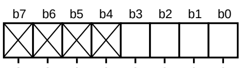


Symbol	Address	When reset
TRGSR0	0383_{16}	00_{16}

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”.

Trigger select register 1



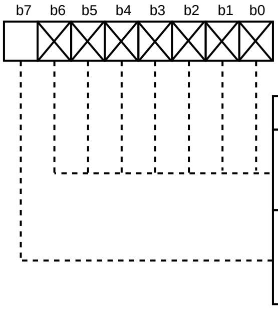
Symbol	Address	When reset
TRGSR1	0343_{16}	$XXXX0000_2$

Bit symbol	Bit name	Function	R	W
TA6TGL	Timer A6 event/trigger select bit	b1 b0 0 0 : Input on $TA6_{IN}$ is selected (Note) 0 1 : TB5 overflow is selected 1 0 : TA5 overflow is selected 1 1 : TA7 overflow is selected	○	○
TA6TGH			○	○
TA7TGL	Timer A7 event/trigger select bit	b3 b2 0 0 : Input on $TA7_{IN}$ is selected (Note) 0 1 : TB5 overflow is selected 1 0 : TA5 overflow is selected 1 1 : TA6 overflow is selected	○	○
TA7TGH			○	○
Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminate.			—	—

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

Figure 1.13.7. Timer A-related registers (4)

Clock prescaler reset flag



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

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

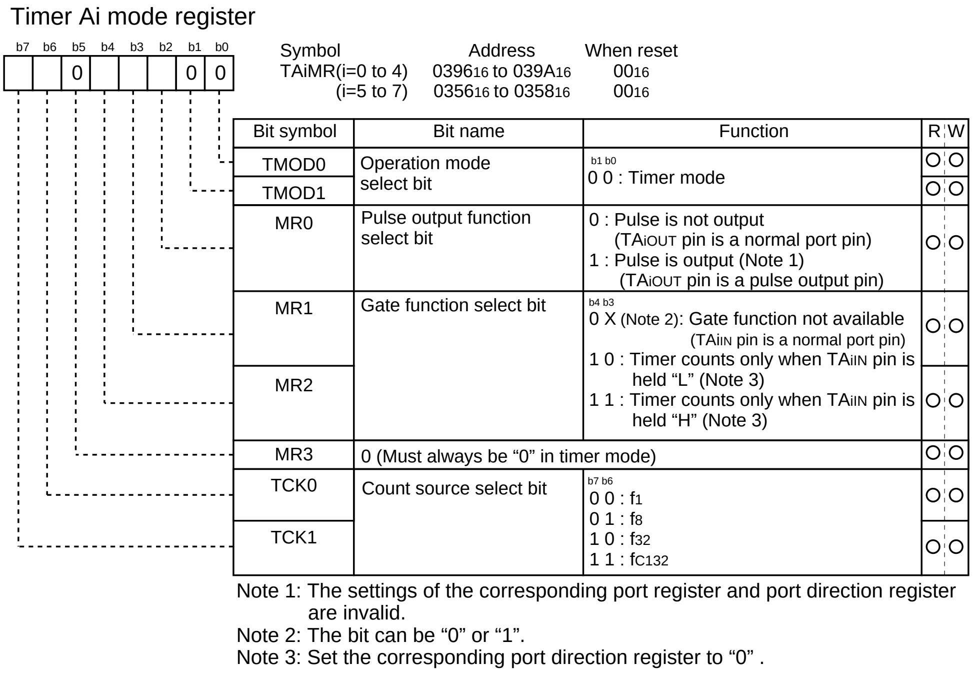
Figure 1.13.8. Timer A-related registers (5)

(1) Timer mode

In this mode, the timer counts an internally generated count source. (See Table 1.13.1.) Figure 1.13.9 shows the timer Ai mode register in timer mode.

Table 1.13.1. Specifications of timer mode

Item	Specification
Count source	f_1, f_8, f_{32}, f_{C132}
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
TAi$_{IN}$ pin function	Programmable I/O port or gate input
TAi$_{OUT}$ 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 TAi$_{IN}$ pin's input signal • Pulse output function Each time the timer underflows, the TAi$_{OUT}$ pin's polarity is reversed



Symbol	Address	When reset
TAiMR(i=0 to 4)	0396_{16} to $039A_{16}$	00_{16}
(i=5 to 7)	0356_{16} to 0358_{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 (TAi$_{OUT}$ pin is a normal port pin) 1 : Pulse is output (Note 1) (TAi$_{OUT}$ pin is a pulse output pin)	○	○
MR1	Gate function select bit	b4 b3 0 X (Note 2): Gate function not available (TAi$_{IN}$ pin is a normal port pin) 1 0 : Timer counts only when TAi$_{IN}$ pin is held "L" (Note 3) 1 1 : Timer counts only when TAi$_{IN}$ pin is held "H" (Note 3)	○	○
MR2			○	○
MR3	0 (Must always be "0" in timer mode)		○	○
TCK0	Count source select bit	b7 b6 0 0 : f_1 0 1 : f_8 1 0 : f_{32} 1 1 : f_{C132}	○	○
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 1.13.9. 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, A1, A5 and A6 can count a single-phase external signal. Timers A2, A3, A4 and A7 can count a single-phase and a two-phase external signal. Table 1.13.2 lists timer specifications when counting a single-phase external signal. Figure 1.13.10 shows the timer Ai mode register in event counter mode.

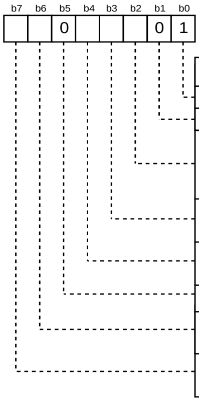
Table 1.13.3 lists timer specifications when counting a two-phase external signal. Figure 1.13.11 shows the timer Ai mode register in event counter mode.

Table 1.13.2. 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, TB5 overflow, TAj overflow, TAk 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
(When not using two-phase pulse signal processing)



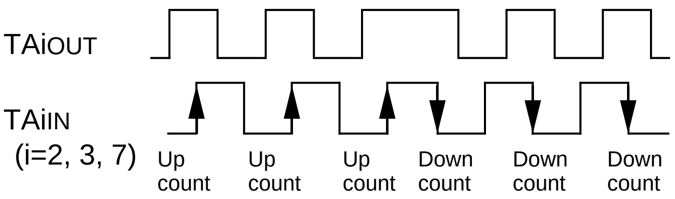
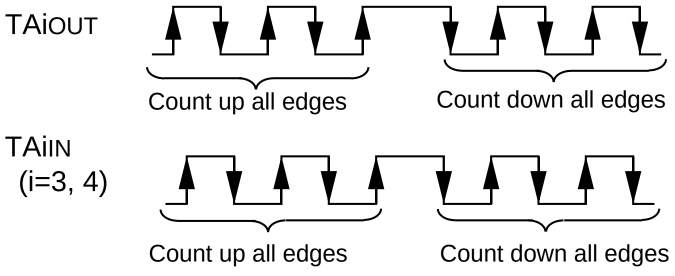
Symbol	Address	When reset
TAiMR(i = 0 to 4)	0396_{16} to $039A_{16}$	00_{16}
(i = 5 to 7)	0356_{16} to 0358_{16}	00_{16}

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 "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 0342_{16}, 0343_{16}, 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 1.13.10. Timer Ai mode register in event counter mode

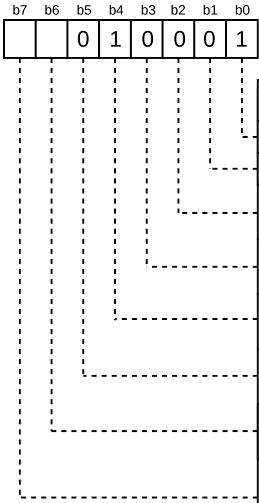
Table 1.13.3. Timer specifications in event counter mode (when processing two-phase pulse signal with timers A2, A3, A4 and A7)

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 1)
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, A4 or A7 register
Write to timer	• When counting stopped When a value is written to timer A2, A3, A4 or A7 register, it is written to both reload register and counter • When counting in progress When a value is written to timer A2, A3, A4 or A7 register, it is written to only reload register. (Transferred to counter at next reload time.)
Select function (Note 2)	• 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, 7) 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 1: This does not apply when the free-run function is selected.

Note 2: Timer A3 alone can be selected. Timer A2 and timer A7 are fixed to normal processing operation, and timer A4 is fixed to multiply-by-4 processing operation.

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



Symbol	Address	When reset
TAiMR(i = 2 to 4)	0398_{16} to $039A_{16}$	00_{16}
(i = 7)	0358_{16}	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. Timer A2 and timer A7 are fixed to normal processing operation, and timer A4 is fixed to multiply-by-4 processing operation.

Note 2 : When performing two-phase pulse signal processing, make sure the two-phase pulse signal processing operation select bit (addresses 0384_{16} and 0344_{16}) is set to “1”. Also, always be sure to set the event/trigger select bit (addresses 0383_{16} and 0343_{16}) to “00”.

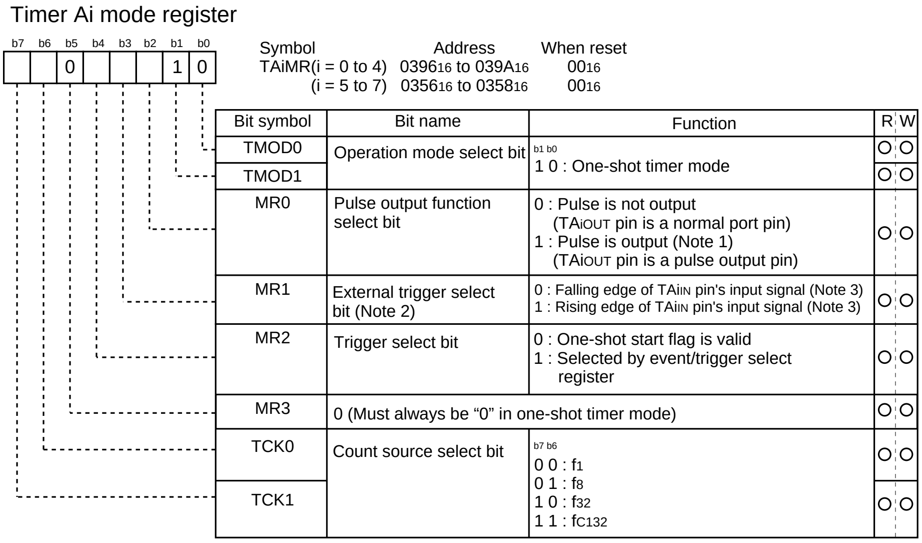
Figure 1.13.11. Timer Ai mode register in event counter mode

(3) One-shot timer mode

In this mode, the timer operates only once. (See Table 1.13.4.) When a trigger occurs, the timer starts up and continues operating for a given period. Figure 1.13.12 shows the timer Ai mode register in one-shot timer mode.

Table 1.13.4. Timer specifications in one-shot timer mode

Item	Specification
Count source	f_1, f_8, f_{32}, f_{C132}
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	Address	When reset
TAiMR(i = 0 to 4)	0396_{16} to $039A_{16}$	00_{16}
(i = 5 to 7)	0356_{16} to 0358_{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 (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_1 0 1 : f_8 1 0 : f_{32} 1 1 : f_{C132}	○	○
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 0342_{16}, 0343_{16}, 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 1.13.12. Timer Ai mode register in one-shot timer mode

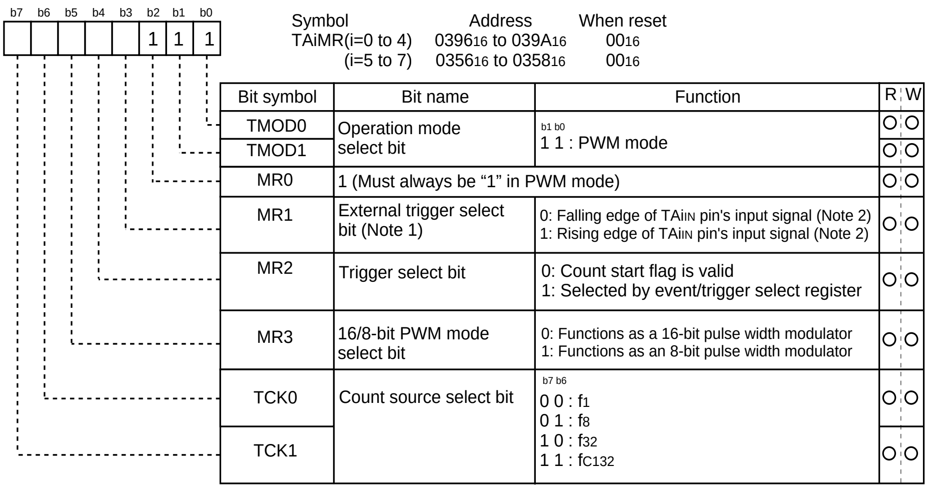
(4) Pulse width modulation (PWM) mode

In this mode, the timer outputs pulses of a given width in succession. (See Table 1.13.5.) In this mode, the counter functions as either a 16-bit pulse width modulator or an 8-bit pulse width modulator. Figure 1.13.13 shows the timer Ai mode register in pulse width modulation mode. Figure 1.13.14 shows the example of how a 16-bit pulse width modulator operates. Figure 1.13.15 shows the example of how an 8-bit pulse width modulator operates.

Table 1.13.5. Timer specifications in pulse width modulation mode

Item	Specification
Count source	f_1, f_8, f_{32}, f_{C132}
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 / fj n : Set value fj=f_1, f_8, f_{32}, f_{C132} • Cycle time $(2^{16}-1)$ / fj fixed
8-bit PWM	• High level width n×(m+1) / fj n : values set to timer Ai register's high-order address • Cycle time (2^8-1)×(m+1) / fj 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”
TAi$_{IN}$ pin function	Programmable I/O port or trigger input
TAi$_{OUT}$ 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)

Timer Ai mode register

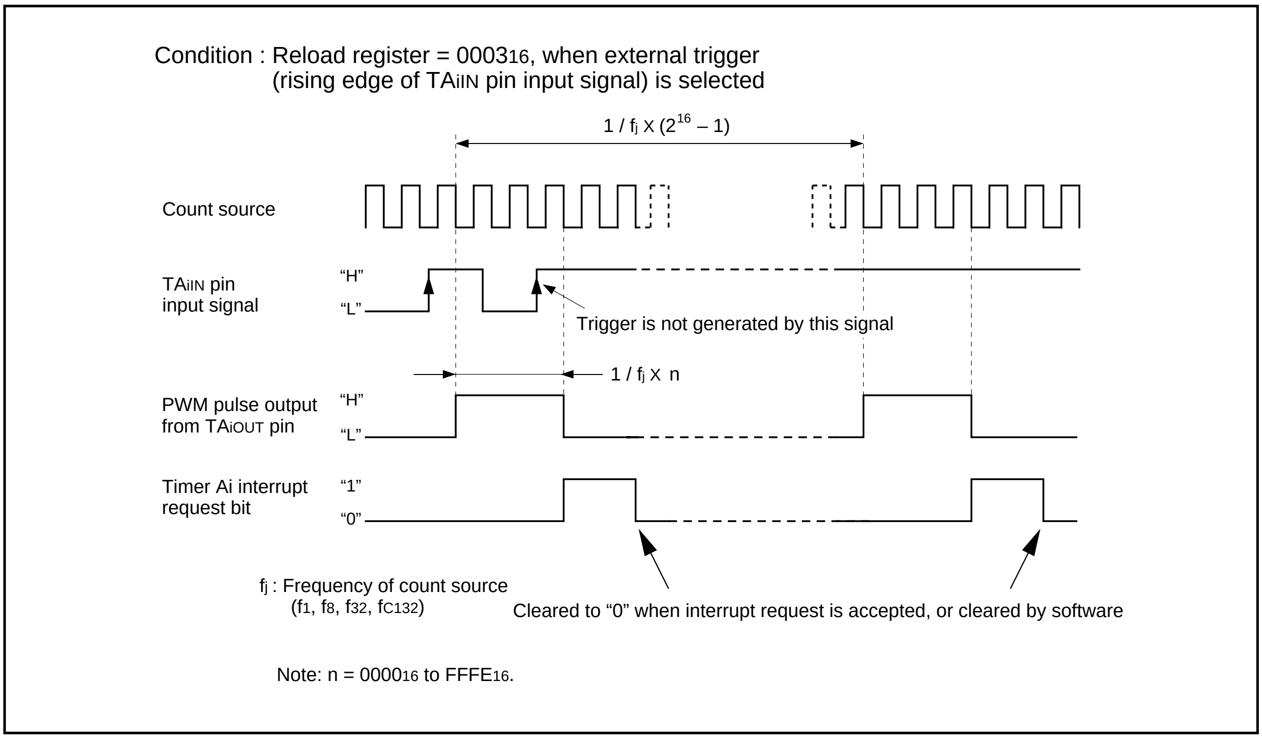


Symbol	Address	When reset
TAiMR(i=0 to 4)	0396_{16} to $039A_{16}$	00_{16}
(i=5 to 7)	0356_{16} to 0358_{16}	00_{16}

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 TAi$_{IN}$ pin's input signal (Note 2) 1: Rising edge of TAi$_{IN}$ 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 : f_1 0 1 : f_8 1 0 : f_{32} 1 1 : f_{C132}	○	○
TCK1			○	○

Note 1: Valid only when the TAi$_{IN}$ pin is selected by the event/trigger select bit (addresses 0342_{16}, 0343_{16}, 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 1.13.13. Timer Ai mode register in pulse width modulation mode



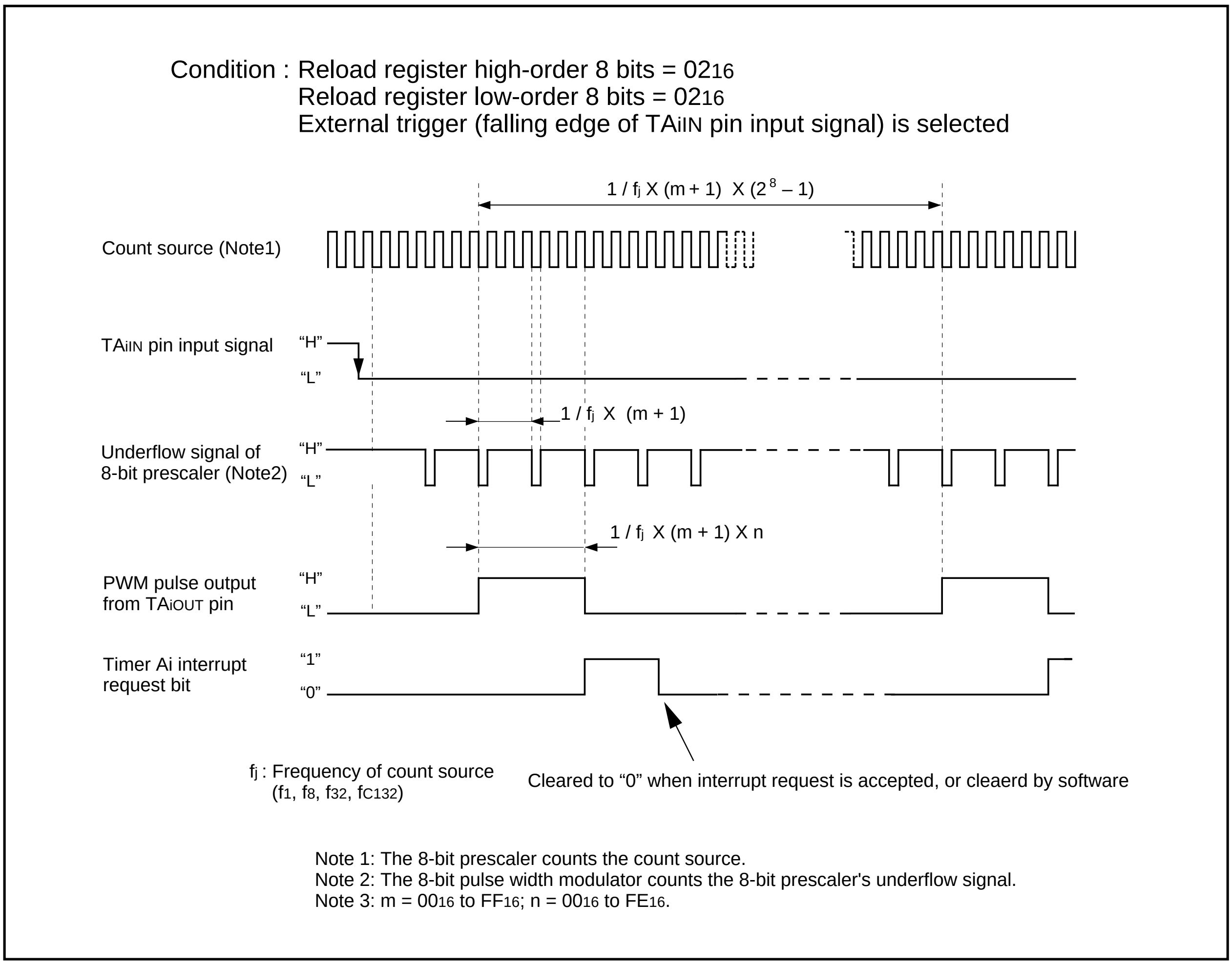
Condition : Reload register = 0003₁₆, when external trigger (rising edge of TAiIN pin input signal) is selected

1 / fj X (2^{16} – 1)
Count source
TAiIN pin input signal
"H"
"L"
Trigger is not generated by this signal
1 / fj X n
PWM pulse output from TAiOUT pin
"H"
"L"
Timer Ai interrupt request bit
"1"
"0"
Cleared to "0" when interrupt request is accepted, or cleared by software

fj : Frequency of count source (f1, f8, f32, fC132)

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

Figure 1.13.14. 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 / fj X (m + 1) X (2^{8} – 1)
Count source (Note1)
TAiIN pin input signal
"H"
"L"
1 / fj X (m + 1)
Underflow signal of 8-bit prescaler (Note2)
"H"
"L"
1 / fj X (m + 1) X n
PWM pulse output from TAiOUT pin
"H"
"L"
Timer Ai interrupt request bit
"1"
"0"
Cleared to "0" when interrupt request is accepted, or cleaerd by software

fj : Frequency of count source (f1, f8, f32, fC132)

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 FF_{16}; n = 00_{16} to FE_{16}.

Figure 1.13.15. Example of how an 8-bit pulse width modulator operates

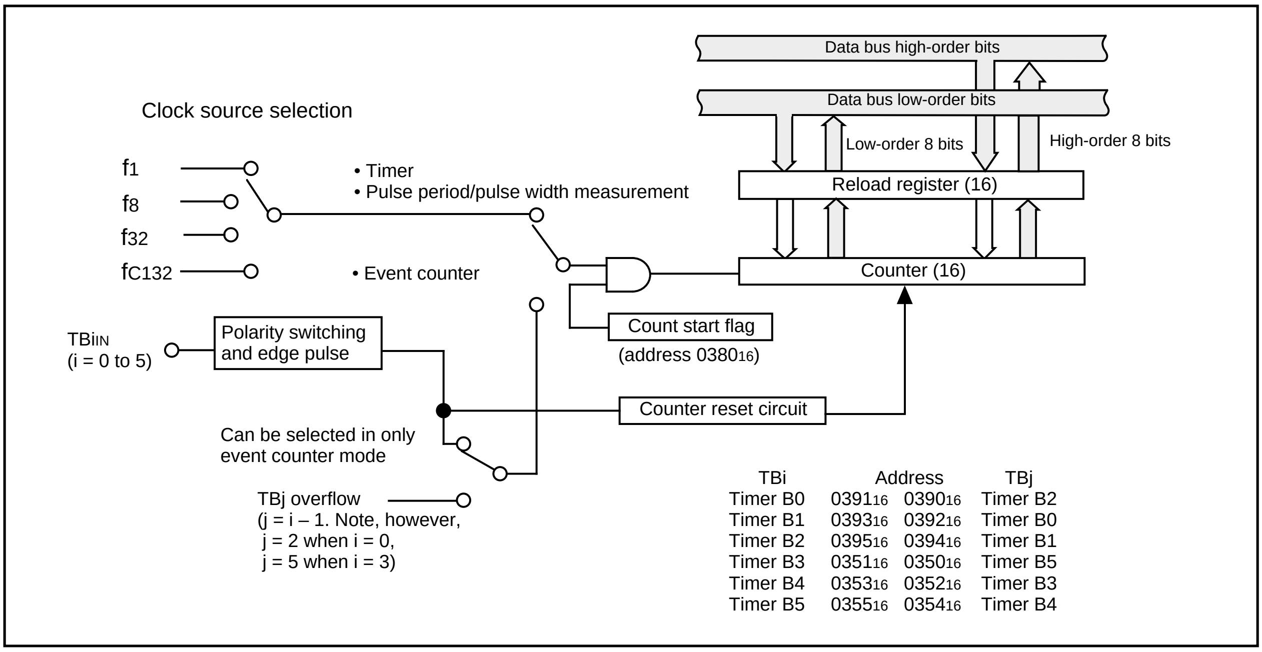
Timer B

Figure 1.13.16 shows the block diagram of timer B. Figures 1.13.17 and 1.13.18 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.

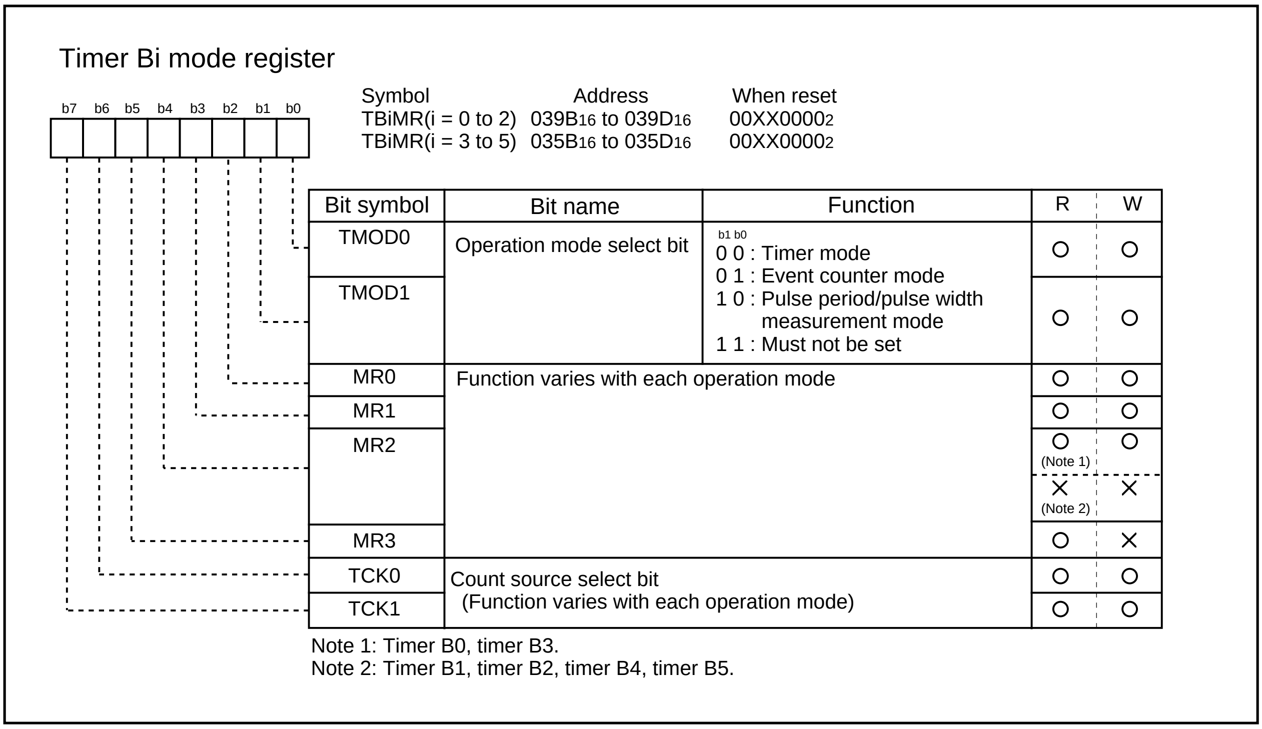
Timer B has three operation modes listed as follows:

- 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	0351_{16}	0350_{16}	Timer B5
Timer B4	0353_{16}	0352_{16}	Timer B3
Timer B5	0355_{16}	0354_{16}	Timer B4

Figure 1.13.16. Block diagram of timer B



Timer Bi mode register

Symbol	Address	When reset
TBiMR(i = 0 to 2)	$039B_{16}$ to $039D_{16}$	$00XX0000_2$
TBiMR(i = 3 to 5)	$035B_{16}$ to $035D_{16}$	$00XX0000_2$

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 : Must not be set	○	○
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 1.13.17. Timer B-related registers (1)

Timer Bi register (Note)

(b15) b7 (b8) b0 b7 b0

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

Function	Values that can be set	R	W
• Timer mode 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 0

b7 b6 b5 b4 b3 b2 b1 b0

Symbol	Address	When reset
TABSR 0	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		○	○

Count start flag 1

b7 b6 b5 b4 b3 b2 b1 b0

Symbol	Address	When reset
TABSR1	0340_{16}	$000XX000_2$

Bit symbol	Bit name	Function	R	W
TA5S	Timer A5 count start flag	0 : Stops counting 1 : Starts counting	○	○
TA6S	Timer A6 count start flag		○	○
TA7S	Timer A7 count start flag		○	○
Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminate.			—	—
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

b7 b6 b5 b4 b3 b2 b1 b0

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

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

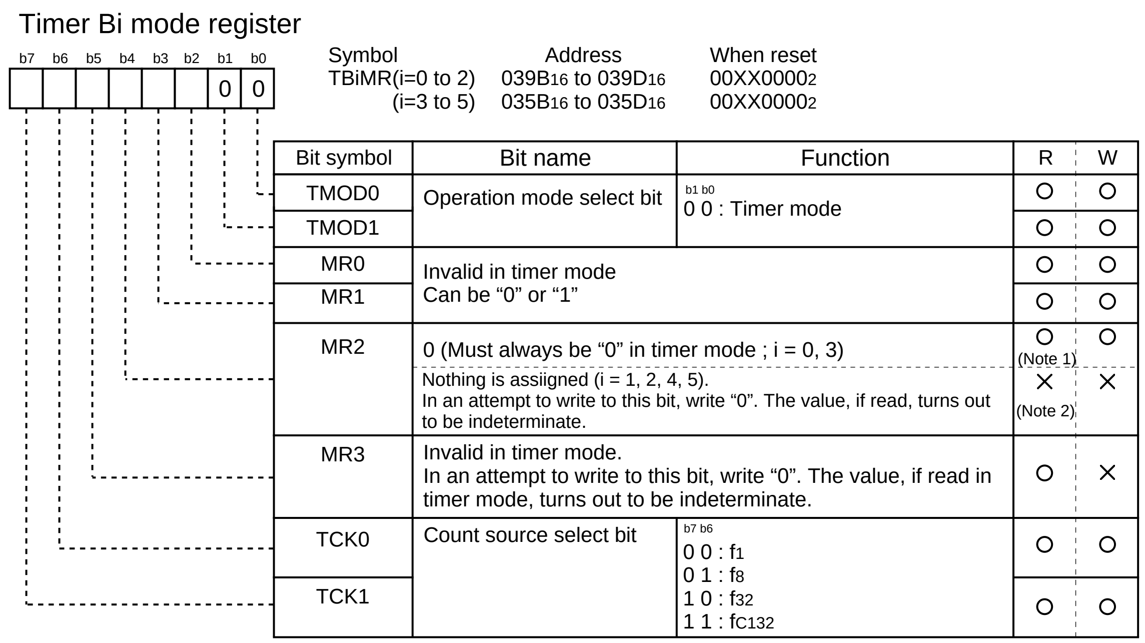
Figure 1.13.18. Timer B-related registers (2)

(1) Timer mode

In this mode, the timer counts an internally generated count source. (See Table 1.13.6.) Figure 1.13.19 shows the timer Bi mode register in timer mode.

Table 1.13.6. Timer specifications in timer mode

Item	Specification
Count source	f_1, f_8, f_{32}, f_{C132}
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 2)	$039B_{16}$ to $039D_{16}$	$00XX0000_2$
(i=3 to 5)	$035B_{16}$ to $035D_{16}$	$00XX0000_2$

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 (Must always be “0” in timer mode ; i = 0, 3)		○ (Note 1)	○
	Nothing is assiigned (i = 1, 2, 4, 5). In an attempt to write to this bit, write “0”. The value, if read, turns out to be indeterminate.		× (Note 2)	×
MR3	Invalid in timer mode. In an attempt to write to this bit, write “0”. The value, if read in timer mode, turns out to be indeterminate.		○	×
TCK0	Count source select bit	b7 b6 0 0 : f_1 0 1 : f_8 1 0 : f_{32} 1 1 : f_{C132}	○	○
TCK1			○	○

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

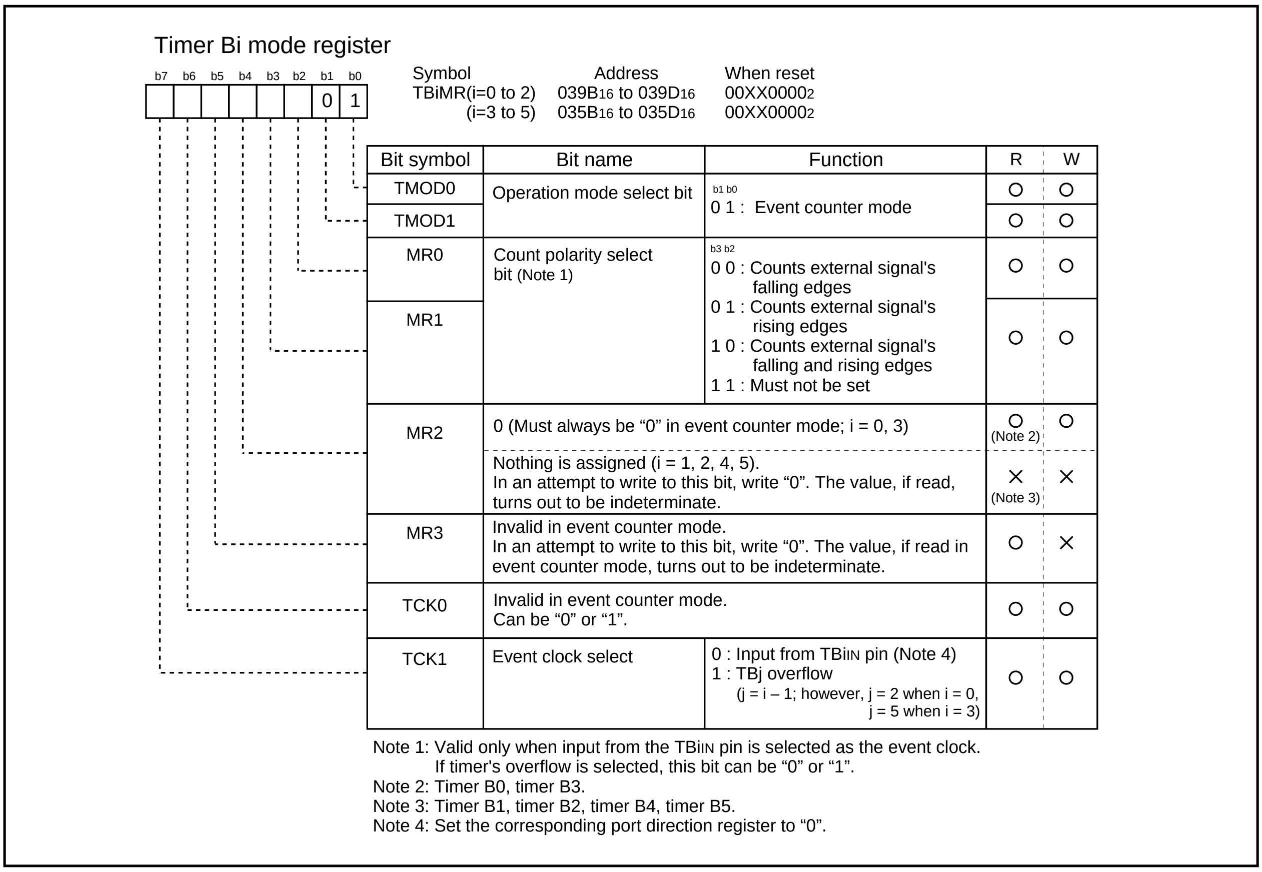
Figure 1.13.19. Timer Bi mode register in timer mode

(2) Event counter mode

In this mode, the timer counts an external signal or an internal timer's overflow. (See Table 1.13.7.) Figure 1.13.20 shows the timer Bi mode register in event counter mode.

Table 1.13.7. Timer specifications in event counter mode

Item	Specification
Count source	• External signals input to TBi_{IN} 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
TBi_{IN} 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)



Symbol	Address	When reset
TBiMR(i=0 to 2)	039B$_{16}$ to 039D$_{16}$	00XX0000$_2$
(i=3 to 5)	035B$_{16}$ to 035D$_{16}$	00XX0000$_2$

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 : Must not be set	○	○
MR1			○	○
MR2	0 (Must always be “0” in event counter mode; i = 0, 3)		○ (Note 2)	○
	Nothing is assigned (i = 1, 2, 4, 5). In an attempt to write to this bit, write “0”. The value, if read, turns out to be indeterminate.		× (Note 3)	×
MR3	Invalid in event counter mode. In an attempt to write to this bit, write “0”. The value, if read in event counter mode, turns out to be indeterminate.		○	×
TCK0	Invalid in event counter mode. Can be “0” or “1”.		○	○
TCK1	Event clock select	0 : Input from TBi_{IN} 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 TBi_{IN} 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 1.13.20. Timer Bi mode register in event counter mode

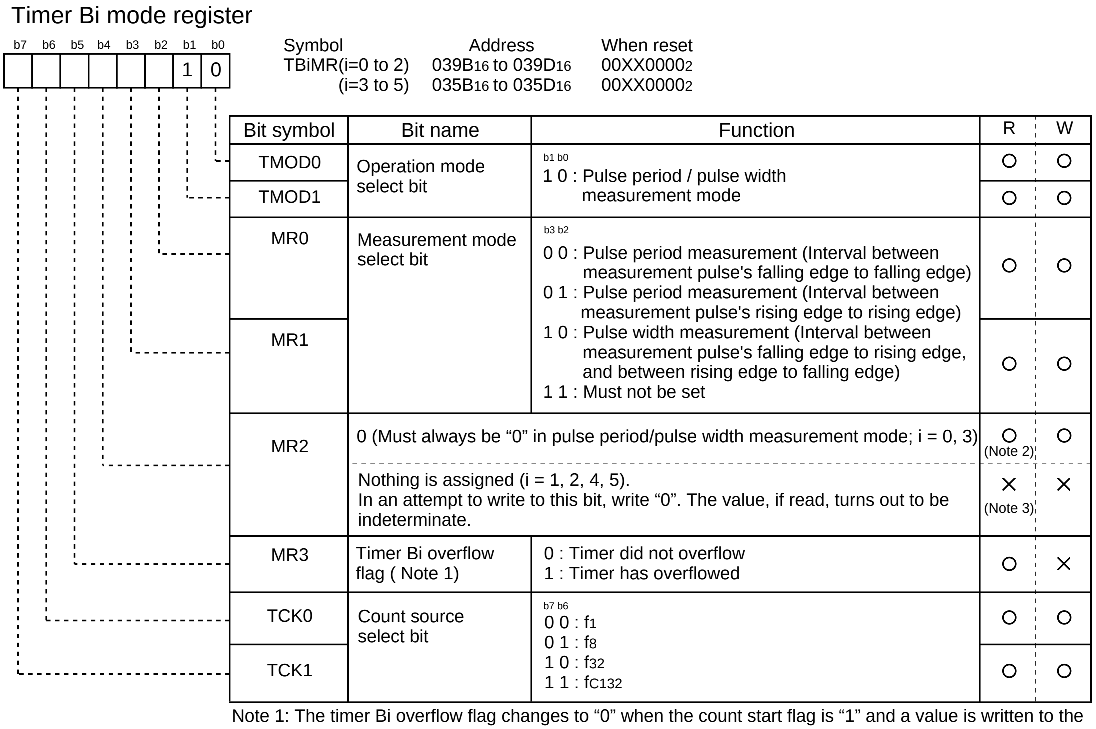
(3) Pulse period/pulse width measurement mode

In this mode, the timer measures the pulse period or pulse width of an external signal. (See Table 1.13.8.) Figure 1.13.21 shows the timer Bi mode register in pulse period/pulse width measurement mode. Figure 1.13.22 shows the operation timing when measuring a pulse period. Figure 1.13.23 shows the operation timing when measuring a pulse width.

Table 1.13.8. Timer specifications in pulse period/pulse width measurement mode

Item	Specification
Count source	f_1, f_8, f_{32}, f_{C132}
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.)
TBiIN pin function	Measurement pulse input
Read from timer	When timer Bi register is read, it indicates the reload register's content (measurement result) (Note 2)
Write to timer	Cannot 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.

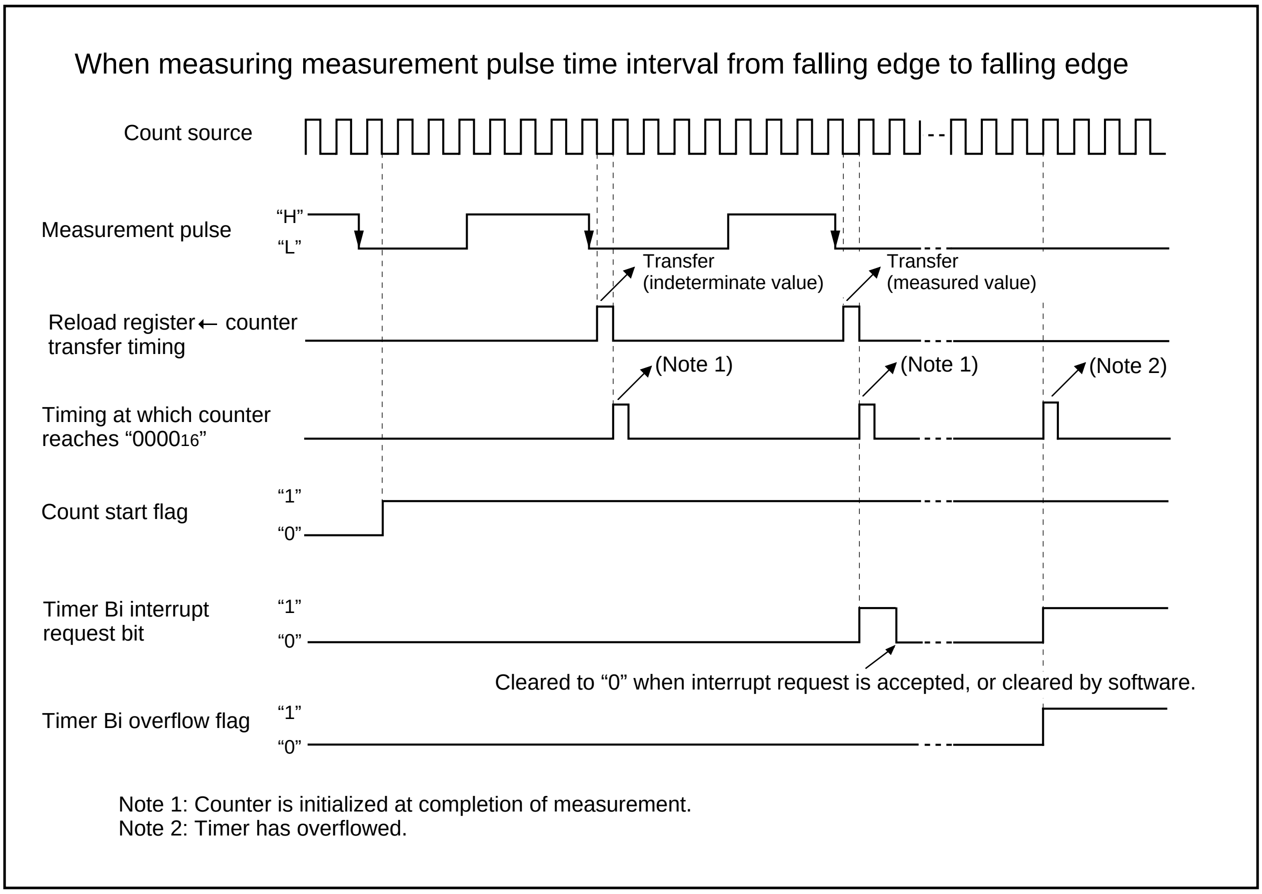


Symbol	Address	When reset
TBiMR(i=0 to 2)	039B$_{16}$ to 039D$_{16}$	00XX0000$_2$
(i=3 to 5)	035B$_{16}$ to 035D$_{16}$	00XX0000$_2$

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 : Must not be set	○	○
MR1			○	○
MR2	0 (Must always be "0" in pulse period/pulse width measurement mode; i = 0, 3)		○ (Note 2)	○
	Nothing is assigned (i = 1, 2, 4, 5). In an attempt to write to this bit, write "0". The value, if read, turns out to be indeterminate.		× (Note 3)	×
MR3	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_1 0 1 : f_8 1 0 : f_{32} 1 1 : f_{C132}	○	○
TCK1			○	○

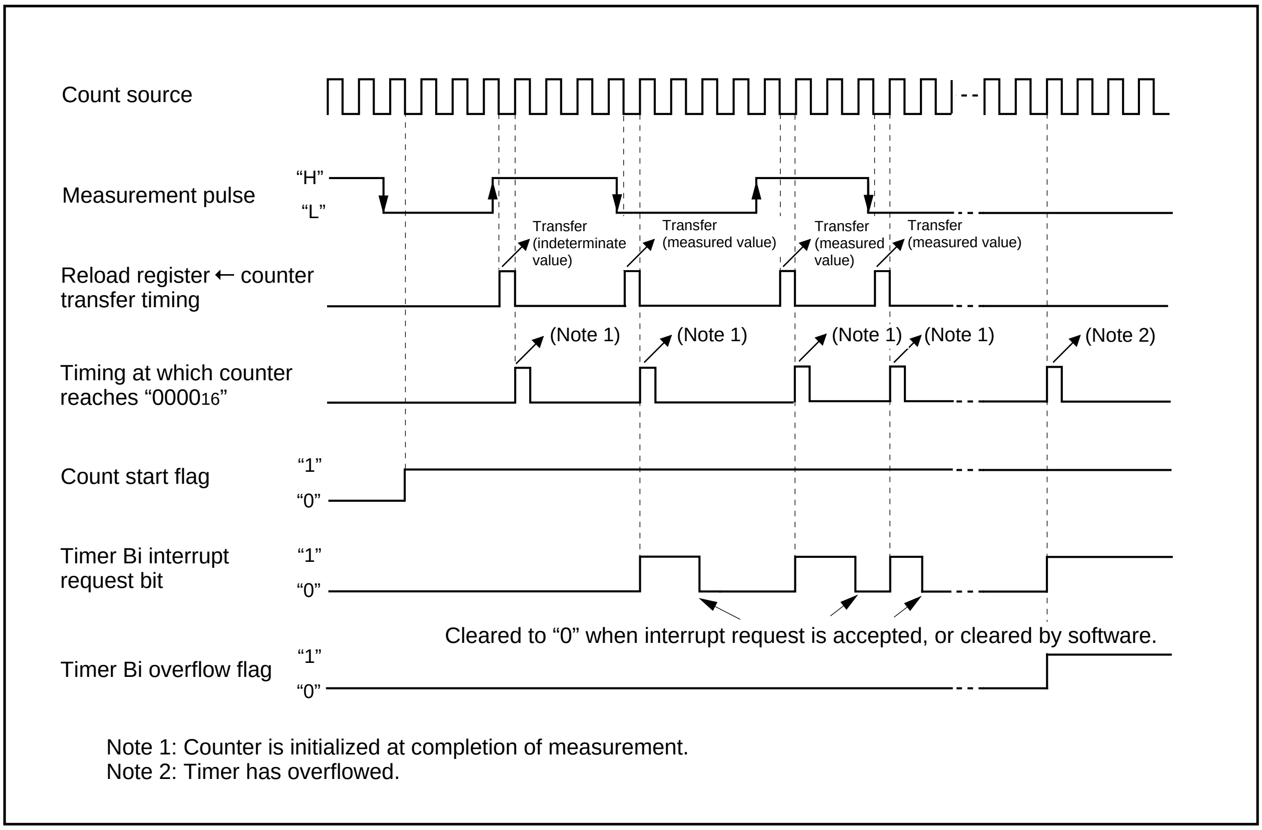
Note 1: The timer Bi overflow flag changes to "0" when the count start flag is "1" and a value is written to the timer Bi mode register. This flag cannot be set to "1" by software.
Note 2: Timer B0, timer B3.
Note 3: Timer B1, timer B2, timer B4, timer B5.

Figure 1.13.21. 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 1.13.22. 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 1.13.23. Operation timing when measuring a pulse width

Real time Port

When real time port output is selected, the real time port data written to the port Pm register is latched into the real time port latch each time the corresponding timer Ai underflows, with the data output from each corresponding port. The real time port data is written to the corresponding port Pm register. When the real time port mode select bit changes state from “0” to “1”, the value of the real time port latch becomes “0”, which is output from the corresponding pin. It is when timer Ai underflows first that the real time port data is output. If the real time port data is modified when the real time port function is enabled, the modified value is output when timer Ai underflows next time. The port functions as an ordinary port when the real time port function is disabled.

Make sure timer Ai for real time port output is set for timer mode, and is set to have “no gate function” using the gate function select bit. Also, before setting the real time port mode select bit to “1”, temporarily turn off the timer Ai used and write its set value to the timer Ai register. Figure 1.14.1 shows the block diagram for real time port output. Figure 1.14.2 shows the real time control register.

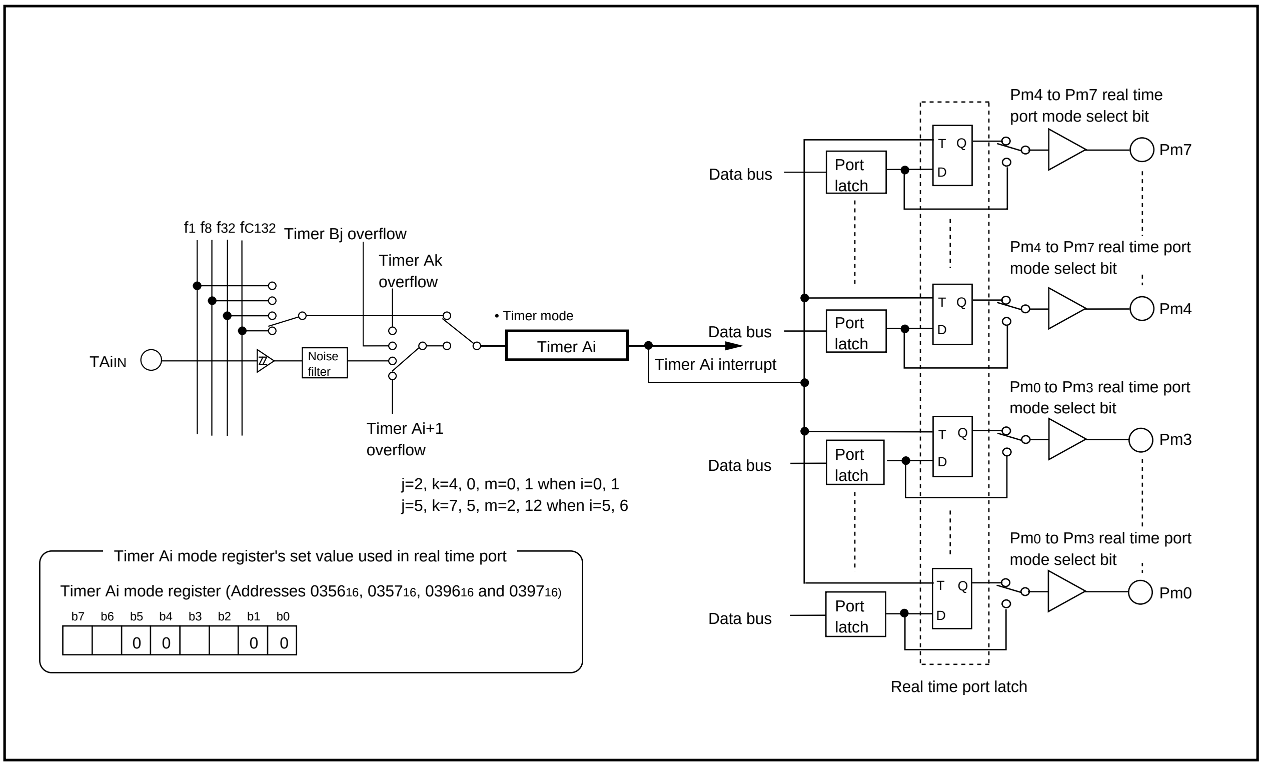
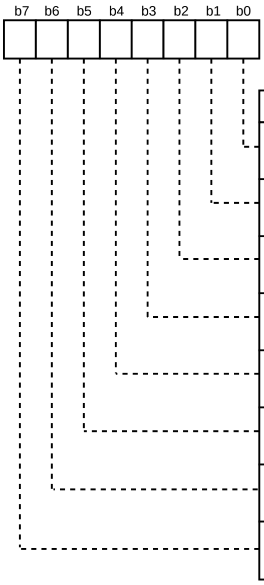


Figure 1.14.1. Block diagram for real time port output

Real time port control register



Symbol	Address	When reset
RTP	03FF$_{16}$	00$_{16}$

Bit symbol	Bit name	Function	R	W
RTP0	$P0_0$ to $P0_3$ real time port mode select bit	0 : I/O port 1 : Real time port output (Note)	○	○
RTP1	$P0_4$ to $P0_7$ real time port mode select bit		○	○
RTP2	$P1_0$ to $P1_3$ real time port mode select bit		○	○
RTP3	$P1_4$ to $P1_7$ real time port mode select bit		○	○
RTP4	$P2_0$ to $P2_3$ real time port mode select bit		○	○
RTP5	$P2_4$ to $P2_7$ real time port mode select bit		○	○
RTP6	$P12_0$ to $P12_3$ real time port mode select bit		○	○
RTP7	$P12_4$ to $P12_7$ real time port mode select bit		○	○

Note : The corresponding port direction register is invalidated.

Figure 1.14.2. Real time port control register

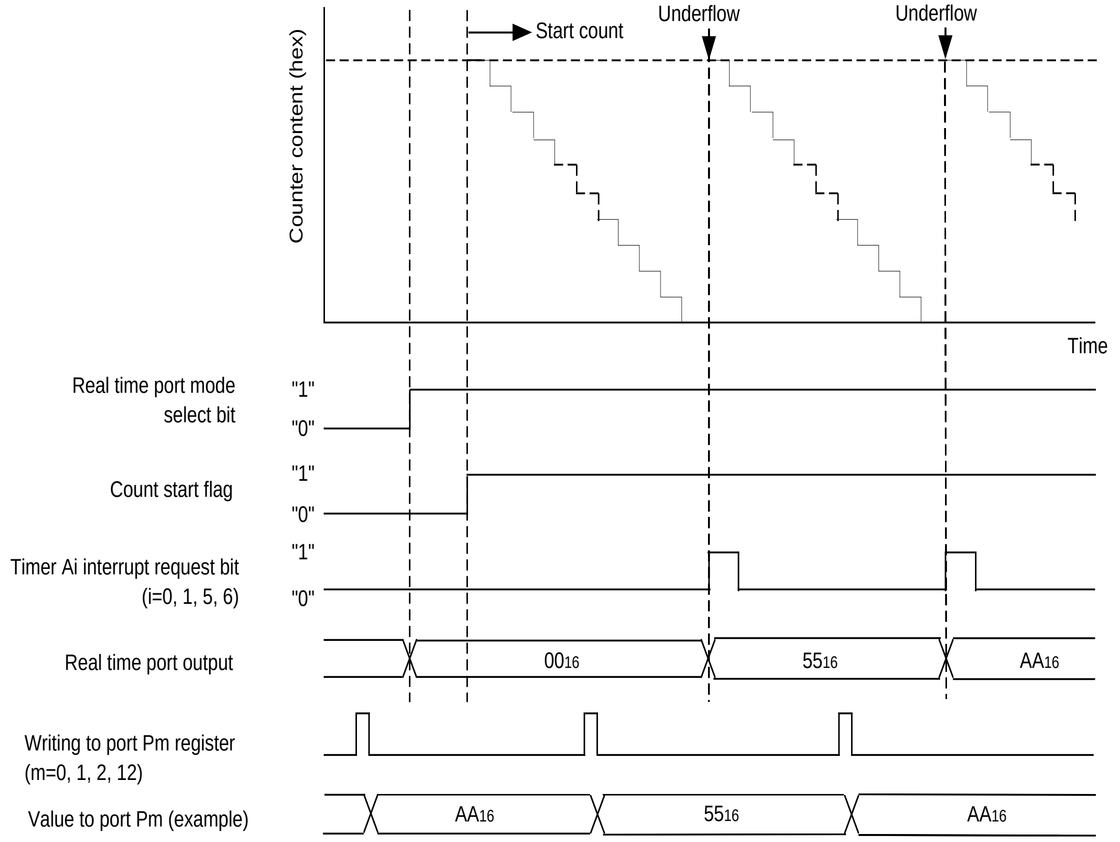

Start count
Underflow
Underflow
Counter content (hex)
Time
Real time port mode select bit
"1"
"0"
Count start flag
"1"
"0"
Timer Ai interrupt request bit (i=0, 1, 5, 6)
"1"
"0"
Real time port output
00₁₆
55₁₆
AA₁₆
Writing to port Pm register (m=0, 1, 2, 12)
Value to port Pm (example)
AA₁₆
55₁₆
AA₁₆

Figure 1.14.3. Timing in real time port output operation

Serial I/O

Serial I/O is configured as three channels: UART0, UART1, UART2.

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 1.15.1 shows the block diagram of UART0, UART1 and UART2. Figures 1.15.2 and 1.15.3 show the block diagram of the transmit/receive unit.

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

UART2, in particular, is used for 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 1.15.1 shows the comparison of functions of UART0 through UART2, and Figures 1.15.4 to 1.15.8 show the registers related to UARTi.

Note: SIM : Subscriber Identity Module

Table 1.15.1. Comparison of functions of UART0 through UART2

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

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

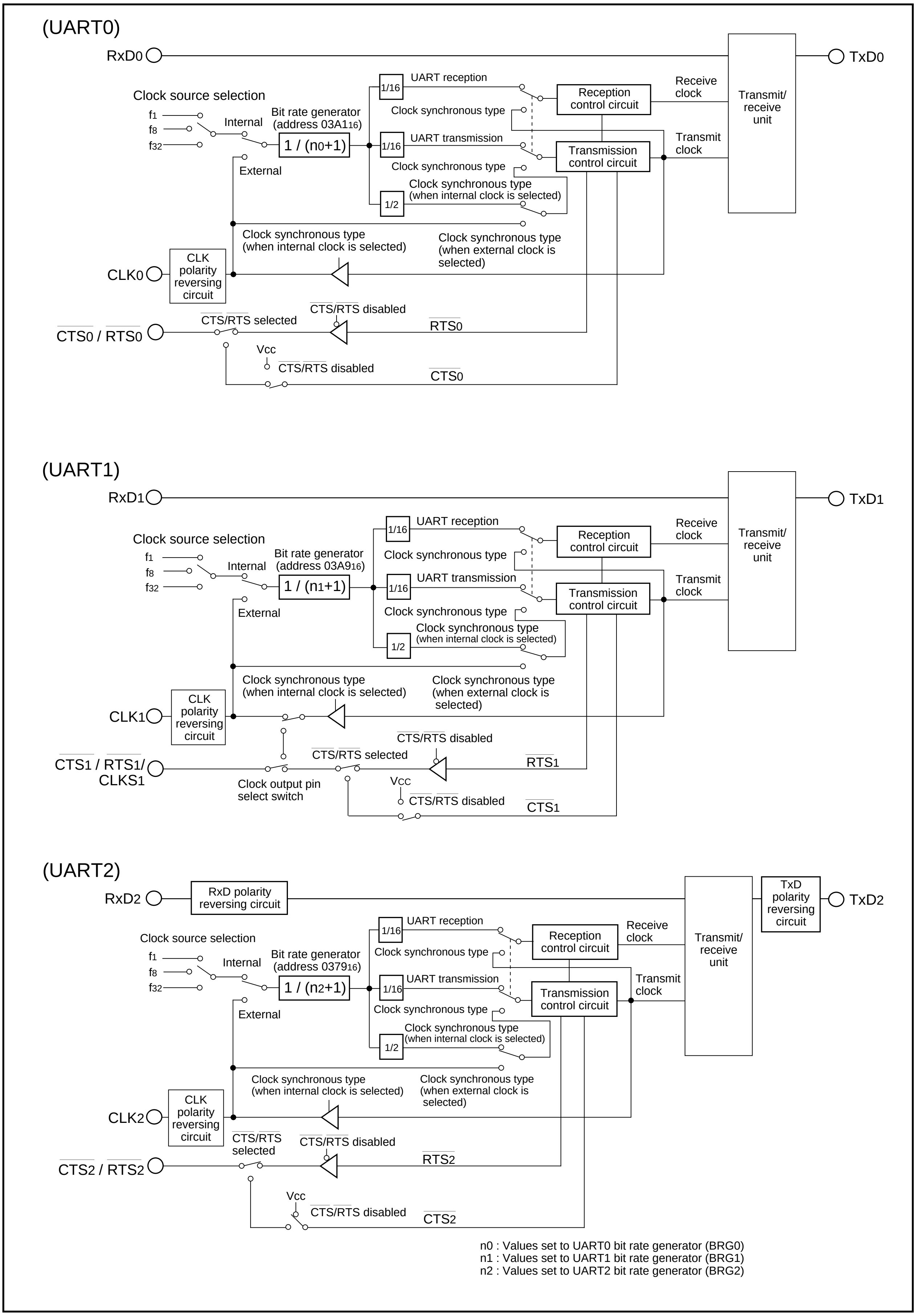

(UART0)
RxD0
TxD0
Clock source selection
f1
f8
f32
Internal
External
Bit rate generator (address 03A116)
1 / (n0+1)
1/16
UART reception
Clock synchronous type
1/16
UART transmission
Clock synchronous type
1/2
Clock synchronous type (when internal clock is selected)
Reception control circuit
Transmission 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
(UART1)
RxD1
TxD1
Clock source selection
f1
f8
f32
Internal
External
Bit rate generator (address 03A916)
1 / (n1+1)
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
CLK1
CLK polarity reversing circuit
Clock synchronous type (when internal clock is selected)
Clock synchronous type (when external clock is selected)
CTS1 / RTS1/ CLKS1
Clock output pin select switch
CTS/RTS selected
CTS/RTS disabled
RTS1
VCC
CTS/RTS disabled
CTS1
(UART2)
RxD2
RxD polarity reversing circuit
TxD2
TxD polarity reversing circuit
Clock source selection
f1
f8
f32
Internal
External
Bit rate generator (address 037916)
1 / (n2+1)
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
CLK2
CLK polarity reversing circuit
Clock synchronous type (when internal clock is selected)
Clock synchronous type (when external clock is selected)
CTS2 / RTS2
CTS/RTS selected
CTS/RTS disabled
RTS2
Vcc
CTS/RTS disabled
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 1.15.1. Block diagram of UARTi (i = 0 to 2)

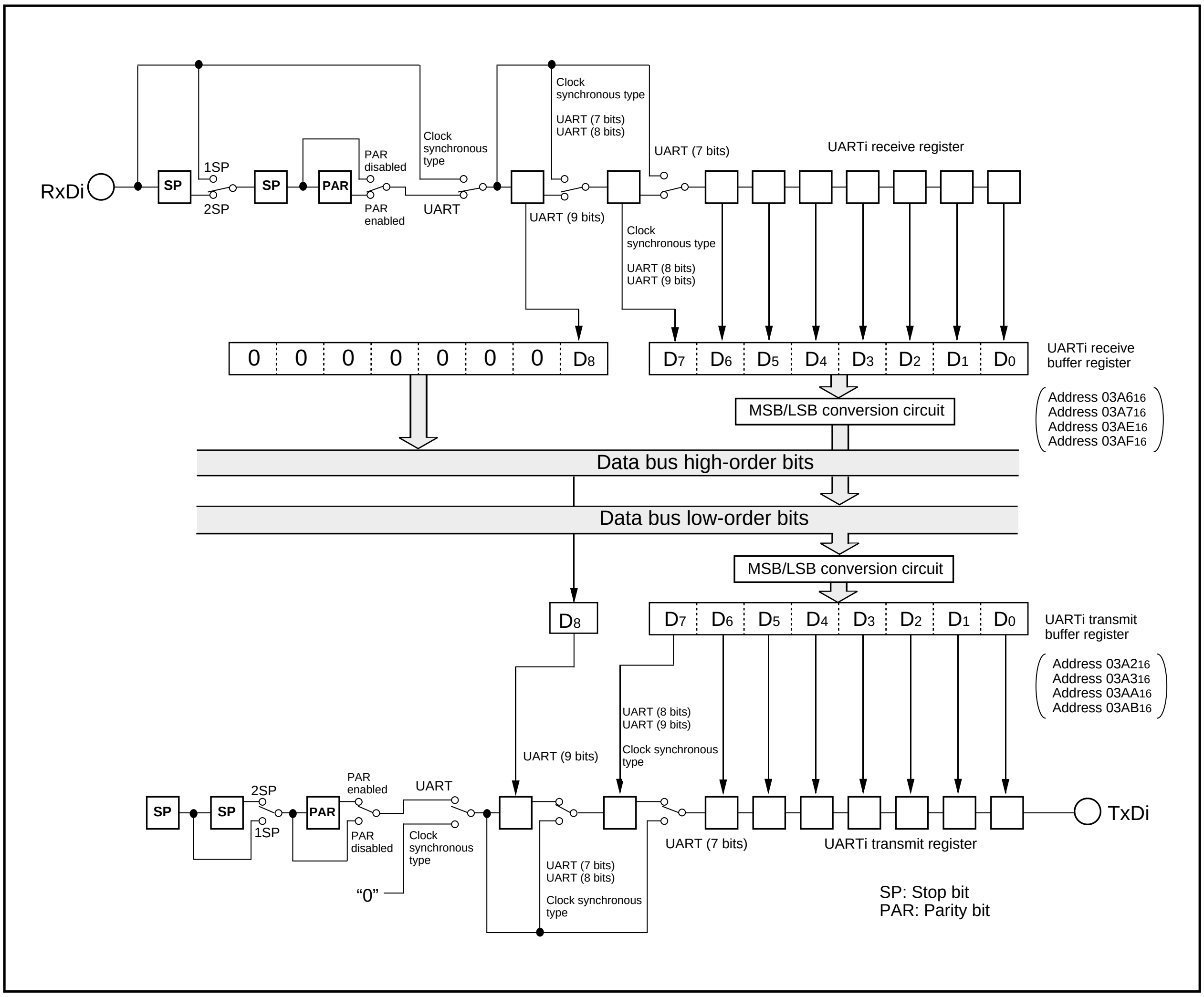


Figure 1.15.2. Block diagram of UARTi (i = 0, 1) transmit/receive unit

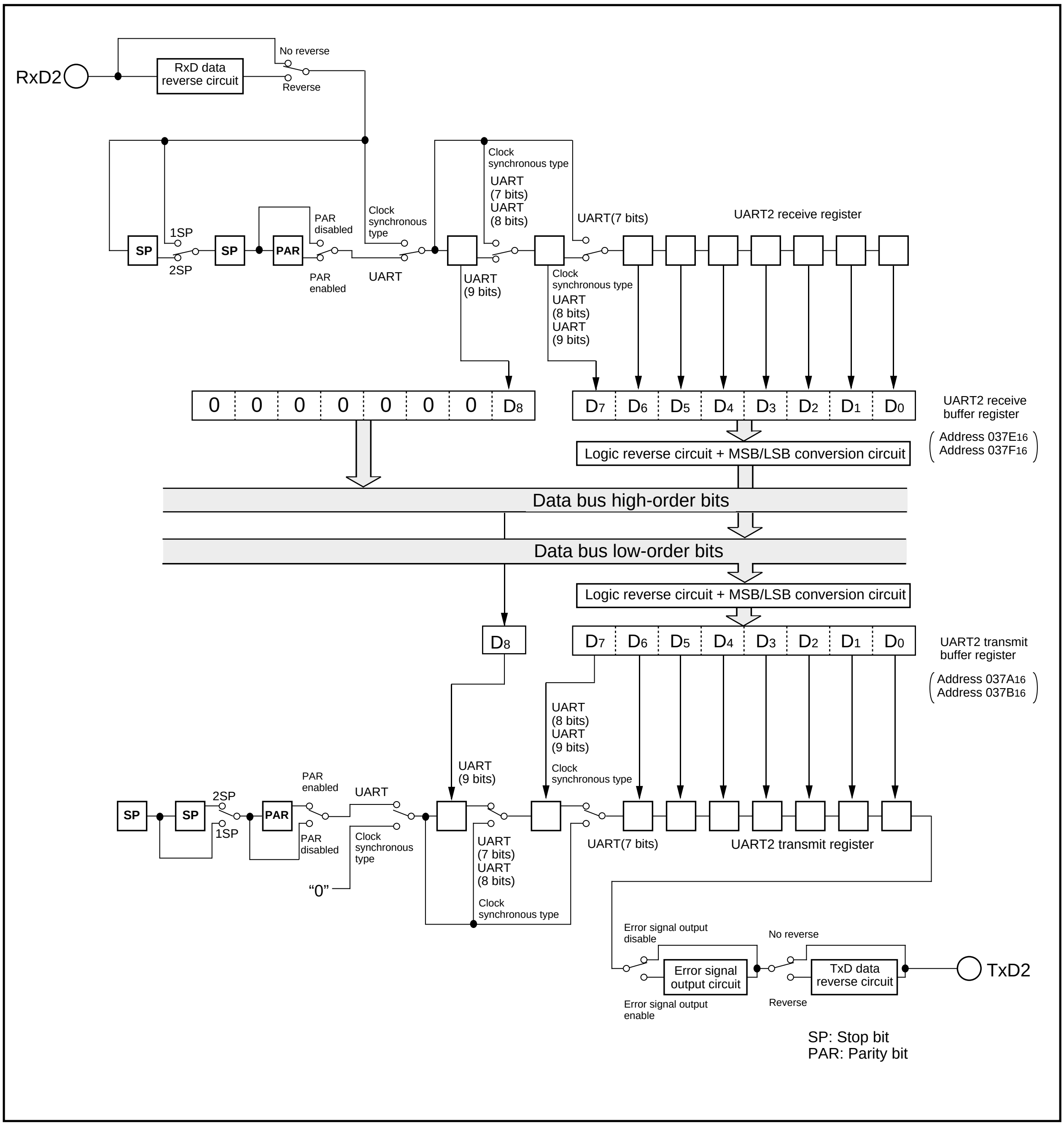
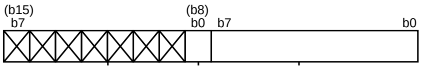


Figure 1.15.3. Block diagram of UART2 transmit/receive unit

UARTi transmit buffer register (Note)

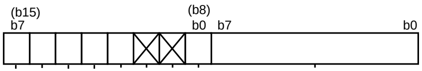


Symbol	Address	When reset
U0TB	$03A3_{16}$, $03A2_{16}$	Indeterminate
U1TB	$03AB_{16}$, $03AA_{16}$	Indeterminate
U2TB	$037B_{16}$, $037A_{16}$	Indeterminate

Function	R	W
Transmit data	×	○
Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turn out to be indeterminate.	–	–

Note: Use MOV instruction to write to this register.

UARTi receive buffer register

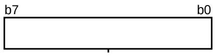


Symbol	Address	When reset
U0RB	$03A7_{16}$, $03A6_{16}$	Indeterminate
U1RB	$03AF_{16}$, $03AE_{16}$	Indeterminate
U2RB	$037F_{16}$, $037E_{16}$	Indeterminate

Bit symbol	Bit name	Function (During clock synchronous serial I/O mode)	Function (During UART mode)	R	W
—	——	Receive data	Receive data	○	×
Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”.				–	–
ABT	Arbitration lost detecting flag (Note 2)	0 : Not detected 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 0378_{16}) are set to “000_2” or the receive enable bit is set to “0”.
(Bit 15 is set to “0” when bits 14 to 12 all are set to “0”.) Bits 14 and 13 are also set to “0” when the lower byte of the UARTi receive buffer register (addresses $03A6_{16}$, $03AE_{16}$ and $037E_{16}$) is read out.
Note 2: Arbitration lost detecting flag is allocated to U2RB and noting but “0” may be written. Nothing is assigned in bit 11 of U0RB and U1RB. These bits can neither be set or reset. When read, the value of this bit is “0”.

UARTi bit rate generator (Note 1, Note 2)



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

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

Note 1: Write a value to this register while transmit/receive halts.
Note 2: Use MOV instruction to write to this register.

Figure 1.15.4. 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 : Must not be set 0 1 1 : Must not be set 1 1 1 : Must not be set	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 : Must not be set 0 1 1 : Must not be set 1 1 1 : Must not be set	○	○
SMD1				○	○
SMD2				○	○
CKDIR	Internal/external clock select bit	0 : Internal clock 1 : External clock (Note)	0 : Internal clock 1 : External clock (Note)	○	○
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	○	○

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

UART2 transmit/receive mode register

b7 b6 b5 b4 b3 b2 b1 b0

Symbol	Address	When reset
U2MR	0378_{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 1) 0 1 1 : Must not be set 1 1 1 : Must not be set	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 : Must not be set 0 1 1 : Must not be set 1 1 1 : Must not be set	○	○
SMD1				○	○
SMD2				○	○
CKDIR	Internal/external clock select bit	0 : Internal clock 1 : External clock (Note 2)	Must always be “0”	○	○
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 1: Bit 2 to bit 0 are set to “010_2” when I^{2}C mode is used.
Note 2: Set the corresponding port direction register to “0”.

Figure 1.15.5. 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_1 is selected 0 1 : f_8 is selected 1 0 : f_{32} is selected 1 1 : Must not be set	b1 b0 0 0 : f_1 is selected 0 1 : f_8 is selected 1 0 : f_{32} is selected 1 1 : Must not be set	○	○
CLK1				○	○
CRS	$\overline{\text{CTS}}$/$\overline{\text{RTS}}$ function select bit	Valid when bit 4 = “0” 0 : $\overline{\text{CTS}}$ function is selected (Note 1) 1 : $\overline{\text{RTS}}$ function is selected (Note 2)	Valid when bit 4 = “0” 0 : $\overline{\text{CTS}}$ function is selected (Note 1) 1 : $\overline{\text{RTS}}$ 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	$\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit	0 : $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ function enabled 1 : CTS/RTS function disabled (P6$_0$ and P6$_4$ function as programmable I/O port)	0 : $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ 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

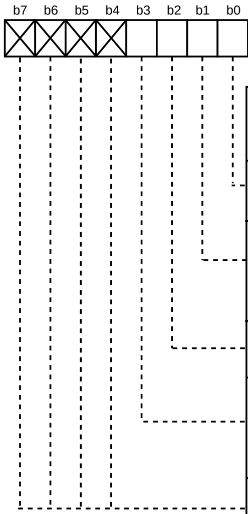
Symbol	Address	When reset
U2C0	037C$_{16}$	08$_{16}$

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_1 is selected 0 1 : f_8 is selected 1 0 : f_{32} is selected 1 1 : Must not be set	b1 b0 0 0 : f_1 is selected 0 1 : f_8 is selected 1 0 : f_{32} is selected 1 1 : Must not be set	○	○
CLK1				○	○
CRS	$\overline{\text{CTS}}$/$\overline{\text{RTS}}$ function select bit	Valid when bit 4 = “0” 0 : $\overline{\text{CTS}}$ function is selected (Note 1) 1 : $\overline{\text{RTS}}$ function is selected (Note 2)	Valid when bit 4 = “0” 0 : $\overline{\text{CTS}}$ function is selected (Note 1) 1 : $\overline{\text{RTS}}$ 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	$\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit	0 : $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ function enabled 1 : CTS/RTS function disabled (P7$_3$ functions programmable I/O port)	0 : $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ function enabled 1 : CTS/RTS function disabled (P7$_3$ functions programmable I/O port)	○	○
Nothing is assigned. In an attempt to write to this bit, write “0”. The value, if read, turns out to be “0”.				—	—
CKPOL	CLK polarity select bit	0 : Transmit data is output at falling edge of transfer clock and receive data is input at rising edge 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.

Figure 1.15.6. Serial I/O-related registers (3)

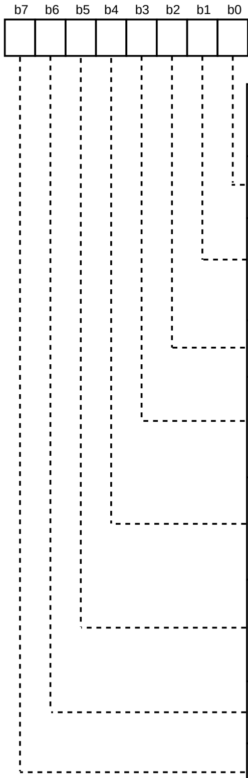
UARTi transmit/receive control register 1



Symbol	Address	When reset
UiC1(i=0,1)	$03A5_{16}$,$03AD_{16}$	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. In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”.				–	–

UART2 transmit/receive control register 1



Symbol	Address	When reset
U2C1	$037D_{16}$	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	Must always be “0”	○	○
U2LCH	Data logic select bit	0 : No reverse 1 : Reverse	0 : No reverse 1 : Reverse	○	○
U2ERE	Error signal output enable bit	Must always be “0”	0 : Output disabled 1 : Output enabled	○	○

Figure 1.15.7. Serial I/O-related registers (4)

UART transmit/receive control register 2

b7	b6	b5	b4	b3	b2	b1	b0
☒	0						

Symbol UCON Address $03B0_{16}$ When reset $X0000000_2$

Bit symbol	Bit name	Function (During clock synchronous serial I/O mode)	Function (During UART mode)	R	W
U0IRS	UART0 transmit interrupt cause select bit	0 : Transmit buffer empty (TI = 1) 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	Must always be “0”	○	○
U1RRM	UART1 continuous receive mode enable bit	0 : Continuous receive mode disabled 1 : Continuous receive mode enabled	Must always be “0”	○	○
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”	○	○
Reserved bit		Must always be set to “0”		○	○
Nothing is assigned. In an attempt to write to this bit, write “0”. The value, if read, turns out to be indeterminate.				–	–

Note: When using multiple pins to output the transfer clock, the following requirements must be met:
- UART1 internal/external clock select bit (bit 3 at address $03A8_{16}$) = “0”.

UART2 special mode register

b7	b6	b5	b4	b3	b2	b1	b0
0							

Symbol U2SMR Address 0377_{16} When reset 00_{16}

Bit symbol	Bit name	Function (During clock synchronous serial I/O mode)	Function (During UART mode)	R	W
IICM	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	○	○
Reserved bit		Must always be set to “0”		–	○

Note: Nothing but "0" may be written.

Figure 1.15.8. 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 1.15.2 and 1.15.3 list the specifications of the clock synchronous serial I/O mode. Figure 1.15.9 shows the UARTi transmit/receive mode register.

Table 1.15.2. Specifications of clock synchronous serial I/O mode (1)

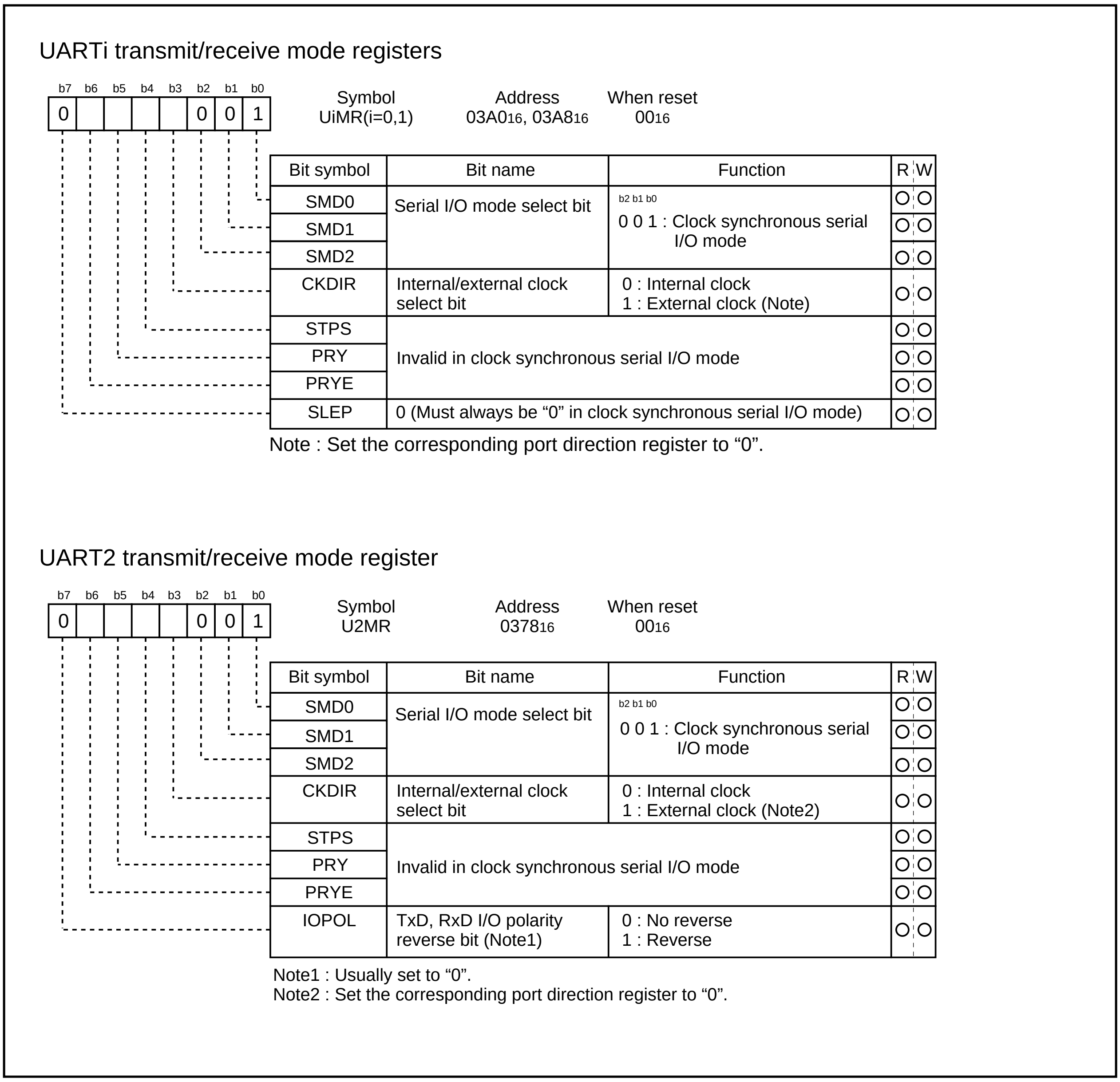
Item	Specification
Transfer data format	• Transfer data length: 8 bits
Transfer clock	• When internal clock is selected (bit 3 at addresses $03A0_{16}$, $03A8_{16}$, 0378_{16} = "0") : fi/ 2(n+1) (Note 1) fi = f_1, f_8, f_{32} • When external clock is selected (bit 3 at addresses $03A0_{16}$, $03A8_{16}$,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

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

Note 2: If an overrun error occurs, the UARTi receive buffer will have the next data written in. Note also that the UARTi receive interrupt request bit does not change.

Table 1.15.3. Specifications of clock synchronous serial I/O mode (2)

Item	Specification
Select function	• CLK polarity selection Whether transmit data is output/input timing at the rising edge or falling edge of thetransfer 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) UART1 transfer clock can be chosen by software to be output from one of the two pins set • 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.



UARTi transmit/receive mode registers

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 (Note)	○	○
STPS	Invalid in clock synchronous serial I/O mode		○	○
PRY			○	○
PRYE			○	○
SLEP	0 (Must always be “0” in clock synchronous serial I/O mode)		○	○

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

UART2 transmit/receive mode register

Symbol	Address	When reset
U2MR	0378$_{16}$	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 (Note2)	○	○
STPS	Invalid in clock synchronous serial I/O mode		○	○
PRY			○	○
PRYE			○	○
IOPOL	TxD, RxD I/O polarity reverse bit (Note1)	0 : No reverse 1 : Reverse	○	○

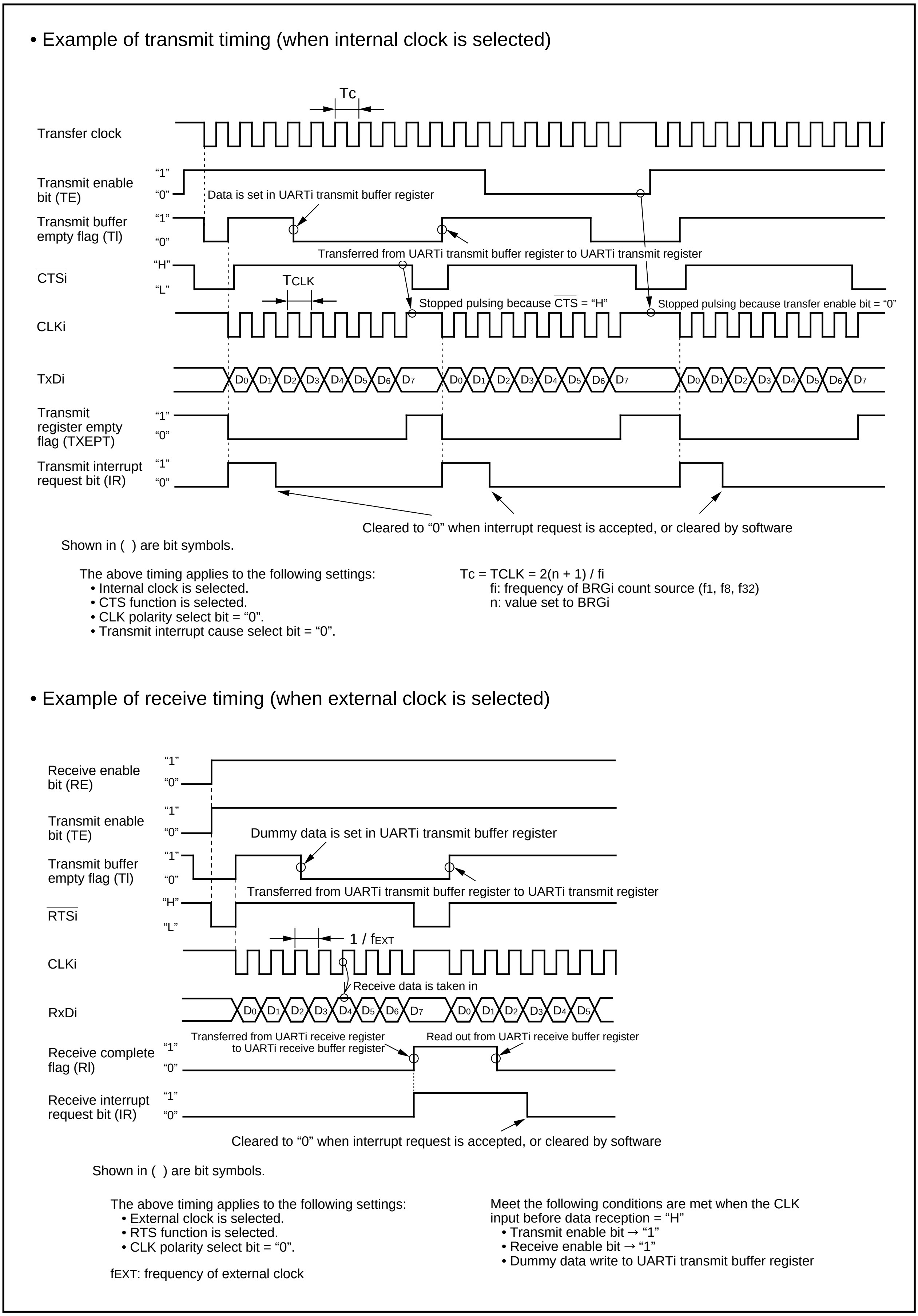
Note1 : Usually set to “0”.
Note2 : Set the corresponding port direction register to “0”.

Figure 1.15.9. UARTi transmit/receive mode register in clock synchronous serial I/O mode

Table 1.15.4 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 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 1.15.4. Input/output pin functions in clock synchronous serial I/O mode
(when transfer clock output from multiple pins is not selected)

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 $03A0_{16}$, $03A8_{16}$, 0378_{16}) = “0”
	Transfer clock input	Internal/external clock select bit (bit 3 at address $03A0_{16}$, $03A8_{16}$, 0378_{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 $03A4_{16}$, $03AC_{16}$, $037C_{16}$) =“0” CTS/RTS function select bit (bit 2 at address $03A4_{16}$, $03AC_{16}$, $037C_{16}$) = “0” Port $P6_0$, $P6_4$ and $P7_3$ direction register (bits 0 and 4 at address $03EE_{16}$, bit 3 at address $03EF_{16}$) = “0”
	$\overline{\text{RTS}}$ output	$\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit (bit 4 at address $03A4_{16}$, $03AC_{16}$, $037C_{16}$) = “0” $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ function select bit (bit 2 at address $03A4_{16}$, $03AC_{16}$, $037C_{16}$) = “1”
	Programmable I/O port	$\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit (bit 4 at address $03A4_{16}$, $03AC_{16}$, $037C_{16}$) = “1”



• Example of transmit timing (when 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 (f_1, f_8, f_{32})
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".

f_{EXT}: 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 1.15.10. Typical transmit/receive timings in clock synchronous serial I/O mode

(a) Polarity select function

As shown in Figure 1.15.11, the CLK polarity select bit (bit 6 at addresses 03A4$_{16}$, 03AC$_{16}$, 037C$_{16}$) allows selection of the polarity of the transfer clock.

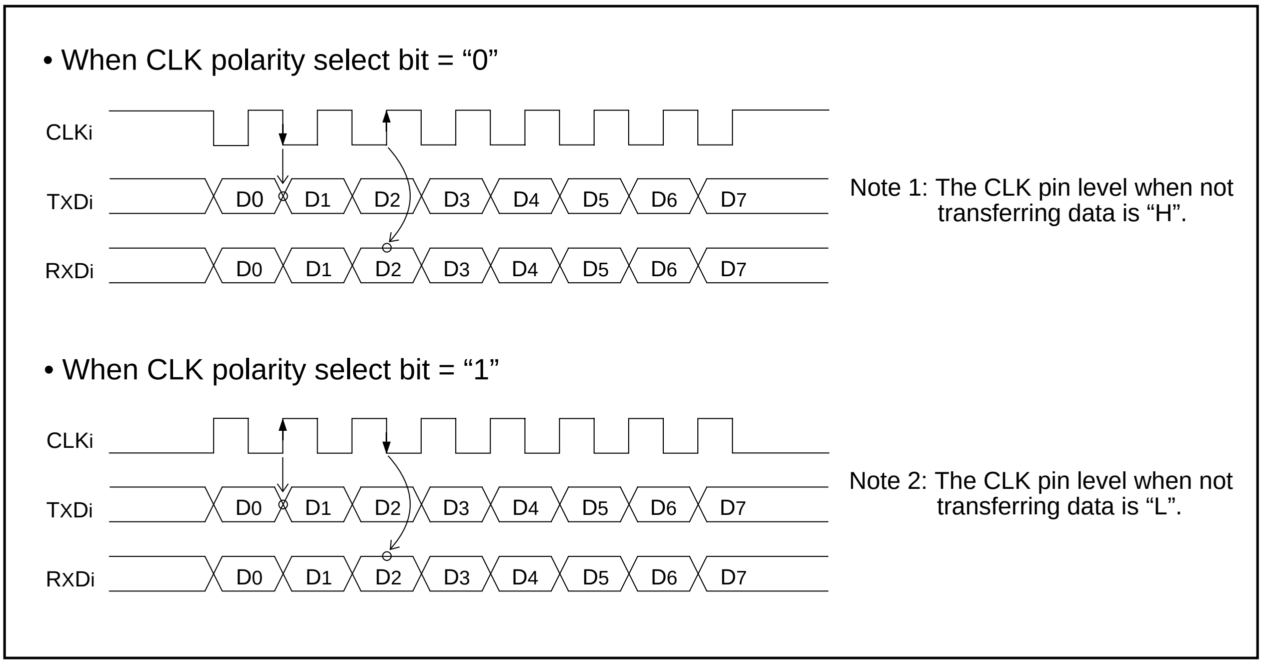

• When CLK polarity select bit = “0”
CLKi
TXDi
RXDi
D0
D1
D2
D3
D4
D5
D6
D7
Note 1: The CLK pin level when not transferring data is “H”.
• When CLK polarity select bit = “1”
CLKi
TXDi
RXDi
D0
D1
D2
D3
D4
D5
D6
D7
Note 2: The CLK pin level when not transferring data is “L”.

Figure 1.15.11. Polarity of transfer clock

(b) LSB first/MSB first select function

As shown in Figure 1.15.12, when the transfer format select bit (bit 7 at addresses 03A4$_{16}$, 03AC$_{16}$, 037C$_{16}$) = “0”, the transfer format is “LSB first”; when the bit = “1”, the transfer format is “MSB first”.

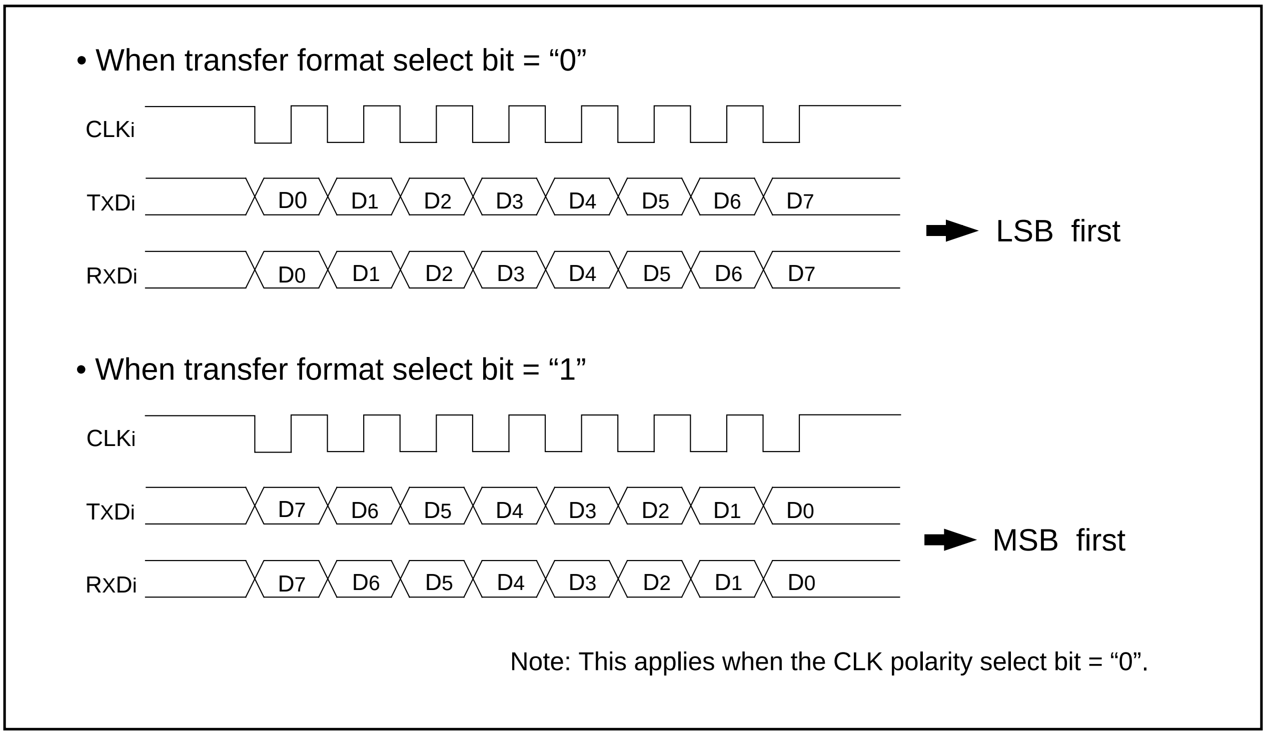
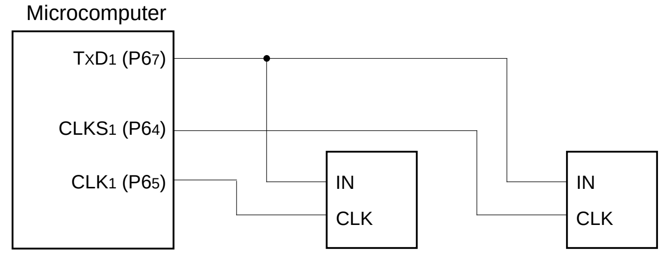

• When transfer format select bit = “0”
CLKi
TXDi
RXDi
D0
D1
D2
D3
D4
D5
D6
D7
LSB first
• When transfer format select bit = “1”
CLKi
TXDi
RXDi
D7
D6
D5
D4
D3
D2
D1
D0
MSB first
Note: This applies when the CLK polarity select bit = “0”.

Figure 1.15.12. 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 1.15.3.) 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 1.15.13. 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 $037D_{16}$) is set to “1”, the unit is placed in continuous receive mode. In this mode, when the receive buffer register is read out, the unit simultaneously goes to a receive enable state without having to set dummy data to the transmit buffer register back again.

(e) Serial data logic switch function (UART2)

When the data logic select bit (bit6 at address $037D_{16}$) = “1”, and writing to transmit buffer register or reading from receive buffer register, data is reversed. Figure 1.15.14 shows the example of serial data logic switch timing.

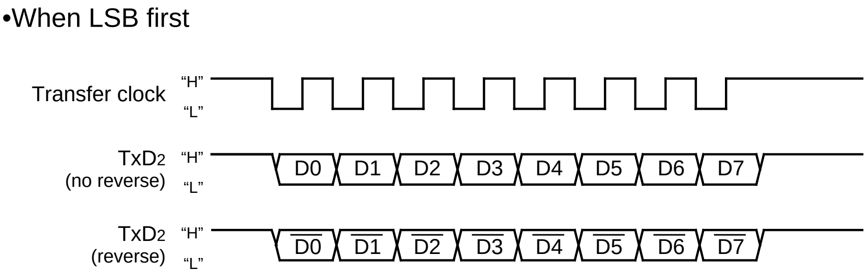

•When LSB first
Transfer clock
“H”
“L”
TxD2
(no reverse)
“H”
“L”
D0 D1 D2 D3 D4 D5 D6 D7
TxD2
(reverse)
“H”
“L”
$\overline{D0}$ $\overline{D1}$ $\overline{D2}$ $\overline{D3}$ $\overline{D4}$ $\overline{D5}$ $\overline{D6}$ $\overline{D7}$

Figure 1.15.14. 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 1.15.5 and 1.15.6 list the specifications of the UART mode. Figure 1.15.15 shows the UARTi transmit/receive mode register.

Table 1.15.5. 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 $03A0_{16}$, $03A8_{16}$, 0378_{16} = “0”) : fi/16(n+1) (Note 1) fi = f_1, f_8, f_{32} • When external clock is selected (bit 3 at addresses $03A0_{16}$, $03A8_{16}$ =“1”) : f_{EXT}/16(n+1) (Note 1) (Note 2) (Do not set external clock for UART2)
Transmission/reception control	• $\overline{\text{CTS}}$ function, $\overline{\text{RTS}}$ function, $\overline{\text{CTS}}$ and $\overline{\text{RTS}}$ function invalid: selectable
Transmission start condition	• To start transmission, the following requirements must be met: - Transmit enable bit (bit 0 at addresses $03A5_{16}$, $03AD_{16}$, $037D_{16}$) = “1” - Transmit buffer empty flag (bit 1 at addresses $03A5_{16}$, $03AD_{16}$, $037D_{16}$) = “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 $03A5_{16}$, $03AD_{16}$, $037D_{16}$) = “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 $037D_{16}$) = “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 $037D_{16}$) = “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.

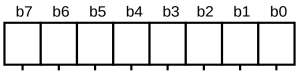
Note 2: f_{EXT} is input from the CLKi pin.

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 does not change.

Table 1.15.6. Specifications of UART Mode (2)

Item	Specification
Select function	• 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

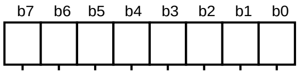


Symbol	Address	When reset
UiMR(i=0,1)	$03A0_{16}$, $03A8_{16}$	00_{16}

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 (Note)	○	○
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	○	○

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

UART2 transmit / receive mode register



Symbol	Address	When reset
U2MR	0378_{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	Must always be “0”	○	○
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”.

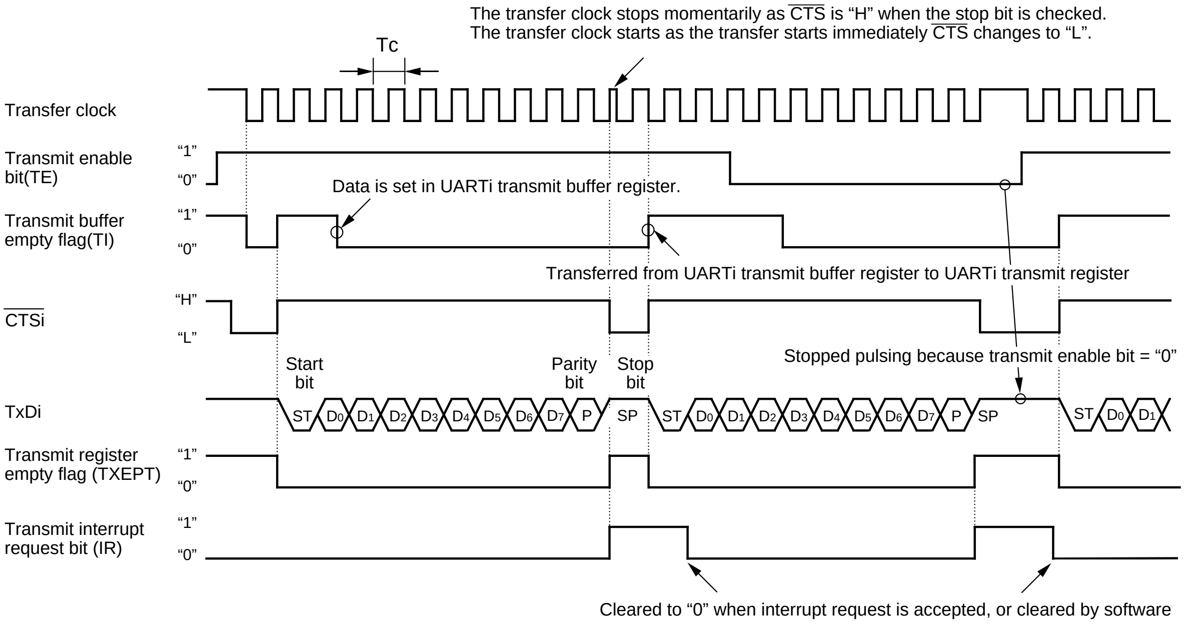
Figure 1.15.15. UARTi transmit/receive mode register in UART mode

Table 1.15.7 lists the functions of the input/output pins during UART mode. Note that for a period from when the UARTi operation mode is selected to when transfer starts, the TxDi pin outputs a “H”. (If the N-channel open-drain is selected, this pin is in floating state.)

Table 1.15.7. Input/output pin functions in UART mode

Pin name	Function	Method of selection
TxDi ($P6_3$, $P6_7$, $P7_0$)	Serial data output	
RxDi ($P6_2$, $P6_6$, $P7_1$)	Serial data input	Port $P6_2$, $P6_6$ and $P7_1$ direction register (bits 2 and 6 at address $03EE_{16}$, bit 1 at address $03EF_{16}$)= “0” (Can be used as an input port when performing transmission only)
CLKi ($P6_1$, $P6_5$, $P7_2$)	Programmable I/O port	Internal/external clock select bit (bit 3 at address $03A0_{16}$, $03A8_{16}$, 0378_{16}) = “0”
	Transfer clock input	Internal/external clock select bit (bit 3 at address $03A0_{16}$, $03A8_{16}$) = “1” Port $P6_1$, $P6_5$ direction register (bits 1 and 5 at address $03EE_{16}$) = “0” (Do not set external clock for UART2)
$\overline{\text{CTS}}$i/$\overline{\text{RTS}}$i ($P6_0$, $P6_4$, $P7_3$)	$\overline{\text{CTS}}$ input	$\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit (bit 4 at address $03A4_{16}$, $03AC_{16}$, $037C_{16}$) =“0” $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ function select bit (bit 2 at address $03A4_{16}$, $03AC_{16}$, $037C_{16}$) = “0” Port $P6_0$, $P6_4$ and $P7_3$ direction register (bits 0 and 4 at address $03EE_{16}$, bit 3 at address $03EF_{16}$) = “0”
	$\overline{\text{RTS}}$ output	$\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit (bit 4 at address $03A4_{16}$, $03AC_{16}$, $037C_{16}$) = “0” $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ function select bit (bit 2 at address $03A4_{16}$, $03AC_{16}$, $037C_{16}$) = “1”
	Programmable I/O port	$\overline{\text{CTS}}$/$\overline{\text{RTS}}$ disable bit (bit 4 at address $03A4_{16}$, $03AC_{16}$, $037C_{16}$) = “1”

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

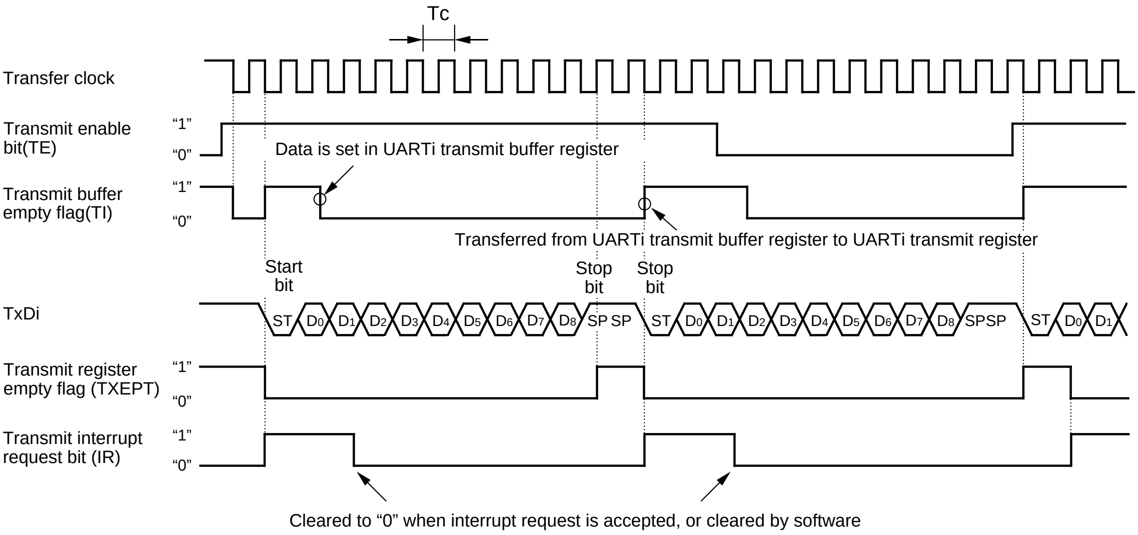

The transfer clock stops momentarily as $\overline{CTS}$ is "H" when the stop bit is checked.
The transfer clock starts as the transfer starts immediately $\overline{CTS}$ changes to "L".
Tc
Transfer clock
Transmit enable bit(TE)
"1"
"0"
Data is set in UARTi transmit buffer register.
Transmit buffer empty flag(TI)
"1"
"0"
Transferred from UARTi transmit buffer register to UARTi transmit register
$\overline{CTSi}$
"H"
"L"
Start bit
Parity bit
Stop bit
Stopped pulsing because transmit enable bit = "0"
TxDi
ST D0 D1 D2 D3 D4 D5 D6 D7 P SP ST D0 D1 D2 D3 D4 D5 D6 D7 P SP ST D0 D1
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 :
- 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) / f_{EXT}
fi : frequency of BRGi count source (f_1, f_8, f_{32})
f_{EXT} : frequency of BRGi count source (external clock)
n : value set to BRGi

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


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
Start bit
Stop bit
Stop bit
TxDi
ST D0 D1 D2 D3 D4 D5 D6 D7 D8 SP SP ST D0 D1 D2 D3 D4 D5 D6 D7 D8 SPSP ST D0 D1
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

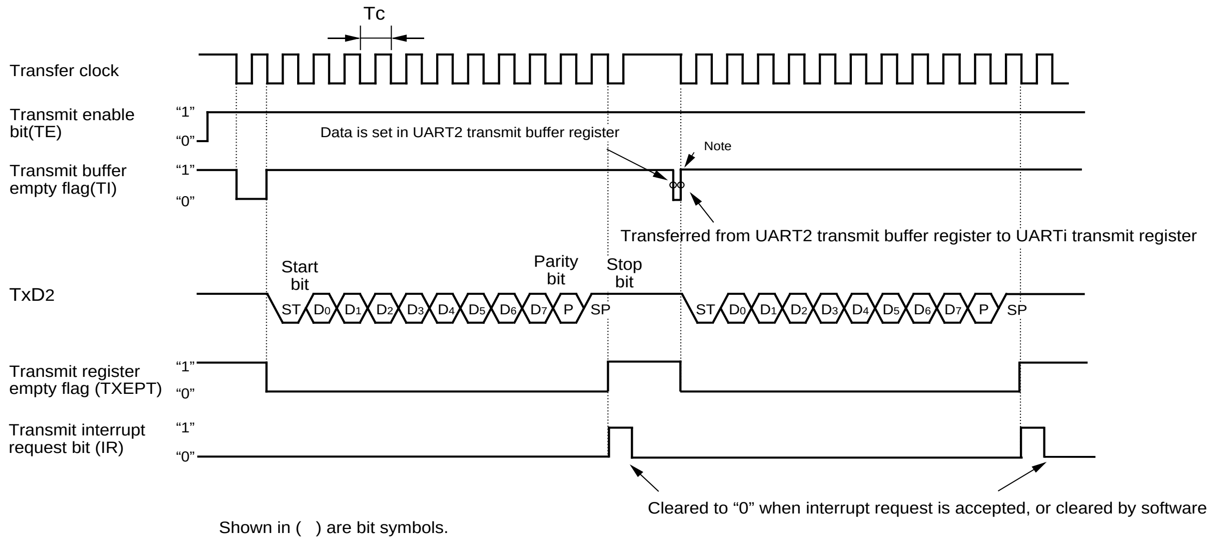
Shown in ( ) are bit symbols.

The above timing applies to the following settings :
- 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) / f_{EXT}
fi : frequency of BRGi count source (f_1, f_8, f_{32})
f_{EXT} : frequency of BRGi count source (external clock)
n : value set to BRGi

Figure 1.15.16. Typical transmit timings in UART mode (UART0,UART1)

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


Tc
Transfer clock
Transmit enable bit(TE)
"1"
"0"
Data is set in UART2 transmit buffer register
Note
Transmit buffer empty flag(TI)
"1"
"0"
Transferred from UART2 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
ST D0 D1 D2 D3 D4 D5 D6 D7 P SP
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.

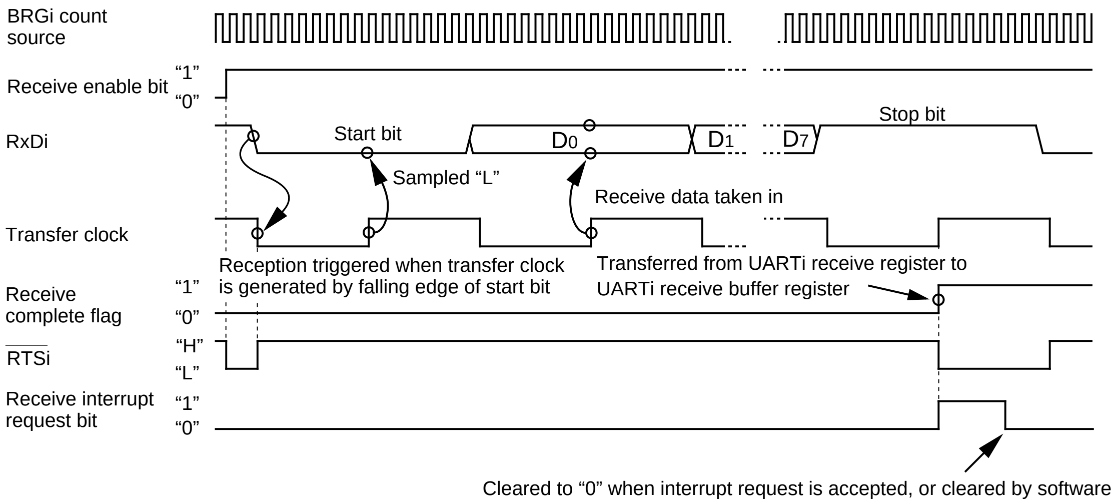
The above timing applies to the following settings :
- Parity is enabled.
- One stop bit.
- Transmit interrupt cause select bit = "1".

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

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

Figure 1.15.17. Typical transmit timings in UART mode (UART2)

• 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 1.15.18. Typical receive timing in UART mode

(a) 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".

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

When the data logic select bit (bit 6 of address 037D16) is assigned 1, data is inverted in writing to the transmission buffer register or reading the reception buffer register. Figure 1.15.19 shows the example of timing for switching serial data logic.

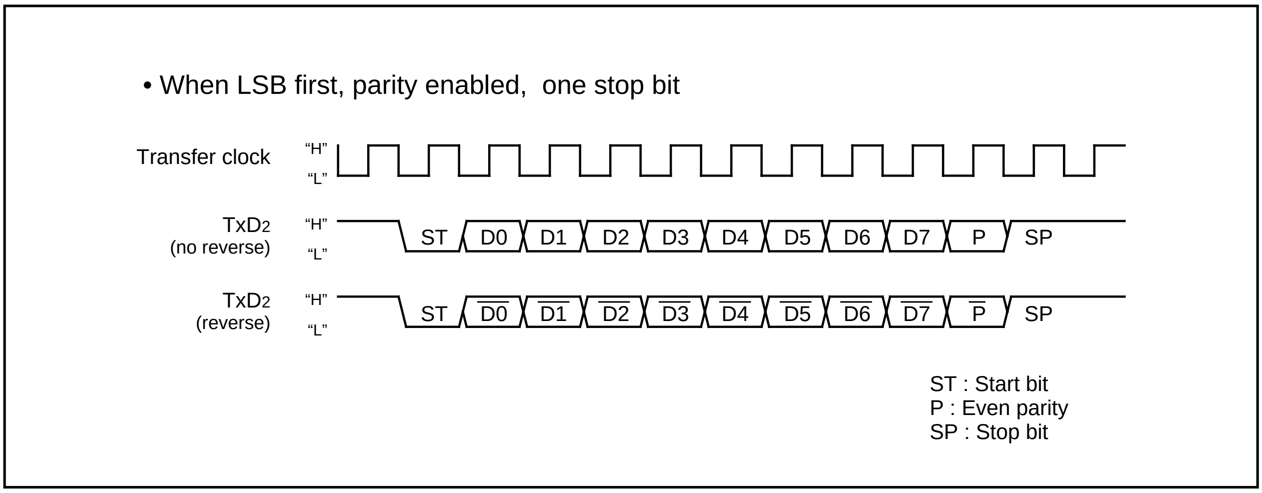


Figure 1.15.19. Timing for switching serial data logic

(c) 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.

(d) Bus collision detection function (UART2)

This function is to sample the output level of the TXD pin and the input level of the RXD pin at the rising edge of the transfer clock; if their values are different, then an interrupt request occurs. Figure 1.15.20 shows the example of detection timing of a bus collision (in UART mode).

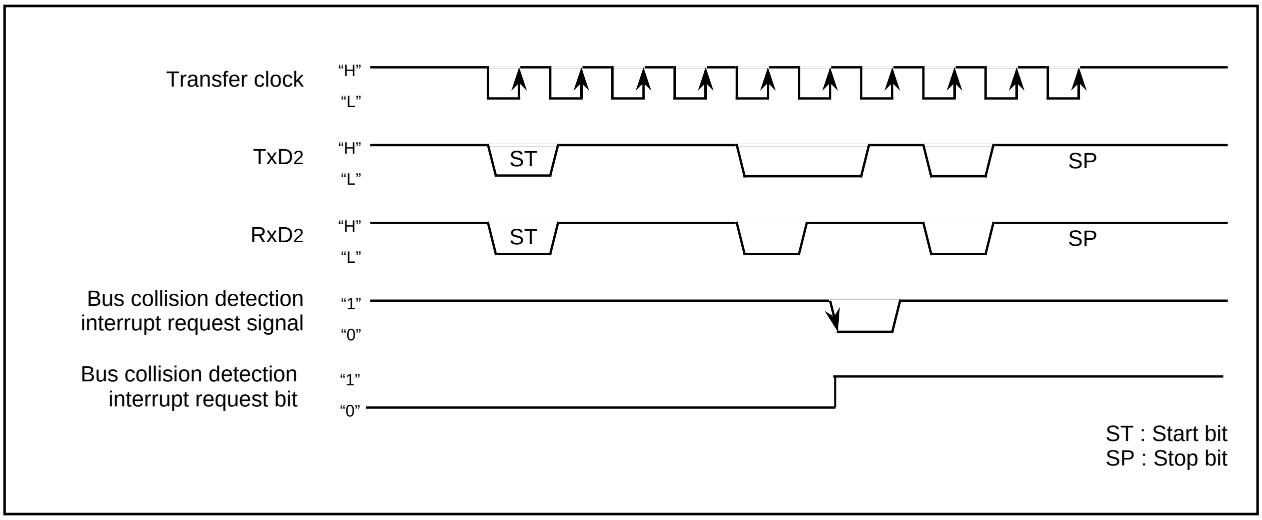

Transfer clock
"H"
"L"
TxD2
ST
SP
RxD2
ST
SP
Bus collision detection interrupt request signal
"1"
"0"
Bus collision detection interrupt request bit
ST : Start bit
SP : Stop bit

Figure 1.15.20. Detection timing of a bus collision (in UART mode)

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

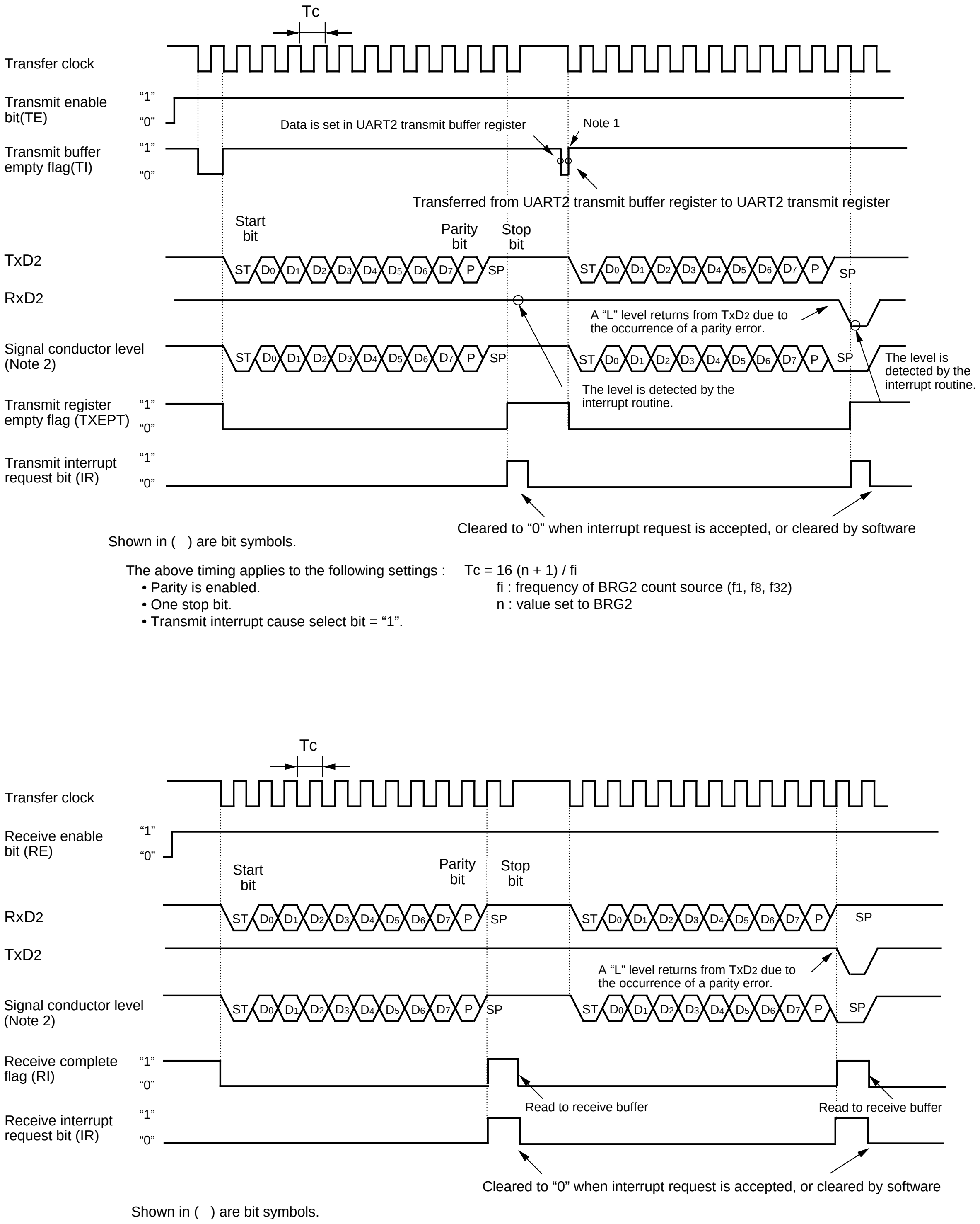
The SIM interface is used for connecting the microcomputer with a memory card or the like; adding some extra settings in UART2 clock-asynchronous serial I/O mode allows the user to effect this function. Table 1.15.8 shows the specifications of clock-asynchronous serial I/O mode (compliant with the SIM interface).

Table 1.15.8. Specifications of clock-asynchronous serial I/O mode (compliant with the SIM interface)

Item	Specification
Transfer data format	• Transfer data 8-bit UART mode (bit 2 through bit 0 of address 037_{16} = "101_2") • One stop bit (bit 4 of address 0378_{16} = "0") • With the direct format chosen Set parity to "even" (bit 5 and bit 6 of address 0378_{16} = "1" and "1" respectively) Set data logic to "direct" (bit 6 of address $037D_{16}$ = "0"). Set transfer format to LSB (bit 7 of address $037C_{16}$ = "0"). • With the inverse format chosen Set parity to "odd" (bit 5 and bit 6 of address 0378_{16} = "0" and "1" respectively) Set data logic to "inverse" (bit 6 of address $037D_{16}$ = "1") Set transfer format to MSB (bit 7 of address $037C_{16}$ = "1")
Transfer clock	• With the internal clock chosen (bit 3 of address 0378_{16} = "0") : fi / 16 (n + 1) (Note 1) : fi=f_1, f_8, f_{32} (Do not set external clock)
Transmission / reception control	• Disable the $\overline{CTS}$ and $\overline{RTS}$ function (bit 4 of address $037C_{16}$ = "1")
Other settings	• The sleep mode select function is not available for UART2 • Set transmission interrupt factor to "transmission completed" (bit 4 of address $037D_{16}$ = "1")
Transmission start condition	• To start transmission, the following requirements must be met: - Transmit enable bit (bit 0 of address $037D_{16}$) = "1" - Transmit buffer empty flag (bit 1 of address $037D_{16}$) = "0"
Reception start condition	• To start reception, the following requirements must be met: - Reception enable bit (bit 2 of address $037D_{16}$) = "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 $037D_{16}$ = "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 2) • 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 T_XD_2 pin by use of the parity error signal output function (bit 7 of address $037D_{16}$ = "1") when a parity error is detected - On the transmission side, a parity error is detected by the level of input to the R_XD_2 pin when a transmission interrupt occurs • 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 UART2 bit rate generator.

Note 2: If an overrun error occurs, the UART2 receive buffer will have the next data written in. Note also that the UART2 receive interrupt request bit does not change.


Tc
Transfer clock
Transmit enable bit(TE)
"1"
"0"
Data is set in UART2 transmit buffer register
Note 1
Transmit buffer empty flag(TI)
Transferred from UART2 transmit buffer register to UART2 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 2)
The level is detected by the interrupt routine.
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
Read to receive buffer
Receive interrupt request bit (IR)
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.
- Transmit interrupt cause select bit = "1".

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

Shown in ( ) are bit symbols.

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

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

Note 1 : The transmit is started with overflow timing of BRG after having written in a value at the transmit buffer in the above timing.
Note 2 : Equal in waveform because TxD2 and RxD2 are connected.

Figure 1.15.21. Typical transmit/receive timing in UART mode (compliant with the SIM interface)

(a) Function for outputting a parity error signal

With the error signal output enable bit (bit 7 of address 037D16) assigned “1”, you can output an “L” level from the TxD2 pin when a parity error is detected. In step with this function, the generation timing of a transmission completion interrupt changes to the detection timing of a parity error signal. Figure 1.15.22 shows the output timing of the parity error signal.

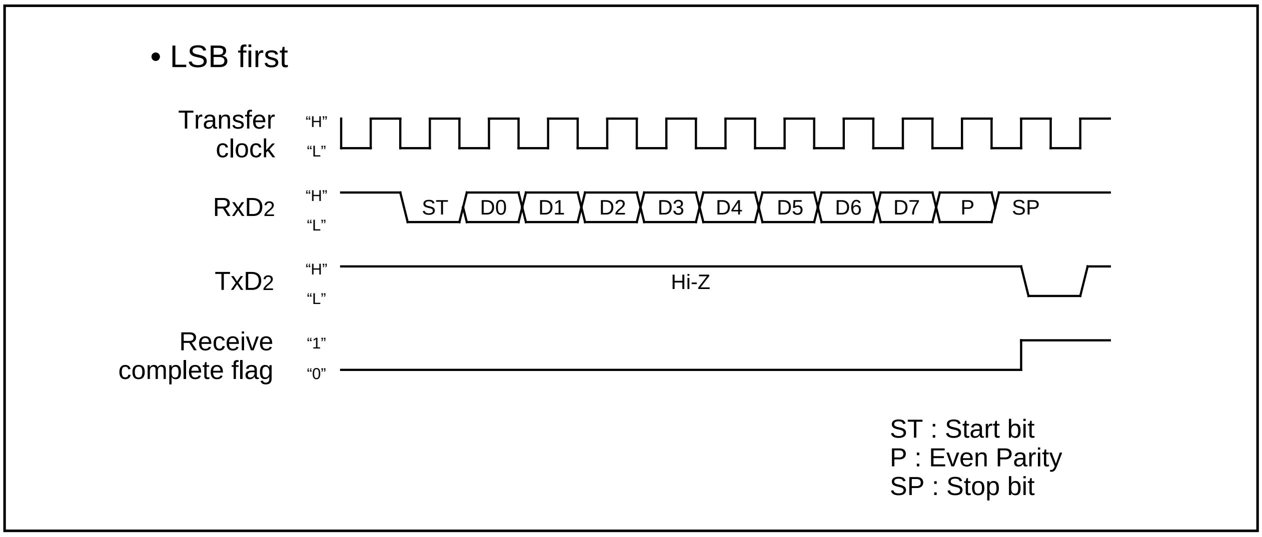


Figure 1.15.22. Output timing of the parity error signal

(b) Direct format/inverse format

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

Figure 1.15.23 shows the SIM interface format.

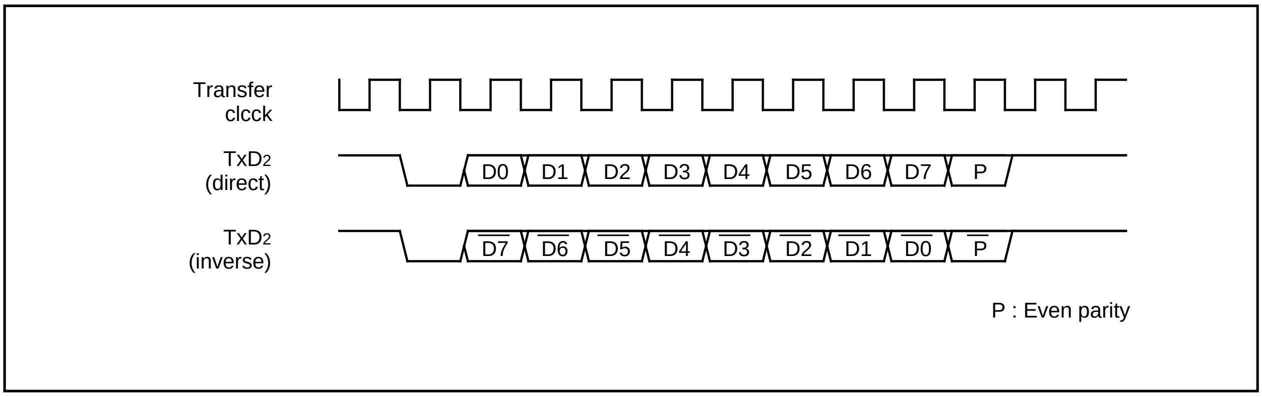


Figure 1.15.23. SIM interface format

Figure 1.15.24 shows the example of connecting the SIM interface. Connect TxD_2 and RxD_2 and apply pull-up.

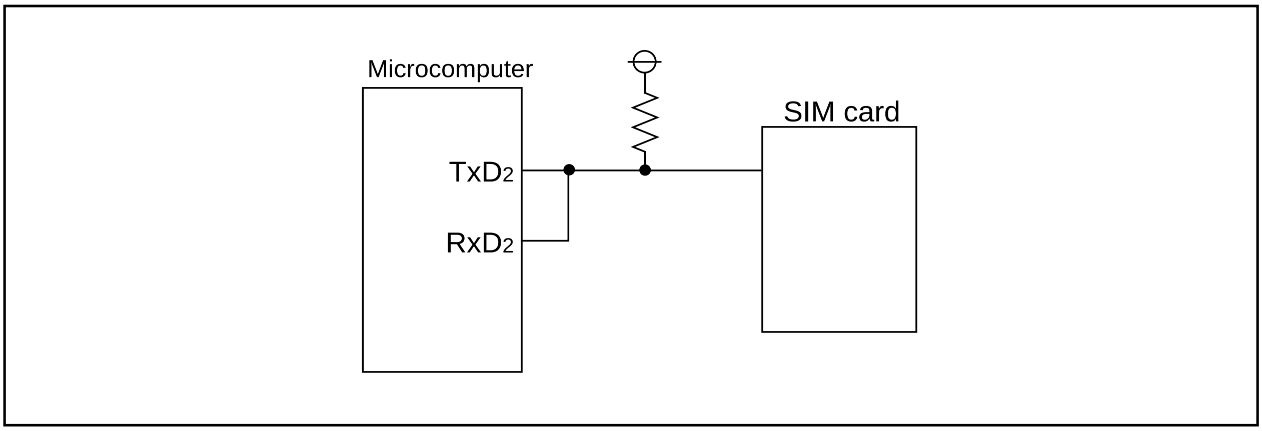
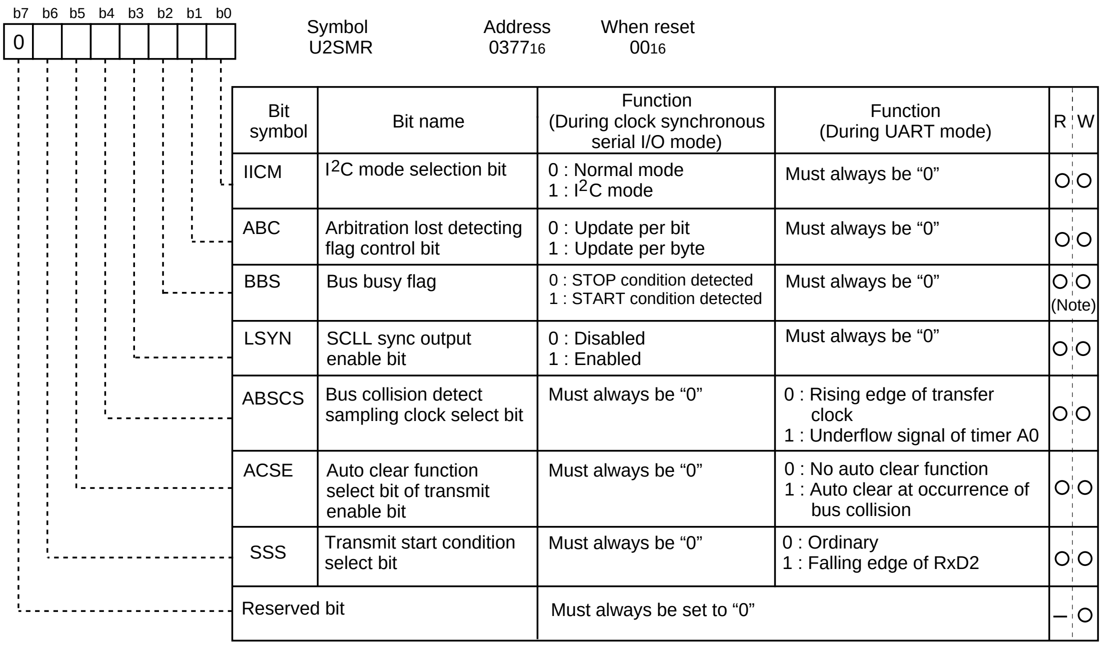


Figure 1.15.24. Connecting the SIM interface

UART2 Special Mode Register

The UART2 special mode register (address 0377$_{16}$) is used to control UART2 in various ways. Figure 1.15.25 shows the UART2 special mode register.

UART2 special mode register



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

Bit symbol	Bit name	Function (During clock synchronous serial I/O mode)	Function (During UART mode)	R	W
IICM	I^{2}C mode selection bit	0 : Normal mode 1 : I^{2}C 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	○	○
Reserved bit		Must always be set to “0”		–	○

Note: Nothing but "0" may be written.

Figure 1.15.25. UART2 special mode register

Table 1.15.9. 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	CLK$_2$	P7$_2$
8	DMA1 factor at the time when 1 1 0 1 is assigned to the DMA request factor selection bits	UART2 reception	Must not be set
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. Choose the MSB First function.

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

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

In the first place, the control bits related to the I2C bus (simplified I2C bus) interface are explained. Bit 0 of the UART special mode register (0377_{16}) is used as the I2C mode selection bit. Setting “1” in the I2C mode select bit (bit 0) goes the circuit to achieve the I2C bus (simplified I2C bus) interface effective. Table 1.15.9 shows the relation between the I2C mode select bit and respective control workings. Since this function uses clock-synchronous serial I/O mode, set this bit to “0” in UART mode.

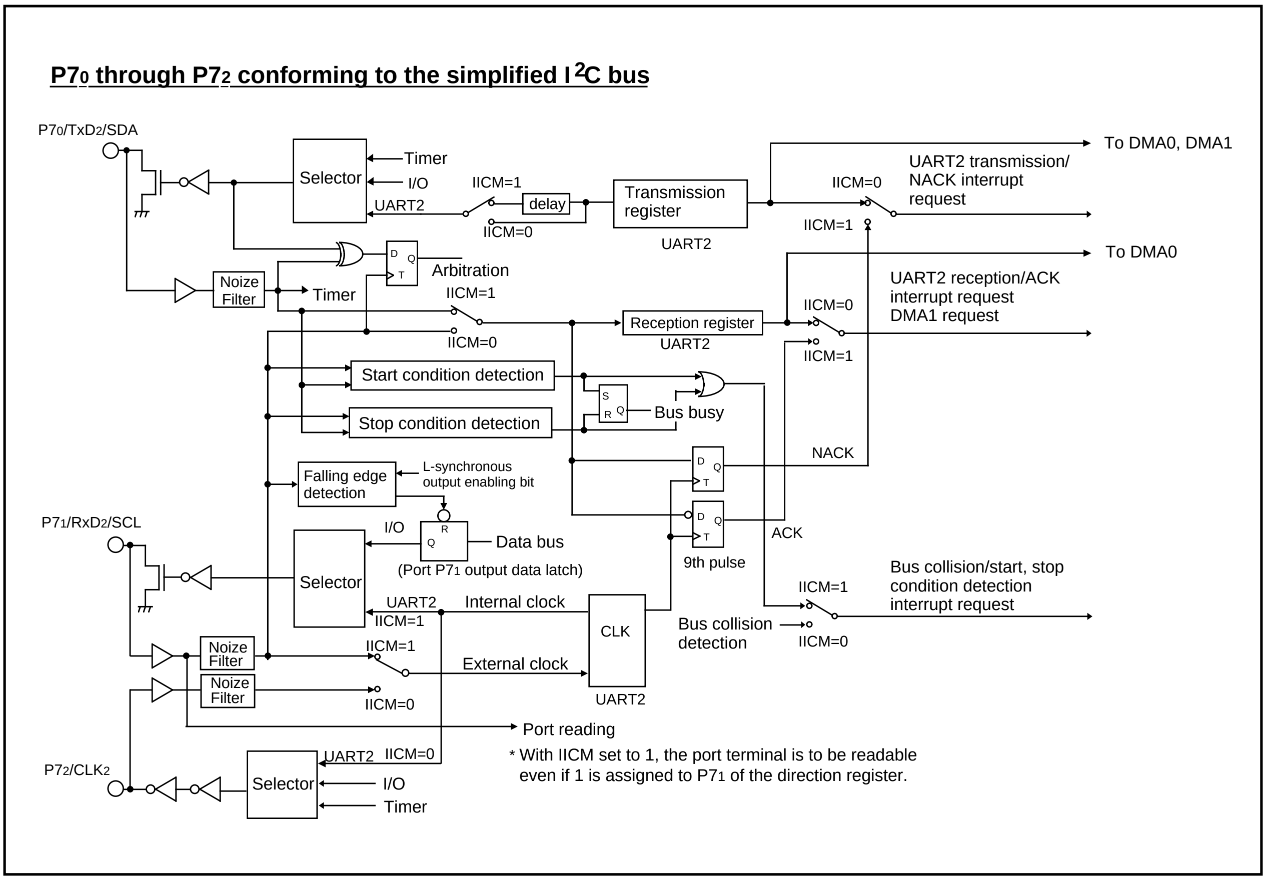


Figure 1.15.26. Functional block diagram for I2C mode

Figure 1.15.26 shows the functional block diagram for I2C mode. Setting “1” in the I2C 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 UART2 special mode register) is set to “1” by the start condition detection, and set to “0” by the stop condition detection.

The acknowledgment non-detection interrupt refers to the interrupt that occurs when the SDA terminal level is detected still staying “H” at the rising edge of the 9th transmission clock. The acknowledgment detection interrupt refers to the interrupt that occurs when SDA terminal’s level is detected already went to “L” at the 9th transmission clock. Bit 1 of the UART2 special mode register (0377_{16}) is used as the arbitration lost detecting flag control bit. Arbitration means the act of detecting the nonconformity between transmission data and SDA terminal data at the timing of the SCL rising edge. This detecting flag is located at bit 3 of the UART2 reception buffer register ($037F_{16}$), and “1” is set in this flag when nonconformity is detected. Use the arbitration lost detecting flag control bit to choose which way to use to update the flag, bit by bit or byte by byte. When setting this bit to “1” and updated the flag byte by byte if nonconformity is detected, the arbitration lost detecting flag is set to “1” at the falling edge of the 9th transmission clock.
If update the flag byte by byte, must judge and clear (“0”) the arbitration lost detecting flag after completing the first byte acknowledge detect and before starting the next one byte transmission.
Bit 3 of the UART2 special mode register is used as SCL- and L-synchronous output enable bit. Setting this bit to “1” goes the $P7_1$ data register to “0” in synchronization with the SCL terminal level going to “L”.

Some other functions added are explained here. Figure 1.15.27 shows their workings.

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

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

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

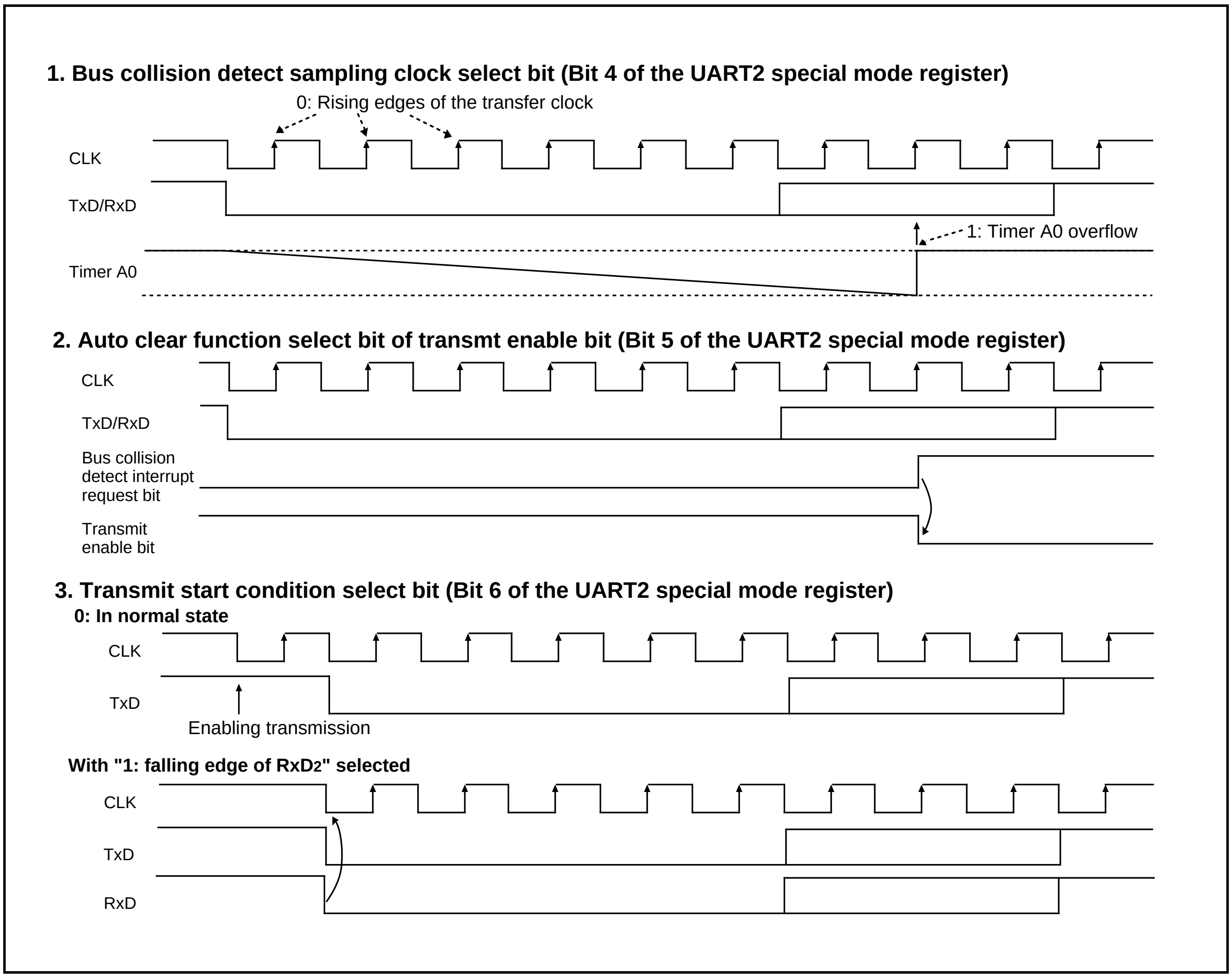
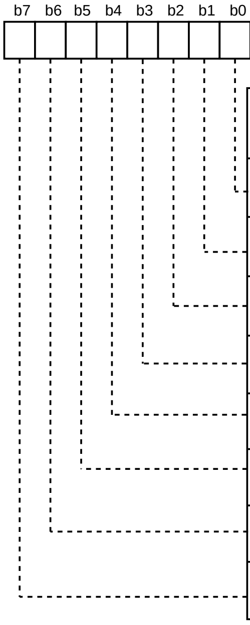

1. Bus collision detect sampling clock select bit (Bit 4 of the UART2 special mode register)
0: Rising edges of the transfer clock
CLK
TxD/RxD
1: Timer A0 overflow
Timer A0
2. Auto clear function select bit of transmt enable bit (Bit 5 of the UART2 special mode register)
CLK
TxD/RxD
Bus collision detect 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 1.15.27. Some other functions added

UART2 Special Mode Register 2

UART2 special mode register 2 (address 0376_{16}) is used to further control UART2 in I^2C mode. Figure 1.15.28 shows the UART2 special mode register 2.

UART2 special mode register 2



Symbol	Address	When reset
U2SMR2	0376_{16}	00_{16}

Bit symbol	Bit name	Function	R	W
IICM2	I^2C mode selection bit 2	Refer to Table 1.15.10	○	○
CSC	Clock-synchronous bit	0 : Disabled 1 : Enabled	○	○
SWC	SCL wait output bit	0 : Disabled 1 : Enabled	○	○
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^2C mode (refer to Table 1.15.11)	○	○

Figure 1.15.28. UART2 special mode register 2

Bit 0 of the UART2 special mode register 2 (address 0376_{16}) is used as the I^2C mode selection bit 2. Table 1.15.10 shows the types of control to be changed by I^2C mode selection bit 2 when the I^2C mode selection bit is set to "1". Table 1.15.11 shows the timing characteristics of detecting the start condition and the stop condition. Set the start/stop condition control bit (bit 7 of UART2 special mode register 2) to "1" in I^2C mode.

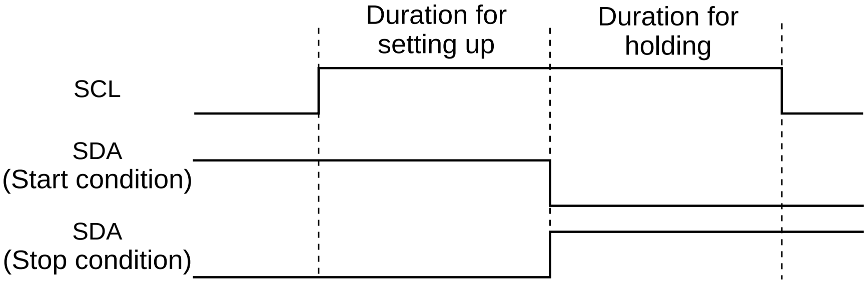
Table 1.15.10. Functions changed by I^2C mode selection bit 2

	Function	IICM2 = 0	IICM2 = 1
1	Factor of interrupt number 15	No acknowledgment detection (NACK)	UART2 transmission (the rising edge of the final bit of the clock)
2	Factor of interrupt number 16	Acknowledgment detection (ACK)	UART2 reception (the falling edge of the final bit of the clock)
3	DMA1 factor at the time when 1 1 0 1 is assigned to the DMA request factor selection bits	Must not be set	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 1.15.11. Timing characteristics of detecting the start condition and the stop condition(Note1)

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

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



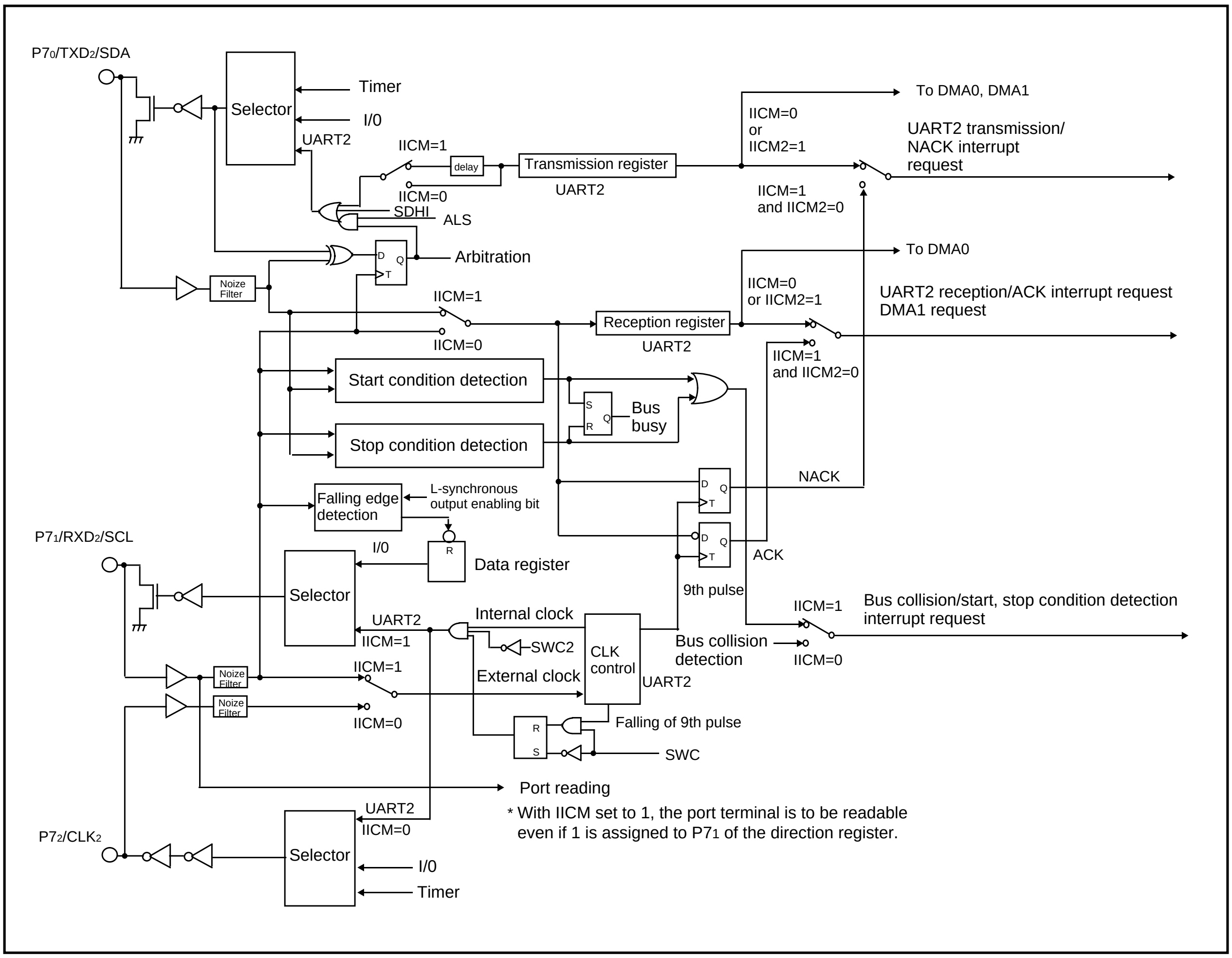


Figure 1.15.29. Functional block diagram for I^{2}C mode

Functions available in I^{2}C mode are shown in Figure 1.15.29 — a functional block diagram.

Bit 3 of the UART2 special mode register 2 (address 0376$_{16}$) 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 at the instant when the arbitration lost detectng flag is set to "1".

Bit 1 of the UART2 special mode register 2 (address 0376$_{16}$) 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 baud rate generator reloads the set value, and start counting within the "L" interval. When the internal SCL changes from "L" to "H" with the SCL pin set to "L", stops counting the baud rate generator, and starts counting it again when the SCL pin turns to "H". Due to this function, the UART2 transmission-reception clock becomes the logical product of the signal flowing through the internal SCL and that flowing through the SCL pin. This function operates over the period from the moment earlier by a half cycle than falling edge of the UART2 first clock to the rising edge of the ninth bit. To use this function, choose the internal clock for the transfer clock.

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

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

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

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

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

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

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

LCD Drive Control Circuit

The M30220 group has the built-in Liquid Crystal Display (LCD) drive control circuit consisting of the following.

- LCD display RAM
- Segment output enable register
- LCD mode register
- Charge-pump
- Selector
- Timing controller
- Common driver
- Segment driver
- Bias control circuit

A maximum of 48 segment output pins and 4 common output pins can be used.

Up to 192 pixels can be controlled for LCD display. When the LCD enable bit is set to “1” after data is set in the LCD mode register, the segment output enable register and the LCD display RAM, the LCD drive control circuit starts reading the display data automatically, performs the bias control and the duty ratio control, and displays the data on the LCD panel. When using the output port function, write data into the LCD display RAM while the time division select bit are “00” and the LCD output enable bit is “0”, and if the LCDRAM output bit is set to “1”, the SEG_0 - SEG_{15} pin and the pin which are selected as segment output by the segment output enable register will respectively output the contents of the bit corresponds to the COM_0 of LCD display RAM.

Table 1.16.1 shows maximum number of display pixels at each duty ratio. Figure 1.16.1 shows the block diagram of LCD controller / driver.

Table 1.16.1. Maximum number of display pixels at each duty ratio

Duty ratio	Maximum number of display pixel
2	96 dots or 8 segment LCD 12 digits
3	144 dots or 8 segment LCD 18 digits
4	192 dots or 8 segment LCD 24 digits

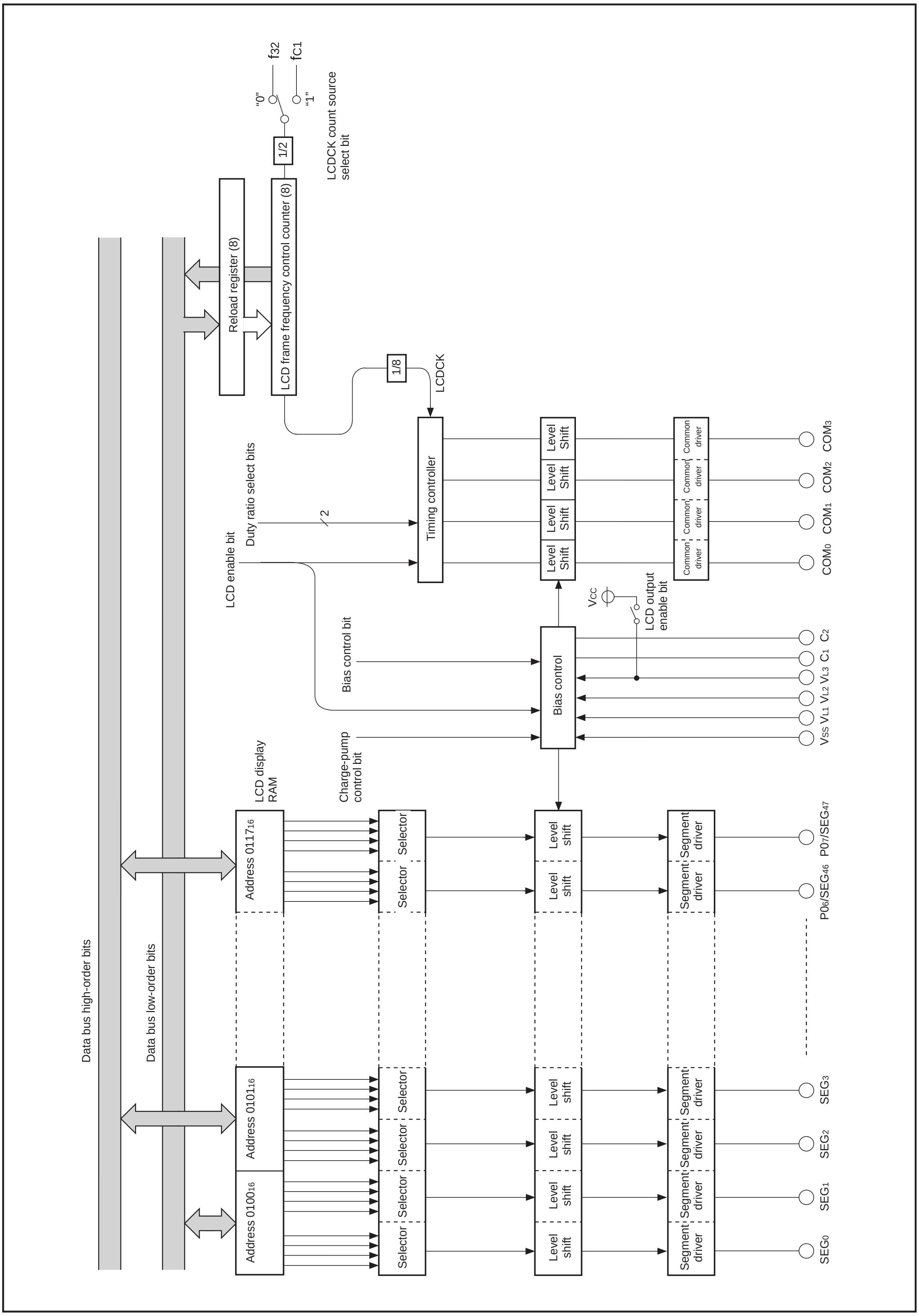
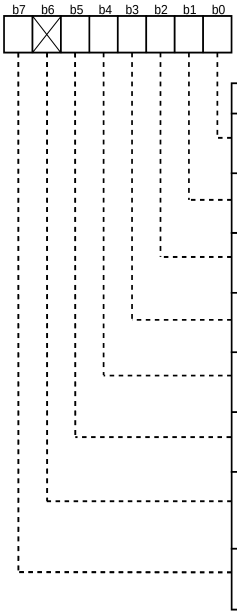


Figure 1.16.1. Block diagram of LCD controller/driver

LCD mode register

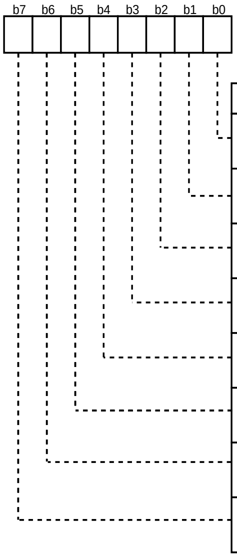


Symbol	Address	When reset
LCDM	0120$_{16}$	0X000000$_2$

Bit symbol	Bit name	Function	R	W
LCDT0	Duty ratio select bit	b1 b0 0 0 : Output port 0 1 : 2 duty (use COM$_0$, COM$_1$) 1 0 : 3 duty (use COM$_0$–COM$_2$) 1 1 : 4 duty (use COM$_0$–COM$_3$)	○	○
LCDT1			○	○
BIAS	Bias control bit	0 : 1/3 bias 1 : 1/2 bias	○	○
LCDEN	LCD enable bit	0 : LCD OFF 1 : LCD ON	○	○
PUMP	Charge-pump control bit	0 : Charge-pump disable 1 : Charge-pump enable(Note2)	○	○
LRAMOUT	LCDRAM output bit	0 : LCD waveform output 1 : LCDRAM data output	○	○
Nothing is assigned. In an attempt to write to this bit, write “0”. The value, if read, turns out to be indeterminate.			—	—
LSRC	LCDCK count source select bit (Note 1)	0 : f$_{32}$ 1 : fc$_1$	○	○

Note 1: LCDCK is a clock for a LCD timing controller.
Note 2: When the Charge-pump is enabled, set "0" to the bias control bit without fail.

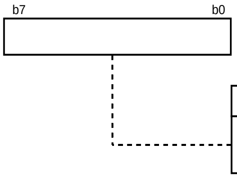
Segment output enable register



Symbol	Address	When reset
SEG	0122$_{16}$	00$_{16}$

Bit symbol	Bit name	Function	R	W
SEGO0	Segment output enable bit 0	0 : I/O ports P10$_0$ to P10$_7$ 1 : Segment output SEG$_{16}$ to SEG$_{23}$	○	○
SEGO1	Segment output enable bit 1	0 : I/O ports P11$_0$ to P11$_4$ 1 : Segment output SEG$_{24}$ to SEG$_{28}$	○	○
SEGO2	Segment output enable bit 2	0 : I/O ports P11$_5$, P11$_6$ 1 : Segment output SEG$_{29}$, SEG$_{30}$	○	○
SEGO3	Segment output enable bit 3	0 : I/O ports P11$_7$ 1 : Segment output SEG$_{31}$	○	○
SEGO4	Segment output enable bit 4	0 : I/O ports P12$_0$ to P12$_5$ 1 : Segment output SEG$_{32}$ to SEG$_{37}$	○	○
SEGO5	Segment output enable bit 5	0 : I/O ports P12$_6$, P12$_7$ 1 : Segment output SEG$_{38}$, SEG$_{39}$	○	○
SEGO6	Segment output enable bit 6	0 : I/O ports P0$_0$ to P0$_7$ 1 : Segment output SEG$_{40}$ to SEG$_{47}$	○	○
SEGO7	LCD output enable bit	0 : disable 1 : enable	○	○

LCD frame frequency counter (Note)



Symbol	Address	When reset
LCDTIM	0124$_{16}$	XX$_{16}$

Function	Values that can be set	R	W
8 bits timer	00$_{16}$ to FF$_{16}$	○	○

Note: Set this register when LCD output enable bit is “0” (disable).

Figure 1.16.2. LCD-related registers

Charge-pump

The charge-pump performs threefold boosting. This circuit inputs a reference voltage for boosting from LCD power input pin V_{L1}.

To activate the charge-pump, by the segment output enable register and the LCD mode register, choose the segment/port, select the time division and the bias control, and set up the LCD frame frequency counter, and select the count source for LCDCK, then set the LCD output enable bit (bit 7 at the address 0122_{16}) to “enable”, apply a voltage equal to or greater than 1.3 V but not exceeding 2.1 V to the V_{L1} pin, after that, set the charge-pump control bit (bit 4 at address 0120_{16}) to “step up enabled”. However, set the bias control to “1/3 bias” without fail.

When using the charge-pump, a voltage that is twice as large as V_{L1} occurs at V_{L2} pin, and a voltage that is three times as large as V_{L1} occurs at the V_{L3} pin.

The charge-pump control bit (bit 4 of the address 0120_{16}) controls the charge-pump.

When not using the charge-pump, enable the LCD output enable bit and apply an appropriate voltage to the LCD power supply input pins (V_{L1} to V_{L3}). When the LCD output enable bit is disabled, the V_{L3} pin is connected to V_{CC} internally.

Bias Control and Applied Voltage to LCD Power Input Pins

To the LCD power input pins (V_{L1} to V_{L3}), apply the voltage shown in Table 1.16.2 according to the bias value. Select a bias value by the bias control bit (bit 2 of the address 0120_{16}).

Table 1.16.2. Bias control and applied voltage to V_{L1} to V_{L3}

Bias value	Voltage value
1/3 bias	$V_{L3} = V_{LCD}$ $V_{L2} = 2/3\ V_{LCD}$ $V_{L1} = 1/3\ V_{LCD}$
1/2 bias	$V_{L3} = V_{LCD}$ $V_{L2} = V_{L1} = 1/2\ V_{LCD}$

Note : V_{LCD} is the maximum value of supplied voltage for the LCD panel.

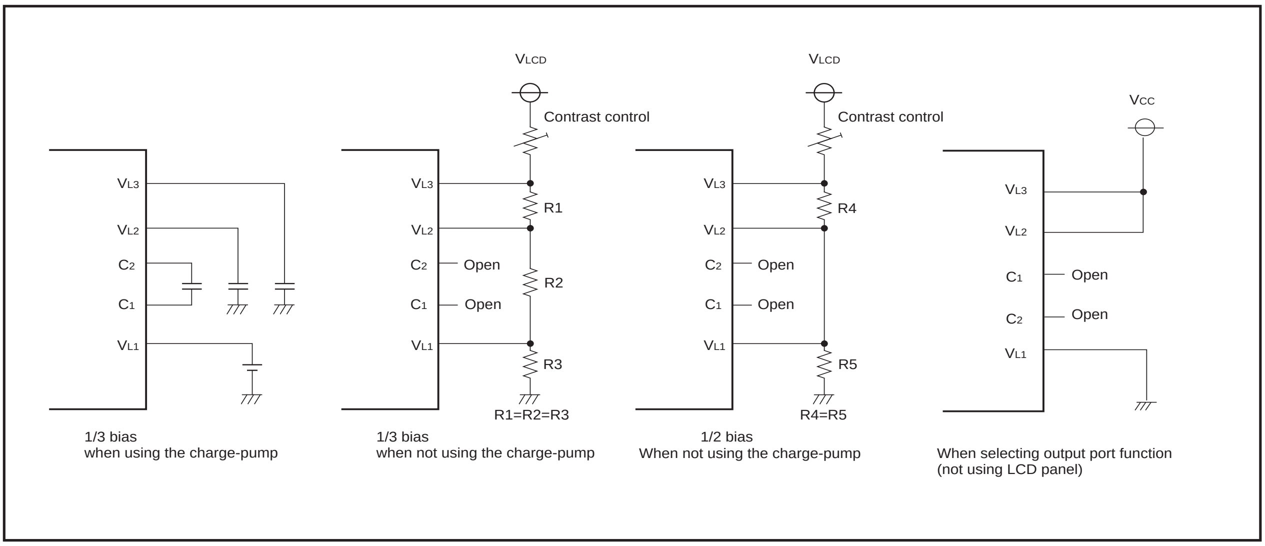


Figure 1.16.3. Example of circuit at each bias

Common Pin and Duty Ratio Control

The common pins (COM0 to COM3) to be used are determined by duty ratio.

Select duty ratio by the duty ratio select bits (bits 0 and 1 of address 0120_{16}).

Table 1.16.3. Duty ratio control and common pins used

Duty	Duty ratio select bit		Common pins used
ratio	Bit 1	Bit 0	
2	0	1	COM0, COM1 (Note 1)
3	1	0	COM0 to COM2 (Note 2)
4	1	1	COM0 to COM3

Note 1 : COM2 and COM3 are open.

Note 2 : COM3 is open.

LCD Display RAM

Address 0100_{16} to 0117_{16} is the designated RAM for the LCD display. When “1” are written to these addresses, the corresponding segments of the LCD display panel are turned on.

Figure 1.16.4 shows the LCD display RAM map.

Bit / Address	7	6	5	4	3	2	1	0	R	W
	COM3	COM2	COM1	COM0	COM3	COM2	COM1	COM0		
0100_{16}	SEG1				SEG0				○	○
0101_{16}	SEG3				SEG2				○	○
0102_{16}	SEG5				SEG4				○	○
0103_{16}	SEG7				SEG6				○	○
0104_{16}	SEG9				SEG8				○	○
0105_{16}	SEG11				SEG10				○	○
0106_{16}	SEG13				SEG12				○	○
0107_{16}	SEG15				SEG14				○	○
0108_{16}	SEG17				SEG16				○	○
0109_{16}	SEG19				SEG18				○	○
$010A_{16}$	SEG21				SEG20				○	○
$010B_{16}$	SEG23				SEG22				○	○
$010C_{16}$	SEG25				SEG24				○	○
$010D_{16}$	SEG27				SEG26				○	○
$010E_{16}$	SEG29				SEG28				○	○
$010F_{16}$	SEG31				SEG30				○	○
0110_{16}	SEG33				SEG32				○	○
0111_{16}	SEG35				SEG34				○	○
0112_{16}	SEG37				SEG36				○	○
0113_{16}	SEG39				SEG38				○	○
0114_{16}	SEG41				SEG40				○	○
0115_{16}	SEG43				SEG42				○	○
0116_{16}	SEG45				SEG44				○	○
0117_{16}	SEG47				SEG46				○	○

Figure 1.16.4. LCD display RAM map

LCD Drive Timing

The LCDCK timing frequency (LCD drive timing) is generated internally and the frame frequency can be determined with the following equation. The LCDCK count source frequency is f_{C1} (same frequency as X_{CIN}) or f_{32} (divide-by-32 of X_{IN} frequency).

$$f(\text{LCDCK}) = \frac{\text{(frequency of count source for LCDCK)}}{16 \text{ X (LCD frame frequency count value + 1)}}$$

$$\text{Frame frequency} = \frac{f(\text{LCDCK})}{\text{duty ratio}}$$

Figure 1.16.5 shows the LCD drive waveform (1/2 bias), Figure 1.16.6 shows the LCD drive waveform (1/3 bias).

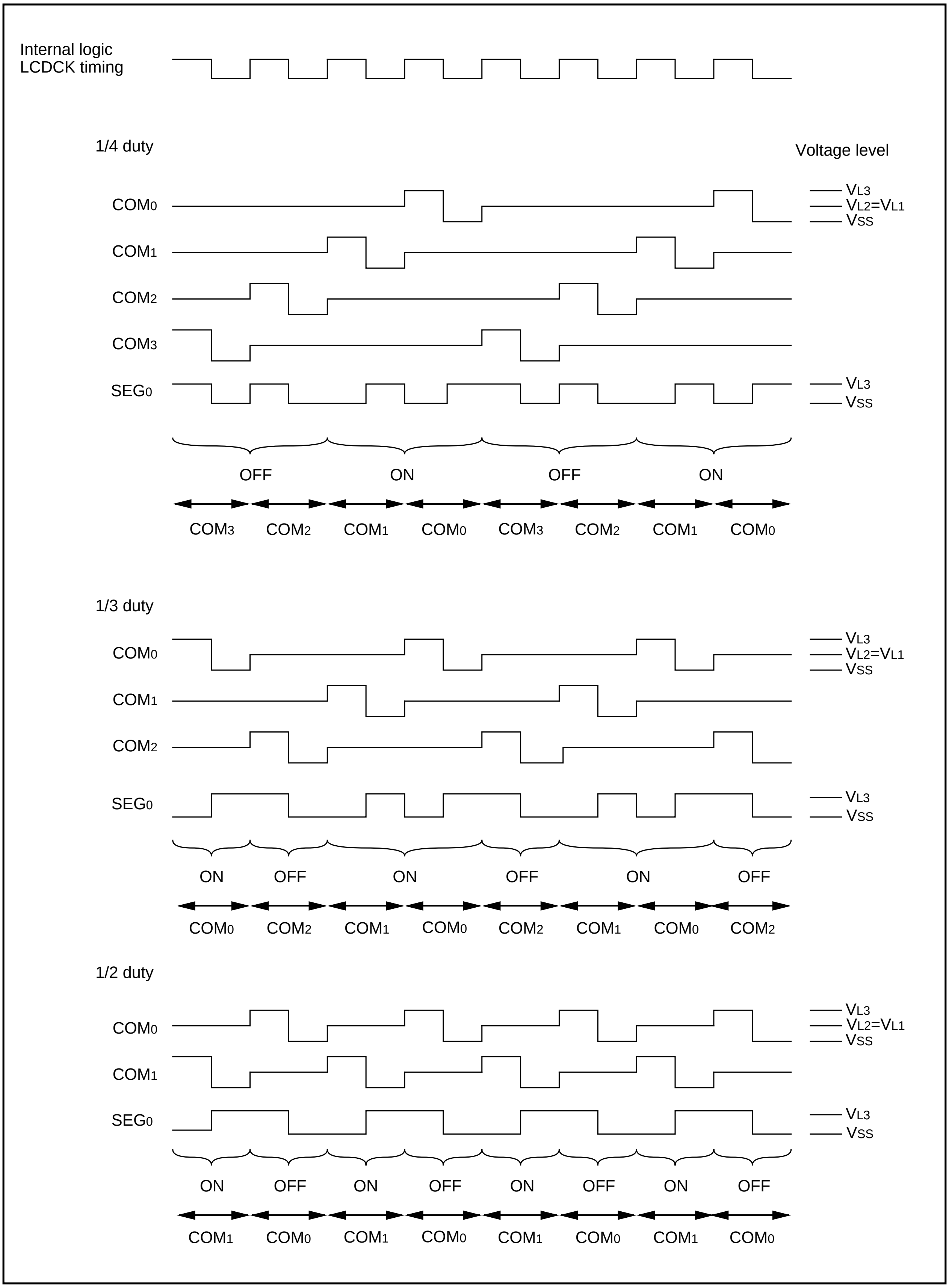


Figure 1.16.5. LCD drive waveform (1/2 bias)

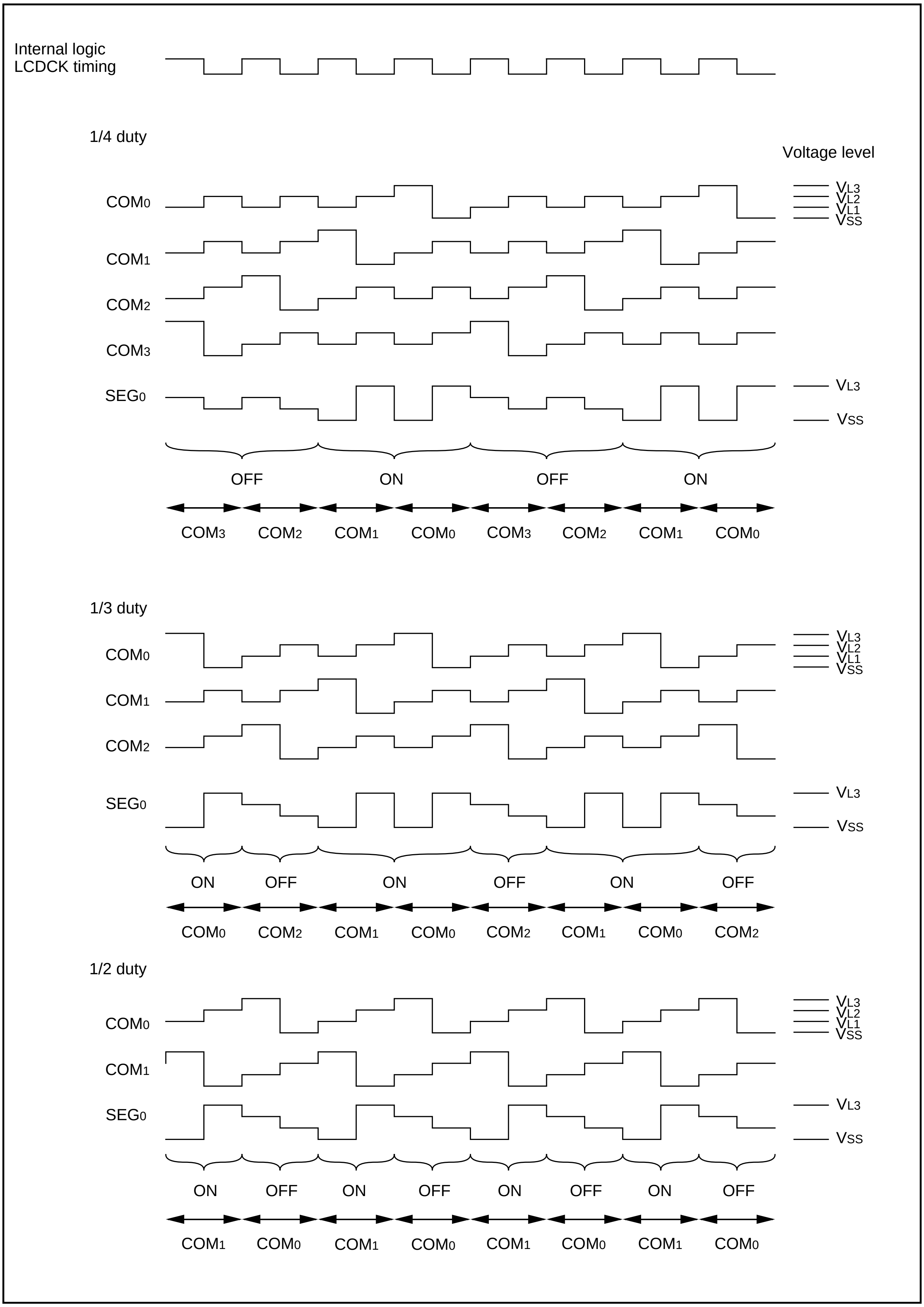


Figure 1.16.6. LCD drive waveform (1/3 bias)

A-D Converter

The A-D converter consists of one 10-bit successive approximation A-D converter circuit with a capacitive coupling amplifier. Pins $P9_0$ to $P9_7$ also function as the analog signal input pins. The direction registers of these pins for A-D conversion must therefore be set to input. The Vref connect bit (bit 5 at address $03D7_{16}$) can be used to isolate the resistance ladder of the A-D converter from the reference voltage input pin (V_{REF}) when the A-D converter is not used. Doing so stops any current flowing into the resistance ladder from V_{REF}, reducing the power dissipation. When using the A-D converter, start A-D conversion only after setting bit 5 of $03D7_{16}$ to connect V_{REF}.

The result of A-D conversion is stored in the A-D registers of the selected pins. When set to 10-bit precision, the low 8 bits are stored in the even addresses and the high 2 bits in the odd addresses. When set to 8-bit precision, the low 8 bits are stored in the even addresses.

Table 1.17.1 shows the performance of the A-D converter. Figure 1.17.1 shows the block diagram of the A-D converter, and Figures 1.17.2 and 1.17.3 show the A-D converter-related registers.

Table 1.17.1. Performance of A-D converter

Item	Performance
Method of A-D conversion	Successive approximation (capacitive coupling amplifier)
Analog input voltage (Note 1)	0V to AV_{CC} (V_{CC})
Operating clock ϕ_{AD} (Note 2)	V_{CC} = 4.0 to 5.5V f_{AD}/divide-by-2 of f_{AD}/divide-by-4 of f_{AD}, f_{AD}=f(X_{IN}) V_{CC} = 2.7 to 4.0V divide-by-2 of f_{AD}/divide-by-4 of f_{AD}, f_{AD}=f(X_{IN})
Resolution	8-bit or 10-bit (selectable)
Absolute precision	V_{CC} = 5V • Without sample and hold function ±3LSB • With sample and hold function (8-bit resolution) ±2LSB • With sample and hold function (10-bit resolution) ±3LSB V_{CC} = 3V • Without sample and hold function (8-bit resolution) ±2LSB
Operating modes	One-shot mode, repeat mode, single sweep mode, repeat sweep mode 0, and repeat sweep mode 1
Analog input pins	8pins (AN_0 to AN_7)
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 $\overline{AD_{TRG}}$/$P13_0$ 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 250kH_Z min.
With the sample and hold function, set the ϕ_{AD} frequency to 1MH_Z min.

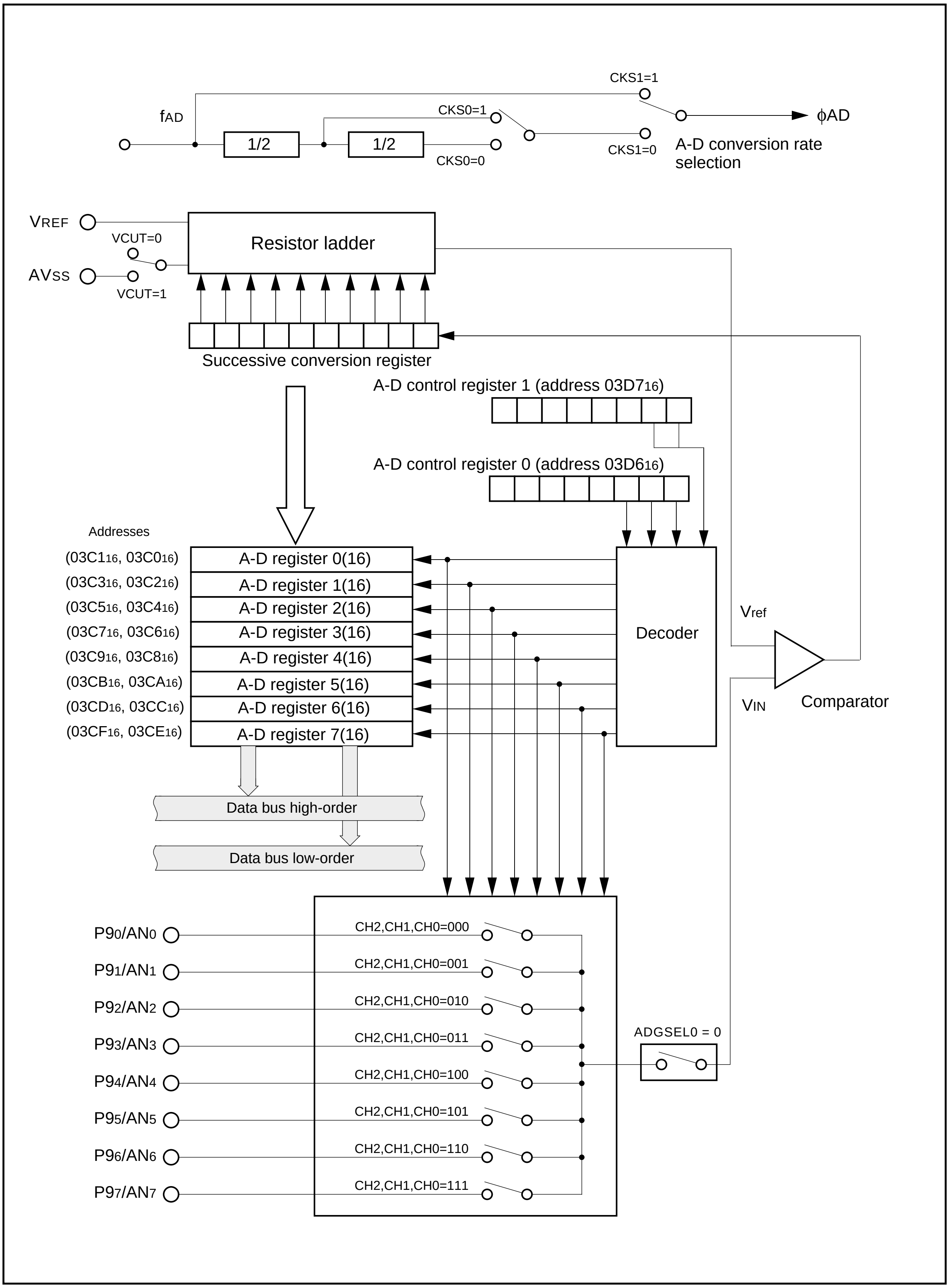


Figure 1.17.1. Block diagram of A-D converter

A-D control register 0 (Note 1)

b7	b6	b5	b4	b3	b2	b1	b0

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

Bit symbol	Bit name	Function	R	W
CH0	Analog input pin select bit	b2 b1 b0 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 0 : One-shot mode 0 1 : Repeat mode 1 0 : Single sweep mode 1 1 : Repeat sweep mode 0 Repeat sweep mode 1 (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_{AD}/4 is selected 1 : f_{AD}/2 is selected	○	○

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

Note 2: When changing A-D operation mode, set analog input pin again.

A-D control register 1 (Note)

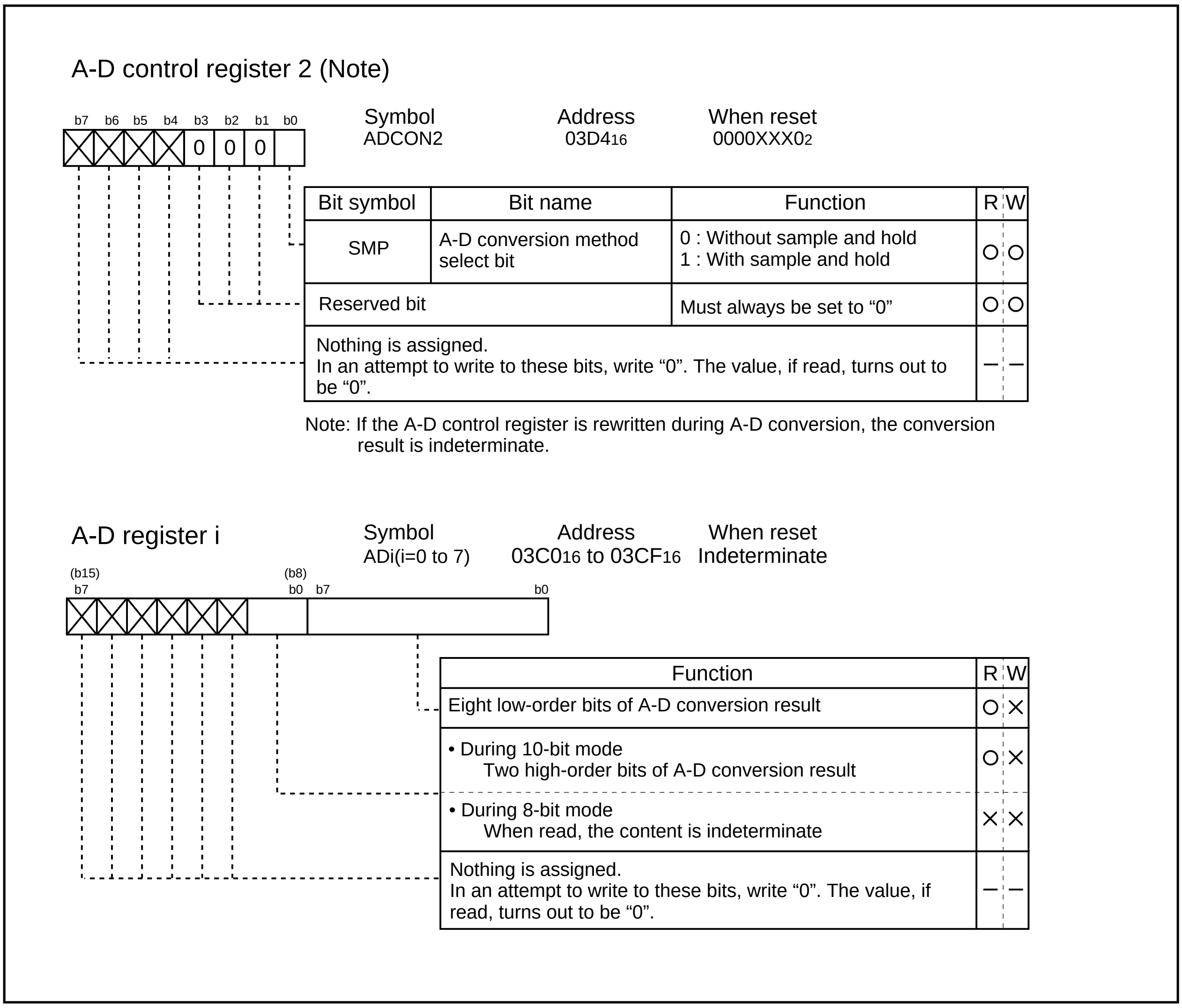
b7	b6	b5	b4	b3	b2	b1	b0
0	0						

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

Bit symbol	Bit name	Function	R	W
SCAN0	A-D sweep pin select bit	When single sweep and repeat sweep mode 0 are selected 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_{AD}/2 or f_{AD}/4 is selected 1 : f_{AD} is selected	○	○
VCUT	Vref connect bit	0 : Vref not connected 1 : Vref connected	○	○
Reserved bit		Must always be set to “0”	—	—

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

Figure 1.17.2. A-D converter-related registers (1)



A-D control register 2 (Note)

Symbol	Address	When reset
ADCON2	03D4$_{16}$	0000XXX0$_2$

Bit symbol	Bit name	Function	R	W
SMP	A-D conversion method select bit	0 : Without sample and hold 1 : With sample and hold	○	○
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 “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. In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”.	–	–

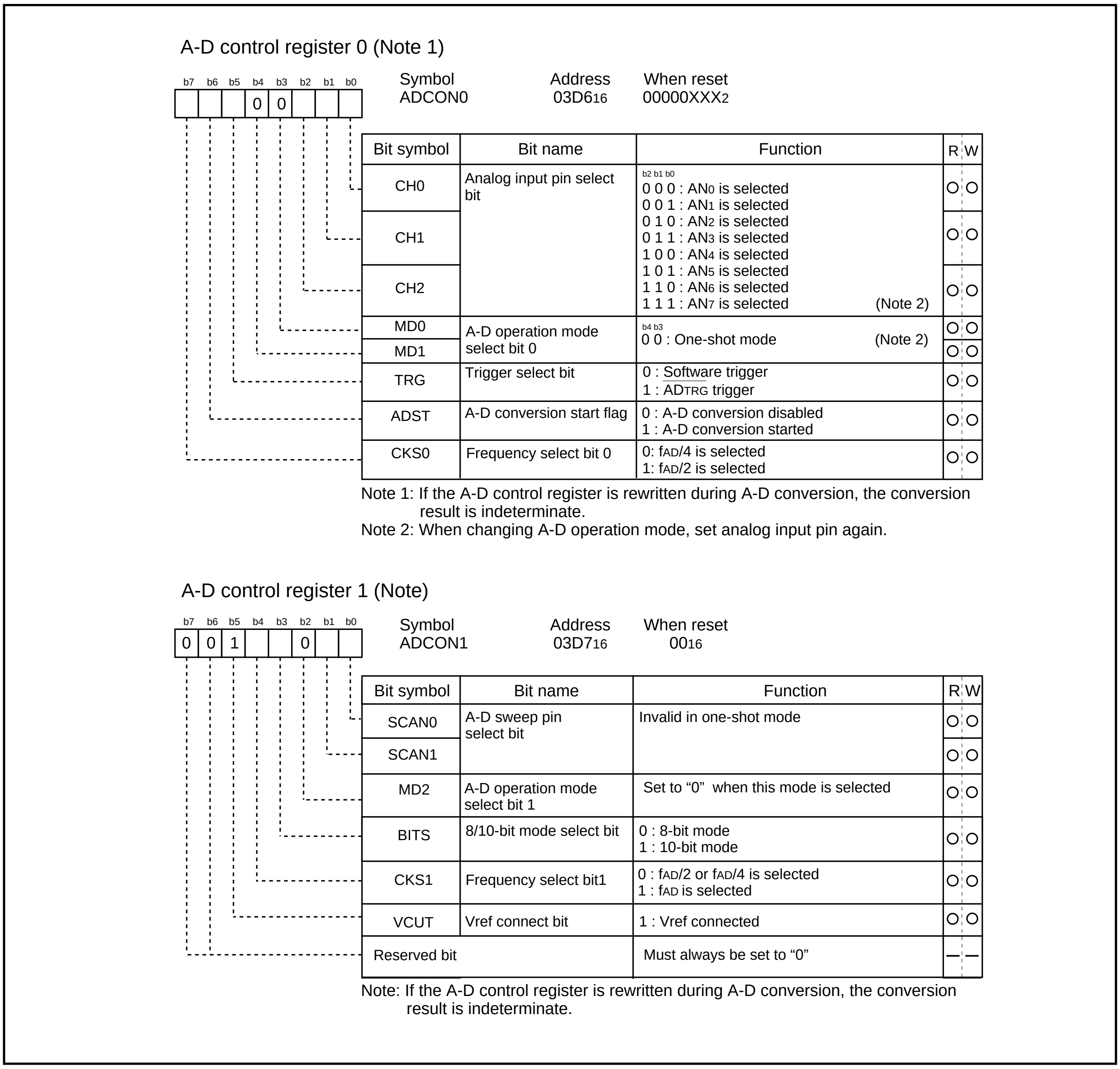
Figure 1.17.3. A-D converter-related registers (2)

(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 1.17.2 shows the specifications of one-shot mode. Figure 1.17.4 shows the A-D control register in one-shot mode.

Table 1.17.2. 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 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 0 : One-shot mode (Note 2)	○	○
MD1			○	○
TRG	Trigger select bit	0 : Software trigger 1 : $\overline{ADTRG}$ trigger	○	○
ADST	A-D conversion start flag	0 : A-D conversion disabled 1 : A-D conversion started	○	○
CKS0	Frequency select bit 0	0: $f_{AD}/4$ is selected 1: $f_{AD}/2$ is selected	○	○

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

A-D control register 1 (Note)

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

Bit symbol	Bit name	Function	R	W
SCAN0	A-D sweep pin select bit	Invalid in one-shot mode	○	○
SCAN1			○	○
MD2	A-D operation mode select bit 1	Set to “0” when this mode is selected	○	○
BITS	8/10-bit mode select bit	0 : 8-bit mode 1 : 10-bit mode	○	○
CKS1	Frequency select bit1	0 : $f_{AD}/2$ or $f_{AD}/4$ is selected 1 : f_{AD} is selected	○	○
VCUT	Vref connect bit	1 : Vref connected	○	○
Reserved bit		Must always be set to “0”	—	—

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

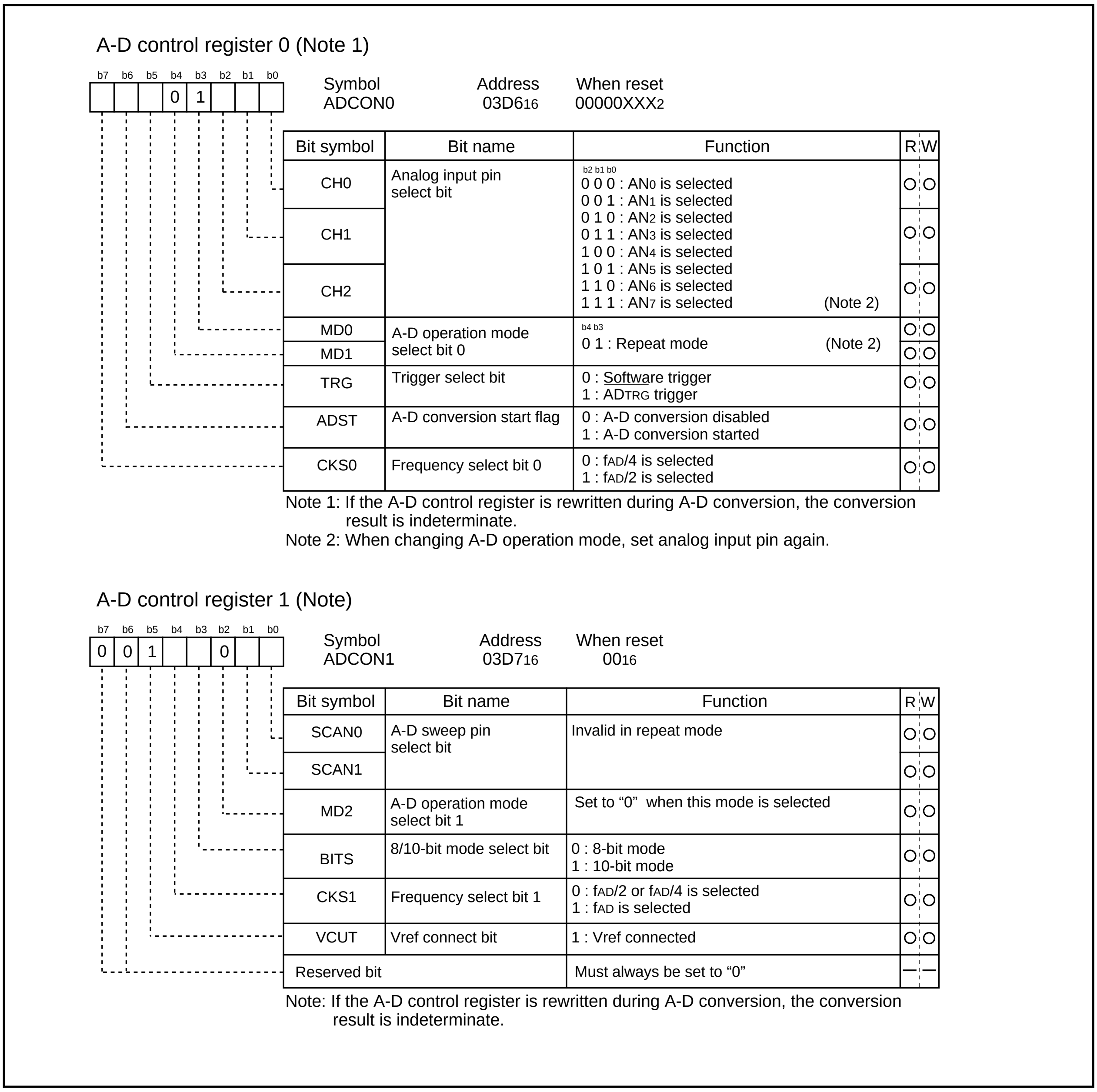
Figure 1.17.4. A-D conversion register in one-shot mode

(2) Repeat mode

In repeat mode, the pin selected using the analog input pin select bit is used for repeated A-D conversion. Table 1.17.3 shows the specifications of repeat mode. Figure 1.17.5 shows the A-D control register in repeat mode.

Table 1.17.3. Repeat mode specifications

Item	Specification
Function	The pin selected by the analog input pin select bit is used for repeated A-D conversion
Star condition	Writing “1” to A-D conversion start flag
Stop condition	Writing “0” to A-D conversion start flag
Interrupt request generation timing	None generated
Input pin	One of AN_0 to AN_7, as selected
Reading of result of A-D converter	Read A-D register corresponding to selected pin (at any time)



A-D control register 0 (Note 1)

b7	b6	b5	b4	b3	b2	b1	b0
			0	1			

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 (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 : ADTRG trigger	○	○
ADST	A-D conversion start flag	0 : A-D conversion disabled 1 : A-D conversion started	○	○
CKS0	Frequency select bit 0	0 : $f_{AD}/4$ is selected 1 : $f_{AD}/2$ is selected	○	○

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

A-D control register 1 (Note)

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

Symbol	Address	When reset
ADCON1	03D7$_{16}$	00$_{16}$

Bit symbol	Bit name	Function	R	W
SCAN0	A-D sweep pin select bit	Invalid in repeat mode	○	○
SCAN1			○	○
MD2	A-D operation mode select bit 1	Set to “0” when this mode is selected	○	○
BITS	8/10-bit mode select bit	0 : 8-bit mode 1 : 10-bit mode	○	○
CKS1	Frequency select bit 1	0 : $f_{AD}/2$ or $f_{AD}/4$ is selected 1 : f_{AD} is selected	○	○
VCUT	Vref connect bit	1 : Vref connected	○	○
Reserved bit		Must always be set to “0”	—	—

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

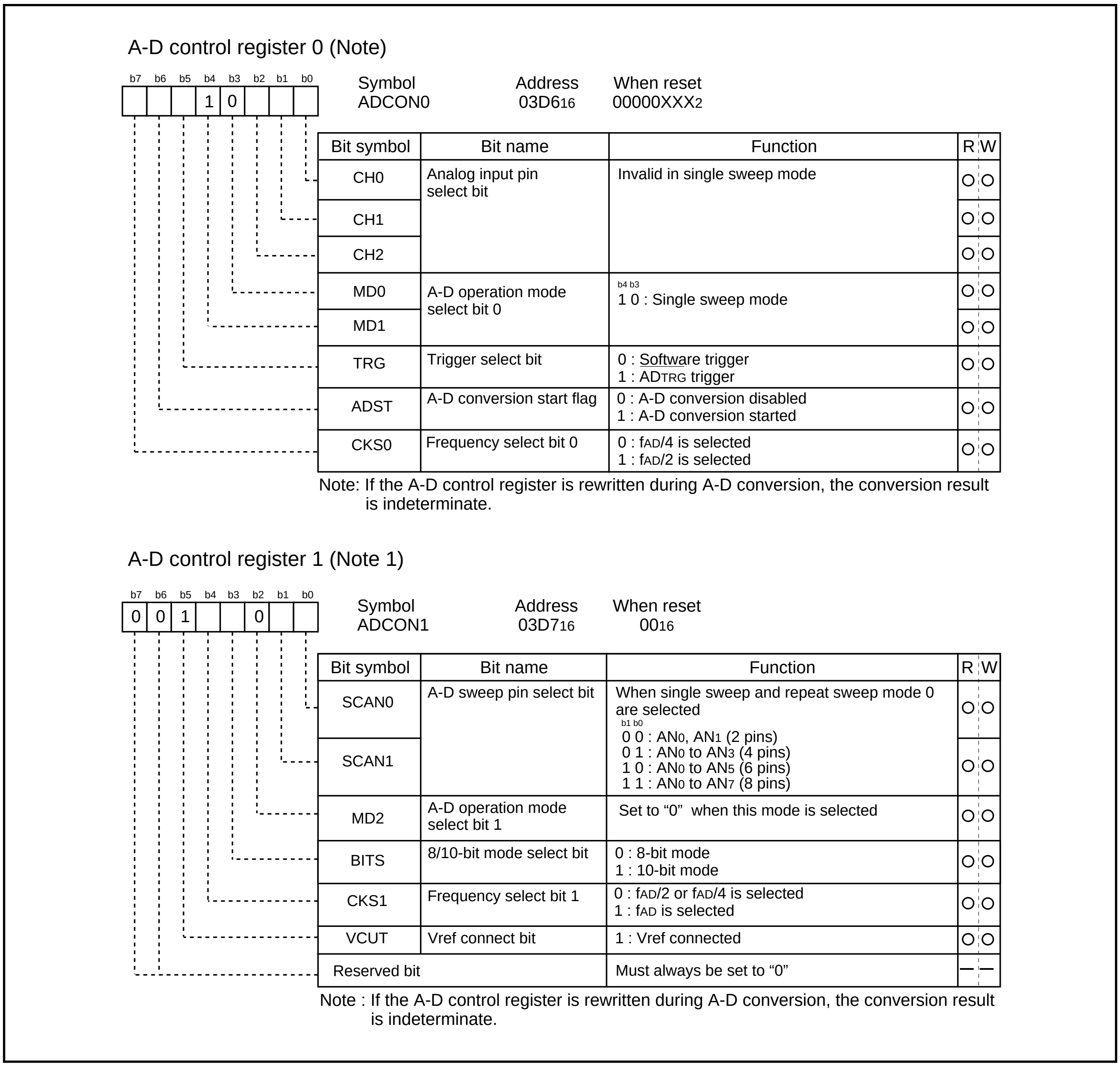
Figure 1.17.5. A-D conversion register in repeat mode

(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 1.17.4 shows the specifications of single sweep mode. Figure 1.17.6 shows the A-D control register in single sweep mode.

Table 1.17.4. 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 ADCON0 Address 03D6$_{16}$ When reset 00000XXX$_2$

Bit symbol	Bit name	Function	R	W
CH0	Analog input pin select bit	Invalid in single sweep mode	○	○
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 : 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$_{AD}$/4 is selected 1 : f$_{AD}$/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 ADCON1 Address 03D7$_{16}$ When reset 00$_{16}$

Bit symbol	Bit name	Function	R	W
SCAN0	A-D sweep pin select bit	When single sweep and repeat sweep mode 0 are selected 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	Set to “0” when this mode is selected	○	○
BITS	8/10-bit mode select bit	0 : 8-bit mode 1 : 10-bit mode	○	○
CKS1	Frequency select bit 1	0 : f$_{AD}$/2 or f$_{AD}$/4 is selected 1 : f$_{AD}$ is selected	○	○
VCUT	Vref connect bit	1 : Vref connected	○	○
Reserved bit		Must always be set to “0”	—	—

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

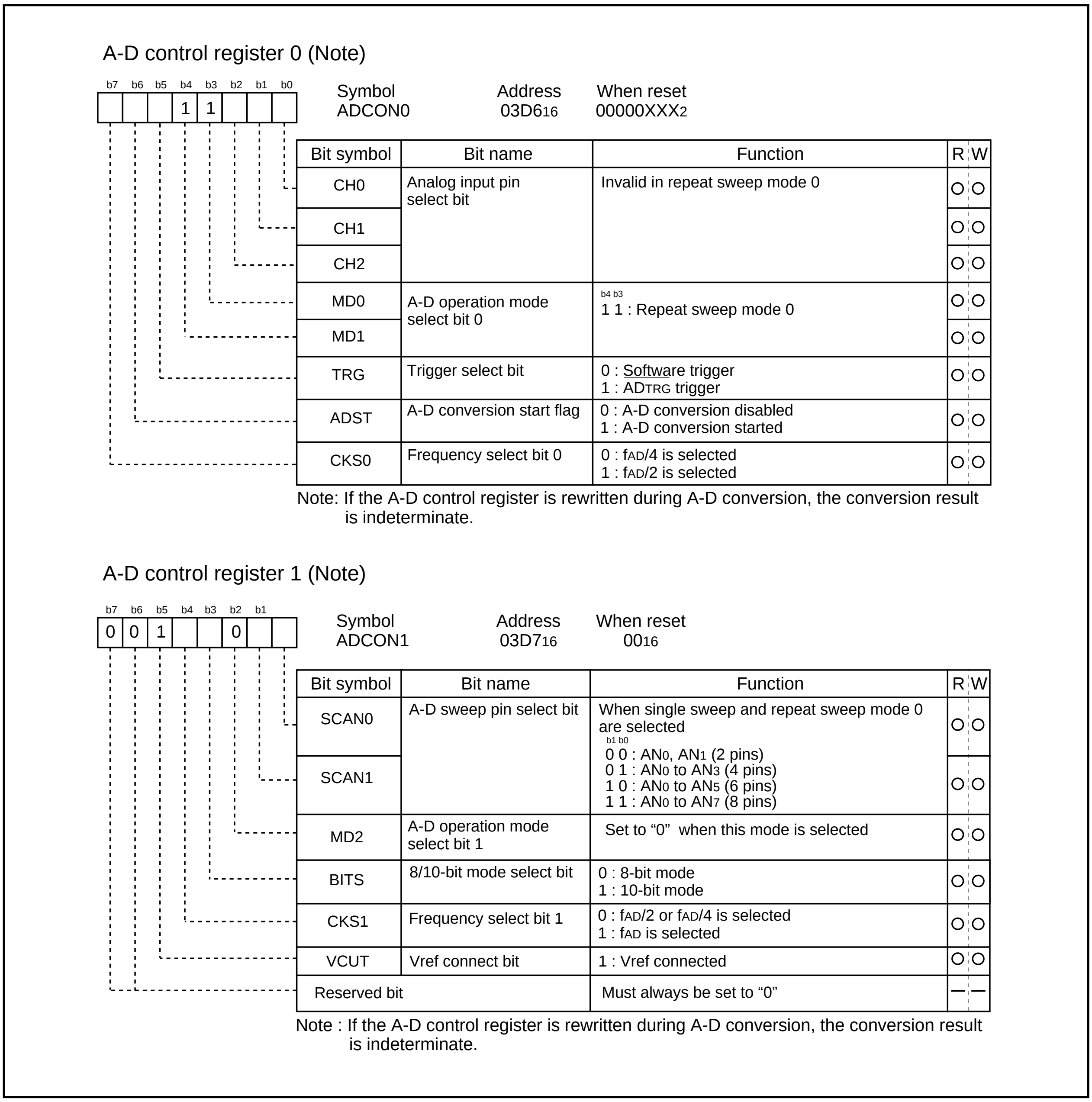
Figure 1.17.6. A-D conversion register in single sweep mode

(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 1.17.5 shows the specifications of repeat sweep mode 0. Figure 1.17.7 shows the A-D control register in repeat sweep mode 0.

Table 1.17.5. Repeat sweep mode 0 specifications

Item	Specification
Function	The pins selected by the A-D sweep pin select bit are used for repeat 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 : ADTRG trigger	○	○
ADST	A-D conversion start flag	0 : A-D conversion disabled 1 : A-D conversion started	○	○
CKS0	Frequency select bit 0	0 : $f_{AD}/4$ is selected 1 : $f_{AD}/2$ 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)

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	Set to “0” when this mode is selected	○	○
BITS	8/10-bit mode select bit	0 : 8-bit mode 1 : 10-bit mode	○	○
CKS1	Frequency select bit 1	0 : $f_{AD}/2$ or $f_{AD}/4$ is selected 1 : f_{AD} is selected	○	○
VCUT	Vref connect bit	1 : Vref connected	○	○
Reserved bit		Must always be set to “0”	—	—

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

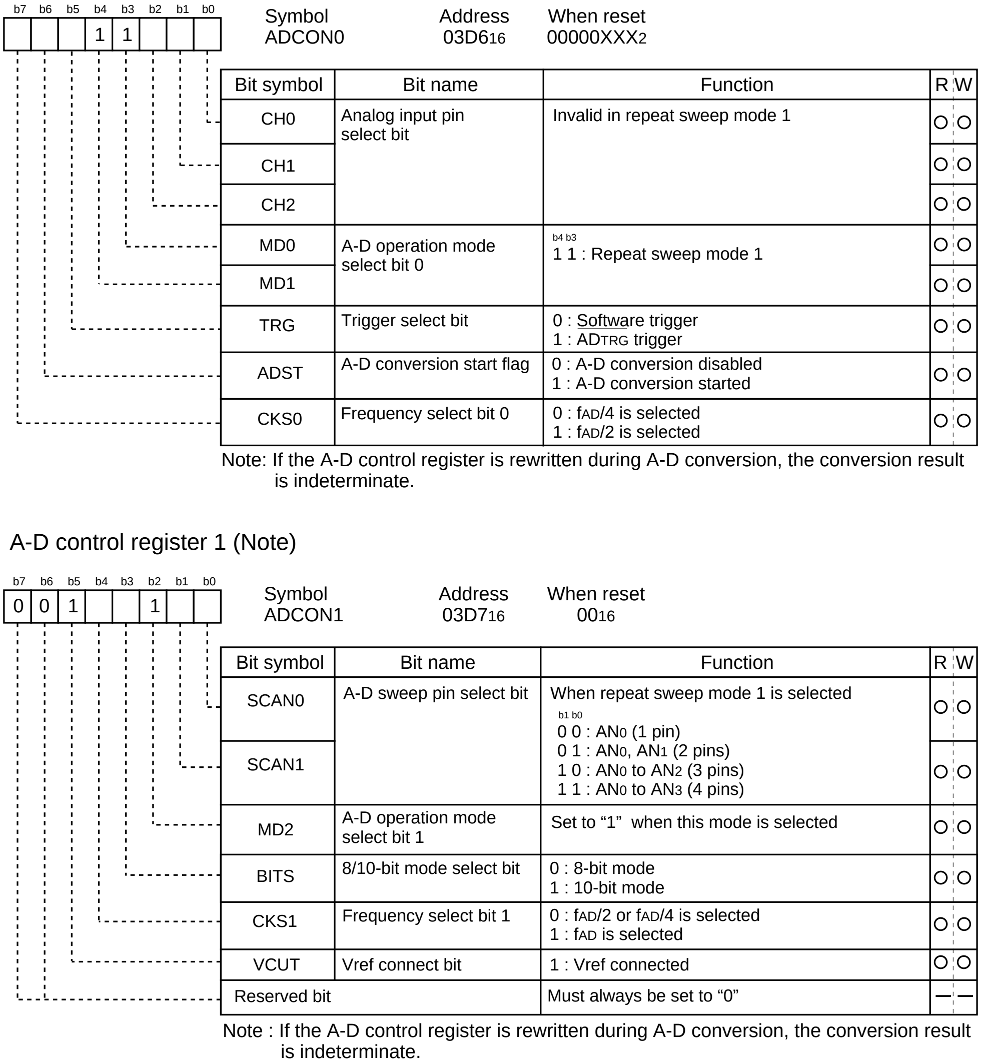
Figure 1.17.7. A-D conversion register in repeat sweep mode 0

(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 1.17.6 shows the specifications of repeat sweep mode 1. Figure 1.17.8 shows the A-D control register in repeat sweep mode 1.

Table 1.17.6. Repeat sweep mode 1 specifications

Item	Specification
Function	All pins perform repeat 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	With emphasis on these pins ; 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)

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$_{AD}$/4 is selected 1 : f$_{AD}$/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)

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	Set to “1” when this mode is selected	○	○
BITS	8/10-bit mode select bit	0 : 8-bit mode 1 : 10-bit mode	○	○
CKS1	Frequency select bit 1	0 : f$_{AD}$/2 or f$_{AD}$/4 is selected 1 : f$_{AD}$ is selected	○	○
VCUT	Vref connect bit	1 : Vref connected	○	○
Reserved bit		Must always be set to “0”	—	—

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

Figure 1.17.8. A-D conversion register in repeat sweep mode 1

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 ϕ_{AD} cycle is achieved with 8-bit resolution and 33 ϕ_{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. When sample and hold is selected, apply a 4.0 V - 5.5 V voltage to Vcc.

D-A Converter

This is an 8-bit, R-2R type D-A converter. The microcomputer contains three 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 to 2 (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.

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

V_{REF} : reference voltage

Table 1.18.1 lists the performance of the D-A converter. Figure 1.18.1 shows the block diagram of the D-A converter. Figure 1.18.2 shows the D-A control register. Figure 1.18.3 shows the D-A converter equivalent circuit.

Table 1.18.1. Performance of D-A converter

Item	Performance
Conversion method	R-2R method
Resolution	8 bits
Analog output pin	3 channels

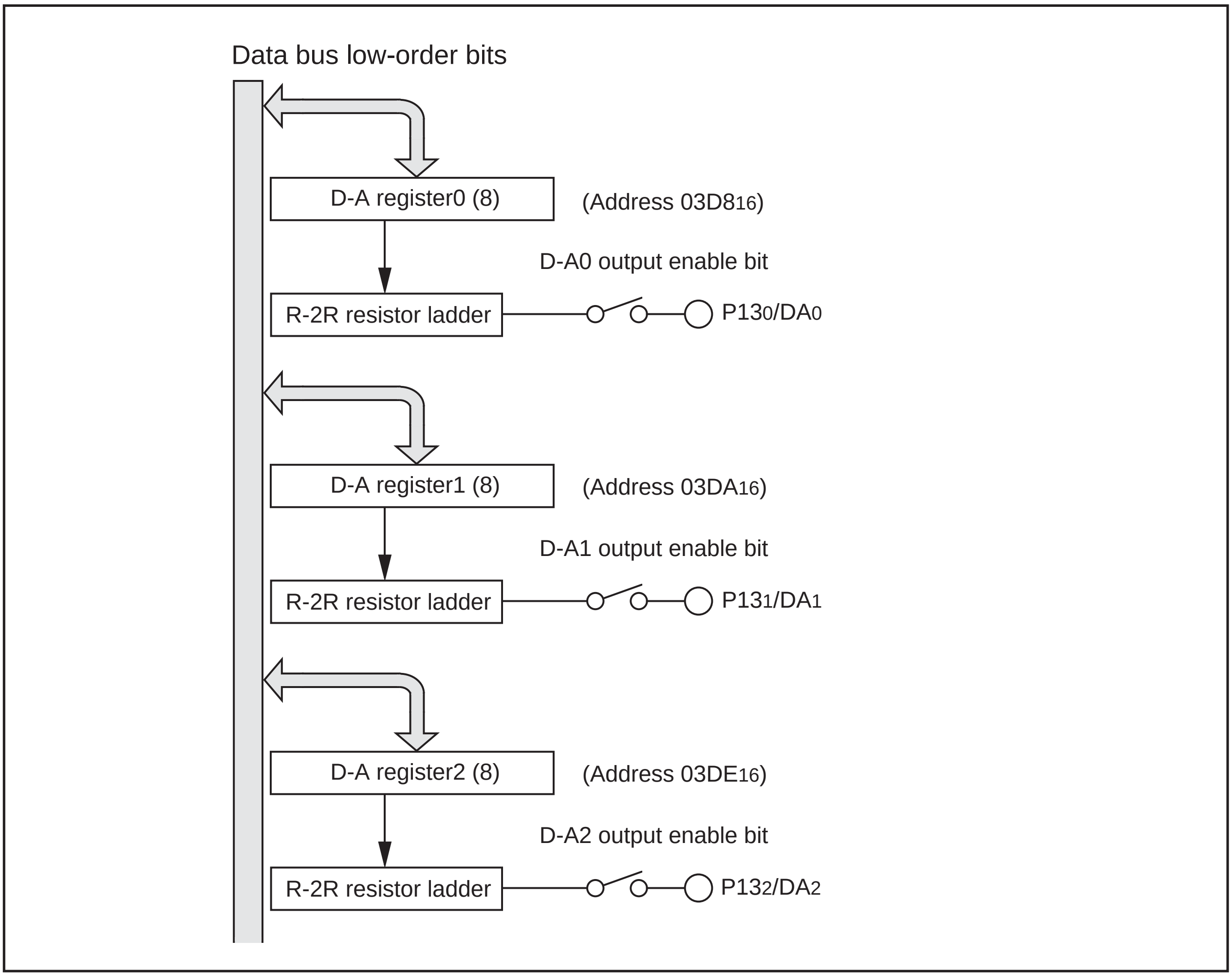
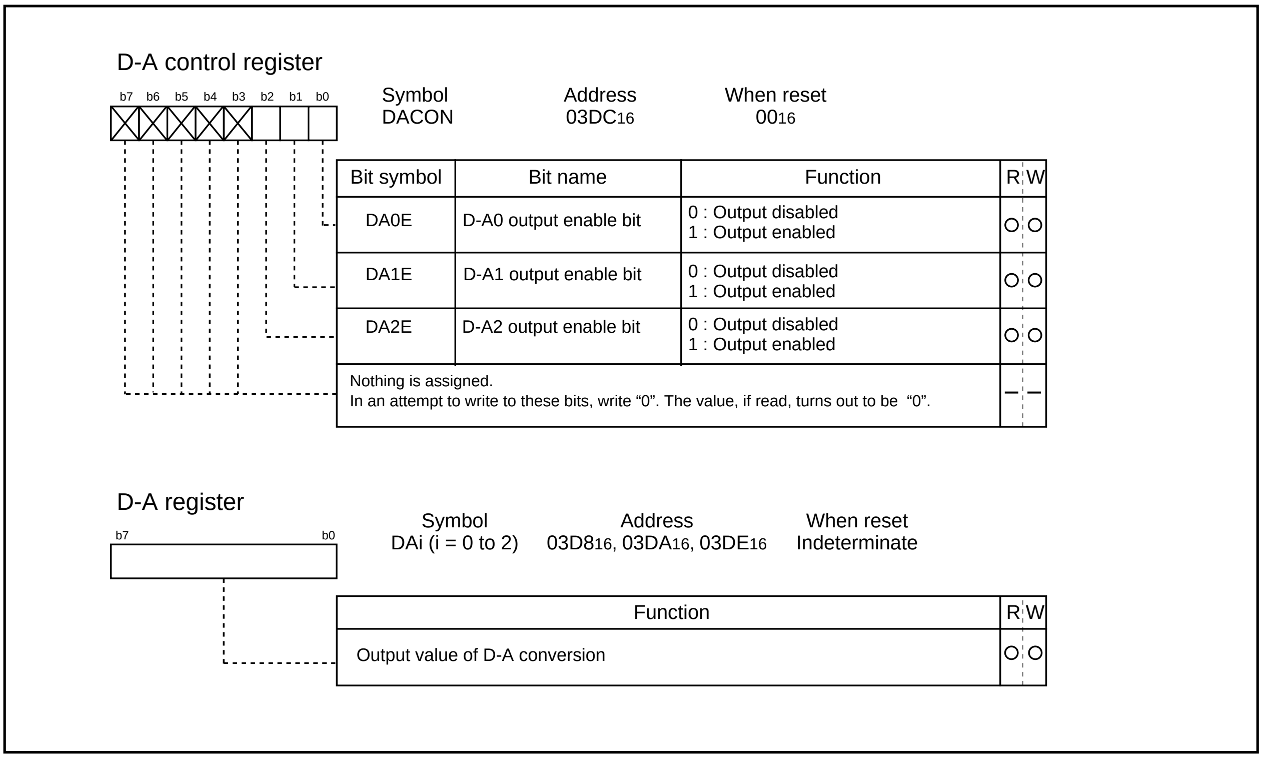


Figure 1.18.1. Block diagram of D-A converter



D-A control register

b7	b6	b5	b4	b3	b2	b1	b0
☒	☒	☒	☒	☒			

Symbol DACON Address 03DC16 When reset 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	○	○
DA2E	D-A2 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”.			–	–

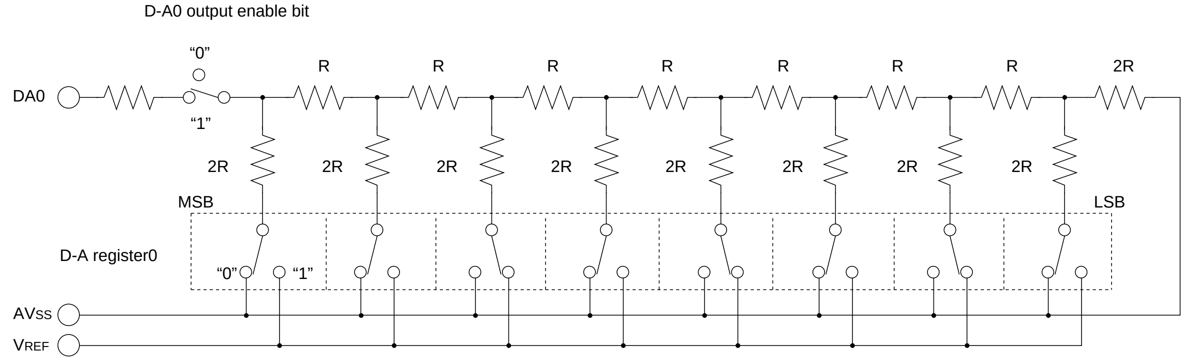
D-A register

b7 b0

Symbol DAi (i = 0 to 2) Address 03D816, 03DA16, 03DE16 When reset Indeterminate

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

Figure 1.18.2. 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 and D-A2.
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 1.18.3. D-A converter equivalent circuit

Programmable I/O Ports

There are 104 programmable I/O ports: P0 to P13 (excluding $P7_7$). Each port can be set independently for input or output using the direction register. A pull-up resistance for each block of 4 ports can be set. $P7_7$ is an input-only port and has no built-in pull-up resistance.

Figures 1.19.1 to 1.19.4 show the programmable I/O ports. Figure 1.19.5 shows the I/O pins.

Each pin functions as a programmable I/O port and as the I/O for the built-in peripheral devices.

To use the pins as the inputs for the built-in peripheral devices, set the direction register of each pin to input mode. When the pins are used as the outputs for the built-in peripheral devices (other than the D-A converter), they function as outputs regardless of the contents of the direction registers. When pins are to be used as the outputs for the D-A converter, do not set the direction registers to output mode. See the descriptions of the respective functions for how to set up the built-in peripheral devices.

(1) Direction registers

Figure 1.19.6 shows the direction registers.

These registers are used to choose the direction of the programmable I/O ports. Each bit in these registers corresponds one for one to each I/O pin.

Note: There is no direction register bit for $P7_7$.

(2) Port registers

Figure 1.19.7 shows the port registers.

These registers are used to write and read data for input and output to and from an external device. A port register consists of a port latch to hold output data and a circuit to read the status of a pin. Each bit in port registers corresponds one for one to each I/O pin.

(3) Pull-up control registers

Figure 1.19.8 shows the pull-up control registers.

The pull-up control register can be set to apply a pull-up resistance to each block of 4 ports. When ports are set to have a pull-up resistance, the pull-up resistance is connected only when the direction register is set for input. The pull-up resistance is not connected for pins that are set for output from peripheral functions, regardless of the setting in the pull-up control register. When pull-up is ON for ports P1 and P2, an intermittent pull-up that pulls up the port for only a set period of time, can be performed from the key input mode register.

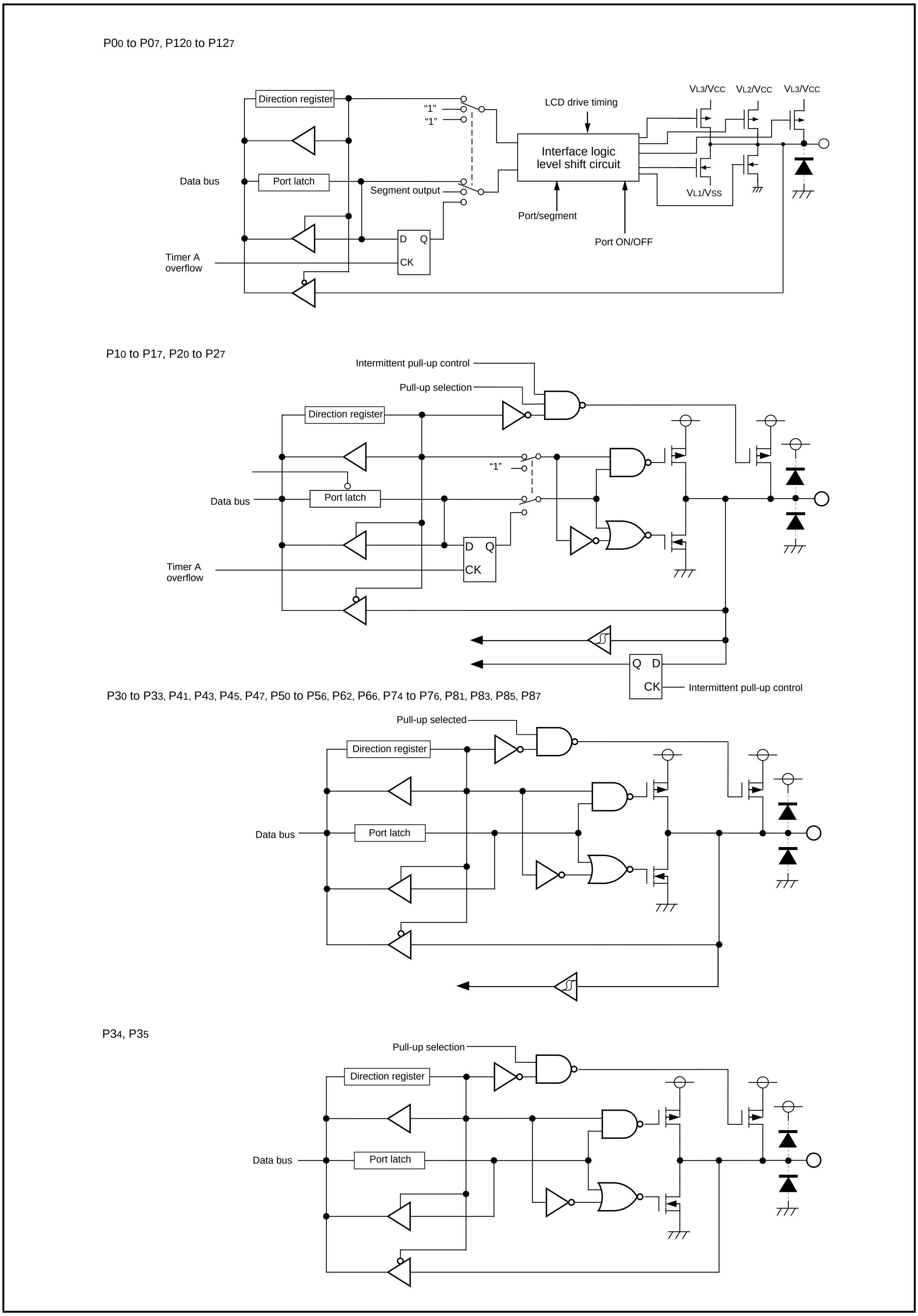
(4) Key input mode register

Figure 1.19.9 shows the key input mode register.

With bits 0 and 1 of this register, it is possible to select both edges or the fall edge of the key input for P1 and P2. Also, with bit 2, it is possible to make the pull-up for a port (P1 or P2), which is set for pull-up using the pull-up control register, automatically connect as an intermittent pull-up. And, using the significant 3 bits, the pull-up resistance can be connected to and disconnected from ports P12 and P13.

(5) Real-time port control register

Figure 1.19.10 shows the real-time port control register. The real-time port control register can be used to set the registers of ports P0, P1, P2 and P12 for real-time port output, whereby output is synchronized with timer overflow of timers A0, A1, A5 and A6 in the timer mode. For details, see “Real-time Port”.



$P0_0$ to $P0_7$, $P12_0$ to $P12_7$

$P1_0$ to $P1_7$, $P2_0$ to $P2_7$

$P3_0$ to $P3_3$, $P4_1$, $P4_3$, $P4_5$, $P4_7$, $P5_0$ to $P5_6$, $P6_2$, $P6_6$, $P7_4$ to $P7_6$, $P8_1$, $P8_3$, $P8_5$, $P8_7$

$P3_4$, $P3_5$

Figure 1.19.1. Programmable I/O ports (1)

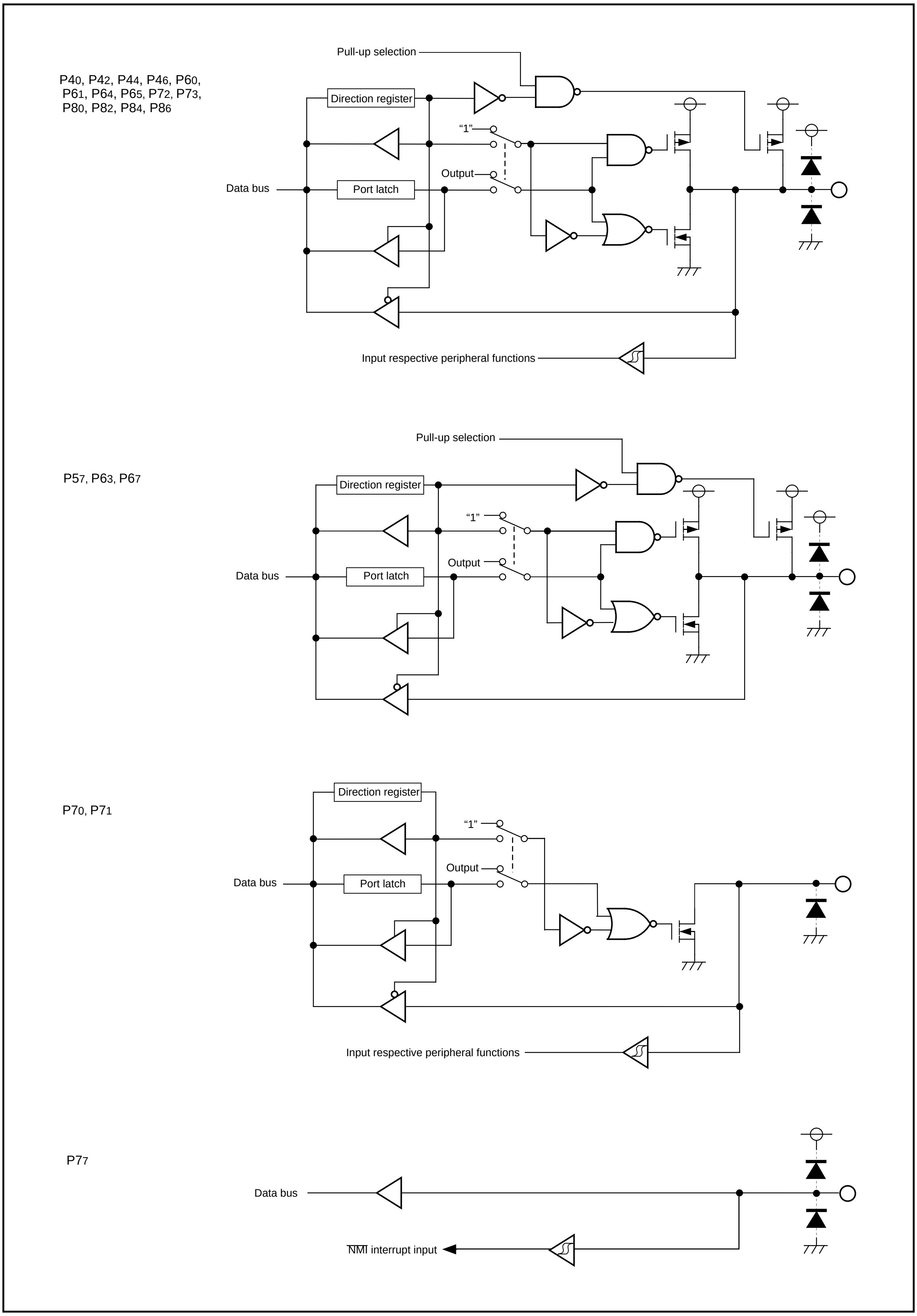
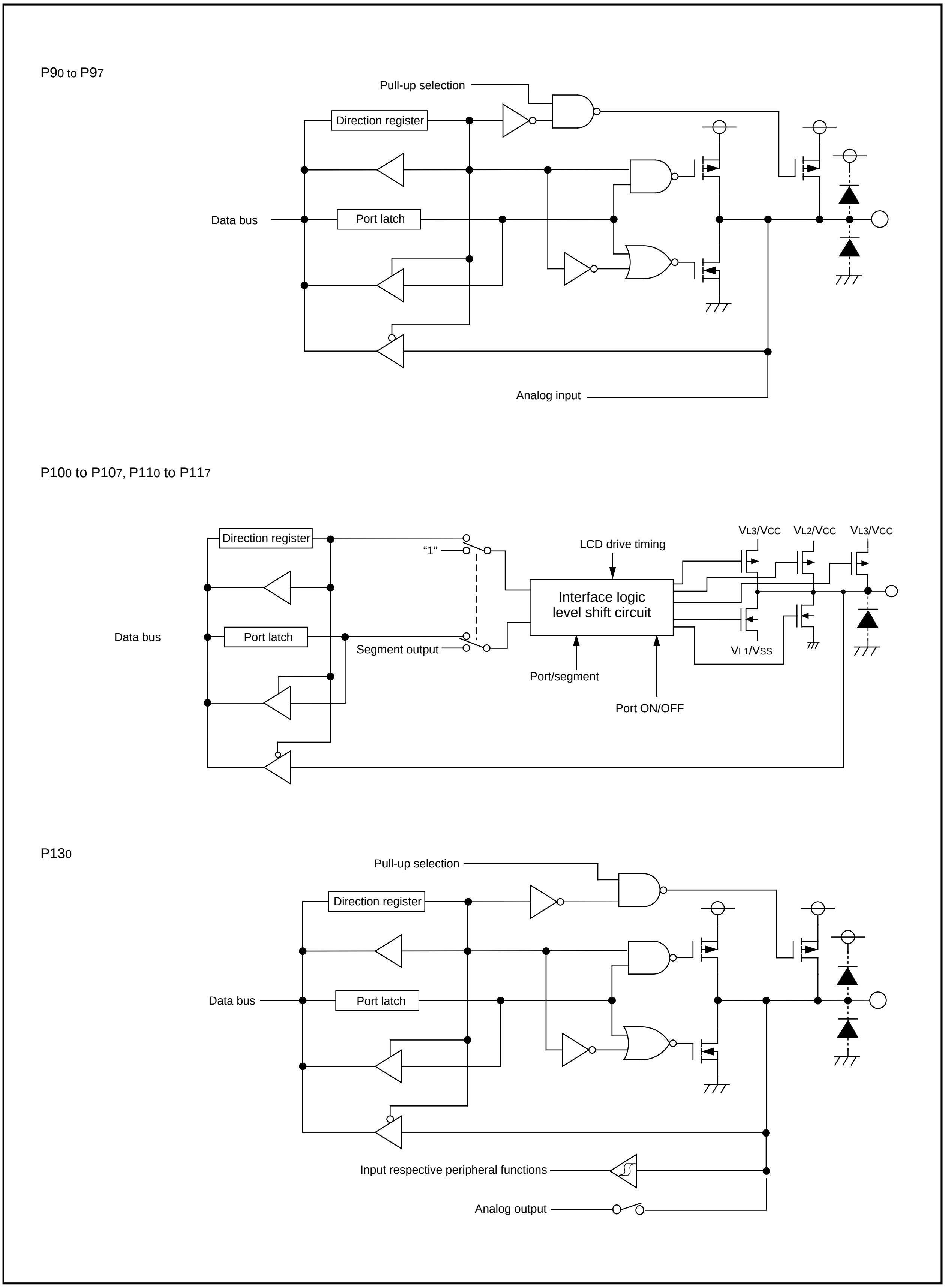


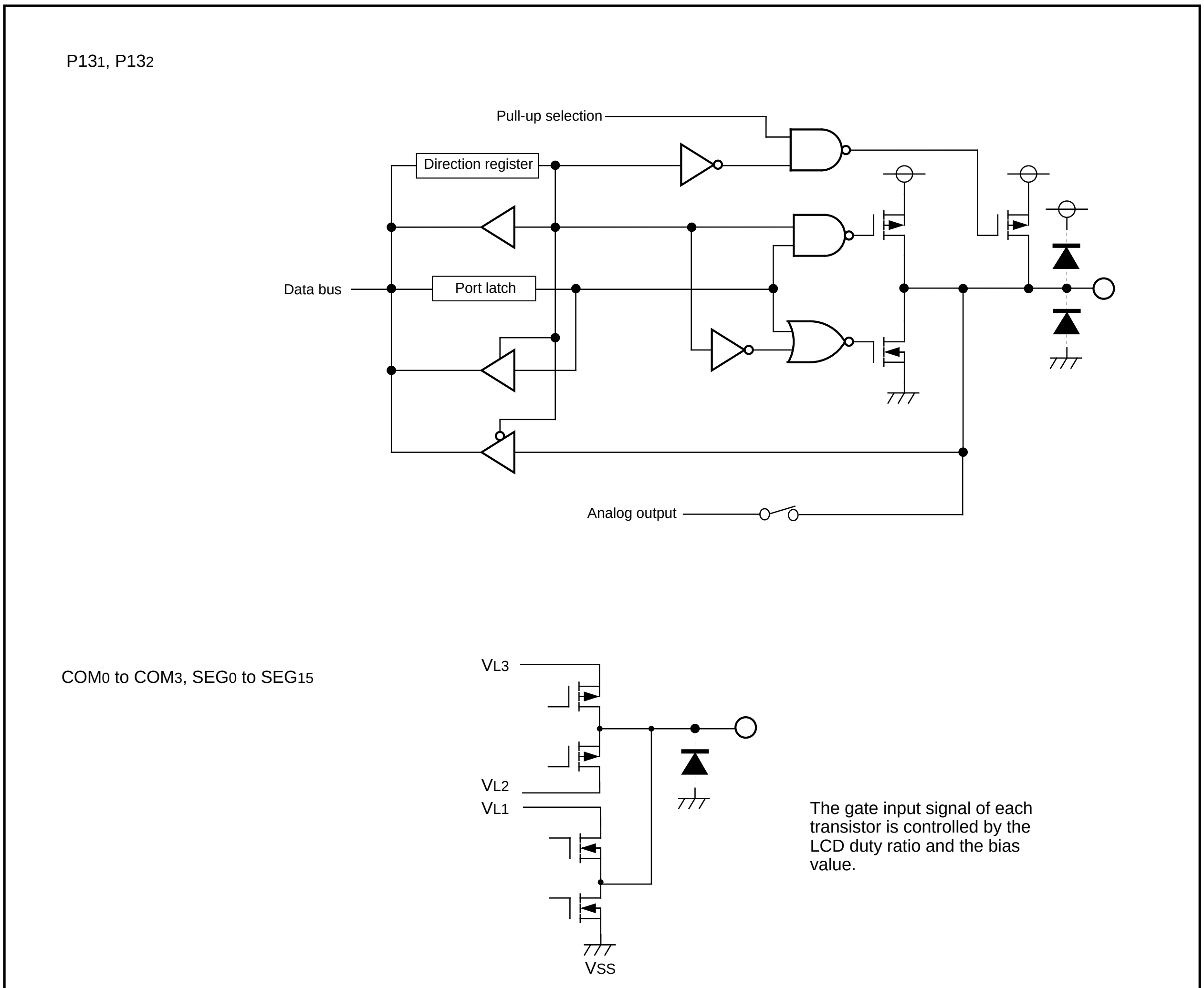
Figure 1.19.2. Programmable I/O ports (2)


P90 to P97

P100 to P107, P110 to P117

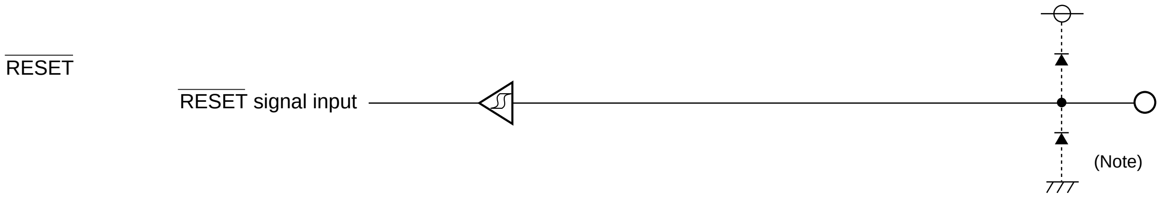
P130

Figure 1.19.3. Programmable I/O ports (3)



The gate input signal of each transistor is controlled by the LCD duty ratio and the bias value.

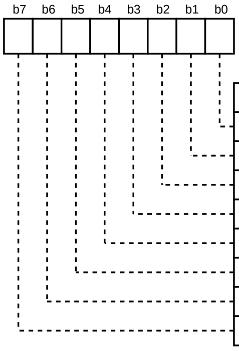
Figure 1.19.4. Programmable I/O ports (4)



Note : ----▶|------ symbolizes a parasitic diode.
Do not apply a voltage higher than VCC to each pin.

Figure 1.19.5. I/O pins

Port Pi direction register (Note)



Symbol: PDi (i = 0 to 12 except 3 and 7)

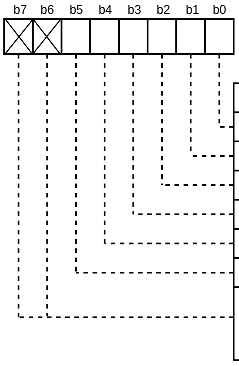
Address: $03E2_{16}$, $03E3_{16}$, $03E6_{16}$, $03EA_{16}$, $03EB_{16}$, $03EE_{16}$, $03F2_{16}$, $03F3_{16}$, $03F6_{16}$, $03F7_{16}$, $03FA_{16}$

When reset: 00_{16}

Bit symbol	Bit name	Function	R	W
PDi_0	Port Pi$_0$ direction register	0 : Input mode (Functions as an input port) 1 : Output mode (Functions as an output port) (i = 0 to 12 except 3 and 7)	○	○
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 : Do not access the Port P12 direction register in words.

Port P3 direction register



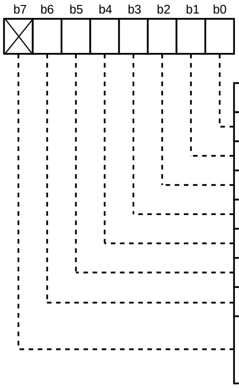
Symbol: PD3

Address: $03E7_{16}$

When reset: $XX000000_2$

Bit symbol	Bit name	Function	R	W
PD3_0	Port P3$_0$ direction register	0: Input mode (Functions as an input port) 1: Output mode (Functions as an output port)	○	○
PD3_1	Port P3$_1$ direction register		○	○
PD3_2	Port P3$_2$ direction register		○	○
PD3_3	Port P3$_3$ direction register		○	○
PD3_4	Port P3$_4$ direction register		○	○
PD3_5	Port P3$_5$ direction register		○	○
Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminate.			—	—

Port P7 direction register



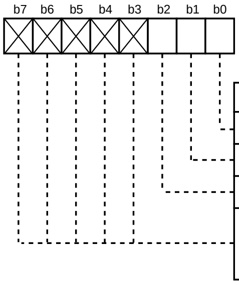
Symbol: PD7

Address: $03EF_{16}$

When reset: $X0000000_2$

Bit symbol	Bit name	Function	R	W
PD7_0	Port P7$_0$ direction register	0: Input mode (Functions as an input port) 1: Output mode (Functions as an output port)	○	○
PD7_1	Port P7$_1$ direction register		○	○
PD7_2	Port P7$_2$ direction register		○	○
PD7_3	Port P7$_3$ direction register		○	○
PD7_4	Port P7$_4$ direction register		○	○
PD7_5	Port P7$_5$ direction register		○	○
PD7_6	Port P7$_6$ direction register		○	○
Nothing is assigned. In an attempt to write to this bit, write “0”. The value, if read, turns out to be indeterminate.			—	—

Port P13 direction register



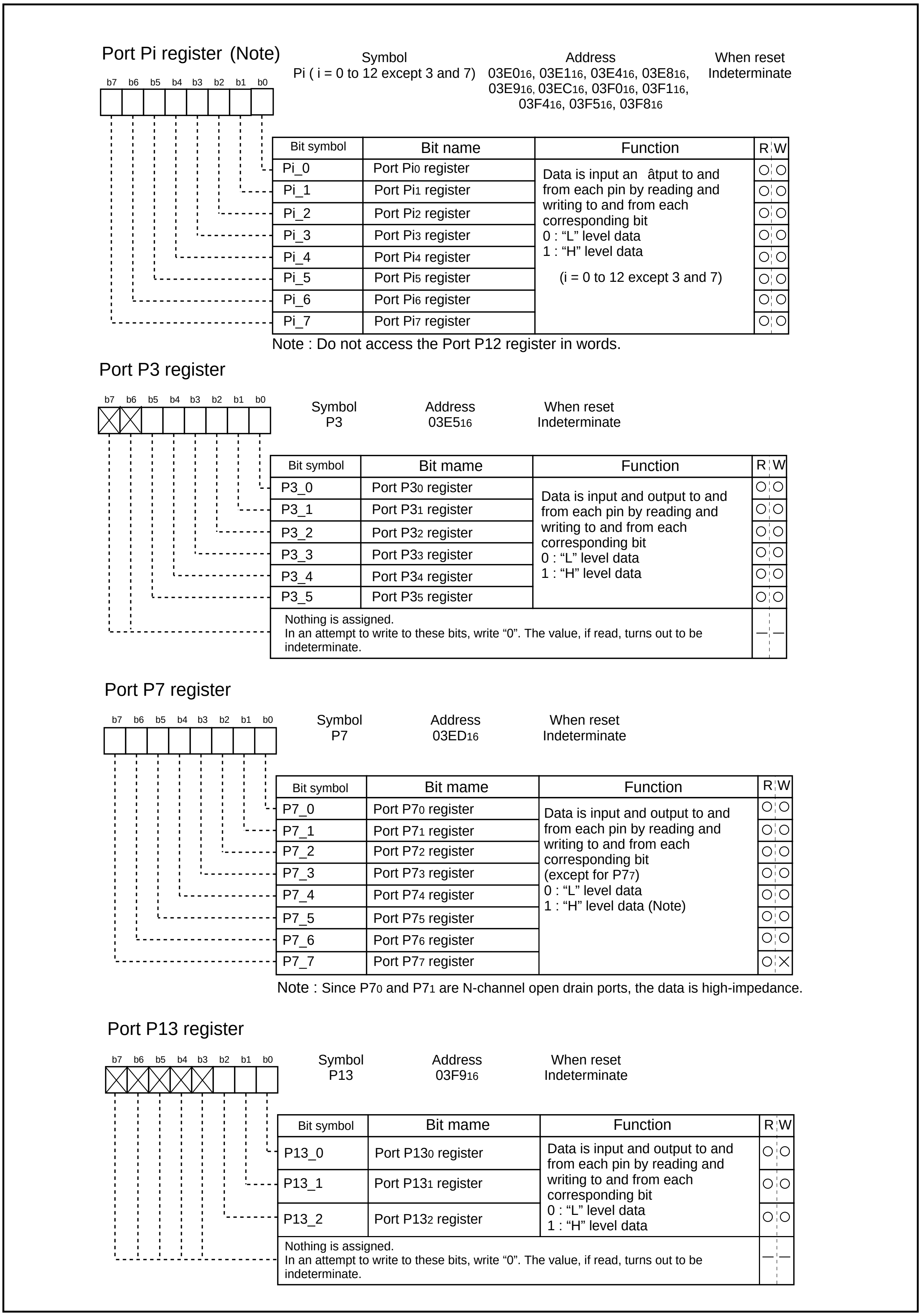
Symbol: PD13

Address: $03FB_{16}$

When reset: $XXXXX000_2$

Bit symbol	Bit name	Function	R	W
PD13_0	Port P13$_0$ direction register	0: Input mode (Functions as an input port) 1: Output mode (Functions as an output port)	○	○
PD13_1	Port P13$_1$ direction register		○	○
PD13_2	Port P13$_2$ direction register		○	○
Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminate.			—	—

Figure 1.19.6. Direction register



Port Pi register (Note)

Symbol: Pi (i = 0 to 12 except 3 and 7)
Address: $03E0_{16}$, $03E1_{16}$, $03E4_{16}$, $03E8_{16}$, $03E9_{16}$, $03EC_{16}$, $03F0_{16}$, $03F1_{16}$, $03F4_{16}$, $03F5_{16}$, $03F8_{16}$
When reset: Indeterminate

Bit symbol	Bit name	Function	R	W
Pi_0	Port Pi_0 register	Data is input an âtput 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 12 except 3 and 7)	○	○
Pi_1	Port Pi_1 register		○	○
Pi_2	Port Pi_2 register		○	○
Pi_3	Port Pi_3 register		○	○
Pi_4	Port Pi_4 register		○	○
Pi_5	Port Pi_5 register		○	○
Pi_6	Port Pi_6 register		○	○
Pi_7	Port Pi_7 register		○	○

Note : Do not access the Port P12 register in words.

Port P3 register

Symbol: P3
Address: $03E5_{16}$
When reset: Indeterminate

Bit symbol	Bit mame	Function	R	W
P3_0	Port $P3_0$ 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	○	○
P3_1	Port $P3_1$ register		○	○
P3_2	Port $P3_2$ register		○	○
P3_3	Port $P3_3$ register		○	○
P3_4	Port $P3_4$ register		○	○
P3_5	Port $P3_5$ register		○	○
Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminate.			—	—

Port P7 register

Symbol: P7
Address: $03ED_{16}$
When reset: Indeterminate

Bit symbol	Bit mame	Function	R	W
P7_0	Port $P7_0$ register	Data is input and output to and from each pin by reading and writing to and from each corresponding bit (except for $P7_7$) 0 : “L” level data 1 : “H” level data (Note)	○	○
P7_1	Port $P7_1$ register		○	○
P7_2	Port $P7_2$ register		○	○
P7_3	Port $P7_3$ register		○	○
P7_4	Port $P7_4$ register		○	○
P7_5	Port $P7_5$ register		○	○
P7_6	Port $P7_6$ register		○	○
P7_7	Port $P7_7$ register		○	×

Note : Since $P7_0$ and $P7_1$ are N-channel open drain ports, the data is high-impedance.

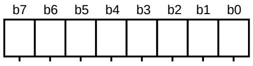
Port P13 register

Symbol: P13
Address: $03F9_{16}$
When reset: Indeterminate

Bit symbol	Bit mame	Function	R	W
P13_0	Port $P13_0$ 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	○	○
P13_1	Port $P13_1$ register		○	○
P13_2	Port $P13_2$ register		○	○
Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminate.			—	—

Figure 1.19.7. Port register

Pull-up control register 0 (Note 1)(Note 2)



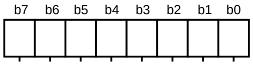
Symbol	Address	When reset
PUR0	$03FC_{16}$	00000011_2

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		○	○

Note 1 : The pull-up resistance is not connected for pins that are set for output from peripheral functions, regardless of the setting in the pull-up control register.

Note 2 : Do not access this register in words.

Pull-up control register 1 (Note)

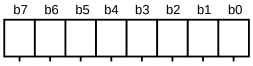


Symbol	Address	When reset
PUR1	$03FD_{16}$	00_{16}

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		○	○
PU17	$P7_4$ to $P7_7$ pull-up		○	○

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

Pull-up control register 2 (Note)



Symbol	Address	When reset
PUR2	$03FE_{16}$	11110000_2

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		○	○
PU22	$P9_0$ to $P9_3$ pull-up		○	○
PU23	$P9_4$ to $P9_7$ pull-up		○	○
PU24	$P10_0$ to $P10_3$ pull-up		○	○
PU25	$P10_4$ to $P10_7$ pull-up		○	○
PU26	$P11_0$ to $P11_3$ pull-up		○	○
PU27	$P11_4$ to $P11_7$ pull-up		○	○

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

Figure 1.19.8. Pull-up control register

Key input mode register

b7 b6 b5 b4 b3 b2 b1 b0

Symbol	Address	When reset
KUPM	0126$_{16}$	01100000$_2$

Bit	Bit name	Function	R	W
P1KIS	P1 key input select bit (Note 1)	0 : Falling edge 1 : Two edges (Note 2)	○	○
P1KIE	P1 key input enable bit	0 : Disable 1 : Enable	○	○
P2KIS	P2 key input select bit (Note 1)	0 : Falling edge 1 : Two edges (Note 2)	○	○
P2KIE	P2 key input enable bit	0 : Disable 1 : Enable	○	○
P3KIE	P3 key input enable bit	0 : Disable 1 : Enable	○	○
PUP12L	P12$_0$ to P12$_3$ pull-up (Note 3)	The corresponding port is pulled high with a pull-up resistor 0 : Not pulled high 1 : Pulled high	○	○
PUP12H	P12$_4$ to P12$_7$ pull-up (Note 3)		○	○
PUP13	P13$_0$ to P13$_2$ pull-up (Note 3)		○	○

Note 1 : If this bit is set for “Two edges” when the corresponding port has been specified to have a pullup, the port is automatically pulled high intermittently. Operating sub-clock.
Note 2 : When this bit is set for “Two edges” and the input from either of the corresponding pin is “L”, if the pullup control register 0 of the corresponding port (bit 2 to 5 at the address 03FC$_{16}$) is changed, there may be the thing that the key input interruption request is set to “1”.
Note 3 : The pull-up resistance is not connected for pins that are set for output from peripheral functions, regardless of the setting in the pull-up control register.

Figure 1.19.9. Key input mode register

Real time port control register

b7 b6 b5 b4 b3 b2 b1 b0

Symbol	Address	When reset
RTP	03FF$_{16}$	00$_{16}$

Bit symbol	Bit name	Function	R	W
RTP0	P0$_0$ to P0$_3$ real time port mode select bit	0 : I/O port 1 : Real time port output (Note)	○	○
RTP1	P0$_4$ to P0$_7$ real time port mode select bit		○	○
RTP2	P1$_0$ to P1$_3$ real time port mode select bit		○	○
RTP3	P1$_4$ to P1$_7$ real time port mode select bit		○	○
RTP4	P2$_0$ to P2$_3$ real time port mode select bit		○	○
RTP5	P2$_4$ to P2$_7$ real time port mode select bit		○	○
RTP6	P12$_0$ to P12$_3$ real time port mode select bit		○	○
RTP7	P12$_4$ to P12$_7$ real time port mode select bit		○	○

Note : The corresponding port direction register is invalidated.

Figure 1.19.10. Realtime port control register

Table 1.19.1. Example connection of unused pins in single-chip mode

Pin name	Connection
Ports P0 to P13 (excluding P77)	After setting for output mode, leave these pins open; or after setting for input mode, connect every pin to VSS via a resistor (Note 1, Note 3).
XOUT (Note 2), XCOUT	Open
XCIN	Connect this pin to VSS via a resistor (pull-down)
NMI	Connect this pin to VCC via a resistor (pull-up)
AVCC	Connect to VCC
AVSS, VREF	Connect to VSS
COM0 to COM3	Open
SEG0 to SEG15	Open
C1, C2	Open
VL2, VL3	Connect to VCC
VL1	Connect to VSS
CNVSS	Connect this pin to VSS via a resistor (pull-down)

Note 1: If setting these pins in output mode and opening them, ports are in input mode until switched into output mode by use of software after reset. Thus the voltage levels of the pins become unstable, and there can be instances in which the power source current increases while the ports are in input mode.
In view of an instance in which the contents of the direction registers change due to a runaway generated by noise or other causes, setting the contents of the direction registers periodically by use of software increases program reliability.

Note 2: When an external clock is input to the XIN pin.

Note 3: Output "L" if port P70 and P71 are set to output mode.
Port P70 and P71 are N channel open drain.

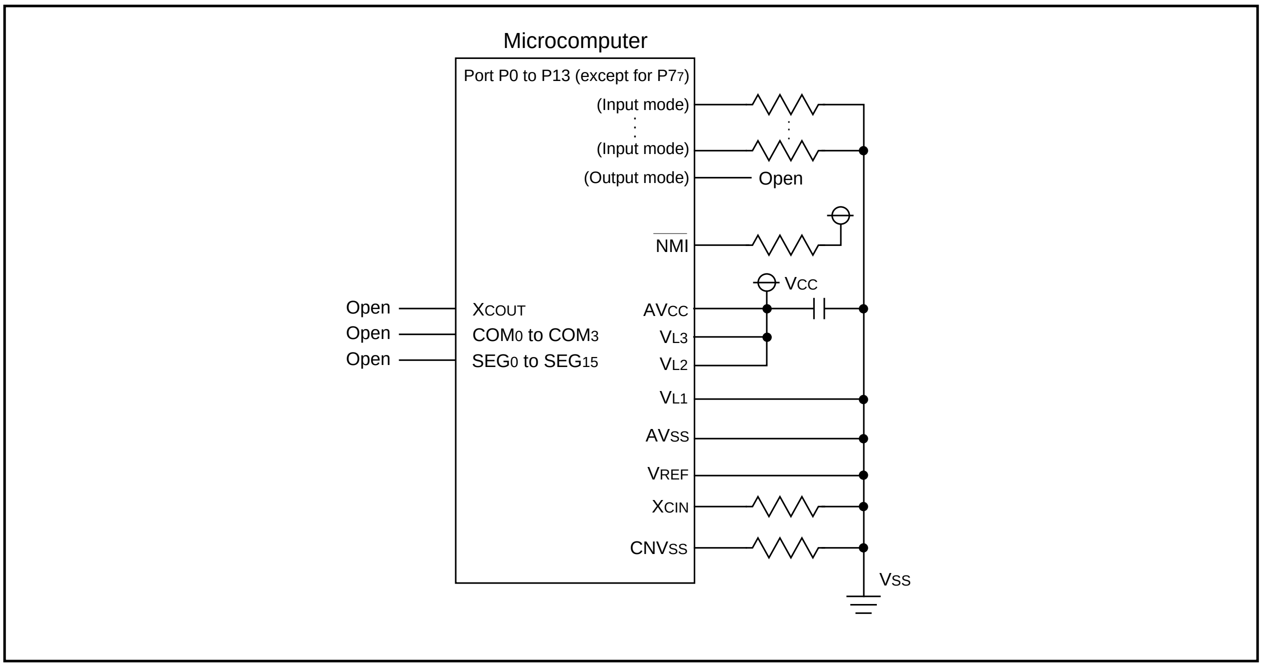


Figure 1.19.11. Example connection of unused pins

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.

Real time port

(1) Make sure timer Ai for real time port output is set for timer mode, and is set to have “no gate function” using the gate function select bit.

(2) Before setting the real time port mode select bit to “1”, temporarily turn off the timer Ai used and write its set value to the timer Ai register.

Serial I/O

When the IIC mode select bit (bit 0 at address 0377_{16}) is set to “1”;

(1) When setting up port P7 (address $03EF_{16}$), write immediate values. If you use Read/Modify/Write instructions (BSET, BCLR, AND, OR, etc..) on the port P7 direction register, the value of $P7_1$ direction register may change to unknown data.

(2) Only for the mask ROM version, when the internal/external clock select bit (bit 3 of address 0378_{16}) is set to “1 (set to slave)”, the SCL wait output bit (bit 2 of address 0376_{16}) and SCL wait output bit 2 (bit 5 of address 0376_{16}) do not function.

(3) Only for the mask ROM version, when the internal/external clock select bit (bit 3 of address 0378_{16}) is set to “1 (set to slave)”, the port $P7_1$ cannot be read unless the port $P7_1$ direction register (bit 1 of address $03EF_{16}$) is set to “0”, although it is specified as follows; “When IICM=1, the port pin shall be able to be read even if the $P7_1$ direction register=1.”

A-D Converter

(1) Write to each bit (except bit 6) of A-D control register 0, to each bit of A-D control register 1, and to bit 0 of A-D control register 2 when A-D conversion is stopped (before a trigger occurs).

In particular, when the Vref connection bit is changed from “0” to “1”, start A-D conversion after an elapse of 1 µs or longer.

(2) When changing A-D operation mode, select analog input pin again.

(3) Using one-shot mode or single sweep mode

Read the correspondence A-D register after confirming A-D conversion is finished. (It is known by A-D conversion interrupt request bit.)

(4) Using repeat mode, repeat sweep mode 0 or repeat sweep mode 1

Use the undivided main clock as the internal CPU clock.

Stop Mode and Wait Mode

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

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

(3) When the MCU running in low-speed or low power dissipation mode, do not enter WAIT mode with WAIT peripheral function clock stop bit set to “1”.

Interrupts

(1) Reading address 00000_{16}

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

(2) Setting the stack pointer

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

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

- The $\overline{\text{NMI}}$ interrupt can not be disabled. Be sure to connect $\overline{\text{NMI}}$ pin to Vcc via a resistor (pull-up) if unused. Be sure to work on it.
- Do not get either into stop mode with the $\overline{\text{NMI}}$ pin set to “L”.

(4) External interrupt

- When the polarity of the $\overline{\text{INT}_0}$ to $\overline{\text{INT}_5}$ pins is changed, the interrupt request bit is sometimes set to "1". After changing the polarity, set the interrupt request bit to "0".

(5) Rewrite the interrupt control register

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

Example 1:

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

Example 2:

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

Example 3:

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

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

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

Table 1.21.1. Absolute maximum ratings

Symbol	Parameter		Condition	Rated value	Unit
Vcc	Supply voltage		Vcc=AVcc	– 0.3 to 6.5	V
AVcc	Analog supply voltage		Vcc=AVcc	– 0.3 to 6.5	V
V_I	Input voltage	$\overline{RESET}$, V_{REF}, X_{IN} $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_5$, $P4_0$ 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 $P10_7$, $P11_0$ to $P11_7$, $P12_0$ to $P12_7$, $P13_0$ to $P13_2$ (Mask ROM version CNVss)		– 0.3 to Vcc+0.3	V
		VL1		– 0.3 to VL2	
		VL2		VL1 to VL3	
		VL3		VL2 to 6.5	
		$P7_0$, $P7_1$, C_1, C_2 (flash memory version CNVss)		– 0.3 to 6.5	
V_O	Output voltage	$P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_5$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_6$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $P13_0$ to $P13_2$, X_{OUT}		– 0.3 to Vcc+0.3	V
		$P0_0$ to $P0_7$, $P10_0$ to $P10_7$, $P11_0$ to $P11_7$, $P12_0$ to $P12_7$,	When output port	– 0.3 to Vcc	
			When segment output	– 0.3 to VL3	
		$P7_0$, $P7_1$		– 0.3 to 6.5	
P_d	Power dissipation		Ta = 25°C	300	mW
T_{opr}	Operating ambient temperature			– 20 to 85	°C
T_{stg}	Storage temperature			– 40 to 150	°C

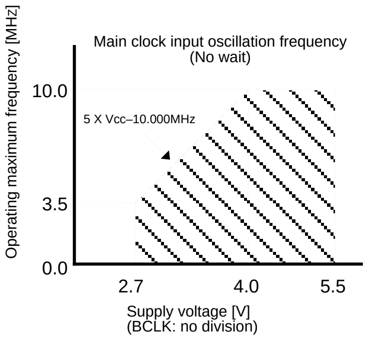
Table 1.21.2. Recommended operating conditions (referenced to V_{CC} = 2.7V to 5.5V at Ta = – 20 to 85^{O}C unless otherwise specified)

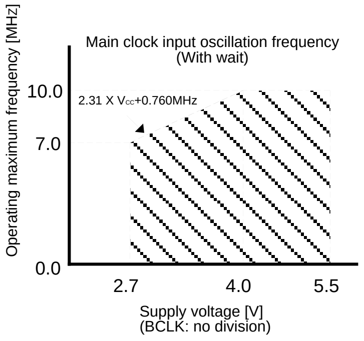
Symbol	Parameter			Standard			Unit
				Min	Typ.	Max.	
V_{CC}	Supply voltage			2.7	5.0	5.5	V
AV_{CC}	Analog supply voltage				V_{CC}		V
V_{SS}	Analog supply voltage				0		V
AV_{SS}	Analog supply voltage				0		V
V_{IH}	HIGH input voltage	$P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_5$, $P4_0$ 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 $P10_7$, $P11_0$ to $P11_7$, $P12_0$ to $P12_7$, 13_0 to $P13_2$, X_{IN}, RESET, CNV_{SS}		0.8Vcc		Vcc	V
		$P7_0$, $P7_1$		0.8Vcc		6.5	
V_{IL}	LOW input voltage	$P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_5$, $P4_0$ 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 $P10_7$, $P11_0$ to $P11_7$, $P12_0$ to $P12_7$, 13_0 to $P13_2$, X_{IN}, RESET, CNV_{SS}		0		0.2Vcc	V
$I_{OH (peak)}$	HIGH peak output current (Note 2)	$P0_0$ to $P0_7$, $P10_0$ to $P10_7$, $P11_0$ to $P11_7$, $P12_0$ to $P12_7$				–0.5	mA
		$P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_5$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_6$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $P13_0$ to $P13_2$				–10.0	
$I_{OH (avg)}$	HIGH average output current (Note 1)	$P0_0$ to $P0_7$, $P10_0$ to $P10_7$, $P11_0$ to $P11_7$, $P12_0$ to $P12_7$				–0.1	mA
		$P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_5$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_6$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $P13_0$ to $P13_2$				–5.0	
$I_{OL (peak)}$	LOW peak output current (Note 2)	$P0_0$ to $P0_7$, $P10_0$ to $P10_7$, $P11_0$ to $P11_7$, $P12_0$ to $P12_7$				5.0	mA
		$P1_0$ to $P1_7$, $P2_0$ to $P2_7$,$P3_0$ to $P3_5$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_6$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $P13_0$ to $P13_2$				10.0	
$I_{OL (avg)}$	LOW average output current (Note 1)	$P0_0$ to $P0_7$, $P10_0$ to $P10_7$, $P11_0$ to $P11_7$, $P12_0$ to $P12_7$				2.5	mA
		$P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_5$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_6$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $P13_0$ to $P13_2$				5.0	
f (X_{IN})	Main clock input oscillation frequency (Note 3)	No wait	V_{CC}=4.0V to 5.5V	0		10	MHz
			V_{CC}=2.7V to 4.0V	0		5 x V_{CC} –10.000	MHz
		With wait	V_{CC}=4.0V to 5.5V	0		10	MHz
			V_{CC}=2.7V to 4.0V	0		2.31 x V_{CC} +0.760	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, $P3_0$ to $P3_5$, P4, P5, P6, $P7_0$ to $P7_6$ and $P12_2$ to $P12_7$ must be 80mA max. The total I_{OH} (peak) for ports P0, P1, P2, $P3_0$ to $P3_5$, P4, P5, P6, $P7_2$ to $P7_6$ and $P12_2$ to $P12_7$ must be 80mA max. The total I_{OL} (peak) for ports P8, P9, P10, P11, $P12_0$, $P12_1$ and $P13_0$ to $P13_2$ must be 80mA max. The total I_{OH} (peak) for ports P8, P9, P10, P11, $P12_0$,$P12_1$ and $P13_0$ to $P13_2$ must be 80mA max.

Note 3: Relationship between main clock oscillation frequency and supply voltage.





V_{CC} = 5V

Table 1.21.3. Electrical characteristics (referenced to V_{CC} = 5V, V_{SS} = 0V at Ta = 25°C, f(X_{IN})=10MHz unless otherwise specified)

Symbol	Parameter			Measuring condition	Standard			Unit
					Min	Typ.	Max.	
V_{OH}	HIGH output voltage	$P0_0$ to $P0_7$, $P10_0$ to $P10_7$, $P11_0$ to $P11_7$, $P12_0$ to $P12_7$		I_{OH}= –0.1mA	3.0			V
V_{OH}	HIGH output voltage	$P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_5$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_6$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $P13_0$ to $P13_2$		I_{OH}= –5mA	3.0			V
				I_{OH}= –200μA	4.7			
V_{OH}	HIGH output voltage	X_{OUT}	HIGHPOWER	I_{OH}= –1mA	3.0			V
			LOWPOWER	I_{OH}= –0.5mA	3.0			
V_{OH}	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 $P3_5$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_6$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, $P11_0$ to $P11_7$, $P12_0$ to $P12_7$, $P13_0$ to $P13_2$		I_{OL}=5mA			2.0	V
				I_{OL}=200μA			0.45	
V_{OL}	LOW output voltage	X_{OUT}	HIGHPOWER	I_{OL}=1mA			2.0	V
			LOWPOWER	I_{OL}=0.5mA			2.0	
V_{OL}	LOW output voltage	X_{COUT}	HIGHPOWER	With no load applied		0		V
			LOWPOWER	With no load applied		0		
V_{T+}-V_{T-}	Hysteresis	$TA0_{IN}$ to $TA7_{IN}$, $TB0_{IN}$ to $TB5_{IN}$, $\overline{INT_0}$ to $\overline{INT_5}$, $\overline{AD_{TRG}}$, $\overline{CTS_0}$, $\overline{CTS_1}$, CLK_0, CLK_1, $\overline{NMI}$, $TA2_{OUT}$ to $TA4_{OUT}$, $TA7_{OUT}$, $\overline{KI_0}$ to $\overline{KI_{15}}$ (Note), $\overline{KI_{16}}$ to $\overline{KI_{19}}$			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 $P3_5$, $P4_0$ 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 $P10_7$, $P11_0$ to $P11_7$, $P12_0$ to $P12_7$, $P13_0$ to $P13_2$, X_{IN}, $\overline{RESET}$, CNV_{SS}		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 $P3_5$, $P4_0$ 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 $P10_7$, $P11_0$ to $P11_7$, $P12_0$ to $P12_7$, $P13_0$ to $P13_2$, X_{IN}, $\overline{RESET}$, CNV_{SS}		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 $P3_5$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_6$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, $P11_0$ to $P11_7$, $P12_0$ to $P12_7$, $P13_0$ to $P13_2$		V_I=0V	30.0	50.0	167.0	kΩ
R_{fXIN}	Feedback resistance	X_{IN}				1.0		MΩ
R_{fXCIN}	Feedback resistance	X_{CIN}				6.0		MΩ
V_{RAM}	RAM retention voltage			When clock is stopped	2.0			V

Note : Has no effect during intermittent pullup operation.

VCC = 5V

Table 1.21.4. Electrical characteristics (referenced to VCC = 5V, VSS = 0V at Ta = 25°C, f(XIN)=10MHZ unless otherwise specified)

Symbol	Parameter		Measuring condition		Standard			Unit
					Min.	Typ.	Max.	
I_{CC}	Power supply current	I/o pin is no load applied	Mask ROM, flash memory versions	f(XIN)=10MHz Square wave, no division		19.0	38.0	mA
			Mask ROM version	f(XCIN)=32kHz Square wave		90.0		μA
			Flash memory version	f(XCIN)=32kHz Square wave		200.0		μA
			Mask ROM, flash memory versions	f(XCIN)=32kHz When a WAIT instruction is executed		4.0		μA
				When clock is stopped Ta=25 °C			1.0	μA
				When clock is stopped Ta=85 °C			20.0	
V_{L3}	Supply voltage (V_{L3}) (Note)		When charge-pump not used		2.7		6.5	V
V_{L1}	Supply voltage (V_{L1})		When charge-pump used		1.3	1.7	2.1	V
I_{L1}	Power supply current (V_{L1})		VL1=1.7V, f(LCDCK) = 200Hz			3.0	6.0	μA

Note: Rating: V_{L1}=-0.3 V to V_{L2}, V_{L2}=V_{L1} to V_{L3}, V_{L3}=V_{L2} to 6.5V.

Table 1.21.5. A-D conversion characteristics (referenced to VCC = AVCC = VREF = 5V, Vss = AVSS = 0V at Ta = 25°C, f(XIN) = 10MHZ unless otherwise specified)

Symbol	Parameter		Measuring condition	Standard			Unit
				Min.	Typ.	Max.	
–	Resolution		V_{REF} =V_{CC}			10	Bits
–	Absolute accuracy	Sample & hold function not available	V_{REF} =V_{CC} = 5V			±3	LSB
		Sample & hold function available(10bit)	V_{REF} =V_{CC}= 5V			±3	LSB
		Sample & hold function available(8bit)	V_{REF} = V_{CC} = 5V			±2	LSB
R_{LADDER}	Ladder resistance		V_{REF} =V_{CC}	10		40	kΩ
t_{CONV}	Conversion time(10bit)			3.3			μs
t_{CONV}	Conversion time(8bit)			2.8			μs
t_{SAMP}	Sampling time			0.3			μs
V_{REF}	Reference voltage			2		V_{CC}	V
V_{IA}	Analog input voltage			0		V_{REF}	V

Table 1.21.6. D-A conversion characteristics (referenced to VCC = AVCC =VREF =5V, VSS = AVSS = 0V at Ta = 25°C, f(XIN) = 10MHZ unless otherwise specified)

Symbol	Parameter	Measuring condition	Standard			Unit
			Min.	Typ.	Max.	
–	Resolution				8	Bits
–	Absolute accuracy				1.0	%
t_{su}	Setup time				3	μs
R_O	Output resistance		4	10	20	kΩ
I_{VREF}	Reference power supply input current	(Note)			1.5	mA

Note: This applies when using one D-A converter, with the D-A register for the unused D-A converter set to "00_{16}".

The A-D converter's ladder resistance is not included.

Also, when the Vref is unconnected at the A-D control register, I_{VREF} is sent.

VCC = 5V

Timing requirements (referenced to VCC = 5V, VSS = 0V at Ta = 25°C unless otherwise specified)

Table 1.21.7. External clock input

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t_c	External clock input cycle time	100		ns
$t_{w(H)}$	External clock input HIGH pulse width	40		ns
$t_{w(L)}$	External clock input LOW pulse width	40		ns
t_r	External clock rise time		15	ns
t_f	External clock fall time		15	ns

Table 1.21.8. 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 1.21.9. 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 1.21.10. Timer A input (external trigger input in one-shot timer mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(TA)}$	TAi$_{IN}$ input cycle time	200		ns
$t_{w(TAH)}$	TAi$_{IN}$ input HIGH pulse width	100		ns
$t_{w(TAL)}$	TAi$_{IN}$ input LOW pulse width	100		ns

Table 1.21.11. Timer A input (external trigger input in pulse width modulation mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{w(TAH)}$	TAi$_{IN}$ input HIGH pulse width	100		ns
$t_{w(TAL)}$	TAi$_{IN}$ input LOW pulse width	100		ns

Table 1.21.12. 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
$t_{su(UP-TIN)}$	TAi$_{OUT}$ input setup time	400		ns
$t_{h(TIN-UP)}$	TAi$_{OUT}$ input hold time	400		ns

V_{CC} = 5V

Timing requirements (referenced to V_{CC} = 5V, V_{SS} = 0V at Ta = 25^{o}C unless otherwise specified)

Table 1.21.13. 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 1.21.14. 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 1.21.15. 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 1.21.16. A-D trigger input

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(AD)}$	$\overline{\text{AD}_{\text{TRG}}}$ input cycle time (trigger able minimum)	1000		ns
$t_{w(ADL)}$	$\overline{\text{AD}_{\text{TRG}}}$ input LOW pulse width	125		ns

Table 1.21.17. 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 1.21.18. External interrupt $\overline{\text{INTi}}$ inputs

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{w(INH)}$	$\overline{\text{INTi}}$ input HIGH pulse width	250		ns
$t_{w(INL)}$	$\overline{\text{INTi}}$ input LOW pulse width	250		ns

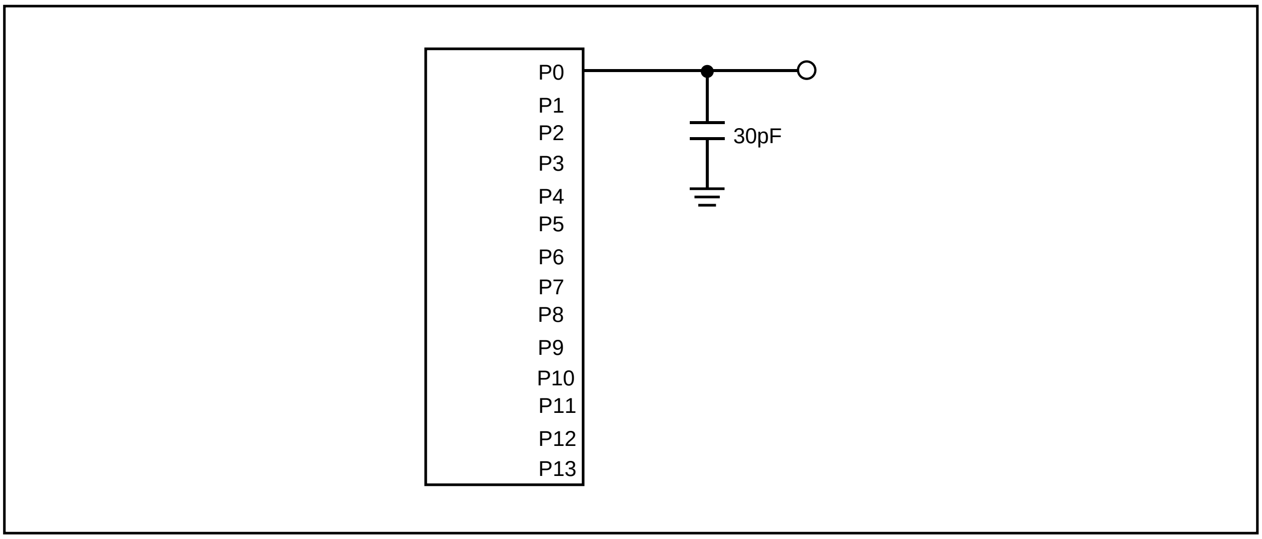
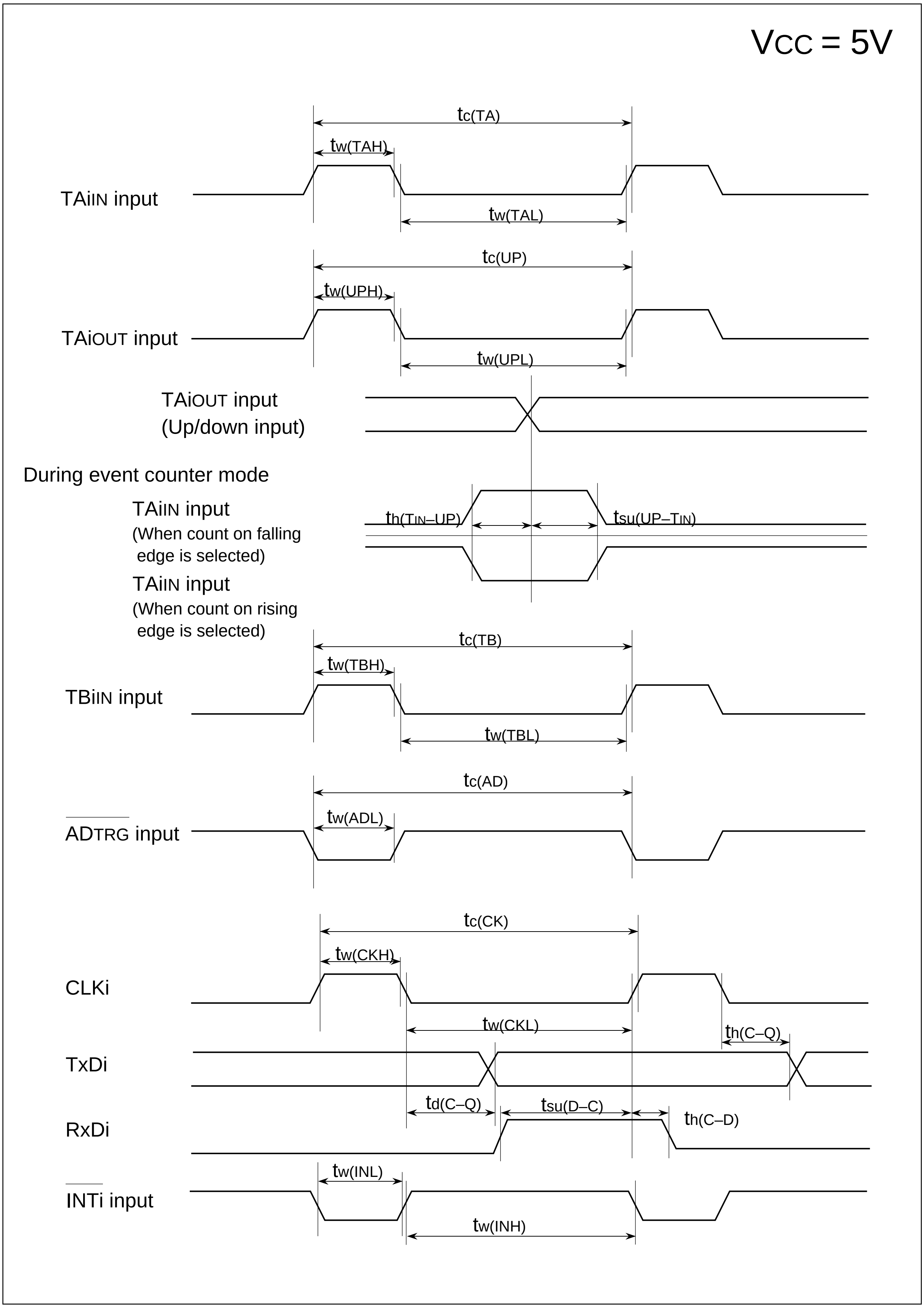


Figure 1.21.1. Port P0 to P13 measurement circuit


VCC = 5V
tc(TA)
tw(TAH)
TAiIN input
tw(TAL)
tc(UP)
tw(UPH)
TAiOUT input
tw(UPL)
TAiOUT input
(Up/down input)
During event counter mode
TAiIN input
(When count on falling edge is selected)
th(TIN–UP)
tsu(UP–TIN)
TAiIN input
(When count on rising edge is selected)
tc(TB)
tw(TBH)
TBiIN input
tw(TBL)
tc(AD)
tw(ADL)
ADTRG input
tc(CK)
tw(CKH)
CLKi
tw(CKL)
th(C–Q)
TxDi
td(C–Q)
tsu(D–C)
th(C–D)
RxDi
tw(INL)
INTi input
tw(INH)

V_{CC} = 3V

Table 1.21.19. Electrical characteristics (referenced to V_{CC} = 3V, V_{SS} = 0V at Ta = 25^{o}C, f(X_{IN}) = 7MH$_Z$, with wait)

Symbol	Parameter			Measuring condition	Standard			Unit
					Min	Typ.	Max.	
V_{OH}	HIGH output voltage	$P0_0$ to $P0_7$, $P10_0$ to $P10_7$, $P11_0$ to $P11_7$, $P12_0$ to $P12_7$		I_{OH}= –20µA	2.0			V
V_{OH}	HIGH output voltage	$P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_5$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_6$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $P13_0$ to $P13_2$		I_{OH}= –1mA	2.5			V
V_{OH}	HIGH output voltage	X_{OUT}	HIGHPOWER	I_{OH}= –0.1mA	2.5			V
			LOWPOWER	I_{OH}= –50µA	2.5			
V_{OH}	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 $P3_5$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_6$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, $P11_0$ to $P11_7$, $P12_0$ to $P12_7$, $P13_0$ to $P13_2$		I_{OL}=1mA			0.5	V
V_{OL}	LOW output voltage	X_{OUT}	HIGHPOWER	I_{OL}=0.1mA			0.5	V
			LOWPOWER	I_{OL}=50µA			0.5	
V_{OL}	LOW output voltage	X_{COUT}	HIGHPOWER	With no load applied		0		V
			LOWPOWER	With no load applied		0		
V_{T+}-V_{T-}	Hysteresis	$TA0_{IN}$ to $TA7_{IN}$, $TB0_{IN}$ to $TB5_{IN}$, $\overline{INT_0}$ to $\overline{INT_5}$, $\overline{AD_{TRG}}$, $\overline{CTS_0}$, $\overline{CTS_1}$, CLK_0, CLK_1, $\overline{NMI}$, $TA2_{OUT}$ to $TA4_{OUT}$, $TA7_{OUT}$, $\overline{KI_0}$ to $\overline{KI_{15}}$ (Note), $\overline{KI_{16}}$ to $\overline{KI_{19}}$			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 $P3_5$, $P4_0$ 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 $P10_7$, $P11_0$ to $P11_7$, $P12_0$ to $P12_7$, $P13_0$ to $P13_2$, X_{IN}, $\overline{RESET}$, CNV_{SS}		V_I=3V			4.0	µA
I_{IL}	LOW input current	$P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_5$, $P4_0$ 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 $P10_7$, $P11_0$ to $P11_7$, $P12_0$ to $P12_7$, $P13_0$ to $P13_2$, X_{IN}, $\overline{RESET}$, CNV_{SS}		V_I=0V			–4.0	µA
R_{PULLUP}	Pull-up resistance	$P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_5$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_6$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, $P11_0$ to $P11_7$, $P12_0$ to $P12_7$, $P13_0$ to $P13_2$		V_I=0V	66.0	120.0	500.0	kΩ
R_{fXIN}	Feedback resistance	X_{IN}				3.0		MΩ
R_{fXCIN}	Feedback resistance	X_{CIN}				10.0		MΩ
V_{RAM}	RAM retention voltage			When clock is stopped	2.0			V

Note : Has no effect during intermittent pullup operation.

V_{CC} = 3V

Table 1.21.20. Electrical characteristics (referenced to V_{CC} = 3V, V_{SS} = 0V at Ta = 25^{O}C, f(X_{IN}) = 7MH$_{Z}$, with wait)

Symbol	Parameter		Measuring condition		Standard			Unit
					Min.	Typ.	Max.	
Icc	Power supply current	I/o pin is no load applied	Mask ROM, flash memory versions	f(X_{IN})=7MHz Square wave, no division		6.0	15.0	mA
			Mask ROM version	f(X_{CIN})=32kHz Square wave		40.0		µA
			Flash memory version	f(X_{CIN})=32kHz Square wave		150.0		µA
			Mask ROM, flash memory versions	f(X_{CIN})=32kHz When a WAIT instruction is executed Oscillation capacity High (Note)		2.8		µA
				f(X_{CIN})=32kHz When a WAIT instruction is executed Oscillation capacity Low (Note 1)		0.9		µA
				When clock is stopped Ta=25 °C			1.0	µA
				When clock is stopped Ta=85 °C			20.0	
V_{L3}	Supply voltage (V_{L3}) (Note 2)		When charge-pump not used		2.7		6.5	V
V_{L1}	Supply voltage (V_{L1})		When charge-pump used		1.3	1.7	2.1	V
I_{L1}	Power supply current (V_{L1})		V_{L1}=1.7V, f(LCDCK)=200Hz			3.0	6.0	µA

Note 1: With one timer operated using f_{C32}.
Note 2: Rating: V_{L1}=-0.3 V to V_{L2}, V_{L2}=V_{L1} to V_{L3}, V_{L3}=V_{L2} to 6.5V.

Table 1.21.21. A-D conversion characteristics (referenced to V_{CC} = AV_{CC} = V_{REF} = 3V, V_{SS} = AV_{SS} = 0V at Ta = 25^{O}C, f(X_{IN}) = 7MH$_{Z}$, with wait unless otherwise specified)

Symbol	Parameter		Measuring condition	Standard			Unit
				Min.	Typ.	Max.	
–	Resolution		V_{REF} =V_{CC}			10	Bits
–	Absolute accuracy	Sample & hold function not available(8bit)	V_{REF} =V_{CC} = 3V, $ø_{AD}$=f_{AD}/2			±2	LSB
R_{LADDER}	Ladder resistance		V_{REF} =V_{CC}	10		40	kΩ
t_{CONV}	Conversion time(8bit)			14.0			µs
V_{REF}	Reference voltage			2.7		V_{CC}	V
V_{IA}	Analog input voltage			0		V_{REF}	V

Table 1.21.22. D-A conversion characteristics (referenced to V_{CC} = AV_{CC}= V_{REF}= 3V, V_{SS} = AV_{SS} = 0V, at Ta = 25^{O}C, f(X_{IN}) = 7MH$_{Z}$ unless otherwise specified)

Symbol	Parameter	Measuring condition	Standard			Unit
			Min.	Typ.	Max.	
–	Resolution				8	Bits
–	Absolute accuracy				1.0	%
t_{su}	Setup time				3	µs
R_{O}	Output resistance		4	10	20	kΩ
I_{VREF}	Reference power supply input current	(Note)			1.0	mA

Note : This applies when using one D-A converter, with the D-A register for the unused D-A converter set to “00_{16}”. The A-D converter's ladder resistance is not included.

Also, when the Vref is unconnected at the A-D control register, I_{VREF} is sent.

VCC = 3V

Timing requirements (referenced to VCC = 3V, VSS = 0V at Ta = 25°C unless otherwise specified)

Table 1.21.23. External clock input

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t_c	External clock input cycle time	143		ns
$t_{w(H)}$	External clock input HIGH pulse width	60		ns
$t_{w(L)}$	External clock input LOW pulse width	60		ns
t_r	External clock rise time		18	ns
t_f	External clock fall time		18	ns

Table 1.21.24. Timer A input (counter input in event counter mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(TA)}$	TAiIN input cycle time	150		ns
$t_{w(TAH)}$	TAiIN input HIGH pulse width	60		ns
$t_{w(TAL)}$	TAiIN input LOW pulse width	60		ns

Table 1.21.25. Timer A input (gating input in timer mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(TA)}$	TAiIN input cycle time	600		ns
$t_{w(TAH)}$	TAiIN input HIGH pulse width	300		ns
$t_{w(TAL)}$	TAiIN input LOW pulse width	300		ns

Table 1.21.26. Timer A input (external trigger input in one-shot timer mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(TA)}$	TAiIN input cycle time	300		ns
$t_{w(TAH)}$	TAiIN input HIGH pulse width	150		ns
$t_{w(TAL)}$	TAiIN input LOW pulse width	150		ns

Table 1.21.27. Timer A input (external trigger input in pulse width modulation mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{w(TAH)}$	TAiIN input HIGH pulse width	150		ns
$t_{w(TAL)}$	TAiIN input LOW pulse width	150		ns

Table 1.21.28. Timer A input (up/down input in event counter mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(UP)}$	TAiOUT input cycle time	3000		ns
$t_{w(UPH)}$	TAiOUT input HIGH pulse width	1500		ns
$t_{w(UPL)}$	TAiOUT input LOW pulse width	1500		ns
$t_{su(UP-T_{IN})}$	TAiOUT input setup time	600		ns
$t_{h(T_{IN}-UP)}$	TAiOUT input hold time	600		ns

V_{CC} = 3V

Timing requirements (referenced to V_{CC} = 3V, V_{SS} = 0V at Ta = 25°C unless otherwise specified)

Table 1.21.29. 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)	150		ns
$t_{w(TBH)}$	TBi$_{IN}$ input HIGH pulse width (counted on one edge)	60		ns
$t_{w(TBL)}$	TBi$_{IN}$ input LOW pulse width (counted on one edge)	60		ns
$t_{c(TB)}$	TBi$_{IN}$ input cycle time (counted on both edges)	300		ns
$t_{w(TBH)}$	TBi$_{IN}$ input HIGH pulse width (counted on both edges)	160		ns
$t_{w(TBL)}$	TBi$_{IN}$ input LOW pulse width (counted on both edges)	160		ns

Table 1.21.30. Timer B input (pulse period measurement mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(TB)}$	TBi$_{IN}$ input cycle time	600		ns
$t_{w(TBH)}$	TBi$_{IN}$ input HIGH pulse width	300		ns
$t_{w(TBL)}$	TBi$_{IN}$ input LOW pulse width	300		ns

Table 1.21.31. Timer B input (pulse width measurement mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(TB)}$	TBi$_{IN}$ input cycle time	600		ns
$t_{w(TBH)}$	TBi$_{IN}$ input HIGH pulse width	300		ns
$t_{w(TBL)}$	TBi$_{IN}$ input LOW pulse width	300		ns

Table 1.21.32. A-D trigger input

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(AD)}$	$\overline{AD_{TRG}}$ input cycle time (trigger able minimum)	1500		ns
$t_{w(ADL)}$	$\overline{AD_{TRG}}$ input LOW pulse width	200		ns

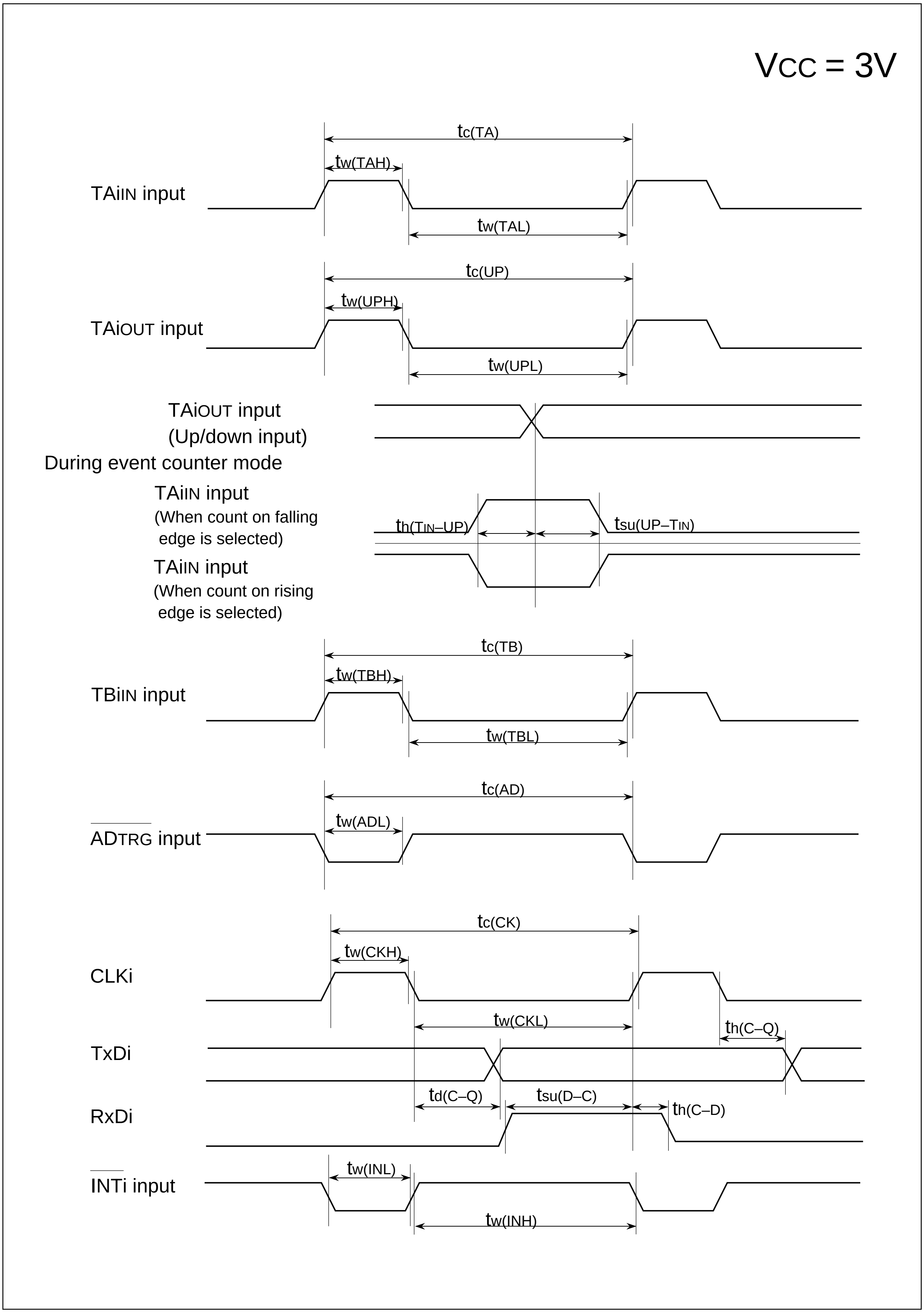
Table 1.21.33. Serial I/O

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(CK)}$	CLKi input cycle time	300		ns
$t_{w(CKH)}$	CLKi input HIGH pulse width	150		ns
$t_{w(CKL)}$	CLKi input LOW pulse width	150		ns
$t_{d(C-Q)}$	TxDi output delay time		160	ns
$t_{h(C-Q)}$	TxDi hold time	0		ns
$t_{su(D-C)}$	RxDi input setup time	50		ns
$t_{h(C-D)}$	RxDi input hold time	90		ns

Table 1.21.34. External interrupt $\overline{INTi}$ inputs

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{w(INH)}$	$\overline{INTi}$ input HIGH pulse width	380		ns
$t_{w(INL)}$	$\overline{INTi}$ input LOW pulse width	380		ns

Timing (Vcc = 3V)



GZZ—SH56—83B <97B0>

Mask ROM number	

MITSUBISHI ELECTRIC SINGLE-CHIP 16-BIT MICROCOMPUTER M30220MA-XXXGP/RP MASK ROM CONFIRMATION FORM

Receipt	Date :	
	Section head signature	Supervisor signature

Note : Please complete all items marked ※.

※	Customer	Company name	TEL ( )	Issuance signature	Submitted by	Supervisor
		Date issued	Date :			

※ 1. Check sheet

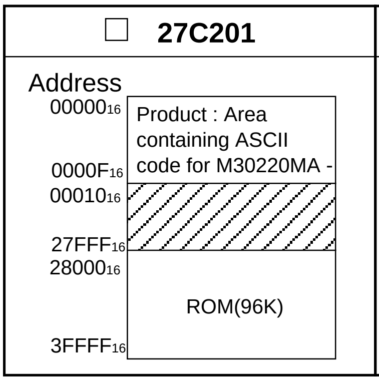
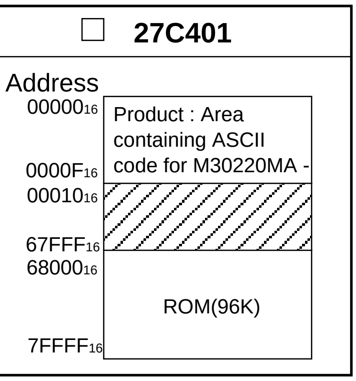
Name the product you order, and choose which to give in, EPROMs or floppy disks.
If you order by means of EPROMs, three sets of EPROMs are required per pattern. If you order by means of floppy disks, one floppy disk is required per pattern.

□ In the case of EPROMs
Mitsubishi will create the mask using the data on the EPROMs supplied, providing the data is the same on at least two of those sets. Mitsubishi will, therefore, only accept liability if there is any discrepancy between the data on the EPROM sets and the ROM data written to the product.
Please carefully check the data on the EPROMs being submitted to Mitsubishi.

Microcomputer type No. : □ M30220MA-XXXGP □ M30220MA-XXXRP

Checksum code for total EPROM area : | | | | | (hex)

EPROM type :

□ 27C201	□ 27C401
 Address 0000016 0000F16 0001016 27FFF16 2800016 3FFFF16 Product : Area containing ASCII code for M30220MA - ROM(96K)	 Address 0000016 0000F16 0001016 67FFF16 6800016 7FFFF16 Product : Area containing ASCII code for M30220MA - ROM(96K)

(1) Write “FF16” to the lined area.
(2) The area from 0000016 to 0000F16 is for storing data on the product type name.
The ASCII code for 'M30220MA-' is shown at right.
The data in this table must be written to address 0000016 to 0000F16.
Both address and data are shown in hex.

Address		Address	
00000_{16}	'M' = $4D_{16}$	00008_{16}	'—' = $2D_{16}$
00001_{16}	'3' = 33_{16}	00009_{16}	FF_{16}
00002_{16}	'0' = 30_{16}	$0000A_{16}$	FF_{16}
00003_{16}	'2' = 32_{16}	$0000B_{16}$	FF_{16}
00004_{16}	'2' = 32_{16}	$0000C_{16}$	FF_{16}
00005_{16}	'0' = 30_{16}	$0000D_{16}$	FF_{16}
00006_{16}	'M' = $4D_{16}$	$0000E_{16}$	FF_{16}
00007_{16}	'A' = 41_{16}	$0000F_{16}$	FF_{16}

GZZ— SH56— 83B <97B0>

Mask ROM number	

MITSUBISHI ELECTRIC SINGLE-CHIP 16-BIT MICROCOMPUTER M30220MA-XXXGP/RP MASK ROM CONFIRMATION FORM

The ASCII code for the type No. can be written to EPROM addresses 00000_{16} to $0000F_{16}$ by specifying the pseudo-instructions for the respective EPROM type shown in the following table at the beginning of the assembler source program.

EPROM type	27C201	27C401
Code entered in source program	△ .SECTION△ ASCIICODE, ROM DATA △ .ORG △ 0C0000H △ .BYTE △ ' M30220MA- '	△ .SECTION△ ASCIICODE, ROM DATA △ .ORG △ 080000H △ .BYTE △ ' M30220MA- '

Note: The ROM cannot be processed if the type No. written to the EPROM does not match the type No. in the check sheet.

□ In the case of floppy disks

Mitsubishi processes the mask files generated by the mask file generation utilities out of those held on the floppy disks you give in to us, and forms them into masks. Hence, we assume liability provided that there is any discrepancy between the contents of these mask files and the ROM data to be burned into products we produce. Check thoroughly the contents of the mask files you give in.
Prepare 3.5 inches 2HD(IBM format) floppy disks. And store only one mask file in a floppy disk.

Microcomputer type No. : □ M30220MA-XXXGP □ M30220MA-XXXRP

File code :									(hex)
Mask file name :									.MSK (alpha-numeric 8-digit)

※2. Mark specification

The mark specification differs according to the type of package. After entering the mark specification on the separate mark specification sheet (for each package), attach that sheet to this masking check sheet for submission to Mitsubishi.
For the M30220MA-XXXRP, submit the 144PFB mark specification sheet. For the M30220MA-XXXGP, submit the 144P6Q mark specification sheet.

※ 3. Usage Conditions

For our reference when of testing our products, please reply to the following questions about the usage of the products you ordered.

(1) Which kind of XIN-XOUT oscillation circuit is used?

□ Ceramic resonator □ Quartz-crystal oscillator
□ External clock input □ Other ( )

What frequency do you use?

f(XIN) = □ MHZ

GZZ— SH56— 83B <97B0>

Mask ROM number	

MITSUBISHI ELECTRIC SINGLE-CHIP 16-BIT MICROCOMPUTER M30220MA-XXXGP/RP MASK ROM CONFIRMATION FORM

(2) Which kind of XCIN-XCOUT oscillation circuit is used?

□ Ceramic resonator　□ Quartz-crystal oscillator

□ External clock input　□ Other (　　　　　　)

What frequency do you use?

f(XCIN) = □ kHZ

(3) Which operating ambient temperature do you use?

□ –10 °C to 75 °C　□ –20 °C to 75 °C　□ –40 °C to 75 °C

□ –10 °C to 85 °C　□ –20 °C to 85 °C　□ –40 °C to 85 °C

(4) Which operating supply voltage do you use?

□ 2.7V to 3.2V　□ 3.2V to 3.7V　□ 3.7V to 4.2V

□ 4.2V to 4.7V　□ 4.7V to 5.2V　□ 5.2V to 5.5V

(5) Do you use I^2C (Inter IC) bus function?

□ Not use　□ Use

(6) Do you use IE (Inter Equipment) bus function?

□ Not use　□ Use

Thank you cooperation.

※ 4. Special item (Indicate none if there is no specified item)

Outline Performance

Table 1.22.1 shows the outline performance of the M30220 (flash memory version).

Table 1.22.1. Outline performance of the M30220 (flash memory version)

Item		Performance
Power supply voltage		V_{CC}=2.7V to 5.5 V (Note 1) V_{CC}=2.7V to 3.6 V (Note 2)
Program/erase voltage		V_{PP}=5.0V ± 10%
Flash memory operation mode		Three modes (parallel I/O, standard serial I/O, CPU rewrite)
Erase block division	User ROM area	See Figure 1.22.1
	Boot ROM area	No division (8 K bytes) (Note 3)
Program method		In units of words
Erase method		Collective erase/block erase
Program/erase control method		Program/erase control by software command
Number of commands		6 commands
Program/erase count		100 times
ROM code protect		Parallel I/O and standard serial I/O modes are supported.

Note 1: Use a 4.5 - 5.5 V power supply voltage when program/erase.
Note 2: Use a 3.0 - 3.6 V power supply voltage when program/erase.
Note 3: 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.

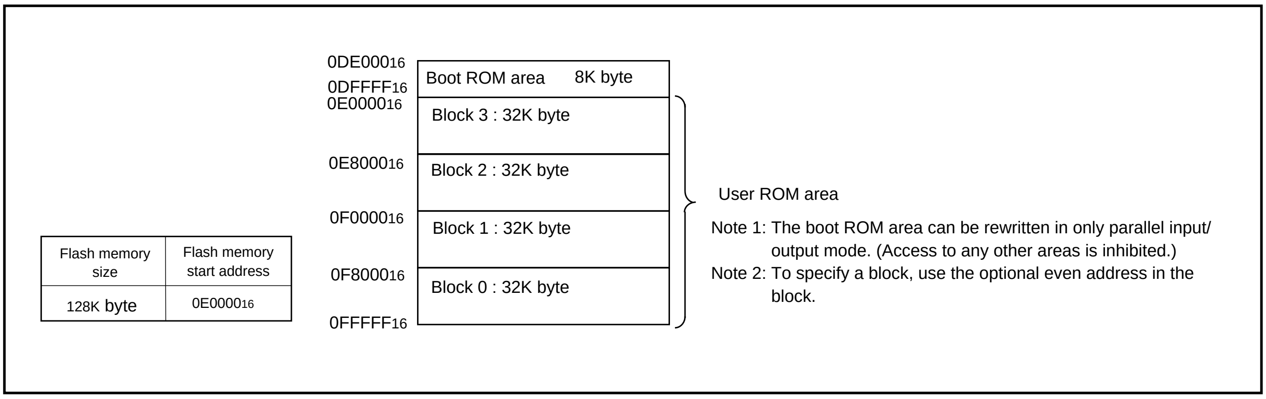
Flash Memory

The M30220 (flash memory version) has an internal new DINOR (DIvided bit line NOR) flash memory that can be rewritten with a single power source when VCC is 4.5 to 5.5 V, and 2 power sources when VCC is 2.7 to 4.5V.

For this flash memory, three flash memory modes are available in which to read, program, and erase: parallel I/O and standard serial I/O modes in which the flash memory can be manipulated using a programmer and a CPU rewrite mode in which the flash memory can be manipulated by the Central Processing Unit (CPU). Each mode is detailed in the pages to follow.

The flash memory is divided into several blocks as shown in Figure 1.22.1, so that memory can be erased one block at a time.

In addition to the ordinary user ROM area to store a microcomputer operation control program, the flash memory has a boot ROM area that is used to store a program to control rewriting in CPU rewrite and standard serial I/O modes. This boot ROM area has had a standard serial I/O mode control program stored in it when shipped from the factory. However, the user can write a rewrite control program in this area that suits the user's application system. This boot ROM area can be rewritten in only parallel I/O mode.



Flash memory size	Flash memory start address
128K byte	$0E0000_{16}$

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 optional even address in the block.

Figure 1.22.1. Block diagram of flash memory version

CPU Rewrite Mode

In CPU rewrite mode, the on-chip flash memory can be operated on (read, program, or erase) under control of the Central Processing Unit (CPU).

In CPU rewrite mode, only the user ROM area shown in Figure 1.22.1 can be rewritten; the boot ROM area cannot be rewritten. Make sure the program and block erase commands are issued for only the user ROM area and each block area.

The control program for CPU rewrite mode can be stored in either user ROM or boot ROM area. In the CPU rewrite mode, because the flash memory cannot be read from the CPU, the rewrite control program must be transferred to any area other than the internal flash memory before it can be executed.

Microcomputer Mode and Boot Mode

The control program for CPU rewrite mode must be written into the user ROM or boot ROM area in parallel I/O mode beforehand. (If the control program is written into the boot ROM area, the standard serial I/O mode becomes unusable.)

See Figure 1.22.1 for details about the boot ROM area.

Normal microcomputer mode is entered when the microcomputer is reset with pulling CNV$_{SS}$ pin low. In this case, the CPU starts operating using the control program in the user ROM area.

When the microcomputer is reset by pulling the P7$_4$ ($\overline{CE}$) pin high, the CNV$_{SS}$ pin high, the CPU starts operating using the control program in the boot ROM area (program start address is DE000$_{16}$ fixation). This mode is called the “boot” mode.

Block Address

Block addresses refer to the optional even address of each block. These addresses are used in the block erase command.

Outline Performance (CPU Rewrite Mode)

In the CPU rewrite mode, the CPU erases, programs and reads the internal flash memory as instructed by software commands. This rewrite control program must be transferred to internal RAM before it can be excuted.

The CPU rewrite mode is accessed by applying 5V ± 10% to the CNVSS pin and writing “1” for the CPU rewrite mode select bit (bit 1 in address $03B4_{16}$). Software commands are accepted once the mode is accessed.

In the CPU rewrite mode, write to and read from software commands and data into even-numbered address (“0” for byte address A0) in 16-bit units. Always write 8-bit software commands into even-numbered address. Commands are ignored with odd-numbered addresses.

Use software commands to control program and erase operations. Whether a program or erase operation has terminated normally or in error can be verified by reading the status register.

Figure 1.23.1 shows the flash memory control register.

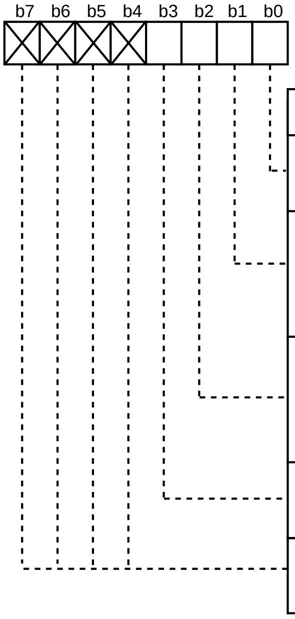
Bit 0 is the RY/$\overline{\text{BY}}$ status flag used exclusively to read the operating status of the flash memory. During programming and erase operations, it is “0”. Otherwise, it is “1”.

Bit 1 is the CPU rewrite mode select bit. When this bit is set to “1” and 5V ± 10% are applied to the CNVSS pin, the M30220 accesses the CPU rewrite mode. Software commands are accepted once the mode is accessed. In CPU rewrite mode, the CPU becomes unable to access the internal flash memory directly. Therefore, use the control program in RAM for write to bit 1. To set this bit to “1”, it is necessary to write “0” and then write “1” in succession. The bit can be set to “0” by only writing a “0” .

Bit 2 is the CPU rewrite mode entry flag. This bit can be read to check whether the CPU rewrite mode has been entered or not.

Bit 3 is the flash memory reset bit used to reset the control circuit of the internal flash memory. This bit is used when exiting CPU rewrite mode and when flash memory access has failed. When the CPU rewrite mode select bit is “1”, writing “1” for this bit resets the control circuit. To release the reset, it is necessary to set this bit to “0”. If the control circuit is reset while erasing is in progress, a 5 ms wait is needed so that the flash memory can restore normal operation. Figure 1.23.2 shows a flowchart for setting/releasing the CPU rewrite mode.

Flash memory control register



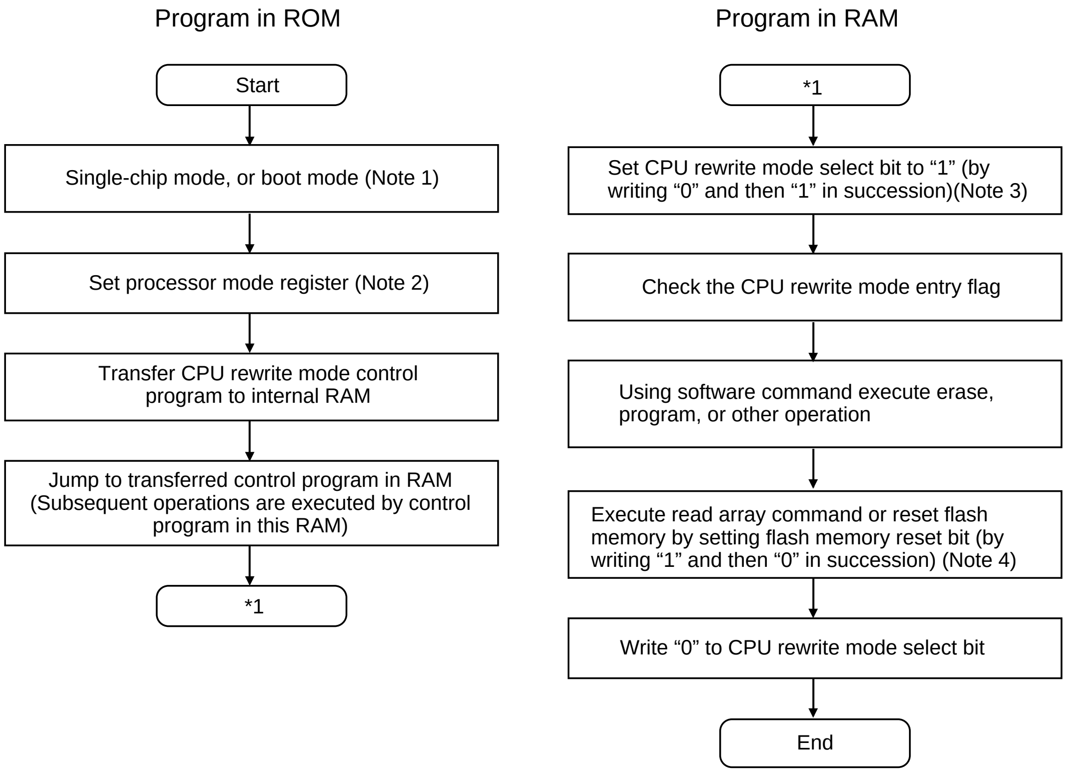
Symbol	Address	When reset
FMCR	$03B4_{16}$	$XXXX0001_2$

Bit symbol	Bit name	Function	R	W
FMCR0	RY/$\overline{BY}$ status flag	0: Busy (being written or erased) 1: Ready	○	—
FMCR1	CPU rewrite mode select bit (Note 1)	0: Normal mode (Software commands invalid) 1: CPU rewrite mode	○	○
FMCR2	CPU rewrite mode entry flag	0: Normal mode (Software commands invalid) 1: CPU rewrite mode (Software commands acceptable)	○	—
FMCR3	Flash memory reset bit (Note 2)	0: Normal operation 1: Reset	○	○
Nothing is assigned. When write, set "0". When read, values are indeterminate.			—	—

Note 1: For this bit to be set to “1”, the user needs to write a “0” and then a “1” to it in succession. When it is not this procedure, it is not enacted in “1”. This is necessary to ensure that no interrupt or DMA transfer will be executed during the interval. Use the control program in the RAM for write to this bit.

Note 2: Effective only when the CPU rewrite mode select bit = 1. Set this bit to 0 subsequently after setting it to 1 (reset).

Figure 1.23.1. Flash memory control registers


Program in ROM
Start
Single-chip mode, or boot mode (Note 1)
Set processor mode register (Note 2)
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)
*1
Program in RAM
*1
Set CPU rewrite mode select bit to “1” (by writing “0” and then “1” in succession)(Note 3)
Check the CPU rewrite mode entry flag
Using software command execute erase, program, or other operation
Execute read array command or reset flash memory by setting flash memory reset bit (by writing “1” and then “0” in succession) (Note 4)
Write “0” to CPU rewrite mode select bit
End

Note 1: Apply 5V ± 10 % to CNVSS pin by confirmation of CPU rewrite mode entry flag when started operation with single-chip mode.
Note 2: During CPU rewrite mode, set the main clock frequency as shown below using the main clock divide ratio select bit (bit 6 at address 0006$_{16}$ and bits 6 and 7 at address 0007$_{16}$):
5.0 MHz or less when wait bit (bit 7 at address 0005$_{16}$) = “0” (without internal access wait state)
10.0 MHz or less when wait bit (bit 7 at address 0005$_{16}$) = “1” (with internal access wait state)
Note 3: For CPU rewrite mode select bit to be set to “1”, the user needs to write a “0” and then a “1” to it in succession. When it is not this procedure, it is not enacted in “1”. This is necessary to ensure that no interrupt or DMA transfer will be executed during the interval.
Note 4: 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.

Figure 1.23.2. CPU rewrite mode set/reset flowchart

Precautions on CPU Rewrite Mode

Described below are the precautions to be observed when rewriting the flash memory in CPU rewrite mode.

(1) Operation speed

During CPU rewrite mode, set the main clock frequency as shown below using the main clock divide ratio select bit (bit 6 at address 0006_{16} and bits 6 and 7 at address 0007_{16}):
5.0 MHz or less when wait bit (bit 7 at address 0005_{16}) = 0 (without internal access wait state)
10.0 MHz or less when wait bit (bit 7 at address 0005_{16}) = 1 (with internal access wait state)

(2) Instructions inhibited against use

The instructions listed below cannot be used during CPU rewrite mode because they refer to the internal data of the flash memory:
UND instruction, INTO instruction, JMPS instruction, JSRS instruction, and BRK instruction

(3) Interrupts inhibited against use

The $\overline{\text{NMI}}$, address match, and watchdog timer interrupts cannot be used during CPU rewrite mode because they refer to the internal data of the flash memory. If interrupts have their vector in the variable vector table, they can be used by transferring the vector into the RAM area.

(4) Reset

If the control circuit is reset while erasing is in progress, a 5 ms wait is needed so that the flash memory can restore normal operation. Set a 5 ms wait to release the reset operation.
Also, when the reset has been released, the program execute start address is automatically set to $0DE000_{16}$, therefore program so that the execute start address of the boot ROM is $0DE000_{16}$.

(5) Writing in the user ROM area

If power is lost while rewriting blocks that contain the flash rewrite program with the CPU rewrite mode, those blocks may not be correctly rewritten and it is possible that the flash memory can no longer be rewritten after that. Therefore, it is recommended to use the standard serial I/O mode or parallel I/O mode to rewrite these blocks.

Software Commands

Table 1.23.1 lists the software commands available with the M30220 (flash memory version).
After setting the CPU rewrite mode select bit to 1, write a software command to specify an erase or program operation. Note that when entering a software command, the upper byte (D_8 to D_{15}) is ignored.
The content of each software command is explained below.

Table 1.23.1. List of software commands (CPU rewrite mode)

Command	Cycle number	First bus cycle			Second bus cycle		
		Mode	Address	Data (D_0 to D_7)	Mode	Address	Data (D_0 to D_7)
Read array	1	Write	X (Note 5)	FF$_{16}$			
Read status register	2	Write	X	70$_{16}$	Read	X	SRD (Note 2)
Clear status register	1	Write	X	50$_{16}$			
Program (Note 3)	2	Write	X	40$_{16}$	Write	WA (Note 3)	WD (Note 3)
Erase all block	2	Write	X	20$_{16}$	Write	X	20$_{16}$
Block erase	2	Write	X	20$_{16}$	Write	BA (Note 4)	D0$_{16}$

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
Note 4: BA = Block Address (Enter the optional address of each block that is an even address.)
Note 5: X denotes a given address in the user ROM area (that is an even address).

Read Array Command (FF$_{16}$)

The read array mode is entered by writing the command code "FF$_{16}$" in the first bus cycle. When an even address to be read is input in one of the bus cycles that follow, the content of the specified address is read out at the data bus (D_0–D_{15}), 16 bits at a time.
The read array mode is retained intact until another command is written.

Read Status Register Command (70$_{16}$)

When the command code "70$_{16}$" is written in the first bus cycle, the content of the status register is read out at the data bus (D_0–D_7) by a read in the second bus cycle.
The status register is explained in the next section.

Clear Status Register Command (50$_{16}$)

This command is used to clear the bits SR4 to SR5 of the status register after they have been set. These bits indicate that operation has ended in an error. To use this command, write the command code "50$_{16}$" in the first bus cycle.

Program Command (40_{16})

Program operation starts when the command code “40_{16}” is written in the first bus cycle. Then, if the address and data to program are written in the 2nd bus cycle, program operation (data programming and verification) will start.

Whether the write operation is completed can be confirmed by reading the status register or the RY/$\overline{\text{BY}}$ status flag. When the program starts, the read status register mode is accessed automatically and the content of the status register is read into the data bus (D0 - D7). The status register bit 7 (SR7) is set to 0 at the same time the write operation starts and is returned to 1 upon completion of the write operation. In this case, the read status register mode remains active until the Read Array command (FF_{16}) is written.

The RY/$\overline{\text{BY}}$ status flag is 0 during write operation and 1 when the write operation is completed as is the status register bit 7.

At program end, program results can be checked by reading the status register.

Erase All Blocks Command (20_{16}/20_{16})

By writing the command code “20_{16}” in the first bus cycle and the confirmation command code “20_{16}” in the second bus cycle that follows, the system starts erase all blocks(erase and erase verify).

Whether the erase all blocks command is terminated can be confirmed by reading the status register or the RY/$\overline{\text{BY}}$ status flag. When the erase all blocks operation starts, the read status register mode is accessed automatically and the content of the status register can be read out. The status register bit 7 (SR7) is set to 0 at the same time the erase operation starts and is returned to 1 upon completion of the erase operation. In this case, the read status register mode remains active until the Read Array command (FF_{16}) is written.

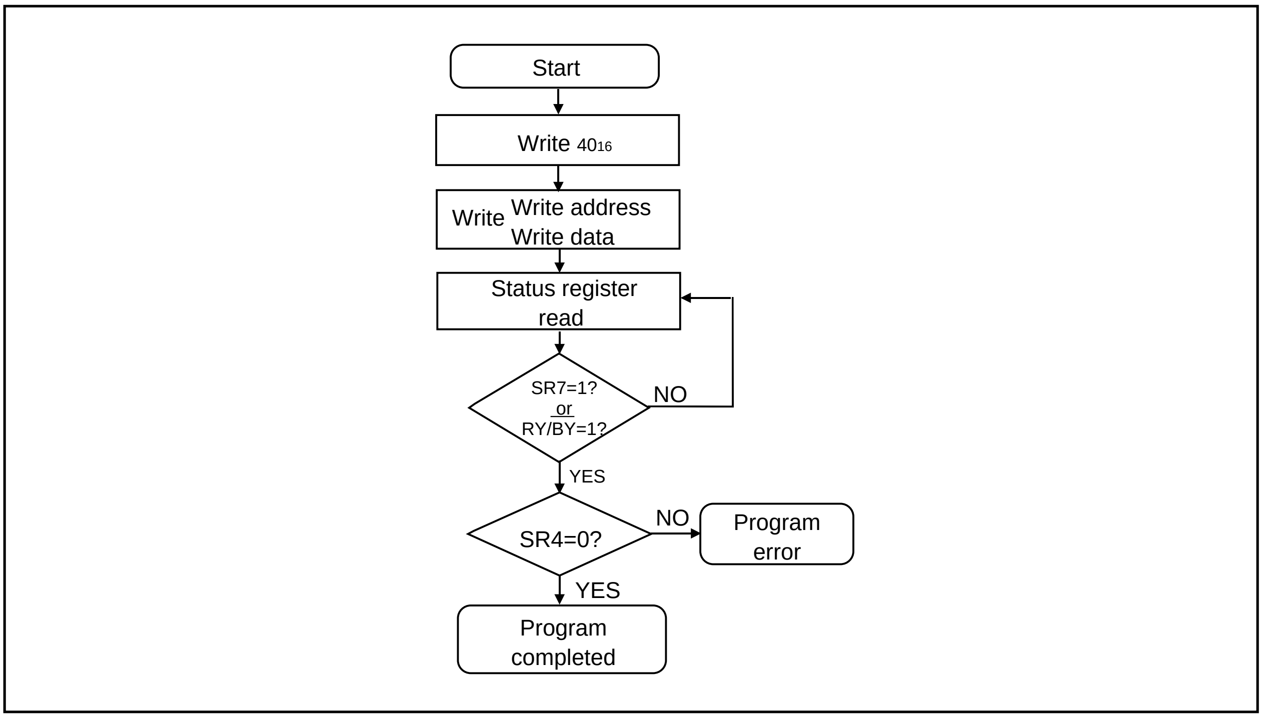


Figure 1.23.3. Program flowchart

The RY/$\overline{\text{BY}}$ status flag is 0 during erase operation and 1 when the erase operation is completed as is the status register bit 7.

At erase all blocks end, erase results can be checked by reading the status register. For details, refer to the section where the status register is detailed.

Block Erase Command (20_{16}/$D0_{16}$)

By writing the command code "20_{16}" in the first bus cycle and the confirmation command code "$D0_{16}$" in the second bus cycle that follows to the block address of a flash memory block, the system initiates a block erase (erase and erase verify) operation.

Whether the block erase operation is completed can be confirmed by reading the status register or the RY/$\overline{\text{BY}}$ status flag. At the same time the block erase operation starts, the read status register mode is automatically entered, so the content of the status register can be read out. The status register bit 7 (SR7) is set to 0 at the same time the block erase operation starts and is returned to 1 upon completion of the block erase operation. In this case, the read status register mode remains active until the Read Array command (FF_{16}).

The RY/$\overline{\text{BY}}$ status flag is 0 during block erase operation and 1 when the block erase operation is completed as is the status register bit 7.

After the block erase operation is completed, the status register can be read out to know the result of the block erase operation. For details, refer to the section where the status register is detailed.

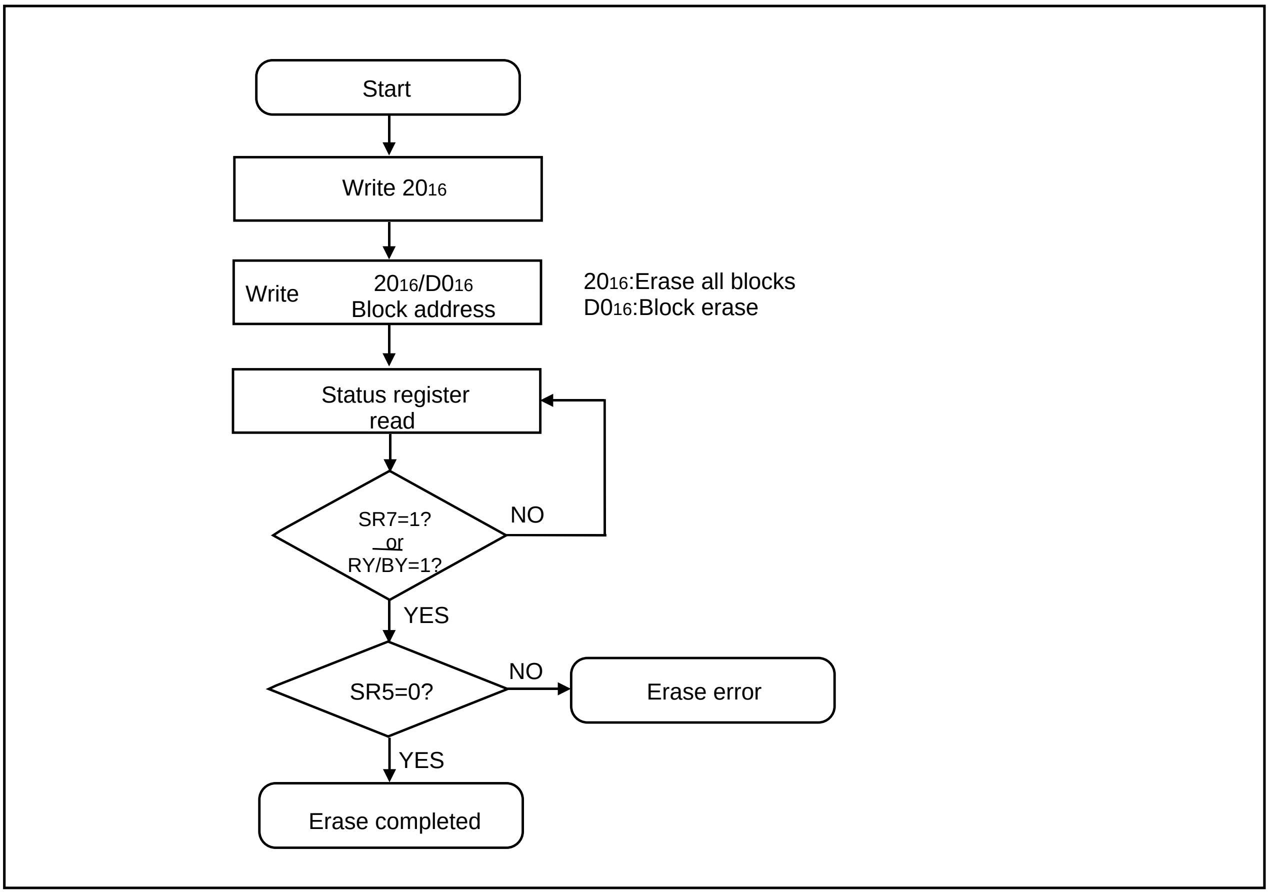


Figure 1.23.4. Erase flowchart

Status Register

The status register shows the operating state of the flash memory and whether erase operations and programs ended successfully or in error. It can be read in the following ways.

(1) By reading an arbitrary address from the user ROM area after writing the read status register command (70_{16})

(2) By reading an arbitrary address from the user ROM area in the period from when the program starts or erase operation starts to when the read array command (FF_{16}) is input

Table 1.23.2 shows the status register.
Also, the status register can be cleared in the following way.
(1) 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.

Sequencer status (SR7)

After power-on, the sequencer status is set to 1(ready).
The sequencer status indicates the operating status of the device. This status bit is set to 0 (busy) during write or erase operation and is set to 1 upon completion of these operations.

Erase status (SR5)

The erase status informs the operating status of erase operation to the CPU. When an erase error occurs, it is set to 1.
The erase status is reset to 0 when cleared.

Program status (SR4)

The program status informs the operating status of write operation to the CPU. When a write error occurs, it is set to 1.
The program status is reset to 0 when cleared.

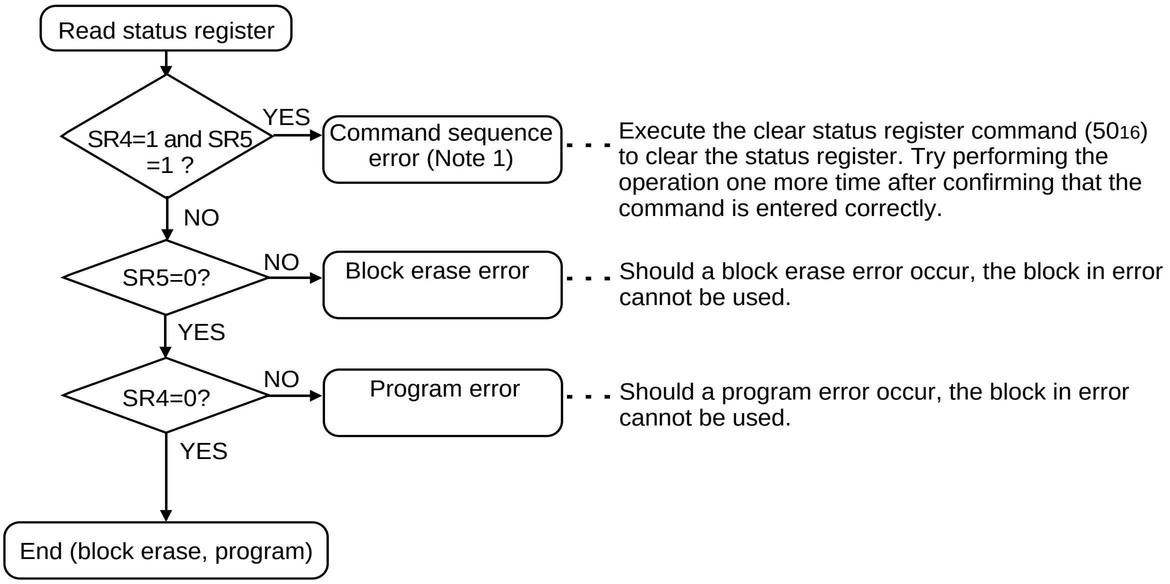
If “1” is written for any of the SR5 or SR4 bits, the program, erase all blocks, and block erase commands are not accepted. Before executing these commands, execute the clear status register command (50_{16}) and clear the status register.
Also, any commands are not correct, both SR5 and SR4 are set to 1.

Table 1.23.2. Definition of each bit in status register

Each bit of SRD	Status name	Definition	
		"1"	"0"
SR7 (bit7)	Sequencer 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)	Reserved	-	-
SR2 (bit2)	Reserved	-	-
SR1 (bit1)	Reserved	-	-
SR0 (bit0)	Reserved	-	-

Full Status Check

By performing full status check, it is possible to know the execution results of erase and program operations. Figure 1.23.5 shows a full status check flowchart and the action to be taken when each error occurs.



Execute the clear status register command (50₁₆) 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.

Should a program error occur, the block in error cannot be used.

Note 1: Either SR5 or SR4 may become "0". Take care about it during program debugging.
Note 2: When one of SR5 to SR4 is set to 1, none of the program, erase all blocks, and block erase commands is accepted. Execute the clear status register command (50_{16}) before executing these commands.

Figure 1.23.5. Full status check flowchart and remedial procedure for errors

Functions To Inhibit Rewriting Flash Memory Version

To prevent the contents of the flash memory version from being read out or rewritten easily, the device incorporates a ROM code protect function for use in parallel I/O mode and an ID code check function for use in standard serial I/O mode.

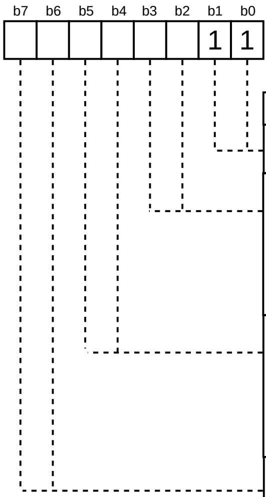
ROM code protect function

The ROM code protect function is used to prohibit reading out or modifying the contents of the flash memory during parallel I/O mode and is set by using the ROM code protect control address register ($0FFFFF_{16}$). Figure 1.23.6 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 flash memory version are protected against readout and modification. ROM code protect is implemented in two levels. If level 2 is selected, the flash memory is protected even against readout by a shipment inspection LSI tester, etc. When an attempt is made to select both level 1 and level 2, level 2 is selected by default.

If both of the two ROM code protect reset bits are set to “00,” ROM code protect is turned off, so that the contents of the flash memory version can be read out or modified. Once ROM code protect is 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.

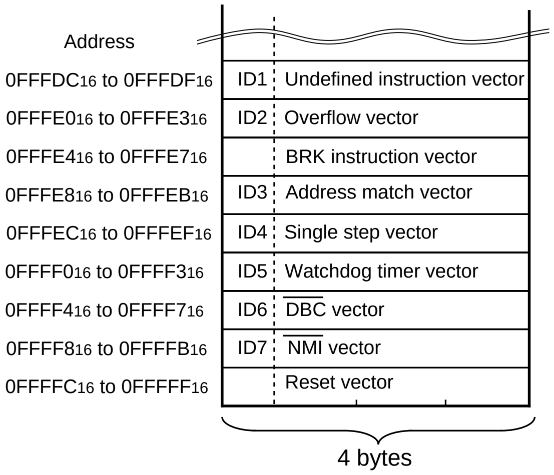
Note 2: When ROM code protect level 2 is turned on, ROM code readout by a shipment inspection LSI tester, etc. also is inhibited.

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 1.23.6. 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 are not blank, the ID code sent from the peripheral unit is compared with the ID code written in the flash memory to see if they match. If the ID codes do not match, the commands sent from the peripheral unit are not accepted. The ID code consists of 8-bit data, the areas of which, beginning with the first byte, are $0FFFDF_{16}$, $0FFFE3_{16}$, $0FFFEB_{16}$, $0FFFEF_{16}$, $0FFFF3_{16}$, $0FFFF7_{16}$, and $0FFFFB_{16}$. Write a program which has had the ID code preset at these addresses to the flash memory.



Address		
$0FFFDC_{16}$ to $0FFFDF_{16}$	ID1	Undefined instruction vector
$0FFFE0_{16}$ to $0FFFE3_{16}$	ID2	Overflow vector
$0FFFE4_{16}$ to $0FFFE7_{16}$		BRK instruction vector
$0FFFE8_{16}$ to $0FFFEB_{16}$	ID3	Address match vector
$0FFFEC_{16}$ to $0FFFEF_{16}$	ID4	Single step vector
$0FFFF0_{16}$ to $0FFFF3_{16}$	ID5	Watchdog timer vector
$0FFFF4_{16}$ to $0FFFF7_{16}$	ID6	$\overline{\text{DBC}}$ vector
$0FFFF8_{16}$ to $0FFFFB_{16}$	ID7	$\overline{\text{NMI}}$ vector
$0FFFFC_{16}$ to $0FFFFF_{16}$		Reset vector

Figure 1.23.7. ID code store addresses

Parallel I/O Mode

The parallel I/O mode inputs and outputs the software commands, addresses and data needed to operate (read, program, erase, etc.) the internal flash memory. This I/O is parallel.
Use an exclusive programer supporting M30220 (flash memory version).
Refer to the instruction manual of each programer maker for the details of use.

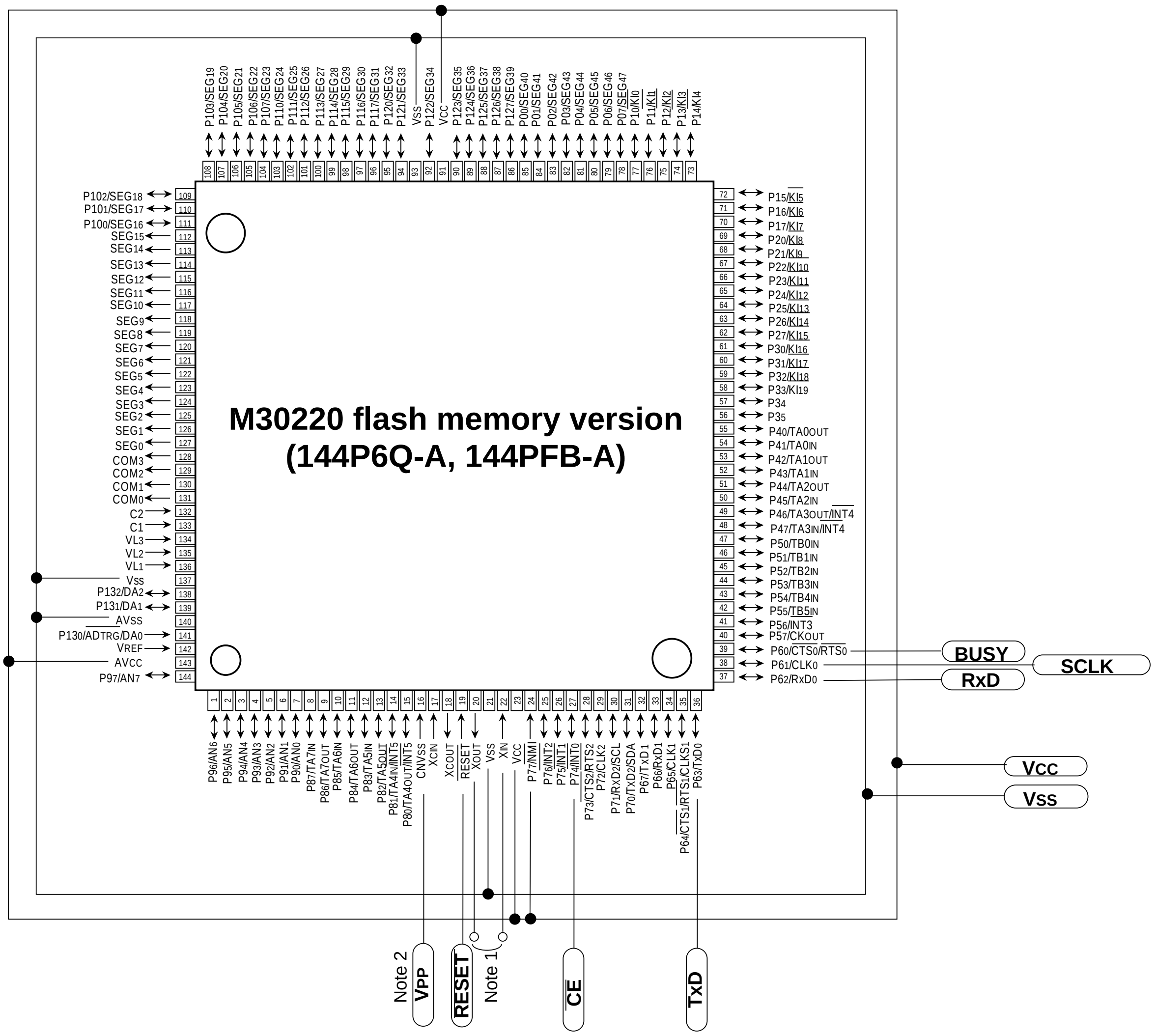
User ROM and Boot ROM Areas

In parallel I/O mode, the user ROM and boot ROM areas shown in Figure 1.22.1 can be rewritten. Both areas of flash memory can be operated on in the same way.
Program and block erase operations can be performed in the user ROM area. The user ROM area and its blocks are shown in Figure 1.22.1.
The boot ROM area is 8 Kbytes in size. In parallel I/O mode, it is located at addresses $0DE000_{16}$ through $0DFFFF_{16}$. Make sure program and block erase operations are always performed within this address range. (Access to any location outside this address range is prohibited.)
In the boot ROM area, an erase block operation is applied to only one 8 Kbyte block. The boot ROM area has had a standard serial I/O mode control program stored in it when shipped from the Mitsubishi factory. Therefore, using the device in standard serial input/output mode, you do not need to write to the boot ROM area.

Pin functions (Flash memory standard serial I/O mode) (Note 1)

Pin	Name	I/O	Description
V_{CC},V_{SS}	Power input		Apply a 3.0 - 3.6 V (Note 2) or 4.5 - 5.5 V (Note 3) voltage on the Vcc pin, and 0 V voltage on the Vss pin.
CNV_{SS}	CNV_{SS}	I	Connect to V_{CC} when V_{CC} = 4.5V to 5.5 V. Connect to Vpp (=4.5 V to 5.5 V) when V_{CC} = 3.0V to 3.6 V.
$\overline{RESET}$	Reset input	I	Reset input pin. While reset is "L" level, a 20 cycle or longer clock must be input to X_{IN} pin.
X_{IN}	Clock input	I	Connect a ceramic resonator or crystal oscillator between X_{IN} and X_{OUT} pins. To input an externally generated clock, input it to X_{IN} pin and open X_{OUT} pin.
X_{OUT}	Clock output	O	
X_{CIN}	Sub-clock input	I	Connect a crystal oscillator between X_{CIN} and X_{COUT} pins. To input an externally generated clock, input it to X_{CIN} pin and open X_{COUT} pin.
X_{COUT}	Sub-clock output	O	
AV_{CC}, AV_{SS}	Analog power supply input		Connect AV_{SS} to V_{SS} and AV_{CC} to V_{CC}, respectively.
V_{REF}	Reference voltage input	I	Enter the reference voltage for AD from this pin.
$P0_0$ to $P0_7$	Input port P0	I	Input "H" or "L" level signal or open.
$P1_0$ to $P1_7$	Input port P1	I	Input "H" or "L" level signal or open.
$P2_0$ to $P2_7$	Input port P2	I	Input "H" or "L" level signal or open.
$P3_0$ to $P3_5$	Input port P3	I	Input "H" or "L" level signal or open.
$P4_0$ to $P4_7$	Input port P4	I	Input "H" or "L" level signal or open.
$P5_0$ to $P5_7$	Input port P5	I	Input "H" or "L" level signal or open.
$P6_0$	BUSY output	O	Standard serial mode 1: BUSY signal output pin Standard serial mode 2: Monitors the program operation check
$P6_1$	SCLK input	I	Standard serial mode 1: Serial clock input pin Standard serial mode 2: Input "L".
$P6_2$	RxD input	I	Serial data input pin
$P6_3$	TxD output	O	Serial data output pin
$P6_4$ to $P6_7$	Input port P6	I	Input "H" or "L" level signal or open.
$P7_0$ to $P7_3$, $P7_5$, $P7_6$	Input port P7	I	Input "H" or "L" level signal or open.
$P7_4$	$\overline{CE}$ input	I	Input "H" level signal.
$P7_7$	$\overline{NMI}$ input	I	Connect this pin to Vcc.
$P8_0$ to $P8_7$	Input port P8	I	Input "H" or "L" level signal or open.
$P9_0$ to $P9_7$	Input port P9	I	Input "H" or "L" level signal or open.
$P10_0$ to $P10_7$	Input port P10	I	Input "H" or "L" level signal or open.
$P11_0$ to $P11_7$	Input port P11	I	Input "H" or "L" level signal or open.
$P12_0$ to $P12_7$	Input port P12	I	Input "H" or "L" level signal or open.
$P13_0$ to $P13_2$	Input port P13	I	Input "H" or "L" level signal or open.
SEG_0 to SEG_{15}	Segment output	O	Open when not used LCD control circuit.
COM_0 to COM_3	Common output	O	Open when not used LCD control circuit.
V_{L3} to V_{L1}	Power supply input for LCD		Input LCD power source. However, do not input the power when the power supply voltages for V_{CC} and LCD are different.
C_1 to C_2	Charge-pump capacitor pin		Connect a condenser between C_1 and C_2 when used LCD charge-pump.

Note 1: About the unused pins, see the example of processing for unused pins in the single-chip mode.
Note 2: The power supply voltage is V_{CC}=2.7 - 3.6 V in the single-chip mode.
Note 3: The power supply voltage is V_{CC}=2.7 - 5.5 V in the single-chip mode.



Mode setup method

Signal	Value
CNVss	**4.5 to 5.5V**
RESET	**Vss → Vcc**
CE	**Vcc**

Note 1: Connect oscillator circuit.
Note 2: Connect to VCC when VCC = 4.5V to 5.5 V.
Connect to Vpp (=4.5V to 5.5 V) when VCC = 3.0V to 3.6 V.

Figure 1.25.1. Pin connections for serial I/O mode (1)

Standard serial I/O mode

The standard serial I/O mode inputs and outputs the software commands, addresses and data needed to operate (read, program, erase, etc.) the internal flash memory. This I/O is serial. There are actually two standard serial I/O modes: mode 1, which is clock synchronized, and mode 2, which is asynchronized. Both modes require a purpose-specific peripheral unit.

The standard serial I/O mode is different from the parallel I/O mode in that the CPU controls flash memory rewrite (uses the CPU's rewrite mode), rewrite data input and so forth. The standard serial I/O mode is started by connecting “H” to the $P7_4$ ($\overline{CE}$) pin and “H” to the CNV_{SS} pin (when V_{CC} = 4.5 V to 5.5 V, connect to V_{CC}; when V_{CC} = 2.7 V to 4.5 V, supply 4.5 V to 5.5 V to Vpp from an external source), and releasing the reset operation. (In the ordinary command mode, set CNVss pin to "L" level.)
This control program is written in the boot ROM area when the product is shipped from Mitsubishi. Accordingly, make note of the fact that the standard serial I/O mode cannot be used if the boot ROM area is rewritten in the parallel I/O mode. Figure 1.25.1 shows the pin connections for the standard serial I/O mode. Serial data I/O uses UART0 and transfers the data serially in 8-bit units. Standard serial I/O switches between mode 1 (clock synchronized) and mode 2 (clock asynchronized) according to the level of CLK_0 pin when the reset is released.

To use standard serial I/O mode 1 (clock synchronized), set the CLK_0 pin to "H" level and release the reset. The operation uses the four UART0 pins CLK_0, RxD_0, TxD_0 and RTS_0 (BUSY). The CLK_0 pin is the transfer clock input pin through which an external transfer clock is input. The TxD_0 pin is for CMOS output. The RTS_0 (BUSY) pin outputs an "L" level when ready for reception and an "H" level when reception starts.

To use standard serial I/O mode 2 (clock asynchronized), set the CLK_0 pin to "L" level and release the reset. The operation uses the two UART0 pins RxD_0 and TxD_0.

In the standard serial I/O mode, only the user ROM area indicated in Figure 1.22.1 can be rewritten. The boot ROM cannot.
In the standard serial I/O mode, a 7-byte ID code is used. When there is data in the flash memory, commands sent from the peripheral unit (programmer) are not accepted unless the ID code matches.

Overview of standard serial I/O mode 1 (clock synchronized)

In standard serial I/O mode 1, software commands, addresses and data are input and output between the MCU and peripheral units (serial programer, etc.) using 4-wire clock-synchronized serial I/O (UART0).
Standard serial I/O mode 1 is engaged by releasing the reset with the $P6_1$ (CLK_0) pin "H" level.
In reception, software commands, addresses and program data are synchronized with the rise of the transfer clock that is input to the CLK_0 pin, and are then input to the MCU via the RxD_0 pin. In transmission, the read data and status are synchronized with the fall of the transfer clock, and output from the TxD_0 pin.
The TxD_0 pin is for CMOS output. Transfer is in 8-bit units with LSB first.
When busy, such as during transmission, reception, erasing or program execution, the RTS_0 (BUSY) pin is "H" level. Accordingly, always start the next transfer after the RTS_0 (BUSY) pin is "L" level.
Also, data and status registers in memory can be read after inputting software commands. Status, such as the operating state of the flash memory or whether a program or erase operation ended successfully or not, can be checked by reading the status register. Here following are explained software commands, status registers, etc.

Software Commands

Table 1.25.1 lists software commands. In the standard serial I/O mode 1, erase operations, programs and reading are controlled by transferring software commands via the RxD$_0$ pin. Software commands are explained here below.

Table 1.25.1. Software commands (Standard serial I/O mode 1)

	Control command	1st byte transfer	2nd byte	3rd byte	4th byte	5th byte	6th byte		When ID is not verified
1	Page read	FF$_{16}$	Address (middle)	Address (high)	Data output	Data output	Data output	Data output to 259th byte	Not acceptable
2	Page program	41$_{16}$	Address (middle)	Address (high)	Data input	Data input	Data input	Data input to 259th byte	Not acceptable
3	Block erase	20$_{16}$	Address (middle)	Address (high)	D0$_{16}$				Not acceptable
4	Erase all 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	ID check function	F5$_{16}$	Address (low)	Address (middle)	Address (high)	ID size	ID1	To ID7	Acceptable
8	Download function	FA$_{16}$	Size (low)	Size (high)	Check-sum	Data input	To required number of times		Not acceptable
9	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
10	Boot ROM area output function	FC$_{16}$	Address (middle)	Address (high)	Data output	Data output	Data output	Data output to 259th byte	Not acceptable
11	Read check data	FD$_{16}$	Check data (low)	Check data (high)					Not acceptable

Note 1: Shading indicates transfer from flash memory microcomputer to peripheral unit. All other data is transferred from the peripheral unit to the flash memory microcomputer.

Note 2: SRD refers to status register data. SRD1 refers to status register 1 data.

Note 3: All commands can be accepted when the flash memory is totally blank.

Page Read Command

This command reads the specified page (256 bytes) in the flash memory sequentially one byte at a time. Execute the page read command as explained here following.

(1) Transfer the “FF_{16}” command code with the 1st byte.

(2) Transfer addresses A_8 to A_{15} and A_{16} to A_{23} with the 2nd and 3rd bytes respectively.

(3) From the 4th byte onward, data (D_0–D_7) for the page (256 bytes) specified with addresses A_8 to A_{23} will be output sequentially from the smallest address first in sync with the fall of the clock.

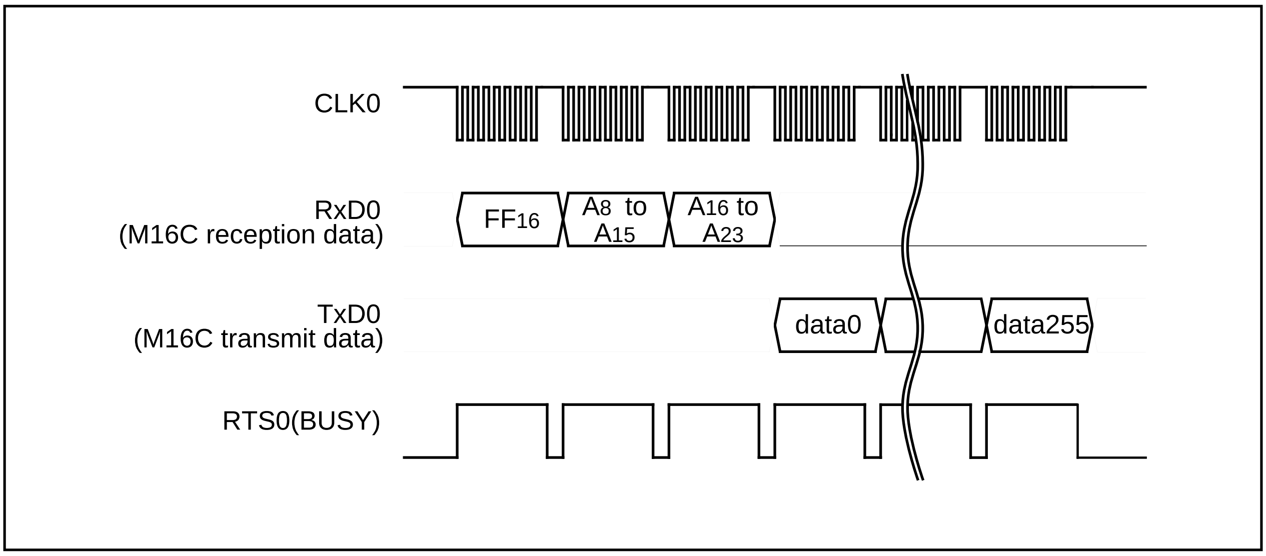


Figure 1.25.2. Timing for page read

Read Status Register Command

This command reads status information. When the “70_{16}” command code is sent with the 1st byte, the contents of the status register (SRD) specified with the 2nd byte and the contents of status register 1 (SRD1) specified with the 3rd byte are read.

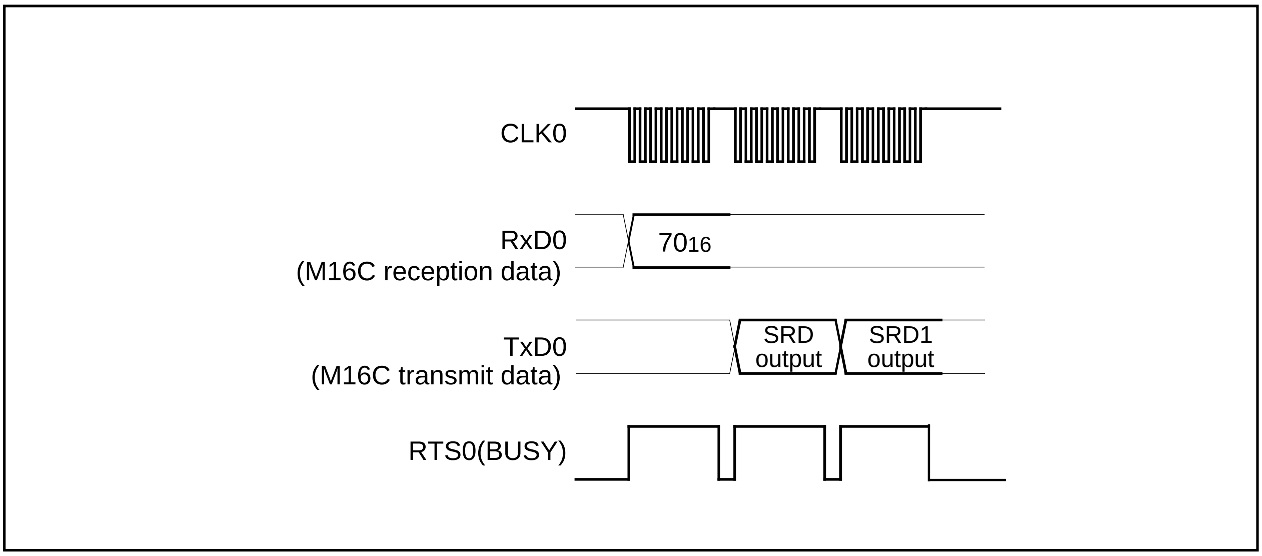


Figure 1.25.3. Timing for reading the status register

Clear Status Register Command

This command clears the bits (SR4–SR5) which are set when the status register operation ends in error. When the “50_{16}” command code is sent with the 1st byte, the aforementioned bits are cleared. When the clear status register operation ends, the RTS_0 (BUSY) signal changes from the “H” to the “L” level.

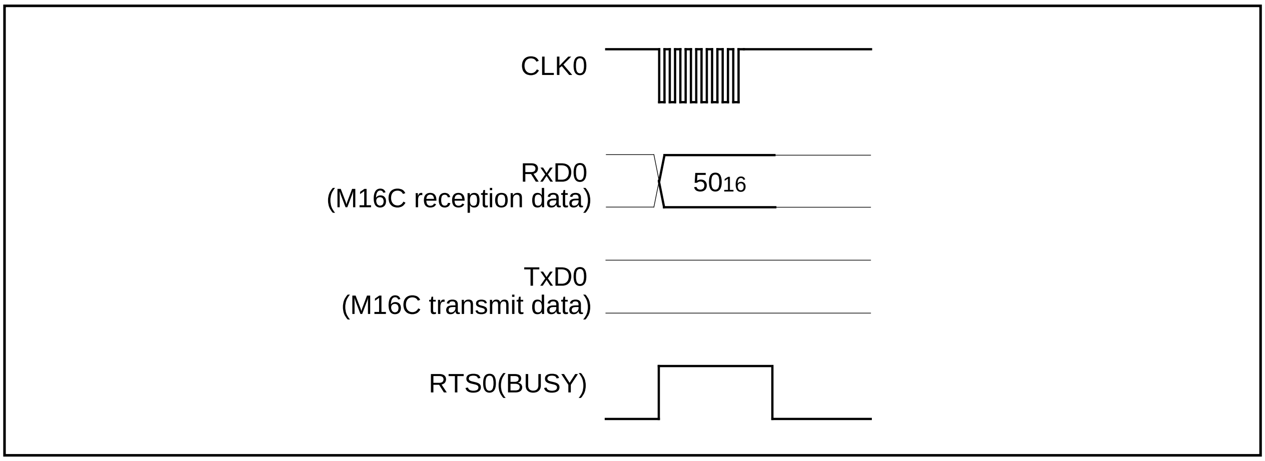


Figure 1.25.4. Timing for clearing the status register

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 here following.

(1) Transfer the “41_{16}” command code with the 1st byte.

(2) Transfer addresses A_8 to A_{15} and A_{16} to A_{23} with the 2nd and 3rd bytes respectively.

(3) From the 4th byte onward, as write data (D_0–D_7) for the page (256 bytes) specified with addresses A_8 to A_{23} is input sequentially from the smallest address first, that page is automatically written.

When reception setup for the next 256 bytes ends, the RTS_0 (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.

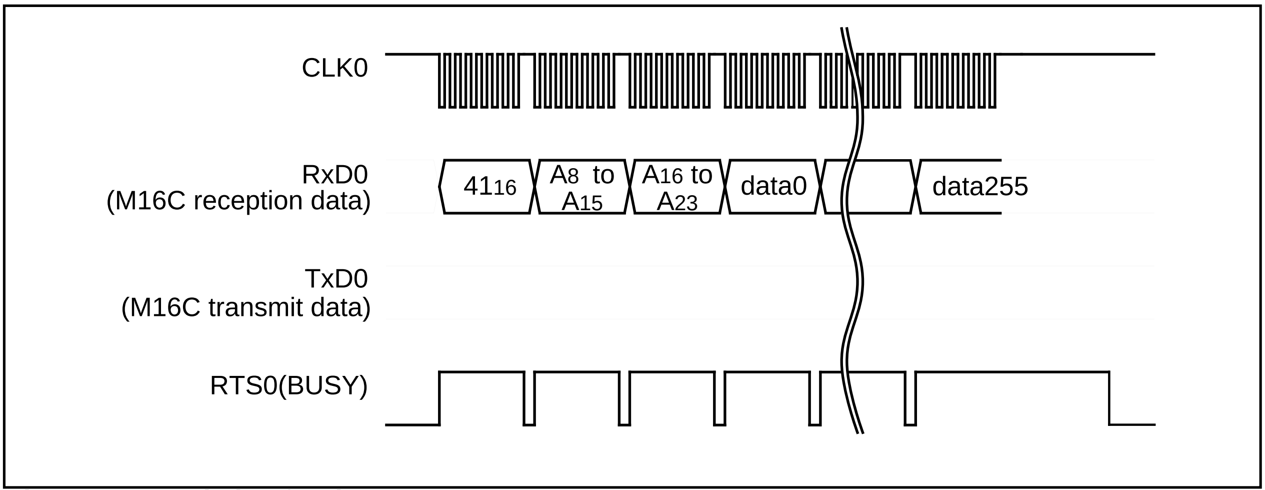


Figure 1.25.5. Timing for the page program

Block Erase Command

This command erases the data in the specified block. Execute the block erase command as explained here following.

(1) Transfer the “20_{16}” command code with the 1st byte.

(2) Transfer addresses A_8 to A_{15} and A_{16} to A_{23} with the 2nd and 3rd bytes respectively.

(3) Transfer the verify command code “$D0_{16}$” with the 4th byte. With the verify command code, the erase operation will start for the specified block in the flash memory. Write the optional even address of the specified block for addresses A_8 to A_{23}.

When block erasing ends, the RTS_0 (BUSY) signal changes from the “H” to the “L” level. After block erase ends, the result of the block erase operation can be known by reading the status register. For more information, see the section on the status register.

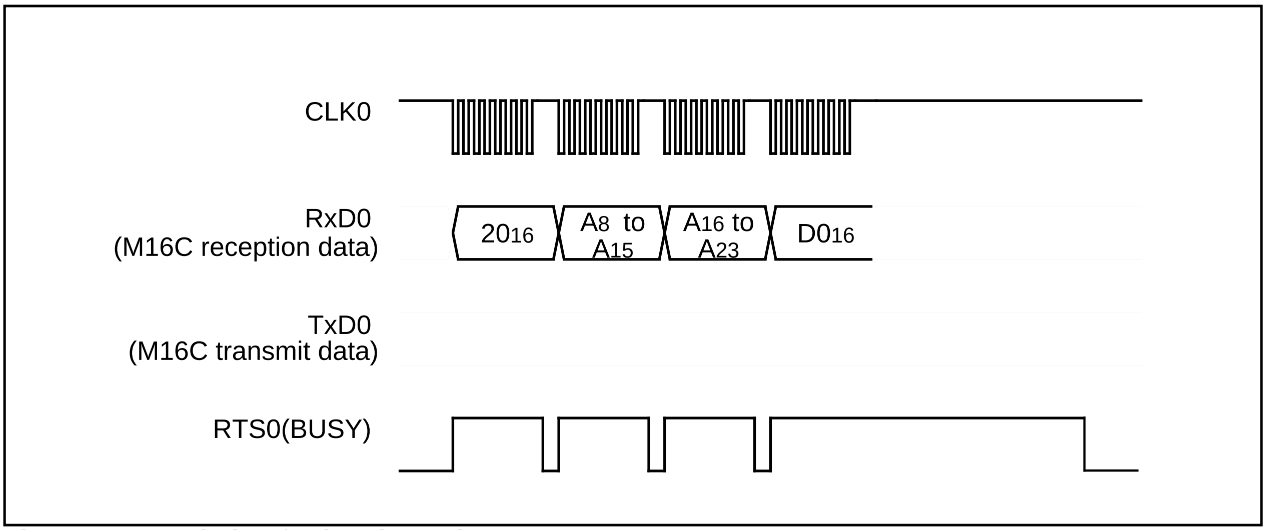


Figure 1.25.6. Timing for block erasing

Erase All Blocks Command

This command erases the content of all blocks. Execute the erase all blocks command as explained here following.

(1) Transfer the “A7$_{16}$” command code with the 1st byte.

(2) Transfer the verify command code “D0$_{16}$” with the 2nd byte. With the verify command code, the erase operation will start and continue for all blocks in the flash memory.

When block erasing ends, the RTS$_0$ (BUSY) signal changes from the “H” to the “L” level. The result of the erase operation can be known by reading the status register.

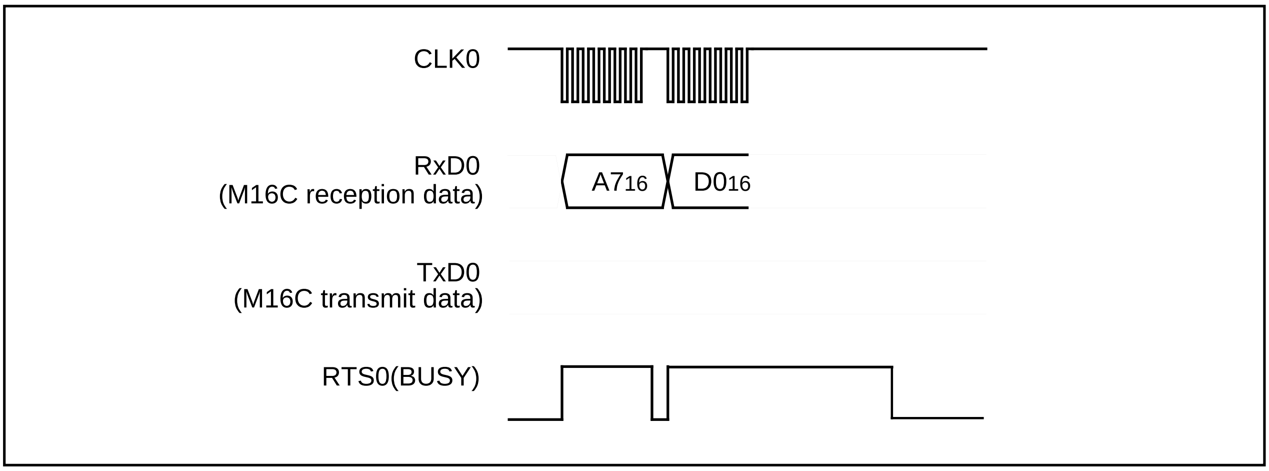


Figure 1.25.7. Timing for erasing all blocks

Download Command

This command downloads a program to the RAM for execution. Execute the download command as explained here following.

(1) Transfer the “FA$_{16}$” command code with the 1st byte.

(2) Transfer the program size with the 2nd and 3rd bytes.

(3) Transfer the check sum with the 4th byte. The check sum is added to all data sent with the 5th byte onward.

(4) The program to execute is sent with the 5th byte onward.

When all data has been transmitted, if the check sum matches, the downloaded program is executed. The size of the program will vary according to the internal RAM.

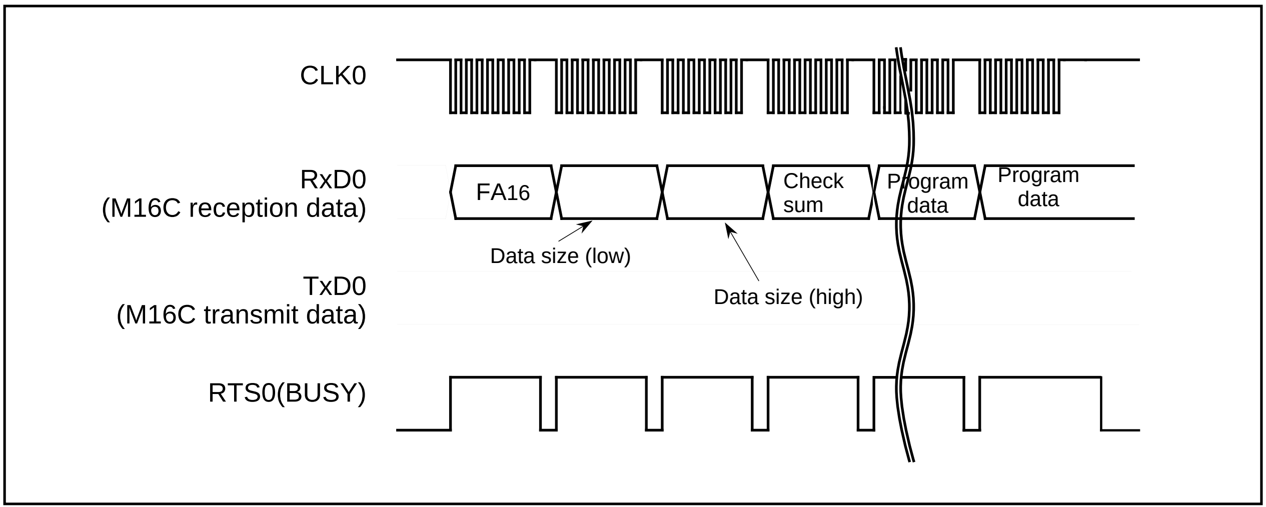


Figure 1.25.8. Timing for download

Version Information Output Command

This command outputs the version information of the control program stored in the boot area. Execute the version information output command as explained here following.

(1) Transfer the “FB_{16}” command code with the 1st byte.

(2) The version information will be output from the 2nd byte onward. This data is composed of 8 ASCII code characters.

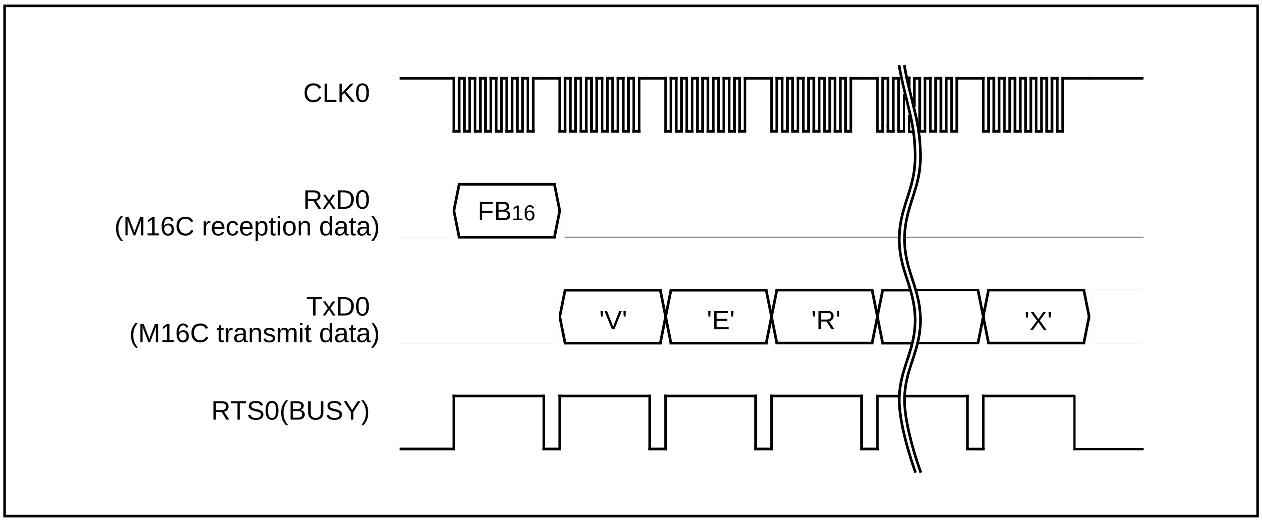


Figure 1.25.9. Timing for version information output

Boot ROM Area Output Command

This command outputs the control program stored in the boot ROM area in one page blocks (256 bytes). Execute the boot ROM area output command as explained here following.

(1) Transfer the “FC_{16}” command code with the 1st byte.

(2) Transfer addresses A_8 to A_{15} and A_{16} to A_{23} with the 2nd and 3rd bytes respectively.

(3) From the 4th byte onward, data (D_0–D_7) for the page (256 bytes) specified with addresses A_8 to A_{23} will be output sequentially from the smallest address first, in sync with the fall of the clock.

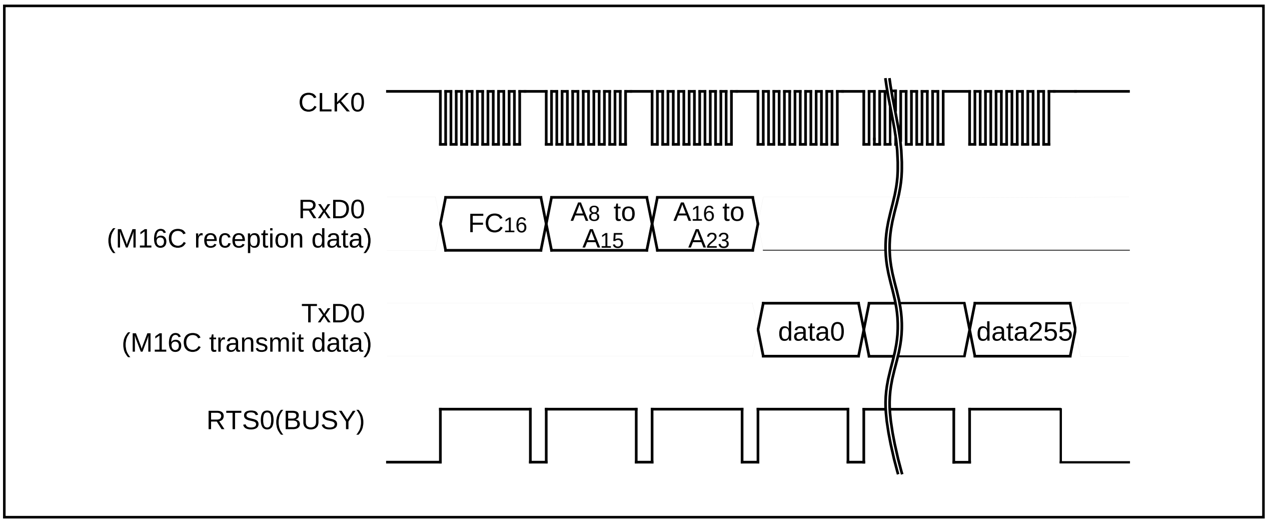


Figure 1.25.10. Timing for boot ROM area output

ID Check

This command checks the ID code. Execute the boot ID check command as explained here following.

(1) Transfer the “$F5_{16}$” command code with the 1st byte.

(2) Transfer addresses A_0 to A_7, A_8 to A_{15} and A_{16} to A_{23} of the 1st byte of the ID code with the 2nd, 3rd and 4th bytes respectively.

(3) Transfer the number of data sets of the ID code with the 5th byte.

(4) The ID code is sent with the 6th byte onward, starting with the 1st byte of the code.

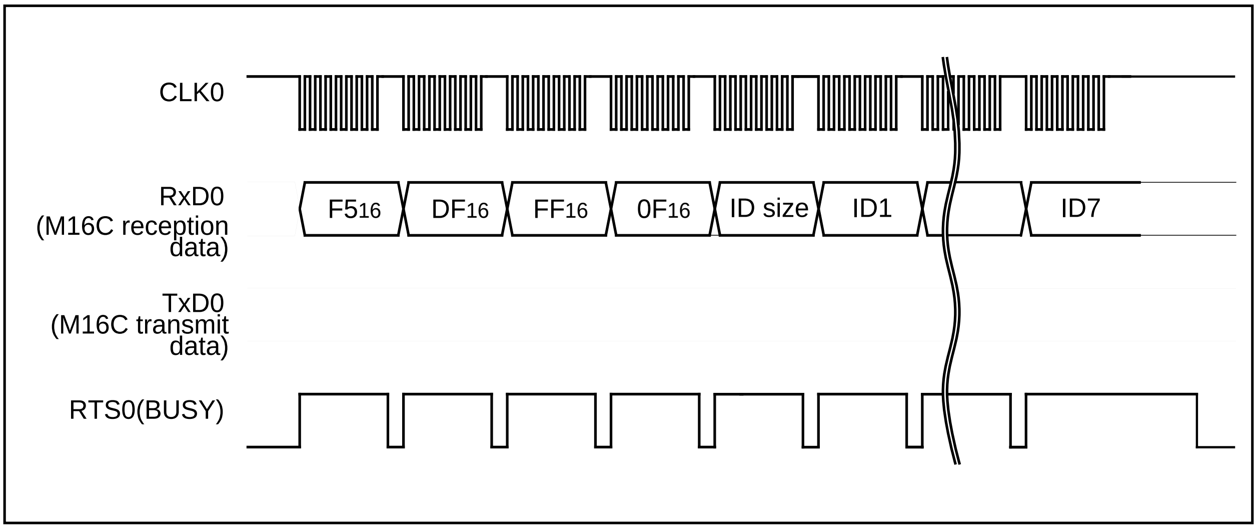
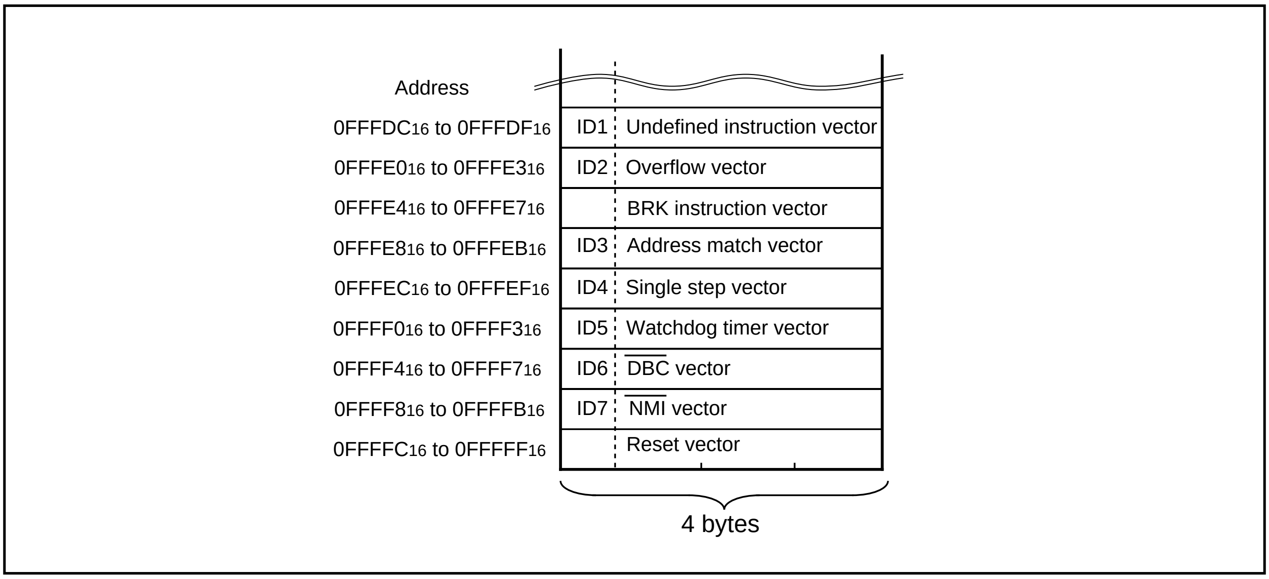


Figure 1.25.11. Timing for the ID check

ID Code

When the flash memory is not blank, the ID code sent from the peripheral units and the ID code written in the flash memory are compared to see if they match. If the codes do not match, the command sent from the peripheral units is not accepted. An ID code contains 8 bits of data. Area is, from the 1st byte, addresses $0FFFDF_{16}$, $0FFFE3_{16}$, $0FFFEB_{16}$, $0FFFEF_{16}$, $0FFFF3_{16}$, $0FFFF7_{16}$ and $0FFFFB_{16}$. Write a program into the flash memory, which already has the ID code set for these addresses.



Address		
$0FFFDC_{16}$ to $0FFFDF_{16}$	ID1	Undefined instruction vector
$0FFFE0_{16}$ to $0FFFE3_{16}$	ID2	Overflow vector
$0FFFE4_{16}$ to $0FFFE7_{16}$		BRK instruction vector
$0FFFE8_{16}$ to $0FFFEB_{16}$	ID3	Address match vector
$0FFFEC_{16}$ to $0FFFEF_{16}$	ID4	Single step vector
$0FFFF0_{16}$ to $0FFFF3_{16}$	ID5	Watchdog timer vector
$0FFFF4_{16}$ to $0FFFF7_{16}$	ID6	$\overline{\text{DBC}}$ vector
$0FFFF8_{16}$ to $0FFFFB_{16}$	ID7	$\overline{\text{NMI}}$ vector
$0FFFFC_{16}$ to $0FFFFF_{16}$		Reset vector

Figure 1.25.12. ID code storage addresses

Read Check Data

This command reads the check data that confirms that the write data, which was sent with the page program command, was successfully received.

(1) Transfer the "FD_{16}" command code with the 1st byte.

(2) The check data (low) is received with the 2nd byte and the check data (high) with the 3rd.

To use this read check data command, first execute the command and then initialize the check data. Next, execute the page program command the required number of times. After that, when the read check command is executed again, the check data for all of the read data that was sent with the page program command during this time is read. Check data adds write data in 1 byte units and obtains the two's-compliment of the insignificant 2 bytes of the accumulated data.

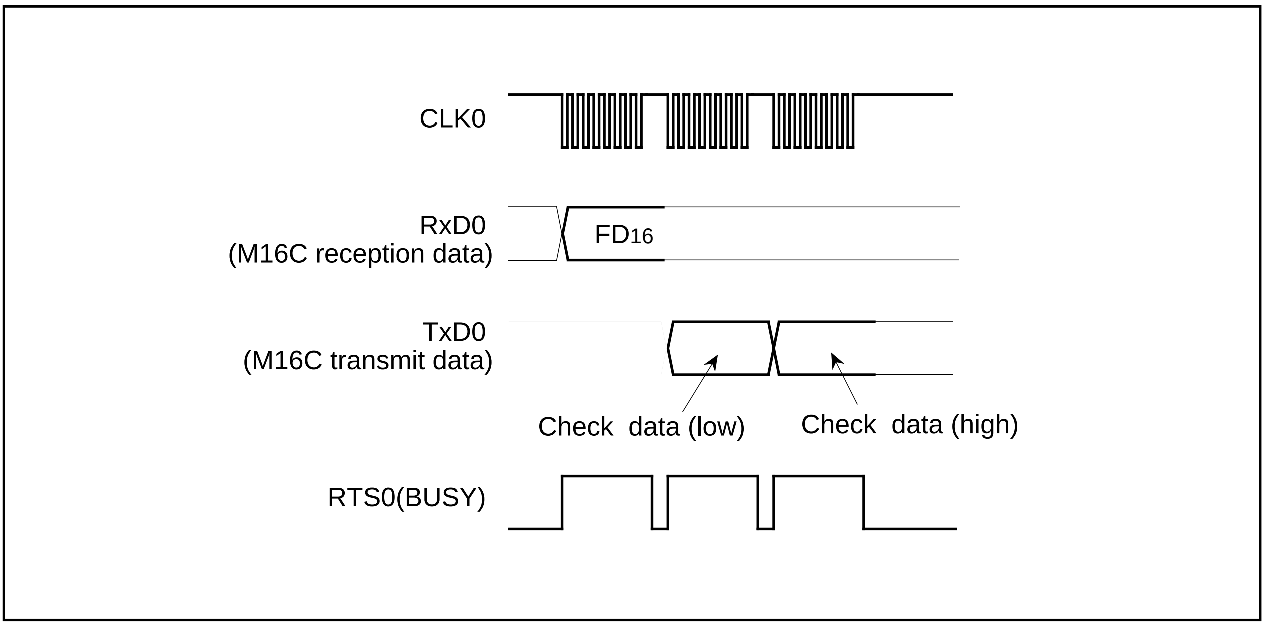


Figure 1.25.13. Timing for the read check data

Status Register (SRD)

The status register indicates operating status of the flash memory and status such as whether an erase operation or a program ended successfully or in error. It can be read by writing the read status register command (70_{16}). Also, the status register is cleared by writing the clear status register command (50_{16}). Table 1.25.2 gives the definition of each status register bit. After clearing the reset, the status register outputs “80_{16}”.

Table 1.25.2. Status register (SRD)

SRD0 bits	Status name	Definition	
		"1"	"0"
SR7 (bit7)	Sequencer 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)	Reserved	-	-
SR2 (bit2)	Reserved	-	-
SR1 (bit1)	Reserved	-	-
SR0 (bit0)	Reserved	-	-

Sequencer status (SR7)

After power-on, the sequencer status is set to 1(ready).

The sequencer status indicates the operating status of the device. This status bit is set to 0 (busy) during write or erase operation and is set to 1 upon completion of these operations.

Erase Status (SR5)

The erase status reports the operating status of the 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”.

Status Register 1 (SRD1)

Status register 1 indicates the status of serial communications, results from ID checks and results from check sum comparisons. It can be read after the SRD by writing the read status register command (70_{16}). Also, status register 1 is cleared by writing the clear status register command (50_{16}).
Table 1.25.3 gives the definition of each status register 1 bit. “00_{16}” is output when power is turned ON and the flag status is maintained even after the reset.

Table 1.25.3. Status register 1 (SRD1)

SRD1 bits	Status name	Definition	
		"1"	"0"
SR15 (bit7)	Boot update completed bit	Update completed	Not update
SR14 (bit6)	Reserved	-	-
SR13 (bit5)	Reserved	-	-
SR12 (bit4)	Check sum 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	-	-

Boot Update Completed Bit (SR15)

This flag indicates whether the control program was downloaded to the RAM or not, using the download function.

Check Sum Match Bit (SR12)

This flag indicates whether the check sum matches or not when a program, is downloaded for execution using the download function.

ID Check Completed Bits (SR11 and SR10)

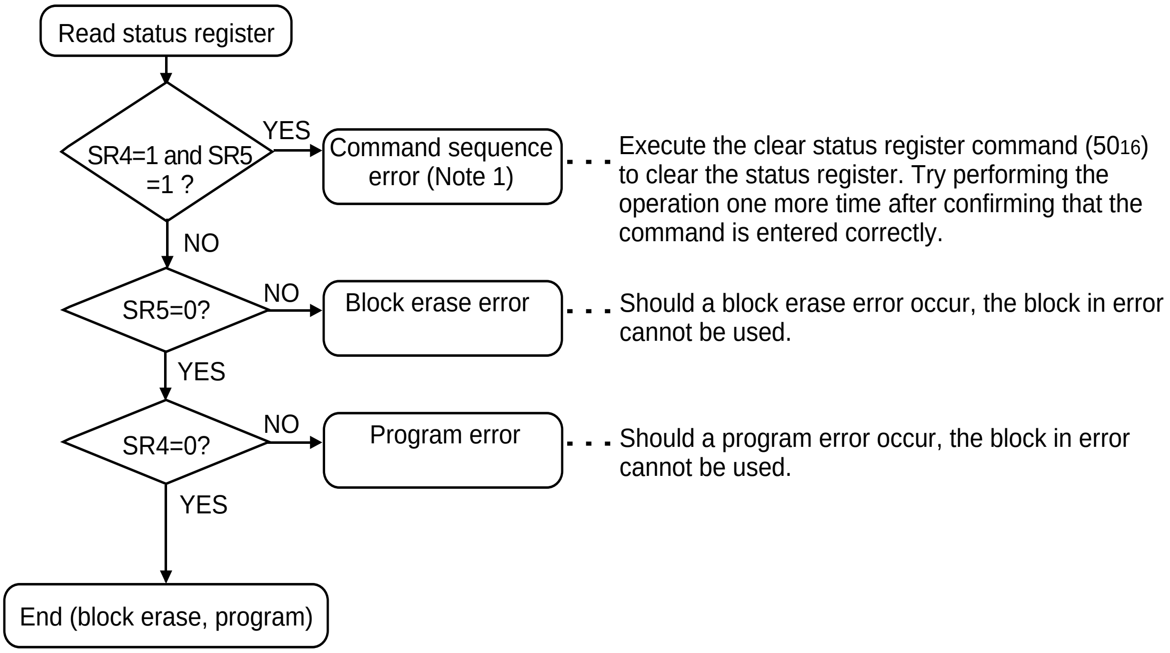
These flags indicate the result of ID checks. Some commands cannot be accepted without an ID check.

Data Receive Time Out (SR9)

This flag indicates when a time out error is generated during data reception. If this flag is attached during data reception, the received data is discarded and the microcomputer returns to the command wait state.

Full Status Check

Results from executed erase and program operations can be known by running a full status check. Figure 1.25.14 shows a flowchart of the full status check and explains how to remedy errors which 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.

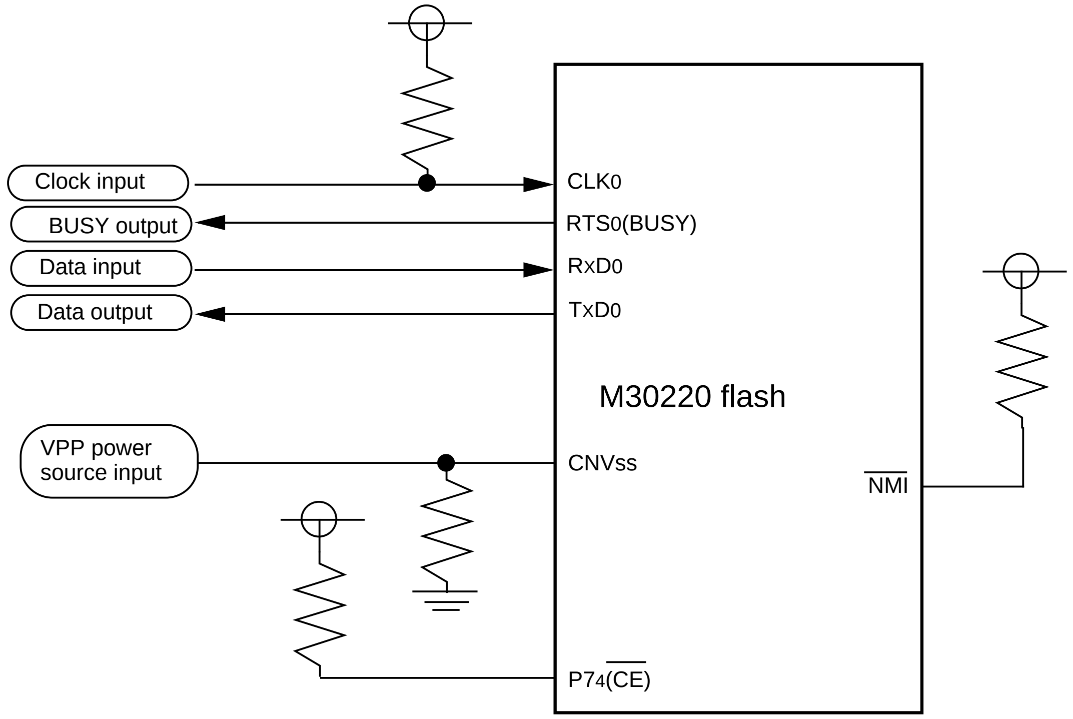
. . . Should a program error occur, the block in error cannot be used.

Note 1: Either SR5 or SR4 may become "0". Take care about it during program debugging.
Note 2: When one of SR5 to SR4 is set to 1, none of the program, erase all blocks, and block erase commands is accepted. Execute the clear status register command (50_{16}) before executing these commands.

Figure 1.25.14. Full status check flowchart and remedial procedure for errors

Example Circuit Application for The Standard Serial I/O Mode 1

The below figure shows a circuit application for the standard serial I/O mode 1. Control pins will vary according to programmer, therefore see the peripheral unit manual for more information.



(1) Control pins and external circuitry will vary according to peripheral unit. For more information, see the peripheral unit manual.
(2) In this example, the Vpp power supply is supplied from an external source (writer). To use the user's power source, connect to 4.5V to 5.5 V.

Figure 1.25.15. Example circuit application for the standard serial I/O mode 1

Overview of standard serial I/O mode 2 (clock asynchronized)

In standard serial I/O mode 2, software commands, addresses and data are input and output between the MCU and peripheral units (serial programer, etc.) using 2-wire clock-asynchronized serial I/O (UART0).

Standard serial I/O mode 2 is engaged by releasing the reset with the P6$_1$ (CLK$_0$) pin "L" level.

The TxD$_0$ pin is for CMOS output. Data transfer is in 8-bit units with LSB first, 1 stop bit and parity OFF.

After the reset is released, connections can be established at 9,600 bps when initial communications (Figure 1.25.16) are made with a peripheral unit. However, this requires a main clock with a minimum 2 MHz input oscillation frequency. Baud rate can also be changed from 9,600 bps to 19,200, 38,400 or 57,600 bps by executing software commands. However, communication errors may occur because of the oscillation frequency of the main clock. If errors occur, change the main clock's oscillation frequency and the baud rate.

After executing commands from a peripheral unit that requires time to erase and write data, as with erase and program commands, allow a sufficient time interval or execute the read status command and check how processing ended, before executing the next command.

Data and status registers in memory can be read after transmitting software commands. Status, such as the operating state of the flash memory or whether a program or erase operation ended successfully or not, can be checked by reading the status register. Here following are explained initial communications with peripheral units, how frequency is identified and software commands.

Initial communications with peripheral units

After the reset is released, the bit rate generator is adjusted to 9,600 bps to match the oscillation frequency of the main clock, by sending the code as prescribed by the protocol for initial communications with peripheral units (Figure 1.25.16).

(1) Transmit "B0$_{16}$" from a peripheral unit. If the oscillation frequency input by the main clock is 10 MHz, the MCU with internal flash memory outputs the "B0$_{16}$" check code. If the oscillation frequency is anything other than 10 MHz, the MCU does not output anything.

(2) Transmit "00$_{16}$" from a peripheral unit 16 times. (The MCU with internal flash memory sets the bit rate generator so that "00$_{16}$" can be successfully received.)

(3) The MCU with internal flash memory outputs the "B0$_{16}$" check code and initial communications end successfully *^{1}. Initial communications must be transmitted at a speed of 9,600 bps and a transfer interval of a minimum 15 ms. Also, the baud rate at the end of initial communications is 9,600 bps.

*1. If the peripheral unit cannot receive "B0$_{16}$" successfully, change the oscillation frequency of the main clock.

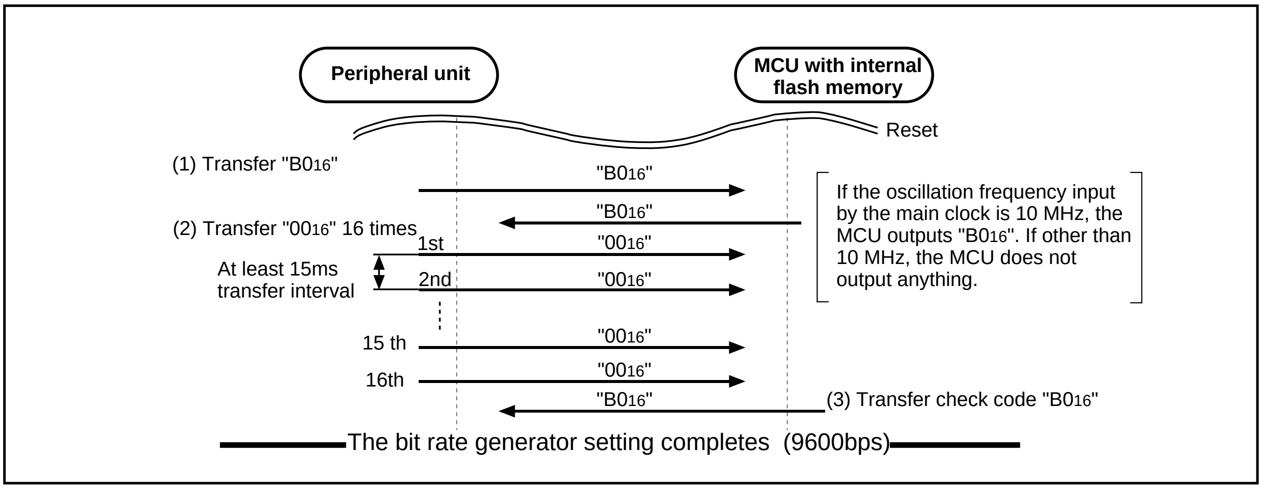

Peripheral unit
MCU with internal flash memory
Reset
(1) Transfer "B0$_{16}$"
"B0$_{16}$"
"B0$_{16}$"
If the oscillation frequency input by the main clock is 10 MHz, the MCU outputs "B0$_{16}$". If other than 10 MHz, the MCU does not output anything.
(2) Transfer "00$_{16}$" 16 times
1st
"00$_{16}$"
At least 15ms transfer interval
2nd
"00$_{16}$"
15 th
"00$_{16}$"
16th
"00$_{16}$"
"B0$_{16}$"
(3) Transfer check code "B0$_{16}$"
The bit rate generator setting completes (9600bps)

Figure 1.25.16. Peripheral unit and initial communication

How frequency is identified

When "00_{16}" data is received 16 times from a peripheral unit at a baud rate of 9,600 bps, the value of the bit rate generator is set to match the operating frequency (2 - 10 MHz). The highest speed is taken from the first 8 transmissions and the lowest from the last 8. These values are then used to calculate the bit rate generator value for a baud rate of 9,600 bps.

Baud rate cannot be attained with some operating frequencies. Table 1.25.4 gives the operation frequency and the baud rate that can be attained for.

Table 1.25.4 Operation frequency and the baud rate

Operation frequency (MHz)	Baud rate 9,600bps	Baud rate 19,200bps	Baud rate 38,400bps	Baud rate 57,600bps
10MHz	√	√	–	√
8MHz	√	√	–	√
7.3728MHz	√	√	√	√
6MHz	√	√	√	–
5MHz	√	√	–	–
4.5MHz	√	√	–	√
4.194304MHz	√	√	√	–
4MHz	√	√	–	–
3.58MHz	√	√	√	√
3MHz	√	√	√	–
2MHz	√	–	–	–

√ : Communications possible
– : Communications not possible

Software Commands

Table 1.25.5 lists software commands. In the standard serial I/O mode 2, erase operations, programs and reading are controlled by transferring software commands via the RxD$_0$ pin. Standard serial I/O mode 2 adds four transmission speed commands - 9,600, 19,200, 38,400 and 57,600 bps - to the software commands of standard serial I/O mode 1. Software commands are explained here below.

Table 1.25.5. Software commands (Standard serial I/O mode 2)

	Control command	1st byte transfer	2nd byte	3rd byte	4th byte	5th byte	6th byte		When ID is not verified
1	Page read	FF$_{16}$	Address (middle)	Address (high)	Data output	Data output	Data output	Data output to 259th byte	Not acceptable
2	Page program	41$_{16}$	Address (middle)	Address (high)	Data input	Data input	Data input	Data input to 259th byte	Not acceptable
3	Block erase	20$_{16}$	Address (middle)	Address (high)	D0$_{16}$				Not acceptable
4	Erase all unlocked blocks	A7$_{16}$	D0$_{16}$						Not acceptable
5	Read status register	70$_{16}$	SRD output	SRD1 output					Acceptable
6	Clear status register	50$_{16}$							Not acceptable
7	ID check function	F5$_{16}$	Address (low)	Address (middle)	Address (high)	ID size	ID1	To ID7	Acceptable
8	Download function	FA$_{16}$	Size (low)	Size (high)	Check-sum	Data input	To required number of times		Not acceptable
9	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
10	Boot ROM area output function	FC$_{16}$	Address (middle)	Address (high)	Data output	Data output	Data output	Data output to 259th byte	Not acceptable
11	Read check data	FD$_{16}$	Check data (low)	Check data (high)					Not acceptable
12	Baud rate 9600	B0$_{16}$	B0$_{16}$						Acceptable
13	Baud rate 19200	B1$_{16}$	B1$_{16}$						Acceptable
14	Baud rate 38400	B2$_{16}$	B2$_{16}$						Acceptable
15	Baud rate 57600	B3$_{16}$	B3$_{16}$						Acceptable

Note 1: Shading indicates transfer from flash memory microcomputer to peripheral unit. All other data is transferred from the peripheral unit to the flash memory microcomputer.

Note 2: SRD refers to status register data. SRD1 refers to status register 1 data.

Note 3: All commands can be accepted when the flash memory is totally blank.

Page Read Command

This command reads the specified page (256 bytes) in the flash memory sequentially one byte at a time. Execute the page read command as explained here following.

(1) Transfer the “FF16” command code with the 1st byte.

(2) Transfer addresses A_8 to A_{15} and A_{16} to A_{23} with the 2nd and 3rd bytes respectively.

(3) From the 4th byte onward, data (D_0–D_7) for the page (256 bytes) specified with addresses A_8 to A_{23} will be output sequentially from the smallest address first.

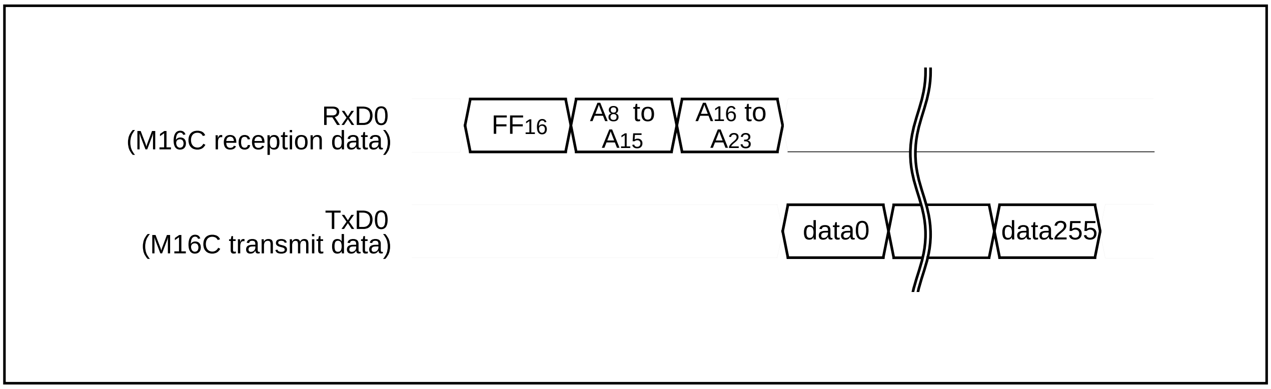


Figure 1.25.17. Timing for page read

Read Status Register Command

This command reads status information. When the “7016” command code is sent with the 1st byte, the contents of the status register (SRD) specified with the 2nd byte and the contents of status register 1 (SRD1) specified with the 3rd byte are read.

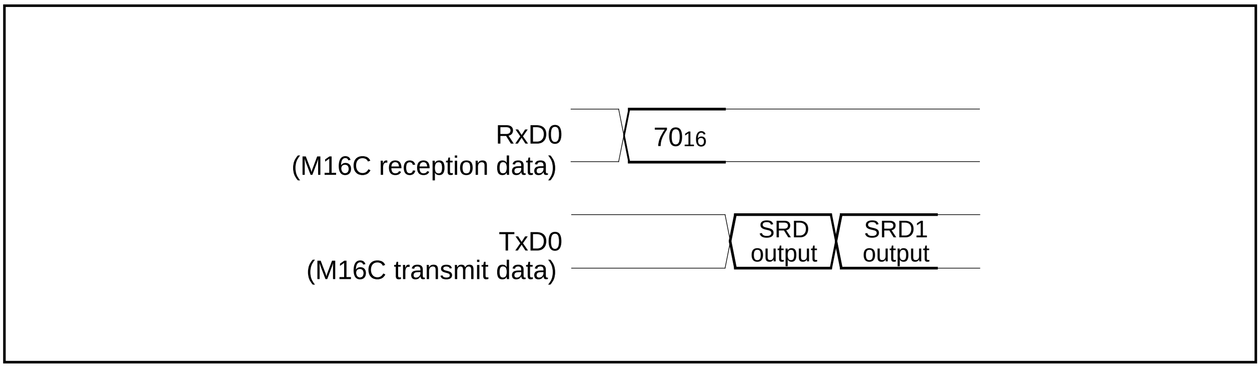


Figure 1.25.18. Timing for reading the status register

Clear Status Register Command

This command clears the bits (SR4–SR5) which are set when the status register operation ends in error. When the “50_{16}” command code is sent with the 1st byte, the aforementioned bits are cleared.

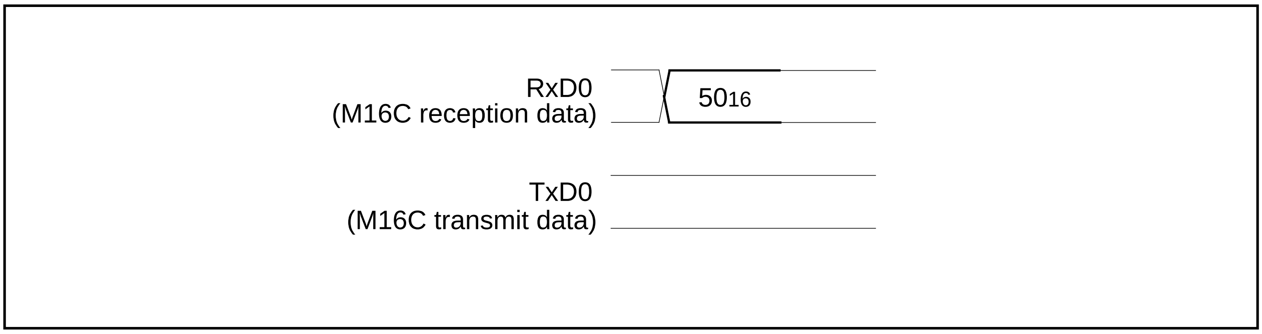


Figure 1.25.19. Timing for clearing the status register

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 here following.

(1) Transfer the “41_{16}” command code with the 1st byte.

(2) Transfer addresses A_8 to A_{15} and A_{16} to A_{23} with the 2nd and 3rd bytes respectively.

(3) From the 4th byte onward, as write data (D_0–D_7) for the page (256 bytes) specified with addresses A_8 to A_{23} is input sequentially from the smallest address first, that page is automatically written.

The result of the page program can be known by reading the status register. For more information, see the section on the status register.

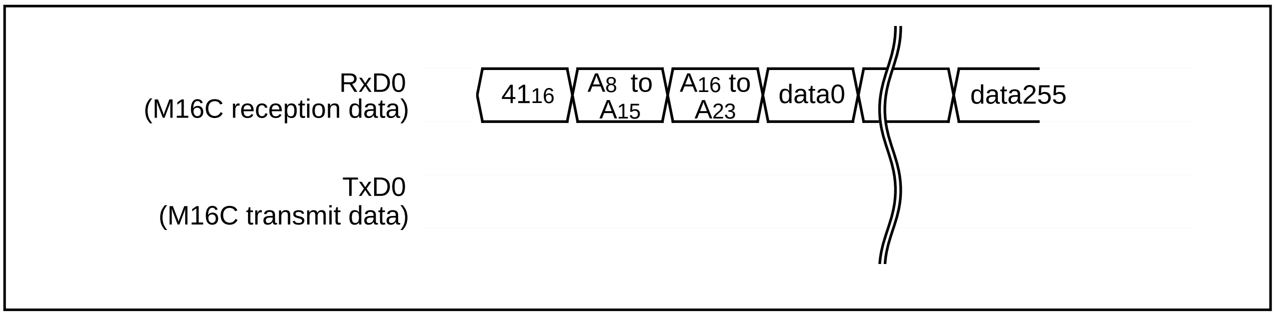


Figure 1.25.20. Timing for the page program

Block Erase Command

This command erases the data in the specified block. Execute the block erase command as explained here following.

(1) Transfer the “20_{16}” command code with the 1st byte.

(2) Transfer addresses A_8 to A_{15} and A_{16} to A_{23} with the 2nd and 3rd bytes respectively.

(3) Transfer the verify command code “$D0_{16}$” with the 4th byte. With the verify command code, the erase operation will start for the specified block in the flash memory. Write the optional even address of the specified block for addresses A_8 to A_{23}.

After block erase ends, the result of the block erase operation can be known by reading the status register. For more information, see the section on the status register.

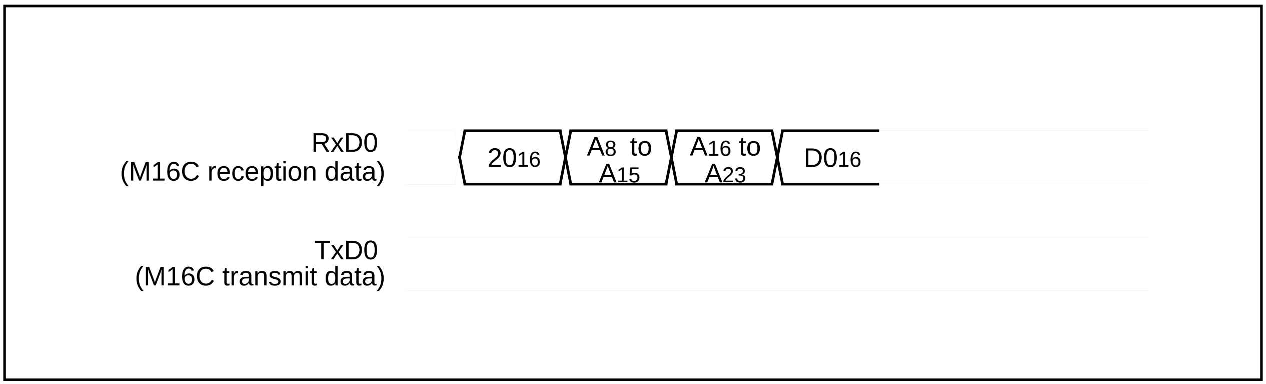


Figure 1.25.21. Timing for block erasing

Erase All Blocks Command

This command erases the content of all blocks. Execute the erase all blocks command as explained here following.

(1) Transfer the “$A7_{16}$” command code with the 1st byte.

(2) Transfer the verify command code “$D0_{16}$” with the 2nd byte. With the verify command code, the erase operation will start and continue for all blocks in the flash memory.

The result of the erase operation can be known by reading the status register.

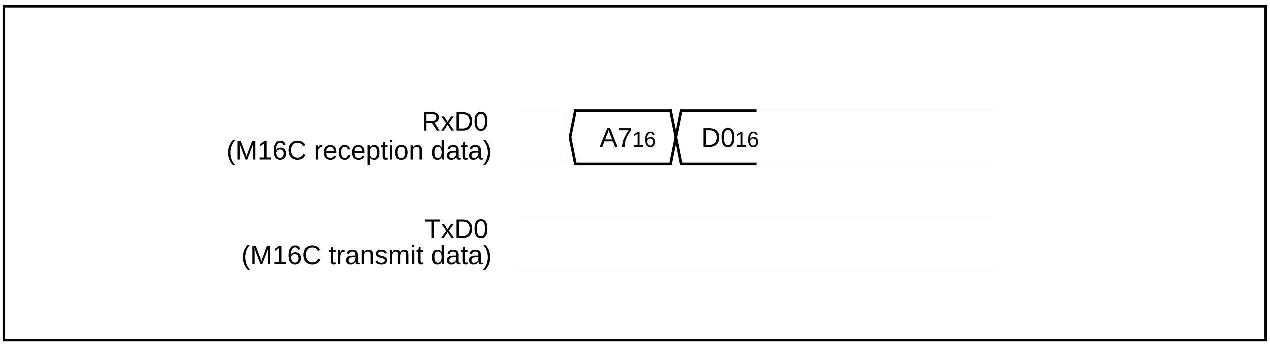


Figure 1.25.22. Timing for erasing all blocks

Download Command

This command downloads a program to the RAM for execution. Execute the download command as explained here following.

(1) Transfer the “FA_{16}” command code with the 1st byte.

(2) Transfer the program size with the 2nd and 3rd bytes.

(3) Transfer the check sum with the 4th byte. The check sum is added to all data sent with the 5th byte onward.

(4) The program to execute is sent with the 5th byte onward.

When all data has been transmitted, if the check sum matches, the downloaded program is executed. The size of the program will vary according to the internal RAM.

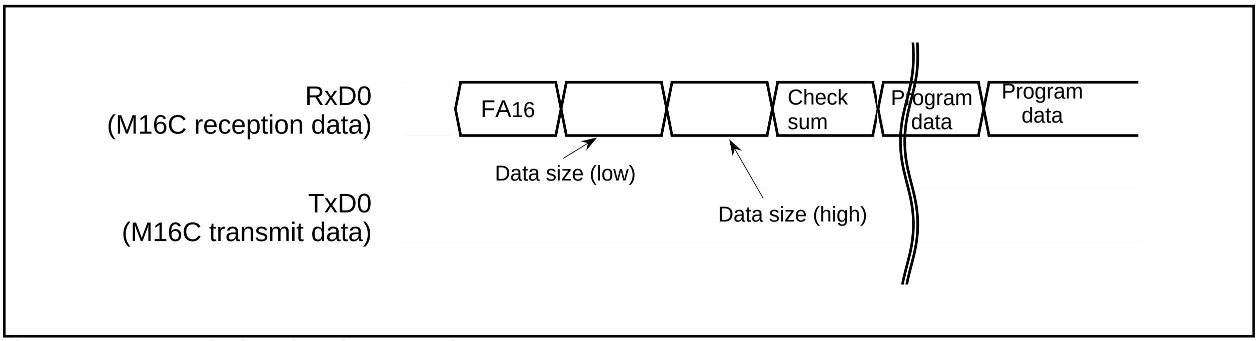


Figure 1.25.23. Timing for download

Version Information Output Command

This command outputs the version information of the control program stored in the boot area. Execute the version information output command as explained here following.

(1) Transfer the “FB_{16}” command code with the 1st byte.

(2) The version information will be output from the 2nd byte onward. This data is composed of 8 ASCII code characters.

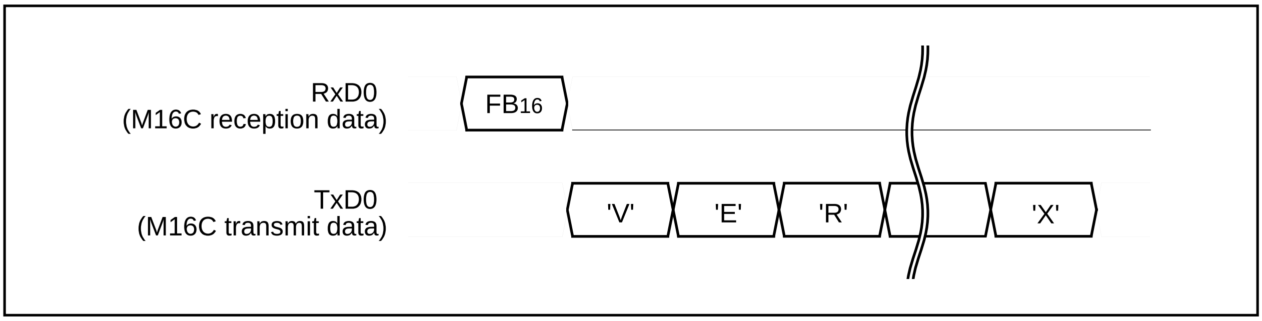


Figure 1.25.24. Timing for version information output

Boot ROM Area Output Command

This command outputs the control program stored in the boot ROM area in one page blocks (256 bytes). Execute the boot ROM area output command as explained here following.

(1) Transfer the “FC_{16}” command code with the 1st byte.

(2) Transfer addresses A_8 to A_{15} and A_{16} to A_{23} with the 2nd and 3rd bytes respectively.

(3) From the 4th byte onward, data (D_0–D_7) for the page (256 bytes) specified with addresses A_8 to A_{23} will be output sequentially from the smallest address first.

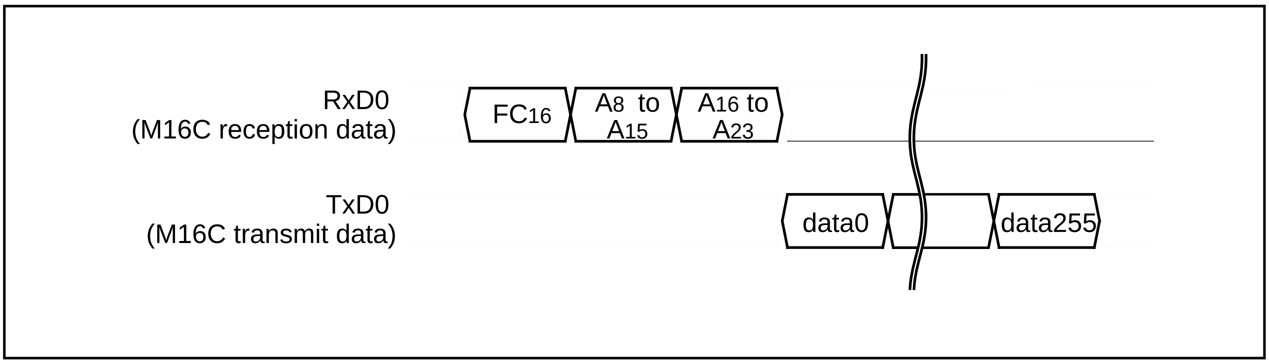


Figure 1.25.25. Timing for boot ROM area output

ID Check

This command checks the ID code. Execute the boot ID check command as explained here following.

(1) Transfer the “F5$_{16}$” command code with the 1st byte.

(2) Transfer addresses A$_0$ to A$_7$, A$_8$ to A$_{15}$ and A$_{16}$ to A$_{23}$ of the 1st byte of the ID code with the 2nd, 3rd and 4th bytes respectively.

(3) Transfer the number of data sets of the ID code with the 5th byte.

(4) The ID code is sent with the 6th byte onward, starting with the 1st byte of the code.

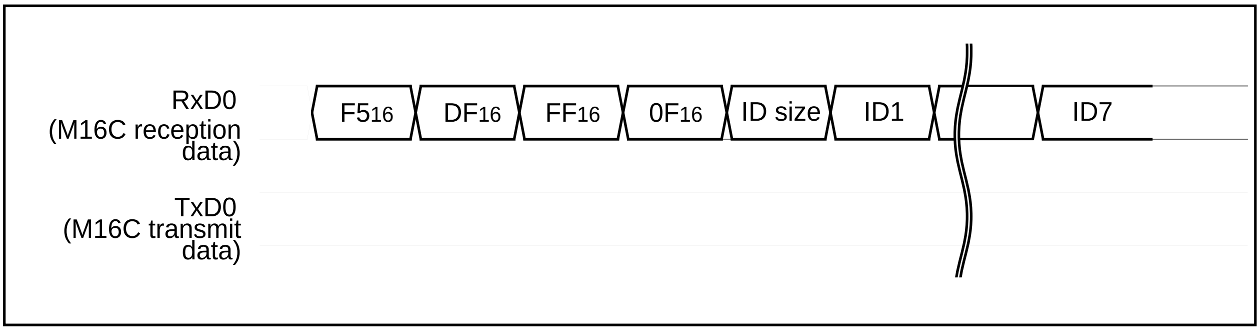
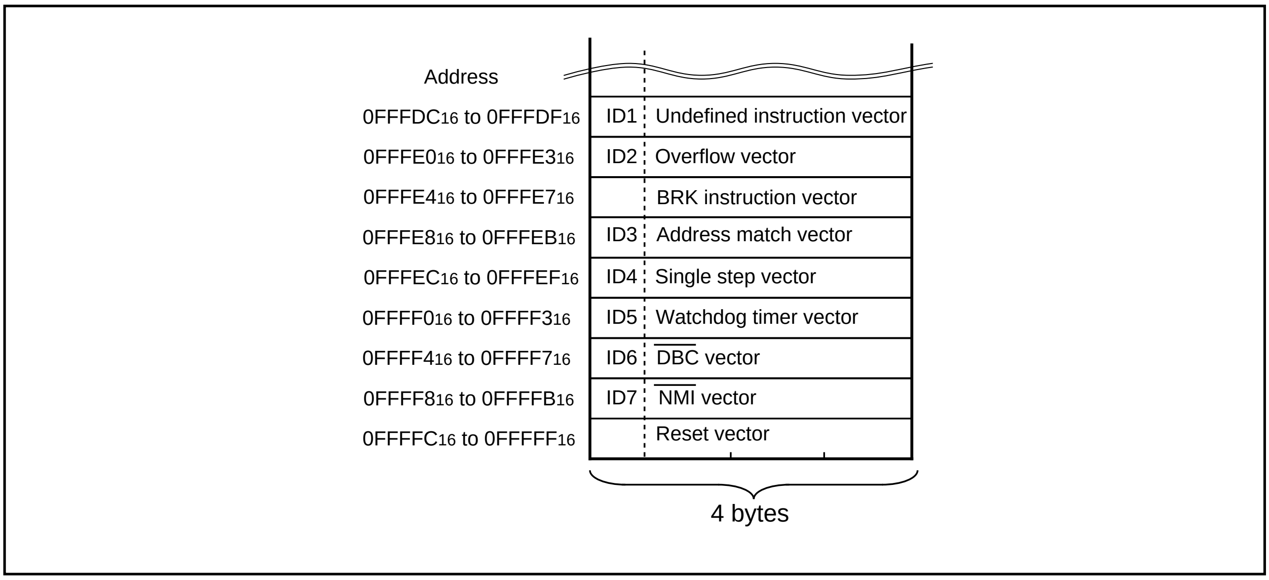


Figure 1.25.26. Timing for the ID check

ID Code

When the flash memory is not blank, the ID code sent from the peripheral units and the ID code written in the flash memory are compared to see if they match. If the codes do not match, the command sent from the peripheral units is not accepted. An ID code contains 8 bits of data. Area is, from the 1st byte, addresses 0FFFDF$_{16}$, 0FFFE3$_{16}$, 0FFFEB$_{16}$, 0FFFEF$_{16}$, 0FFFF3$_{16}$, 0FFFF7$_{16}$ and 0FFFFB$_{16}$. Write a program into the flash memory, which already has the ID code set for these addresses.



Address		
0FFFDC$_{16}$ to 0FFFDF$_{16}$	ID1	Undefined instruction vector
0FFFE0$_{16}$ to 0FFFE3$_{16}$	ID2	Overflow vector
0FFFE4$_{16}$ to 0FFFE7$_{16}$		BRK instruction vector
0FFFE8$_{16}$ to 0FFFEB$_{16}$	ID3	Address match vector
0FFFEC$_{16}$ to 0FFFEF$_{16}$	ID4	Single step vector
0FFFF0$_{16}$ to 0FFFF3$_{16}$	ID5	Watchdog timer vector
0FFFF4$_{16}$ to 0FFFF7$_{16}$	ID6	$\overline{\text{DBC}}$ vector
0FFFF8$_{16}$ to 0FFFFB$_{16}$	ID7	$\overline{\text{NMI}}$ vector
0FFFFC$_{16}$ to 0FFFFF$_{16}$		Reset vector

Figure 1.25.27. ID code storage addresses

Read Check Data

This command reads the check data that confirms that the write data, which was sent with the page program command, was successfully received.

(1) Transfer the "FD16" command code with the 1st byte.

(2) The check data (low) is received with the 2nd byte and the check data (high) with the 3rd.

To use this read check data command, first execute the command and then initialize the check data. Next, execute the page program command the required number of times. After that, when the read check command is executed again, the check data for all of the read data that was sent with the page program command during this time is read. Check data adds write data in 1 byte units and obtains the two's-compliment of the insignificant 2 bytes of the accumulated data.

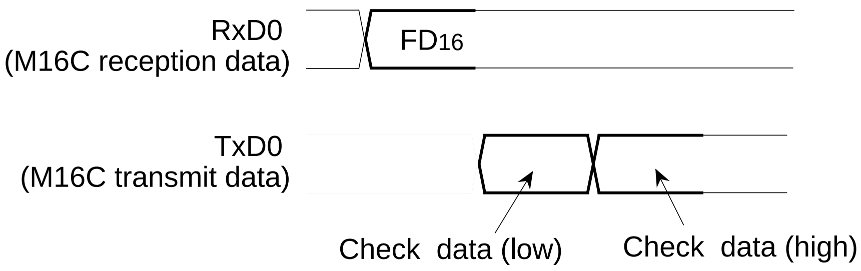


Figure 1.25.28. Timing for the read check data

Baud Rate 9600

This command changes baud rate to 9,600 bps. Execute it as follows.

(1) Transfer the "B016" command code with the 1st byte.

(2) After the "B016" check code is output with the 2nd byte, change the baud rate to 9,600 bps.

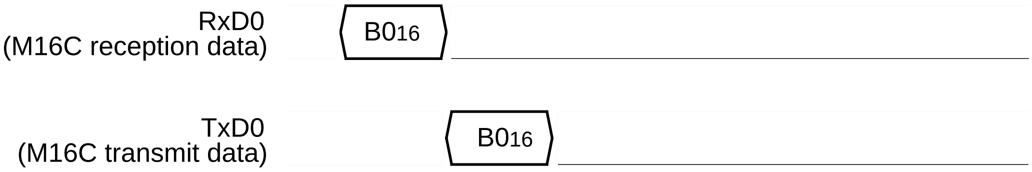


Figure 1.25.29. Timing of baud rate 9600

Baud Rate 19200

This command changes baud rate to 19,200 bps. Execute it as follows.

(1) Transfer the "$B1_{16}$" command code with the 1st byte.

(2) After the "$B1_{16}$" check code is output with the 2nd byte, change the baud rate to 19,200 bps.

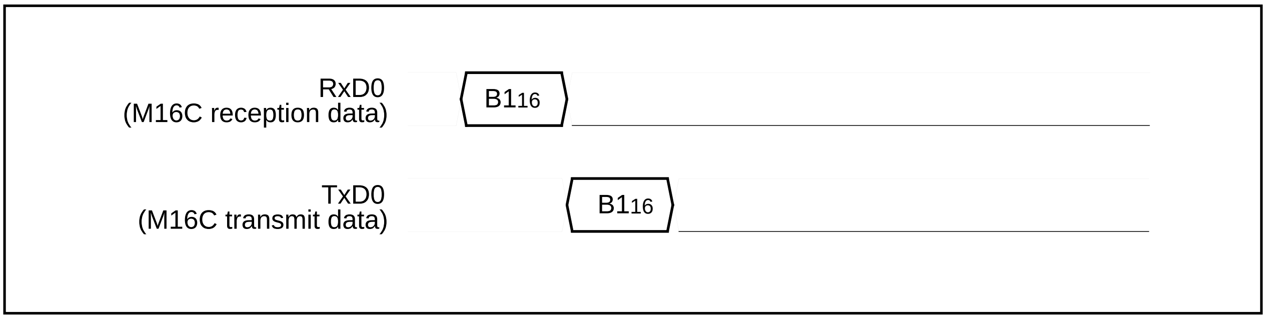


Figure 1.25.30. Timing of baud rate 19200

Baud Rate 38400

This command changes baud rate to 38,400 bps. Execute it as follows.

(1) Transfer the "$B2_{16}$" command code with the 1st byte.

(2) After the "$B2_{16}$" check code is output with the 2nd byte, change the baud rate to 38,400 bps.

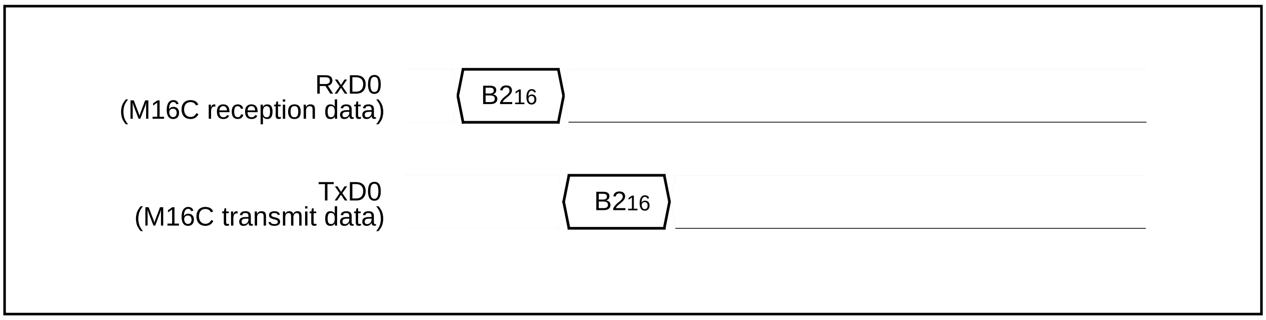


Figure 1.25.31. Timing of baud rate 38400

Baud Rate 57600

This command changes baud rate to 57,600 bps. Execute it as follows.

(1) Transfer the "$B3_{16}$" command code with the 1st byte.

(2) After the "$B3_{16}$" check code is output with the 2nd byte, change the baud rate to 57,600 bps.

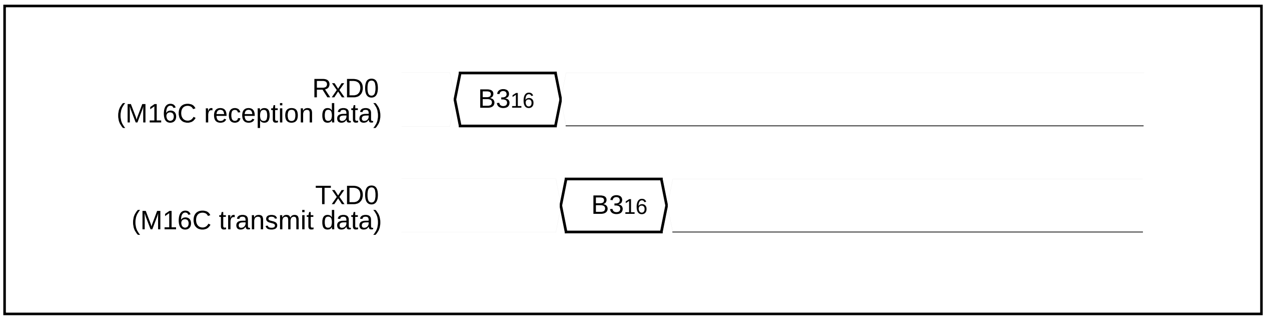
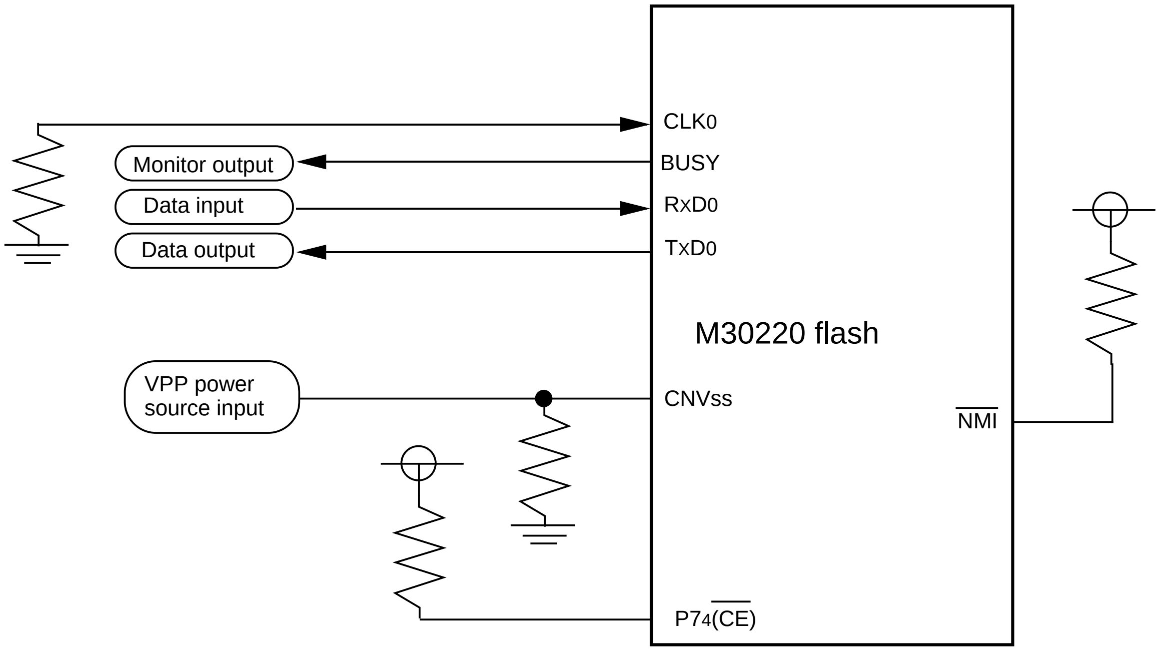


Figure 1.25.32. Timing of baud rate 57600

Example Circuit Application for The Standard Serial I/O Mode 2

The below figure shows a circuit application for the standard serial I/O mode 2.



(1) In this example, the Vpp power supply is supplied from an external source (writer). To use the user's power source, connect to 4.5V to 5.5 V.

Figure 1.25.23. Example circuit application for the standard serial I/O mode 2

Absolute maximum ratings

Symbol	Parameter		Condition	Rated value	Unit
Vcc	Supply voltage		Vcc=AVcc	- 0.3 to 6.5	V
AVcc	Analog supply voltage		Vcc=AVcc	- 0.3 to 6.5	V
V_I	Input voltage	$\overline{\text{RESET}}$, V_{REF}, X_{IN} $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_5$,$P4_0$ 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 $P10_7$, $P11_0$ to $P11_7$, $P12_0$ to $P12_7$, $P13_0$ to $P13_2$ (Mask ROM version CNVss)		-0.3 to Vcc+0.3	V
		V_{L1}		- 0.3 to V_{L2}	
		V_{L2}		V_{L1} to V_{L3}	
		V_{L3}		V_{L2} to 6.5	
		$P7_0$, $P7_1$, C_1, C_2 (Flash memory version CNVss)		- 0.3 to 6.5	
V_O	Output voltage	$P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_5$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_6$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $P13_0$ to $P13_2$, X_{OUT}		- 0.3 to Vcc+0.3	V
		$P0_0$ to $P0_7$, $P10_0$ to $P10_7$, $P11_0$ to $P11_7$, $P12_0$ to $P12_7$,	When output port	- 0.3 to Vcc	
			When segment otput	- 0.3 to V_{L3}	
		$P7_0$, $P7_1$		- 0.3 to 6.5	
P_d	Power dissipation		Ta = 25 °C	300	mW
T_{opr}	Operating ambient temperature (Note)			25±5	°C
T_{stg}	Storage temperature			- 40 to 150	°C

Note: It is a value in flash memory mode. Other parameter becomes same as a value in microcomputer mode.

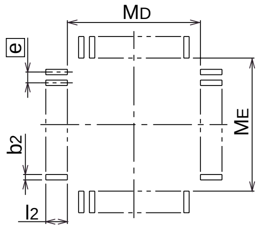
DC electrical characteristics
(referenced to V_{CC} = 4.5V to 5.5V at Ta = 25^{o}C unless otherwise specified)

Symbol	Parameter	Condition	Rated value			Unit
			Min.	Typ.	Max.	
I_{PP1}	V_{PP} power supply current (at read)	V_{PP} =V_{CC}			100	µA
I_{PP2}	V_{PP} power supply current (at program)	V_{PP} =V_{CC}			60	mA
I_{PP3}	V_{PP} power supply current (at erase)	V_{PP} =V_{CC}			30	mA
V_{PP}	V_{PP} power supply voltage		4.5		5.5	V

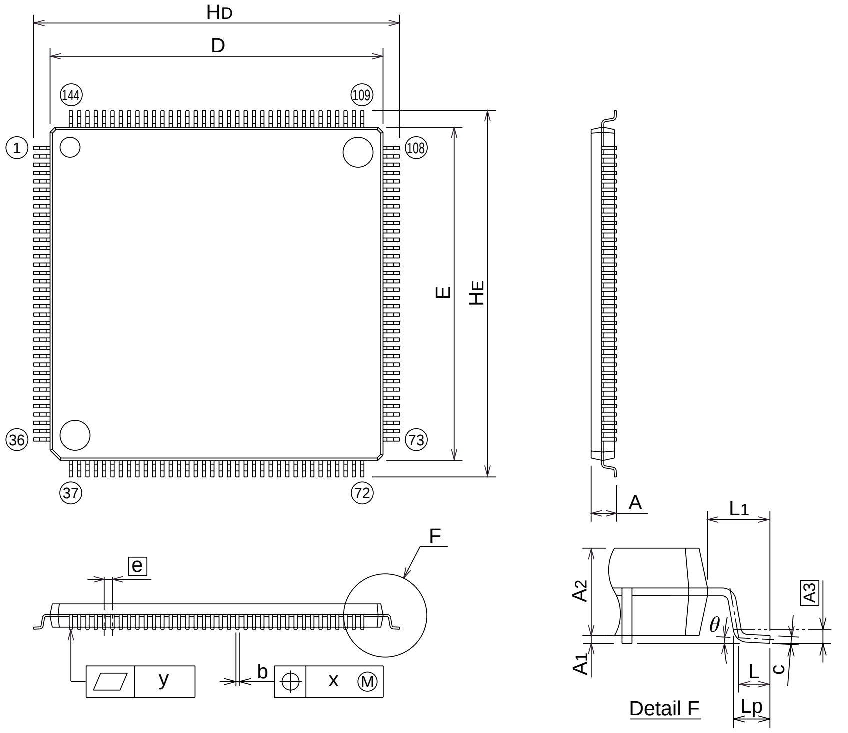
144P6Q-A MMP

Plastic 144pin 20×20mm body LQFP

EIAJ Package Code	JEDEC Code	Weight(g)	Lead Material
LQFP144-P-2020-0.50	–	1.23	Cu Alloy



Recommended Mount Pad

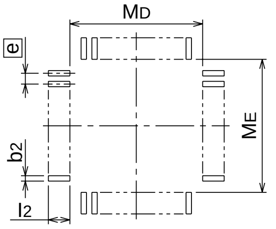


Symbol	Dimension in Millimeters		
	Min	Nom	Max
A	–	–	1.7
A_1	0.05	0.125	0.2
A_2	–	1.4	–
b	0.17	0.22	0.27
c	0.105	0.125	0.175
D	19.9	20.0	20.1
E	19.9	20.0	20.1
e	–	0.5	–
H_D	21.8	22.0	22.2
H_E	21.8	22.0	22.2
L	0.35	0.5	0.65
L_1	–	1.0	–
Lp	0.45	0.6	0.75
A_3	–	0.25	–
x	–	–	0.08
y	–	–	0.1
θ	0°	–	8°
b_2	–	0.225	–
I_2	0.95	–	–
M_D	–	20.4	–
M_E	–	20.4	–

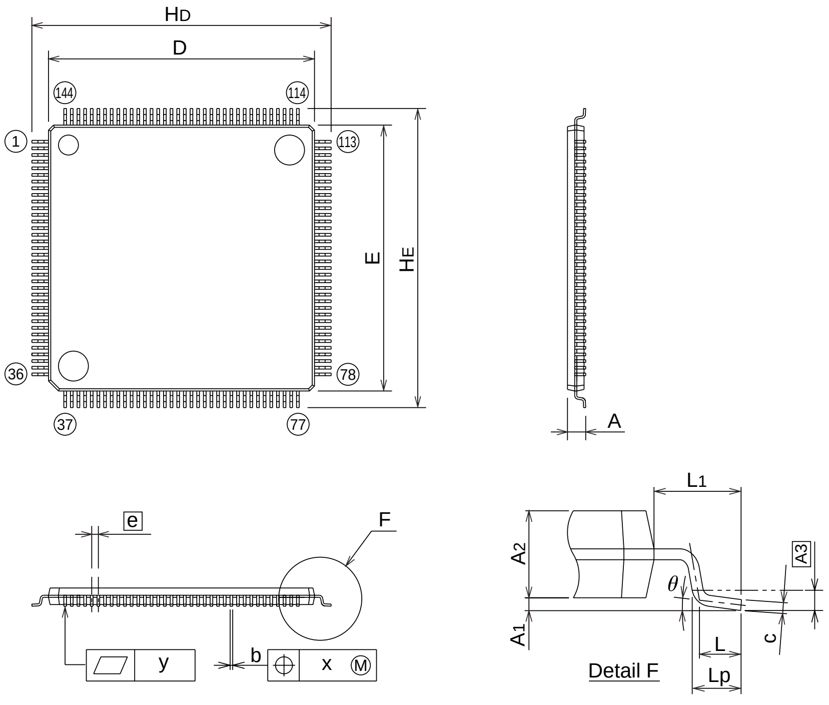
144PFB-A MMP

Plastic 144pin 16×16mm body TQFP

EIAJ Package Code	JEDEC Code	Weight(g)	Lead Material
TQFP144-P-1616-0.40	–	0.62	Cu Alloy



Recommended Mount Pad



Symbol	Dimension in Millimeters		
	Min	Nom	Max
A	–	–	1.2
A_1	0.05	0.1	0.15
A_2	–	1.0	–
b	0.13	0.18	0.23
c	0.105	0.125	0.175
D	15.9	16.0	16.1
E	15.9	16.0	16.1
e	–	0.4	–
H_D	17.8	18.0	18.2
H_E	17.8	18.0	18.2
L	0.4	0.5	0.6
L_1	–	1.0	–
Lp	0.45	0.6	0.75
A_3	–	0.25	–
x	–	–	0.07
y	–	–	0.08
θ	0°	–	8°
b_2	–	0.225	–
I_2	1.0	–	–
M_D	–	16.4	–
M_E	–	16.4	–

Rev.	Date	Description	
		Page	Summary
H	01/12/17	1	Features are partly revised.
		1	Applications is partly added.
		1	Page numbers of Table of Contents are partly revised.
		2	Figure 1.1.1 is partly revised.
		4	Table 1.1.1 is partly revised.
		5	Figure 1.1.3 is partly revised.
		5	Figure 1.1.4 is partly revised.
		6	Pin description is partly revised.
		8	Figure 1.4.1 is partly revised.
		12	Figure 1.6.1 is partly added.
		13	Figure 1.6.3 is partly revised.
		15	Figure 1.7.1 is partly revised. Note is added.
		16	Figure 1.7.2 is partly revised. Note 2 is added.
		17	Figure 1.7.3 is partly revised. Note is added.
		18	Processor mode register 0 in Figure 1.8.1 is partly revised.
		23	System clock control register in Figure 1.9.4 is partly revised. Note 8 is partly revised.
		25	Explanation of “Wait Mode” is partly revised.
		32	Explanation of “Hardware Interrupts” is partly revised.
		34	Table 1.10.2 is partly revised. Note 3 and note 4 are partly revised.
		42	Explanation of “Saving Registers is” partly revised. Note is partly revised.
		44	Figure 1.10.9 is partly revised.
		47	Explanation of “Key Input Interrupt” is partly revised.
		47	Figure 1.10.13 is partly revised.
		48	Figure 1.10.14 and 1.10.15 are partly revised.
		50	Explanation of “(3) The $\overline{\text{NMI}}$ interrupt” is partly revised.
		53	Explanation of “Watchdog timer” is partly added.
		56	Table 1.12.1 is partly revised.
		63	Explanation of “(1) Interrupt factors” is partly revised.
		67	Figure 1.13.3 is partly revised.
		68	Timer Ai register in Figure 1.13.5 is partly revised.
		69	Up/down flag 0 and Up/down flag 1 in Figure 1.13.6 is partly revised. Note is added.
		73	Table 1.13.2 is partly revised.
		73	Figure 1.13.10 is partly revised.
		74	Table 1.13.3 is partly revised.
		75	Figure 1.13.11 is partly revised.
		77	Table 1.13.5 is partly revised.
		78	Figure 1.13.14 and Figure 1.13.15 are partly revised.
		86	Figure 1.14.2 and Figure 1.14.3 are partly revised.
		91	UARTi transmit buffer register in Figure1.15.4 is partly revised. Note is added.
		91	UARTi bit rate generator in Figure1.15.4 is partly revised. Note is added.
		123	Explanation of “LCD Drive Control Circuit” is partly revised.
		125	LCD mode register in Figure 1.16.2 is partly revised.
		126	Explanation of “Voltage Multiplier” is partly revised.
		126	Figure 1.16.3 is partly revised.
		130	Table 1.17.1 is partly revised.
		139	Explanation of “Sample and hold” is partly revised.
		147	Port P13 direction register in Figure 1.19.6 is partly revised. Note is deleted.
		148	Port P7 register in Figure 1.19.7 is partly revised. Note is added.
		148	Port P13 register in Figure 1.19.7 is partly revised. Note is deleted.

Rev.	Date	Description	
		Page	Summary
H	01/12/17	149	Figure 1.19.8 is partly revised. Note is added.
		150	Figure 1.19.9 and Figure 1.19.10 are partly revised.
		151	Table 1.19.1 is partly revised.
		151	Figure 1.19.8 is partly revised.
		153	Explanation of “Serial I/O usage precaution” is added.
		153	Explanation of “Stop Mode and Wait Mode usage precaution” is partly revised.
		154	Explanation of “(3) The $\overline{\text{NMI}}$ interrupt” is partly revised.
		158	Table 1.21.4 is partly revised.
		163	Table 1.21.19 is partly revised.
		164	Table 1.21.20 is partly revised.
		168-170	MASK ROM CONFIRMATION FORM is added.
		171	Table 1.22.1 is partly revised.
		172	Explanation of “Flash Memory” is partly revised.
		172	Figure 1.22.1 is partly revised.
		173	Explanation of “Block Address” is partly revised.
		177	Explanation of “Writing in the user ROM area” is partly revised.
		178	Table 1.23.1 is partly revised. Note 4 is partly revised.
		182	Figure 1.23.5 is partly revised.
		183	Explanation of “ROM code protect function” is partly revised.
		186	Pin description (Flash memory standard serial I/O mode) is partly revised.
		187	Figure 1.25.1 is partly revised.
		191	Explanation of “Page Read Command” is partly revised.
		193	Explanation of “Block Erase Command” is partly revised.
		195	Explanation of “Boot ROM Area Output Command” is partly revised.
		200	Figure 1.25.14 is partly revised.
		205	Explanation of “Page Read Command” is partly revised.
		207	Explanation of “Block Erase Command” is partly revised.
		209	Explanation of “Boot ROM Area Output Command” is partly revised.
		214-215	Timing (Flash memory version) is added.
		216	Package is added.

Keep safety first in your circuit designs!

- Mitsubishi Electric Corporation puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage. Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of non-flammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

- These materials are intended as a reference to assist our customers in the selection of the Mitsubishi semiconductor product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Mitsubishi Electric Corporation or a third party.
- Mitsubishi Electric Corporation assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
- All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Mitsubishi Electric Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Mitsubishi Electric Corporation or an authorized Mitsubishi Semiconductor product distributor for the latest product information before purchasing a product listed herein.

  The information described here may contain technical inaccuracies or typographical errors. Mitsubishi Electric Corporation assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.

  Please also pay attention to information published by Mitsubishi Electric Corporation by various means, including the Mitsubishi Semiconductor home page (http://www.mitsubishichips.com).
- When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Mitsubishi Electric Corporation assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
- Mitsubishi Electric Corporation semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Mitsubishi Electric Corporation or an authorized Mitsubishi Semiconductor product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
- The prior written approval of Mitsubishi Electric Corporation is necessary to reprint or reproduce in whole or in part these materials.
- If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.

  Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
- Please contact Mitsubishi Electric Corporation or an authorized Mitsubishi Semicon ductor product distributor for further details on these materials or the products con tained therein.

MITSUBISHI SEMICONDUCTORS
M30220 Group Specification REV.H

Dec. First Edition 2001

Editioned by
Committee of editing of Mitsubishi Semiconductor

Published by
Mitsubishi Electric Corp., Kitaitami Works

This book, or parts thereof, may not be reproduced in any form without permission of Mitsubishi Electric Corporation.
©2001 MITSUBISHI ELECTRIC CORPORATION