

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

MITSUBISHI 16-BIT SINGLE-CHIP MICROCOMPUTER  
M16C FAMILY / M16C/20 SERIES

**M30218**  
Group

User's Manual

<http://www.infocom.maec.co.jp/indexe.htm>

Before using this material, please visit the above website to confirm that this is the most current document available.

### 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 Semiconductor product distributor for further details on these materials or the products contained therein.

## **Preface**

This user's manual describes the function and features of the Mitsubishi M30218 Group CMOS 16-bit microcomputer. The software features are explained to help designers take full advantage of the M16C functions.

For details about the software, please refer to the "M16C/60, M16C/20 series software manual", and for the development support tools, please refer to the related instruction manual.

# How to Use This Manual

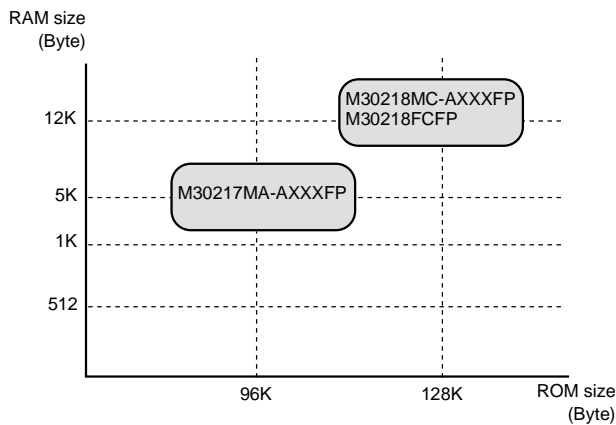
This user's manual is written for the M30218 group.

The reader of this manual is expected to have the basic knowledge of electric and logic circuits and microcomputers.

This manual is for the use of the models below.

- M30217MA-AXXXFP
- M30218MC-AXXXFP
- M30218FCFP

These products have similar features except for the memories, which differ from one product to another. This manual gives descriptions of M30218MC-AXXXFP. Memories built-in are as shown below. Be careful when writing a program, as the memories have different capacities.



The figure of each register configuration describes its functions, contents at reset, and attributes as follows :

- Bit attribute

R.....Read

W.....Write

O.....Possible to read

O.....Possible to write

X.....Impossible to read

X.....Impossible to write

One-shot start flag

Bit symbol	Bit name	Function	Bit attribute
b7	TA0OS	Timer A0 one-shot start flag	O
b6	TA1OS	Timer A1 one-shot start flag	O
b5	TA2OS	Timer A2 one-shot start flag	O
b4	TA3OS	Timer A3 one-shot start flag	O
b3	TA4OS	Timer A4 one-shot start flag	O
b2	Nothing is assigned. In an attempt to write to these bits, write "0". The value, if read, turns out to be indeterminate.		X
b1	TA0TGL	Timer A0 event/trigger select bit b7 b6 0 0 : Input on TA0in is selected (Note) 0 1 : TB2 overflow is selected 1 0 : TA4 overflow is selected 1 1 : TA1 overflow is selected	O
b0	TA0TGH		O

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

Note: Set the corresponding port direction register to "0".  
When TAIin is selected, TAIOUT assigned on same pin can not be used. (i=0 to 4)

This manual comprises of five chapters. Use the suggested chapters as a reference for the following topics:

- \* To understand hardware specifications ..... Chapter 1 Hardware
- \* To understand the basic way of using peripheral features and the operation timing ..... Chapter 2 Peripheral Functions Usage
- \* To observe applications of peripheral features ..... Chapter 3 Examples of Peripheral Functions Applications
- \* To understand interrupt timing in detail ..... Chapter 4 Interrupts
- \* To understand standard data ..... Chapter 5 Standard Characteristics

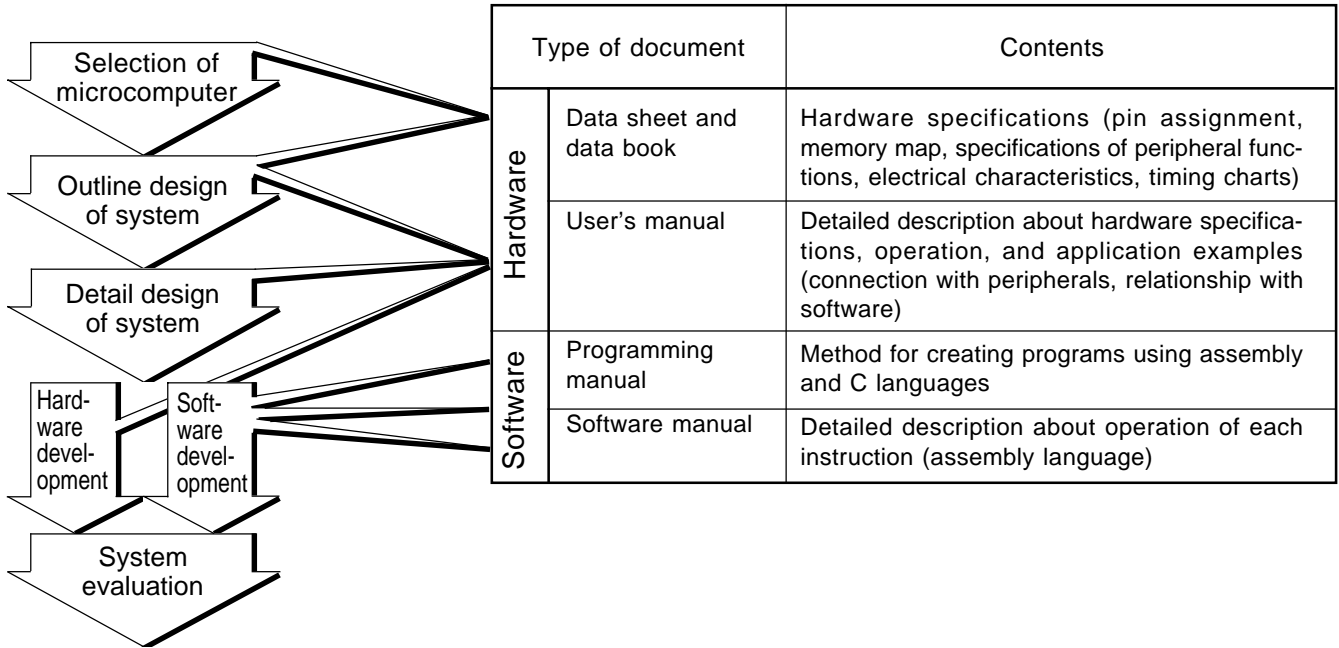
This manual includes a quick reference immediately following the Table of Contents, indicate the page of the topic to be pursued.

- \* To find a page describing a specific register by the register address ..... Quick Reference to Pages Classified by Address

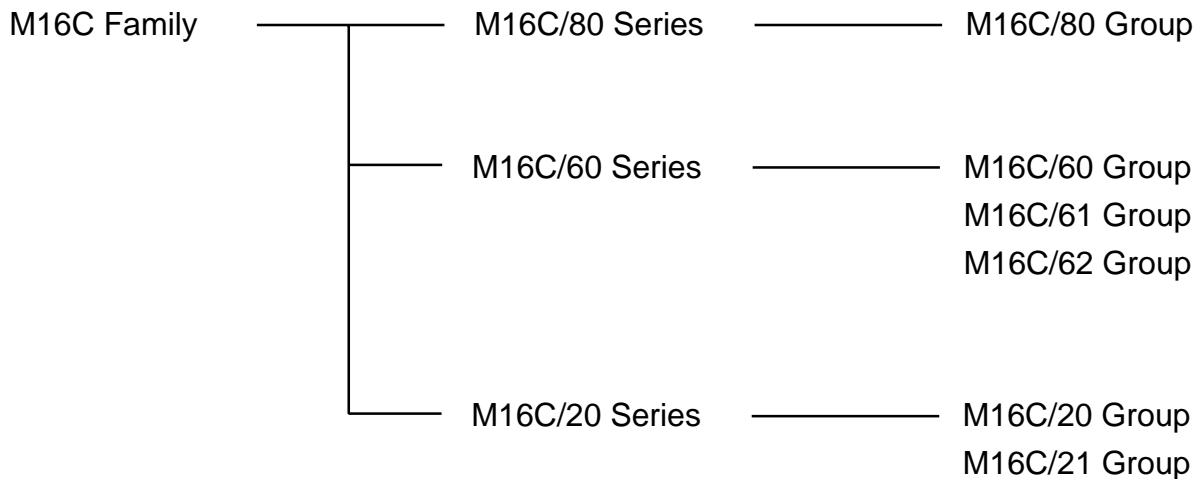
# M16C Family-related document list

## Usages

(Microcomputer development flow)



## M16C Family Line-up



# Table of Contents

## Chapter 1 Hardware

---

Description .....	2
Pin Description .....	7
Memory .....	9
Central Processing Unit (CPU) .....	12
Reset .....	15
Software Reset .....	18
Clock Generating Circuit .....	19
Clock Output .....	23
Stop Mode .....	23
Wait Mode .....	23
Status Transition Of BCLK .....	24
Power Control .....	25
Protection .....	27
Overview of Interrupt .....	28
Watchdog Timer .....	46
DMAC .....	48
FLD Controller .....	54
Timer .....	71
Timer A .....	72
Timer B .....	82
Serial I/O .....	88
Serial I/O2 .....	102
A-D Converter .....	115
D-A Converter .....	125
CRC Calculation Circuit .....	127
Programmable I/O Ports .....	129
Exclusive High-breakdown-voltage Output Ports .....	129
Flash Memory .....	154



## Chapter 2 Peripheral Functions Usage

---

2.1 Protect .....	178
2.1.1 Overview .....	178
2.1.2 Protect Operation .....	178
2.2 Timer A .....	180
2.2.1 Overview .....	180
2.2.2 Operation of Timer A (timer mode) .....	186
2.2.3 Operation of Timer A (timer mode, gate function selected) .....	188
2.2.4 Operation of Timer A (timer mode, pulse output function selected) .....	190
2.2.5 Operation of Timer A (event counter mode, reload type selected) .....	192
2.2.6 Operation of Timer A (event counter mode, free run type selected) .....	194
2.2.7 Operation of timer A (2-phase pulse signal process in event counter mode, normal mode selected) .....	196
2.2.8 Operation of timer A (2-phase pulse signal process in event counter mode, multiply-by-4 mode selected) .....	198
2.2.9 Operation of Timer A (one-shot timer mode) .....	200
2.2.10 Operation of Timer A (one-shot timer mode, external trigger selected) .....	202
2.2.11 Operation of Timer A (pulse width modulation mode, 16-bit PWM mode selected) .....	204
2.2.12 Operation of Timer A (pulse width modulation mode, 8-bit PWM mode selected) .....	206
2.2.13 Precautions for Timer A (timer mode) .....	208
2.2.14 Precautions for Timer A (event counter mode) .....	209
2.2.15 Precautions for Timer A (one-shot timer mode) .....	210
2.2.16 Precautions for Timer A (pulse width modulation mode) .....	211
2.3 Timer B .....	212
2.3.1 Overview .....	212
2.3.2 Operation of Timer B (timer mode) .....	216
2.3.3 Operation of Timer B (event counter mode) .....	218
2.3.4 Operation of Timer B (pulse period measurement mode) .....	220
2.3.5 Operation of Timer B (pulse width measurement mode) .....	222
2.3.6 Precautions for Timer B (timer mode, event counter mode) .....	224
2.3.7 Precautions for Timer B (pulse period/pulse width measurement mode) .....	225
2.4 Clock-Synchronous Serial I/O .....	226
2.4.1 Overview .....	226
2.4.2 Operation of Serial I/O (transmission in clock-synchronous serial I/O mode) .....	232
2.4.3 Operation of the Serial I/O (transmission in clock-synchronous serial I/O mode, transfer clock	

output from multiple pins function selected) .....	236
2.4.4 Operation of Serial I/O (reception in clock-synchronous serial I/O mode) .....	240
2.4.5 Precautions for Serial I/O (in clock-synchronous serial I/O) .....	244
2.5 Clock-Asynchronous Serial I/O (UART) .....	246
2.5.1 Overview .....	246
2.5.2 Operation of Serial I/O (transmission in UART mode) .....	254
2.5.3 Operation of Serial I/O (reception in UART mode) .....	258
2.6 Serial I/O2 .....	262
2.6.1 Overview .....	262
2.6.2 Serial I/O2 connection examples .....	267
2.6.3 Serial I/O2 modes .....	269
2.6.4 Serial I/O2 Operations (transmission in 8-bit serial I/O mode) .....	270
2.6.5 Serial I/O2 Operations (transmission/reception in automatic transfer serial I/O mode) .....	274
2.6.6 Serial I/O2 Operations (transmission/reception in automatic transfer serial I/O mode, using handshake signal) .....	278
2.6.7 Precautions for Serial I/O2 .....	282
2.7 FLD (VFD) Controller .....	286
2.7.1 Overview .....	286
2.7.2 FLD operation (FLD automatic display and key-scan using segments) .....	296
2.7.3 FLD operation (FLD automatic display and key-scan using digits) .....	302
2.7.4 FLD operation (FLD display and key-scan using segment by software) .....	306
2.7.5 FLD operation (Display with digit expander M35501FP) .....	312
2.7.6 FLD operation (Display with digit expander M35501FP: column discrepancy) .....	318
2.7.7 Precautions for FLD controller .....	325
2.8 A-D Converter .....	326
2.8.1 Overview .....	326
2.8.2 Operation of A-D converter (one-shot mode) .....	332
2.8.3 Operation of A-D Converter (in repeat mode) .....	334
2.8.4 Operation of A-D Converter (in single sweep mode) .....	336
2.8.5 Operation of A-D Converter (in repeat sweep mode 0) .....	338
2.8.6 Operation of A-D Converter (in repeat sweep mode 1) .....	340
2.8.7 Precautions for A-D Converter .....	342
2.8.8 Method of A-D Conversion (10-bit mode) .....	343
2.8.9 Method of A-D Conversion (8-bit mode) .....	345
2.8.10 Absolute Accuracy and Differential Non-Linearity Error .....	347

2.8.11 Internal Equivalent Circuit of Analog Input .....	349
2.8.12 Sensor's Output Impedance under A-D Conversion .....	350
2.9 D-A Converter .....	352
2.9.1 Overview .....	352
2.9.2 D-A Converter Operation .....	353
2.10 DMAC .....	354
2.10.1 Overview .....	354
2.10.2 Operation of DMAC (one-shot transfer mode) .....	358
2.10.3 Operation of DMAC (repeated transfer mode) .....	360
2.11 CRC Calculation Circuit .....	362
2.11.1 Overview .....	362
2.11.2 Operation of CRC Calculation Circuit .....	363
2.12 Watchdog Timer .....	364
2.12.1 Overview .....	364
2.12.2 Operation of Watchdog Timer .....	366
2.13 Address Match Interrupt .....	368
2.13.1 Overview .....	368
2.13.2 Operation of Address Match Interrupt .....	370
2.14 Power Control .....	372
2.14.1 Overview .....	372
2.14.2 Stop Mode Set-Up .....	377
2.14.3 Wait Mode Set-Up .....	378
2.14.4 Precautions in Power Control .....	379
2.15 Programmable I/O Ports .....	380
2.15.1 Overview .....	380

## **Chapter 3 Examples of Peripheral functions Applications \_\_\_\_\_**

3.1 Long-Period Timers .....	388
3.2 Variable-Period Variable-Duty PWM Output .....	392
3.3 Delayed One-Shot Output .....	396
3.4 Buzzer Output .....	400
3.5 Solution for External Interrupt Pins Shortage .....	402
3.6 Memory to Memory DMA Transfer .....	404
3.7 Controlling Power Using Stop Mode .....	408
3.8 Controlling Power Using Wait Mode .....	412

## Chapter 4 Interrupt

---

4.1 Overview of Interrupt .....	408
4.1.1 Type of Interrupts .....	408
4.1.2 Software Interrupts .....	409
4.1.3 Hardware Interrupts .....	420
4.1.4 Interrupts and Interrupt Vector Tables .....	421
4.2 Interrupt Control .....	423
4.2.1 Interrupt Enable Flag .....	425
4.2.2 Interrupt Request Bit .....	425
4.2.3 Interrupt Priority Level Select Bit and Processor Interrupt Priority Level (IPL) .....	426
4.2.4 Rewrite the interrupt control register .....	427
4.3 Interrupt Sequence .....	428
4.3.1 Interrupt Response Time .....	428
4.3.2 Variation of IPL when Interrupt Request is Accepted .....	429
4.3.3 Saving Registers .....	430
4.4 Returning from an Interrupt Routine .....	432
4.5 Interrupt Priority .....	432
4.6 Multiple Interrupts .....	434
4.7 Precautions for Interrupts .....	436

## Chapter 5 Standard Characteristics

---

5.1 Standard DC Characteristics .....	440
5.1.1 Standard Ports Characteristics .....	440
5.1.2 Characteristics of ICC-f(XIN) .....	444
5.2 Standard Characteristics of A-D Converter .....	446
5.3 Standard Characteristics of D-A Converter .....	448
5.4 Standard Characteristics of Pull-Up Resistor .....	451

This page kept blank for layout purposes.

## Quick Reference to Pages Classified by Address

Address	Register	Page	Address	Register	Page	
0000 <sub>16</sub>			0040 <sub>16</sub>			
0001 <sub>16</sub>			0041 <sub>16</sub>			
0002 <sub>16</sub>			0042 <sub>16</sub>			
0003 <sub>16</sub>			0043 <sub>16</sub>			
0004 <sub>16</sub>	Processor mode register 0 (PM0)	18	0044 <sub>16</sub>			
0005 <sub>16</sub>	Processor mode register 1 (PM1)			0045 <sub>16</sub>		
0006 <sub>16</sub>	System clock control register 0 (CM0)	22	0046 <sub>16</sub>			
0007 <sub>16</sub>	System clock control register 1 (CM1)			0047 <sub>16</sub>	INT3 interrupt control register (INT3IC)	34
0008 <sub>16</sub>			0048 <sub>16</sub>	INT4 interrupt control register (INT4IC)		
0009 <sub>16</sub>	Address match interrupt enable register (AIER)	43	0049 <sub>16</sub>	INT5 interrupt control register (INT5IC)		
000A <sub>16</sub>	Protect register (PRCR)	27	004A <sub>16</sub>			
000B <sub>16</sub>			004B <sub>16</sub>	DMA0 interrupt control register (DM0IC)	34	
000C <sub>16</sub>			004C <sub>16</sub>	DMA1 interrupt control register (DM1IC)		
000D <sub>16</sub>			004D <sub>16</sub>			
000E <sub>16</sub>	Watchdog timer start register (WDTS)	47	004E <sub>16</sub>	A-D conversion interrupt control register (ADIC)	34	
000F <sub>16</sub>	Watchdog timer control register (WDC)			004F <sub>16</sub>		SI/O2 automatic transfer interrupt control register (ASIOIC)
0010 <sub>16</sub>			0050 <sub>16</sub>	FLD interrupt control register (FLDIC)		
0011 <sub>16</sub>	Address match interrupt register 0 (RMAD0)	43	0051 <sub>16</sub>	UART0 transmit interrupt control register (S0TIC)		
0012 <sub>16</sub>			0052 <sub>16</sub>	UART0 receive interrupt control register (S0RIC)		
0013 <sub>16</sub>			0053 <sub>16</sub>	UART1 transmit interrupt control register (S1TIC)		
0014 <sub>16</sub>			0054 <sub>16</sub>	UART1 receive interrupt control register (S1RIC)		
0015 <sub>16</sub>	Address match interrupt register 1 (RMAD1)	43	0055 <sub>16</sub>	Timer A0 interrupt control register (TA0IC)		
0016 <sub>16</sub>			0056 <sub>16</sub>	Timer A1 interrupt control register (TA1IC)		
0017 <sub>16</sub>			0057 <sub>16</sub>	Timer A2 interrupt control register (TA2IC)		
0018 <sub>16</sub>			0058 <sub>16</sub>	Timer A3 interrupt control register (TA3IC)		
0019 <sub>16</sub>			0059 <sub>16</sub>	Timer A4 interrupt control register (TA4IC)		
001A <sub>16</sub>			005A <sub>16</sub>	Timer B0 interrupt control register (TB0IC)		
001B <sub>16</sub>			005B <sub>16</sub>	Timer B1 interrupt control register (TB1IC)		
001C <sub>16</sub>			005C <sub>16</sub>	Timer B2 interrupt control register (TB2IC)		
001D <sub>16</sub>			005D <sub>16</sub>	INT0 interrupt control register (INT0IC)		
001E <sub>16</sub>			005E <sub>16</sub>	INT1 interrupt control register (INT1IC)		
001F <sub>16</sub>			005F <sub>16</sub>	INT2 interrupt control register (INT2IC)		
0020 <sub>16</sub>			≈	≈	≈	
0021 <sub>16</sub>	DMA0 source pointer (SAR0)	51	0340 <sub>16</sub>	Serial I/O2 automatic transfer data pointer (SIO2DP)	105	
0022 <sub>16</sub>			0341 <sub>16</sub>			
0023 <sub>16</sub>			0342 <sub>16</sub>	Serial I/O2 control register 1 (SIO2CON1)	104	
0024 <sub>16</sub>			0343 <sub>16</sub>			
0025 <sub>16</sub>	DMA0 destination pointer (DAR0)	51	0344 <sub>16</sub>	Serial I/O2 control register 2 (SIO2CON2)	104	
0026 <sub>16</sub>			0345 <sub>16</sub>			
0027 <sub>16</sub>			0346 <sub>16</sub>	Serial I/O2 register / transfer counter (SIO2)	105	
0028 <sub>16</sub>	DMA0 transfer counter (TCR0)	51	0347 <sub>16</sub>			
0029 <sub>16</sub>			0348 <sub>16</sub>	Serial I/O2 control register 3 (SIO2CON3)	105	
002A <sub>16</sub>			0349 <sub>16</sub>			
002B <sub>16</sub>			034A <sub>16</sub>			
002C <sub>16</sub>	DMA0 control register (DM0CON)	50	034B <sub>16</sub>			
002D <sub>16</sub>			034C <sub>16</sub>			
002E <sub>16</sub>			034D <sub>16</sub>			
002F <sub>16</sub>			034E <sub>16</sub>			
0030 <sub>16</sub>			034F <sub>16</sub>			
0031 <sub>16</sub>	DMA1 source pointer (SAR1)	51	0350 <sub>16</sub>	FLD mode register (FLDM)	56	
0032 <sub>16</sub>			0351 <sub>16</sub>	FLD output control register (FLDCON)		
0033 <sub>16</sub>			0352 <sub>16</sub>	Tdisp time set register (TDISP)		
0034 <sub>16</sub>			0353 <sub>16</sub>			
0035 <sub>16</sub>	DMA1 destination pointer (DAR1)	51	0354 <sub>16</sub>	Toff1 time set register (TOFF1)	57	
0036 <sub>16</sub>			0355 <sub>16</sub>			
0037 <sub>16</sub>			0356 <sub>16</sub>	Toff2 time set register (TOFF2)	57	
0038 <sub>16</sub>			0357 <sub>16</sub>			
0039 <sub>16</sub>	DMA1 transfer counter (TCR1)	51	0358 <sub>16</sub>	FLD data pointer (FLDDP)	57	
003A <sub>16</sub>			0359 <sub>16</sub>	P2 FLD/port switch register (P2FPR)		
003B <sub>16</sub>			035A <sub>16</sub>	P3 FLD/port switch register (P3FPR)		
003C <sub>16</sub>	DMA1 control register (DM1CON)	50	035B <sub>16</sub>	P4 FLD/port switch register (P4FPR)	58	
003D <sub>16</sub>			035C <sub>16</sub>	P5 digit output set register (P5DOR)		
003E <sub>16</sub>			035D <sub>16</sub>	P6 digit output set register (P6DOR)	59	
003F <sub>16</sub>			035E <sub>16</sub>			
			035F <sub>16</sub>			

## Quick Reference to Pages Classified by Address

Address	Register	Page	Address	Register	Page
0380 <sub>16</sub>	Count start flag (TABSR)	73	03C0 <sub>16</sub>	A-D register 0 (AD0)	118
0381 <sub>16</sub>	Clock prescaler reset flag (CPSRF)	74	03C1 <sub>16</sub>	A-D register 1 (AD1)	
0382 <sub>16</sub>	One-shot start flag (ONSF)		03C2 <sub>16</sub>	A-D register 1 (AD1)	
0383 <sub>16</sub>	Trigger select register (TRGSR)		03C3 <sub>16</sub>	A-D register 1 (AD1)	
0384 <sub>16</sub>	Up-down flag (UDF)	73	03C4 <sub>16</sub>	A-D register 2 (AD2)	
0385 <sub>16</sub>			03C5 <sub>16</sub>	A-D register 2 (AD2)	
0386 <sub>16</sub>	Timer A0 (TA0)	73	03C6 <sub>16</sub>	A-D register 3 (AD3)	
0387 <sub>16</sub>			03C7 <sub>16</sub>	A-D register 3 (AD3)	
0388 <sub>16</sub>	Timer A1 (TA1)		03C8 <sub>16</sub>	A-D register 4 (AD4)	
0389 <sub>16</sub>			03C9 <sub>16</sub>	A-D register 4 (AD4)	
038A <sub>16</sub>	Timer A2 (TA2)		03CA <sub>16</sub>	A-D register 5 (AD5)	
038B <sub>16</sub>			03CB <sub>16</sub>	A-D register 5 (AD5)	
038C <sub>16</sub>	Timer A3 (TA3)		03CC <sub>16</sub>	A-D register 6 (AD6)	
038D <sub>16</sub>		03CD <sub>16</sub>	A-D register 6 (AD6)		
038E <sub>16</sub>	Timer A4 (TA4)	83	03CE <sub>16</sub>	A-D register 7 (AD7)	
038F <sub>16</sub>			03CF <sub>16</sub>	A-D register 7 (AD7)	
0390 <sub>16</sub>	Timer B0 (TB0)		03D0 <sub>16</sub>		
0391 <sub>16</sub>			03D1 <sub>16</sub>		
0392 <sub>16</sub>	Timer B1 (TB1)		03D2 <sub>16</sub>		
0393 <sub>16</sub>			03D3 <sub>16</sub>		
0394 <sub>16</sub>	Timer B2 (TB2)		03D4 <sub>16</sub>	A-D control register 2 (ADCON2)	
0395 <sub>16</sub>		03D5 <sub>16</sub>			
0396 <sub>16</sub>	Timer A0 mode register (TA0MR)	72	03D6 <sub>16</sub>	A-D control register 0 (ADCON0)	
0397 <sub>16</sub>	Timer A1 mode register (TA1MR)		03D7 <sub>16</sub>	A-D control register 1 (ADCON1)	
0398 <sub>16</sub>	Timer A2 mode register (TA2MR)		03D8 <sub>16</sub>	D-A register 0 (DA0)	
0399 <sub>16</sub>	Timer A3 mode register (TA3MR)		03D9 <sub>16</sub>		
039A <sub>16</sub>	Timer A4 mode register (TA4MR)		03DA <sub>16</sub>	D-A register 1 (DA1)	
039B <sub>16</sub>	Timer B0 mode register (TB0MR)	82	03DB <sub>16</sub>		
039C <sub>16</sub>	Timer B1 mode register (TB1MR)		03DC <sub>16</sub>	D-A control register (DACON)	
039D <sub>16</sub>	Timer B2 mode register (TB2MR)		03DD <sub>16</sub>		
039E <sub>16</sub>			03DE <sub>16</sub>		
039F <sub>16</sub>			03DF <sub>16</sub>		
03A0 <sub>16</sub>	UART0 transmit/receive mode register (U0MR)	91	03E0 <sub>16</sub>	Port P0 (P0)	
03A1 <sub>16</sub>	UART0 bit rate generator (U0BRG)	90	03E1 <sub>16</sub>	Port P1 (P1)	
03A2 <sub>16</sub>	UART0 transmit buffer register (U0TB)		03E2 <sub>16</sub>		
03A3 <sub>16</sub>			03E3 <sub>16</sub>		
03A4 <sub>16</sub>	UART0 transmit/receive control register 0 (U0C0)	91	03E4 <sub>16</sub>	Port P2 (P2)	
03A5 <sub>16</sub>	UART0 transmit/receive control register 1 (U0C1)	92	03E5 <sub>16</sub>	Port P3 (P3)	
03A6 <sub>16</sub>	UART0 receive buffer register (U0RB)	90	03E6 <sub>16</sub>		
03A7 <sub>16</sub>			03E7 <sub>16</sub>	Port P3 direction register (PD3)	
03A8 <sub>16</sub>	UART1 transmit/receive mode register (U1MR)	91	03E8 <sub>16</sub>	Port P4 (P4)	
03A9 <sub>16</sub>	UART1 bit rate generator (U1BRG)	90	03E9 <sub>16</sub>	Port P5 (P5)	
03AA <sub>16</sub>	UART1 transmit buffer register (U1TB)		03EA <sub>16</sub>	Port P4 direction register (PD4)	
03AB <sub>16</sub>			03EB <sub>16</sub>		
03AC <sub>16</sub>	UART1 transmit/receive control register 0 (U1C0)	91	03EC <sub>16</sub>	Port P6 (P6)	
03AD <sub>16</sub>	UART1 transmit/receive control register 1 (U1C1)	92	03ED <sub>16</sub>	Port P7 (P7)	
03AE <sub>16</sub>	UART1 receive buffer register (U1RB)	90	03EE <sub>16</sub>		
03AF <sub>16</sub>			03EF <sub>16</sub>	Port P7 direction register (PD7)	
03B0 <sub>16</sub>	UART transmit/receive control register 2 (UCON)	92	03F0 <sub>16</sub>	Port P8 (P8)	
03B1 <sub>16</sub>			03F1 <sub>16</sub>	Port P9 (P9)	
03B2 <sub>16</sub>			03F2 <sub>16</sub>	Port P8 direction register (PD8)	
03B3 <sub>16</sub>			03F3 <sub>16</sub>	Port P9 direction register (PD9)	
03B4 <sub>16</sub>	Flash memory control register 0 (FCON0) (Note)	155	03F4 <sub>16</sub>	Port P10 (P10)	
03B5 <sub>16</sub>	Flash memory control register 1 (FCON1) (Note)		03F5 <sub>16</sub>		
03B6 <sub>16</sub>	Flash command register (FCMD) (Note)		03F6 <sub>16</sub>	Port P10 direction register (PD10)	
03B7 <sub>16</sub>			03F7 <sub>16</sub>		
03B8 <sub>16</sub>	DMA0 request cause select register (DM0SL)	50	03F8 <sub>16</sub>		
03B9 <sub>16</sub>			03F9 <sub>16</sub>		
03BA <sub>16</sub>	DMA1 request cause select register (DM1SL)	50	03FA <sub>16</sub>		
03BB <sub>16</sub>			03FB <sub>16</sub>		
03BC <sub>16</sub>	CRC data register (CRCD)	127	03FC <sub>16</sub>		
03BD <sub>16</sub>			03FD <sub>16</sub>	Pull-up control register 0 (PUR0)	
03BE <sub>16</sub>	CRC input register (CRCIN)		03FE <sub>16</sub>	Pull-up control register 1 (PUR1)	
03BF <sub>16</sub>			03FF <sub>16</sub>		

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

# Chapter 1

---

Hardware



## Description

**Description**

The M30218 group of single-chip microcomputers are built using the high-performance silicon gate CMOS process using a M16C/60 Series CPU core and are packaged in a 100-pin plastic molded QFP. These single-chip microcomputers operate using sophisticated instructions featuring a high level of instruction efficiency. With 1M bytes of address space, they are capable of executing instructions at high speed. They also feature a built-in multiplier and DMAC, making them ideal for controlling musical instruments, household appliances and other high-speed processing applications.

The M30218 group includes a wide range of products with different internal memory types and sizes and various package types.

**Features**

- Basic machine instructions ..... Compatible with the M16C/60 series
- Memory capacity ..... ROM / RAM (See figure memory expansion)
- Shortest instruction execution time . 100ns ( $f(X_{IN})=10\text{MHz}$ )
- Supply voltage ..... 4.0V to 5.5V ( $f(X_{IN})=10\text{MHz}$ )  
2.7V to 5.5V ( $f(X_{IN})=3.5\text{MHz}$ )(Note)
- Interrupts ..... 19 internal and 6 external interrupt sources, 4 software
- Multifunction 16-bit timer ..... Timer A X 5, Timer B X 3
- FLD controller ..... total 56 pins  
(high-breakdown-voltage P-channel open-drain output : 52pins)
- Serial I/O ..... 2 channels for UART or clock synchronous,  
1 channels for clock synchronous  
(max.256 bytes automatic transfer function)
- DMAC ..... 2 channels (triggers: 15 sources)
- A-D converter ..... 10 bits X 8 channels
- D-A converter ..... 8 bits X 2 channels
- CRC calculation circuit ..... 1 circuit
- Watchdog timer ..... 1 pin
- Programmable I/O ..... 48 pins
- High-breakdown-voltage output ..... 52 pins
- Clock generating circuit ..... 2 built-in clock generation circuit  
(built-in feedback resistor, and external ceramic or quartz oscillator)

Note: Only mask ROM version.

**Applications**

Household appliances, office equipment, Audio etc.

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

Central Processing Unit (CPU) .....	12	Timer .....	71
Reset.....	15	Serial I/O .....	88
Clock Generating Circuit .....	19	A-D Converter .....	115
Protection .....	27	D-A Converter .....	125
Interrupts .....	28	CRC Calculation Circuit .....	127
Watchdog Timer .....	46	Programmable I/O Ports .....	129
DMAC .....	48	Flash memory .....	154
FLD controller .....	54		

Description

Pin Configuration

Figure 1 shows the pin configuration (top view).

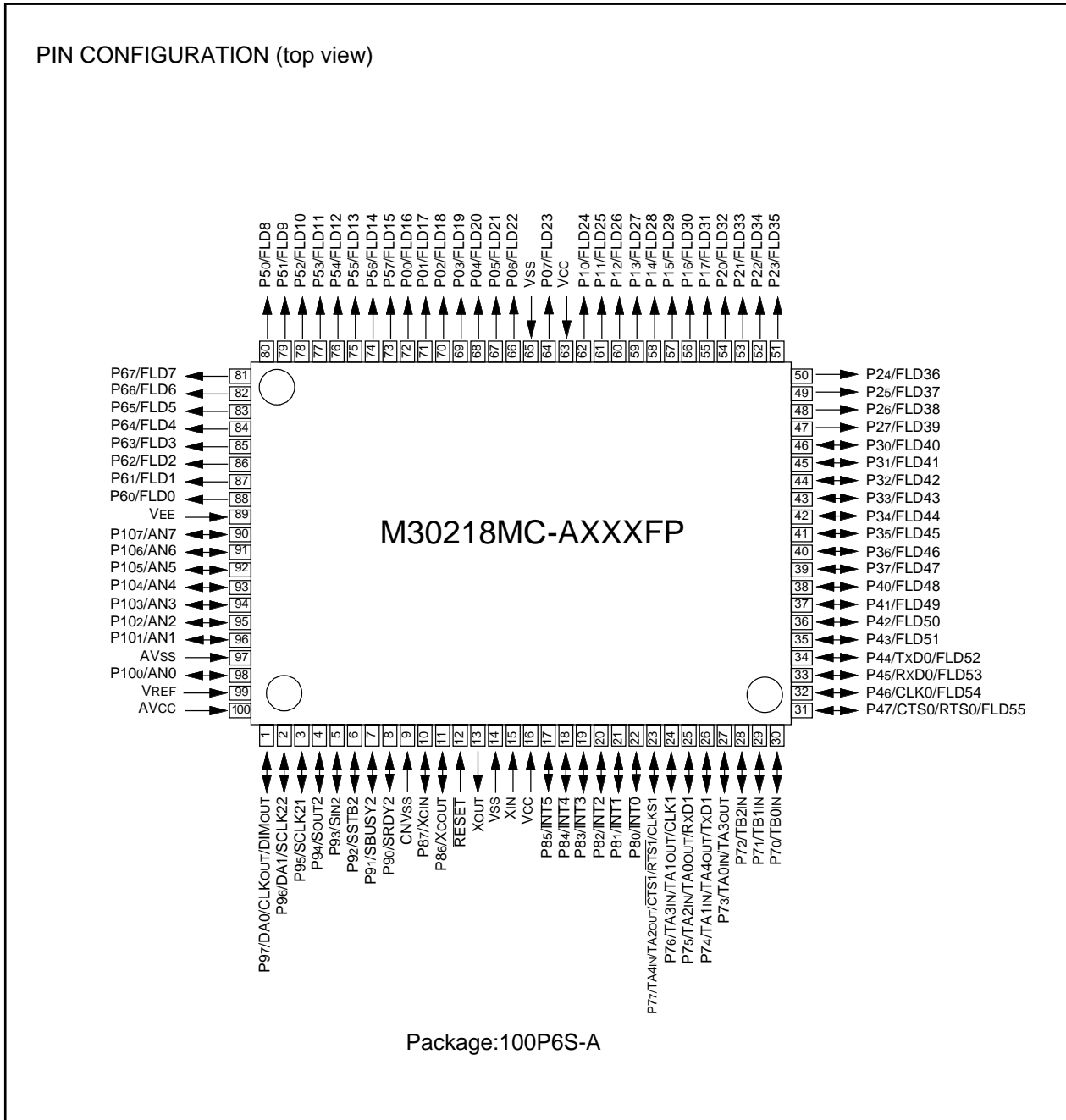


Figure 1. Pin configuration (top view)

## Description

## Block Diagram

Figure 2 shows a block diagram of the M30218 group.

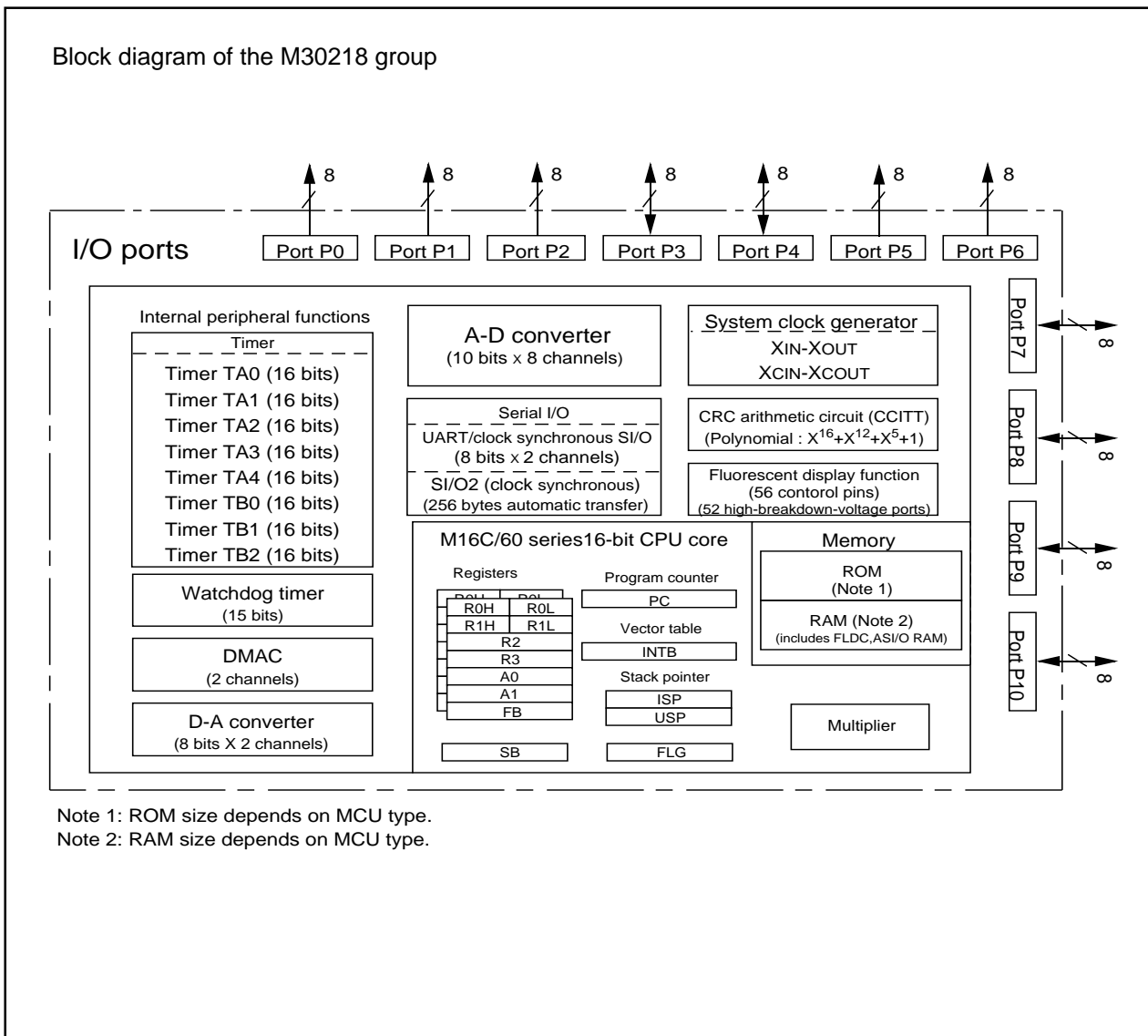


Figure 2. Block diagram of M30218 group

## Description

## Performance Outline

Table 1 shows a performance outline of M30218 group.

**Table 1. Performance outline of M30218 group**

Item		Performance	
Number of basic instructions		91 instructions	
Shortest instruction execution time		100ns(f(XIN)=10MHz)	
Memory capacity	ROM	See figure memory expansion	
	RAM	See figure memory expansion	
I/O port	P3, P4, P7 to P10	8 bits x 6	
Output port	P0 to P2, P5, P6	8 bit x 5	
Multifunction timer	TA0, TA1, TA2, TA3, TA4	16 bits x 5	
	TB0, TB1, TB2	16 bits x 3	
Serial I/O	UART0, UART1	(UART or clock synchronous) x 2	
	SI/O2	(Clock synchronous) x 1 (with automatic transfer function)	
Fluorescent display		56 pins	
A-D converter		10 bits x 8 channels	
D-A converter		8 bits x 2	
DMAC		2 channels (triggers :15 sources)	
CRC calculation circuit		1 circuit (polynomial: $X^{16} + X^{12} + X^5 + 1$ )	
Watchdog timer		15 bits x 1 (with prescaler)	
Interrupt		19 internal and 6 external sources, 4 software sources, 7 levels	
Clock generating circuit		2 built-in clock generation circuits (built-in feedback resistor, and external ceramic or quartz oscillator)	
Supply voltage		4.0 to 5.5V (f(XIN)=10MHz) 2.7 to 5.5V (f(XIN)=3.5MHz) (Note)	
Power consumption		18 mW (VCC=3V, f(XIN)=5MHz)	
I/O characteristics	I/O withstand voltage		
	VCC-48V (output ports : P0 to P2, P5, P6, I/O ports : P3, P40 to P43) 0 to VCC (I/O ports :P44 to P47, P7 to P10)		
	Output current	H	- 18mA (P0 to P3, P40 to P43, P5, P6) :high-breakdown-voltage, P-channel open-drain
		L	- 5mA (P44 to P47, P7 to P10) 5mA (P44 to P47, P7 to P10)
Operating ambient temperature		-20 to 85°C	
Device configuration		CMOS silicon gate	
Package		100-pin plastic mold QFP	

**Note: Only mask ROM version.**

## Description

Mitsubishi plans to release the following products in the M30218 group:

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

(2) Memory capacity

(3) Package

100P6S : Plastic molded QFP (mask ROM version and flash memory version)

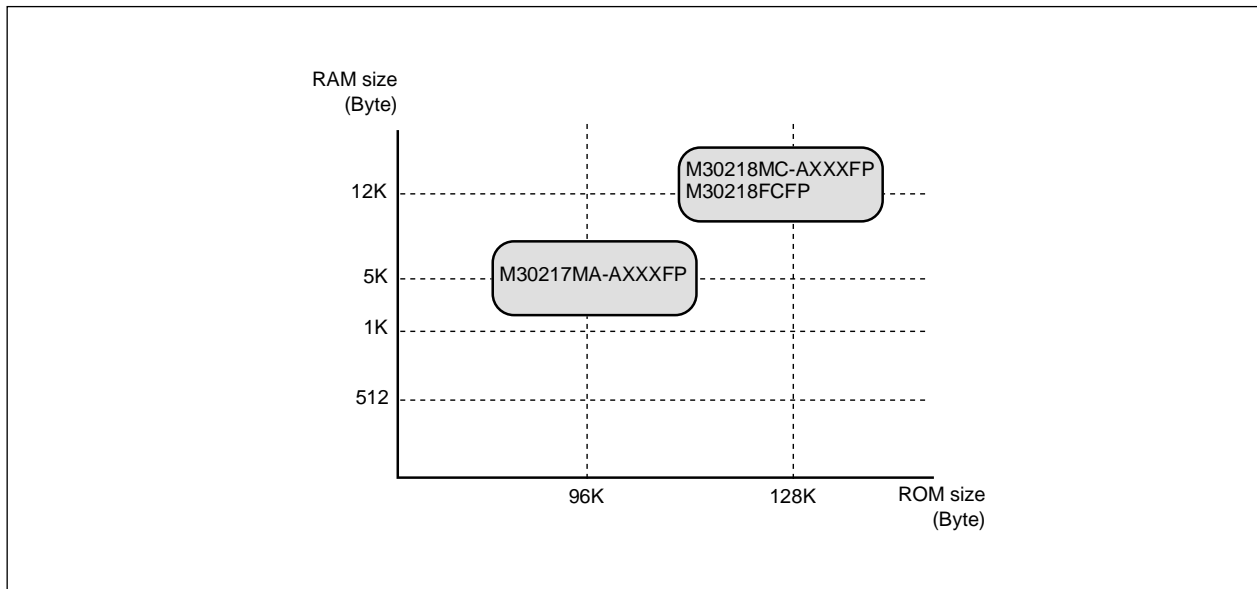


Figure 3. ROM expansion

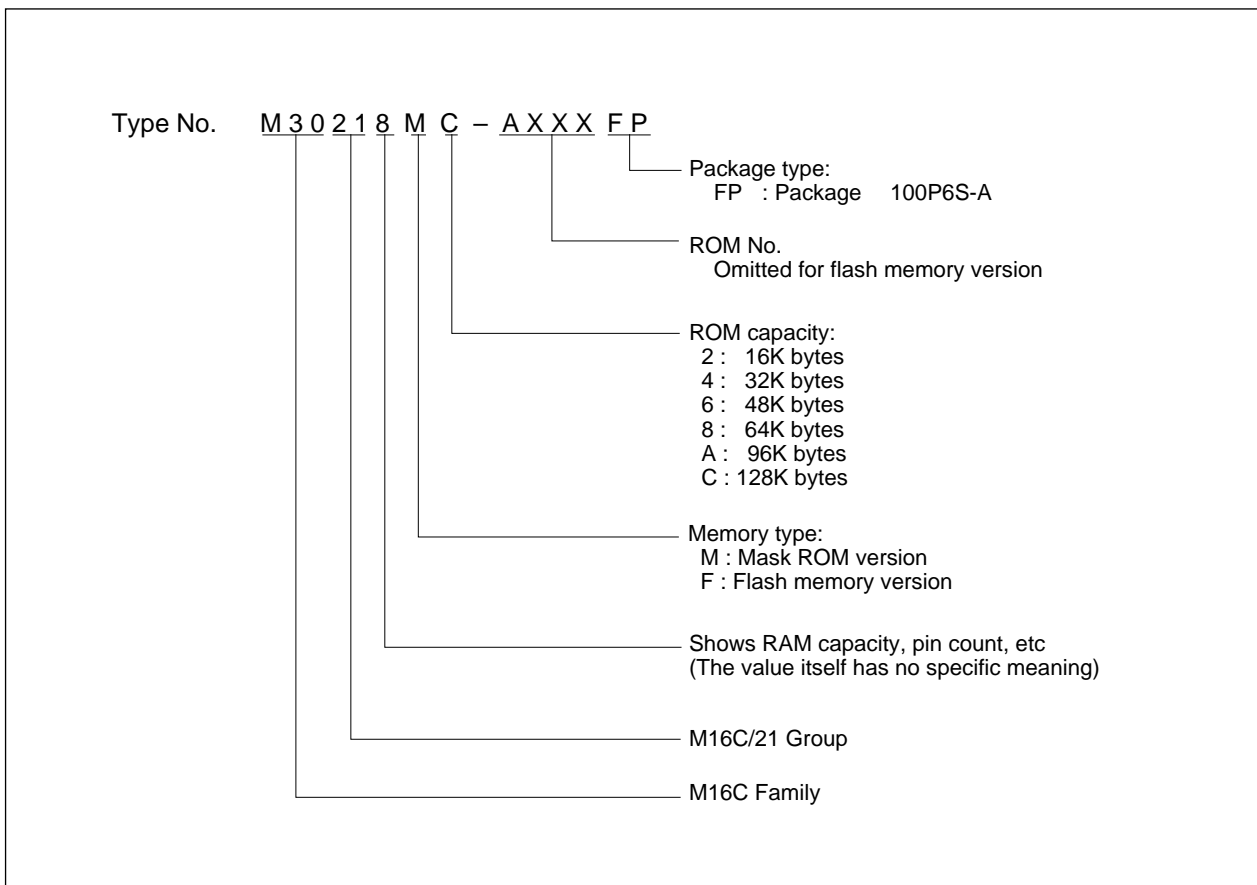


Figure 4. Type No., memory size, and package

## Pin Description

## Pin Description

Pin name	Signal name	I/O type	Function
Vcc, Vss	Power supply input		Supply 2.7V(Note1) to 5.5 V to the Vcc pin. Supply 0 V to the Vss pin. Connect a bypass capacitor across the Vcc pin and Vss pin.
CNVss	CNVss	Input	Connect it to the Vss pin.
$\overline{\text{RESET}}$	Reset input	Input	A "L" on this input resets the microcomputer.
XIN XOUT	Clock input Clock output	Input Output	These pins are provided for the main clock generating circuit. Connect a ceramic resonator or crystal between the XIN and the XOUT pins. To use an externally derived clock, input it to the XIN pin and leave the XOUT pin open.
AVcc	Analog power supply input		This pin is a power supply input for the A-D converter. Connect this pin to Vcc.
AVss	Analog power supply input		This pin is a power supply input for the A-D converter. Connect this pin to Vss.
VREF	Reference voltage input	Input	This pin is a reference voltage input for the A-D converter.
VEE	pull-down power source		Apply voltage supplied to pull-down resistors of ports P0 to P1, P5, P6.
P00/FLD <sub>16</sub> to P07/FLD <sub>23</sub>	Output port P0	Output	This is an 8-bit CMOS output port and high-breakdown-voltage P-channel open-drain output structure. A pull-down resistor is built in between port P0 and VEE pin. At reset, this port is set to VEE level. P0 function as FLD controller output pins as selected by software.
P10/FLD <sub>24</sub> to P17/FLD <sub>31</sub>	Output port P1	Output	This is an 8-bit output port equivalent to P0. Pins in this port also function as FLD controller output pins as selected by software.
P20/FLD <sub>32</sub> to P27/FLD <sub>39</sub>	Output port P2	Output	This is an 8-bit output port equivalent to P0. A pull-down resistor is not built in between P2 and VEE pin. Pins in this port also function as FLD controller output pins as selected by software.
P30/FLD <sub>40</sub> to P37/FLD <sub>47</sub>	I/O port P3	Input/output	This is an 8-bit I/O port. A pull-down resistor is not built in between P3 and VEE pin. It has an input/output port direction register that allows the user to set each pin for input or output. This is low-voltage input level, and high-breakdown-voltage P-channel open-drain output structure. Pins in this port also function as FLD controller output pins as selected by software.
P40/FLD <sub>48</sub> to P47/FLD <sub>56</sub>	I/O port P4	Input/output	This is an 8-bit I/O port equivalent to P3. This is low-voltage input level. P4 <sub>0</sub> to P4 <sub>3</sub> is high-breakdown-voltage P-channel open-drain output structure, P4 <sub>4</sub> to P4 <sub>7</sub> is CMOS output. A pull-down resistor is not built in between P4(P4 <sub>0</sub> to P4 <sub>3</sub> ) and VEE pin. Pins in this port also function as FLD controller output pins as selected by software. P4 <sub>4</sub> to P4 <sub>7</sub> also function as UART0 I/O pins as selected by software. When set for input, the user can specify in units of four bits by software whether or not they are tied to a pull-up resistor.
P50/FLD <sub>8</sub> to P57/FLD <sub>15</sub>	Output port P5	Output	This is an 8-bit output port equivalent to P0. Pins in this port also function as FLD controller output pins as selected by software.
P60/FLD <sub>0</sub> to P67/FLD <sub>7</sub>	Output port P6	Output	This is an 8-bit output port equivalent to P0. Pins in this port also function as FLD controller output pins as selected by software.

## Pin Description

## Pin Description

Pin name	Signal name	I/O type	Function
P7 <sub>0</sub> to P7 <sub>7</sub>	I/O port P7	Input/output	This is an 8-bit I/O port equivalent to P3. This is CMOS input/output. When set for input, the user can specify in units of four bits by software whether or not they are tied to a pull-up resistor. P7 <sub>0</sub> to P7 <sub>2</sub> function as TimerB0 to B2 input pins as selected by software. P7 <sub>3</sub> function as TimerA0 I/O pin as selected by software. P7 <sub>4</sub> to P7 <sub>7</sub> function as TimerA1 to A4 I/O pins, and UART1 I/O pins as selected by software.
P8 <sub>0</sub> to P8 <sub>7</sub>	I/O port P8	Input/output	This is an 8-bit I/O port equivalent to P7. When set for input, the user can specify in units of four bits by software whether or not they are tied to a pull-up resistor. P8 <sub>0</sub> to P8 <sub>5</sub> function as external interrupt input pins as selected by software. P8 <sub>6</sub> , P8 <sub>7</sub> function as sub-clock input pin as selected by software. In this case, connect a quartz oscillator between P8 <sub>6</sub> (XOUT pin) and P8 <sub>7</sub> (XCIN pin)
P9 <sub>0</sub> to P9 <sub>7</sub>	I/O port P9	Input/output	This is an 8-bit I/O port equivalent to P7. When set for input, the user can specify in units of four bits by software whether or not they are tied to a pull-up resistor. P9 <sub>7</sub> function as D-A converter output pins, clock output pins (same frequency of X <sub>IN</sub> /8, X <sub>IN</sub> /32 or X <sub>CIN</sub> ) and DIM signal output pin of FLD controller as selected by software. P9 <sub>6</sub> function as D-A converter output pins and clock I/O pin of serial I/O with automatic transfer as selected by software. P9 <sub>0</sub> to P9 <sub>5</sub> function as I/O pin of serial I/O with automatic transfer as selected by software.
P10 <sub>0</sub> to P10 <sub>7</sub>	I/O port P10	Input/output	This is an 8-bit I/O port equivalent to P7. When set for input, the user can specify in units of four bits by software whether or not they are tied to a pull-up resistor. Pins in this port also function as A-D converter input pins as selected by software.

Note 1: Supply 4.0V to 5.5V to the Vcc pin in flash memory version.

## Memory

## Operation of Functional Blocks

The M30218 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, FLD controller, serial I/O, D-A converter, DMAC, CRC calculation circuit, A-D converter, and I/O ports.

The following explains each unit.

## Memory

Figure 5 is a memory map of the M30218 group. The address space extends the 1M bytes from address  $00000_{16}$  to  $FFFFFF_{16}$ . From  $FFFFFF_{16}$  down is ROM. For example, in the M30218MC-AXXXFP, there is 128K bytes of internal ROM from  $E0000_{16}$  to  $FFFFFF_{16}$ . The vector table for fixed interrupts such as the reset are mapped to  $FFDC_{16}$  to  $FFFFFF_{16}$ . The starting address of the interrupt routine is stored here. The address of the vector table for timer interrupts, etc., can be set as desired using the internal register (INTB). See the section on interrupts for details.

From  $00400_{16}$  up is RAM. For example, in the M30218MC-AXXXFP, there is 12K bytes of internal RAM from  $00400_{16}$  to  $033FF_{16}$ . In addition to storing data, the RAM also stores the stack used when calling subroutines and when interrupts are generated. (From  $00400_{16}$  to  $004FF_{16}$  is RAM for SIO2. From  $00500_{16}$  to  $005DF_{16}$  is RAM for FLD.)

The SFR area is mapped to  $00000_{16}$  to  $003FF_{16}$ . This area accommodates the control registers for peripheral devices such as I/O ports, A-D converter, serial I/O, and timers, etc. 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  $FFDB_{16}$ . 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.

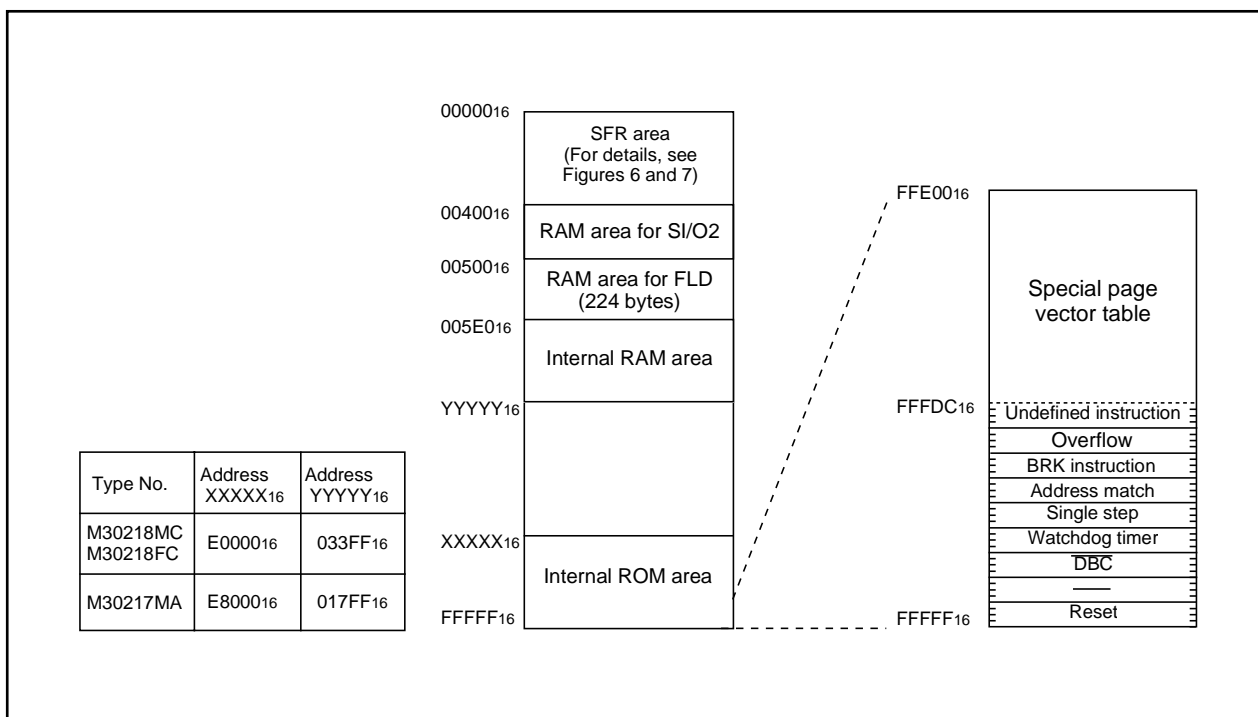


Figure 5. Memory map



## Memory

0000 <sub>16</sub>		0040 <sub>16</sub>	
0001 <sub>16</sub>		0041 <sub>16</sub>	
0002 <sub>16</sub>		0042 <sub>16</sub>	
0003 <sub>16</sub>		0043 <sub>16</sub>	
0004 <sub>16</sub>	Processor mode register 0 (PM0)	0044 <sub>16</sub>	
0005 <sub>16</sub>	Processor mode register 1 (PM1)	0045 <sub>16</sub>	
0006 <sub>16</sub>	System clock control register 0 (CM0)	0046 <sub>16</sub>	
0007 <sub>16</sub>	System clock control register 1 (CM1)	0047 <sub>16</sub>	INT3 interrupt control register (INT3IC)
0008 <sub>16</sub>		0048 <sub>16</sub>	INT4 interrupt control register (INT4IC)
0009 <sub>16</sub>	Address match interrupt enable register (AIER)	0049 <sub>16</sub>	INT5 interrupt control register (INT5IC)
000A <sub>16</sub>	Protect register (PRCR)	004A <sub>16</sub>	
000B <sub>16</sub>		004B <sub>16</sub>	DMA0 interrupt control register (DM0IC)
000C <sub>16</sub>		004C <sub>16</sub>	DMA1 interrupt control register (DM1IC)
000D <sub>16</sub>		004D <sub>16</sub>	
000E <sub>16</sub>	Watchdog timer start register (WDTS)	004E <sub>16</sub>	A-D conversion interrupt control register (ADIC)
000F <sub>16</sub>	Watchdog timer control register (WDC)	004F <sub>16</sub>	SI/O automatic transfer interrupt control register (ASIOIC)
0010 <sub>16</sub>		0050 <sub>16</sub>	FLD interrupt control register (FLDIC)
0011 <sub>16</sub>	Address match interrupt register 0 (RMAD0)	0051 <sub>16</sub>	UART0 transmit interrupt control register (S0TIC)
0012 <sub>16</sub>		0052 <sub>16</sub>	UART0 receive interrupt control register (S0RIC)
0013 <sub>16</sub>		0053 <sub>16</sub>	UART1 transmit interrupt control register (S1TIC)
0014 <sub>16</sub>		0054 <sub>16</sub>	UART1 receive interrupt control register (S1RIC)
0015 <sub>16</sub>	Address match interrupt register 1 (RMAD1)	0055 <sub>16</sub>	Timer A0 interrupt control register (TA0IC)
0016 <sub>16</sub>		0056 <sub>16</sub>	Timer A1 interrupt control register (TA1IC)
0017 <sub>16</sub>		0057 <sub>16</sub>	Timer A2 interrupt control register (TA2IC)
0018 <sub>16</sub>		0058 <sub>16</sub>	Timer A3 interrupt control register (TA3IC)
0019 <sub>16</sub>		0059 <sub>16</sub>	Timer A4 interrupt control register (TA4IC)
001A <sub>16</sub>		005A <sub>16</sub>	Timer B0 interrupt control register (TB0IC)
001B <sub>16</sub>		005B <sub>16</sub>	Timer B1 interrupt control register (TB1IC)
001C <sub>16</sub>		005C <sub>16</sub>	Timer B2 interrupt control register (TB2IC)
001D <sub>16</sub>		005D <sub>16</sub>	INT0 interrupt control register (INT0IC)
001E <sub>16</sub>		005E <sub>16</sub>	INT1 interrupt control register (INT1IC)
001F <sub>16</sub>		005F <sub>16</sub>	INT2 interrupt control register (INT2IC)
0020 <sub>16</sub>		0340 <sub>16</sub>	Serial I/O2 automatic transfer data pointer (SIO2DP)
0021 <sub>16</sub>	DMA0 source pointer (SAR0)	0341 <sub>16</sub>	
0022 <sub>16</sub>		0342 <sub>16</sub>	Serial I/O2 control register 1 (SIO2CON1)
0023 <sub>16</sub>		0343 <sub>16</sub>	
0024 <sub>16</sub>		0344 <sub>16</sub>	Serial I/O2 control register 2 (SIO2CON2)
0025 <sub>16</sub>	DMA0 destination pointer (DAR0)	0345 <sub>16</sub>	
0026 <sub>16</sub>		0346 <sub>16</sub>	Serial I/O2 register / transfer counter (SIO2)
0027 <sub>16</sub>		0347 <sub>16</sub>	
0028 <sub>16</sub>	DMA0 transfer counter (TCR0)	0348 <sub>16</sub>	Serial I/O2 control register 3 (SIO2CON3)
0029 <sub>16</sub>		0349 <sub>16</sub>	
002A <sub>16</sub>		034A <sub>16</sub>	
002B <sub>16</sub>		034B <sub>16</sub>	
002C <sub>16</sub>	DMA0 control register (DM0CON)	034C <sub>16</sub>	
002D <sub>16</sub>		034D <sub>16</sub>	
002E <sub>16</sub>		034E <sub>16</sub>	
002F <sub>16</sub>		034F <sub>16</sub>	
0030 <sub>16</sub>		0350 <sub>16</sub>	FLD mode register (FLDM)
0031 <sub>16</sub>	DMA1 source pointer (SAR1)	0351 <sub>16</sub>	FLD output control register (FLDCON)
0032 <sub>16</sub>		0352 <sub>16</sub>	Tdisp time set register (TDISP)
0033 <sub>16</sub>		0353 <sub>16</sub>	
0034 <sub>16</sub>		0354 <sub>16</sub>	Toff1 time set register (TOFF1)
0035 <sub>16</sub>	DMA1 destination pointer (DAR1)	0355 <sub>16</sub>	
0036 <sub>16</sub>		0356 <sub>16</sub>	Toff2 time set register (TOFF2)
0037 <sub>16</sub>		0357 <sub>16</sub>	
0038 <sub>16</sub>		0358 <sub>16</sub>	FLD data pointer (FLDDP)
0039 <sub>16</sub>	DMA1 transfer counter (TCR1)	0359 <sub>16</sub>	P2 FLD/port switch register (P2FPR)
003A <sub>16</sub>		035A <sub>16</sub>	P3 FLD/port switch register (P3FPR)
003B <sub>16</sub>		035B <sub>16</sub>	P4 FLD/port switch register (P4FPR)
003C <sub>16</sub>	DMA1 control register (DM1CON)	035C <sub>16</sub>	P5 digit output set register (P5DOR)
003D <sub>16</sub>		035D <sub>16</sub>	P6 digit output set register (P6DOR)
003E <sub>16</sub>		035E <sub>16</sub>	
003F <sub>16</sub>		035F <sub>16</sub>	

Figure 6. Location of peripheral unit control registers (1)

## Memory

0380 <sub>16</sub>	Count start flag (TABSR)	03C0 <sub>16</sub>	A-D register 0 (AD0)
0381 <sub>16</sub>	Clock prescaler reset flag (CPSRF)	03C1 <sub>16</sub>	
0382 <sub>16</sub>	One-shot start flag (ONSF)	03C2 <sub>16</sub>	A-D register 1 (AD1)
0383 <sub>16</sub>	Trigger select register (TRGSR)	03C3 <sub>16</sub>	
0384 <sub>16</sub>	Up-down flag (UDF)	03C4 <sub>16</sub>	A-D register 2 (AD2)
0385 <sub>16</sub>		03C5 <sub>16</sub>	
0386 <sub>16</sub>	Timer A0 (TA0)	03C6 <sub>16</sub>	A-D register 3 (AD3)
0387 <sub>16</sub>		03C7 <sub>16</sub>	
0388 <sub>16</sub>	Timer A1 (TA1)	03C8 <sub>16</sub>	A-D register 4 (AD4)
0389 <sub>16</sub>		03C9 <sub>16</sub>	
038A <sub>16</sub>	Timer A2 (TA2)	03CA <sub>16</sub>	A-D register 5 (AD5)
038B <sub>16</sub>		03CB <sub>16</sub>	
038C <sub>16</sub>	Timer A3 (TA3)	03CC <sub>16</sub>	A-D register 6 (AD6)
038D <sub>16</sub>		03CD <sub>16</sub>	
038E <sub>16</sub>	Timer A4 (TA4)	03CE <sub>16</sub>	A-D register 7 (AD7)
038F <sub>16</sub>		03CF <sub>16</sub>	
0390 <sub>16</sub>	Timer B0 (TB0)	03D0 <sub>16</sub>	
0391 <sub>16</sub>		03D1 <sub>16</sub>	
0392 <sub>16</sub>	Timer B1 (TB1)	03D2 <sub>16</sub>	
0393 <sub>16</sub>		03D3 <sub>16</sub>	
0394 <sub>16</sub>	Timer B2 (TB2)	03D4 <sub>16</sub>	A-D control register 2 (ADCON2)
0395 <sub>16</sub>		03D5 <sub>16</sub>	
0396 <sub>16</sub>	Timer A0 mode register (TA0MR)	03D6 <sub>16</sub>	A-D control register 0 (ADCON0)
0397 <sub>16</sub>	Timer A1 mode register (TA1MR)	03D7 <sub>16</sub>	A-D control register 1 (ADCON1)
0398 <sub>16</sub>	Timer A2 mode register (TA2MR)	03D8 <sub>16</sub>	D-A register 0 (DA0)
0399 <sub>16</sub>	Timer A3 mode register (TA3MR)	03D9 <sub>16</sub>	
039A <sub>16</sub>	Timer A4 mode register (TA4MR)	03DA <sub>16</sub>	D-A register 1 (DA1)
039B <sub>16</sub>	Timer B0 mode register (TB0MR)	03DB <sub>16</sub>	
039C <sub>16</sub>	Timer B1 mode register (TB1MR)	03DC <sub>16</sub>	D-A control register (DACON)
039D <sub>16</sub>	Timer B2 mode register (TB2MR)	03DD <sub>16</sub>	
039E <sub>16</sub>		03DE <sub>16</sub>	
039F <sub>16</sub>		03DF <sub>16</sub>	
03A0 <sub>16</sub>	UART0 transmit/receive mode register (U0MR)	03E0 <sub>16</sub>	Port P0 (P0)
03A1 <sub>16</sub>	UART0 bit rate generator (U0BRG)	03E1 <sub>16</sub>	Port P1 (P1)
03A2 <sub>16</sub>		03E2 <sub>16</sub>	
03A3 <sub>16</sub>	UART0 transmit buffer register (U0TB)	03E3 <sub>16</sub>	
03A4 <sub>16</sub>	UART0 transmit/receive control register 0 (U0C0)	03E4 <sub>16</sub>	Port P2 (P2)
03A5 <sub>16</sub>	UART0 transmit/receive control register 1 (U0C1)	03E5 <sub>16</sub>	Port P3 (P3)
03A6 <sub>16</sub>		03E6 <sub>16</sub>	
03A7 <sub>16</sub>	UART0 receive buffer register (U0RB)	03E7 <sub>16</sub>	Port P3 direction register (PD3)
03A8 <sub>16</sub>	UART1 transmit/receive mode register (U1MR)	03E8 <sub>16</sub>	Port P4 (P4)
03A9 <sub>16</sub>	UART1 bit rate generator (U1BRG)	03E9 <sub>16</sub>	Port P5 (P5)
03AA <sub>16</sub>		03EA <sub>16</sub>	Port P4 direction register (PD4)
03AB <sub>16</sub>	UART1 transmit buffer register (U1TB)	03EB <sub>16</sub>	
03AC <sub>16</sub>	UART1 transmit/receive control register 0 (U1C0)	03EC <sub>16</sub>	Port P6 (P6)
03AD <sub>16</sub>	UART1 transmit/receive control register 1 (U1C1)	03ED <sub>16</sub>	Port P7 (P7)
03AE <sub>16</sub>		03EE <sub>16</sub>	
03AF <sub>16</sub>	UART1 receive buffer register (U1RB)	03EF <sub>16</sub>	Port P7 direction register (PD7)
03B0 <sub>16</sub>	UART transmit/receive control register 2 (UCON)	03F0 <sub>16</sub>	Port P8 (P8)
03B1 <sub>16</sub>		03F1 <sub>16</sub>	Port P9 (P9)
03B2 <sub>16</sub>		03F2 <sub>16</sub>	Port P8 direction register (PD8)
03B3 <sub>16</sub>		03F3 <sub>16</sub>	Port P9 direction register (PD9)
03B4 <sub>16</sub>	Flash memory control register 0 (FCON0) (Note)	03F4 <sub>16</sub>	Port P10 (P10)
03B5 <sub>16</sub>	Flash memory control register 1 (FCON1) (Note)	03F5 <sub>16</sub>	
03B6 <sub>16</sub>	Flash command register (FCMD) (Note)	03F6 <sub>16</sub>	Port P10 direction register (PD10)
03B7 <sub>16</sub>		03F7 <sub>16</sub>	
03B8 <sub>16</sub>	DMA0 request cause select register (DM0SL)	03F8 <sub>16</sub>	
03B9 <sub>16</sub>		03F9 <sub>16</sub>	
03BA <sub>16</sub>	DMA1 request cause select register (DM1SL)	03FA <sub>16</sub>	
03BB <sub>16</sub>		03FB <sub>16</sub>	
03BC <sub>16</sub>		03FC <sub>16</sub>	
03BD <sub>16</sub>	CRC data register (CRCD)	03FD <sub>16</sub>	Pull-up control register 0 (PUR0)
03BE <sub>16</sub>	CRC input register (CRCIN)	03FE <sub>16</sub>	Pull-up control register 1 (PUR1)
03BF <sub>16</sub>		03FF <sub>16</sub>	

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

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

## Central Processing Unit (CPU)

The CPU has a total of 13 registers shown in Figure 8. Seven of these registers (R0, R1, R2, R3, A0, A1, and FB) come in two sets; therefore, these have two register banks.

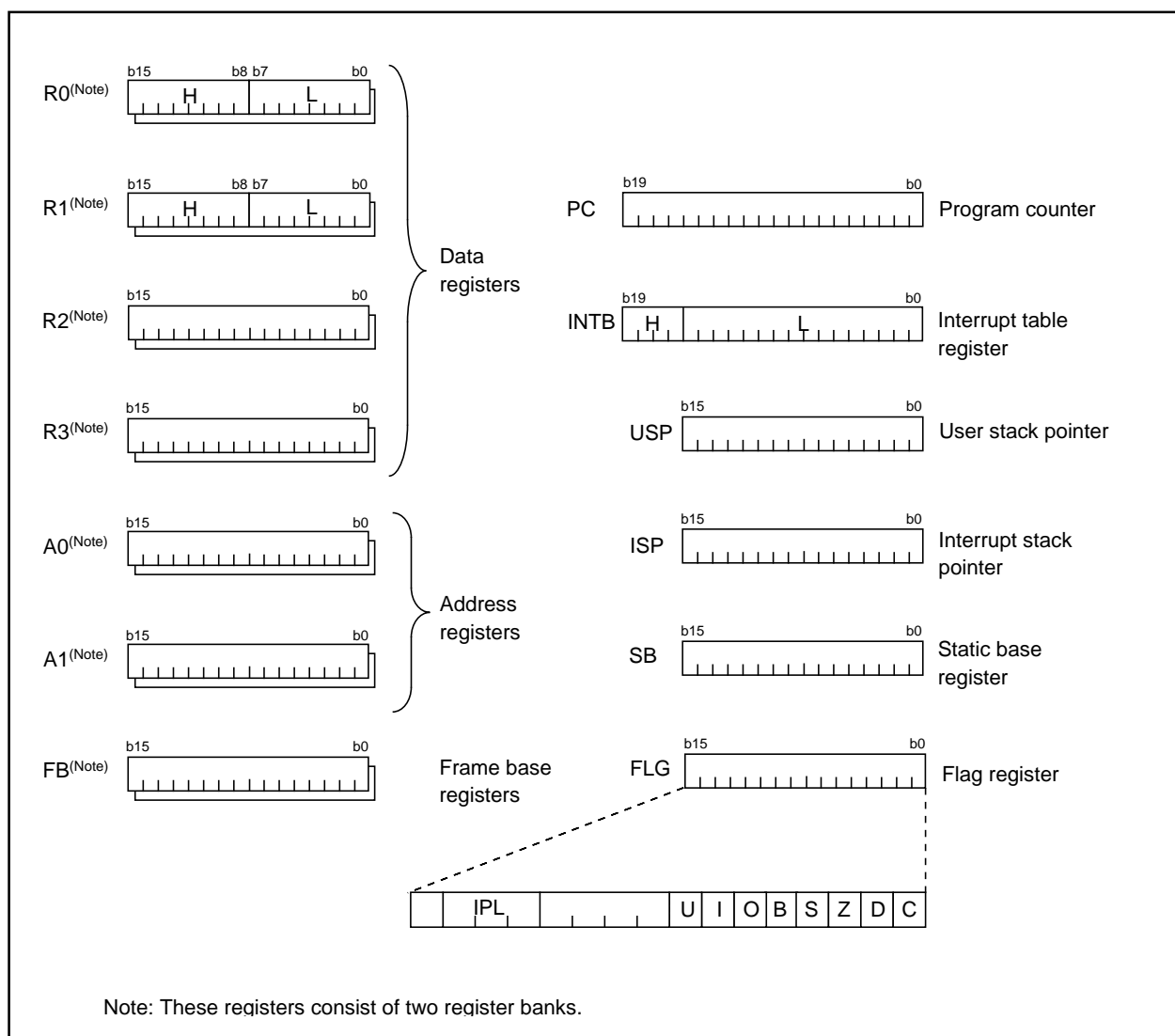


Figure 8. Central processing unit register

### (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 CA-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.

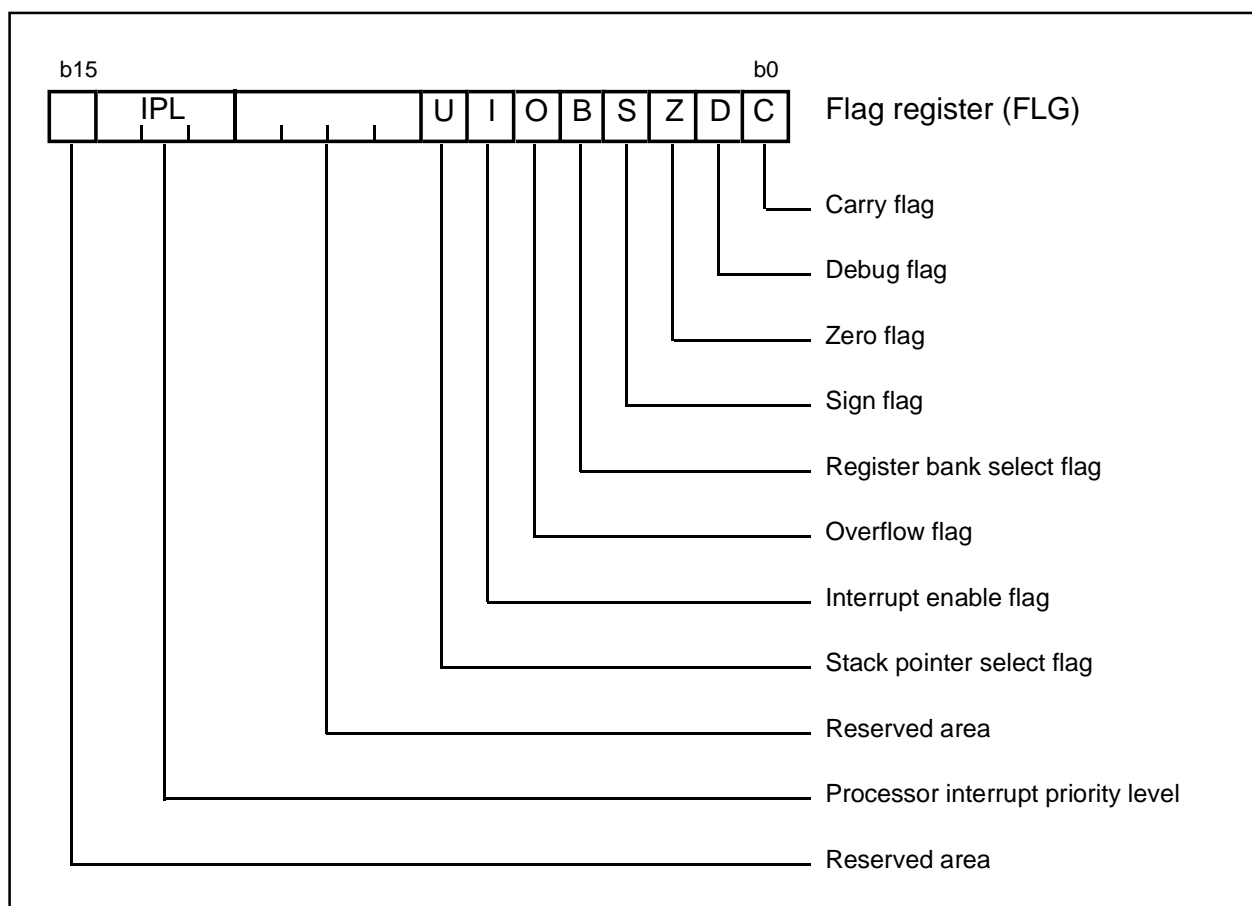


Figure 9. Flag register (FLG)

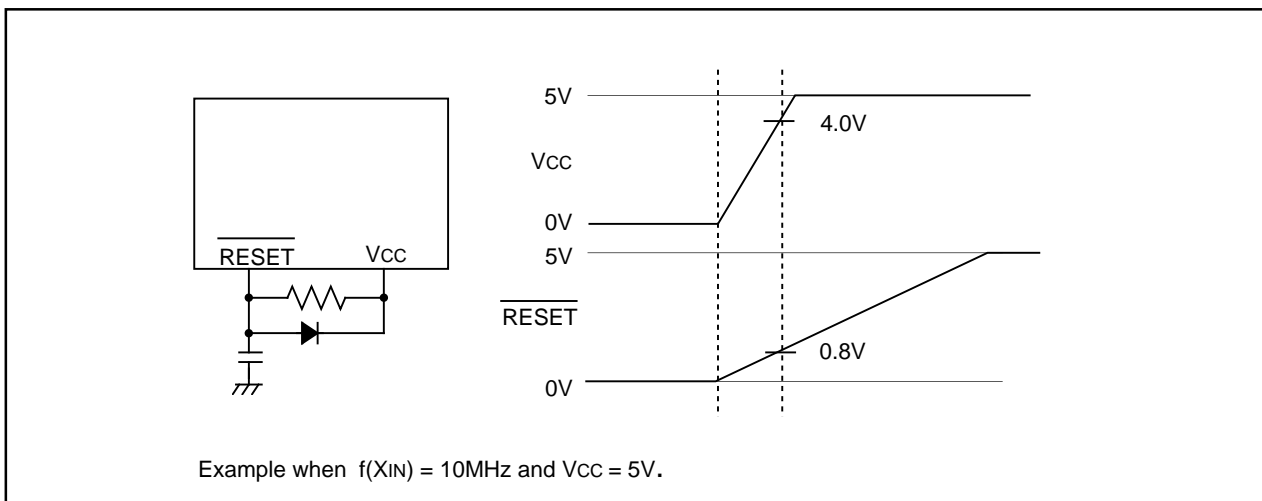
Reset

**Reset**

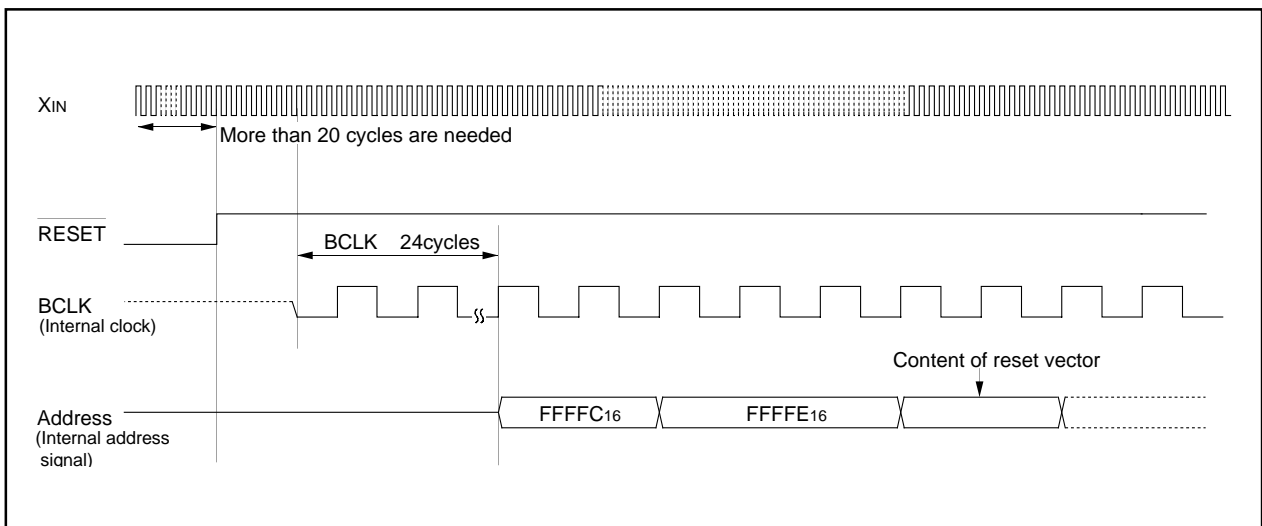
There are two kinds of resets; hardware and software. In both cases, operation is the same after the reset. (See "Software Reset" for details of software resets.) This section explains on hardware resets.

When the supply voltage is in the range where operation is guaranteed, a reset is effected by holding the reset pin level "L" (0.2V<sub>CC</sub> 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 10 shows the example reset circuit. Figure 11 shows the reset sequence.



**Figure 10. Example reset circuit**



**Figure 11. Reset sequence**



## Reset

(48) Count start flag	(0380 <sub>16</sub> )...	00 <sub>16</sub>	(77) Port P3 direction register	(03E7 <sub>16</sub> )...	00 <sub>16</sub>
(49) Clock prescaler reset flag	(0381 <sub>16</sub> )...	0XXXXXX	(78) Port P4 direction register	(03EA <sub>16</sub> )...	00 <sub>16</sub>
(50) One-shot start flag	(0382 <sub>16</sub> )...	00XX0000	(79) Port P7 direction register	(03EF <sub>16</sub> )...	00 <sub>16</sub>
(51) Trigger select flag	(0383 <sub>16</sub> )...	00 <sub>16</sub>	(80) Port P8 direction register	(03F2 <sub>16</sub> )...	00 <sub>16</sub>
(52) Up-down flag	(0384 <sub>16</sub> )...	00 <sub>16</sub>	(81) Port P9 direction register	(03F3 <sub>16</sub> )...	00 <sub>16</sub>
(53) Timer A0 mode register	(0396 <sub>16</sub> )...	00 <sub>16</sub>	(82) Port P10 direction register	(03F6 <sub>16</sub> )...	00 <sub>16</sub>
(54) Timer A1 mode register	(0397 <sub>16</sub> )...	00 <sub>16</sub>	(83) Pull-up control register 0	(03FD <sub>16</sub> )...	00 <sub>16</sub>
(55) Timer A2 mode register	(0398 <sub>16</sub> )...	00 <sub>16</sub>	(84) Pull-up control register 1	(03FE <sub>16</sub> )...	00 <sub>16</sub>
(56) Timer A3 mode register	(0399 <sub>16</sub> )...	00 <sub>16</sub>	(85) Data registers (R0/R1/R2/R3)		0000 <sub>16</sub>
(57) Timer A4 mode register	(039A <sub>16</sub> )...	00 <sub>16</sub>	(86) Address registers (A0/A1)		0000 <sub>16</sub>
(58) Timer B0 mode register	(039B <sub>16</sub> )...	00?XX000	(87) Frame base register (FB)		0000 <sub>16</sub>
(59) Timer B1 mode register	(039C <sub>16</sub> )...	00?XX000	(88) Interrupt table register (INTB)		00000 <sub>16</sub>
(60) Timer B2 mode register	(039D <sub>16</sub> )...	00?XX000	(89) User stack pointer (USP)		0000 <sub>16</sub>
(61) UART0 transmit/receive mode register	(03A0 <sub>16</sub> )...	00 <sub>16</sub>	(90) Interrupt stack pointer (ISP)		0000 <sub>16</sub>
(62) UART0 transmit/receive control register 0	(03A4 <sub>16</sub> )...	00001000	(91) Static base register (SB)		0000 <sub>16</sub>
(63) UART0 transmit/receive control register 1	(03A5 <sub>16</sub> )...	00000010	(92) Flag register (FLG)		0000 <sub>16</sub>
(64) UART1 transmit/receive mode register	(03A8 <sub>16</sub> )...	00 <sub>16</sub>			
(65) UART1 transmit/receive control register 0	(03AC <sub>16</sub> )...	00001000			
(66) UART1 transmit/receive control register 1	(03AD <sub>16</sub> )...	00000010			
(67) UART transmit/receive control register 2	(03B0 <sub>16</sub> )...	XXXXXX00			
(68) Flash memory control register 0 (Note)	(03B4 <sub>16</sub> )...	00100000			
(69) Flash memory control register 1 (Note)	(03B5 <sub>16</sub> )...	XXXXXXXX00			
(70) Flash command register (Note)	(03B6 <sub>16</sub> )...	00 <sub>16</sub>			
(71) DMA0 cause select register	(03B8 <sub>16</sub> )...	00 <sub>16</sub>			
(72) DMA1 cause select register	(03BA <sub>16</sub> )...	00 <sub>16</sub>			
(73) A-D control register 2	(03D4 <sub>16</sub> )...	XXXXXXXX0			
(74) A-D control register 0	(03D6 <sub>16</sub> )...	00000???			
(75) A-D control register 1	(03D7 <sub>16</sub> )...	00 <sub>16</sub>			
(76) D-A control register	(03DC <sub>16</sub> )...	00 <sub>16</sub>			

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

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

Figure 13. Device's internal status after a reset is cleared



## Software Reset

### Software Reset

Writing "1" to bit 3 of the processor mode register 0 (address 0004<sub>16</sub>) applies a (software reset) reset to the microcomputer. A software reset has almost the same effect as a hardware reset. The contents of internal RAM are preserved.

Figure 14 shows the processor mode register 0 and 1.

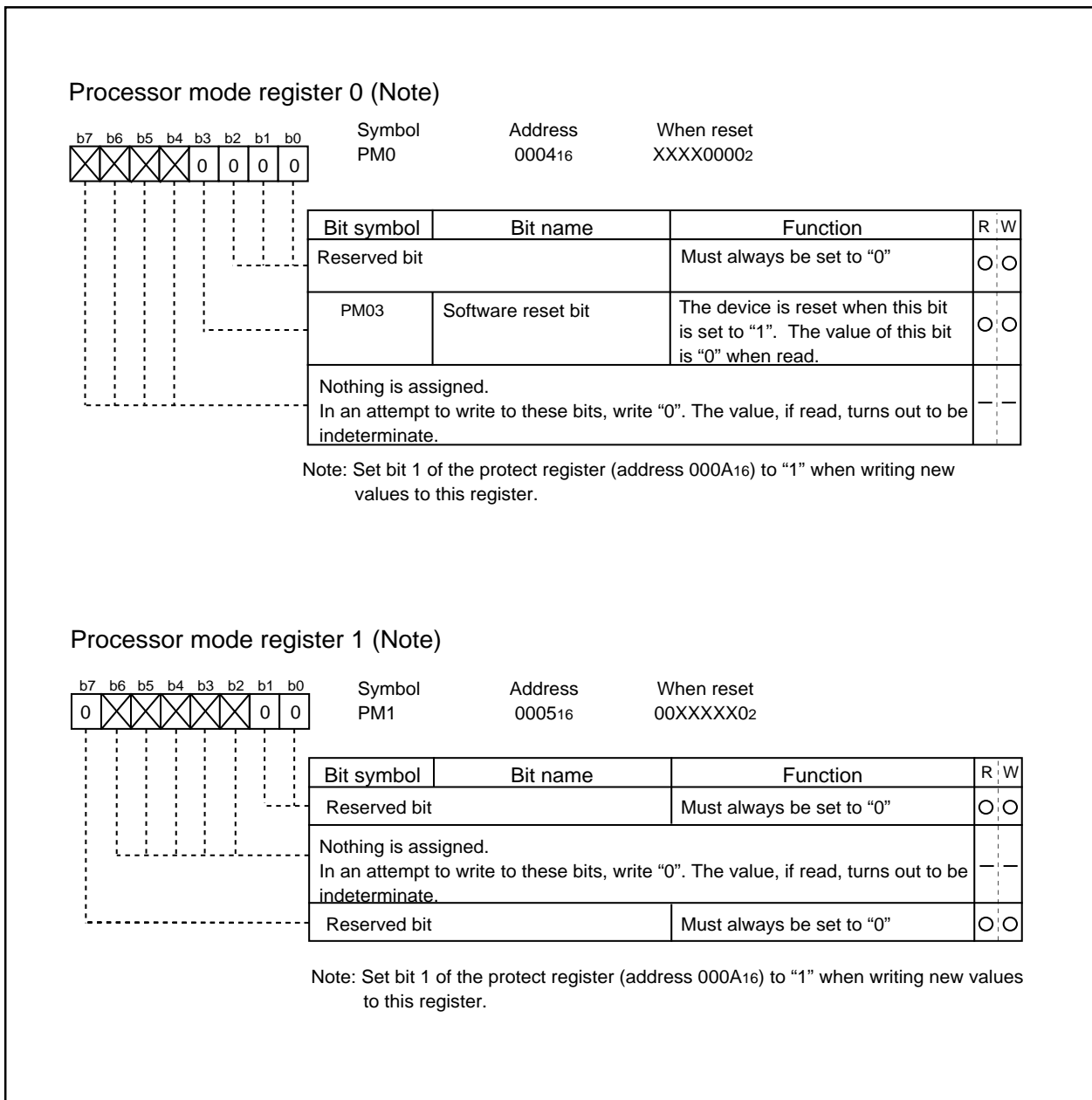


Figure 14. Processor mode register 0 and 1

## Clock Generating Circuit

### Clock Generating Circuit

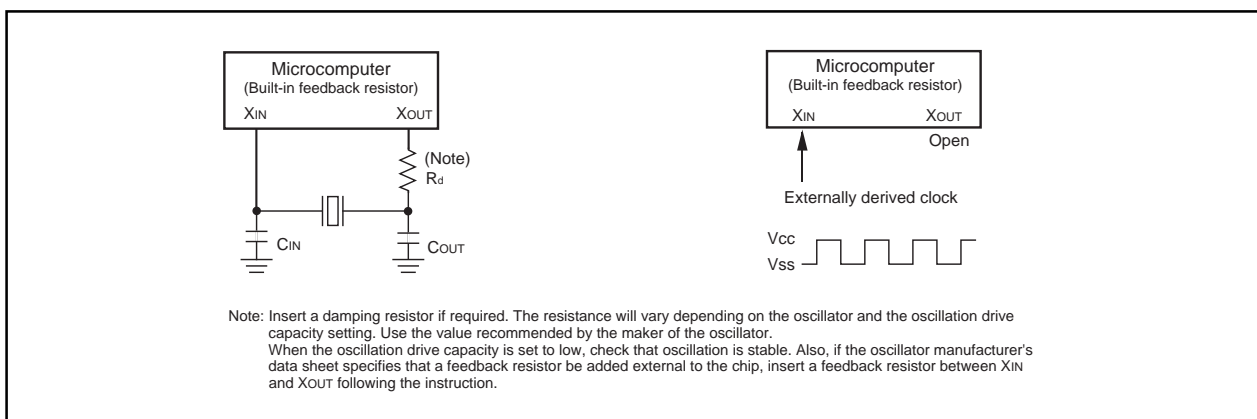
The clock generating circuit contains two oscillator circuits that supply the operating clock sources to the CPU and internal peripheral units.

**Table 2. Main clock and sub clock generating circuits**

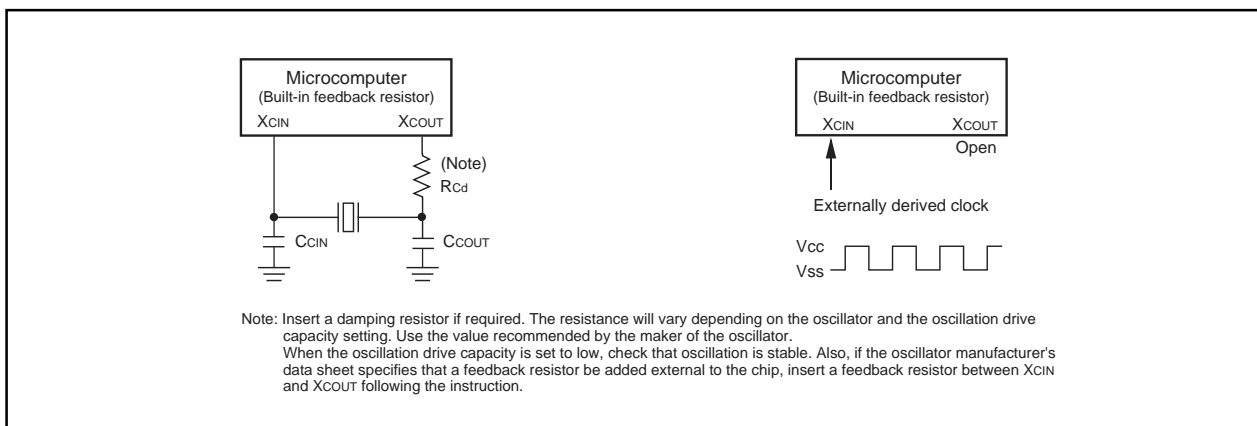
	Main clock generating circuit	Sub clock generating circuit
Use of clock	<ul style="list-style-type: none"> <li>• CPU's operating clock source</li> <li>• Internal peripheral units' operating clock source</li> </ul>	<ul style="list-style-type: none"> <li>• CPU's operating clock source</li> <li>• Timer A/B's count clock source</li> </ul>
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 15 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 16 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 15 and 16 vary with each oscillator used. Use the values recommended by the manufacturer of your oscillator.



**Figure 15. Examples of main clock**



**Figure 16. Examples of sub clock**

Clock Generating Circuit

**Clock Control**

Figure 17 shows the block diagram of the clock generating circuit.

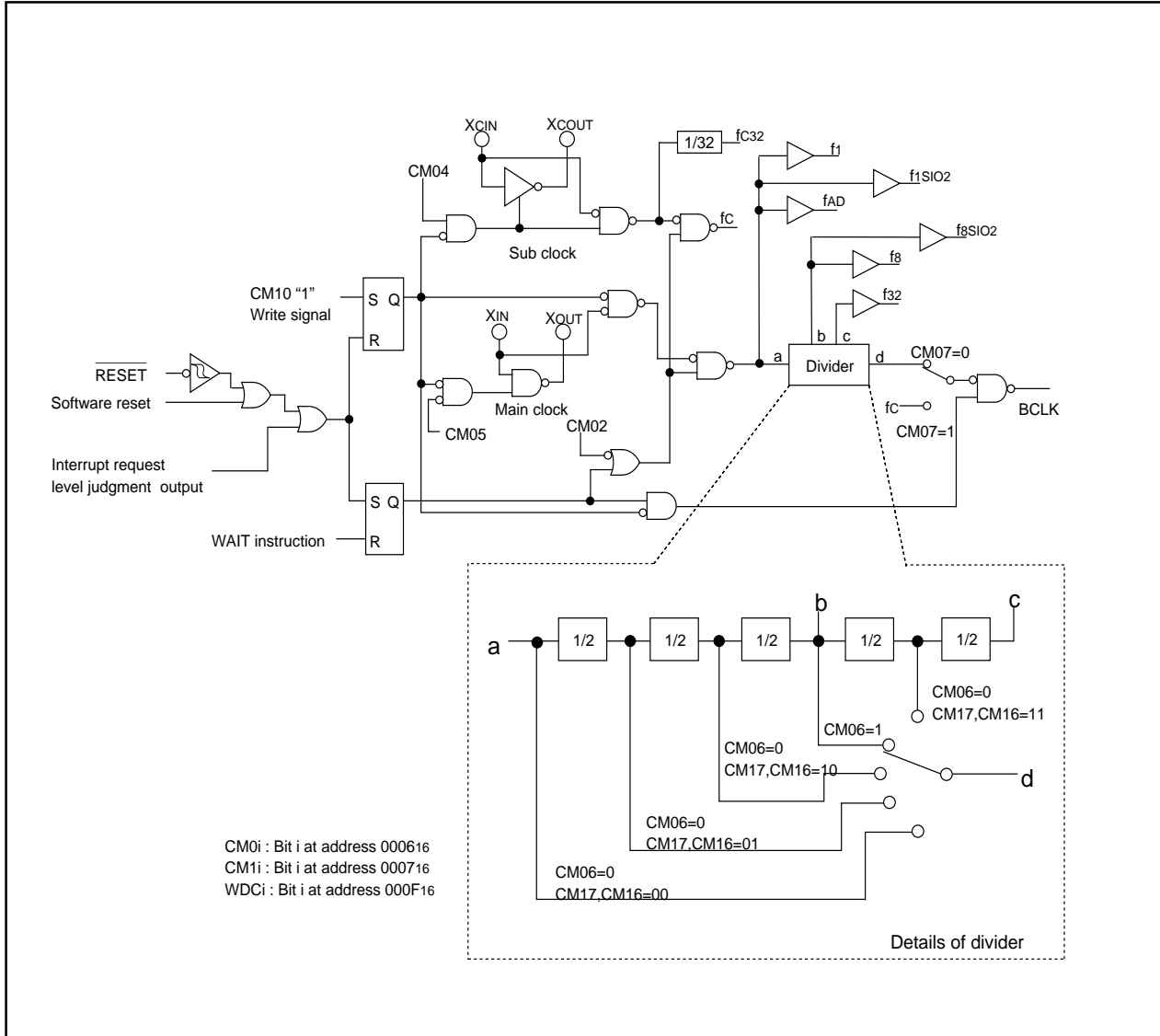


Figure 17. Clock generating circuit

## 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<sub>16</sub>). Stopping the clock, after switching the operating clock source of CPU to the sub-clock, reduces the power dissipation. After the oscillation of the main clock oscillation circuit has stabilized, the drive capacity of the main clock oscillation circuit can be reduced using the XIN-XOUT drive capacity select bit (bit 5 at address 0007<sub>16</sub>). 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<sub>16</sub>), the sub-clock can be selected as the BCLK by using the system clock select bit (bit 7 at address 0006<sub>16</sub>). However, be sure that the sub-clock oscillation has fully stabilized before switching.

After the oscillation of the sub-clock oscillation circuit has stabilized, the drive capacity of the sub-clock oscillation circuit can be reduced using the XCIN-XCOUT drive capacity select bit (bit 3 at address 0006<sub>16</sub>). 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  $f_c$  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 BCLK signal can be output from BCLK pin by the BCLK output disable bit (bit 7 at address 0004<sub>16</sub>) in the memory expansion and the microprocessor modes.

The main clock division select bit 0 (bit 6 at address 0006<sub>16</sub>) 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( $f_1$ , $f_8$ , $f_{32}$ , $f_{AD}$ , $f_{SIO2}$ , $f_{8SIO2}$ )

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<sub>16</sub>) to "1" and then executing a WAIT instruction.

### (5) $f_{c32}$

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

### (6) $f_c$

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

## Clock Generating Circuit

Figure 18 shows the system clock control registers 0 and 1.

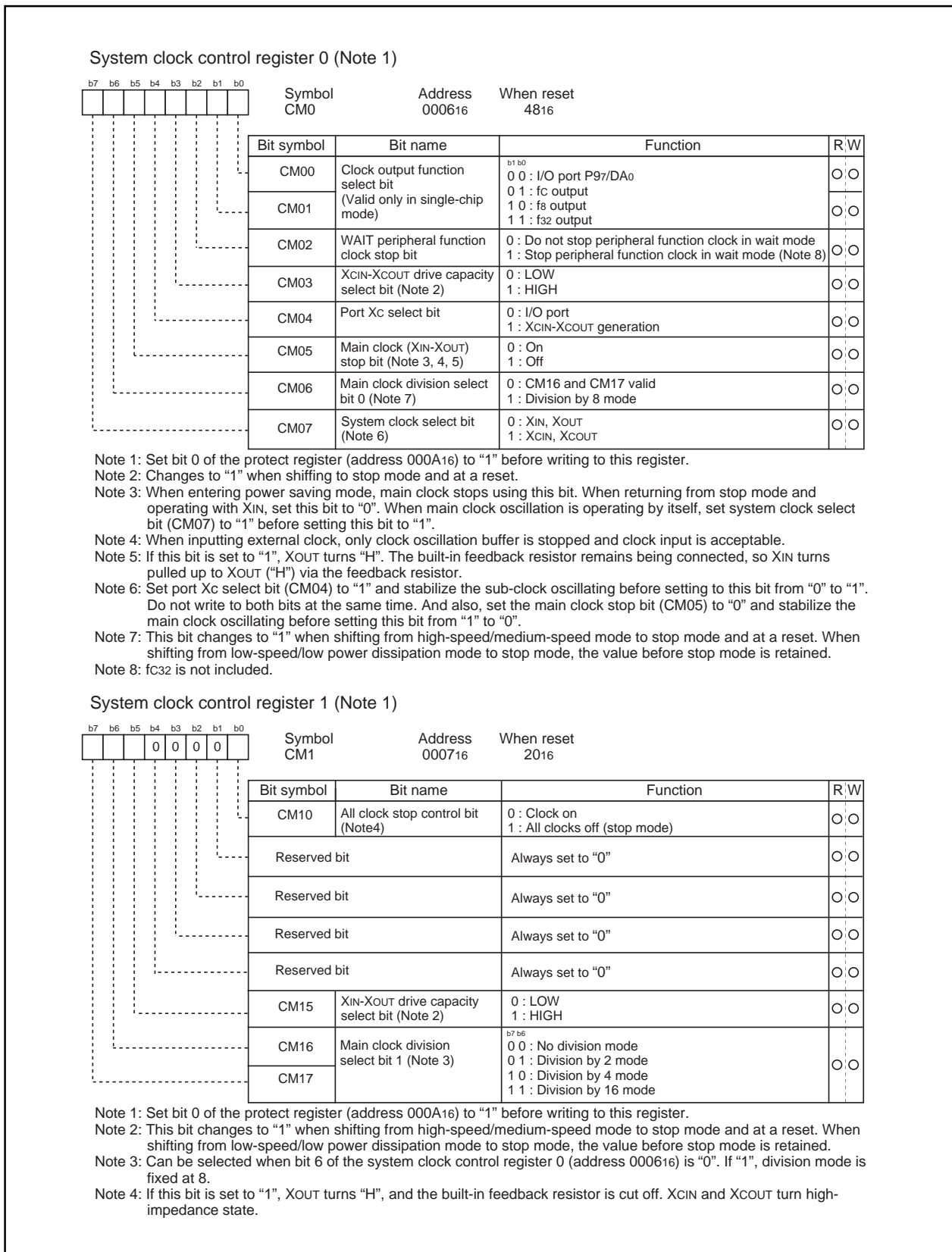


Figure 18. Clock control registers 0 and 1

## Clock Output

### Clock Output

The clock output function select bit (bit 0,1 at address 0006<sub>16</sub>) allows you to choose the clock from f<sub>8</sub>, f<sub>32</sub>, or f<sub>c</sub> to be output from the P97/DA0/CLKOUT/DIMOUT pin. When the WAIT peripheral function clock stop bit (bit 2 at address 0006<sub>16</sub>) is set to "1", the output of f<sub>8</sub> and f<sub>32</sub> stop by executing of WAIT instruction.

### Stop Mode

Writing "1" to the all-clock stop control bit (bit 0 at address 0007<sub>16</sub>) stops all oscillation and the microcomputer enters stop mode. In stop mode, the content of the internal RAM is retained provided that VCC remains above 2V.

Because the oscillation of BCLK, f<sub>1</sub> to f<sub>32</sub>, f<sub>c</sub>, f<sub>c32</sub>, and f<sub>AD</sub> stops in stop mode, peripheral functions such as the fluorescent display function, serial I/O 2, 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 and UART2 functions provided an external clock is selected. Table 3 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<sub>16</sub>) is set to "1". When shifting from low-speed/low power dissipation mode to stop mode, the value before stop mode is retained.

**Table 3. Port status during stop mode**

Pin		States
Port		Retains status before stop mode
CLKOUT	When f <sub>c</sub> selected	"H"
	When f <sub>8</sub> , f <sub>32</sub> selected	Retains status before stop mode

### Wait Mode

When a WAIT instruction is executed, BCLK stops and the microcomputer enters the wait mode. In this mode, oscillation continues but BCLK and watchdog timer stop. Writing "1" to the WAIT peripheral function clock stop bit and executing a WAIT instruction stops the clock being supplied to the internal peripheral functions, allowing power dissipation to be reduced. Table 4 shows the status of the ports in wait mode.

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

**Table 4. Port status during wait mode**

Pin		States
Port		Retains status before wait mode
CLKOUT	When f <sub>c</sub> selected	Does not stop
	When f <sub>8</sub> , f <sub>32</sub> selected	Does not stop when the WAIT peripheral function clock stop bit is "0". (Note) When the WAIT peripheral function clock stop bit is "1", the status immediately prior to entering wait mode is maintained.

Note: Attention that reducing the power dissipation is impossible.

## Status Transition Of BCLK

Power dissipation can be reduced and low-voltage operation achieved by changing the count source for BCLK. Table WA-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 000616) 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

fc 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

fc is the BCLK and the main clock is stopped.

Note : Before the count source for BCLK can be changed from XIN to XCIN 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 5. 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.

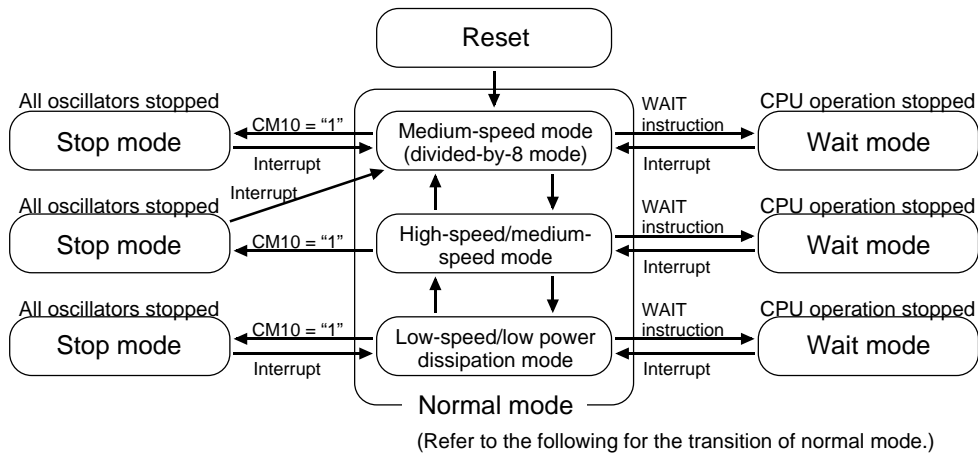
#### (c) Stop mode

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.

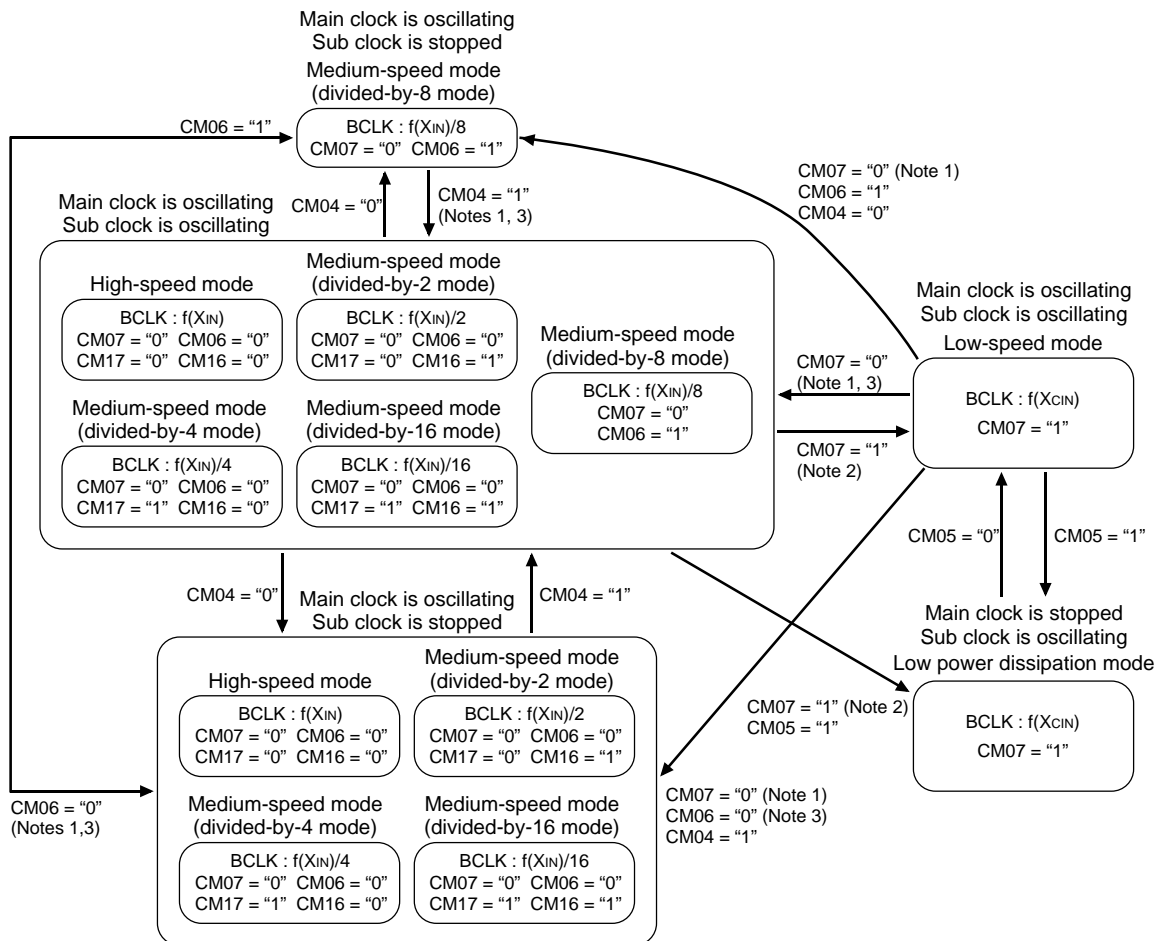
Figure 19 shows the state transition diagram of the above modes.



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 19. State transition diagram of Power control mode

## Protection

## 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 20 shows the protect register. The values in the processor mode register 0 (address 0004<sub>16</sub>), processor mode register 1 (address 0005<sub>16</sub>), system clock control register 0 (address 0006<sub>16</sub>), and system clock control register 1 (address 0007<sub>16</sub>) 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 address 000A<sub>16</sub>) and processor mode register 0 and 1 write-enable bit (bit 1 at address 000A<sub>16</sub>) 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".

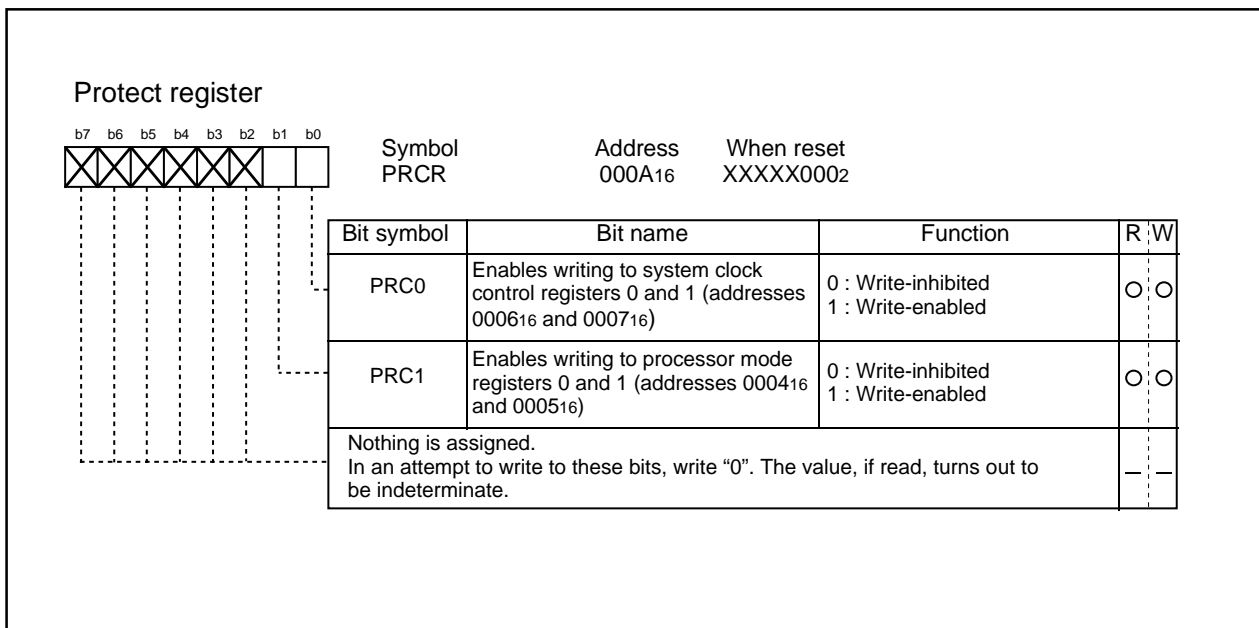
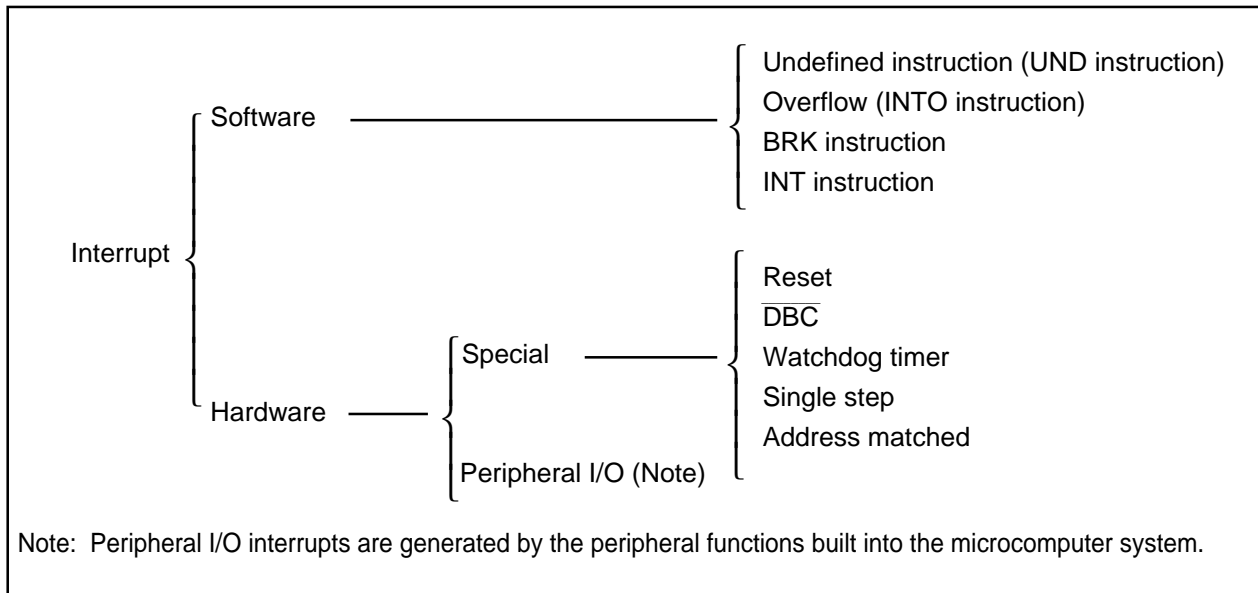


Figure 20. Protect register

## Overview of Interrupt

### Type of Interrupts

Figure 21 shows the types of interrupts.



**Figure 21. Classification of interrupts**

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

## Software Interrupts

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

- **Undefined instruction interrupt**

An undefined instruction interrupt occurs when executing the UND instruction.

- **Overflow interrupt**

An overflow interrupt occurs when 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.

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

- **DMA0 interrupt, DMA1 interrupt**

These are interrupts that DMA generates.

- **A-D conversion interrupt**

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

- **UART0 and UART1 transmission interrupt**

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

- **UART0 and UART1 reception interrupt**

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

- **SI/O automatic transfer interrupt**

This is an interrupt that the SI/O automatic transfer generates.

- **Timer A0 interrupt through timer A4 interrupt**

These are interrupts that timer A generates

- **Timer B0 interrupt through timer B2 interrupt**

These are interrupts that timer B generates.

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

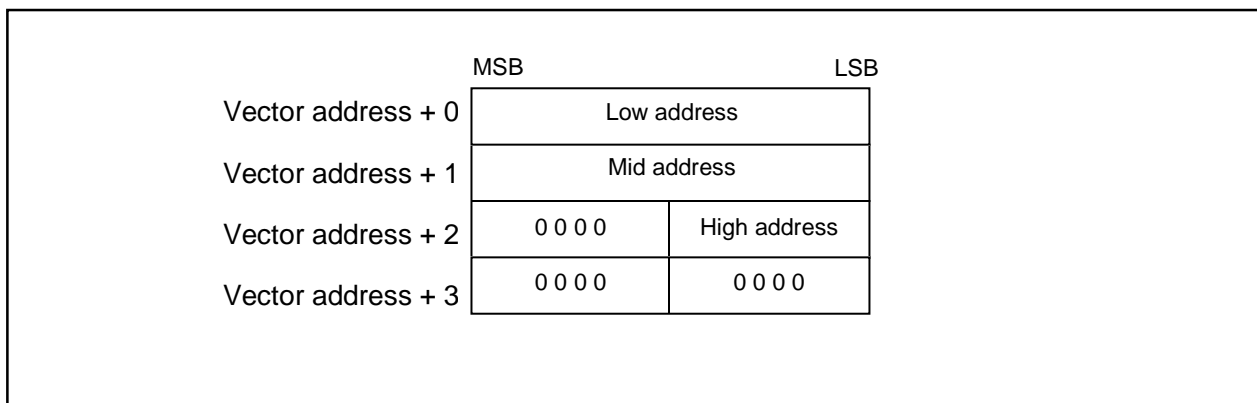
An  $\overline{\text{INT}}$  interrupt occurs if either a rising edge or a falling edge is input to the  $\overline{\text{INT}}$  pin.

## Interrupt

## 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 22 shows the format for specifying the address.

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



**Figure 22. Format for specifying interrupt vector addresses**

### • Fixed vector tables

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

**Table 6. 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	FFFD <sub>16</sub> to FFFD <sub>16</sub> F	Interrupt on UND instruction
Overflow	FFFE <sub>16</sub> 0 to FFFE <sub>16</sub> 3	Interrupt on INTO instruction
BRK instruction	FFFE <sub>16</sub> 4 to FFFE <sub>16</sub> 7	If the vector contains FF <sub>16</sub> , program execution starts from the address shown by the vector in the variable vector table
Address match	FFFE <sub>16</sub> 8 to FFFE <sub>16</sub> B	There is an address-matching interrupt enable bit
Single step (Note)	FFFE <sub>16</sub> C to FFFE <sub>16</sub> F	Do not use
Watchdog timer	FFFF <sub>16</sub> 0 to FFFF <sub>16</sub> 3	
DBC (Note)	FFFF <sub>16</sub> 4 to FFFF <sub>16</sub> 7	Do not use
-	FFFF <sub>16</sub> 8 to FFFF <sub>16</sub> B	-
Reset	FFFF <sub>16</sub> C to FFFFF <sub>16</sub> F	

Note: Interrupts used for debugging purposes only.

## Interrupt

- 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 7 shows the interrupts assigned to the variable vector tables and addresses of vector tables.

**Table 7. 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)	BRK instruction	Cannot be masked I flag
—		—	
Software interrupt number 7	+28 to +31 (Note)	$\overline{\text{INT3}}$	
Software interrupt number 8	+32 to +35 (Note)	$\overline{\text{INT4}}$	
Software interrupt number 9	+36 to +39 (Note)	$\overline{\text{INT5}}$	
—		—	
Software interrupt number 11	+44 to +47 (Note)	DMA0	
Software interrupt number 12	+48 to +51 (Note)	DMA1	
—		—	
Software interrupt number 14	+56 to +59 (Note)	A-D	
Software interrupt number 15	+60 to +63 (Note)	SI/O automatic transfer	
Software interrupt number 16	+64 to +67 (Note)	FLD	
Software interrupt number 17	+68 to +71 (Note)	UART0 transmit	
Software interrupt number 18	+72 to +75 (Note)	UART0 receive	
Software interrupt number 19	+76 to +79 (Note)	UART1 transmit	
Software interrupt number 20	+80 to +83 (Note)	UART1 receive	
Software interrupt number 21	+84 to +87 (Note)	Timer A0	
Software interrupt number 22	+88 to +91 (Note)	Timer A1	
Software interrupt number 23	+92 to +95 (Note)	Timer A2	
Software interrupt number 24	+96 to +99 (Note)	Timer A3	
Software interrupt number 25	+100 to +103 (Note)	Timer A4	
Software interrupt number 26	+104 to +107 (Note)	Timer B0	
Software interrupt number 27	+108 to +111 (Note)	Timer B1	
Software interrupt number 28	+112 to +115 (Note)	Timer B2	
Software interrupt number 29	+116 to +119 (Note)	$\overline{\text{INT0}}$	
Software interrupt number 30	+120 to +123 (Note)	$\overline{\text{INT1}}$	
Software interrupt number 31	+124 to +127 (Note)	$\overline{\text{INT2}}$	
Software interrupt number 32 to Software interrupt number 63	+128 to +131 (Note) to +252 to +255 (Note)	Software interrupt	Cannot be masked I flag

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

## 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 23 shows the memory map of the interrupt control registers.



## Interrupt

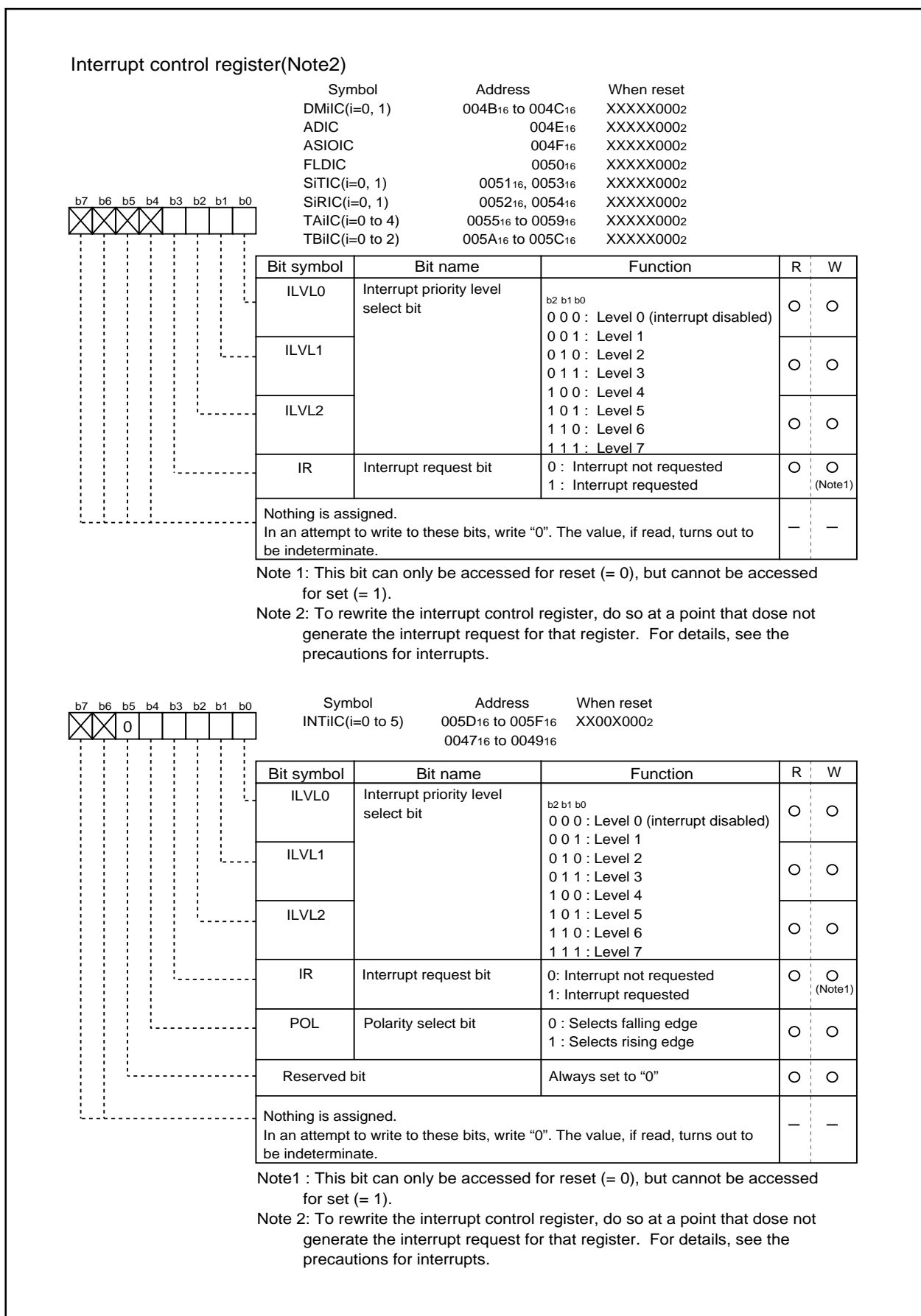


Figure 23. Interrupt control registers

## Interrupt

### 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 8 shows the settings of interrupt priority levels and Table 9 shows the interrupt levels enabled, according to the consist of the IPL.

The following are conditions under which an interrupt is accepted:

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

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

**Table 8. 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 ↓ High
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	

**Table 9. Interrupt levels enabled according to the contents of the IPL**

IPL	Enabled interrupt priority levels
IPL <sub>2</sub> IPL <sub>1</sub> IPL <sub>0</sub> 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                               ;
  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        ;
  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 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

### Interrupt Sequence

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

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

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

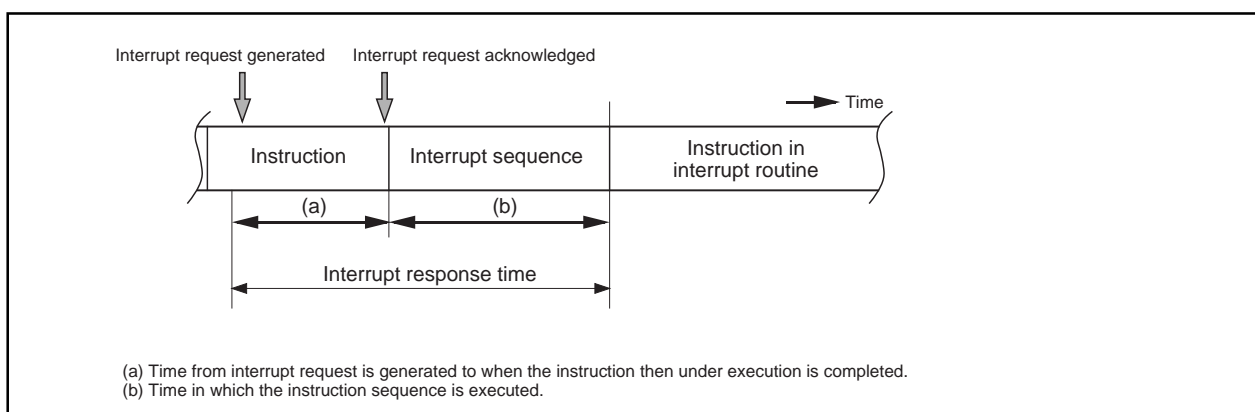
- (1) CPU gets the interrupt information (the interrupt number and interrupt request level) by reading address  $00000_{16}$ .
- (2) Saves the content of the flag register (FLG) as it was immediately before the start of interrupt sequence in the temporary register (Note) within the CPU.
- (3) Sets the interrupt enable flag (I flag), the debug flag (D flag), and the stack pointer select flag (U flag) to "0" (the U flag, however does not change if the INT instruction, in software interrupt numbers 32 through 63, is executed)
- (4) Saves the content of the temporary register (Note) within the CPU in the stack area.
- (5) Saves the content of the program counter (PC) in the stack area.
- (6) Sets the interrupt priority level of the accepted instruction in the IPL.

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

Note: This register cannot be utilized by the user.

### Interrupt Response Time

'Interrupt response time' is the period between the instant an interrupt occurs and the instant the first instruction within the interrupt routine has been executed. This time comprises the period from the 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 24 shows the interrupt response time.



**Figure 24. Interrupt response time**

## Interrupt

Time (a) is dependent on the instruction under execution. Thirty cycles is the maximum required for the DIVX instruction.

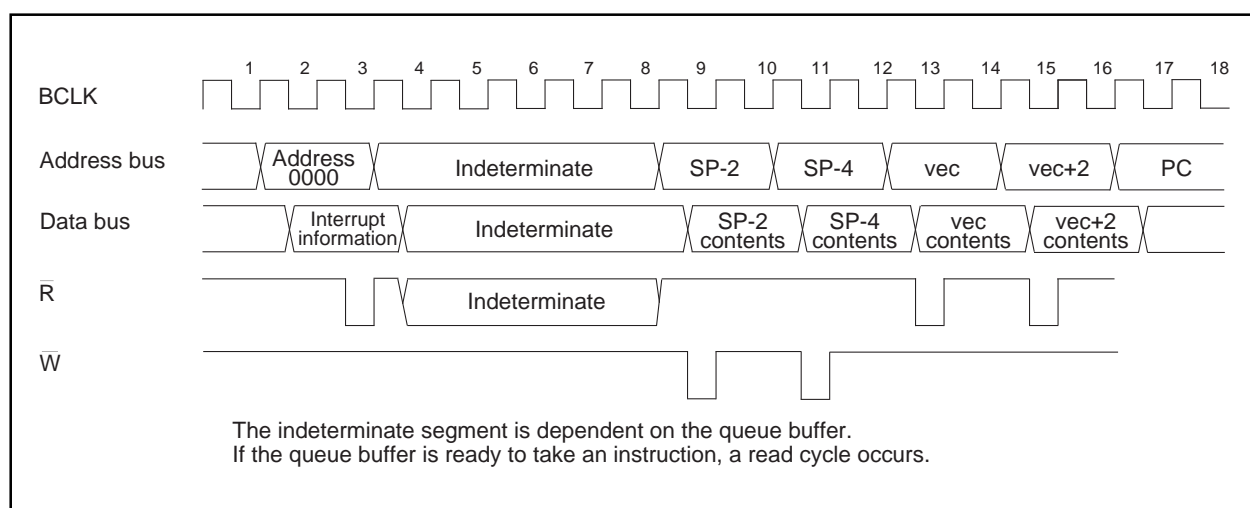
Time (b) is as shown in Table 10.

**Table 10. Time required for executing the interrupt sequence**

Interrupt vector address	Stack pointer (SP) value	16-Bit bust	8-Bit bus
Even	Even	18 cycles (Note 1)	20 cycles (Note 1)
Even	Odd	19 cycles (Note 1)	20 cycles (Note 1)
Odd (Note 2)	Even	19 cycles (Note 1)	20 cycles (Note 1)
Odd (Note 2)	Odd	20 cycles (Note 1)	20 cycles (Note 1)

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

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



**Figure 25. Time required for executing the interrupt sequence**

### Variation of IPL when Interrupt Request is Accepted

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

If an interrupt request, that does not have an interrupt priority level, is accepted, one of the values shown in Table 11 shows set in the IPL.

**Table 11. Relationship between interrupts without interrupt priority levels and IPL**

Interrupt sources without priority levels	Value set in the IPL
Watchdog timer	7
Reset	0
Other	Not changed

## Interrupt

## 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 26 shows the state of the stack as it was before the acceptance of the interrupt request, and the state the stack after the acceptance of the interrupt request.

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

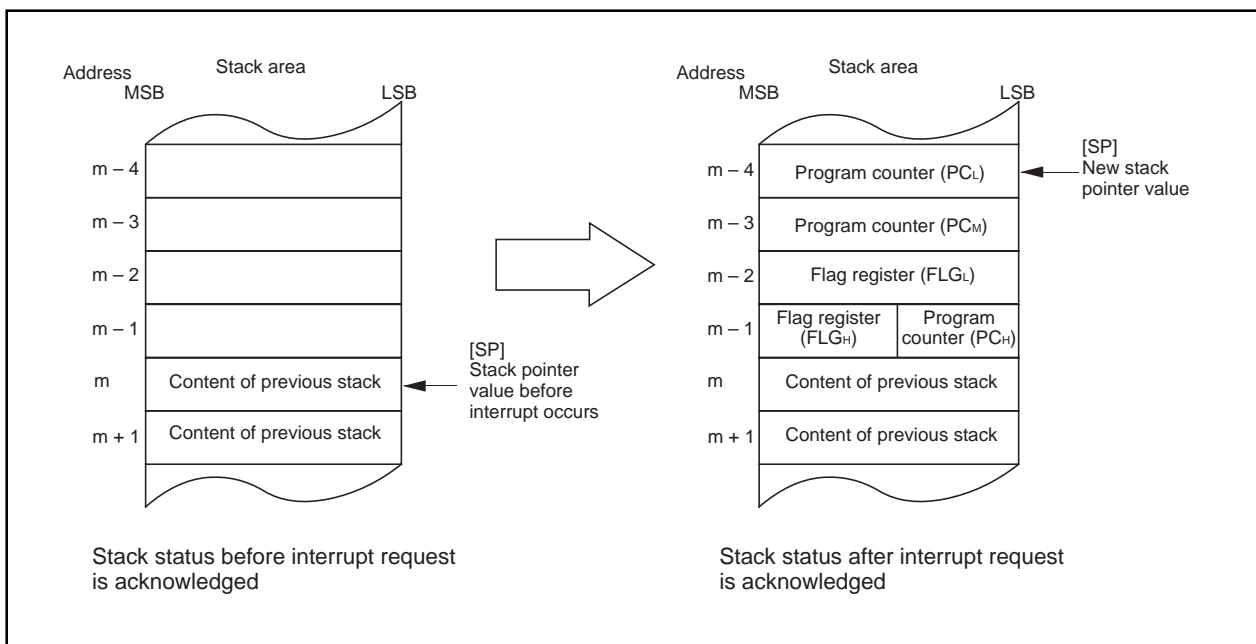


Figure 26. State of stack before and after acceptance of interrupt request

## Interrupt

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 27 shows the operation of the saving registers.

Note: Stack pointer indicated by U flag.

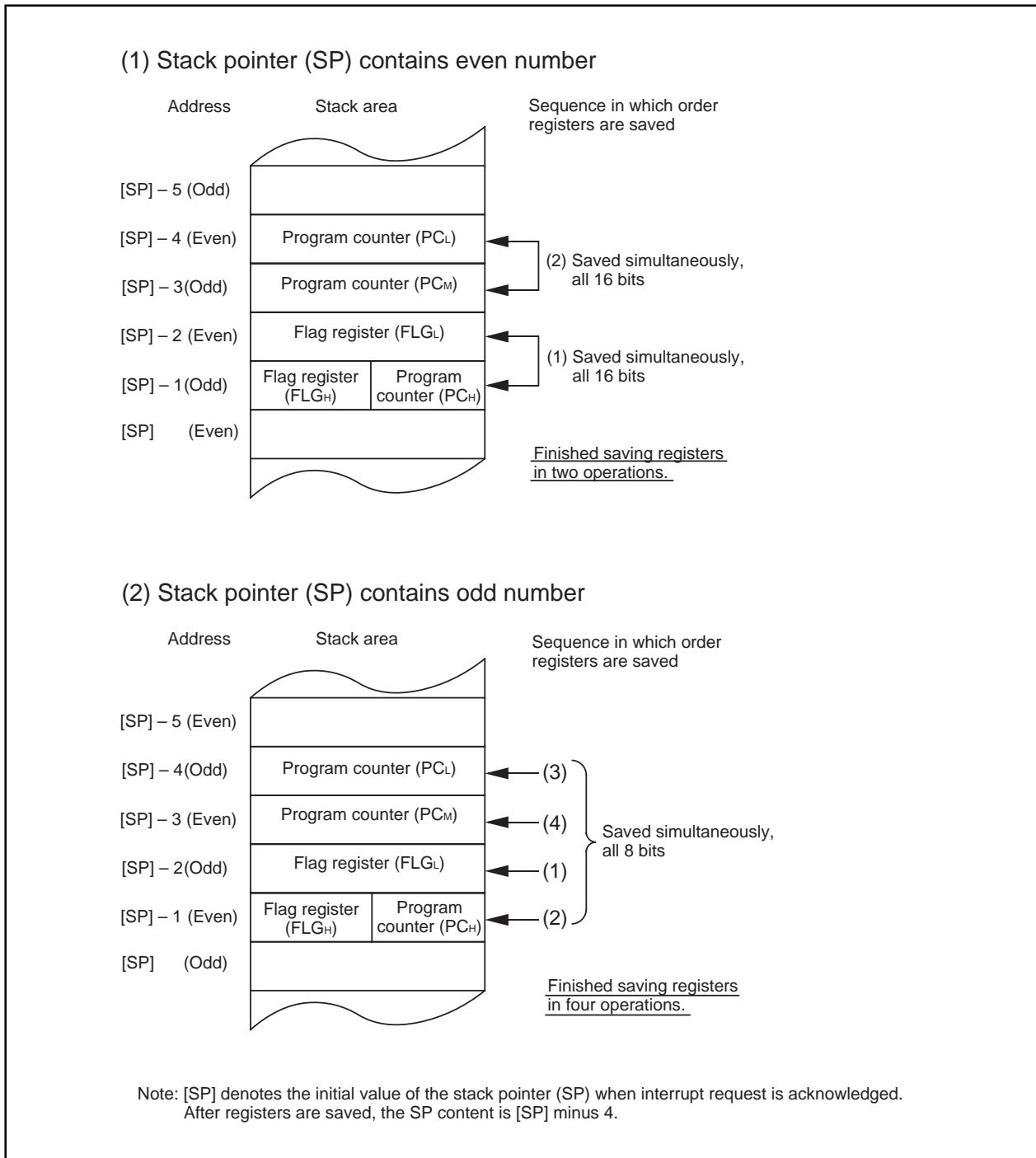


Figure 27. Operation of saving registers

## Returning from an Interrupt Routine

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

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

## Interrupt Priority

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

Assign an arbitrary priority to maskable interrupts (peripheral I/O interrupts) using the interrupt priority level select 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 28 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 >  $\overline{\text{DBC}}$  > Watchdog timer > Peripheral I/O > Single step > Address match

Figure 28. Hardware interrupts priorities

## Interrupt resolution circuit

When two or more interrupts are generated simultaneously, this circuit selects the interrupt with the highest priority level. Figure 29 shows the circuit that judges the interrupt priority level.



Interrupt

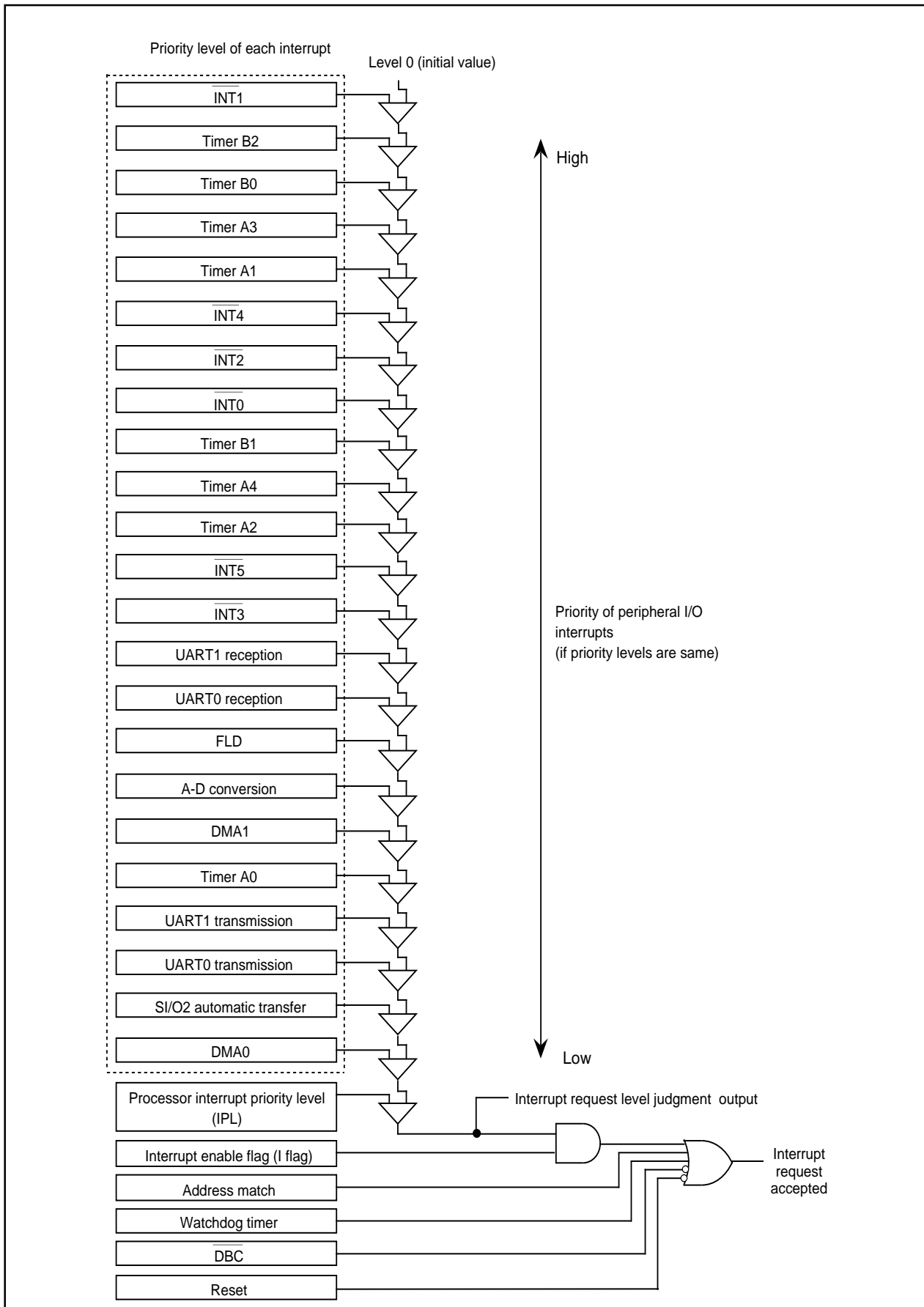


Figure 29. Maskable interrupts priorities

## Address Match Interrupt

## Address Match Interrupt

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

Figure 30 shows the address match interrupt-related registers.

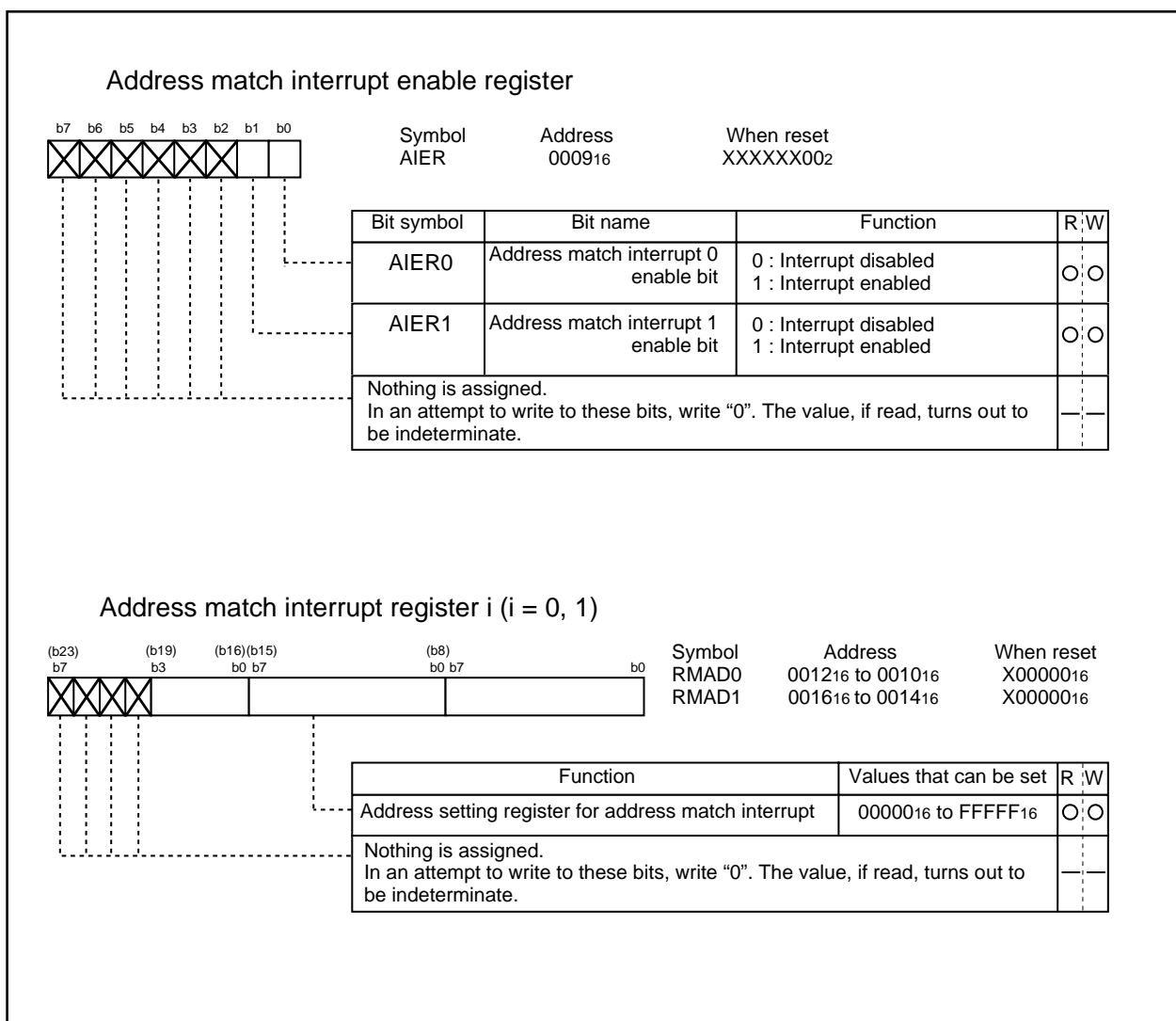


Figure 30. Address match interrupt-related registers

## Precautions for Interrupts

## Precautions for Interrupts

**(1) Reading address 00000<sub>16</sub>**

- 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<sub>16</sub> will then be set to "0".

Reading address 00000<sub>16</sub> 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<sub>16</sub> by software.

**(2) Setting the stack pointer**

- The value of the stack pointer immediately after reset is initialized to 0000<sub>16</sub>. 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.

**(3) 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  $\overline{\text{INT}}_0$  through  $\overline{\text{INT}}_5$  pins is changed, the interrupt request bit is sometimes set to "1". After changing the polarity, set the interrupt request bit to "0". Figure 31 shows the procedure for changing the  $\overline{\text{INT}}$  interrupt generate factor.

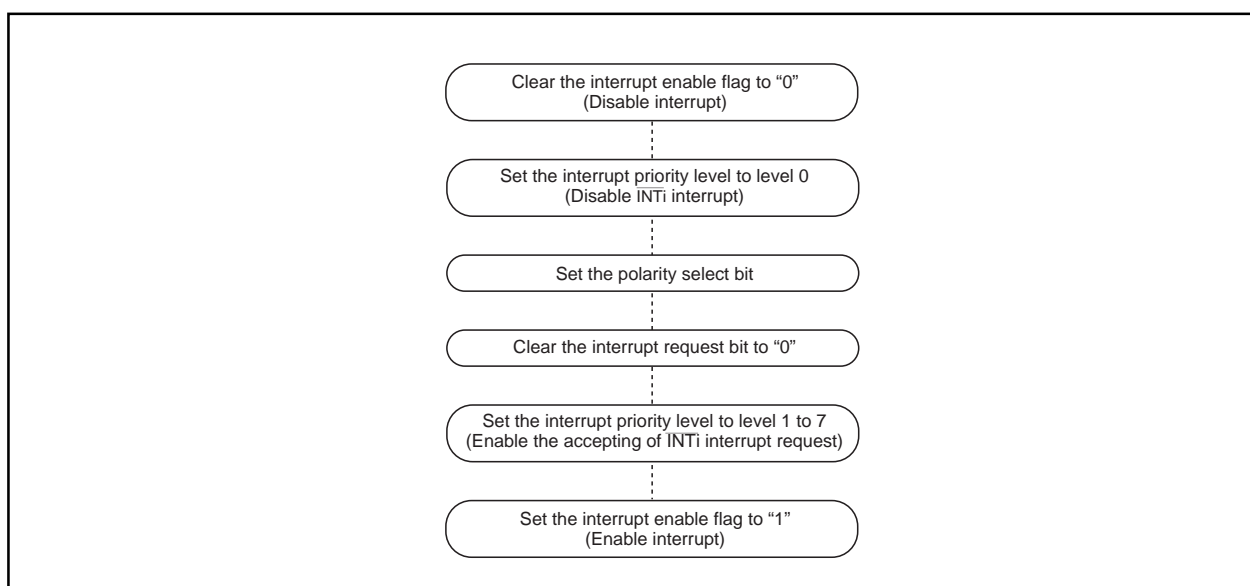


Figure 31. Switching condition of  $\overline{\text{INT}}$  interrupt request

## Precautions for Interrupts

### (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
  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
  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 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

### Watchdog Timer

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

#### With XIN chosen for BCLK

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

#### With XCIN chosen for BCLK

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

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 000E16) 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 000E16).

Figure 32 shows the block diagram of the watchdog timer. Figure 33 shows the watchdog timer-related registers.

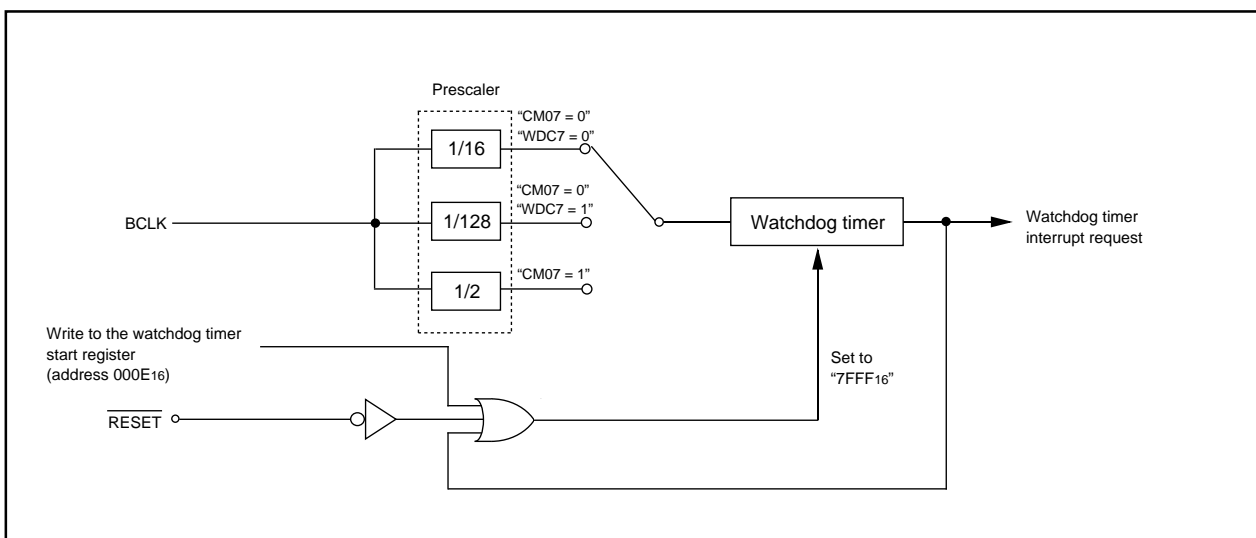


Figure 32. Block diagram of watchdog timer

## Watchdog Timer

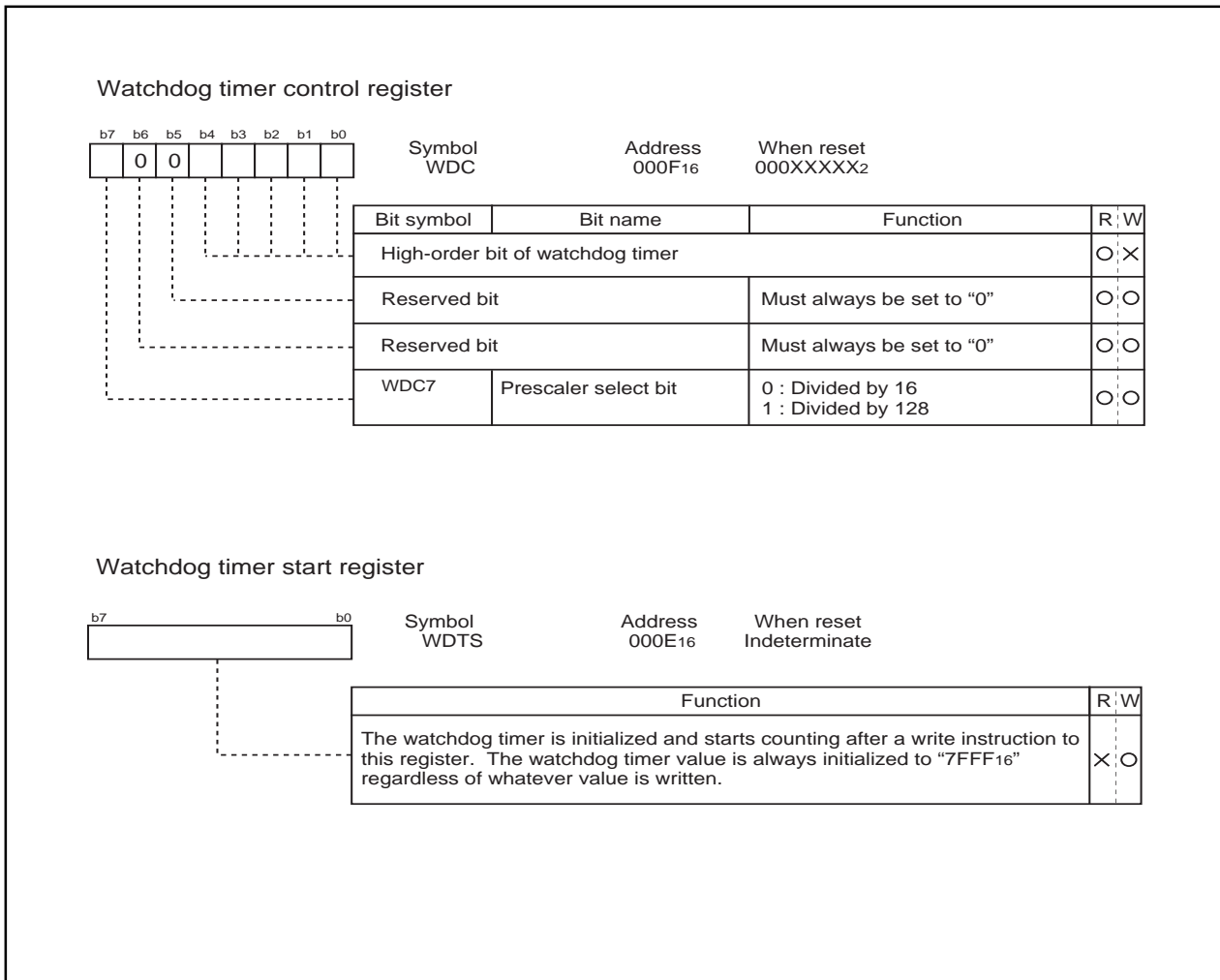


Figure 33. Watchdog timer control and start registers

## DMAC

## 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 34 shows the block diagram of the DMAC. Table 12 shows the DMAC specifications. Figure 35 to Figure 36 show the registers used by the DMAC.

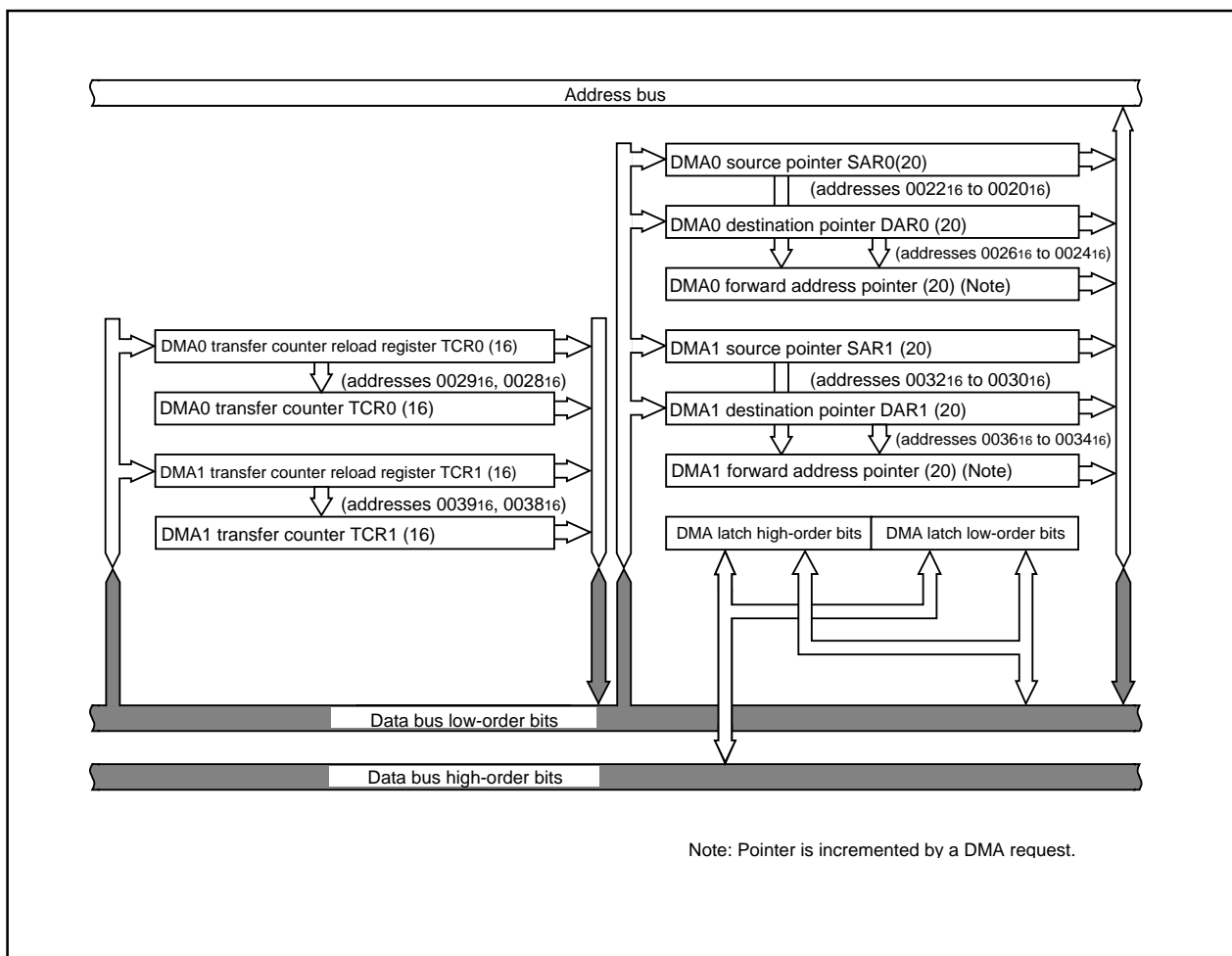


Figure 34. Block diagram of DMAC

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 12. DMAC specifications

Item	Specification
No. of channels	2 (cycle steal method)
Transfer memory space	<ul style="list-style-type: none"> <li>• From any address in the 1M bytes space to a fixed address</li> <li>• From a fixed address to any address in the 1M bytes space</li> <li>• From a fixed address to a fixed address</li> </ul> (Note that DMA-related registers [0020 <sub>16</sub> to 003F <sub>16</sub> ] 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 $\overline{INT0}$ or $\overline{INT1}$ ( $\overline{INT0}$ can be selected by DMA0, $\overline{INT1}$ by DMA1) Timer A0 to timer A4 interrupt requests Timer B0 to timer B2 interrupt requests UART0 transmission and reception interrupt requests UART1 transmission 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 or fixed (forward direction cannot be specified for both source and destination simultaneously)
Transfer mode	<ul style="list-style-type: none"> <li>• Single transfer mode After the transfer counter underflows, the DMA enable bit turns to "0", and the DMAC turns inactive</li> <li>• 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.</li> </ul>
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	<ul style="list-style-type: none"> <li>• When the DMA enable bit is set to "0", the DMAC is inactive.</li> <li>• After the transfer counter underflows in single transfer mode</li> </ul>
Forward address pointer and load timing for 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.



## DMAC

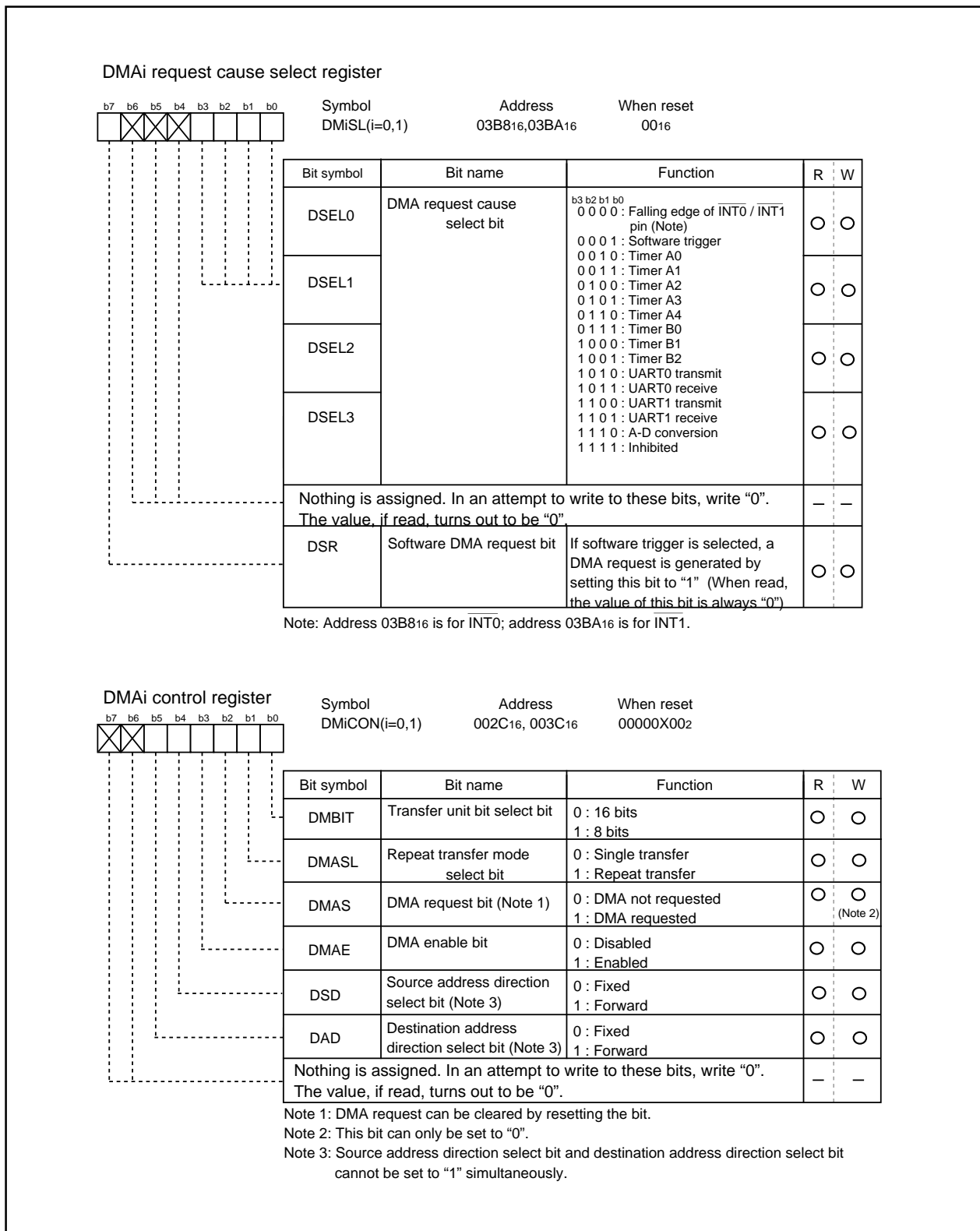


Figure 35. DMAC-related registers (1)

DMAC

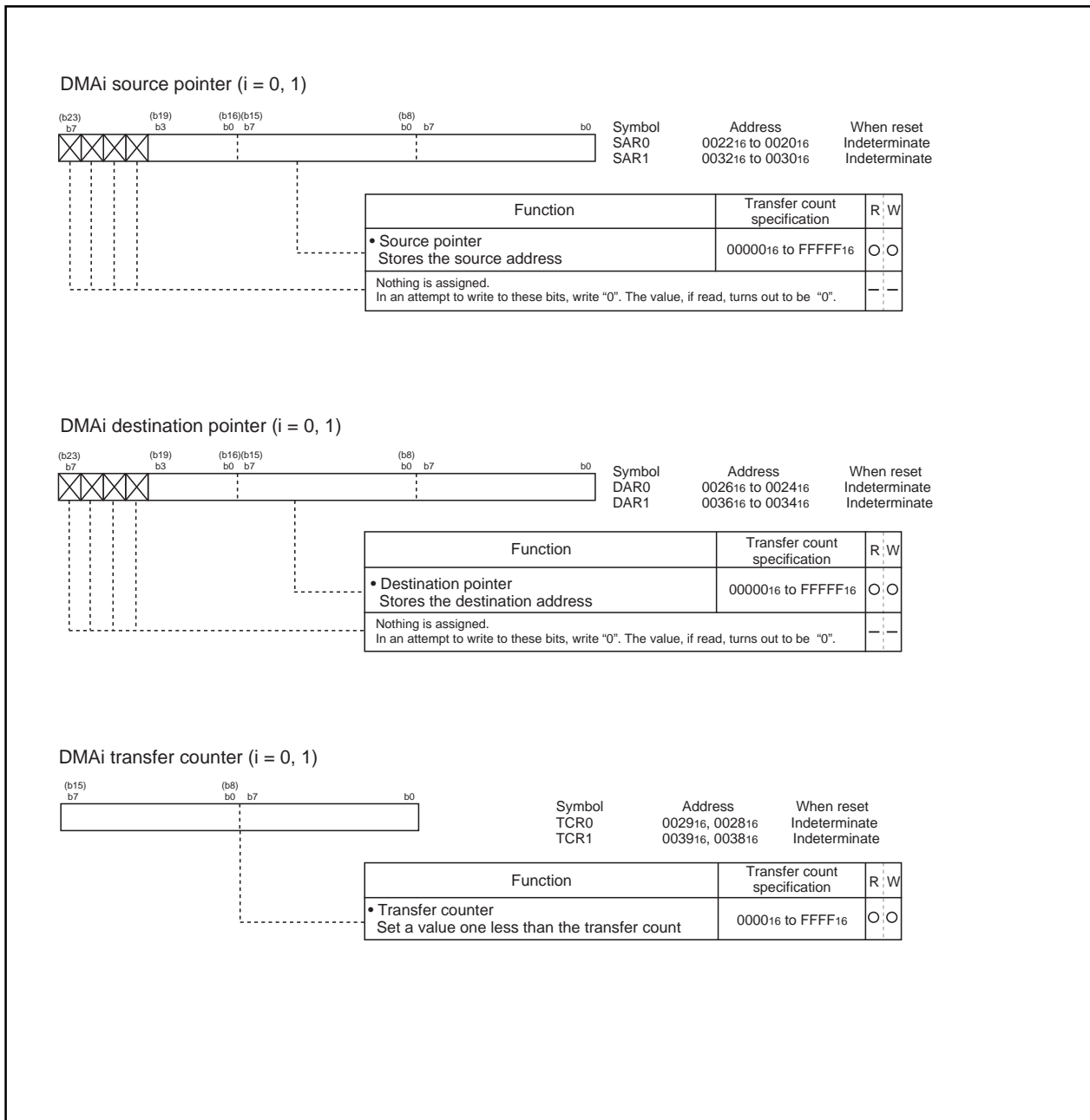


Figure 36. DMAC-related registers (2)

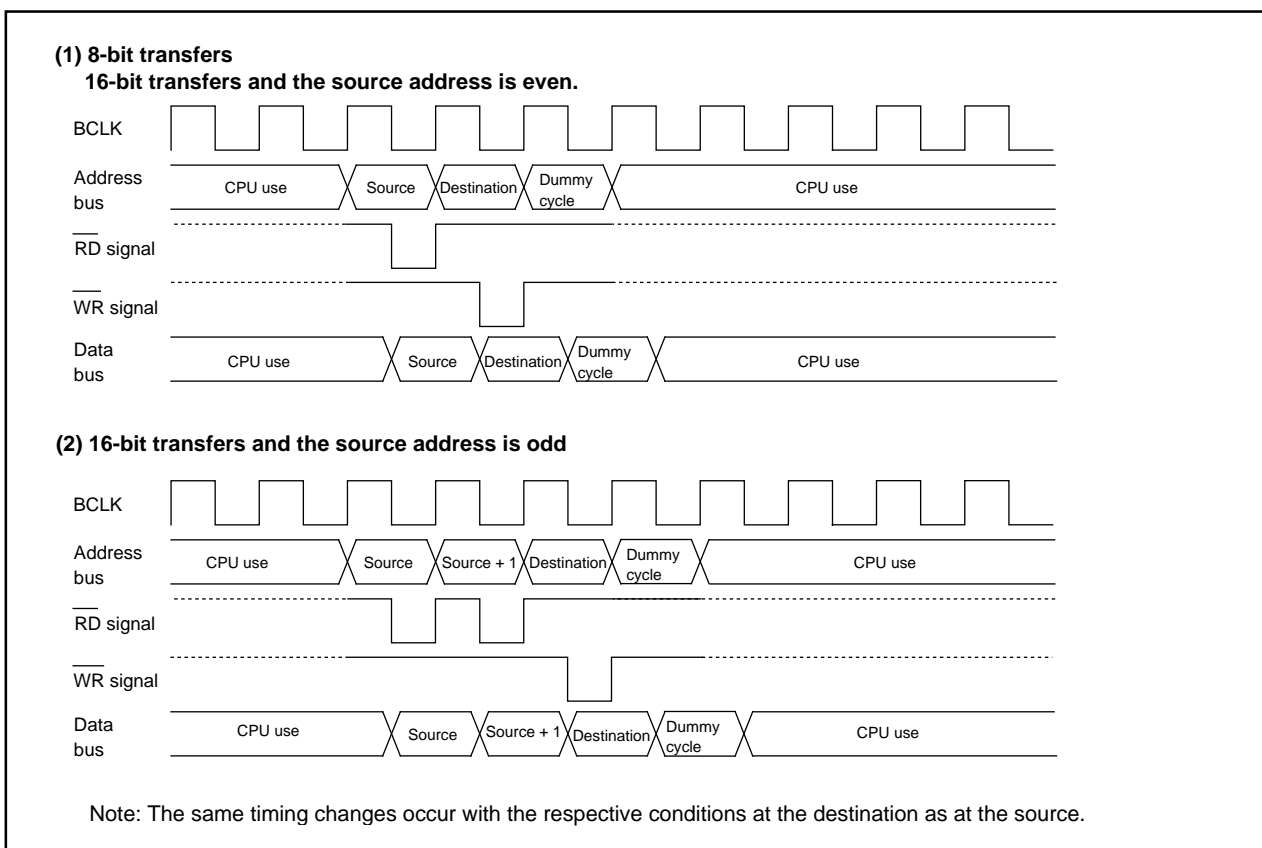
## (1) Transfer cycle

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.

### (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.

Figure 37 shows the example of the transfer cycles (a state of internal bus) 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.



**Figure 37. Example of transfer cycles for a source read (the state of internal bus)**

**(2) DMAC Transfer**

Any combination of even or odd transfer read and write addresses is possible. Table 13 shows the number of DMAC transfer cycles.

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

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

**Table 13. No. of DMAC transfer cycles**

Transfer unit	Access address	singelchip mode	
		No. of read cycles	No. of write 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	SFR area
1	2

## FLD Controller

The M30218 group has fluorescent display (FLD) drive and control circuits.

Table 14 shows the FLD controller specifications.

**Table 14. FLD controller specifications**

Item		Specification
FLD controller port	High-breakdown-voltage output port	• 52 pins ( 20 pins can switch general purpose port)
	CMOS port	• 4 pins ( 4 pins can switch general purpose port) (A driver must be installed externally)
Display pixel number		<ul style="list-style-type: none"> <li>• Used FLD output 28 segment X 28 digit (segment number + digit number ≤ 56)</li> <li>• Used digit output 40 segment X 16 digit (segment number ≤ 40, digit number ≤ 16)</li> <li>• Connected to M35501 56 segment X (connect number of M35501) digit (segment number ≤ 56, digit number ≤ number of M35501 X 16)</li> <li>• Used P44 to P47 expansion 52 segment X 16 digit (segment number ≤ 52, digit number ≤ 16)</li> </ul>
Period		<ul style="list-style-type: none"> <li>• 3.2 μs to 819.2 μs (count source X<sub>IN</sub>/32,10MHz)</li> <li>• 12.8 μs to 3276.8 μs (count source X<sub>IN</sub>/128,10MHz)</li> </ul>
Dimmer time		<ul style="list-style-type: none"> <li>• 3.2 μs to 819.2 μs (count source X<sub>IN</sub>/32,10MHz)</li> <li>• 12.8 μs to 3276.8 μs (count source X<sub>IN</sub>/128,10MHz)</li> </ul>
Interrupt		<ul style="list-style-type: none"> <li>• Digit interrupt</li> <li>• FLD blanking interrupt</li> </ul>
Key-scan		<ul style="list-style-type: none"> <li>• Key-scan used digit</li> <li>• Key-scan used segment</li> </ul>
Expand function		<ul style="list-style-type: none"> <li>• Digit pulse output function This function automatically outputs digit pulse.</li> <li>• M35501 connect function The number of digits can be increased easily by using the output of DIMOUT(P97) as CLK for the M35501.</li> <li>• Toff section generate / not generate function This function does not generate Toff1 section when the connected outputs are the same.</li> <li>• Gradation display function This function allows each segment to be set for dark or bright display.</li> <li>• P44 to P47 expansion function This function provides 16 lines of digit outputs from four ports by attaching a 4 — 16 decoder.</li> </ul>

FLD controller

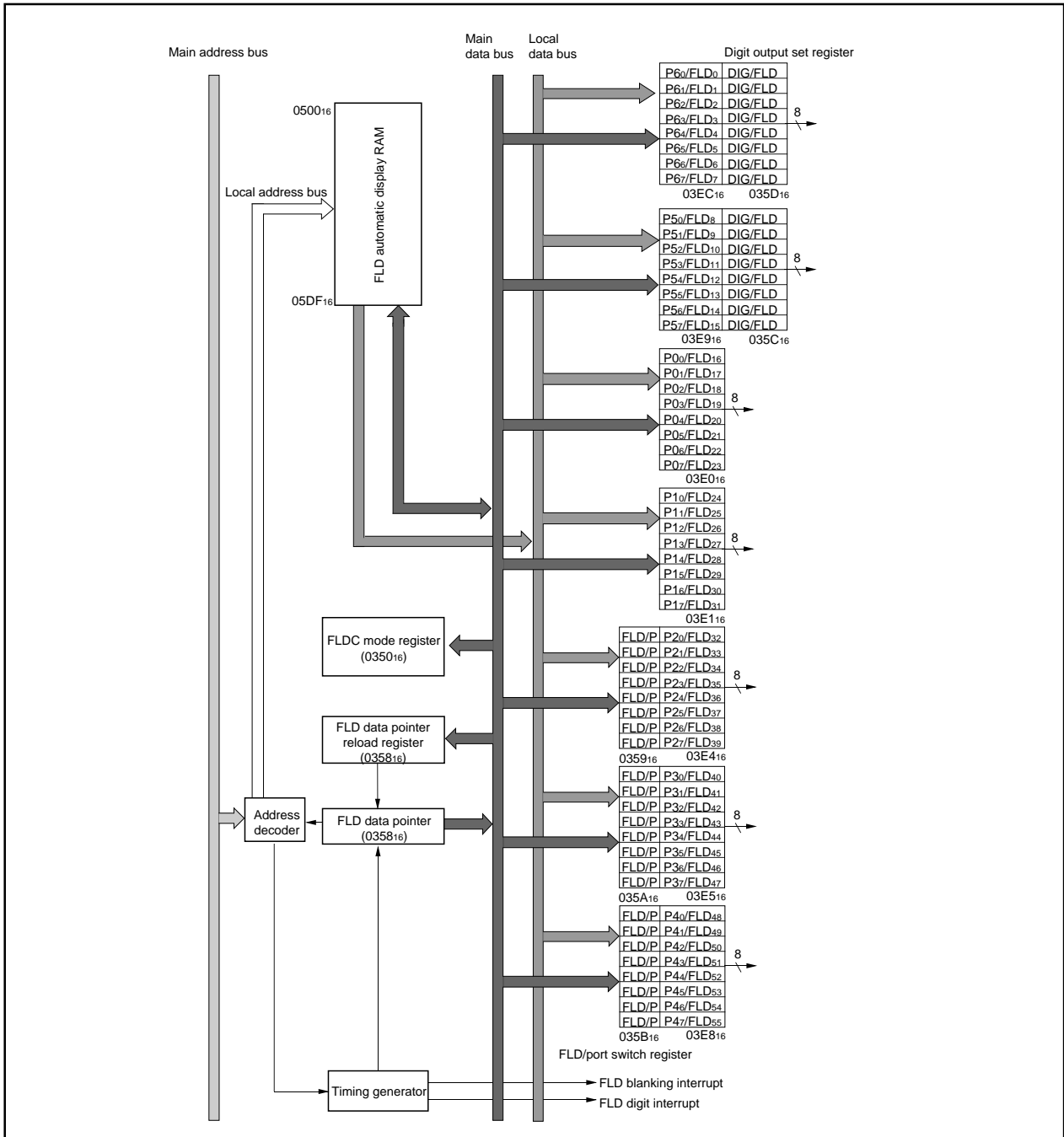


Figure 38. Block Diagram for FLD Control Circuit

## FLD controller

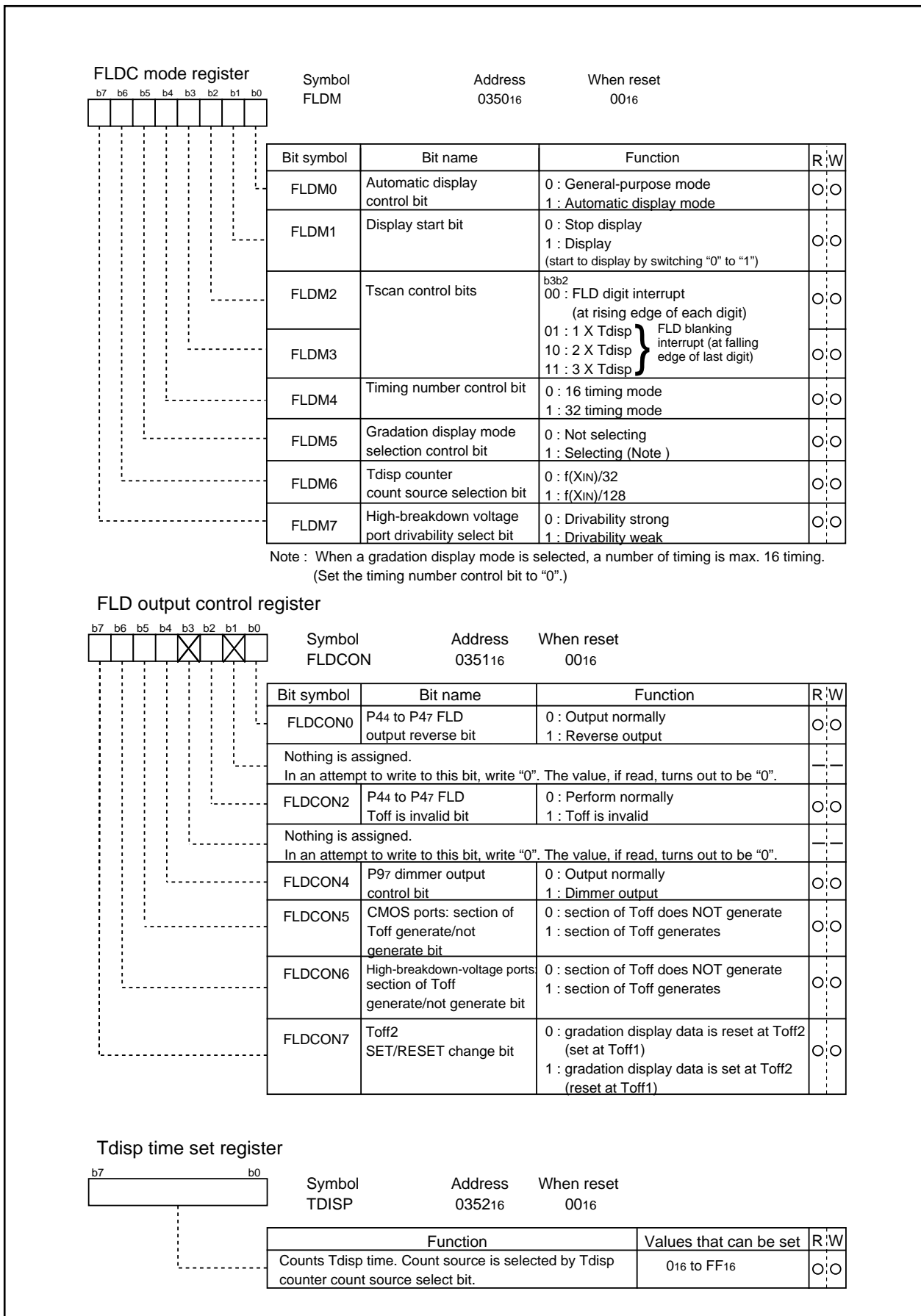


Figure 39. FLDC-related Register(1)

FLD controller

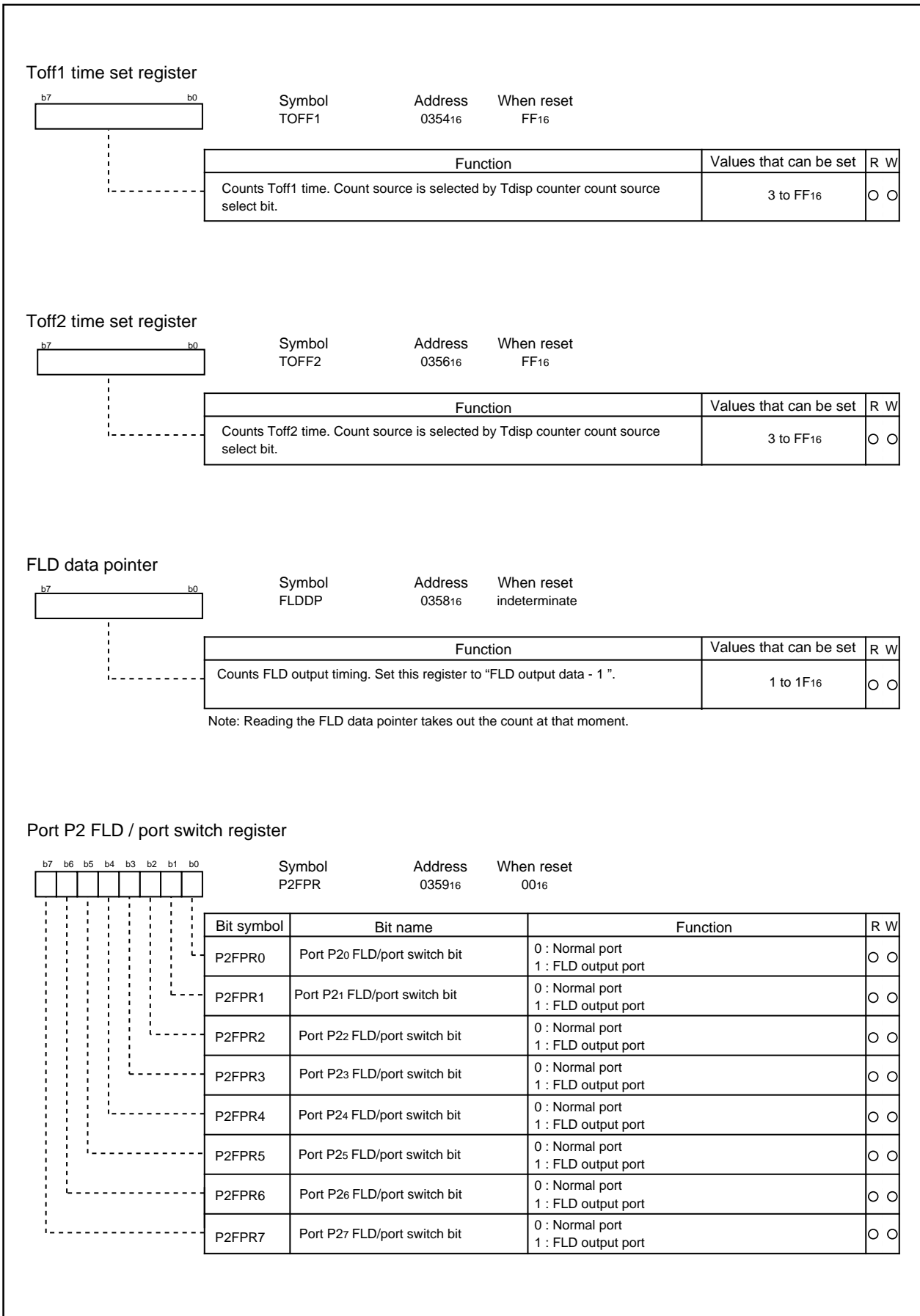


Figure 40. FLDC-related Register(2)



## FLD controller

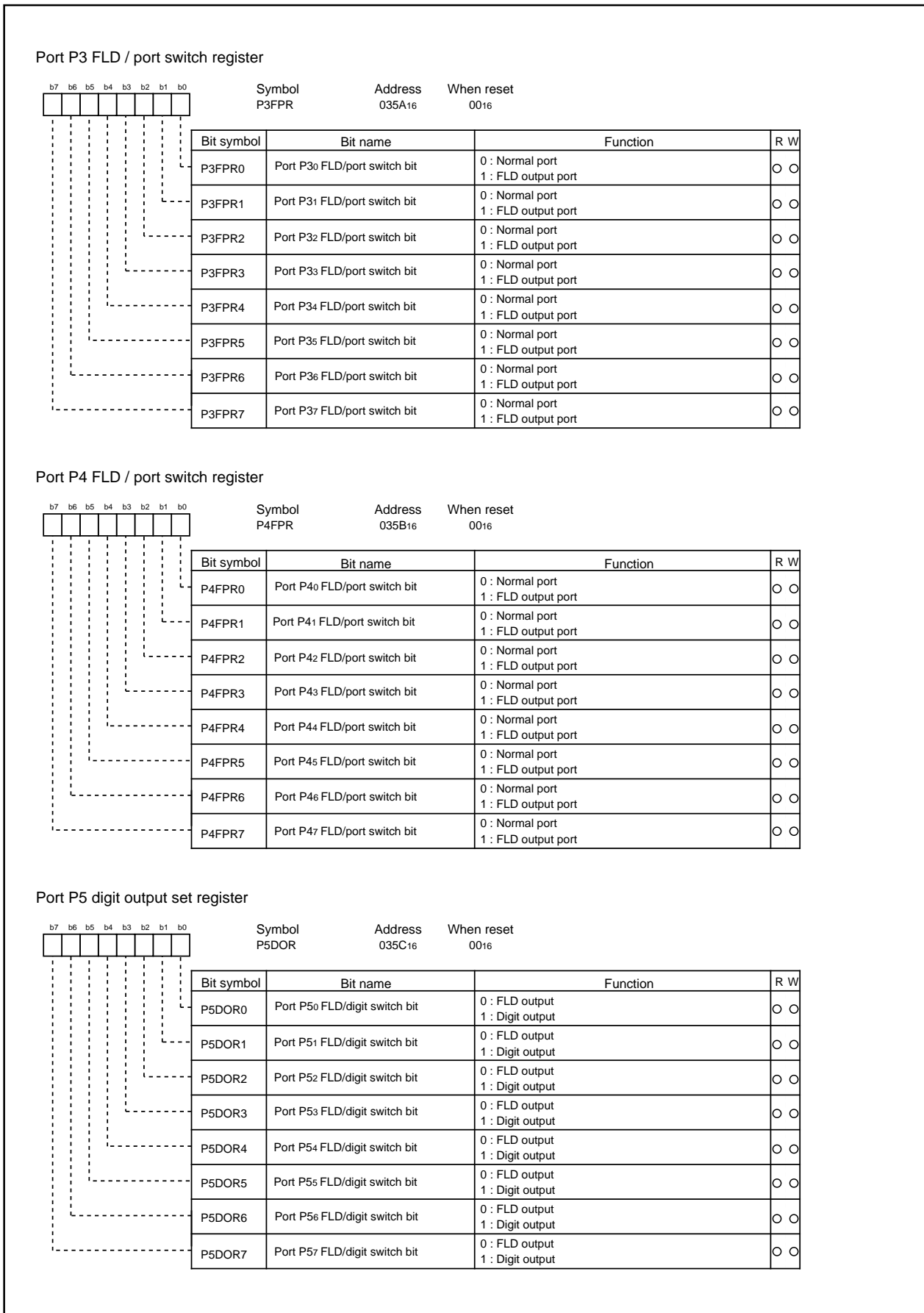


Figure 41. FLDC-related Register(3)

## FLD controller

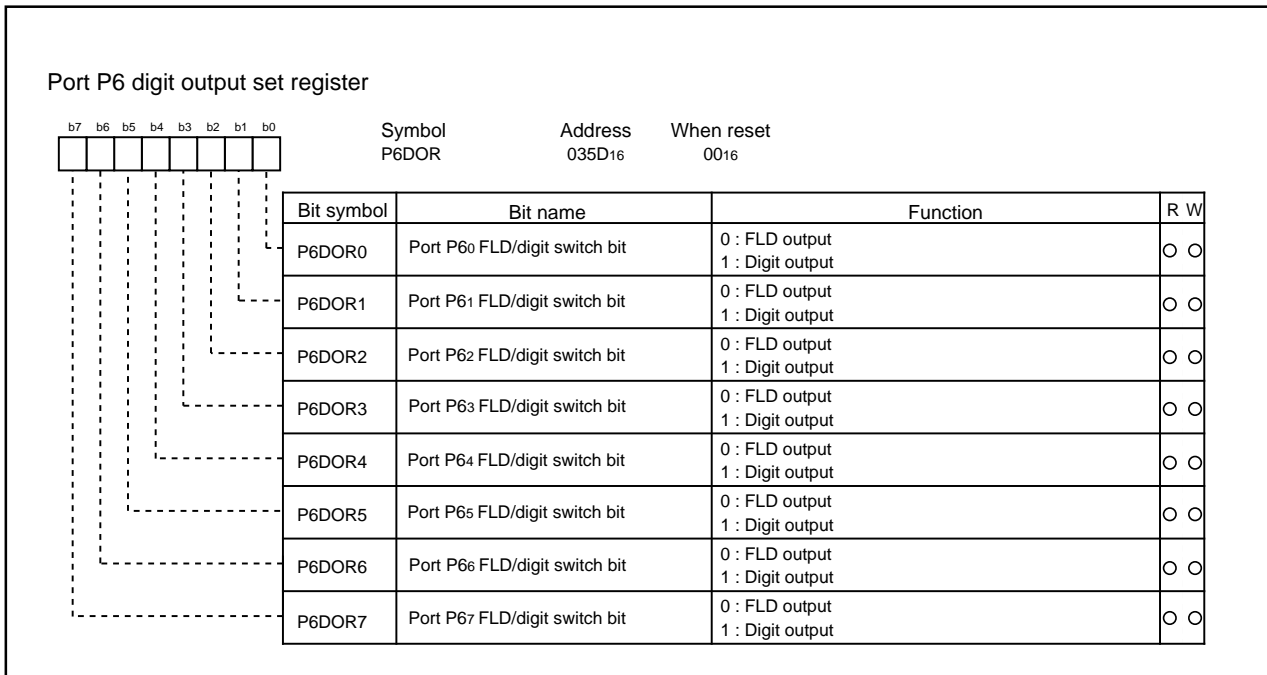


Figure 42. FLDC-related Register(4)

## FLD controller

## FLD automatic display pins

P0 to P6 are the pins capable of automatic display output for the FLD. The FLD start operating by setting the automatic display control bit (bit 0 at address 035016) to "1". There is the FLD output function that outputs RAM contents from the port every timing or the digit output function that drives the port high with digit timing. The FLD can be displayed using the FLD output for the segments and the digit or FLD output for the digits. When using the FLD output for the digits, be sure to write digit display patterns to the RAM in advance. The remaining segment and digit lines can be used as general-purpose ports. Settings of each port are shown below.

Table 15. Pins in FLD Automatic Display Mode

Port Name	Automatic Display Pins	Setting Method
P5, P6	FLD <sub>0</sub> to FLD <sub>15</sub>	The individual bits of the digit output set register (address 035C16, 035D16) can set each pin either FLD port ("0") or digit port ("1"). When the pins are set for the digit port, the digit pulse output function is enabled, so the digit pulses can always be output regardless the value of FLD automatic display RAM.
P0, P1	FLD <sub>16</sub> to FLD <sub>31</sub>	FLD exclusive use port (automatic display control bit (bit 0 of address 035016)="1")
P2, P3, P44 to P43	FLD <sub>32</sub> to FLD <sub>51</sub>	The individual bits of the FLD/port switch register (addresses 035916 to 035B16) can set each pin to either FLD port ("1") or general-purpose port ("0").
P44 to P47	FLD <sub>52</sub> to FLD <sub>55</sub>	The individual bits of the FLD/port switch register (address 035B16) can set each pin to either FLD port ("1") or general-purpose port ("0"). The digit pulse output function turns to available, and the digit pulse can output by setting of the FLD output set register (address 035116). The port output format is the CMOS output. When using the port as a display pin, a driver must be installed externally.

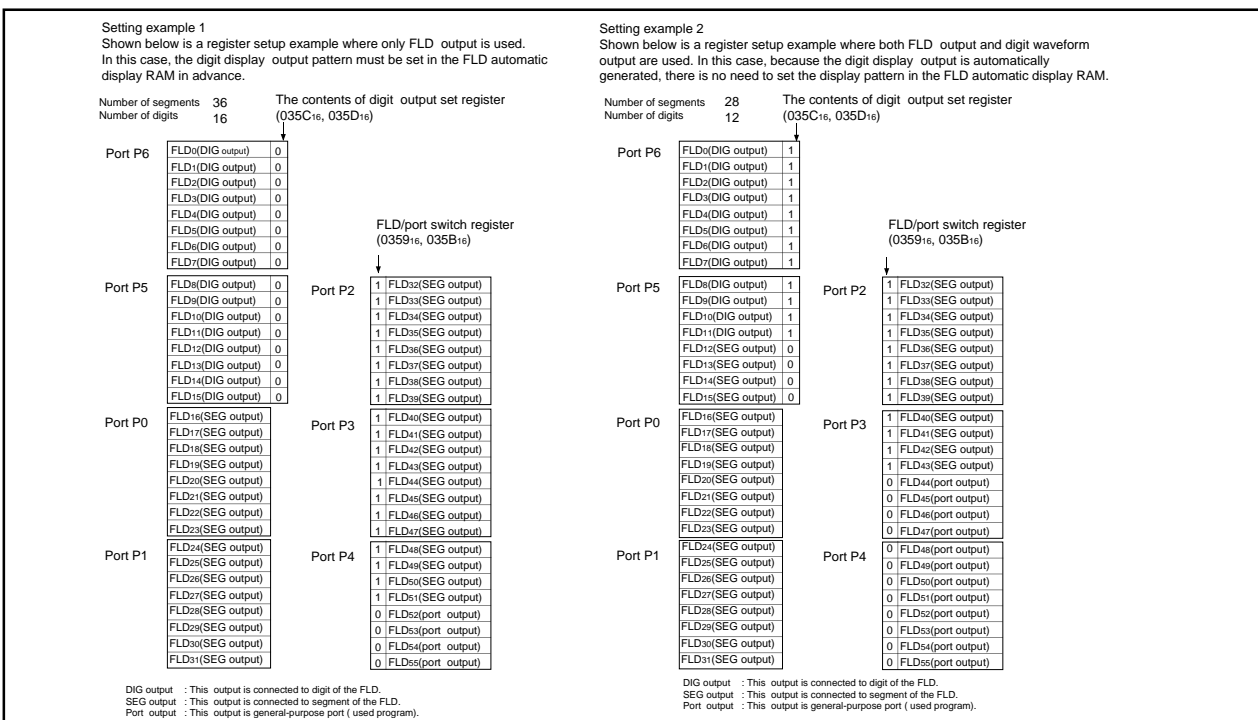


Figure 43. Segment/Digit Setting Example

## FLD automatic display RAM

The FLD automatic display RAM uses the 224 bytes of addresses 0500<sub>16</sub> to 05DF<sub>16</sub>. For FLD, the 3 modes of 16-timing ordinary mode, 16-timing•gradation display mode and 32-timing mode are available depending on the number of timings and the use/not use of gradation display.

The automatic display RAM in each mode is as follows:

### (1) 16-timing•Ordinary Mode

This mode is used when the display timing is 16 or less. The 112 bytes of addresses 0570<sub>16</sub> to 05DF<sub>16</sub> are used as a FLD display data store area. Because addresses 0500<sub>16</sub> to 056F<sub>16</sub> are not used as the automatic display RAM, they can be the ordinary RAM.

### (2) 16-timing•Gradation Display Mode

This mode is used when the display timing is 16 or less, in which mode each segment can be set for dark or bright display. The 224 bytes of addresses 0500<sub>16</sub> to 05DF<sub>16</sub> are used. The 112 bytes of addresses 0570<sub>16</sub> to 05DF<sub>16</sub> are used as an FLD display data store area, while the 112 bytes of addresses 0500<sub>16</sub> to 056F<sub>16</sub> are used as a gradation display control data store area.

### (3) 32-timing Mode

This mode is used when the display timing is 16 or greater. This mode can be used for up to 32-timing. The 224 bytes of addresses 0500<sub>16</sub> to 05DF<sub>16</sub> are used as an FLD display data store area.

The FLD data pointer (address 0358<sub>16</sub>) is a register to count display timings. This pointer has a reload register and when the terminal count is reached, it starts counting over again after being reloaded with the initial count. Make sure the timing count – 1 is set to the FLD data pointer. When writing data to this address, the data is written to the FLD data pointer reload register; when reading data from this address, the value in the FLD data pointer is read.

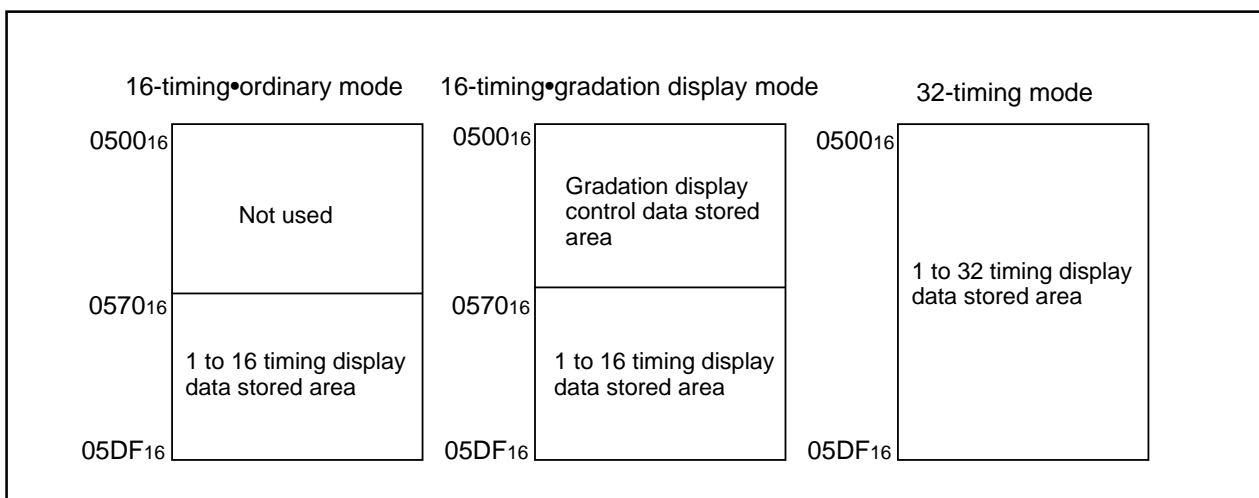


Figure 44. FLD Automatic Display RAM Assignment

## FLD controller

## Data setup

**(1) 16-timing•Ordinary Mode**

The area of addresses 0570<sub>16</sub> to 05DF<sub>16</sub> are used as a FLD automatic display RAM.

When data is stored in the FLD automatic display RAM, the last data of FLD port P4 is stored at address 0570<sub>16</sub>, the last data of FLD port P3 is stored at address 0580<sub>16</sub>, the last data of FLD port P2 is stored at address 0590<sub>16</sub>, the last data of FLD port P1 is stored at address 05A0<sub>16</sub>, the last data of FLD port P0 is stored at address 05B0<sub>16</sub>, the last data of FLD port P5 is stored at address 05C0<sub>16</sub>, and the last data of FLD port P6 is stored at address 05D0<sub>16</sub>, to assign in sequence from the last data respectively.

The first data of the FLD port P4, P3, P2, P1, P0, P5, and P6 is stored at an address which adds the value of (the timing number – 1) to the corresponding address 0570<sub>16</sub>, 0580<sub>16</sub>, 0590<sub>16</sub>, 05A0<sub>16</sub>, 05B0<sub>16</sub>, 05C0<sub>16</sub> and 05DF<sub>16</sub>.

Set the FLD data pointer reload register to the value given by the number of digits – 1.

**(2) 16-timing•Gradation Display Mode**

Display data setting is performed in the same way as that of the 16-timing•ordinary mode. Gradation display control data is arranged at an address resulting from subtracting 0070<sub>16</sub> from the display data store address of each timing and pin. Bright display is performed by setting “0”, and dark display is performed by setting “1” .

**(3) 32-timing Mode**

The area of addresses 0500<sub>16</sub> to 05DF<sub>16</sub> are used as a FLD automatic display RAM.

When data is stored in the FLD automatic display RAM, the last data of FLD port P4 is stored at address 0500<sub>16</sub>, the last data of FLD port P3 is stored at address 0520<sub>16</sub>, the last data of FLD port P2 is stored at address 0540<sub>16</sub>, the last data of FLD port P1 is stored at address 0560<sub>16</sub>, the last data of FLD port P0 is stored at address 0580<sub>16</sub>, the last data of FLD port P5 is stored at address 05A0<sub>16</sub>, and the last data of FLD port P6 is stored at address 05C0<sub>16</sub>, to assign in sequence from the last data respectively.

The first data of the FLD port P4, P3, P2, P0, P1, P5, and P6 is stored at an address which adds the value of (the timing number – 1) to the corresponding address 0500<sub>16</sub>, 0520<sub>16</sub>, 0540<sub>16</sub>, 0560<sub>16</sub>, 0580<sub>16</sub>, 05A0<sub>16</sub> and 05C0<sub>16</sub>.

Set the FLD data pointer reload register to the value given by the number of digits - 1.

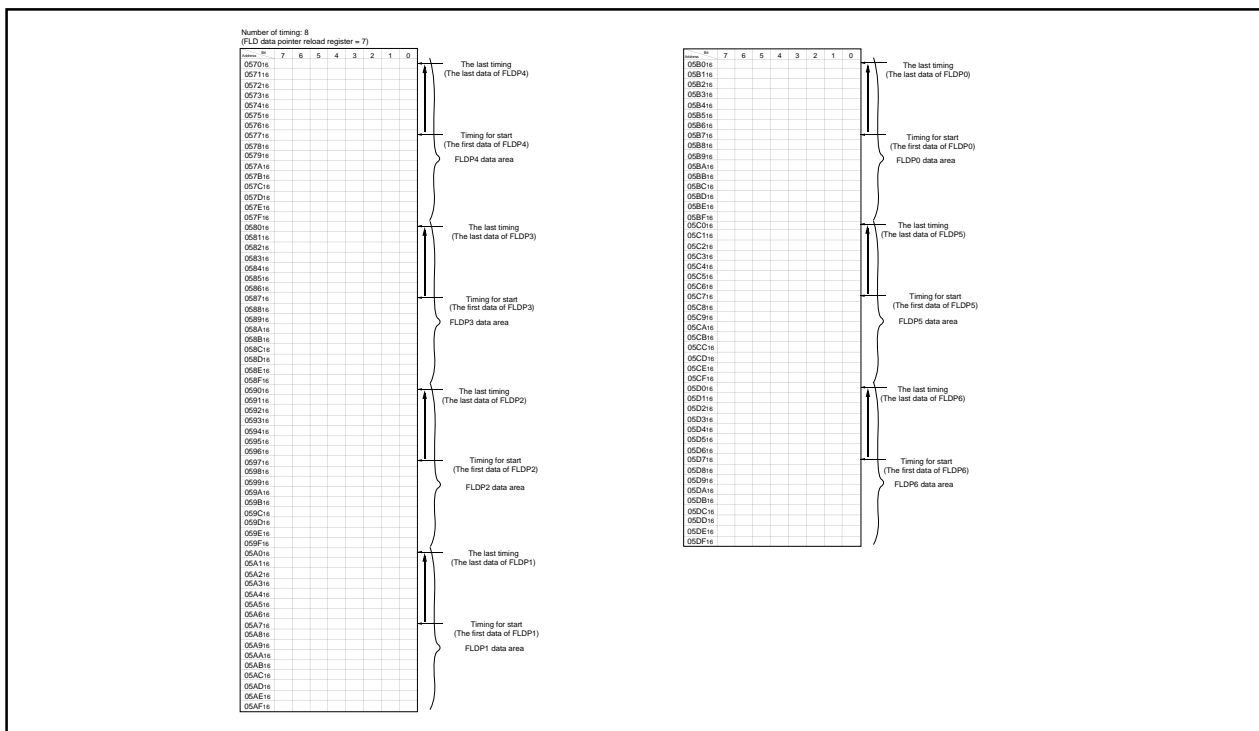


Figure 45. Example of Using the FLD Automatic Display RAM in 16-timing•Ordinary Mode

FLD controller

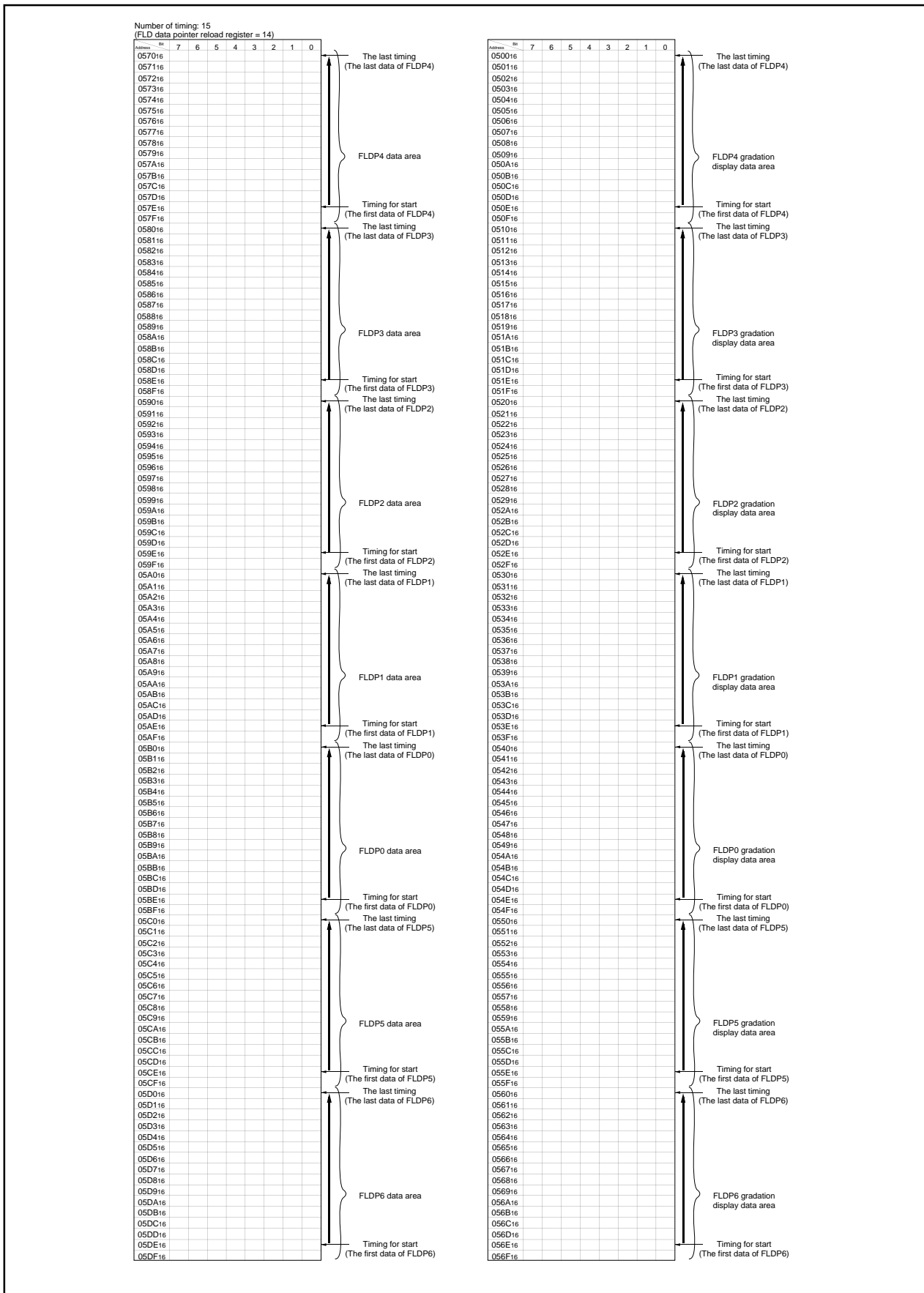


Figure 46. Example of Using the FLD Automatic Display RAM in 16-timing\*Gradation Display Mode

FLD controller

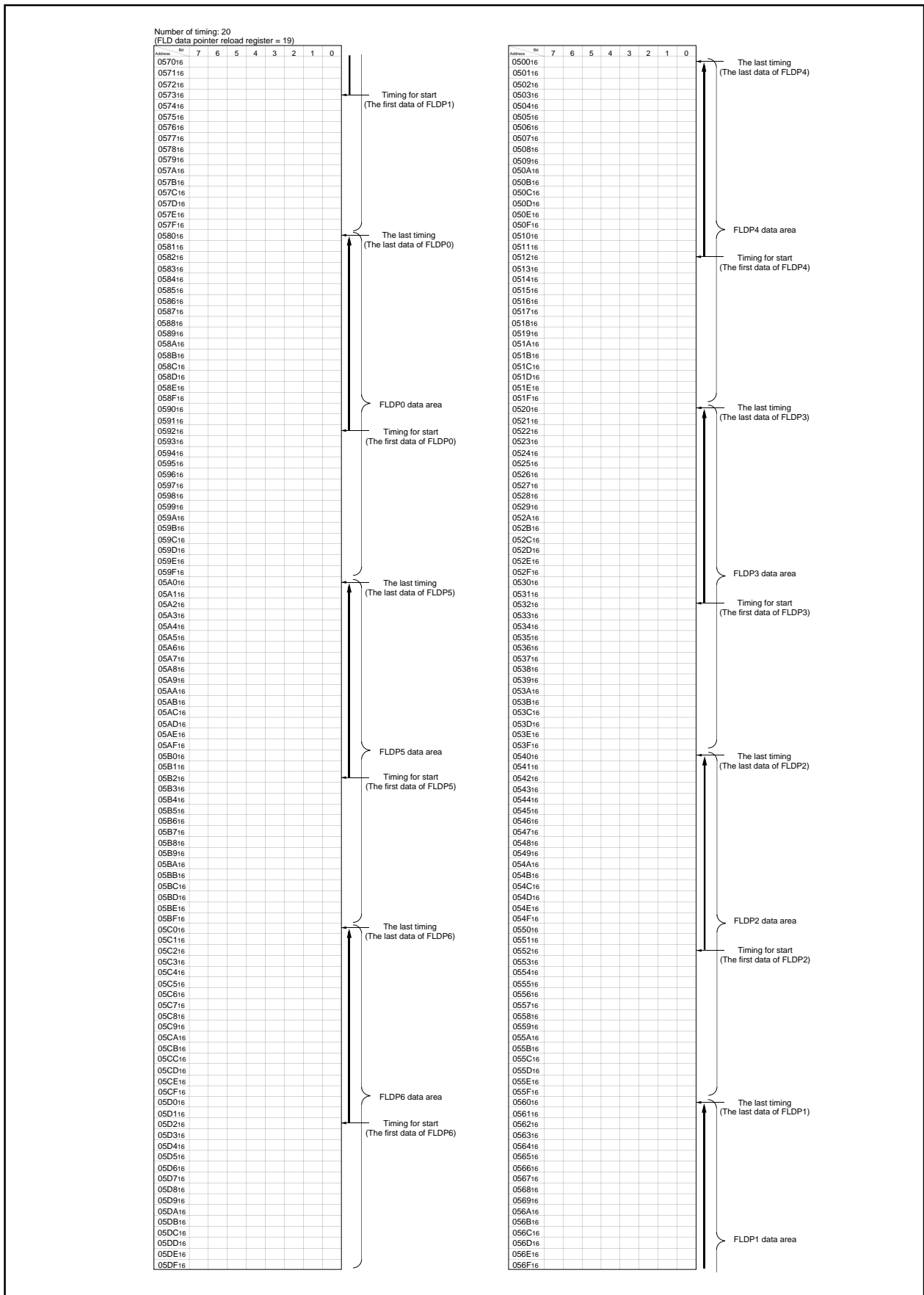


Figure 47. Example of Using the FLD Automatic Display RAM in 32-timing Mode

## Timing setting

Each timing is set by the FLDC mode register, Tdisp time set register, Toff1 time set register, and Toff2 time set register.

### •Tdisp time setting

The Tdisp time represents the length of display timing. In non-gradation display mode, it consists of a FLD display output period and a Toff1 time. In gradation display mode, it consists of the display output period and Toff1 time plus a low signal output period for dark display. Set the Tdisp time by the Tdisp counter count source select bit of the FLDC mode register and the Tdisp time set register. Supposing that the value of the Tdisp time set register is n, the Tdisp time is represented as  $T_{disp} = (n+1) \times t$  (t: count source). When the Tdisp counter count source select bit of the FLDC mode register is "0" and the value of the Tdisp time set register is 200 (C8<sub>16</sub>), the Tdisp time is:  $T_{disp} = (200+1) \times 3.2$  (at  $X_{IN} = 10$  MHz) = 643  $\mu$ s. When reading the Tdisp time set register, the value in the counter is read out.

### •Toff1 time setting

The Toff1 time represents a non-output (low signal output) time to prevent blurring of FLD, and to dim the display. Use the Toff1 time set register to set this Toff1 time. Make sure the value set to Toff1 is smaller than Tdisp and Toff2. Supposing that the value of the Toff1 time set register is n1, the Toff1 time is represented as  $T_{off1} = n1 \times t$ . When the Tdisp counter count source select bit of the FLDC mode register is "0" and the value of the Toff1 time set register is 30 (1E<sub>16</sub>),  $T_{off1} = 30 \times 3.2$  (at  $X_{IN} = 10$  MHz) = 96  $\mu$ s.

### •Toff2 time setting

The Toff2 time is provided for dark display. For bright display, the FLD display output remains effective until the counter that is counting Tdisp reaches the terminal count. For dark display, however, "L" (or "off") signal is output when the counter that is counting Toff2 reaches the terminal count. This Toff2 time setting is valid only for FLD ports which are in the gradation display mode and whose gradation display control RAM value is "1".

Set the Toff2 time by the Toff2 time set register. Make sure the value set to Toff2 is smaller than Tdisp but larger than Toff1. Supposing that the value of the Toff2 time set register is n2, the Toff2 time is represented as  $T_{off2} = n2 \times t$ . When the Tdisp counter count source select bit of the FLDC mode register is "0" and the value of the Toff2 time set register is 180 (B4<sub>16</sub>),  $T_{off2} = 180 \times 3.2$  (at  $X_{IN} = 10$  MHz) = 576  $\mu$ s.

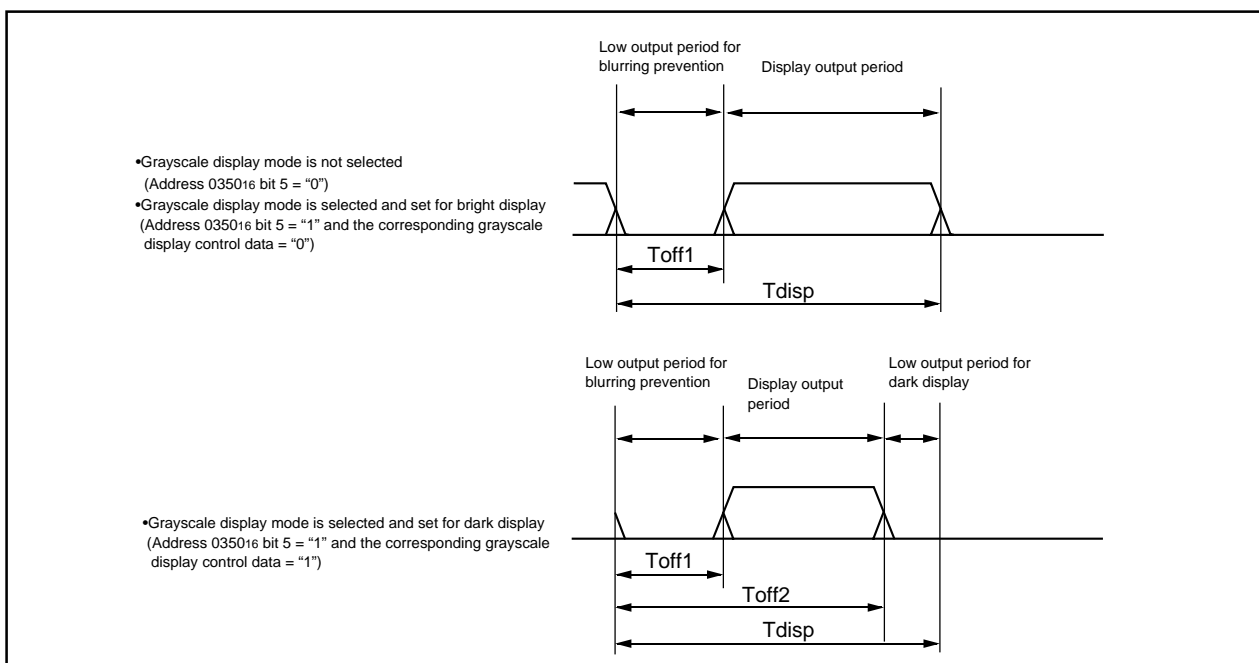


Figure 48. FLDC Timing



## FLD automatic display start

Automatic display starts by setting both the automatic display control bit (bit 0 of address 0350<sub>16</sub>) and the display start bit (bit 1 of address 0350<sub>16</sub>) to "1". The RAM content at a location apart from the start address of the automatic display RAM for each port by (FLD data pointer (address 0358<sub>16</sub>) – 1) is output to each port. The FLD data pointer (address 0358<sub>16</sub>) counts down in the T<sub>disp</sub> interval. When the count "FF<sub>16</sub>" is reached, the pointer is reloaded and starts counting over again. Before setting the display start bit (bit 1 of address 0350<sub>16</sub>) to "1", be sure to set the FLD/port switch register, FLD/DIG switch register, FLDC mode register, T<sub>disp</sub> time set register, T<sub>off1</sub> time set register, T<sub>off2</sub> time set register, and FLD data pointer.

During FLD automatic display, bit 1 of the FLDC mode register (address 0350<sub>16</sub>) always keeps "1", and FLD automatic display can be interrupted by writing "0" to bit 1.

## Key-scan and interrupt

Either a FLD digit interrupt or FLD blanking interrupt can be selected using the Tscan control bits (bits 2, 3 of address 0350<sub>16</sub>).

The FLD digit interrupt is generated when the T<sub>off1</sub> time in each timing expires (at rising edge of digit output). Key scanning that makes use of FLD digits can be achieved using each FLD digit interrupt. To use FLD digit interrupts for key scanning, follow the procedure described below.

- (1) Read the port value each time the interrupt occurs.
- (2) The key is fixed on the last digit interrupt.

The digit positions output can be determined by reading the FLD data pointer (address 0358<sub>16</sub>).

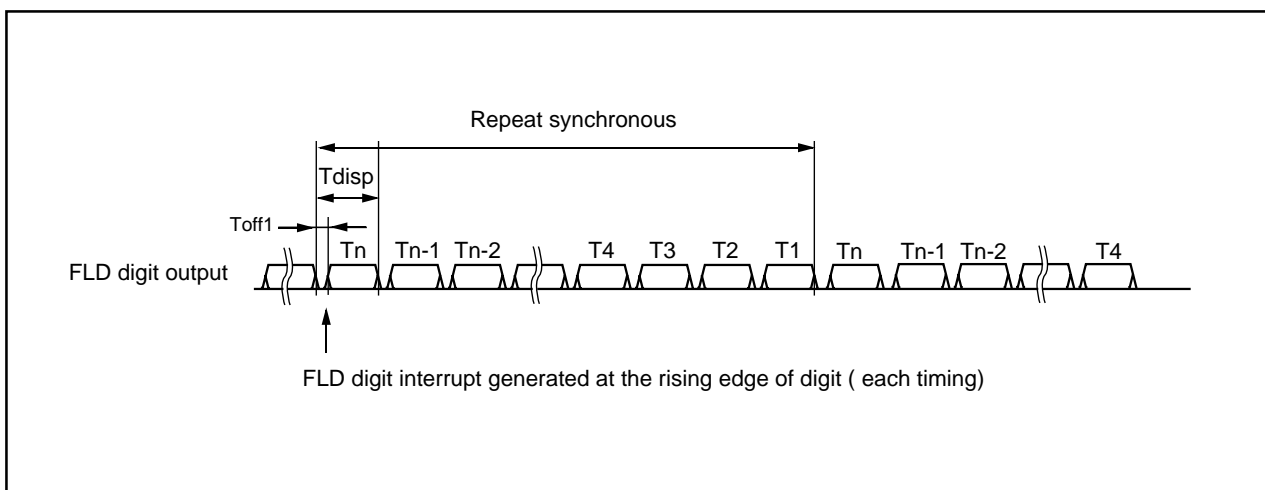


Figure 49. Timing using digit interrupt

## FLD controller

The FLD blanking interrupt is generated when the FLD data pointer (address 035816) reaches "FF16". The FLD automatic display output is turned off for a duration of  $1 \times T_{disp}$ ,  $2 \times T_{disp}$ , or  $3 \times T_{disp}$  depending on post-interrupt settings. During this time, key scanning that makes use of FLD segments can be achieved.

When a key-scan is performed with the segment during key-scan blanking period  $T_{scan}$ , take the following sequence:

1. Write "0" to bit 0 of the FLDC mode register (address 035016).
2. Set the port corresponding to the segment for key-scan to the output port.
3. Perform the key-scan.
4. After the key-scan is performed, write "1" to bit 0 of FLDC mode register (address 035016).

•**Note:**

When performing a key-scan according to the above steps 1 to 4, take the following points into consideration.

1. Do not set "0" in bit 1 of the FLDC mode register (address 035016).
2. Do not set "1" in the ports corresponding to digits.

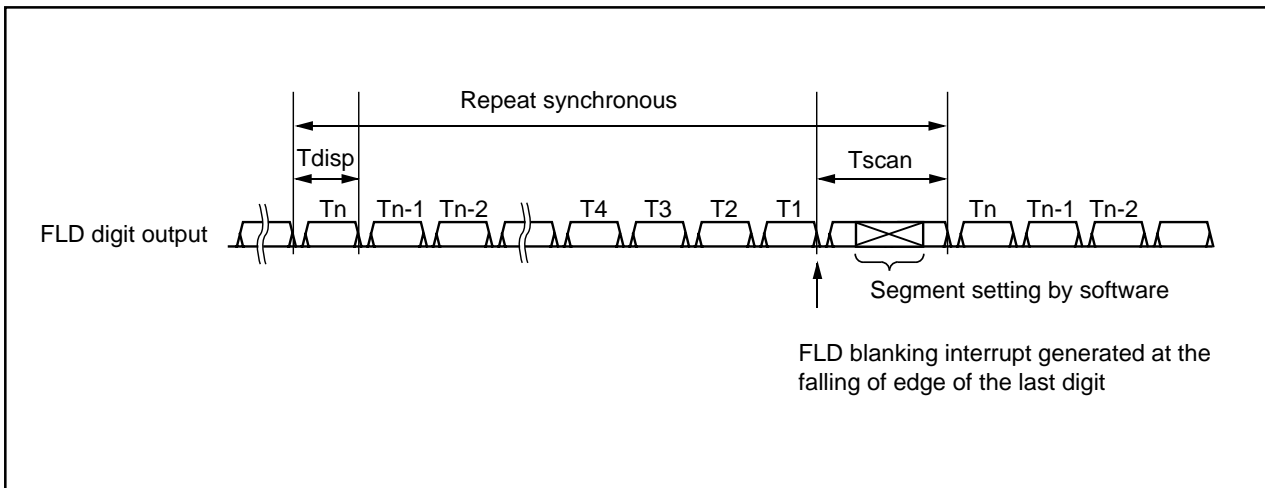


Figure 50. Timing using FLD blanking interrupt

## P44 to P47 Expansion Function

P44 to P47 are CMOS output-type ports. FLD digit outputs can be increased as many as 16 lines by connecting a 4-bit to 16-bit decoder to these ports. P44 to P47 have the function to allow for connection to a 4-bit to 16-bit decoder.

### (1) P44 to P47 Toff invalid Function

This function disables the Toff1 time and Toff2 time and outputs display data for the duration of Tdisp. (See Figure 51.) This can be accomplished by setting the P44 to P47 Toff disable bit (address 0350<sub>16</sub> bit 2) to "1".

Unlike the Toff section generate/not generate function, this function disables all display data.

### (2) Dimmer signal output Function

This function allows a dimmer signal creation signal to be output from DIMOUT (P97). The dimmer function can be materialized by controlling the decoder with this signal. (See Figure 51.) This function can be set by writing P97 dimmer output control bit (bit 4 of address 0351<sub>16</sub>) to "1".

### (3) P44 to P47 FLD Output Reverse Bit

P44 to P47 are provided with a function to reverse the polarity of the FLD output. This function is useful in adjusting the polarity when using an externally installed driver.

The output polarity can be reversed by setting bit 0 of the FLD output control register (address 0351<sub>16</sub>) to "1".

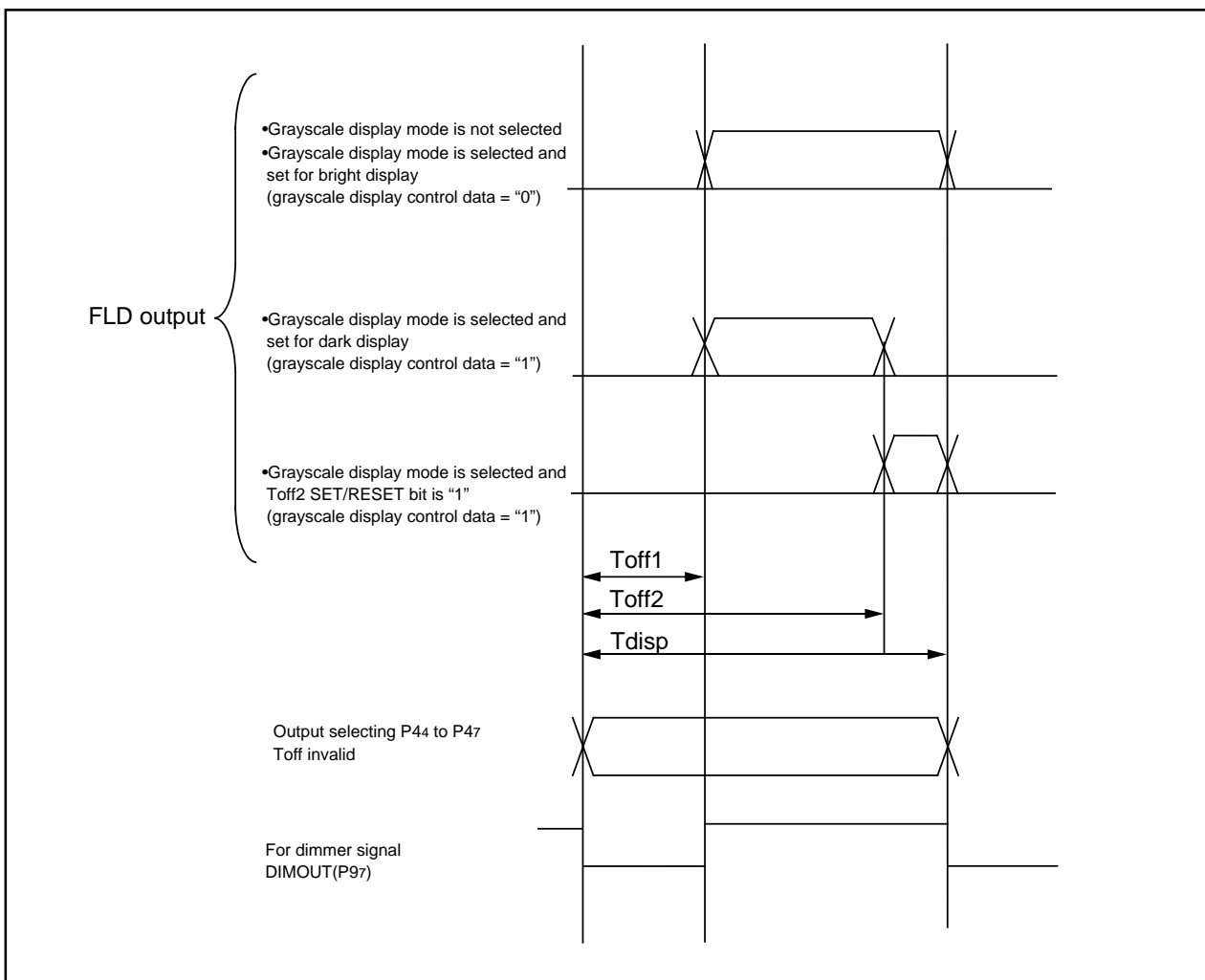


Figure 51. P4 to P47 FLD Output pulses

## Toff section generate/not generate Function

The function is for reduction of useless noises which generated as every switching of ports, because of the combined capacity of among FLD ports. In case the continuous data output to each FLD ports, the Toff1 section of the continuous parts is not generated. (See Figure 52)

If it needs Toff1 section on FLD pulses, set "CMOS ports: section of Toff generate / not generate bit" to "1" and set "high-breakdown-voltage ports: section of Toff generate / not generate bit" to "1". High-breakdown-voltage ports (P5, P6, P3, P2, P1, P0, P40 to P43, total 52 pins) generate Toff1 section, by setting "high-breakdown-voltage ports: section of Toff generate / not generate bit" to "1".

The CMOS ports ( P44 to P47, total 4 pins ) generate Toff1 section, by setting "high-breakdown-voltage ports: section of Toff generate / not generate bit" to "1".

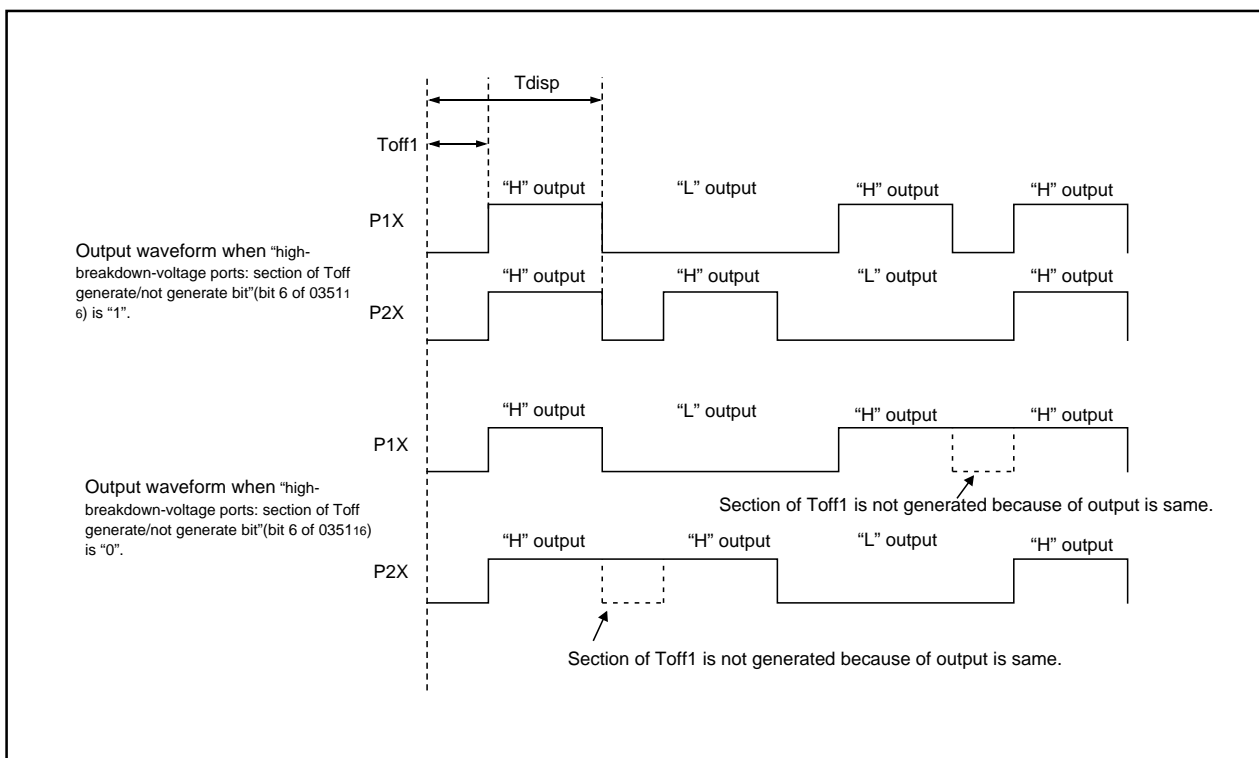


Figure 52. Toff Section Generated/not generated Function

## Toff2 SET/RESET change bit

In gradation display mode, the values set by the Toff2 time set register (TOFF2) are effective. When the FLD output control register (bit 7 of address 035116) in the initial state = "0", RAM data is output to the FLD output ports (SET) at the time that is set by TOFF1 and is turned to "0" (RESET) at the time that is set by TOFF2. When bit 7 = "1", RAM data is output (SET) at the time that is set by TOFF2 and is turned to "0" (RESET) when the Tdisp time expires.

## Digit pulses output Function

P50 to P57 and P60 to P67 allow digit pulses to be output using the FLD/digit switch register. Set the digit output set register by writing as many consecutive 1s as the timing count from P60. The contents of FLD automatic display RAM for the ports that have been selected for digit output are disabled, and the pulse shown in Figure 53 is output automatically. In gradation display mode use,  $T_{off2}$  time becomes effective for the port which selected digit output. Because the contents of FLD automatic display RAM are disabled, the segment data can be changed easily even when segment data and digit data coexist at the same address in the FLD automatic display RAM.

This function is effective in 16-timing normal mode and 16-timing gradation display mode. If a value is set exceeding the timing count (FLD data pointer reload register's set value + 1) for any port, the output of such port is "L".

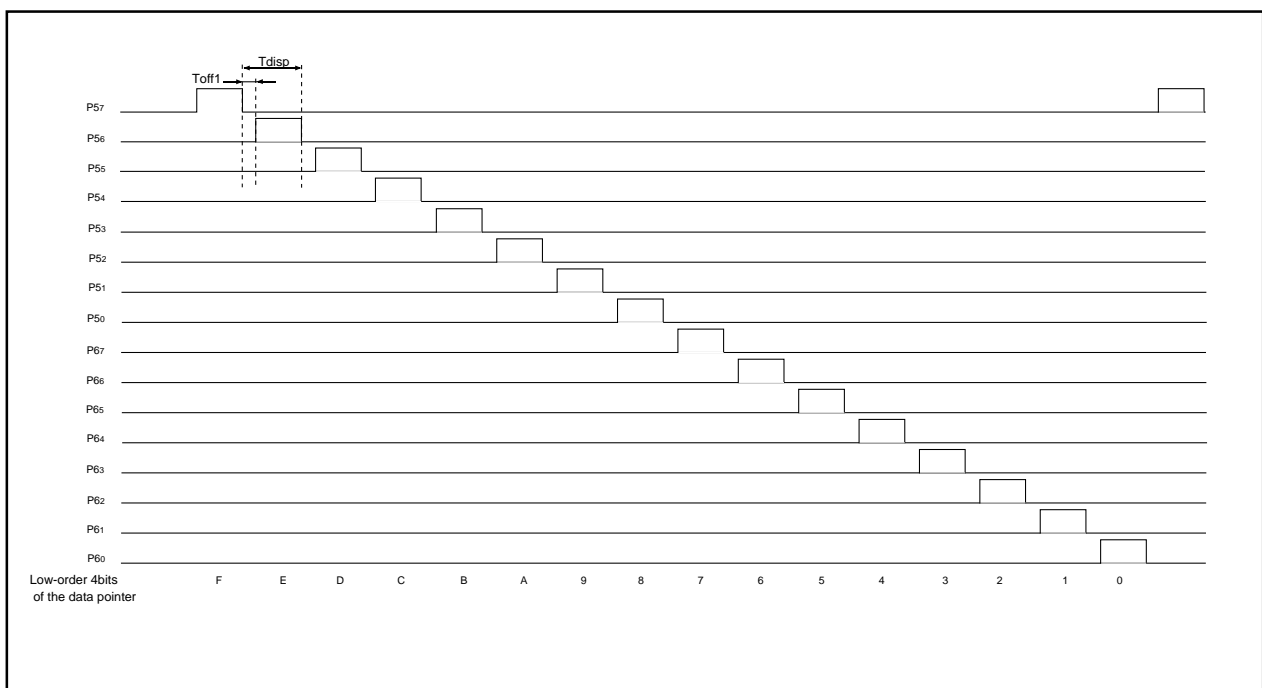


Figure 53. Digit Pulses Output Function

## Timer

## Timer

There are eight 16-bit timers. These timers can be classified by function into timers A (five) and timers B (three). All these timers function independently. Figure 54 shows the block diagram of timers.

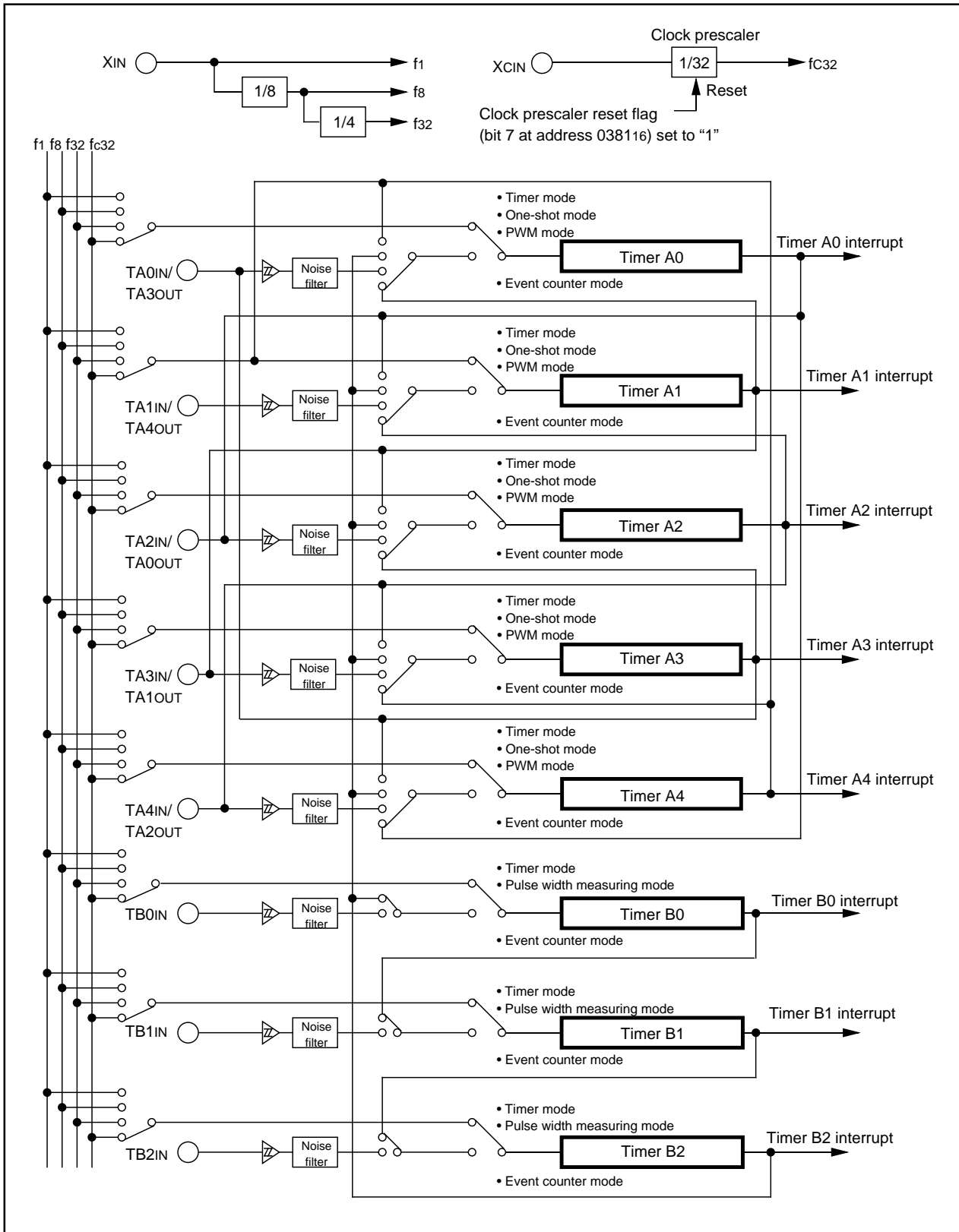


Figure 54. Timer block diagram

## TimerA

## Timer A

Figure 55 shows the block diagram of timer A. Figures 56 to 58 show the timer A-related registers.

Except in event counter mode, timers A0 through A4 all have the same function. Use the timer Ai mode register ( $i = 0$  to 4) bits 0 and 1 to choose the desired mode.

Timer A has the four operation modes listed as follows:

- Timer mode: The timer counts an internal count source.
- Event counter mode: The timer counts pulses from an external source or a timer's over flow.
- One-shot timer mode: The timer stops counting when the count reaches "0000<sub>16</sub>".
- Pulse width modulation (PWM) mode: The timer outputs pulses of a given width.

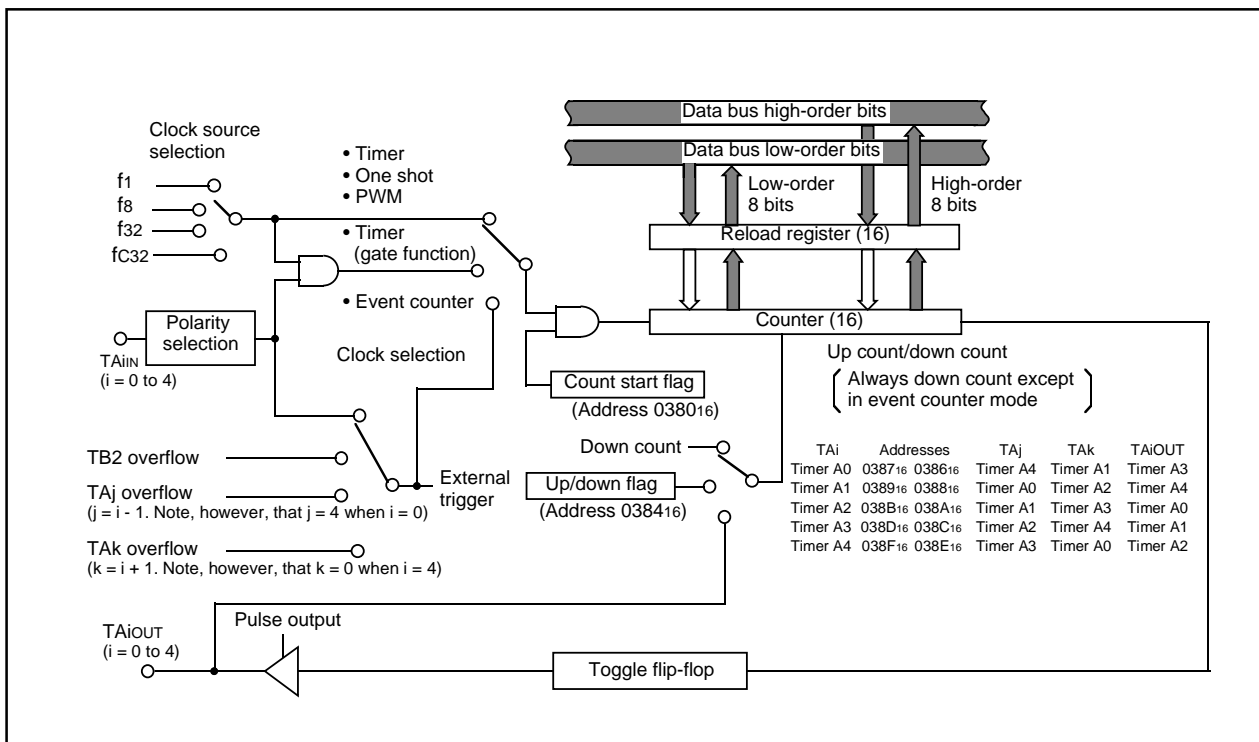


Figure 55. Block diagram of timer A

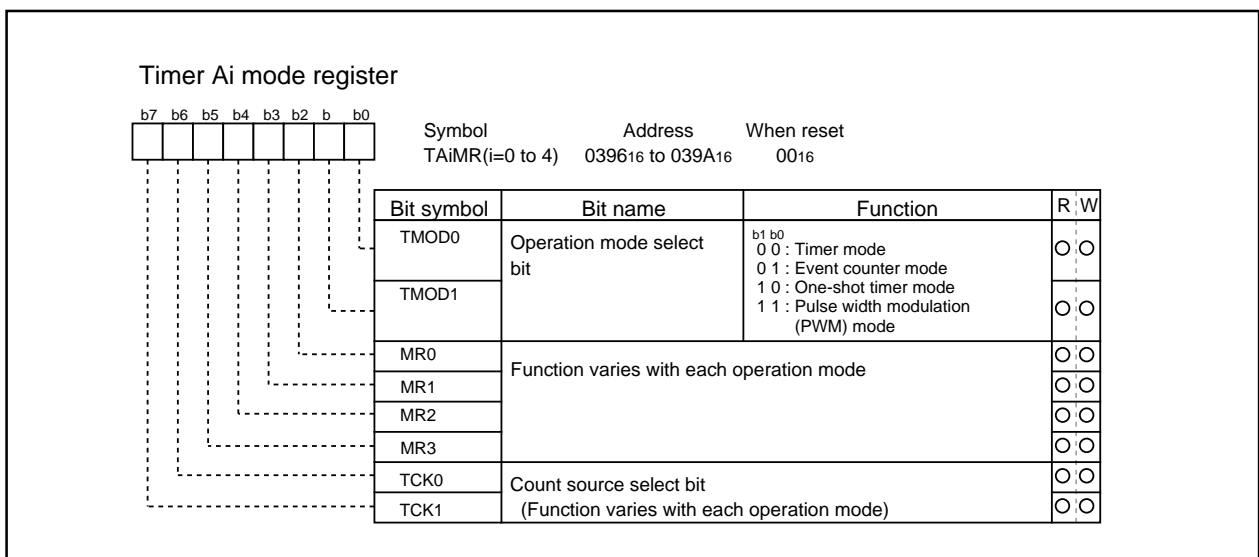


Figure 56. Timer A-related registers (1)

## TimerA

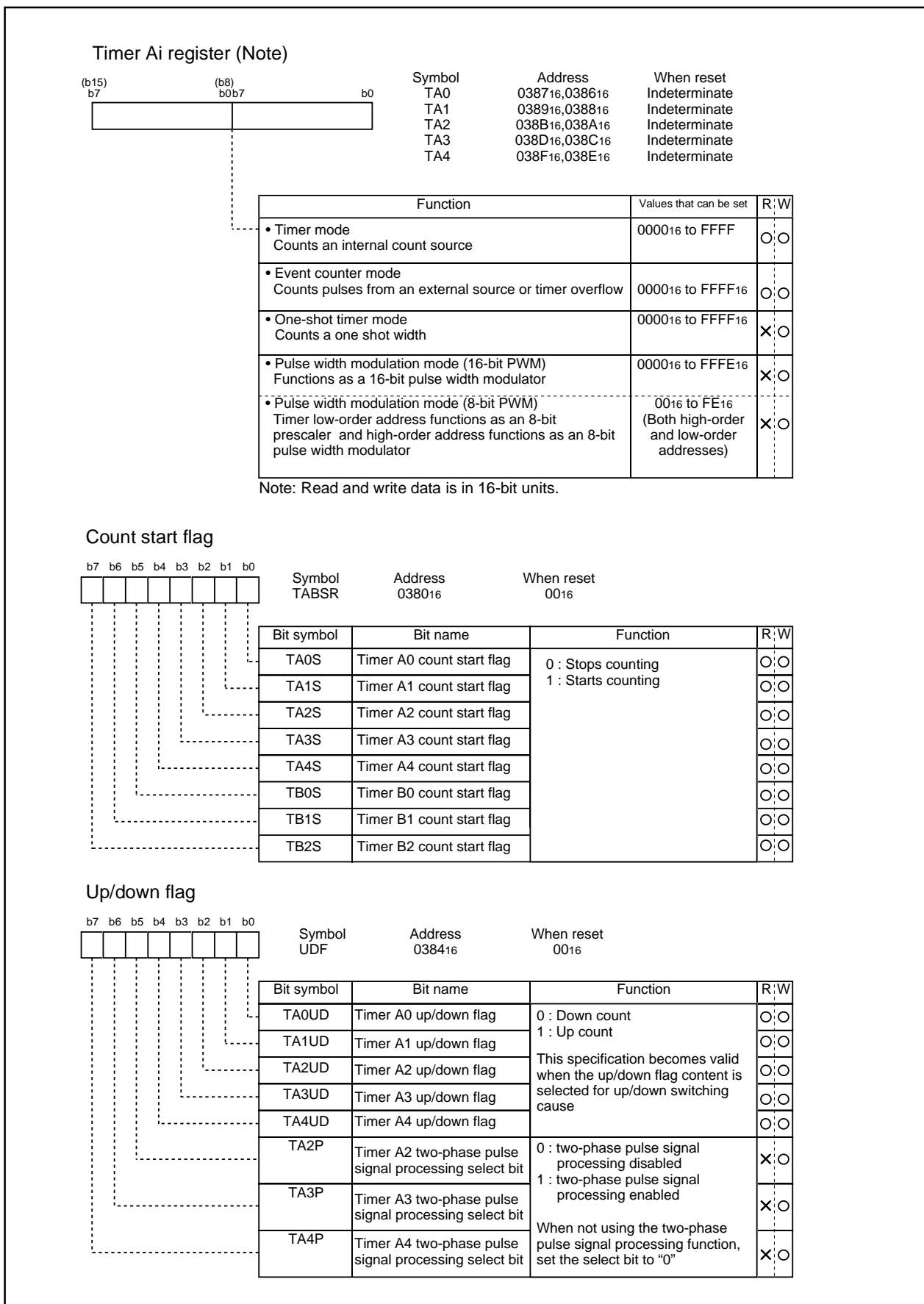


Figure 57. Timer A-related registers (2)



## TimerA

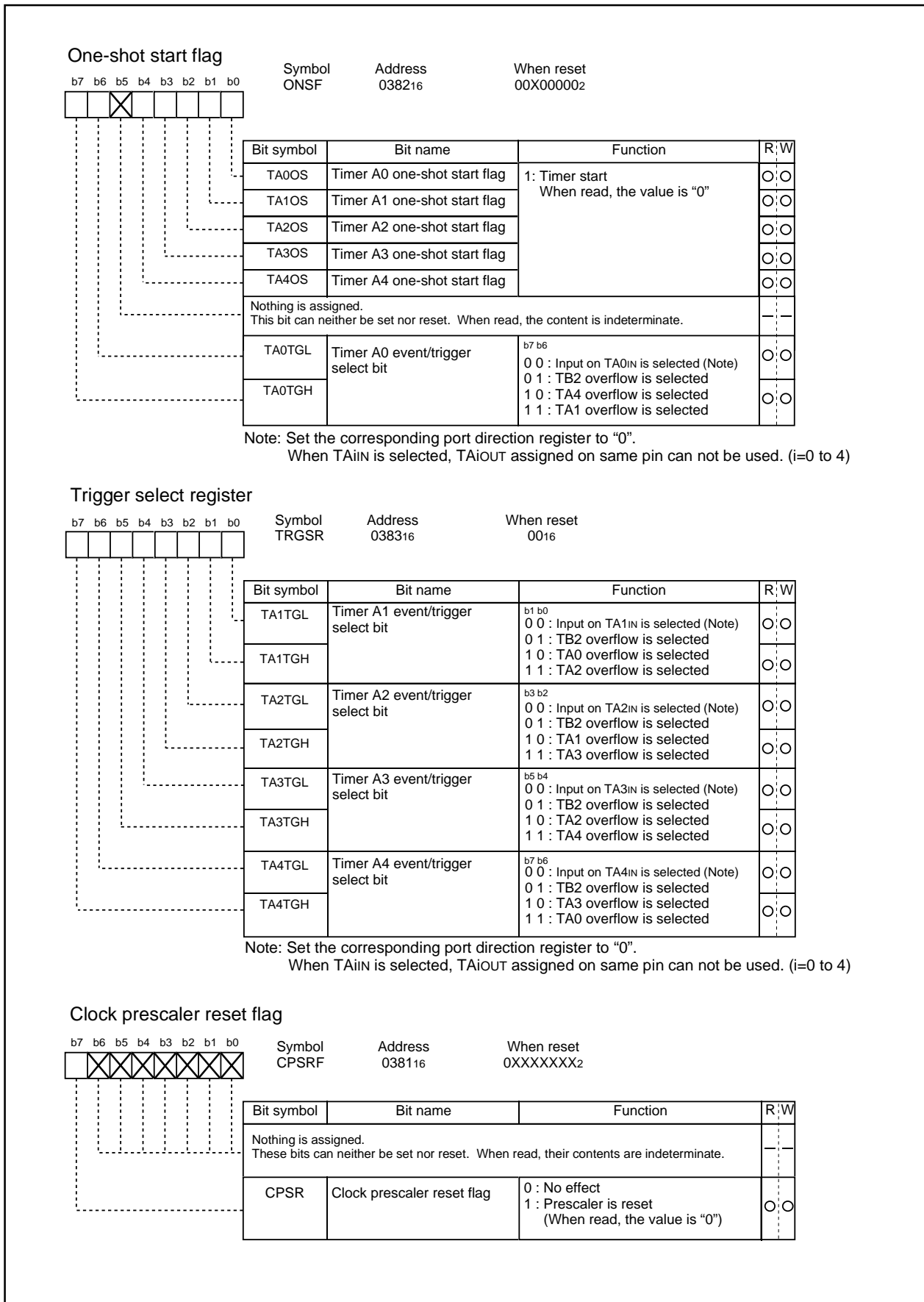


Figure 58 Timer A-related registers (3)

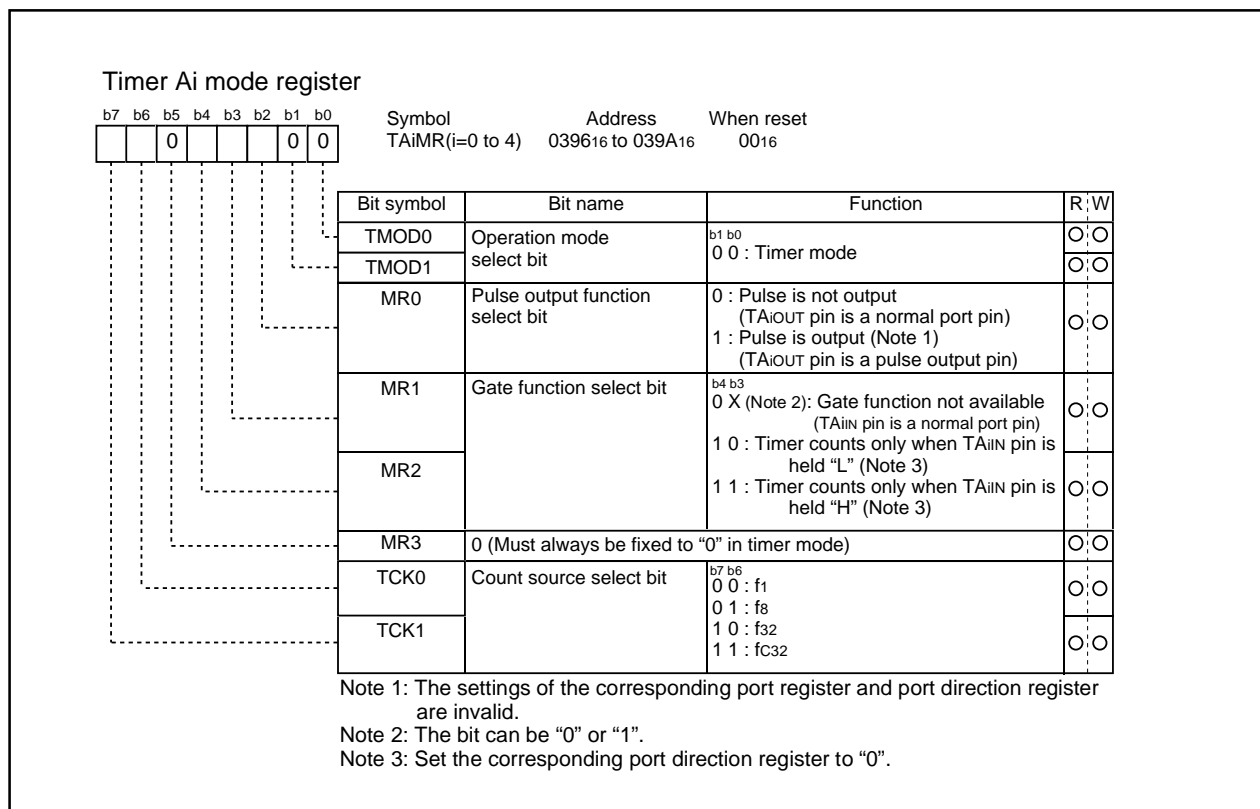
## TimerA

**(1) Timer mode**

In this mode, the timer counts an internally generated count source. (See Table16.) Figure 59 shows the timer Ai mode register in timer mode.

**Table 16. Specifications of timer mode**

Item	Specification
Count source	f1, f8, f32, fC32
Count operation	<ul style="list-style-type: none"> <li>Down count</li> <li>When the timer underflows, it reloads the reload register contents before continuing counting</li> </ul>
Divide ratio	$1/(n+1)$ n : Set value
Count start condition	Count start flag is set (= 1)
Count stop condition	Count start flag is reset (= 0)
Interrupt request generation timing	When the timer underflows
TAiIN pin function	Programmable I/O port or gate input
TAiOUT pin function	Programmable I/O port or pulse output
Read from timer	Count value can be read out by reading timer Ai register
Write to timer	<ul style="list-style-type: none"> <li>When counting stopped When a value is written to timer Ai register, it is written to both reload register and counter</li> <li>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)</li> </ul>
Select function	<ul style="list-style-type: none"> <li>Gate function Counting can be started and stopped by the TAiIN pin's input signal</li> <li>Pulse output function Each time the timer underflows, the TAiOUT pin's polarity is reversed</li> </ul>

**Figure 59. Timer Ai mode register in timer mode**

## (2) Event counter mode

In this mode, the timer counts an external signal or an internal timer's overflow. Timers A0 and A1 can count a single-phase external signal. Timers A2, A3, and A4 can count a single-phase and a two-phase external signal. Table 17 lists timer specifications when counting a single-phase external signal. Figure 60 shows the timer Ai mode register in event counter mode.

Table 18 lists timer specifications when counting a two-phase external signal. Figure 61 shows the timer Ai mode register in event counter mode.

Table 17. Timer specifications in event counter mode (when not processing two-phase pulse signal)

Item	Specification
Count source	<ul style="list-style-type: none"> <li>External signals input to TAIiN pin (effective edge can be selected by software)</li> <li>TB2 overflow, TAJ overflow</li> </ul>
Count operation	<ul style="list-style-type: none"> <li>Up count or down count can be selected by external signal or software</li> <li>When the timer overflows or underflows, the reload register's content is reloaded and the timer starts over again.(Note)</li> </ul>
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	<ul style="list-style-type: none"> <li>When counting stopped When a value is written to timer Ai register, it is written to both reload register and counter</li> <li>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)</li> </ul>
Select function	<ul style="list-style-type: none"> <li>Free-run count function Even when the timer overflows or underflows, the reload register content is not reloaded to it</li> <li>Pulse output function Each time the timer overflows or underflows, the TAIOUT pin's polarity is reversed</li> </ul>

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

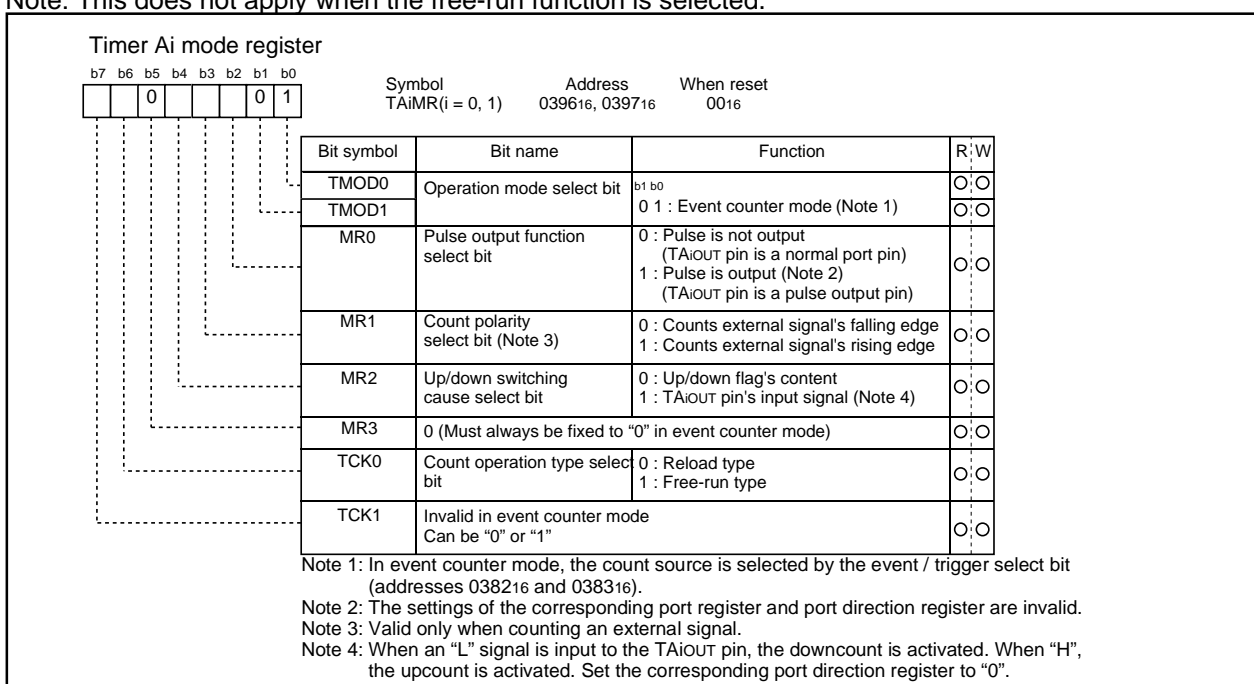


Figure 60. Timer Ai mode register in event counter mode

## TimerA

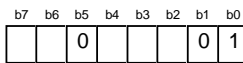
Table 18. Timer specifications in event counter mode (when processing two-phase pulse signal with timer A2,A3 and A4)

Item	Specification
Count source	•Two-phase pulse signals input to TAIIN or TAIOUT pin
Count operation	•Up count or down count can be selected by two-phase pulse signal •When the timer overflows or underflows, the reload register content is reloaded and the timer starts over again (Note)
Divide ratio	1/ (FFFF <sub>16</sub> - n + 1) for up count 1/ (n + 1) for down count                      n : Set value
Count start condition	Count start flag is set (= 1)
Count stop condition	Count start flag is reset (= 0)
Interrupt request generation timing	Timer overflows or underflows
TAiIN pin function	Two-phase pulse input
TAiOUT pin function	Two-phase pulse input
Read from timer	Count value can be read out by reading timer A2, A3, or A4 register
Write to timer	•When counting stopped When a value is written to timer A2, A3, or A4 register, it is written to both reload register and counter •When counting in progress When a value is written to timer A2, A3, or A4 register, it is written to only reload register. (Transferred to counter at next reload time.)
Select function	<p>•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"</p> <p>(i=2,3)    Up count    Up count    Up count    Down count    Down count    Down count</p> <p>•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.</p> <p>TAiOUT    Count up all edges    Count down all edges</p> <p>TAiIN (i=3,4)    Count up all edges    Count down all edges</p>

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

## TimerA

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



Symbol Address When reset  
TAiMR(i = 2 to 4) 0398<sub>16</sub> to 039A<sub>16</sub> 00<sub>16</sub>

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

Note 1: The settings of the corresponding port register and port direction register are invalid

Note 2: This bit is valid when only counting an external signal.

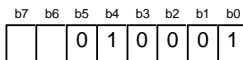
Note 3: Set the corresponding port direction register to "0".

Note 4: This bit is valid for timer A3 mode register.

For timer A2 and A4 mode registers, this bit can be "0" or "1".

Note 5: When performing two-phase pulse signal processing, make sure the two-phase pulse signal processing operation select bit (address 0384<sub>16</sub>) is set to "1". Also, always be sure to set the event/trigger select bit (addresses 0382<sub>16</sub> and 0383<sub>16</sub>) to "00".

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



Symbol Address When reset  
TAiMR(i = 2 to 4) 0398<sub>16</sub> to 039A<sub>16</sub> 00<sub>16</sub>

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

Note 1: This bit is valid for timer A3 mode register.

For timer A2 and A4 mode registers, this bit can be "0" or "1".

Note 2: When performing two-phase pulse signal processing, make sure the two-phase pulse signal processing operation select bit (address 0384<sub>16</sub>) is set to "1". Also, always be sure to set the event/trigger select bit (addresses 0382<sub>16</sub> and 0383<sub>16</sub>) to "00".

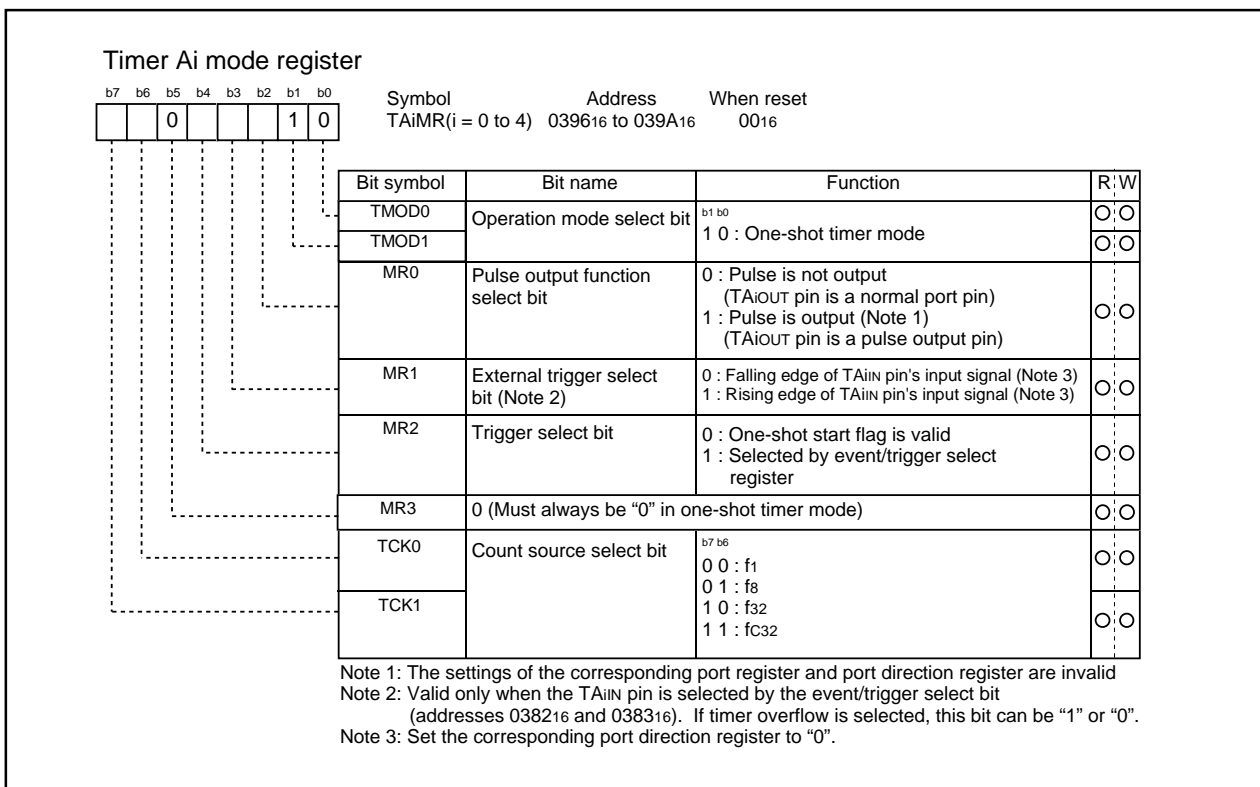
Figure 61. Timer Ai mode register in event counter m

### (3) One-shot timer mode

In this mode, the timer operates only once. (See Table 19.) When a trigger occurs, the timer starts up and continues operating for a given period. Figure 62 shows the timer Ai mode register in one-shot timer mode.

**Table 19. Timer specifications in one-shot timer mode**

Item	Specification
Count source	f1, f8, f32, fc32
Count operation	<ul style="list-style-type: none"> <li>The timer counts down</li> <li>When the count reaches 0000<sub>16</sub>, the timer stops counting after reloading a new count</li> <li>If a trigger occurs when counting, the timer reloads a new count and restarts counting</li> </ul>
Divide ratio	1/n    n : Set value
Count start condition	<ul style="list-style-type: none"> <li>An external trigger is input</li> <li>The timer overflows</li> <li>The one-shot start flag is set (= 1)</li> </ul>
Count stop condition	<ul style="list-style-type: none"> <li>A new count is reloaded after the count has reached 0000<sub>16</sub></li> <li>The count start flag is reset (= 0)</li> </ul>
Interrupt request generation timing	The count reaches 0000 <sub>16</sub>
TAiIN pin function	Programmable I/O port or trigger input
TAiOUT pin function	Programmable I/O port or pulse output
Read from timer	When timer Ai register is read, it indicates an indeterminate value
Write to timer	<ul style="list-style-type: none"> <li>When counting stopped When a value is written to timer Ai register, it is written to both reload register and counter</li> <li>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)</li> </ul>



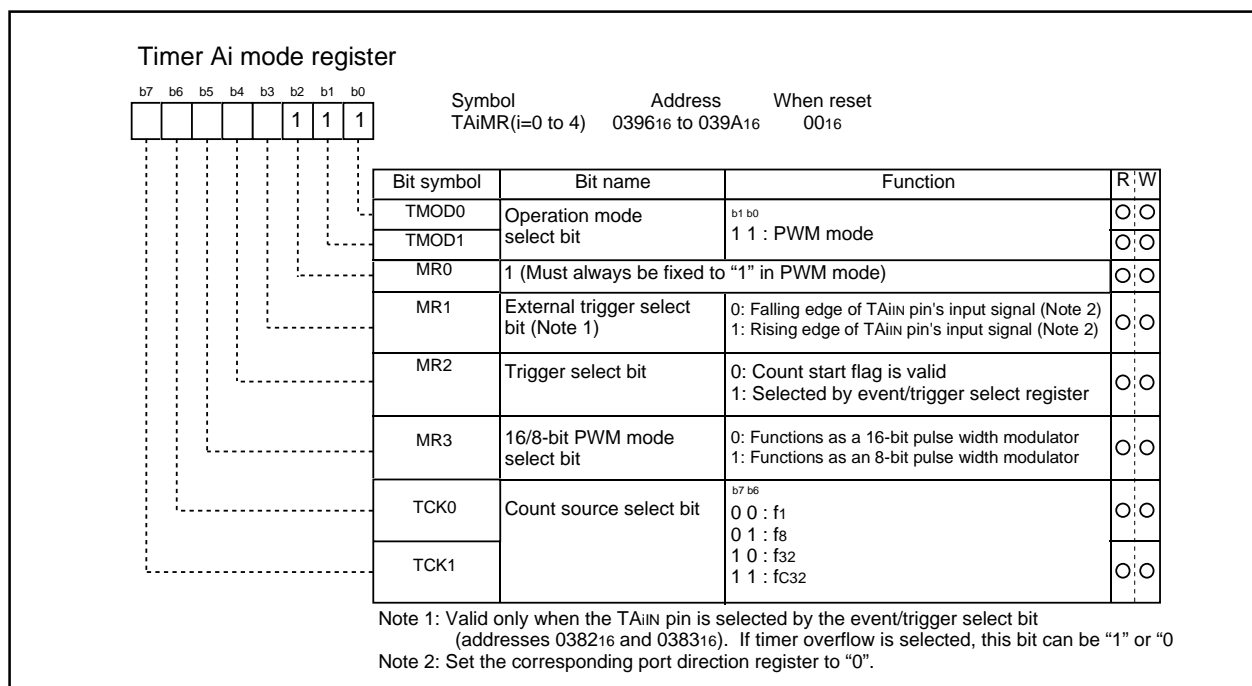
**Figure 62. Timer Ai mode register in one-shot timer mode**

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

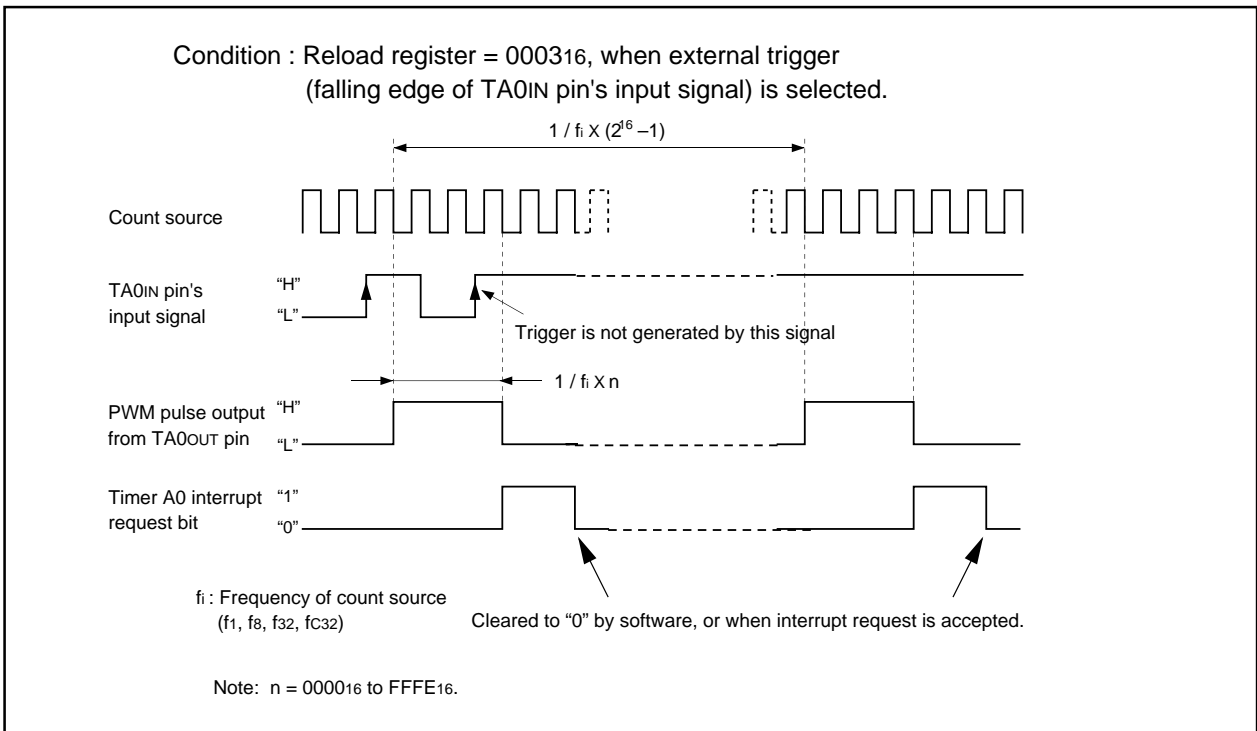
In this mode, the timer outputs pulses of a given width in succession. (See Table 20.) In this mode, the counter functions as either a 16-bit pulse width modulator or an 8-bit pulse width modulator. Figure 63 shows the timer Ai mode register in pulse width modulation mode. Figure 64 shows the example of how a 16-bit pulse width modulator operates. Figure 65 shows the example of how an 8-bit pulse width modulator operates.

**Table 20. Timer specifications in pulse width modulation mode**

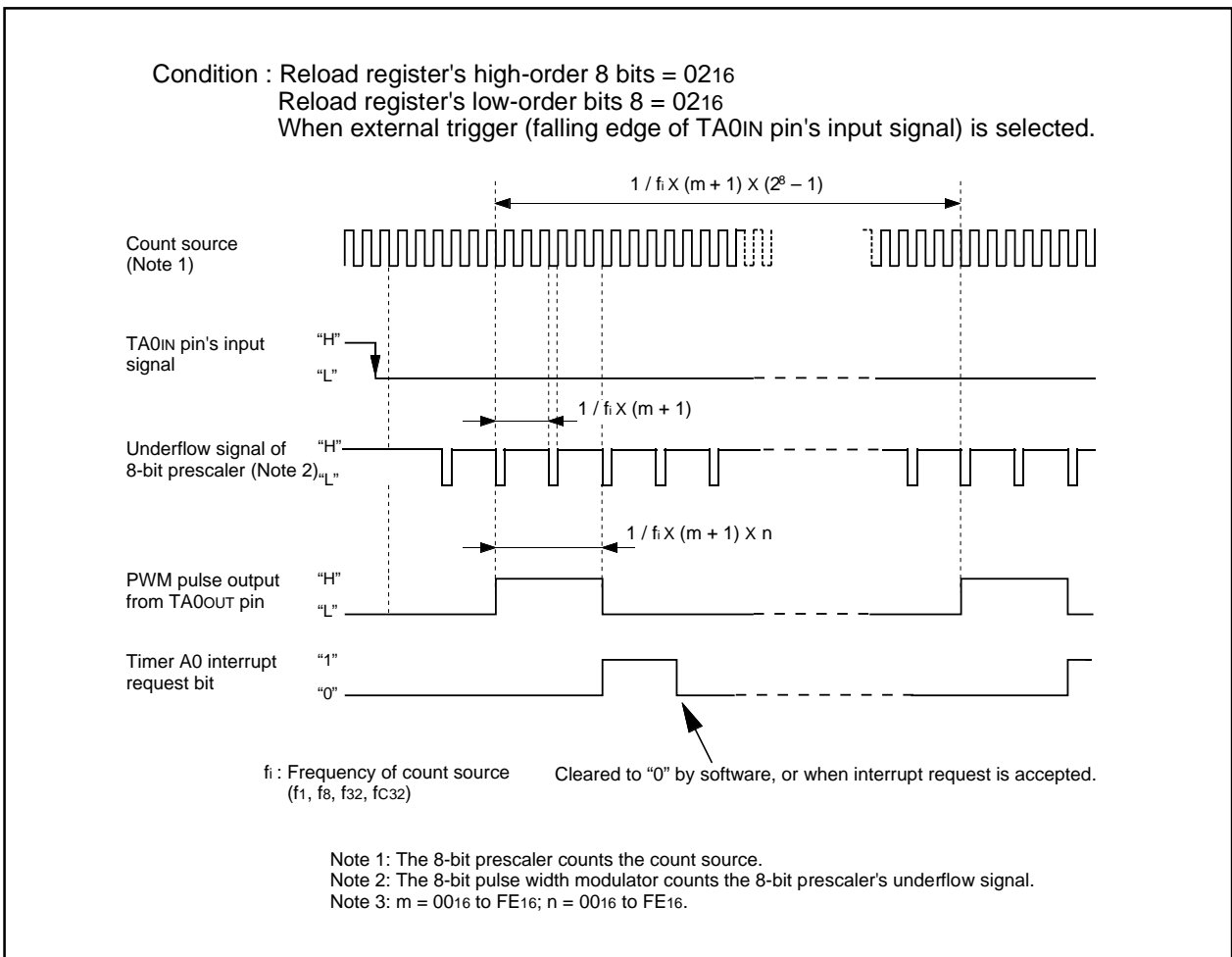
Item	Specification
Count source	f1, f8, f32, fc32
Count operation	<ul style="list-style-type: none"> <li>•The timer counts down (operating as an 8-bit or a 16-bit pulse width modulator)</li> <li>•The timer reloads a new count at a rising edge of PWM pulse and continues counting</li> <li>•The timer is not affected by a trigger that occurs when counting</li> </ul>
16-bit PWM	<ul style="list-style-type: none"> <li>•High level width <math>n / f_i</math> n : Set value</li> <li>•Cycle time <math>(2^{16}-1) / f_i</math> fixed</li> </ul>
8-bit PWM	<ul style="list-style-type: none"> <li>•High level width <math>n \times (m+1) / f_i</math> n : values set to timer Ai register's high-order address</li> <li>•Cycle time <math>(2^8-1) \times (m+1) / f_i</math> m : values set to timer Ai register's low-order address</li> </ul>
Count start condition	<ul style="list-style-type: none"> <li>•External trigger is input</li> <li>•The timer overflows</li> <li>•The count start flag is set (= 1)</li> </ul>
Count stop condition	•The count start flag is reset (= 0)
Interrupt request generation timing	PWM pulse goes "L"
TAiIN pin function	Programmable I/O port or trigger input
TAiOUT pin function	Pulse output
Read from timer	When timer Ai register is read, it indicates an indeterminate value
Write to timer	<ul style="list-style-type: none"> <li>•When counting stopped When a value is written to timer Ai register, it is written to both reload register and counter</li> <li>•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)</li> </ul>



**Figure 63. Timer Ai mode register in pulse width modulation mode**



**Figure 64. Example of how a 16-bit pulse width modulator operates**



**Figure 65. Example of how an 8-bit pulse width modulator operates**



## TimerB

## Timer B

Figure 66 shows the block diagram of timer B. Figures 67 and 68 show the timer B-related registers.

Use the timer Bi mode register ( $i = 0$  to 2) bits 0 and 1 to choose the desired mode.

Timer B has three operation modes listed as follows:

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

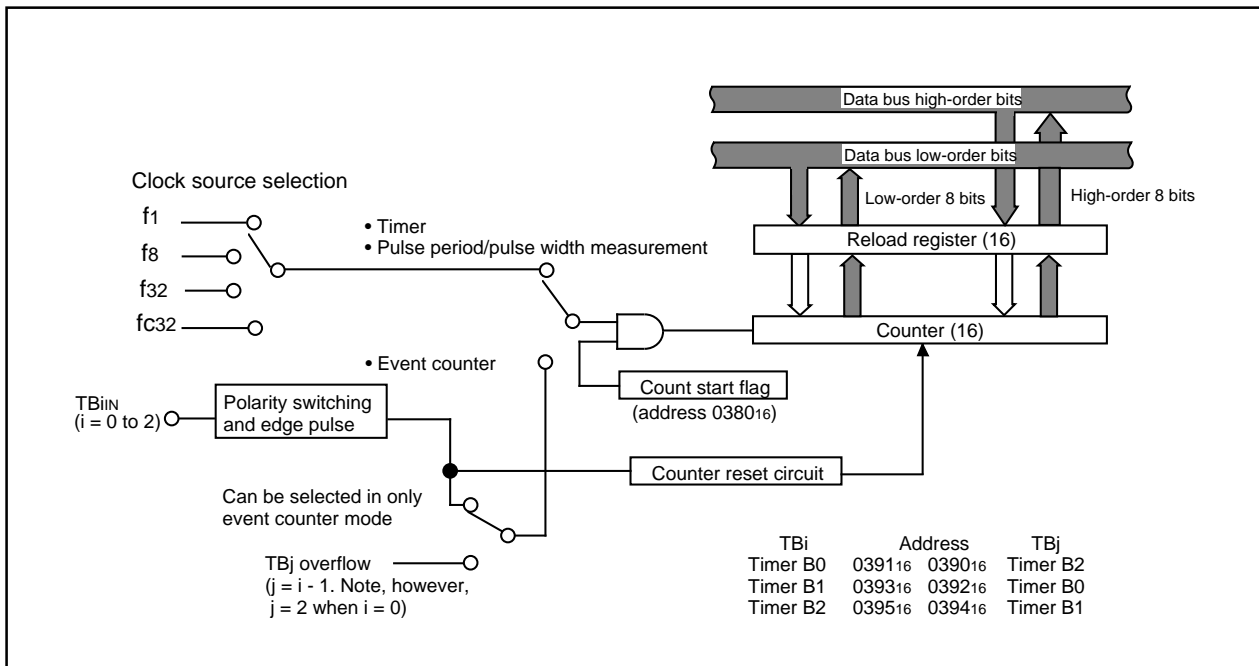


Figure 66. Block diagram of timer B

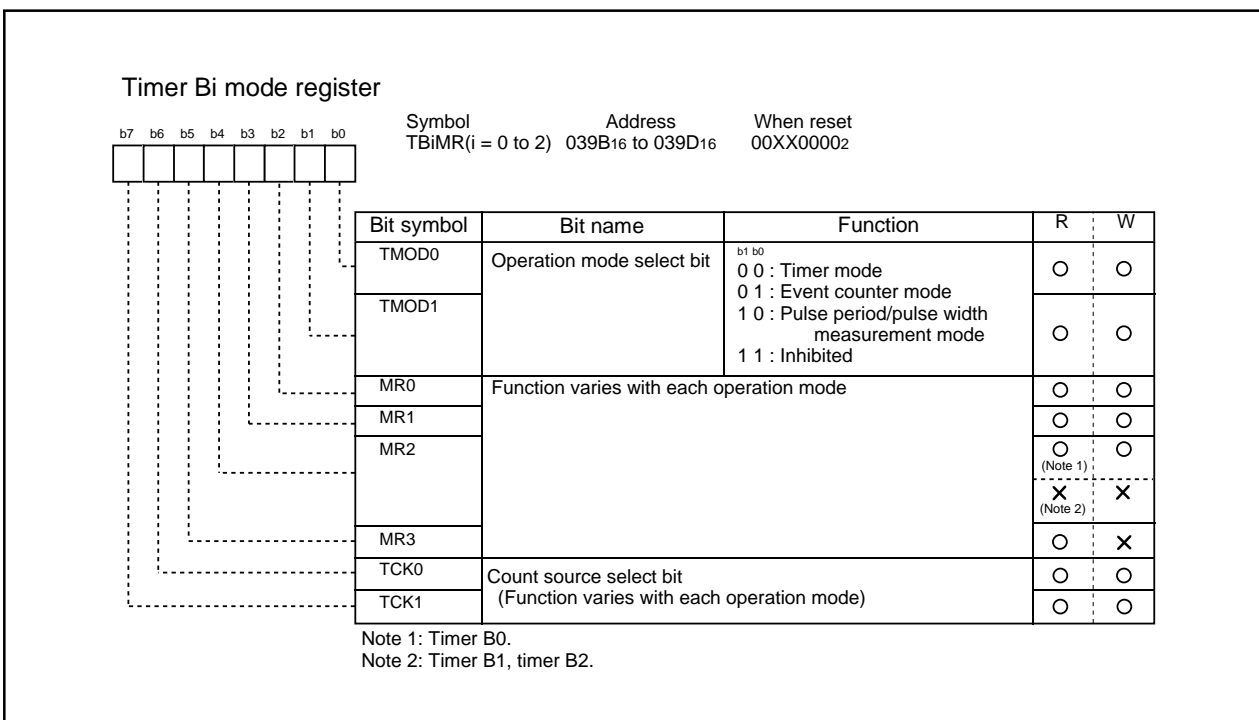


Figure 67. Timer B-related registers (1)

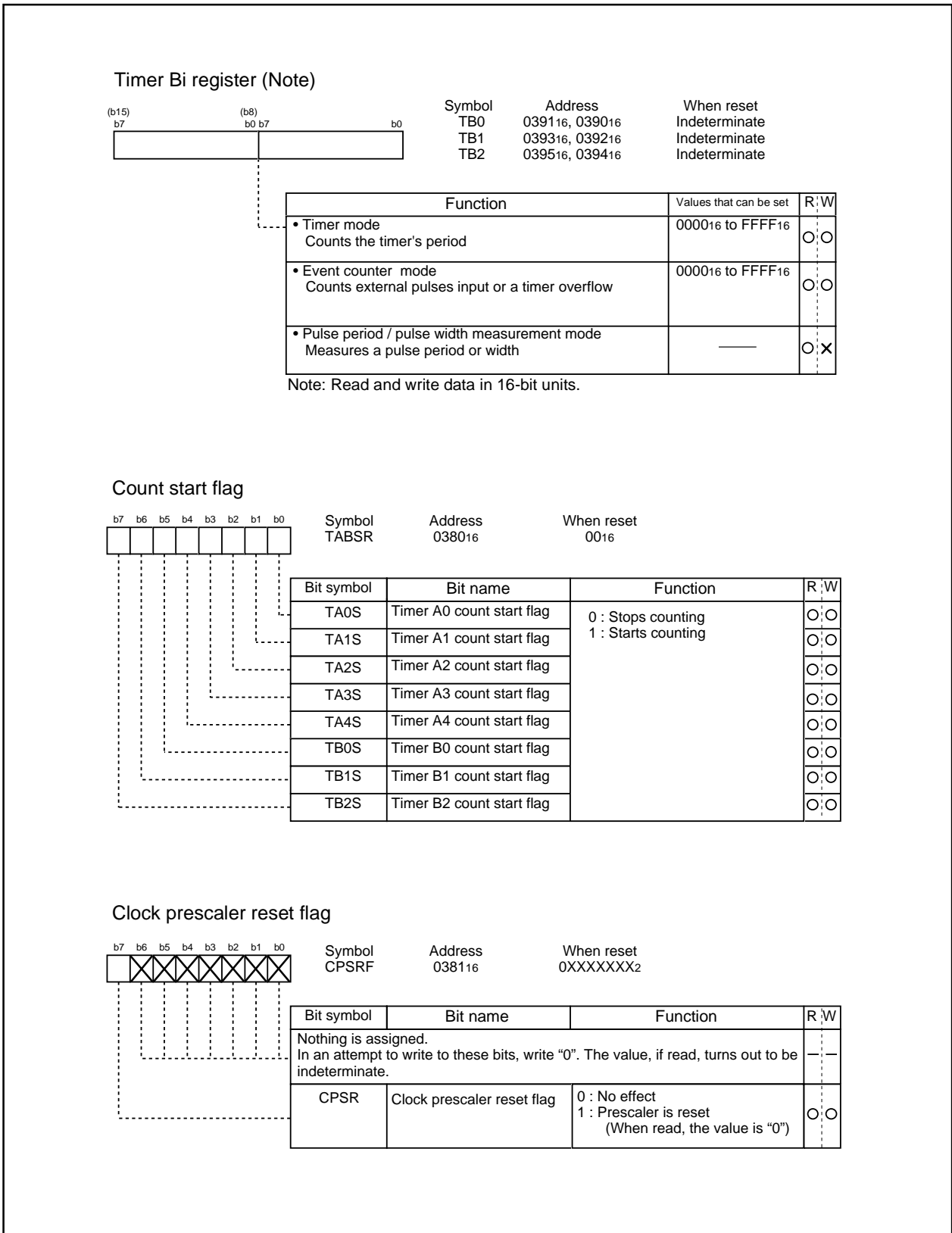


Figure 68. Timer B-related registers (2)

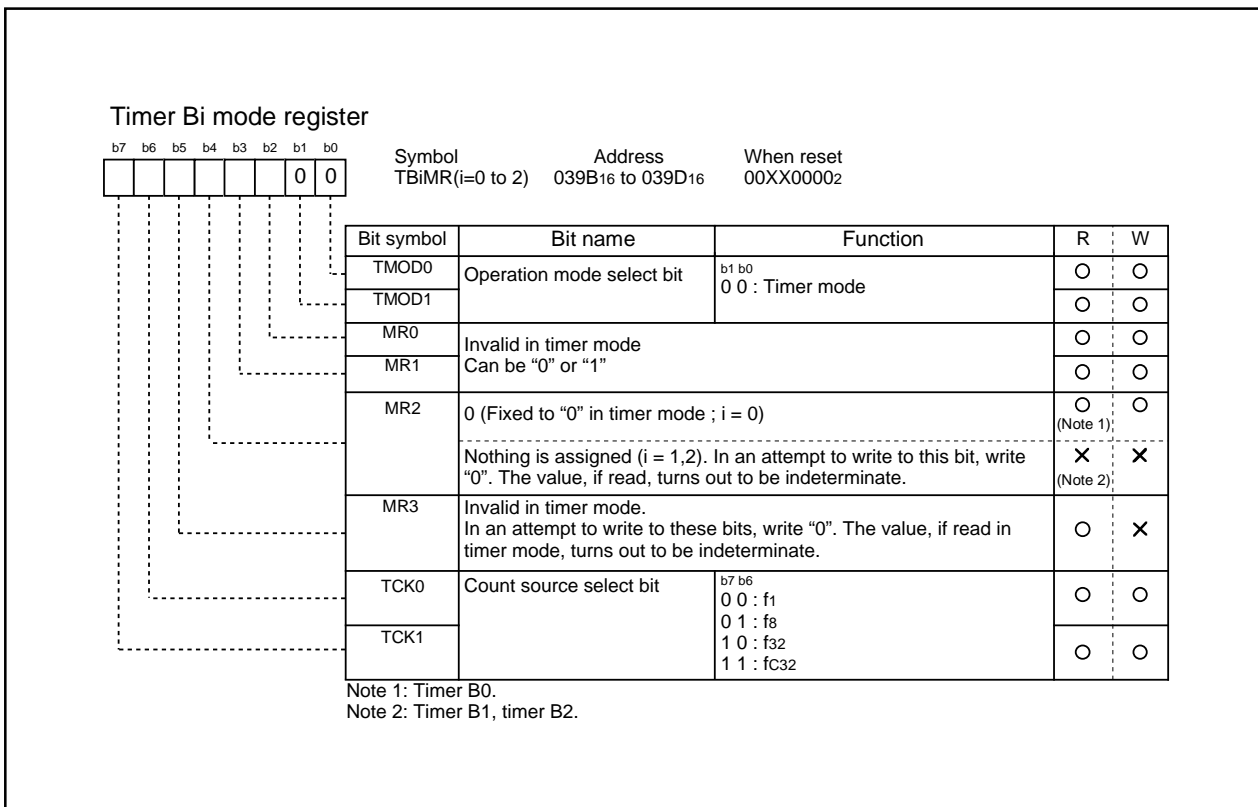
## TimerB

**(1) Timer mode**

In this mode, the timer counts an internally generated count source. (See Table 21.) Figure 69 shows the timer Bi mode register in timer mode.

**Table 21. Timer specifications in timer mode**

Item	Specification
Count source	f1, f8, f32, fc32
Count operation	<ul style="list-style-type: none"> <li>Counts down</li> <li>When the timer underflows, the reload register's content is reloaded and the timer starts over again.</li> </ul>
Divide ratio	1/(n+1) n : Set value
Count start condition	Count start flag is set (= 1)
Count stop condition	Count start flag is reset (= 0)
Interrupt request generation timing	The timer underflows
TBiIN pin function	Programmable I/O port
Read from timer	Count value is read out by reading timer Bi register
Write to timer	<ul style="list-style-type: none"> <li>When counting stopped When a value is written to timer Bi register, it is written to both reload register and counter</li> <li>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)</li> </ul>

**Figure 69. Timer Bi mode register in timer mode**

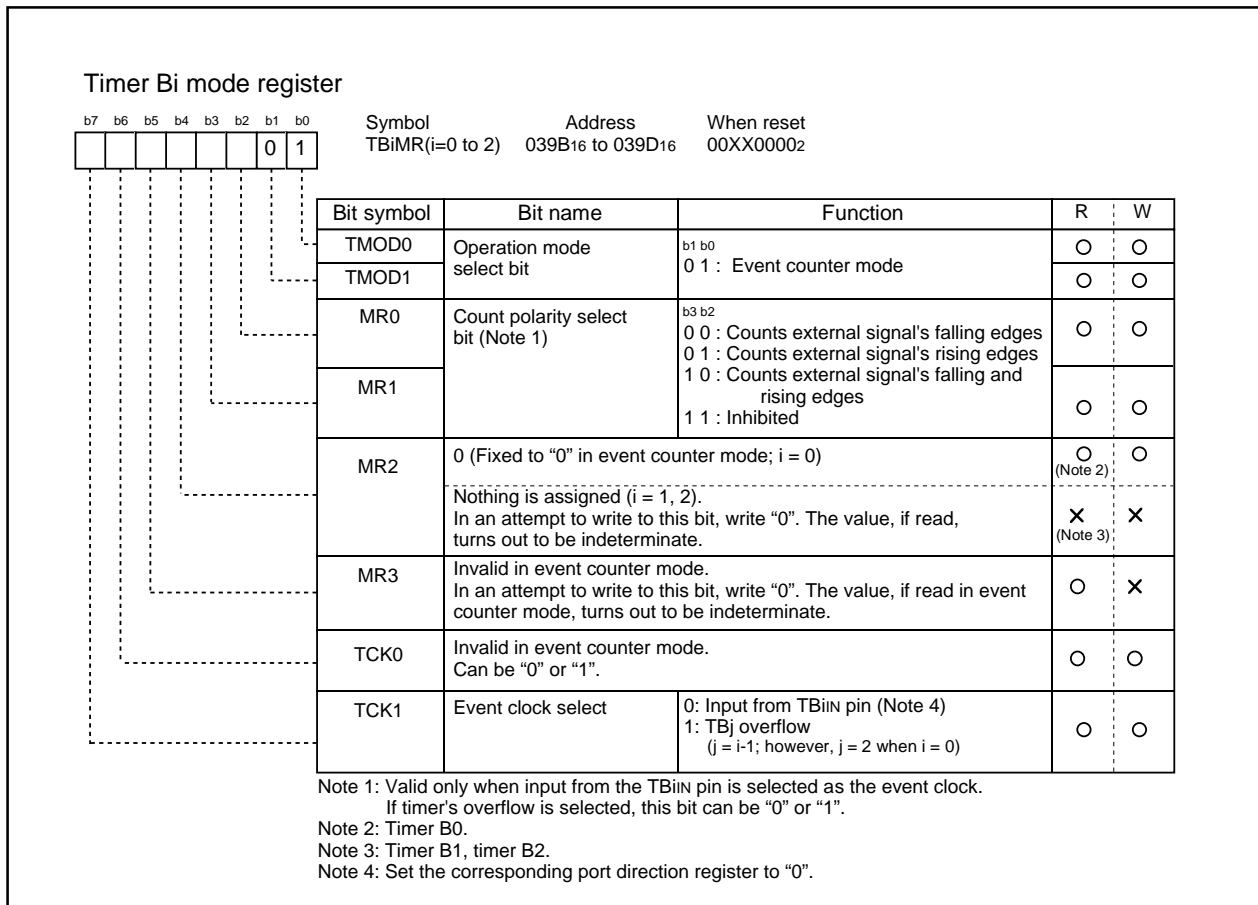
## TimerB

**(2) Event counter mode**

In this mode, the timer counts an external signal or an internal timer's overflow. (See Table 22.) Figure 70 shows the timer Bi mode register in event counter mode.

**Table 22. Timer specifications in event counter mode**

Item	Specification
Count source	<ul style="list-style-type: none"> <li>External signals input to TBIIN pin</li> <li>Effective edge of count source can be a rising edge, a falling edge, or falling and rising edges as selected by software</li> </ul>
Count operation	<ul style="list-style-type: none"> <li>Counts down</li> <li>When the timer underflows, it reloads the reload register contents before continuing counting</li> </ul>
Divide ratio	$1/(n+1)$ $n$ : Set value
Count start condition	Count start flag is set (= 1)
Count stop condition	Count start flag is reset (= 0)
Interrupt request generation timing	The timer underflows
TBIIN pin function	Count source input
Read from timer	Count value can be read out by reading timer Bi register
Write to timer	<ul style="list-style-type: none"> <li>When counting stopped When a value is written to timer Bi register, it is written to both reload register and counter</li> <li>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)</li> </ul>

**Figure 70. Timer Bi mode register in event counter mode**

## TimerB

**(3) Pulse period/pulse width measurement mode**

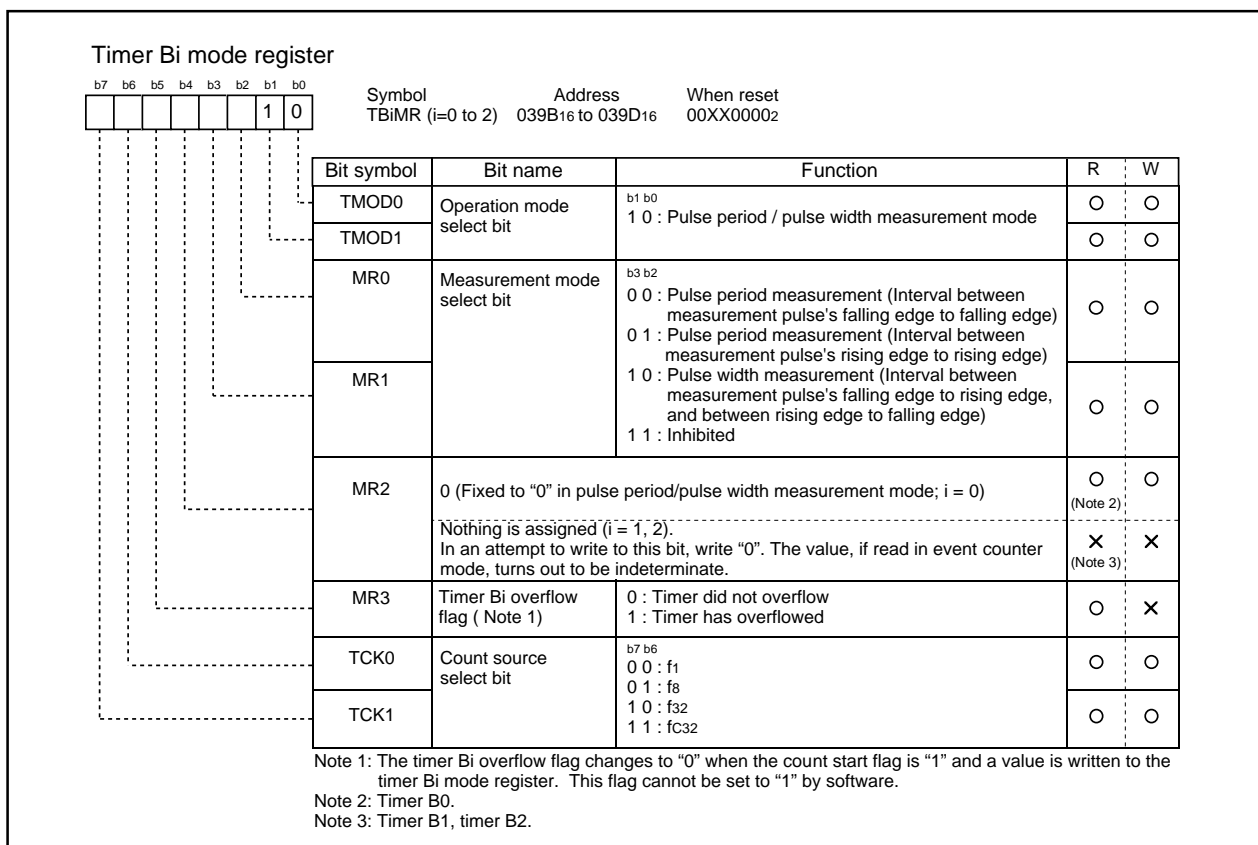
In this mode, the timer measures the pulse period or pulse width of an external signal. (See Table 23.) Figure 71 shows the timer Bi mode register in pulse period/pulse width measurement mode. Figure 72 shows the operation timing when measuring a pulse period. Figure 73 shows the operation timing when measuring a pulse width.

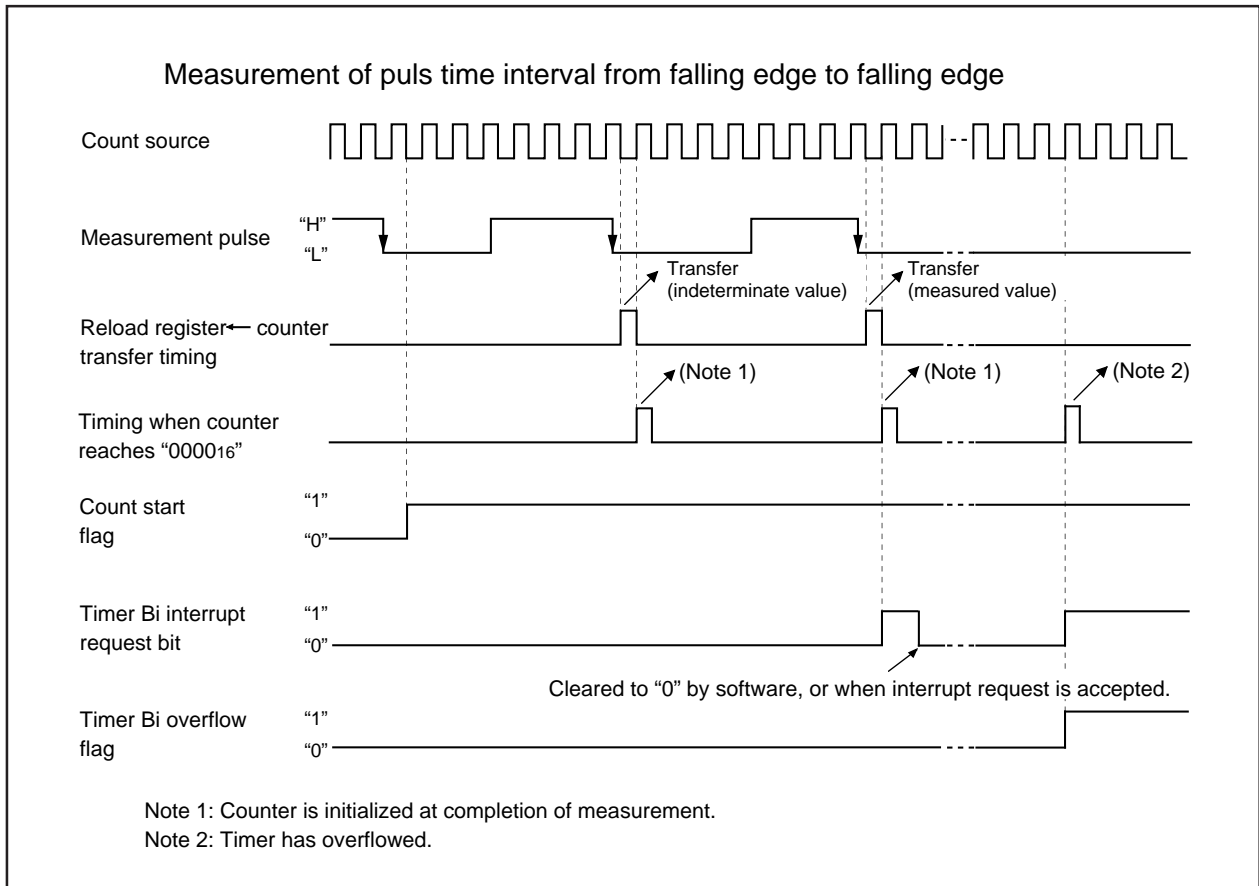
**Table 23. Timer specifications in pulse period/pulse width measurement mode**

Item	Specification
Count source	f1, f8, f32, fc32
Count operation	<ul style="list-style-type: none"> <li>•Up count</li> <li>•Counter value "0000<sub>16</sub>" is transferred to reload register at measurement pulse's effective edge and the timer continues counting</li> </ul>
Count start condition	Count start flag is set (= 1)
Count stop condition	Count start flag is reset (= 0)
Interrupt request generation timing	<ul style="list-style-type: none"> <li>•When measurement pulse's effective edge is input (Note 1)</li> <li>•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.)</li> </ul>
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

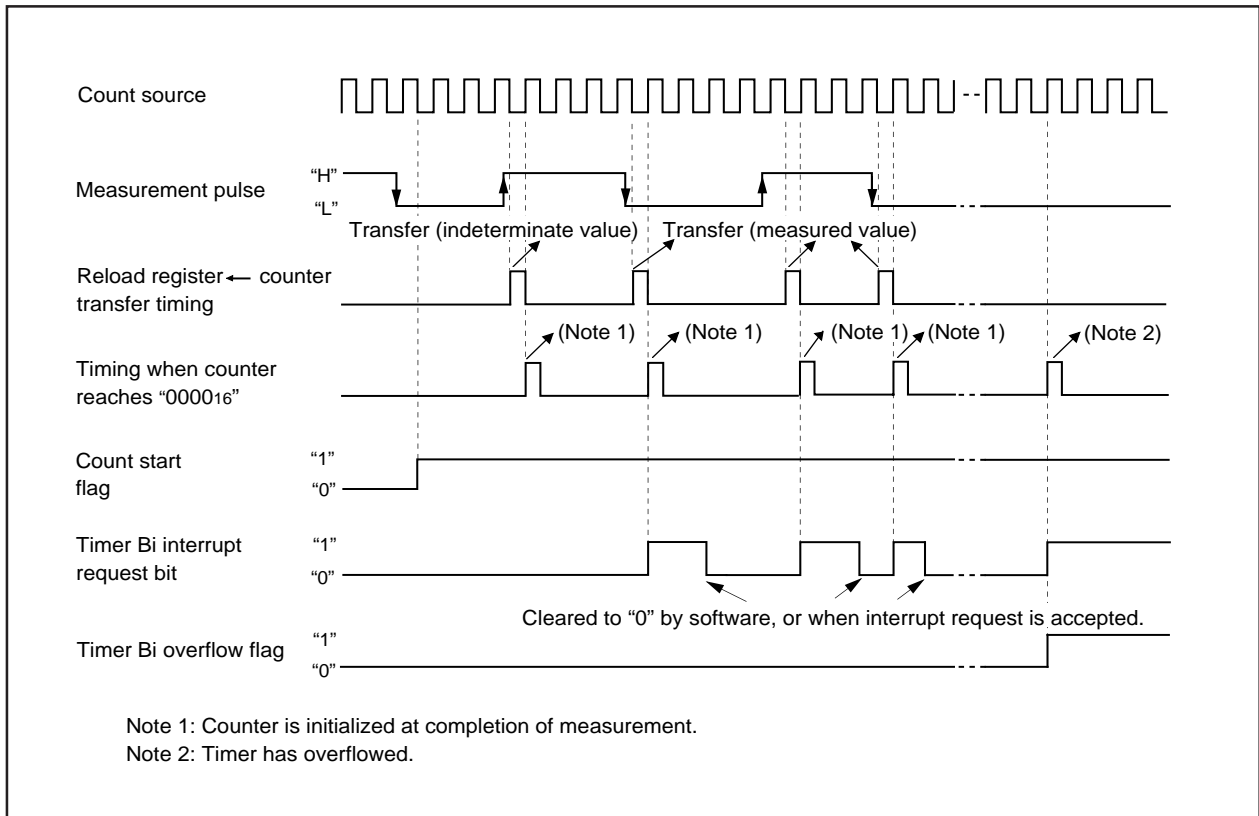
Note 1: An interrupt request is not generated when the first effective edge is input after the timer has started counting.

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

**Figure 71. Timer Bi mode register in pulse period/pulse width measurement mode**



**Figure 72. Operation timing when measuring a pulse period**



**Figure 73. Operation timing when measuring a pulse width**

## Serial I/O

## Serial I/O

Serial I/O is configured as two channels: UART0 and UART1.

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

Figure 74 shows the block diagram of UART0 and UART1. Figure 75 shows the block diagram of the transmit/receive unit.

UART<sub>i</sub> (i=0, 1) 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<sub>16</sub> and 03A8<sub>16</sub>) determine whether UART<sub>i</sub> is used as a clock synchronous serial I/O or as a UART.

Although a few functions are different, UART0 and UART1 have almost same functions.

Figures 76 through 78 show the registers related to UART<sub>i</sub>.

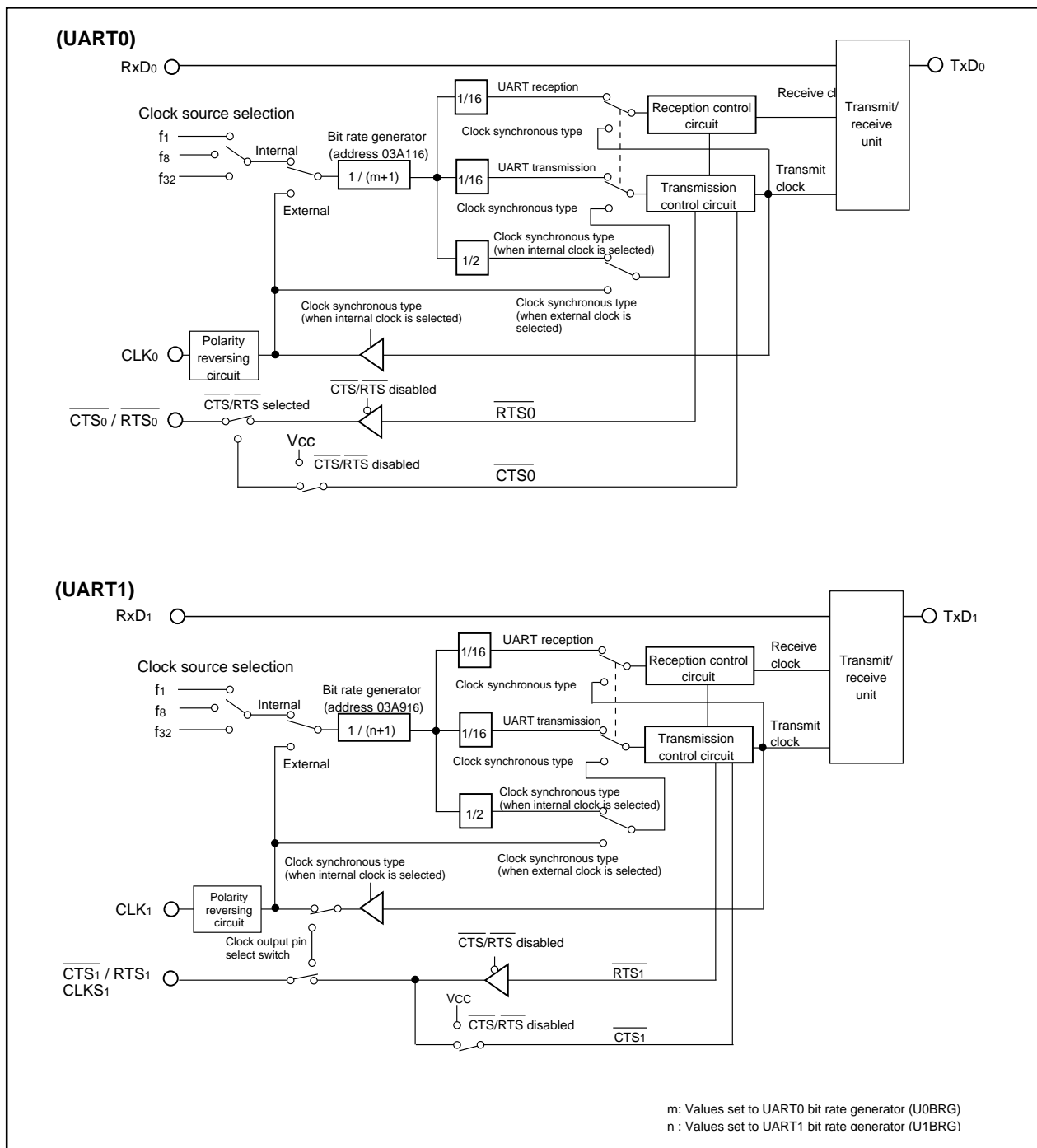


Figure 74. Block diagram of UART<sub>i</sub> (i = 0, 1)

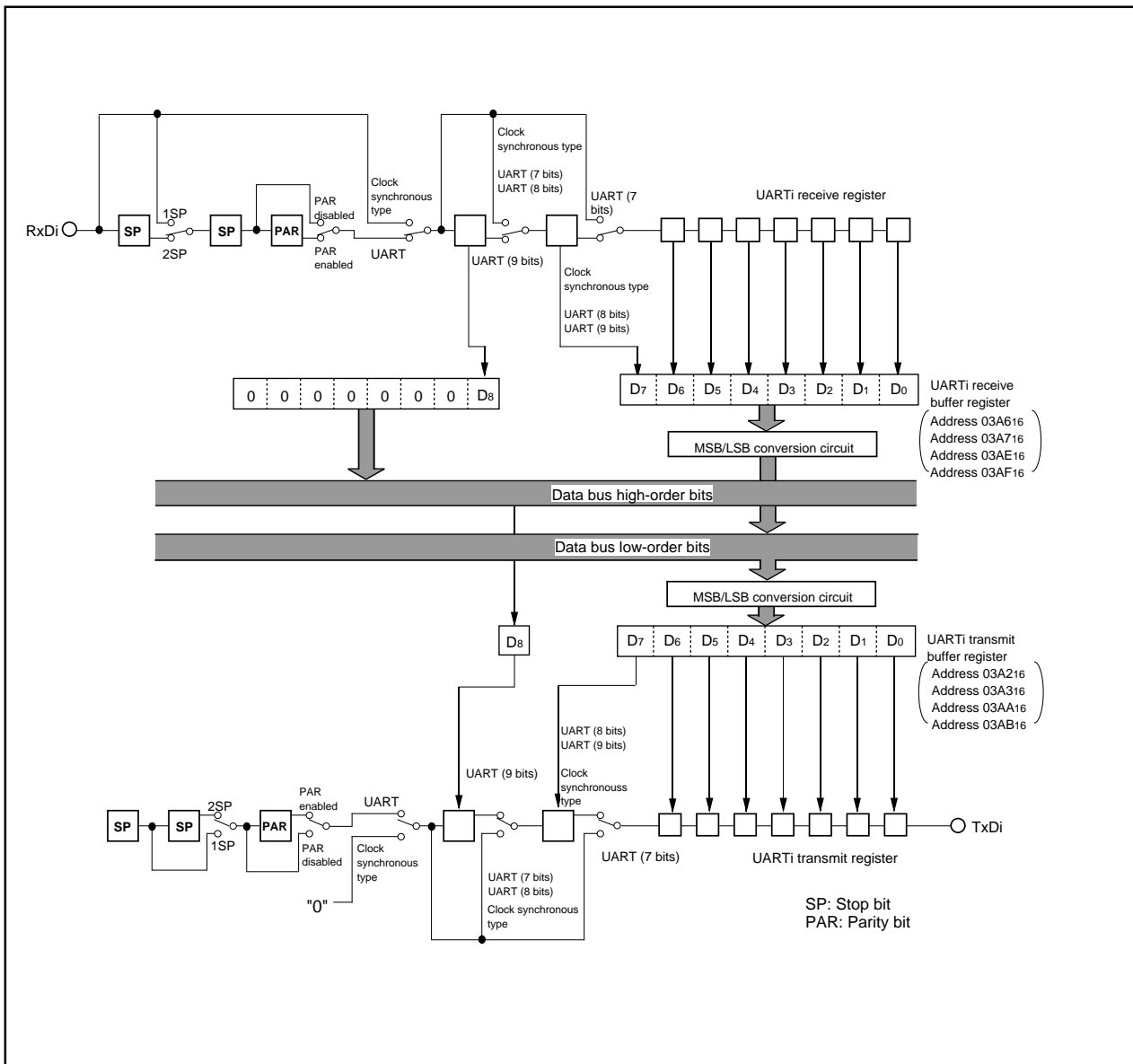


Figure 75. Block diagram of transmit/receive unit



Serial I/O

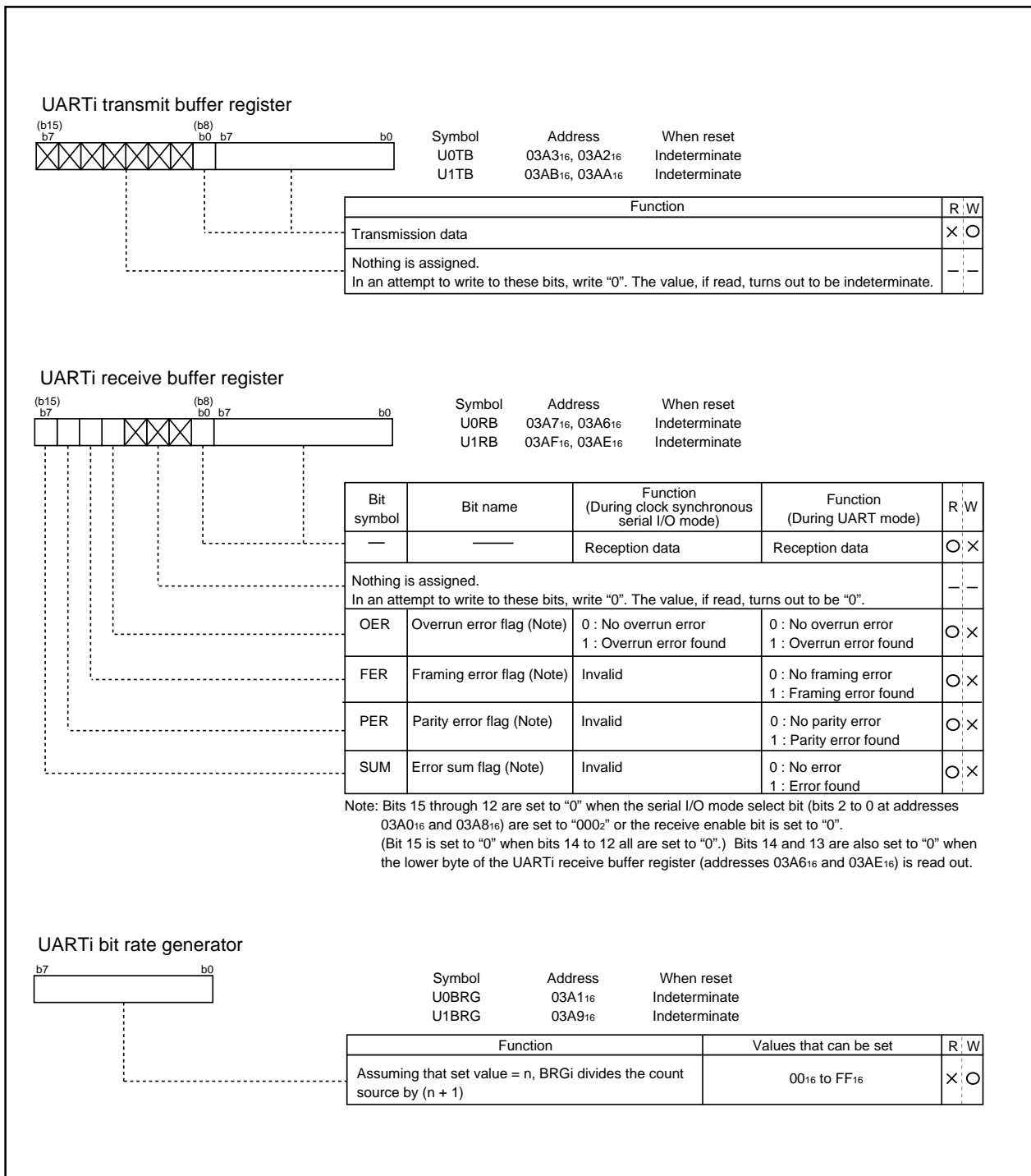


Figure 76. Serial I/O-related registers (1)

## Serial I/O

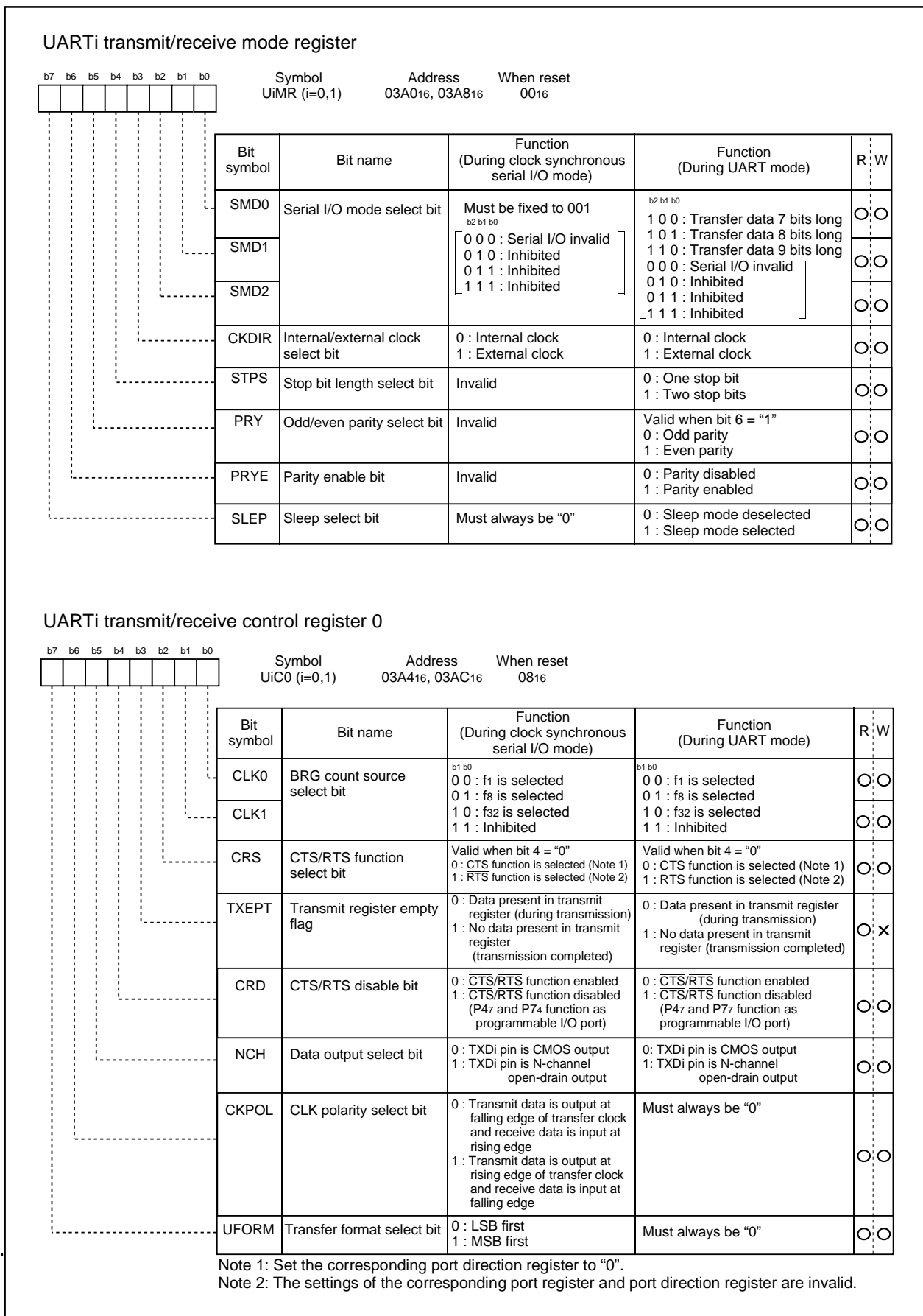


Figure 77. Serial I/O-related registers (2)

## Serial I/O

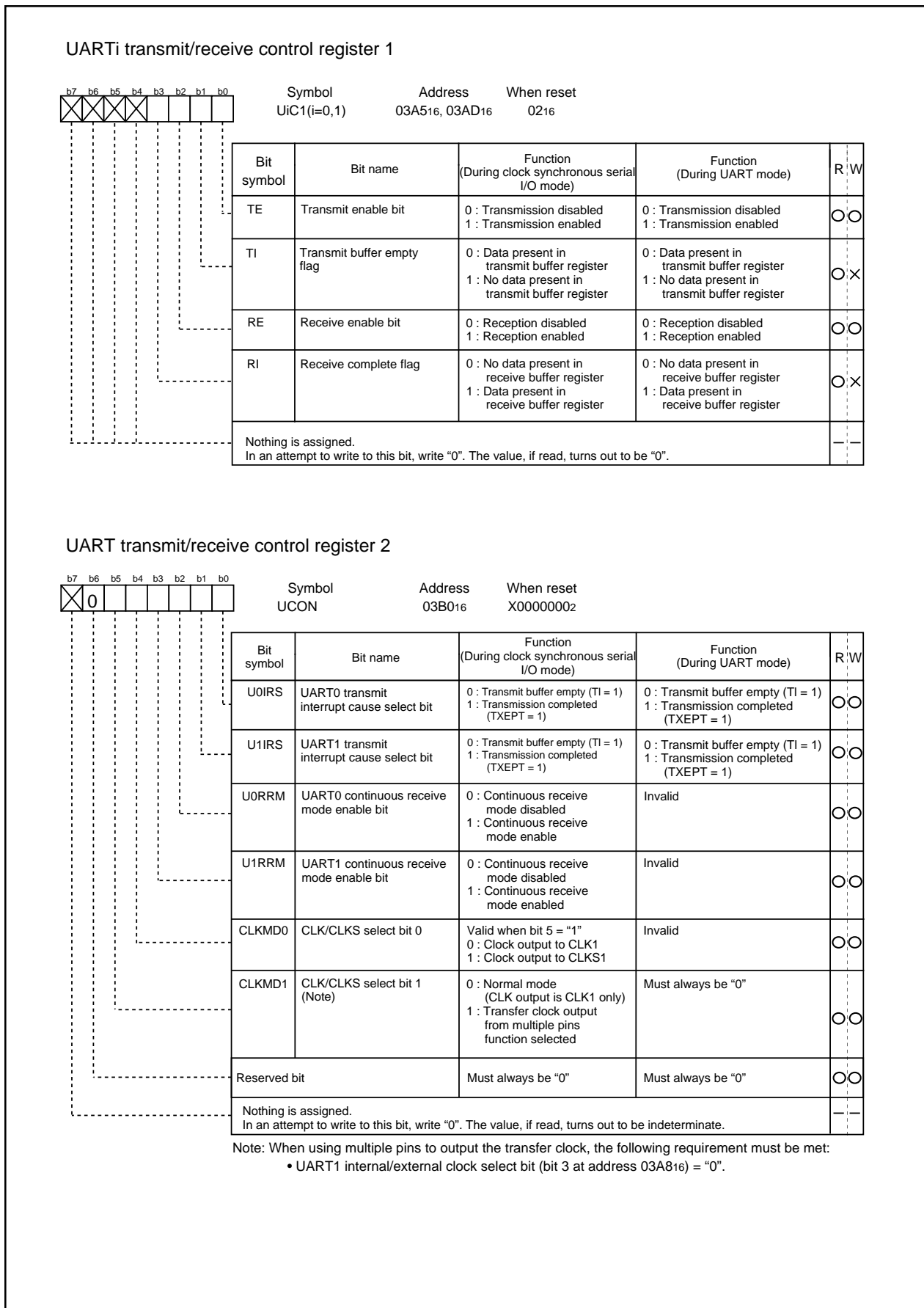


Figure 78. Serial I/O-related registers (3)

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

The clock synchronous serial I/O mode uses a transfer clock to transmit and receive data. Table 24 lists the specifications of the clock synchronous serial I/O mode. Figure 79 shows the UART<sub>i</sub> transmit/receive mode register.

**Table 24. Specifications of clock synchronous serial I/O mode**

Item	Specification
Transfer data format	• Transfer data length: 8 bits
Transfer clock	• When internal clock is selected (bit 3 at address 03A0 <sub>16</sub> , 03A8 <sub>16</sub> = "0") : $f_i/2(n+1)$ (Note 1) $f_i = f_1, f_8, f_{32}$ • When external clock is selected (bit 3 at address 03A0 <sub>16</sub> , 03A8 <sub>16</sub> = "1") : Input from CLK <sub>i</sub> pin (Note 2)
Transmission/reception control	• CTS function/ RTS function/ CTS,RTS function chosen to be invalid
Transmission start condition	• To start transmission, the following requirements must be met: – Transmit enable bit (bit 0 at address 03A5 <sub>16</sub> , 03AD <sub>16</sub> ) = "1" – Transmit buffer empty flag (bit 1 at addresses 03A5 <sub>16</sub> , 03AD <sub>16</sub> ) = "0" – When CTS function is selected, CTS input level = "L" • Furthermore, if external clock is selected, the following requirements must also be met: – CLK <sub>i</sub> polarity select bit (bit 6 at address 03A4 <sub>16</sub> , 03AC <sub>16</sub> ) = "0": CLK <sub>i</sub> input level = "H" – CLK <sub>i</sub> polarity select bit (bit 6 at address 03A4 <sub>16</sub> , 03AC <sub>16</sub> ) = "1": CLK <sub>i</sub> input level = "L"
Reception start condition	• To start reception, the following requirements must be met: – Receive enable bit (bit 2 at address 03A5 <sub>16</sub> , 03AD <sub>16</sub> ) = "1" – Transmit enable bit (bit 0 at address 03A5 <sub>16</sub> , 03AD <sub>16</sub> ) = "1" – Transmit buffer empty flag (bit 1 at address 03A5 <sub>16</sub> , 03AD <sub>16</sub> ) = "0" • Furthermore, if external clock is selected, the following requirements must also be met: – CLK <sub>i</sub> polarity select bit (bit 6 at address 03A4 <sub>16</sub> , 03AC <sub>16</sub> ) = "0": CLK <sub>i</sub> input level = "H" – CLK <sub>i</sub> polarity select bit (bit 6 at address 03A4 <sub>16</sub> , 03AC <sub>16</sub> ) = "1": CLK <sub>i</sub> input level = "L"
Interrupt request generation timing	• When transmitting – Transmit interrupt cause select bit (bits 0,1 at address 03B0 <sub>16</sub> ) = "0": Interrupts requested when data transfer from UART <sub>i</sub> transfer buffer register to UART <sub>i</sub> transmit register is completed – Transmit interrupt cause select bit (bits 0,1 at address 03B0 <sub>16</sub> ) = "1": Interrupts requested when data transmission from UART <sub>i</sub> transfer register is completed • When receiving – Interrupts requested when data transfer from UART <sub>i</sub> receive register to UART <sub>i</sub> receive buffer register is completed
Error detection	• Overrun error (Note 3) This error occurs when the next data is ready before contents of UART <sub>i</sub> receive buffer register are read out
Select function	• CLK polarity selection Whether transmit data is output/input at the rising edge or falling edge of the transfer clock can be selected • LSB first/MSB first selection Whether transmission/reception begins with bit 0 or bit 7 can be selected • Continuous receive mode selection Reception is enabled simultaneously by a read from the receive buffer register • Transfer clock output from multiple pins selection UART <sub>1</sub> transfer clock can be set 2 pins, and can be selected to output from which pin.

Note 1: "n" denotes the value 00<sub>16</sub> to FF<sub>16</sub> that is set to the UART bit rate generator.

Note 2: Maximum 5 Mbps.

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

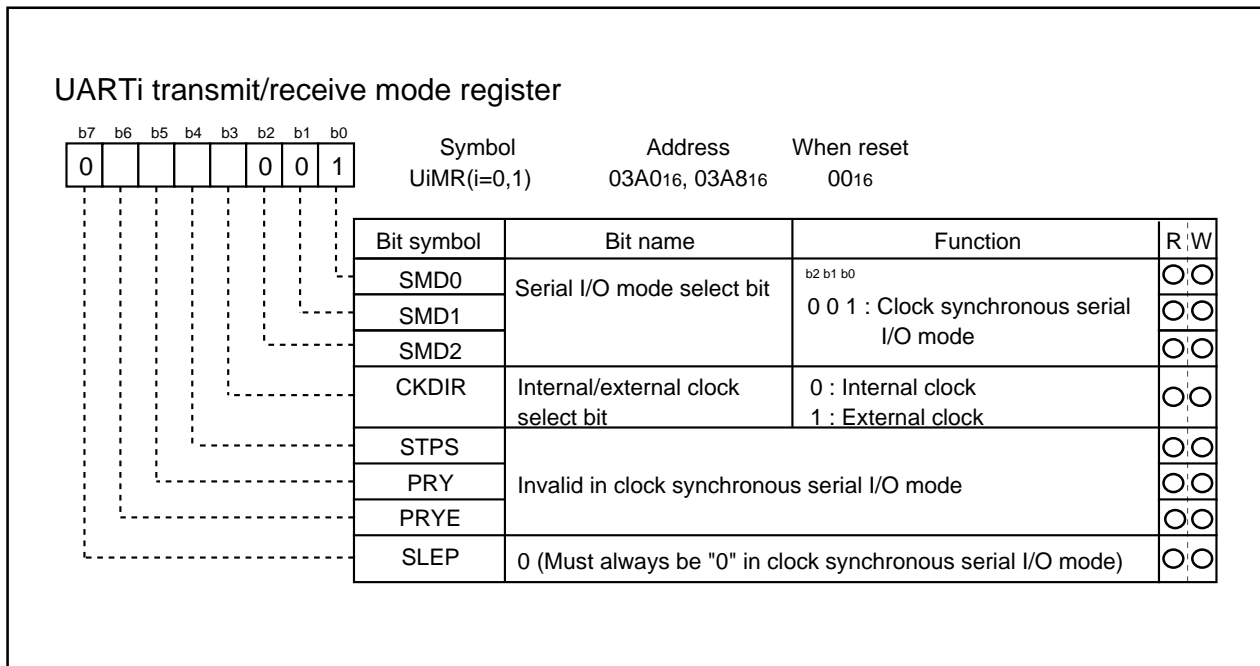


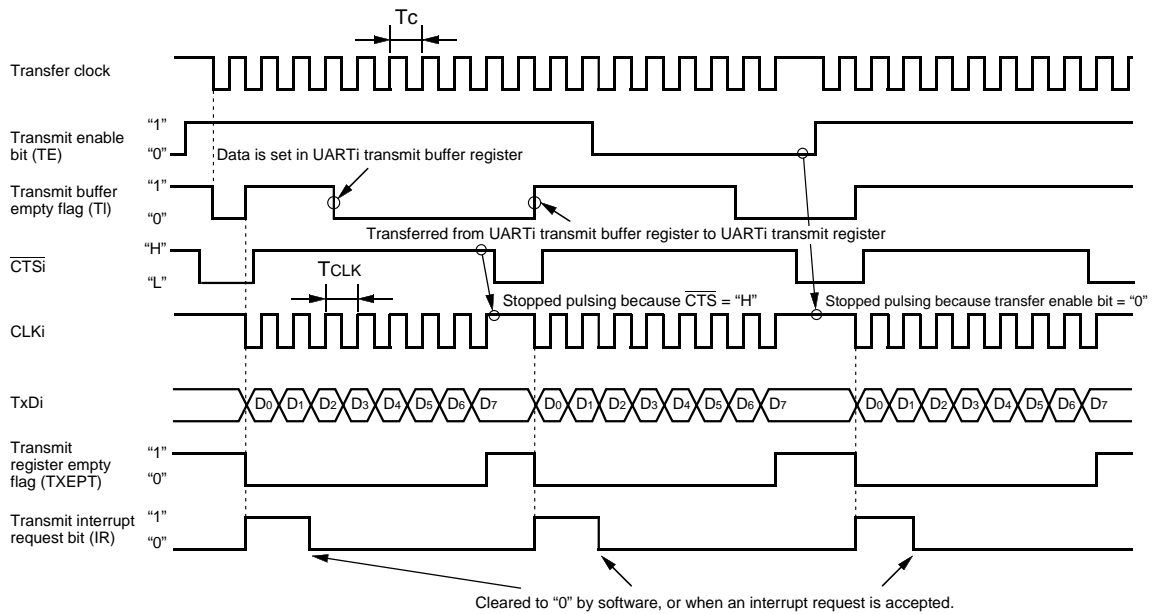
Figure 79. UART<sub>i</sub> transmit/receive mode register in clock synchronous serial I/O mode (*i*=0,1)

Table 25 lists the functions of the input/output pins during clock synchronous serial I/O mode. Note that for a period from when the UART<sub>i</sub> 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 25. Input/output pin functions in clock synchronous serial I/O mode (*i*=0,1)

Pin name	Function	Method of selection
TxD <sub>i</sub> (P44, P74)	Serial data output	(Outputs dummy data when performing reception only)
RxD <sub>i</sub> (P45, P75)	Serial data input	Port P45, P75 direction register (bits 5 at address 03EA <sub>16</sub> and 03EF <sub>16</sub> ) = "0" (Can be used as an input port when performing transmission only)
CLK <sub>i</sub> (P46, P76)	Transfer clock output	Internal/external clock select bit (bit 3 at address 03A0 <sub>16</sub> , 03A8 <sub>16</sub> ) = "0"
	Transfer clock input	Internal/external clock select bit (bit 3 at address 03A0 <sub>16</sub> , 03A8 <sub>16</sub> ) = "1" Port P46, P76 direction register (bits 6 at address 03EA <sub>16</sub> and 03EF <sub>16</sub> ) = "0"
CTS <sub>i</sub> /RTS <sub>i</sub> (P47, P77)	CTS input	CTS/RTS disable bit (bit 4 at address 03A4 <sub>16</sub> , 03AC <sub>16</sub> ) = "0" CTS/RTS function select bit (bit 2 at address 03A4 <sub>16</sub> , 03AC <sub>16</sub> ) = "0" Port P47, P77 direction register (bits 7 address 03EA <sub>16</sub> and 03EF <sub>16</sub> ) = "0"
	RTS output	CTS/RTS disable bit (bit 4 at address 03A4 <sub>16</sub> , 03AC <sub>16</sub> ) = "0" CTS/RTS function select bit (bit 2 at address 03A4 <sub>16</sub> , 03AC <sub>16</sub> ) = "1"
	Programmable I/O port	CTS/RTS disable bit (bit 4 at address 03A4 <sub>16</sub> , 03AC <sub>16</sub> ) = "1"

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



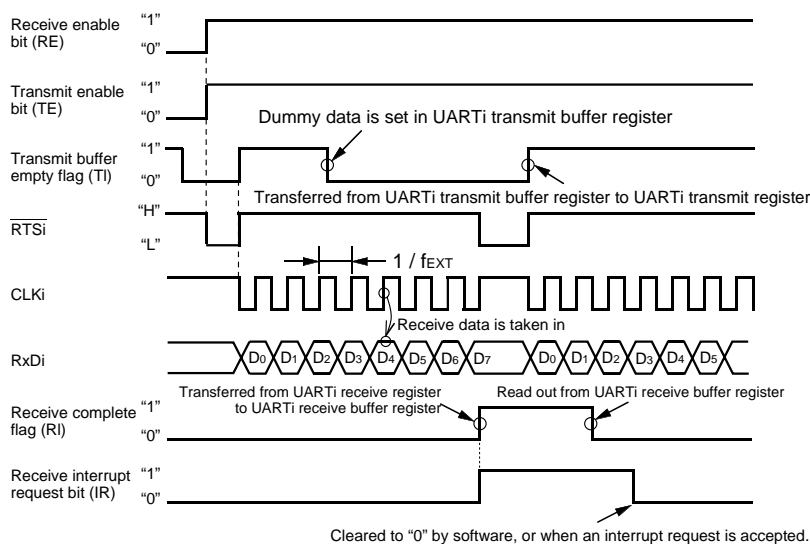
Shown in ( ) are bit symbols.  
 The above timing applies to the following settings:

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

$$T_c = T_{CLK} = 2(n + 1) / f_i$$

$f_i$ : frequency of BRGi's count source ( $f_1, f_8, f_{32}$ )  
 $n$ : value set to BRGi

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



$f_{EXT}$ : frequency of external clock

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".

Meet the following conditions 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 80. Typical transmit/receive timings in clock synchronous serial I/O mode

**(a) Polarity select function**

As shown in Figure 81, the CLK polarity select bit (bit 6 at addresses 03A4<sub>16</sub>, 03AC<sub>16</sub>) allows selection of the polarity of the transfer clock.

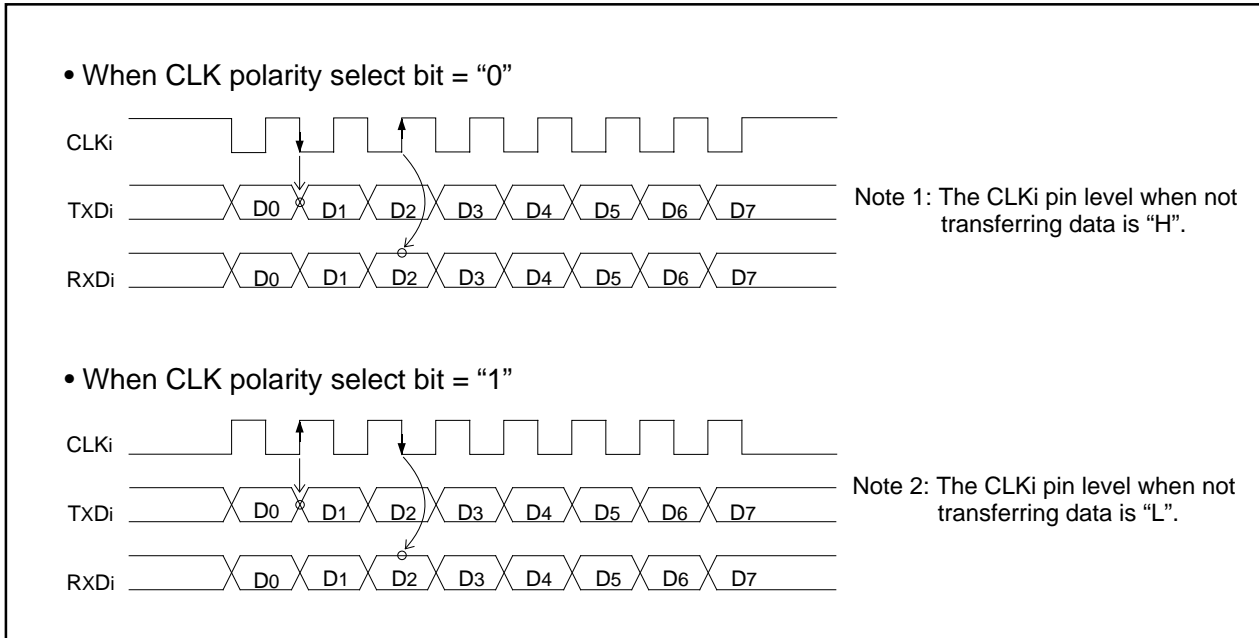


Figure 81. Polarity of transfer clock

**(b) LSB first/MSB first sel82GA-9, when the transfer format select bit (bit 7 at addresses 03A4<sub>16</sub>, 03AC<sub>16</sub>) = "0", the transfer format is "LSB first"; when the bit = "1", the transfer format is "MSB first".**

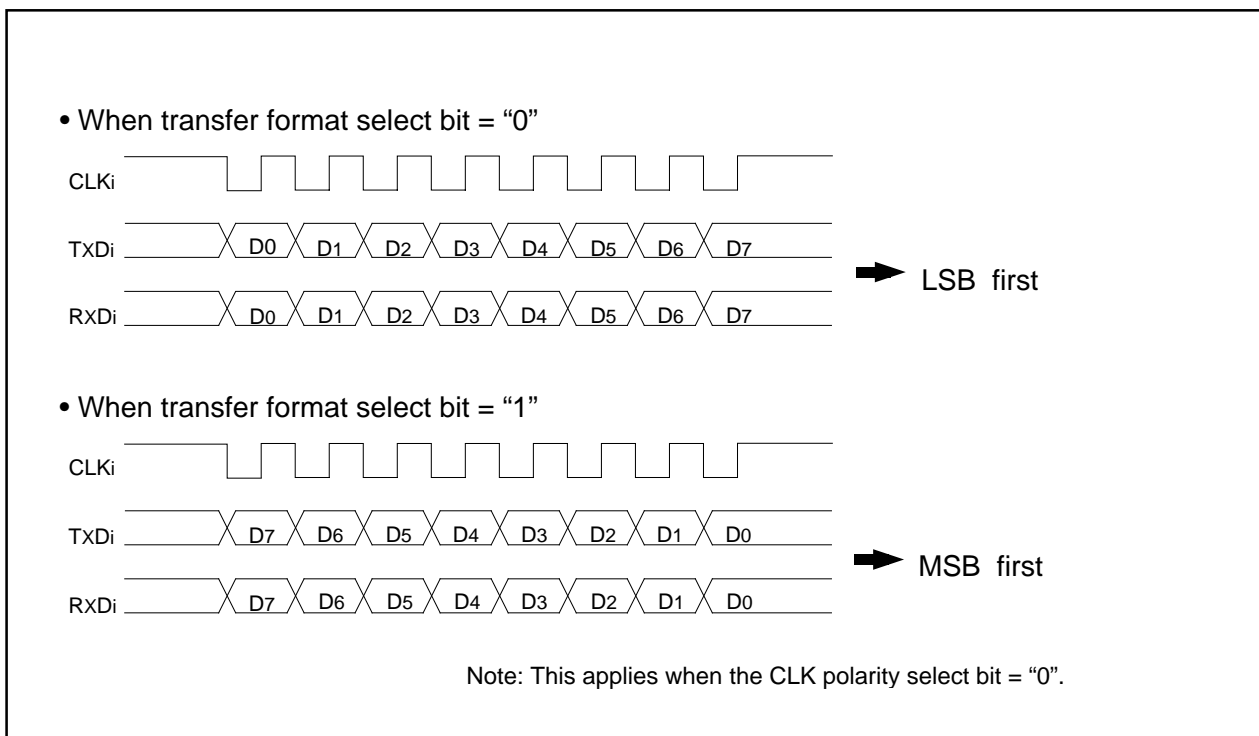
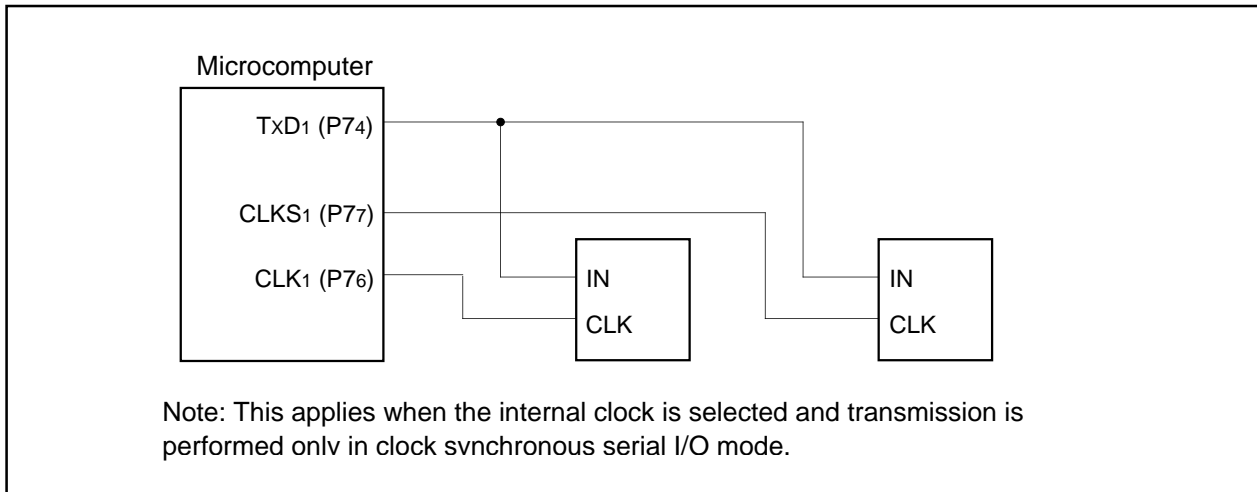


Figure 82. Transfer format

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

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 03B016). (See Figure 83.) The multiple pins function is valid only when the internal clock is selected for UART1. Note that when this function is selected,  $\overline{\text{CTS}}/\overline{\text{RTS}}$  function of UART1 cannot be used.



**Figure 83. 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 03B016) 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.



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

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

The UART allows transmitting and receiving data after setting the desired transfer rate and transfer data format. Table 26 lists the specifications of the UART mode. Figure 84 shows the UART<sub>i</sub> transmit/receive mode register.

**Table 26. Specifications of clock synchronous serial I/O mode**

Item	Specification
Transfer data format	<ul style="list-style-type: none"> <li>•Character bit (transfer data): 7 bits, 8 bits or 9 bits as selected</li> <li>•Start bit: 1 bit</li> <li>•Parity bit: Odd, even or nothing as selected</li> <li>•Stop bit: 1 bit or 2 bits as selected</li> </ul>
Transfer clock	<ul style="list-style-type: none"> <li>•When internal clock is selected (bit 3 at addresses 03A0<sub>16</sub>, 03A8<sub>16</sub> = "0") : f<sub>i</sub>/16(n+1) (Note 1) f<sub>i</sub> = f<sub>1</sub>, f<sub>8</sub>, f<sub>32</sub></li> <li>•When external clock is selected (bit 3 at addresses 03A0<sub>16</sub>, 03A8<sub>16</sub> = "1") : f<sub>EXT</sub>/16(n+1) (Note 1) (Note 2)</li> </ul>
Transmission/reception control	<ul style="list-style-type: none"> <li>•CTS function/RTS function/CTS, RTS function chosen to be invalid</li> </ul>
Transmission start condition	<ul style="list-style-type: none"> <li>•To start transmission, the following requirements must be met: <ul style="list-style-type: none"> <li>- Transmit enable bit (bit 0 at addresses 03A5<sub>16</sub>, 03AD<sub>16</sub>) = "1"</li> <li>- Transmit buffer empty flag (bit 1 at addresses 03A5<sub>16</sub>, 03AD<sub>16</sub>) = "0"</li> <li>- When CTS function is selected, CTS input level = "L"</li> </ul> </li> </ul>
Reception start condition	<ul style="list-style-type: none"> <li>•To start reception, the following requirements must be met: <ul style="list-style-type: none"> <li>- Receive enable bit (bit 2 at addresses 03A5<sub>16</sub>, 03AD<sub>16</sub>) = "1"</li> <li>- Start bit detection</li> </ul> </li> </ul>
Interrupt request generation timing	<ul style="list-style-type: none"> <li>•When transmitting <ul style="list-style-type: none"> <li>- Transmit interrupt cause select bits (bits 0,1 at address 03B0<sub>16</sub>) = "0": Interrupts requested when data transfer from UART<sub>i</sub> transfer buffer register to UART<sub>i</sub> transmit register is completed</li> <li>- Transmit interrupt cause select bits (bits 0, 1 at address 03B0<sub>16</sub>) = "1": Interrupts requested when data transmission from UART<sub>i</sub> transfer register is completed</li> </ul> </li> <li>•When receiving <ul style="list-style-type: none"> <li>- Interrupts requested when data transfer from UART<sub>i</sub> receive register to UART<sub>i</sub> receive buffer register is completed</li> </ul> </li> </ul>
Error detection	<ul style="list-style-type: none"> <li>•Overrun error (Note 3) This error occurs when the next data is ready before contents of UART<sub>i</sub> receive buffer register are read out</li> <li>•Framing error This error occurs when the number of stop bits set is not detected</li> <li>•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</li> <li>•Error sum flag This flag is set (= 1) when any of the overrun, framing, and parity errors is encountered</li> </ul>
select function	<ul style="list-style-type: none"> <li>•Sleep mode selection This mode is used to transfer data to and from one of multiple slave microcomputers</li> </ul>

Note 1: 'n' denotes the value 00<sub>16</sub> to FF<sub>16</sub> that is set to the UART<sub>i</sub> bit rate generator.

Note 2: f<sub>EXT</sub> is input from the CLK<sub>i</sub> pin.

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

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

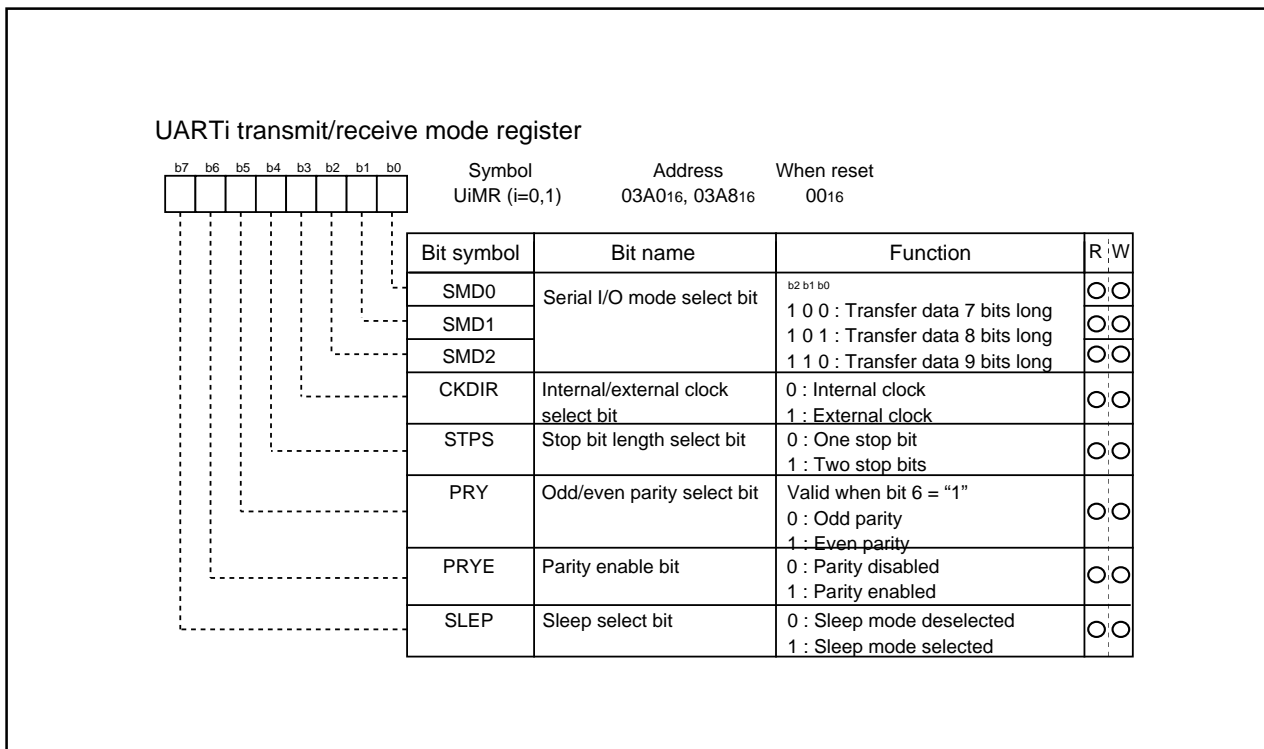
Figure 84. UART<sub>i</sub> transmit/receive mode register in UART mode

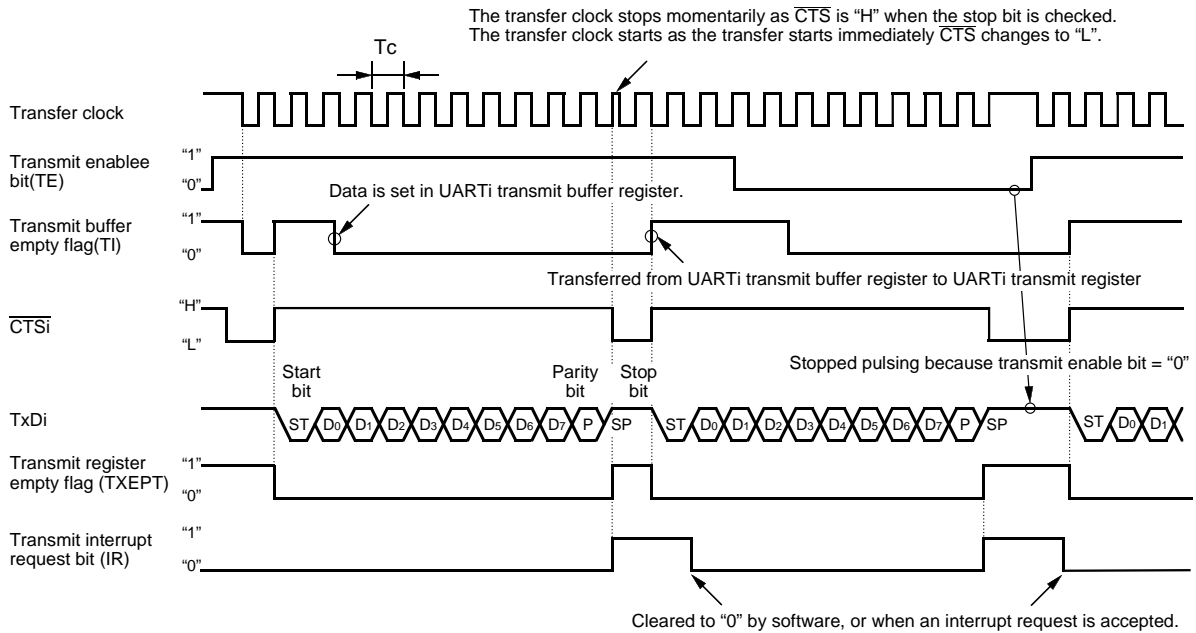
Table 27 lists the functions of the input/output pins during UART mode. Note that for a period from when the UART<sub>i</sub> operation mode is selected to when transfer starts, the Tx<sub>D</sub><sub>i</sub> pin outputs a "H". (If the N-channel open-drain is selected, this pin is in floating state.)

Table 27. Input/output pin functions in UART mode (i=0,1)

Pin name	Function	Method of selection
TxD <sub>i</sub> (P44, P74)	Serial data output	(Outputs dummy data when performing reception only)
RxD <sub>i</sub> (P45, P75)	Serial data input	Port P45, P75 direction register (bits 5 at address 03EA <sub>16</sub> and 03EF <sub>16</sub> ) = "0" (Can be used as an input port when performing transmission only)
CLK <sub>i</sub> (P46, P76)	Programmable I/O port	Internal/external clock select bit (bit 3 at address 03A0 <sub>16</sub> , 03A8 <sub>16</sub> ) = "0"
	Transfer clock input	Internal/external clock select bit (bit 3 at address 03A0 <sub>16</sub> , 03A8 <sub>16</sub> ) = "1"
CTS <sub>i</sub> /RTS <sub>i</sub> (P47, P77)	CTS input	CTS/RTS disable bit (bit 4 at address 03A4 <sub>16</sub> , 03AC <sub>16</sub> ) = "0" CTS/RTS function select bit (bit 2 at address 03A4 <sub>16</sub> , 03AC <sub>16</sub> ) = "0" Port P47, P77 direction register (bits 7 at address 03EA <sub>16</sub> and 03EF <sub>16</sub> ) = "0"
	RTS output	CTS/RTS disable bit (bit 4 at address 03A4 <sub>16</sub> , 03AC <sub>16</sub> ) = "0" CTS/RTS function select bit (bit 2 at address 03A4 <sub>16</sub> , 03AC <sub>16</sub> ) = "1"
	Programmable I/O port	CTS/RTS disable bit (bit 4 at address 03A4 <sub>16</sub> , 03AC <sub>16</sub> ) = "1"

Clock asynchronous serial I/O (UART) mode

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



Shown in ( ) are bit symbols.

The above timing applies to the following settings :

- Parity is enabled.
- One stop bit.
- $\overline{CTS}$  function is selected.
- Transmit interrupt cause select bit = "1".

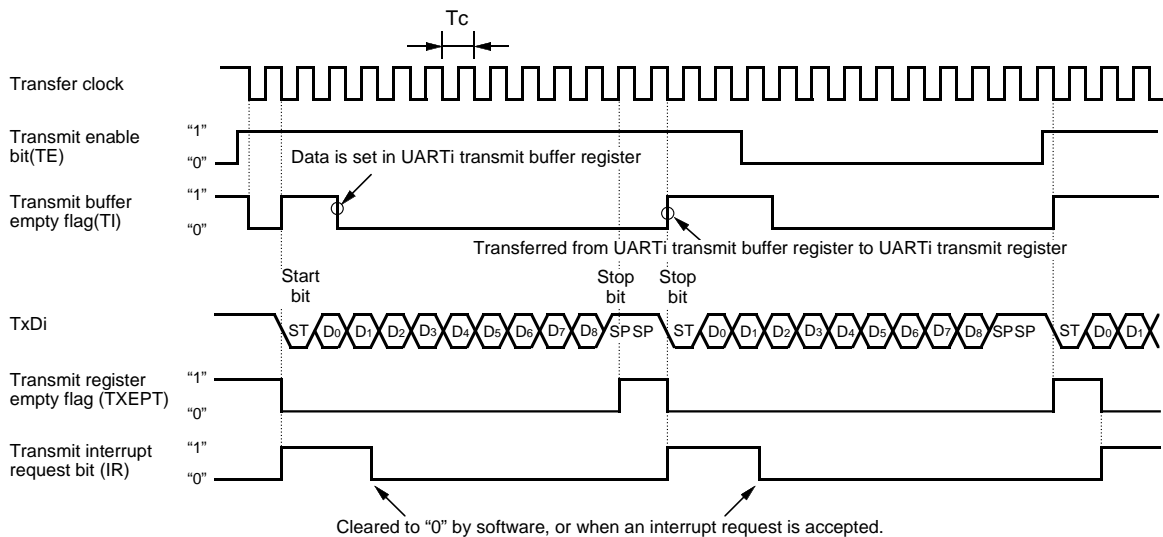
$$T_c = 16(n + 1) / f_i \text{ or } 16(n + 1) / f_{EXT}$$

$f_i$  : frequency of BRGi's count source ( $f_1, f_8, f_{32}$ )

$f_{EXT}$  : frequency of BRGi's 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)



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 causes select bit = "0".

$$T_c = 16(n + 1) / f_i \text{ or } 16(n + 1) / f_{EXT}$$

$f_i$  : frequency of BRGi's count source ( $f_1, f_8, f_{32}$ )

$f_{EXT}$  : frequency of BRGi's count source (external clock)

$n$  : value set to BRGi

Figure 85. Typical transmit timings in UART mode

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

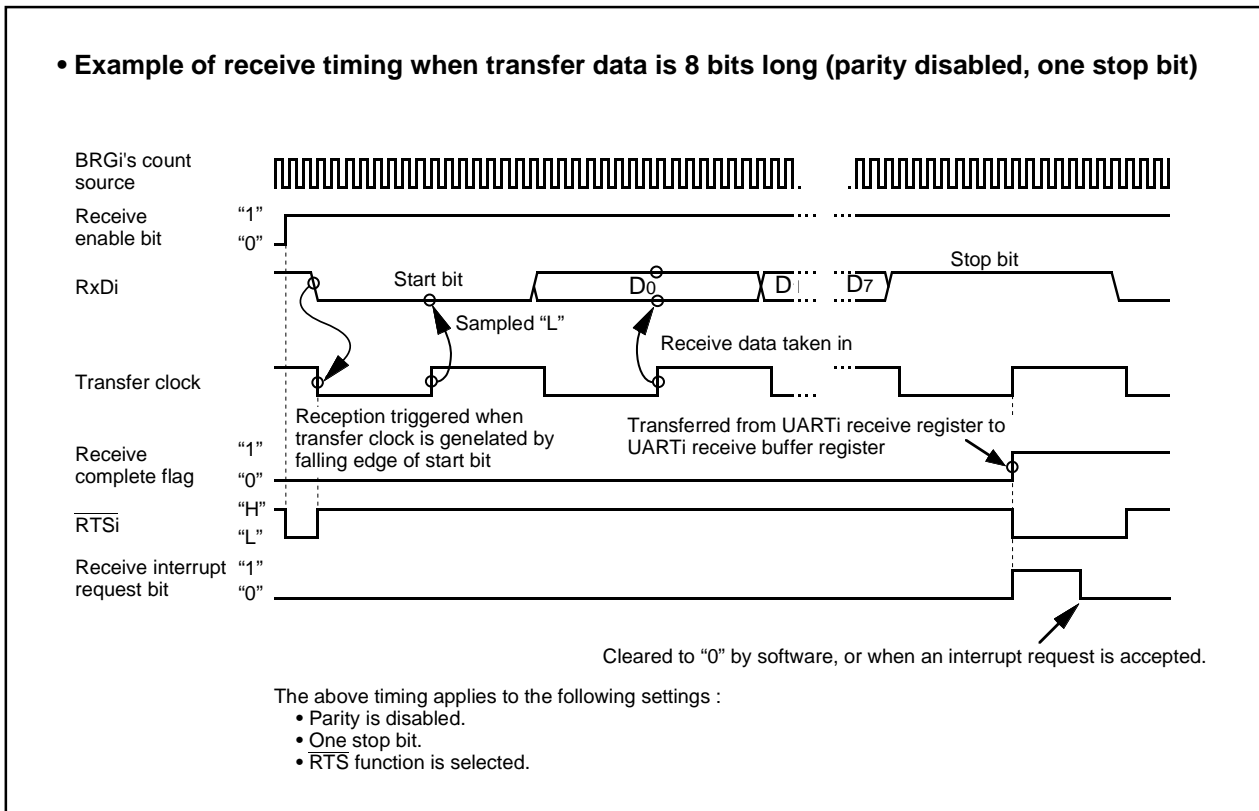


Figure 86. Typical receive timing in UART mode

**(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 03A016, 03A816) 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".

## Serial I/O2

Serial I/O2 is used as the clock synchronous serial I/O and has an ordinary mode and an automatic transfer mode. In the automatic transfer mode, serial transfer is performed through the serial I/O automatic transfer RAM which has up to 256 bytes (addresses 0040016 to 004FF16).

The SRDY2, SBUSY2 and SSTB2 pins each have a handshake I/O signal function and can select either "H" active or "L" active for active logic.

**Table 28. Specifications of clock synchronous serial I/O2**

Item	Specification
Serial mode	<ul style="list-style-type: none"> <li>• 8-bit serial I/O mode (non-automatic transfer)</li> <li>• Automatic transfer serial I/O mode</li> </ul>
Transfer data format	<ul style="list-style-type: none"> <li>• Transfer data length: 8 bits</li> <li>• Full duplex mode / transmit-only mode selected by bit 5 at address 034216</li> </ul>
Transfer clock	<ul style="list-style-type: none"> <li>• When internal clock is selected (bit 2 at address 034216 = "0") : selected by bits 5 to 7 at address 034816</li> <li>• When external clock is selected (bit 2 at address 034216 = "1") : Input from SCLK21 pin, SCLK22 pin(Note 2)</li> </ul>
Transfer rate	<ul style="list-style-type: none"> <li>• When internal clock is selected : <math>f(X_{IN})/4</math>, <math>f(X_{IN})/8</math>, <math>f(X_{IN})/16</math>, <math>f(X_{IN})/32</math>, <math>f(X_{IN})/64</math>, <math>f(X_{IN})/128</math>, <math>f(X_{IN})/256</math></li> <li>• When external clock is selected : input cycle 0.95 <math>\mu</math>s or less</li> </ul>
Transmission/reception control	<ul style="list-style-type: none"> <li>• SSTB2 output / SBUSY2 input or output / SRDY2 input or output chosen</li> </ul>
Transmission / reception start condition	<ul style="list-style-type: none"> <li>• To start transmission / reception, the following requirements must be met: <ul style="list-style-type: none"> <li>– Serial I/O initialization bit (bit 4 at address 034216) = "1"</li> <li>– When <math>\overline{SBUSY2}</math> input, or <math>\overline{SRDY2}</math> input is selected : selected input level = "H"</li> <li>– When <math>\overline{SBUSY2}</math> input, or <math>\overline{SRDY2}</math> input is selected : selected input level = "L"</li> </ul> </li> <li>• Furthermore, if external clock is selected, the following requirements must also be met: <ul style="list-style-type: none"> <li>– Input level of SCLK21 or SCLK22 = "H"</li> </ul> </li> </ul>
Transmission and reception stop condition	<ul style="list-style-type: none"> <li>• To stop transmission and reception, set serial I/O initialization bit (bit 4 at address 034216) to "0" regardless internal clock and external clock.</li> </ul>
Interrupt request generation timing	<ul style="list-style-type: none"> <li>• 8-bit serial I/O mode : Interrupts requested when 8-bit data transfer is completed</li> <li>• Automatic transfer serial I/O mode : Interrupts requested when last receive data transfer to Automatic transfer RAM</li> </ul>
Select function	<ul style="list-style-type: none"> <li>• SOUT2 P-channel output disable function CMOS output or N-channel open-drain output can be selected</li> <li>• LSB first/MSB first selection Whether transmission/reception begins with bit 0 or bit 7 can be selected</li> <li>• Serial I/O2 clock pin select bit Serial clock input/output can be selected; SCLK21 or SCLK22</li> <li>• SBUSY output, SSTB2 output select function (only automatic transfer serial mode) SBUSY output, SSTB2 output can be selected; 1-byte data transfer unit or all data transfer unit</li> <li>• SOUT2 pin control bit Either output active or high-impedance can be selected as a SOUT2 pin state at serial non-transfer .</li> </ul>

Note 1: It is necessary to set the serial I/O clock pin select bit ( bit 7 at address 034216)

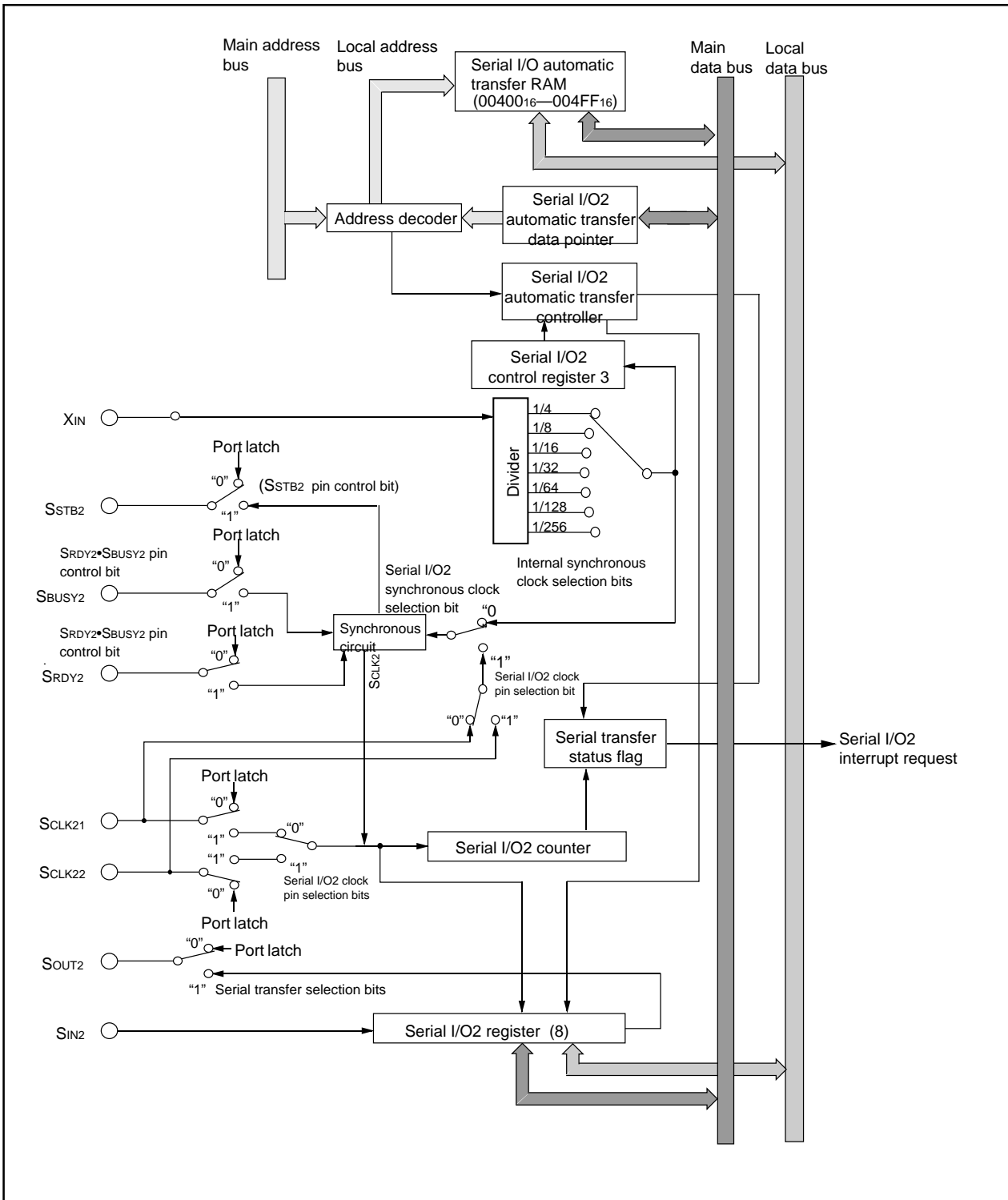


Figure 87. Block Diagram of Serial I/O2

## Serial I/O2

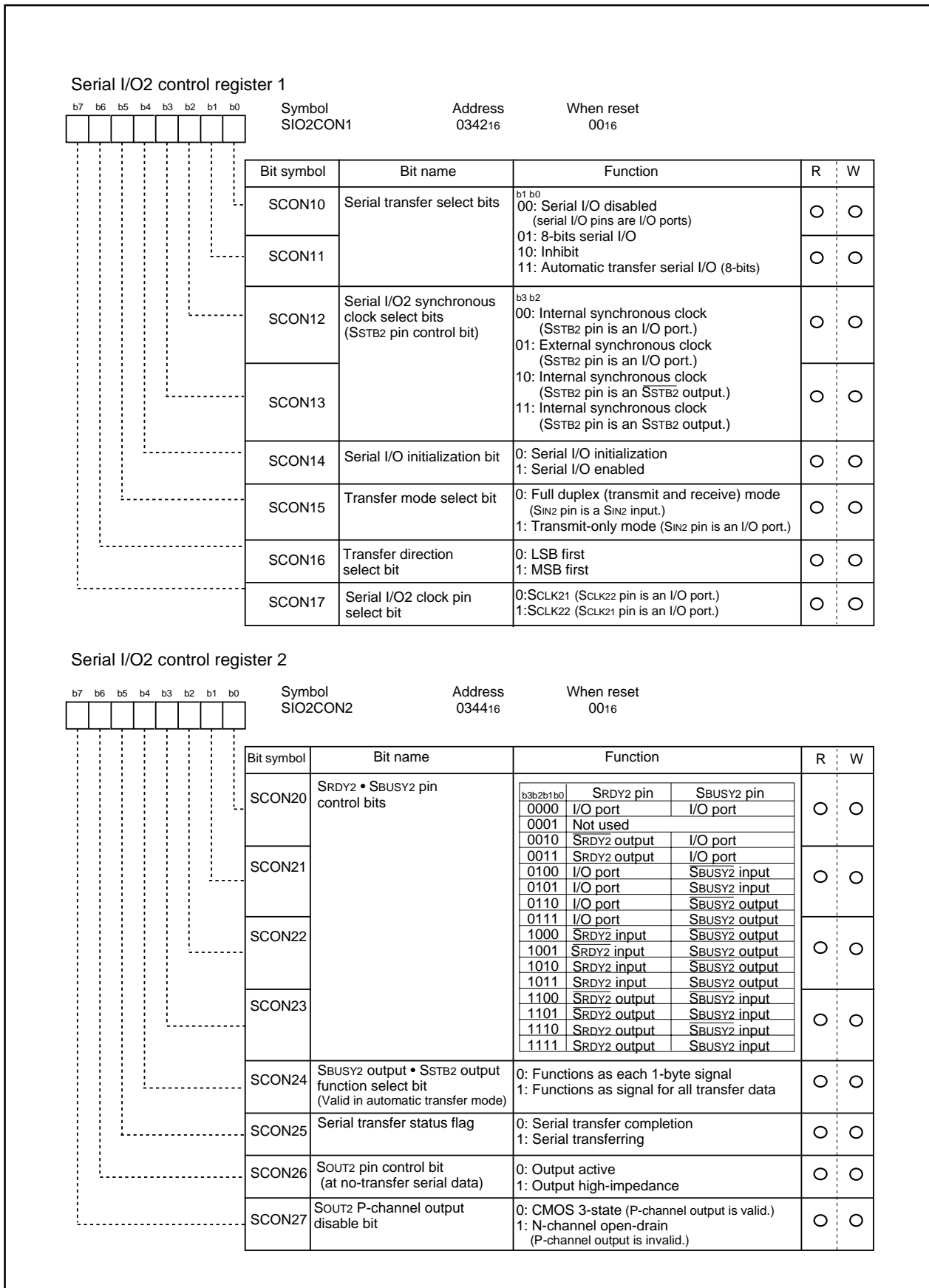


Figure 88. Serial I/O2 Control Registers 1, 2

## Serial I/O2

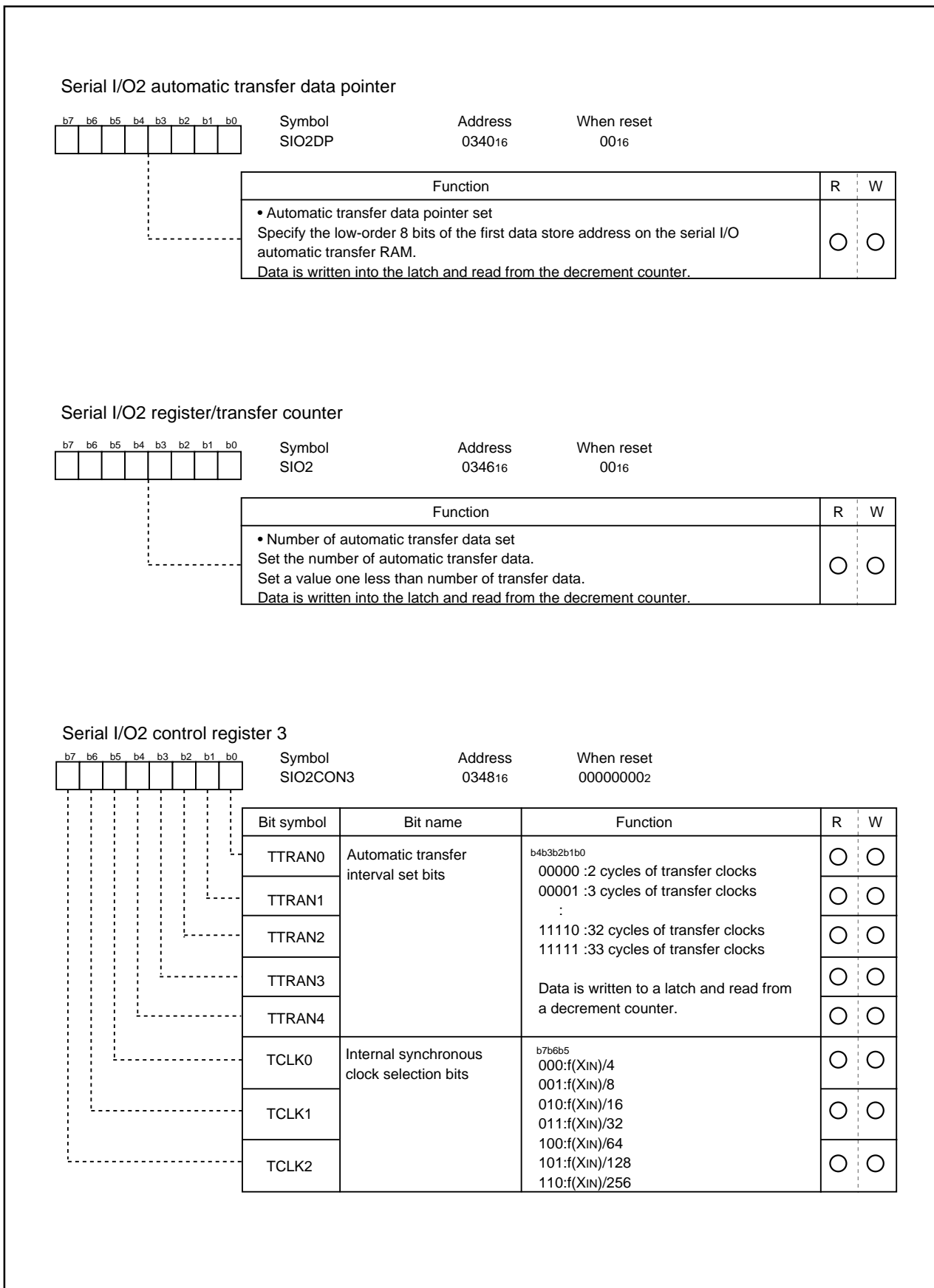


Figure 89. Serial I/O2 automatic transfer data pointer



Table 29 lists the functions of the serial I/O2 input/output pins

**Table 29. Functions of the serial I/O2 input/output pins**

Pin name	Function	Method of selection
SOUT2 (P94)	Serial data output	Port P94 direction register (bit 4 at address 03F316)= "1" SOUT2 P-channel output disable bit (bit 7 at address 034416)= "0", "1" SOUT2 pin control bit (bit 6 at address 034416)= "0", "1" (Outputs dummy data when performing reception only)
SIN2 (P93)	Serial data input	Port P93 direction register (bit 4 at address 03F316)= "0" Transfer mode select bit (bit 5 at address 034216)= "0" (Input/output port when transfer mode select bit (bit 5 at address 034216)= "1")
SCLK21 (P95)	Transfer clock output	Serial I/O2 synchronous clock select bits (bits 2, 3 at address 034216) = "00", "01" Serial I/O2 clock pin select bit (bit 7 at address 034216) = "0"
	Transfer clock input	Serial I/O2 synchronous clock select bits (bits 2, 3 at address 034216) = "01", "11" Serial I/O2 clock pin select bit (bit 7 at address 034216) = "0" Port P95 direction register (bit 5 at address 03F316)= "0"
SCLK22 (P96)	Transfer clock output	Serial I/O2 synchronous clock select bits (bits 2, 3 at address 034216) = "00", "01" Serial I/O2 clock pin select bit (bit 7 at address 034216) = "1"
	Transfer clock input	Serial I/O2 synchronous clock select bits (bits 2, 3 at address 034216) = "01", "11" Serial I/O2 clock pin select bit (bit 7 at address 034216) = "1" Port P96 direction register (bit 6 at address 03F316)= "0"
SRDY2 (P90)	SRDY input / output	Set by SRDY2 • SBUSY2 pin control bits (bits 0 to 3 at address 034416)
SBUSY2 (P91)	SBUSY input / output	Set by SRDY2 • SBUSY2 pin control bits (bits 0 to 3 at address 034416) SBUSY2 output • SSTB2 output function select bit (bit 4 at address 034416)= "0", "1"
SSTB2 (P92)	SSTB input / output	Serial I/O2 synchronous clock select bits (bits 2, 3 at address 034216) = "10", "11" SBUSY2 output • SSTB2 output function select bit (bit 4 at address 034416)= "0", "1"

## SOUT2 Output

Either output active or high-impedance can be selected as a SOUT2 pin state at serial non-transfer by the SOUT2 pin control bit (bit 6 of address 034416).

However, when the external synchronous clock is selected, perform the following setup to put the SOUT2 pin into a high-impedance state.

When the SCLK2i (i = 1, 2) input is "H" after completion of transfer, set the SOUT2 pin control bit to "1". When the SCLK2i (i = 1, 2) input goes to "L" after the start of the next serial transfer, the SOUT2 pin control bit is automatically reset to "0" and put into an output active state.

## Serial I/O2 Mode

There are two types of serial I/O2 modes: 8-bit serial I/O mode where automatic transfer RAM is not used, and an automatic transfer serial I/O mode.

### (1) 8-bit Serial I/O Mode

Address 0346<sub>16</sub> is assigned to the serial I/O2 register. When the internal synchronous clock is selected, a serial transfer of the 8-bit serial I/O is started by a write signal to the serial I/O2 register (address 0346<sub>16</sub>).

The serial transfer status flag (bit 5 of address 0344<sub>16</sub>) is set to "1" by writing into the serial I/O2 register and reset to "0" after completion of 8-bit transfer. At the same time, a serial I/O2 interrupt request occurs. If the transfer is completed, the receive data is read out from serial I/O2 register. When the external synchronous clock is selected, the contents of the serial I/O2 register are continuously shifted while transfer clocks are input to SCLK21 or SCLK22. Therefore, the clock needs to be controlled externally.

### (2) Automatic Transfer Serial I/O Mode

Address 0346<sub>16</sub> is assigned to the transfer counter (1-byte units). The serial I/O2 automatic transfer controller controls the write and read operations of the serial I/O2 register. The serial I/O automatic transfer RAM is mapped to addresses 00400<sub>16</sub> to 004FF<sub>16</sub>. Before starting transfer, make sure the 8 low-order bits of the address that contains the beginning data to be serially transferred is set to the automatic transfer data pointer (address 0340<sub>16</sub>).

When the internal synchronous clock is selected, the transfer interval is inserted between one data and another in the following cases:

1. When using no handshake signal
2. When using the SRDY2 output, SBUSY2 output, and SSTB2 output of the handshake signal independently
3. When using a combination of SRDY2 output and SSTB2 output or a combination of SBUSY2 output and SSTB2 output of the handshake signal

The transfer interval can be set in the range of 2 to 23 cycles using the automatic transfer interval set bit (bits 0–4 of address 0348<sub>16</sub>).

Also, when using SBUSY2 output as a signal for each occurrence of the all transfer data, a transfer interval is inserted before the system starts sending or receiving the first data and after the system finished sending or receiving the last data, not just between one data and another.

Furthermore, when using SSTB2 output, the transfer interval between each 1-byte data is extended by 2 cycles from the set value no matter how the SBUSY2 output. SSTB2 output function select bit (bit 4 of address 0344<sub>16</sub>) is set.

When using SBUSY2 output and SSTB2 output in combination as a signal for each occurrence of the all transfer data, the transfer interval after the system finished sending or receiving the last data is extended by 2 cycles from the set value.

When an external synchronous clock is selected, the automatic transfer interval is disabled.

When the internal synchronous clock is selected, automatic serial transfer starts by writing 1 less than the number of transfer bytes to the transfer counter (address 0346<sub>16</sub>). When an external sync clock is selected, automatic serial transfer starts by writing 1 less than the number of transfer bytes to the transfer counter and the transfer clock is input. In this case, allow for at least 5 cycles of internal system clock before the transfer clock is input after writing to the transfer counter.

Also, for data to data transfer intervals, allow at least 5 cycles of internal system clock reckoning from a rise of clock at the last bit of one-byte data.

Regardless of whether the internal or external synchronous clock is selected, the automatic transfer data pointer and the transfer counter are decreased after each 1-byte data is received and then written into the automatic transfer RAM. The serial transfer status flag (bit5 of address 0344<sub>16</sub>) is set to "1" by writing data into the transfer counter. The serial transfer status flag is reset to "0" after the last data is written into the automatic transfer RAM. At the same time, a serial I/O2 interrupt request occurs.

The values written in the automatic transfer data pointer (address 0340<sub>16</sub>) and the automatic transfer interval set bits (bit 0 to bit 4 of address 0348<sub>16</sub>) are held in the latch.

When data is written into the transfer counter, the values latched in the automatic transfer data pointer (address 0340<sub>16</sub>) and the automatic transfer interval set bits (bit 0 to bit 4) are transferred to the decrement counter.

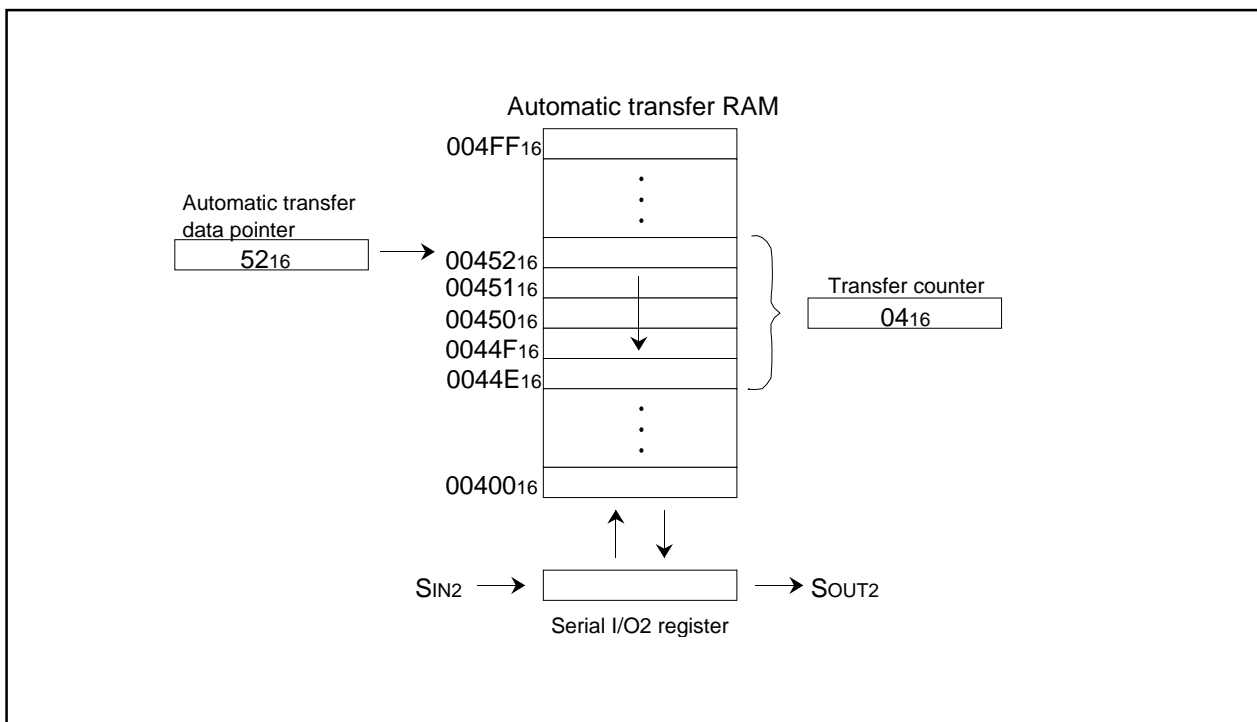


Figure 90. Automatic Transfer Serial I/O Operation

## Handshake Signal

There are five types of handshake signal : SSTB2 output, SBUSY2 input/output, and SRDY2 input/output.

### (1) SSTB2 output signal

The SSTB2 output is a signal to inform an end of transmission/reception to the serial transfer destination. The SSTB2 output signal can be used only when the internal synchronous clock is selected. In the initial status [ serial I/O initialization bit (bit 4 of address 034216) = "0" ], the SSTB2 output goes to "L" (bits 2, 3 of address 034216=11), or the  $\overline{\text{SSTB2}}$  output goes to "H" (bits 2, 3 of address 034216=10).

At the end of transmit/receive operation, after the all data of the serial I/O2 register (address 034616) is output from SOUT2, SSTB2 output is "H" (or  $\overline{\text{SSTB2}}$  output is "L") in the period of 1 cycle of the transfer clock. Furthermore, after 1 cycle, the serial transfer status flag (bit 5 of address 034416) is reset to "0".

In the automatic transfer serial I/O mode, whether the SSTB2 output is to be output at an end of each 1-byte data or after completion of transfer of all data can be selected by the SBUSY2 output • SSTB2 output function select bit (bit 4 of address 034416).

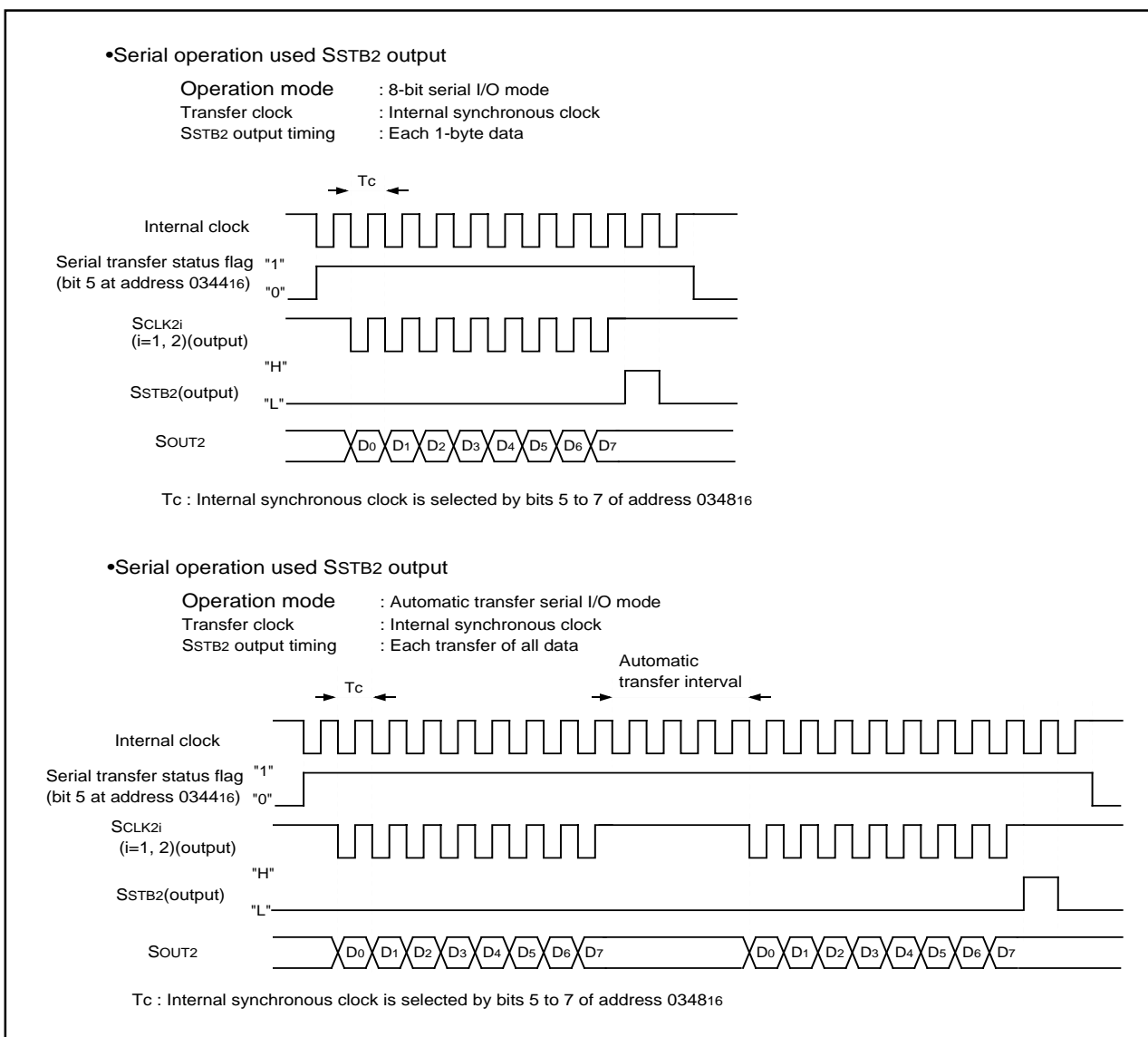
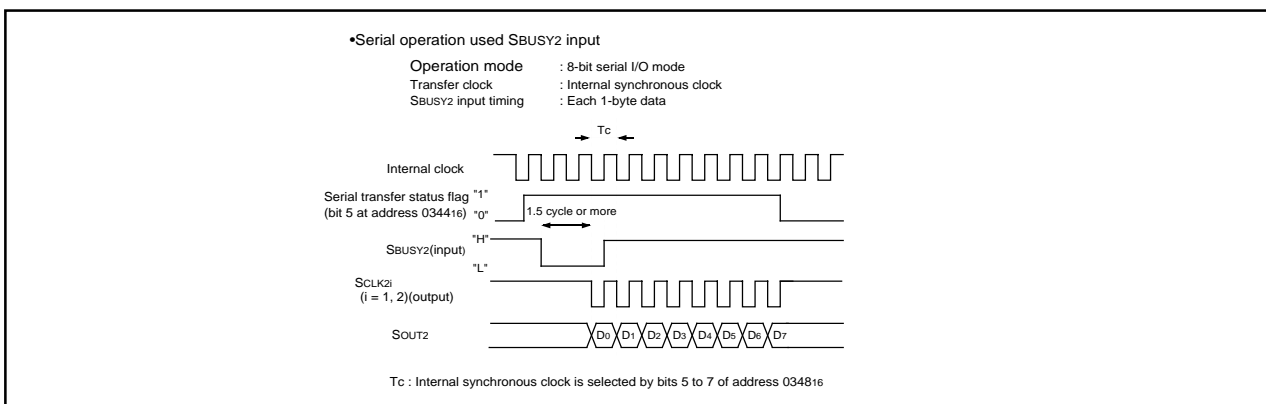


Figure 91. SSTB2 Output Operation

**(2) S<sub>BUSY2</sub> input signal**

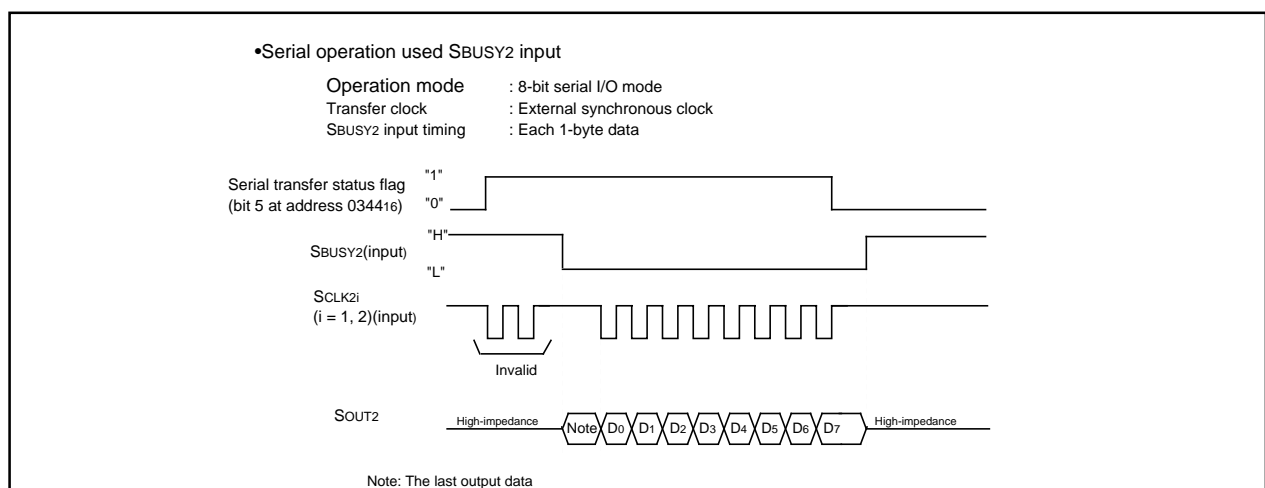
The S<sub>BUSY2</sub> input is a signal requested to stop of transmission/reception from the serial transfer destination.

When the internal synchronous clock is selected, input a "H" level signal into the S<sub>BUSY2</sub> input (or a "L" level signal into the  $\overline{\text{S}}\text{BUSY2}$  input) in the initial status [serial I/O initialization bit (bit 4 of address 034216) = "0"]. When a "L" level signal into the S<sub>BUSY2</sub> ( or "H" on  $\overline{\text{S}}\text{BUSY2}$  ) input for 1.5 cycles or more of transfer clock, transfer clocks are output from SCLK2<sub>i</sub> (i = 1, 2), and transmit/receive operation is started. When S<sub>BUSY2</sub> input is driven "H" ( or  $\overline{\text{S}}\text{BUSY2}$  input is driven "L" ) during transmit/receive operation, the transfer clock being output from SCLK2<sub>i</sub> (i = 1, 2) remains active until after the system finishes sending or receiving the designated number of bits, without stopping the transmit/receive operation immediately. The handshake unit of the 8-bit serial I/O is 8 bits, and that of the automatic transfer serial I/O is 8 bits.

**Figure 92. S<sub>BUSY2</sub> Input Operation (1)**

When the external synchronous clock is selected, input a "H" level signal into the S<sub>BUSY2</sub> input (or a "L" level signal into the  $\overline{\text{S}}\text{BUSY2}$  input) in the initial status[serial I/O initialization bit (bit 4 of address 034216) = "0"]. At this time, the transfer clock become invalid. The transfer clock become valid while a "L" level signal is input into the S<sub>BUSY2</sub> input (or a "H" level signal into the  $\overline{\text{S}}\text{BUSY2}$  input) and transmit/receive operation work.

When changing the input values into the S<sub>BUSY2</sub> (or  $\overline{\text{S}}\text{BUSY2}$ ) input at these operations, change them when the transfer clock input is in a "H" state. When the high-impedance of the SOUT2 output is selected by the SOUT2 pin control bit (bit 6 of address 034416), the SOUT2 becomes high-impedance, while a "H" level signal is input into the S<sub>BUSY2</sub> input (or a "L" level signal into the  $\overline{\text{S}}\text{BUSY2}$  input.)

**Figure 93. S<sub>BUSY2</sub> Input Operation (2)**

### (3) S<sub>BUSY2</sub> output signal

The S<sub>BUSY2</sub> output is a signal which requests to stop of transmission/reception to the serial transfer destination. In the automatic transfer serial I/O mode, regardless of the internal or external synchronous clock, whether the S<sub>BUSY2</sub> output is to be output at transfer of each 1-byte data or during transfer of all data can be selected by the S<sub>BUSY2</sub> output • S<sub>STB2</sub> output function select bit (bit 4 of address 034416). In the initial status[ serial I/O initialization bit (bit 4 of address 034216) = "0" ], the status in which the S<sub>BUSY2</sub> outputs "H" (or the  $\overline{\text{S}}\text{BUSY2}$  outputs "L").

When the internal synchronous clock is selected, in the 8-bit serial I/O mode and the automatic transfer serial I/O mode (S<sub>BUSY2</sub> output function: each 1-byte signal is selected), the S<sub>BUSY2</sub> output goes to "L" (or the  $\overline{\text{S}}\text{BUSY2}$  output goes to "H") before 0.5 cycle of the timing at which the transfer clock goes to "L". In the automatic transfer serial I/O mode (the S<sub>BUSY2</sub> output function: all transfer data is selected), the S<sub>BUSY2</sub> output goes to "L" (or the  $\overline{\text{S}}\text{BUSY2}$  output goes to "H") when the first transmit data is written into the serial I/O2 register (address 034616).

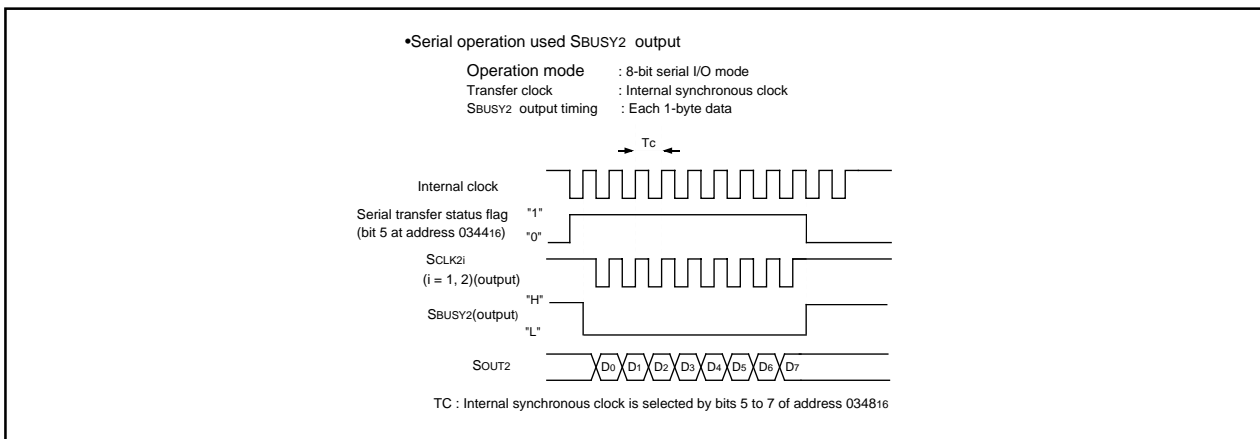


Figure 94. S<sub>BUSY2</sub> Output Operation (1)

When the external synchronous clock is selected, the S<sub>BUSY2</sub> output goes to "L" (or the  $\overline{\text{S}}\text{BUSY2}$  output goes to "H") when transmit data is written into the serial I/O2 register(address 034616), regardless of the serial I/O transfer mode.

At termination of transmit/receive operation, in the 8-bit serial I/O mode, the S<sub>BUSY2</sub> output goes to "H" (or the  $\overline{\text{S}}\text{BUSY2}$  output returns to "L"), when the serial transfer status flag is set to "0", regardless of whether the internal or external synchronous clock is selected. Furthermore, in the automatic transfer serial I/O mode (S<sub>BUSY2</sub> output function: each 1-byte signal is selected), the S<sub>BUSY2</sub> output goes to "H" (or the  $\overline{\text{S}}\text{BUSY2}$  output goes to "L") each time 1-byte of receive data is written into the automatic transfer RAM.

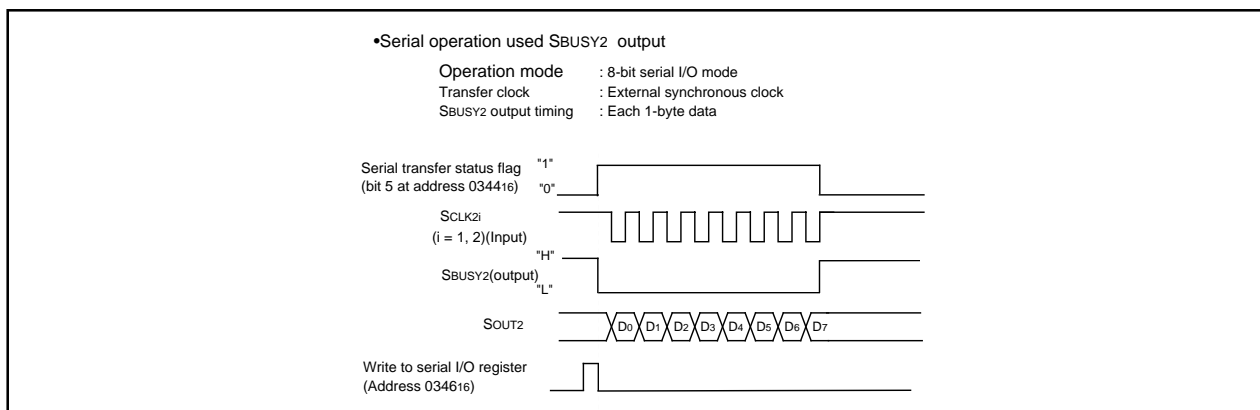


Figure 95. S<sub>BUSY2</sub> Output Operation (2)

Serial I/O2

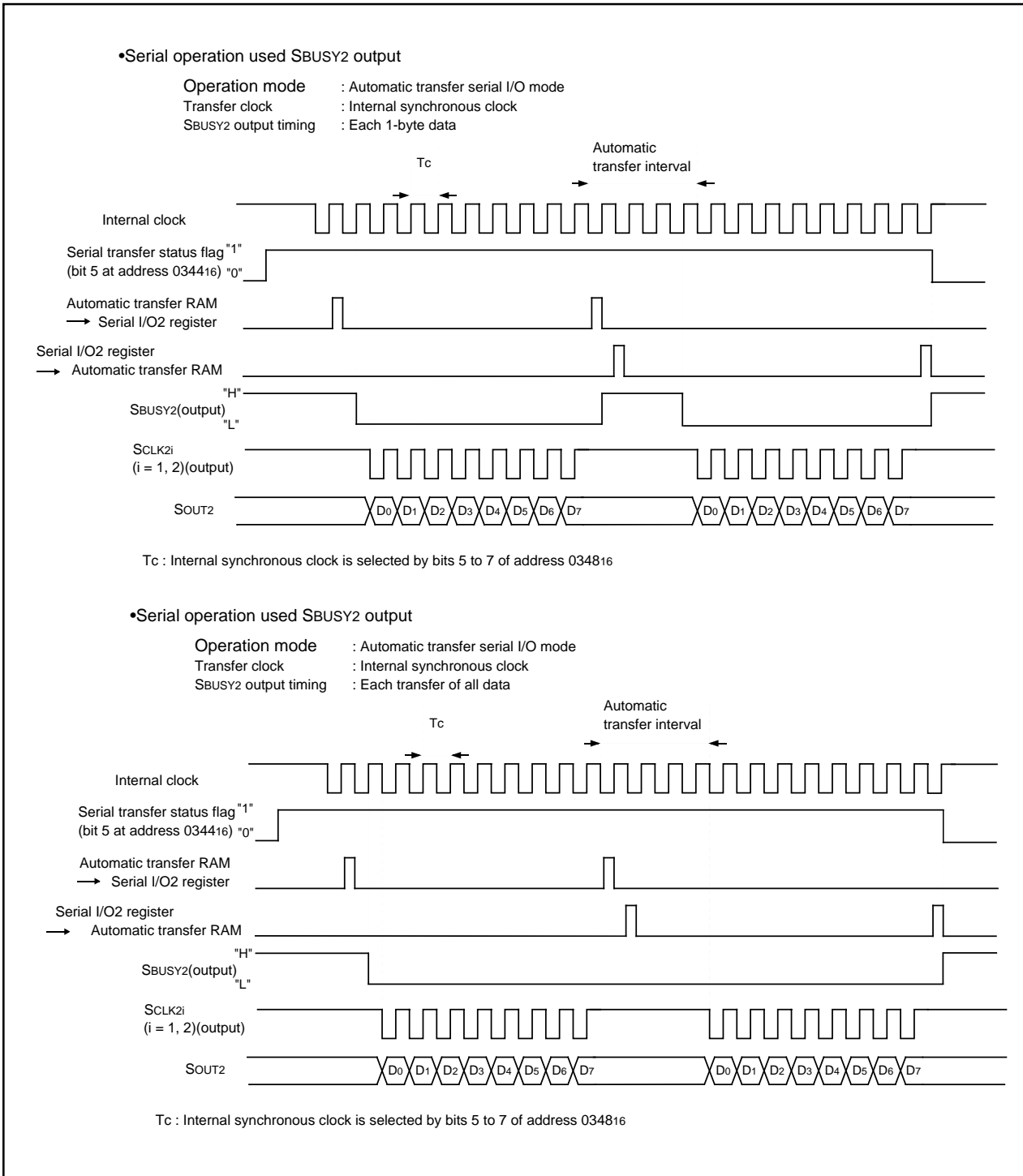


Figure 96. SBUSY2 Output Operation (3)

#### (4) SRDY2 output signal

The SRDY2 output is a transmit/receive enable signal which informs the serial transfer destination that transmit/receive is ready. In the initial status[serial I/O initialization bit (bit 4 of address 034216) = "0" ], the SRDY2 output goes to "L" (or the  $\overline{\text{SRDY2}}$  output goes to "H"). When the transmitted data is written to the serial I/O2 register (address 034616), the SRDY2 output goes to "H" (or the  $\overline{\text{SRDY2}}$  output goes to "L"). When a transmit/receive operation is started and the transfer clock goes to "L", the SRDY2 output goes to "L" (or the  $\overline{\text{SRDY2}}$  output goes to "H").

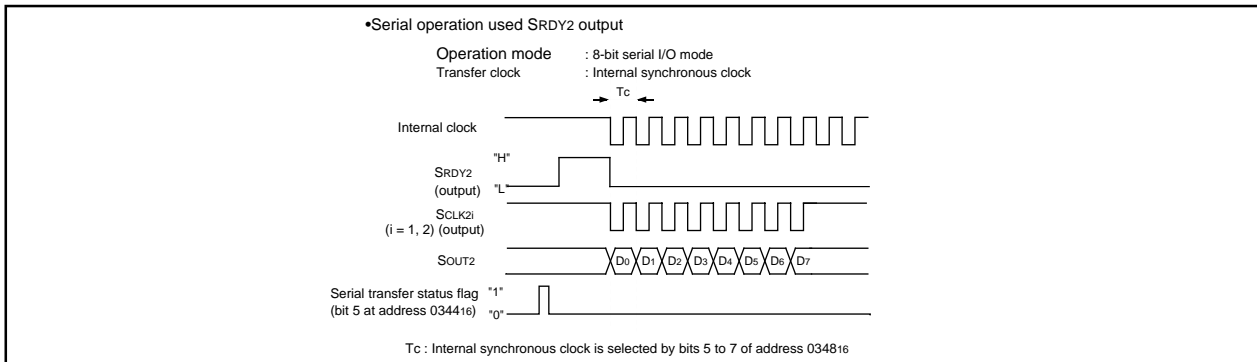


Figure 97. SRDY2 Output Operation

#### (5) SRDY2 input signal

The SRDY2 input is a signal for receiving a transmit/receive ready completion signal from the serial transfer destination. The SRDY2 input signal becomes valid only when the SRDY2 input and the SBUSY2 output are used.

When the internal synchronous clock is selected, input a "L" level signal into the SRDY2 input (or a "H" level signal into the  $\overline{\text{SRDY2}}$  input) in the initial status[serial I/O initialization bit (bit 4 of address 034216) = "0" ]. When a "H" level signal is input into the SRDY2 input (or a "L" level signal is input into the  $\overline{\text{SRDY2}}$  input) for a period of 1.5 cycles or more of transfer clock, transfer clocks are output from the SCLK2i (i = 1, 2) output and a transmit/receive operation is started. When SRDY2 input is driven "L" (or  $\overline{\text{SRDY2}}$  input is driven "H") during transmit/receive operation, the transfer clock being output from SCLK2i (i = 1, 2) remains active until after the system finishes sending or receiving the designated number of bits, without stopping the transmit/receive operation immediately.

The handshake unit of the 8-bit serial I/O is 8 bits, and that of the automatic transfer serial I/O is 8 bits. When the external synchronous clock is selected, the SRDY2 input becomes one of the triggers to output the SBUSY2 signal. To start a transmit/receive operation (SBUSY2 output: "L", (or  $\overline{\text{SBUSY2}}$  output: "H")), input a "H" level signal into the SRDY2 input (or a "L" level signal into the  $\overline{\text{SRDY2}}$  input,) and also write transmit data into the serial I/O2 register (address 034616).

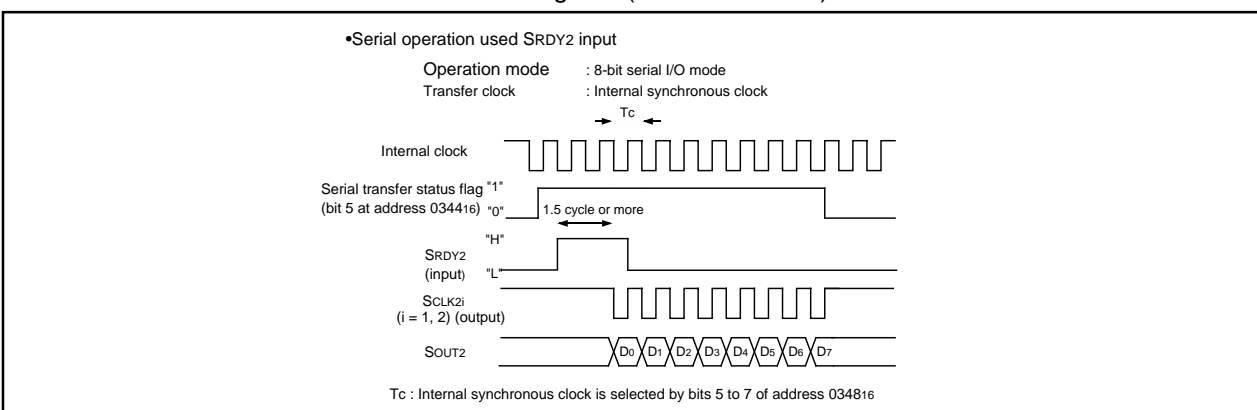


Figure 98. SRDY2 Input Operation



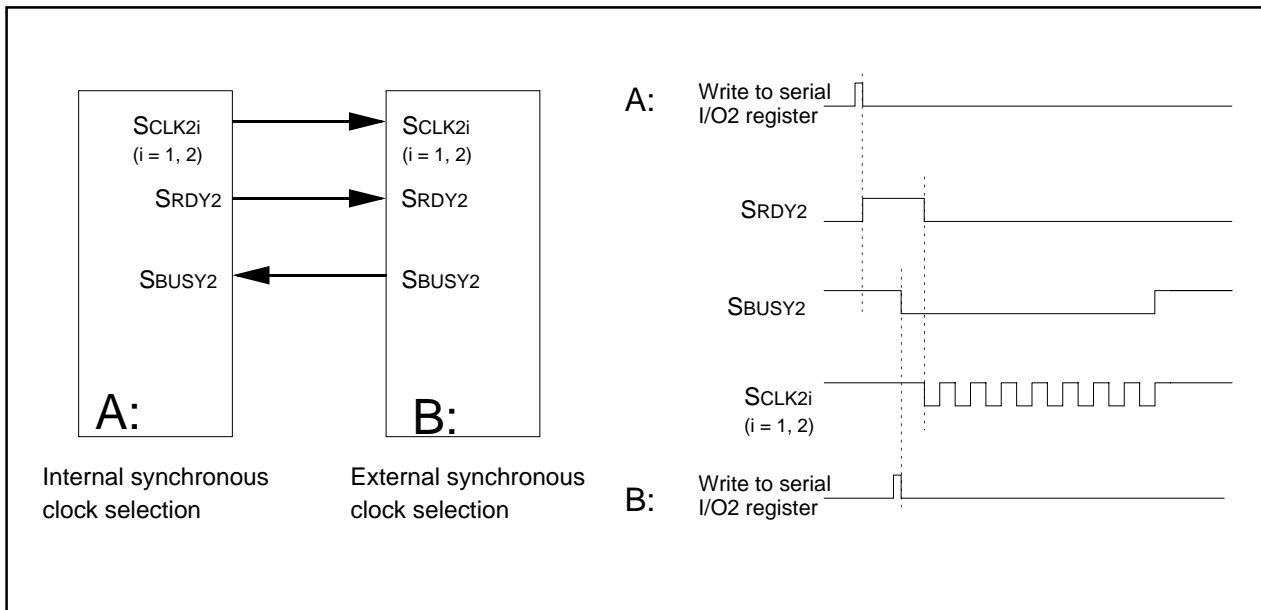


Figure 99. Handshake Operation at Serial I/O2 Mutual Connecting (1)

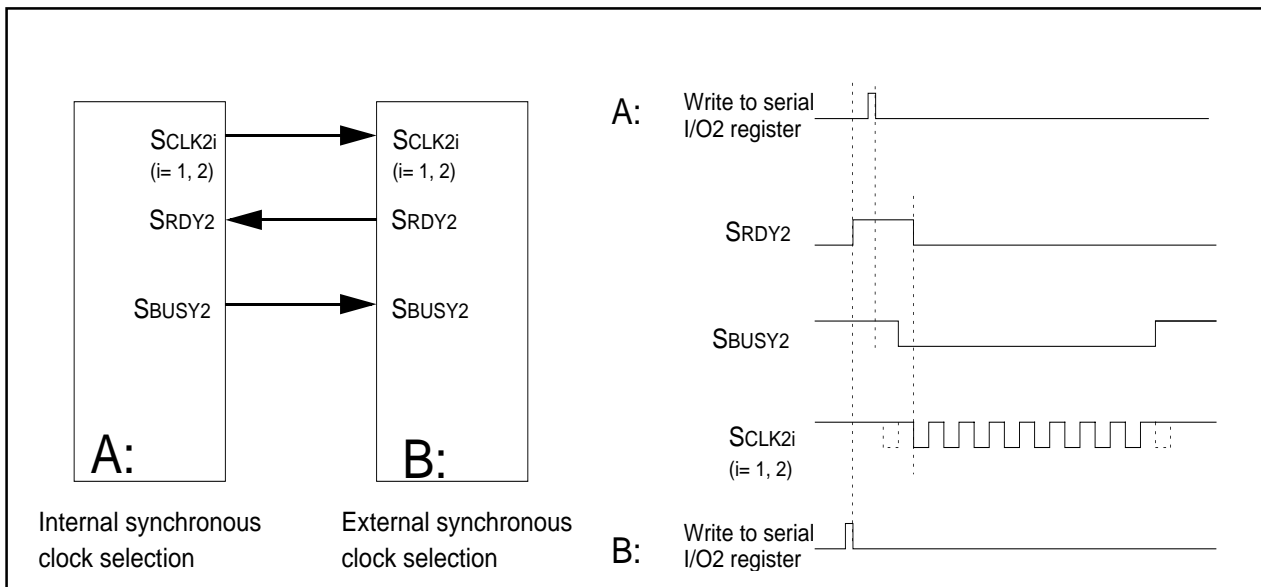


Figure 100. Handshake Operation at Serial I/O2 Mutual Connecting (2)

## A-D Converter

The A-D converter consists of one 10-bit successive approximation A-D converter circuit with a capacitive coupling amplifier. Pins P10<sub>0</sub> to P10<sub>7</sub> 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<sub>16</sub>) can be used to isolate the resistance ladder of the A-D converter from the reference voltage input pin (VREF) when the A-D converter is not used. Doing so stops any current flowing into the resistance ladder from VREF, reducing the power dissipation. When using the A-D converter, start A-D conversion only after setting bit 5 of 03D7<sub>16</sub> to connect VREF.

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

Table 30 shows the performance of the A-D converter. Figure 101 shows the block diagram of the A-D converter, and Figures 102 and 103 show the A-D converter-related registers.

**Table 30. Performance of A-D converter**

Item	Performance
Method of A-D conversion	Successive approximation (capacitive coupling amplifier)
Analog input voltage (Note 1)	0V to AVCC (VCC)
Operating clock $\phi_{AD}$ (Note 2)	VCC = 5V $f_{AD}$ /divide-by-2 of $f_{AD}$ /divide-by-4 of $f_{AD}$ , $f_{AD}=f(XIN)$ VCC = 3V divide-by-2 of $f_{AD}$ /divide-by-4 of $f_{AD}$ , $f_{AD}=f(XIN)$
Resolution	8-bit or 10-bit (selectable)
Absolute precision	VCC = 5V <ul style="list-style-type: none"> <li>• Without sample and hold function <math>\pm 3\text{LSB}</math></li> <li>• With sample and hold function (8-bit resolution) <math>\pm 2\text{LSB}</math></li> <li>• Without sample and hold function (10-bit resolution) <math>\pm 3\text{LSB}</math></li> </ul> VCC = 3V <ul style="list-style-type: none"> <li>• Without sample and hold function (8-bit resolution)(Note 3) <math>\pm 2\text{LSB}</math></li> </ul>
Operating modes	One-shot mode, repeat mode, single sweep mode, repeat sweep mode 0, and repeat sweep mode 1
Analog input pins	8pins (AN <sub>0</sub> to AN <sub>7</sub> )
A-D conversion start condition	•Software trigger A-D conversion starts when the A-D conversion start flag changes to "1"
Conversion speed per pin	•Without sample and hold function 8-bit resolution: 49 $\phi_{AD}$ cycles, 10-bit resolution: 59 $\phi_{AD}$ cycles • With sample and hold function 8-bit resolution: 28 $\phi_{AD}$ cycles, 10-bit resolution: 33 $\phi_{AD}$ cycles

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

Note 2: Without sample and hold function, set the  $\phi_{AD}$  frequency to 250kHz min.

With the sample and hold function, set the  $\phi_{AD}$  frequency to 1MHz min.

Note 3: Only mask ROM version.

A-D converter

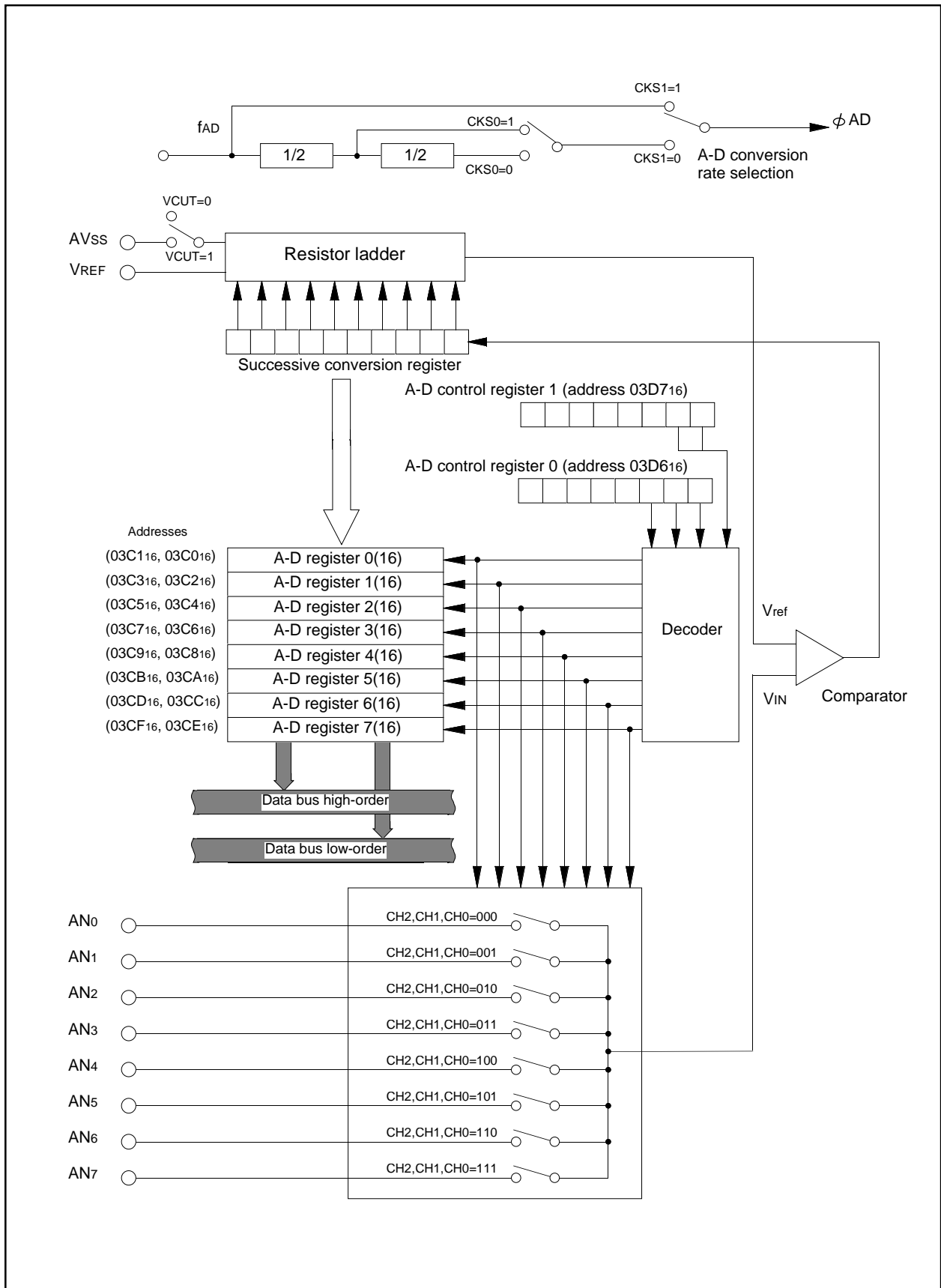


Figure 101. Block diagram of A-D converter

## A-D converter

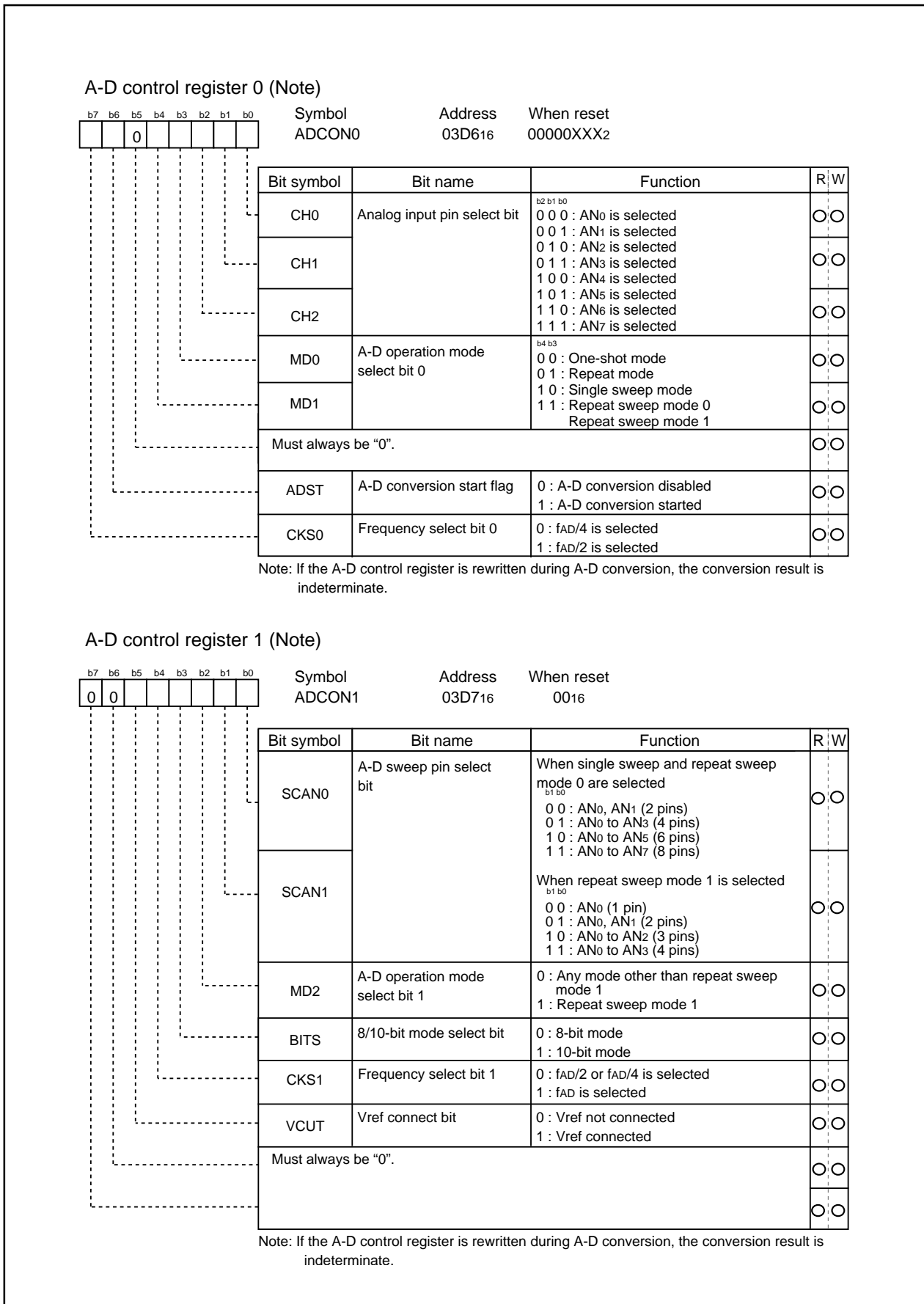


Figure 102. A-D converter-related registers (1)

## A-D converter

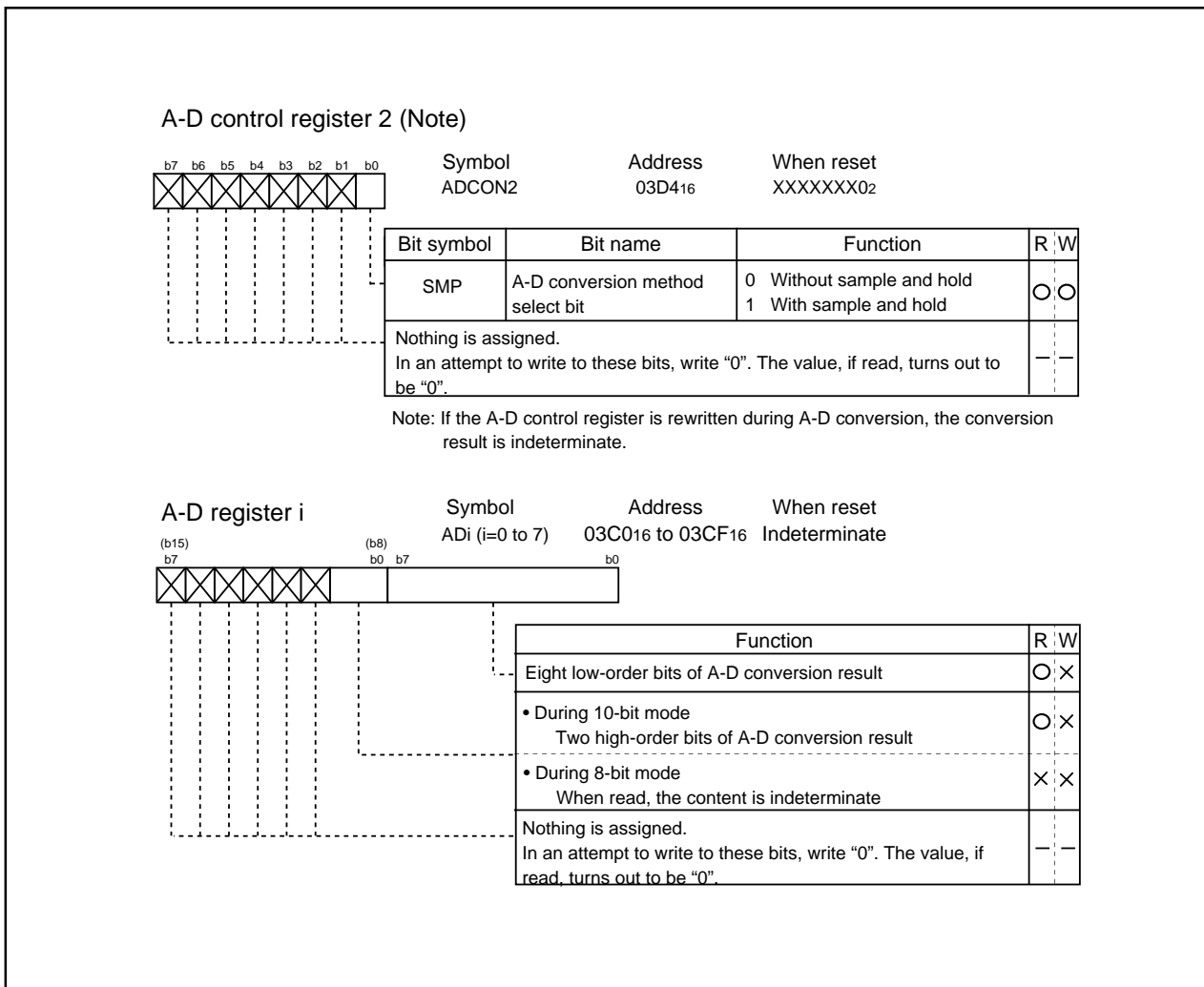


Figure 103. A-D converter-related registers (2)

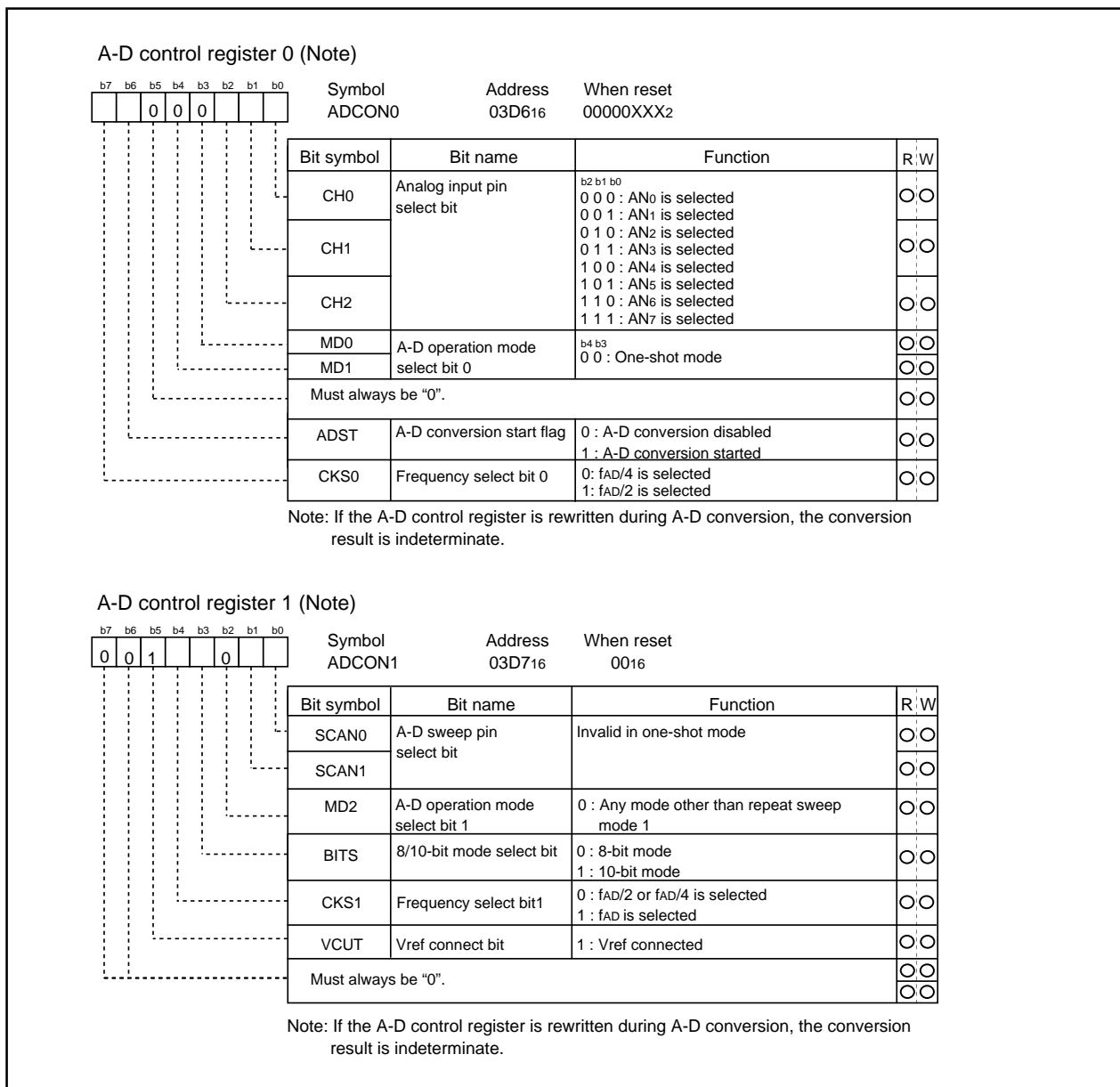
## A-D converter

**(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 31 shows the specifications of one-shot mode. Figure 104 shows the A-D control register in one-shot mode.

**Table 31. 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	<ul style="list-style-type: none"> <li>End of A-D conversion (A-D conversion start flag changes to "0")</li> <li>Writing "0" to A-D conversion start flag</li> </ul>
Interrupt request generation timing	End of A-D conversion
Input pin	One of AN0 to AN7, as selected
Reading of result of A-D converter	Read A-D register corresponding to selected pin

**Figure 104. A-D conversion register in one-shot mode**

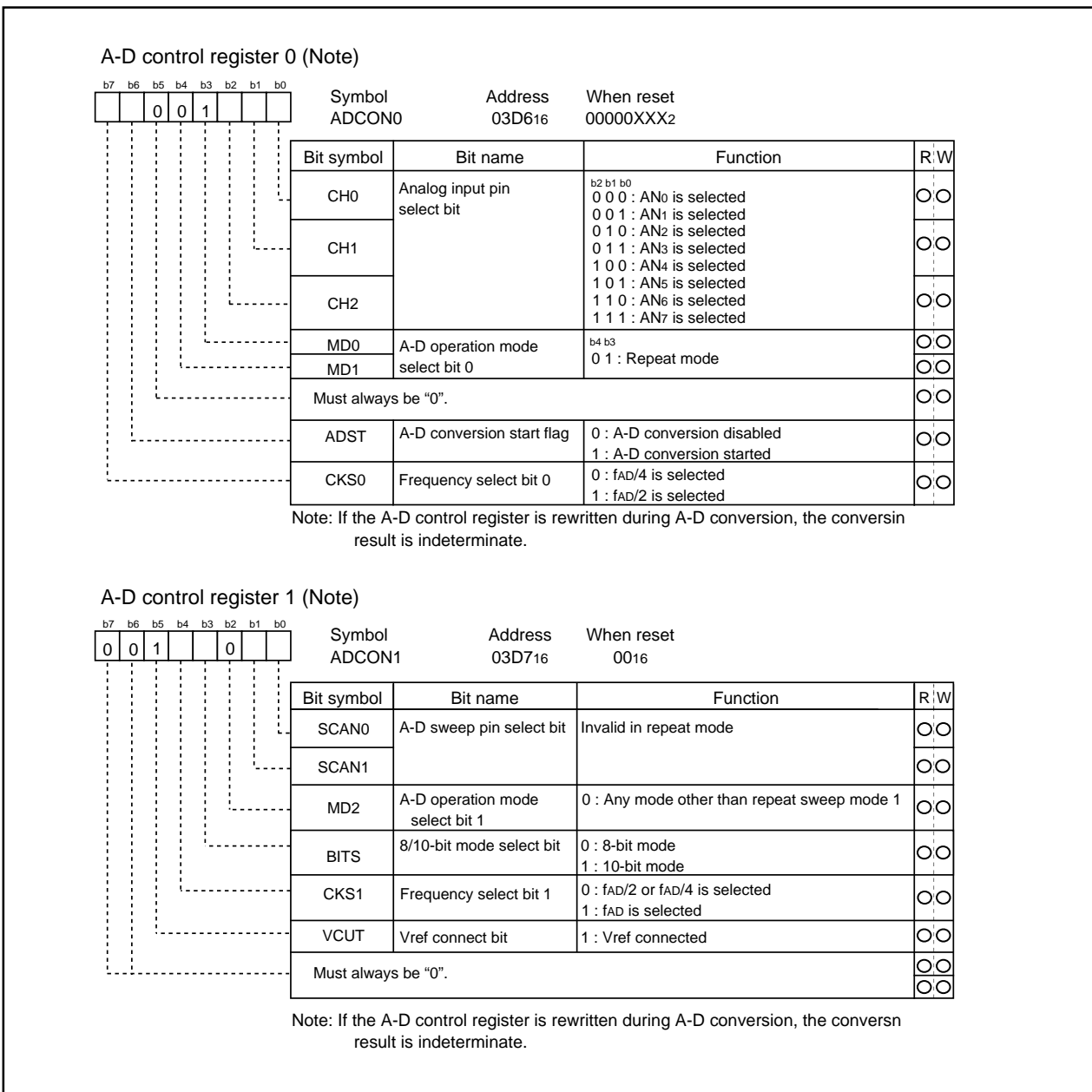
## A-D converter

**(2) Repeat mode**

In repeat mode, the pin selected using the analog input pin select bit is used for repeated A-D conversion. Table 32 shows the specifications of repeat mode. Figure 105 shows the A-D control register in repeat mode.

**Table 32. Repeat mode specifications**

Item	Specification
Function	The pin selected by the analog input pin select bit is used for repeated A-D conversion
Start condition	Writing "1" to A-D conversion start flag
Stop condition	Writing "0" to A-D conversion start flag
Interrupt request generation timing	None generated
Input pin	One of AN0 to AN7, as selected
Reading of result of A-D converter	Read A-D register corresponding to selected pin

**Figure 105. A-D conversion register in repeat mode**

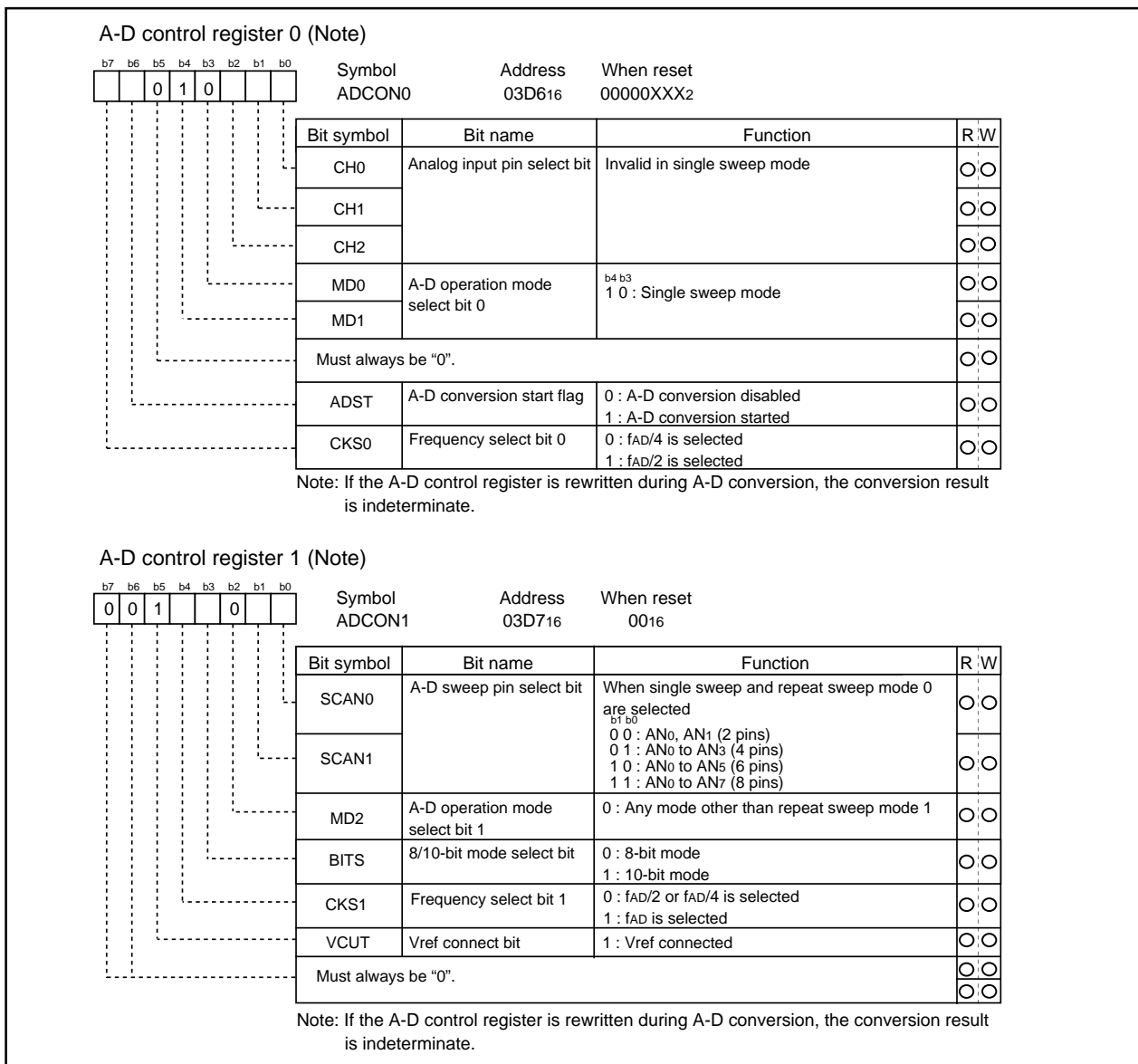
## A-D converter

**(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 33 shows the specifications of single sweep mode. Figure 106 shows the A-D control register in single sweep mode.

**Table 33. 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	<ul style="list-style-type: none"> <li>•End of A-D conversion (A-D conversion start flag changes to "0", except when external trigger is selected)</li> <li>•Writing "0" to A-D conversion start flag</li> </ul>
Interrupt request generation timing	End of A-D conversion
Input pin	AN0 and AN1 (2 pins), AN0 to AN3 (4 pins), AN0 to AN5 (6 pins), or AN0 to AN7 (8 pins)
Reading of result of A-D converter	Read A-D register corresponding to selected pin

**Figure 106. A-D conversion register in single sweep mode**



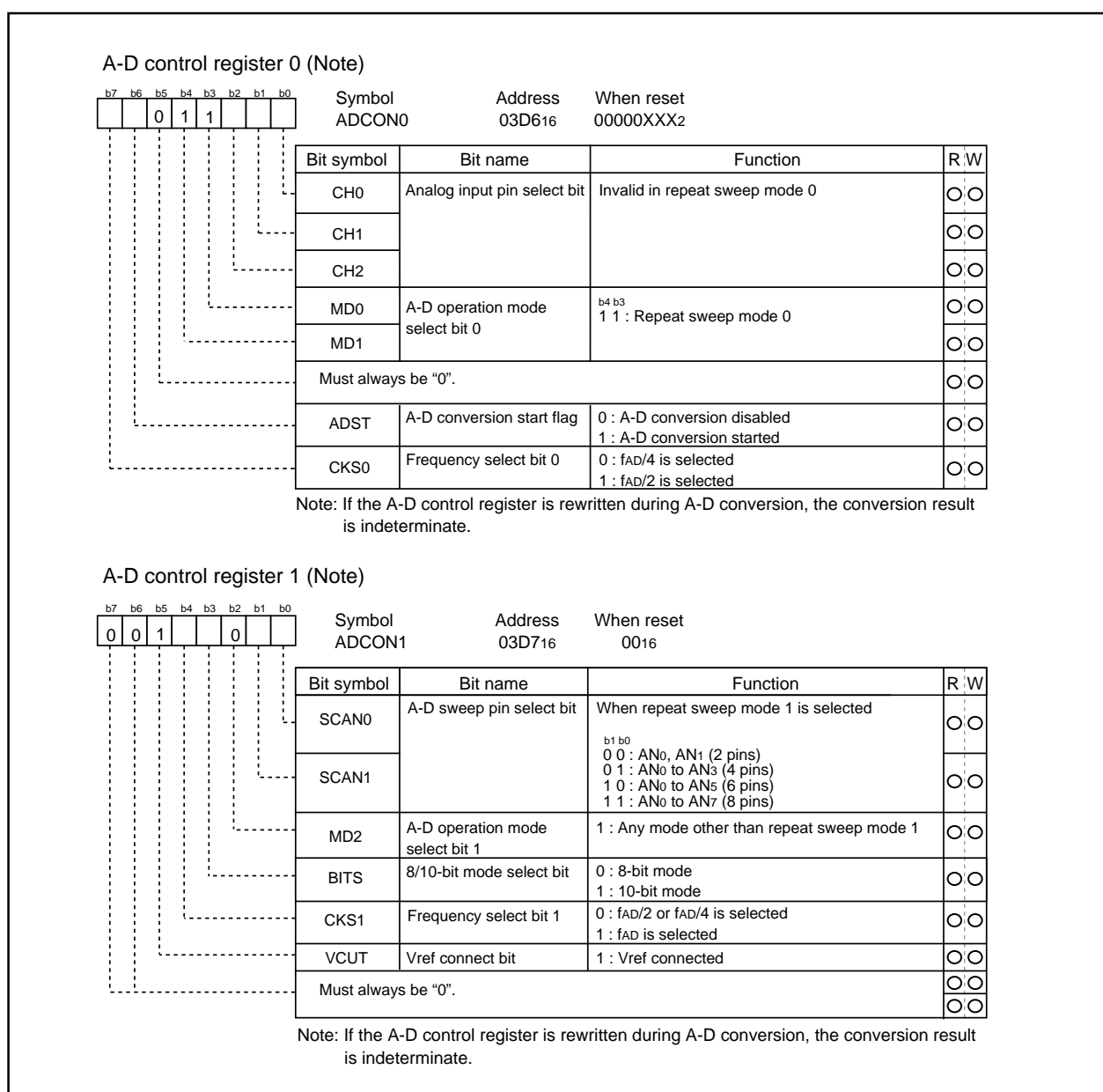
## A-D converter

**(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 34 shows the specifications of repeat sweep mode 0. Figure 107 shows the A-D control register in repeat sweep mode 0.

**Table 34. Repeat sweep mode 0 specifications**

Item	Specification
Function	The pins selected by the A-D sweep pin select bit are used for repeat sweep A-D conversion
Start condition	Writing "1" to A-D conversion start flag
Stop condition	Writing "0" to A-D conversion start flag
Interrupt request generation timing	None generated
Input pin	AN <sub>0</sub> and AN <sub>1</sub> (2 pins), AN <sub>0</sub> to AN <sub>3</sub> (4 pins), AN <sub>0</sub> to AN <sub>5</sub> (6 pins), or AN <sub>0</sub> to AN <sub>7</sub> (8 pins)
Reading of result of A-D converter	Read A-D register corresponding to selected pin (at any time)

**Figure 107. A-D conversion register in repeat sweep mode 0**

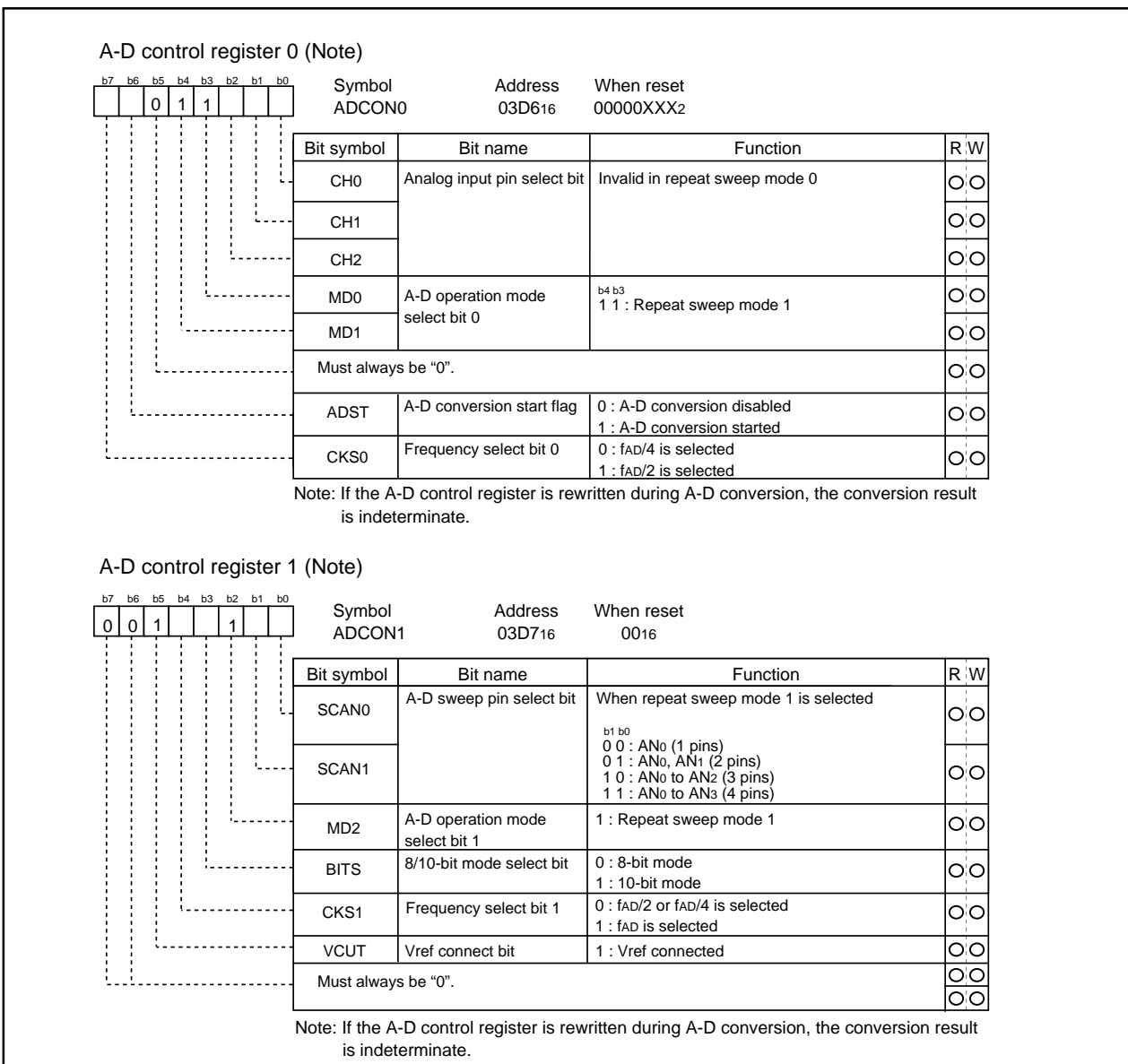
## A-D converter

**(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 35 shows the specifications of repeat sweep mode 1. Figure 108 shows the A-D control register in repeat sweep mode 1.

**Table 35. Repeat sweep mode 1 specifications**

Item	Specification
Function	All pins perform repeat sweep A-D conversion, with emphasis on the pin or pins selected by the A-D sweep pin select bit Example : AN <sub>0</sub> selected -> AN <sub>0</sub> -> AN <sub>1</sub> -> AN <sub>0</sub> -> AN <sub>2</sub> -> AN <sub>0</sub> -> AN <sub>3</sub> , 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	Emphasis on the pin AN <sub>0</sub> (1 pin), AN <sub>0</sub> and AN <sub>1</sub> (2 pins), AN <sub>0</sub> to AN <sub>2</sub> (3 pins), AN <sub>0</sub> to AN <sub>3</sub> (4 pins)
Reading of result of A-D converter	Read A-D register corresponding to selected pin (at any time)

**Figure 108. A-D conversion register in repeat sweep mode 1**

**(a) Sample and hold**

Sample and hold is selected by setting bit 0 of the A-D control register 2 (address 03D4<sub>16</sub>) to "1". When sample and hold is selected, the rate of conversion of each pin increases. As a result, 28  $\phi$  AD cycles are achieved with 8-bit resolution and 33  $\phi$  AD cycles 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.

## D-A converter

**D-A Converter**

This is an 8-bit, R-2R type D-A converter. The microcomputer contains two independent D-A converters of this type. D-A conversion is performed when a value is written to the corresponding D-A register. Bits 0 and 1 (D-A output enable bits) of the D-A control register decide if the result of conversion is to be output. Do not set the target port to output mode if D-A conversion is to be performed.

Output analog voltage (V) is determined by a set value (n : decimal) in the D-A register.

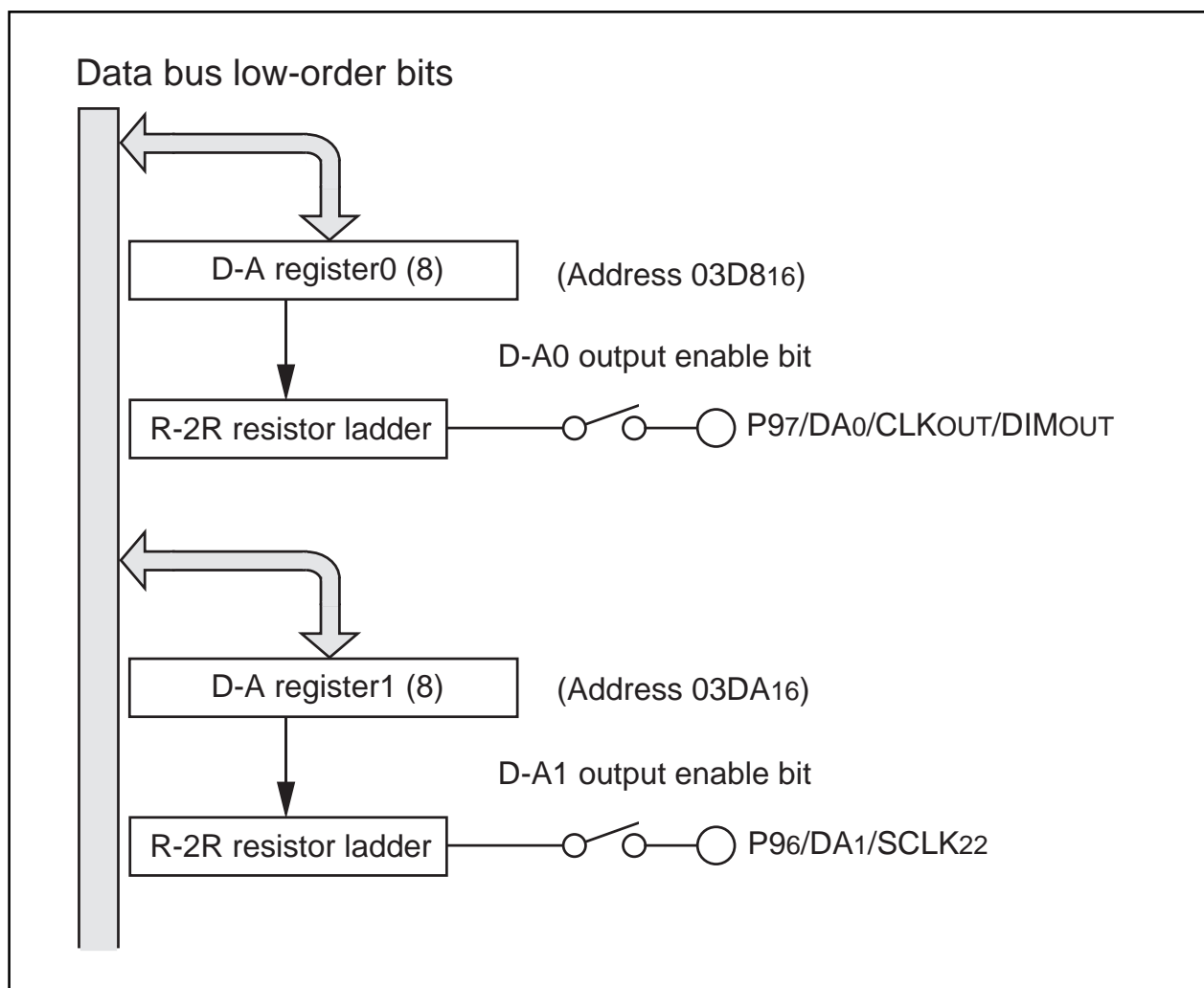
$$V = V_{REF} \times n / 256 \quad (n = 0 \text{ to } 255)$$

$V_{REF}$  : reference voltage

Table 36 lists the performance of the D-A converter. Figure 109 shows the block diagram of the D-A converter. Figure 110 shows the D-A control register. Figure 111 shows the D-A converter equivalent circuit.

**Table 36. Performance of D-A converter**

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



**Figure 109. Block diagram of D-A converter**

D-A converter

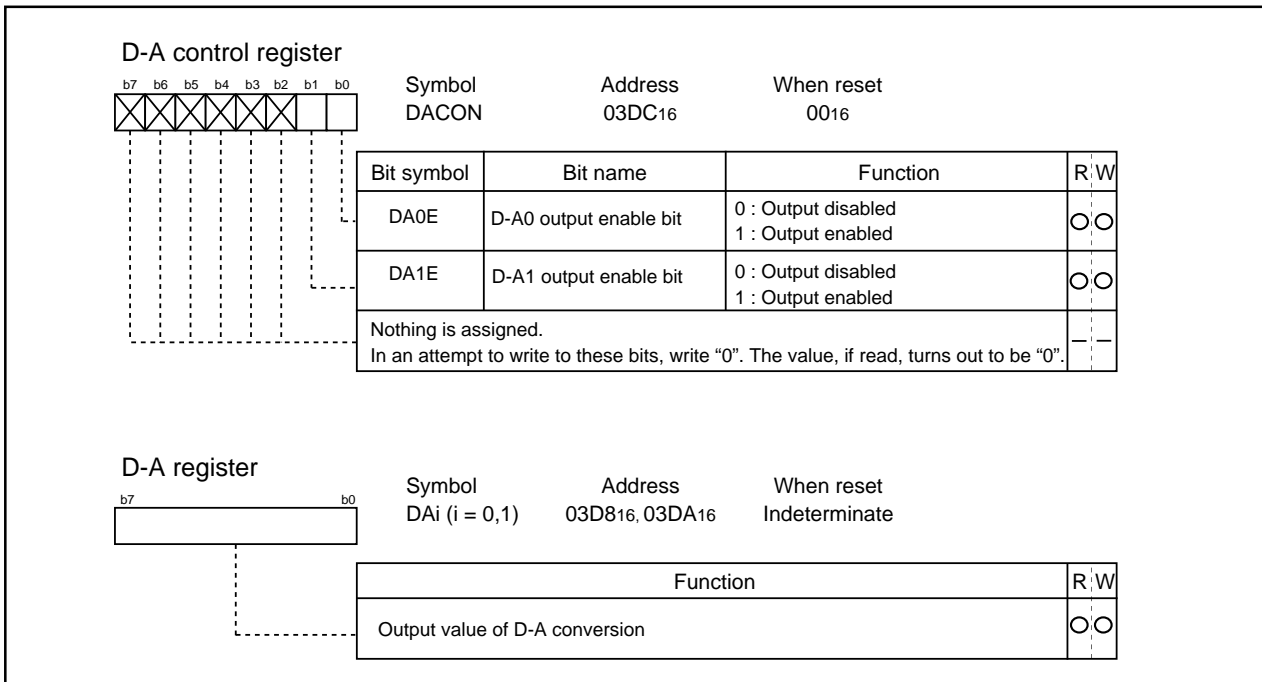


Figure 110. D-A control register

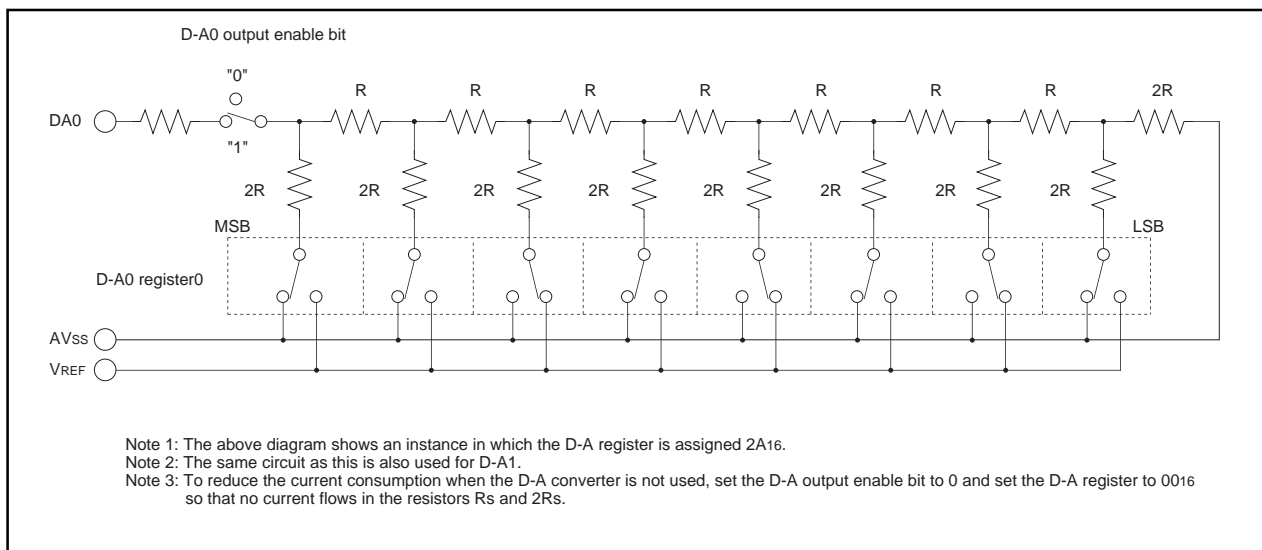


Figure 111. D-A converter equivalent circuit

CRC Calculation Circuit

**CRC Calculation Circuit**

The Cyclic Redundancy Check (CRC) calculation circuit detects an error in data blocks. The microcomputer uses a generator polynomial of CRC\_CCITT ( $X^{16} + X^{12} + X^5 + 1$ ) to generate CRC code.

The CRC code is a 16-bit code generated for a block of a given data length in multiples of 8 bits. The CRC code is set in a CRC data register each time one byte of data is transferred to a CRC input register after writing an initial value into the CRC data register. Generation of CRC code for one byte of data is completed in two machine cycles.

Figure 112 shows the block diagram of the CRC circuit. Figure 113 shows the CRC-related registers. Figure 114 shows the calculation example using the CRC calculation circuit

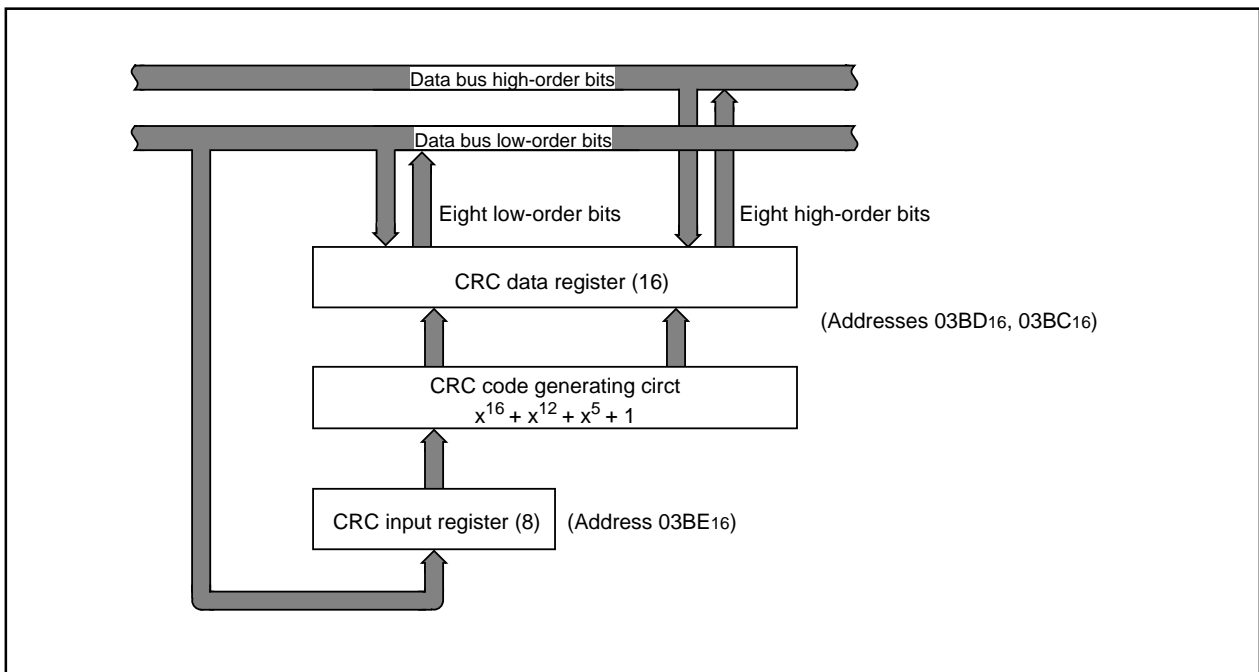


Figure 112. Block diagram of CRC circuit

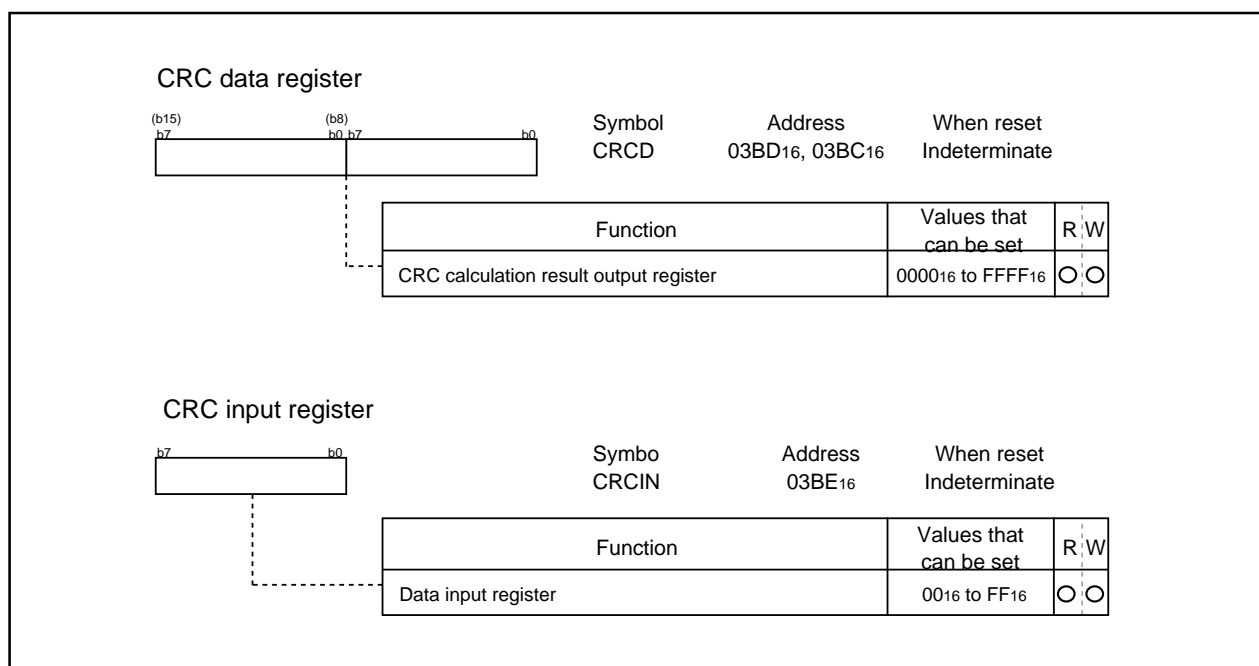


Figure 113. CRC-related registers

CRC Calculation Circuit

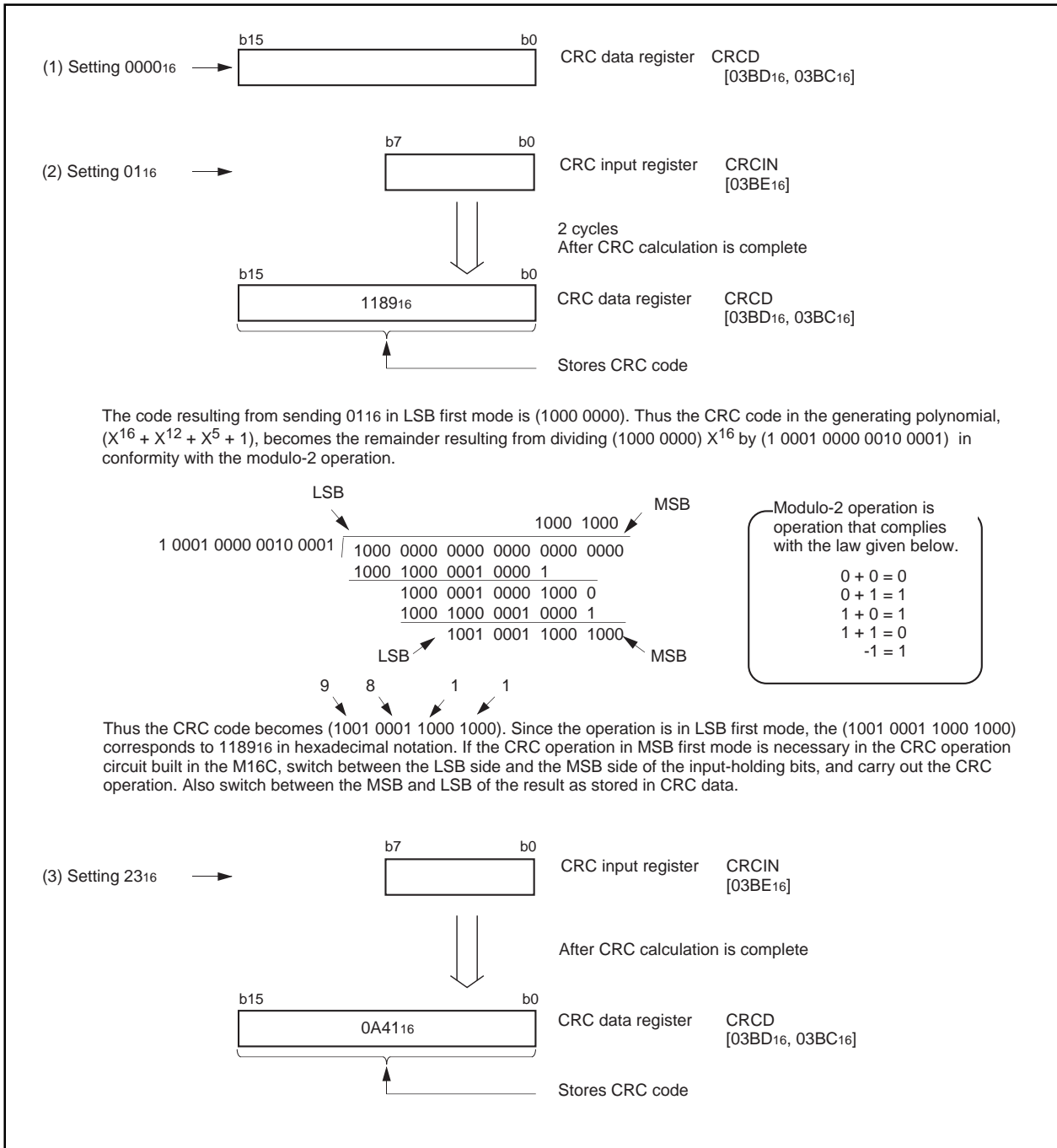


Figure 114. Calculation example using the CRC calculation circuit

## Programmable I/O Ports

There are 48 programmable I/O ports: P3, P4 and P7 to P10. 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.

P3 and P40 to P43 are high-breakdown-voltage, P-channel open drain outputs, and have no built-in pull-down resistance.

Figures 115, 116 show the programmable I/O ports.

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

To use the pins as the inputs for the built-in peripheral devices, set the direction register of each pin to input mode. When the pins are used as the outputs for the built-in peripheral devices (other than the D-A converter), they function as outputs regardless of the contents of the direction registers. 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 117 shows the direction registers.

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

### (2) Port registers

Figure 118 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 119 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.

Note: P3, P40 to P43 have no built-in pull-up resistance, because of these pin's are high-breakdown-voltage, P-channel open drain outputs.

## Exclusive High-breakdown-voltage Output Ports

There are 40 exclusive output Ports: P0 to P2, P5 and P6.

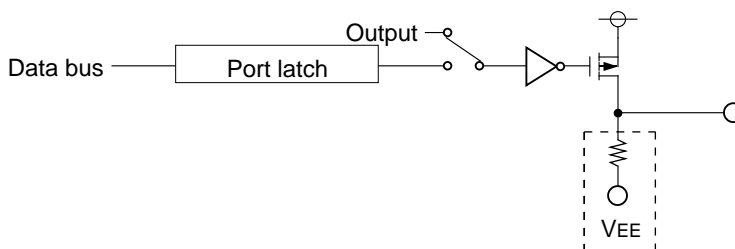
All ports have structure of high-breakdown-voltage P-channel open drain output. Exclusive output ports except P2 have built-in pull-down resistance.

Figure UA-1 shows the configuration of the exclusive high-breakdown-voltage output ports.

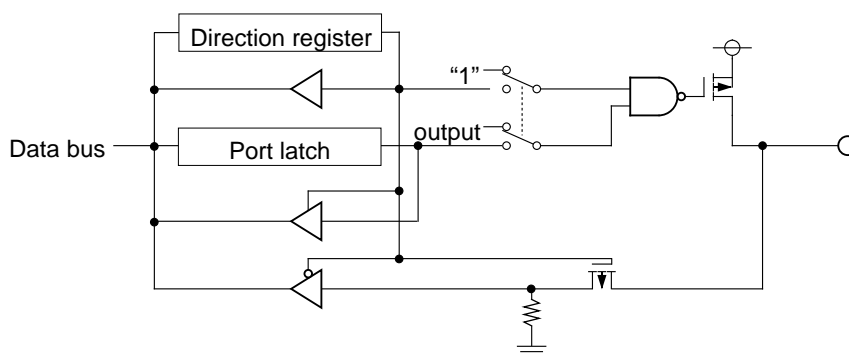


Programmable I/O Ports

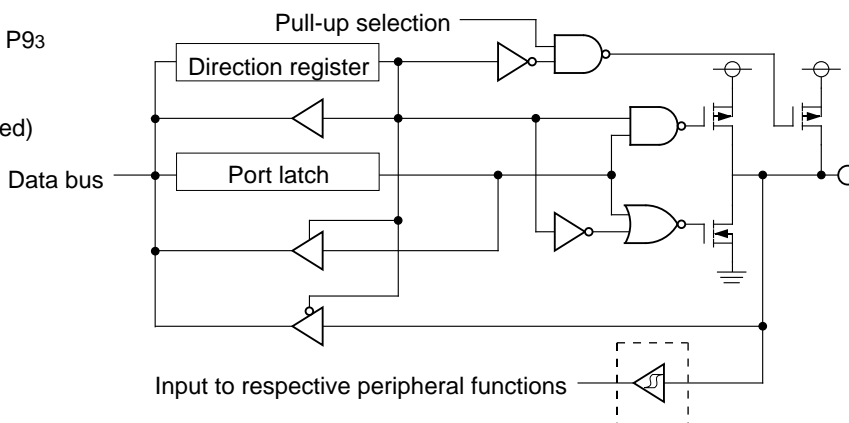
P0<sub>0</sub> to P0<sub>7</sub>, P1<sub>0</sub> to P1<sub>7</sub>,  
 P5<sub>0</sub> to P5<sub>7</sub>, P6<sub>0</sub> to P6<sub>7</sub>,  
 (inside dotted-line included)  
 P2<sub>0</sub> to P2<sub>7</sub>  
 (inside dotted-line not included)



P3<sub>0</sub> to P3<sub>7</sub>, P4<sub>0</sub> to P4<sub>3</sub>



P7<sub>0</sub> to P7<sub>2</sub>, P8<sub>0</sub> to P8<sub>5</sub>, P8<sub>7</sub>, P9<sub>3</sub>  
 (inside dotted-line included)  
 P8<sub>6</sub>  
 (inside dotted-line not included)



P4<sub>4</sub>, P9<sub>2</sub>, P9<sub>4</sub>

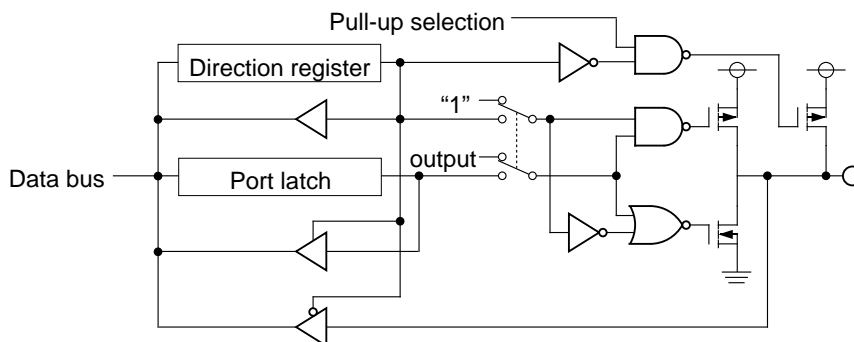


Figure 115. Programmable I/O ports (1)

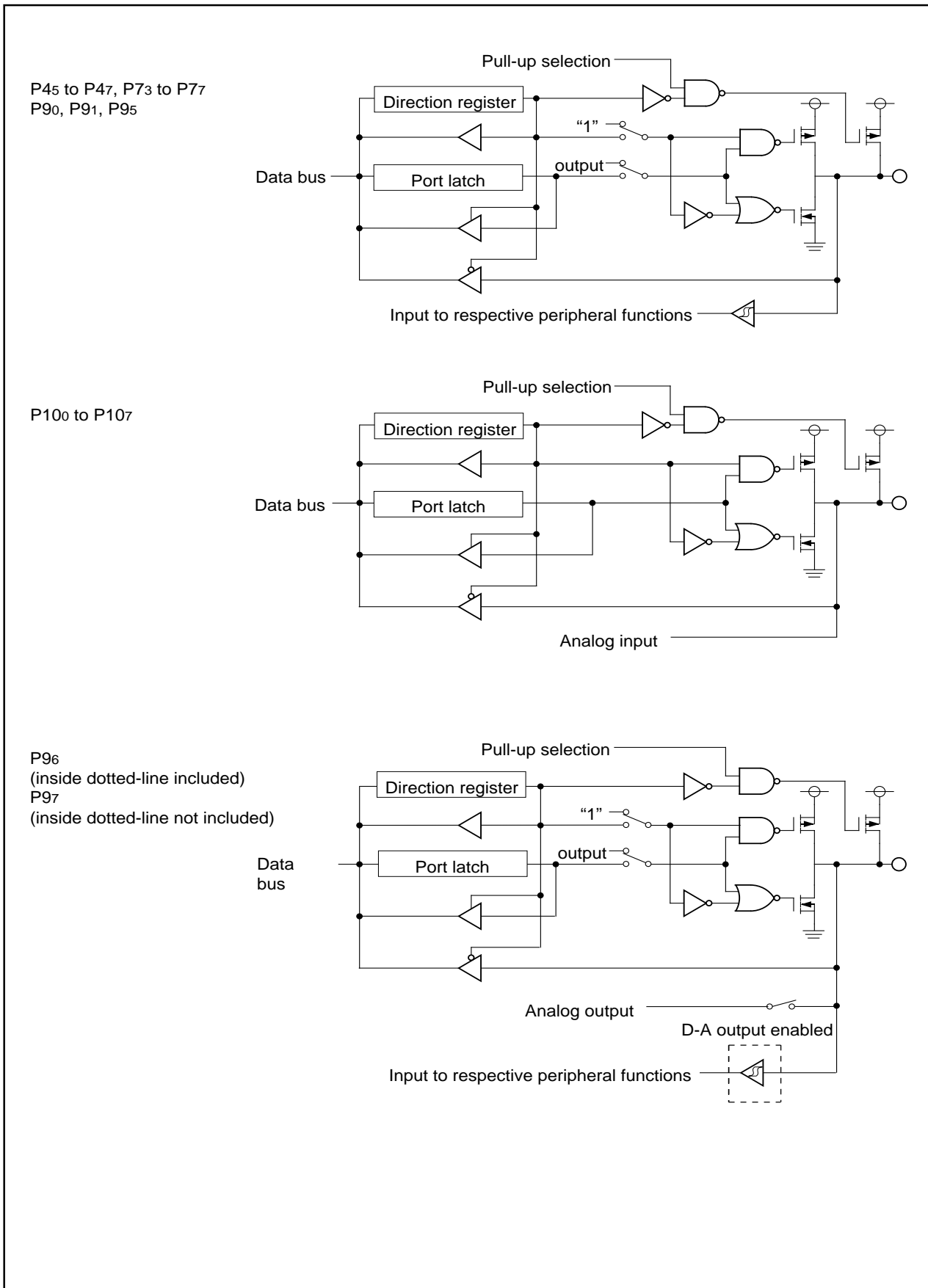


Figure 116. Programmable I/O ports (2)

## Programmable I/O Ports

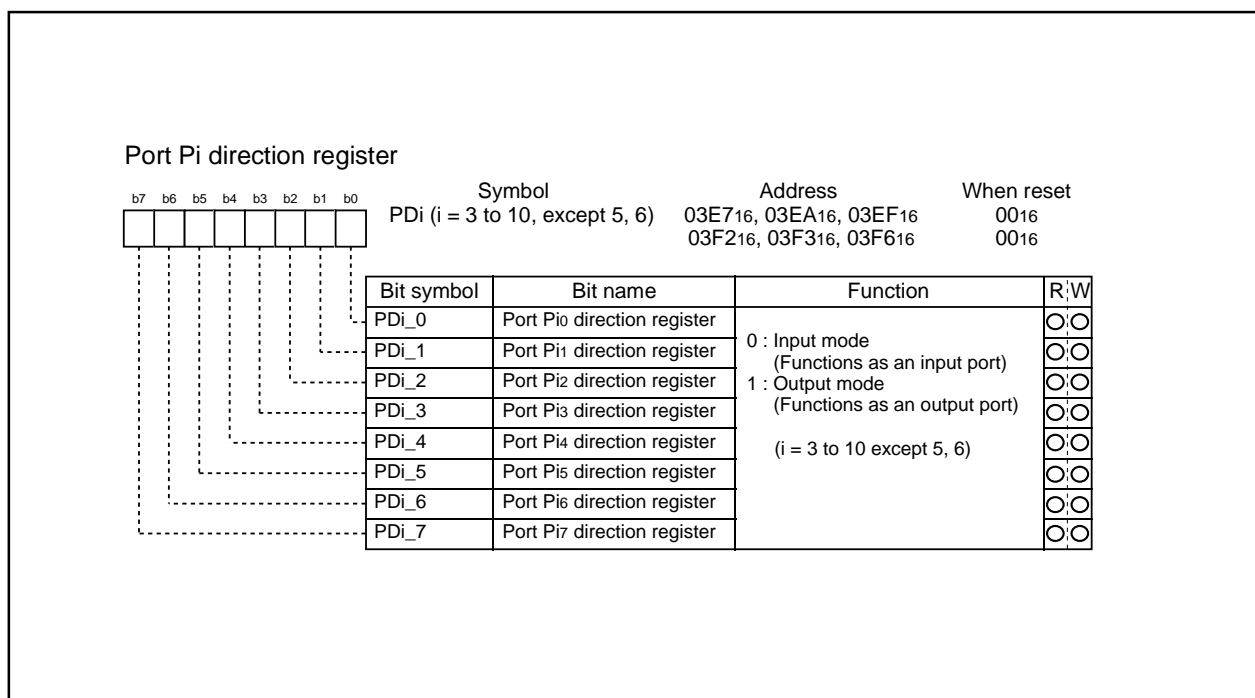


Figure 117. Direction register

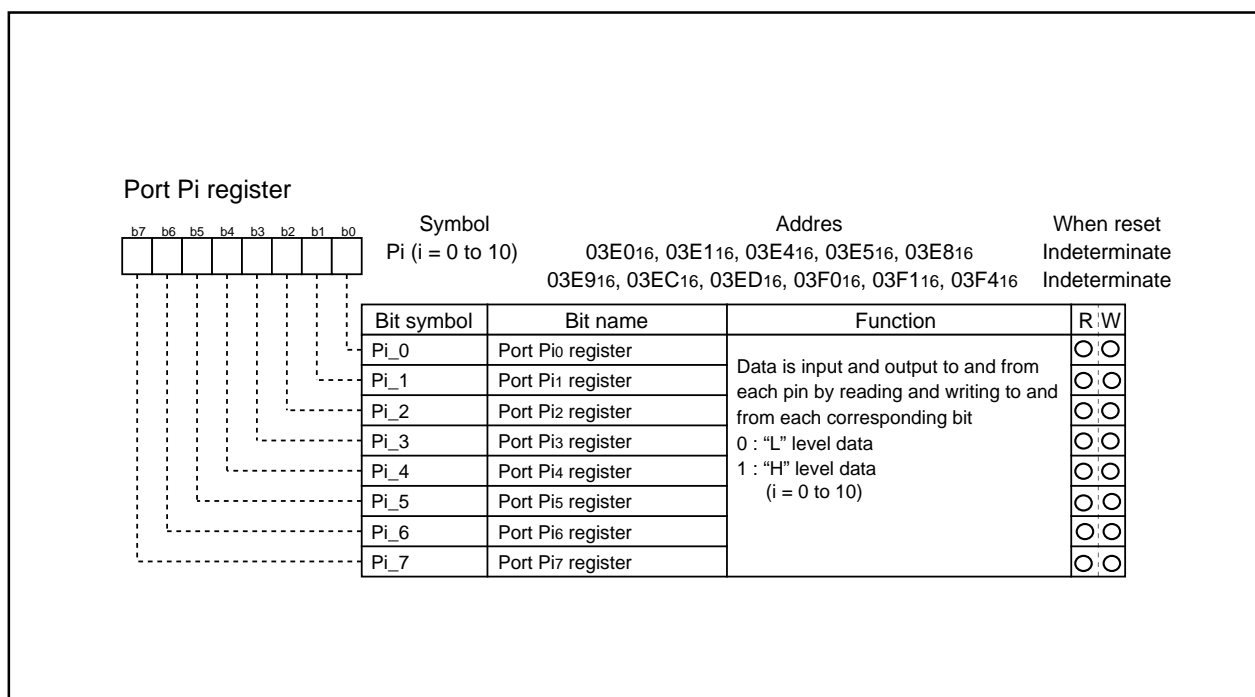


Figure 118. Port register

## Programmable I/O Ports

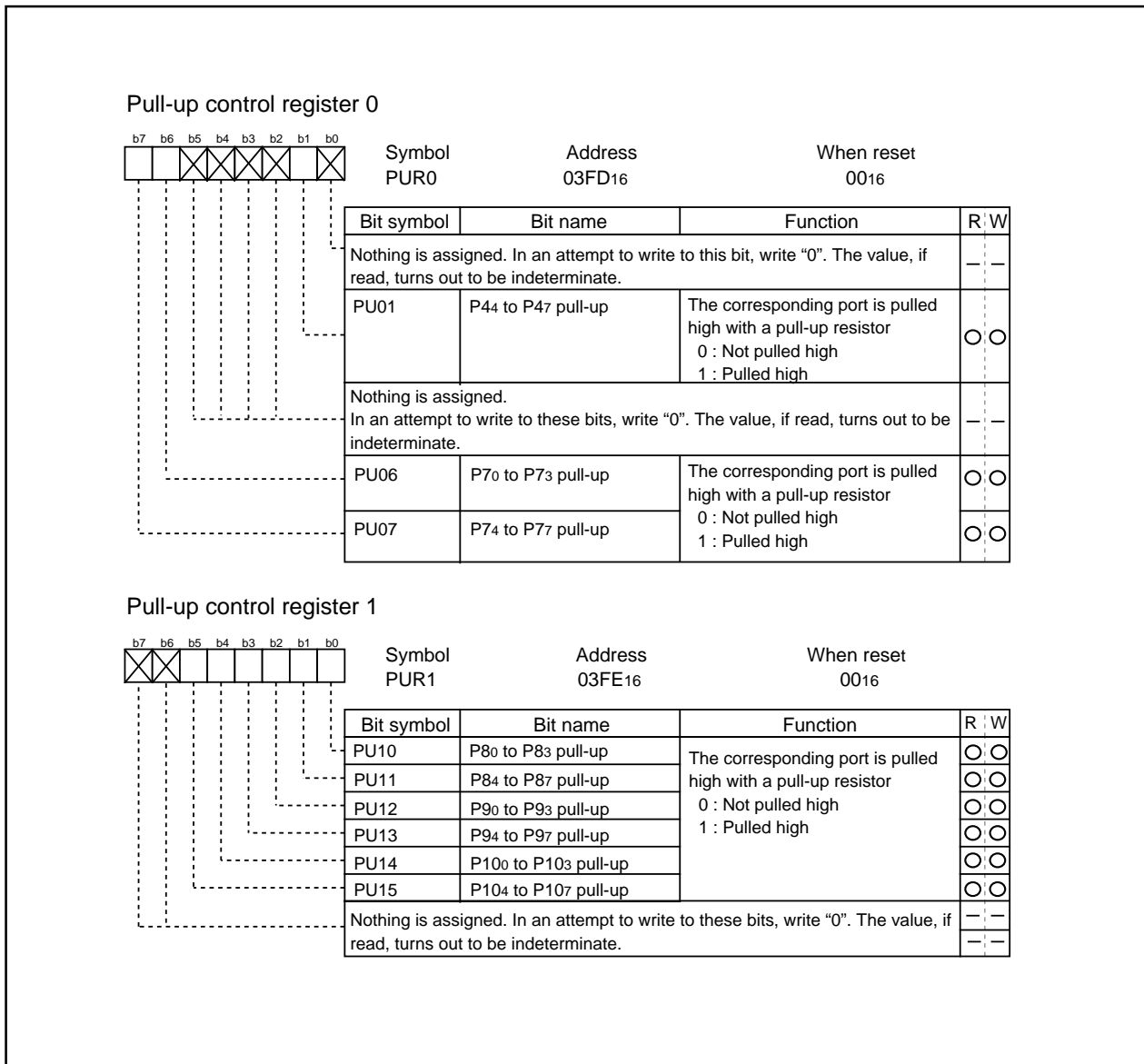


Figure 119. Pull-up control register

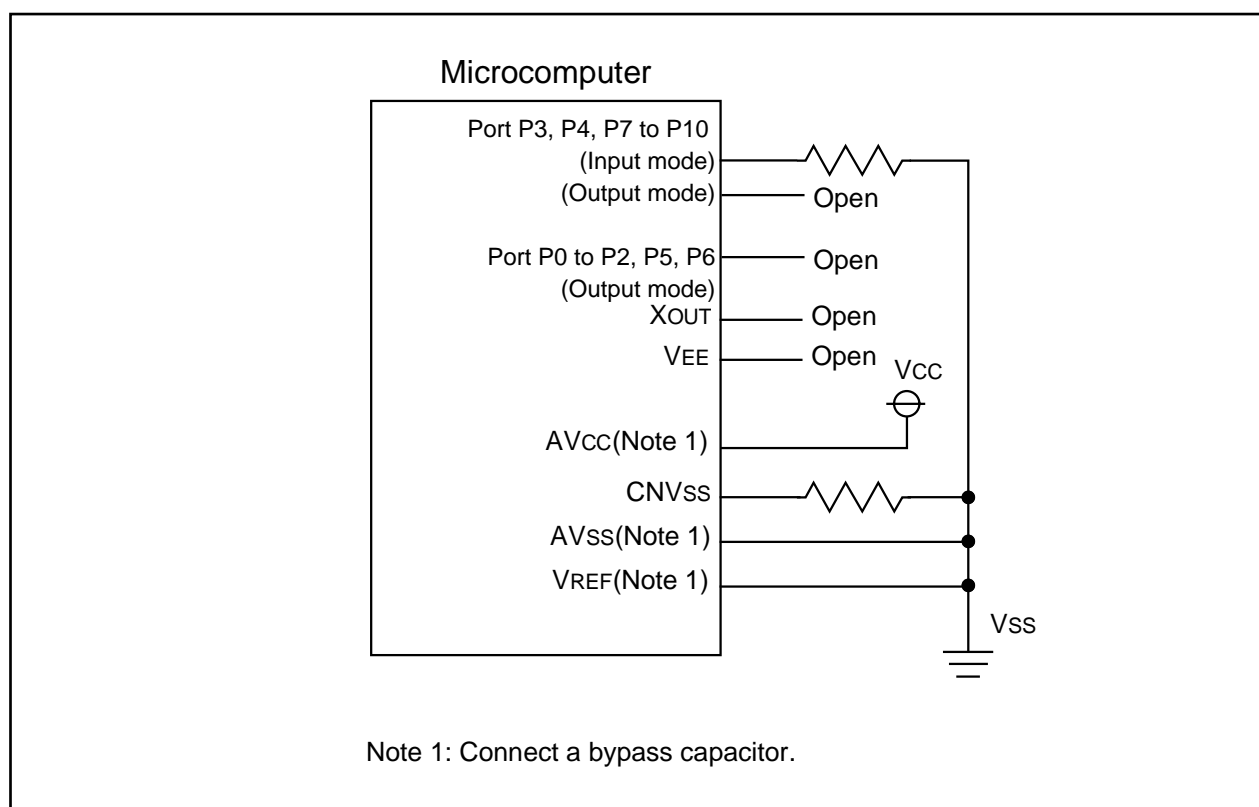
## Programmable I/O Ports

**Table 37. Example connection of unused pins**

Pin name	Connection
Ports P3, P4, P7 to P10	Specify output mode, and leave these pins open; or specify input mode, and connect to VSS via resistor (pull-down)
Ports P0 to P2, P5, P6	Leave these pins open
XOUT (Note 1), VEE	Open
AVCC	Connect to VCC (Note 2)
AVSS, VREF	Connect to VSS (Note 2)
CNVss	Connect to VSS via resistor

Note 1: With external clock input to XIN pin.

Note 2: Connect a bypass capacitor.

**Figure 120. Example connection of unused pins**

## Power Dissipation Calculating Method

(Fixed number depending on microcomputer's standard)

- $V_{OH}$  output fall voltage of high-breakdown port  
2 V (max.); | Current value | = at 18 mA
- Resistor value = 68 k (min.)
- Power dissipation of internal circuit (CPU, ROM, RAM etc.) = 5 V X 38 mA = 190 mW

(Fixed number depending on use condition)

- Apply voltage to VEE pin:  $V_{cc} - 50$  V
- Timing number a; digit number b; segment number c
- Ratio of Toff time corresponding Tdisp time: 1/16
- Turn ON segment number during repeat cycle: d
- All segment number during repeat cycle: e (= a X c)
- Total number of built-in resistor: for digit; f, for segment; g
- Digit pin current value h (mA)
- Segment pin current value i (mA)

(1) Digit pin power dissipation

$$\{h \times b \times (1 - T_{off} / T_{disp}) \times \text{voltage}\} / a$$

(2) Segment pin power dissipation

$$\{i \times d \times (1 - T_{off} / T_{disp}) \times \text{voltage}\} / a$$

(3) Pull-down resistor power dissipation (digit)

$$\{\text{power dissipation per 1 digit} \times (b \times f / b) \times (1 - T_{off} / T_{disp})\} / a$$

(4) Pull-down resistor power dissipation (segment)

$$\{\text{power dissipation per 1 segment} \times (d \times g / c) \times (1 - T_{off} / T_{disp})\} / a$$

(5) Internal circuit power dissipation (CPU, ROM, RAM etc.) = 190 mW

$$(1) + (2) + (3) + (4) + (5) = \underline{\underline{X \text{ mW}}}$$

## Power Dissipation Calculating example 1

Fixed number depending on microcomputer's standard

- $V_{OH}$  output fall voltage of high-breakdown port  
2 V (max.); | Current value | = at 18 mA
- Resistor value 68 k $\Omega$  (min.)
- Power dissipation of internal circuit (CPU, ROM, RAM etc.) = 5 V X 38 mA = 190 mW

Fixed number depending on use condition

- Apply voltage to VEE pin:  $V_{cc} - 50$  V
- Timing number 17; digit number 16; segment number 20
- Ratio of Toff time corresponding Tdisp time: 1/16
- Turn ON segment number during repeat cycle: 31
- All segment number during repeat cycle: 340 (= 17 X 20)
- Total number of built-in resistor: for digit; 16, for segment; 20
- Digit pin current value: 18 (mA)
- Segment pin current value: 3 (mA)

(1) Digit pin power dissipation

$$\{18 \times 16 \times (1 - 1/16) \times 2\} / 17 = 31.77 \text{ mW}$$

(2) Segment pin power dissipation

$$\{3 \times 31 \times (1 - 1/16) \times 2\} / 17 = 10.26 \text{ mW}$$

(3) Pull-down resistor power dissipation (digit)

$$(50 - 2)^2 / 68 \times (16 \times 16/16) \times (1 - 1/16) / 17 = 29.90 \text{ mW}$$

(4) Pull-down resistor power dissipation (segment)

$$(50 - 2)^2 / 68 \times (31 \times 20/20) \times (1 - 1/16) / 17 = 57.93 \text{ mW}$$

(5) Internal circuit power dissipation (CPU, ROM, RAM etc.) = 190.00 mW

$$(1) + (2) + (3) + (4) + (5) = \underline{\underline{319.86 \text{ mW}}}$$

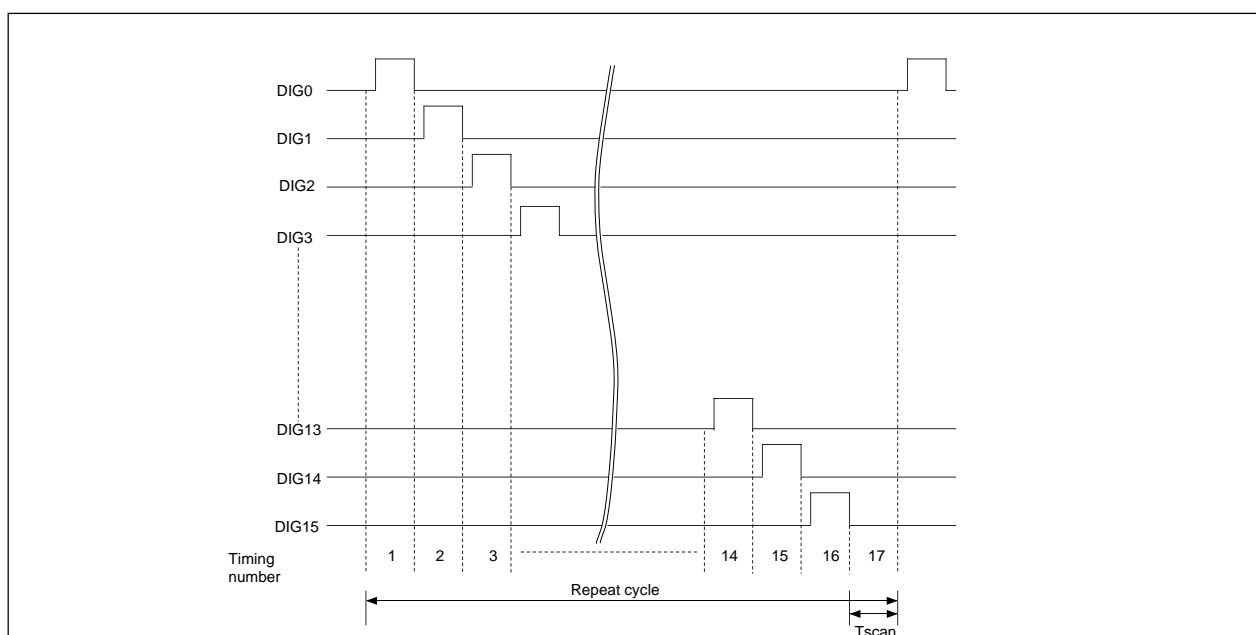


Figure 121. Digit timing waveform (1)

## Pull-down

**Power Dissipation Calculating example 2 (when 2 or more digit is turned ON at same time)**

Fixed number depending on microcomputer's standard

- $V_{OH}$  output fall voltage of high-breakdown port  
2 V (max.); | Current value | = at 18 mA
- Resistor value 68 k $\Omega$  (min.)
- Power dissipation of internal circuit (CPU, ROM, RAM etc.) = 5 V X 38 mA = 190 mW

Fixed number depending on use condition

- Apply voltage to VEE pin:  $V_{cc} - 50$  V
- Timing number 11; digit number 12; segment number 24
- Ratio of Toff time corresponding Tdisp time: 1/16
- Turn ON segment number during repeat cycle: 114
- All segment number during repeat cycle: 264 (= 11 X 24)
- Total number of built-in resistor: for digit; 10, for segment; 22
- Digit pin current value: 18 (mA)
- Segment pin current value: 3 (mA)

(1) Digit pin power dissipation

$$\{18 \times 12 \times (1 - 1/16) \times 2\} / 11 = 36.82 \text{ mW}$$

(2) Segment pin power dissipation

$$\{3 \times 114 \times (1 - 1/16) \times 2\} / 11 = 58.30 \text{ mW}$$

(3) Pull-down resistor power dissipation (digit)

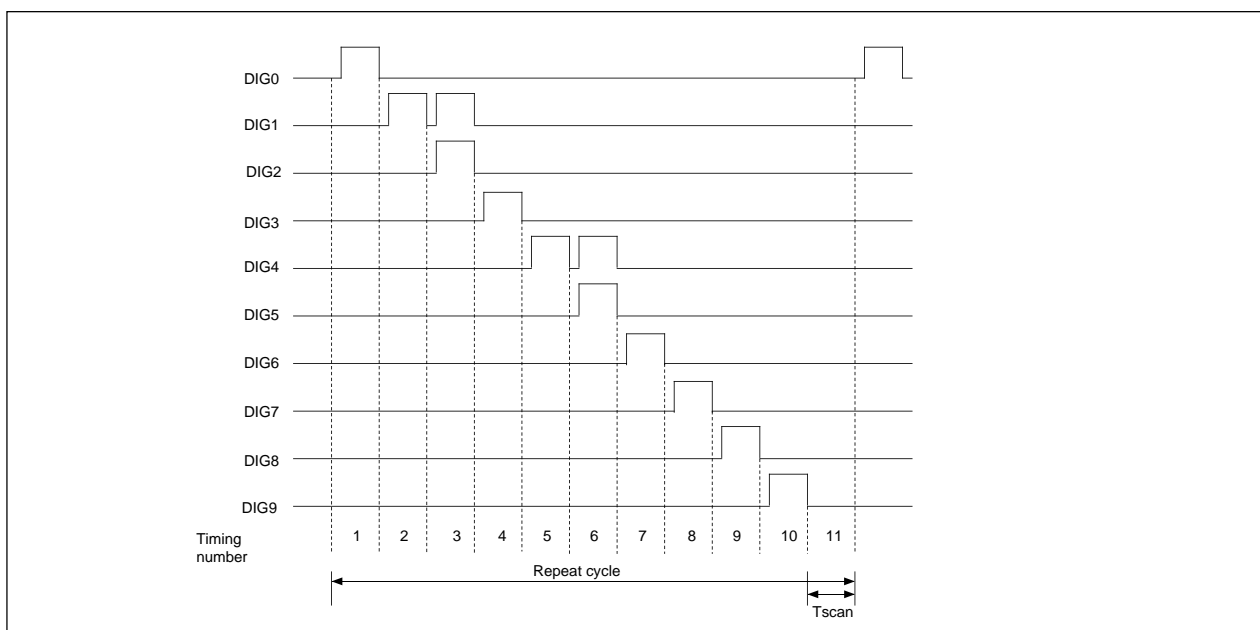
$$(50 - 2)^2 / 68 \times (12 \times 10 / 12) \times (1 - 1/16) / 11 = 28.88 \text{ mW}$$

(4) Pull-down resistor power dissipation (segment)

$$(50 - 2)^2 / 68 \times (114 \times 22 / 24) \times (1 - 1/16) / 11 = 301.77 \text{ mW}$$

(5) Internal circuit power dissipation (CPU, ROM, RAM etc.) = 190.00 mW

$$(1) + (2) + (3) + (4) + (5) = \underline{\underline{615.77 \text{ mW}}} \text{ (There is a limit of use temperature)}$$



**Figure 122. Digit timing waveform (2)**



### Power Dissipation Calculating example 3

(when 2 or more digit is turned ON at same time, and used Toff invalid function)

Fixed number depending on microcomputer's standard

- V<sub>OH</sub> output fall voltage of high-breakdown port 2 V (max.); | Current value | = at 18 mA
- Resistor value 68 kΩ (min.)
- Power dissipation of internal circuit (CPU, ROM, RAM etc.) = 5 V X 38 mA = 190 mW

Fixed number depending on use condition

- Apply voltage to V<sub>EE</sub> pin: V<sub>cc</sub> – 50 V
- Timing number 11; digit number 12; segment number 24
- Ratio of Toff time corresponding T<sub>disp</sub> time: 1/16
- Turn ON segment number during repeat cycle: 114 ( for Toff invalid waveform;50)
- All segment number during repeat cycle: 264 (= 11 X 24)
- Total number of built-in resistor: for digit; 10, for segment; 22
- Digit pin current value: 18 (mA)
- Segment pin current value: 3 (mA)

(1) Digit pin power dissipation

$$[(18 \times 10 \times (1 - 1/16) \times 2) + \{18 \times 2 \times 2\}] / 11 = 37.23 \text{ mW}$$

(2) Segment pin power dissipation

$$[3 \times 64 \times (1 - 1/16) \times 2] + \{3 \times 50 \times 2\} / 11 = 60.00 \text{ mW}$$

(3) Pull-down resistor power dissipation (digit)

$$[(50 - 2)^2 / 68 \times (10 \times 10 / 12) \times (1 - 1 / 16)] + \{(50 - 2)^2 / 68 \times (2 \times 10 / 12)\} / 11 = 29.20 \text{ mW}$$

(4) Pull-down resistor power dissipation (segment)

$$[(50 - 2)^2 / 68 \times (64 \times 22 / 24) \times (1 - 1 / 16)] + \{(50 - 2)^2 / 68 \times (50 \times 22 / 24)\} / 11 = 310.59 \text{ mW}$$

(5) Internal circuit power dissipation (CPU, ROM, RAM etc.) = 190.00 mW

$$(1) + (2) + (3) + (4) + (5) = \underline{627.02 \text{ mW}} \text{ (There is a limit of use temperature)}$$

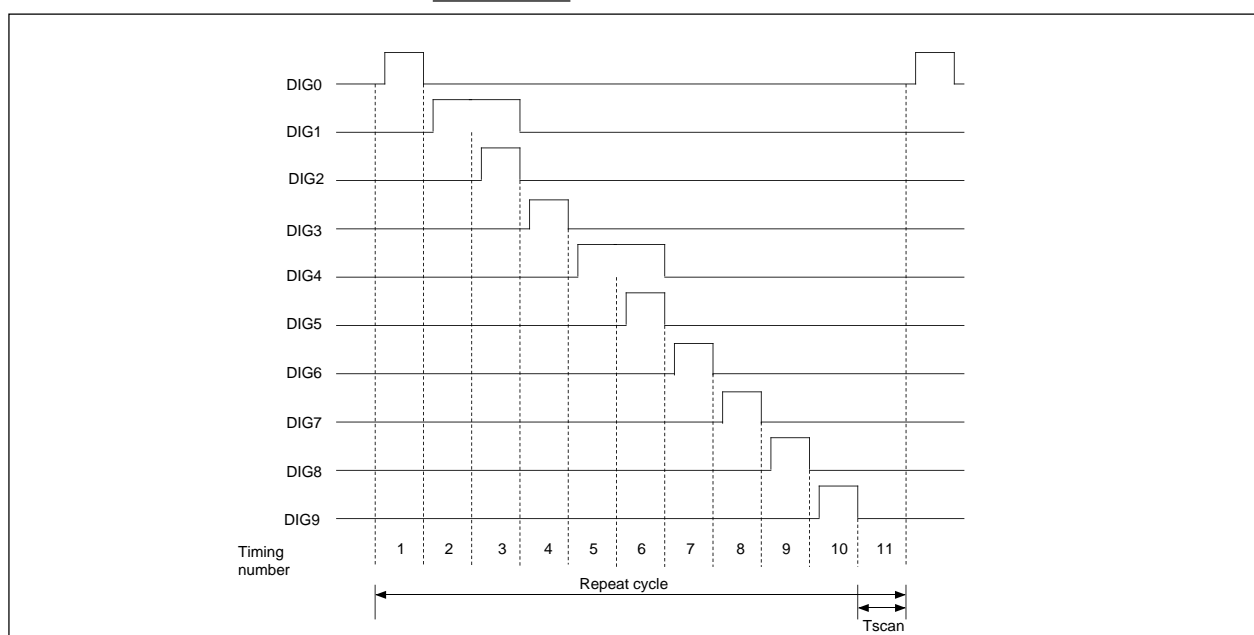


Figure 123. Digit timing waveform (3)

## Electrical characteristics

**Table 38. Absolute maximum ratings**

Symbol	Parameter	Condition	Standard	Unit
V <sub>cc</sub>	Supply voltage		- 0.3 to 6.5	V
AV <sub>cc</sub>	Analog supply voltage		- 0.3 to 6.5	V
V <sub>EE</sub>	Pull-down supply voltage		V <sub>cc</sub> - 50 to V <sub>cc</sub> +0.3V	V
V <sub>I</sub>	Input voltage RESET, CNV <sub>ss</sub> , P44 to P47, P70 to P77, P80 to P87, P90 to P97, P100 to P107, V <sub>REF</sub> , X <sub>IN</sub>		- 0.3 to V <sub>cc</sub> +0.3 (Note)	V
V <sub>I</sub>	Input voltage P30 to P37, P40 to P43		V <sub>cc</sub> - 50 to V <sub>cc</sub> +0.3	V
V <sub>O</sub>	Output voltage P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P43, P50 to P57, P60 to P67		V <sub>cc</sub> - 50 to V <sub>cc</sub> +0.3	V
V <sub>O</sub>	Output voltage P44 to P47, P70 to P77, P80 to P87, P90 to P97, P100 to P107, X <sub>OUT</sub>		-0.3 to V <sub>cc</sub> +0.3	V
P <sub>d</sub>	Power dissipation	T <sub>a</sub> =-20 to 60 °C	750	mW
		T <sub>a</sub> =60 to 85 °C	750-12 X (T <sub>a</sub> -60)	mW
T <sub>opr</sub>	Operating ambient temperature		-20 to 85	°C
T <sub>stg</sub>	Storage temperature		-40 to 150	°C

Note 1: When writing to flash ,only CNV<sub>ss</sub> is -0.3 to 13 (V) .

**Table 39. Recommended operating conditions (referenced to V<sub>cc</sub> = 2.7V to 5.5V at T<sub>a</sub> = - 20 to 85°C unless otherwise specified) (Note)**

Symbol	Parameter	Standard			Unit
		Min	Typ.	Max.	
V <sub>cc</sub>	Supply voltage	2.7(Note1)	5.0	5.5	V
AV <sub>cc</sub>	Analog supply voltage		V <sub>cc</sub>		V
V <sub>ss</sub>	Supply voltage		0		V
AV <sub>ss</sub>	Analog supply voltage		0		V
V <sub>EE</sub>	Pull-down supply voltage	V <sub>cc</sub> -48		V <sub>cc</sub>	V
V <sub>IH</sub>	HIGH input voltage P70 to P77, P80 to P87, P90 to P97, P100 to P107, X <sub>IN</sub> , RESET, CNV <sub>ss</sub>	0.8V <sub>cc</sub>		V <sub>cc</sub>	V
V <sub>IH</sub>	HIGH input voltage P44 to P47	0.50V <sub>cc</sub>		V <sub>cc</sub>	V
V <sub>IH</sub>	HIGH input voltage P30 to P37, P40 to P43	0.52V <sub>cc</sub>		V <sub>cc</sub>	V
V <sub>IL</sub>	LOW input voltage P70 to P77, P80 to P87, P90 to P97, P100 to P107, X <sub>IN</sub> , RESET, CNV <sub>ss</sub>	0		0.2V <sub>cc</sub>	V
V <sub>IL</sub>	LOW input voltage P30 to P37, P40 to P43	0		0.16V <sub>cc</sub>	V
V <sub>IL</sub>	LOW input voltage P44 to P47	0		0.16V <sub>cc</sub>	V

Note: V<sub>cc</sub> = 4.0V to 5.5V in flash memory version.

## Electrical characteristics

**Table 40. Recommended operating conditions (referenced to  $V_{CC} = 2.7V$  to  $5.5V$  at  $T_a = -20$  to  $85^\circ C$  unless otherwise specified) (Note 6)**

Symbol	Parameter	Standard			Unit
		Min	Typ.	Max	
$I_{OH}$ (peak)	HIGH total peak output current (Note 1) P0 <sub>0</sub> to P0 <sub>7</sub> , P5 <sub>0</sub> to P5 <sub>7</sub> , P6 <sub>0</sub> to P6 <sub>7</sub>			-240	mA
$I_{OH}$ (peak)	HIGH total peak output current (Note 1) P1 <sub>0</sub> to P1 <sub>7</sub> , P2 <sub>0</sub> to P2 <sub>7</sub> , P3 <sub>0</sub> to P3 <sub>7</sub> , P4 <sub>0</sub> to P4 <sub>3</sub>			-240	mA
$I_{OH}$ (peak)	HIGH total peak output current (Note 1) P4 <sub>4</sub> to P4 <sub>7</sub> , P7 <sub>0</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>5</sub>			-80	mA
$I_{OH}$ (peak)	HIGH total peak output current (Note 1) P8 <sub>6</sub> , P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub>			-80	mA
$I_{OL}$ (peak)	LOW total peak output current (Note 1) P4 <sub>4</sub> to P4 <sub>7</sub> , P7 <sub>0</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>5</sub>			80	mA
$I_{OL}$ (peak)	LOW total peak output current (Note 1) P8 <sub>6</sub> , P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub>			80	mA
$I_{OH}$ (avg)	HIGH total average output current (Note 1) P0 <sub>0</sub> to P0 <sub>7</sub> , P5 <sub>0</sub> to P5 <sub>7</sub> , P6 <sub>0</sub> to P6 <sub>7</sub>			-120	mA
$I_{OH}$ (avg)	HIGH total average output current (Note 1) P1 <sub>0</sub> to P1 <sub>7</sub> , P2 <sub>0</sub> to P2 <sub>7</sub> , P3 <sub>0</sub> to P3 <sub>7</sub> , P4 <sub>0</sub> to P4 <sub>3</sub>			-120	mA
$I_{OH}$ (avg)	HIGH total average output current (Note 1) P4 <sub>4</sub> to P4 <sub>7</sub> , P7 <sub>0</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>5</sub>			-40	mA
$I_{OH}$ (avg)	HIGH total average output current (Note 1) P8 <sub>6</sub> , P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub>			-40	mA
$I_{OL}$ (avg)	LOW total average output current (Note 1) P4 <sub>4</sub> to P4 <sub>7</sub> , P7 <sub>0</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>5</sub>			40	mA
$I_{OL}$ (avg)	LOW total average output current (Note 1) P8 <sub>6</sub> , P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub>			40	mA
$I_{OH}$ (peak)	HIGH peak output current (Note 2) P0 <sub>0</sub> to P0 <sub>7</sub> , P1 <sub>0</sub> to P1 <sub>7</sub> , P2 <sub>0</sub> to P2 <sub>7</sub> , P3 <sub>0</sub> to P3 <sub>7</sub> , P4 <sub>0</sub> to P4 <sub>3</sub> , P5 <sub>0</sub> to P5 <sub>7</sub> , P6 <sub>0</sub> to P6 <sub>7</sub>			-40	mA
$I_{OH}$ (peak)	HIGH peak output current (Note 2) P4 <sub>4</sub> to P4 <sub>7</sub> , P7 <sub>0</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub>			-10	mA
$I_{OL}$ (peak)	LOW peak output current (Note 2) P4 <sub>4</sub> to P4 <sub>7</sub> , P7 <sub>0</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub>			10	mA
$I_{OH}$ (avg)	HIGH average output current (Note 3) P0 <sub>0</sub> to P0 <sub>7</sub> , P1 <sub>0</sub> to P1 <sub>7</sub> , P2 <sub>0</sub> to P2 <sub>7</sub> , P3 <sub>0</sub> to P3 <sub>7</sub> , P4 <sub>0</sub> to P4 <sub>3</sub> , P5 <sub>0</sub> to P5 <sub>7</sub> , P6 <sub>0</sub> to P6 <sub>7</sub>			-18	mA
$I_{OH}$ (avg)	HIGH average output current (Note 3) P4 <sub>4</sub> to P4 <sub>7</sub> , P7 <sub>0</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub>			-5	mA
$I_{OL}$ (avg)	LOW average output current (Note 3) P4 <sub>4</sub> to P4 <sub>7</sub> , P7 <sub>0</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub>			5	mA
$f$ (XIN)	Main clock input oscillation frequency (Note 4, 7)	$V_{CC}=4.0V$ to $5.5V$	0	10	MHz
		$V_{CC}=2.7V$ to $4.0V$	0	5 X $V_{CC}-10$	MHz
$f$ (XCIN)	Sub clock oscillation frequency (Note 4, 5)		32.768	50	kHz

Note 1: The total output current is the sum of all the currents through the applicable ports. The total average value measured over 100ms. The total peak current is the peak of all the currents.

Note 2: The peak output current is the peak current flowing in each port.

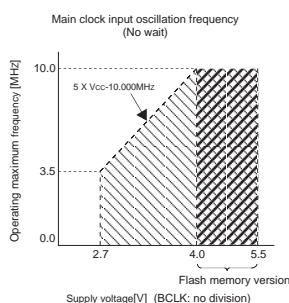
Note 3: The average output current in an average value measured over 100ms.

Note 4: When the oscillating frequency has a duty cycle of 50 %.

Note 5: When using the microcomputer in low-speed mode, set the sub-clock input oscillation frequency on condition that  $f(XCIN) < f(XIN) / 3$ .

Note 6:  $V_{CC}=4.0V$  to  $5.5V$  in flash memory version.

Note 7: Relationship between main clock oscillation frequency and supply voltage.



Electrical characteristics ( $V_{CC}=5V$ ) $V_{CC}=5V$ 

**Table 41. Electrical characteristics (referenced to  $V_{CC} = 5V$ ,  $V_{SS} = 0V$  at  $T_a = 25^\circ C$ ,  
 $f(X_{IN}) = 10MHz$  unless otherwise specified)**

Symbol	Parameter		Measuring condition	Standard			Unit
				Min.	Typ.	Max.	
V <sub>OH</sub>	HIGH output voltage	P0 <sub>0</sub> to P0 <sub>7</sub> , P1 <sub>0</sub> to P1 <sub>7</sub> , P2 <sub>0</sub> to P2 <sub>7</sub> , P3 <sub>0</sub> to P3 <sub>7</sub> , P4 <sub>0</sub> to P4 <sub>3</sub> , P5 <sub>0</sub> to P5 <sub>7</sub> , P6 <sub>0</sub> to P6 <sub>7</sub>	I <sub>OH</sub> = - 18mA	3.5			V
			I <sub>OH</sub> = - 5mA	4.5			
V <sub>OH</sub>	HIGH output voltage	P4 <sub>4</sub> to P4 <sub>7</sub> , P7 <sub>0</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub>	I <sub>OH</sub> = - 5mA	3.0			V
V <sub>OH</sub>	HIGH output voltage	X <sub>OUT</sub>	HIGH POWER	I <sub>OH</sub> = - 1mA	3.0		V
			LOW POWER	I <sub>OH</sub> = - 0.5mA	3.0		
V <sub>OL</sub>	LOW output voltage	P4 <sub>4</sub> to P4 <sub>7</sub> , P7 <sub>0</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub>	I <sub>OL</sub> =5mA			2.0	V
V <sub>OL</sub>	LOW output voltage	X <sub>OUT</sub>	HIGH POWER	I <sub>OL</sub> =1mA		2.0	V
			LOW POWER	I <sub>OL</sub> =0.5mA		2.0	
V <sub>T+</sub> -V <sub>T-</sub>	Hysteresis	TA0 <sub>IN</sub> to TA4 <sub>IN</sub> , TB0 <sub>IN</sub> to TB2 <sub>IN</sub> , INT <sub>0</sub> to INT <sub>5</sub> , CTS <sub>0</sub> , CTS <sub>1</sub> , CLK <sub>0</sub> , CLK <sub>1</sub> , SRDY2 <sub>IN</sub> , SBSY2 <sub>IN</sub> , SIN <sub>2</sub> , SCLK2 <sub>1</sub> , SCLK2 <sub>2</sub> , RxD <sub>0</sub> , RxD <sub>1</sub>		0.2		0.8	V
V <sub>T+</sub> -V <sub>T-</sub>	Hysteresis	RESET		0.2		1.8	V
I <sub>IH</sub>	HIGH input current	P4 <sub>4</sub> to P4 <sub>7</sub> , P7 <sub>0</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub> , X <sub>IN</sub> , RESET, CNV <sub>SS</sub>	V <sub>I</sub> =5V			5.0	μA
I <sub>IH</sub>		P3 <sub>0</sub> to P3 <sub>7</sub> , P4 <sub>0</sub> to P4 <sub>3</sub> (Note 1)	V <sub>I</sub> =5V			5.0	μA
I <sub>IL</sub>	LOW input current	P4 <sub>4</sub> to P4 <sub>7</sub> , P7 <sub>0</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub> , X <sub>IN</sub> , RESET, CNV <sub>SS</sub>	V <sub>I</sub> =0V			- 5.0	μA
I <sub>IL</sub>		P3 <sub>0</sub> to P3 <sub>7</sub> , P4 <sub>0</sub> to P4 <sub>3</sub> (Note1)	V <sub>I</sub> =0V			- 5.0	μA
R <sub>PULLUP</sub>	Pull-up resistance	P4 <sub>4</sub> to P4 <sub>7</sub> , P7 <sub>0</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub>	V <sub>I</sub> =0V	30.0	50.0	167.0	kΩ
R <sub>PULLD</sub>	Pull-down resistance	P0 <sub>0</sub> to P0 <sub>7</sub> , P1 <sub>0</sub> to P1 <sub>7</sub> , P5 <sub>0</sub> to P5 <sub>7</sub> , P6 <sub>0</sub> to P6 <sub>7</sub>	V <sub>EE</sub> =V <sub>CC</sub> - 48V, V <sub>OL</sub> =V <sub>CC</sub> Output transistors "off"	68	80	120	kΩ
I <sub>LEAK</sub>	Output leak current	P0 <sub>0</sub> to P0 <sub>7</sub> , P1 <sub>0</sub> to P1 <sub>7</sub> , P2 <sub>0</sub> to P2 <sub>7</sub> , P3 <sub>0</sub> to P3 <sub>7</sub> , P4 <sub>0</sub> to P4 <sub>4</sub> , P5 <sub>0</sub> to P5 <sub>7</sub> , P6 <sub>0</sub> to P6 <sub>7</sub>	V <sub>EE</sub> =V <sub>CC</sub> - 48V, V <sub>OL</sub> =V <sub>CC</sub> - 48V Output transistors "off"			- 10	μA
R <sub>I<sub>XIN</sub></sub>	Feedback resistance X <sub>IN</sub>				1.0		MΩ
R <sub>I<sub>XCIN</sub></sub>	Feedback resistance X <sub>CIN</sub>				6.0		MΩ
V <sub>RAM</sub>	RAM retention voltage		When clock is stopped	2.0			V
I <sub>CC</sub>	Power supply current (Note 3)	The output pins are open and other pins are V <sub>SS</sub>	f(X <sub>IN</sub> )=10MHz Square wave, no division		19.0	38.0	mA
			f(X <sub>IN</sub> )=10MHz Square wave, 8 division		4.2		
			f(X <sub>CIN</sub> )=32kHz Square wave (Note2)		90.0		μA
			f(X <sub>CIN</sub> )=32kHz When a WAIT instruction is executed (Note2)		4.0		μA
			T <sub>a</sub> =25 °C when clock is stopped			1.0	μA
			T <sub>a</sub> =85 °C when clock is stopped			20.0	

Note 1: Except when reading ports P3, P4<sub>0</sub> to P4<sub>3</sub>.

Note 2: Fixed X<sub>CIN</sub>-X<sub>COUT</sub> drive capacity select bit to "HIGH" and X<sub>IN</sub> pin to "H" level.

Note 3: This contains an electric current to flow into AV<sub>CC</sub> pin.

Electrical characteristics ( $V_{CC}=5V$ ) $V_{CC}=5V$ 

**Table 42. A-D conversion characteristics (referenced to  $V_{CC} = AV_{CC} = V_{REF} = 5V$ ,  $V_{SS} = AV_{SS} = 0V$   
at  $T_a = 25^\circ C$ ,  $f(X_{IN}) = 10MHz$  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	$V_{REF} = V_{CC} = 5V$			$\pm 3$	LSB
		Sample & hold function available(10bit)	$V_{REF} = V_{CC} = 5V$ AN0 to AN7 input			$\pm 3$	LSB
		Sample & hold function available(8bit)	$V_{REF} = V_{CC} = 5V$			$\pm 2$	LSB
RLADDER	Ladder resistance		$V_{REF} = V_{CC}$	10		40	$k\Omega$
tCONV	Conversion time (10bit)			3.3			$\mu s$
tCONV	Conversion time (8bit)			2.8			$\mu s$
tsAMP	Sampling time			0.3			$\mu s$
VREF	Reference voltage			2		$V_{CC}$	V
VIA	Analog input voltage			0		$V_{REF}$	V

**Table 43. D-A conversion characteristics (referenced to  $V_{CC} = 5V$ ,  $V_{SS} = AV_{SS} = 0V$ ,  $V_{REF} = 5V$   
at  $T_a = 25^\circ C$ ,  $f(X_{IN}) = 10MHz$  unless otherwise specified)**

Symbol	Parameter		Measuring condition	Standard			Unit
				Min.	Typ.	Max.	
-	Resolution					8	Bits
-	Absolute accuracy					1.0	%
t <sub>su</sub>	Setup time					3	$\mu s$
R <sub>o</sub>	Output resistance			4	10	20	$k\Omega$
I <sub>VREF</sub>	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 "0016".

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

Also, when the Vref is unconnected at the A-D control register, I<sub>VREF</sub> is sent.

Timing ( $V_{CC}=5V$ ) $V_{CC}=5V$ Timing requirements (referenced to  $V_{CC} = 5V$ ,  $V_{SS} = 0V$  at  $T_a = 25^\circ C$  unless otherwise specified)

Table 44. 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

Switching characteristics (referenced to  $V_{CC} = 5V$ ,  $V_{SS} = 0V$  at  $T_a = 25^\circ C$  unless otherwise specified)

Table 45. High-breakdown voltage p-channel open-drain output port

Symbol	Parameter	Measuring condition	Standard			Unit
			Min.	Typ.	Max.	
$t_r(\text{Pch-strg})$	P-channel high-breakdown voltage output rising time (Note 1)	$C_L=100\text{pF}$ $V_{EE}=V_{CC} - 43V$		55		ns
$t_r(\text{Pch-weak})$	P-channel high-breakdown voltage output rising time (Note 2)	$C_L=100\text{pF}$ $V_{EE}=V_{CC} - 43V$		1.8		$\mu\text{s}$

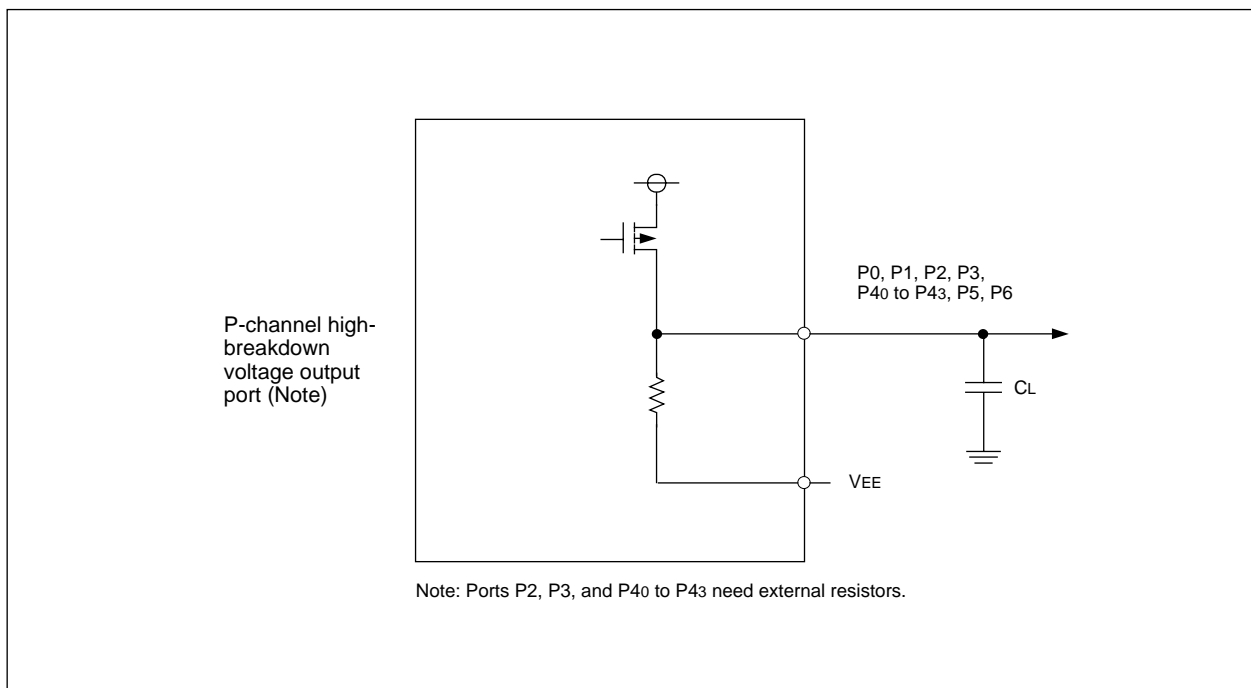
Note 1: When bit 7 of the FLDC mode register (address 0350<sub>16</sub>) is at "0".Note 2: When bit 7 of the FLDC mode register (address 0350<sub>16</sub>) is at "1".

Figure 124. Circuit for measuring output switching characteristics

Timing ( $V_{CC}=5V$ ) $V_{CC}=5V$ Timing requirements (referenced to  $V_{CC} = 5V$ ,  $V_{SS} = 0V$  at  $T_a = 25^\circ C$  unless otherwise specified)

Table 46. Timer A input (counter input in event counter mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(TA)}$	TAiIN input cycle time	100		ns
$t_w(TAH)$	TAiIN input HIGH pulse width	40		ns
$t_w(TAL)$	TAiIN input LOW pulse width	40		ns

Table 47. Timer A input (gating input in timer mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(TA)}$	TAiIN input cycle time	400		ns
$t_w(TAH)$	TAiIN input HIGH pulse width	200		ns
$t_w(TAL)$	TAiIN input LOW pulse width	200		ns

Table 48. Timer A input (external trigger input in one-shot timer mode)

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

Table 49. Timer A input (external trigger input in pulse width modulation mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_w(TAH)$	TAiIN input HIGH pulse width	100		ns
$t_w(TAL)$	TAiIN input LOW pulse width	100		ns

Table 50. Timer A input (up/down input in event counter mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(UP)}$	TAiOUT input cycle time	2000		ns
$t_w(UPH)$	TAiOUT input HIGH pulse width	1000		ns
$t_w(UPL)$	TAiOUT input LOW pulse width	1000		ns
$t_{su(UP-TIN)}$	TAiOUT input setup time	400		ns
$t_h(TIN-UP)$	TAiOUT input hold time	400		ns

Timing ( $V_{CC}=5V$ ) $V_{CC}=5V$ Timing requirements (referenced to  $V_{CC} = 5V$ ,  $V_{SS} = 0V$  at  $T_a = 25^\circ C$  unless otherwise specified)

Table 51. Timer B input (counter input in event counter mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(TB)}$	TBiN input cycle time (counted on one edge)	100		ns
$t_{w(TBH)}$	TBiN input HIGH pulse width (counted on one edge)	40		ns
$t_{w(TBL)}$	TBiN input LOW pulse width (counted on one edge)	40		ns
$t_{c(TB)}$	TBiN input cycle time (counted on both edges)	200		ns
$t_{w(TBH)}$	TBiN input HIGH pulse width (counted on both edges)	80		ns
$t_{w(TBL)}$	TBiN input LOW pulse width (counted on both edges)	80		ns

Table 52. Timer B input (pulse period measurement mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(TB)}$	TBiN input cycle time	400		ns
$t_{w(TBH)}$	TBiN input HIGH pulse width	200		ns
$t_{w(TBL)}$	TBiN input LOW pulse width	200		ns

Table 53. Timer B input (pulse width measurement mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(TB)}$	TBiN input cycle time	400		ns
$t_{w(TBH)}$	TBiN input HIGH pulse width	200		ns
$t_{w(TBL)}$	TBiN input LOW pulse width	200		ns

Table 54. 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 55. External interrupt  $\overline{INTi}$  inputs

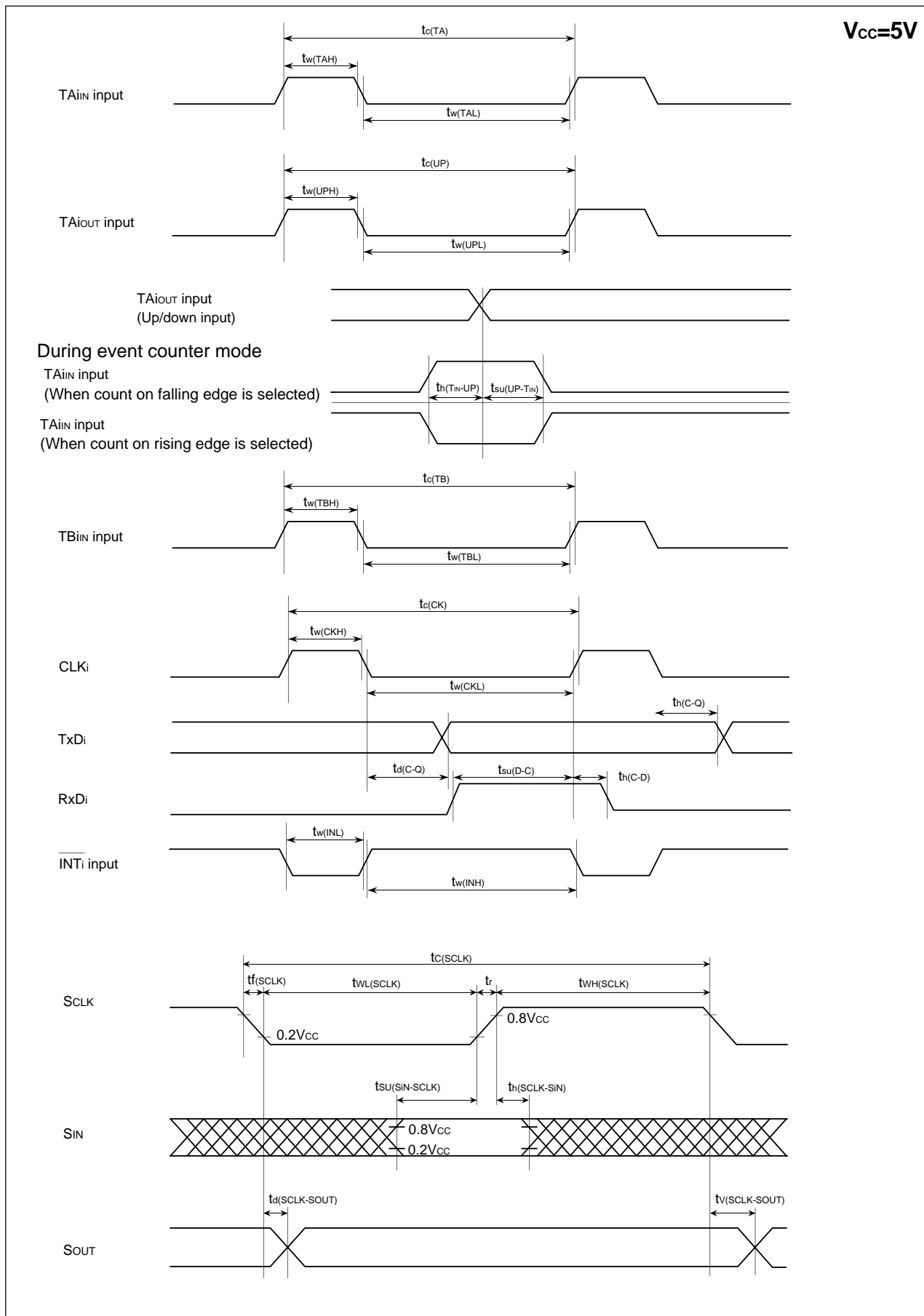
Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{w(INH)}$	$\overline{INTi}$ input HIGH pulse width	250		ns
$t_{w(INL)}$	$\overline{INTi}$ input LOW pulse width	250		ns

Table 56. Automatic transfer serial I/O

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(SCLK)}$	Serial I/O clock input cycle time	0.95		$\mu s$
$t_{wH(SCLK)}$	Serial I/O clock input HIGH pulse width	400		ns
$t_{wL(SCLK)}$	Serial I/O clock input LOW pulse width	400		ns
$t_{su(SCLK-SIN)}$	Serial I/O input setup time	200		ns
$t_{h(SCLK-SIN)}$	Serial I/O input hold time	200		ns



Timing ( $V_{CC}=5V$ )



$V_{CC}=3V$ 

**Table 57. Electrical characteristics (referenced to  $V_{CC} = 3V$ ,  $V_{SS} = 0V$  at  $T_a = 25^\circ C$ ,  
 $f(X_{IN}) = 5MHz$  unless otherwise specified)**

Symbol	Parameter		Measuring condition	Standard			Unit
				Min.	Typ.	Max.	
V <sub>OH</sub>	HIGH output voltage	P0 <sub>0</sub> to P0 <sub>7</sub> , P1 <sub>0</sub> to P1 <sub>7</sub> , P2 <sub>0</sub> to P2 <sub>7</sub> , P3 <sub>0</sub> to P3 <sub>7</sub> , P4 <sub>0</sub> to P4 <sub>3</sub> , P5 <sub>0</sub> to P5 <sub>7</sub> , P6 <sub>0</sub> to P6 <sub>7</sub>	I <sub>OH</sub> = - 18mA	1.5			V
			I <sub>OH</sub> = - 5mA	2.5			
V <sub>OH</sub>	HIGH output voltage	P4 <sub>4</sub> to P4 <sub>7</sub> , P7 <sub>0</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub>	I <sub>OH</sub> = - 1mA	2.5			V
V <sub>OH</sub>	HIGH output voltage	X <sub>OUT</sub>	HIGH POWER	I <sub>OH</sub> = - 0.1mA	2.5		V
			LOW POWER	I <sub>OH</sub> = - 50μA	2.5		
V <sub>OL</sub>	LOW output voltage	P4 <sub>4</sub> to P4 <sub>7</sub> , P7 <sub>0</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub>	I <sub>OL</sub> =1mA			0.5	V
V <sub>OL</sub>	LOW output voltage	X <sub>OUT</sub>	HIGH POWER	I <sub>OL</sub> =0.1mA			0.5
			LOW POWER	I <sub>OL</sub> =50μA			0.5
V <sub>T+</sub> -V <sub>T-</sub>	Hysteresis	TA0 <sub>IN</sub> to TA4 <sub>IN</sub> , TB0 <sub>IN</sub> to TB2 <sub>IN</sub> , INT0 to INT5, CTS0, CTS1, CLK0, CLK1, SRDY2 <sub>IN</sub> , SBSY2 <sub>IN</sub> , SIN2, SCLK21, SCLK22, RTS0, RTS1		0.2		0.8	V
V <sub>T+</sub> -V <sub>T-</sub>	Hysteresis	RESET		0.2		1.8	V
I <sub>IH</sub>	HIGH input current	P4 <sub>4</sub> to P4 <sub>7</sub> , P7 <sub>0</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub> , X <sub>IN</sub> , RESET, CNV <sub>SS</sub>	V <sub>i</sub> =3V			4.0	μA
I <sub>IH</sub>		P3 <sub>0</sub> to P3 <sub>7</sub> , P4 <sub>0</sub> to P4 <sub>3</sub> (Note 1)	V <sub>i</sub> =3V			4.0	μA
I <sub>IL</sub>	LOW input current	P4 <sub>4</sub> to P4 <sub>7</sub> , P7 <sub>0</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub> , X <sub>IN</sub> , RESET, CNV <sub>SS</sub>	V <sub>i</sub> =0V			- 4.0	μA
I <sub>IL</sub>		P3 <sub>0</sub> to P3 <sub>7</sub> , P4 <sub>0</sub> to P4 <sub>3</sub> (Note 1)	V <sub>i</sub> =0V			- 4.0	μA
R <sub>PULLUP</sub>	Pull-up resistance	P4 <sub>4</sub> to P4 <sub>7</sub> , P7 <sub>0</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub>	V <sub>i</sub> =0V	66.0	120.0	500.0	kΩ
R <sub>PULLD</sub>	Pull-down resistance	P0 <sub>0</sub> to P0 <sub>7</sub> , P1 <sub>0</sub> to P1 <sub>7</sub> , P5 <sub>0</sub> to P5 <sub>7</sub> , P6 <sub>0</sub> to P6 <sub>7</sub>	V <sub>EE</sub> =V <sub>CC</sub> - 48V, V <sub>OL</sub> =V <sub>CC</sub> Output transistors "off"	68	80	120	kΩ
I <sub>LEAK</sub>	Output leak current	P0 <sub>0</sub> to P0 <sub>7</sub> , P1 <sub>0</sub> to P1 <sub>7</sub> , P2 <sub>0</sub> to P2 <sub>7</sub> , P3 <sub>0</sub> to P3 <sub>7</sub> , P4 <sub>0</sub> to P4 <sub>4</sub> , P5 <sub>0</sub> to P5 <sub>7</sub> , P6 <sub>0</sub> to P6 <sub>7</sub>	V <sub>EE</sub> =V <sub>CC</sub> - 48V, V <sub>OL</sub> =V <sub>CC</sub> - 48V Output transistors "off"			- 10	μA
R <sub>XIN</sub>	Feedback resistance	X <sub>IN</sub>			3.0		MΩ
R <sub>XCIN</sub>	Feedback resistance	X <sub>CIN</sub>			10.0		MΩ
V <sub>RAM</sub>	RAM retention voltage		When clock is stopped	2.0			V
I <sub>CC</sub>	Power supply current (Note 3)	The output pins are open and other pins are V <sub>SS</sub>	f(X <sub>IN</sub> )=5MHz Square wave, no division	6.0	15.0		mA
			f(X <sub>IN</sub> )=5MHz Square wave, 8 division	1.6			mA
			f(X <sub>CIN</sub> )=32kHz Square wave	50.0			μA
			f(X <sub>CIN</sub> )=32kHz When a WAIT instruction is executed. Oscillation capacity High (Note2)	2.8			μA
			f(X <sub>CIN</sub> )=32kHz When a WAIT instruction is executed. Oscillation capacity Low (Note2)	0.9			μA
			T <sub>a</sub> =25 °C when clock is stopped			1.0	μA
			T <sub>a</sub> =85 °C when clock is stopped			20.0	

Note 1: Except when reading ports P3, P40 to P43.

Note 2: With one timer operated using fc32.

Note 3: This contains an electric current to flow into AV<sub>CC</sub> pin.

$V_{CC}=3V$ 

**Table 58. A-D conversion characteristics (referenced to  $V_{CC} = AV_{CC} = V_{REF} = 3V$ ,  $V_{SS} = AV_{SS} = 0V$   
at  $T_a = 25^\circ C$ ,  $f(X_{IN}) = 5MHz$  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 (8 bit) $V_{REF} = V_{CC} = 3V$ , $\phi_{AD} = f(X_{IN})/2$			$\pm 2$	LSB
RLADDER	Ladder resistance	$V_{REF} = V_{CC}$	10		40	k $\Omega$
tCONV	Conversion time (8bit)		14.0			$\mu s$
VREF	Reference voltage		2.7		$V_{CC}$	V
VIA	Analog input voltage		0		$V_{REF}$	V

**Table 59. D-A conversion characteristics (referenced to  $V_{CC} = 3V$ ,  $V_{SS} = AV_{SS} = 0V$ ,  $V_{REF} = 3V$   
at  $T_a = 25^\circ C$ ,  $f(X_{IN}) = 5MHz$  unless otherwise specified)**

Symbol	Parameter	Measuring condition	Standard			Unit
			Min.	Typ.	Max	
-	Resolution				8	Bits
-	Absolute accuracy				1.0	%
t <sub>SU</sub>	Setup time				3	$\mu s$
R <sub>O</sub>	Output resistance		4	10	20	k $\Omega$
I <sub>VREF</sub>	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 "0016".

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

Also, when the Vref is unconnected at the A-D control register, I<sub>VREF</sub> is sent.

Timing( $V_{CC}=3V$ , only mask ROM version) **$V_{CC}=3V$** Timing requirements (referenced to  $V_{CC} = 3V$ ,  $V_{SS} = 0V$  at  $T_a = 25^{\circ}C$  unless otherwise specified)

Table 60. External clock input

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_c$	External clock input cycle time	200		ns
$t_{w(H)}$	External clock input HIGH pulse width	85		ns
$t_{w(L)}$	External clock input LOW pulse width	85		ns
$t_r$	External clock rise time		18	ns
$t_f$	External clock fall time		18	ns

Timing( $V_{CC}=3V$ , only mask ROM version) **$V_{CC}=3V$** **Timing requirements (referenced to  $V_{CC} = 3V$ ,  $V_{SS} = 0V$  at  $T_a = 25^\circ C$  unless otherwise specified)****Table 61. 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 62. 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 63. 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 64. 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 65. 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
$t_{su(UP-TIN)}$	TAiOUT input setup time	600		ns
$t_{h(TIN-UP)}$	TAiOUT input hold time	600		ns

Timing ( $V_{CC}=3V$ , only mask ROM version) $V_{CC}=3V$ Timing requirements (referenced to  $V_{CC} = 3V$ ,  $V_{SS} = 0V$  at  $T_a = 25^\circ C$  unless otherwise specified)

Table 66. Timer B input (counter input in event counter mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(TB)}$	TBiIn input cycle time (counted on one edge)	150		ns
$t_{w(TBH)}$	TBiIn input HIGH pulse width (counted on one edge)	60		ns
$t_{w(TBL)}$	TBiIn input LOW pulse width (counted on one edge)	60		ns
$t_{c(TB)}$	TBiIn input cycle time (counted on both edges)	300		ns
$t_{w(TBH)}$	TBiIn input HIGH pulse width (counted on both edges)	160		ns
$t_{w(TBL)}$	TBiIn input LOW pulse width (counted on both edges)	160		ns

Table 67. Timer B input (pulse period measurement mode)

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

Table 68. Timer B input (pulse width measurement mode)

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

Table 69. 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

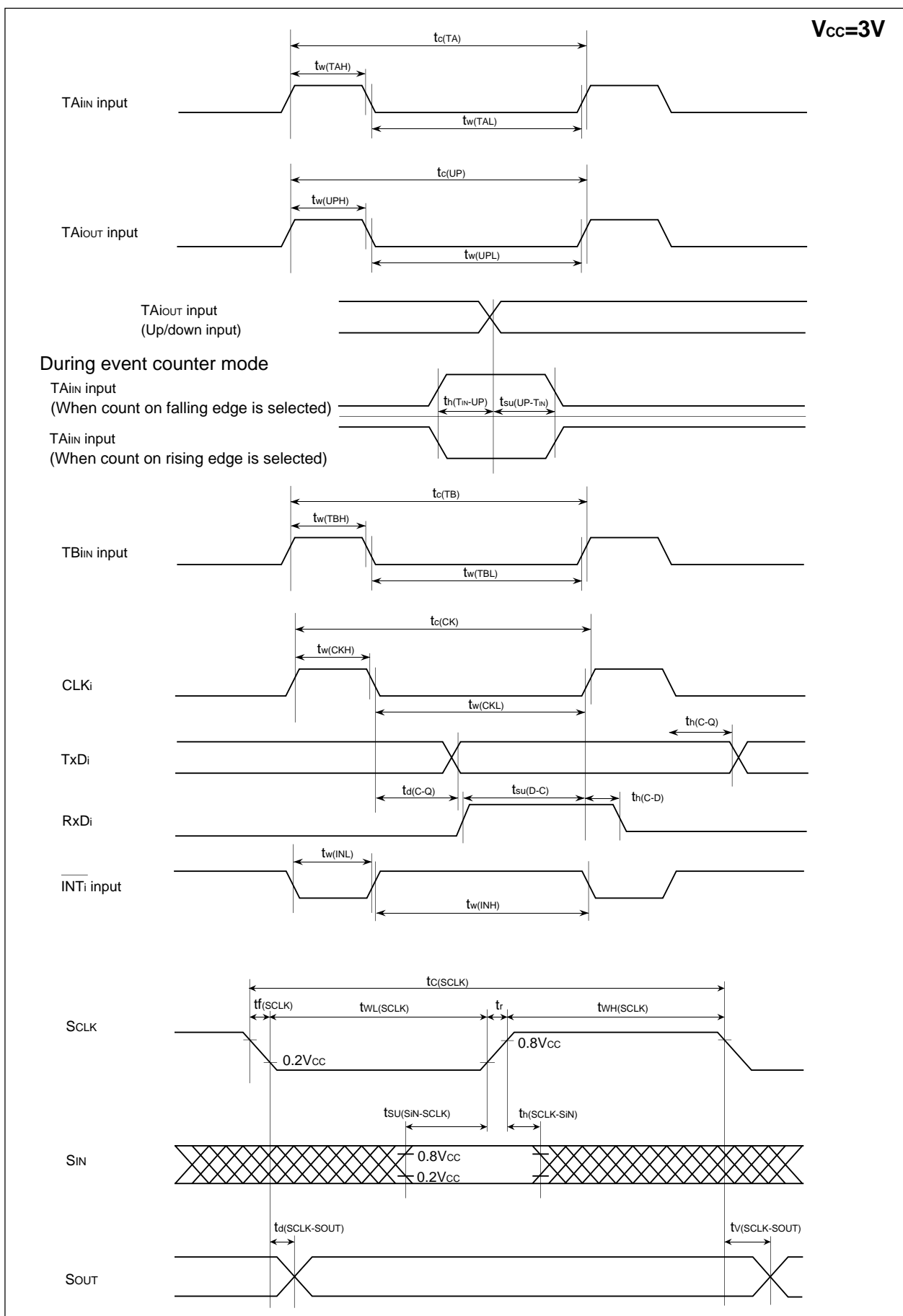
Table 70. External interrupt  $\overline{INT}_i$  inputs

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

Table 71. Automatic transfer serial I/O

Symbol	Parameter	Standard		Unit
		Min.	Max.	
$t_{c(SCLK)}$	Serial I/O clock input cycle time	2.0		$\mu s$
$t_{wH(SCLK)}$	Serial I/O clock input HIGH pulse width	1000		ns
$t_{wL(SCLK)}$	Serial I/O clock input LOW pulse width	1000		ns
$t_{su}(SCLK-SIN)$	Serial I/O input setup time	400		ns
$t_h(SCLK-SIN)$	Serial I/O input hold time	400		ns

Timing ( $V_{CC}=3V$ , only mask ROM version)



## Description

**Outline Performance**

Table 72 shows the outline performance of the M30218 group (flash memory version).

**Table 72. Outline Performance of the M30218 group (flash memory version)**

Item		Performance
Power supply voltage		4.0V to 5.5 V (f(XIN)=10MHz)
Program/erase voltage		VPP=12V ± 5% (f(XIN)=10MHz)
		VCC=5V ± 10% (f(XIN)=10MHz)
Flash memory operation mode		Three modes (parallel I/O, standard serial I/O, CPU rewrite)
Erase block division	User ROM area	See Figure 1.AA.3.
	Boot ROM area	One division (3.5 K bytes) (Note)
Program method		In units of byte
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		Standard serial I/O mode is supported.

Note: The boot ROM area contains a standard serial I/O mode control program which is stored in it when shipped from the factory. This area can be erased and programmed in only parallel I/O mode.



## Description

## Flash Memory

The M30218 group (flash memory version) contains the NOR type of flash memory that requires a high-voltage  $V_{PP}$  power supply for program/erase operations, in addition to the  $V_{CC}$  power supply for device operation. 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.

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

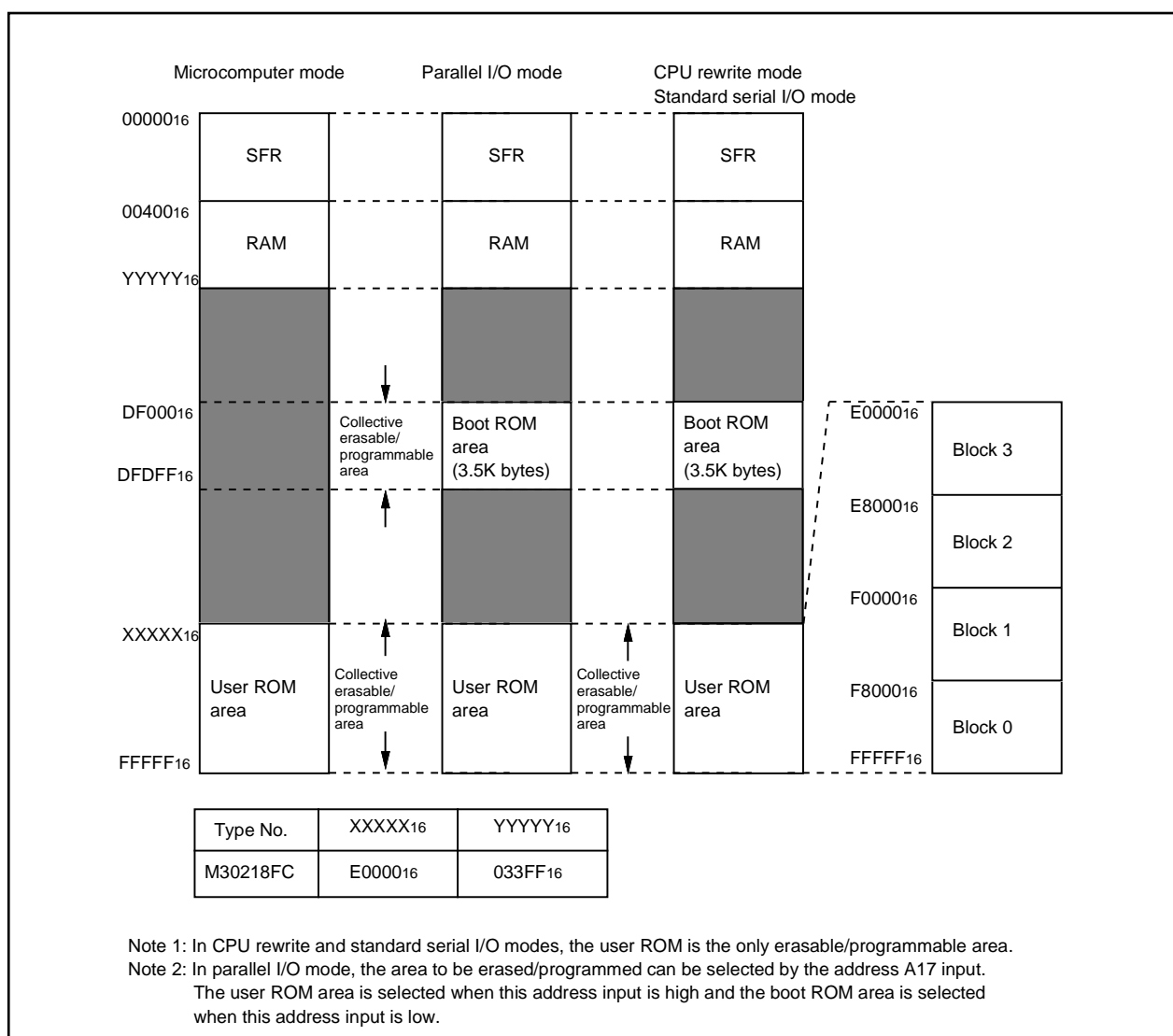


Figure 125. Block diagram of flash memory version

## CPU Rewrite Mode

## 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, the flash memory can be operated on by reading or writing to the flash memory control register and flash command register. Figure 126, Figure 127 show the flash memory control register, and flash command register respectively.

Also, in CPU rewrite mode, the CNVSS pin is used as the VPP power supply pin. Apply the power supply voltage, VPPH, from an external source to this pin.

In CPU rewrite mode, only the user ROM area shown in Figure 128 can be rewritten; the boot ROM area cannot be rewritten. Make sure the program and block commands are issued for only the user ROM 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 internal RAM before it can be executed.

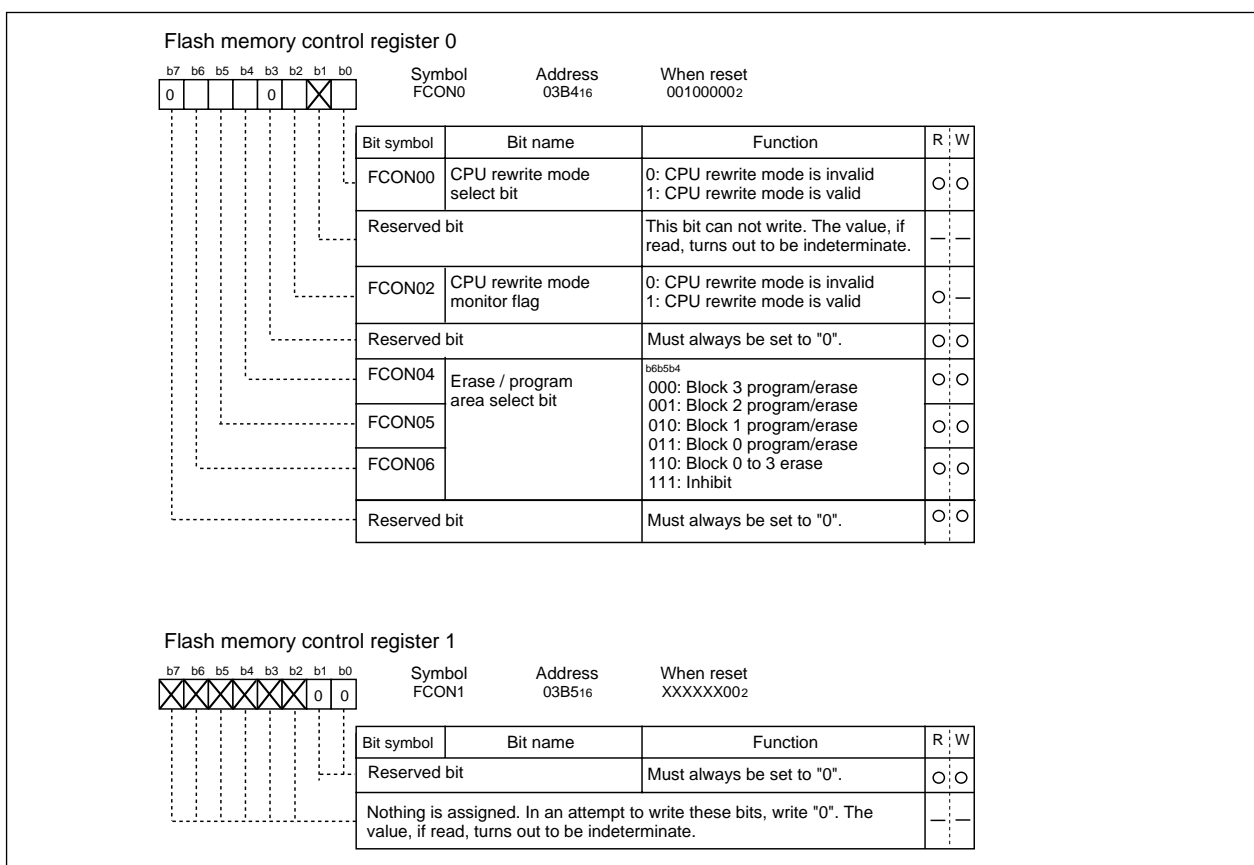


Figure 126. Flash memory control register

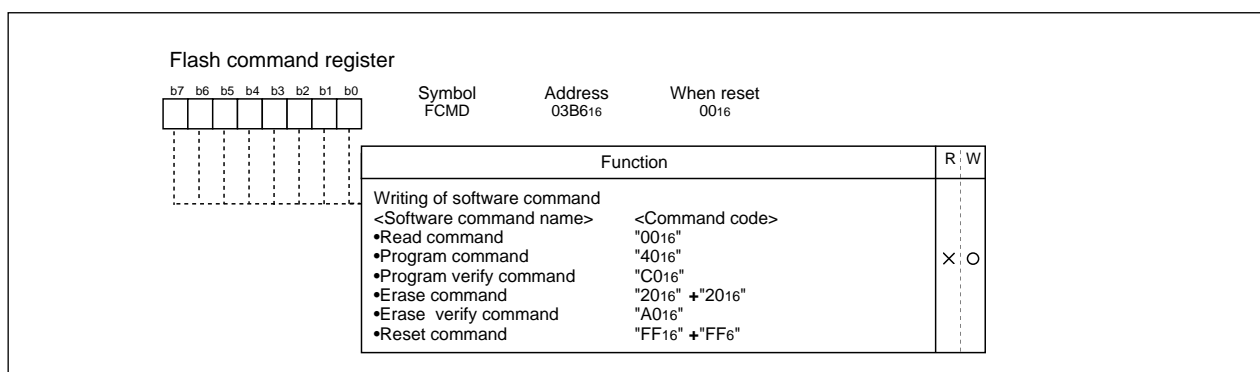


Figure 127. Flash command register

## 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 125 for details about the boot ROM area.

Normal microcomputer mode is entered when the microcomputer is reset with pulling CNVSS pin low (Vss). In this case, the CPU starts operating using the control program in the user ROM area.

When the microcomputer is reset by pulling the P52 pin high (VCC), the CNVSS pin high (VPPH), the CPU starts operating using the control program in the boot ROM area. This mode is called the "boot" mode.

The control program in the boot ROM area can also be used to rewrite the user ROM area.

## CPU rewrite mode operation procedure

The internal flash memory can be operated on to program, read, verify, or erase it while being placed on-board by writing commands from the CPU to the flash memory control register (addresses 03B416, 03B516) and flash command register (address 03B616). Note that when in CPU rewrite mode, the boot ROM area cannot be accessed for program, read, verify, or erase operations. Before this can be accomplished, a CPU write control program must be written into the boot ROM area in parallel input/output mode. The following shows a CPU rewrite mode operation procedure.

### <Start procedure (Note 1)>

- (1) Apply VPPH to the CNVSS/VPP pin and VCC to the port P46 pin for reset release. Or the user can jump from the user ROM area to the boot ROM area using the JMP instruction and execute the CPU write control program. In this case, set the CPU write mode select bit of the flash memory control register to "1" before applying VPPH to the CNVSS/VPP pin.
- (2) After transferring the CPU write control program from the boot ROM area to the internal RAM, jump to this control program in RAM. (The operations described below are controlled by this program.)
- (3) Set the CPU rewrite mode select bit to "1".
- (4) Read the CPU rewrite mode monitor flag to see that the CPU rewrite mode is enabled.
- (5) Execute operation on the flash memory by writing software commands to the flash command register.

Note 1: In addition to the above, various other operations need to be performed, such as for entering the data to be written to flash memory from an external source (e.g., serial I/O), initializing the ports, and writing to the watchdog timer.

### <Clearing procedure>

- (1) Apply VSS to the CNVSS/VPP pin.
- (2) Set the CPU rewrite mode select bit to "0".

## 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 erase/program mode, set BCLK to one of the following frequencies by changing the divide ratio:

5 MHz or less when wait bit (bit 7 at address 0005<sub>16</sub>) = 0 (without internal access wait state)

10 MHz or less when wait bit (bit 7 at address 0005<sub>16</sub>) = 1 (with internal access wait state)(Note 1)

### (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

No interrupts can be used that look up the fixed vector table in the flash memory area. Maskable interrupts may be used by setting the interrupt vector table in a location outside the flash memory area.

Note 1: Internal access wait state can be set in CPU rewrite mode. In this time, the following function is only used.

- CPU, ROM, RAM, timer, UART, SI/O2(non-automatic transfer), port

In case of setting internal access wait state, refer to the following explain (software wait).

### Software wait

A software wait can be inserted by setting the wait bit (bit 7) of the processor mode register 1 (address 0005<sub>16</sub>) (Note 2).

A software wait is inserted in the internal ROM/RAM area by setting the wait bit of the processor mode register 1. When set to "0", each bus cycle is executed in one BCLK cycle. When set to "1", each bus cycle is executed in two BCLK cycles. After the microcomputer has been reset, this bit defaults to "0".

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

Table 73 shows the software wait and bus cycles. Figure 128 shows example bus timing when using software waits.

Note 2: Before attempting to change the contents of the processor mode register 1, set bit 1 of the protect register (address 000A<sub>16</sub>) to "1".

**Table 73. 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

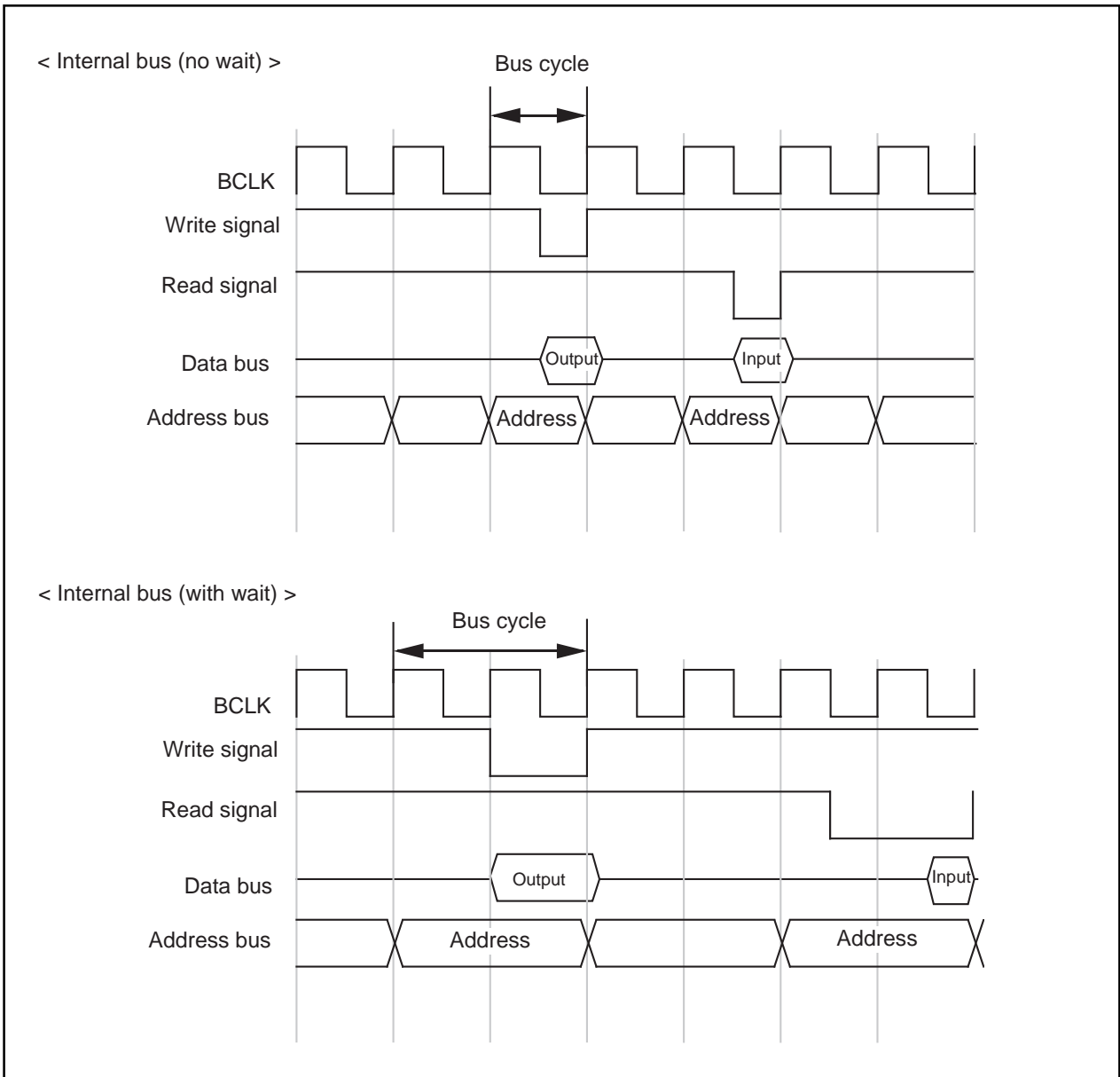


Figure 128. Typical bus timings using software wait

## Software Commands

Table 74 lists the software commands available with the M30218 group (flash memory version).

When CPU rewrite mode is enabled, write software commands to the flash command register to specify the operation to erase or program.

The content of each software command is explained below.

**Table 74. List of Software Commands (CPU Rewrite Mode)**

Command	First bus cycle			Second bus cycle		
	Mode	Address	Data (D <sub>0</sub> to D <sub>7</sub> )	Mode	Address	Data (D <sub>0</sub> to D <sub>7</sub> )
Read	Write	03B6 <sub>16</sub>	00 <sub>16</sub>			
Program	Write	03B6 <sub>16</sub>	40 <sub>16</sub>	Write	Program address	Program data
Program verify	Write	03B6 <sub>16</sub>	C0 <sub>16</sub>	Read	Verify address	Verify data
Erase	Write	03B6 <sub>16</sub>	20 <sub>16</sub>	Write	03B6 <sub>16</sub>	20 <sub>16</sub>
Erase verify	Write	03B6 <sub>16</sub>	A0 <sub>16</sub>	Read	Verify address	Verify data
Reset	Write	03B6 <sub>16</sub>	FF <sub>16</sub>	Write	03B6 <sub>16</sub>	FF <sub>16</sub>

### Read Command (00<sub>16</sub>)

The read mode is entered by writing the command code “00<sub>16</sub>” to the flash command register in the first bus cycle. When an 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<sub>0</sub>–D<sub>7</sub>), 8 bits at a time.

The read mode is retained intact until another command is written.

After reset and after the reset command is executed, the read mode is set.

### Program Command (40<sub>16</sub>)

The program mode is entered by writing the command code “40<sub>16</sub>” to the flash command register in the first bus cycle. When the user execute an instruction to write byte data to the desired address (e.g., STE instruction) in the second bus cycle, the flash memory control circuit executes the program operation. The program operation requires approximately 20 μs. Wait for 20 μs or more before the user go to the next processing.

During program operation, the watchdog timer remains idle, with the value “7FFF<sub>16</sub>” set in it.

Note 1: The write operation is not completed immediately by writing a program command once. The user must always execute a program-verify command after each program command executed. And if verification fails, the user need to execute the program command repeatedly until the verification passes. See Figure 129 for an example of a programming flowchart.

**Program-verify command (C0<sub>16</sub>)**

The program-verify mode is entered by writing the command code "C0<sub>16</sub>" to the flash command register in the first bus cycle. When the user execute an instruction (e.g., LDE instruction) to read byte data from the address to be verified (the previously programmed address) in the second bus cycle, the content that has actually been written to the address is read out from the memory.

The CPU compares this read data with the data that it previously wrote to the address using the program command. If the compared data do not match, the user need to execute the program and program-verify operations one more time.

**Erase command (20<sub>16</sub> + 20<sub>16</sub>)**

The flash memory control circuit executes an erase operation by writing command code "20<sub>16</sub>" to the flash command register in the first bus cycle and the same command code to the flash command register again in the second bus cycle. The erase operation requires approximately 20 ms. Wait for 20 ms or more before the user go to the next processing.

Before this erase command can be performed, all memory locations to be erased must have had data "00<sub>16</sub>" written to by using the program and program-verify commands. During erase operation, the watchdog timer remains idle, with the value "7FFF<sub>16</sub> set in it.

Note 1: The erase operation is not completed immediately by writing an erase command once. The user must always execute an erase-verify command after each erase command executed. And if verification fails, the user need to execute the erase command repeatedly until the verification passes. See Figure 129 for an example of an erase flowchart.

**Erase-verify command (A0<sub>16</sub>)**

The erase-verify mode is entered by writing the command code "A0<sub>16</sub>" to the flash command register in the first bus cycle. When the user execute an instruction to read byte data from the address to be verified (e.g., LDE instruction) in the second bus cycle, the content of the address is read out.

The CPU must sequentially erase-verify memory contents one address at a time, over the entire area erased. If any address is encountered whose content is not "FF<sub>16</sub>" (not erased), the CPU must stop erase-verify at that point and execute erase and erase-verify operations one more time.

Note 1: If any unerased memory location is encountered during erase-verify operation, be sure to execute erase and erase-verify operations one more time. In this case, however, the user does not need to write data "00<sub>16</sub>" to memory before erasing.

## CPU Rewrite Mode

**Reset command (FF16 + FF16)**

The reset command is used to stop the program command or the erase command in the middle of operation. After writing command code "4016" or "2016" twice to the flash command register, write command code "FF16" to the flash command register in the first bus cycle and the same command code to the flash command register again in the second bus cycle. The program command or erase command is disabled, with the flash memory placed in read mode.

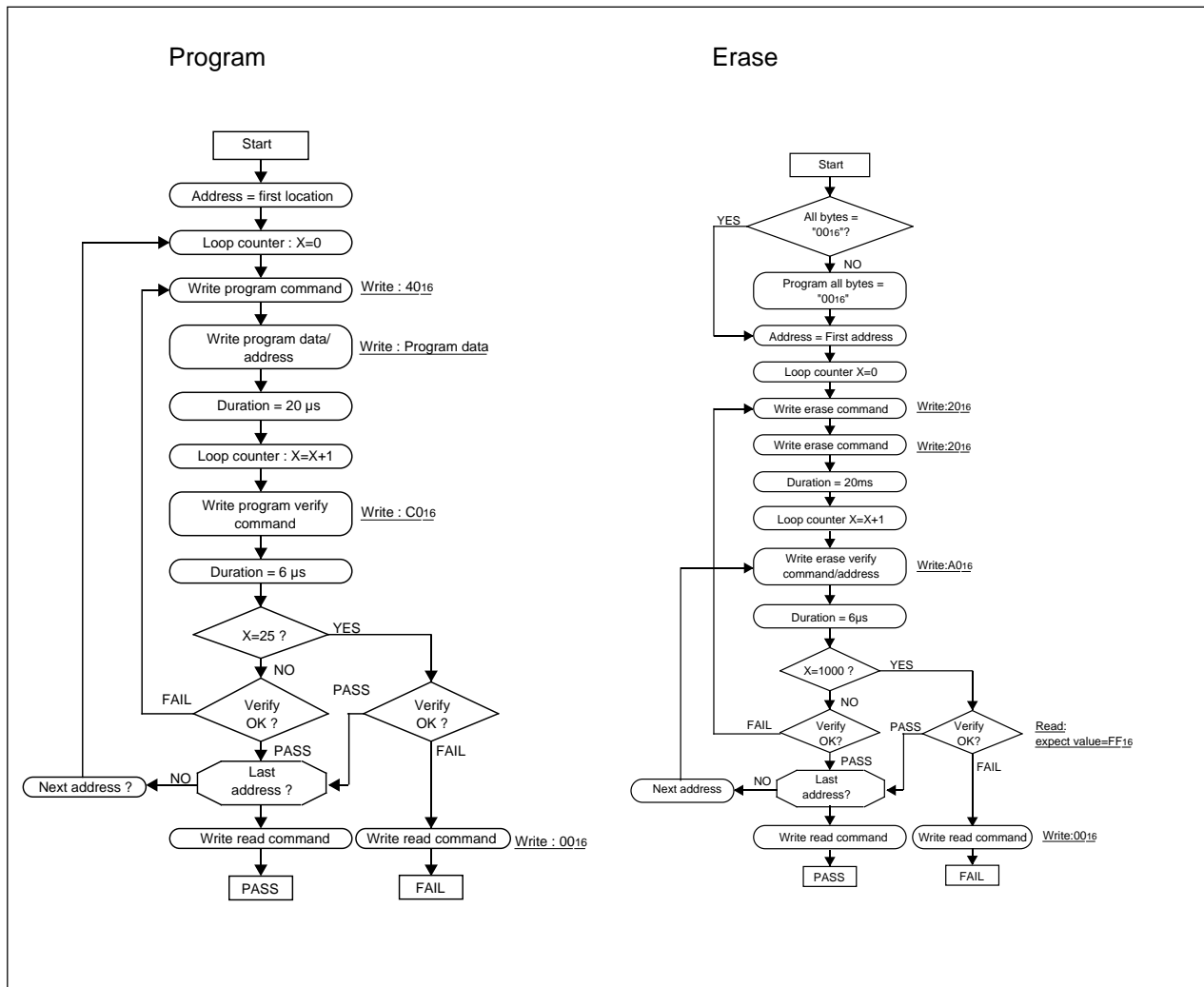


Figure 129. Program and erase execution flowchart in the CPU rewrite mode



## Appendix Standard Serial I/O Mode

## Pin functions (Flash memory standard serial I/O mode)

Pin	Name	I/O	Description
Vcc, Vss	Power input		Apply 5V $\pm$ 10 % to Vcc pin and 0 V to Vss pin.
CNVss	CNVss	I	Apply 12V $\pm$ 5 % to this pin.
RESET	Reset input	I	Reset input pin. While reset is "L" level, a 20 cycle or longer clock must be input to XIN pin.
XIN	Clock input	I	Connect a ceramic resonator or crystal oscillator between XIN and XOUT pins. To input an externally generated clock, input it to XIN pin and open XOUT pin.
XOUT	Clock output	O	
AVcc, AVss	Analog power supply input		Connect AVss to Vss and AVcc to Vcc, respectively.
VREF	Reference voltage input	I	Enter the reference voltage for AD from this pin.
P00 to P07	Output port P0	O	Output exclusive use pin.
P10 to P17	Output port P1	O	Output exclusive use pin.
P20 to P27	Output port P2	O	Output exclusive use pin.
P30 to P37	Input port P3	I	Input "H" or "L" level signal or open.
P40 to P43	Input port P4	I	Input "H" or "L" level signal or open.
P44	TxD output	O	Serial data output pin.
P45	RxD input	I	Serial data input pin.
P46	SCLK input	I	Serial clock input pin.
P47	BUSY output	O	BUSY signal output pin.
P50 to P57	Output port P5	O	Output exclusive use pin.
P60 to P67	Output port P6	O	Output exclusive use pin.
P70 to P77	Input port P7	I	Input "H" or "L" level signal or open.
P80 to P87	Input port P8	I	Input "H" or "L" level signal or open.
P90 to P97	Input port P9	I	Input "H" or "L" level signal or open.
P100 to P107	Input port P10	I	Input "H" or "L" level signal or open.

Appendix Standard Serial I/O Mode

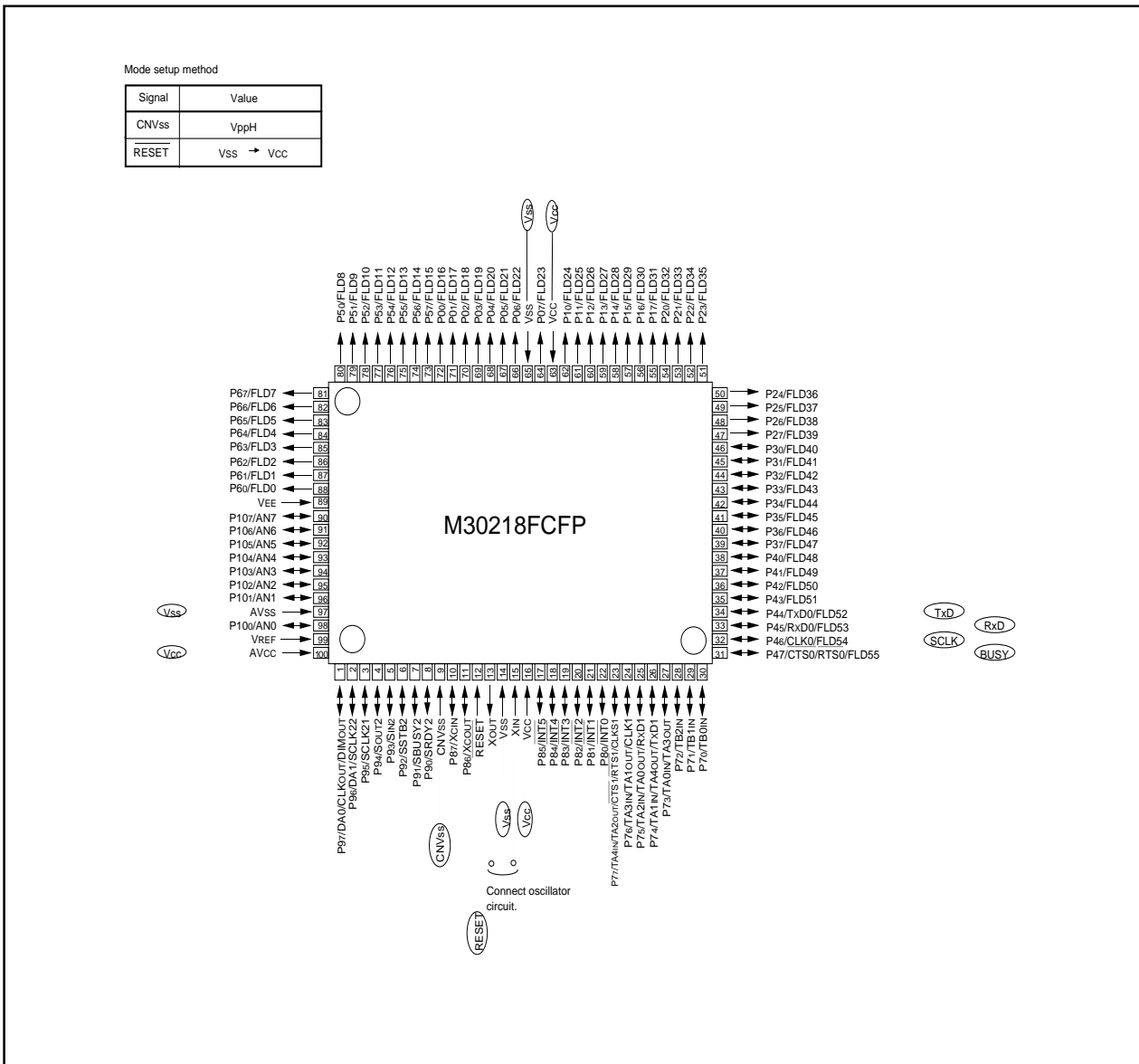


Figure 130. Pin connections for serial I/O mode (1)

## Standard Serial I/O Mode

The standard serial I/O mode serially inputs and outputs the software commands, addresses and data necessary for operating (read, program, erase, etc.) the internal flash memory. It uses a purpose-specific serial programmer.

The standard serial I/O mode differs from the parallel I/O mode in that the CPU controls operations like rewriting (uses the CPU rewrite mode) in the flash memory or serial input for rewriting data. The standard serial I/O mode is started by clearing the reset with  $V_{PPH}$  at the  $CNV_{ss}$  pin. (For the normal microprocessor mode, set  $CNV_{ss}$  to "L".)

This control program is written in the boot ROM area when shipped from Mitsubishi Electric. Therefore, if the boot ROM area is rewritten in the parallel I/O mode, the standard serial I/O mode cannot be used. Figure 130 shows the pin connections for the standard serial I/O mode. Serial data I/O uses three UART0 pins:  $CLK_0$ ,  $RxD_0$ ,  $TxD_0$ , and  $RTS_0$  (BUSY).

The  $CLK_0$  pin is the transfer clock input pin and it transfers the external transfer clock. The  $TxD_0$  pin outputs the CMOS signal. The  $RTS_0$  (BUSY) pin outputs an "L" level when reception setup ends and an "H" level when the reception operation starts. Transmission and reception data is transferred serially in 8-byte blocks.

In the standard serial I/O mode, only the user ROM area shown in Figure 125 can be rewritten, the boot ROM area cannot.

The standard serial I/O mode has a 7-byte ID code. When the flash memory is not blank and the ID code does not match the content of the flash memory, the command sent from the programmer is not accepted.

## Function Overview (Standard Serial I/O Mode)

In the standard serial I/O mode, software commands, addresses and data are input and output between the flash memory and an external device (serial programmer, etc.) using a clock synchronized serial I/O (UART0). In reception, the software commands, addresses and program data are synchronized with the rise of the transfer clock input to the  $CLK_0$  pin and input into the flash memory via the  $RxD_0$  pin.

In transmission, the read data and status are synchronized with the fall of the transfer clock and output to the outside from the  $TxD_0$  pin.

The  $TxD_0$  pin is CMOS output. Transmission is in 8-bit blocks and LSB first.

When busy, either during transmission or reception, or while executing an erase operation or program, the  $RTS_0$  (BUSY) pin is "H" level. Accordingly, do not start the next transmission until the  $RTS_0$  (BUSY) pin is "L" level.

Also, data in memory and the status register can be read after inputting a software command. It is possible to check flash memory operating status or whether a program or erase operation ended successfully or in error by reading the status register.

Software commands and the status register are explained here following.

## Appendix Standard Serial I/O Mode

## Software Commands

Table 75 lists software commands. In the standard serial I/O mode, erase operations, programs and reading are controlled by transferring software commands via the RxD pin. Software commands are explained here below.

Table 75. Software commands (Standard serial I/O mode)

	Control command	1st byte transfer	2nd byte	3rd byte	4th byte	5th byte	6th byte		When ID is not verificate
1	Page read	FF <sub>16</sub>	Address (middle)	Address (high)	Data output	Data output	Data output	Data output to 259th byte	Not acceptable
2	Page program	41 <sub>16</sub>	Address (middle)	Address (high)	Data input	Data input	Data input	Data input to 259th byte	Not acceptable
3	Bclock ease	20 <sub>16</sub>	Address (middle)	Address (high)	D0 <sub>16</sub>				Not acceptable
4	Erase all unlocked blocks	A7 <sub>16</sub>	D0 <sub>16</sub>						Not acceptable
5	Read status register	70 <sub>16</sub>	SRD output	SRD1 output					Acceptable
6	Clear status register	50 <sub>16</sub>							Not acceptable
7	Read lockbit status	71 <sub>16</sub>	Address (middle)	Address (high)	Lock bit data output				Not acceptable
8	ID check function	F5 <sub>16</sub>	Address (low)	Address (middle)	Address (high)	ID size	ID1	To ID7	Acceptable
9	Download function	FA <sub>16</sub>	Size (low)	Size (high)	Check-sum	Data input	To required number of times		Not acceptable
10	Version data output function	FB <sub>16</sub>	Version data output	Version data output	Version data output	Version data output	Version data output	Version data output to 9th byte	Acceptable
11	Boot area output function	FC <sub>16</sub>	Address (middle)	Address (high)	Data output	Data output	Data output	Data output to 259th byte	Not acceptable

Note1: Shading indicates transfer from flash memory microcomputer to serial programmer. All other data is transferred from the serial programmer to the flash memory microcomputer.

Note2: SRD refers to status register data. SRD1 refers to status register 1 data.

Note3: 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) Send the "FF16" command code in the 1st byte of the transmission.
- (2) Send addresses A8 to A15 and A16 to A23 in the 2nd and 3rd bytes of the transmission respectively.
- (3) From the 4th byte onward, data (D0–D7) for the page (256 bytes) specified with addresses A8 to A23 will be output sequentially from the smallest address first in sync with the rise of the clock.

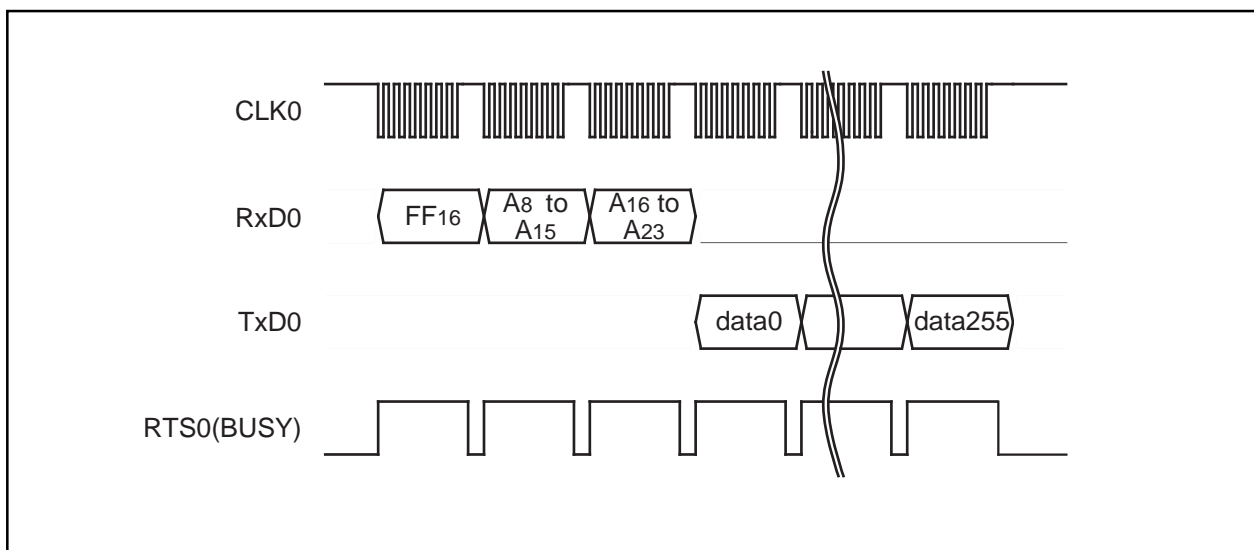


Figure 131. Timing for page read

### Read Status Register Command

This command reads status information. When the "7016" command code is sent in the 1st byte of the transmission, the contents of the status register (SRD) specified in the 2nd byte of the transmission and the contents of status register 1 (SRD1) specified in the 3rd byte of the transmission are read.

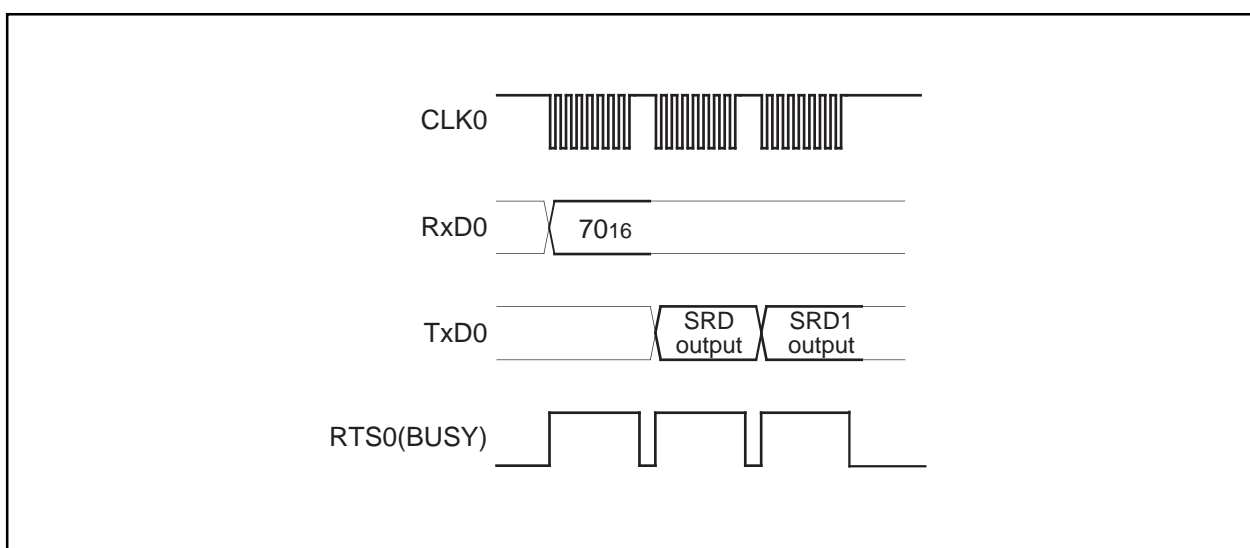


Figure 132. Timing for reading the status register

### Clear Status Register Command

This command clears the bits (SR3–SR4) which are set when the status register operation ends in error. When the “5016” command code is sent in the 1st byte of the transmission, the aforementioned bits are cleared. When the clear status register operation ends, the RTS0 (BUSY) signal changes from the “H” to the “L” level.

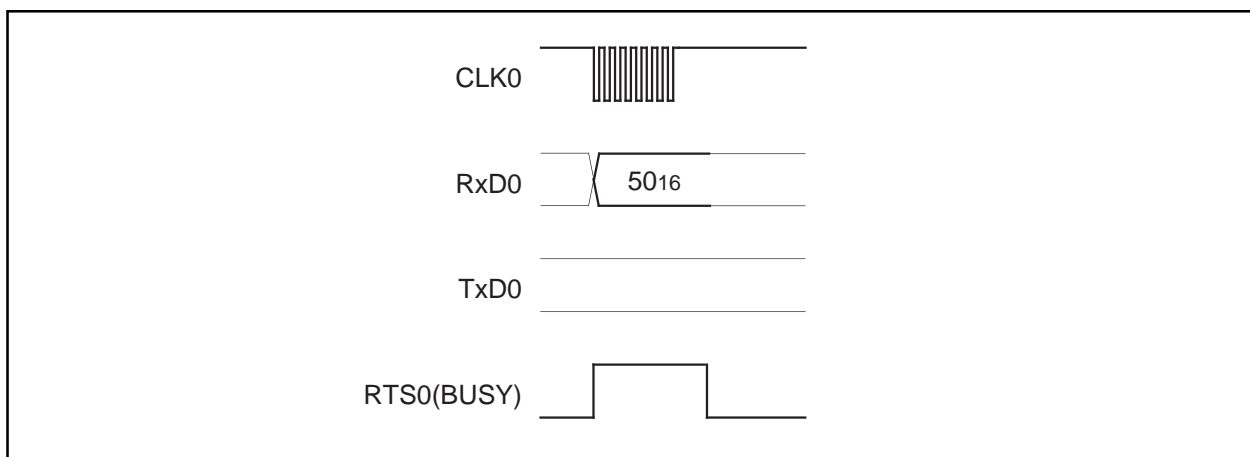


Figure 133. Timing for clearing the status register

### 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) Send the “4116” command code in the 1st byte of the transmission.
- (2) Send addresses A8 to A15 and A16 to A23 in the 2nd and 3rd bytes of the transmission respectively.
- (3) From the 4th byte onward, as write data (D0–D7) for the page (256 bytes) specified with addresses A8 to A23 is input sequentially from the smallest address first, that page is automatically written.

When reception setup for the next 256 bytes ends, the RTS0 (BUSY) signal changes from the “H” to the “L” level. The result of the page program can be known by reading the status register. For more information, see the section on the status register.

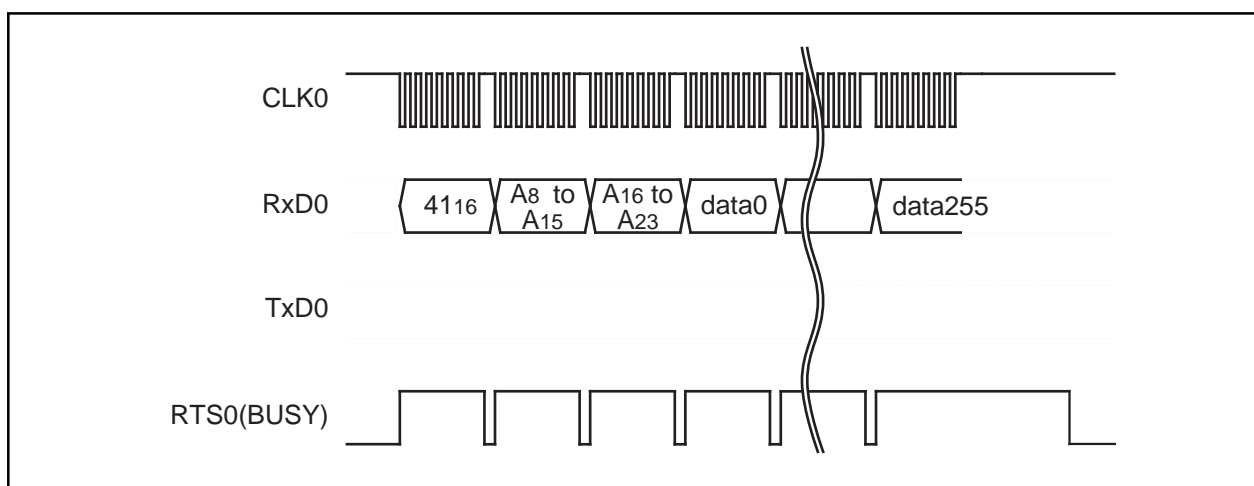


Figure 134. Timing for the page program

### Block Erase Command

This command erases the data in the specified block. Execute the block erase command as explained here following.

- (1) Send the "20<sub>16</sub>" command code in the 1st byte of the transmission.
- (2) Send addresses A<sub>8</sub> to A<sub>15</sub> and A<sub>16</sub> to A<sub>23</sub> in the 2nd and 3rd bytes of the transmission respectively.
- (3) Send the verify command code "D0<sub>16</sub>" in the 4th byte of the transmission. With the verify command code, the erase operation will start for the specified block in the flash memory. Write the highest address of the specified block for addresses A<sub>16</sub> to A<sub>23</sub>.

When block erasing ends, the RTS0 (BUSY) signal changes from the "H" to the "L" level. After block erase ends, the result of the block erase operation can be known by reading the status register. For more information, see the section on the status register.

Each block can be erase-protected with the lock bit. For more information, see the section on the data protection function.

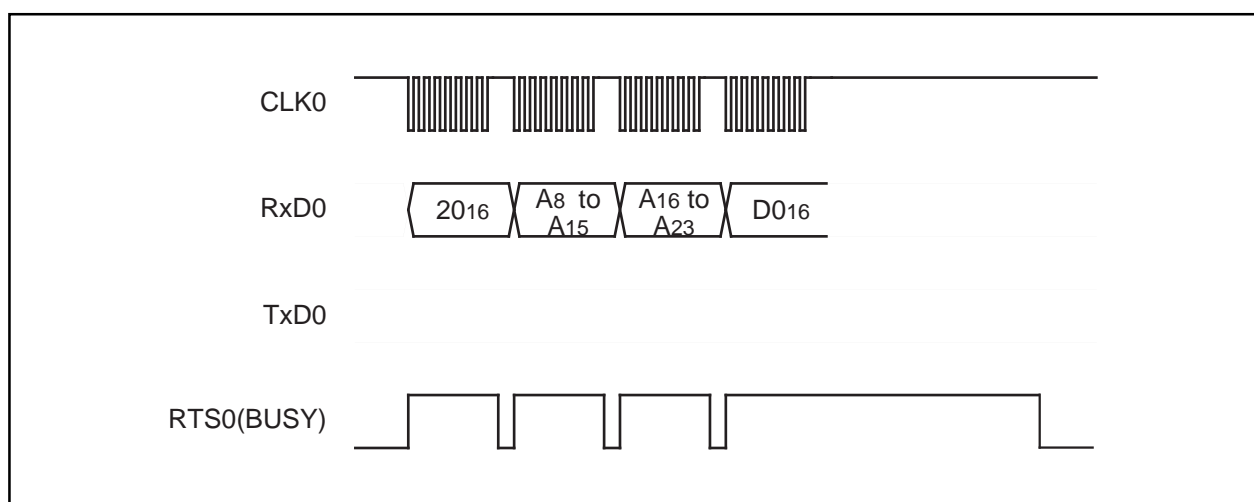


Figure 135. Timing for block erasing

### Erase All Unlocked Blocks Command

This command erases the content of all blocks. Execute the erase all unlocked blocks command as explained here following.

- (1) Send the "A7<sub>16</sub>" command code in the 1st byte of the transmission.
- (2) Send the verify command code "D0<sub>16</sub>" in the 2nd byte of the transmission. 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<sub>0</sub> (BUSY) signal changes from the "H" to the "L" level. The result of the erase operation can be known by reading the status register.

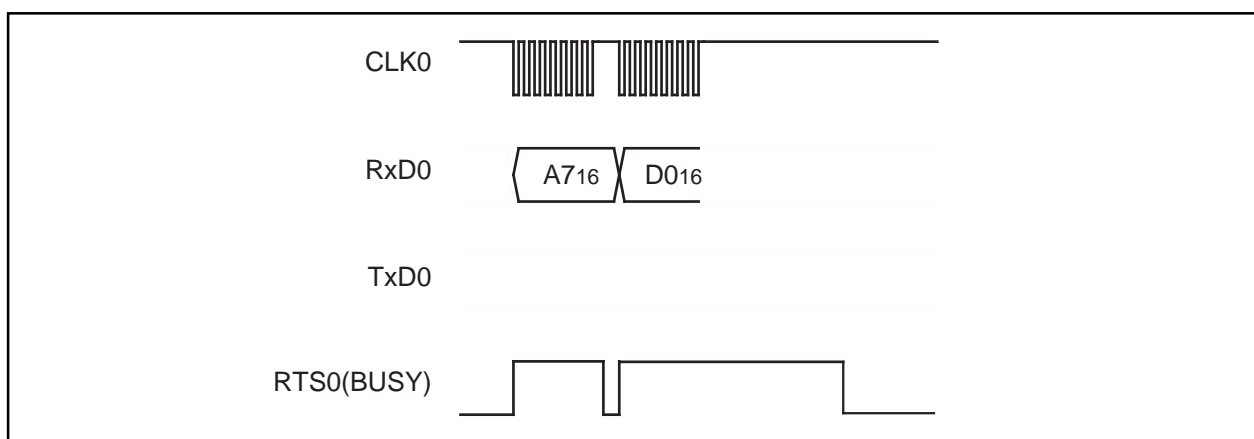


Figure 136. Timing for erasing all unlocked blocks

### Read Lock Bit Status Command

This command reads the lock bit status of the specified block. Execute the read lock bit status command as explained here following.

- (1) Send the "71<sub>16</sub>" command code in the 1st byte of the transmission.
- (2) Send addresses A<sub>8</sub> to A<sub>15</sub> and A<sub>16</sub> to A<sub>23</sub> in the 2nd and 3rd bytes of the transmission respectively.
- (3) The lock bit data of the specified block is output in the 4th byte of the transmission. Write the highest address of the specified block for addresses A<sub>8</sub> to A<sub>23</sub>.

The M30218 group (flash memory version) does not have the lock bit, so the read value is always "1" (block unlock).

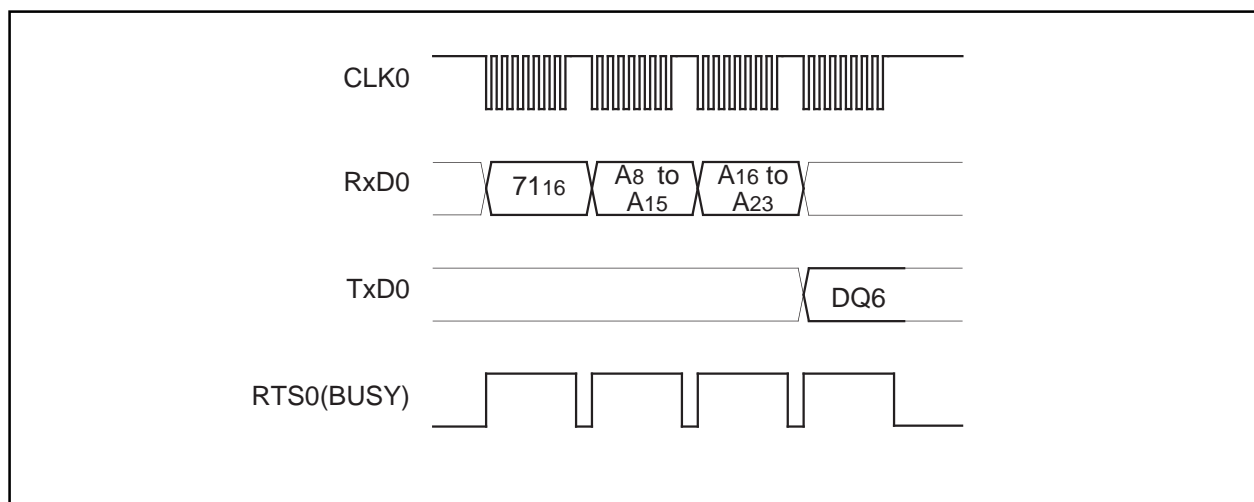


Figure 137. Timing for reading lock bit status



### Download Command

This command downloads a program to the RAM for execution. Execute the download command as explained here following.

- (1) Send the "FA16" command code in the 1st byte of the transmission.
- (2) Send the program size in the 2nd and 3rd bytes of the transmission.
- (3) Send the check sum in the 4th byte of the transmission. The check sum is added to all data sent in the 5th byte onward.
- (4) The program to execute is sent in 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.

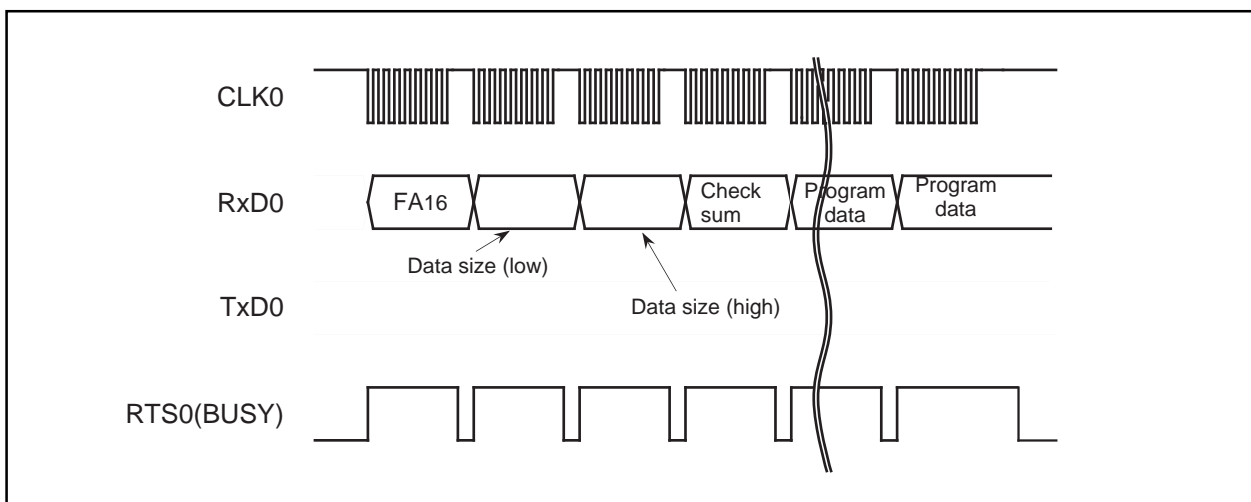


Figure 138. Timing for download

### 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) Send the "FB16" command code in the 1st byte of the transmission.
- (2) The version information will be output from the 2nd byte onward. This data is composed of 8 ASCII code characters.

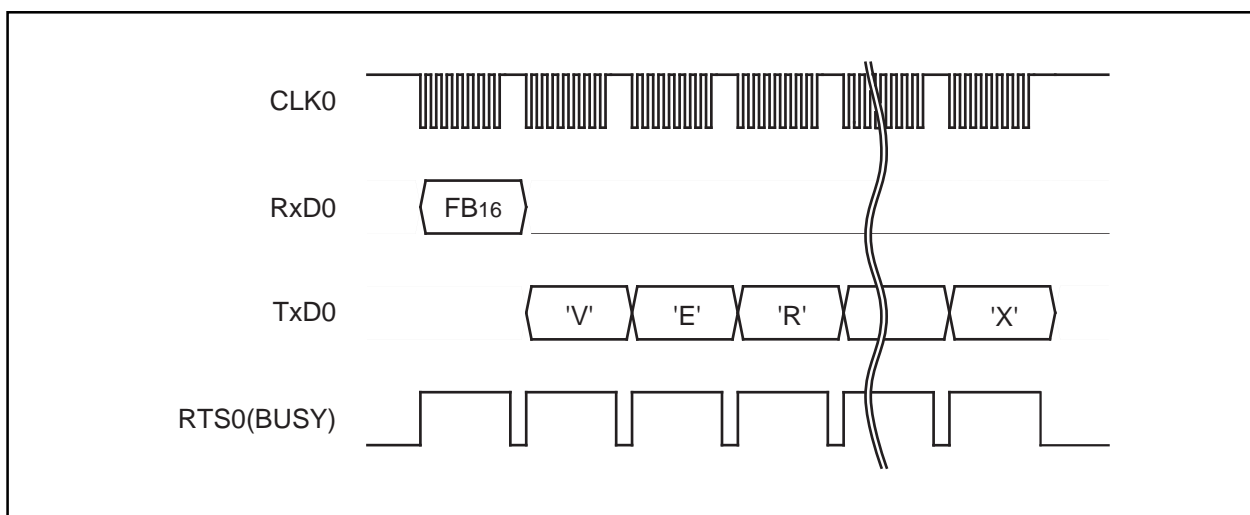


Figure 139. Timing for version information output

### Boot Area Output Command

This command outputs the control program stored in the boot area in one page blocks (256 bytes). Execute the boot area output command as explained here following.

- (1) Send the "FC16" command code in the 1st byte of the transmission.
- (2) Send addresses A8 to A15 and A16 to A23 in the 2nd and 3rd bytes of the transmission respectively.
- (3) From the 4th byte onward, data (D0–D7) for the page (256 bytes) specified with addresses A8 to A23 will be output sequentially from the smallest address first, in sync with the rise of the clock.

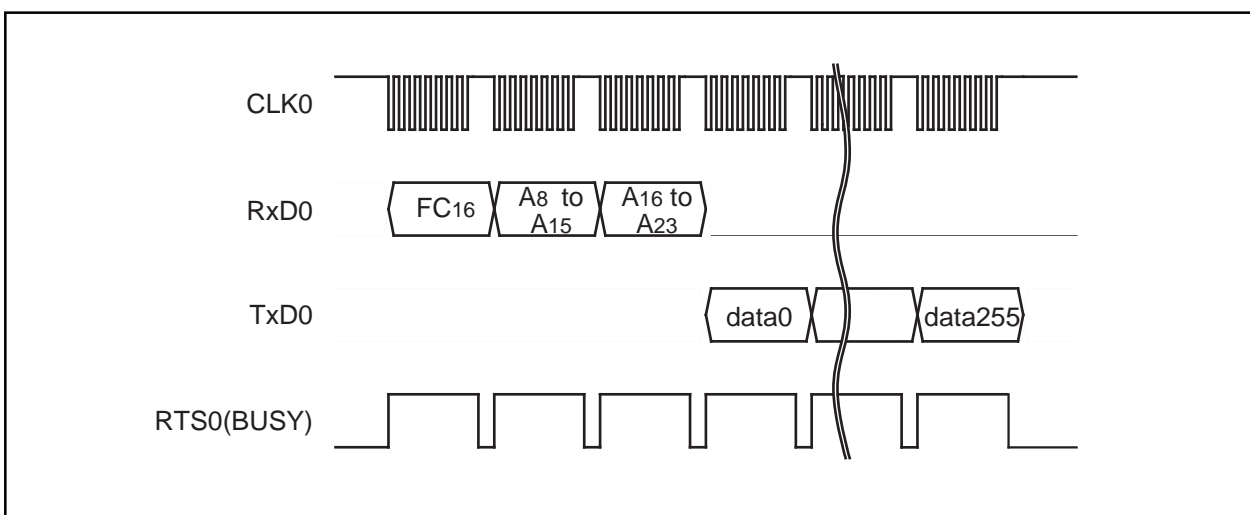


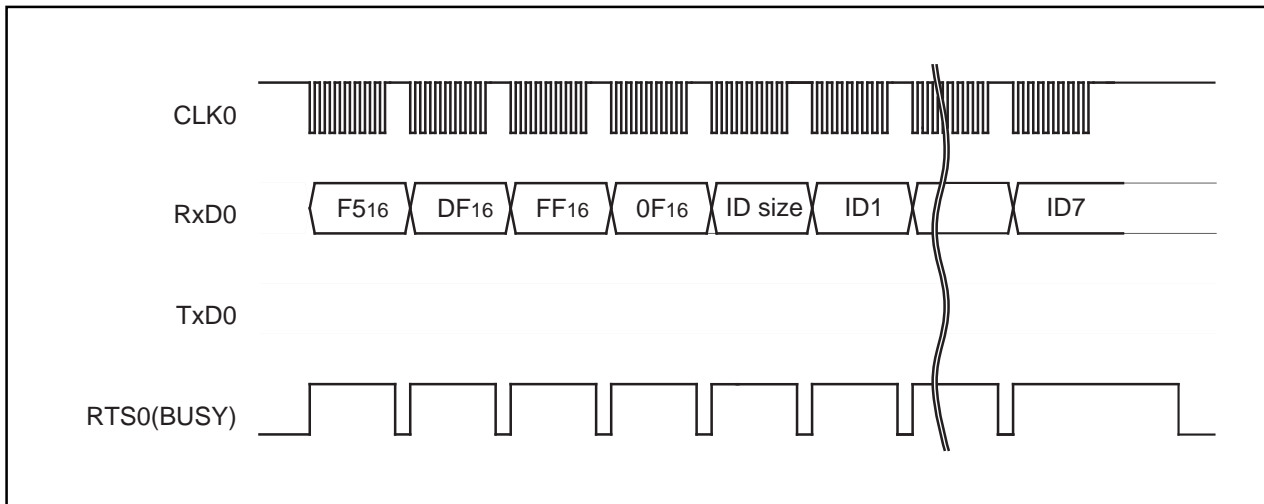
Figure 140. Timing for boot area output

## Appendix Standard Serial I/O Mode

**ID Check**

This command checks the ID code. Execute the boot ID check command as explained here following.

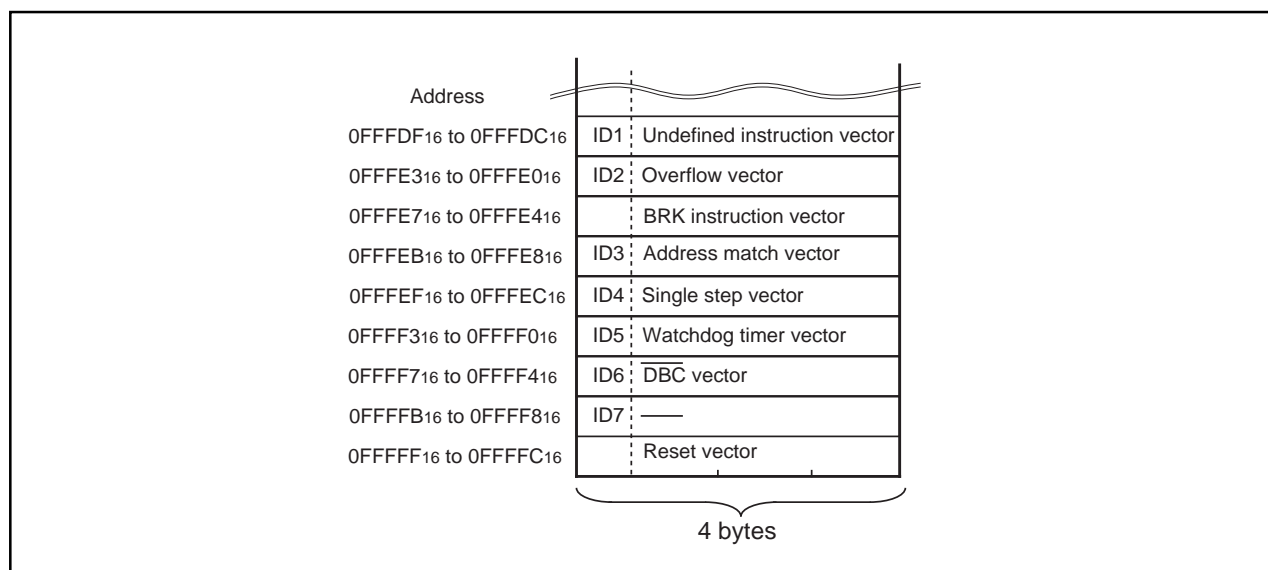
- (1) Send the "F5<sub>16</sub>" command code in the 1st byte of the transmission.
- (2) Send addresses A<sub>0</sub> to A<sub>7</sub>, A<sub>8</sub> to A<sub>15</sub> and A<sub>16</sub> to A<sub>23</sub> of the 1st byte of the ID code in the 2nd, 3rd and 4th bytes of the transmission respectively.
- (3) Send the number of data sets of the ID code in the 5th byte.
- (4) The ID code is sent in the 6th byte onward, starting with the 1st byte of the code.



**Figure 141. Timing for the ID check**

**ID Code**

When the flash memory is not blank, the ID code sent from the serial programmer and the ID code written in the flash memory are compared to see if they match. If the codes do not match, the command sent from the serial programmer is not accepted. An ID code contains 8 bits of data. Area is, from the 1st byte, addresses 0FFFDF<sub>16</sub>, 0FFFE3<sub>16</sub>, 0FFFE7<sub>16</sub>, 0FFFEF<sub>16</sub>, 0FFFF3<sub>16</sub>, and 0FFFF7<sub>16</sub>. Write a program into the flash memory, which already has the ID code set for these addresses.



**Figure 142. ID code storage addresses**

## 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<sub>16</sub>). Also, the status register is cleared by writing the clear status register command (50<sub>16</sub>). Table 76 gives the definition of each status register bit. After clearing the reset, the status register outputs "80<sub>16</sub>".

**Table 76. Status register (SRD)**

SRD0 bits	Status name	Definition	
		"1"	"0"
SR7 (bit7)	Status bit	Ready	Busy
SR6 (bit6)	Reserved	-	-
SR5 (bit5)	Erase bit	Terminated in error	Terminated normally
SR4 (bit4)	Program bit	Terminated in error	Terminated normally
SR3 (bit3)	Reserved	-	-
SR2 (bit2)	Reserved	-	-
SR1 (bit1)	Reserved	-	-
SR0 (bit0)	Reserved	-	-

### Status Bit (SR7)

The status bit indicates the operating status of the flash memory. When power is turned on, "1" (ready) is set for it. The bit is set to "0" (busy) during an auto write or auto erase operation, but it is set back to "1" when the operation ends.

### Erase Bit (SR5)

The erase bit 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 Bit (SR4)

The program bit 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 (7016). Also, status register 1 is cleared by writing the clear status register command (5016).

Table 77 gives the definition of each status register 1 bit. "0016" is output when power is turned ON and the flag status is maintained even after the reset.

**Table 77. 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)	Checksum match bit	Match	Mismatch
SR11 (bit3) SR10 (bit2)	ID check completed bits	00 01 10 11	Not verified Verification mismatch Reserved 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 Consistency Bit (SR12)

This flag indicates whether the check sum matches or not when a program, is downloaded for execution using the download function.

### ID Check Completed Bits (SR11 and SR10)

These flags indicate the result of ID checks. Some commands cannot be accepted without an ID check.

### Data Reception Time Out (SR9)

This flag indicates when a time out error is generated during data reception. If this flag is attached during data reception, the received data is discarded and the microcomputer returns to the command wait state.

### Example Circuit Application for The Standard Serial I/O Mode

The below figure shows a circuit application for the standard serial I/O mode. Control pins will vary according to programmer, therefore see the programmer manual for more information.

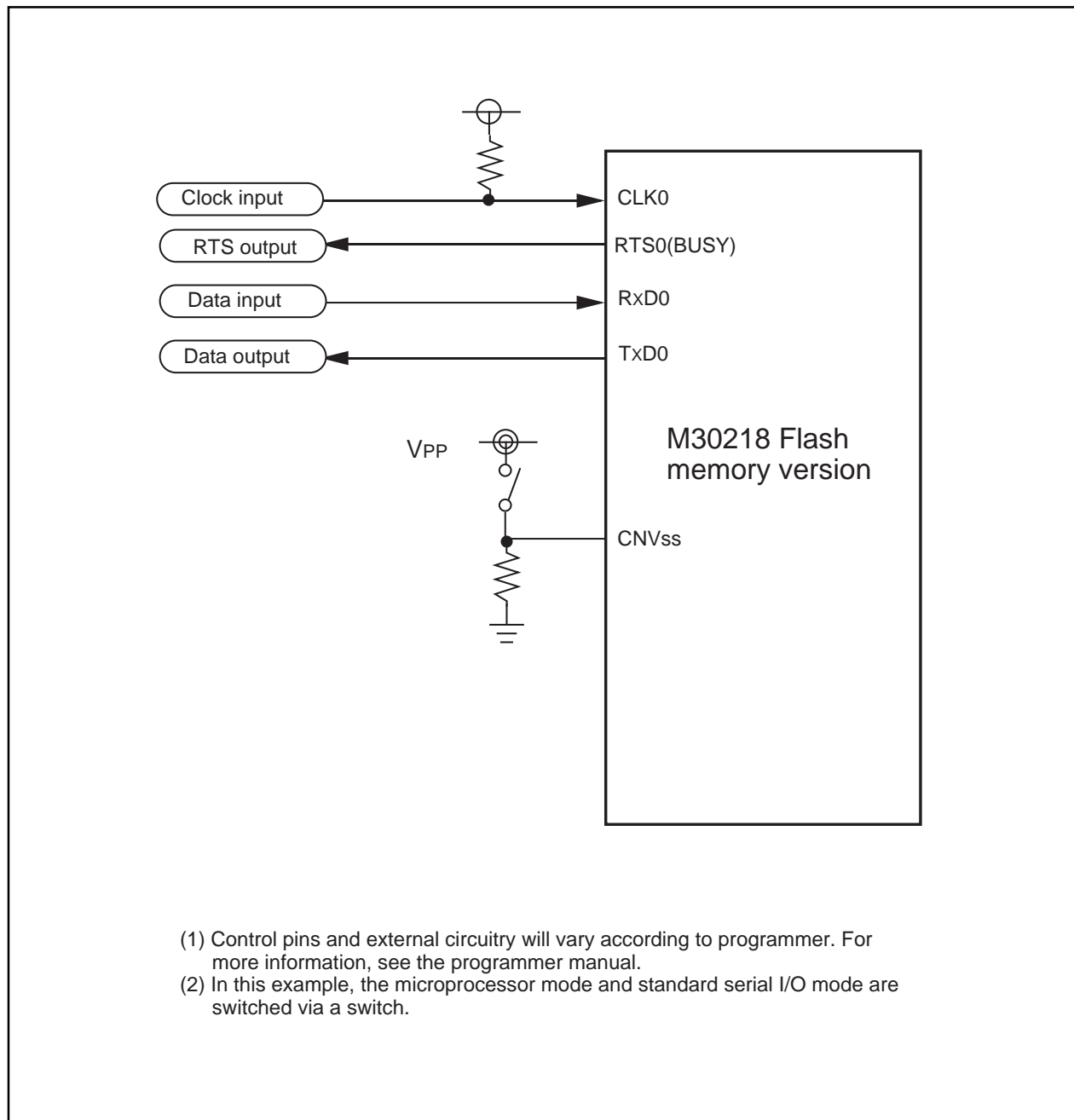
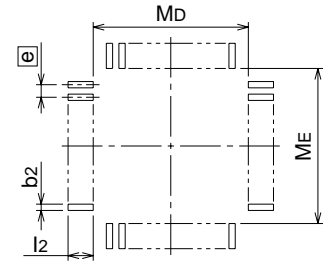
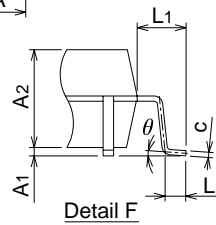
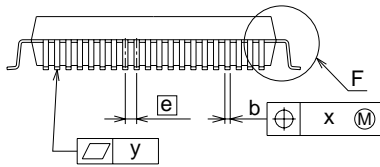
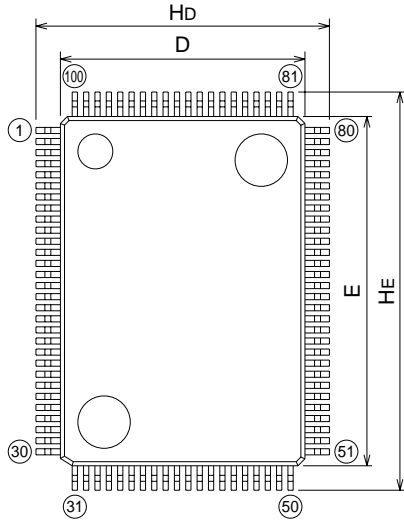


Figure 143. Example circuit application for the standard serial I/O mode

**100P6S-A**

**Plastic 100pin 14X20mm body QFP**

EIAJ Package Code	JEDEC Code	Weight(g)	Lead Material
QFP100-P-1420-0.65	-	1.58	Alloy 42



Recommended Mount Pad

Symbol	Dimension in Millimeters		
	Min	Nom	Max
A	-	-	3.05
A1	0	0.1	0.2
A2	-	2.8	-
b	0.25	0.3	0.4
c	0.13	0.15	0.2
D	13.8	14.0	14.2
E	19.8	20.0	20.2
e	-	0.65	-
HD	16.5	16.8	17.1
HE	22.5	22.8	23.1
L	0.4	0.6	0.8
L1	-	1.4	-
x	-	-	0.13
y	-	-	0.1
$\theta$	0°	-	10°
b2	-	0.35	-
l2	1.3	-	-
MD	-	14.6	-
ME	-	20.6	-

# Chapter 2

---

## Peripheral Functions Usage



## 2.1 Protect

### 2.1.1 Overview

'Protect' is a function that causes a value held in a register to be unchanged even when a program runs away. The following is an overview of the protect function:

#### (1) Registers affected by the protect function

The registers affected by the protect function are:

- (a) System clock control registers 0, 1 (addresses 0006<sub>16</sub> and 0007<sub>16</sub>)
- (b) Processor mode registers 0, 1 (addresses 0004<sub>16</sub> and 0005<sub>16</sub>)

The values in registers (1) through (2) cannot be changed in write-protect state. To change values in the registers, put the individual registers in write-enabled state.

#### (2) Protect register

Figure 2.1.1 shows protect register.

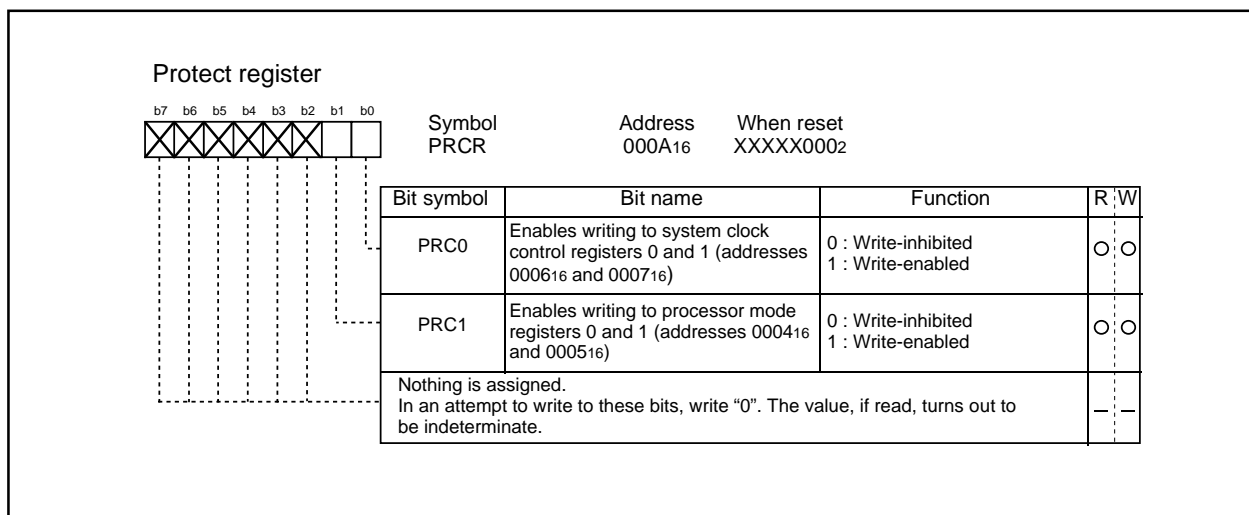
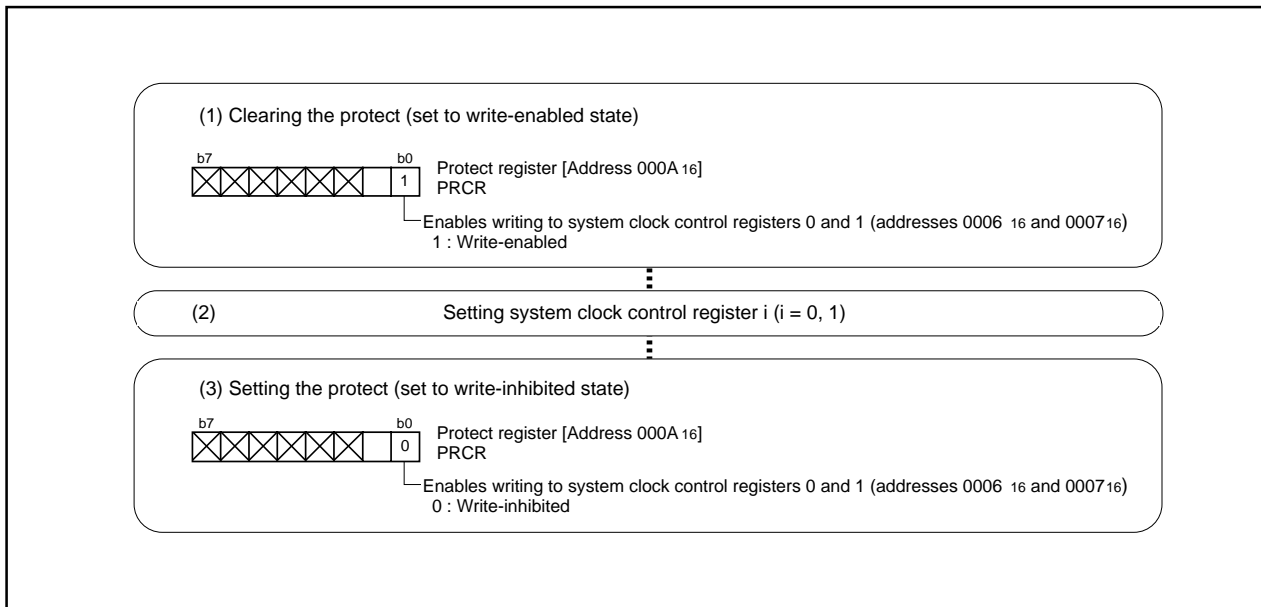


Figure 2.1.1. Protect register

### 2.1.2 Protect Operation

The following explains the protect operation. Figure 2.1.2 shows the set-up procedure.

- Operation
- (1) Setting "1" in the write-enable bit of system clock control registers 0 and 1 causes system clock control register 0 and system clock control register 1 to be in write-enabled state.
  - (2) The contents of system clock control register 0 and that of system clock control register 1 are changed.
  - (3) Setting "0" in the write-enable bit of system control registers 0 and 1 causes system clock control register 0 and system control register 1 to be in write-inhibited state.
  - (4) To change the contents of processor mode register 0 and that of processor mode register 1, follow the same steps as in dealing with system clock control registers.



**Figure 2.1.2. Set-up procedure for protect function**

## 2.2 Timer A

### 2.2.1 Overview

The following is an overview for timer A, a 16-bit timer.

#### (1) Mode

Timer A operates in one of the four modes:

##### (a) Timer mode

In this mode, the internal count source is counted. Two functions can be selected: the pulse output function that reverses output from a port every time an overflow occurs, or the gate function which controls the count start/stop according to the input signal from a port.

- Timer mode operation ..... P186
- Timer mode, gate function operation ..... P188
- Timer mode, pulse output function operation ..... P190

##### (b) Event counter mode

This mode counts the pulses from the outside and the number of underflows in other timers. The free-run type, in which nothing is reloaded from the reload register, can be selected when an underflow occurs. The pulse output function can also be selected. Please refer to the timer mode explanation for details, as the operation is identical.

- Event counter mode operation ..... P192
- Event counter mode, free run type operation ..... P194

Furthermore, Timer A has a 2-phase pulse signal processing function which generates an up count or down count in the event counter mode, depending on the phase of the two input signals.

- Operation of the 2-phase pulse signal processing function in normal event counter mode ..... P196
- Operation of the 2-phase pulse signal processing function in 4-multiplication mode ..... P198

##### (c) One-shot timer mode

In this mode, the timer is started by the trigger and stops when the timer goes to "0". The trigger can be selected from the following 3 types: an external input signal, an overflow of the timer, or a software trigger. The pulse output function can also be selected. Please refer to the timer mode explanation for details, as the operation is identical.

- Operation in one-shot timer mode effected by software ..... P200
- Operation in one-shot timer mode effected by an external trigger ..... P202

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

In this mode, the arbitrary pulses are successively output. Either a 16-bit fixed-period PWM mode or 8-bit variable-period mode can be selected. The trigger for initiating output can also be selected. Please refer to the one-shot timer mode explanation for details, as the operation is identical.

- 16-bit PWM mode operation ..... P204
- 8-bit PWM mode operation ..... P206

**(2) Count source**

The internal count source can be selected from f1, f8, f32, and fc32. Clocks f1, f8, and f32 are derived by dividing the CPU's main clock by 1, 8, and 32 respectively. Clock fc32 is derived by dividing the CPU's secondary clock by 32.

**(3) Frequency division ratio**

In timer mode or pulse width modulation mode, [the value set in the timer register + 1] becomes the frequency division ratio. In event counter mode, [the set value + 1] becomes the frequency division ratio when a down count is performed, or [FFFF<sub>16</sub> - the set value + 1] becomes the frequency division ratio when an up count is performed. In one-shot timer mode, the value set in the timer register becomes the frequency division ratio.

The counter overflows (or underflows) when a count source equal to a frequency division ratio is input, and an interrupt occurs. For the pulse output function, the output from the port varies (the value in the port register does not vary).

**(4) Reading the timer**

Either in timer mode or in event counter mode, reading the timer register takes out the count at that moment. Read it in 16-bit units. The data either in one-shot timer mode or in pulse width modulation mode is indeterminate.

**(5) Writing to the timer**

To write to the timer register when a count is in progress, the value is written only to the reload register. When writing to the timer register when a count is stopped, the value is written both to the reload register and to the counter. Write a value in 16-bit units.

**(6) Relation between the input/output to/from the timer and the direction register**

With the output function of the timer, pulses are output regardless of the direction register of the relevant port. To input an external signal to the timer, set the direction register of the relevant port to input.

**(7) Pins related to timer A**

- |  |  |
|--|--|
| (a) TA0IN, TA1IN, TA2IN, TA3IN, TA4IN      | Input pins to timer A.   |
| (b) TA0OUT, TA1OUT, TA2OUT, TA3OUT, TA4OUT | Output pins from timer A. They become input pins to timer A when event counter mode is active. |

**(8) Registers related to timer A**

Figure 2.2.1 shows the memory map of timer A-related registers. Figures 2.2.2 through 2.2.5 show timer A-related registers.

Timer A

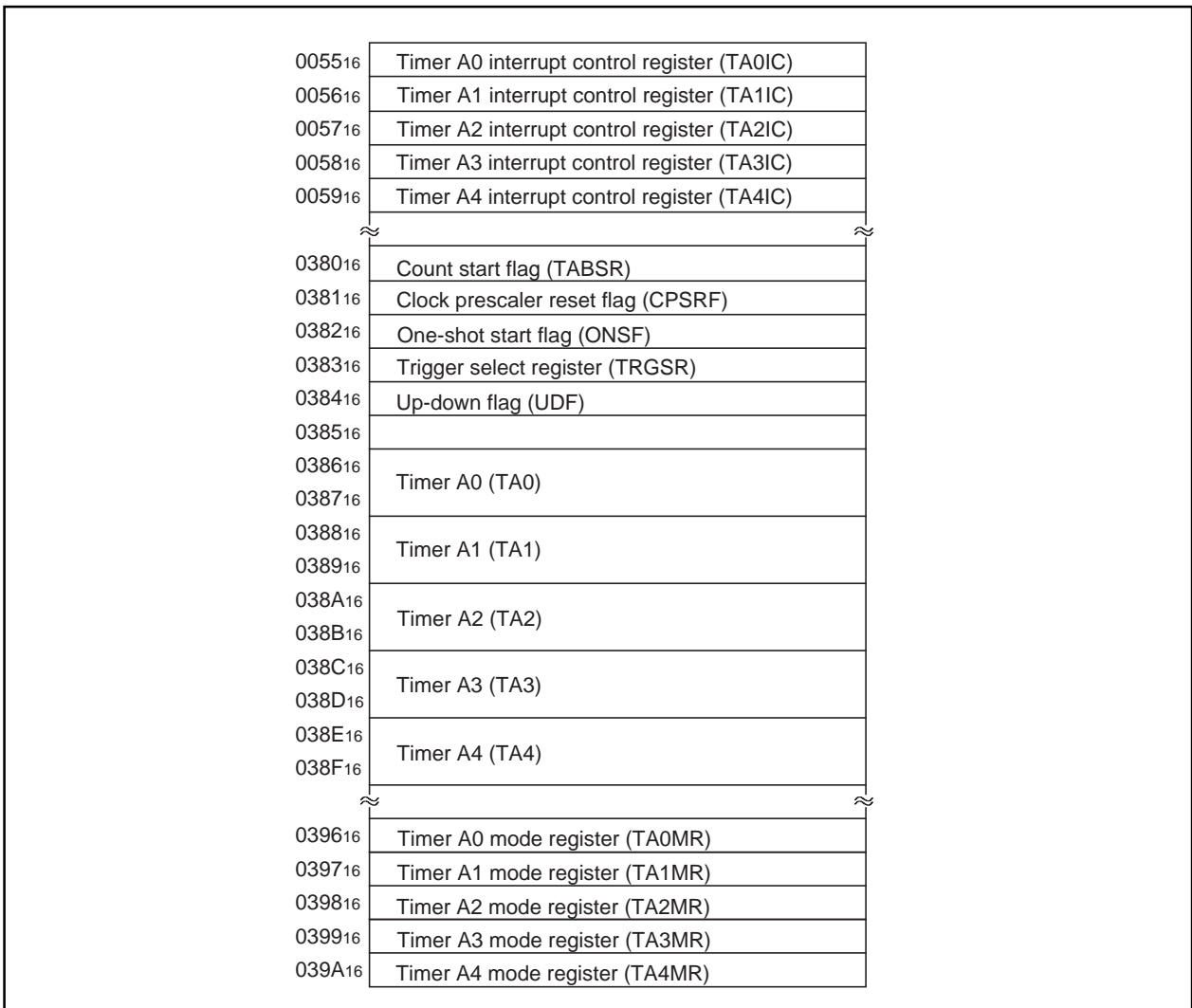


Figure 2.2.1. Memory map of timer A-related registers

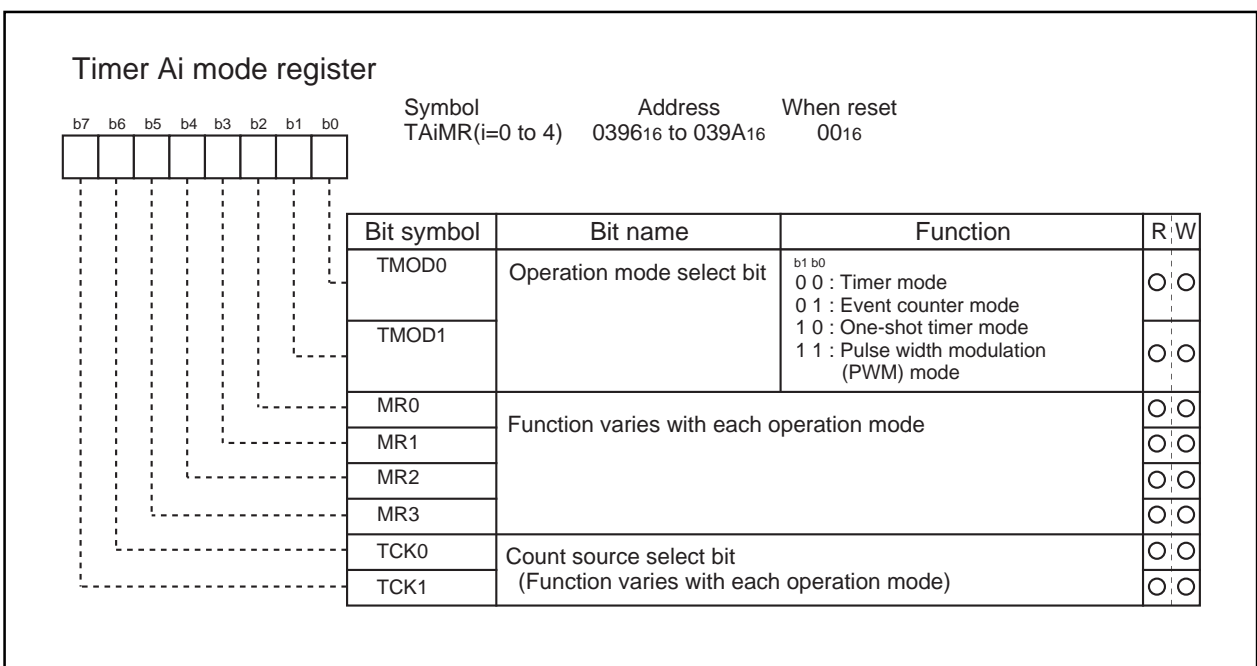


Figure 2.2.2. Timer A-related registers (1)

Timer A

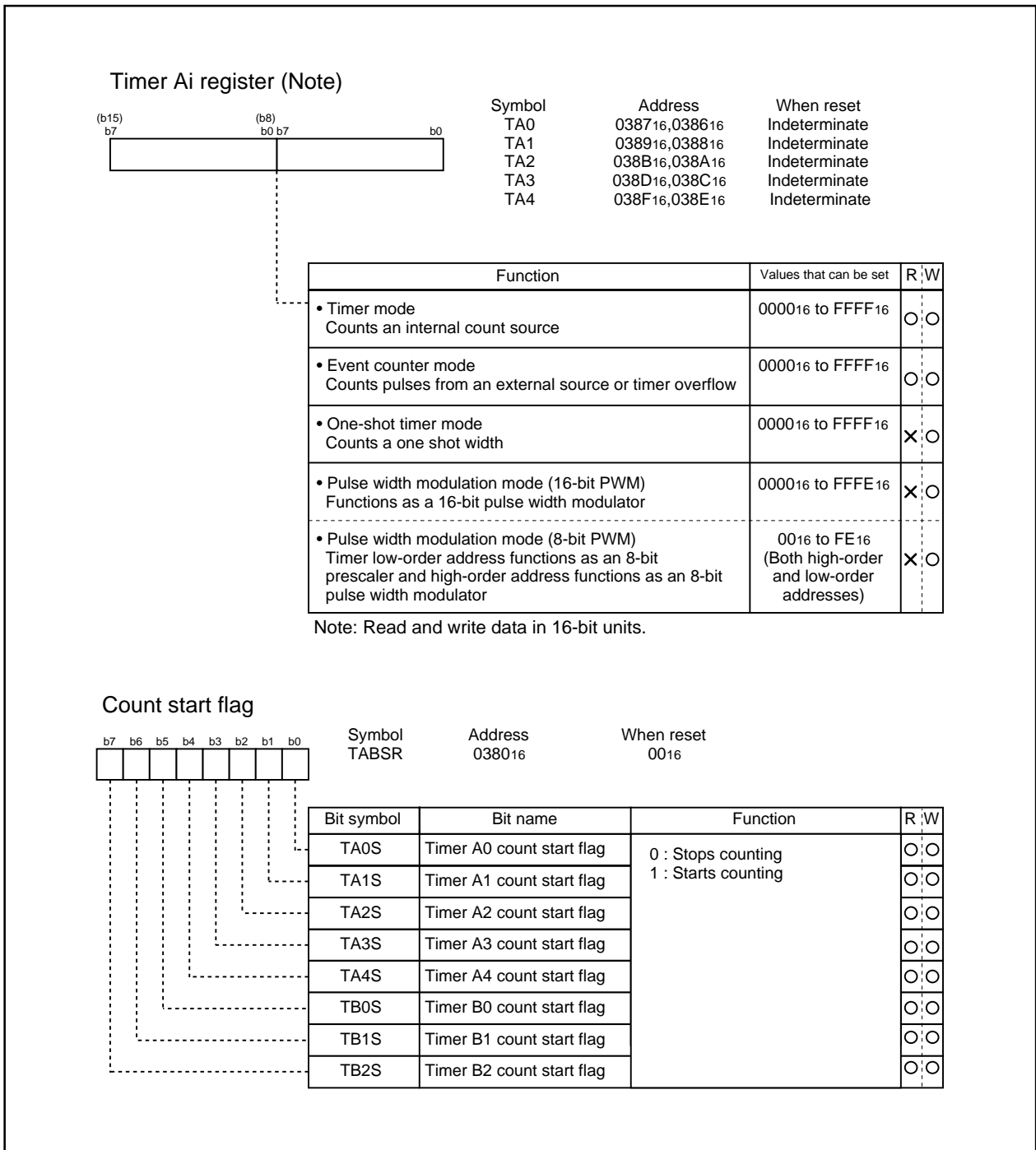


Figure 2.2.3. Timer A-related registers (2)

## Timer A

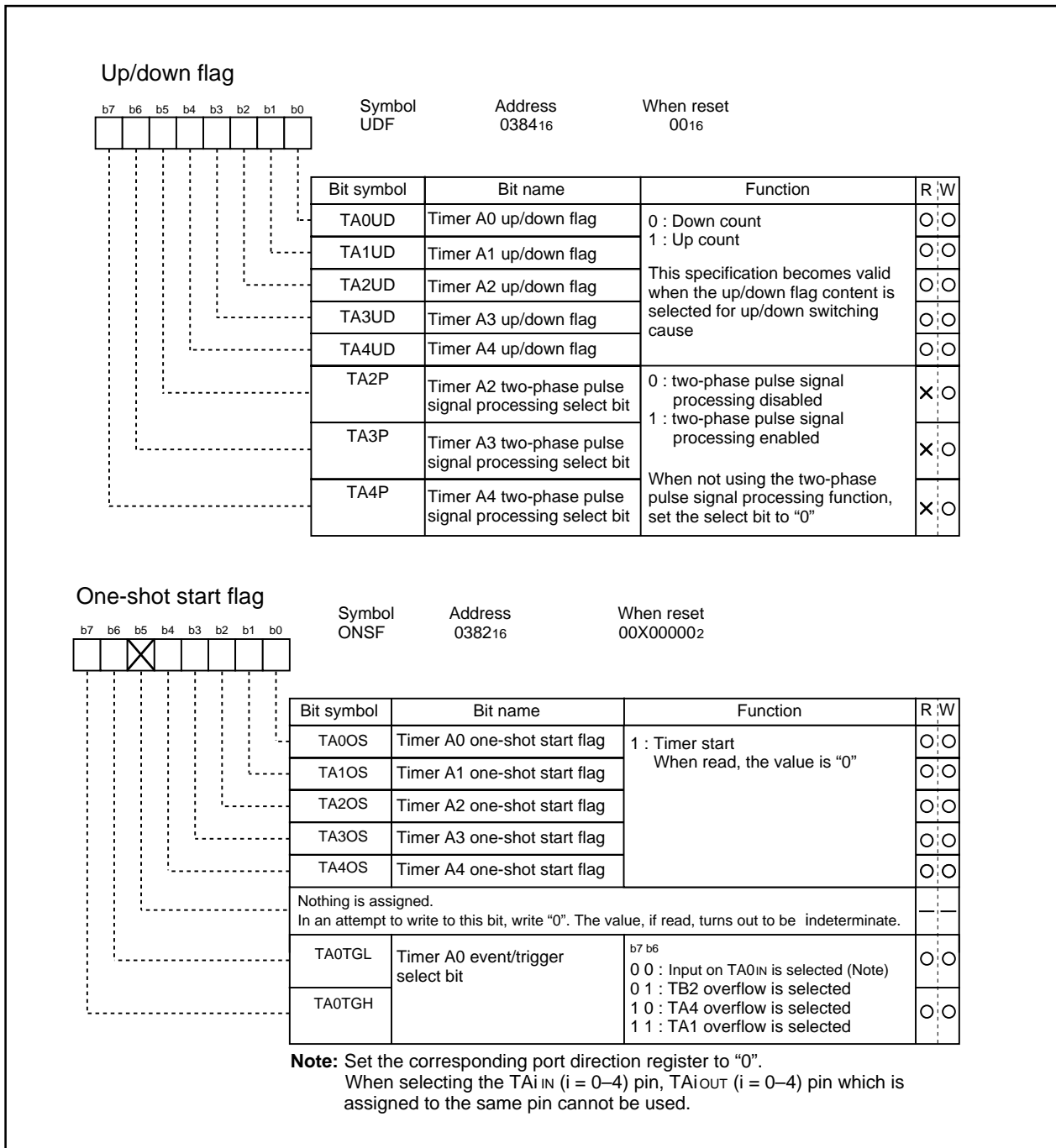


Figure 2.2.4. Timer A-related registers (3)

## Timer A

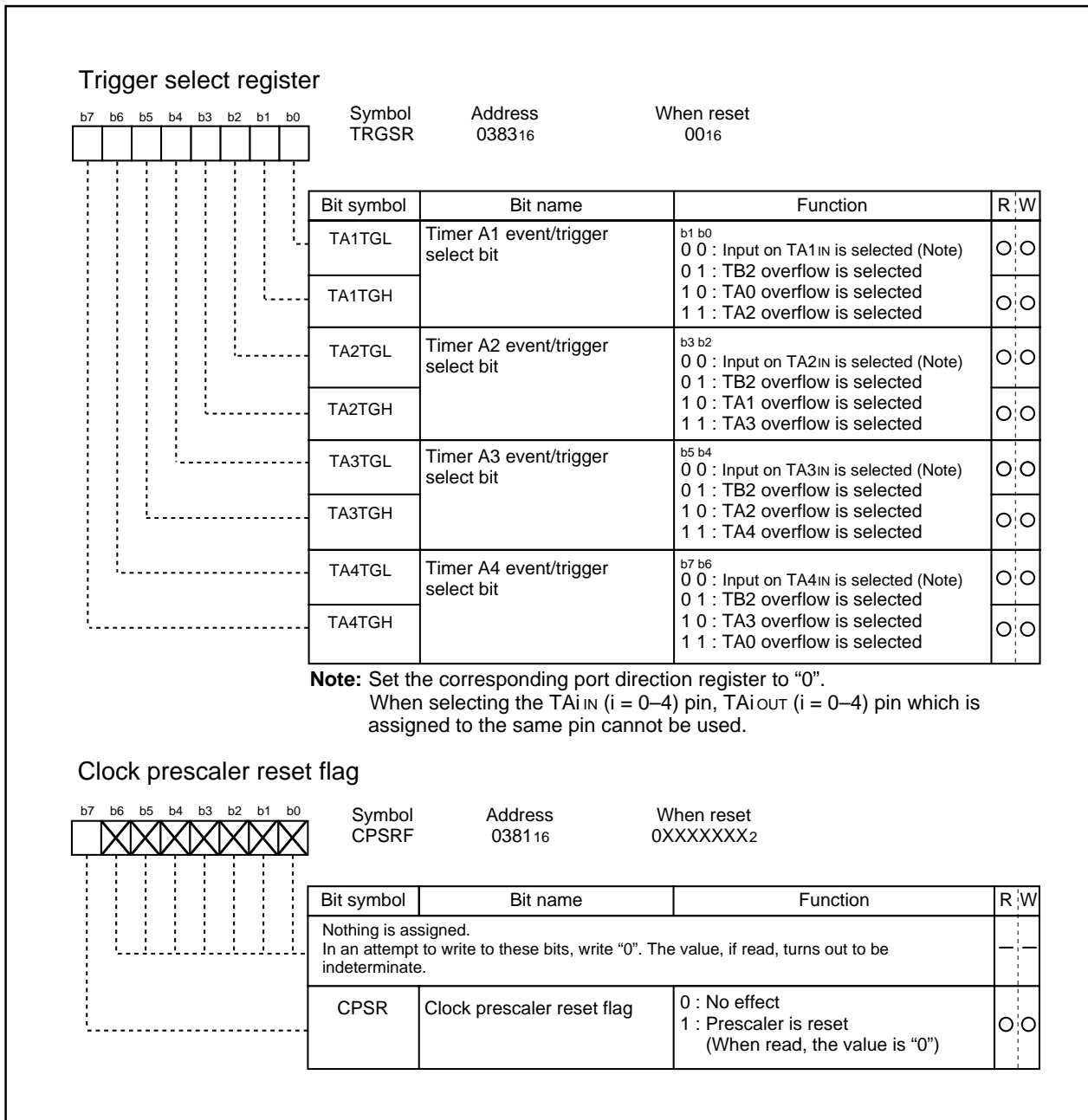


Figure 2.2.5. Timer A-related registers (4)



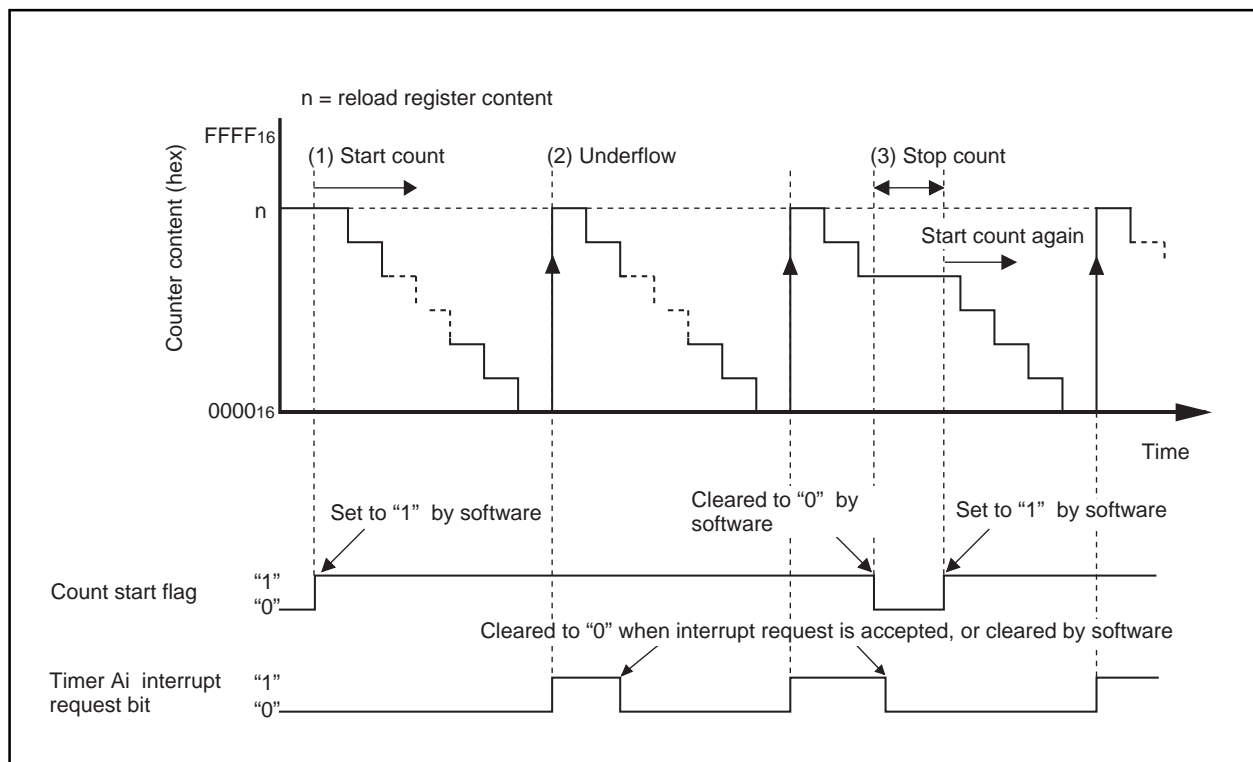
### 2.2.2 Operation of Timer A (timer mode)

In timer mode, choose functions from those listed in Table 2.2.1. Operations of the circled items are described below. Figure 2.2.6 shows the operation timing, and Figure 2.2.7 shows the set-up procedure.

**Table 2.2.1. Chosed functions**

Item	Set-up
Count source	○ Internal count source (f <sub>1</sub> / f <sub>8</sub> / f <sub>32</sub> / f <sub>c32</sub> )
Pulse output function	○ No pulses output
	Pulses output
Gate function	○ No gate function
	Performs count only for the period in which the TAIIN pin is at "L" level
	Performs count only for the period in which the TAIIN pin is at "H" level

- Operation
- (1) Setting the count start flag to "1" causes the counter to perform a down count on the count source.
  - (2) If an underflow occurs, the content of the reload register is reloaded, and the count continues. At this time, the timer Ai interrupt request bit goes to "1".
  - (3) Setting the count start flag to "0" causes the counter to hold its value and to stop.



**Figure 2.2.6. Operation timing of timer mode**

Timer A

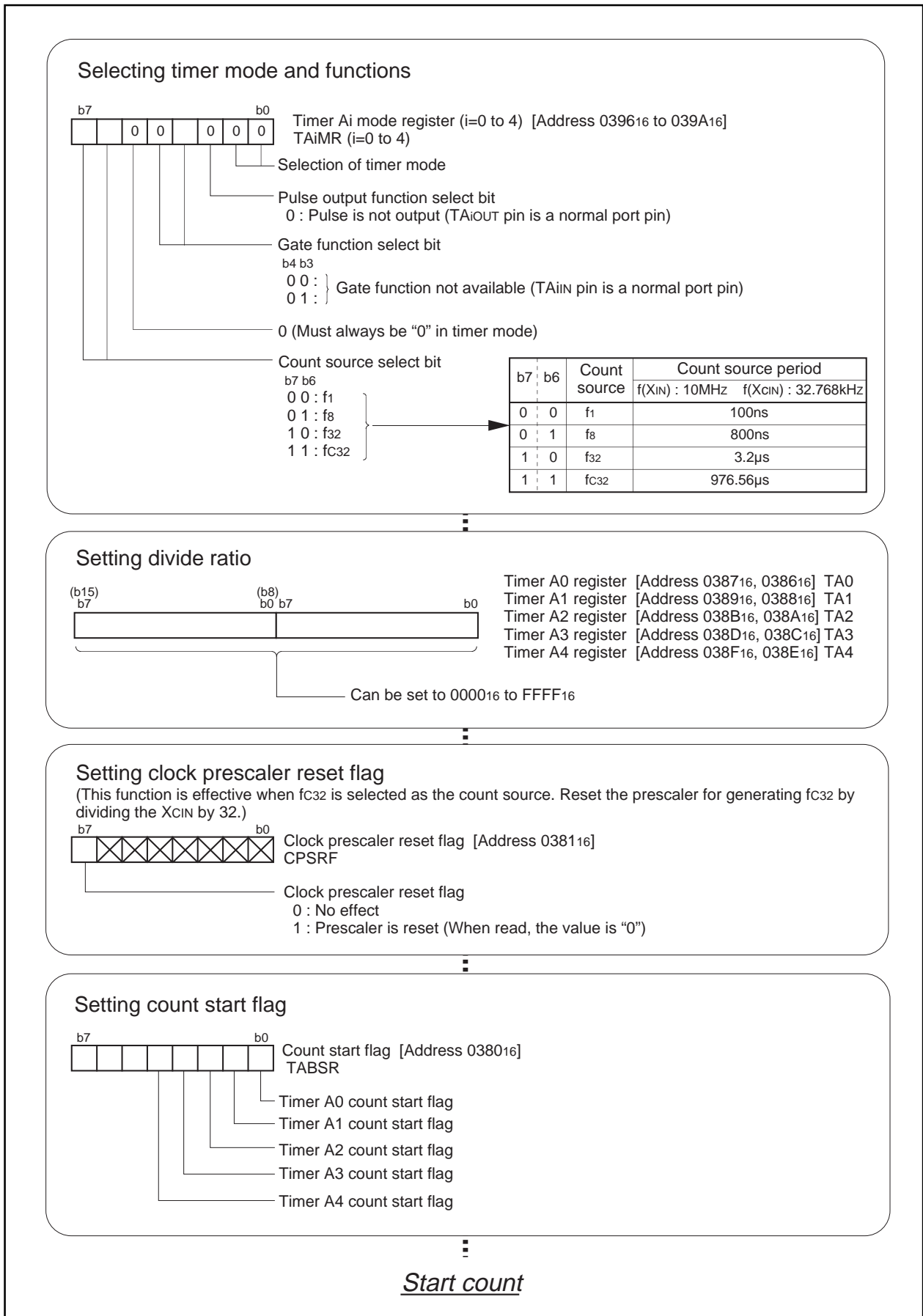


Figure 2.2.7. Set-up procedure of timer mode

### 2.2.3 Operation of Timer A (timer mode, gate function selected)

In timer mode, choose functions from those listed in Table 2.2.2. Operations of the circled items are described below. Figure 2.2.8 shows the operation timing, and Figure 2.2.9 shows the set-up procedure.

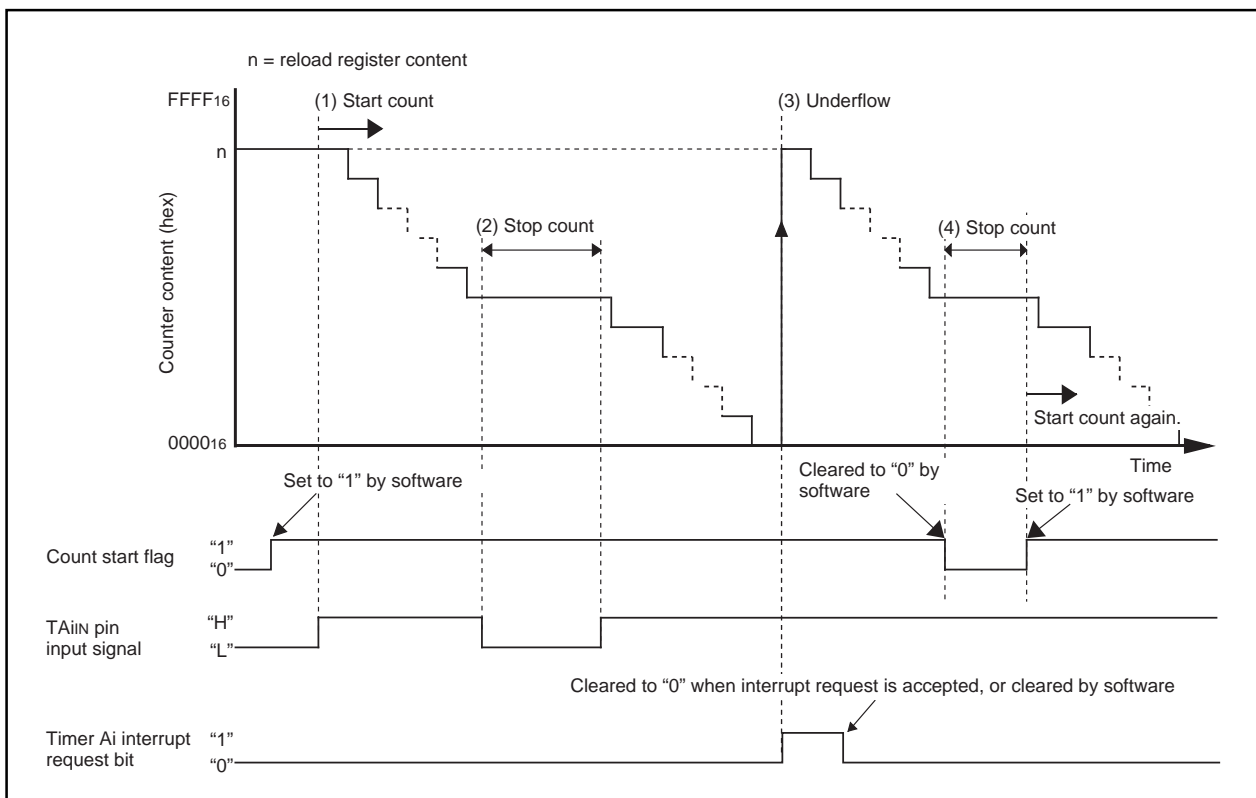
**Table 2.2.2. Chosed functions**

Item	Set-up	
Count source	○	Internal count source( $f_1 / f_8 / f_{32} / f_{c32}$ )
Pulse output function	○	No pulses output
		Pulses output
Gate function		No gate function
		Performs count only for the period in which the TAIin pin is at "L" level
	○	Performs count only for the period in which the TAIin pin is at "H" level

- Operation
- (1) When the count start flag is set to "1" and the TAIin pin inputs at "H" level, the counter performs a down count on the count source.
  - (2) When the TAIin pin inputs at "L" level, the counter holds its value and stops.
  - (3) If an underflow occurs, the content of the reload register is reloaded and the count continues. At this time, the timer Ai interrupt request bit goes to "1".
  - (4) Setting the count start flag to "0" causes the counter to hold its value and to stop.

Note

- Make the pulse width of the signal input to the TAIin pin not less than two cycles of the count source.



**Figure 2.2.8. Operation timing of timer mode, gate function selected**

## Timer A

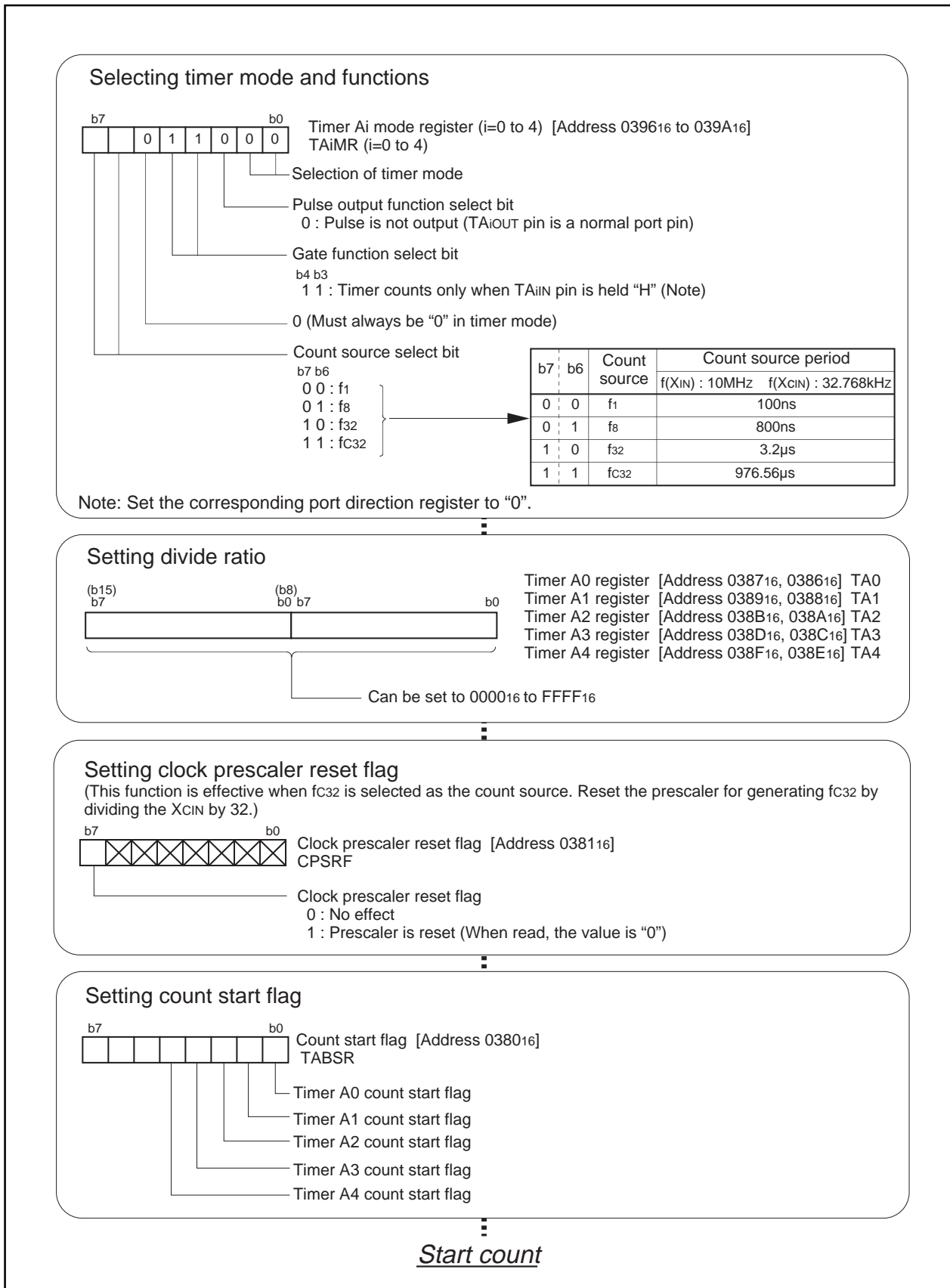


Figure 2.2.9. Set-up procedure of timer mode, gate function selected

## Timer A

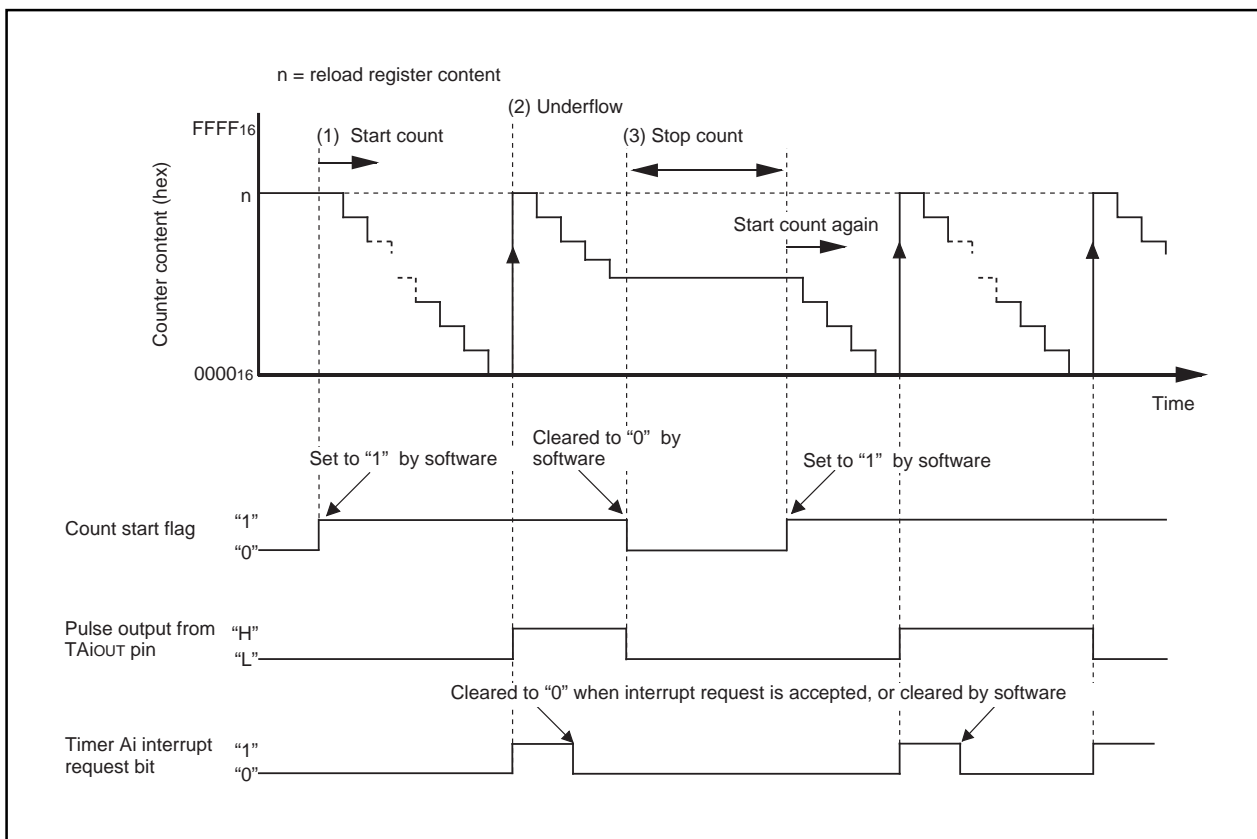
## 2.2.4 Operation of Timer A (timer mode, pulse output function selected)

In timer mode, choose functions from those listed in Table 2.2.3. Operations of the circled items are described below. Figure 2.2.10 shows the operation timing, and Figure 2.2.11 shows the set-up procedure.

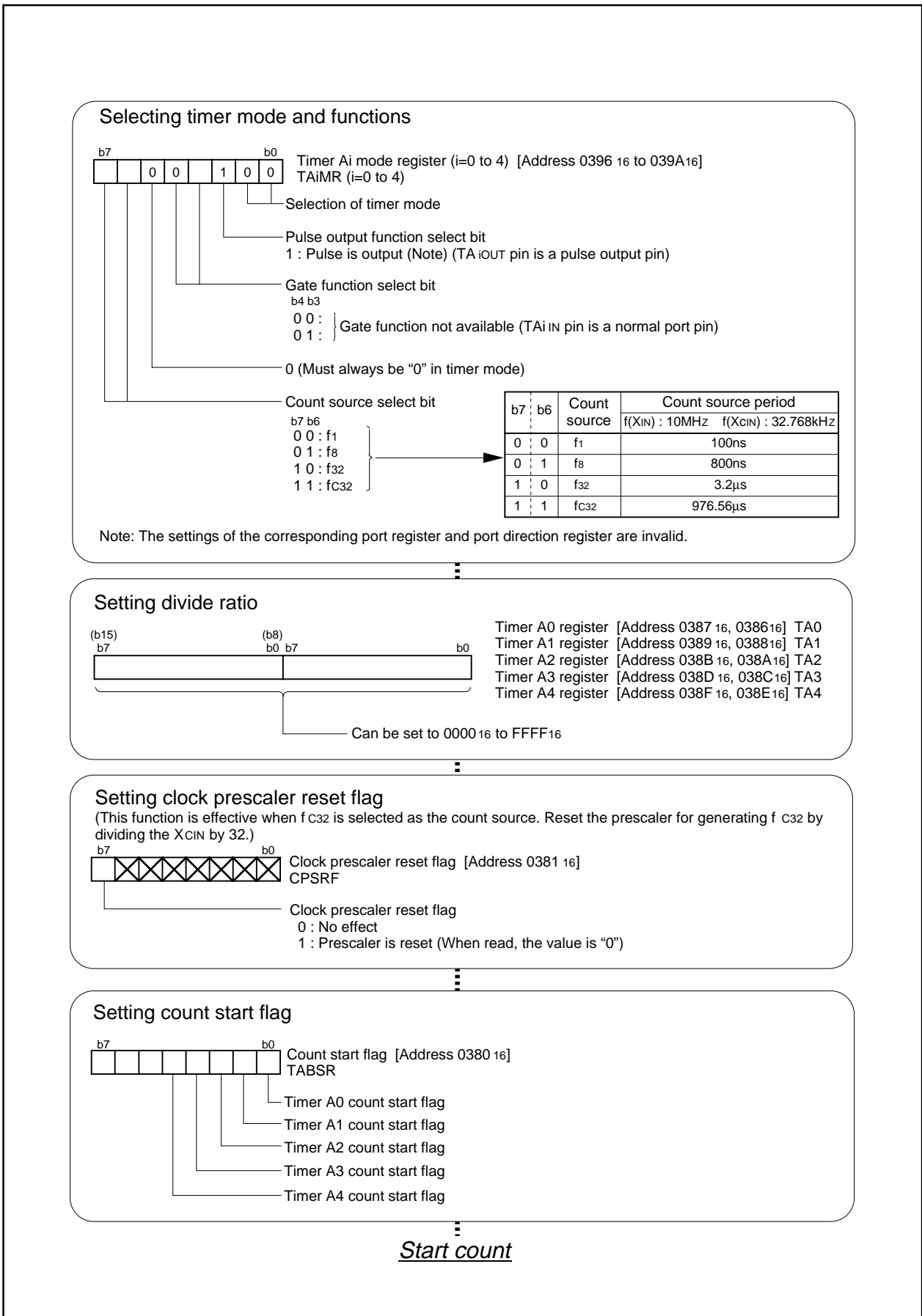
**Table 2.2.3. Chosed functions**

Item	Set-up	
Count source	O	Internal count source( $f_1 / f_8 / f_{32} / f_{c32}$ )
Pulse output function		No pulses output
	O	Pulses output
Gate function	O	No gate function
		Performs count only for the period in which the TAIIN pin is at "L" level
		Performs count only for the period in which the TAIIN pin is at "H" level

- Operation
- (1) Setting the count start flag to "1" causes the counter to perform a down count on the count source.
  - (2) If an underflow occurs, the content of the reload register is reloaded and the count continues. At this time, the timer Ai interrupt request bit goes to "1". Also, the output polarity of the TAIOUT pin reverses.
  - (3) Setting the count start flag to "0" causes the counter to hold its value and to stop. Also, the TAIOUT pin outputs an "L" level.



**Figure 2.2.10. Operation timing of timer mode, pulse output function selected**



### 2.2.5 Operation of Timer A (event counter mode, reload type selected)

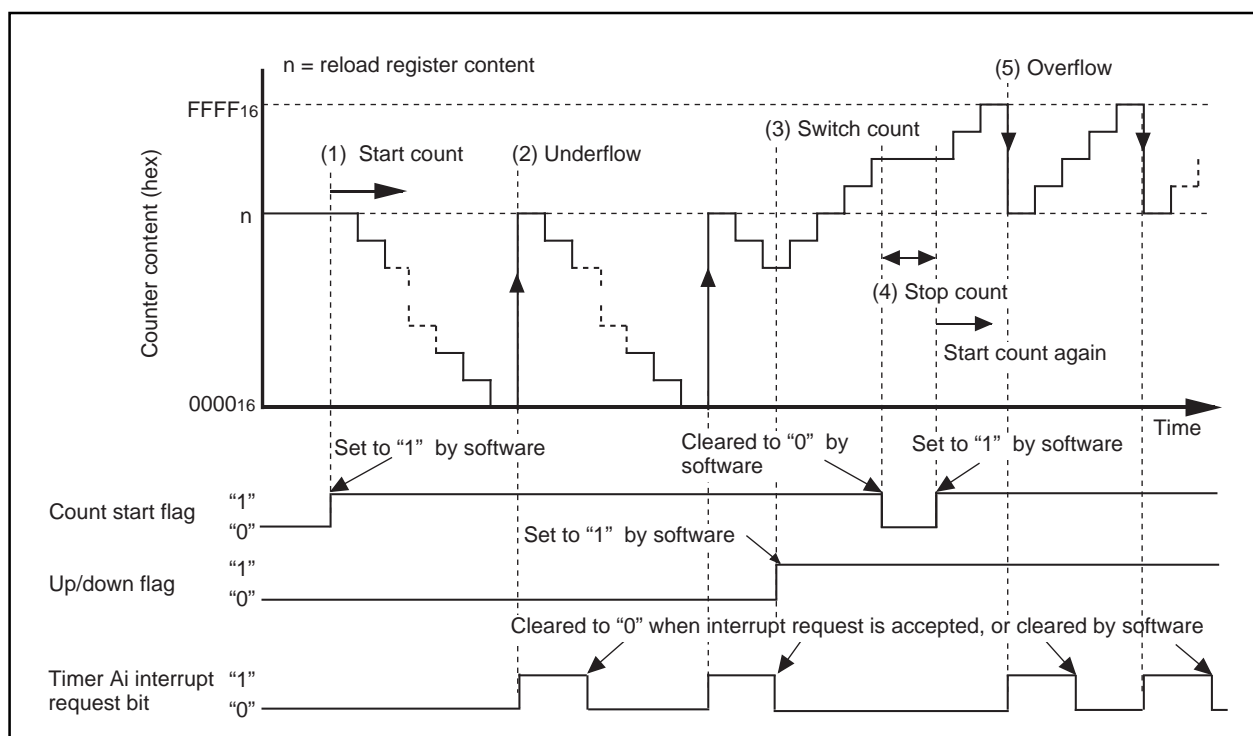
In event counter mode, choose functions from those listed in Table 2.2.4. Operations of the circled items are described below. Figure 2.2.12 shows the operation timing, and Figure 2.2.13 shows the set-up procedure.

**Table 2.2.4. Chosed functions**

Item	Set-up	Item	Set-up
Count source	○ Input signal to TAIin (counting falling edges)	Pulse output function	○ No pulses output
	Input signal to TAIin (counting rising edges)		Pulses output
	Timer overflow (TB2/TAj overflow)	Count operation type	○ Reload type
			Free-run type
		Factor for switching between up and down	○ Content of up/down flag
			Input signal to TAIout

Note:  $j = i - 1$ , but  $j = 4$  when  $i = 0$ .

- Operation
- (1) Setting the count start flag to "1" causes the counter to count the falling edges of the count source.
  - (2) If an underflow occurs, the content of the reload register is reloaded, and the count continues. At this time, the timer Ai interrupt request bit goes to "1".
  - (3) If switching from an up count to a down count or vice versa while a count is in progress, the switch takes effect from the next effective edge of the count source.
  - (4) Setting the count start flag to "0" causes the counter to hold its value and to stop.
  - (5) If an overflow occurs, the content of the reload register is reloaded, and the count continues. At this time, the timer Ai interrupt request bit goes to "1".



**Figure 2.2.12. Operation timing of event counter mode, reload type selected**

## Timer A

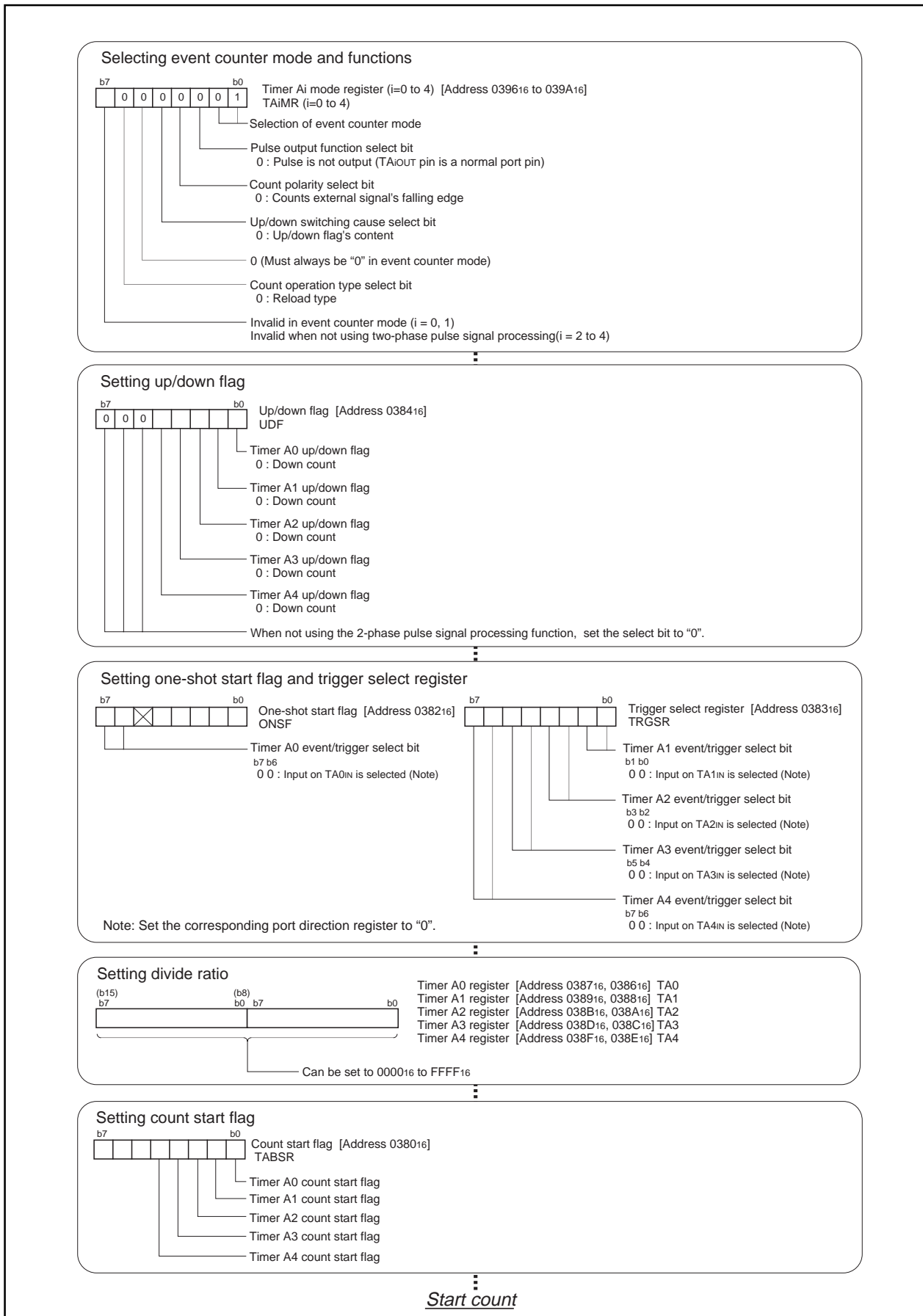


Figure 2.2.13. Set-up procedure of event counter mode, reload type selected



### 2.2.6 Operation of Timer A (event counter mode, free run type selected)

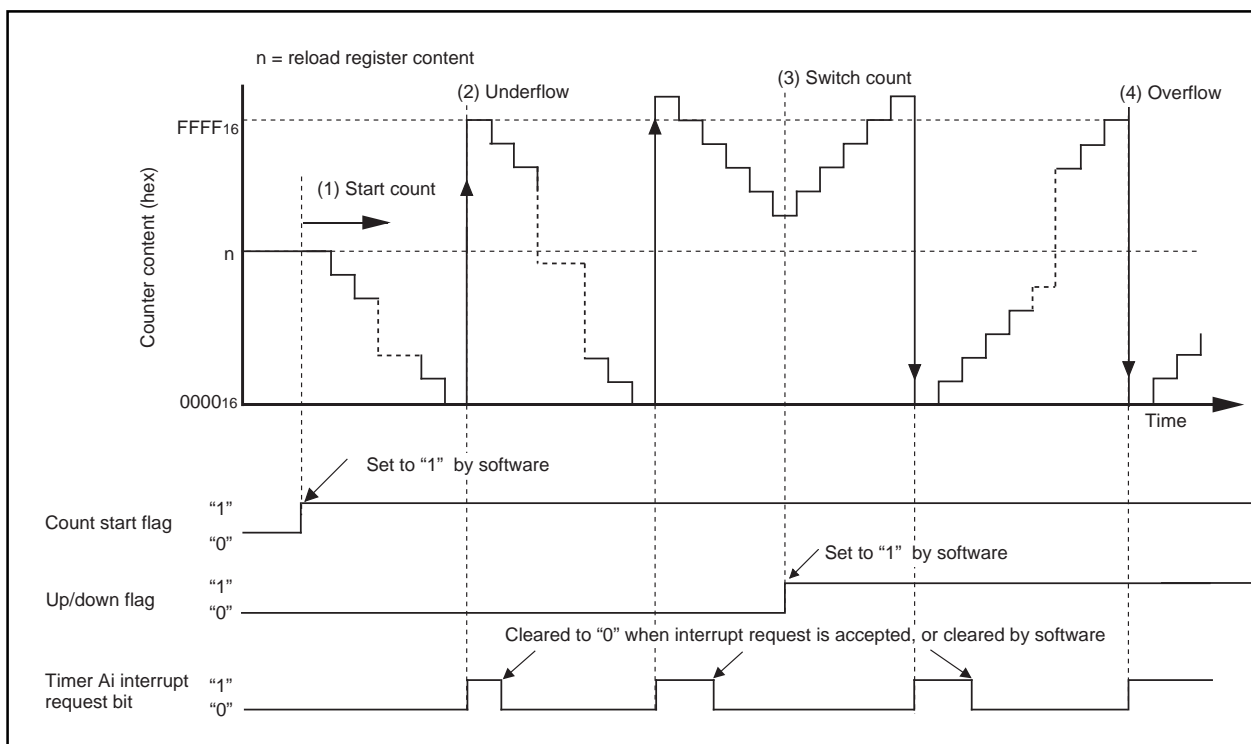
In event counter mode, choose functions from those listed in Table 2.2.5. Operations of the circled items are described below. Figure 2.2.14 shows the operation timing, and Figure 2.2.15 shows the set-up procedure.

**Table 2.2.5. Chosed functions**

Item	Set-up		Item	Set-up	
Count source	○	Input signal to TAIin (counting falling edges)	Pulse output function	○	No pulses output
		Input signal to TAIin (counting rising edges)			Pulses output
	Timer overflow (TB2/TAj overflow)		Count operation type	○	Free-run type
				○	Content of up/down flag
					Input signal to TAIout

Note:  $j = i - 1$ , but  $j = 4$  when  $i = 0$

- Operation
- (1) Setting the count start flag to "1" causes the counter to count the falling edges of the count source.
  - (2) Even if an underflow occurs, the content of the reload register is not reloaded, but the count continues. At this time, the timer Ai interrupt request bit goes to "1".
  - (3) If switching from an up count to a down count or vice versa while a count is in progress, the switch takes effect from the next effective edge of the count source.
  - (4) Even if an overflow occurs, the content of the reload register is not reloaded, but the count continues. At this time, the timer Ai interrupt request bit goes to "1".



**Figure 2.2.14. Operation timing of event counter mode, free run type selected**

## Timer A

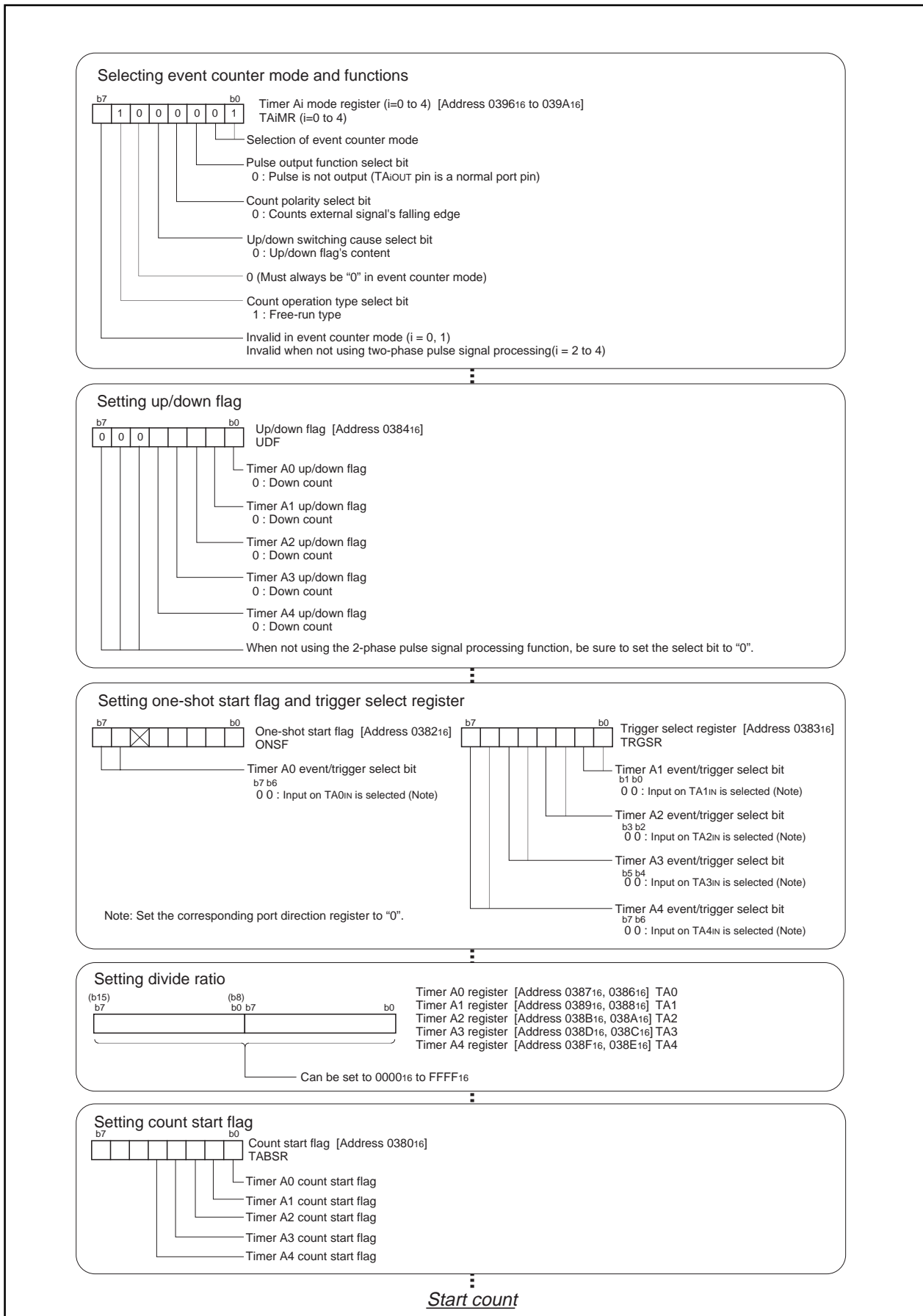


Figure 2.2.15. Set-up procedure of event counter mode, free run type selected

### 2.2.7 Operation of timer A (2-phase pulse signal process in event counter mode, normal mode selected)

In processing 2-phase pulse signals in event counter mode, choose functions from those listed in Table 2.2.6. Operations of the circled items are described below. Figure 2.2.16 shows the operation timing, and Figure 2.2.17 shows the set-up procedure.

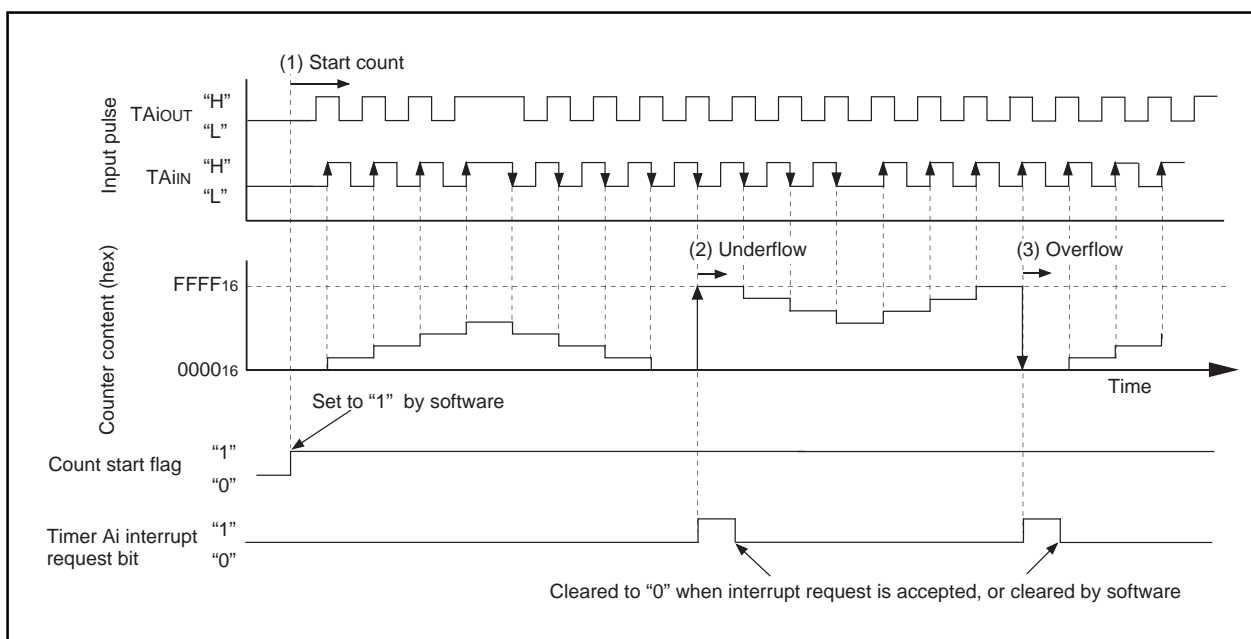
**Table 2.2.6. Chosed functions**

Item	Set-up
Count operation type	Reload type
	○ Free run type
2-phase pulses process (Note)	○ Normal processing
	4-multiplication processing

Note: Timer A3 alone can be selected. Timer A2 is solely used for normal processes, and timer A4 is solely used for 4 multiplication processes.

- Operation
- (1) Setting the count start flag to "1" causes the counter to count effective edges of the count source.
  - (2) Even if an underflow occurs, the content of the reload register is not reloaded, but the count continues. At this time, the timer Ai interrupt request bit goes to "1".
  - (3) Even if an overflow occurs, the content of the reload register is not reloaded, but the count continues. At this time, the timer Ai interrupt request bit goes to "1".

- Note
- The up count or down count conditions are as follows:
    - If a rising edge is present at the TAIiN pin when the input signal level to the TAIoUT pin is "H", an up count is performed.
    - If a falling edge is present at the TAIiN pin when the input signal level to the TAIoUT pin is "H", a down count is performed.



**Figure 2.2.16. Operation timing of 2-phase pulse signal process in event counter mode, normal mode selected**



## 2.2.8 Operation of timer A (2-phase pulse signal process in event counter mode, multiply-by-4 mode selected)

In processing 2-phase pulse signals in event counter mode, choose functions from those listed in Table 2.2.7. Operations of the circled items are described below. Figure 2.2.18 shows the operation timing, and Figure 2.2.19 shows the set-up procedure.

**Table 2.2.7. Chosed functions**

Item	Set-up		Item	Set-up	
Count operation type		Reload type	Processing 2 phase pulses (Note)		Normal processing
	○	Free run type		○	4-multiplication processing

Note: Timer A3 alone can be selected. Timer A2 is solely used for normal processes, and timer A4 is solely used for 4-multiplication processes.

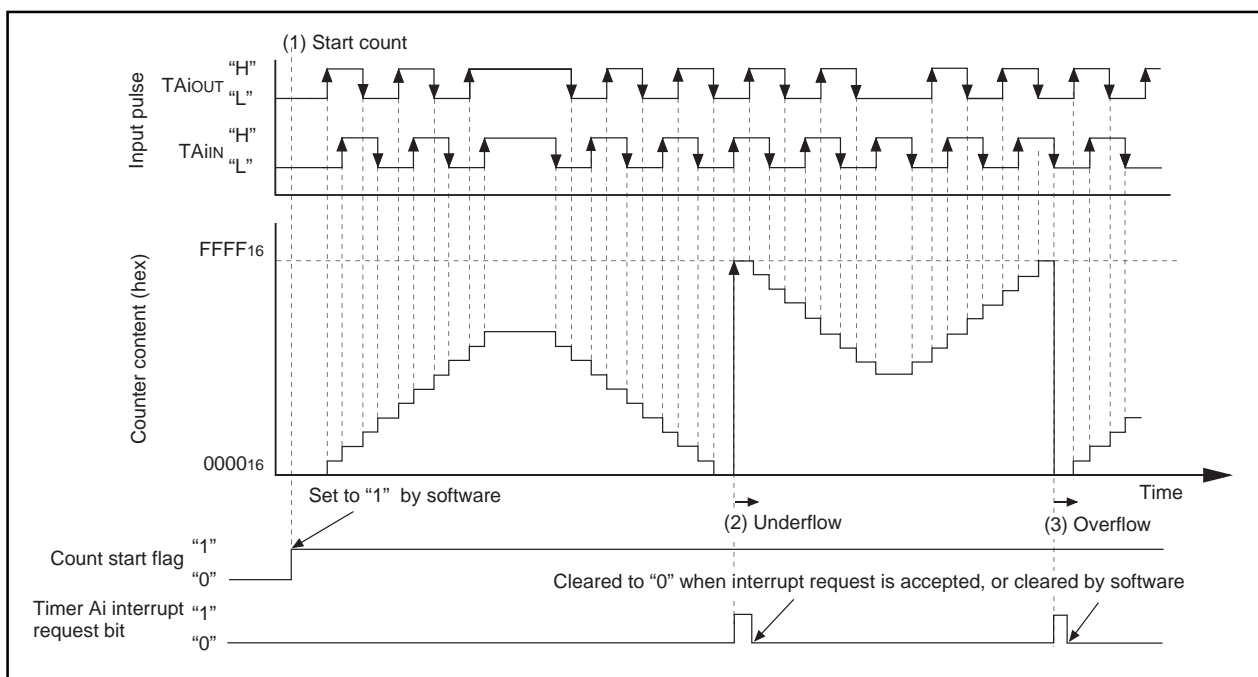
- Operation
- (1) Setting the count start flag to "1" causes the counter to count effective edges of the count source.
  - (2) Even if an underflow occurs, the content of the reload register is not reloaded, but the count continues. At this time, the interrupt request bit goes to "1".
  - (3) Even if an overflow occurs, the content of the reload register is not reloaded, but the count continues. At this time, the interrupt request bit goes to "1".

Note

- The up count or down count conditions are as follows:

**Table 2.2.8. The up count or down count conditions**

	Input signal to the TAIOUT pin	Input signal to the TAIIN pin		Input signal to the TAIOUT pin	Input signal to the TAIIN pin
Up count	"H" level	Rising	Down count	"H" level	Falling
	"L" level	Falling		"L" level	Rising
	Rising	"L" level		Rising	"H" level
	Falling	"H" level		Falling	"L" level



**Figure 2.2.18. Operation timing of 2-phase pulse signal process in event counter mode, multiply-by-4 mode selected**

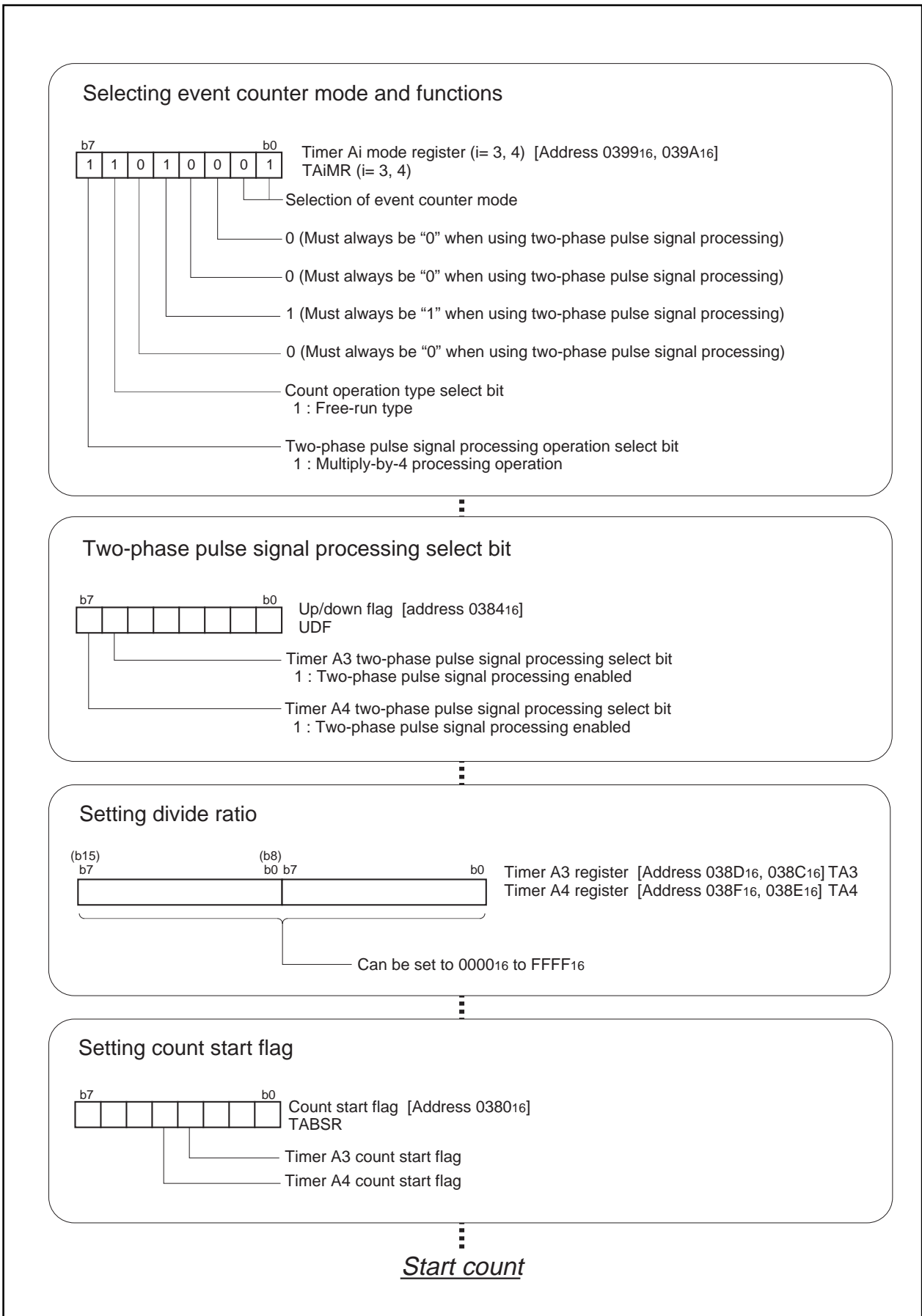


Figure 2.2.19. Set-up procedure of 2-phase pulse signal process in event counter mode, multiply-by-4 mode selected

## 2.2.9 Operation of Timer A (one-shot timer mode)

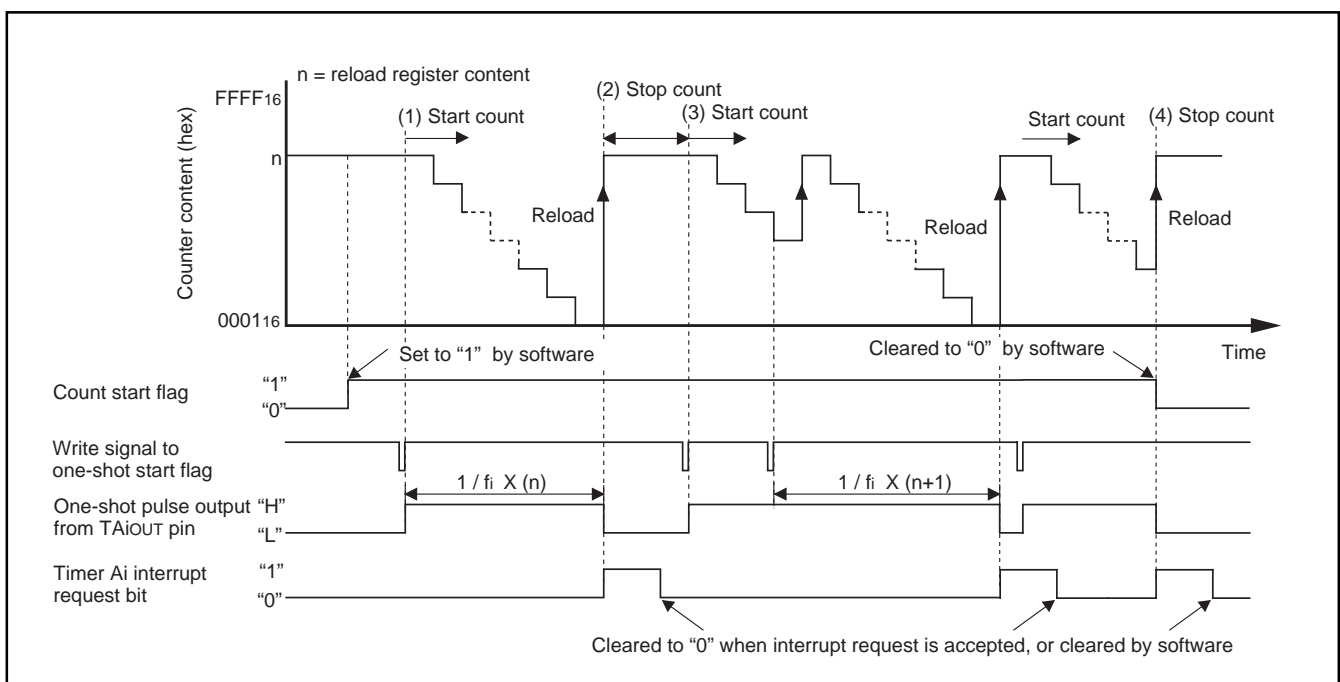
In one-shot timer mode, choose functions from those listed in Table 2.2.9. Operations of the circled items are described below. Figure 2.2.20 shows the operation timing, and Figure 2.2.21 shows the set-up procedure.

**Table 2.2.9. Chosed functions**

Item	Set-up	
Count source	○	Internal count source ( $f_1 / f_8 / f_{32} / f_{c32}$ )
Pulse output function		No pulses output
	○	Pulses output
Count start condition		External trigger input (falling edge of input signal to the TAIIN pin)
		External trigger input (rising edge of input signal to the TAIIN pin)
		Timer overflow (TB2/TAj/TAK overflow)
	○	Writing "1" to the one-shot start flag

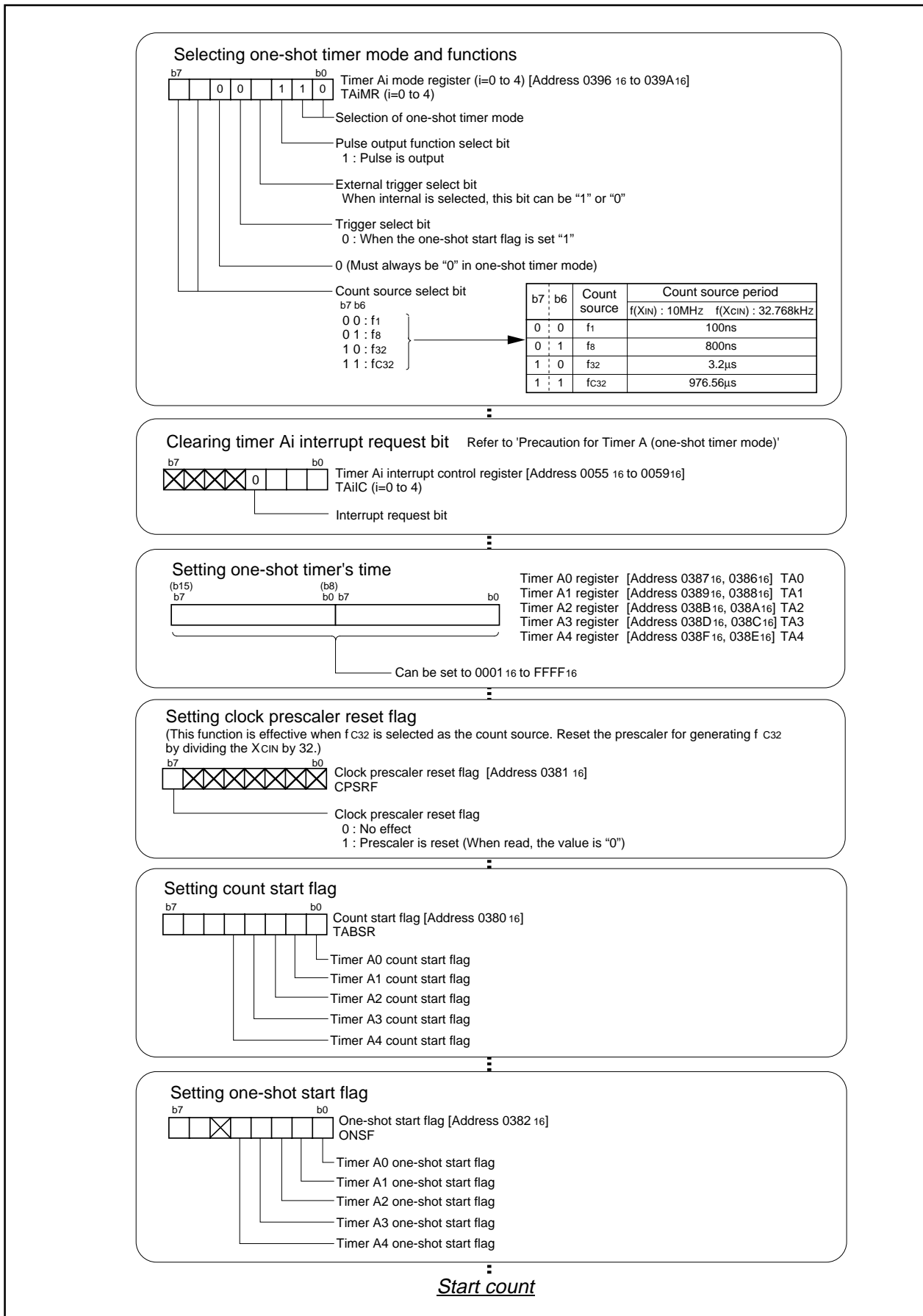
Note:  $j = i - 1$ , but  $j = 4$  when  $i = 0$ ;  $k = i + 1$ , but  $k = 0$  when  $i = 4$ .

- Operation
- (1) Setting the one-shot start flag to "1" with the count start flag set to "1" causes the counter to perform a down count on the count source. At this time, the TAIOUT pin outputs an "H" level.
  - (2) The instant the value of the counter becomes "0000<sub>16</sub>", the TAIOUT pin outputs an "L" level, and the counter reloads the content of the reload register and stops counting. At this time, the timer Ai interrupt request bit goes to "1".
  - (3) If a trigger occurs while a count is in progress, the counter reloads the value in the reload register again and continues counting. The reload timing is in step with the next count source input after the trigger.
  - (4) Setting the count start flag to "0" causes the counter to stop and to reload the content of the reload register. Also, the TAIOUT pin outputs an "L" level. At this time, the timer Ai interrupt request bit goes to "1".



**Figure 2.2.20. Operation timing of one-shot mode**

Timer A





## Timer A

## 2.2.10 Operation of Timer A (one-shot timer mode, external trigger selected)

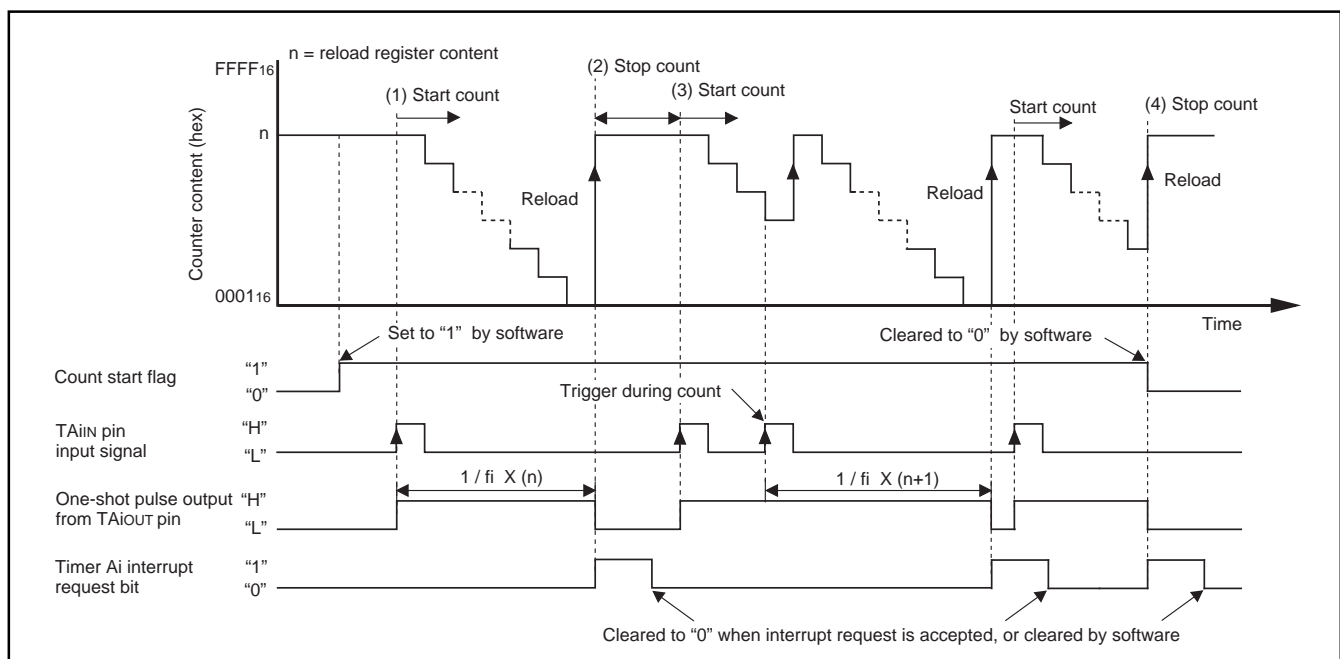
In one-shot timer mode, choose functions from those listed in Table 2.2.10. Operations of the circled items are described below. Figure 2.2.22 shows the operation timing, and Figure 2.2.23 shows the set-up procedure.

**Table 2.2.10. Chosed functions**

Item	Set-up	
Count source	<b>○</b>	Internal count source ( $f_1 / f_8 / f_{32} / f_{c32}$ )
Pulse output function		No pulses output
	<b>○</b>	Pulses output
Count start condition		External trigger input (falling edge of input signal to the TAIin pin)
	<b>○</b>	External trigger input (rising edge of input signal to the TAIin pin)
		Timer overflow (TB2/TAj/TAk overflow)
		Writing "1" to the one-shot start flag

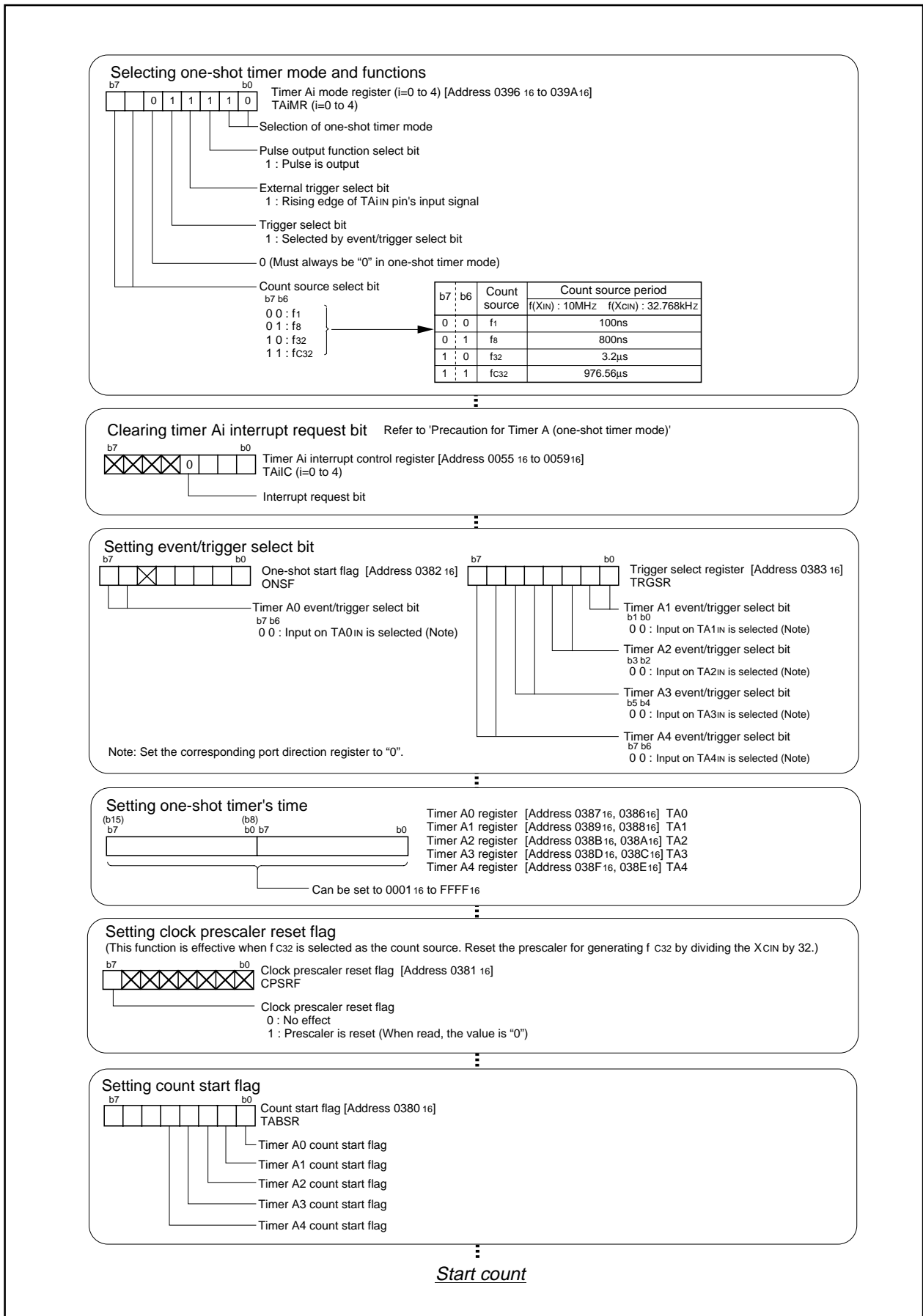
Note:  $j = i - 1$ , but  $j = 4$  when  $i = 0$ ;  $k = i + 1$ , but  $k = 0$  when  $i = 4$ .

- Operation
- (1) If the TAIin pin input level changes from "L" to "H" with the count start flag set to "1", the counter performs a down count on the count source. At this time, the TAIOUT pin output level goes to "H" level.
  - (2) If the value of the counter becomes "0000<sub>16</sub>", the TAIOUT pin outputs an "L" level, and the counter reloads the content of the reload register and stops counting. At this time, the timer Ai interrupt request bit goes to "1".
  - (3) If a trigger occurs while a count is in progress, the counter reloads the value of the reload register again and continues counting. The reload timing is in step with the next count source input after the trigger.
  - (4) Setting the count start flag to "0" causes the counter to stop and to reload the content of the reload register. Also, the TAIOUT pin outputs an "L" level. At this time, the timer Ai interrupt request bit goes to "1".



**Figure 2.2.22. Operation timing of one-shot mode, external trigger selected**

Timer A



### 2.2.11 Operation of Timer A (pulse width modulation mode, 16-bit PWM mode selected)

In pulse width modulation mode, choose functions from those listed in Table 2.2.11. Operations of the circled items are described below. Figure 2.2.24 shows the operation timing, and Figure 2.2.25 shows the set-up procedure.

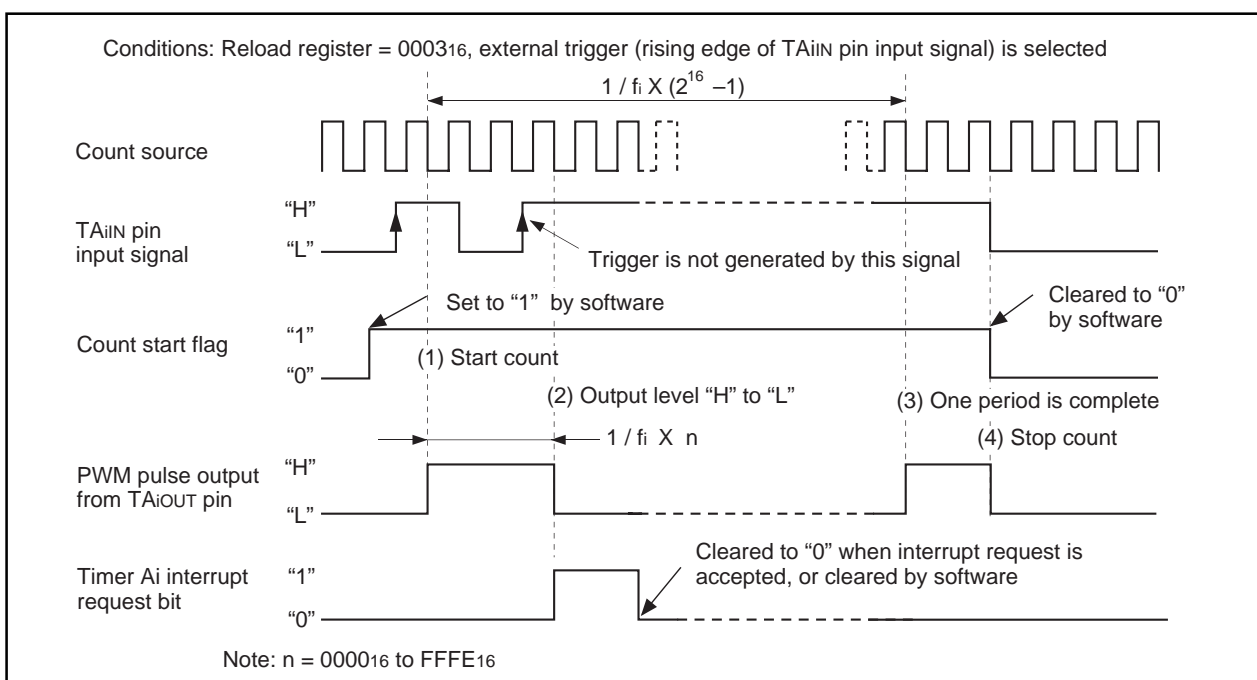
**Table 2.2.11. Chosed functions**

Item	Set-up	
Count source	○	Internal count source ( $f_1 / f_8 / f_{32} / f_{c32}$ )
PWM mode	○	16-bit PWM
		8-bit PWM
Count start condition		External trigger input (falling edge of input signal to the TAIIN pin)
	○	External trigger input (rising edge of input signal to the TAIIN pin)
		Timer overflow (TB2/TAj/TAK overflow)

Note:  $j = i - 1$ , but  $j = 4$  when  $i = 0$ ;  $k = i + 1$ , but  $k = 0$  when  $i = 4$ .

- Operation
- (1) If the TAIIN pin input level changes from "L" to "H" with the count start flag set to "1", the counter performs a down count on the count source. Also, the TAIOUT pin outputs an "H" level.
  - (2) The TAIOUT pin output level changes from "H" to "L" when a set time period elapses. At this time, the timer Ai interrupt request bit goes to "1".
  - (3) The counter reloads the content of the reload register every time PWM pulses are output for one cycle, and continues counting.
  - (4) Setting the count start flag to "0" causes the counter to hold its value and to stop. Also, the TAIOUT outputs an "L" level.

- Note
- The period of PWM pulses becomes  $(2^{16} - 1)/f_i$ , and the "H" level pulse width becomes  $n/f_i$ . If the timer Ai register is set to "0000<sub>16</sub>", the pulse width modulator does not work, and the TAIOUT pin output level remains at "L".
- ( $f_i$  : frequency of the count source  $f_1, f_8, f_{32}, f_{c32}$ ;  $n$  : value of the timer)



**Figure 2.2.24. Operation timing of pulse width modulation mode, 16-bit PWM mode selected**

Timer A

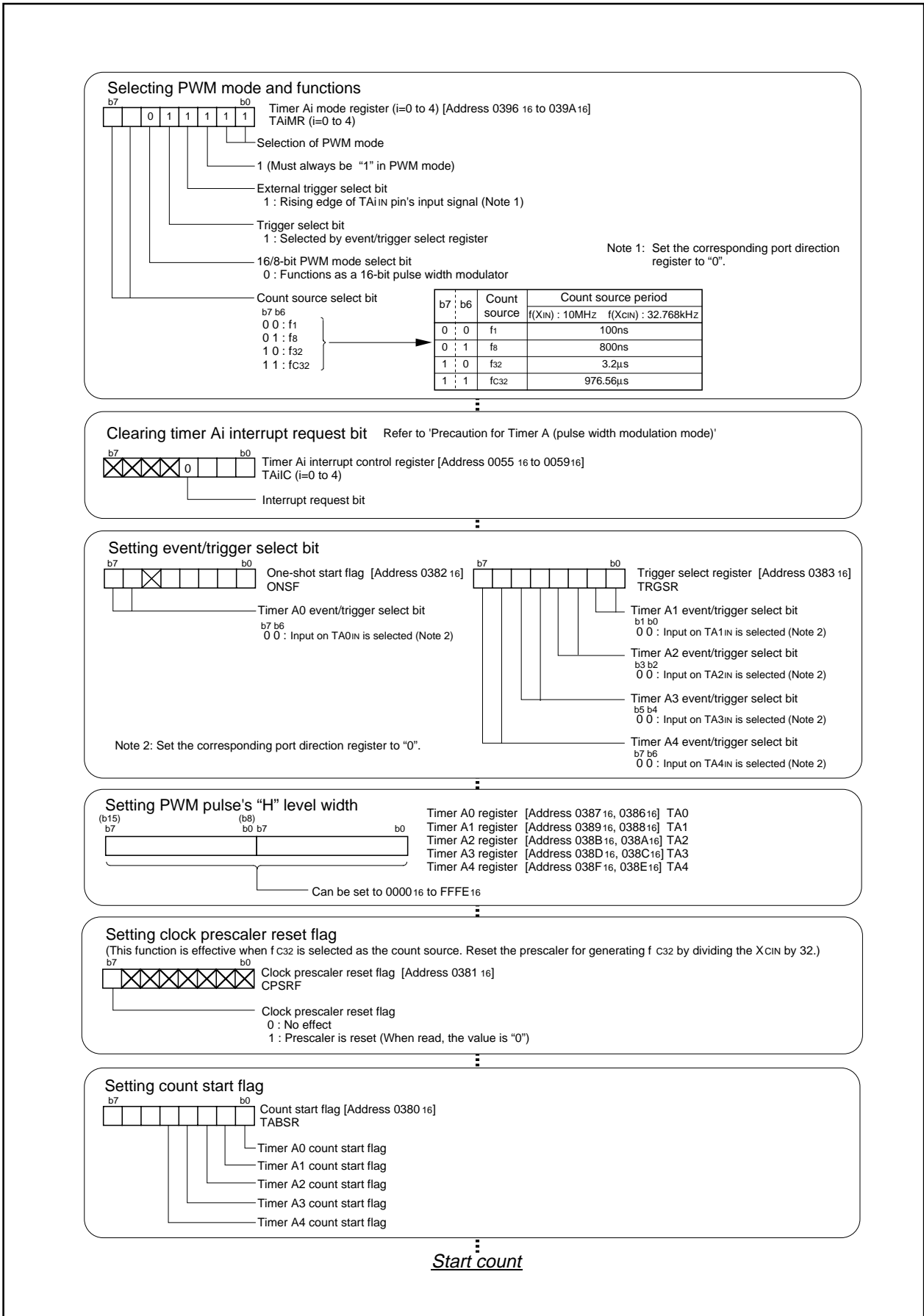


Figure 2.2.25. Set-up procedure of pulse width modulation mode, 16-bit PWM mode selected

## 2.2.12 Operation of Timer A (pulse width modulation mode, 8-bit PWM mode selected)

In pulse width modulation mode, choose functions from those listed in Table 2.2.12. Operations of the circled items are described below. Figure 2.2.26 shows the operation timing, and Figure 2.2.27 shows the set-up procedure.

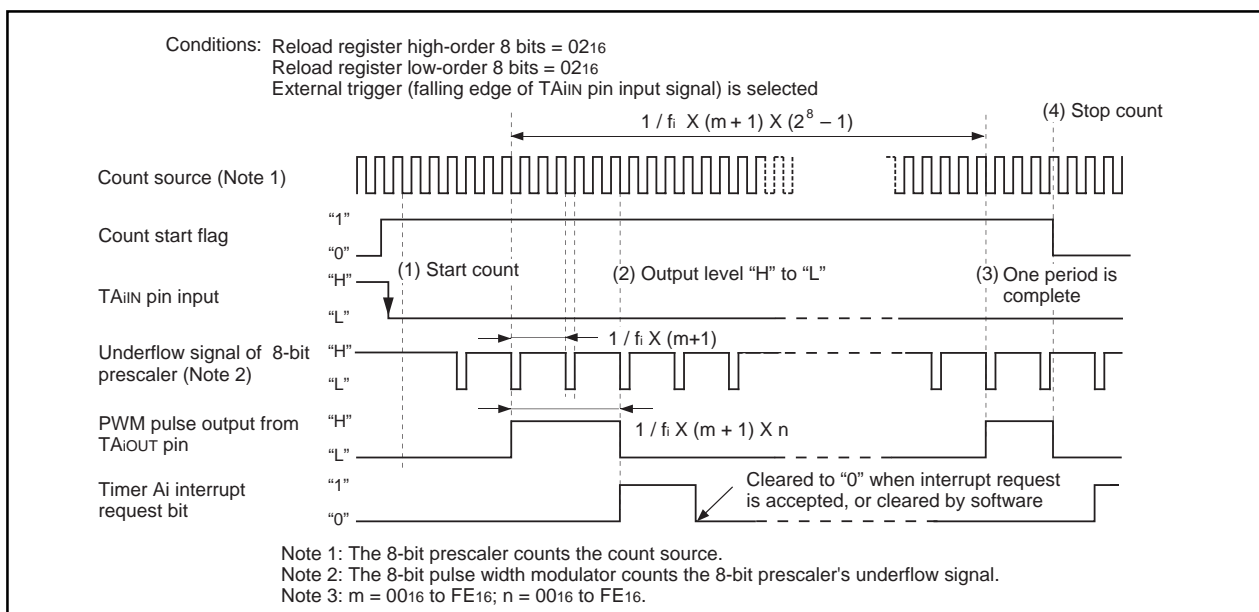
**Table 2.2.12. Chosed functions**

Item	Set-up
Count source	○ Internal count source ( $f_1 / f_8 / f_{32} / f_{c32}$ )
PWM mode	16-bit PWM
	○ 8-bit PWM
Count start condition	○ External trigger input (falling edge of input signal to the TAIIN pin)
	External trigger input (rising edge of input signal to the TAIIN pin)
	Timer overflow (TB2/TAj/TAK overflow)

Note:  $j = i - 1$ , but  $j = 4$  when  $i = 0$ ;  $k = i + 1$ , but  $k = 0$  when  $i = 4$ .

- Operation
- (1) If the TAIIN pin input level changes from "H" to "L" with the count start flag set to "1", the counter performs a down count on the count source. Also, the TAIOUT pin outputs an "H" level.
  - (2) The TAIOUT pin output level changes from "H" to "L" when a set time period elapses. At this time, the timer Ai interrupt request bit goes to "1".
  - (3) The counter reloads the content of the reload register every time PWM pulses are output for one cycle, and continues counting.
  - (4) Setting the count start flag to "0" causes the counter to hold its value and to stop. Also, the TAIOUT pin outputs an "L" level.

- Note
- The period of PWM pulses becomes  $(m + 1) \times (2^8 - 1) / f_i$ , and the "H" level pulse width becomes  $n \times (m + 1) / f_i$ . If "0016" is set in the eight higher-order bits of the timer Ai register, the pulse width modulator does not work, and the TAIOUT pin output level remains at "L".  
( $f_i$  : frequency of the count source  $f_1, f_8, f_{32}, f_{c32}$ ;  $n$  : value of the timer)
  - When a trigger is generated, the TAIOUT pin outputs "L" level of same amplitude as "H" level of the set PWM pulse, after which it starts PWM pulse output.



**Figure 2.2.26. Operation timing of pulse width modulation mode, with 8-bit PWM mode selected**

Timer A

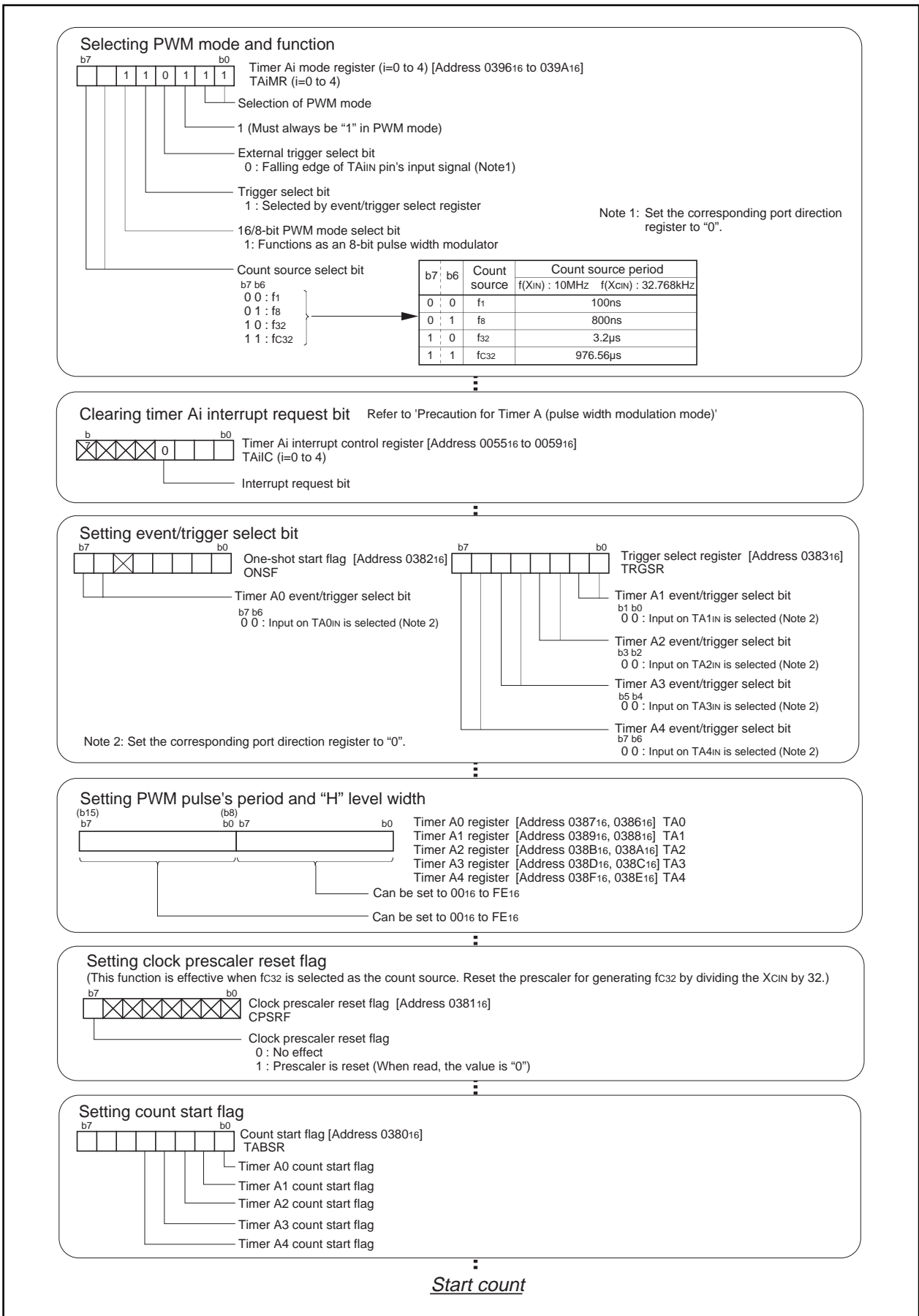
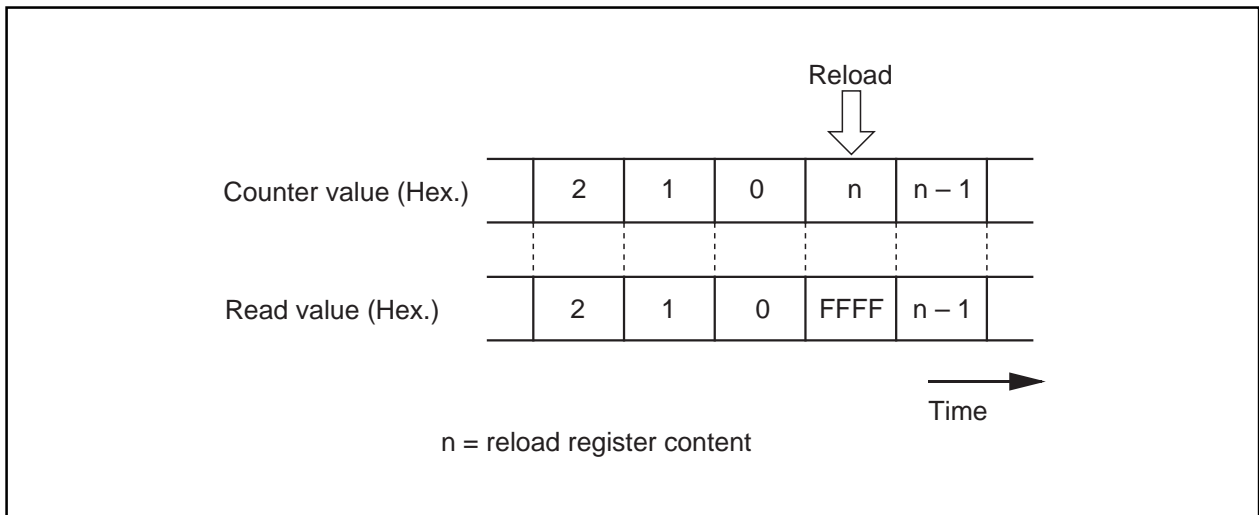


Figure 2.2.27. Set-up procedure of pulse width modulation mode, 8-bit PWM mode selected

**2.2.13 Precautions for Timer A (timer mode)**

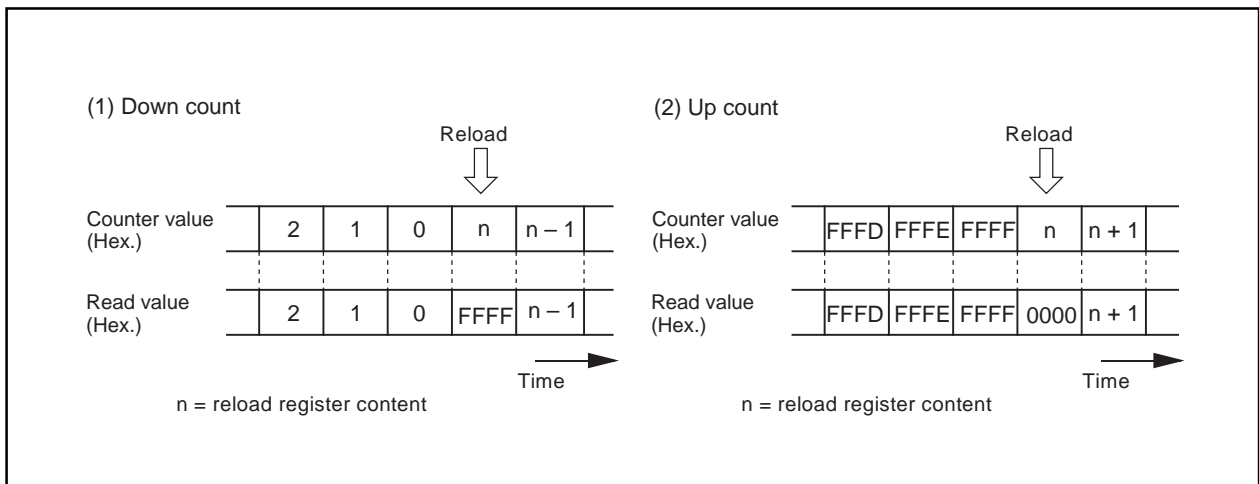
- (1) To clear reset, the count start flag is set to "0". Set a value in the timer Ai register, then set the flag to "1".
- (2) Reading the timer Ai register while a count is in progress allows reading, with arbitrary timing, the value of the counter. Reading the timer Ai register with the reload timing shown in Figure 2.2.28 gets "FFFF<sub>16</sub>". 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.



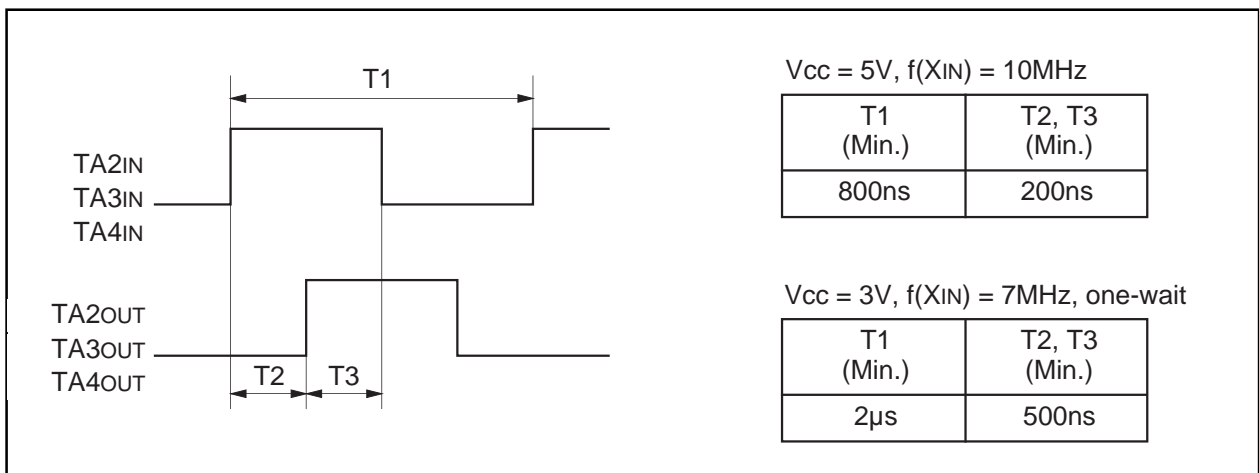
**Figure 2.2.28. Reading timer Ai register**

**2.2.14 Precautions for Timer A (event counter mode)**

- (1) To clear reset, the count start flag is set to "0". Set a value in the timer Ai register, then set the flag to "1".
- (2) 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 shown in Figure 2.2.29 gets "FFFF16" by underflow or "000016" 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.
- (3) Please note the standards for the differences between the 2 pulses used in the 2-phase pulse signals input signals to the TAiIN pin and TAiOUT pin (i = 2, 3, 4), as shown in Figure 2.2.30.
- (4) When free run type is selected, if count is stopped, set a value in the timer Ai register again.



**Figure 2.2.29. Reading timer Ai register**



**Figure 2.2.30. Standard of 2-phase pulses**



### 2.2.15 Precautions for Timer A (one-shot timer mode)

- (1) To clear reset, the count start flag is set to "0". Set a value in the timer Ai register, then set the flag to "1".
- (2) Setting the count start flag to "0" while a count is in progress causes as follows:
  - The counter stops counting and a content of reload register is reloaded.
  - The TAIOUT pin outputs "L" level.
  - The interrupt request is generated and the timer Ai interrupt request bit goes to "1".
- (3) The output from the one-shot timer synchronizes with the count source generated internally. Therefore, when an external trigger has been selected, a delay of one cycle of the maximum count source occurs between the trigger input to the TAIIN pin and the one-shot timer output.
- (4) 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.
- (5) If a trigger occurs while a count is in progress, after the counter performs one down count following the reoccurrence of a trigger, the reload register contents are reloaded, and the count continues. To generate a trigger while a count is in progress, generate the second trigger after an elapse longer than one cycle of the timer's count source after the previous trigger occurred.

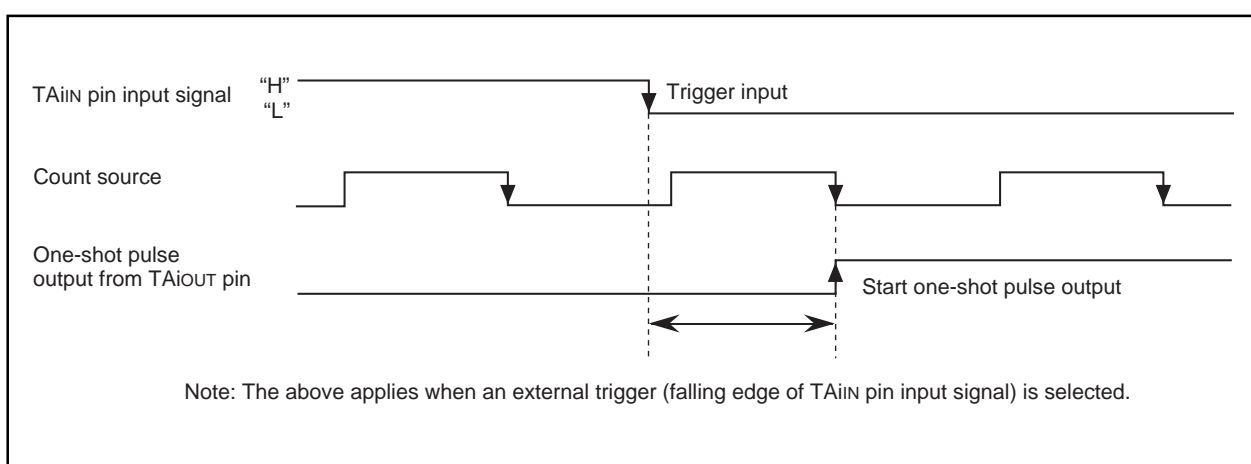


Figure 2.2.31. One-shot timer delay

### 2.2.16 Precautions for Timer A (pulse width modulation mode)

- (1) To clear reset, the count start flag is set to "0". Set a value in the timer Ai register, then set the flag to "1".
- (2) 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.
- (3) Setting the count start flag to "0" while PWM pulses are being output causes the counter to stop counting. If the TAIOUT pin is outputting an "H" level in this instance, the output level goes to "L", and the timer Ai interrupt request bit goes to "1". If the TAIOUT pin is outputting an "L" level in this instance, the level does not change, and the timer Ai interrupt request bit does not become "1".

## 2.3 Timer B

### 2.3.1 Overview

The following is an overview for timer B, a 16-bit timer.

#### (1) Mode

Timer B operates in one of three modes:

##### (a) Timer mode

The internal count source is counted.

- Operation in timer mode ..... P216

##### (b) Event counter mode

The number of pulses coming from outside and the number of the timer overflows are counted.

- Operation in event counter mode ..... P218

##### (c) Pulse period measurement/pulse width measurement mode

External pulse period or external pulse widths are measured. If pulse period measurement mode is selected, the periods of input pulses are continuously measured. If pulse width measurement mode is selected, widths of "H" level pulses and those of "L" level pulses are continuously measured.

- Operation in pulse period measurement mode ..... P220
- Operation in pulse width measurement mode ..... P222

#### (2) Count source

An internal count source can be selected from f1, f8, f32, and fc32. f1, f8, and f32 are clocks obtained by dividing the CPU main clock by 1, 8, and 32 respectively. fc32 is the clock obtained by dividing the CPU secondary clock by 32.

#### (3) Frequency division ratio

The frequency division ratio equals [the value set in the timer register + 1]. The counter underflows when a count source equal to a frequency division ratio is input, and an interrupt request occurs.

#### (4) Reading the timer

In timer mode or event counter mode, the count value at the time of reading the timer register will be read. Read the register in 16-bit increments. In both the pulse period measurement mode and pulse width measurement mode, an indeterminate value is read until the second effective edge is input after a count is started, otherwise, the measurement results are read.

#### (5) Writing to the timer

When writing to the timer register while a count is in progress, the value is written only to the reload register. When writing to the timer register while a count has stopped, the value is written both to the reload register and the count. Write the value in 16-bit increments. The timer register cannot be written to in either the pulse period measurement mode or the pulse width measurement mode.

## Timer B

**(6) Input to the timer and the direction register**

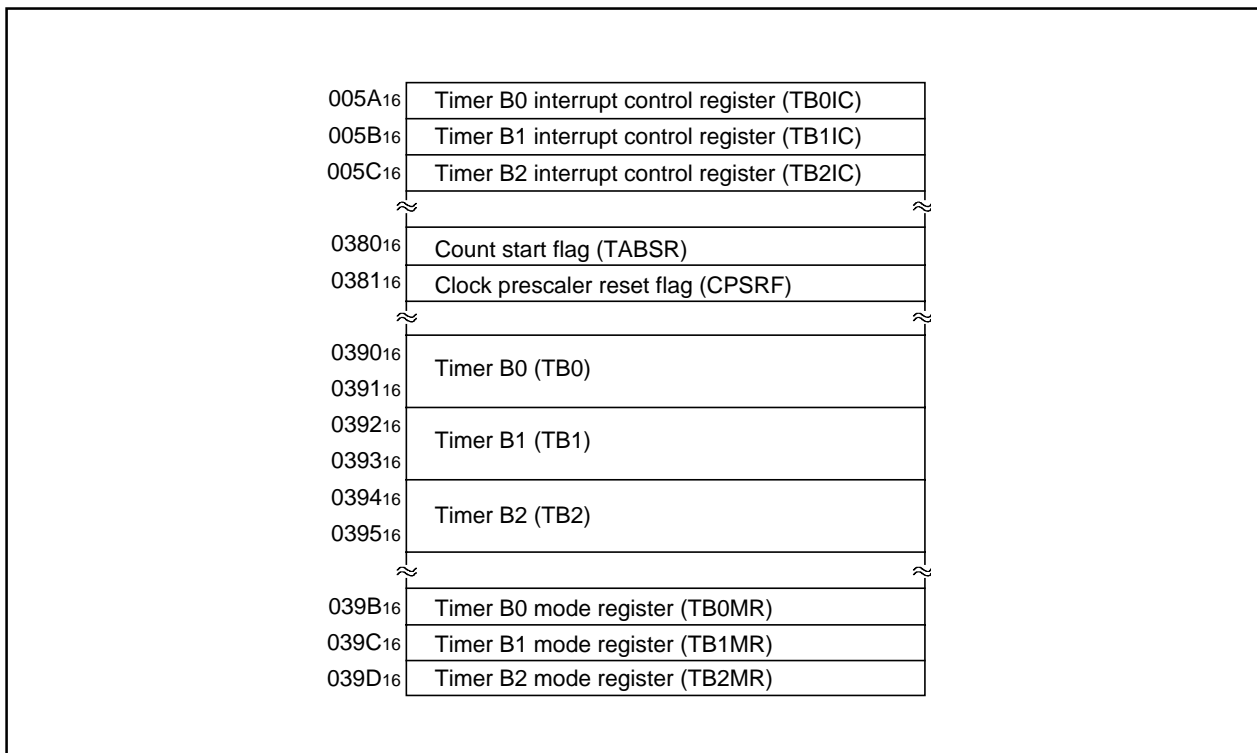
To input an external signal to the timer, set the direction register of the relevant port to input.

**(7) Pins related to timer B**

(a) TB0IN, TB1IN, TB2IN      Input pins to timer B.

**(8) Registers related to timer B**

Figure 2.3.1 shows the memory map of timer B-related registers. Figures 2.3.2 and 2.3.3 show timer B-related registers.



**Figure 2.3.1. Memory map of timer B-related registers**

## Timer B

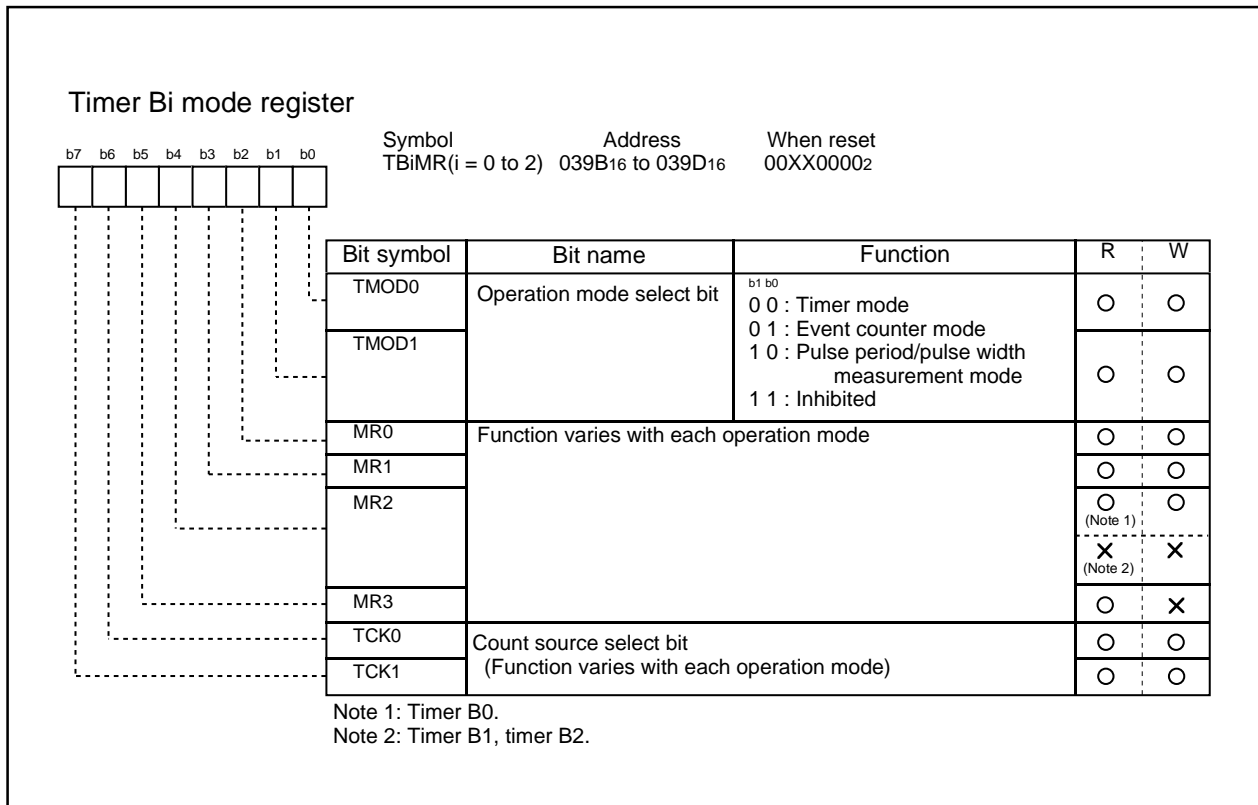


Figure 2.3.2. Timer B-related registers (1)

## Timer B

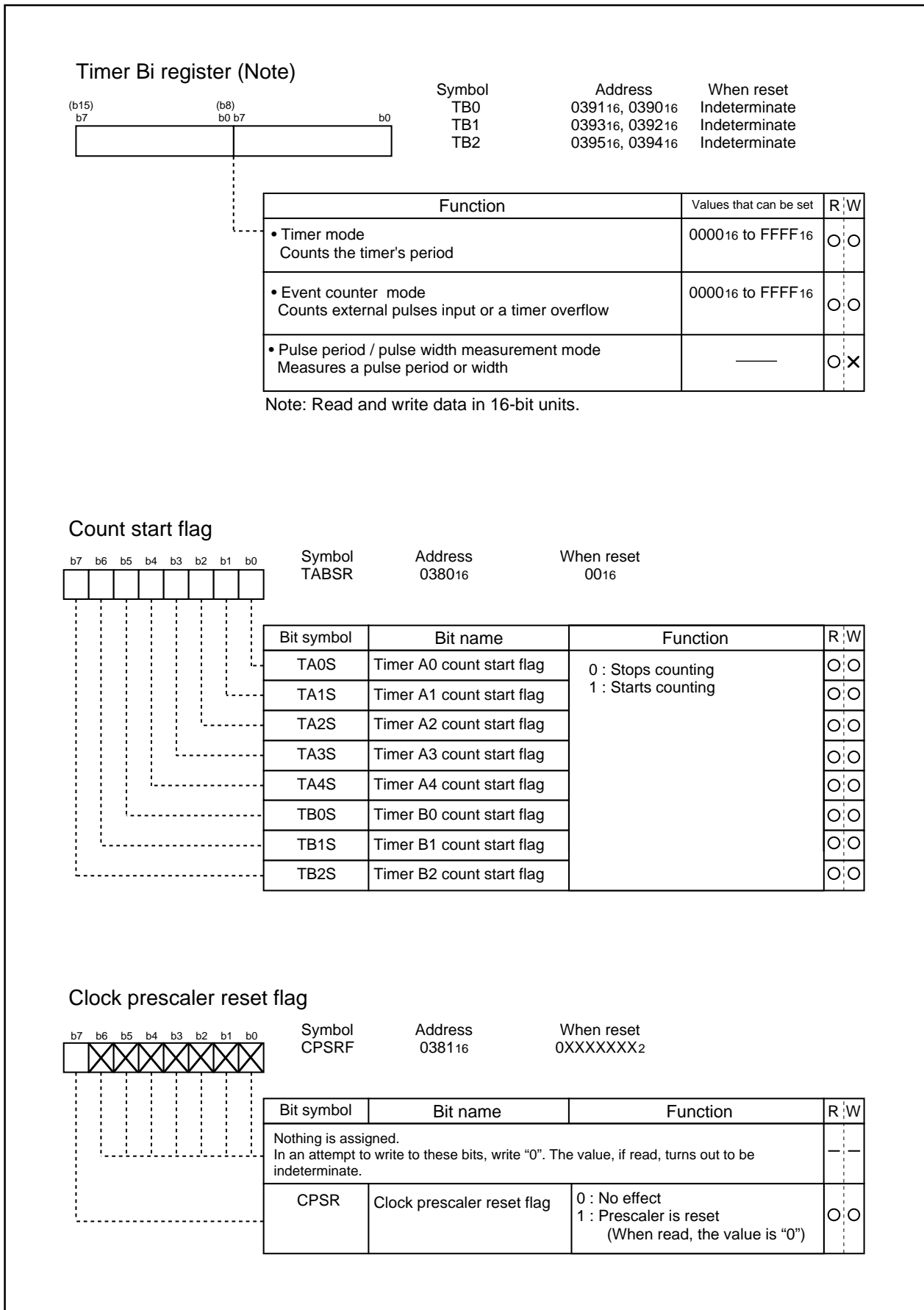


Figure 2.3.3. Timer B-related registers (2)

## Timer B

## 2.3.2 Operation of Timer B (timer mode)

In timer mode, choose functions from those listed in Table 2.3.1. Operations of the circled items are described below. Figure 2.3.4 shows the operation timing, and Figure 2.3.5 shows the set-up procedure.

Table 2.3.1. Chosed functions

Item	Set-up	
Count source	○	Internal count source (f1 / f8 / f32 / fc32)

- Operation
- (1) Setting the count start flag to "1" causes the counter to perform a down count on the count source.
  - (2) If an underflow occurs, the content of the reload register is reloaded, and the counter continues counting. At this time, the timer Bi interrupt request bit goes to "1".
  - (3) Setting the count start flag to "0" causes the counter to hold its value and to stop.

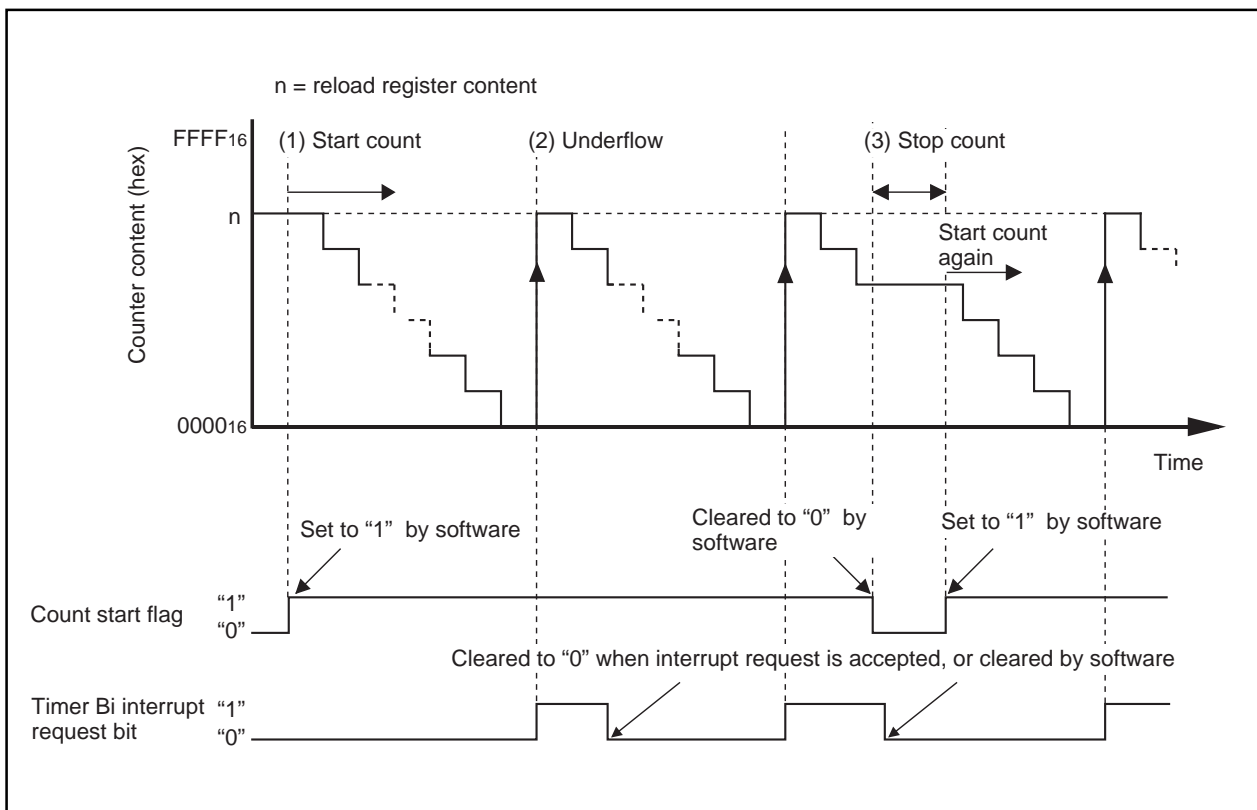


Figure 2.3.4. Operation timing of timer mode

Timer B

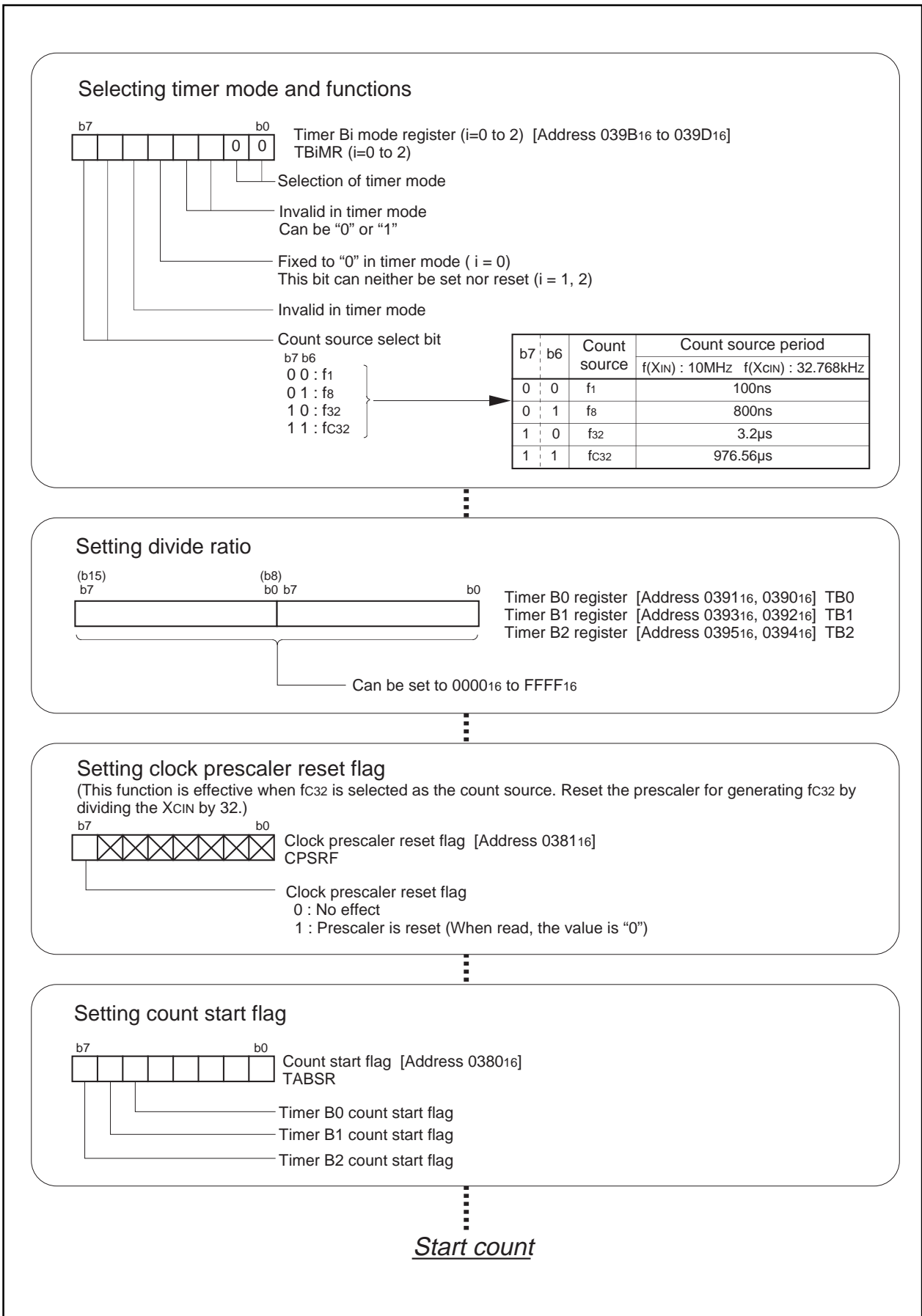


Figure 2.3.5. Set-up procedure of timer mode



## Timer B

## 2.3.3 Operation of Timer B (event counter mode)

In event counter mode, choose functions from those listed in Table 2.3.2. Operations of the circled items are described below. Figure 2.3.6 shows the operation timing, and Figure 2.3.7 shows the set-up procedure.

Table 2.3.2. Chosed functions

Item	Set-up
Count source	○ Input signal to the TB <sub>i</sub> IN pin (counting falling edges)
	○ Input signal to the TB <sub>i</sub> IN pin (counting rising edges)
	○ Input signal to the TB <sub>i</sub> IN pin (counting rising edges and falling edges)
	○ Timer overflow(TB <sub>j</sub> overflow)

Note:  $j = i - 1$ , but  $j = 2$  when  $i = 0$

- Operation
- (1) Setting the count start flag to "1" causes the counter to count the falling edges of the count source.
  - (2) If an underflow occurs, the content of the reload register is reloaded, and the count continues. At this time, the timer Bi interrupt request bit goes to "1".
  - (3) Setting the count start flag to "0" causes the counter to hold its value and to stop.

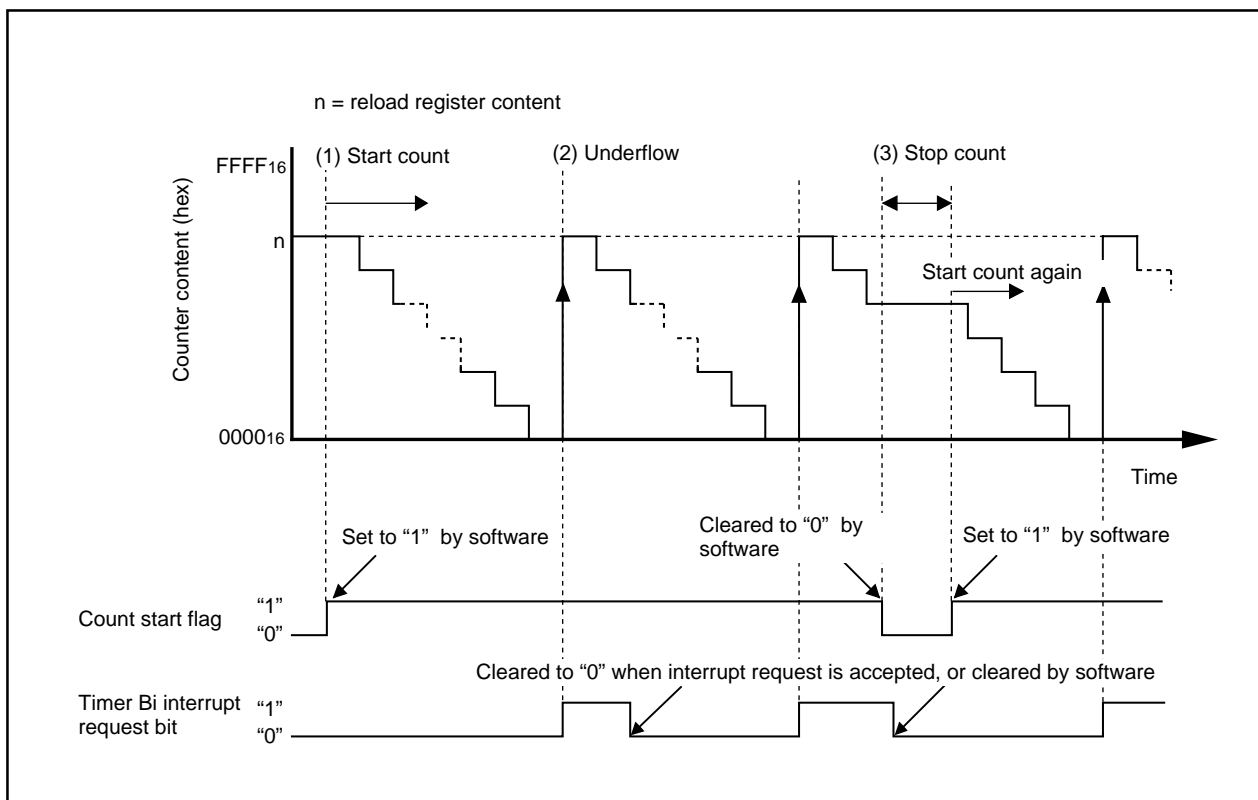


Figure 2.3.6. Operation timing of event counter mode

## Timer B

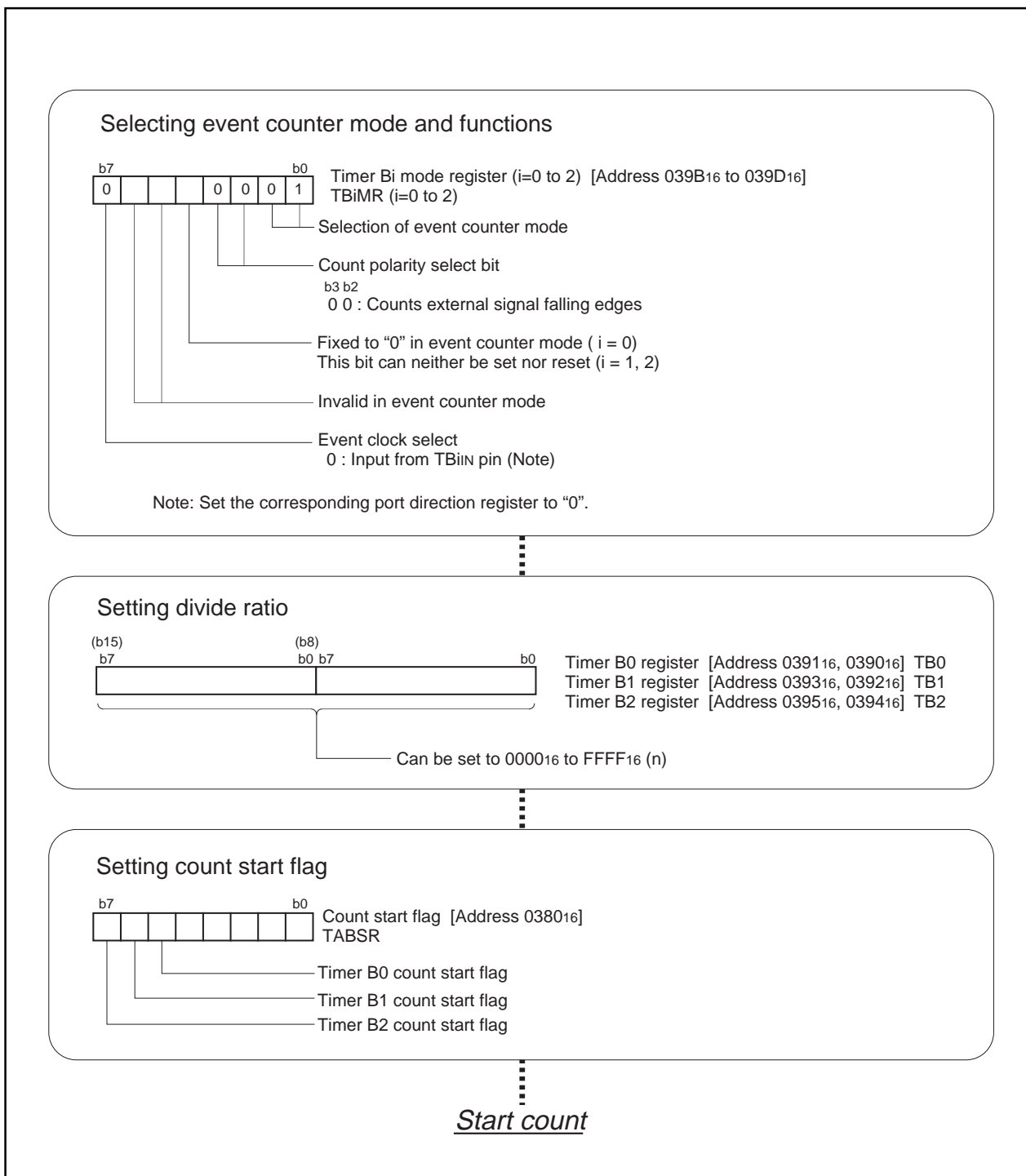


Figure 2.3.7. Set-up procedure of event counter mode

## Timer B

### 2.3.4 Operation of Timer B (pulse period measurement mode)

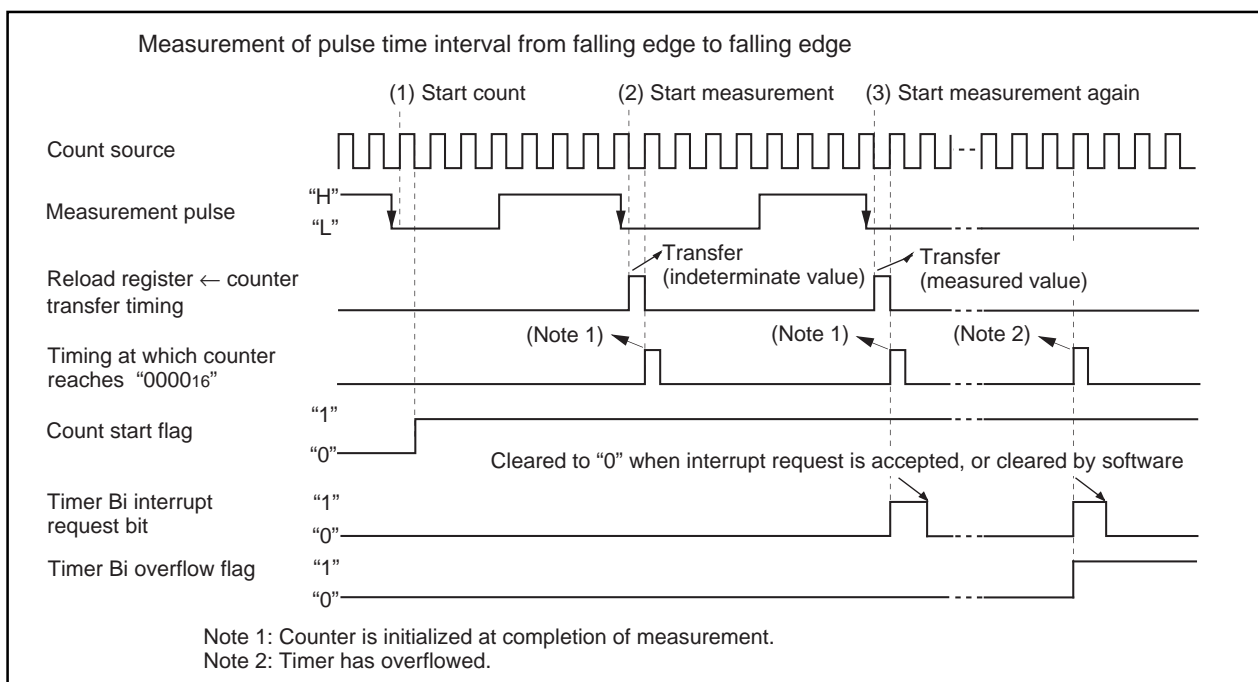
In pulse period/pulse width measurement mode, choose functions from those listed in Table 2.3.3. Operations of the circled items are described below. Figure 2.3.8 shows the operation timing, and Figure 2.3.9 shows the set-up procedure.

**Table 2.3.3. Chosed functions**

Item	Set-up
Count source	○ Internal count source ( $f_1 / f_8 / f_{32} / f_{c32}$ )
Measurement mode	○ Pulse period measurement (interval between measurement pulse falling edge to falling edge)
	○ Pulse period measurement (interval between measurement pulse rising edge to rising edge)
	○ Pulse width measurement (interval between measurement pulse falling edge to rising edge, and between rising edge to falling edge)

- Operation
- (1) Setting the count start flag to "1" causes the counter to start counting the count source.
  - (2) If a measurement pulse changes from "H" to "L", the value of the counter goes to "0000<sub>16</sub>", and measurement is started. In this instance, an indeterminate value is transferred to the reload register. The timer Bi interrupt request is not generated.
  - (3) If a measurement pulse changes from "H" to "L" again, the value of the counter is transferred to the reload register, and the timer Bi interrupt request bit goes to "1". Then the value of the counter becomes "0000<sub>16</sub>", and the measurement is started again.

- Note
- The timer Bi interrupt request bit goes to "1" when an effective edge of a measurement pulse is input or timer Bi is overflowed. The factor of interrupt request can be determined by use of the timer Bi overflow flag within the interrupt routine.
  - The value of the counter at the beginning of a count is indeterminate. Thus there can be instances in which the timer Bi overflow flag goes to "1" immediately after a count is performed.
  - The timer Bi overflow flag goes to "0" if timer Bi mode register is written to when the count start flag is "1". This flag cannot be set to "1" by software.



**Figure 2.3.8. Operation timing of pulse period measurement mode**

## Timer B

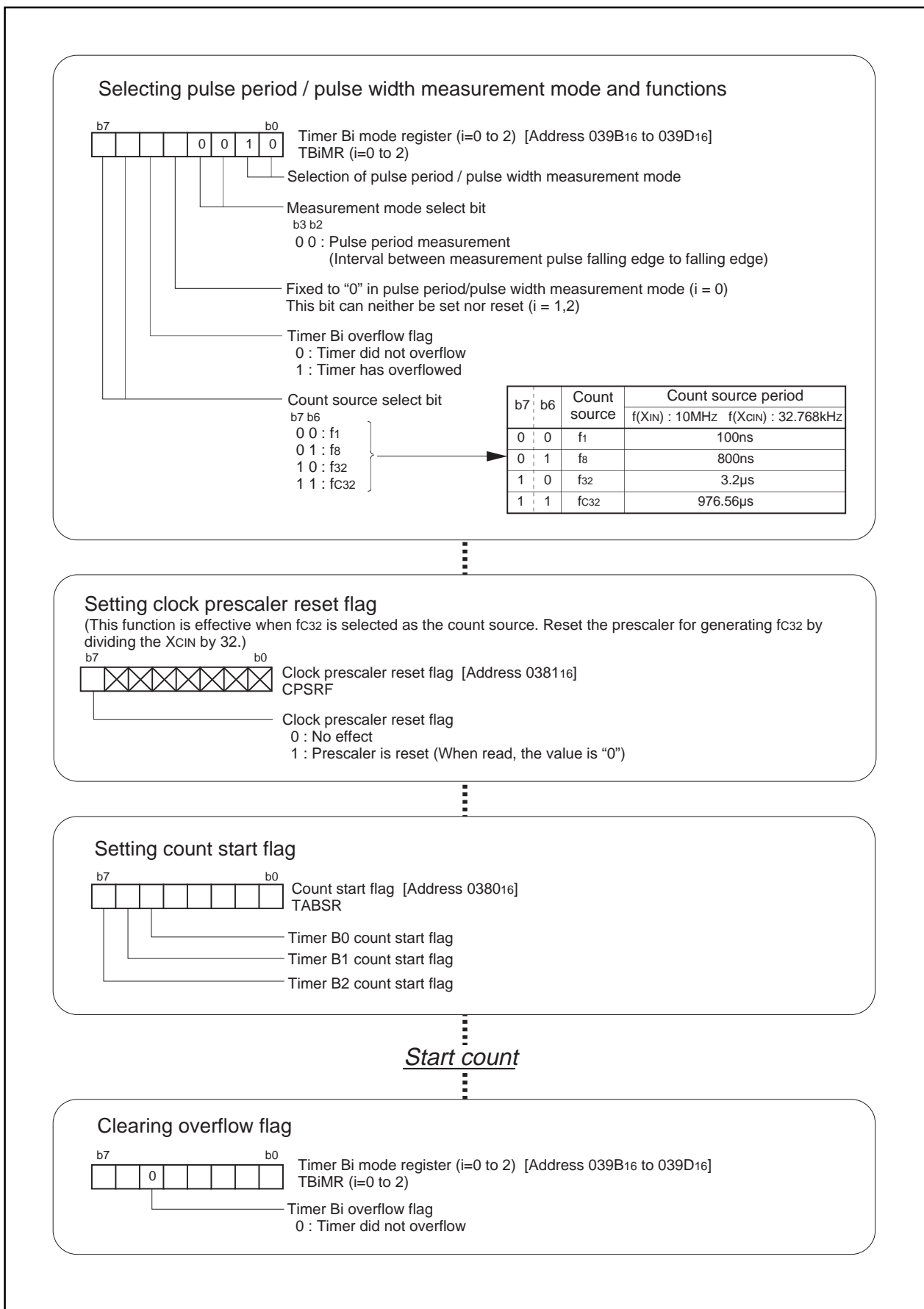


Figure 2.3.9. Set-up procedure of pulse period measurement mode

### 2.3.5 Operation of Timer B (pulse width measurement mode)

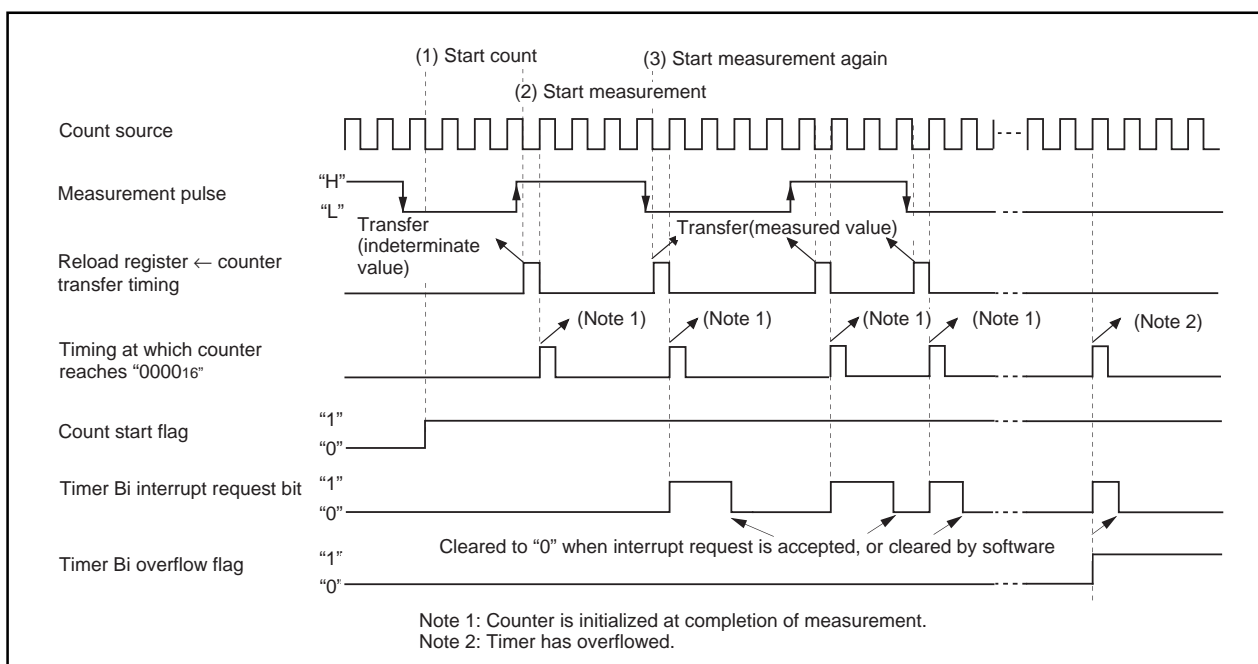
In pulse period/pulse width measurement mode, choose functions from those listed in Table 2.3.4. Operations of the circled items are described below. Figure 2.3.10 shows the operation timing, and Figure 2.3.11 shows the set-up procedure.

**Table 2.3.4. Chosed functions**

Item	Set-up
Count source	○ Internal count source ( $f_1 / f_8 / f_{32} / f_{c32}$ )
Measurement mode	Pulse period measurement (interval between measurement pulse falling edge to falling edge)
	Pulse period measurement (interval between measurement pulse rising edge to rising edge)
	○ Pulse width measurement (interval between measurement pulse falling edge to rising edge, and between rising edge to falling edge)

- Operation
- (1) Setting the count start flag to "1" causes the counter to start counting the count source.
  - (2) If an effective edge of a pulse to be measured is input, the value of the counter goes to "0000<sub>16</sub>", and measurement is started. In this instance, an indeterminate value is transferred to the reload register. The timer Bi interrupt request is not generated.
  - (3) If an effective edge of a pulse to be measured is input again, the value of the counter is transferred to the reload register, and the timer Bi interrupt request bit goes to "1". Then the value of the counter becomes "0000<sub>16</sub>", and measurement is started again.

- Note
- The timer Bi interrupt request bit goes to "1" when an effective edge of a pulse to be measured is input or timer Bi overflows. The factor of interrupt request can be determined by use of the timer Bi overflow flag within the interrupt routine.
  - The value of the counter at the beginning of a count is indeterminate. Thus there can be instances in which the timer Bi overflow flag goes to "1" immediately after a count is performed.
  - The timer Bi overflow flag goes to "0" if timer Bi mode register is written to when the count start flag is "1". This flag cannot be set to "1" by software.



**Figure 2.3.10. Operation timing of pulse width measurement mode**

## Timer B

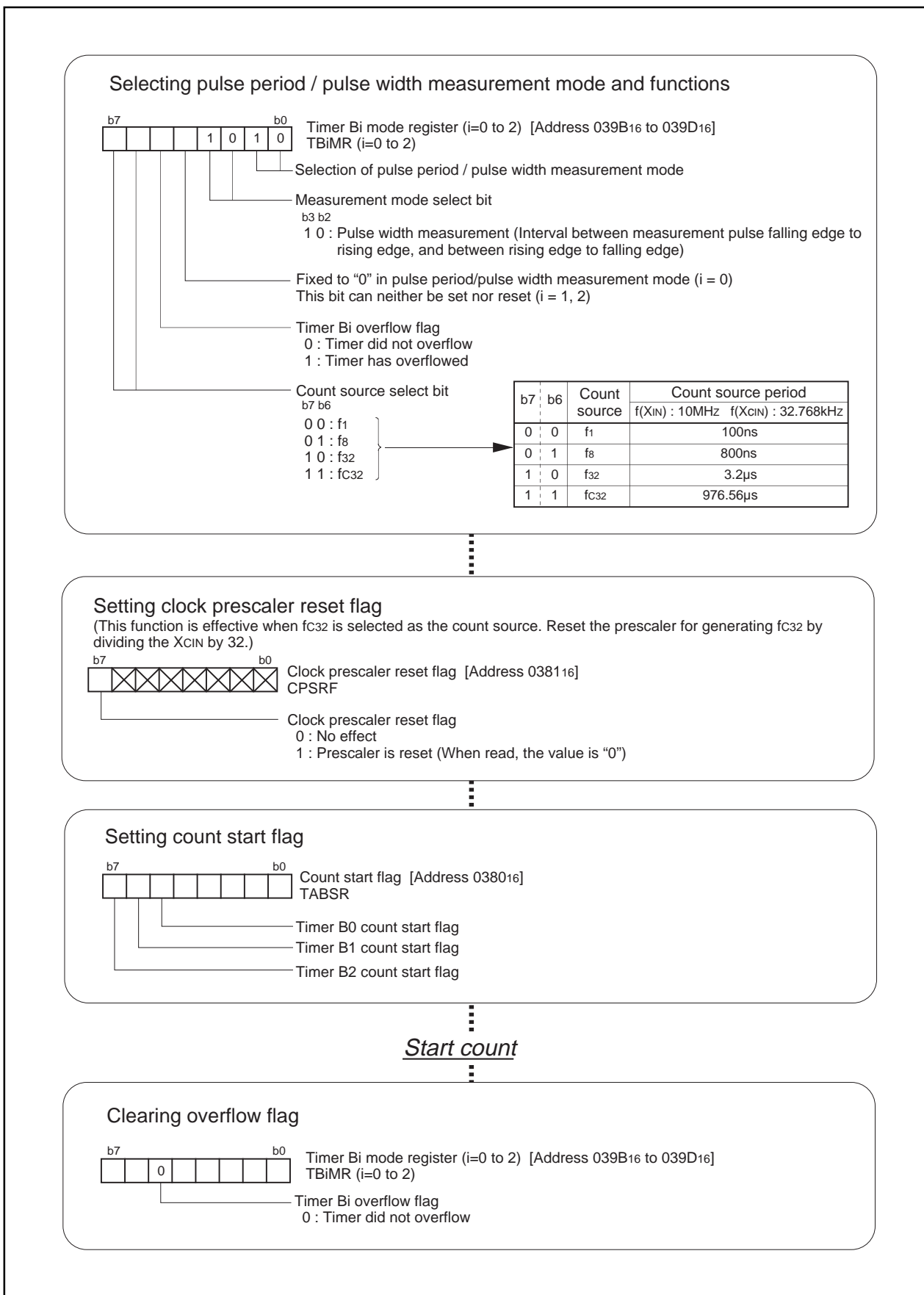
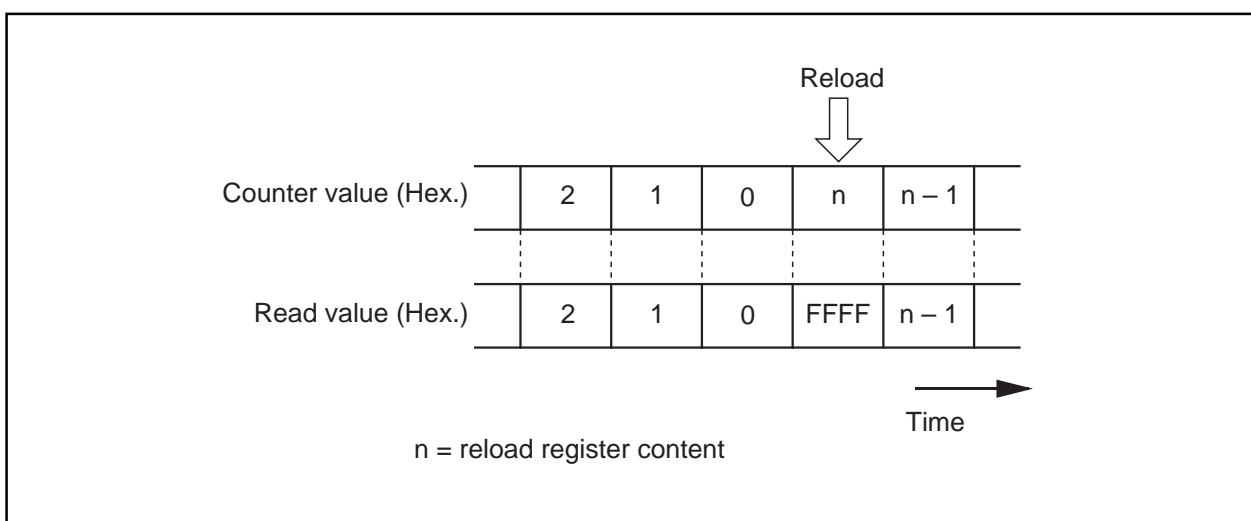


Figure 2.3.11. Set-up procedure of pulse width measurement mode

**2.3.6 Precautions for Timer B (timer mode, event counter mode)**

- (1) To clear reset, the count start flag is set to "0". Set a value in the timer Bi register, then set the flag to "1".
- (2) 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 shown in Figure 2.3.12 gets "FFFF<sub>16</sub>". 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.



**Figure 2.3.12. Reading timer Bi register**

### 2.3.7 Precautions for Timer B (pulse period/pulse width measurement mode)

- (1) The timer Bi interrupt request bit goes to "1" when an effective edge of a measurement pulse is input or timer Bi is overflowed. The factor of interrupt request can be determined by use of the timer Bi overflow flag within the interrupt routine.
- (2) If the timer overflow occurs simultaneously with the input of a measurement pulse, and if the interrupt factor cannot be determined from the timer Bi overflow flag, connect the timers and count the number of overflows.
- (3) When reset, the timer Bi overflow flag goes to "1". This flag can be set to "0" by writing to the timer Bi mode register when the count start flag is "1".
- (4) Use the timer Bi interrupt request bit to detect only overflows. Use the timer Bi overflow flag only to determine the interrupt factor within the interrupt routine.
- (5) 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.
- (6) The value of the counter is indeterminate at the beginning of a count. Therefore the timer Bi overflow flag may go to "1" immediately after a count is started.
- (7) If changing the measurement mode select bit is set after a count is started, the timer Bi interrupt request bit goes to "1".
- (8) If the input signal to the TBiIN pin is affected by noise, precise measurement may not be performed in some cases. It is recommended to see that measurements fall within a specific range by use of software.
- (9) For pulse width measurement, pulse widths are successively measured. Use software to check whether the measurement result is an "H" level width or an "L" level width.



## 2.4 Clock-Synchronous Serial I/O

### 2.4.1 Overview

Clock-synchronous serial I/O carries out 8-bit data communications in synchronization with the clock. The following is an overview of the clock-synchronous serial I/O.

#### (1) Transmission/reception format

8-bit data

#### (2) Transfer rate

If the internal clock is selected as the transfer clock, the divide-by-2 frequency, resulting from the bit rate generator division, becomes the transfer rate. The bit rate generator count source can be selected from the following: f<sub>1</sub>, f<sub>8</sub>, and f<sub>32</sub>. Clocks f<sub>1</sub>, f<sub>8</sub>, and f<sub>32</sub> are derived by dividing the CPU's main clock by 1, 8, and 32 respectively.

Furthermore, if an external clock is selected as the transfer clock, the clock frequency input to the CLK pin becomes the transfer rate.

#### (3) Error detection

Only overrun error can be detected. Overrun error is an error that occurs when the next data is made ready before the reception buffer register is read.

#### (4) How to deal with an error

When receiving data, read an error flag and reception data simultaneously to determine which error has occurred. If the data read is erroneous, initialize the error flag and the UART<sub>i</sub> receive buffer register, then receive the data again.

##### To initialize the UART<sub>i</sub> receive buffer register

1. Set the receive enable bit to "0" (disable reception).
2. Set the serial I/O mode select bit to "0002" (invalid serial I/O).
3. Set the serial I/O mode select bit.
4. Set the receive enable bit to "1" again (enable reception).

To transmit data again due to an error on the reception side, set the UART<sub>i</sub> transmit buffer register again, then transmit the data again.

##### To set the UART<sub>i</sub> transmit buffer register again

1. Set the serial I/O mode select bits to "0002" (invalidate serial I/O).
2. Set the serial I/O mode select bits again.
3. Set the transmit enable bit to "1" (enable transmission), then set transmission data in the UART<sub>i</sub> transmit buffer register.

#### (5) Function selection

For clock-synchronous serial I/O, the following functions can be selected:

##### (a) CTS/RTS function

In the CTS function, an external IC can start transmission/reception by inputting an "L" level to the CTS pin. The CTS pin input level is detected when transmission/reception starts. Therefore, if the level is set to "H" during transmission/reception, it will stop from the next data.

The RTS function informs an external IC that RTS is reception-ready and has changed to "L". RTS goes to "H" at the falling edge of the transfer clock.

The clock-synchronous serial I/O has three types of CTS/RTS functions to choose from:

- CTS/RTS functions disabled     CTS/RTS pin is a programmable I/O port.
- CTS function only enabled     CTS/RTS pin performs the CTS function.
- RTS function only enabled     CTS/RTS pin performs the RTS function.

**(b) Function for choosing polarity**

This function switches the polarity of the transfer clock. The following operations are available:

- Data is input at the falling edge of the transfer clock, and is output at the rising edge.
- Data is input at the rising edge of the transfer clock, and is output at the falling edge.

**(c) Function for choosing which bit to transmit first**

This function is to choose whether to transmit data from bit 0 or from bit 7. Choose either of the following:

- LSB first      Data is transmitted from bit 0.
- MSB first     Data is transmitted from bit 7.

**(d) Function for choosing successive reception mode**

Successive reception mode is a mode in which reading the receive buffer register makes the reception-enabled status ready. In this mode, there is no need to write dummy data to the transmit buffer register so as to make the reception-enabled status ready. But at the time of starting reception, read the receive buffer register into a dummy manner.

- Normal mode                      Writing dummy data to the transmit buffer register makes the reception enabled status ready.
- Successive reception mode      Reading the reception buffer register makes the reception-enabled status ready.

**(e) Function for outputting transfer clock to multiple pins**

This function is to switch among pins to output the transfer clock. This function is effective only when selecting the internal clock. Switching among pins for outputting the transfer clock allows data transmission to two external ICs in a time-sharing manner.

**(f) Function for choosing a transmission interrupt factor**

The timing to generate a transmission interrupt can be selected from the following: the instant the transmission buffer is emptied or the instant the transmission register is emptied. When transmission buffer empty timing is selected, an interrupt occurs when transmitted data is moved from the transmission buffer to the transmission register. Therefore, data can be transmitted in succession. When transmission register empty timing is selected, an interrupt occurs when data transmission is complete.

Following are some examples in which various functions (a) through (f) are selected:

- Transmission Operation WITH:  $\overline{\text{CTS}}$  function, transmission at falling edge of transfer clock, LSB First, interrupt at instant transmission buffer is emptied; WITHOUT transfer clock output to multiple pins function ..... P232
- Transmission Operation WITH:  $\overline{\text{CTS/RTS}}$  function disabled, transmission at falling edge of transfer clock, LSB First, interrupt at instant transmission is completed; WITH transfer clock output to multiple pins function (UART0 selection available) ..... P236
- Reception WITH:  $\overline{\text{RTS}}$  function, reception at falling edge of transfer clock, LSB First, successive reception mode disabled; WITHOUT transfer clock output to multiple pins function ..... P240

**(6) Input to the serial I/O and the direction register**

To input an external signal to the serial I/O, set the direction register of the relevant port to input.

**(7) Pins related to the serial I/O**

- $\overline{\text{CTS}}_0$ ,  $\text{CTS}_1$  pins     Input pins for the  $\overline{\text{CTS}}$  function
- $\overline{\text{RTS}}_0$ ,  $\text{RTS}_1$  pins     Output pins for the  $\overline{\text{RTS}}$  function
- $\text{CLK}_0$ ,  $\text{CLK}_1$  pins     Input/output pins for the transfer clock
- $\text{RxD}_0$ ,  $\text{RxD}_1$  pins     Input pins for data
- $\text{TxD}_0$ ,  $\text{TxD}_1$  pins     Output pins for data.
- $\text{CLKS}_1$  pin                     Output pin for transfer clock. Can be used as transfer clock output pin in the transfer clock output to multiple pins function.

**(8) Registers related to the serial I/O**

Figure 2.4.1 shows the memory map of serial I/O-related registers, and Figures 2.4.2 to 2.4.4 show serial I/O-related registers.

0051 <sub>16</sub>	UART0 transmit interrupt control register (S0TIC)
0052 <sub>16</sub>	UART0 receive interrupt control register (S0RIC)
0053 <sub>16</sub>	UART1 transmit interrupt control register (S1TIC)
0054 <sub>16</sub>	UART1 receive interrupt control register (S1RIC)
≈	≈
03A0 <sub>16</sub>	UART0 transmit/receive mode register (U0MR)
03A1 <sub>16</sub>	UART0 bit rate generator (U0BRG)
03A2 <sub>16</sub>	UART0 transmit buffer register (U0TB)
03A3 <sub>16</sub>	
03A4 <sub>16</sub>	UART0 transmit/receive control register 0 (U0C0)
03A5 <sub>16</sub>	UART0 transmit/receive control register 1 (U0C1)
03A6 <sub>16</sub>	UART0 receive buffer register (U0RB)
03A7 <sub>16</sub>	
03A8 <sub>16</sub>	UART1 transmit/receive mode register (U1MR)
03A9 <sub>16</sub>	UART1 bit rate generator (U1BRG)
03AA <sub>16</sub>	UART1 transmit buffer register (U1TB)
03AB <sub>16</sub>	
03AC <sub>16</sub>	UART1 transmit/receive control register 0 (U1C0)
03AD <sub>16</sub>	UART1 transmit/receive control register 1 (U1C1)
03AE <sub>16</sub>	UART1 receive buffer register (U1RB)
03AF <sub>16</sub>	
03B0 <sub>16</sub>	UART transmit/receive control register 2 (UCON)
03B1 <sub>16</sub>	

**Figure 2.4.1. Memory map of serial I/O-related registers**

Clock-Synchronous Serial I/O

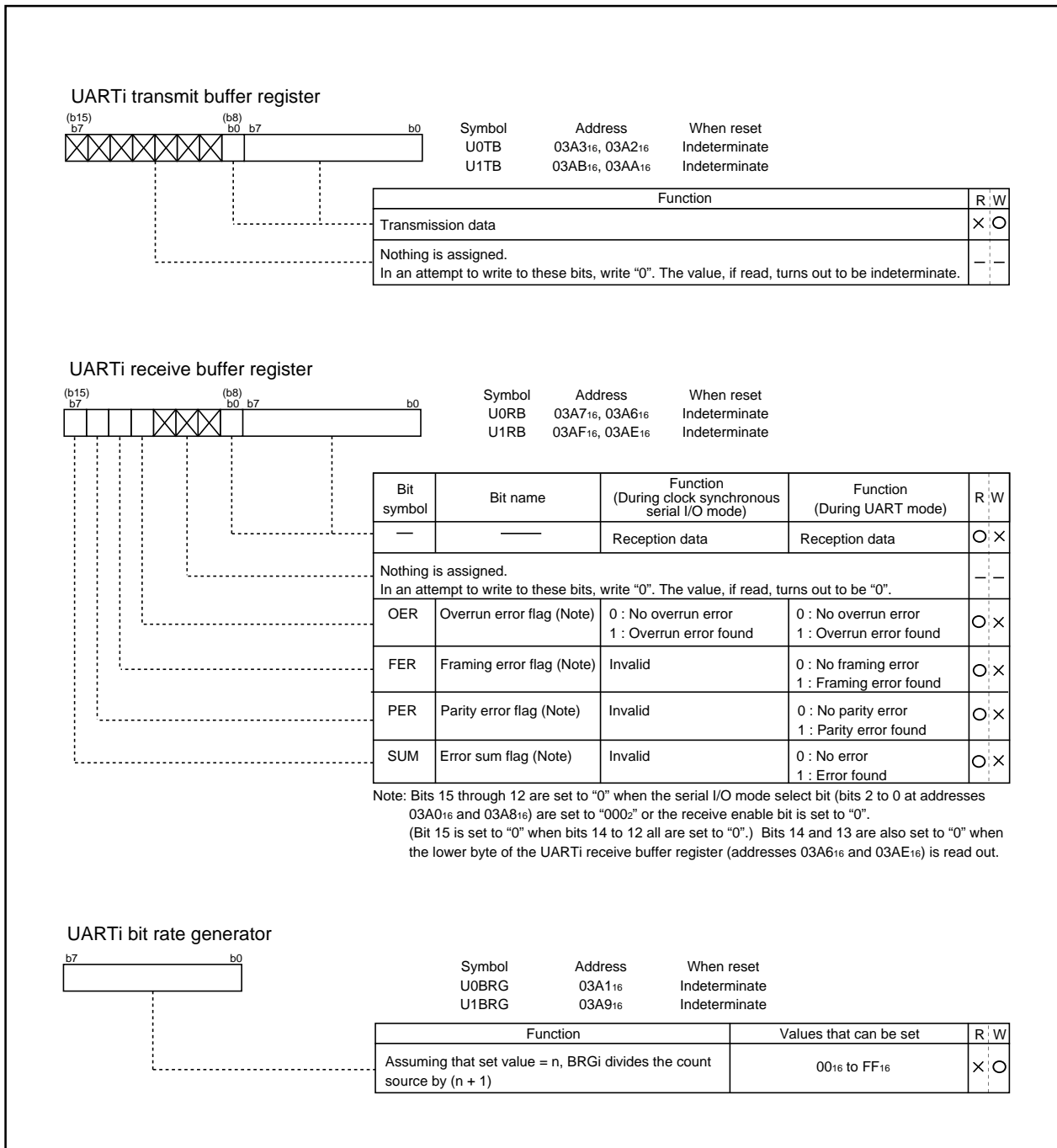


Figure 2.4.2. Serial I/O-related registers (1)

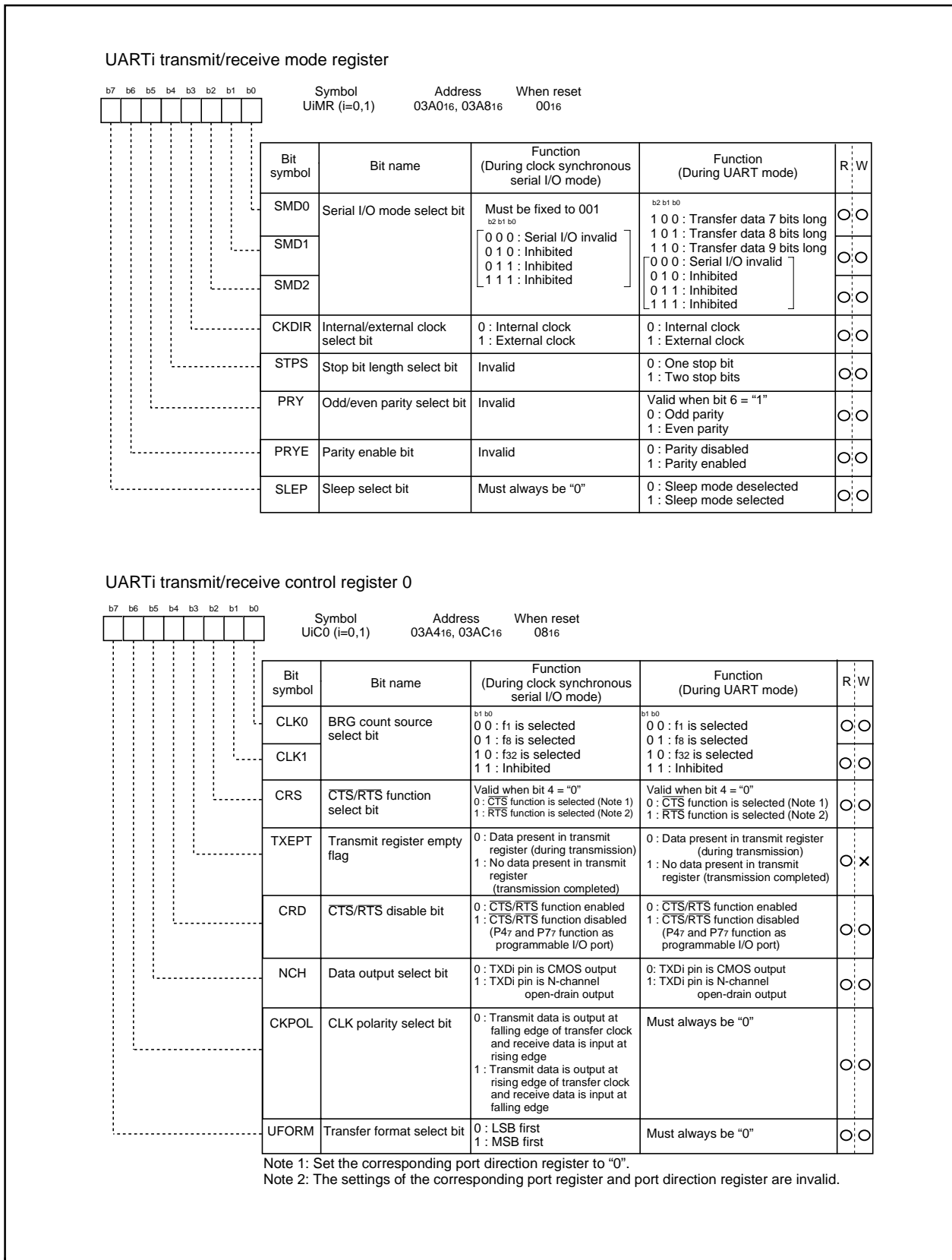


Figure 2.4.3. Serial I/O-related registers (2)

## Clock-Synchronous Serial I/O

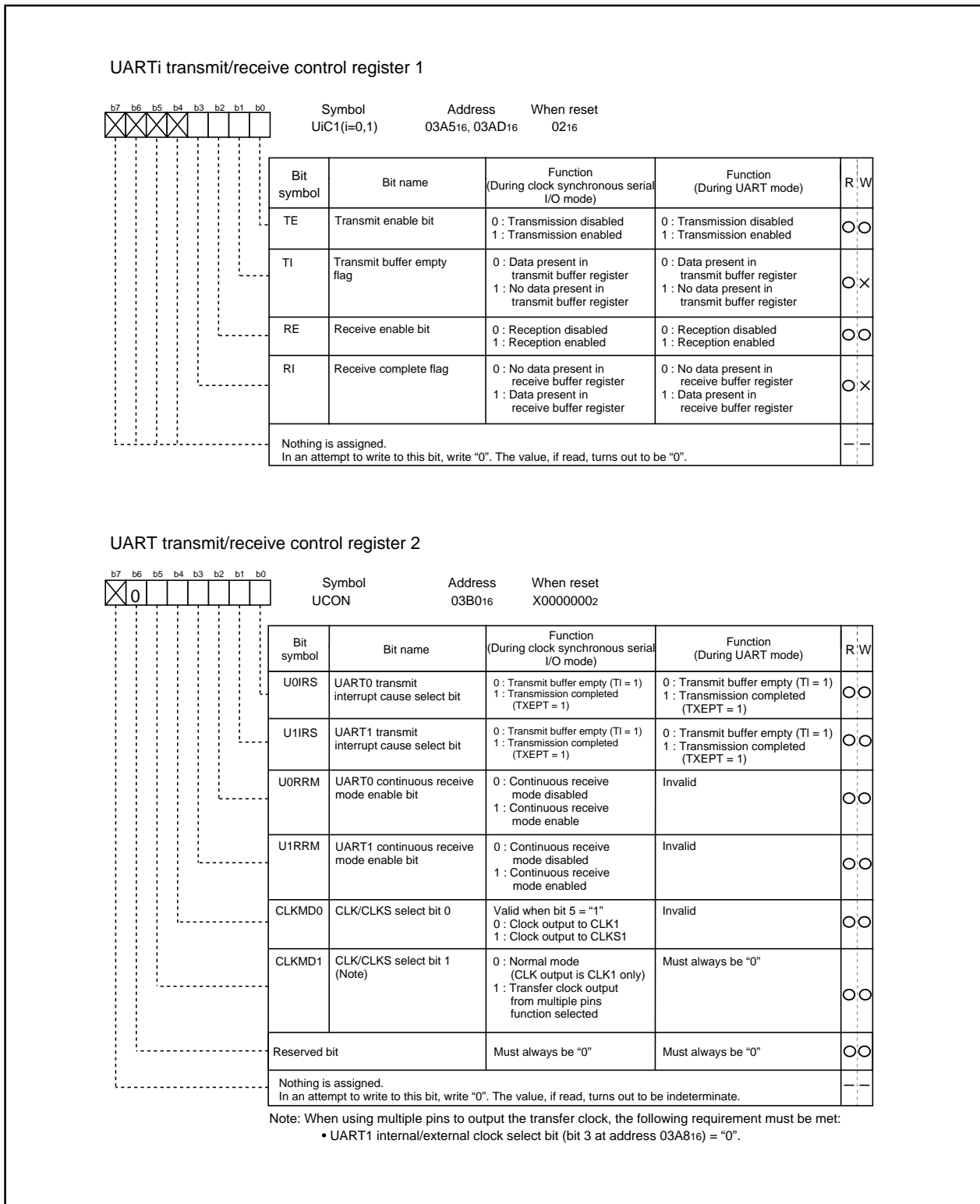


Figure 2.4.4. Serial I/O-related registers (3)

## 2.4.2 Operation of Serial I/O (transmission in clock-synchronous serial I/O mode)

In transmitting data in clock-synchronous serial I/O mode, choose functions from those listed in Table 2.4.1. Operations of the circled items are described below. Figure 2.4.5 shows the operation timing, and Figures 2.4.6 and 2.4.7 show the set-up procedures.

**Table 2.4.1. Chosed functions**

Item	Set-up		Item	Set-up	
Transfer clock source	<input type="radio"/>	Internal clock ( $f_1 / f_8 / f_{32}$ )	Transmission interrupt factor	<input type="radio"/>	Transmission buffer empty
		External clock (CLKi pin)			Transmission complete
$\overline{\text{CTS}}$ function	<input type="radio"/>	$\overline{\text{CTS}}$ function enabled	Output transfer clock to multiple pins (Note)	<input type="radio"/>	Not selected
		$\overline{\text{CTS}}$ function disabled			Selected
CLK polarity	<input type="radio"/>	Output transmission data at the falling edge of the transfer clock			
		Output transmission data at the rising edge of the transfer clock			
Transfer clock	<input type="radio"/>	LSB first			
		MSB first			

Note: This can be selected only when UART1 is used in combination with the internal clock. When this function is selected, UART1  $\overline{\text{CTS}}$ /RTS function cannot be utilized. Set the UART1  $\overline{\text{CTS}}$ /RTS disable bit to "1".

- Operation
- (1) Setting the transmit enable bit to "1" and writing transmission data to the UARTi transmit buffer register makes data transmissible status ready.
  - (2) When input to the  $\overline{\text{CTS}}_i$  pin goes to "L" level, transmission starts (the  $\overline{\text{CTS}}_i$  pin must be controlled on the reception side).
  - (3) In synchronization with the first falling edge of the transfer clock, transmission data held in the UARTi transmit buffer register is transmitted to the UARTi transmit register. At this time, the UARTi transmit interrupt request bit goes to "1". Also, the first bit of the transmission data is transmitted from the TxDi pin. Then the data is transmitted bit by bit from the lower order in synchronization with the falling edges.
  - (4) When transmission of 1-byte data is completed, the transmit register empty flag goes to "1", which indicates that transmission is completed. The transfer clock stops at "H" level.
  - (5) If the next transmission data is set in the UARTi transmit buffer register while transmission is in progress (before the eighth bit has been transmitted), the data is transmitted in succession.

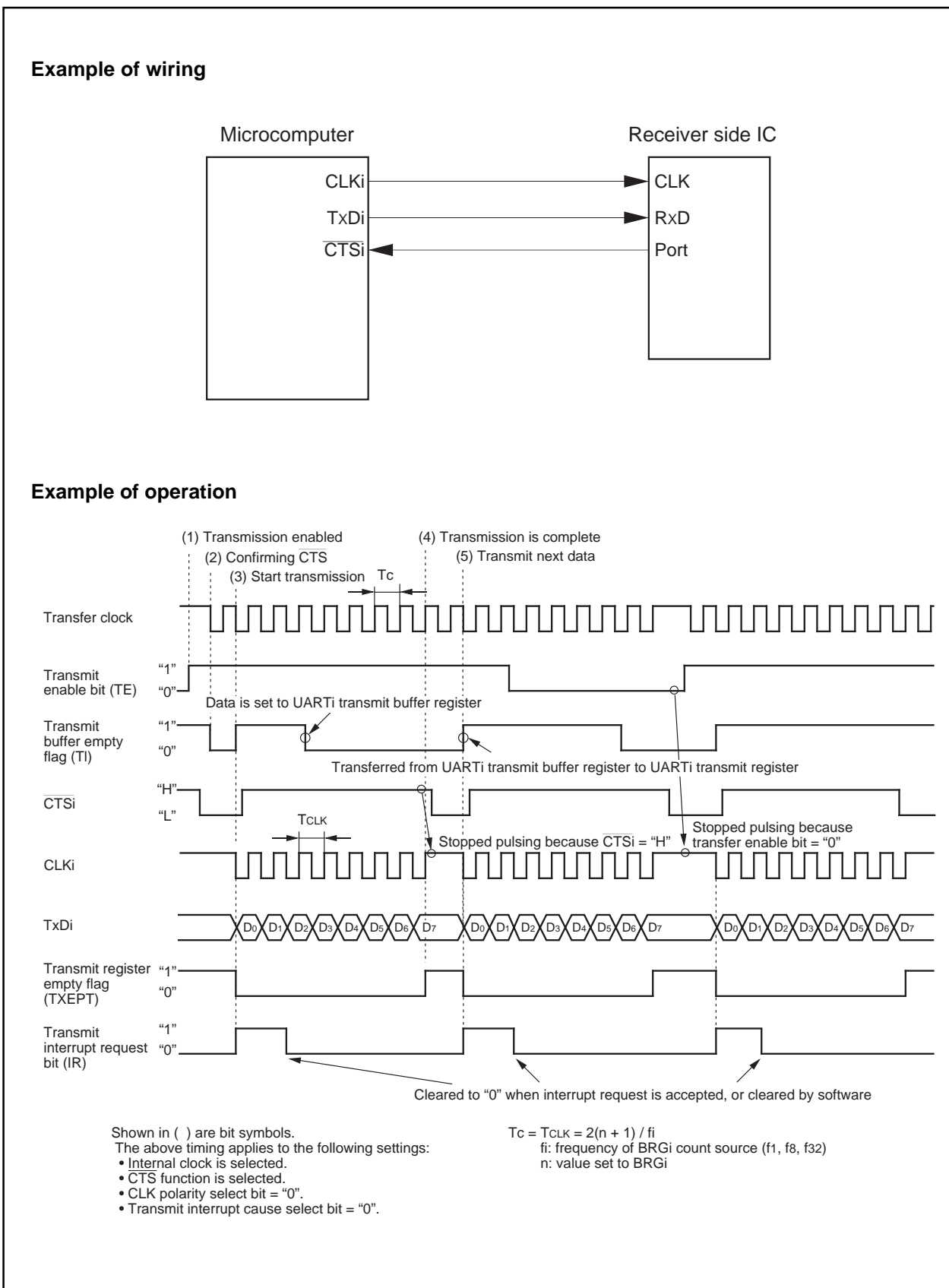


Figure 2.4.5. Operation timing of transmission in clock-synchronous serial I/O mode



Clock-Synchronous Serial I/O

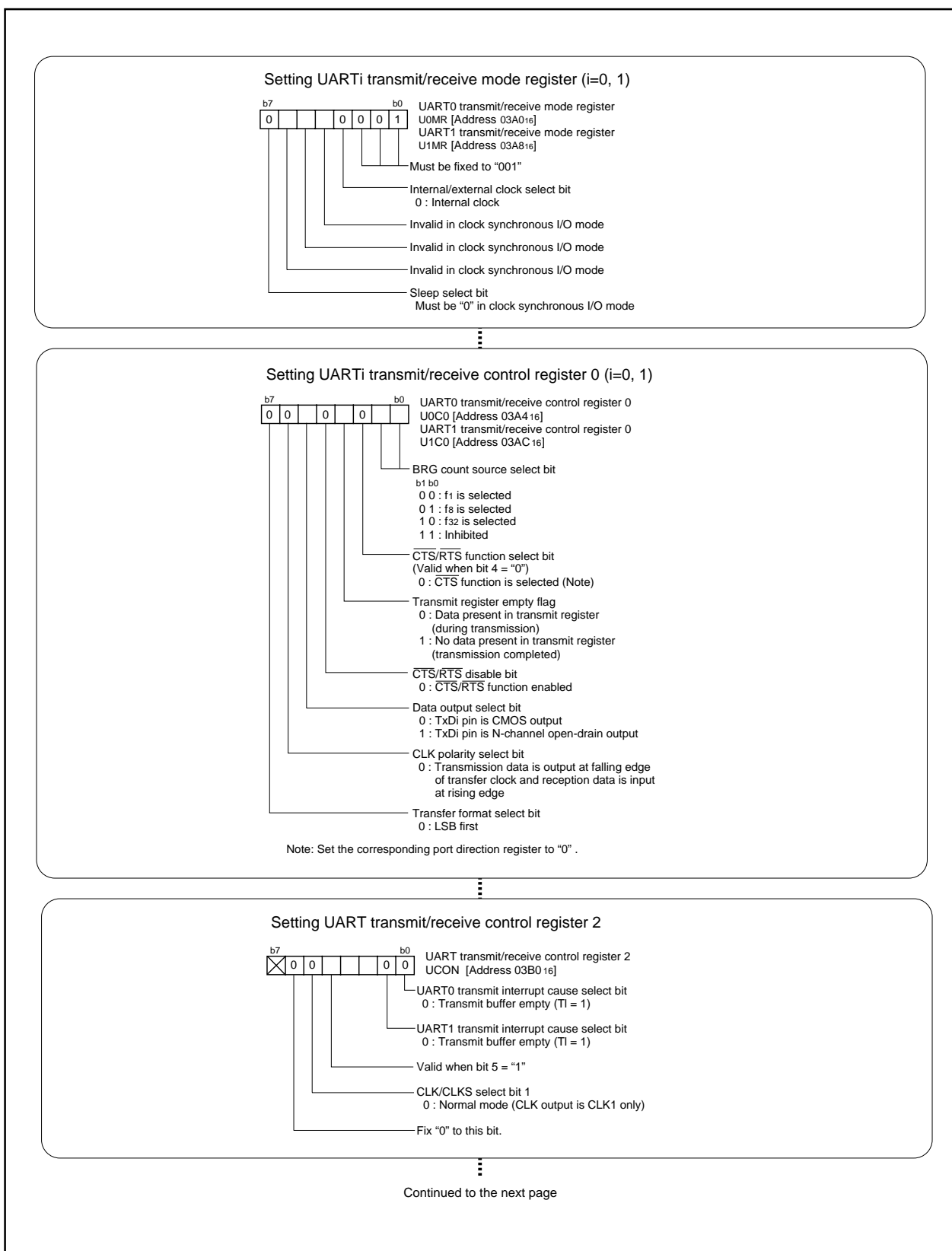


Figure 2.4.6. Set-up procedure of transmission in clock-synchronous serial I/O mode (1)

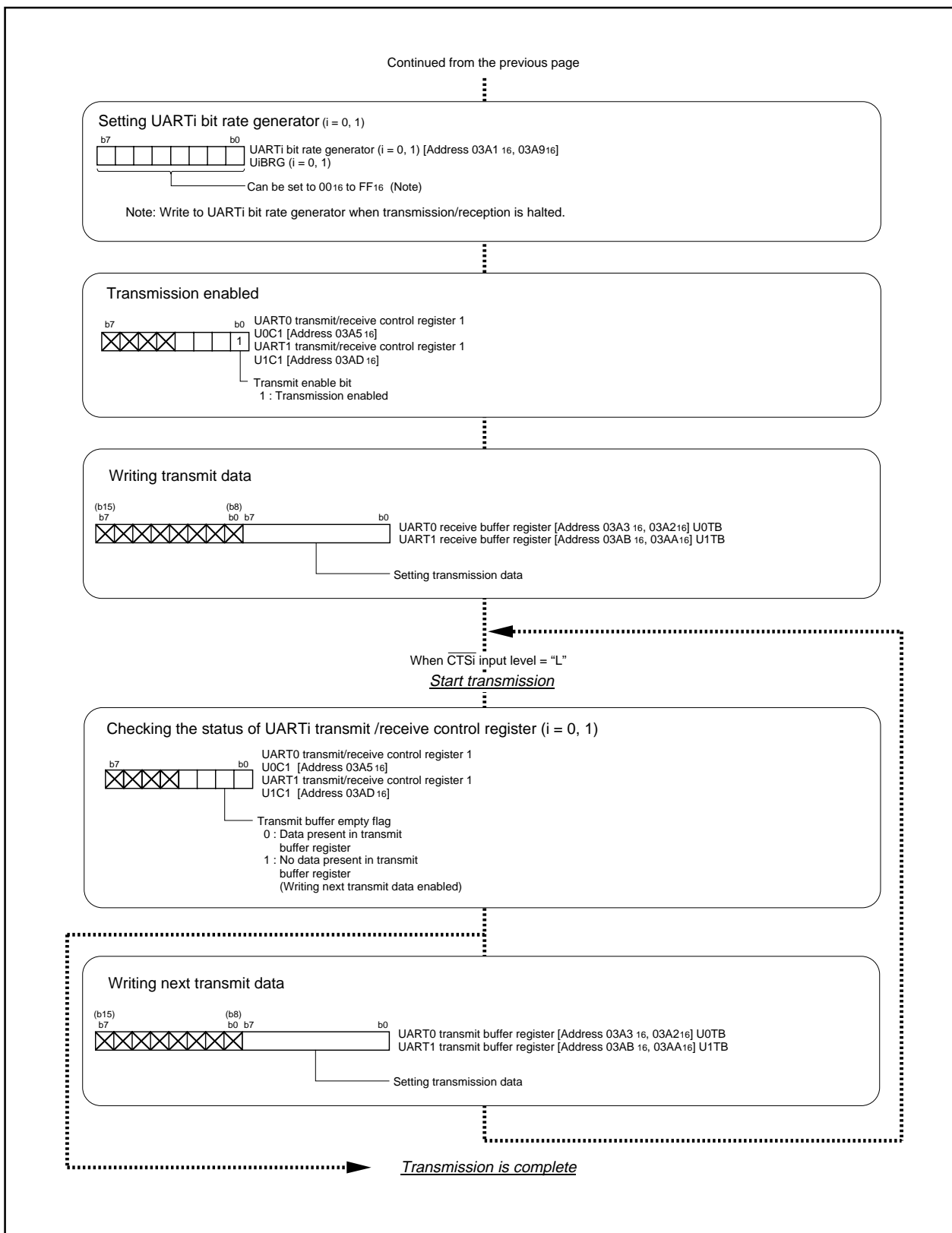


Figure 2.4.7. Set-up procedure of transmission in clock-synchronous serial I/O mode (2)

### 2.4.3 Operation of the Serial I/O (transmission in clock-synchronous serial I/O mode, transfer clock output from multiple pins function selected)

In transmitting data in clock-synchronous serial I/O mode, choose functions from those listed in Table 2.4.2. Operations of the circled items are described below. Figure 2.4.8 shows the operation timing, and Figures 2.4.9 and 2.4.10 show the set-up procedures.

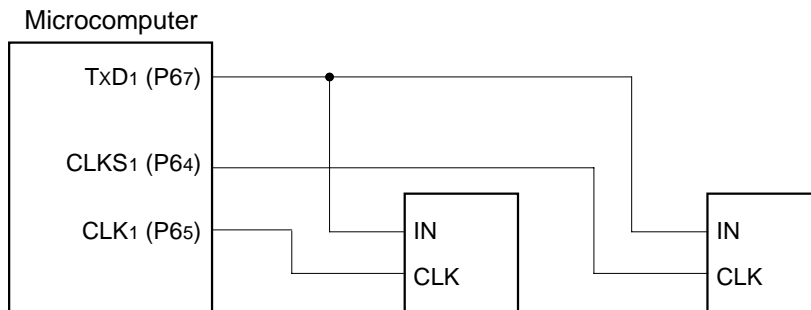
**Table 2.4.2. Chosed functions**

Item	Set-up		Item	Set-up	
Transfer clock source	<input type="radio"/>	Internal clock ( $f_1 / f_8 / f_{32}$ )	Transmission interrupt factor		Transmission buffer empty
	<input type="radio"/>	External clock (CLKi pin)		<input type="radio"/>	Transmission complete
CTS function		CTS function enabled	Output transfer clock to multiple pins (Note 1)		Not selected
	<input type="radio"/>	CTS function disabled		<input type="radio"/>	Selected
CLK polarity	<input type="radio"/>	Output transmission data at the falling edge of the transfer clock			
		Output transmission data at the rising edge of the transfer clock			
Transfer clock	<input type="radio"/>	LSB first			
		MSB first			

Note 1: This can be selected only when UART1 is used in combination with the internal clock. When this function is selected, UART1 CTS/RTS function cannot be utilized. Set the UART1 CTS/RTS disable bit to "1".

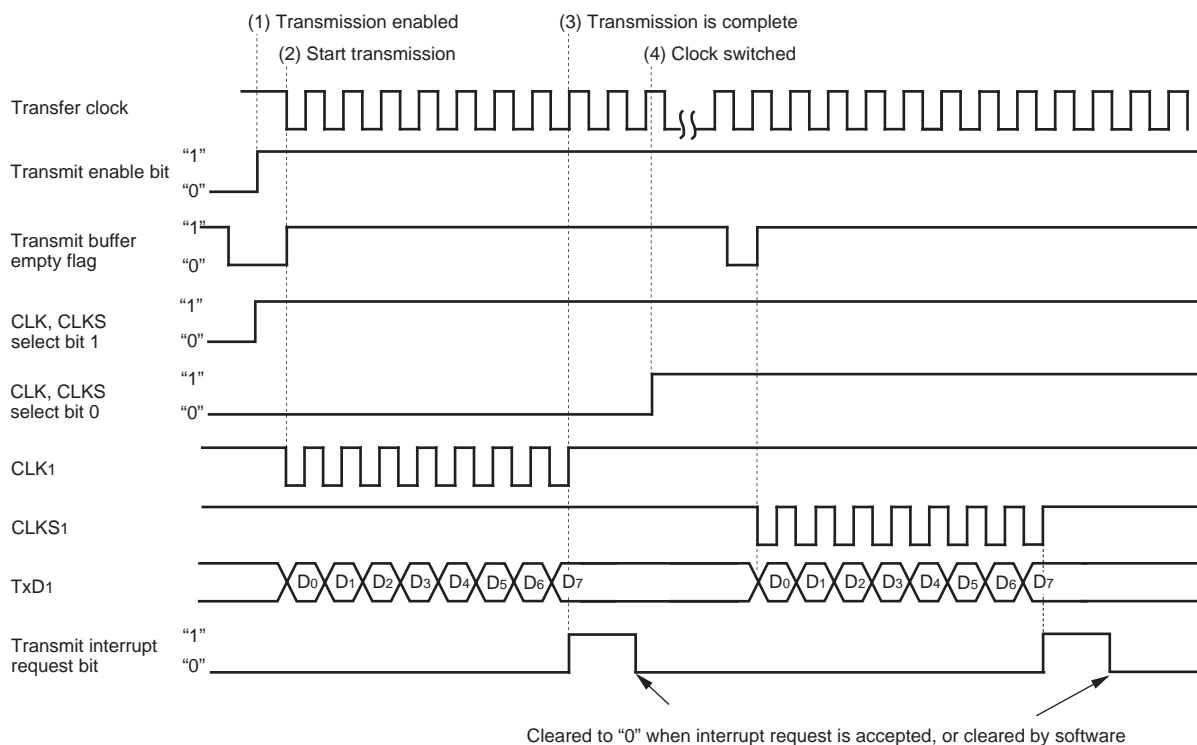
- Operation
- (1) Setting the transmit enable bit to "1" makes data transmissible status ready.
  - (2) When transmission data is written to the UART1 transmit buffer register, transmission data held in the UART1 transmit buffer register is transmitted to the UART1 transmit register in synchronization with the first falling edge of the transfer clock. At this time, the first bit of the transmission data is transmitted from the TxD1 pin. Then the data is transmitted bit by bit from the lower order in synchronization with the falling edges of the transfer clock.
  - (3) When transmission of 1-byte data is completed, the transmit register empty flag goes to "1", which indicates that the transmission is completed. The transfer clock stops at "H" level. At this time, the UART1 transmit interrupt request bit goes to "1".
  - (4) Setting CLK/CLKS select bit 1 to "1" and setting CLK/CLKS select bit 0 to "1" causes the CLKS1 pin to go to the transfer clock output pin. Change the transfer clock output pin when transmission is halted.

**Example of wiring**



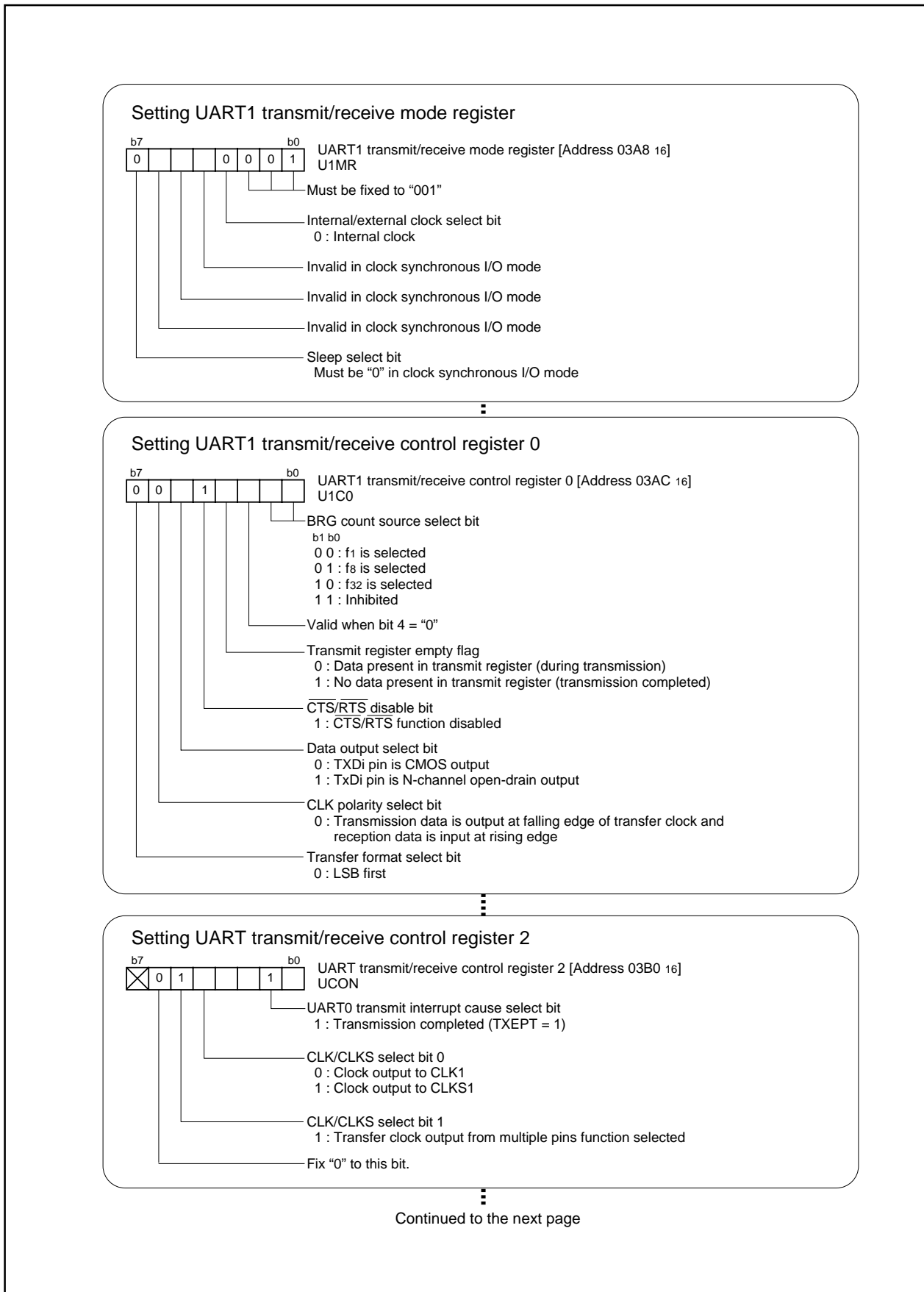
Note: This applies when performing only transmission with an internal clock selected in the clock synchronous serial I/O mode.

**Example of operation**

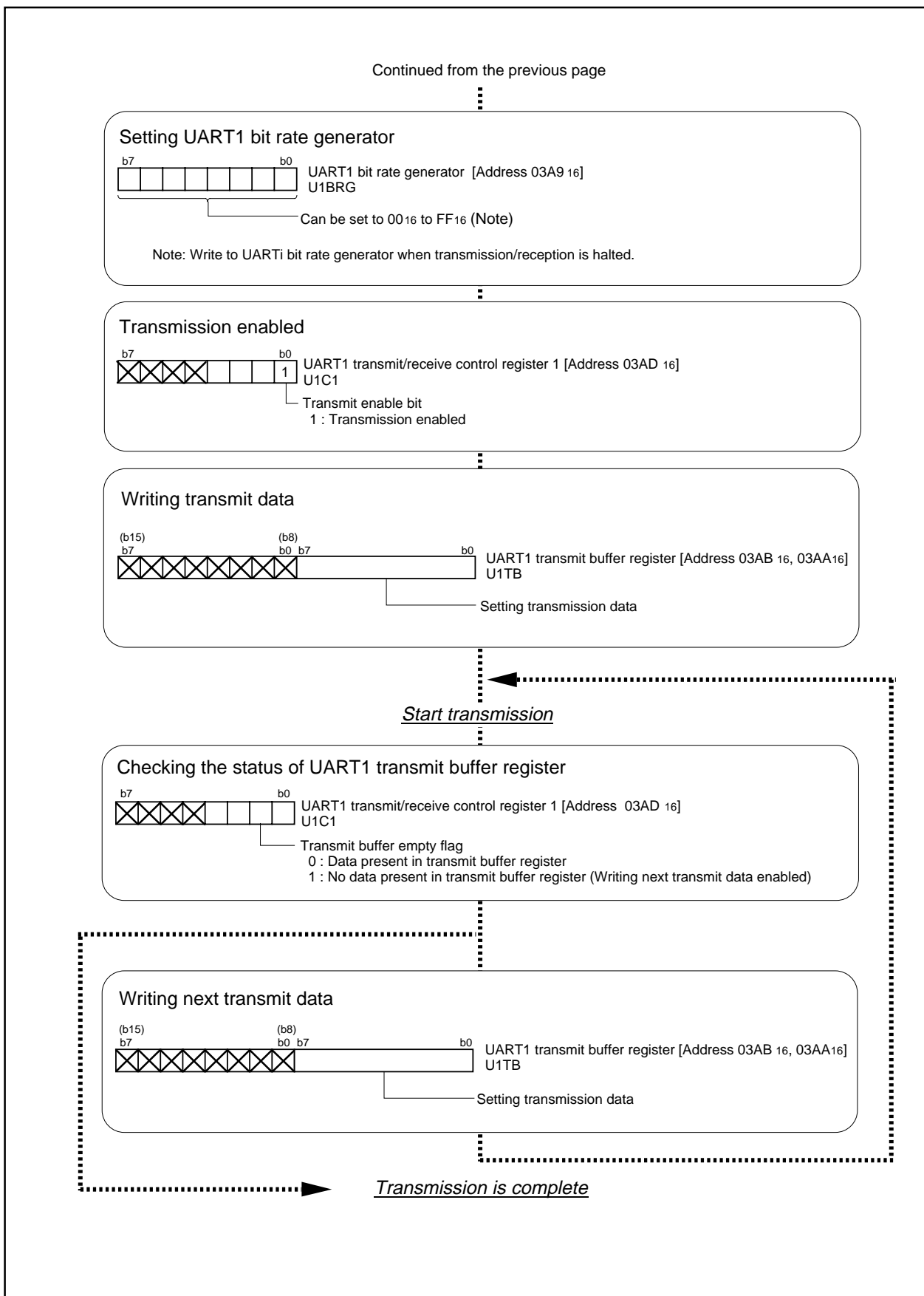


**Figure 2.4.8. Operation timing of transmission in clock-synchronous serial I/O mode, transfer clock output from multiple pins function selected**

## Clock-Synchronous Serial I/O



**Figure 2.4.9. Set-up procedure of transmission in clock-synchronous serial I/O mode, transfer clock output from multiple pins function selected (1)**



**Figure 2.4.10. Set-up procedure of transmission in clock-synchronous serial I/O mode, transfer clock output from multiple pins function selected (2)**

### 2.4.4 Operation of Serial I/O (reception in clock-synchronous serial I/O mode)

In receiving data in clock-synchronous serial I/O mode, choose functions from those listed in Table 2.4.3. Operations of the circled items are described below. Figure 2.4.11 shows the operation timing, and Figures 2.4.12 and 2.4.13 show the set-up procedures.

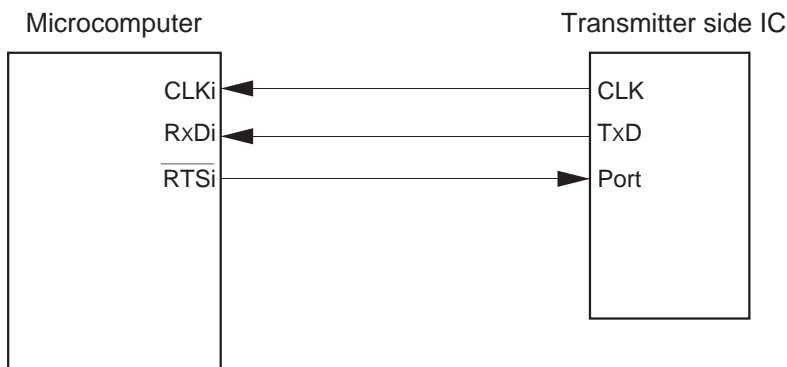
**Table 2.4.3. Chosed functions**

Item	Set-up		Item	Set-up	
Transfer clock source		Internal clock (f <sub>1</sub> / f <sub>8</sub> / f <sub>32</sub> )	Continuous receive mode	<input type="radio"/>	Disabled
	<input type="radio"/>	External clock (CLK <sub>i</sub> pin)		<input type="radio"/>	Enabled
RTS function	<input type="radio"/>	RTS function enabled	Output transfer clock to multiple pins (Note 1)	<input type="radio"/>	Not selected
		RTS function disabled		<input type="radio"/>	Selected
CLK polarity	<input type="radio"/>	Input reception data at the rising edge of the transfer clock			
		Input reception data at the falling edge of the transfer clock			
Transfer clock	<input type="radio"/>	LSB first			
		MSB first			

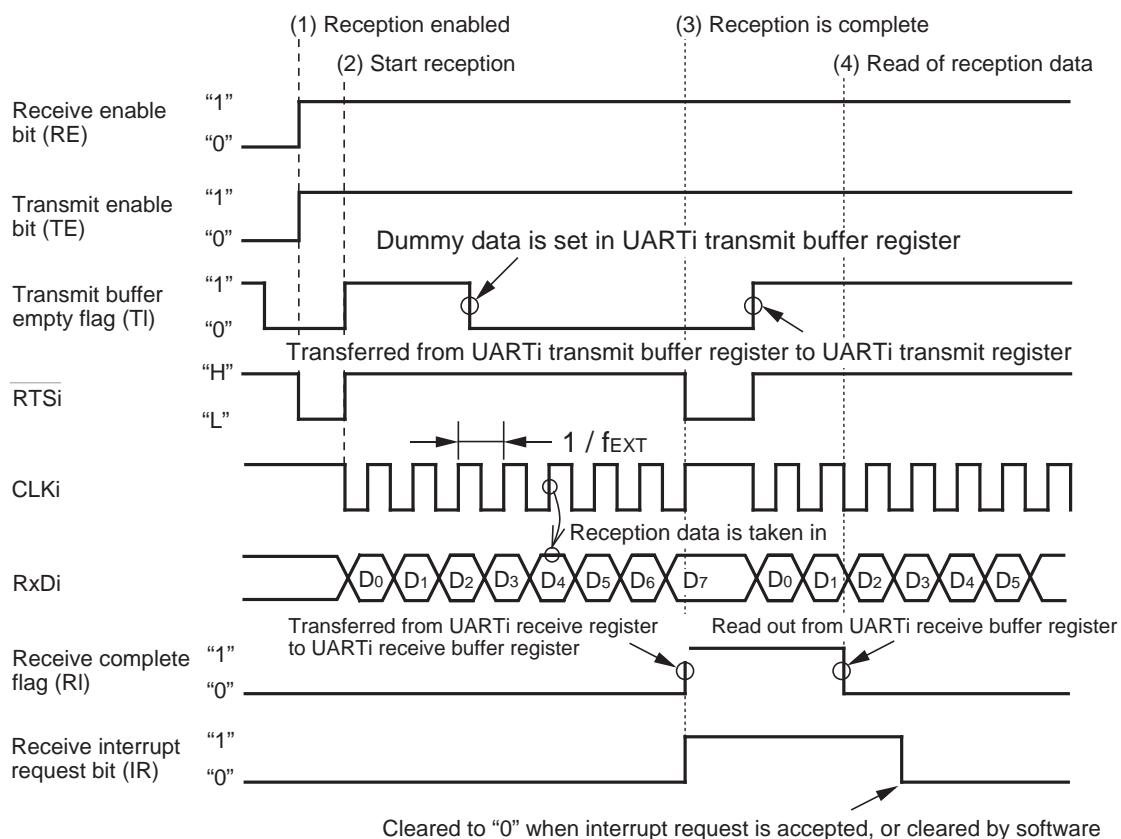
Note 1: This can be selected only when UART1 is used in combination with the internal clock. When this function is selected, UART1 CTS/RTS function cannot be utilized. Set the UART1 CTS/RTS disable bit to "1".

- Operation
- (1) Writing dummy data to the UART<sub>i</sub> transmit buffer register, setting the receive enable bit to "1", and the transmit enable bit to "1", makes the data receivable status ready. At this time, the output from the  $\overline{\text{RTS}}_i$  pin goes to "L" level, which informs the transmission side that the data receivable status is ready (output the transfer clock from the IC on the transmission side after checking that the  $\overline{\text{RTS}}$  output has gone to "L" level).
  - (2) In synchronization with the first rising edge of the transfer clock, the input signal to the RxDi pin is stored in the highest bit of the UART<sub>i</sub> receive register. Then, data is taken in by shifting right the content of the UART<sub>i</sub> reception data in synchronization with the rising edges of the transfer clock.
  - (3) When 1-byte data lines up in the UART<sub>i</sub> receive register, the content of the UART<sub>i</sub> receive register is transmitted to the UART<sub>i</sub> receive buffer register. The transfer clock stops at "H" level. At this time, the receive complete flag and the UART<sub>i</sub> receive interrupt request bit goes to "1".
  - (4) The receive complete flag goes to "0" when the lower-order byte of the UART<sub>i</sub> receive buffer register is read.

**Example of wiring**



**Example of operation**



Shown in ( ) are bit symbols.

The above timing applies to the following settings:

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

fEXT: frequency of external clock

Make sure that the following conditions are met when the CLKi pin input = "H" before data reception

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

**Figure 2.4.11. Operation timing of reception in clock-synchronous serial I/O mode**



## Clock-Synchronous Serial I/O

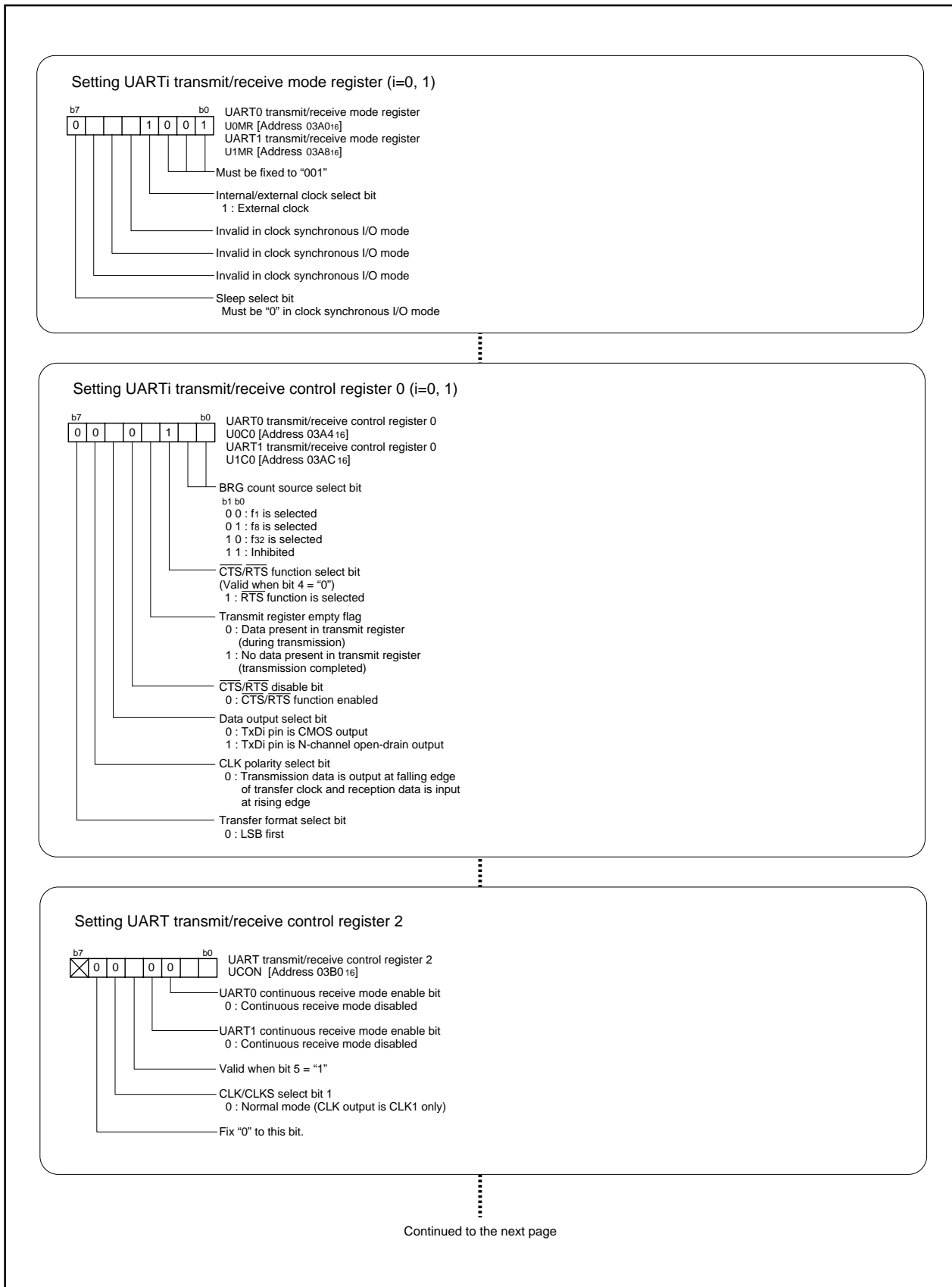


Figure 2.4.12. Set-up procedure of reception in clock-synchronous serial I/O mode (1)

Clock-Synchronous Serial I/O

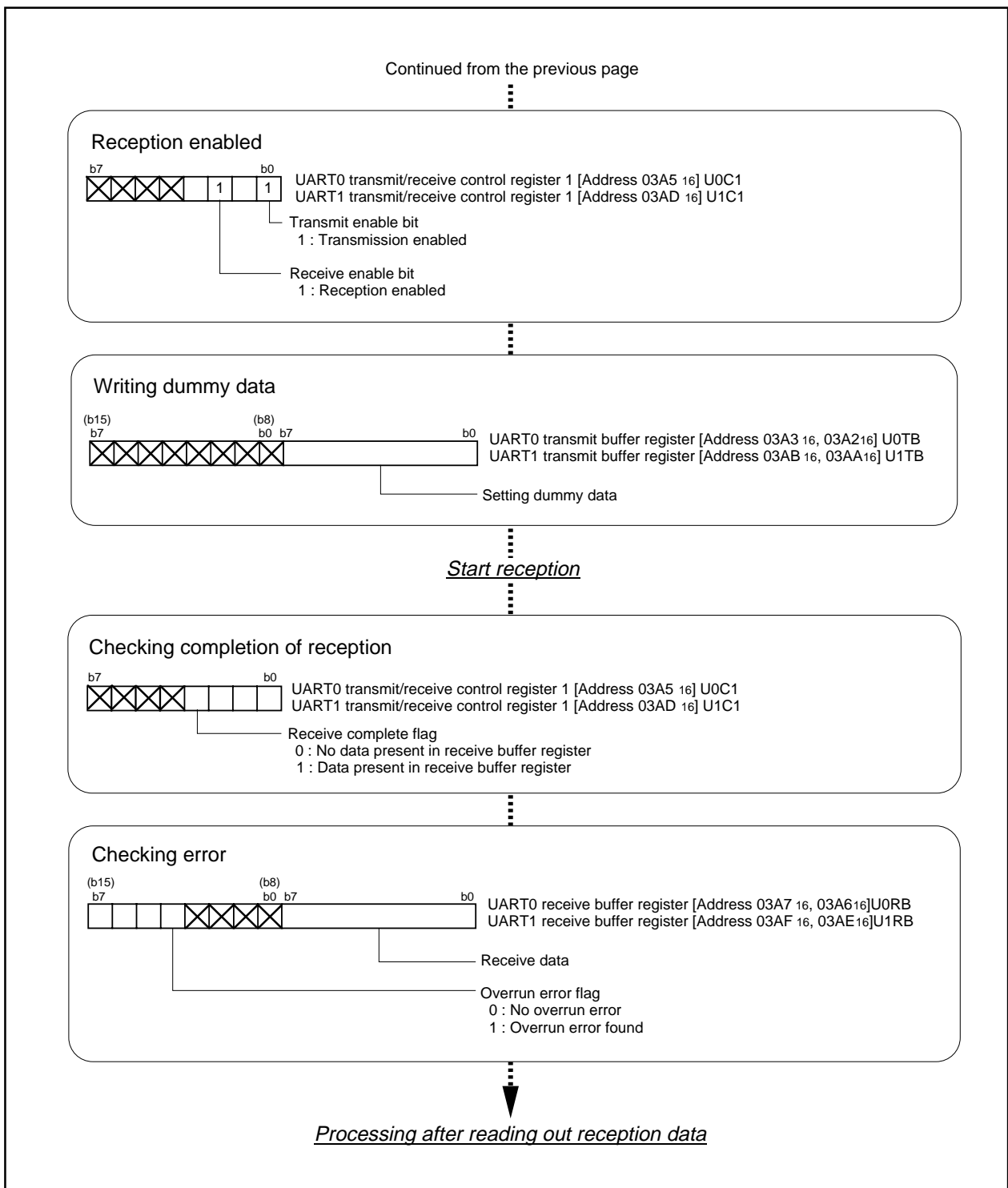


Figure 2.4.13. Set-up procedure of reception in clock-synchronous serial I/O mode (2)

### 2.4.5 Precautions for Serial I/O (in clock-synchronous serial I/O)

#### Transmission/reception

- (1) With an external clock selected, and choosing the  $\overline{\text{RTS}}$  function, the output level of the  $\overline{\text{RTSi}}$  pin goes to "L" when the data-receivable status becomes ready, which informs the transmission side that the reception has become ready. The output level of the  $\overline{\text{RTSi}}$  pin goes to "H" when reception starts. So if the  $\overline{\text{RTSi}}$  pin is connected to the  $\overline{\text{CTS}}$  pin on the transmission side, the circuit can transmission and reception data with consistent timing. With the internal clock, the  $\overline{\text{RTS}}$  function has no effect. Figure 2.4.14 shows an example of wiring.

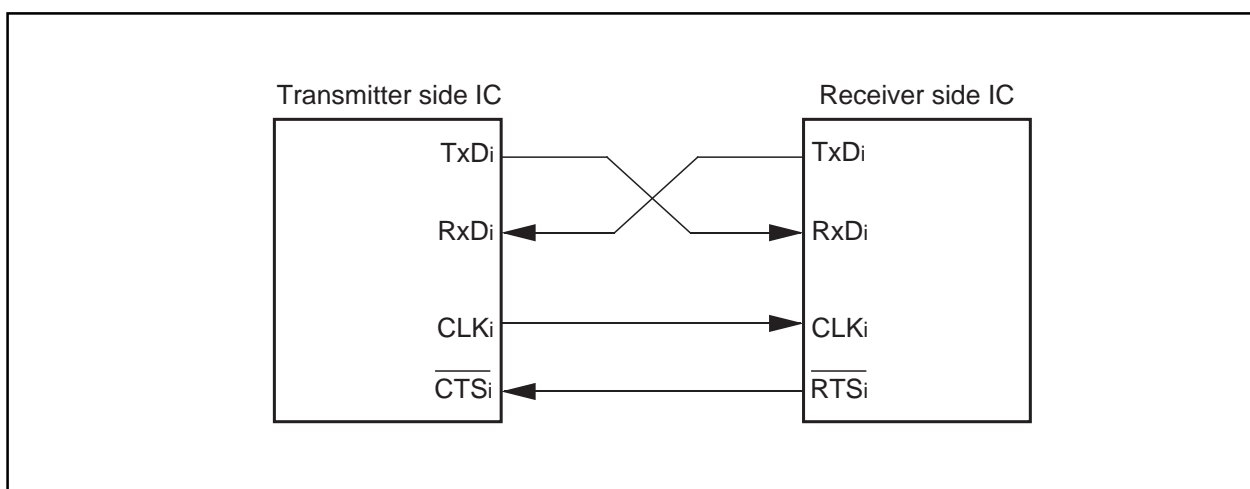


Figure 2.4.14. Example of wiring

## Clock-Synchronous Serial I/O

### Transmission

- (1) With an external clock selected, perform the following set-up procedure with the CLKi pin input level = "H" if the CLK polarity select bit = "0" or with the CLKi pin input level = "L" if the CLK polarity select bit = "1":
1. Set the transmit enable bit (to "1")
  2. Write transmission data to the UARTi transmit buffer register
  3. "L" level input to the  $\overline{\text{CTS}}_i$  pin (when the  $\overline{\text{CTS}}$  function is selected)

Reception (1) In operating the clock-synchronous serial I/O, operating a transmitter generates a shift clock. Fix settings for transmission even when using the device only for reception. Dummy data is output to the outside from the TxDi pin (transmission pin) when receiving data.

- (2) With the internal clock selected, setting the transmit enable bit to "1" (transmission-enabled status) and setting dummy data in the UARTi transmission buffer register generates a shift clock.

With the external clock selected, a shift clock is generated when the transmit enable bit is set to "1", dummy data is set in the UARTi transmit buffer register, and the external clock is input to the CLKi pin.

- (3) In receiving data in succession, an overrun error occurs when the next reception data is made ready in the UARTi receive register with the receive complete flag set to "1" (before the content of the UARTi receive buffer register is read), and overrun error flag is set to "1". In this instance, the next data is written to the UARTi receive buffer register, so handle with this problem by writing programs on transmission side and reception side so that the previous data is transmitted again.

If an overrun error occurs, the UARTi receive interrupt request bit does not go to "1".

- (4) To receive data in succession, set dummy data in the lower-order byte of the UARTi transmit buffer register every time reception is made.

- (5) With an external clock selected, perform the following set-up procedure with the CLKi pin input level = "H" if the CLK polarity select bit = "0" or with the CLKi pin input level = "L" if the CLK polarity select bit = "1":

1. Set receive enable bit (to "1")
2. Set transmit enable bit (to "1")
3. Write dummy data to the UARTi transmit buffer register

- (6) Output from the  $\overline{\text{RTS}}$  pin goes to "L" level as soon as the receive enable bit is set to "1". This is not related to the content of the transmit buffer empty flag or the content of the transmit enable bit.

Output from the  $\overline{\text{RTS}}$  pin goes to "H" level when reception starts, and goes to "L" level when reception is completed. This is not related to the content of the transmit buffer empty flag or the content of the receive complete flag.

## 2.5 Clock-Asynchronous Serial I/O (UART)

### 2.5.1 Overview

UART handles communications by means of character-by-character synchronization. The transmission side and the reception side are independent of each other, so full-duplex communication is possible. The following is an overview of the clock-asynchronous serial I/O.

#### (1) Transmission/reception format

Figure 2.5.1 shows the transmission/reception format, and Table 2.5.1 shows the names and functions of transmission data.

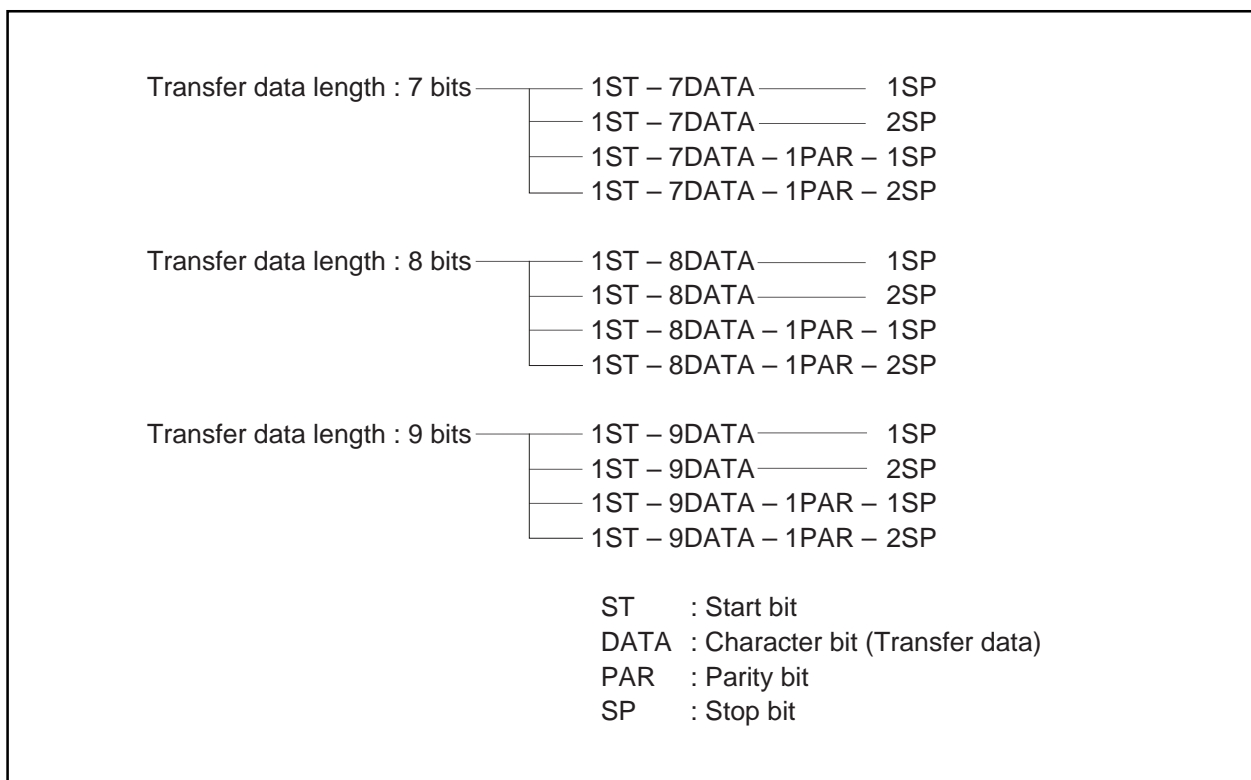


Figure 2.5.1. Transmission/reception format

Table 2.5.1. Transmission data names and functions

Name	Function
ST (start bit)	A 1-bit “L” signal to be added immediately before character bits. This bit signals the start of data transmission.
DATA (character bits)	Transmission data set in the UARTi transmit buffer register.
PAR (parity bit)	A signal to be added immediately after character bits so as to increase data reliability. The level of this signal so varies that the total number of 1's in character bits and this bit always becomes even or odd depending on which parity is chosen, even or odd.
SP (stop bit)	Either 1-bit or 2-bit “H” signal to be added immediately after character bits (after the parity bit if parity is checked). This / they signals the end of data transmission.

## UART

**(2) Transfer rate**

The divide-by-16 frequency, resulting from division in the bit rate generator (BRG), becomes the transfer rate. The count source for the bit rate generator can be selected from  $f_1$ ,  $f_8$ ,  $f_{32}$ , and the input from the CLK pin. Clocks  $f_1$ ,  $f_8$ ,  $f_{32}$  are derived by dividing the CPU's main clock by 1, 8, and 32 respectively.

**Table 2.5.2. Example of baud rate setting**

Baud rate (bps)	BRG's count source	System clock : 10MHz		System clock : 7.3728MHz	
		BRG's set value : n	Actual time (bps)	BRG's set value : n	Actual time (bps)
600	$f_8$	129 ( $81_{16}$ )	600	95 ( $5F_{16}$ )	600
1200	$f_8$	64 ( $40_{16}$ )	1201	47 ( $2F_{16}$ )	1200
2400	$f_8$	32 ( $20_{16}$ )	2367	23 ( $17_{16}$ )	2400
4800	$f_1$	129 ( $81_{16}$ )	4807	95 ( $5F_{16}$ )	4800
9600	$f_1$	64 ( $40_{16}$ )	9615	47 ( $2F_{16}$ )	9600
14400	$f_1$	42 ( $2A_{16}$ )	14534	31 ( $1F_{16}$ )	14400
19200	$f_1$	32 ( $20_{16}$ )	18939	23 ( $17_{16}$ )	19200
28800	$f_1$	21 ( $15_{16}$ )	28409	15 ( $F_{16}$ )	28800
31250	$f_1$	19 ( $13_{16}$ )	31250		

**(3) An error detection**

In clock-asynchronous serial I/O mode, detect errors are shown in Table 2.5.3.

**Table 2.5.3. Error detection**

Type of error	Description	When the flag turns on	How to clear the flag
Overrun error	<ul style="list-style-type: none"> <li>This error occurs when the next data lines up before the content of the UART<sub>i</sub> receive buffer register is read.</li> <li>The next data is written to the UART<sub>i</sub> receive buffer register.</li> <li>The UART<sub>i</sub> receive interrupt request bit does not go to "1".</li> </ul>	The error is detected when data is transferred from the UART <sub>i</sub> receive register to the UART <sub>i</sub> receive buffer register.	<ul style="list-style-type: none"> <li>Set the serial I/O mode select bits to "0002".</li> <li>Set the receive enable bit to "0".</li> </ul>
Framing error	<ul style="list-style-type: none"> <li>This error occurs when the stop bit falls short of the set number of stop bits.</li> </ul>		<ul style="list-style-type: none"> <li>Set the serial I/O mode select bits to "0002".</li> <li>Set the receive enable bit to "0".</li> </ul>
Parity error	<ul style="list-style-type: none"> <li>With parity enabled, this error occurs when the total number of 1's in character bits and the parity bit is different from the specified number.</li> </ul>		<ul style="list-style-type: none"> <li>Read the lower-order byte of the UART<sub>i</sub> receive buffer register.</li> </ul>
Error-sum flag	<ul style="list-style-type: none"> <li>This flag turns on when any error (overrun, framing, or parity) is detected.</li> </ul>		<ul style="list-style-type: none"> <li>When all error (overrun, framing, and parity) are removed, the flag is cleared.</li> </ul>

**(4) How to deal with an error**

When receiving data, read an error flag and reception data simultaneously to determine which error has occurred. If the data read is erroneous, initialize the error flag and the UARTi receive buffer register, then receive the data again.

**To initialize the UARTi receive buffer register**

1. Set the receive enable bit to "0" (disable reception).
2. Set the receive enable bit to "1" again (enable reception).

To transmit data again due to an error on the reception side, set the UARTi transmit buffer register again, then transmit the data again.

**To set the UARTi transmit buffer register again**

1. Set the serial I/O mode select bits to "0002" (invalidate serial I/O).
2. Set the serial I/O mode select bits again.
3. Set the transmit enable bit to "1" (enable transmission), then set transmission data in the UARTi transmit buffer register.

**(5) Functions selection**

In operating UART, the following functions can be used:

**(a) CTS/RTS function**

$\overline{\text{CTS}}$  function is a function in which an external IC can start transmission/reception by means of inputting an "L" level to the  $\overline{\text{CTS}}$  pin. The  $\overline{\text{CTS}}$  pin input level is detected when transmission/reception starts, so if the level is gone to "H" while transmission/reception is in progress, transmission/reception stops at the next data.

$\overline{\text{RTS}}$  function is a function to inform an external IC that  $\overline{\text{RTS}}$  pin output level has changed to "L" when reception is ready.  $\overline{\text{RTS}}$  regoes to "H" at the falling edge of the transfer clock.

When using clock-asynchronous serial I/O, choose one of three types of  $\overline{\text{CTS}}/\overline{\text{RTS}}$  functions.

- |   |  |
|---|--|
| • CTS/RTS functions disabled                    | $\overline{\text{CTS}}/\overline{\text{RTS}}$ pin is a programmable I/O port.                    |
| • $\overline{\text{CTS}}$ function only enabled | $\overline{\text{CTS}}/\overline{\text{RTS}}$ pin performs the $\overline{\text{CTS}}$ function. |
| • RTS function only enabled                     | $\overline{\text{CTS}}/\overline{\text{RTS}}$ pin performs the $\overline{\text{RTS}}$ function. |

**(b) Sleep mode**

Sleep mode is a mode in which data is transferred to a particular microcomputer among those connected by use of clock-asynchronous serial I/O devices.

**(c) Data logic select function**

This function is to reserve data when writing to transmit buffer register or reading from receive buffer register.

The following are examples in which functions (a) to (c) are chosen:

- Transmission WITH:  $\overline{\text{CTS}}$  function, WITHOUT: other functions ..... P254
- Reception WITH:  $\overline{\text{RTS}}$  function, WITHOUT: other functions ..... P258

**(6) Input to the serial I/O and the direction register**

To input an external signal to the serial I/O, set the direction register of the relevant port to input.

**(7) Pins related to the serial I/O**

- $\overline{\text{CTS}}_0$ ,  $\overline{\text{CTS}}_1$  pins :Input pins for the  $\overline{\text{CTS}}$  function
- $\overline{\text{RTS}}_0$ ,  $\overline{\text{RTS}}_1$  pins :Output pins for the  $\overline{\text{RTS}}$  function
- $\text{CLK}_0$ ,  $\text{CLK}_1$  pins :Input pins for the transfer clock
- $\text{RxD}_0$ ,  $\text{RxD}_1$  pins :Input pins for data
- $\text{TxD}_0$ ,  $\text{TxD}_1$  pins :Output pins for data/



**(8) Registers related to the serial I/O**

Figure 2.5.2 shows the memory map of serial I/O-related registers, and Figures 2.5.3 to 2.5.5 show UARTi-related registers.

0051 <sub>16</sub>	UART0 transmit interrupt control register (S0TIC)
0052 <sub>16</sub>	UART0 receive interrupt control register (S0RIC)
0053 <sub>16</sub>	UART1 transmit interrupt control register(S1TIC)
0054 <sub>16</sub>	UART1 receive interrupt control register(S1RIC)
~	
03A0 <sub>16</sub>	UART0 transmit/receive mode register (U0MR)
03A1 <sub>16</sub>	UART0 bit rate generator (U0BRG)
03A2 <sub>16</sub>	UART0 transmit buffer register (U0TB)
03A3 <sub>16</sub>	
03A4 <sub>16</sub>	UART0 transmit/receive control register 0 (U0C0)
03A5 <sub>16</sub>	UART0 transmit/receive control register 1 (U0C1)
03A6 <sub>16</sub>	UART0 receive buffer register (U0RB)
03A7 <sub>16</sub>	
03A8 <sub>16</sub>	UART1 transmit/receive mode register (U1MR)
03A9 <sub>16</sub>	UART1 bit rate generator (U1BRG)
03AA <sub>16</sub>	UART1 transmit buffer register (U1TB)
03AB <sub>16</sub>	
03AC <sub>16</sub>	UART1 transmit/receive control register 0 (U1C0)
03AD <sub>16</sub>	UART1 transmit/receive control register 1 (U1C1)
03AE <sub>16</sub>	UART1 receive buffer register (U1RB)
03AF <sub>16</sub>	
03B0 <sub>16</sub>	UART transmit/receive control register 2 (UCON)

**Figure 2.5.2. Memory map of UARTi-related registers**

UART

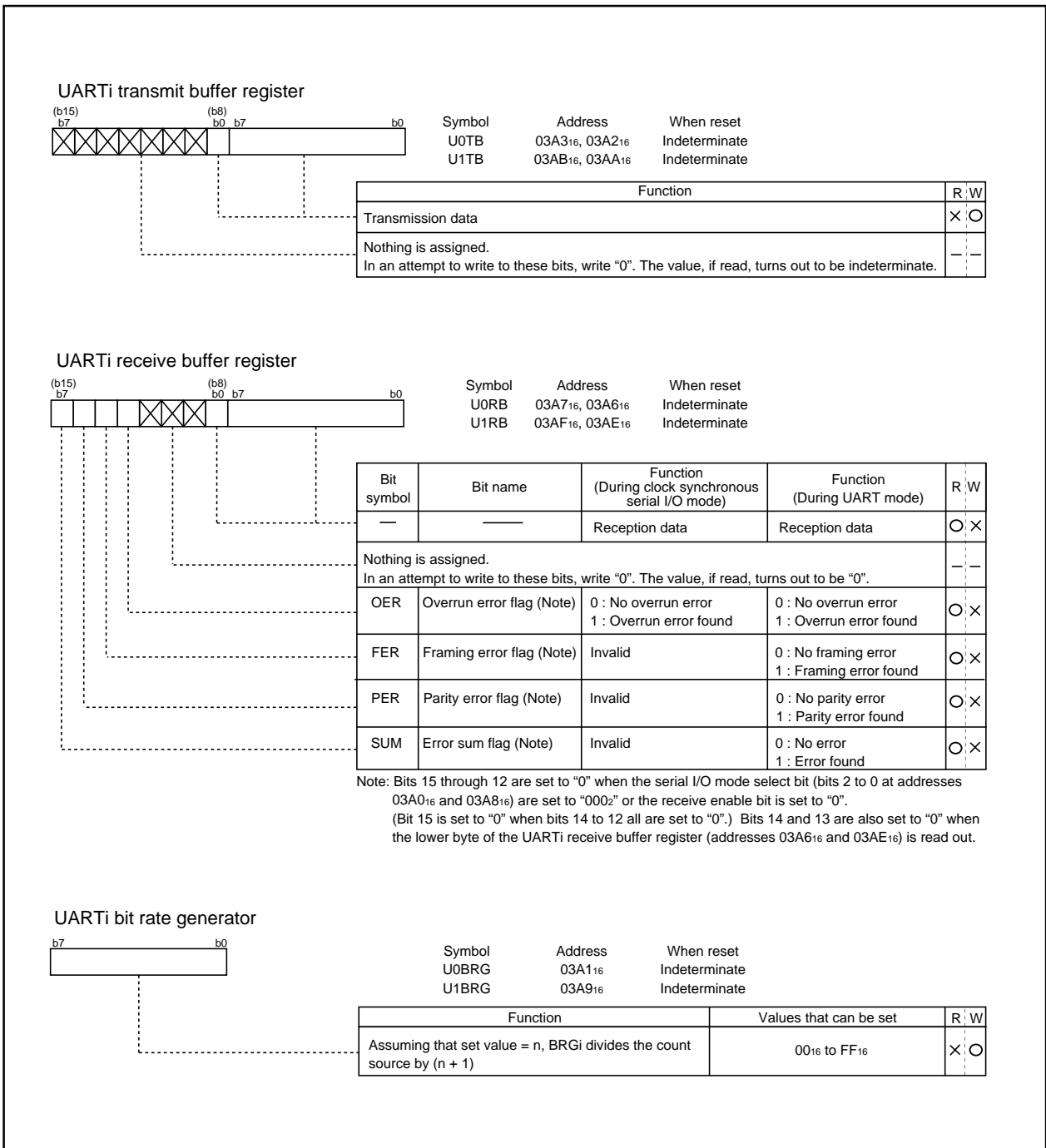


Figure 2.5.3. UARTi-related registers (1)

## UART

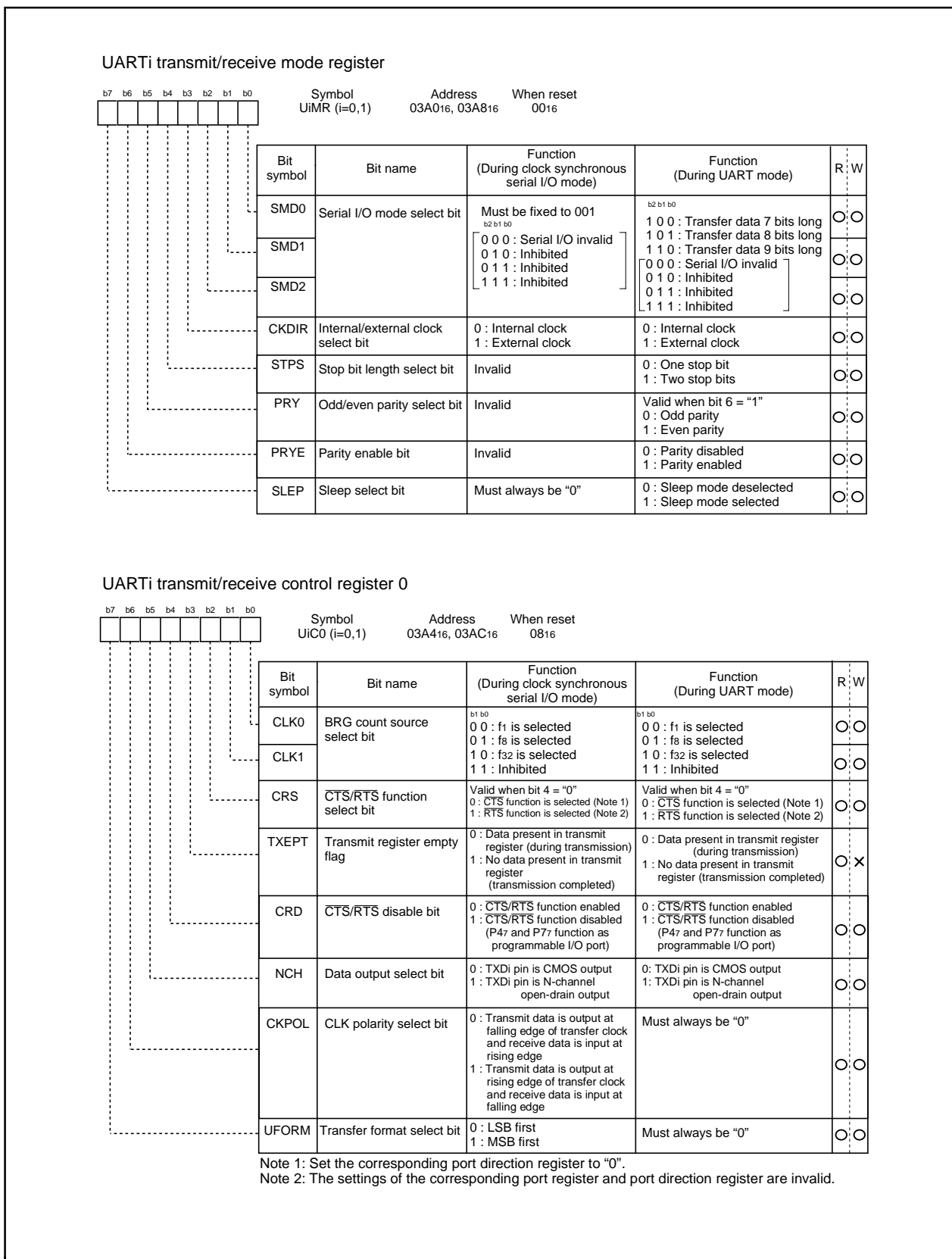


Figure 2.5.4. UARTi-related registers (2)

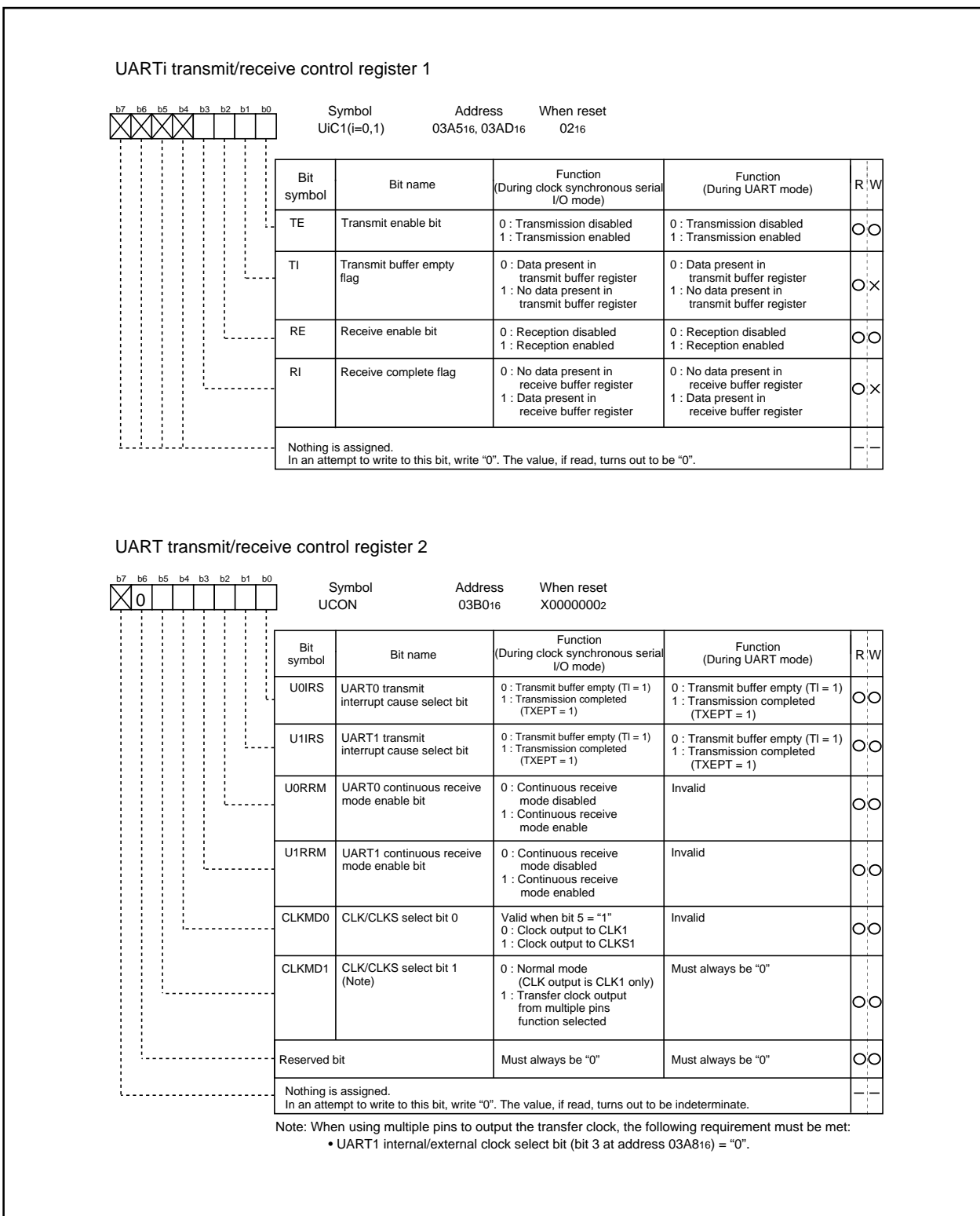


Figure 2.5.5. UARTi-related registers (3)

## 2.5.2 Operation of Serial I/O (transmission in UART mode)

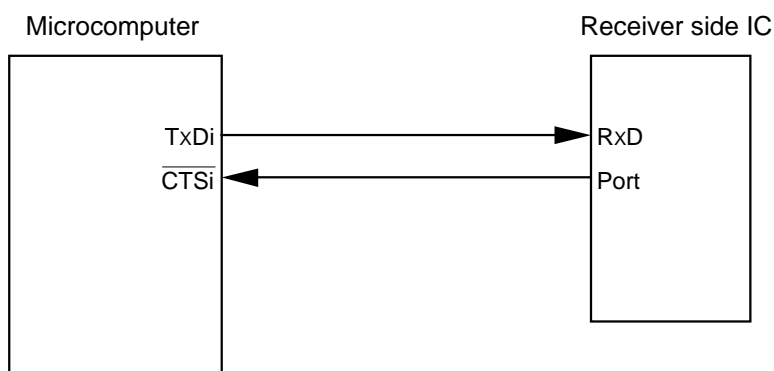
In transmitting data in UART mode, choose functions from those listed in Table 2.5.4. Operations of the circled items are described below. Figure 2.5.6 shows the operation timing, and Figures 2.5.7 and 2.5.8 show the set-up procedures.

**Table 2.5.4. Chosed functions**

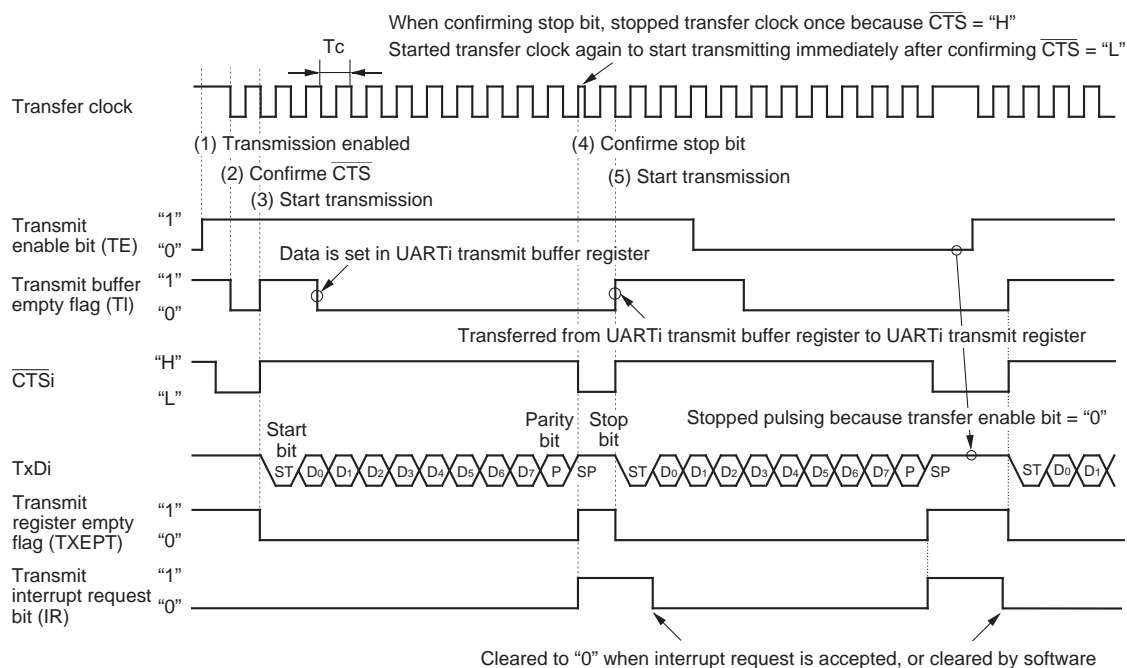
Item	Set-up	
Transfer clock source	<input type="radio"/>	Internal clock ( $f_1 / f_8 / f_{32}$ )
	<input type="radio"/>	External clock (CLKi pin)
$\overline{\text{CTS}}$ function	<input type="radio"/>	$\overline{\text{CTS}}$ function enabled
	<input type="radio"/>	$\overline{\text{CTS}}$ function disabled
Transmission interrupt factor	<input type="radio"/>	Transmission buffer empty
	<input type="radio"/>	Transmission complete
Sleep mode	<input type="radio"/>	Sleep mode off
	<input type="radio"/>	Sleep mode selected

- Operation
- (1) Setting the transmit enable bit to "1" and writing transmission data to the UARTi transmit buffer register readies the data transmissible status.
  - (2) When input to the  $\overline{\text{CTS}}_i$  pin goes to "L", transmission starts (the  $\overline{\text{CTS}}_i$  pin needs to be controlled on the reception side).
  - (3) Transmission data held in the UARTi transmit buffer register is transmitted to the UARTi transmit register. At this time, the first bit (the start bit) of the transmission data is transmitted from the TxDi pin. Then, data is transmitted, bit by bit, in sequence: LSB, ..., MSB, parity bit, and stop bit(s).
  - (4) When the stop bit(s) is (are) transmitted, the transmit register empty flag goes to "1", which indicates that transmission is completed. At this time, the UARTi transmit interrupt request bit goes to "1". The transfer clock stops at "H" level.
  - (5) If the transmission condition of the next data is ready when transmission is completed, a start bit is generated following to stop bit(s), and the next data is transmitted.

Example of wiring



Example of operation



Shown in ( ) are bit symbols.

The above timing applies to the following settings :

- Parity is enabled.
- One stop bit.
- CTS function is selected.
- Transmit interrupt cause select bit = "1".

$$T_c = 16(n + 1) / f_i \text{ or } 16(n + 1) / f_{EXT}$$

$f_i$  : 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 2.5.6. Operation timing of transmission in UART mode**

UART

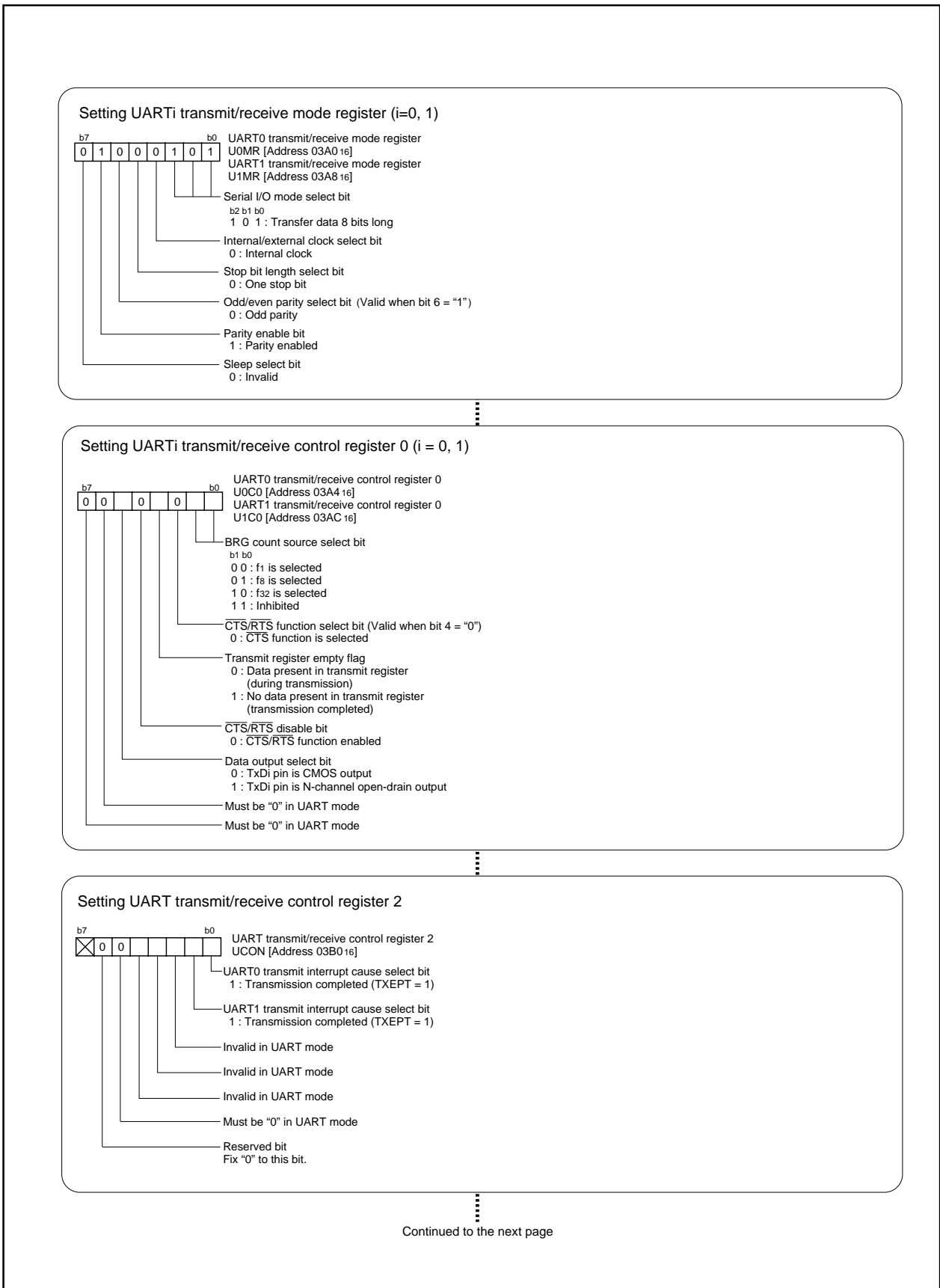


Figure 2.5.7. Set-up procedure of transmission in UART mode (1)

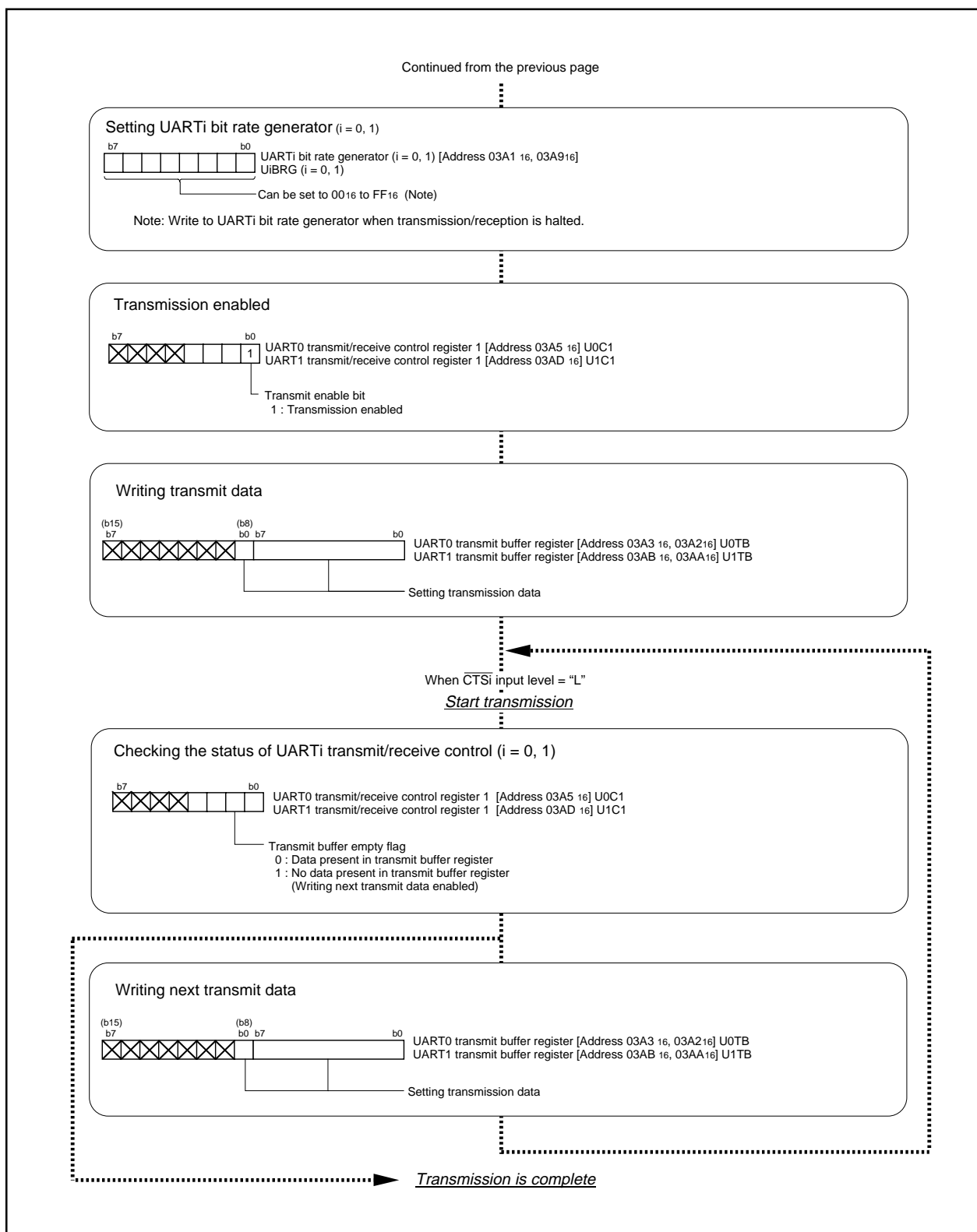


Figure 2.5.8. Set-up procedure of transmission in UART mode (2)



### 2.5.3 Operation of Serial I/O (reception in UART mode)

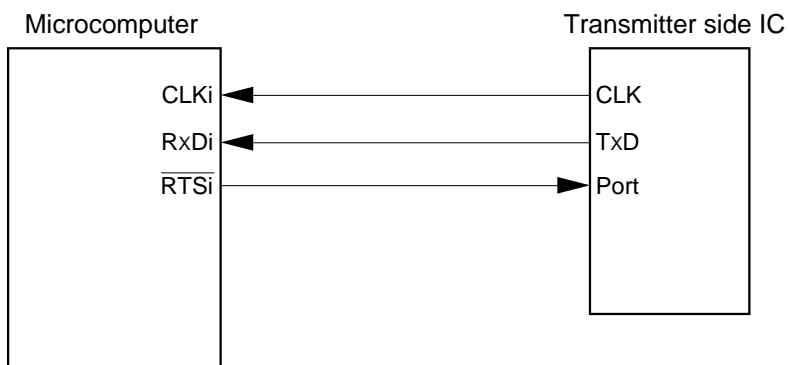
In receiving data in UART mode, choose functions from those listed in Table 2.5.5. Operations of the circled items are described below. Figure 2.5.9 shows the operation timing, and Figures 2.5.10 and 2.5.11 show the set-up procedures.

**Table 2.5.5. Chosed functions**

Item	Set-up	
Transfer clock source		Internal clock ( $f_1 / f_8 / f_{32}$ )
	○	External clock (CLKi pin)
RTS function	○	$\overline{\text{RTS}}$ function enabled
		$\overline{\text{RTS}}$ function disabled
Sleep mode	○	Sleep mode off
		Sleep mode selected

- Operation
- (1) Setting the receive enable bit to "1" readies data-receivable status. At this time, output from the  $\overline{\text{RTSi}}$  pin goes to "L" level to inform the transmission side that the receivable status is ready.
  - (2) When the first bit (the start bit) of reception data is received from the RxDi pin, output from the  $\overline{\text{RTS}}$  goes to "H" level. Then, data is received, bit by bit, in sequence: LSB, ..., MSB, and stop bit(s).
  - (3) When the stop bit(s) is (are) received, the content of the UARTi receive register is transmitted to the UARTi receive buffer register.  
At this time, the receive complete flag goes to "1" to indicate that the reception is completed, the UARTi receive interrupt request bit goes to "1", and output from the  $\overline{\text{RTS}}$  pin goes to "L" level.
  - (4) The receive complete flag goes to "0" when the lower-order byte of the UARTi receive buffer register is read.

Example of wiring



Example of operation

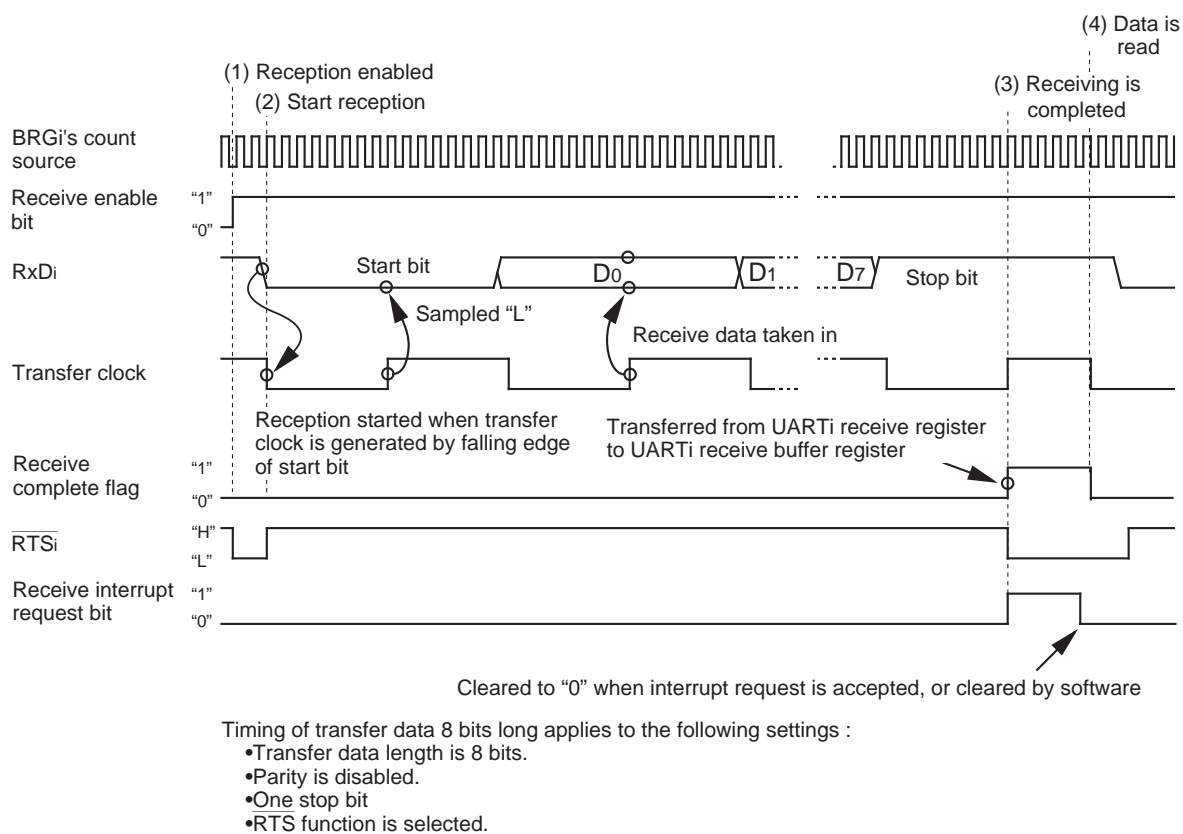


Figure 2.5.9. Operation timing of reception in UART mode

UART

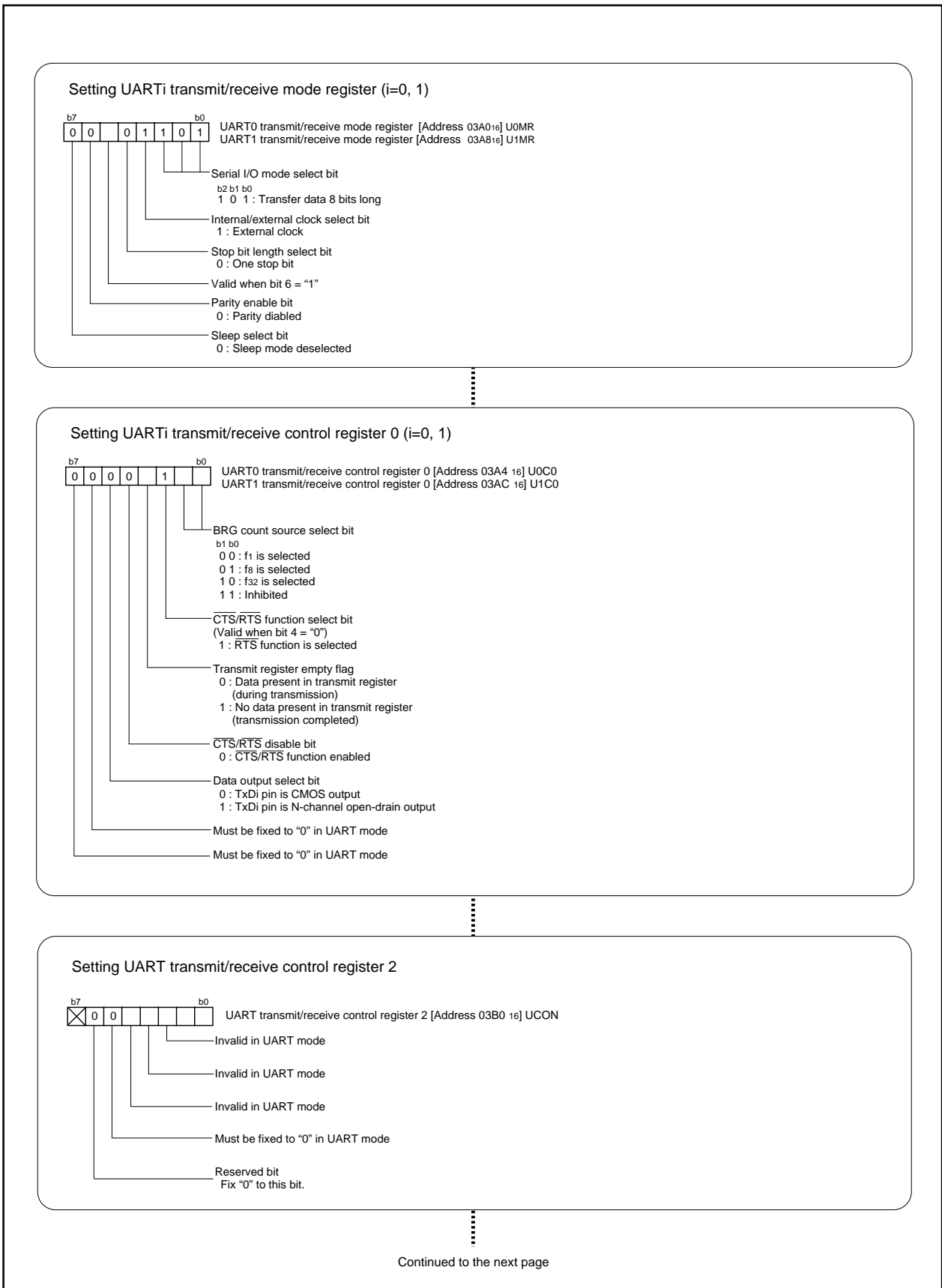
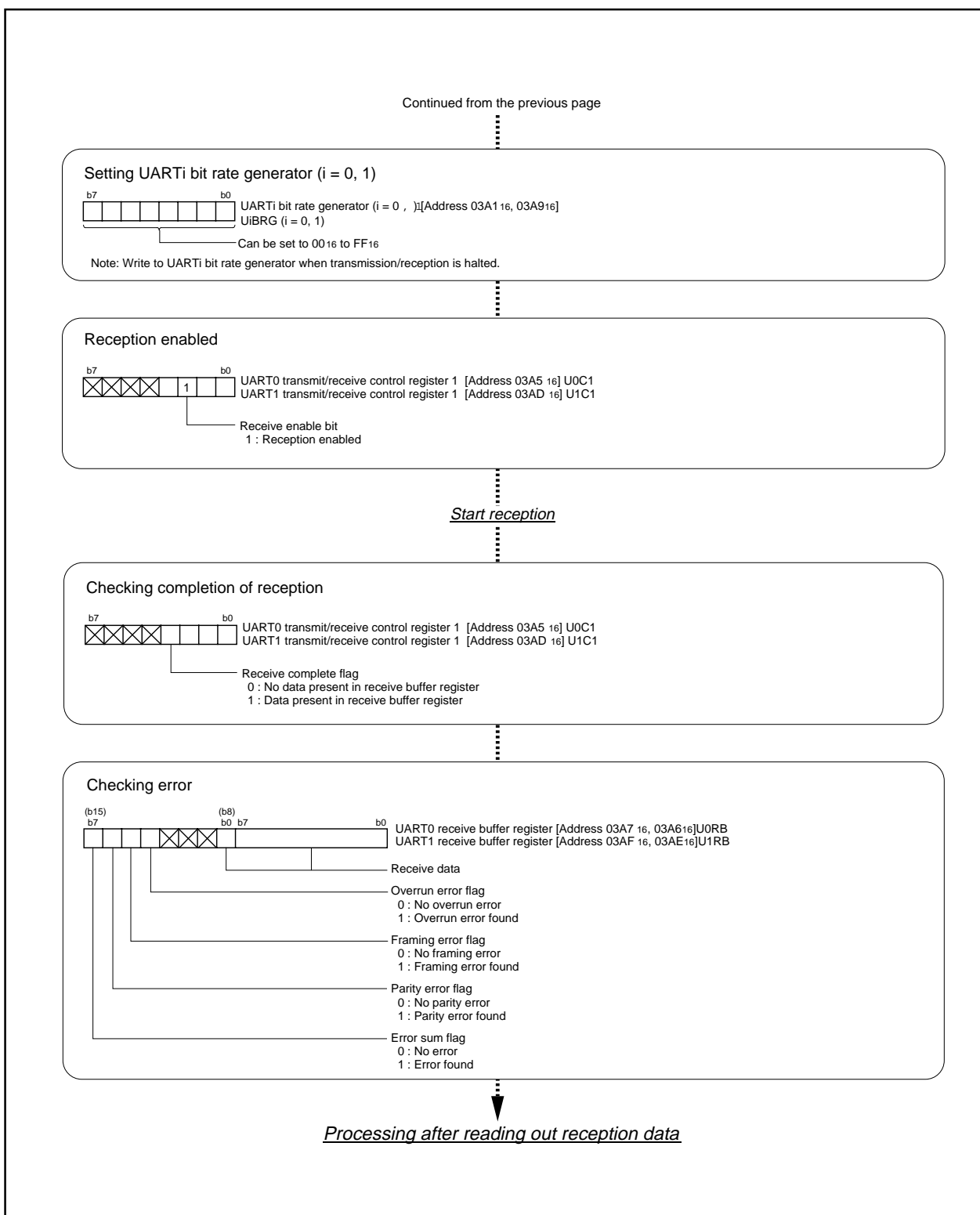


Figure 2.5.10. Set-up procedure of reception in UART mode (1)



**Figure 2.5.11. Set-up procedure of reception in UART mode (2)**

## 2.6 Serial I/O2

### 2.6.1 Overview

Serial I/O2 performs 8-bit data serial communication, synchronized with the clocks. In the automatic transfer serial I/O mode, serial communication of up to 256 bytes can be continuously performed without use of the CPU. The following is the Serial I/O2 overview.

#### (1) Transfer format

Transfer format is 8-bit data.

#### (2) Transfer rate

When selecting an internal clock as the transfer clock, the transfer rate is the division ratio selected by the internal synchronous clock selection bits. Any one of  $f(X_{IN})/4$ ,  $f(X_{IN})/8$ ,  $f(X_{IN})/16$ ,  $f(X_{IN})/32$ ,  $f(X_{IN})/64$ ,  $f(X_{IN})/128$  or  $f(X_{IN})/256$  can be selected by the internal synchronous clock selection bits. When selecting an external clock as the transfer clock, the transfer rate is the frequency of the clock input to the CLK pin.

#### (3) Automatic transfer serial I/O mode

Clock synchronous communication, which does not depend on the CPU, can be continuously performed up to 256 bytes.

#### (4) Selection function

The following selection functions can be applied to Serial I/O2.

##### (a) SSTB2 output (for selecting internal synchronous clock)

- STB function invalid: SSTB2 output pin is used as a programmable I/O pin.
- STB function valid: SSTB2 output pin functions as SSTB2 or  $\overline{\text{SSTB2}}$  output.

##### (b) SBUSY2 input/output

- SBUSY2 input/output function invalid: SBUSY2 pin is used as a programmable I/O pin.
- SBUSY2 input/output function valid: SBUSY2 pin functions as input/output of SBUSY2 or  $\overline{\text{SBUSY2}}$ .

##### (c) SRDY2 input/output

- SRDY2 input/output function invalid: SRDY2 pin is used as a programmable I/O pin.
- SRDY2 input/output function valid: SRDY2 pin functions as input/output of SRDY2 or  $\overline{\text{SRDY2}}$ .

##### (d) SOUT2 P-channel output disable (invalid for P94 as I/O port)

The SOUT2 output pin can be switched between C-MOS 3 state and N-channel open-drain when in the 8-bit or the automatic transfer serial I/O mode. The mode is selected by the serial transfer select bits.

##### (e) LSB first/MSB first

This function switches the starting bit for the transmission/reception; either bit 0 or bit 7. The following two types can be selected with the transfer direction select bit:

- LSB first: transmission/reception begins with from bit 0.
- MSB first: transmission/reception begins with from bit 7.

**(f) Transfer mode**

Either the full duplex mode or the transmit-only mode can be selected. The SIN2 pin can be used as a programmable input/output port in the transmit-only mode.

**(g) Plural transfer clock input/output pin**

This function switches the pins for transfer clock input/output. By switching the transfer clock pins, data can be transmitted/received to two external ICs in a time-sharing manner.

**(h) SOUT2 pin control**

This pin selects either output active (value of last transmitted data or undefined value) or high-impedance as the SOUT2 pin state for non-transfer periods (i.e. before and after serial transfers).

**(5) Input to serial I/O2 and direction register**

When inputting external signals to Serial I/O2, set the corresponding port direction register to input.

**(6) Serial I/O2-related pins**

**(a) SSTB2 pin:** Output pin for STB and  $\overline{\text{STB}}$  functions.

**(b) SBUSY2 pin:** Input/output pin for BUSY and  $\overline{\text{BUSY}}$  functions.

**(c) SRDY2 pin:** Input/output pin for RDY and  $\overline{\text{RDY}}$  functions.

**(d) SCLK21, SCLK22 pins:** Input/output pins for transfer clocks. Pin is selectable by user.

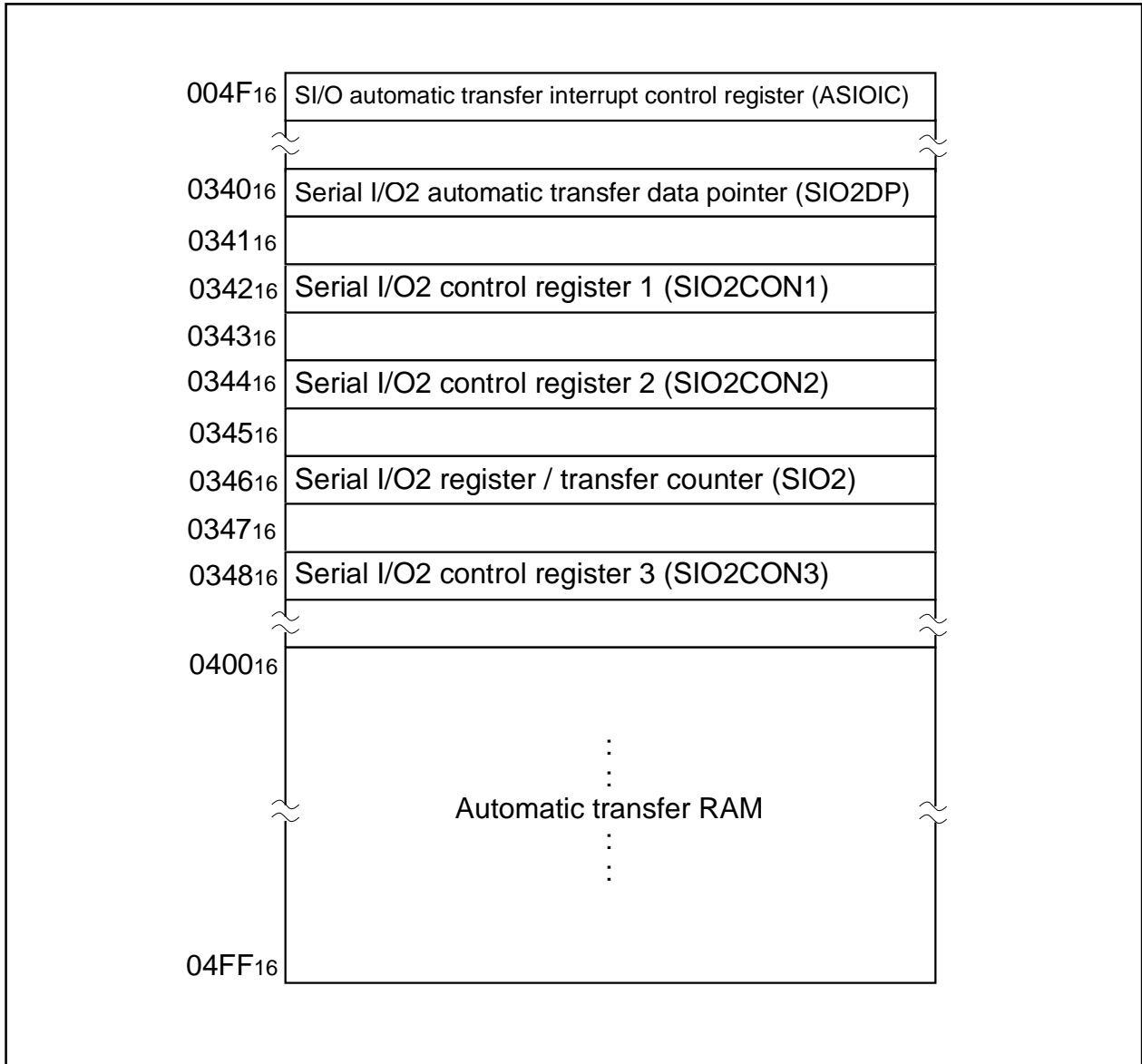
**(e) SIN2 pin:** Data input pin.

**(f) SOUT2 pin:** Data output pin.

**(7) Registers related to Serial I/O2**

Figure 2.6.1 shows the memory map of Serial I/O2 related-registers.

Figures 2.6.2 and 2.6.3 show the Serial I/O2 related-registers.



**Figure 2.6.1. Memory map of Serial I/O2 related-registers**

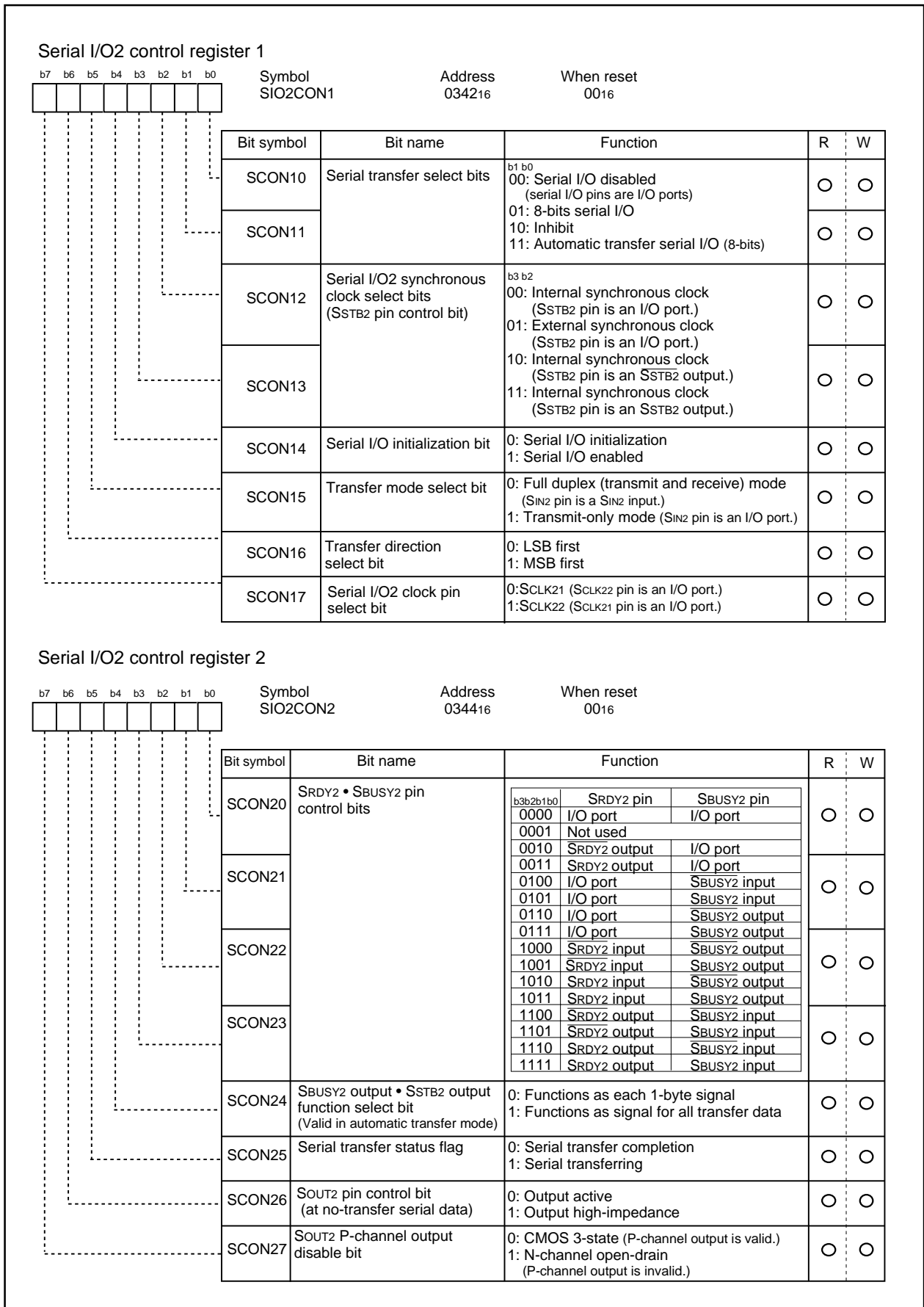


Figure 2.6.2. Serial I/O2 related-registers (1)



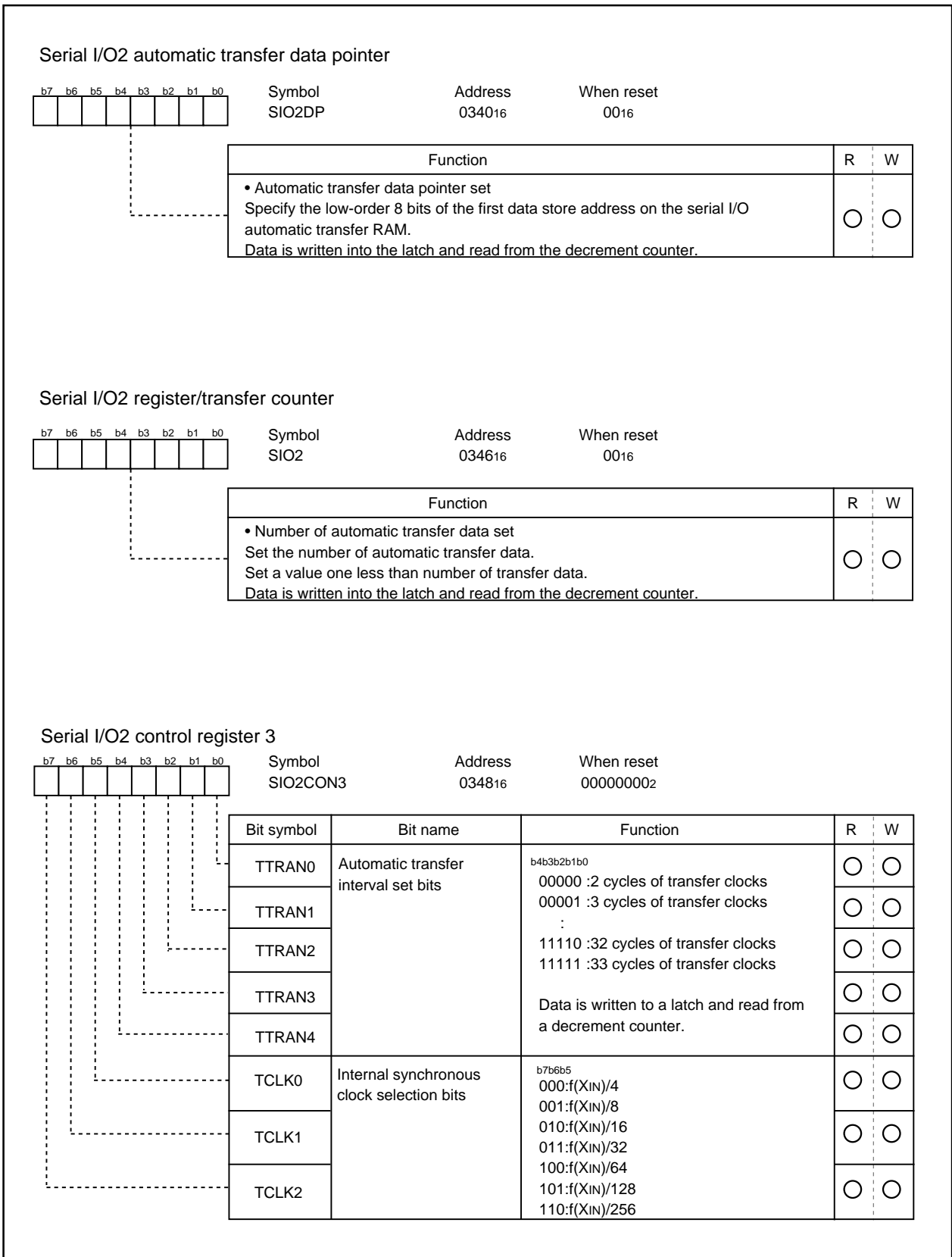


Figure 2.6.3. Serial I/O2 related-registers (2)

### 2.6.2 Serial I/O2 connection examples

#### (1) Control of peripheral IC equipped with CS pin

Figure 2.6.4 shows connection examples with peripheral ICs which have the CS pin. The automatic transfer function can be used in all examples.

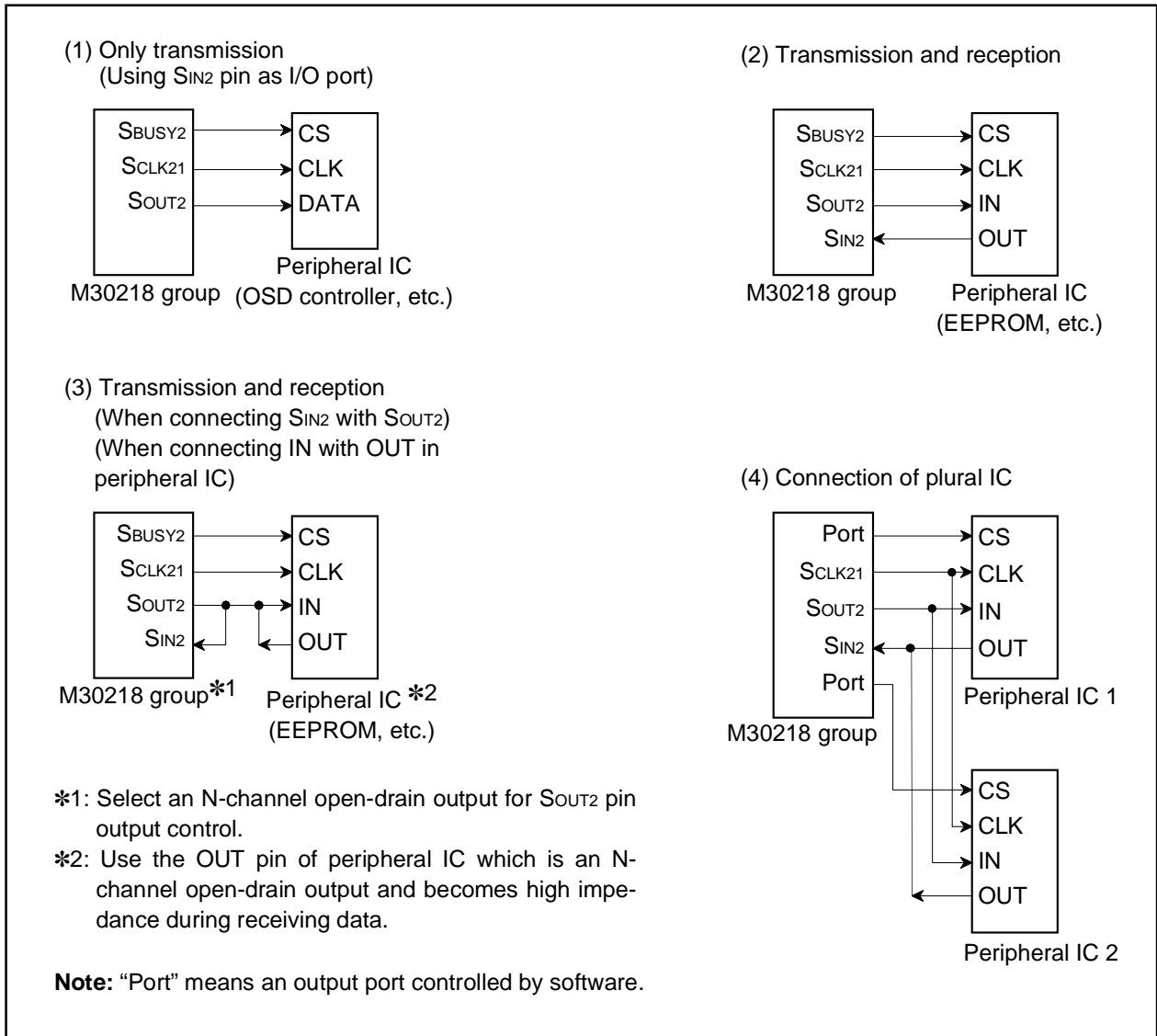
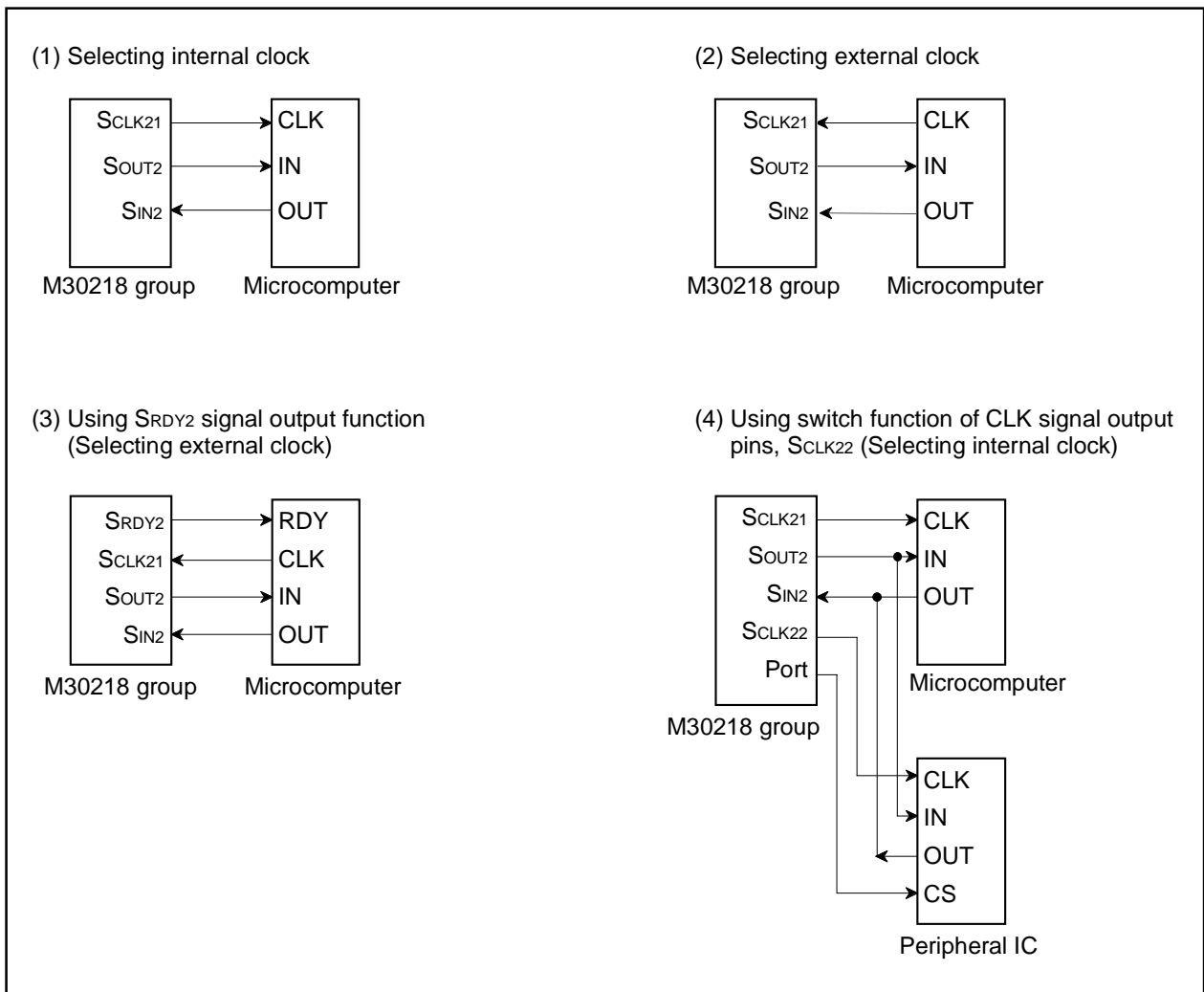


Figure 2.6.4. Serial I/O2 onnection examples (1)

**(2) MCU connections**

Figure 2.6.5 shows connection examples with other MCUs.



**Figure 2.6.5. Serial I/O2 onnection examples (2)**

### 2.6.3 Serial I/O2 modes

Figure 2.6.6 shows Serial I/O2 modes.

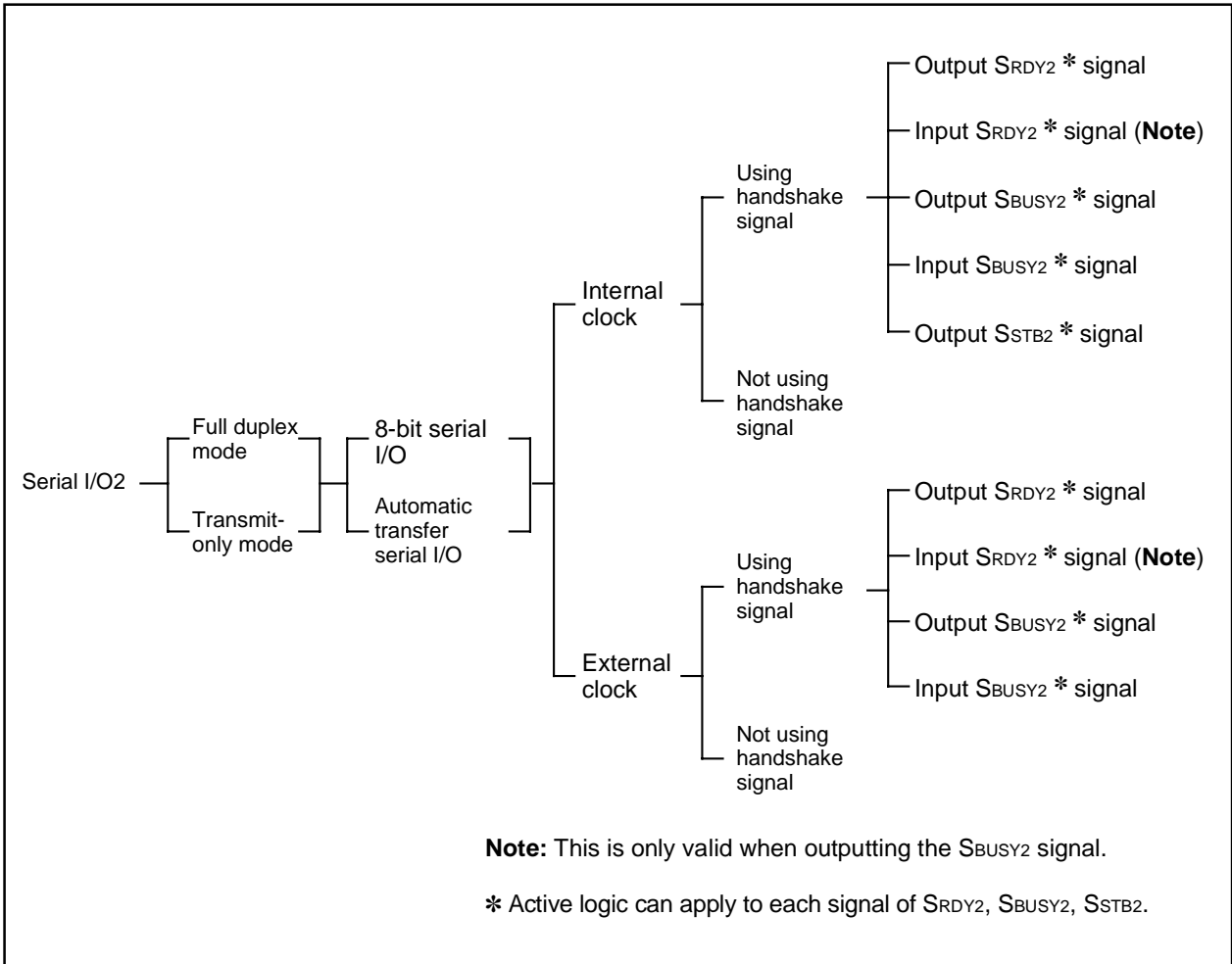


Figure 2.6.6. Serial I/O2 modes

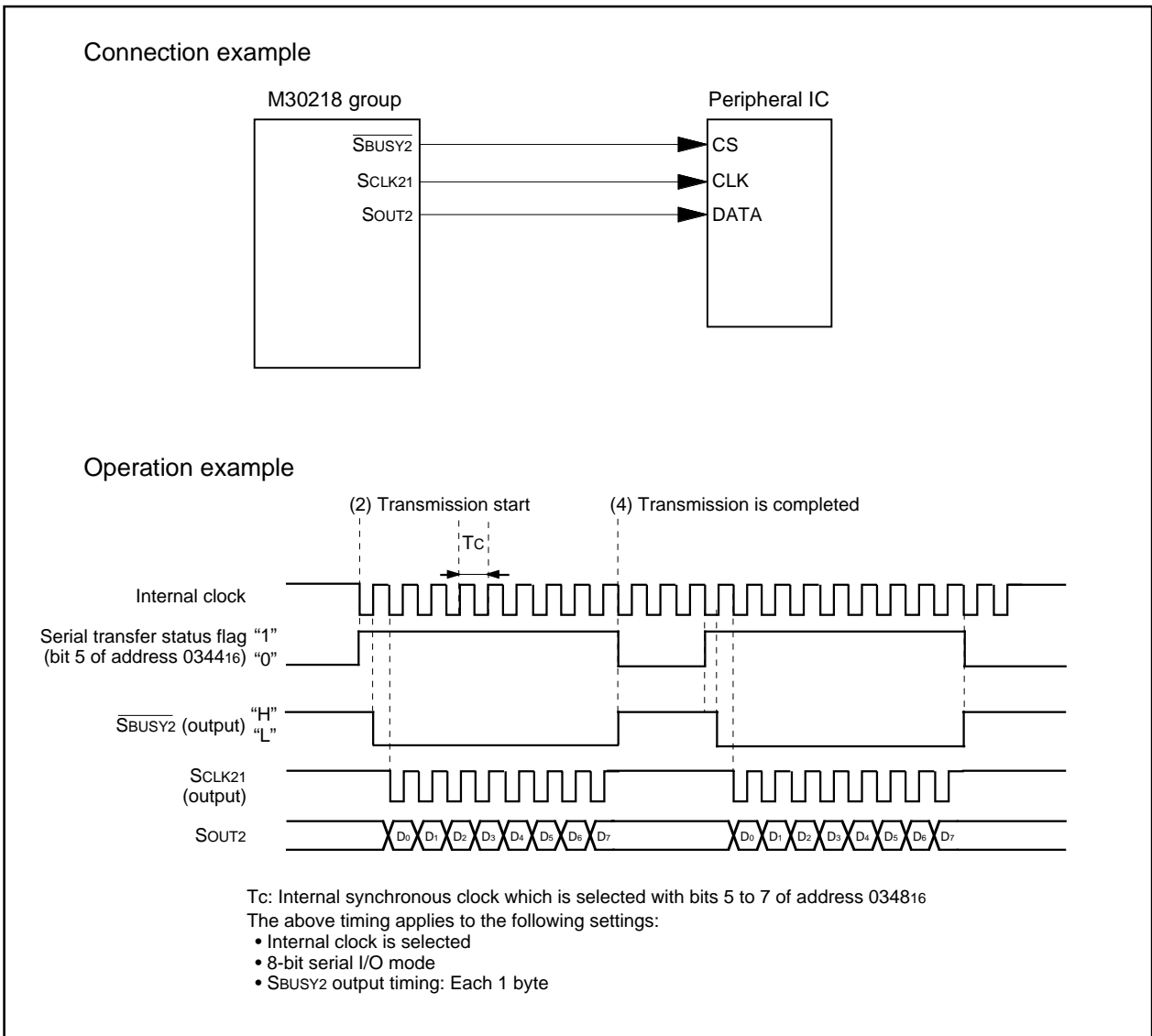
### 2.6.4 Serial I/O2 Operations (transmission in 8-bit serial I/O mode)

The functions listed in Table 2.6.1 can be selected in the 8-bit serial I/O mode for Serial I/O2 transmission/reception. Operations of the circled items are described below. Figure 2.6.7 shows the operation timing, and Figures 2.6.8 and 2.6.9 show the set-up procedure.

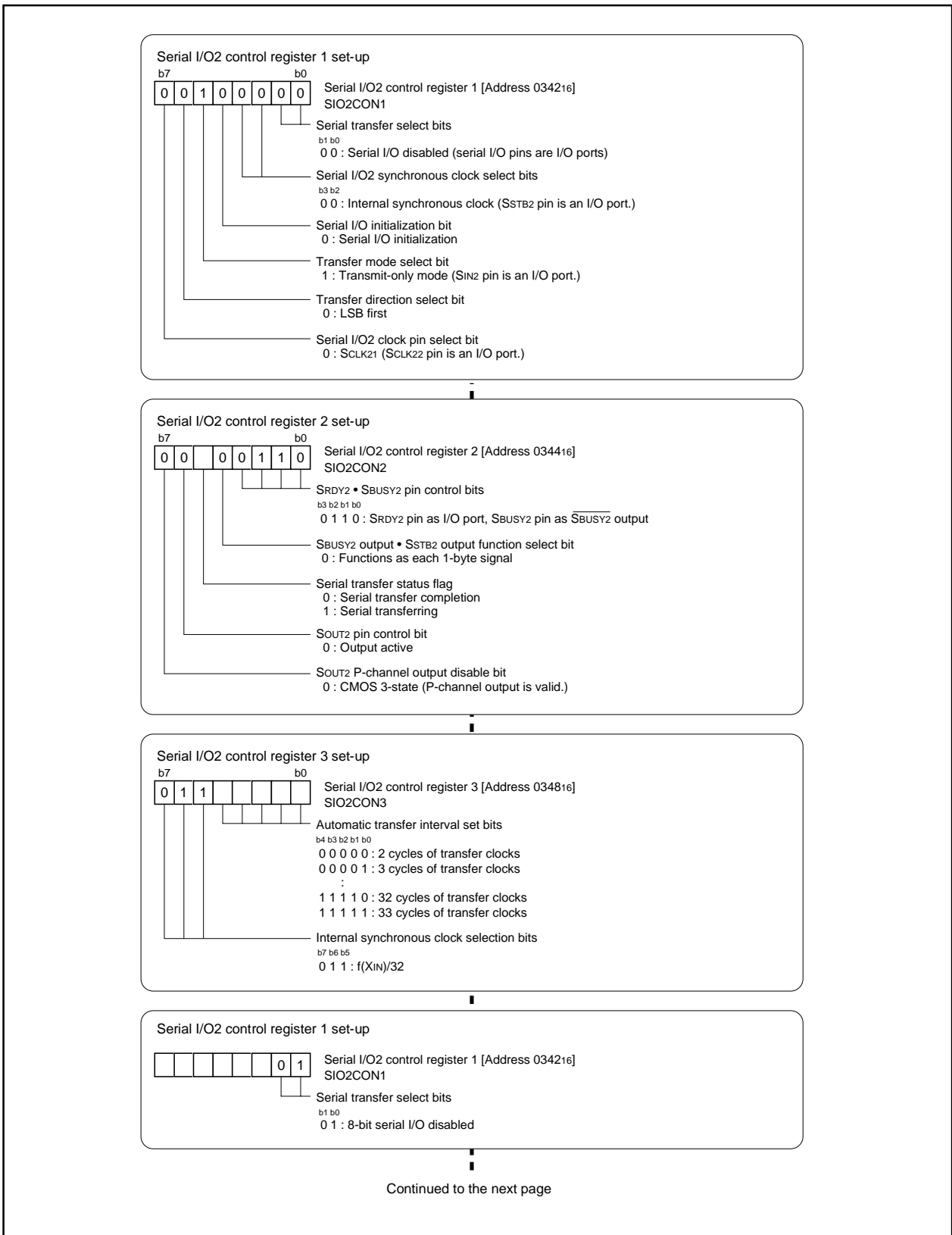
**Table 2.6.1. Selectable functions**

Item	Set-up		Item	Set-up		
Transfer clock source	<input type="radio"/>	Internal clock (f <sub>1</sub> / f <sub>8</sub> / f <sub>32</sub> )	S <sub>BDY2</sub> function	<input type="checkbox"/>	Not selected	
		External clock (CLK <sub>i</sub> pin)		<input type="checkbox"/>	S <sub>BDY2</sub> input	
Automatic transfer serial I/O	<input type="radio"/>	Not selected		<input type="checkbox"/>	S <sub>BDY2</sub> output ("H" at stop required)	
		Selected		<input type="radio"/>	S <sub>BDY2</sub> output ("L" at stop required)	
S <sub>STB2</sub> output function	<input type="radio"/>	Not selected		S <sub>RDY2</sub> function	<input type="radio"/>	Not selected
		S <sub>STB2</sub> ("H" at transmission/reception completed)			<input type="checkbox"/>	S <sub>RDY2</sub> input
	S <sub>STB2</sub> ("L" at transmission/reception completed)	<input type="checkbox"/>	S <sub>RDY2</sub> output			
Transfer direction	<input type="radio"/>	LSB first	<input type="checkbox"/>		S <sub>RDY2</sub> output ("H" at ready)	
		MSB first	<input type="checkbox"/>		S <sub>RDY2</sub> output ("L" at ready)	

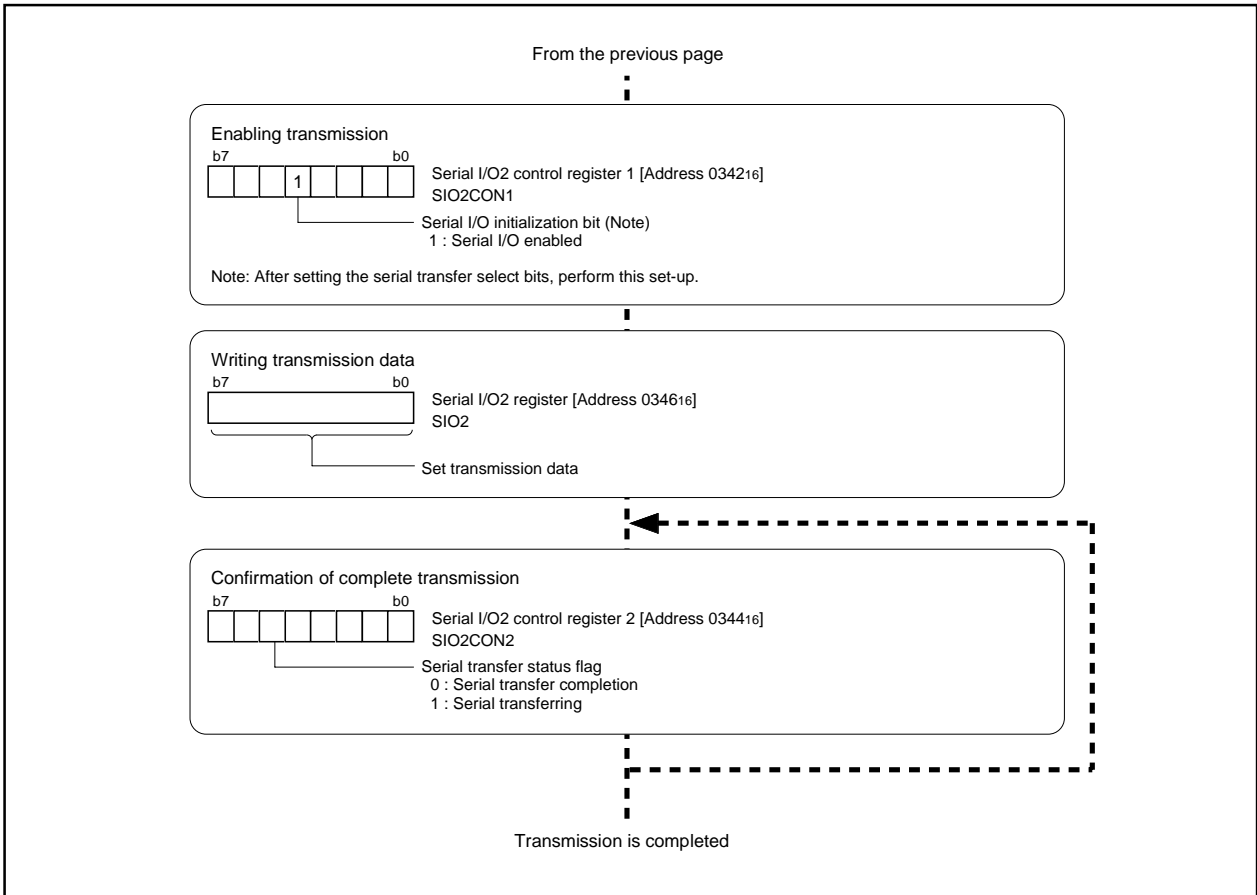
- Operation
- (1) Serial I/O2 becomes transmission-enabled with the following settings: serial transfer select bits SCON10 to "1" and SCON11 to "0"; transfer mode select bit (SCON15) to "1"; serial I/O initialization bit (SCON14) to "1".
  - (2) When transmission data is written to the serial I/O2 register, transmission starts and the serial transfer status flag is set to "1".
  - (3) The transmission data is transmitted bit by bit from the lower bits, synchronized with each falling edge.
  - (4) When one-byte data transmission is completed, the serial transfer status flag is set to "0" to indicate the transmission completion. The transfer clock stops at "H" level.
  - (5) Continuous transmission can be performed by setting the next transmission data in the serial I/O2 register during transmission, before output of the 8th bit.



**Figure 2.6.7. Operation timing of transmission in 8-bit serial I/O mode, using plural transfer clock output function output**



**Figure 2.6.8. Set-up procedure for transmission in 8-bit serial I/O mode, using plural transfer clock output function output (1)**



**Figure 2.6.9. Set-up procedure for transmission in 8-bit serial I/O mode, using plural transfer clock output function output (2)**



### 2.6.5 Serial I/O2 Operations (transmission/reception in automatic transfer serial I/O mode)

The functions listed in Table 2.6.2 can be selected in the automatic transfer serial I/O mode for Serial I/O2 transmission/reception. Operations of the circled items are described below. Figure 2.6.10 shows the operation timing, and Figures 2.6.11 and 2.6.12 show the set-up procedure.

**Table 2.6.2. Selectable functions**

Item	Set-up		Item	Set-up	
Transfer clock source	<input type="radio"/>	Internal clock (f1 / f8 / f32)	SBSY2 function	<input type="radio"/>	Not selected
		External clock (CLKi pin)			SBSY2 input
Automatic transfer serial I/O		Not selected			SBSY2 output ("H" at stop required)
	<input type="radio"/>	Selected			SBSY2 output ("L" at stop required)
SSTB2 output function	<input type="radio"/>	Not selected	SRDY2 function	<input type="radio"/>	Not selected
		SSTB2 ("H" at transmission/reception completed)			SRDY2 input
		SSTB2 ("L" at transmission/reception completed)			SRDY2 output
Transfer direction	<input type="radio"/>	LSB first			SRDY2 output ("H" at ready)
		MSB first		SRDY2 output ("L" at ready)	

- Operation
- (1) After setting the relevant registers, by writing the transfer byte number to the serial I/O2 transfer counter, the serial transfer status flag is set to "1" and automatic transfer starts.
  - (2) The transmission data is transmitted bit by bit from the lower bits, synchronized with each falling edge. The reception data is received bit by bit from the upper bits, synchronized with each rising edge.
  - (3) When eight-byte data transmission/reception is completed, the serial transfer status flag is set to "0" to indicate the transmission/reception completion. The transfer clock stops at "H" level.

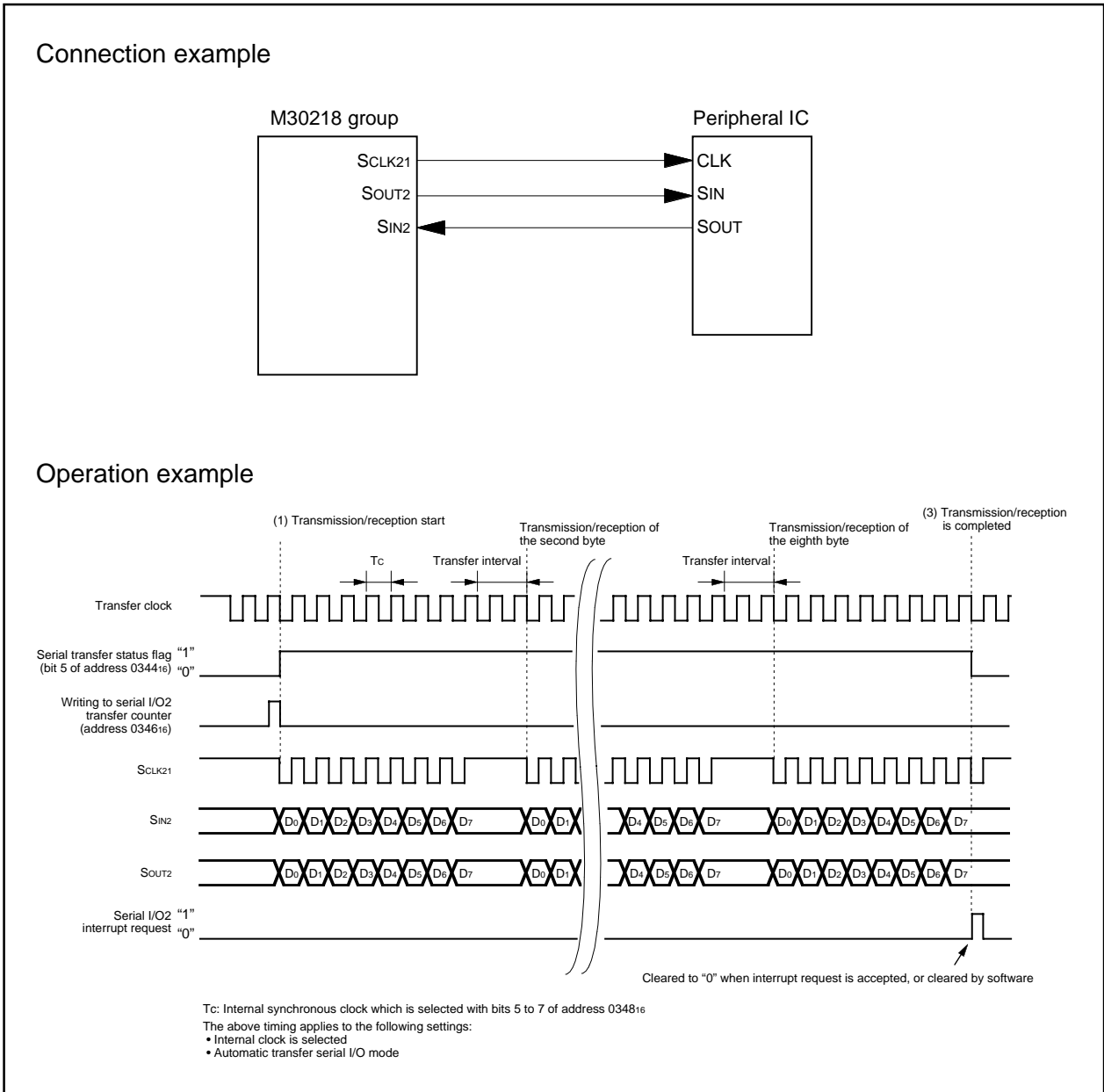


Figure 2.6.10. Operation timing of transmission/reception in automatic transfer serial I/O mode

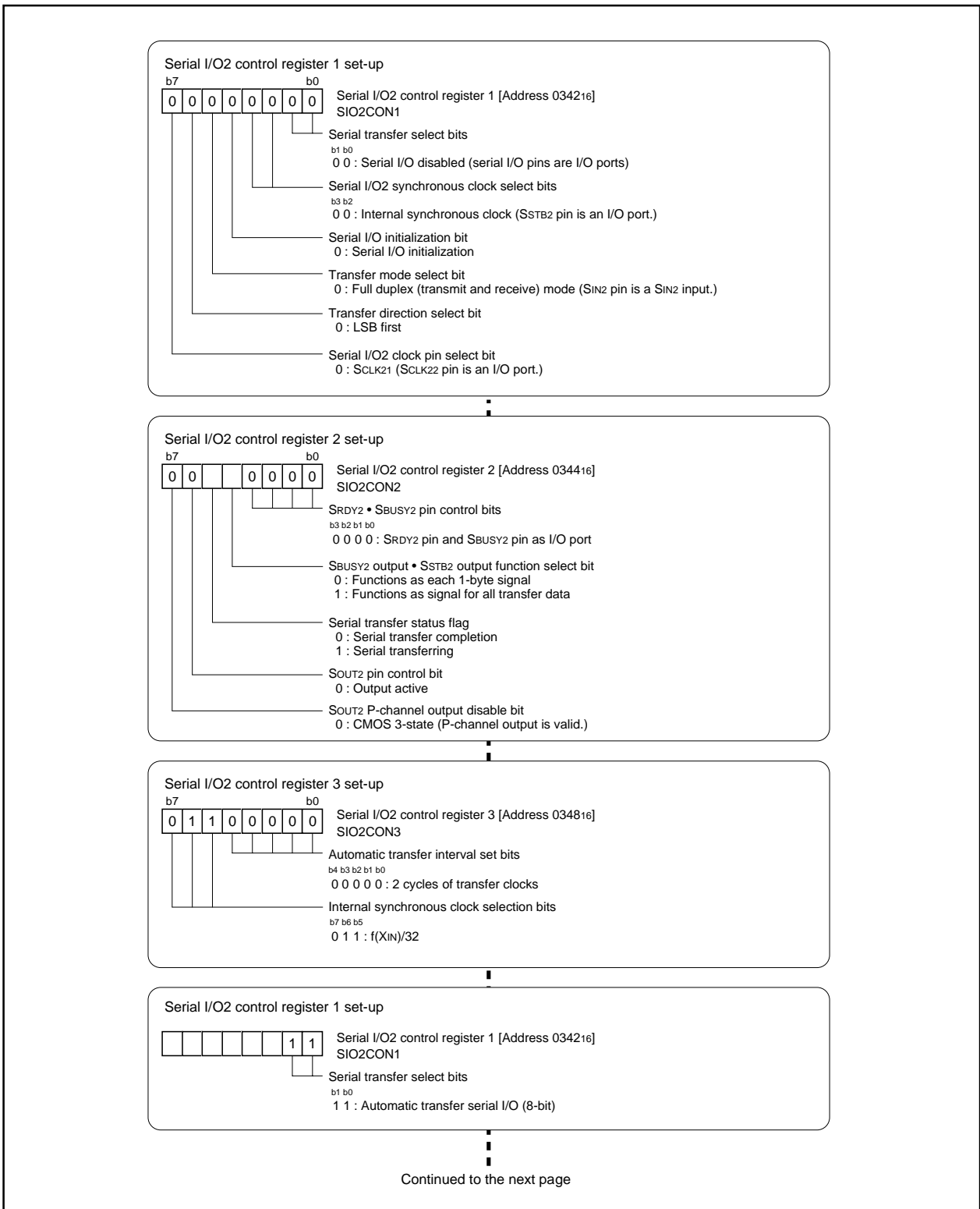


Figure 2.6.11. Set-up procedure for transmission/reception in automatic transfer serial I/O mode (1)

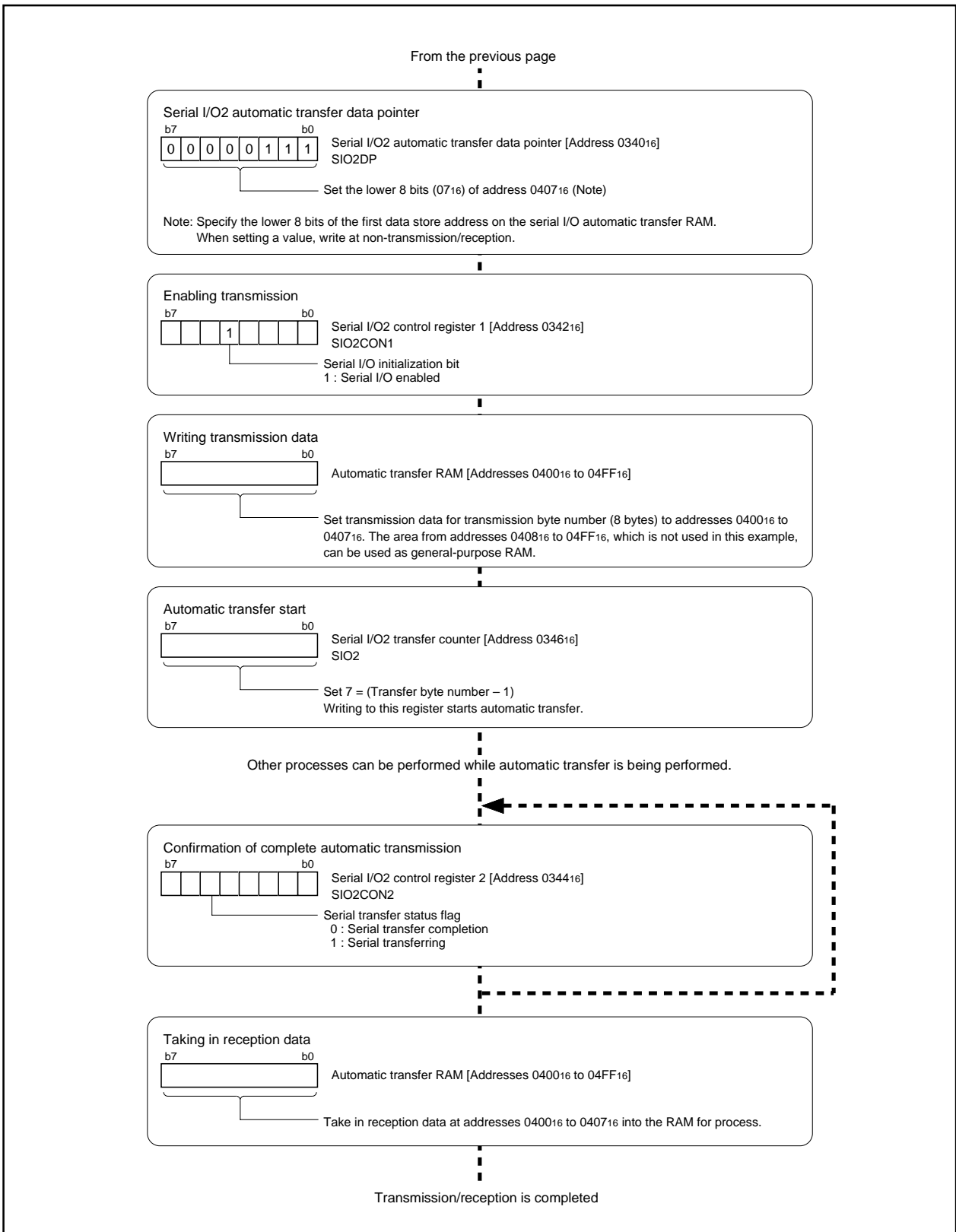


Figure 2.6.12. Set-up procedure for transmission/reception in automatic transfer serial I/O mode (2)

### 2.6.6 Serial I/O2 Operations (transmission/reception in automatic transfer serial I/O mode, using handshake signal)

The functions listed in Table 2.6.3 can be selected in the automatic transfer serial I/O mode for Serial I/O2 transmission/reception. Operations of the circled items are described below. Figure 2.6.13 shows the operation timing, and Figures 2.6.14 and 2.6.15 show the set-up procedure.

**Table 2.6.3. Selectable functions**

Item	Set-up	Item	Set-up
Transfer clock source	<input type="radio"/> Internal clock (f <sub>1</sub> / f <sub>8</sub> / f <sub>32</sub> )	S <sub>BUSY2</sub> function	Not selected
	<input type="radio"/> External clock (CLK <sub>i</sub> pin)		<input type="radio"/> S <sub>BUSY2</sub> input
Automatic transfer serial I/O	Not selected		S <sub>BUSY2</sub> output ("H" at stop required)
	<input type="radio"/> Selected		S <sub>BUSY2</sub> output ("L" at stop required)
S <sub>STB2</sub> output function	<input type="radio"/> Not selected	S <sub>RDY2</sub> function	Not selected
	<input type="radio"/> S <sub>STB2</sub> ("H" at transmission/reception completed)		S <sub>RDY2</sub> input
	<input type="radio"/> S <sub>STB2</sub> ("L" at transmission/reception completed)		<input type="radio"/> S <sub>RDY2</sub> output
Transfer direction	<input type="radio"/> LSB first		S <sub>RDY2</sub> output ("H" at ready)
	<input type="radio"/> MSB first	S <sub>RDY2</sub> output ("L" at ready)	

- Operation
- (1) After setting the relevant registers, by writing the transfer byte number to the serial I/O2 transfer counter, the serial transfer status flag is set to "1" and automatic transfer starts. S<sub>RDY2</sub> output simultaneously goes to "H" level.
  - (2) When "L" level is input to the S<sub>BUSY2</sub> pin, the S<sub>RDY2</sub> output goes to "L" level, synchronized with the falling edge of the transfer clock, and the serial transfer starts.
  - (3) The transmission data is transmitted bit by bit from the lower bits, synchronized with each falling edge. The reception data is received bit by bit from the upper bits, synchronized with each rising edge.
  - (4) When sixteen-byte data transmission/reception is completed, the serial transfer status flag is set to "0" to indicate the transmission/reception completion. The transfer clock stops at "H" level.

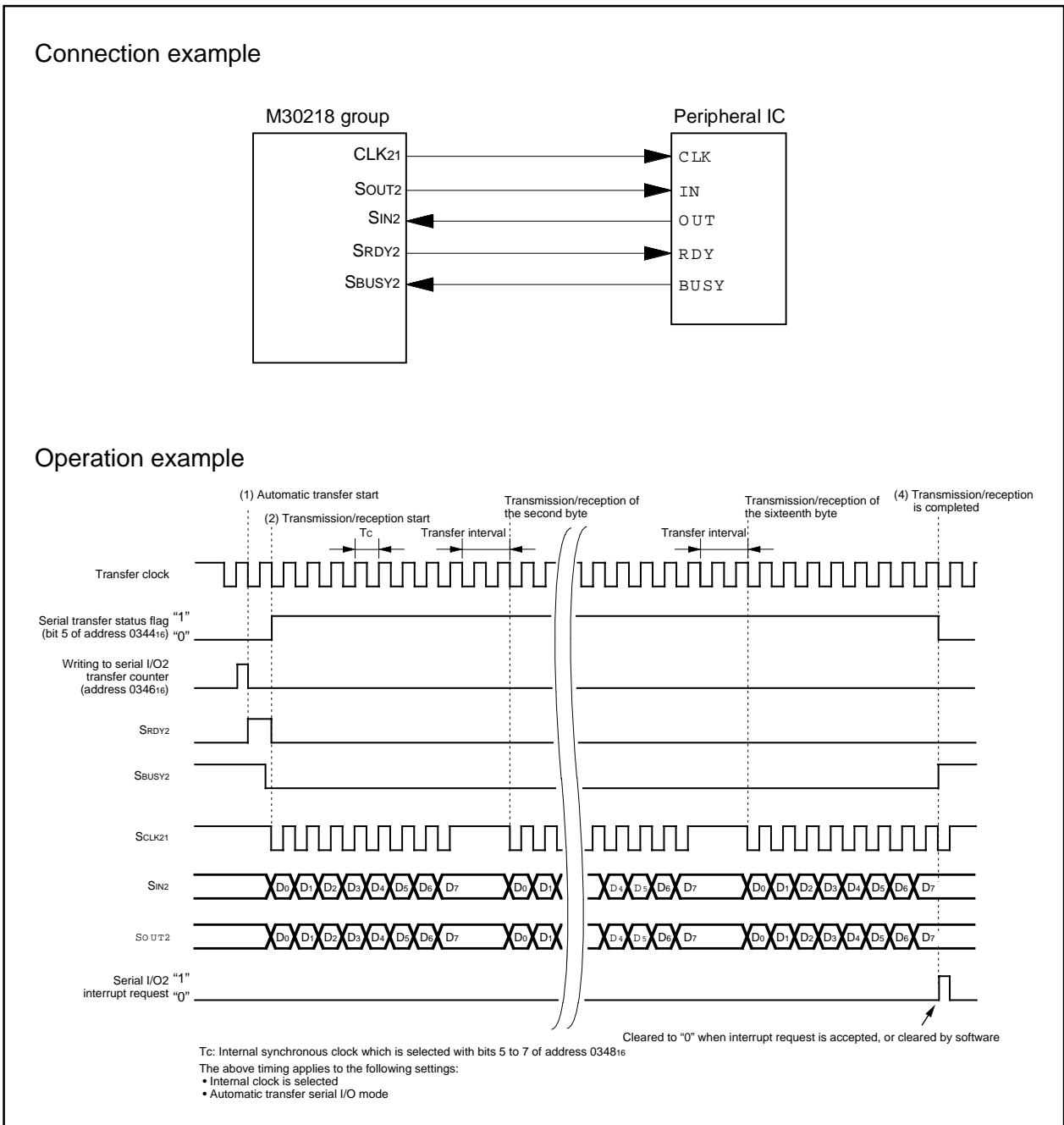


Figure 2.6.13. Operation timing of transmission/reception in automatic transfer serial I/O mode

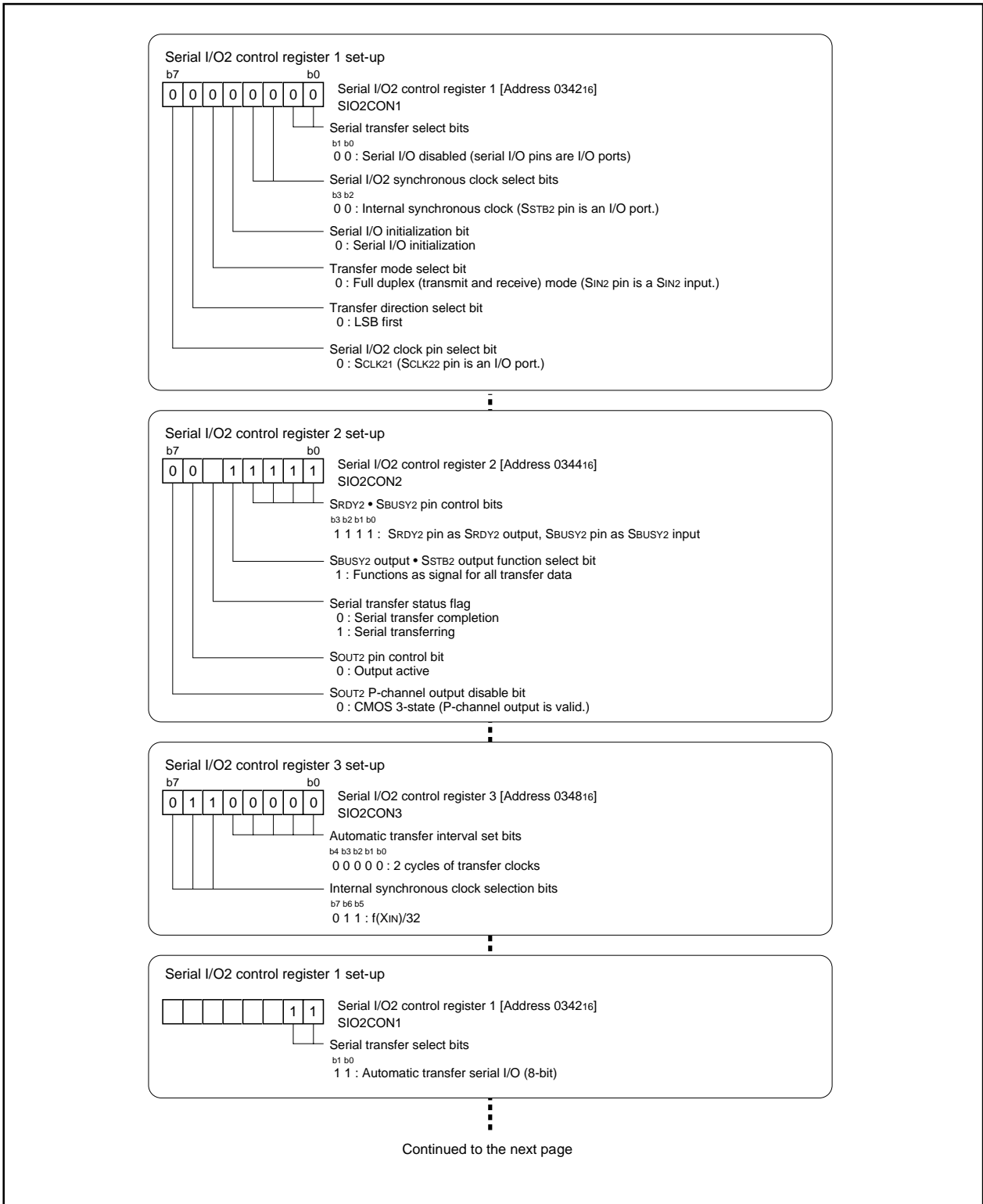


Figure 2.6.14. Set-up procedure for transmission/reception in automatic transfer serial I/O mode (1)

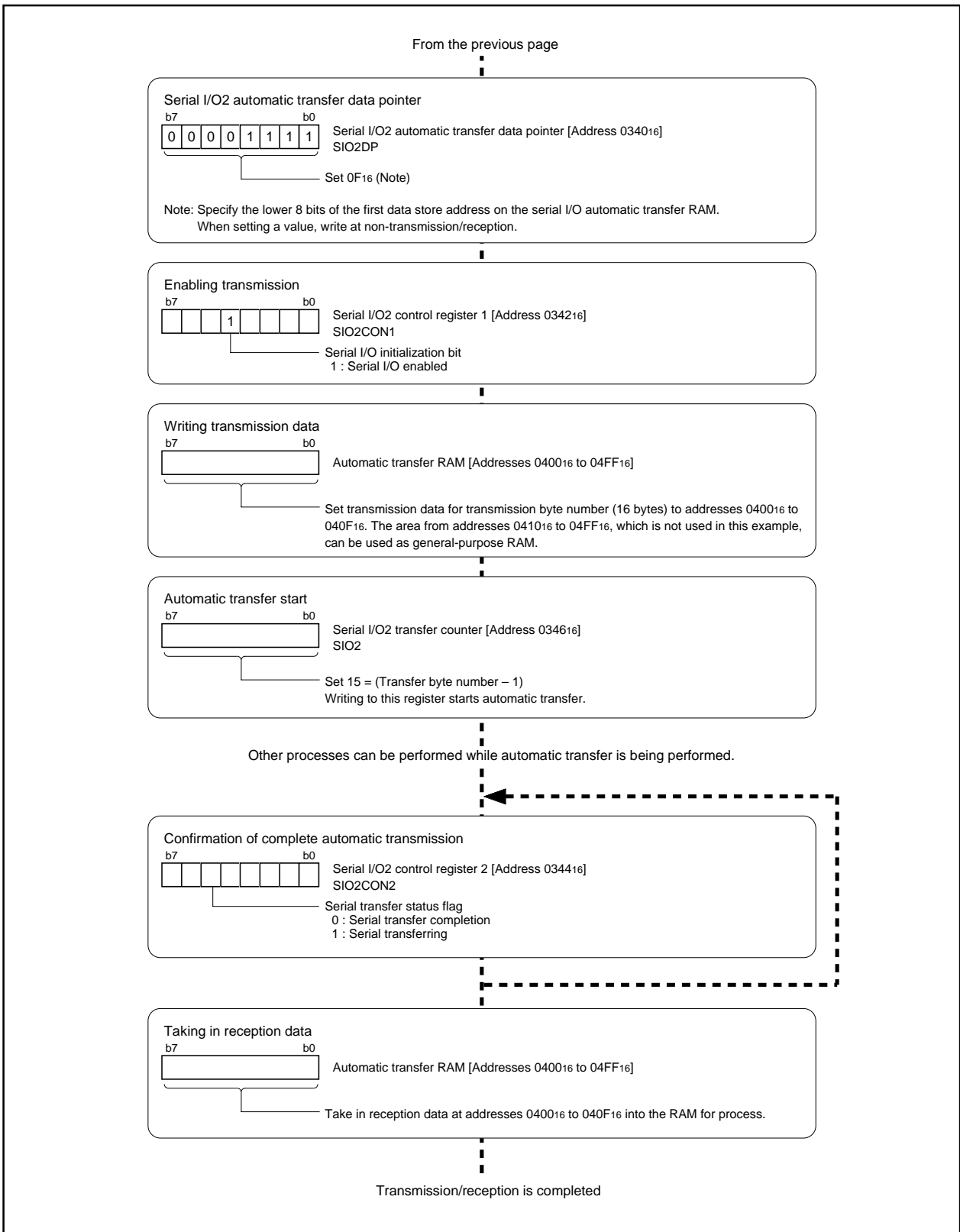


Figure 2.6.15. Set-up procedure for transmission/reception in automatic transfer serial I/O mode (2)



## 2.6.7 Precautions for Serial I/O2

### (1) Clock

#### (a) Using internal clock

After setting the synchronous clock to an internal clock, clear the serial I/O interrupt request bit before performing a normal serial I/O transfer or a serial I/O automatic transfer.

#### (b) Using external clock

After inputting “H” level to the external clock input pin, clear the serial I/O interrupt request bit before performing a normal serial I/O transfer or a serial I/O automatic transfer.

### (2) Using Serial I/O2 interrupt

Clear bit 3 of the interrupt control register to “0” by software before enabling interrupts.

### (3) State of SOUT2 pin

The SOUT2 pin control bit of the serial I/O2 control register 2 can be used to select the SOUT2 pin state for non-transfer periods. Either output active or high-impedance can be selected. However, when using an external synchronous clock, set the SOUT2 pin control bit to “1” while the serial I/O2 clock input is in “H” level (after transfer completion) in order to put the SOUT2 pin in the high-impedance state.

### (4) Serial I/O initialization bit

- To terminate a serial transfer while transferring, set “0” to the serial I/O initialization bit of the serial I/O2 control register 1.
- When “1” is written to the serial I/O initialization bit, Serial I/O2 is enabled, however, each register is not initialized. The value of each register needs to be set by software.

### (5) Handshake signal

#### (a) SBUSY2 input signal

Input “H” level to the SBUSY2 input and “L” level to the  $\overline{\text{SBUSY2}}$  input in the initial state. When using the external synchronous clock, switch the input level to the SBUSY2 input and the  $\overline{\text{SBUSY2}}$  input while the serial I/O2 clock input is in “H” level.

#### (b) SRDY2 input/output signal

When using the internal synchronous clock, input “L” level to the SRDY2 input and “H” level to the  $\overline{\text{SRDY2}}$  input in the initial state.

### (6) In 8-bit serial I/O mode

When the external synchronous clock is used, the contents of the serial I/O2 register are being shifted continually while the transfer clock is input to the serial I/O2 clock pin. At this time, the clock must be controlled externally.

**(7) In automatic transfer serial I/O mode**

**<How to set automatic transfer interval>**

**(a)**

When using

•SBUSY2 output and,

•SBUSY2 output•SSTB2 output function as signals for each transfer data, which is set by SBUSY2 output•SSTB2 output function select bit of the serial I/O2 control register 2,

then the transfer interval is inserted before the first data is transmitted/received and after the last data is transmitted/received.

Accordingly, regardless of the contents of the SBUSY2 output•SSTB2 output function select bit, the transfer interval for each 1-byte data becomes 2 cycles longer than the value set by the automatic transfer interval set bits of the serial I/O2 control register 3.

**(b)**

When using SSTB2 output, regardless of the contents of the SBUSY2 output•SSTB2 output function select bit, the transfer interval for each 1-byte data becomes 2 cycles longer than the value set by the automatic transfer interval set bits of the serial I/O2 control register 3.

**(c)**

When using the combined output of SBUSY2 and SSTB2 as the signal for each transfer data set, the transfer interval after completion of transmission/reception of the last data becomes 2 cycles longer than the value set by the automatic transfer interval set bits.

(d)

Set the automatic transfer interval for each 1-byte data transfer as explained below to avoid incorrect transmit/receive of the serial data.

•Not using FLD controller

Keep the interval open for 5 cycles or more of the internal system clock from the rising edge of the last bit of 1-byte data.

•Using FLD controller

a. Gradation display OFF

Keep the interval open for 17 cycles or more of the internal system clock from the rising edge of the last bit of 1-byte data.

b. Gradation Display ON

Keep the interval open for 27 cycles or more of the internal system clock from the rising edge of the last bit of 1-byte data.

Tables 2.6.4 and 2.6.5 show the serial I/O2 control register 3 (address 0348<sub>16</sub>) setting example.

(e)

When using an external clock, the automatic transfer interval setting becomes invalid.

**Table 2.6.4 Serial I/O2 control register 3, SIO2CON3 (address 0348<sub>16</sub>) setting example (with internal synchronous clock)**

Serial I/O2 control register 3, SIO2CON3 (address 0348 <sub>16</sub> )		Not using FLDC	Gradation display mode OFF	Gradation display mode ON
Internal synchronous clock selection bits	Automatic transfer interval set bits (b4 to b0)			
<sup>b7 b6 b5</sup> 0 0 0 : f(XIN) / 4	0 0 0 0 0 : 2 cycles of transfer clocks	Usable	Prohibited	Prohibited
	0 0 0 0 1 : 3 cycles of transfer clocks	Usable	Prohibited	Prohibited
	0 0 0 1 0 : 4 cycles of transfer clocks	Usable	Prohibited	Prohibited
	0 0 0 1 1 : 5 cycles of transfer clocks	Usable	Usable	Prohibited
	0 0 1 0 0 : 6 cycles of transfer clocks	Usable	Usable	Prohibited
	0 0 1 0 1 : 7 cycles of transfer clocks	Usable	Usable	Usable
0 0 1 : f(XIN) / 8	0 0 0 0 0 : 2 cycles of transfer clocks	Usable	Prohibited	Prohibited
	0 0 0 0 1 : 3 cycles of transfer clocks	Usable	Usable	Prohibited
	0 0 0 1 0 : 4 cycles of transfer clocks	Usable	Usable	Usable
0 1 0 : f(XIN) / 16	0 0 0 0 0 : 2 cycles of transfer clocks	Usable	Usable	Usable

Note: Do not perform the following in the automatic transfer serial I/O mode:

- Transfer within the RAM area (addresses 00400<sub>16</sub> to 005FF<sub>16</sub>) using the DMAC
- Transfer within the RAM area (addresses 00400<sub>16</sub> to 005FF<sub>16</sub>) using assembler instructions SMOVF and SMOVB.

**Table 2.6.5 Serial I/O2 control register 3, SIO2CON3 (address 0348<sub>16</sub>) setting example (with external synchronous clock)**

Serial I/O2 control register 3, SIO2CON3 (address 0348 <sub>16</sub> ); Automatic transfer interval set bits	"n" cycles of transfer clocks
Not using FLDC	Transfer clock X n cycles ≥ 5 cycles of internal system clock
Gradation display mode OFF	Transfer clock X n cycles ≥ 17 cycles of internal system clock
Gradation display mode ON	Transfer clock X n cycles ≥ 27 cycles of internal system clock

### <How to set serial I/O2 transfer counter>

#### (a)

Write the value of the number of transfer-data decreased by 1 to the serial I/O2 transfer counter.

#### (b)

When using an external clock, after writing a value to the serial I/O2 register/transfer counter, wait for 5 or more cycles of the internal system clock before inputting the transfer clock to the serial I/O2 clock pin.

### <Serial I/O initialization bit>

The serial I/O automatic transfer interrupt request occurs when "0" is written to the serial I/O initialization bit during an operation. Use software to set this interrupt priority level to level 0 (interrupt disabled), or any other methods which will disable it.

### <Interrupt request bit>

The occurrence timing of serial I/O automatic transfer interrupt request may be delayed:

- Normally, the maximum delay is 17 cycles. In the FLD gradation display mode ON, the maximum delay increases to 27 cycles.
- If the occurrence timing of the serial I/O2 interrupt request is delayed, the flags and the signals which change simultaneously with the timing of the interrupt request, such as the serial transfer status flag and the handshake signals, will also change in accordance to the delay.

## 2.7 FLD (VFD) Controller

### 2.7.1 Overview

The FLD controller drives and controls FLDs (fluorescent display). The following is the FLD controller overview.

#### (1) FLDC port

There are a total of 56 ports, consisting of 52 high-breakdown-voltage (HBV) ports and 4 CMOS ports. 20 of the 52 HBV ports can be switched to normal ports, and all of the 4 CMOS ports can be switched to general purpose ports. However, when using CMOS ports as display pins, external drivers must be installed.

Ports P0, P1, P5 and P6, totaling 32 ports, have built-in pull-down resistors.

#### (2) Display pixel number

##### (a) Using all ports for FLD output

28 segments × 28 digits (segment number + digit number ≤ 56)

##### (b) Using digit pulse output function

40 segments × 16 digits (segment number + digit number ≤ 56, however, digit number ≤ 16)

##### (c) Using P44 to P47 expansion function

52 segments × 16 digits (segment number ≤ 52, digit number ≤ 16)

#### (3) Selection function

The following selection functions can be applied to the FLD controller.

##### (a) Tscan control

Two types of interrupt sources can be selected, using the Tscan control bits (bits 2, 3 of address 0350<sub>16</sub>):

###### •FLD digit interrupt

This is generated when the Toff1 time for each timing ends (at rising edge of digit output). Key scanning, which makes use of FLD digits, can be applied by using each FLD digit interrupt.

###### •FLD blanking interrupt

This is generated when the FLD data pointer (address 0358<sub>16</sub>) reaches FF<sub>16</sub>.

The FLD automatic display output is turned off for a duration of 1 × Tdisp, 2 × Tdisp, or 3 × Tdisp, depending on post-interrupt settings. Key scanning, which makes use of FLD segments, can be applied during this time.

##### (b) Timing number

The following two types of timing can be selected:

###### •16-timing

This timing is used when the display timing is 16 sets or less.

###### •32-timing

This timing is used when the display timing is more than 16 sets. This can be used for up to 32 sets.

**(c) Gradation display mode**

The gradation display mode can apply bright/dark display for each segment when the display timing is 16 or less.

Selection of gradation mode is as follows:

- Gradation display mode ON

Make sure to fix the timing number control bit (bit 4 of address 0350<sub>16</sub>) to "0" as the maximum timing is 16. Additionally, set the value to the Toff2 time set register (address 0356<sub>16</sub>) so that Toff2 time can be less than Tdisp time and more than Toff1 time.

- Gradation display mode OFF

**(d) HBV port drivability**

Two types of drivability, strong or weak, can be selected for HBV ports. This setting is also valid when using HBV ports as general purpose ports.

**(e) P44 to P47 FLD output reverse**

Selecting this function enables the polarity reversal of the FLD output from P44 to P47. This function is useful for adjusting the polarity when using an externally installed driver.

**(f) P44 to P47 Toff invalid**

Selecting this function disables Toff1 time and Toff2 time and outputs display data for the duration of Tdisp.

**(g) P97 dimmer signal output**

Selecting this function outputs a signal from DIMOUT (P97) to the decoder which, in turn, sends out the dimmer signal. The decoder controls this signal to enable the dimmer function.

**(h) Toff section generate/not generate**

This function can be applied to all of the HBV ports (P0, P1, P2, P3, P40 to P43, P5, P6) and CMOS ports (P44 to P47). Two types can be selected:

- Generate Toff section

The Toff section is generated.

- No Toff section

This function reduces unwanted noises generated whenever a port switches due to the combined capacity of the FLD ports. When continuous data is output to each FLD port, the Toff1 section of the continuous parts is not generated.

**(i) Toff2 SET/RESET change**

In gradation display mode, this function specifies either output (SET) or "0" (RESET) depending on Toff2 time for FLD output of dark display data (when gradation display control data is "1"). Two types can be selected:

- Toff2SET

RAM data is output to the FLD output ports (SET) at the time set by Toff2 and is returned to "0" (RESET) when the Tdisp time ends.

- Toff2RESET

RAM data is output to the FLD output ports (SET) at the time set by Toff1 and is returned to "0" (RESET) at the time set by Toff2.

**(4) Expansion function**

The FLD controller is equipped with an expansion function.

**(a) Digit pulses output function**

Digit pulses can be output automatically from ports P5 and P6. When the same number of "1s" as the timing number are consecutively written from P60 to the digit output set registers (addresses 035C<sub>16</sub>, 035D<sub>16</sub>), the contents of the FLD automatic display RAM for the ports that have been selected for digit output are disabled. The digit pulses are then automatically output.

If a value exceeding the timing number for any port is set, the output of such port becomes "L" level.

**(b) P44 to P47 expansion function**

These ports have CMOS output structure. This function provides 16 lines of FLD digit outputs to these four ports by connecting the decoder which converts 4-bit data to 16-bit data.

**(5) Registers related to FLD controller**

Figure 2.7.1 shows the memory map of FLDC related-registers. Figures 2.7.2 to 2.7.6 show FLDC related-registers.

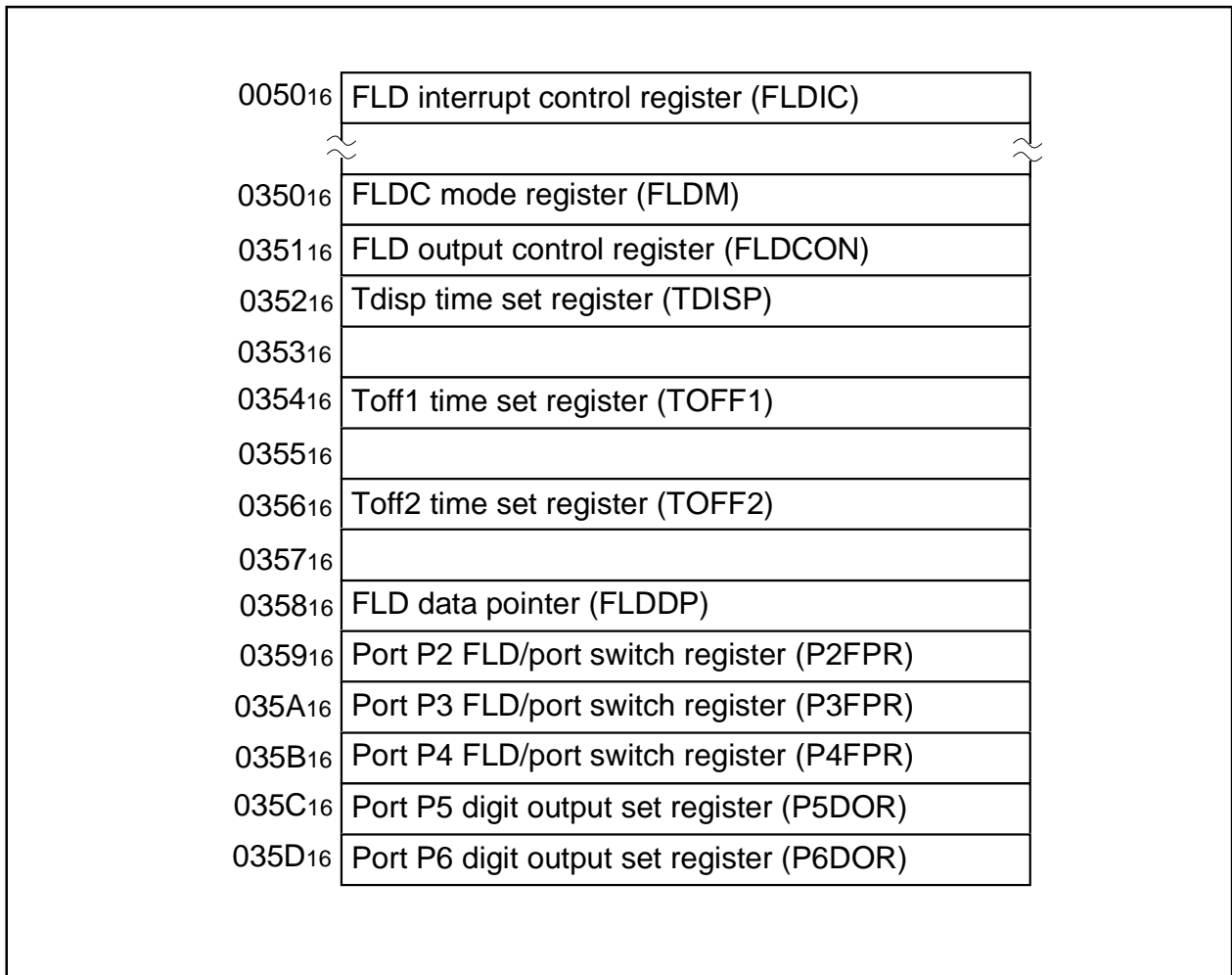


Figure 2.7.1. Memory map of FLDC related-registers



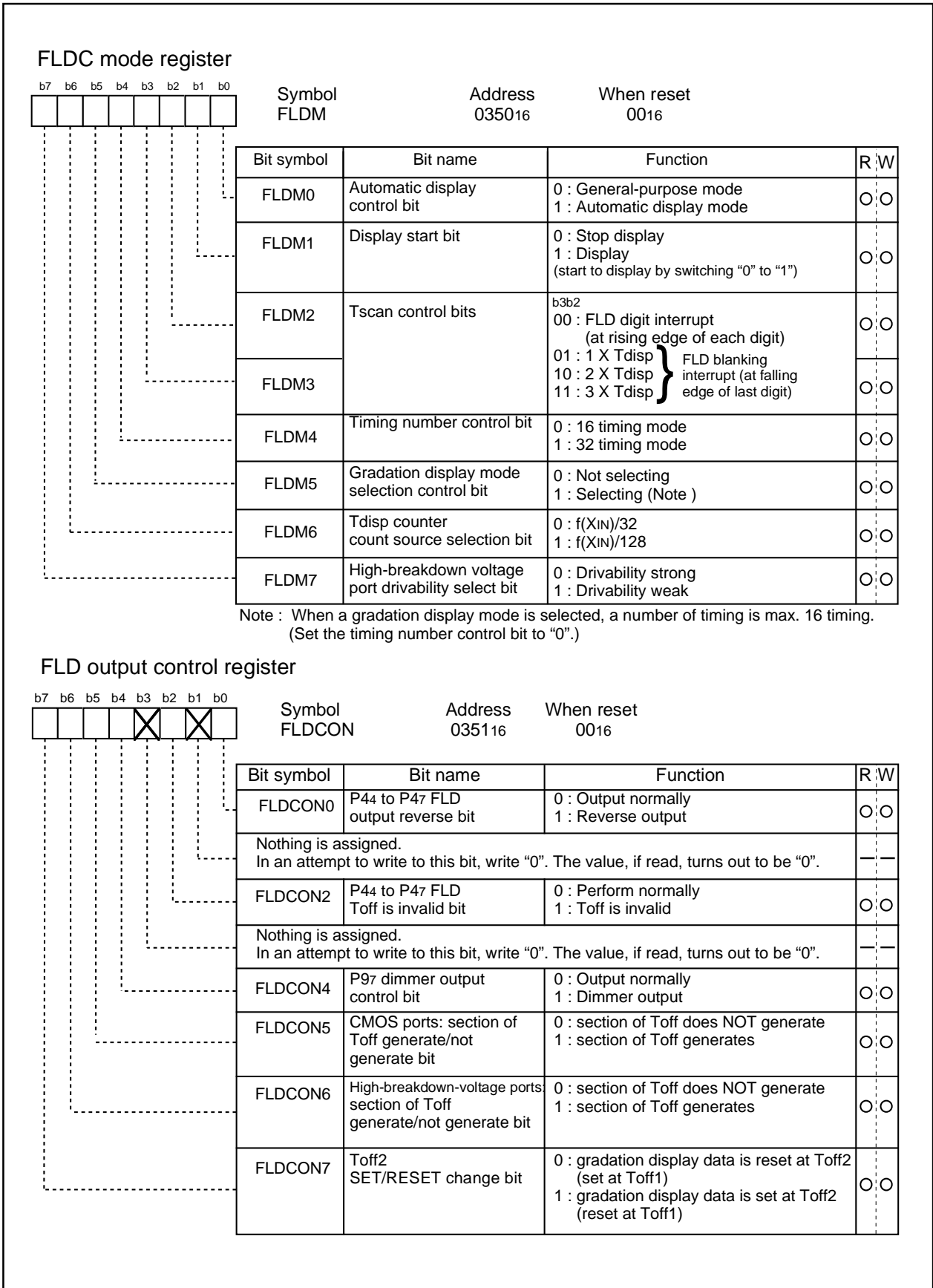


Figure 2.7.2. FLDC related-registers (1)

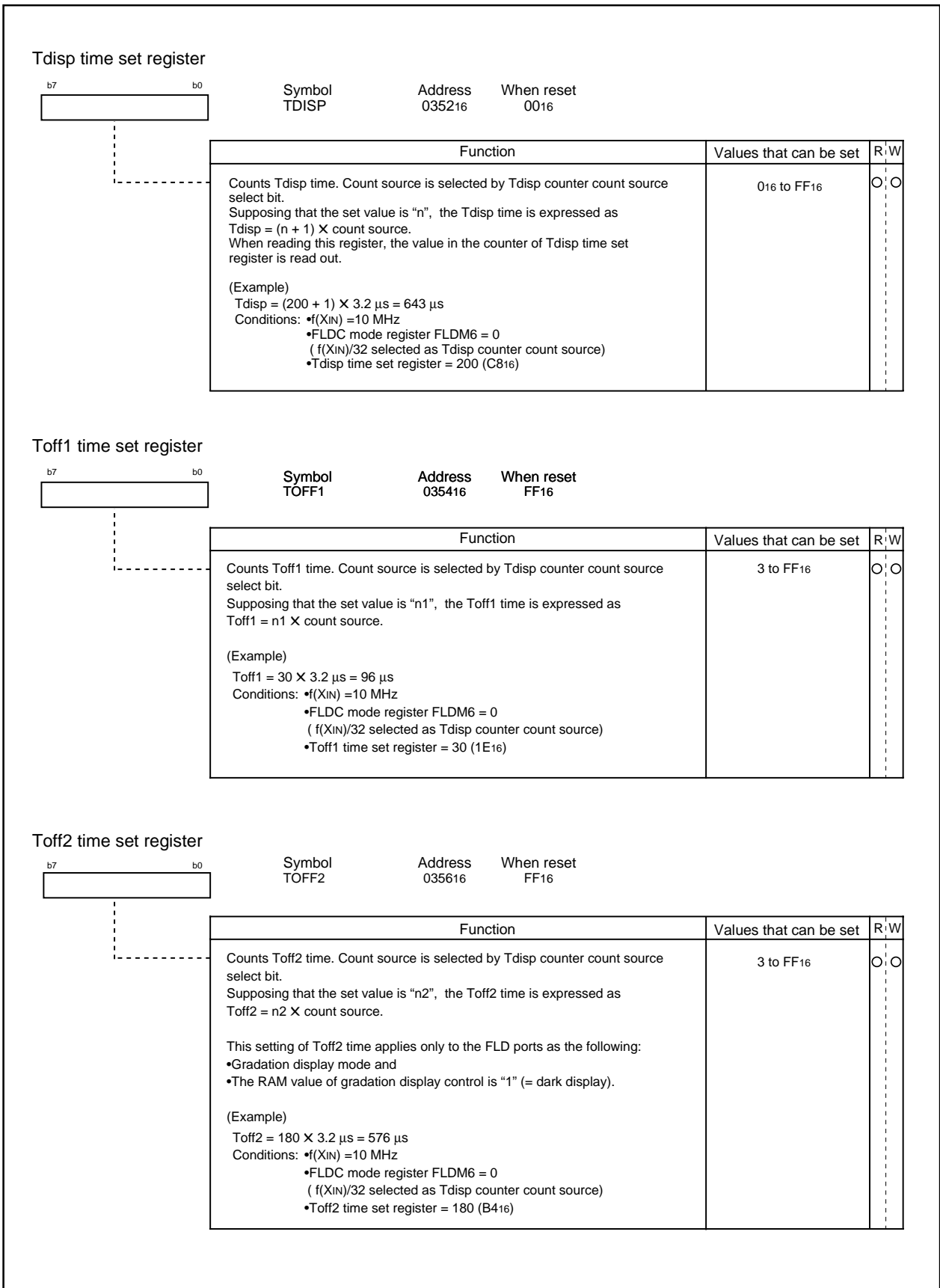


Figure 2.7.3. FLDC related-registers (2)

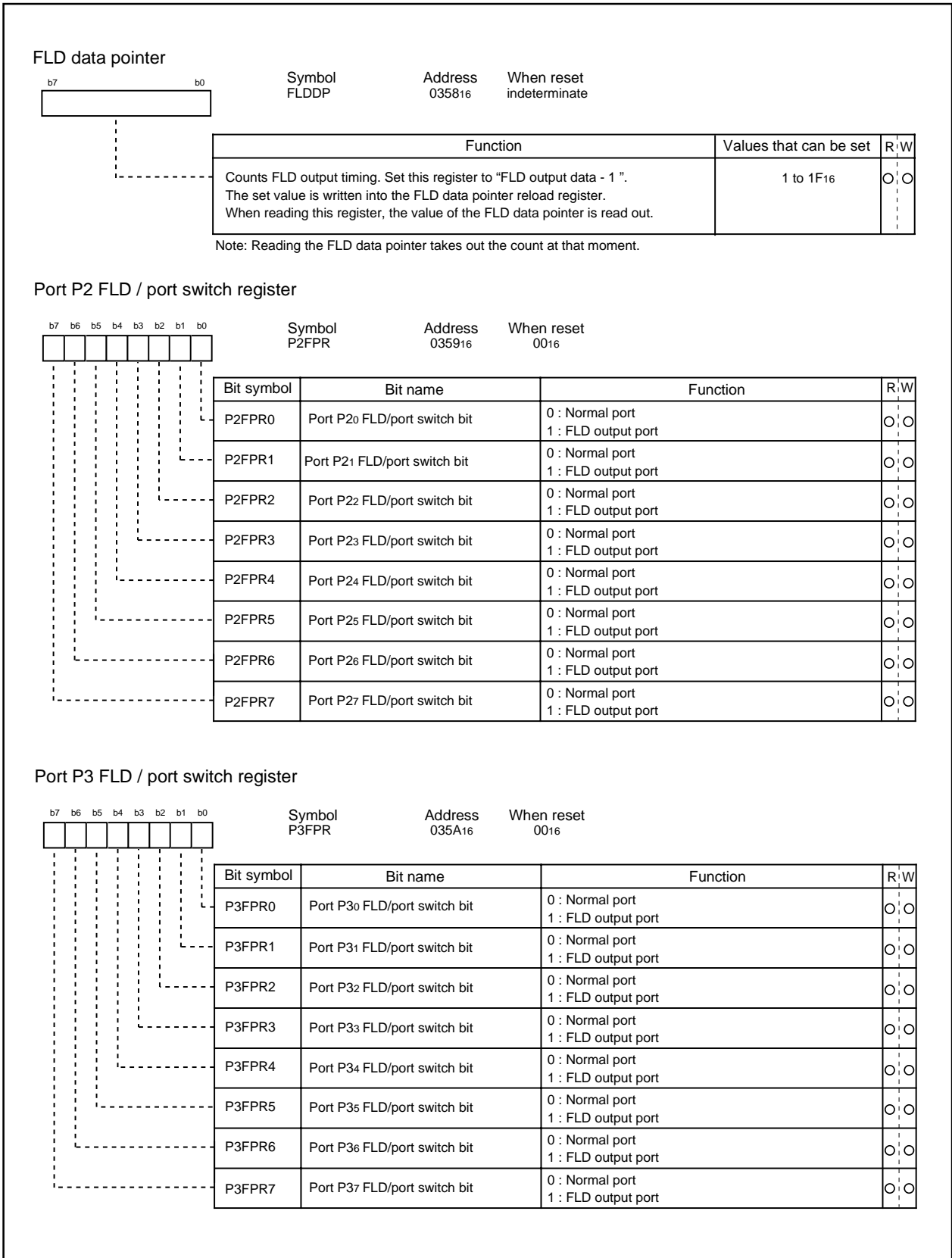


Figure 2.7.4. FLDC related-registers (3)

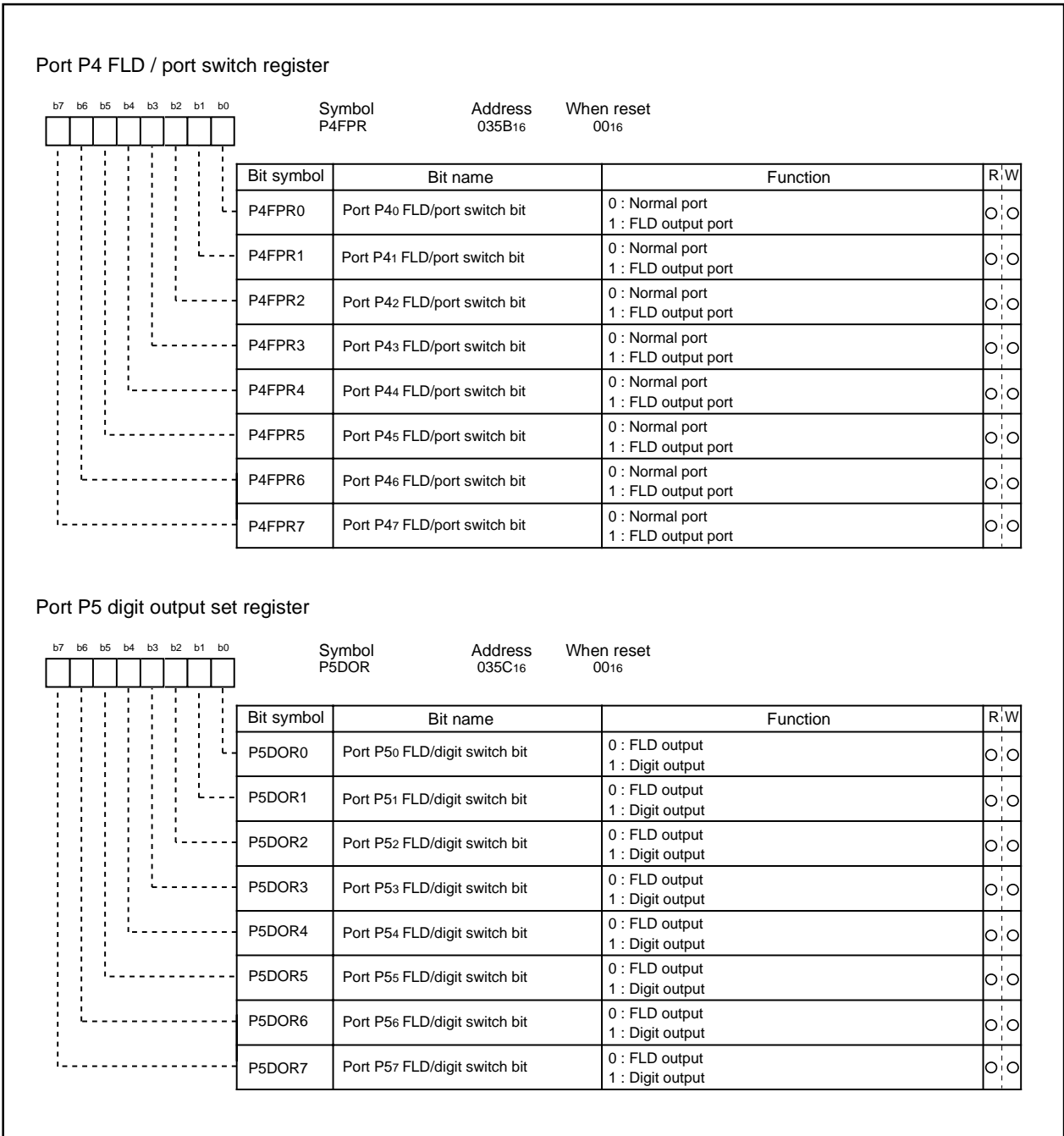


Figure 2.7.5. FLDC related-registers (4)

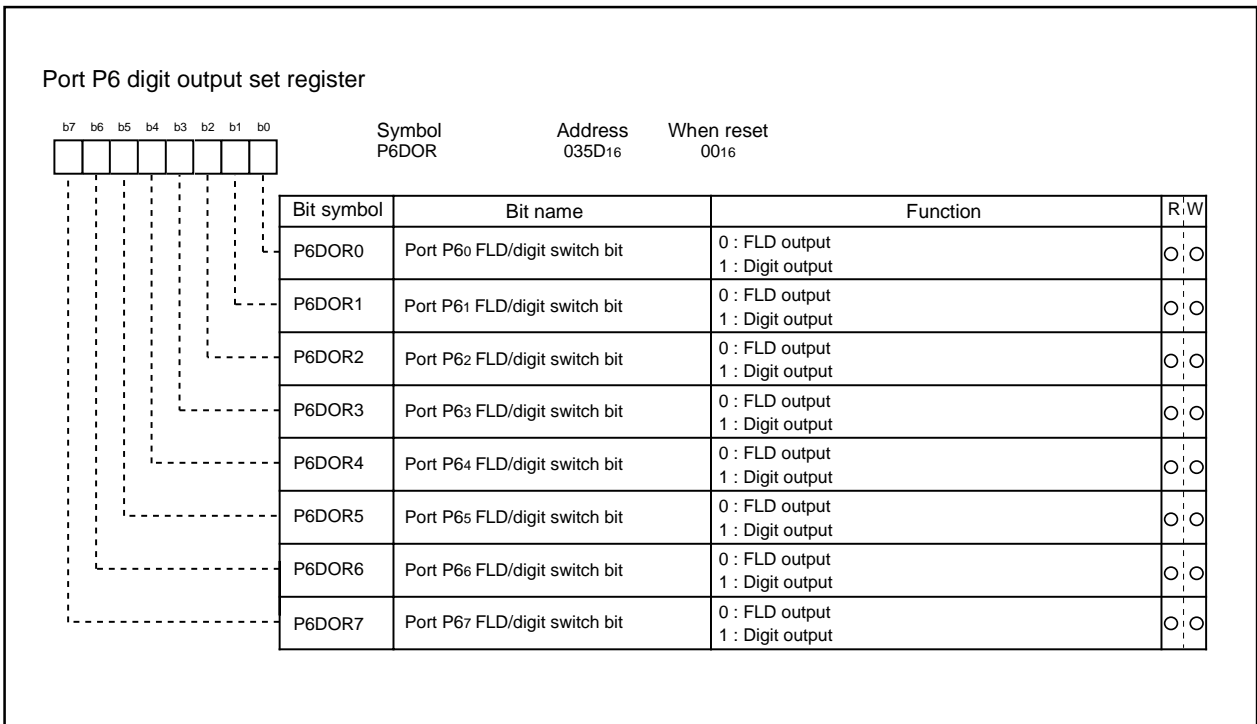


Figure 2.7.6. FLDC related-registers (5)

This page kept blank for layout purposes.

### 2.7.2 FLD operation (FLD automatic display and key-scan using segments)

The FLD controller can choose functions from those listed in Table 2.7.1. The circled items are described in detail below. Figure 2.7.7 shows the operation timing, and Figures 2.7.8 to 2.7.10 show the set-up procedures.

**Table 2.7.1. Selectable functions**

Item	Set-up	Item	Set-up
Tscan control (Note 1)	FLD digit interrupt	High-breakdown voltage port drivability	Strong
	<input type="radio"/> FLD blanking interrupt		<input type="radio"/> Weak
Timing number	<input type="radio"/> 16-timing	P97 dimmer output	<input type="radio"/> Normal port
	32-timing		Dimmer output
Tdisp counter count source	<input type="radio"/> $f(X_{IN})/32$	High-breakdown-voltage ports: Section of Toff generate/not generate	Section of Toff does NOT generate
	$f(X_{IN})/128$		<input type="radio"/> Section of Toff generates
Gradation display mode (Note 2)	Not selecting	Toff2 SET/RESET	<input type="radio"/> Reset at Toff2
	<input type="radio"/> Selecting		Set at Toff2

Note 1: When selecting the FLD blanking interrupt, any one of 1 X Tdisp, 2 X Tdisp, or 3 X Tdisp can be selected as Tscan time.

Note 2: When selecting the gradation display mode, make sure to use 16-timing as the timing number.

- Operation
- (1) The FLD starts an automatic display when both the automatic display control bit and the display start bit are set to “1”.
  - (2) The display data, the contents from the first address through the last address, in the FLD automatic display RAM for each port is output to each port. The last address is the result of decreasing the number indicated in the FLD data pointer from the first address. The gradation display control data is arranged at an address which is calculated by subtracting “7016” from the stored address in the FLD automatic display RAM of the corresponding timing and pin. Bright display is performed by setting “0”, and dark display is performed by setting “1”. However, the contents of the FLD automatic display RAM for ports P50, P51, and P60 to P67 are disabled by selection of the digit pulse output function, and the digit pulses are automatically output.
  - (3) The FLD data pointer counts down during Tdisp time. When the count reaches “FF16”, the pointer is reloaded and starts counting over again.
  - (4) The FLD interrupt request bit is set to “1” simultaneously with the falling edge of the last timing. The FLD automatic display output is turned off for a duration of 1 X Tdisp, 2 X Tdisp, or 3 X Tdisp, depending on post-interrupt settings. During this time, key scanning, which makes use of FLD segments, can be applied.
  - (5) During FLD automatic display, the FLD automatic display can be interrupted by writing “0” to the display start bit.

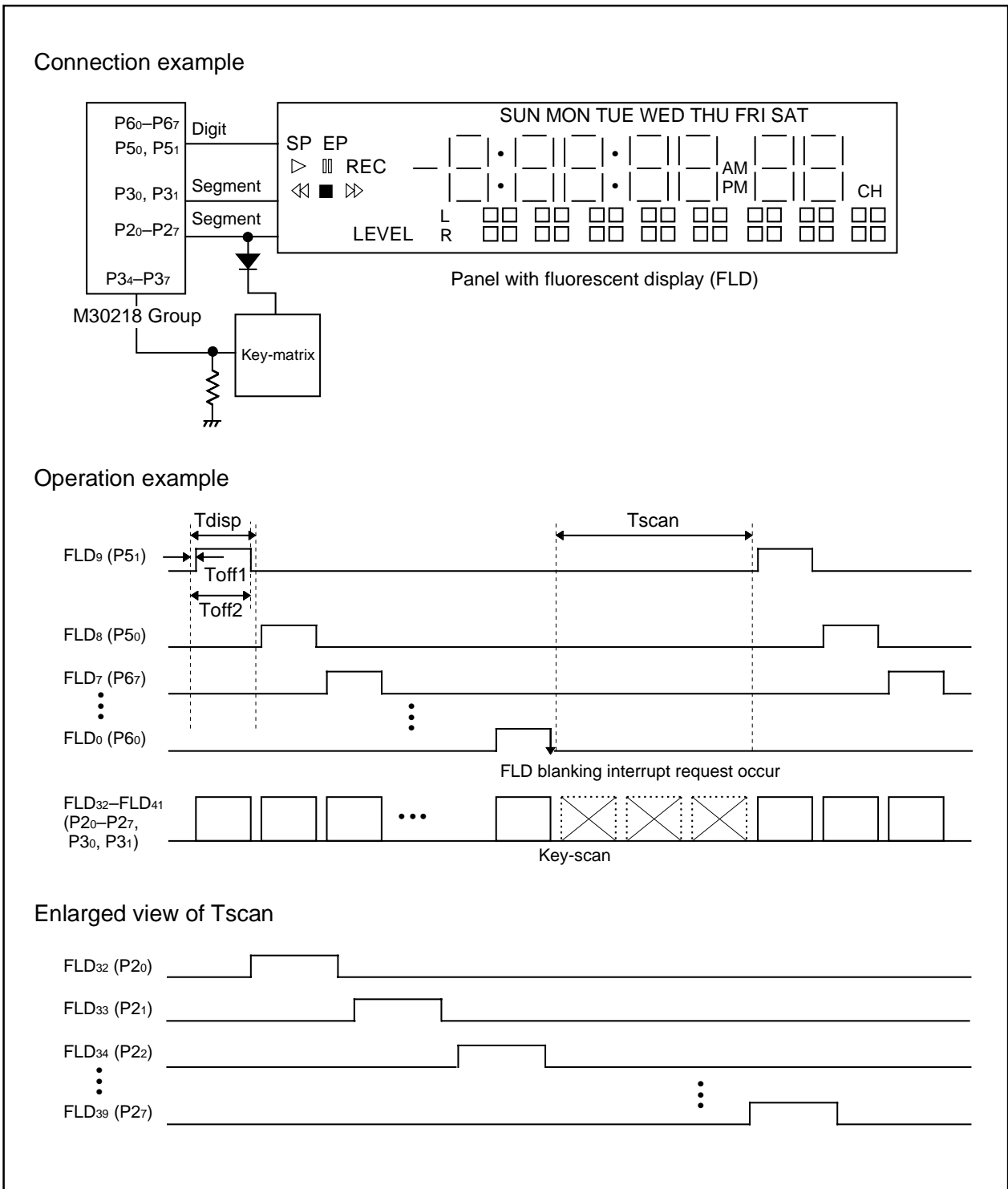


Figure 2.7.7. Operation timing of FLD automatic display



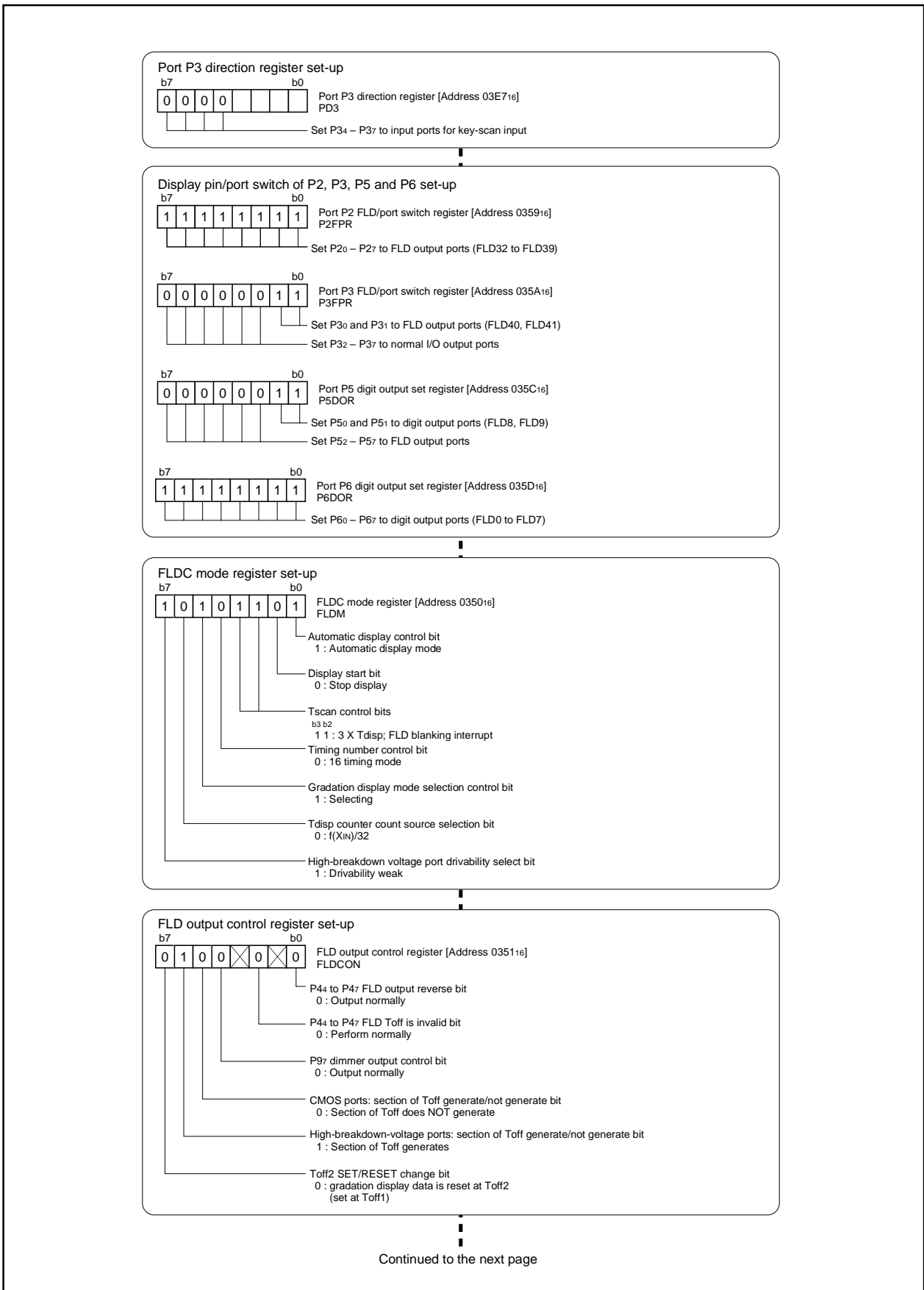


Figure 2.7.8. Set-up procedure for FLD automatic display (1)

Continued from the previous page

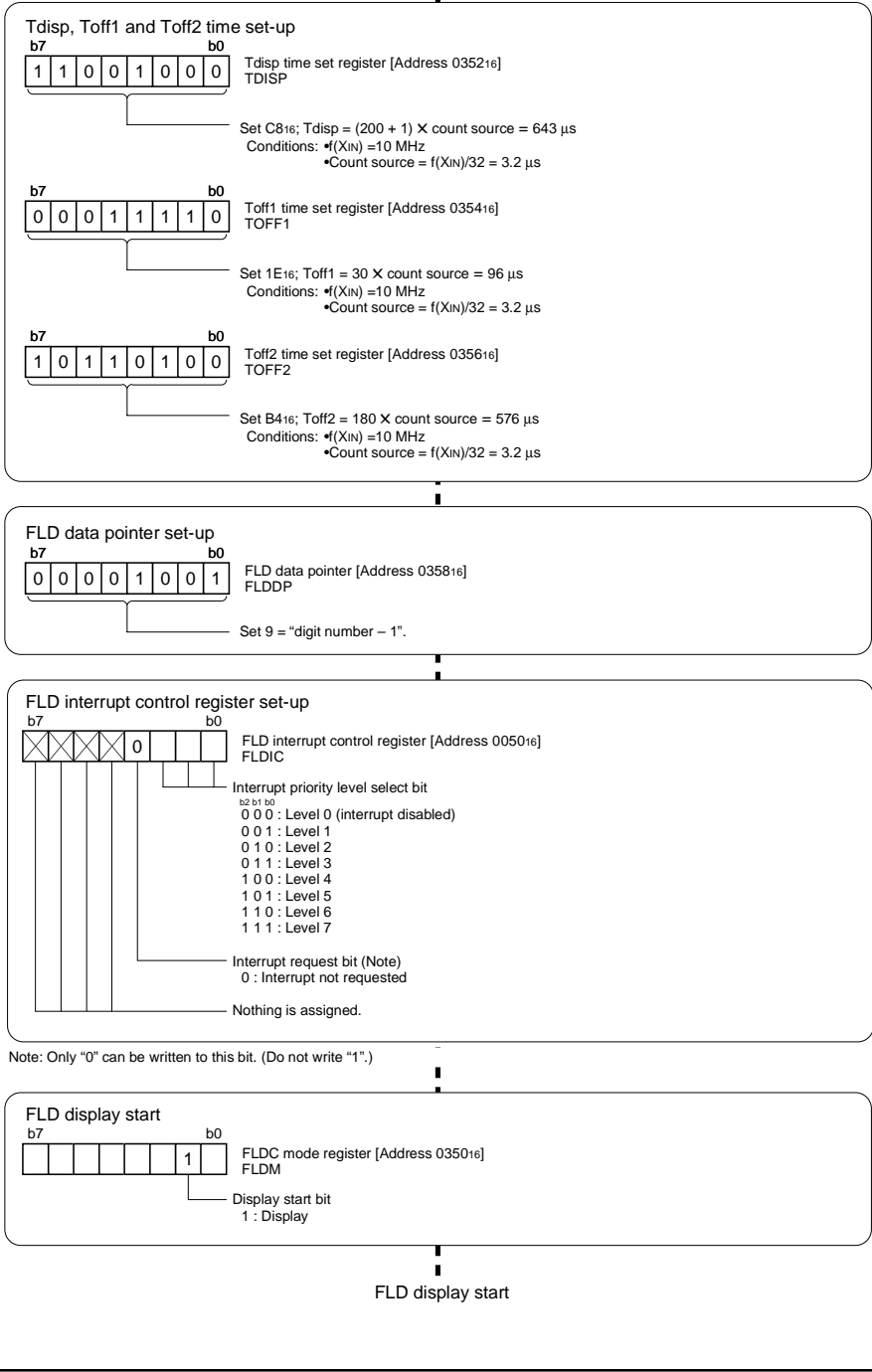


Figure 2.7.9. Set-up procedure for FLD automatic display (2)

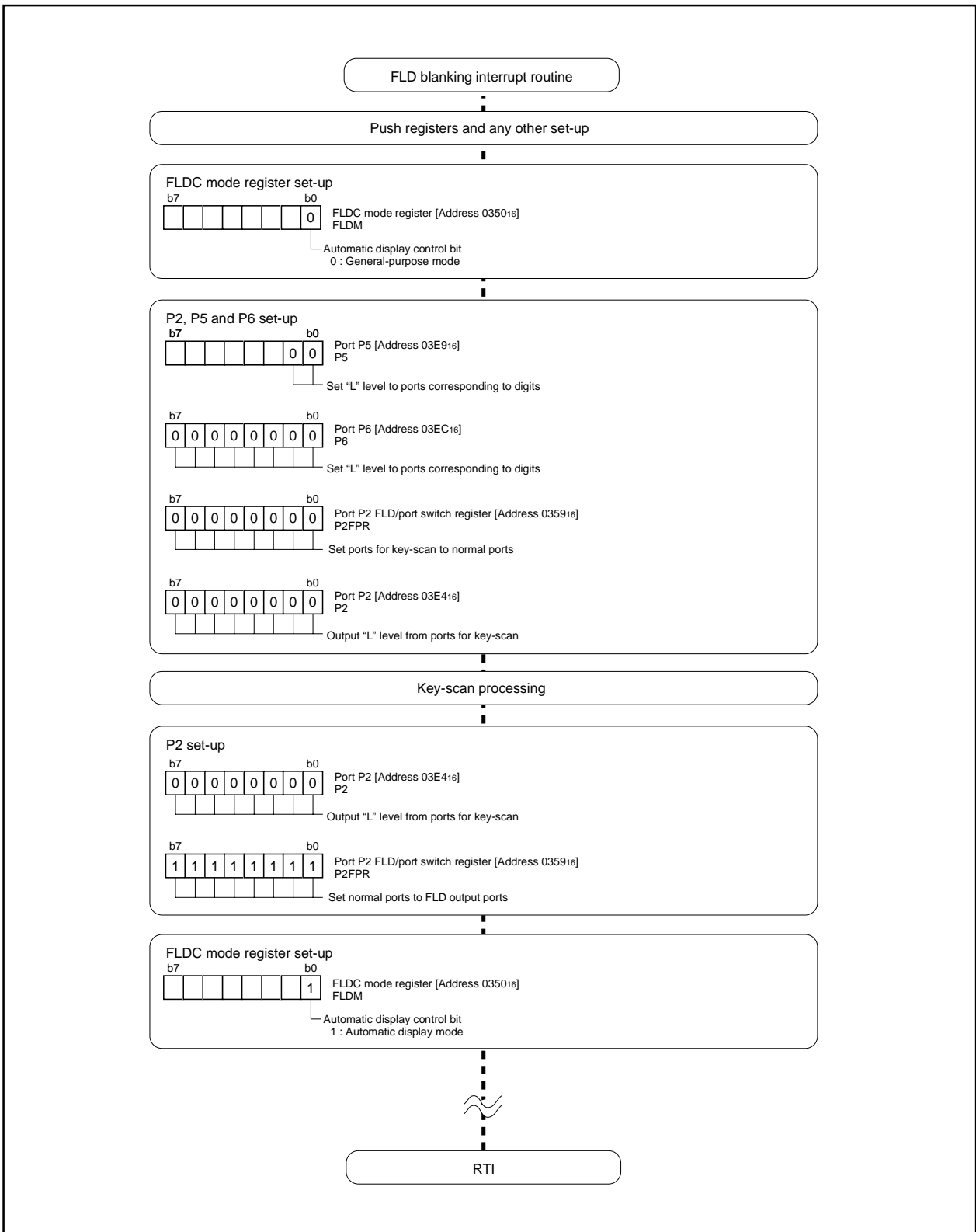


Figure 2.7.10. Set-up procedure for key-scan processing

This page kept blank for layout purposes.

### 2.7.3 FLD operation (FLD automatic display and key-scan using digits)

The FLD controller can choose functions from those listed in Table 2.7.2. The circled items are described in detail below. Figure 2.7.11 shows the operation timing, and Figures 2.7.12 and 2.7.13 show the set-up procedures.

**Table 2.7.2. Selectable functions**

Item	Set-up		Item	Set-up	
Tscan control (Note 1)	<input type="radio"/>	FLD digit interrupt	High-breakdown voltage port drivability		Strong
		FLD blanking interrupt		<input type="radio"/>	Weak
Timing number	<input type="radio"/>	16-timing	P97 dimmer output	<input type="radio"/>	Normal port
		32-timing			Dimmer output
Tdisp counter count source	<input type="radio"/>	$f(X_{IN})/32$	High-breakdown-voltage ports: Section of Toff generate/not generate		Section of Toff does NOT generate
		$f(X_{IN})/128$		<input type="radio"/>	Section of Toff generates
Gradation display mode (Note 2)		Not selecting	Toff2 SET/RESET	<input type="radio"/>	Reset at Toff2
	<input type="radio"/>	Selecting			Set at Toff2

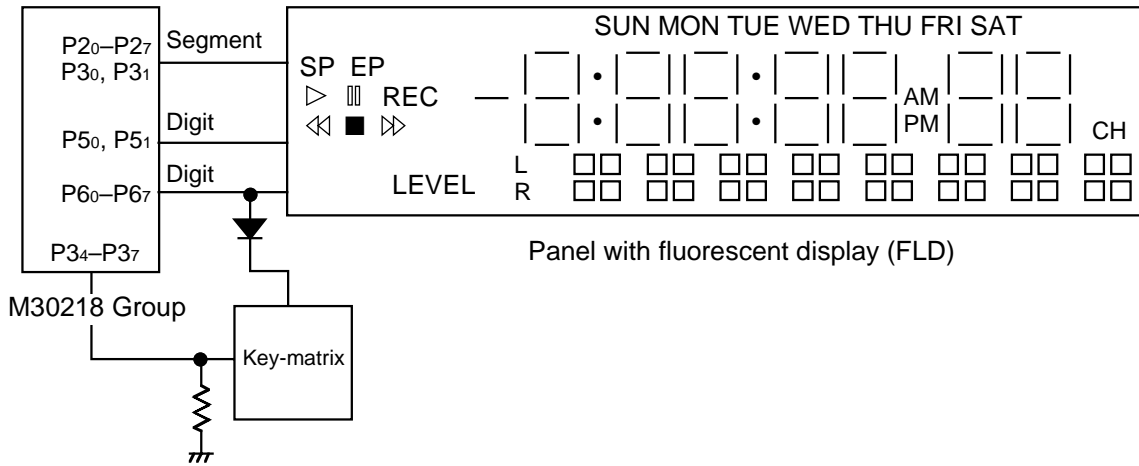
Note 1: When selecting the FLD blanking interrupt, any one of 1 X Tdisp, 2 X Tdisp, or 3 X Tdisp can be selected as Tscan time.

Note 2: When selecting the gradation display mode, make sure to use 16-timing as the timing number.

- Operation
- (1) The FLD starts an automatic display when both the automatic display control bit and the display start bit are set to "1".
  - (2) The display data, the contents from the first address through the last address, in the FLD automatic display RAM for each port is output to each port. The last address is the result of decreasing the number indicated in the FLD data pointer from the first address. The gradation display control data is arranged at an address which is calculated by subtracting "70<sub>16</sub>" from the stored address in the FLD automatic display RAM of the corresponding timing and pin. Bright display is performed by setting "0", and dark display is performed by setting "1". However, the contents of the FLD automatic display RAM for ports P50, P51, and P60 to P67 are disabled by selection of the digit pulse output function, and the digit pulses are automatically output.
  - (3) The FLD data pointer counts down during Tdisp time. When the count reaches "FF<sub>16</sub>", the pointer is reloaded and starts counting over again.
  - (4) The FLD interrupt request bit is set to "1" simultaneously with the end of Toff1 time (at the rising edge of a digit) for each timing. Key scanning, which makes use of FLD digits, can be applied by using each FLD digit interrupt.
  - (5) During FLD automatic display, the FLD automatic display can be interrupted by writing "0" to the display start bit.

FLD controller

Connection example



Operation example

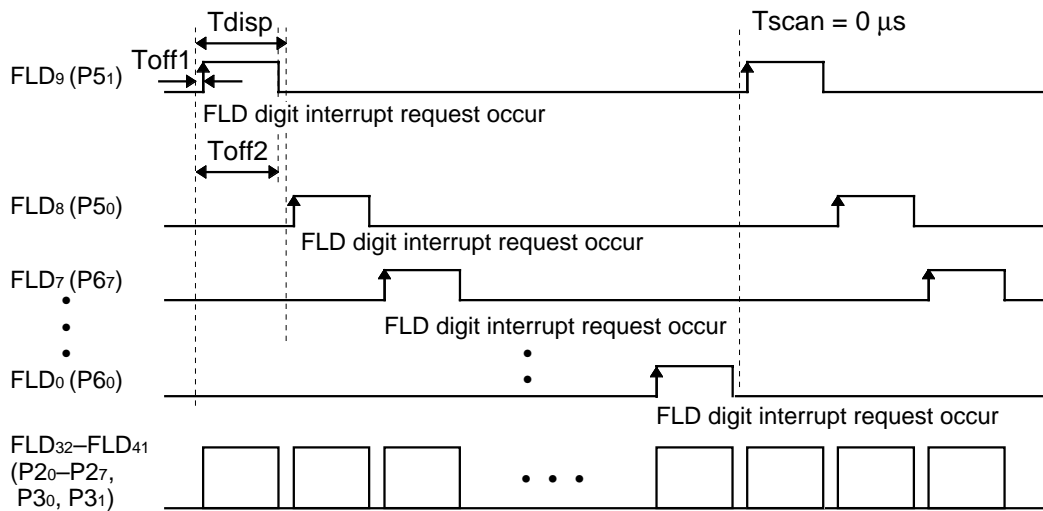


Figure 2.7.11. Operation timing of FLD automatic display

FLD controller

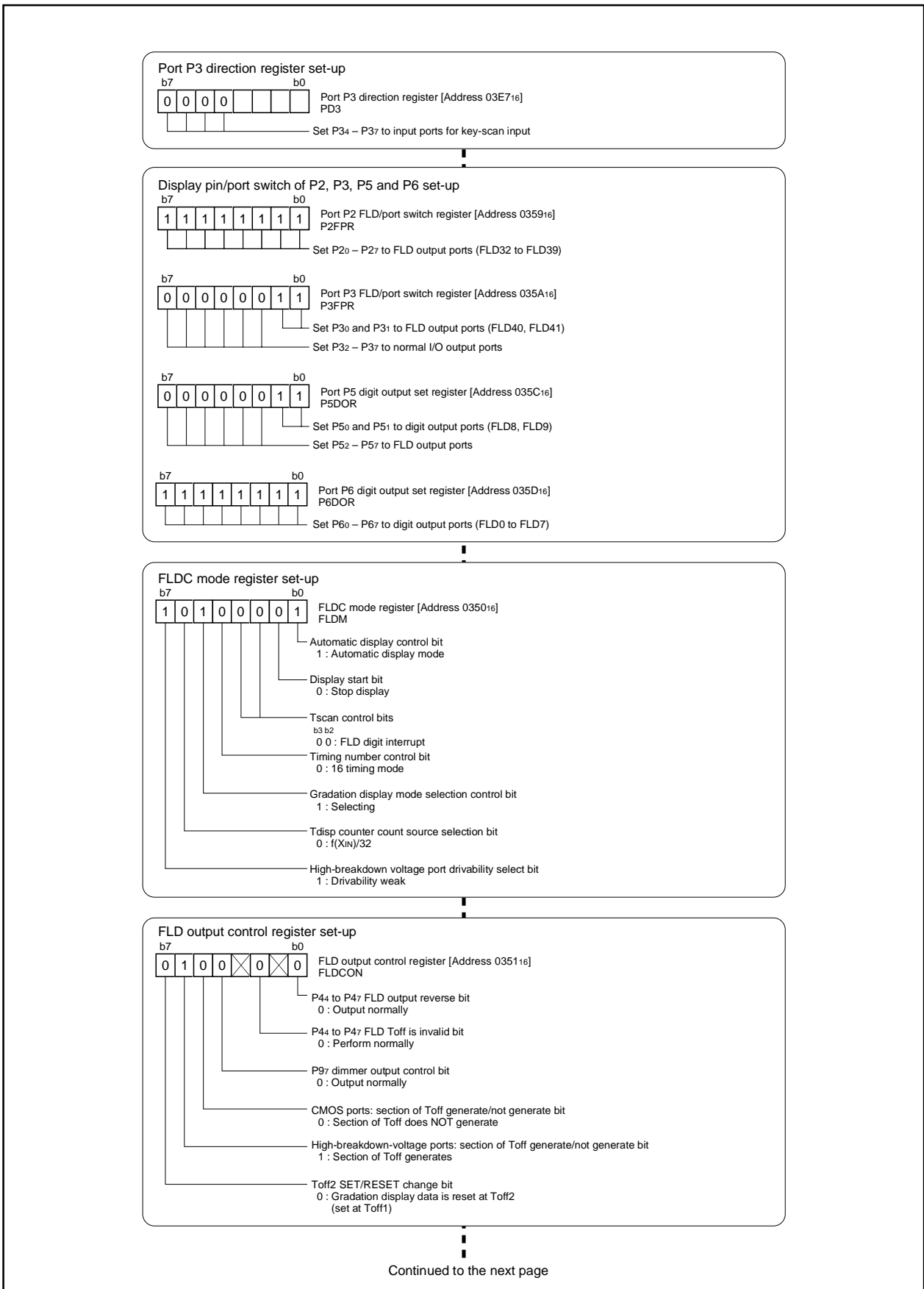
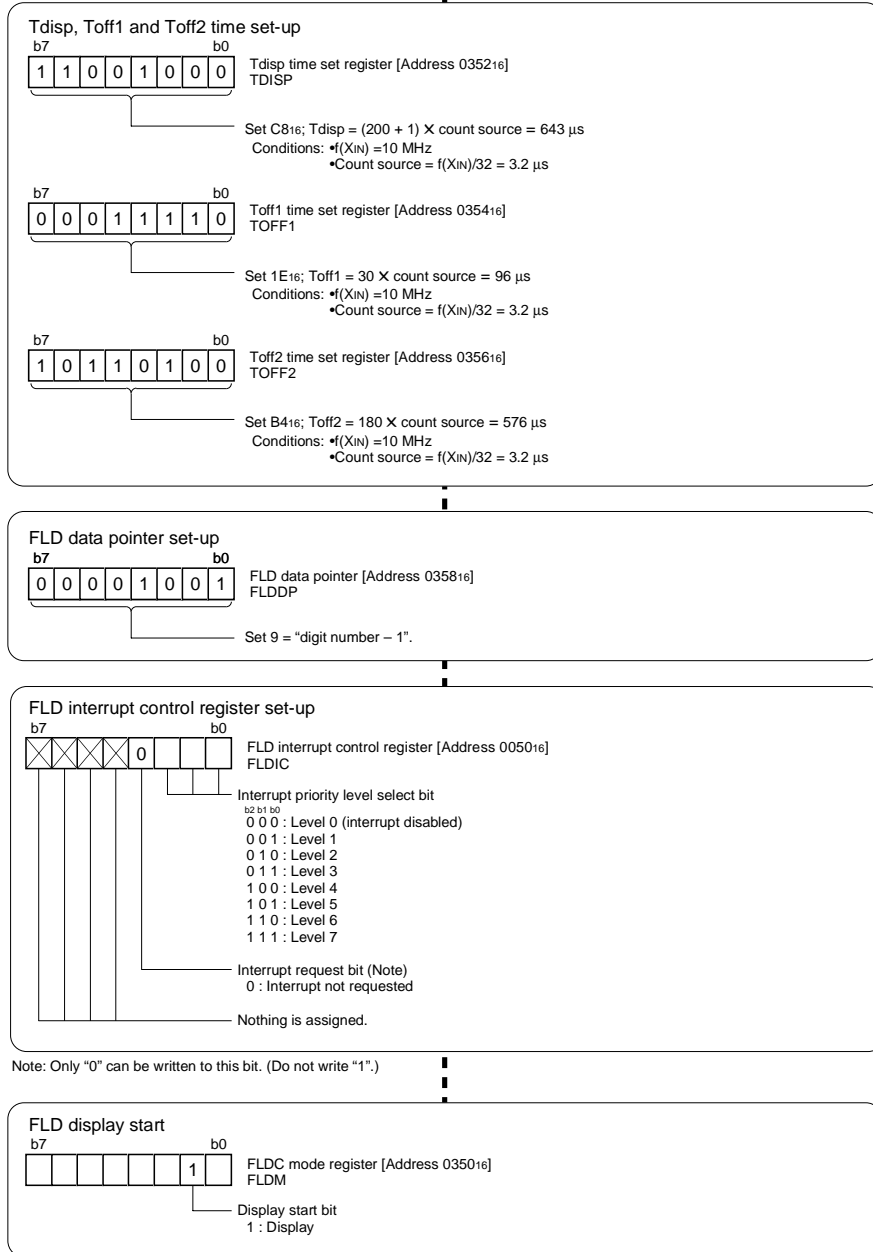


Figure 2.7.12. Set-up procedure for FLD automatic display (1)

FLD controller

Continued from the previous page



FLD display start

Figure 2.7.13. Set-up procedure for FLD automatic display (2)



### 2.7.4 FLD operation (FLD display and key-scan using segment by software)

FLD display and key-scan using the Timer A0 interrupt are explained in detail below. Figure 2.7.14 shows the operation timing, and Figures 2.7.15 to 2.7.17 show the set-up procedures.

- Operation
- (1) Set both the automatic display control bit and the display start bit to "0".
  - (2) Output segment data and digit data from each port during the Timer A0 interrupt processing.
  - (3) After finishing display of all digits, perform key-scan within during the Timer A0 interrupt processing.

FLD controller

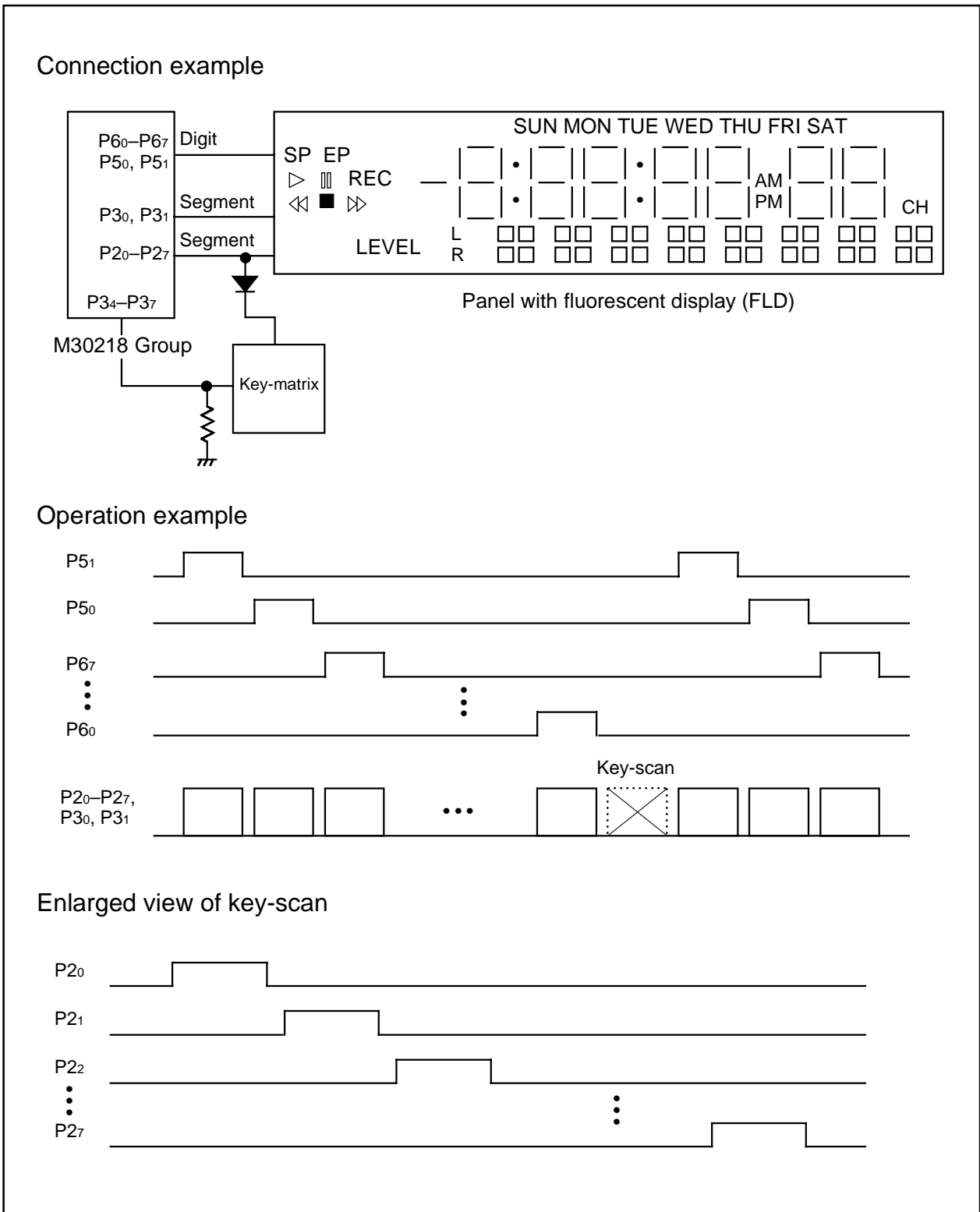


Figure 2.7.14. Operation timing of FLD display

FLD controller

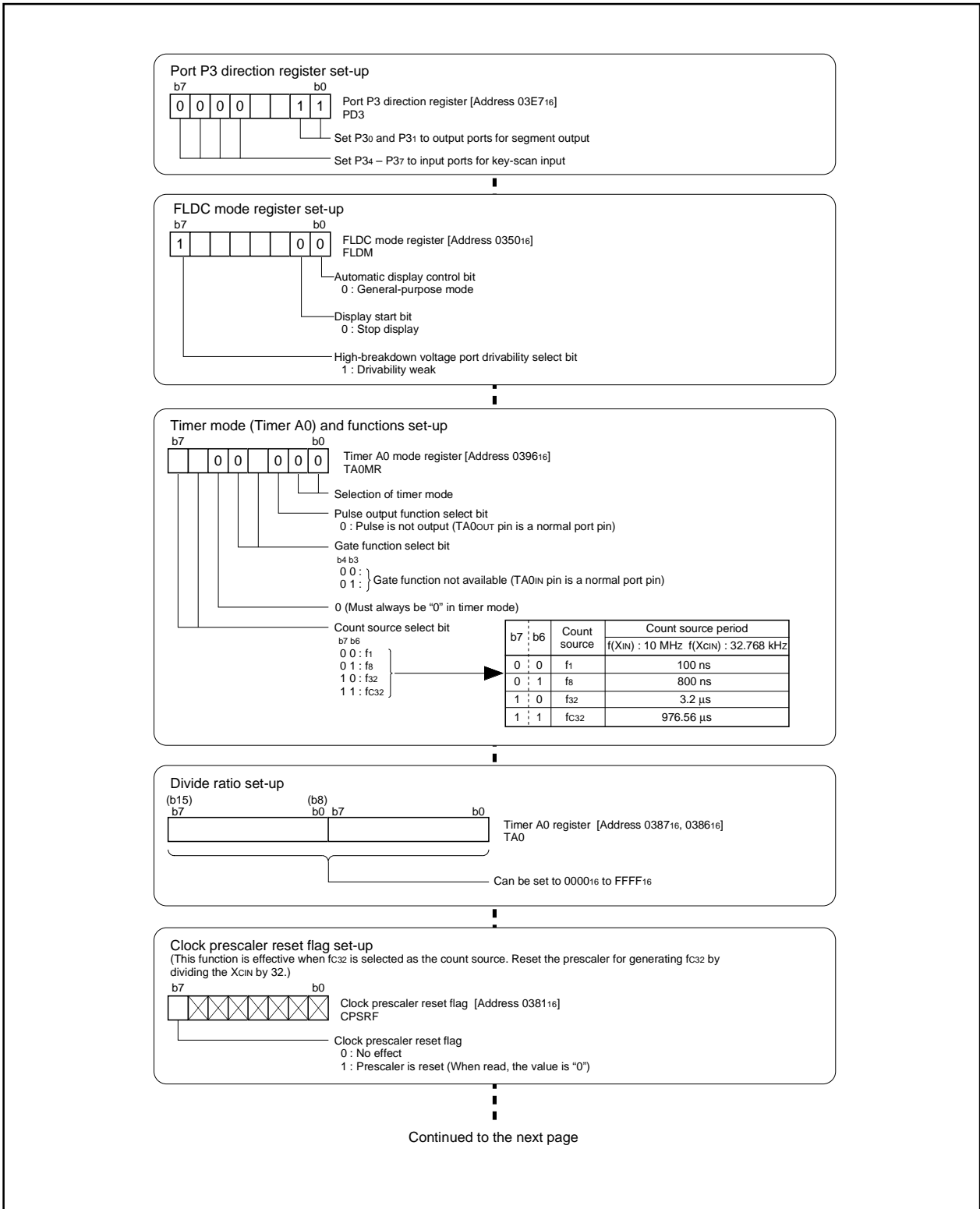
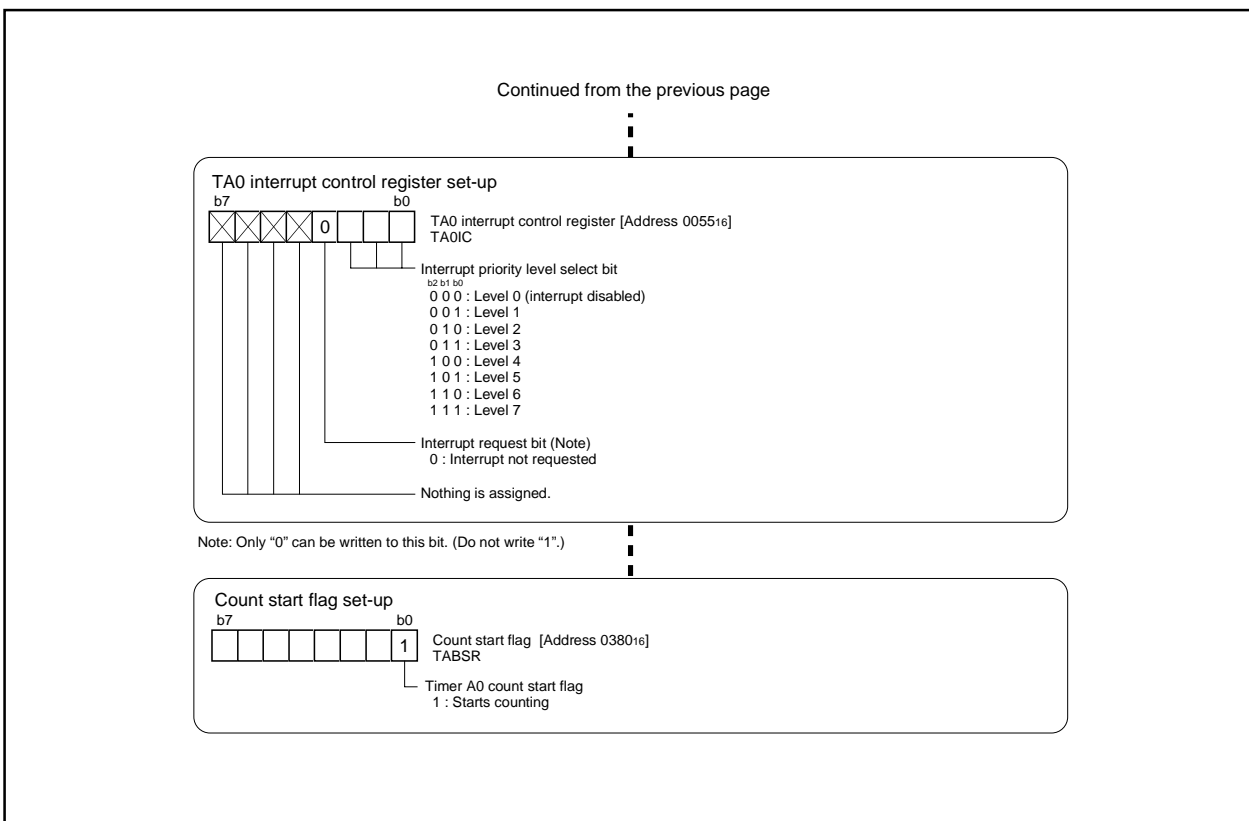


Figure 2.7.15. Set-up procedure for FLD display (1)



**Figure 2.7.16. Set-up procedure for FLD display (2)**

FLD controller

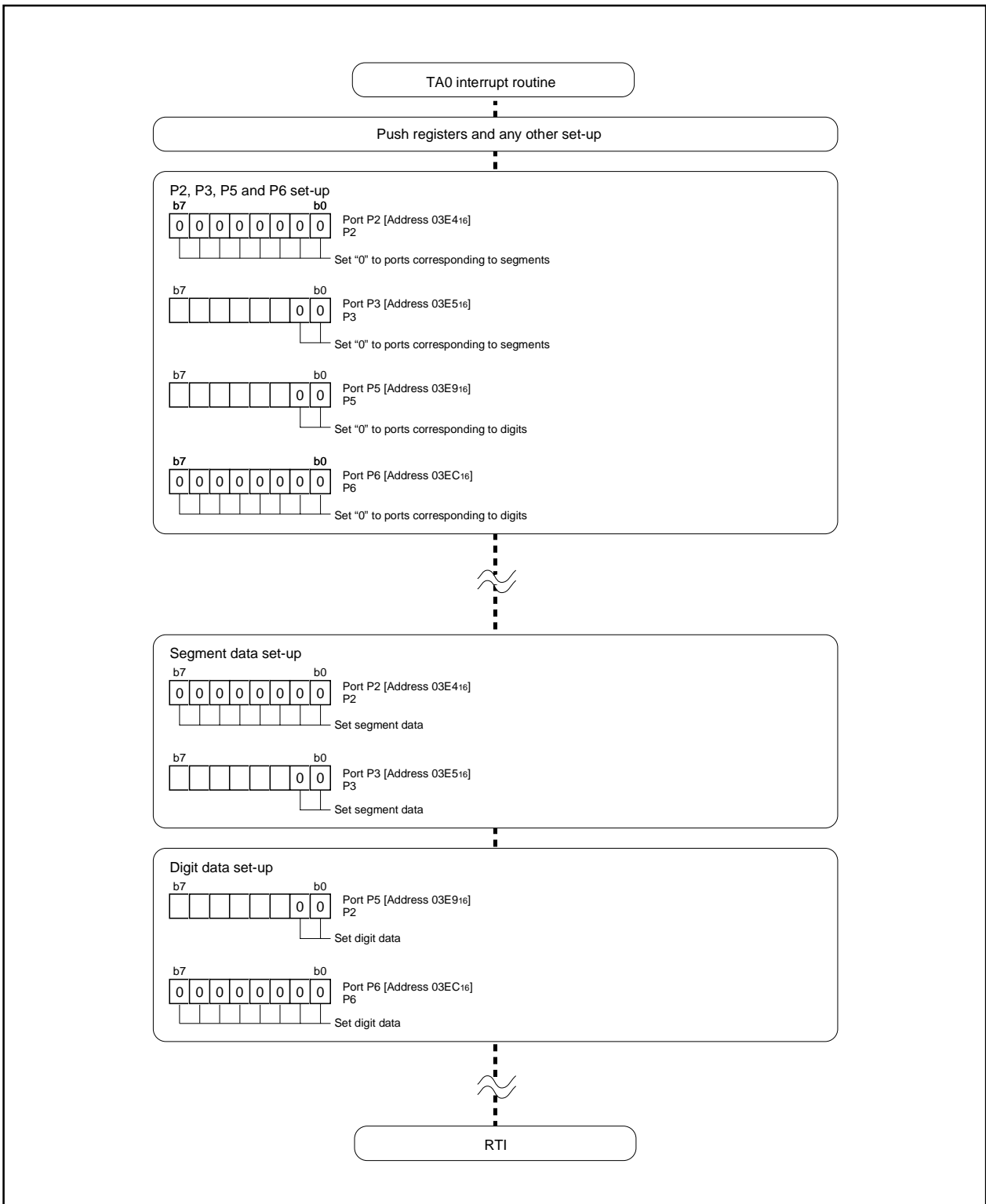


Figure 2.7.17. Set-up procedure for key-scan processing

This page kept blank for layout purposes.

### 2.7.5 FLD operation (Display with digit expander M35501FP)

The FLD controller can choose functions from those listed in Table 2.7.3. The circled items are described in detail below. Figure 2.7.18 shows the connection example and Figure 2.7.19 shows the operation timing, and Figures 2.7.20 and 2.7.21 show the set-up procedures.

Remarks: Also refer to the M35501FP data sheet on <http://www.infocom.mesc.co.jp>

**Table 2.7.3. Selectable functions**

Item	Set-up		Item	Set-up	
Tscan control (Note 1)	<input type="radio"/>	FLD digit interrupt	High-breakdown voltage port drivability		Strong
		FLD blanking interrupt		<input type="radio"/>	Weak
Timing number	<input type="radio"/>	16-timing	P97 dimmer output		Normal port
		32-timing		<input type="radio"/>	Dimmer output
Tdisp counter count source	<input type="radio"/>	f(XIN)/32	High-breakdown-voltage ports: Section of Toff generate/not generate		Section of Toff does NOT generate
		f(XIN)/128		<input type="radio"/>	Section of Toff generates
Gradation display mode (Note 2)		Not selecting	Toff2 SET/RESET	<input type="radio"/>	Reset at Toff2
	<input type="radio"/>	Selecting			Set at Toff2

Note 1: When selecting the FLD blanking interrupt, any one of 1 X Tdisp, 2 X Tdisp, or 3 X Tdisp can be selected as Tscan time.

Note 2: When selecting the gradation display mode, make sure to use 16-timing as the timing number.

- Operation
- (1) The FLD starts an automatic display when both the automatic display control bit and the display start bit are set to "1".
  - (2) The display data, the contents from the first address through the last address, in the FLD automatic display RAM for each port is output to each port. The last address is the result of decreasing the number indicated in the FLD data pointer from the first address. The gradation display control data is arranged at an address which is calculated by subtracting "70<sub>16</sub>" from the stored address in the FLD automatic display RAM of the corresponding timing and pin. Bright display is performed by setting "0", and dark display is performed by setting "1".
  - (3) The FLD data pointer counts down during Tdisp time. When the count reaches "FF<sub>16</sub>", the pointer is reloaded and starts counting over again.
  - (4) Supply signals to the RESET pin and SEL pin of the M35501FP from ports P70 and P71, respectively. Supply the dimmer signal to the CLK pin from the DIMOUT (P97).
  - (5) During FLD automatic display, the FLD automatic display can be interrupted by writing "0" to the display start bit.

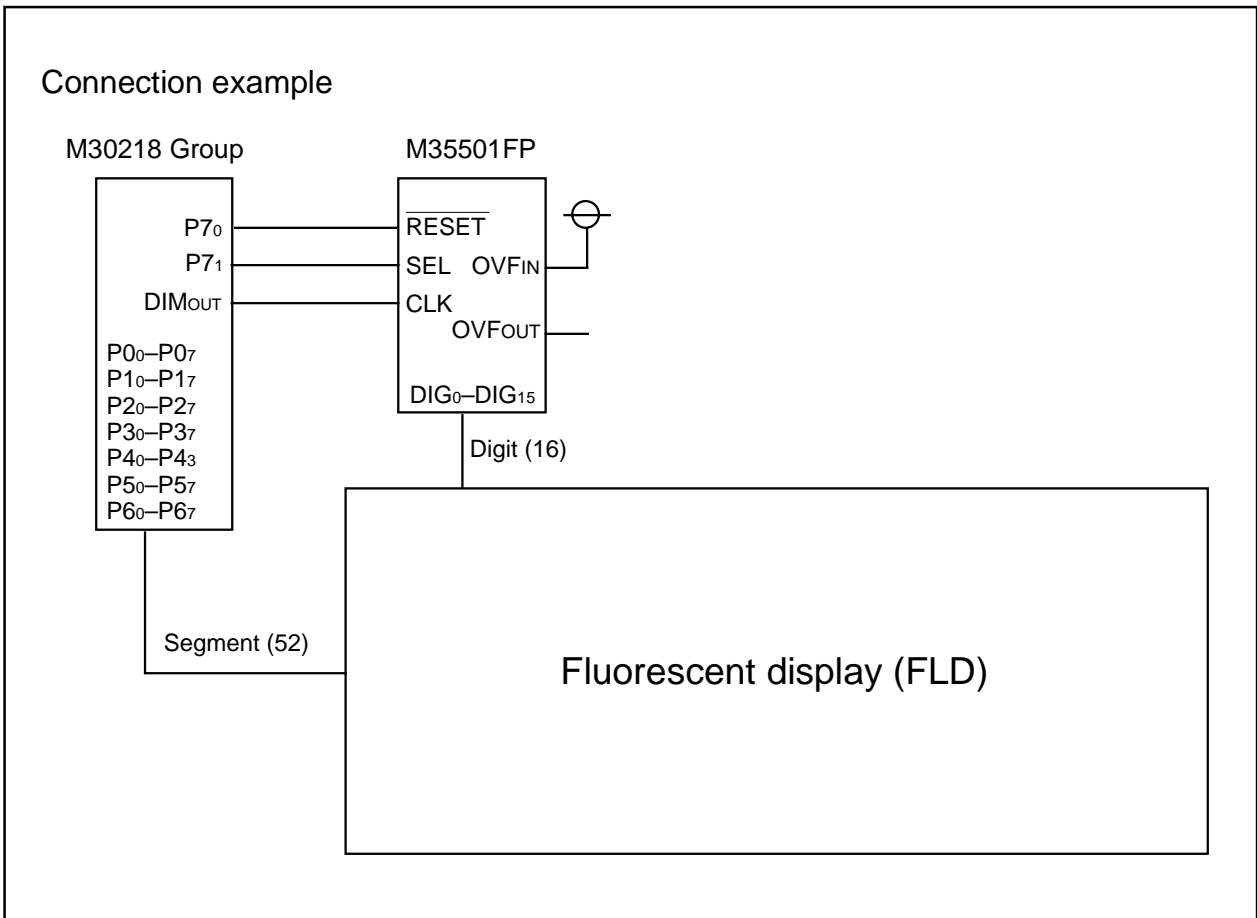


Figure 2.7.18. Connection example of FLD automatic display (1)



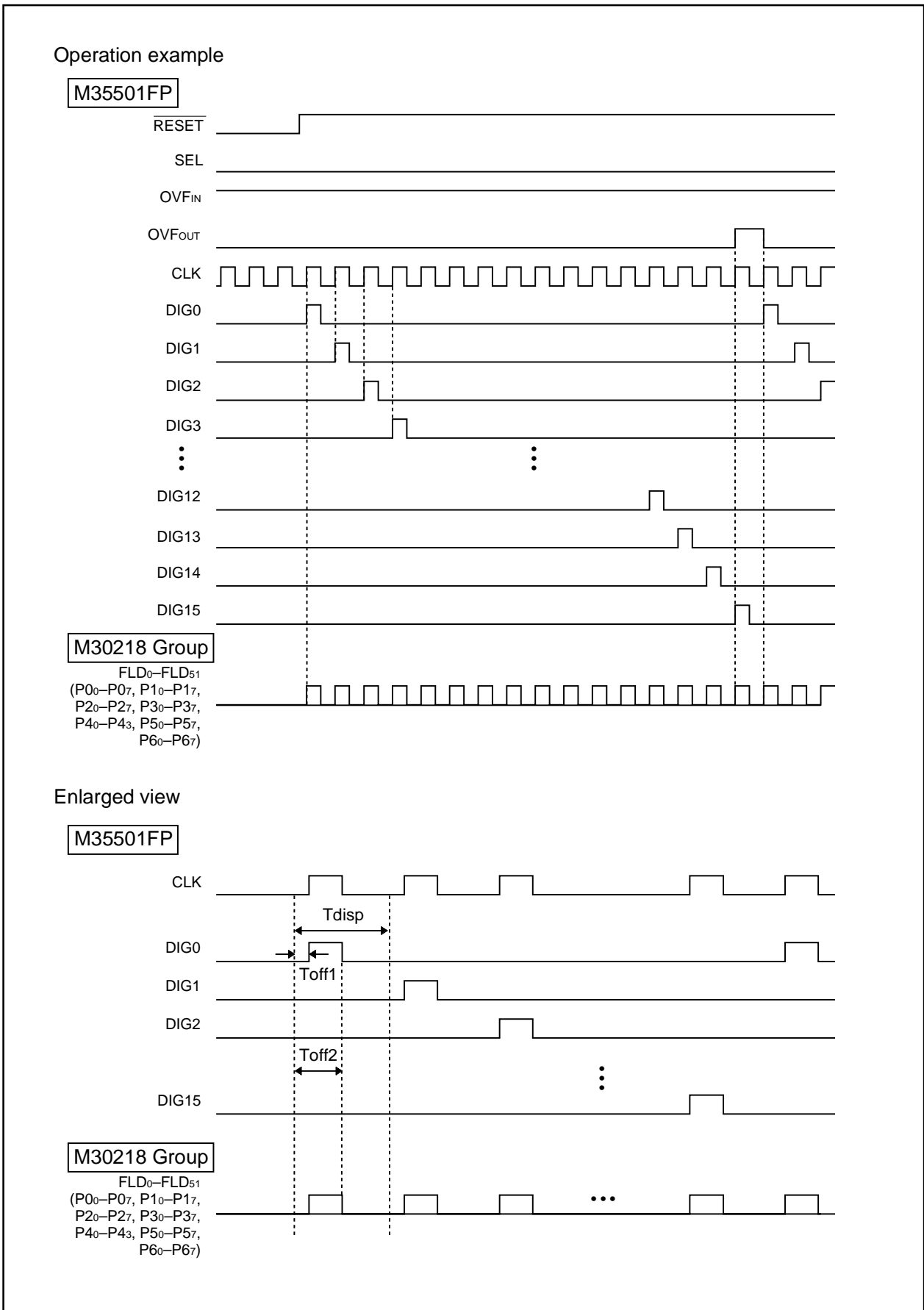
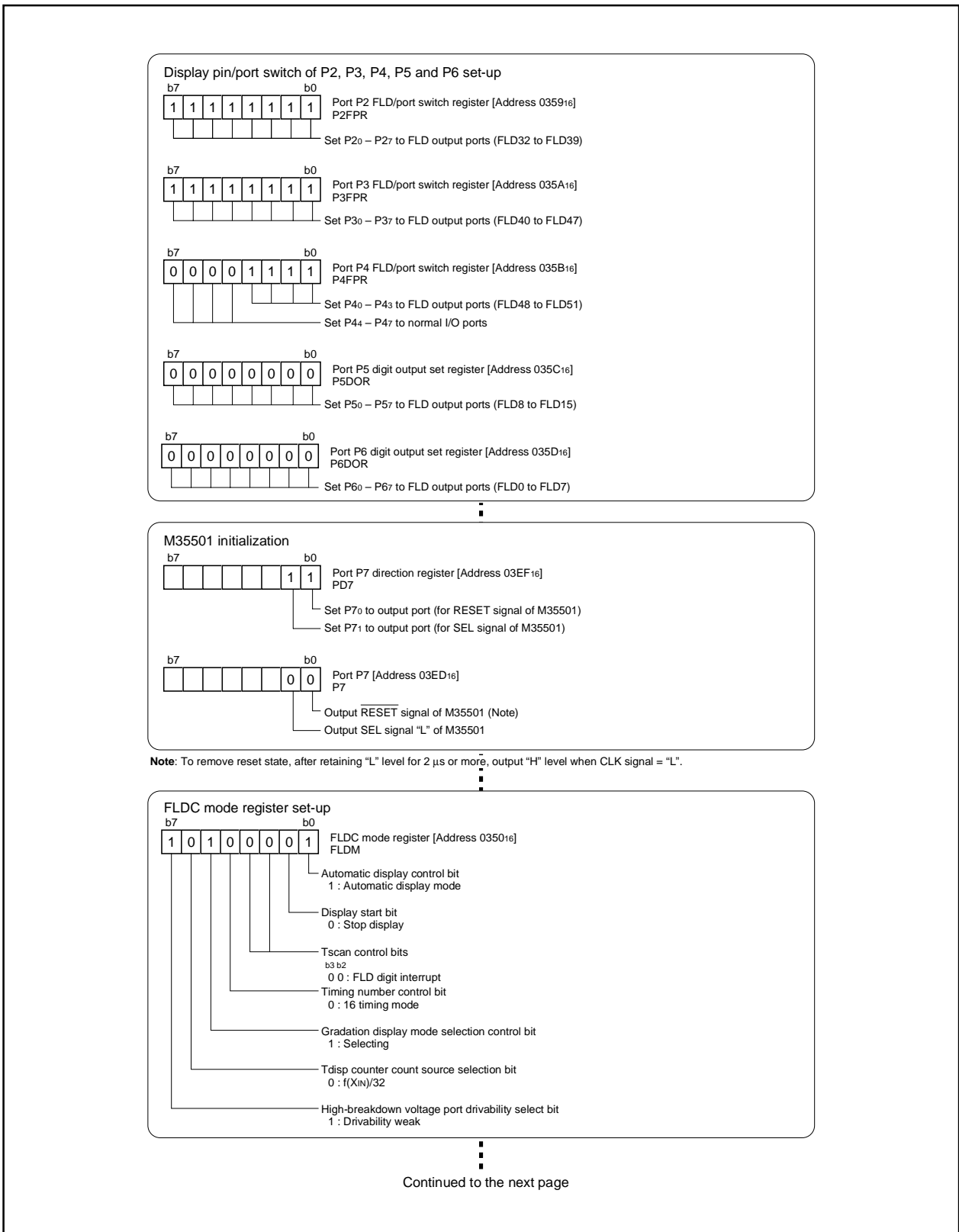


Figure 2.7.19. Operation timing of FLD automatic display



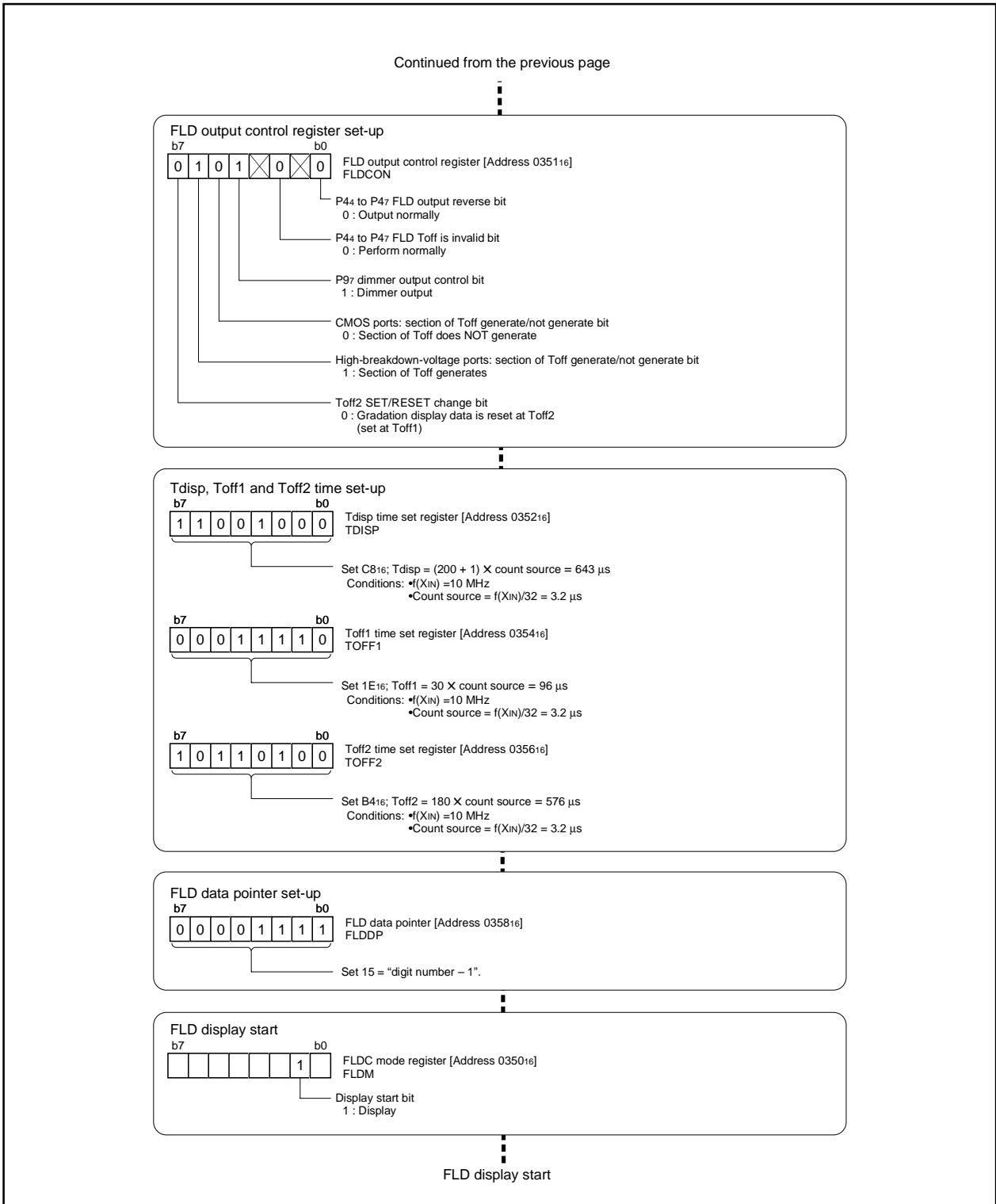


Figure 2.7.21. Set-up procedure for FLD automatic display (2)

This page kept blank for layout purposes.

### 2.7.6 FLD operation (Display with digit expander M35501FP: column discrepancy)

The FLD controller can choose functions from those listed in Table 2.7.4. The circled items are described in detail below. Figure 2.7.22 shows the connection example and Figure 2.7.23 shows the operation timing, and Figures 2.7.24 and 2.7.27 show the set-up procedures.

Remarks: Also refer to the M35501FP data sheet on <http://www.infocom.mesc.co.jp>

**Table 2.7.4. Selectable functions**

Item	Set-up		Item	Set-up	
Tscan control (Note 1)	<input type="radio"/>	FLD digit interrupt	High-breakdown voltage port drivability		Strong
		FLD blanking interrupt		<input type="radio"/>	Weak
Timing number	<input type="radio"/>	16-timing	P97 dimmer output		Normal port
		32-timing		<input type="radio"/>	Dimmer output
Tdisp counter count source	<input type="radio"/>	f(XIN)/32	High-breakdown-voltage ports: Section of Toff generate/not generate		Section of Toff does NOT generate
		f(XIN)/128		<input type="radio"/>	Section of Toff generates
Gradation display mode (Note 2)		Not selecting	Toff2 SET/RESET	<input type="radio"/>	Reset at Toff2
	<input type="radio"/>	Selecting			Set at Toff2

Note 1: When selecting the FLD blanking interrupt, any one of 1 X Tdisp, 2 X Tdisp, or 3 X Tdisp can be selected as Tscan time.

Note 2: When selecting the gradation display mode, make sure to use 16-timing as the timing number.

- Operation
- (1) The FLD starts an automatic display when both the automatic display control bit and the display start bit are set to "1".
  - (2) The display data, the contents from the first address through the last address, in the FLD automatic display RAM for each port is output to each port. The last address is the result of decreasing the number indicated in the FLD data pointer from the first address. The gradation display control data is arranged at an address which is calculated by subtracting "70<sub>16</sub>" from the stored address in the FLD automatic display RAM of the corresponding timing and pin. Bright display is performed by setting "0", and dark display is performed by setting "1".
  - (3) The FLD data pointer counts down during Tdisp time. When the count reaches "FF<sub>16</sub>", the pointer is reloaded and starts counting over again.
  - (4) Supply signals to the RESET pin and SEL pin of the M35501FP from ports P70 and P71, respectively. Supply the dimmer signal to the CLK pin from the DIMOUT (P97).
  - (5) Input the OVFOU output of the M35501FP to TB2IN (P72) and count the input signals as a count source with Timer B2. Generate the Timer A0 interrupt at FLD display intervals and confirm the value of Timer B2. If the value is incorrect, reset the M35501FP.
  - (6) During FLD automatic display, the FLD automatic display can be interrupted by writing "0" to the display start bit.

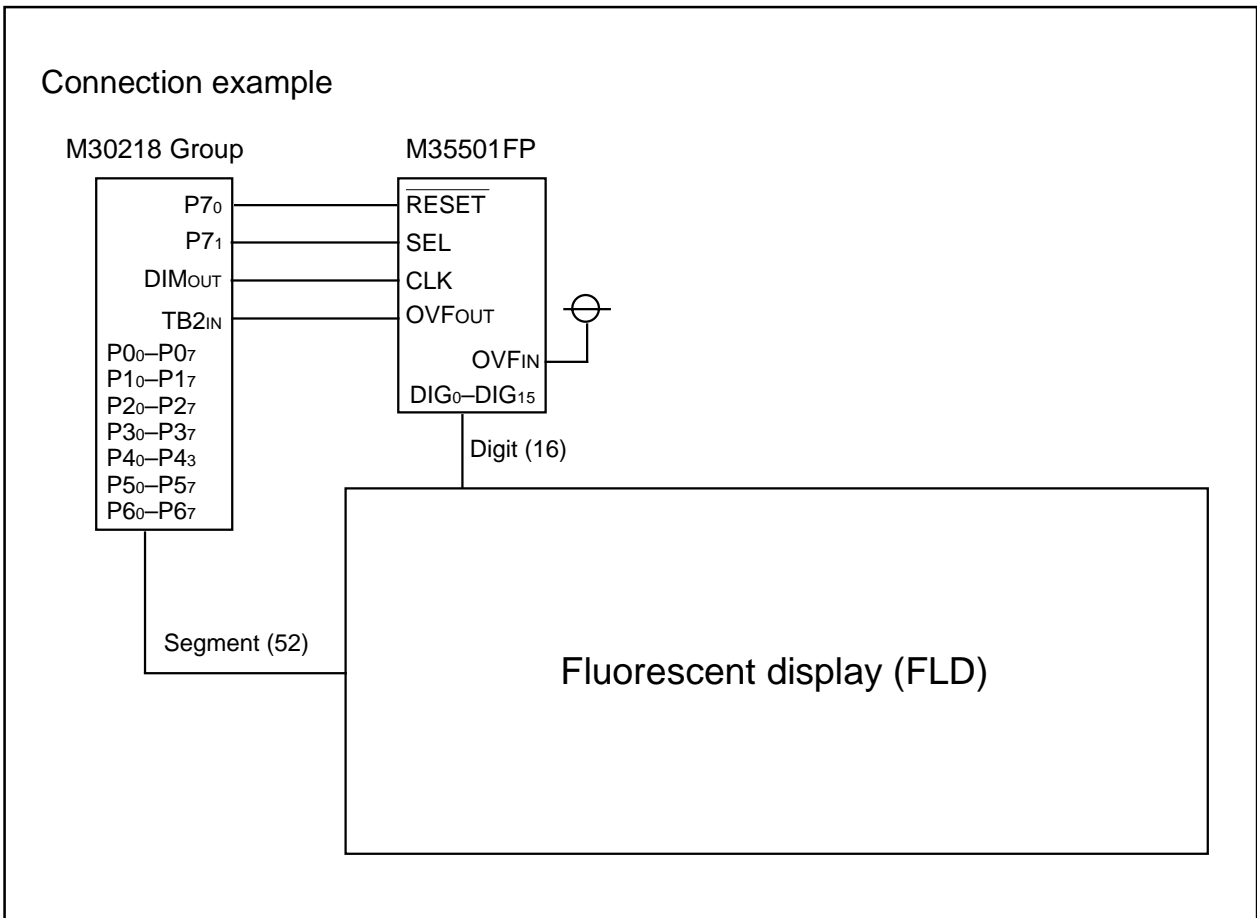


Figure 2.7.22. Connection example of FLD automatic display

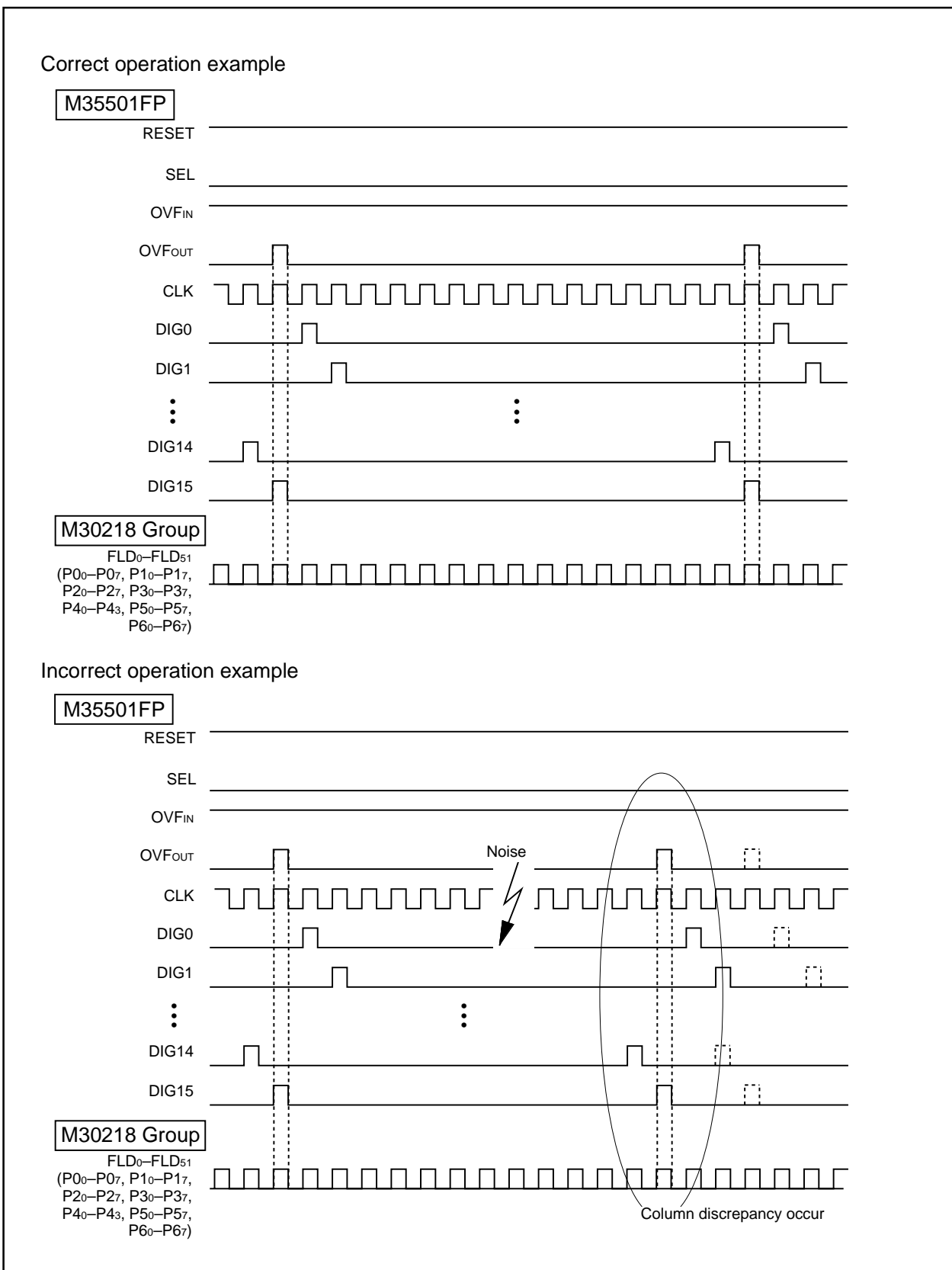


Figure 2.7.23. Operation timing of FLD automatic display

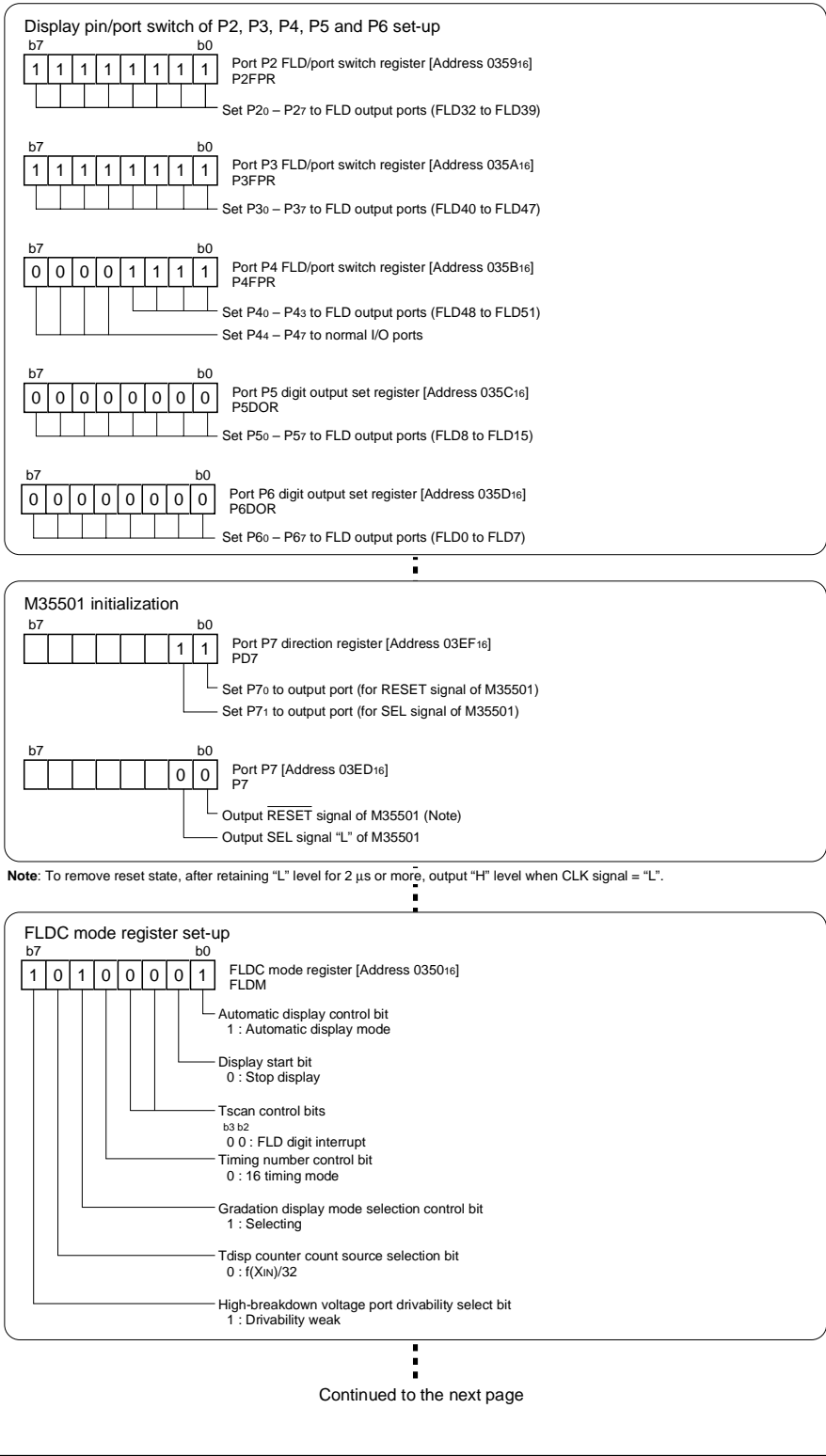
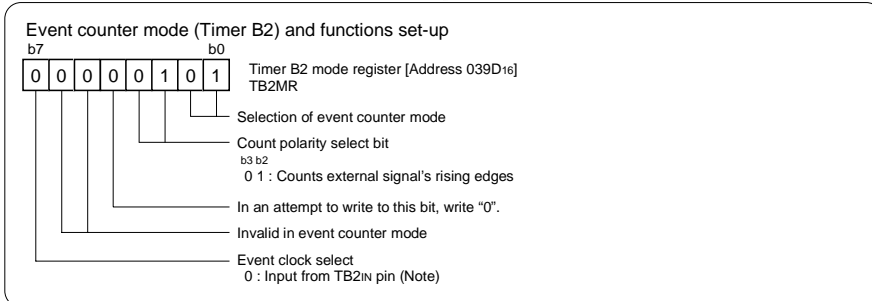
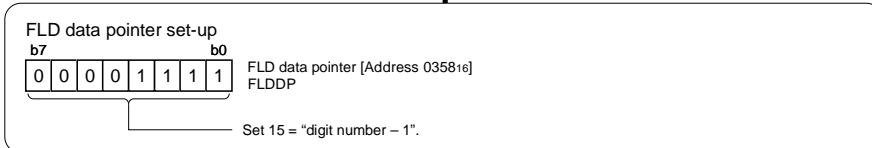
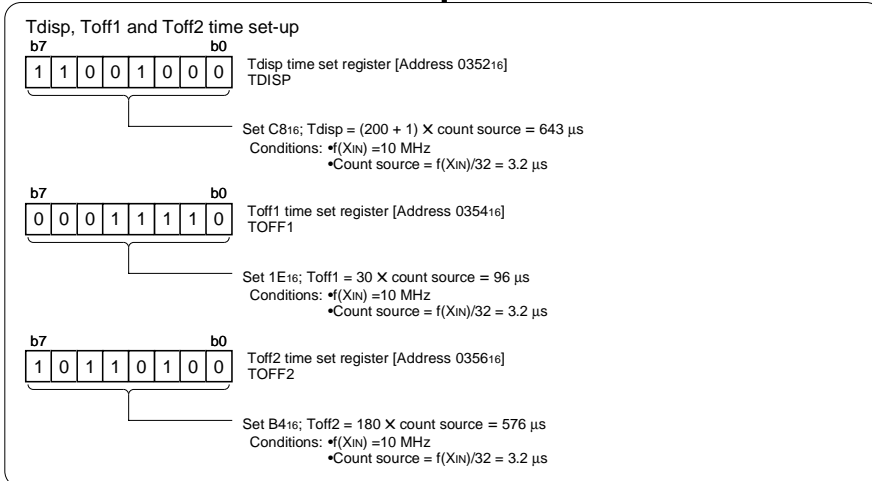
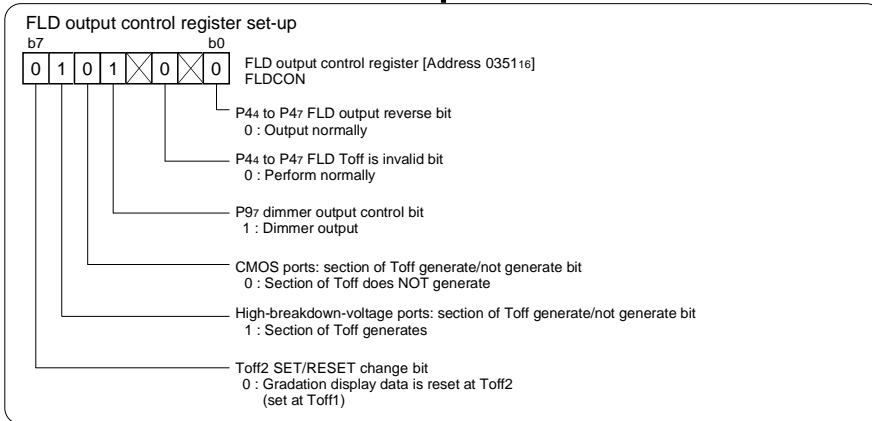


Figure 2.7.24. Set-up procedure for FLD automatic display (1)



Continued from the previous page

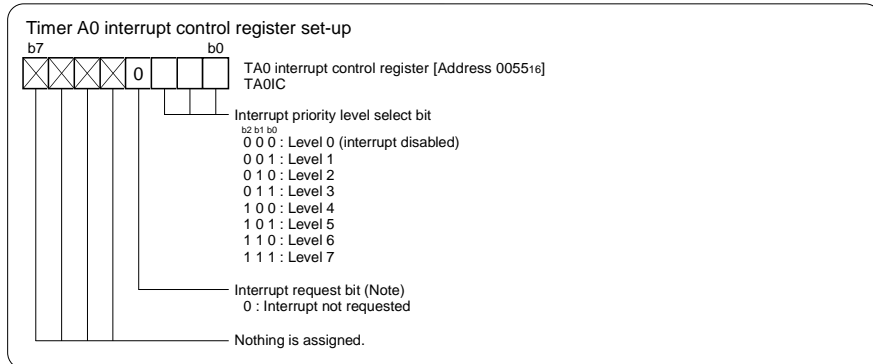
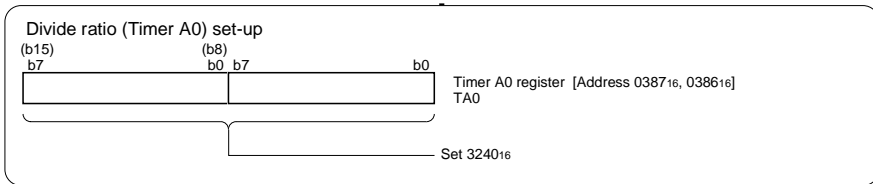
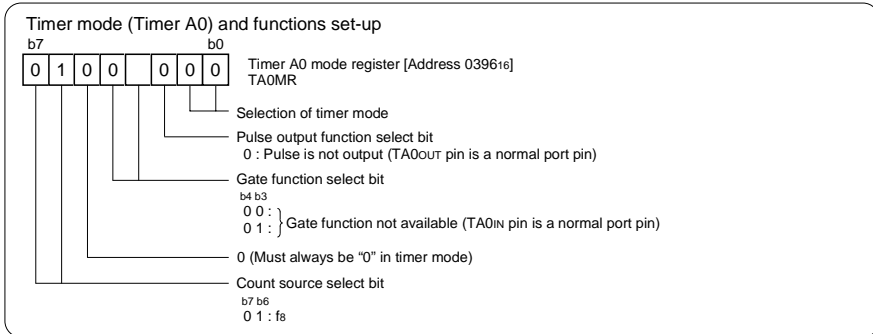
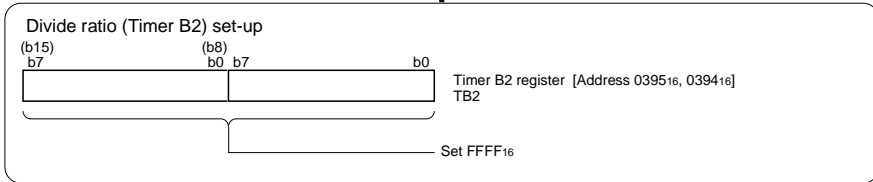


Note: Set the corresponding port direction register to "0".

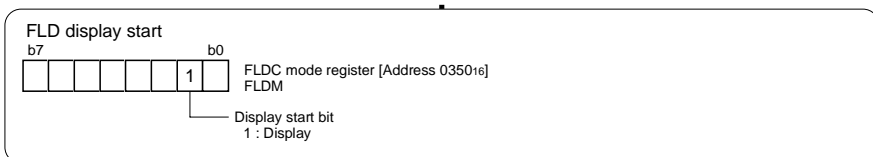
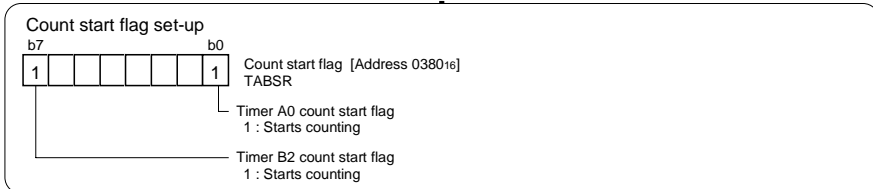
Continued to the next page

Figure 2.7.25. Set-up procedure for FLD automatic display (2)

Continued from the previous page



Note: Only "0" can be written to this bit. (Do not write "1".)



FLD display start

Figure 2.7.26. Set-up procedure for FLD automatic display (3)

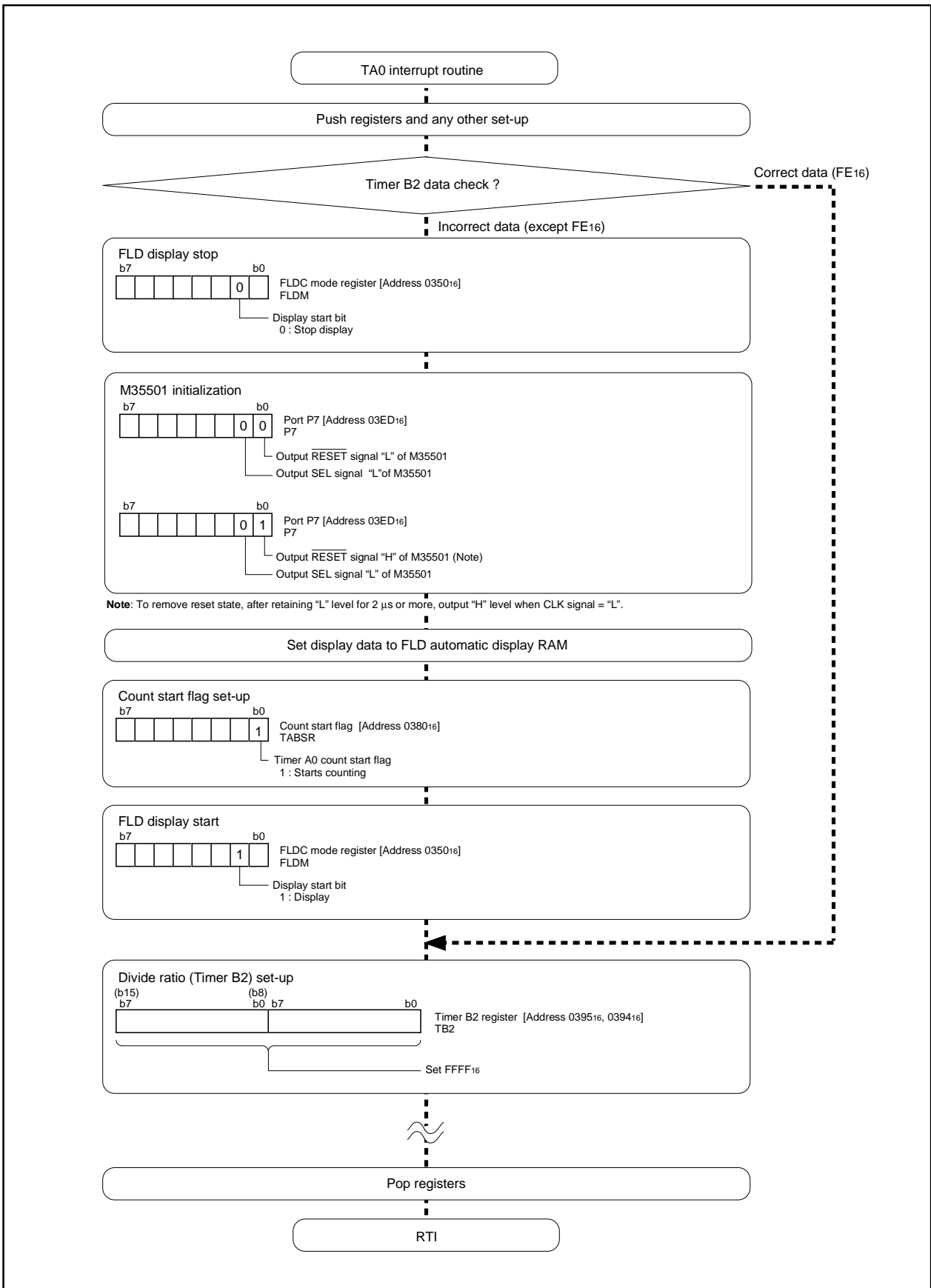


Figure 2.7.27. Set-up procedure for FLD automatic display when detecting column discrepancy

### **2.7.7 Precautions for FLD controller**

- (1) Set a value of "0316" or more to the Toff1 time set register.
- (2) When displaying in the gradation display mode, select the 16-timing mode with the timing number control bit.

## 2.8 A-D Converter

### 2.8.1 Overview

The A-D converter used in the M30218 group operates on a successive conversion basis. The following is an overview of the A-D converter.

#### (1) Mode

The A-D converter operates in one of five modes:

##### (a) One-shot mode

Carries out A-D conversion on input level of one specified pin only once.

##### (b) Repetition mode

Repeatedly carries out A-D conversion on input level of one specified pin.

##### (c) One-shot sweep mode

Carries out A-D conversion on input level of two or more specified pins only once.

##### (d) Repeated sweep mode 0

Repeatedly carries out A-D conversion on input level of two or more pins.

##### (e) Repeated sweep mode 1

Repeatedly carries out A-D conversion on input level of two or more pins. This mode is different from the repeated sweep mode 0 in that weights can be assigned to specifying pins control the number of conversion times.

#### (2) Operation clock

The operation clock can be selected from the following:  $f_{AD}$ , divide-by-2  $f_{AD}$ , and divide-by-4  $f_{AD}$ . The  $f_{AD}$  frequency is equal to that of the CPU's main clock.

#### (3) Conversion time

Number of conversion for A-D convertor varies depending on resolution as given. Table 2.8.1 shows relation between the A-D converter operation clock and conversion time.

Sample & Hold function selected:

33 cycles for 10-bit resolution, or 28 cycles for 8-bit resolution

No Sample & Hold function:

59 cycles for 10-bit resolution, or 49 cycles for 8-bit resolution

**Table 2.8.1. Conversion time every operation clock**

Frequency selection bit 1		0		1
Frequency selection bit 0		0	1	Invalid
A-D converter's operation clock		$\phi_{AD} = \frac{f_{AD}}{4}$	$\phi_{AD} = \frac{f_{AD}}{2}$	$\phi_{AD} = f_{AD}$
Min. conversion cycles (Note 1)	8-bit mode	28 X $\phi_{AD}$		
	10-bit mode	33 X $\phi_{AD}$		
Min. conversion time (Note 2)	8-bit mode	11.2 $\mu$ s	5.6 $\mu$ s	2.8 $\mu$ s
	10-bit mode	13.2 $\mu$ s	6.6 $\mu$ s	3.3 $\mu$ s

Note 1: The number of conversion cycles per one analog input pin.

Note 2: The conversion time per one analog input pin (when  $f_{AD} = f(X_{IN}) = 10$  MHz)

**(4) Functions selection****(a) Sample & Hold function**

Sample & Hold function samples input voltage when A-D conversion starts and carries out A-D conversion on the voltage sampled. When A-D conversion starts, input voltage is sampled for 3 cycles of the operation clock. When the Sample & Hold function is selected, set the operation clock for A-D conversion to 1 MHz or higher.

**(b) 8-bit A-D to 10-bit A-D switching function**

Either 8-bit resolution or 10-bit resolution can be selected. When 8-bit resolution is selected, the 8 higher-order bits of the 10-bit A-D are subjected to A-D conversion. The equations for 10-bit resolution and 8-bit resolution are given below:

$$\text{10-bit resolution} \quad (V_{\text{ref}} \times n / 2^{10}) - (V_{\text{ref}} \times 0.5 / 10^{10}) \quad (n = 1 \text{ to } 1023), 0 (n = 0)$$

$$\text{8-bit resolution} \quad (V_{\text{ref}} \times n / 2^8) - (V_{\text{ref}} \times 0.5 / 2^{10}) \quad (n = 1 \text{ to } 255), 0 (n = 0)$$

**(c) Connecting or cutting Vref**

Cutting Vref allows decrease of the current flowing into the A-D converter. To decrease the microcomputer's power consumption, cut Vref. To carry out A-D conversion, start A-D conversion 1  $\mu$ s or longer after connecting Vref.

The following are examples in which functions (a) through (c) are selected:

- One-shot mode ..... P332
- Repeat mode, software trigger ..... P334
- One-shot sweep mode, software trigger ..... P336
- Repeated sweep mode 0, software trigger ..... P338
- Repeated sweep mode 1, software trigger ..... P340

## A-D Converter

**(5) Input to A-D converter and direction register**

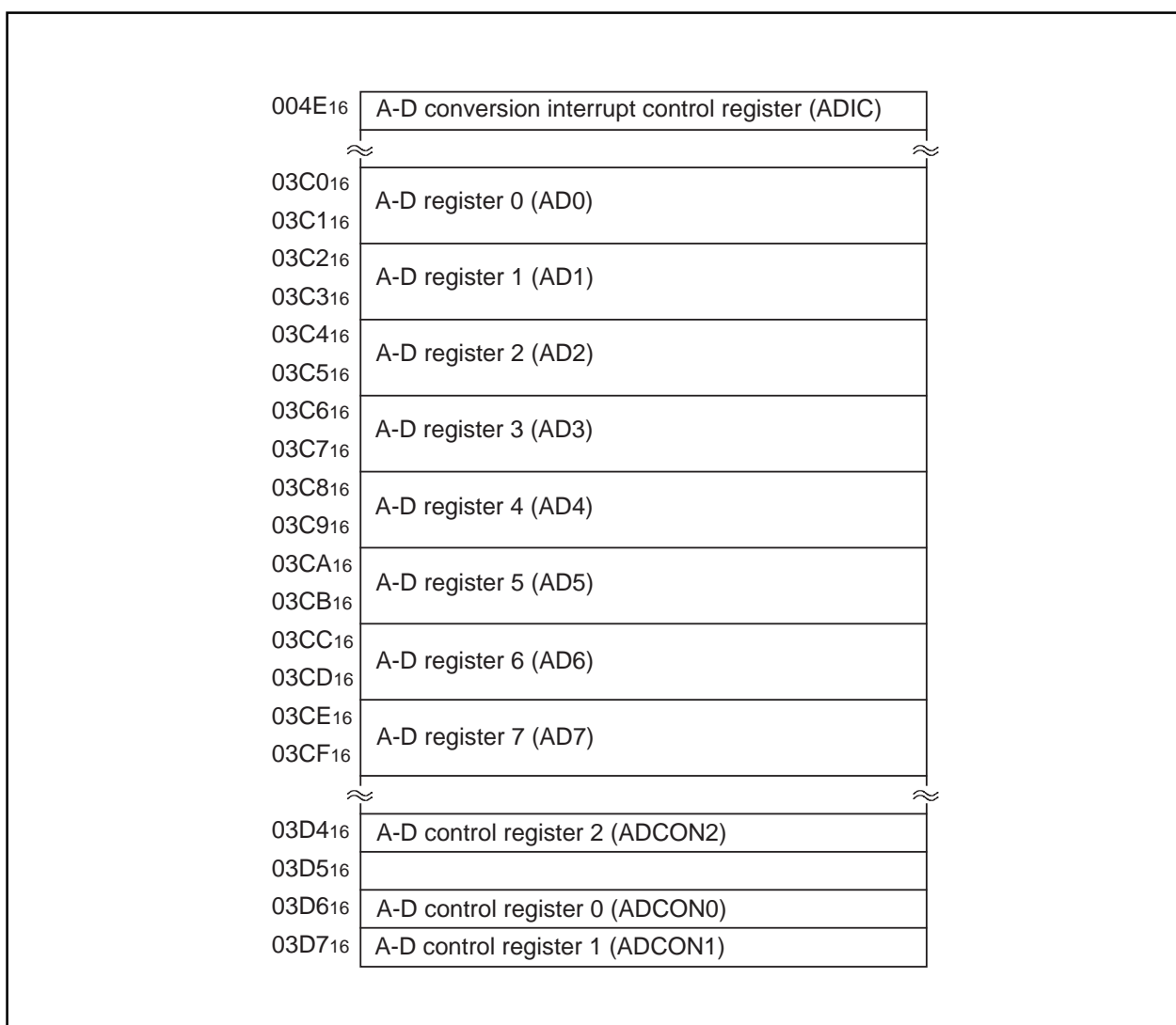
To use the A-D converter, set the direction register of the relevant port to input.

**(6) Pins related to A-D converter**

- |   |  |
|---|--|
| (a) AN <sub>0</sub> pin through AN <sub>7</sub> pin | Input pins of the A-D converter        |
| (b) AV <sub>cc</sub> pin                            | Power source pin of the analog section |
| (c) V <sub>REF</sub> pin                            | Input pin of reference voltage         |
| (d) AV <sub>ss</sub> pin                            | GND pin of the analog section          |

**(7) A-D converter and related registers**

Figure 2.8.1 shows the memory map of A-D converter-related registers, and Figures 2.8.2 through 2.8.4 show A-D converter-related registers.



**Figure 2.8.1. Memory map of A-D converter-related registers**

## A-D Converter

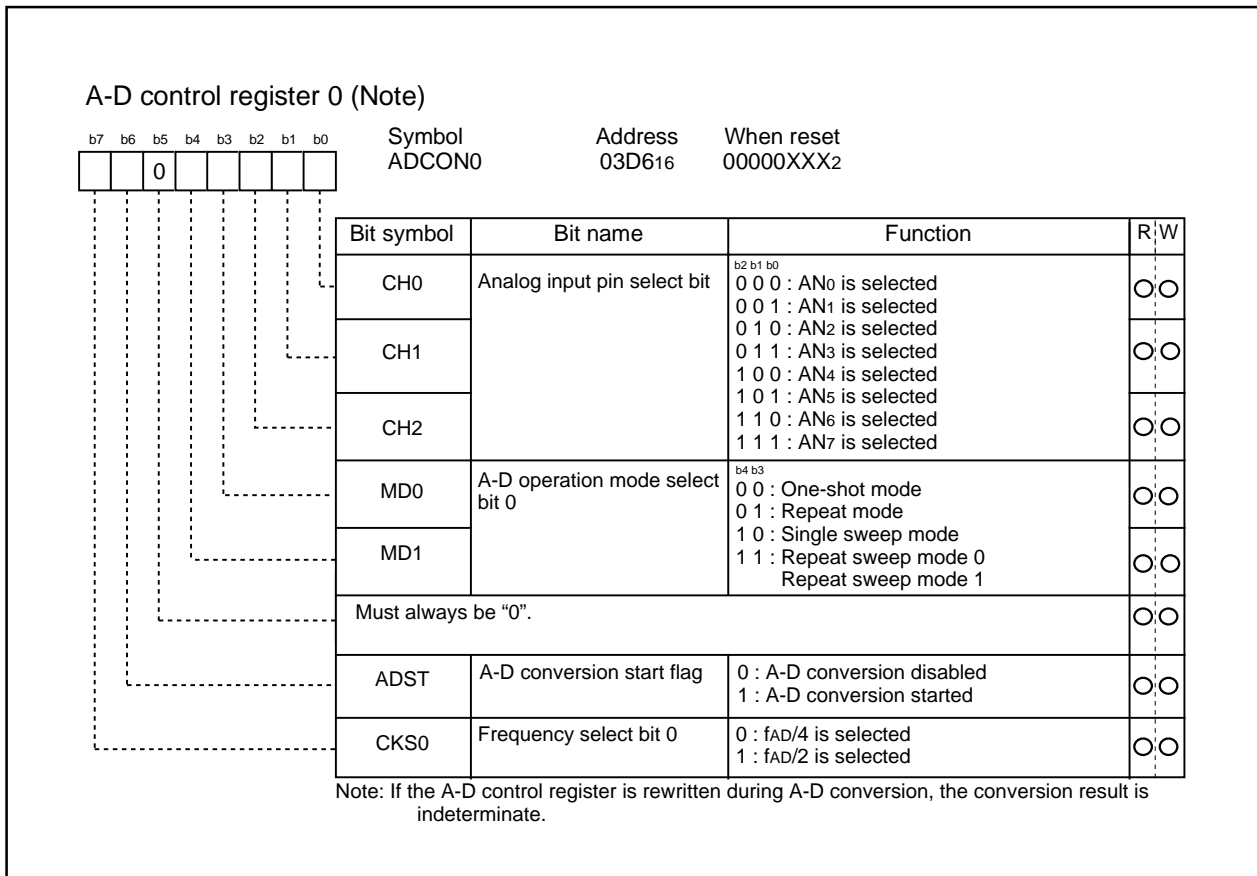


Figure 2.8.2. A-D converter-related registers (1)



## A-D Converter

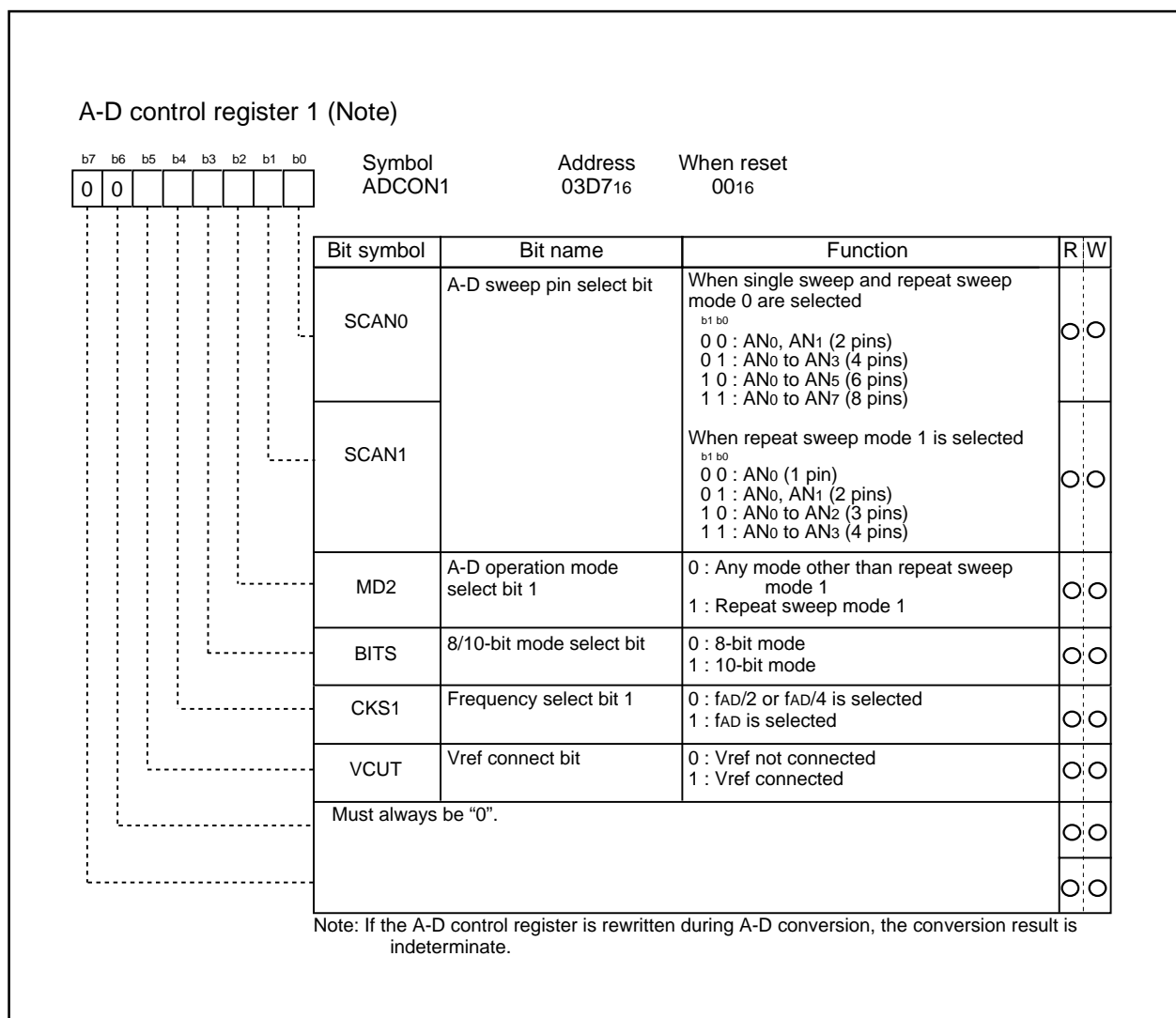


Figure 2.8.3. A-D converter-related registers (2)

A-D Converter

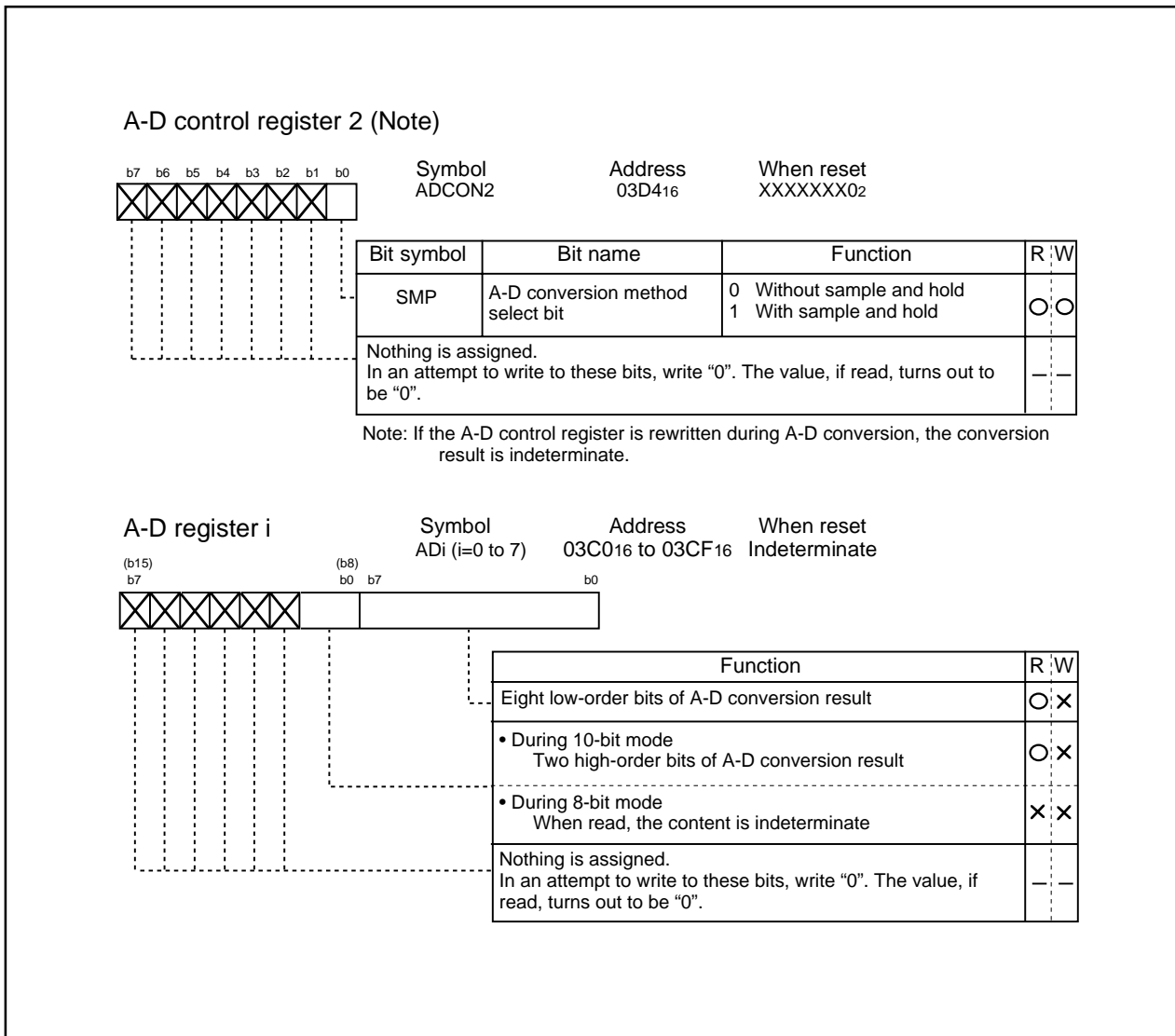


Figure 2.8.4. A-D converter-related registers (3)

## A-D Converter

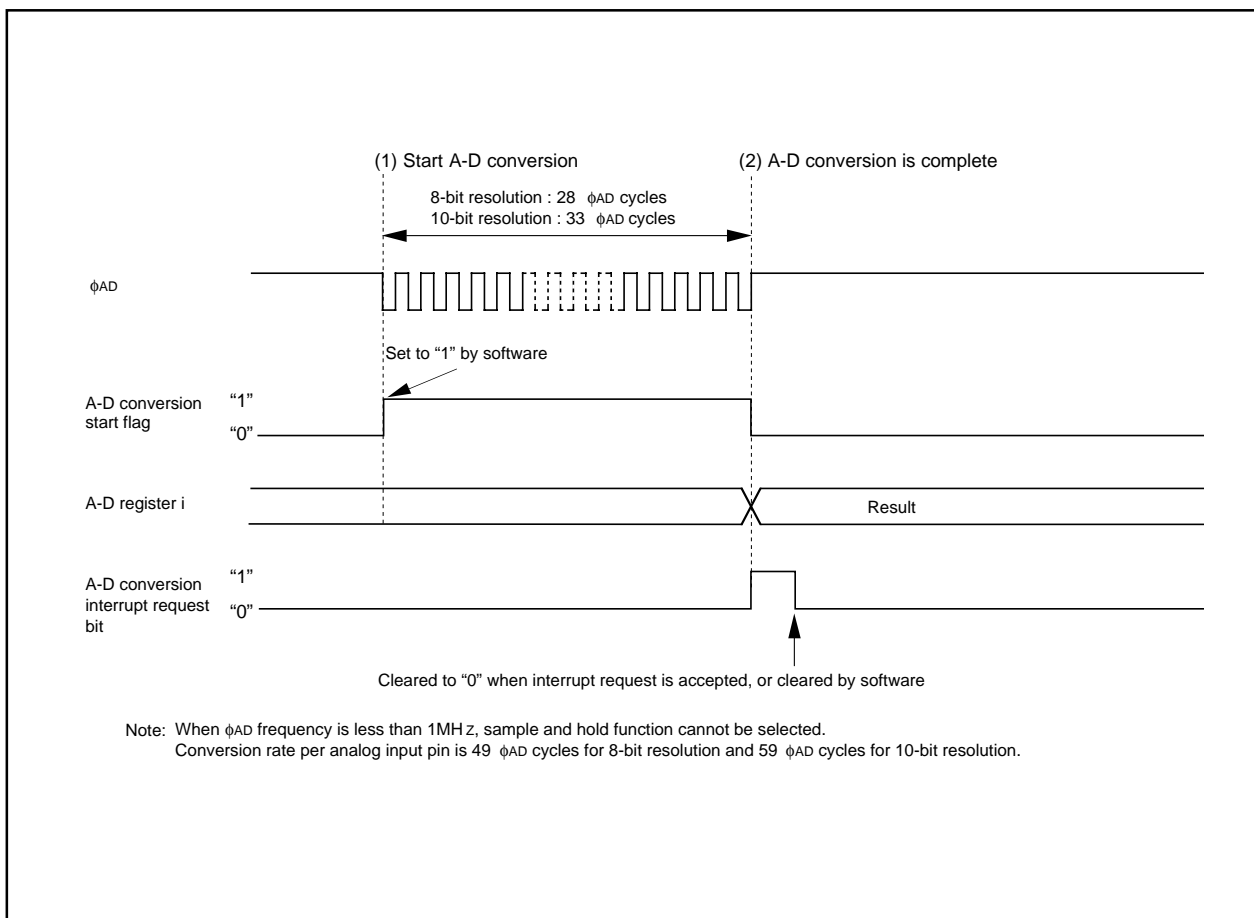
## 2.8.2 Operation of A-D converter (one-shot mode)

In one-shot mode, choose functions from those listed in Table 2.8.2. Operations of the circled items are described below. Figure 2.8.5 shows the operation timing, and Figure 2.8.6 shows the set-up procedure.

**Table 2.8.2. Chosed functions**

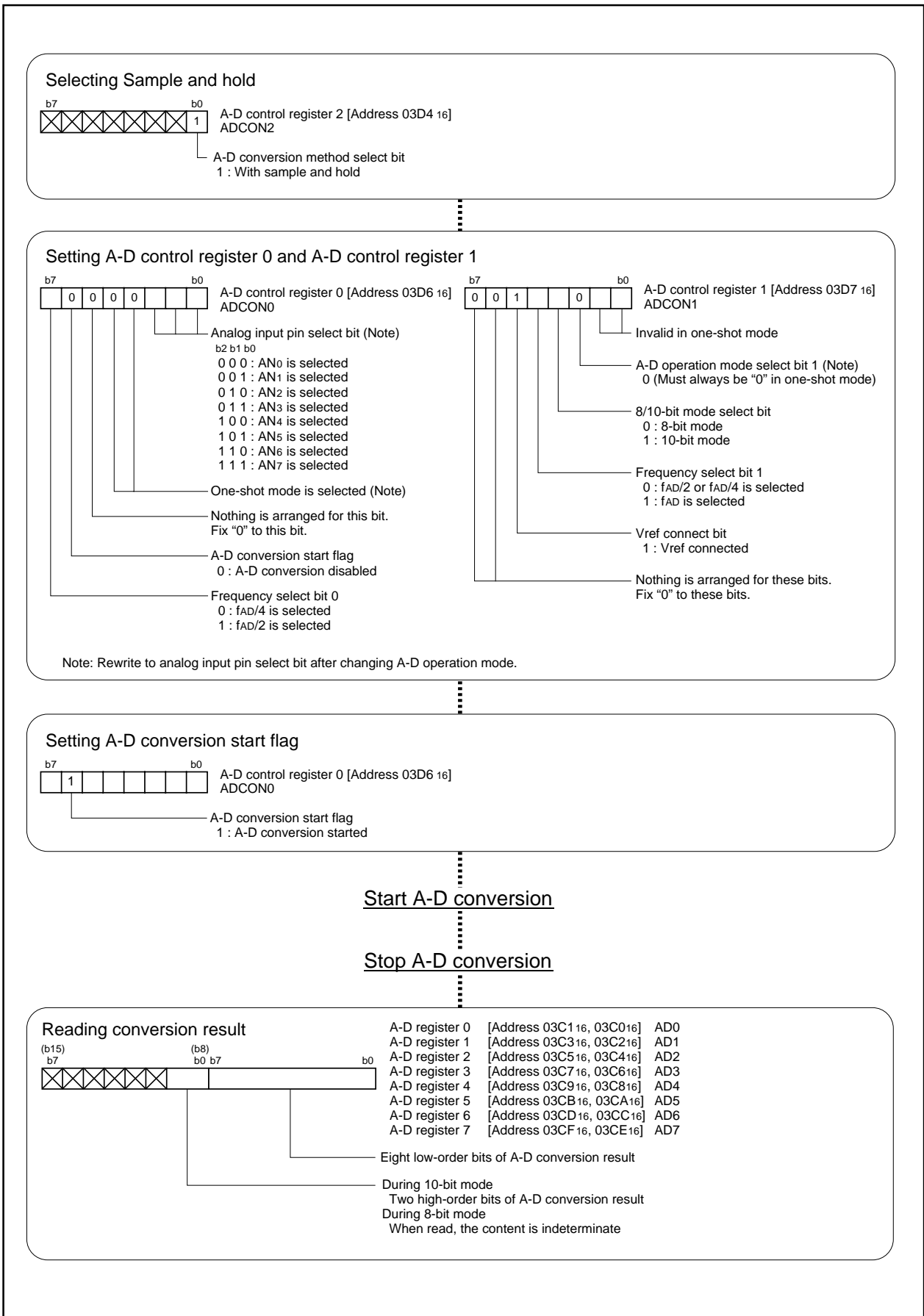
Item	Set-up	
Operation clock $\phi_{AD}$	○	Divided-by-4 $f_{AD}$ / divided-by-2 $f_{AD}$ / $f_{AD}$
Resolution	○	8-bit / 10-bit
Analog input pin	○	One of AN <sub>0</sub> pin to AN <sub>7</sub> pin
Sample & Hold	○	Not activated
		Activated

- Operation (1) Setting the A-D conversion start flag to "1" causes the A-D converter to begin operating.
- (2) After A-D conversion is completed, the content of the successive comparison register (conversion result) is transmitted to A-D register i. At this time, the A-D conversion interrupt request bit goes to "1". Also, the A-D conversion start flag goes to "0", and the A-D converter stops operating.



**Figure 2.8.5. Operation timing of one-shot mode**

A-D Converter



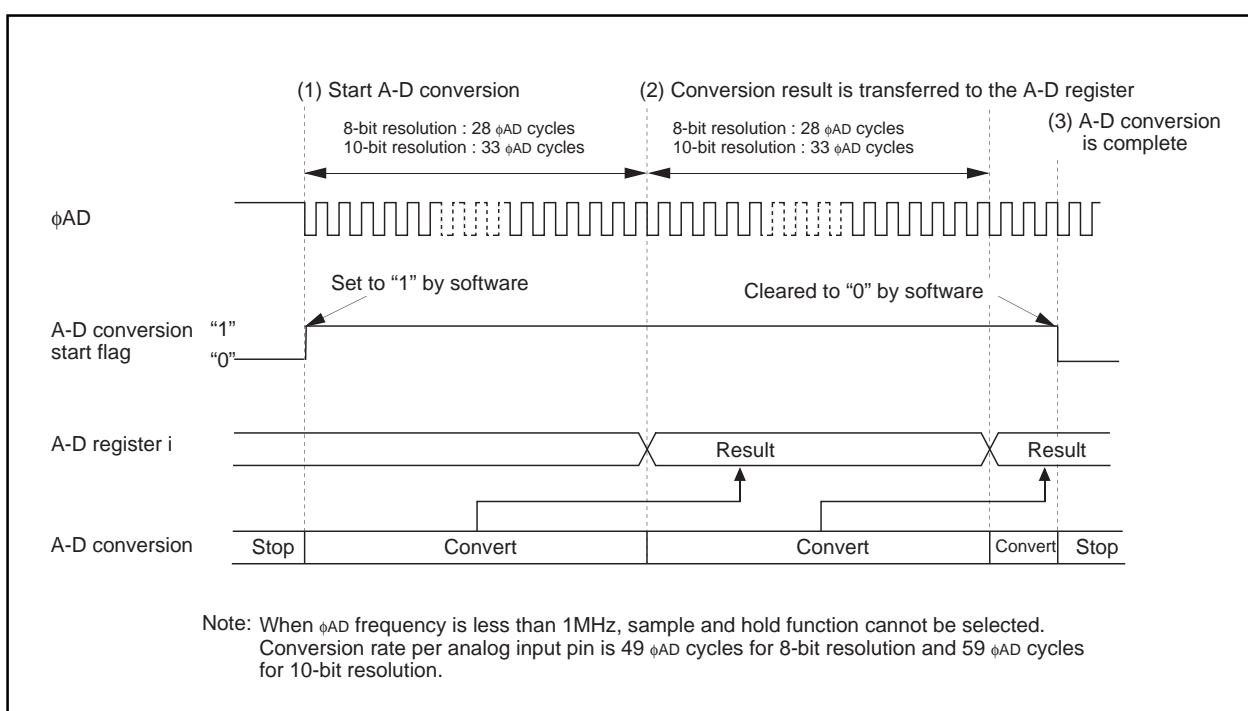
### 2.8.3 Operation of A-D Converter (in repeat mode)

In repeat mode, choose functions from those listed in Table 2.8.3. Operations of the circled items are described below. Figure 2.8.7 shows timing chart, and Figure 2.8.8 shows the set-up procedure.

**Table 2.8.3. Chosed functions**

Item	Set-up	
Operation clock $\phi_{AD}$	○	Divided-by-4 $f_{AD}$ / divided-by-2 $f_{AD}$ / $f_{AD}$
Resolution	○	8-bit / 10-bit
Analog input pin	○	One of AN <sub>0</sub> pin to AN <sub>7</sub> pin
Sample & Hold		Not activated
	○	Activated

- Operation
- (1) Setting the A-D conversion start flag to "1" causes the A-D converter to start operating.
  - (2) After the first conversion is completed, the content of the successive comparison register (conversion result) is transmitted to A-D register i. The A-D conversion interrupt request bit does not go to "1".
  - (3) The A-D converter continues operating until the A-D conversion start flag is set to "0" by software. The conversion result is transmitted to A-D register i every time a conversion is completed.



**Figure 2.8.7. Operation timing of repeat mode**

A-D Converter

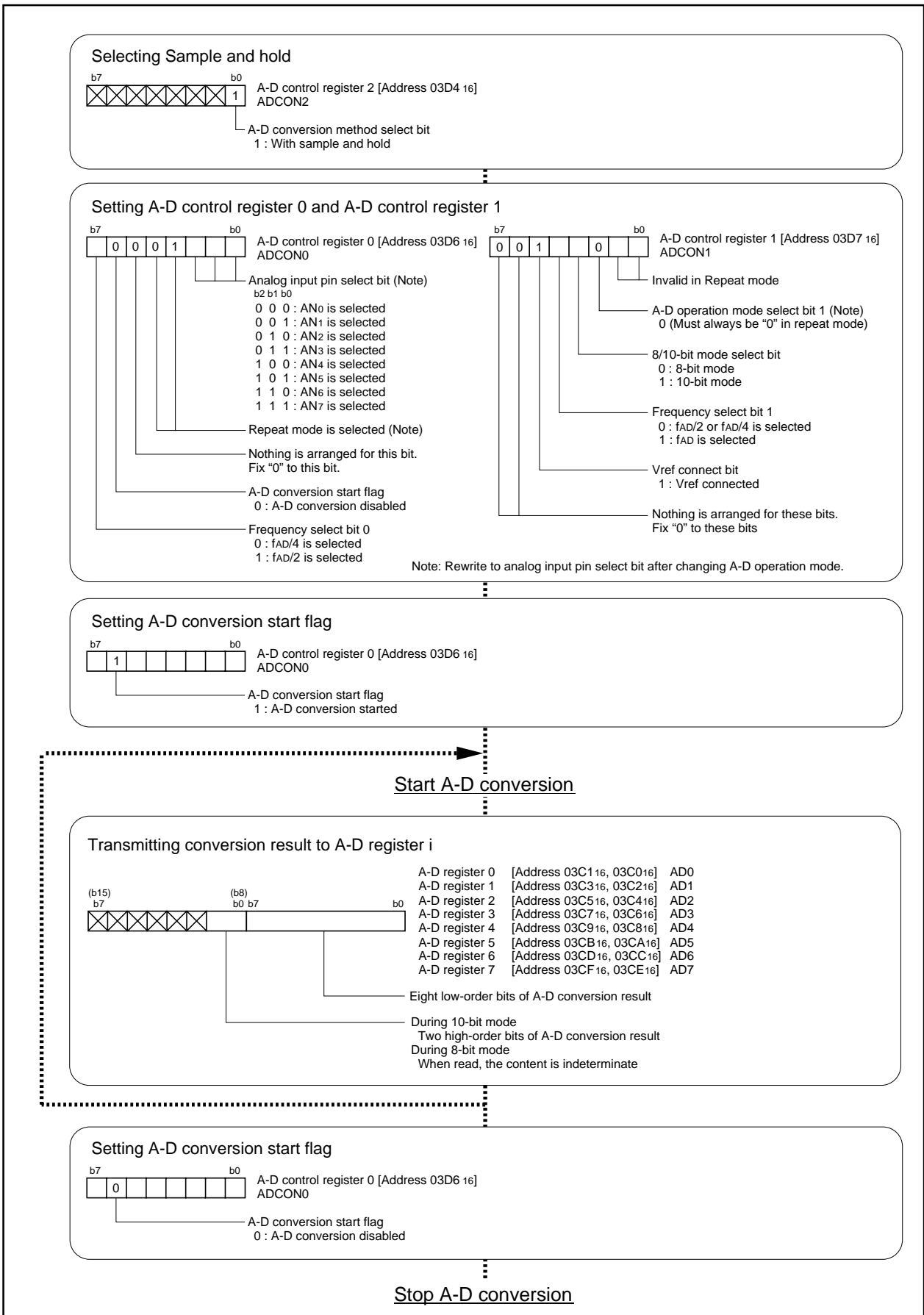


Figure 2.8.8. Set-up procedure of repeat mode

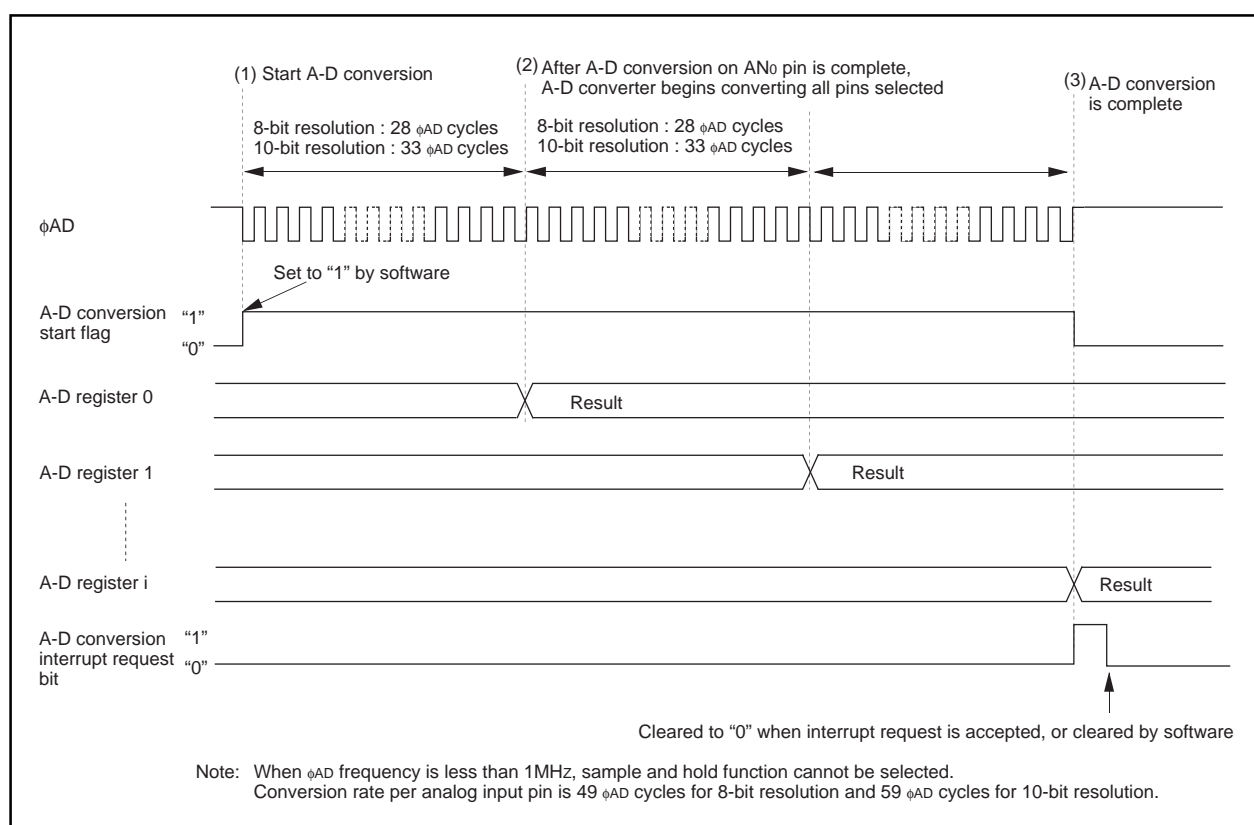
## 2.8.4 Operation of A-D Converter (in single sweep mode)

In single sweep mode, choose functions from those listed in Table 2.8.4. Operations of the circled items are described below. Figure 2.8.9 shows timing chart, and Figure 2.8.10 shows the set-up procedure.

**Table 2.8.4. Chosed functions**

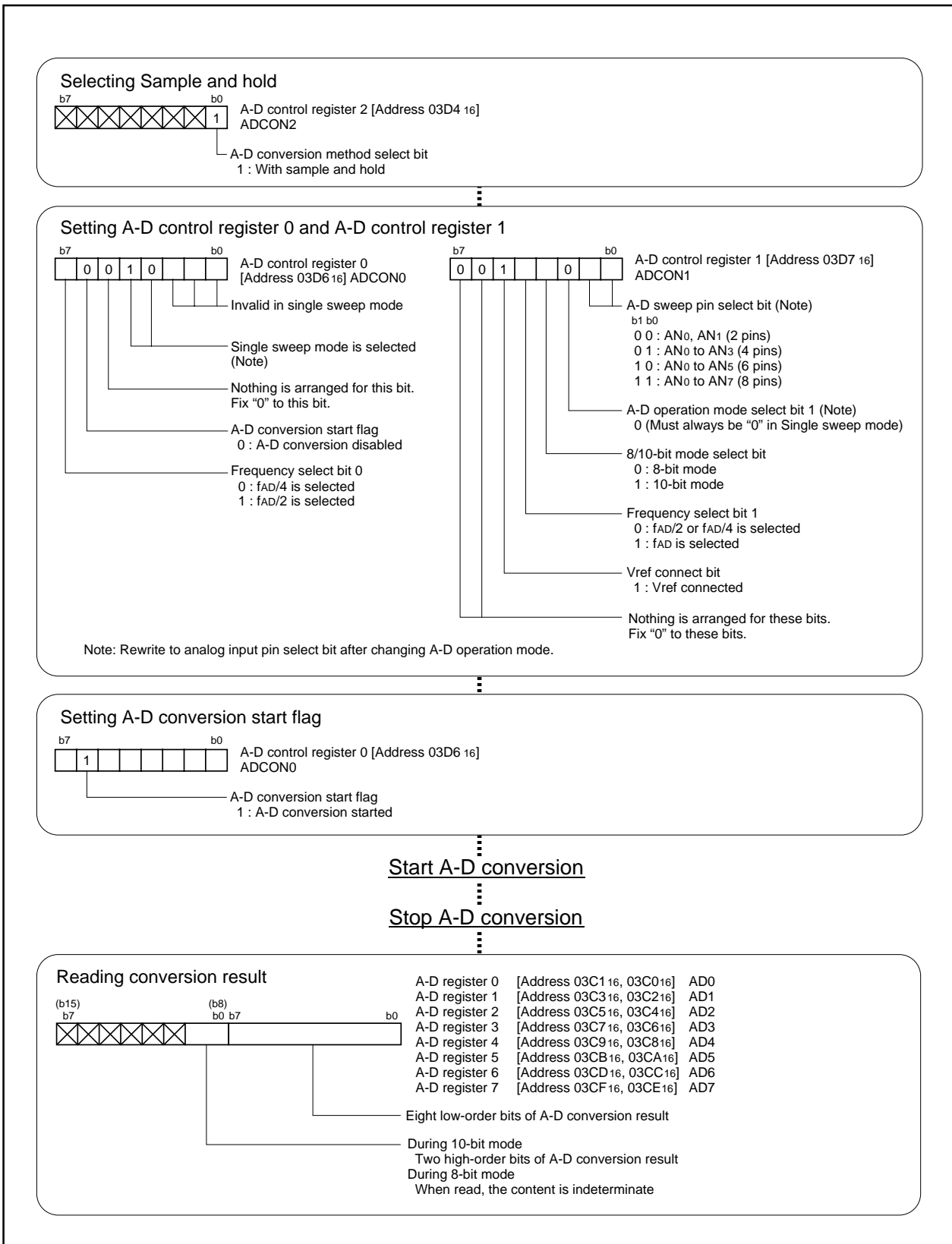
Item	Set-up	Item	Set-up
Operation clock $\phi_{AD}$	○ Divided-by-4 $f_{AD}$ / divided-by-2 $f_{AD}$ / $f_{AD}$	Sample & Hold	Not activated
			○ Activated
Resolution	○ 8-bit / 10-bit		
Analog input pin	○ $AN_0$ and $AN_1$ (2 pins) / $AN_0$ to $AN_3$ (4 pins) / $AN_0$ to $AN_5$ (6 pins) / $AN_0$ to $AN_7$ (8 pins)		

- Operation
- (1) Setting the A-D conversion start flag to "1" causes the A-D converter to start the conversion on voltage input to the  $AN_0$  pin.
  - (2) After the A-D conversion of voltage input to the  $AN_0$  pin is completed, the content of the successive comparison register (conversion result) is transmitted to A-D register 0. The A-D converter converts all analog input pins selected by the user. The conversion result is transmitted to A-D register  $i$  corresponding to each pin, every time conversion on one pin is completed.
  - (3) When the A-D conversion on all the analog input pins selected is completed, the A-D conversion interrupt request bit goes to "1". At this time, the A-D conversion start flag goes to "0". The A-D converter stops operating.



**Figure 2.8.9. Operation timing of single sweep mode**

A-D Converter





## A-D Converter

## 2.8.5 Operation of A-D Converter (in repeat sweep mode 0)

In repeat sweep mode 0, choose functions from those listed in Table 2.8.5. Operations of the circled items are described below. Figure 2.8.11 shows timing chart, and Figure 2.8.12 shows the set-up procedure.

Table 2.8.5. Chosed functions

Item	Set-up		Item	Set-up	
Operation clock $\phi_{AD}$	○	Divided-by-4 $f_{AD}$ / divided-by-2 $f_{AD}$ / $f_{AD}$	Sample & Hold		Not activated
				○	Activated
Resolution	○	8-bit / 10-bit			
Analog input pin	○	AN <sub>0</sub> and AN <sub>1</sub> (2 pins) / AN <sub>0</sub> to AN <sub>3</sub> (4 pins) / AN <sub>0</sub> to AN <sub>5</sub> (6 pins) / AN <sub>0</sub> to AN <sub>7</sub> (8 pins)			

- Operation
- (1) Setting the A-D conversion start flag to "1" causes the A-D converter to start the conversion on voltage input to the AN<sub>0</sub> pin.
  - (2) After the A-D conversion of voltage input to the AN<sub>0</sub> pin is completed, the content of the successive comparison register (conversion result) is transmitted to A-D register 0.
  - (3) The A-D converter converts all pins selected by the user. The conversion result is transmitted to A-D register i corresponding to each pin every time A-D conversion on the pin is completed. The A-D conversion interrupt request bit does not go to "1".
  - (4) The A-D converter continues operating until the A-D conversion start flag is set to "0" by software.

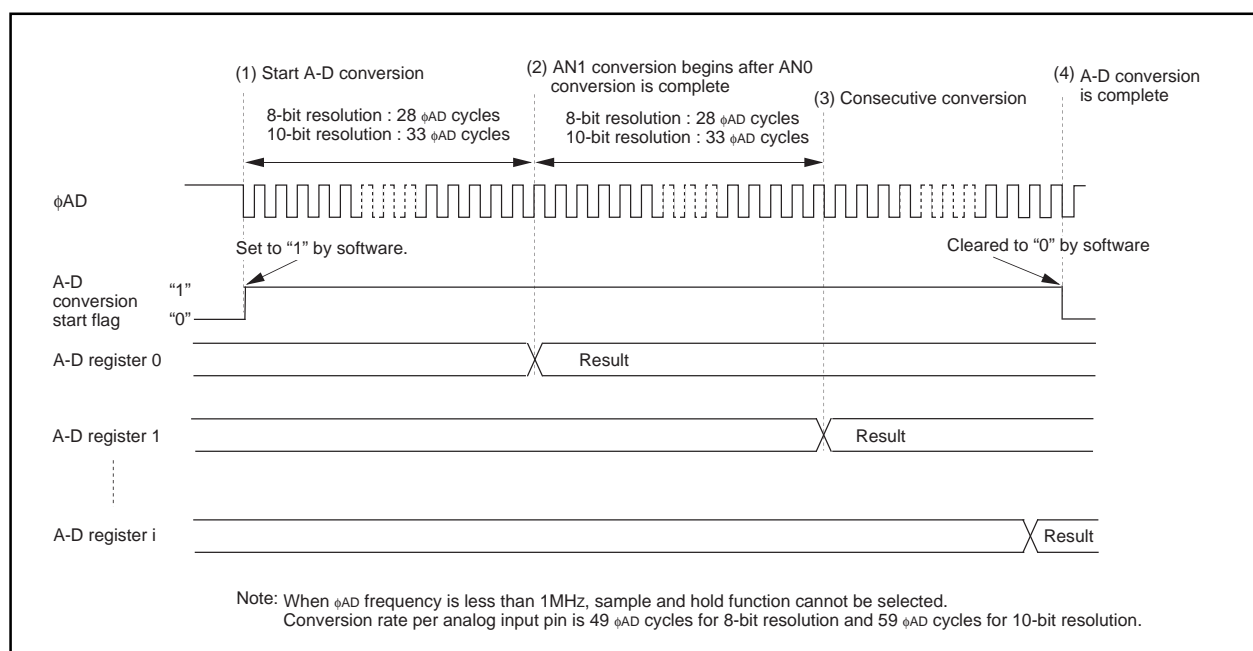


Figure 2.8.11. Operation timing of repeat sweep mode 0

A-D Converter

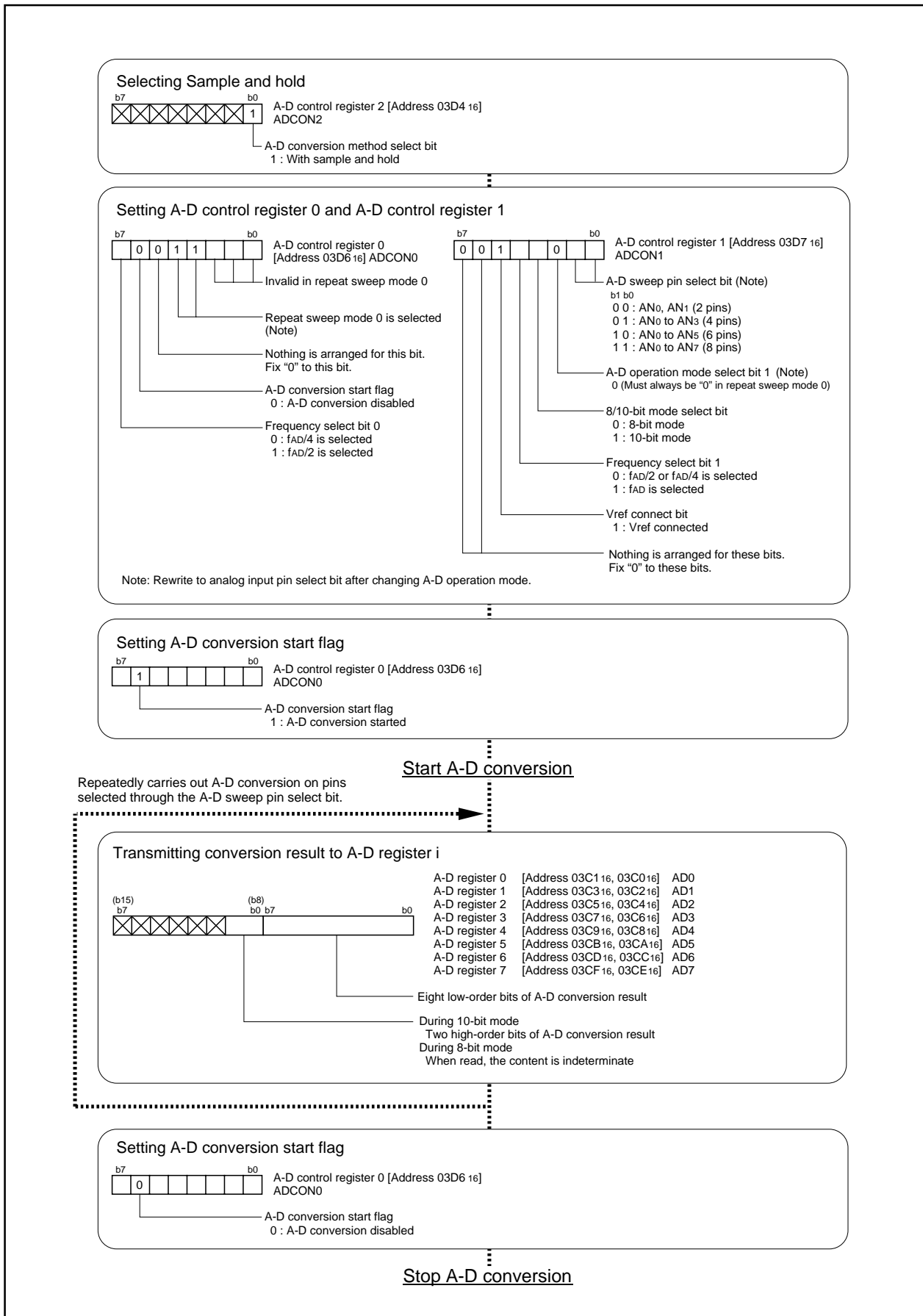


Figure 2.8.12. Set-up procedure of repeat sweep mode 0

## A-D Converter

## 2.8.6 Operation of A-D Converter (in repeat sweep mode 1)

In repeat sweep mode 1, choose functions from those listed in Table 2.8.6. Operations of the circled items are described below. Figure 2.8.13 shows ANi pin's sweep sequence, Figure 2.8.14 shows timing chart, and Figure 2.8.15 shows the set-up procedure.

Table 2.8.6. Chosed functions

Item	Set-up	Item	Set-up
Operation clock $\phi_{AD}$	○ Divided-by-4 $f_{AD}$ / divided-by-2 $f_{AD}$ / $f_{AD}$	Sample & Hold	○ Not activated
			○ Activated
Resolution	○ 8-bit / 10-bit		
Analog input pin	○ An0 (1 pin) / AN0 and AN1 (2 pins) / AN0 to AN2 (3 pins) / AN0 to AN3 (4 pins)		

- Operation
- (1) Setting the A-D conversion start flag to "1" causes the A-D converter to start the conversion on voltage input to the AN0 pin.
  - (2) After the A-D conversion on voltage input to the AN0 pin is completed, the content of the successive comparison register (conversion result) is transmitted to A-D register 0.
  - (3) Every time the A-D converter carries out A-D conversion on a selected analog input pin, the A-D converter carries out A-D conversion on only one unselected pin, and then the A-D converter carries out A-D conversion from the AN0 pin again. (See Figure 2.8.13.) The conversion result is transmitted to A-D register i every time conversion on a pin is completed. The A-D conversion interrupt request bit does not go to "1".
  - (4) The A-D converter continues operating until software goes the A-D conversion start flag to "0".

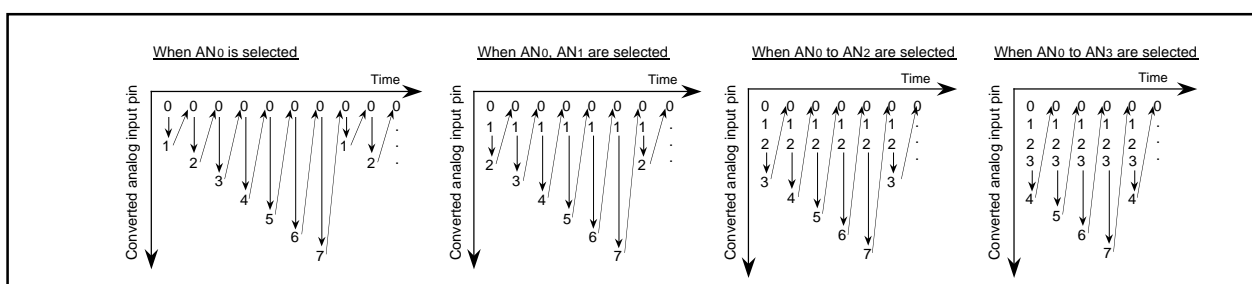


Figure 2.8.13. ANi pin's sweep sequence in repeat sweep mode 1

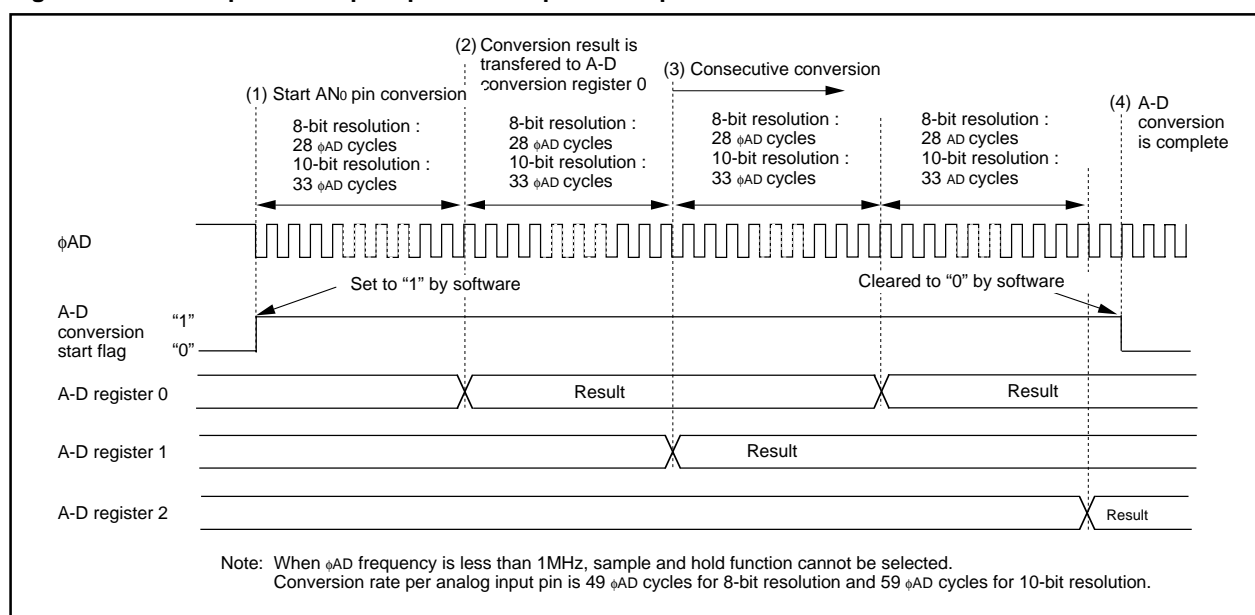


Figure 2.8.14. Operation timing of repeat sweep mode 1

A-D Converter

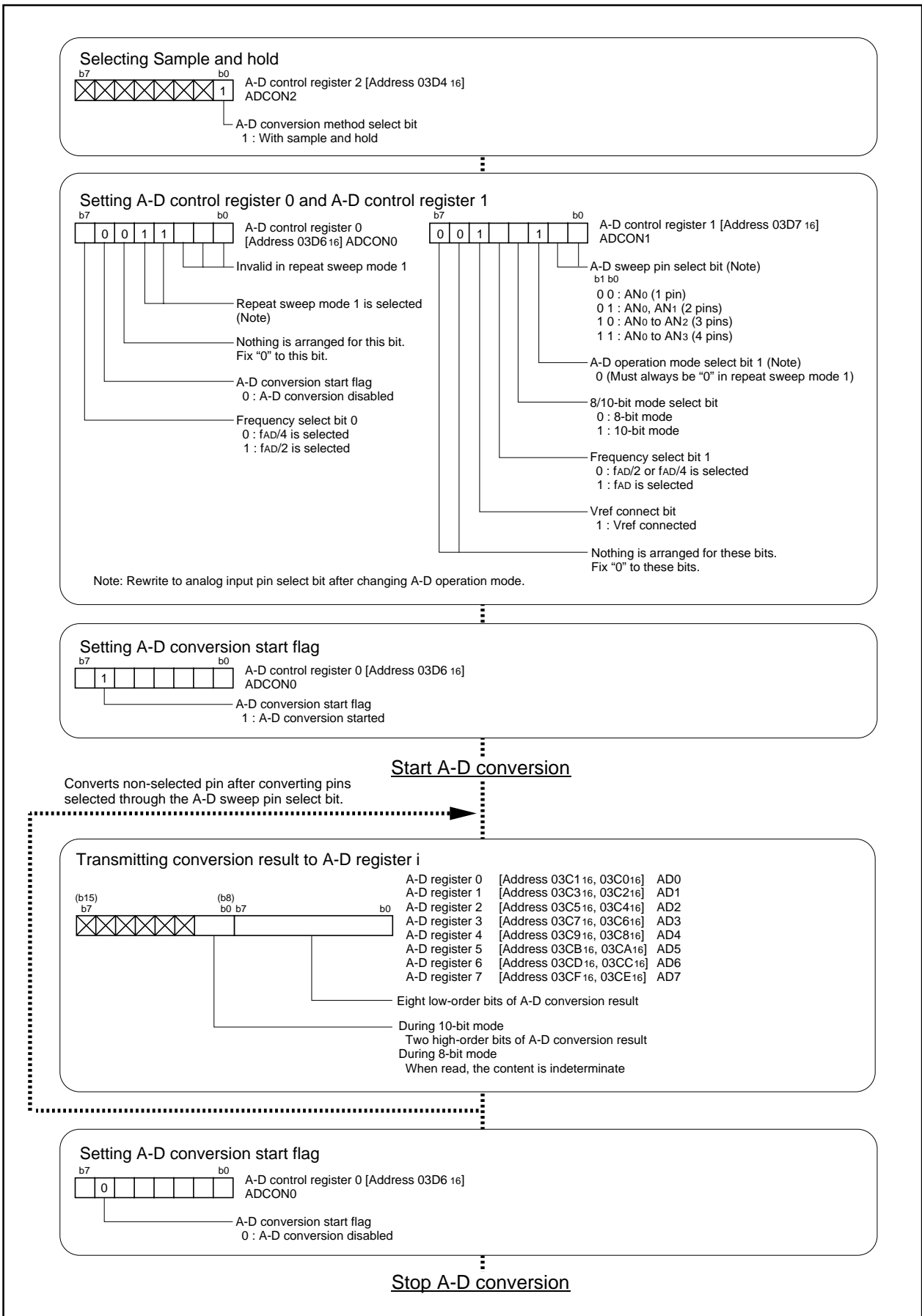


Figure 2.8.15. Set-up procedure of repeat sweep mode 1

### 2.8.7 Precautions for 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.  
In particular, when the Vref connection bit is changed from 0 to 1, start A-D conversion after an elapse of 1  $\mu$ s or longer.
- (2) To reduce conversion error due to noise, connect a voltage to the AVcc pin and to the VREF pin from an independent source. It is recommended to connect a capacitor between the AVss pin and the AVcc pin, between the AVss pin and the VREF pin, and between the AVss pin and the analog input pin (ANi). Figure 2.8.16 shows an example of connecting the capacitors to these pins.

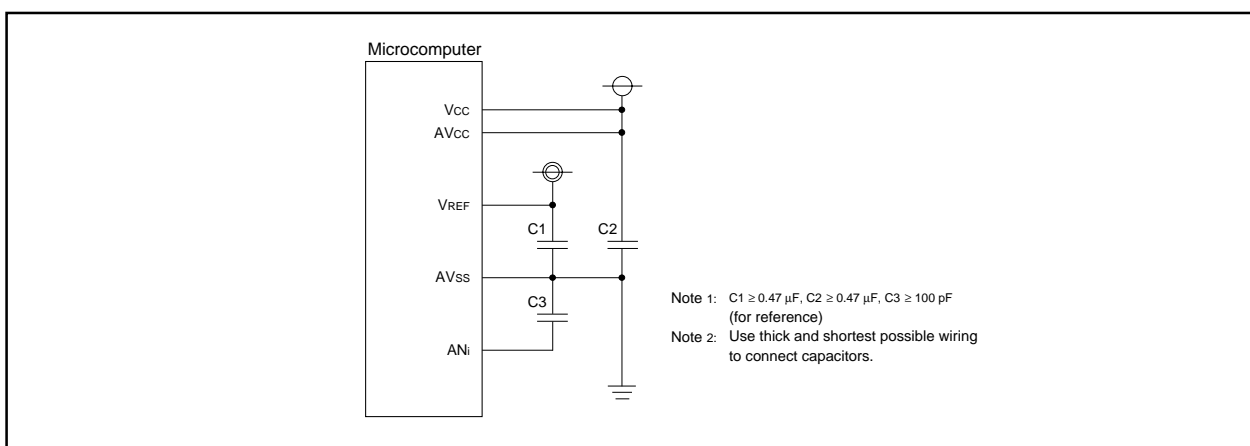


Figure 2.8.16. Use of capacitors to reduce noise

- (3) Set the direction register of the the port corresponding to a pin to be used as an analog input pin to input.
- (4) Rewrite to analog input pin after changing A-D operation mode. The two cannot be set at the same time.
- (5) When using the one-shot or single sweep mode  
Confirm that A-D conversion is complete before reading the A-D register.  
(Note: When A-D conversion interrupt request bit is set, it shows that A-D conversion is completed.)
- (6) When using the repeat mode or repeat sweep mode 0 or 1  
Use the undivided main clock as the internal CPU clock.

### 2.8.8 Method of A-D Conversion (10-bit mode)

(1) The A-D converter compares the reference voltage (Vref) generated internally based on the contents of the successive comparison register with the analog input voltage (VIN) input from the analog input pin. Each bit of the comparison result is stored in the successive comparison register until analog-to-digital conversion (successive comparison method) is complete. If a trigger occurs, the A-D converter carries out the following:

1. Fixes bit 9 of the successive comparison register.

Compares Vref with VIN: [In this instance, the contents of the successive comparison register are "1000000002" (default).]

Bit 9 of the successive comparison register varies depending on the comparison result as follows.

If  $V_{ref} < V_{IN}$ , then "1" is assigned to bit 9.

If  $V_{ref} > V_{IN}$ , then "0" is assigned to bit 9.

2. Fixes bit 8 of the successive comparison register.

Sets bit 8 of the successive comparison register to "1", then compares Vref with VIN.

Bit 8 of the successive comparison register varies depending on the comparison result as follows:

If  $V_{ref} < V_{IN}$ , then "1" is assigned to bit 8.

If  $V_{ref} > V_{IN}$ , then "0" is assigned to bit 8.

3. Fixes bit 7 through bit 0 of the successive comparison register.

Carries out step 2 above on bit 7 through bit 0.

After bit 0 is fixed, the contents of the successive comparison register (conversion result) are transmitted to A-D register i.

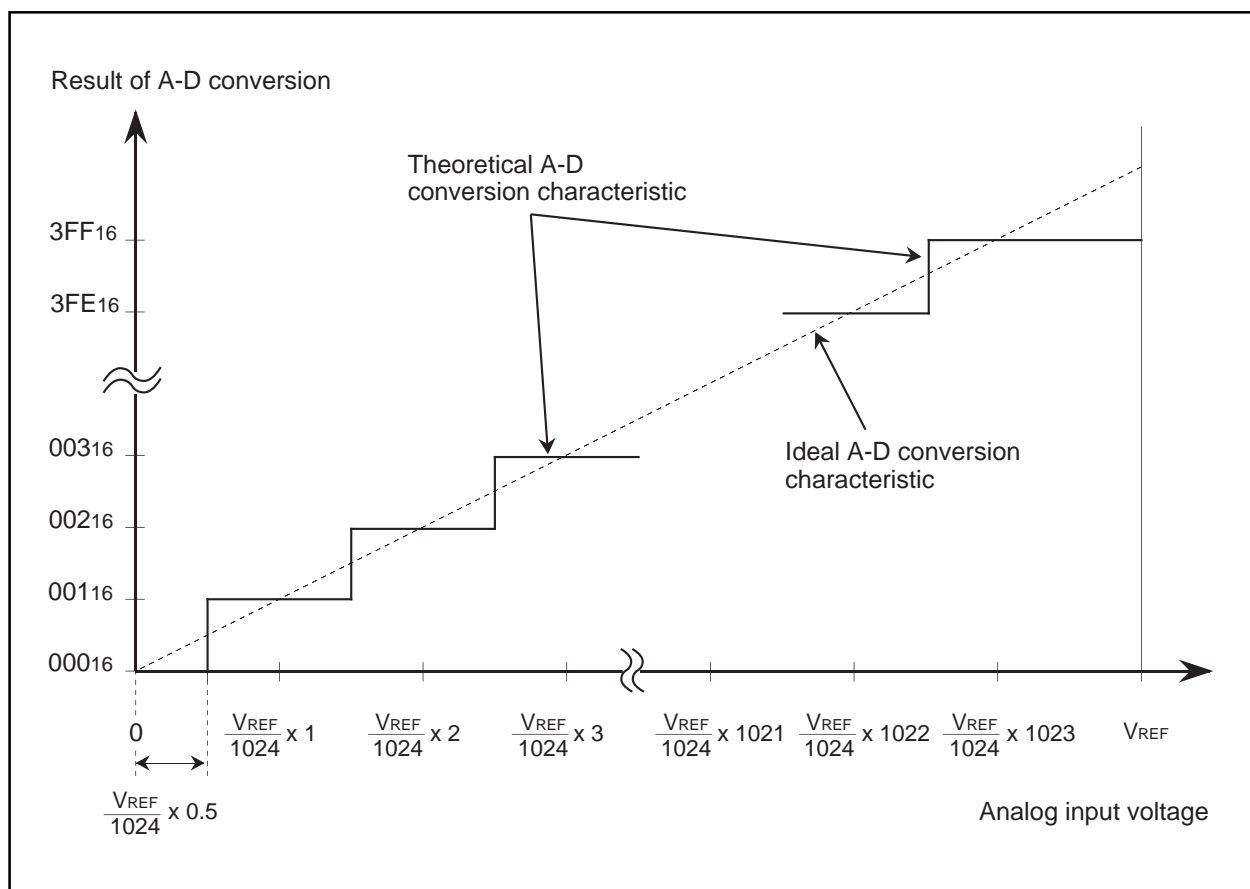
Vref is generated based on the latest content of the successive comparison register. Table 2.8.7 shows the relationship of the successive comparison register contents and Vref. Table 2.8.8 shows how the successive comparison register and Vref vary while A-D conversion is in progress. Figure 2.8.17 shows theoretical A-D conversion characteristics.

**Table 2.8.7. Relationship of the successive comparison register contents and Vref**

Successive approximation register : n	Vref (V)
0	0
1 to 1023	$\frac{V_{REF}}{1024} \times n - \frac{V_{REF}}{2048}$

**Table 2.8.8. Variation of the successive comparison register and Vref while A-D conversion is in progress (10-bit mode)**

	Successive approximation register	Vref change																						
A-D converter stopped	<table border="1" style="width: 100%; text-align: center;"> <tr> <td>b9</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>b0</td> </tr> <tr> <td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td> </tr> </table>	b9										b0	1	0	0	0	0	0	0	0	0	0	0	$\frac{V_{REF}}{2}$ [V]
b9										b0														
1	0	0	0	0	0	0	0	0	0	0														
1st comparison	<table border="1" style="width: 100%; text-align: center;"> <tr> <td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td> </tr> </table>	1	0	0	0	0	0	0	0	0	0	0	$\frac{V_{REF}}{2} - \frac{V_{REF}}{2048}$ [V]											
1	0	0	0	0	0	0	0	0	0	0														
2nd comparison	<table border="1" style="width: 100%; text-align: center;"> <tr> <td>n9</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td> </tr> </table> <p style="text-align: center;">↑ 1st comparison result</p>	n9	1	0	0	0	0	0	0	0	0	0	$\frac{V_{REF}}{2} \pm \frac{V_{REF}}{4} - \frac{V_{REF}}{2048}$ [V] $\left( \begin{matrix} n_9 = 1 & + & \frac{V_{REF}}{4} \\ n_9 = 0 & - & \frac{V_{REF}}{4} \end{matrix} \right)$											
n9	1	0	0	0	0	0	0	0	0	0														
3rd comparison	<table border="1" style="width: 100%; text-align: center;"> <tr> <td>n9</td><td>n8</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td> </tr> </table> <p style="text-align: center;">↑ 2nd comparison result</p>	n9	n8	1	0	0	0	0	0	0	0	0	$\frac{V_{REF}}{2} \pm \frac{V_{REF}}{4} \pm \frac{V_{REF}}{8} - \frac{V_{REF}}{2048}$ [V] $\left( \begin{matrix} n_8 = 1 & + & \frac{V_{REF}}{8} \\ n_8 = 0 & - & \frac{V_{REF}}{8} \end{matrix} \right)$											
n9	n8	1	0	0	0	0	0	0	0	0														
⋮	⋮	⋮																						
10th comparison	<table border="1" style="width: 100%; text-align: center;"> <tr> <td>n9</td><td>n8</td><td>n7</td><td>n6</td><td>n5</td><td>n4</td><td>n3</td><td>n2</td><td>n1</td><td>0</td> </tr> </table>	n9	n8	n7	n6	n5	n4	n3	n2	n1	0	$\frac{V_{REF}}{2} \pm \frac{V_{REF}}{4} \pm \frac{V_{REF}}{8} \pm \dots \pm \frac{V_{REF}}{1024} - \frac{V_{REF}}{2048}$ [V]												
n9	n8	n7	n6	n5	n4	n3	n2	n1	0															
Conversion complete	<table border="1" style="width: 100%; text-align: center;"> <tr> <td>n9</td><td>n8</td><td>n7</td><td>n6</td><td>n5</td><td>n4</td><td>n3</td><td>n2</td><td>n1</td><td>n0</td> </tr> </table> <p style="text-align: center;">This data transfers to the bit 0 to bit 9 of A-D register.</p>	n9	n8	n7	n6	n5	n4	n3	n2	n1	n0													
n9	n8	n7	n6	n5	n4	n3	n2	n1	n0															



**Figure 2.8.17. Theoretical A-D conversion characteristics (10-bit mode)**

## A-D Converter

## 2.8.9 Method of A-D Conversion (8-bit mode)

- (1) In 8-bit mode, 8 higher-order bits of the 10-bit successive comparison register becomes A-D conversion result. Hence, if compared to a result obtained by using an 8-bit A-D converter, the voltage compared is different by  $3 V_{REF}/2048$  (see what are underscored in Table 2.8.9), and differences in stepping points of output codes occur as shown in Figure 2.8.18.

Table 2.8.9. The comparison voltage in 8-bit mode compared to 8-bit A-D converter

		8-bit mode	8-bit A-D converter
Comparison voltage $V_{REF}$	$n = 0$	0	0
	$n = 1$ to 255	$\frac{V_{REF}}{2^8} \times n - \frac{V_{REF}}{2^{10}} \times 0.5$	$\frac{V_{REF}}{2^8} \times n - \frac{V_{REF}}{2^8} \times 0.5$

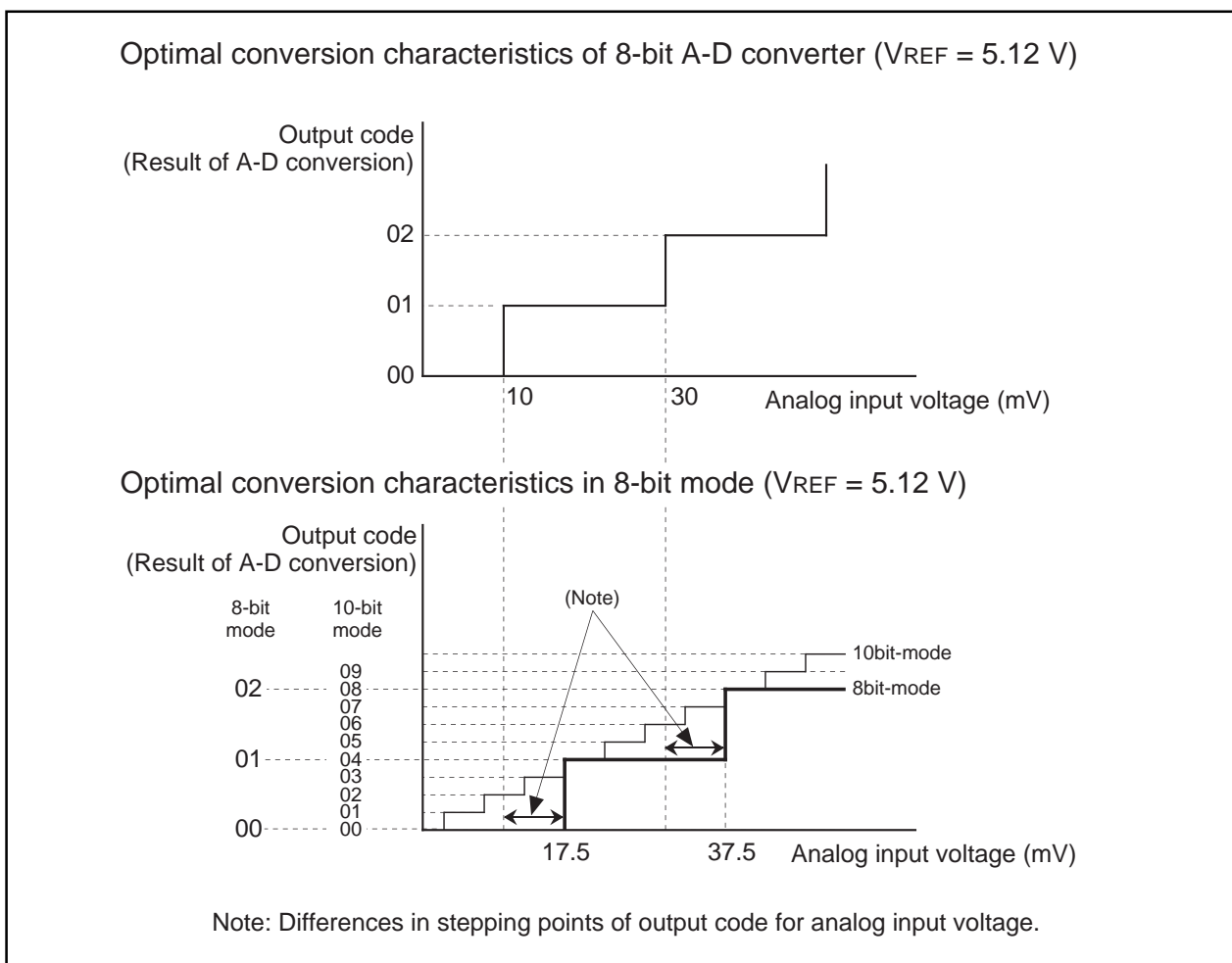
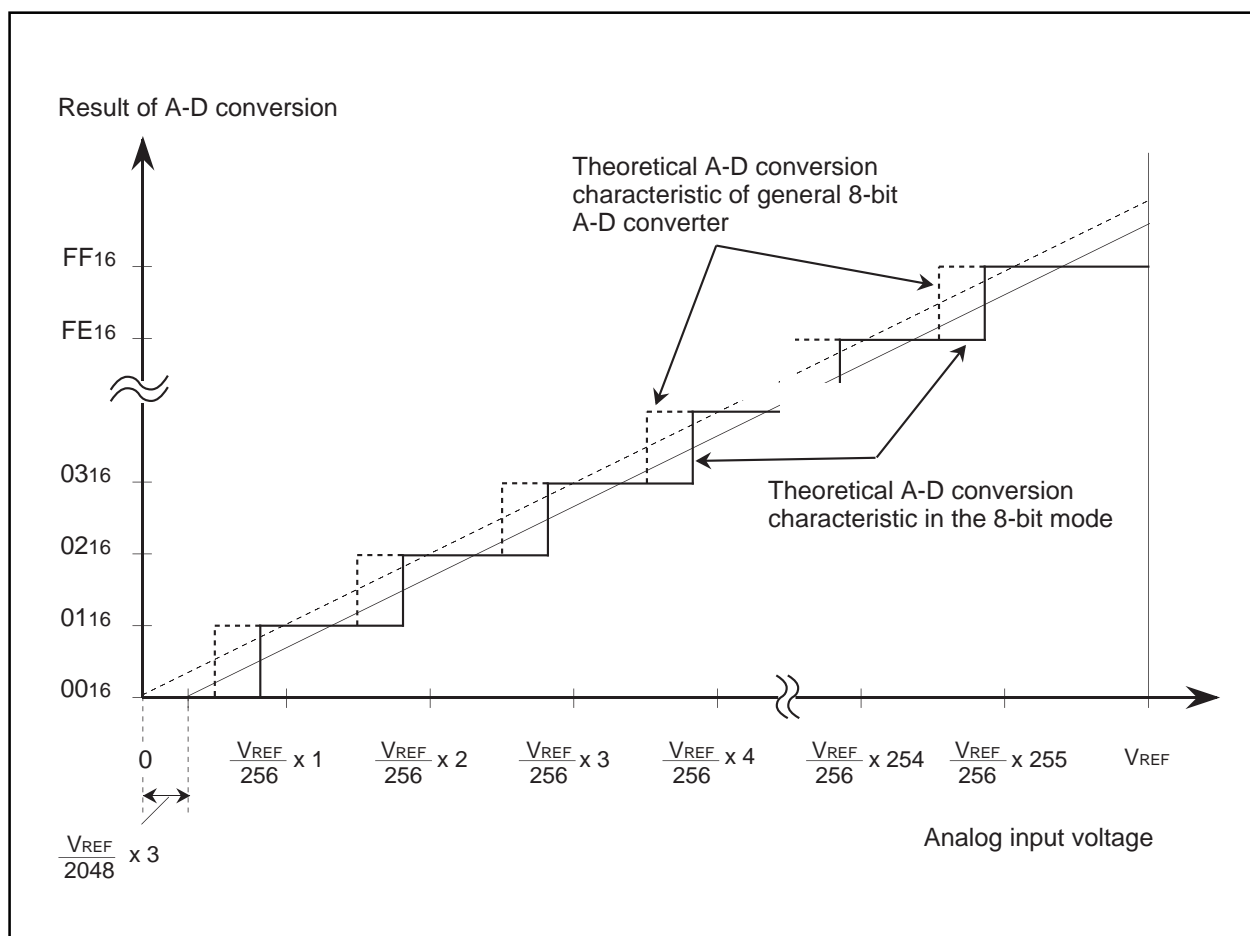


Figure 2.8.18. The level conversion characteristics of 8-bit mode and 8-bit A-D converter



**Table 2.8.10. Variation of the successive comparison register and Vref while A-D conversion is in progress (8-bit mode)**

	Successive approximation register	Vref change
A-D converter stopped	b9 1 0 0 0 0 0 0 0 0 0 b0	$\frac{V_{REF}}{2}$ [V]
1st comparison	1 0 0 0 0 0 0 0 0 0	$\frac{V_{REF}}{2} - \frac{V_{REF}}{2048}$ [V]
2nd comparison	n9 1 0 0 0 0 0 0 0 0	$\frac{V_{REF}}{2} \pm \frac{V_{REF}}{4} - \frac{V_{REF}}{2048}$ [V] $\begin{cases} n_9 = 1 & + \frac{V_{REF}}{4} \\ n_9 = 0 & - \frac{V_{REF}}{4} \end{cases}$
3rd comparison	n9 n8 1 0 0 0 0 0 0 0 ↑ 1st comparison result	
8th comparison	n9 n8 n7 n6 n5 n4 n3 1 0 0 ↑ 2nd comparison result	$\frac{V_{REF}}{2} \pm \frac{V_{REF}}{4} \pm \frac{V_{REF}}{8} - \frac{V_{REF}}{2048}$ [V] $\begin{cases} n_8 = 1 & + \frac{V_{REF}}{8} \\ n_8 = 0 & - \frac{V_{REF}}{8} \end{cases}$
Conversion complete	n9 n8 n7 n6 n5 n4 n3 n2 0 0 This data transfers to bit 0 to bit 7 of A-D register.	



**Figure 2.8.19. Theoretical A-D conversion characteristics (8-bit mode)**

## 2.8.10 Absolute Accuracy and Differential Non-Linearity Error

### • Absolute accuracy

Absolute accuracy is the difference between output code based on the theoretical A-D conversion characteristics, and actual A-D conversion result. When measuring absolute accuracy, the voltage at the middle point of the width of analog input voltage (1-LSB width), that can meet the expectation of outputting an equal code based on the theoretical A-D conversion characteristics, is used as an analog input voltage. For example, if 10-bit resolution is used and if  $V_{REF}$  (reference voltage) = 5.12 V, then 1-LSB width becomes 5 mV, and 0 mV, 5 mV, 10 mV, 15 mV, 20 mV, ... are used as analog input voltages. If analog input voltage is 25 mV, "absolute accuracy =  $\pm 3\text{LSB}$ " refers to the fact that actual A-D conversion falls on a range from "002<sub>16</sub>" to "008<sub>16</sub>" though an output code, "005<sub>16</sub>", can be expected from the theoretical A-D conversion characteristics. Zero error and full-scale error are included in absolute accuracy.

Also, all the output codes for analog input voltage between  $V_{REF}$  and  $V_{CC}$  becomes "3FF<sub>16</sub>".

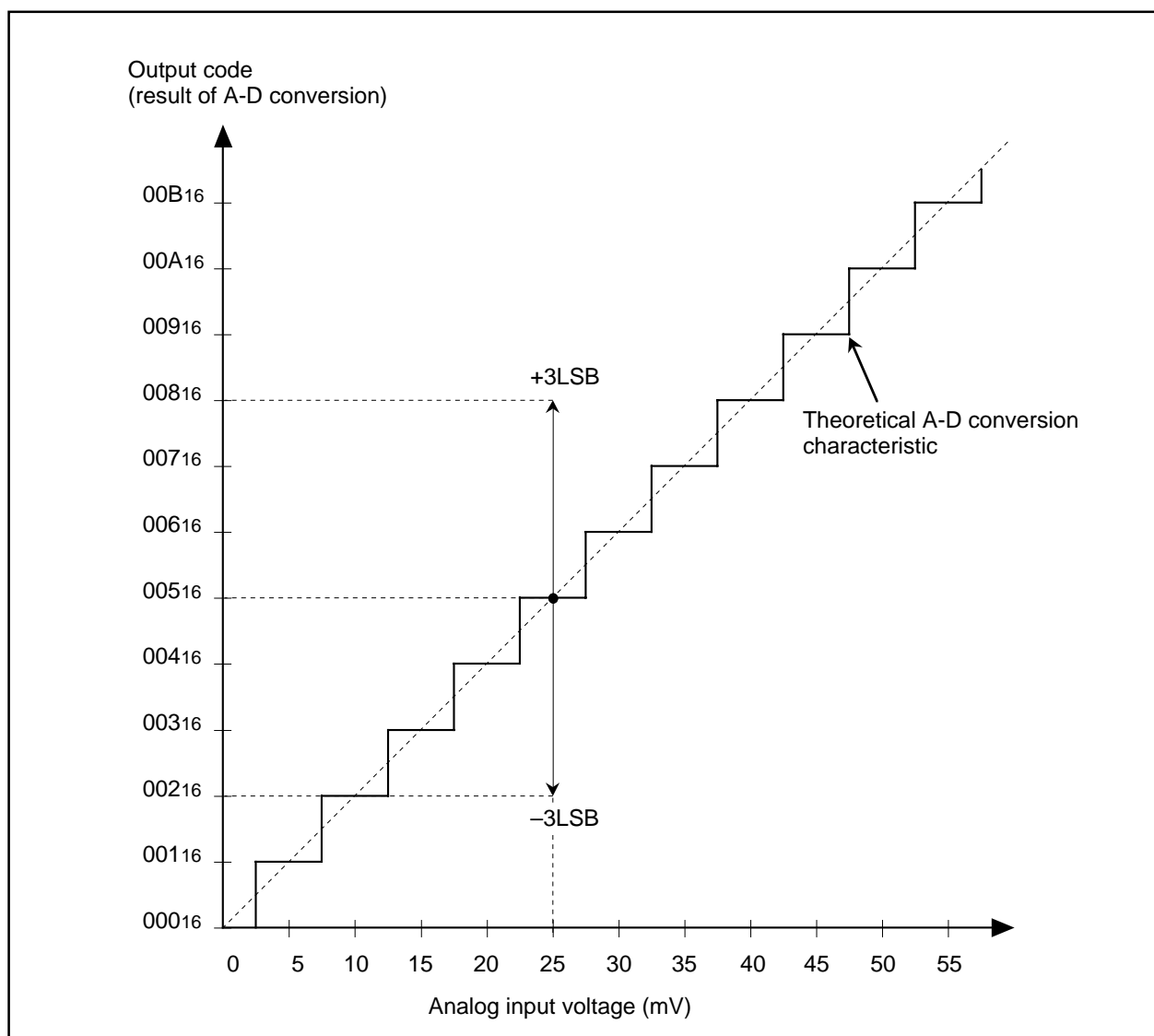


Figure 2.8.20. Absolute accuracy (10-bit resolution)

## A-D Converter

- **Differential non-linearity error**

Differential non-linearity error refers to the difference between 1-LSB width based on the theoretical A-D conversion characteristics (an analog input width that can meet the expectation of outputting an equal code) and an actually measured 1-LSB width (analog input voltage width that outputs an equal code). If 10-bit resolution is used and if  $V_{REF}$  (reference voltage) = 5.12 V, "differential non-linearity error =  $\pm 1$ LSB" refers to the fact that 1-LSB width actually measured falls on a range from 0 mV to 10 mV though 1-LSB width based on the theoretical A-D conversion characteristics is 5 mV (see 5.2 A-D converter's standard characteristics).

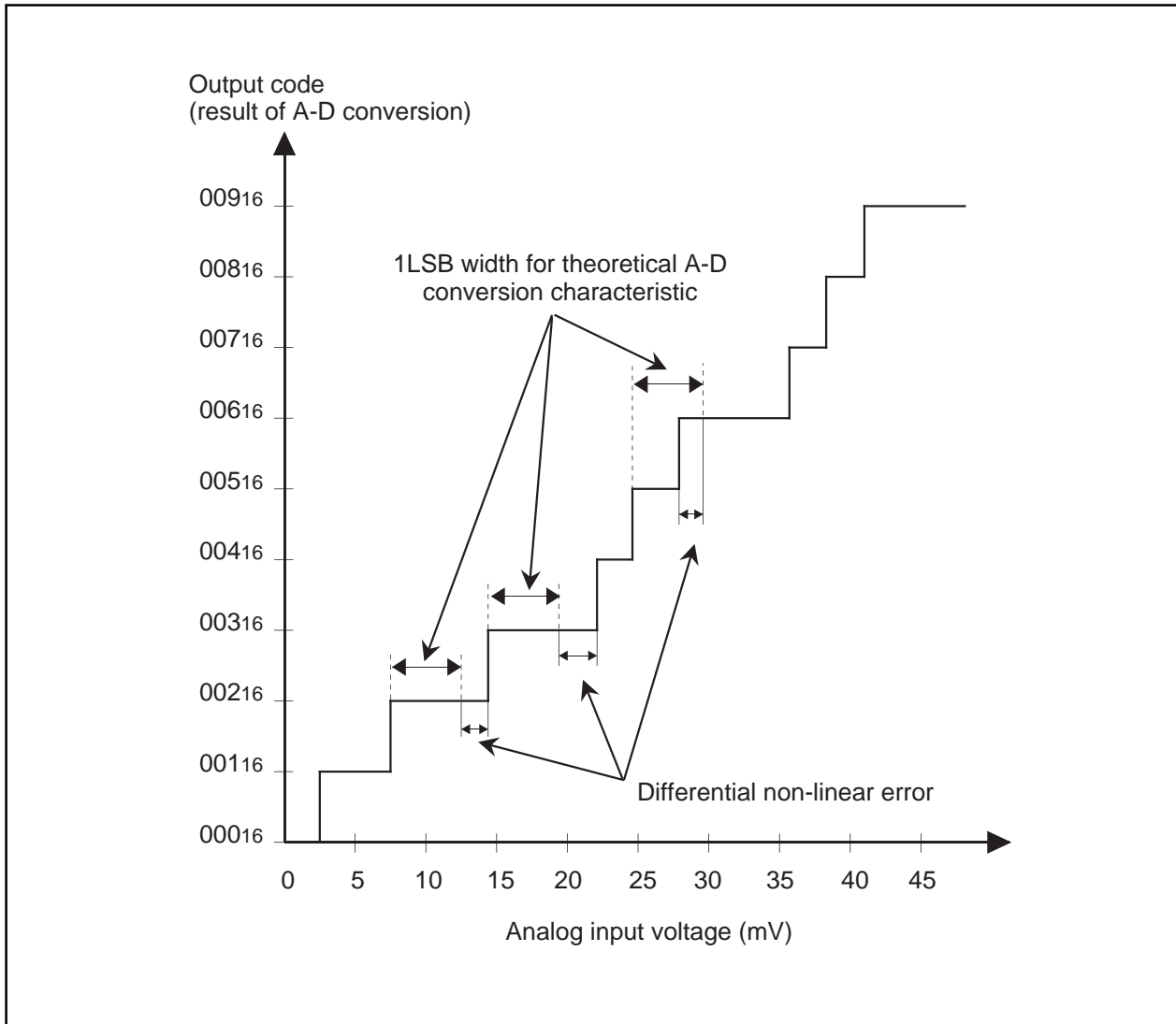


Figure 2.8.21. Differential non-linearity error (10-bit resolution)

### 2.8.11 Internal Equivalent Circuit of Analog Input

Figure 2.8.22 shows the internal equivalent circuit of analog input.

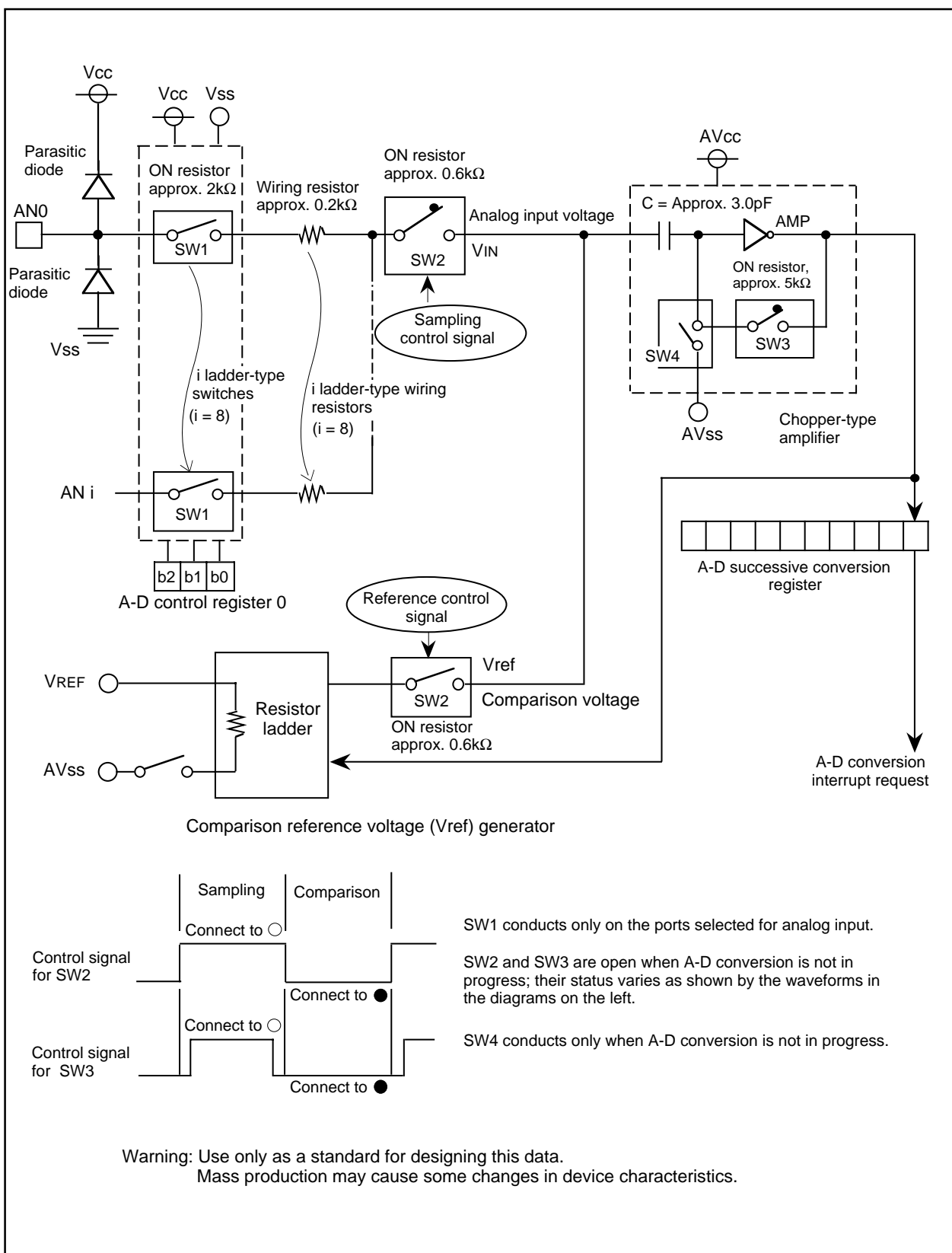


Figure 2.8.22. Internal equivalent circuit to analog input

### 2.8.12 Sensor's Output Impedance under A-D Conversion

To carry out A-D conversion properly, charging the internal capacitor C shown in Figure 2.8.23 has to be completed within a specified period of time. With T as the specified time, time T is the time that switches SW2 and SW3 are connected to O in Figure 2.8.22. Let output impedance of sensor equivalent circuit be R0, microcomputer's internal resistance be R, precision (error) of the A-D converter be X, and the A-D converter's resolution be Y.

$$V_c \text{ is generally } V_c = V_{IN} \left\{ 1 - e^{-\frac{t}{C(R_0 + R)}} \right\}$$

$$\text{And when } t = T, \quad V_c = V_{IN} - \frac{X}{Y} V_{IN} = V_{IN} \left( 1 - \frac{X}{Y} \right)$$

$$e^{-\frac{T}{C(R_0 + R)}} = \frac{X}{Y}$$

$$-\frac{T}{C(R_0 + R)} = \ln \frac{X}{Y}$$

$$\text{Hence, } R_0 = -\frac{T}{C \cdot \ln \frac{X}{Y}} - R$$

Each value is R = 7.8 kΩ, C = 3 pF, T = 0.3 μs in the A-D conversion mode with sample & hold. For example, when the A-D converter's resolution is 10 bits and precision (error) of the A-D converter is 0.1 LSB, Y = 10, X = 0.1 LSB. Hence,

$$R_0 = -\frac{0.3 \times 10^{-6}}{3.0 \times 10^{-12} \cdot \ln \frac{0.1}{1024}} - 7.8 \times 10^3 \approx 3.0 \times 10^3$$

Thus, the allowable output impedance of the sensor circuit capable of thoroughly driving the A-D converter turns out to be approximately 3.0 kΩ. Tables 2.8.11 and 2.8.12 show output impedance values based on the LSB values.

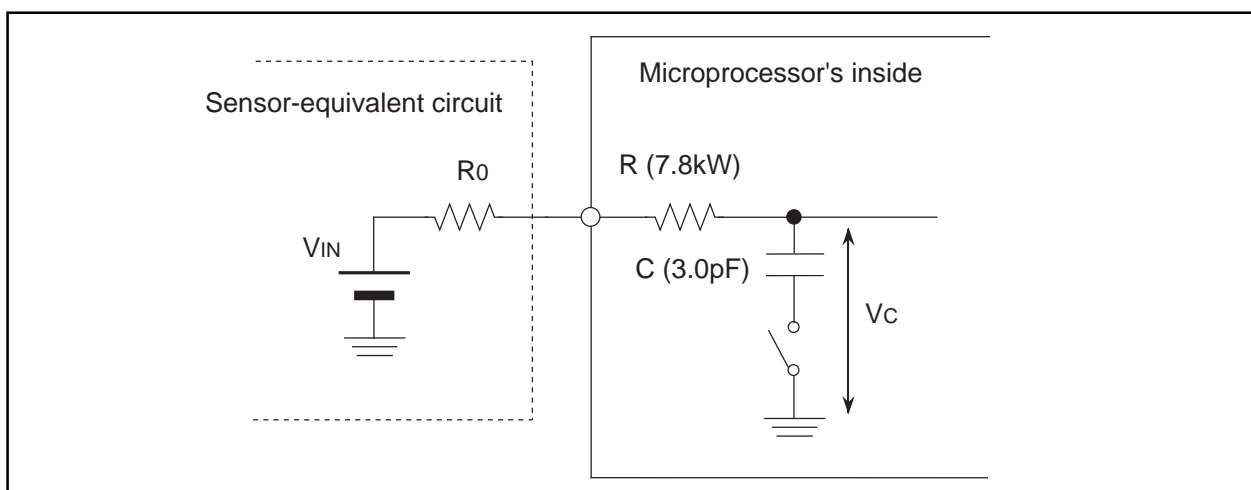


Figure 2.8.23 A circuit equivalent to the A-D conversion terminal

## A-D Converter

**Table 2.8.11. Output impedance values based on the LSB values (1)**

f(XIN) (MHz)	Cycle ( $\mu$ s)	Sampling time ( $\mu$ s)	R (kohm)	C (pF)	Accuracy (LSB)	R0 (kohm)
10	0.1	0.3 (3 X cycle, sample & hold bit is enabled)	7.8	3.0	0.1	3.0
					0.3	4.5
					0.5	5.3
					0.7	5.9
					0.9	6.4
					1.1	6.8
					1.3	7.2
					1.5	7.5
					1.7	7.8
1.9	8.1					
10	0.1	0.2 (2 X cycle, sample & hold bit is disabled)	7.8	3.0	0.3	0.4
					0.5	0.9
					0.7	1.3
					0.9	1.7
					1.1	2.0
					1.3	2.2
					1.5	2.4
					1.7	2.6
					1.9	2.8

**Table 2.8.12. Output impedance values based on the LSB values (2)**

f(XIN) (MHz)	Cycle ( $\mu$ s)	Sampling time ( $\mu$ s)	R (kohm)	C (pF)	Accuracy (LSB)	R0 (kohm)
10	0.1	0.3 (3 X cycle, sample & hold bit is enabled)	7.8	3.0	0.1	4.9
					0.3	7.0
					0.5	8.2
					0.7	9.1
					0.9	9.9
					1.1	10.5
					1.3	11.1
					1.5	11.7
					1.7	12.1
1.9	12.6					
10	0.1	0.2 (2 X cycle, sample & hold bit is disabled)	7.8	3.0	0.1	0.7
					0.3	2.1
					0.5	2.9
					0.7	3.5
					0.9	4.0
					1.1	4.4
					1.3	4.8
					1.5	5.2
					1.7	5.5
1.9	5.8					

## D-A Converter

## 2.9 D-A Converter

## 2.9.1 Overview

The D-A converter used in the M30218 group is based on the 8-bit R-2R technique.

## (1) Output voltage

The D-A converter outputs voltage within a range from 0 V to VREF. The output voltage is determined by  $V_{REF}/(256) \times$  the D-A register contents.

The D-A converter is not effected by the Vref connection bit of the A-D converter.

## (2) Conversion time

$t_{su} = 3 \mu s$

## (3) Output from the D-A converter and the direction register

To use the D-A converter, do not set the direction register of the relevant port to output.

## (4) Pins related to the D-A converter

- DA0 pin, DA1 pin     Output pins of the D-A converter
- AVcc pin             The power source pin of the analog section
- VREF pin             Input pin of the reference voltage
- AVss pin             The GND pin of the analog section

## (5) Registers related to the D-A converter

Figure 2.9.1 shows the memory map of D-A converter-related registers, and Figure 2.9.2 shows D-A converter-related registers.

## (6) Note

D-A output pins shared with P97 and P96. The two pins are input ports and floating at the reset.

03D8 <sub>16</sub>	D-A register 0 (DA0)
03D9 <sub>16</sub>	
03DA <sub>16</sub>	D-A register 1 (DA1)
03DB <sub>16</sub>	
03DC <sub>16</sub>	D-A control register (DACON)

Figure 2.9.1. Memory map of D-A converter-related registers

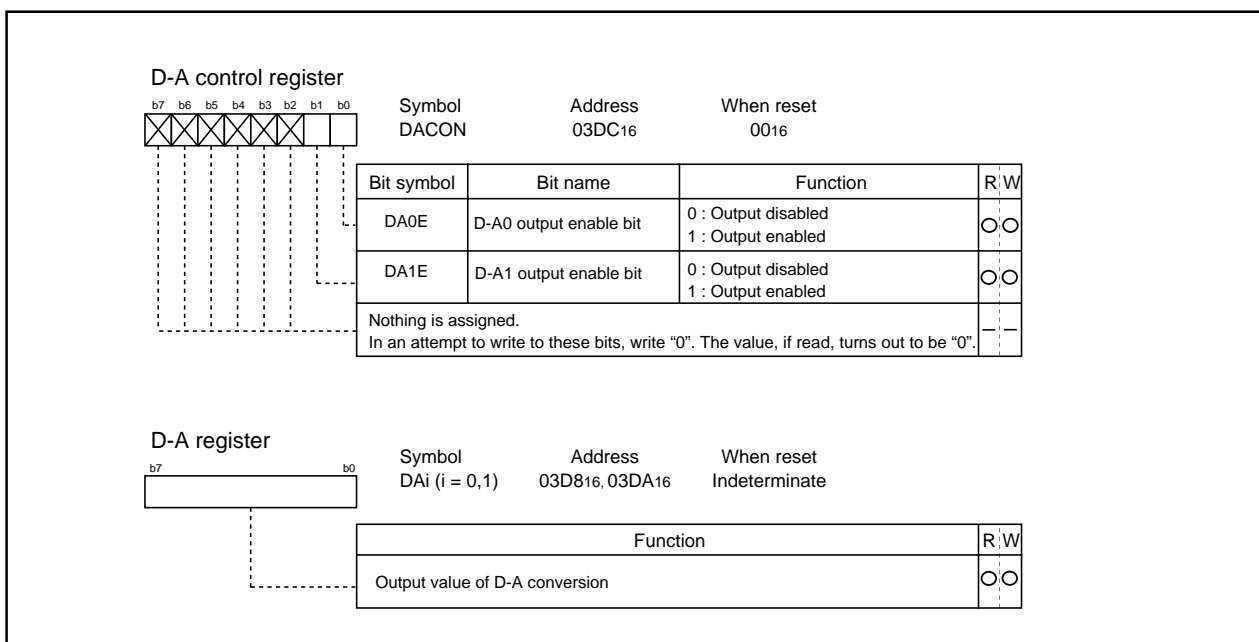
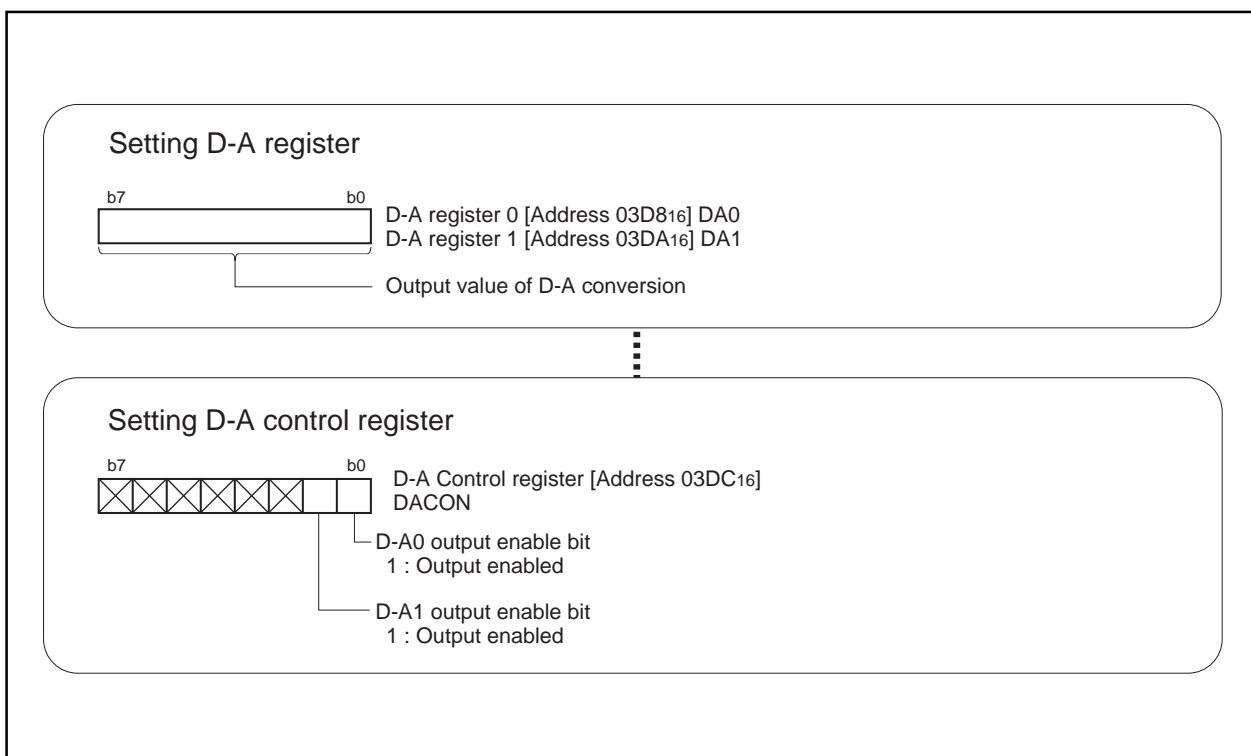


Figure 2.9.2. D-A converter-related registers

### 2.9.2 D-A Converter Operation

The following is the D-A converter operation. Figure 2.9.3 shows the set-up procedure.

- Operation
- (1) Writing a value to the D-A register *i* starts D-A conversion.
  - (2) Setting the D-Ai output enable bit to "1" outputs an analog signal on the DAi pin.
  - (3) The D-A converter continues outputting an analog signal until the D-A output enable bit is set to "0".



**Figure 2.9.3. Set-up procedure of D-A converter**



## 2.10 DMAC

### 2.10.1 Overview

DMAC transfers one data item held in the source address to the destination address every time a transfer request is generated. The following is a DMAC overview.

#### (1) Source address and destination address

Both the register which indicates a source and the register which indicates a destination comprise of 24 bits, so that each can cover a 1M bytes space. After transfer of one bit of data is completed, the address in either the source register or the destination register can be incremented. However, both registers cannot be incremented. The links between the source and destination are as follows:

- (a) A fixed address from an arbitrary 1M bytes space
- (b) An arbitrary 1M bytes space from a fixed address
- (c) A fixed address from another fixed address

#### (2) The number of bits of data transferred

The number of bit of data indicated by the transfer counter is transferred. If a 16-bit transfer is selected, up to 128 K bytes can be transferred. If an 8-bit transfer is selected, up to 64K bytes can be transferred. The transfer counter is decremented each time one bit of data is transferred, and a DMA interrupt occurs when the transfer counter underflows.

#### (3) DMA transfer factor

The DMA transfer factor can be selected from the following 15 factors: falling edge of  $\overline{INT0}/\overline{INT1}$  pin, timer A0 interrupt request through timer A4 interrupt request, timer B0 interrupt request through timer B2 interrupt request, UART0 transmission interrupt request, UART0 reception interrupt request, UART1 transmission interrupt request, UART1 reception interrupt request, A-D conversion interrupt request, and software trigger.

When software trigger is selected, DMA transfer is generated by writing "1" to software DMA interrupt request bit. When other factor is selected, DMA transfer is generated by generating corresponding interrupt request.

#### (4) Channel priority

If DMA0 transfer request and DMA1 transfer request occur simultaneously, priority is given to DMA0.

#### (5) Writing to a register

When writing to the source register or the destination register with DMA enabled, the content of the register with a fixed address will change at the time of writing. Therefore, the user should not write to a register with a fixed address when the DMA enable bit is set to "1". The contents of the register with 'forward direction' selected, and the transfer counter, are changed when reloaded. A reload occurs either when the transfer counter underflows, or when the DMA enable bit is re-enabled, after having been disabled.

The reload register can be written to, as in normal conditions.

#### (6) Reading to a register

The reload register can be read to, as in normal conditions.

**(7) Switching function**

**(a) Switching between one-shot transfer and repeated transfer**

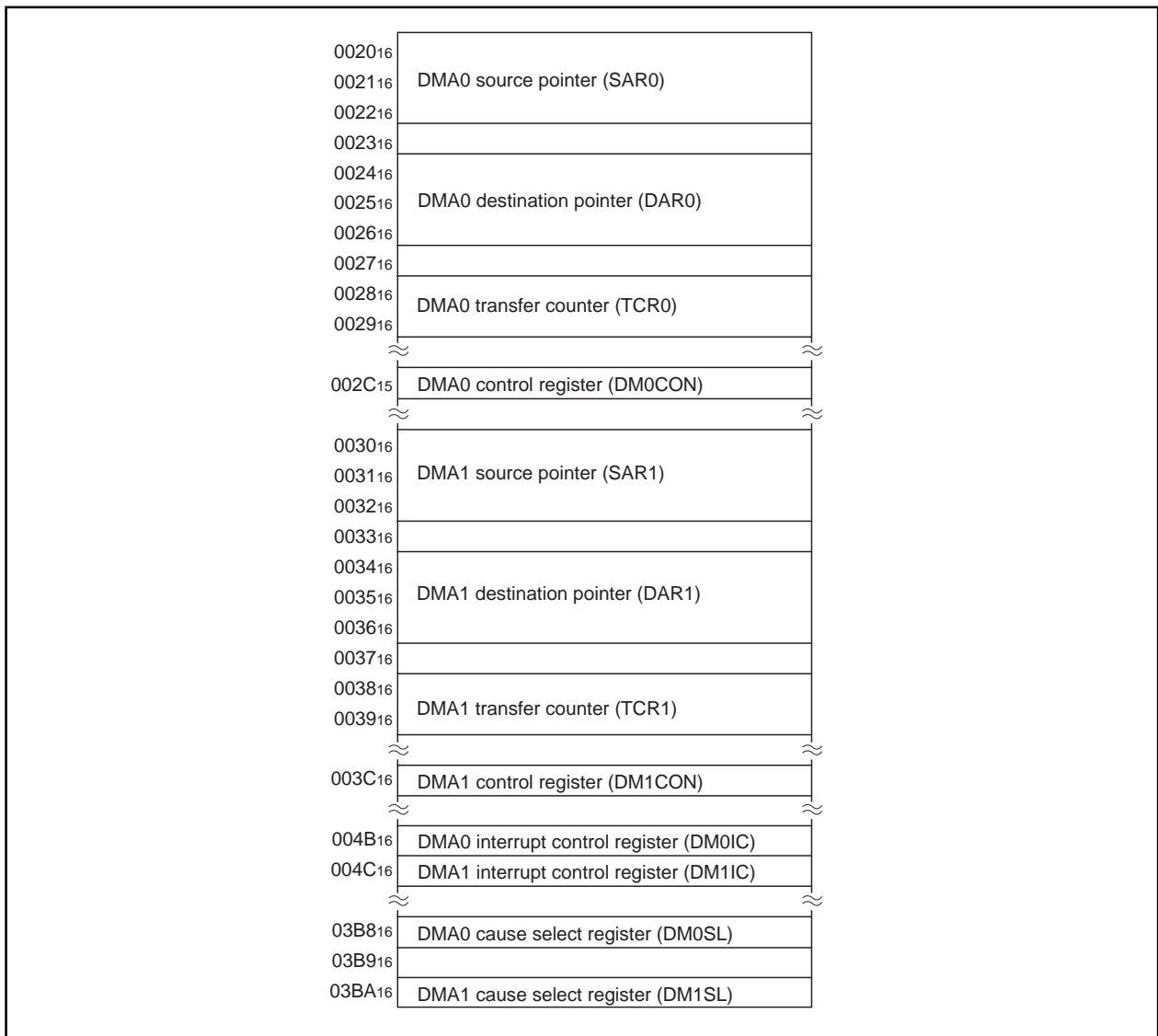
'One-shot transfer' refers to a mode in which DMA is disabled after the transfer counter underflows. 'Repeated transfer' refers to a mode in which a reload is carried out after the transfer counter underflows. The reload is carried out for the transfer counter and on the address pointer subjected to forward direction.

The following are examples of operation in which the options listed are selected.

- A fixed address from an arbitrary 1M byte space, one-shot transfer ..... P358
- An arbitrary 1M byte space from a fixed address, repeated transfer ..... P360

**(8) Registers related to DMAC**

Figure 2.10.1 shows the memory map of DMAC-related registers, and Figures 2.10.2 and 2.10.3 show DMAC-related registers.



**Figure 2.10.1. Memory map of DMAC-related registers**

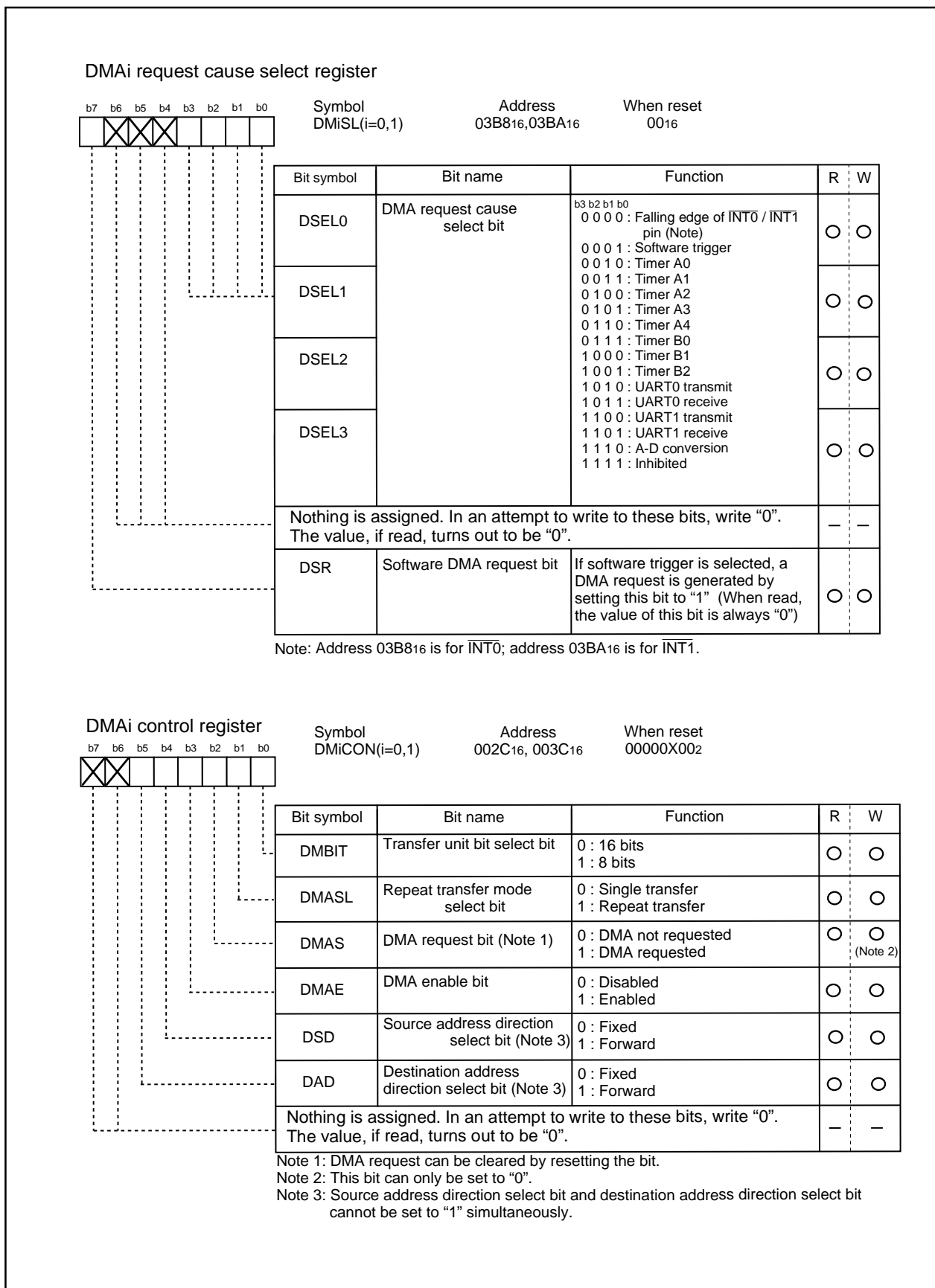


Figure 2.10.2. DMAC-related registers (1)

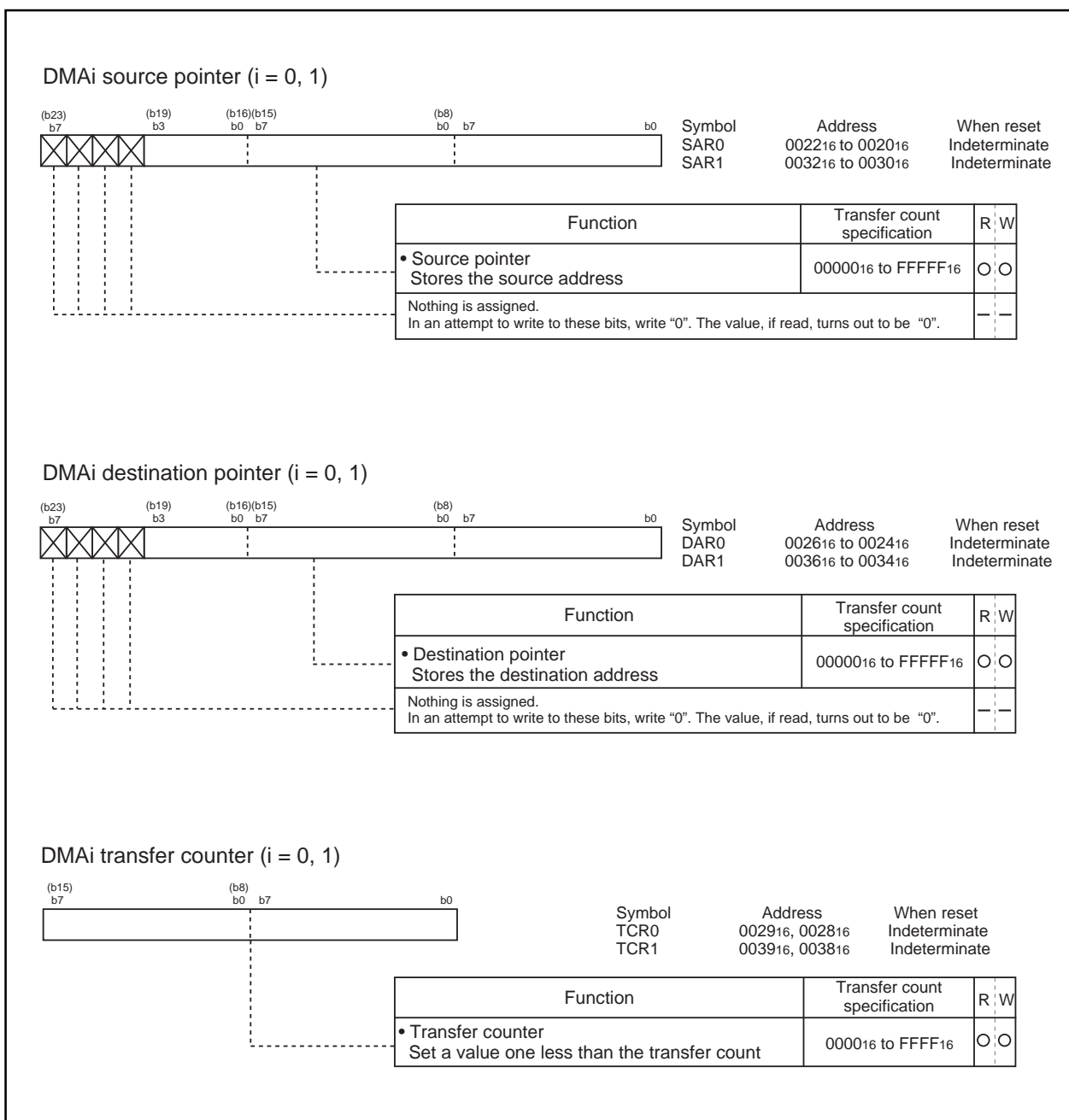


Figure 2.10.3. DMAC-related registers (2)

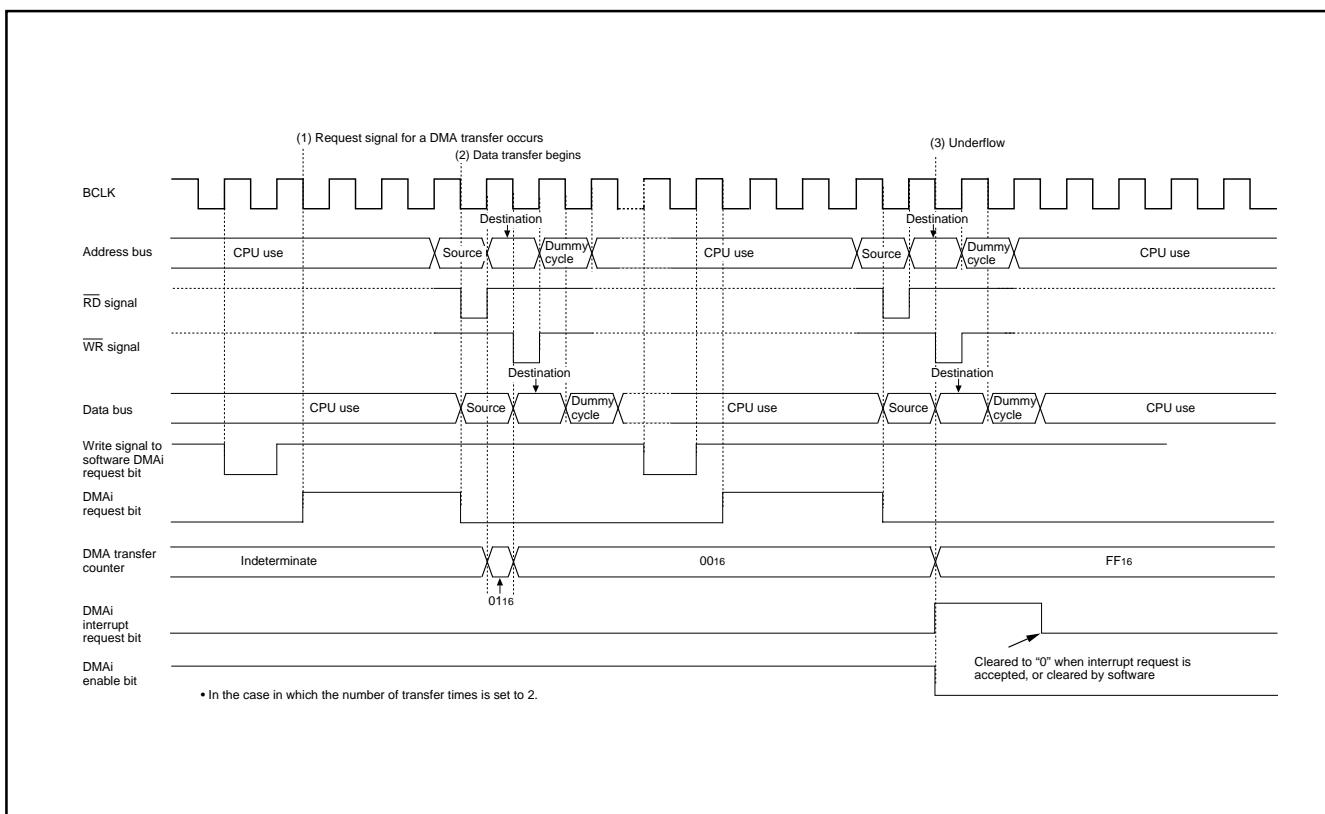
### 2.10.2 Operation of DMAC (one-shot transfer mode)

In one-shot transfer mode, choose functions from the items shown in Table 2.10.1. Operations of the circled items are described below. Figure 2.10.4 shows an example of operation and Figure 2.10.5 shows the set-up procedure.

**Table 2.10.1. Chosed functions**

Item	Set-up	
Transfer space	<b>○</b>	Fixed address from an arbitrary 1 M bytes space
		Arbitrary 1 M bytes space from a fixed address
		Fixed address from fixed address
Unit of transfer	<b>○</b>	8 bits
		16 bits

- Operation
- (1) When software trigger is selected, setting software DMA request bit to “1” generates a DMA transfer request signal.
  - (2) If DMAC is active, data transfer starts, and the contents of the address indicated by the DMAi forward-direction address pointer are transferred to the address indicated by the DMAi destination pointer. When data transfer starts directly after DMAC becomes active, the value of the DMAi transfer counter reload register is reloaded to the DMAi transfer counter, and the value of the DMAi source pointer is reloaded by the DMAi forward-direction address pointer. Each time a DMA transfer request signal is generated, 1 byte of data is transferred. The DMAi transfer counter is down counted, and the DMAi forward-direction address pointer is up counted.
  - (3) If the DMA transfer counter underflows, the DMA enable bit changes to “0” and DMA transfer is completed. The DMA interrupt request bit changes to “1” simultaneously.



**Figure 2.10.4. Example of operation of one-shot transfer mode**

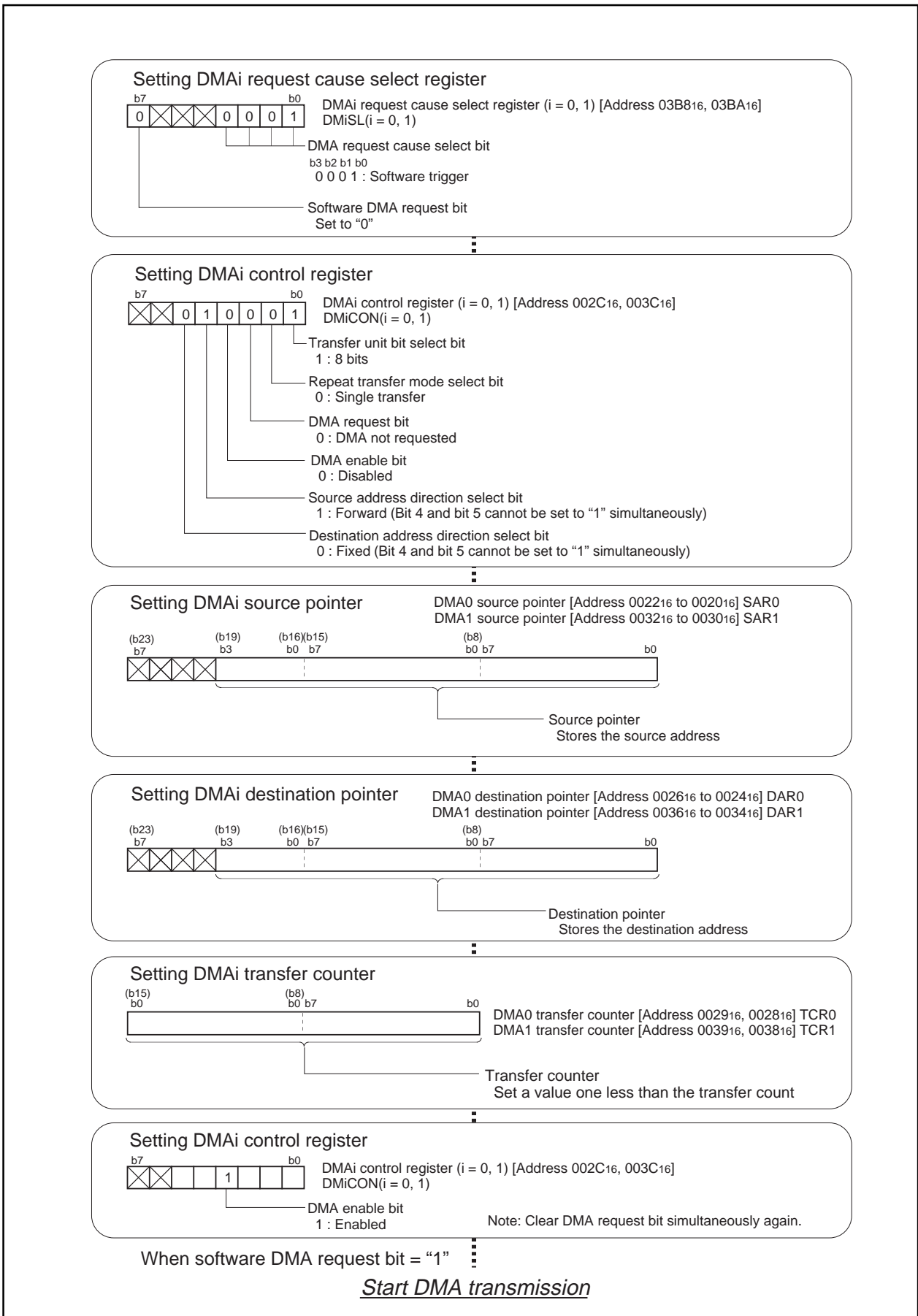


Figure 2.10.5. Set-up procedure of one-shot transfer mode

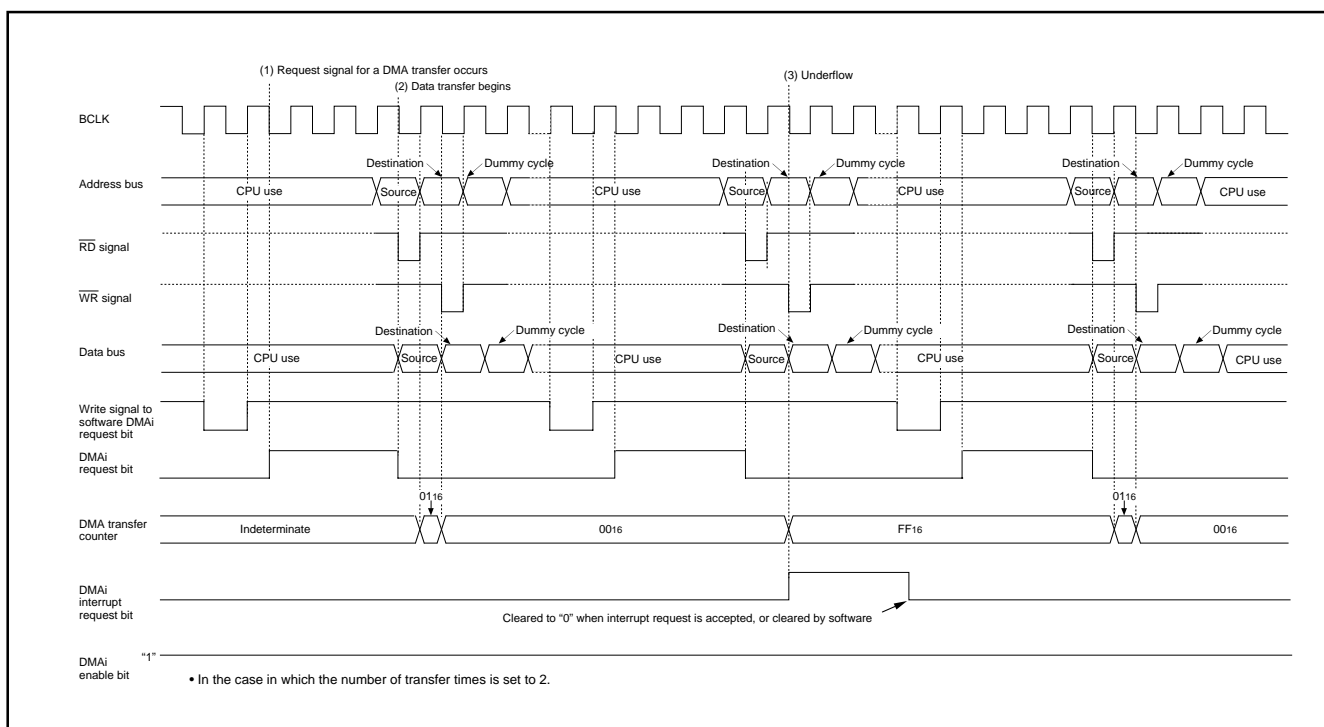
### 2.10.3 Operation of DMAC (repeated transfer mode)

In repeat transfer mode, choose functions from the items shown in Table 2.10.2. Operations of the circled items are described below. Figure 2.10.6 shows an example of operation and Figure 2.10.7 shows the set-up procedure.

**Table 2.10.2. Chosed functions**

Item	Set-up
Transfer space	Fixed address from an arbitrary 1 M bytes space
	<input checked="" type="radio"/> Arbitrary 1 M bytes space from a fixed address
	Fixed address from fixed address
Unit of transfer	8 bits
	<input checked="" type="radio"/> 16 bits

- Operation
- (1) When software trigger is selected, setting software DMA request bit to “1” generates a DMA transfer request signal.
  - (2) If DMAC is active, data transfer starts, and the contents of the address indicated by the DMAi forward-direction address pointer are transferred to the address indicated by the DMAi destination pointer. When data transfer starts directly after DMAC becomes active, the value of the DMAi transfer counter reload register is reloaded to the DMAi transfer counter, and the value of the DMAi source pointer is reloaded by the DMAi forward-direction address pointer. Each time a DMA transfer request signal is generated, 2 byte of data is transferred. The DMAi transfer counter is down counted, and the DMAi forward-direction address pointer is up counted.
  - (3) Though DMAi transfer counter is underflowed, DMA enable bit is still “1”. The DMA interrupt request bit changes to “1” simultaneously.
  - (4) After DMAi transfer counter is underflowed, when the next DMA request is generated, DMA transfer is repeated from (1).



**Figure 2.10.6. Example of operation of repeated transfer mode**

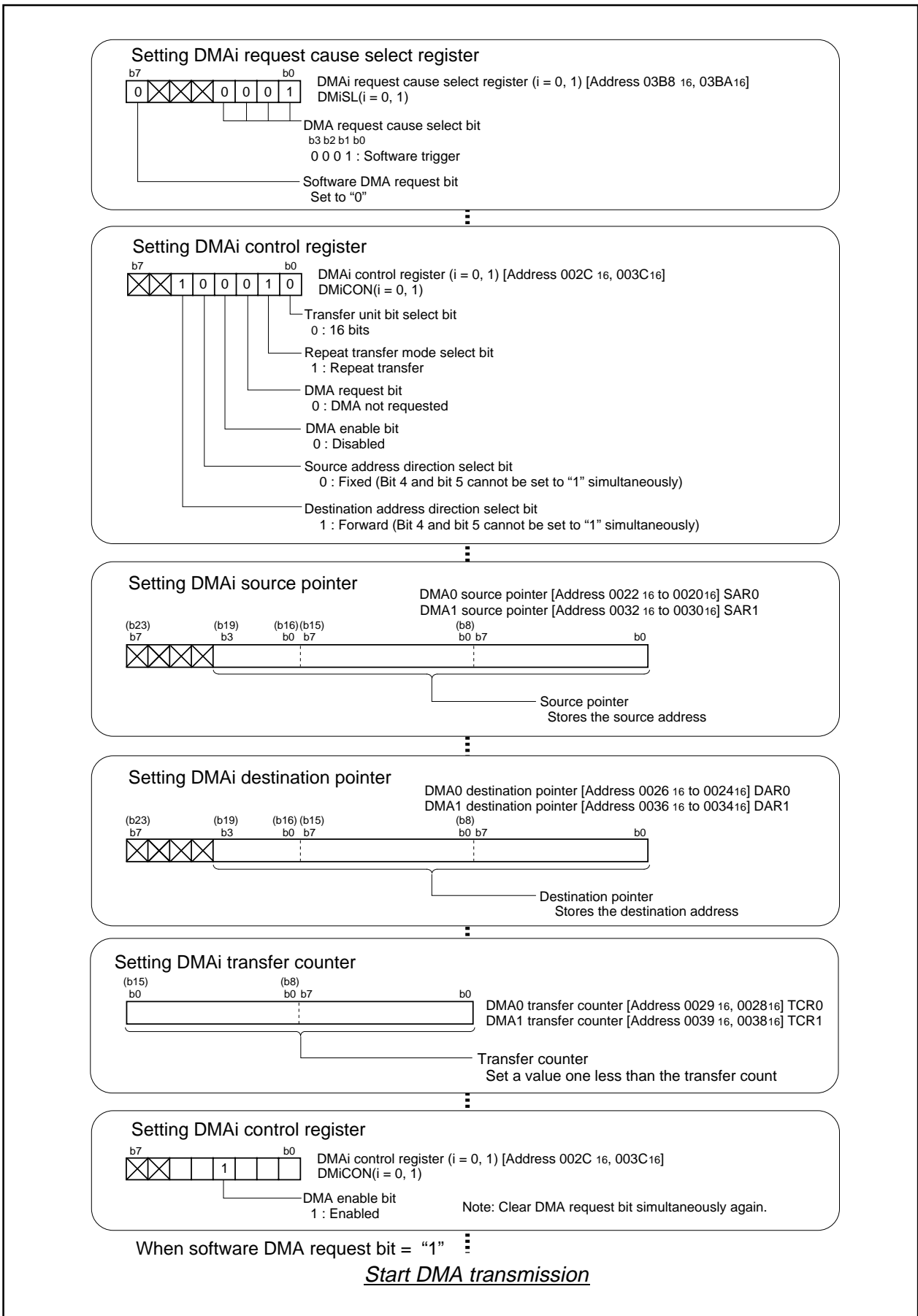


Figure 2.10.7. Set-up procedure of repeated transfer mode



## CRC Calculation Circuit

## 2.11 CRC Calculation Circuit

## 2.11.1 Overview

Cyclic Redundancy Check (CRC) is a method that compares CRC code formed from transmission data by use of a polynomial generation with CRC check data so as to detect errors in transmission data. Using the CRC calculation circuit allows generation of CRC code. A polynomial counter is used for the polynomial generation.

## (1) Registers related to CRC calculation circuit

Figure 2.11.1 shows the memory map of CRC-related registers, and Figure 2.11.2 shows CRC-related registers.

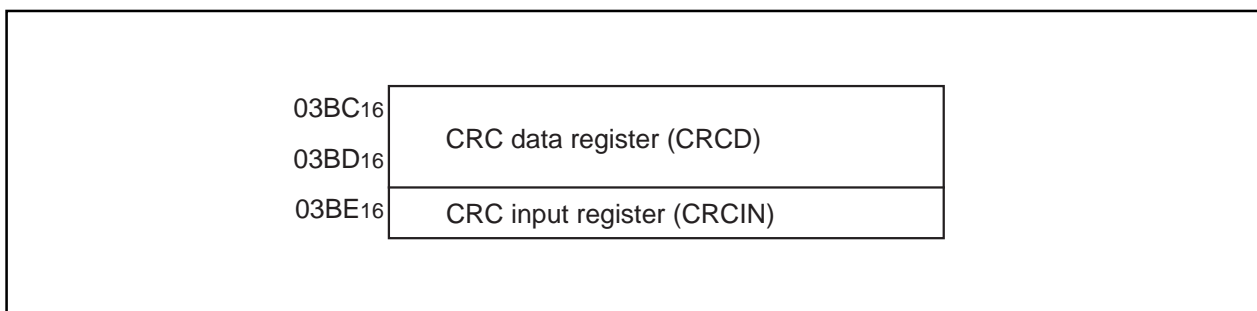


Figure 2.11.1. Memory map of CRC-related registers

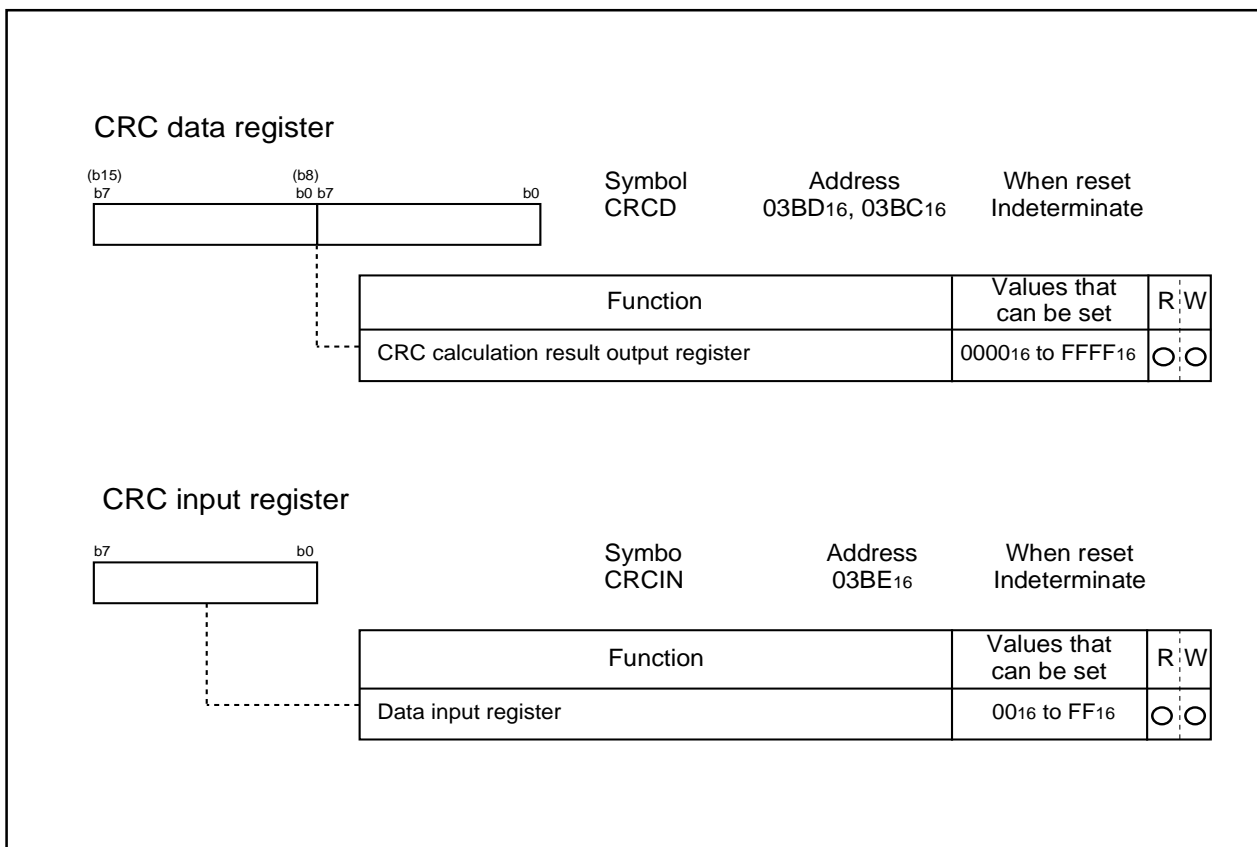


Figure 2.11.2. CRC-related registers

CRC Calculation Circuit

2.11.2 Operation of CRC Calculation Circuit

The following describes the operation of the CRC calculation. Figure 2.11.3 shows an example of calculation data 0123<sub>16</sub> using the CRC calculation circuit.

- Operation (1) The CRC calculation circuit sets an initial value in the CRC data register.
- (2) Writing 1 byte data to the CRC input register generates CRC code based on the data register. CRC code generation for 1 byte data finishes in two machine cycles.
- (3) The CRC calculation circuit detects an error by means of comparing the CRC-checking data with the content of the CRC data register, after the next data is written to the CRC input register.
- (4) The content of CRC data register after all data is written becomes CRC code.

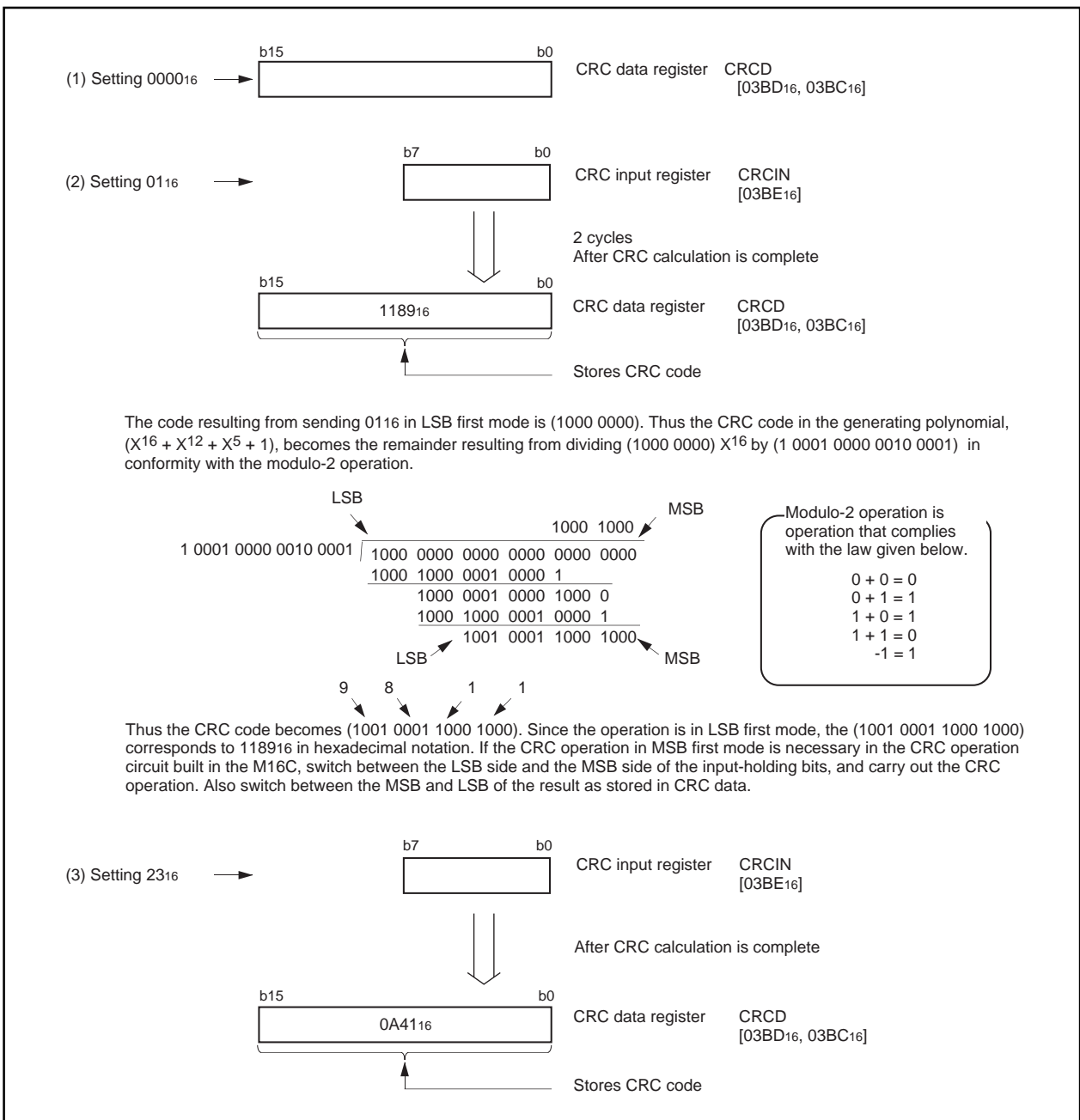


Figure 2.11.3. Calculation example using the CRC calculation circuit

## 2.12 Watchdog Timer

### 2.12.1 Overview

The watchdog timer can detect a runaway program using its 15-bit timer prescaler. The following is an overview of the watchdog timer.

#### (1) Watchdog timer start procedure

When reset, the watchdog timer is in stopped state. Writing to the watchdog timer start register initializes the watchdog timer to 7FFF<sub>16</sub> and causes it to start performing a down count. The watchdog timer, once started operating, cannot be stopped by any means other than stopping conditions.

#### (2) Watchdog timer stop conditions

The watchdog timer stops in any one of the following states:

- (a) Period in which the CPU is in stopped state
- (b) Period in which the CPU is in waiting state
- (c) Period in which the microcomputer is in hold state

#### (3) Watchdog timer initialization

The watchdog timer is initialized to 7FFF<sub>16</sub> in the cases given below, and begins a down count.

- (a) When the watchdog timer writes to the watchdog timer start register while a count is in progress
- (b) When the watchdog timer underflows

#### (4) Runaway detection

When the watchdog timer underflows, a watchdog timer interrupt occurs. In writing a program, write to the watchdog timer start register before the watchdog timer underflows. The watchdog timer interrupt occurs regardless of the status of the interrupt enable flag (I flag). In processing a watchdog timer interrupt, set the software reset bit to "1" to reset software.

#### (5) Watchdog timer cycle

The watchdog timer cycle varies depending on the BCLK and the frequency division ratio of the prescaler selected.

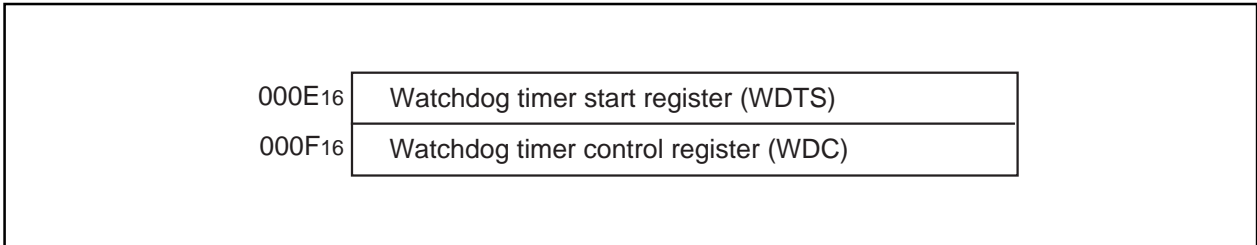
**Table 2.12.1. The watchdog timer cycle**

CM07	CM06	CM17	CM16	BCLK	WDC7	Period
0	0	0	0	10MHz	0	Approx. 52.4ms (Note)
					1	Approx. 419.4ms (Note)
0	0	0	1	5MHz	0	Approx. 104.9ms (Note)
					1	Approx. 838.9ms (Note)
0	0	1	0	2.5MHz	0	Approx. 209.7ms (Note)
					1	Approx. 1.68s (Note)
0	0	1	1	0.625MHz	0	Approx. 838.9ms (Note)
					1	Approx. 6.71s (Note)
0	1	Invalid	Invalid	1.25MHz	0	Approx. 419.4ms (Note)
					1	Approx. 3.36s (Note)
1	Invalid	Invalid	Invalid	32kHz	Invalid	Approx. 2s (Note)

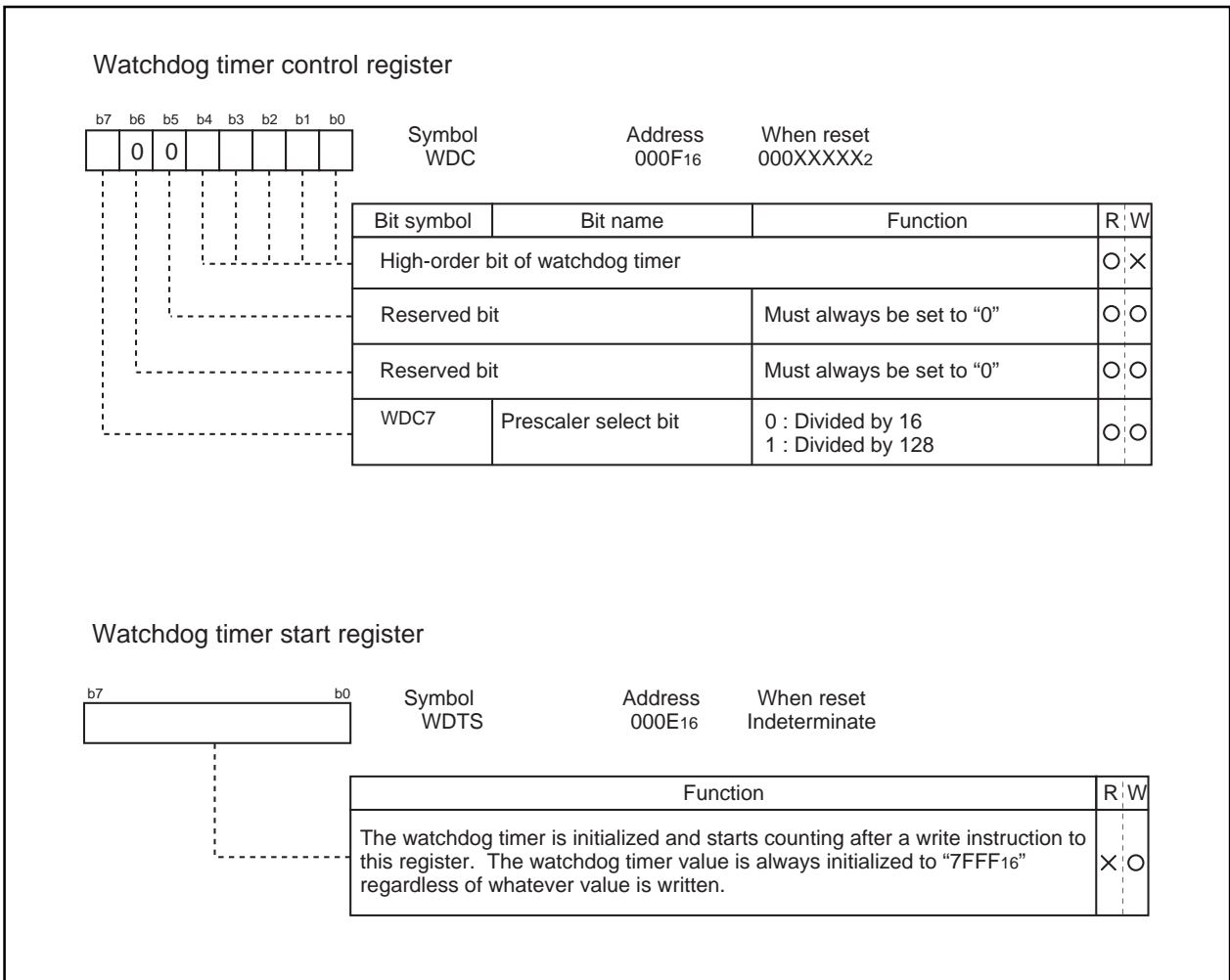
Note: An error due to the prescaler occurs.

**(6) Registers related to the watchdog timer**

Figure 2.12.1 shows the memory map of watchdog timer-related registers, and Figure 2.12.2 shows watchdog timer-related registers.



**Figure 2.12.1. Memory map of watchdog timer-related registers**

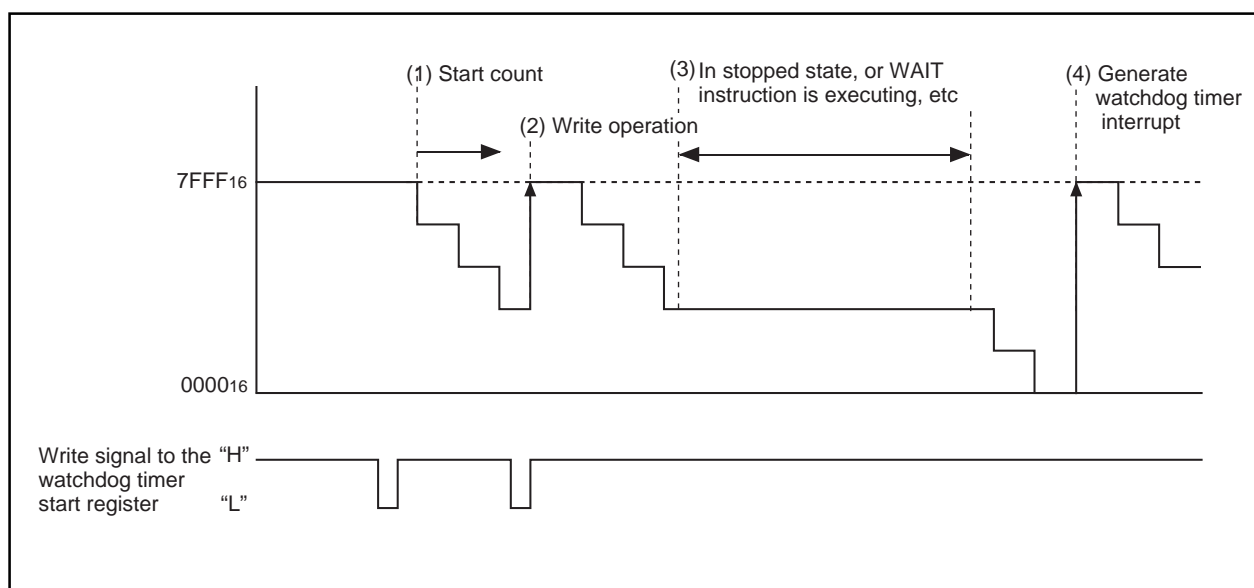


**Figure 2.12.2. Watchdog timer-related registers**

### 2.12.2 Operation of Watchdog Timer

The following is an operation of the watchdog timer. Figure 2.12.3 shows the operation timing, and Figure 2.12.4 shows the set-up procedure.

- Operation
- (1) Writing to the watchdog timer start register initializes the watchdog timer to  $7FFF_{16}$  and causes it to start a down count.
  - (2) With a count in progress, writing to the watchdog timer start register again initializes the watchdog timer to  $7FFF_{16}$  and causes it to resume counting.
  - (3) Either executing the WAIT instruction or going to the stopped state causes the watchdog timer to hold the count in progress and to stop counting. The watchdog timer resumes counting after returning from the execution of the WAIT instruction or from the stopped state.
  - (4) If the watchdog timer underflows, it is initialized to  $7FFF_{16}$  and continues counting. At this time, a watchdog timer interrupt occurs.



**Figure 2.12.3. Operation timing of watchdog timer**

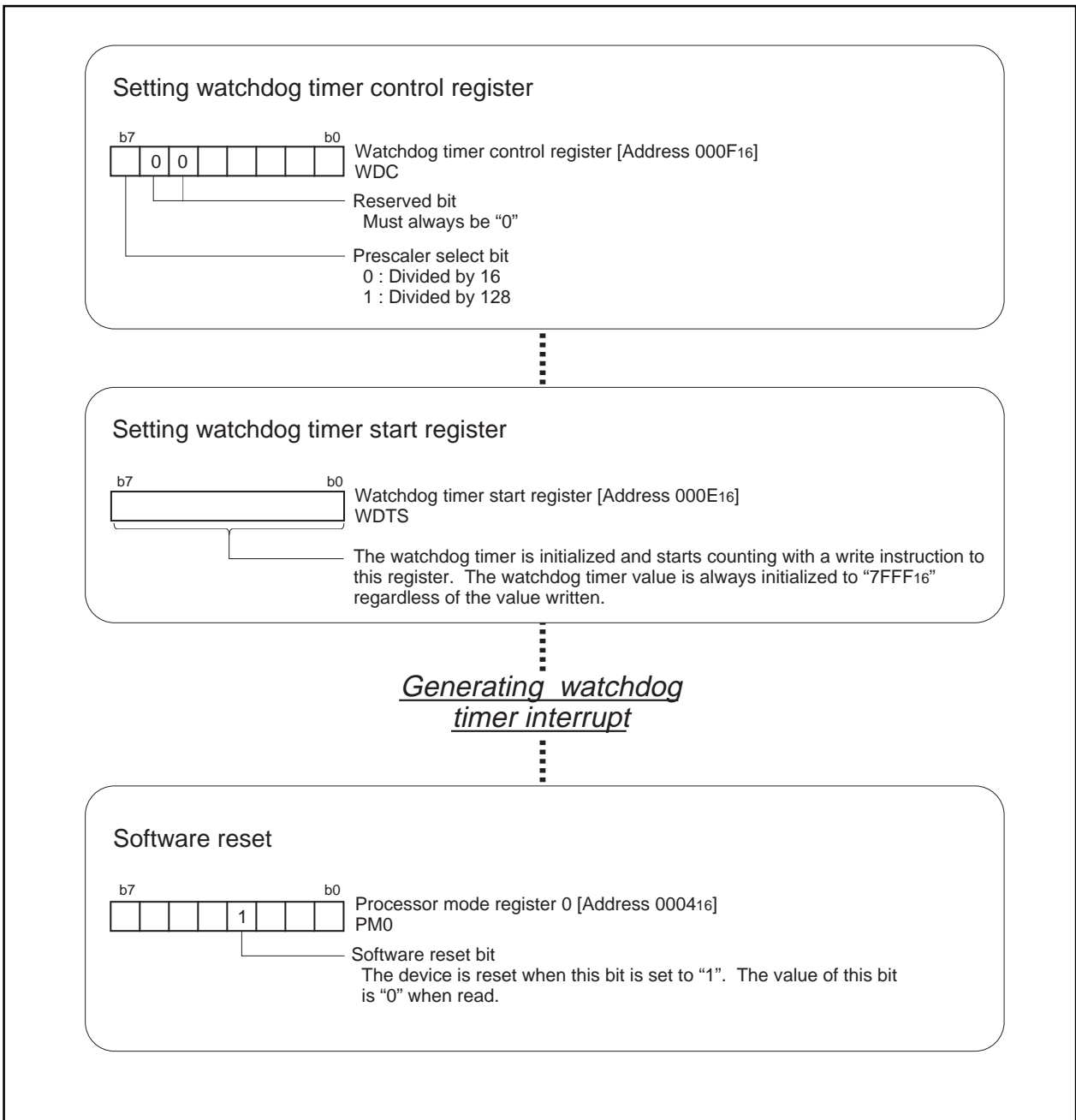


Figure 2.12.4. Set-up procedure of watchdog timer

## 2.13 Address Match Interrupt

### 2.13.1 Overview

The address match interrupt is used for correcting a ROM or for a simplified debugging-purpose monitor. The following is an overview of the address match interrupt.

#### (1) Enabling/disabling the address match interrupt

The address match interrupt enable bit can be used to enable and disable an address match interrupt. It is affected neither by the processor interrupt priority level (IPL) nor the interrupt enable flag (I flag).

#### (2) Timing of the address match interrupt

An interrupt occurs immediately before executing the instruction in the address indicated by the address match interrupt register. Set the first address of the instruction in the address match interrupt register. Setting a half address of an instruction or an address of tabulated data does not generate an address match interrupt.

The first instruction of an interrupt routine does not generate an address match interrupt either.

#### (3) Returning from an address match interrupt

The return address put in the stack when an address match interrupt occurs depends on the instruction not yet executed (the instruction the address match interrupt register indicates). The return address is not put in the stack. For this reason, to return from an address match interrupt, either rewrite the content of the stack and use the REIT instruction or use the POP instruction to restore the stack to the state as it was before the interrupt occurred and return by use of a jump instruction.

Figure 2.13.1 shows unexecuted instructions and corresponding the stacked addresses.

<Instructions whose address is added to by 2 when an address match interrupt occurs>					
• 16-bit operation code instructions					
• 8-bit operation code instructions given below					
ADD.B:S	#IMM8,dest	SUB.B:S	#IMM8,dest	AND.B:S	#IMM8,dest
OR.B:S	#IMM8,dest	MOV.B:S	#IMM8,dest	STZ.B:S	#IMM8,dest
STNZ.B:S	#IMM8,dest	STZX.B:S	#IMM81,#IMM82,dest		
CMP.B:S	#IMM8,dest	PUSHM	src	POPM	dest
JMPS	#IMM8	JSRS	#IMM8		
MOV.B:S	#IMM,dest	(However, dest = A0/A1)			
<Instructions whose address is added to by 1 when an address match interrupt occurs>					
• Instructions other than those listed above					

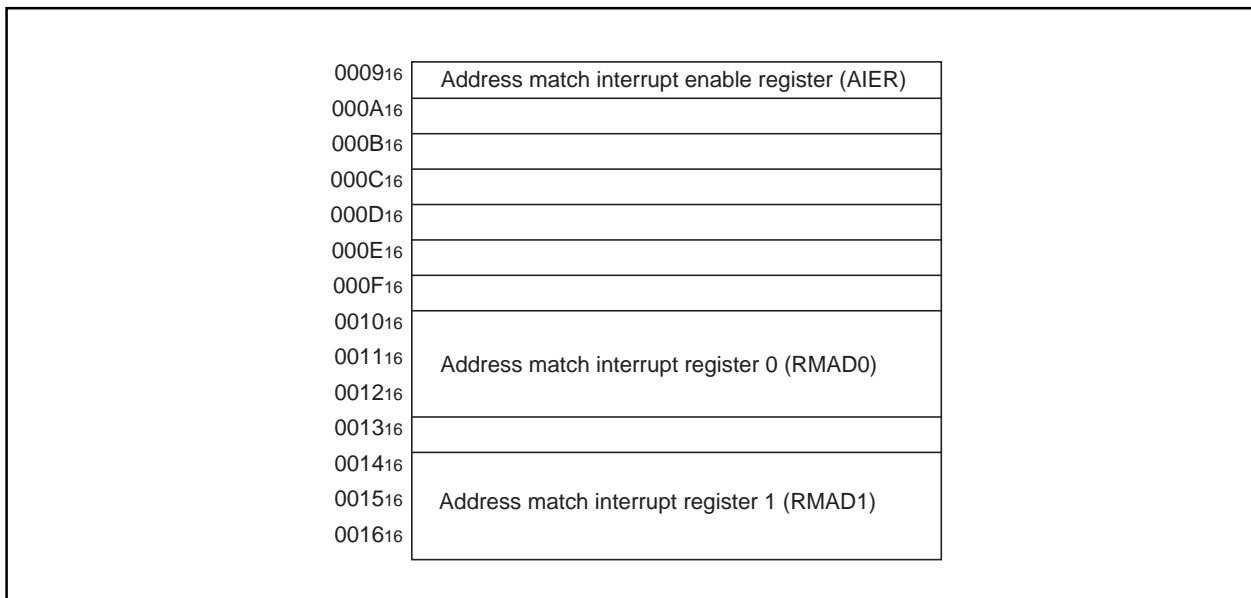
Figure 2.13.1. Unexecuted instructions and corresponding stacked addresses

#### (4) How to determine an address match interrupt

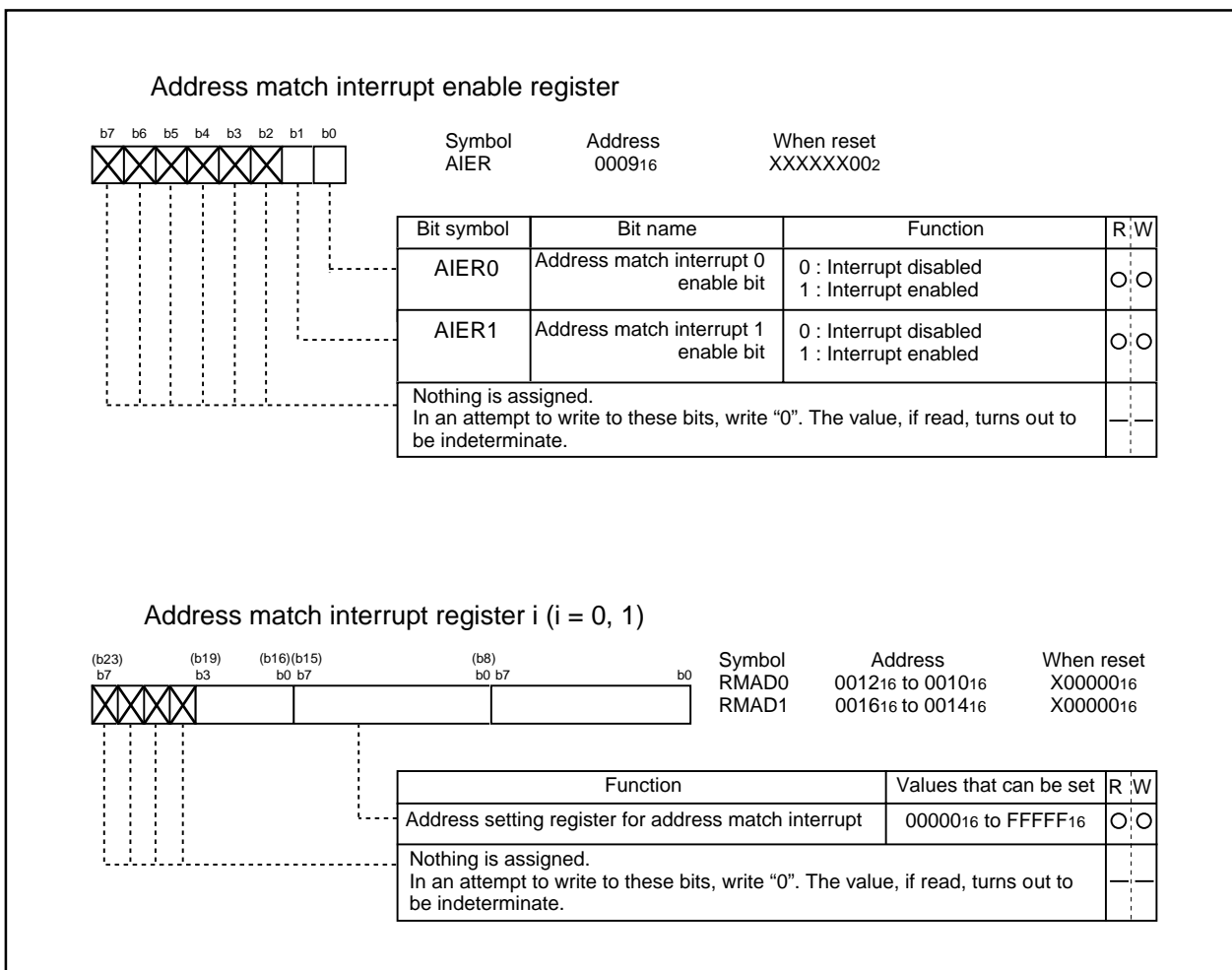
Address match interrupts can be set at two different locations. However, both location will have the same vector address. Therefore, it is necessary to determine which interrupt has occurred; address match interrupt 0 or address match interrupt 1. Using the content of the stack, etc., determine which interrupt has occurred according to the first part of the address match interrupt routine.

**(5) Registers related to the address match interrupt**

Figure 2.13.2 shows the memory map of address match interrupt-related registers, and Figure 2.13.3 shows address match interrupt-related registers.



**Figure 2.13.2. Memory map of address match interrupt-related registers**



**Figure 2.13.3. Address match interrupt-related registers**



## Address Match Interrupt

## 2.13.2 Operation of Address Match Interrupt

The following is an operation of address match interrupt. Figure 2.13.4 shows the set-up procedure of address match interrupt, and Figure 2.13.5 shows the overview of the address match interrupt handling routine.

- Operation
- (1) The address match interrupt handling routine sets an address to be used to cause the address match interrupt register to generate an interrupt.
  - (2) Setting the address match enable flag to "1" enables an interrupt to occur.
  - (3) An address match interrupt occurs immediately before the instruction in the address indicated by the address match interrupt register as a program is executed.

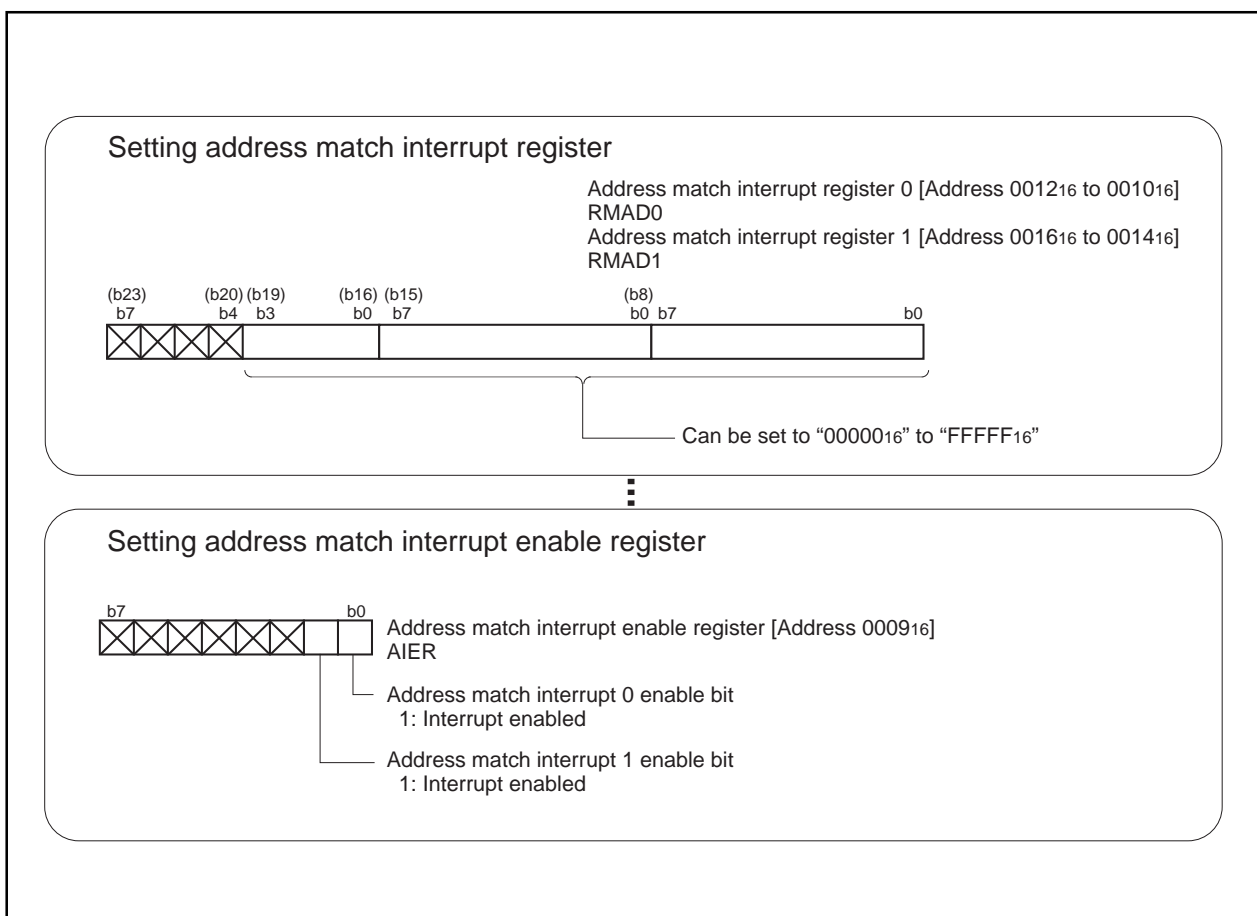


Figure 2.13.4. Set-up procedure of address match interrupt

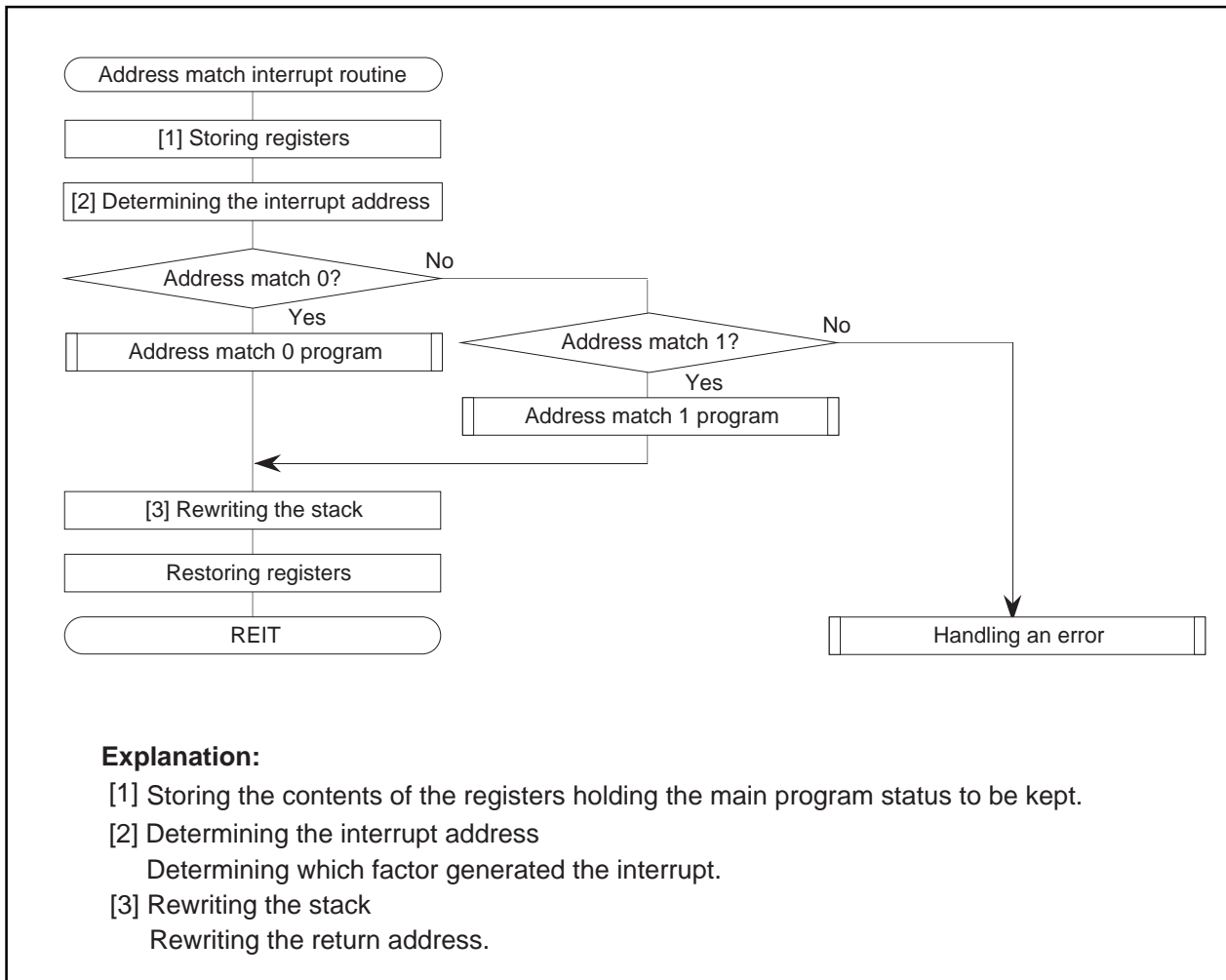


Figure 2.13.5. Overview of the address match interrupt handling routine

## 2.14 Power Control

### 2.14.1 Overview

'Power Control' refers to the reduction of CPU power consumption by stopping the CPU and oscillators, or decreasing the operation clock. The following is a description of the three available power control modes:

#### (1) 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 BCLK 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 BCLK 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 2.14.1 shows the state transition diagram of the above modes.

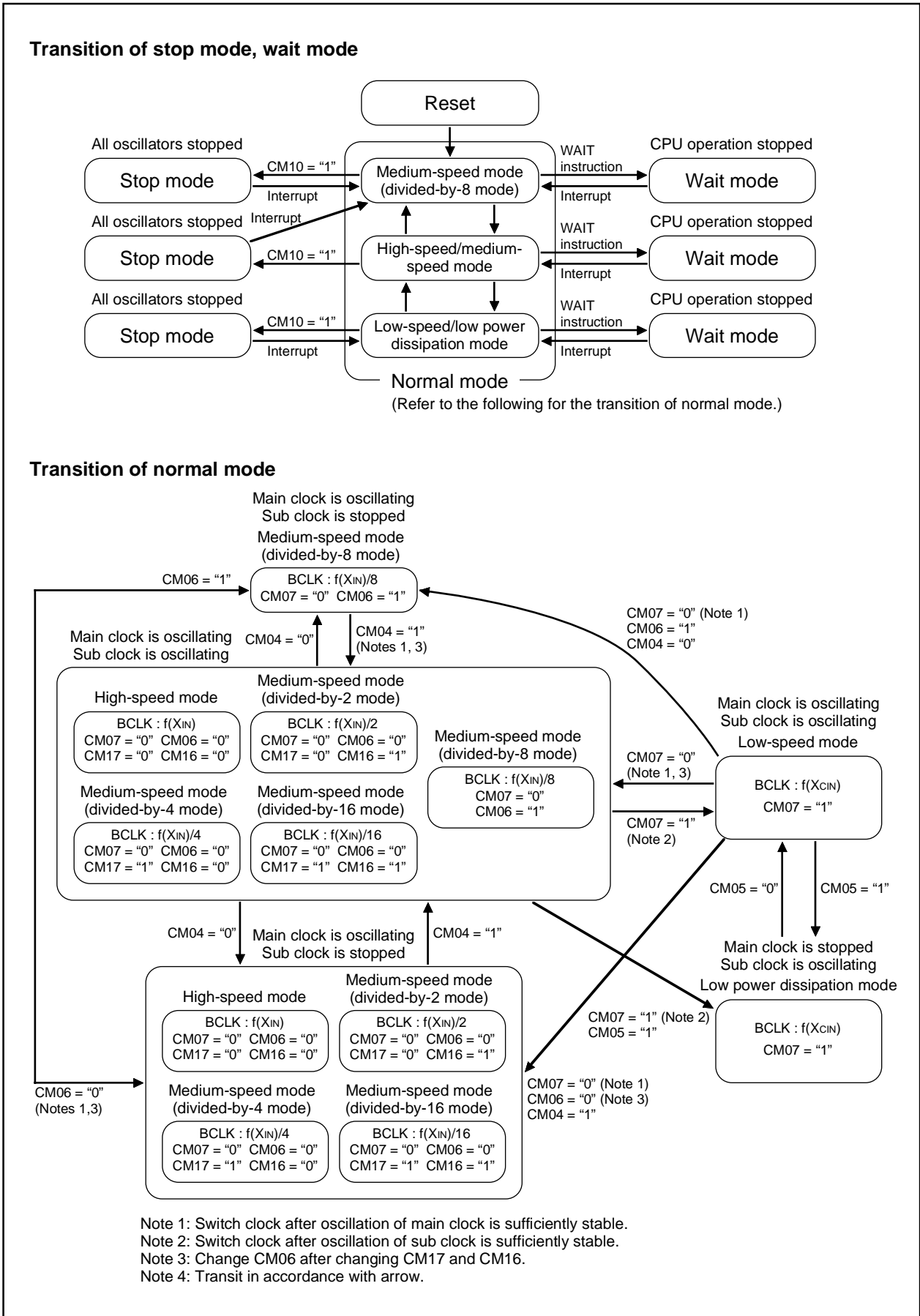


Figure 2.14.1. State transition diagram of power control mode

**(2) Switching the driving capacity of the oscillation circuit**

Both the main clock and the secondary clock have the ability to switch the driving capacity. Reducing the driving capacity after the oscillation stabilizes allows for further reduction in power consumption.

**(3) Clearing stop mode and wait mode**

The stop mode and wait mode can be cleared by generating an interrupt request, or by resetting hardware. Set the priority level of the interrupt to be used for clearing, higher than the processor interrupt priority level (IPL), and enable the interrupt enable flag (I flag). When an interrupt clears a mode, that interrupt is processed. Table 2.14.1 shows the interrupts that can be used for clearing a stop mode and wait mode.

**(4) BCLK in returning from wait mode or stop mode****(a) Returning from wait mode**

The processor immediately returns to the BCLK, which was in use before entering wait mode.

**(b) Returning from stop mode**

CM06 is set to "1" when the device enters stop mode after selecting the main clock for BCLK. CM17, CM16, and CM07 do not change state. In this case, when restored from stop mode, the device starts operating in divided-by-8 mode.

When the device enters stop mode after selecting the subclock for BCLK, CM06, CM17, CM16, and CM07 all do not change state. In this case, when restored from stop mode, the device starts operating in low-speed mode.

**Table 2.14.1. Interrupts available for clearing stop mode and wait mode**

Interrupt for clearing	Wait mode		Stop mode
	CM02 = 0	CM02 = 1	
DMA0 interrupt	Impossible	Impossible	Impossible
DMA1 interrupt	Impossible	Impossible	Impossible
A-D interrupt	Note 3	Impossible	Impossible
UART0 transmit interrupt	Possible	Note 1	Note 1
UART0 receive interrupt	Possible	Note 1	Note 1
UART1 transmit interrupt	Possible	Note 1	Note 1
UART1 receive interrupt	Possible	Note 1	Note 1
SI/O automatic transfer interrupt	Possible	Impossible	Impossible
FLD interrupt	Possible	Impossible	Impossible
Timer A0 interrupt	Possible	Note 2	Note 2
Timer A1 interrupt	Possible	Note 2	Note 2
Timer A2 interrupt	Possible	Note 2	Note 2
Timer A3 interrupt	Possible	Note 2	Note 2
Timer A4 interrupt	Possible	Note 2	Note 2
Timer B0 interrupt	Possible	Note 2	Note 2
Timer B1 interrupt	Possible	Note 2	Note 2
Timer B2 interrupt	Possible	Note 2	Note 2
$\overline{\text{INT0}}$ interrupt	Possible	Possible	Possible
$\overline{\text{INT1}}$ interrupt	Possible	Possible	Possible
$\overline{\text{INT2}}$ interrupt	Possible	Possible	Possible
$\overline{\text{INT3}}$ interrupt	Possible	Possible	Possible
$\overline{\text{INT4}}$ interrupt	Possible	Possible	Possible
$\overline{\text{INT5}}$ interrupt	Possible	Possible	Possible

Note 1: Can be used when an external clock in clock synchronous serial I/O mode is selected.

Note 2: Can be used when the external signal is being counted in event counter mode.

Note 3: Can be used in one-shot mode and one-shot sweep mode.

**(5) Sequence of returning from stop mode**

Sequence of returning from stop mode is oscillation start-up time and interrupt sequence.

When interrupt is generated in stop mode, CM10 becomes "0" and clearing stop mode.

Starting oscillation and supplying BCLK execute the interrupt sequence as follow:

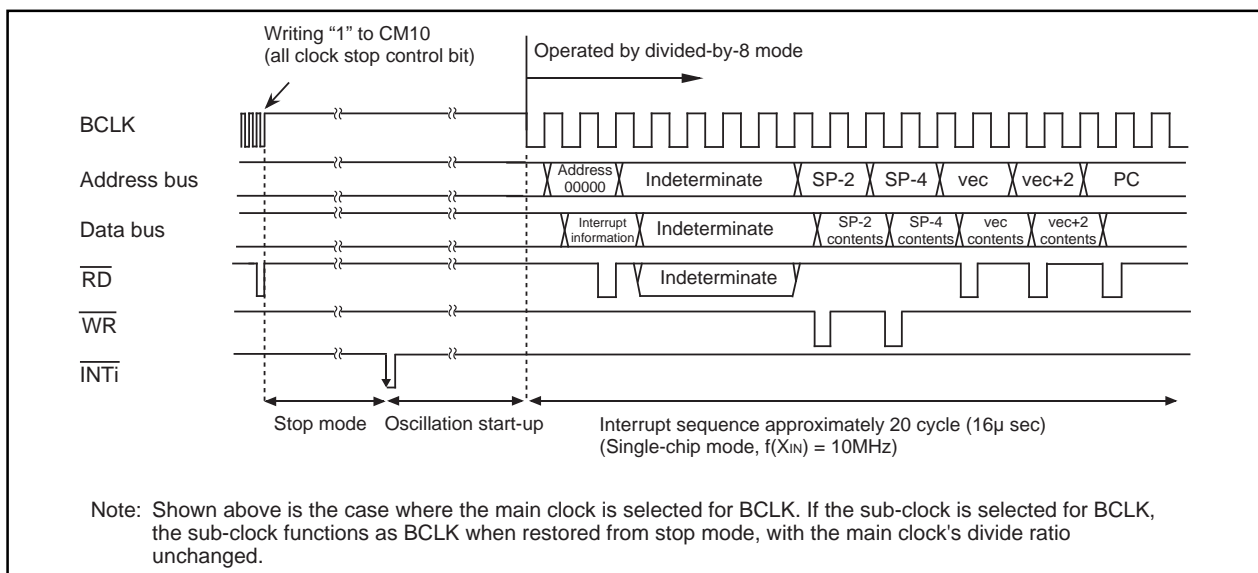
In the interrupt sequence, the processor carries out the following in sequence given:

- CPU gets the interrupt information (the interrupt number and interrupt request level) by reading address 00000<sub>16</sub>. The interrupt request bit of the interrupt written in address 00000<sub>16</sub> will then be set to "0".
- 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.
- Sets the interrupt enable flag (I flag), the debug flag (D flag), and the stack pointer assignment 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)
- Saves the content of the temporary register (Note) within the CPU in the stack area.
- Saves the content of the program counter (PC) in the stack area.
- Sets the interrupt priority level of the accepted instruction in the IPL.

Note: This register cannot be utilized by the user.

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

Figure 2.14.2 shows the sequence of returning from stop mode.



**Figure 2.14.2. Sequence of returning from stop mode**

**(6) Registers related to power control**

Figure 2.14.3 shows the memory map of power control-related registers, and Figure 2.14.4 shows power control-related registers.

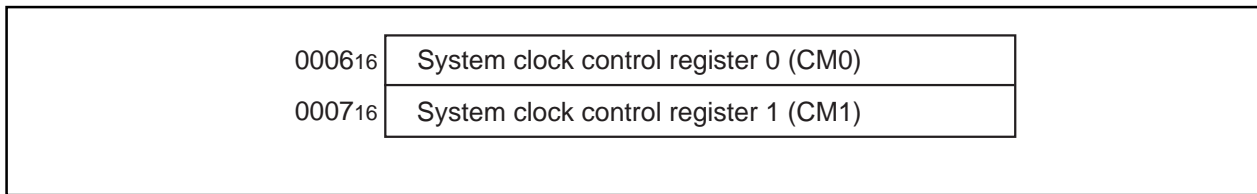


Figure 2.14.3. Memory map of power control-related registers

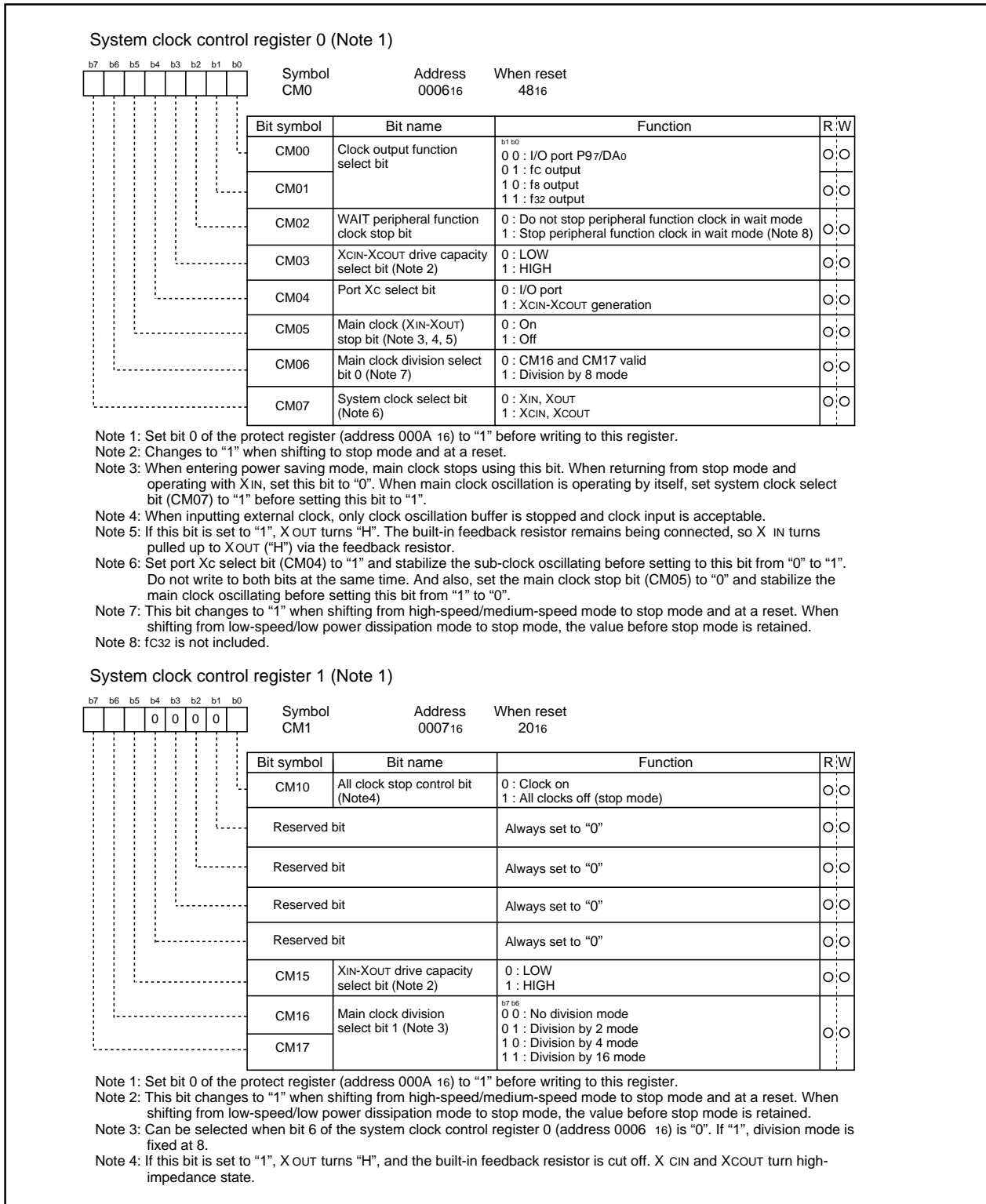


Figure 2.14.4. Power control-related registers

### 2.14.2 Stop Mode Set-Up

Settings and operation for entering stop mode are described here.

- Operation
- (1) Enables the interrupt used for returning from stop mode.
  - (2) Sets the interrupt enable flag (I flag) to "1".
  - (3) Clearing the protection and setting every-clock stop bit to "1" stops oscillation and causes the processor to go into stop mode.

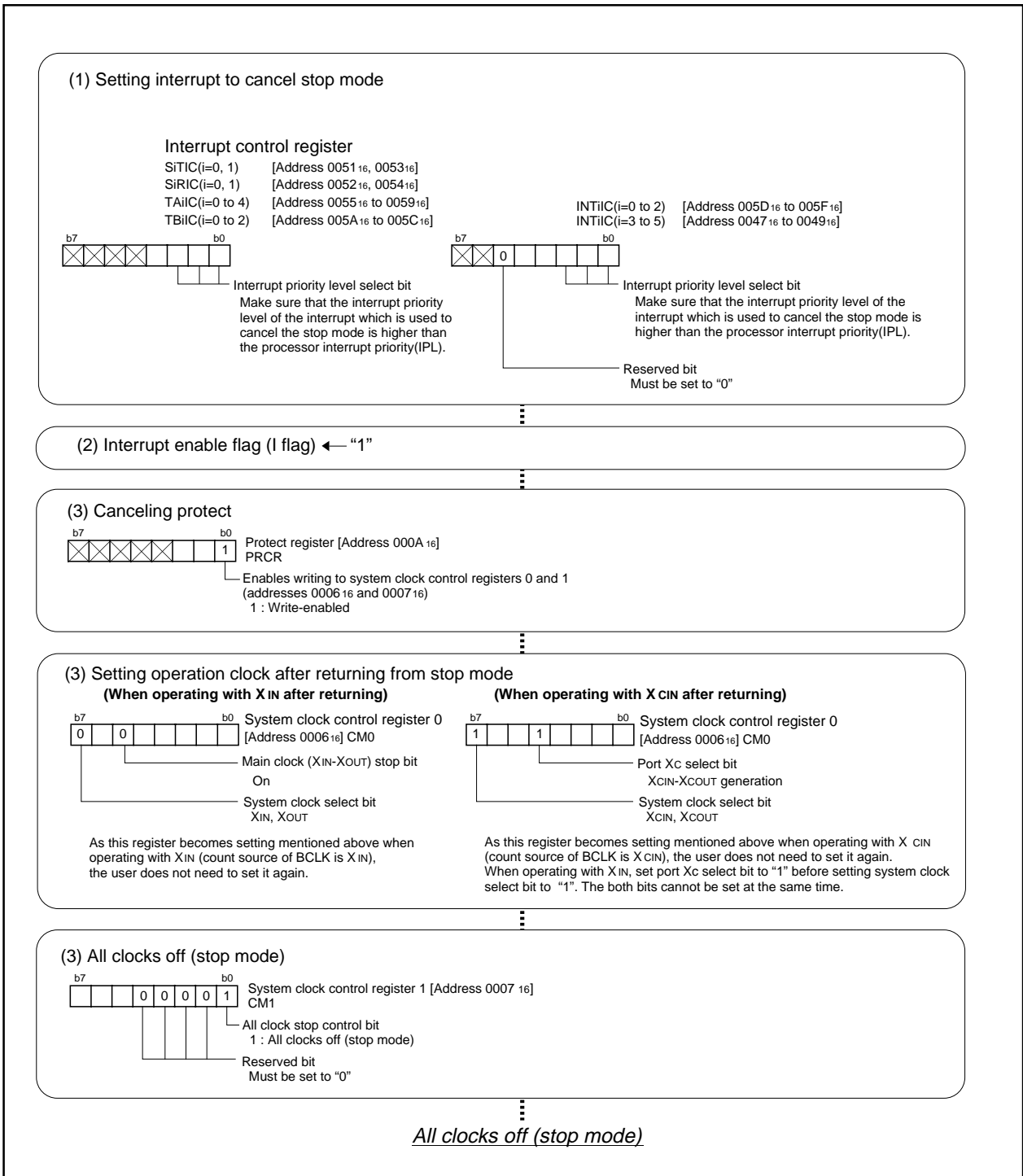


Figure 2.14.5. Example of stop mode set-up



### 2.14.3 Wait Mode Set-Up

Settings and operation for entering wait mode are described here.

- Operation
- (1) Enables the interrupt used for returning from wait mode.
  - (2) Sets the interrupt enable flag (I flag) to "1".
  - (3) Clears the protection and changes the content of the system clock control register.
  - (4) Executes the WAIT instruction.

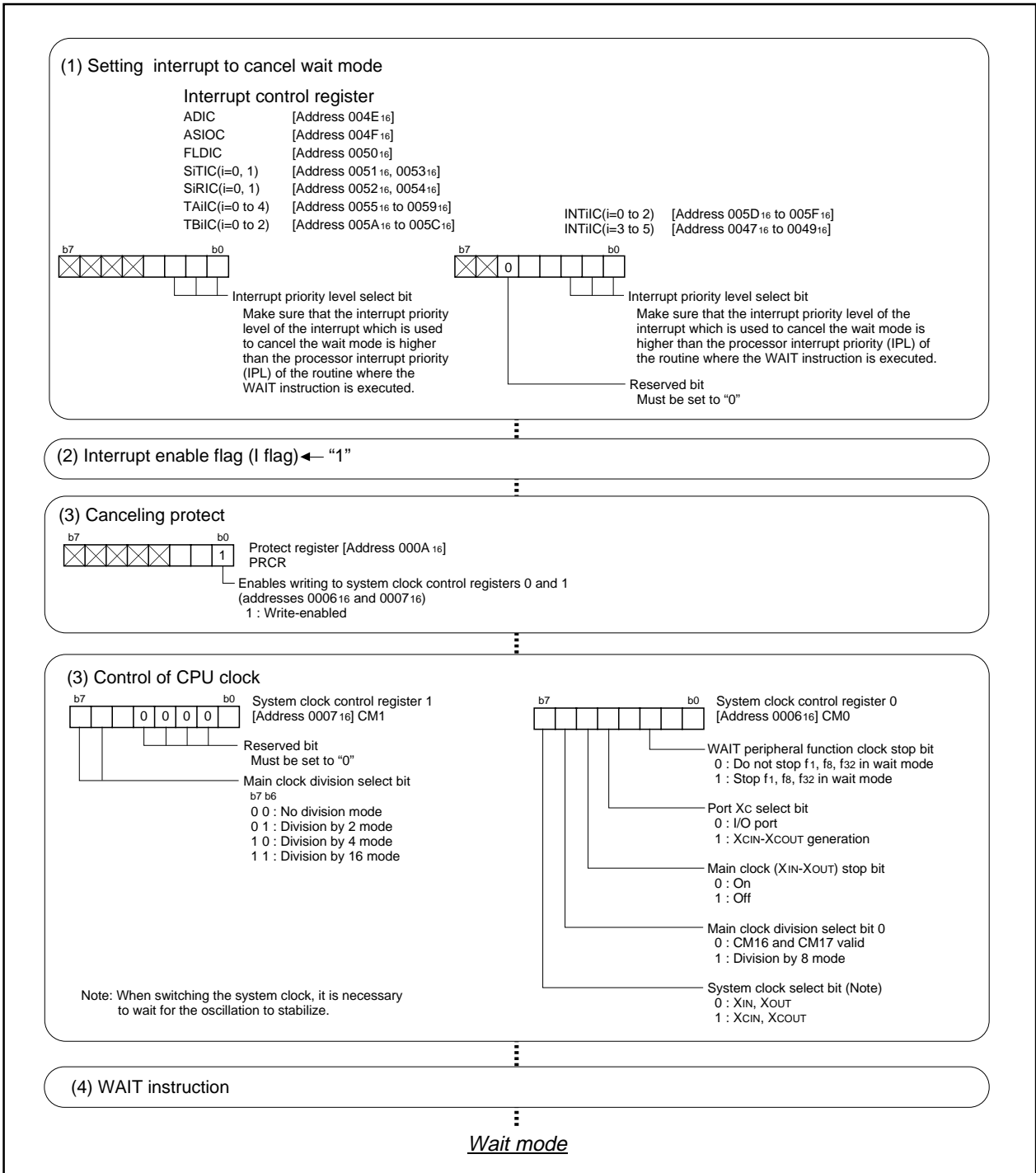


Figure 2.14.6. Example of wait mode set-up

### 2.14.4 Precautions in Power Control

- (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) Before the count source for BCLK can be changed from XIN to XCIN 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.

- (4) Suggestions to reduce power consumption

- **Ports**

The processor retains the state of each programmable I/O port even when it goes to wait mode or to stop mode. A current flows in active I/O ports. A pass current flows in input ports that float. When entering wait mode or stop mode, set non-used ports to input and stabilize the potential.

- (a) **A-D converter**

A current always flows in the VREF pin. When entering wait mode or stop mode, set the Vref connection bit to "0" so that no current flows into the VREF pin.

- (b) **D-A converter**

The processor retains the D-A state even when entering wait mode or stop mode. Disable the output from the D-A converter then work on the programmable I/O ports.

- (c) **Stopping peripheral functions**

In wait mode, stop non-used wait peripheral functions using the peripheral function clock stop bit.

- (d) **Switching the oscillation-driving capacity**

Set the driving capacity to "LOW" when oscillation is stable.

- (e) **External clock**

When using an external clock input for the CPU clock, set the main clock stop bit to "1". Setting the main clock stop bit to "1" causes the XOUT pin not to operate and the power consumption goes down (when using an external clock input, the clock signal is input regardless of the content of the main clock stop bit).

## 2.15 Programmable I/O Ports

### 2.15.1 Overview

Forty-eight programmable I/O ports and forty high-breakdown-voltage output ports are available. I/O pins also serve as I/O pins for built-in peripheral functions.

Each port has a direction register that defines the I/O direction and also has a port register for I/O data. In addition, each port has a pull-up control register that defines pull-up in terms of 4 bits. Ports P2, P3, and P40–P43 are high-breakdown-voltage P-channel open-drain output structure. These ports have no pull-up resistance.

The following is an overview of the programmable I/O ports:

#### (1) Writing to a port register

With the direction register set to output, the level of the written values from each relevant pin is output by writing to a port register. The output level conforms to CMOS output or P-channel open-drain output. "L" level of port which is built-in pull-down resistor is apply voltage to the VEE pin. Writing to the port register, with the direction register set to input, inputs a value to the port register, but nothing is output to the relevant pins. The output level remains floating.

#### (2) Reading a port register

With the direction register set to output, reading a port register takes out the content of the port register, not the content of the pin. When the FLD controller is used, reading the port register takes out FLD output. With the direction register set to input, reading the port register takes out the content of the pin.

#### (3) Exclusive high-breakdown-voltage output port

There are 40 exclusive output Ports: P0 to P2, P5 and P6.

All ports have structure of high-breakdown-voltage P-channel open drain output. Exclusive output ports except P2 have built-in pull-down resistance.

#### (4) Setting pull-up

The pull-up control bit allows setting of the pull-up, in terms of 4 bits, either in use or not in use. For the four bits chosen, pull-up is effective only in the ports whose direction register is set to input. Pull-up is not effective in ports whose direction register is set to output.

Do not set pull-up of corresponding pin when XCIN/XCOUT is set or a port is used as A-D input.

**(5) I/O functions of built-in peripheral devices**

Table 2.15.1 shows relation between ports and I/O functions of built-in peripheral devices.

**Table 2.15.1. Relation between ports and I/O functions of built-in peripheral devices**

Port	Internal peripheral device I/O pins
P0 to P3	FLD controller output pins
P40 to P43	FLD controller output pins
P44 to P47	FLD controller output pins/UART0 I/O pins
P5, P6	FLD controller output pins
P70 to P72	Timer B0 to B2 input pins
P73	Timer A0 I/O pin
P74 to P77	Timer A1 to A4 input pins/UART1 I/O pins
P80 to P85	External interrupt input pins
P86, P87	Sub-clock input pins
P90 to P95	I/O pins of serial I/O with automatic transfer function
P96	D-A converter output pin/Clock I/O pin of serial I/O with automatic transfer function
P97	D-A converter output pin/ X IN division clock output pin / DIM signal output pin of FLD controller
P100 to P107	A-D converter input pins

**(6) Examples of working on non-used pins**

Table 2.15.2 contains examples of working on non-used pins. There are shown here for mere examples. In practical use, make suitable changes and perform sufficient evaluation in compliance with you application.

**(a) Single-chip mode**

**Table 2.15.2. Examples of working on unused pins in single-chip mode**

Pin name	Connection
Ports P3, P4, P7 to P10	After setting for input mode, connect every pin to V <sub>SS</sub> or V <sub>CC</sub> via a resistor; or after setting for output mode, leave these pins open. (Note 1)
Ports P0 to P2, P5, P6	Open
XOUT (Note 2), V <sub>EE</sub>	Open
AV <sub>SS</sub> , V <sub>REF</sub>	Connect to V <sub>SS</sub>

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 X IN pin.

**(7) Registers related to the programmable I/O ports**

Figure 2.15.1 shows the memory map of programmable I/O ports-related registers, and Figures 2.15.2 to 2.15.4 show programmable I/O ports-related registers.

0359 <sub>16</sub>	P2 FLD/port switch register (P2FPR)
035A <sub>16</sub>	P3 FLD/port switch register (P3FPR)
035B <sub>16</sub>	P4 FLD/port switch register (P4FPR)
035C <sub>16</sub>	P5 digit output set register (P5DOR)
035D <sub>16</sub>	P6 digit output set register (P6DOR)
≈	≈
03E0 <sub>16</sub>	Port P0 (P0)
03E1 <sub>16</sub>	Port P1 (P1)
03E2 <sub>16</sub>	
03E3 <sub>16</sub>	
03E4 <sub>16</sub>	Port P2 (P2)
03E5 <sub>16</sub>	Port P3 (P3)
03E6 <sub>16</sub>	
03E7 <sub>16</sub>	Port P3 direction register (PD3)
03E8 <sub>16</sub>	Port P4 (P4)
03E9 <sub>16</sub>	Port P5 (P5)
03EA <sub>16</sub>	Port P4 direction register (PD4)
03EB <sub>16</sub>	
03EC <sub>16</sub>	Port P6 (P6)
03ED <sub>16</sub>	Port P7 (P7)
03EE <sub>16</sub>	
03EF <sub>16</sub>	Port P7 direction register (PD7)
03F0 <sub>16</sub>	Port P8 (P8)
03F1 <sub>16</sub>	Port P9 (P9)
03F2 <sub>16</sub>	Port P8 direction register (PD8)
03F3 <sub>16</sub>	Port P9 direction register (PD9)
03F4 <sub>16</sub>	Port P10 (P10)
03F5 <sub>16</sub>	
03F6 <sub>16</sub>	Port P10 direction register (PD10)
≈	≈
03FC <sub>16</sub>	
03FD <sub>16</sub>	Pull-up control register 0 (PUR0)
03FE <sub>16</sub>	Pull-up control register 1 (PUR1)
03FF <sub>16</sub>	

**Figure 2.15.1. Memory map of programmable I/O ports-related registers**

Programmable I/O Ports

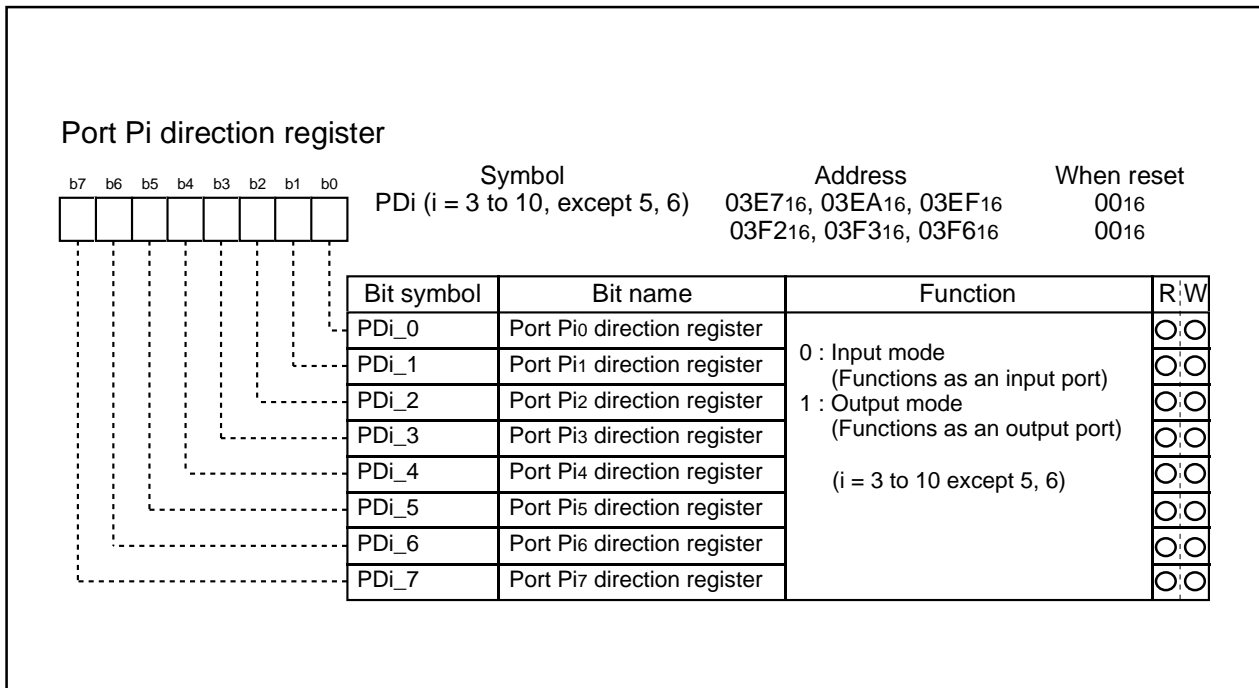


Figure 2.15.2. Programmable I/O ports-related registers (1)

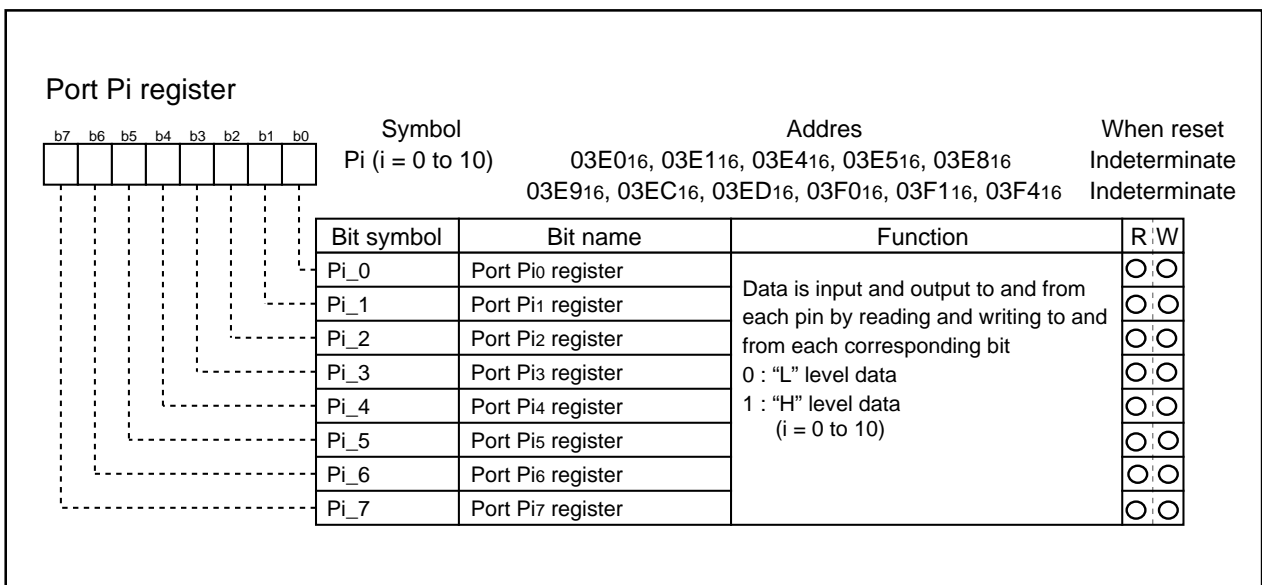


Figure 2.15.3. Programmable I/O ports-related registers (2)

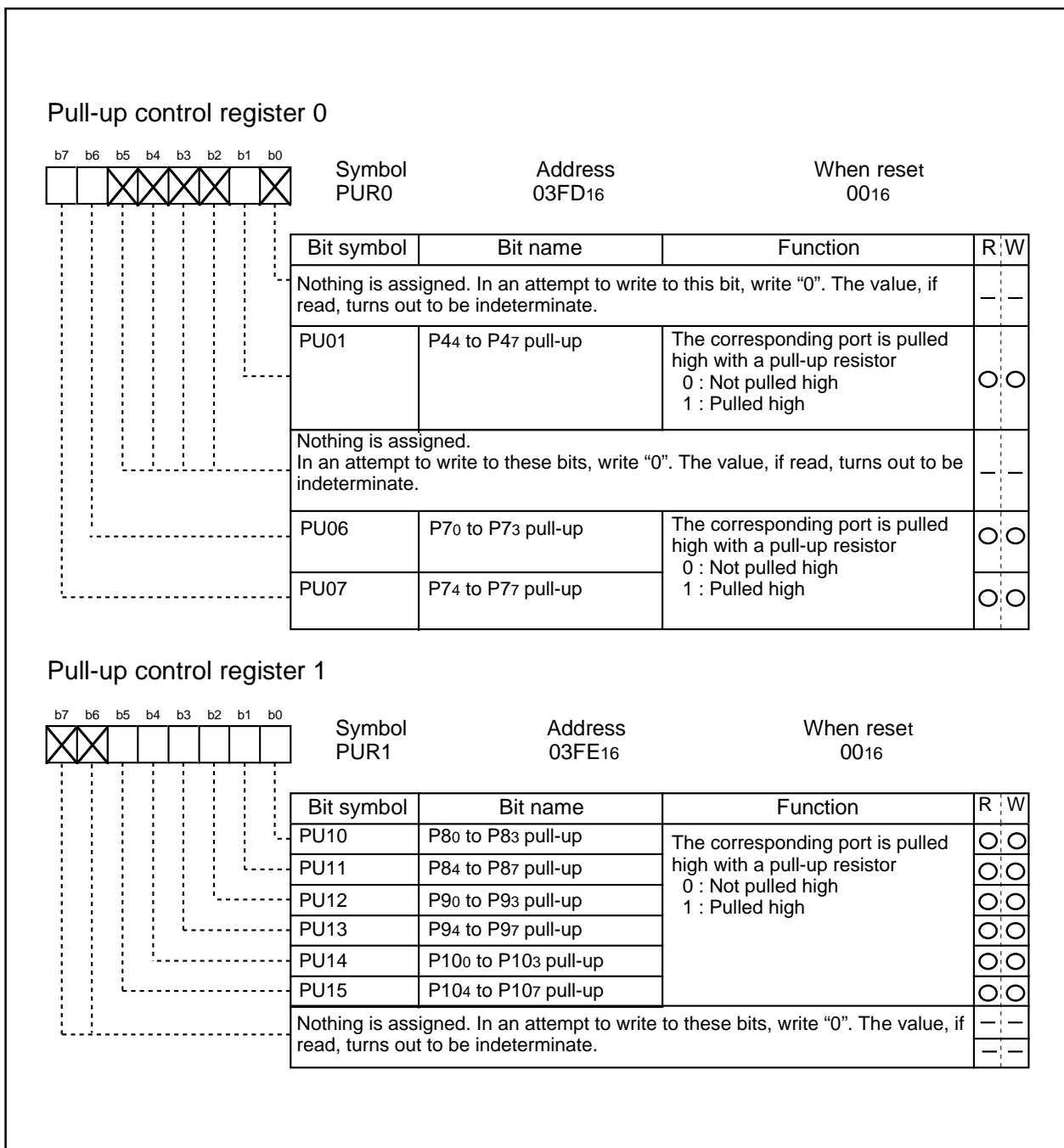


Figure 2.15.4. Programmable I/O ports-related registers (3)

## Chapter 3

---

Examples of Peripheral functions Applications



This chapter presents applications in which peripheral functions built in the M30218 are used. They are shown here as examples. In practical use, make suitable changes and perform sufficient evaluation. For basic use, see Chapter 2 Peripheral Functions Usage.

Here follows the list of applications that appear in this chapter.

- 3.1 Long-period timers ..... P388
- 3.2 Variable-period variable-duty PWM output ..... P392
- 3.3 Delayed one-shot output ..... P396
- 3.4 Buzzer output ..... P400
- 3.5 Solution for external interrupt pins shortage ..... P402
- 3.6 Memory to memory DMA transfer ..... P404
- 3.7 Controlling power using stop mode ..... P408
- 3.8 Controlling power using wait mode ..... P412

This page kept blank for layout purposes.

## Timer A Applications

## 3.1 Long-Period Timers

**Overview** In this process, Timer A0 and Timer A1 are connected to make a 16-bit timer with a 16-bit prescaler. Figure 3.1.1 shows the operation timing, Figure 3.1.2 shows the connection diagram, and Figures 3.1.3 and 3.1.4 show the set-up procedure.

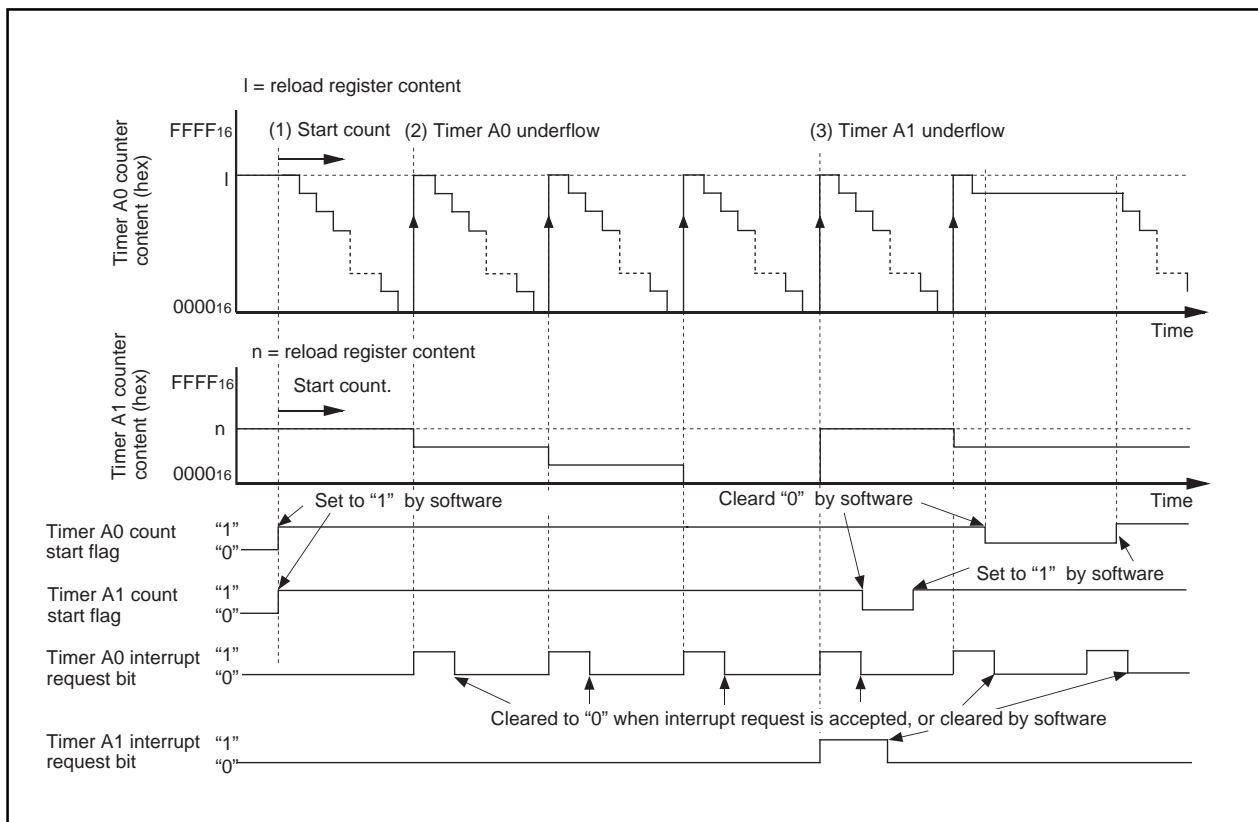
Use the following peripheral functions:

- Timer mode of timer A
- Event counter mode of timer A

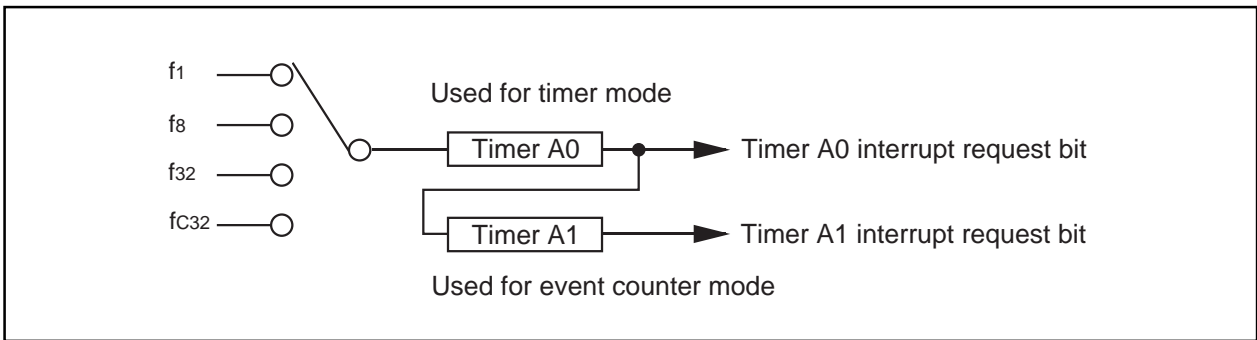
## Specifications

- (1) Set timer A0 to timer mode, and set timer A1 to event counter mode.
- (2) Perform a count on count source f1 using timer A0 to count for 1 ms, and perform a count on timer A0 using timer A1 to count for 1 second.
- (3) Connect a 10-MHz oscillator to XIN.

- Operation**
- (1) Setting the count start flag to "1" causes the counter to begin counting. The counter of timer A0 performs a down count on count source f1.
  - (2) If the counter of timer A0 underflows, the counter reloads the content of the reload register and continues counting. At this time, the timer A0 interrupt request bit goes to "1". The counter of timer A1 performs a down count on underflows in timer A0.
  - (3) If the counter of timer A1 underflows, the counter reloads the content of the reload register and continues counting. At this time, the timer A1 interrupt request bit goes to "1".



**Figure 3.1.1. Operation timing of long-period timers**



**Figure 3.1.2. Connection diagram of long-period timers**

Timer A Applications

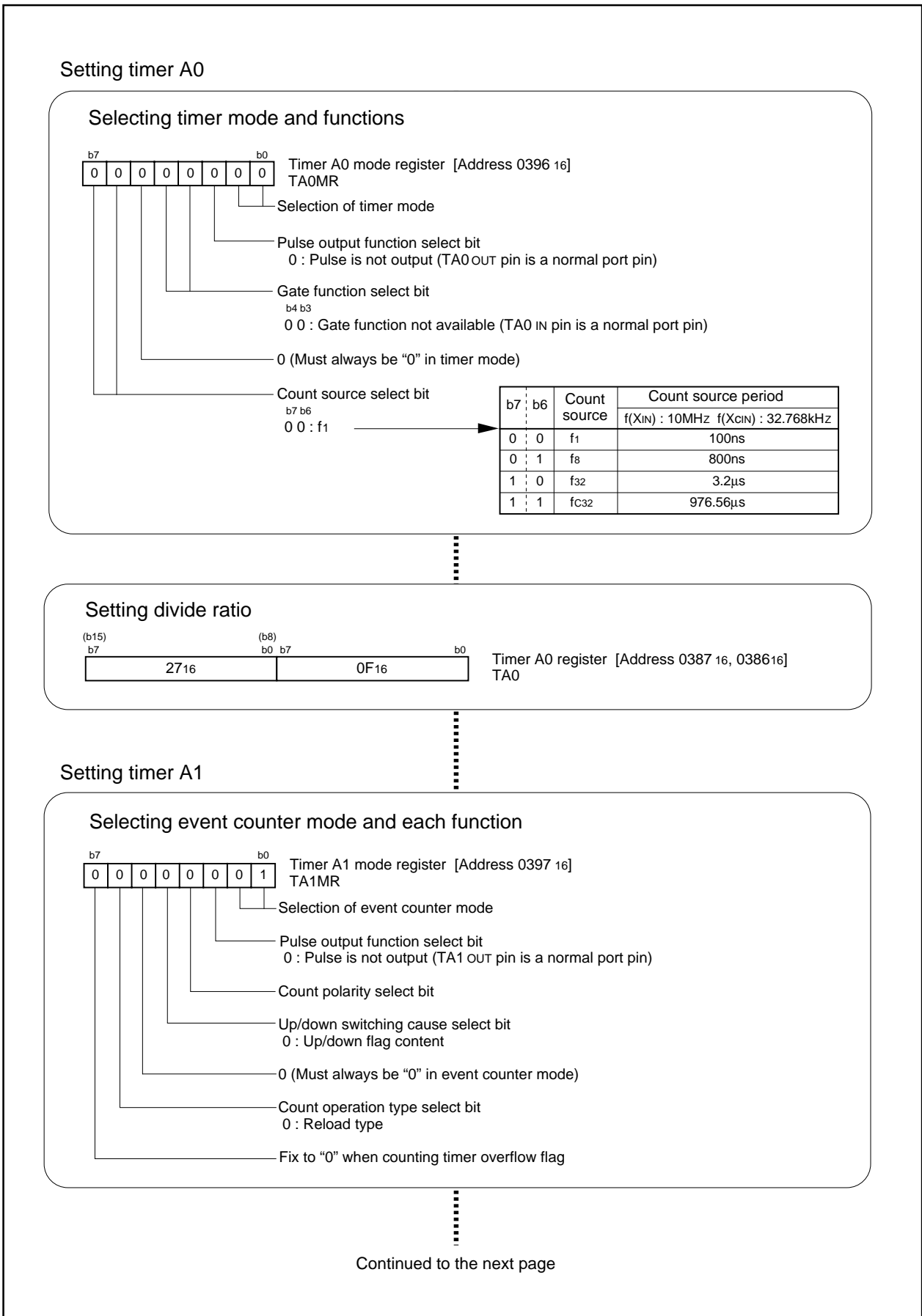


Figure 3.1.3. Set-up procedure of long-period timers (1)

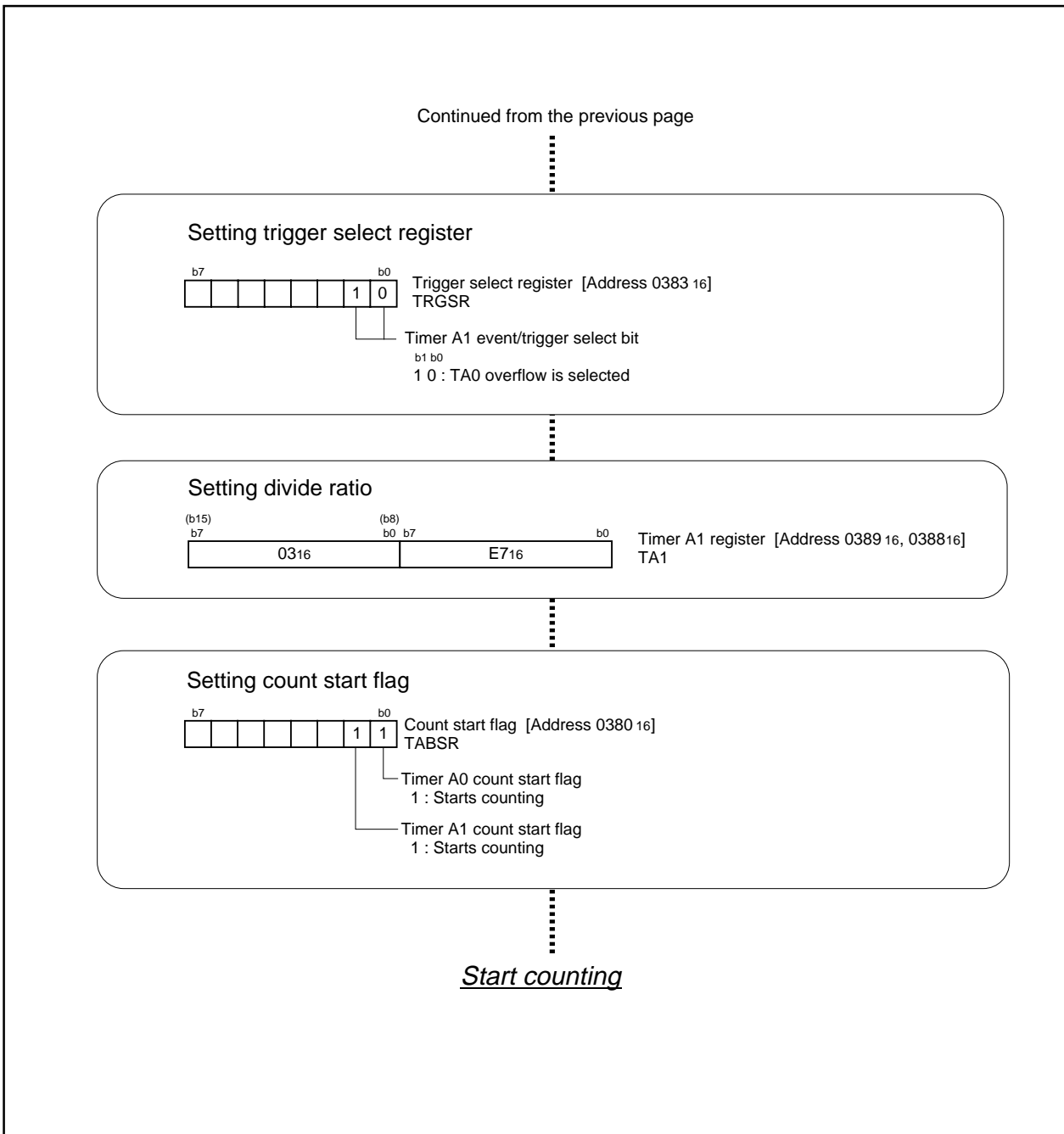


Figure 3.1.4. Set-up procedure of long-period timers (2)

### 3.2 Variable-Period Variable-Duty PWM Output

**Overview** In this process, Timer A0 and A1 are used to generate variable-period, variable-duty PWM output. Figure 3.2.1 shows the operation timing, Figure 3.2.2 shows the connection diagram, and Figures 3.2.3 and 3.2.4 show the set-up procedure.

Use the following peripheral functions:

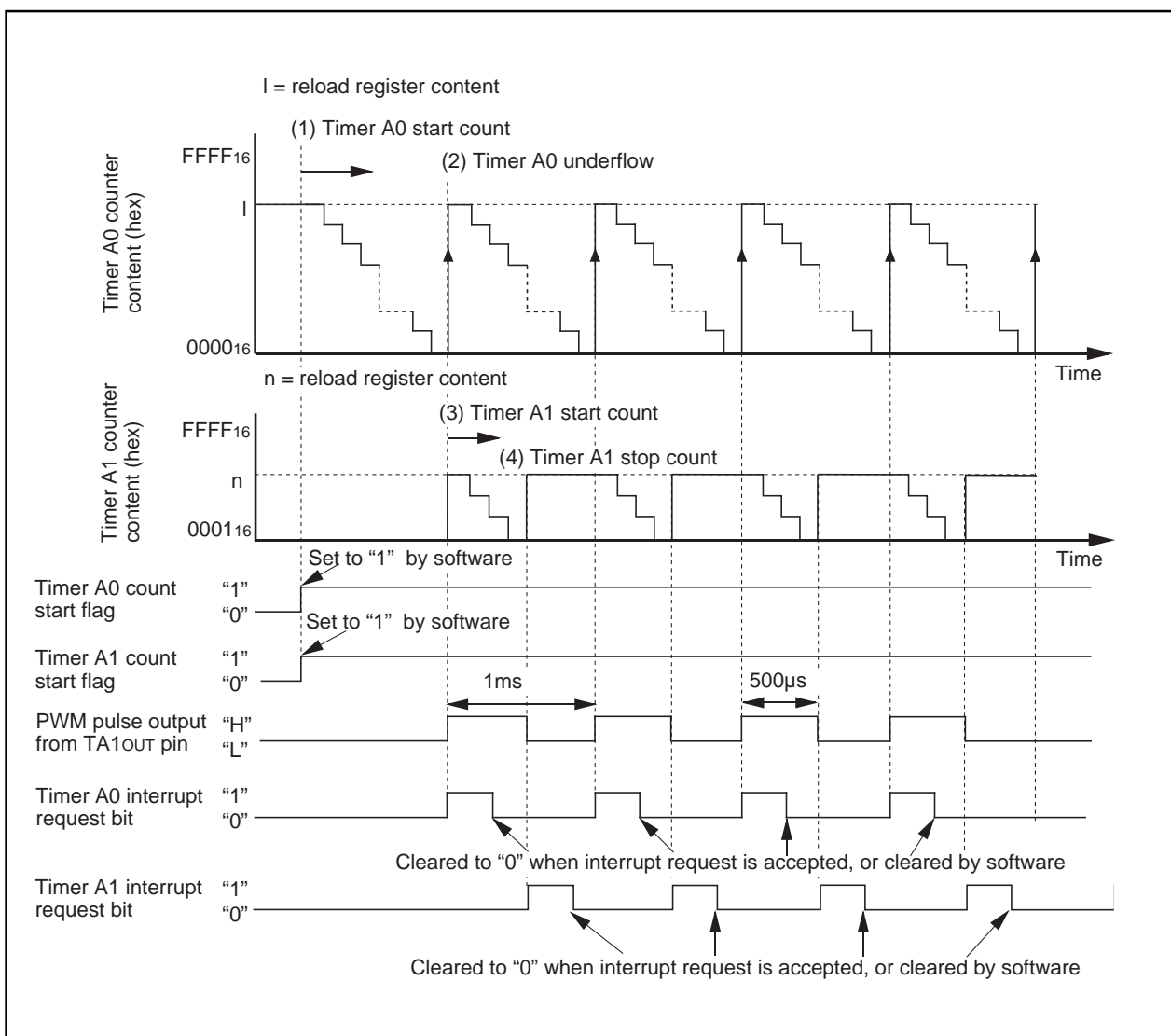
- Timer mode of timer A
- One-shot timer mode of timer A

#### Specifications

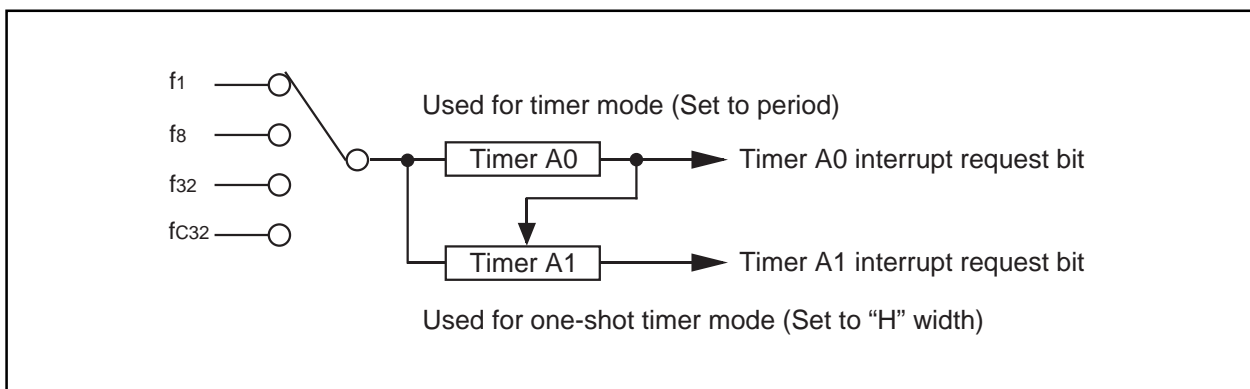
- (1) Set timer A0 in timer mode, and set timer A1 in one-shot timer mode with pulse-output function.
- (2) Set 1 ms, the PWM period, to timer A0. Set 500  $\mu$ s, the width of PWM "H" pulse, to timer A1. Both timer A0 and timer A1 use f1 for the count source.
- (3) Connect a 10-MHz oscillator to XIN.

**Operation**

- (1) Setting the count start flag to "1" causes the counter of timer A0 to begin counting. The counter of timer A0 performs a down count on count source f1.
- (2) If the counter of timer A0 underflows, the counter reloads the content of the reload register and continues counting. At this time, the timer A0 interrupt request bit goes to "1".
- (3) An underflow in timer A0 triggers the counter of timer A1 and causes it to begin counting. When the counter of timer A1 begins counting, the output level of the TA1OUT pin goes to "H".
- (4) As soon as the count of the counter of timer A1 becomes "0000<sub>16</sub>", the output level of TA1OUT pin goes to "L", and the counter reloads the content of the reload register and stops counting. At the same time, the timer A1 interrupt request bit goes to "1".



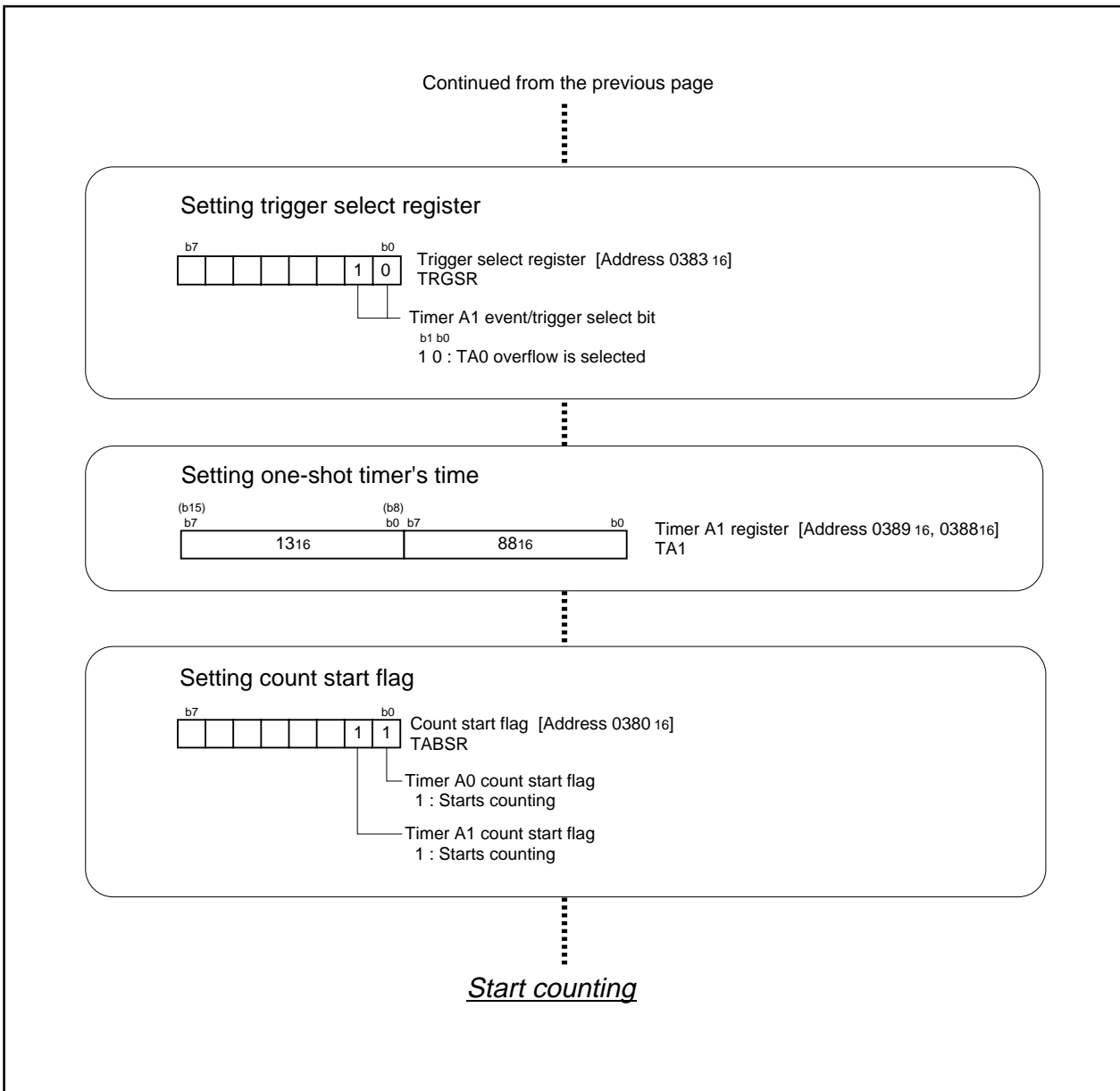
**Figure 3.2.1. Operation timing of variable-period variable-duty PWM output**



**Figure 3.2.2. Connection diagram of variable-period variable-duty PWM output**







**Figure 3.2.4. Set-up procedure of variable-period variable-duty PWM output (2)**

### 3.3 Delayed One-Shot Output

**Overview** The following are steps of outputting a pulse only once after a specified elapse since an external trigger is input. Figure 3.3.1 shows the operation timing, Figure 3.3.2 shows the connection diagram, and Figures 3.3.3 and 3.3.4 show the set-up procedure.

Use the following peripheral function:

- One-shot timer mode of timer A

#### Specifications

- (1) Set timer A0 in one-shot timer mode, and set timer A1 in one-shot timer mode with pulse-output function.
- (2) Set 1 ms, an interval before a pulse is output, in timer A0; and set 50  $\mu$ s, a pulse width, in timer A1. Both timer A0 and timer A1 use f1 for the count source.
- (3) Connect a 10-MHz oscillator to XIN.

- Operation**
- (1) Setting the trigger select bit to "1" and setting the count start flag to "1" enables the counter of timer A0 to count.
  - (2) If an effective edge, selected by use of the external trigger select bit, is input to the TA0IN pin, the counter begins a down count. The counter of timer A0 performs a down count on count source f1.
  - (3) As soon as the counter of timer A0 becomes "0000<sub>16</sub>", the counter reloads the content of the reload register and stops counting. At this time, the timer A0 interrupt request bit goes to "1".
  - (4) An underflow in timer A0 triggers the counter of timer A1 and causes it to begin counting. When timer A1 begins counting, the output level of the TA1OUT pin goes to "H".
  - (5) As soon as the counter of timer A1 becomes "0000<sub>16</sub>", the output level of the TA1OUT pin goes to "L", the counter reloads the content of the reload register, and stops counting. At this time, timer A1 interrupt request bit goes to "1".

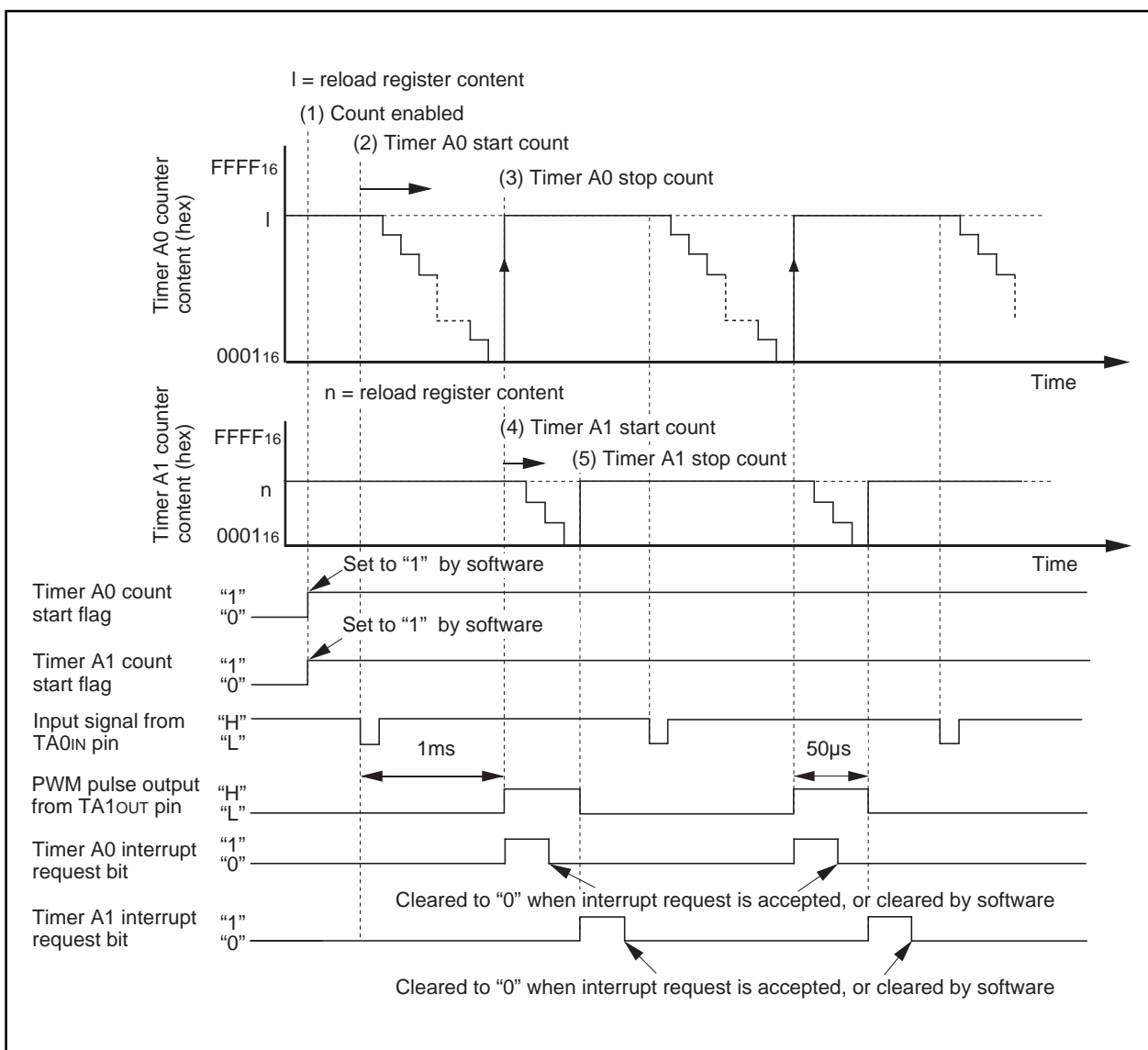


Figure 3.3.1. Operation timing of delayed one-shot output

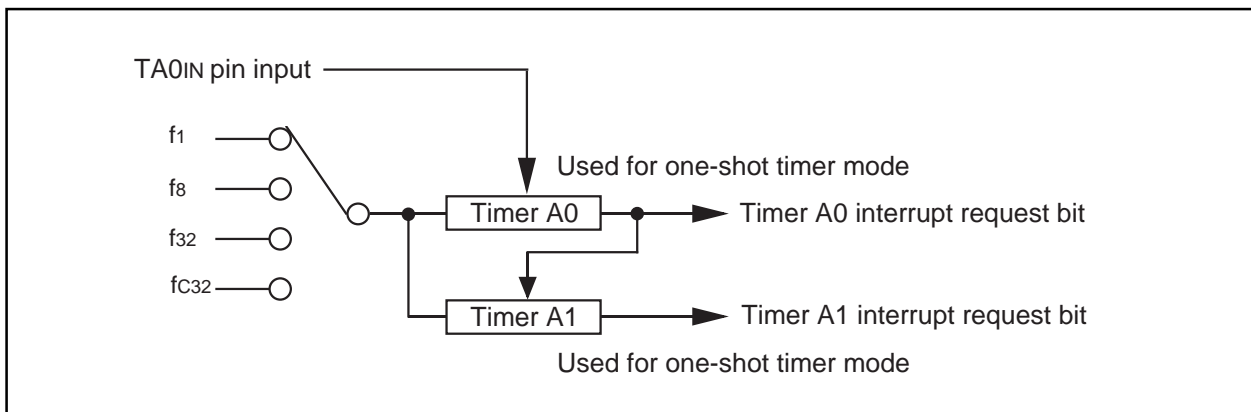


Figure 3.3.2. Connection diagram of delayed one-shot output



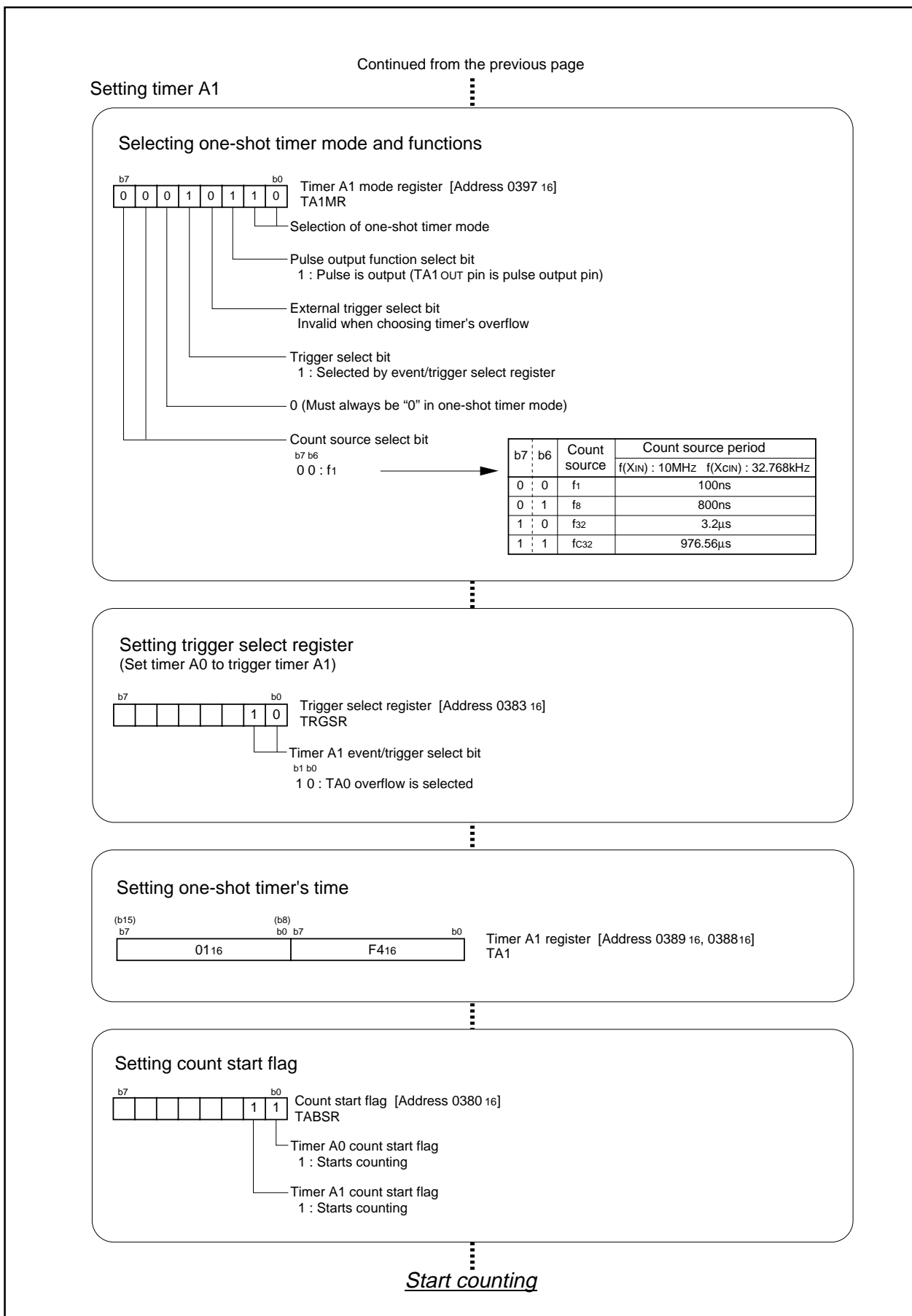


Figure 3.3.4. Set-up procedure of delayed one-shot output (2)

### 3.4 Buzzer Output

**Overview** The timer mode is used to make the buzzer ring. Figure 3.4.1 shows the operation timing, and Figure 3.4.2 shows the set-up procedure.

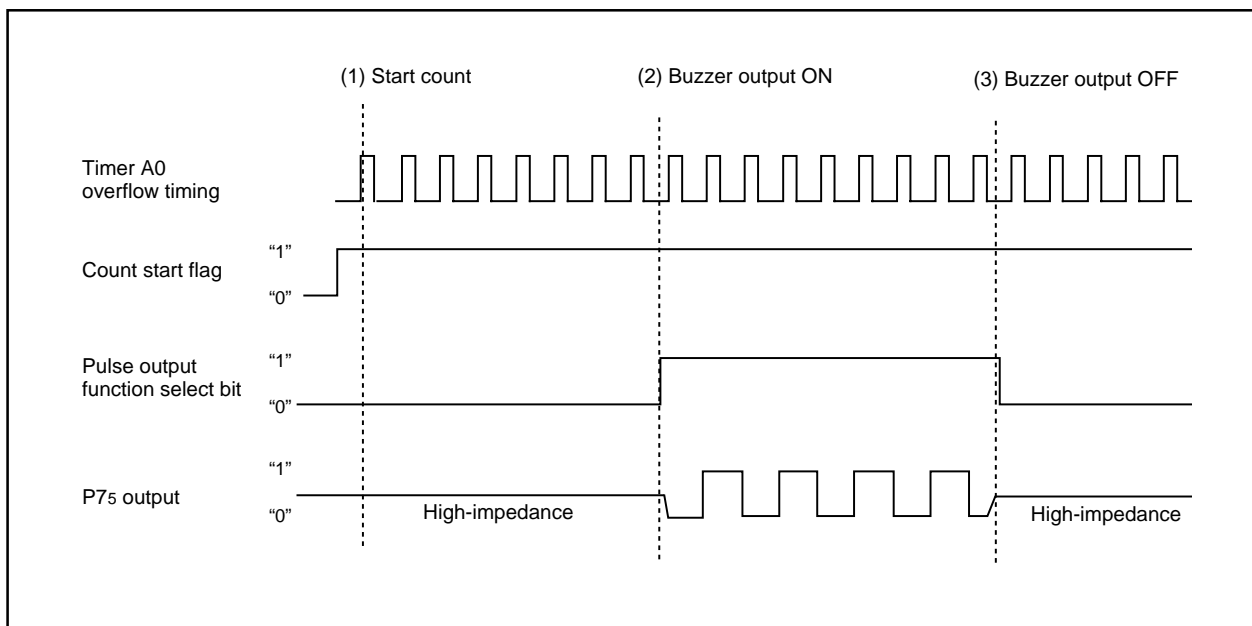
Use the following peripheral function:

- The pulse-outputting function in timer mode of timer A.

**Specifications**

- (1) Sound a 2-kHz buzz beep by use of timer A0.
- (2) Effect pull-up in the relevant port by use of a pull-up resistor. When the buzzer is off, set the port high-impedance, and stabilize the potential resulting from pulling up.
- (3) Connect a 10-MHz oscillator to XIN.

**Operation** (1) The microcomputer begins performing a count on timer A0. Timer A0 has disabled interrupts. (2) The microcomputer begins pulse output by setting the pulse output function select bit to "Pulse output effected". P75 changes into TA0OUT pin and outputs 2-kHz pulses. (3) The microcomputer stops outputting pulses by setting the pulse output function select bit to "Pulse output not effected". P75 goes to an input pin, and the output from the pin becomes high-impedance.



**Figure 3.4.1. Operation timing of buzzer output**

Timer A Applications

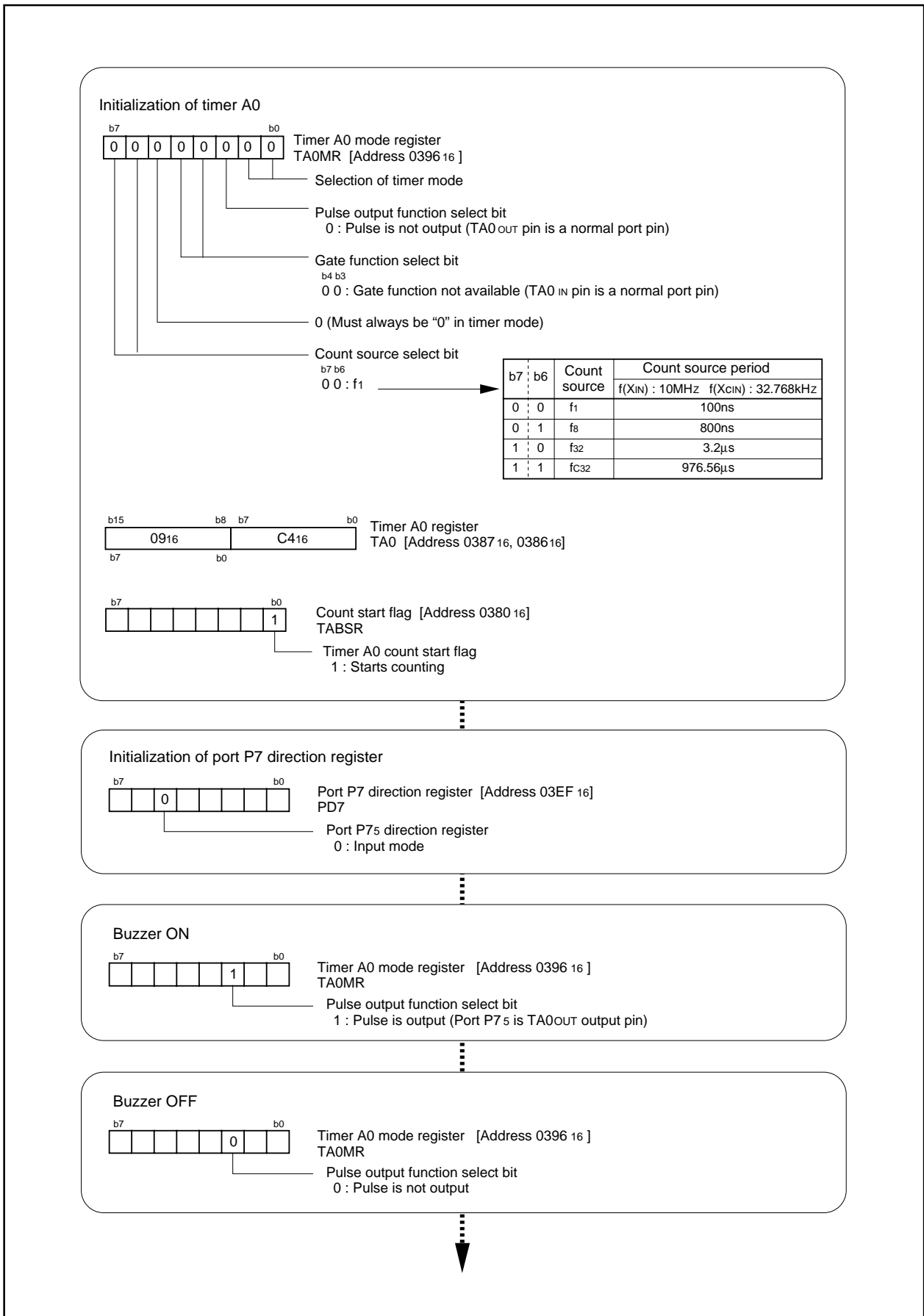


Figure 3.4.2. Set-up procedure of buzzer output



### 3.5 Solution for External Interrupt Pins Shortage

**Overview** The following are solution for external interrupt pins shortage. Figure 3.5.1 shows the set-up procedure.

Use the following peripheral function:

- Event counter mode of timer A

**Specifications**

(1) Inputting a falling edge to the TA0IN pin generates a timer A0 interrupt.

**Operation** (1) Set timer A0 to event counter mode, set timer to "0", and set interrupt priority levels in timer A0.  
(2) Inputting a falling edge to the TA0IN pin generates a timer A0 interrupt.

## Timer A Applications

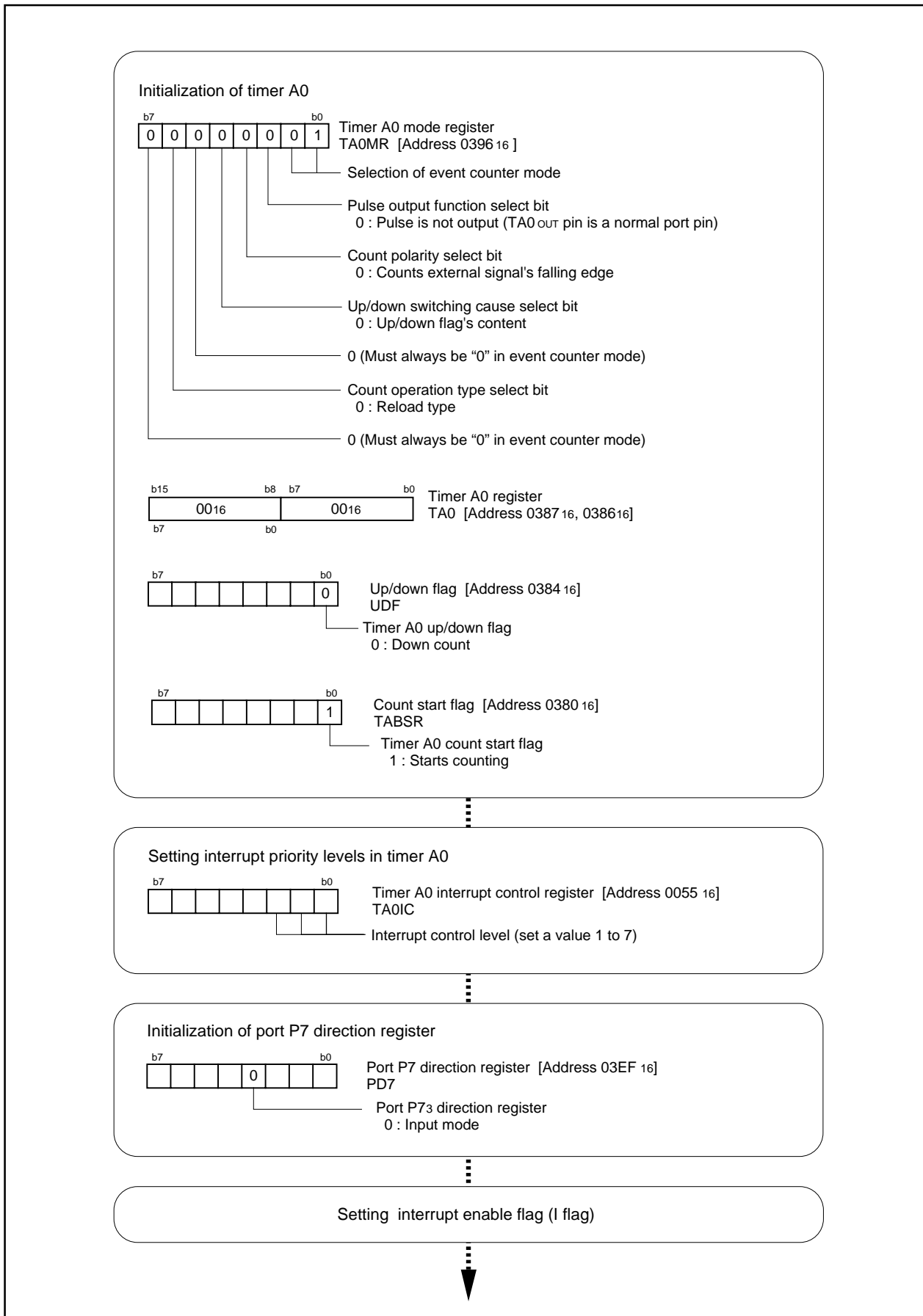


Figure 3.5.1. Set-up procedure of solution for a shortage of external interrupt pins

### 3.6 Memory to Memory DMA Transfer

**Overview** The following are steps for changing both source address and destination address to transfer data from memory to another. The DMA transfer utilizes the workings that assign a higher priority to the DMA0 transfer if transfer requests simultaneously occur in two DMA channels. Figure 3.6.1 shows the operation timing, Figure 3.6.2 shows the block diagram, and Figures 3.6.3 and 3.6.4 show the set-up procedure.

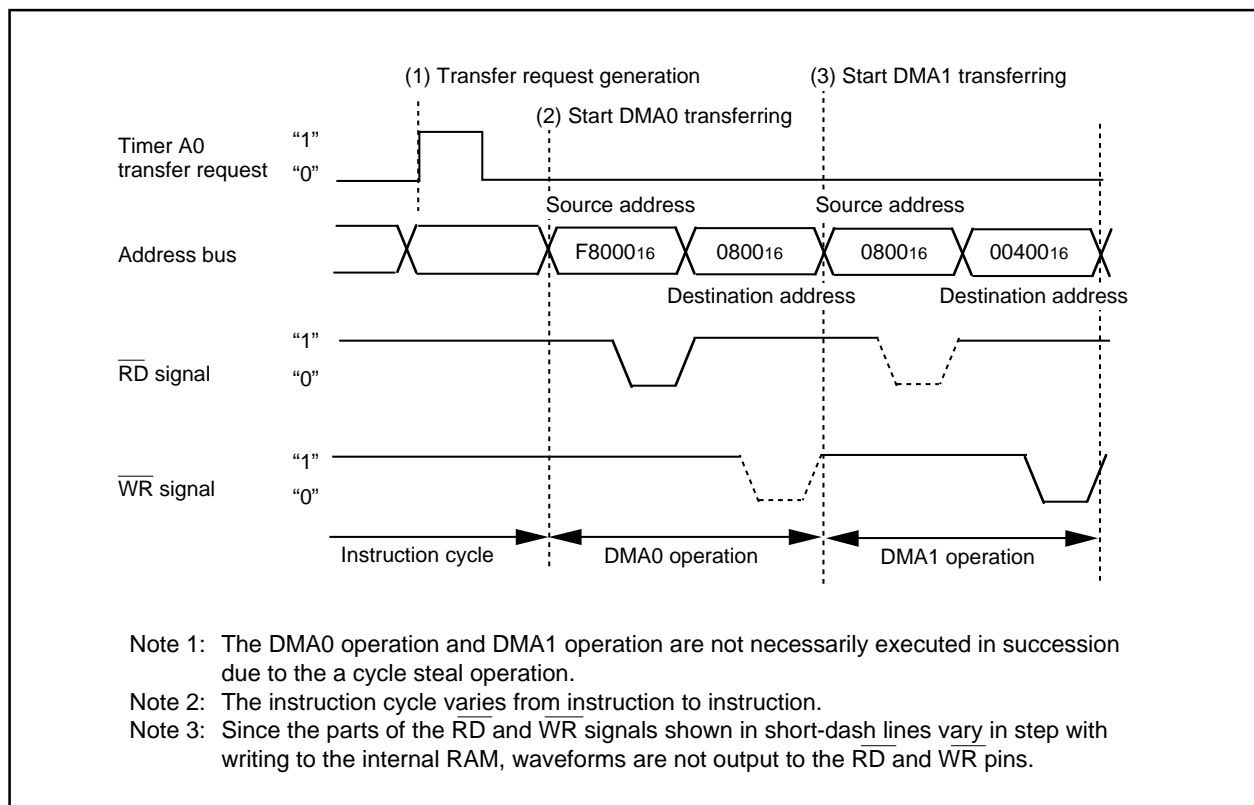
Use the following peripheral functions:

- Timer mode of timer A
- Two DMAC channels
- One-byte temporary RAM (address 0800<sub>16</sub>)

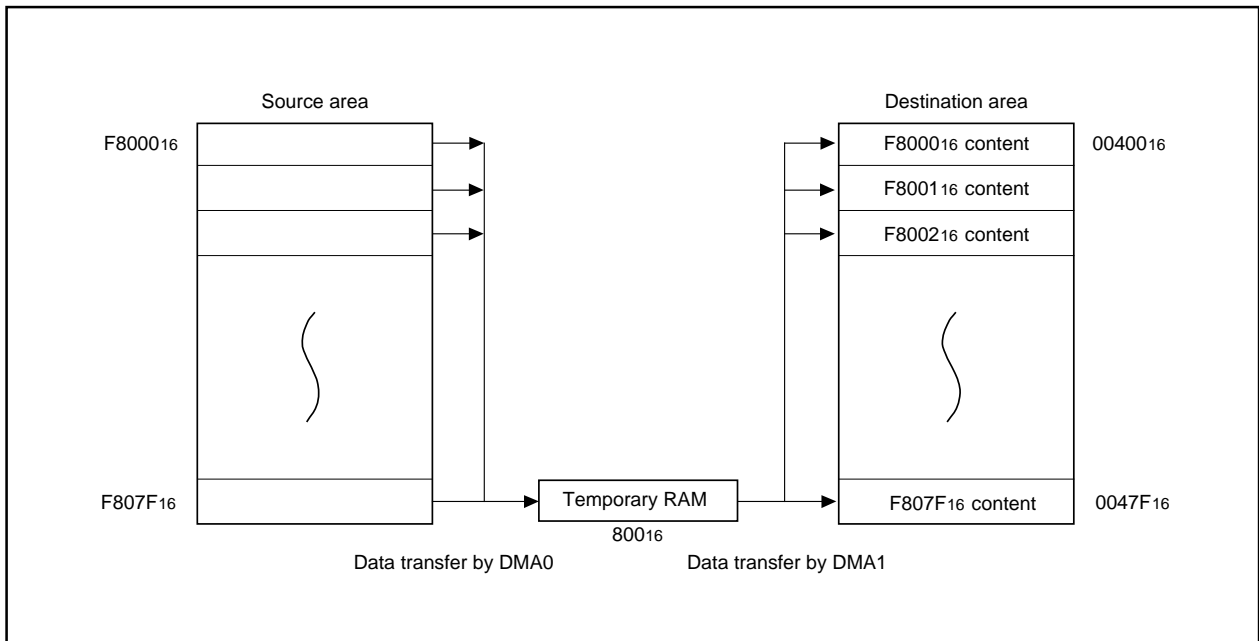
**Specifications**

- (1) Transfer the content of memory extending over 128 bytes from address F8000<sub>16</sub> to a 128-byte area starting from address 00400<sub>16</sub>. Transfer the content every time a timer A0 interrupt request occurs.
- (2) Use DMA0 for a transfer from the source to built-in memory, and DMA1 for a transfer from built-in memory to the destination.

- Operation**
- (1) A timer A interrupt request occurs. Though both a DMA0 transfer request and a DMA1 transfer request occur simultaneously, the former is executed first.
  - (2) DMA0 receives a transfer request and transfers data from the source to the built-in memory. At this time, the source address is incremented.
  - (3) Next, DMA1 receives a transfer request and transfers data involved from built-in memory to the destination. At this time, the destination address is incremented.



**Figure 3.6.1. Operation timing of memory to memory DMA transfer**



**Figure 3.6.2. Block diagram of memory to memory DMA transfer**

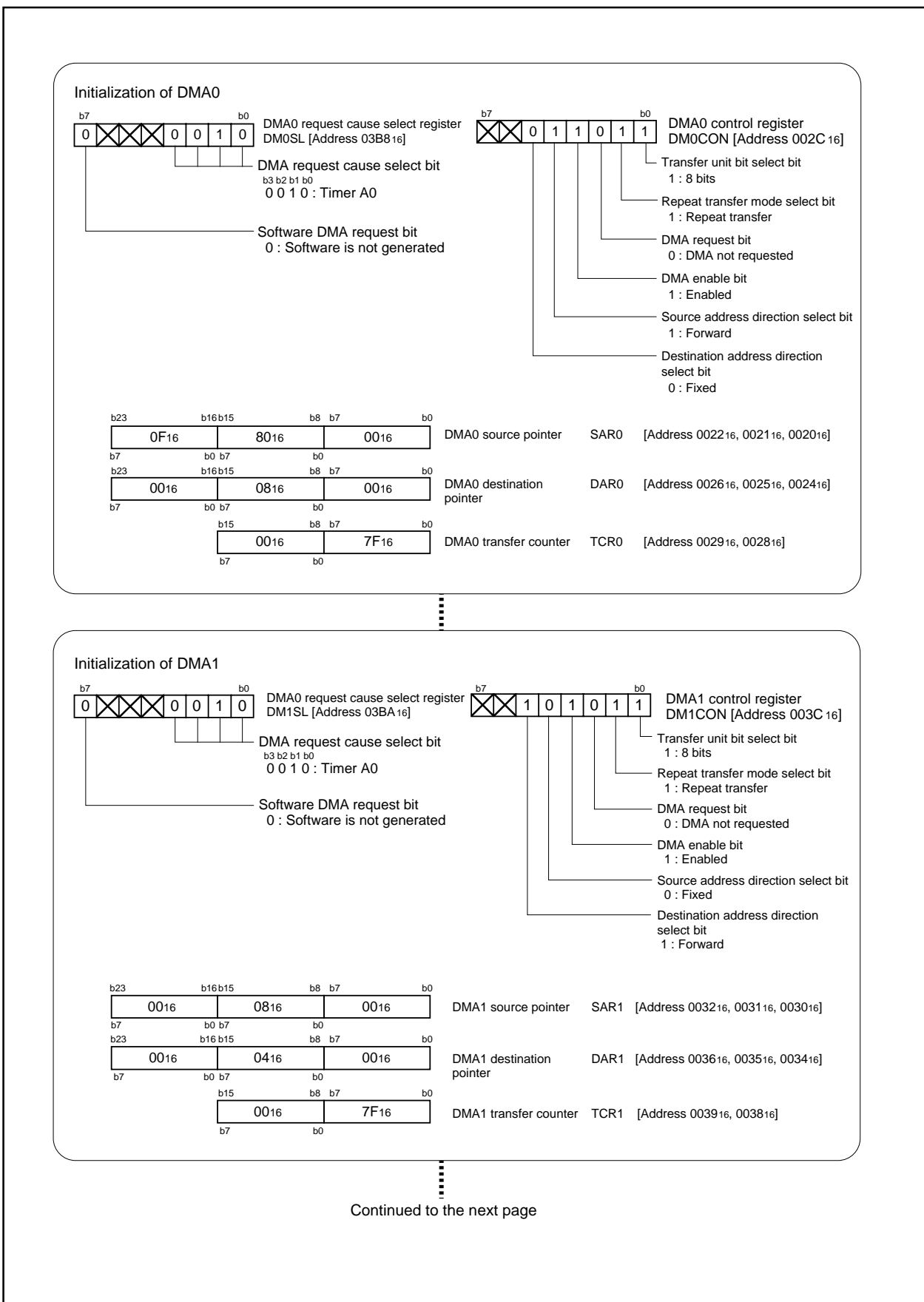
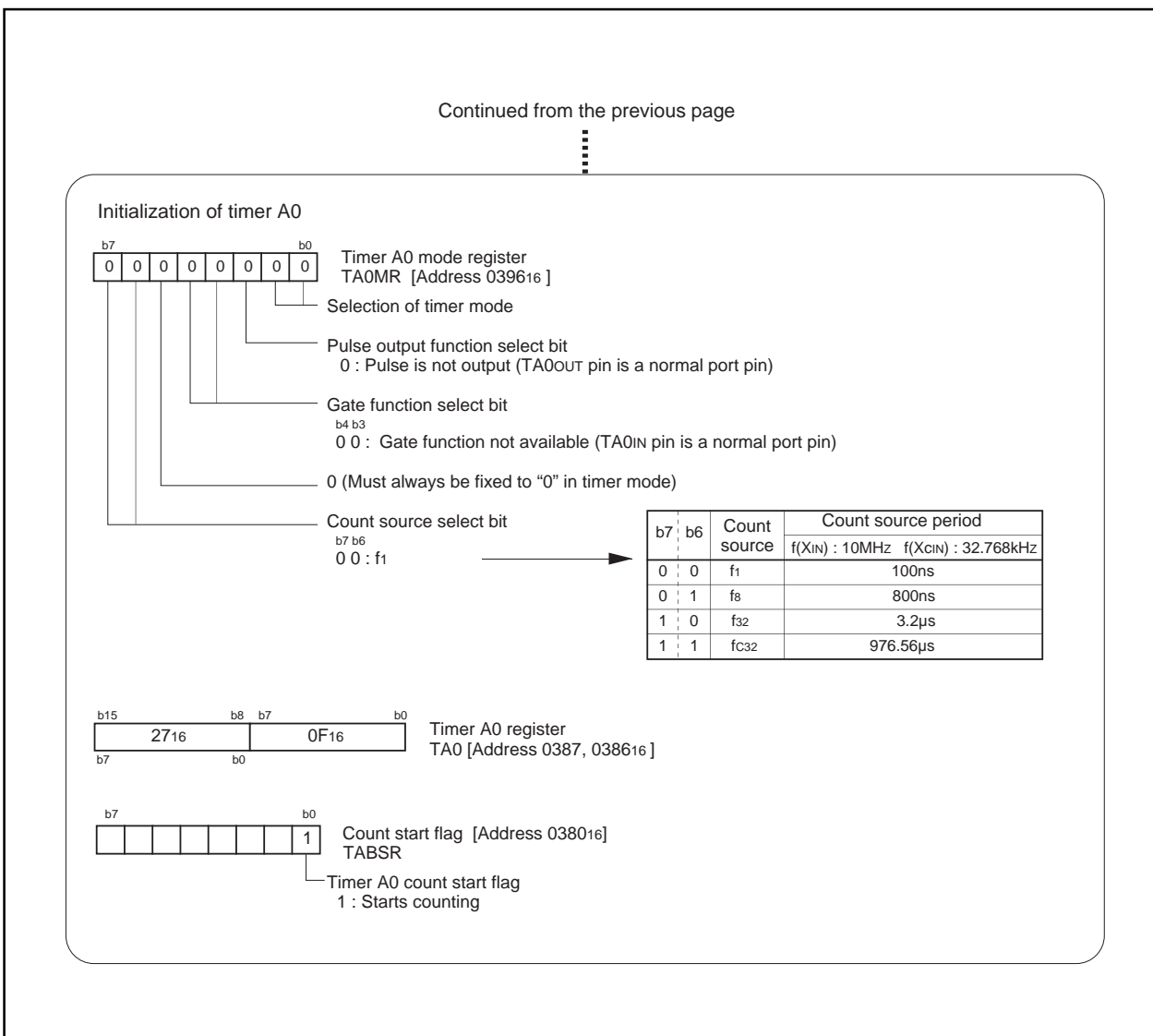


Figure 3.6.3. Set-up procedure of memory to memory DMA transfer (1)



**Figure 3.6.4. Set-up procedure of memory to memory DMA transfer (2)**

## Controlling Power Applications

### 3.7 Controlling Power Using Stop Mode

**Overview** The following are steps for controlling power using stop mode. Figure 3.7.1 shows the operation timing, Figure 3.7.2 shows an example of circuit, and Figures 3.7.3 and 3.7.4 show the set-up procedure.

Use the following peripheral functions:

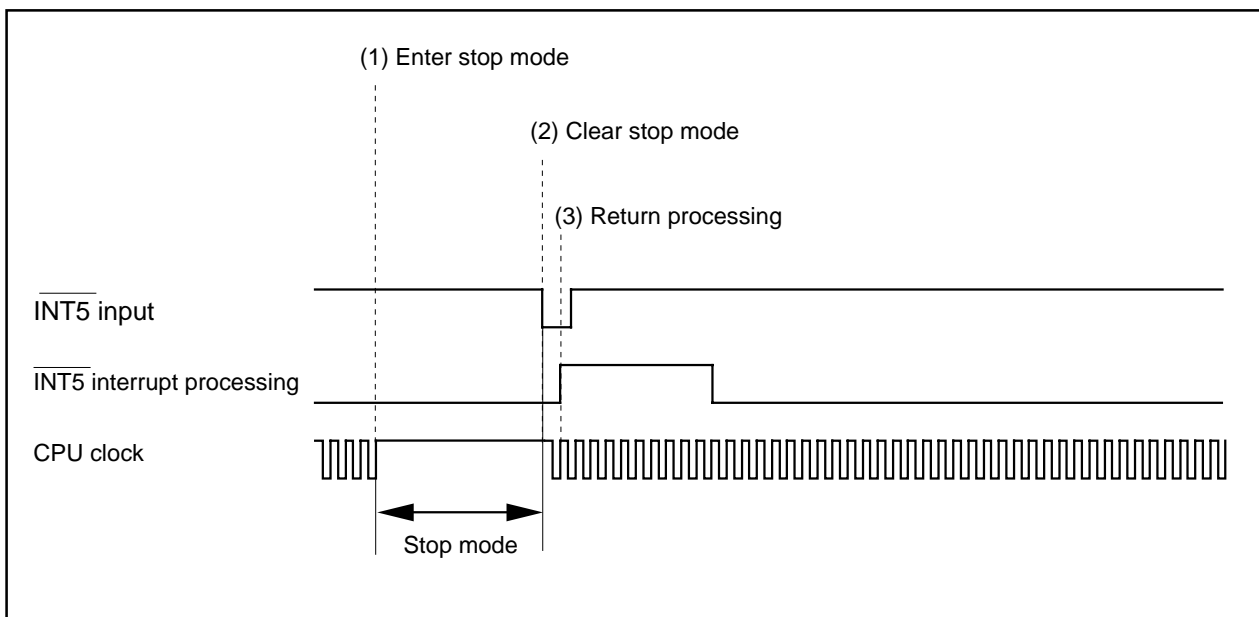
- $\overline{\text{INT5}}$  interrupt
- Stop mode

#### Specifications

- (1) Use  $\overline{\text{INT5}}$  for the INT interrupt. Use the P85/INT5 pin as an input pin.
- (2) When a  $\overline{\text{INT5}}$  interrupt request occurs, the stop mode is cleared.

#### Operation

- (1) Enable  $\overline{\text{INT5}}$  interrupt and set the pull-up function to the P85 pin.
- (2) Stop  $X_{IN}$  to enter the stop mode. Enable  $\overline{\text{INT5}}$  interrupt at this time.
- (3) When a  $\overline{\text{INT5}}$  interrupt request occurs by falling edge input to the P85 pin, the stop mode is cleared. Execute the return processing for the other interrupts, which are stopped, in the  $\overline{\text{INT5}}$  interrupt processing and others.



**Figure 3.7.1. Operation timing of controlling power using stop mode**

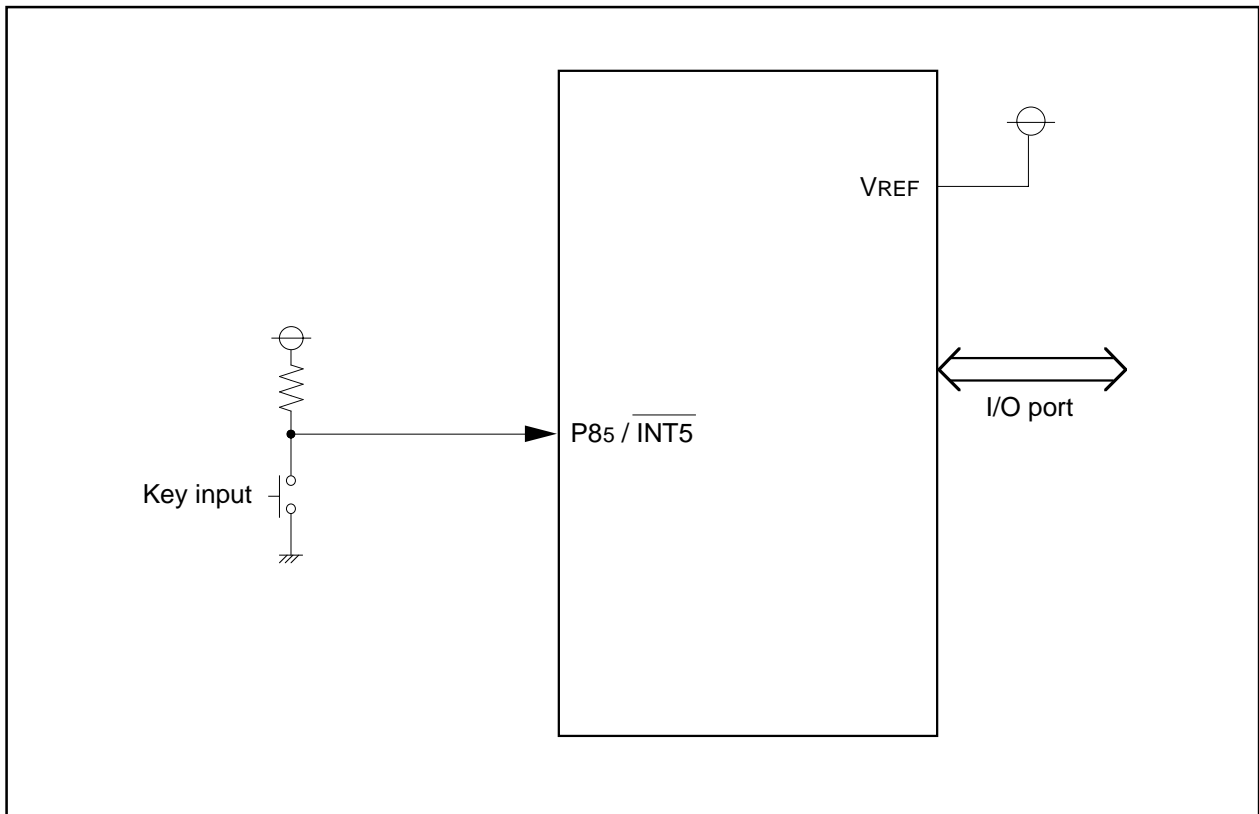


Figure 3.7.2. Example of circuit of controlling power using stop mode



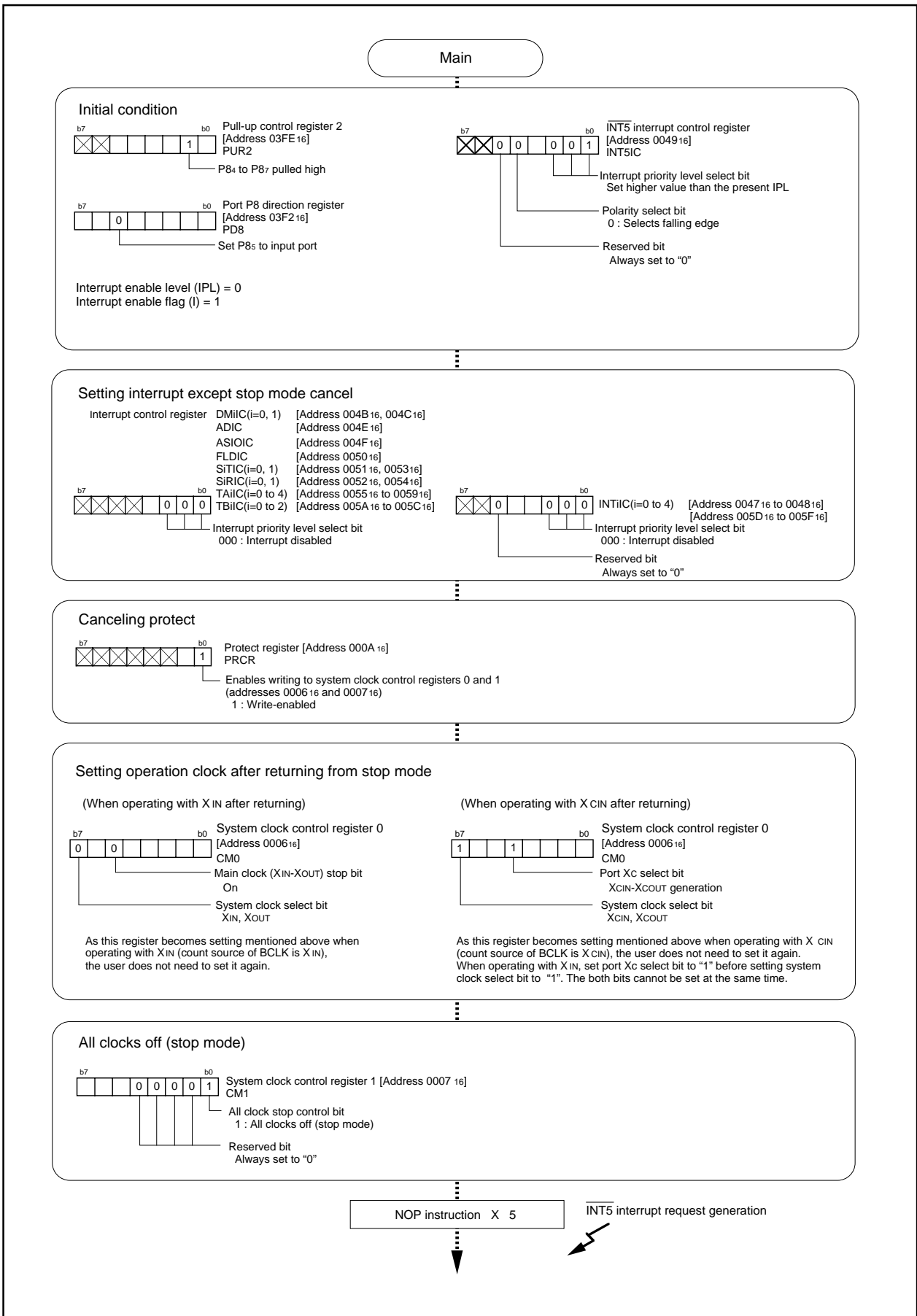


Figure 3.7.3. Set-up procedure of controlling power using stop mode (1)

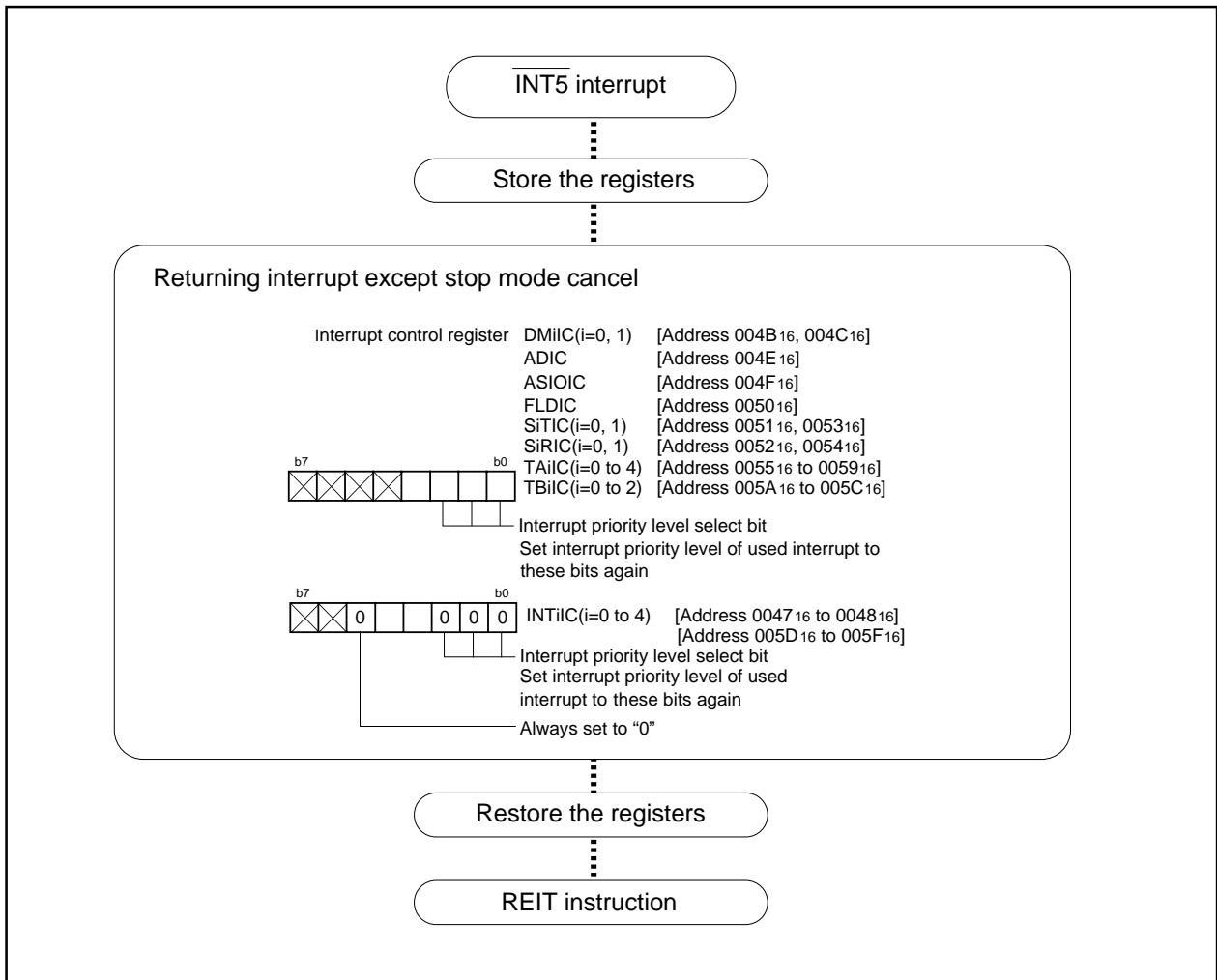


Figure 3.7.4. Set-up procedure of controlling power using stop mode (2)

## Controlling Power Applications

### 3.8 Controlling Power Using Wait Mode

**Overview** The following are steps for controlling power using wait mode. Figure 3.8.1 shows the operation timing, and Figures 3.8.2 to 3.8.4 show the set-up procedure.

Use the following peripheral functions:

- Timer mode of timer B
- Wait mode

A flag named "F-WIT" is used in the set-up procedure. The purpose of this flag is to decide whether or not to clear wait mode. If F\_WIT = "1" in the main program, the wait mode is entered; if F\_WIT = "0", the wait mode is cleared.

#### Specifications

- (1) Connect a 32.768-kHz oscillator to XCIN to serve as the timer count source. As interrupts occur every one second, which is a count the timer reaches, the controller returns from wait mode and count the clock using a program.
- (2) Clear wait mode if a  $\overline{\text{INT0}}$  interrupt request occurs.

#### Operation

- (1) Switch the system clock from XIN to XCIN to get low-speed mode.
- (2) Stop XIN and enter wait mode. In this instance, enable the timer B2 interrupt and the  $\overline{\text{INT0}}$  interrupt.
- (3) When a timer B2 interrupt request occurs (at 1-second intervals), start supplying the BCLK from XCIN. At this time, count the clock within the routine that handles the timer B2 interrupts and enter wait mode again.
- (4) If a  $\overline{\text{INT0}}$  interrupt occurs, start supplying the BCLK from XCIN. Start the XIN oscillation within the  $\overline{\text{INT0}}$  interrupt, and switch the BCLK count source to XIN after oscillation is stabilized.

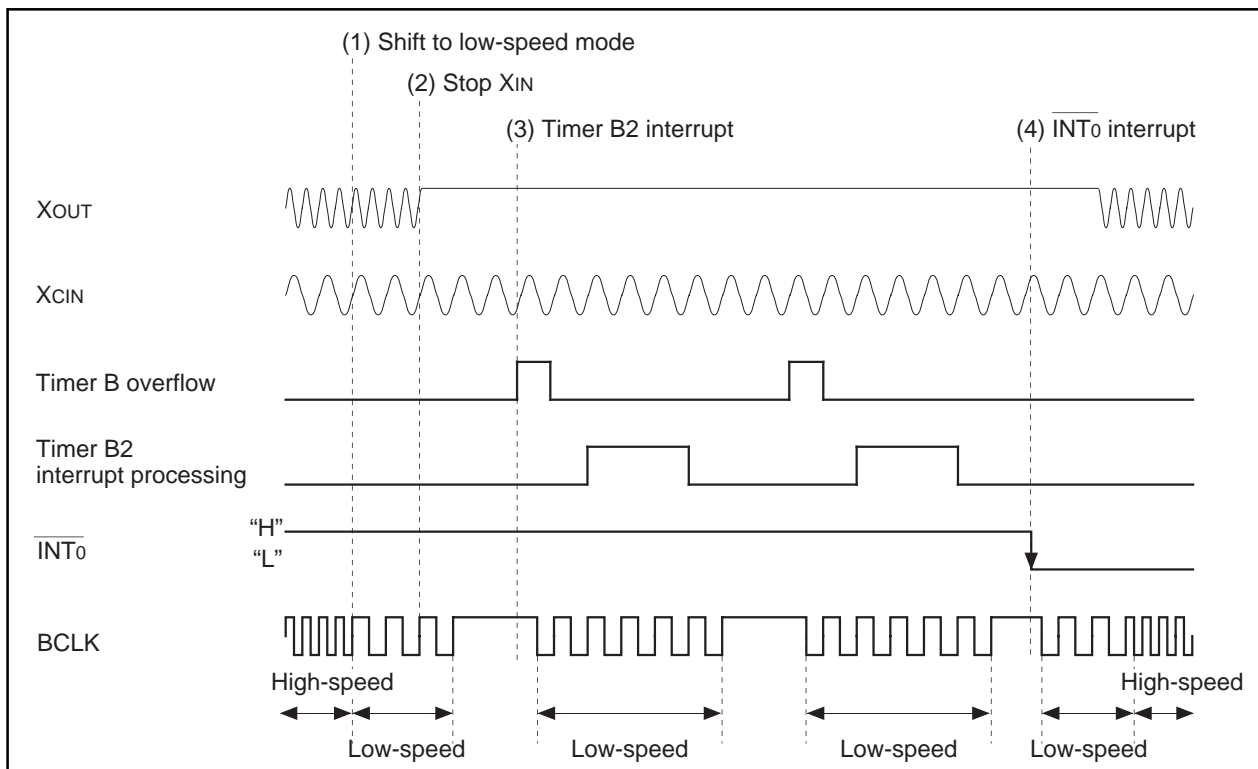


Figure 3.8.1. Operation timing of controlling power using wait mode

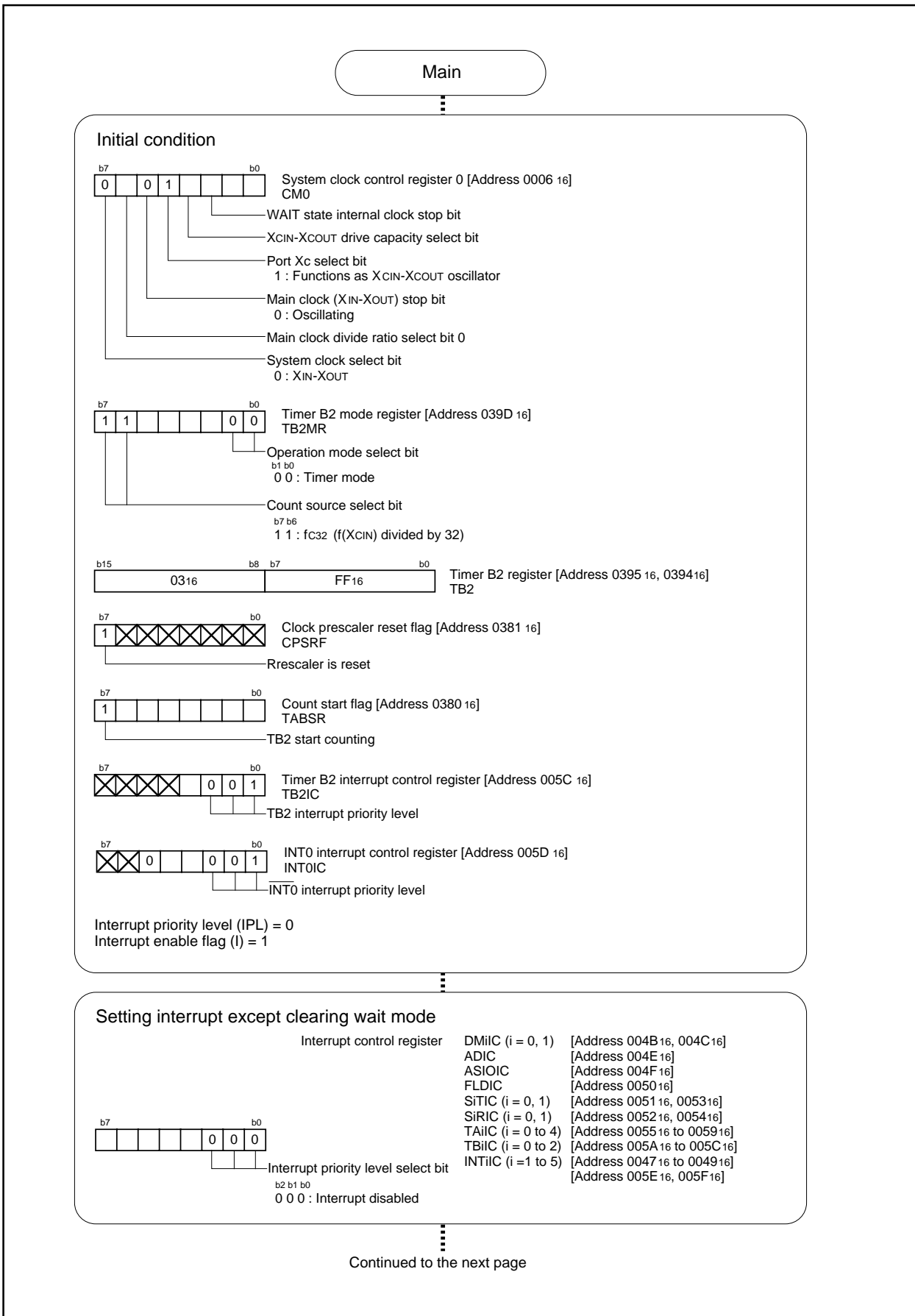


Figure 3.8.2. Set-up procedure of controlling power using wait mode (1)

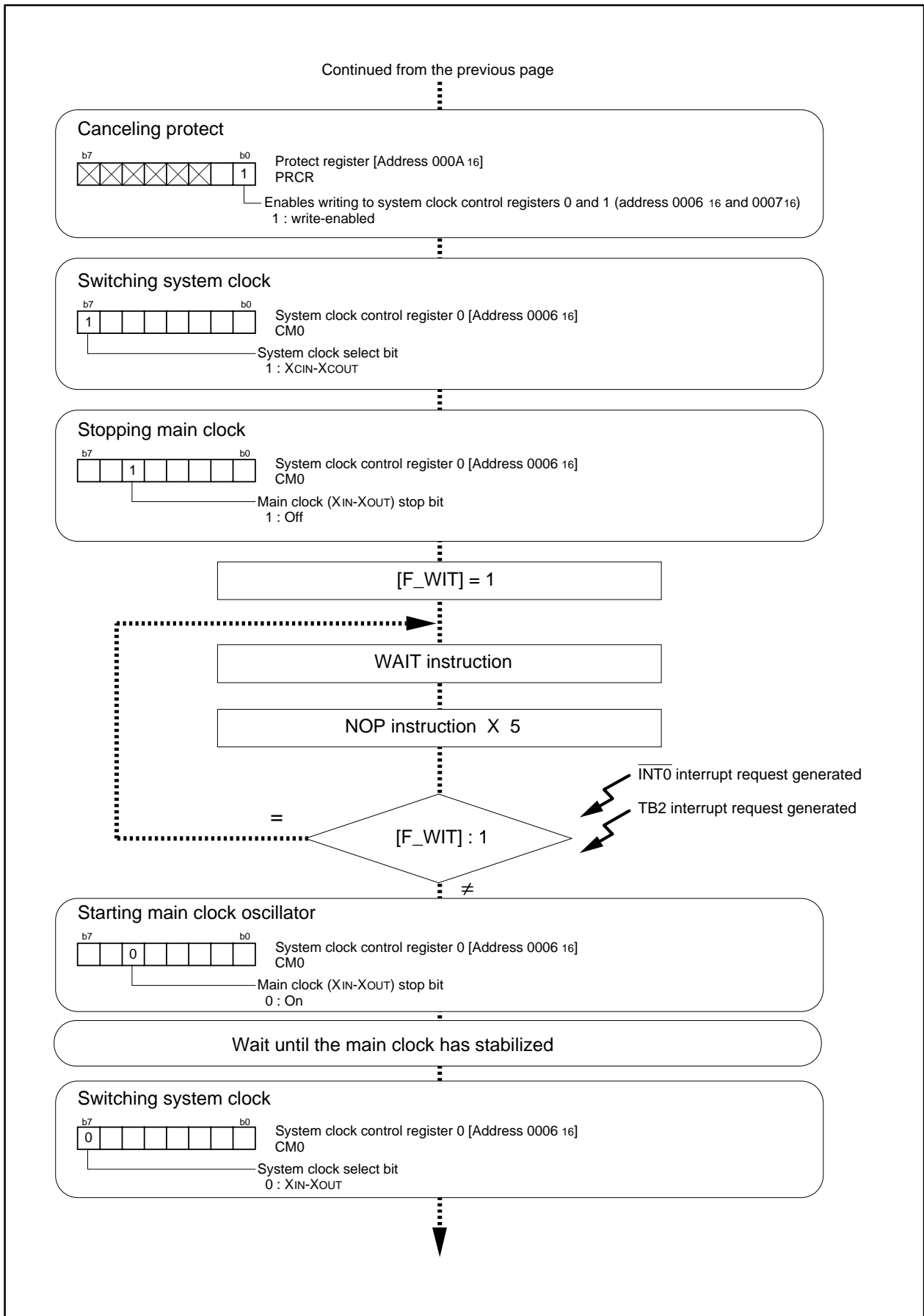


Figure 3.8.3. Set-up procedure of controlling power using wait mode (2)

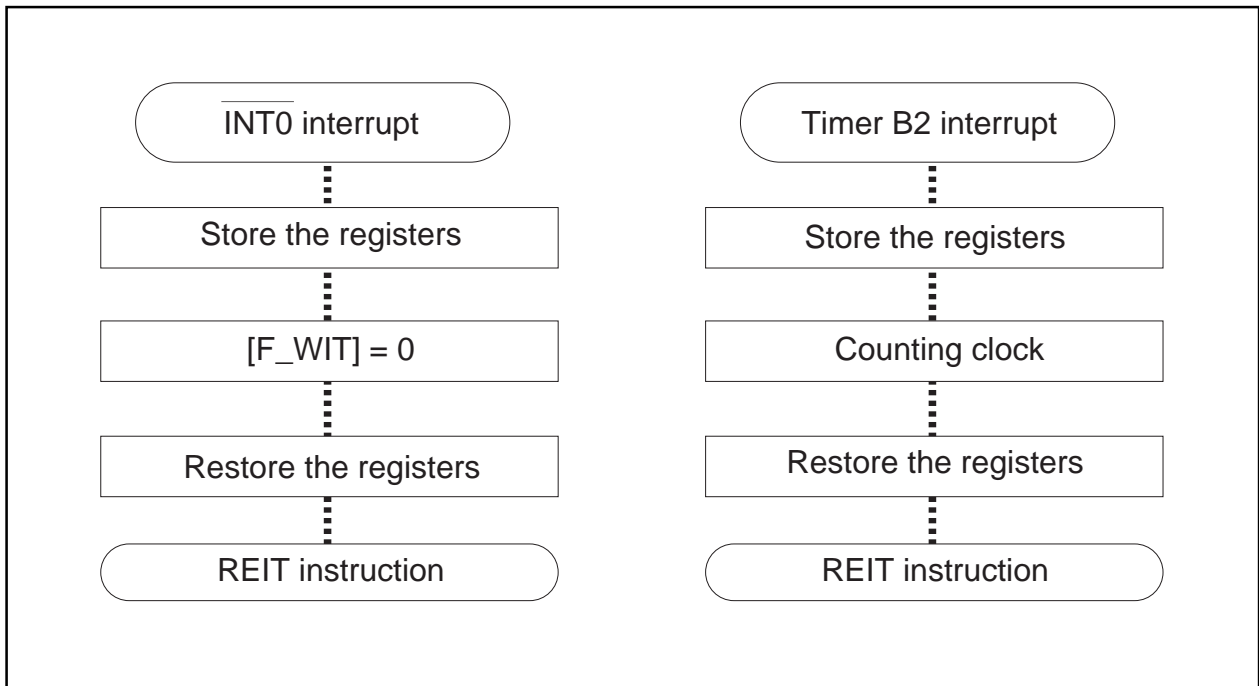


Figure 3.8.4. Set-up procedure of controlling power using wait mode (3)

This page kept blank for layout purposes.

# Chapter 4

---

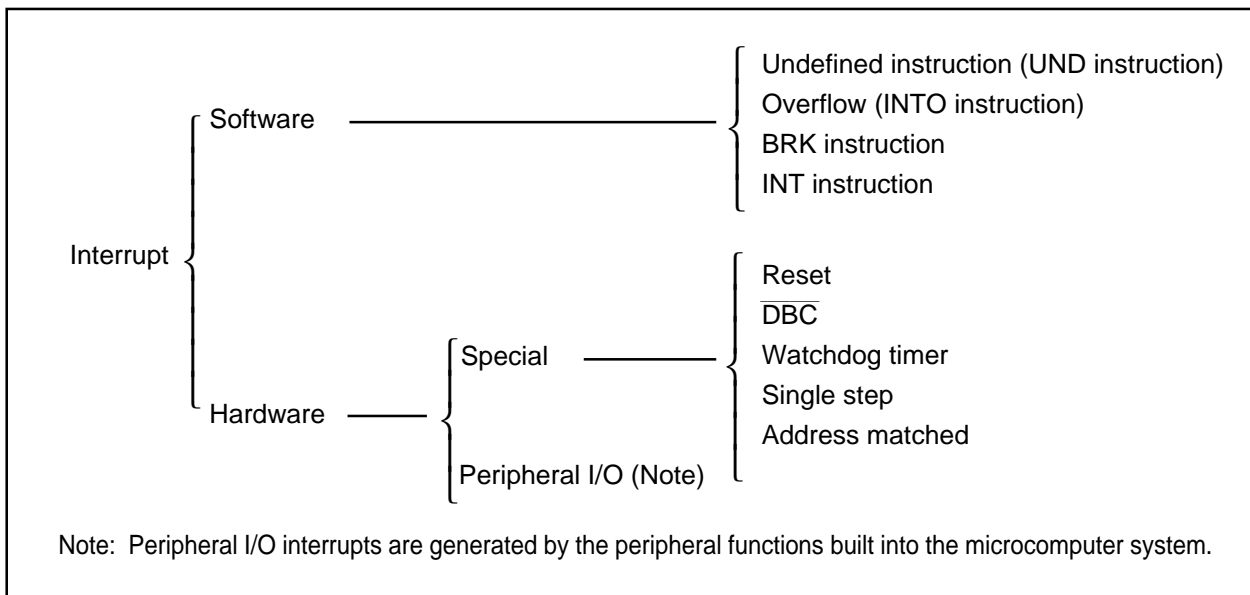
Interrupt



## 4.1 Overview of Interrupt

### 4.1.1 Type of Interrupts

Figure 4.1.1 lists the types of interrupts.



**Figure 4.1.1. 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.

### 4.1.2 Software Interrupts

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

- **Undefined instruction interrupt**

An undefined instruction interrupt occurs when executing the UND instruction.

- **Overflow interrupt**

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

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

- **BRK interrupt**

A BRK interrupt occurs when executing the BRK instruction.

- **INT interrupt**

An INT interrupt occurs when assigning one of software interrupt numbers 0 through 63 and executing the INT instruction. Software interrupt numbers 0 through 31 are assigned to peripheral 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.

### 4.1.3 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.

- **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. For address match interrupt, see 2.13 Address match Interrupt.

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

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

- **DMA0 interrupt, DMA1 interrupt**

These are interrupts DMA generates.

- **A-D conversion interrupt**

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

- **UART0 and UART1 transmission interrupt**

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

- **UART0 and UART1 reception interrupt**

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

- **SI/O automatic transfer interrupt**

This is an interrupt that the SI/O automatic transfer generates.

- **FLD interrupt**

This is an interrupt that FLD generates.

- **Timer A0 interrupt through timer A4 interrupt**

These are interrupts that timer A generates.

- **Timer B0 interrupt through timer B2 interrupt**

These are interrupts that timer B generates.

- **INT0 interrupt through INT5 interrupt**

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

#### 4.1.4 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. 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.

- **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<sub>16</sub> to FFFFF<sub>16</sub>. One vector table comprises four bytes. Set the first address of interrupt routine in each vector table. Table 4.1.1 shows the interrupts assigned to the fixed vector tables and addresses of vector tables.

**Table 4.1.1. 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	FFFD <sub>C16</sub> to FFFD <sub>F16</sub>	Interrupt on UND instruction
Overflow	FFFE <sub>016</sub> to FFFE <sub>316</sub>	Interrupt on INTO instruction
BRK instruction	FFFE <sub>416</sub> to FFFE <sub>716</sub>	If the vector contains FF <sub>16</sub> , program execution starts from the address shown by the vector in the variable vector table
Address match	FFFE <sub>816</sub> to FFFE <sub>B16</sub>	There is an address-matching interrupt enable bit
Single step (Note)	FFFE <sub>C16</sub> to FFFE <sub>F16</sub>	Do not use
Watchdog timer	FFFF <sub>016</sub> to FFFF <sub>316</sub>	
DBC (Note)	FFFF <sub>416</sub> to FFFF <sub>716</sub>	Do not use
-	FFFF <sub>816</sub> to FFFF <sub>B16</sub>	-
Reset	FFFF <sub>C16</sub> to FFFF <sub>F16</sub>	

Note: Interrupts used for debugging purposes only.

## Interrupt

- 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 4.1.2 shows the interrupts assigned to the variable vector tables and addresses of vector tables.

**Table 4.1.2. 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)	BRK instruction —	Cannot be masked I flag
Software interrupt number 7	+28 to +31 (Note)	$\overline{\text{INT3}}$	
Software interrupt number 8	+32 to +35 (Note)	$\overline{\text{INT4}}$	
Software interrupt number 9 —	+36 to +39 (Note)	$\overline{\text{INT5}}$ —	
Software interrupt number 11	+44 to +47 (Note)	DMA0	
Software interrupt number 12 —	+48 to +51 (Note)	DMA1 —	
Software interrupt number 14	+56 to +59 (Note)	A-D	
Software interrupt number 15	+60 to +63 (Note)	SI/O automatic transfer	
Software interrupt number 16	+64 to +67 (Note)	FLD	
Software interrupt number 17	+68 to +71 (Note)	UART0 transmit	
Software interrupt number 18	+72 to +75 (Note)	UART0 receive	
Software interrupt number 19	+76 to +79 (Note)	UART1 transmit	
Software interrupt number 20	+80 to +83 (Note)	UART1 receive	
Software interrupt number 21	+84 to +87 (Note)	Timer A0	
Software interrupt number 22	+88 to +91 (Note)	Timer A1	
Software interrupt number 23	+92 to +95 (Note)	Timer A2	
Software interrupt number 24	+96 to +99 (Note)	Timer A3	
Software interrupt number 25	+100 to +103 (Note)	Timer A4	
Software interrupt number 26	+104 to +107 (Note)	Timer B0	
Software interrupt number 27	+108 to +111 (Note)	Timer B1	
Software interrupt number 28	+112 to +115 (Note)	Timer B2	
Software interrupt number 29	+116 to +119 (Note)	$\overline{\text{INT0}}$	
Software interrupt number 30	+120 to +123 (Note)	$\overline{\text{INT1}}$	
Software interrupt number 31	+124 to +127 (Note)	$\overline{\text{INT2}}$	
Software interrupt number 32 to Software interrupt number 63	+128 to +131 (Note) to +252 to +255 (Note)	Software interrupt	Cannot be masked I flag

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

## 4.2 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 non-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 4.2.1 shows the memory map of the interrupt control registers, and Figure 4.2.2 shows the interrupt control registers.

0047 <sub>16</sub>	INT3 interrupt control register (INT3IC)
0048 <sub>16</sub>	INT4 interrupt control register (INT4IC)
0049 <sub>16</sub>	INT5 interrupt control register (INT5IC)
004A <sub>16</sub>	
004B <sub>16</sub>	DMA0 interrupt control register (DM0IC)
004C <sub>16</sub>	DMA1 interrupt control register (DM1IC)
004D <sub>16</sub>	
004E <sub>16</sub>	A-D conversion interrupt control register (ADIC)
004F <sub>16</sub>	SI/O2 transmit interrupt control register (ASIOIC)
0050 <sub>16</sub>	FLD interrupt control register (FLDIC)
0051 <sub>16</sub>	UART0 transmit interrupt control register (S0TIC)
0052 <sub>16</sub>	UART0 receive interrupt control register (S0RIC)
0053 <sub>16</sub>	UART1 transmit interrupt control register (S1TIC)
0054 <sub>16</sub>	UART1 receive interrupt control register (S1RIC)
0055 <sub>16</sub>	Timer A0 interrupt control register (TA0IC)
0056 <sub>16</sub>	Timer A1 interrupt control register (TA1IC)
0057 <sub>16</sub>	Timer A2 interrupt control register (TA2IC)
0058 <sub>16</sub>	Timer A3 interrupt control register (TA3IC)
0059 <sub>16</sub>	Timer A4 interrupt control register (TA4IC)
005A <sub>16</sub>	Timer B0 interrupt control register (TB0IC)
005B <sub>16</sub>	Timer B1 interrupt control register (TB1IC)
005C <sub>16</sub>	Timer B2 interrupt control register (TB2IC)
005D <sub>16</sub>	INT0 interrupt control register (INT0IC)
005E <sub>16</sub>	INT1 interrupt control register (INT1IC)
005F <sub>16</sub>	INT2 interrupt control register (INT2IC)

Figure 4.2.1. Memory map of the interrupt control registers

## Interrupt

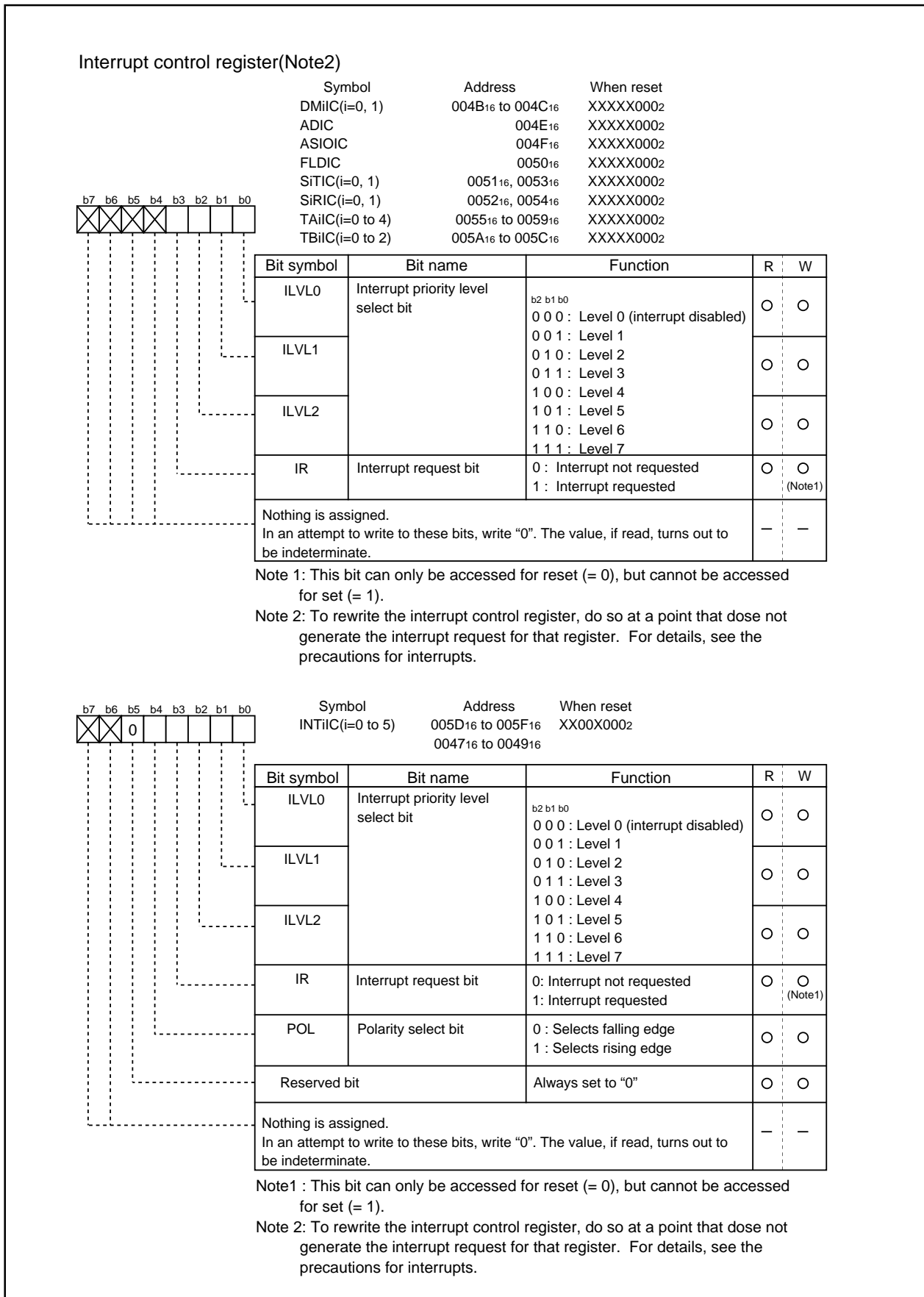


Figure 4.2.2. Interrupt control registers

### 4.2.1 Interrupt Enable 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.

The content is changed when the I flag is changed causes the acceptance of the interrupt request in the following timing:

- When changing the I flag using the REIT instruction, the acceptance of the interrupt takes effect as the REIT instruction is executed.
- When changing the I flag using one of the FCLR, FSET, POPC, and LDC instructions, the acceptance of the interrupt is effective as the next instruction is executed.

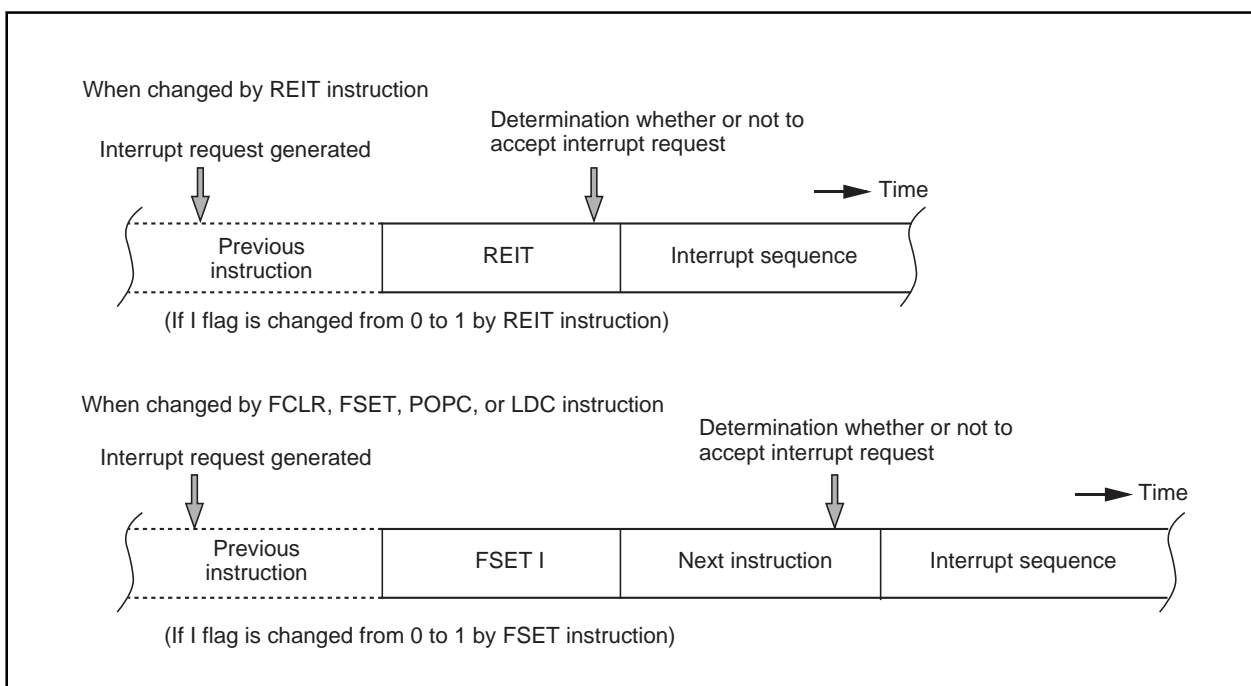


Figure 4.2.3. The timing of reflecting the change in the I flag to the interrupt

### 4.2.2 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").



### 4.2.3 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 4.2.1 shows the settings of interrupt priority levels and Table 4.2.2 shows the interrupt levels enabled, according to the consist of the IPL.

The following are conditions under which an interrupt is accepted:

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

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

**Table 4.2.1. 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  High
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	

**Table 4.2.2. Interrupt levels enabled according**

IPL	Enabled interrupt priority levels
IPL <sub>2</sub> IPL <sub>1</sub> IPL <sub>0</sub> 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

When either the IPL or the interrupt priority level is changed, the new level is reflected to the interrupt in the following timing:

- When changing the IPL using the REIT instruction, the reflection takes effect as of the instruction that is executed in 2 clock cycles after the last clock cycle involved in the REIT instruction.
- When changing the IPL using either the POPC, LDC or LDIPL instruction, the reflection takes effect as of the instruction that is executed in 3 cycles after the last clock cycle involved in the instruction used.
- When changing the interrupt priority level using the MOV or similar instruction, the reflection takes effect as of the instruction that is executed in 2 clock cycles after the last clock cycle involved in the instruction used.

#### 4.2.4 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
  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
  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 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

### 4.3 Interrupt Sequence

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

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

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

- (1) CPU gets the interrupt information (the interrupt number and interrupt request level) by reading address 00000<sub>16</sub>.
- (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 1) 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.

#### 4.3.1 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 4.3.1 shows the interrupt response time.

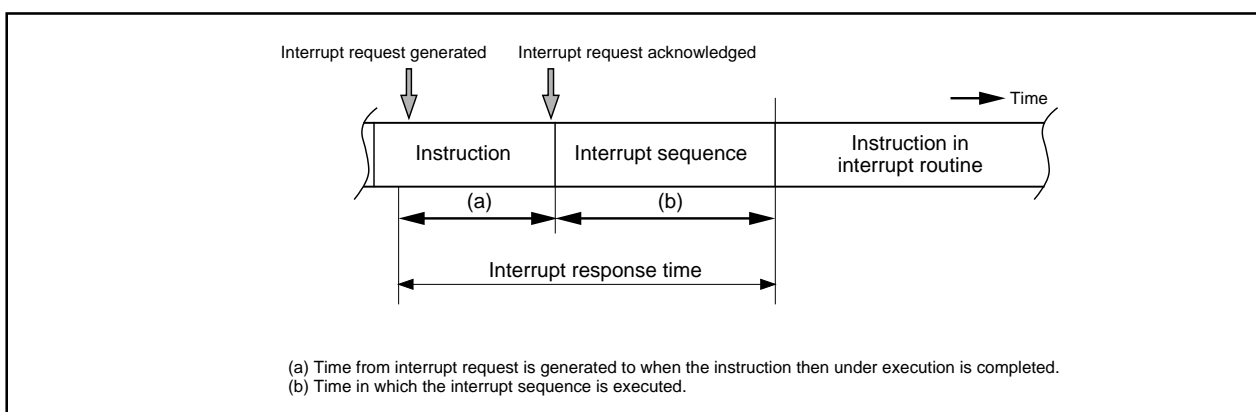


Figure 4.3.1. Interrupt response time

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

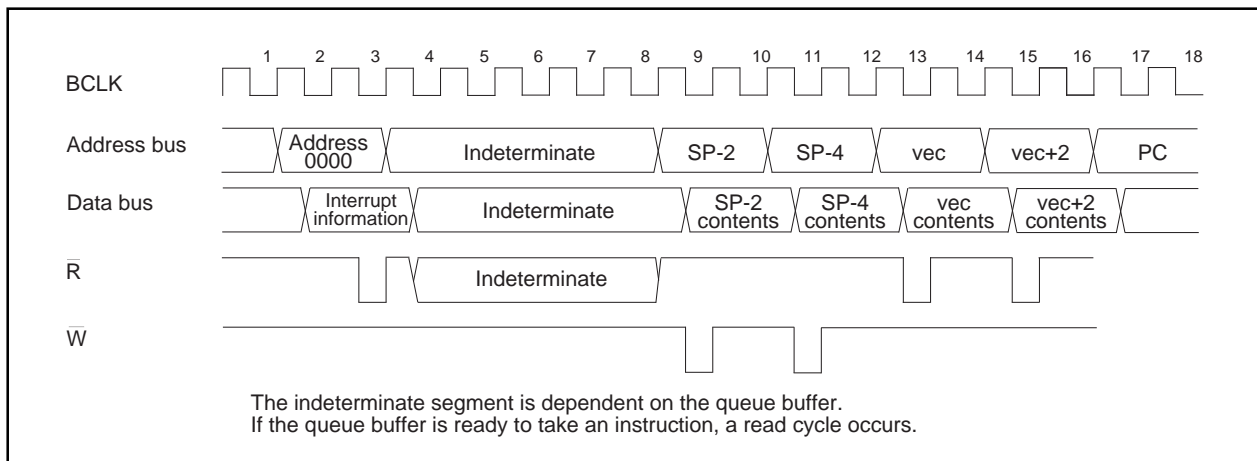
Time (b) is as shown in Table 4.3.1.

**Table 4.3.1. 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  $\overline{DBC}$  interrupt; add 1 cycle in the case either of an address match interrupt or of a single-step interrupt.

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



**Figure 4.3.2. Time required for executing the interrupt sequence**

### 4.3.2 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 4.3.2 is set in the IPL.

**Table 4.3.2. Relationship between interrupts without interrupt priority levels and IPL**

Interrupt sources without priority levels	Value set in the IPL
Watchdog timer	7
Reset	0
Other	Not changed

### 4.3.3 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 4.3.3 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).

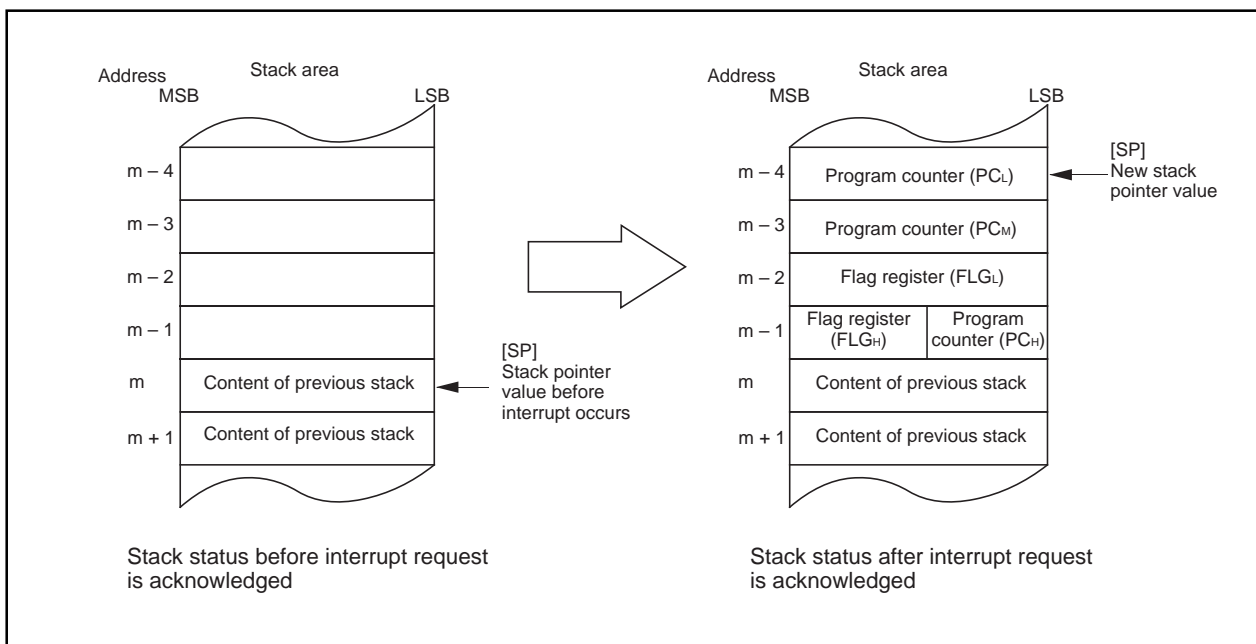


Figure 4.3.3. State of stack before and after acceptance of interrupt request

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 4.3.4 shows the operation of the saving registers.

Note: Stack pointer indicated by U flag.

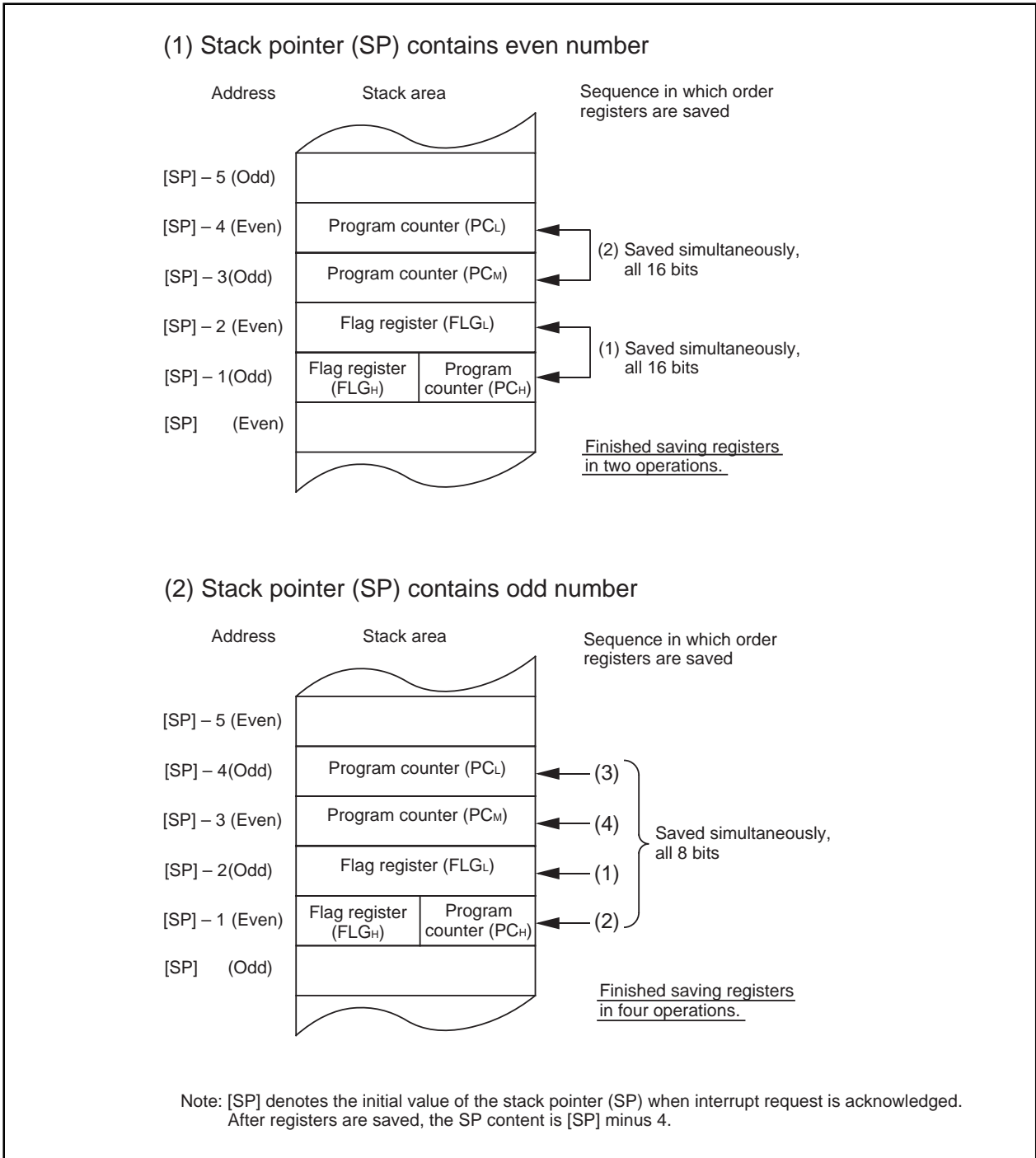


Figure 4.3.4. Operation of saving registers

#### 4.4 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.

#### 4.5 Interrupt Priority

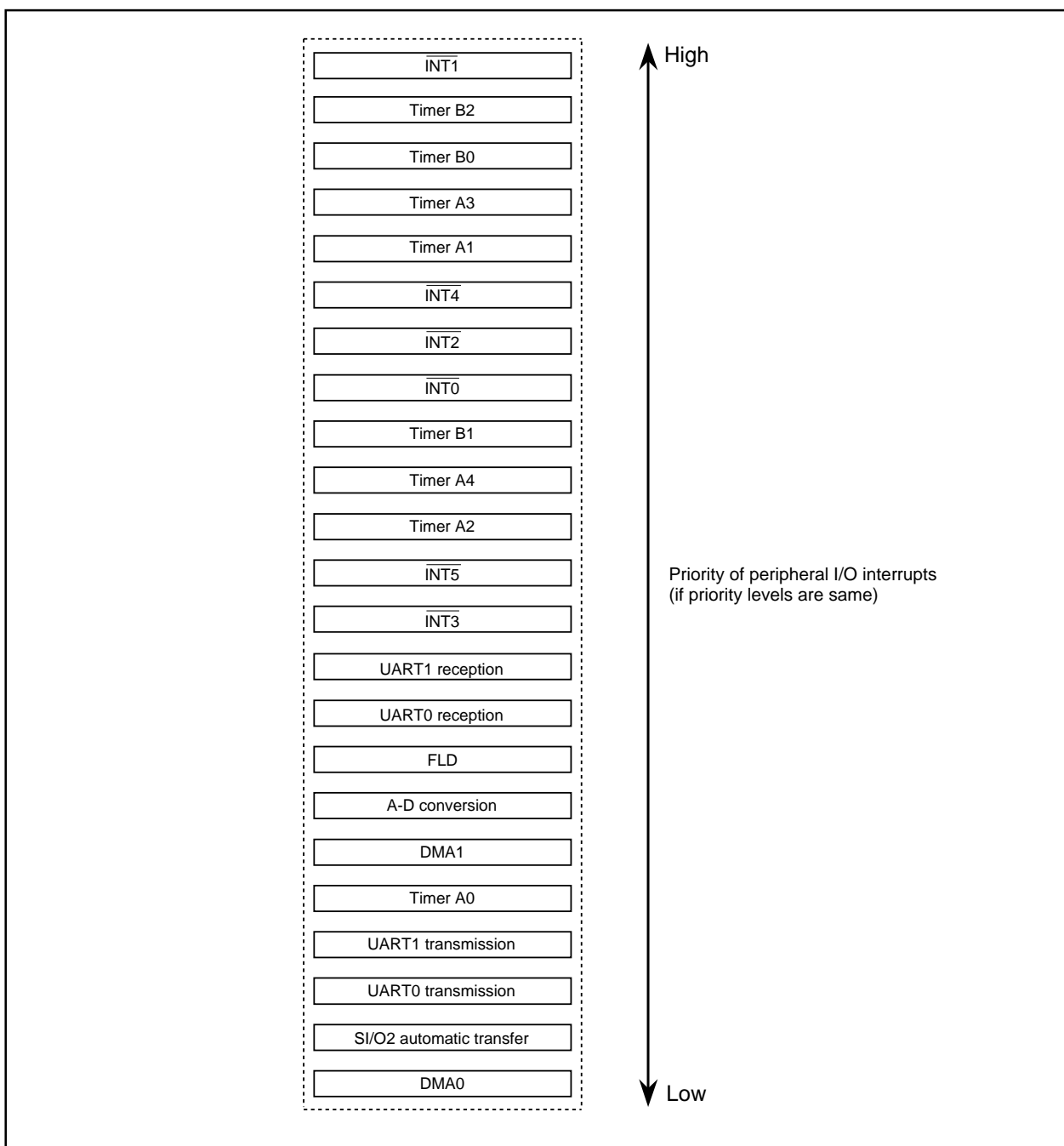
If there are two or more interrupt requests occurring at a point in time within a single sampling (checking whether interrupt requests are made), the interrupt assigned a higher priority is accepted.

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 (see Figure 4.5.1).

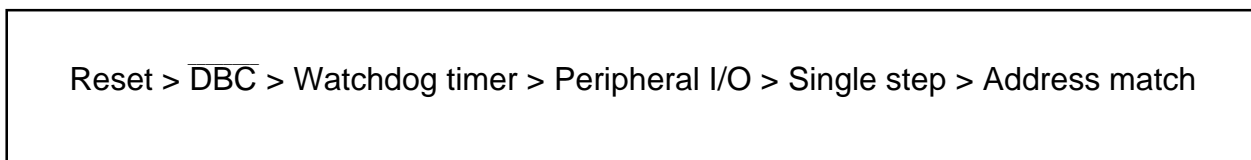
Priorities of the special interrupts, such as Reset (dealt with as an interrupt assigned the highest priority), watchdog timer interrupt, etc. are regulated by hardware.

Figure 4.5.2 shows the priorities of hardware interrupts.

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



**Figure 4.5.1. Maskable interrupts priorities (peripheral I/O interrupts)**



**Figure 4.5.2. Hardware interrupts priorities**



## 4.6 Multiple Interrupts

The state when control branched to an interrupt routine is described below:

- The interrupt enable flag (I flag) is set to "0" (the interrupt is disabled).
- The interrupt request bit of the accepted interrupt is set to "0".
- The processor interrupt priority level (IPL) is assigned to the same interrupt priority level as assigned to the accepted interrupt.

Setting the interrupt enable flag (I flag) to "1" within an interrupt routine allows an interrupt request assigned a priority higher than the IPL to be accepted. Figure 4.6.1 shows the scheme of multiple interrupts.

An interrupt request that is not accepted because of low priority will be held. If the condition following is met when the REIT instruction returns the IPL and the interrupt priority is determined, then the interrupt request being held is accepted.

Interrupt priority level of the interrupt request being held > Returned the IPL

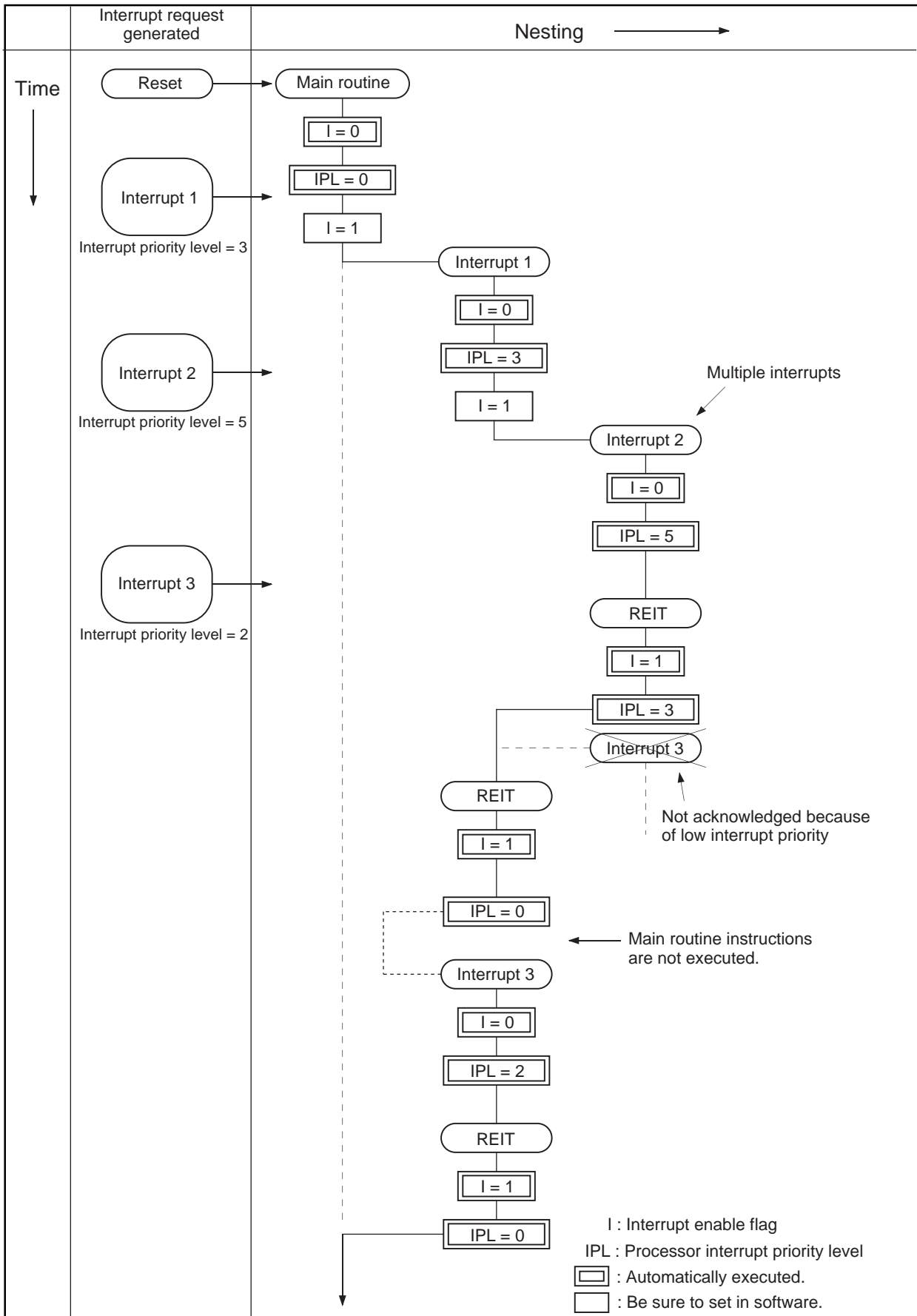


Figure 4.6.1. Multiple interrupts

## 4.7 Precautions for Interrupts

### (1) Reading address 00000<sub>16</sub>

- 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<sub>16</sub> will then be set to "0".

Reading address 00000<sub>16</sub> 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<sub>16</sub> by software.

### (2) Setting the stack pointer

- The value of the stack pointer immediately after reset is initialized to 0000<sub>16</sub>. 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.

### (3) External interrupt

- Either an "L" level or an "H" level of at least 250 ns width is necessary for the signal input to pins  $\overline{INT_0}$  through  $\overline{INT_5}$  regardless of the CPU operation clock.
- When the polarity of the  $\overline{INT_0}$  to  $\overline{INT_5}$  pins is changed, the interrupt request bit is sometimes set to "1". After changing the polarity, set the interrupt request bit to "0". Figure 4.7.1 shows the procedure for changing the  $\overline{INT}$  interrupt generate factor.

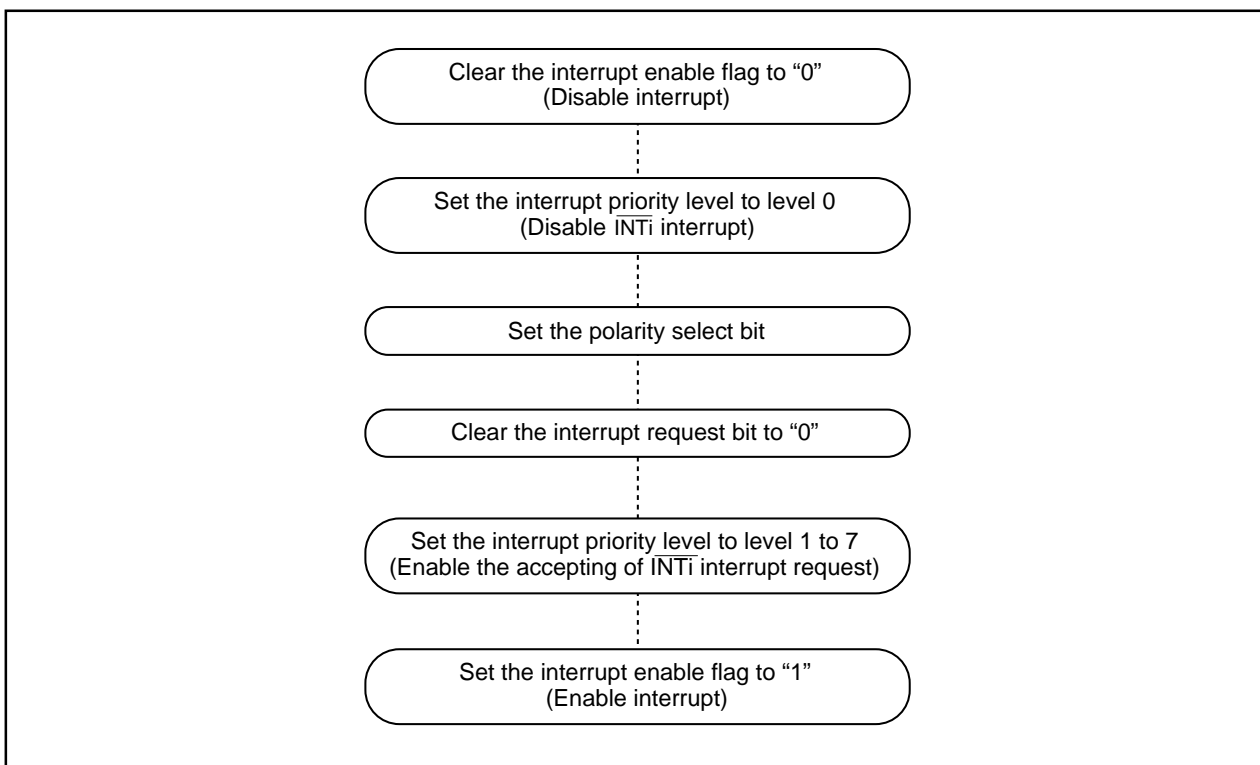


Figure 4.7.1. Switching condition of  $\overline{INT}$  interrupt request

**(4) 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                    ;
  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        ;
  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 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

This page kept blank for layout purposes.

# Chapter 5

---

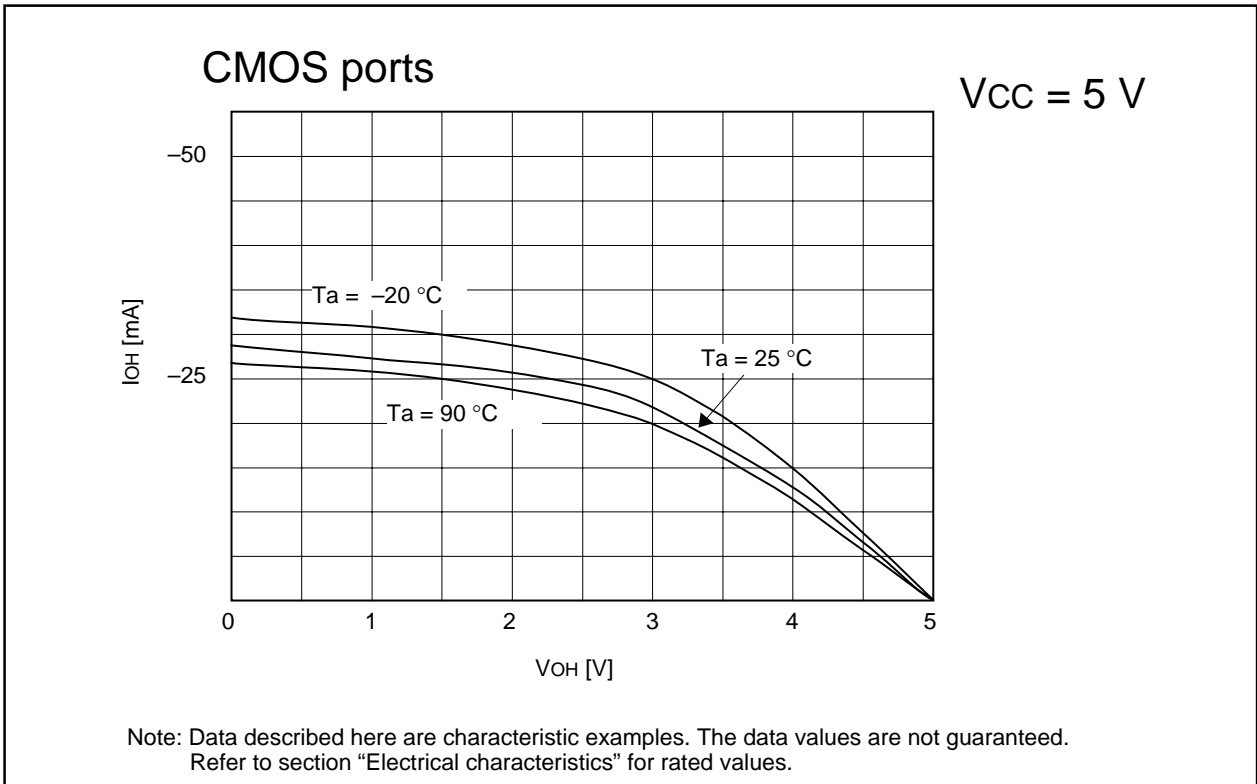
## Standard Characteristics

## **5.1 Standard DC Characteristics**

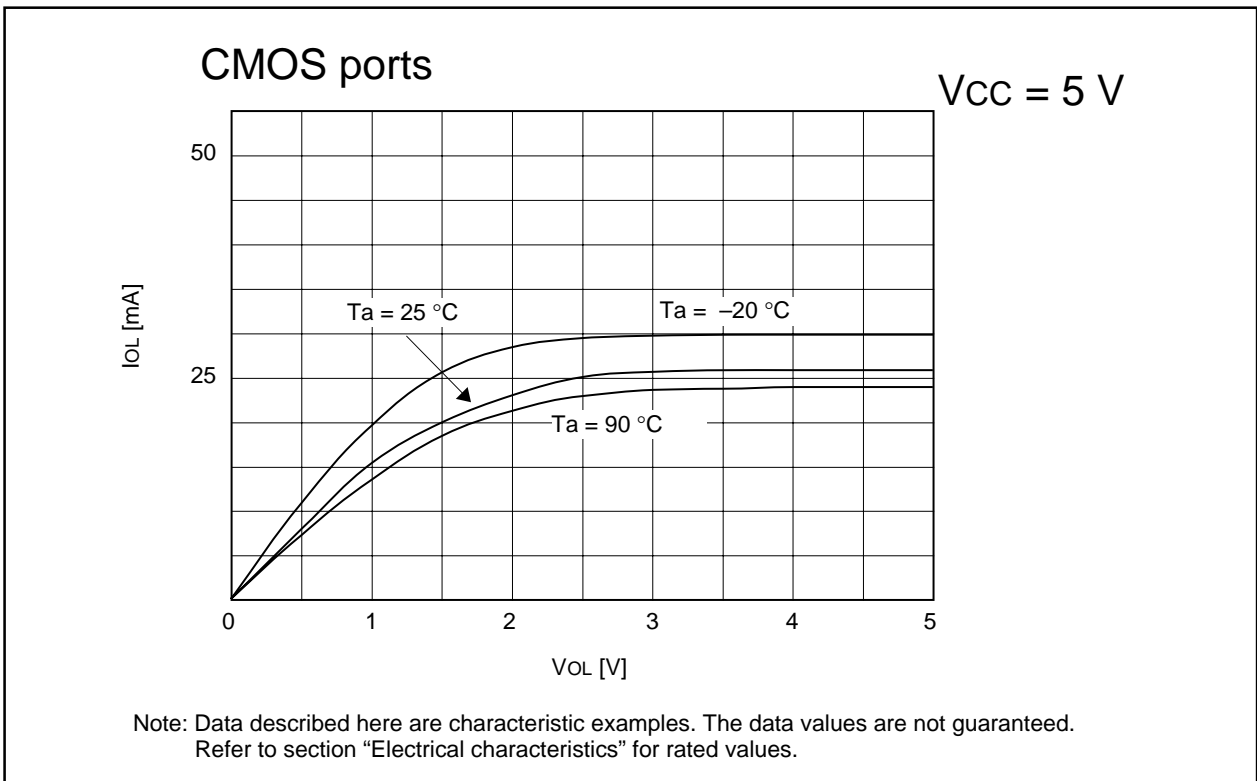
The standard characteristics given in this section are examples of M30218MC-AXXXFP. The contents of these examples cannot be guaranteed. For standardized values, see "Electric characteristics".

### **5.1.1 Standard Ports Characteristics**

Figures 5.1.1 through 5.1.6 show the standard ports characteristics.

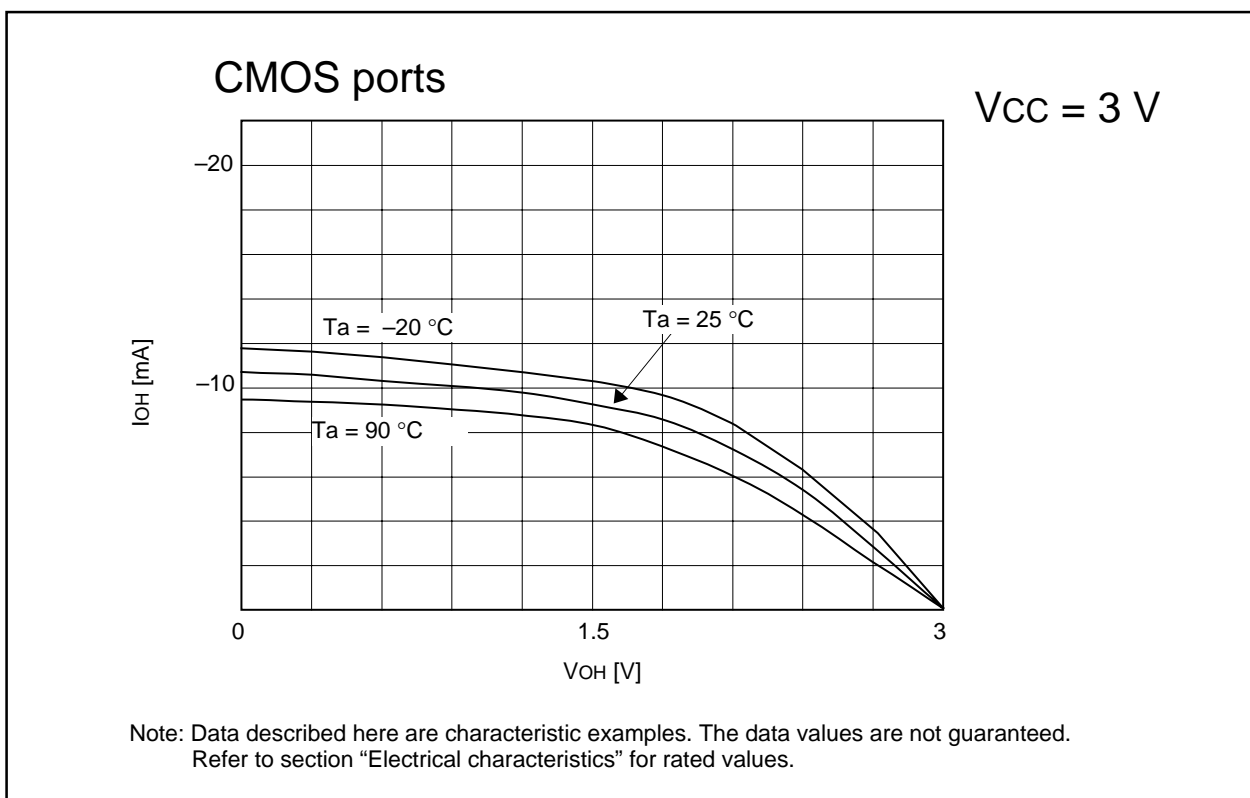


**Figure 5.1.1.  $I_{OH}$  -  $V_{OH}$  standard characteristics of ports P44 to P47, P7 to P10 ( $V_{CC} = 5\text{V}$ )**

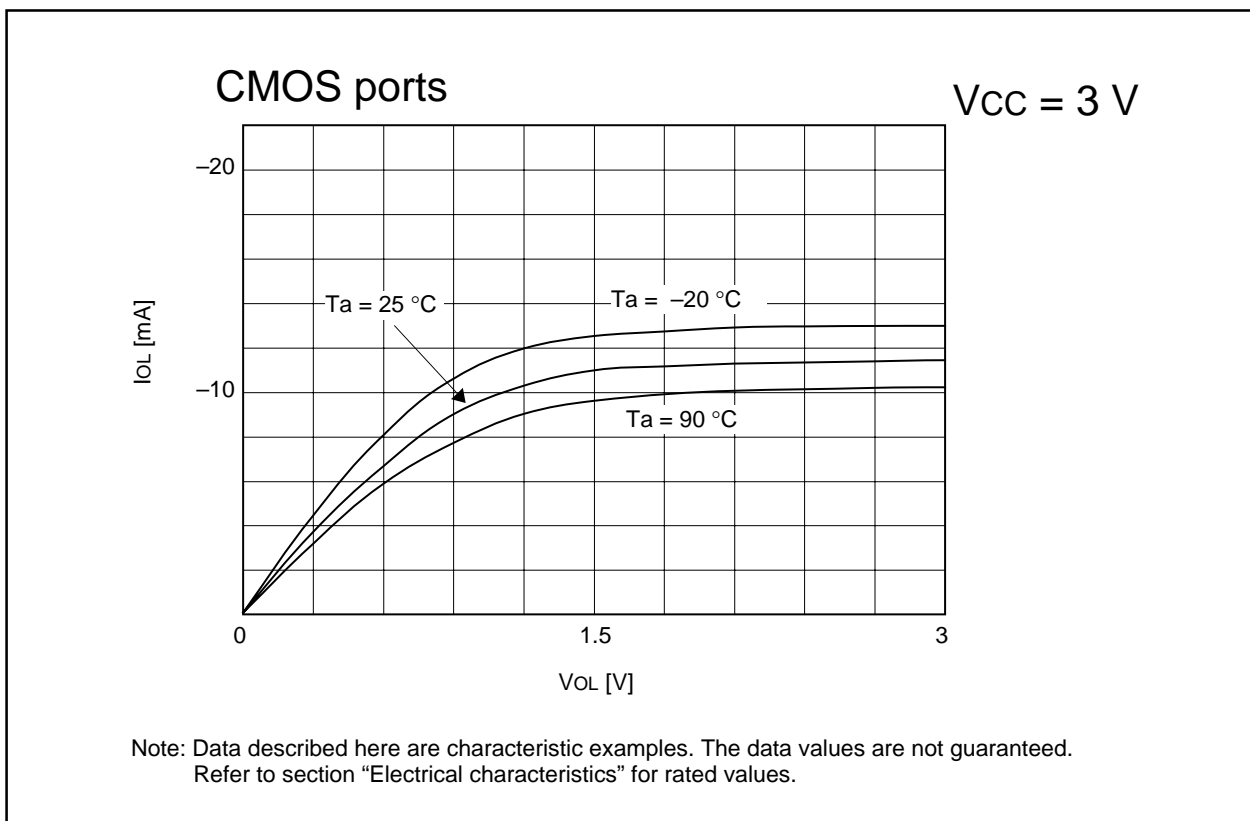


**Figure 5.1.2.  $I_{OL}$  -  $V_{OL}$  standard characteristics of ports P44 to P47, P7 to P10 ( $V_{CC} = 5\text{V}$ )**

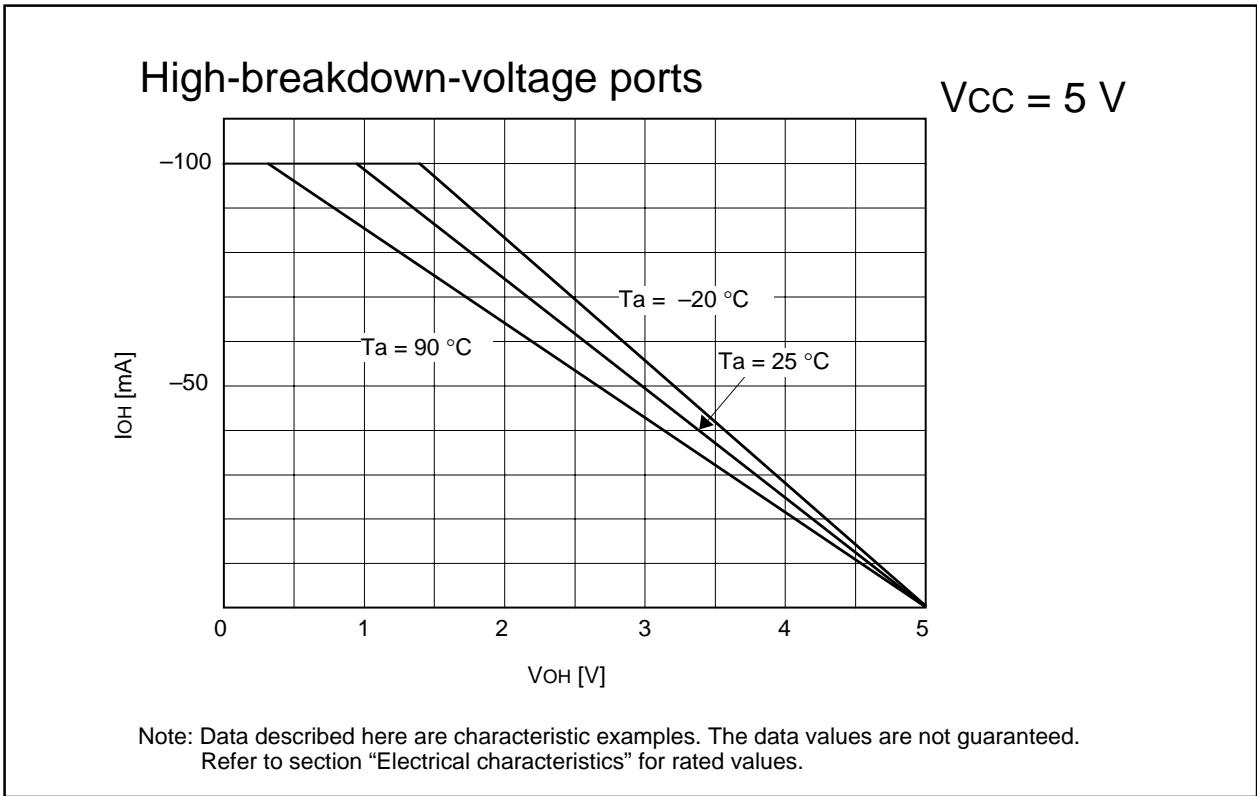




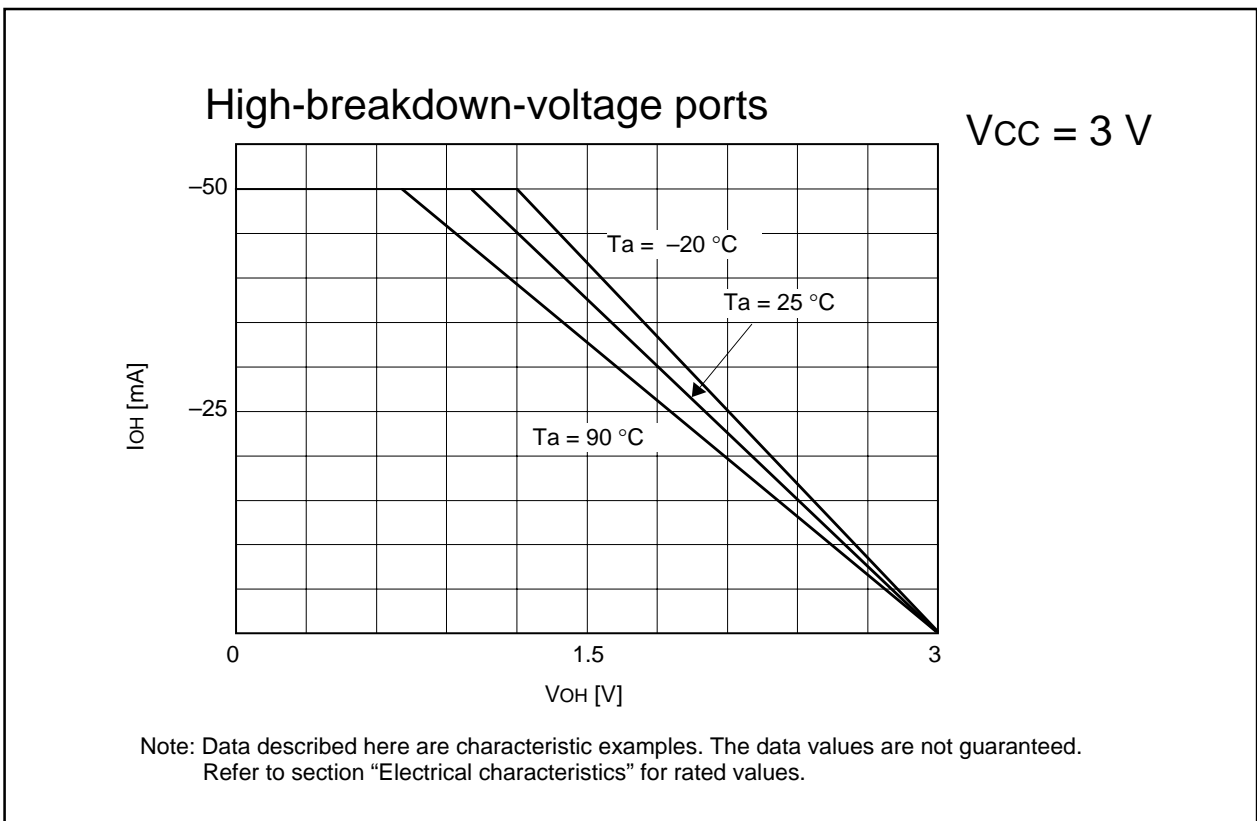
**Figure 5.1.3.  $I_{OH}$  -  $V_{OH}$  standard characteristics of ports P44 to P47, P7 to P10 ( $V_{CC} = 3V$ )**



**Figure 5.1.4.  $I_{OL}$  -  $V_{OL}$  standard characteristics of ports P44 to P47, P7 to P10 ( $V_{CC} = 3V$ )**



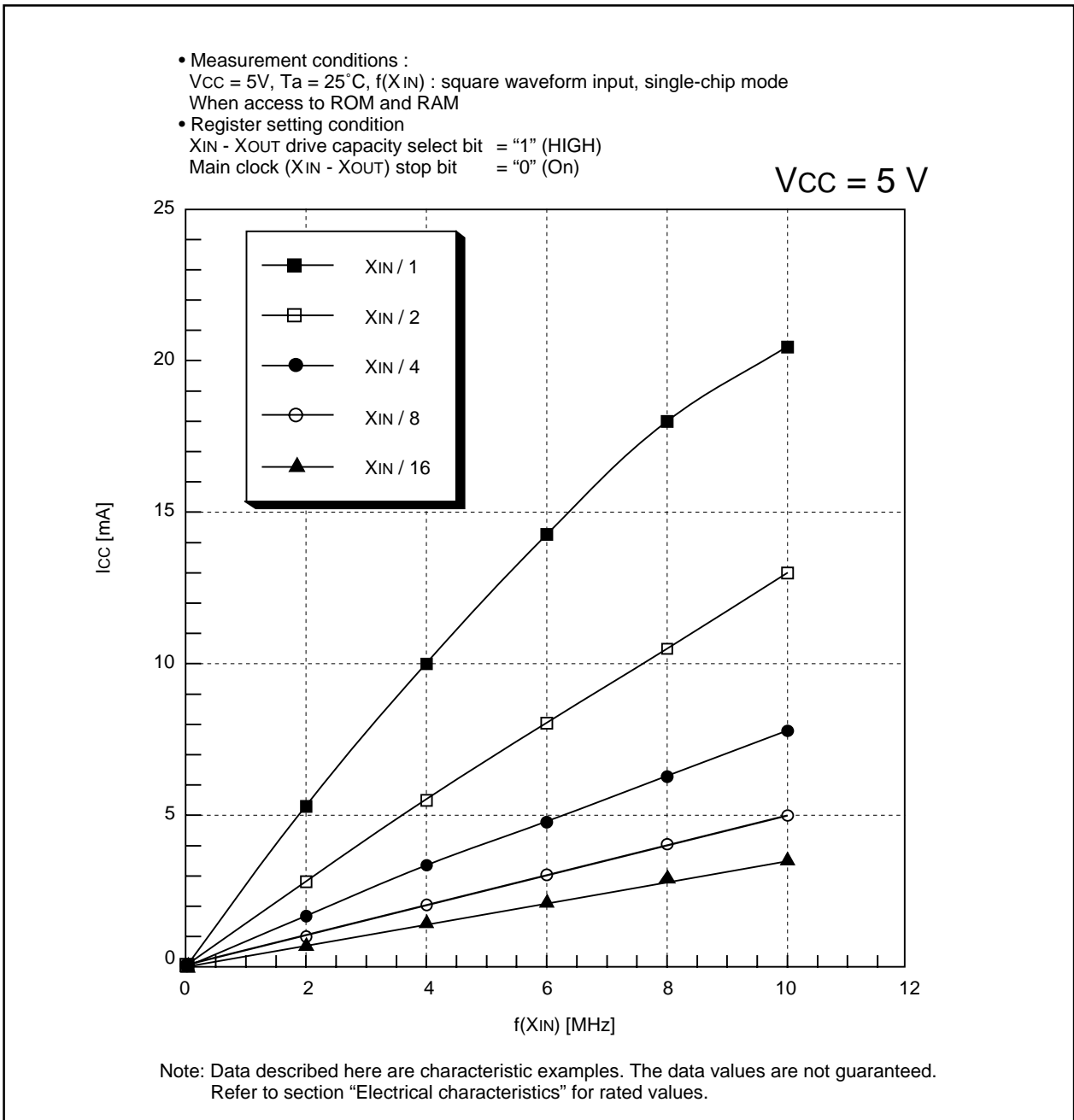
**Figure 5.1.5.  $I_{OH}$  -  $V_{OH}$  standard characteristics of ports P0 to P3, P40 to P43, P5, P6 ( $V_{CC} = 5\text{ V}$ )**



**Figure 5.1.6.  $I_{OL}$  -  $V_{OL}$  standard characteristics of ports P0 to P3, P40 to P43, P5, P6 ( $V_{CC} = 3\text{ V}$ )**

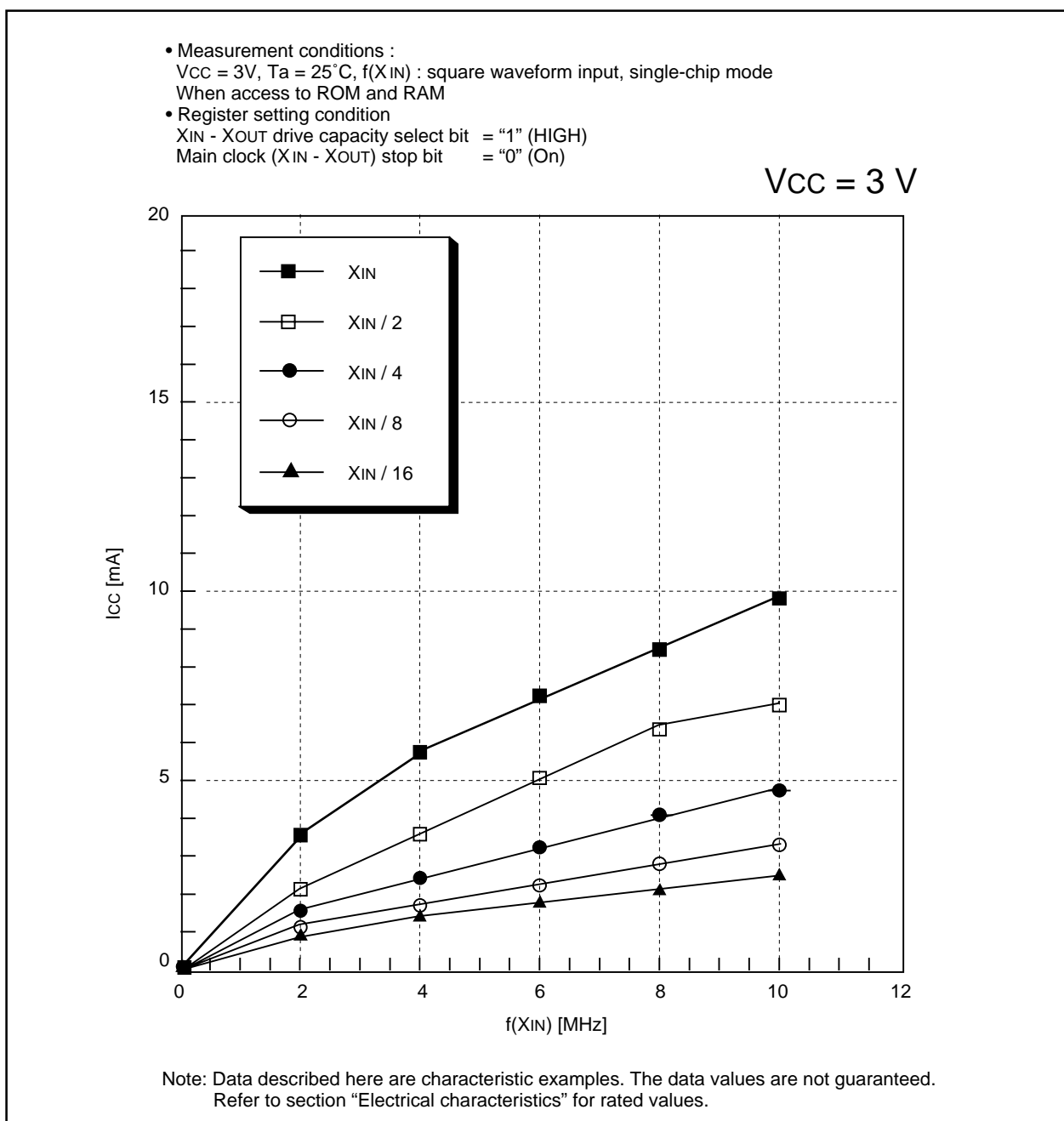
**5.1.2 Characteristics of  $I_{CC}$ - $f(X_{IN})$**

Figures 5.1.7 and 5.1.8 show the Characteristics of  $I_{CC}$ - $f(X_{IN})$ .



**Figures 5.1.7. Characteristics of  $I_{CC}$ - $f(X_{IN})$  ( $V_{CC} = 5V$ )**

## Standard Characteristics



Figures 5.1.8. Characteristics of  $I_{CC}-f(X_{IN})$  ( $V_{CC} = 3V$ )

### 5.2 Standard Characteristics of A-D Converter

The standard characteristics given in this section are an example of M30218MC-AXXXFP. The contents of these examples cannot be guaranteed. For standardize values, see “Electric characteristics”.

Figures 5.2.1 and 5.2.2 show the standard characteristics of the A-D converter.

The line on the top side of the graph represents absolute errors.

The line on the bottom side of the graph represents the width of input voltage bearing the equal output code.

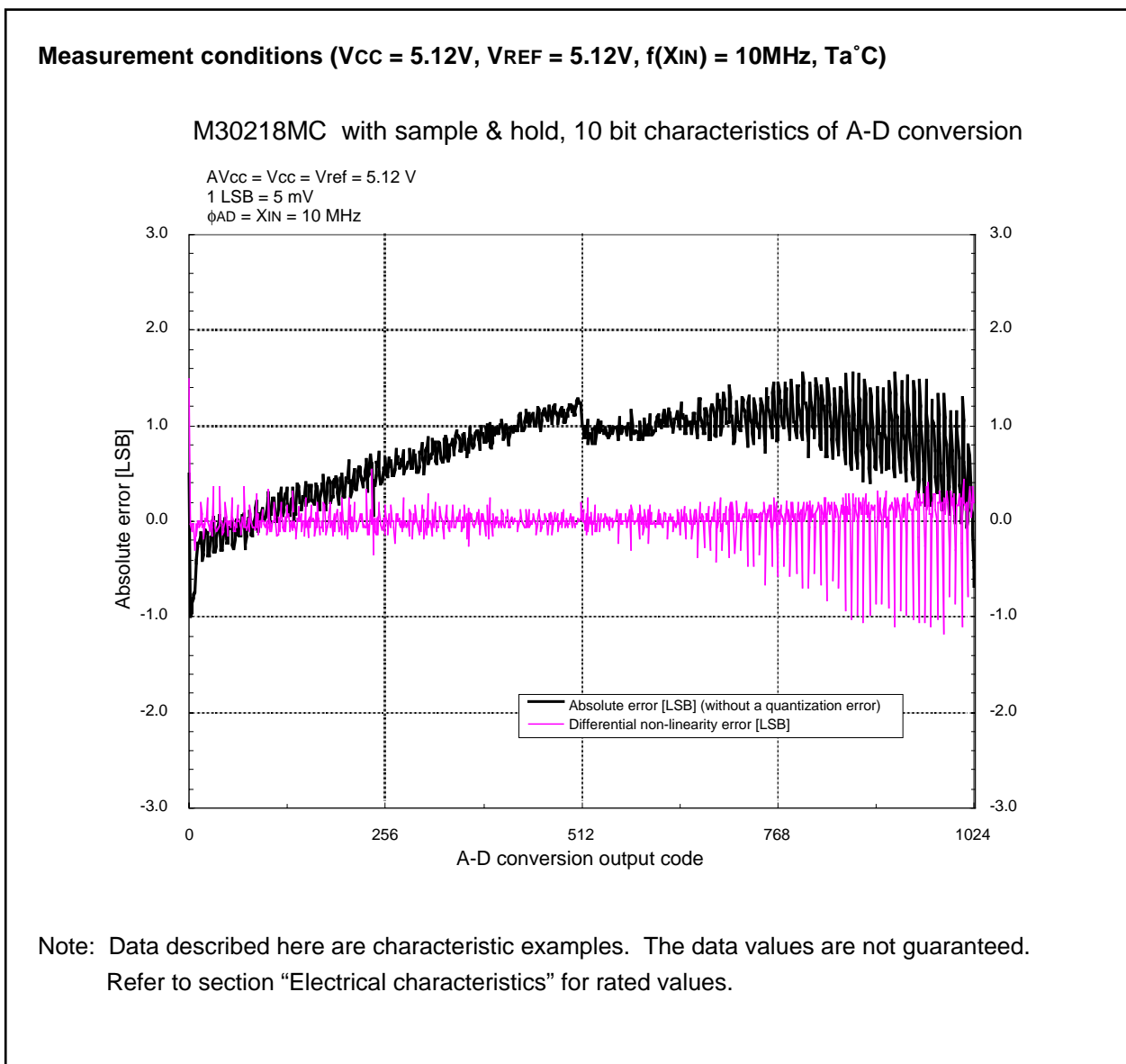


Figure 5.2.1. Standard characteristics of the A-D converter

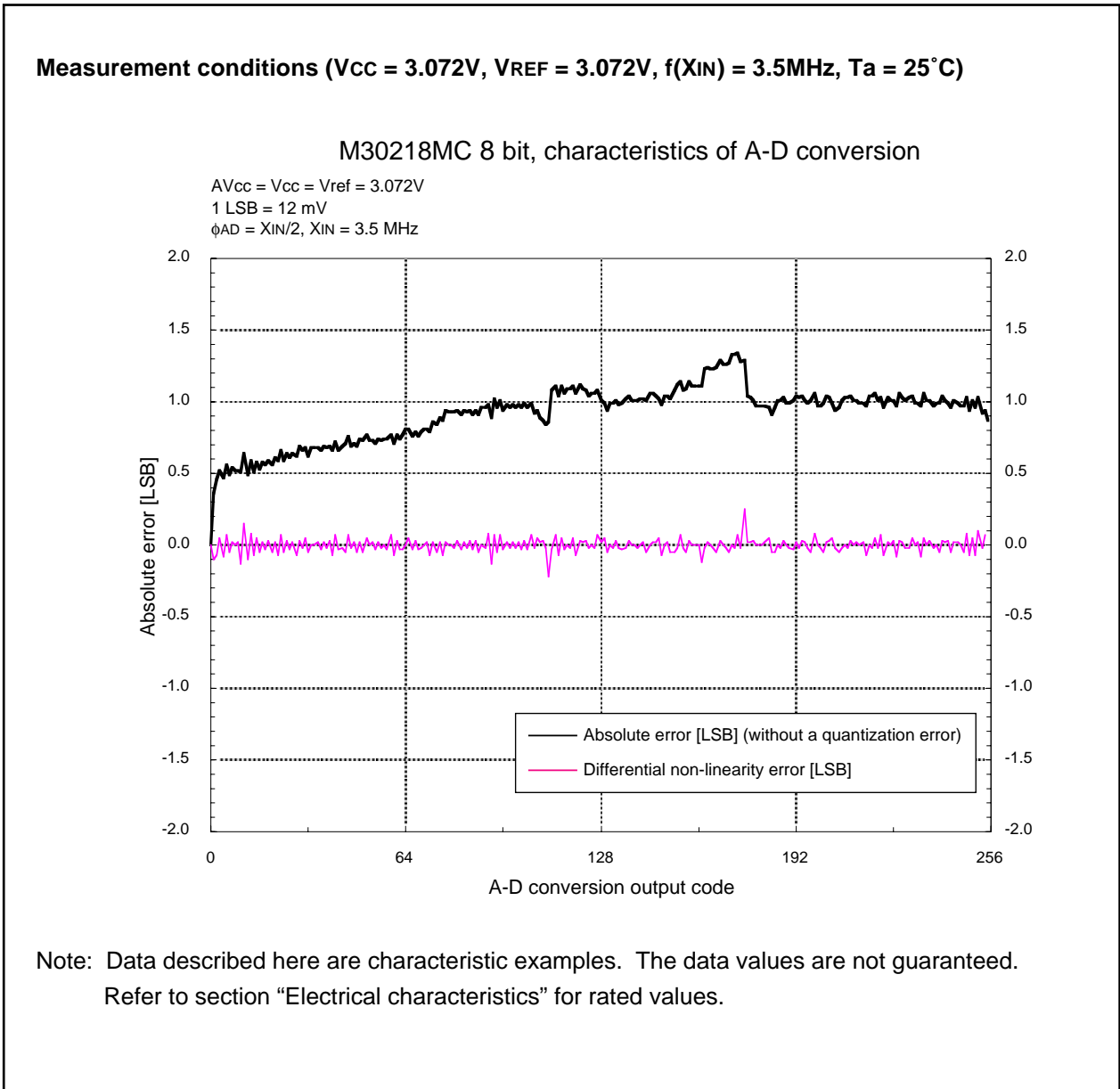


Figure 5.2.2. Standard characteristics of the A-D converter

### 5.3 Standard Characteristics of D-A Converter

The standard characteristics given in this section are an example of M30218MC-AXXXFP. The contents of these examples cannot be guaranteed. For standardized values, see "Electric characteristics".

Figures 5.3.1 and 5.3.2 show the standard characteristics of the D-A converter.

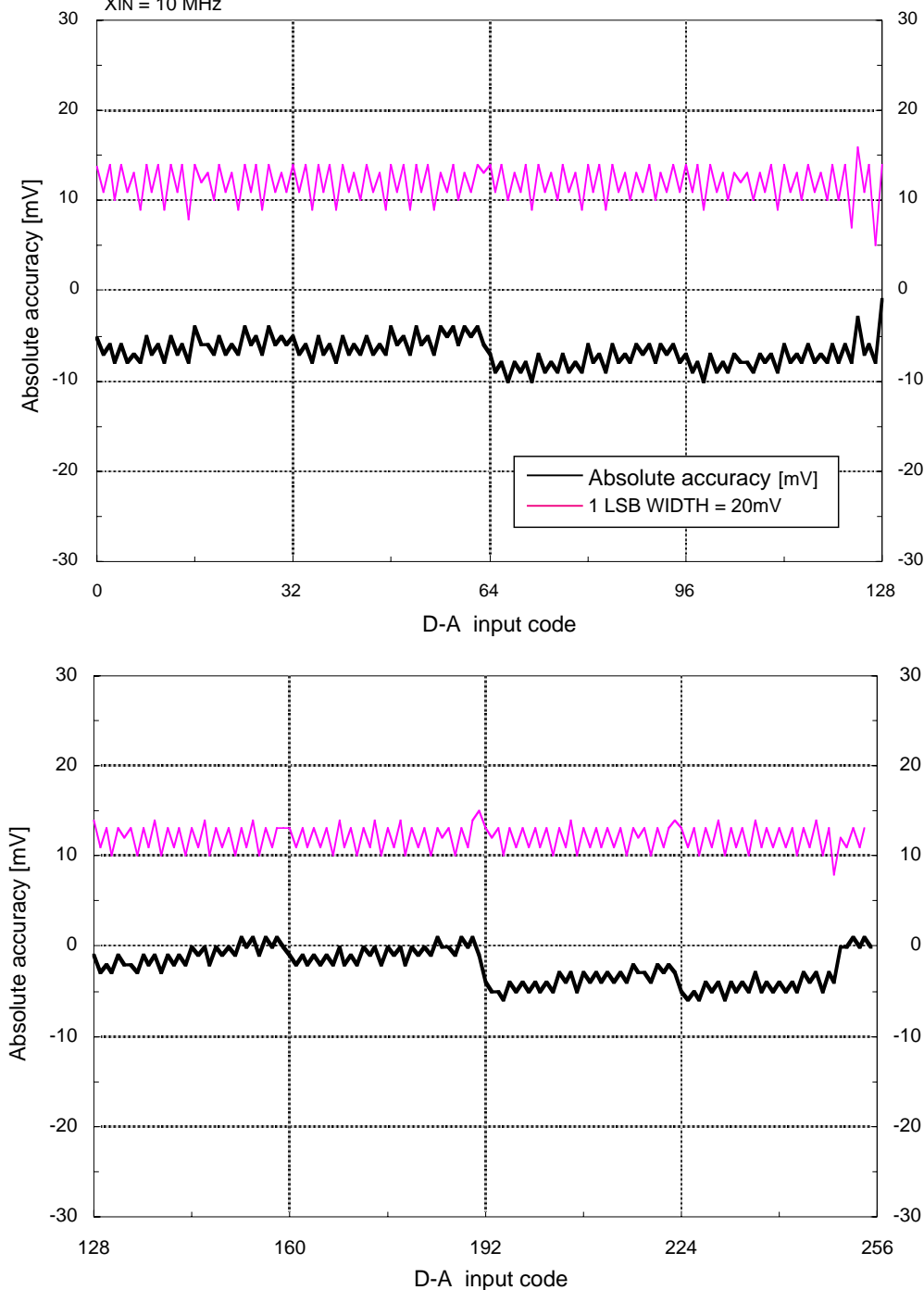
The line on the bottom side of the graph represents absolute errors. This indicates the difference between the measurement and the ideal analog value corresponding to the input code.

The line on the top side of the graph represents the variation width of analog output value corresponding to 1-bit variation in the input code.

Measurement conditions ( $V_{CC} = 5.12V$ ,  $V_{REF} = 5.12V$ ,  $f(X_{IN}) = 10MHz$ ,  $T_a = 25^\circ C$ )

M30218MC 8 bit, characteristics of D-A conversion

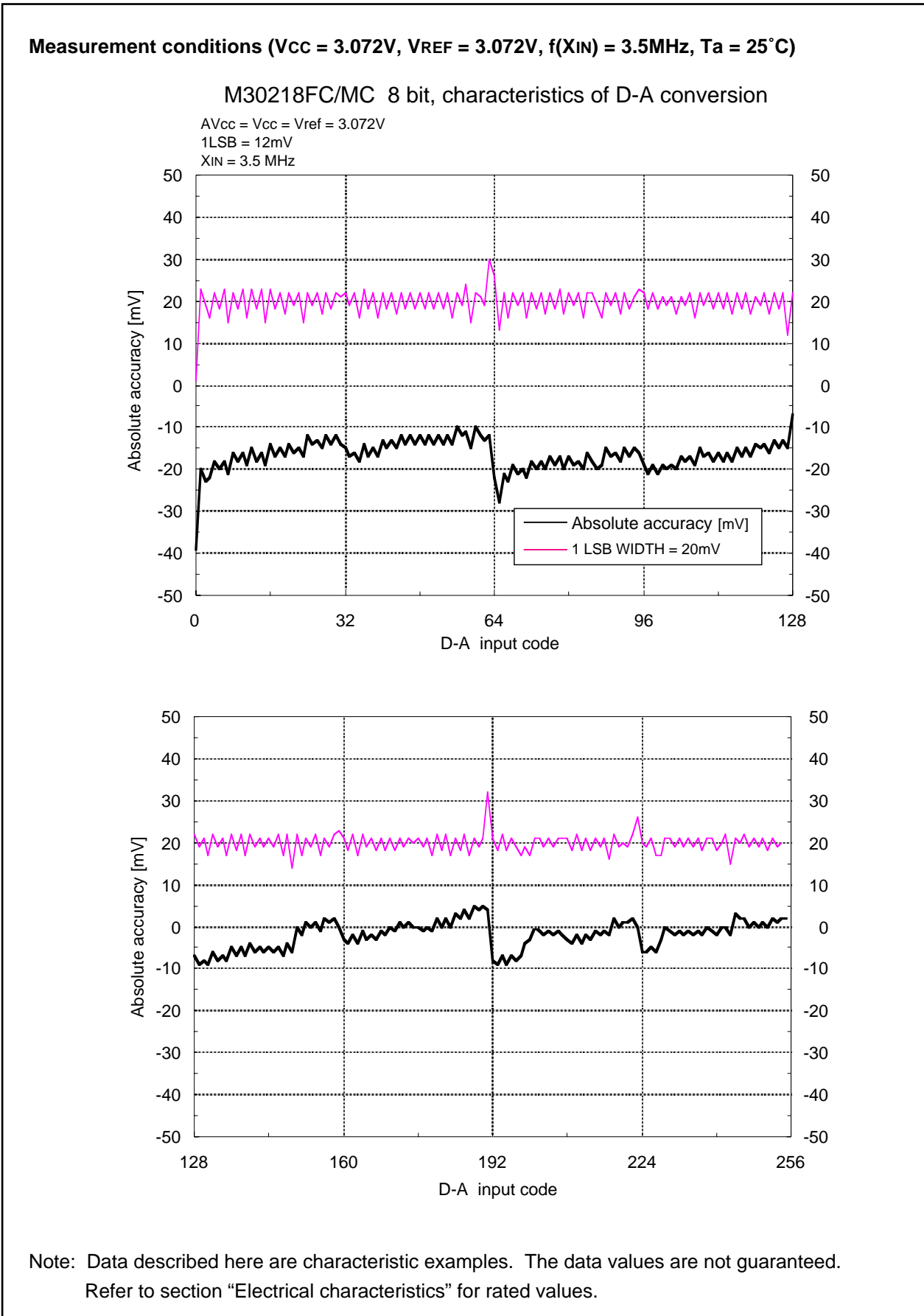
$A_{VCC} = V_{CC} = V_{REF} = 5.12V$   
 1LSB = 20mV  
 $X_{IN} = 10 MHz$



Note: Data described here are characteristic examples. The data values are not guaranteed.  
 Refer to section "Electrical characteristics" for rated values.

Figure 5.3.1. Characteristics of the D-A converter





**Figure 5.3.2. Characteristics of the D-A converter**

## 5.4 Standard Characteristics of Pull-Up Resistor

Figure 5.4.1 shows an example of the standard characteristics of the pull-up resistor.

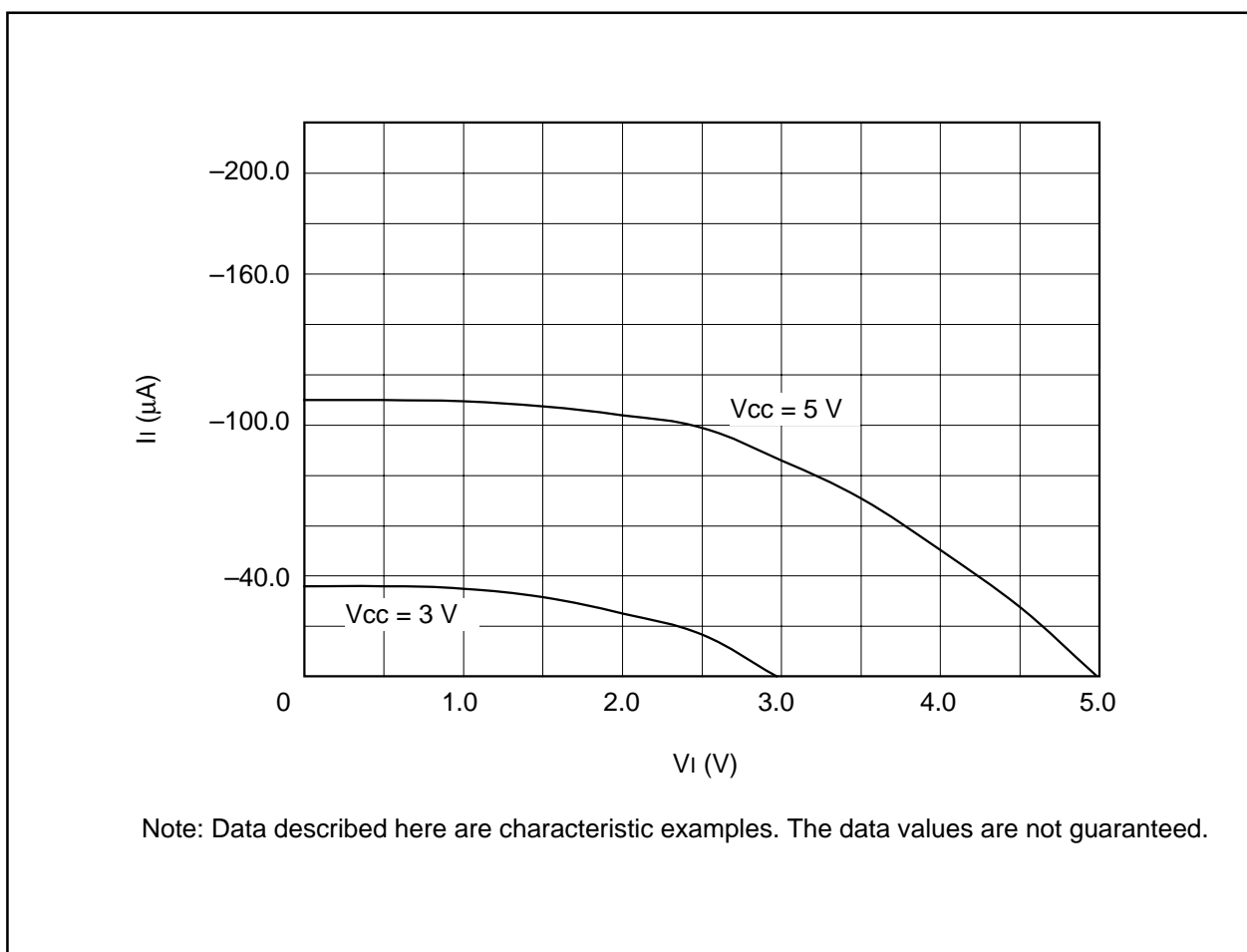


Figure 5.4.1. Example of the standard characteristics of the pull-up resistor

**MITSUBISHI SEMICONDUCTORS  
USER'S MANUAL  
M30218 Group**

---

REV.B Jun. 2001

Edited by  
Committee of editing of Mitsubishi Semiconductor USER'S MANUAL

Published by  
Mitsubishi Electric Corp., Semiconductor Marketing Division

---

This book, or parts thereof, may not be reproduced in any form without permission of Mitsubishi Electric Corporation.

**©2001 MITSUBISHI ELECTRIC CORPORATION**

# User's Manual

## M30218 Group

REVISION DESCRIPTION LIST

M30218 GROUP USER'S MANUAL

Rev. No.	Revision Description	Rev. date
A	First Edition	991125
A1	<p>The followings are updated:</p> <p>Page 56 Figure 39: FLDC mode register b3b2 (at rising edge of each <u>digit</u>) <u>10</u> : 2 X Tdisp</p> <p>Page 447 Figure 5.2.2, Page 450 Figure 5.3.2 : Measurement conditions....f(XIN)=<u>3.5MHz</u></p>	991221
B	<p>The followings are updated:</p> <p>Pages 3, 6, 7, 8, 9, 129, 134, 135, 141, 143, 147, 286, 440, 446, and 448: Delete about mask option specification of pull-down resistor</p> <p>Page 151 Table 71 Automatic transfer serial I/O: Decided electrical standard values (at VCC = 3 V)</p> <p>Page 154 Figure 125 Block diagram of flash memory version: User ROM area block number E0000<sub>16</sub> to E7FFF<sub>16</sub> <u>Block 0</u> ---&gt; <u>Block 3</u> E8000<sub>16</sub> to EFFFF<sub>16</sub> <u>Block 1</u> ---&gt; <u>Block 2</u> F0000<sub>16</sub> to F7FFF<sub>16</sub> <u>Block 2</u> ---&gt; <u>Block 1</u> F8000<sub>16</sub> to FFFFF<sub>16</sub> <u>Block 3</u> ---&gt; <u>Block 0</u></p> <p>Page 155 Figure 126 Flash memory control register bit6 bit5 bit4 0 0 0 : <u>Block 0</u> program/erase ---&gt; <u>Block 3</u> program/erase 0 0 1 : <u>Block 1</u> program/erase ---&gt; <u>Block 2</u> program/erase 0 1 0 : <u>Block 2</u> program/erase ---&gt; <u>Block 1</u> program/erase 0 1 1 : <u>Block 3</u> program/erase ---&gt; <u>Block 0</u> program/erase</p> <p>Page 225 2.3.7 Precautions for Timer B (pulse period/pulse width measurement mode) Line 7 ....This flag <u>cannot</u> be set to "0" by.... ---&gt; ....This flag <u>can</u> be set to "0" by....</p>	010627