

# R32C/161 Group

User's Manual: Hardware

RENESAS MCU

M16C Family / R32C/100 Series

All information contained in these materials, including products and product specifications, represents information on the product at the time of publication and is subject to change by Renesas Electronics Corp. without notice. Please review the latest information published by Renesas Electronics Corp. through various means, including the Renesas Electronics Corp. website (<http://www.renesas.com>).

## Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
3. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below.  
"Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots etc.  
"High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.  

Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical implantations etc.), or may cause serious property damages (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
6. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
11. This document may not be reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

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

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

## General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this manual, refer to the relevant sections of the manual. If the descriptions under General Precautions in the Handling of MPU/MCU Products and in the body of the manual differ from each other, the description in the body of the manual takes precedence.

### 1. Handling of Unused Pins

Handle unused pins in accord with the directions given under Handling of Unused Pins in the manual.

- The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.

### 2. Processing at Power-on

The state of the product is undefined at the moment when power is supplied.

- The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.

In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.

In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

### 3. Prohibition of Access to Reserved Addresses

Access to reserved addresses is prohibited.

- The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

### 4. Clock Signals

After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.

- When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.

### 5. Differences between Products

Before changing from one product to another, i.e. to one with a different part number, confirm that the change will not lead to problems.

- The characteristics of MPU/MCU in the same group but having different part numbers may differ because of the differences in internal memory capacity and layout pattern. When changing to products of different part numbers, implement a system-evaluation test for each of the products.

# About This Manual

## 1. Purpose and Target User

This manual is designed to be read primarily by application developers who have an understanding of this microcomputer (MCU) including its hardware functions and electrical characteristics. The user should have a basic understanding of electric circuits, logic circuits and, MCUs.

This manual consists of 29 chapters covering six main categories: Overview, CPU, System Control, Peripherals, Electrical Characteristics, and Usage Notes.

Carefully read all notes in this document prior to use. Notes are found throughout each chapter, at the end of each chapter, and in the dedicated Usage Notes chapter.

The Revision History at the end of this manual summarizes primary modifications and additions to the previous versions. For details, please refer to the relative chapters or sections of this manual.

The R32C/161 Group includes the documents listed below. Verify this manual is the latest version by visiting the Renesas Electronics website.

Type of Document	Contents	Document Name	Document Number
Datasheet	Overview of Hardware and Electrical Characteristics	R32C/161 Group Datasheet	R01DS0078EJ0120
User's Manual: Hardware	Specifications and detailed descriptions of: -pin layout -memory map -peripherals -electrical characteristics -timing characteristics Refer to the Application Manual for peripheral usage.	R32C/161 Group User's Manual: Hardware	This publication
User's Manual: Software/Software Manual	Descriptions of instruction set	R32C/100 Series Software Manual	REJ09B0267-0100
Application Note	-Usages -Applications -Sample programs -Programming technics using Assembly language or C programming language	Available on the Renesas Electronics website.	
Renesas Technical Update	Bulletins on product specifications, documents, etc.		

## 2. Numbers and Symbols

The following explains the denotations used in this manual for registers, bits, pins and various numbers.

(1) Registers, bits, and pins

Registers, bits, and pins are indicated by symbols. Each symbol has a register/bit/pin identifier after the symbol.

Example: PM03 bit in the PM0 register

P3\_5 pin, VCC pin

(2) Numbers

A binary number has the suffix "b" except for a 1-bit value.

A hexadecimal number has the suffix "h".

A decimal number has no suffix.

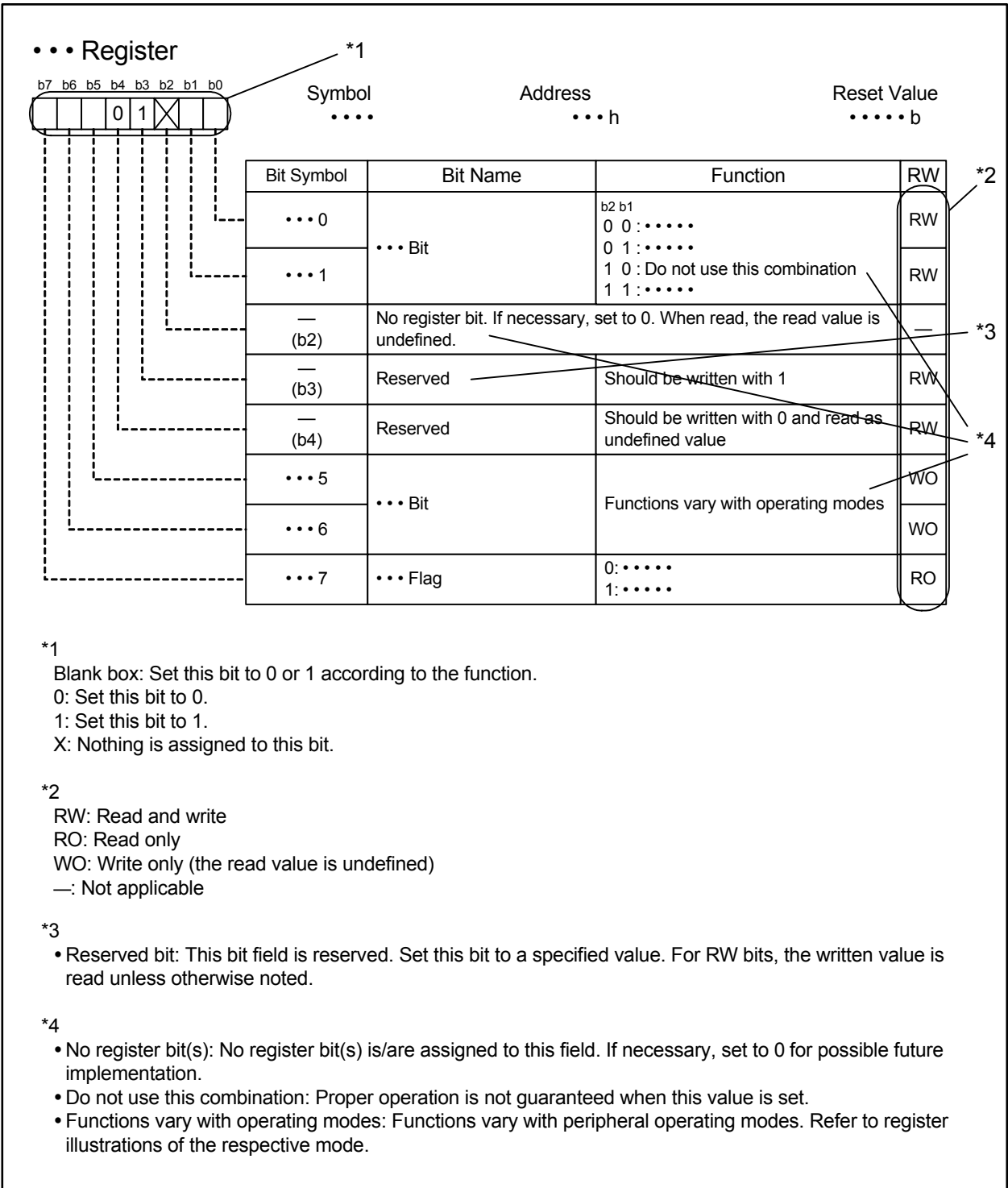
Example: Binary notation: 11b

Hexadecimal notation: EFA0h

Decimal notation: 1234

### 3. Registers

The following illustration describes registers used throughout this manual.



## 4. Abbreviations and Acronyms

The following acronyms and terms are used throughout this manual.

Abbreviation/Acronym	Meaning
ACIA	Asynchronous Communication Interface Adapter
bps	bits per second
CRC	Cyclic Redundancy Check
DMA	Direct Memory Access
DMAC	Direct Memory Access Controller
GSM	Global System for Mobile Communications
Hi-Z	High Impedance
IEBus	Inter Equipment Bus
I/O	Input/Output
IrDA	Infrared Data Association
LSB	Least Significant Bit
MSB	Most Significant Bit
NC	Non-Connection
PLL	Phase Locked Loop
PWM	Pulse Width Modulation
SIM	Subscriber Identity Module
UART	Universal Asynchronous Receiver/Transmitter
VCO	Voltage Controlled Oscillator

# TABLE OF CONTENTS

1.	Overview	1
1.1	Features.....	1
1.1.1	Applications .....	1
1.1.2	Performance Overview .....	2
1.2	Product Information .....	4
1.3	Block Diagram .....	6
1.4	Pin Assignment.....	7
1.5	Pin Definitions and Functions .....	10
2.	Central Processing Unit (CPU)	12
2.1	General Purpose Registers .....	13
2.1.1	Data Registers (R2R0, R3R1, R6R4, and R7R5).....	13
2.1.2	Address Registers (A0, A1, A2, and A3) .....	13
2.1.3	Static Base Register (SB).....	13
2.1.4	Frame Base Register (FB).....	13
2.1.5	Program Counter (PC).....	13
2.1.6	Interrupt Vector Table Base Register (INTB) .....	13
2.1.7	User Stack Pointer (USP) and Interrupt Stack Pointer (ISP) .....	13
2.1.8	Flag Register (FLG).....	13
2.2	Fast Interrupt Registers .....	15
2.2.1	Save Flag Register (SVF).....	15
2.2.2	Save PC Register (SVP) .....	15
2.2.3	Vector Register (VCT) .....	15
2.3	DMAC-associated Registers.....	15
2.3.1	DMA Mode Registers (DMD0, DMD1, DMD2, and DMD3) .....	15
2.3.2	DMA Terminal Count Registers (DCT0, DCT1, DCT2, and DCT3) .....	15
2.3.3	DMA Terminal Count Reload Registers (DCR0, DCR1, DCR2, and DCR3) .....	15
2.3.4	DMA Source Address Registers (DSA0, DSA1, DSA2, and DSA3).....	15
2.3.5	DMA Source Address Reload Registers (DSR0, DSR1, DSR2, and DSR3).....	15
2.3.6	DMA Destination Address Registers (DDA0, DDA1, DDA2, and DDA3) .....	15
2.3.7	DMA Destination Address Reload Registers (DDR0, DDR1, DDR2, and DDR3) .....	15
3.	Memory	16
4.	Special Function Registers (SFRs)	17
5.	Resets	70
5.1	Hardware Reset.....	70
5.2	Software Reset .....	72
5.3	Watchdog Timer Reset .....	72
5.4	Reset Vector .....	73



<b>6.</b>	<b>Power Management</b>	<b>74</b>
6.1	Voltage Regulators for Internal Logic.....	74
6.1.1	Decoupling Capacitor .....	75
6.2	Low Voltage Detector.....	76
6.2.1	Operational State of Low Voltage Detector.....	79
6.2.2	Low Voltage Detection Interrupt .....	79
6.2.3	Application Example of the Low Voltage Detector.....	80
<b>7.</b>	<b>Clock Generator</b>	<b>81</b>
7.1	Clock Generator Types.....	81
7.1.1	Main Clock.....	90
7.1.2	Sub Clock (fC).....	91
7.1.3	PLL Clock .....	92
7.1.4	On-chip Oscillator Clock .....	95
7.2	Oscillator Stop Detection .....	96
7.2.1	How to Use Oscillator Stop Detection.....	96
7.3	Base Clock.....	96
7.4	CPU Clock and Peripheral Bus Clock.....	97
7.5	Peripheral Clock .....	97
7.6	Clock Output Function .....	98
7.7	Power Control.....	99
7.7.1	Normal Operating Mode .....	100
7.7.2	Wait Mode.....	105
7.7.3	Stop Mode .....	108
7.8	System Clock Protection.....	110
7.9	Notes on Clock Generator .....	111
7.9.1	Sub Clock .....	111
7.9.2	Power Control.....	111
<b>8.</b>	<b>Bus</b>	<b>112</b>
8.1	Bus Setting .....	112
8.2	Peripheral Bus Timing Setting .....	113
<b>9.</b>	<b>Protection</b>	<b>114</b>
9.1	Protect Register (PRCR Register).....	114
9.2	Protect Register 2 (PRCR2 Register).....	115
9.3	Protect Register 3 (PRCR3 Register).....	115
9.4	Protect Register 4 (PRCR4 Register).....	116
9.5	Protect Release Register (PRR Register).....	117
<b>10.</b>	<b>Interrupts</b>	<b>118</b>
10.1	Interrupt Types.....	118
10.2	Software Interrupts .....	119

10.3	Hardware Interrupts .....	120
10.3.1	Special Interrupts .....	120
10.3.2	Peripheral Interrupts .....	120
10.4	Fast Interrupt .....	121
10.5	Interrupt Vectors .....	121
10.5.1	Fixed Vector Table .....	122
10.5.2	Relocatable Vector Table .....	122
10.6	Interrupt Request Acceptance .....	127
10.6.1	I Flag and IPL .....	127
10.6.2	Interrupt Control Registers .....	128
10.6.3	Wake-up IPL Setting Register .....	131
10.6.4	Interrupt Sequence .....	132
10.6.5	Interrupt Response Time .....	133
10.6.6	IPL after Accepting an Interrupt Request .....	134
10.6.7	Register Saving .....	134
10.7	Register Restoring from Interrupt Handler .....	135
10.8	Interrupt Priority .....	135
10.9	Priority Resolver .....	135
10.10	External Interrupt .....	137
10.11	NMI .....	138
10.12	Intelligent I/O Interrupt .....	139
10.13	Notes on Interrupts .....	142
10.13.1	ISP Setting .....	142
10.13.2	NMI .....	142
10.13.3	External Interrupts .....	142
11.	Watchdog Timer .....	143
12.	DMAC .....	147
12.1	Transfer Cycle .....	156
12.1.1	Effect of Transfer Address and Data Bus Width .....	156
12.1.2	Effect of Bus Timing .....	157
12.2	DMA Transfer Cycle .....	159
12.3	Channel Priority and DMA Transfer Timing .....	160
12.4	Notes on DMAC .....	161
12.4.1	DMAC-associated Register Settings .....	161
12.4.2	Reading DMAC-associated Registers .....	161
13.	DMAC II .....	162
13.1	DMAC II Settings .....	162
13.1.1	Registers RIPL1 and RIPL2 .....	163
13.1.2	DMAC II Index .....	164

13.1.3	Interrupt Control Register of the Peripherals .....	167
13.1.4	Relocatable Vector Table of the Peripherals.....	167
13.1.5	IRLT Bit in the IIOiE Register (i = 0 to 11).....	167
13.2	DMAC II Operation .....	167
13.3	Transfer Types.....	167
13.3.1	Memory-to-memory Transfer .....	167
13.3.2	Immediate Data Transfer .....	168
13.3.3	Calculation Result Transfer .....	168
13.4	Transfer Modes.....	168
13.4.1	Single Transfer .....	168
13.4.2	Burst Transfer .....	168
13.4.3	Multiple Transfer.....	168
13.5	Chain Transfer .....	169
13.6	DMA II Transfer Complete Interrupt.....	169
13.7	Execution Time .....	170
<b>14.</b>	<b>Programmable I/O Ports</b> .....	<b>171</b>
14.1	Port Pi Register (Pi register, i = 0 to 9) .....	173
<b>15.</b>	<b>Timers</b> .....	<b>174</b>
15.1	Timer A .....	176
15.1.1	Timer Mode.....	183
15.1.2	Event Counter Mode.....	185
15.1.3	One-shot Timer Mode.....	189
15.1.4	Pulse-width Modulation Mode.....	191
15.2	Timer B .....	194
15.2.1	Timer Mode.....	197
15.2.2	Event Counter Mode.....	199
15.2.3	Pulse Period/Pulse-width Measure Mode.....	202
15.3	Notes on Timers.....	205
15.3.1	Timer A and Timer B.....	205
15.3.2	Timer A .....	205
15.3.3	Timer B .....	207
<b>16.</b>	<b>Three-phase Motor Control Timers</b> .....	<b>208</b>
16.1	Modulation Modes of Three-phase Motor Control Timers .....	214
16.2	Timer B2 .....	215
16.3	Timers A4, A1, and A2.....	217
16.4	Simultaneous Conduction Prevention and Dead Time Timer .....	221
16.5	Three-phase Motor Control Timer Operation.....	222
16.6	Notes on Three-phase Motor Control Timers .....	225
16.6.1	Shutdown.....	225

16.6.2	Register Setting .....	225
<b>17.</b>	<b>Serial Interface</b>	<b>226</b>
17.1	Synchronous Serial Interface Mode.....	240
17.1.1	Reset Procedure on Transmit/Receive Error.....	245
17.1.2	CLK Polarity.....	245
17.1.3	LSB First and MSB First Selection .....	246
17.1.4	Continuous Receive Mode .....	246
17.1.5	Serial Data Logic Inversion.....	247
17.1.6	CTS/RTS Function.....	247
17.2	Asynchronous Serial Interface Mode (UART Mode).....	248
17.2.1	Bit Rate .....	253
17.2.2	Reset Procedure on Transmit/Receive Error.....	254
17.2.3	LSB First and MSB First Selection .....	254
17.2.4	Serial Data Logic Inversion.....	255
17.2.5	TXD and RXD I/O Polarity Inversion .....	256
17.2.6	CTS/RTS Function.....	256
17.3	Special Mode 1 (I <sup>2</sup> C Mode).....	257
17.3.1	Start Condition and Stop Condition Detection .....	263
17.3.2	Start Condition and Stop Condition Generation.....	263
17.3.3	Arbitration .....	264
17.3.4	SCL Control and Clock Synchronization .....	265
17.3.5	SDA Output .....	267
17.3.6	SDA Input .....	267
17.3.7	Acknowledge .....	267
17.3.8	Initialization of Transmit/Receive Operation Reset.....	267
17.4	Special Mode 2 .....	268
17.4.1	$\overline{SS}_i$ Input Pin Function (i = 0 to 2).....	270
17.4.2	Clock Phase Setting .....	271
17.5	Notes on Serial Interface .....	273
17.5.1	Changing the UiBRG Register (i = 0 to 4) .....	273
17.5.2	Pin Output .....	273
17.5.3	Synchronous Serial Interface Mode .....	273
17.5.4	Special Mode 1 (I <sup>2</sup> C Mode) .....	273
17.5.5	Reset Procedure on Communication Error.....	274
<b>18.</b>	<b>A/D Converter</b>	<b>275</b>
18.1	Mode Descriptions .....	283
18.1.1	One-shot Mode.....	283
18.1.2	Repeat Mode .....	284
18.1.3	Single Sweep Mode.....	285
18.1.4	Repeat Sweep Mode 0 .....	286

18.1.5	Repeat Sweep Mode 1 .....	287
18.2	Functions .....	288
18.2.1	Resolution Selection .....	288
18.2.2	Sample and Hold Function .....	288
18.2.3	Trigger Selection.....	288
18.2.4	DMAC Operating Mode .....	288
18.2.5	Function-extended Analog Input Pins.....	289
18.2.6	External Operating Amplifier (Op-Amp) Connection Mode.....	289
18.2.7	Self Test/Open-circuit Detection Assist.....	290
18.2.8	Power Saving .....	291
18.2.9	Output Impedance of Sensor Equivalent Circuit under A/D Conversion .....	291
18.3	Notes on A/D Converter.....	293
18.3.1	Notes on Designing Boards.....	293
18.3.2	Notes on Programming.....	294
19.	CRC Calculator .....	295
20.	X-Y Conversion .....	298
20.1	Data Conversion When Reading .....	299
20.2	Data Conversion When Writing.....	301
21.	Intelligent I/O .....	302
21.1	Base Timer.....	314
21.2	Time Measurement.....	319
21.3	Waveform Generation .....	323
21.3.1	Single-phase Waveform Output Mode.....	324
21.3.2	Inverted Waveform Output Mode.....	326
21.3.3	Set/Reset Waveform Output Mode (SR Waveform Output Mode) .....	328
21.3.4	Phase Shift Waveform Output Mode .....	331
21.3.5	Digital Debounce Circuit .....	334
22.	Serial Bus Interface .....	336
22.1	Synchronous Serial Communication Mode and 4-wire Serial Bus Mode .....	337
22.1.1	Transmit/Receive Clock.....	349
22.1.2	Transmit/Receive Shift Register .....	351
22.1.3	Data I/O Pin and SS0 Shift Register.....	353
22.1.4	Interrupt Requests .....	354
22.1.5	Communication Modes and Pin Functions .....	355
22.1.6	Synchronous Serial Communication Mode .....	356
22.1.7	4-wire Serial Bus Mode .....	363
22.2	Note on Serial Bus Interface.....	370
22.2.1	Note on Synchronous Serial Communication Mode and 4-wire Serial Bus Mode.....	370

<b>23. LIN Module</b>	<b>371</b>
23.1 LIN Module Associated Registers	373
23.1.1 LIN Common Registers	373
23.1.2 LIN Channel Dedicated Registers	373
23.1.3 Register Window and Channel Switching	374
23.2 Operating Modes	387
23.2.1 LIN Reset Mode	388
23.2.2 LIN Operating Mode	388
23.2.3 LIN Wake-up Mode	388
23.3 Operational Overview	389
23.3.1 Header Transmission	389
23.3.2 Response Transmission	390
23.3.3 Response Reception	391
23.4 Baud Rate Generator	392
23.5 Data Transmission and Reception	394
23.5.1 Data Transmission	394
23.5.2 Data Reception	395
23.6 Buffer Processing of Transmit/Receive Data	396
23.6.1 Transmission of LIN Frame	396
23.6.2 Reception of LIN Frame	397
23.7 Wake-up Transmission and Reception	398
23.7.1 Wake-up Transmission	398
23.7.2 Wake-up Reception	399
23.8 Low Power Mode Control Using Input Signal Low Detection	400
23.9 Operational Status	401
23.10 Error Status	402
23.10.1 Error Status Types	402
23.10.2 LIN Error Detection Targets	403
23.11 LIN Interrupts	404
<b>24. CAN Module</b>	<b>405</b>
24.1 CAN SFRs	408
24.1.1 CANi Control Register (CiCTLR) (i = 0, 1)	409
24.1.2 CANi Clock Select Register (CiCLKR) (i = 0, 1)	413
24.1.3 CANi Bit Configuration Register (CiBCR) (i = 0, 1)	414
24.1.4 CANi Mask Register k (CiMKRk) (i = 0, 1; k = 0 to 7)	416
24.1.5 CANi FIFO Received ID Compare Register n (CiFIDCR0 and CiFIDCR1) (i = 0, 1; n = 0, 1)	417
24.1.6 CANi Mask Invalid Register (CiMKIVLR) (i = 0, 1)	419
24.1.7 CANi Mailbox (CiMBj) (i = 0, 1; j = 0 to 31)	420
24.1.8 CANi Mailbox Interrupt Enable Register (CiMIER) (i = 0, 1)	424
24.1.9 CANi Message Control Register j (CiMCTLj) (i = 0, 1; j = 0 to 31)	425

24.1.10	CANi Receive FIFO Control Register (CiRFCR) (i = 0, 1).....	428
24.1.11	CANi Receive FIFO Pointer Control Register (CiRFPCR) (i = 0, 1).....	431
24.1.12	CANi Transmit FIFO Control Register (CiTFCR) (i = 0, 1).....	432
24.1.13	CANi Transmit FIFO Pointer Control Register (CiTFPCR) (i = 0, 1).....	434
24.1.14	CANi Status Register (CiSTR) (i = 0, 1).....	435
24.1.15	CANi Mailbox Search Mode Register (CiMSMR) (i = 0, 1).....	438
24.1.16	CANi Mailbox Search Status Register (CiMSSR) (i = 0, 1).....	439
24.1.17	CANi Channel Search Support Register (CiCSSR) (i = 0, 1).....	441
24.1.18	CANi Acceptance Filter Support Register (CiAFSR) (i = 0, 1).....	442
24.1.19	CANi Error Interrupt Enable Register (CiEIER) (i = 0, 1).....	443
24.1.20	CANi Error Interrupt Factor Judge Register (CiEIFR) (i = 0, 1).....	445
24.1.21	CANi Receive Error Count Register (CiRECR) (i = 0, 1).....	448
24.1.22	CANi Transmit Error Count Register (CiTECR) (i = 0, 1).....	449
24.1.23	CANi Error Code Store Register (CiECSR) (i = 0, 1).....	450
24.1.24	CANi Time Stamp Register (CiTSR) (i = 0, 1).....	452
24.1.25	CANi Test Control Register (CiTCR) (i = 0, 1).....	453
24.2	Operating Modes .....	456
24.2.1	CAN Reset Mode.....	457
24.2.2	CAN Halt Mode.....	458
24.2.3	CAN Sleep Mode.....	459
24.2.4	CAN Operation Mode (Excluding Bus-off State).....	460
24.2.5	CAN Operation Mode (Bus-off State).....	461
24.3	CAN Communication Speed Configuration.....	462
24.3.1	CAN Clock Configuration.....	462
24.3.2	Bit Timing Configuration .....	462
24.3.3	Bit rate .....	463
24.4	Mailbox and Mask Register Structure .....	464
24.5	Acceptance Filtering and Masking Function .....	466
24.6	Reception and Transmission .....	469
24.6.1	Reception .....	470
24.6.2	Transmission .....	472
24.7	CAN Interrupts .....	473
25.	I/O Pins .....	474
25.1	Port Pi Direction Register (PDi Register, i = 0 to 9).....	475
25.2	Output Function Select Registers .....	476
25.3	Input Function Select Registers.....	487
25.4	Pull-up Control Registers 0 to 2 (Registers PUR0 to PUR2).....	490
25.5	Port Control Register (PCR Register).....	492
25.6	Configuring Unused Pins .....	493

<b>26. Flash Memory</b>	<b>495</b>
26.1 Overview .....	495
26.2 Flash Memory Protection .....	497
26.2.1 Lock Bit Protection .....	497
26.2.2 ROM Code Protection .....	497
26.2.3 ID Code Protection .....	498
26.3 CPU Rewrite Mode .....	499
26.3.1 Flash Memory Rewrite Bus Timing .....	506
26.3.2 Software Commands .....	510
26.3.3 Mode Transition .....	511
26.3.4 Issuing Software Commands .....	512
26.3.5 Status Check .....	518
26.4 Standard Serial I/O Mode .....	519
26.5 Parallel I/O mode .....	522
26.6 Notes on Flash Memory Rewriting .....	523
26.6.1 Note on Power Supply .....	523
26.6.2 Note on Hardware Reset .....	523
26.6.3 Note on Flash Memory Protection .....	523
26.6.4 Notes on Programming .....	523
26.6.5 Notes on Interrupts .....	523
26.6.6 Notes on Rewrite Control Program .....	524
26.6.7 Notes on Number of Program/Erase Cycles and Software Command Execution Time .....	524
26.6.8 Other Notes .....	524
<b>27. E<sup>2</sup>PROM Emulation Data Flash</b>	<b>525</b>
27.1 Overview .....	525
27.2 Block Configuration .....	529
27.3 Operational Procedures .....	530
27.4 Note on E <sup>2</sup> dataFlash .....	533
<b>28. Electrical Characteristics</b>	<b>534</b>
<b>29. Usage Notes</b>	<b>569</b>
29.1 Notes on Board Designing .....	569
29.1.1 Power Supply Pins .....	569
29.1.2 Supply Voltage .....	569
29.2 Notes on Register Setting .....	570
29.2.1 Registers with Write-only Bits .....	570
29.3 Notes on Clock Generator .....	572
29.3.1 Sub Clock .....	572
29.3.2 Power Control .....	572
29.4 Notes on Interrupts .....	573



29.4.1	ISP Setting.....	573
29.4.2	NMI .....	573
29.4.3	External Interrupts .....	573
29.5	Notes on DMAC.....	574
29.5.1	DMAC-associated Register Settings .....	574
29.5.2	Reading DMAC-associated Registers .....	574
29.6	Notes on Timers.....	575
29.6.1	Timer A and Timer B.....	575
29.6.2	Timer A .....	575
29.6.3	Timer B .....	577
29.7	Notes on Three-phase Motor Control Timers .....	578
29.7.1	Shutdown.....	578
29.7.2	Register Setting .....	578
29.8	Notes on Serial Interface .....	579
29.8.1	Changing the UiBRG Register (i = 0 to 4) .....	579
29.8.2	Pin Output .....	579
29.8.3	Synchronous Serial Interface Mode .....	579
29.8.4	Special Mode 1 (I <sup>2</sup> C Mode) .....	579
29.8.5	Reset Procedure on Communication Error.....	580
29.9	Notes on A/D Converter.....	581
29.9.1	Notes on Designing Boards.....	581
29.9.2	Notes on Programming.....	582
29.10	Note on Serial Bus Interface.....	583
29.10.1	Note on Synchronous Serial Communication Mode and 4-wire Serial Bus Mode.....	583
29.11	Notes on Flash Memory Rewriting.....	584
29.11.1	Note on Power Supply.....	584
29.11.2	Note on Hardware Reset .....	584
29.11.3	Note on Flash Memory Protection .....	584
29.11.4	Notes on Programming.....	584
29.11.5	Notes on Interrupts .....	584
29.11.6	Notes on Rewrite Control Program.....	585
29.11.7	Notes on Number of Program/Erase Cycles and Software Command Execution Time .....	585
29.11.8	Other Notes .....	585
29.12	Note on E <sup>2</sup> dataFlash .....	586
Appendix 1. Package Dimensions		587
INDEX		588

## **1. Overview**

### **1.1 Features**

The M16C Family offers a robust platform of 32-/16-bit CISC microcomputers (MCUs) featuring high ROM code efficiency, extensive EMI/EMS noise immunity, ultra-low power consumption, high-speed processing in actual applications, and numerous and varied integrated peripherals. Extensive device scalability from low- to high-end, featuring a single architecture as well as compatible pin assignments and peripheral functions, provides support for a vast range of application fields.

The R32C/100 Series is a high-end microcontroller series in the M16C Family. With a 4-Gbyte memory space, it achieves maximum code efficiency and high-speed processing with 32-bit CISC architecture, multiplier, multiply-accumulate unit, and floating point unit. The selection from the broadest choice of on-chip peripheral devices — UART, CRC, DMAC, A/D converter, timers, I<sup>2</sup>C, and watchdog timer enables to minimize external components.

The R32C/100 Series, in particular, provides the R32C/161 Group, a product specific to vehicle network. This product, provided as 80-pin plastic molded LQFP package, has two channels of CAN module, one channel of LIN module, and standard peripherals.

#### **1.1.1 Applications**

Automotive, audio, communication equipment, industrial equipment, etc.

### 1.1.2 Performance Overview

Tables 1.1 and 1.2 show the performance overview of the R32C/161 Group.

**Table 1.1 Performance Overview (1/2)**

Unit	Function	Explanation
CPU	Central processing unit	R32C/100 Series CPU Core <ul style="list-style-type: none"> <li>• Basic instructions: 108</li> <li>• Minimum instruction execution time: 20.83 ns (<math>f(\text{CPU}) = 48 \text{ MHz}</math>)</li> <li>• Multiplier: 32-bit <math>\times</math> 32-bit <math>\rightarrow</math> 64-bit</li> <li>• Multiply-accumulate unit: 32-bit <math>\times</math> 32-bit + 64-bit <math>\rightarrow</math> 64-bit</li> <li>• IEEE-754 compatible FPU: Single precision</li> <li>• 32-bit barrel shifter</li> <li>• Operating mode: Single-chip mode</li> </ul>
Memory		Flash memory: 128/256 Kbytes RAM: 12/20 Kbytes Data flash: 4 Kbytes $\times$ 2 blocks E <sup>2</sup> dataFlash: none <sup>(1)</sup> /4 Kbytes Refer to Table 1.3 for details
Voltage Detector	Low voltage detector	Optional <sup>(2)</sup> Low voltage detection interrupt
Clock	Clock generator	<ul style="list-style-type: none"> <li>• 4 circuits (main clock, sub clock, PLL, on-chip oscillator)</li> <li>• Oscillation stop detector: Main clock oscillator stop/restart detection</li> <li>• Frequency divide circuit: Divide-by-2 to divide-by-24 selectable</li> <li>• Low power modes: Wait mode, stop mode</li> </ul>
Interrupts		Interrupt vectors: 261 External interrupt inputs: $\overline{\text{NMI}}$ , $\overline{\text{INT}} \times 6$ Interrupt priority levels: 7
Watchdog Timer		15 bits $\times$ 1 (selectable input frequency from prescaler output) Automatic timer start function is available
DMA	DMAC	4 channels <ul style="list-style-type: none"> <li>• Cycle-steal transfer mode</li> <li>• Request sources: 44</li> <li>• 2 transfer modes: Single transfer, repeat transfer</li> </ul>
	DMAC II	<ul style="list-style-type: none"> <li>• Triggered by an interrupt request of any peripheral</li> <li>• 3 characteristic transfer functions: Immediate data transfer, calculation result transfer, chain transfer</li> </ul>
I/O Ports	Programmable I/O ports	<ul style="list-style-type: none"> <li>• 2 input-only ports</li> <li>• 64 CMOS I/O ports</li> <li>• A pull-up resistor is selectable for every 4 input ports</li> </ul>

Notes:

1. Contact a Renesas Electronics sales office to use the non-E<sup>2</sup>dataFlash version.
2. Contact a Renesas Electronics sales office to use the optional features.

**Table 1.2 Performance Overview (2/2)**

Unit	Function	Explanation
Timer	Timer A	16-bit timer × 5 Timer mode, event counter mode, one-shot timer mode, pulse-width modulation (PWM) mode Two-phase pulse signal processing in event counter mode (two-phase encoder input) × 3
	Timer B	16-bit timer × 6 Timer mode, event counter mode, pulse frequency measurement mode, pulse-width measurement mode
	Three-phase motor control timer	Three-phase motor control timer × 1 (timers A1, A2, A4, and B2 used) 8-bit programmable dead time timer
Serial Interface	UART0 to UART4	Asynchronous/synchronous serial interface × 5 channels • I <sup>2</sup> C-bus (UART0 to UART2) • Special mode 2 (UART0 to UART2)
A/D Converter		10-bit resolution × 23 channels Sample and hold functionality integrated Self test/Open-circuit detection assist
CRC Calculator		CRC-CCITT ( $X^{16} + X^{12} + X^5 + 1$ )
X-Y Converter		16 bits × 16 bits
Intelligent I/O		Time measurement (input capture): 16 bits × 8 Digital debounce circuit contained Waveform generation (output compare): 16 bits × 8 Phase shift waveform output mode contained
Serial Bus Interface		1 channel • Synchronous serial communication mode • 4-wire serial bus mode Programmable character length: 8 to 16 bits
LIN Module		1 channel
CAN Module		2 channels CAN functionality compliant with ISO 11898-1 32 mailboxes
Flash Memory		Programming and erasure supply voltage: VCC = 3.0 to 5.5 V Minimum endurance: 1,000 program/erase cycles Security protection: ROM code protect, ID code protect Debugging: On-chip debug, on-board flash programming
E <sup>2</sup> dataFlash		Minimum endurance: 100,000 program/erase cycles
Operating Frequency/Supply Voltage		48 MHz/VCC = 3.0 to 5.5 V
Operating Temperature		-40°C to 85°C (J version) -40°C to 105°C (L version) <sup>(1)</sup> -40°C to 125°C (K version)
Current Consumption		31 mA (VCC = 5.0 V, f(CPU) = 48 MHz) 8 μA (VCC = 3.3 V, f(XCIN) = 32.768 kHz, wait mode)
Package		80-pin plastic molded LQFP (PLQP0080KB-A)

Note:

1. Contact a Renesas Electronics sales office to use the L version products.

## 1.2 Product Information

Table 1.3 lists the product information and Figure 1.1 shows the details of the part number.

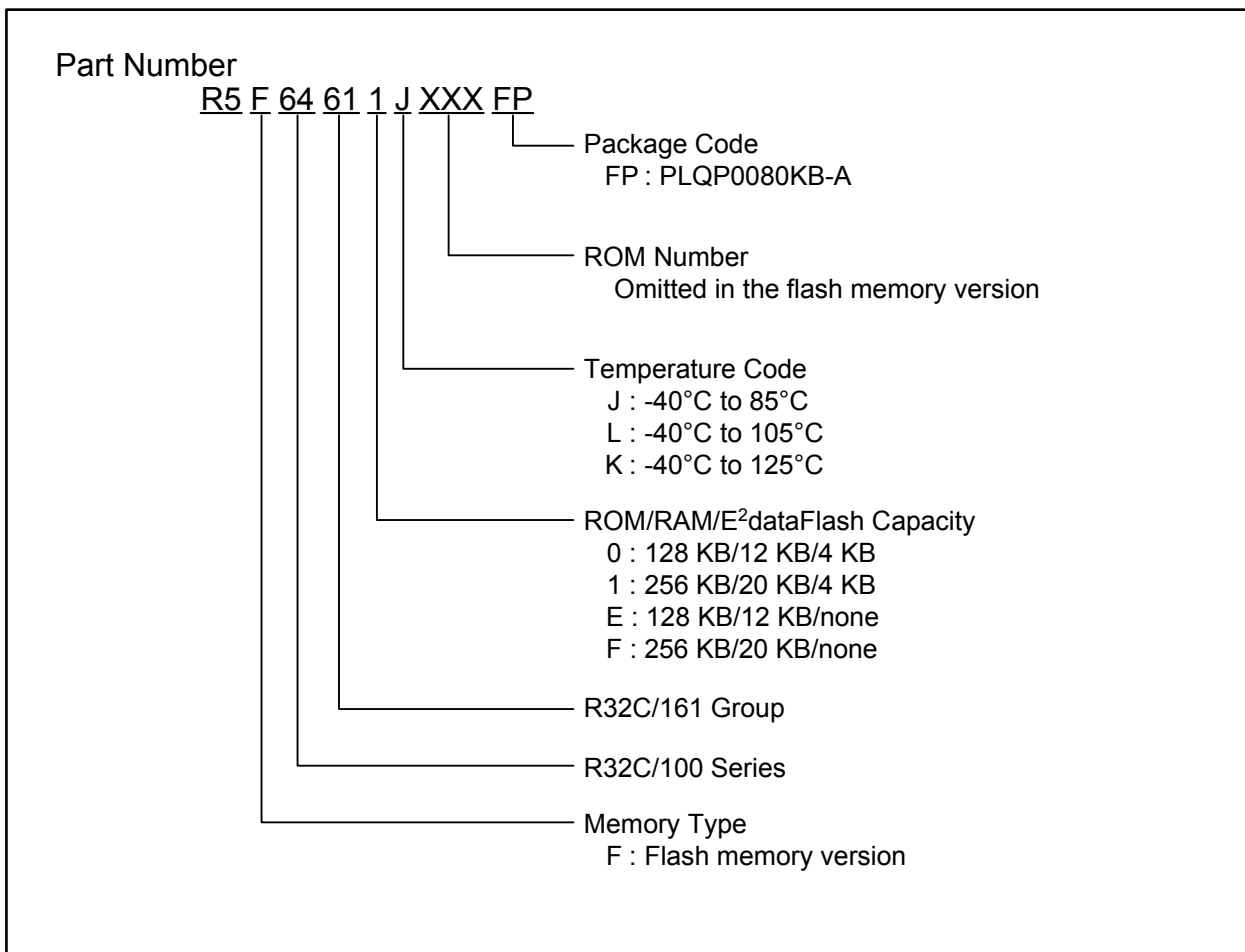
**Table 1.3 R32C/161 Group Product List**

**As of July, 2012**

Part Number	Package Code (1)	ROM Capacity (2)	RAM Capacity	E <sup>2</sup> dataFlash	Remarks
R5F64610JFP	PLQP0080KB-A	128 Kbytes + 8 Kbytes	12 Kbytes	4 Kbytes	J Version
R5F64610LFP					L Version (3)
R5F64610KFP					K Version
R5F6461EJFP				NA (3)	J Version
R5F6461ELFP					L Version (3)
R5F6461EKFP					K Version
R5F64611JFP		256 Kbytes + 8 Kbytes	20 Kbytes	4 Kbytes	J Version
R5F64611LFP					L Version (3)
R5F64611KFP					K Version
R5F6461FJFP				NA (3)	J Version
R5F6461FLFP					L Version (3)
R5F6461FKFP					K Version

Notes:

- The old package code is as follows:  
PLQP0080KB-A: 80P6Q-A
- "+ 8 Kbytes" in the ROM capacity column indicates the data flash capacity.
- Contact a Renesas Electronics sales office to use the non-E<sup>2</sup>dataFlash version or the L version products.



**Figure 1.1 Part Numbering**

### 1.3 Block Diagram

Figure 1.2 shows a block diagram of the R32C/161 Group.

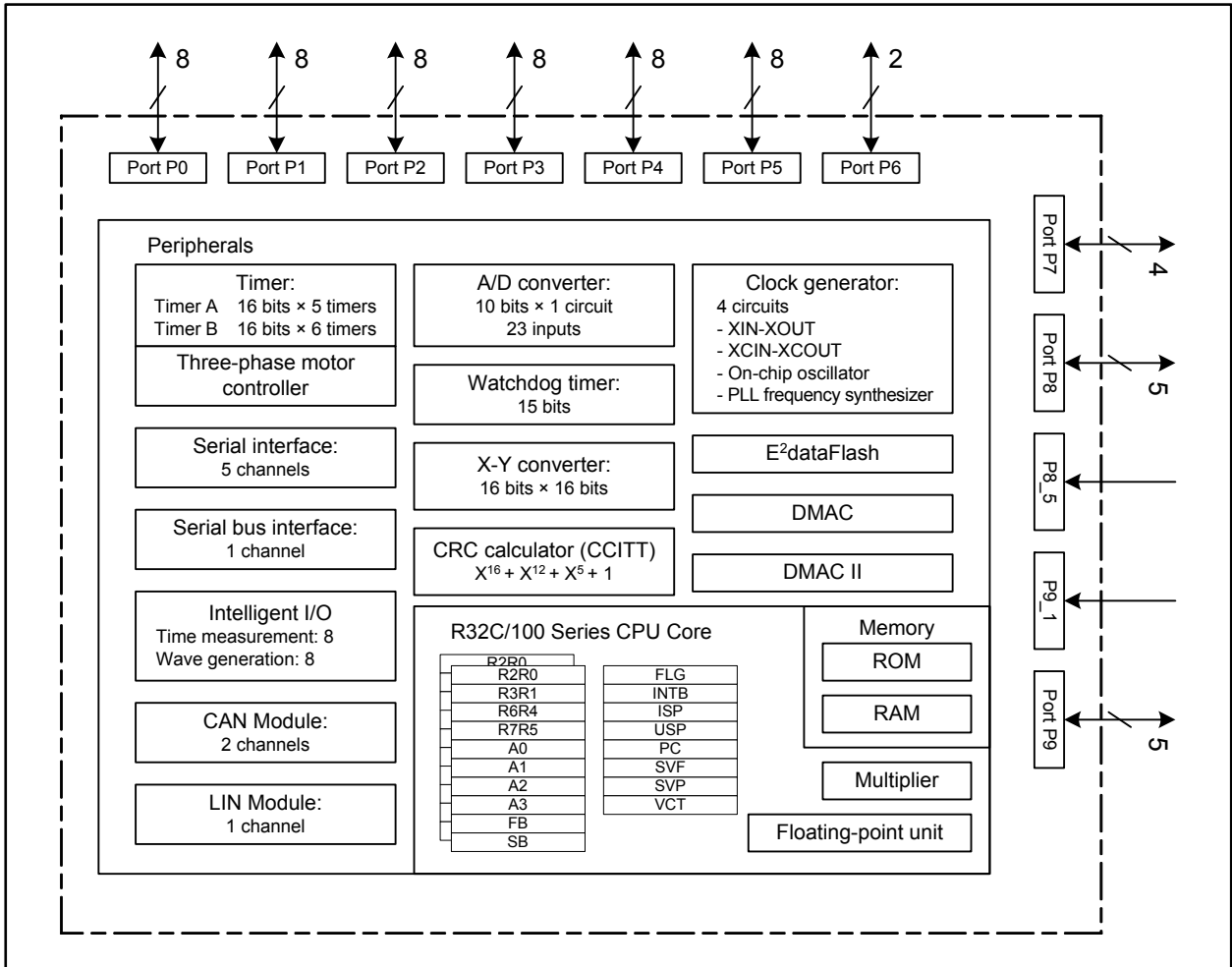
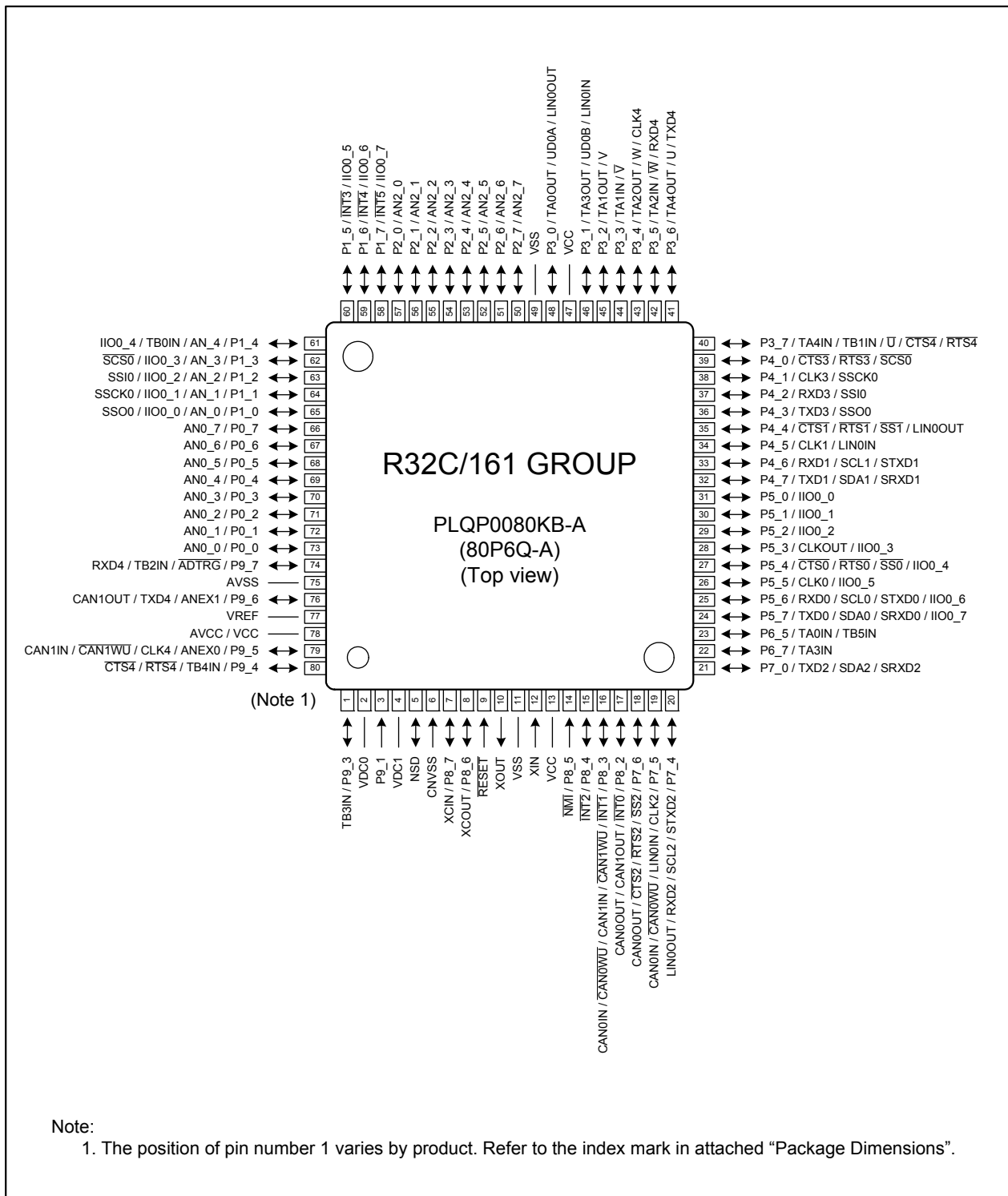


Figure 1.2 R32C/161 Group Block Diagram

### 1.4 Pin Assignment

Figure 1.3 shows the pin assignment (top view) and Tables 1.4 and 1.5 show the pin characteristics.



**Figure 1.3 Pin Assignment (top view)**



**Table 1.4 Pin Characteristics (1/2)**

Pin No.	Control Pin	Port	Interrupt Pin	Timer Pin	UART / SBI Pin	Intelligent I/O Pin	LIN / CAN Module Pin	Analog Pin
1		P9_3		TB3IN				
2	VDC0							
3		P9_1						
4	VDC1							
5	NSD							
6	CNVSS							
7	XCIN	P8_7						
8	XCOU	P8_6						
9	RESET							
10	XOUT							
11	VSS							
12	XIN							
13	VCC							
14		P8_5	NMI					
15		P8_4	INT2					
16		P8_3	INT1				CAN0IN/CAN0WU/ CAN1IN/CAN1WU	
17		P8_2	INT0				CAN0OUT/ CAN1OUT	
18		P7_6			CTS2/RTS2/SS2		CAN0OUT	
19		P7_5			CLK2		LIN0IN/CAN0IN/ CAN0WU	
20		P7_4			RXD2/SCL2/STXD2		LIN0OUT	
21		P7_0			TXD2/SDA2/SRXD2			
22		P6_7		TA3IN				
23		P6_5		TA0IN/ TB5IN				
24		P5_7			TXD0/SDA0/SRXD0	IIO0_7		
25		P5_6			RXD0/SCL0/STXD0	IIO0_6		
26		P5_5			CLK0	IIO0_5		
27		P5_4			CTS0/RTS0/SS0	IIO0_4		
28	CLKOUT	P5_3				IIO0_3		
29		P5_2				IIO0_2		
30		P5_1				IIO0_1		
31		P5_0				IIO0_0		
32		P4_7			TXD1/SDA1/SRXD1			
33		P4_6			RXD1/SCL1/STXD1			
34		P4_5			CLK1		LIN0IN	
35		P4_4			CTS1/RTS1/SS1		LIN0OUT	
36		P4_3			TXD3/SS00			
37		P4_2			RXD3/SSI0			
38		P4_1			CLK3/SSCK0			
39		P4_0			CTS3/RTS3/SCS0			

**Table 1.5 Pin Characteristics (2/2)**

Pin No.	Control Pin	Port	Interrupt Pin	Timer Pin	UART / SBI Pin	Intelligent I/O Pin	LIN / CAN Module Pin	Analog Pin
40		P3_7		TA4IN/ TB1IN/ $\bar{U}$	CTS4/RTS4			
41		P3_6		TA4OUT/U	TXD4			
42		P3_5		TA2IN/ $\bar{W}$	RXD4			
43		P3_4		TA2OUT/W	CLK4			
44		P3_3		TA1IN/ $\bar{V}$				
45		P3_2		TA1OUT/V				
46		P3_1		TA3OUT		UD0B	LIN0IN	
47	VCC							
48		P3_0		TA0OUT		UD0A	LIN0OUT	
49	VSS							
50		P2_7						AN2_7
51		P2_6						AN2_6
52		P2_5						AN2_5
53		P2_4						AN2_4
54		P2_3						AN2_3
55		P2_2						AN2_2
56		P2_1						AN2_1
57		P2_0						AN2_0
58		P1_7	$\bar{\text{INT}}5$			IIO0_7		
59		P1_6	$\bar{\text{INT}}4$			IIO0_6		
60		P1_5	$\bar{\text{INT}}3$			IIO0_5		
61		P1_4		TB0IN		IIO0_4		AN_4
62		P1_3			$\overline{\text{SCS}}0$	IIO0_3		AN_3
63		P1_2			SSI0	IIO0_2		AN_2
64		P1_1			SSCK0	IIO0_1		AN_1
65		P1_0			SSO0	IIO0_0		AN_0
66		P0_7						AN0_7
67		P0_6						AN0_6
68		P0_5						AN0_5
69		P0_4						AN0_4
70		P0_3						AN0_3
71		P0_2						AN0_2
72		P0_1						AN0_1
73		P0_0						AN0_0
74		P9_7		TB2IN	RXD4			$\overline{\text{ADTRG}}$
75	AVSS							
76		P9_6			TXD4		CAN1OUT	ANEX1
77	VREF							
78	AVCC/ VCC							
79		P9_5			CLK4		CAN1IN/CAN1WU	ANEX0
80		P9_4		TB4IN	CTS4/RTS4			

## 1.5 Pin Definitions and Functions

Tables 1.6 and 1.7 shows the pin definitions and functions.

**Table 1.6 Pin Definitions and Functions (1/2)**

Function	Symbol	I/O	Description
Power supply	VCC, VSS	I	Applicable as follows: VCC = 3.0 to 5.5 V, VSS = 0 V
Connecting pins for decoupling capacitor	VDC0, VDC1	—	A decoupling capacitor for internal voltage should be connected between VDC0 and VDC1
Analog power supply	AVCC, AVSS	I	Power supply for the A/D converter. AVSS should be connected to VSS
Reset input	$\overline{\text{RESET}}$	I	The MCU is reset when this pin is driven low
CNVSS	CNVSS	I	This pin should be connected to VSS via a resistor
Debug port	NSD	I/O	This pin is to communicate with a debugger. It should be connected to VCC via a resistor of 1 to 4.7 k $\Omega$
Main clock input	XIN	I	Input/output for the main clock oscillator. A crystal, or a ceramic resonator should be connected between pins XIN and XOUT. An external clock should be input at the XIN while leaving the XOUT open
Main clock output	XOUT	O	
Sub clock input	XCIN	I	Input/output for the sub clock oscillator. A crystal oscillator should be connected between pins XCIN and XCOU. An external clock should be input at the XCIN while leaving the XCOU open
Sub clock output	XCOU	O	
Clock output	CLKOUT	O	Output of the clock with the same frequency as low speed clocks, f8, or f32
External interrupt input	INT0 to INT5	I	Input for external interrupts
NMI input	P8_5/ $\overline{\text{NMI}}$	I	Input for NMI
I/O ports	P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_7, P5_0 to P5_7, P6_5, P6_7, P7_0, P7_4 to P7_6, P8_2 to P8_4, P8_6, P8_7, P9_3 to P9_7	I/O	I/O ports in CMOS. Each port can be programmed to input or output under the control of the direction register. Pull-up resistors are selected for the following 4-pin units, but are enabled only for the input pins: Pi_0 to Pi_3 and Pi_4 to Pi_7 (i = 0 to 9)
Input port	P9_1	I	Input port in CMOS. Pull-up resistors are selectable for P9_1 and P9_3

**Table 1.7 Pin Definitions and Functions (2/2)**

Function	Symbol	I/O	Description
Timer A	TA0OUT to TA4OUT	I/O	Timers A0 to A4 input/output
	TA0IN to TA4IN	I	Timers A0 to A4 input
Timer B	TB0IN to TB5IN	I	Timers B0 to B5 input
Three-phase motor control timer output	U, $\bar{U}$ , V, $\bar{V}$ , W, $\bar{W}$	O	Three-phase motor control timer output
Serial interface	CTS0 to CTS4	I	Handshake input
	RTS0 to RTS4	O	Handshake output
	CLK0 to CLK4	I/O	Transmit/receive clock input/output
	RXD0 to RXD4	I	Serial data input
	TXD0 to TXD4	O	Serial data output
I <sup>2</sup> C-bus (simplified)	SDA0 to SDA2	I/O	Serial data input/output
	SCL0 to SCL2	I/O	Transmit/receive clock input/output
Serial interface special functions	STXD0 to STXD2	O	Serial data output in slave mode
	SRXD0 to SRXD2	I	Serial data input in slave mode
	$\overline{SS0}$ to $\overline{SS2}$	I	Input to control serial interface special functions
A/D converter	AN_0 to AN_4, AN0_0 to AN0_7, AN2_0 to AN2_7	I	Analog input for the A/D converter
	ADTRG	I	External trigger input for the A/D converter
	ANEX0	I/O	Expanded analog input for the A/D converter and output in external op-amp connection mode
	ANEX1	I	Expanded analog input for the A/D converter
Reference voltage input	VREF	I	Reference voltage input for the A/D converter and D/A converter
Intelligent I/O	IIO0_0 to IIO0_7	I/O	Input/output for the Intelligent I/O group 0. Either input capture or output compare is selectable
	UD0A, UD0B	I	Input for the two-phase encoder
Serial bus interface	SSO0	I/O	Serial data output. Functions as serial data input/output in 4-wire serial bus mode
	SSI0	I/O	Serial data input. Functions as serial data input/output in 4-wire serial bus mode
	SCK0	I/O	Transmit/receive clock input/output
	$\overline{SCS0}$	I/O	Input/output to control the synchronous serial interface
LIN module	LIN0OUT	O	Transmit data output for the LIN communications
	LIN0IN	I	Receive data input for the LIN communications
CAN module	CAN0IN, CAN1IN	I	Receive data input for the CAN communications
	CAN0OUT, CAN1OUT	O	Transmit data output for the CAN communications
	CAN0WU, CAN1WU	I	Input for the CAN wake-up interrupt

## 2. Central Processing Unit (CPU)

The CPU contains the registers shown below. There are two register banks each consisting of registers R2R0, R3R1, R6R4, R7R5, A0 to A3, SB, and FB.

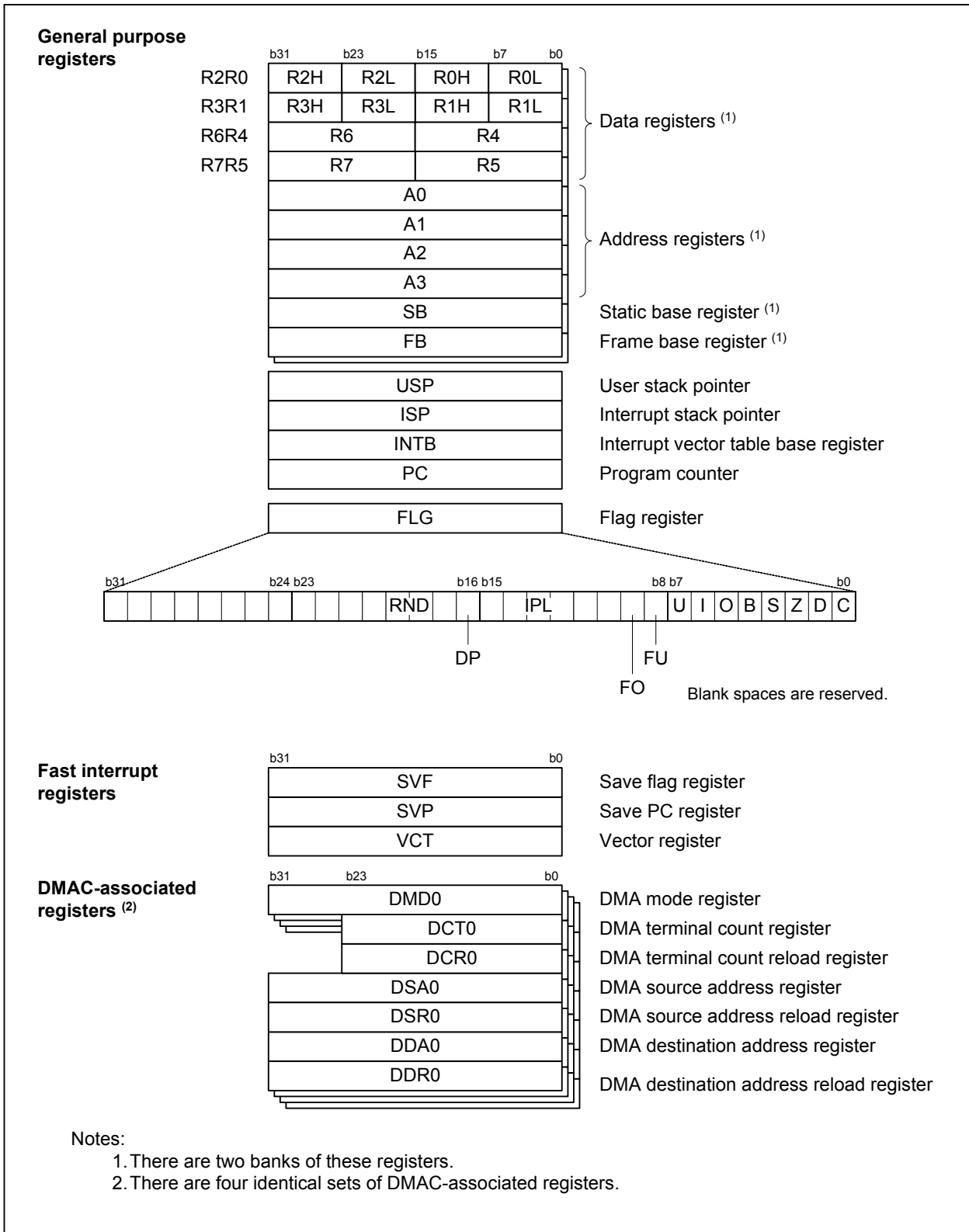


Figure 2.1 CPU Registers

## 2.1 General Purpose Registers

### 2.1.1 Data Registers (R2R0, R3R1, R6R4, and R7R5)

These 32-bit registers are primarily used for transfers and arithmetic/logic operations.

Each of the registers can be divided into upper and lower 16-bit registers, e.g. R2R0 can be divided into R2 and R0, R3R1 can be divided into R3 and R1, etc.

Moreover, data registers R2R0 and R3R1 can be divided into four 8-bit data registers: upper (R2H and R3H), mid-upper (R2L and R3L), mid-lower (R0H and R1H), and lower (R0L and R1L).

### 2.1.2 Address Registers (A0, A1, A2, and A3)

These 32-bit registers have functions similar to data registers. They are also used for address register indirect addressing and address register relative addressing.

### 2.1.3 Static Base Register (SB)

This 32-bit register is used for SB relative addressing.

### 2.1.4 Frame Base Register (FB)

This 32-bit register is used for FB relative addressing.

### 2.1.5 Program Counter (PC)

This 32-bit counter indicates the address of the instruction to be executed next.

### 2.1.6 Interrupt Vector Table Base Register (INTB)

This 32-bit register indicates the start address of a relocatable vector table.

### 2.1.7 User Stack Pointer (USP) and Interrupt Stack Pointer (ISP)

Two types of 32-bit stack pointers (SPs) are provided: user stack pointer (USP) and interrupt stack pointer (ISP).

Use the stack pointer select flag (U flag) to select either the user stack pointer (USP) or the interrupt stack pointer (ISP). The U flag is bit 7 in the flag register (FLG). Refer to 2.1.8 "Flag Register (FLG)" for details.

To minimize the overhead of interrupt sequence due to less memory access, set the user stack pointer (USP) or the interrupt stack pointer (ISP) to a multiple of 4.

### 2.1.8 Flag Register (FLG)

This 32-bit register indicates the CPU status.

#### 2.1.8.1 Carry Flag (C flag)

This flag retains a carry, borrow, or shifted-out bit generated by the arithmetic logic unit (ALU).

#### 2.1.8.2 Debug Flag (D flag)

This flag is only for debugging. Only set this bit to 0.

#### 2.1.8.3 Zero Flag (Z flag)

This flag becomes 1 when the result of an operation is 0; otherwise it is 0.

#### 2.1.8.4 Sign Flag (S flag)

This flag becomes 1 when the result of an operation is a negative value; otherwise it is 0.

### 2.1.8.5 Register Bank Select Flag (B flag)

This flag selects a register bank. It indicates 0 when register bank 0 is selected, and 1 when register bank 1 is selected.

### 2.1.8.6 Overflow Flag (O flag)

This flag becomes 1 when the result of an operation overflows; otherwise it is 0.

### 2.1.8.7 Interrupt Enable Flag (I flag)

This flag enables maskable interrupts. To disable maskable interrupts, set this flag to 0. To enable them, set this flag to 1. When an interrupt is accepted, the flag becomes 0.

### 2.1.8.8 Stack Pointer Select Flag (U flag)

To select the interrupt stack pointer (ISP), set this flag to 0. To select the user stack pointer (USP), set this flag to 1.

It becomes 0 when a hardware interrupt is accepted or when an INT instruction designated by a software interrupt number from 0 to 127 is executed.

### 2.1.8.9 Floating-point Underflow Flag (FU flag)

This flag becomes 1 when an underflow occurs in a floating-point operation; otherwise it is 0. It also becomes 1 when the operand contains invalid numbers (subnormal numbers).

### 2.1.8.10 Floating-point Overflow Flag (FO flag)

This flag becomes 1 when an overflow occurs in a floating-point operation; otherwise it is 0. It also becomes 1 when the operand contains invalid numbers (subnormal numbers).

### 2.1.8.11 Processor Interrupt Priority Level (IPL)

The processor interrupt priority level (IPL), consisting of 3 bits, selects a processor interrupt priority level from level 0 to 7. An interrupt is enabled when the interrupt request level is higher than the selected IPL.

When the processor interrupt priority level (IPL) is set to 111b (level 7), all interrupts are disabled.

### 2.1.8.12 Fixed-point Radix Point Designation Bit (DP bit)

This bit designates the radix point. It also specifies which portion of the fixed-point multiplication result to extract. It is used for the MULX instruction.

### 2.1.8.13 Floating-point Rounding Mode (RND)

The 2-bit floating-point rounding mode selects a rounding mode for floating-point calculation results.

### 2.1.8.14 Reserved

Only set this bit to 0. The read value is undefined.

## 2.2 Fast Interrupt Registers

The following three registers are provided to minimize the overhead of the interrupt sequence. Refer to 10.4 "Fast Interrupt" for details.

### 2.2.1 Save Flag Register (SVF)

This 32-bit register is used to save the flag register when a fast interrupt occurs.

### 2.2.2 Save PC Register (SVP)

This 32-bit register is used to save the program counter when a fast interrupt occurs.

### 2.2.3 Vector Register (VCT)

This 32-bit register is used to indicate a jump address when a fast interrupt occurs.

## 2.3 DMAC-associated Registers

There are seven types of DMAC-associated registers. Refer to 12. "DMAC" for details.

### 2.3.1 DMA Mode Registers (DMD0, DMD1, DMD2, and DMD3)

These 32-bit registers are used to set DMA transfer mode, bit rate, etc.

### 2.3.2 DMA Terminal Count Registers (DCT0, DCT1, DCT2, and DCT3)

These 24-bit registers are used to set the number of DMA transfers.

### 2.3.3 DMA Terminal Count Reload Registers (DCR0, DCR1, DCR2, and DCR3)

These 24-bit registers are used to set the reloaded values for DMA terminal count registers.

### 2.3.4 DMA Source Address Registers (DSA0, DSA1, DSA2, and DSA3)

These 32-bit registers are used to set DMA source addresses.

### 2.3.5 DMA Source Address Reload Registers (DSR0, DSR1, DSR2, and DSR3)

These 32-bit registers are used to set the reloaded values for DMA source address registers.

### 2.3.6 DMA Destination Address Registers (DDA0, DDA1, DDA2, and DDA3)

These 32-bit registers are used to set DMA destination addresses.

### 2.3.7 DMA Destination Address Reload Registers (DDR0, DDR1, DDR2, and DDR3)

These 32-bit registers are used to set reloaded values for DMA destination address registers.



### 3. Memory

Figure 3.1 shows the memory map of the R32C/161 Group.

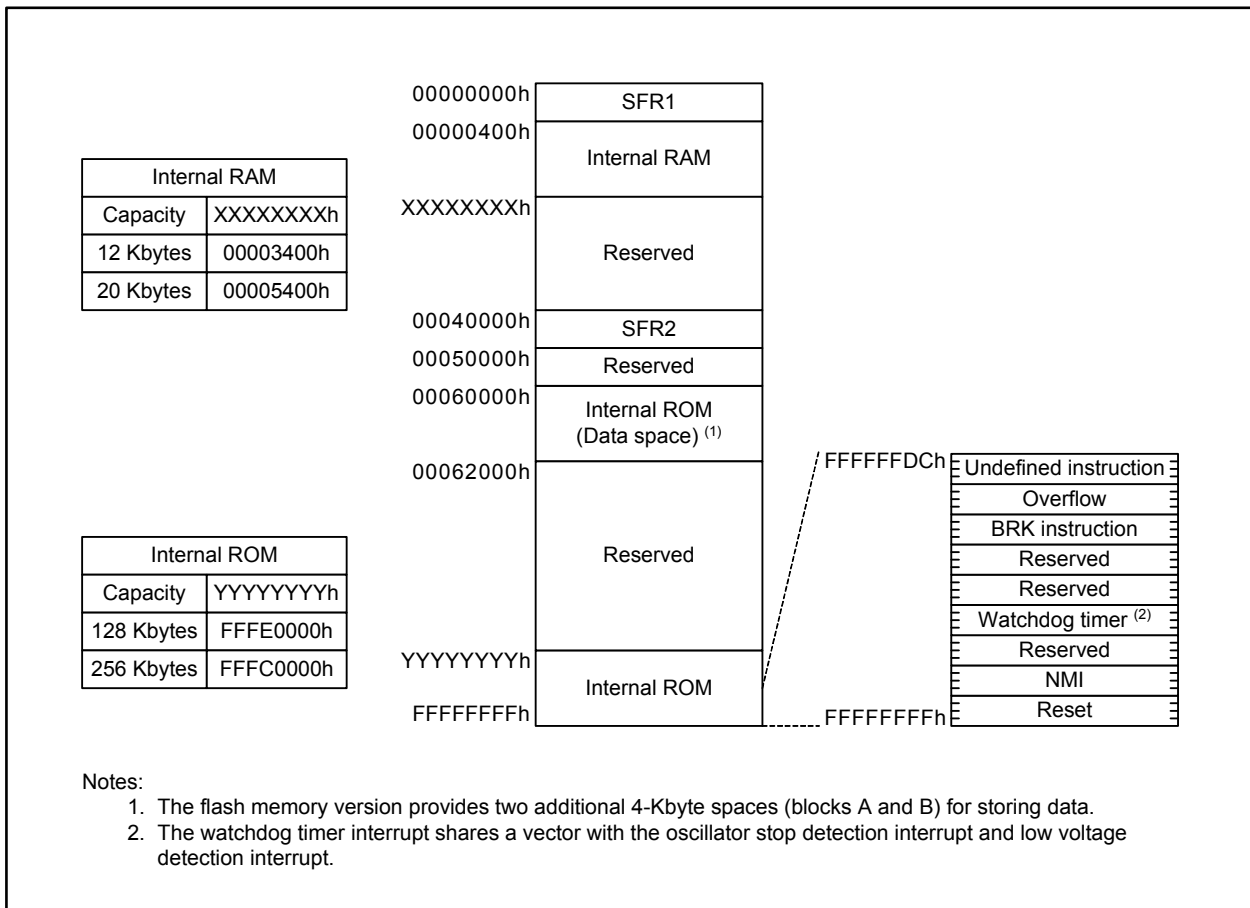
The R32C/161 Group provides a 4-Gbyte address space from 00000000h to FFFFFFFFh.

The internal ROM is mapped from address FFFFFFFFh in the inferior direction. For example, the 256-Kbyte internal ROM is mapped from FFFC0000h to FFFFFFFFh.

The fixed interrupt vector table contains the start address of interrupt handlers and is mapped from FFFFFFFDCh to FFFFFFFFh.

The internal RAM is mapped from address 00000400h in the superior direction. For example, the 20-Kbyte internal RAM is mapped from 00000400h to 000053FFh. Besides being used for data storage, the internal RAM functions as a stack(s) for subroutine calls and/or interrupt handlers.

Special function registers (SFRs), which are control registers for peripheral functions, are mapped from 00000000h to 000003FFh, and from 00040000h to 0004FFFFh. Unoccupied SFR locations are reserved, and no access is allowed.



**Figure 3.1 Memory Map**

## 4. Special Function Registers (SFRs)

SFRs are memory-mapped peripheral registers that control the operation of peripherals. Table 4.1 SFR List (1) to Table 4.53 SFR List (53) list the SFR details.

**Table 4.1 SFR List (1)**

Address	Register	Symbol	Reset Value
000000h			
000001h			
000002h			
000003h			
000004h	Clock Control Register	CCR	0001 1000b
000005h			
000006h	Flash Memory Control Register	FMCR	0000 0001b
000007h	Protect Release Register	PRR	00h
000008h			
000009h			
00000Ah			
00000Bh			
00000Ch			
00000Dh			
00000Eh			
00000Fh			
000010h			
000011h			
000012h			
000013h			
000014h			
000015h			
000016h			
000017h			
000018h			
000019h			
00001Ah			
00001Bh			
00001Ch	Flash Memory Rewrite Bus Control Register	FEBC	0000h
00001Dh			
00001Eh	Peripheral Bus Control Register	PBC	0504h
00001Fh			
000020h to 00005Fh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.2 SFR List (2)**

Address	Register	Symbol	Reset Value
000060h			
000061h	Timer B5 Interrupt Control Register	TB5IC	XXXX X000b
000062h			
000063h	UART2 Receive/ACK Interrupt Control Register	S2RIC	XXXX X000b
000064h			
000065h			
000066h			
000067h			
000068h	DMA0 Transfer Complete Interrupt Control Register	DM0IC	XXXX X000b
000069h	UART0 Start Condition/Stop Condition Detection Interrupt Control Register	BCN0IC	XXXX X000b
00006Ah	DMA2 Transfer Complete Interrupt Control Register	DM2IC	XXXX X000b
00006Bh	A/D Converter 0 Convert Completion Interrupt Control Register	AD0IC	XXXX X000b
00006Ch	Timer A0 Interrupt Control Register	TA0IC	XXXX X000b
00006Dh	Intelligent I/O Interrupt Control Register 0	IIO0IC	XXXX X000b
00006Eh	Timer A2 Interrupt Control Register	TA2IC	XXXX X000b
00006Fh	Intelligent I/O Interrupt Control Register 2	IIO2IC	XXXX X000b
000070h	Timer A4 Interrupt Control Register	TA4IC	XXXX X000b
000071h	Intelligent I/O Interrupt Control Register 4	IIO4IC	XXXX X000b
000072h	UART0 Receive/ACK Interrupt Control Register	S0RIC	XXXX X000b
000073h	Intelligent I/O Interrupt Control Register 6	IIO6IC	XXXX X000b
000074h	UART1 Receive/ACK Interrupt Control Register	S1RIC	XXXX X000b
000075h	Intelligent I/O Interrupt Control Register 8	IIO8IC	XXXX X000b
000076h	Timer B1 Interrupt Control Register	TB1IC	XXXX X000b
000077h	Intelligent I/O Interrupt Control Register 10	IIO10IC	XXXX X000b
000078h	Timer B3 Interrupt Control Register	TB3IC	XXXX X000b
000079h			
00007Ah	INT5 Interrupt Control Register	INT5IC	XX00 X000b
00007Bh	CAN0 Wake-up Interrupt Control Register	C0WIC	XXXX X000b
00007Ch	INT3 Interrupt Control Register	INT3IC	XX00 X000b
00007Dh			
00007Eh	INT1 Interrupt Control Register	INT1IC	XX00 X000b
00007Fh	LIN Low Detection Interrupt Control Register	LLDIC	XXXX X000b
000080h			
000081h	UART2 Transmit/NACK Interrupt Control Register	S2TIC	XXXX X000b
000082h			
000083h			
000084h			
000085h			
000086h			
000087h	UART2 Start Condition/Stop Condition Detection Interrupt Control Register	BCN2IC	XXXX X000b

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.3 SFR List (3)**

Address	Register	Symbol	Reset Value
000088h	DMA1 Transfer Complete Interrupt Control Register	DM1IC	XXXX X000b
000089h	UART1 Start Condition/Stop Condition Detection Interrupt Control Register	BCN1IC	XXXX X000b
00008Ah	DMA3 Transfer Complete Interrupt Control Register	DM3IC	XXXX X000b
00008Bh			
00008Ch	Timer A1 Interrupt Control Register	TA1IC	XXXX X000b
00008Dh	Intelligent I/O Interrupt Control Register 1	IIO1IC	XXXX X000b
00008Eh	Timer A3 Interrupt Control Register	TA3IC	XXXX X000b
00008Fh	Intelligent I/O Interrupt Control Register 3	IIO3IC	XXXX X000b
000090h	UART0 Transmit/NACK Interrupt Control Register	S0TIC	XXXX X000b
000091h	Intelligent I/O Interrupt Control Register 5	IIO5IC	XXXX X000b
000092h	UART1 Transmit/NACK Interrupt Control Register	S1TIC	XXXX X000b
000093h	Intelligent I/O Interrupt Control Register 7	IIO7IC	XXXX X000b
000094h	Timer B0 Interrupt Control Register	TB0IC	XXXX X000b
000095h	Intelligent I/O Interrupt Control Register 9	IIO9IC	XXXX X000b
000096h	Timer B2 Interrupt Control Register	TB2IC	XXXX X000b
000097h	Intelligent I/O Interrupt Control Register 11	IIO11IC	XXXX X000b
000098h	Timer B4 Interrupt Control Register	TB4IC	XXXX X000b
000099h			
00009Ah	INT4 Interrupt Control Register	INT4IC	XX00 X000b
00009Bh	CAN1 Wake-up Interrupt Control Register	C1WIC	XXXX X000b
00009Ch	INT2 Interrupt Control Register	INT2IC	XX00 X000b
00009Dh			
00009Eh	INT0 Interrupt Control Register	INT0IC	XX00 X000b
00009Fh			
0000A0h	Intelligent I/O Interrupt Request Register 0	IIO0IR	0000 00X1b
0000A1h	Intelligent I/O Interrupt Request Register 1	IIO1IR	0000 00X1b
0000A2h	Intelligent I/O Interrupt Request Register 2	IIO2IR	0000 0001b
0000A3h	Intelligent I/O Interrupt Request Register 3	IIO3IR	0000 00X1b
0000A4h	Intelligent I/O Interrupt Request Register 4	IIO4IR	0000 00X1b
0000A5h	Intelligent I/O Interrupt Request Register 5	IIO5IR	0000 00X1b
0000A6h	Intelligent I/O Interrupt Request Register 6	IIO6IR	0000 00X1b
0000A7h	Intelligent I/O Interrupt Request Register 7	IIO7IR	000X 00X1b
0000A8h	Intelligent I/O Interrupt Request Register 8	IIO8IR	0000 0001b
0000A9h	Intelligent I/O Interrupt Request Register 9	IIO9IR	0000 0001b
0000AAh	Intelligent I/O Interrupt Request Register 10	IIO10IR	0000 0001b
0000ABh	Intelligent I/O Interrupt Request Register 11	IIO11IR	0000 00X1b
0000ACh			
0000ADh			
0000AEh			
0000AFh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.4 SFR List (4)**

Address	Register	Symbol	Reset Value
0000B0h	Intelligent I/O Interrupt Enable Register 0	IIO0IE	00h
0000B1h	Intelligent I/O Interrupt Enable Register 1	IIO1IE	00h
0000B2h	Intelligent I/O Interrupt Enable Register 2	IIO2IE	00h
0000B3h	Intelligent I/O Interrupt Enable Register 3	IIO3IE	00h
0000B4h	Intelligent I/O Interrupt Enable Register 4	IIO4IE	00h
0000B5h	Intelligent I/O Interrupt Enable Register 5	IIO5IE	00h
0000B6h	Intelligent I/O Interrupt Enable Register 6	IIO6IE	00h
0000B7h	Intelligent I/O Interrupt Enable Register 7	IIO7IE	00h
0000B8h	Intelligent I/O Interrupt Enable Register 8	IIO8IE	00h
0000B9h	Intelligent I/O Interrupt Enable Register 9	IIO9IE	00h
0000BAh	Intelligent I/O Interrupt Enable Register 10	IIO10IE	00h
0000BBh	Intelligent I/O Interrupt Enable Register 11	IIO11IE	00h
0000BCh			
0000BDh			
0000BEh			
0000BFh			
0000C0h	Serial Bus Interface 0 Interrupt Control Register	SS0IC	XXXX X000b
0000C1h	CAN0 Transmit Interrupt Control Register	C0TIC	XXXX X000b
0000C2h			
0000C3h	CAN0 Error Interrupt Control Register	C0EIC	XXXX X000b
0000C4h			
0000C5h	CAN1 Receive Interrupt Control Register	C1RIC	XXXX X000b
0000C6h			
0000C7h			
0000C8h			
0000C9h			
0000CAh			
0000CBh			
0000CCh			
0000CDh			
0000CEh			
0000CFh			
0000D0h	CAN0 Transmit FIFO Interrupt Control Register	C0FTIC	XXXX X000b
0000D1h			
0000D2h	CAN1 Transmit FIFO Interrupt Control Register	C1FTIC	XXXX X000b
0000D3h			
0000D4h			
0000D5h	LIN0 Interrupt Control Register	L0IC	XXXX X000b
0000D6h			
0000D7h			
0000D8h	E <sup>2</sup> dataFlash Interrupt Control Register	E2FIC	XXXX X000b
0000D9h			
0000DAh			
0000DBh			
0000DCh			
0000DDh	UART3 Transmit Interrupt Control Register	S3TIC	XXXX X000b
0000DEh			
0000DFh	UART4 Transmit Interrupt Control Register	S4TIC	XXXX X000b

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.5 SFR List (5)**

Address	Register	Symbol	Reset Value
0000E0h			
0000E1h	CAN0 Receive Interrupt Control Register	C0RIC	XXXX X000b
0000E2h			
0000E3h	CAN1 Transmit Interrupt Control Register	C1TIC	XXXX X000b
0000E4h			
0000E5h	CAN1 Error Interrupt Control Register	C1EIC	XXXX X000b
0000E6h			
0000E7h			
0000E8h			
0000E9h			
0000EAh			
0000EBh			
0000ECh			
0000EDh			
0000EEh			
0000EFh			
0000F0h	CAN0 Receive FIFO Interrupt Control Register	C0FRIC	XXXX X000b
0000F1h			
0000F2h	CAN1 Receive FIFO Interrupt Control Register	C1FRIC	XXXX X000b
0000F3h			
0000F4h			
0000F5h			
0000F6h			
0000F7h			
0000F8h			
0000F9h			
000FAh			
000FBh			
000FCh			
000FDh	UART3 Receive Interrupt Control Register	S3RIC	XXXX X000b
000FEh			
000FFh	UART4 Receive Interrupt Control Register	S4RIC	XXXX X000b
000100h			
000101h			
000102h			
000103h			
000104h			
000105h			
000106h			
000107h			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.6 SFR List (6)**

Address	Register	Symbol	Reset Value
000108h to 00016Fh			
000170h			
000171h			
000172h			
000173h			
000174h			
000175h			
000176h			
000177h			
000178h			
000179h			
00017Ah			
00017Bh			
00017Ch			
00017Dh			
00017Eh			
00017Fh			
000180h	Group 0 Time Measurement/Waveform Generation Register 0	G0TM0/G0PO0	XXXXh
000181h			
000182h	Group 0 Time Measurement/Waveform Generation Register 1	G0TM1/G0PO1	XXXXh
000183h			
000184h	Group 0 Time Measurement/Waveform Generation Register 2	G0TM2/G0PO2	XXXXh
000185h			
000186h	Group 0 Time Measurement/Waveform Generation Register 3	G0TM3/G0PO3	XXXXh
000187h			
000188h	Group 0 Time Measurement/Waveform Generation Register 4	G0TM4/G0PO4	XXXXh
000189h			
00018Ah	Group 0 Time Measurement/Waveform Generation Register 5	G0TM5/G0PO5	XXXXh
00018Bh			
00018Ch	Group 0 Time Measurement/Waveform Generation Register 6	G0TM6/G0PO6	XXXXh
00018Dh			
00018Eh	Group 0 Time Measurement/Waveform Generation Register 7	G0TM7/G0PO7	XXXXh
00018Fh			
000190h	Group 0 Waveform Generation Control Register 0	G0POCR0	0000 X000b
000191h	Group 0 Waveform Generation Control Register 1	G0POCR1	0X00 X000b
000192h	Group 0 Waveform Generation Control Register 2	G0POCR2	0X00 X000b
000193h	Group 0 Waveform Generation Control Register 3	G0POCR3	0X00 X000b
000194h	Group 0 Waveform Generation Control Register 4	G0POCR4	0X00 X000b
000195h	Group 0 Waveform Generation Control Register 5	G0POCR5	0X00 X000b
000196h	Group 0 Waveform Generation Control Register 6	G0POCR6	0X00 X000b
000197h	Group 0 Waveform Generation Control Register 7	G0POCR7	0X00 X000b
000198h	Group 0 Time Measurement Control Register 0	G0TMCR0	00h
000199h	Group 0 Time Measurement Control Register 1	G0TMCR1	00h
00019Ah	Group 0 Time Measurement Control Register 2	G0TMCR2	00h
00019Bh	Group 0 Time Measurement Control Register 3	G0TMCR3	00h
00019Ch	Group 0 Time Measurement Control Register 4	G0TMCR4	00h
00019Dh	Group 0 Time Measurement Control Register 5	G0TMCR5	00h
00019Eh	Group 0 Time Measurement Control Register 6	G0TMCR6	00h
00019Fh	Group 0 Time Measurement Control Register 7	G0TMCR7	00h

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.7 SFR List (7)**

Address	Register	Symbol	Reset Value
0001A0h	Group 0 Base Timer Register	G0BT	XXXXh
0001A1h			
0001A2h	Group 0 Base Timer Control Register 0	G0BCR0	0000 0000b
0001A3h	Group 0 Base Timer Control Register 1	G0BCR1	0000 0000b
0001A4h	Group 0 Time Measurement Prescaler Register 6	G0TPR6	00h
0001A5h	Group 0 Time Measurement Prescaler Register 7	G0TPR7	00h
0001A6h	Group 0 Function Enable Register	G0FE	00h
0001A7h	Group 0 Function Select Register	G0FS	00h
0001A8h			
0001A9h			
0001AAh			
0001ABh			
0001ACh			
0001ADh			
0001AEh			
0001AFh			
0001B0h			
0001B1h			
0001B2h			
0001B3h			
0001B4h			
0001B5h			
0001B6h			
0001B7h			
0001B8h			
0001B9h			
0001BAh			
0001BBh			
0001BCh			
0001BDh			
0001BEh			
0001BFh			
0001C0h			
0001C1h			
0001C2h			
0001C3h			
0001C4h			
0001C5h			
0001C6h			
0001C7h			
0001C8h			
0001C9h			
0001CAh			
0001CBh			
0001CCh			
0001CDh			
0001CEh			
0001CFh			

X: Undefined

Blanks are reserved. No access is allowed.



**Table 4.8 SFR List (8)**

Address	Register	Symbol	Reset Value
0001D0h			
0001D1h			
0001D2h			
0001D3h			
0001D4h			
0001D5h			
0001D6h			
0001D7h			
0001D8h			
0001D9h			
0001DAh			
0001DBh			
0001DCh			
0001DDh			
0001DEh			
0001DFh			
0001E0h	UART3 Transmit/Receive Mode Register	U3MR	00h
0001E1h	UART3 Bit Rate Register	U3BRG	XXh
0001E2h	UART3 Transmit Buffer Register	U3TB	XXXXh
0001E3h			
0001E4h	UART3 Transmit/Receive Control Register 0	U3C0	00X0 1000b
0001E5h	UART3 Transmit/Receive Control Register 1	U3C1	XXXX 0010b
0001E6h	UART3 Receive Buffer Register	U3RB	XXXXh
0001E7h			
0001E8h	UART4 Transmit/Receive Mode Register	U4MR	00h
0001E9h	UART4 Bit Rate Register	U4BRG	XXh
0001EAh	UART4 Transmit Buffer Register	U4TB	XXXXh
0001EBh			
0001ECh	UART4 Transmit/Receive Control Register 0	U4C0	00X0 1000b
0001EDh	UART4 Transmit/Receive Control Register 1	U4C1	XXXX 0010b
0001EEh	UART4 Receive Buffer Register	U4RB	XXXXh
0001EFh			
0001F0h	UART3, UART4 Transmit/Receive Control Register 2	U34CON	X000 0000b
0001F1h			
0001F2h			
0001F3h			
0001F4h			
0001F5h			
0001F6h			
0001F7h			
0001F8h			
0001F9h			
0001FAh			
0001FBh			
0001FCh			
0001FDh			
0001FEh			
0001FFh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.9 SFR List (9)**

Address	Register	Symbol	Reset Value
000200h	Group0 Phase Shift Waveform Output Mode Clock Division Setting Register	G0SDR	00h
000201h	Group0 Phase Shift Waveform Output Mode Control Register	G0PSCR	00h
000202h			
000203h			
000204h			
000205h			
000206h			
000207h			
000208h	Timer B Event Clock Select Register	TBECKS	0000 0000b
000209h			
00020Ah			
00020Bh			
00020Ch			
00020Dh			
00020Eh			
00020Fh			
000210h	IIO0_7 Digital Debounce Register	IC07DDR	FFh
000211h			
000212h			
000213h			
000214h			
000215h			
000216h			
000217h			
000218h			
000219h			
00021Ah			
00021Bh			
00021Ch			
00021Dh			
00021Eh			
00021Fh			
000220h	Timer A1 Mirror Register	TA1M	XXXXh
000221h			
000222h	Timer A1-1 Mirror Register	TA11M	XXXXh
000223h			
000224h	Timer A2 Mirror Register	TA2M	XXXXh
000225h			
000226h	Timer A2-1 Mirror Register	TA21M	XXXXh
000227h			
000228h	Timer A4 Mirror Register	TA4M	XXXXh
000229h			
00022Ah	Timer A4-1 Mirror Register	TA41M	XXXXh
00022Bh			
00022Ch			
00022Dh			
00022Eh			
00022Fh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.10 SFR List (10)**

Address	Register	Symbol	Reset Value
000230h to 0002BFh			
0002C0h 0002C1h	X0 Register/Y0 Register	X0R/Y0R	XXXXh
0002C2h 0002C3h	X1 Register/Y1 Register	X1R/Y1R	XXXXh
0002C4h 0002C5h	X2 Register/Y2 Register	X2R/Y2R	XXXXh
0002C6h 0002C7h	X3 Register/Y3 Register	X3R/Y3R	XXXXh
0002C8h 0002C9h	X4 Register/Y4 Register	X4R/Y4R	XXXXh
0002CAh 0002CBh	X5 Register/Y5 Register	X5R/Y5R	XXXXh
0002CCh 0002CDh	X6 Register/Y6 Register	X6R/Y6R	XXXXh
0002CEh 0002CFh	X7 Register/Y7 Register	X7R/Y7R	XXXXh
0002D0h 0002D1h	X8 Register/Y8 Register	X8R/Y8R	XXXXh
0002D2h 0002D3h	X9 Register/Y9 Register	X9R/Y9R	XXXXh
0002D4h 0002D5h	X10 Register/Y10 Register	X10R/Y10R	XXXXh
0002D6h 0002D7h	X11 Register/Y11 Register	X11R/Y11R	XXXXh
0002D8h 0002D9h	X12 Register/Y12 Register	X12R/Y12R	XXXXh
0002DAh 0002DBh	X13 Register/Y13 Register	X13R/Y13R	XXXXh
0002DCh 0002DDh	X14 Register/Y14 Register	X14R/Y14R	XXXXh
0002DEh 0002DFh	X15 Register/Y15 Register	X15R/Y15R	XXXXh
0002E0h 0002E1h	X-Y Control Register	XYC	XXXX XX00b
0002E2h 0002E3h			
0002E4h	UART1 Special Mode Register 4	U1SMR4	00h
0002E5h	UART1 Special Mode Register 3	U1SMR3	00h
0002E6h	UART1 Special Mode Register 2	U1SMR2	00h
0002E7h	UART1 Special Mode Register	U1SMR	00h
0002E8h	UART1 Transmit/Receive Mode Register	U1MR	00h
0002E9h	UART1 Bit Rate Register	U1BRG	XXh
0002EAh 0002EBh	UART1 Transmit Buffer Register	U1TB	XXXXh
0002ECh	UART1 Transmit/Receive Control Register 0	U1C0	0000 1000b
0002EDh	UART1 Transmit/Receive Control Register 1	U1C1	0000 0010b
0002EEh 0002EFh	UART1 Receive Buffer Register	U1RB	XXXXh

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.11 SFR List (11)**

Address	Register	Symbol	Reset Value
0002F0h			
0002F1h			
0002F2h			
0002F3h			
0002F4h			
0002F5h			
0002F6h			
0002F7h			
0002F8h			
0002F9h			
0002FAh			
0002FBh			
0002FCh			
0002FDh			
0002FEh			
0002FFh			
000300h	Count Start Register for Timers B3, B4, and B5	TBSR	000X XXXXb
000301h			
000302h	Timer A1-1 Register	TA11	XXXXh
000303h			
000304h	Timer A2-1 Register	TA21	XXXXh
000305h			
000306h	Timer A4-1 Register	TA41	XXXXh
000307h			
000308h	Three-phase PWM Control Register 0	INVC0	00h
000309h	Three-phase PWM Control Register 1	INVC1	00h
00030Ah	Three-phase Output Buffer Register 0	IDB0	XX11 1111b
00030Bh	Three-phase Output Buffer Register 1	IDB1	XX11 1111b
00030Ch	Dead Time Timer	DTT	XXh
00030Dh	Timer B2 Interrupt Generating Frequency Set Counter	ICTB2	XXh
00030Eh			
00030Fh			
000310h	Timer B3 Register	TB3	XXXXh
000311h			
000312h	Timer B4 Register	TB4	XXXXh
000313h			
000314h	Timer B5 Register	TB5	XXXXh
000315h			
000316h			
000317h			
000318h			
000319h			
00031Ah			
00031Bh	Timer B3 Mode Register	TB3MR	00XX 0000b
00031Ch	Timer B4 Mode Register	TB4MR	00XX 0000b
00031Dh	Timer B5 Mode Register	TB5MR	00XX 0000b
00031Eh			
00031Fh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.12 SFR List (12)**

Address	Register	Symbol	Reset Value
000320h			
000321h			
000322h			
000323h			
000324h			
000325h			
000326h			
000327h			
000328h			
000329h			
00032Ah			
00032Bh			
00032Ch			
00032Dh			
00032Eh			
00032Fh			
000330h			
000331h			
000332h			
000333h			
000334h	UART2 Special Mode Register 4	U2SMR4	00h
000335h	UART2 Special Mode Register 3	U2SMR3	00h
000336h	UART2 Special Mode Register 2	U2SMR2	00h
000337h	UART2 Special Mode Register	U2SMR	00h
000338h	UART2 Transmit/Receive Mode Register	U2MR	00h
000339h	UART2 Bit Rate Register	U2BRG	XXh
00033Ah	UART2 Transmit Buffer Register	U2TB	XXXXh
00033Bh			
00033Ch	UART2 Transmit/Receive Control Register 0	U2C0	0000 1000b
00033Dh	UART2 Transmit/Receive Control Register 1	U2C1	0000 0010b
00033Eh	UART2 Receive Buffer Register	U2RB	XXXXh
00033Fh			
000340h	Count Start Register	TABSR	0000 0000b
000341h	Clock Prescaler Reset Register	CPSRF	0XXX XXXXb
000342h	One-shot Start Register	ONSF	0000 0000b
000343h	Trigger Select Register	TRGSR	0000 0000b
000344h	Increment/Decrement Select Register	UDF	0000 0000b
000345h			
000346h	Timer A0 Register	TA0	XXXXh
000347h			
000348h	Timer A1 Register	TA1	XXXXh
000349h			
00034Ah	Timer A2 Register	TA2	XXXXh
00034Bh			
00034Ch	Timer A3 Register	TA3	XXXXh
00034Dh			
00034Eh	Timer A4 Register	TA4	XXXXh
00034Fh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.13 SFR List (13)**

Address	Register	Symbol	Reset Value
000350h	Timer B0 Register	TB0	XXXXh
000351h			
000352h	Timer B1 Register	TB1	XXXXh
000353h			
000354h	Timer B2 Register	TB2	XXXXh
000355h			
000356h	Timer A0 Mode Register	TA0MR	0000 0000b
000357h	Timer A1 Mode Register	TA1MR	0000 0000b
000358h	Timer A2 Mode Register	TA2MR	0000 0000b
000359h	Timer A3 Mode Register	TA3MR	0000 0000b
00035Ah	Timer A4 Mode Register	TA4MR	0000 0000b
00035Bh	Timer B0 Mode Register	TB0MR	00XX 0000b
00035Ch	Timer B1 Mode Register	TB1MR	00XX 0000b
00035Dh	Timer B2 Mode Register	TB2MR	00XX 0000b
00035Eh	Timer B2 Special Mode Register	TB2SC	XXXX XXX0b
00035Fh	Count Source Prescaler Register	TCSPR	0000 0000b
000360h			
000361h			
000362h			
000363h			
000364h	UART0 Special Mode Register 4	U0SMR4	00h
000365h	UART0 Special Mode Register 3	U0SMR3	00h
000366h	UART0 Special Mode Register 2	U0SMR2	00h
000367h	UART0 Special Mode Register	U0SMR	00h
000368h	UART0 Transmit/Receive Mode Register	U0MR	00h
000369h	UART0 Bit Rate Register	U0BRG	XXh
00036Ah	UART0 Transmit Buffer Register	U0TB	XXXXh
00036Bh			
00036Ch	UART0 Transmit/Receive Control Register 0	U0C0	0000 1000b
00036Dh	UART0 Transmit/Receive Control Register 1	U0C1	0000 0010b
00036Eh	UART0 Receive Buffer Register	U0RB	XXXXh
00036Fh			
000370h			
000371h			
000372h			
000373h			
000374h			
000375h			
000376h			
000377h			
000378h			
000379h			
00037Ah			
00037Bh			
00037Ch	CRC Data Register	CRCD	XXXXh
00037Dh			
00037Eh	CRC Input Register	CRCIN	XXh
00037Fh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.14 SFR List (14)**

Address	Register	Symbol	Reset Value
000380h	A/D0 Register 0	AD00	00XXh
000381h			
000382h	A/D0 Register 1	AD01	00XXh
000383h			
000384h	A/D0 Register 2	AD02	00XXh
000385h			
000386h	A/D0 Register 3	AD03	00XXh
000387h			
000388h	A/D0 Register 4	AD04	00XXh
000389h			
00038Ah	A/D0 Register 5	AD05	00XXh
00038Bh			
00038Ch	A/D0 Register 6	AD06	00XXh
00038Dh			
00038Eh	A/D0 Register 7	AD07	00XXh
00038Fh			
000390h			
000391h			
000392h			
000393h	A/D0 Control Register 5	AD0CON5	00h
000394h	A/D0 Control Register 2	AD0CON2	X00X X000b
000395h	A/D0 Control Register 3	AD0CON3	XXXX X000b
000396h	A/D0 Control Register 0	AD0CON0	00h
000397h	A/D0 Control Register 1	AD0CON1	00h
000398h			
000399h			
00039Ah			
00039Bh			
00039Ch			
00039Dh			
00039Eh			
00039Fh			
0003A0h			
0003A1h			
0003A2h			
0003A3h			
0003A4h			
0003A5h			
0003A6h			
0003A7h			
0003A8h			
0003A9h			
0003AAh			
0003ABh			
0003ACh			
0003ADh			
0003AEh			
0003AFh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.15 SFR List (15)**

Address	Register	Symbol	Reset Value
0003B0h			
0003B1h			
0003B2h			
0003B3h			
0003B4h			
0003B5h			
0003B6h			
0003B7h			
0003B8h			
0003B9h			
0003BAh			
0003BBh			
0003BCh			
0003BDh			
0003BEh			
0003BFh			
0003C0h	Port P0 Register	P0	XXh
0003C1h	Port P1 Register	P1	XXh
0003C2h	Port P0 Direction Register	PD0	0000 0000b
0003C3h	Port P1 Direction Register	PD1	0000 0000b
0003C4h	Port P2 Register	P2	XXh
0003C5h	Port P3 Register	P3	XXh
0003C6h	Port P2 Direction Register	PD2	0000 0000b
0003C7h	Port P3 Direction Register	PD3	0000 0000b
0003C8h	Port P4 Register	P4	XXh
0003C9h	Port P5 Register	P5	XXh
0003CAh	Port P4 Direction Register	PD4	0000 0000b
0003CBh	Port P5 Direction Register	PD5	0000 0000b
0003CCh	Port P6 Register	P6	XXh
0003CDh	Port P7 Register	P7	XXh
0003CEh	Port P6 Direction Register	PD6	0000 0000b
0003CFh	Port P7 Direction Register	PD7	0000 0000b
0003D0h	Port P8 Register	P8	XXh
0003D1h	Port P9 Register	P9	XXh
0003D2h	Port P8 Direction Register	PD8	00X0 0000b
0003D3h	Port P9 Direction Register	PD9	0000 0000b
0003D4h			
0003D5h			
0003D6h			
0003D7h			
0003D8h			
0003D9h			
0003DAh			
0003DBh			
0003DCh			
0003DDh			
0003DEh			
0003DFh			

X: Undefined

Blanks are reserved. No access is allowed.



**Table 4.16 SFR List (16)**

Address	Register	Symbol	Reset Value
0003E0h			
0003E1h			
0003E2h			
0003E3h			
0003E4h			
0003E5h			
0003E6h			
0003E7h			
0003E8h			
0003E9h			
0003EAh			
0003EBh			
0003ECh			
0003EDh			
0003EEh			
0003EFh			
0003F0h	Pull-up Control Register 0	PUR0	0000 0000b
0003F1h	Pull-up Control Register 1	PUR1	XXXX 0000b
0003F2h	Pull-up Control Register 2	PUR2	0000 0000b
0003F3h			
0003F4h			
0003F5h			
0003F6h			
0003F7h			
0003F8h			
0003F9h			
0003FAh			
0003FBh			
0003FCh			
0003FDh			
0003FEh			
0003FFh	Port Control Register	PCR	XXXX XXX0b

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.17 SFR List (17)**

Address	Register	Symbol	Reset Value
040000h	Flash Memory Control Register 0	FMR0	0X01 XX00b
040001h	Flash Memory Status Register 0	FMSR0	1000 0000b
040002h			
040003h			
040004h			
040005h			
040006h			
040007h			
040008h	Flash Register Protection Unlock Register 0	FPR0	00h
040009h	Flash Memory Control Register 1	FMR1	0000 0010b
04000Ah	Block Protect Bit Monitor Register 0	FBPM0	X?X? ???b (1)
04000Bh	Block Protect Bit Monitor Register 1	FBPM1	XXX? ?XXXb (1)
04000Ch			
04000Dh			
04000Eh			
04000Fh			
040010h			
040011h			
040012h			
040013h			
040014h			
040015h			
040016h			
040017h			
040018h			
040019h			
04001Ah			
04001Bh			
04001Ch			
04001Dh			
04001Eh			
04001Fh			
040020h	PLL Control Register 0	PLC0	0000 0001b
040021h	PLL Control Register 1	PLC1	0001 1111b
040022h			
040023h			
040024h	PLL Status Register	PLS	1XXX XX00b
040025h			
040026h			
040027h			
040028h			
040029h			
04002Ah			
04002Bh			
04002Ch			
04002Dh			
04002Eh			
04002Fh			

X: Undefined

Blanks are reserved. No access is allowed.

Note:

1. The reset value reflects the value of the protect bit for each block in the flash memory.

**Table 4.18 SFR List (18)**

Address	Register	Symbol	Reset Value
040030h to 04003Fh			
040040h			
040041h			
040042h			
040043h			
040044h	Processor Mode Register 0	PM0	1000 0000b
040045h			
040046h	System Clock Control Register 0	CM0	0000 1000b
040047h	System Clock Control Register 1	CM1	0010 0000b
040048h	Processor Mode Register 3	PM3	00h
040049h			
04004Ah	Protect Register	PRCR	XXXX X000b
04004Bh			
04004Ch	Protect Register 3	PRCR3	0000 0000b
04004Dh	Oscillator Stop Detection Register	CM2	00h
04004Eh			
04004Fh			
040050h			
040051h			
040052h			
040053h	Processor Mode Register 2	PM2	00h
040054h			
040055h			
040056h			
040057h			
040058h			
040059h			
04005Ah	Low Speed Mode Clock Control Register	CM3	XXXX XX00b
04005Bh			
04005Ch			
04005Dh			
04005Eh			
04005Fh			
040060h	Voltage Regulator Control Register	VRCR	0000 0000b
040061h			
040062h	Low Voltage Detector Control Register	LVDC	0000 XX00b
040063h			
040064h	Detection Voltage Configuration Register	DVCR	0000 XXXXb
040065h			
040066h			
040067h			
040068h to 040093h			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.19 SFR List (19)**

Address	Register	Symbol	Reset Value
040094h			
040095h			
040096h			
040097h	Three-phase Output Buffer Control Register	IOBC	0XXX XX0Xb
040098h	Input Function Select Register 0	IFS0	X0X0 X000b
040099h	Input Function Select Register 1	IFS1	XXXX X0X0b
04009Ah	Input Function Select Register 2	IFS2	0000 0000b
04009Bh			
04009Ch			
04009Dh	Input Function Select Register 5	IFS5	XXX0 X0X0b
04009Eh	Input Function Select Register 6	IFS6	XXXX 0000b
04009Fh			
0400A0h	Port P0_0 Function Select Register	P0_0S	0XXX X000b
0400A1h	Port P1_0 Function Select Register	P1_0S	0XXX X000b
0400A2h	Port P0_1 Function Select Register	P0_1S	0XXX X000b
0400A3h	Port P1_1 Function Select Register	P1_1S	0XXX X000b
0400A4h	Port P0_2 Function Select Register	P0_2S	0XXX X000b
0400A5h	Port P1_2 Function Select Register	P1_2S	0XXX X000b
0400A6h	Port P0_3 Function Select Register	P0_3S	0XXX X000b
0400A7h	Port P1_3 Function Select Register	P1_3S	0XXX X000b
0400A8h	Port P0_4 Function Select Register	P0_4S	0XXX X000b
0400A9h	Port P1_4 Function Select Register	P1_4S	0XXX X000b
0400AAh	Port P0_5 Function Select Register	P0_5S	0XXX X000b
0400ABh	Port P1_5 Function Select Register	P1_5S	XXXX X000b
0400ACh	Port P0_6 Function Select Register	P0_6S	0XXX X000b
0400ADh	Port P1_6 Function Select Register	P1_6S	XXXX X000b
0400AEh	Port P0_7 Function Select Register	P0_7S	0XXX X000b
0400AFh	Port P1_7 Function Select Register	P1_7S	XXXX X000b
0400B0h	Port P2_0 Function Select Register	P2_0S	0XXX X000b
0400B1h	Port P3_0 Function Select Register	P3_0S	XXXX X000b
0400B2h	Port P2_1 Function Select Register	P2_1S	0XXX X000b
0400B3h	Port P3_1 Function Select Register	P3_1S	XXXX X000b
0400B4h	Port P2_2 Function Select Register	P2_2S	0XXX X000b
0400B5h	Port P3_2 Function Select Register	P3_2S	XXXX X000b
0400B6h	Port P2_3 Function Select Register	P2_3S	0XXX X000b
0400B7h	Port P3_3 Function Select Register	P3_3S	XXXX X000b
0400B8h	Port P2_4 Function Select Register	P2_4S	0XXX X000b
0400B9h	Port P3_4 Function Select Register	P3_4S	XXXX X000b
0400BAh	Port P2_5 Function Select Register	P2_5S	0XXX X000b
0400BBh	Port P3_5 Function Select Register	P3_5S	XXXX X000b
0400BCh	Port P2_6 Function Select Register	P2_6S	0XXX X000b
0400BDh	Port P3_6 Function Select Register	P3_6S	XXXX X000b
0400BEh	Port P2_7 Function Select Register	P2_7S	0XXX X000b
0400BFh	Port P3_7 Function Select Register	P3_7S	XXXX X000b

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.20 SFR List (20)**

Address	Register	Symbol	Reset Value
0400C0h	Port P4_0 Function Select Register	P4_0S	XXXX X000b
0400C1h	Port P5_0 Function Select Register	P5_0S	XXXX X000b
0400C2h	Port P4_1 Function Select Register	P4_1S	XXXX X000b
0400C3h	Port P5_1 Function Select Register	P5_1S	XXXX X000b
0400C4h	Port P4_2 Function Select Register	P4_2S	XXXX X000b
0400C5h	Port P5_2 Function Select Register	P5_2S	XXXX X000b
0400C6h	Port P4_3 Function Select Register	P4_3S	XXXX X000b
0400C7h	Port P5_3 Function Select Register	P5_3S	XXXX X000b
0400C8h	Port P4_4 Function Select Register	P4_4S	XXXX X000b
0400C9h	Port P5_4 Function Select Register	P5_4S	XXXX X000b
0400CAh	Port P4_5 Function Select Register	P4_5S	XXXX X000b
0400CBh	Port P5_5 Function Select Register	P5_5S	XXXX X000b
0400CCh	Port P4_6 Function Select Register	P4_6S	XXXX X000b
0400CDh	Port P5_6 Function Select Register	P5_6S	XXXX X000b
0400CEh	Port P4_7 Function Select Register	P4_7S	XXXX X000b
0400CFh	Port P5_7 Function Select Register	P5_7S	XXXX X000b
0400D0h			
0400D1h	Port P7_0 Function Select Register	P7_0S	XXXX X000b
0400D2h			
0400D3h			
0400D4h			
0400D5h			
0400D6h			
0400D7h			
0400D8h			
0400D9h	Port P7_4 Function Select Register	P7_4S	XXXX X000b
0400DAh	Port P6_5 Function Select Register	P6_5S	XXXX X000b
0400DBh	Port P7_5 Function Select Register	P7_5S	XXXX X000b
0400DCh			
0400DDh	Port P7_6 Function Select Register	P7_6S	XXXX X000b
0400DEh	Port P6_7 Function Select Register	P6_7S	XXXX X000b
0400DFh			
0400E0h			
0400E1h			
0400E2h			
0400E3h			
0400E4h	Port P8_2 Function Select Register	P8_2S	XXXX X000b
0400E5h			
0400E6h	Port P8_3 Function Select Register	P8_3S	XXXX X000b
0400E7h	Port P9_3 Function Select Register	P9_3S	0XXX X000b
0400E8h	Port P8_4 Function Select Register	P8_4S	XXXX X000b
0400E9h	Port P9_4 Function Select Register	P9_4S	0XXX X000b
0400EAh			
0400EBh	Port P9_5 Function Select Register	P9_5S	0XXX X000b
0400ECh	Port P8_6 Function Select Register	P8_6S	XXXX X000b
0400EDh	Port P9_6 Function Select Register	P9_6S	0XXX X000b
0400EEh	Port P8_7 Function Select Register	P8_7S	XXXX X000b
0400EFh	Port P9_7 Function Select Register	P9_7S	XXXX X000b

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.21 SFR List (21)**

Address	Register	Symbol	Reset Value
0400F0h to 04403Fh			
044040h			
044041h			
044042h			
044043h			
044044h			
044045h			
044046h			
044047h			
044048h			
044049h			
04404Ah			
04404Bh			
04404Ch	Protect Register 4	PRCR4	0000 0000b
04404Dh	Watchdog Timer Clock Control Register	WDK	0000 0000b
04404Eh	Watchdog Timer Start Register	WDTS	XXXX XXXXb
04404Fh	Watchdog Timer Control Register	WDC	000X XXXXb
044050h			
044051h			
044052h			
044053h			
044054h			
044055h			
044056h			
044057h			
044058h			
044059h			
04405Ah			
04405Bh			
04405Ch			
04405Dh			
04405Eh			
04405Fh	Protect Register 2	PRCR2	0XXX XXXXb

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.22 SFR List (22)**

Address	Register	Symbol	Reset Value
044060h			
044061h			
044062h			
044063h			
044064h			
044065h			
044066h			
044067h			
044068h			
044069h			
04406Ah			
04406Bh			
04406Ch			
04406Dh			
04406Eh			
04406Fh	External Interrupt Request Source Select Register 0	IFSR0	0000 0000b
044070h	DMA0 Request Source Select Register 2	DM0SL2	XX00 0000b
044071h	DMA1 Request Source Select Register 2	DM1SL2	XX00 0000b
044072h	DMA2 Request Source Select Register 2	DM2SL2	XX00 0000b
044073h	DMA3 Request Source Select Register 2	DM3SL2	XX00 0000b
044074h			
044075h			
044076h			
044077h			
044078h	DMA0 Request Source Select Register	DM0SL	XXX0 0000b
044079h	DMA1 Request Source Select Register	DM1SL	XXX0 0000b
04407Ah	DMA2 Request Source Select Register	DM2SL	XXX0 0000b
04407Bh	DMA3 Request Source Select Register	DM3SL	XXX0 0000b
04407Ch			
04407Dh	Wake-up IPL Setting Register 2	RIPL2	XX0X 0000b
04407Eh			
04407Fh	Wake-up IPL Setting Register 1	RIPL1	XX0X 0000b
044080h	External Interrupt Input Filter Select Register 0	INTF0	0000 0000b
044081h			
044082h	External Interrupt Input Filter Select Register 1	INTF1	0000 0000b
044083h			
044084h			
044085h			
044086h			
044087h			
044088h			
044089h			
04408Ah			
04408Bh			
04408Ch			
04408Dh			
04408Eh			
04408Fh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.23 SFR List (23)**

Address	Register	Symbol	Reset Value
044090h to 044DFFh			
044E00h	LIN Channel Window Select/Input Signal Low Detection Status Register	LCW	0000 0000b
044E01h	LIN Baud Rate Generator Control Register	LBRG	0000 0000b
044E02h	LIN Baud Rate Prescaler 0	LBRP0	00h
044E03h	LIN Baud Rate Prescaler 1	LBRP1	00h
044E04h	LIN Mode Register 0	LMD0	0000 0000b
044E05h	LIN Mode Register 1	LMD1	00h
044E06h	LIN Wake-up Setting Register	LWUP	00h
044E07h			
044E08h	LIN Break Field Setting Register	LBRK	0000 0000b
044E09h	LIN Space Setting Register	LSPC	0000 0000b
044E0Ah	LIN Response Field Setting Register	LRFC	0000 0000b
044E0Bh	LIN ID Buffer Register	LIDB	00h
044E0Ch	LIN Status Control Register	LSC	0000 0000b
044E0Dh	LIN Transmission Control Register	LTC	0000 0000b
044E0Eh	LIN Status Register	LST	0000 0000b
044E0Fh	LIN Error Status Register	LEST	0000 0000b
044E10h	LIN Data 1 Buffer Register	LDB1	00h
044E11h	LIN Data 2 Buffer Register	LDB2	00h
044E12h	LIN Data 3 Buffer Register	LDB3	00h
044E13h	LIN Data 4 Buffer Register	LDB4	00h
044E14h	LIN Data 5 Buffer Register	LDB5	00h
044E15h	LIN Data 6 Buffer Register	LDB6	00h
044E16h	LIN Data 7 Buffer Register	LDB7	00h
044E17h	LIN Data 8 Buffer Register	LDB8	00h
044E18h			
044E19h			
044E1Ah			
044E1Bh			
044E1Ch			
044E1Dh			
044E1Eh			
044E1Fh			

X: Undefined

Blanks are reserved. No access is allowed.



**Table 4.24 SFR List (24)**

Address	Register	Symbol	Reset Value
044E20h to 044EFFh			
044F00h			
044F01h			
044F02h			
044F03h			
044F04h			
044F05h			
044F06h	SS0 Receive Data Register	SS0RDR	FFh
044F07h	SS0 Receive Data Register (H)	SS0RDR (H)	FFh
044F08h	SS0 Control Register H	SS0CRH	00h
044F09h	SS0 Control Register L	SS0CRL	0111 1101b
044F0Ah	SS0 Mode Register	SS0MR	0001 0000b
044F0Bh	SS0 Enable Register	SS0ER	00h
044F0Ch	SS0 Status Register	SS0SR	00h
044F0Dh	SS0 Mode Register 2	SS0MR2	00h
044F0Eh	SS0 Transmit Data Register	SS0TDR	FFh
044F0Fh	SS0 Transmit Data Register (H)	SS0TDR (H)	FFh
044F10h			
044F11h			
044F12h			
044F13h			
044F14h			
044F15h			
044F16h			
044F17h			
044F18h			
044F19h			
044F1Ah			
044F1Bh			
044F1Ch			
044F1Dh			
044F1Eh			
044F1Fh			
044F20h			
044F21h			
044F22h			
044F23h			
044F24h			
044F25h			
044F26h			
044F27h			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.25 SFR List (25)**

Address	Register	Symbol	Reset Value
044F28h to 044FDFh			
044FE0h 044FE1h 044FE2h 044FE3h 044FE4h 044FE5h 044FE6h 044FE7h	E <sup>2</sup> dataFlash Address Register	E2FA	XXXX 0000h
044FE8h 044FE9h 044FEAh 044FEBh	E <sup>2</sup> dataFlash Instruction Register	E2FI	XX00h
044FEC 044FEDh 044FEEh 044FEFh	E <sup>2</sup> dataFlash Data Register	E2FD	XXXXh
044FF0h 044FF1h	E <sup>2</sup> dataFlash Mode Register	E2FM	0000 0000b
044FF2h 044FF3h	E <sup>2</sup> dataFlash Control Register	E2FC	XXXX XXX0b
044FF4h 044FF5h 044FF6h 044FF7h 044FF8h 044FF9h 044FFAh 044FFBh 044FFCh 044FFDh 044FFEh 044FFFh	E <sup>2</sup> dataFlash Status Register 1	E2FS1	XXXX XXX0b
045000h			
045001h 045002h 045003h 045004h 045005h 045006h 045007h	E <sup>2</sup> dataFlash Status Register 0	E2FS0	XXXX XXXXb
045008h to 045FFFh			
046000h to 0467FFh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.26 SFR List (26)**

Address	Register	Symbol	Reset Value				
046800h to 0477FFh							
047800h 047801h 047802h 047803h 047804h	CAN1 Mailbox 0: Message Identifier	C1MB0	XXXX XXXXh				
047805h	CAN1 Mailbox 0: Data Length						
047806h 047807h 047808h 047809h 04780Ah 04780Bh 04780Ch 04780Dh	CAN1 Mailbox 0: Data Field						
04780Eh 04780Fh	CAN1 Mailbox 0: Time Stamp						
047810h 047811h 047812h 047813h 047814h	CAN1 Mailbox 1: Message Identifier			C1MB1	XXXX XXXXh		
047815h	CAN1 Mailbox 1: Data Length						
047816h 047817h 047818h 047819h 04781Ah 04781Bh 04781Ch 04781Dh	CAN1 Mailbox 1: Data Field						
04781Eh 04781Fh	CAN1 Mailbox 1: Time Stamp						
047820h 047821h 047822h 047823h 047824h	CAN1 Mailbox 2: Message Identifier					C1MB2	XXXX XXXXh
047825h	CAN1 Mailbox 2: Data Length						
047826h 047827h 047828h 047829h 04782Ah 04782Bh 04782Ch 04782Dh	CAN1 Mailbox 2: Data Field						
04782Eh 04782Fh	CAN1 Mailbox 2: Time Stamp						

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.27 SFR List (27)**

Address	Register	Symbol	Reset Value
047830h	CAN1 Mailbox 3: Message Identifier	C1MB3	XXXX XXXXh
047831h			
047832h			
047833h			
047834h			
047835h	CAN1 Mailbox 3: Data Length		XXh
047836h	CAN1 Mailbox 3: Data Field		XXXX XXXX XXXX XXXXh
047837h			
047838h			
047839h			
04783Ah			
04783Bh			
04783Ch			
04783Dh			
04783Eh			
04783Fh			
047840h	CAN1 Mailbox 4: Message Identifier	C1MB4	XXXX XXXXh
047841h			
047842h			
047843h			
047844h			
047845h	CAN1 Mailbox 4: Data Length		XXh
047846h	CAN1 Mailbox 4: Data Field		XXXX XXXX XXXX XXXXh
047847h			
047848h			
047849h			
04784Ah			
04784Bh			
04784Ch			
04784Dh			
04784Eh			
04784Fh			
047850h	CAN1 Mailbox 5: Message Identifier	C1MB5	XXXX XXXXh
047851h			
047852h			
047853h			
047854h			
047855h	CAN1 Mailbox 5: Data Length		XXh
047856h	CAN1 Mailbox 5: Data Field		XXXX XXXX XXXX XXXXh
047857h			
047858h			
047859h			
04785Ah			
04785Bh			
04785Ch			
04785Dh			
04785Eh			
04785Fh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.28 SFR List (28)**

Address	Register	Symbol	Reset Value			
047860h	CAN1 Mailbox 6: Message Identifier	C1MB6	XXXX XXXXh			
047861h						
047862h						
047863h						
047864h						
047865h	CAN1 Mailbox 6: Data Length		XXh			
047866h	CAN1 Mailbox 6: Data Field		XXXX XXXX XXXX XXXXh			
047867h						
047868h						
047869h						
04786Ah						
04786Bh						
04786Ch						
04786Dh						
04786Eh				CAN1 Mailbox 6: Time Stamp		XXXXh
04786Fh						
047870h	CAN1 Mailbox 7: Message Identifier	C1MB7	XXXX XXXXh			
047871h						
047872h						
047873h						
047874h						
047875h	CAN1 Mailbox 7: Data Length		XXh			
047876h	CAN1 Mailbox 7: Data Field		XXXX XXXX XXXX XXXXh			
047877h						
047878h						
047879h						
04787Ah						
04787Bh						
04787Ch						
04787Dh						
04787Eh				CAN1 Mailbox 7: Time Stamp		XXXXh
04787Fh						
047880h	CAN1 Mailbox 8: Message Identifier	C1MB8	XXXX XXXXh			
047881h						
047882h						
047883h						
047884h						
047885h	CAN1 Mailbox 8: Data Length		XXh			
047886h	CAN1 Mailbox 8: Data Field		XXXX XXXX XXXX XXXXh			
047887h						
047888h						
047889h						
04788Ah						
04788Bh						
04788Ch						
04788Dh						
04788Eh				CAN1 Mailbox 8: Time Stamp		XXXXh
04788Fh						

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.29 SFR List (29)**

Address	Register	Symbol	Reset Value
047890h	CAN1 Mailbox 9: Message Identifier	C1MB9	XXXX XXXXh
047891h			
047892h			
047893h			
047894h			
047895h	CAN1 Mailbox 9: Data Length		XXh
047896h	CAN1 Mailbox 9: Data Field		XXXX XXXX XXXX XXXXh
047897h			
047898h			
047899h			
04789Ah			
04789Bh			
04789Ch			
04789Dh			
04789Eh	CAN1 Mailbox 9: Time Stamp		XXXXh
04789Fh			
0478A0h	CAN1 Mailbox 10: Message Identifier	C1MB10	XXXX XXXXh
0478A1h			
0478A2h			
0478A3h			
0478A4h			
0478A5h	CAN1 Mailbox 10: Data Length		XXh
0478A6h	CAN1 Mailbox 10: Data Field		XXXX XXXX XXXX XXXXh
0478A7h			
0478A8h			
0478A9h			
0478AAh			
0478ABh			
0478ACh			
0478ADh			
0478AEh	CAN1 Mailbox 10: Time Stamp		XXXXh
0478AFh			
0478B0h	CAN1 Mailbox 11: Message Identifier	C1MB11	XXXX XXXXh
0478B1h			
0478B2h			
0478B3h			
0478B4h			
0478B5h	CAN1 Mailbox 11: Data Length		XXh
0478B6h	CAN1 Mailbox 11: Data Field		XXXX XXXX XXXX XXXXh
0478B7h			
0478B8h			
0478B9h			
0478BAh			
0478BBh			
0478BCh			
0478BDh			
0478BEh	CAN1 Mailbox 11: Time Stamp		XXXXh
0478BFh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.30 SFR List (30)**

Address	Register	Symbol	Reset Value
0478C0h	CAN1 Mailbox 12: Message Identifier	C1MB12	XXXX XXXXh
0478C1h			
0478C2h			
0478C3h			
0478C4h			
0478C5h	CAN1 Mailbox 12: Data Length		XXh
0478C6h	CAN1 Mailbox 12: Data Field		XXXX XXXX XXXX XXXXh
0478C7h			
0478C8h			
0478C9h			
0478CAh			
0478CBh			
0478CCh			
0478CDh			
0478CEh	CAN1 Mailbox 12: Time Stamp		XXXXh
0478CFh			
0478D0h	CAN1 Mailbox 13: Message Identifier	C1MB13	XXXX XXXXh
0478D1h			
0478D2h			
0478D3h			
0478D4h			
0478D5h	CAN1 Mailbox 13: Data Length		XXh
0478D6h	CAN1 Mailbox 13: Data Field		XXXX XXXX XXXX XXXXh
0478D7h			
0478D8h			
0478D9h			
0478DAh			
0478DBh			
0478DCh			
0478DDh			
0478DEh	CAN1 Mailbox 13: Time Stamp		XXXXh
0478DFh			
0478E0h	CAN1 Mailbox 14: Message Identifier	C1MB14	XXXX XXXXh
0478E1h			
0478E2h			
0478E3h			
0478E4h			
0478E5h	CAN1 Mailbox 14: Data Length		XXh
0478E6h	CAN1 Mailbox 14: Data Field		XXXX XXXX XXXX XXXXh
0478E7h			
0478E8h			
0478E9h			
0478EAh			
0478EBh			
0478ECh			
0478EDh			
0478EEh	CAN1 Mailbox 14: Time Stamp		XXXXh
0478EFh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.31 SFR List (31)**

Address	Register	Symbol	Reset Value
0478F0h	CAN1 Mailbox 15: Message Identifier	C1MB15	XXXX XXXXh
0478F1h			
0478F2h			
0478F3h			
0478F4h			
0478F5h	CAN1 Mailbox 15: Data Length		XXh
0478F6h	CAN1 Mailbox 15: Data Field		XXXX XXXX XXXX XXXXh
0478F7h			
0478F8h			
0478F9h			
0478FAh			
0478FBh			
0478FCh			
0478FDh			
0478FEh	CAN1 Mailbox 15: Time Stamp		XXXXh
0478FFh			
047900h	CAN1 Mailbox 16: Message Identifier	C1MB16	XXXX XXXXh
047901h			
047902h			
047903h			
047904h			
047905h	CAN1 Mailbox 16: Data Length		XXh
047906h	CAN1 Mailbox 16: Data Field		XXXX XXXX XXXX XXXXh
047907h			
047908h			
047909h			
04790Ah			
04790Bh			
04790Ch			
04790Dh			
04790Eh	CAN1 Mailbox 16: Time Stamp		XXXXh
04790Fh			
047910h	CAN1 Mailbox 17: Message Identifier	C1MB17	XXXX XXXXh
047911h			
047912h			
047913h			
047914h			
047915h	CAN1 Mailbox 17: Data Length		XXh
047916h	CAN1 Mailbox 17: Data Field		XXXX XXXX XXXX XXXXh
047917h			
047918h			
047919h			
04791Ah			
04791Bh			
04791Ch			
04791Dh			
04791Eh	CAN1 Mailbox 17: Time Stamp		XXXXh
04791Fh			

X: Undefined

Blanks are reserved. No access is allowed.



**Table 4.32 SFR List (32)**

Address	Register	Symbol	Reset Value
047920h	CAN1 Mailbox 18: Message Identifier	C1MB18	XXXX XXXXh
047921h			
047922h			
047923h			
047924h			
047925h	CAN1 Mailbox 18: Data Length		XXh
047926h	CAN1 Mailbox 18: Data Field		XXXX XXXX XXXX XXXXh
047927h			
047928h			
047929h			
04792Ah			
04792Bh			
04792Ch			
04792Dh			
04792Eh			
04792Fh			
047930h	CAN1 Mailbox 19: Message Identifier	C1MB19	XXXX XXXXh
047931h			
047932h			
047933h			
047934h			
047935h	CAN1 Mailbox 19: Data Length		XXh
047936h	CAN1 Mailbox 19: Data Field		XXXX XXXX XXXX XXXXh
047937h			
047938h			
047939h			
04793Ah			
04793Bh			
04793Ch			
04793Dh			
04793Eh			
04793Fh			
047940h	CAN1 Mailbox 20: Message Identifier	C1MB20	XXXX XXXXh
047941h			
047942h			
047943h			
047944h			
047945h	CAN1 Mailbox 20: Data Length		XXh
047946h	CAN1 Mailbox 20: Data Field		XXXX XXXX XXXX XXXXh
047947h			
047948h			
047949h			
04794Ah			
04794Bh			
04794Ch			
04794Dh			
04794Eh			
04794Fh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.33 SFR List (33)**

Address	Register	Symbol	Reset Value
047950h	CAN1 Mailbox 21: Message Identifier	C1MB21	XXXX XXXXh
047951h			
047952h			
047953h			
047954h			
047955h	CAN1 Mailbox 21: Data Length		XXh
047956h	CAN1 Mailbox 21: Data Field		XXXX XXXX XXXX XXXXh
047957h			
047958h			
047959h			
04795Ah			
04795Bh			
04795Ch			
04795Dh			
04795Eh			
04795Fh			
047960h	CAN1 Mailbox 22: Identifier	C1MB22	XXXX XXXXh
047961h			
047962h			
047963h			
047964h			
047965h	CAN1 Mailbox 22: Data Length		XXh
047966h	CAN1 Mailbox 22: Data Field		XXXX XXXX XXXX XXXXh
047967h			
047968h			
047969h			
04796Ah			
04796Bh			
04796Ch			
04796Dh			
04796Eh			
04796Fh			
047970h	CAN1 Mailbox 23: Message Identifier	C1MB23	XXXX XXXXh
047971h			
047972h			
047973h			
047974h			
047975h	CAN1 Mailbox 23: Data Length		XXh
047976h	CAN1 Mailbox 23: Data Field		XXXX XXXX XXXX XXXXh
047977h			
047978h			
047979h			
04797Ah			
04797Bh			
04797Ch			
04797Dh			
04797Eh			
04797Fh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.34 SFR List (34)**

Address	Register	Symbol	Reset Value
047980h	CAN1 Mailbox 24: Message Identifier	C1MB24	XXXX XXXXh
047981h			
047982h			
047983h			
047984h			
047985h	CAN1 Mailbox 24: Data Length		XXh
047986h	CAN1 Mailbox 24: Data Field		XXXX XXXX XXXX XXXXh
047987h			
047988h			
047989h			
04798Ah			
04798Bh			
04798Ch			
04798Dh			
04798Eh	CAN1 Mailbox 24: Time Stamp		XXXXh
04798Fh			
047990h	CAN1 Mailbox 25: Message Identifier	C1MB25	XXXX XXXXh
047991h			
047992h			
047993h			
047994h			
047995h	CAN1 Mailbox 25: Data Length		XXh
047996h	CAN1 Mailbox 25: Data Field		XXXX XXXX XXXX XXXXh
047997h			
047998h			
047999h			
04799Ah			
04799Bh			
04799Ch			
04799Dh			
04799Eh	CAN1 Mailbox 25: Time Stamp		XXXXh
04799Fh			
0479A0h	CAN1 Mailbox 26: Message Identifier	C1MB26	XXXX XXXXh
0479A1h			
0479A2h			
0479A3h			
0479A4h			
0479A5h	CAN1 Mailbox 26: Data Length		XXh
0479A6h	CAN1 Mailbox 26: Data Field		XXXX XXXX XXXX XXXXh
0479A7h			
0479A8h			
0479A9h			
0479AAh			
0479ABh			
0479ACh			
0479ADh			
0479AEh	CAN1 Mailbox 26: Time Stamp		XXXXh
0479AFh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.35 SFR List (35)**

Address	Register	Symbol	Reset Value
0479B0h	CAN1 Mailbox 27: Message Identifier	C1MB27	XXXX XXXXh
0479B1h			
0479B2h			
0479B3h			
0479B4h			
0479B5h	CAN1 Mailbox 27: Data Length		XXh
0479B6h	CAN1 Mailbox 27: Data Field		XXXX XXXX XXXX XXXXh
0479B7h			
0479B8h			
0479B9h			
0479BAh			
0479BBh			
0479BCh			
0479BDh			
0479BEh	CAN1 Mailbox 27: Time Stamp		XXXXh
0479BFh			
0479C0h	CAN1 Mailbox 28: Message Identifier	C1MB28	XXXX XXXXh
0479C1h			
0479C2h			
0479C3h			
0479C4h			
0479C5h	CAN1 Mailbox 28: Data Length		XXh
0479C6h	CAN1 Mailbox 28: Data Field		XXXX XXXX XXXX XXXXh
0479C7h			
0479C8h			
0479C9h			
0479CAh			
0479CBh			
0479CCh			
0479CDh			
0479CEh	CAN1 Mailbox 28: Time Stamp		XXXXh
0479CFh			
0479D0h	CAN1 Mailbox 29: Message Identifier	C1MB29	XXXX XXXXh
0479D1h			
0479D2h			
0479D3h			
0479D4h			
0479D5h	CAN1 Mailbox 29: Data Length		XXh
0479D6h	CAN1 Mailbox 29: Data Field		XXXX XXXX XXXX XXXXh
0479D7h			
0479D8h			
0479D9h			
0479DAh			
0479DBh			
0479DCh			
0479DDh			
0479DEh	CAN1 Mailbox 29: Time Stamp		XXXXh
0479DFh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.36 SFR List (36)**

Address	Register	Symbol	Reset Value
0479E0h	CAN1 Mailbox 30: Message Identifier	C1MB30	XXXX XXXXh
0479E1h			
0479E2h			
0479E3h			
0479E4h			
0479E5h	CAN1 Mailbox 30: Data Length		XXh
0479E6h	CAN1 Mailbox 30: Data Field		XXXX XXXX XXXX XXXXh
0479E7h			
0479E8h			
0479E9h			
0479EAh			
0479EBh			
0479ECh			
0479EDh			
0479EEh	CAN1 Mailbox 30: Time Stamp		XXXXh
0479EFh			
0479F0h	CAN1 Mailbox 31: Message Identifier	C1MB31	XXXX XXXXh
0479F1h			
0479F2h			
0479F3h			
0479F4h			
0479F5h	CAN1 Mailbox 31: Data Length		XXh
0479F6h	CAN1 Mailbox 31: Data Field		XXXX XXXX XXXX XXXXh
0479F7h			
0479F8h			
0479F9h			
0479FAh			
0479FBh			
0479FCh			
0479FDh			
0479FEh	CAN1 Mailbox 31: Time Stamp		XXXXh
0479FFh			
047A00h	CAN1 Mask Register 0	C1MKR0	XXXX XXXXh
047A01h			
047A02h			
047A03h			
047A04h	CAN1 Mask Register 1	C1MKR1	XXXX XXXXh
047A05h			
047A06h			
047A07h			
047A08h	CAN1 Mask Register 2	C1MKR2	XXXX XXXXh
047A09h			
047A0Ah			
047A0Bh			
047A0Ch	CAN1 Mask Register 3	C1MKR3	XXXX XXXXh
047A0Dh			
047A0Eh			
047A0Fh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.37 SFR List (37)**

Address	Register	Symbol	Reset Value
047A10h	CAN1 Mask Register 4	C1MKR4	XXXX XXXXh
047A11h			
047A12h			
047A13h			
047A14h	CAN1 Mask Register 5	C1MKR5	XXXX XXXXh
047A15h			
047A16h			
047A17h			
047A18h	CAN1 Mask Register 6	C1MKR6	XXXX XXXXh
047A19h			
047A1Ah			
047A1Bh			
047A1Ch	CAN1 Mask Register 7	C1MKR7	XXXX XXXXh
047A1Dh			
047A1Eh			
047A1Fh			
047A20h	CAN1 FIFO Received ID Compare Register 0	C1FIDCR0	XXXX XXXXh
047A21h			
047A22h			
047A23h			
047A24h	CAN1 FIFO Received ID Compare Register 1	C1FIDCR1	XXXX XXXXh
047A25h			
047A26h			
047A27h			
047A28h	CAN1 Mask Invalid Register	C1MKIVLR	XXXX XXXXh
047A29h			
047A2Ah			
047A2Bh			
047A2Ch	CAN1 Mailbox Interrupt Enable Register	C1MIER	XXXX XXXXh
047A2Dh			
047A2Eh			
047A2Fh			
047A30h			
047A31h			
047A32h			
047A33h			
047A34h			
047A35h			
047A36h			
047A37h			
047A38h			
047A39h			
047A3Ah			
047A3Bh			
047A3Ch			
047A3Dh			
047A3Eh			
047A3Fh			
047A40h to 047B1Fh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.38 SFR List (38)**

Address	Register	Symbol	Reset Value
047B20h	CAN1 Message Control Register 0	C1MCTL0	00h
047B21h	CAN1 Message Control Register 1	C1MCTL1	00h
047B22h	CAN1 Message Control Register 2	C1MCTL2	00h
047B23h	CAN1 Message Control Register 3	C1MCTL3	00h
047B24h	CAN1 Message Control Register 4	C1MCTL4	00h
047B25h	CAN1 Message Control Register 5	C1MCTL5	00h
047B26h	CAN1 Message Control Register 6	C1MCTL6	00h
047B27h	CAN1 Message Control Register 7	C1MCTL7	00h
047B28h	CAN1 Message Control Register 8	C1MCTL8	00h
047B29h	CAN1 Message Control Register 9	C1MCTL9	00h
047B2Ah	CAN1 Message Control Register 10	C1MCTL10	00h
047B2Bh	CAN1 Message Control Register 11	C1MCTL11	00h
047B2Ch	CAN1 Message Control Register 12	C1MCTL12	00h
047B2Dh	CAN1 Message Control Register 13	C1MCTL13	00h
047B2Eh	CAN1 Message Control Register 14	C1MCTL14	00h
047B2Fh	CAN1 Message Control Register 15	C1MCTL15	00h
047B30h	CAN1 Message Control Register 16	C1MCTL16	00h
047B31h	CAN1 Message Control Register 17	C1MCTL17	00h
047B32h	CAN1 Message Control Register 18	C1MCTL18	00h
047B33h	CAN1 Message Control Register 19	C1MCTL19	00h
047B34h	CAN1 Message Control Register 20	C1MCTL20	00h
047B35h	CAN1 Message Control Register 21	C1MCTL21	00h
047B36h	CAN1 Message Control Register 22	C1MCTL22	00h
047B37h	CAN1 Message Control Register 23	C1MCTL23	00h
047B38h	CAN1 Message Control Register 24	C1MCTL24	00h
047B39h	CAN1 Message Control Register 25	C1MCTL25	00h
047B3Ah	CAN1 Message Control Register 26	C1MCTL26	00h
047B3Bh	CAN1 Message Control Register 27	C1MCTL27	00h
047B3Ch	CAN1 Message Control Register 28	C1MCTL28	00h
047B3Dh	CAN1 Message Control Register 29	C1MCTL29	00h
047B3Eh	CAN1 Message Control Register 30	C1MCTL30	00h
047B3Fh	CAN1 Message Control Register 31	C1MCTL31	00h

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.39 SFR List (39)**

Address	Register	Symbol	Reset Value
047B40h	CAN1 Control Register	C1CTLR	0000 0101b
047B41h			0000 0000b
047B42h	CAN1 Status Register	C1STR	0000 0101b
047B43h			0000 0000b
047B44h	CAN1 Bit Configuration Register	C1BCR	00 0000h
047B45h			
047B46h			
047B47h	CAN1 Clock Select Register	C1CLKR	000X 0000b
047B48h	CAN1 Receive FIFO Control Register	C1RFCR	1000 0000b
047B49h	CAN1 Receive FIFO Pointer Control Register	C1RFPCR	XXh
047B4Ah	CAN1 Transmit FIFO Control Register	C1TFCR	1000 0000b
047B4Bh	CAN1 Transmit FIFO Pointer Control Register	C1TFPCR	XXh
047B4Ch	CAN1 Error Interrupt Enable Register	C1EIER	00h
047B4Dh	CAN1 Error Interrupt Factor Judge Register	C1EIFR	00h
047B4Eh	CAN1 Receive Error Count Register	C1RECR	00h
047B4Fh	CAN1 Transmit Error Count Register	C1TECR	00h
047B50h	CAN1 Error Code Store Register	C1ECSR	00h
047B51h	CAN1 Channel Search Support Register	C1CSSR	XXh
047B52h	CAN1 Mailbox Search Status Register	C1MSSR	1000 0000b
047B53h	CAN1 Mailbox Search Mode Register	C1MSMR	0000 0000b
047B54h	CAN1 Time Stamp Register	C1TSR	0000h
047B55h			
047B56h	CAN1 Acceptance Filter Support Register	C1AFSR	XXXXh
047B57h			
047B58h	CAN1 Test Control Register	C1TCR	00h
047B59h			
047B5Ah			
047B5Bh			
047B5Ch			
047B5Dh			
047B5Eh			
047B5Fh			
047B60h to 047BFFh			

X: Undefined

Blanks are reserved. No access is allowed.



**Table 4.40 SFR List (40)**

Address	Register	Symbol	Reset Value			
047C00h	CAN0 Mailbox 0: Message Identifier	COMB0	XXXX XXXXh			
047C01h						
047C02h						
047C03h						
047C04h						
047C05h	CAN0 Mailbox 0: Data Length		XXh			
047C06h	CAN0 Mailbox 0: Data Field		XXXX XXXX XXXX XXXXh			
047C07h						
047C08h						
047C09h						
047C0Ah						
047C0Bh						
047C0Ch						
047C0Dh						
047C0Eh				CAN0 Mailbox 0: Time Stamp		XXXXh
047C0Fh						
047C10h	CAN0 Mailbox 1: Message Identifier	COMB1	XXXX XXXXh			
047C11h						
047C12h						
047C13h						
047C14h						
047C15h	CAN0 Mailbox 1: Data Length		XXh			
047C16h	CAN0 Mailbox 1: Data Field		XXXX XXXX XXXX XXXXh			
047C17h						
047C18h						
047C19h						
047C1Ah						
047C1Bh						
047C1Ch						
047C1Dh						
047C1Eh				CAN0 Mailbox 1: Time Stamp		XXXXh
047C1Fh						
047C20h	CAN0 Mailbox 2: Message Identifier	COMB2	XXXX XXXXh			
047C21h						
047C22h						
047C23h						
047C24h						
047C25h	CAN0 Mailbox 2: Data Length		XXh			
047C26h	CAN0 Mailbox 2: Data Field		XXXX XXXX XXXX XXXXh			
047C27h						
047C28h						
047C29h						
047C2Ah						
047C2Bh						
047C2Ch						
047C2Dh						
047C2Eh				CAN0 Mailbox 2: Time Stamp		XXXXh
047C2Fh						

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.41 SFR List (41)**

Address	Register	Symbol	Reset Value
047C30h	CAN0 Mailbox 3: Message Identifier	COMB3	XXXX XXXXh
047C31h			
047C32h			
047C33h			
047C34h			
047C35h	CAN0 Mailbox 3: Data Length		XXh
047C36h	CAN0 Mailbox 3: Data Field		XXXX XXXX XXXX XXXXh
047C37h			
047C38h			
047C39h			
047C3Ah			
047C3Bh			
047C3Ch			
047C3Dh			
047C3Eh	CAN0 Mailbox 3: Time Stamp		XXXXh
047C3Fh			
047C40h	CAN0 Mailbox 4: Message Identifier	COMB4	XXXX XXXXh
047C41h			
047C42h			
047C43h			
047C44h			
047C45h	CAN0 Mailbox 4: Data Length		XXh
047C46h	CAN0 Mailbox 4: Data Field		XXXX XXXX XXXX XXXXh
047C47h			
047C48h			
047C49h			
047C4Ah			
047C4Bh			
047C4Ch			
047C4Dh			
047C4Eh	CAN0 Mailbox 4: Time Stamp		XXXXh
047C4Fh			
047C50h	CAN0 Mailbox 5: Message Identifier	COMB5	XXXX XXXXh
047C51h			
047C52h			
047C53h			
047C54h			
047C55h	CAN0 Mailbox 5: Data Length		XXh
047C56h	CAN0 Mailbox 5: Data Field		XXXX XXXX XXXX XXXXh
047C57h			
047C58h			
047C59h			
047C5Ah			
047C5Bh			
047C5Ch			
047C5Dh			
047C5Eh	CAN0 Mailbox 5: Time Stamp		XXXXh
047C5Fh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.42 SFR List (42)**

Address	Register	Symbol	Reset Value
047C60h	CAN0 Mailbox 6: Message Identifier	COMB6	XXXX XXXXh
047C61h			
047C62h			
047C63h			
047C64h			
047C65h	CAN0 Mailbox 6: Data Length		XXh
047C66h	CAN0 Mailbox 6: Data Field		XXXX XXXX XXXX XXXXh
047C67h			
047C68h			
047C69h			
047C6Ah			
047C6Bh			
047C6Ch			
047C6Dh			
047C6Eh	CAN0 Mailbox 6: Time Stamp		XXXXh
047C6Fh			
047C70h	CAN0 Mailbox 7: Message Identifier	COMB7	XXXX XXXXh
047C71h			
047C72h			
047C73h			
047C74h			
047C75h	CAN0 Mailbox 7: Data Length		XXh
047C76h	CAN0 Mailbox 7: Data Field		XXXX XXXX XXXX XXXXh
047C77h			
047C78h			
047C79h			
047C7Ah			
047C7Bh			
047C7Ch			
047C7Dh			
047C7Eh	CAN0 Mailbox 7: Time Stamp		XXXXh
047C7Fh			
047C80h	CAN0 Mailbox 8: Message Identifier	COMB8	XXXX XXXXh
047C81h			
047C82h			
047C83h			
047C84h			
047C85h	CAN0 Mailbox 8: Data Length		XXh
047C86h	CAN0 Mailbox 8: Data Field		XXXX XXXX XXXX XXXXh
047C87h			
047C88h			
047C89h			
047C8Ah			
047C8Bh			
047C8Ch			
047C8Dh			
047C8Eh	CAN0 Mailbox 8: Time Stamp		XXXXh
047C8Fh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.43 SFR List (43)**

Address	Register	Symbol	Reset Value
047C90h	CAN0 Mailbox 9: Message Identifier	COMB9	XXXX XXXXh
047C91h			
047C92h			
047C93h			
047C94h			
047C95h	CAN0 Mailbox 9: Data Length		XXh
047C96h	CAN0 Mailbox 9: Data Field		XXXX XXXX XXXX XXXXh
047C97h			
047C98h			
047C99h			
047C9Ah			
047C9Bh			
047C9Ch			
047C9Dh			
047C9Eh	CAN0 Mailbox 9: Time Stamp		XXXXh
047C9Fh			
047CA0h	CAN0 Mailbox 10: Message Identifier	COMB10	XXXX XXXXh
047CA1h			
047CA2h			
047CA3h			
047CA4h			
047CA5h	CAN0 Mailbox 10: Data Length		XXh
047CA6h	CAN0 Mailbox 10: Data Field		XXXX XXXX XXXX XXXXh
047CA7h			
047CA8h			
047CA9h			
047CAAh			
047CABh			
047CACh			
047CADh			
047CAEh	CAN0 Mailbox 10: Time Stamp		XXXXh
047CAFh			
047CB0h	CAN0 Mailbox 11: Message Identifier	COMB11	XXXX XXXXh
047CB1h			
047CB2h			
047CB3h			
047CB4h			
047CB5h	CAN0 Mailbox 11: Data Length		XXh
047CB6h	CAN0 Mailbox 11: Data Field		XXXX XXXX XXXX XXXXh
047CB7h			
047CB8h			
047CB9h			
047CBAh			
047CBBh			
047CBCCh			
047CBDh			
047CBEh	CAN0 Mailbox 11: Time Stamp		XXXXh
047CBFh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.44 SFR List (44)**

Address	Register	Symbol	Reset Value
047CC0h	CAN0 Mailbox 12: Message Identifier	COMB12	XXXX XXXXh
047CC1h			
047CC2h			
047CC3h			
047CC4h			
047CC5h	CAN0 Mailbox 12: Data Length		XXh
047CC6h	CAN0 Mailbox 12: Data Field		XXXX XXXX XXXX XXXXh
047CC7h			
047CC8h			
047CC9h			
047CCAh			
047CCBh			
047CCCh			
047CCDh			
047CCEh			
047CCFh			
047CD0h	CAN0 Mailbox 13: Message Identifier	COMB13	XXXX XXXXh
047CD1h			
047CD2h			
047CD3h			
047CD4h			
047CD5h	CAN0 Mailbox 13: Data Length		XXh
047CD6h	CAN0 Mailbox 13: Data Field		XXXX XXXX XXXX XXXXh
047CD7h			
047CD8h			
047CD9h			
047CDAh			
047CDBh			
047CDCh			
047CDDh			
047CDEh			
047CDFh			
047CE0h	CAN0 Mailbox 14: Message Identifier	COMB14	XXXX XXXXh
047CE1h			
047CE2h			
047CE3h			
047CE4h			
047CE5h	CAN0 Mailbox 14: Data Length		XXh
047CE6h	CAN0 Mailbox 14: Data Field		XXXX XXXX XXXX XXXXh
047CE7h			
047CE8h			
047CE9h			
047CEAh			
047CEBh			
047CECh			
047CEDh			
047CEEh			
047CEFh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.45 SFR List (45)**

Address	Register	Symbol	Reset Value
047CF0h	CAN0 Mailbox 15: Message Identifier	C0MB15	XXXX XXXXh
047CF1h			
047CF2h			
047CF3h			
047CF4h			
047CF5h	CAN0 Mailbox 15: Data Length		XXh
047CF6h	CAN0 Mailbox 15: Data Field		XXXX XXXX XXXX XXXXh
047CF7h			
047CF8h			
047CF9h			
047CFAh			
047CFBh			
047CFCh			
047CFDh			
047CFEh	CAN0 Mailbox 15: Time Stamp		XXXXh
047CFFh			
047D00h	CAN0 Mailbox 16: Message Identifier	C0MB16	XXXX XXXXh
047D01h			
047D02h			
047D03h			
047D04h			
047D05h	CAN0 Mailbox 16: Data Length		XXh
047D06h	CAN0 Mailbox 16: Data Field		XXXX XXXX XXXX XXXXh
047D07h			
047D08h			
047D09h			
047D0Ah			
047D0Bh			
047D0Ch			
047D0Dh			
047D0Eh	CAN0 Mailbox 16: Time Stamp		XXXXh
047D0Fh			
047D10h	CAN0 Mailbox 17: Message Identifier	C0MB17	XXXX XXXXh
047D11h			
047D12h			
047D13h			
047D14h			
047D15h	CAN0 Mailbox 17: Data Length		XXh
047D16h	CAN0 Mailbox 17: Data Field		XXXX XXXX XXXX XXXXh
047D17h			
047D18h			
047D19h			
047D1Ah			
047D1Bh			
047D1Ch			
047D1Dh			
047D1Eh	CAN0 Mailbox 17: Time Stamp		XXXXh
047D1Fh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.46 SFR List (46)**

Address	Register	Symbol	Reset Value
047D20h	CAN0 Mailbox 18: Message Identifier	C0MB18	XXXX XXXXh
047D21h			
047D22h			
047D23h			
047D24h			
047D25h	CAN0 Mailbox 18: Data Length		XXh
047D26h	CAN0 Mailbox 18: Data Field		XXXX XXXX XXXX XXXXh
047D27h			
047D28h			
047D29h			
047D2Ah			
047D2Bh			
047D2Ch			
047D2Dh			
047D2Eh	CAN0 Mailbox 18: Time Stamp		XXXXh
047D2Fh			
047D30h	CAN0 Mailbox 19: Message Identifier	C0MB19	XXXX XXXXh
047D31h			
047D32h			
047D33h			
047D34h			
047D35h	CAN0 Mailbox 19: Data Length		XXh
047D36h	CAN0 Mailbox 19: Data Field		XXXX XXXX XXXX XXXXh
047D37h			
047D38h			
047D39h			
047D3Ah			
047D3Bh			
047D3Ch			
047D3Dh			
047D3Eh	CAN0 Mailbox 19: Time Stamp		XXXXh
047D3Fh			
047D40h	CAN0 Mailbox 20: Message Identifier	C0MB20	XXXX XXXXh
047D41h			
047D42h			
047D43h			
047D44h			
047D45h	CAN0 Mailbox 20: Data Length		XXh
047D46h	CAN0 Mailbox 20: Data Field		XXXX XXXX XXXX XXXXh
047D47h			
047D48h			
047D49h			
047D4Ah			
047D4Bh			
047D4Ch			
047D4Dh			
047D4Eh	CAN0 Mailbox 20: Time Stamp		XXXXh
047D4Fh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.47 SFR List (47)**

Address	Register	Symbol	Reset Value
047D50h	CAN0 Mailbox 21: Message Identifier	C0MB21	XXXX XXXXh
047D51h			
047D52h			
047D53h			
047D54h			
047D55h	CAN0 Mailbox 21: Data Length		XXh
047D56h	CAN0 Mailbox 21: Data Field		XXXX XXXX XXXX XXXXh
047D57h			
047D58h			
047D59h			
047D5Ah			
047D5Bh			
047D5Ch			
047D5Dh			
047D5Eh	CAN0 Mailbox 21: Time Stamp		XXXXh
047D5Fh			
047D60h	CAN0 Mailbox 22: Message Identifier	C0MB22	XXXX XXXXh
047D61h			
047D62h			
047D63h			
047D64h			
047D65h	CAN0 Mailbox 22: Data Length		XXh
047D66h	CAN0 Mailbox 22: Data Field		XXXX XXXX XXXX XXXXh
047D67h			
047D68h			
047D69h			
047D6Ah			
047D6Bh			
047D6Ch			
047D6Dh			
047D6Eh	CAN0 Mailbox 22: Time Stamp		XXXXh
047D6Fh			
047D70h	CAN0 Mailbox 23: Message Identifier	C0MB23	XXXX XXXXh
047D71h			
047D72h			
047D73h			
047D74h			
047D75h	CAN0 Mailbox 23: Data Length		XXh
047D76h	CAN0 Mailbox 23: Data Field		XXXX XXXX XXXX XXXXh
047D77h			
047D78h			
047D79h			
047D7Ah			
047D7Bh			
047D7Ch			
047D7Dh			
047D7Eh	CAN0 Mailbox 23: Time Stamp		XXXXh
047D7Fh			

X: Undefined

Blanks are reserved. No access is allowed.



**Table 4.48 SFR List (48)**

Address	Register	Symbol	Reset Value
047D80h	CAN0 Mailbox 24: Message Identifier	C0MB24	XXXX XXXXh
047D81h			
047D82h			
047D83h			
047D84h			
047D85h	CAN0 Mailbox 24: Data Length		XXh
047D86h	CAN0 Mailbox 24: Data Field		XXXX XXXX XXXX XXXXh
047D87h			
047D88h			
047D89h			
047D8Ah			
047D8Bh			
047D8Ch			
047D8Dh			
047D8Eh	CAN0 Mailbox 24: Time Stamp		XXXXh
047D8Fh			
047D90h	CAN0 Mailbox 25: Message Identifier	C0MB25	XXXX XXXXh
047D91h			
047D92h			
047D93h			
047D94h			
047D95h	CAN0 Mailbox 25: Data Length		XXh
047D96h	CAN0 Mailbox 25: Data Field		XXXX XXXX XXXX XXXXh
047D97h			
047D98h			
047D99h			
047D9Ah			
047D9Bh			
047D9Ch			
047D9Dh			
047D9Eh	CAN0 Mailbox 25: Time Stamp		XXXXh
047D9Fh			
047DA0h	CAN0 Mailbox 26: Message Identifier	C0MB26	XXXX XXXXh
047DA1h			
047DA2h			
047DA3h			
047DA4h			
047DA5h	CAN0 Mailbox 26: Data Length		XXh
047DA6h	CAN0 Mailbox 26: Data Field		XXXX XXXX XXXX XXXXh
047DA7h			
047DA8h			
047DA9h			
047DAAh			
047DABh			
047DACH			
047DADh			
047DAEh	CAN0 Mailbox 26: Time Stamp		XXXXh
047DAFh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.49 SFR List (49)**

Address	Register	Symbol	Reset Value
047DB0h	CAN0 Mailbox 27: Message Identifier	C0MB27	XXXX XXXXh
047DB1h			
047DB2h			
047DB3h			
047DB4h			
047DB5h	CAN0 Mailbox 27: Data Length		XXh
047DB6h	CAN0 Mailbox 27: Data Field		XXXX XXXX XXXX XXXXh
047DB7h			
047DB8h			
047DB9h			
047DBAh			
047DBBh			
047DBCh			
047DBDh			
047DBEh			
047DBEh	CAN0 Mailbox 27: Time Stamp		XXXXh
047DBFh			
047DC0h	CAN0 Mailbox 28: Message Identifier	C0MB28	XXXX XXXXh
047DC1h			
047DC2h			
047DC3h			
047DC4h			
047DC5h	CAN0 Mailbox 28: Data Length		XXh
047DC6h	CAN0 Mailbox 28: Data Field		XXXX XXXX XXXX XXXXh
047DC7h			
047DC8h			
047DC9h			
047DCAh			
047DCBh			
047DCCCh			
047DCDh			
047DCEh			
047DCEh	CAN0 Mailbox 28: Time Stamp		XXXXh
047DCFh			
047DD0h	CAN0 Mailbox 29: Message Identifier	C0MB29	XXXX XXXXh
047DD1h			
047DD2h			
047DD3h			
047DD4h			
047DD5h	CAN0 Mailbox 29: Data Length		XXh
047DD6h	CAN0 Mailbox 29: Data Field		XXXX XXXX XXXX XXXXh
047DD7h			
047DD8h			
047DD9h			
047DDAh			
047DDBh			
047DDCh			
047DDDh			
047DDEh			
047DDEh	CAN0 Mailbox 29: Time Stamp		XXXXh
047DDFh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.50 SFR List (50)**

Address	Register	Symbol	Reset Value
047DE0h	CAN0 Mailbox 30: Message Identifier	COMB30	XXXX XXXXh
047DE1h			
047DE2h			
047DE3h			
047DE4h			
047DE5h	CAN0 Mailbox 30: Data Length		XXh
047DE6h	CAN0 Mailbox 30: Data Field		XXXX XXXX XXXX XXXXh
047DE7h			
047DE8h			
047DE9h			
047DEAh			
047DEBh			
047DECh			
047DEDh			
047DEEh			
047DEEh	CAN0 Mailbox 30: Time Stamp		XXXXh
047DEFh			
047DF0h	CAN0 Mailbox 31: Message Identifier	COMB31	XXXX XXXXh
047DF1h			
047DF2h			
047DF3h			
047DF4h			
047DF5h	CAN0 Mailbox 31: Data Length		XXh
047DF6h	CAN0 Mailbox 31: Data Field		XXXX XXXX XXXX XXXXh
047DF7h			
047DF8h			
047DF9h			
047DFAh			
047DFBh			
047DFCh			
047DFDh			
047DFEh			
047DFEh	CAN0 Mailbox 31: Time Stamp		XXXXh
047DFFh			
047E00h	CAN0 Mask Register 0	COMKR0	XXXX XXXXh
047E01h			
047E02h			
047E03h			
047E04h	CAN0 Mask Register 1	COMKR1	XXXX XXXXh
047E05h			
047E06h			
047E07h			
047E08h	CAN0 Mask Register 2	COMKR2	XXXX XXXXh
047E09h			
047E0Ah			
047E0Bh			
047E0Ch	CAN0 Mask Register 3	COMKR3	XXXX XXXXh
047E0Dh			
047E0Eh			
047E0Fh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.51 SFR List (51)**

Address	Register	Symbol	Reset Value
047E10h	CAN0 Mask Register 4	COMKR4	XXXX XXXXh
047E11h			
047E12h			
047E13h			
047E14h	CAN0 Mask Register 5	COMKR5	XXXX XXXXh
047E15h			
047E16h			
047E17h			
047E18h	CAN0 Mask Register 6	COMKR6	XXXX XXXXh
047E19h			
047E1Ah			
047E1Bh			
047E1Ch	CAN0 Mask Register 7	COMKR7	XXXX XXXXh
047E1Dh			
047E1Eh			
047E1Fh			
047E20h	CAN0 FIFO Receive ID Compare Register 0	C0FIDCR0	XXXX XXXXh
047E21h			
047E22h			
047E23h			
047E24h	CAN0 FIFO Receive ID Compare Register 1	C0FIDCR1	XXXX XXXXh
047E25h			
047E26h			
047E27h			
047E28h	CAN0 Mask Invalid Register	COMKIVLR	XXXX XXXXh
047E29h			
047E2Ah			
047E2Bh			
047E2Ch	CAN0 Mailbox Interrupt Enable Register	COMIER	XXXX XXXXh
047E2Dh			
047E2Eh			
047E2Fh			
047E30h			
047E31h			
047E32h			
047E33h			
047E34h			
047E35h			
047E36h			
047E37h			
047E38h			
047E39h			
047E3Ah			
047E3Bh			
047E3Ch			
047E3Dh			
047E3Eh			
047E3Fh			
047E40h to 047F1Fh			

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.52 SFR List (52)**

Address	Register	Symbol	Reset Value
047F20h	CAN0 Message Control Register 0	COMCTL0	00h
047F21h	CAN0 Message Control Register 1	COMCTL1	00h
047F22h	CAN0 Message Control Register 2	COMCTL2	00h
047F23h	CAN0 Message Control Register 3	COMCTL3	00h
047F24h	CAN0 Message Control Register 4	COMCTL4	00h
047F25h	CAN0 Message Control Register 5	COMCTL5	00h
047F26h	CAN0 Message Control Register 6	COMCTL6	00h
047F27h	CAN0 Message Control Register 7	COMCTL7	00h
047F28h	CAN0 Message Control Register 8	COMCTL8	00h
047F29h	CAN0 Message Control Register 9	COMCTL9	00h
047F2Ah	CAN0 Message Control Register 10	COMCTL10	00h
047F2Bh	CAN0 Message Control Register 11	COMCTL11	00h
047F2Ch	CAN0 Message Control Register 12	COMCTL12	00h
047F2Dh	CAN0 Message Control Register 13	COMCTL13	00h
047F2Eh	CAN0 Message Control Register 14	COMCTL14	00h
047F2Fh	CAN0 Message Control Register 15	COMCTL15	00h
047F30h	CAN0 Message Control Register 16	COMCTL16	00h
047F31h	CAN0 Message Control Register 17	COMCTL17	00h
047F32h	CAN0 Message Control Register 18	COMCTL18	00h
047F33h	CAN0 Message Control Register 19	COMCTL19	00h
047F34h	CAN0 Message Control Register 20	COMCTL20	00h
047F35h	CAN0 Message Control Register 21	COMCTL21	00h
047F36h	CAN0 Message Control Register 22	COMCTL22	00h
047F37h	CAN0 Message Control Register 23	COMCTL23	00h
047F38h	CAN0 Message Control Register 24	COMCTL24	00h
047F39h	CAN0 Message Control Register 25	COMCTL25	00h
047F3Ah	CAN0 Message Control Register 26	COMCTL26	00h
047F3Bh	CAN0 Message Control Register 27	COMCTL27	00h
047F3Ch	CAN0 Message Control Register 28	COMCTL28	00h
047F3Dh	CAN0 Message Control Register 29	COMCTL29	00h
047F3Eh	CAN0 Message Control Register 30	COMCTL30	00h
047F3Fh	CAN0 Message Control Register 31	COMCTL31	00h

X: Undefined

Blanks are reserved. No access is allowed.

**Table 4.53 SFR List (53)**

Address	Register	Symbol	Reset Value
047F40h	CAN0 Control Register	C0CTLR	0000 0101b
047F41h			0000 0000b
047F42h	CAN0 Status Register	C0STR	0000 0101b
047F43h			0000 0000b
047F44h	CAN0 Bit Configuration Register	C0BCR	00 0000h
047F45h			
047F46h			
047F47h	CAN0 Clock Select Register	C0CLKR	000X 0000b
047F48h	CAN0 Receive FIFO Control Register	C0RFCR	1000 0000b
047F49h	CAN0 Receive FIFO Pointer Control Register	C0RFPCR	XXh
047F4Ah	CAN0 Transmit FIFO Control Register	C0TFCR	1000 0000b
047F4Bh	CAN0 Transmit FIFO Pointer Control Register	C0TFPCR	XXh
047F4Ch	CAN0 Error Interrupt Enable Register	C0EIER	00h
047F4Dh	CAN0 Error Interrupt Factor Judge Register	C0EIFR	00h
047F4Eh	CAN0 Receive Error Count Register	C0RECR	00h
047F4Fh	CAN0 Transmit Error Count Register	C0TECR	00h
047F50h	CAN0 Error Code Store Register	C0ECSR	00h
047F51h	CAN0 Channel Search Support Register	C0CSSR	XXh
047F52h	CAN0 Mailbox Search Status Register	C0MSSR	1000 0000b
047F53h	CAN0 Mailbox Search Mode Register	C0MSMR	0000 0000b
047F54h	CAN0 Time Stamp Register	C0TSR	0000h
047F55h			
047F56h	CAN0 Acceptance Filter Support Register	C0AFSR	XXXXh
047F57h			
047F58h	CAN0 Test Control Register	C0TCR	00h
047F59h			
047F5Ah			
047F5Bh			
047F5Ch			
047F5Dh			
047F5Eh			
047F5Fh			
047F60h to 047FFFh			
048000h to 04FFFFh			

X: Undefined

Blanks are reserved. No access is allowed.

## 5. Resets

There are three types of operations for resetting the MCU: hardware reset, software reset, and watchdog timer reset.

### 5.1 Hardware Reset

A hardware reset is generated when a low signal is applied to the  $\overline{\text{RESET}}$  pin under the recommended operating conditions of the supply voltage. When the  $\overline{\text{RESET}}$  pin is driven low, all pins, and oscillators are reset (refer to Table 5.1 for details), and the main clock starts oscillating. The CPU and SFRs are reset by a low-to-high transition on the  $\overline{\text{RESET}}$  pin. Then, the CPU starts executing the program from the address indicated by the reset vector. Internal RAM is not affected by a hardware reset. However, if a hardware reset occurs during a write operation to the internal RAM, the value is undefined.

Figure 5.1 shows an example of the reset circuit. Figure 5.2 shows the reset sequence. Table 5.1 lists pin states while the  $\overline{\text{RESET}}$  pin is held low. Figure 5.3 shows CPU register states after a reset. Refer to 4. “Special Function Registers (SFRs)” for details on the states of SFRs after a reset.

#### A. Reset when the supply voltage is stable

- (1) Drive the  $\overline{\text{RESET}}$  pin low.
- (2) Input at least 20 clock cycles to the XIN pin.
- (3) Drive the  $\overline{\text{RESET}}$  pin high.

#### B. Reset when turning on the power

- (1) Drive the  $\overline{\text{RESET}}$  pin low.
- (2) Raise the supply voltage to the recommended operating voltage.
- (3) Wait  $t_d(\text{P-R})$  ms until the internal voltage is stabilized.
- (4) Input at least 20 clock cycles to the XIN pin.
- (5) Drive the  $\overline{\text{RESET}}$  pin high.

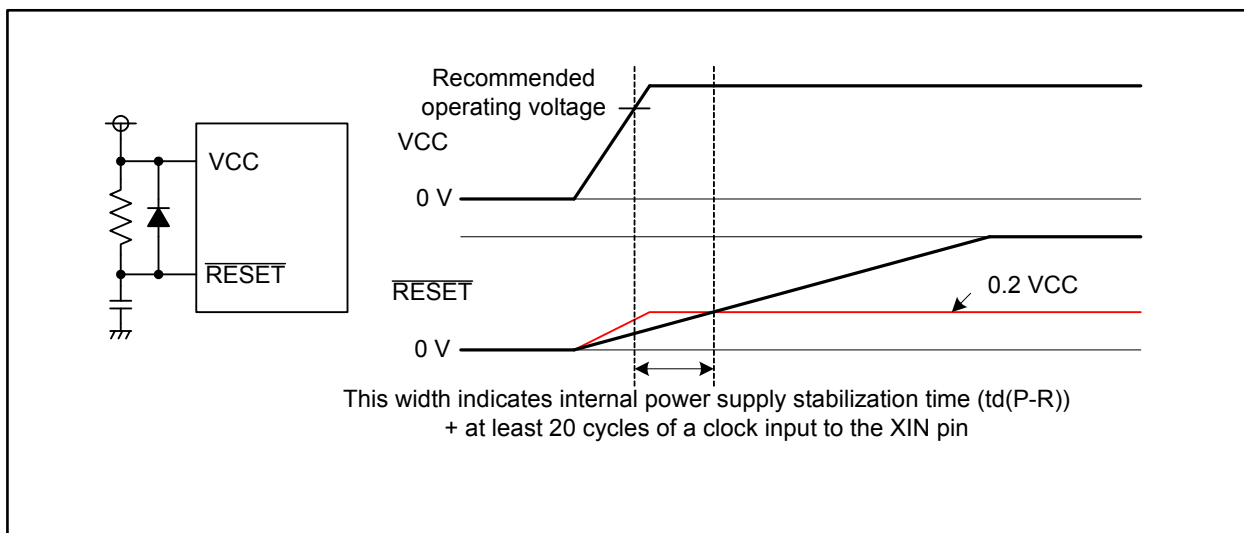


Figure 5.1 Reset Circuitry





## 5.2 Software Reset

The CPU, SFRs, and pins are reset when the PM03 bit in the PM0 register is set to 1 (the MCU is reset). Then, the CPU executes the program from the address indicated by the reset vector. Figure 5.4 shows the PM0 register.

Set the PM03 bit to 1 while the PLL clock is selected as the CPU clock source and the main clock oscillation is completely stable.

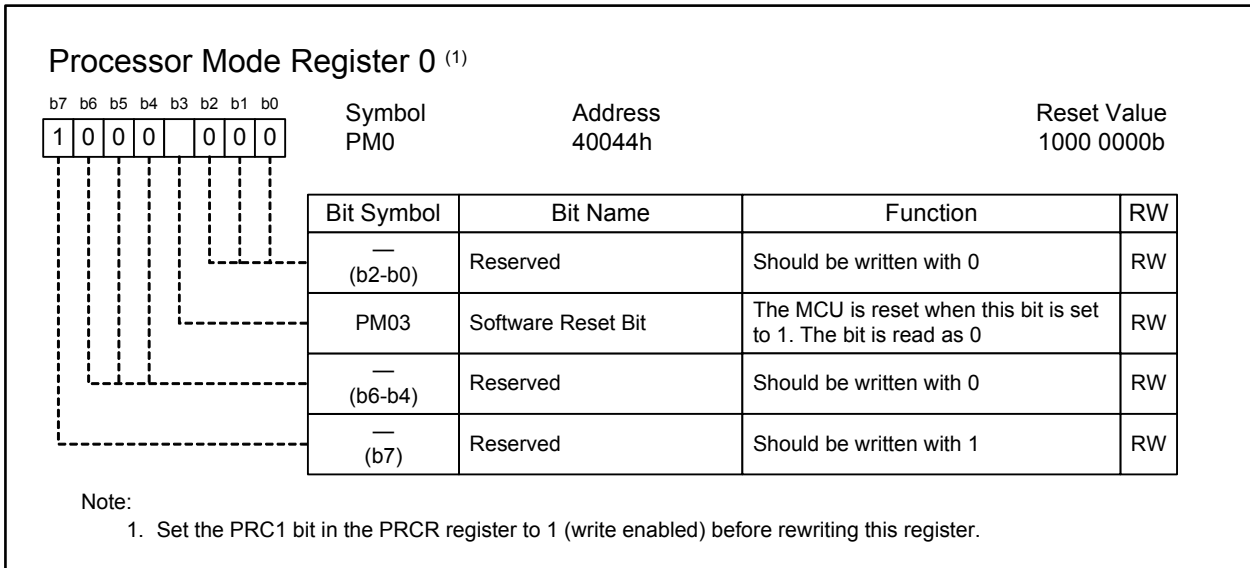


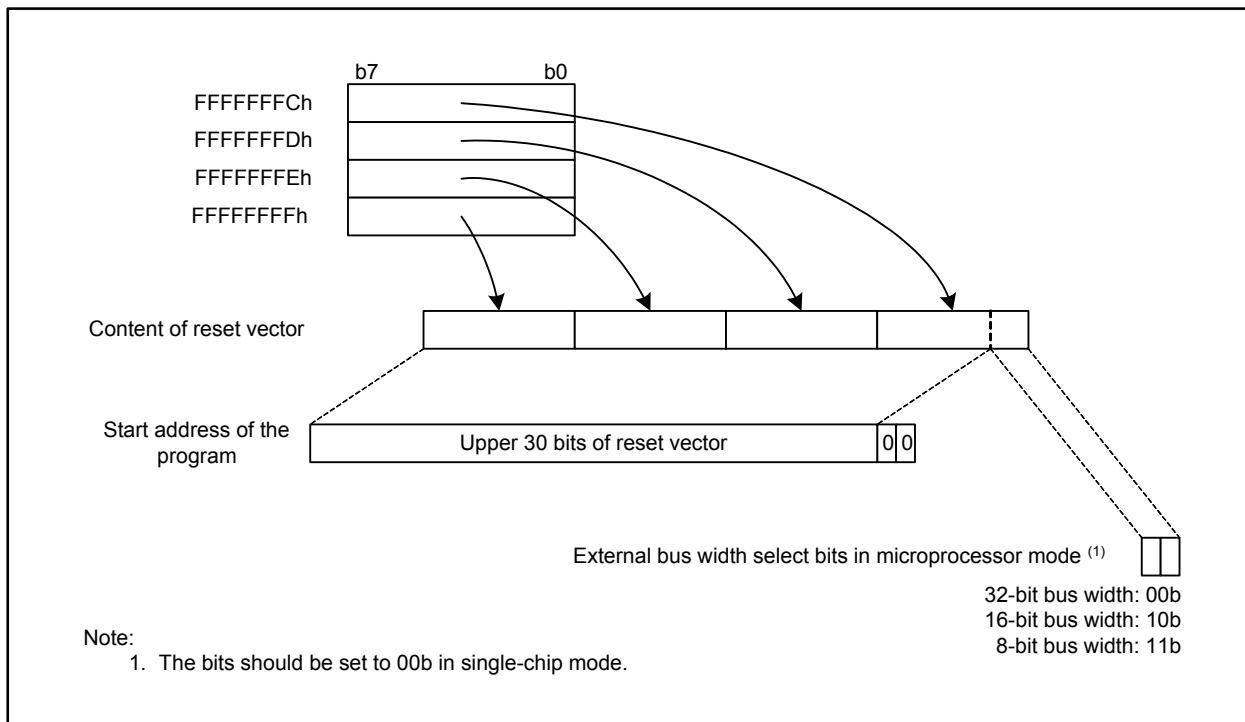
Figure 5.4 PM0 Register

## 5.3 Watchdog Timer Reset

The CPU, SFRs, and pins are reset when the watchdog timer underflows while the CM06 bit in the CM0 register is 1 (reset when watchdog timer underflows). Then, the CPU executes the program from the address indicated by the reset vector.

### 5.4 Reset Vector

The reset vector in the R32C/100 Series is configured as shown in Figure 5.5. The start address of a program consists of the upper 30 bits of the reset vector and 00b as lower 2 bits. The lower 2 bits of the reset vector are bits to select the external bus width in microprocessor mode. Therefore, the start address of a program requires 4-byte alignment so that the lower 2 bits are 00b. In single-chip mode, set the external bus width select bits to 00b.



**Figure 5.5 Reset Vector Configuration**

## 6. Power Management

### 6.1 Voltage Regulators for Internal Logic

The supply voltage for internal logic is generated by reducing the input voltage from the VCC pin with the voltage regulators. Figure 6.1 shows a block diagram of the voltage regulators for internal logic, and Figure 6.2 shows the VRCCR register.

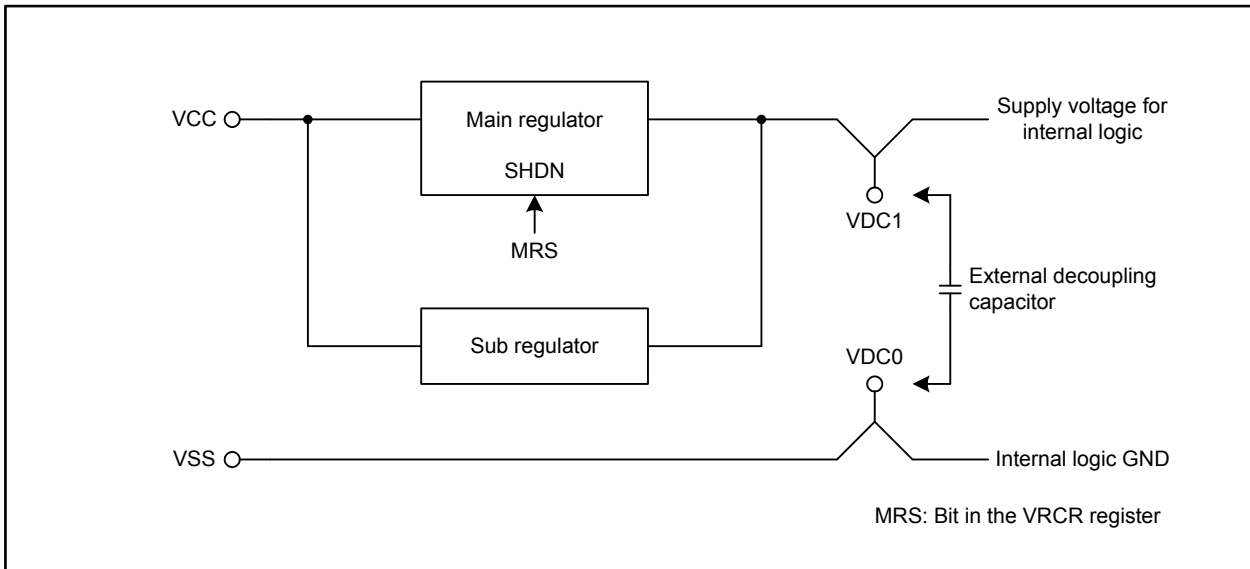


Figure 6.1 Block Diagram of Voltage Regulators for Internal Logic

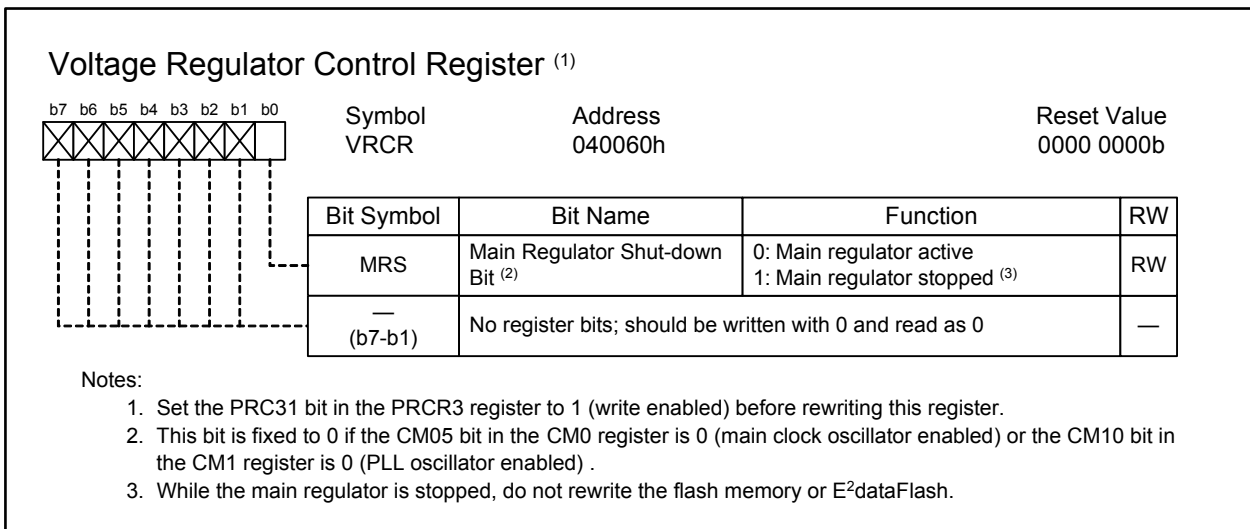


Figure 6.2 VRCCR Register

### 6.1.1 Decoupling Capacitor

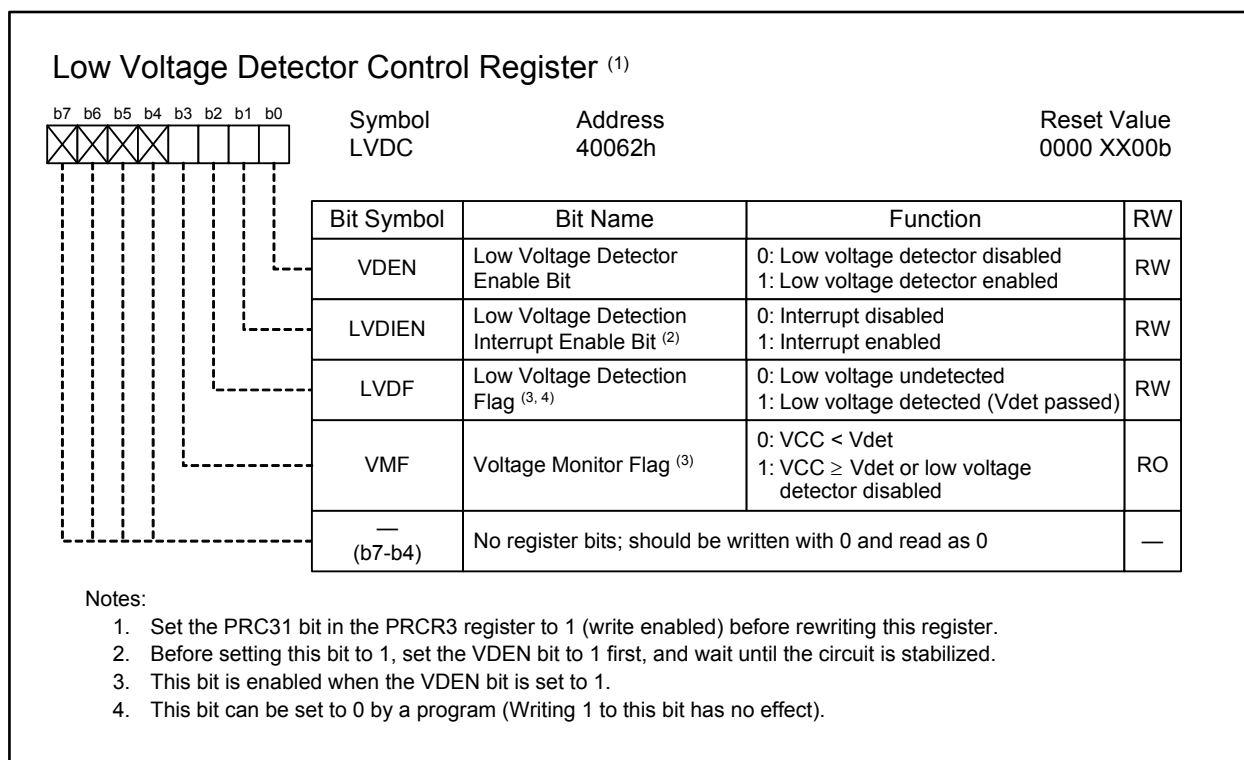
An external decoupling capacitor is required to stabilize internal voltage. The capacitor should be beneficially effective at higher frequencies and maintain a more stable capacitance irrespective of temperature change. In general, ceramic capacitors are recommended. The capacitance varies by conditions such as operating temperature, DC bias, and aging. To select an appropriate capacitor, these conditions should be considered. Also, refer to the recommended capacitor specifications listed in Table 6.1.

The traces between the capacitor and the VDC1/VDC0 pins should be as short and wide as physically possible.

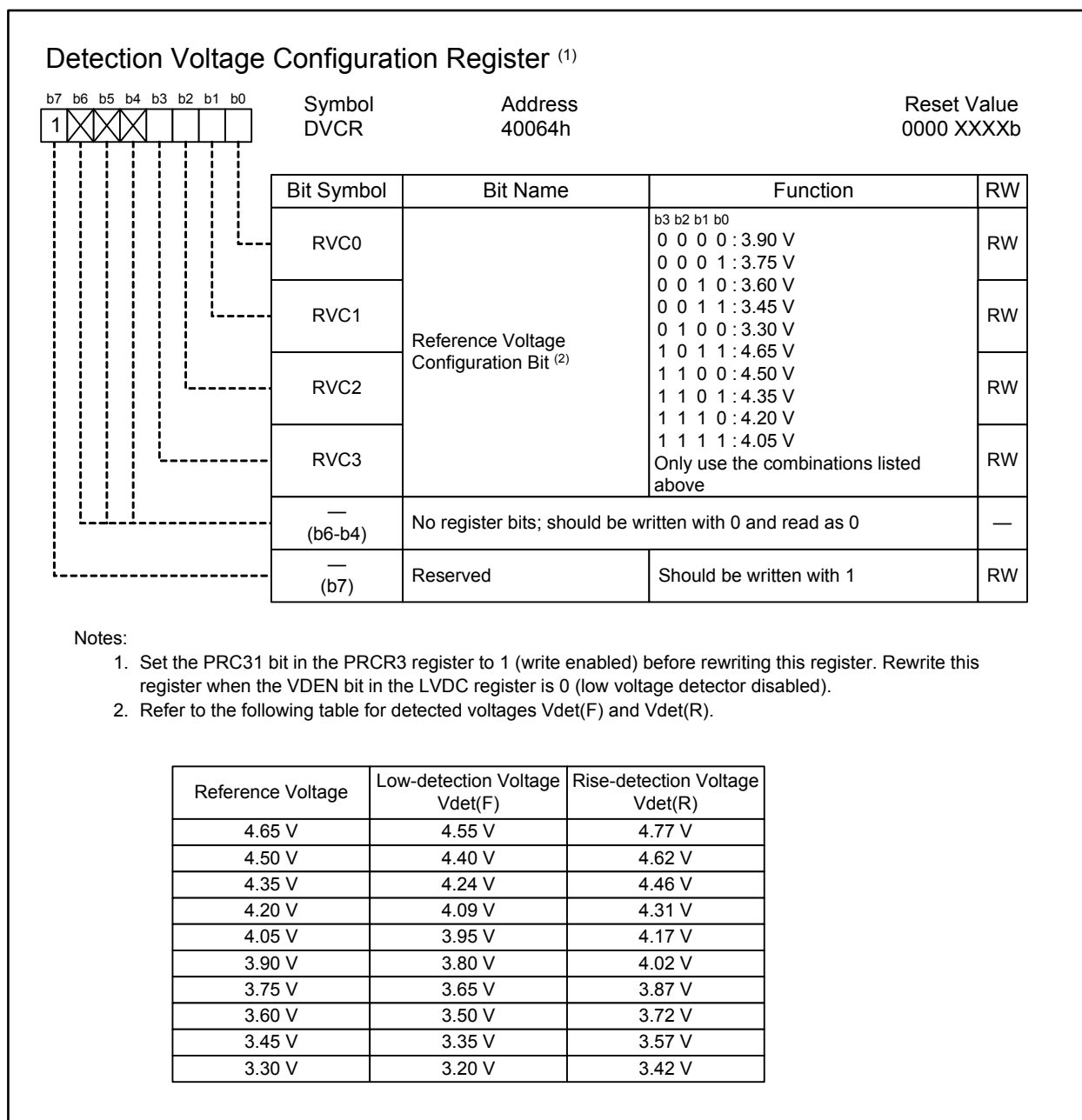
**Table 6.1 Recommended Capacitor Specifications**

Applicable standard		Temperature Characteristics		Rated Voltage (V)	Nominal Capacitance ( $\mu$ F)	Capacitance Tolerance (%)
		Operating temperature range ( $^{\circ}$ C)	Capacitance change (%)			
B	JIS	-25 to 85	$\pm$ 10	6.3 or higher	4.7	$\pm$ 20 or better
R	JIS	-55 to 125	$\pm$ 15	6.3 or higher	4.7	$\pm$ 20 or better
X5R	EIA	-55 to 85	$\pm$ 15	6.3 or higher	4.7	$\pm$ 20 or better
X7R	EIA	-55 to 125	$\pm$ 15	6.3 or higher	4.7	$\pm$ 20 or better
X8R	EIA	-55 to 150	$\pm$ 15	6.3 or higher	4.7	$\pm$ 20 or better
X6S	EIA	-55 to 105	$\pm$ 22	6.3 or higher	4.7	$\pm$ 20 or better
X7S	EIA	-55 to 125	$\pm$ 22	6.3 or higher	4.7	$\pm$ 20 or better





**Figure 6.4 LVDC Register**



**Figure 6.5 DVCR Register**

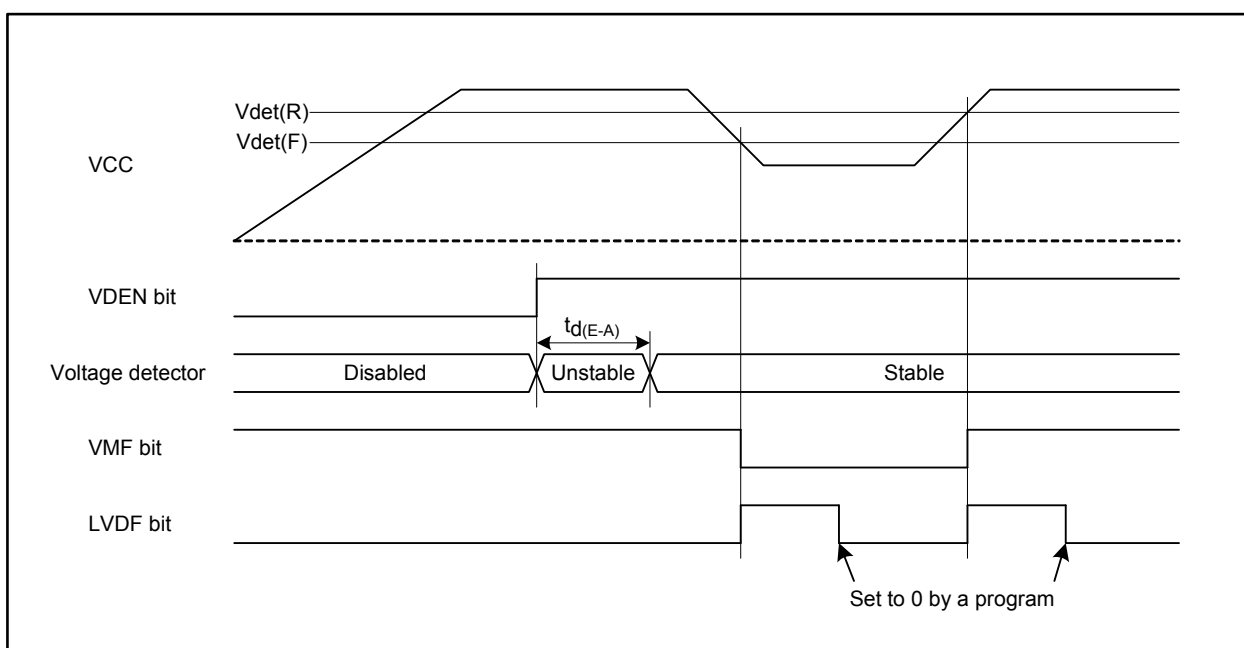
### 6.2.1 Operational State of Low Voltage Detector

The low voltage detector starts operating stably after  $t_{d(E-A)}$  when the VDEN bit in the LVDC register is set to 1 (low voltage detector enabled).

When the input voltage to the VCC pin drops below  $V_{det(F)}$ , the VMF bit becomes 0 ( $V_{CC} < V_{det}$ ) and the LVDF bit becomes 1 (low voltage detected ( $V_{det}$  passed)). At this point an interrupt request is generated when the LVDIEN bit is 1 (low voltage detection interrupt enabled). Set the LVDF bit to 0 (low voltage undetected) by a program.

When the voltage rises above  $V_{det(R)}$  again, the VMF bit becomes to 1 ( $V_{CC} \geq V_{det}$ ) and the LVDF bit becomes 1. At this point an interrupt request is generated when the LVDIEN bit is 1.

Figure 6.6 shows the operation of the low voltage detector.



**Figure 6.6** Low Voltage Detector Operation

### 6.2.2 Low Voltage Detection Interrupt

A low voltage detection interrupt request is generated when the input voltage at the VCC pin rises above the  $V_{det(R)}$  level, or falls below the  $V_{det(F)}$  level while the LVDIEN bit in the LVDC register is 1 (low voltage detection interrupt enabled).

This interrupt shares the interrupt vector with the watchdog timer interrupt and oscillator stop detection interrupt. When using the low voltage detection interrupt either or both with the other interrupts at the same time, read the LVDF bit in the LVDC register in the interrupt handler and confirm that the low voltage detection interrupt has been occurred.

The LVDF bit becomes 1 when the input voltage at the VCC pin passes the  $V_{det(R)}$  level or  $V_{det(F)}$  level. When the LVDF bit changes from 0 to 1, a low voltage detection interrupt request is generated. Set this bit to 0 (low voltage undetected) by a program.



### 6.2.3 Application Example of the Low Voltage Detector

Figure 6.7 shows an example of the low voltage detection interrupt.

The supply voltage for internal logic is generated by reducing the input voltage from the VCC pin with the voltage regulators. When the input voltage begins to fall, the internal voltage remains steady. However, as the VCC supply voltage continues to fall, the supply voltage for the internal logic also begins to fall, which may affect MCU operation. Consequently, the system can be safely shut down between when the VCC supply voltage begins to fall and when the supply voltage for internal logic begins to fall. The low voltage detection interrupt can be applied to detect the falling input voltage.

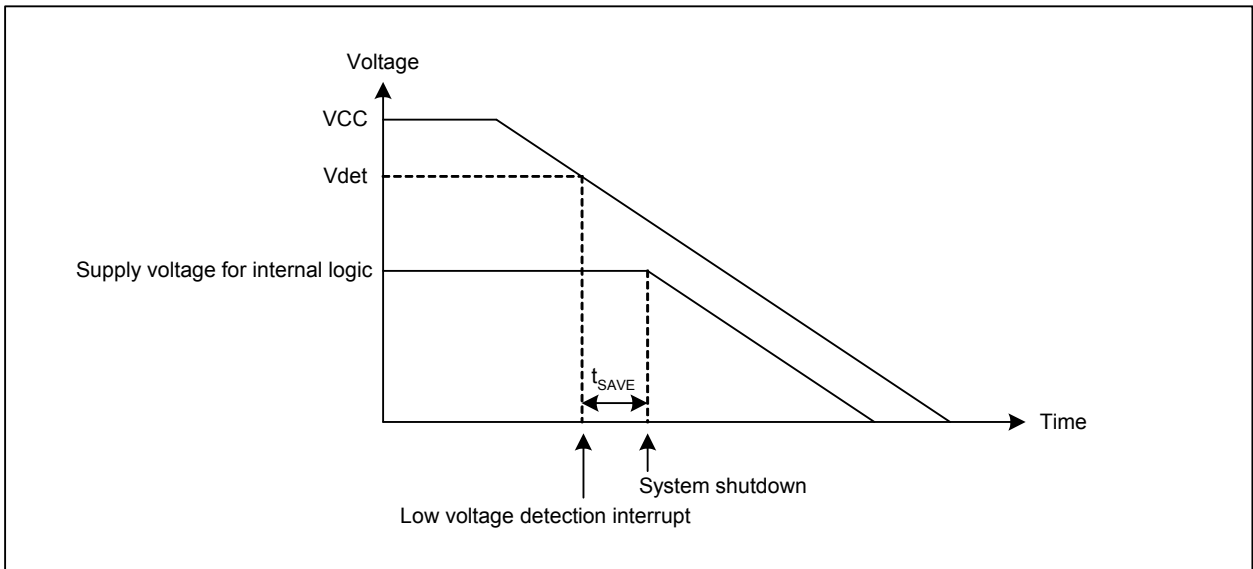


Figure 6.7 Example of the Low Voltage Detection Interrupt

## 7. Clock Generator

### 7.1 Clock Generator Types

The clock generator consists of four circuits:

- Main clock oscillator
- Sub clock oscillator
- PLL frequency synthesizer
- On-chip oscillator (OCO)

Table 7.1 lists the specifications of the clock generator. Figure 7.1 shows a block diagram of the clock generator, and Figures 7.2 to 7.10 show registers associated with clock control.

**Table 7.1 Clock Generator Specifications**

Item	Main Clock Oscillator	Sub Clock Oscillator	PLL Frequency Synthesizer	On-chip Oscillator
Used as	<ul style="list-style-type: none"> <li>• PLL reference clock source</li> <li>• Peripheral clock source</li> </ul>	<ul style="list-style-type: none"> <li>• CPU clock source</li> <li>• Clock source for timers A and B</li> </ul>	<ul style="list-style-type: none"> <li>• CPU clock source</li> <li>• Peripheral clock source</li> </ul>	<ul style="list-style-type: none"> <li>• CPU clock source</li> <li>• Clock source for timers A and B</li> </ul>
Clock frequency	4 to 8 MHz	32.768 kHz	$f_{SO(PLL)}$ or $f_{(PLL)}$	Approx. 125 kHz
Connectable oscillators or additional circuits	Ceramic resonator Crystal oscillator	Crystal oscillator	—	—
Pins for oscillators or additional circuits	XIN, XOUT	XCIN, XCOU	—	—
Oscillator stop/restart function	Available	Available	Available	Available
Oscillator state after a reset	Running	Stopped	Running	Stopped
Note	Externally generated clock can be input	Externally generated clock can be input	When the main clock oscillator stops running, the PLL frequency synthesizer oscillates at its own frequency of $f_{SO(PLL)}$	The on-chip oscillator starts running by setting the CSPM bit in the OFS area to 0 after a reset

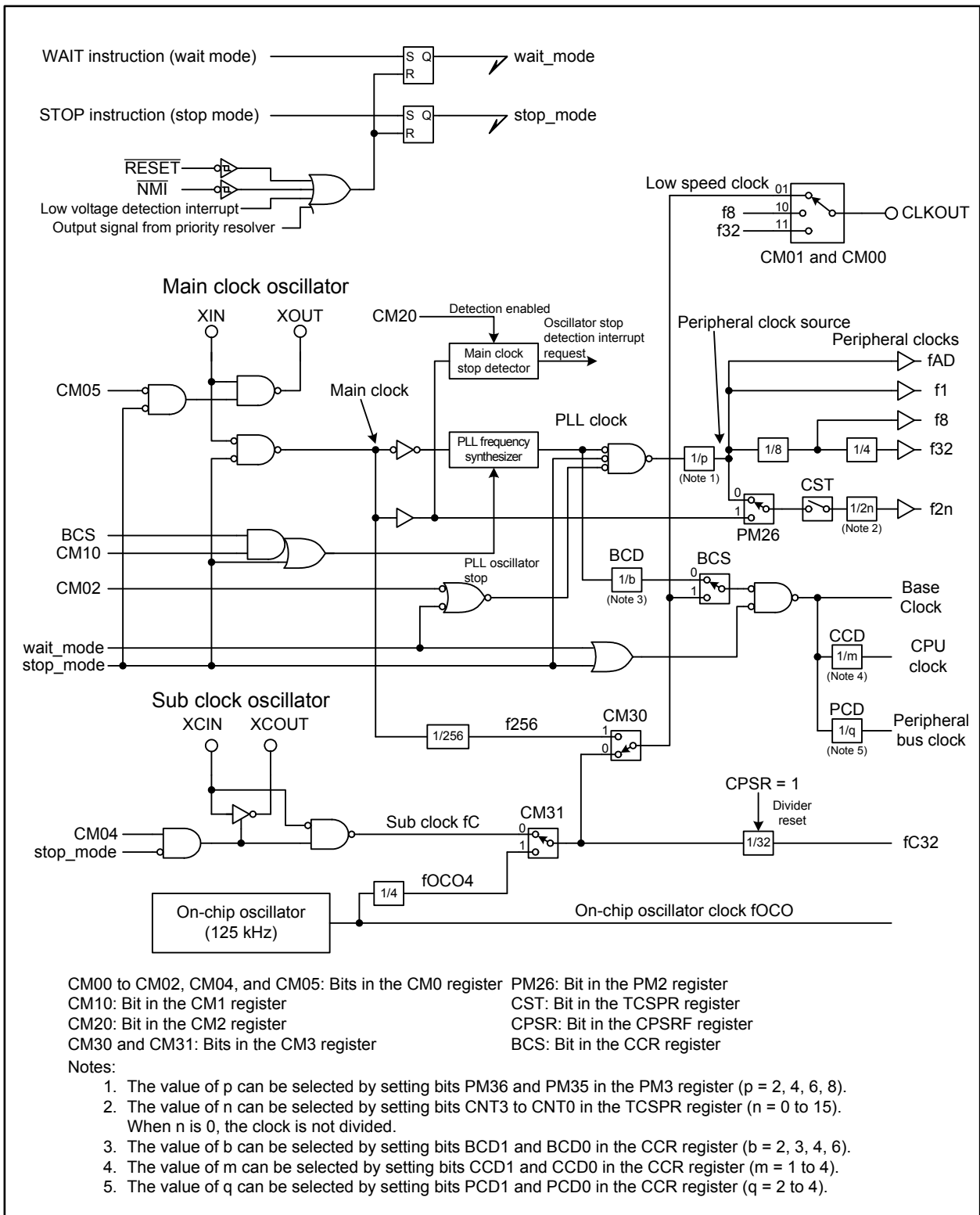
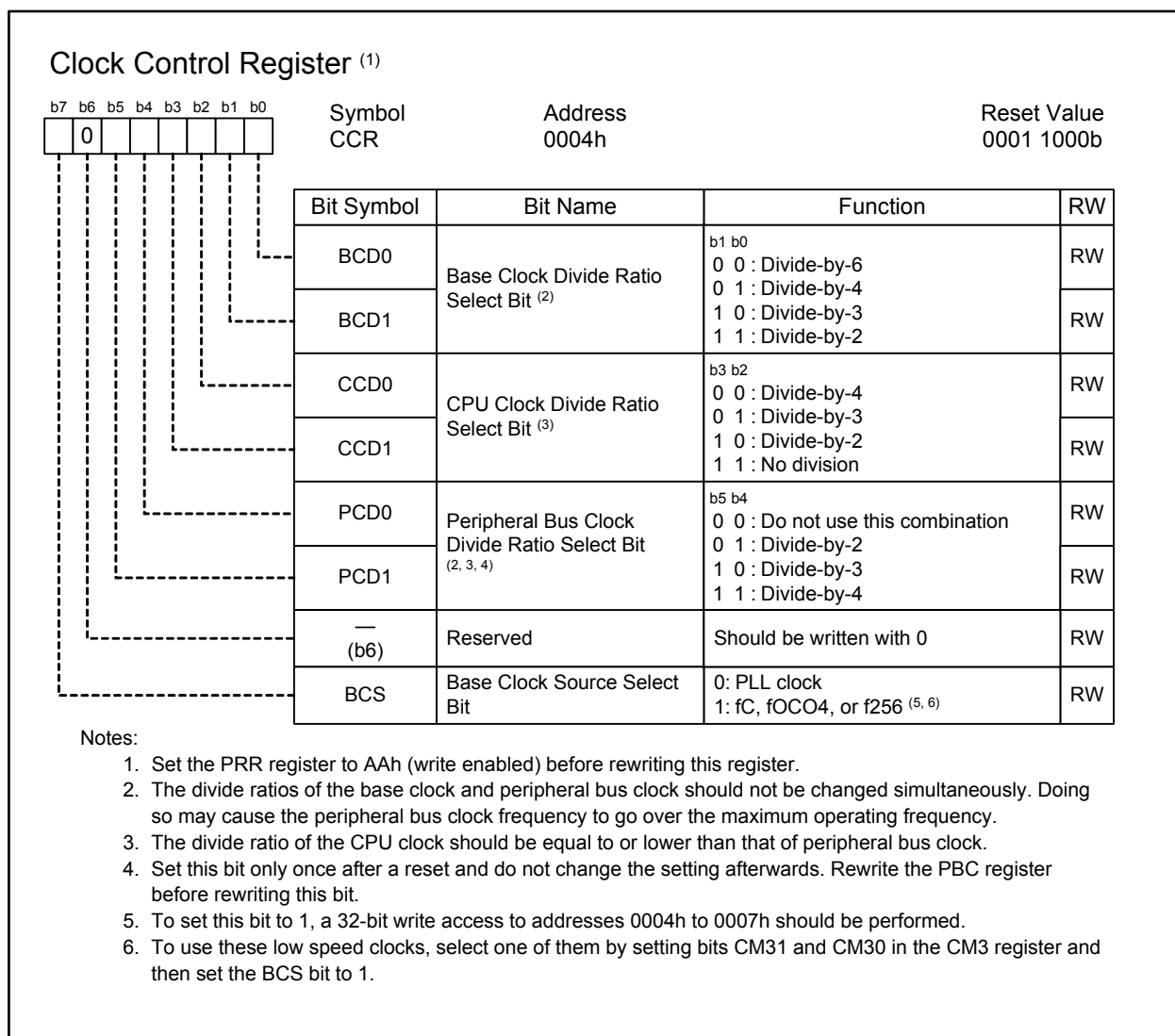
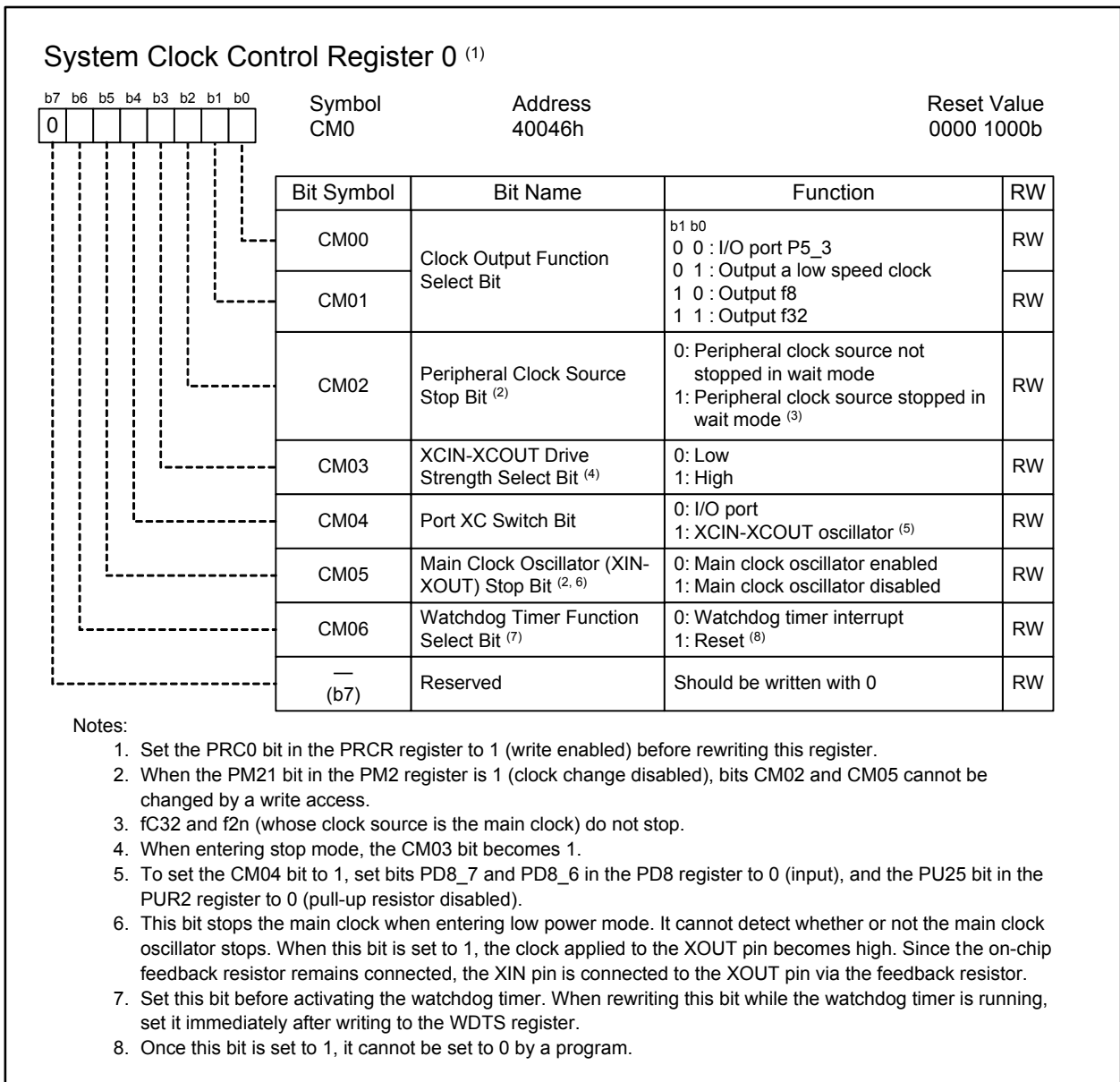


Figure 7.1 Clock Generation Circuitry



**Figure 7.2 CCR Register**



**Figure 7.3 CM0 Register**

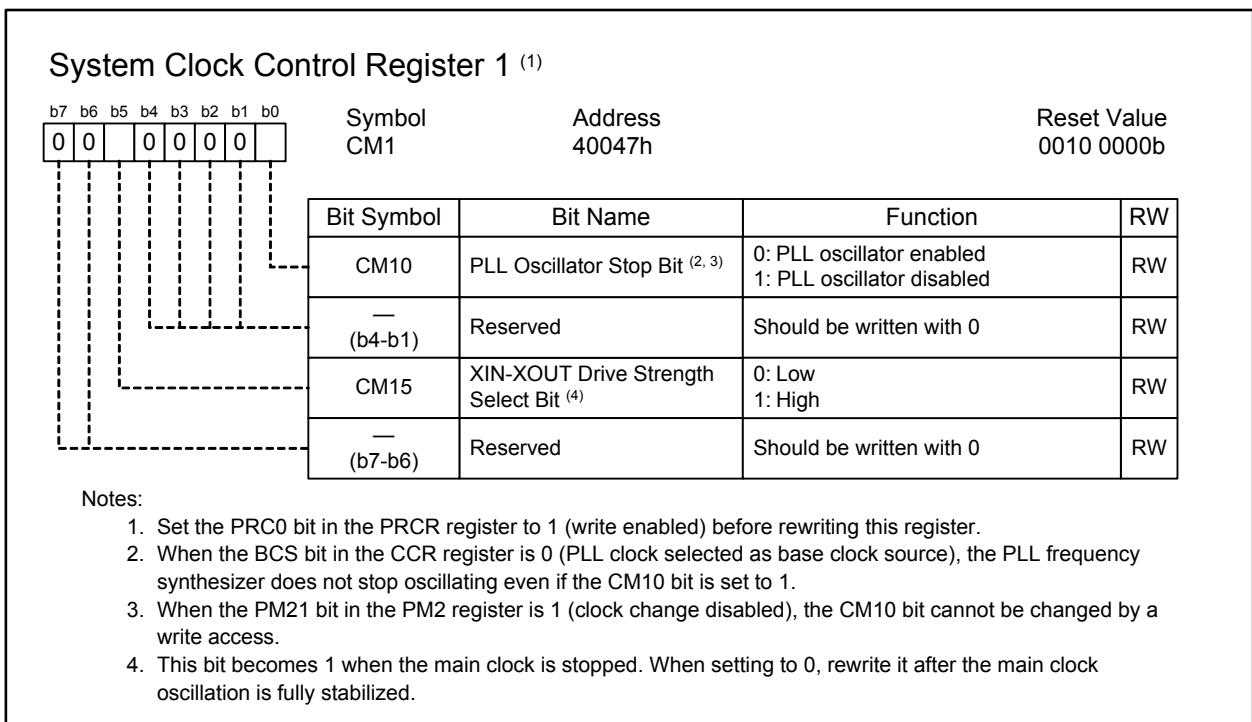


Figure 7.4 CM1 Register

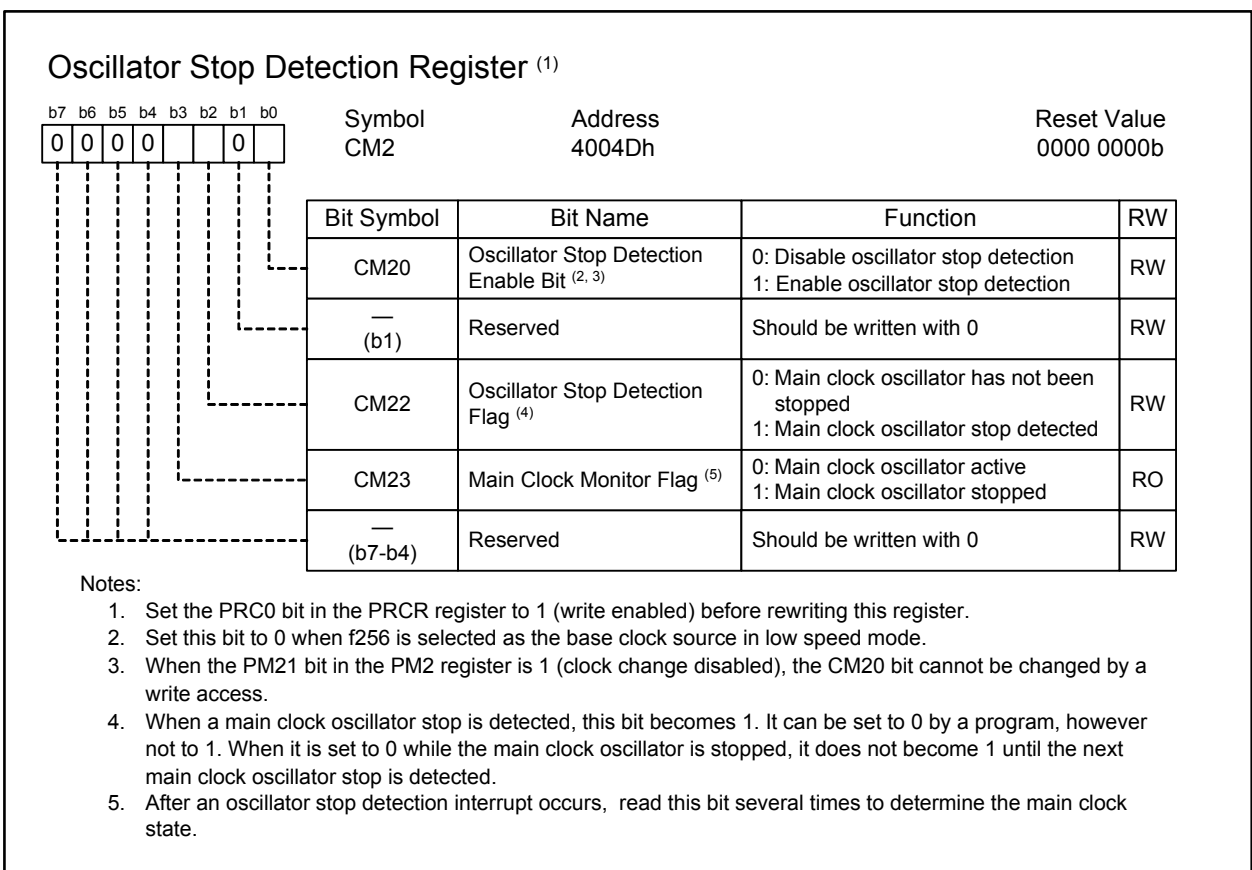
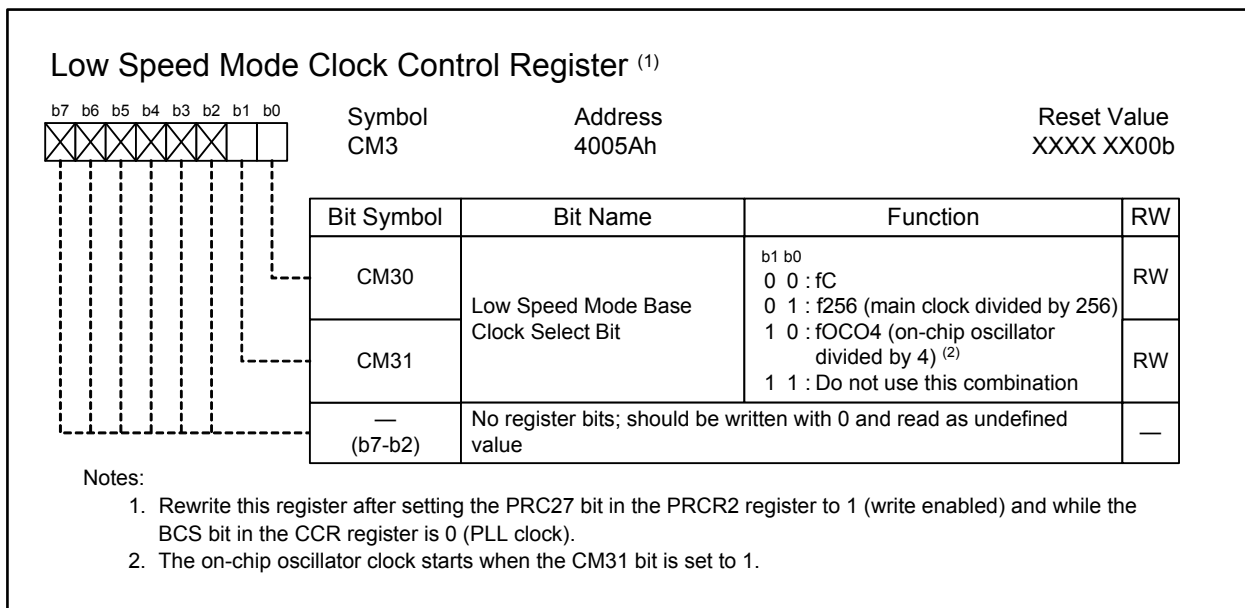
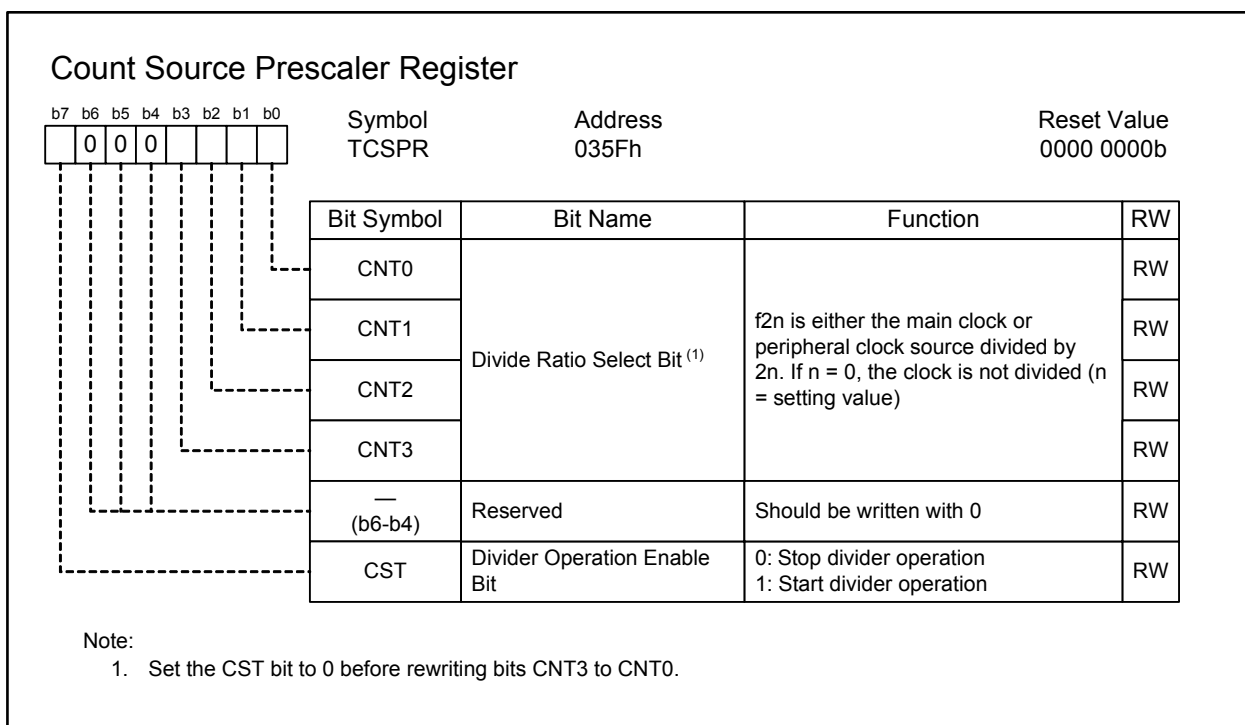


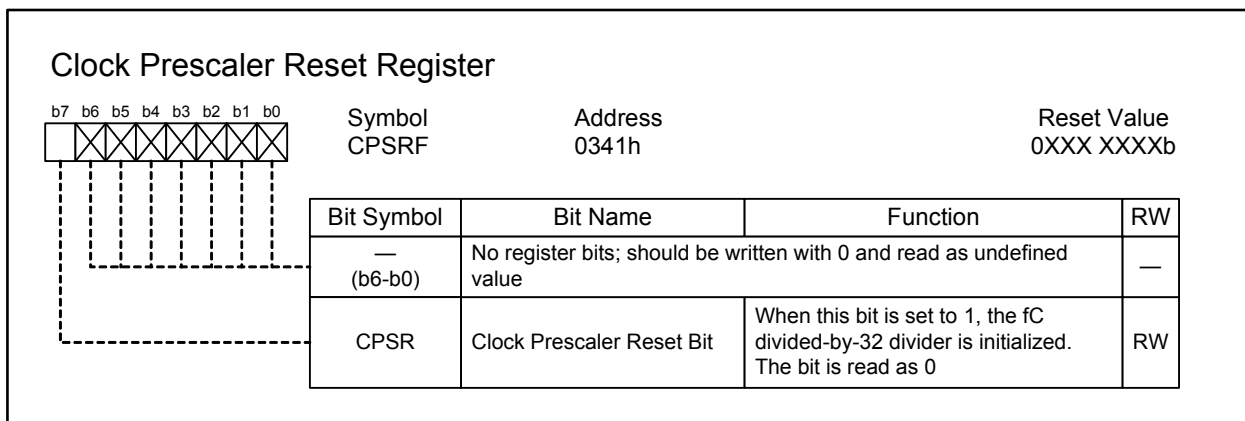
Figure 7.5 CM2 Register



**Figure 7.6 CM3 Register**

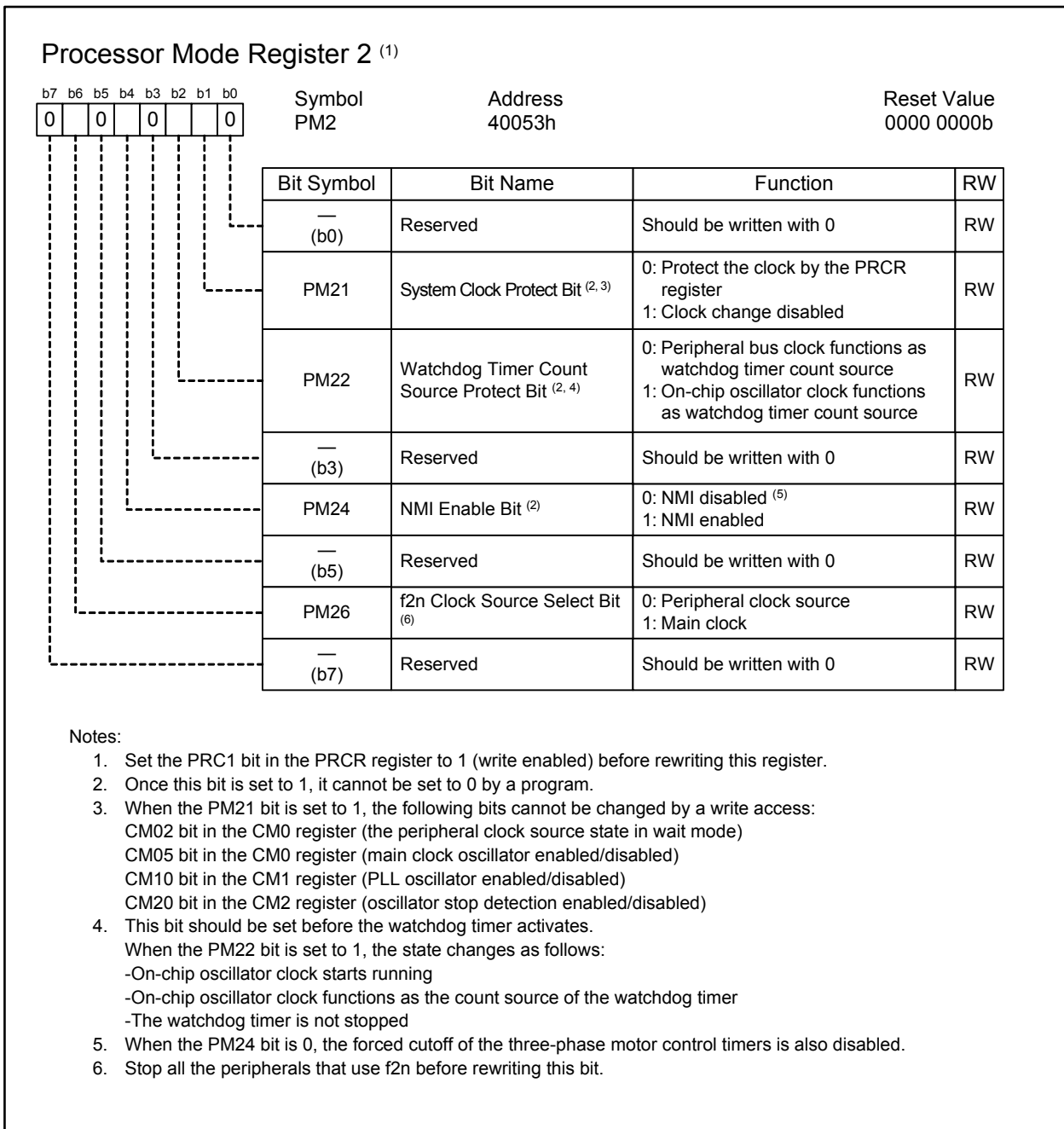


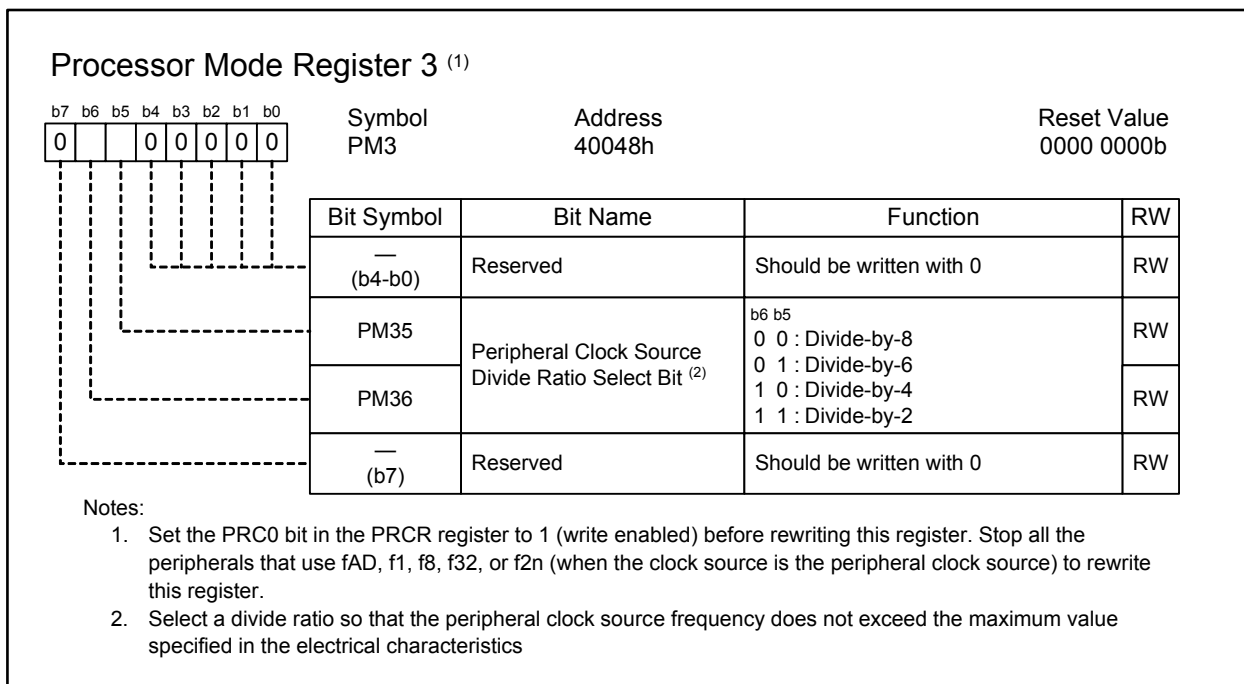
**Figure 7.7 TCSPR Register**



**Figure 7.8 CPSRF Register**



**Figure 7.9 PM2 Register**



**Figure 7.10 PM3 Register**

The following sections illustrate clocks generated in clock generators.

### 7.1.1 Main Clock

The main clock is generated by the main clock oscillator. This clock can be a clock source for the PLL reference clock or peripheral clocks. It also functions as an operating clock for the CAN module.

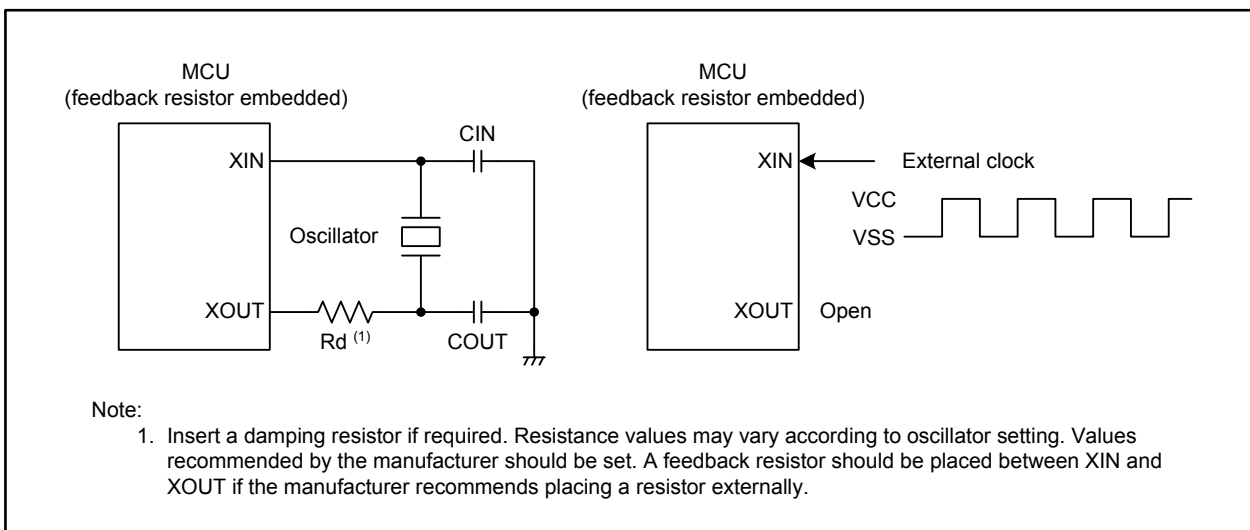
The main clock oscillator is configured with two pins, XIN and XOUT, connected by an oscillator or resonator. The circuit has an on-chip feedback resistor which is separated from the oscillator in stop mode to save power consumption. An external clock can be applied to the XIN pin in this circuit. Figure 7.11 shows an example of a main clock circuit connection.

Circuit constants vary depending on the oscillator. Circuit constants should be set as per the oscillator manufacturer's recommendations.

After a reset, the main clock oscillator is still independently active and disconnected from the PLL frequency synthesizer. A PLL frequency synthesizer self-oscillating clock divided by 12 is provided to the CPU.

Setting the CM05 bit in the CM0 register to 1 (main clock oscillator disabled) enables power-saving. In this case, the clock applied to the XOUT pin becomes high. The XIN pin connected to the XOUT pin by an embedded feedback resistor is also driven high. Do not set the CM05 bit to 1 when an external clock is applied to the XIN pin.

All clocks, including the main clock, stop in stop mode. Refer to 7.7 "Power Control" for details.



**Figure 7.11 Main Clock Circuit Connection**

### 7.1.2 Sub Clock (fC)

The sub clock is generated by the sub clock oscillator. This clock can be a clock source for the CPU clock and a count source for timers A and B. It can be output from the CLKOUT pin.

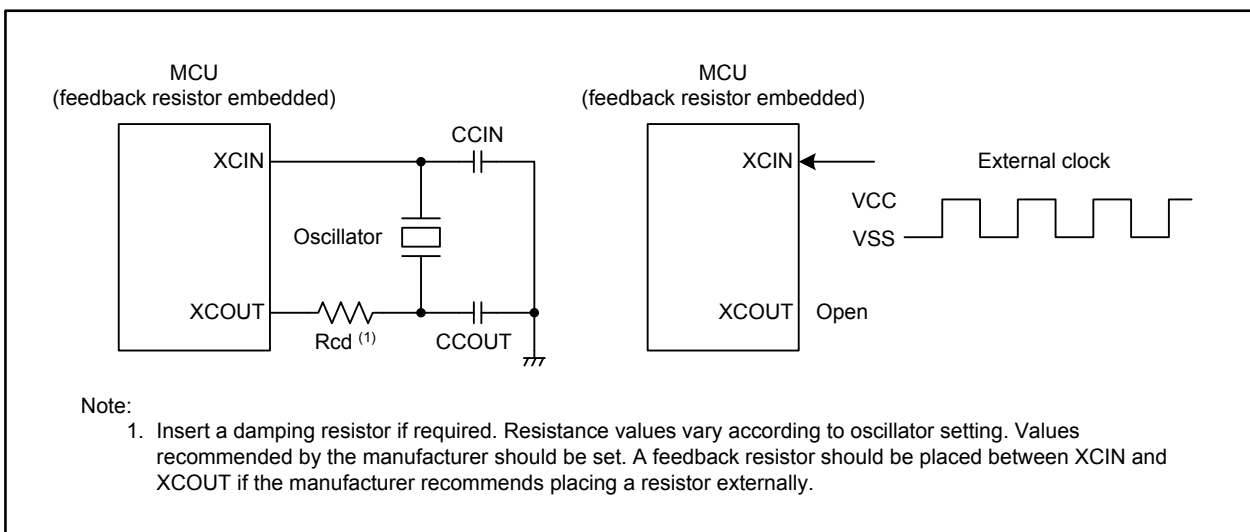
The sub clock oscillator is configured with pins XCIN and XCOU connected by a crystal oscillator. The circuit has a on-chip feedback resistor which is separated from the oscillator in stop mode to save power consumption. An external clock can be applied to the XCIN pin. Figure 7.12 shows an example of a sub clock circuit connection. Circuit constants vary depending on the oscillator. Circuit constants should be set as per the oscillator manufacturer's recommendations.

After a reset, the sub clock is stopped and the feedback resistor is separated from the oscillator. In order to start the sub clock oscillation, first set bits PD8\_6 and PD8\_7 in the PD8 register to 0 (input mode), and the PU25 bit in the PUR2 register to 0 (pull-up resistor disabled). Then, set the CM04 bit in the CM0 register to 1 (XCIN-XCOU oscillator).

To input an external clock to the XCIN pin, set bits PD8\_7 and PU25 to 0 and then the CM04 bit to 1. The clock applied to the XCIN pin becomes a clock source for the sub clock.

When the CM3 register is set to 00h (fC) and the BCS bit in the CCR register is set to 1 (fC, fOCO4, or f256) after the sub clock oscillation has stabilized, the sub clock becomes the base clock of the CPU clock and the peripheral bus clock.

All clocks, including the sub clock, stop in stop mode. Refer to 7.7 "Power Control" for details.



**Figure 7.12 Sub Clock Circuit Connection**

### 7.1.3 PLL Clock

The PLL clock is generated by the PLL frequency synthesizer based on the main clock. This clock can be a clock source for any clock including the CPU clock and the peripheral clock.

Figure 7.13 shows a block diagram of the PLL frequency synthesizer. Figures 7.14 and 7.15 show registers PLC0 and PLC1, respectively. Figure 7.16 shows the PLS register.

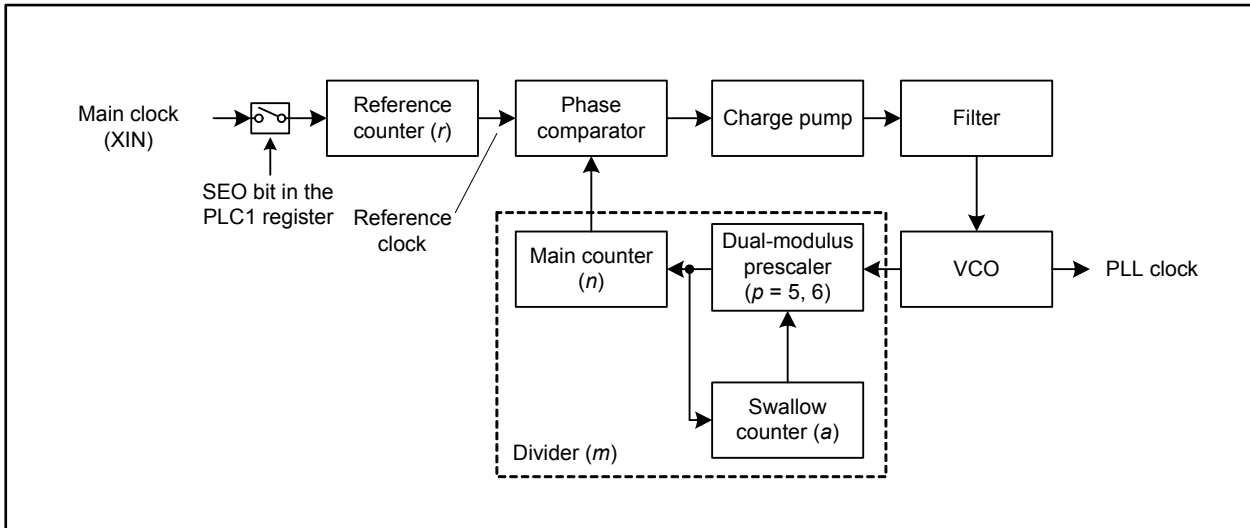


Figure 7.13 PLL Frequency Synthesizer Block Diagram

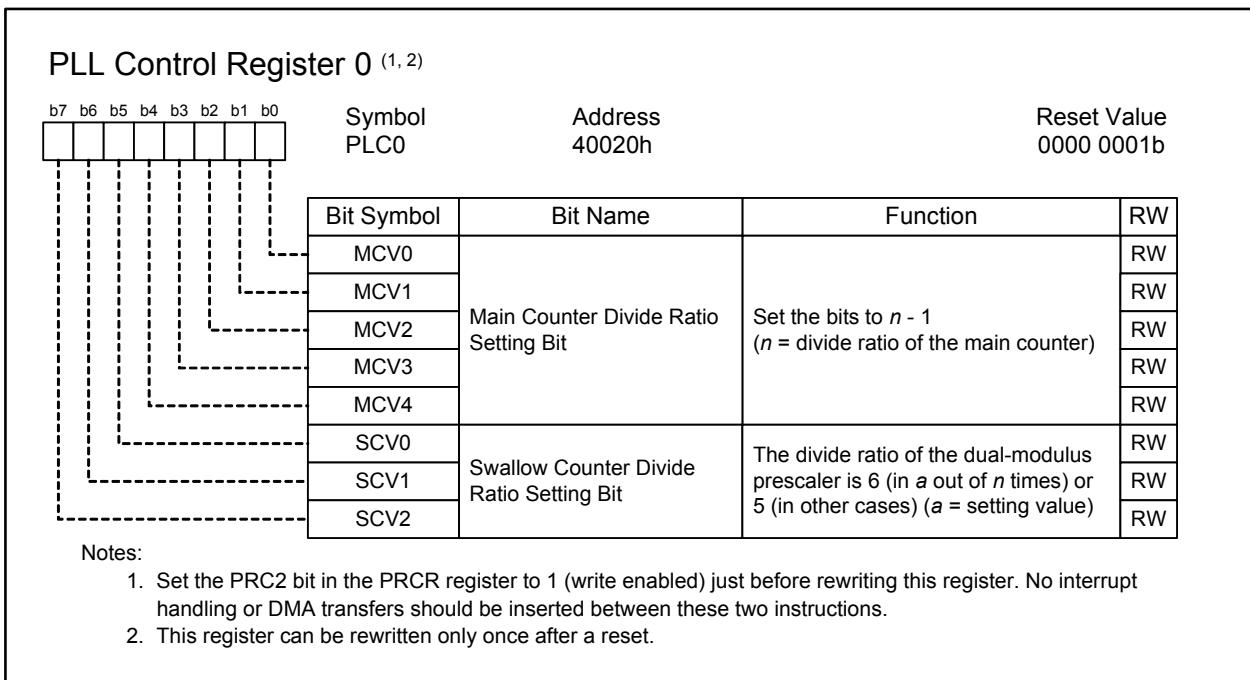


Figure 7.14 PLC0 Register

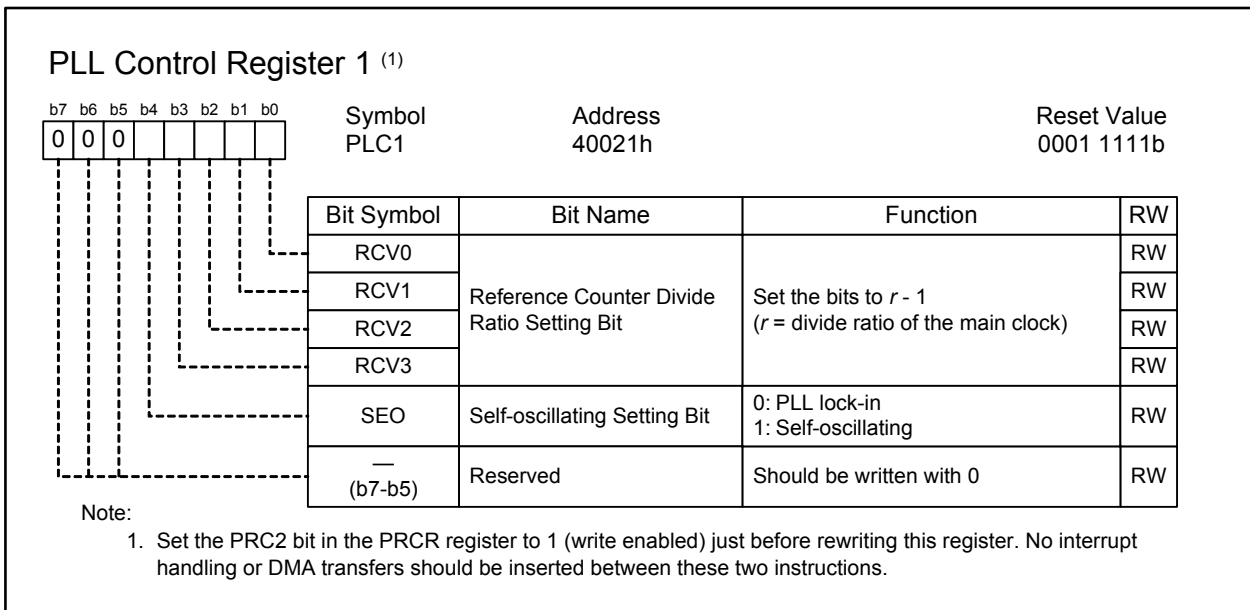


Figure 7.15 PLC1 Register

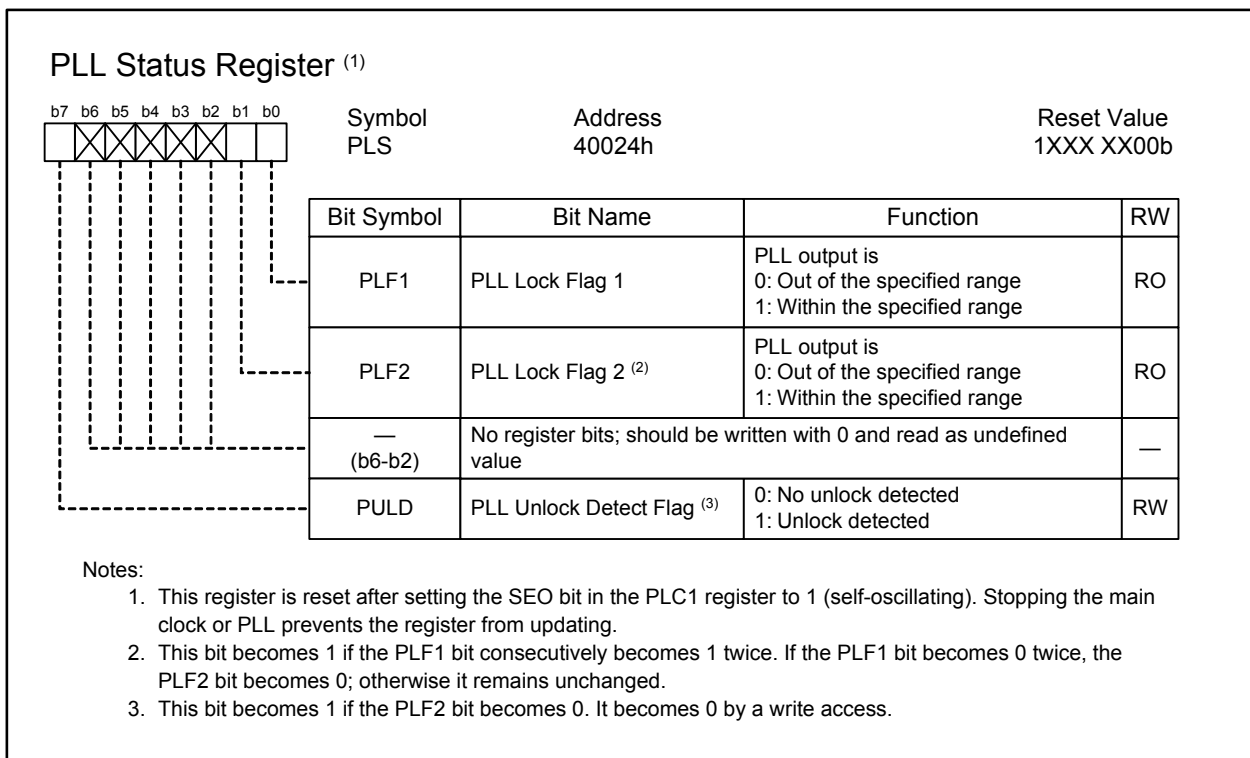


Figure 7.16 PLS Register

In the PLL frequency synthesizer, the pulse-swallow operation is implemented. The divide ratio  $m$  is simply expressed by  $n \times p$ . However, with the swallow counter, the divide ratio  $p$  is 6 in  $a$  out of  $n$ , or 5 in other cases, the actual  $m$  is therefore given by the formula below:

$$\begin{aligned} m &= n \times p \\ &= n \times \left( \frac{a}{n} \cdot 6 + \frac{n-a}{n} \cdot 5 \right) \\ &= 5n + a \end{aligned}$$

The setting range of  $a$  is  $0 \leq a < 5$ ,  $0 \leq a \leq n$ .

As  $r$  is the divide ratio of the reference counter, the PLL clock has a  $m/r$  times the main clock (XIN) frequency.

$$\begin{aligned} \text{PLL clock frequency } f(PLL) &= \frac{m}{r} \cdot \text{main clock frequency} \\ &= \frac{5n + a}{r} \cdot \text{main clock frequency} \end{aligned}$$

After a reset, the reference counter is divided by 16, and the PLL frequency synthesizer is multiplied by 10. Since the main clock as a reference clock is disconnected, the PLL frequency synthesizer may self-oscillate at its own frequency of  $f_{\text{SO(PLL)}}$ .

Each register should be set to meet the following conditions:

- The reference clock, which is the main clock divided by  $r$ , should be between 2 to 4 MHz.
- The divide ratio  $m$  is  $25 \leq m \leq 100$ .

For the setting of registers PLC1 and PLC0, Table 7.2 should be applied. While the main clock oscillation is stable, a wait time of  $t_{\text{LOCK(PLL)}}$  is necessary between rewriting registers PLC1 and PLC0, and the PLL clock becoming stable.

**Table 7.2 PLC1 and PLC0 Register Settings (1)**

Main Clock	$r$	Reference Clock	$n$	$a$	$m$	PLC1 Register Setting	PLC0 Register Setting	$m/r$	PLL Clock
4 MHz	2	2 MHz	9	3	48	01h	68h	24	96 MHz
6 MHz	2	3 MHz	6	2	32	01h	45h	16	96 MHz
8 MHz	3	2.6667 MHz	7	1	36	02h	26h	12	96 MHz
4 MHz	1	4 MHz	6	0	30	00h	05h	30	120 MHz
5 MHz	2	2.5 MHz	9	3	48	01h	68h	24	120 MHz
6 MHz	2	3 MHz	8	0	40	01h	07h	20	120 MHz
8 MHz	2	4 MHz	6	0	30	01h	05h	15	120 MHz
4 MHz	1	4 MHz	6	2	32	00h	45h	32	128 MHz
6 MHz	3	2 MHz	12	4	64	02h	8Bh	21.3333	128 MHz
8 MHz	2	4 MHz	6	2	32	01h	45h	16	128 MHz

Note:

1. Registers PLC1 and PLC0 should be set according to the list above.

#### 7.1.4 On-chip Oscillator Clock

The on-chip oscillator clock is generated by the on-chip oscillator (OCO). This clock can be a clock source for the CPU clock and a count source for timers A and B. This clock has a frequency of approximately 125 kHz. The on-chip oscillator clock divided by 4 can be used as the base clock for the CPU clock and peripheral bus clock.

When the CSPM bit in the OFS area is 1, the on-chip oscillator clock is stopped after a reset. It starts running if the CM31 bit in the CM3 register or the PM22 bit in the PM2 register is set to 1. It is not necessary to wait for stabilization because the on-chip oscillator instantly starts oscillating.



## 7.2 Oscillator Stop Detection

This function detects the main clock is stopped when its oscillator stops running due to an external factor. When the CM20 bit in the CM2 register is 1 (enable oscillator stop detection), an oscillator stop detection interrupt request is generated as soon as the main clock stops. Simultaneously, the PLL frequency synthesizer starts to self-oscillate at its own frequency. If the PLL frequency synthesizer is the clock source for CPU clock and peripheral clock, these clocks continue running.

When an oscillator stop is detected, the following bits in the CM2 register become 1:

- The CM22 bit: main clock oscillator stop detected
- The CM23 bit: main clock oscillator stopped

### 7.2.1 How to Use Oscillator Stop Detection

The oscillator stop detection interrupt shares vectors with the watchdog timer interrupt and the low voltage detection interrupt. When using these interrupts simultaneously, read the CM22 bit with an interrupt handler to determine if an oscillator stop detection interrupt request has been generated.

When the main clock oscillator resumes running after an oscillator stop is detected, the PLL clock frequency may temporarily exceed the preset value before the PLL frequency synthesizer oscillation stabilizes. As soon as an oscillator stop is detected, the main clock oscillator should be stopped from resuming (set the CM05 bit in the CM0 register to 1) or the divide ratios of the base clock and peripheral clock source should be increased by a program. They can be set using bits BCD1 and BCD0 in the CCR register and bits PM36 and PM35 in the PM3 register.

In low speed mode, when the main clock oscillator stops running, an oscillator stop detection interrupt request is generated if the CM20 bit is set to 1 (enable oscillator stop detection). The CPU clock remains running with a low speed clock source. Note that if the base clock is f256, which is the main clock divided by 256, oscillator stop detection cannot be used.

The oscillator stop detection is provided to handle main clock stop caused by external factors. To stop the main clock oscillator by a program, i.e., to enter stop mode or to set the CM05 bit to 1 (main clock oscillator disabled), the CM20 bit in the CM2 register should be set to 0 (disable oscillator stop detection). To enter wait mode, this bit should be also set to 0.

The oscillator stop detection functions depending on the voltage of a capacitor which is being changed. In more concrete terms, this function detects that the oscillator is stopped when the main clock goes lower than approximately 500 kHz. Note that if the CM22 bit is set to 0 by a program in an interrupt handler while the frequency is around 500 kHz, a stack overflow may occur due to multiple interrupt requests.

## 7.3 Base Clock

The base clock is a reference clock for the CPU clock and peripheral bus clock. The base clock after a reset is the PLL clock divided by 6.

The base clock source is selected between the PLL clock and the low speed clocks which contain the sub clock (fC), on-chip oscillator clock divided by 4 (fOCO4), and main clock divided by 256 (f256).

If the PLL clock is selected, it is divided by 2, 3, 4, or 6 to become the base clock. If a low speed clock is selected, the clock itself can be the base clock.

The base clock source is set using the BCS bit in the CCR register and the divide ratio for the PLL clock is set using bits BCD1 and BCD0. Bits CM31 and CM30 in the CM3 register select a low speed clock.

## 7.4 CPU Clock and Peripheral Bus Clock

The CPU operating clock is referred to as the CPU clock. The CPU clock after a reset is the base clock divided by 2.

The CPU clock source is the base clock, and its divide ratio is selected by setting bits CCD1 and CCD0 in the CCR register. The base clock divided by 2 to 4 becomes the peripheral bus clock. Its divide ratio is selected by setting bits PCD1 and PCD0 in the CCR register. The peripheral bus clock also functions as a count source for the watchdog timer and operating clock for the serial bus interface and the CAN module. To prevent the CPU clock, whose clock source is the PLL clock, from stopping when the CPU becomes out of control, set the following while the CM05 bit in the CM0 register is 0 (main clock oscillator enabled) and the BCS bit in the CCR register is 0 (PLL clock selected as base clock source):

- (1) Set the PRC1 bit in the PRCR register to 1 (write enabled to the PM2 register).
- (2) Set the PM21 bit in the PM2 register to 1 (clock change disabled).

## 7.5 Peripheral Clock

The peripheral clock is an operating clock or a count source for the peripherals excluding the watchdog timer, the serial bus interface, and the CAN module. The source of this clock is generated by a clock, which has the same frequency as the PLL clock, divided by 2, 4, 6, or 8 according to the settings of bits PM36 and PM35 in the PM3 register. The peripheral clock is classified into three types of clock as follows:

### (1) f1, f8, f32, f2n

f1, f8, and f32 are the peripheral clock sources divided by 1, 8, and 32, respectively. The clock source for f2n is selected between the peripheral clock source and the main clock by setting the PM26 bit in the PM2 register. The f2n divide ratio can be set using bits CNT3 to CNT0 in the TCSPR register (n = 1 to 15, not divided when n = 0).

f1, f8, f32, and f2n, whose clock source is the peripheral clock source, stop in low power mode or when the CM02 bit is set to 1 (peripheral clock source stopped in wait mode) to enter wait mode.

f1, f8, and f2n are used as a count source for timers A and B or an operating clock for the serial interface and the LIN module. f1 is used as an operating clock for the intelligent I/O as well. The f32 is used as an operating clock for the LIN module as well.

f8 and f32 can be output from the CLKOUT pin. Refer to 7.6 "Clock Output Function" for details.

### (2) fAD

fAD, which has the same frequency as peripheral clock source, is an operating clock for the A/D converter.

This clock stops in low power mode or when the CM02 bit is set to 1 (peripheral clock source stopped in wait mode) to enter wait mode.

### (3) fC32

fC32, which is a sub clock divided by 32, or on-chip oscillator clock divided by 128, is used as the count source for timers A and B. This clock is available when the sub clock or on-chip oscillator clock is active.

## 7.6 Clock Output Function

Low speed clocks, f8, and f32 can be output from the CLKOUT pin.

Table 7.3 lists the CLKOUT pin functions.

**Table 7.3 CLKOUT Pin Functions**

CM0 Register (1)		CLKOUT Pin Function
CM01	CM00	
0	0	I/O port P5_3
0	1	Output a low speed clock
1	0	Output f8
1	1	Output f32

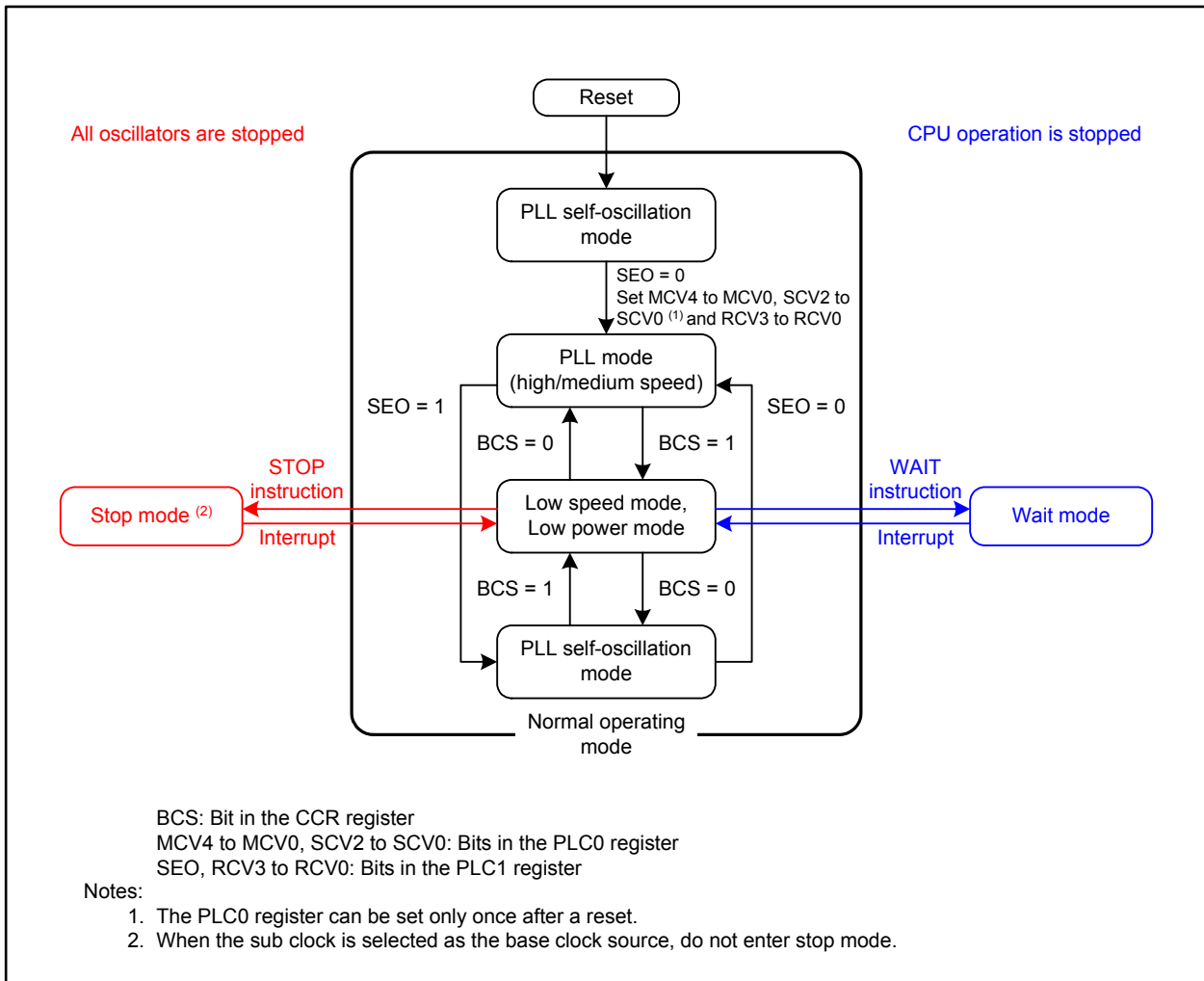
Note:

1. Set the PRC0 bit in the PRCR register 1 (write enabled) before rewriting this register.

## 7.7 Power Control

Power control has three modes: wait mode, stop mode, and normal operating mode.

The name “normal operating mode” is used restrictively in this chapter, and it indicates all other modes except wait mode and stop mode. Figure 7.17 shows a block diagram of the state transition in normal operating mode, stop mode, and wait mode.



**Figure 7.17 State Transition in Stop Mode and Wait Mode**

### 7.7.1 Normal Operating Mode

Normal operating mode is classified into the five modes shown below.

In normal operating mode, the CPU clock and peripheral clock are provided to operate the CPU and peripherals. Power consumption is controlled by the CPU clock frequency. The higher the CPU clock frequency is, the more processing power increases. The lower the CPU clock frequency is, the less power consumption is required. Power consumption can be reduced by stopping oscillators that are not being used.

#### (1) PLL Mode (high speed mode)

In this mode, the PLL clock is selected as the base clock source, and the main clock is provided as the reference clock source for the PLL frequency synthesizer. High speed mode enables the CPU to operate at the maximum operating frequency. The PLL clock divided by 2 becomes the base clock. The base clock frequency should be identical to that of the CPU clock.  $f_{AD}$ ,  $f_1$ ,  $f_8$ ,  $f_{32}$ , and  $f_{2n}$  can be used as the peripheral clocks. When the sub clock or the on-chip oscillator clock is provided,  $f_{C32}$  can be used as the count source for timers A and B.

#### (2) PLL Mode (medium speed mode)

This mode indicates all modes in PLL mode except high speed mode. The PLL clock divided by 2, 3, 4, or 6 becomes the base clock and the base clock divided by 1 to 4 becomes the CPU clock.  $f_{AD}$ ,  $f_1$ ,  $f_8$ ,  $f_{32}$ , and  $f_{2n}$  can be used as the peripheral clocks. When the sub clock or the on-chip oscillator clock is provided,  $f_{C32}$  can be used as the count source for timers A and B.

#### (3) Low Speed Mode

In this mode, a low speed clock is used as the base clock source. The low speed clock becomes the base clock and the base clock divided by 1 to 4 becomes the CPU clock.  $f_{AD}$ ,  $f_1$ ,  $f_8$ ,  $f_{32}$ , and  $f_{2n}$  can be used as the peripheral clocks. When the sub clock or the on-chip oscillator clock is provided,  $f_{C32}$  can be used as the count source for timers A and B.

#### (4) Low Power Mode

This is a state where the main clock oscillator and the PLL frequency synthesizer are stopped after switching to low speed mode. The sub clock or the on-chip oscillator clock divided by 4 becomes the base clock and the base clock divided by 1 to 4 becomes the CPU clock.  $f_{C32}$ , which is the only peripheral clock available, can be used as the count source for timers A and B. By setting the MRS bit in the VRCCR register to 1 (main regulator stopped), this mode consumes even less power than the modes above.

#### (5) PLL Self-oscillation Mode

In this mode, the PLL clock is selected as the base clock source, and the main clock is not provided as the reference clock source for the PLL frequency synthesizer. The PLL frequency synthesizer self-oscillates at its own frequency. The PLL clock divided by 2, 3, 4, or 6 becomes the base clock and the base clock divided by 1 to 4 becomes the CPU clock.  $f_{AD}$ ,  $f_1$ ,  $f_8$ ,  $f_{32}$ , and  $f_{2n}$  can be used as the peripheral clocks. When the sub clock or the on-chip oscillator clock is provided,  $f_{C32}$  can be used as the count source for timers A and B.

The state transition within normal operating mode can be very complicated; therefore only the block diagrams of typical state transitions are shown. Figures 7.18 to 7.20 show block diagrams of the respective state transitions: state when the sub clock is used, state when the main clock divided by 256 is used, and state when the on-chip oscillator clock is used. As for the state transitions other than the above, setting of each register and the usage notes below can be used as references.

- PLL can be switched from PLL oscillating to self-oscillating by setting the SEO bit in the PLC1 register to 1. Set the SEO bit to 1 (self-oscillating) before setting the CM05 bit in the CM0 register to 0 (main clock oscillator disabled) to stop the main clock.
- The divide ratio of the clock should be increased and the frequency should be decreased by using bits BCD1 to BCD0 in the CCR register or bits PM36 to PM35 in the PM3 register before setting the SEO bit to 0 (PLL oscillating) in order to switch back PLL self-oscillation mode to PLL mode. Set back the settings of bits BCD1 to BCD0 and bits PM36 to PM35 once PLL oscillation is stabilized after setting the SEO bit to 0.
- Before switching the CPU clock to another clock, that clock should be stabilized. In particular, the sub clock oscillator may require more time to stabilize <sup>(1)</sup>. Therefore, certain waiting time to switch should be taken by a program immediately after turning the MCU on or exiting stop mode.

Note:

1. Contact the oscillator manufacturer for details on oscillator stabilization time.

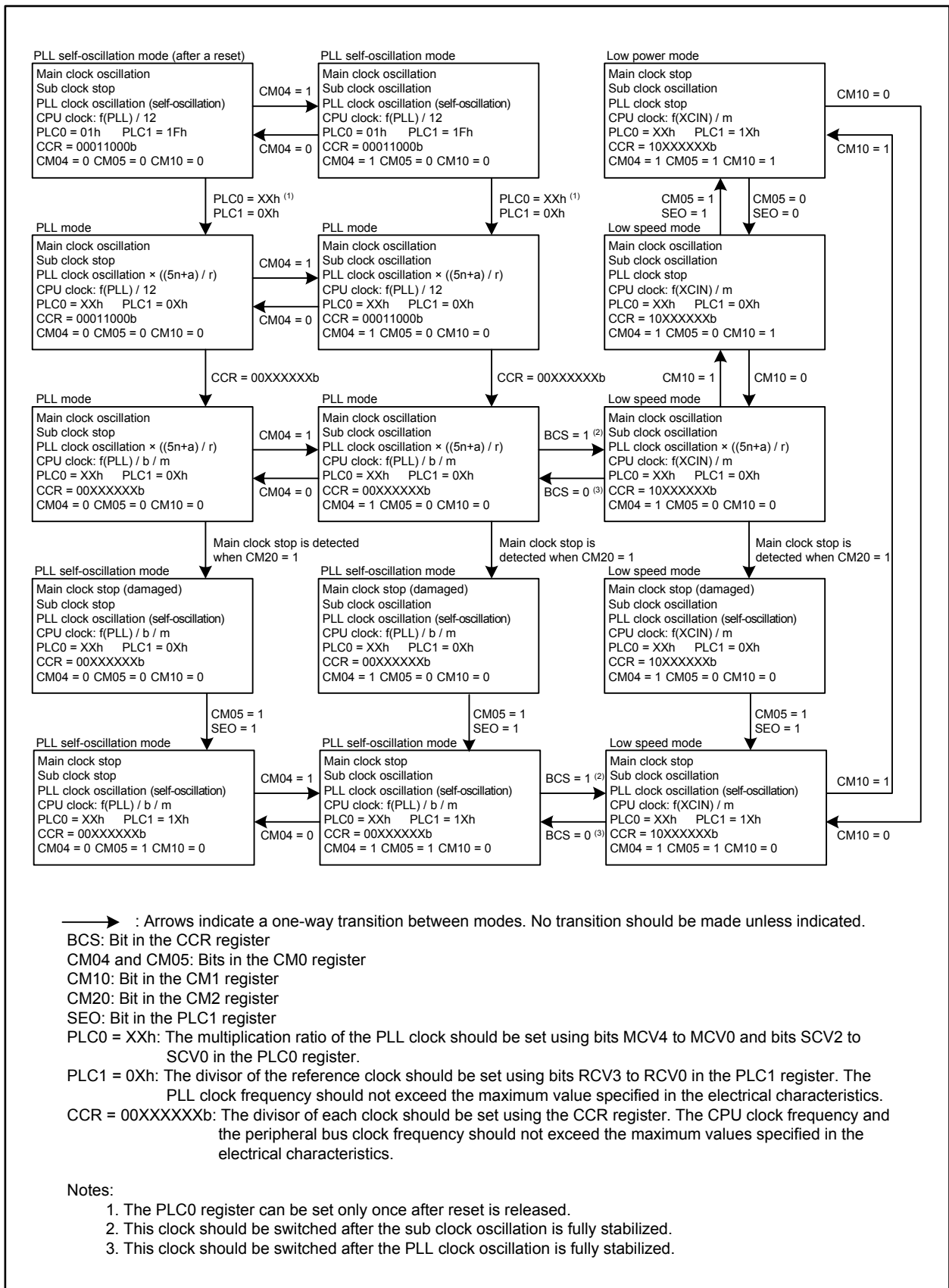


Figure 7.18 State Transition When Using the Sub Clock

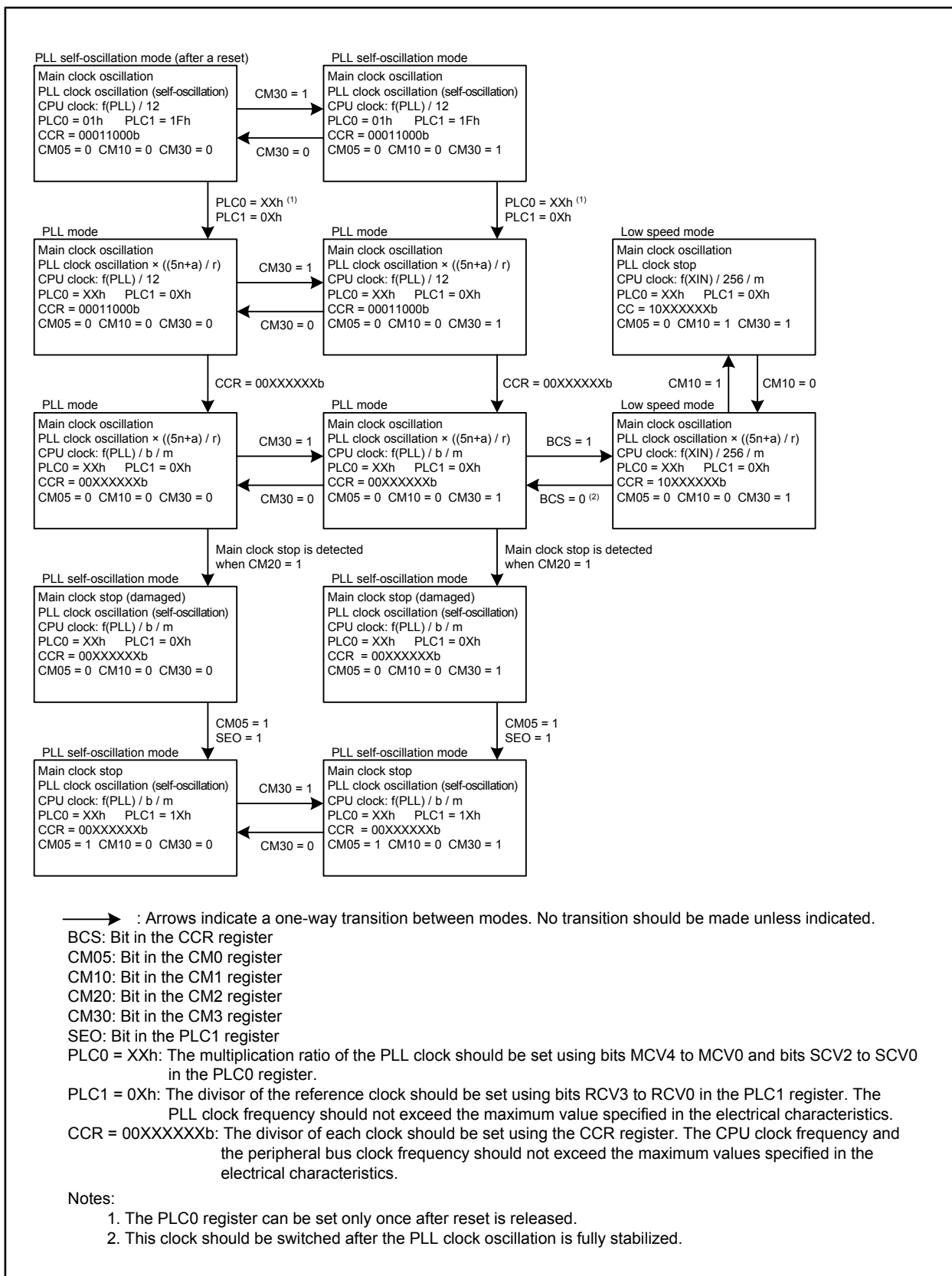


Figure 7.19 State Transition When Using the Main Clock Divided by 256



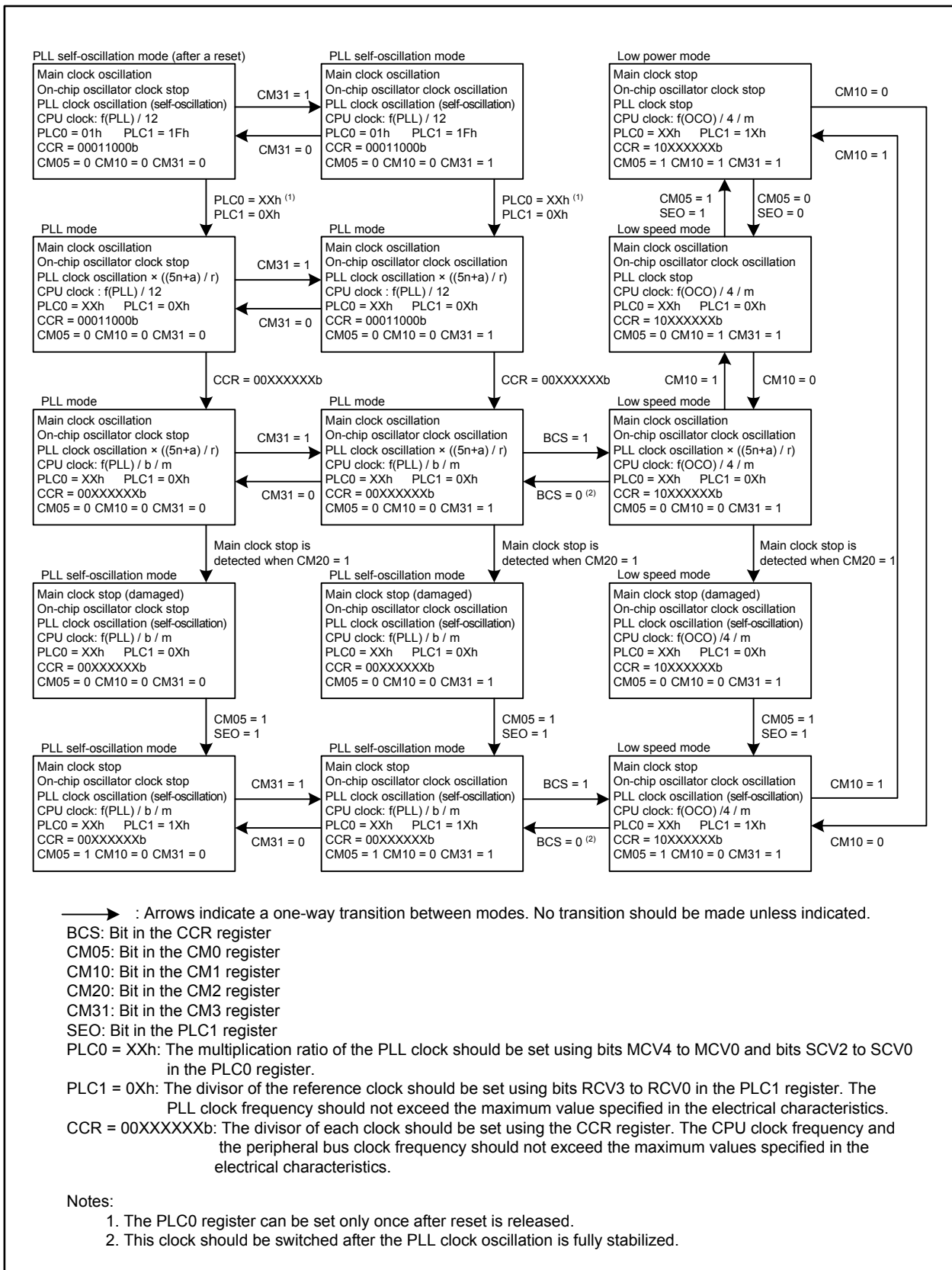


Figure 7.20 State Transition When Using the On-chip Oscillator Clock

## 7.7.2 Wait Mode

The base clock stops in wait mode so that clocks generated by the base clock, the CPU clock and peripheral bus clock, stop running as well. Thus the CPU and watchdog timer, operated by these two clocks, also stop. However, the watchdog timer continues operating when the PM22 bit in the PM2 register is 1 (on-chip oscillator selected as count source for the watchdog timer). Since the main clock, sub clock, PLL clock, and on-chip oscillator clock continue running, the peripherals using these clocks also continue operating.

### 7.7.2.1 Peripheral Clock Source Stop Function

When the CM02 bit in the CM0 register is 1 (peripheral clock source stopped in wait mode), power consumption is reduced since peripheral clocks f1, f8, f32, f2n (when the clock source is the peripheral clock source), and fAD stop running in wait mode. fC32 and f2n (when the clock source is the main clock) do not stop running.

### 7.7.2.2 Entering Wait Mode

To enter wait mode, the following procedures should be completed before the WAIT instruction is executed.

- Initial setting
  - Set the wake-up interrupt priority level (bits RLVL2 to RLVL0 in registers RIPL1 and RIPL2) to 7. Then set each interrupt request level.
- Steps before entering wait mode
  - (1) Set the I flag to 0.
  - (2) Set the interrupt request level for each interrupt source (interrupt number from 1 to 127) to 0, if its interrupt request level is not 0.
  - (3) Perform a dummy read of any of the interrupt control registers.
  - (4) Set the processor interrupt priority level (IPL) in the flag register to 0.
  - (5) Enable interrupts temporarily by executing the following instructions:
 

```
FSET I
NOP
NOP
FCLR I
```
  - (6) Set the interrupt request level for the interrupt to exit wait mode. Do not rewrite the interrupt control register after this step.
  - (7) Set the IPL in the flag register.
  - (8) Set the interrupt priority level for resuming to the same level as the IPL.
 

Interrupt request level for the interrupt to exit wait mode > IPL = Interrupt priority level for resuming
  - (9) Set the CM20 bit in the CM2 register to 0 (disable oscillator stop detection) when the oscillator stop detection is used.
  - (10) Enter either low speed mode or low power mode.
  - (11) Set the I flag to 1.
  - (12) Execute the WAIT instruction.
- After exiting wait mode
  - Set the wake-up interrupt priority level to 7 immediately after exiting wait mode.

### 7.7.2.3 Pin State in Wait Mode

Table 7.4 lists the pin state in wait mode.

**Table 7.4 Pin State in Wait Mode**

Pin		State in Wait Mode
Ports		The state immediately before entering wait mode is held
CLKOUT	When a low speed clock is selected	The clock is output
	When f8 or f32 is selected	The clock is output when the CM02 bit in the CM0 register is 0 (no peripheral clock source stopped in wait mode). The state immediately before entering wait mode is held when the CM02 bit is 1 (peripheral clock source stopped in wait mode)

### 7.7.2.4 Exiting Wait Mode

The MCU exits wait mode by a hardware reset, an NMI, or a peripheral interrupt assigned to software interrupt number from 0 to 63.

To exit wait mode using either a hardware reset or NMI, without using peripheral interrupts, set bits ILVL2 to ILVL0 for the peripheral interrupts to 000b (interrupt disabled) before executing the WAIT instruction.

The CM02 bit setting in the CM0 register affects the peripheral interrupts. When the CM02 bit is 0 (peripheral clock source not stopped in wait mode), peripheral interrupts for software interrupt numbers from 0 to 63 can be used to exit wait mode. When this bit is 1 (peripheral clock source stopped in wait mode), peripherals operated using clocks (f1, f8, f32, f2n whose clock source is the peripheral clock source, and fAD) generated by the peripheral clock source stop operating. Therefore, the peripheral interrupts cannot be used to exit wait mode. However, peripherals operated using clocks which are independent from the peripheral clock source (fC32, external clock, and f2n whose clock source is the main clock) do not stop operating. Thus, interrupts generated by these peripherals and assigned to software interrupt numbers from 0 to 63 can be used to exit wait mode.

The CPU clock used when exiting wait mode by a peripheral interrupt or an NMI is the same clock used when the WAIT instruction is executed.

Table 7.5 lists interrupts used to exit wait mode and usage conditions.

**Table 7.5 Interrupts for Exiting Wait Mode and Usage Conditions**

Interrupt	When the CM02 Bit is 0	When the CM02 Bit is 1
NMI	Available	Available
External interrupt	Available	Available
Low voltage detection interrupt	Available	Available
Watchdog timer interrupt	Should not be used	Should not be used
Timer A interrupt Timer B interrupt	Available in any mode	Available in event counter mode, or when the count source is fC32 or f2n (when the main clock is selected as the clock source)
Serial interface interrupt <sup>(1)</sup>	Available when an internal or external clock is used	Available when the external clock or f2n (when the main clock is selected as the clock source) is used
A/D conversion interrupt	Available in single mode or single-sweep mode	Should not be used
Intelligent I/O interrupt	Available	Should not be used
LIN Low detection interrupt	Available	Available
CAN wake-up interrupt	Available	Available

Note:

1. UART3 and UART4 are excluded.

### 7.7.3 Stop Mode

In stop mode, all of the clocks, except for those that are protected, stop running. That is, the CPU and peripherals, operated by the CPU clock and peripheral clock, also stop. This mode saves the most power.

#### 7.7.3.1 Entering Stop Mode

To enter stop mode, the following procedures should be done before the STOP instruction is executed.

- Initial setting
  - Set the wake-up interrupt priority level (bits RLVL2 to RLVL0 in registers RIPL1 and RIPL2) to 7.
  - Then set each interrupt request level.
- Steps before entering stop mode
  - (1) Set the I flag to 0.
  - (2) Set the interrupt request level for each interrupt source (interrupt number from 1 to 127) to 0, if the interrupt request level is not 0.
  - (3) Perform a dummy read of any of the interrupt control registers.
  - (4) Set the processor interrupt priority level (IPL) in the flag register to 0.
  - (5) Enable interrupts temporarily by executing the following instructions:
 

```
FSET I
NOP
NOP
FCLR I
```
  - (6) Set the interrupt request level for the interrupt to exit stop mode.
    - Do not rewrite the interrupt control register after this step.
  - (7) Set the IPL in the flag register.
  - (8) Set the interrupt priority level for resuming to the same level as the IPL.
    - Interrupt request level for the interrupt to exit stop mode > IPL = Interrupt priority level for resuming
  - (9) Set the CM20 bit in the CM2 register to 0 (oscillator stop detection disabled) when the oscillator stop detection is used.
  - (10) Change the base clock to either the main clock divided by 256 (f256) or the on-chip oscillator clock divided by 4 (fOCO4).
  - (11) Set the I flag to 1.
  - (12) Execute the STOP instruction.
- After exiting stop mode
  - Set the wake-up interrupt priority level to 7 immediately after exiting stop mode.

### 7.7.3.2 Pin State in Stop Mode

Table 7.6 lists the pin state in stop mode.

**Table 7.6 Pin State in Stop Mode**

Pin		State in Stop Mode
Ports		The state immediately before entering stop mode is held
CLKOUT	When a low speed clock is selected	High
	When f8 or f32 is selected	The state immediately before entering stop mode is held
XIN		High-impedance
XOUT		High
XCIN, XCOU		High-impedance

### 7.7.3.3 Exiting Stop Mode

The MCU exits stop mode by a hardware reset, NMI, low voltage detection interrupt, or a peripheral interrupt assigned to software interrupt number from 0 to 63.

To exit stop mode using either a hardware reset or NMI, without using peripheral interrupts, set bits ILVL2 to ILVL0 for the peripheral interrupts to 000b (interrupt disabled) before executing the STOP instruction.

The CPU clock used when exiting stop mode by a peripheral interrupt or NMI is the same clock used when the STOP instruction is executed.

Table 7.7 lists interrupts used to exit stop mode and usage conditions.

**Table 7.7 Interrupts for Exiting Stop Mode and Usage Conditions**

Interrupt	Usage Condition
NMI	
Low voltage detection interrupt	
External interrupt	
Timer A interrupt Timer B interrupt	Available when a timer counts an external pulse with a frequency of 100 Hz or less in event counter mode
Serial interface interrupt <sup>(1)</sup>	Available when an external clock is used
LIN Low detection interrupt	
CAN wake-up interrupt	

Note:

1. UART3 and UART4 are excluded.

## 7.8 System Clock Protection

The system clock protection disables clock change when the PLL clock is selected as the base clock source. This prevents the CPU clock from stopping due to a runaway program.

When the PM21 bit in the PM2 register is set to 1 (clock change disabled), the following bits cannot be written to:

- Bits CM02 and CM05 in the CM0 register
- The CM10 bit in the CM1 register
- The CM20 bit in the CM2 register
- The PM27 bit in the PM2 register

To use the system clock protection, set the CM05 bit in the CM0 register to 0 (main clock oscillator enabled) and the BCS bit in the CCR register to 0 (PLL clock selected as base clock source) before the following procedure is done:

- (1) Set the PRC1 bit in the PRCR register to 1 (write to the PM2 register enabled).
- (2) Set the PM21 bit in the PM2 register to 1 (clock change disabled).
- (3) Set the PRC1 bit in the PRCR register to 0 (write to the PM2 register disabled).

## 7.9 Notes on Clock Generator

### 7.9.1 Sub Clock

#### 7.9.1.1 Oscillator Constant Matching

The constant matching of the sub clock oscillator should be evaluated in both cases when the drive strength is high and low.

Contact the oscillator manufacturer for details on the oscillation circuit constant matching.

### 7.9.2 Power Control

Do not switch the base clock source until the oscillation of the clock to be used has stabilized. However, this does not apply to the on-chip oscillator since it starts running immediately after the CM31 bit in the CM3 register is set to 1.

To switch the base clock source from the PLL clock to a low speed clock, use the MOV.L or OR.L instruction to set the BCS bit in the CCR register to 1.

- Program example in assembly language

```
OR.L    #80h, 0004h
```

- Program example in C language

```
asm("OR.L #80h, 0004h");
```

#### 7.9.2.1 Stop Mode

- To exit stop mode using a reset, apply a low signal to the  $\overline{\text{RESET}}$  pin until the main clock oscillation stabilizes.

#### 7.9.2.2 Suggestions for Power Saving

The following are suggestions to reduce power consumption when programming or designing systems.

- I/O pins:

If inputs are floating, both transistors may be conducting. Set unassigned pins to input mode and connect each of them to VSS via a resistor, or set them to output mode and leave them open.

- A/D converter:

When not performing the A/D conversion, set the VCUT bit in the AD0CON1 register to 0 (VREF disconnected). To perform the A/D conversion, set the VCUT bit to 1 (VREF connected) and wait at least 1  $\mu\text{s}$  before starting conversion.

- Peripheral clock stop:

When entering wait mode, power consumption can be reduced by setting the CM02 bit in the CM0 register to 1 to stop the peripheral clock source. However, this setting does not stop the fC32.



## 8. Bus

This MCU has a fast bus (CPU bus) and a slow bus (peripheral bus). Figure 8.1 shows a block diagram of the bus.

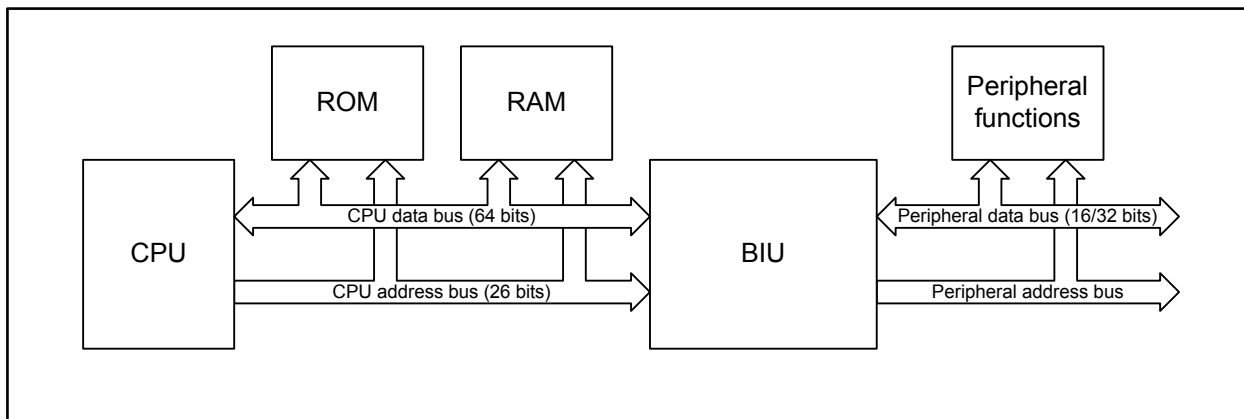


Figure 8.1 Bus Block Diagram

### 8.1 Bus Setting

The bus setting is controlled by the PBC register.

Table 8.1 lists the bus setting and its source.

Table 8.1 Bus Setting and Source

Bus Setting	Source
Internal SFR bus timing	PBC register

## 8.2 Peripheral Bus Timing Setting

The 16-/32-bit wide peripheral bus operates at a frequency up to 32 MHz (the theoretical value and the maximum frequency of each product group are as defined by  $f(\text{BCLK})$  in 28. "Electrical Characteristics"). The timing adjustment and bus-width conversion with the faster, 64-bit wide CPU bus are controlled in the bus interface unit (BIU).

Figure 8.2 shows the PBC register which determines the peripheral bus timing.

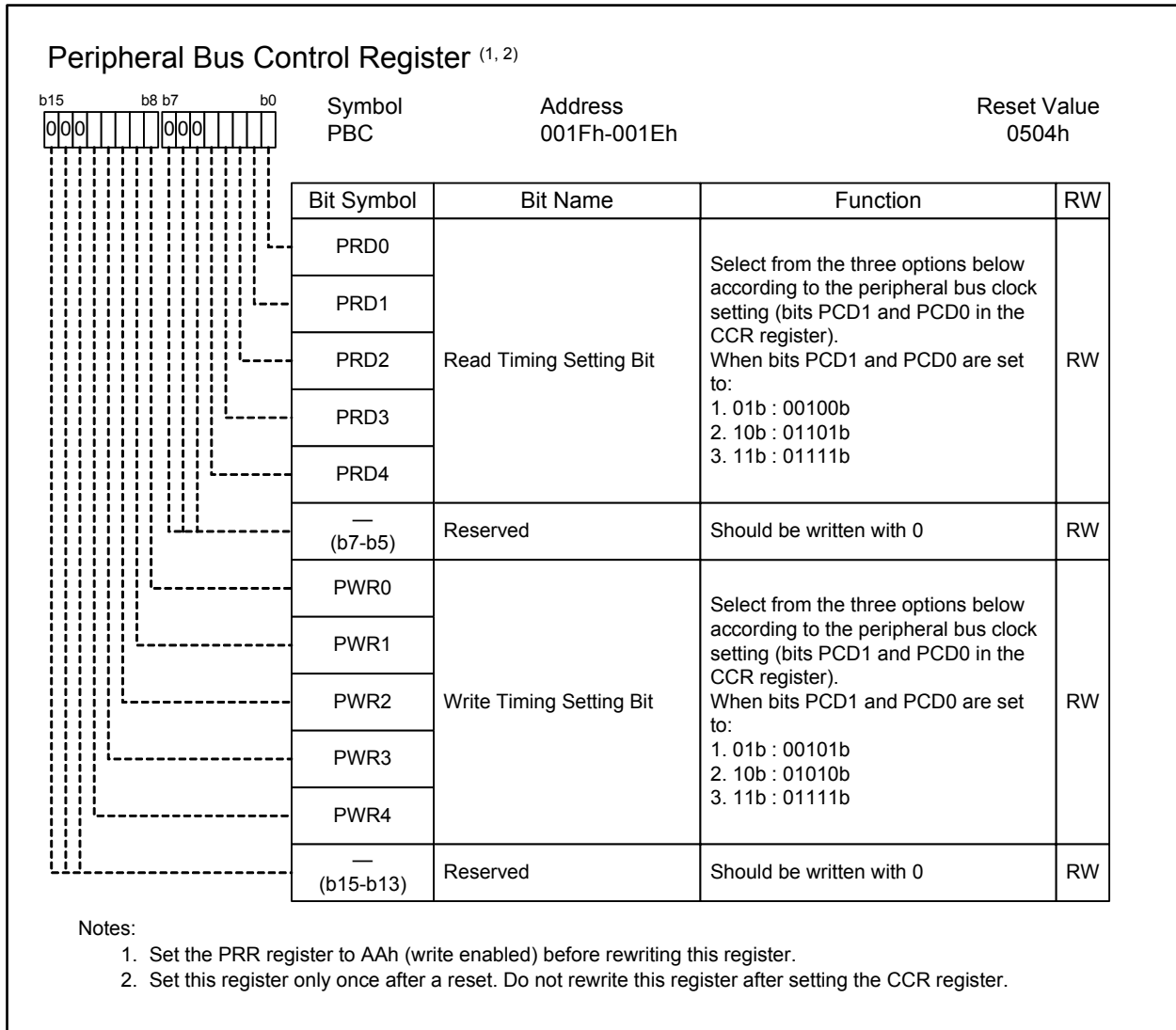


Figure 8.2 PBC Register

## 9. Protection

This function protects important registers from being easily overwritten when a program goes out of control. Registers used to protect other registers from being rewritten are as follows: PRCR, PRCR2 to PRCR4, and PRR.

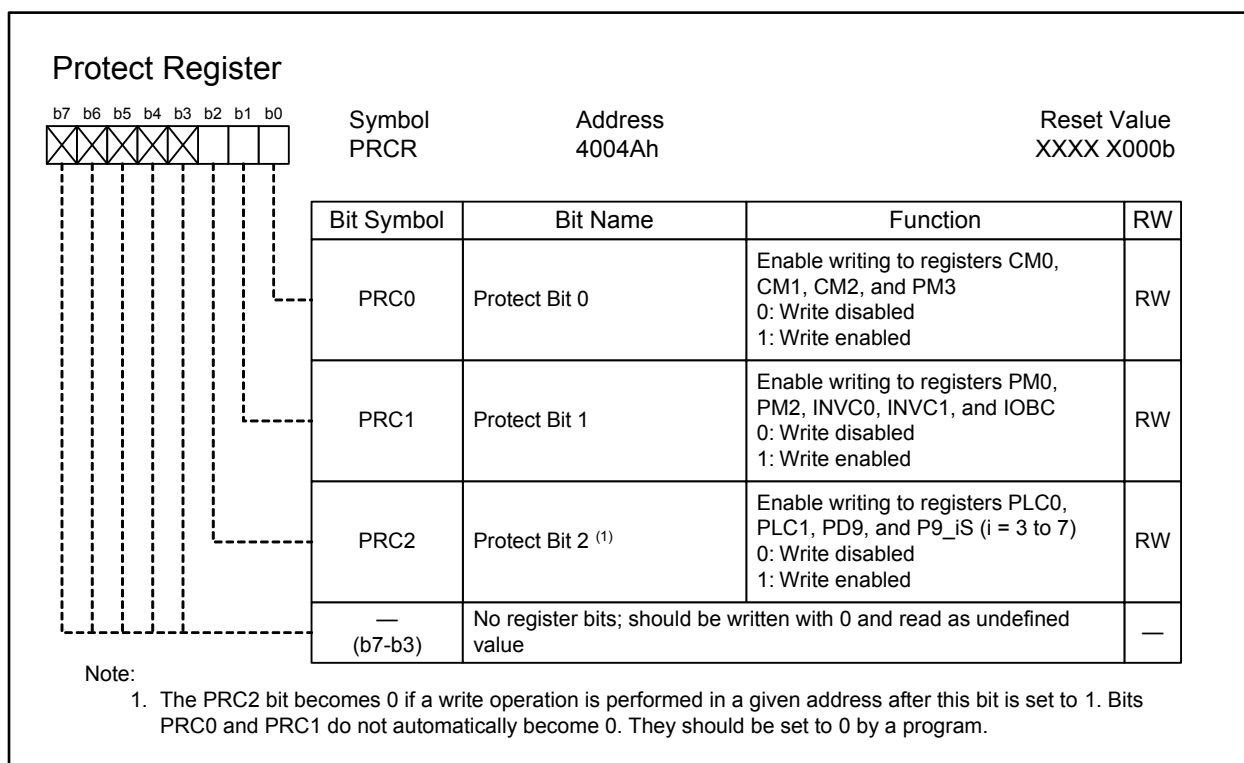
### 9.1 Protect Register (PRCR Register)

Figure 9.1 shows the PRCR register. Registers protected by bits in the PRCR register are listed in Table 9.1.

**Table 9.1 Registers Protected by the PRCR Register**

Bit	Protected Registers
PRC0	CM0, CM1, CM2, and PM3
PRC1	PM0, PM2, INVC0, INVC1, and IOBC
PRC2	PLC0, PLC1, PD9, and P9_iS (i = 3 to 7)

The PRC2 bit becomes 0 (write disabled) when a write operation is performed in a given address after this bit is set to 1 (write enabled). Set the PRC2 bit to 1 just before rewriting registers PD9, P9\_iS, PLC0, and PLC1 (i = 3 to 7). No interrupt handling or DMA transfers should be inserted between these two instructions. Bits PRC0 and PRC1 do not become 0 even if data is written to a given address. These bits should be set to 0 by a program.



**Figure 9.1 PRCR Register**

## 9.2 Protect Register 2 (PRCR2 Register)

Figure 9.2 shows the PRCR2 register which protects the CM3 register only.

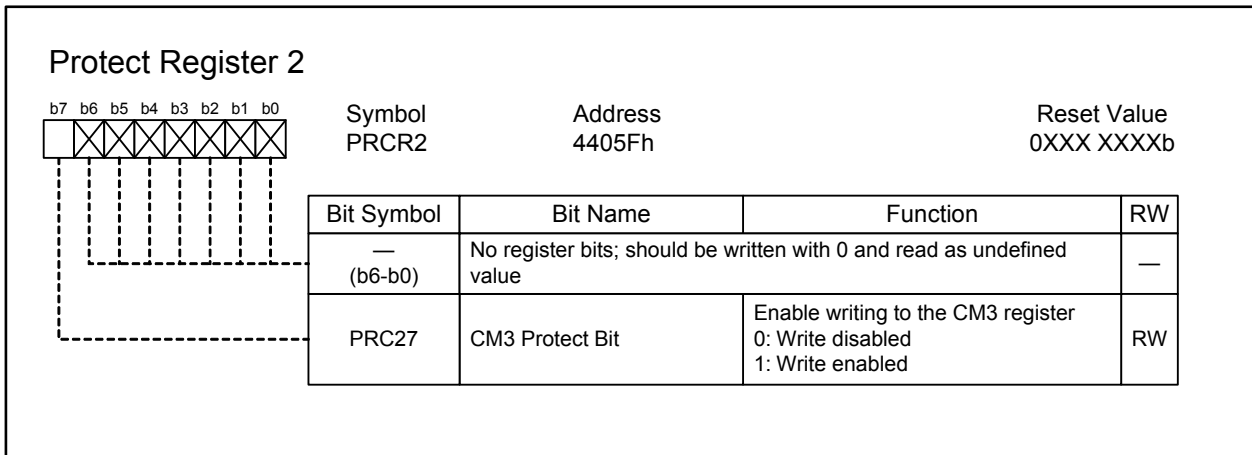


Figure 9.2 PRCR2 Register

## 9.3 Protect Register 3 (PRCR3 Register)

Figure 9.3 shows the PRCR3 register. Registers protected by the bits in the PRCR3 register are listed in Table 9.2.

Table 9.2 Registers Protected by the PRCR3 Register (i = 0 to 7)

Bit	Protected Registers
PRC30	PD3, P3_iS, PD7, P7_iS, PD8, and P8_iS
PRC31	VRCR, LVDC, and DVCR

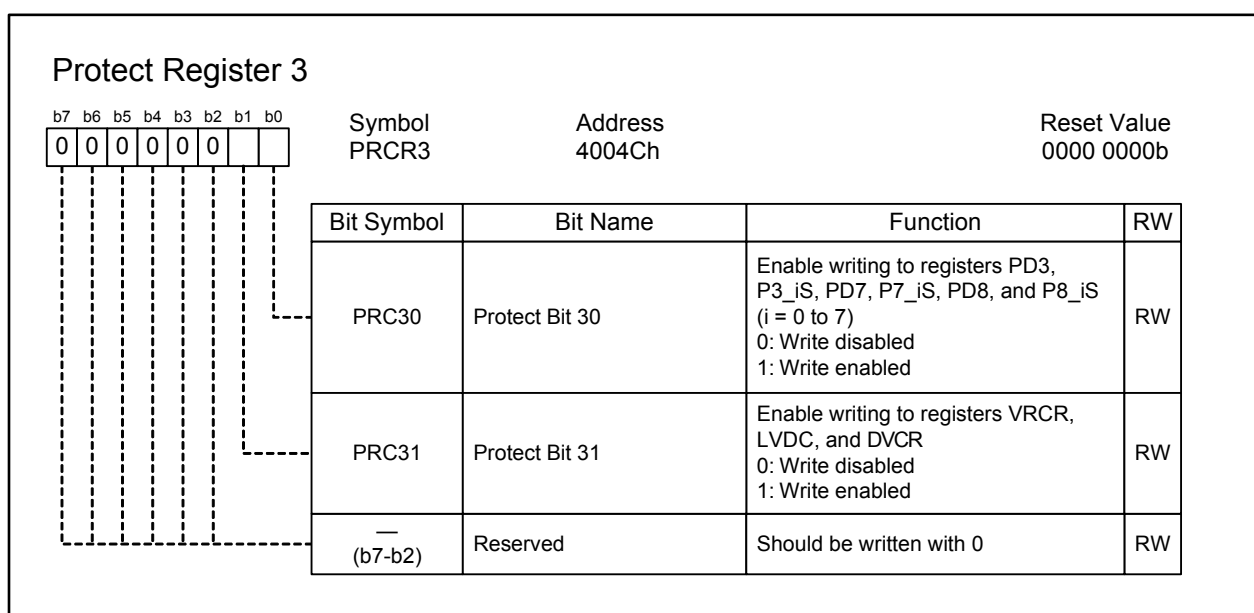


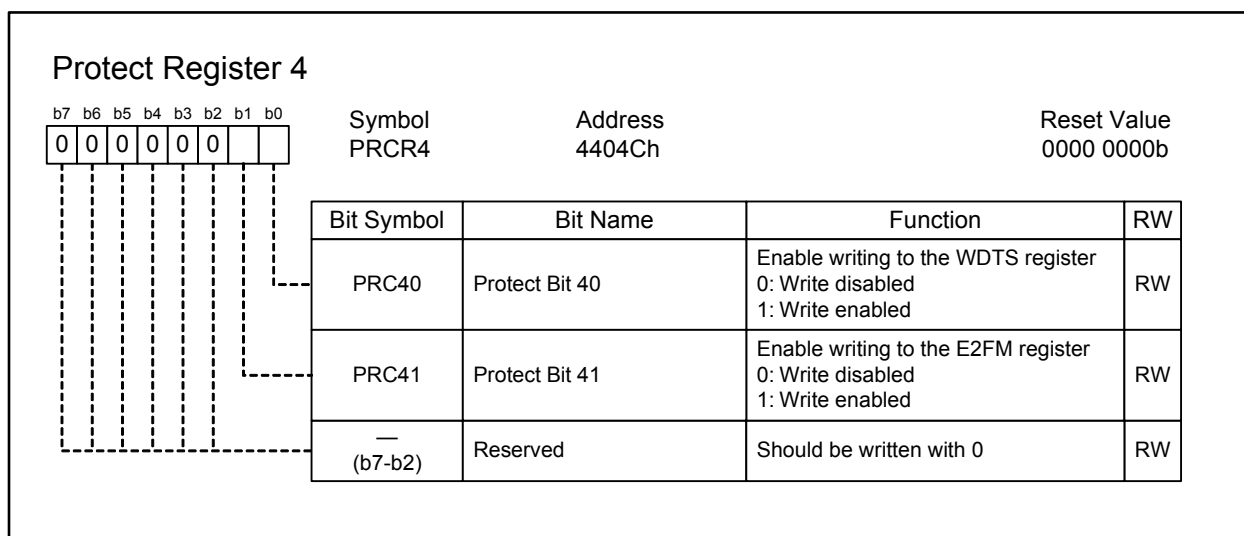
Figure 9.3 PRCR3 Register

## 9.4 Protect Register 4 (PRCR4 Register)

Figure 9.4 shows the PRCR4 register. Registers protected by the bits in the PRCR4 register are listed in Table 9.3.

**Table 9.3 Registers Protected by the PRCR4 Register**

Bit	Protected Registers
PRC40	WDTS
PRC41	E2FM

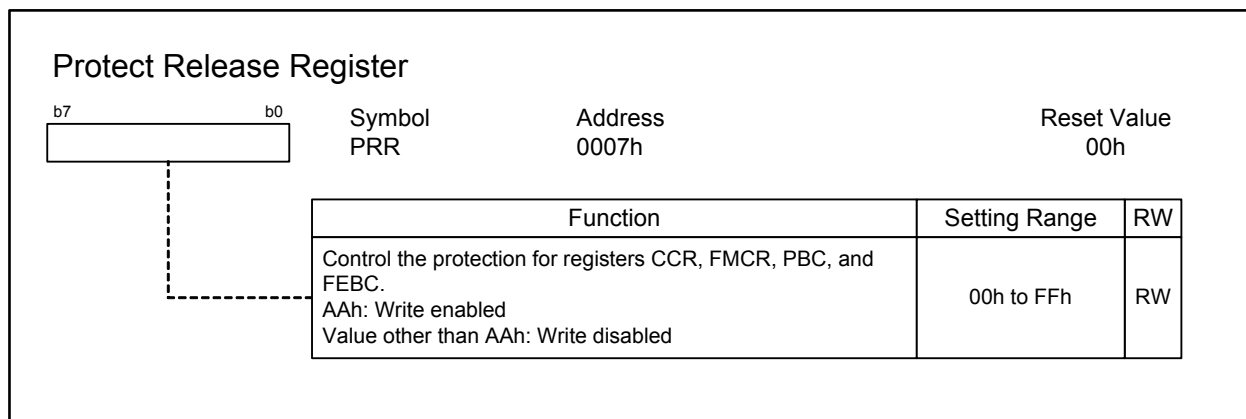


**Figure 9.4 PRCR4 Register**

## 9.5 Protect Release Register (PRR Register)

Figure 9.5 shows the PRR register. Registers protected by the PRR register are as follows: CCR, FMCR, PBC, and FEBC.

To write to the registers above, the PRR register should be set to AAh (write enabled). Otherwise, the PRR register should be set to any value other than AAh to protect the above registers from unexpected write accesses.

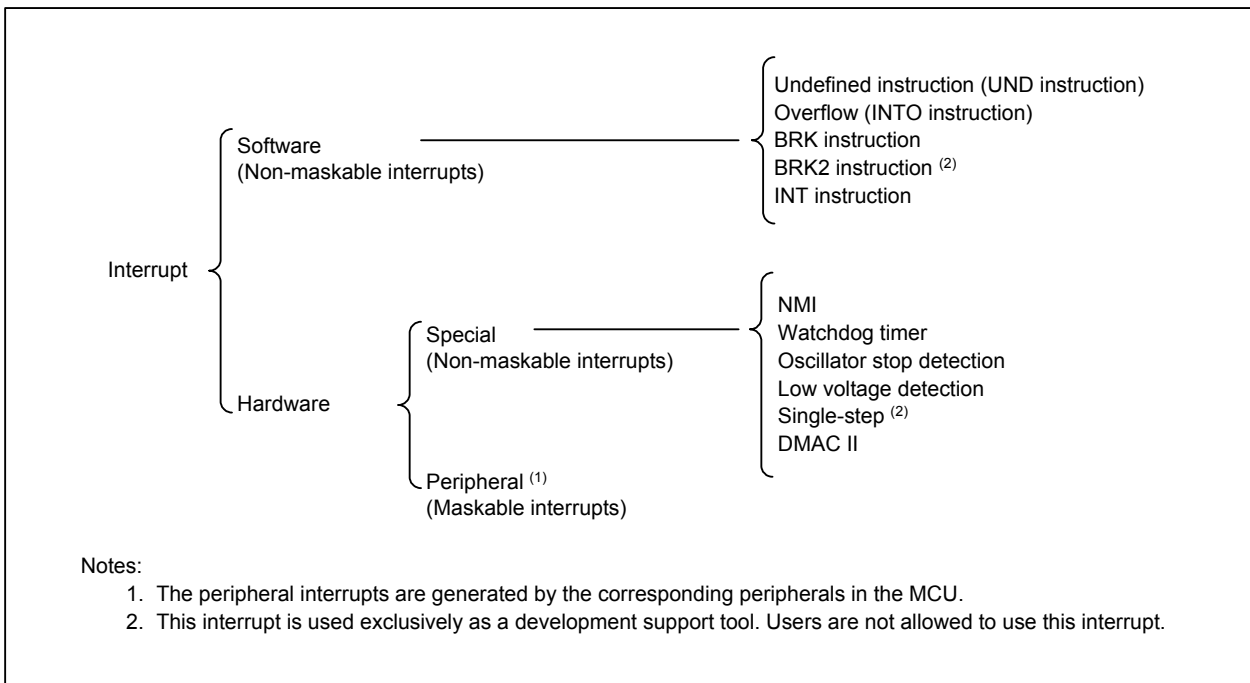


**Figure 9.5 PRR Register**

## 10. Interrupts

### 10.1 Interrupt Types

Figure 10.1 shows the types of interrupts.



**Figure 10.1** Interrupts

Interrupts are also classified into maskable/non-maskable.

#### (1) Maskable Interrupts

Maskable interrupts can be disabled by the interrupt enable flag (I flag).  
The priority can be configured by assigning an interrupt request level.

#### (2) Non-maskable Interrupts

Maskable interrupts cannot be disabled by the interrupt enable flag (I flag).  
The interrupt priority cannot be configured.

## 10.2 Software Interrupts

Software interrupts are non-maskable. A software interrupt occurs by executing an instruction. There are five types of software interrupts shown below.

### (1) Undefined Instruction Interrupt

This interrupt occurs when the UND instruction is executed.

### (2) Overflow Interrupt

This interrupt occurs when the INTO instruction is executed while the O flag is 1. The following instructions may change the O flag to 1, depending on the operation result:

ABS, ADC, ADCF, ADD, ADDF, ADSF, CMP, CMPF, CNVIF, DIV, DIVF, DIVU, DIVX, EDIV, EDIVU, EDIVX, MUL, MULF, MULU, MULX, NEG, RMPA, ROUND, SBB, SCMPU, SHA, SUB, SUBF, SUNTIL, and SWHILE

### (3) BRK Instruction Interrupt

This interrupt occurs when the BRK instruction is executed.

### (4) BRK2 Instruction Interrupt

This interrupt occurs when the BRK2 instruction is executed.

This interrupt is only meant for use as a development support tool and users are not allowed to use it.

### (5) INT Instruction Interrupt

This interrupt occurs when the INT instruction is executed with a selected software interrupt number from 0 to 255. Software interrupt numbers 0 to 127 are designated for peripheral interrupts. That is, the INT instruction with a software interrupt number from 0 to 127 has the same interrupt handler as that for peripheral interrupts.

The stack pointer (SP) used for this interrupt differs depending on the software interrupt numbers. For software interrupt numbers 0 to 127, when an interrupt request is accepted, the U flag is saved and set to 0 to select the interrupt stack pointer (ISP) during the interrupt sequence. The saved data of the U flag is restored upon returning from the interrupt handler. For software interrupt numbers 128 to 255, the stack pointer does not change during the interrupt sequence.



## 10.3 Hardware Interrupts

There are two kinds of hardware interrupts: special interrupts and peripheral interrupts. In peripheral interrupts, only one interrupt with the highest priority can be specified as a fast interrupt.

### 10.3.1 Special Interrupts

Special interrupts are non-maskable. There are five special interrupts shown below.

(1) NMI (Non Maskable Interrupt)

This interrupt occurs when an input signal at the  $\overline{\text{NMI}}$  pin switches from high to low. Refer to 10.11 “NMI” for details.

(2) Watchdog Timer Interrupt

The watchdog timer generates this interrupt. Refer to 11. “Watchdog Timer” for details.

(3) Oscillator Stop Detection Interrupt

This interrupt occurs when the MCU detects a main clock oscillator stop. Refer to 7.2 “Oscillator Stop Detection” for details.

(4) Low Voltage Detection Interrupt

This interrupt occurs when a low voltage input to VCC is detected by the voltage detector. Refer to 6.2 “Low Voltage Detector” for details.

(5) Single-step Interrupt

This interrupt is only meant for use as a development support tool and users are not allowed to use it.

### 10.3.2 Peripheral Interrupts

Peripheral interrupts occur when an interrupt request from a peripheral in the MCU is accepted. They share the interrupt vector with software interrupt numbers 0 to 127 for the INT instruction. Peripheral interrupts are maskable.

Refer to Tables 10.2 to 10.5 for details on the interrupt sources. Refer to the relevant descriptions for details on each function.

## 10.4 Fast Interrupt

A fast interrupt enables the CPU to accelerate interrupt response. In peripheral interrupts, only one interrupt with the highest priority can be specified as the fast interrupt.

Use the following procedure to enable a fast interrupt:

- (1) Set the both FSIT bit in registers RIPL1 and RIPL2 to 1 (interrupt request level 7 available for fast interrupt).
- (2) Set the both DMAII bit in registers RIPL1 and RIPL2 to 0 (interrupt request level 7 available for interrupts).
- (3) Set the start address of the fast interrupt handler to the VCT register.

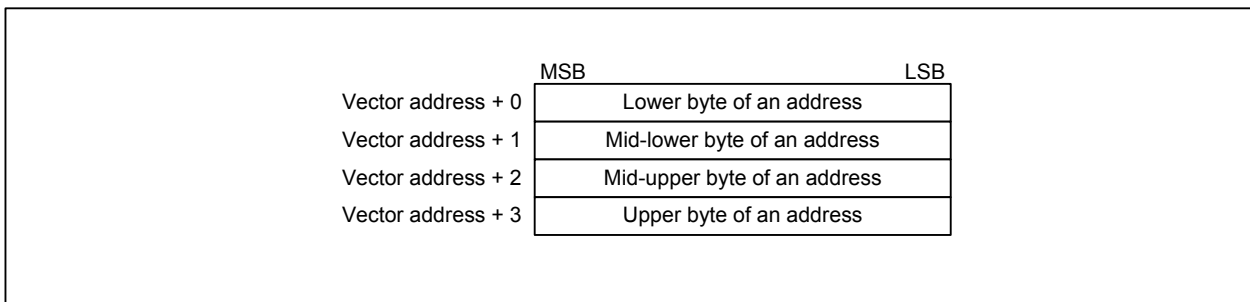
Under the conditions above, bits ILVL2 to ILVL0 in the interrupt control register should be set to 111b (level 7) to enable the fast interrupt. No other interrupts should be set to interrupt request level 7.

When the fast interrupt is accepted, the flag register (FLG) and program counter (PC) are saved to the save flag register (SVF) and save PC register (SVP), respectively. The program is executed from the address indicated by the VCT register.

To return from the fast interrupt handler, the FREIT instruction should be executed. The values saved into registers SVF and SVP are restored to the FLG register and PC, respectively.

## 10.5 Interrupt Vectors

Each interrupt vector has a 4-byte memory space, in which the start address of the associated interrupt handler is stored. When an interrupt request is accepted, a jump to the address set in the interrupt vector takes place. Figure 10.2 shows an interrupt vector.



**Figure 10.2** Interrupt Vector

### 10.5.1 Fixed Vector Table

The fixed vector table is allocated in addresses FFFFFFFDCh to FFFFFFFFh. Table 10.1 lists the fixed vector table.

**Table 10.1 Fixed Vector Table**

Interrupt Source	Vector Addresses (Address (L) to Address (H))	Remarks	Reference
Undefined instruction	FFFFFFDCh to FFFFFFFDFh	Interrupt by the UND instruction	R32C/100 Series Software Manual
Overflow	FFFFFFE0h to FFFFFFFE3h	Interrupt by the INTO instruction	
BRK instruction	FFFFFFE4h to FFFFFFFE7h	If address FFFFFFFE7h is FFh, a jump to the interrupt vector of software interrupt number 0 in the relocatable vector table takes place	
—	FFFFFFE8h to FFFFFFFEBh	Reserved	
—	FFFFFFECh to FFFFFFFEFh	Reserved	
Watchdog timer Oscillator stop detection Low voltage detection	FFFFFFF0h to FFFFFFF3h	These addresses are shared by the watchdog timer interrupt, oscillator stop detection interrupt, and low voltage detection interrupt	11. "Watchdog Timer" 7. "Clock Generator" 6.2 "Low Voltage Detector"
—	FFFFFFF4h to FFFFFFF7h	Reserved	
NMI	FFFFFFF8h to FFFFFFFBh	External interrupt by the $\overline{\text{NMI}}$ pin	
Reset	FFFFFFFCh to FFFFFFFFh		5. "Resets"

### 10.5.2 Relocatable Vector Table

The relocatable vector table occupies a 1024-byte memory space from the start address set in the INTB register. Tables 10.2 to 10.5 list the relocatable vector table entries.

An address in a multiple of 4 should be set in the INTB register for a faster interrupt sequence.

**Table 10.2 Relocatable Vector Table (1/4)**

Interrupt Source	Vector Table Relative Addresses (Address (L) to Address (H)) <sup>(1)</sup>	Software Interrupt Number	Reference
BRK instruction <sup>(2)</sup>	+0 to +3 (0000h to 0003h)	0	R32C/100 Series Software Manual
Reserved	+4 to +7 (0004h to 0007h)	1	
Reserved	+8 to +11 (0008h to 000Bh)	2	17. "Serial Interface" (Reserved)
Reserved	+12 to +15 (000Ch to 000Fh)	3	
Reserved	+16 to +19 (0010h to 0013h)	4	
Reserved	+20 to +23 (0014h to 0017h)	5	
Reserved	+24 to +27 (0018h to 001Bh)	6	
Reserved	+28 to +31 (001Ch to 001Fh)	7	
DMA0 transfer complete	+32 to +35 (0020h to 0023h)	8	12. "DMAC"
DMA1 transfer complete	+36 to +39 (0024h to 0027h)	9	
DMA2 transfer complete	+40 to +43 (0028h to 002Bh)	10	
DMA3 transfer complete	+44 to +47 (002Ch to 002Fh)	11	
Timer A0	+48 to +51 (0030h to 0033h)	12	15.1 "Timer A"
Timer A1	+52 to +55 (0034h to 0037h)	13	
Timer A2	+56 to +59 (0038h to 003Bh)	14	
Timer A3	+60 to +63 (003Ch to 003Fh)	15	
Timer A4	+64 to +67 (0040h to 0043h)	16	
UART0 transmission, NACK <sup>(3)</sup>	+68 to +71 (0044h to 0047h)	17	
UART0 reception, ACK <sup>(3)</sup>	+72 to +75 (0048h to 004Bh)	18	
UART1 transmission, NACK <sup>(3)</sup>	+76 to +79 (004Ch to 004Fh)	19	
UART1 reception, ACK <sup>(3)</sup>	+80 to +83 (0050h to 0053h)	20	
Timer B0	+84 to +87 (0054h to 0057h)	21	15.2 "Timer B"
Timer B1	+88 to +91 (0058h to 005Bh)	22	
Timer B2	+92 to +95 (005Ch to 005Fh)	23	
Timer B3	+96 to +99 (0060h to 0063h)	24	
Timer B4	+100 to +103 (0064h to 0067h)	25	
INT5	+104 to +107 (0068h to 006Bh)	26	
INT4	+108 to +111 (006Ch to 006Fh)	27	
INT3	+112 to +115 (0070h to 0073h)	28	
INT2	+116 to +119 (0074h to 0077h)	29	
INT1	+120 to +123 (0078h to 007Bh)	30	
INT0	+124 to +127 (007Ch to 007Fh)	31	
Timer B5	+128 to +131 (0080h to 0083h)	32	15.2 "Timer B"

## Notes:

- Each entry is relative to the base address in the INTB register.
- Interrupts from this source cannot be disabled by the I flag.
- In I<sup>2</sup>C mode, interrupts are generated by NACK, ACK, or detection of a start condition/stop condition.

**Table 10.3 Relocatable Vector Table (2/4)**

Interrupt Source	Vector Table Relative Addresses (Address (L) to Address (H)) <sup>(1)</sup>	Software Interrupt Number	Reference
UART2 transmission, NACK <sup>(2)</sup>	+132 to +135 (0084h to 0087h)	33	17. "Serial Interface"
UART2 reception, ACK <sup>(2)</sup>	+136 to +139 (0088h to 008Bh)	34	
Reserved	+140 to +143 (008Ch to 008Fh)	35	
Reserved	+144 to +147 (0090h to 0093h)	36	
Reserved	+148 to +151 (0094h to 0097h)	37	
Reserved	+152 to +155 (0098h to 009Bh)	38	
Start condition detection or stop condition detection (UART2) <sup>(2)</sup>	+156 to +159 (009Ch to 009Fh)	39	
Start condition detection or stop condition detection (UART0) <sup>(2)</sup>	+160 to +163 (00A0h to 00A3h)	40	
Start condition detection or stop condition detection (UART1) <sup>(2)</sup>	+164 to +167 (00A4h to 00A7h)	41	
A/D0	+168 to +171 (00A8h to 00ABh)	42	18. "A/D Converter"
Reserved	+172 to +175 (00ACh to 00AFh)	43	
Intelligent I/O interrupt 0	+176 to +179 (00B0h to 00B3h)	44	10.12 "Intelligent I/O Interrupt", 21. "Intelligent I/O"
Intelligent I/O interrupt 1	+180 to +183 (00B4h to 00B7h)	45	
Intelligent I/O interrupt 2	+184 to +187 (00B8h to 00BBh)	46	
Intelligent I/O interrupt 3	+188 to +191 (00BCh to 00BFh)	47	
Intelligent I/O interrupt 4	+192 to +195 (00C0h to 00C3h)	48	
Intelligent I/O interrupt 5	+196 to +199 (00C4h to 00C7h)	49	
Intelligent I/O interrupt 6	+200 to +203 (00C8h to 00CBh)	50	
Intelligent I/O interrupt 7	+204 to +207 (00CCh to 00CFh)	51	
Intelligent I/O interrupt 8	+208 to +211 (00D0h to 00D3h)	52	
Intelligent I/O interrupt 9	+212 to +215 (00D4h to 00D7h)	53	
Intelligent I/O interrupt 10	+216 to +219 (00D8h to 00DBh)	54	
Intelligent I/O interrupt 11	+220 to +223 (00DCh to 00DFh)	55	
Reserved	+224 to +227 (00E0h to 00E3h)	56	
Reserved	+228 to +231 (00E4h to 00E7h)	57	
CAN0 wakeup	+232 to +235 (00E8h to 00EBh)	58	24. "CAN Module"
CAN1 wakeup	+236 to +239 (00ECh to 00EFh)	59	
Reserved	+240 to +243 (00F0h to 00F3h)	60	
Reserved	+244 to +247 (00F4h to 00F7h)	61	
LIN Low detection	+248 to +251 (00F8h to 00FBh)	62	23. "LIN Module"
Reserved	+252 to +255 (00FCh to 00FFh)	63	

## Notes:

- Each entry is relative to the base address in the INTB register.
- In I<sup>2</sup>C mode, interrupts are generated by NACK, ACK, or detection of a start condition/stop condition.

**Table 10.4 Relocatable Vector Table (3/4) (1)**

Interrupt Source	Vector Table Relative Addresses (Address (L) to Address (H)) (2)	Software Interrupt Number	Reference
Serial bus interface 0	+256 to +259 (0100h to 0103h)	64	22. "Serial Bus Interface"
Reserved	+260 to +263 (0104h to 0107h)	65	
Reserved	+264 to +267 (0108h to 010Bh)	66	
Reserved	+268 to +271 (010Ch to 010Fh)	67	
Reserved	+272 to +275 (0110h to 0113h)	68	
Reserved	+276 to +279 (0114h to 0117h)	69	
Reserved	+280 to +283 (0118h to 011Bh)	70	
Reserved	+284 to +287 (011Ch to 011Fh)	71	
Reserved	+288 to +291 (0120h to 0123h)	72	
Reserved	+292 to +295 (0124h to 0127h)	73	
Reserved	+296 to +299 (0128h to 012Bh)	74	
Reserved	+300 to +303 (012Ch to 012Fh)	75	
Reserved	+304 to +307 (0130h to 0133h)	76	
Reserved	+308 to +311 (0134h to 0137h)	77	
Reserved	+312 to +315 (0138h to 013Bh)	78	
Reserved	+316 to +319 (013Ch to 013Fh)	79	
CAN0 transmit FIFO	+320 to +323 (0140h to 0143h)	80	
CAN0 receive FIFO	+324 to +327 (0144h to 0147h)	81	
CAN1 transmit FIFO	+328 to +331 (0148h to 014Bh)	82	
CAN1 receive FIFO	+332 to +335 (014Ch to 014Fh)	83	
Reserved	+336 to +339 (0150h to 0153h)	84	
Reserved	+340 to +343 (0154h to 0157h)	85	
Reserved	+344 to +347 (0158h to 015Bh)	86	
Reserved	+348 to +351 (015Ch to 015Fh)	87	27. "E2PROM Emulation Data Flash"
E <sup>2</sup> dataFlash	+352 to +355 (0160h to 0163h)	88	
Reserved	+356 to +359 (0164h to 0167h)	89	
Reserved	+360 to +363 (0168h to 016Bh)	90	
Reserved	+364 to +367 (016Ch to 016Fh)	91	
Reserved	+368 to +371 (0170h to 0173h)	92	
Reserved	+372 to +375 (0174h to 0177h)	93	
Reserved	+376 to +379 (0178h to 017Bh)	94	
Reserved	+380 to +383 (017Ch to 017Fh)	95	

## Notes:

1. Entries in this table cannot be used to exit wait mode or stop mode.
2. Each entry is relative to the base address in the INTB register.

**Table 10.5 Relocatable Vector Table (4/4) (1)**

Interrupt Source	Vector Table Relative Addresses (Address (L) to Address (H)) (2)	Software Interrupt Number	Reference	
CAN0 transmission	+384 to +387 (0180h to 0183h)	96	24. "CAN Module"	
CAN0 reception	+388 to +391 (0184h to 0187h)	97		
CAN0 error	+392 to +395 (0188h to 018Bh)	98		
CAN1 transmission	+396 to +399 (018Ch to 018Fh)	99		
CAN1 reception	+400 to +403 (0190h to 0193h)	100		
CAN1 error	+404 to +407 (0194h to 0197h)	101		
Reserved	+408 to +411 (0198h to 019Bh)	102		
Reserved	+412 to +415 (019Ch to 019Fh)	103		
Reserved	+416 to +419 (01A0h to 01A3h)	104		
Reserved	+420 to +423 (01A4h to 01A7h)	105		
Reserved	+424 to +427 (01A8h to 01ABh)	106		
Reserved	+428 to +431 (01ACh to 01AFh)	107		
Reserved	+432 to +435 (01B0h to 01B3h)	108		
Reserved	+436 to +439 (01B4h to 01B7h)	109		
Reserved	+440 to +443 (01B8h to 01BBh)	110		
Reserved	+444 to +447 (01BCh to 01BFh)	111		
Reserved	+448 to +451 (01C0h to 01C3h)	112		23. "LIN Module"
Reserved	+452 to +455 (01C4h to 01C7h)	113		
Reserved	+456 to +459 (01C8h to 01CBh)	114		
Reserved	+460 to +463 (01CCh to 01CFh)	115		
LIN0	+464 to +467 (01D0h to 01D3h)	116		
Reserved	+468 to +471 (01D4h to 01D7h)	117		
Reserved	+472 to +475 (01D8h to 01DBh)	118		
Reserved	+476 to +479 (01DCh to 01DFh)	119		
Reserved	+480 to +483 (01E0h to 01E3h)	120		
Reserved	+484 to +487 (01E4h to 01E7h)	121		
Reserved	+488 to +491 (01E8h to 01EBh)	122		
Reserved	+492 to +495 (01ECh to 01EFh)	123		
UART3 transmission	+496 to +499 (01F0h to 01F3h)	124	17. "Serial Interface"	
UART3 reception	+500 to +503 (01F4h to 01F7h)	125		
UART4 transmission	+504 to +507 (01F8h to 01FBh)	126		
UART4 reception	+508 to +511 (01FCh to 01FFh)	127		
INT instruction (3)	+0 to +3 (0000h to 0003h) to +1020 to +1023 (03FCh to 03FFh)	0 to 255	10.2 "Software Interrupts"	

## Notes:

1. Entries in this table cannot be used to exit wait mode or stop mode.
2. Each entry is relative to the base address in the INTB register.
3. Interrupts from this source cannot be disabled by the I flag.

## 10.6 Interrupt Request Acceptance

Software interrupts and special interrupts are accepted whenever their interrupt request is generated. Peripheral interrupts, however, are only accepted if the conditions below are met:

- I flag is 1
- IR bit is 1
- Bits ILVL2 to ILVL0 > IPL

The I flag, IPL, IR bit, and bits ILVL2 to ILVL0 do not affect each other. The I flag and IPL are in the FLG register. The IR bit and bits ILVL2 to ILVL0 are in the interrupt control register.

The following section describes these flag and bits.

### 10.6.1 I Flag and IPL

The I flag (interrupt enable flag) enables or disables maskable interrupts. When the I flag is set to 1 (enabled), all maskable interrupts are enabled; when it is set to 0 (disabled), they are disabled. The I flag becomes 0 after a reset.

The IPL (processor interrupt priority level) consists of 3 bits and indicates eight interrupt priority levels from 0 to 7. An interrupt becomes acceptable when its interrupt request level is higher than the specified IPL (bits ILVL2 to ILVL0 > IPL).

Table 10.6 lists interrupt request levels classified by the IPL.

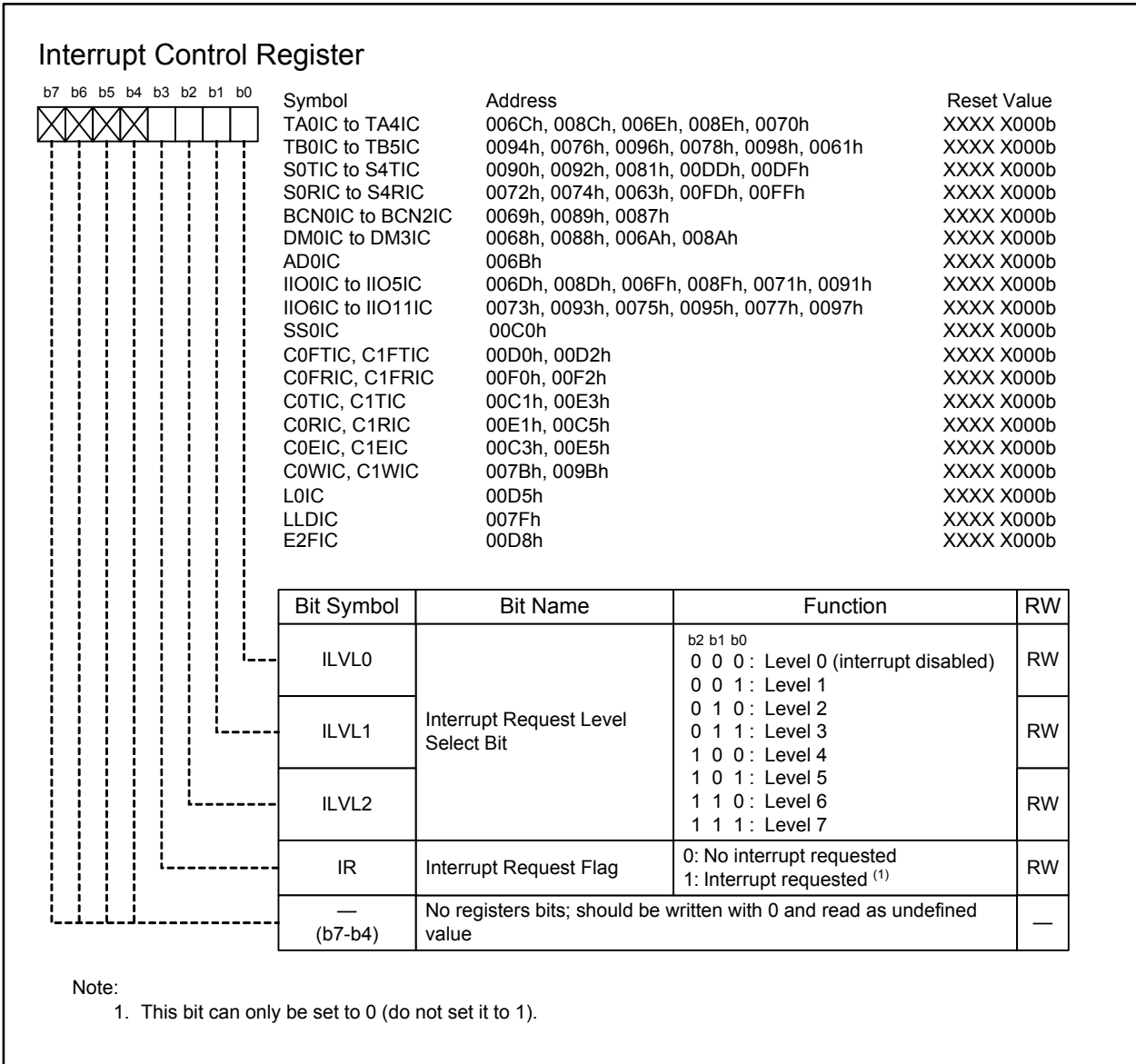
**Table 10.6 Acceptable Interrupt Request Levels and IPL**

IPL			Acceptable Interrupt Request Levels
IPL2	IPL1	IPL0	
1	1	1	All maskable interrupts are disabled
1	1	0	Level 7 only
1	0	1	Level 6 and above
1	0	0	Level 5 and above
0	1	1	Level 4 and above
0	1	0	Level 3 and above
0	0	1	Level 2 and above
0	0	0	Level 1 and above

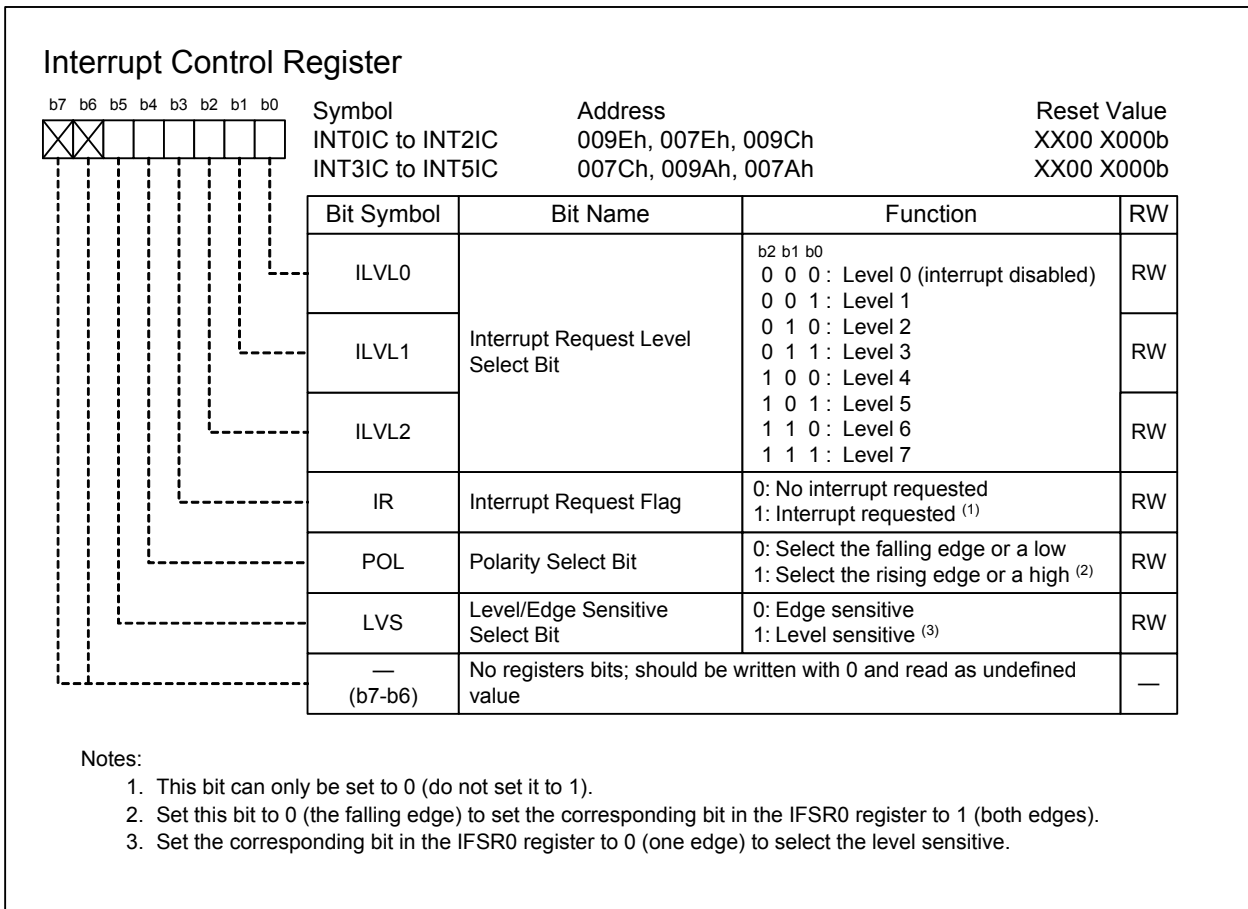


## 10.6.2 Interrupt Control Registers

Each peripheral interrupt is controlled by an interrupt control register. Figures 10.3 and 10.4 show the interrupt control registers.



**Figure 10.3 Interrupt Control Register (1/2)**



**Figure 10.4 Interrupt Control Register (2/2)**

#### Bits ILVL2 to ILVL0

The interrupt request level is selected by setting bits ILVL2 to ILVL0. The higher the level is, the higher interrupt priority is.

When an interrupt request is generated, its request level is compared to the IPL. The interrupt is accepted only when the interrupt request level is higher than the IPL. When bits ILVL2 to ILVL0 are set to 000b, the interrupt is disabled.

#### IR bit

The IR bit becomes 1 (interrupt requested) when an interrupt request is generated; this bit setting is retained until the interrupt request is accepted. When the request is accepted and a jump to the corresponding interrupt vector takes place, the IR bit becomes 0 (no interrupt requested).

The IR bit can be set to 0 by a program. This bit should not be set to 1.

When rewriting the interrupt control register, no corresponding interrupt request should be generated. If there is a possibility that an interrupt request may be generated, disable all maskable interrupts before rewriting the register.

When enabling an interrupt immediately after changing the interrupt control register, insert NOPs between two instructions or perform a dummy read of the interrupt control register so that the interrupt enable flag (I flag) cannot become 1 (interrupt enabled) before writing to the interrupt control register is completed.

If an interrupt request is generated for the register being rewritten, the IR bit may not become 1 depending on the instruction being used. If it matters, use one of the following instructions to rewrite the register:

- AND
- OR
- BCLR
- BSET

If the AND or BCLR instruction is used to set the IR bit to 0, the IR bit may not become 0 as these instructions cause the interrupt request to be retained during the rewrite. To prevent this from happening, rewrite the register using the MOV instruction. To set just the IR bit to 0, first temporarily store the read value to memory or a CPU internal register, then execute either the AND or BCLR instruction in the stored area. After that, write the value back to the register using the MOV instruction.

### 10.6.3 Wake-up IPL Setting Register

Set the wake-up IPL setting registers (registers RIPL1 and RIPL2) when using an interrupt to exit wait or stop mode, or using the fast interrupt.

Refer to 7.7.2 “Wait Mode”, 7.7.3 “Stop Mode”, or 10.4 “Fast Interrupt” for details.

Figure 10.5 shows registers RIPL1 and RIPL2.

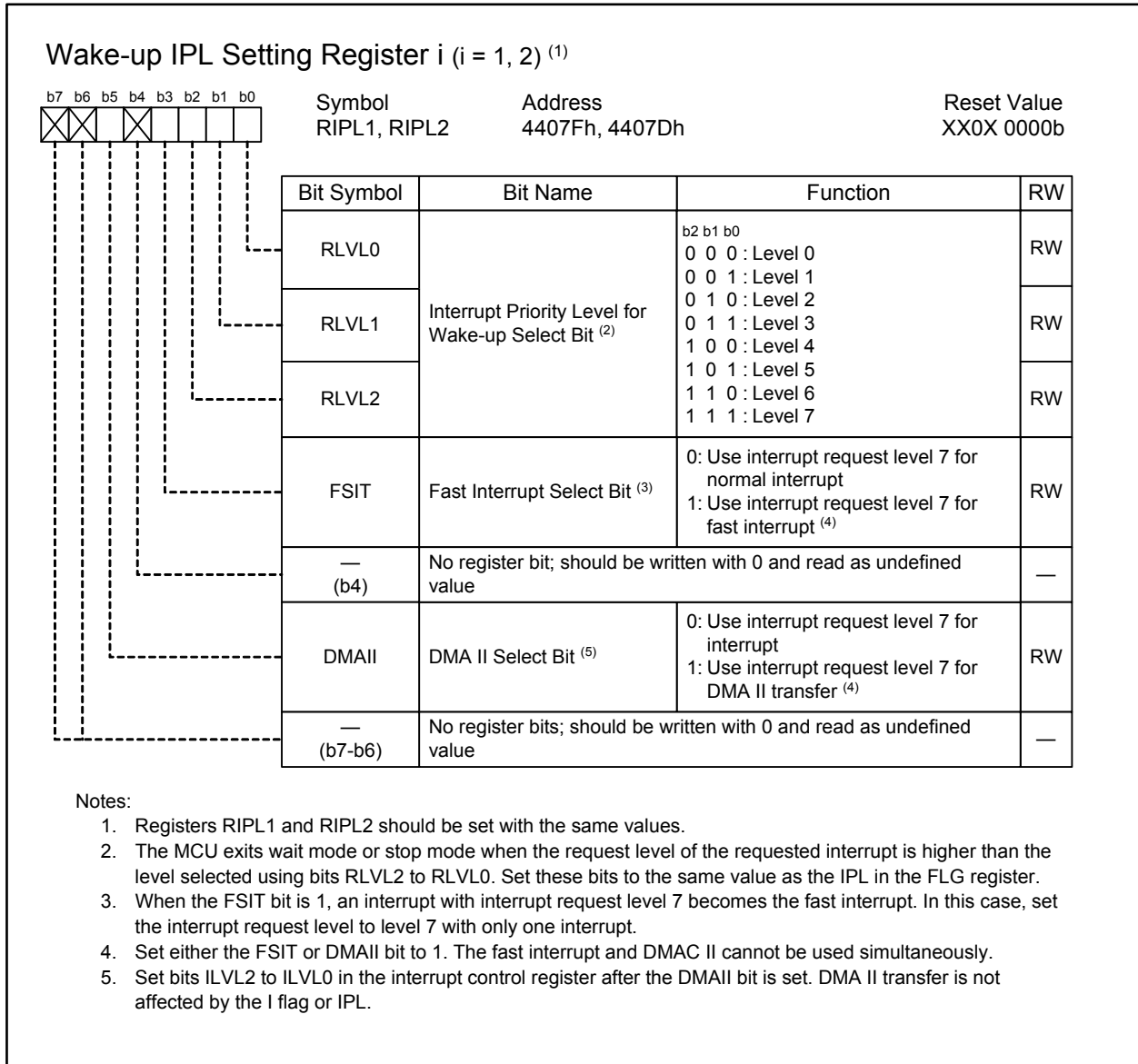


Figure 10.5 Registers RIPL1 and RIPL2

### 10.6.4 Interrupt Sequence

An interrupt sequence is performed from when an interrupt request has been accepted until the interrupt handler starts.

When an interrupt request is generated while an instruction is being executed, the requested interrupt is evaluated in the priority resolver after the current instruction is completed, and the interrupt sequence starts from the next cycle.

However, for instructions RMPA, SCMPU, SIN, SMOVB, SMOVF, SMOVU, SOUT, SSTR, SUNTIL, and SWHILE, when an interrupt request is generated while an instruction is being executed, the current instruction is suspended, and the interrupt sequence starts.

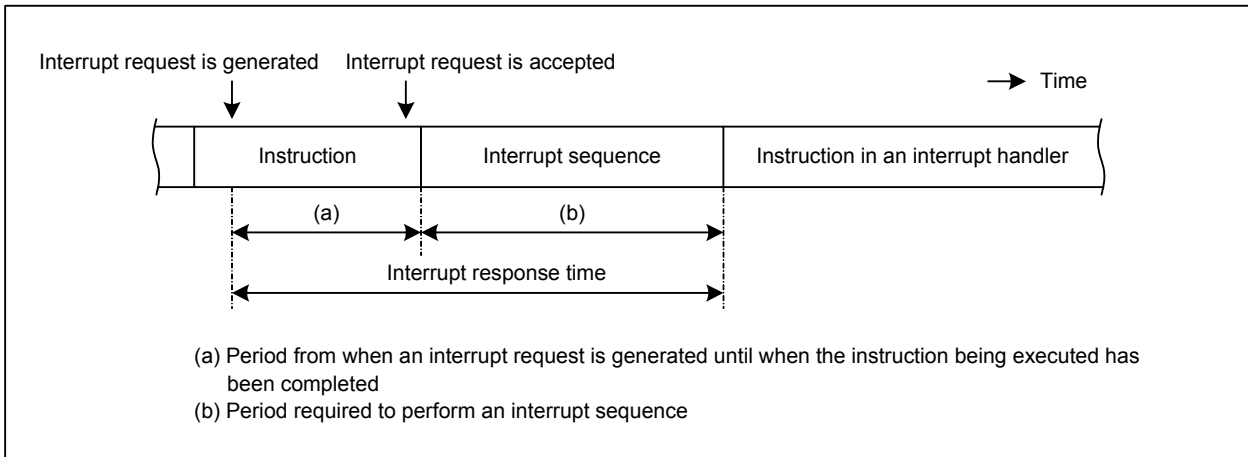
The interrupt sequence is as follows:

- (1) The CPU acknowledges the interrupt request to obtain the interrupt information (the interrupt number, and the interrupt request level) from the interrupt controller. Then the corresponding IR bit becomes 0 (no interrupt requested).
- (2) The state of the FLG register before the interrupt sequence is stored to a temporary register in the CPU. The temporary register is inaccessible to users.
- (3) The following bits in the FLG register become 0:
  - The I flag (interrupt enable flag): interrupt disabled
  - The D flag (debug flag): single-step interrupt disabled
  - The U flag (stack pointer select flag): ISP selected
- (4) The content of the temporary register in the CPU is saved to the stack, or to the SVF register in case of the fast interrupt.
- (5) The content of the PC is saved to the stack, or to the SVP register in case of the fast interrupt.
- (6) The interrupt request level for the accepted interrupt is set in the IPL (processor interrupt priority level).
- (7) The corresponding interrupt vector is read from the interrupt vector table.
- (8) This interrupt vector is stored into the PC.

After the interrupt sequence is completed, an instruction is executed from the start address of the interrupt handler.

### 10.6.5 Interrupt Response Time

The interrupt response time, as shown in Figure 10.6, consists of two non-overlapping time segments: (a) the period from when an interrupt request is generated until the instruction being executed is completed; and (b) the period required for the interrupt sequence.



**Figure 10.6** Interrupt Response Time

Period (a) varies depending on the instruction being executed. Instructions, such as LDCTX and STCTX in which registers are sequentially saved into or restored from the stack, require the longest time. For example, the STCTX instruction requires at least 30 cycles for 10 registers to be saved. It requires more time if the WAIT instruction is in the stack.

Period (b) is listed in Table 10.7.

**Table 10.7** Interrupt Sequence Execution Time <sup>(1)</sup>

Interrupt	Execution Time in Terms of CPU Clock
Peripherals	13 + $\alpha$ cycles <sup>(2)</sup>
INT instruction	11 cycles
NMI	10 cycles
Watchdog timer Oscillator stop detection Low voltage detection	11 cycles
Undefined instruction	12 cycles
Overflow	12 cycles
BRK instruction (relocatable vector table)	16 cycles
BRK instruction (fixed vector table)	19 cycles
BRK2 instruction	19 cycles
Fast interrupt	11 cycles

Notes:

- These are the values when the interrupt vectors are aligned to the addresses in multiples of 4 in the internal ROM. However, the condition does not apply to the fast interrupt.
- $\alpha$  is the number of waits to access SFRs minus 2.

### 10.6.6 IPL after Accepting an Interrupt Request

When a peripheral interrupt request is accepted, the interrupt request level is set in the IPL (processor interrupt priority level).

Software interrupts and special interrupts have no interrupt request level. When these interrupt requests are accepted, the value listed in Table 10.8 is set in the IPL as the interrupt request level.

**Table 10.8 Interrupts without Interrupt Request Level and IPL**

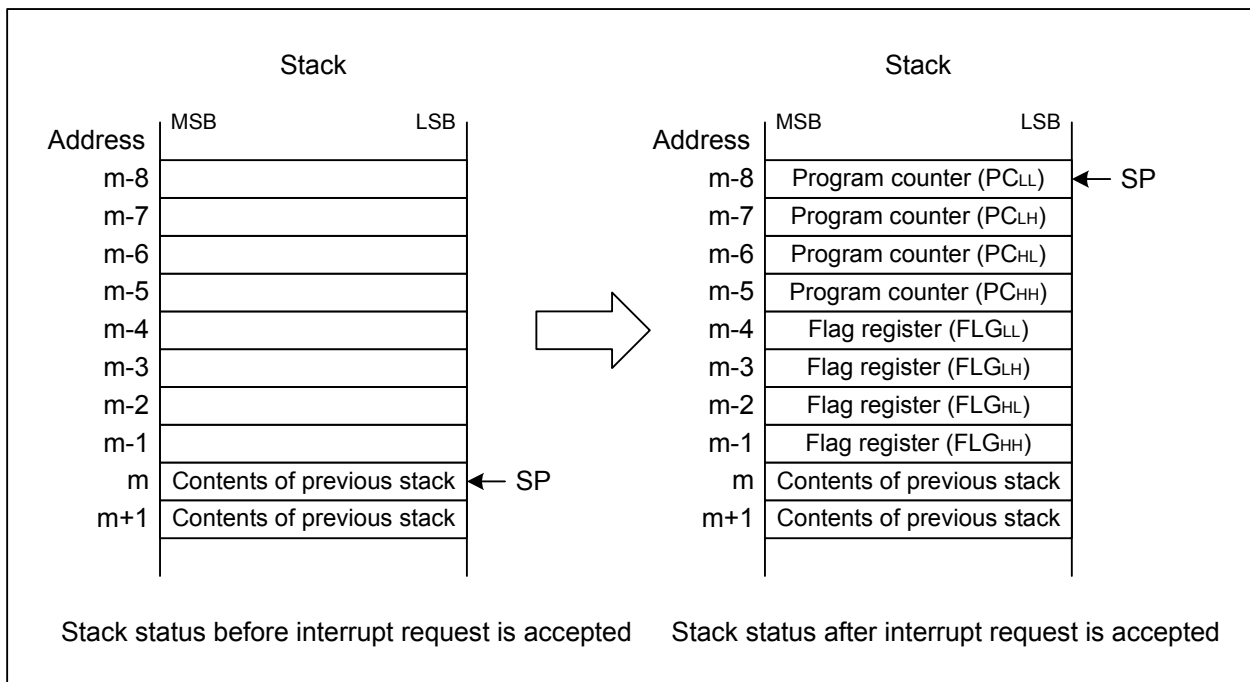
Interrupt Sources without Interrupt Request Level	IPL Value to be Set
NMI, watchdog timer, oscillator stop detection, low voltage detection	7
Reset	0
Software	Unchanged

### 10.6.7 Register Saving

In the interrupt sequence, the FLG register and PC values are saved to the stack, in that order. Figure 10.7 shows the stack status before and after an interrupt request is accepted.

In the fast interrupt sequence, the FLG register and PC values are saved to registers SVF and SVP, respectively.

If there are any other registers to be saved to the stack, save them at the beginning of the interrupt handler. A single PUSHM instruction saves all registers except the frame base register (FB) and stack pointer (SP).



**Figure 10.7 Stack Status Before and After an Interrupt Request is Accepted**

## 10.7 Register Restoring from Interrupt Handler

When the REIT instruction is executed at the end of the interrupt handler, the FLG register and PC values, which are saved in the stack, are restored, and the program resumes the operation that was interrupted. In the fast interrupt, execute the FREIT instruction to restore them from the save registers, instead.

To restore the register values which are saved by software in the interrupt handler, use an instruction such as POPM before the REIT or FREIT instruction.

If the register bank is switched in the interrupt handler, the bank is automatically switched back to the original register bank by the REIT or FREIT instruction.

## 10.8 Interrupt Priority

If two or more interrupt requests are detected at an interrupt request sampling point, the interrupt request with higher priority is accepted.

For maskable interrupts (peripheral interrupts), the interrupt request level select bits (bits ILVL2 to ILVL0) select a request level. If two or more interrupt requests have the same request level, the interrupt with higher priority, predetermined by hardware, is accepted.

The priorities of the reset and special interrupts, such as the watchdog timer interrupt, are determined by the hardware. Note that the reset has the highest priority. The following is the priority order determined by the hardware:

Watchdog timer  
Reset > Oscillator stop detection > NMI > Peripherals  
Low voltage detection

Software interrupts are not governed by priority. A jump to the interrupt handler takes place whenever the relevant instruction is executed.

## 10.9 Priority Resolver

The priority resolver selects an interrupt that has the highest priority among requested interrupts detected at the same sampling point.

Figure 10.8 shows the priority resolver.



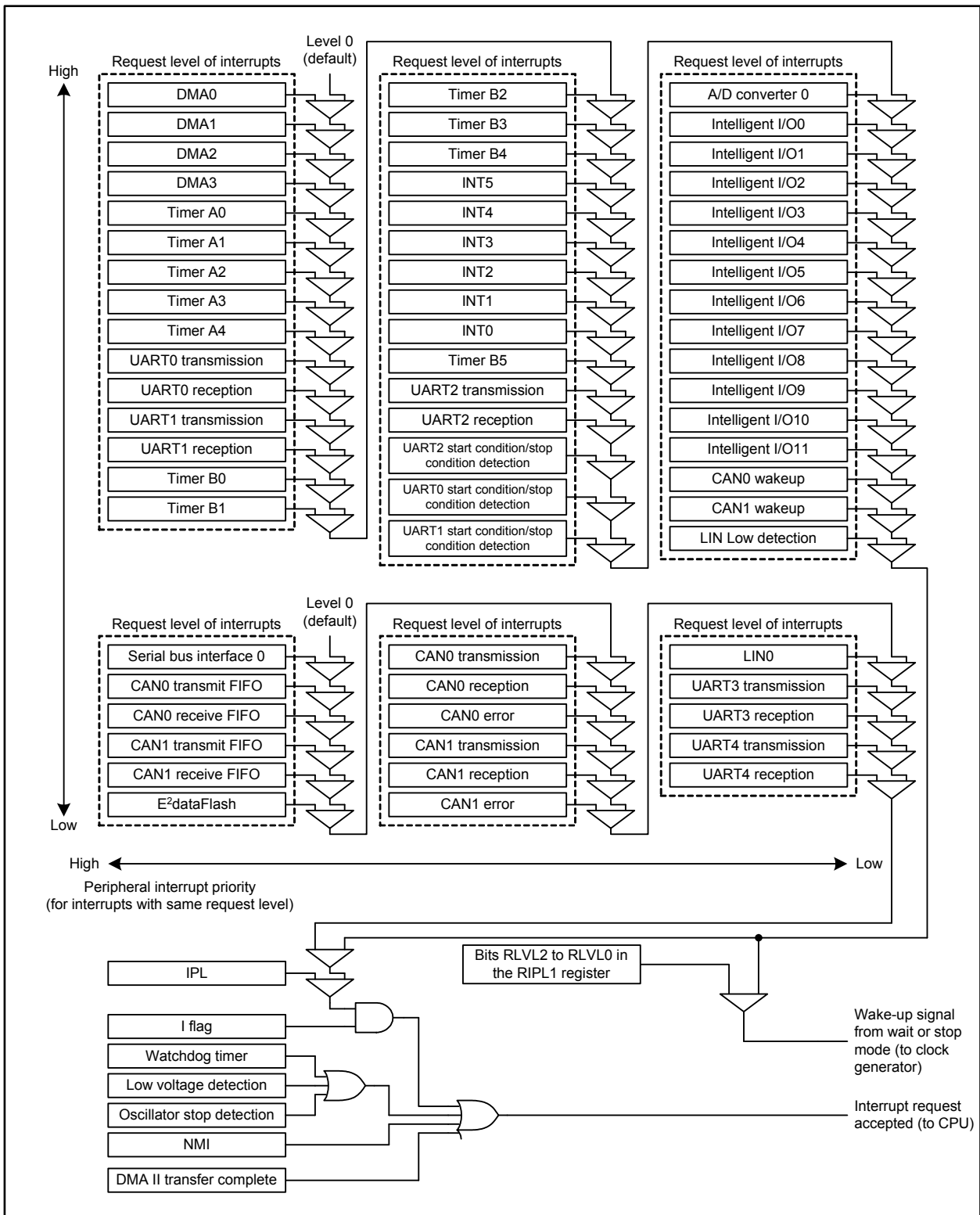


Figure 10.8 Priority Resolver

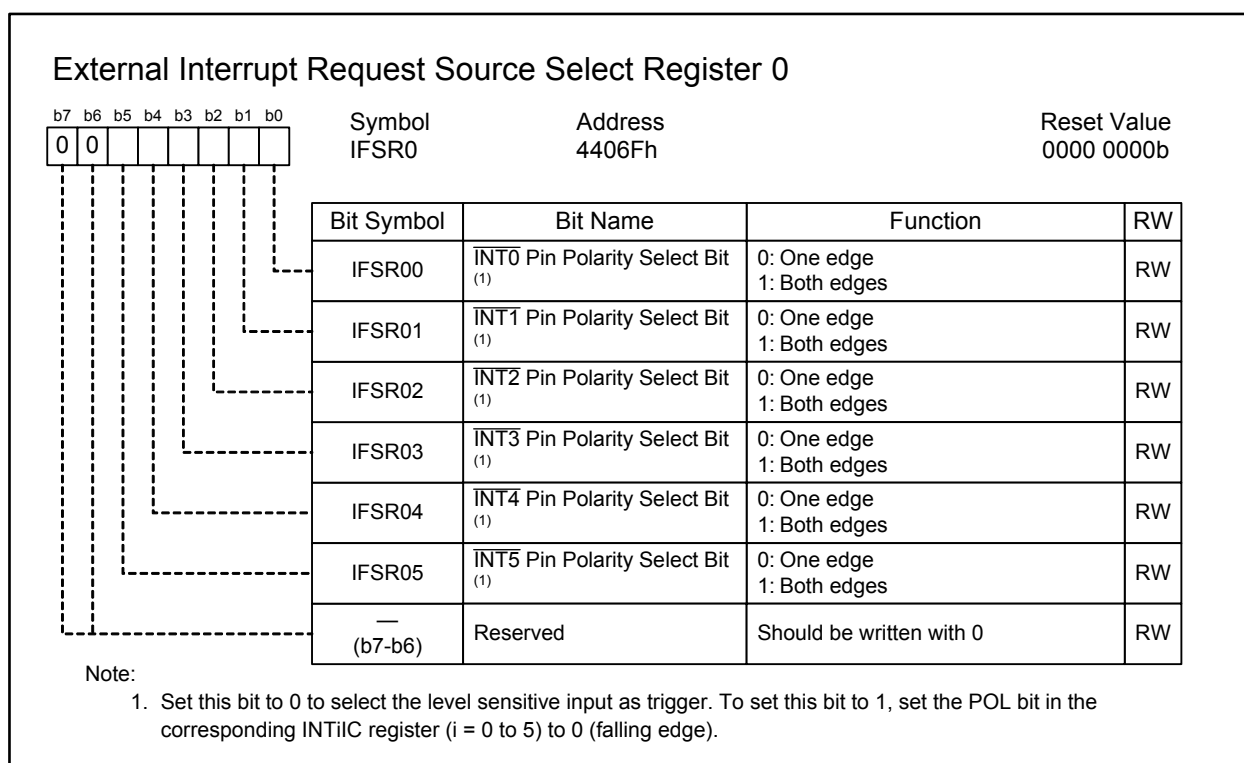
## 10.10 External Interrupt

An external interrupt occurs by an external input applied to the  $\overline{\text{INT}}_i$  pin ( $i = 0$  to 5). Set the LVS bit in the INTiC register to select whether an interrupt is triggered by the effective edge(s) (edge sensitive), or by the effective level (level sensitive) of the input signal. The polarity of the input signal is selected by setting the POL bit in the same register.

When using edge-triggered interrupts, setting the IFSR0i bit in the IFSR0 register to 1 (both edges) causes interrupt requests to be generated on both rising and falling edges of the external input. Set the POL bit in the corresponding register to 0 (falling edge) to set the IFSR0i bit to 1.

When using level-triggered interrupts, set the IFSR0i bit to 0 (one edge). When an effective level, which is selected by the POL bit, is detected on the  $\overline{\text{INT}}_i$  pin, the IR bit in the INTiC register becomes 1. The IR bit remains unchanged until the INTi interrupt is accepted, or set to 0 by a program, even if the signal level at the  $\overline{\text{INT}}_i$  pin changes.

Figure 10.9 shows the IFSR0 register.



**Figure 10.9 IFSR0 Register**

The external interrupt signal has a digital filtering function for noise reduction. This function enables the interrupt controller to sample input signals with a selected clock and to pass pulses only which match the level three times sequentially.

Figures 10.10 and 10.11 show registers INTF0 and INTF1.

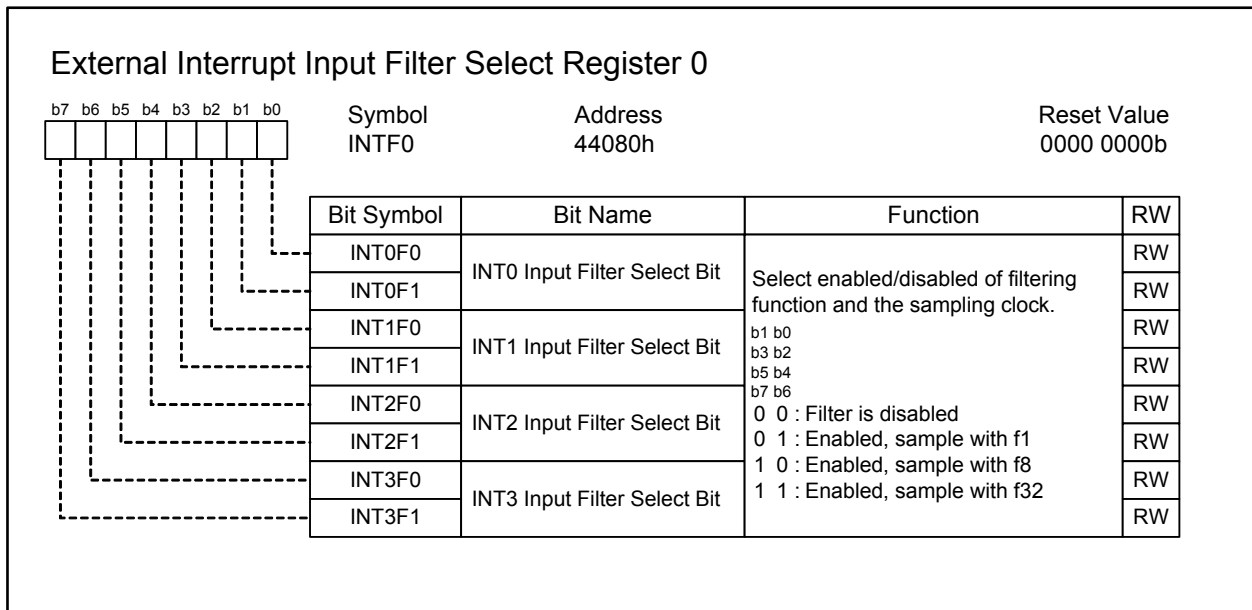


Figure 10.10 INTF0 Register

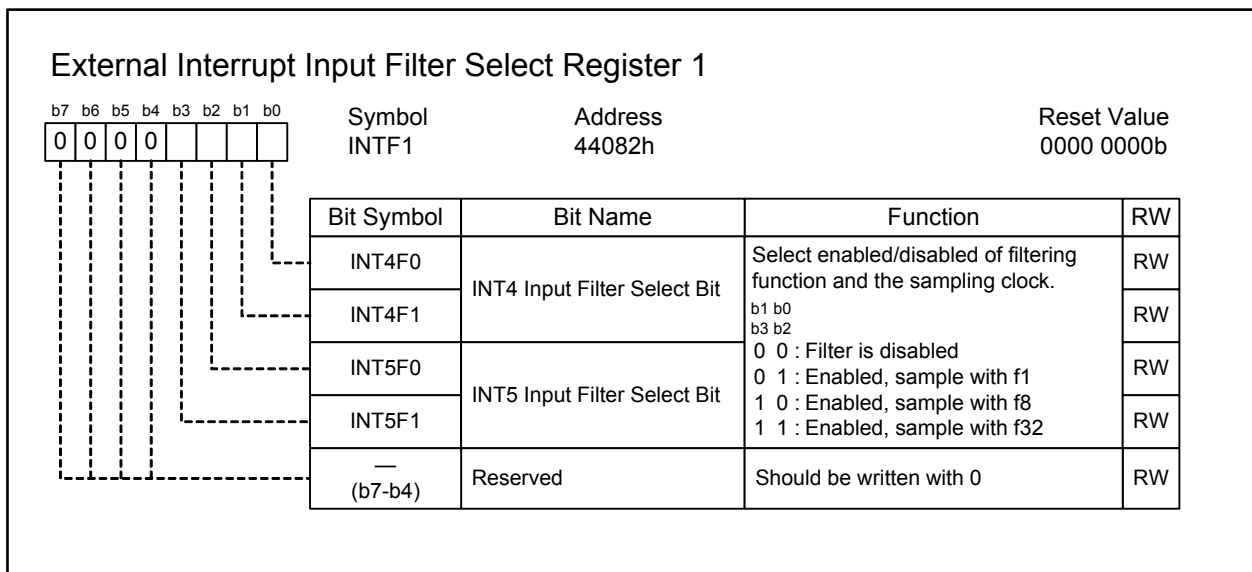


Figure 10.11 INTF1 Register

## 10.11 NMI

The NMI (non maskable interrupt) occurs when an input signal at the  $\overline{\text{NMI}}$  pin switches from high to low. This non maskable interrupt is disabled after a reset. To enable this interrupt, set the PM24 bit in the PM2 register to 1 after setting the interrupt stack pointer (ISP) at the beginning of the program. The  $\overline{\text{NMI}}$  pin shares a pin with port P8\_5, which enables the P8\_5 bit in the P8 register to indicate the input level at the  $\overline{\text{NMI}}$  pin.

Note:

1. When not using the NMI, do not change the reset value of the PM24 bit in the PM2 register.

## 10.12 Intelligent I/O Interrupt

The intelligent I/O interrupt is assigned to software interrupt numbers 44 to 55.

Figure 10.12 shows a block diagram of the intelligent I/O interrupt. Figures 10.13 and 10.14 show registers IIOiIR and IIOiIE, respectively ( $i = 0$  to 11).

To use the intelligent I/O interrupt, set the IRLT bit in the IIOiIE register to 1 (interrupt requests used for interrupt).

The intelligent I/O interrupt has multiple request sources. When an interrupt request is generated with an intelligent I/O function, the corresponding bit in the IIOiIR register becomes 1 (interrupt requested). If the corresponding bit in the IIOiIE register is 1 (interrupt enabled), the IR bit in the corresponding IIOiC register changes to 1 (interrupt requested).

After the IR bit setting changes from 0 to 1, it remains unchanged if a bit in the IIOiIR register becomes 1 by another interrupt request source and the corresponding bit in the IIOiIE register is 1.

Bits in the IIOiIR register do not become 0 even if an interrupt is accepted. They should be set to 0 by either the AND or BCLR instruction. Note that every generated interrupt request is ignored until these bits are set to 0.

To use the intelligent I/O interrupt as a DMAC II trigger, set the IRLT bit in the IIOiIE register to 0 (interrupt requests used for DMA or DMA II) and the bit used for the interrupt source to 1 (interrupt enabled) in the IIOiIE register.

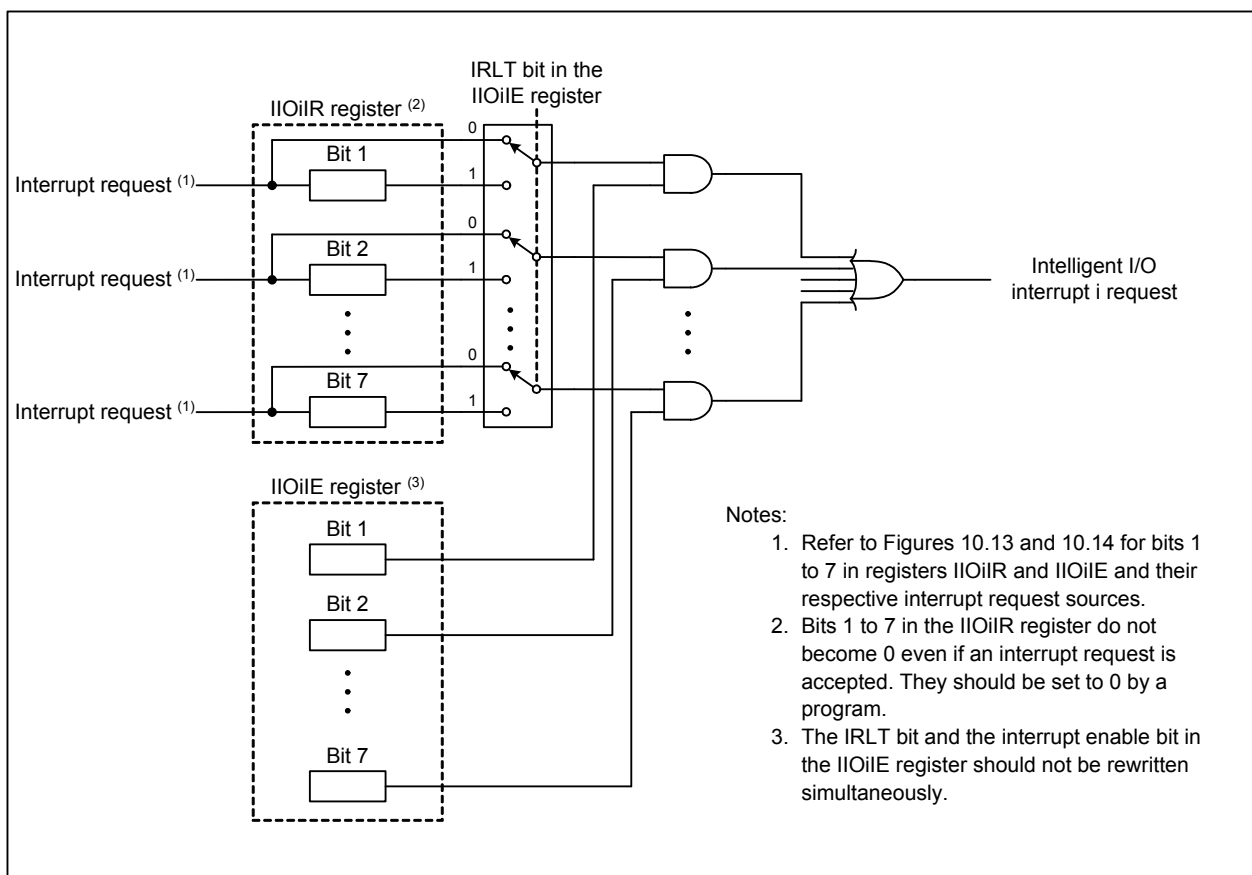


Figure 10.12 Intelligent I/O Interrupt Block Diagram ( $i = 0$  to 11)

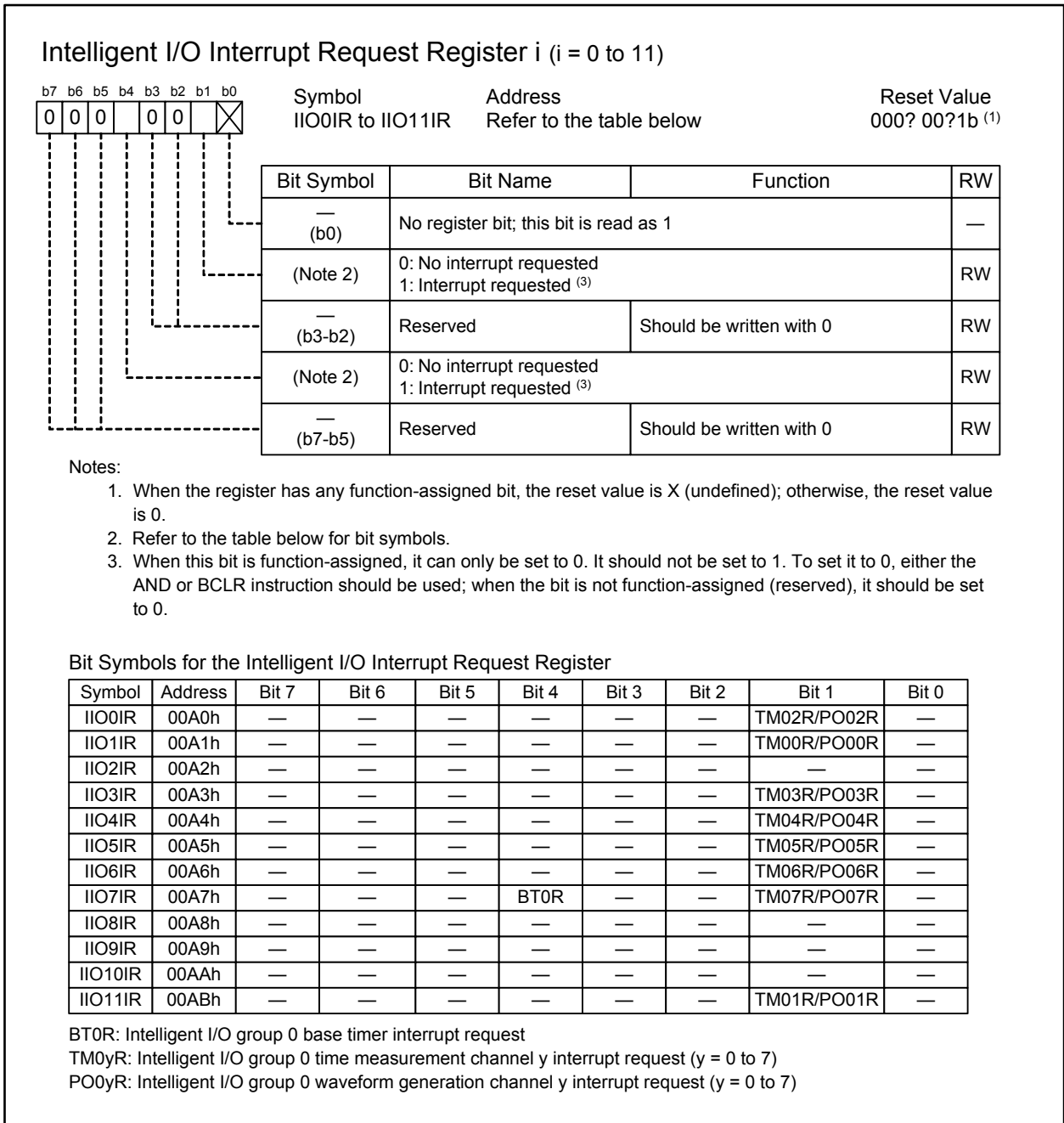


Figure 10.13 Registers IIO0IR to IIO11IR

## Intelligent I/O Interrupt Enable Register i (i = 0 to 11)

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

Symbol  
IIO0IE to IIO11IE

Address  
Refer to the table below.

Reset Value  
0000 0000b

Bit Symbol	Bit Name	Function	RW
IRLT	Interrupt Request Select Bit <sup>(2)</sup>	0: Use interrupt requests for DMA or DMA II 1: Use interrupt requests for interrupt	RW
(Note 1)		0: Disable the interrupt of bit 1 in the IIOiIR register 1: Enable the interrupt of bit 1 in the IIOiIR register	RW
— (b3-b2)	Reserved	Should be written with 0	RW
(Note 1)		0: Disable the interrupt of bit 4 in the IIOiIR register 1: Enable the interrupt of bit 4 in the IIOiIR register	RW
— (b7-b5)	Reserved	Should be written with 0	RW

## Notes:

1. Refer to the table below for bit symbols.
2. To use interrupt requests for interrupt, the IRLT bit should be set to 1, then bits 1 and 4 should be set to 1.

## Bit Symbols for the Intelligent I/O Interrupt Enable Register

Symbol	Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IIO0IE	00B0h	—	—	—	—	—	—	TM02E/PO02E	IRLT
IIO1IE	00B1h	—	—	—	—	—	—	TM00E/PO00E	IRLT
IIO2IE	00B2h	—	—	—	—	—	—	—	IRLT
IIO3IE	00B3h	—	—	—	—	—	—	TM03E/PO03E	IRLT
IIO4IE	00B4h	—	—	—	—	—	—	TM04E/PO04E	IRLT
IIO5IE	00B5h	—	—	—	—	—	—	TM05E/PO05E	IRLT
IIO6IE	00B6h	—	—	—	—	—	—	TM06E/PO06E	IRLT
IIO7IE	00B7h	—	—	—	BT0E	—	—	TM07E/PO07E	IRLT
IIO8IE	00B8h	—	—	—	—	—	—	—	IRLT
IIO9IE	00B9h	—	—	—	—	—	—	—	IRLT
IIO10IE	00BAh	—	—	—	—	—	—	—	IRLT
IIO11IE	00BBh	—	—	—	—	—	—	TM01E/PO01E	IRLT

BT0E: Intelligent I/O group 0 base timer interrupt enabled

TM0yE: Intelligent I/O group 0 time measurement channel y interrupt enabled (y = 0 to 7)

PO0yE: Intelligent I/O group 0 waveform generation channel y interrupt enabled (y = 0 to 7)

Figure 10.14 Registers IIO0IE to IIO11IE

## 10.13 Notes on Interrupts

### 10.13.1 ISP Setting

The interrupt stack pointer (ISP) is initialized to 00000000h after a reset. Set a value to the ISP before an interrupt is accepted, otherwise the program may go out of control. A multiple of 4 should be set to the ISP, which enables faster interrupt sequence due to less memory access.

When using NMI, in particular, since this interrupt cannot be disabled, set the PM24 bit in the PM2 register to 1 (NMI enabled) after setting the ISP at the beginning of the program.

### 10.13.2 NMI

- NMI cannot be disabled once the PM24 bit in the PM2 register is set to 1 (NMI enabled). This bit setting should be done only when using NMI.
- When the PM24 bit in the PM2 register is 1 (NMI enabled), the P8\_5 bit in the P8 register is enabled just for monitoring the  $\overline{\text{NMI}}$  pin state. It is not enabled as a general port.

### 10.13.3 External Interrupts

- The input signal to the  $\overline{\text{INTi}}$  pin requires the pulse width specified in the electrical characteristics (i = 0 to 5). If the pulse width is narrower than the specification, an external interrupt may not be accepted.
- When the effective level or edge of the  $\overline{\text{INTi}}$  pin (i = 0 to 5) is changed by the following bits: bits POL, LVS in the INTiIC register, the IFSR0i bit (i = 0 to 5) in the IFSR0 register, the corresponding IR bit may become 1 (interrupt requested). When setting the above mentioned bits, preset bits ILVL2 to ILVL0 in the INTiIC register to 000b (interrupt disabled). After setting the above mentioned bits, set the corresponding IR bit to 0 (no interrupt requested), then rewrite bits ILVL2 to ILVL0.

## 11. Watchdog Timer

The watchdog timer is used to detect program runaway. The 15-bit watchdog counter decrements with the cycle which is the peripheral bus clock frequency or on-chip oscillator clock frequency divided by the prescaler.

Select either an interrupt request or a reset with the CM06 bit in the CM0 register for when the watchdog timer underflows. Once the CM06 bit is set to 1 (reset), it cannot be changed to 0 (watchdog timer interrupt) by a program. It can be set to 0 only by a reset.

The watchdog timer has two prescalers. One is the on-chip oscillator clock divided by 1, 2, 4 or 8; the other is the peripheral bus clock divided by 16 or 128. To select the divide ratio for the former, set bits WDK3 and WDK2 in the WDK register. To select the divide ratio for the latter, set the WDC7 bit in the WDC register.

The count source for the watchdog timer is set by the PM22 bit in the PM2 register. When the peripheral bus clock is selected as the count source, the watchdog timer is stopped in wait mode, stop mode, or when the HOLD signal is driven low. It resumes counting from the value held when exiting the mode or state. When the on-chip oscillator clock is selected, the watchdog timer does not stop.

The general formula to calculate a watchdog timer period is:

$$\text{Watchdog timer period} = \frac{\text{Prescaler divisor (16 or 128)} \times 32768}{\text{Peripheral bus clock frequency}}$$

or

$$\text{Watchdog timer period} = \frac{\text{Prescaler divisor (1, 2, 4, or 8)} \times 2048}{\text{On-chip oscillator clock frequency}}$$

For example, when the peripheral bus clock is selected as the count source and it is 1/2 of 48 MHz CPU clock and the prescaler has a divide-by-16 operation, the watchdog timer period is approximately 21.8 ms. Depending on the timing of when a value is written to the WDTS register, a marginal error of one prescaler output cycle (maximum) may occur in the watchdog timer period.

The watchdog timer is initialized when a write operation to the WDTS register is performed or when a watchdog timer interrupt request is generated. The prescaler is initialized only when the MCU is reset. After a reset, the watchdog timer starts counting automatically if the OFS area of the flash memory are preset. When the WDTON bit in the OFS area is 1, both the watchdog timer and the prescaler are stopped. They start counting when a write operation to the WDTS register is performed. When the WDTON bit is 0, both the watchdog timer and the prescaler automatically start counting after a reset.

Figure 11.1 shows a block diagram of the watchdog timer. Figures 11.2 to 11.5 show registers associated with the watchdog timer.



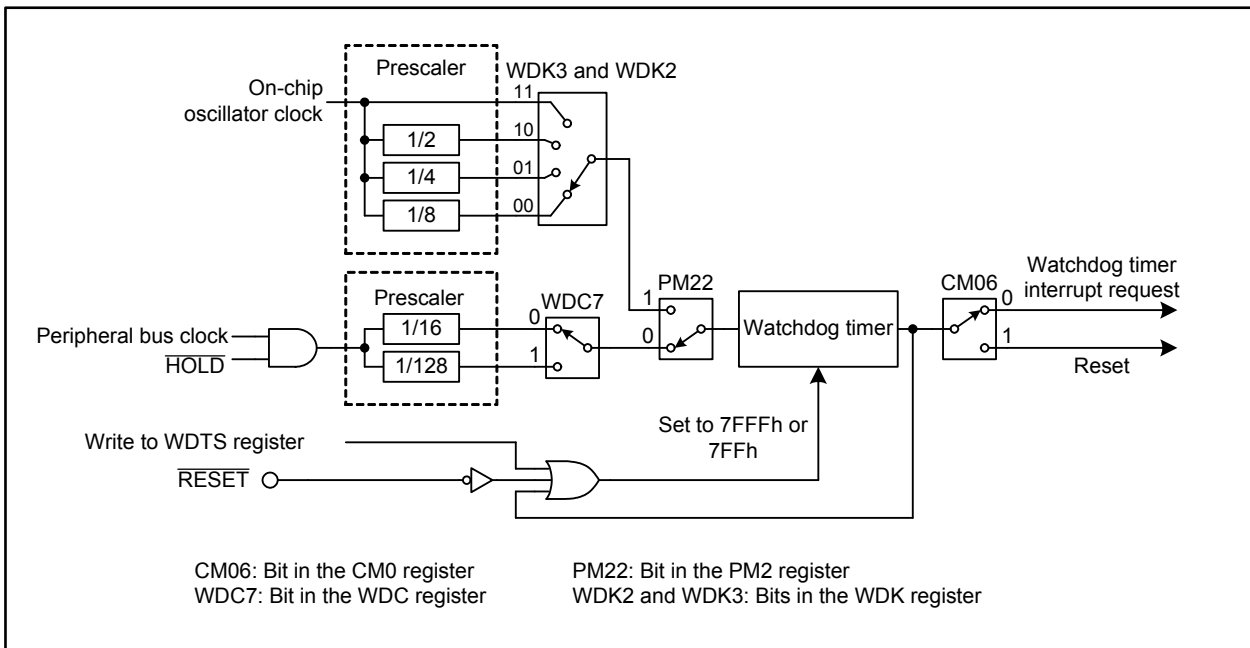


Figure 11.1 Watchdog Timer Block Diagram

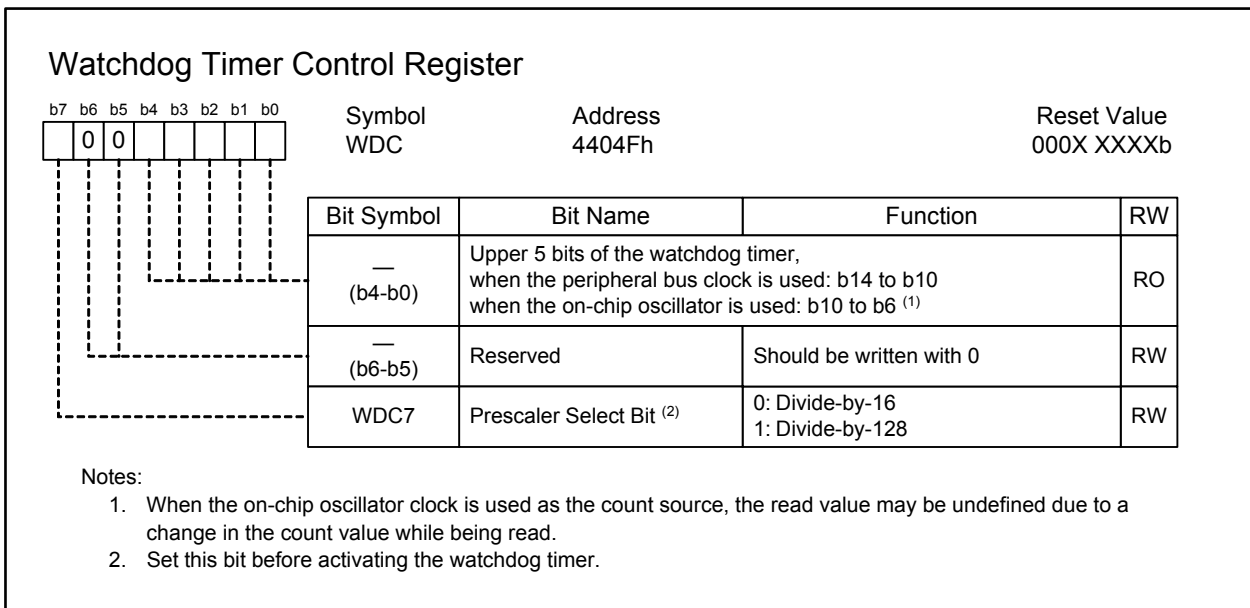


Figure 11.2 WDC Register

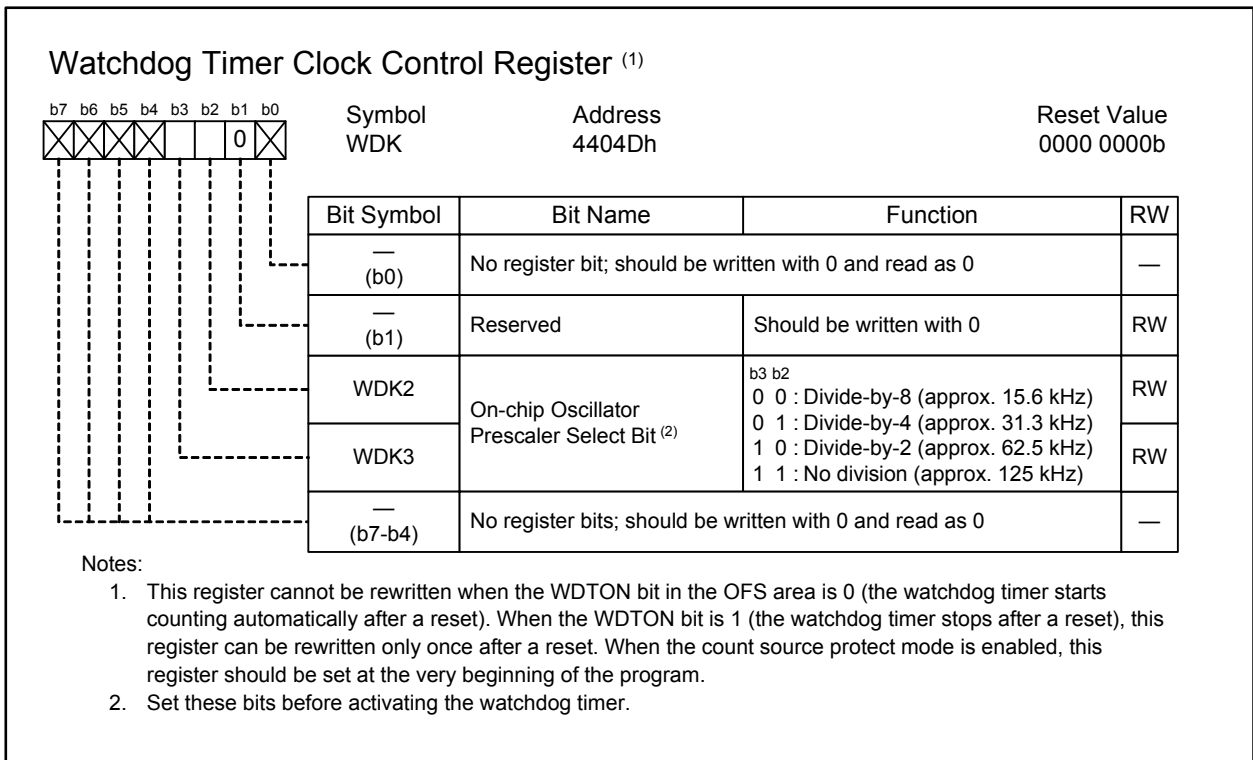


Figure 11.3 WDK Register

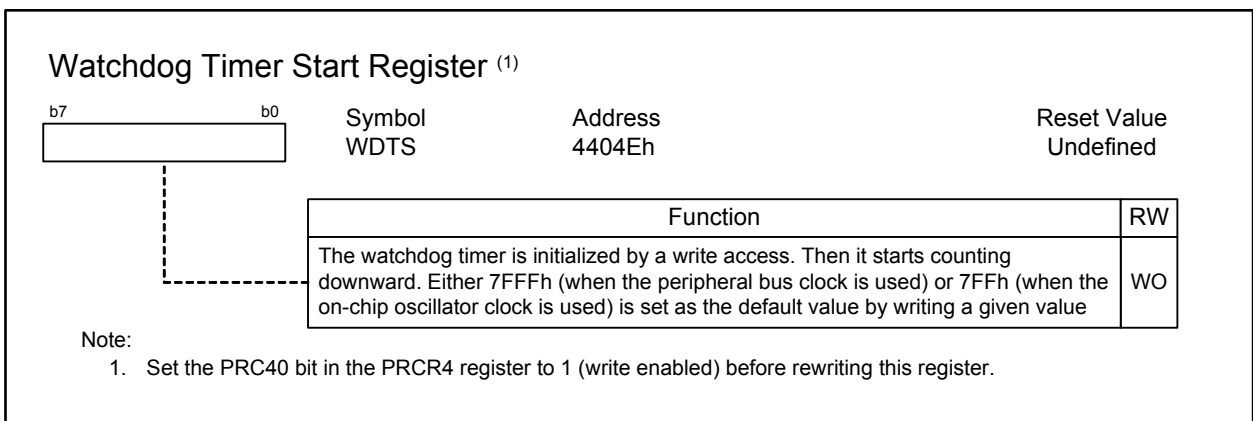


Figure 11.4 WDTS Register

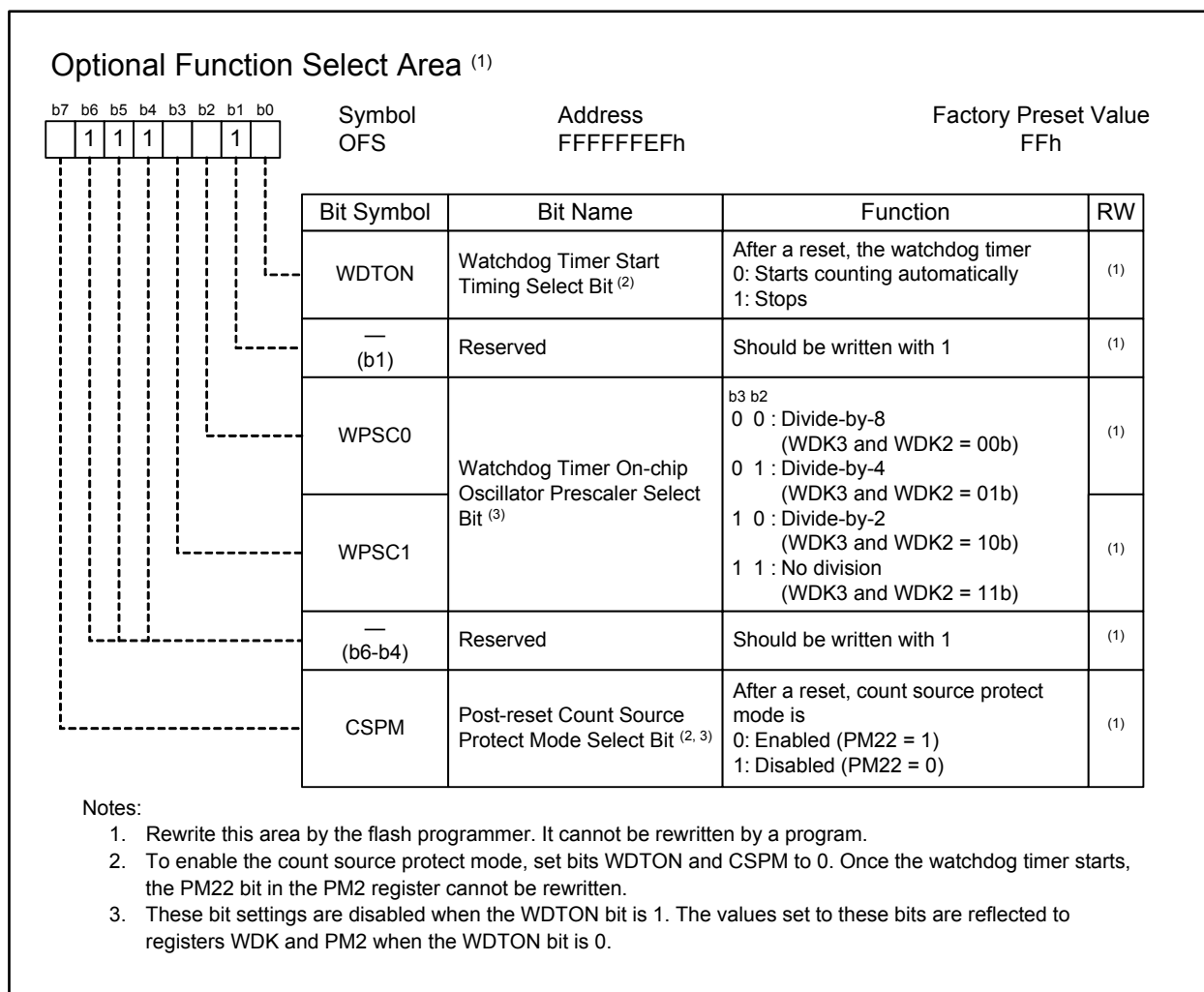


Figure 11.5 OFS Area

## 12. DMAC

Direct memory access (DMA) is a system that can control data transfer without using a CPU instruction. The R32C/100 Series' four channel DMA controller (DMAC) transmits 8-bit (byte), 16-bit (word), or 32-bit (long word) data in cycle-steal mode from a source address to a destination address each time a transfer request is generated.

The DMAC, which shares a data bus with the CPU, has a higher bus access priority than the CPU. This allows the DMAC to perform fast data transfer when a transfer request is generated.

Figure 12.1 shows a map of the CPU-internal registers associated with DMAC. Table 12.1 lists DMAC specifications. Figures 12.2 to 12.10 show registers associated with DMAC. Since the registers shown in Figure 12.1 are allocated in the CPU, the LDC or STC instruction should be used to write to the registers.

DMAC-associated Registers	
DMD0	DMA0 mode register
DMD1	DMA1 mode register
DMD2	DMA2 mode register
DMD3	DMA3 mode register
DCT0	DMA0 terminal count register
DCT1	DMA1 terminal count register
DCT2	DMA2 terminal count register
DCT3	DMA3 terminal count register
DCR0	DMA0 terminal count reload register <sup>(1)</sup>
DCR1	DMA1 terminal count reload register <sup>(1)</sup>
DCR2	DMA2 terminal count reload register <sup>(1)</sup>
DCR3	DMA3 terminal count reload register <sup>(1)</sup>
DSA0	DMA0 source address register
DSA1	DMA1 source address register
DSA2	DMA2 source address register
DSA3	DMA3 source address register
DSR0	DMA0 source address reload register <sup>(1)</sup>
DSR1	DMA1 source address reload register <sup>(1)</sup>
DSR2	DMA2 source address reload register <sup>(1)</sup>
DSR3	DMA3 source address reload register <sup>(1)</sup>
DDA0	DMA0 destination address register
DDA1	DMA1 destination address register
DDA2	DMA2 destination address register
DDA3	DMA3 destination address register
DDR0	DMA0 destination address reload register <sup>(1)</sup>
DDR1	DMA1 destination address reload register <sup>(1)</sup>
DDR2	DMA2 destination address reload register <sup>(1)</sup>
DDR3	DMA3 destination address reload register <sup>(1)</sup>

Note:

1. This register is used for repeat transfer, not for single transfer.

**Figure 12.1 CPU-internal Registers for DMAC**

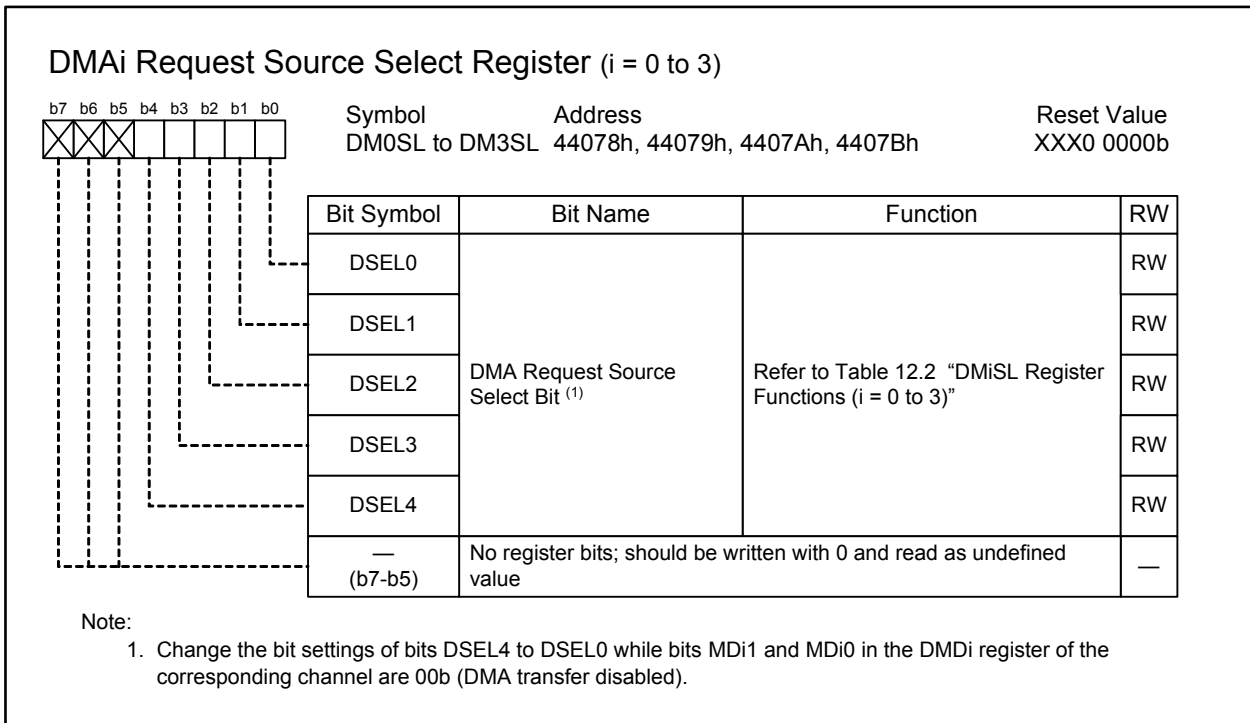
**Table 12.1 DMAC Specifications (i = 0 to 3)**

Item		Specification
Channels		4
Bus request mode		Cycle-steal mode
Transfer memory spaces		From a given address in a 64-Mbyte space (00000000h to 01FFFFFFh and FE000000h to FFFFFFFFh) to another given address in the same space
Maximum transfer bytes		64-Mbytes (when 32-bit data is transferred), 32-Mbytes (when 16-bit data is transferred), 16-Mbytes (when 8-bit data is transferred)
DMA request sources <sup>(1)</sup>		Falling edge or both edges of signals applied to pins INTO to INT3 Interrupt requests from timers A0 to A4 Interrupt requests from timers B0 to B5 Transmit/receive interrupt requests from UART0 to UART4 A/D conversion interrupt requests Intelligent I/O interrupt requests Serial bus interface interrupt requests Software trigger
Channel priority		DMA0 > DMA1 > DMA2 > DMA3 (DMA0 has the highest priority)
Transfer sizes		8 bits, 16 bits, or 32 bits
Addressing modes		Incrementing addressing or non-incrementing addressing
Transfer modes	Single transfer	Transfer is completed when the DCTi register becomes 00000000h
	Repeat transfer	When the DCTi register becomes 00000000h, the value of the DCRi register is reloaded into the DCTi register to continue the DMA transfer
DMA transfer complete interrupt request generation timing		When the DCTi register changes from 00000001h to 00000000h
DMA transfer start-up	Single transfer	When a DMA transfer request is generated after the DCTi register is set to a value other than 00000000h and bits MDi1 and MDi0 in the DMDi register are set to 01b (single transfer)
	Repeat transfer	When a DMA transfer request is generated after the DCTi register is set to a value other than 00000000h and bits MDi1 and MDi0 are set to 11b (repeat transfer)
DMA transfer stop	Single transfer	When bits MDi1 and MDi0 are set to 00b (DMA transfer disabled)
	Repeat transfer	When bits MDi1 and MDi0 are set to 00b (DMA transfer disabled)
Reload timing to DCTi, DSAi, or DDAi register		When the DCTi register changes from 00000001h to 00000000h in repeat transfer mode
Minimum DMA transfer cycles		3

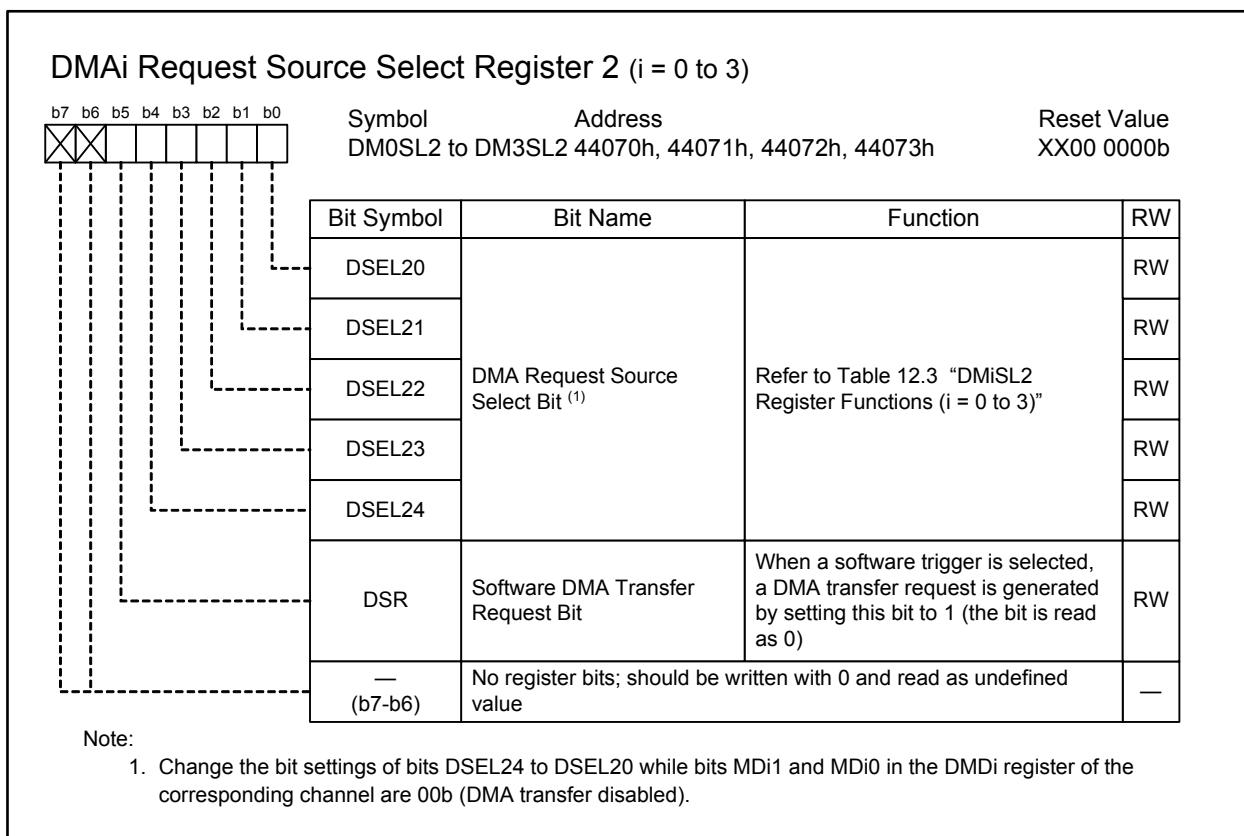
Note:

1. DMA transfer does not affect any interrupts.

The DMA transfer request is available by two different sources: software and hardware. More concretely, they are a write access to the DSR bit in the DMiSL2 register and an interrupt request output from a function specified in bits DSEL4 to DSEL0 in the DMiSL register, and in bits DSEL24 to DSEL20 in the DMiSL2 register ( $i = 0$  to 3). Unlike interrupt requests, the DMA transfer request is not affected by the I flag or the interrupt control register. Therefore this request can be accepted even when interrupts are disabled. Since the DMA transfer does not affect any interrupts, either, the IR bit in the interrupt control register is not changed by the DMA transfer.



**Figure 12.2 Registers DM0SL to DM3SL**



**Figure 12.3 Registers DM0SL2 to DM3SL2**

**Table 12.2 DMiSL Register Functions (i = 0 to 3)**

Setting Value b4 b3 b2 b1 b0	DMA Request Source			
	DMA0	DMA1	DMA2	DMA3
0 0 0 0 0	Select from DMiSL2 register			
0 0 0 0 1	Falling edge of $\overline{\text{INT0}}$ <sup>(1)</sup>	Falling edge of $\overline{\text{INT1}}$ <sup>(1)</sup>	Falling edge of $\overline{\text{INT2}}$ <sup>(1)</sup>	Falling edge of $\overline{\text{INT3}}$ <sup>(1)</sup>
0 0 0 1 0	Both edges of $\overline{\text{INT0}}$ <sup>(1)</sup>	Both edges of $\overline{\text{INT1}}$ <sup>(1)</sup>	Both edges of $\overline{\text{INT2}}$ <sup>(1)</sup>	Both edges of $\overline{\text{INT3}}$ <sup>(1)</sup>
0 0 0 1 1	Timer A0 interrupt request			
0 0 1 0 0	Timer A1 interrupt request			
0 0 1 0 1	Timer A2 interrupt request			
0 0 1 1 0	Timer A3 interrupt request			
0 0 1 1 1	Timer A4 interrupt request			
0 1 0 0 0	Timer B0 interrupt request			
0 1 0 0 1	Timer B1 interrupt request			
0 1 0 1 0	Timer B2 interrupt request			
0 1 0 1 1	Timer B3 interrupt request			
0 1 1 0 0	Timer B4 interrupt request			
0 1 1 0 1	Timer B5 interrupt request			
0 1 1 1 0	UART0 transmit interrupt request			
0 1 1 1 1	UART0 receive interrupt request or ACK interrupt request <sup>(2)</sup>			
1 0 0 0 0	UART1 transmit interrupt request			
1 0 0 0 1	UART1 receive interrupt request or ACK interrupt request <sup>(2)</sup>			
1 0 0 1 0	UART2 transmit interrupt request			
1 0 0 1 1	UART2 receive interrupt request or ACK interrupt request <sup>(2)</sup>			
1 0 1 0 0	Reserved			
1 0 1 0 1	Reserved			
1 0 1 1 0	Reserved			
1 0 1 1 1	Reserved			
1 1 0 0 0	A/D0 interrupt request			
1 1 0 0 1	Intelligent I/O interrupt 0 request	Intelligent I/O interrupt 7 request	Intelligent I/O interrupt 2 request	Intelligent I/O interrupt 9 request
1 1 0 1 0	Intelligent I/O interrupt 1 request	Intelligent I/O interrupt 8 request	Intelligent I/O interrupt 3 request	Intelligent I/O interrupt 10 request
1 1 0 1 1	Intelligent I/O interrupt 2 request	Intelligent I/O interrupt 9 request	Intelligent I/O interrupt 4 request	Intelligent I/O interrupt 11 request
1 1 1 0 0	Intelligent I/O interrupt 3 request	Intelligent I/O interrupt 10 request	Intelligent I/O interrupt 5 request	Intelligent I/O interrupt 0 request
1 1 1 0 1	Intelligent I/O interrupt 4 request	Intelligent I/O interrupt 11 request	Intelligent I/O interrupt 6 request	Intelligent I/O interrupt 1 request
1 1 1 1 0	Intelligent I/O interrupt 5 request	Intelligent I/O interrupt 0 request	Intelligent I/O interrupt 7 request	Intelligent I/O interrupt 2 request
1 1 1 1 1	Intelligent I/O interrupt 6 request	Intelligent I/O interrupt 1 request	Intelligent I/O interrupt 8 request	Intelligent I/O interrupt 3 request

**Notes:**

1. The falling edge and both edges of signals applied to the  $\overline{\text{INTi}}$  pin become the DMA request sources (i = 0 to 3). These request sources are not affected by external interrupts (the IFSR0 register and bits POL and LVS in the INTiIC register), and vice versa.
2. Registers UiSMR and UiSMR2 are used to switch between the UARTi receive interrupt and ACK interrupt (i = 0 to 2).



**Table 12.3 DMiSL2 Register Functions (i = 0 to 3)**

Setting Value					DMA Request Source			
b4	b3	b2	b1	b0	DMA0	DMA1	DMA2	DMA3
0	0	0	0	0	Software trigger			
0	0	0	0	1	Reserved			
0	0	0	1	0	Reserved			
0	0	0	1	1	Reserved			
0	0	1	0	0	Serial bus interface 0 interrupt request			
0	0	1	0	1	Reserved			
0	0	1	1	0	Reserved			
0	0	1	1	1	Reserved			
0	1	0	0	0	Reserved			
0	1	0	0	1	Reserved			
0	1	0	1	0	Reserved			
0	1	0	1	1	Reserved			
0	1	1	0	0	Reserved			
0	1	1	0	1	Reserved			
0	1	1	1	0	Reserved			
0	1	1	1	1	Reserved			
1	0	0	0	0	Reserved			
1	0	0	0	1	Reserved			
1	0	0	1	0	Reserved			
1	0	0	1	1	Reserved			
1	0	1	0	0	Reserved			
1	0	1	0	1	Reserved			
1	0	1	1	0	Reserved			
1	0	1	1	1	Reserved			
1	1	0	0	0	UART3 transmit interrupt request			
1	1	0	0	1	UART3 receive interrupt request			
1	1	0	1	0	UART4 transmit interrupt request			
1	1	0	1	1	UART4 receive interrupt request			
1	1	1	0	0	Reserved			
1	1	1	0	1	Reserved			
1	1	1	1	0	Reserved			
1	1	1	1	1	Reserved			

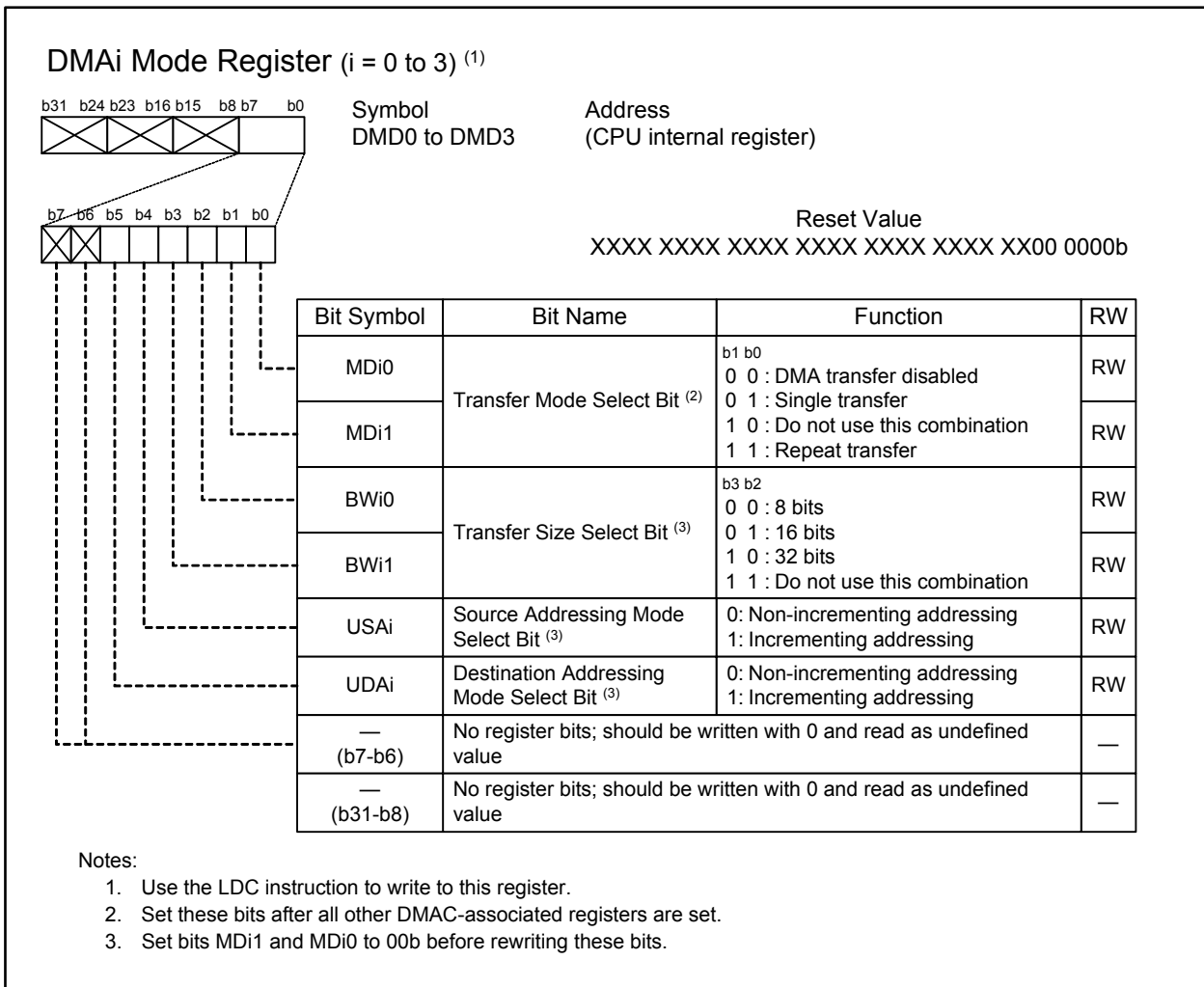


Figure 12.4 Registers DMD0 to DMD3

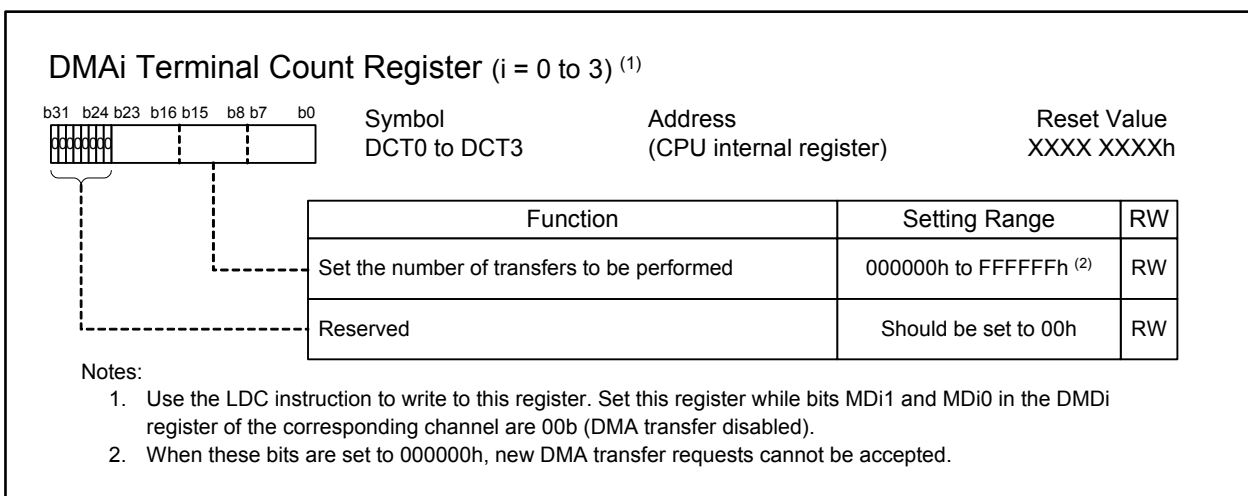


Figure 12.5 Registers DCT0 to DCT3

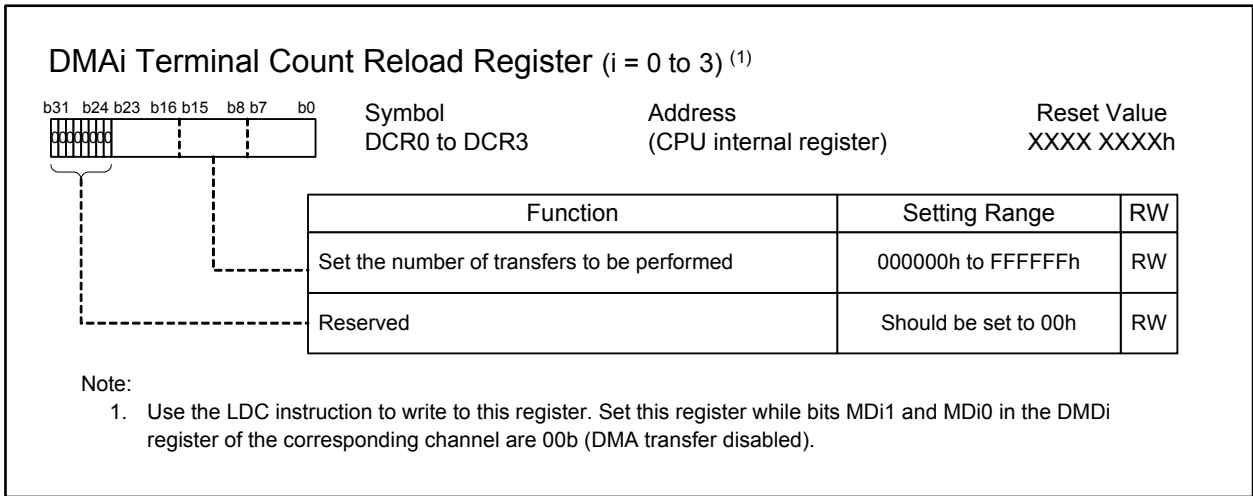


Figure 12.6 Registers DCR0 to DCR3

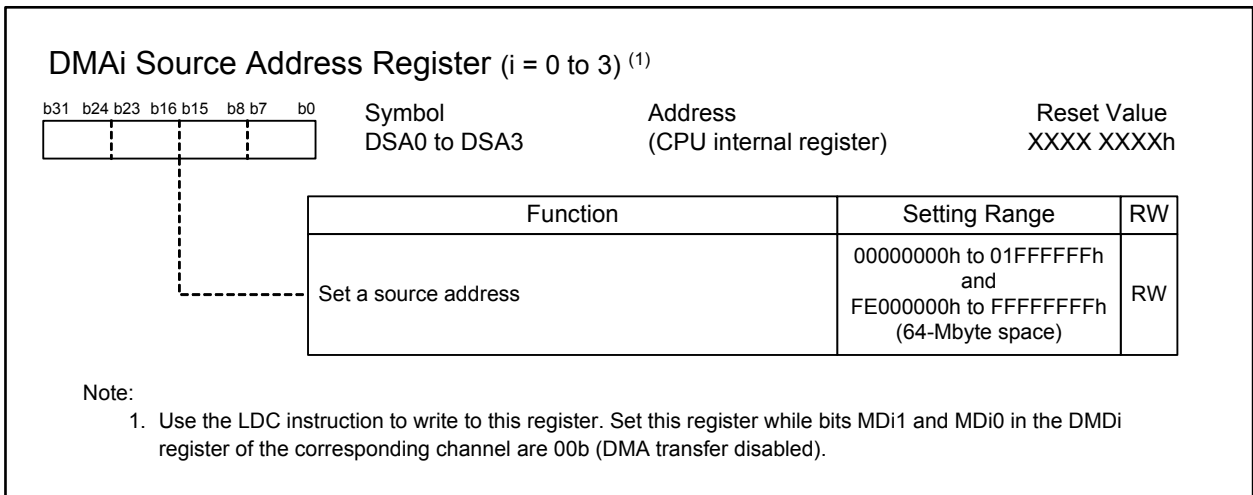
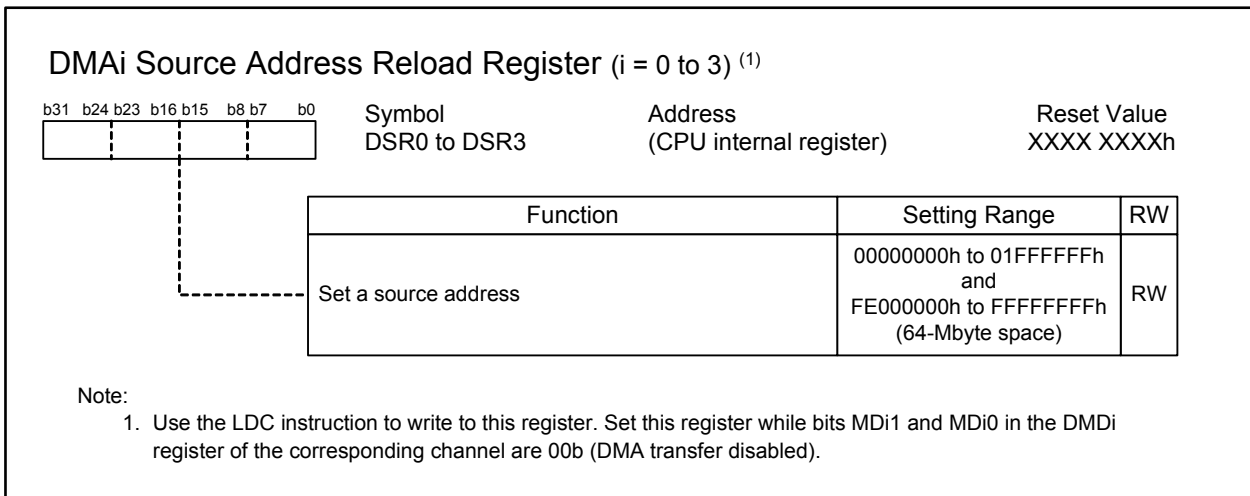
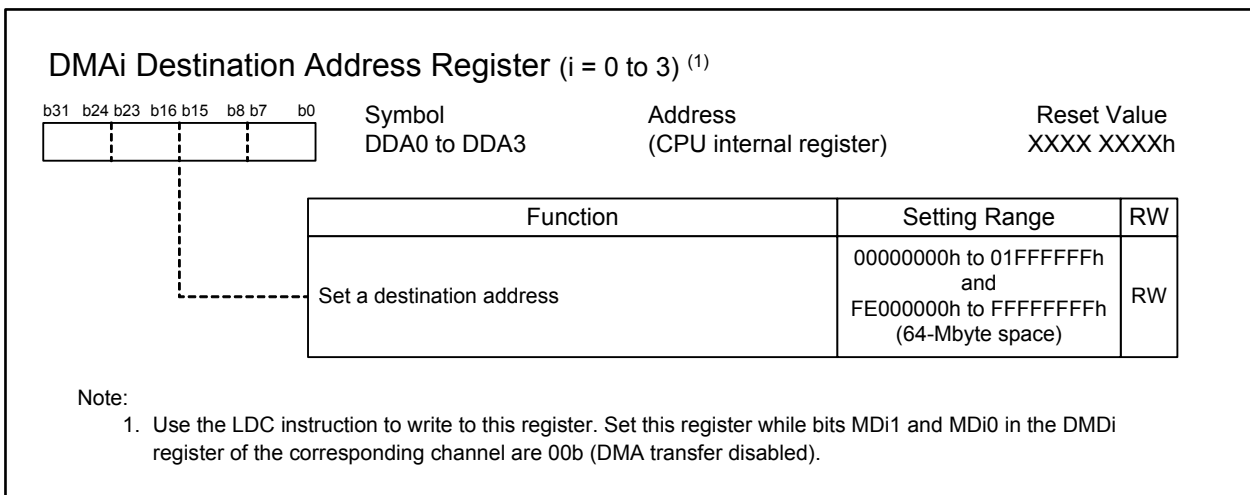
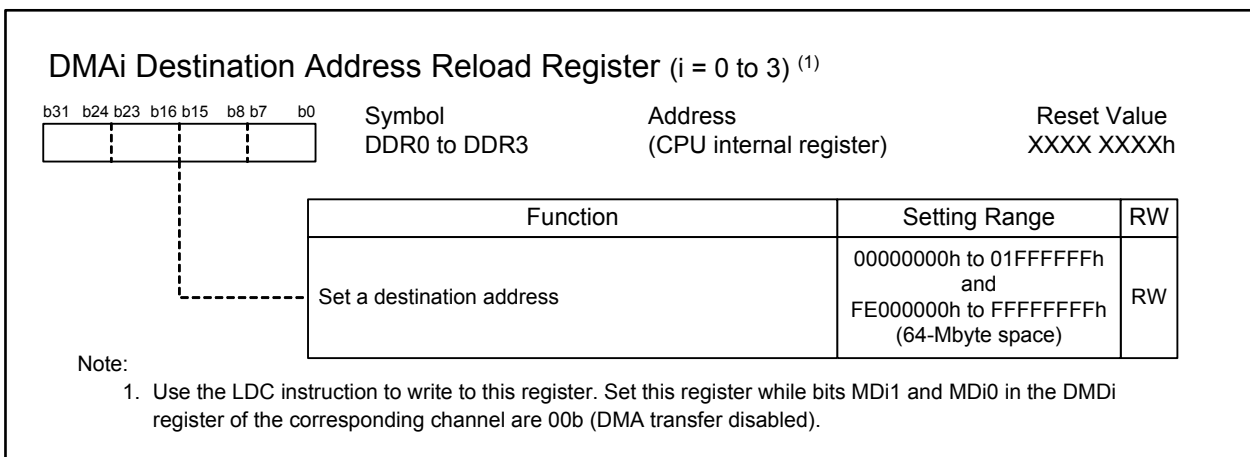


Figure 12.7 Registers DSA0 to DSA3

**Figure 12.8 Registers DSR0 to DSR3****Figure 12.9 Registers DDA0 to DDA3****Figure 12.10 Registers DDR0 to DDR3**

## 12.1 Transfer Cycle

The transfer cycle is composed of bus cycles to read data from (source read) or to write data to (destination write) memory or an SFR.

The read and write bus cycles vary with the setting of registers DSA<sub>i</sub> and DDA<sub>i</sub>, the width and timing of the data bus connected to the relevant device ( $i = 0$  to 3).

### 12.1.1 Effect of Transfer Address and Data Bus Width

Table 12.4 lists the incremental bus cycles caused by transfer address alignment or data bus width.

**Table 12.4 Incremental Bus Cycles Caused by Transfer Address and Data Bus Width**

Transfer Data Unit	Data Bus Width	Transfer Address	Bus Cycles to be Incremented	Bus Cycles Generated
8-bit transfer	8 to 64 bits	$n$	0	$[n]$
16-bit transfer	8 bits	$n$	+1	$[n] - [n + 1]$
		$2n$	0	$[2n]$
	16 bits	$2n + 1$	+1	$[2n + 1] - [2n + 2]$
		$4n$	0	$[4n]$
		$4n + 1$	0	$[4n + 1]$
		$4n + 2$	0	$[4n + 2]$
	32 bits	$4n + 3$	+1	$[4n + 3] - [4n + 4]$
		$8n$	0	$[8n]$
		$8n + 1$	0	$[8n + 1]$
	64 bits	$8n + 2$	0	$[8n + 2]$
		$8n + 3$	0	$[8n + 3]$
		$8n + 4$	0	$[8n + 4]$
		$8n + 5$	0	$[8n + 5]$
		$8n + 6$	0	$[8n + 6]$
		$8n + 7$	+1	$[8n + 7] - [8n + 8]$
32-bit transfer		8 bits	$n$	+3
	16 bits	$4n$	+1	$[4n] - [4n + 2]$
		$4n + 1$	+2	$[4n + 1] - [4n + 2] - [4n + 4]$
		$4n + 2$	+1	$[4n + 2] - [4n + 4]$
		$4n + 3$	+2	$[4n + 3] - [4n + 4] - [4n + 6]$
	32 bits	$4n$	0	$[4n]$
		$4n + 1$	+1	$[4n + 1] - [4n + 4]$
		$4n + 2$	+1	$[4n + 2] - [4n + 4]$
	64 bits	$4n + 3$	+1	$[4n + 3] - [4n + 4]$
		$8n$	0	$[8n]$
		$8n + 1$	0	$[8n + 1]$
		$8n + 2$	0	$[8n + 2]$
		$8n + 3$	0	$[8n + 3]$
		$8n + 4$	0	$[8n + 4]$
		$8n + 5$	+1	$[8n + 5] - [8n + 8]$
$8n + 6$		+1	$[8n + 6] - [8n + 8]$	
$8n + 7$	+1	$[8n + 7] - [8n + 8]$		

### 12.1.2 Effect of Bus Timing

In the R32C/100 Series, a separate bus is connected to each device. The bus width and bus timing vary with each device. Table 12.5 lists the bus width and access cycles for each device.

**Table 12.5 Bus Width and Bus Cycles**

Device	Addresses (1)	Bus Width	Access Cycles (2)	Reference Clock
Flash memory	FFE00000h to FFFFFFFFh	64-bit	2 or 3 (3)	CPU clock
Data flash	00060000h to 00061FFFh	64-bit	5	CPU clock
RAM	00000400h to 0003FFFFh	64-bit	1 or 2 (4)	CPU clock
SFR space	00000000h to 0000001Fh	16-bit	3 (5)	Peripheral bus clock
	00000020h to 000003FFh	16-bit	2 (5)	Peripheral bus clock
SFR2 space	00040000h to 00041FFFh	16-bit	2 (5)	Peripheral bus clock
	00042000h to 00043FFFh	32-bit	2 (5)	Peripheral bus clock
	00044000h to 000440DFh	16-bit	2 (5, 6)	Peripheral bus clock
	000440E0h to 000443FFh	16-bit	3 (5, 6)	Peripheral bus clock
	00044400h to 00045FFFh	16-bit	2 (5, 6)	Peripheral bus clock
	00046000h to 000467FFh	32-bit	3 (5, 6)	Peripheral bus clock
	00046800h to 00047FFFh	32-bit	2 (5, 6)	Peripheral bus clock
	00048000h to 0004FFFFh	64-bit	2	CPU clock

Notes:

1. Reserved spaces are included.
2. Access cycles are based on each bus clock.
3. An access to the same page as the previous time requires two cycles. Otherwise, three cycles are required.
4. If write cycles are generated sequentially, each write cycle except the initial one has two access cycles. A read cycle just after a write cycle has also two access cycles.
5. If SFRs are sequentially accessed, each access except the initial one has one additional base clock cycle.
6. Up to one access cycle may be added depending on the phase of peripheral bus clock.

Figure 12.11 shows an example of source-read bus cycles in a transfer cycle. In this figure, the number of source-read bus cycles is shown under different conditions, provided that the destination address is in an internal RAM with one bus cycle of destination-write. In a real operation, the transfer cycles change according to conditions for destination-write bus cycles as well as for source-read bus cycles. To calculate a transfer cycle, respective conditions should be applied to both destination-write bus cycle and source-read bus cycle. In (B) of Figure 12.11, for example, if two bus cycles are generated, bus cycles required for the destination-write is two as well as for the source-read bus cycle.

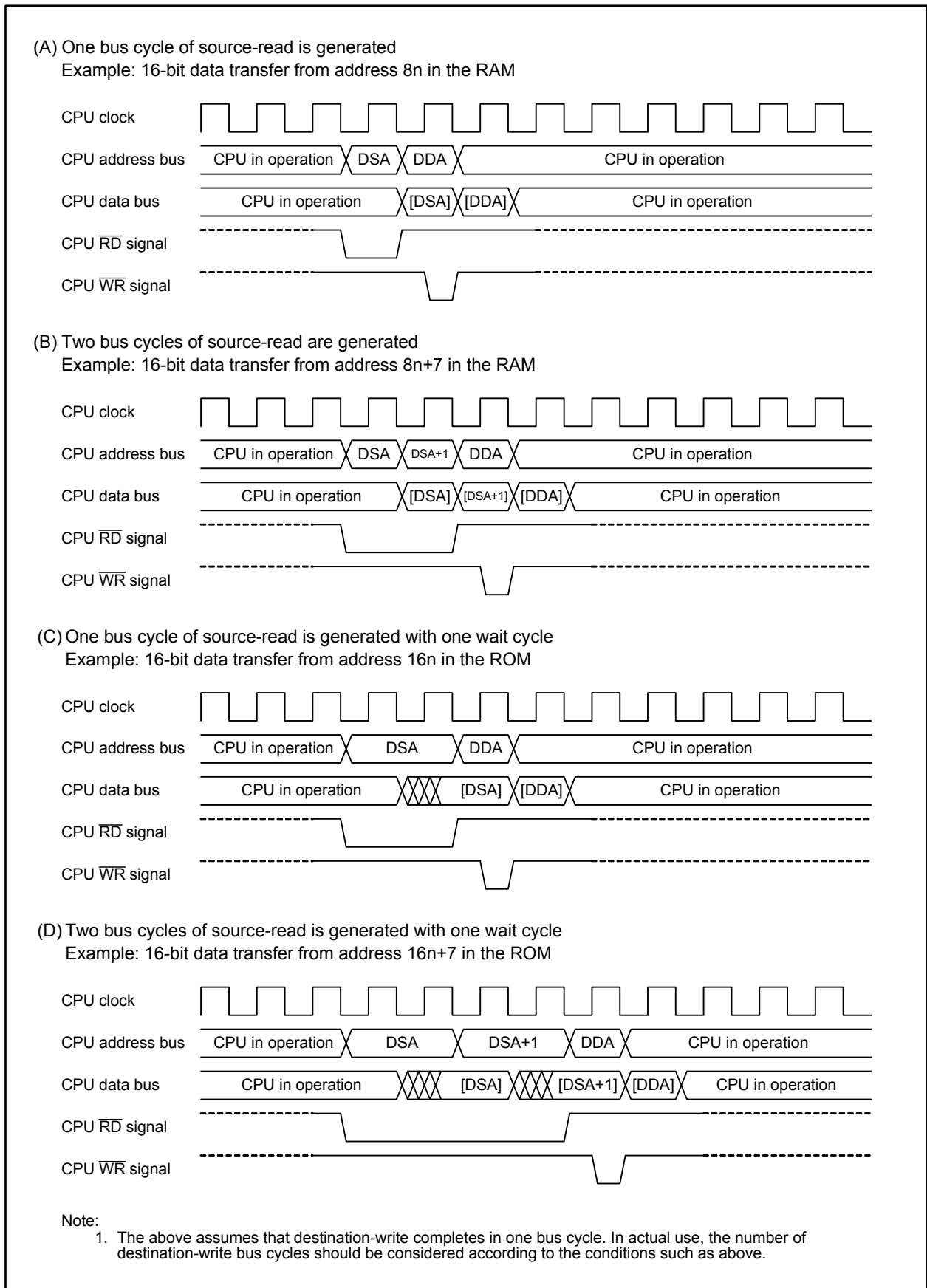


Figure 12.11 Source-read Bus Cycles in a Transfer Cycle

## 12.2 DMA Transfer Cycle

The DMA transfer cycles are calculated as follows:

$$\text{Number of transfer cycles} = \text{Source-read bus cycles} \times j + \text{Destination-write bus cycles} \times k + 1$$

where:

$j$  = access cycles for read

$k$  = access cycles for write (refer to Table 12.5)

Each bus cycle, source-read, and destination-write requires one or more cycles. In addition, more cycles may be required depending on the transfer address. Refer to Table 12.4 for details on the required bus cycles.

“+1” in the formula above means a cycle required to decrement the value of DCTi register ( $i = 0$  to 3).

The following are calculation examples:

To transfer 32-bit data from address 400h in the RAM to address 800h in the RAM,

$$\begin{aligned} \text{Number of the transfer cycles} &= 1 \times 1 + 1 \times 1 + 1 \\ &= 3 \end{aligned}$$

Thus, there are three cycles.

To transfer 16-bit data from the AD00 register at address 380h to registers P1 and P0 at addresses 3C1h and 3C0h, respectively, when the peripheral bus clock frequency is half the CPU clock,

$$\begin{aligned} \text{Number of the transfer cycles} &= 1 \times 2 \times 2 + 1 \times 2 \times 2 + 1 \\ &= 9 \end{aligned}$$

Thus, there are nine cycles.



### 12.3 Channel Priority and DMA Transfer Timing

When multiple DMA transfer requests are generated in the same sampling period, between the falling edge of the CPU clock and the next falling edge, these requests are simultaneously input into the DMAC. Channel priority in this case is: DMA0 > DMA1 > DMA2 > DMA3.

Figure 12.12 shows an example of the DMA transfer by external source, specifically when DMA0 and DMA1 requests are simultaneously generated. The DMA0, whose request priority is higher than that of DMA1, is received first to start the transfer and then hands over the bus to the CPU after completing one DMA0 transfer. Once the CPU completes one bus access, the DMA1 transfer starts. The CPU takes the bus back from the DMA1 after one DMA1 transfer is completed.

DMA transfer requests cannot be counted. Only a single transfer is performed even when an  $\overline{\text{INT}}_i$  interrupt occurs more than once before the bus is granted, as shown by DMA1 in Figure 12.12.

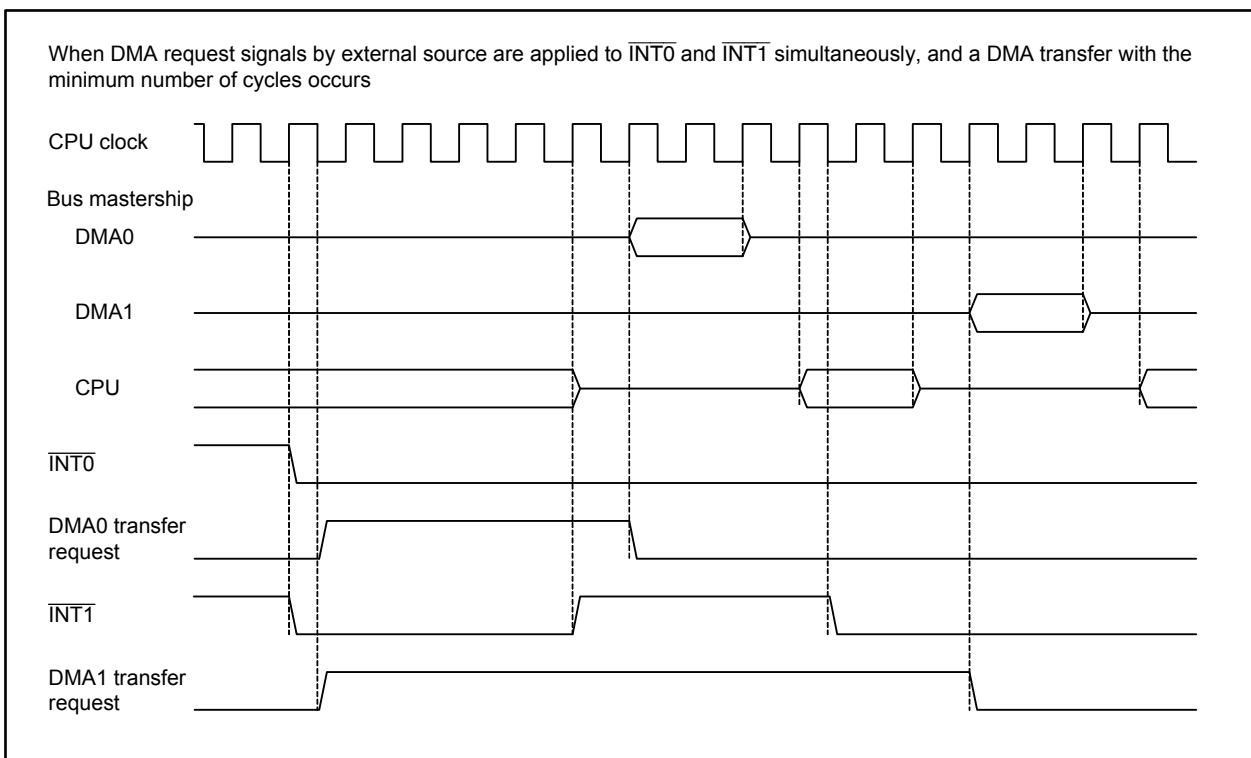


Figure 12.12 DMA Transfer by External Source

## 12.4 Notes on DMAC

### 12.4.1 DMAC-associated Register Settings

- Set DMAC-associated registers while bits MDi1 and MDi0 in the DMDi register are 00b (DMA transfer disabled) (i = 0 to 3). Then, set bits MDi1 and MDi0 to 01b (single transfer) or 11b (repeat transfer) at the end of the setup procedure. This procedure also applies when rewriting bits UDAi, USAi, and BWi1 and BWi0 in the DMDi register.
- When rewriting the DMAC-associated registers while DMA transfer is enabled, stop the peripherals that can be DMA triggers so that no DMA transfer request is generated, then set bits MDi1 and MDi0 in the DMDi register of the corresponding channel to 00b (DMA transfer disabled).
- Once a DMA transfer request is accepted, DMA transfer cannot be disabled even if setting bits MDi1 and MDi0 in the DMDi register to 00b (DMA transfer disabled). Do not change the settings of any DMAC-associated registers other than bits MDi1 and MDi0 until the DMA transfer is completed.
- After setting registers DMiSL and DMiSL2, wait at least six peripheral bus clocks to set bits MDi1 and MDi0 in the DMDi register to 01b (single transfer) or 11b (repeat transfer).

### 12.4.2 Reading DMAC-associated Registers

- Use the following read order to sequentially read registers DMiSL and DMiSL2:  
DM0SL, DM1SL, DM2SL, and DM3SL  
DM0SL2, DM1SL2, DM2SL2, and DM3SL2

## 13. DMAC II

DMAC II starts by an interrupt request from any peripheral and performs data transfer without a CPU instruction. Transfer sources are selectable from memory, immediate data, memory + memory, and immediate data + memory.

Table 13.1 lists specifications of DMAC II.

**Table 13.1 DMAC II Specifications**

Item	Specification
DMAC II request sources	Interrupt requests from the peripherals of which bits ILVL2 to ILVL0 in the corresponding interrupt control register are set to 111b (level 7)
Transfer types	<ul style="list-style-type: none"> <li>• Data in memory is transferred to memory (memory-to-memory transfer)</li> <li>• Immediate data is transferred to memory (immediate data transfer)</li> <li>• Data in memory + data in memory are transferred to memory (calculation result transfer)</li> <li>• Immediate data + data in memory are transferred to memory (calculation result transfer)</li> </ul>
Transfer sizes	8 bits or 16 bits
Transfer memory spaces	From a given address in a 64-Mbyte space (00000000h to 01FFFFFFh and FE000000h to FFFFFFFFh) to another given address in the same space <sup>(1)</sup>
Addressing modes	<p>Individually selectable for each source address and destination address from the following two modes:</p> <ul style="list-style-type: none"> <li>• Non-incrementing addressing: Address is held constant throughout a data transfer/DMAC II transaction</li> <li>• Incrementing addressing: Address increments by 1 (when 8-bit data is transferred) or 2 (when 16-bit data is transferred) after each data transfer</li> </ul>
Transfer modes	<ul style="list-style-type: none"> <li>• Single transfer: Only one data transfer is performed by one transfer request</li> <li>• Burst transfer: Data transfers are continuously performed for the number of times set in the transfer counter by one transfer request</li> <li>• Multiple transfer: Multiple memory-to-memory transfers are performed from different source addresses to different destination addresses by one transfer request</li> </ul>
Chain transfer	Data transfer is sequentially performed by switching among multiple DMAC II indexes (transfer information)
DMA II transfer complete interrupt request	An interrupt request is generated when the transfer counter reaches 0000h

Note:

1. When the transfer size is 16 bits and the destination address is FFFFFFFFh, data is transferred to FFFFFFFFh and 00000000h. This also applies when the source address is FFFFFFFFh.

### 13.1 DMAC II Settings

To use DMAC II, set the following:

- Registers RIPL1 and RIPL2
- DMAC II index
- Interrupt control registers of the peripherals that trigger DMAC II
- Relocatable vectors of the peripherals that trigger DMAC II
- The IRLT bit in the IIOiE register when using the intelligent I/O interrupt (i = 0 to 11). Refer to 10. "Interrupts" for details on the IIOiE register.

### 13.1.1 Registers RIPL1 and RIPL2

When the DMAII bit in registers RIPL1 and RIPL2 is set to 1 (DMA II transfer selected) and the FSIT bit is set to 0 (normal interrupt selected), DMAC II starts by an interrupt request from any peripheral whose bits ILVL2 to ILVL0 in the corresponding interrupt control register are set to 111b (level 7).

Figure 13.1 shows registers RIPL1 and RIPL2.

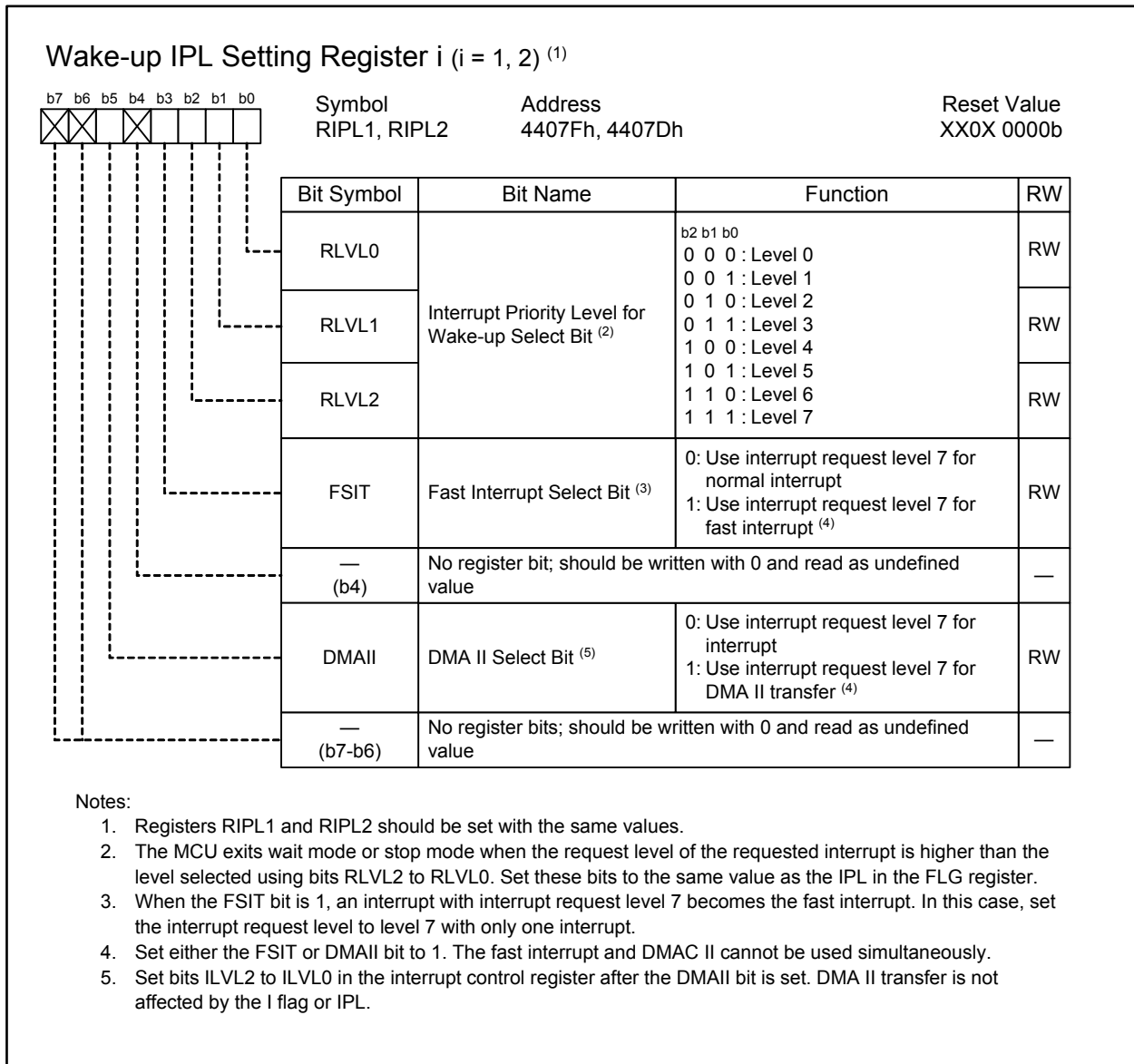


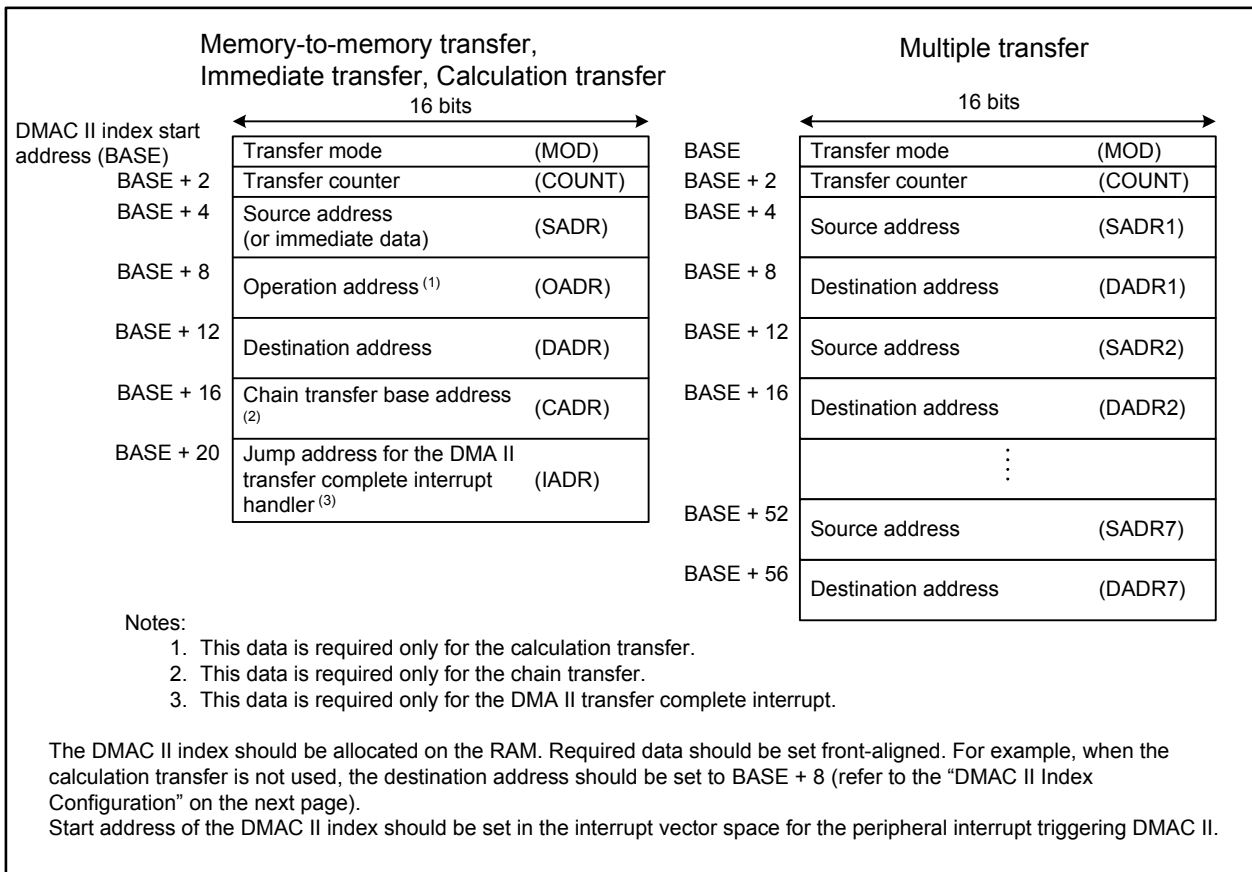
Figure 13.1 Registers RIPL1 and RIPL2

### 13.1.2 DMAC II Index

The DMAC II index is a data table of 12 to 60 bytes. It stores parameters for transfer mode, transfer counter, source address (or immediate data), operation address as an address to be calculated, destination address, chain transfer base address, and jump address for the DMA II transfer complete interrupt handler.

This DMAC II index should be allocated on the RAM.

Figure 13.2 shows a configuration of the DMAC II index and Table 13.2 lists a configuration example of the DMAC II index.



**Figure 13.2 DMAC II Index**

The following are the details on the DMAC II index. These parameters should be aligned in the order listed in Table 13.2 according to the transfer mode to be used.

- Transfer mode (MOD)  
Set a transfer mode in 2 bytes. Refer to Figure 13.3 for details on the setting of MOD.
- Transfer counter (COUNT)  
Set a number of transfers in 2 bytes.
- Source address (SADR)  
Set a source address or immediate data in 4 bytes. Note that the two upper bytes of immediate data are ignored.
- Operation address (OADR)  
Set an address in a to-be calculated memory in 4 bytes. This data setting is required only for the calculation transfer.
- Destination address (DADR)  
Set a destination address in 4 bytes.
- Chain transfer base address (CADR)  
Set the start address of the DMAC II index for the next transfer (BASE) in 4 bytes. This data setting is required only for the chain transfer.
- Jump address for the DMA II transfer complete interrupt handler (IADR)  
Set the start address for the DMA II transfer complete interrupt handler in 4 bytes. This data setting is required only for the DMA II transfer complete interrupt.

The symbols above are hereinafter used in place of their respective parameters.

**Table 13.2 DMAC II Index Configuration**

Transfer Data	Memory-to-memory Transfer/ Immediate Data Transfer				Calculation Transfer				Multiple Transfer	
	Not used	Used	Not used	Used	Not used	Used	Not used	Used		
Chain transfer	Not used	Used	Not used	Used	Not used	Used	Not used	Used	Not available	
DMA II transfer complete interrupt	Not used	Not used	Used	Used	Not used	Not used	Used	Used	Not available	
DMAC II index	MOD	MOD	MOD	MOD	MOD	MOD	MOD	MOD	MOD	
	COUNT	COUNT	COUNT	COUNT	COUNT	COUNT	COUNT	COUNT	COUNT	
	SADR	SADR	SADR	SADR	SADR	SADR	SADR	SADR	SADR1	
	DADR	DADR	DADR	DADR	OADR	OADR	OADR	OADR	DADR1	
	12 bytes	CADR	IADR	CADR	IADR	DADR	DADR	DADR	DADR	SADRi
	24 bytes	IADR	IADR	IADR	DADRi					
						i = 1 to 7 max. 60 bytes (when i = 7)				

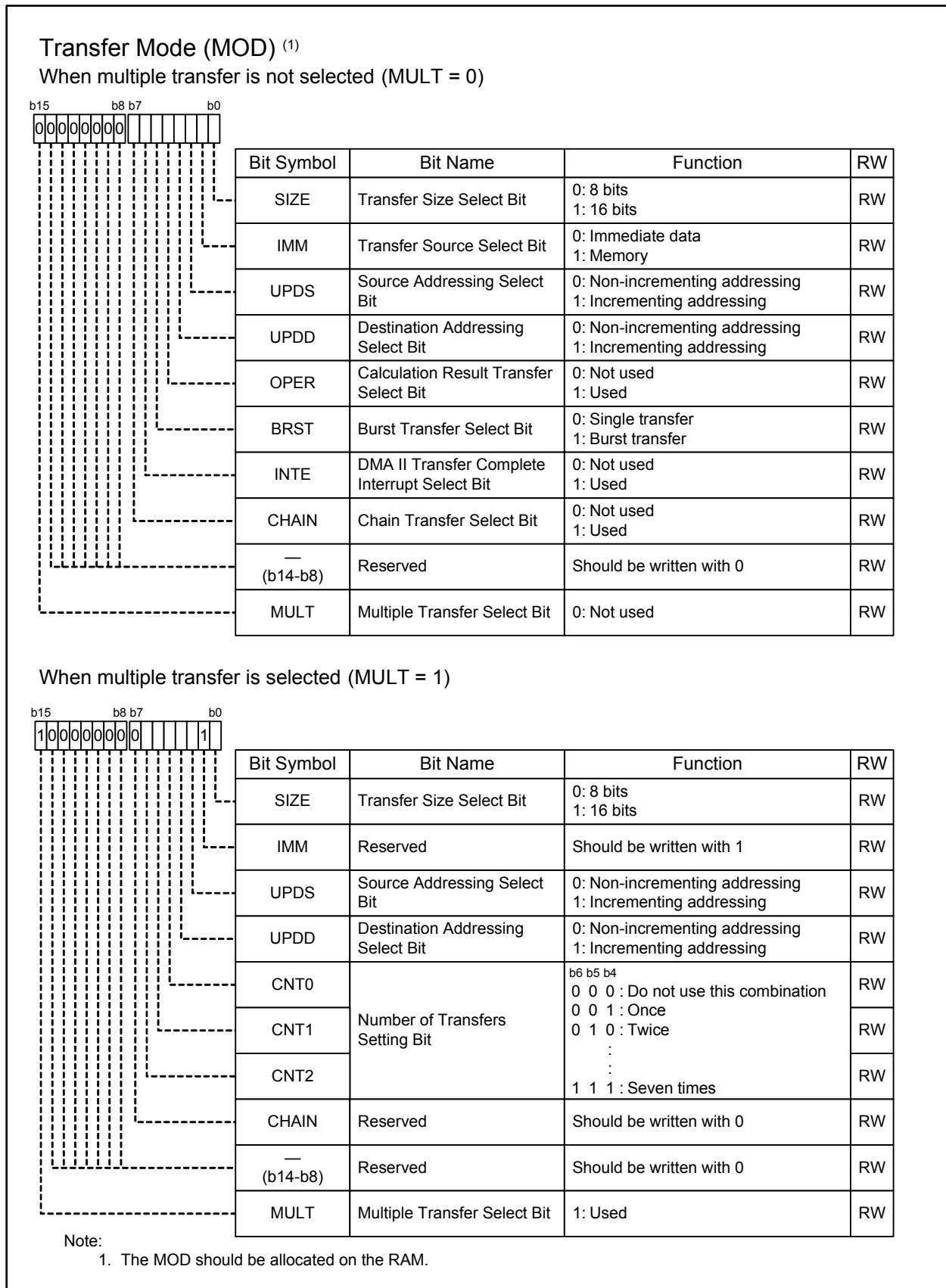


Figure 13.3 MOD

### 13.1.3 Interrupt Control Register of the Peripherals

Set bits ILVL2 to ILVL0 in the interrupt control register for the peripheral interrupt triggering DMAC II to 111b (level 7).

### 13.1.4 Relocatable Vector Table of the Peripherals

Set the start address of the DMAC II index in the interrupt vector space for the peripheral interrupt triggering DMAC II.

To use the chain transfer, allocate the relocatable vector table on the RAM.

### 13.1.5 IRLT Bit in the IIOiE Register (i = 0 to 11)

To use the intelligent I/O interrupt as a trigger for DMAC II, set the IRLT bit in the corresponding IIOiE register to 0 (interrupt request for DMA or DMA II used).

## 13.2 DMAC II Operation

Set the DMAII bit in registers RIPL1 and RIPL2 to 1 (interrupt request level 7 used for DMA II transfer) to perform a DMA II transfer. DMAC II starts by an interrupt request from any peripheral whose bits ILVL2 to ILVL0 in the corresponding interrupt control register are set to 111b (level 7). These peripheral interrupt requests are available only for DMA II transfer and cannot be used for the CPU.

When an interrupt request is generated with interrupt request level 7, DMAC II starts irrespective of the state of the I flag or IPL.

When a peripheral interrupt request triggering DMAC II and a higher-priority request such as the watchdog timer interrupt, low voltage detection interrupt, oscillator stop detection interrupt, or NMI are simultaneously generated, the higher-priority interrupt is accepted prior to the DMA II transfer, and the DMA II transfer starts after the higher-priority interrupt sequence.

## 13.3 Transfer Types

DMAC II transfers three types of 8-bit or 16-bit data as follows:

- Memory-to-memory transfer: Data is transferred from a given memory location in a 64-Mbyte space (addresses 00000000h to 01FFFFFFh and FE000000h to FFFFFFFFh) to another given memory location in the same space.
- Immediate data transfer: Immediate data is transferred to a given memory location in a 64-Mbyte space.
- Calculation transfer: Two data are added together and the result is transferred to a given memory location in a 64-Kbyte space.

When 16-bit data is transferred to DADR at FFFFFFFFh, it is transferred to 00000000h as well as FFFFFFFFh. The same transfer is performed when SADR is FFFFFFFFh.

### 13.3.1 Memory-to-memory Transfer

Data transfer between any two memory locations can be:

- A transfer from a fixed address to another fixed address
- A transfer from a fixed address to an address range in memory
- A transfer from an address range in memory to a fixed address
- A transfer from an address range in memory to another address range in memory

When increment addressing mode is selected, SADR and DADR increment by 1 in an 8-bit transfer and by 2 in a 16-bit transfer after a data transfer for the next transfer. When SADR or DADR exceeds FFFFFFFFh by the incrementation, it returns to 00000000h. Likewise, when SADR or DADR exceeds 01FFFFFFh, it becomes 02000000h, but an actual transfer is performed for FE000000h.



### 13.3.2 Immediate Data Transfer

DMAC II transfers immediate data to a given memory location. Either incrementing or non-incrementing addressing mode can be selected for the destination address. Store the immediate data to be transferred into SADR. To transfer 8-bit immediate data, set the data to the lower 1 byte of SADR. The upper 3 bytes are ignored. To transfer 16-bit immediate data, set the data to the lower 2 bytes. The upper 2 bytes are ignored.

### 13.3.3 Calculation Result Transfer

After two memory data or immediate data and memory data are added together, DMAC II transfers the calculated result to a given memory location. Set an address to be calculated or immediate data to SADR and set the other address to be calculated to OADR. Either incrementing or non-incrementing addressing mode can be selected for source and destination addresses when performing data in memory + data in memory calculation transfer. If the source addressing is in incrementing mode, the operation addressing is also in incrementing mode. When performing immediate data + data in memory calculation transfer, the addressing mode is selectable only for the destination address.

## 13.4 Transfer Modes

DMAC II provides three types of basic transfer mode: single transfer, burst transfer, and multiple transfer. COUNT determines the number of transfers to be performed. Transfers are not performed when COUNT is 0000h.

### 13.4.1 Single Transfer

Set the BRST bit in the MOD to 0.

A single data transfer is performed by one transfer request.

When incrementing addressing mode is selected for the source and/or destination address, the address or addresses increment after a data transfer for the next transfer.

COUNT is decremented each time a data transfer is performed. When COUNT reaches 0000h, the DMA II transfer complete interrupt request is generated if the INTE bit in the MOD is 1 (DMA II transfer complete interrupt used).

### 13.4.2 Burst Transfer

Set the BRST bit in the MOD to 1.

DMAC II continuously transfers data for the number of times determined by COUNT with one transfer request. COUNT decrements each time a data transfer is performed. When COUNT reaches 0000h, the burst transfer is completed. The DMA II transfer complete interrupt request is generated if the INTE bit is 1 (DMA II transfer complete interrupt used).

No interrupts are accepted during a burst transfer.

### 13.4.3 Multiple Transfer

Set the MULT bit in the MOD to 1.

Multiple memory-to-memory transfers are performed from different source addresses to different destination addresses using one transfer request.

Set bits CNT2 to CNT0 in the MOD to select the number of transfers to be performed from 001b (once) to 111b (seven times). Do not set these bits to 000b.

Allocate the required number of SDARs and DADRs alternately following MOD and COUNT.

When the multiple transfer is selected, the following transfer functions are not available: calculation result transfer, burst transfer, chain transfer, and DMA II transfer complete interrupt.

### 13.5 Chain Transfer

The chain transfer is available when the CHAIN bit in the MOD is 1.

The chain transfer is performed as follows:

- (1) When a transfer request is generated, a data transfer is performed according to the DMAC II index specified by the corresponding interrupt vector. Either a single transfer (the BRST bit in the MOD is 0) or burst transfer (the BRST bit is 1) is performed according to the BRST bit setting.
- (2) When COUNT reaches 0000h, the value in the interrupt vector in (1) above is overwritten with the value in CADR. Simultaneously, the DMA II transfer complete interrupt request is generated when the INTE bit in the MOD is 1.
- (3) When the next DMA II transfer request is generated, the data transfer is performed according to the DMAC II index specified by the peripheral interrupt vector in (2) above.

Figure 13.4 shows the relocatable vector and DMAC II index in a chain transfer.

To use the chain transfer, the relocatable vector table should be allocated on the RAM.

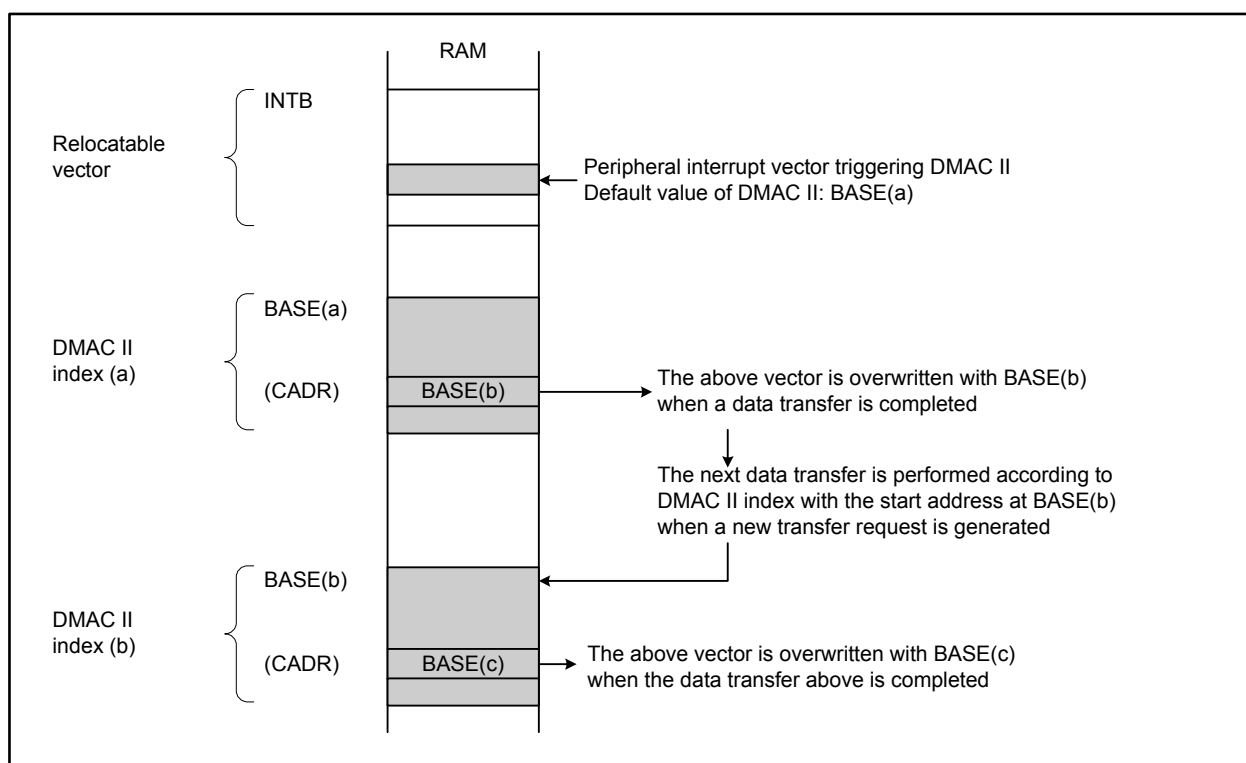


Figure 13.4 Relocatable Vector and DMAC II Index in a Chain Transfer

### 13.6 DMA II Transfer Complete Interrupt

The DMA II transfer complete interrupt is available when the INTE bit in the MOD is 1.

Set the start address of the DMA II transfer complete interrupt handler to IADR. The interrupt request is generated when COUNT reaches 0000h.

The initial instruction of the interrupt handler is executed in the eighth cycle after a DMA II transfer is completed.

### 13.7 Execution Time

The DMAC II execution cycle is calculated by the following equations:

Mode other than multiple transfer:  $t = 6 + (26 + a + b + c + d) \times m + (4 + e) \times n$  cycles

When using multiple transfer:  $t = 21 + (11 + b + c) \times k$  cycles

- a: When IMM is 0 (transfer source is immediate data), a is 0;  
When IMM is 1 (transfer source is memory), a is -1
- b: When UPDS is 1 (source addressing is incrementing), b is 0;  
When UPDS is 0 (source addressing is non-incrementing), b is 1
- c: When UPDD is 1 (destination addressing is incrementing), c is 0;  
When UPDD is 0 (destination addressing is non-incrementing), c is 1
- d: When OPER is 0 (calculation transfer is not selected), d is 0;  
When OPER is 1 (calculation transfer is selected) and UPDS is 0 (source addressing is immediate data or non-incrementing), d is 7;  
When OPER is 1 (calculation transfer is selected) and UPDS is 1 (source addressing is incrementing), d is 8
- e: When CHAIN is 0 (chain transfer is not selected), e is 0;  
When CHAIN is 1 (chain transfer is selected), e is 4
- m: When BRST is 0 (single transfer), m is 1;  
When BRST is 1 (burst transfer), m is COUNT
- n: When COUNT is 0001h, n is 0; if COUNT is 0002h or more, n is 1
- k: The number of transfers set using bits CNT2 to CNT0

The equations above are estimations. The number of cycles may vary depending on CPU state, bus wait state, and DMAC II index allocation.

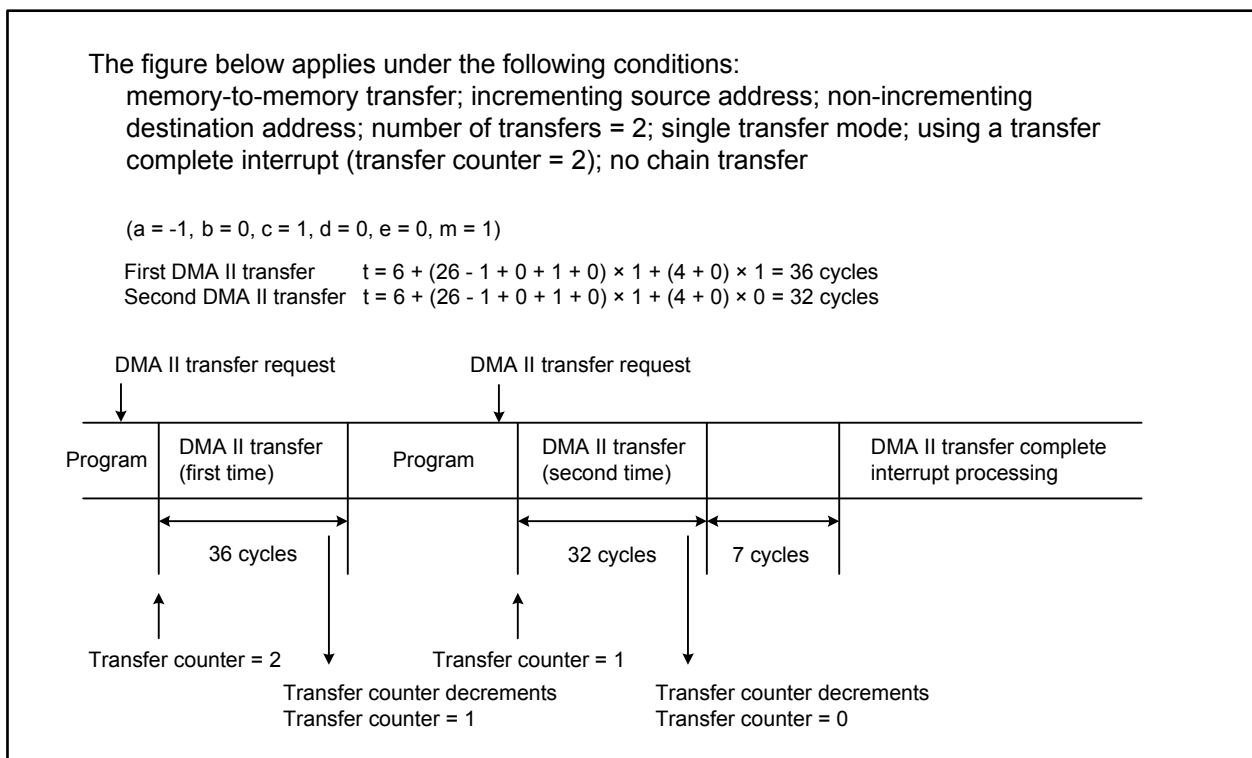


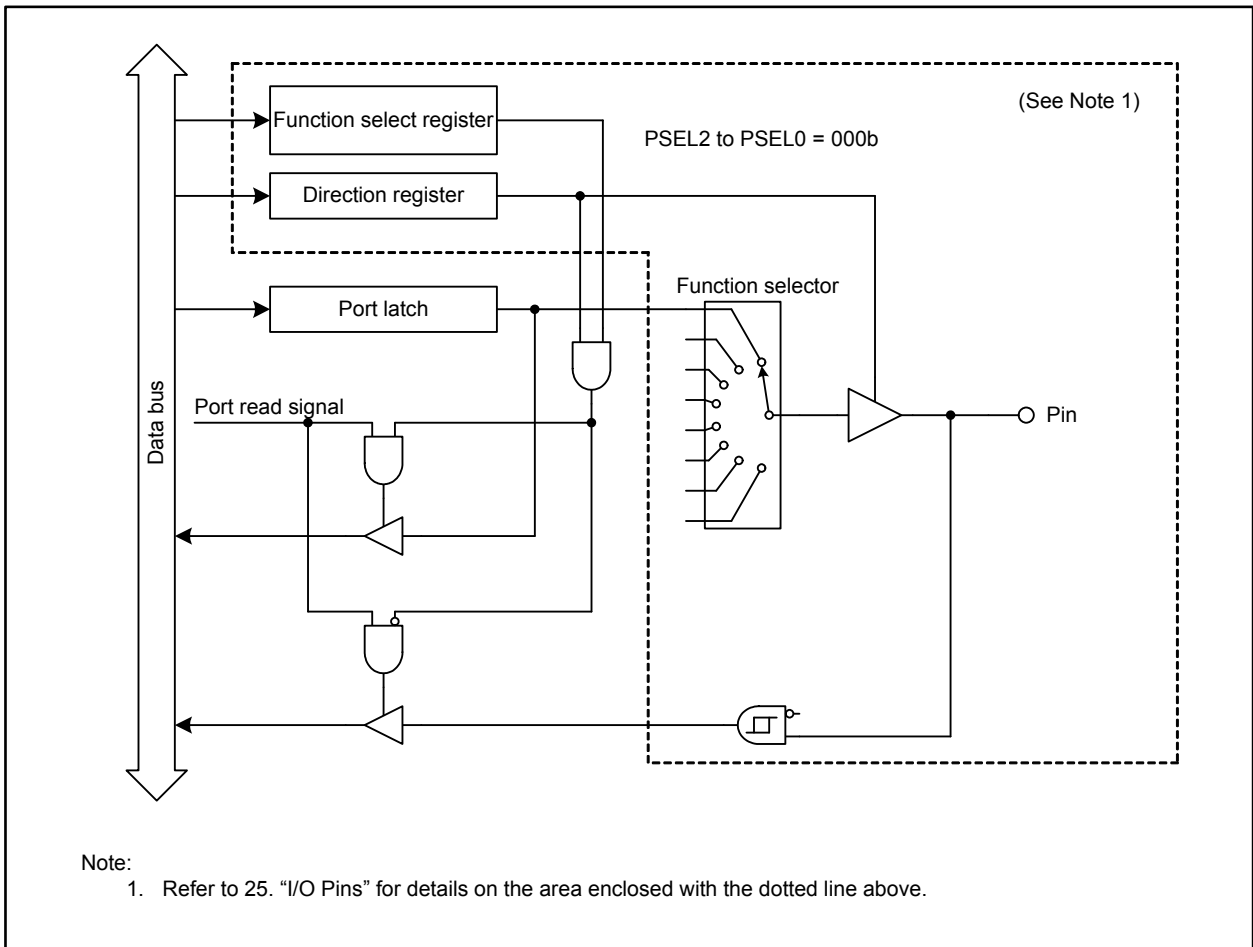
Figure 13.5 Transfer Cycles

## 14. Programmable I/O Ports

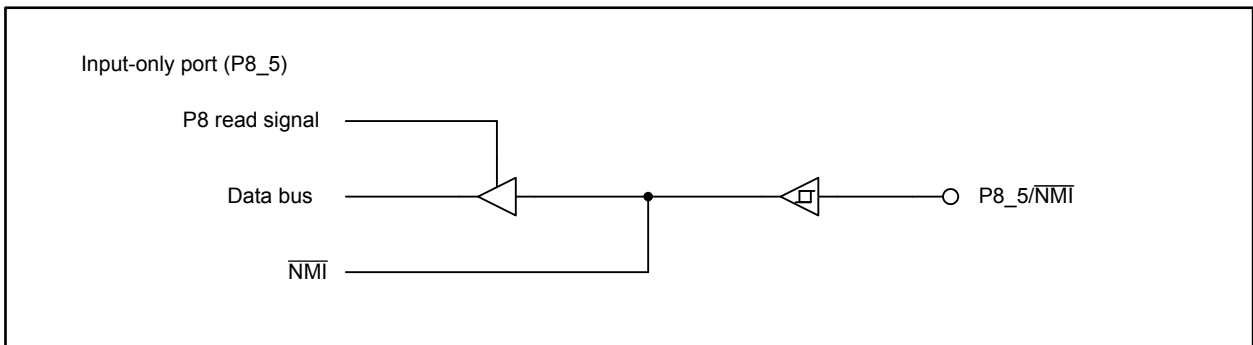
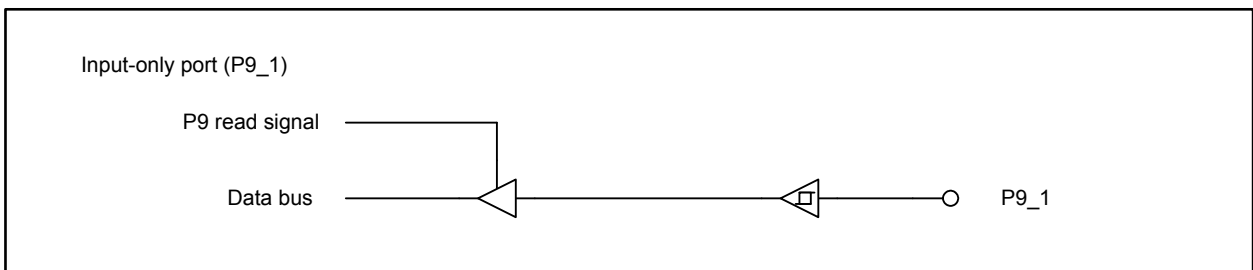
There are 64 programmable I/O ports designated from P0 to P9 (excluding P6\_0 to P6\_4, P6\_6, P7\_1 to P7\_3, P7\_7, P8\_0, P8\_1, and P9\_0 to P9\_2).

Each port status, input or output, can be selected using the direction register except P8\_5 and P9\_1 which are input only. The P8\_5 bit in the P8 register indicates an  $\overline{\text{NMI}}$  input level since the P8\_5 shares a pin with the  $\overline{\text{NMI}}$ .

Figure 14.1 shows a configuration of programmable I/O ports, and Figures 14.2 and Figure 14.3 show a configuration of each input-only port.



**Figure 14.1 Programmable I/O Port Configuration**

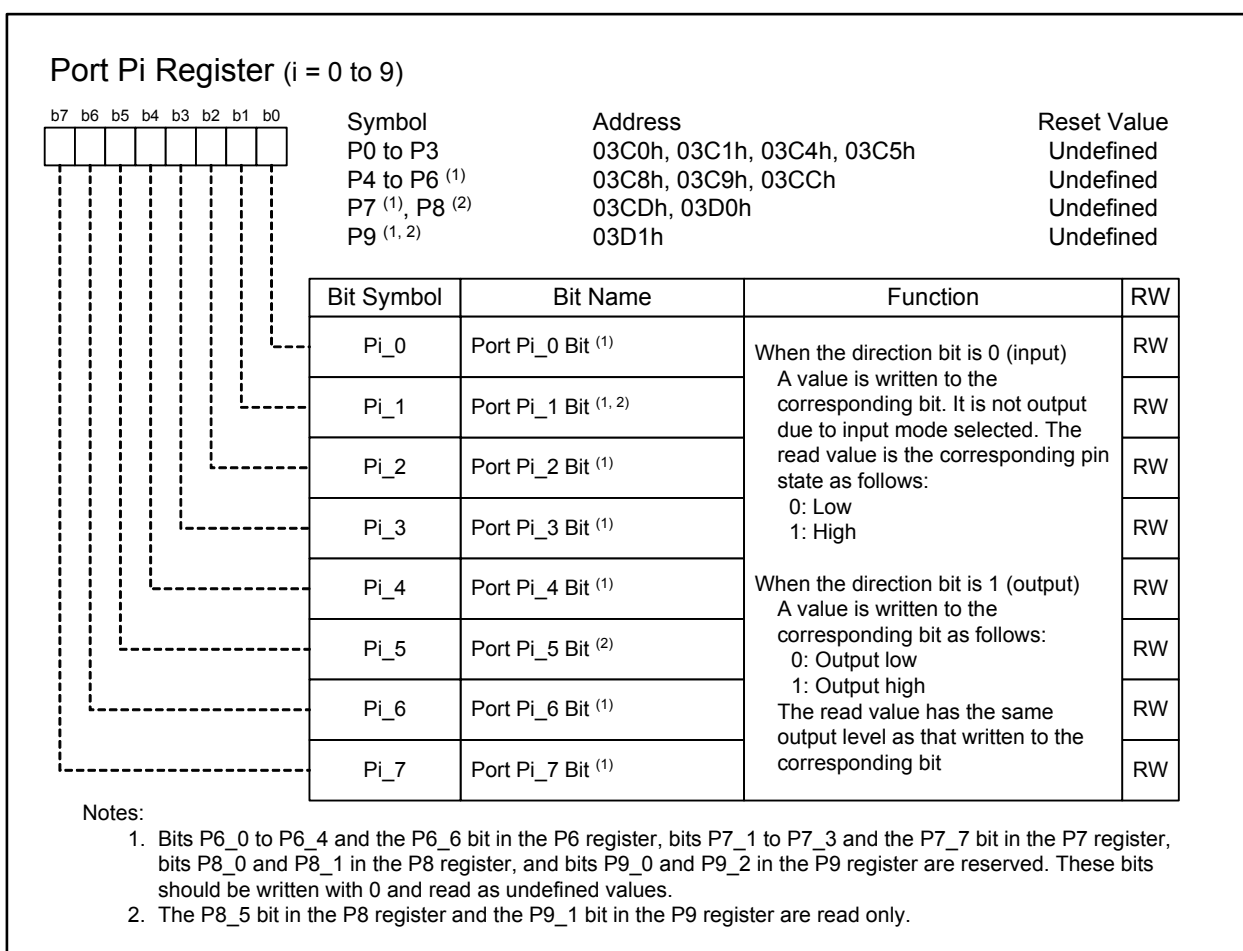
**Figure 14.2 Input-only Port Configuration (1/2)****Figure 14.3 Input-only Port Configuration (2/2)**

### 14.1 Port Pi Register (Pi register, i = 0 to 9)

A write/read operation to the Pi register is required to communicate with external devices. This register consists of a port latch to hold output data and a circuit to read pin states. Bits in the Pi register correspond to respective ports.

When a programmable I/O port is selected in the output function select register, the value in the port latch is read for output and the pin state is read for input.

Figure 14.4 shows the Pi register.

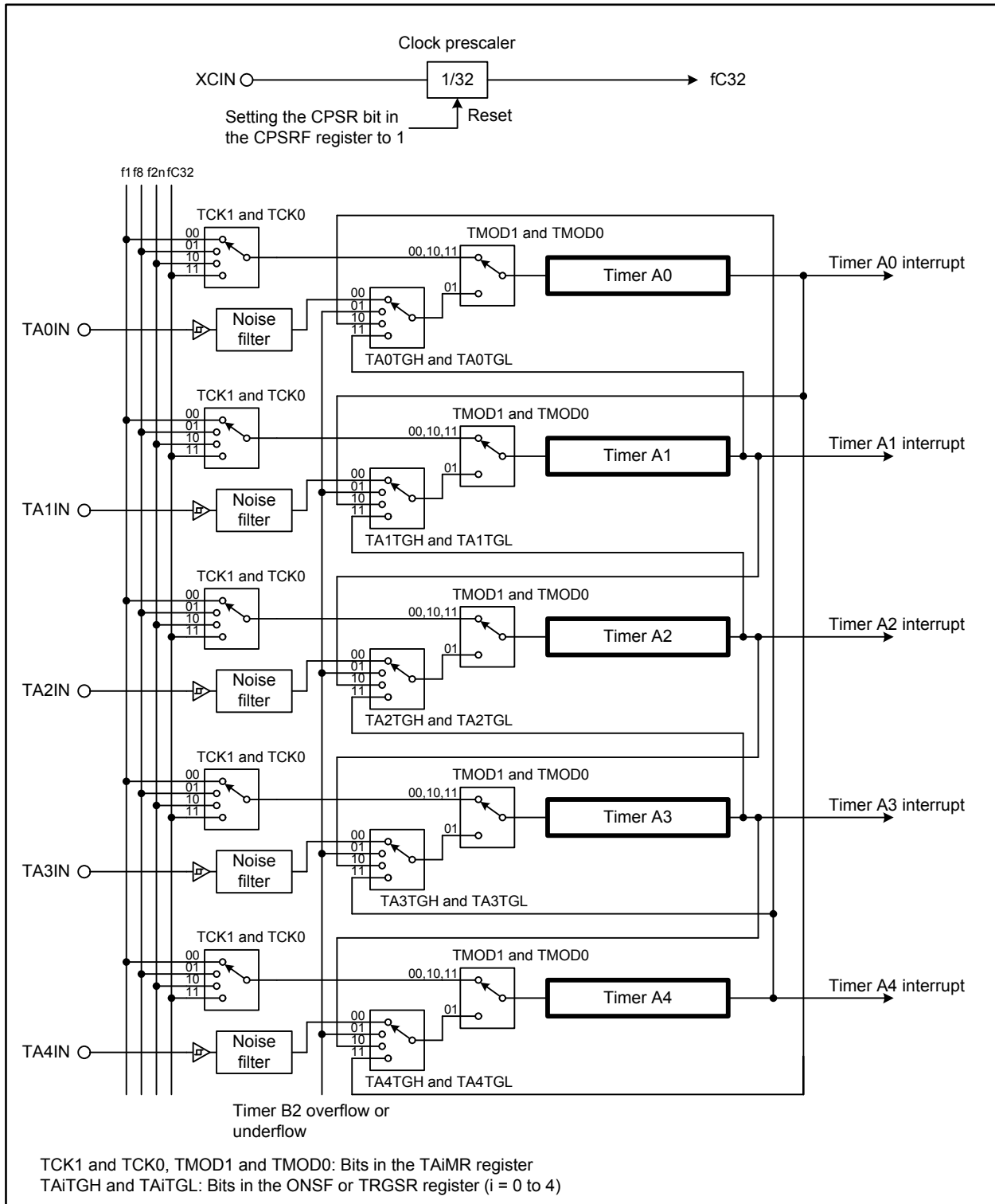


**Figure 14.4 Registers P0 to P9**

## 15. Timers

This MCU has eleven 16-bit timers which are divided into two groups according to their functions: five timer As and six timer Bs. Each timer functions individually. The count source of each timer provides the clock for timer operations such as counting and reloading.

Figure 15.1 and Figure 15.2 show the configuration of timers A and B, respectively.



**Figure 15.1 Timer A Configuration**

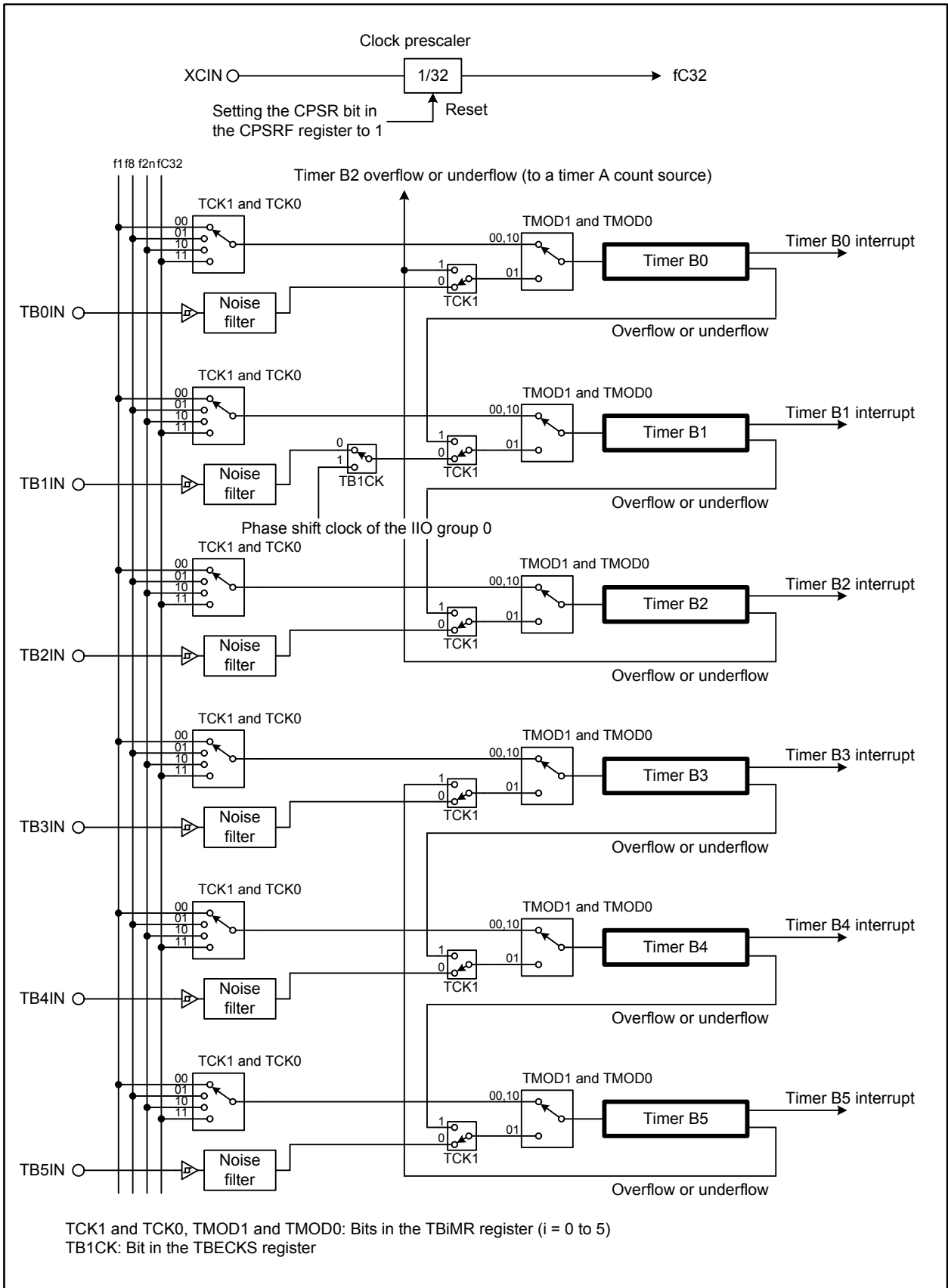


Figure 15.2 Timer B Configuration

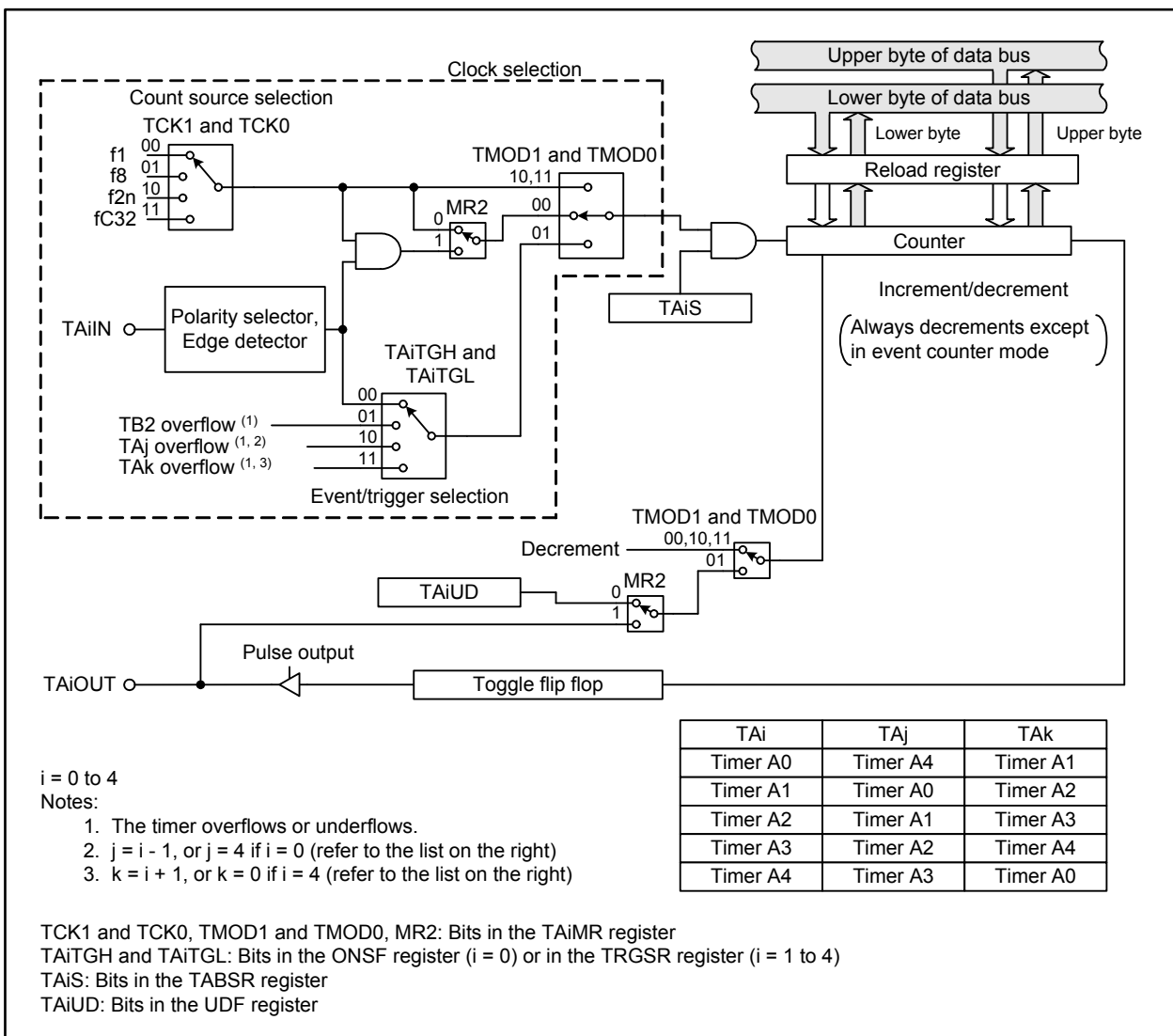


### 15.1 Timer A

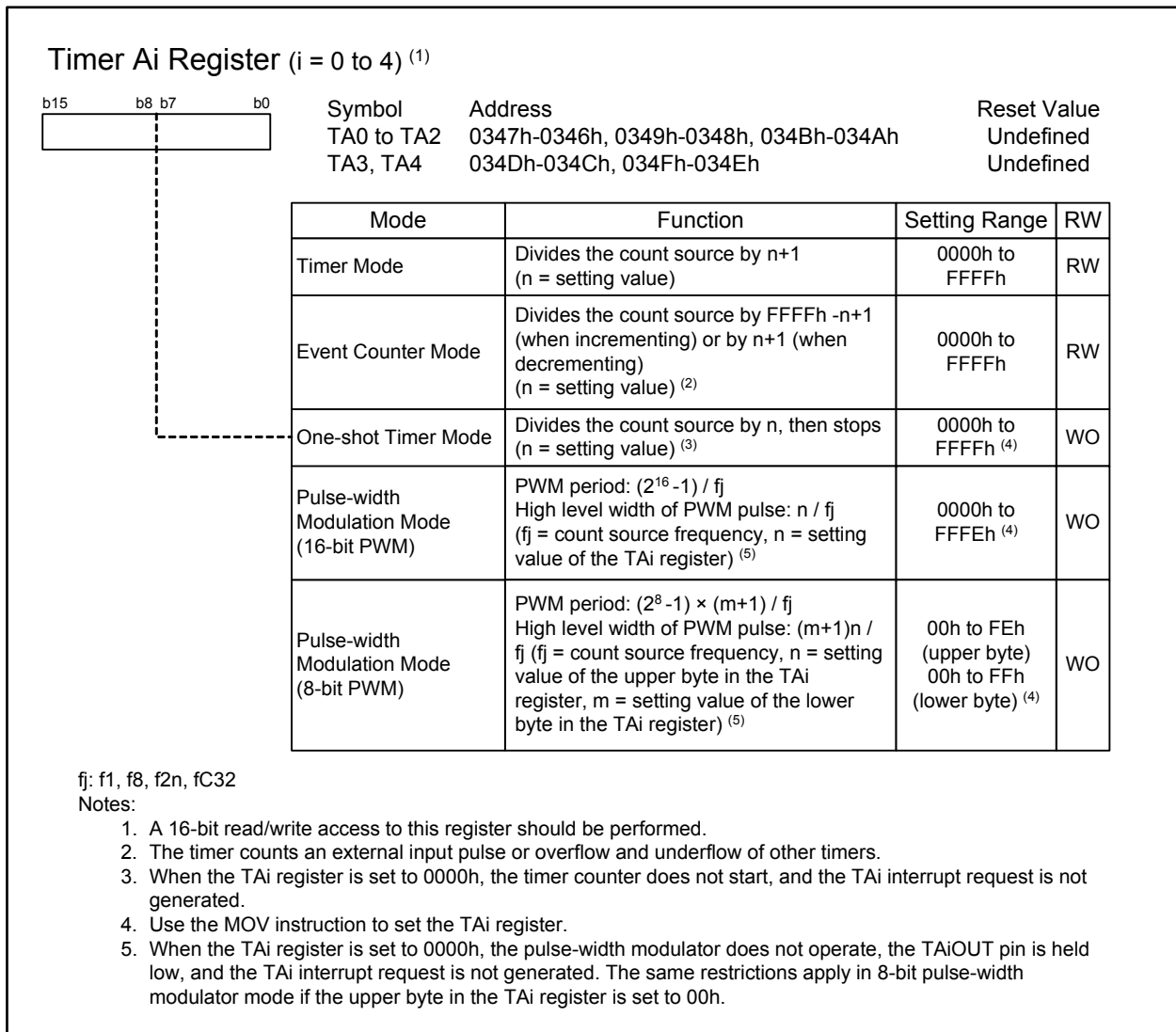
Figure 15.3 shows a block diagram of timer A and Figure 15.4 to Figure 15.10 show registers associated with timer A.

Timer A supports the four modes shown below. Timers A0 to A4 in any mode other than the event counter mode have the same function. Select a mode by setting bits TMOD1 and TMOD0 in the TAI<sub>i</sub>MR register (i = 0 to 4).

- Timer mode: The timer counts an internal count source
- Event counter mode: The timer counts an external pulse or overflow and underflow of other timers
- One-shot timer mode: The timer outputs one valid pulse before the counter reaches 0000h
- Pulse-width modulation mode: The timer successively outputs pulses of a given width



**Figure 15.3 Timer A Block Diagram**



**Figure 15.4 Registers TA0 to TA4**

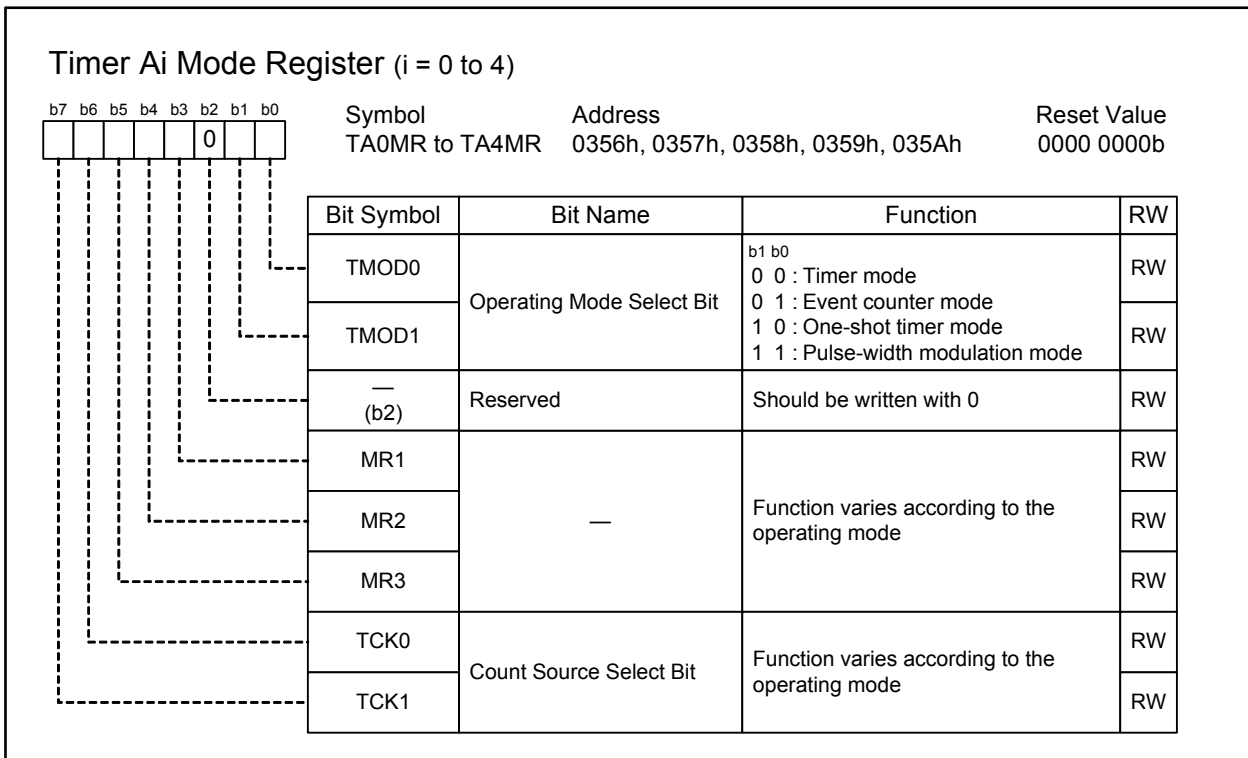


Figure 15.5 Registers TA0MR to TA4MR

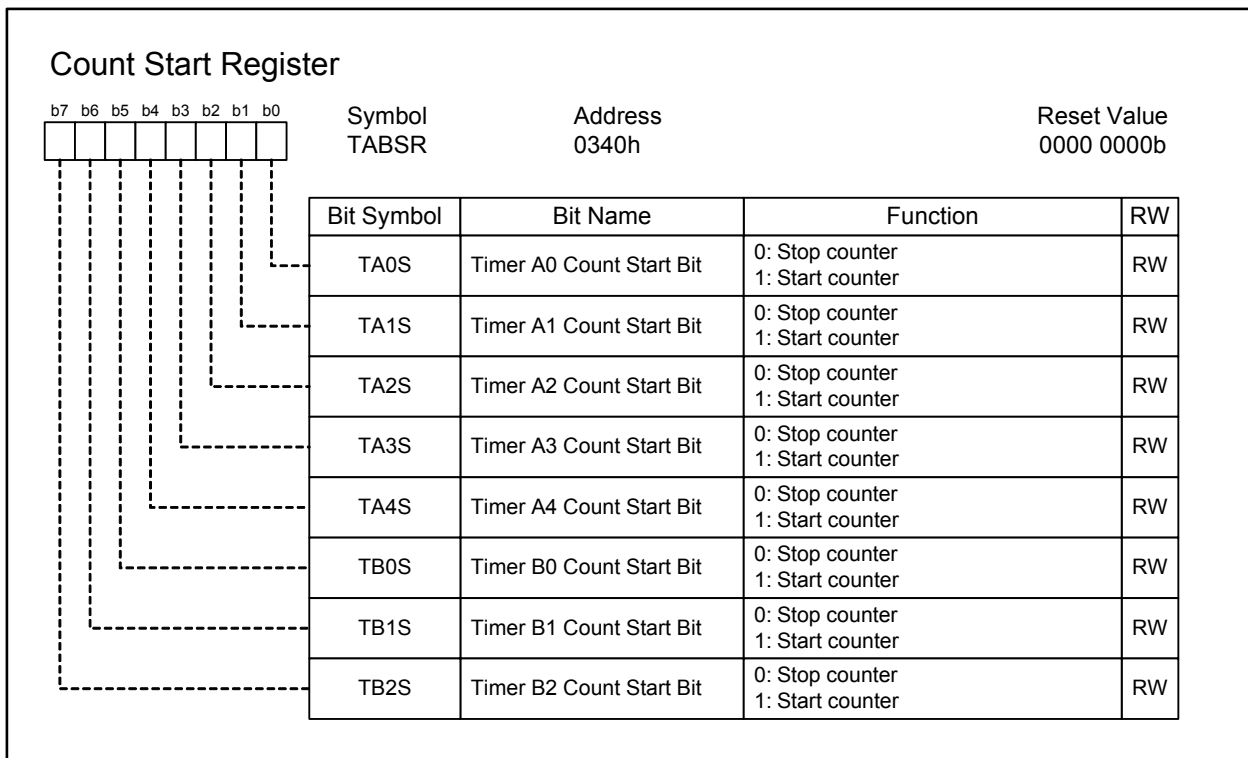


Figure 15.6 TABSR Register

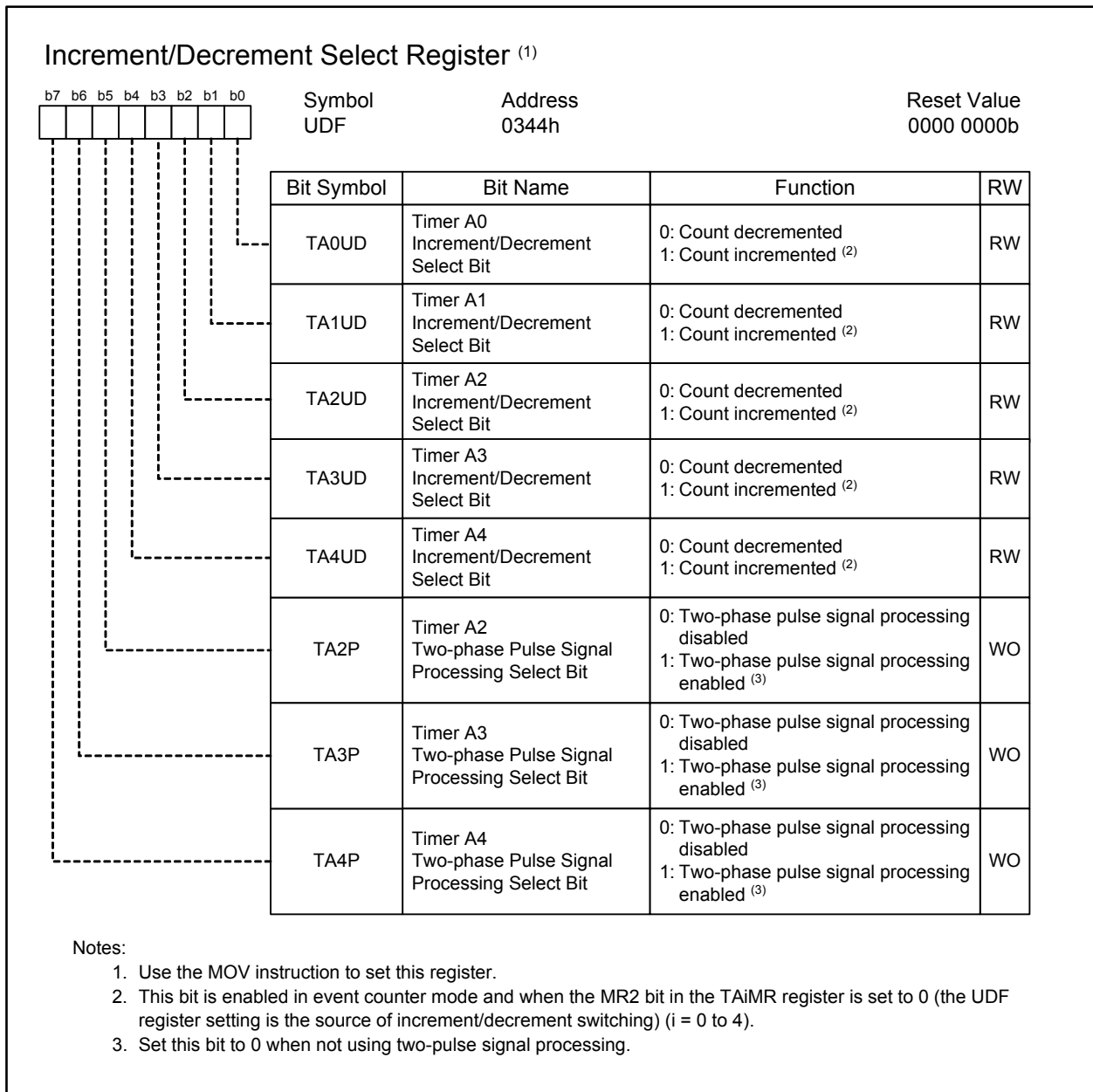
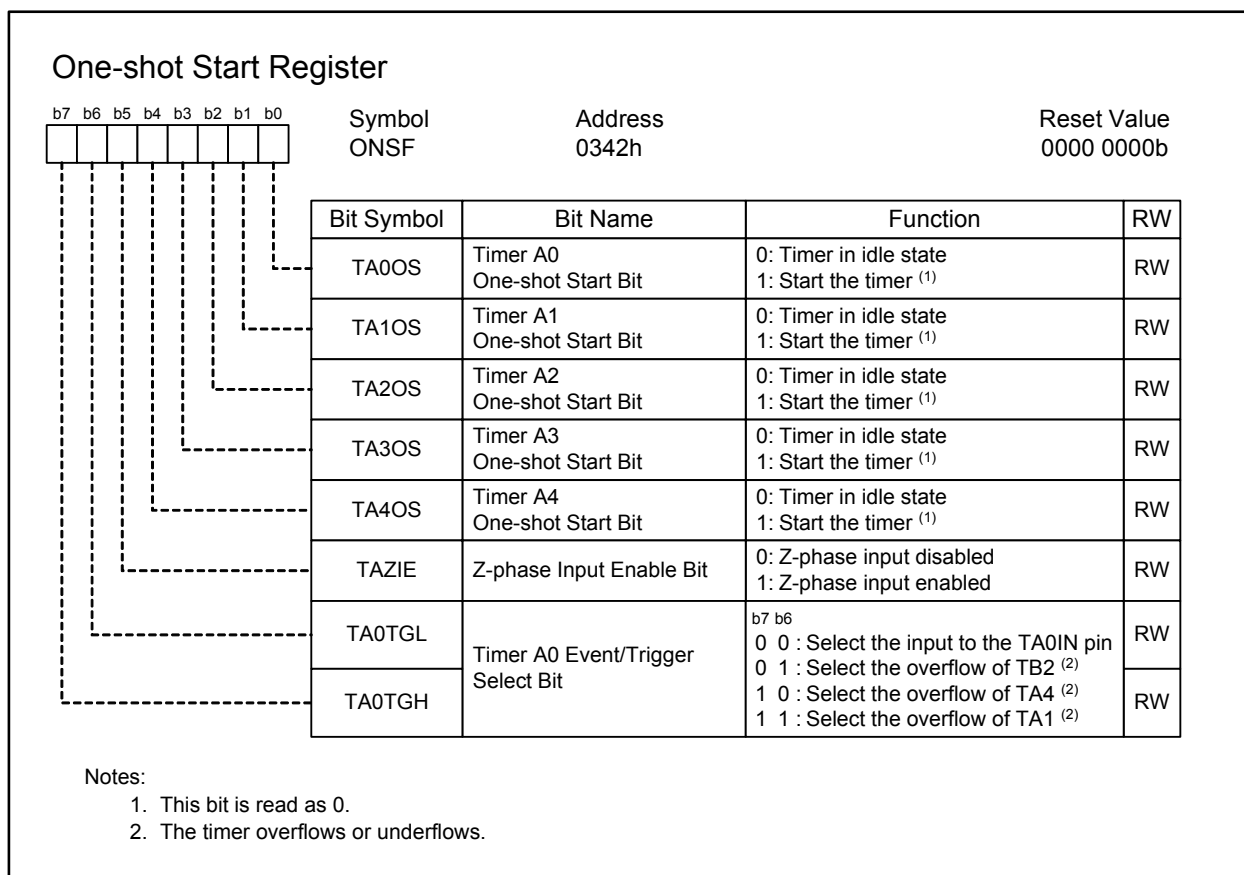
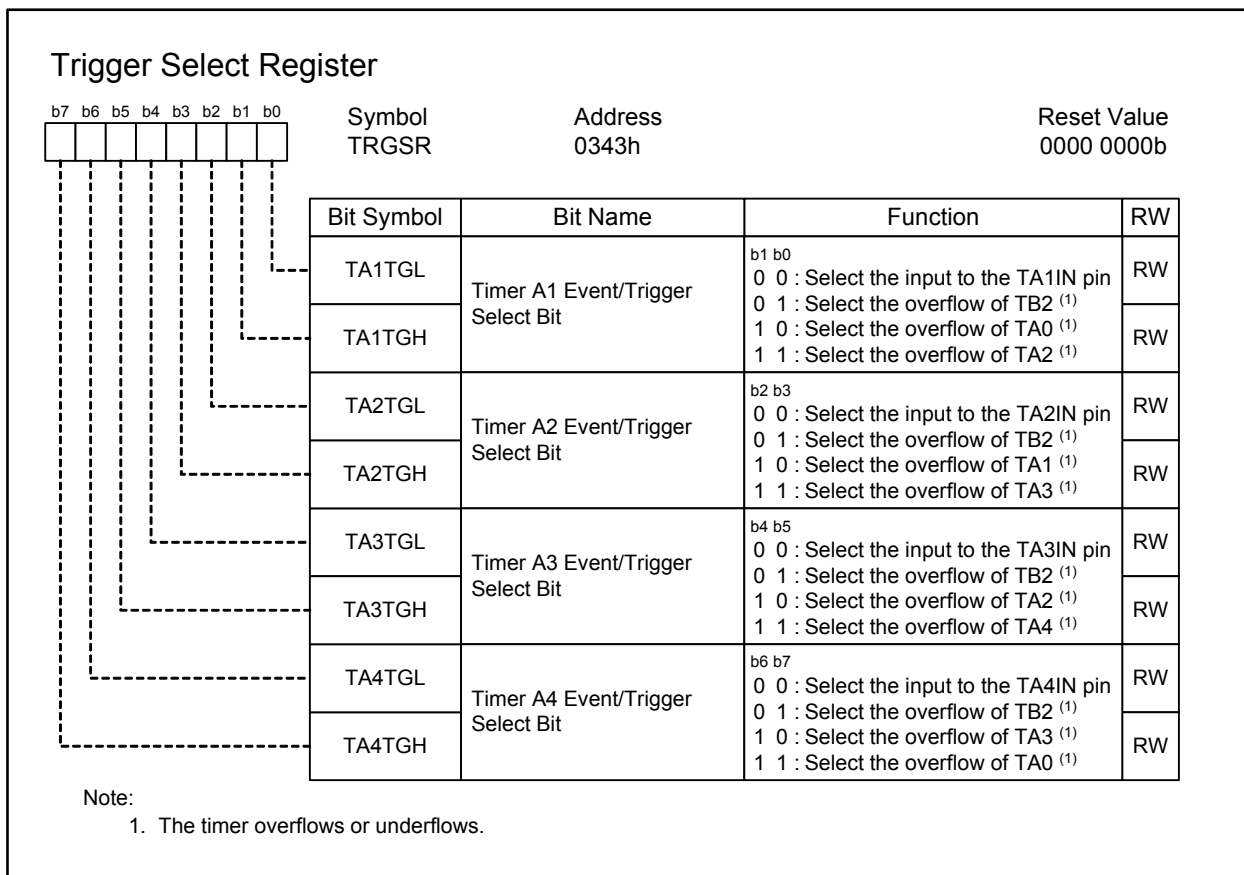


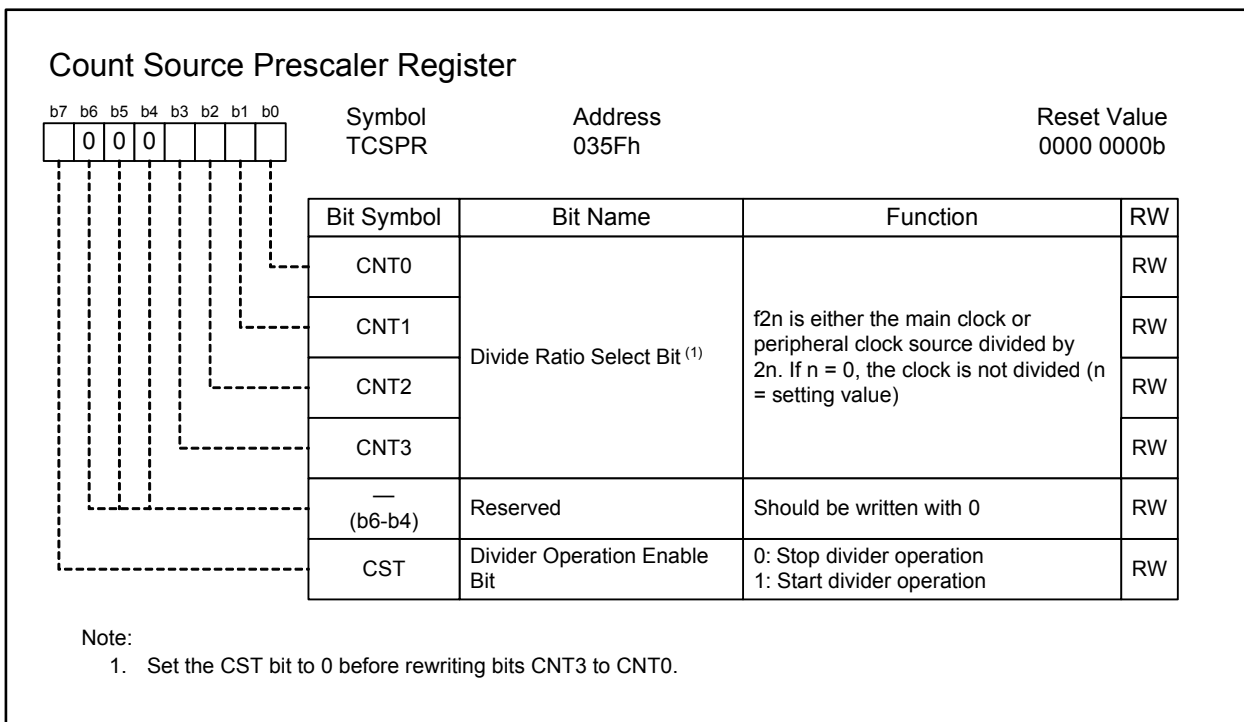
Figure 15.7 UDF Register



**Figure 15.8 ONSF Register**



**Figure 15.9 TRGSR Register**



**Figure 15.10 TCSPR Register**

### 15.1.1 Timer Mode

In timer mode, the timer counts an internally generated count source. Table 15.1 lists the specifications of timer mode. Figure 15.11 shows registers TA0MR to TA4MR in this mode.

**Table 15.1 Timer Mode Specifications (i = 0 to 4)**

Item	Specification
Count sources	f1, f8, f2n, or fC32
Count operations	<ul style="list-style-type: none"> <li>• Decrement</li> <li>• When the timer counter underflows, the reload register value is reloaded into the counter to continue counting</li> </ul>
Divide ratio	$\frac{1}{n+1}$ <i>n</i> : TAI register setting value, 0000h to FFFFh
Count start condition	The TAI <sub>S</sub> bit in the TABSR register is 1 (start counter)
Count stop condition	The TAI <sub>S</sub> bit in the TABSR register is 0 (stop counter)
Interrupt request generating timing	When the timer counter underflows
TAiIN pin function	Functions as a programmable I/O port or a gate input
TAiOUT pin function	Functions as a programmable I/O port or a pulse output
Read from timer	The TAI register indicates the counter value
Write to timer	<ul style="list-style-type: none"> <li>• While the timer counter is stopped or before the initial count source is input after starting to count, the value written to the TAI register is written to both the reload register and the counter</li> <li>• While the timer counter is running, the value written to the TAI register is written to the reload register (it is transferred to the counter at the next reload timing)</li> </ul>
Other functions	<ul style="list-style-type: none"> <li>• Gate function Input signal to the TAI<sub>IN</sub> pin can control the count start/stop</li> <li>• Pulse output function The polarity of the TAI<sub>OUT</sub> pin is inverted each time the timer counter underflows. A low is output while the TAI<sub>S</sub> bit holds 0 (stop counter)</li> </ul>



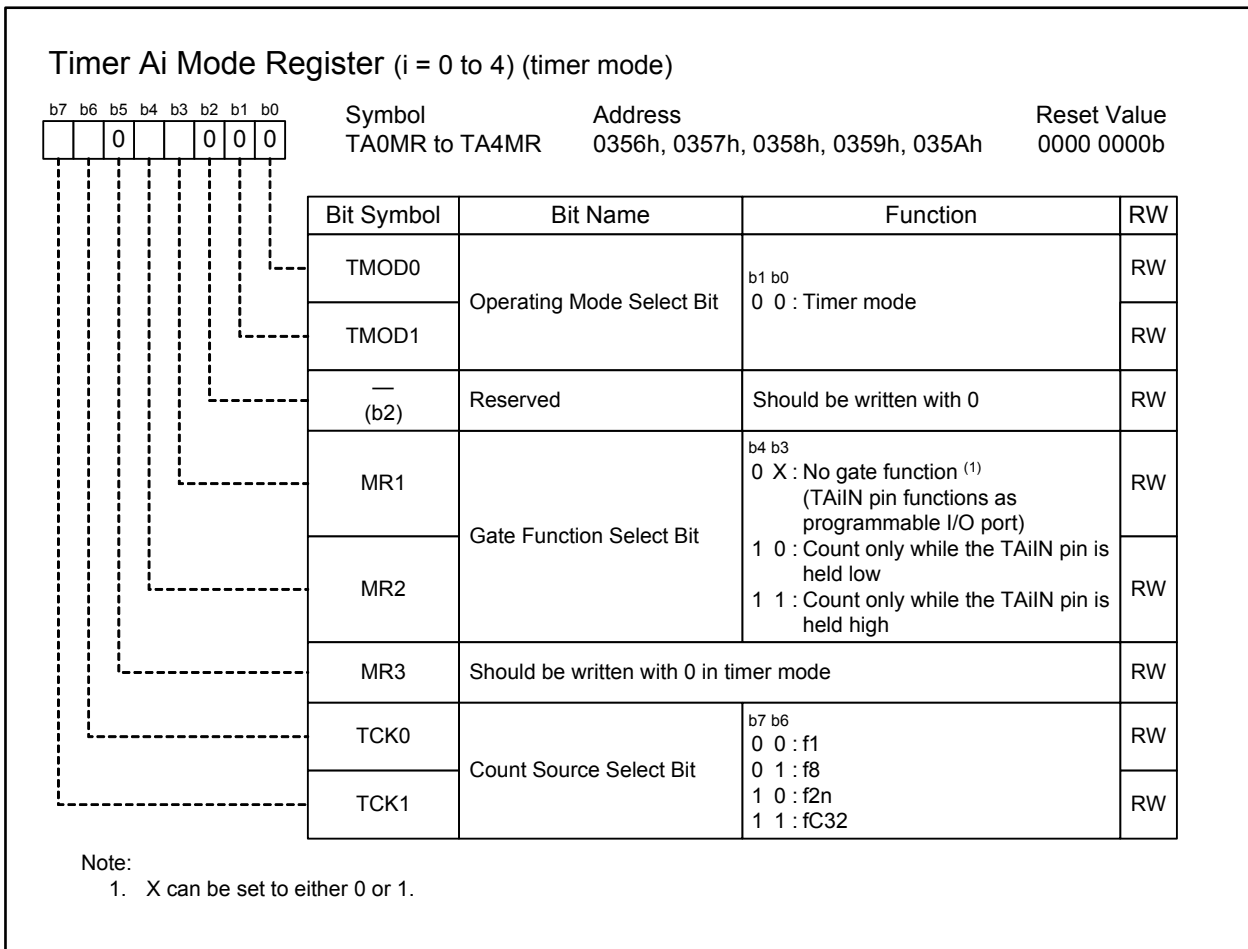


Figure 15.11 Registers TA0MR to TA4MR in Timer Mode

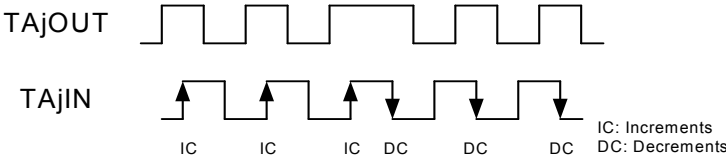
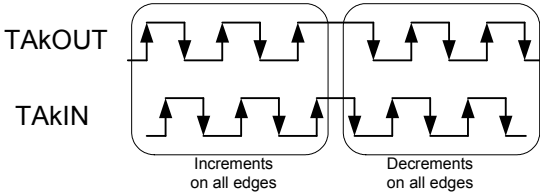
### 15.1.2 Event Counter Mode

In event counter mode, the timer counts an external signal or an overflow and underflow of other timers. Timers A2, A3, and A4 can count two-phase external signals. Table 15.2 lists the specifications in event count mode and Table 15.3 also lists the specifications when the timers use two-phase pulse signal processing. Figure 15.12 shows registers TA0MR to TA4MR in this mode.

**Table 15.2 Event Counter Mode Specifications (without two-phase pulse signal processing)  
(i = 0 to 4)**

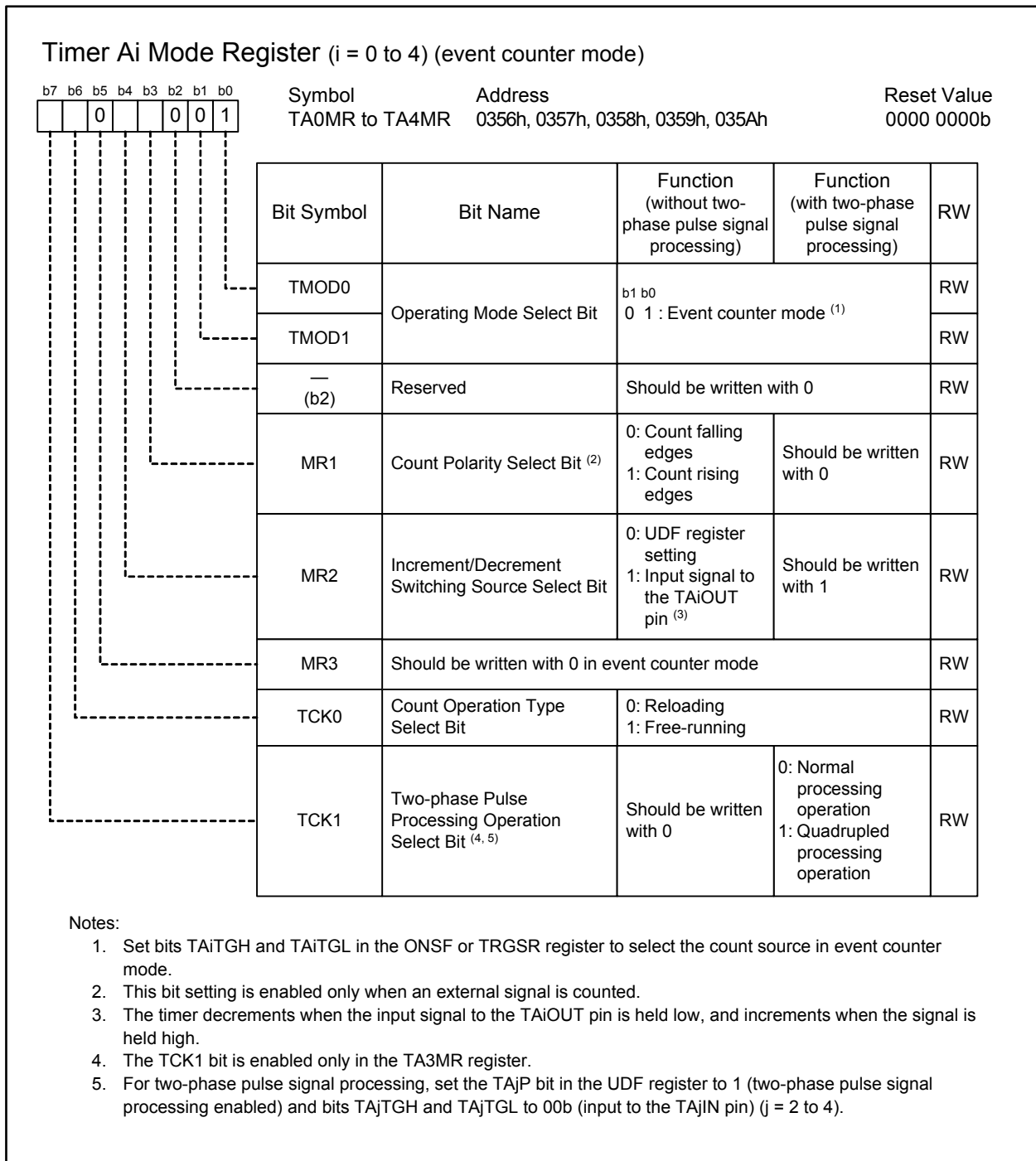
Item	Specification
Count sources	<ul style="list-style-type: none"> <li>External signal applied to the TAIIN pin (valid edge is selectable by a program)</li> <li>One of the following: the overflow and/or underflow signal of timer B2, the overflow and/or underflow signal of timer Aj (j = i - 1, or j = 4 if i = 0), or the overflow and/or underflow signal of timer Ak (k = i + 1, or k = 0 if i = 4)</li> </ul>
Count operations	<ul style="list-style-type: none"> <li>Increment/decrement can be switched by an external signal or program</li> <li>When the timer counter underflows or overflows, the reload register value is reloaded into the counter to continue counting. In a free-running count operation, the timer counter continues counting without reloading</li> </ul>
Divide ratio	<ul style="list-style-type: none"> <li><math>\frac{1}{FFFFh - n + 1}</math> when incrementing</li> <li><math>\frac{1}{n + 1}</math> when decrementing</li> </ul> n: TAI register setting value, 0000h to FFFFh
Count start condition	The TAI S bit in the TABSR register is 1 (start counter)
Count stop condition	The TAI S bit in the TABSR register is 0 (stop counter)
Interrupt request generating timing	When the timer counter overflows or underflows
TAiIN pin function	Functions as a programmable I/O port or a count source input
TAiOUT pin function	Functions as a programmable I/O port, a pulse output, or an input for switching between increment/decrement
Read from timer	The TAI register indicates a counter value
Write to timer	<ul style="list-style-type: none"> <li>While the timer counter is stopped or before the initial count source is input after starting to count, the value written to the TAI register is written to both the reload register and the counter</li> <li>While the timer counter is running, the value written to the TAI register is written to the reload register (it is transferred to the counter at the next reload timing)</li> </ul>
Other functions	<ul style="list-style-type: none"> <li>Free-running count function The reload register value is not reloaded even if the timer counter overflows or underflows</li> <li>Pulse output function The polarity of the TAIOUT pin is inverted whenever the timer counter overflows or underflows. A low is output while the TAI S bit holds 0 (stop counter)</li> </ul>

**Table 15.3 Event Counter Mode Specifications (with two-phase pulse signal processing on timers A2 to A4) (i = 2 to 4)**

Item	Specification
Count sources	Two-phase pulse signal applied to pins TAIIN and TAIOUT
Count operations	<ul style="list-style-type: none"> <li>Increment/decrement can be switched by a two-phase pulse signal</li> <li>When the timer counter underflows or overflows, the reload register value is reloaded into the counter to continue counting. In a free-running count operation, the timer counter continues counting without reloading</li> </ul>
Divide ratio	<ul style="list-style-type: none"> <li><math>\frac{1}{FFFFh - n + 1}</math> when incrementing</li> <li><math>\frac{1}{n + 1}</math> when decrementing</li> </ul> <i>n</i> : TAI register setting value, 0000h to FFFFh
Count start condition	The TAI <sub>S</sub> bit in the TABSR register is 1 (start counter)
Count stop condition	The TAI <sub>S</sub> bit in the TABSR register is 0 (stop counter)
Interrupt request generating timing	When the timer counter overflows or underflows
TAIIN pin function	A two-phase pulse input
TAIOUT pin function	A two-phase pulse input
Read from timer	The TAI register indicates a counter value
Write to timer	<ul style="list-style-type: none"> <li>While the timer counter is stopped or before the initial count source is input after starting to count, the value written to the TAI register is written to both the reload register and the counter</li> <li>While the timer counter is running, the value written to the TAI register is written to the reload register (it is transferred to the counter at the next reload timing)</li> </ul>
Other functions (1)	<ul style="list-style-type: none"> <li>Normal processing operation (timers A2 and A3) While the input signal applied to the TAJOUT pin is held high, the timer increments on the rising edge of the TAJIN pin and decrements on the falling edge (j = 2 or 3)</li> </ul>  <ul style="list-style-type: none"> <li>Quadrupled processing operation (timers A3 and A4) When the input signal applied to the TAKOUT pin is held high on the rising edge of the TAKIN pin, the timer increments on both the rising and falling edges of pins TAKOUT and TAKIN (k = 3 or 4). When the signal is held high on the falling edge of the TAKIN pin, the timer decrements on both the rising and falling edges of pins TAKOUT and TAKIN</li> </ul>  <ul style="list-style-type: none"> <li>Counter reset by Z-phase input (timer A3) The counter value is set to 0 by Z-phase input</li> </ul>

Note:

- Only timer A3 is available for any of the other functions. Timer A2 is exclusively for normal processing operations and timer A4 is for the quadrupled processing operation.



**Figure 15.12 Registers TA0MR to TA4MR in Event Counter Mode**

### 15.1.2.1 Counter Reset by Two-phase Pulse Signal Processing

A Z-phase input signal resets the timer counter when a two-phase pulse signal is being processed. This function can be used under the following conditions: timer A3 event counter mode, two-phase pulse signal processing, free-running count operation, and quadrupled processing. The Z-phase signal is applied to the INT2 pin.

When the TAZIE bit in the ONSF register is set to 1 (Z-phase input enabled), the timer counter can be reset by Z-phase input. To reset the counter, set the TA3 register to 0000h beforehand.

A Z-phase signal applied to the INT2 pin is detected on an edge. The edge polarity is selected using the POL bit in the INT2IC register. The Z-phase signal should be input in order to have a pulse width of one or more count source cycles for timer A3. Figure 15.13 shows the two-phase pulse (phases A and B) and the Z-phase.

The timer counter is reset at the initial count source input after Z-phase input is detected. Figure 15.14 shows the counter reset timing.

When timer A3 overflows or underflows during a reset by the Z-phase input, two timer A3 interrupt requests are successively generated. To avoid this, the timer A3 interrupt request should not be used when using this function.

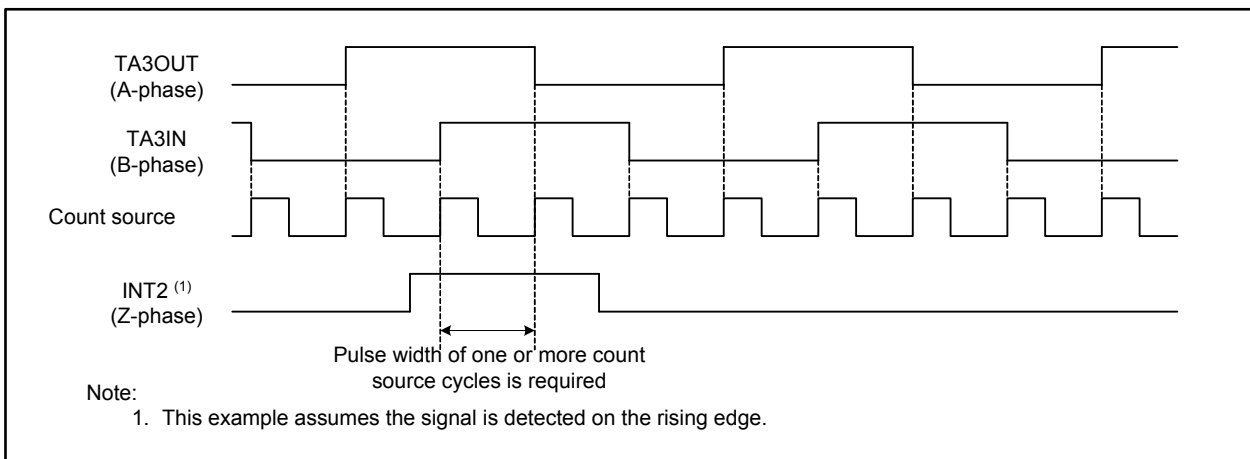


Figure 15.13 Two-phase Pulse (phases A and B) and Z-phase

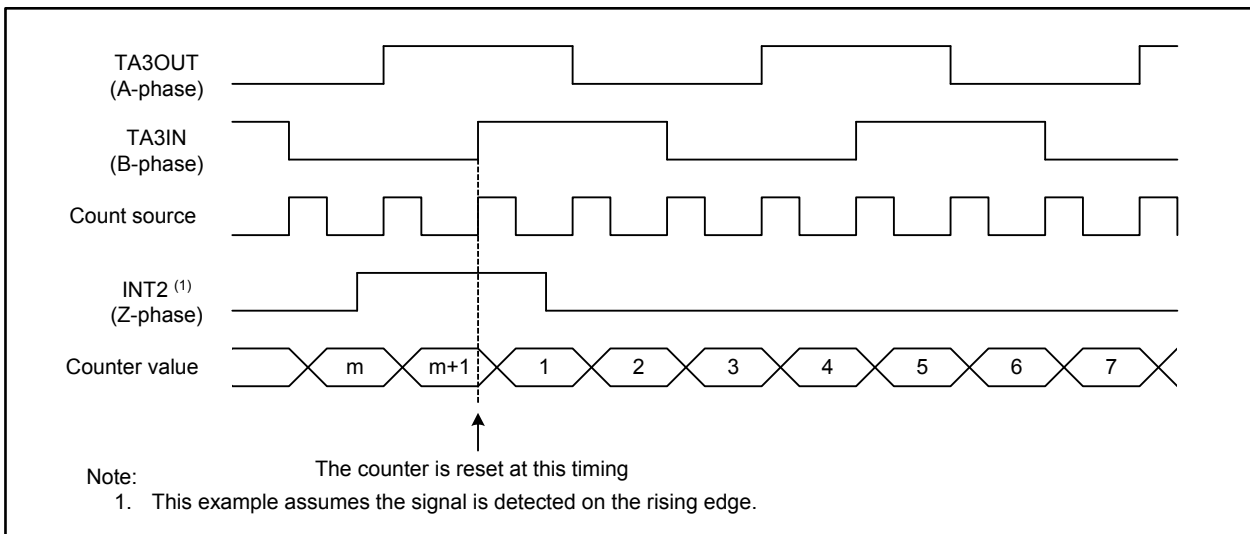


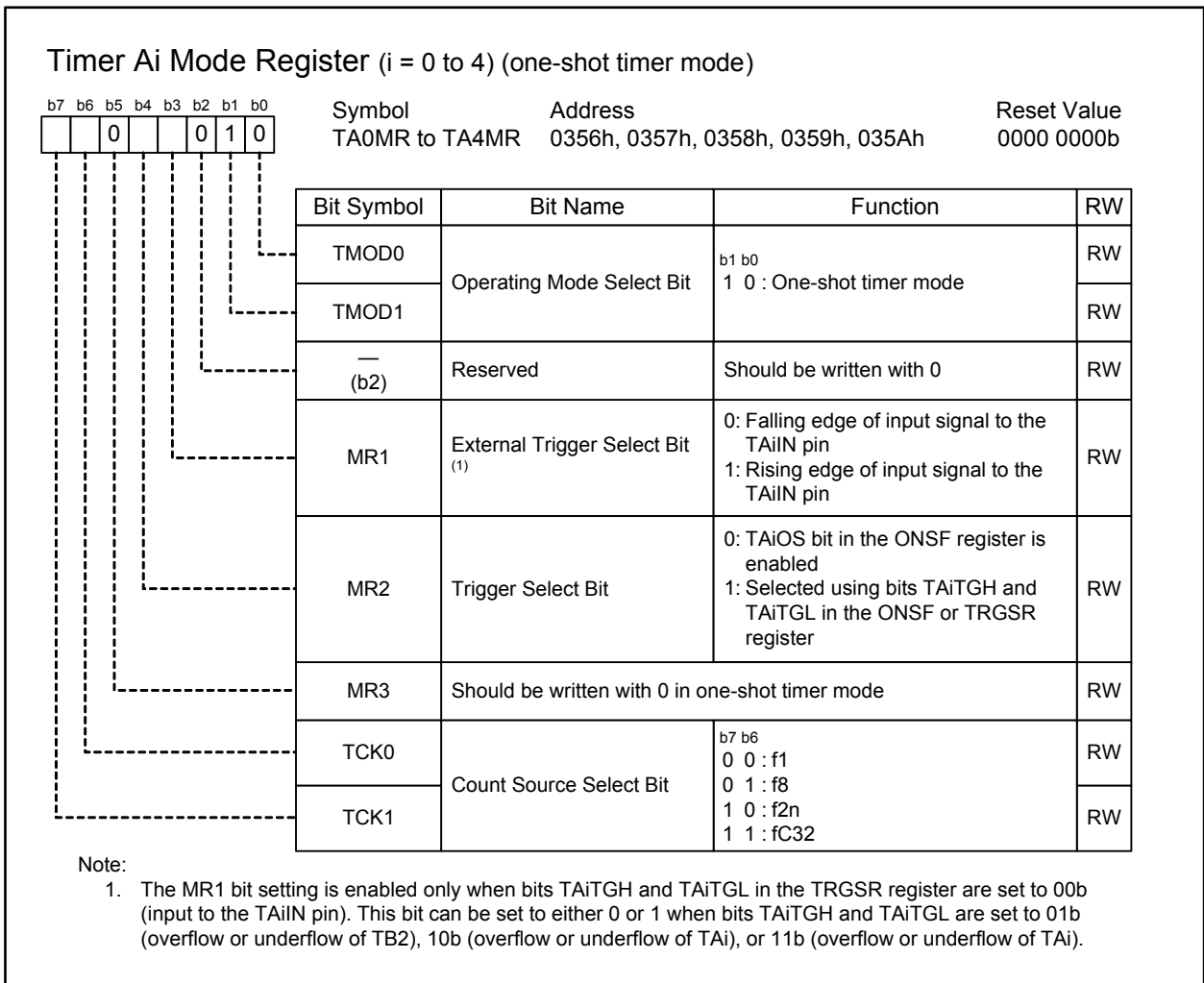
Figure 15.14 Counter Reset Timing

### 15.1.3 One-shot Timer Mode

In one-shot timer mode, the timer operates only once for each trigger. Table 15.4 lists specifications of one-shot timer mode. Once a trigger occurs, the timer starts and operates for a given period. Figure 15.15 shows registers TA0MR to TA4MR in this mode.

**Table 15.4 One-shot Timer Mode Specifications (i = 0 to 4)**

Item	Specification
Count sources	f1, f8, f2n, or fC32
Count operations	<ul style="list-style-type: none"> <li>Decrement</li> <li>When the timer counter reaches 0000h, it stops running after the reload register value is reloaded</li> <li>When a trigger occurs while counting, the reload register value is reloaded into the counter to continue counting</li> </ul>
Divide ratio	$\frac{1}{n}$ n: TAI register setting value, 0000h to FFFFh (Note that the timer counter does not run if n = 0000h)
Count start conditions	<p>The TAI<sub>S</sub> bit in the TABSR register is 1 (start counter) and any of following triggers occurs:</p> <ul style="list-style-type: none"> <li>An external trigger applied to the TAI<sub>IN</sub> pin</li> <li>One of the following: the overflow and/or underflow signal of timer B2, the overflow and/or underflow signal of timer A<sub>j</sub> (j = i - 1, or j = 4 if i = 0), or the overflow and/or underflow signal of timer A<sub>k</sub> (k = i + 1, or k = 0 if i = 4)</li> <li>The TAI<sub>OS</sub> bit in the ONSF register is 1 (start the timer)</li> </ul>
Count stop conditions	<ul style="list-style-type: none"> <li>The timer counter reaches 0000h and the reload register value is reloaded</li> <li>The TAI<sub>S</sub> bit in the TABSR register is 0 (stop counter)</li> </ul>
Interrupt request generating timing	When the timer counter reaches 0000h
TAI <sub>IN</sub> pin function	A programmable I/O port or a trigger input
TAI <sub>OUT</sub> pin function	A programmable I/O port or a pulse output
Read from timer	The TAI register indicates an undefined value
Write to timer	<ul style="list-style-type: none"> <li>While the timer counter is stopped or before the initial count source is input after starting to count, the value written to the TAI register is written to both the reload register and the counter</li> <li>While the timer counter is running, the value written to the TAI register is written to the reload register (it is transferred to the counter at the next reload timing)</li> </ul>
Other function	<ul style="list-style-type: none"> <li>Pulse output function</li> </ul> <p>A low is output while the timer counter is stopped and a high is output while the timer counter is running</p>



**Figure 15.15 Registers TA0MR to TA4MR in One-shot Timer Mode**

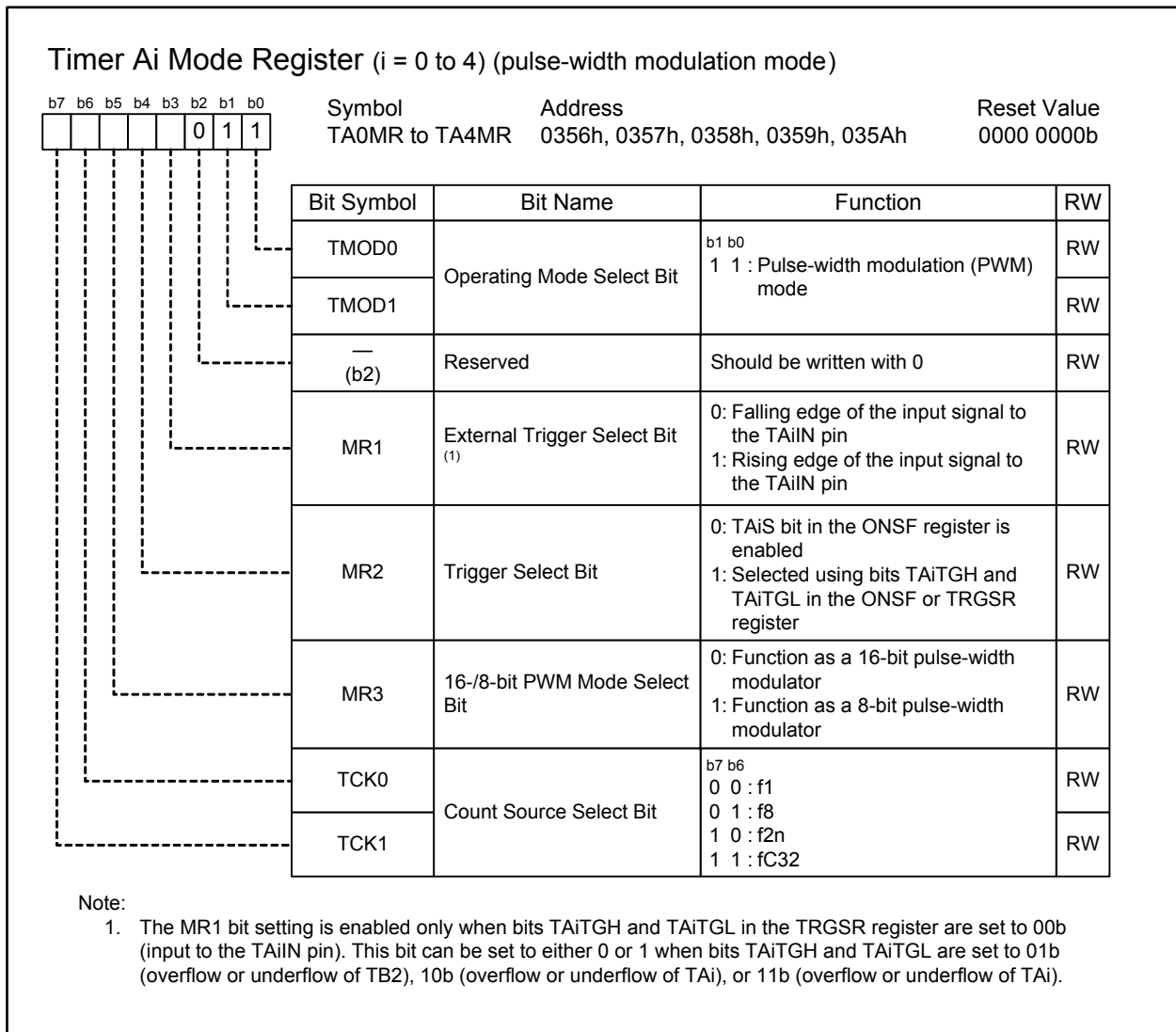
### 15.1.4 Pulse-width Modulation Mode

In pulse-width modulation mode, the timer outputs pulses of given width successively. Table 15.5 lists specifications of pulse-width modulation mode. The timer counter functions as either a 16-bit or 8-bit pulse-width modulator. Figure 15.16 shows registers TA0MR to TA4MR in this mode. Figures 15.17 and 15.18 show operation examples of 16-bit and 8-bit pulse-width modulators.

**Table 15.5 Pulse-width Modulation Mode Specifications (i = 0 to 4)**

Item	Specification
Count sources	f1, f8, f2n, or fC32
Count operations	<ul style="list-style-type: none"> <li>Decrement (the timer counter functions as an 8-bit or a 16-bit pulse-width modulator)</li> <li>The reload register value is reloaded on the rising edge of a PWM pulse to continue counting</li> <li>The timer is not affected by a trigger that occurs while the counter is running</li> </ul>
16-bit PWM	<ul style="list-style-type: none"> <li>High level width: <math>\frac{n}{fj}</math> <i>n</i>: TAI register setting value, 0000h to FFFEh <i>fj</i>: Count source frequency</li> <li>Cycles: fixed to <math>\frac{2^{16}-1}{fj}</math></li> </ul>
8-bit PWM	<ul style="list-style-type: none"> <li>High level width: <math>\frac{n \times (m+1)}{fj}</math></li> <li>Cycles: <math>\frac{(2^8-1) \times (m+1)}{fj}</math> <i>n</i>: Upper byte of the TAI register setting value, 00h to FEh <i>m</i>: Lower byte of the TAI register setting value, 00h to FFh</li> </ul>
Count start conditions	<ul style="list-style-type: none"> <li>The TAI<sub>S</sub> bit in the TABSR register is 1 (start counter)</li> <li>The TAI<sub>S</sub> bit is 1 and an external trigger is applied to the TAI<sub>IN</sub> pin</li> <li>The TAI<sub>S</sub> bit is 1 and any of following triggers occurs: the overflow and/or underflow signal of timer B2, the overflow and/or underflow signal of timer A<sub>j</sub> (j = i - 1, or j = 4 if i = 0), or the overflow and/or underflow signal of timer A<sub>k</sub> (k = i + 1, or k = 0 if i = 4)</li> </ul>
Count stop condition	The TAI <sub>S</sub> bit in the TABSR register is 0 (stop counter)
Interrupt request generating timing	On the falling edge of the PWM pulse
TAI <sub>IN</sub> pin function	A programmable I/O port or trigger input
TAI <sub>OUT</sub> pin function	A pulse output
Read from timer	The TAI register indicates an undefined value
Write to timer	<ul style="list-style-type: none"> <li>While the timer counter is stopped or before the initial count source is input after starting to count, the value written to the TAI register is written to both the reload register and the counter</li> <li>While the timer counter is running, the value written to the TAI register is written to the reload register (it is transferred to the counter at the next reload timing)</li> </ul>





**Figure 15.16 Registers TA0MR to TA4MR in Pulse-width Modulation Mode**

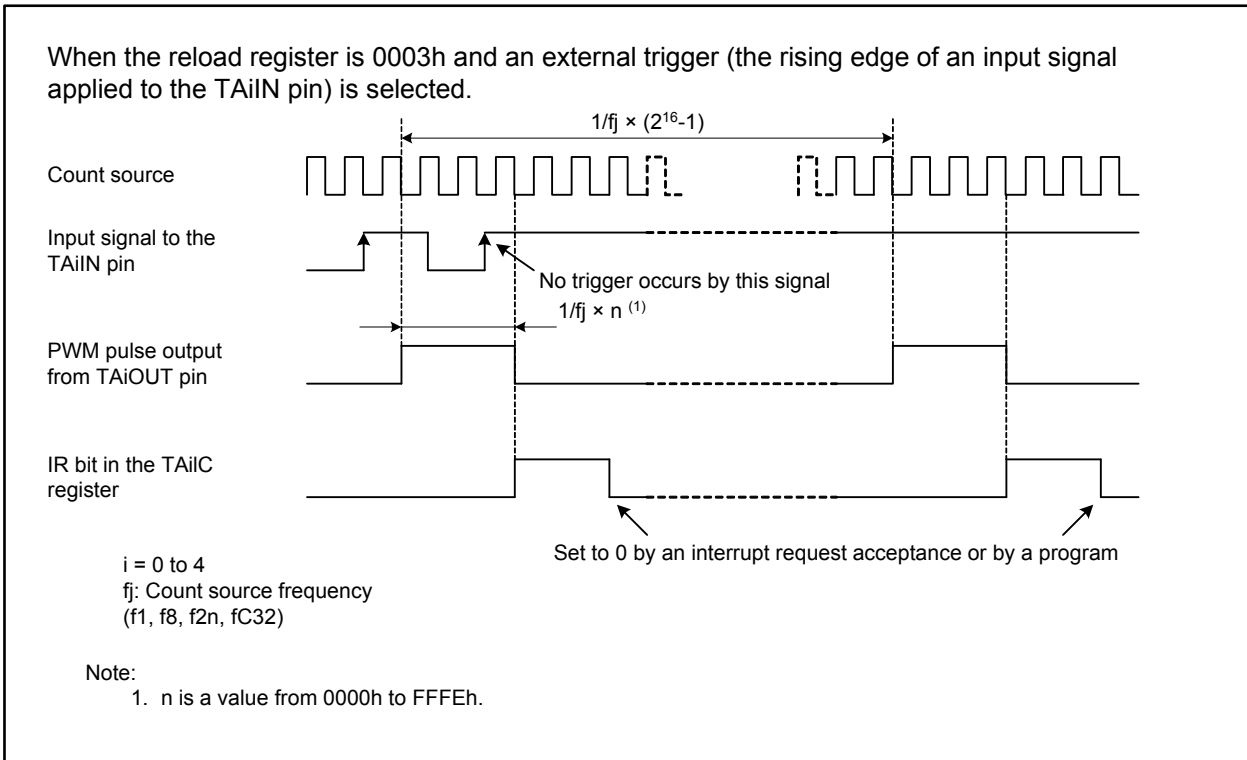


Figure 15.17 16-bit Pulse-width Modulator Operation

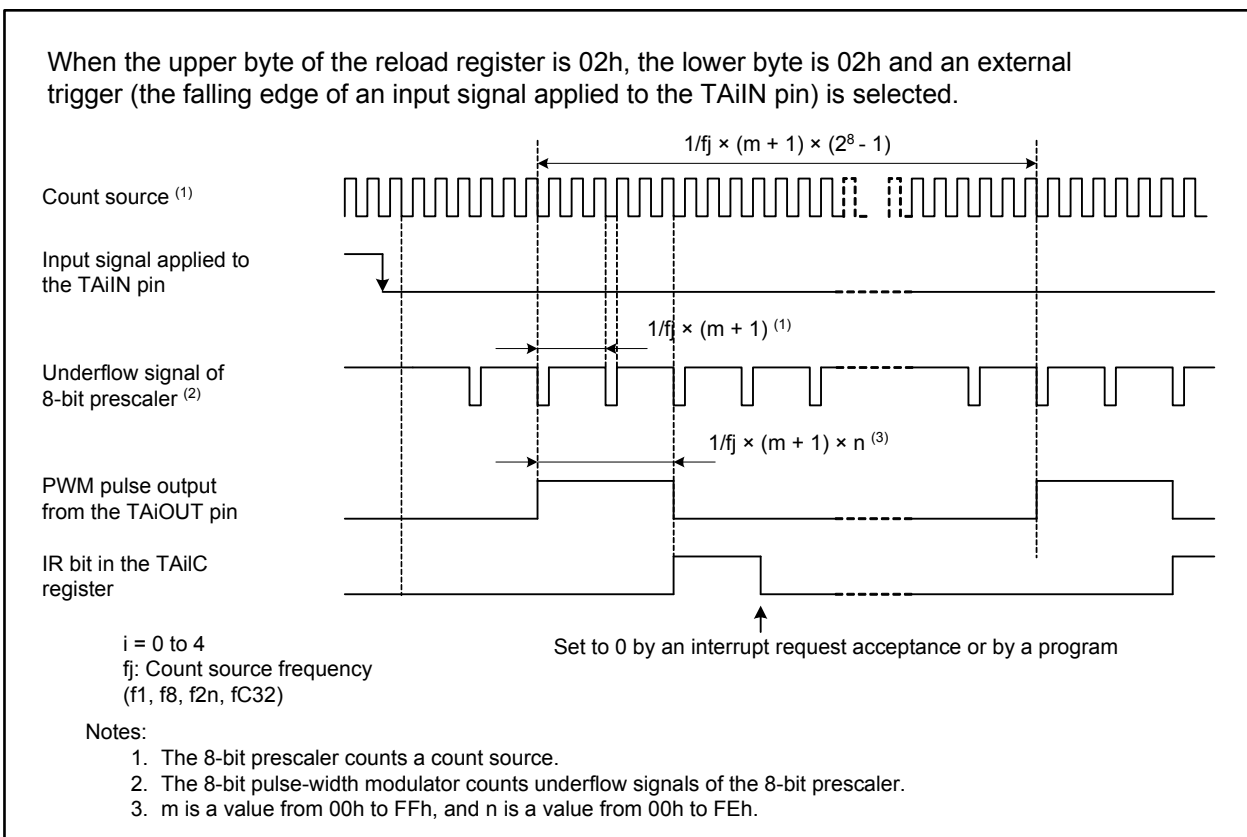


Figure 15.18 8-bit Pulse-width Modulator Operation

## 15.2 Timer B

Figure 15.19 shows a block diagram of timer B, and Figure 15.20 to Figure 15.23 show registers associated with timer B.

Timer B supports the three modes shown below. Select a mode by setting bits TMOD1 and TMOD0 in the TBiMR register ( $i = 0$  to 5).

- Timer mode: The timer counts an internal count source.
- Event counter mode: The timer counts an external pulse or an overflow and underflow of other timers.
- Pulse period/pulse-width measure mode: The timer measures the pulse period or pulse width of an external signal.

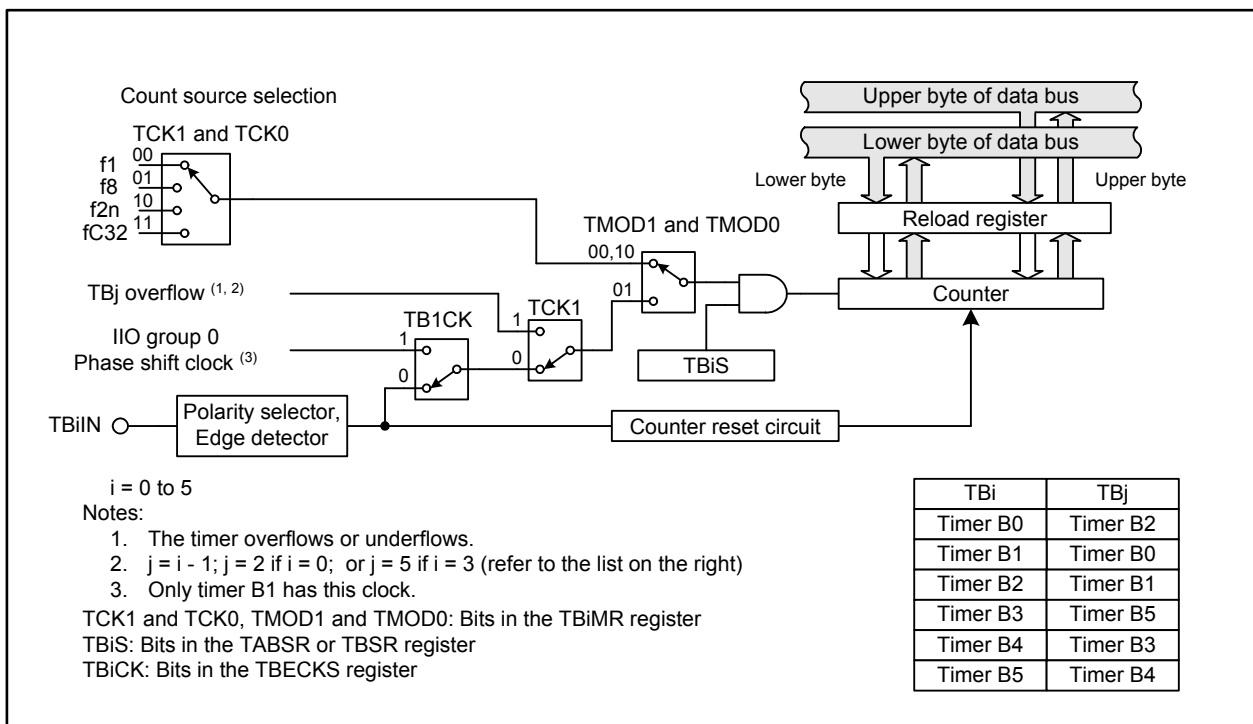


Figure 15.19 Timer B Block Diagram

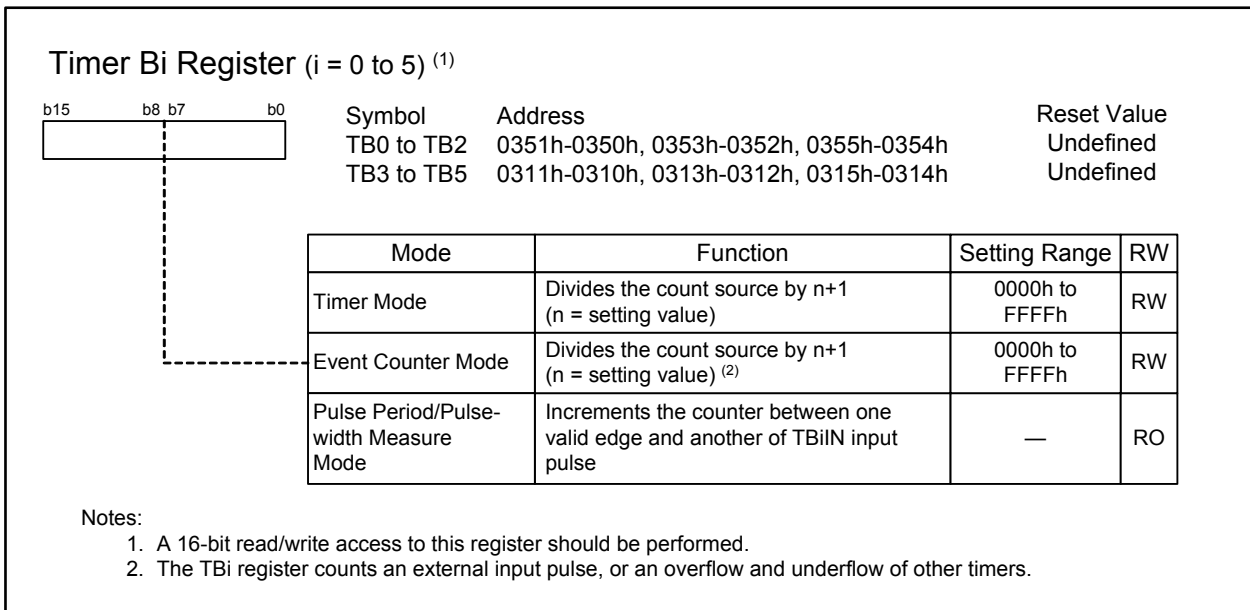


Figure 15.20 Registers TB0 to TB5

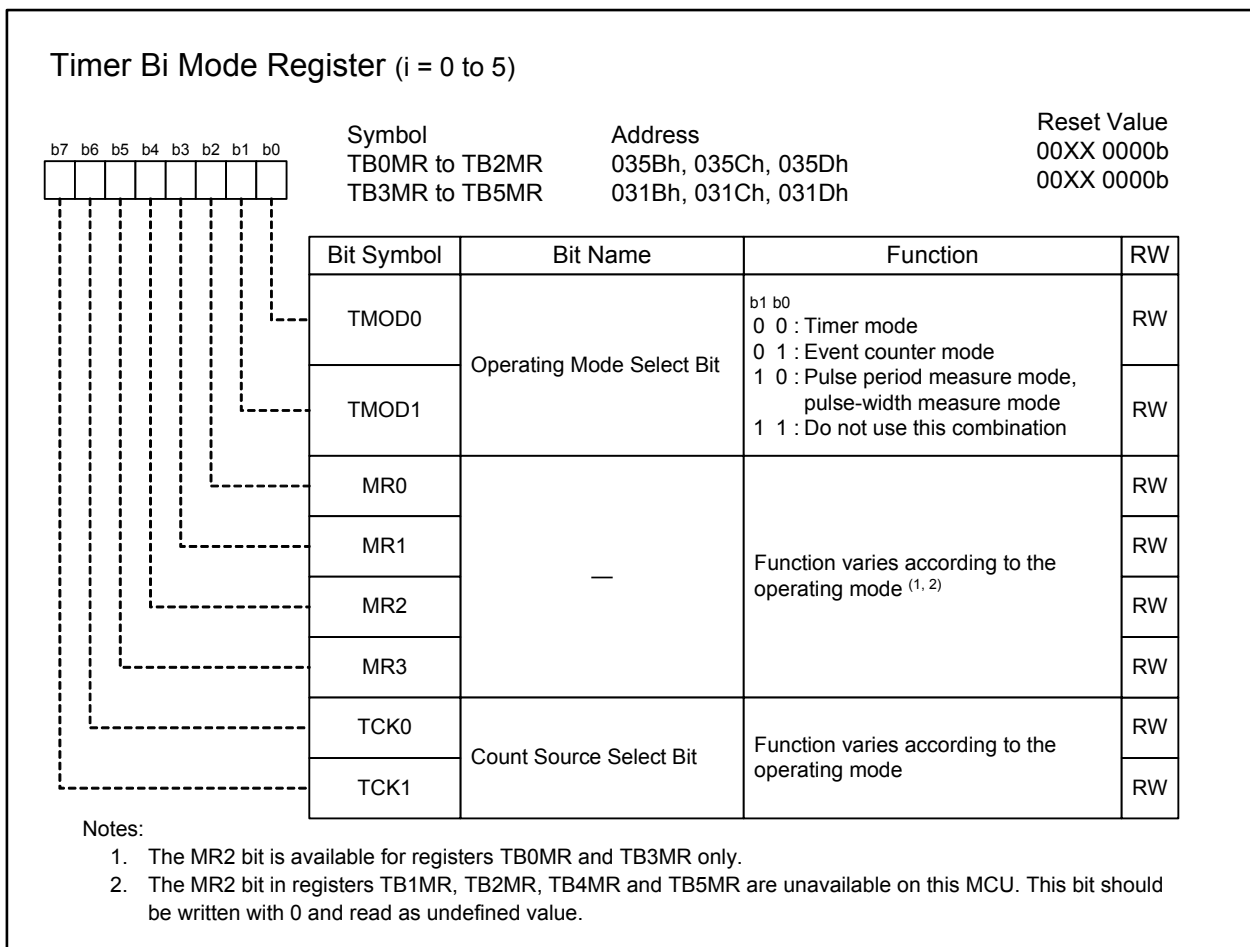


Figure 15.21 Registers TB0MR to TB5MR

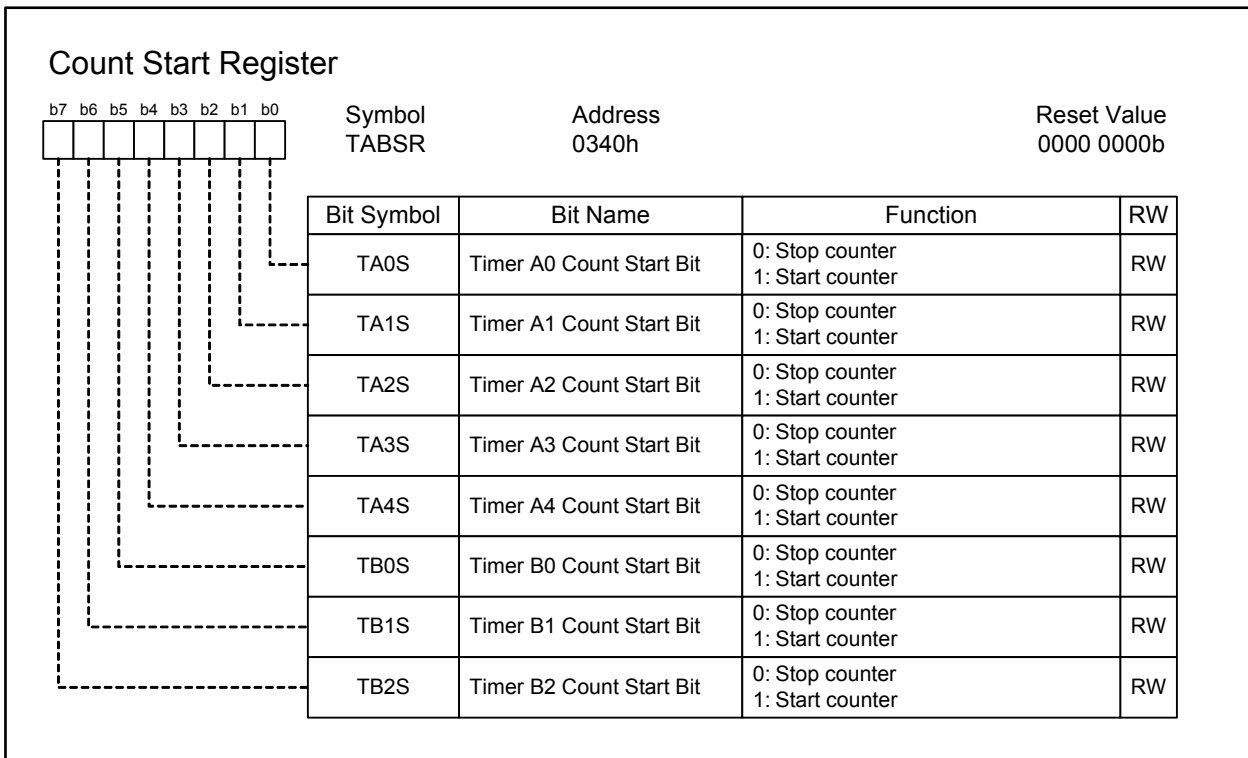


Figure 15.22 TABSR Register

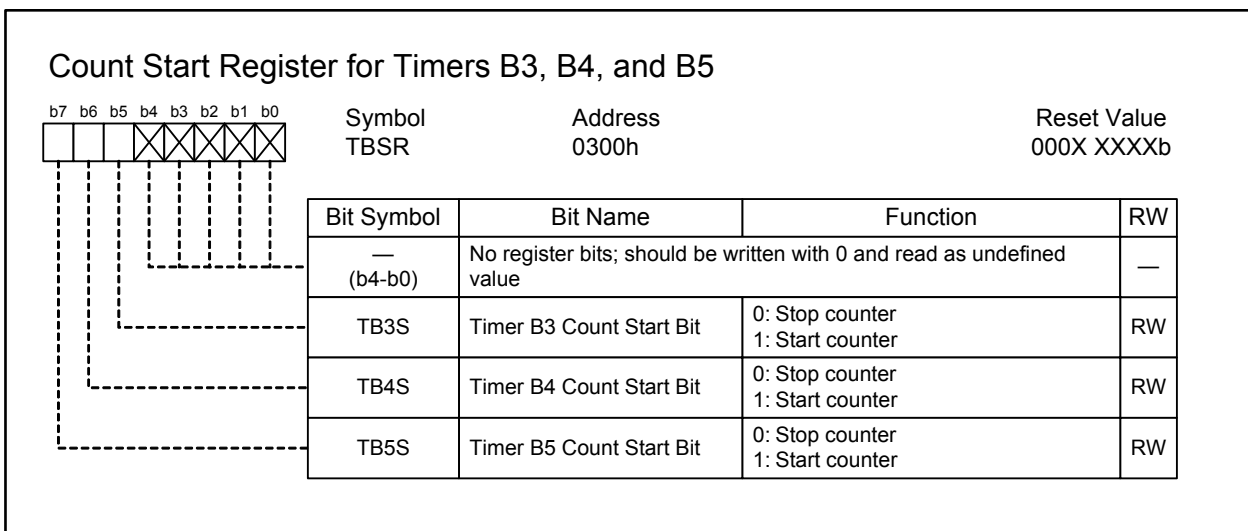


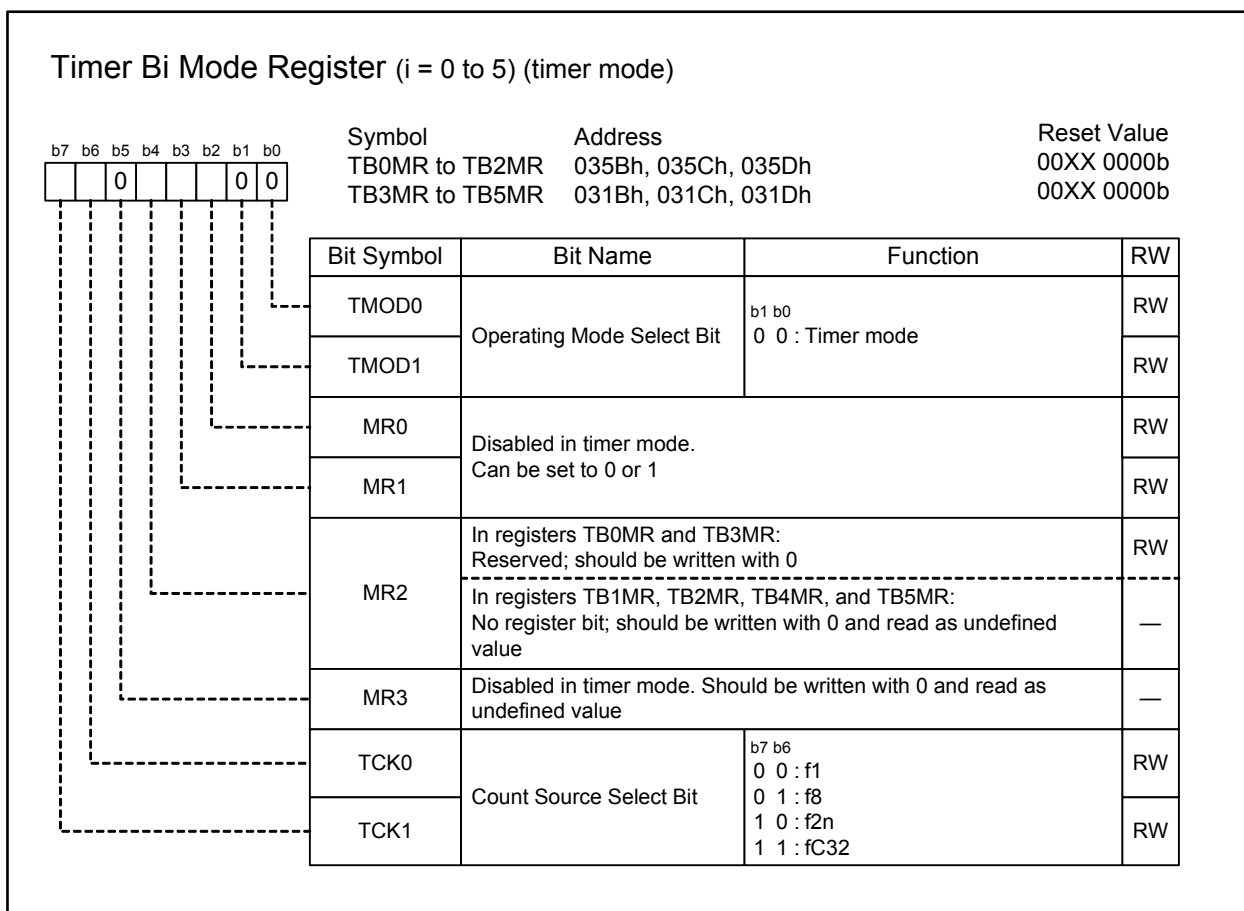
Figure 15.23 TBSR Register

### 15.2.1 Timer Mode

In timer mode, the timer counts an internally generated count source. Table 15.6 lists specifications of timer mode. Figure 15.24 shows registers TB0MR to TB5MR in this mode.

**Table 15.6 Timer Mode Specifications (i = 0 to 5)**

Item	Specification
Count sources	f1, f8, f2n, or fC32
Count operations	<ul style="list-style-type: none"> <li>• Decrement</li> <li>• When the timer counter underflows, the reload register value is reloaded into the counter to continue counting</li> </ul>
Divide ratio	$\frac{1}{n+1}$ <i>n</i> : TBi register setting value, 0000h to FFFFh
Count start condition	The TBiS bit in the TABSR or TBSR register is 1 (start counter)
Count stop condition	The TBiS bit in the TABSR or TBSR register is 0 (stop counter)
Interrupt request generating timing	When the timer counter underflows
TBiIN pin function	Functions as a programmable I/O port
Read from timer	The TBi register indicates a counter value
Write to timer	<ul style="list-style-type: none"> <li>• While the timer counter is stopped or before the initial count source is input after starting to count, the value written to the TBi register is written to both the reload register and the counter</li> <li>• While the timer counter is running, the value written to the TBi register is written to the reload register (it is transferred to the counter at the next reload timing)</li> </ul>



**Figure 15.24 Registers TB0MR to TB5MR in Timer Mode**

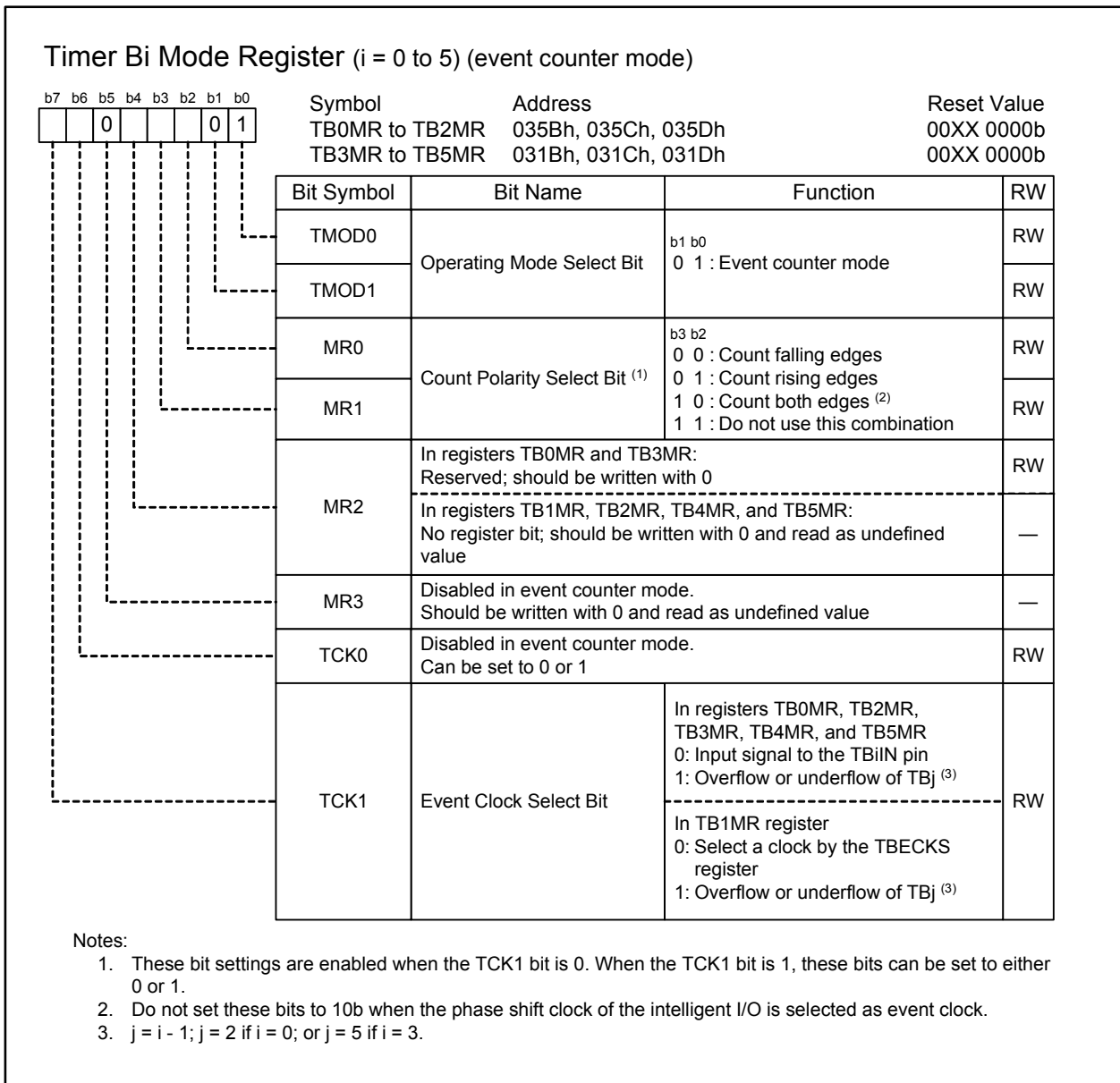
### 15.2.2 Event Counter Mode

In event counter mode, the timer counts an external signal or the overflow or underflow of other timers. Table 15.7 lists specifications of event counter mode. Figure 15.25 shows the TBiMR register in this mode (i = 0 to 5).

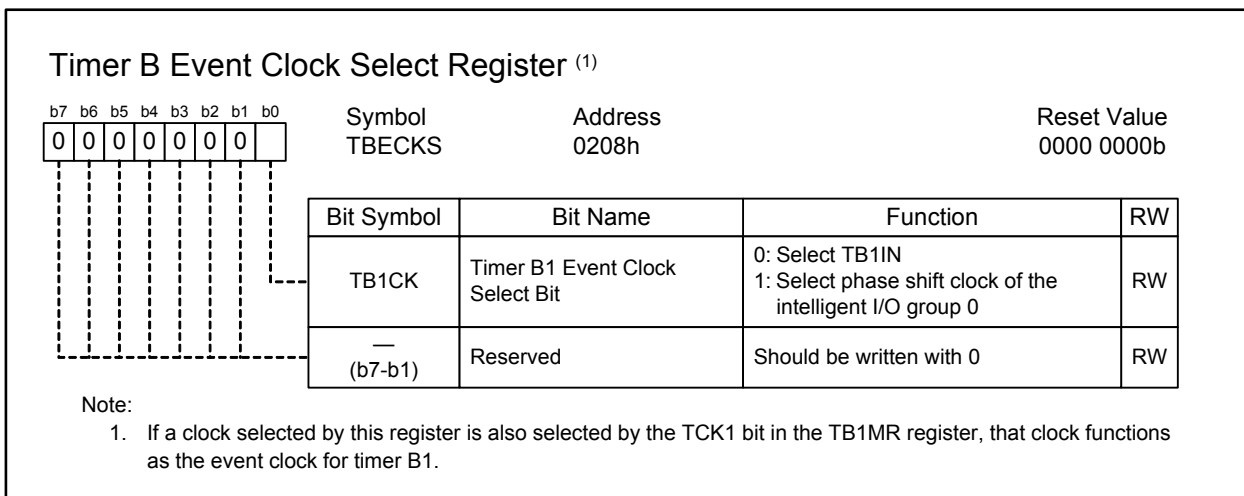
**Table 15.7 Event Counter Mode Specifications (i = 0 to 5)**

Item	Specification
Count sources	<ul style="list-style-type: none"> <li>External signal applied to the TBiIN pin (valid edge is selectable among the falling edge, the rising edge, or both)</li> <li>The overflow or underflow signal of TBj (j = i - 1; j = 2 if i = 0; or j = 5 if i = 3)</li> <li>Phase shift clock of the intelligent I/O group 0</li> </ul>
Count operations	<ul style="list-style-type: none"> <li>Decrement</li> <li>When the timer counter underflows, the reload register value is reloaded into the counter to continue counting</li> </ul>
Divide ratio	$\frac{1}{n+1}$ n: TBi register setting value, 0000h to FFFFh
Count start condition	The TBiS bit in the TABSR or TBSR register is 1 (start counter)
Count stop condition	The TBiS bit in the TABSR or TBSR register is 0 (stop counter)
Interrupt request generation timing	When the timer counter underflows
TBiIN pin function	Functions as a programmable I/O port or count source input
Read from timer	The TBi register indicates a counter value
Write to timer	<ul style="list-style-type: none"> <li>While the timer counter is stopped or before the initial count source is input after starting to count, the value written to the TBi register is written to both the reload register and the counter</li> <li>While the timer counter is running, the value written to the TBi register is written to the reload register (it is transferred to the counter at the next reload timing)</li> </ul>





**Figure 15.25 Registers TB0MR to TB5MR in Event Counter Mode**



**Figure 15.26 TBECKS Register**

### 15.2.3 Pulse Period/Pulse-width Measure Mode

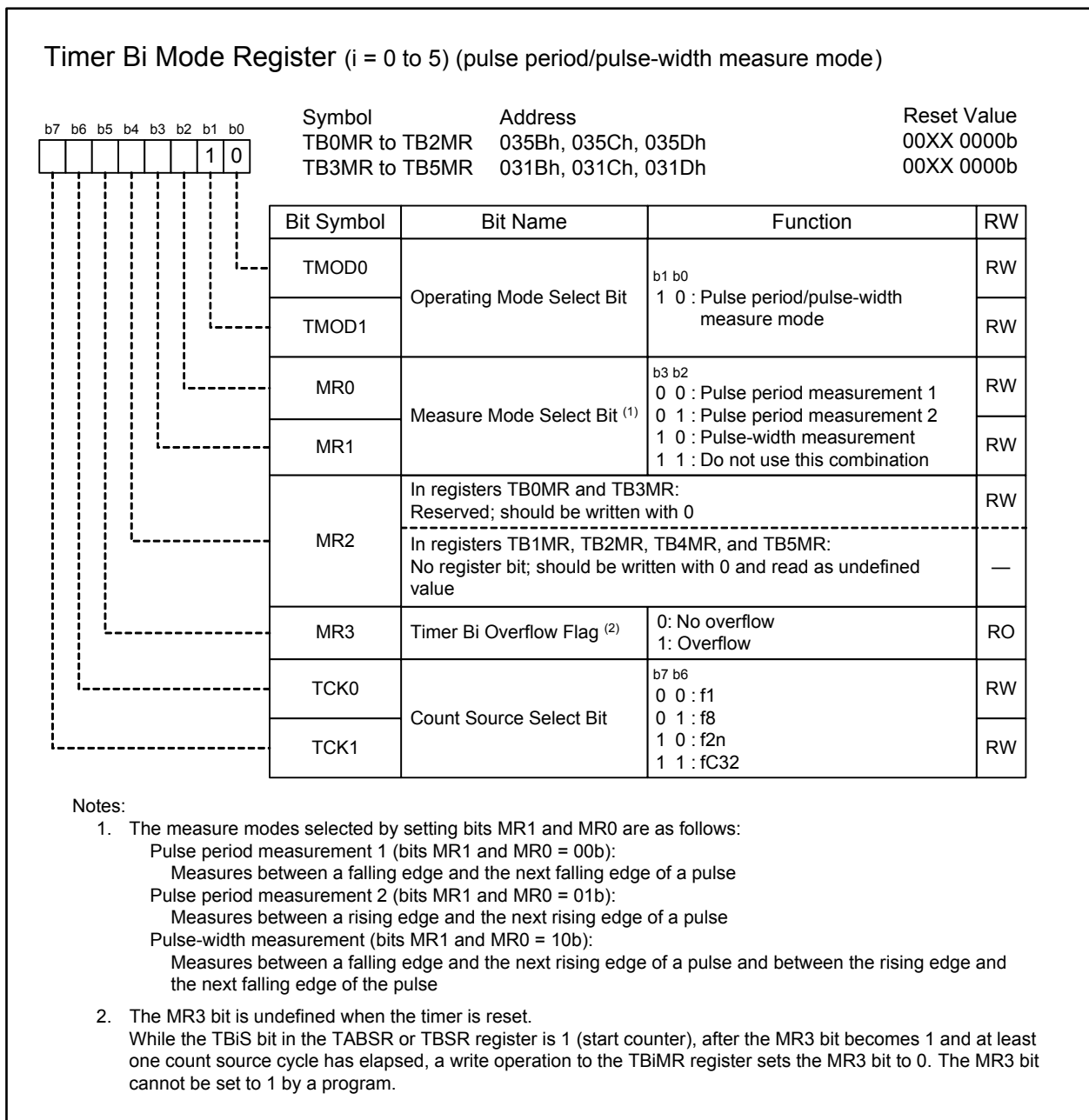
In pulse period/pulse-width measure mode, the timer measures the pulse period or pulse width of an external signal. Table 15.8 lists specifications of the pulse period/pulse-width measure mode. Figure 15.27 shows registers TB0MR to TB5MR in this mode. Figures 15.28 and 15.29 show an operation example of pulse period measurement and pulse-width measurement, respectively.

**Table 15.8 Pulse Period/Pulse-width Measure Mode Specifications (i = 0 to 5)**

Item	Specification
Count sources	f1, f8, f2n, or fC32
Count operations	<ul style="list-style-type: none"> <li>• Increment</li> <li>• The counter value is transferred to the reload register on the valid edge of a pulse to be measured, then it is set to 0000h to resume counting</li> </ul>
Count start condition	The TBiS bit in the TABSR or TBSR register is 1 (start counter)
Count stop condition	The TBiS bit in the TABSR or TBSR register is 0 (stop counter)
Interrupt request generating timing	<ul style="list-style-type: none"> <li>• On the valid edge of a pulse to be measured <sup>(1)</sup></li> <li>• When the timer counter overflows (when the MR3 bit in the TBIMR register becomes 1 (overflow)) <sup>(2)</sup></li> </ul>
TBiIN pin function	A pulse input to be measured
Read from timer	The TBi register indicates a reload register value (measurement results) <sup>(3)</sup>
Write to timer	The value written to the TBi register is written to neither the reload register nor the counter

Notes:

1. No interrupt request is generated when the pulse to be measured is applied on the initial valid edge after the timer counter starts.
2. While the TBiS bit is 1 (start counter), after the MR3 bit becomes 1 (overflow) and at least one count source cycle has elapsed, a write operation to the TBIMR register sets the MR3 bit to 0 (no overflow).
3. The TBi register indicates an undefined value until the pulse to be measured is applied on the second valid edge after the timer counter starts.



**Figure 15.27 Registers TB0MR to TB5MR in Pulse Period/Pulse-width Measure Mode**

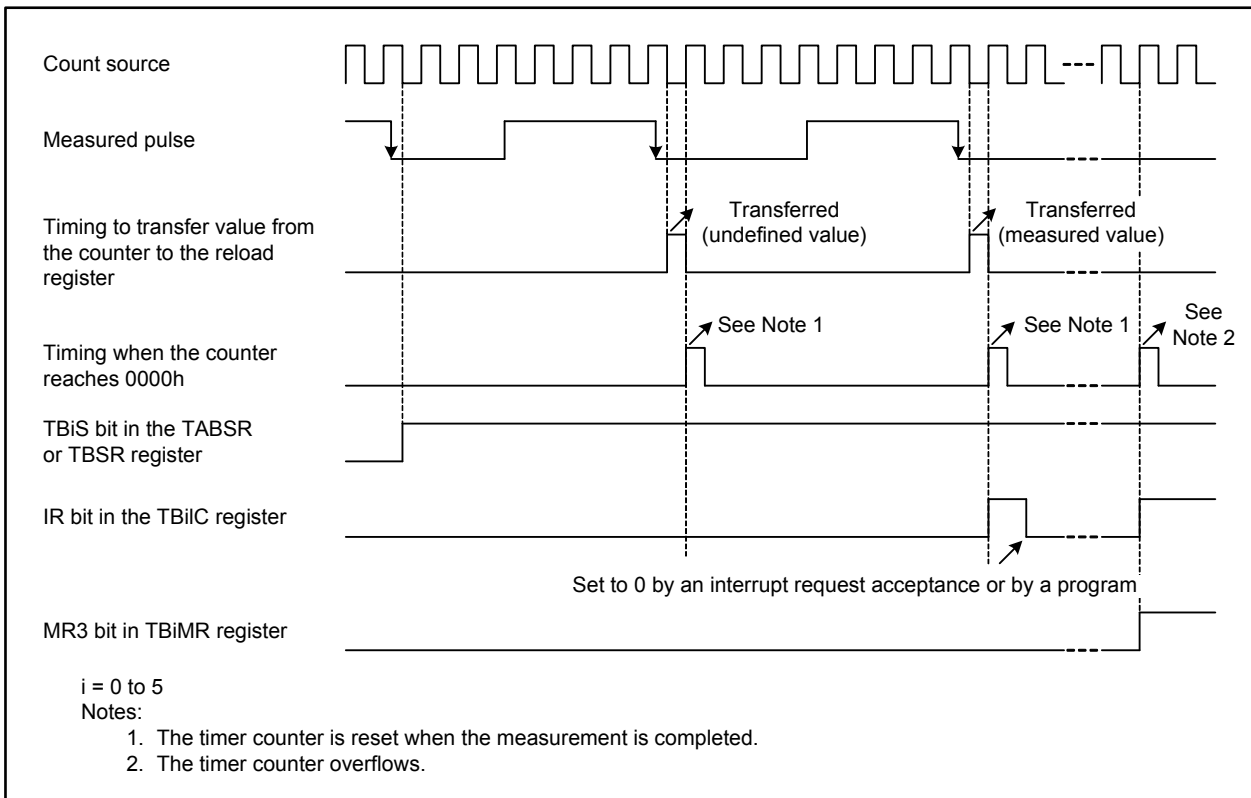


Figure 15.28 Operation Example in Pulse Period Measurement

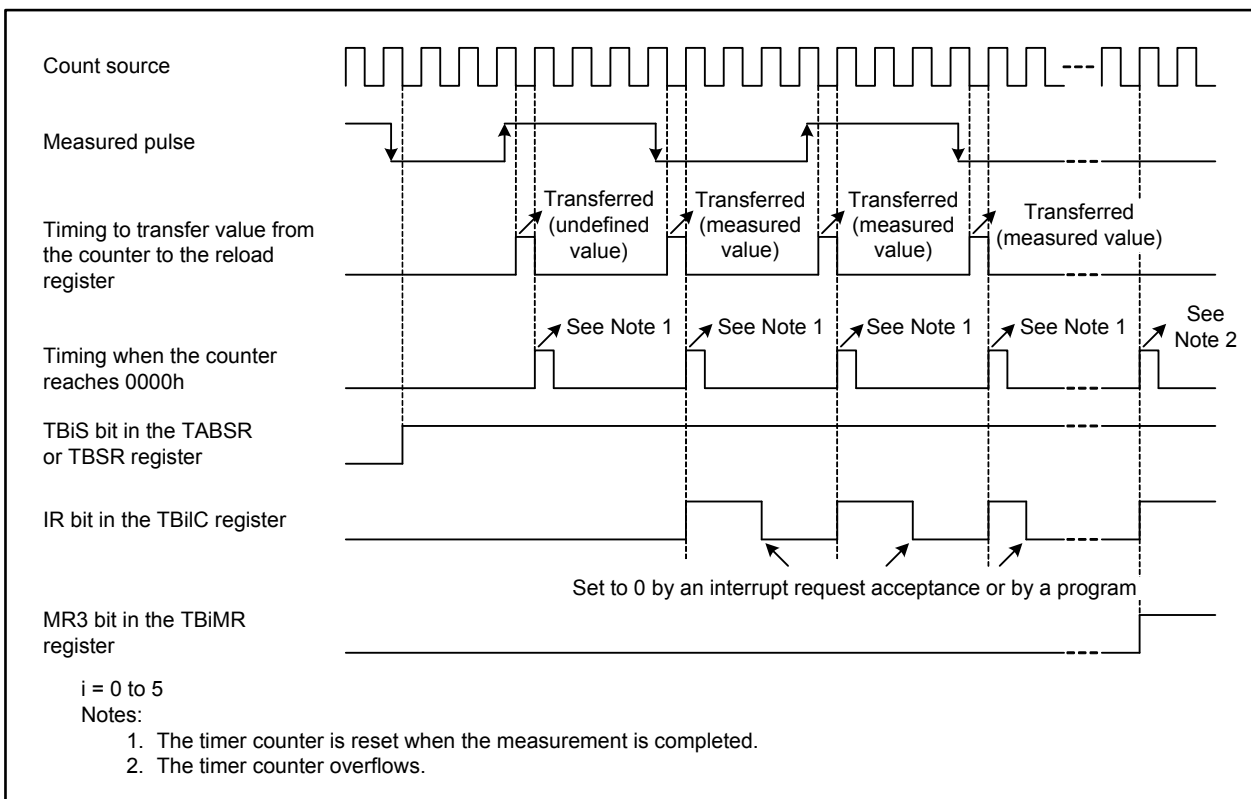


Figure 15.29 Operation Example in Pulse-width Measurement

## 15.3 Notes on Timers

### 15.3.1 Timer A and Timer B

All timers are stopped after a reset. To restart timers, configure parameters such as operating mode, count source, and counter value, then set the TAI<sub>S</sub> bit or TB<sub>J</sub>S bit in the TABSR or TBSR register to 1 (count starts) (i = 0 to 4; j = 0 to 5).

The following registers and bits should be set while the TAI<sub>S</sub> bit or TB<sub>J</sub>S bit is 0 (count stops):

- Registers TAI<sub>MR</sub> and TB<sub>J</sub>MR
- UDF register
- Bits TAZIE, TA0TGL, and TA0TGH in the ONSF register
- TRGSR register

### 15.3.2 Timer A

#### 15.3.2.1 Timer Mode

- While the timer counter is running, the TAI register indicates a counter value at any given time. However, FFFFh is read while reloading is in progress. A set value is read if the TAI register is set while the timer counter is stopped.

#### 15.3.2.2 Event Counter Mode

- While the timer counter is running, the TAI register indicates a counter value at any given time. However, FFFFh is read if the timer counter underflows or 0000h if overflows while reloading is in progress. A set value is read if the TAI register is set while the timer counter is stopped.

#### 15.3.2.3 One-shot Timer Mode

- If the TAI<sub>S</sub> bit in the TABSR register is set to 0 (count stops) while the timer counter is running, the following operations are performed:
  - The timer counter stops and the setting value of the TAI register is reloaded.
  - A low signal is output at the TAI<sub>OUT</sub> pin.
  - The IR bit in the TAI<sub>IC</sub> register becomes 1 (interrupts requested) after one CPU clock cycle.
- The one-shot timer is operated by an internal count source. When the trigger is an input to the TAI<sub>IN</sub> pin, the signal is output with a maximum one count source clock delay after a trigger input to the TAI<sub>IN</sub> pin.
- The IR bit becomes 1 by any of the settings below. To use the timer Ai interrupt, set the IR bit to 0 after one of the settings below is done:
  - Select one-shot timer mode after a reset.
  - Switch operating modes from timer mode to one-shot timer mode.
  - Switch operating modes from event counter mode to one-shot timer mode.
- If a retrigger occurs while counting, the timer counter decrements by one, reloads the setting value of the TAI register, and then continues counting. To generate a retrigger while counting, wait one or more count source cycles after the last trigger is generated.
- When an external trigger input is selected to start counting in timer A one-shot mode, do not provide an external retrigger for 300 ns before the timer counter reaches 0000h. Otherwise, it may stop counting.

#### 15.3.2.4 Pulse-width Modulation Mode

- The IR bit becomes 1 by any of the settings below. To use the timer Ai interrupt, set the IR bit to 0 after one of the settings below is done (i = 0 to 4):
  - Select pulse-width modulation mode after a reset.
  - Switch operating modes from timer mode to pulse-width modulation mode.
  - Switch operating modes from event counter mode to pulse-width modulation mode.
  
- If the TAI<sub>S</sub> bit in the TABSR register is set to 0 (count stops) while PWM pulse is output, the following operations are performed:
  - The timer counter stops.
  - The output level at the TAI<sub>OUT</sub> pin changes from high to low. The IR bit becomes 1.
  - When a low signal is output at the TAI<sub>OUT</sub> pin, it does not change. The IR bit does not change, either.

### 15.3.3 Timer B

#### 15.3.3.1 Timer Mode and Event Counter Mode

- While the timer counter is running, the T Bj register indicates a counter value at any given time (j = 0 to 5). However, FFFFh is read while reloading is in progress. When a value is set to the T Bj register while the timer counter is stopped, if the T Bj register is read before the count starts, the set value is read.

#### 15.3.3.2 Pulse Period/Pulse-width Measure Mode

- While the T B j S bit in the TABSR or TBSR register is 1 (start counter), after the MR3 bit becomes 1 (overflow) and at least one count source cycle has elapsed, a write operation to the T B j MR register sets the MR3 bit to 0 (no overflow).
- Use the IR bit in the T B j IC register to detect overflow. The MR3 bit is used only to determine an interrupt request source within the interrupt handler.
- The counter value is undefined when the timer counter starts. Therefore, the timer counter may overflow before a measured pulse is applied on the initial valid edge and cause a timer B j interrupt request to be generated.
- When the measured pulse is applied on the initial valid edge after the timer counter starts, an undefined value is transferred to the reload register. At this time, a timer B j interrupt request is not generated.
- The IR bit may become 1 (interrupt requested) by changing bits MR1 and MR0 in the T B j MR register after the timer counter starts. However, if the same value is rewritten to bits MR1 and MR0, the IR bit does not change.
- Pulse width is continuously measured in pulse-width measure mode. Whether the measurement result is high-level width or not is determined by a program.
- When an overflow occurs at the same time a pulse is applied on the valid edge, this pulse is not recognized since an interrupt request is generated only once. Do not let an overflow occur in pulse period measure mode.
- In pulse-width measure mode, determine whether an interrupt source is a pulse applied on the valid edge or an overflow by reading the port level in the timer B j interrupt handler.



## 16. Three-phase Motor Control Timers

A three-phase motor driving waveform can be output using timers A1, A2, A4, and B2. The three-phase motor control timers are enabled by setting the INV02 bit in the INVC0 register to 1. Timer B2 is used for carrier wave control, and timers A1, A2, and A4 for three-phase PWM output (U,  $\bar{U}$ , V,  $\bar{V}$ , W, and  $\bar{W}$ ) control. Table 16.1 lists the specifications of the three-phase motor control timers and Figure 16.1 shows its block diagram. Figures 16.2 to 16.6 show registers associated with this function.

**Table 16.1 Specifications for Three-phase Motor Control Timers**

Item	Specification
Three-phase PWM waveform output pins	Six pins: U, $\bar{U}$ , V, $\bar{V}$ , W, and $\bar{W}$
Forced cutoff (1)	A low input to the $\overline{\text{NMI}}$ pin
Timers	Timers A4, A1, and A2 are used in one-shot timer mode: Timer A4 is used for U- and $\bar{U}$ -phase waveform control Timer A1 is used for V- and $\bar{V}$ -phase waveform control Timer A2 is used for W- and $\bar{W}$ -phase waveform control Timer B2 is used in timer mode Carrier wave cycle control Dead time timer (three 8-bit timers share a reload register): Dead time control
Output waveforms	Triangular wave modulation and sawtooth wave modulation • Output of a high or a low waveform for one cycle • Separately settable levels of high side and low side
Carrier wave cycles	Triangular wave modulation: count source $\times (m + 1) \times 2$ Sawtooth wave modulation: count source $\times (m + 1)$ m: TB2 register setting value from 0000h to FFFFh Count source: f1, f8, f2n, or fC32
Three-phase PWM output width	Triangular wave modulation: count source $\times n \times 2$ Sawtooth wave modulation: count source $\times n$ n: Setting value of registers TA4, TA1, and TA2 (registers TA4, TA41, TA1, TA11, TA2, and TA21 when the INV11 bit in the INVC1 register is 1) from 0001h to FFFFh Count source: f1, f8, f2n, or fC32
Dead time (width)	Count source $\times p$ or no dead time p: DTT register setting value from 01h to FFh Count source: f1 or f1 divided by 2
Active level	Selectable either active high or active low
Simultaneous conduction prevention	Function to detect simultaneous turn-on signal outputs, function to disable signal output when simultaneous turn-on signal outputs are detected
Interrupt frequency	Selectable from one through 15 time-carrier wave cycle-to-cycle basis for the timer B2 interrupt

Note:

1. Forced cutoff by a signal input to the  $\overline{\text{NMI}}$  pin can be performed when the PM24 bit in the PM2 register is 1 (NMI enabled), the SDE bit in the IOBC register is 1 (shutdown enabled), the INV02 bit in the INVC0 register is 1 (three-phase motor control timers used), and the INV03 bit is 1 (three-phase motor control timer output enabled).

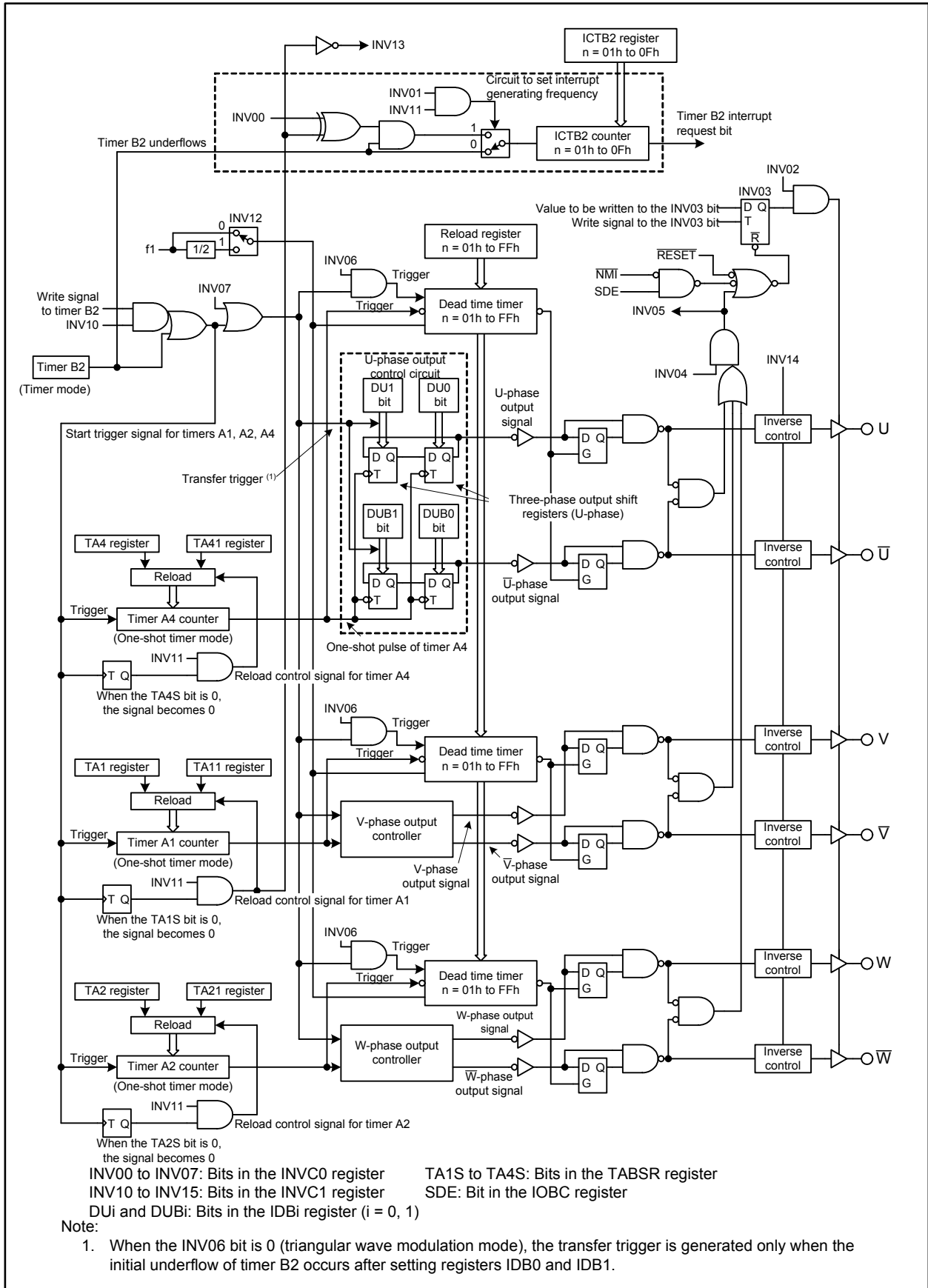


Figure 16.1 Block Diagram for Three-phase Motor Control Timers

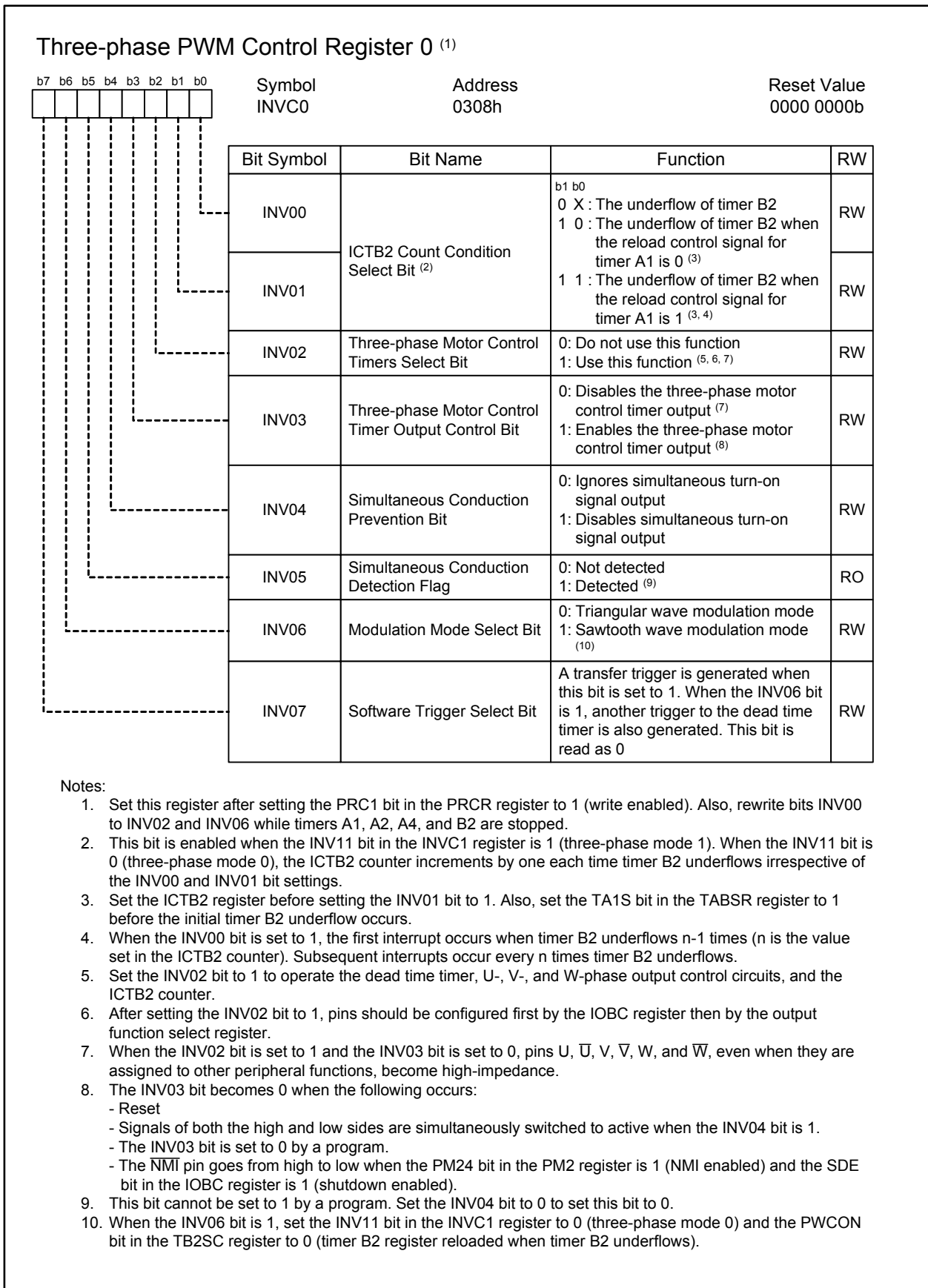


Figure 16.2 INVC0 Register

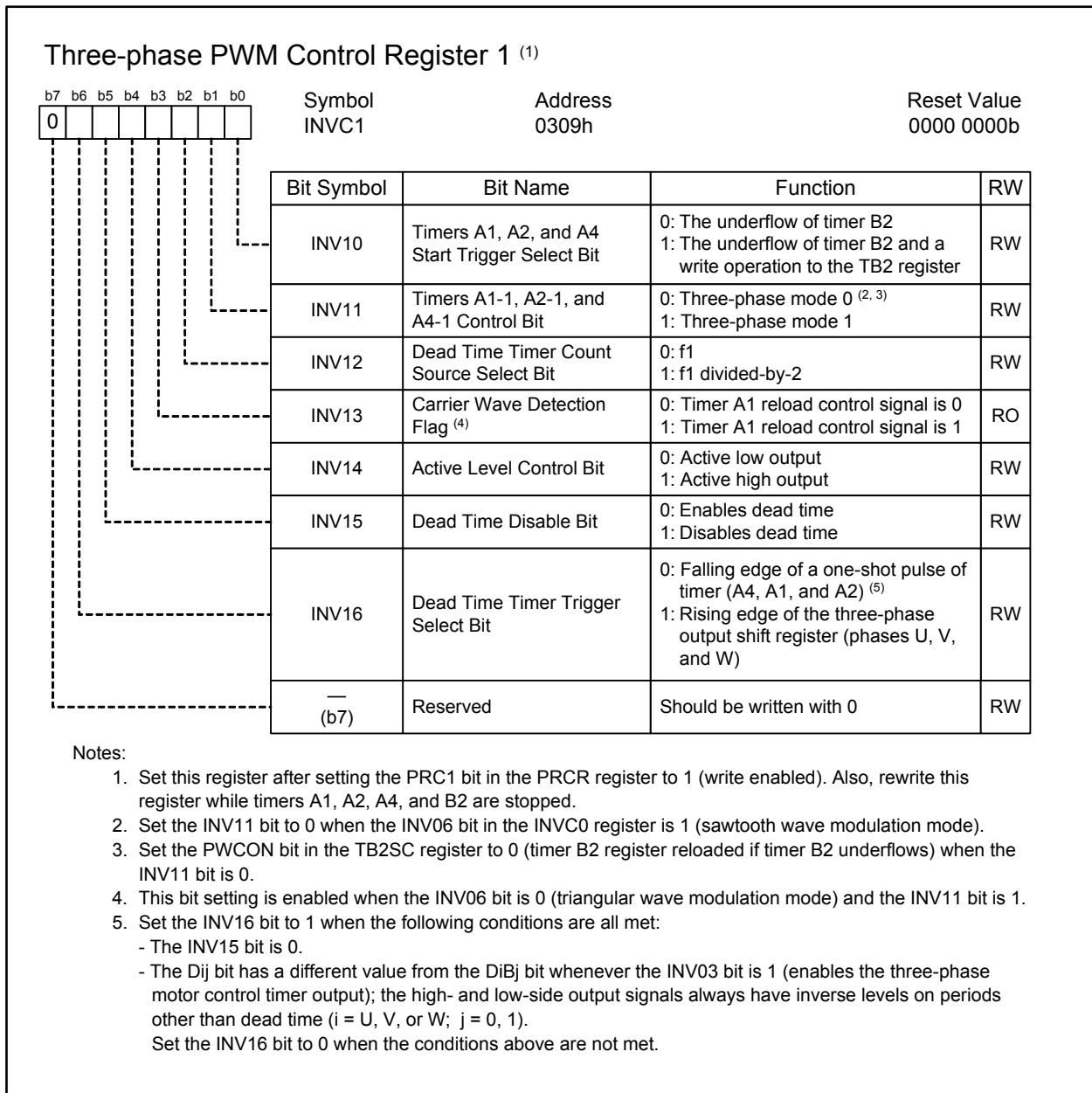


Figure 16.3 INVC1 Register

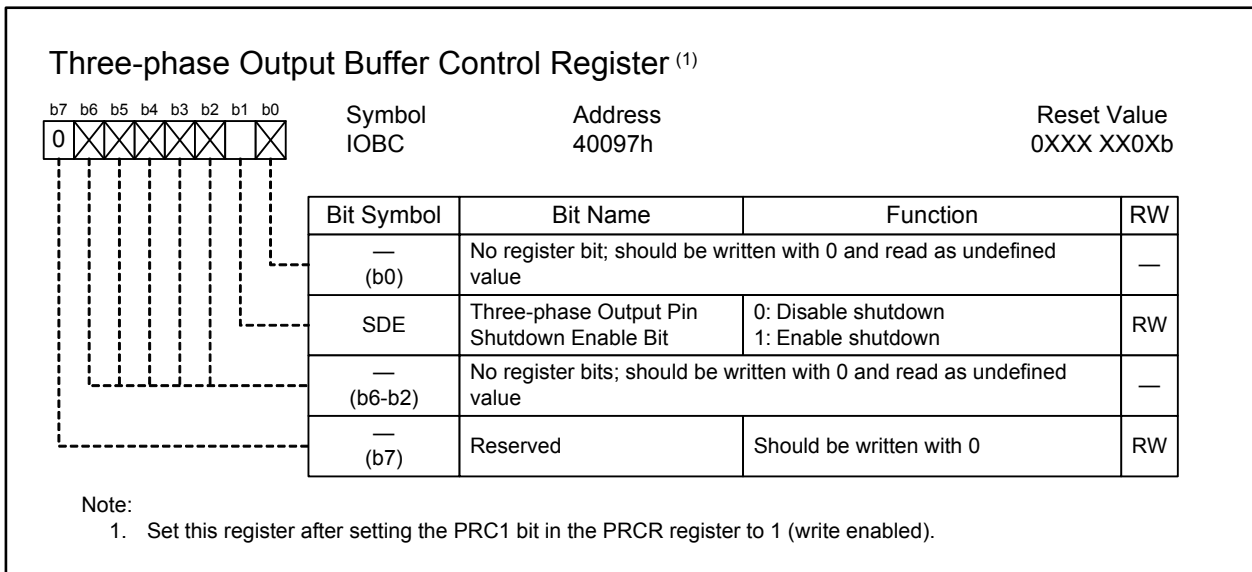


Figure 16.4 IOBC Register

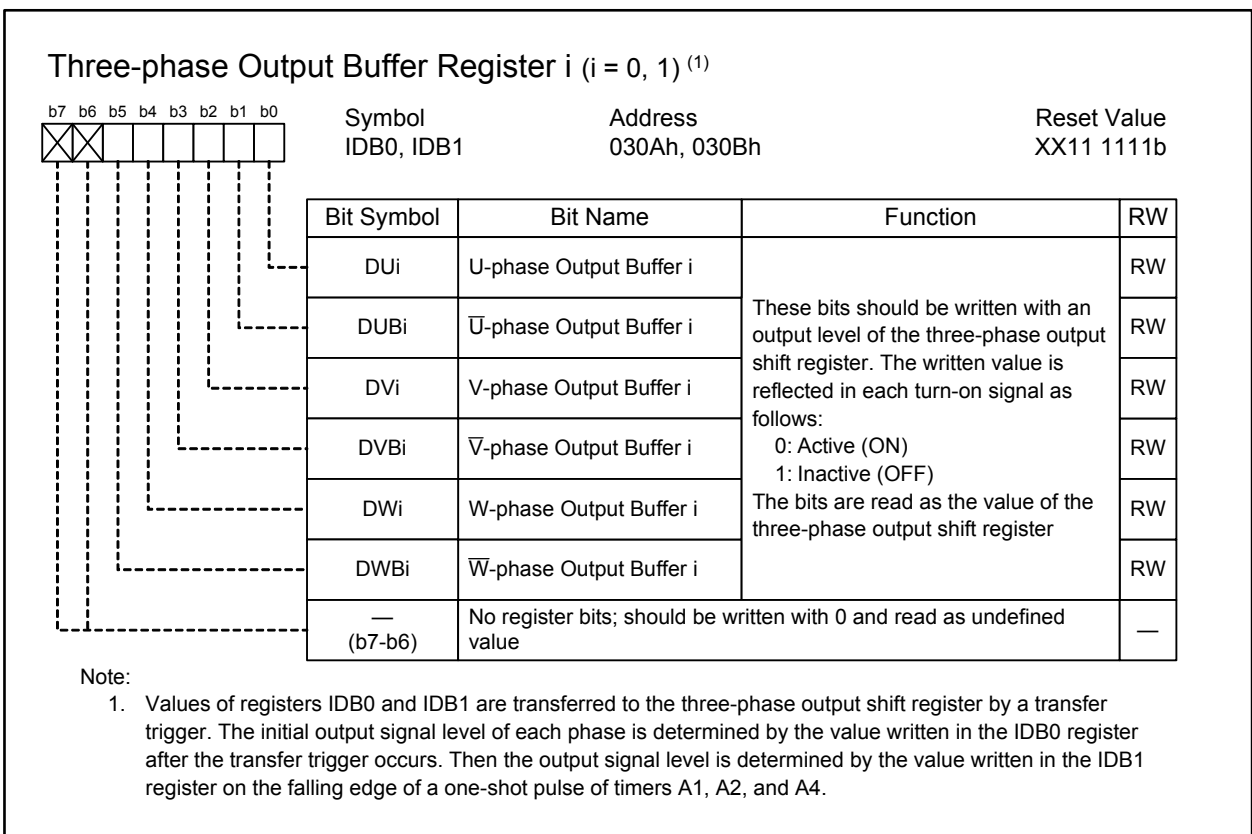
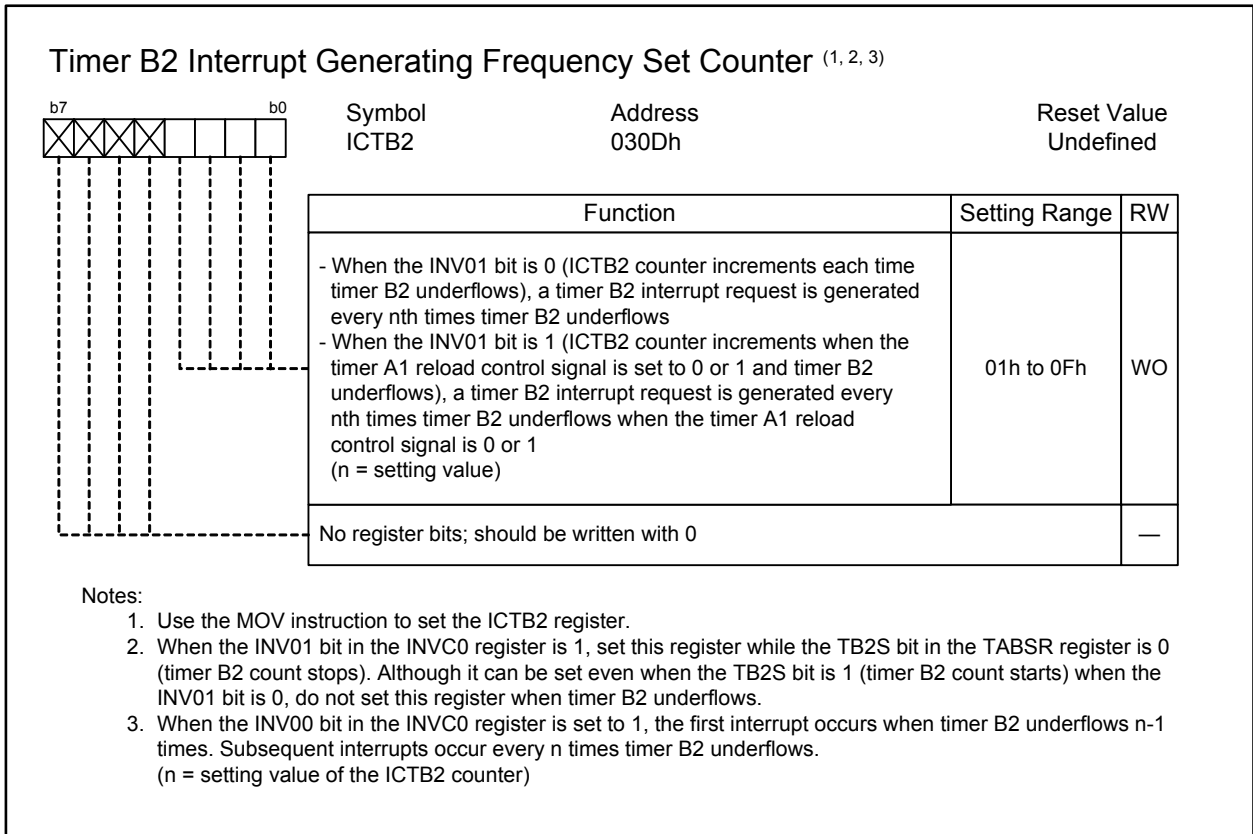


Figure 16.5 Registers IDB0 and IDB1



**Figure 16.6 ICTB2 Register**

## 16.1 Modulation Modes of Three-phase Motor Control Timers

The three-phase motor control timers support two modulation modes: triangular wave modulation mode and sawtooth wave modulation mode. The triangular wave modulation mode has two modes: three-phase mode 0 and three-phase mode 1. Table 16.2 lists bit settings and characteristics of each mode.

**Table 16.2 Modulation Modes**

Item	Triangular Wave Modulation Mode		Sawtooth Wave Modulation Mode
	Three-phase mode 0	Three-phase mode 1	(Three-phase mode 0)
Bit settings	INV06 is 0, INV11 is 0, PWCON is 0	INV06 is 0, INV11 is 1	INV06 is 1, INV11 is 0, PWCON is 0
Waveform	Triangular wave		Sawtooth wave
Registers TA11, TA21, and TA41	Not used	Used	Not used
Timing to transfer data from registers IDB0 and IDB1 to the three-phase output shift register	Only once when a transfer trigger <sup>(1)</sup> occurs after setting registers IDB0 and IDB1		Whenever a transfer trigger <sup>(1)</sup> occurs
Timing to trigger the dead time timer when the INV16 bit is 0	On the falling edge of a one-shot pulse of timers A1, A2, and A4		When a transfer trigger occurs, or on the falling edge of a one-shot pulse of timers A1, A2, and A4
Bits INV00 and INV01 in the INVC0 register	Disabled. The ICTB2 counter increments each time timer B2 underflows, irrespective of the INV00 and INV01 bit settings	Enabled	Disabled. The ICTB2 counter increments each time timer B2 underflows, irrespective of the INV00 and INV01 bit settings
INV13 bit	Disabled	Enabled	Disabled

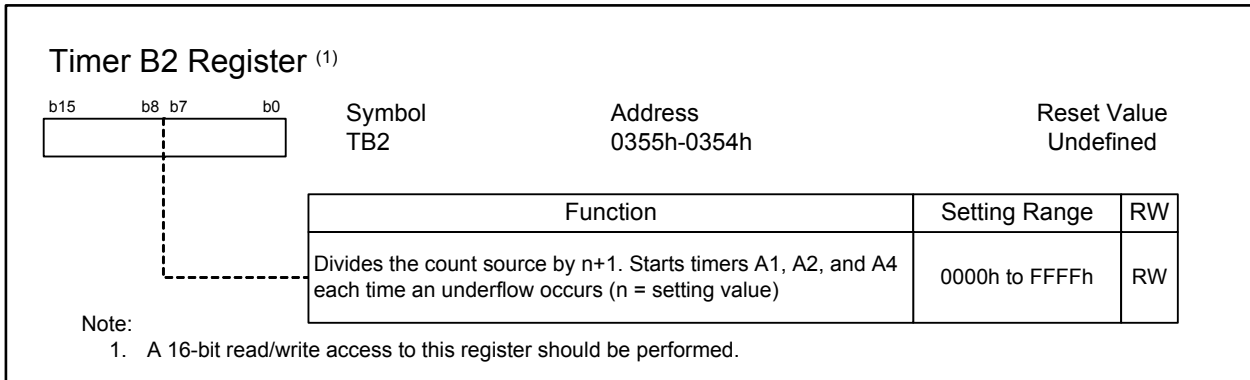
Note:

1. The transfer trigger is a timer B2 underflow, a write operation to the INV07 bit, or a write operation to the TB2 register when the INV10 bit is 1.

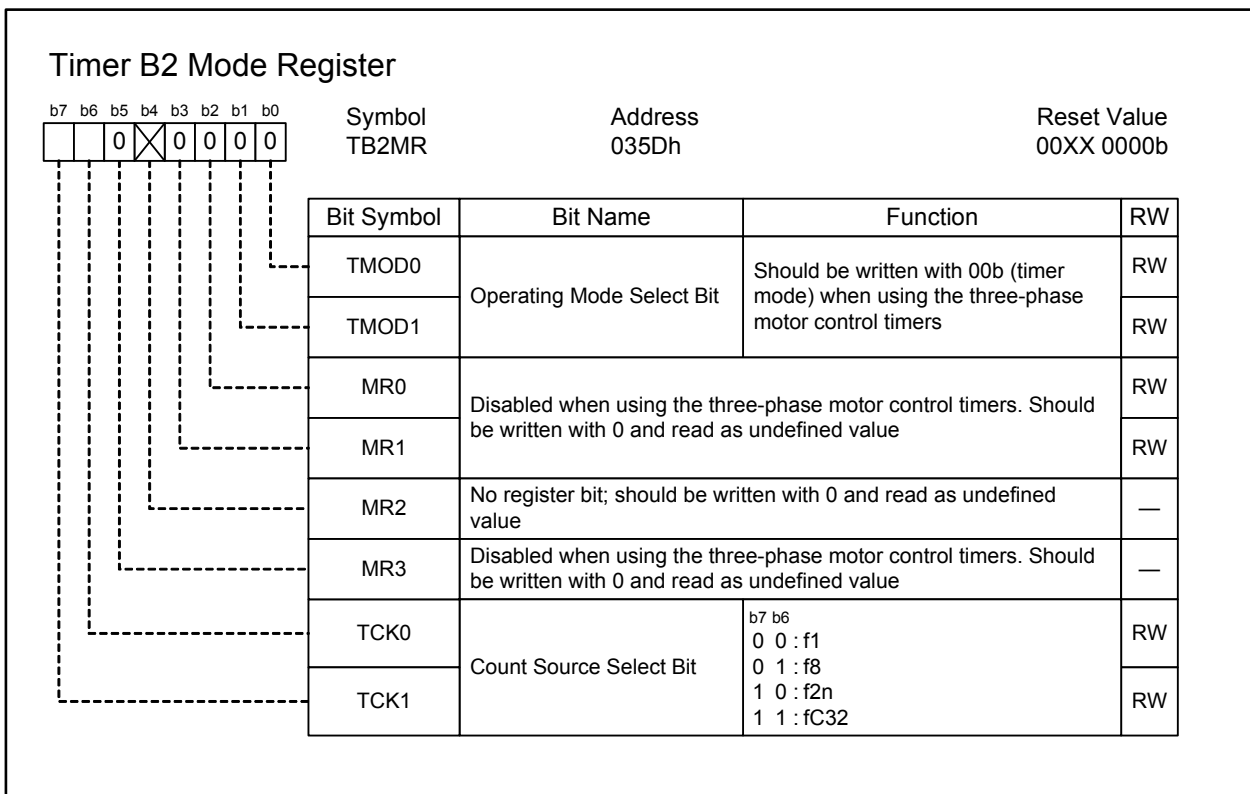
### 16.2 Timer B2

Timer B2, which operates in timer mode, is used for carrier wave control in the three-phase motor control timers.

Figures 16.7 and 16.8 show registers TB2 and TB2MR in this function, respectively. Figure 16.9 shows the TB2SC register which switches timing to change the carrier wave frequency in three-phase mode 1.

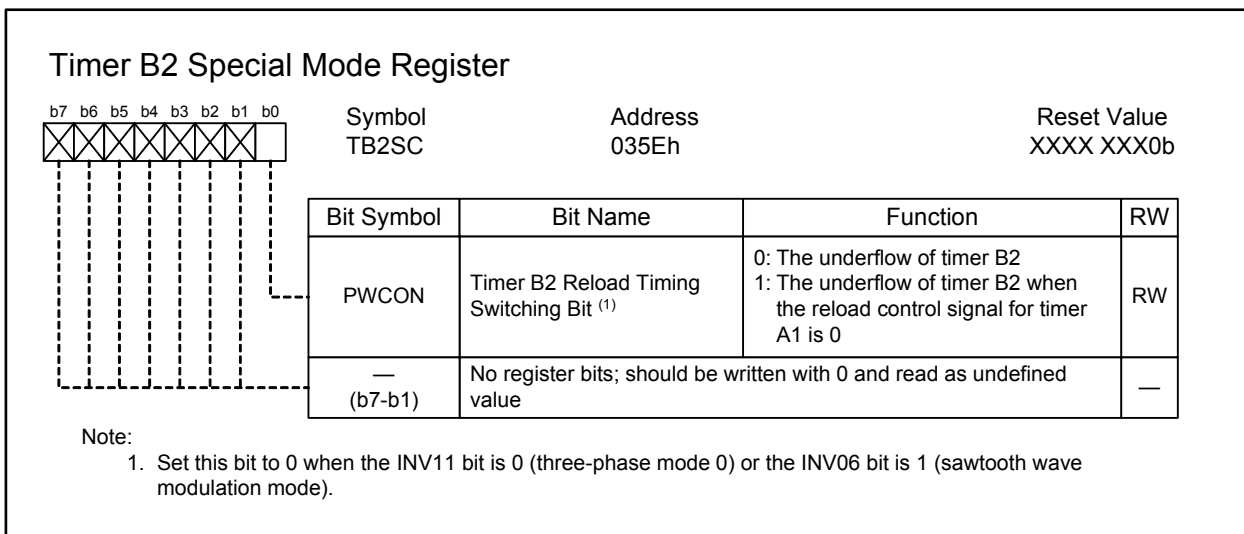


**Figure 16.7 TB2 Register When Using Three-phase Motor Control Timers**



**Figure 16.8 TB2MR Register When Using Three-phase Motor Control Timers**





**Figure 16.9 TB2SC Register**

### 16.3 Timers A4, A1, and A2

Timers A4, A1, and A2 are used for three-phase PWM output (U,  $\bar{U}$ , V,  $\bar{V}$ , W, and  $\bar{W}$ ) control when using the three-phase motor control timers.

These timers should be operated in one-shot timer mode. Every time timer B2 underflows, a trigger is input to timers A4, A1, and A2 to generate a one-shot pulse. If the values of registers TA4, TA1, and TA2 are rewritten every time a timer B2 interrupt occurs, the duty cycle of the PWM waveform can be varied. In three-phase mode 1, the value of registers TAI and TAI-1 is alternately reloaded to the counter at each timer B2 interrupt, which halves the timer B2 interrupt frequency ( $i = 4, 1, 2$ ).

Figure 16.10 shows registers TA1, TA2, TA4, TA11, TA21, and TA41 in the three-phase motor control timers. Figure 16.11 shows registers TA1M, TA2M, TA4M, TA11M, TA21M, and TA41M in this function. Figure 16.12 shows registers TA1MR, TA2MR, and TA4MR in this function. Figures 16.13 and 16.14 show registers TRGSR and TABSR, respectively, in this function.

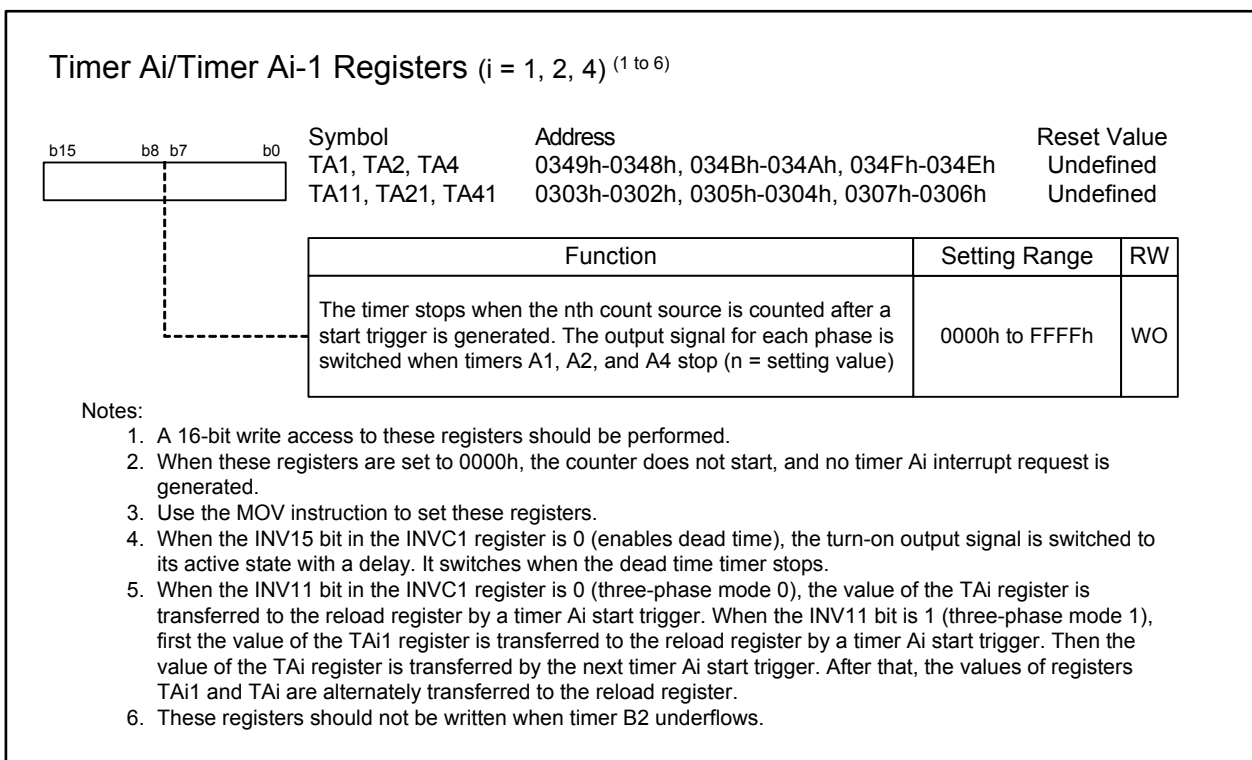
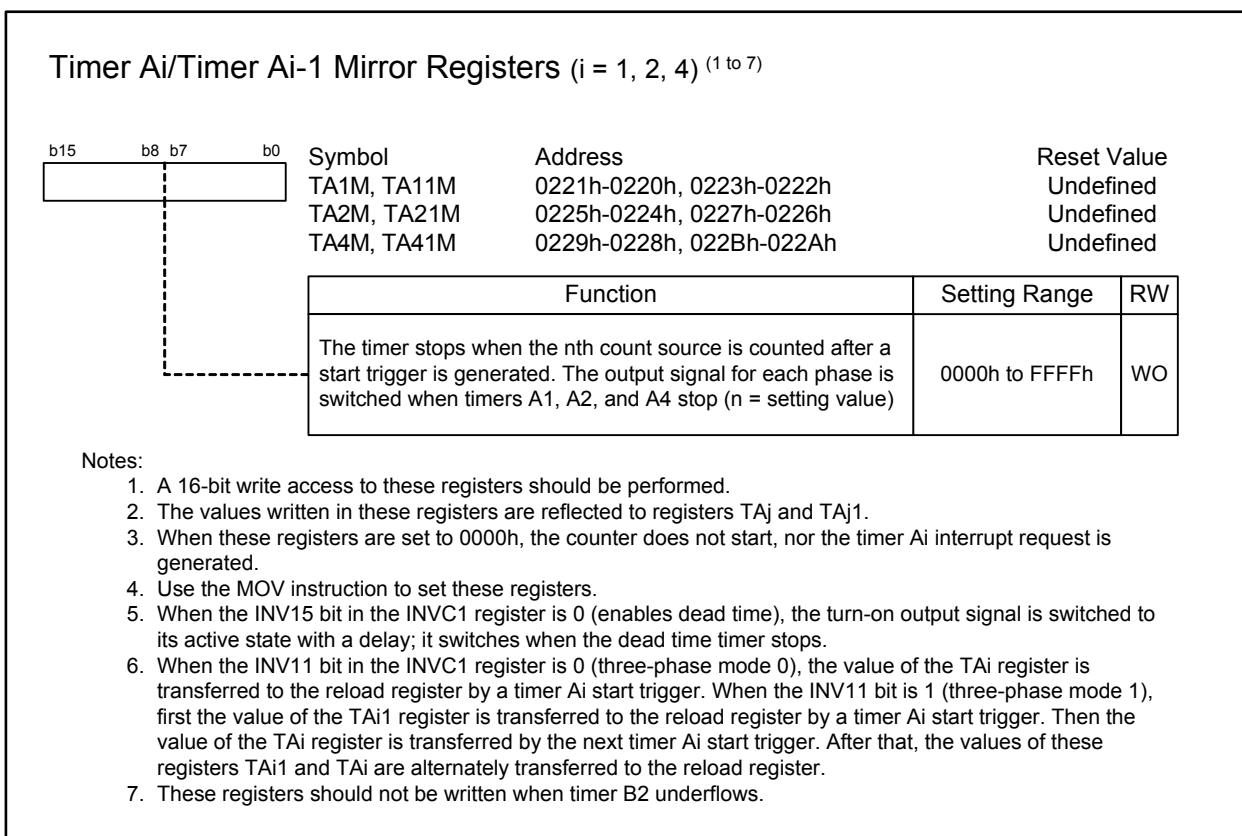
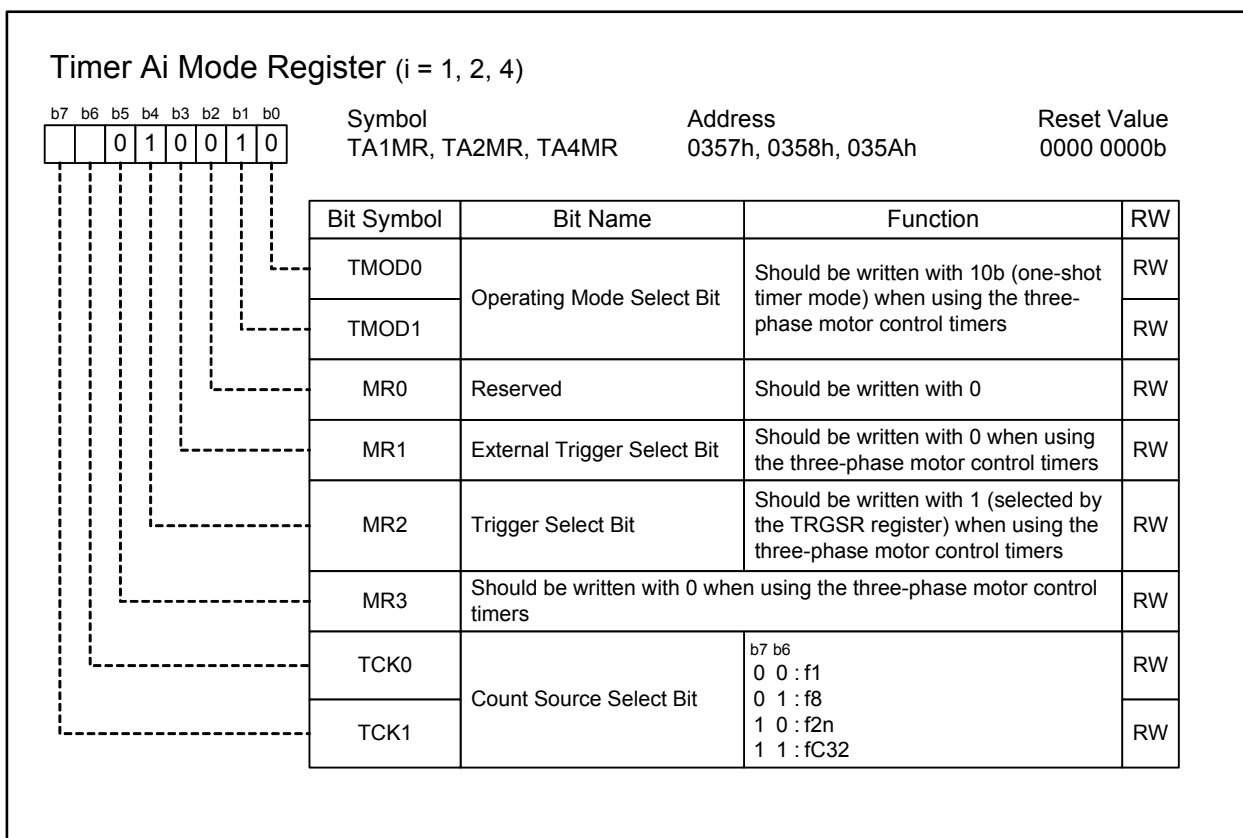


Figure 16.10 Registers TA1, TA2, TA4, TA11, TA21, and TA41



**Figure 16.11 Registers TA1M, TA2M, TA4M, TA11M, TA21M, and TA41M**



**Figure 16.12 Registers TA1MR, TA2MR, and TA4MR When Using Three-phase Motor Control Timers**

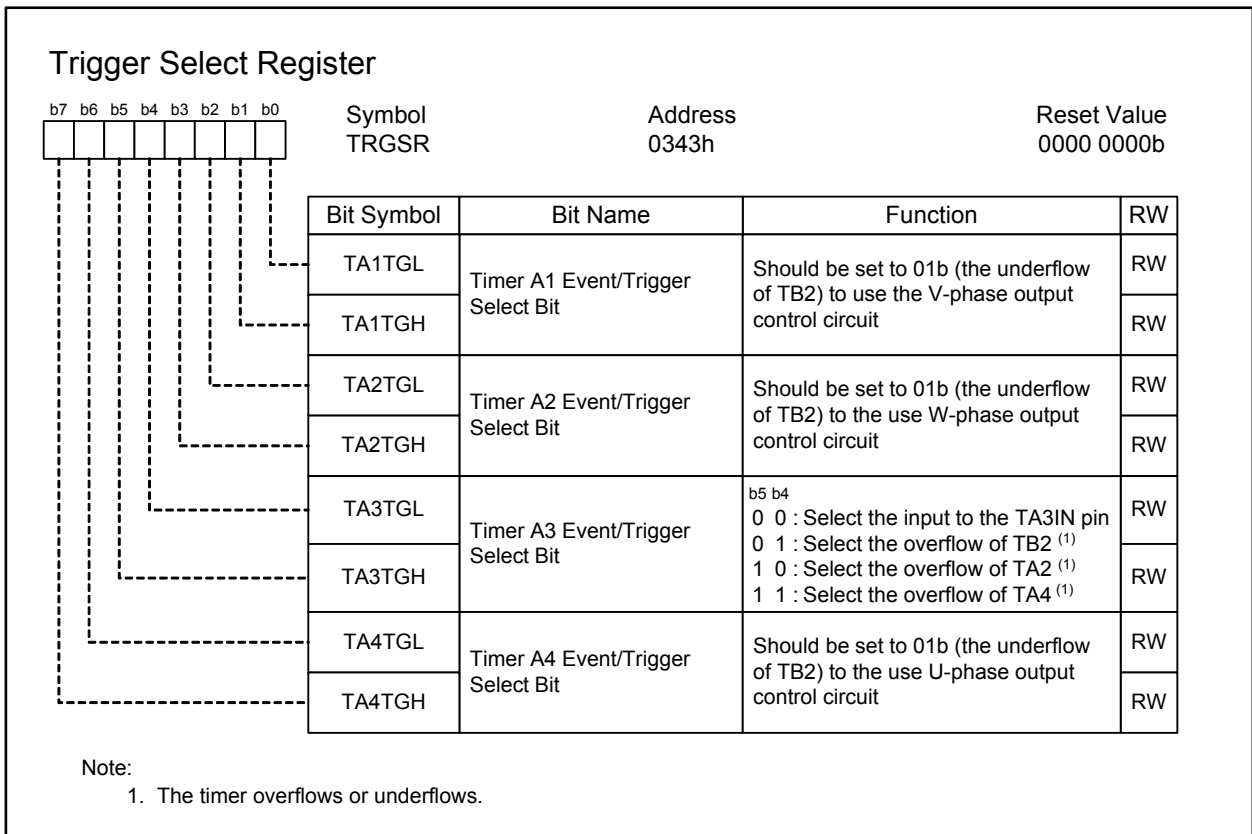


Figure 16.13 TRGSR Register in Three-phase Motor Control Timers

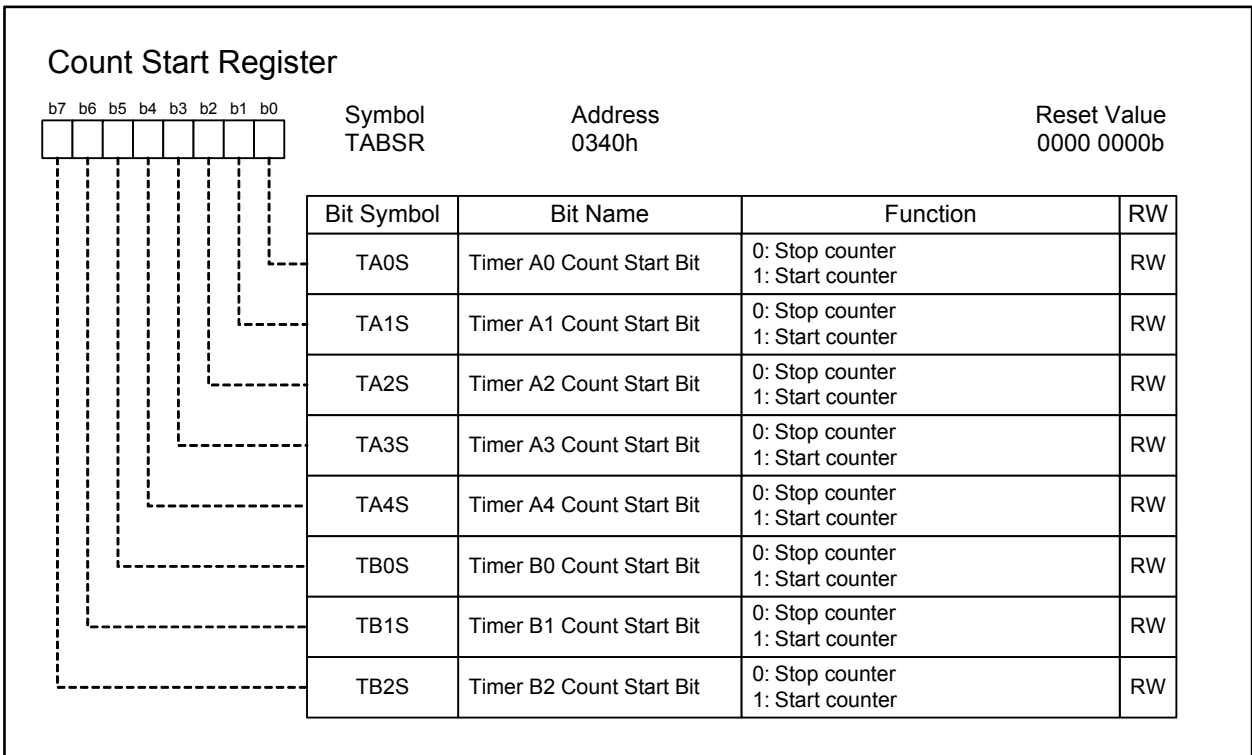


Figure 16.14 TABSR Register

## 16.4 Simultaneous Conduction Prevention and Dead Time Timer

The three-phase motor control timers offer two ways to avoid shoot-through, which occurs when high-side and low-side transistors are simultaneously turned on.

One is “simultaneous turn-on signal output disable function”. This function prevents high-side and low-side transistors from being inadvertently switched to active due to events like program errors. The other is by the use of dead time timers. A dead time timer delays the turn-on of one transistor in order to ensure that an adequate time (the dead time) passes after the other is turned off.

To disable simultaneous turn-on output signals, the INV04 bit in the INVC0 register should be set to 1. If outputs for any pair of phases (U and  $\bar{U}$ , V and  $\bar{V}$ , or W and  $\bar{W}$ ) are simultaneously switched to an active state, every three-phase motor control output pin becomes high-impedance. Figure 16.15 shows an example of output waveform when simultaneous turn-on signal output is disabled.

To enable the dead time timer, the INV15 bit in the INVC1 register should be set to 0. The DTT register determines the dead time. Figure 16.16 shows the DTT register and Figure 16.17 shows an example of output waveform on using dead time timer.

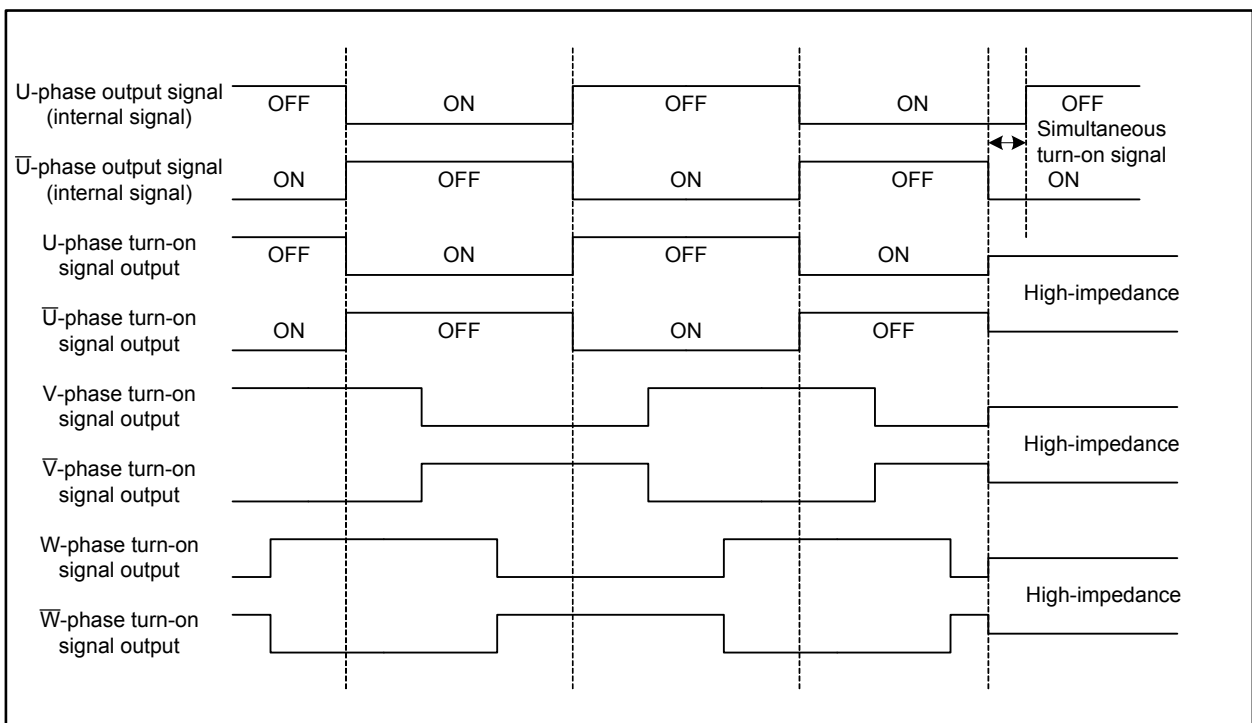
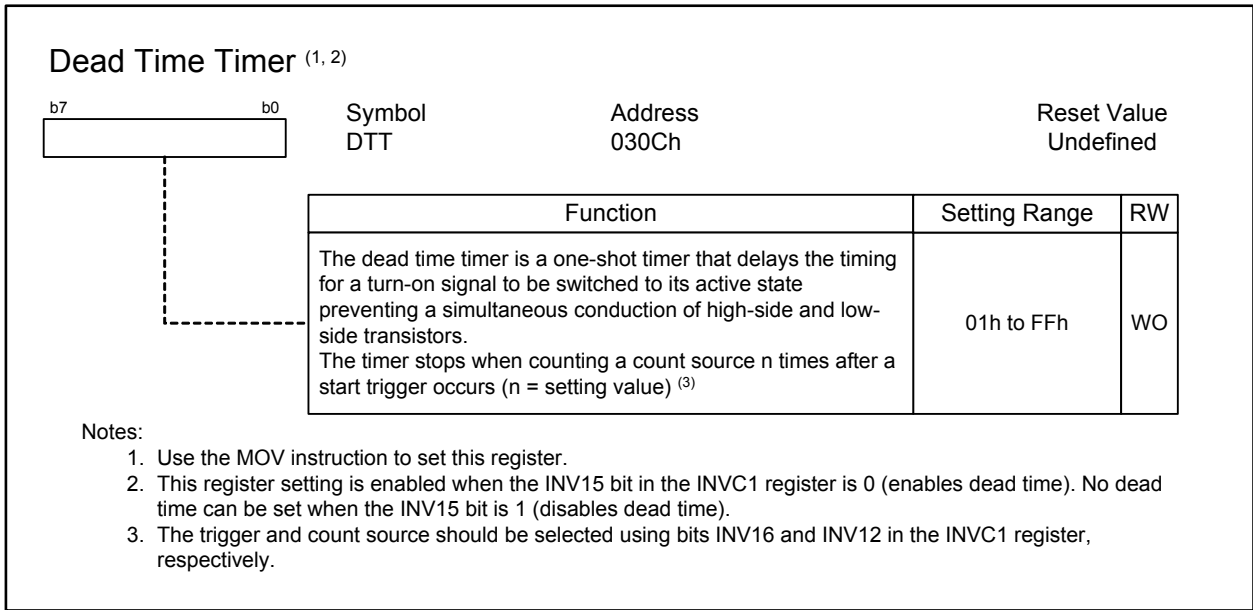
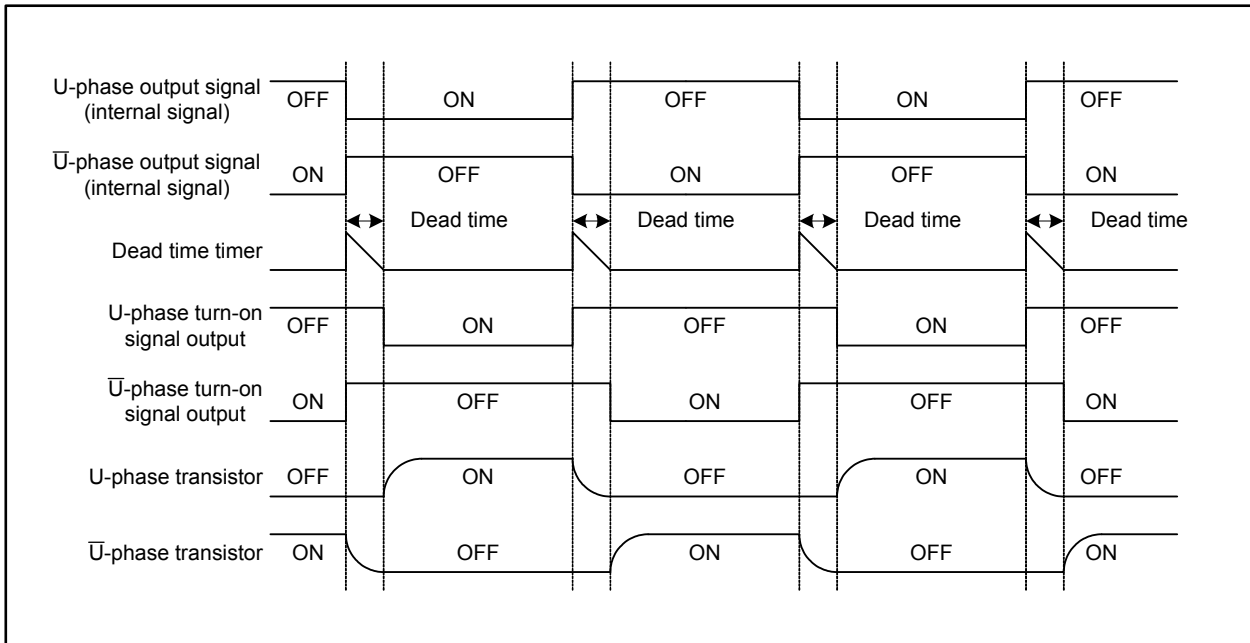


Figure 16.15 Output Waveform When Simultaneous Turn-on Signal Output is Disabled



**Figure 16.16 DTT Register**



**Figure 16.17 Output Waveform When Using Dead Time Timer**

### 16.5 Three-phase Motor Control Timer Operation

Figures 16.18 and 16.19 show an operation example of triangular wave modulation and sawtooth wave modulation, respectively.

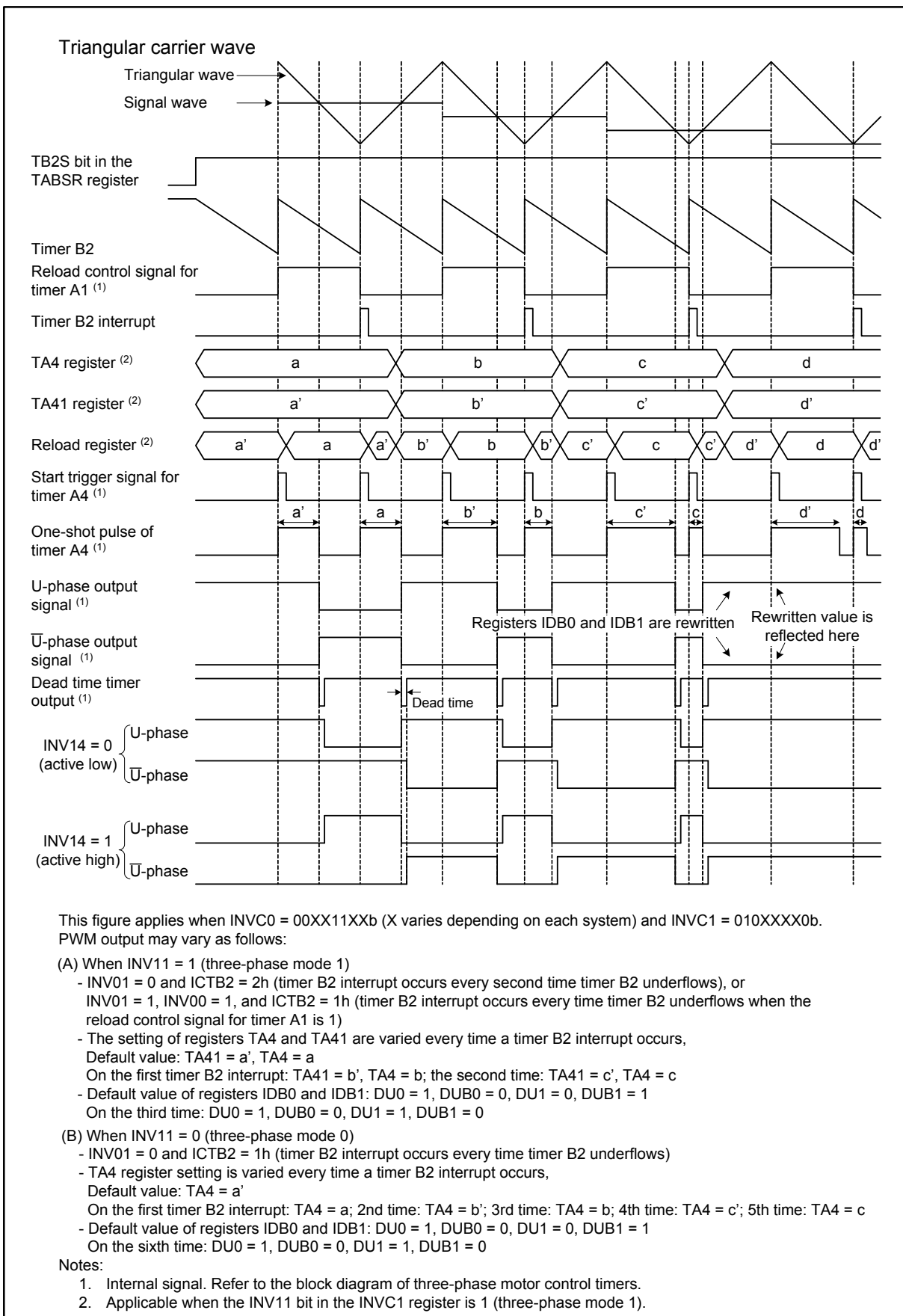
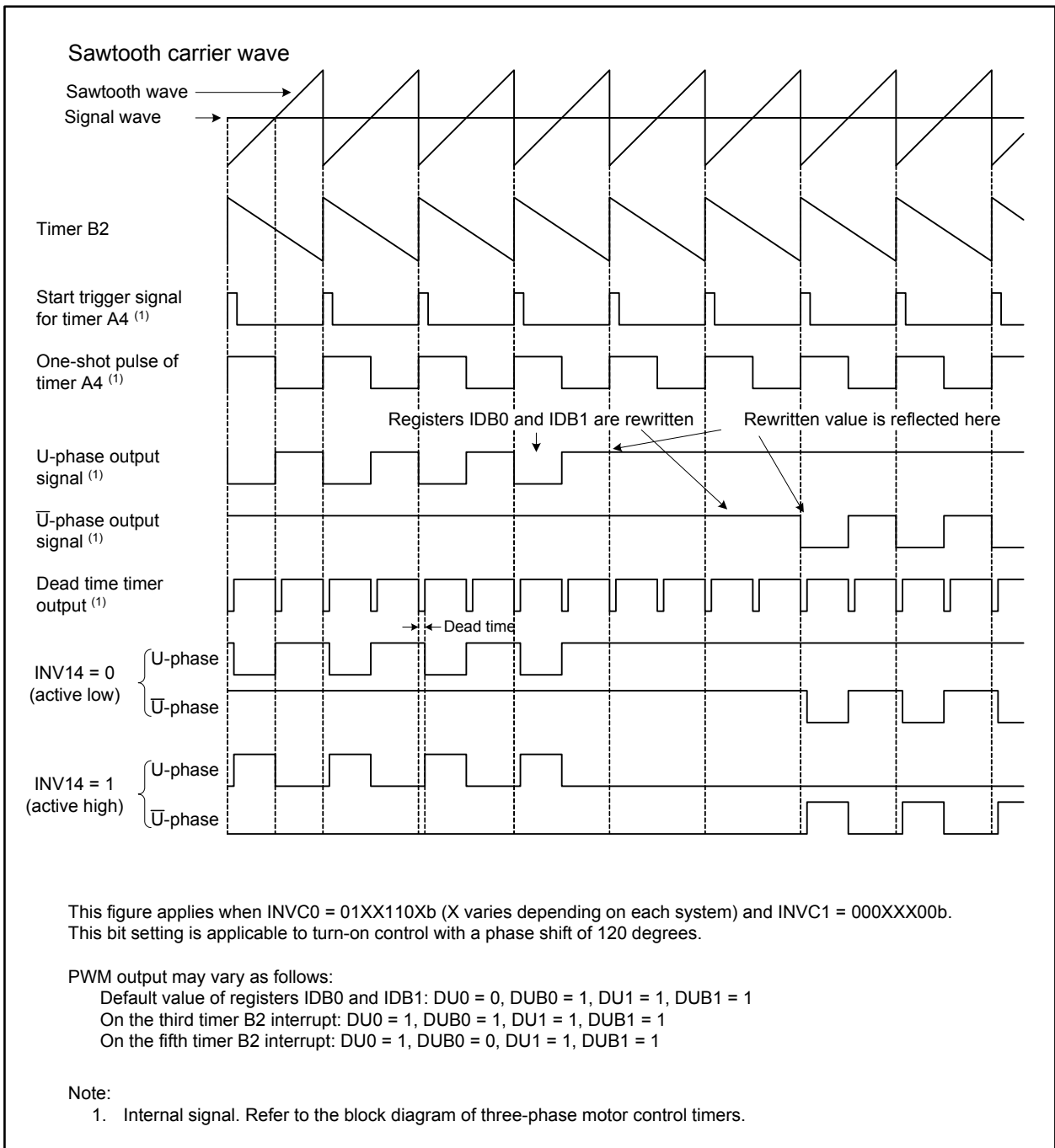


Figure 16.18 Triangular Wave Modulation Operation





**Figure 16.19 Sawtooth Wave Modulation Operation**

## 16.6 Notes on Three-phase Motor Control Timers

### 16.6.1 Shutdown

- When a low signal is applied to the  $\overline{\text{NMI}}$  pin with the following bit settings, pins TA1OUT, TA2OUT, and TA4OUT become high-impedance: the PM24 bit in the PM2 register is 1 (NMI enabled), the SDE bit in the IOBC register is 1 (shutdown enabled), the INV02 bit in the INVC0 register is 1 (three-phase motor control timers used), and the INV03 bit is 1 (three-phase motor control timer output enabled).

### 16.6.2 Register Setting

- Do not write to the TAI1 register before and after timer B2 underflows ( $i = 1, 2, 4$ ). Before writing to the TAI1 register, read the TB2 register to verify that sufficient time remains until timer B2 underflows. Then, immediately write to the TAI1 register so no interrupt handling is performed during this write procedure. If the TB2 register indicates little time remains until the underflow, write to the TAI1 register after timer B2 underflows.

## 17. Serial Interface

The serial interface consists of five channels: UART0 to UART4.

Each channel has an exclusive timer to generate the transmit/receive clock and operates independently.

Figures 17.1 and 17.2 show block diagrams of UART0 to UART2 and UART3 and UART4, respectively.

UART<sub>i</sub> supports the following modes:

- Synchronous serial interface mode (for UART0 to UART4)
- Asynchronous serial interface mode (UART mode) (for UART0 to UART4)
- Special mode 1 (I<sup>2</sup>C mode) (for UART0 to UART2)
- Special mode 2 (for UART0 to UART2)

Figures 17.3 to 17.17 show registers associated with UART<sub>i</sub> (i = 0 to 4).

Refer to the tables listing each mode for registers and pin settings.

**Table 17.1 Comparison UART0 to UART4 Functions**

Mode/Function	UART0 to UART2	UART3, UART4
Synchronous serial interface mode	Available	Available
Serial data logic inversion	Available	Not available
UART mode	Available	Available
CTS/RTS function selection	Available	Available
TXD and RXD I/O polarity selection	Available	Not available
Special mode 1 (I <sup>2</sup> C mode)	Available	Not available
Special mode 2	Available	Not available

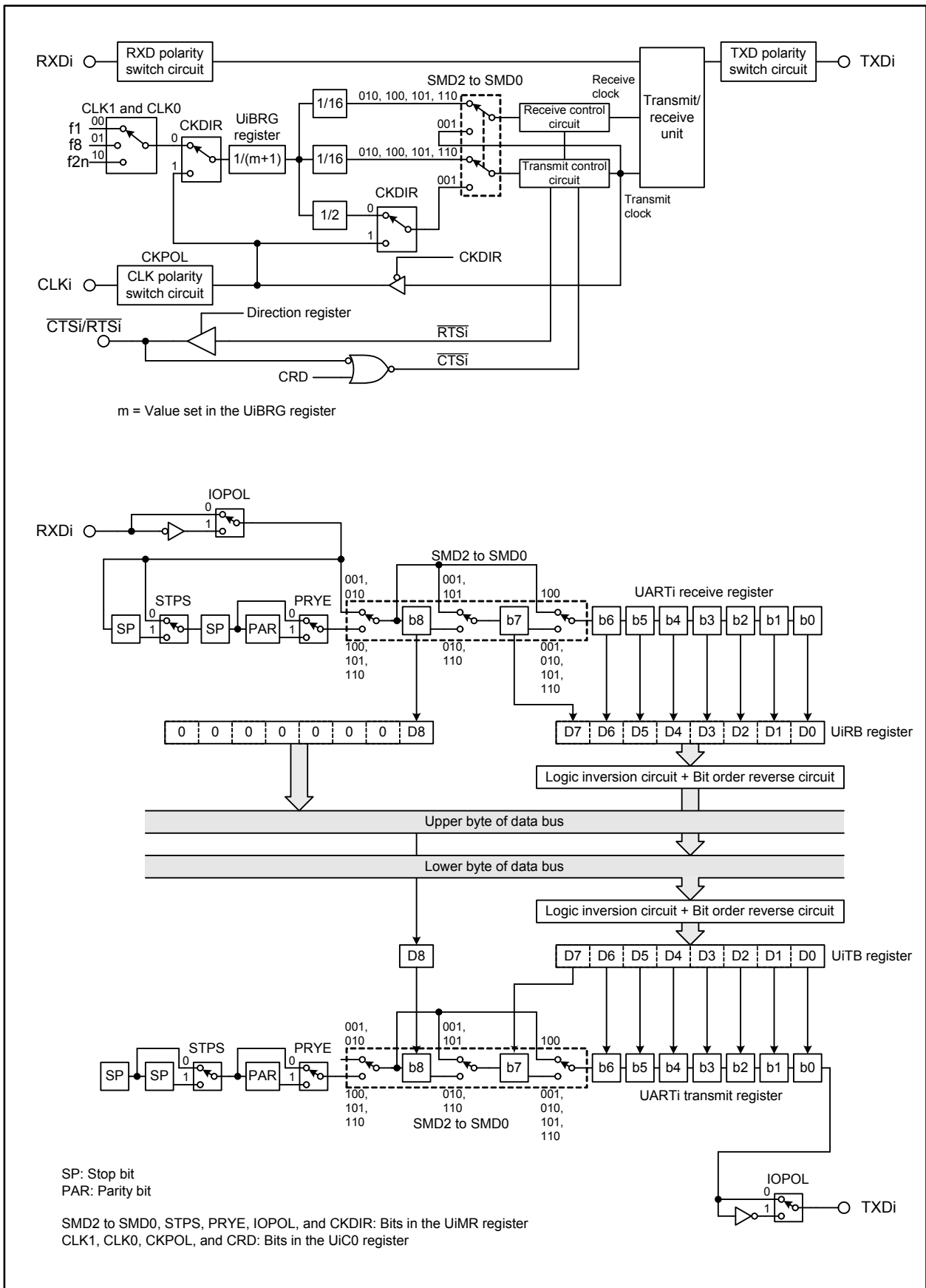


Figure 17.1 UARTi Block Diagram (i = 0 to 2)

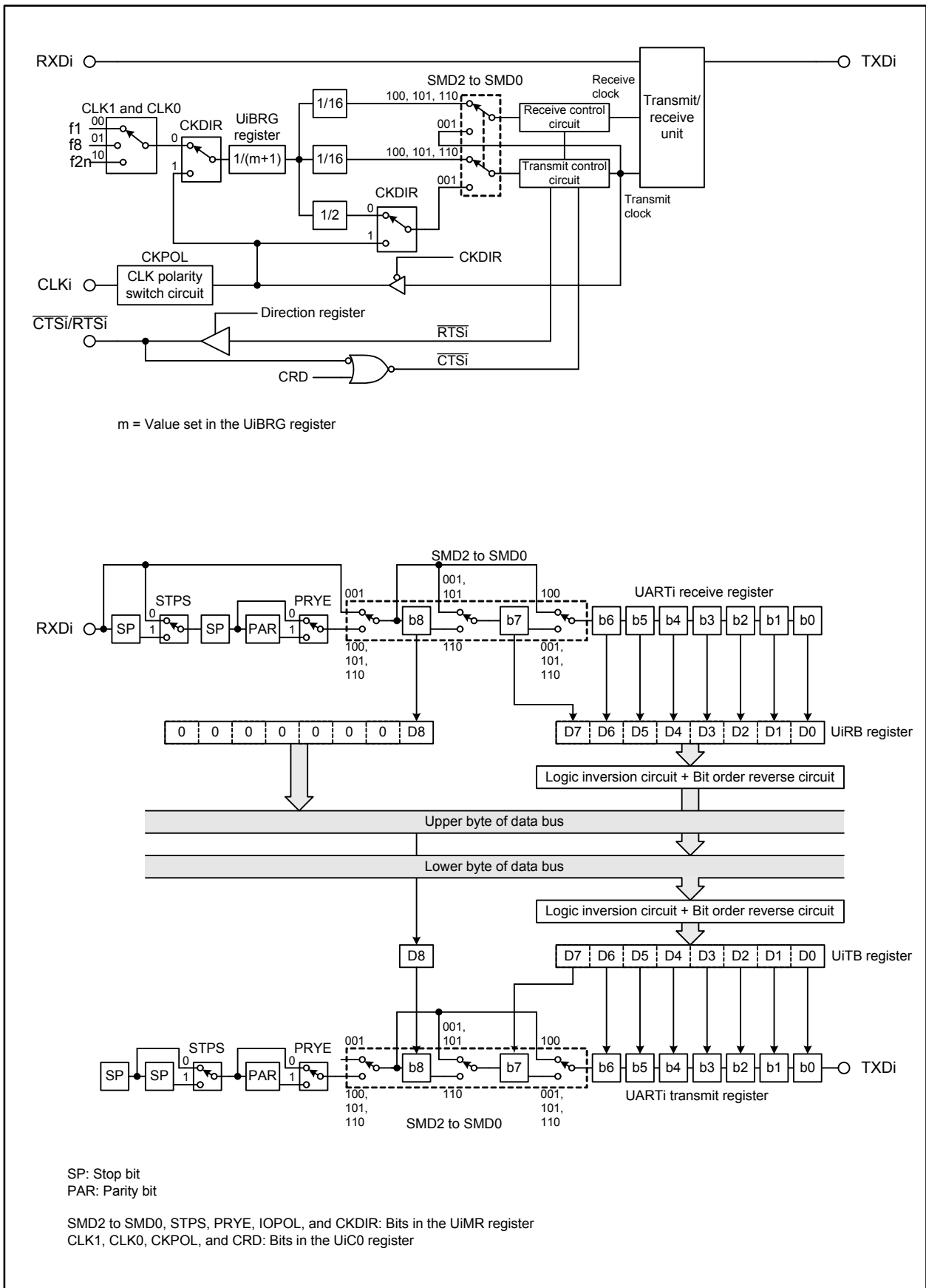


Figure 17.2 UARTi Block Diagram (i = 3, 4)

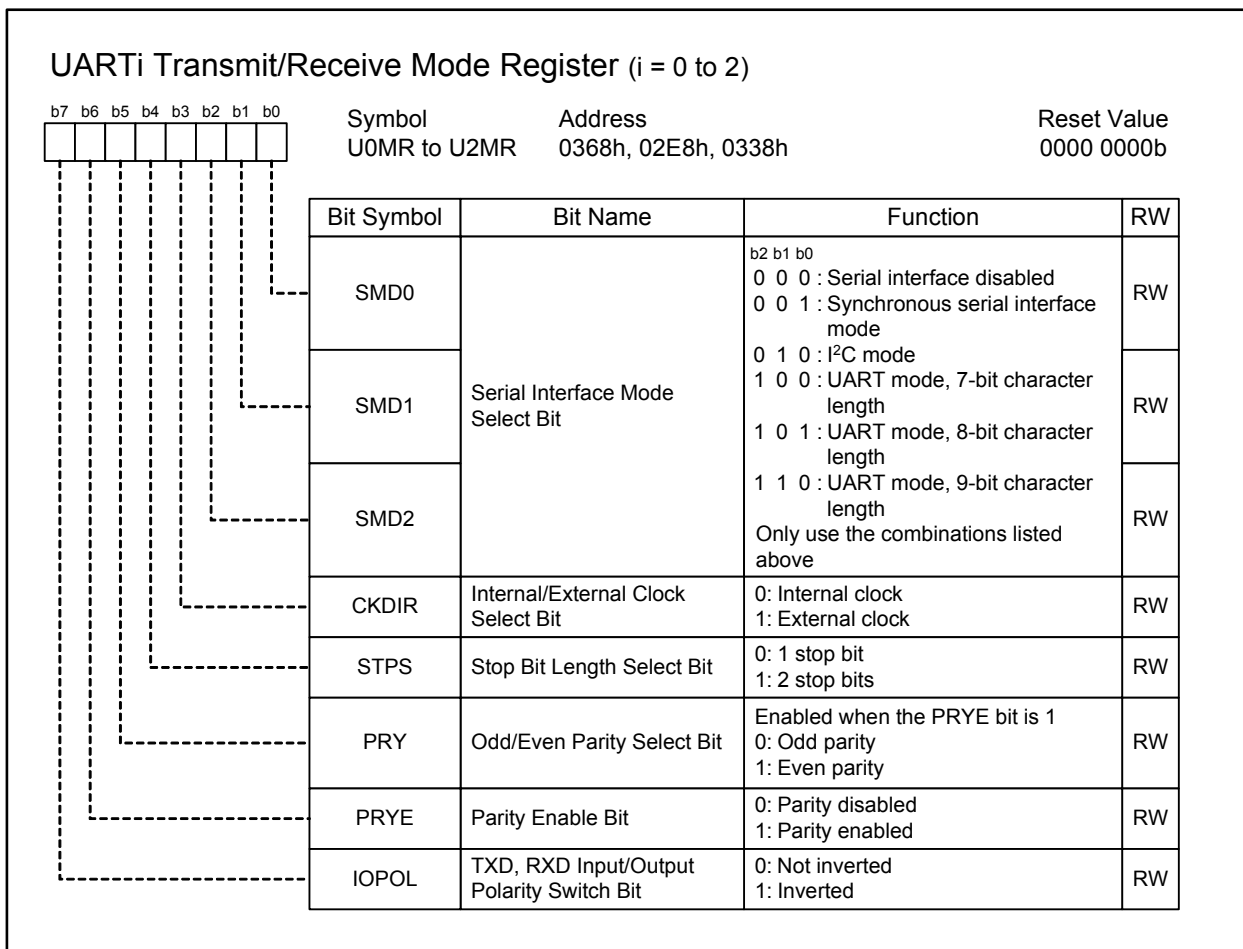


Figure 17.3 Registers U0MR to U2MR

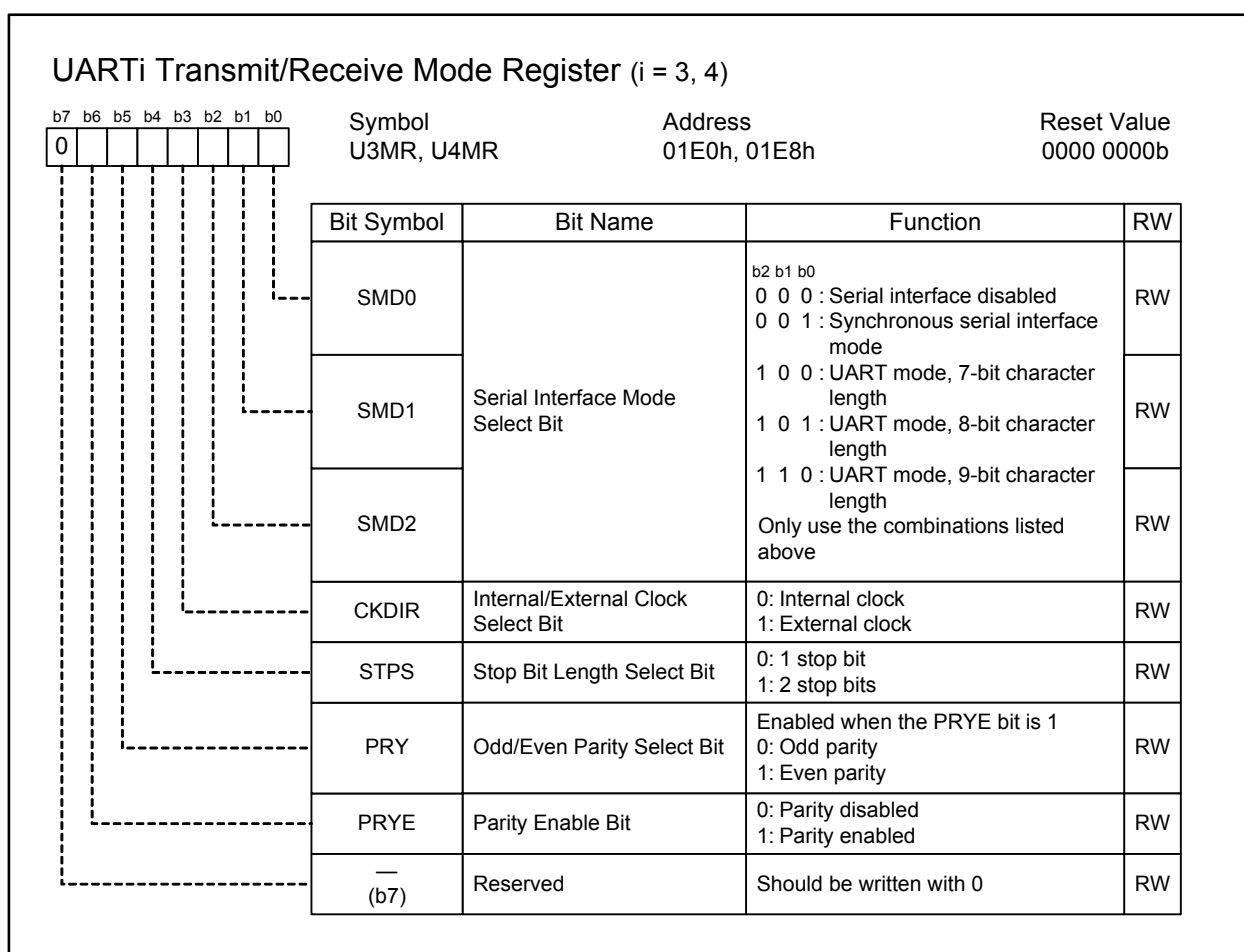
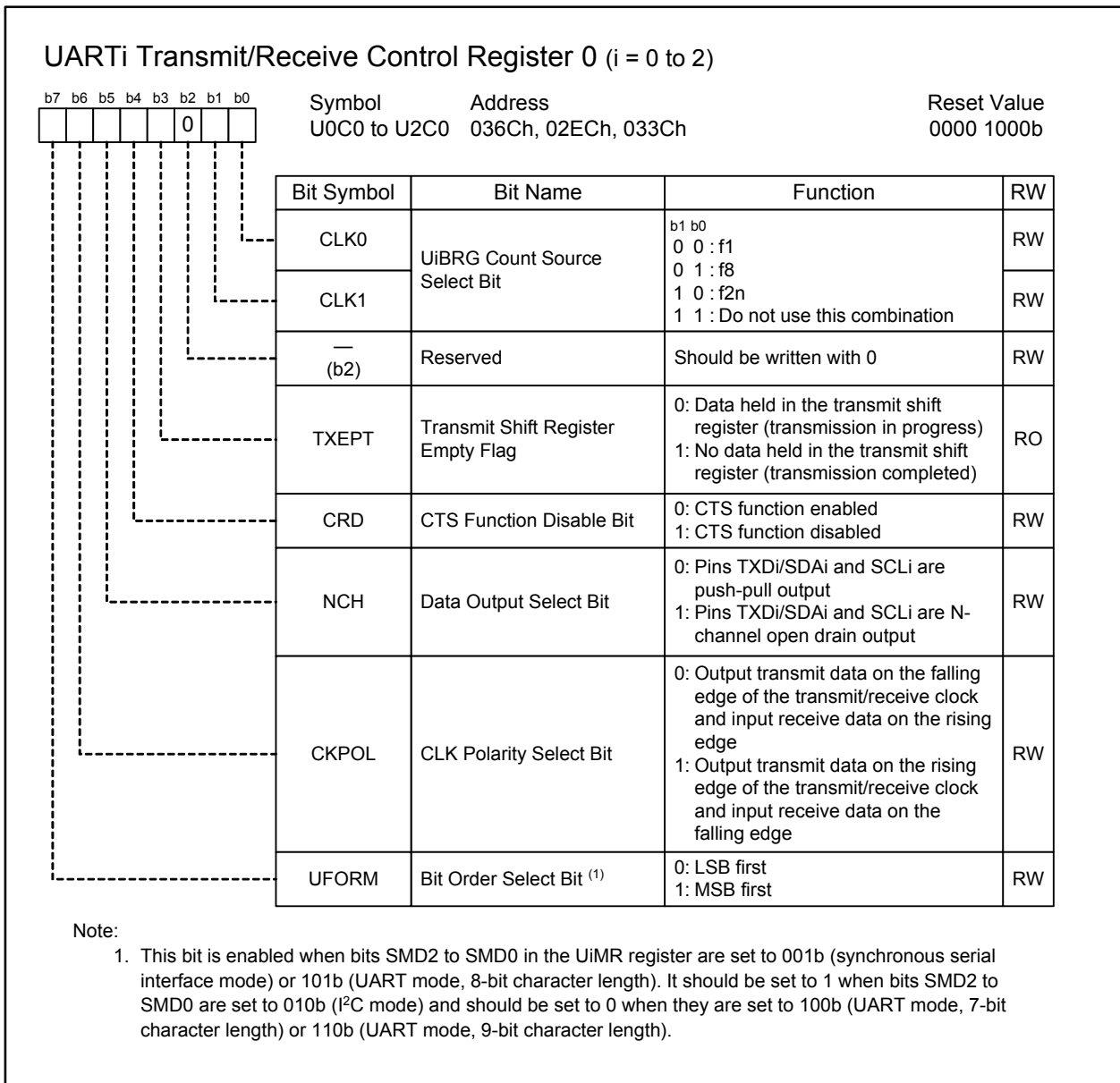
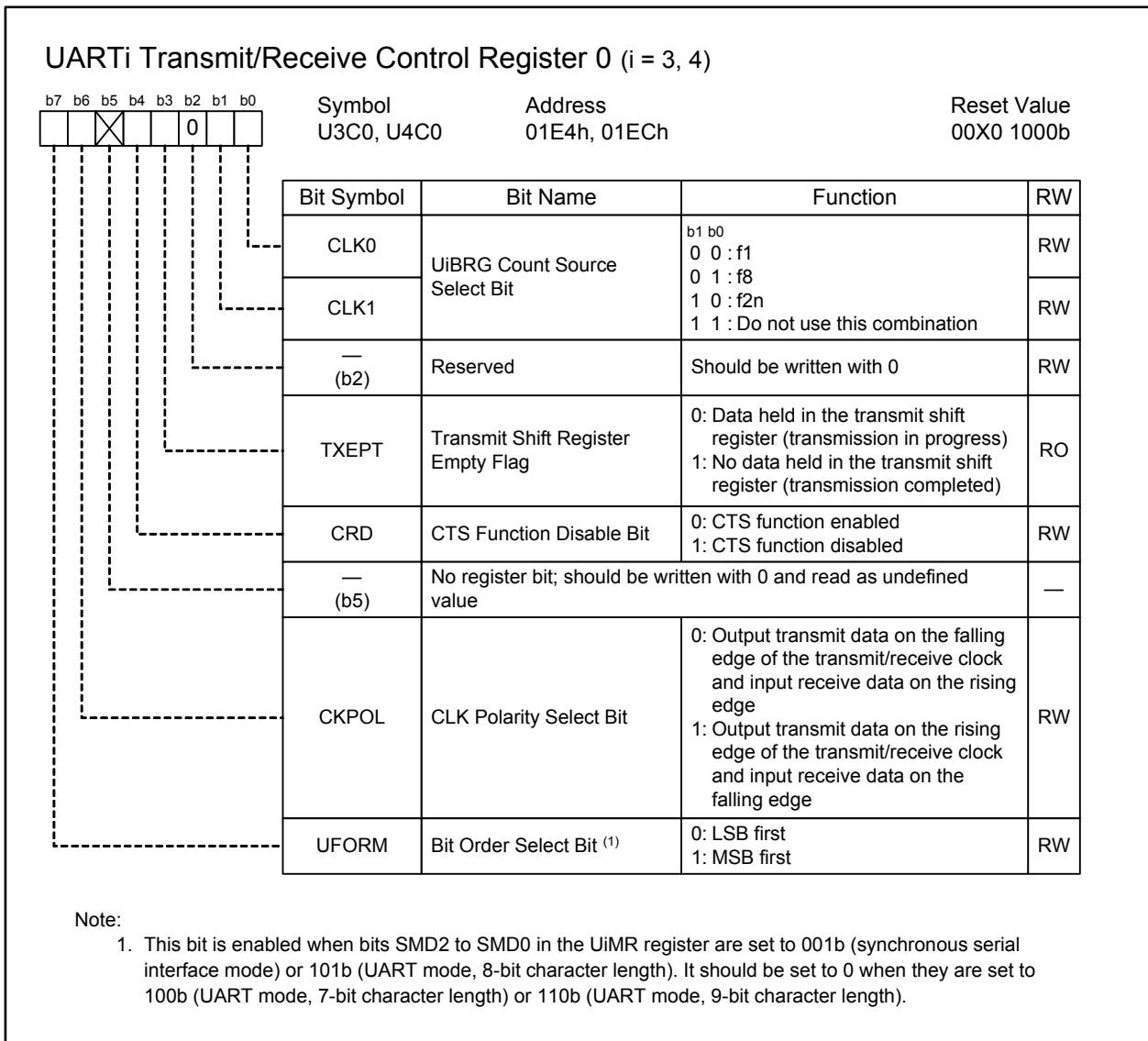


Figure 17.4 Registers U3MR and U4MR



**Figure 17.5 Registers U0C0 to U2C0**





**Figure 17.6 Registers U3C0 and U4C0**

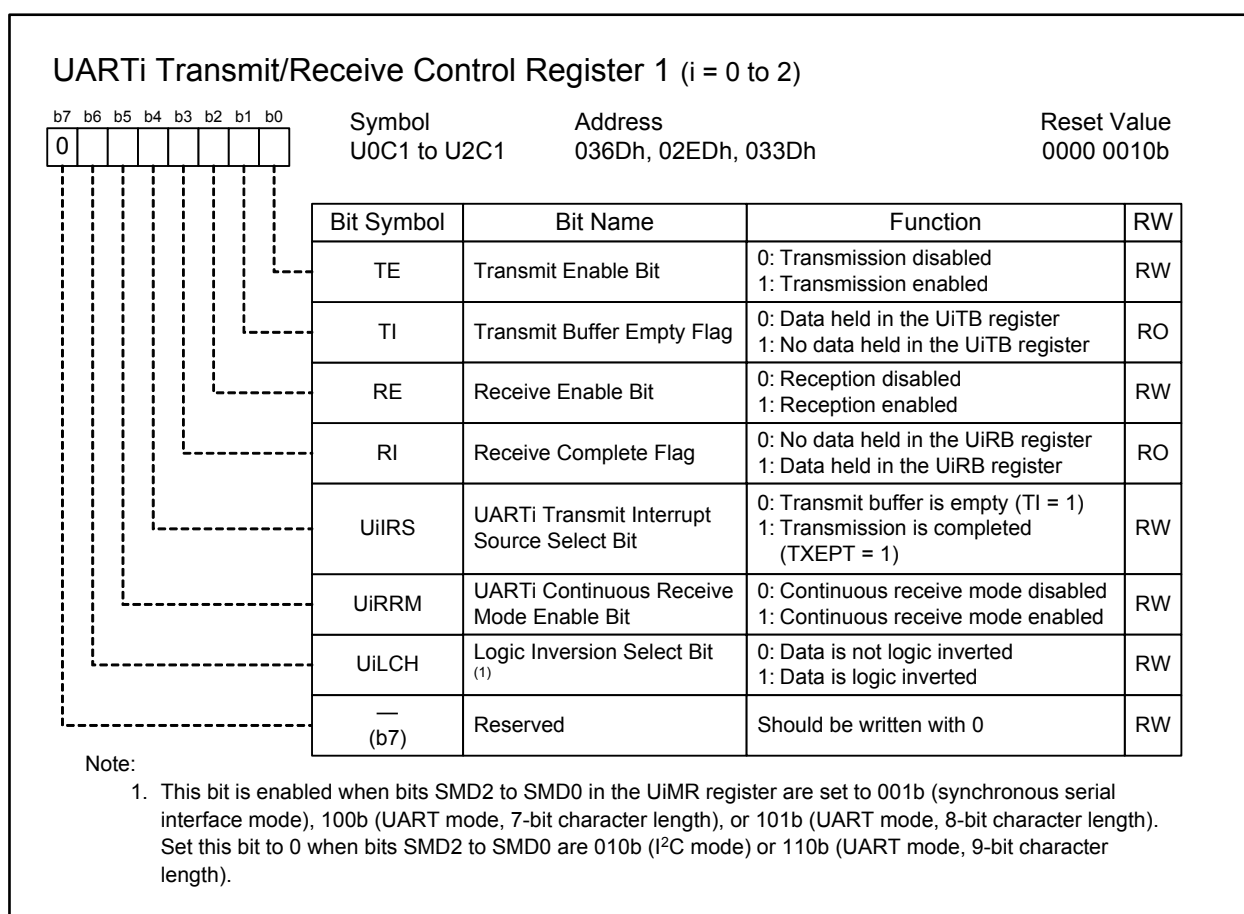


Figure 17.7 Registers U0C1 to U2C1

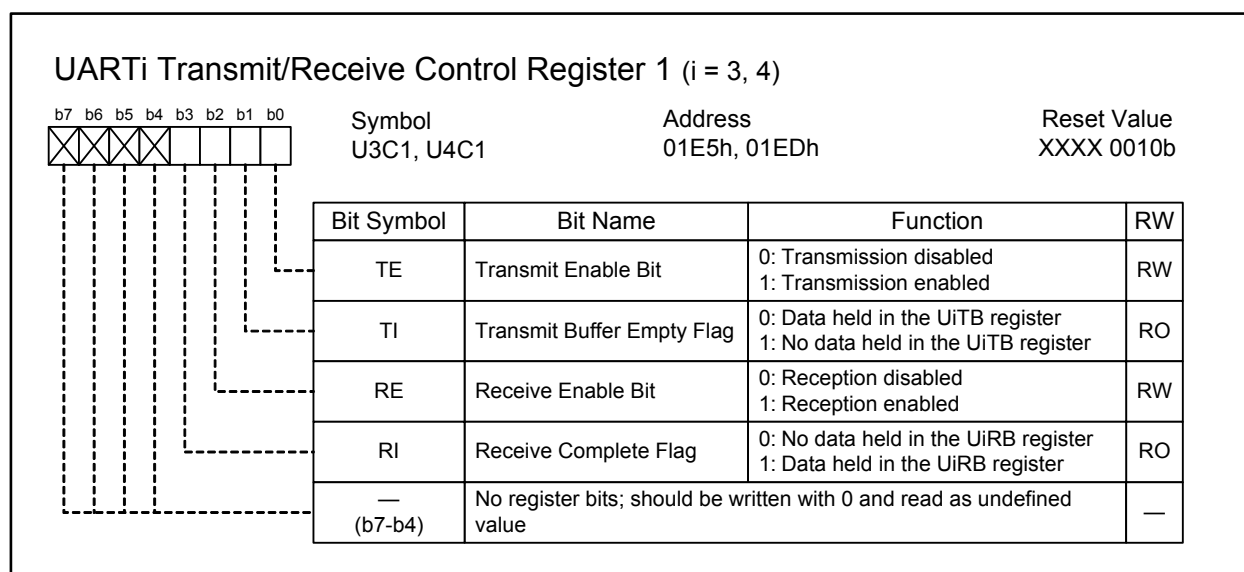


Figure 17.8 Registers U3C1 and U4C1

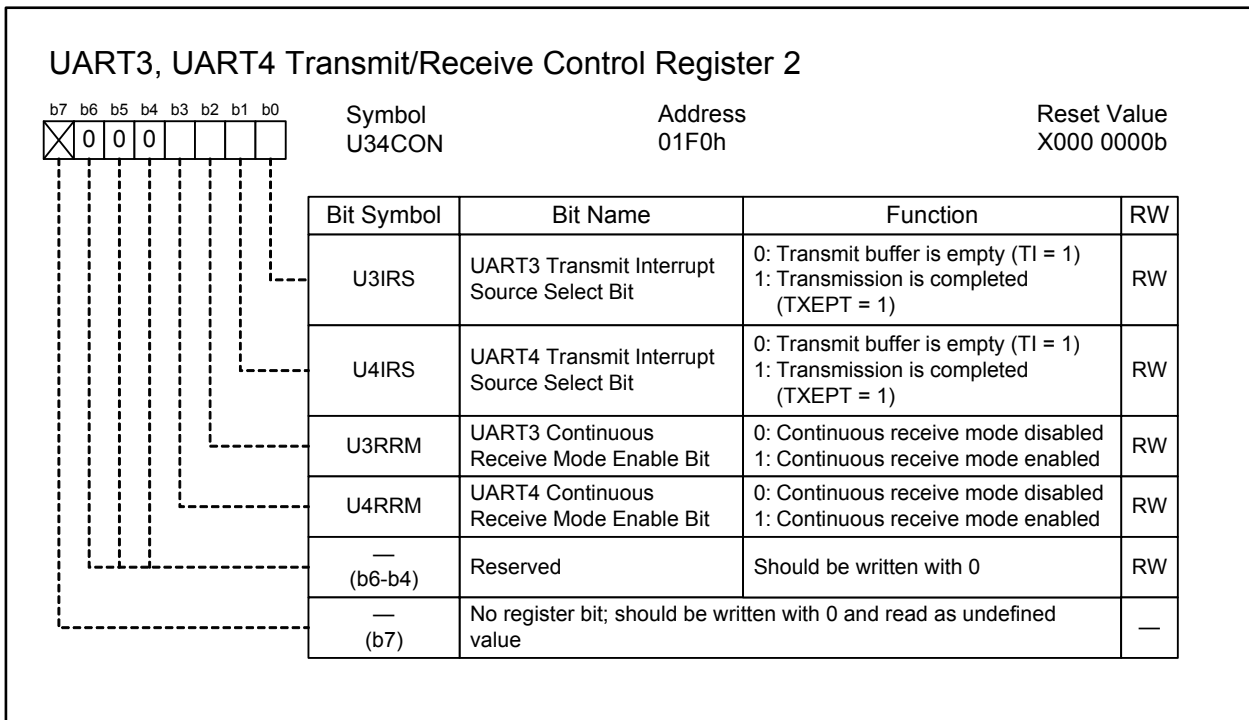


Figure 17.9 U34CON Register

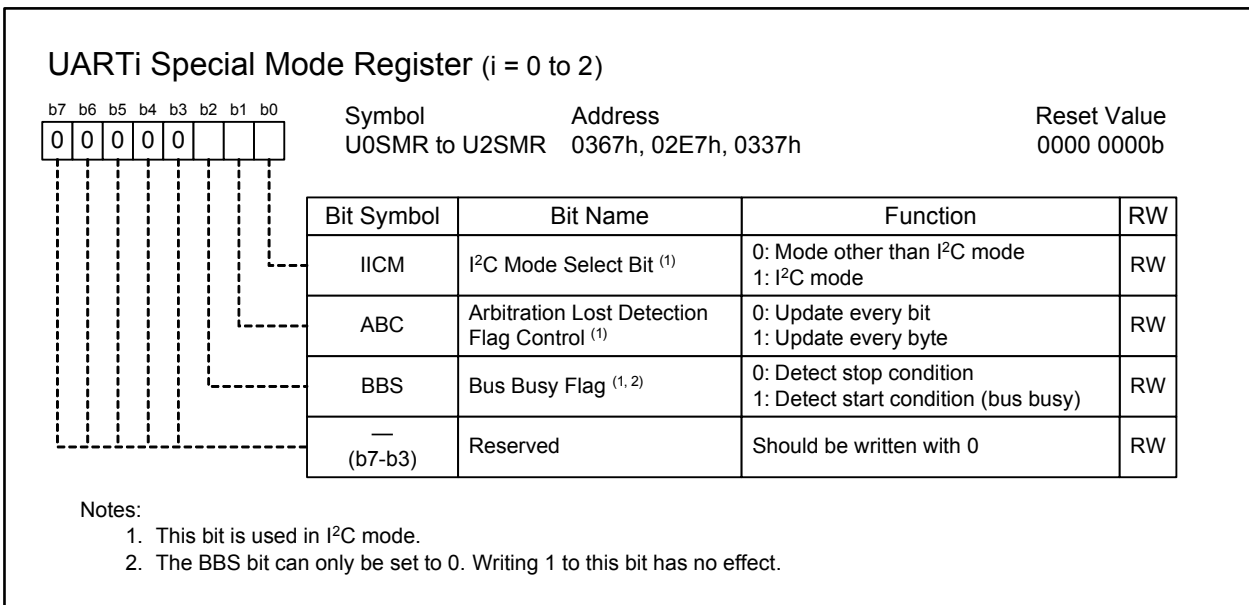
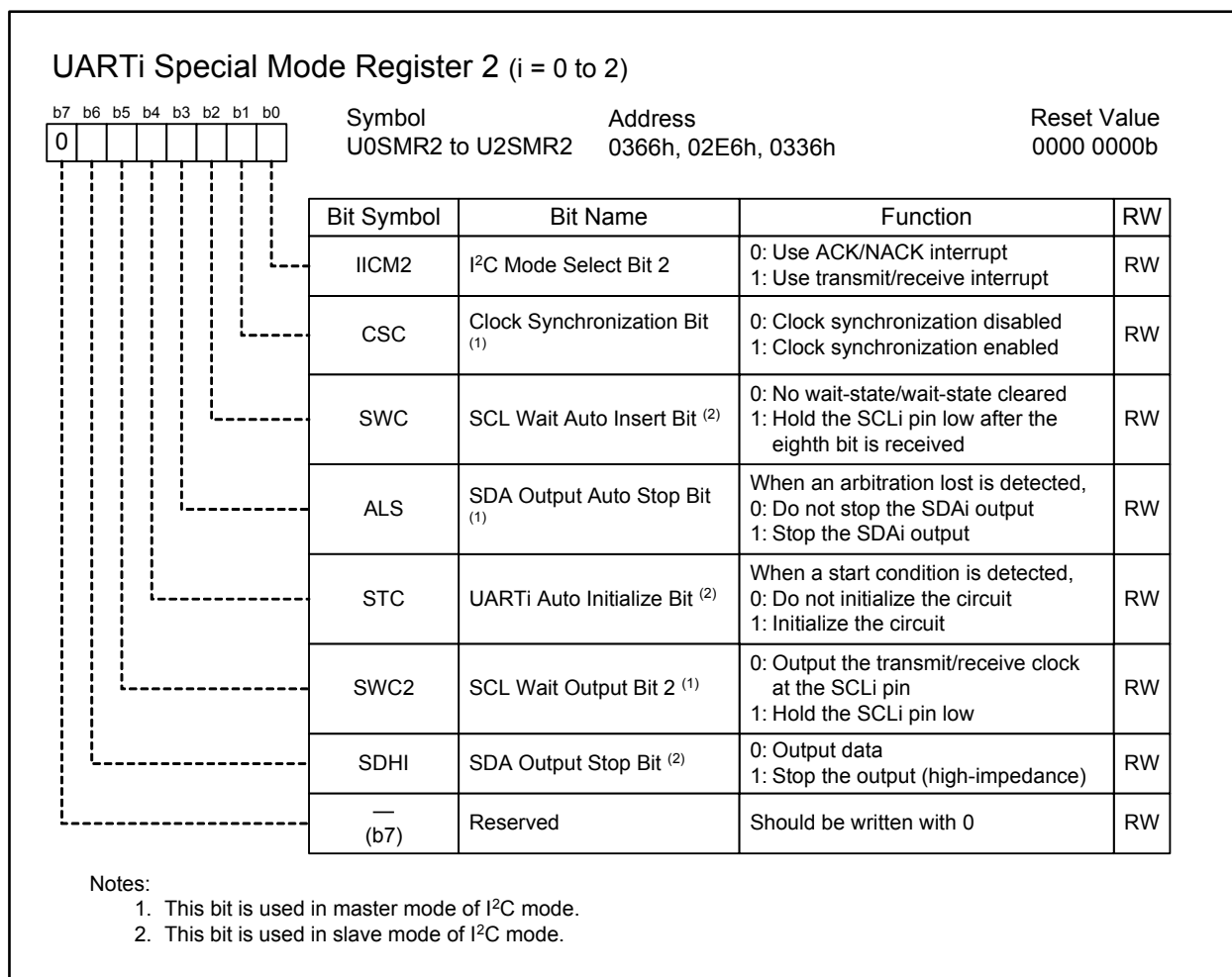


Figure 17.10 Registers U0SMR to U2SMR



**Figure 17.11 Registers U0SMR2 to U2SMR2**

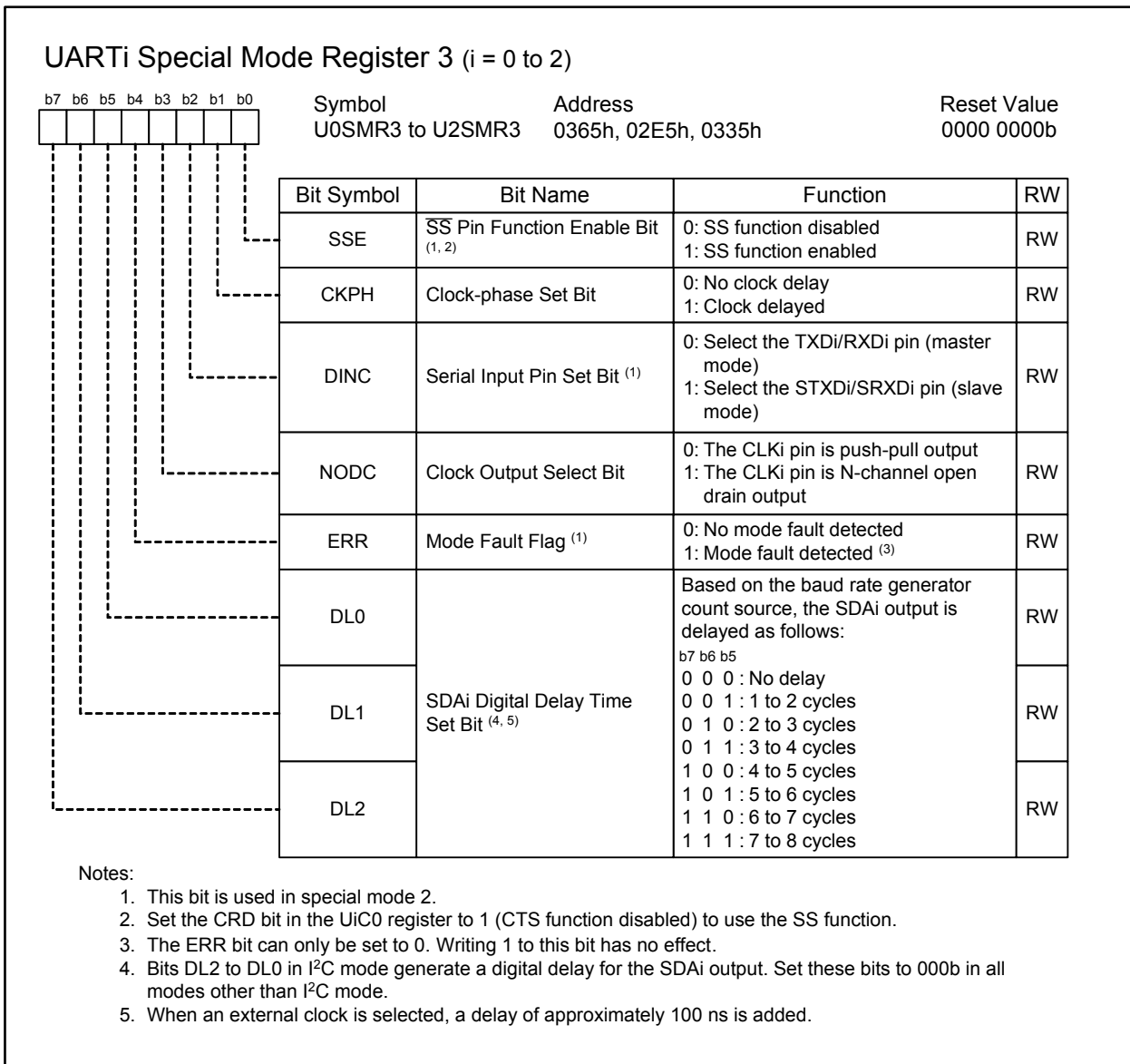


Figure 17.12 Registers U0SMR3 to U2SMR3

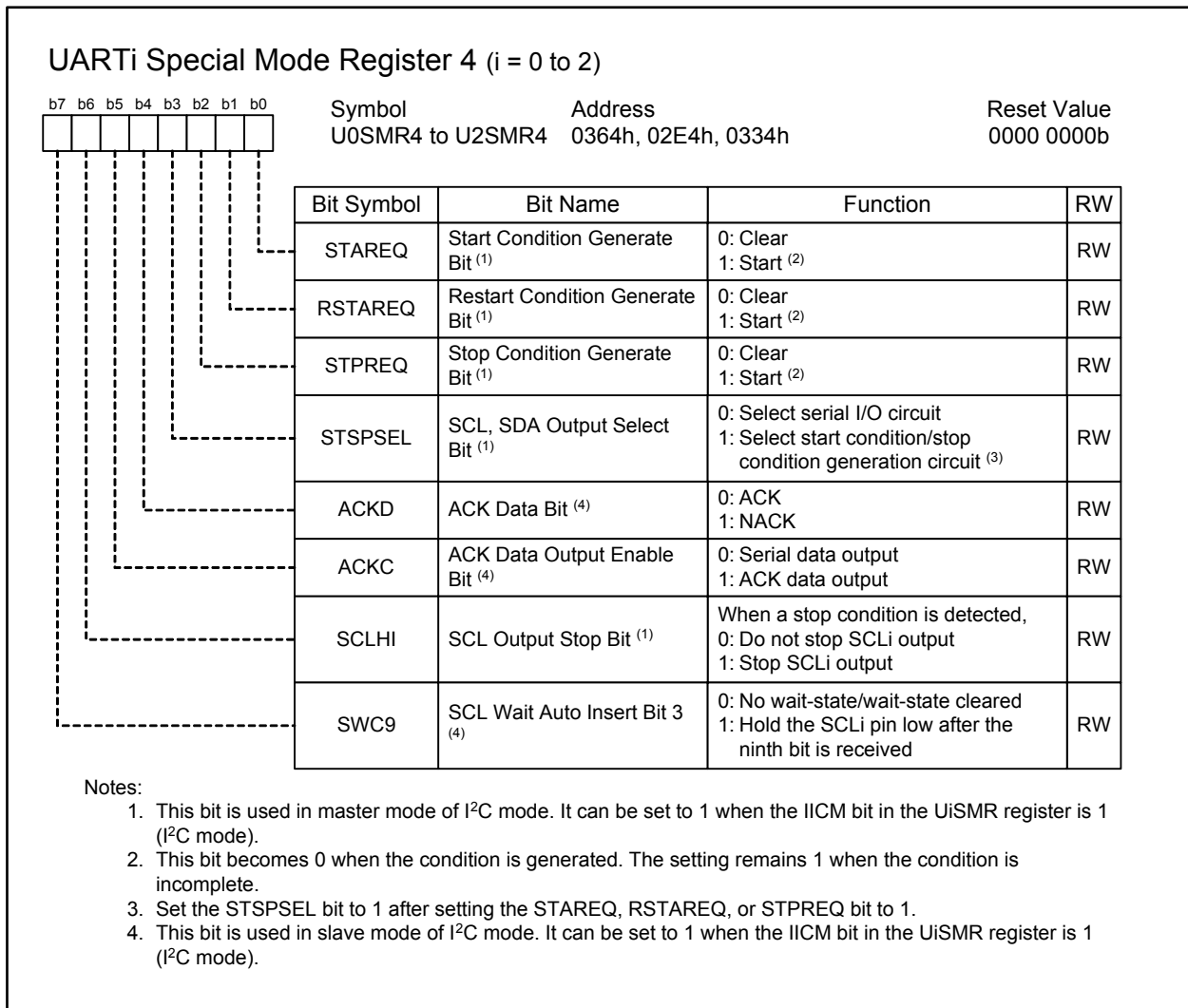


Figure 17.13 Registers U0SMR4 to U2SMR4

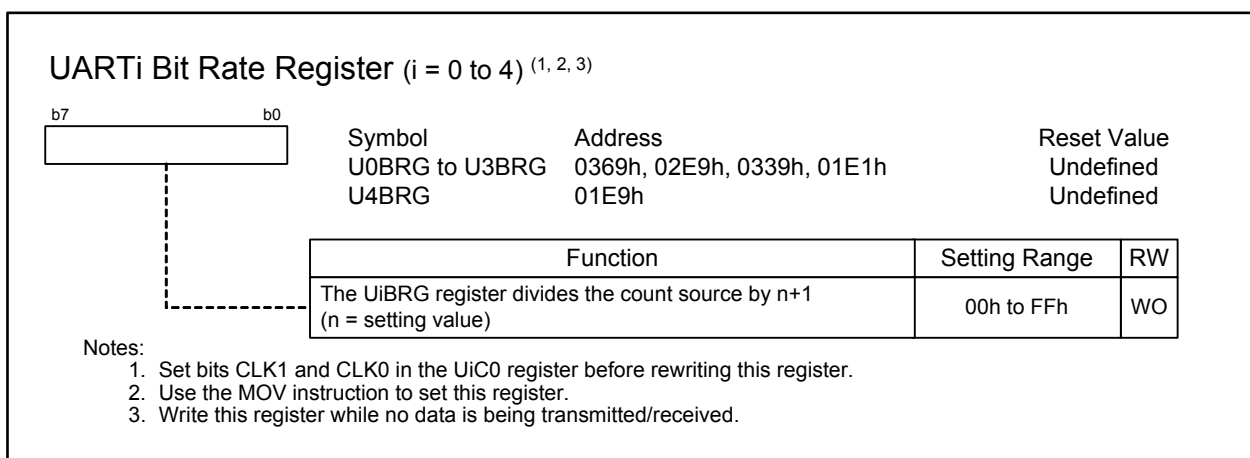


Figure 17.14 Registers U0BRG to U4BRG

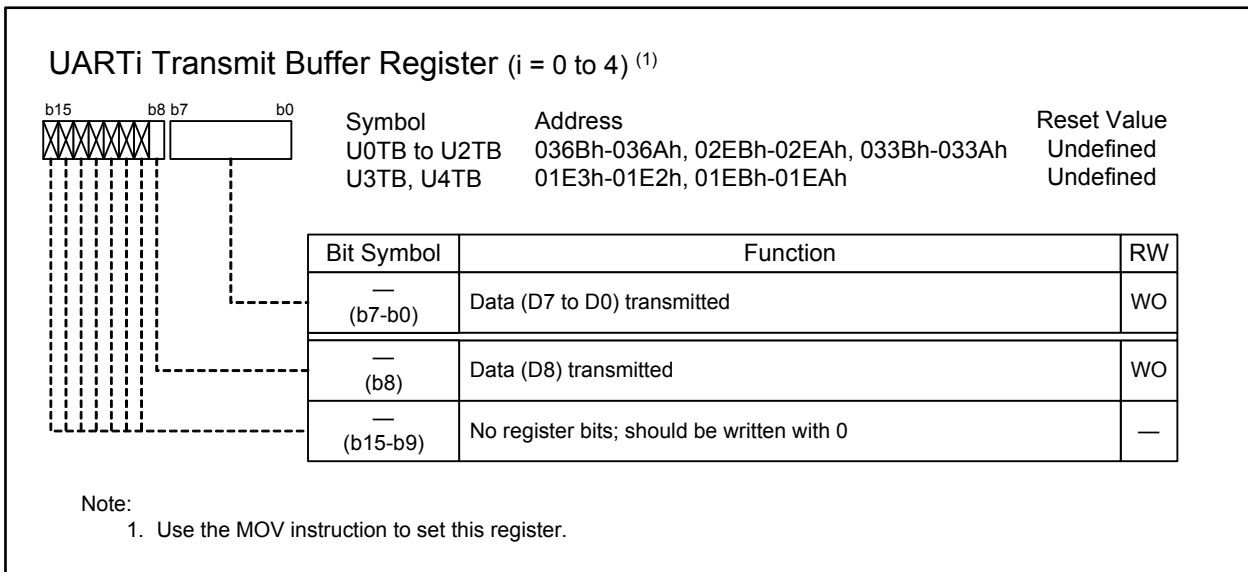


Figure 17.15 Registers U0TB to U4TB

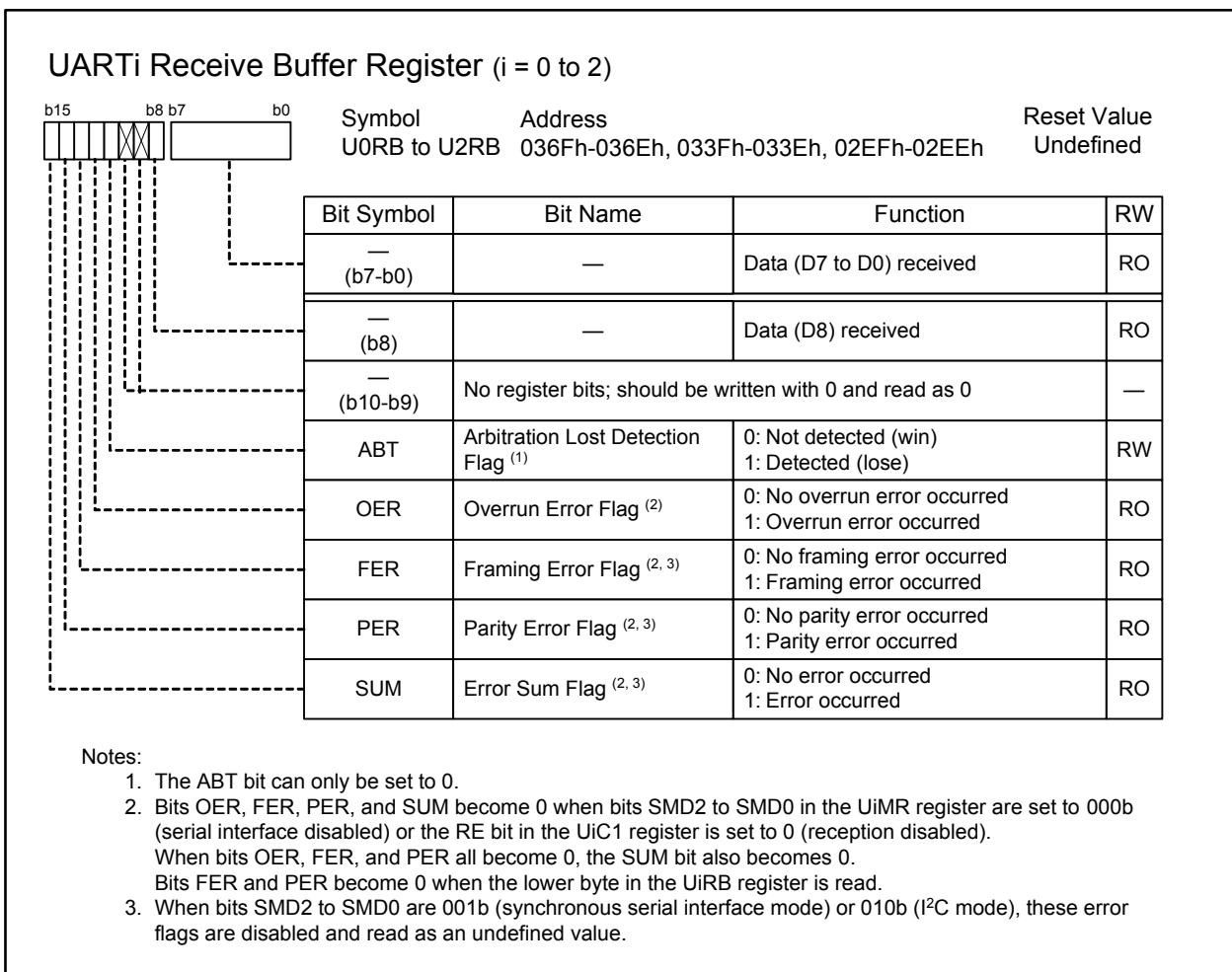
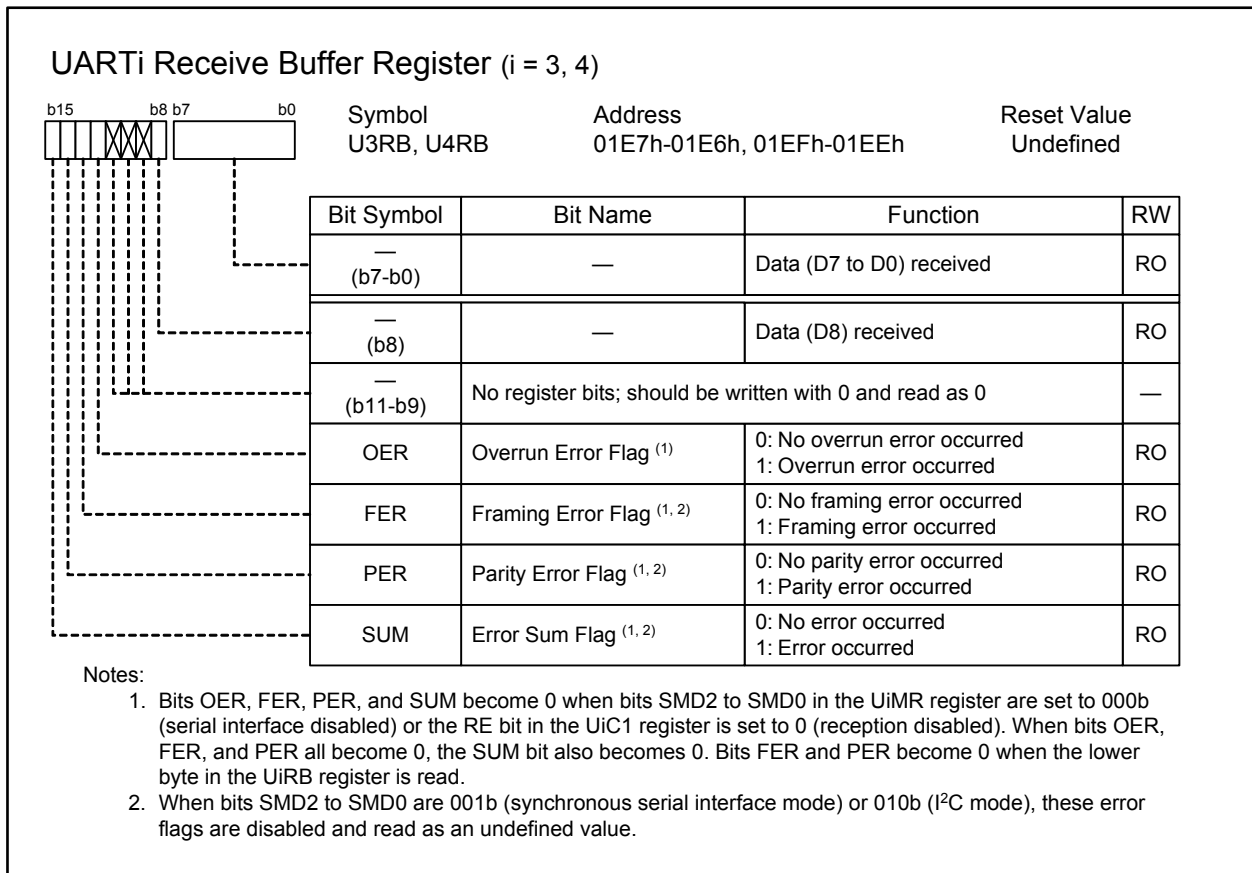


Figure 17.16 Registers U0RB to U2RB



**Figure 17.17 Registers U3RB and U4RB**



## 17.1 Synchronous Serial Interface Mode

The synchronous serial interface mode allows data transmission/reception synchronized with the transmit/receive clock. Table 17.2 lists specifications of synchronous serial interface mode.

**Table 17.2 Synchronous Serial Interface Mode Specifications**

Item	Specification
Data format	8-bit character length
Transmit/receive clock	<ul style="list-style-type: none"> <li>The CKDIR bit in the UiMR register is 0 (internal clock) (<math>i = 0</math> to 4):  <math display="block">\frac{f_x}{2(m+1)} \quad f_x = f_1, f_8, f_{2n}; m: \text{UiBRG register setting value, } 00\text{h to FFh}</math> </li> <li>The CKDIR bit is 1 (external clock): input to the CLKi pin</li> </ul>
Transmit/receive control	CTS function enabled, RTS function enabled, or CTS/RTS function disabled
Transmit start conditions	<p>The conditions for starting data transmission are as follows <sup>(1)</sup>:</p> <ul style="list-style-type: none"> <li>The TE bit in the UiC1 register is 1 (transmission enabled)</li> <li>The TI bit in the UiC1 register is 0 (data held in the UiTB register)</li> <li>Input level at the CTSi pin is low when the CTS function is selected</li> </ul>
Receive start conditions	<p>The conditions for starting data reception are as follows <sup>(1)</sup>:</p> <ul style="list-style-type: none"> <li>The RE bit in the UiC1 register is 1 (reception enabled)</li> <li>The TE bit in the UiC1 register is 1 (transmission enabled)</li> <li>The TI bit in the UiC1 register is 0 (data held in the UiTB register)</li> <li>Input level at the CTSi pin is low when the CTS function is selected</li> </ul>
Interrupt request generating timing	<p>In transmit interrupt, one of the following conditions can be selected by setting the UiIRS bit in registers U0C1 to U2C1 and U34CON:</p> <ul style="list-style-type: none"> <li>The UiIRS bit is 0 (transmit buffer is empty): when data is transferred from the UiTB register to the UARTi transmit register (when the transmission has started)</li> <li>The UiIRS bit is 1 (transmission is completed): when data transmission from the UARTi transmit register is completed</li> </ul> <p>In receive interrupt,</p> <ul style="list-style-type: none"> <li>When data is transferred from the UARTi receive register to the UiRB register (when the reception is completed)</li> </ul>
Error detection	<p>Overflow error <sup>(2)</sup></p> <p>This error occurs when the seventh bit of the next data is received before the UiRB register is read</p>
Other functions	<ul style="list-style-type: none"> <li>CLK polarity Rising or falling edge of the transmit/receive clock for output and input of transmit/receive data</li> <li>Bit order selection LSB first or MSB first</li> <li>Continuous receive mode Data reception is enabled by a read access to the UiRB register</li> <li>Serial data logic inversion (UART0 to UART2) This function logically inverts transmit/receive data</li> </ul>

Notes:

- When selecting an external clock, the following preconditions should be met:
  - The CLKi pin is held high when the CKPOL bit in the UiC0 register is set to 0 (transmit data output on the falling edge of the transmit/receive clock and receive data input on the rising edge).
  - The CLKi pin is held low when the CKPOL bit is set to 1 (transmit data output on the rising edge of the transmit/receive clock and receive data input on the falling edge).
- The UiRB register is undefined when an overflow error occurs. The IR bit in the SiRIC register does not change to 1 (interrupt requested).

Tables 17.3 and 17.4 list register settings. When UART<sub>i</sub> operating mode is selected, a high is output at the TXD<sub>i</sub> pin until transmission starts (the TXD<sub>i</sub> pin is high-impedance when the N-channel open drain output is selected) (i = 0 to 4).

Figures 17.18 and 17.19 show examples of transmit and receive operations in synchronous serial interface mode, respectively.

**Table 17.3 Register Settings in Synchronous Serial Interface Mode (for UART0 to UART2)**

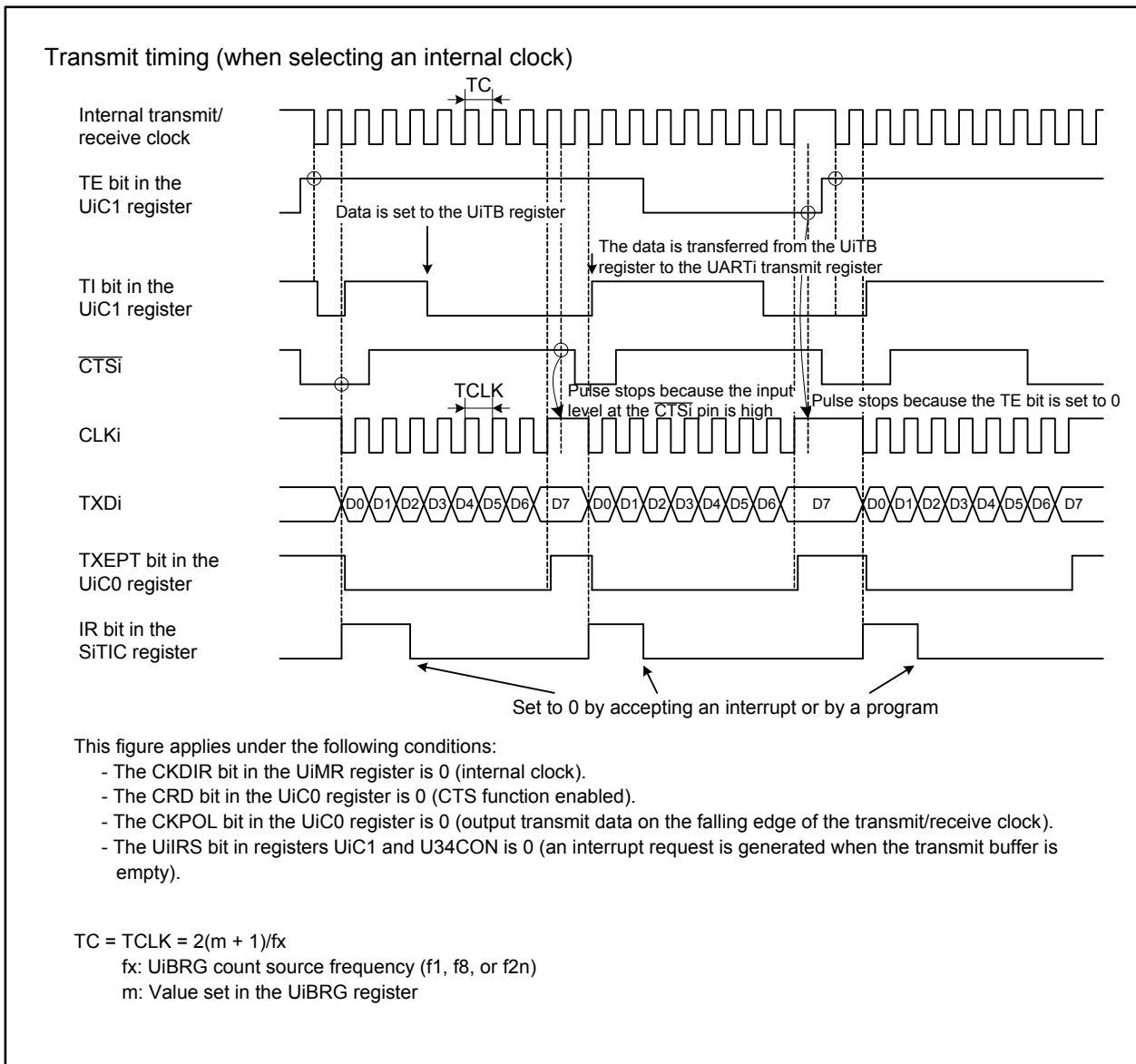
Register	Bits	Function
UiMR	7 to 4	Set the bits to 0000b
	CKDIR	Select either an internal clock or external clock
	SMD2 to SMD0	Set the bits to 001b
UiC0	UFORM	Select either LSB first or MSB first
	CKPOL	Select a transmit/receive clock polarity
	NCH	Select an output mode of the TXD <sub>i</sub> pin
	CRD	Select CTS function enabled or disabled
	TXEPT	Transmit register empty flag
	2	Set the bit to 0
	CLK1 and CLK0	Select a count source for the UiBRG register
UiC1	7	Set the bit to 0
	UiLCH	Set the bit to 1 to use logic inversion
	UiRRM	Set the bit to 1 to use continuous receive mode
	UiIRS	Select a source for the UART <sub>i</sub> transmit interrupt
	RI	Receive complete flag
	RE	Set the bit to 1 to enable data reception
	TI	Transmit buffer empty flag
	TE	Set the bit to 1 to enable data transmission/reception
UiSMR	7 to 0	Set the bits to 00h
UiSMR2	7 to 0	Set the bits to 00h
UiSMR3	7 to 4	Set the bits to 0000b
	NODC	Select a clock output mode
	2 to 0	Set the bits to 000b
UiSMR4	7 to 0	Set the bits to 00h
UiBRG	7 to 0	Set the bit rate
UiTB	7 to 0	Set the data to be transmitted
UiRB	OER	Overrun error flag
	7 to 0	Received data is read

i = 0 to 2

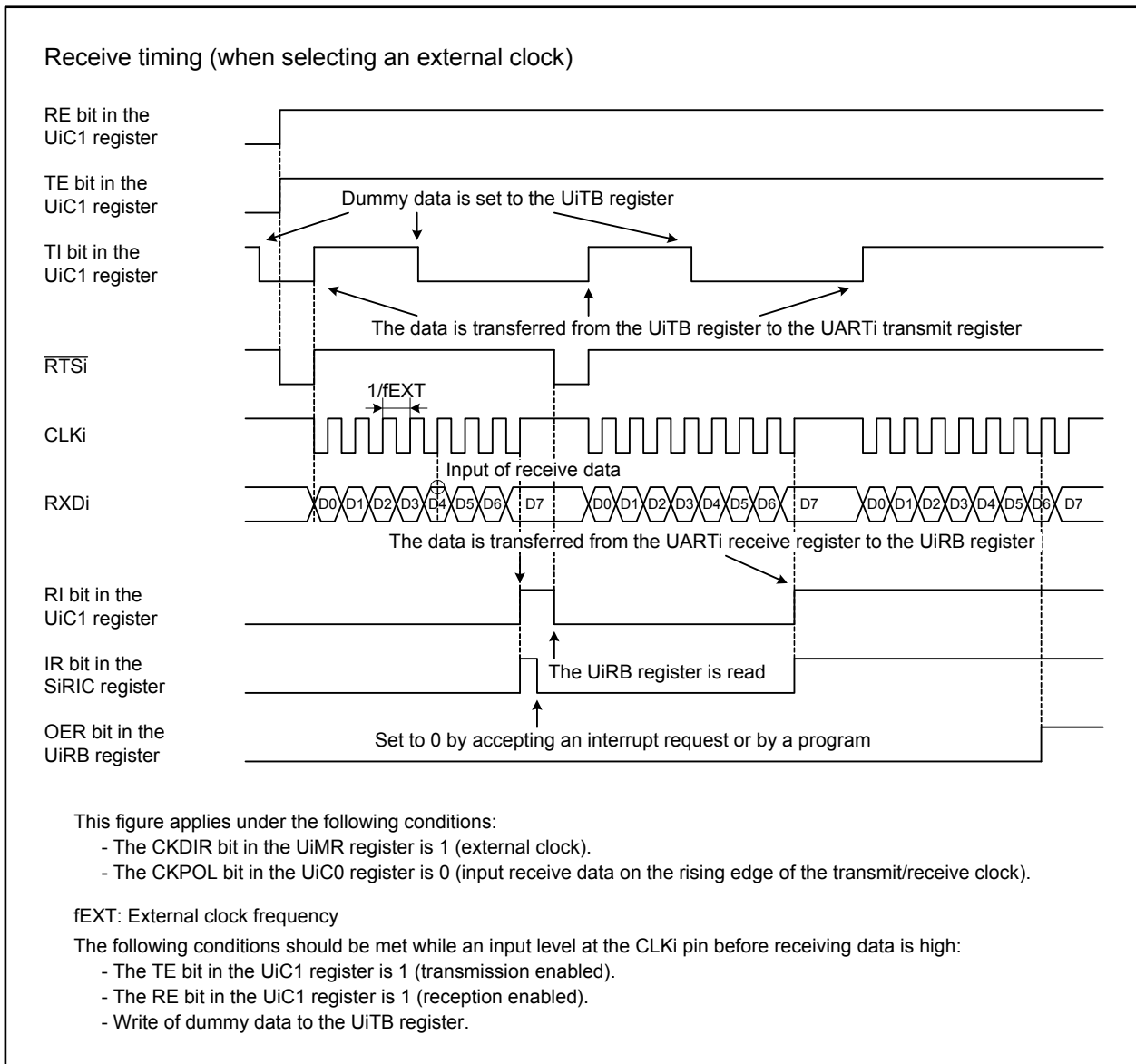
**Table 17.4 Register Settings in Synchronous Serial Interface Mode (for UART3 and UART4)**

Register	Bits	Function
UiMR	7 to 4	Set the bits to 0000b
	CKDIR	Select an internal clock or external clock
	SMD2 to SMD0	Set the bits to 001b
UiC0	UFORM	Select either LSB first or MSB first
	CKPOL	Select a transmit/receive clock polarity
	5	Set the bit to 0
	CRD	Select CTS function enabled or disabled
	TXEPT	Transmit register empty flag
	2	Set the bit to 0
	CLK1 and CLK0	Select a count source for the UiBRG register
UiC1	RI	Receive complete flag
	RE	Set the bit to 1 to enable data reception
	TI	Transmit buffer empty flag
	TE	Set the bit to 1 to enable data transmission/reception
U34CON	UiRRM	Set the bit to 1 to use continuous receive mode
	UiIRS	Select an interrupt source for UARTi transmit
IFS0	IFS04	Select input pins for CLK4, RXD4, and CTS4
UiBRG	7 to 0	Set the bit rate
UiTB	7 to 0	Set the data to be transmitted
UiRB	OER	Overrun error flag
	7 to 0	Received data can be read

i = 3, 4



**Figure 17.18 Transmit Operation in Synchronous Serial Interface Mode**



**Figure 17.19 Receive Operation in Synchronous Serial Interface Mode**

### 17.1.1 Reset Procedure on Transmit/Receive Error

When a transmit/receive error occurs in synchronous serial interface mode, follow the procedures below to perform a reset:

#### A. Reset procedure for the UiRB register (i = 0 to 4)

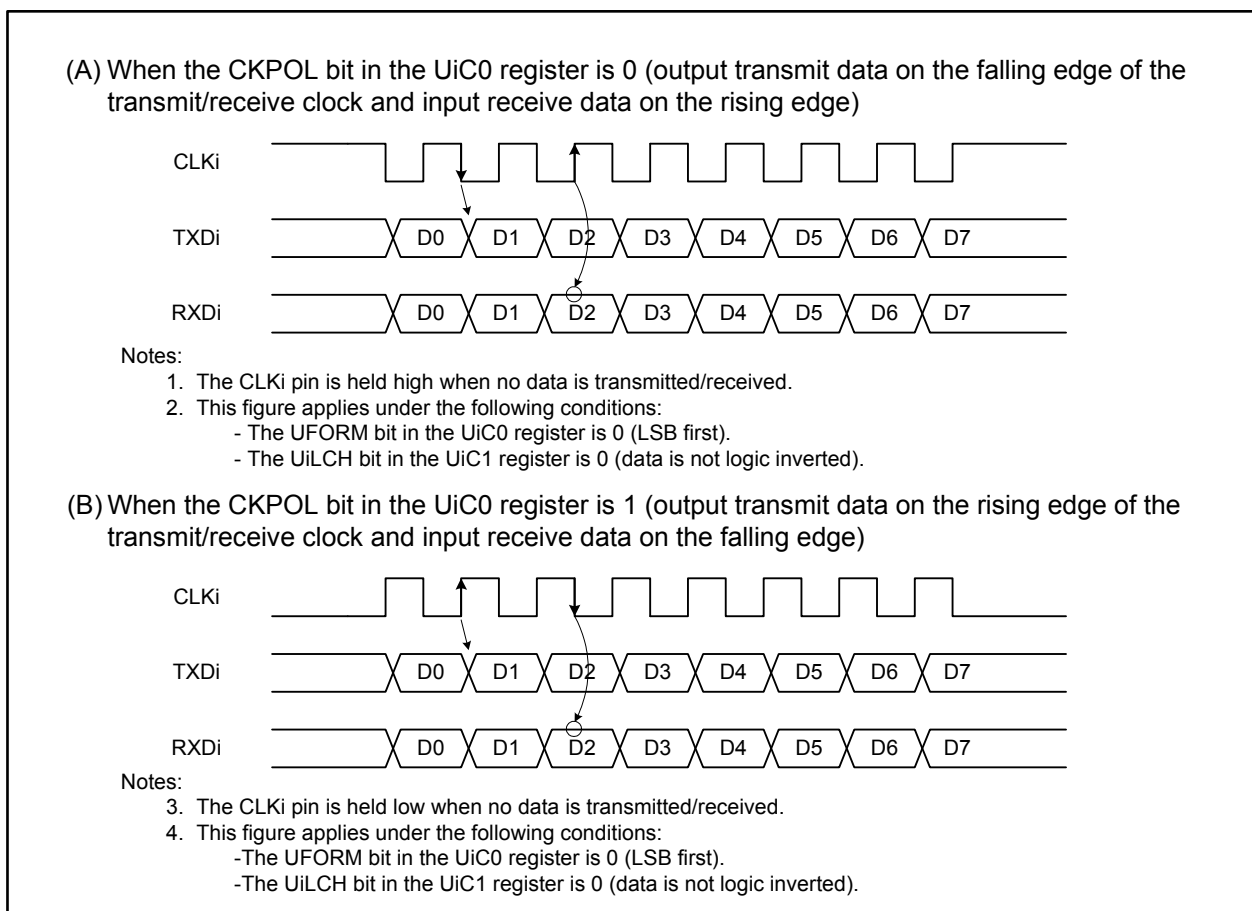
- (1) Set the RE bit in the UiC1 register to 0 (reception disabled).
- (2) Set bits SMD2 to SMD0 in the UiMR register to 000b (serial interface disabled).
- (3) Set bits SMD2 to SMD0 to 001b (synchronous serial interface mode).
- (4) Set the RE bit in the UiC1 register to 1 (reception enabled).

#### B. Reset procedure for the UiTB register

- (1) Set bits SMD2 to SMD0 in the UiMR register to 000b (serial interface disabled).
- (2) Set bits SMD2 to SMD0 to 001b (synchronous serial interface mode).
- (3) Irrespective of its status, set the TE bit in the UiC1 register to 1 (transmission enabled).

### 17.1.2 CLK Polarity

As shown in Figure 17.20, the polarity of the transmit/receive clock is selected using the CKPOL bit in the UiC0 register (i = 0 to 4).



**Figure 17.20 Transmit/Receive Clock Polarity (i = 0 to 4)**

### 17.1.3 LSB First and MSB First Selection

As shown in Figure 17.21, the bit order is selected by setting the UFORM bit in the UiC0 register ( $i = 0$  to 4).

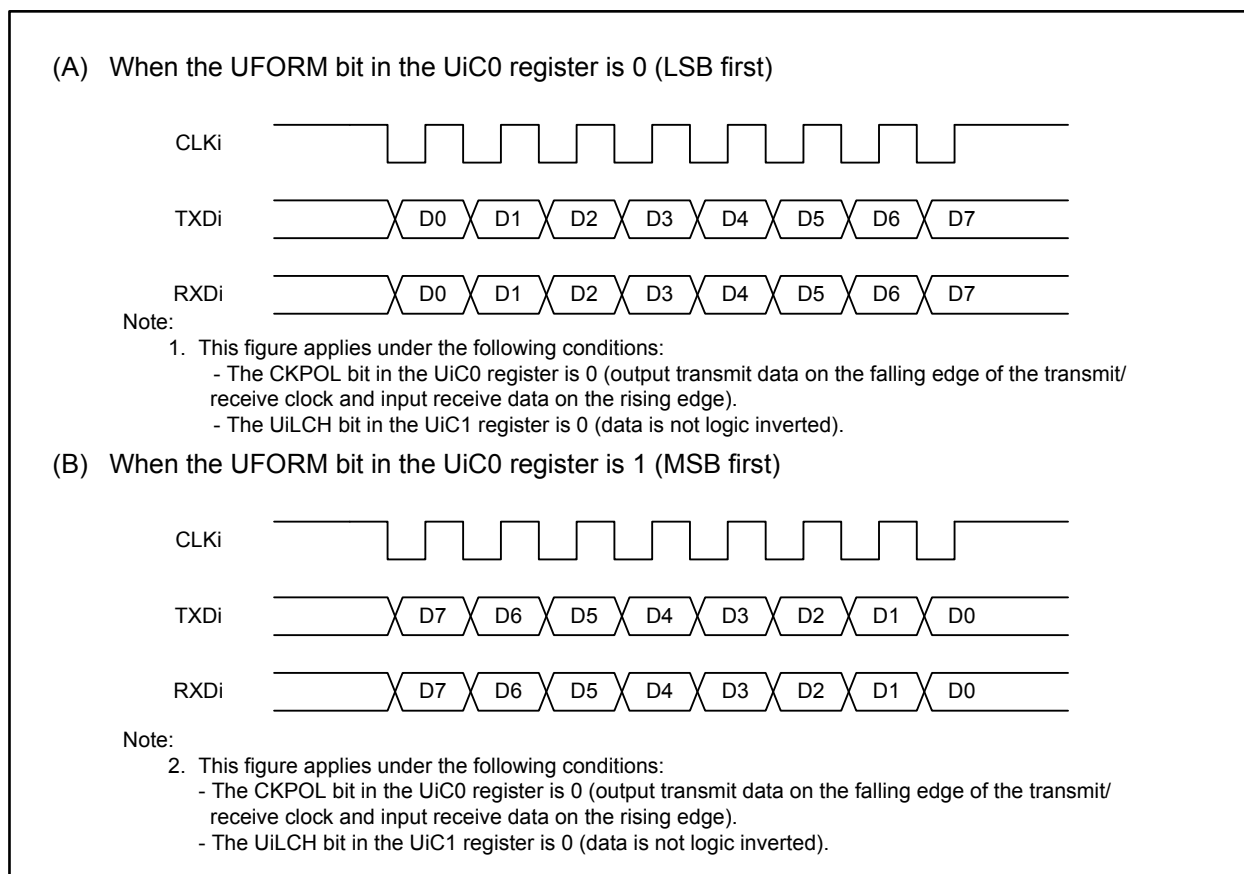


Figure 17.21 Bit Order ( $i = 0$  to 4)

### 17.1.4 Continuous Receive Mode

In continuous receive mode, data reception is automatically enabled by a read access to the receive buffer register without writing dummy data to the transmit buffer register. To start data reception, however, dummy data is required to read the receive buffer register.

When the UiRRM bit in registers U0C1 to U2C1 and the U34CON register is set to 1 (continuous receive mode enabled), the TI bit in the UiC1 register becomes 0 (data held in the UiTB register) by a read access to the UiRB register ( $i = 0$  to 4). In this UiRRM bit setting, no dummy data should be written to the UiTB register.

### 17.1.5 Serial Data Logic Inversion

When the UiLCH bit in the UiC1 register is 1 (data is logic inverted), the logical value written in the UiTB register is inverted before being transmitted ( $i = 0$  to 2). The UiRB register is read as logic-inverted receive data. Figure 17.22 shows the logic inversion of serial data.

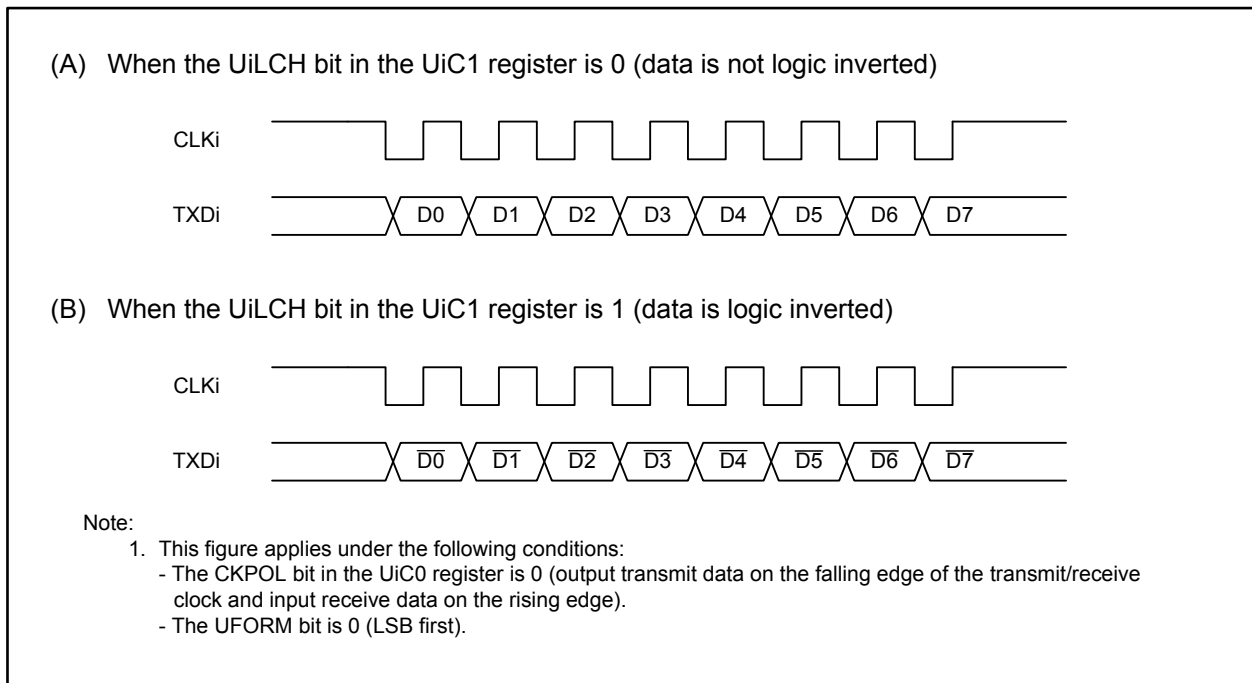


Figure 17.22 Serial Data Logic Inversion ( $i = 0$  to 2)

### 17.1.6 CTS/RTS Function

CTS function controls data transmission using the  $\overline{\text{CTS}}_i/\overline{\text{RTS}}_i$  pin ( $i = 0$  to 4). When an input level at the pin becomes low, data transmission starts. If the input level changes to high during transmission, the transmission of the next data is stopped.

In synchronous serial interface mode, the transmitter is required to operate even during the receive operation. If CTS function is enabled, the input level at the  $\overline{\text{CTS}}_i/\overline{\text{RTS}}_i$  pin should be low to start data reception as well.

RTS function indicates receiver status using the  $\overline{\text{CTS}}_i/\overline{\text{RTS}}_i$  pin. When data reception is ready, the output level at the pin becomes low. It becomes high on the first falling edge of the CLKi pin.



## 17.2 Asynchronous Serial Interface Mode (UART Mode)

The UART mode enables data transmission/reception synchronized with an internal clock generated by a trigger on the falling edge of the start bit. Table 17.5 lists specifications of UART mode.

**Table 17.5 UART Mode Specifications**

Item	Specification
Data format	<ul style="list-style-type: none"> <li>Start bit: 1-bit</li> <li>Data bit (data character): 7-bit, 8-bit, or 9-bit</li> <li>Parity bit: odd, even, or none</li> <li>Stop bit: 1-bit or 2-bit</li> </ul>
Transmit/receive clock	<ul style="list-style-type: none"> <li>The CKDIR bit in the UiMR register is 0 (internal clock) (<math>i = 0</math> to 4):  <math display="block">\frac{f_x}{16(m+1)} \quad f_x = f1, f8, f2n; m: \text{UiBRG register setting value, 00h to FFh}</math> </li> <li>The CKDIR bit is 1 (external clock)  <math display="block">\frac{f_{EXT}}{16(m+1)} \quad f_{EXT}: \text{Clock applied to the CLKi pin}</math> </li> </ul>
Transmit/receive control	CTS function enabled, RTS function enabled, or CTS/RTS function disabled
Transmit start conditions	<p>The conditions for starting data transmission are as follows:</p> <ul style="list-style-type: none"> <li>The TE bit in the UiC1 register is 1 (transmission enabled)</li> <li>The TI bit in the UiC1 register is 0 (data held in the UiTB register)</li> <li>Input level at the CTS<math>_i</math> pin is low when CTS function is selected</li> </ul>
Receive start conditions	<p>The conditions for starting data reception are as follows:</p> <ul style="list-style-type: none"> <li>The RE bit in the UiC1 register is 1 (reception enabled)</li> <li>The start bit is detected</li> </ul>
Interrupt request generating timing	<p>In transmit interrupt, one of the following conditions can be selected by setting the UiIRS bit in registers U0C1 to U2C1 and the U34CON register:</p> <ul style="list-style-type: none"> <li>The UiIRS bit is 0 (transmit buffer is empty): when data is transferred from the UiTB register to the UART<math>_i</math> transmit register (when the transmission has started)</li> <li>The UiIRS bit is 1 (transmission is completed): when data transmission from the UART<math>_i</math> transmit register is completed</li> </ul> <p>In receive interrupt,</p> <ul style="list-style-type: none"> <li>When data is transferred from the UART<math>_i</math> receive register to the UiRB register (when reception is completed)</li> </ul>
Error detection	<ul style="list-style-type: none"> <li>Overrun error <sup>(1)</sup> This error occurs when 1 bit prior to the stop bit (when 1 stop bit length is selected) or the first stop bit (when 2 stop bit length is selected) of the next data is received before the UiRB register is read</li> <li>Framing error This error occurs when the required number of stop bits is not detected</li> <li>Parity error This error occurs when an even number of 1's in parity and character bits is detected while the odd number is set, or vice versa. The parity should be enabled</li> <li>Error sum flag This flag becomes 1 when any of overrun error, framing error, or parity error occurs</li> </ul>
Other functions	<ul style="list-style-type: none"> <li>Bit order selection LSB first or MSB first</li> <li>Serial data logic inversion This function logically inverts transmit/receive data. The start bit and stop bit are not inverted</li> <li>TXD/RXD I/O polarity switching The output level from the TXD pin and the input level to the RXD pin are inverted. All I/O levels are inverted</li> </ul>

Note:

- The UiRB register is undefined when an overrun error occurs. The IR bit in the SiRIC register does not change to 1 (interrupt requested).

Tables 17.6 and 17.7 list register settings. When UART<sub>i</sub> operating mode is selected, a high is output at the TXD<sub>i</sub> pin until transmission starts (the TXD<sub>i</sub> pin is high-impedance when the N-channel open drain output is selected) (i = 0 to 4).

Figures 17.23 and 17.24 show examples of transmit operations in UART mode. Figure 17.25 shows an example of receive operation.

**Table 17.6 Register Settings in UART Mode (UART0 to UART2)**

Register	Bits	Function	
UiMR	IOPOL	Select I/O polarity of pins TXD and RXD	
	PRY and PRYE	Select parity enabled or disabled, and odd or even	
	STPS	Select a stop bit length	
	CKDIR	Select an internal clock or external clock	
	SMD2 to SMD0		Set the bits to 100b in 7-bit character length
			Set the bits to 101b in 8-bit character length
		Set the bits to 110b in 9-bit character length	
UiC0	UFORM	Select LSB first or MSB first in 8-bit character length. Set the bit to 0 in 7-bit or 9-bit character length	
	CKPOL	Set the bit to 0	
	NCH	Select an output mode for the TXD <sub>i</sub> pin	
	CRD	Select CTS function enabled or disabled	
	TXEPT	Transmit register empty flag	
	2	Set the bit to 0	
	CLK1 and CLK0	Select a count source for the UiBRG register	
UiC1	7	Set the bit to 0	
	UiLCH	Set the bit to 1 to use logic inversion	
	UiRRM	Set the bit to 0	
	UiIRS	Select an interrupt source for UART <sub>i</sub> transmission	
	RI	Receive complete flag	
	RE	Set the bit to 1 to enable data reception	
	TI	Transmit buffer empty flag	
	TE	Set the bit to 1 to enable data transmission	
UiSMR	7 to 0	Set the bits to 00h	
UiSMR2	7 to 0	Set the bits to 00h	
UiSMR3	7 to 0	Set the bits to 00h	
UiSMR4	7 to 0	Set the bits to 00h	
UiBRG	7 to 0	Set the bit rate	
UiTB	8 to 0	Set the data to be transmitted <sup>(1)</sup>	
UiRB	OER, FER, PER, and SUM	Error flag	
	8 to 0	Received data is read <sup>(1)</sup>	

i = 0 to 2

Note:

- The bits used are as follows: 7-bit character length: bits 6 to 0  
8-bit character length: bits 7 to 0  
9-bit character length: bits 8 to 0

**Table 17.7 Register Settings in UART Mode (UART3, UART4)**

Register	Bits	Function
UiMR	PRY and PRYE	Select parity enabled or disabled, and odd or even
	STPS	Select a stop bit length
	CKDIR	Select an internal clock or external clock
	SMD2 to SMD0	Set the bits to 100b in 7-bit character length Set the bits to 101b in 8-bit character length Set the bits to 110b in 9-bit character length
UiC0	UFORM	Select LSB first or MSB first in 8-bit character length. Set the bit to 0 in 7-bit or 9-bit character length
	CKPOL	Set the bit to 0
	5	Set the bit to 0
	CRD	Select CTS function enabled or disabled
	TXEPT	Transmit register empty flag
	2	Set the bit to 0
	CLK1 and CLK0	Select a count source for the UiBRG register
UiC1	RI	Receive complete flag
	RE	Set the bit to 1 to enable data reception
	TI	Transmit buffer empty flag
	TE	Set the bit to 1 to enable data transmission
U34CON	UiRRM	Set the bit to 0
	UiIRS	Select an interrupt source for UARTi transmission
UiBRG	7 to 0	Set the bit rate
IFS0	IFS04	Select input pins for CLK4, RXD4, and $\overline{\text{CTS4}}$
UiTB	8 to 0	Set the data to be transmitted <sup>(1)</sup>
UiRB	OER, FER, PER, and SUM	Error flag
	8 to 0	Received data is read <sup>(1)</sup>

i = 3, 4

Note:

- The bits used are as follows: 7-bit character length: bits 6 to 0  
8-bit character length: bits 7 to 0  
9-bit character length: bits 8 to 0

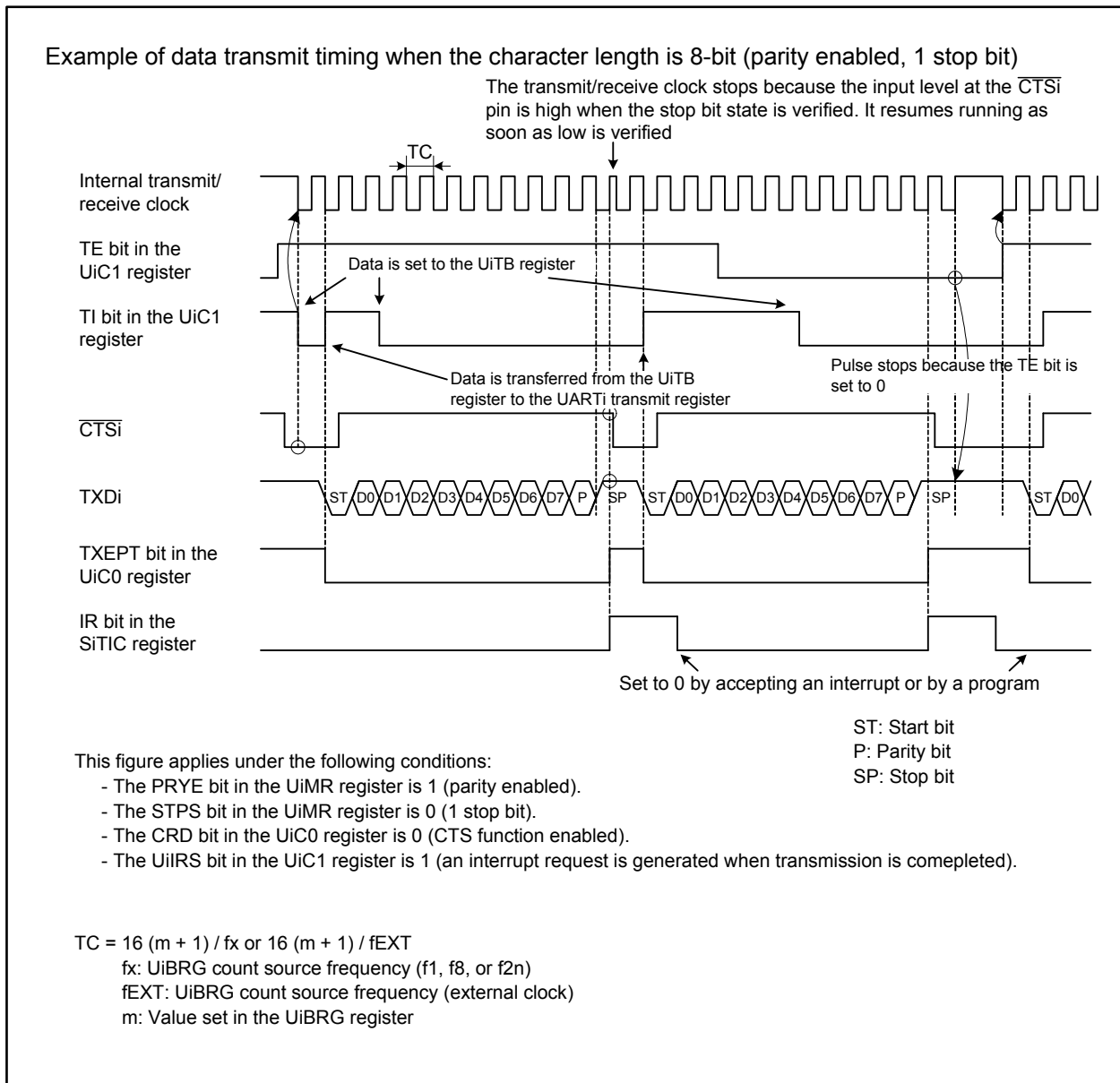


Figure 17.23 Transmit Operation in UART Mode (1/2) (i = 0 to 4)

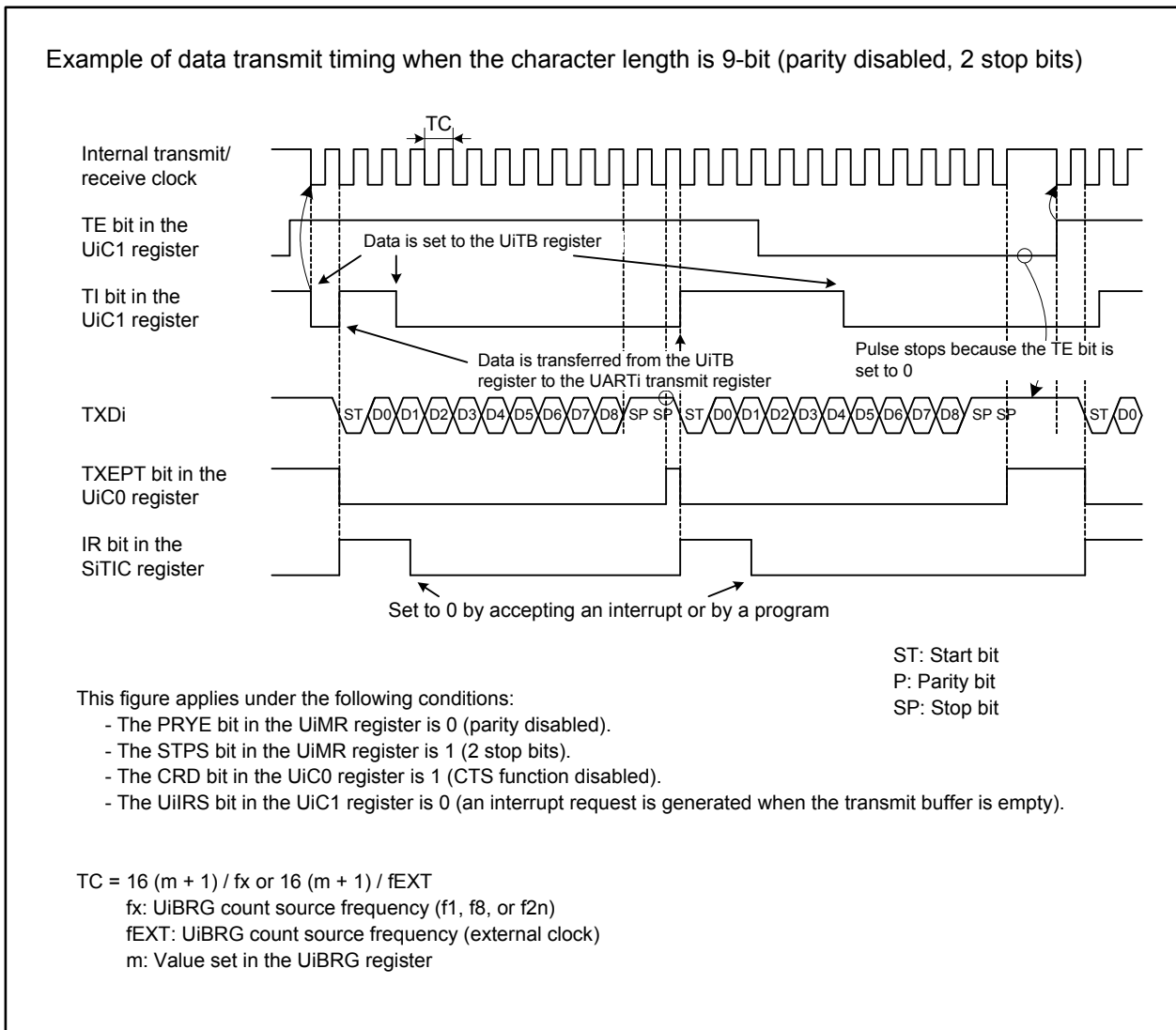


Figure 17.24 Transmit Operation in UART Mode (2/2) (i = 0 to 4)

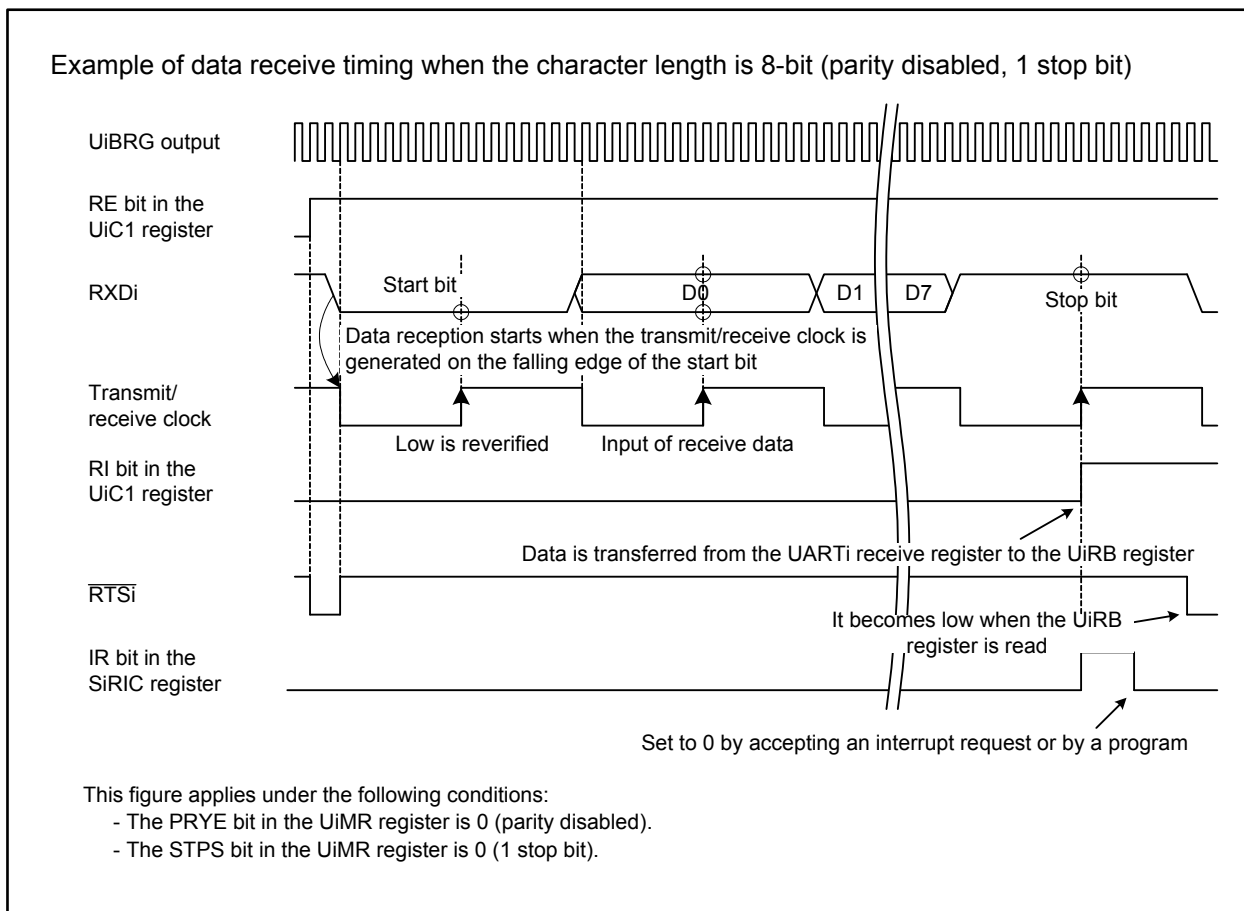


Figure 17.25 Receive Operation in UART Mode (i = 0 to 4)

### 17.2.1 Bit Rate

In UART mode, the bit rate is a clock frequency which is divided by a setting value of the UiBRG register and again divided by 16 (i = 0 to 4). Table 17.8 lists an example of bit rate setting.

Table 17.8 Bit Rate Setting

Bit Rate (bps)	Count Source of BRG	Peripheral Clock: 30 MHz		Peripheral Clock: 32 MHz	
		Setting value of BRG: n	Actual bit rate (bps)	Setting value of BRG: n	Actual bit rate (bps)
1200	f8	194 (C2h)	1202	207 (CHh)	1202
2400	f8	97 (61h)	2392	103 (67h)	2404
4800	f8	48 (30h)	4783	51 (33h)	4808
9600	f1	194 (C2h)	9615	207 (CFh)	9615
14400	f1	129 (81h)	14423	138 (8Ah)	14388
19200	f1	97 (61h)	19133	103 (67h)	19231
28800	f1	64 (40h)	28846	68 (44h)	28986
31250	f1	59 (3Bh)	31250	63 (3Fh)	31250
38400	f1	48 (30h)	38265	51 (33h)	38462
51200	f1	36 (24h)	50676	38 (26h)	51282

### 17.2.2 Reset Procedure on Transmit/Receive Error

When a transmit/receive error occurs in UART mode, follow the procedure below to perform a reset:

- A. Reset procedure for the UiRB register ( $i = 0$  to 4)
  - (1) Set the RE bit in the UiC1 register to 0 (reception disabled).
  - (2) Set the RE bit in the UiC1 register to 1 (reception enabled).
  
- B. Reset procedure for the UiTB register
  - (1) Set bits SMD2 to SMD0 in the UiMR register to 000b (serial interface disabled).
  - (2) Set again bits SMD2 to SMD0 to either of 001b, 101b, or 110b.
  - (3) Irrespective of its status, set the TE bit in the UiC1 register to 1 (transmission enabled).

### 17.2.3 LSB First and MSB First Selection

As shown in Figure 17.26, the bit order is selected by setting the UFORM bit in the UiC0 register ( $i = 0$  to 4). This function is available when the character length is 8-bit.

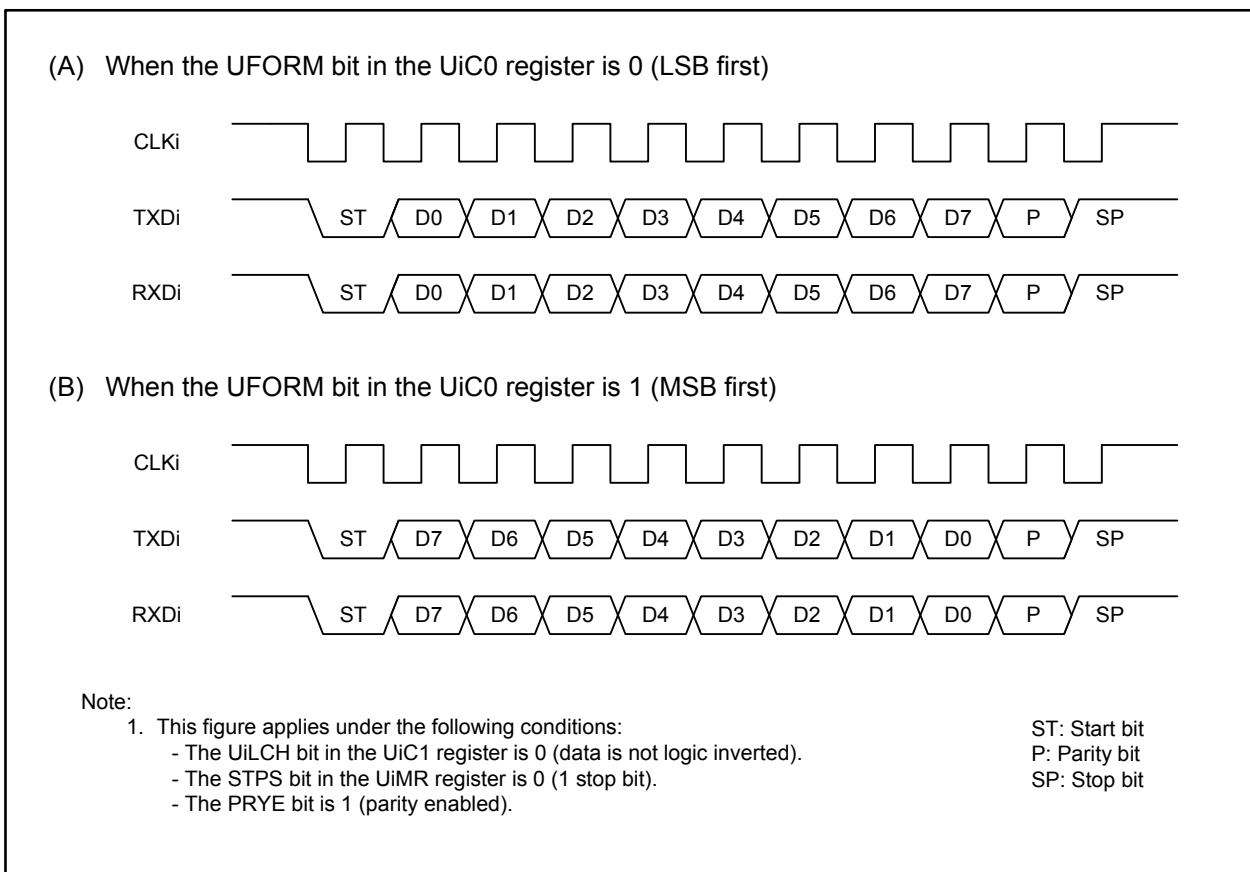


Figure 17.26 Bit Order ( $i = 0$  to 4)

### 17.2.4 Serial Data Logic Inversion

When the UiLCH bit in the UiC1 register is 1 (data is logic inverted), the logical value written in the UiTB register is inverted before being transmitted ( $i = 0$  to 2). The UiRB register is read as logic-inverted receive data. The parity bit is not inverted. Figure 17.27 shows the logic inversion of serial data.

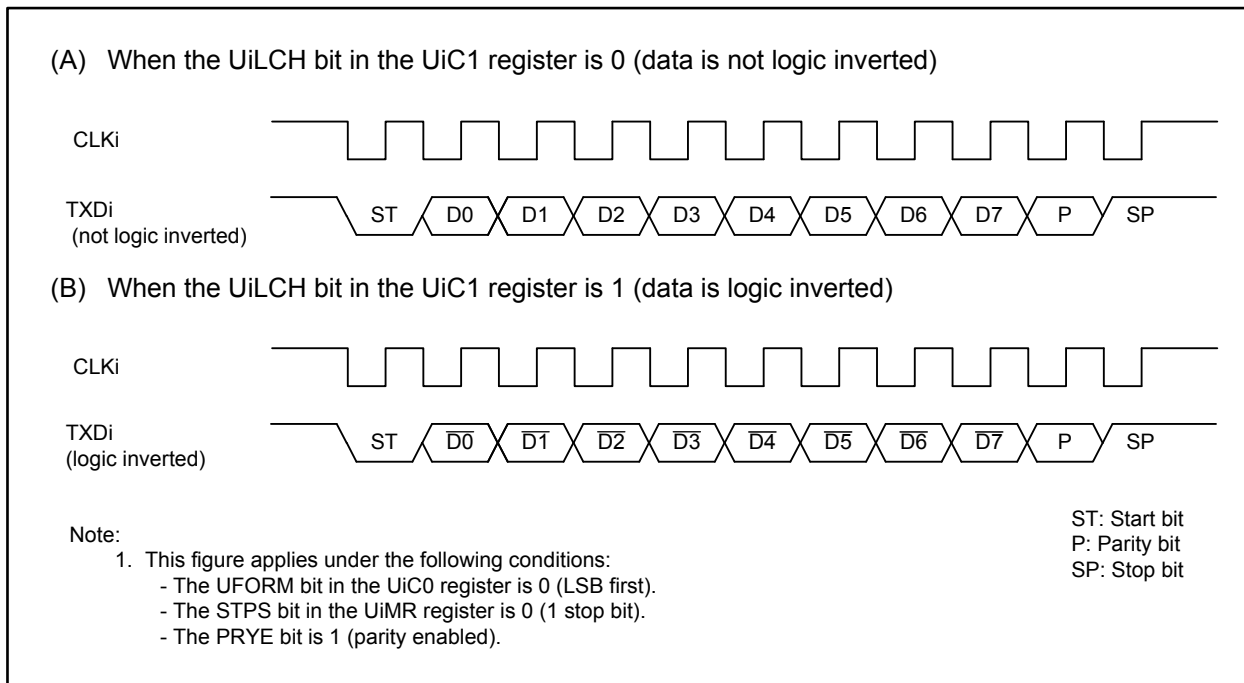


Figure 17.27 Serial Data Logic Inversion ( $i = 0$  to 2)



### 17.2.5 TXD and RXD I/O Polarity Inversion

The output level at the TXD pin and the input level at the RXD pin are inverted by this function. All I/O data levels, including the start bit, stop bit, and parity bit are inverted by setting the IOPOL bit in the UiMR register to 1 (inverted) ( $i = 0$  to 2). Figure 17.28 shows TXD and RXD I/O polarity inversion.

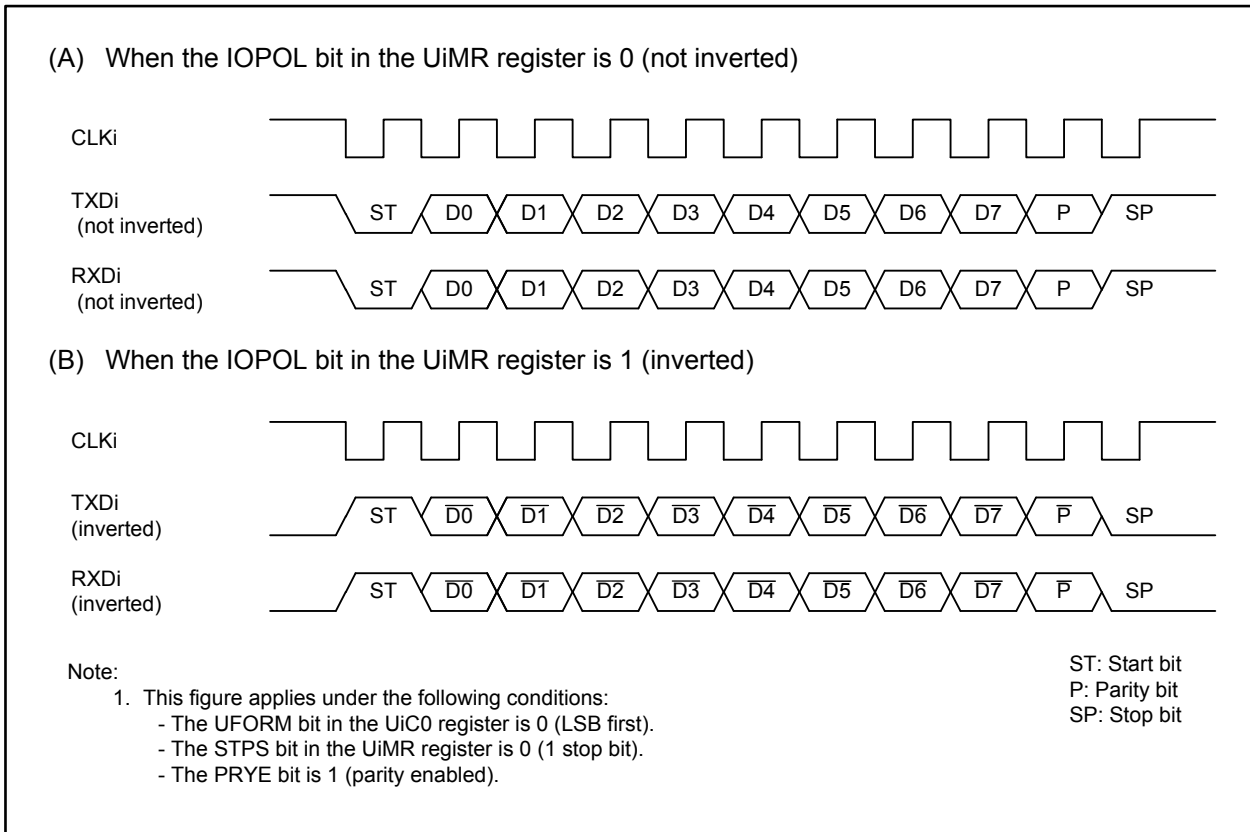


Figure 17.28 TXD and RXD I/O Polarity Inversion ( $i = 0$  to 2)

### 17.2.6 CTS/RTS Function

CTS function controls data transmission using the  $\overline{\text{CTS}}_i/\overline{\text{RTS}}_i$  pin ( $i = 0$  to 4). When an input level at the pin becomes low, data transmission starts. If the input level changes to high during transmit operation, transmission of the next data is stopped.

RTS function indicates receiver status using the  $\overline{\text{CTS}}_i/\overline{\text{RTS}}_i$  pin. When the MCU is ready to receive data, the output level at the pin becomes low. It becomes high on the first falling edge of the CLK<sub>i</sub> pin.

### 17.3 Special Mode 1 (I<sup>2</sup>C Mode)

This mode uses an I<sup>2</sup>C-typed interface for communication. Table 17.9 lists specifications of the I<sup>2</sup>C mode.

**Table 17.9 I<sup>2</sup>C Mode Specifications**

Item	Specification
Data format	8-bit character length
Transmit/receive clock	<p>In master mode</p> <ul style="list-style-type: none"> <li>The CKDIR bit in the UiMR register is 0 (internal clock) (i = 0 to 2):</li> </ul> $\frac{fx}{2(m+1)} \quad fx = f1, f8, f2n$ <p style="text-align: center;"><i>m</i>: UiBRG register setting value, 00h to FFh</p> <p>In slave mode</p> <ul style="list-style-type: none"> <li>The CKDIR bit is 1 (external clock): input to the SCLi pin</li> </ul>
Transmit start conditions	<p>The conditions for starting data transmission are as follows (1):</p> <ul style="list-style-type: none"> <li>The TE bit in the UiC1 register is 1 (transmission enabled)</li> <li>The TI bit in the UiC1 register is 0 (data held in the UiTB register)</li> </ul>
Receive start conditions	<p>The conditions for starting data reception are as follows (1):</p> <ul style="list-style-type: none"> <li>The RE bit in the UiC1 register is 1 (reception enabled)</li> <li>The TE bit in the UiC1 register is 1 (transmission enabled)</li> <li>The TI bit in the UiC1 register is 0 (data held in the UiTB register)</li> </ul>
Interrupt request generating timing	When any of the following is detected: start condition, stop condition, NACK (not-acknowledge), or ACK (acknowledge)
Error detection	<p>Overflow error (2)</p> <p>This error occurs when the eighth bit of the next data is received before the UiRB register is read</p>
Other functions	<ul style="list-style-type: none"> <li>Arbitration lost Update timing of the ABT bit in the UiRB register can be selected</li> <li>SDAi digital delay No digital delay or two to eight cycles of digital delay of UiBRG count source</li> <li>Clock phase setting Clock delayed or no clock delay</li> </ul>

**Notes:**

- When an external clock is selected, the conditions should be met while the external clock signal is held high.
- The UiRB register is undefined when an overrun error occurs. The IR bit in the SiRIC register does not change to 1 (interrupt requested).

Table 17.10 lists register settings in I<sup>2</sup>C mode, and Tables 17.11 and 17.12 list I<sup>2</sup>C mode functions. Figure 17.29 shows a block diagram of I<sup>2</sup>C mode, and Figure 17.30 shows timings for the transfer to the UiRB register and the interrupt (i = 0 to 2).

As shown in Tables 17.11 and 17.12, UARTi enters this mode when bits SMD2 to SMD0 in the UiMR register are set to 010b, and the IICM bit in the UiSMR register is set to 1 (i = 0 to 2). Since a transmit signal at the SDAi pin is output via the delay circuit, it changes after the SCLi pin is stably held low.



**Table 17.10 Register Settings in I<sup>2</sup>C Mode (i = 0 to 2)**

Register	Bits	Function	
		Master	Slave
UiMR	IOPOL	Set the bit to 0	
	CKDIR	Set the bit to 0	Set the bit to 1
	SMD2 to SMD0	Set the bit to 010b	
UiC0	7 to 4	Set the bits to 1011b	
	TXEPT	Transmit register empty flag	
	2	Set the bit to 0	
	CLK1 and CLK0	Select a count source for the UiBRG register	Disabled
UiC1	7 to 5	Set the bits to 000b	
	UIIRS	Set the bit to 1	
	RI	Receive complete flag	
	RE	Set the bit to 1 to enable data reception	
	TI	Transmit buffer empty flag	
	TE	Set the bit to 1 to enable data transmission/reception	
UiSMR	7 to 3	Set the bits to 00000b	
	BBS	Bus busy flag	
	ABC	Select an arbitration lost detection timing	Disabled
	IICM	Set the bit to 1	
UiSMR2	7	Set the bit to 0	
	SDHI	Set the bit to 1 to disable the SDA output	
	SWC2	Set the bit to 1 to hold the SCL output at a forcible low	
	STC	Set the bit to 0	Set the bit to 1 to reset UARTi by detecting the start condition
	ALS	Set the bit to 1 to stop the output at the SDAi pin to detect an arbitration lost	Set the bit to 0
	SWC	Set the bit to 1 to hold a low output at the SCLi pin after receiving the eighth bit of the clock	
	CSC	Set the bit to 1 to enable clock synchronization	Set the bit to 0
	IICM2	Refer to Tables 17.11 and 17.12	
UiSMR3	DL2 to DL0	Set the digital delay value of SDAi	
	4 to 2	Set the bit to 000b	
	CKPH	Refer to Tables 17.11 and 17.12	
	SSE	Set the bit to 0	
UiSMR4	SWC9	Set the bit to 0	Set the bit to 1 to hold a low output at the SCLi pin after receiving the ninth bit of the clock
	SCLHI	Set the bit to 1 to stop the SCL output to detect stop condition	Set the bit to 0
	ACKC	Set the bit to 1 for ACK data output	
	ACKD	Select ACK or NACK	
	STSPSEL	Set the bit to 1 when any condition is output	Set the bit to 0
	STPREQ	Set the bit to 1 to generate a stop condition	Set the bit to 0
	RSTAREQ	Set the bit to 1 to generate a restart condition	Set the bit to 0
	STAREQ	Set the bit to 1 to generate a start condition	Set the bit to 0
UiBRG	7 to 0	Set the bit rate	Disabled
UiTB	8	Set the bit to 1 when transmitting. Set the bit to the value of the ACK bit when receiving	
	7 to 0	Set the data to be transmitted when transmitting. Set the register to FFh when receiving	
UiRB	OER	Overrun error flag	
	ABT	Arbitration lost detection flag	Disabled
	8	D0 is loaded immediately after a receive interrupt occurs. ACK or NACK is loaded after a transmit interrupt occurs	
	7 to 0	D7 to D1 are read immediately after a receive interrupt occurs. D7 to D0 are read after a transmit interrupt occurs	

**Table 17.11 I<sup>2</sup>C Mode Functions (i = 0 to 2) (1/2)**

Function	Synchronous Serial Interface Mode (SMD2 to SMD0 are 001b, IICM is 0)	I <sup>2</sup> C Mode (SMD2 to SMD0 are 010b, IICM is 1)			
		IICM2 is 0 (ACK/NACK interrupt)		IICM2 is 1 (Transmit/receive interrupt)	
		CKPH is 0 (No clock delay)	CKPH is 1 (Clock delayed)	CKPH is 0 (No clock delay)	CKPH is 1 (Clock delayed)
Source of software interrupt numbers 39 to 41 <sup>(1)</sup> (refer to Figure 17.30)	—	Start condition or stop condition detection (refer to Table 17.13)			
Source of software interrupt numbers 17, 19, and 33 <sup>(1)</sup> (refer to Figure 17.30)	UARTi transmission: Transmission started or completed (selected using the UiIRS register)	NACK detection: Rising edge of the ninth bit of SCLi	UARTi transmission: Rising edge of the ninth bit of SCLi	UARTi transmission: Falling edge of the ninth bit of SCLi	
Source of software interrupt numbers 18, 20, and 34 <sup>(1)</sup> (refer to Figure 17.30)	UARTi reception: Receiving at eighth bit CKPOL is 0 (rising edge) CKPOL is 1 (falling edge)	ACK detection: Rising edge of the ninth bit of SCLi	UARTi reception: Falling edge of the eighth bit of SCLi		
Data transfer timing from the UART receive register to the UiRB register	CKPOL is 0 (rising edge) CKPOL is 1 (falling edge)	Rising edge of the ninth bit of SCLi	Falling edge of the eighth bit of SCLi	Falling edge of the eighth bit and rising edge of the ninth bit of SCLi	
UARTi transmit output delay	No delay	Delayed			
Pins P5_7, P4_7, and P7_0	TXDi output	SDAi I/O			
Pins P5_6, P4_6, and P7_4	RXDi input	SCLi I/O			
Pins P5_5, P4_5, and P7_5	Select CLKi input or output	— (Not used in I <sup>2</sup> C mode)			

**Note:**

1. Steps to change an interrupt source are as follows:
  - (1) Disable the interrupt of the corresponding software interrupt number.
  - (2) Change the source of interrupt.
  - (3) Set the IR bit of the corresponding software interrupt number to 0 (no interrupt requested).
  - (4) Set bits ILVL2 to ILVL0 of the corresponding software interrupt number.

**Table 17.12 I<sup>2</sup>C Mode Functions (i = 0 to 2) (2/2)**

Function	Synchronous Serial Interface Mode (SMD2 to SMD0 are 001b, IICM is 0)	I <sup>2</sup> C Mode (SMD2 to SMD0 are 010b, IICM is 1)			
		IICM2 is 0 (ACK/NACK interrupt)		IICM2 is 1 (Transmit/receive interrupt)	
		CKPH is 0 (No clock delay)	CKPH is 1 (Clock delayed)	CKPH is 0 (No clock delay)	CKPH is 1 (Clock delayed)
Read level at pins RXDi and SCLi	Readable irrespective of the port direction bit				
Default output value at the SDAi pin	—	High (Value set in the port Pi register if the I/O port is selected by output function select registers (i = 0 to 7))			
SCLi default and end values	—	High	Low	High	Low
DMA source (refer to Figure 17.30)	UARTi reception	ACK detection		UARTi reception: Falling edge of the eighth bit of SCLi	
Store received data	The first to eighth bits of received data are stored into bits 0 to 7 in the UiRB register	The first to eighth bits of received data are stored into bits 7 to 0 in the UiRB register		The first to seventh bits of received data are stored into bits 6 to 0 in the UiRB register and the eighth bit is stored into bit 8	Same as on the left column on the first data storing. <sup>(1)</sup> The first to eighth bits of received data are stored into 7 to 0 bits in the UiRB register and the ninth bit is stored into bit 8 on the second data storing <sup>(2)</sup>
Read received data	The UiRB register status is read as it is			Bits 6 to 0 in the UiRB register are read as bits 7 to 1 and bit 8 is read as bit 0	Same as on the left column on the first read. <sup>(1)</sup> The UiRB register status is read as it is on the second read <sup>(2)</sup>

**Notes:**

1. The first data transfer to the UiRB register starts on the rising edge of the eighth bit of SCLi.
2. The second data transfer to the UiRB register starts on the rising edge of the ninth bit of SCLi.

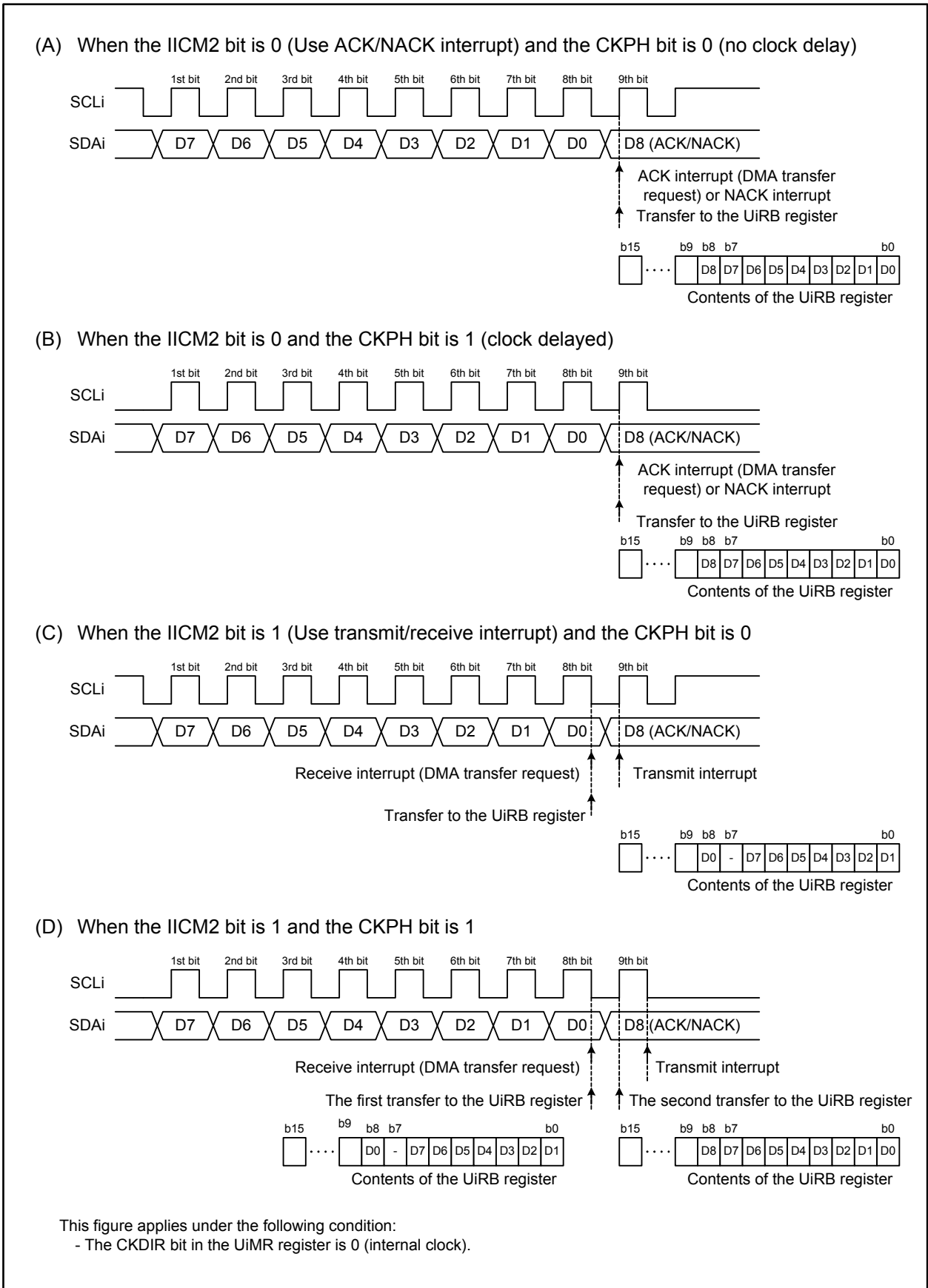


Figure 17.30 Timings for the Transfer and Interrupt to the UIRB Register (i = 0 to 2)

### 17.3.1 Start Condition and Stop Condition Detection

The start condition and stop condition are detected by their respective detectors.

The start condition detection interrupt request is generated by a high-to-low transition at the SDA<sub>i</sub> pin while the SCL<sub>i</sub> pin is held high ( $i = 0$  to 2). The stop condition detection interrupt request is generated by a low-to-high transition at the SDA<sub>i</sub> pin while the SCL<sub>i</sub> pin is held high.

The start condition detection interrupt shares interrupt control registers and vectors with the stop condition detection interrupt. The BBS bit in the UiSMR register determines which interrupt is requested.

To detect a start condition or stop condition, both set-up and hold times require at least six cycles of the peripheral clock ( $f_1$ ) as shown in Figure 17.31. To meet the condition for the Fast-mode specification,  $f_1$  must be at least 10 MHz.

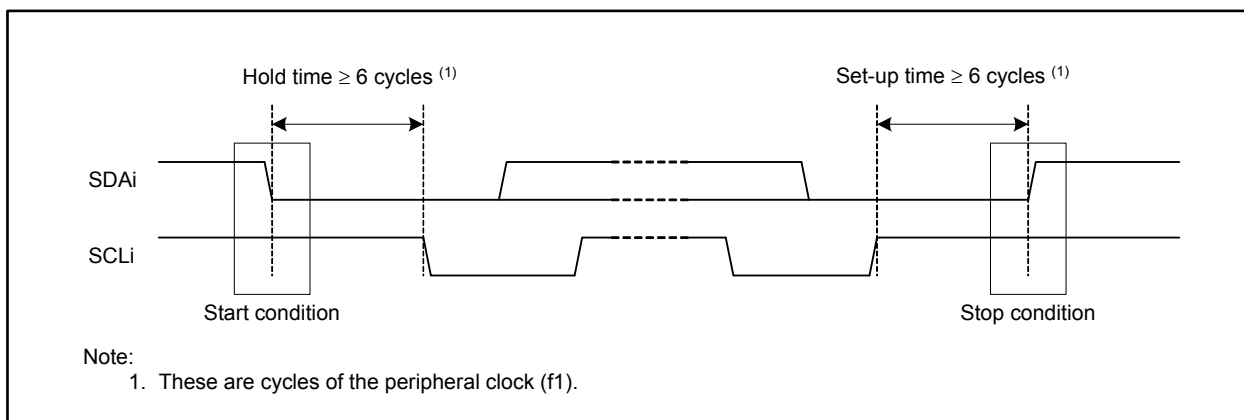


Figure 17.31 Start Condition and Stop Condition Detection Timing ( $i = 0$  to 2)

### 17.3.2 Start Condition and Stop Condition Generation

The start condition, restart condition, and stop condition are generated by bits STAREQ, RSTAREQ, and STPREQ in the UiSMR4 register, respectively ( $i = 0$  to 2). To output a start condition, set the STSPSEL bit in the UiSMR4 register to 1 (select start condition/stop condition generation circuit) after setting the STAREQ bit to 1 (start). To output a restart condition or stop condition, set the STSPSEL bit to 1 after setting RSTAREQ bit or STPREQ bit to 1, respectively.

Table 17.13 and Figure 17.32 show the functions of the STSPSEL bit.

Table 17.13 STSPSEL Bit Functions

Function	STSPSEL is 0	STSPSEL is 1
Start condition and stop condition generation	Output is provided by the program with port (no auto generation by hardware)	Start condition or stop condition is output according to the STAREQ, RSTAREQ, or STPREQ bit
Start condition and stop condition interrupt request generating timing	When start condition or stop condition is detected	When start condition or stop condition generation is completed





### 17.3.4 SCL Control and Clock Synchronization

Data transmission/reception in I<sup>2</sup>C mode uses the transmit/receive clock as shown in Figure 17.30. The clock speed increase makes it difficult to secure the required time for ACK generation and data transmit procedure. I<sup>2</sup>C mode supports a function of wait-state insertion to secure this required time and a function of clock synchronization with a wait-state inserted by other devices.

The SWC bit in the UiSMR2 register is used to insert a wait-state for ACK generation ( $i = 0$  to 2). When the SWC bit is 1 (hold the SCLi pin low after the eighth bit is received), the SCLi pin is held low on the falling edge of the eighth bit of the SCLi. When the SWC bit is 0 (no wait-state/wait-state cleared), the SCLi line is released.

When the SWC2 bit in the UiSMR2 register is 1 (hold the SCLi pin low), the SCLi pin is forced low even during transmission or reception. When the SWC2 bit is 0 (output the transmit/receive clock at the SCLi pin), the SCLi line is released to output the transmit/receive clock.

The SWC9 bit in the UiSMR4 register is used to insert a wait-state for checking received acknowledge bits. While the CKPH bit in the UiSMR3 register is 1 (clock delayed), when the SWC9 bit is set to 1 (hold the SCLi pin low after the ninth bit is received), the SCLi pin is held low on the falling edge of the ninth bit of the SCLi. When the SWC9 bit is set to 0 (no wait-state/wait-state cleared), the SCLi line is released.

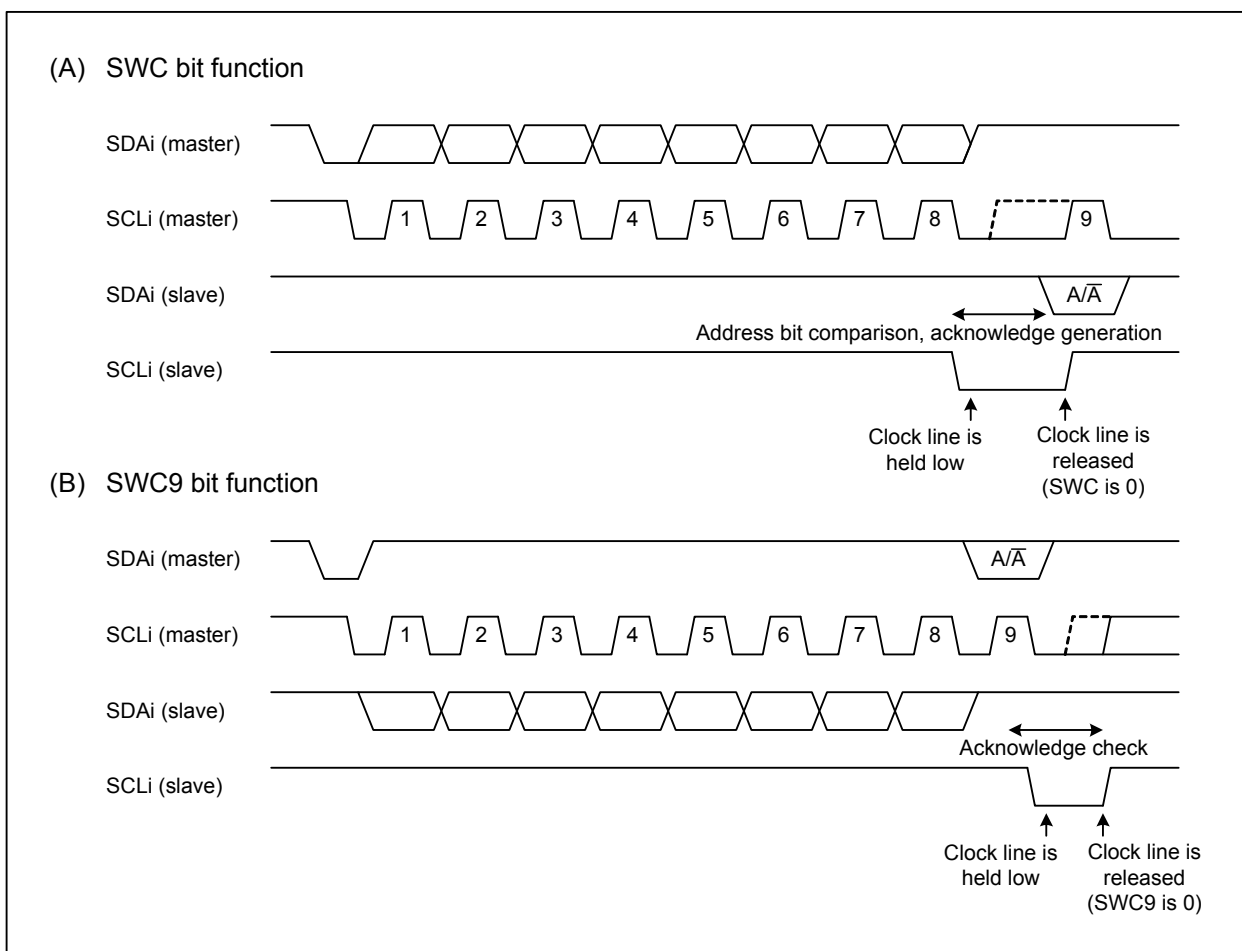


Figure 17.33 Wait-state Insertion Using the SWC or SWC9 Bit ( $i = 0$  to 2)

The CSC bit in the UiSMR2 register is used to synchronize an internally generated clock with the clock applied to the SCLi pin. For example, if a wait-state is inserted from another device, the two clocks are not synchronized. While the CSC bit is 1 (clock synchronization enabled) and the internal clock is held high, when a high at the SCLi pin changes to low, the internal clock becomes low in order to reload the value of the UiBRG register and to resume counting. While the SCLi pin is held low, when the internal clock changes from low to high, the count is stopped until the SCLi pin becomes high. That is, the UARTi transmit/receive clock is the logical AND of the internal clock and the SCLi. The synchronized period starts from one clock prior to the first synchronized clock and ends when the ninth clock is completed. The CSC bit can be set to 1 only when the CKDIR bit in the UiMR register is 0 (internal clock).

The SCLHI bit in the UiSMR4 register is used to leave the SCLi pin open when another master generates a stop condition while the master is in transmit/receive operation. If the SCLHI bit is set to 1 (stop SCLi output), the SCLi pin is open (the pin is high-impedance) when a stop condition is detected and the clock output is stopped.

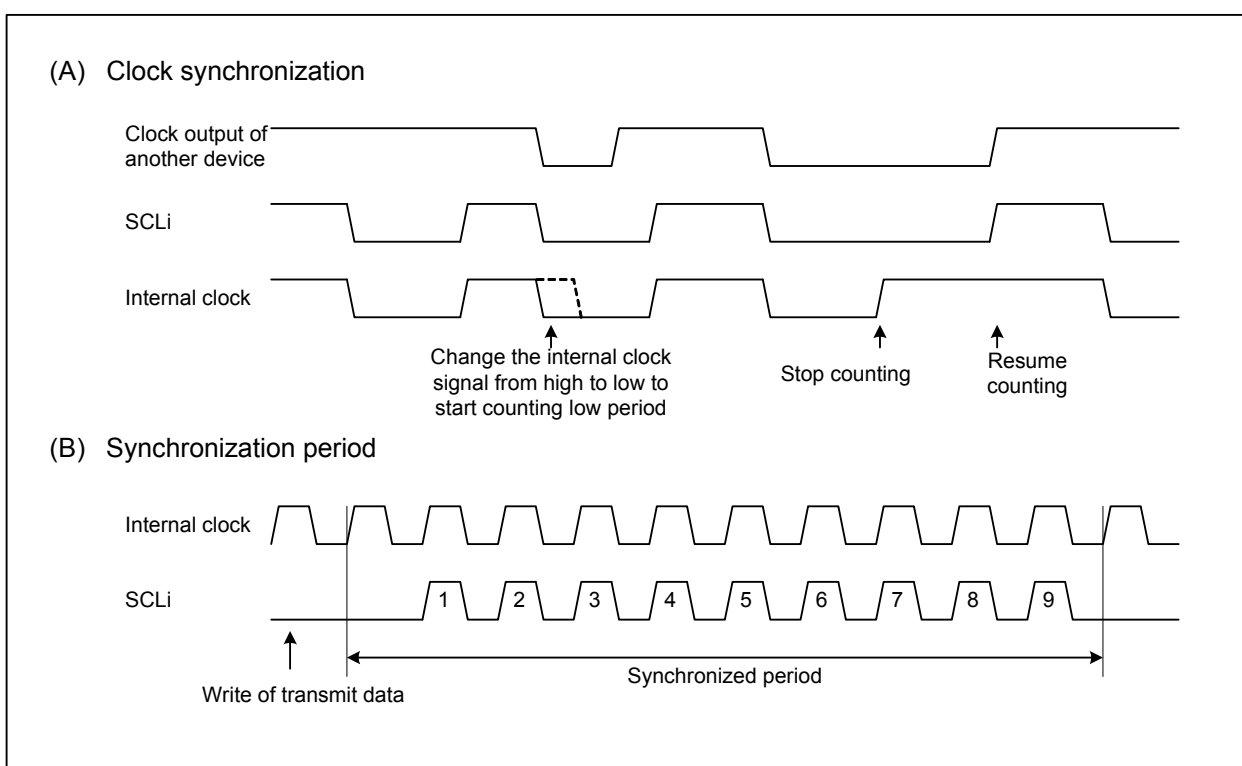


Figure 17.34 Clock Synchronization (i = 0 to 2)

### 17.3.5 SDA Output

Values set to bits 8 to 0 (D8 to D0) in the UiTB register are output starting from D7 to D0, and lastly D8, which is a bit for the acknowledge signal ( $i = 0$  to 2). When transmitting, D8 should be set to 1 to free the bus. When receiving, D8 should be set to ACK or NACK.

Bits DL2 to DL0 in the UiSMR3 register set a delay time of the SDA<sub>i</sub> on the falling edge of the SCL<sub>i</sub>. Based on the UiBRG count source, the delay time can be selected from zero cycles (no delay) or two to eight cycles.

The SDA<sub>i</sub> pin can be high-impedance at any given time once the SDHI bit in the UiSMR2 register is set to 1 (stop the output). Output at the SDA<sub>i</sub> pin is low if an I/O port is selected for the SDA<sub>i</sub> and the pin is specified as the output port after selecting I<sup>2</sup>C mode. In this case, if the SDHI bit is 1, the SDA<sub>i</sub> pin becomes high-impedance.

When the SDHI bit is rewritten while the SCL<sub>i</sub> pin is held high, a start condition or stop condition is generated. When it is rewritten immediately before the rising edge of SCL<sub>i</sub>, arbitration lost may be accidentally detected. Therefore, the SDHI bit should be rewritten so the SDA<sub>i</sub> pin level changes while the SCL<sub>i</sub> pin is low.

### 17.3.6 SDA Input

When the IICM2 bit in the UiSMR2 register is 0 (use ACK/NACK interrupt), the first 8 bits of received data (D7 to D0) are stored into bits 7 to 0 in the UiRB register and the ninth bit (ACK/NACK) is stored into bit 8 ( $i = 0$  to 2).

When the IICM2 bit is 1, the first 7 bits of received data (D7 to D1) are stored into bits 6 to 0 in the UiRB register and eighth bit (D0) is stored into bit 8.

If the IICM2 bit is 1 and the CKPH bit in the UiSMR3 register is 1 (clock delayed), the same data that is set when the IICM2 bit is 0 can be read. To read this data, read the UiRB register after data in the ninth bit is latched on the rising edge of the SCL<sub>i</sub>.

### 17.3.7 Acknowledge

When data is to be received in master mode, ACK is output after 8 bits are received by setting the UiTB register to 00FFh as dummy data. When the STSPSEL bit in the UiSMR4 register is 0 (select serial I/O circuit) and the ACKC bit is 1 (ACK data output), the value of the ACKD bit is output at the SDA<sub>i</sub> pin ( $i = 0$  to 2).

If the IICM2 bit is 0, a NACK interrupt request is generated when the SDA<sub>i</sub> pin is high on the rising edge of the ninth bit of the SCL<sub>i</sub>. An ACK interrupt request is generated when the SDA<sub>i</sub> pin is low.

When the DMA request source is "UART<sub>i</sub> receive interrupt request or ACK interrupt request", the DMA transfer starts when an ACK is detected.

### 17.3.8 Initialization of Transmit/Receive Operation Reset

When the CKDIR bit in the UiMR register is 1 (external clock), the STC bit in the UiSMR2 register is 1 (initialize the circuit), and a start condition is detected, the following three operations are performed ( $i = 0$  to 2):

- The transmit register is reset and the UiTB register value is transferred to the transmit register. New data transmission starts on the falling edge of the first bit of the next SCL<sub>i</sub> as transmit clock. The transmit register value before the reset is output at the SDA<sub>i</sub> pin in the period from the falling edge of the SCL<sub>i</sub> until the first data output.
- The receive register is reset and the new data reception starts on the falling edge of the first bit of the next SCL<sub>i</sub>.
- The SWC bit in the UiSMR2 register becomes 1 (hold the SCL<sub>i</sub> pin low after the eighth bit is received).

The TI bit in the UiC1 register does not change when using this function to start the UART<sub>i</sub> transmission/reception.

## 17.4 Special Mode 2

Special mode 2 enables serial communication between one or multiple masters and multiple slaves. The  $\overline{SS}_i$  input pin controls serial bus communication ( $i = 0$  to  $2$ ). Table 17.14 lists specifications of special mode 2.

**Table 17.14 Special Mode 2 Specifications**

Item	Specification
Data format	8-bit character length
Transmit/receive clock	<ul style="list-style-type: none"> <li>The CKDIR bit in the UiMR register is set to 0 (internal clock) (<math>i = 0</math> to <math>2</math>):  <math display="block">\frac{f_x}{2(m+1)} \quad f_x = f_1, f_8, f_{2n} \quad m: \text{UiBRG register setting value, 00h to FFh}</math> </li> <li>The CKDIR bit is set to 1 (external clock): input to the CLK<sub>i</sub> pin</li> </ul>
Transmit/receive control	SS function
Transmit start conditions	<p>The conditions for starting data transmission are as follows (1):</p> <ul style="list-style-type: none"> <li>The TE bit in the UiC1 register is 1 (transmission enabled)</li> <li>The TI bit in the UiC1 register is 0 (data held in the UiTB register)</li> </ul>
Receive start conditions	<p>The conditions for starting data reception are as follows (1):</p> <ul style="list-style-type: none"> <li>The RE bit in the UiC register is 1 (reception enabled)</li> <li>The TE bit in the UiC1 register is 1 (transmission enabled)</li> <li>The TI bit in the UiC1 register is 0 (data held in the UiTB register)</li> </ul>
Interrupt request generating timing	<p>In transmit interrupt, one of the following conditions can be selected by setting the UiIRS bit in registers U0C1 to U2C1:</p> <ul style="list-style-type: none"> <li>The UiIRS bit is 0 (transmit buffer is empty): when data is transferred from the UiTB register to the UAR<sub>Ti</sub> transmit register (when the transmission has started)</li> <li>The UiIRS bit is 1 (transmission is completed): when data transmission from the UAR<sub>Ti</sub> transmit register is completed</li> </ul> <p>In receive interrupt,</p> <ul style="list-style-type: none"> <li>When data is transferred from the UAR<sub>Ti</sub> receive register to the UiRB register (when the reception is completed)</li> </ul>
Error detection	<p>Overflow error (2)</p> <p>This error occurs when the seventh bit of the next data has been received before reading the UiRB register</p>
Other functions	<ul style="list-style-type: none"> <li>CLK polarity Rising or falling edge of the transmit/receive clock for transfer data input and output</li> <li>Bit order selection LSB first or MSB first</li> <li>Continuous receive mode Data reception is enabled by a read access to the UiRB register</li> <li>Serial data logic inversion This function logically inverts transmit/receive data</li> <li>Clock phase selection One of four combinations of transmit/receive clock polarity and phases</li> <li><math>\overline{SS}_i</math> input pin function Output pin can be high-impedance when the <math>\overline{SS}_i</math> pin is high</li> </ul>

Notes:

- When selecting an external clock, the following preconditions should be met:
  - The CLK<sub>i</sub> pin is held high when the CKPOL bit in the UiC0 register is 0 (transmit data output on the falling edge of the transmit/receive clock and receive data input on the rising edge).
  - The CLK<sub>i</sub> pin is held low when the CKPOL bit is 1 (transmit data output on the rising edge of the transmit/receive clock and receive data input on the falling edge).
- The UiRB register is undefined when an overrun error occurs. The IR bit in the SiRIC register does not change to 1 (interrupt requested).

Table 17.15 lists register settings in special mode 2.

**Table 17.15 Register Settings in Special Mode 2 (i = 0 to 2)**

Register	Bits	Function
UiMR	7 to 4	Set the bits to 0000b
	CKDIR	Set the bit to 0 in master mode and set it to 1 in slave mode
	SMD2 to SMD0	Set the bits to 001b
UiC0	UFORM	Select either LSB first or MSB first
	CKPOL	Clock phase can be set by the combination of bits CKPOL and CKPH in the UiSMR3 register
	NCH	Select an output mode for the TXDi pin
	CRD	Set the bit to 1
	TXEPT	Transmit register empty flag
	2	Set the bit to 0
	CLK1 and CLK0	Select a count source for the UiBRG register
UiC1	7 and 6	Set the bits to 00b
	UiRRM	Set the bit to 1 to use continuous receive mode
	UiIRS	Select a source for UARTi transmit interrupt
	RI	Receive complete flag
	RE	Set the bit to 1 to enable data reception
	TI	Transmit buffer empty flag
	TE	Set the bit to 1 to enable data transmission/reception
UiSMR	7 to 0	Set the bits to 00h
UiSMR2	7 to 0	Set the bits to 00h
UiSMR3	7 to 5	Set the bits to 000b
	ERR	Mode fault flag
	NODC	Set the bit to 0
	DINC	Set to 0 in master mode and set to 1 in slave mode
	CKPH	Clock phase can be set by the combination of bits CKPH and CKPOL in the UiC0 register
	SSE	Set the bit to 1
UiSMR4	7 to 0	Set the bits to 00h
UiBRG	7 to 0	Set the bit rate
UiTB	7 to 0	Set the data to be transmitted
UiRB	OER	Overrun error flag
	7 to 0	Received data is read

### 17.4.1 $\overline{SS}_i$ Input Pin Function (i = 0 to 2)

Special mode 2 is selected by setting the SSE bit in the UiSMR3 register to 1 (SS function enabled). The  $\overline{CTS}_i/\overline{RTS}_i/\overline{SS}_i$  pin functions as  $\overline{SS}_i$  input.

The DINC bit in the UiSMR3 register determines which MCU performs as a master or slave.

When multiple MCUs perform as masters (multi-master system), the  $\overline{SS}_i$  pin setting determines which master MCU is active and when.

#### 17.4.1.1 SS Function in Slave Mode

When the DINC bit is 1 (slave mode) while input at the  $\overline{SS}_i$  pin is high, the STXD<sub>i</sub> pin becomes high-impedance and the clock input at the CLK<sub>i</sub> pin is ignored. When input at the  $\overline{SS}_i$  pin is low, the clock input is valid and serial data is output from the STXD<sub>i</sub> pin to enable serial communication.

#### 17.4.1.2 SS Function in Master Mode

When the DINC bit is 0 (master mode) while input at the  $\overline{SS}_i$  pin is high, which means there is the only one master MCU or no other master MCU is active, the MCU as master starts communication. The master provides the transmit/receive clock output at the CLK<sub>i</sub> pin. When input at the  $\overline{SS}_i$  pin is low, which means that there are more masters, pins TXD<sub>i</sub> and CLK<sub>i</sub> become high-impedance. This error is called a mode fault. It can be verified using the ERR bit in the UiSMR3 register. The ongoing data transmission/reception does not stop even if a mode fault occurs. To stop transmission/reception, bits SMD2 to SMD0 in the UiMR register should be set to 000b (serial interface disabled).

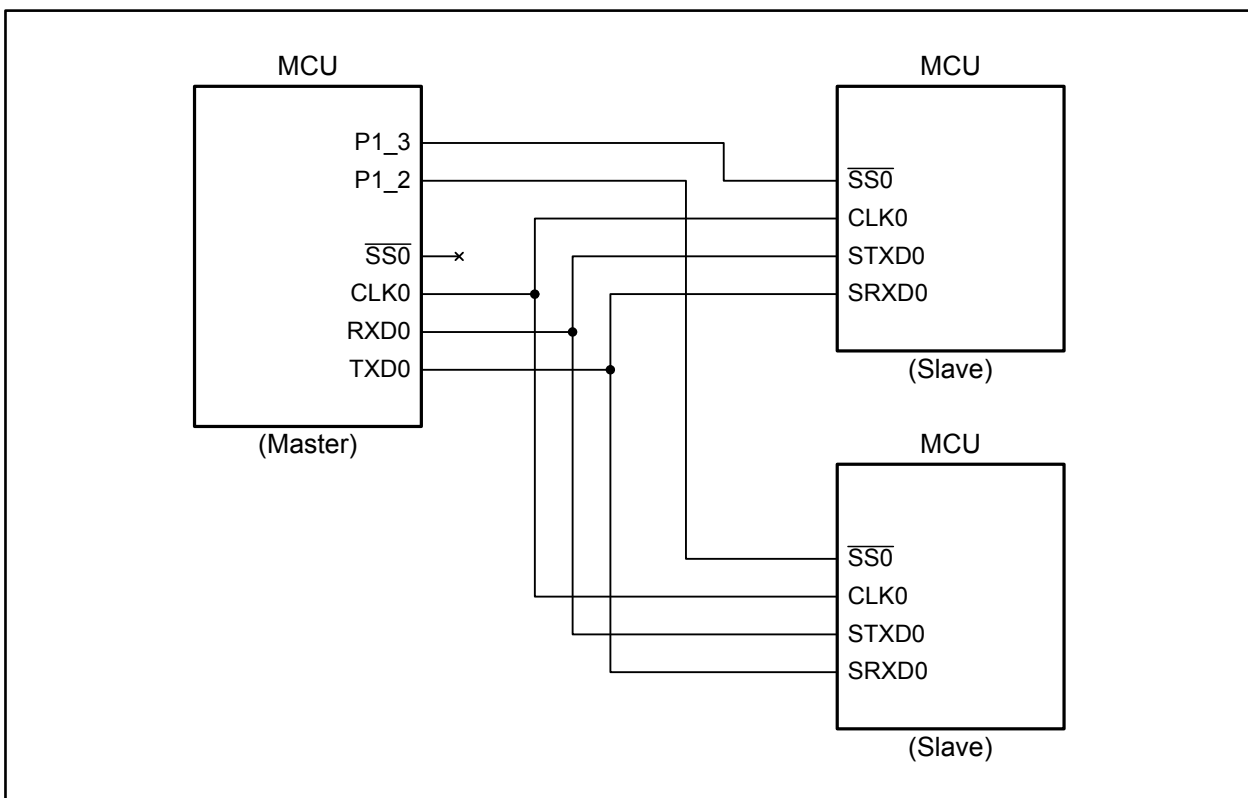


Figure 17.35 Serial Bus Communication Control with the  $\overline{SS}_i$  Pin

## 17.4.2 Clock Phase Setting

The CKPH bit in the UiSMR3 register and the CKPOL bit in the UiC0 register select one of four combinations of transmit/receive clock polarity and serial clock phase ( $i = 0$  to 2).

The transmit/receive clock phase and polarity should be identical for the master device and the communicating slave device.

### 17.4.2.1 Transmit/Receive Timing in Master Mode

When the DINC bit is 0 (master mode), the CKDIR bit in the UiMR register should be set to 0 (internal clock) to generate the clock. Figure 17.36 shows transmit/receive timing of each clock phase.

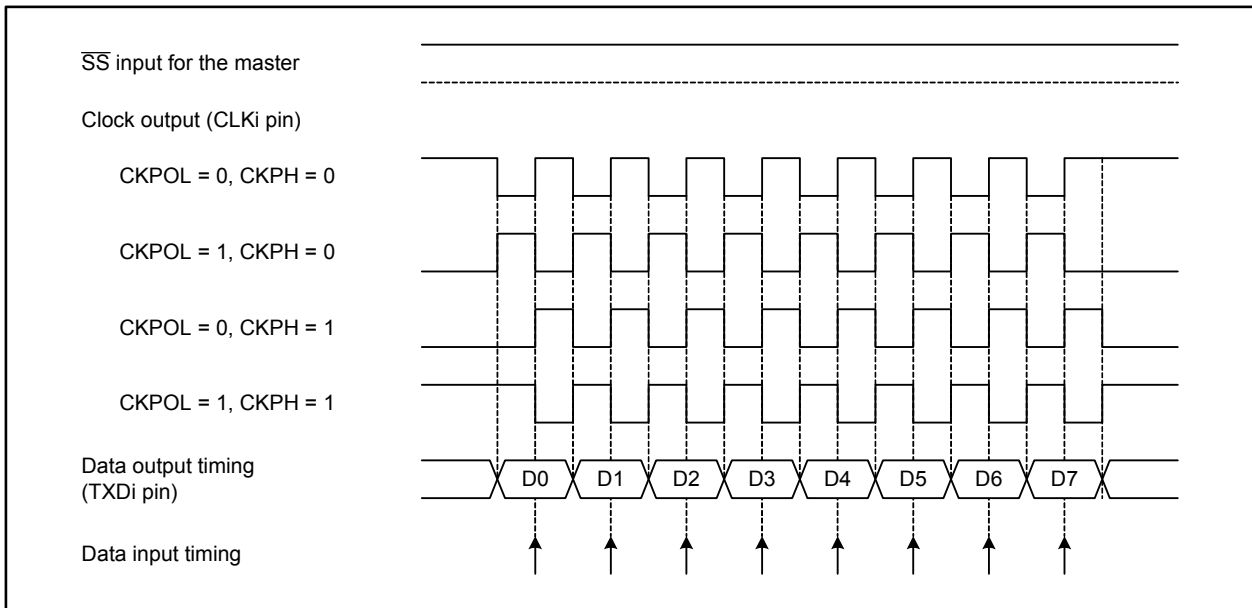


Figure 17.36 Transmit/Receive Timing in Master Mode



### 17.4.2.2 Transmit/Receive Timing in Slave Mode

When the DINC bit is 1 (slave mode), the CKDIR bit in the UIMR register should be set to 1 (external clock).

When the CKPH bit is 0 (no clock delay) while input at the  $\overline{SSi}$  pin is high, the STXD<sub>i</sub> pin becomes high-impedance. When input at the  $\overline{SSi}$  pin is low, the conditions for data transmission are all met, but output is undefined. Then the data transmission/reception starts synchronizing with the clock. Figure 17.37 shows the transmit/receive timing.

When the CKPH bit is 1 (clock delayed) while input at the  $\overline{SSi}$  pin is high, the STXD<sub>i</sub> pin becomes high-impedance. When input at the  $\overline{SSi}$  pin is low, the first data is output. Then the data transmission starts synchronizing with the clock. Figure 17.38 shows the transmit/receive timing.

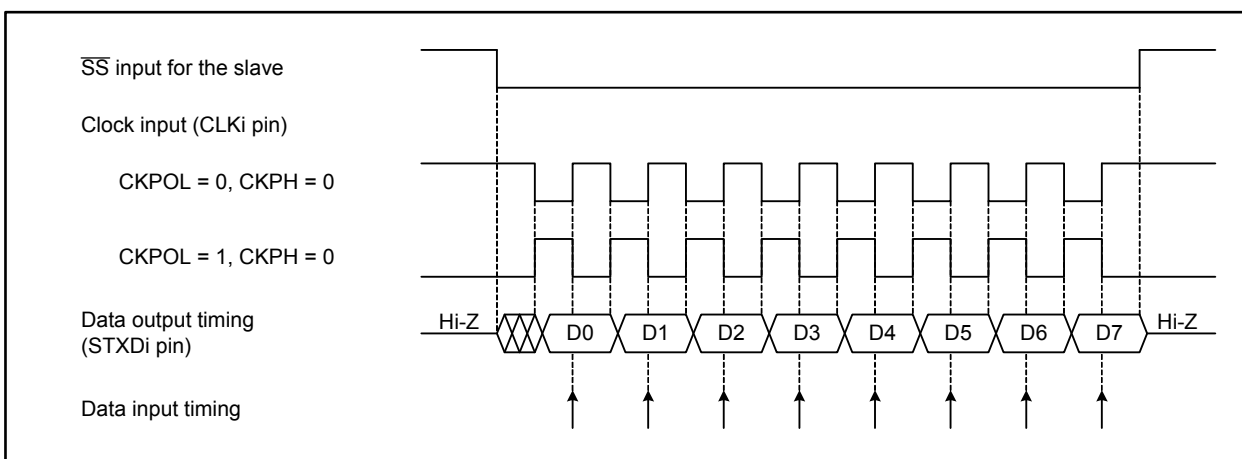


Figure 17.37 Transmit/Receive Timing in Slave Mode (CKPH = 0)

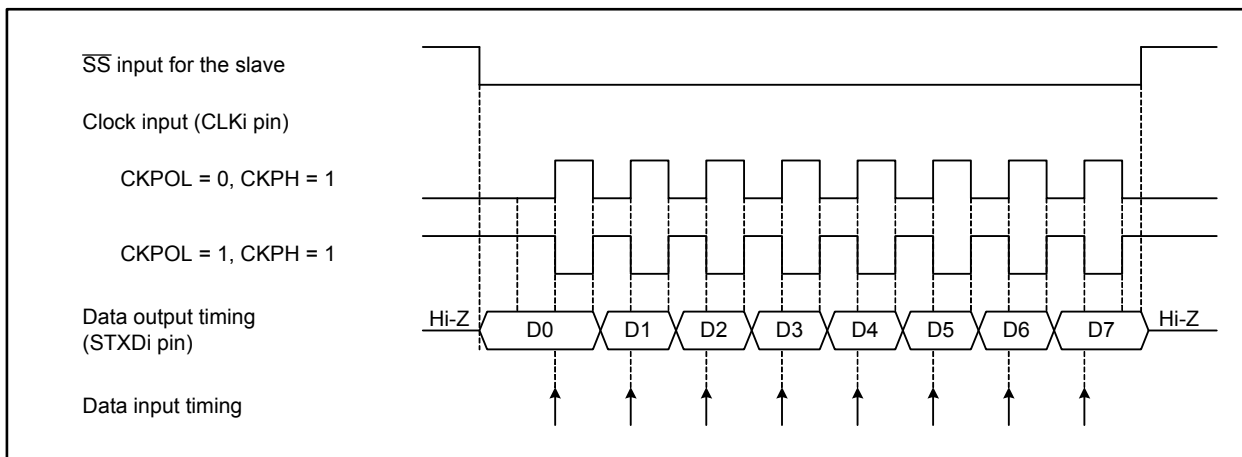


Figure 17.38 Transmit/Receive Timing in Slave Mode (CKPH = 1)

## 17.5 Notes on Serial Interface

### 17.5.1 Changing the UiBRG Register (i = 0 to 4)

- Set the UiBRG register after setting bits CLK1 and CLK0 in the UiC0 register. When these bits are changed, the UiBRG register must be set again.
- When a clock is input immediately after the UiBRG register is set to 00h, the counter may become FFh. In this case, it requires extra 256 clocks to reload 00h to the register. Once 00h is reloaded, the counter performs the operation without dividing the count source according to the setting.

### 17.5.2 Pin Output

When the NCH bit in the U2C0 register is set to 1, pins assigned for TXD2/SDA2 and SCL2 function as N-channel open drain output even if the UART output function is not selected in output function select register.

When the NODC bit in the U2SMR3 register is set to 1, CLK2 pins function as N-channel open drain output even if the UART output function is not selected in output function select register.

### 17.5.3 Synchronous Serial Interface Mode

#### 17.5.3.1 Selecting an External Clock

- If an external clock is selected, the following conditions must be met while the external clock is held high when the CKPOL bit in the UiC0 register is 0 (transmit data output on the falling edge of the transmit/receive clock and receive data input on the rising edge), or while the external clock is held low when the CKPOL bit is 1 (transmit data output on the rising edge of the transmit/receive clock and receive data input on the falling edge) (i = 0 to 4):
  - The TE bit in the UiC1 register is 1 (transmission enabled).
  - The RE bit in the UiC1 register is 1 (reception enabled). This bit setting is not required when only transmitting.
  - The TI bit in the UiC1 register is 0 (data held in the UiTB register).

#### 17.5.3.2 Receive Operation

- In synchronous serial interface mode, the transmit/receive clock is controlled by the transmit control circuit. Set UARTi-associated registers for a transmit operation, even if the MCU is used only for receive operation (i = 0 to 4). Dummy data is output from the TXDi pin while receiving when the TXDi pin is set to output mode.
- When data is received continuously, an overrun error occurs when the RI bit in the UiC1 register is 1 (data held in the UiRB register) and the seventh bit of the next data is received in the UARTi receive shift register. Then, the OER bit in the UiRB register becomes 1 (overrun error occurred). In this case, the UiRB register becomes undefined. If an overrun error occurs, the IR bit in the SiRIC register does not change to 1.

### 17.5.4 Special Mode 1 (I<sup>2</sup>C Mode)

- To generate a start condition, stop condition, or restart condition, set the STSPSEL bit in the UiSMR4 register to 0 (i = 0 to 2). Then, wait a half or more clock cycles of the transmit/receive clock to change the condition generate bits (STAREQ, RSTAREQ, or STPREQ bit) from 0 to 1.

### 17.5.5 Reset Procedure on Communication Error

Operations which result in communication errors such as rewriting function select registers during transmission/reception should not be performed. Follow the procedure below to reset the internal circuit once the communication error occurs in the following cases: when the operation above is performed by a receiver or transmitter or when a bit slip is caused by noise.

#### A. Synchronous Serial Interface Mode

- (1) Set the TE bit in the UiC1 register to 0 (transmission disabled) and the RE bit to 0 (reception disabled) (i = 0 to 4).
- (2) Set bits SMD2 to SMD0 in the UiMR register to 000b (serial interface disabled).
- (3) Set bits SMD2 to SMD0 in the UiMR register to 001b (synchronous serial interface mode).
- (4) Set the TE bit in the UiC1 register to 1 (transmission enabled) and the RE bit to 1 (reception enabled) if necessary.

#### B. UART Mode

- (1) Set the TE bit in the UiC1 register to 0 (transmission disabled) and the RE bit to 0 (reception disabled).
- (2) Set bits SMD2 to SMD0 in the UiMR register to 000b (serial interface disabled).
- (3) Set bits SMD2 to SMD0 in the UiMR register to 100b (UART mode, 7-bit character length), 101b (UART mode, 8-bit character length), or 110b (UART mode, 9-bit character length).
- (4) Set the TE bit in the UiC1 register to 1 (transmission enabled) and the RE bit to 1 (reception enabled) if necessary.

## 18. A/D Converter

The A/D converter consists of one 10-bit successive approximation A/D converter with a capacitive coupling amplifier.

A/D converted results are stored in the A/D registers corresponding to selected pins. Results are stored in the AD00 register only when the DMAC operating mode is enabled.

When the A/D converter is not in use, power consumption can be reduced by setting the VCUT bit in the AD0CON1 register to 0 (VREF disconnected). This bit setting enables the power supply from the VREF pin to the resistor ladder to stop.

Table 18.1 lists specifications of the A/D converter. Figure 18.1 shows a block diagram of the A/D converter. Figures 18.2 to 18.7 show registers associated with the A/D converter.

**Table 18.1 A/D Converter Specifications**

Item	Specification
A/D conversion method	Capacitance-based successive approximation
Analog input voltage <sup>(1)</sup>	0 V to AVCC (VCC)
Operating clock, $\phi_{AD}$ <sup>(2)</sup>	fAD, fAD divided by 2, fAD divided by 3, fAD divided by 4, fAD divided by 6, or fAD divided by 8
Resolution	8 bits or 10 bits
Operating modes	One-shot mode, repeat mode, single sweep mode, repeat sweep mode 0, repeat sweep mode 1, and self test mode
Analog input pins <sup>(3)</sup>	23 5 pins for AN, 8 pins each for AN0 and AN2 2 function-extended input pins (ANEX0 and ANEX1)
A/D conversion start conditions	<ul style="list-style-type: none"> <li>• Software trigger The ADST bit in the AD0CON0 register is set to 1 (A/D conversion started) by a program</li> <li>• External trigger (retrigger is enabled) An input signal at the <math>\overline{ADTRG}</math> pin switches from high to low after the ADST bit is set to 1 by a program</li> <li>• Hardware trigger (retrigger is enabled) <ul style="list-style-type: none"> <li>• Generation of a timer B2 interrupt request which has passed through the circuit to set an interrupt generating frequency in the three-phase motor control timers after the ADST bit is set to 1 by a program</li> <li>• Generation of a timer A0 interrupt request after the ADST bit is set to 1</li> </ul> </li> </ul>
Conversion rates per pin	<ul style="list-style-type: none"> <li>• Without sample and hold function 49 <math>\phi_{AD}</math> cycles at 8-bit resolution 59 <math>\phi_{AD}</math> cycles at 10-bit resolution including 2 <math>\phi_{AD}</math> cycles for sampling time</li> <li>• With sample and hold function 28 <math>\phi_{AD}</math> cycles at 8-bit resolution 33 <math>\phi_{AD}</math> cycles at 10-bit resolution including 3 <math>\phi_{AD}</math> cycles for sampling time</li> </ul>

**Notes:**

1. The analog input voltage is not dependent on whether the sample and hold function is enabled or disabled.
2. The  $\phi_{AD}$  frequency should be as follows:
  - When VCC = 4.2 to 5.5 V, 16 MHz or below
  - When VCC = 3.0 to 4.2 V, 10 MHz or below
  - When not using the sample and hold function, 250 kHz or above
  - When using the sample and hold function, 1 MHz or above
3. When AVCC = VREF = VCC, A/D input voltage for pins AN\_0 to AN\_4, AN0\_0 to AN0\_7, AN2\_0 to AN2\_7, ANEX0, and ANEX1 should be VCC or lower.

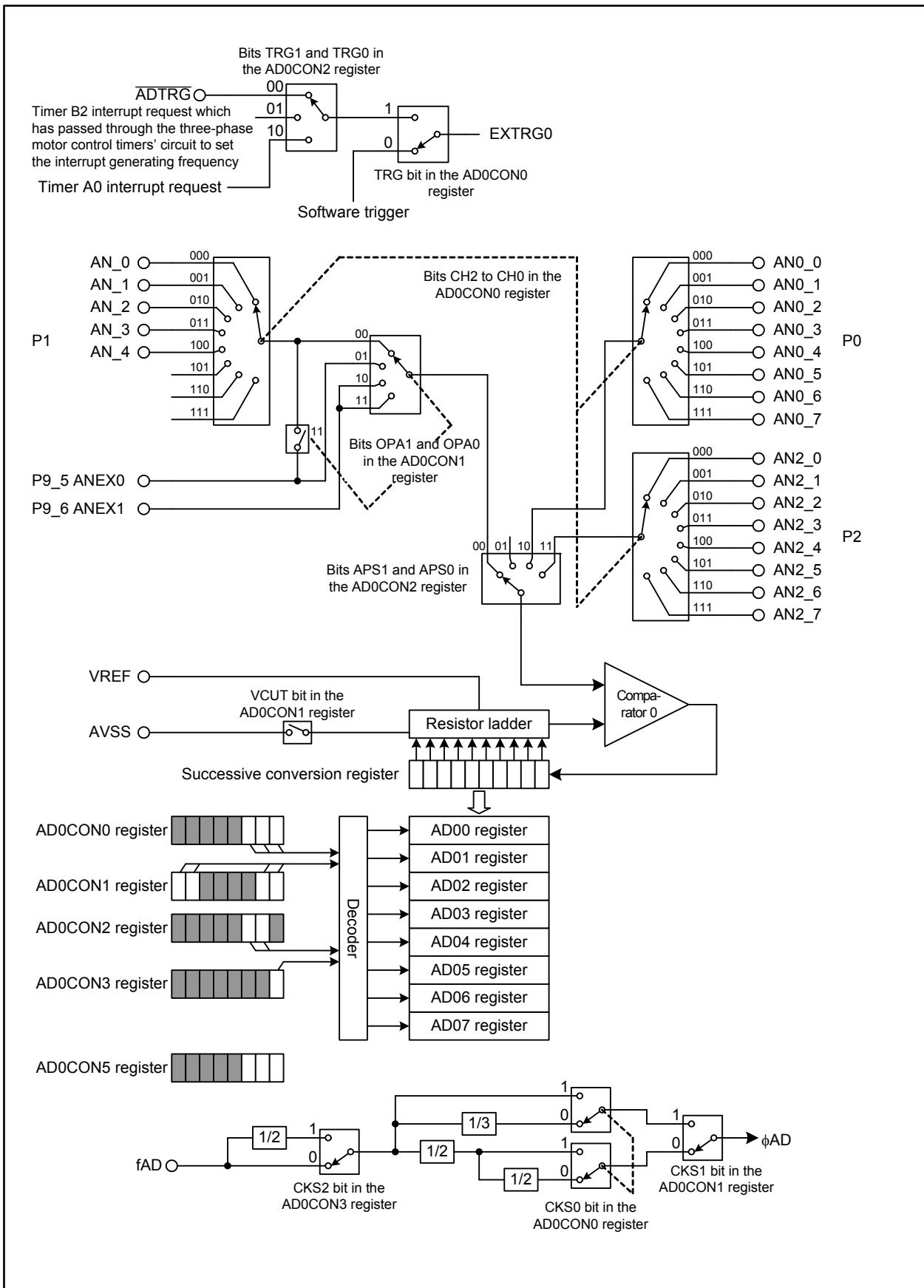


Figure 18.1 A/D Converter Block Diagram

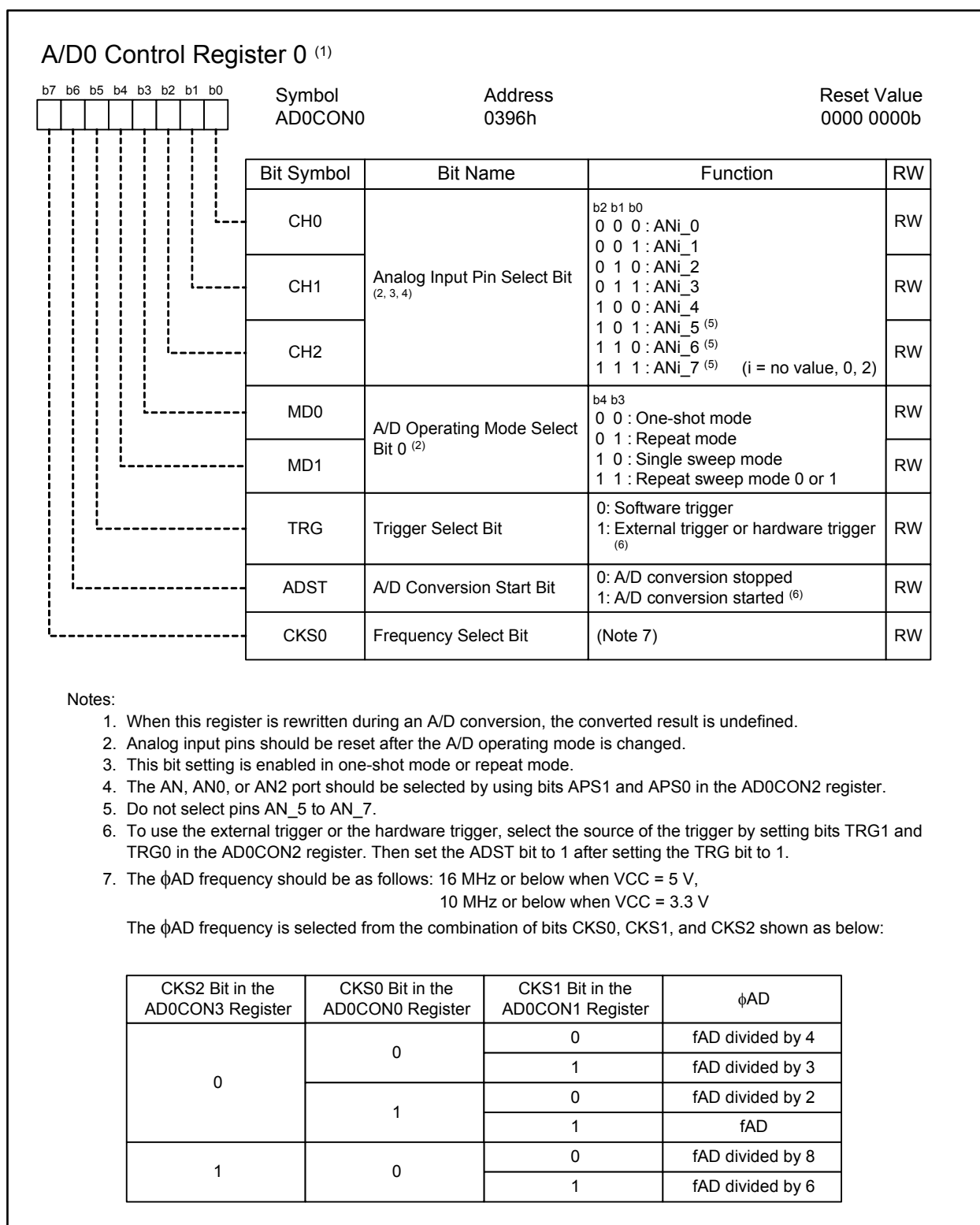


Figure 18.2 AD0CON0 Register

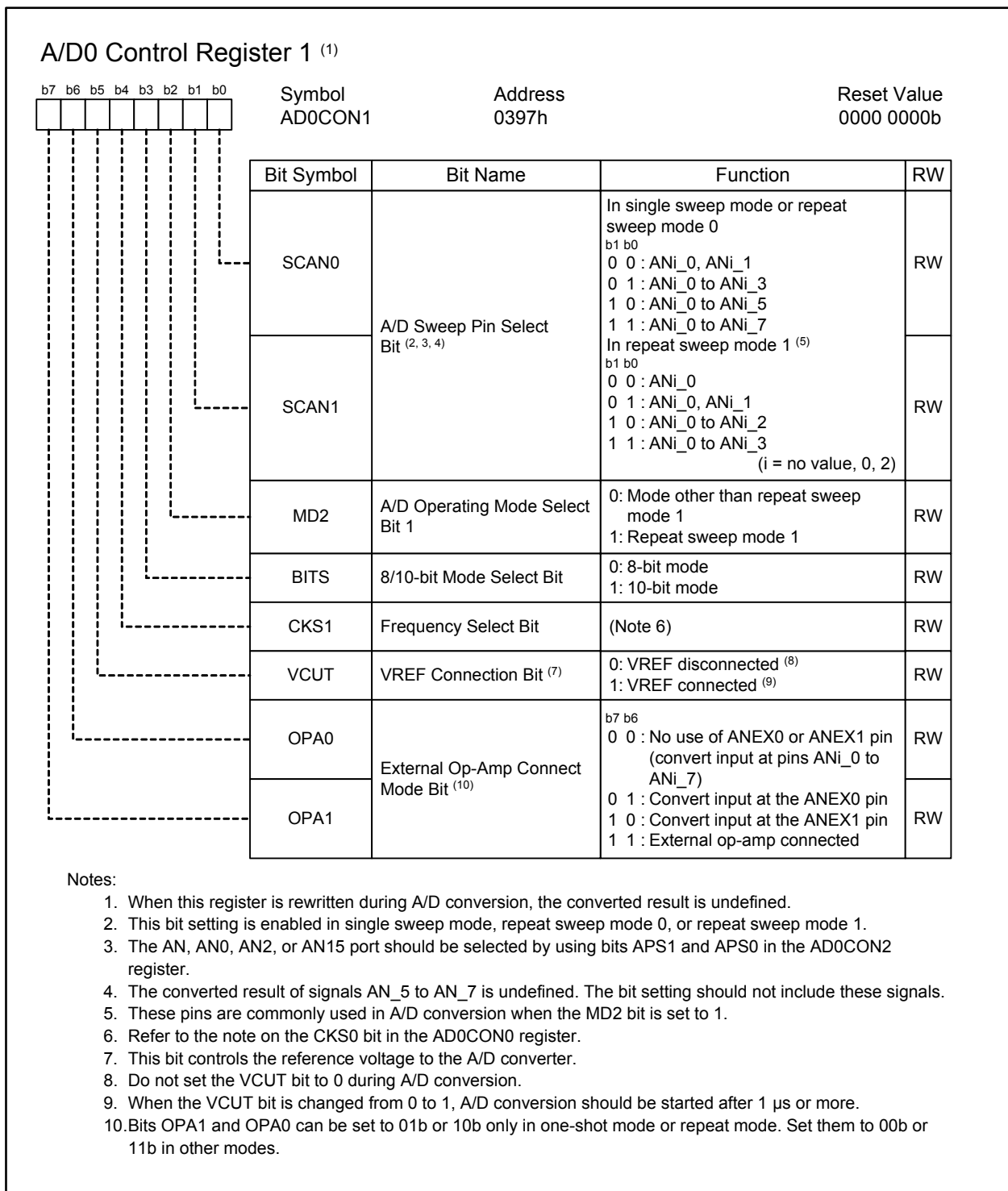


Figure 18.3 AD0CON1 Register



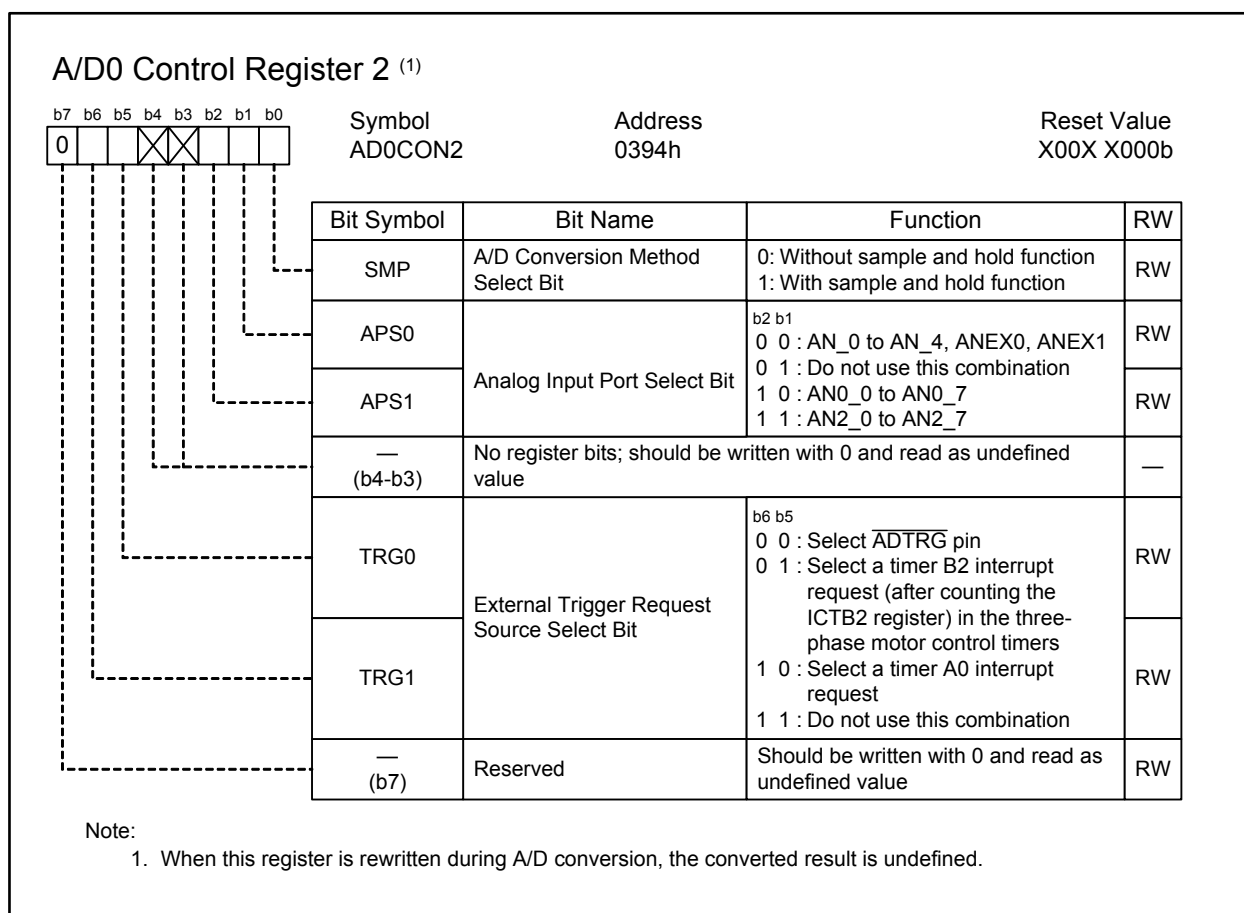
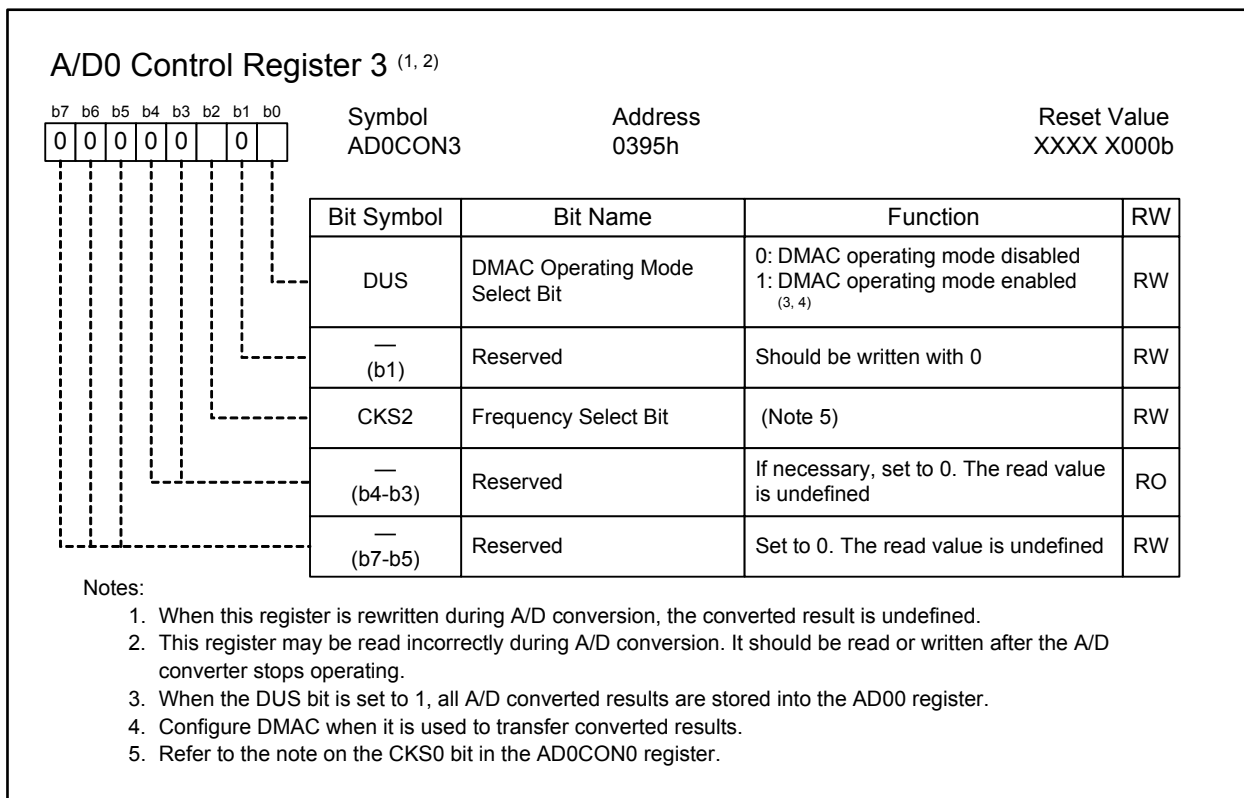


Figure 18.4 AD0CON2 Register



**Figure 18.5 AD0CON3 Register**

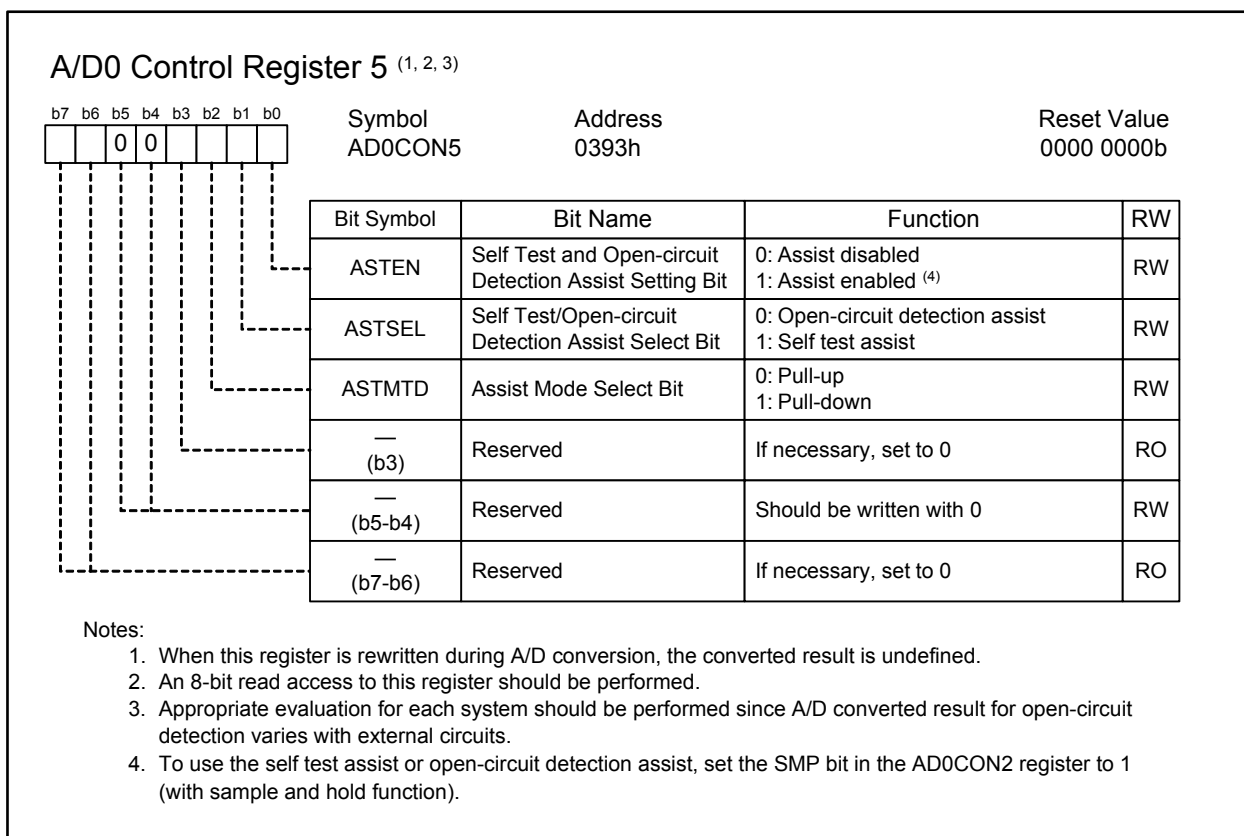


Figure 18.6 AD0CON5 Register

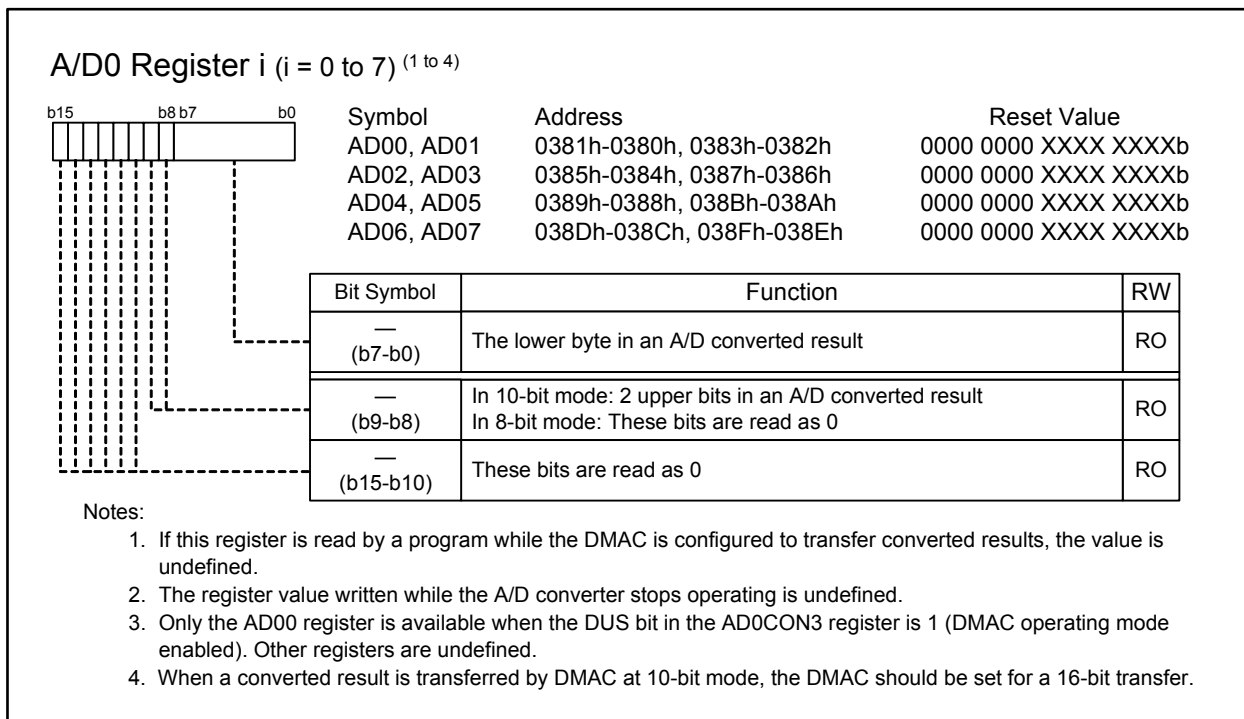


Figure 18.7 Registers AD00 to AD07

## 18.1 Mode Descriptions

### 18.1.1 One-shot Mode

In one-shot mode, the analog voltage applied to a selected pin is converted into a digital code only once. Table 18.2 lists specifications of one-shot mode.

**Table 18.2 One-shot Mode Specifications**

Item	Specification
Function	Converts the analog voltage applied to a pin into a digital code only once. The pin is selected by setting bits CH2 to CH0 in the AD0CON0 register, bits OPA1 and OPA0 in the AD0CON1 register, and bits APS1 and APS0 in the AD0CON2 register
Start conditions	<p>When the TRG bit in the AD0CON0 register is 0 (software trigger)            The ADST bit in the AD0CON0 register is set to 1 (A/D conversion started) by a program.</p> <p>When the TRG bit is 1 (external trigger or hardware trigger)            Set bits TRG1 and TRG0 in the AD0CON2 register to select external trigger request source.</p> <ul style="list-style-type: none"> <li>• When 00b is selected,              an input signal at the <math>\overline{\text{ADTRG}}</math> pin switches from high to low after the ADST bit is set to 1 by a program.</li> <li>• When 01b is selected,              generation of a timer B2 interrupt request which has passed through the circuit to set the interrupt generating frequency in the three-phase motor control timers after the ADST bit is set to 1 by a program.</li> <li>• When 10b is selected,              generation of a timer A0 interrupt request after the ADST bit is set to 1</li> </ul>
Stop conditions	<ul style="list-style-type: none"> <li>• A/D conversion is completed (the ADST bit is set to 0 when the software trigger is selected)</li> <li>• The ADST bit is set to 0 (A/D conversion stopped) by a program</li> </ul>
Interrupt request generation timing	When A/D conversion is completed, an interrupt request is generated
Input pin to be selected	One pin is selected from among AN_0 to AN_4, AN0_0 to AN0_7, AN2_0 to AN2_7, ANEX0, and ANEX1
Reading A/D converted result	<p>When the DUS bit in the AD0CON3 register is 0 (DMAC operating mode disabled)            Read the AD0j register corresponding to the selected pin (j = 0 to 7)</p> <p>When the DUS bit is 1 (DMAC operating mode enabled)            Configure the DMAC (refer to 12. "DMAC").            Then the A/D converted result is stored in the AD00 register after the conversion is completed. The DMAC transfers the converted result from the AD00 register to a given memory space.            Do not read the AD00 register by a program</p>

### 18.1.2 Repeat Mode

In repeat mode, the analog voltage applied to a selected pin is repeatedly converted into a digital code. Table 18.3 lists specifications of repeat mode.

**Table 18.3 Repeat Mode Specifications**

Item	Specification
Function	Converts the analog voltage input to a pin into a digital code repeatedly. The pin is selected by setting bits CH2 to CH0 in the AD0CON0 register, bits OPA1 and OPA0 in the AD0CON1 register, and bits APS1 and APS0 in the AD0CON2 register
Start conditions	<p>When the TRG bit in the AD0CON0 register is 0 (software trigger) The ADST bit in the AD0CON0 register is set to 1 (A/D conversion started) by a program.</p> <p>When the TRG bit is 1 (external trigger or hardware trigger) Set bits TRG1 and TRG0 in the AD0CON2 register to select external trigger request source.</p> <ul style="list-style-type: none"> <li>• When 00b is selected, an input signal at the ADTRG pin switches from high to low after the ADST bit is set to 1 by a program.</li> <li>• When 01b is selected, generation of a timer B2 interrupt request which has passed through the circuit to set the interrupt generating frequency in the three-phase motor control timers after the ADST bit is set to 1 by a program.</li> <li>• When 10b is selected, generation of a timer A0 interrupt request after the ADST bit is set to 1</li> </ul>
Stop conditions	The ADST bit is set to 0 (A/D conversion stopped) by a program
Interrupt request generation timing	<p>When the DUS bit in the AD0CON3 register is 0 (DMAC operating mode disabled), no interrupt request is generated.</p> <p>When the DUS bit is 1 (DMAC operating mode enabled), each time A/D conversion is completed, an interrupt request is generated</p>
Analog voltage input pins	One pin is selected from among AN_0 to AN_4, AN0_0 to AN0_7, AN2_0 to AN2_7, ANEX0, and ANEX1
Reading A/D converted result	<p>When the DUS bit in the AD0CON3 register is 0 (DMAC operating mode disabled) Read the AD0j register corresponding to the selected pin (j = 0 to 7)</p> <p>When the DUS bit is 1 (DMAC operating mode enabled)</p> <ul style="list-style-type: none"> <li>• When the converted result is transferred by DMAC Configure the DMAC (refer to 12. "DMAC"). Then the A/D converted result is stored in the AD00 register after the conversion is completed. The DMAC transfers the converted result from the AD00 register to a given memory space. Do not read the AD00 register by a program</li> <li>• When the converted result is transferred by a program Read the AD00 register after the IR bit in the AD0IC register becomes 1. Set the IR bit back to 0</li> </ul>

### 18.1.3 Single Sweep Mode

In single sweep mode, the analog voltage applied to selected pins is converted one-by-one into a digital code. Table 18.4 lists specifications of single sweep mode.

**Table 18.4 Single Sweep Mode Specifications**

Item	Specification
Function	Converts the analog voltage input to a set of pins into a digital code one-by-one. The pins are selected by setting bits SCAN1 and SCAN0 in the AD0CON1 register and bits APS1 and APS0 in the AD0CON2 register
Start conditions	When the TRG bit in the AD0CON0 register is 0 (software trigger) The ADST bit in the AD0CON0 register is set to 1 (A/D conversion started) by a program. When the TRG bit is 1 (external trigger or hardware trigger) Set bits TRG1 and TRG0 in the AD0CON2 register to select external trigger request source. <ul style="list-style-type: none"> <li>• When 00b is selected, an input signal at the <math>\overline{\text{ADTRG}}</math> pin switches from high to low after the ADST bit is set to 1 by a program.</li> <li>• When 01b is selected, generation of a timer B2 interrupt request which has passed through the circuit to set the interrupt generating frequency in the three-phase motor control timers after the ADST bit is set to 1 by a program.</li> <li>• When 10b is selected, generation of a timer A0 interrupt request after the ADST bit is set to 1</li> </ul>
Stop conditions	<ul style="list-style-type: none"> <li>• A/D conversion is completed (the ADST bit is set to 0 when the software trigger is selected)</li> <li>• The ADST bit is set to 0 (A/D conversion stopped) by a program</li> </ul>
Interrupt request generation timing	When the DUS bit in the AD0CON3 register is 0 (DMAC operating mode disabled) when a sweep is completed, an interrupt request is generated. When the DUS bit is 1 (DMAC operating mode enabled), each time A/D conversion is completed, an interrupt request is generated
Analog voltage input pins	Selected from a group of 2 pins (AN <sub>i_0</sub> and AN <sub>i_1</sub> ), 4 pins (AN <sub>i_0</sub> to AN <sub>i_3</sub> ), 6 pins (AN <sub>i_0</sub> to AN <sub>i_5</sub> ), or 8 pins (AN <sub>i_0</sub> to AN <sub>i_7</sub> ) (i = no value, 0, 2) <sup>(1)</sup>
Reading A/D converted result	When the DUS bit in the AD0CON3 register is 0 (DMAC operating mode disabled) Read the AD0 <sub>j</sub> register corresponding to the selected pin (j = 0 to 7) When the DUS bit is 1 (DMAC operating mode enabled) Configure the DMAC (refer to 12. "DMAC"). Then the A/D converted result is stored in the AD00 register after the conversion is completed. The DMAC transfers the converted result from the AD00 register to a given memory space. Do not read the AD00 register by a program

Note:

1. When bits APS1 and APS0 are set to 00b (AN<sub>0</sub> to AN<sub>4</sub>, ANEX<sub>0</sub>, and ANEX<sub>1</sub> selected), set bits SCAN1 and SCAN0 to 00b (AN<sub>i\_0</sub> and AN<sub>i\_1</sub> selected) or 01b (AN<sub>i\_0</sub> to AN<sub>i\_3</sub> selected). If bits SCAN1 and SCAN0 are set to 10b (AN<sub>i\_0</sub> to AN<sub>i\_5</sub> selected), the converted result of the AN<sub>5</sub> signal is undefined.

### 18.1.4 Repeat Sweep Mode 0

In repeat sweep mode 0, the analog voltage applied to selected pins is repeatedly converted into a digital code. Table 18.5 lists specifications of repeat sweep mode 0.

**Table 18.5 Repeat Sweep Mode 0 Specifications**

Item	Specification
Function	Converts the analog voltage input to a set of pins into a digital code repeatedly. The pins are selected by setting bits SCAN1 and SCAN0 in the AD0CON1 register and APS1 and APS0 in the AD0CON2 register
Start conditions	When the TRG bit in the AD0CON0 register is 0 (software trigger) The ADST bit in the AD0CON0 register is set to 1 (A/D conversion started) by a program. When the TRG bit is 1 (external trigger or hardware trigger) Set bits TRG1 and TRG0 in the AD0CON2 register to select external trigger request source. <ul style="list-style-type: none"> <li>• When 00b is selected, an input signal at the <math>\overline{\text{ADTRG}}</math> pin switches from high to low after the ADST bit is set to 1 by a program.</li> <li>• When 01b is selected, generation of a timer B2 interrupt request which has passed through the circuit to set the interrupt generating frequency in the three-phase motor control timers after the ADST bit is set to 1 by a program.</li> <li>• When 10b is selected, generation of a timer A0 interrupt request after the ADST bit is set to 1</li> </ul>
Stop conditions	The ADST bit is set to 0 (A/D conversion stopped) by a program
Interrupt request generation timing	When the DUS bit in the AD0CON3 register is 0 (DMAC operating mode disabled), no interrupt request is generated. When the DUS bit is 1 (DMAC operating mode enabled), each time A/D conversion is completed, an interrupt request is generated
Analog voltage input pins	Selected from a group of 2 pins (AN <sub>i_0</sub> and AN <sub>i_1</sub> ), 4 pins (AN <sub>i_0</sub> to AN <sub>i_3</sub> ), 6 pins (AN <sub>i_0</sub> to AN <sub>i_5</sub> ), or 8 pins (AN <sub>i_0</sub> to AN <sub>i_7</sub> ) (i = no value, 0, 2) <sup>(1)</sup>
Reading A/D converted result	When the DUS bit in the AD0CON3 register is 0 (DMAC operating mode disabled) Read the AD0 <sub>j</sub> register corresponding to the selected pin (j = 0 to 7) When the DUS bit is 1 (DMAC operating mode enabled) <ul style="list-style-type: none"> <li>• When the converted result is transferred by DMAC Configure the DMAC (refer to 12. "DMAC"). Then the A/D converted result is stored in the AD00 register after the conversion is completed. The DMAC transfers the converted result from the AD00 register to a given memory space. Do not read the AD00 register by a program</li> <li>• When the converted result is transferred by a program Read the AD00 register after the IR bit in the AD0IC register becomes 1. Set the IR bit back to 0</li> </ul>

Note:

1. When bits APS1 and APS0 are set to 00b (AN<sub>0</sub> to AN<sub>4</sub>, ANEX0, and ANEX1 selected), set bits SCAN1 and SCAN0 to 00b (AN<sub>i\_0</sub> and AN<sub>i\_1</sub> selected) or 01b (AN<sub>i\_0</sub> to AN<sub>i\_3</sub> selected). If bits SCAN1 and SCAN0 are set to 10b (AN<sub>i\_0</sub> to AN<sub>i\_5</sub> selected), the converted result of the AN<sub>5</sub> signal is undefined.

### 18.1.5 Repeat Sweep Mode 1

In repeat sweep mode 1, the analog voltage applied to eight selected pins including one to four prioritized pins is repeatedly converted into a digital code. Table 18.6 lists specifications of repeat sweep mode 1.

**Table 18.6 Repeat Sweep Mode 1 Specifications**

Item	Specification
Function	The analog voltage applied to eight selected pins including one to four prioritized pins is repeatedly converted into a digital code. The prioritized pins are selected by setting bits SCAN1 and SCAN0 in the AD0CON1 register and bits APS1 and APS0 in the AD0CON2 register For example, when AN0_0 is selected, the A/D conversion is performed in the following order: AN0_0→AN0_1→AN0_0→AN0_2→AN0_0→AN0_3...
Start conditions	When the TRG bit in the AD0CON0 register is 0 (software trigger) The ADST bit in the AD0CON0 register is set to 1 (A/D conversion started) by a program. When the TRG bit is 1 (external trigger or hardware trigger) Set bits TRG1 and TRG0 in the AD0CON2 register to select external trigger request source. <ul style="list-style-type: none"> <li>• When 00b is selected, an input signal at the ADTRG pin switches from high to low after the ADST bit is set to 1 by a program. Retrigger is invalid.</li> <li>• When 01b is selected, generation of a timer B2 interrupt request which has passed through the circuit to set the interrupt generating frequency in the three-phase motor control timers after the ADST bit is set to 1 by a program.</li> <li>• When 10b is selected, generation of a timer A0 interrupt request after the ADST bit is set to 1</li> </ul>
Stop conditions	The ADST bit is set to 0 (A/D conversion stopped) by a program
Interrupt request generation timing	When the DUS bit in the AD0CON3 register is 0 (DMAC operating mode disabled), no interrupt request is generated. When the DUS bit is 1 (DMAC operating mode enabled), each time A/D conversion is completed, an interrupt request is generated
Analog voltage input pins	8 (ANi_0 to ANi_7) (i = 0, 2) <sup>(1)</sup>
Prioritized pin(s)	Selected from a group of 1 pin (ANi_0), 2 pins (ANi_0 and ANi_1), 3 pins (ANi_0 to ANi_2), or 4 pins (ANi_0 to ANi_3)
Reading A/D converted result	When the DUS bit in the AD0CON3 register is 0 (DMAC operating mode disabled) Read the AD0j register corresponding to the selected pin (j = 0 to 7) When the DUS bit is 1 (DMAC operating mode enabled) <ul style="list-style-type: none"> <li>• When the converted result is transferred by DMAC Configure the DMAC (refer to 12. "DMAC"). Then the A/D converted result is stored in the AD00 register after the conversion is completed. The DMAC transfers the converted result from the AD00 register to a given memory space. Do not read the AD00 register by a program</li> <li>• When the converted result is transferred by a program Read the AD00 register after the IR bit in the AD0IC register becomes 1. Set the IR bit back to 0</li> </ul>

Note:

1. The converted result of signals AN\_5 to AN\_7 is undefined.



## 18.2 Functions

### 18.2.1 Resolution Selection

Resolution is selected by setting the BITS bit in the AD0CON1 register. When the BITS bit is set to 1 (10-bit precision), the A/D converted result is stored into bits 9 to 0 in the AD0i register (i = 0 to 7). When the BITS bit is set to 0 (8-bit precision), the result is stored into bits 7 to 0 in the AD0i register.

### 18.2.2 Sample and Hold Function

This function improves the conversion rate per pin to 28  $\phi$ AD cycles at 8-bit resolution and 33  $\phi$ AD cycles for 10-bit resolution. This function is available in all operating modes and is enabled by setting the SMP bit in the AD0CON2 register to 1 (with sample and hold function). Start A/D conversion after setting the SMP bit.

### 18.2.3 Trigger Selection

A trigger to start A/D conversion is specified by the combination of TRG bit in the AD0CON0 register and bits TRG1 and TRG0 in the AD0CON2 register. Table 18.7 lists the settings of the trigger selection.

**Table 18.7 Trigger Selection Settings**

Bit and Setting		Trigger
AD0CON0 register	AD0CON2 register	
TRG = 0	—	Software trigger The ADST bit in the AD0CON0 register is set to 1
TRG = 1 (1, 2)	TRG1 = 0, TRG0 = 0	External trigger Falling edge of a signal applied to the $\overline{\text{ADTRG}}$ pin
	TRG1 = 0, TRG0 = 1	Hardware trigger Generation of a timer B2 interrupt request which has passed through the circuit to set the interrupt generating frequency in the three-phase motor control timers
	TRG1 = 1, TRG0 = 0	Hardware trigger Generation of a timer A0 interrupt request

Notes:

1. A/D conversion starts when a trigger is generated while the ADST bit is 1 (A/D conversion started).
2. When an external trigger or a hardware trigger is generated during A/D conversion, the A/D converter aborts the operation in progress. Then, it restarts the operation.

### 18.2.4 DMAC Operating Mode

DMAC operating mode can be used in all operating modes. When the DUS bit in the AD0CON3 register is set to 1 (DMAC operating mode enabled), all A/D converted results are stored in the AD00 register. The DMAC transfers the data from the AD00 register to a given memory space every time A/D conversion is completed at a pin. 8-bit DMA transfer should be selected for 8-bit resolution. For 10-bit resolution, 16-bit DMA transfer should be selected. Refer to 12. "DMAC" for details.

### 18.2.5 Function-extended Analog Input Pins

In one-shot mode and repeat mode, pins ANEX0 and ANEX1 can be used as analog input pins by setting bits OPA1 and OPA0 in the AD0CON1 register (refer to Table 18.8). The A/D converted results of pins ANEX0 and ANEX1 are stored into registers AD00 and AD01, respectively. However, when the DUS bit in the AD0CON3 register is set to 1 (DMAC operating mode enabled), all results are stored into the AD00 register.

To use function-extended analog input pins, bits APS1 and APS0 in the AD0CON2 register should be set to 00b (AN0 to AN7, ANEX0, ANEX1 function as analog input ports).

**Table 18.8 Function-extended Analog Input Pin Settings**

AD0CON1 Register		ANEX0	ANEX1
OPA1	OPA0		
0	0	Not used	Not used
0	1	Analog input	Not used
1	0	Not used	Analog input
1	1	Output to an external op-amp	Input from an external op-amp

### 18.2.6 External Operating Amplifier (Op-Amp) Connection Mode

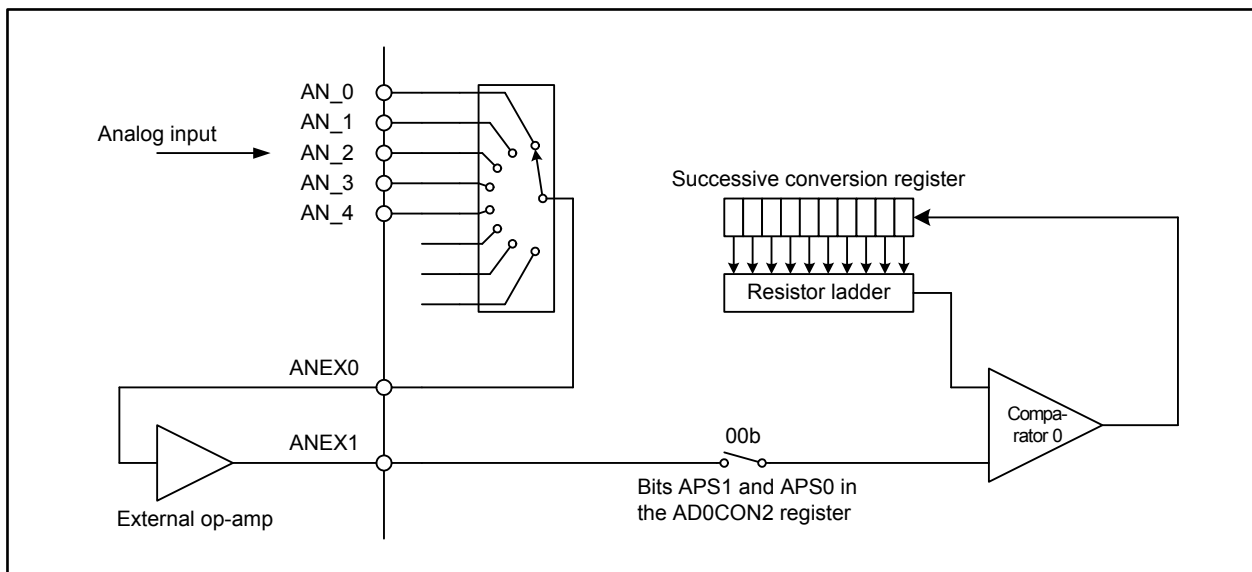
In external op-amp connection mode, multiple analog inputs can be amplified by one external op-amp using function-extended analog input pins ANEX0 and ANEX1.

When bits OPA1 and OPA0 in the AD0CON1 register are 11b (external op-amp connected), the voltage applied to pins AN0 to AN7 is output from the ANEX0 pin. This output signal should be amplified by an external op-amp and applied to the ANEX1 pin.

The analog voltage applied to the ANEX1 pin is converted into a digital code. The converted result is stored in the corresponding AD0i register ( $i = 0$  to 7). The conversion rate varies with the response of the external op-amp. Note that the ANEX0 pin should not be connected to the ANEX1 pin directly.

To use external op-amp connection mode, set bits APS1 and APS0 in the AD0CON2 register to 00b.

Figure 18.8 shows an example of an external op-amp connection.

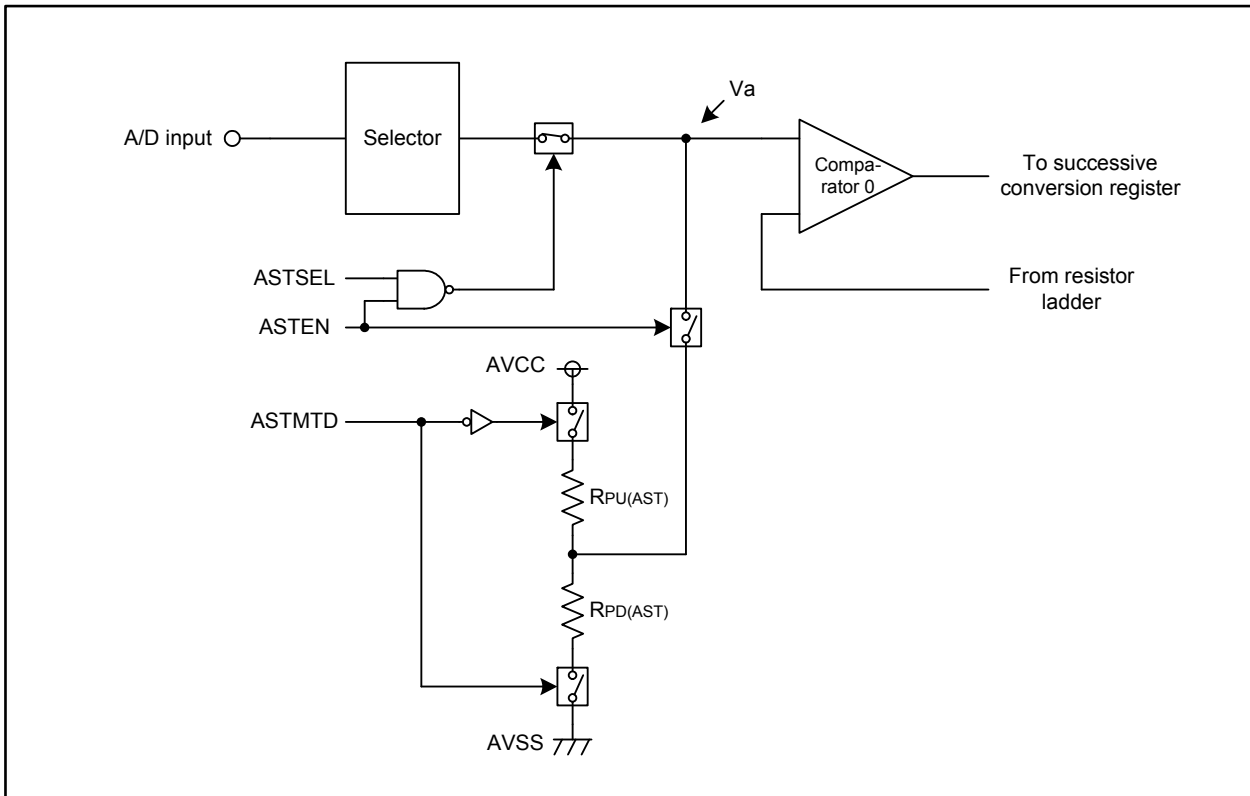


**Figure 18.8 External Op-Amp Connection**

### 18.2.7 Self Test/Open-circuit Detection Assist

This function enables the MCU to detect open-circuit in analog input pins. It also enables it to perform a self test.

Figure 18.9 shows a block diagram of open-circuit detection assist circuit.



**Figure 18.9 Open-circuit Detection Assist Circuit**

To detect open-circuit, the ASTSEL bit in the AD0CON5 register should be set to 0 (open-circuit detection assist).  $V_a$  in Figure 18.9 above is a voltage with a value between AVCC and the A/D applied voltage in the following bit settings: the ASTEN bit is 1 (assist enabled), the ASTSEL bit is 0, and the ASTMTD bit is 0 (pull-up). If the A/D input pin is open,  $V_a$  is almost equal to AVCC. When the ASTEN bit is 1, the ASTSEL bit is 0, and the ASTMTD bit is 1 (pull-down),  $V_a$  is between AVSS and the A/D applied voltage. If the A/D input pin is open,  $V_a$  is almost AVSS. That is, the A/D input pin is considered open if the result value of A/D conversion is almost the maximum/minimum voltage.

To enable the self test function, the ASTSEL bit in the AD0CON5 register should be set to 1 (self test assist).  $V_a$  is almost equal to AVCC in the bit settings as the ASTEN bit is 1, the ASTSEL bit is 1, and the ASTMTD bit is 0. When the ASTEN bit is 1, the ASTSEL bit is 1, and the ASTMTD bit is 1,  $V_a$  is almost AVSS. That is, if the result value of A/D conversion is almost the maximum/minimum voltage in each bit setting, the A/D converter is considered to be functioning normally.

### 18.2.8 Power Saving

When the A/D converter is not in use, power consumption can be reduced by setting the VCUT bit in the AD0CON1 to 0 (VREF disconnected). With this bit setting, the reference voltage input pin (VREF) can be disconnected from the resistor ladder, which enables the power supply from the VREF to the resistor ladder to stop.

To use the A/D converter, set the VCUT bit to 1 (VREF connected) and wait at least 1  $\mu\text{s}$  before setting the ADST bit in the AD0CON0 register to 1 (A/D conversion started). Bits ADST and VCUT should not be set to 1 simultaneously. The VCUT bit should not be set to 0 during A/D conversion.

The VCUT bit does not affect VREF performance of the D/A converter (refer to Figure 18.10).

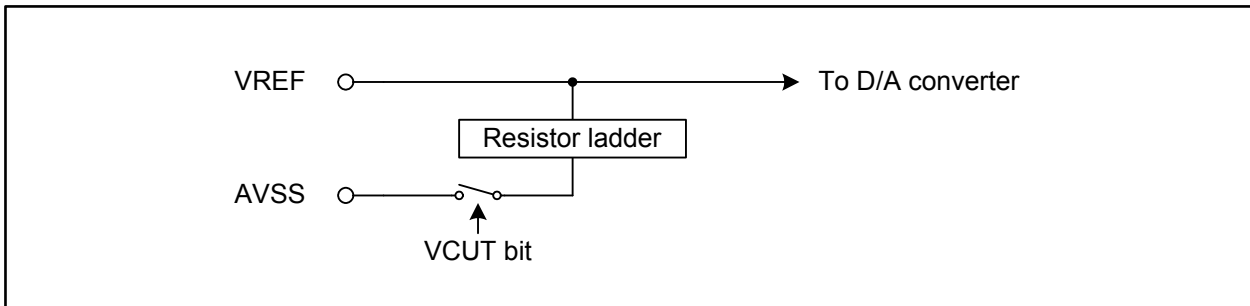


Figure 18.10 Power Supply by VCUT Bit

### 18.2.9 Output Impedance of Sensor Equivalent Circuit under A/D Conversion

Figure 18.11 shows an analog input pin and external sensor equivalent circuit.

To perform A/D conversion correctly, the internal capacitor (C) charging, shown in Figure 18.11, should be completed within the specified period. This period, called the sampling time, is 2  $\phi\text{AD}$  cycles for conversion without the sample and hold function and 3  $\phi\text{AD}$  cycles for conversion with this function.

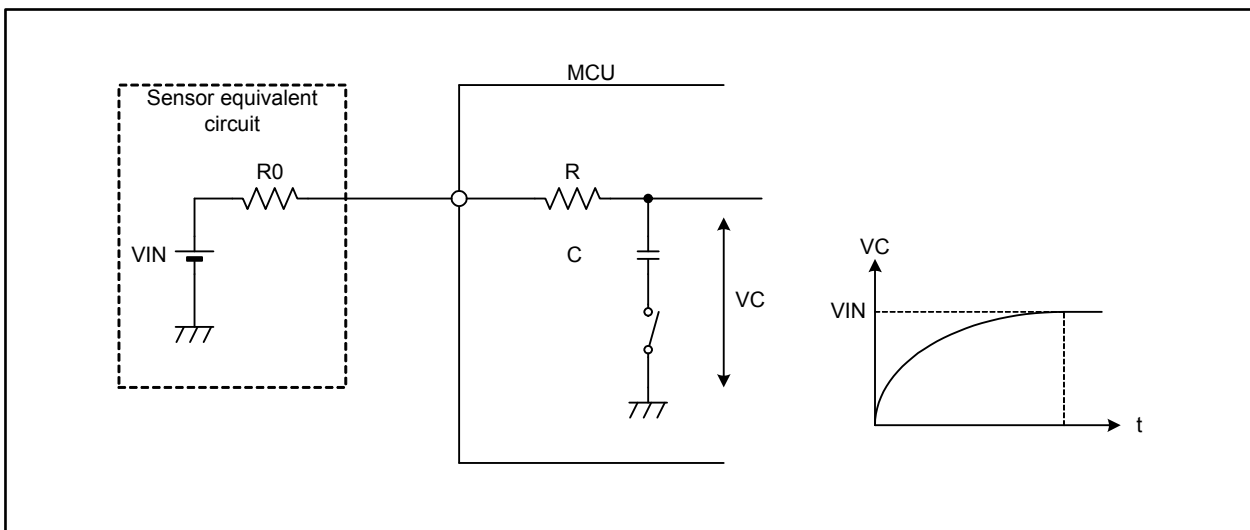


Figure 18.11 Analog Input Pin and External Sensor Equivalent Circuitry

The voltage between pins (VC) is expressed as follows:

$$VC = VIN \left\{ 1 - e^{-\frac{t}{C(R0+R)}} \right\}$$

When  $t = T$  and the precision (error) is  $x$  or less,

$$VC = VIN - \frac{x}{y} VIN = VIN \left( 1 - \frac{x}{y} \right)$$

Thus, output impedance of the sensor equivalent circuit (R0) is determined by the following formulas:

$$e^{-\frac{T}{C(R0+R)}} = \frac{x}{y}$$

$$-\frac{T}{C(R0+R)} = \ln \frac{x}{y}$$

$$R0 = -\frac{T}{C \ln \frac{x}{y}} - R$$

where:

T[s] = Sampling time

R0[Ω] = Output impedance of the sensor equivalent circuit

VC = Potential difference between edges of capacitor C

R[Ω] = Internal resistance of the MCU

x[LSB] = Precision (error) of the A/D converter

y[step] = Resolution of the A/D converter (1024 steps at 10-bit mode, 256 steps at 8-bit mode)

When  $\phi_{AD} = 10$  MHz, the A/D conversion mode is 10-bit resolution with the sample and hold function, the output impedance (R0) with the precision (error) of 0.1 LSB or less is determined by the following formula:

Using  $T = 0.3 \mu\text{s}$ ,  $R = 2.0 \text{ k}\Omega$  (reference value),  $C = 6.5 \text{ pF}$  (reference value),  $x = 0.1$ ,  $y = 1024$ ,

$$R0 = -\frac{0.3 \times 10^{-6}}{6.5 \times 10^{-12} \times \ln \frac{0.1}{1024}} - 2.0 \times 10^3$$

$$= 2998$$

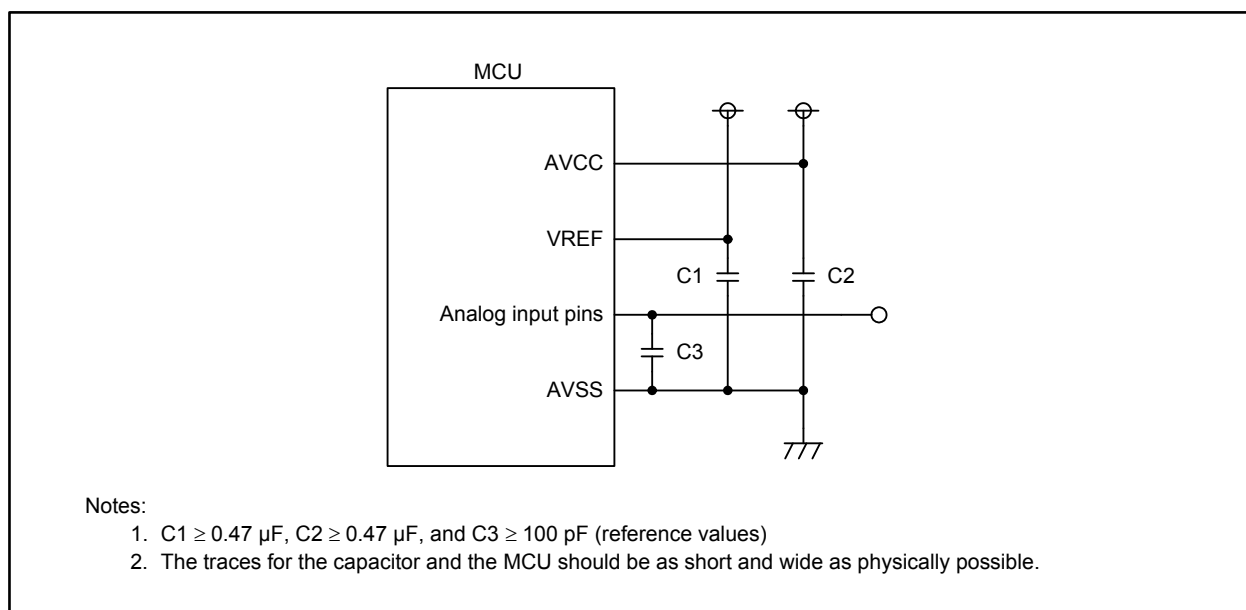
Thus, the allowable output impedance of the sensor equivalent circuit (R0), making the precision (error) of 0.1 LSB or less, should be less than 3 kΩ.

The actual error, however, is the value of absolute precision added to the 0.1 LSB mentioned above.

## 18.3 Notes on A/D Converter

### 18.3.1 Notes on Designing Boards

- Three capacitors should be placed between the AVSS pin and pins such as AVCC, VREF, and analog inputs (AN\_0 to AN\_4, AN0\_0 to AN0\_7, and AN2\_0 to AN2\_7) to avoid erroneous operations caused by noise or latchup, and to reduce conversion errors. Figure 18.12 shows an example of pin configuration for A/D converter.



**Figure 18.12 Pin Configuration for the A/D Converter**

- When  $AVCC = VREF = VCC$ , A/D input voltage for pins AN\_0 to AN\_4, AN0\_0 to AN0\_7, AN2\_0 to AN2\_7, ANEX0, and ANEX1 should be VCC or lower.

### 18.3.2 Notes on Programming

- The following registers should be written while A/D conversion is stopped. That is, before a trigger occurs: AD0CON0 (except the ADST bit), AD0CON1, AD0CON2, AD0CON3, and AD0CON5.
- When the VCUT bit in the AD0CON1 register is changed from 0 (VREF connected) to 1 (VREF disconnected), wait for at least 1  $\mu$ s before starting A/D conversion. When not performing A/D conversion, set the VCUT bit to 0 to reduce power consumption.
- Set the port direction bit for the pin to be used as an analog input pin to 0 (input). Set the ASEL bit of the corresponding port function select register to 1 (port is used as A/D input).
- When the TRG bit in the AD0CON0 register is 1 (external trigger or hardware trigger), set the corresponding port direction bit (PD9\_7 bit) for the  $\overline{\text{ADTRG}}$  pin to 0 (input).
- The  $\phi_{\text{AD}}$  frequency should be 16 MHz or lower when VCC is 4.2 to 5.5 V, and 10 MHz or lower when VCC is 3.0 to 4.2 V. It should be 1 MHz or higher when the sample and hold function is enabled. If not, it should be 250 kHz or higher.
- When A/D operating mode (bits MD1 and MD0 in the AD0CON0 register or the MD2 bit in the AD0CON1 register) has been changed, reselect analog input pins by setting bits CH2 to CH0 in the AD0CON0 register or bits SCAN1 and SCAN0 in the AD0CON1 register.
- If the AD0i register is read when the A/D converted result is stored to the register, the stored value may have an error ( $i = 0$  to 7). Read the AD0i register after A/D conversion is completed. In one-shot mode or single sweep mode, read the AD0i register after the IR bit in the AD0IC register becomes 1 (interrupt requested). In repeat mode, repeat sweep mode 0, or repeat sweep mode 1, an interrupt request can be generated each time A/D conversion is completed when the DUS bit in the AD0CON3 register is 1 (DMAC operating mode enabled). Similar to the other modes above, read the AD00 register after the IR bit in the AD0IC register becomes 1 (interrupt requested).
- When an A/D conversion is halted by setting the ADST bit in the AD0CON0 register to 0, the converted result is undefined. In addition, the unconverted AD0i register may also become undefined. Consequently, the AD0i register should not be used just after A/D conversion is halted.
- External triggers cannot be used in DMAC operating mode. When the DMAC is configured to transfer converted results, do not read the AD00 register by a program.
- While in single sweep mode, if A/D conversion is halted by setting the ADST bit in the AD0CON0 register to 0 (A/D conversion is stopped), an interrupt request may be generated even though the sweep is not completed. To halt A/D conversion, disable interrupts before setting the ADST bit to 0.

## 19. CRC Calculator

The Cyclic Redundancy Check (CRC) calculator is used for detecting errors in data blocks. A generator polynomial of CRC-CCITT ( $X^{16} + X^{12} + X^5 + 1$ ) generates the CRC.

The CRC is a 16-bit code generated for a given set of blocks of 8-bit data. It is set in the CRCD register every time 1-byte data is written to the CRCIN register after a default value is set to the CRCD register.

Figure 19.1 shows a block diagram of the CRC calculator. Figures 19.2 and 19.3 show registers associated with the CRC. Figure 19.4 shows an example of the CRC calculation.

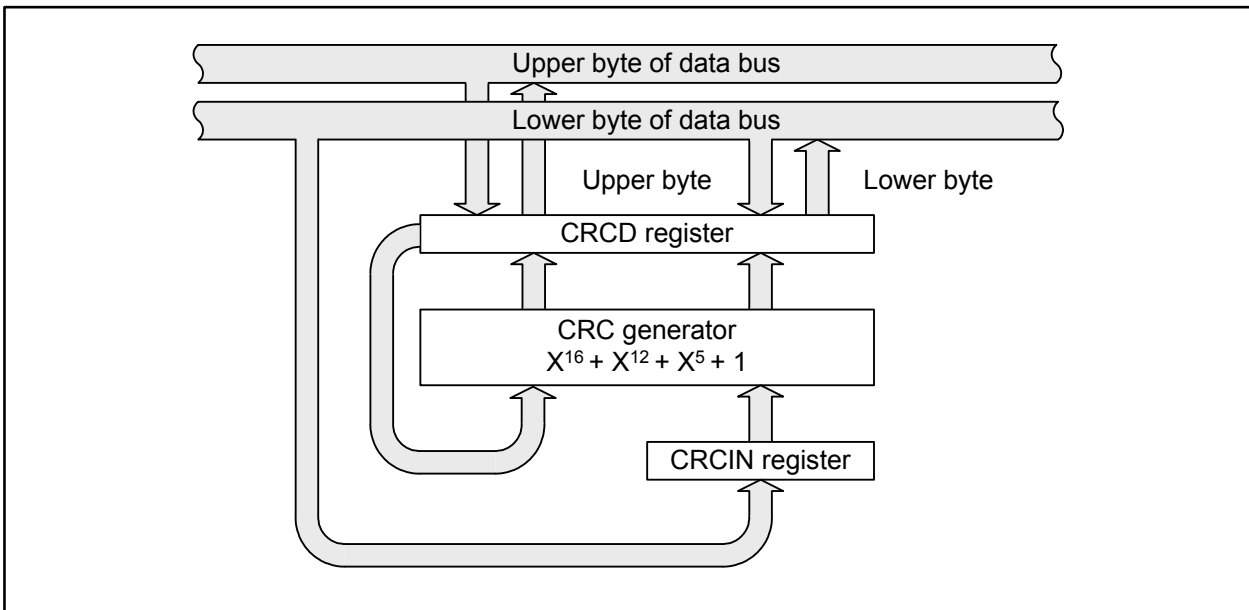


Figure 19.1 CRC Calculator Block Diagram

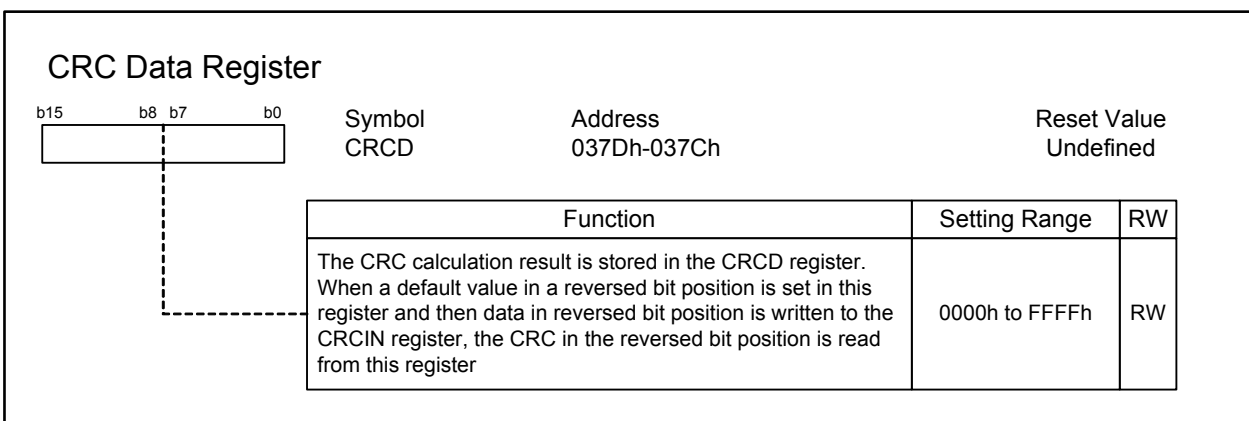
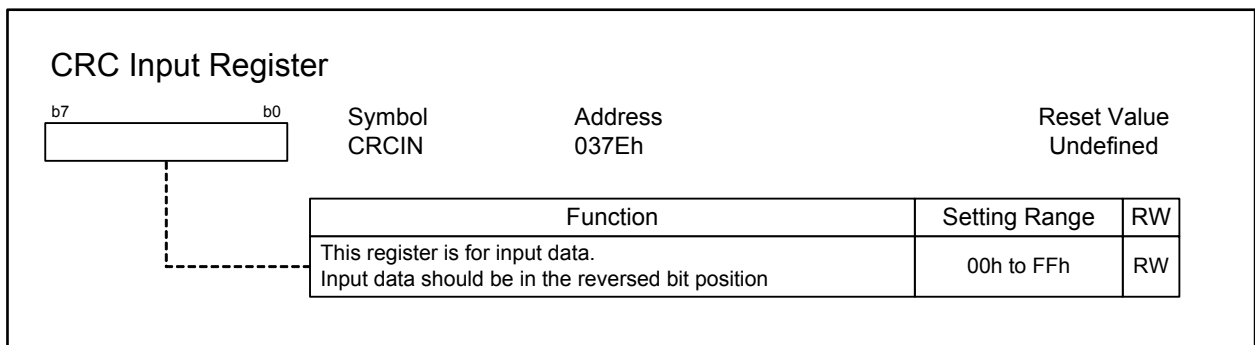


Figure 19.2 CRCD Register





**Figure 19.3 CRCIN Register**

### CRC Calculation and Setting Procedure to Generate CRC for 80C4h

- **CRC Calculation for R32C**

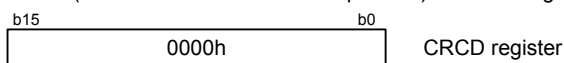
CRC: a remainder of the division as follows: 
$$\frac{\text{reversed-bit-position value in the CRCIN register}}{\text{generator polynomial}}$$

Generator Polynomial:  $X^{16} + X^{12} + X^5 + 1$  (1 0001 0000 0010 0001b)

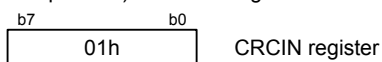
- **Setting Procedure**

(1) Reverse the bit position of 80C4h in 1-byte units by a program  
80h to 01h, C4h to 23h

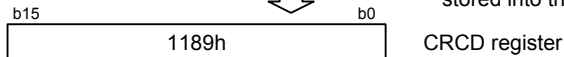
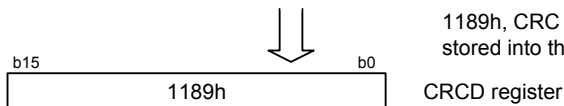
(2) Set 0000h (default value in reversed bit position) in CRCD register



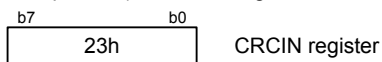
(3) Set 01h (80h in reversed bit position) in CRCIN register



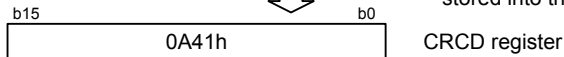
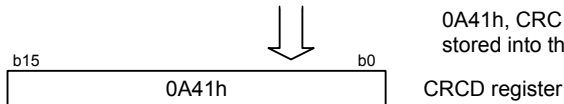
1189h, CRC for 80h (9188h) in reversed bit position is stored into the CRCD register in the third cycle.



(4) Set 23h (C4h in reversed bit position) in CRCIN register

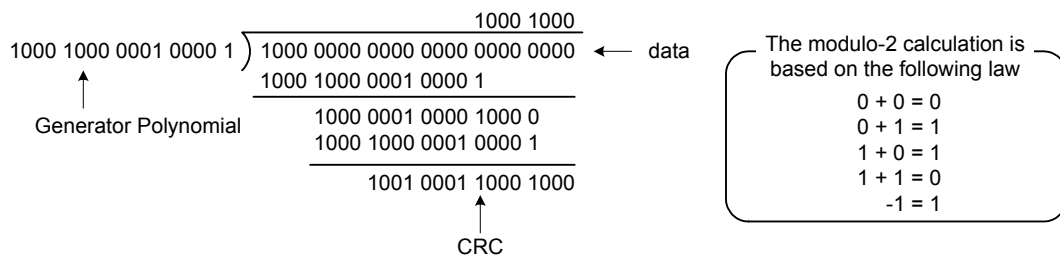


0A41h, CRC for 80C4h (8250h) in reversed bit position is stored into the CRCD register in the third cycle.



- **Details of the CRC Calculation**

As shown in (3) above, add 1000 0000 0000 0000 0000 0000b as 80h (1000 0000b) plus 16 digits to 0000 0000 0000 0000 0000 0000b as the default value of the CRCD register, 0000h plus eight digits to perform the modulo-2 division.



0001 0001 1000 1001b (1189h), the reversed-bit-position value of remainder 1001 0001 1000 1000b (9188h) can be read from the CRCD register.

When continuing on to (4) above, add 1100 0100 0000 0000 0000 0000b as C4h (1100 0100b) plus 16 digits to 1001 0001 1000 1000 0000 0000b as the remainder of (3) left in the CRCD register plus eight digits to perform the modulo-2 division. 0000 1010 0100 0001b (0A41h), the reversed-bit-position value of remainder 1000 0010 0101 0000b (8250h) can be read from the CRCD register.

Figure 19.4 CRC Calculation

## 20. X-Y Conversion

X-Y conversion rotates a 16 × 16-bit matrix data 90 degrees or reverses the bit position of 16-bit data.

X-Y conversion is set using the XYC register shown in Figure 20.1.

Data is written to the write-only XiR registers and converted data is read from the read-only YjR register (i = 0 to 15; j = 0 to 15). These registers are allocated to the same address. Figures 20.2 and 20.3 show registers XiR and YjR, respectively. A write/read access to registers XiR and YjR should be performed in 16-bit units from an even address. 8-bit access operation results are undefined.

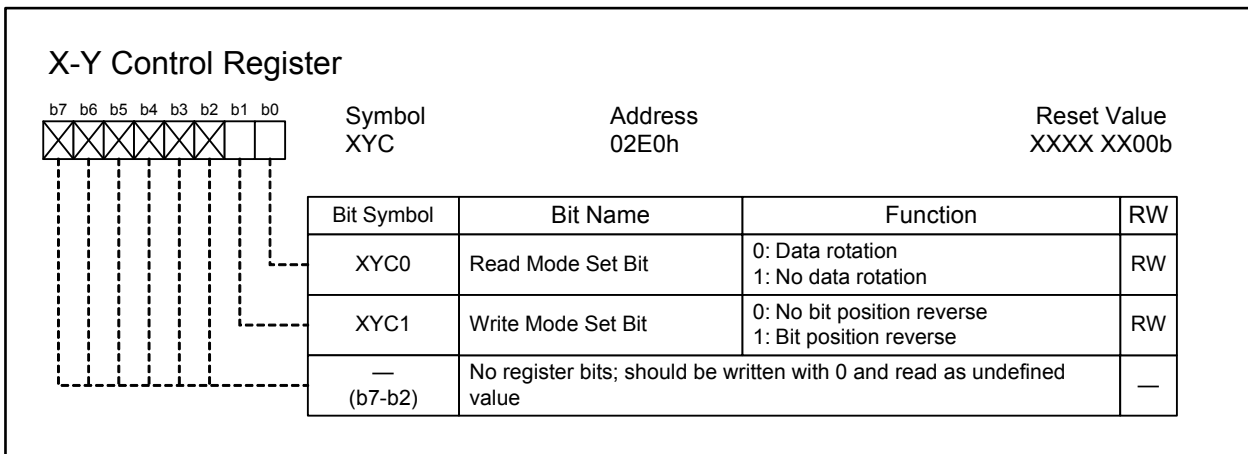


Figure 20.1 XYC Register

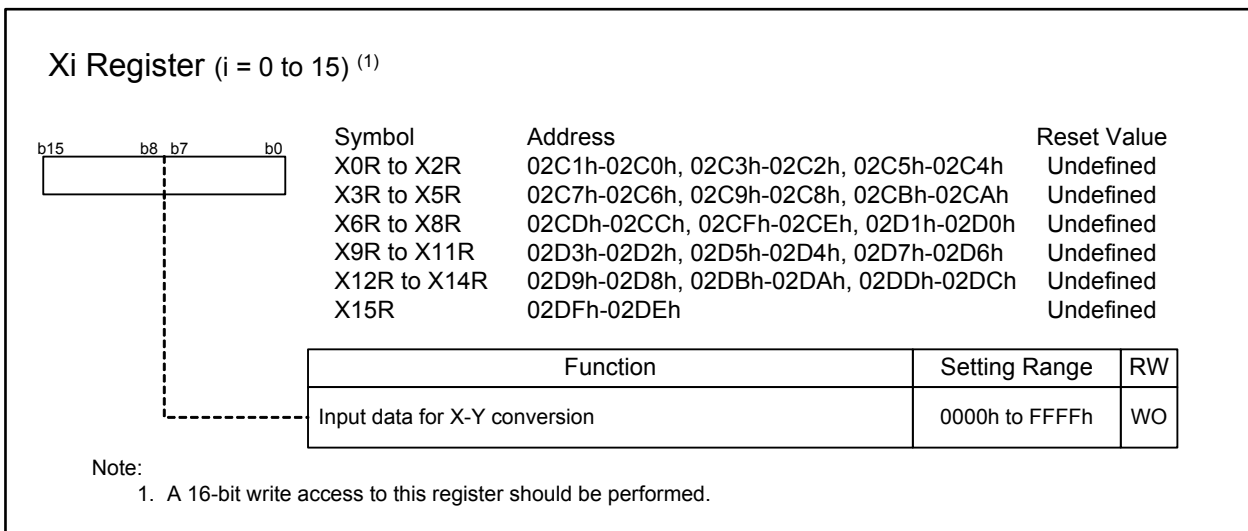
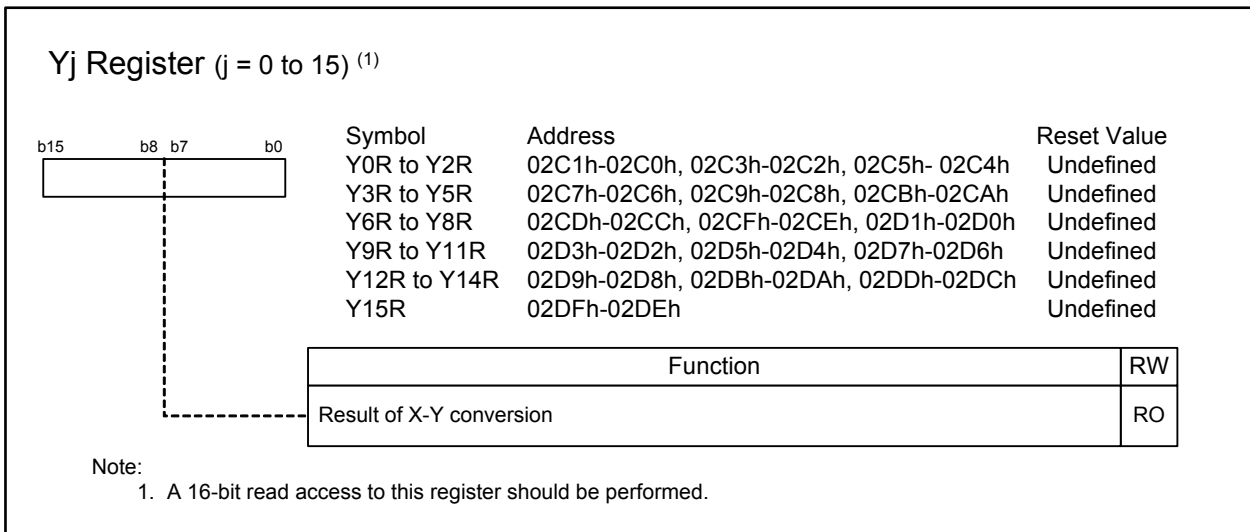


Figure 20.2 Registers X0R to X15R



**Figure 20.3 Registers Y0R to Y15R**

## 20.1 Data Conversion When Reading

Set the XYC0 bit in the XYC register to select a read mode for the Y<sub>j</sub>R register. When the XYC0 bit is 0 (data rotation), bit j in the corresponding registers X0R to X15R is automatically read upon reading the Y<sub>j</sub>R register (j = 0 to 15).

More concretely, upon reading bit i (i = 0 to 15) in the Y0R register, the data of bit 0 in the X<sub>i</sub>R register is read. That is, the read data of bit 0 in the Y15R register means the data of bit 15 in the X0R register and the data of bit 15 in the Y0R register is identical to that of bit 0 in the X15R register.

Figure 20.4 shows the conversion table when the XYC0 bit is 0 and Figure 20.5 shows an example of X-Y conversion.

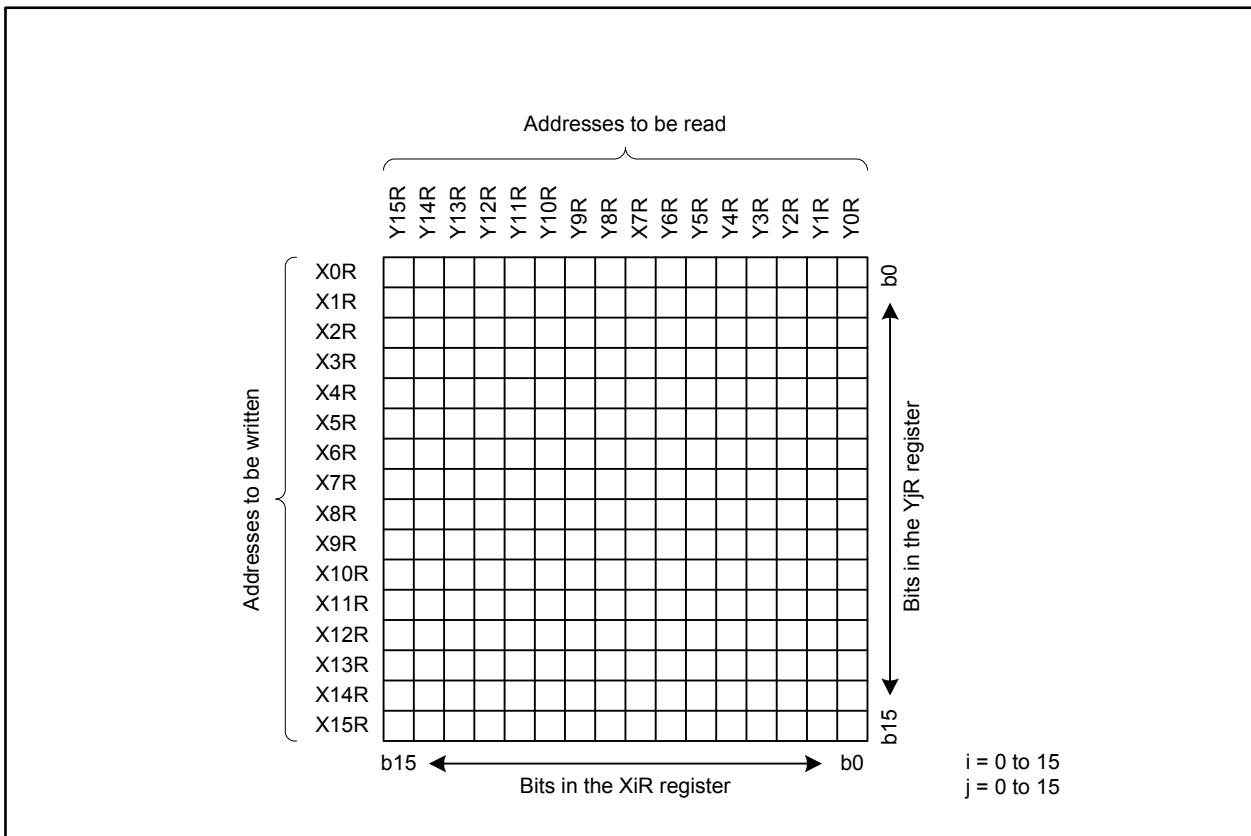


Figure 20.4 Conversion Table (XYC0 Bit is 0)

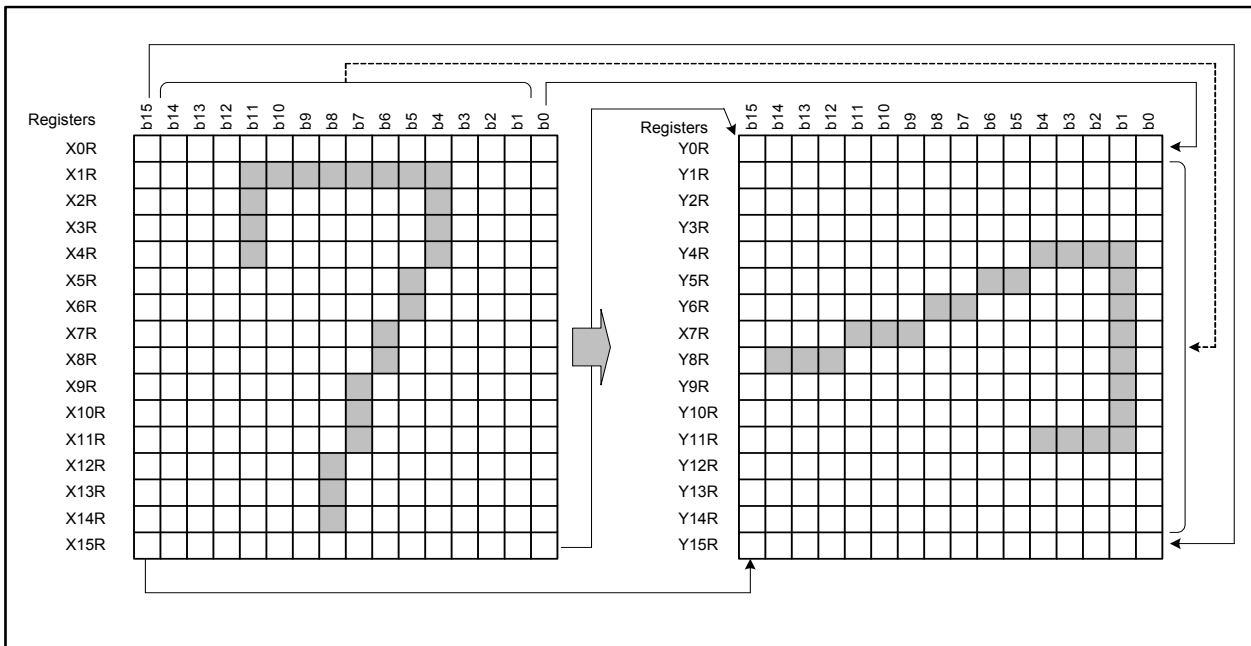


Figure 20.5 X-Y Conversion

When the  $XYC0$  bit is set to 1 (no data rotation), the data of each bit in the  $YjR$  register is identical to that written in the  $XiR$  register. Figure 20.6 shows the conversion table when the  $XYC0$  bit is set to 1.

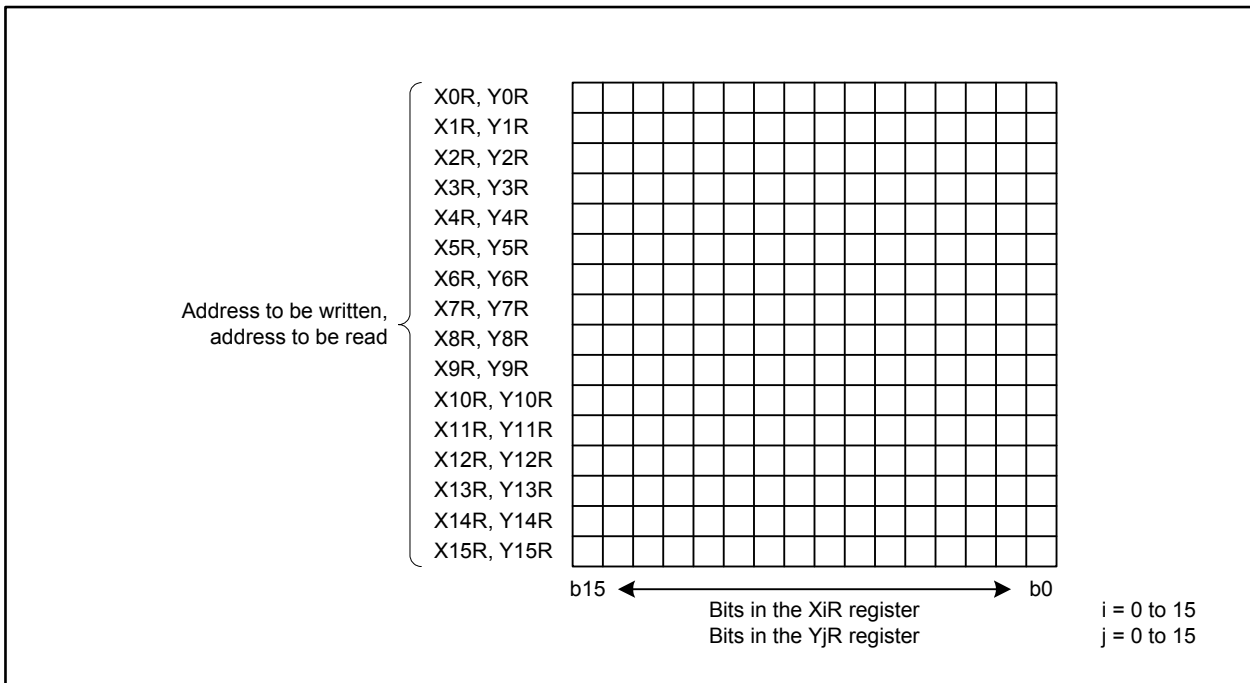


Figure 20.6 Conversion Table ( $XYC0$  Bit is 1)

### 20.2 Data Conversion When Writing

Set the  $XYC1$  bit in the  $XYC$  register to select a write mode for the  $XiR$  register.

When the  $XYC1$  bit is set to 0 (no bit position reverse), the data is written in order. When it is set to 1 (bit position reverse), the data is written in reversed order. Figure 20.7 shows the conversion table when the  $XYC1$  bit is set to 1.

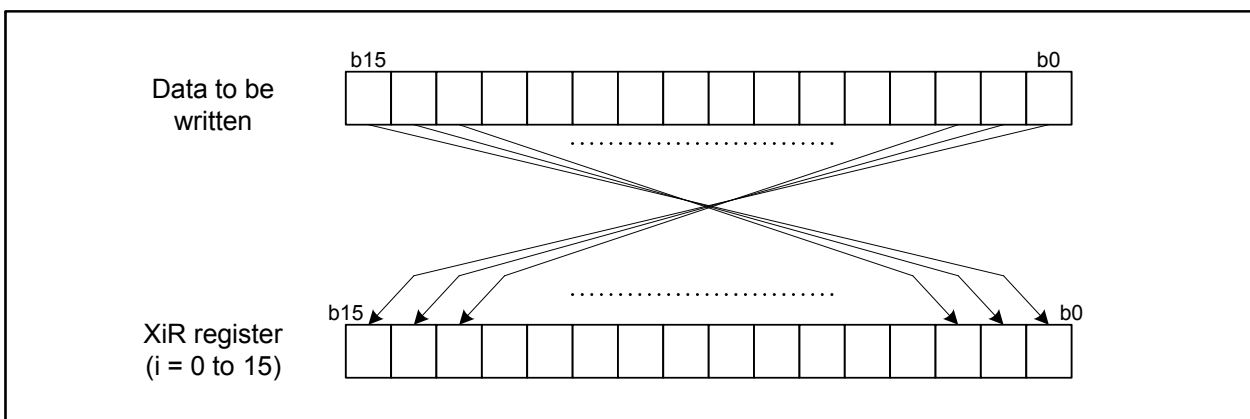


Figure 20.7 Conversion Table ( $XYC1$  Bit is 1)

## 21. Intelligent I/O

The intelligent I/O is a multifunctional I/O port for time measurement and waveform generation. It contains a module “group 0” which has one free-running 16-bit base timer and eight 16-bit registers for time measurement or waveform generation.

Table 21.1 lists the functions and channels of the intelligent I/O.

**Table 21.1 Intelligent I/O Functions and Channels**

	Functions	Group 0
Time measurement <sup>(1)</sup>	Digital filter	8 channels
	Prescaler	2 channels
	Gating	2 channels
	Digital debounce	1 channel
Waveform generation <sup>(1)</sup>	Single-phase waveform output mode	8 channels
	Inverted waveform output mode	8 channels
	SR waveform output mode	8 channels
	Phase shift waveform output mode	8 channels

Note:

1. The time measurement and waveform generation functions share a pin.

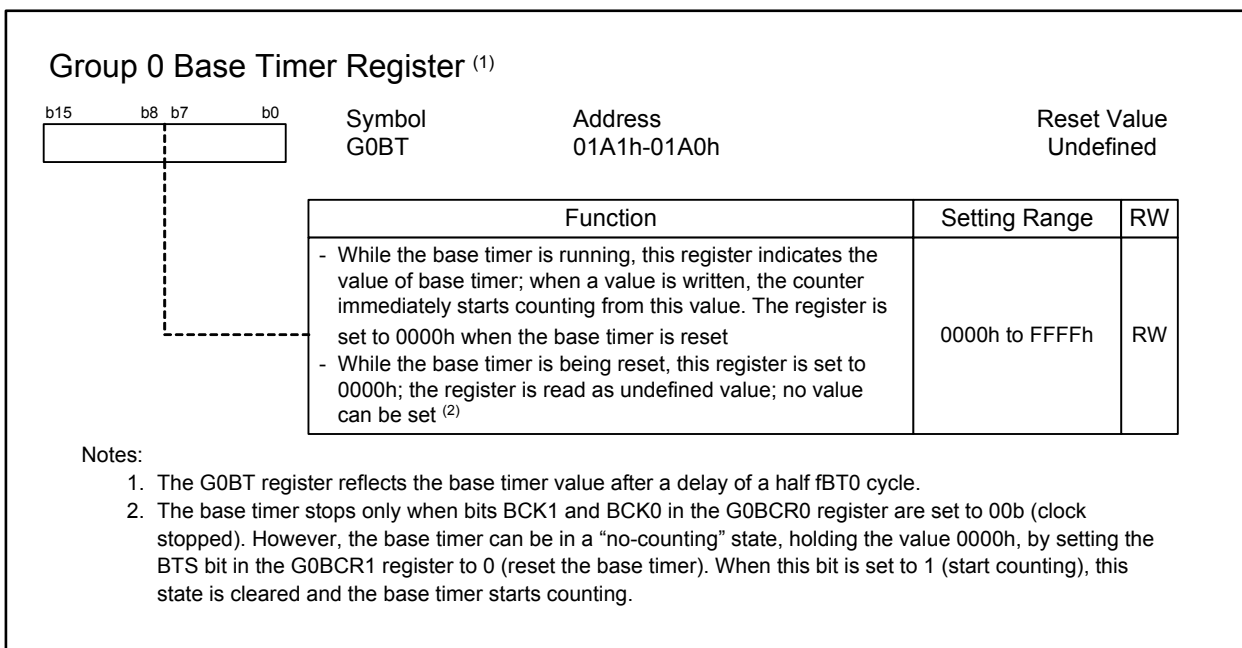
Each channel can be individually assigned for time measurement or waveform generation function.

Figure 21.1 shows a block diagram of the intelligent I/O.





Figures 21.2 to 21.13 show registers associated with the intelligent I/O base timer, time measurement, and waveform generation.



**Figure 21.2 G0BT Register**

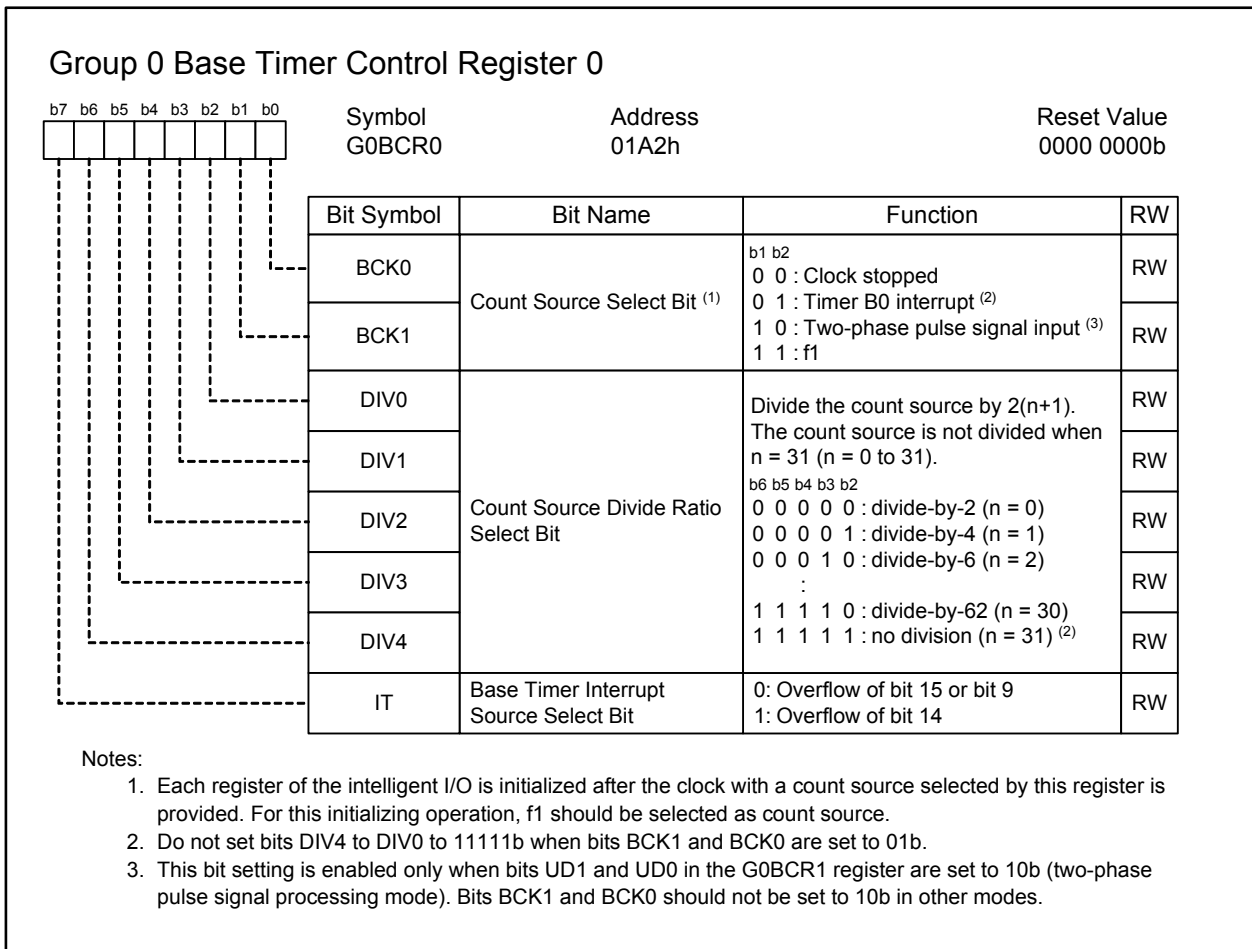
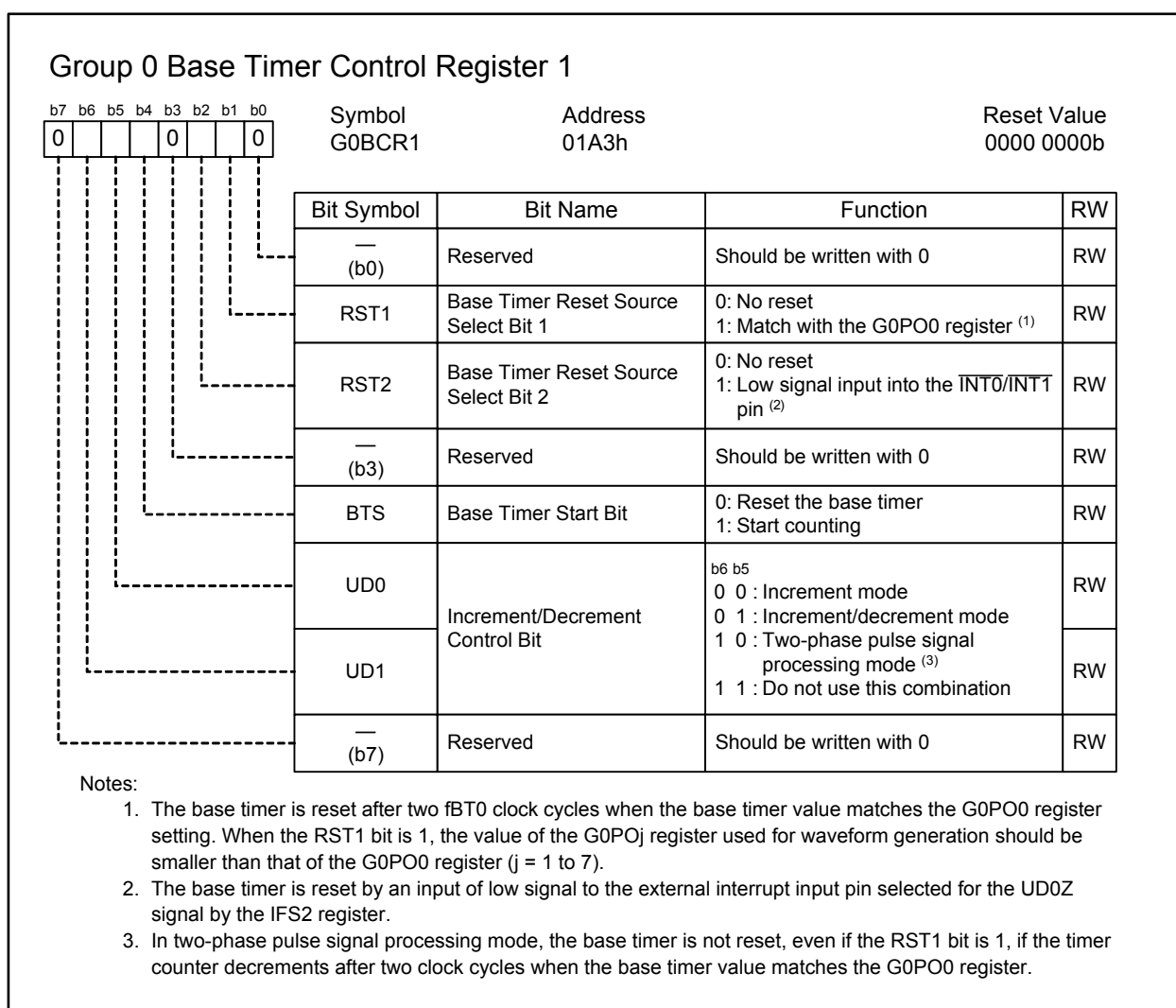


Figure 21.3 G0BCR0 Register



**Figure 21.4 G0BCR1 Register**

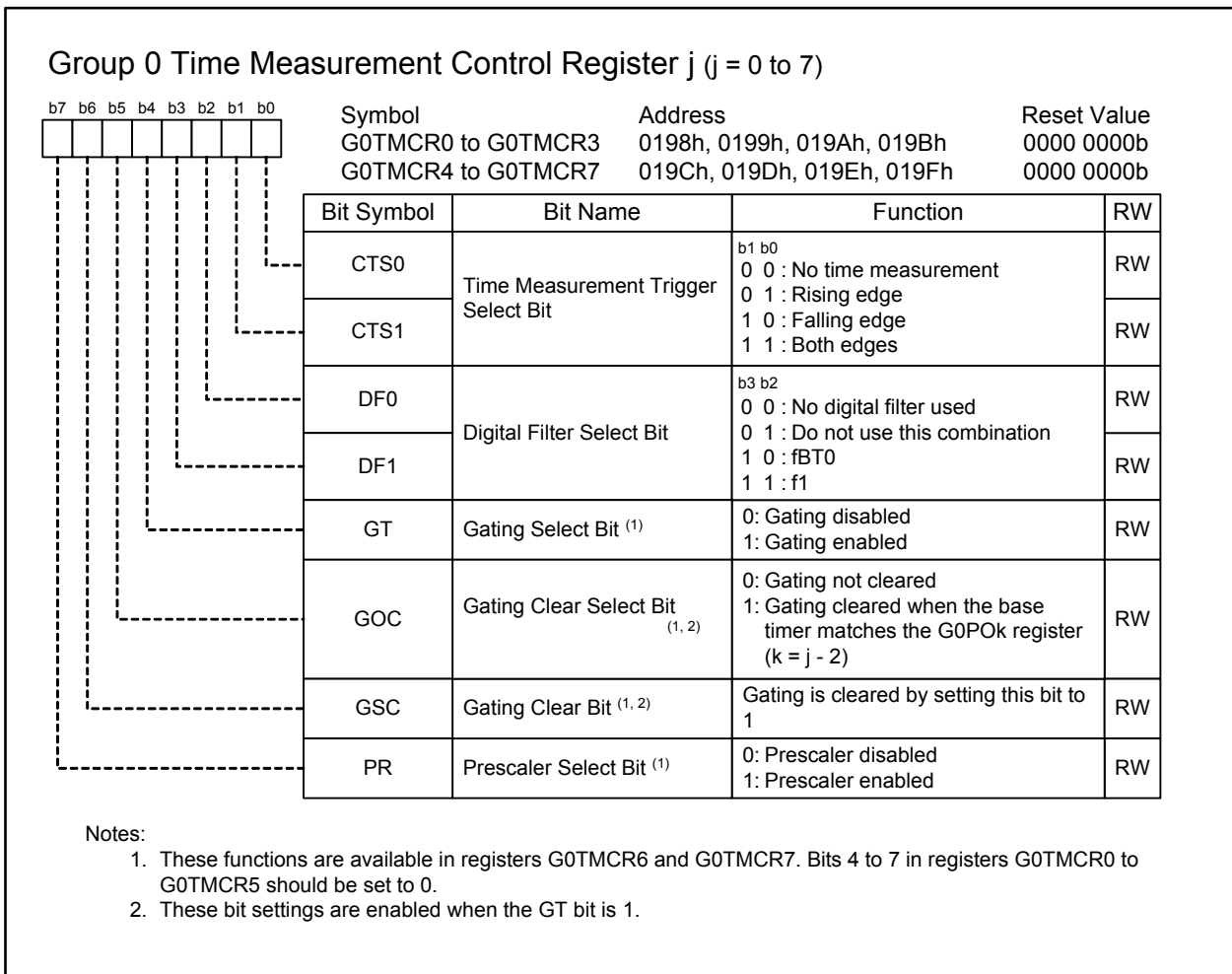


Figure 21.5 Registers G0TMCR0 to G0TMCR7

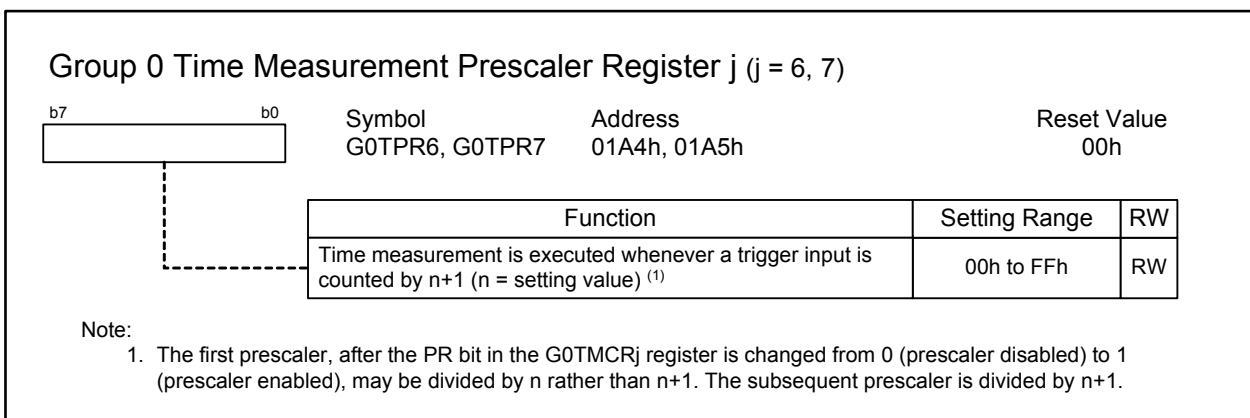


Figure 21.6 Registers G0TPR6 and G0TPR7

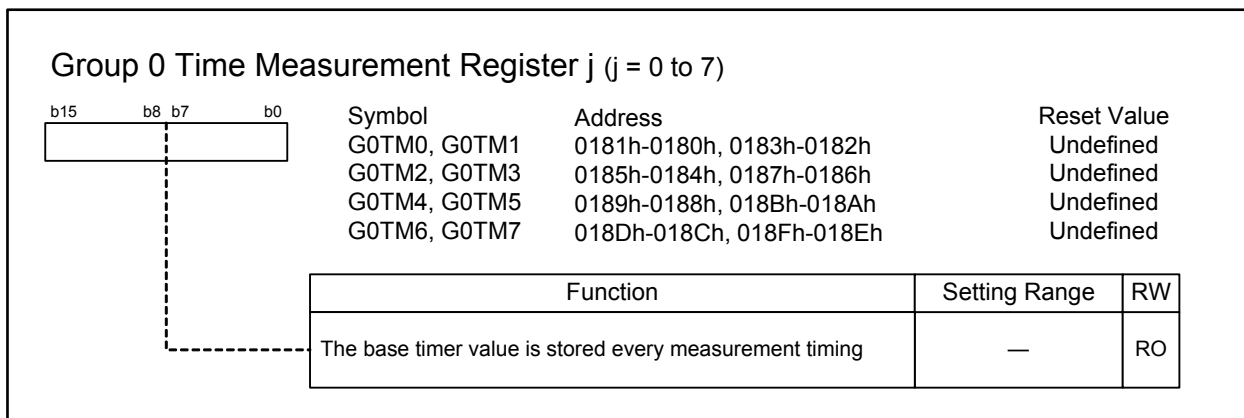
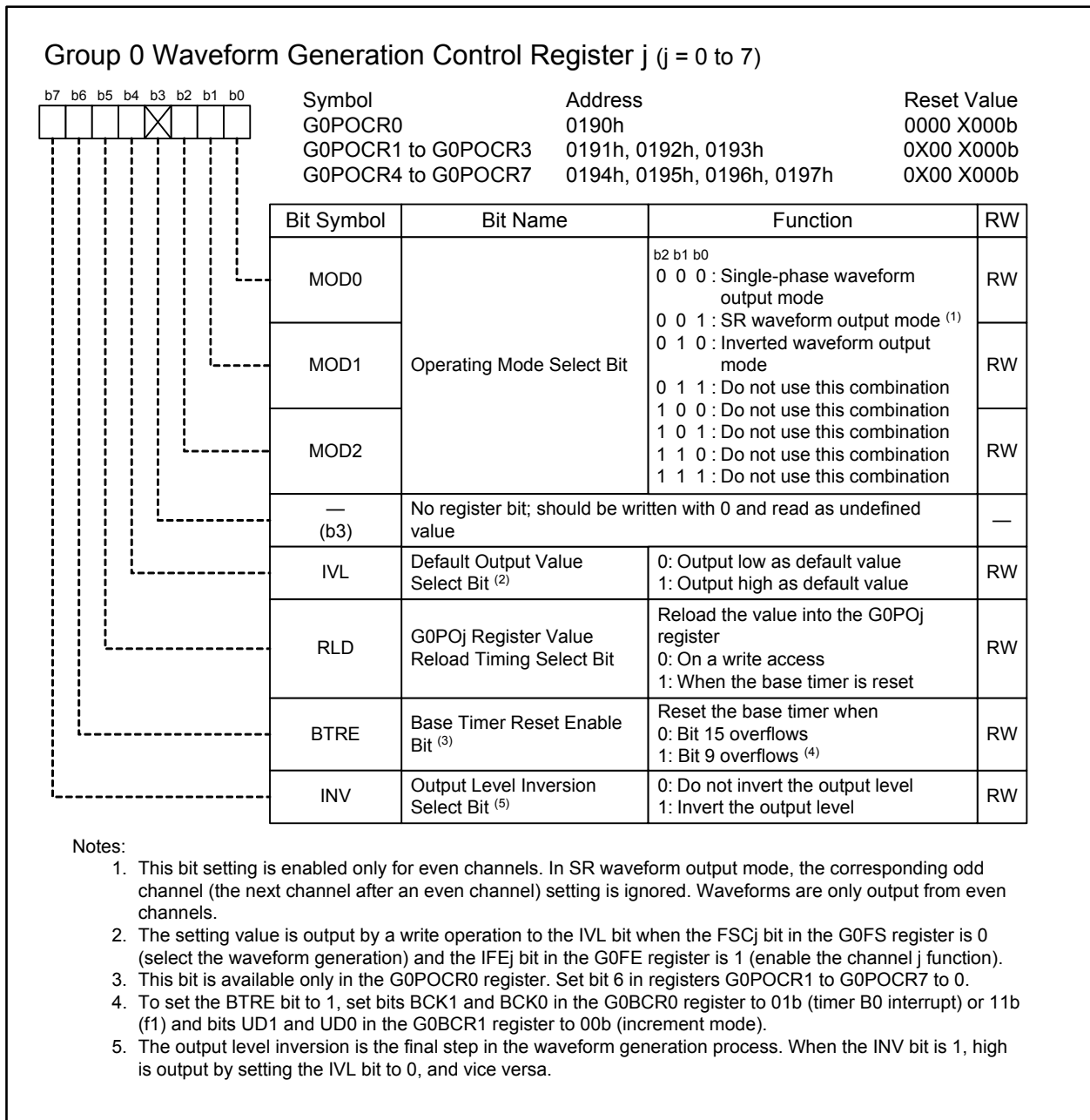


Figure 21.7 Registers G0TM0 to G0TM7



**Figure 21.8 Registers G0POCR0 to G0POCR7**

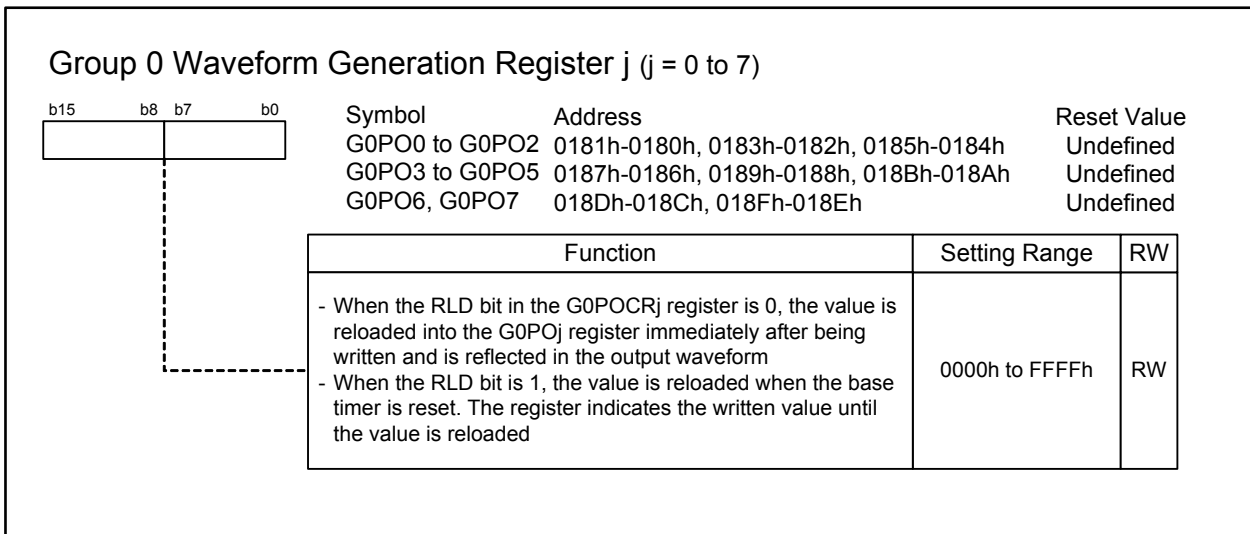


Figure 21.9 Registers G0PO0 to G0PO7

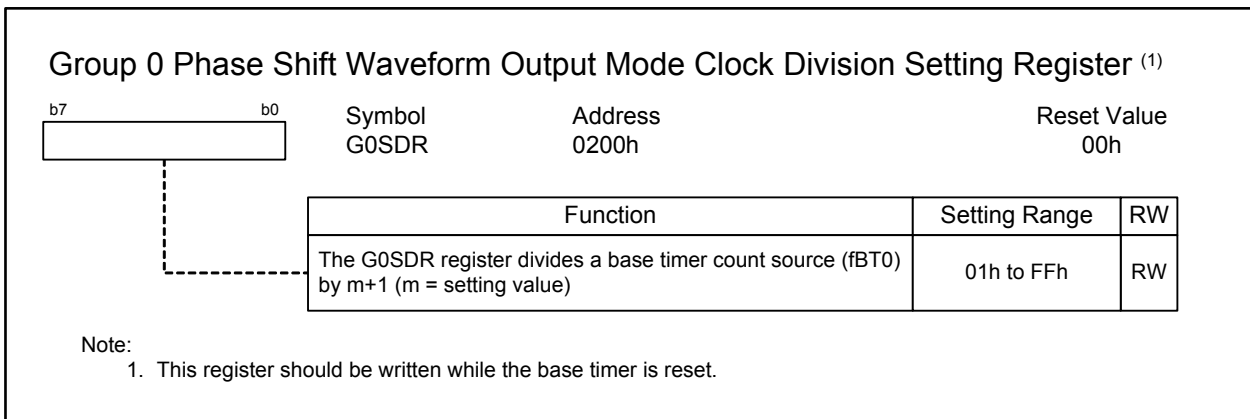
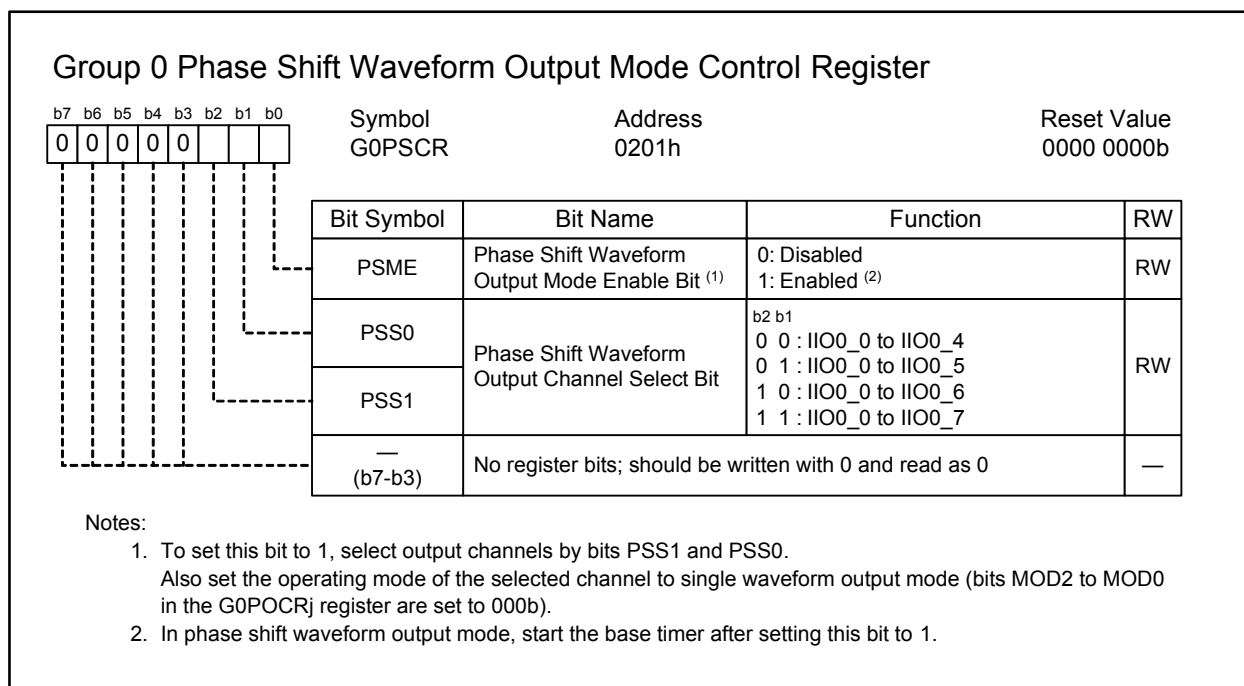


Figure 21.10 G0SDR Register

**Figure 21.11 G0PSCR Register**



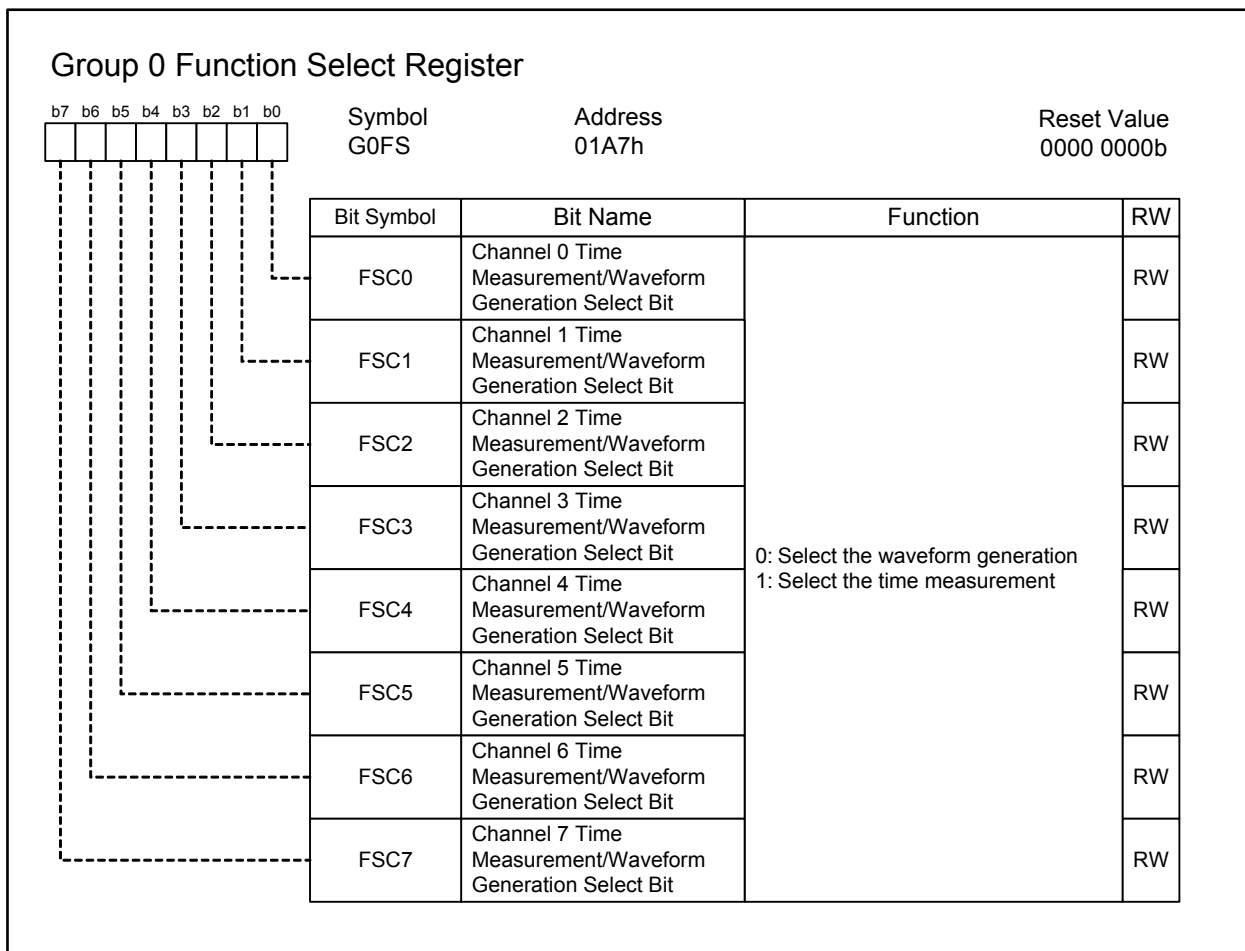
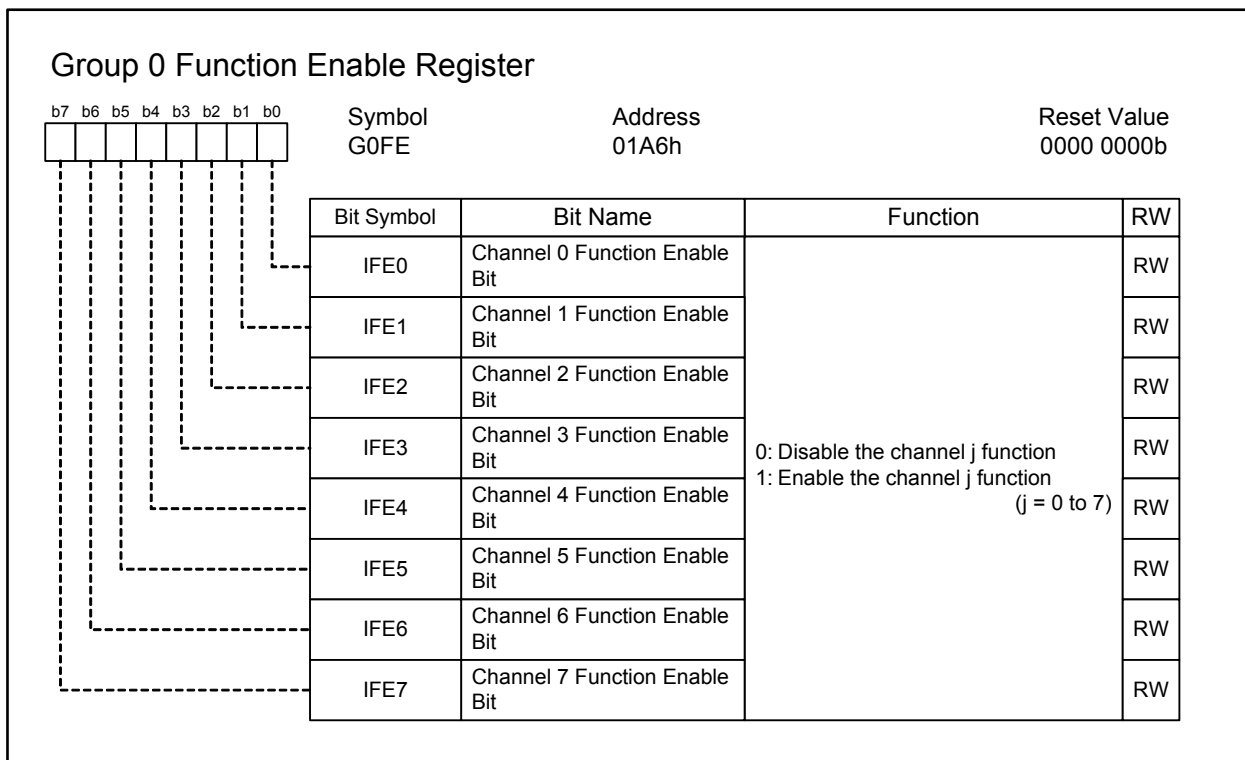


Figure 21.12 G0FS Register



**Figure 21.13 G0FE Register**

## 21.1 Base Timer

The base timer is a free-running counter that counts an internally generated count source. Table 21.2 lists specifications of the base timer. Figures 21.2 to 21.13 show registers associated with the base timer. Figure 21.14 shows a block diagram of the base timer. Figures 21.15, 21.16, and 21.17 show operation examples of the base timer in increment mode, increment/decrement mode, and two-phase pulse signal processing mode, respectively.

**Table 21.2 Base Timer Specifications**

Item	Specification
Count source (fBT0)	f1 divided by $2^{(n+1)}$ , two-phase pulse input divided by $2^{(n+1)}$ n: setting value using bits DIV4 to DIV0 in the G0BCR0 register n = 0 to 31; however no division when n = 31 Timer B0 interrupt divided by $2^{(m+1)}$ m: setting value using bits DIV4 to DIV0 in the G0BCR0 register m = 0 to 30
Count operations	<ul style="list-style-type: none"> <li>• Increment</li> <li>• Increment/decrement</li> <li>• Two-phase pulse signal processing</li> </ul>
Count start conditions	The BTS bit in the G0BCR1 register is 1 (start counting)
Count stop condition	The BTS bit in the G0BCR1 register is 0 (reset the base timer)
Reset conditions	<ul style="list-style-type: none"> <li>• The base timer value matches the G0PO0 register setting</li> <li>• An input of low signal into the external interrupt pin (<math>\overline{INT0}</math> or <math>\overline{INT1}</math>) as follows:  for group 0: selected using the IFS22 bit in the IFS2 register</li> <li>• The overflow of bit 15 or bit 9 in the base timer</li> </ul>
Reset value	0000h
Interrupt request	When the BT0R bit in the interrupt request register becomes 1 (interrupt requested) by the overflow of bit 9, 14, or 15 in the base timer (refer to Figure 10.12)
Read from base timer	<ul style="list-style-type: none"> <li>• The G0BT register indicates a counter value while the base timer is running</li> <li>• The G0BT register is undefined while the base timer is being reset</li> </ul>
Write to base timer	When a value is written while the base timer is running, the timer counter immediately starts counting from this value. No value can be written while the base timer is being reset
Other functions	<ul style="list-style-type: none"> <li>• Increment/decrement mode  The base timer starts counting when the BTS bit is set to 1. When the base timer reaches FFFFh, it starts decrementing. When the RST1 bit in the G0BCR1 register is 1 (the base timer is reset by matching with the G0PO0 register), the timer counter starts decrementing two counts after the base timer value matches the G0PO0 register setting. When the timer counter reaches 0000h, it starts incrementing again (refer to Figure 21.16).</li> <li>• Two-phase pulse signal processing mode  Two-phase pulse signals at pins UD0A and UD0B are counted (refer to Figure 21.17).</li> </ul> <div style="text-align: center;"> <p>The timer counter increments on all edges      The timer counter decrements on all edges</p> </div>

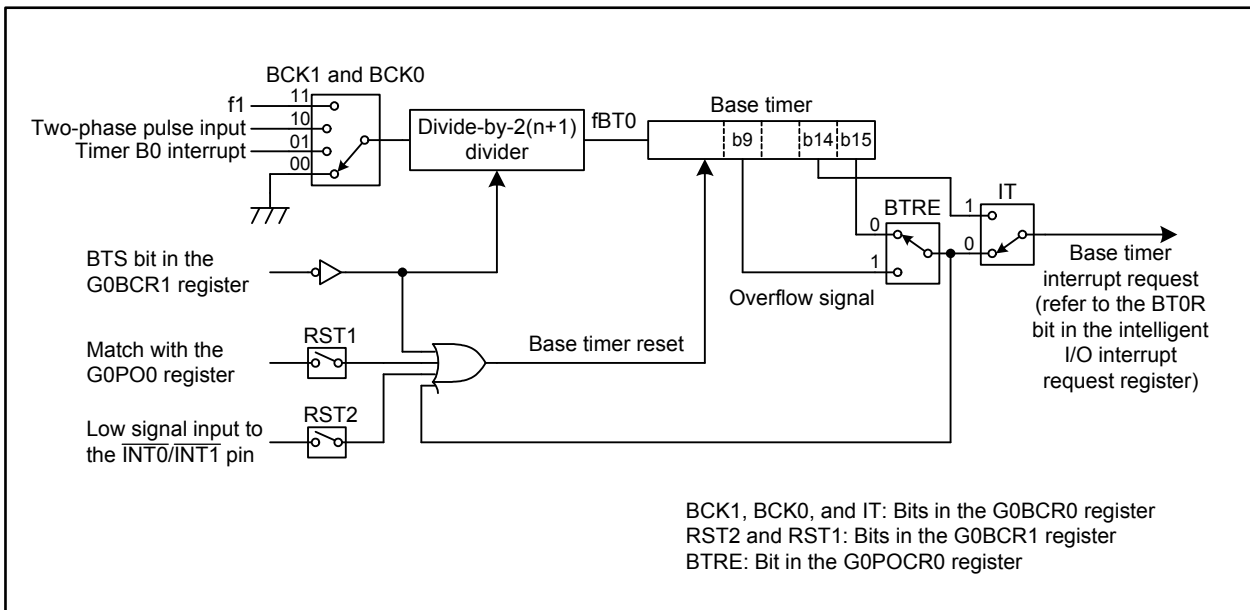


Figure 21.14 Base Timer Block Diagram

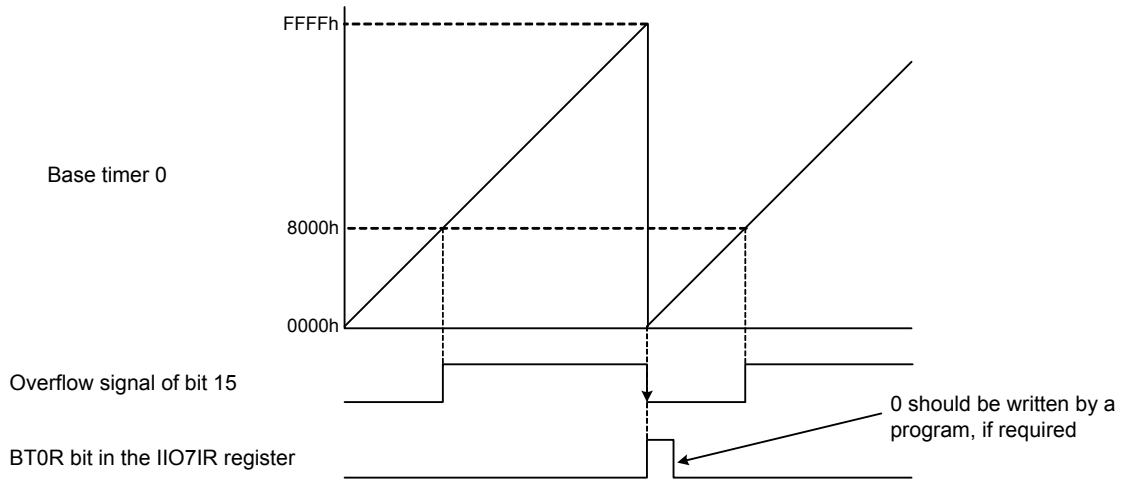
Table 21.3 Base Timer Associated Register Settings (Common Settings for Time Measurement, Waveform Generation, and Serial Interface)

Register	Bits	Function
G0BCR0	BCK1 and BCK0	Select a count source
	DIV4 to DIV0	Select a count source divide ratio
	IT	Select a base timer interrupt source
G0BCR1	RST2 and RST1	Select a timing for base timer reset
	BTS	Use this bit when each base timer individually starts counting
	UD1 and UD0	Select a count mode
G0PCR0	BTRE	Select a source for base timer reset
G0BT	—	Read or write the base timer value

The following register settings are required to set the RST1 bit to 1 (the base timer is reset by matching with the G0PO0 register).

G0PCR0	MOD2 to MOD0	Set to 000b (single-phase waveform output mode)
G0PO0	—	Set the reset cycle
G0FS	FSC0	Set the bit to 0 (select the waveform generation)
G0FE	IFE0	Set the bit to 1 (channel operation starts)

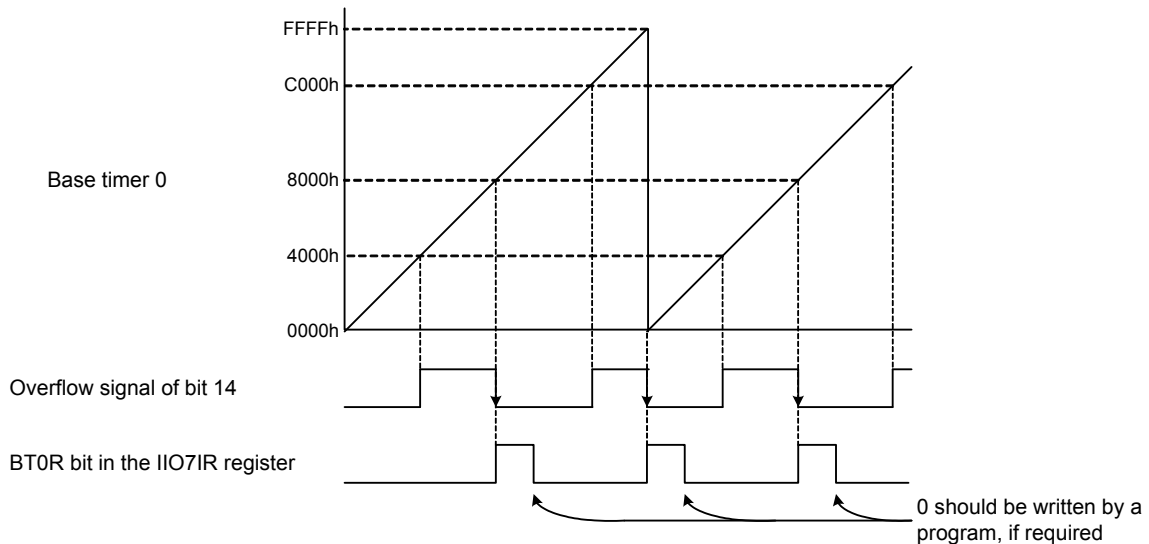
(A) When the IT bit in the G0BCR0 register is 0  
(an interrupt is requested by the overflow of bit 15 in the base timer)



This figure applies under the following conditions:

- The RST1 bit in the G0BCR1 register is 0 (the match with the G0PO0 register is not the reset source for the base timer)
- Bits UD1 and UD0 in the G0BCR1 register are 00b (increment mode)

(B) When the IT bit in the G0BCR0 register is 1  
(an interrupt is requested by the overflow of bit 14 in the base timer)



This figure applies under the following conditions:

- The RST1 bit in the G0BCR1 register is 0 (the match with the G0PO0 register is not the reset source for the base timer)
- Bits UD1 and UD0 in the G0BCR1 register are 00b (increment mode)

**Figure 21.15 Base Timer Increment Mode**

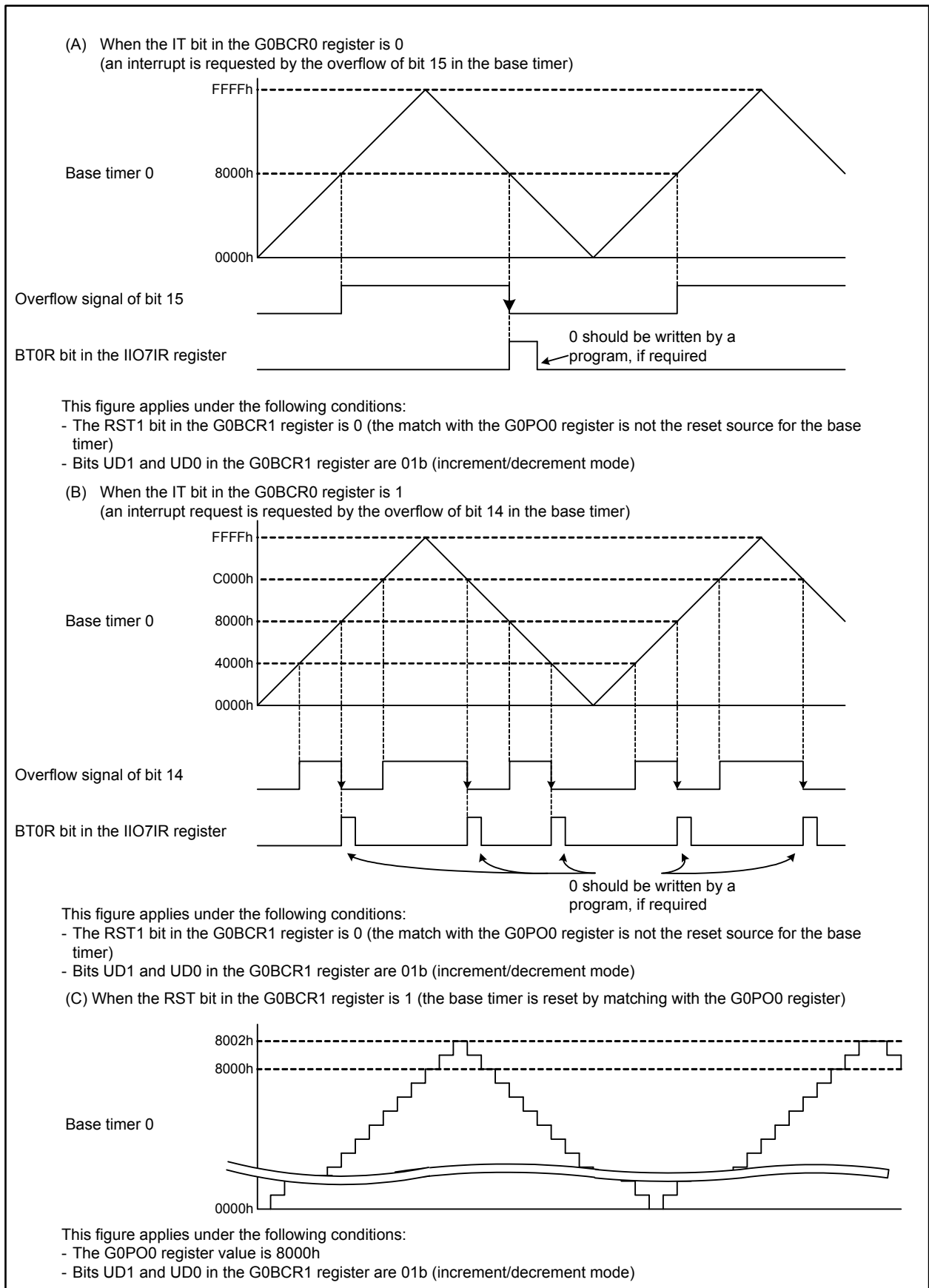


Figure 21.16 Base Timer Increment/Decrement

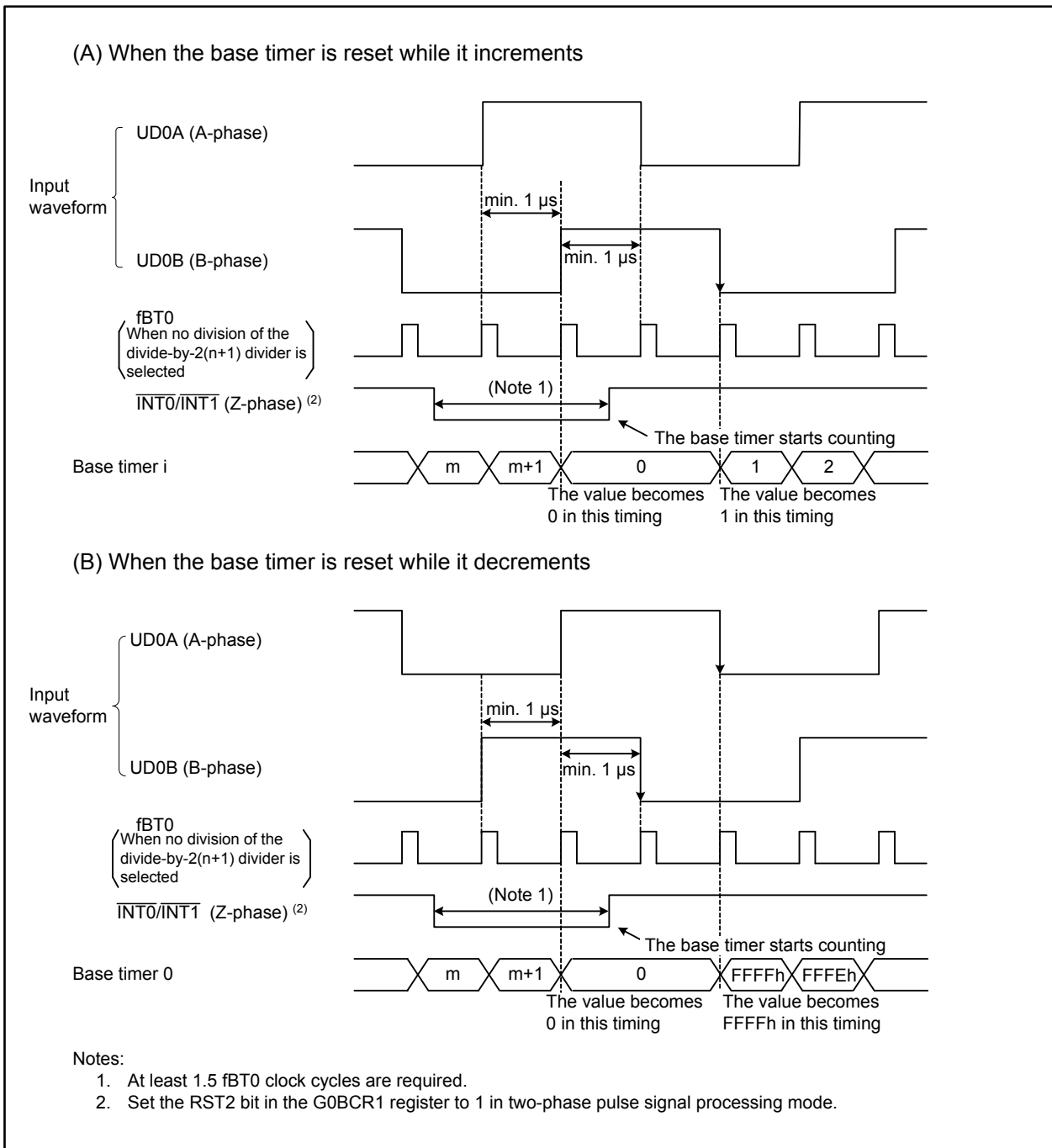


Figure 21.17 Base Timer Two-phase Pulse Signal Processing Mode

## 21.2 Time Measurement

Every time an external trigger is input, the base timer value is stored into the G0TMj register (j = 0 to 7). Table 21.4 lists specifications of the time measurement and Table 21.5 lists its register settings. Figures 21.18 and 21.19 show operation examples of the time measurement and Figure 21.20 shows operation examples with the prescaler or gate function.

**Table 21.4 Time Measurement Specifications (j = 0 to 7)**

Item	Specification
Time measurement channels	Group 0: Channels 0 to 7
Trigger input polarity	Rising edge, falling edge, or both edges of the IIO0_j pin
Time measurement start condition	The IFEj bit in the G0FE register is 1 (enable the channel j function) while the FSCj bit in the G0FS register is 1 (select the time measurement)
Time measurement stop condition	The IFEj bit is 0 (disable the channel j function)
Time measurement timing	<ul style="list-style-type: none"> <li>• Without the prescaler: every time a trigger is input</li> <li>• With the prescaler for channels 6 and 7: every [G0TPRk register value + 1] times a trigger is input (k = 6, 7)</li> </ul>
Interrupt request	When the TM0jR bit in the interrupt request register becomes 1 (interrupt requested) (refer to Figure 10.12)
IIO0_j input pin function	Trigger input
Other functions	<ul style="list-style-type: none"> <li>• Digital filter The digital filter determines a trigger input level every f1 or fBT0 cycle and passes the signals holding the same level during three sequential cycles</li> <li>• Prescaler for channels 6 and 7 Time measurement is executed every [G0TPRk register value + 1] times a trigger is input</li> <li>• Gating for channels 6 and 7 This function disables any trigger input to be accepted after the time measurement by the first trigger input. However, the trigger input can be accepted again if any of following conditions are met while the GOC bit in the G0TMCRk register is 1 (the gating is cleared when the base timer matches the G0POp register) (p = 4, 5; p = 4 when k = 6; p = 5 when k = 7): <ul style="list-style-type: none"> <li>• The base timer value matches the G0POp register setting</li> <li>• The GSC bit in the G0TMCRk register is 1</li> </ul> </li> </ul>

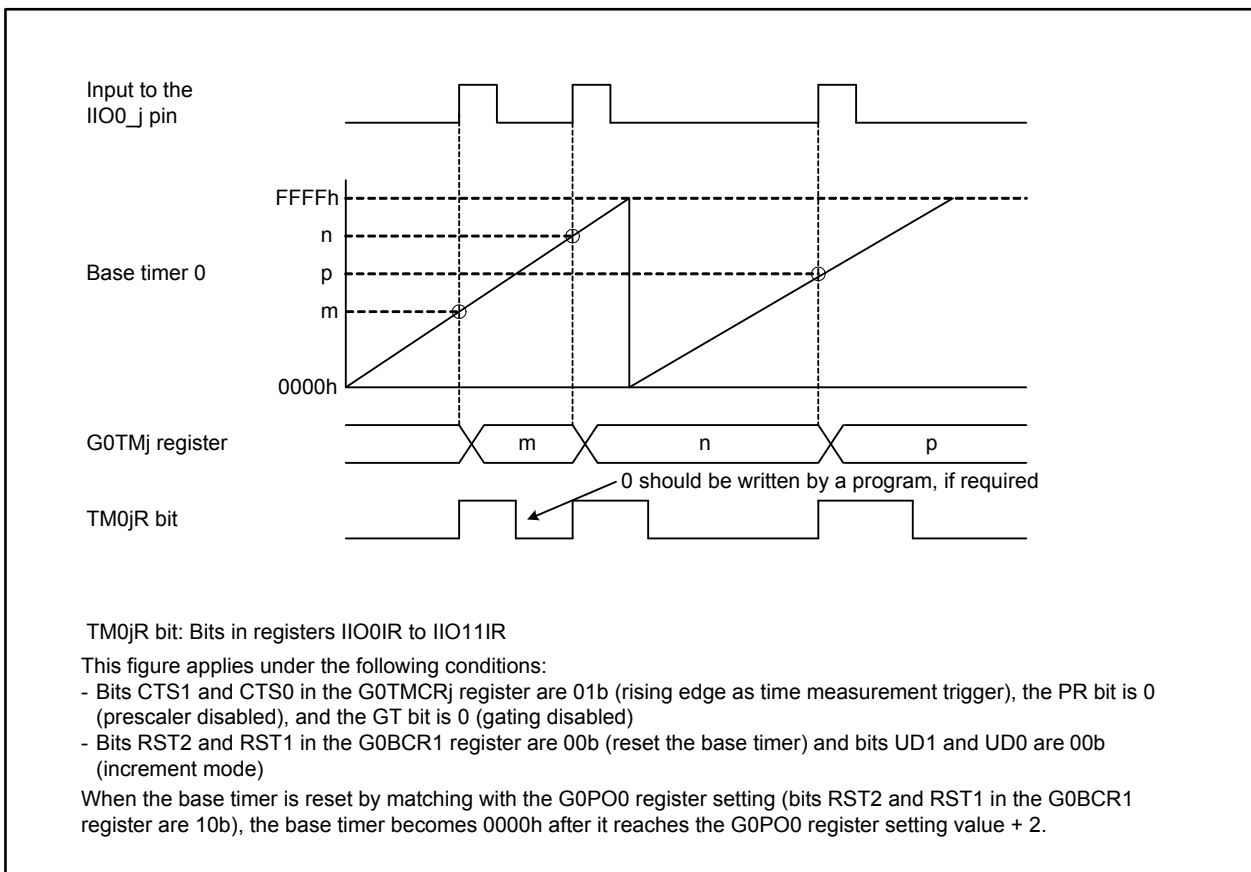


**Table 21.5 Time Measurement Associated Register Settings (j = 0 to 7; k = 6, 7)**

Register	Bits	Function
G0TMCRj	CTS1 and CTS0	Select a time measurement trigger
	DF1 and DF0	Select a digital filter
	GT, GOC, GSC	Select if the gating is used
	PR	Select if the prescaler is used
G0TPRk	—	Set the prescaler value
G0FS	FSCj	Set the bit to 1 (select the time measurement)
G0FE	IFEj	Set the bit to 1 (enable the channel j function)

Bit configurations and functions vary with channels.

Registers associated with the time measurement should be set after setting the base timer-associated registers.

**Figure 21.18 Time Measurement Operation (1/2) (j = 0 to 7)**

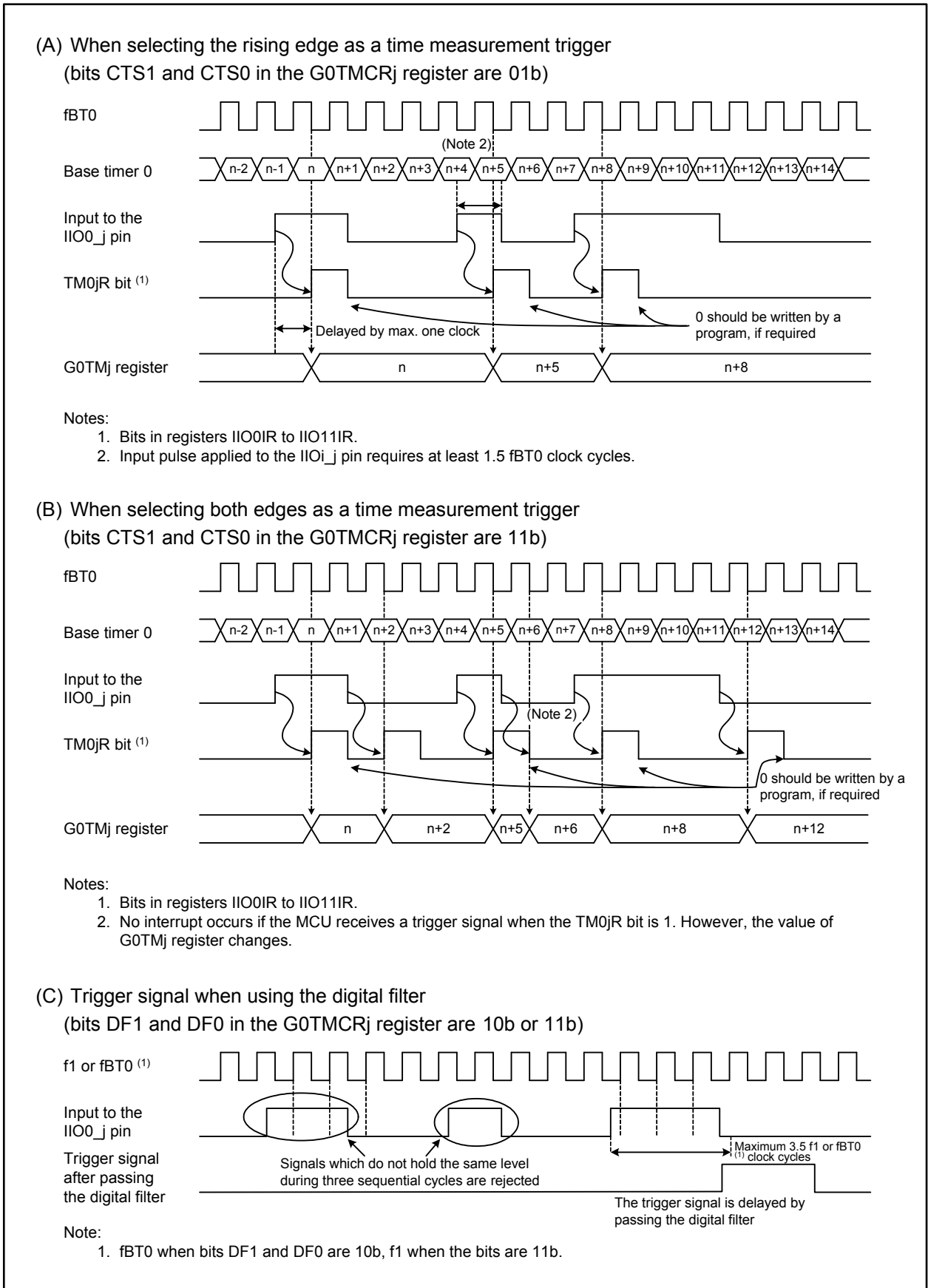
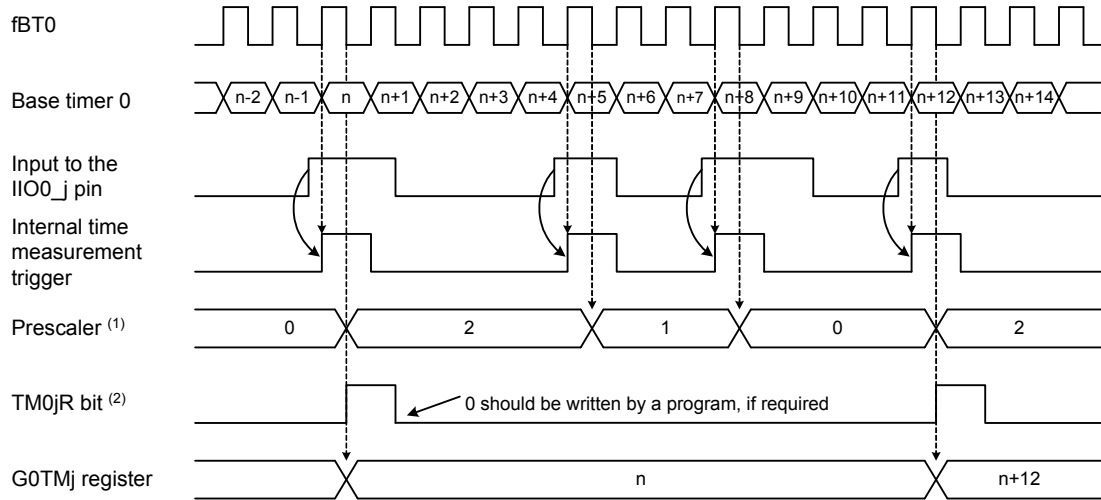


Figure 21.19 Time Measurement Operation (2/2) (j = 0 to 7)

(A) Operation with the prescaler

(the GOTPRj register is 02h and the PR bit in the GOTMCRj register is 1)

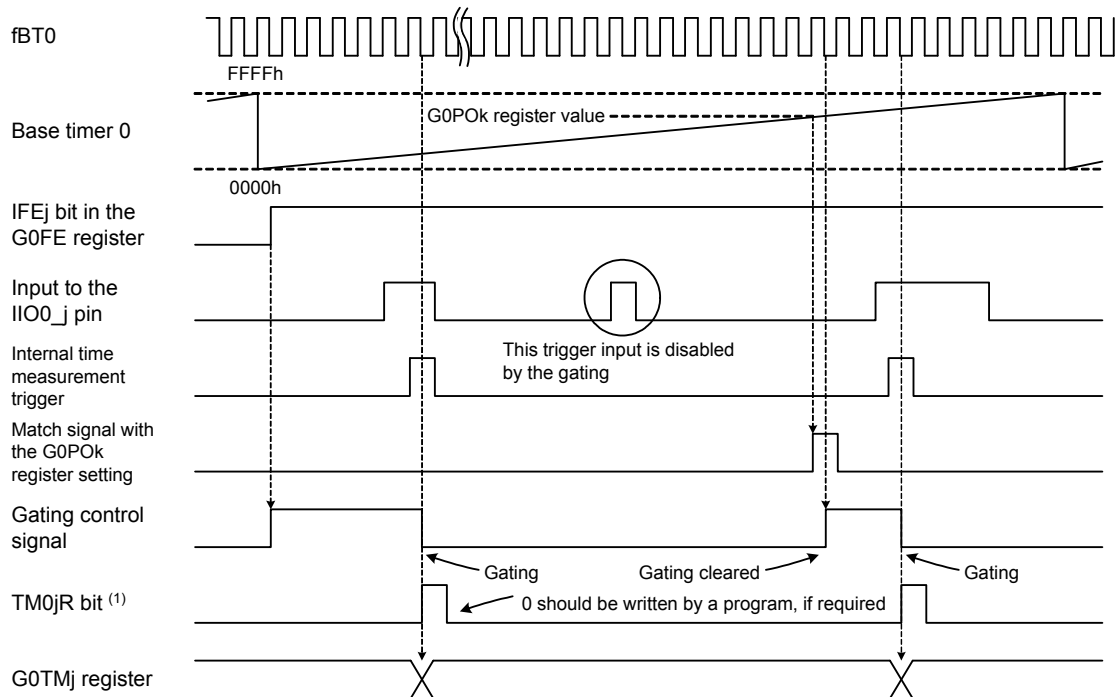


Notes:

1. This example applies to cycles following the first cycle after the PR bit in the GOTMCRj register is set to 1 (prescaler enabled).
2. Bits in registers IIO0IR to IIO11IR.

(B) Operation with the gating

(the gating is cleared by matching the base timer value with the G0POk register setting, and bits GT and GOC in the GOTMCRj register are 1, respectively)



Note:

1. Bits in registers IIO0IR to IIO11IR.

Figure 21.20 Prescaler and Gate Operations (j = 6, 7; k = 4, 5)

### 21.3 Waveform Generation

Waveforms are generated when the base timer value matches the G0POj register setting (j = 0 to 7).

Waveform generation has the following four modes:

- Single-phase waveform output mode
- Inverted waveform output mode
- Set/reset waveform output (SR waveform output) mode
- Phase shift waveform output mode

Table 21.6 lists registers associated with the waveform generation.

**Table 21.6 Waveform Generation Associated Register Settings (j = 0 to 7)**

Register	Bits	Function
G0POCRj	MOD2 to MOD0	Select a waveform output mode
	IVL	Select a default value
	RLD	Select a timing to reload the value into the G0POj register
	INV	Select if output level is inverted
G0POj	—	Set the timing to invert output waveform level
G0FS	FSCj	Set the bit to 0 (select the waveform generation)
G0FE	IFEj	Set the bit to 1 (enable the channel j function)

Bit configurations and functions vary with channels.

Registers associated with the waveform generation should be set after setting the base timer-associated registers.

### 21.3.1 Single-phase Waveform Output Mode

The output level at the IIO0\_j pin becomes high when the base timer value matches the G0POj register (j = 0 to 7). It switches to low when the base timer reaches 0000h. If the IVL bit in the G0POCRj register is set to 1 (output high as default value), a high level output is provided when a waveform output starts. If the INV bit is set to 1 (invert the output level), a waveform with an inverted level is output. Refer to Figure 21.21 for details on single-phase waveform mode operation.

Table 21.7 lists specifications of single-phase waveform output mode.

**Table 21.7 Single-phase Waveform Output Mode Specifications**

Item	Specification
Output waveform (1)	<ul style="list-style-type: none"> <li>Free-running operation (when bits RST2 and RST1 in the G0BCR1 register are 00b)               <ul style="list-style-type: none"> <li>Cycle: <math>\frac{65536}{f_{BT0}}</math></li> <li>Low level width: <math>\frac{m}{f_{BT0}}</math></li> <li>High level width: <math>\frac{65536 - m}{f_{BT0}}</math></li> </ul> </li> <li><i>m</i>: G0POj register setting value (j = 0 to 7), 0000h to FFFFh</li> <li>The base timer is reset by matching the base timer value with the G0PO0 register setting (when bits RST2 and RST1 are 01b)               <ul style="list-style-type: none"> <li>Cycle: <math>\frac{n + 2}{f_{BT0}}</math></li> <li>Low level width: <math>\frac{m}{f_{BT0}}</math></li> <li>High level width: <math>\frac{n + 2 - m}{f_{BT0}}</math></li> </ul> </li> <li><i>m</i>: G0POj register setting value (j = 1 to 7), 0000h to FFFFh</li> <li><i>n</i>: G0PO0 register setting value, 0001h to FFFDh</li> <li>If <math>m \geq n + 2</math>, the output level is fixed to low</li> </ul>
Waveform output start condition (2)	The IFEj bit in the G0FE register is 1 (enable the channel j function) (j = 0 to 7)
Waveform output stop condition	The IFEj bit is 0 (disable the channel j function)
Interrupt request	When the PO0jR bit in the intelligent I/O interrupt request register becomes 1 (interrupt requested) by matching the base timer value with the G0POj register setting (refer to Figure 10.12)
IIO0_j output pin function	Pulse signal output
Other functions	<ul style="list-style-type: none"> <li>Default value setting This function determines the starting waveform output level</li> <li>Output level inversion This function inverts the waveform output level and outputs the inverted signal from the IIO0_j pin</li> </ul>

Notes:

- When the INV bit in the G0POCRj register is 1 (invert the output level), the high and low widths are inverted.
- To use channels shared by time measurement and waveform generation, set the FSCj bit in the G0FS register to 0 (select the waveform generation).

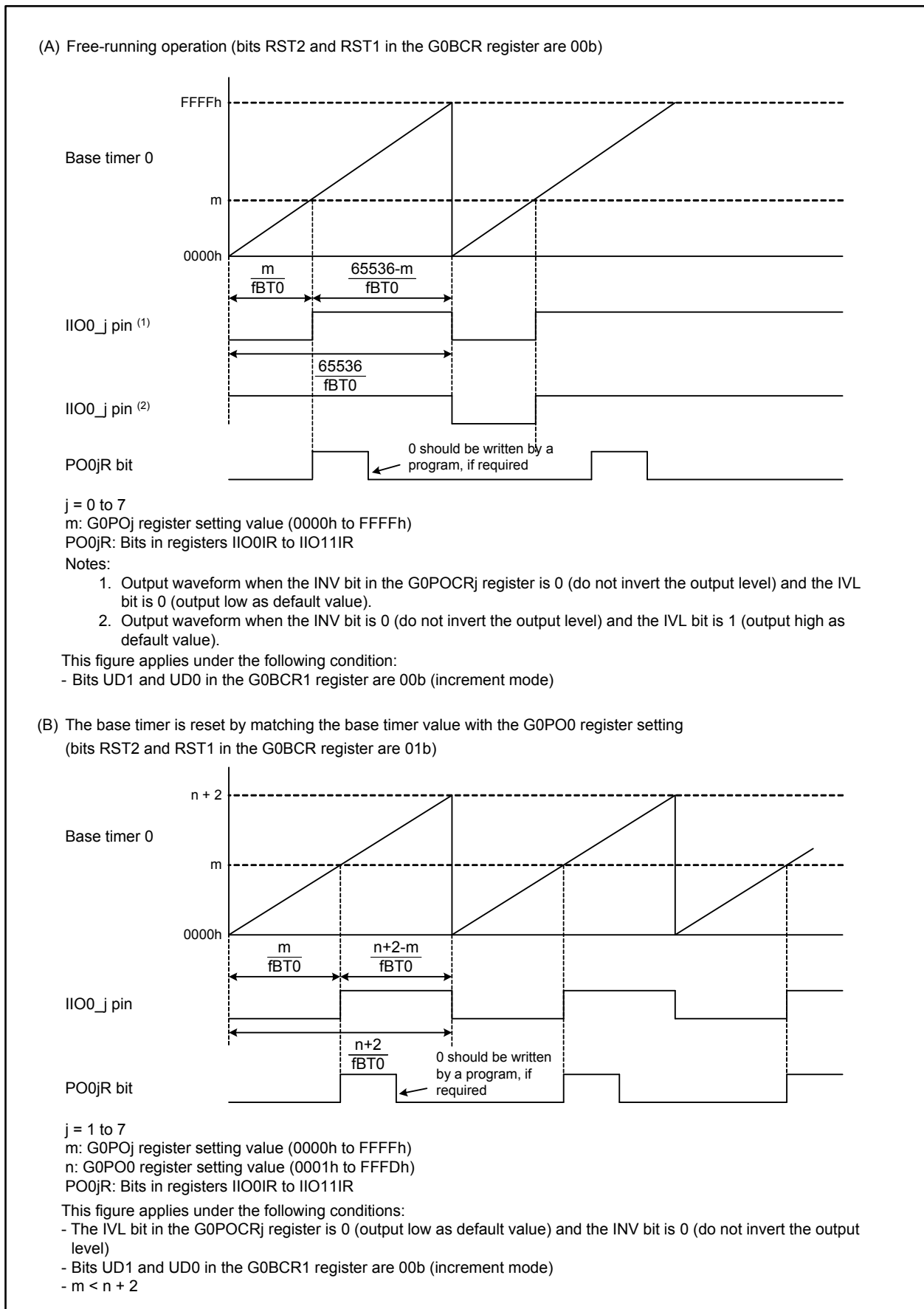


Figure 21.21 Single-phase Waveform Output Mode Operation

### 21.3.2 Inverted Waveform Output Mode

The output level at the IIO0\_j pin is inverted every time the base timer value matches the G0POj register setting (j = 0 to 7).

Table 21.8 lists specifications of the inverted waveform output mode. Figure 21.22 shows an example of the inverted waveform output mode operation.

**Table 21.8 Inverted Waveform Output Mode Specifications**

Item	Specification
Output waveform	<ul style="list-style-type: none"> <li>Free-running operation (when bits RST2 and RST1 in the G0BCR1 register are 00b)           <ul style="list-style-type: none"> <li>Cycle: <math>\frac{65536 \times 2}{f_{BT0}}</math></li> <li>High or low level width: <math>\frac{65536}{f_{BT0}}</math></li> <li>m: G0POj register setting value (j = 0 to 7), 0000h to FFFFh</li> </ul> </li> <li>The base timer is reset by matching the base timer value with the G0PO0 register setting (when bits RST2 and RST1 are 01b)           <ul style="list-style-type: none"> <li>Cycle: <math>\frac{2(n+2)}{f_{BT0}}</math></li> <li>High or low level width: <math>\frac{n+2}{f_{BT0}}</math></li> <li>n: G0PO0 register setting value, 0001h to FFFDh</li> <li>G0POj register setting value (j = 1 to 7), 0000h to FFFFh</li> <li>If the G0POj register setting <math>\geq n+2</math>, the output level is not inverted</li> </ul> </li> </ul>
Waveform output start condition <sup>(1)</sup>	The IFEj bit in the G0FE register is 1 (enable the channel j function) (j = 0 to 7)
Waveform output stop condition	The IFEj bit is 0 (disable the channel j function)
Interrupt request	When the PO0jR bit in the intelligent I/O interrupt request register becomes 1 (interrupt requested) by matching the base timer value with the G0POj register setting (refer to Figure 10.12)
IIO0_j output pin function	Pulse signal output
Other functions	<ul style="list-style-type: none"> <li>Default value setting This function determines the starting waveform output level</li> <li>Output level inversion This function inverts the waveform output level and outputs the inverted signal from the IIO0_j pin</li> </ul>

Note:

- To use channels shared by time measurement and waveform generation, set the FSCj bit in the G0FS register to 0 (select the waveform generation).

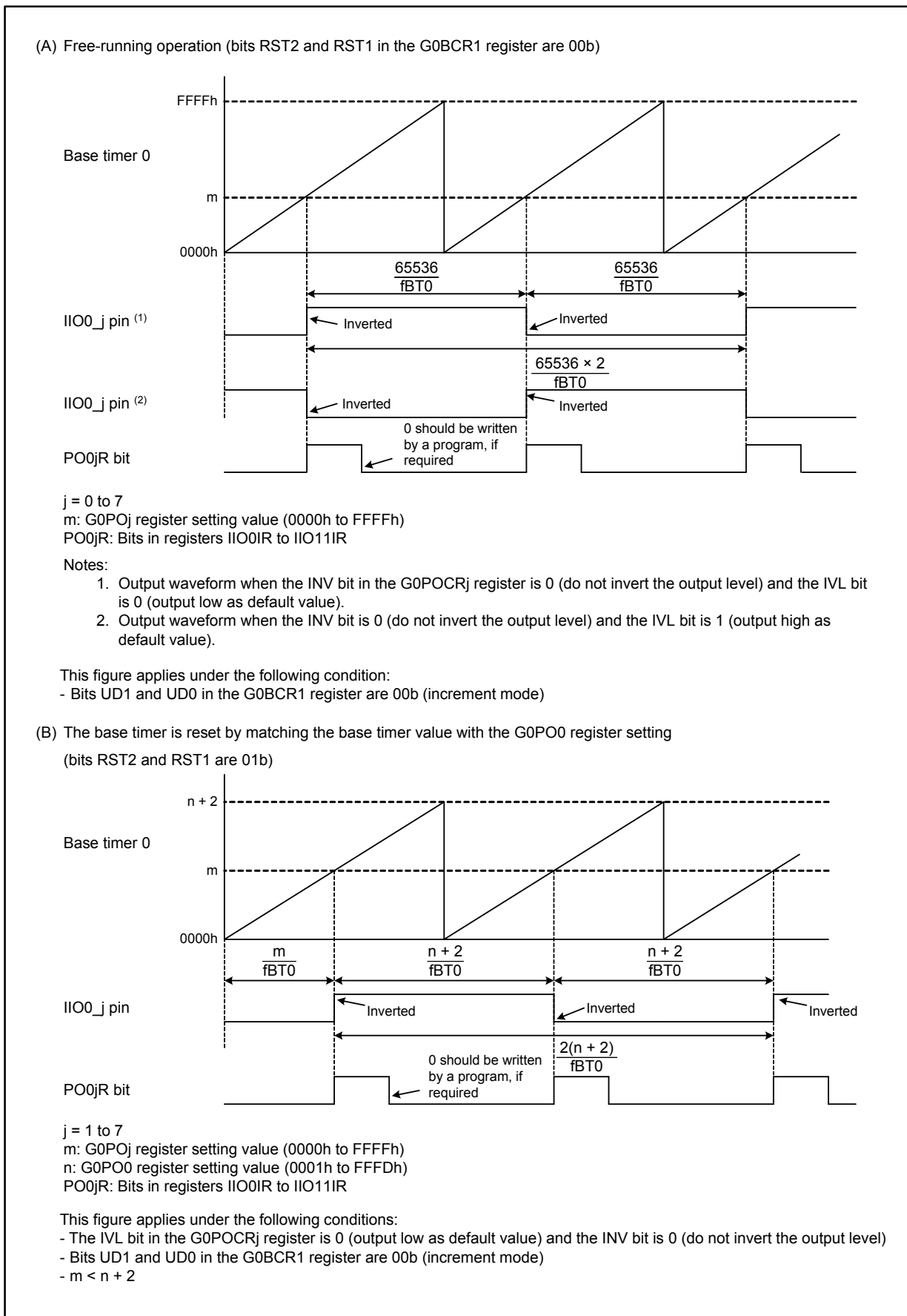


Figure 21.22 Inverted Waveform Output Mode Operation



### 21.3.3 Set/Reset Waveform Output Mode (SR Waveform Output Mode)

The output level at the IIO0\_j pin becomes high when the base timer value matches the G0POj register setting ( $j = 0, 2, 4, 6$ ). It becomes low when the base timer value matches the G0POk register setting or the base timer reaches 0000h ( $k = j + 1$ ). When the IVL bit in the G0POCRj register is set to 1 (output high as default value), a high output level is provided when a waveform output starts ( $j = 0$  to 7). When the INV bit is set to 1 (invert the output level), a waveform with inverted level is output. Refer to Figure 21.23 for details on SR waveform mode operation. Tables 21.9 and 21.10 list specifications of SR waveform output mode.

**Table 21.9 SR Waveform Output Mode Specifications (1/2)**

Item	Specification
Output waveform <sup>(1)</sup>	<ul style="list-style-type: none"> <li>Free-running operation (when bits RST2 and RST1 in the G0BCR1 register are 00b)               <ul style="list-style-type: none"> <li>(A) <math>m &lt; n</math> <ul style="list-style-type: none"> <li>High level width: <math>\frac{n - m}{f_{BT0}}</math></li> <li>Low level width: <math>\frac{m}{f_{BT0}}</math> (See Note 2) + <math>\frac{65536 - n}{f_{BT0}}</math> (See Note 3)</li> </ul> </li> <li>(B) <math>m \geq n</math> <ul style="list-style-type: none"> <li>High level width: <math>\frac{65536 - m}{f_{BT0}}</math></li> <li>Low level width: <math>\frac{m}{f_{BT0}}</math></li> </ul> </li> </ul> </li> <li><math>m</math>: G0POj register setting value (<math>j = 0, 2, 4, 6</math>), 0000h to FFFFh  <math>n</math>: G0POk register setting value (<math>k = j + 1</math>), 0000h to FFFFh</li> <li>The base timer is reset by matching with the G0PO0 register (when bits RST2 and RST1 are 01b) <sup>(4)</sup> <ul style="list-style-type: none"> <li>(A) <math>m &lt; n &lt; p + 2</math> <ul style="list-style-type: none"> <li>High level width: <math>\frac{n + m}{f_{BT0}}</math></li> <li>Low width: <math>\frac{m}{f_{BT0}}</math> (See Note 2) + <math>\frac{p + 2 - n}{f_{BT0}}</math> (See Note 3)</li> </ul> </li> <li>(B) <math>m &lt; p + 2 \leq n</math> <ul style="list-style-type: none"> <li>High level width: <math>\frac{p + 2 - m}{f_{BT0}}</math></li> <li>Low level width: <math>\frac{m}{f_{BT0}}</math></li> </ul> </li> <li>(C) <math>m \geq p + 2</math>, output level is fixed to low  <math>p</math>: G0PO0 register setting value, 0001h to FFFDh  <math>m</math>: G0POj register setting value (<math>j = 2, 4, 6</math>), 0000h to FFFFh  <math>n</math>: G0POk register setting value (<math>k = j + 1</math>), 0000h to FFFFh</li> </ul> </li></ul>

Notes:

- When the INV bit in the G0POCRj register is 1 (invert the output level), the high and low widths are inverted.
- Output period from a base timer reset until when the output level becomes high.
- Output period from when the output level becomes low until the next base timer reset.
- When the G0PO0 register resets the base timer, channel 0 and channel 1 SR waveform generation functions are not available.

**Table 21.10 SR Waveform Output Mode Specifications (2/2)**

Item	Specification
Waveform output start condition (1)	The IFEq bit in the G0FE register is 1 (enable the channel q function) (q = 0 to 7)
Waveform output stop condition	The IFEq bit is 0 (disable the channel q function)
Interrupt request	When the PO0jR bit in the intelligent I/O interrupt request register becomes 1 (interrupt requested) by matching the base timer value with the G0POj register setting. When the PO0kR bit becomes 1 (interrupt requested) by matching the base timer value with the G0POk register setting (refer to Figure 10.12)
IIO0_j output pin function	Pulse signal output
Other functions	<ul style="list-style-type: none"> <li>• Default value setting This function determines the starting waveform output level</li> <li>• Output level inversion This function inverts the waveform output level and outputs the inverted signal from the IIO0_j pin</li> </ul>

Note:

1. To use channels shared by time measurement and waveform generation, set the FSCj bit in the G0FS register to 0 (select the waveform generation).

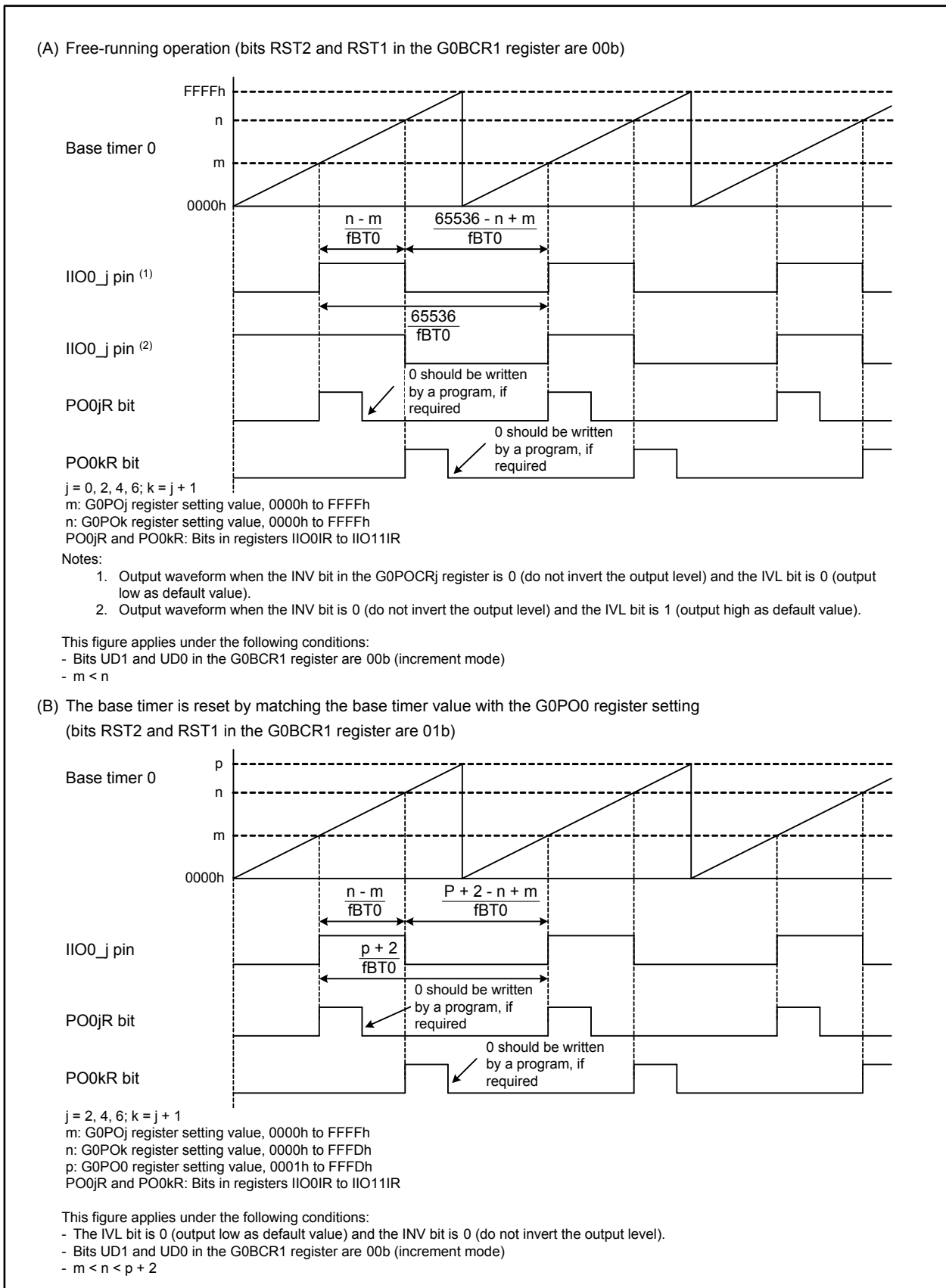


Figure 21.23 SR Waveform Output Mode Operation

### 21.3.4 Phase Shift Waveform Output Mode

PWM output is phase-shifted at every channel in this mode. This mode enables functions that help to reduce switching noise and instantaneous power consumption.

The following bit settings are necessary to enable this mode; set bits MOD2 to MOD0 in the G0POCRj register to 000b (single-phase waveform output mode), the IVL bit to 0 (output low as default value), the BTRE bit to 1 (reset the base timer when bit 9 overflows), the INV bit to 1 (invert the output level), and the IT bit in the G0BCR0 register to 0 (interrupt occurs when bit 15 or bit 9 overflows) ( $j = 0$  to 7).

When the phase shift waveform output mode is disabled, the output level at the IIO0\_j pin becomes low if the base timer matches the G0POj register, and becomes high when the base timer reaches 0000h (refer to (A) in Figure 21.24).

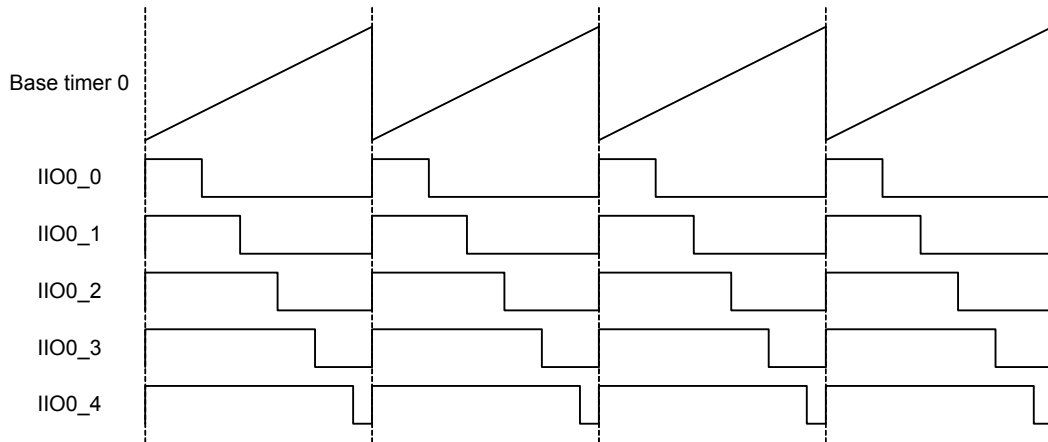
When this mode is enabled, the phase shifter controls the timing of the output level to be high. The phase shift clock to be input to the phase shifter is the count source for the base timer fBT0 divided by  $m + 1$ . Every time the phase shift clock is input, each output level of the IIO0\_j pin starting from the IIO0\_0 sequentially becomes high (refer to (B) in Figure 21.24). The base timer overflow interrupt signal BTOR controls reloading of the divided-by- $m + 1$  divider and the reset of the phase shifter.

Figure 21.25 shows some more examples of output waveforms using different settings.

Note that if all channels are set to phase shift waveform output mode and the phase shift clock frequency is set to the lowest, the phase shift clock for channel 7 is not generated. Consequently, the output level at the IIO0\_7 pin is held low.

When phase shift output mode is enabled, the output level of a pin cannot be fixed to low. To hold the level low, the corresponding pin should be switched to the port by the output function select register.

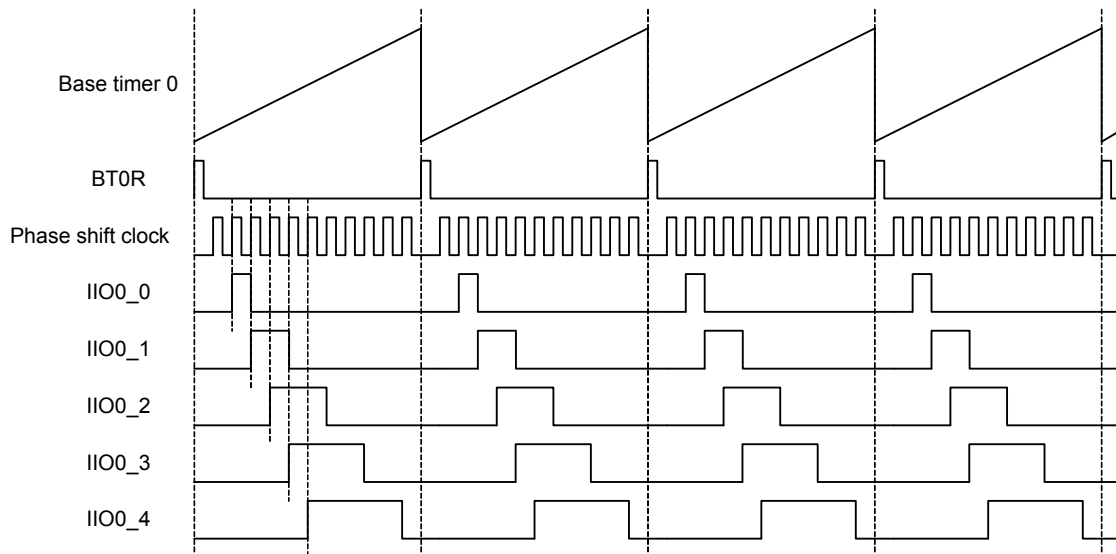
## (A) The phase shift waveform output mode is disabled



This figure applies under the following conditions:

- The PSME bit in the G0PSCR register is 0 (phase shift waveform output mode is disabled)
- Bits MOD2 to MOD0 in the G0POCRj register are 000b (single-phase waveform output mode), the IVL bit is 0 (output low as default value), and the INV bit is 1 (invert the output level) (j = 0 to 4)
- Bits RST2 and RST1 in the G0BCR1 register are 00b (base timer is not reset) and bits UD1 and UD0 are 00b (increment mode)

## (B) The phase shift waveform output mode is enabled



This figure applies under the following conditions:

- The PSME bit in the G0PSCR register is 1 (phase shift waveform output mode is enabled) and bits PSS1 and PSS0 are 00b (pins IIO0\_0 to IIO0\_4 are available)
- Bits MOD2 to MOD0 in the G0POCRj register are 000b (single-phase waveform output mode), the IVL bit is 0 (output low as default value), the BTRE bit is 1 (reset the base timer when bit 9 overflows), and the INV bit is 1 (invert the output level) (j = 0 to 4)
- The IT bit in the G0BCR0 register is 0 (interrupt occurs when bit 15 or bit 9 overflows)
- Bits RST2 and RST1 in the G0BCR1 register are 00b (base timer is not reset) and bits UD1 and UD0 are 00b (increment mode)

Figure 21.24 Phase Shift Waveform Output Mode (1/2)

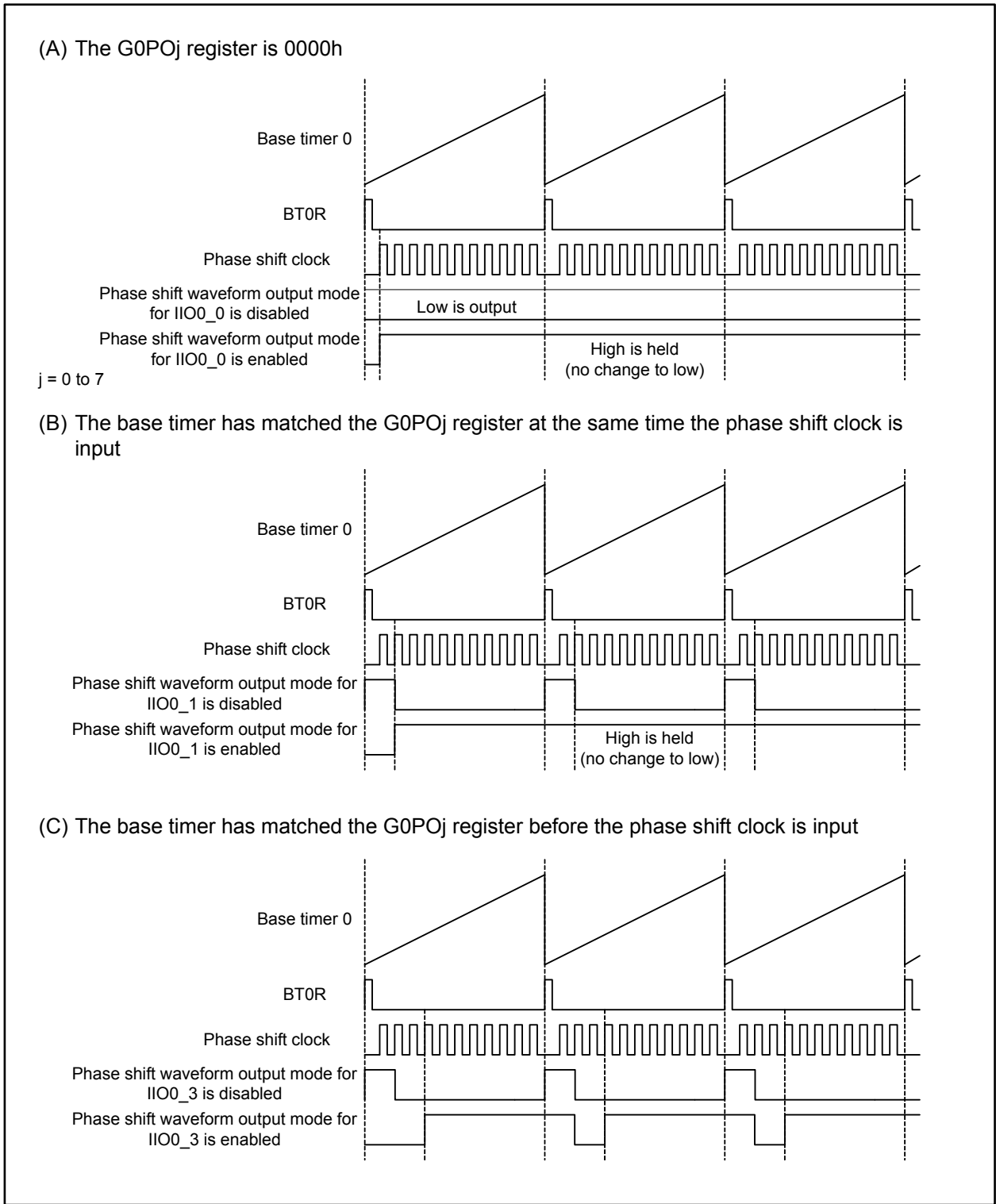


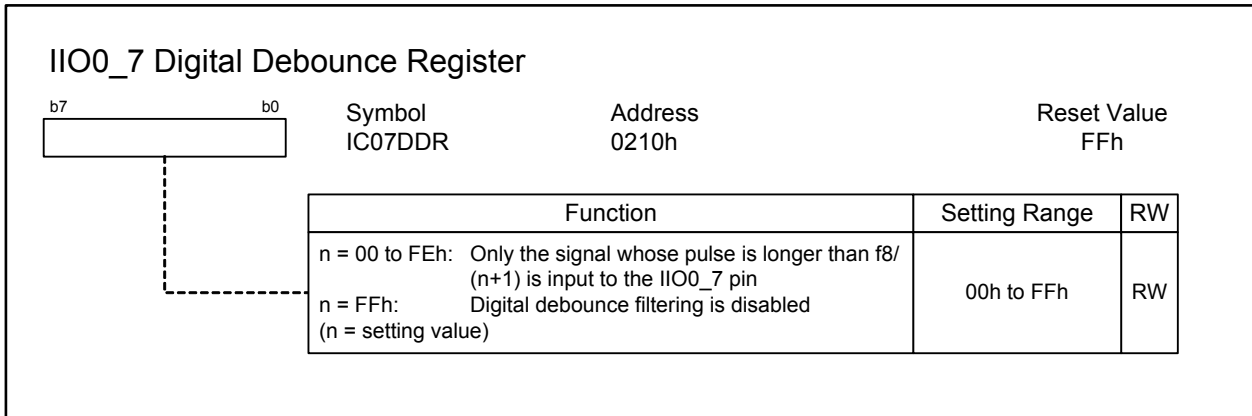
Figure 21.25 Phase Shift Waveform Output Mode (2/2)

### 21.3.5 Digital Debounce Circuit

The IIO0\_7 pin has a digital debounce function for noise reduction. This function helps to determine the signal level when the pulse becomes longer than the filter width set by a program after the signal is input on a rising or falling edge.

This function does not affect any signals that share the IIO0\_7 pin.

Figure 21.26 shows the registers IC07DDR.



**Figure 21.26 Registers IC07DDR**

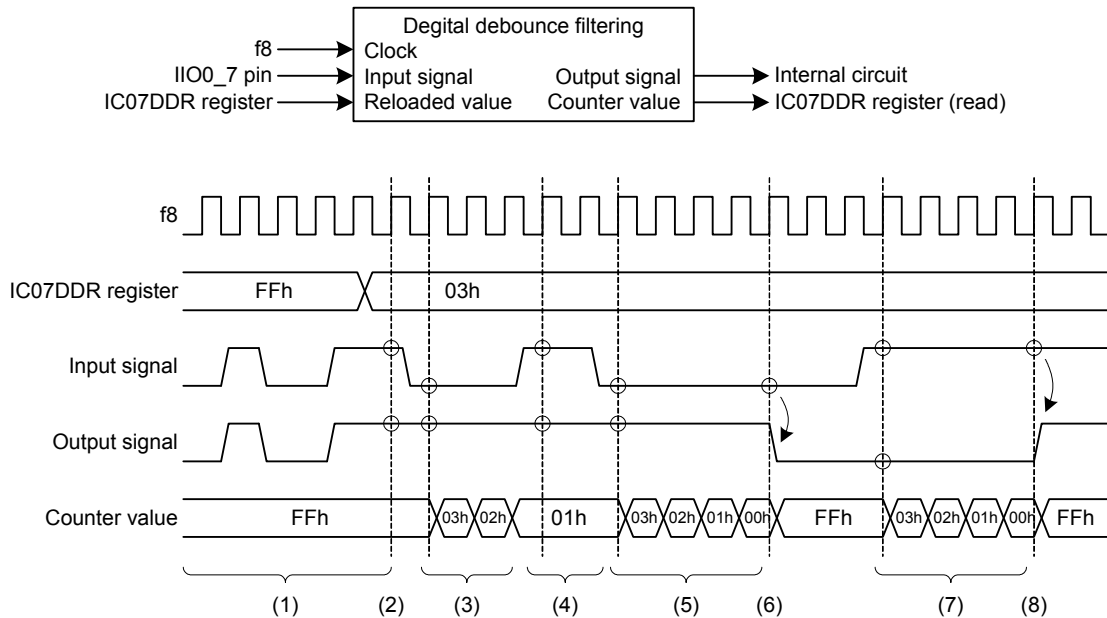
The counter for the digital debounce at the IIO0\_7 pin decrements with a count source of f8. Every time the signal level at the pin changes, the setting value of the IC07DDR register is reloaded to resume counting. The counted value can be read from the IC07DDR register.

The input signal to the IIO0\_7 pin is output from the digital debounce circuit on the first rising edge of f8 after the counter value becomes 00h.

A value from 00h to FEh can be set to the IC07DDR register when using the digital debounce function. When FFh is set to this register, this function is disabled and the input signal is transmitted without being filtered.

Figure 21.27 shows an example of digital debounce filtering.

An example of digital debounce function of IIO0\_7 signal (IC07DDR = 03h)



- (1) The IC07DDR register is FFh after a reset. That is, the filtering is disabled, and the input signal is output without being filtered.
- (2) The counter is stopped if the input and output levels are the same when setting 03h to the IC07DDR register.
- (3) When detecting the input and output levels being different, the counter reloads the setting value of the IC07DDR register and starts counting.
- (4) When the input and output levels become the same during counting, the counter stops. The output level does not change when the pulse width of the input signal is too short for the counter to become 00h.
- (5) When detecting the input and output levels being different again, the counter reloads the setting value of the IC07DDR register and starts counting.
- (6) The low signal is applied on the first rising edge of f8 after the counter reaches 00h. The output signal becomes low and the counter stops.
- (7) When detecting the input and output levels being different again, the counter reloads the setting value of the IC07DDR register and starts counting.
- (8) The high signal is applied on the first rising edge of f8 after the counter reaches 00h. The output signal becomes high and the counter stops.

When the IC07DDR register is set to FFh, the counter value becomes FFh after 1.5 to 2.5 cycles of f1, and the counter stops. Simultaneously, the filtering is disabled, and the input signal is output without being filtered.

**Figure 21.27 Digital Debounce Filtering**



## 22. Serial Bus Interface

The serial bus interface has one independent channel: SBI0.

SBI0 has its own transmit/receive clock generator.

SBI0 has the following modes:

- Synchronous serial communication mode
- 4-wire serial bus mode

To select a mode, set the SSUMS bit in the SS0MR2 register.

Note that each mode has its own bit names, bit symbols, and bit functions in some registers.

## 22.1 Synchronous Serial Communication Mode and 4-wire Serial Bus Mode

To switch between synchronous serial communication mode and 4-wire serial bus mode, set the SSUMS bit in the SS0MR2 register.

Table 22.1 and Figure 22.1 show the specifications of synchronous serial communication mode and its block diagram, respectively. Table 22.2 and Figure 22.2 show the specifications and block diagram of 4-wire serial bus mode, respectively.

Figures 22.3 to 22.11 show the associated registers.

**Table 22.1 Synchronous Serial Communication Mode Specifications**

Item	Specification
Data format	Character length: 8 bits (fixed length)
Master/slave mode	Selectable
I/O pins	SSCK0 (I/O): Clock I/O pin SSIO (input): Data input pin SSO0 (output): Data output pin
Transmit/receive clocks	<ul style="list-style-type: none"> <li>When the MSS bit in the SS0CRH register is 0 (slave mode): External clock (input at the SSCK0 pin)</li> <li>When the MSS bit is 1 (master mode): Internal clock (output from the SSCK0 pin)</li> </ul> $\frac{f_{(BCLK)}}{n}$ $f_{(BCLK)}: \text{peripheral bus clock frequency}$ $n: 4, 8, 16, 32, 64, 128, \text{ or } 256$
Transmit start conditions	<ul style="list-style-type: none"> <li>The TE bit in the SS0ER register is 1 (transmission enabled)</li> <li>The TDRE bit in the SS0SR register is 0 (unsent data left in the SS0TDR register)</li> <li>The ORER bit in the SS0SR register is 0 (no overrun error)</li> <li>Clock input at the SSCK0 pin (applicable only when the MSS bit in the SS0CRH register is 0 (slave mode))</li> </ul>
Receive start conditions	<ul style="list-style-type: none"> <li>The RE bit in the SS0ER register is 1 (reception enabled)</li> <li>The ORER bit in the SS0SR register is 0 (no overrun error)</li> <li>Read operation to the SS0RDR register (applicable only when the MSS bit in the SS0CRH register is 1 (master mode))</li> <li>Clock input at the SSCK0 pin (applicable only when the MSS bit in the SS0CRH register is 0 (slave mode))</li> </ul>
Error detection	<ul style="list-style-type: none"> <li>Overrun error</li> </ul> <p>An overrun error occurs when the next serial data reception is completed while the RDRF bit in the SS0SR register is 1 (received data left in the SS0RDR register)</p>
Interrupt request sources	Four interrupt request sources: transmit end, transmit data register empty, receive data register full, and overrun error <sup>(1)</sup>
Other functions	<ul style="list-style-type: none"> <li>Bit order select Selectable either LSB first or MSB first</li> <li>SSCK0 clock polarity Selectable either low or high for the SSCK0 pin when clock is stopped</li> <li>SSCK0 clock phase Combination of odd and even edges for changing and loading data</li> </ul>

Note:

- Each channel has its own interrupt vector.

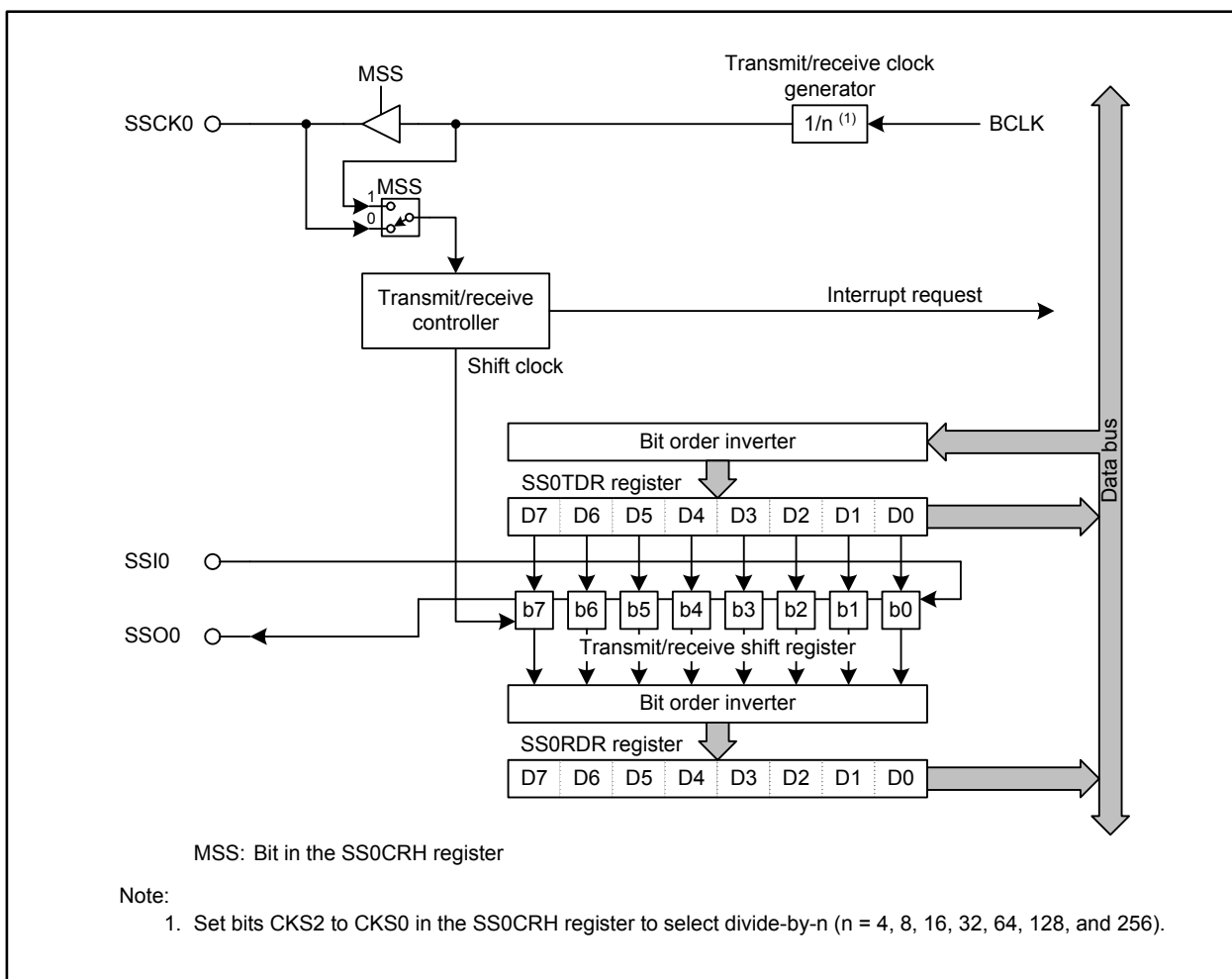


Figure 22.1 Block Diagram of Synchronous Serial Communication Mode

**Table 22.2 4-wire Serial Bus Mode Specifications**

Item	Specification
Data format	Character length: 8 to 16 bits (variable length)
Master/slave mode	Selectable
I/O pins	SSCK0 (I/O): Clock I/O pin SSIO (I/O): Data I/O pin SSO0 (I/O): Data I/O pin SCS0 (I/O): Chip select I/O pin
Transmit/receive clocks	<ul style="list-style-type: none"> <li>When the MSS bit in the SS0CRH register is 0 (slave mode): External clock (input at the SSCK0 pin)</li> <li>When the MSS bit 1 (master mode): Internal clock (output from the SSCK0 pin)</li> </ul> $\frac{f_{(BCLK)}}{n}$ $f_{(BCLK)}$ peripheral bus clock frequency $n$ : 4, 8, 16, 32, 64, 128, or 256
Transmit start conditions	<ul style="list-style-type: none"> <li>The TE bit in the SS0ER register is 1 (transmission enabled)</li> <li>The TDRE bit in the SS0SR register is 0 (unsent data left in the SS0TDR register)</li> <li>The ORER bit in the SS0SR register is 0 (no overrun error)</li> <li>Clock input at the SSCK0 pin while the SCS0 pin is low (applicable only when the MSS bit in the SS0CRH register is 0 (slave mode))</li> </ul>
Receive start conditions	<ul style="list-style-type: none"> <li>The RE bit in the SS0ER register is 1 (reception enabled)</li> <li>The ORER bit in the SS0SR register is 0 (no overrun error)</li> <li>Read operation to the SS0RDR register while the TE bit in the SS0ER register is 0 (transmission disabled) (applicable only when the MSS bit in the SS0CRH register is 1 (master mode))</li> <li>Clock input at the SSCK0 pin (applicable only when the MSS bit in the SS0CRH register is 0 (slave mode))</li> </ul>
Error detection	<ul style="list-style-type: none"> <li>Overrun error An overrun error occurs when the next serial data is received while the RDRF bit in the SS0SR register is 1 (received data left in the SS0RDR register)</li> <li>Conflict error A conflict error occurs in either of the following cases:               <ol style="list-style-type: none"> <li>An attempt to start serial communication while the <math>\overline{SCS0}</math> pin is low (applicable when the MSS bit in the SS0CRH register is 1 (master mode))</li> <li>The <math>\overline{SCS0}</math> pin becomes high during data transfer (applicable when the MSS bit in the SS0CRH register 0 (slave mode))</li> </ol> </li> </ul>
Interrupt request sources	Five interrupt request sources: transmit end, transmit data register empty, receive data register full, overrun error, and conflict error (1)
Other functions	<ul style="list-style-type: none"> <li>Bit order select Selectable either LSB first or MSB first</li> <li>SSCK0 clock polarity Selectable either low or high for the SSCK0 pin when clock is stopped</li> <li>SSCK0 clock phase Combination of odd and even edges for changing and loading data</li> </ul>

Note:

- Each channel has its own interrupt vector.

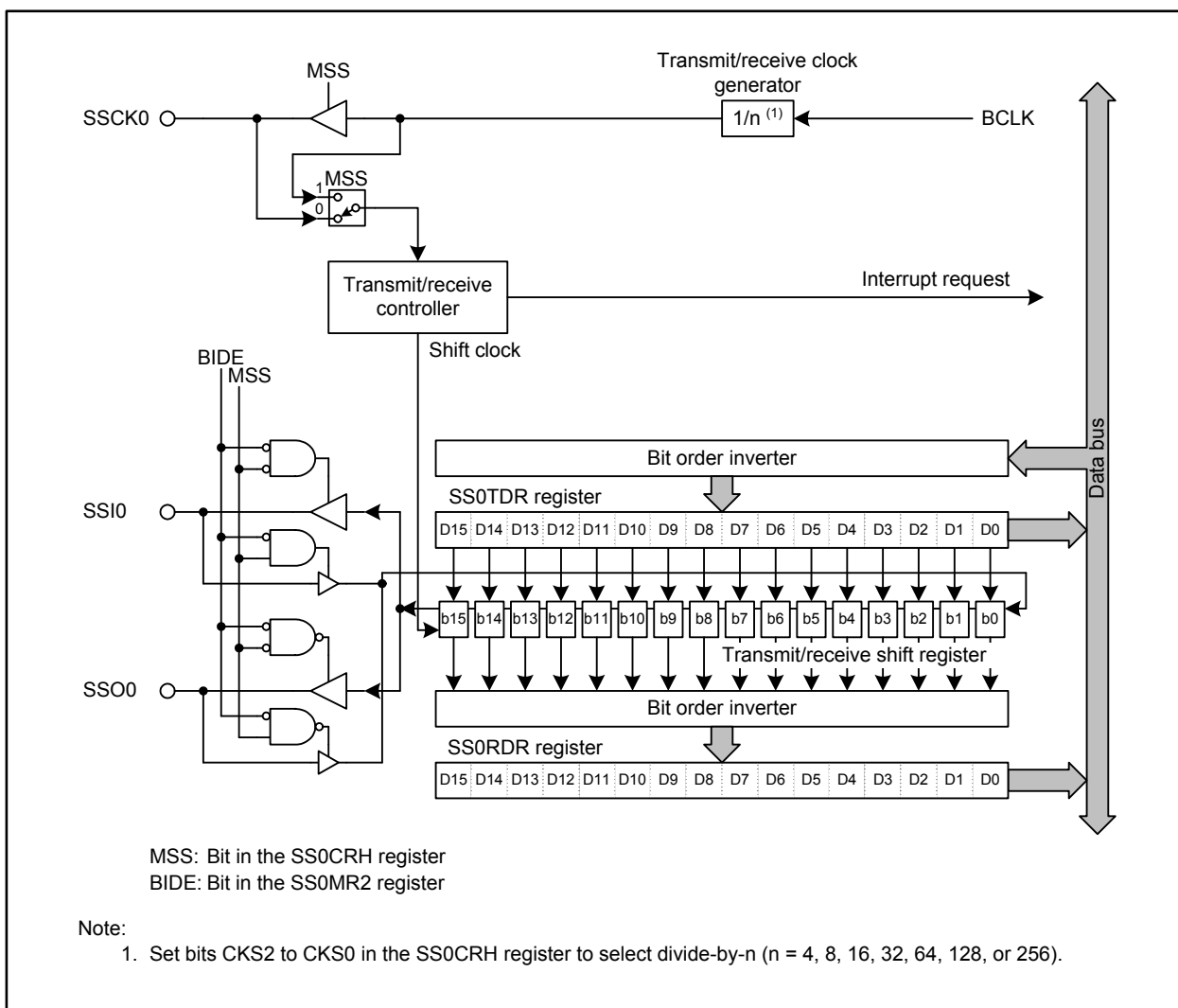


Figure 22.2 Block Diagram of 4-wire Serial Bus Mode

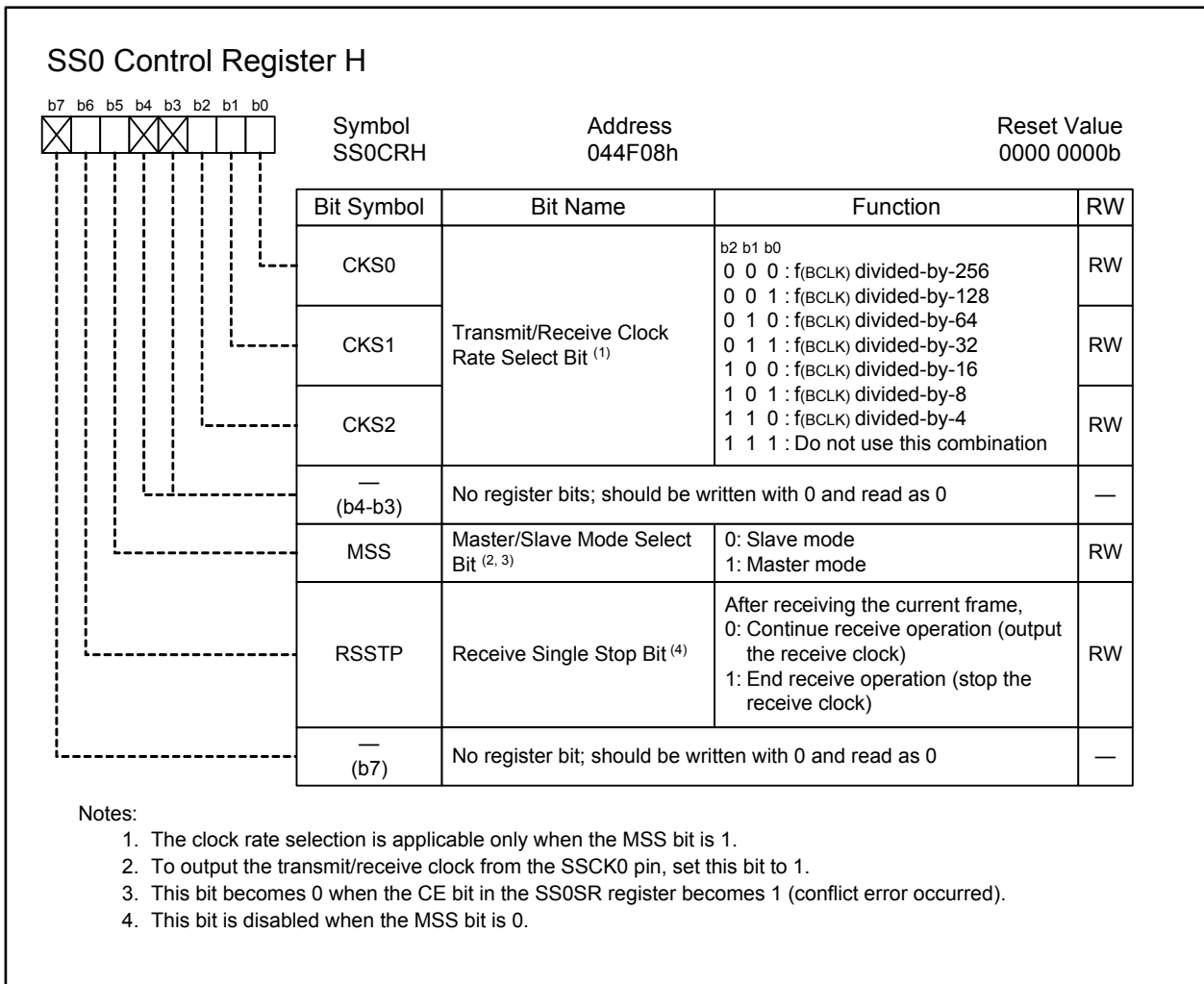


Figure 22.3 SS0CRH Register

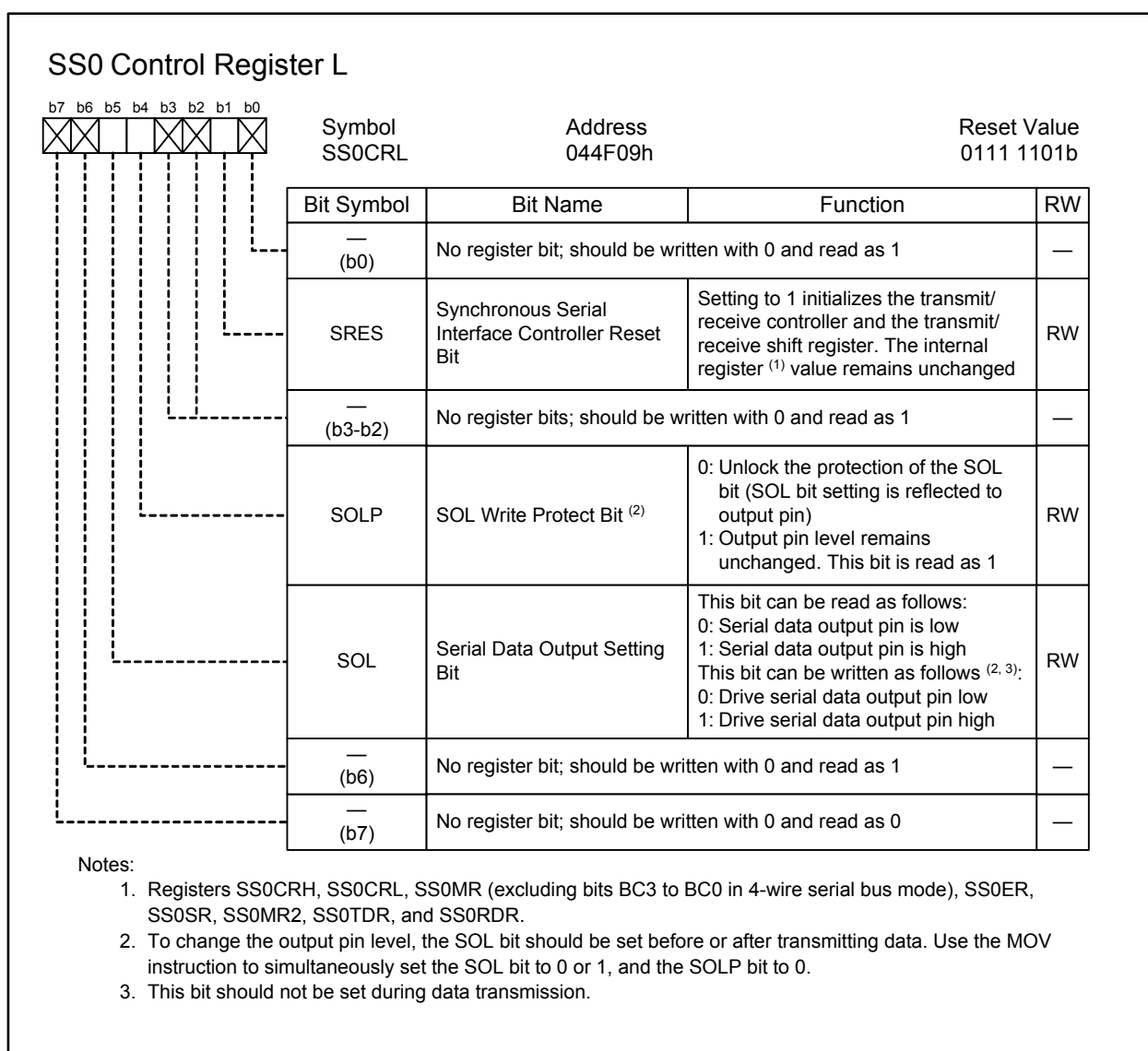
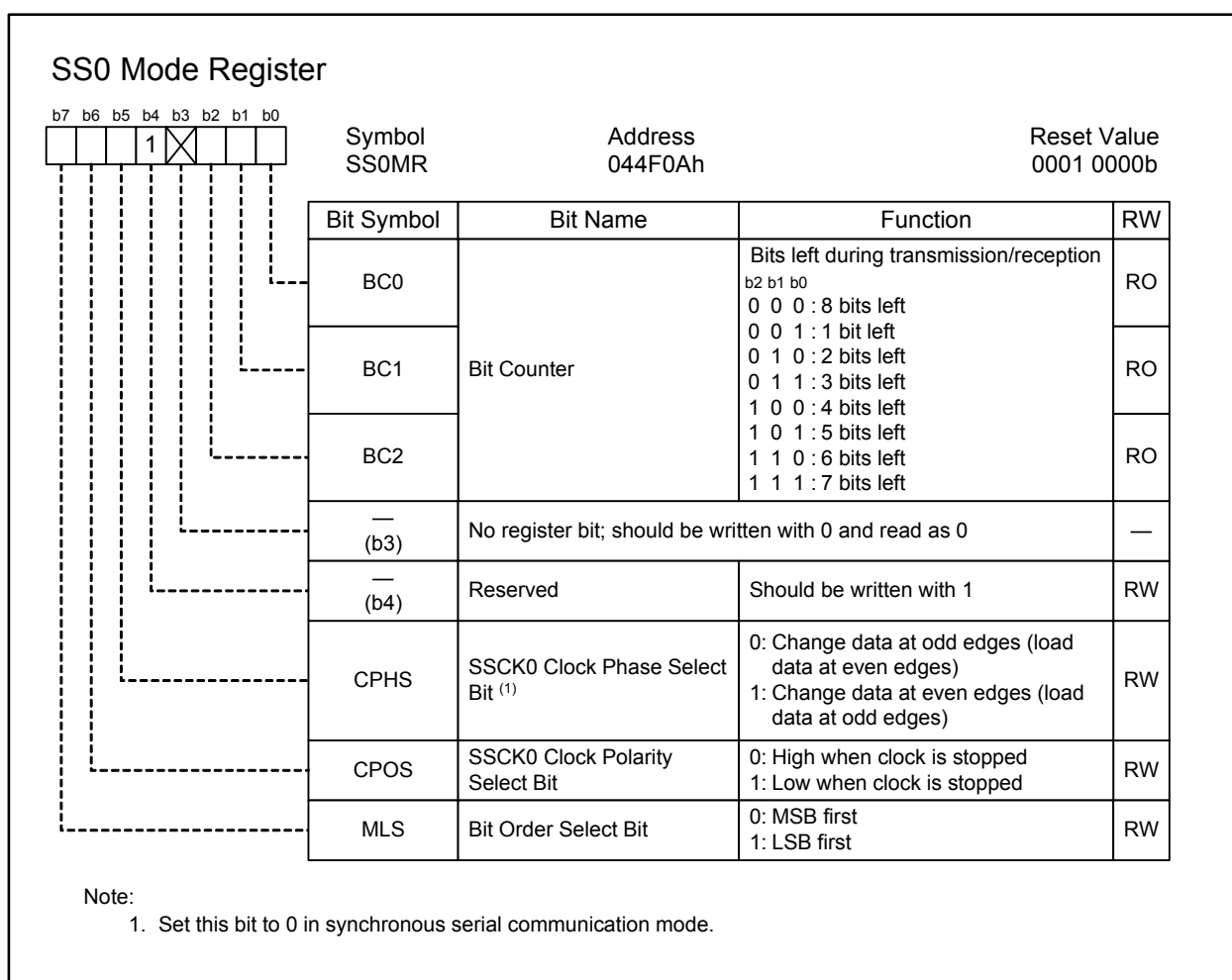


Figure 22.4 SS0CRL Register



**Figure 22.5 SS0MR Register in Synchronous Serial Communication Mode**



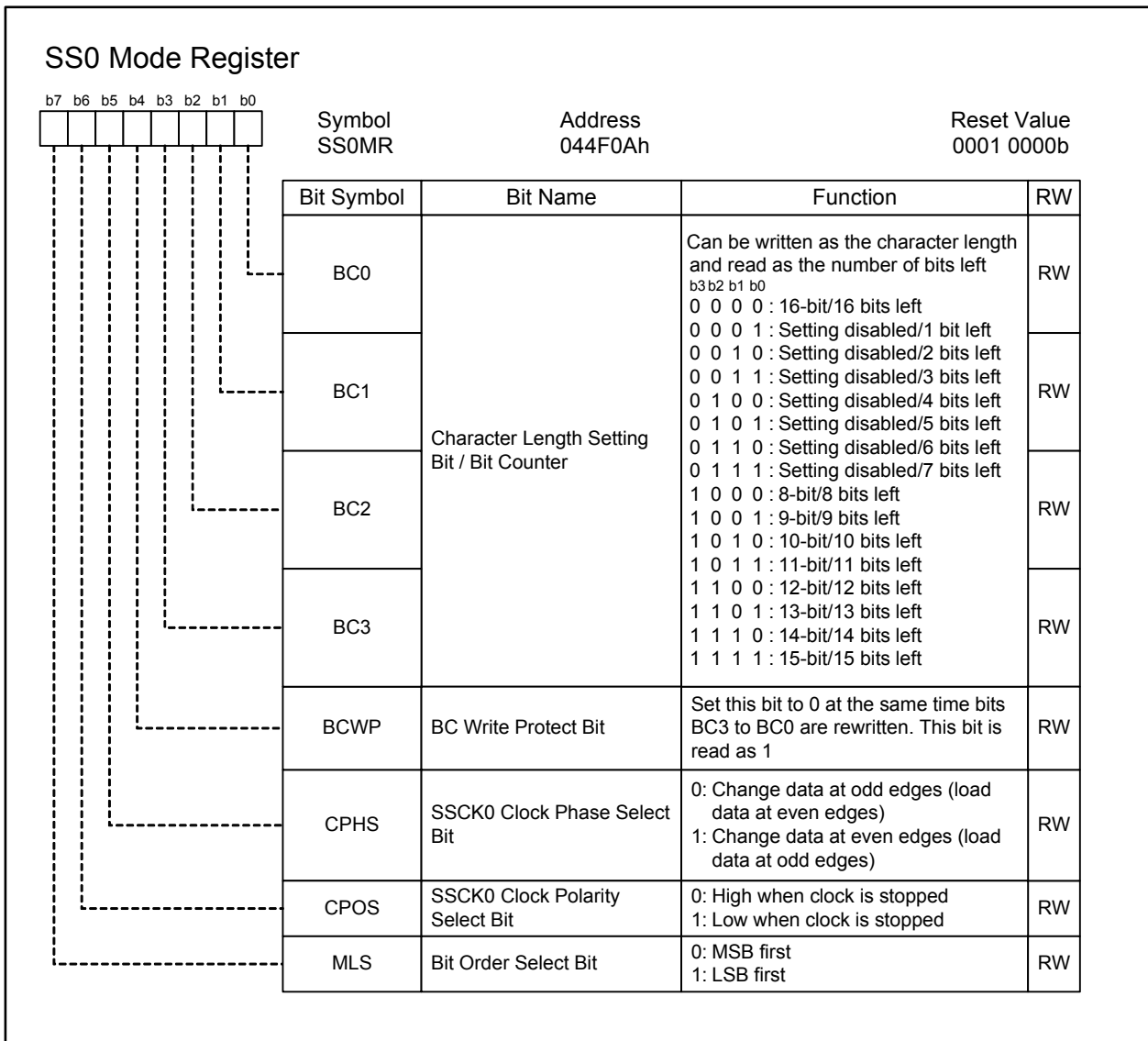


Figure 22.6 SS0MR Register in 4-wire Serial Bus Mode

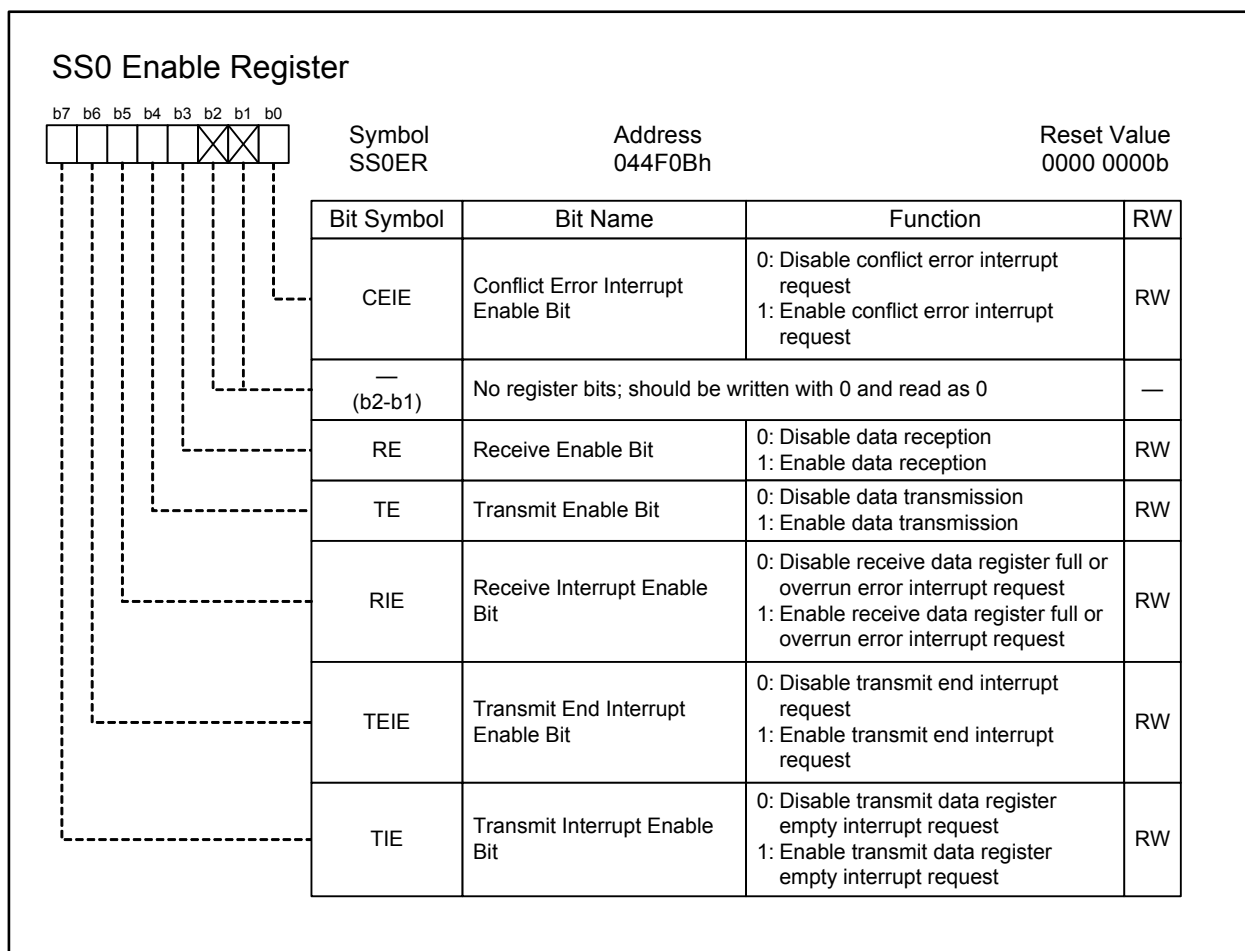


Figure 22.7 SS0ER Register

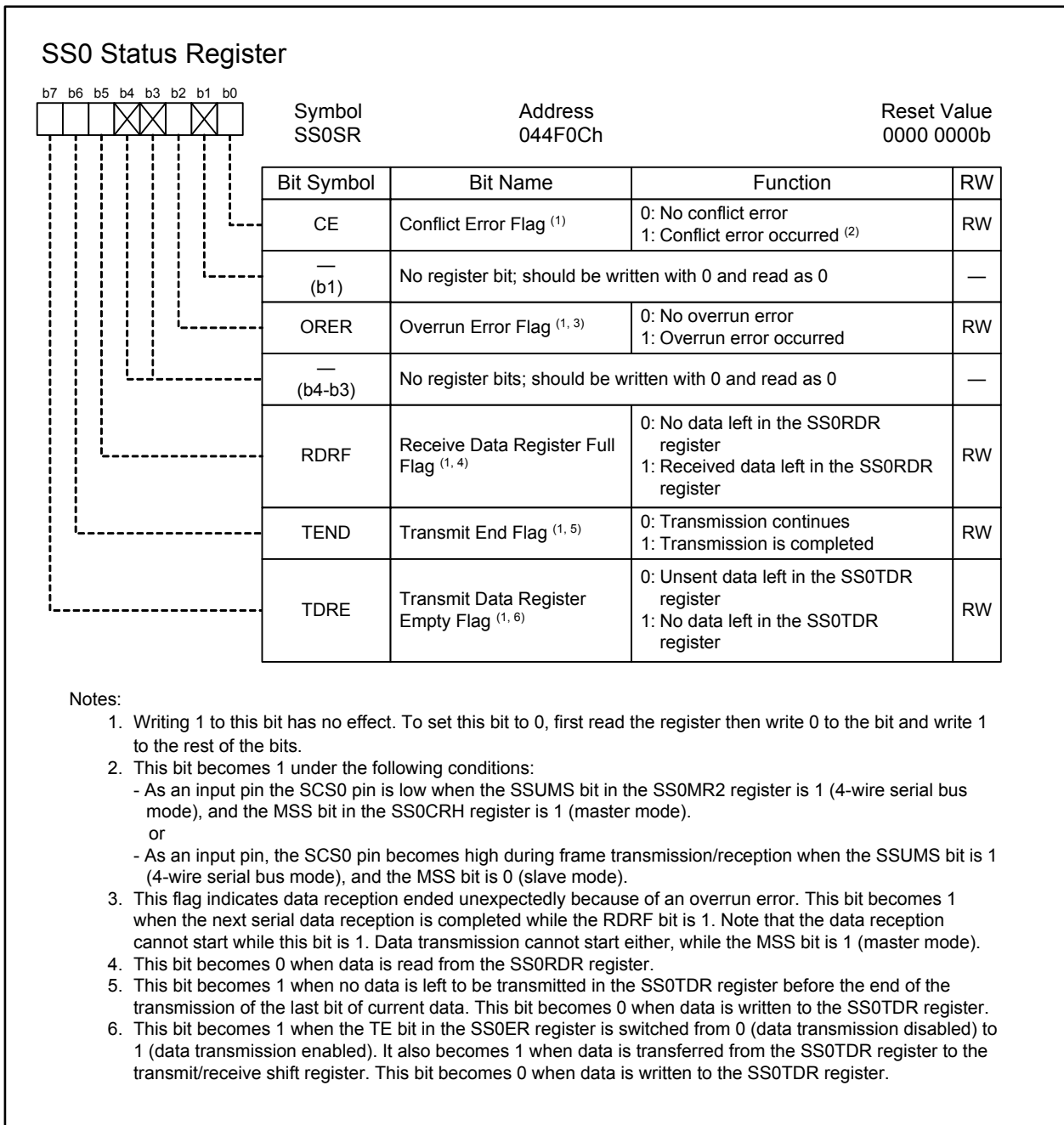


Figure 22.8 SS0SR Register

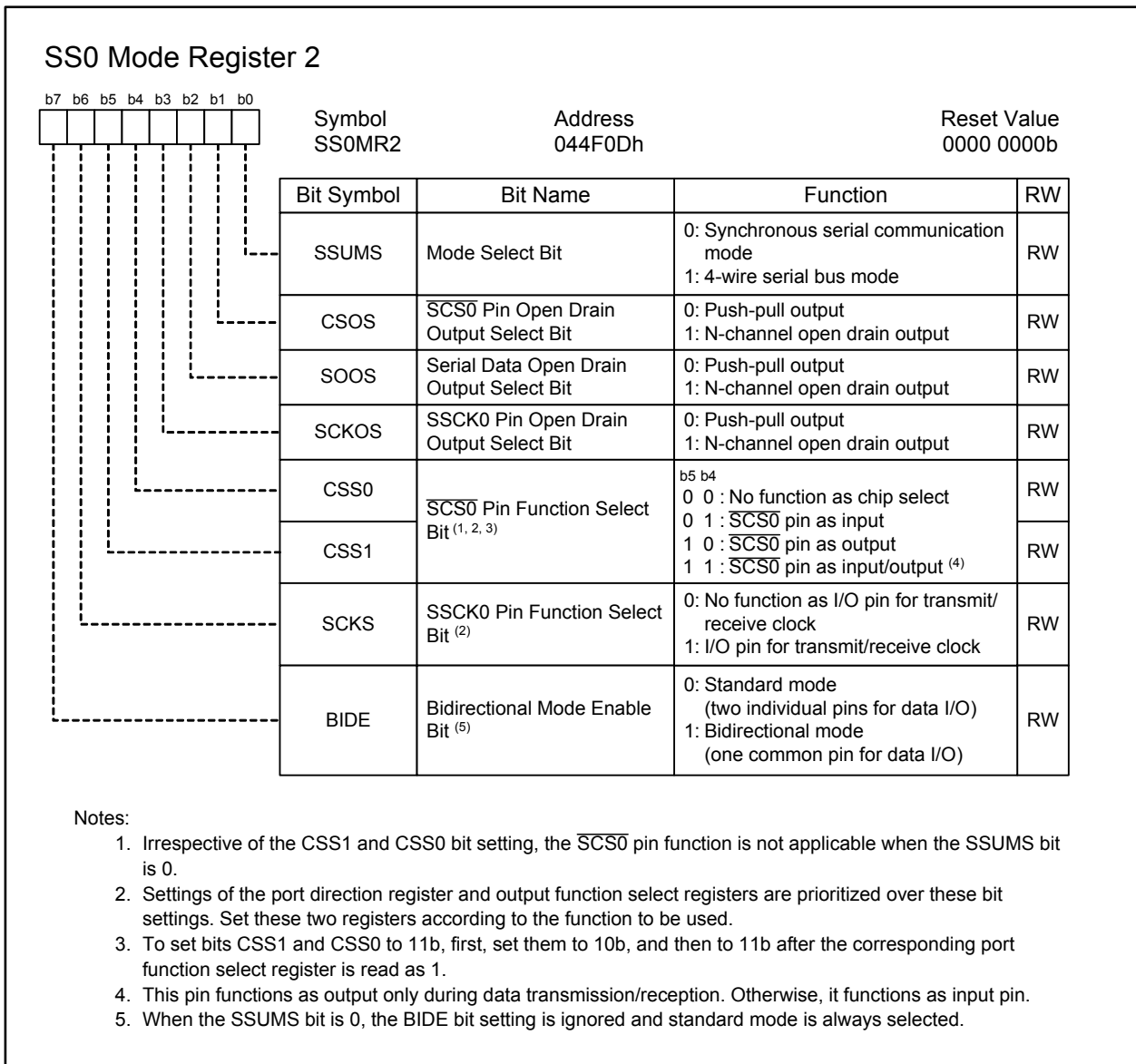


Figure 22.9 SS0MR2 Register

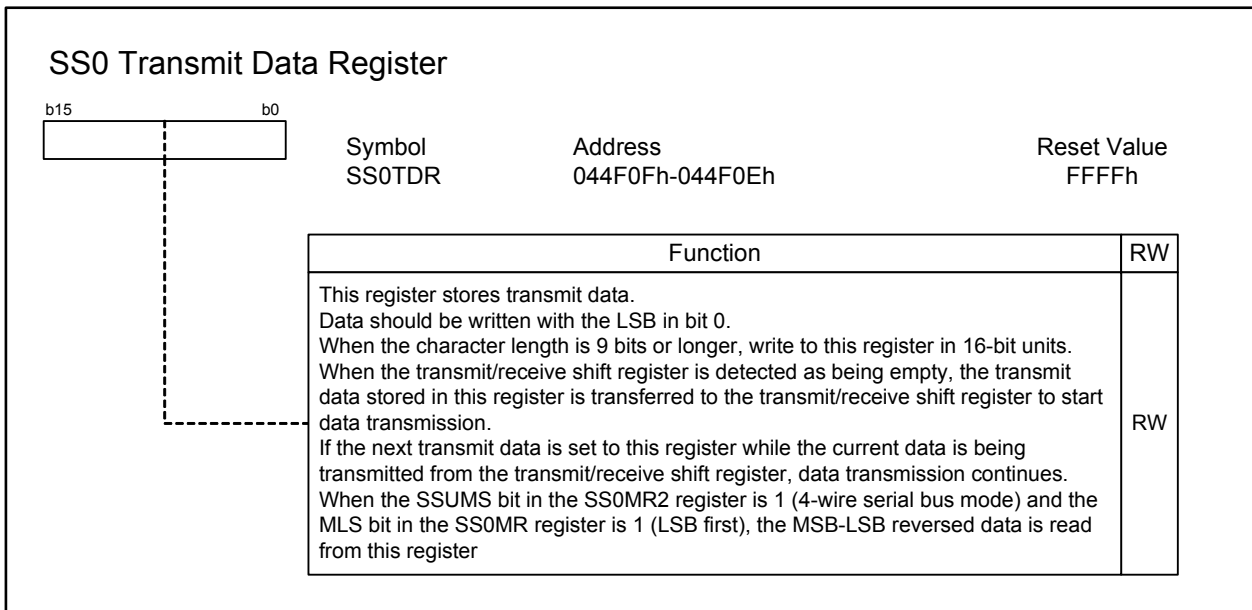


Figure 22.10 SS0TDR Register

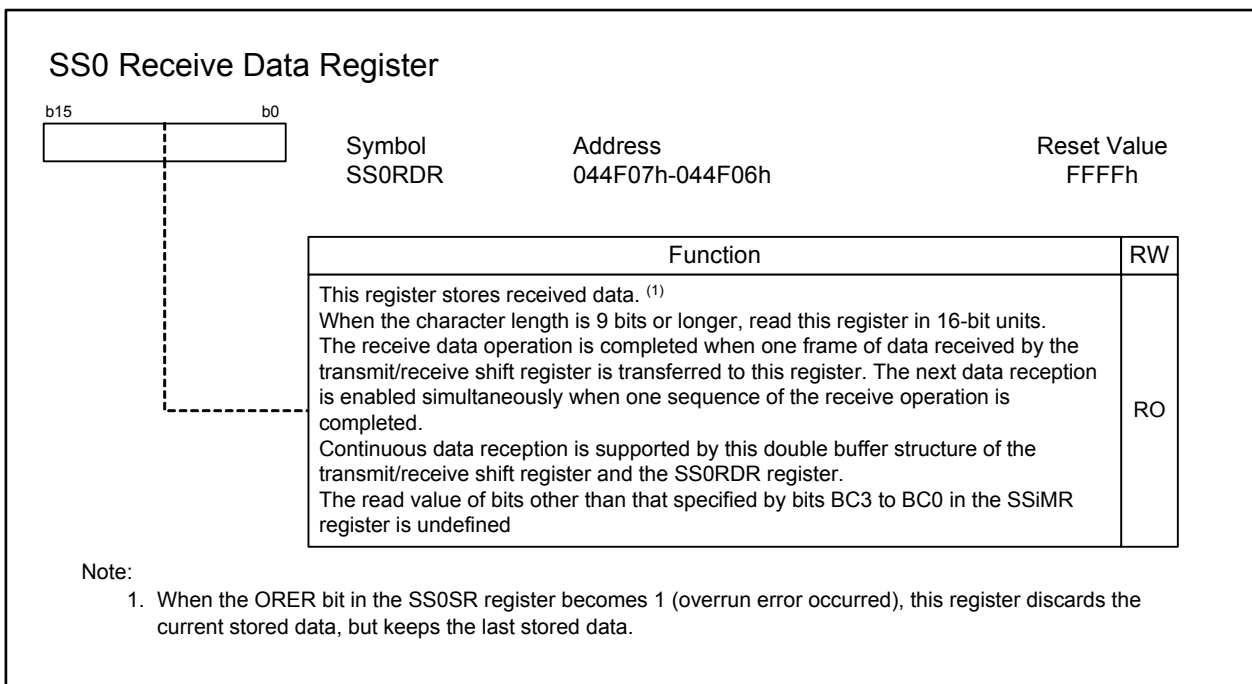


Figure 22.11 SS0RDR Register

### 22.1.1 Transmit/Receive Clock

To use the serial bus interface in synchronous serial communication mode or 4-wire serial bus mode, set the SCKS bit in the SS0MR2 register to 1 to configure the SSCK0 pin to be a transmit/receive clock pin.

(1) To use the serial bus interface in master mode

When the MSS bit in the SS0CRH register is 1 (master mode), select a transmit/receive clock from among seven internal clocks ( $f(\text{BCLK})$  divided-by-4, -8, -16, -32, -64, -128, and -256) with bits CKS2 to CKS0.

Set the corresponding pin to SSCK0 output by the output function select register and set its direction to output by the port direction register.

Since the SSCK0 pin functions as clock output, the selected transmit/receive clock is output from the SSCK0 pin as soon as data transmission/reception starts.

(2) To use the serial bus interface in slave mode

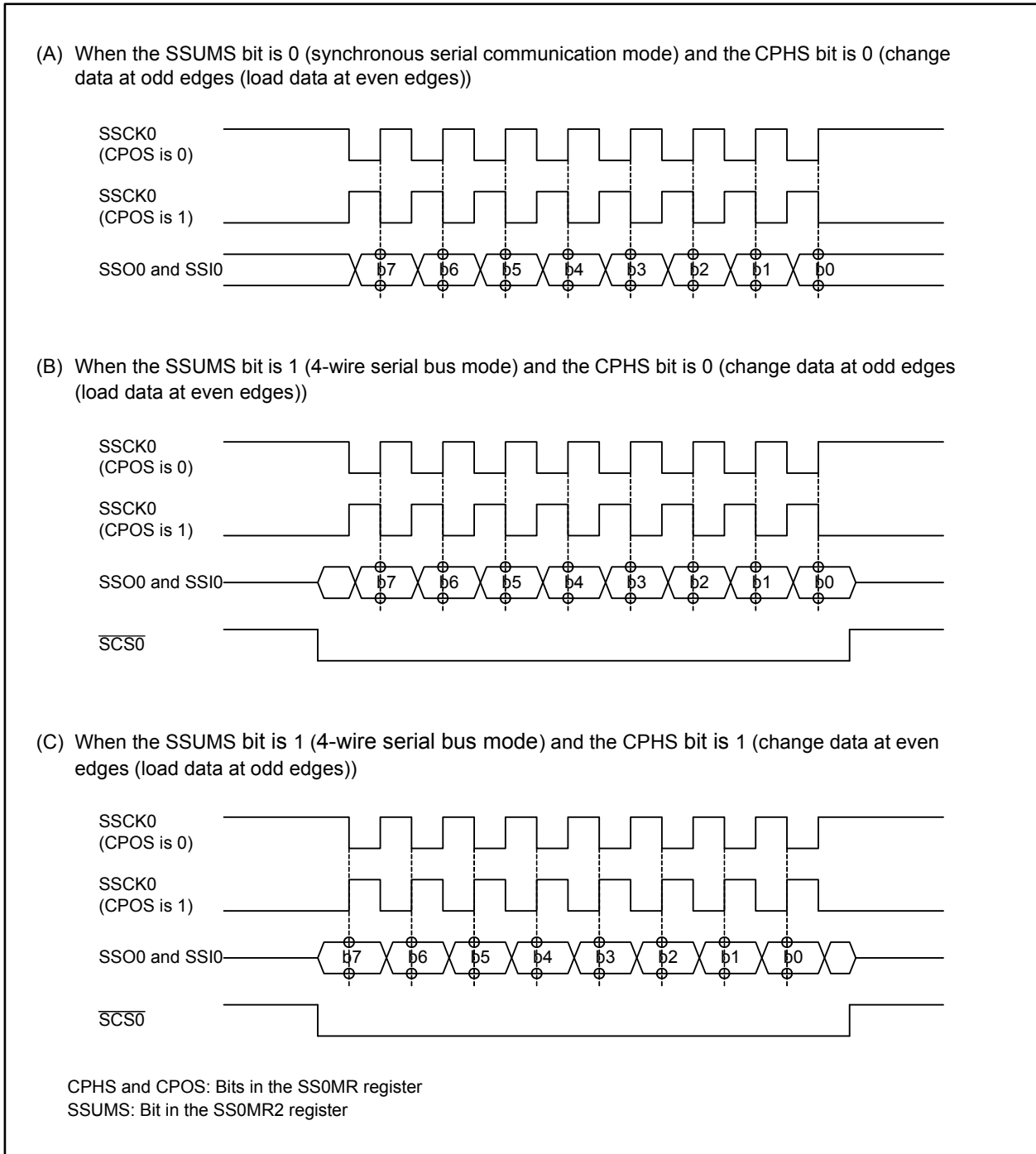
When the MSS bit in the SS0CRH register is 0 (slave mode), the external clock is used.

Set the corresponding pin to input by the port direction register. The SSCK0 pin functions as clock input.

### 22.1.1.1 Transmit/Receive Clock Polarity and Phase

The relation between the transmit/receive clock polarity and phase for transmit/receive data varies with the setting combination of the SSUMS bit in the SS0MR2 register, and bits CPHS and CPOS in the SS0MR register.

Figure 22.12 shows the relation between those parameters.



**Figure 22.12 Transmit/Receive Clock Polarity and Phase of Transmit/Receive Data**

## 22.1.2 Transmit/Receive Shift Register

The transmit/receive shift register transmits and receives serial data. Data transferred to the transmit/receive shift register is shifted-out with the MSB first. Received data is shifted-in with the LSB first.

Figure 22.13 shows transmit/receive shift register operation.

### 22.1.2.1 Bit Order

The bit order to be transmitted/received can be selected using the MLS bit in the SS0MR register. When the MLS bit is 1 (LSB first), transmission starts from the LSB and completes with the MSB. In a receive operation, the first received bit is considered to be the LSB. When the MLS bit is 0 (MSB first), the bit order transmitted is reversed and the first received bit is considered to be the MSB.

#### (1) Transmit operation

When the MLS bit is 0 (MSB first), the written value is reflected in the SS0TDR register in the order written.

When the MLS bit is 1 (LSB first), the written value is reflected in the SS0TDR register in the reversed order, that is, the reversed written value is read from the SS0TDR register.

In both cases, however, data is transferred to the transmit/receive shift register in the transmitted order.

#### (2) Receive operation

When the MLS bit is 0 (MSB first), the received data is transferred from the transmit/receive shift register to the SS0RDR register in the order received.

When the MLS bit is 1 (LSB first), the received data is transferred from the transmit/receive shift register to the SS0RDR register in the reversed order.

### 22.1.2.2 Variable-Length Data Transmission/Reception

When the data length is less than 16 bits, the shift-in/shift-out position is changed to adjust the bit position.

This function enables transmit/receive data with the LSB in bit 0.



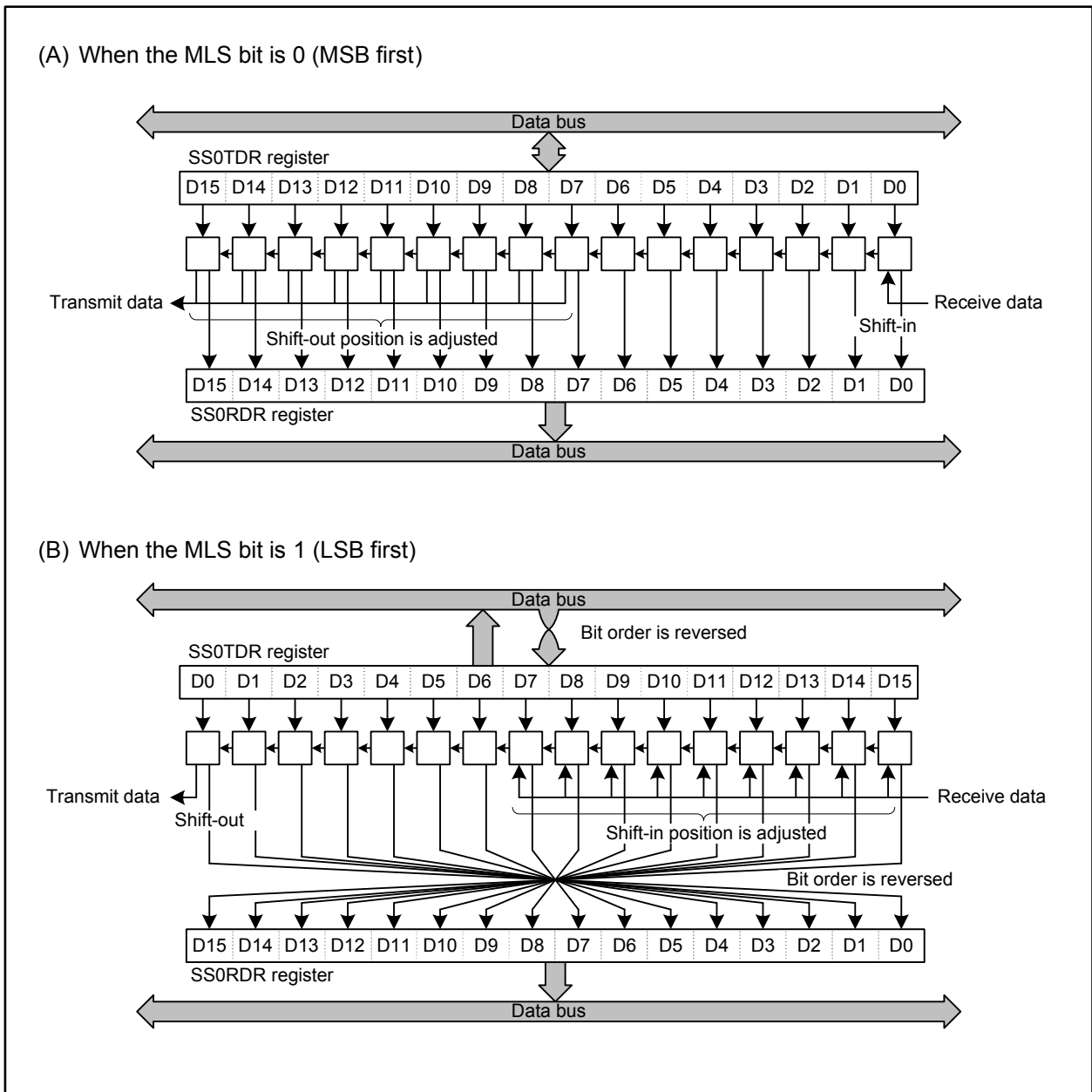


Figure 22.13 Transmit/Receive Shift Register Operation

### 22.1.3 Data I/O Pin and SS0 Shift Register

The connection pattern of the data I/O pins and the transmit/receive shift register varies with the setting combination of the MSS bit in the SS0CRH register, and bits SSUMS and BIDE in the SS0MR2 register. Figure 22.14 shows the relation between those parameters.

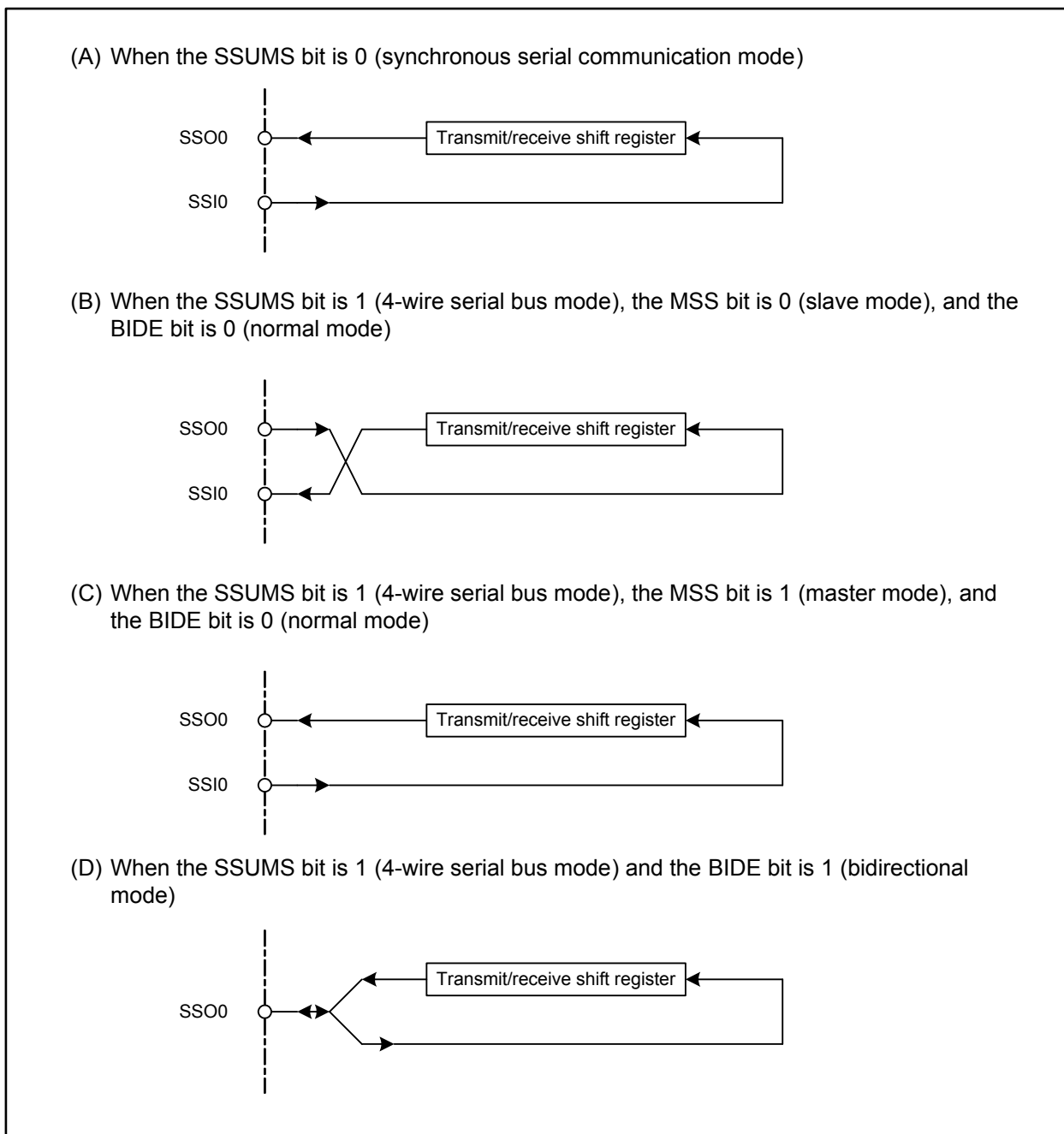


Figure 22.14 Data I/O Pins and Transmit/Receive Shift Register

### 22.1.4 Interrupt Requests

In synchronous serial communication mode and 4-wire serial bus mode, there are five interrupt request sources: transmit data register empty, transmit end, receive data register full, overrun error, and conflict error. Since all of these interrupt request sources are assigned to an interrupt vector table, they should be identified by flags in the interrupt handler.

Table 22.3 lists the serial bus interface interrupt request sources.

**Table 22.3 Interrupt Request Sources in Synchronous Serial Communication Mode and 4-wire Serial Bus Mode**

Interrupt Request Source	Generating Condition for Interrupt Requests (1, 2)
Transmit data register empty (TXI)	TIE = 1 and TDRE = 1
Transmit end (TEI)	TEIE = 1 and TEND = 1
Receive data register full (RXI)	RIE = 1 and RDRF = 1
Overrun error (OEI)	RIE = 1 and ORER = 1
Conflict error (CEI)	CEIE = 1 and CE = 1

Notes:

1. CEIE, RIE, TEIE, and TIE: Bits in the SS0ER register
2. CE, ORER, RDRF, TEND, and TDRE: Bits in the SS0SR register

When one or more conditions to generate an interrupt request listed in Table 22.3 are met, a serial bus interface interrupt request is generated. The corresponding interrupt sources should be set to 0 in the interrupt handler.

However, bits TDRE (transmit data register empty) and TEND (transmit end) automatically become 0 when the next transmit data is written to the SS0TDR register. The RDRF bit (receive data register full) automatically becomes 0 when the SS0RDR register is read.

Note that if no data is being transmitted, the TDRE bit, which became 0 as mentioned above, is set back to 1 (no data left in the SS0TDR register) since the written data is immediately transferred to the transmit/receive shift register. And, if the TDRE bit is set to 0 (unsent data left in SS0TDR register) by a program, it may cause the transmission of an additional data frame.

### 22.1.5 Communication Modes and Pin Functions

In synchronous serial communication mode and 4-wire serial bus mode, the function of I/O pins varies with the setting combinations of the MSS bit in the SS0CRH register, and bits RE and TE in the SS0ER register. The SSCK0 pin is used for output in master mode, and input in slave mode. The port direction register and the port function select register need to be configured according to the direction of pins SSI0, SSO0, and SSCK0.

Table 22.4 shows the relation between communication modes and I/O pins.

**Table 22.4 Communication Modes and I/O Pins**

Communication Mode	Bit Setting					Pin State		
	SSUMS	BIDE	MSS	TE	RE	SSI0	SSO0	SSCK0
Synchronous serial communication mode	0	Disabled	0 (slave)	0	1	Input	– (1)	Input
				1	0	– (1)	Output	
				1	1	Input	Output	
			1 (master)	0	1	Input	– (1)	Output
				1	0	– (1)	Output	
				1	1	Input	Output	
4-wire serial bus mode (in standard mode)	1	0	0 (slave)	0	1	– (1)	Input	Input
				1	0	Output	– (1)	
				1	1	Output	Input	
			1 (master)	0	1	Input	– (1)	Output
				1	0	– (1)	Output	
				1	1	Input	Output	
4-wire bus communication mode (in bidirectional mode) (2)	1	1	0 (slave)	0	1	– (1)	Input	Input
				1	0	– (1)	Output	
			1 (master)	0	1	– (1)	Input	Output
				1	0	– (1)	Output	

**Notes:**

1. This pin can be used as a programmable I/O port.
2. In 4-wire bus communication mode (in bidirectional mode), do not use the following combination: TE bit is 1 and RE bit is 1.

SSUMS and BIDE: Bits in the SS0MR2 register

MSS: Bit in the SS0CRH register

TE and RE: Bits in the SS0ER register

## 22.1.6 Synchronous Serial Communication Mode

### 22.1.6.1 Initialization of Synchronous Communication Mode

Figure 22.15 shows the initialization procedure of synchronous serial communication mode. Set the TE bit in the SS0ER register to 0 (transmission disabled) and the RE bit in the SS0ER register to 0 (reception disabled) before the data transmit/receive operation.

To change the communication mode or the communication data format, set bits TE and RE to 0 in advance. Note that flags RDRF and ORER and the SS0RDR register are unaffected even if the RE bit is set to 0.

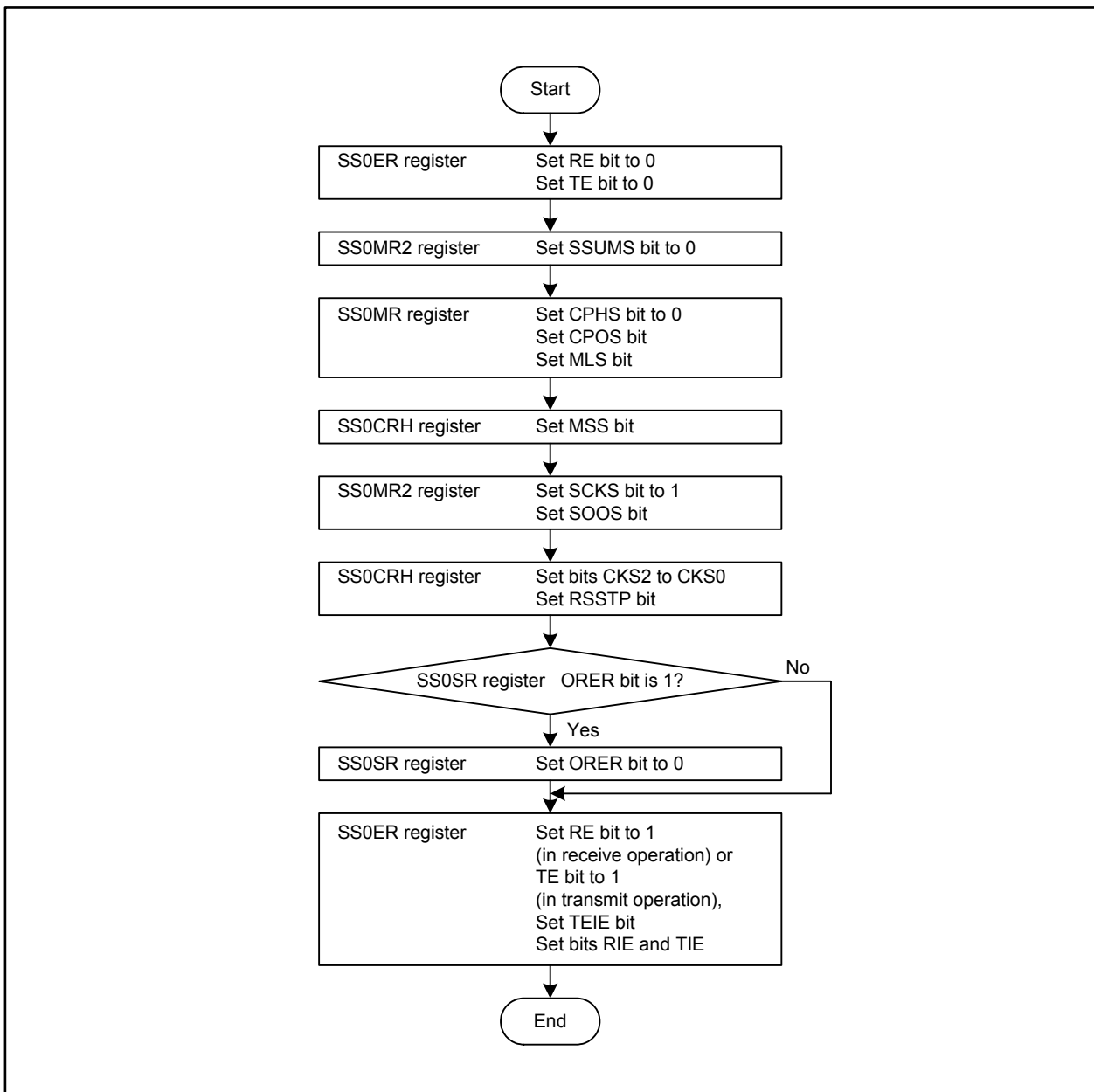


Figure 22.15 Initialization Procedure in Synchronous Communication Mode

### 22.1.6.2 Data Transmission

Figure 22.16 shows an example of the transmit operation in synchronous serial communication mode. During data transmission, the serial bus interface operates as described below.

When operating in master mode, an internally generated clock as a transmit/receive clock is output from the SSCK0 pin. When operating in slave mode, the transmit/receive clock is a clock input from the SSCK0 pin. In both cases, transmit data is output synchronized with the transmit/receive clock.

After setting the TE bit in the SS0ER register to 1 (transmission enabled), if transmit data is written to the SS0TDR register, the TDRE bit in the SS0SR register automatically becomes 0 (unsent data left in the SS0TDR register), and data in the SS0TDR register is transferred to the transmit/receive shift register. Then the TDRE bit becomes 1 (no data left in the SS0TDR register) and data transmission starts. And, if the TIE bit in the SS0ER register is 1 (transmit data register empty interrupt enabled), a transmit data empty interrupt request (TXI) is generated.

When the TDRE bit is 0, after transmitting one frame of data, the data is transferred from the SS0TDR register to the transmit/receive shift register to start the next data transmission. When the TDRE bit is 1, after transmitting the eighth bit of data, the TEND bit in the SS0SR register becomes 1 (transmission is completed). And, if the TEIE bit in the SS0ER register is 1 (transmit end interrupt enabled), a transmit end interrupt request (TEI) is generated.

After data transmission is completed, the SSCK0 pin is fixed to the level specified by the CPOS bit in the SS0MR register.

Note that data cannot be transmitted while the ORER bit in the SS0SR register is 1 (overrun error occurred). Set the ORER bit to 0 before data transmission.

Figure 22.17 shows an example of data transmission procedure in synchronous serial communication mode.

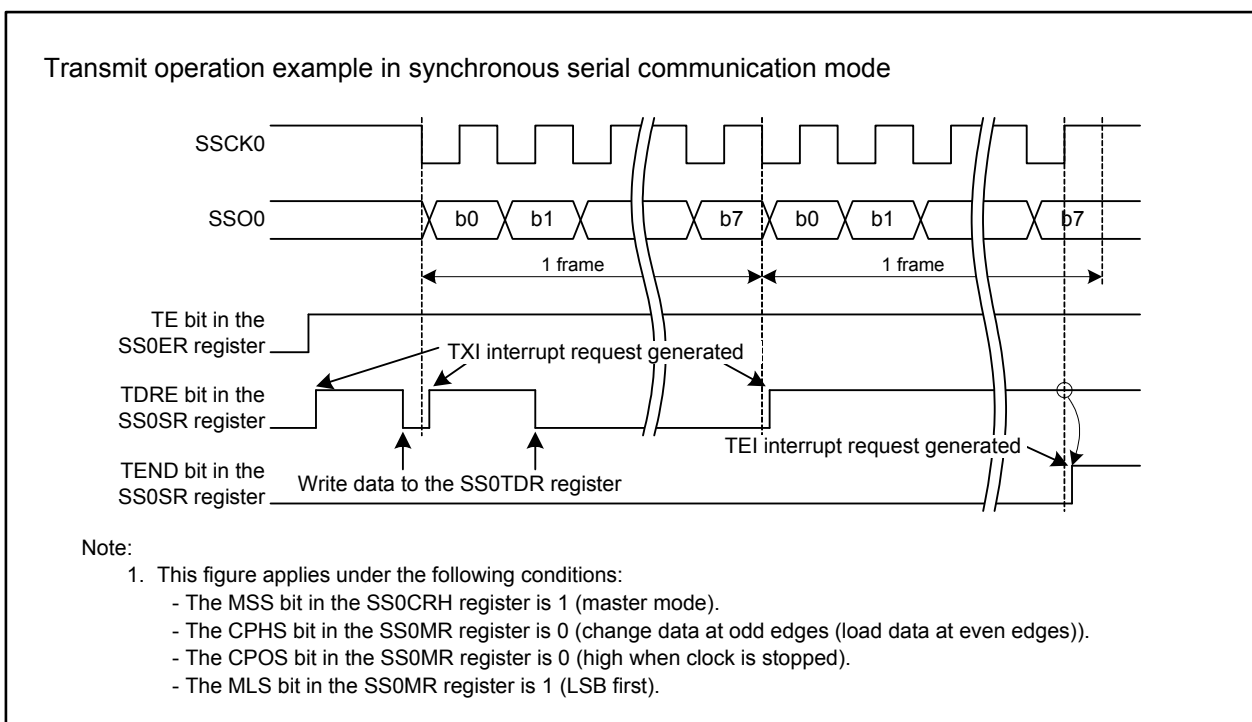
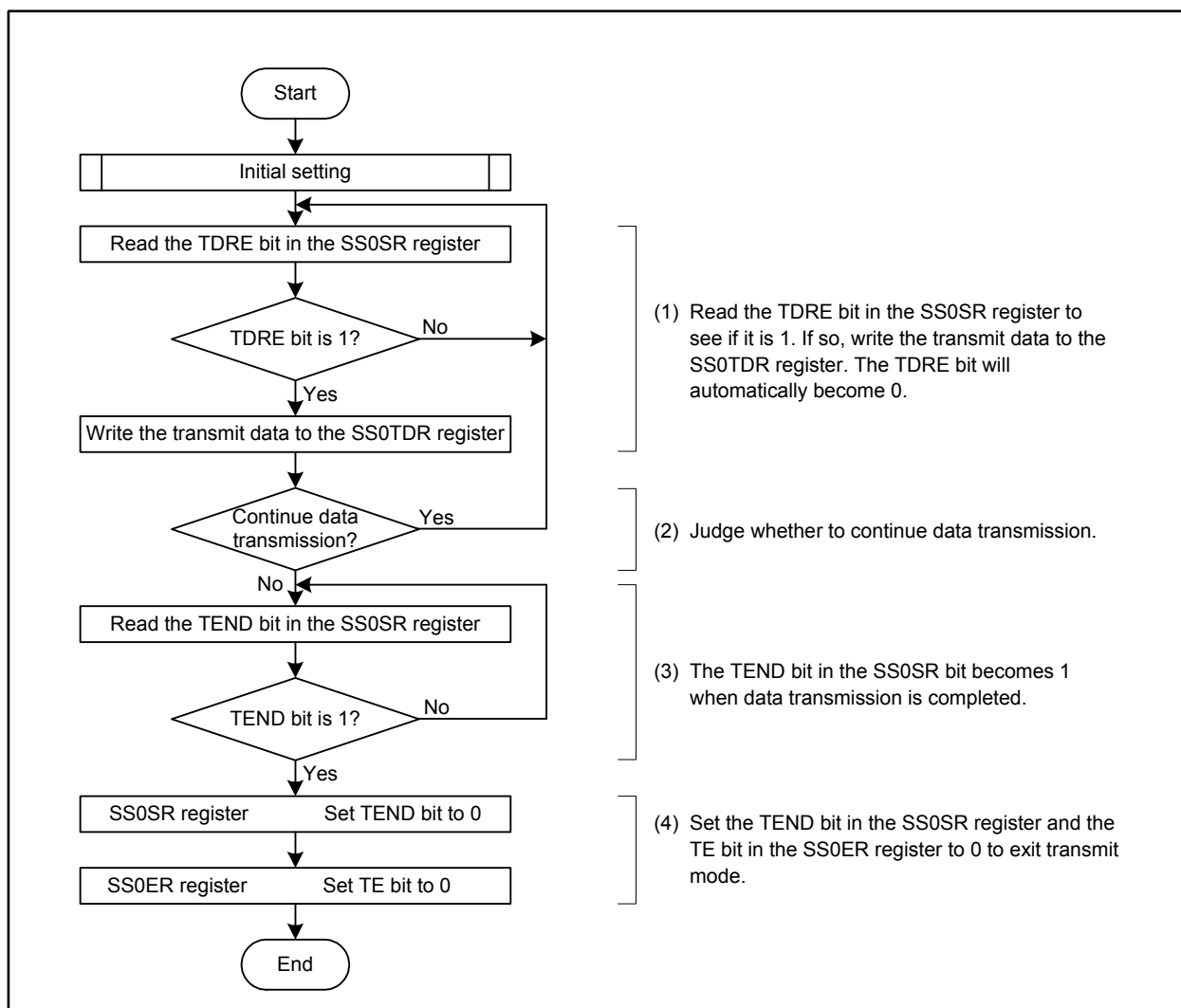


Figure 22.16 Transmit Operation Example in Synchronous Serial Communication Mode



**Figure 22.17 Data Transmission Example in Synchronous Serial Communication Mode**

### 22.1.6.3 Data Reception

Figure 22.18 shows an example of the receive operation in synchronous serial communication mode. During data reception, the serial bus interface operates as described below.

When operating in master mode, an internally generated clock as transmit/receive clock is output from the SSCK0 pin. When operating in slave mode, the transmit/receive clock is a clock input from the SSCK0 pin. In both cases, receive data is input synchronized with the transmit/receive clock.

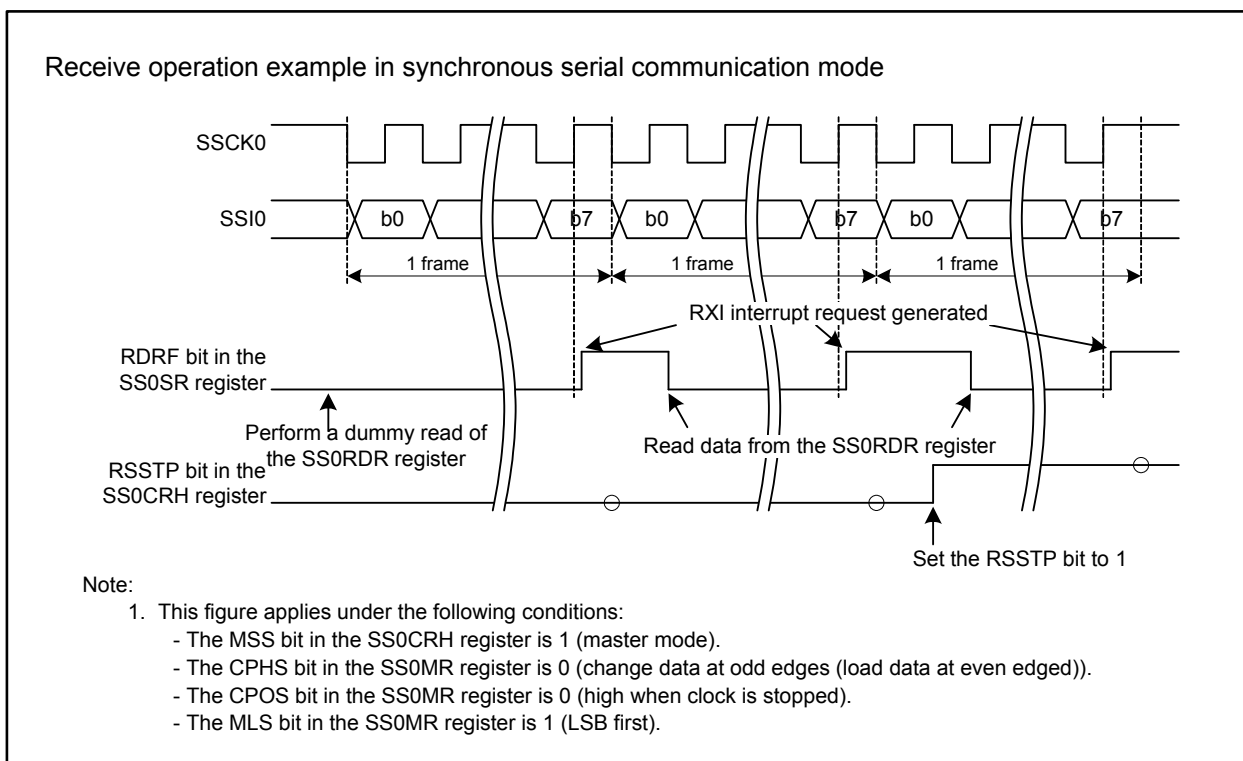
In master mode, a dummy read of the SS0RDR register evokes the receive clock output to start data reception.

When the eighth bit of data is received, the RDRF bit in the SS0SR register becomes 1 (received data left in the SS0RDR register) and the received data is stored into the SS0RDR register. And, if the RIE bit in the SS0ER register is 1 (receive data register full interrupt/overflow interrupt enabled), a receive data register full interrupt request (RXI) is generated. When the SS0RDR register is read, the RDRF bit automatically becomes 0 (no data left in the SS0RDR register).

To end the receive data operation in master mode, set the RSSTP bit in the SS0CRH register to 1 (receive operation ended after receiving the current frame) before reading the data of the second to last frame from the SS0RDR register. Then the transmit/receive clock stops when the last frame reception is completed. When the clock stops, set the RE bit in the SS0ER register to 0 (reception disabled) and the RSSTP bit to 0 (receive operation continued after receiving the current frame) and read the data of the last frame. Note that if the SS0RDR register is read while the RE bit is 1, then the receive clock is output again.

If the eighth bit of the data is received when the RDRF bit is 1, the ORER bit in the SS0SR register becomes 1 (overflow error occurred), and the receive operation stops. Note that data cannot be received while the ORER bit is 1. Set the ORER bit to 0 before resuming data reception.

Figure 22.19 shows an example of the data reception procedure in master mode of synchronous communication mode.



**Figure 22.18 Receive Operation Example in Master Mode of Synchronous Serial Communication Mode**



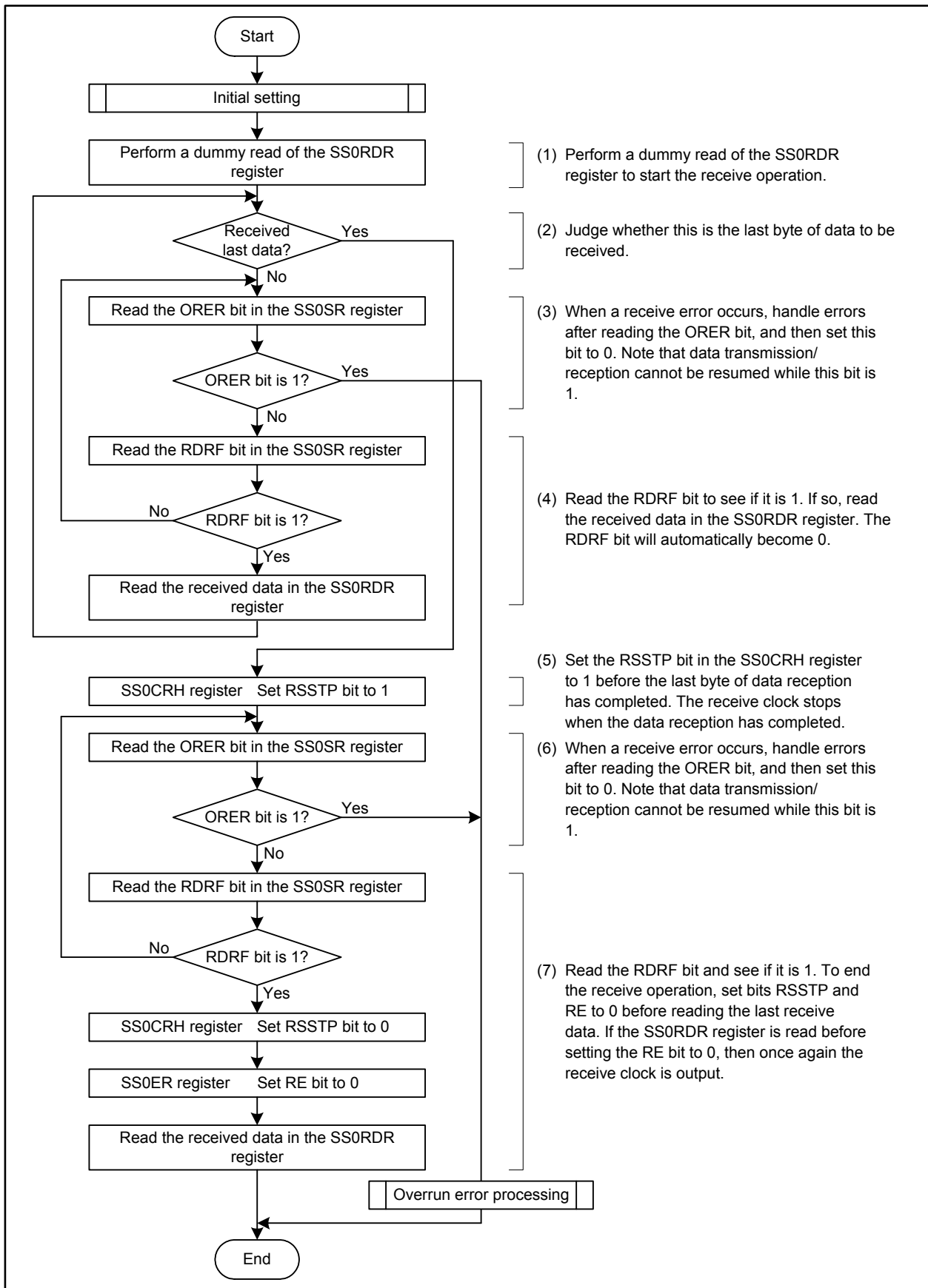


Figure 22.19 Data Reception Example in Master Mode of Synchronous Serial Communication Mode

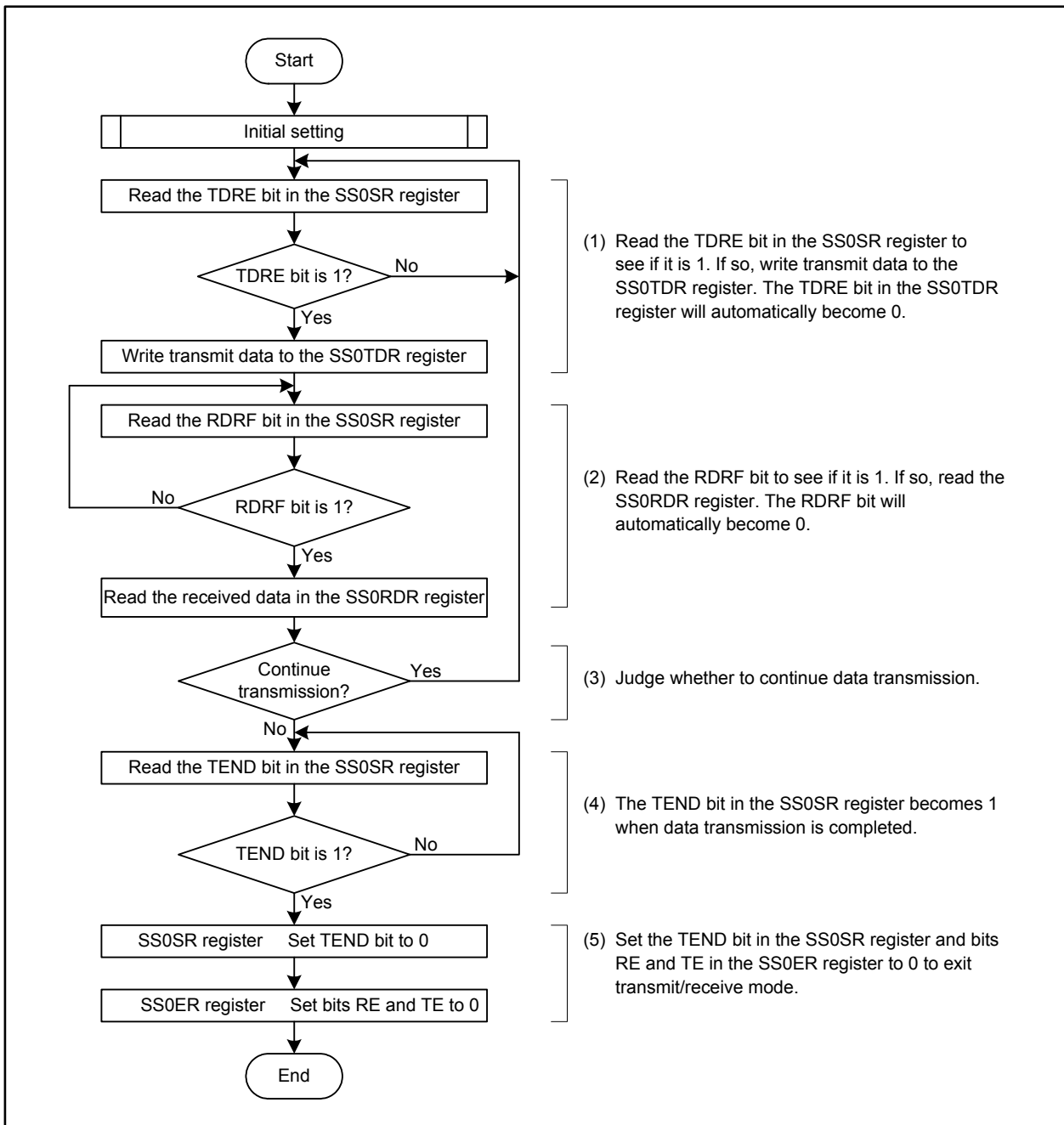
#### 22.1.6.4 Data Transmission/Reception

Data transmission/reception is a combined operation of data transmission and data reception mentioned in sections 22.1.6.2 and 22.1.6.3.

Data transmission/reception starts when data is written to the SS0TDR register, and ends when the eighth bit of data is transmitted while the TDRE bit is 1 (no data left in the SS0TDR register). The data transmit/receive operation stops when the ORER bit becomes 1 (overflow error occurred) due to an error.

To switch modes from transmit mode (TE bit is 1 and RE bit is 0) or receive mode (TE bit is 0 and RE bit is 1) to transmit/receive mode (TE bit is 1 and RE bit is 1), set bits TE and RE to 0. Then, check that the TEND bit is 0 (transmission continued), the RDRF bit is 0 (no data left in the SS0RDR register), the ORER bit is 0 (no overflow error), and subsequently set bits TE and RE to 1 at the same time.

Figure 22.20 shows an example of data transmission/reception procedure in synchronous serial communication mode.



**Figure 22.20 Data Transmission/Reception Example in Synchronous Communication Mode**

### 22.1.7 4-wire Serial Bus Mode

In 4-wire serial bus mode, the serial bus interface operates serial communication via four logic signal lines: clock line, data input line, data output line, and chip select line. This mode also contains bidirectional mode, in which a single pin is used for the data input line and data output line.

The data input line and data output line share the SSI0 pin and SSO0 pin. The combined setting of the MSS bit in the SS0CRH register and the BIDE bit in the SS0MR2 register determines which line is assigned to which pin. Refer to 22.1.3 “Data I/O Pin and SS0 Shift Register” for more details. In this mode, the clock polarity and phase for transmit/receive data can be configured by setting bits CPOS and CPHS in the SS0MR register, respectively. Refer to 22.1.1.1 “Transmit/Receive Clock Polarity and Phase” for more details.

The chip select line functions as an output pin in master mode and as an input pin in slave mode. In master mode, the  $\overline{\text{SCS0}}$  pin can be assigned as an output pin by setting bits CSS1 and CSS0 in the SS0MR2 register to 10b. Also, a programmable I/O port can be assigned as the chip select pin in this mode. In slave mode, the  $\overline{\text{SCS0}}$  pin is assigned as an input pin by setting bits CSS1 and CSS0 to 01b. In 4-wire serial bus mode, serial communication is normally performed with the MSB first (the MLS bit in the SS0MR register is set to 0).

#### 22.1.7.1 Initialization of 4-wire Serial Bus Mode

Figure 22.21 shows the initialization of 4-wire serial bus mode. Initialize the circuit by setting the TE bit in the SS0ER register to 0 (transmission disabled) and the RE bit in the SS0ER register to 0 (reception disabled) before a data transmission/reception.

To change the communication mode or the communication data format, set bits TE and RE to 0 in advance. Note that bits RDRF and ORER and the SS0RDR register are unaffected even if the RE bit is set to 0.

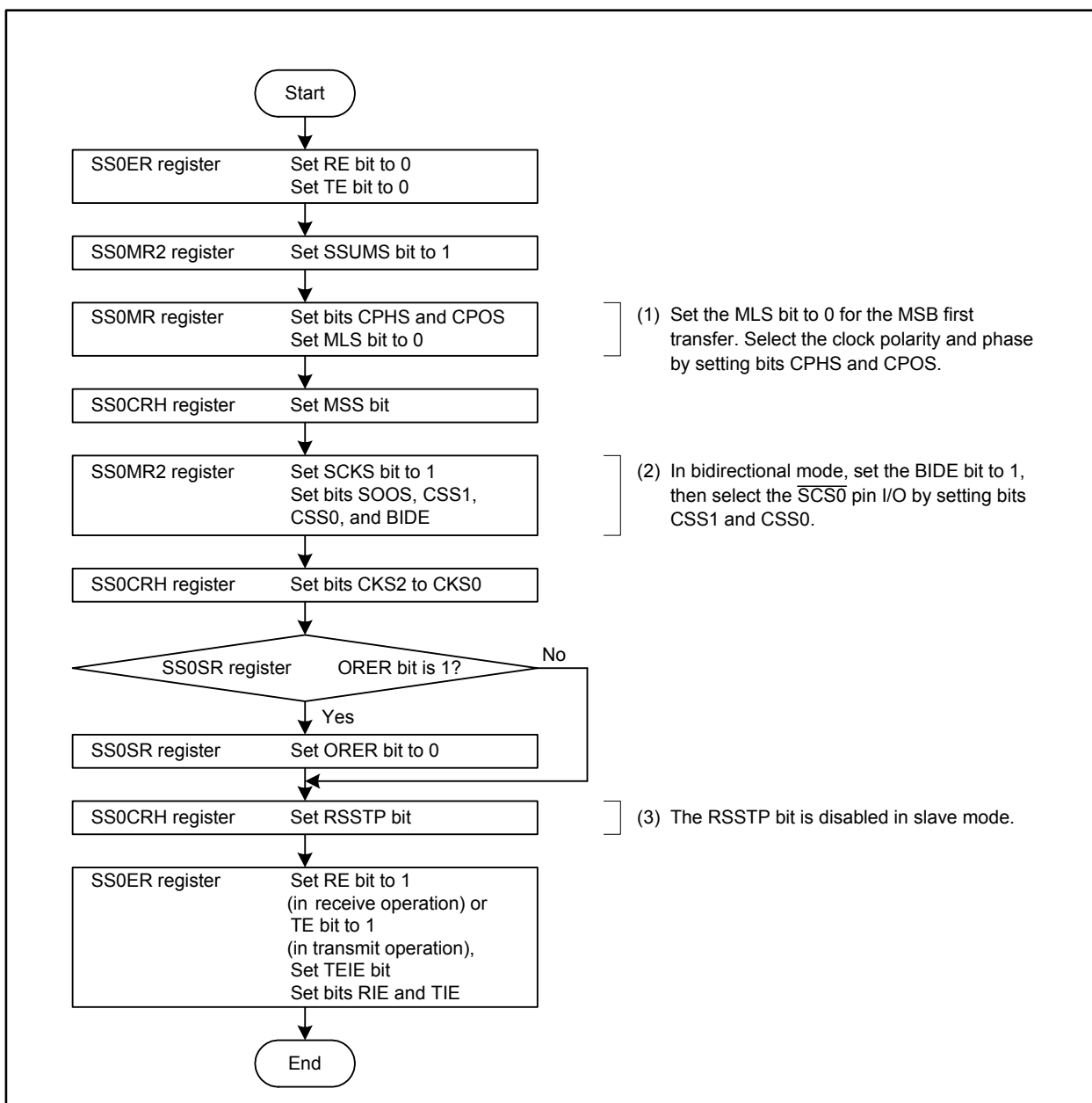


Figure 22.21 Initialization in 4-wire Serial Bus Mode

### 22.1.7.2 Data Transmission

Figure 22.22 shows an example of the transmit operation in 4-wire serial bus mode. During data transmission, the serial bus interface operates as described below.

When operating in master mode, an internally generated clock as a transmit/receive clock is output from the SSCK0 pin. When operating in slave mode, the transmit/receive clock is a clock input from the SSCK0 pin while the  $\overline{\text{SCS0}}$  pin is low. In both cases, transmit data is output synchronized with the transmit/receive clock.

After setting the TE bit in the SS0ER register to 1 (transmission enabled), if transmit data is written to the SS0TDR register, the TDRE bit in the SS0SR register automatically becomes 0 (unsent data left in the SS0TDR register), and data in the SS0TDR register is transferred to the transmit/receive shift register. Then the TDRE bit becomes 1 (no data left in the SS0TDR register) and data transmission starts. And, if the TIE bit in the SS0ER register is 1 (transmit data register empty interrupt enabled), a transmit data empty interrupt request (TXI) is generated.

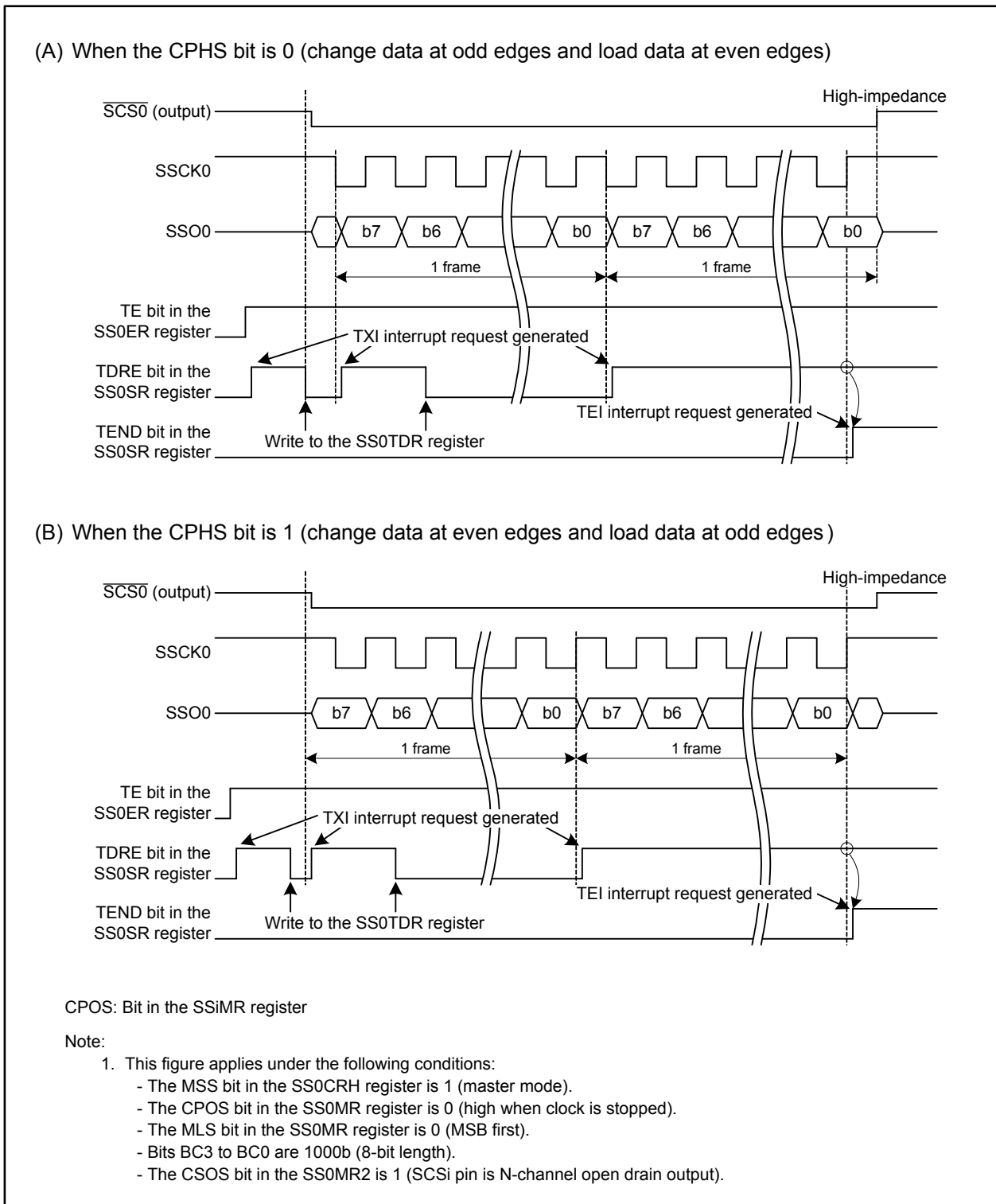
When the TDRE bit is 0, after transmitting one frame of data, the data is transferred from the SS0TDR register to the transmit/receive shift register to start the next data transmission. When the TDRE bit is 1, after transmitting the last bit of data, the TEND bit in the SS0SR register becomes 1 (transmission completed). And, if the TEIE bit in the SS0ER register is 1 (transmit end interrupt enabled), a transmit end interrupt request (TEI) is generated.

After the data transmission is completed, the SSCK0 pin and the  $\overline{\text{SCS0}}$  pin become high. To continue data transmission with the  $\overline{\text{SCS0}}$  pin low, write the data to be transmitted next to the SS0TDR register before the last bit is transmitted.

Note that data cannot be transmitted while the ORER bit in the SS0SR register is 1 (overrun error occurred). Set the ORER bit to 0 before data transmission.

In contrast to synchronous serial communication mode, the  $\overline{\text{SCS0}}$  pin is used in 4-wire serial bus mode. When the serial bus interface is in master mode, the SSO0 pin is high-impedance while the output level at the  $\overline{\text{SCS0}}$  pin is high. In slave mode, the SSli pin is high-impedance while the input level at the  $\overline{\text{SCS0}}$  pin is high.

Refer to Figure 22.17 for an example of data transmission procedure.



**Figure 22.22 Transmit Operation Example in 4-wire Serial Bus Mode**

### 22.1.7.3 Data Reception

Figure 22.23 shows an example of receive operation in 4-wire serial bus mode. During data reception, the serial bus interface operates as described below.

When operating in master mode, an internally generated clock as a transmit/receive clock is output from the SSCK0 pin. When operating in slave mode, the transmit/receive clock is a clock input from the SSCKi pin while the  $\overline{\text{SCS0}}$  pin is low. In both cases, receive data is input synchronized with the transmit/receive clock.

In master mode, a dummy read of the SS0RDR register evokes the receive clock output to start data reception. Note that the TE bit in the SS0ER register should be set to 0 before performing the dummy read to exit transmit mode.

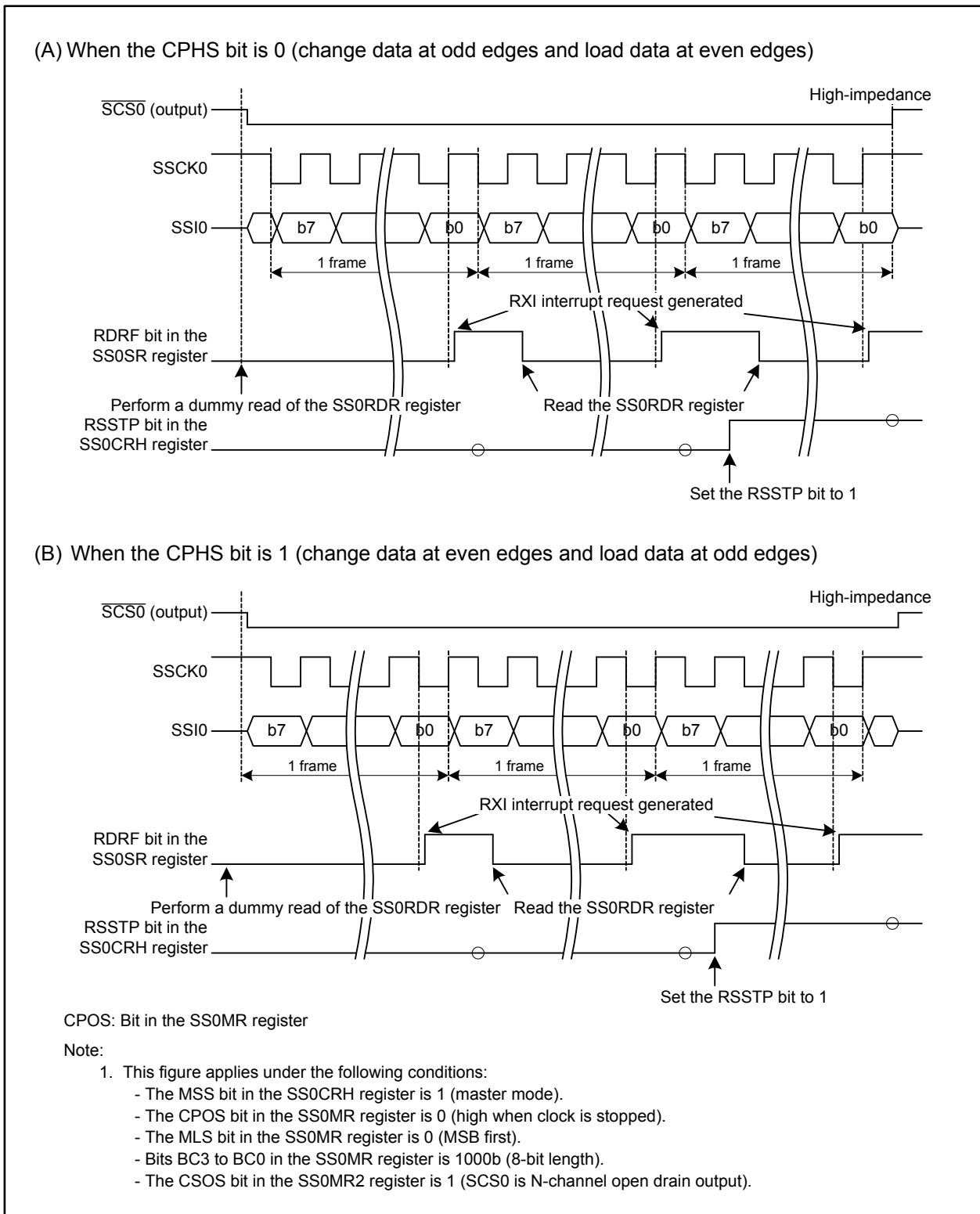
When the specified data length is received, the RDRF bit in the SS0SR register becomes 1 (received data left in the SS0RDR register) and the received data is stored into the SS0RDR register. And, if the RIE bit in the SS0ER register is 1 (receive data register full interrupt/overrun error interrupt enabled), a receive data register full interrupt request (RXI) is generated. When the SS0RDR register is read, the RDRF bit automatically becomes 0 (no data left in the SS0RDR register).

To end the receive data operation in master mode, set the RSSTP bit in the SS0CRH register to 1 (receive operation ended after receiving the current frame) before reading the data of the second to last frame from the SS0RDR register. Then the transmit/receive clock stops when the last frame is received. When the clock stops, set the RE bit in the SS0ER register to 0 (reception disabled) and the RSSTP bit to 0 (receive operation continued after receiving the current frame) and read the data of the last frame. Note that if the SS0RDR register is read while the RE bit is 1, then the receive clock is output again.

If the last bit of the data is received when the RDRF bit is 1, the ORER bit in the SS0SR register becomes 1 (overrun error occurred), and the receive operation stops. Note that data cannot be received while the ORER bit is 1. Set the ORER bit to 0 before resuming data reception.

Refer to Figure 22.19 for an example of data reception procedure.





**Figure 22.23 Receive Operation Example in 4-wire Serial Bus Mode**

#### 22.1.7.4 $\overline{\text{SCS0}}$ Pin Control and Arbitration

In 4-wire serial bus mode, when bits CSS1 and CSS0 are set to 10b (functions as  $\overline{\text{SCS0}}$  output pin), and the MSS bit in the SS0CRH register is 1 (master mode), then arbitration is performed by monitoring the  $\overline{\text{SCS0}}$  pin before starting serial data transmission. When the serial bus interface detects that the synchronized internal  $\overline{\text{SCS0}}$  signal level drops to low before data transmission, the CE bit in the SS0SR register becomes 1 (conflict error occurred) and the MSS bit automatically becomes 0 (slave mode). Figure 22.24 shows the arbitration timing.

Note that the serial bus interface can no longer continue transmitting while the CE bit is 1. To start data transmission, confirm the CE bit is 1 and then set it to 0 (no conflict error).

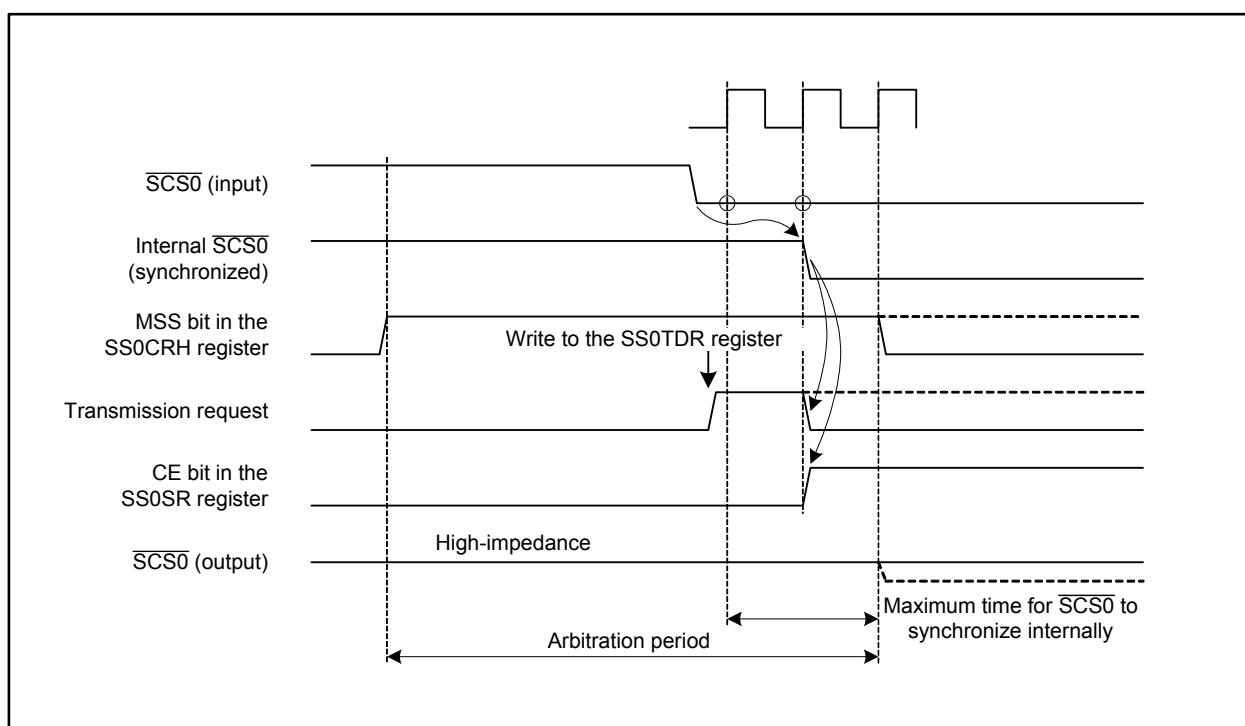


Figure 22.24 Arbitration Timing

## 22.2 Note on Serial Bus Interface

### 22.2.1 Note on Synchronous Serial Communication Mode and 4-wire Serial Bus Mode

#### 22.2.1.1 Accessing SBI Associated Registers

To read the SBI associated registers (44F06h to 44F0Fh), insert at least three instructions after writing to the registers.

- An example of inserting at least three instructions

```
Program example    MOV.B  #00h, 44F0Bh ; Set the SS0ER register to 00h.
                   NOP
                   NOP
                   NOP
                   MOV.B  44F0Bh, R0L
```

## 23. LIN Module

The R32C/161 Group implements a local interconnect network (LIN) module, compliant with LIN Protocol Specification Revisions 1.3, 2.0, and 2.1, which transmits and receives frames, and judges errors automatically.

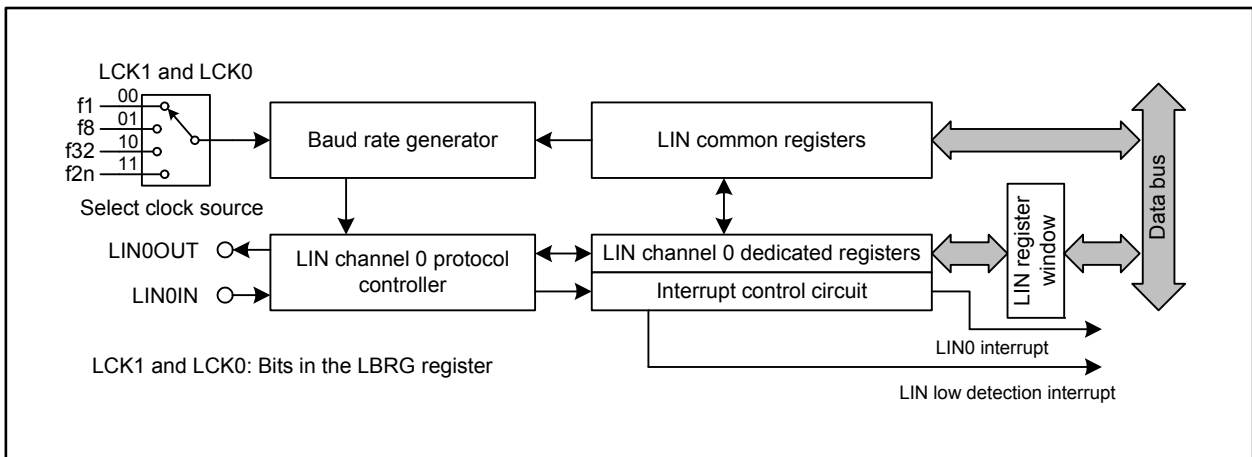
Table 23.1 lists the LIN module specifications and Figure 23.1 shows the LIN module block diagram.

**Table 23.1 LIN Module Specifications**

Item	Specification
Protocols	LIN Protocol Specification Revisions 1.3, 2.0, and 2.1
Channel	1 (LIN master)
Variable frame composition	Transmit break length: 13 to 28 Tbit Transmit break delimiter length: 1 to 4 Tbit Interbyte space (Header): 0 to 7 Tbit (space between Sync field and ID field) <sup>(1)</sup> Response space: 0 to 7 Tbit <sup>(1)</sup> Interbyte space: 0 to 3 Tbit (space between bytes in response space)
Checksum	Classic checksum or enhanced checksum selectable (changeable at each frame)
Data byte(s) in response field	Variable from 0 to 8
Frame transmission modes	Non-frame separate mode or frame separate mode selectable; the former mode transmits a header and response by a single start command and the latter separately transmits them by respective start command
Wake-up transmission/reception	Available in LIN wake-up mode <ul style="list-style-type: none"> <li>• Wake-up transmission (1 to 16 Tbit)</li> <li>• Wake-up reception <ul style="list-style-type: none"> <li>• Input signal low time count (0.5 to 15.5 Tbit)</li> <li>• Input signal low detection</li> </ul> </li> </ul>
Operational status	<ul style="list-style-type: none"> <li>• Frame/wake-up transmit completion</li> <li>• Frame/wake-up receive completion</li> <li>• Data 1 receive completion</li> <li>• Input signal low detection</li> <li>• Error detection</li> <li>• Operating mode</li> </ul>
Error status	<ul style="list-style-type: none"> <li>• Bit error</li> <li>• Checksum error</li> <li>• Frame timeout error</li> <li>• Physical bus error</li> <li>• Framing error</li> <li>• Overrun error</li> </ul> <p>Excluding the checksum error, these errors are detection enable or detection disable selectable</p>
Baud rate	The baud rate generator can generate the baud rates in the LIN specifications

Note:

1. The interbyte space (Header) has the same value as the response space since both spaces are set in the same register.



**Figure 23.1 LIN Module Block Diagram**

- LIN0OUT, LIN0IN: LIN I/O pins
- Baud rate generator: Generates a communication clock for LIN
- LIN common registers: Common registers for the LIN module
- LIN channel 0 dedicated registers:
  - To use these registers, set bits CWS1 and CWS0 in the LCW register to 00
- Interrupt control circuit:
  - Controls the interrupt requests (LIN0 interrupt and LIN low detection interrupt) generated by the LIN module

## 23.1 LIN Module Associated Registers

The LIN module has two types of registers: LIN module common registers and LIN channel dedicated registers.

The latter compose up to four register banks for channels 0 to 3, of which an appropriate register bank is selected by switching a channel to be shown on the window. The number of channels per module depends on total channels incorporated.

### 23.1.1 LIN Common Registers

- LIN channel window select/input signal low detection status register (LCW register)  
This register is to select a channel and monitor the status of input signal low detection.
- LIN baud rate generator control register (LBRG register)  
This register is to select a count source for the baud rate generator and f<sub>LN</sub> clock source. Refer to 23.4 “Baud Rate Generator” for details.
- LIN baud rate prescaler k (LBRPk register) (k = 0, 1)  
This register is to set the division value for the baud rate prescaler to generate a transmit/receive clock with the count source selected by the LBRG register.

### 23.1.2 LIN Channel Dedicated Registers

To use these registers, set bits CWS1 and CWS0 in the LCW register to 00. LIN channel dedicated registers can be set via the following registers:

- LIN mode register 0 (LMD0 register)  
This register is used for the initial setting of LIN module including mode selections and interrupt enable settings.
- LIN mode register 1 (LMD1 register)  
This register is used for the initial setting of LIN module including system clock selection and error detection enable settings.
- LIN wake-up setting register (LWUP register)  
This register sets the low time pulse width of wake-up transmission and reception.
- LIN break field setting register (LBRK register)  
This register sets the transmit break length and transmit break delimiter length in break field.
- LIN space setting register (LSPC register)  
This register sets each space width required for transmission.
- LIN response field setting register (LRFC register)  
This register sets the number of communication data bytes, the transmit/receive direction, and a checksum model.
- LIN ID buffer register (LIDB register)  
This register sets the transmission ID and ID parity bits.
- LIN status control register (LSC register)  
This register changes the LIN operating mode.
- LIN transmission control register (LTC register)  
This register sets to start the frame/wake-up transmission and response field transmission.
- LIN status register (LST register)  
This register monitors the status including frame/wake-up transmit completion, frame/wake-up receive completion, Data 1 receive completion, input signal low detection, error detection, and operating mode.
- LIN error status register (LEST register)  
This register monitors error status caused by bit, checksum, physical bus, frame timeout, framing, and overrun errors).
- LIN data n buffer register (LDBn register) (n = 1 to 8)  
This register, commonly used as transmission and reception data buffer, sets data to be transmitted and to read received data.

### 23.1.3 Register Window and Channel Switching

The LIN channel dedicated registers, which are not directly mapped on the memory map of the CPU, are accessed via a register window. The register window is mapped at addresses 44E04h to 44E17h. According to the setting of bits CWS1 and CWS0 in the LCW register, the LIN channel dedicated registers are collectively mapped on the register window.

Figure 23.2 shows the block diagram of the register window.

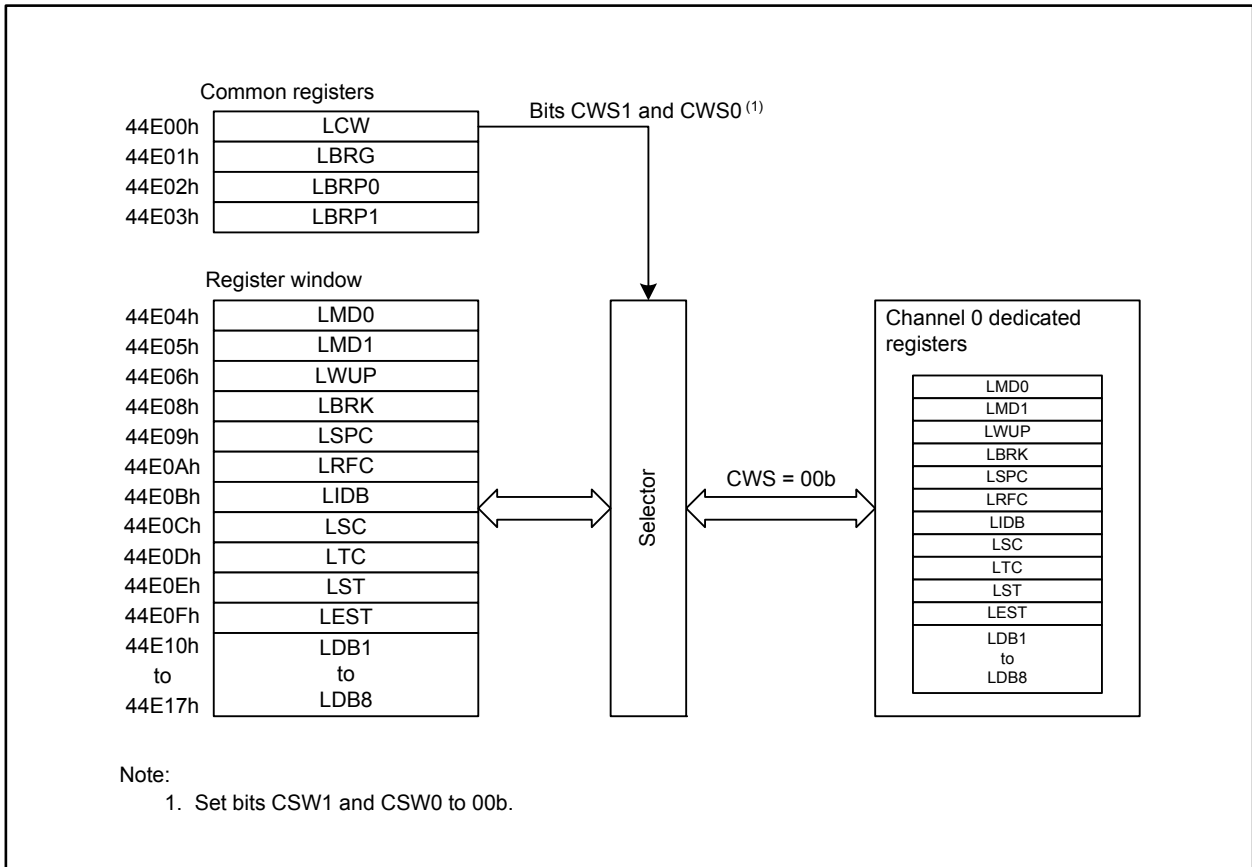
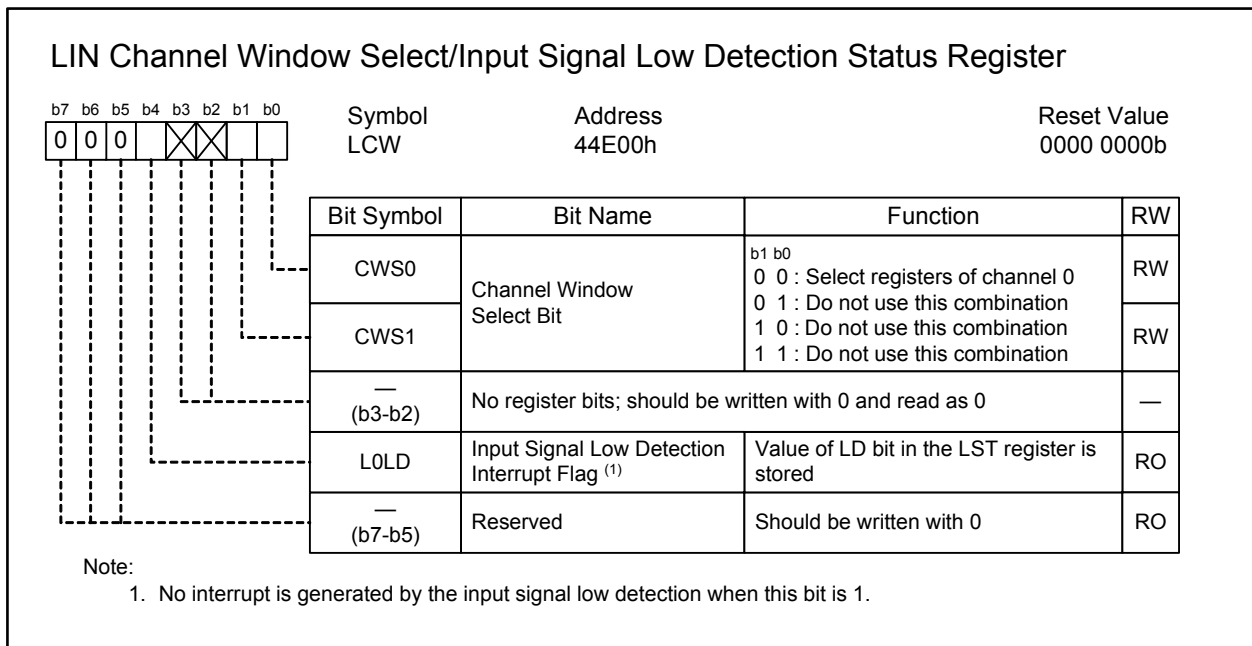
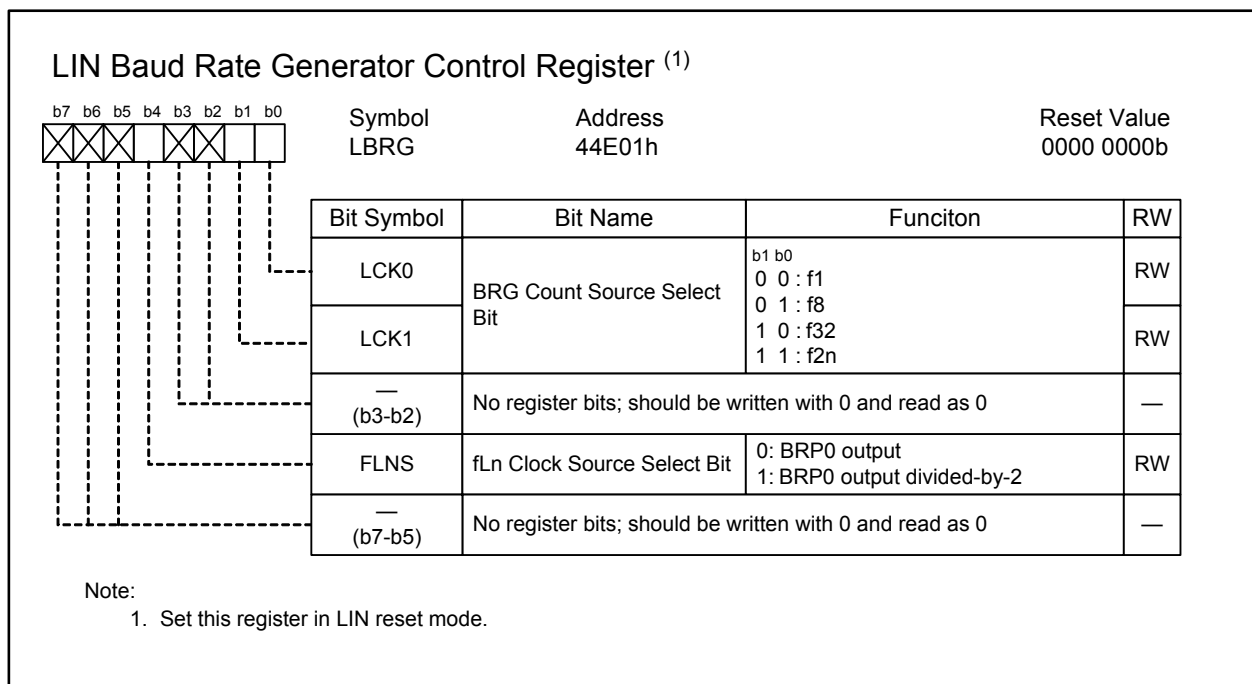


Figure 23.2 Register Window

Figures 23.3 to 23.18 show LIN module associated registers.

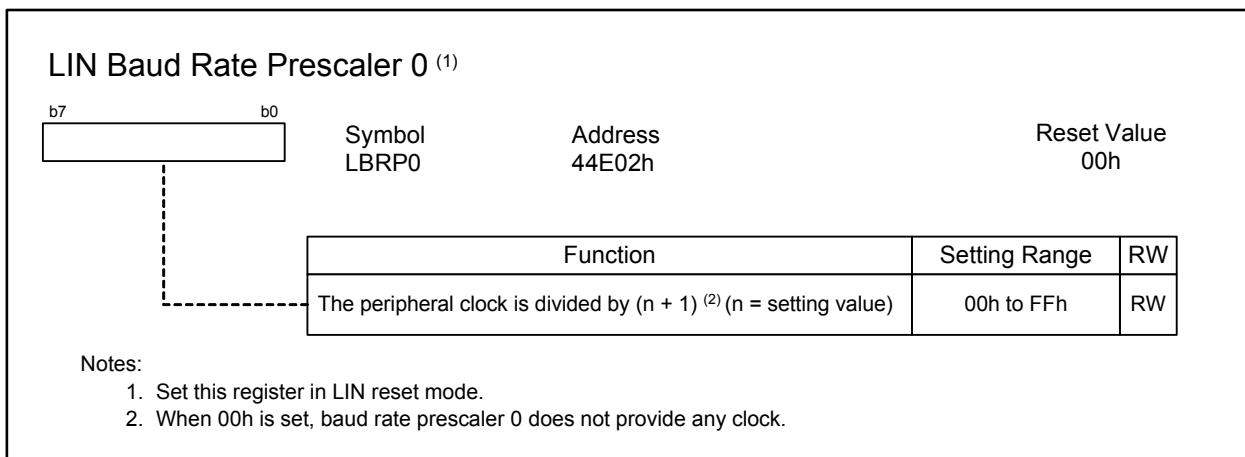
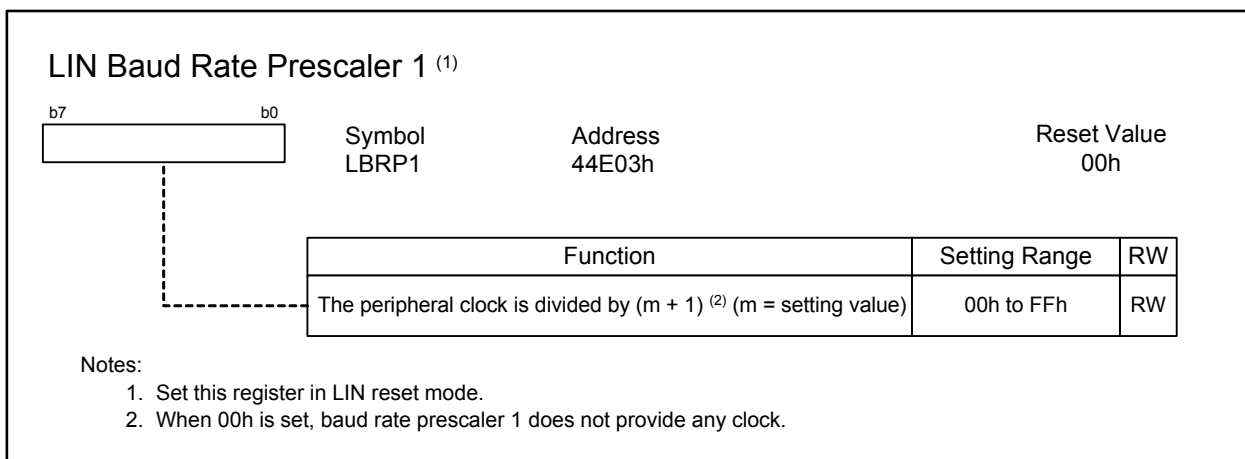


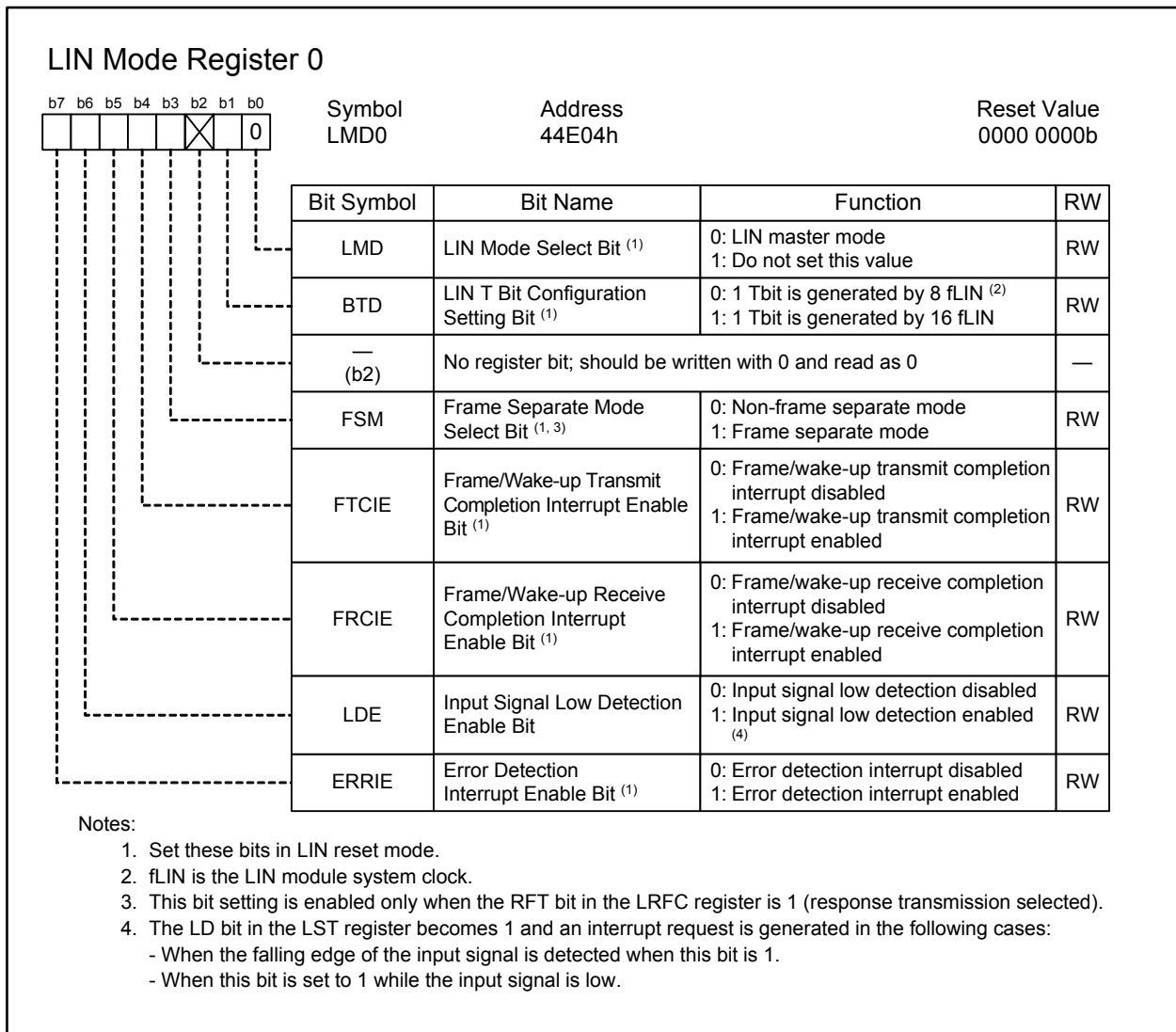
**Figure 23.3 LCW Register**



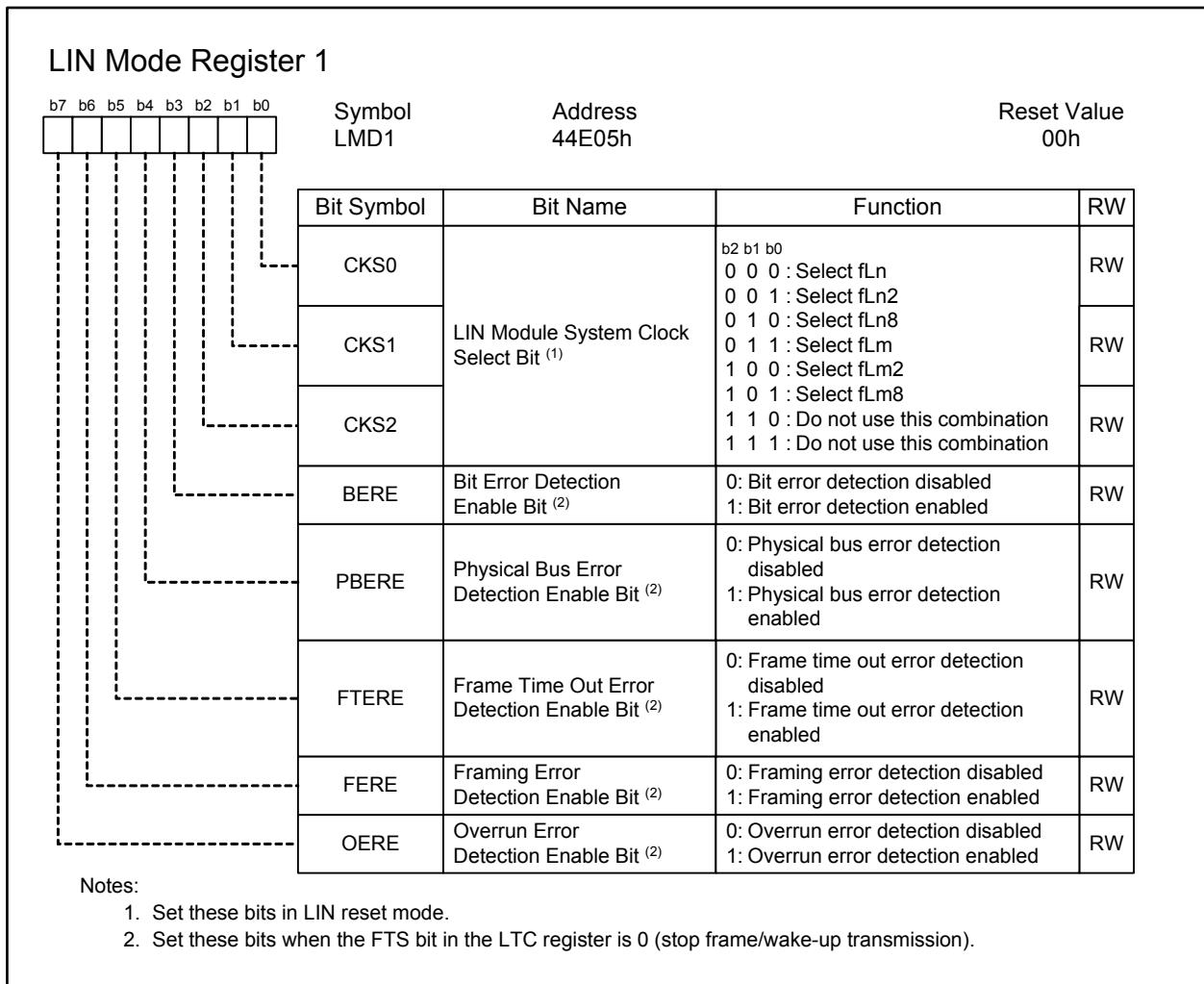
**Figure 23.4 LBRG Register**



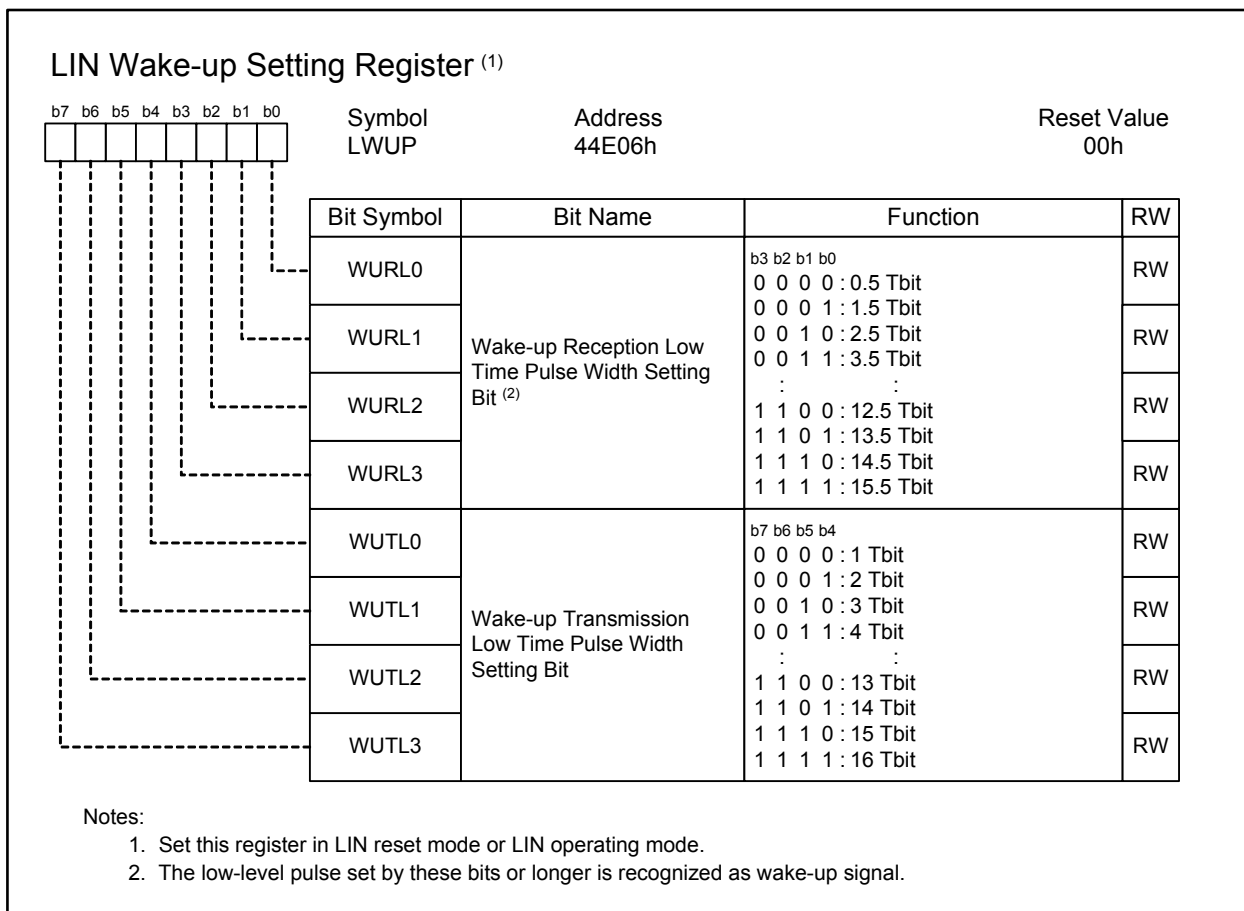
**Figure 23.5 LBRP0 Register****Figure 23.6 LBRP1 Register**



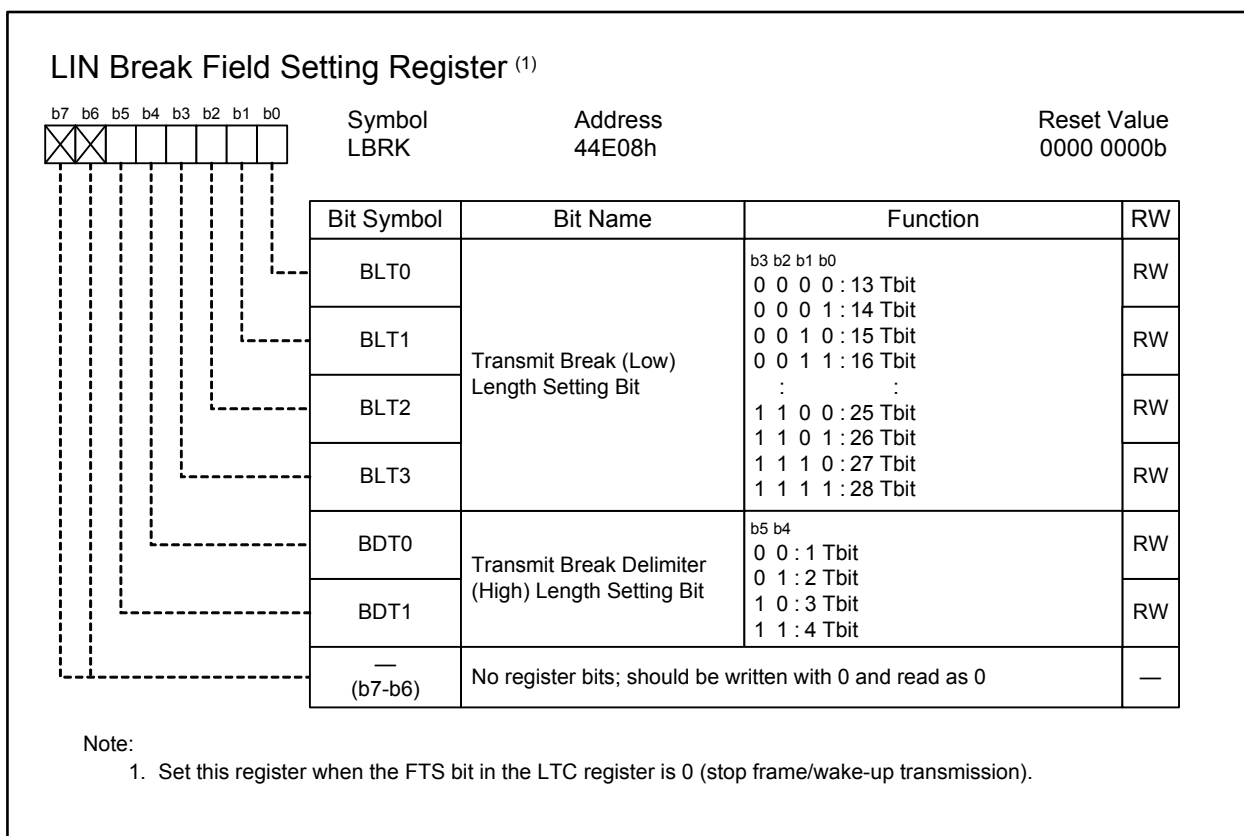
**Figure 23.7 LMD0 Register**



**Figure 23.8 LMD1 Register**



**Figure 23.9 LWUP Register**



**Figure 23.10 LBRK Register**

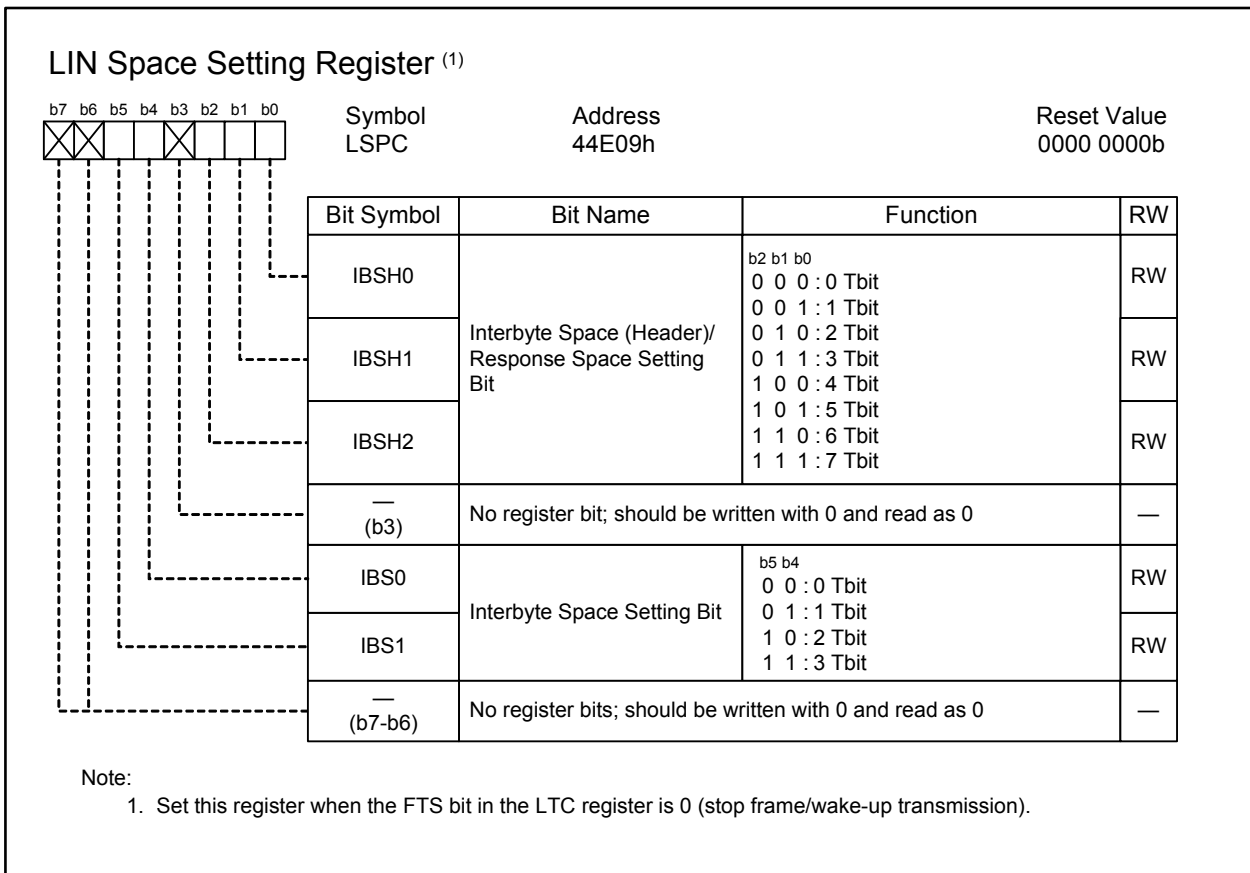


Figure 23.11 LSPC Register

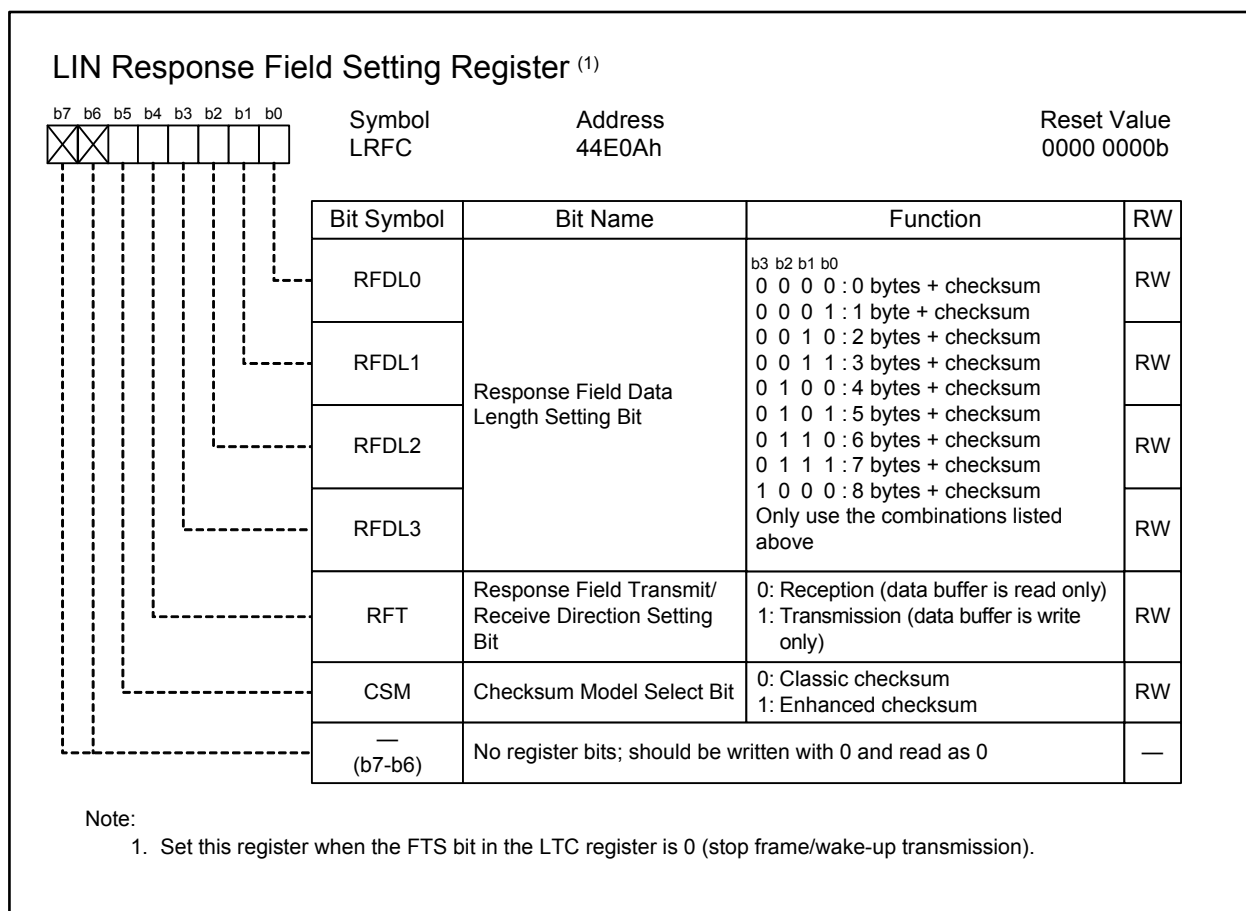


Figure 23.12 LRFC Register

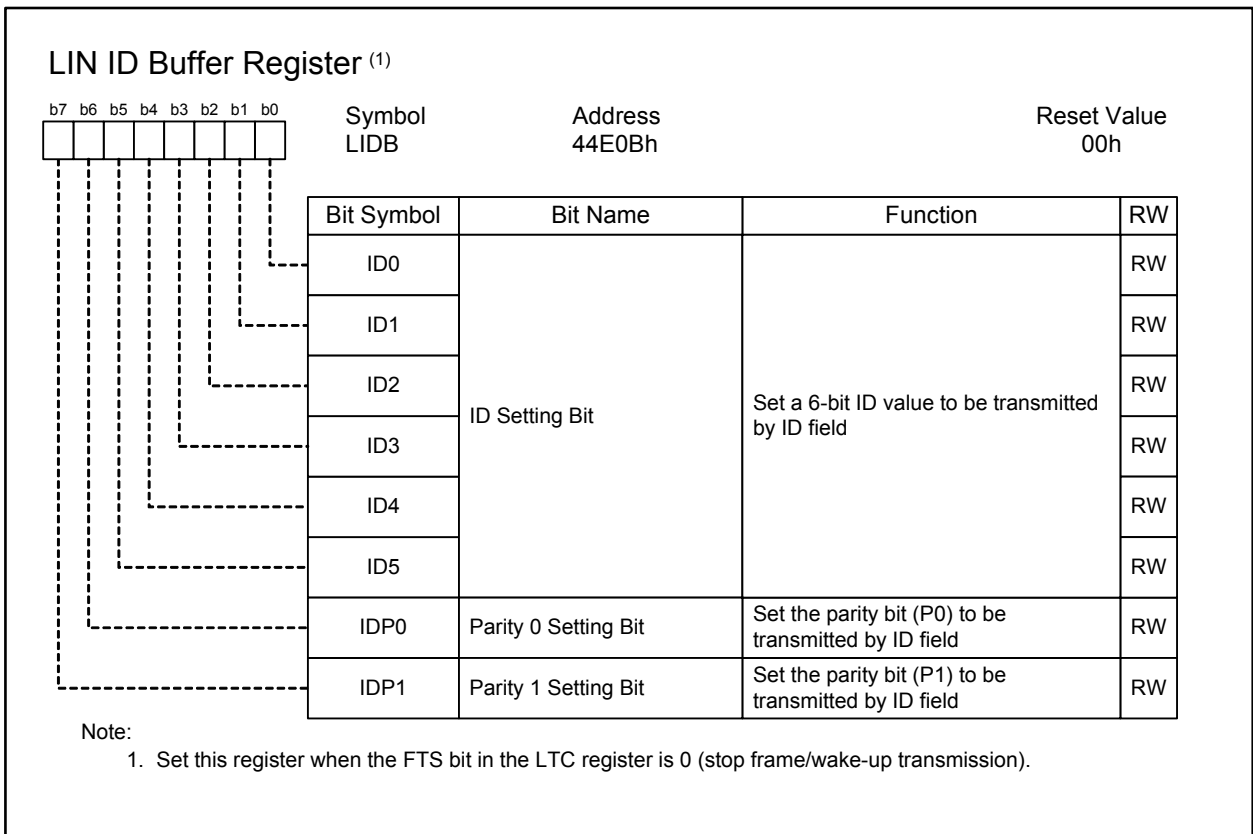


Figure 23.13 LIDB Register

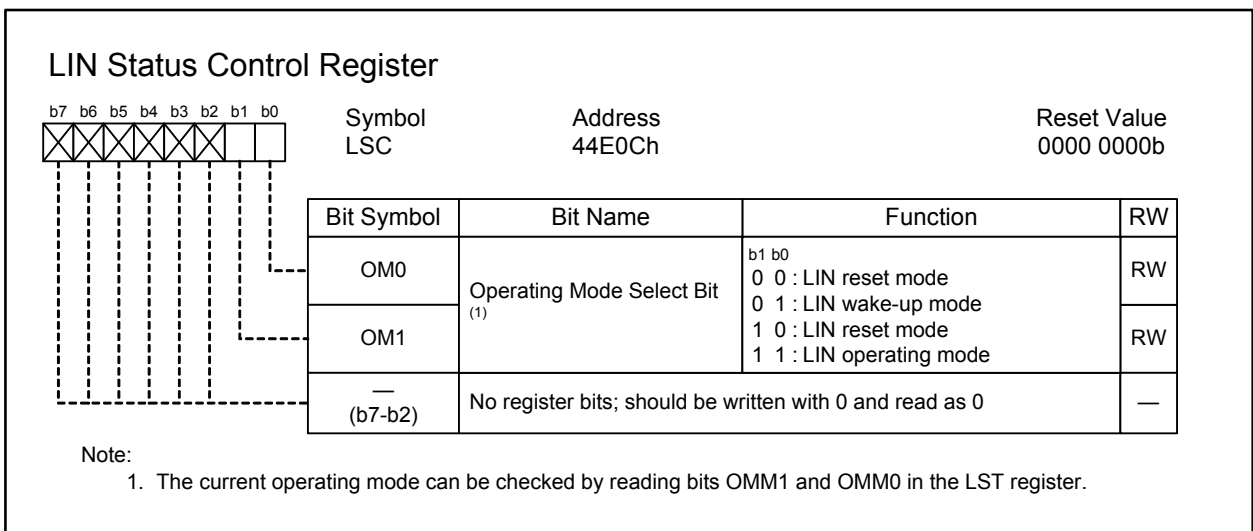


Figure 23.14 LSC Register



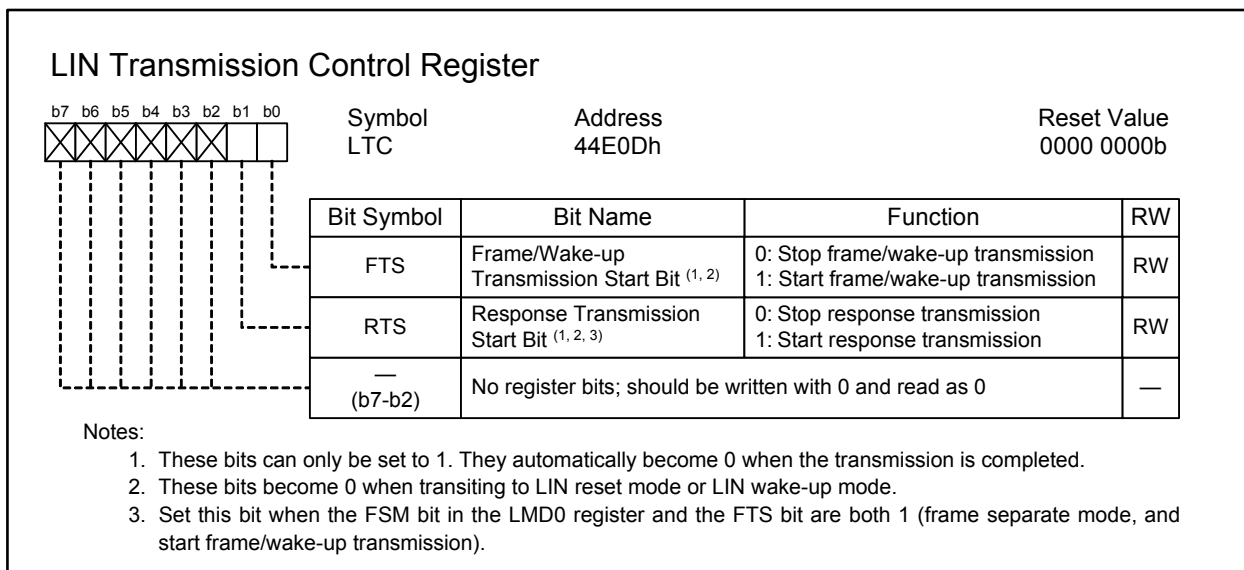


Figure 23.15 LTC Register

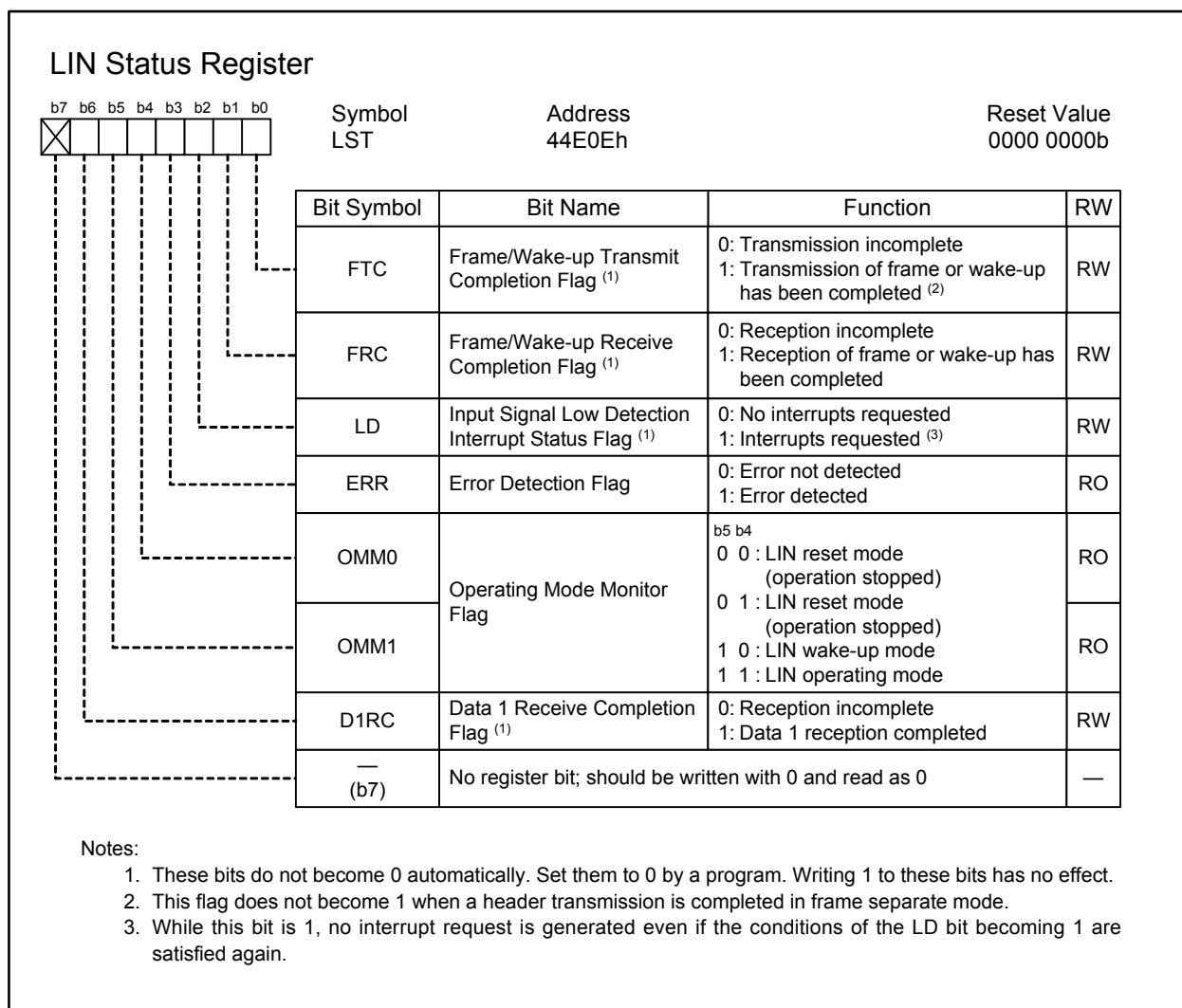


Figure 23.16 LST Register

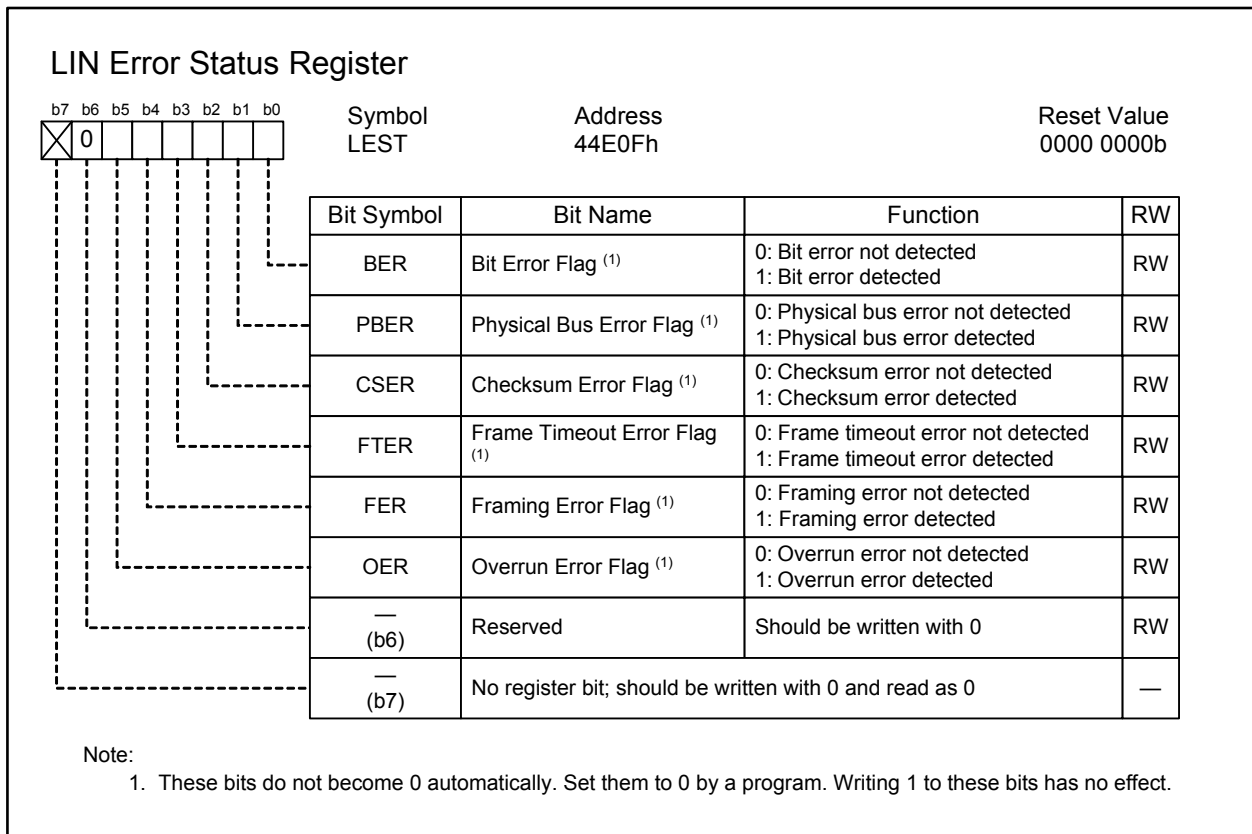


Figure 23.17 LEST Register

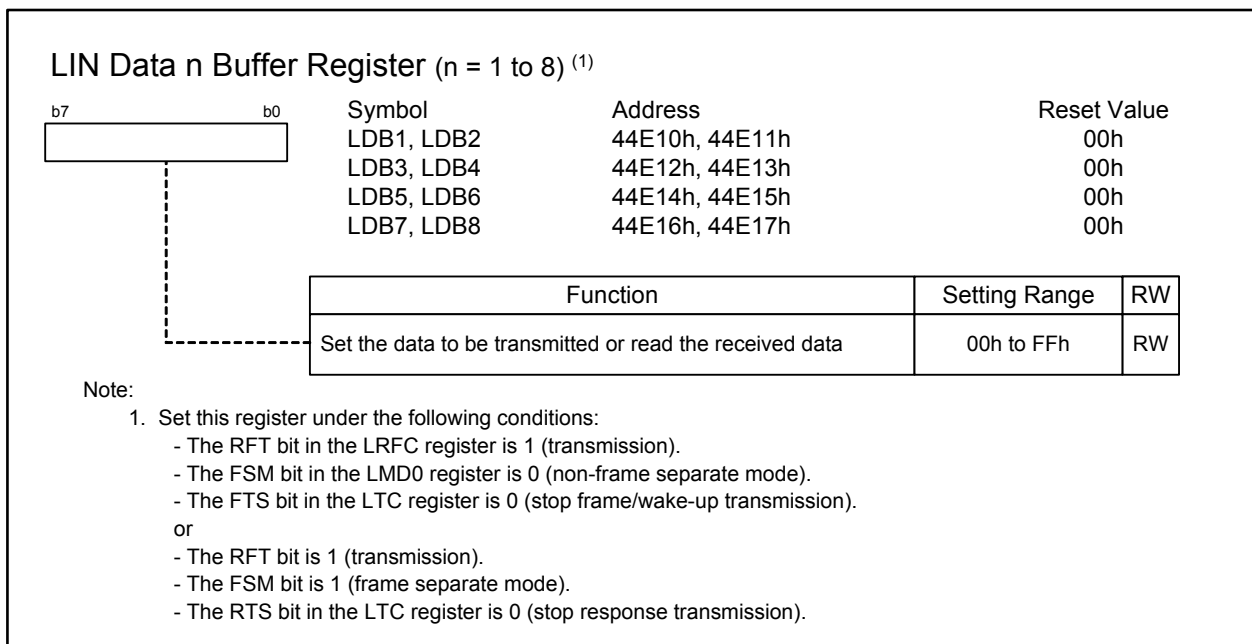


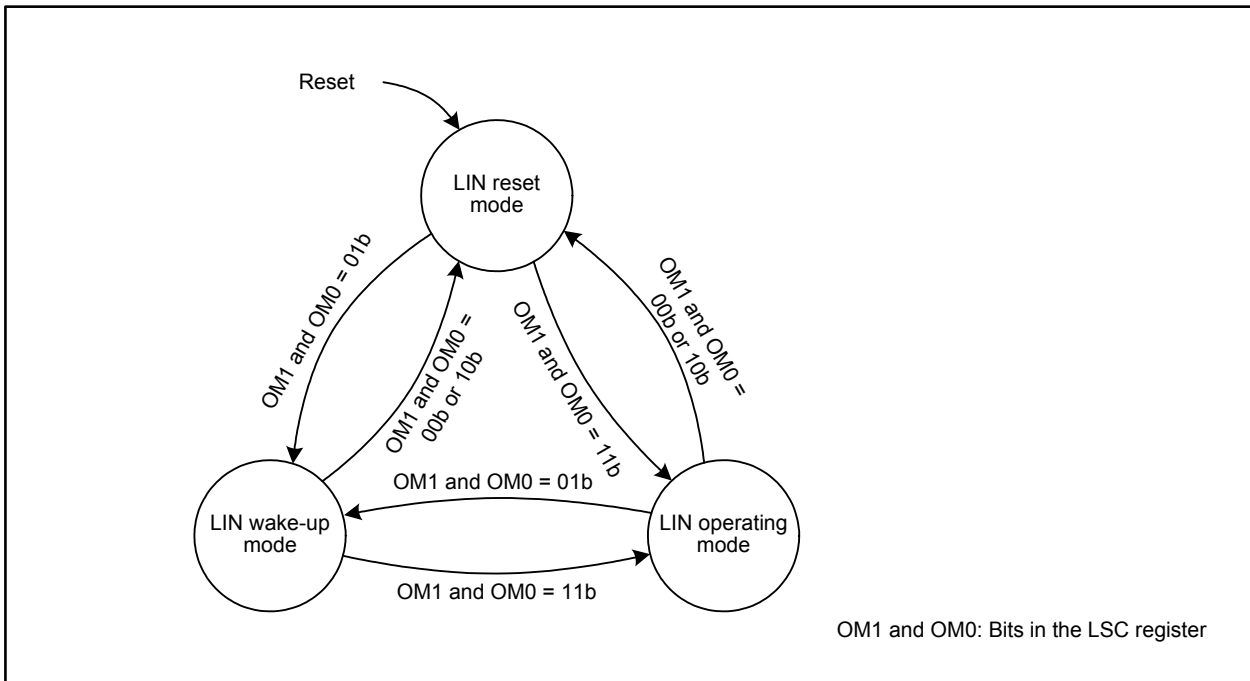
Figure 23.18 Registers LDB1 to LDB8

## 23.2 Operating Modes

The LIN modules have the following three operating modes:

- LIN reset mode
- LIN operating mode
- LIN wake-up mode

In LIN reset mode, power consumption is reduced since the clock is not provided to the LIN module. Figure 23.19 shows the transition processes between LIN operating modes and Table 23.2 lists functions available in each mode.



**Figure 23.19** LIN Operating Mode Transition

**Table 23.2** Functions Available in Each Operating Mode

LIN Reset Mode	LIN Operating Mode	LIN Wake-up Mode
Input signal low detection	Header transmission Response transmission Response reception Error detection Input signal low detection	Wake-up transmission Wake-up reception Error detection Input signal low detection

Read bits OMM1 and OMM0 in the LST register to verify that the LIN module has entered each operating mode.

Note that bits OM1 and OM0 and bits OMM1 and OMM0 are allocated differently in each operating mode.

### 23.2.1 LIN Reset Mode

When bits OM1 and OM0 in the LSC register are set to 00b or 10b, the LIN channel enters LIN reset mode, and bits OMM1 and OMM0 in the LST register become 00b or 01b. In this mode, all LIN functions are stopped as well as the clock supply to the LIN module (fLIN) of the channel.

### 23.2.2 LIN Operating Mode

When bits OM1 and OM0 in the LSC register are set to 11b, the LIN channel enters LIN operating mode, and bits OMM1 and OMM0 in the LST register become 11b.

### 23.2.3 LIN Wake-up Mode

When bits OM1 and OM0 in the LSC register are set to 01b, the LIN channel enters LIN wake-up mode, and bits OMM1 and OMM0 in the LST register become 10b.

## 23.3 Operational Overview

### 23.3.1 Header Transmission

Figure 23.20 shows the operation in header transmission and Table 23.3 lists the processing in header transmission.

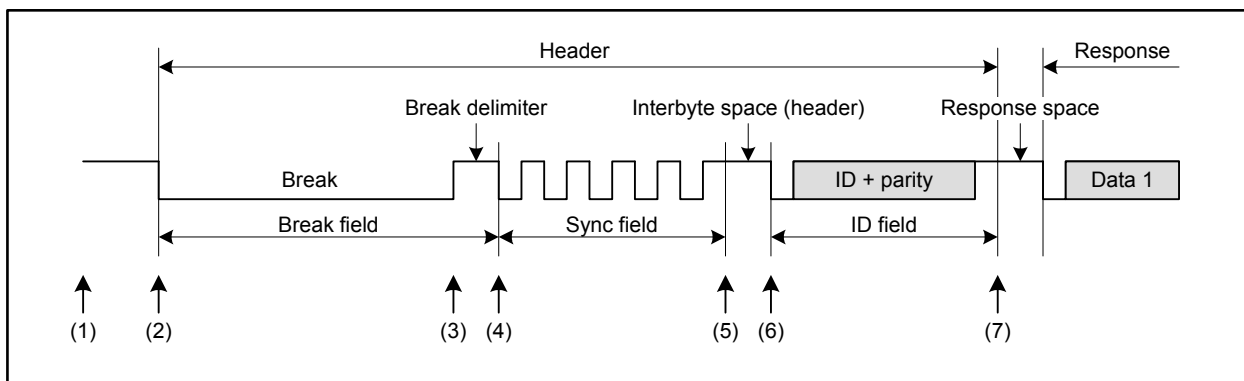


Figure 23.20 Operation in Header Transmission

Table 23.3 Processing in Header Transmission

	Software Processing	LIN Module Processing
(1)	<ul style="list-style-type: none"> <li>• Set a baud rate</li> <li>• Set the following bits in the LMD0 register:               <ul style="list-style-type: none"> <li>The FTCIE bit to 1 (frame/wake-up transmit completion interrupt enabled);</li> <li>The FRCIE bit to 1 (frame/wake-up receive completion interrupt enabled);</li> <li>The ERRIE bit to 1 (error detection interrupt enabled)</li> </ul> </li> <li>• Change the operating mode</li> <li>• Set the following bits in the LBRK register:               <ul style="list-style-type: none"> <li>Bits BLT3 to BLT0 for break low (13 to 28 Tbit);</li> <li>Bits BDT1 and BDT0 for break delimiter (1 to 4 Tbit)</li> </ul> </li> <li>• Set the following bits in the LSPC register:               <ul style="list-style-type: none"> <li>Bits IBSH2 to IBSH0 for interbyte space (Header)/response space (0 to 7 Tbit);</li> <li>Bits IBS1 and IBS0 for interbyte space (0 to 3 Tbit)</li> </ul> </li> <li>• Set the following bits in the LIDB register:               <ul style="list-style-type: none"> <li>Bits ID5 to ID0 for ID value;</li> <li>Bits IDP1 and IDP0 for parity value</li> </ul> </li> <li>• Set the following bits in the LRFC register:               <ul style="list-style-type: none"> <li>Bits RFDL3 to FRDL0 for data length;</li> <li>The RFT bit for response transmit/receive direction</li> <li>The CSM bit for checksum model</li> </ul> </li> <li>• Set the transmit data</li> </ul>	Wait for frame/wake-up transmission start by software (idle)
(2)	<ul style="list-style-type: none"> <li>• Set the FTS bit in the LTC register to 1 (frame/wake-up transmission started)</li> </ul>	Transmit break
(3)	<ul style="list-style-type: none"> <li>• Wait for an interrupt request to be generated</li> </ul>	Transmit break delimiter
(4)		Transmit Sync field (55h)
(5)		Transmit interbyte space (Header)
(6)		Transmit ID field
(7)		Transmit response space

### 23.3.2 Response Transmission

Figure 23.21 shows the operation in response transmission and Table 23.4 lists the processing in response transmission.

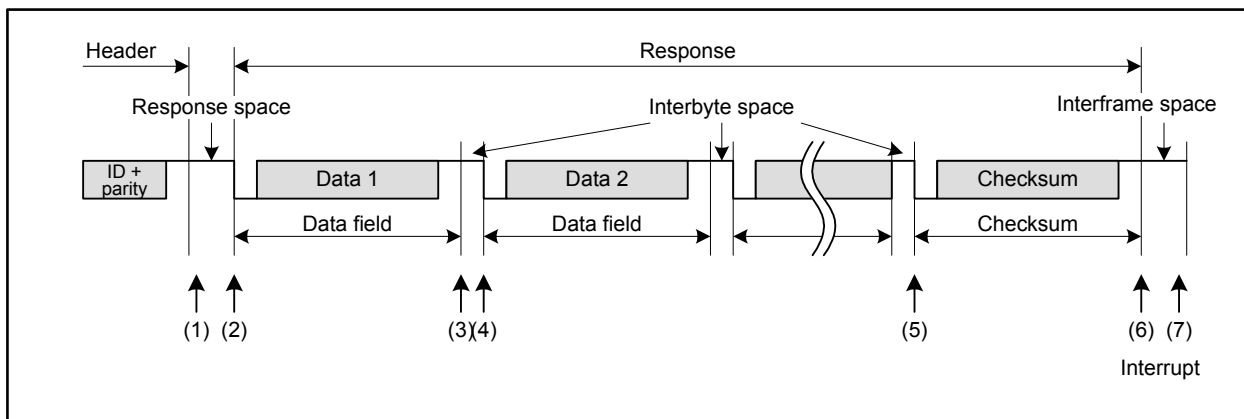


Figure 23.21 Operation in Response Transmission

Table 23.4 Processing in Response Transmission

	Software Processing	LIN Module Processing
(1)	<ul style="list-style-type: none"> <li>In frame separate mode Set the RTS bit in the LTC register to 1 (response transmission started)</li> <li>In non-frame separate mode Wait for an interrupt request to be generated</li> </ul>	<ul style="list-style-type: none"> <li>In frame separate mode Transmit response space while waiting for response transmission enabled</li> <li>In non-frame separate mode Go to (2) if the response space transmission is completed</li> </ul>
(2)	Wait for an interrupt request to be generated	Transmit Data 1
(3)		Transmit interbyte space
(4)		Transmit Data 2, then the next interbyte space Transmit Data 3, then the next interbyte space (Repeat this process for the data length specified in bits RFDL3 to RFDL0 in the LRFC register. Go to (6) if an error occurs.)
(5)		Transmit checksum
(6)		<ul style="list-style-type: none"> <li>Set frame/wake-up transmit completion flag or error flag</li> <li>Set the following bits in the LTC register: The FTS bit to 0 (frame/wake-up transmission stopped); The RTS bit to 0 (response transmission stopped)</li> </ul>
(7)	<ul style="list-style-type: none"> <li>Processing after communication Check the LST register and set flags to 0</li> </ul>	Idle

### 23.3.3 Response Reception

Figure 23.22 shows the operation in response reception and Table 23.5 lists the processing in response reception.

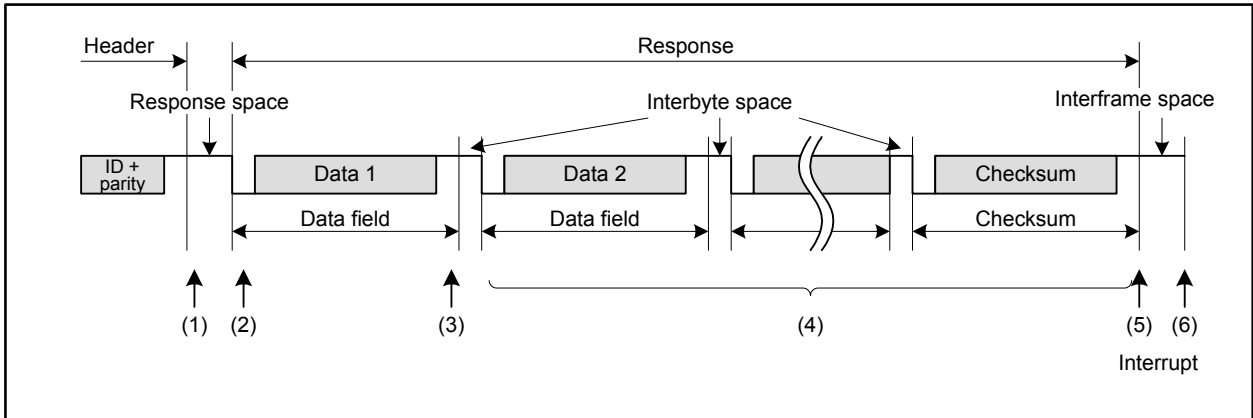


Figure 23.22 Operation in Response Reception

Table 23.5 Processing in Response Reception

	Software Processing	LIN Module Processing
(1)	Wait for an interrupt request to be generated (without processing)	Wait for a start bit to be detected
(2)	Wait for an interrupt request to be generated	Receive Data 1 due to start bit detection
(3)		• Set Data 1 receive completion flag
(4)		Receive Data 2 due to start bit detection Receive Data 3 due to start bit detection (Repeat this process for the data length specified in bits RFDL3 to RFDL0 in the LRFC register. Abort the reception and go to (5) if an error occurs. Checksum judgement is not performed in this case.) : : Receive checksum due to start bit detection
(5)		• Judge the checksum • Set frame receive completion flag or error flag • Set the FTS bit in the LTC register to 0 (frame/wake-up transmission stopped)
(6)	Processing after communication Check the LST register and set flags to 0	Idle



## 23.4 Baud Rate Generator

The LIN module system clock (fLIN) is generated by peripheral clock divided by the baud rate generator. The fLIN divided by 8 or 16 is used as the bit rate. The reciprocal of the bit rate is called bit time and expressed as "Tbit".

When the BTD bit in the LMD0 register is 0 (1 Tbit is generated by 8 fLIN), the FLNS bit in registers LBRG and LBRP0 should be set so that fLn is 153600 Hz (19200 bps × 8). Therefore,

$$f_{Ln} = 19200 \text{ bps} \times 8 \text{ [Hz]}$$

$$f_{Ln2} = 9600 \text{ bps} \times 8 \text{ [Hz]}$$

$$f_{Ln8} = 2400 \text{ bps} \times 8 \text{ [Hz]}$$

Then they are divided by 8 in the bit timing generator, which results in 19200 bps, 9600 bps, and 2400 bps, respectively. The baud rate of 10417 bps is generated by the bit setting in the LBRP1 register.

When the BTD bit is 1 (1 Tbit is generated by 16 fLIN), the FLNS bit in registers LBRG and LBRP0 should be set so fLn is 307200 Hz (= 19200 bps × 16).

Figure 23.23 shows the block diagram of baud rate generation.

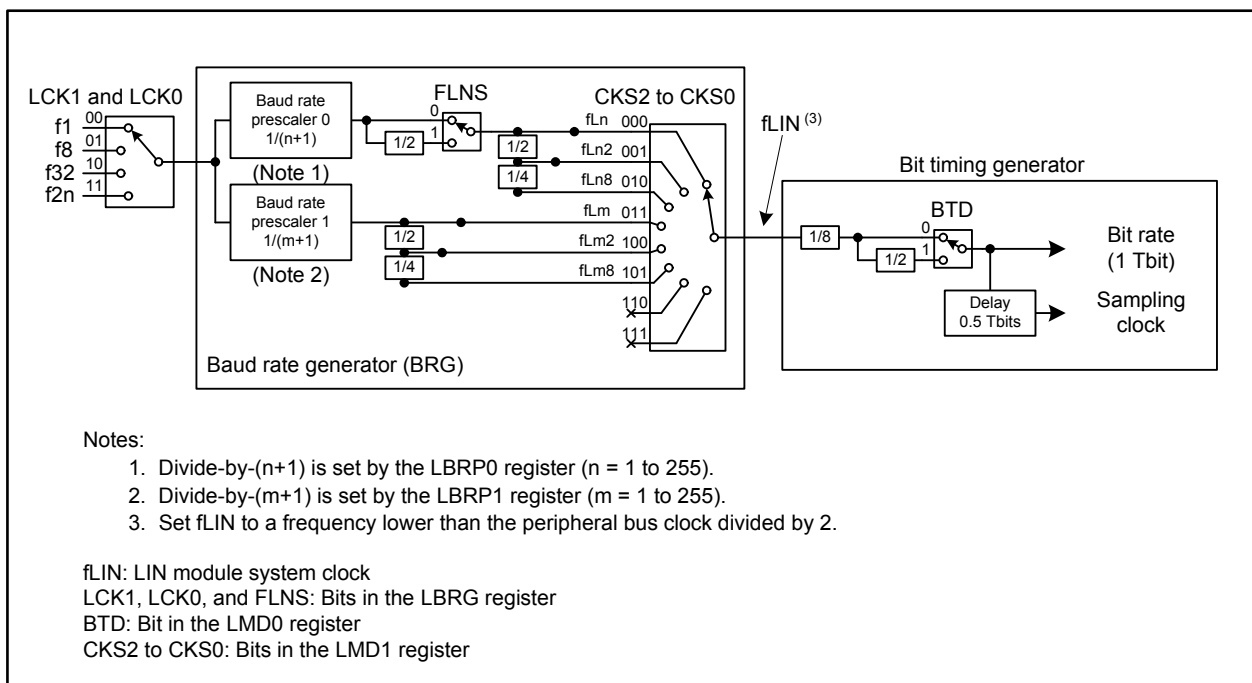


Figure 23.23 Baud Rate Generation Block Diagram

Tables 23.6 and 23.7 list the baud rates (19200, 9600, 2400, and 10417 bps) generated by the peripheral clock frequency, and their errors.

**Table 23.6 Baud Rate Generation Example (for 19200, 9600, and 2400 bps)**

Peripheral Clock	Divide-by- (n + 1)	BRP0 Output	Bit Times	fLn [19200 bps] (unit: bps)	fL2n [9600 bps] (unit: bps)	fL8n [2400 bps] (unit: bps)	Error (unit: %)
32 MHz	104	No division/ divide-by-2	16 fLIN/ 8 fLIN	19230.77	9615.38	2403.85	+0.16
30 MHz	98	No division/ divide-by-2	16 fLIN/ 8 fLIN	19132.65	9566.33	2391.58	-0.35
24 MHz	78	No division/ divide-by-2	16 fLIN/ 8 fLIN	19230.77	9615.38	2403.85	+0.16
20 MHz	65	No division/ divide-by-2	16 fLIN/ 8 fLIN	19230.77	9615.38	2403.85	+0.16
16 MHz	52	No division/ divide-by-2	16 fLIN/ 8 fLIN	19230.77	9615.38	2403.85	+0.16
12 MHz	39	No division/ divide-by-2	16 fLIN/ 8 fLIN	19230.77	9615.38	2403.85	+0.16
10 MHz	65	No division <sup>(1)</sup>	8 fLIN	19230.77	9615.38	2403.85	+0.16
8 MHz	26	No division/ divide-by-2	16 fLIN/ 8 fLIN	19230.77	9615.38	2403.85	+0.16
4 MHz	13	No division/ divide-by-2	16 fLIN/ 8 fLIN	19230.77	9615.38	2403.85	+0.16
2 MHz	13	No division <sup>(1)</sup>	8 fLIN	19230.77	9615.38	2403.85	+0.16

Note:

1. An error does not fulfill the LIN protocol specification ( $\pm 0.5\%$ ) if the BRP0 clock is divided by 2.

**Table 23.7 Baud Rate Generation Example for 10417 bps**

Peripheral Clock	Divide-by- (m + 1)	Bit Times	fLIN	Baud Rate [10417 bps] (unit: bps)	Error (unit:%)
32 MHz	48 / 96	8 fLIN / 16 fLIN	fLm8 / fLm2	10416.67	-0.003
30 MHz	45 / 90	8 fLIN / 16 fLIN	fLm8 / fLm2	10416.67	-0.003
24 MHz	36 / 72	8 fLIN / 16 fLIN	fLm8 / fLm2	10416.67	-0.003
20 MHz	30 / 60	8 fLIN / 16 fLIN	fLm8 / fLm2	10416.67	-0.003
16 MHz	24 / 48	8 fLIN / 16 fLIN	fLm8 / fLm2	10416.67	-0.003
12 MHz	18 / 36	8 fLIN / 16 fLIN	fLm8 / fLm2	10416.67	-0.003
10 MHz	15 / 30	8 fLIN / 16 fLIN	fLm8 / fLm2	10416.67	-0.003
8 MHz	12 / 24	8 fLIN / 16 fLIN	fLm8 / fLm2	10416.67	-0.003
4 MHz	6 / 12	8 fLIN / 16 fLIN	fLm8 / fLm2	10416.67	-0.003
2 MHz	3 / 6	8 fLIN / 16 fLIN	fLm8 / fLm2	10416.67	-0.003

## 23.5 Data Transmission and Reception

### 23.5.1 Data Transmission

The LIN module transmits 1-bit data per Tbit.

The transmitted data returns to the input pin for data reception via the LIN transceiver. Then the received data is compared to the transmitted data and the result is stored in the BER bit in the LEST register (refer to 23.10 "Error Status" for details). The timing to compare the received data is at the fifth clock when the BTD bit in the LMD0 register is 0 (1 Tbit is generated by 8 fLIN), and at the 13th clock when the BTD bit is 1 (1 Tbit is generated by 16 fLIN).

Figure 23.24 shows data transmission timing.

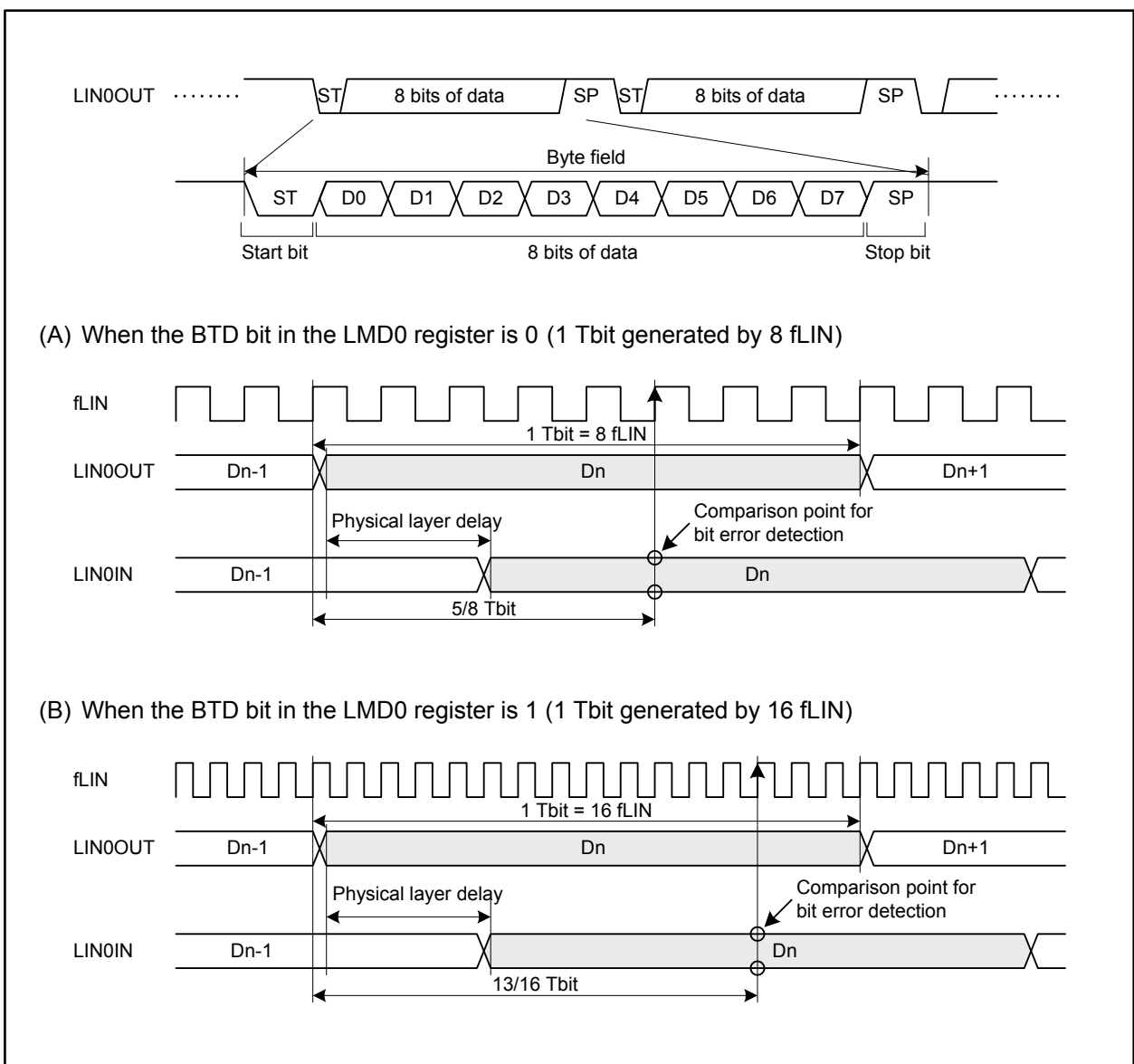


Figure 23.24 Data Transmission Timing

### 23.5.2 Data Reception

Data reception in the LIN module requires an internal signal generated from the input signal at the LIN0IN pin by a double-latch clocking scheme.

The byte field of this internal signal, called the synchronized LIN0IN, is synchronized with fLIN at the falling edge of the start bit. The synchronized LIN0IN signal is sampled again 0.5 Tbit after a falling edge is detected, and if the signal is low, it is verified as the start bit. It is not verified as the start bit if the LIN0IN signal has been low after a reset is released or high is detected on sampling.

Once the start bit is detected, data bits are sampled every Tbit.

Figure 23.25 shows the data reception timing.

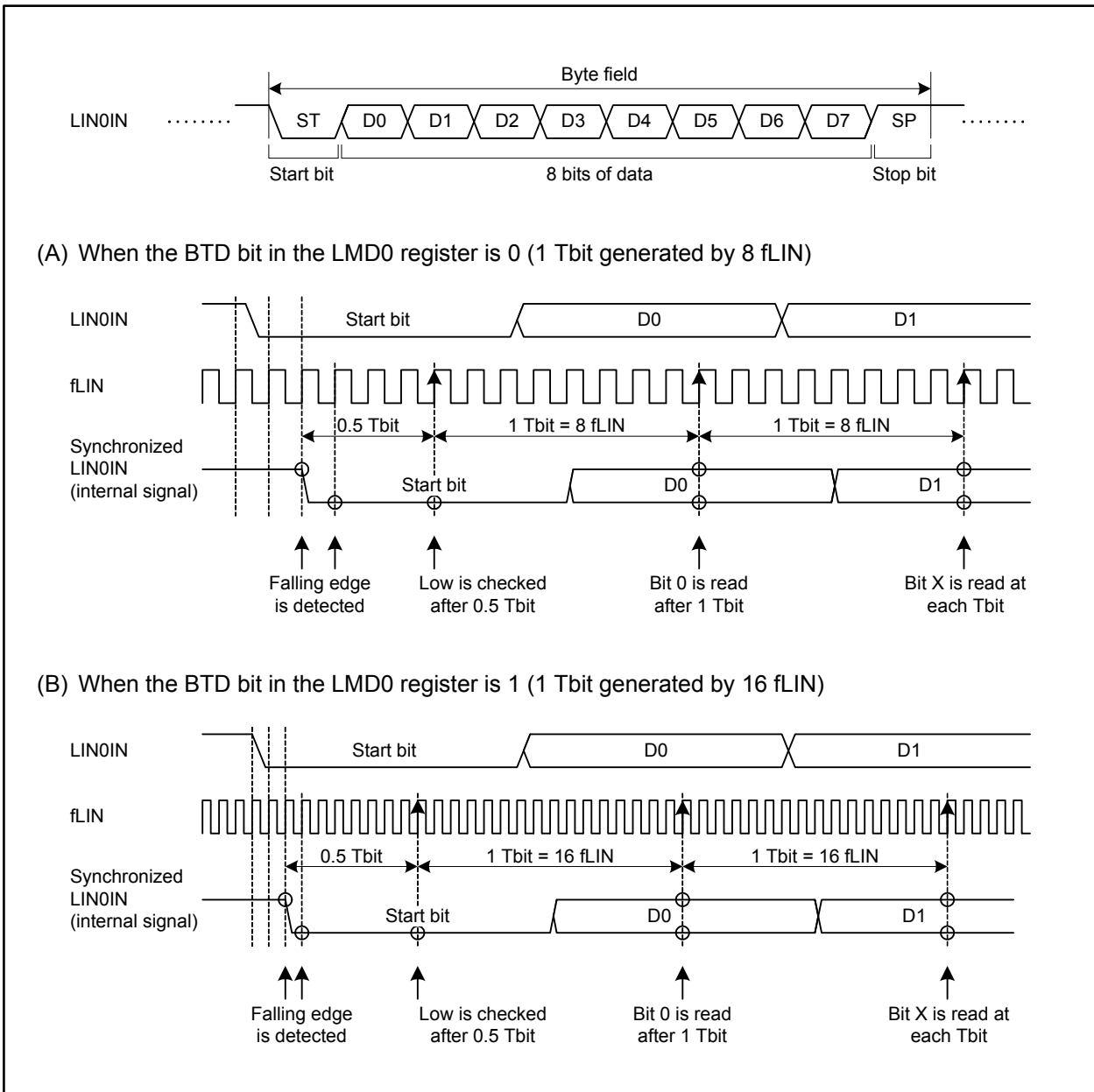


Figure 23.25 Data Reception Timing

## 23.6 Buffer Processing of Transmit/Receive Data

This section describes buffer processing when the LIN module is continuously transmitting and receiving data.

### 23.6.1 Transmission of LIN Frame

In 8-byte transmission, the values stored in registers LDB1 to LDB8 are transmitted in order to Data 1 to Data 8 of the LIN frame.

In 4-byte transmission, the values stored in registers LDB1 to LDB4 are transmitted to Data 1 to Data 4 of the LIN frame, and the values stored in registers LDB5 to LDB8 are not transmitted.

Figure 23.26 shows the processing of LIN transmission and buffer.

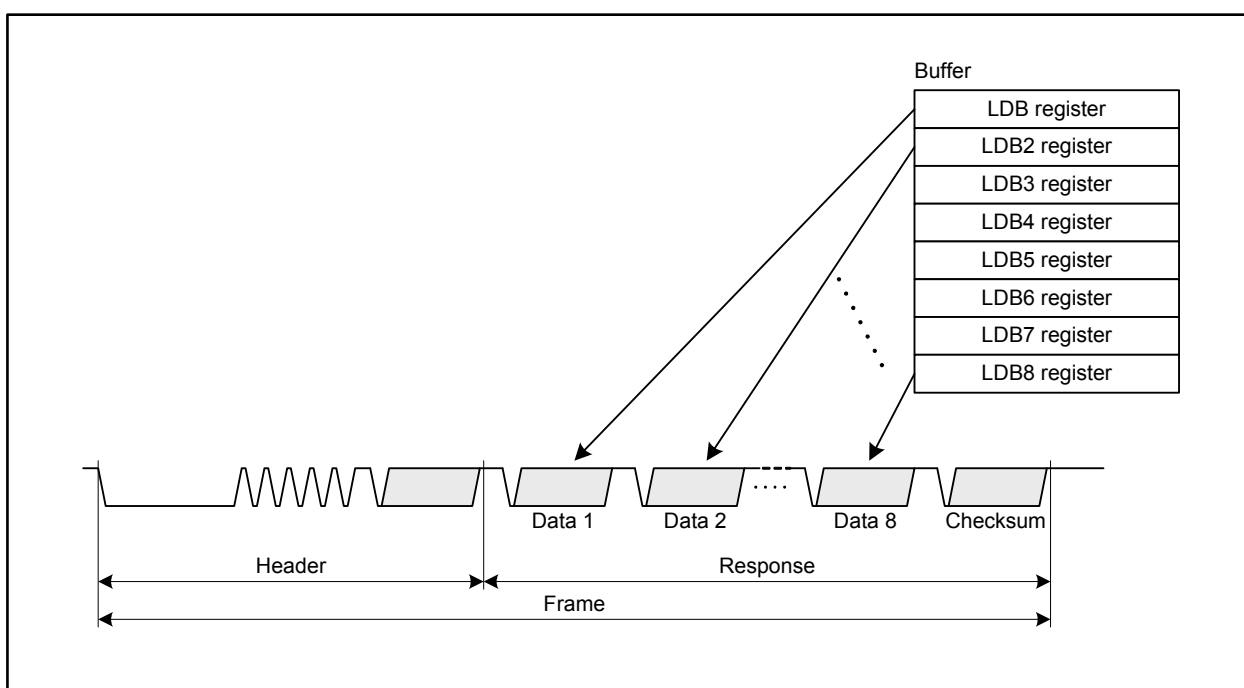


Figure 23.26 Processing of LIN Transmission and Buffer

### 23.6.2 Reception of LIN Frame

In 8-byte reception, the values of Data 1 to Data 8 of the LIN frame are stored in registers LDB1 to LDB8 in order at every completion of stop bit reception. In 4-byte reception, the values of Data 1 to Data 4 of the LIN frame are stored in registers LDB1 to LDB4 respectively, and no bits are stored in registers LDB5 to LDB8.

Figure 23.27 shows the processing of LIN reception and buffer.

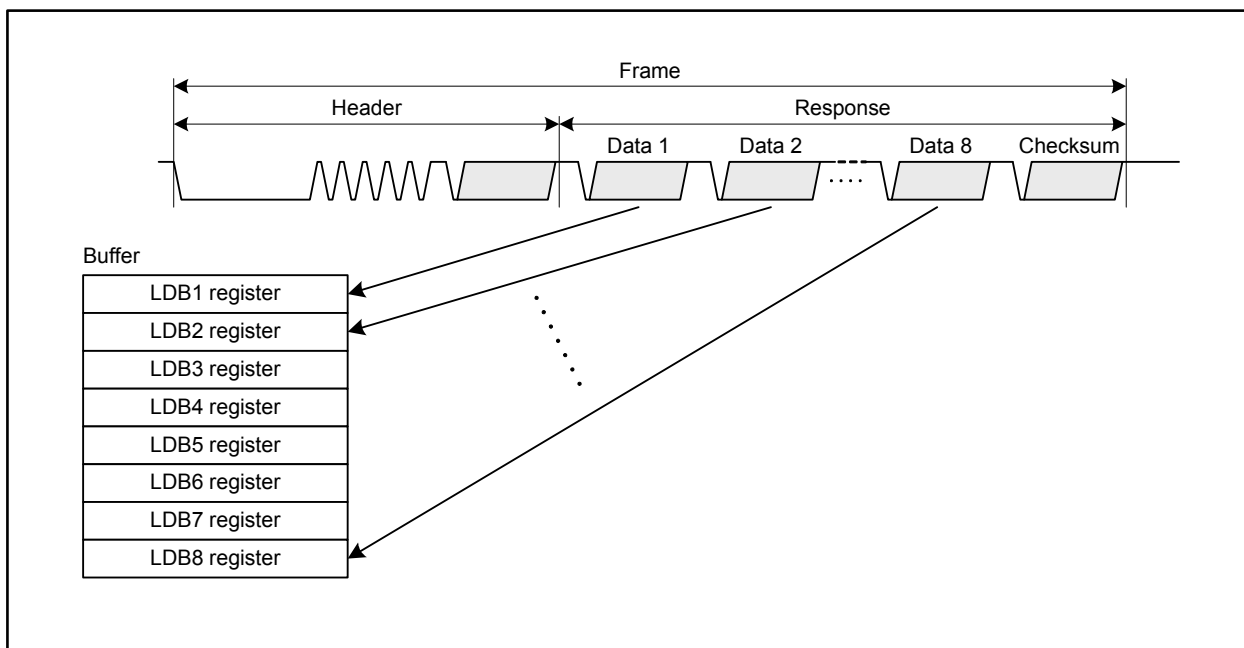


Figure 23.27 Processing of LIN Reception and Buffer

## 23.7 Wake-up Transmission and Reception

Wake-up transmission and reception can be used in LIN wake-up mode.

### 23.7.1 Wake-up Transmission

#### 23.7.1.1 Operation of Wake-up Transmission

In LIN wake-up mode, when the FTS bit in the LTC register is set to 1 (frame/wake-up transmission started), the wake-up signal is output at the output pin. Low time pulse width of the wake-up signal is set by bits WUTL3 to WUTL0 in the LWUP register.

When low of wake-up is output without any bit error detected, the FTC bit in the LST register becomes 1 (frame/wake-up transmission completed). When the FTCIE bit in the LMD0 register is 1 (frame/wake-up transmit completion interrupt enabled), an interrupt request is generated.

When a bit error is detected, the operation is aborted and the BER bit in the LEST register becomes 1 (bit error is detected).

Figure 23.28 shows the wake-up transmission timing.

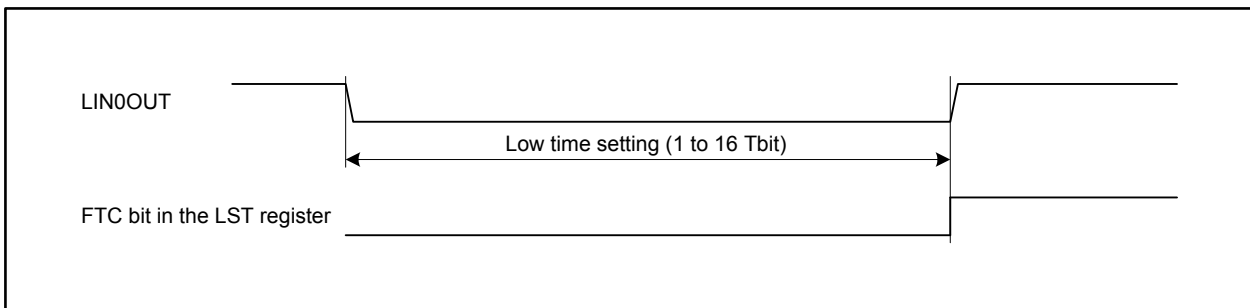


Figure 23.28 Wake-up Transmission Timing

#### 23.7.1.2 Wake-up Collision

When the master and slave transmit wake-up signals simultaneously, a signal collision occurs on the LIN bus. However, no collision of the wake-up signals is detected in the LIN module.

Even if the LIN bus is fixed low, it is not recognized as a wake-up signal: the input signal low time count does not function unless a recessive (high) is detected once after the wake-up signals are transmitted.

## 23.7.2 Wake-up Reception

### 23.7.2.1 Wake-up Reception Operation

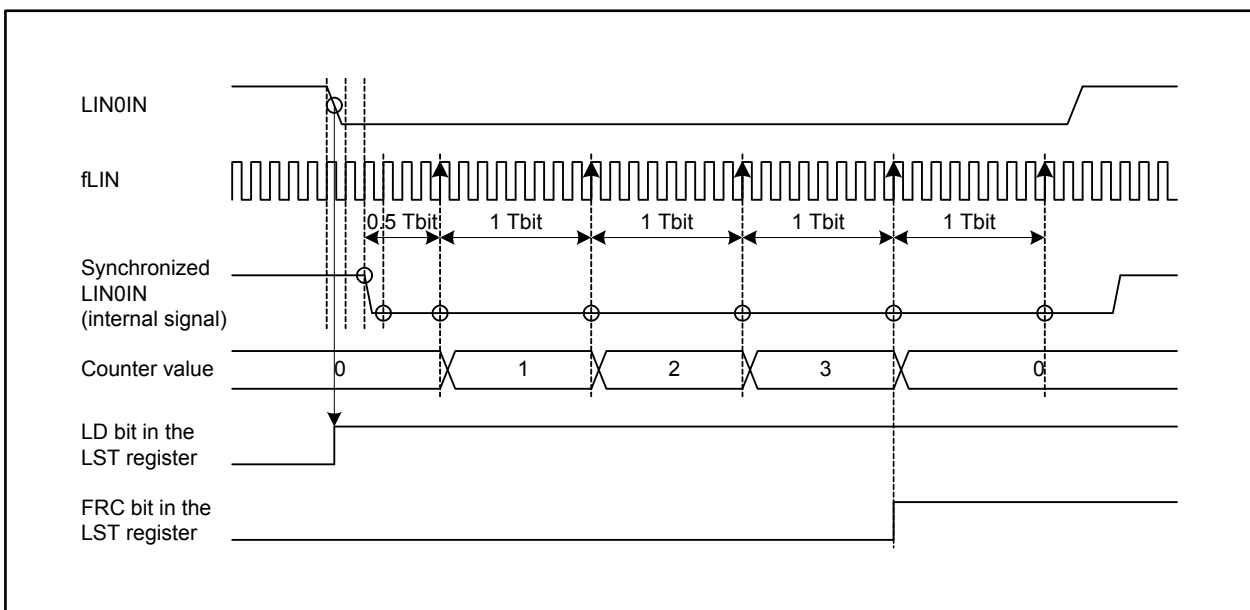
A wake-up is detected by either of the following functions: input signal low time count or input signal low detection. The input signal low time count is available in LIN wake-up mode and the input signal low detection is available in all LIN operating modes.

The input signal low time count measures the low time of an input signal at the LIN0IN pin by counting at the same sampling timing as that of data reception. When the counted low time reaches the value set with bits WURL3 to WURL0 in the LWUP register, the FRC bit in the LST register becomes 1 (frame/wake-up reception completed). When the FRCIE bit in the LMD0 register is 1 (frame/wake-up receive completion interrupt enabled), an interrupt request is generated.

On the other hand, the input signal low detection is to asynchronously detect the falling edge of the input signal at the LIN0IN pin. While the LDE bit in the LMD0 register is 1 (input signal low detection enabled), the LD bit in the LST register becomes 1 (input signal low detected) when the falling edge is detected, and an interrupt request is generated.

Figure 23.29 shows an example of the wake-up detection.

When the bit rate is 19200 bps (1 Tbit = 52  $\mu$ s) and the pulse width is 3.5 Tbit (bits WURL3 to WURL0 is 0011b), the low time pulse over 182  $\mu$ s (= 52  $\mu$ s  $\times$  3.5) is recognized as the wake-up signal.



**Figure 23.29 Wake-up Detection**

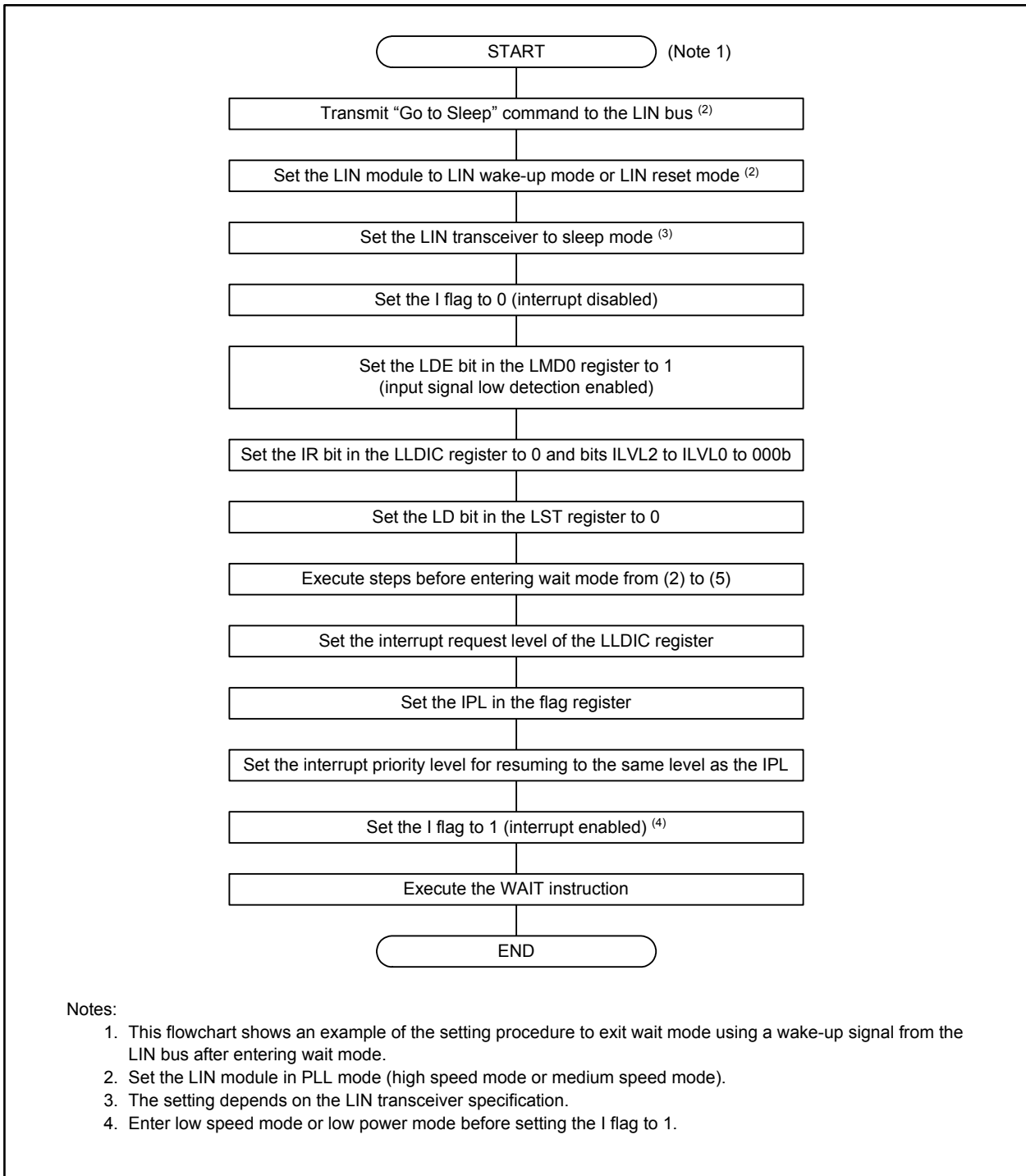
The input signal low time count is not available during the wake-up transmission processing, however, the input signal low detection is. If the FTS bit in the LTC register is set to 1 (start frame/wake-up transmission) while a wake-up reception is being counted, the reception processing is aborted to perform the wake-up transmission.



### 23.8 Low Power Mode Control Using Input Signal Low Detection

The input signal low detection can be used as the interrupt to exit wait mode or stop mode.

Figure 23.30 shows a setting example before entering wait mode. Refer to 7.7.2 “Wait Mode” and 7.7.3 “Stop Mode” for details on entering wait or stop mode.



**Figure 23.30 Example of Setting Before Transition to Wait Mode**

## 23.9 Operational Status

The LIN module detects six types of operational statuses.

Statuses that can generate an interrupt request are: frame/wake-up transmit completion, frame/wake-up receive completion, input signal low detection, and error detection.

Table 23.8 lists the types of operational statuses.

**Table 23.8 Operational Statuses**

Operational Status	Detecting Condition	Detectable Operating Modes	Corresponding Bits in the LST Register
Frame/wake-up transmit completion	When a response frame transmission has been successfully completed in the bit setting as follows: RFT bit in the LRFC register is 1 (transmission selected)	LIN operating mode	FTC
	When a wake-up transmission has been successfully completed	LIN wake-up mode	
Frame/wake-up receive completion	When a response frame reception has been successfully completed in the bit setting as follows: RFT bit in the LRFC register is 0 (reception selected)	LIN operating mode	FRC
	When the low time of the input signal reaches the setting value of bits WURL3 to WURL0 in the LWUP register	LIN wake-up mode	
Input signal low detection	When the falling edge of the input signal at the LIN0IN pin is detected with the setting of the LDE bit in the LMD0 register to 1 (input signal low detection enabled), or when setting the LDE bit to 1 while the LIN0IN pin is low	All LIN modes	LD
Error detection	When any bit from 0 to 5 in the LEST register becomes 1 (error detected)	LIN operating mode LIN wake-up mode	ERR
Operating mode	When LIN module has entered the operating mode which was set by bits OM1 and OM0 in the LSC register	All LIN modes	OMM1 and OMM0
Data 1 receive completion	When the reception of the first byte of a response frame has been completed in the bit setting as follows: RFT bit in the LRFC register is 0 (reception selected) <sup>(1)</sup>	LIN operating mode	D1RC

Note:

1. This status is detected except when bits RFDL3 to RFDL0 in the LRFC register are 0000b (0 bytes + checksum).

## 23.10 Error Status

### 23.10.1 Error Status Types

The LIN module detects six types of error statuses. These error statuses can be checked by bits 0 to 5 in the LEST register.

Table 23.9 lists the types of error statuses.

**Table 23.9 Error Statuses**

Error Status	Error Detecting Condition	Error Detectable Operating Mode	Communication Processing	Detection Enabled/ Disabled	Corresponding Bit in the LEST Register
Bit error	When the transmitted data does not match with that on the LIN bus monitored by the pin for reception	LIN operating mode LIN wake-up mode	Abort <sup>(1)</sup>	Selectable	BER
Physical bus error	When LIN bus has already become low just before the break field is transmitted or when it is detected that LIN bus is being fixed to high	LIN operating mode LIN wake-up mode	Abort	Selectable	PBER
Checksum error	When the checksum judgement of the response frame reception processing results in an error	LIN operating mode	—	Not selectable	CSER
Frame timeout error	When the transmit/receive operation is not completed within a specified period of time <sup>(2)</sup>	LIN operating mode	Abort	Selectable	FTER
Framing error	When the stop bit of each data byte is low in response frame reception processing	LIN operating mode	Abort	Selectable	FER
Overrun error	When the first data byte of the next response frame is received while the FRC bit in the LST register is 1 (frame reception successfully completed)	LIN operating mode	Abort	Selectable	OER

Notes:

- When a bit error is detected, the processing is aborted after the stop bit is transmitted. If it is detected in a non-data space, such as an interbyte space, the transmission processing is aborted after a check in that space is completed. During wake-up transmission, the wake-up transmission is aborted immediately after the error bit is transmitted.
- The period of time (as time out value) depends on the response field data length set by bits RFDL3 to RFDL0 in the LRFC register as follows:

$$\text{Time out value} = 50 + (\text{data bytes} + 1) \times 14 \quad [\text{Tbit}]$$

The above period of time will exceed the value of  $T_{\text{FRAME\_MAX}} \{ (10 \times \text{data bytes} + 44) + 1 \} \times 1.4$  shown in LIN Specification Package Revision 1.3.

### 23.10.2 LIN Error Detection Targets

Figure 23.31 shows the area which the LIN module monitors for error detection.

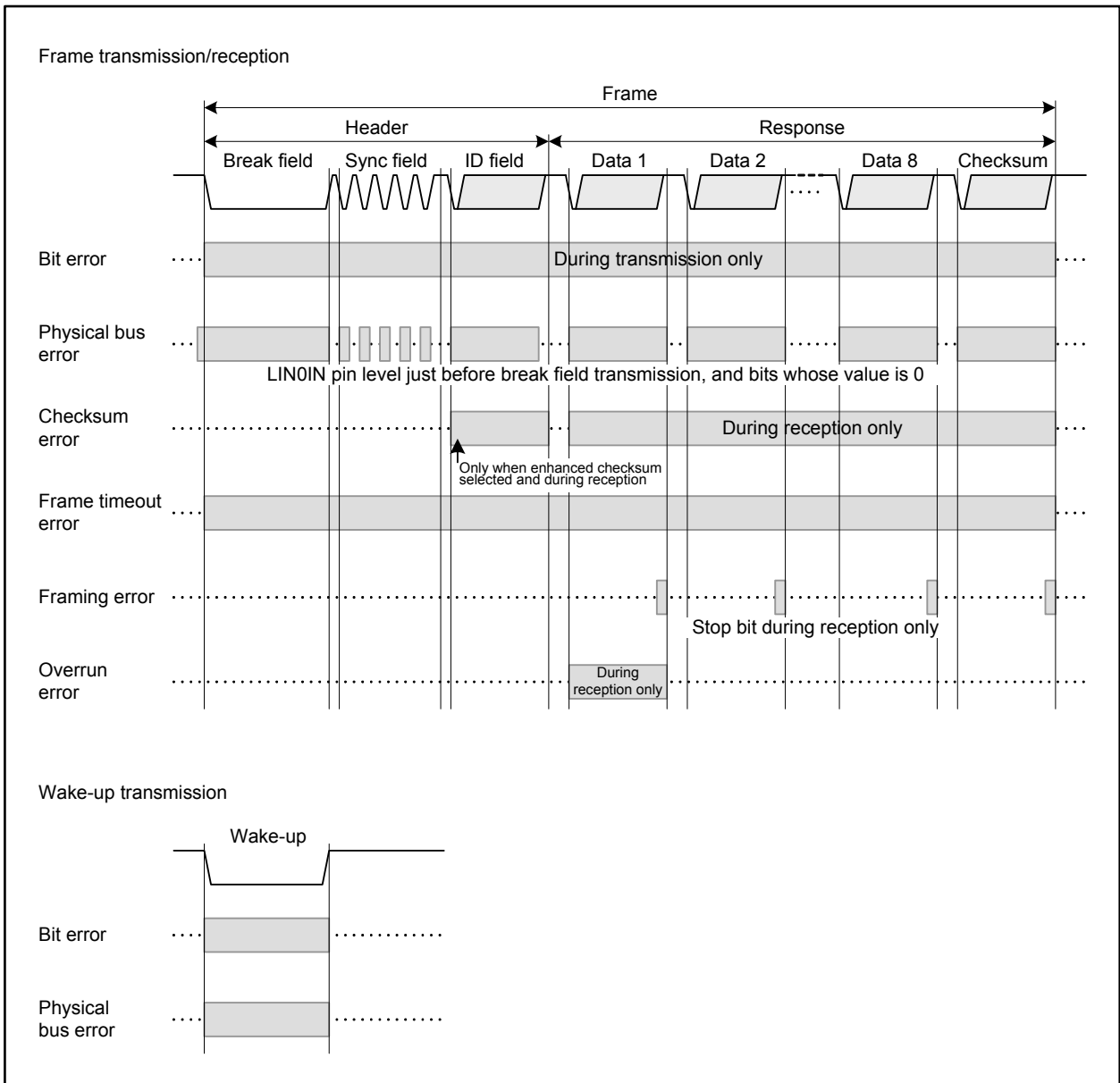


Figure 23.31 LIN Error Detection Targets

### 23.11 LIN Interrupts

The LIN module generates the LIN0 interrupt and LIN low detection interrupt.

The channel has four interrupt sources: frame/wake-up transmit completion, frame/wake-up receive completion, error detection, and input signal low detection.

Interrupt requests by the frame/wake-up transmit completion, frame/wake-up receive completion, and error detection interrupt sources are aggregated by logical OR as an interrupt request "LIN0 interrupt". The input signal low detection interrupt of each channel, on the other hand, is called as LIN low detection interrupt request.

The respective interrupt request is output when the corresponding flag in the LST register becomes 1 while the corresponding bit in the LMD0 register is 1 (interrupt enabled). No new interrupt request is generated by the other sources if any of them is 1 since multiple interrupt sources are aggregated.

Figures 23.32 and 23.33 show the block diagram of the LIN0 interrupt and the LIN low detection interrupt, respectively.

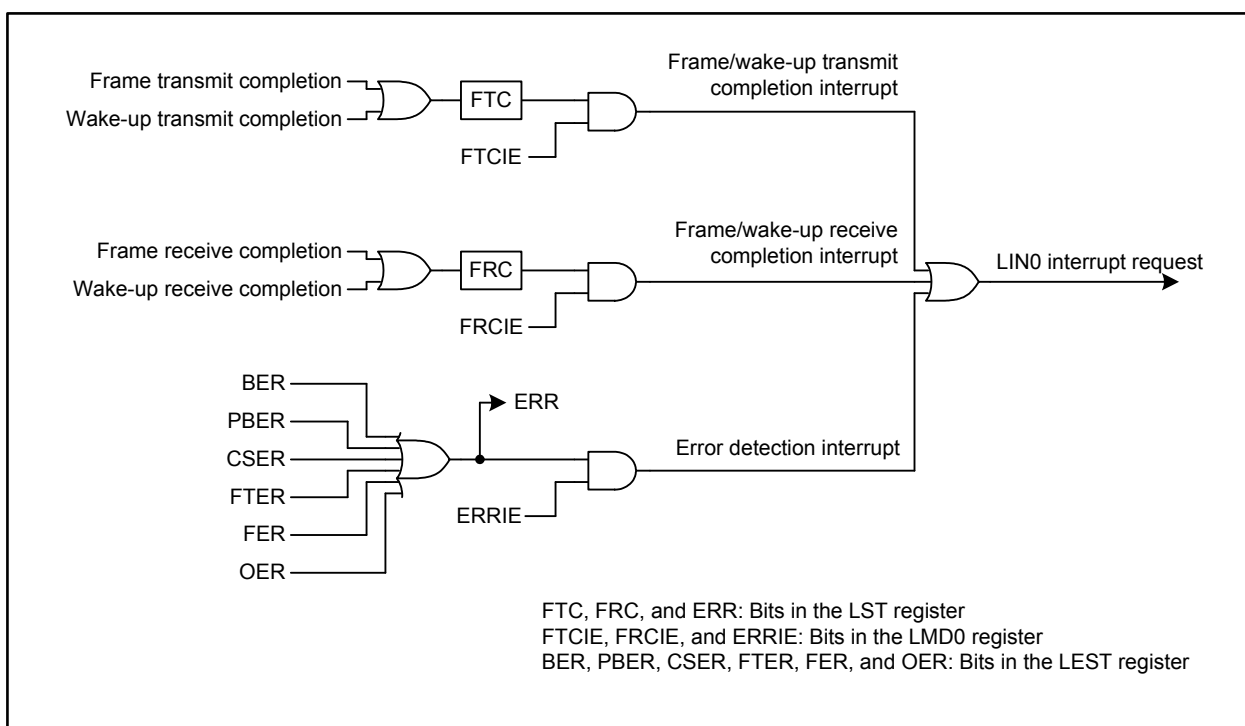


Figure 23.32 LIN0 Interrupt Block Diagram

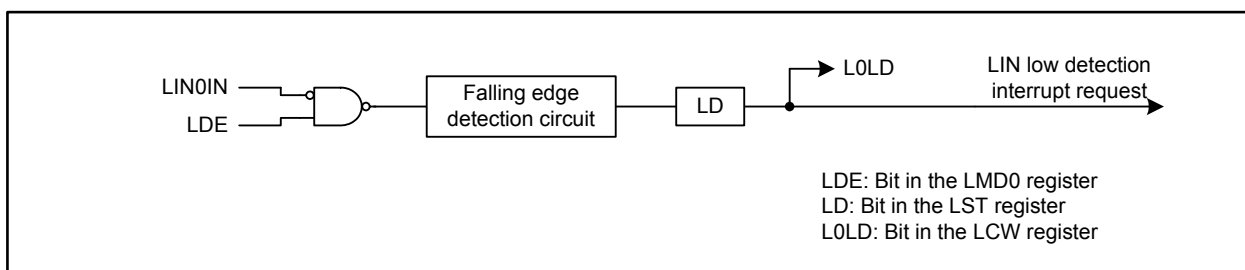


Figure 23.33 LIN Low Detection Interrupt Block Diagram

## 24. CAN Module

The R32C/161 Group implements two channels (CAN0 and CAN1) of the Controller Area Network (CAN) module that complies with the ISO 11898-1 standard. The CAN module transmits and receives both formats of messages, namely the standard identifier (11 bits) (identifier hereafter referred to as ID) and extended ID (29 bits).

Tables 24.1 and 24.2 list the CAN module specifications, and Figure 24.1 shows the CAN module block diagram.

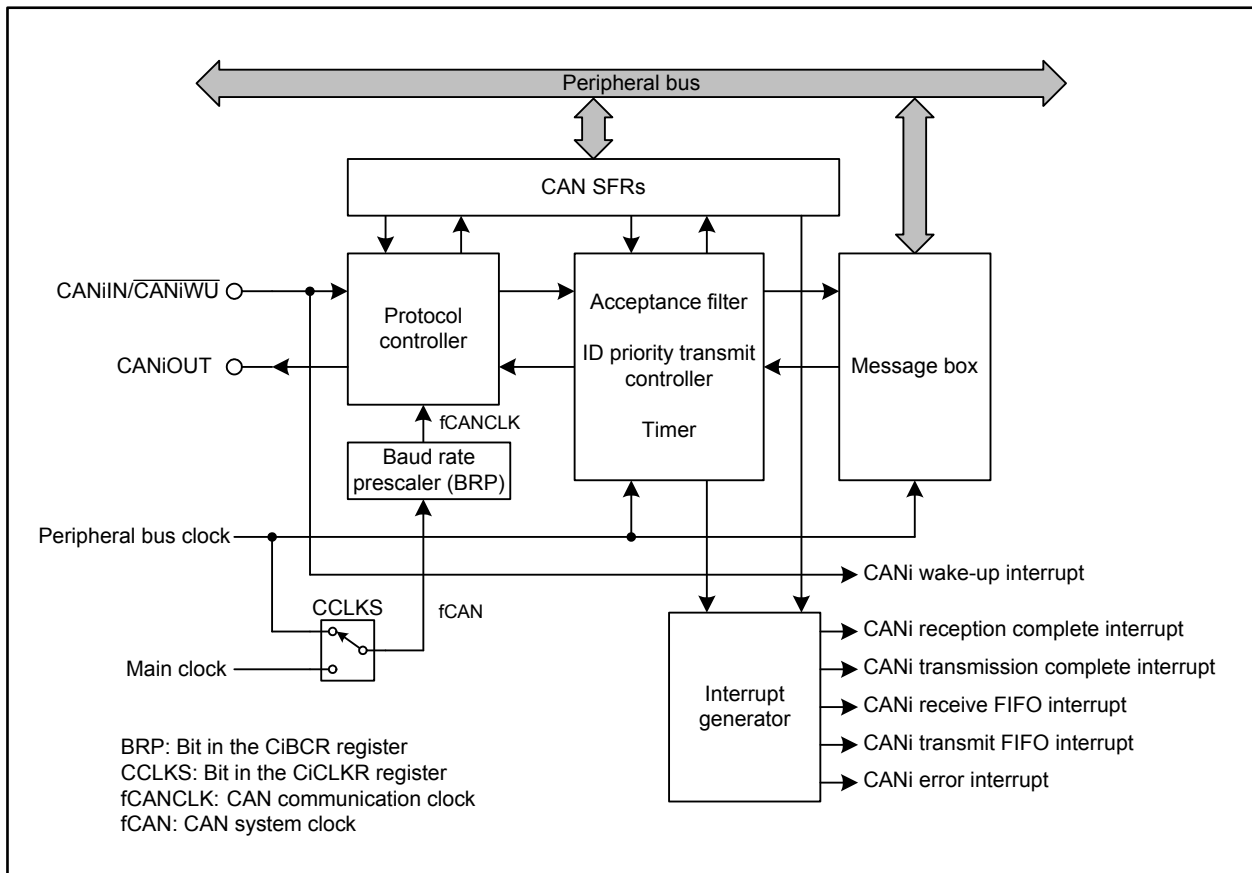
Connect the CAN bus transceiver externally.

**Table 24.1 CAN Module Specifications (1/2)**

Item	Specification
Protocol	ISO 11898-1 compliant
Bit rate	Maximum 1 Mbps
Message boxes	32 mailboxes: Two selectable mailbox modes: • Normal mailbox mode All 32 mailboxes can be individually configured for transmission or reception • FIFO mailbox mode: 24 mailboxes can be individually configured for transmission or reception. 4 of the remaining mailboxes can be configured for transmit FIFO and the other 4 mailboxes for receive FIFO
Reception	<ul style="list-style-type: none"> <li>• Data frames and remote frames can be received</li> <li>• Selectable receiving ID format (standard ID only, extended ID only, or both IDs)</li> <li>• Programmable one-shot reception function</li> <li>• Selectable overwrite mode (message overwritten) or overrun mode (message discarded)</li> <li>• The reception complete interrupt can be enabled or disabled for each mailbox</li> </ul>
Acceptance filtering	8 acceptance masks: 1 mask for every 4 mailboxes The mask can be enabled or disabled for each mailbox
Transmission	<ul style="list-style-type: none"> <li>• Data frames and remote frames can be transmitted</li> <li>• Selectable transmitting ID format (standard ID only, extended ID only, or both IDs)</li> <li>• Programmable one-shot transmission function</li> <li>• Selectable ID priority transmit mode or mailbox number priority transmit mode</li> <li>• Transmission request can be aborted (A completed abort operation can be confirmed with a flag)</li> <li>• The transmission complete interrupt can be enabled or disabled for each mailbox</li> </ul>
Mode transition for bus-off recovery	Mode transition for recovering from the bus-off state can be selected: <ul style="list-style-type: none"> <li>• ISO 11898-1 compliant</li> <li>• Automatic entry to CAN halt mode at bus-off entry</li> <li>• Automatic entry to CAN halt mode at bus-off end</li> <li>• Entry to CAN halt mode by a program</li> <li>• Transition to the error-active state by a program</li> </ul>

**Table 24.2 CAN Module Specifications (2/2)**

Item	Specification
Error status monitoring	<ul style="list-style-type: none"> <li>• CAN bus errors (stuff error, form error, ACK error, CRC error, bit error, and ACK delimiter error) can be monitored</li> <li>• Transition to error states can be detected (error-warning, error-passive, bus-off entry, and bus-off recovery)</li> <li>• The error counters can be read</li> </ul>
Time stamp function	<p>Time stamp function using a 16-bit counter</p> <p>The reference clock can be selected among 1, 2, 4, and 8 bit times</p>
Interrupt sources	<p>6 types:</p> <ul style="list-style-type: none"> <li>• Reception complete</li> <li>• Transmission complete</li> <li>• Receive FIFO</li> <li>• Transmit FIFO</li> <li>• Error</li> <li>• Wake-up</li> </ul>
CAN sleep mode	Current consumption can be reduced by stopping the CAN clock
Software support units	<p>3 software support units:</p> <ul style="list-style-type: none"> <li>• Acceptance filter support</li> <li>• Mailbox search support (receive mailbox search, transmit mailbox search, and message lost search)</li> <li>• Channel search support</li> </ul>
CAN clock source	Peripheral bus clock or main clock selectable
Test modes	<p>3 test modes available for user evaluation:</p> <ul style="list-style-type: none"> <li>• Listen only mode</li> <li>• Self test mode 0 (external loop back)</li> <li>• Self test mode 1 (internal loop back)</li> </ul>



**Figure 24.1 CAN Module Block Diagram (i = 0, 1)**

- CANiIN/CANiOUT (i = 0, 1): CAN I/O pins
- Protocol controller: Handles CAN protocol processing such as bus arbitration, bit timing at transmission and reception, stuffing, error handling, etc.
- Message box: Consists of 32 mailboxes which can be individually configured as either a transmit or receive mailbox. Each mailbox has its own ID, data length code, an 8-byte data field, and a time stamp.
- Acceptance filter: Filters received messages. Registers CiMKR0 to CiMKR7 are used for the filtering process.
- Timer: Used for the time stamp function. The timer value when storing a message into a mailbox is written as the time stamp value.
- Wake-up function: Generates a CANi wake-up interrupt request when a message is detected on the CAN bus.
- Interrupt generator: Generates the following five types of interrupts:
  - CANi reception complete interrupt
  - CANi transmission complete interrupt
  - CANi receive FIFO interrupt
  - CANi transmit FIFO interrupt
  - CANi error interrupt
- CAN SFRs: CAN-associated registers. Refer to 24.1 “CAN SFRs” for details.



## 24.1 CAN SFRs

CAN-associated registers are shown in Figures 24.2 to 24.11, 24.13, 24.14, 24.16 to 24.20, 24.22, and 24.24 to 24.30.

### 24.1.1 CANi Control Register (CiCTLR) (i = 0, 1)

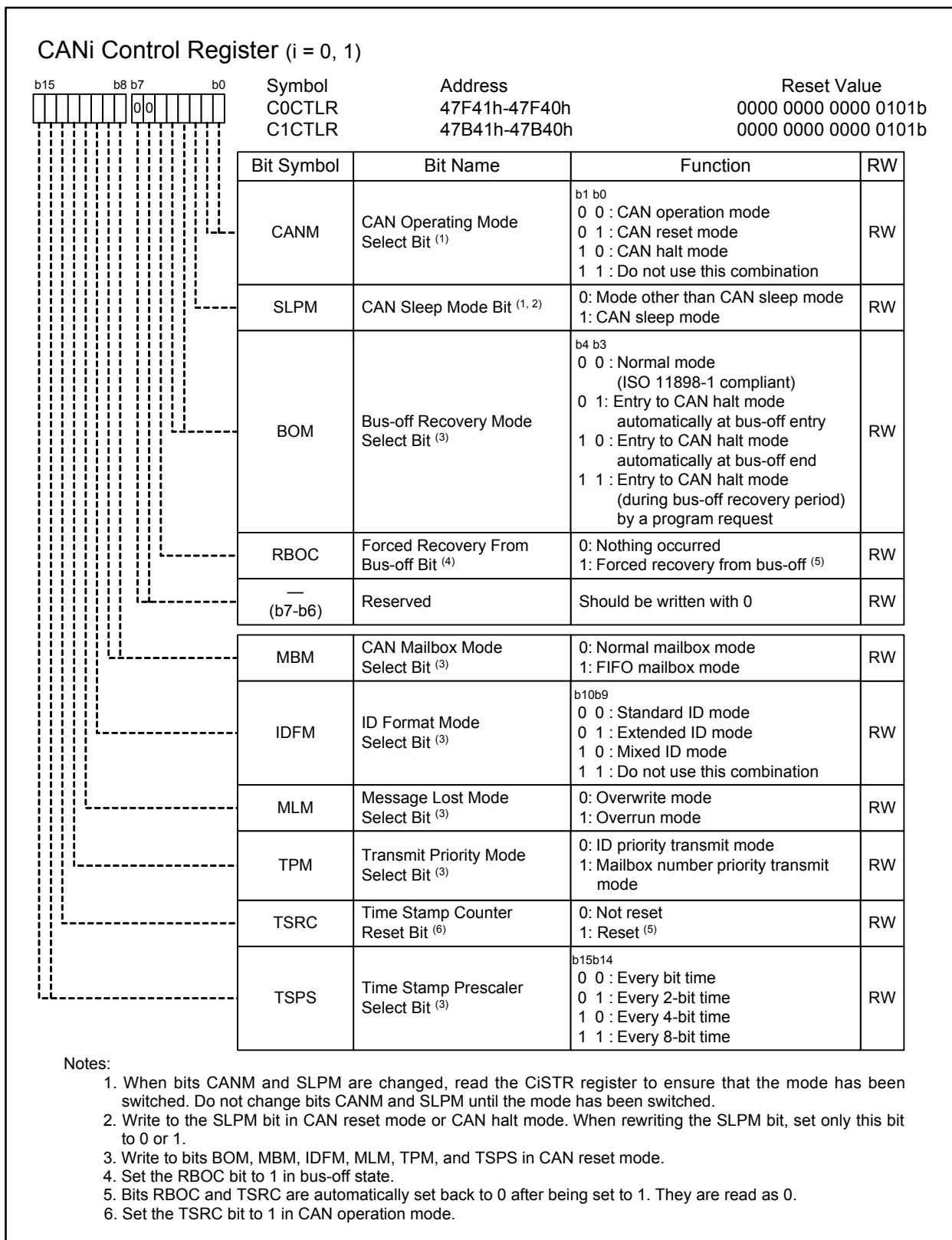


Figure 24.2 Registers C0CTLR and C1CTLR

### 24.1.1.1 CANM Bit

Set the CANM bit to select one of the following modes for the CAN module: CAN operation mode, CAN reset mode, or CAN halt mode. Refer to 24.2 “Operating Modes” for details.

Set the SLPM bit to 1 to select CAN sleep mode.

Do not set the CANM bit to 11b.

When the CAN module enters CAN halt mode according to the setting of the BOM bit, the CANM bit automatically becomes 10b.

### 24.1.1.2 SLPM Bit

When the SLPM bit is set to 1, the CAN module enters CAN sleep mode.

When this bit is set to 0, the CAN module exits CAN sleep mode.

Refer to 24.2 “Operating Modes” for details.

### 24.1.1.3 BOM Bit

Set the BOM bit to select bus-off recovery mode.

When the BOM bit is 00b, the recovery from bus-off is ISO 11898-1 compliant, i.e. the CAN module reenters CAN communication (error-active state) after detecting 11 consecutive recessive bits 128 times. A bus-off recovery interrupt request is generated when recovering from bus-off.

When the BOM bit is 01b, as soon as the CAN module enters the bus-off state, the CANM bit in the CiCTLR register becomes 10b (CAN halt mode) and the CAN module enters CAN halt mode ( $i = 0, 1$ ). No bus-off recovery interrupt request is generated when recovering from bus-off and registers CiTECR and CiRECR become 00h.

When the BOM bit is 10b, the CANM bit becomes 10b as soon as the CAN module enters the bus-off state. The CAN module enters CAN halt mode after recovering from the bus-off state, i.e. after detecting 11 consecutive recessive bits 128 times. A bus-off recovery interrupt request is generated when recovering from bus-off and registers CiTECR and CiRECR become 00h.

When the BOM bit is 11b, the CAN module enters CAN halt mode by setting the CANM bit to 10b while the CAN module is still in the bus-off state. No bus-off recovery interrupt request is generated when recovering from bus-off and registers CiTECR and CiRECR become 00h. However, if the CAN module recovers from bus-off after detecting 11 consecutive recessive bits 128 times before the CANM bit is set to 10b, a bus-off recovery interrupt request is generated.

If the CPU requests an entry to CAN reset mode at the same time as the CAN module attempts to enter CAN halt mode (at bus-off entry when the BOM bit is 01b, or at bus-off end when the BOM bit is 10b), then the CPU request to enter CAN reset mode has higher priority.

### 24.1.1.4 RBOC Bit

When the RBOC bit is set to 1 (forced recovery from bus-off) in the bus-off state, the CAN module forcibly recovers from the bus-off state. This bit automatically becomes 0. The error state changes from bus-off to error-active.

When the RBOC bit is set to 1, registers CiRECR and CiTECR become 00h and the BOST bit in the CiSTR register becomes 0 (CAN module is not in bus-off state). The other registers do not change. No bus-off recovery interrupt request is generated by recovering from the bus-off state.

Use the RBOC bit only when the BOM bit is 00b (normal mode).

### 24.1.1.5 MBM Bit

When the MBM bit is 0 (normal mailbox mode), mailboxes [0] to [31] are configured as transmit or receive mailboxes.

When this bit is 1 (FIFO mailbox mode), mailboxes [0] to [23] are configured as transmit or receive mailboxes, mailboxes [24] to [27] are configured as transmit FIFO, and mailboxes [28] to [31] are as receive FIFO.

Transmit data is written into mailbox [24] (mailbox [24] is a window mailbox for the transmit FIFO).

Receive data is read from mailbox [28] (mailbox [28] is a window mailbox for the receive FIFO).

Table 24.3 lists the mailbox configuration.

**Table 24.3 Mailbox Configuration**

Mailbox	MBM Bit is 0 (Normal mailbox mode)	MBM Bit is 1 <sup>(1)</sup> (FIFO mailbox mode)
Mailboxes [0] to [23]	Normal mailbox	Normal mailbox
Mailboxes [24] to [27]		Transmit FIFO
Mailboxes [28] to [31]		Receive FIFO

Note:

- When the MBM bit is set to 1, note the following:
  - Transmit FIFO is controlled by the CiTFCR register (i = 0, 1).  
The CiMCTLj register for mailboxes [24] to [27] is disabled (j = 0 to 31).  
Registers CiMCTL24 to CiMCTL27 cannot be used.
  - Receive FIFO is controlled by the CiRFCR register.  
The CiMCTLj register for mailboxes [28] to [31] is disabled.  
Registers CiMCTL28 to CiMCTL31 cannot be used.
  - Refer to the CiMIER register for the FIFO interrupts.
  - The corresponding bits in the CiMKIVLR register for mailboxes [24] to [31] are disabled. Set 0 to these bits.
  - Transmit/receive FIFOs can be used for both data frames and remote frames.

### 24.1.1.6 IDFM Bit

Set the IDFM bit to specify the ID format.

When this bit is 00b, all mailboxes (including FIFO mailboxes) handle standard IDs only.

When this bit is 01b, all mailboxes (including FIFO mailboxes) handle extended IDs only.

When this bit is 10b, all mailboxes (including FIFO mailboxes) handle both standard IDs and extended IDs. Standard IDs or extended IDs are specified by setting the IDE bit in the corresponding mailbox in normal mailbox mode. In FIFO mailbox mode, the IDE bit in the corresponding mailbox is used for mailboxes [0] to [23], the IDE bit in registers CiFIDCR0 and CiFIDCR1 is used for the receive FIFO, and the IDE bit in mailbox [24] is used for the transmit FIFO.

Do not set 11b to the IDFM bit.

### 24.1.1.7 MLM Bit

Set the MLM bit to specify the operation when a new message is captured in an unread mailbox. Overwrite mode or overrun mode can be selected. All mailboxes (including the receive FIFO) are set to either overwrite mode or overrun mode.

When the MLM bit is 0, all mailboxes are set to overwrite mode and the new message overwrites the old message.

When this bit is 1, all mailboxes are set to overrun mode and the new message is discarded.

### 24.1.1.8 TPM Bit

Set the TPM bit to specify the priority of modes when transmitting messages. ID priority transmit mode or mailbox number transmit mode can be selected. All mailboxes are set to either ID priority transmission or mailbox number priority transmission.

When the TPM bit is 0, ID priority transmit mode is selected and transmission priority complies with the CAN bus arbitration rule, as specified in the ISO 11898-1 standard. In ID priority transmit mode, mailboxes [0] to [31] (in normal mailbox mode), mailboxes [0] to [23] (in FIFO mailbox mode), and the transmit FIFO are compared with the IDs of mailboxes configured for transmission. If two or more mailbox IDs are the same, the mailbox with the smaller number has higher priority.

Only the next message to be transmitted from the transmit FIFO is included in the transmission arbitration. If a transmit FIFO message is being transmitted, the next pending message within the transmit FIFO is included in the transmission arbitration.

When the TPM bit is 1, mailbox number transmit mode is selected and the transmit mailbox with the smallest mailbox number has the highest priority. In FIFO mailbox mode, the transmit FIFO has lower priority than normal mailboxes (mailboxes [0] to [23]).

### 24.1.1.9 TSRC Bit

Set the TSRC bit to reset the time stamp counter.

When this bit is set to 1, the CiTSR register becomes 0000h (i = 0, 1). It automatically becomes 0.

### 24.1.1.10 TSPS Bit

Set the TSPS bit to select the prescaler for the time stamp.

The reference clock for the time stamp can be selected among 1, 2, 4, and 8 bit times.

## 24.1.2 CAN<sub>i</sub> Clock Select Register (CiCLKR) (i = 0, 1)

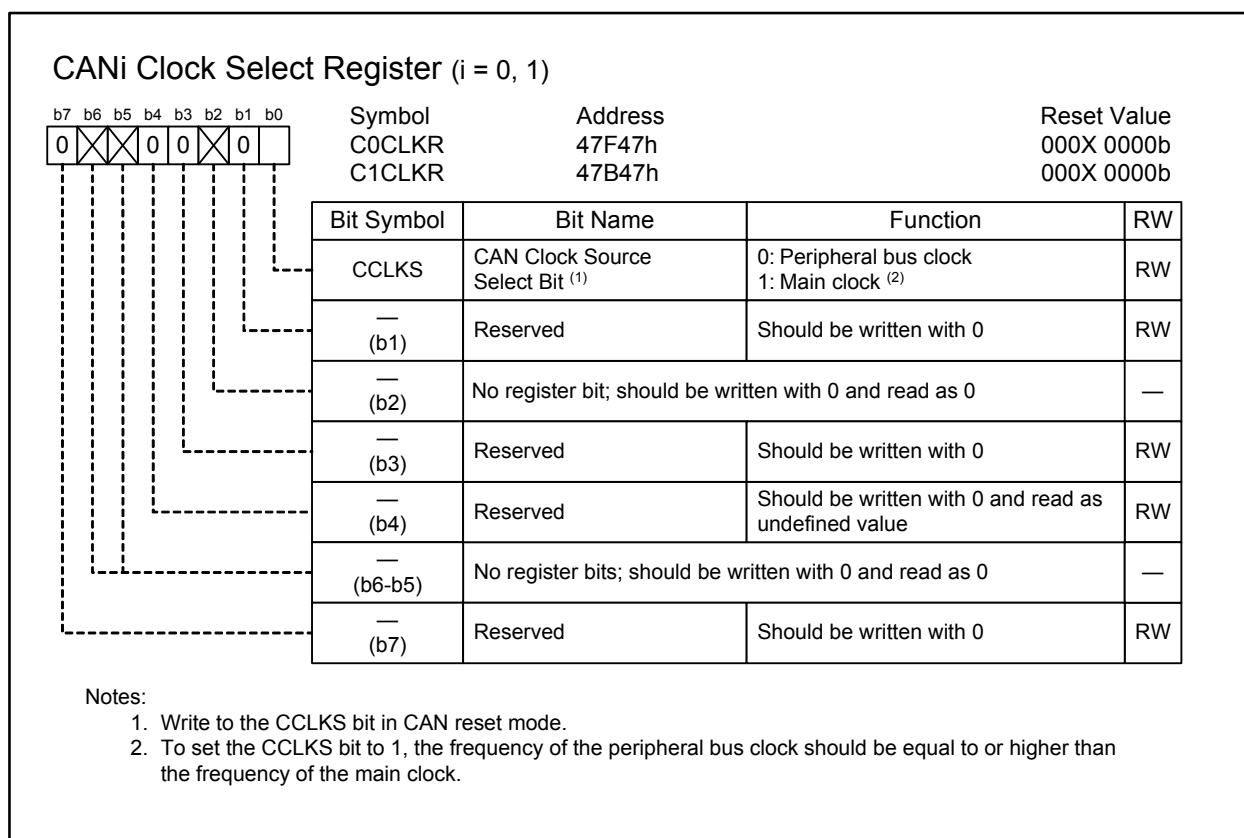


Figure 24.3 Registers C0CLKR and C1CLKR

### 24.1.2.1 CCLKS Bit

When the CCLKS bit is set to 0, the CAN clock source (f<sub>CAN</sub>) originates from the PLL frequency synthesizer.

When this bit is set to 1, f<sub>CAN</sub> originates directly from the external XIN pin bypassing the PLL.

### 24.1.3 CANi Bit Configuration Register (CiBCR) (i = 0, 1)

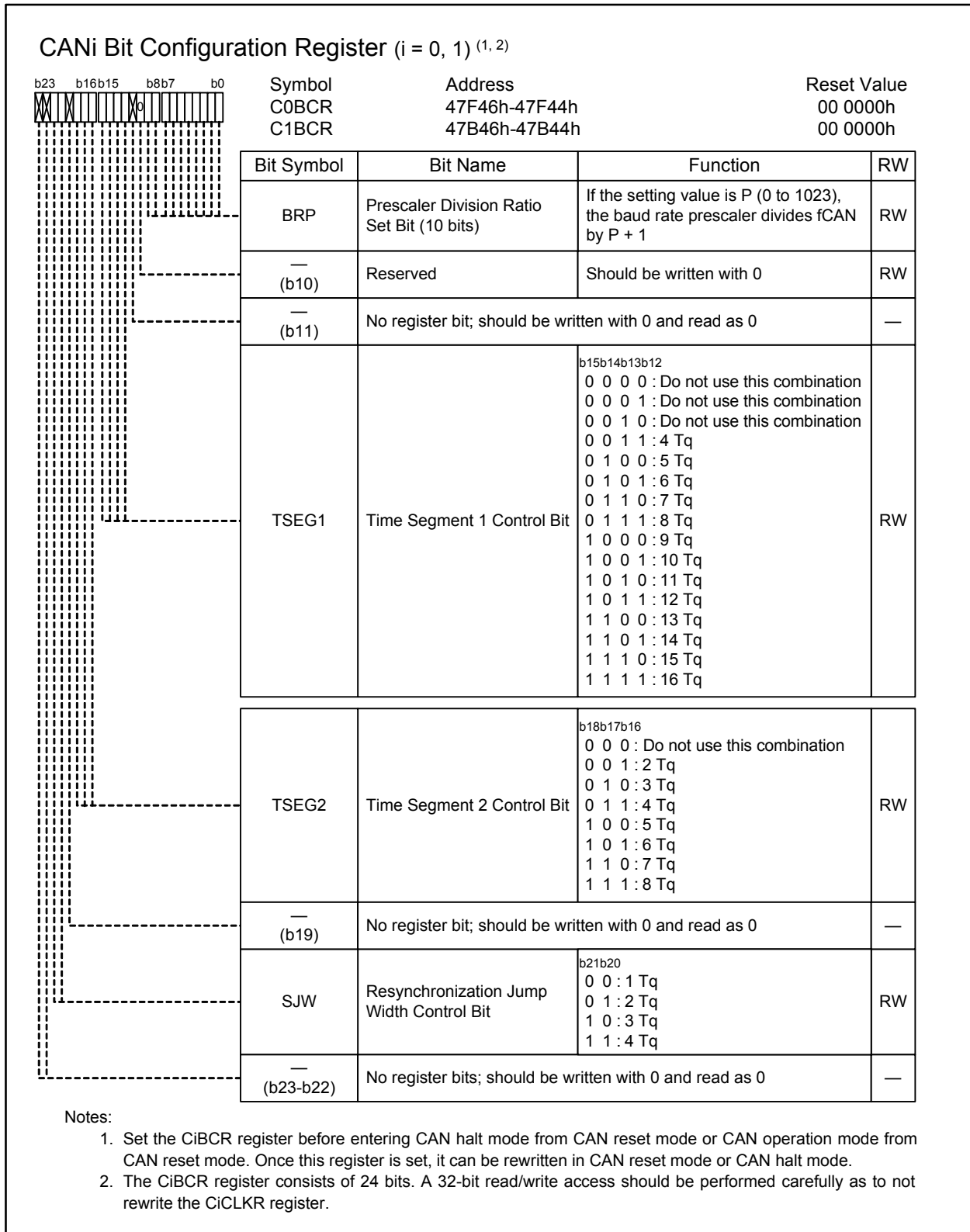


Figure 24.4 Registers C0BCR and C1BCR

Refer to 24.3 “CAN Communication Speed Configuration” for the bit timing configuration.

### 24.1.3.1 BRP Bit

The BRP bit sets the frequency of the CAN communication clock (fCANCLK).  
One fCANCLK cycle is measured as Time Quantum (Tq).

### 24.1.3.2 TSEG1 Bit

Set the TSEG1 bit to specify the total length of the propagation time segment (PROP\_SEG) and phase buffer segment 1 (PHASE\_SEG1) with the value of Tq.  
A value from 4 to 16 Tq can be set.

### 24.1.3.3 TSEG2 Bit

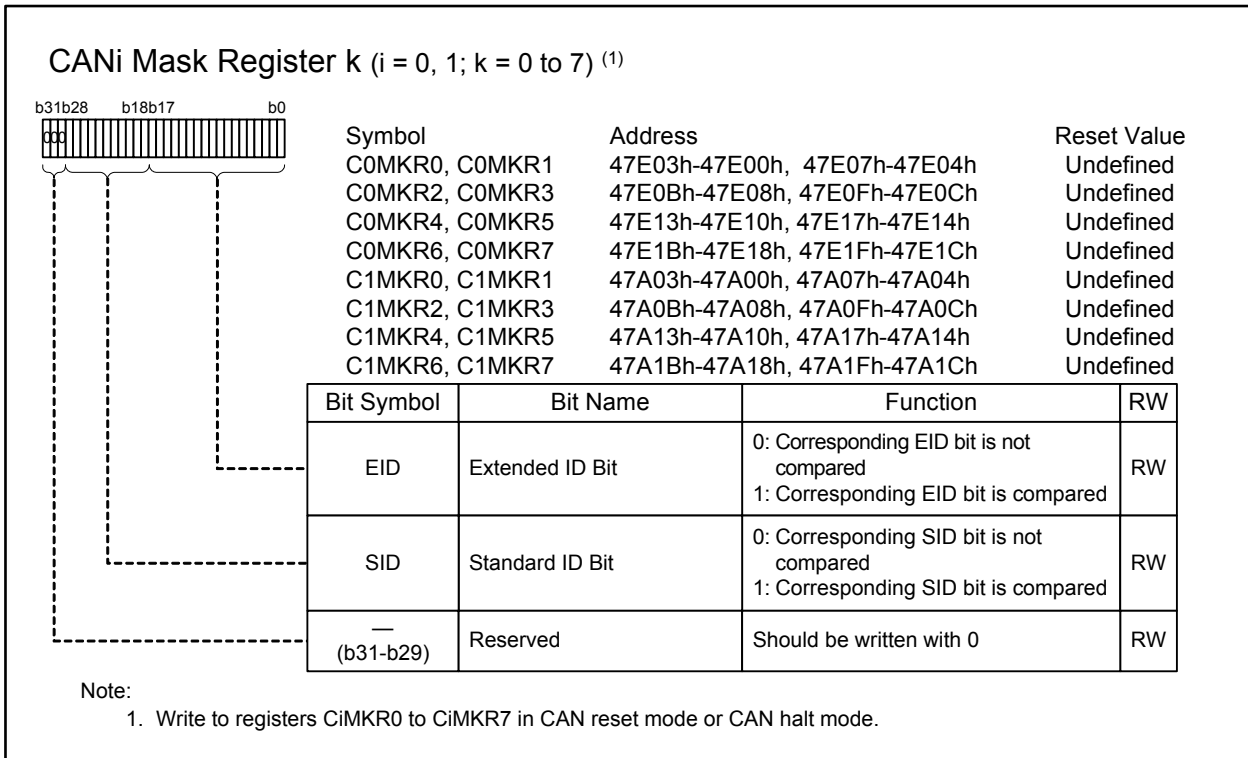
Set the TSEG2 bit to specify the length of phase buffer segment TSEG2 (PHASE\_SEG2) with the value of Tq.  
A value from 2 to 8 Tq can be set.  
Set the value smaller than that of the TSEG1 bit.

### 24.1.3.4 SJW Bit

Set the SJW bit to specify the resynchronization jump width with the value of Tq.  
A value from 1 to 4 Tq can be set.  
Set the value smaller than or equal to that of the TSEG2 bit.



### 24.1.4 CANi Mask Register k (CiMKRk) (i = 0, 1; k = 0 to 7)



**Figure 24.5 Registers C0MKR0 to C1MKR7**

Refer to 24.5 “Acceptance Filtering and Masking Function” for the masking function in FIFO mailbox mode.

#### 24.1.4.1 EID Bit

The EID bit is the filter mask bit corresponding to the CAN extended ID bit. This bit is used to receive extended ID messages.

When the EID bit is 0, the corresponding EID bit is not compared for the received ID and the mailbox ID.

When this bit is 1, the corresponding EID bit is compared for the received ID and the mailbox ID.

#### 24.1.4.2 SID Bit

The SID bit is the filter mask bit corresponding to the CAN standard ID bit. This bit is used to receive both standard ID and extended ID messages.

When the SID bit is 0, the corresponding SID bit is not compared for the received ID and the mailbox ID.

When this bit is 1, the corresponding SID bit is compared for the received ID and the mailbox ID.

### 24.1.5 CANi FIFO Received ID Compare Register n (CiFIDCR0 and CiFIDCR1) (i = 0, 1; n = 0, 1)

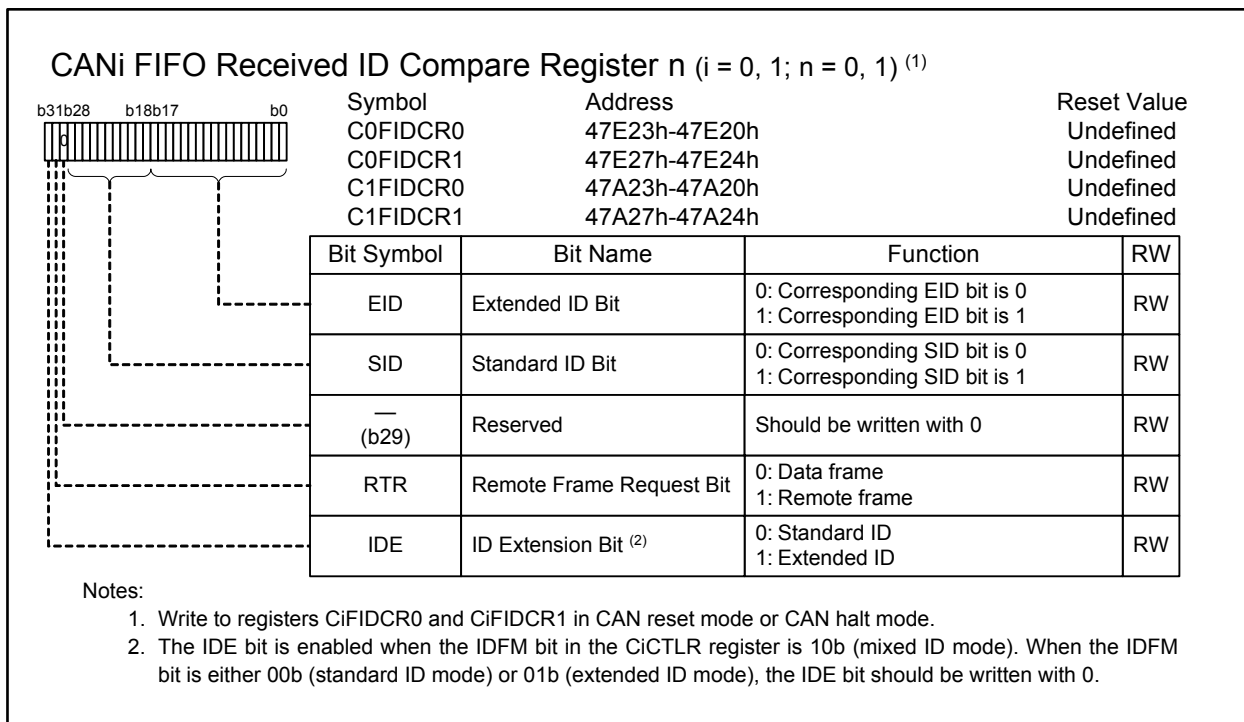


Figure 24.6 Registers C0FIDCR0 to C1FIDCR1

Registers CiFIDCR0 and CiFIDCR1 are enabled when the MBM bit in the CiCTLR register is set to 1 (FIFO mailbox mode). Bits EID, SID, RTR, and IDE in registers CiMB28 to CiMB31 are disabled. Refer to 24.5 “Acceptance Filtering and Masking Function” for details on using these registers.

#### 24.1.5.1 EID Bit

The EID bit sets the extended ID of data frames and remote frames. This bit is used to receive extended ID messages.

#### 24.1.5.2 SID Bit

The SID bit sets the standard ID of data frames and remote frames. This bit is used to receive both standard ID and extended ID messages.

### 24.1.5.3 RTR Bit

The RTR bit sets the specified frame format of data frames or remote frames.

This bit specifies the following operations:

- When both RTR bits in registers CiFIDCR0 and CiFIDCR1 are set to 0, only data frames can be received (i = 0, 1).
- When both RTR bits in registers CiFIDCR0 and CiFIDCR1 are set to 1, only remote frames can be received.
- When the RTR bits in registers CiFIDCR0 and CiFIDCR1 are set with different values, both data frames and remote frames can be received.

### 24.1.5.4 IDE Bit

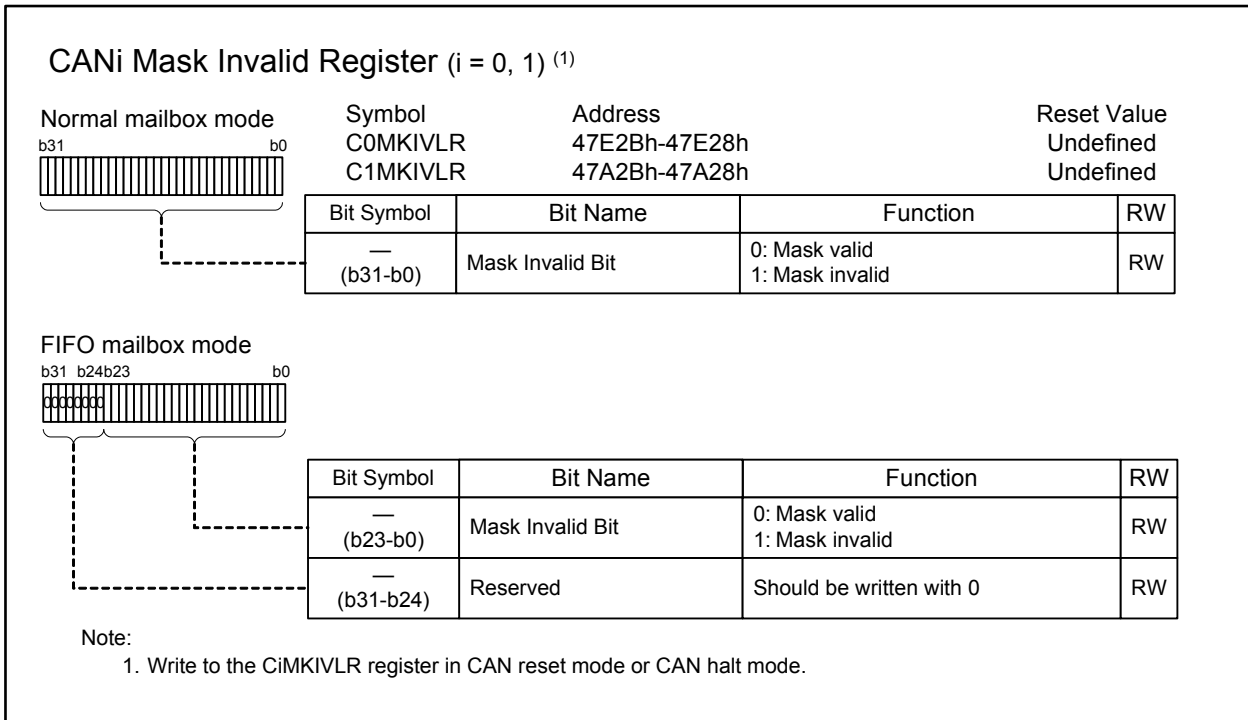
The IDE bit sets the ID format of standard ID or extended ID.

This bit is enabled when the IDFM bit in the CiCTLR register is 10b (mixed ID mode).

When the IDFM bit is 10b, the IDE bit specifies the following operations:

- When both IDE bits in registers CiFIDCR0 and CiFIDCR1 are set to 0, only standard ID frames can be received.
- When both IDE bits in registers CiFIDCR0 and CiFIDCR1 are set to 1, only extended ID frames can be received.
- When the IDE bits in registers CiFIDCR0 and CiFIDCR1 are set with different values, both standard ID and extended ID frames can be received.

### 24.1.6 CANi Mask Invalid Register (CiMKIVLR) (i = 0, 1)



**Figure 24.7 Registers C0MKIVLR and C1MKIVLR**

Each bit corresponds to the mailbox with the same number. When each bit is 1, the acceptance mask for the mailbox corresponding to the bit number is disabled. In this case, a received message is stored in the mailbox only if its ID matches bits SID and EID in the CiMBj register (j = 0 to 31).

### 24.1.7 CANi Mailbox (CiMBj) (i = 0, 1; j = 0 to 31)

Table 24.4 lists the CANi mailbox memory mapping, and Table 24.5 lists the CAN data frame structure. The reset value of the CANi mailbox is undefined.

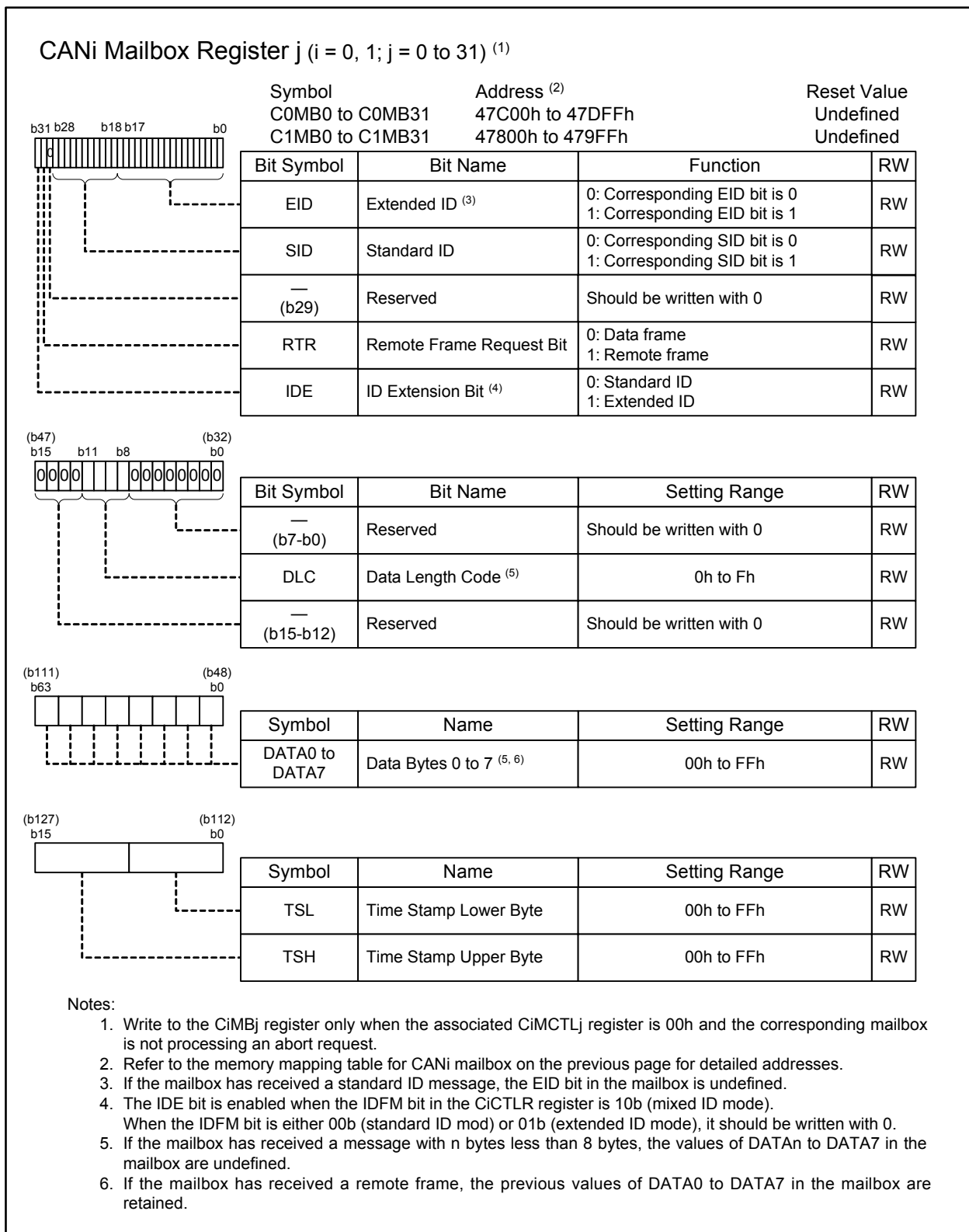
**Table 24.4 CANi Mailbox Memory Mapping (i = 0, 1)**

Address		Message Content
CAN0	CAN1	Memory mapping
$47C00h + j \times 16 + 0$	$47800h + j \times 16 + 0$	EID7 to EID0
$47C00h + j \times 16 + 1$	$47800h + j \times 16 + 1$	EID15 to EID8
$47C00h + j \times 16 + 2$	$47800h + j \times 16 + 2$	SID5 to SID0, EID17, EID16
$47C00h + j \times 16 + 3$	$47800h + j \times 16 + 3$	IDE, RTR, SID10 to SID6
$47C00h + j \times 16 + 4$	$47800h + j \times 16 + 4$	—
$47C00h + j \times 16 + 5$	$47800h + j \times 16 + 5$	Data length code (DLC)
$47C00h + j \times 16 + 6$	$47800h + j \times 16 + 6$	Data byte 0
$47C00h + j \times 16 + 7$	$47800h + j \times 16 + 7$	Data byte 1
⋮	⋮	⋮
⋮	⋮	⋮
⋮	⋮	⋮
$47C00h + j \times 16 + 13$	$47800h + j \times 16 + 13$	Data byte 7
$47C00h + j \times 16 + 14$	$47800h + j \times 16 + 14$	Time stamp lower byte
$47C00h + j \times 16 + 15$	$47800h + j \times 16 + 15$	Time stamp upper byte

j: Mailbox number (j = 0 to 31)

**Table 24.5 CAN Data Frame Structure**

SID10 to SID6	SID5 to SID0	EID17 to EID16	EID15 to EID8	EID7 to EID0	DLC3 to DLC0	DATA0	DATA1	.....	DATA7
------------------	-----------------	-------------------	------------------	-----------------	-----------------	-------	-------	-------	-------

Figure 24.8 Registers C0MB<sub>j</sub> and C1MB<sub>j</sub>

The previous value of each mailbox is retained unless a new message is received.

#### 24.1.7.1 EID Bit

The EID bit sets the extended ID of data frames and remote frames. This bit is used to transmit or receive extended ID messages.

#### 24.1.7.2 SID Bit

The SID bit sets the standard ID of data frames and remote frames. This bit is used to transmit or receive both standard ID and extended ID messages.

#### 24.1.7.3 RTR Bit

The RTR bit sets the frame format of data frames or remote frames.

This bit specifies the following operations:

- The receive mailbox receives only frames with the format specified by the RTR bit.
- The transmit mailbox transmits according to the frame format specified by the RTR bit.
- The receive FIFO mailbox receives the data frame, remote frame, or both frames specified by the RTR bit in registers CiFIDCR0 and CiFIDCR1 (i = 0, 1).
- The transmit FIFO mailbox transmits the data frame or remote frame specified by the RTR bit in the relevant transmitting message.

#### 24.1.7.4 IDE Bit

The IDE bit sets the ID format of standard IDs or extended IDs.

This bit is enabled when the IDFM bit in the CiCTLR register is 10b (mixed ID mode).

When the IDFM bit is 10b, the IDE bit specifies the following operations:

- The receive mailbox receives only the ID format specified by the IDE bit.
- The transmit mailbox transmits according to the ID format specified by the IDE bit.
- The receive FIFO mailbox receives messages with the standard ID, extended ID, or both IDs specified by the IDE bit in registers CiFIDCR0 and CiFIDCR1.
- The transmit FIFO mailbox transmits messages with the standard ID or extended ID specified by the IDE bit in the relevant transmitting message.

### 24.1.7.5 DLC

The DLC sets the number of data bytes to be transmitted in a data frame. When data is requested using a remote frame, the number of data bytes to be requested is set.

When a data frame is received, the number of received data bytes is stored. When a remote frame is received, the number of requested data bytes is stored.

Table 24.6 lists the data length corresponding to the DLC.

**Table 24.6 Data Length Corresponding to the DLC**

DLC[3]	DLC[2]	DLC[1]	DLC[0]	Data Length
0	0	0	0	0 bytes
0	0	0	1	1 byte
0	0	1	0	2 bytes
0	0	1	1	3 bytes
0	1	0	0	4 bytes
0	1	0	1	5 bytes
0	1	1	0	6 bytes
0	1	1	1	7 bytes
1	X	X	X	8 bytes

X: Any value

### 24.1.7.6 DATA0 to DATA7

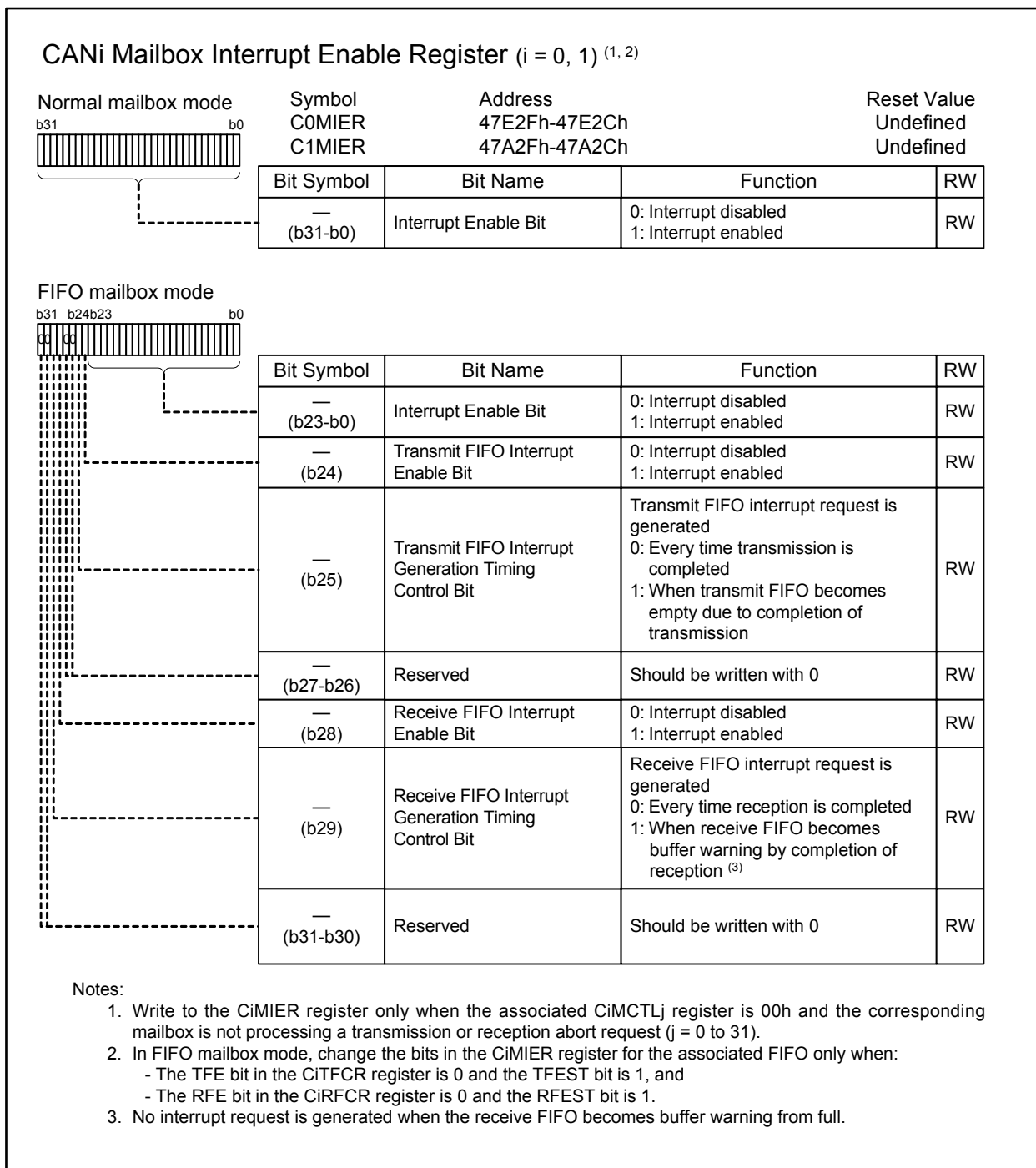
DATA0 to DATA7 store the transmitted or received CAN message data. Transmission or reception starts from DATA0. The bit order on the CAN bus is MSB first, and transmission or reception starts from bit 7.

### 24.1.7.7 TSL and TSH

TSL and TSH store the counter value of the time stamp when received messages are stored in the mailbox.



### 24.1.8 CANi Mailbox Interrupt Enable Register (CiMIER) (i = 0, 1)



**Figure 24.9 Registers COMIER and C1MIER**

Interrupts can be individually enabled for each mailbox.

In normal mailbox mode (bits 0 to 31) and in FIFO mailbox mode (bits 0 to 23), each bit corresponds to the mailbox with the same number. These bits enable or disable transmission/reception complete interrupts for the corresponding mailboxes.

In FIFO mailbox mode, bits 24, 25, 28, and 29 specify whether transmit/receive FIFO interrupts are enabled/disabled and timing when interrupt requests are generated.

“Buffer warning” indicates a state in which the third unread message is stored in the receive FIFO.

### 24.1.9 CANi Message Control Register j (CiMCTLj) (i = 0, 1; j = 0 to 31)

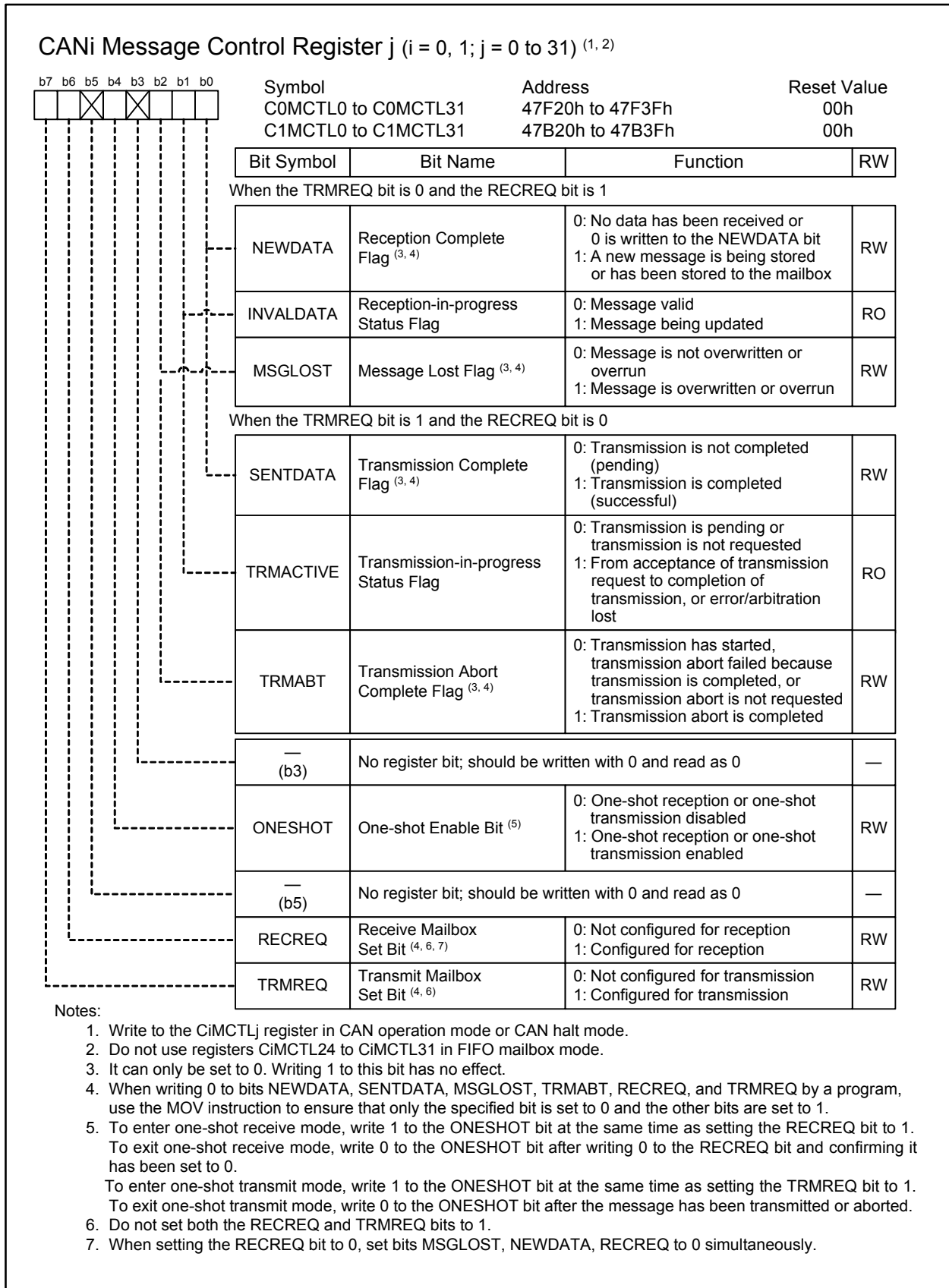


Figure 24.10 Registers C0MCTLj and C1MCTLj

### 24.1.9.1 NEWDATA Bit

The NEWDATA bit becomes 1 when a new message is being stored or has been stored to the mailbox. The timing for setting this bit to 1 is simultaneous with the INVALIDDATA bit.

The NEWDATA bit is set to 0 by writing 0 by a program.

It cannot be set to 0 by a program while the related INVALIDDATA bit is 1.

### 24.1.9.2 SENTDATA Bit

The SENTDATA bit becomes 1 when data transmission from the corresponding mailbox is completed.

This bit is set to 0 by writing 0 by a program.

Set the TRMREQ bit to 0 before setting the SENTDATA bit to 0.

Bits SENTDATA and TRMREQ cannot be set to 0 simultaneously.

To transmit a new message from the corresponding mailbox, set the SENTDATA bit to 0.

### 24.1.9.3 INVALIDDATA Bit

After a message has been received, the INVALIDDATA bit becomes 1 while the received message is being updated into the corresponding mailbox.

This bit becomes 0 immediately after the message has been stored. When the mailbox is read while this bit is 1, the data is undefined.

### 24.1.9.4 TRMACTIVE Bit

The TRMACTIVE bit becomes 1 when the corresponding mailbox of the CAN module begins transmitting a message.

This bit becomes 0 when the CAN module loses CAN bus arbitration, a CAN bus error occurs, or data transmission is completed.

### 24.1.9.5 MSGLOST Bit

While the NEWDATA bit is 1, the MSGLOST bit becomes 1 when the mailbox is overwritten or overrun by a newly received message. This bit becomes 1 at the end of the sixth bit of EOF.

This bit is set to 0 by writing 0 by a program.

In both overwrite and overrun modes, during five cycles of the peripheral bus clock following the sixth bit of EOF, the MSGLOST bit does not become 0 even if it is set to 0 by a program.

### 24.1.9.6 TRMABT Bit

The TRMABT bit becomes 1 in the following cases:

- Following a transmission abort request, the transmission abort is completed before starting transmission.
- Following a transmission abort request, the CAN module detects CAN bus arbitration lost or a CAN bus error.
- In one-shot transmission mode (the RECREQ bit is 0, the TRMREQ bit is 1, and the ONESHOT bit is 1), the CAN module detects CAN bus arbitration lost or a CAN bus error.

The TRMABT bit does not become 1 when data transmission is completed. In this case, the SENTDATA bit becomes 1.

The TRMABT bit can be set to 0 by a program.

### 24.1.9.7 ONESHOT Bit

The ONESHOT bit can be used in receive mode and transmit mode.

#### (1) One-shot Receive Mode

When the ONESHOT bit is set to 1 in receive mode (the RECREQ bit is 1 and the TRMREQ bit is 0), the mailbox receives a message only once. The mailbox does not behave as a receive mailbox after having received a message once. The behavior of bits NEWDATA and INVALIDDATA is the same as in normal reception mode. In one-shot receive mode, the MSGLOST bit does not become 1.

To set the ONESHOT bit to 0, first write 0 to the RECREQ bit and ensure that it has been set to 0.

#### (2) One-shot Transmit Mode

When the ONESHOT bit is set to 1 in transmit mode (the RECREQ bit is 0 and the TRMREQ bit is 1), the CAN module transmits a message only once. The CAN module does not transmit the message again if a CAN bus error or CAN bus arbitration lost occurs. When the transmission is completed, the SENTDATA bit becomes 1. If the transmission cannot be completed due to a CAN bus error or CAN bus arbitration lost, the TRMABT bit becomes 1.

Set the ONESHOT bit to 0 after the SENTDATA or TRMABT bit is set to 1.

### 24.1.9.8 RECREQ Bit

Set the RECREQ bit to select one of the receive modes shown in Table 24.11.

When the RECREQ bit is set to 1, the corresponding mailbox is configured to receive a data frame or a remote frame.

When this bit is set to 0, the corresponding mailbox is not configured for reception of a data frame or a remote frame.

Due to hardware protection, the RECREQ bit cannot be set to 0 by a program during the following period:

Hardware protection is started

- from the acceptance filter procedure (the beginning of the CRC field)

Hardware protection is released

- for the mailbox that is specified to receive the incoming message, after the received data is stored into the mailbox or a CAN bus error occurs (i.e. a maximum period of hardware protection is from the beginning of the CRC field to the end of the seventh bit of EOF)
- for the other mailboxes, after the acceptance filter procedure
- if no mailbox is specified to receive the message, after the acceptance filter procedure

When setting the RECREQ bit to 1, do not set 1 to the TRMREQ bit.

To change the configuration of a mailbox from transmit to receive, first abort the transmission and then set bits SENTDATA and TRMABT to 0 before changing to receive.

### 24.1.9.9 TRMREQ Bit

The TRMREQ bit selects transmit modes shown in Table 24.11.

When this bit is set to 1, the corresponding mailbox is configured to transmit a data frame or a remote frame.

When this bit is set to 0, the corresponding mailbox is not configured to transmit a data frame or a remote frame.

If the TRMREQ bit is changed from 1 to 0 to cancel the corresponding transmission request, either the TRMABT or SENTDATA bit is set to 1.

When setting the TRMREQ bit to 1, do not set the RECREQ bit to 1.

To change the configuration of a mailbox from receive to transmit, first abort the reception and then set bits NEWDATA and MSGLOST to 0 before changing to transmit.

### 24.1.10 CAN<sub>i</sub> Receive FIFO Control Register (CiRFCR) (i = 0, 1)

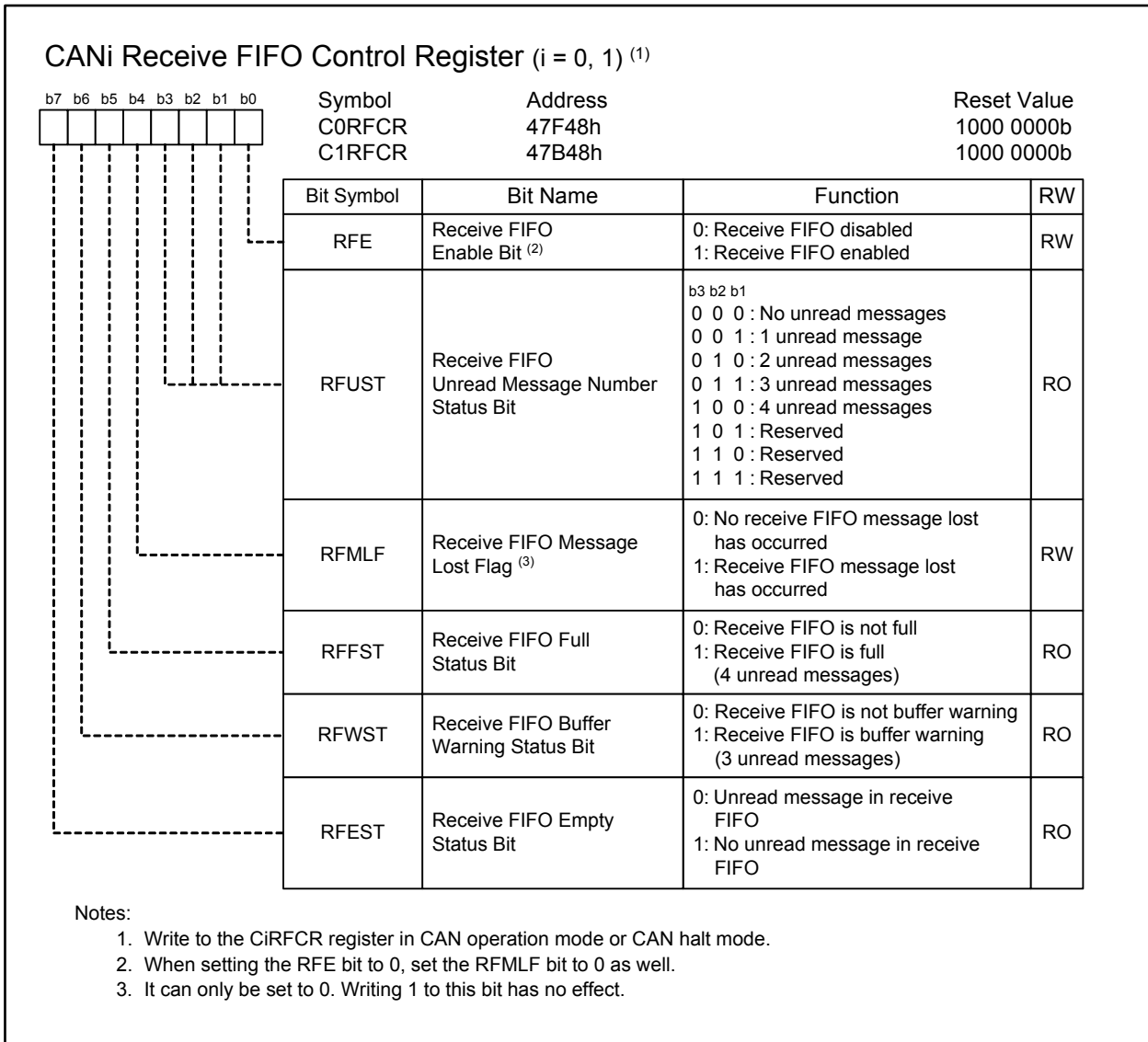


Figure 24.11 Registers C0RFCR and C1RFCR

### 24.1.10.1 RFE Bit

When the RFE bit is set to 1, the receive FIFO is enabled.

When this bit is set to 0, the receive FIFO is disabled for reception and becomes empty (the RFEST bit is 1).

Do not set this bit to 1 in normal mailbox mode (the MBM bit in the CiCTLR register is 0 (i = 0, 1)).

Due to hardware protection, the RFE bit cannot be set to 0 by a program during the following period:

Hardware protection is started

- from the acceptance filter procedure (the beginning of the CRC field)

Hardware protection is released

- if the receive FIFO is specified to receive the incoming message, after the received data is stored into the receive FIFO or a CAN bus error occurs (i.e. a maximum period of hardware protection is from the beginning of the CRC field to the end of the seventh bit of EOF).
- if the receive FIFO is not specified to receive the message, after the acceptance filter procedure.

### 24.1.10.2 RFUST Bit

The RFUST bit indicates the number of unread messages in the receive FIFO.

The value of this bit is initialized to 000b when the RFE bit is set to 0.

### 24.1.10.3 RFMLF Bit

The RFMLF bit becomes 1 (receive FIFO message lost has occurred) when the receive FIFO receives a new message and the receive FIFO is full. This bit becomes 1 at the end of the sixth bit of EOF.

The RFMLF bit is set to 0 by writing 0 by a program.

In both overwrite and overrun modes, this bit cannot be set to 0 (no receive FIFO message lost has occurred) by a program due to hardware protection during the five cycles of the peripheral bus clock following the sixth bit of EOF, when the receive FIFO is full and determined to receive the message.

### 24.1.10.4 RFFST Bit

The RFFST bit becomes 1 (receive FIFO is full) when there are four unread messages in the receive FIFO. This bit becomes 0 (receive FIFO is not full) when there are less than four unread messages in the receive FIFO. This bit becomes 0 when the RFE bit is 0.

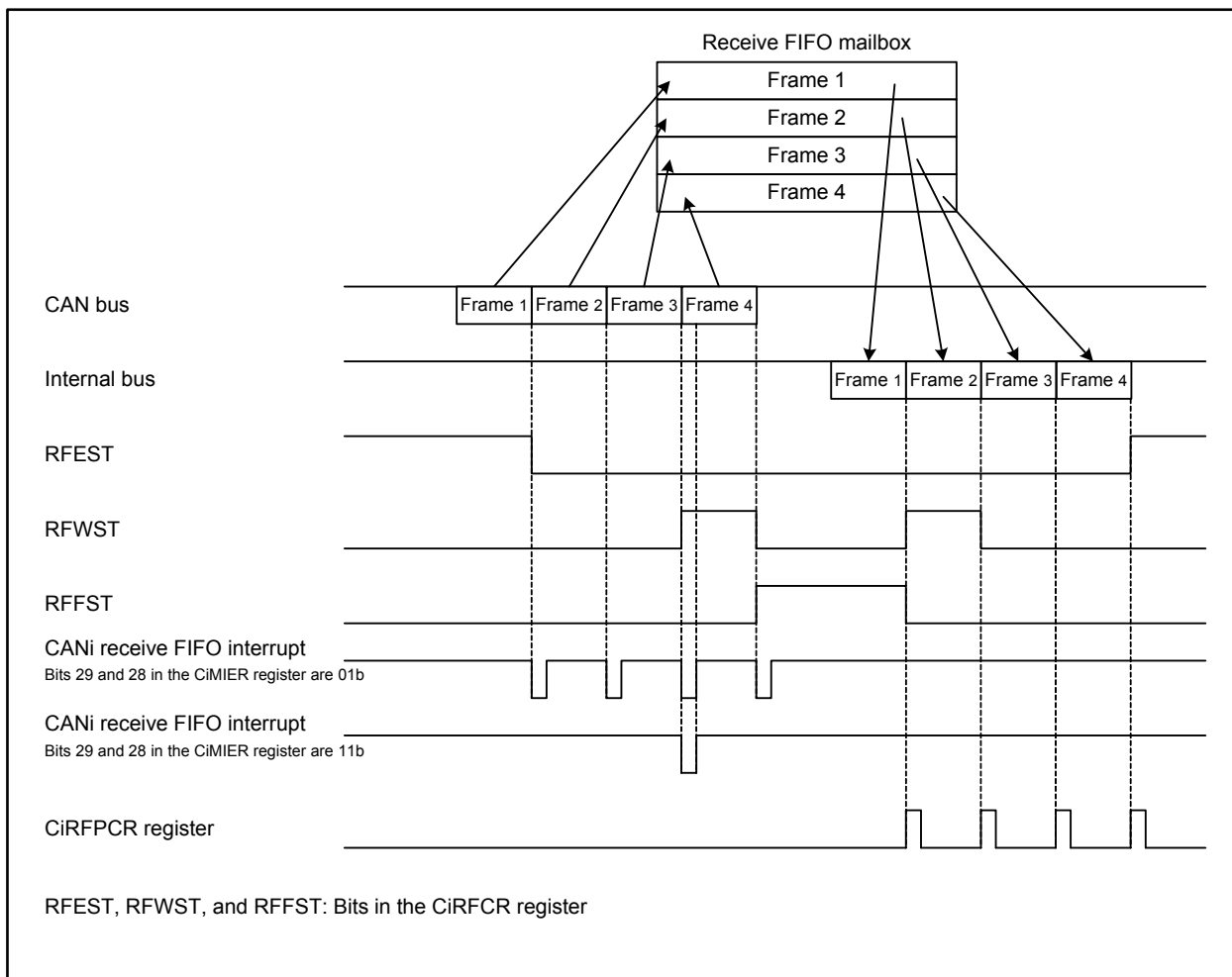
### 24.1.10.5 RFWST Bit

The RFWST bit becomes 1 (receive FIFO is buffer warning) when there are three unread messages in the receive FIFO. This bit becomes 0 (receive FIFO is not buffer warning) when there are less than three or equal to four unread messages in the receive FIFO. This bit becomes 0 when the RFE bit is 0.

### 24.1.10.6 RFEST Bit

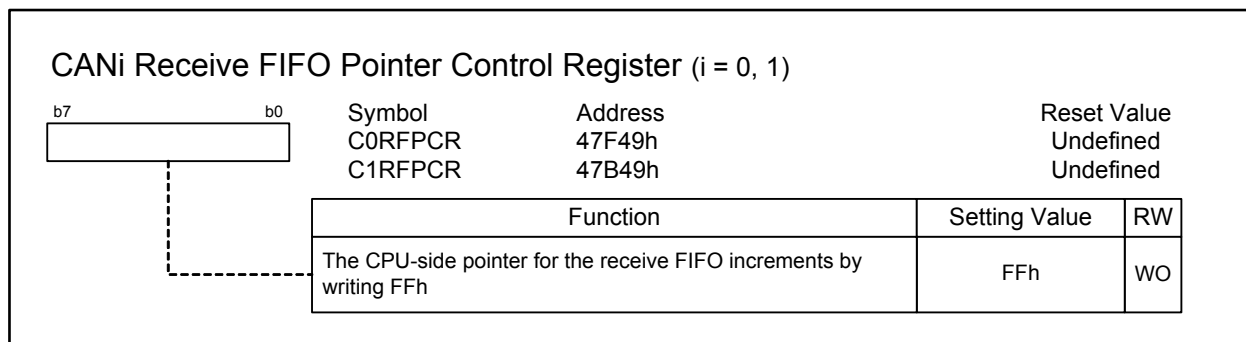
The RFEST bit becomes 1 (no unread message in receive FIFO) when there are no unread messages in the receive FIFO. This bit becomes 1 when the RFE bit is set to 0. The RFEST bit becomes 0 (unread message in receive FIFO) when there is one or more unread messages in the receive FIFO.

Figure 24.12 shows receive FIFO mailbox operation.



**Figure 24.12 Receive FIFO Mailbox Operation (Bits 29 and 28 in CiMIER Register are 01b and 11b) (i = 0, 1)**

### 24.1.11 CANi Receive FIFO Pointer Control Register (CiRFPCR) (i = 0, 1)



**Figure 24.13 Registers C0RFPCR and C1RFPCR**

When there are messages in the receive FIFO, write FFh to the CiRFPCR register by a program to increment the CPU-side pointer for the receive FIFO to the next mailbox location.

Do not write to the CiRFPCR register when the RFE bit in the CiRFPCR register is 0 (receive FIFO disabled).

Both the CAN-side pointer and the CPU-side pointer increment when a new message is received and the RFFST bit is 1 (receive FIFO is full) in overwrite mode. When the RFMLF bit is 1 in this condition, the CPU-side pointer does not increment by writing to the CiRFPCR register by a program.



### 24.1.12 CAN<sub>i</sub> Transmit FIFO Control Register (CiTFCR) (i = 0, 1)

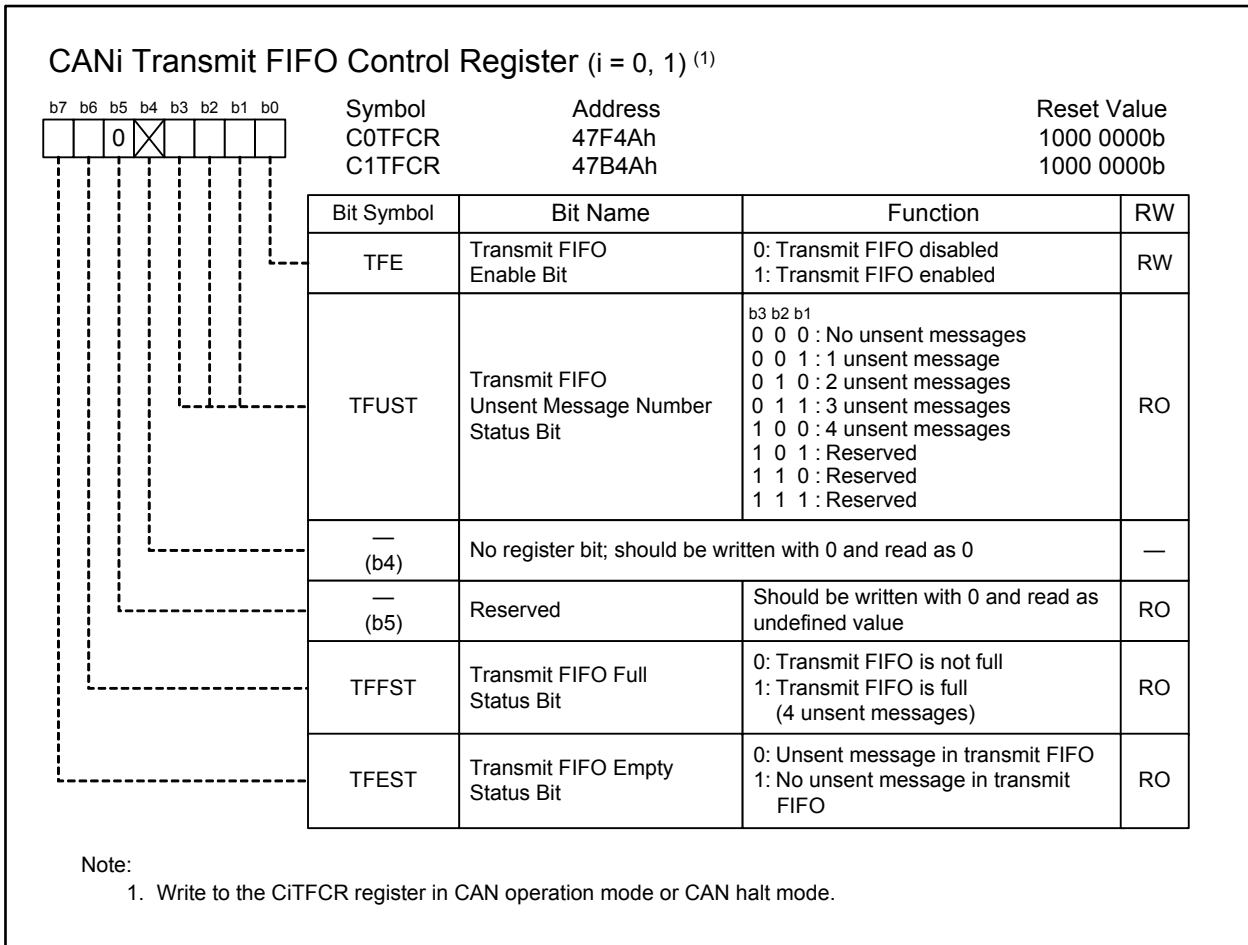


Figure 24.14 Registers C0TFCR and C1TFCR

#### 24.1.12.1 TFE Bit

When the TFE bit is set to 1, the transmit FIFO is enabled.

When this bit is set to 0, the transmit FIFO becomes empty (TFEST bit is 1) and then unsent messages from the transmit FIFO are lost as described below:

- If a message from the transmit FIFO is not scheduled for the next transmission or during transmission.
- Following the completion of a transmission, a CAN bus error, CAN bus arbitration lost, or entry to CAN halt mode if a message from the transmit FIFO is scheduled for the next transmission or already during transmission.

Before setting the TFE bit to 1 again, ensure that the TFEST bit is 1.

After setting the TFE bit to 1, write transmit data to the CiMB24 register.

Do not set this bit to 1 in normal mailbox mode (MBM bit in the CiCTLR register is 0).

#### 24.1.12.2 TFUST Bit

The TFUST bit indicates the number of unsent messages in the transmit FIFO.

After the TFE bit is set to 0, the value of the TFUST bit is initialized to 000b when transmission abort or transmission is completed.

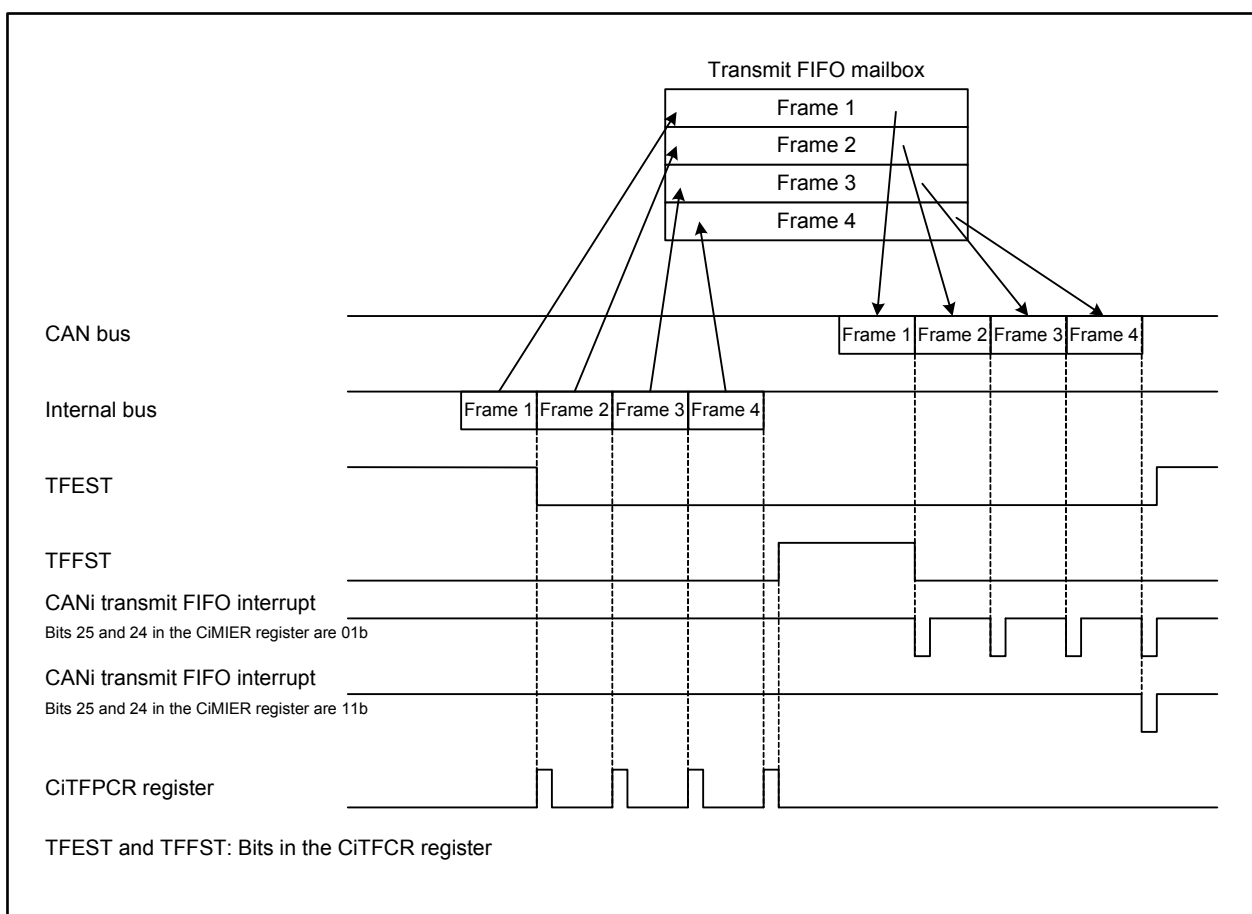
### 24.1.12.3 TFFST Bit

The TFFST bit becomes 1 (transmit FIFO is full) when there are four unsend messages in the transmit FIFO. This bit becomes 0 (transmit FIFO is not full) when there are less than four unsend messages in the transmit FIFO. This bit becomes 0 when a transmission from the transmit FIFO has been aborted.

### 24.1.12.4 TFEST Bit

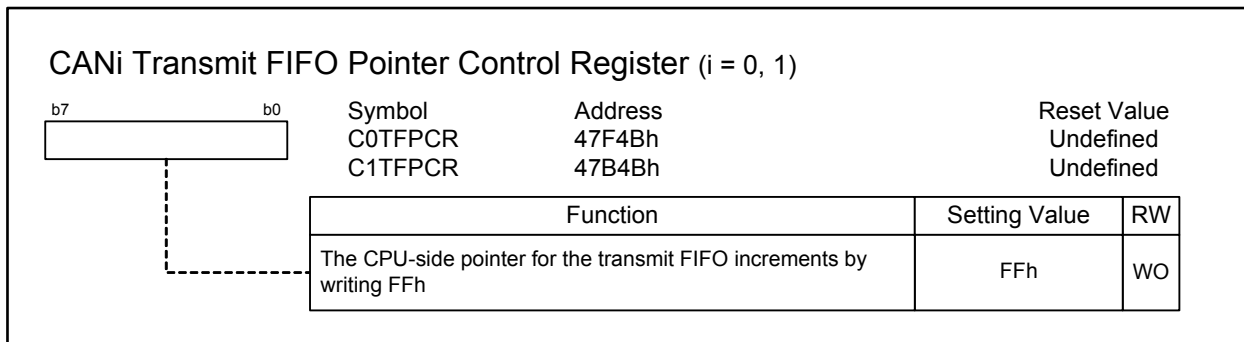
The TFEST bit becomes 1 (no unsend message in transmit FIFO) when there are no unsend messages in the transmit FIFO. This bit becomes 1 when transmission from the transmit FIFO has been aborted. The TFEST bit becomes 0 (unsend message in transmit FIFO) when there is at least one unsend messages in the transmit FIFO.

Figure 24.15 shows transmit FIFO mailbox operation.



**Figure 24.15 Transmit FIFO Mailbox Operation (Bits 25 and 24 in CiMIER Register are 01b and 11b) (i = 0, 1)**

### 24.1.13 CANi Transmit FIFO Pointer Control Register (CiTFPCR) (i = 0, 1)



**Figure 24.16 Registers C0TFPCR and C1TFPCR**

When the transmit FIFO is not full, write FFh to the CiTFPCR register by a program to increment the CPU-side pointer for the transmit FIFO to the next mailbox location.

Do not write to the CiTFPCR register when the TFE bit in the CiTFPCR register is 0 (transmit FIFO disabled).

### 24.1.14 CAN<sub>i</sub> Status Register (CiSTR) (i = 0, 1)

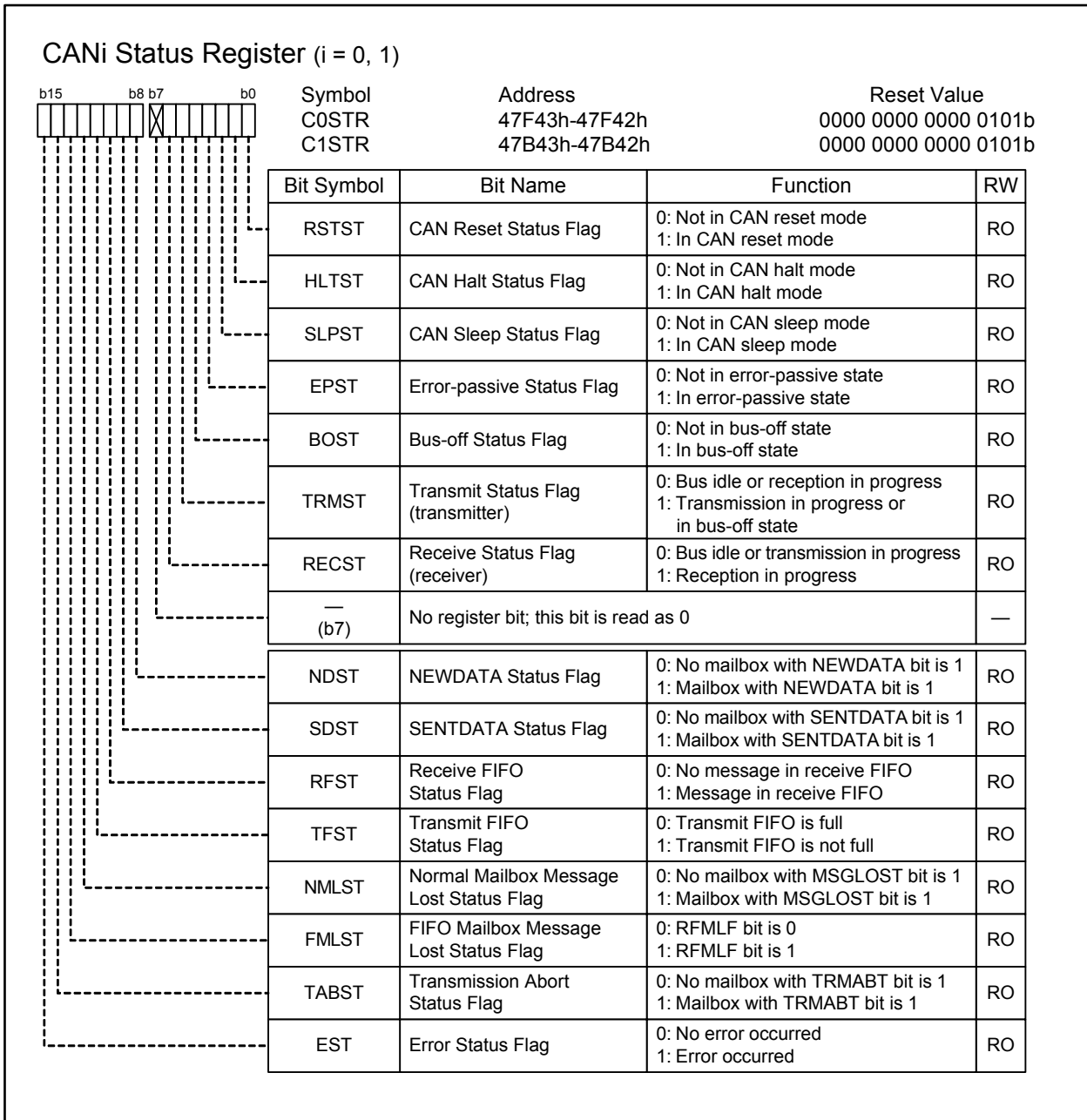


Figure 24.17 Registers C0STR and C1STR

#### 24.1.14.1 RSTST Bit

The RSTST bit becomes 1 when the CAN module enters CAN reset mode.

This bit is 0 when the CAN module is not in CAN reset mode.

Even when the state is changed from CAN reset mode to CAN sleep mode, the RSTST bit remains 1.

#### 24.1.14.2 HLTST Bit

The HLTST bit becomes 1 when the CAN module enters CAN halt mode.

This bit is 0 when the CAN module is not in CAN halt mode.

Even when the state is changed from CAN halt mode to CAN sleep mode, the HLTST bit remains 1.

#### 24.1.14.3 SLPST Bit

The SLPST bit becomes 1 when the CAN module enters CAN sleep mode.

This bit is 0 when the CAN module is not in CAN sleep mode.

#### 24.1.14.4 EPST Bit

The EPST bit becomes 1 when the value of the CiTECR or CiRECR register exceeds 127 and the CAN module enters the error-passive state ( $128 \leq \text{TEC} < 256$  or  $128 \leq \text{REC} < 256$ ) ( $i = 0, 1$ ). This bit is 0 when the CAN module is not in the error-passive state.

TEC indicates the value of the transmit error counter (CiTECR register) and REC indicates the value of the receive error counter (CiRECR register).

#### 24.1.14.5 BOST Bit

The BOST bit becomes 1 when the value of the CiTECR register exceeds 255 and the CAN module enters the bus-off state ( $\text{TEC} \geq 256$ ). This bit is 0 when the CAN module is not in the bus-off state.

#### 24.1.14.6 TRMST Bit

The TRMST bit becomes 1 when the CAN module performs as a transmitter node or enters the bus-off state.

This bit becomes 0 when the CAN module performs as a receiver node or enters the bus-idle state.

#### 24.1.14.7 RECST Bit

The RECST bit becomes 1 when the CAN module performs as a receiver node.

This bit becomes 0 when the CAN module performs as a transmitter node or enters the bus-idle state.

#### 24.1.14.8 NDST Bit

The NDST bit becomes 1 when at least one NEWDATA bit in the CiMCTLj register is 1 regardless of the value of the CiMIER register ( $j = 0$  to 31).

The NDST bit becomes 0 when all NEWDATA bits are 0.

#### 24.1.14.9 SDST Bit

The SDST bit becomes 1 when at least one SENTDATA bit in the CiMCTLj register is 1 regardless of the value of the CiMIER register (i = 0, 1; j = 0 to 31).

The SDST bit becomes 0 when all SENTDATA bits are 0.

#### 24.1.14.10 RFST Bit

The RFST bit is 1 when there are messages in the receive FIFO.

This bit is 0 when the receive FIFO is empty.

This bit becomes 0 when normal mailbox mode is selected.

#### 24.1.14.11 TFST Bit

The TFST bit is 1 when the transmit FIFO is not full.

This bit is 0 when the transmit FIFO is full.

This bit becomes 0 when normal mailbox mode is selected.

#### 24.1.14.12 NMLST Bit

The NMLST bit becomes 1 when at least one MSGLOST bit in the CiMCTLj register is 1 regardless of the value of the CiMIER register.

The NMLST bit becomes 0 when all MSGLOST bits are 0.

#### 24.1.14.13 FMLST Bit

The FMLST bit becomes 1 when the RFMLF bit in the CiRFCR register is 1 regardless of the value of the CiMIER register.

The FMLST bit becomes 0 when the RFMLF bit is 0.

#### 24.1.14.14 TABST Bit

The TABST bit becomes 1 when at least one TRMABT bit in the CiMCTLj register is 1 regardless of the value of the CiMIER register.

The TABST bit becomes 0 when all TRMABT bits are 0.

#### 24.1.14.15 EST Bit

The EST bit becomes 1 when at least one error is detected by the CiEIFR register regardless of the value of the CiEIER register.

This bit becomes 0 when no error is detected by the CiEIFR register.

### 24.1.15 CANi Mailbox Search Mode Register (CiMSMR) (i = 0, 1)

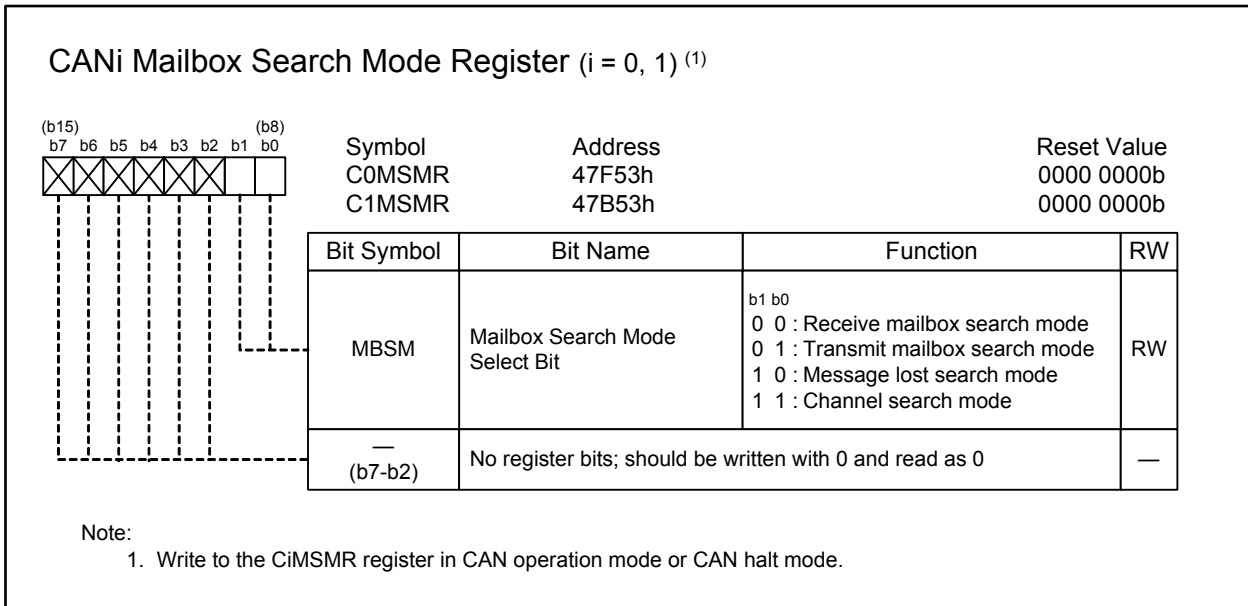


Figure 24.18 Registers C0MSMR and C1MSMR

#### 24.1.15.1 MBSM Bit

Set the MBSM bit to select the search mode for the mailbox search function.

When this bit is 00b, receive mailbox search mode is selected. In this mode, the search targets are the NEWDATA bit in the CiMCTLj register for the normal mailbox and the RFEST bit in the CiRFCR register (j = 0 to 31).

When the MBSM bit is 01b, transmit mailbox search mode is selected. In this mode, the search target is the SENTDATA bit in the CiMCTLj register.

When the MBSM bit is 10b, message lost search mode is selected. In this mode, the search targets are the MSGLOST bit in the CiMCTLj register for the normal mailbox and the RFMLF bit in the CiRFCR register.

When the MBSM bit is 11b, channel search mode is selected. In this mode, the search target is the CiCSSR register.

Refer to 24.1.17 “CANi Channel Search Support Register (CiCSSR) (i = 0, 1)”.

### 24.1.16 CANi Mailbox Search Status Register (CiMSSR) (i = 0, 1)

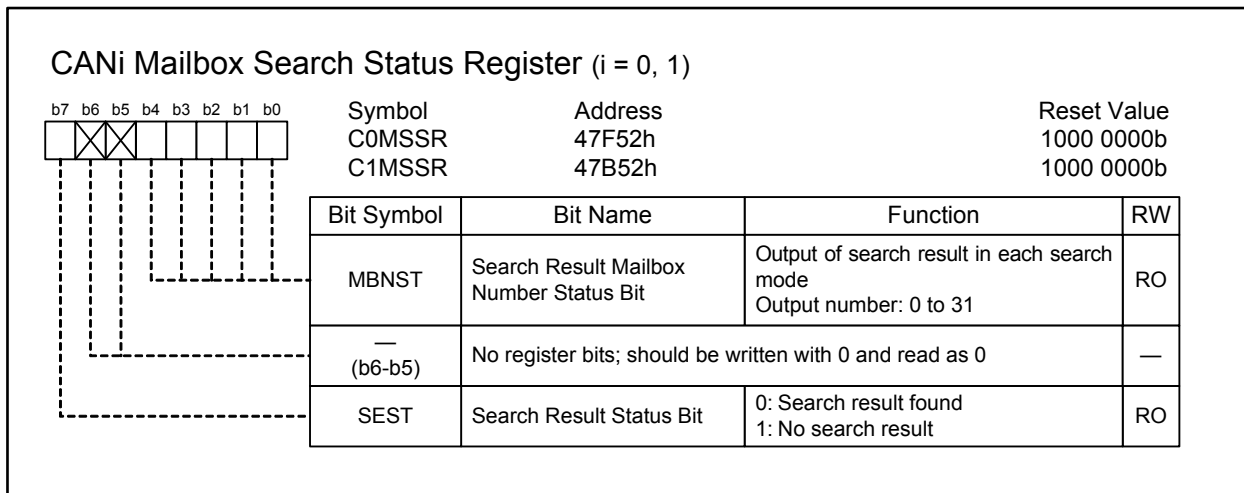


Figure 24.19 Registers C0MSSR and C1MSSR



### 24.1.16.1 MBNST Bit

The MBNST bit outputs the smallest mailbox number that is searched in each mode of the CiMSMR register ( $i = 0, 1$ ).

In receive mailbox, transmit mailbox, and message lost search modes, the value of the mailbox, i.e., the search result to be output, is updated as follows:

- When the NEWDATA, SENTDATA, or MSGLOST bit for the output mailbox is set to 0.
- When the NEWDATA, SENTDATA, or MSGLOST bit for a higher-priority mailbox is set to 1.

In receive mailbox search and message lost search modes, the receive FIFO (mailbox [28]) is output when there are messages in the receive FIFO, and there are no unread received messages or lost messages in any of the normal mailboxes (mailboxes [0] to [23]).

In transmit mailbox search mode, the transmit FIFO (mailbox [24]) is not output.

Table 24.7 lists the operation of MBNST bit in FIFO mailbox mode.

**Table 24.7 Operation of MBNST Bit in FIFO Mailbox Mode**

MBSM Bit	Mailbox [24] (Transmit FIFO)	Mailbox [28] (Receive FIFO)
00b	Mailbox [24] is not output	Mailbox [28] is output when no NEWDATA bit for the normal mailbox is set to 1 and there are messages in the receive FIFO
01b		Mailbox [28] is not output
10b		Mailbox [28] is output when no MSGLOST bit for the normal mailbox is set to 1 and the RFMLF bit is set to 1 in the receive FIFO
11b		Mailbox [28] is not output

In channel search mode, the MBNST bit outputs the corresponding channel number. After the CiMSSR register is read by a program, the next target channel number is output.

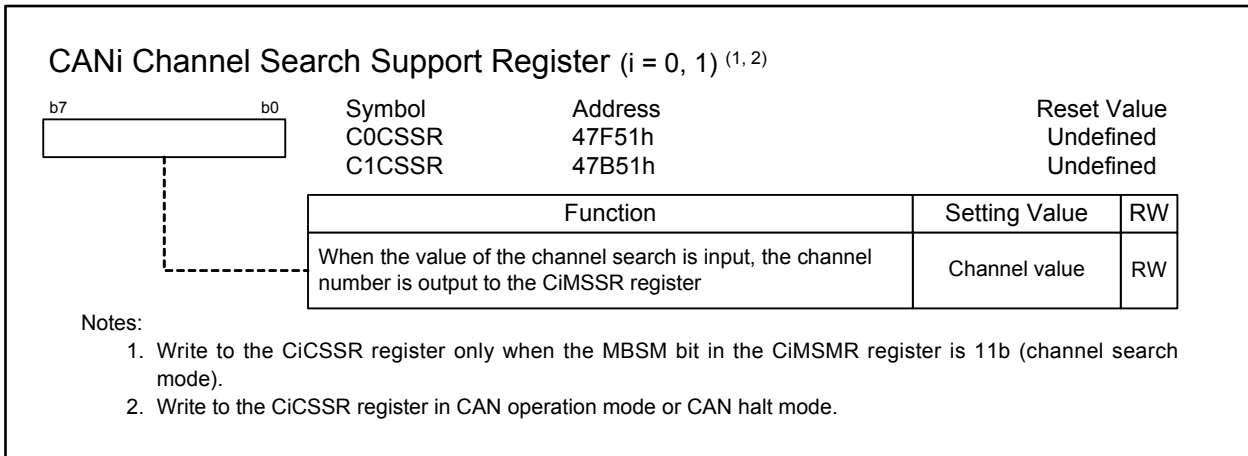
### 24.1.16.2 SEST Bit

The SEST bit becomes 1 when no corresponding mailbox is found after searching all mailboxes.

For example, in transmit mailbox search mode, the SEST bit becomes 1 when no SENTDATA bit for mailboxes is 1. The SEST bit becomes 0 when at least one SENTDATA bit is 1.

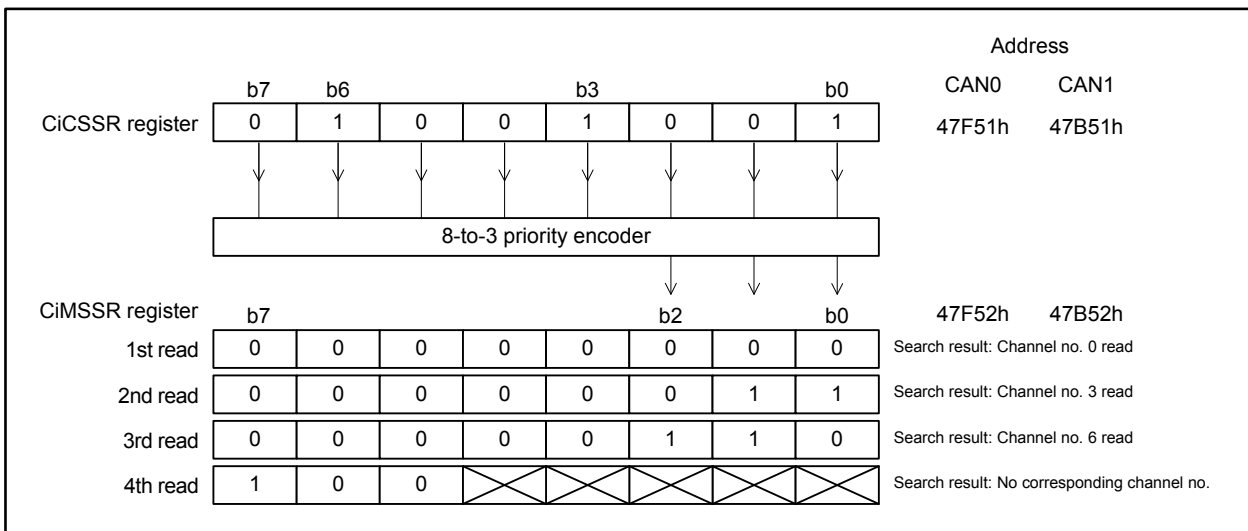
When the SEST bit is 1, the value of the MBNST bit is undefined.

### 24.1.17 CAN<sub>i</sub> Channel Search Support Register (CiCSSR) (i = 0, 1)



**Figure 24.20 Registers C0CSSR and C1CSSR**

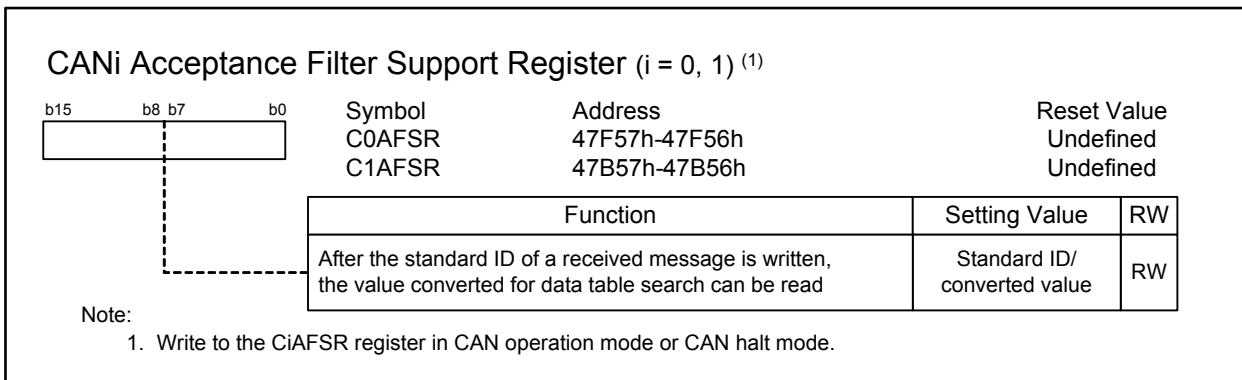
The bits in the CiCSSR register, which are set to 1, are encoded by an 8-to-3 priority encoder (the lower bit position, the higher priority) and output to the MBNST bits in the CiMSSR register. The value of the CiMSSR register is updated whenever the CiMSSR register is read. Figure 24.21 shows the write and read of registers CiCSSR and CiMSSR.



**Figure 24.21 Write and Read of Registers CiCSSR and CiMSSR (i = 0, 1)**

The value of the CiCSSR register is also updated whenever the CiMSSR register is read. When the CiCSSR register is read, the value before the 8-to-3 priority encoder conversion is read.

### 24.1.18 CANi Acceptance Filter Support Register (CiAFSR) (i = 0, 1)



**Figure 24.22 Registers C0AFSR and C1AFSR**

The acceptance filter support unit (ASU) can be used for data table (8 bits × 256) search. In the data table, all standard IDs created by the user are set to be enabled/disabled in bit units. When the CiAFSR register is written with 16-bit unit data including the SID bit in the CiMBj register, in which a received ID is stored, a decoded row (byte offset) position and column (bit) position for data table search can be read (j = 0 to 31). The ASU can be used for standard (11-bit) IDs only.

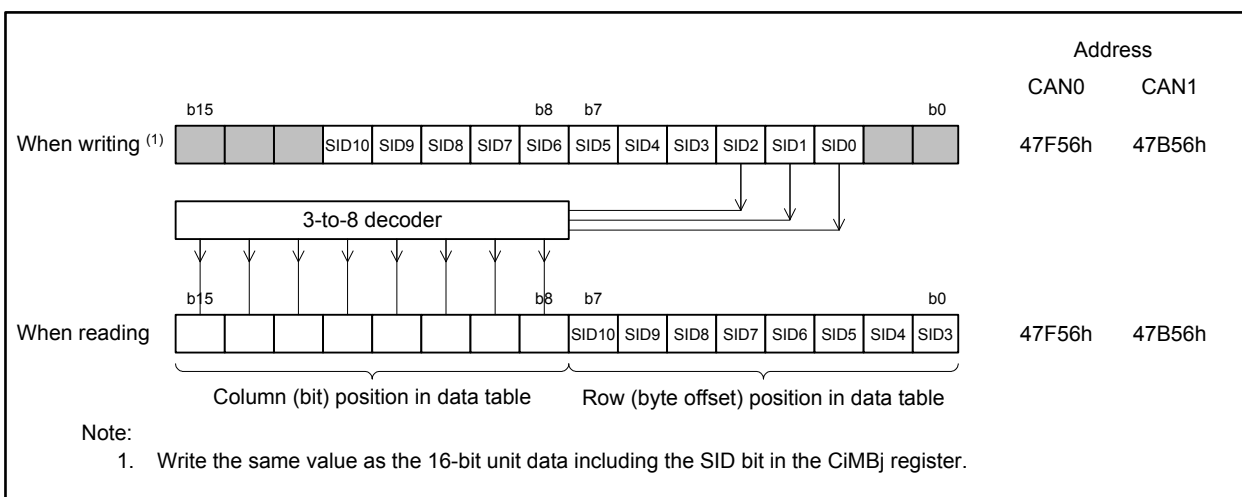
The ASU is enabled in the following cases:

- When the ID to receive cannot be masked by the acceptance filter.

Example: IDs to receive: 078h, 087h, 111h

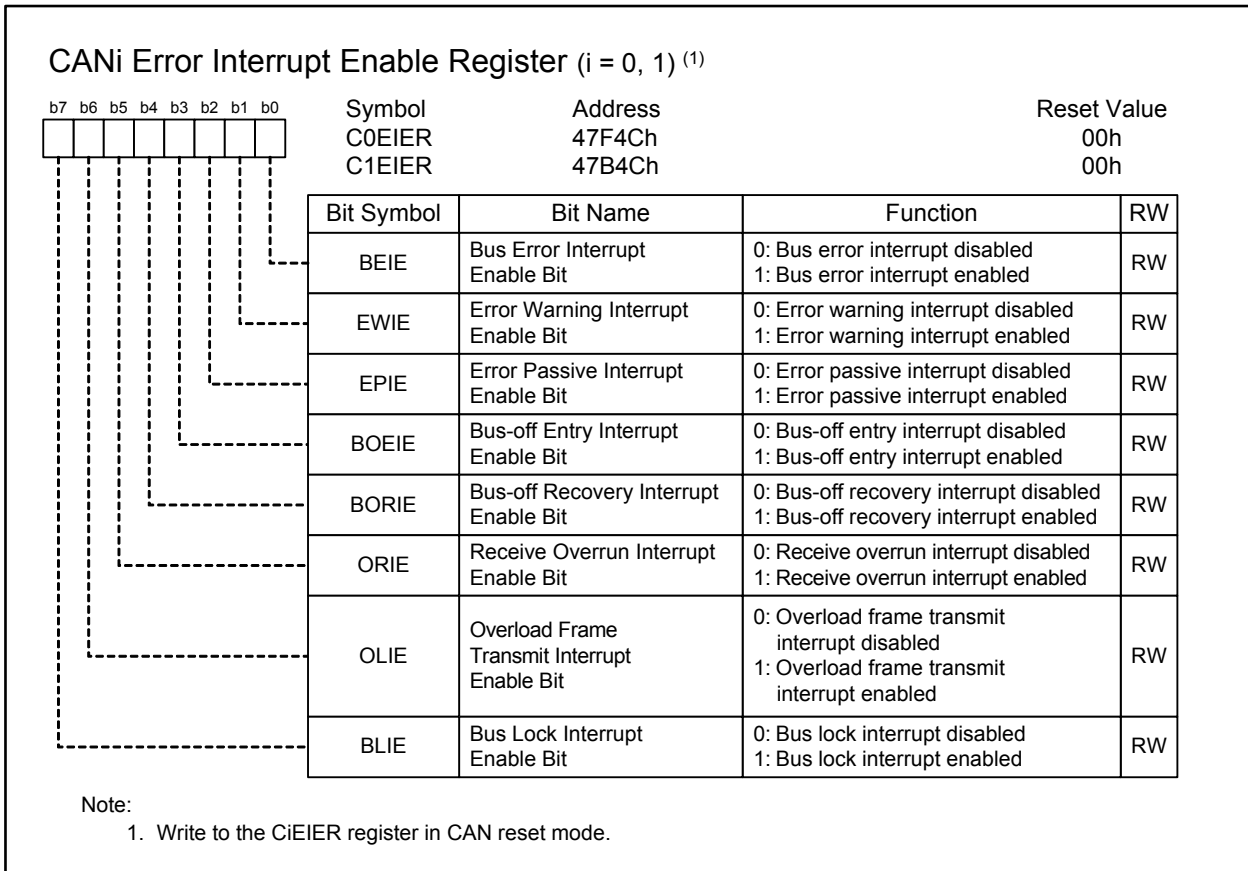
- When there are too many IDs to receive and software filtering time is expected to be shortened.

Figure 24.23 shows the write and read of CiAFSR register.



**Figure 24.23 Write and Read of CiAFSR Register (i = 0, 1; j = 0 to 31)**

### 24.1.19 CAN<sub>i</sub> Error Interrupt Enable Register (CiEIER) (i = 0, 1)



**Figure 24.24 Registers C0EIER and C1EIER**

The CiEIER register is used to set the error interrupt enabled/disabled individually for each error interrupt source in the CiEIFR register.

#### 24.1.19.1 BEIE Bit

When the BEIE bit is 0, no error interrupt request is generated even if the BEIF bit in the CiEIFR register is set to 1 (i = 0, 1).

When the BEIE bit is 1, an error interrupt request is generated if the BEIF bit is set to 1.

#### 24.1.19.2 EWIE Bit

When the EWIE bit is 0, no error interrupt request is generated even if the EWIF bit in the CiEIFR register is set to 1.

When the EWIE bit is 1, an error interrupt request is generated if the EWIF bit is set to 1.

#### 24.1.19.3 EPIE Bit

When the EPIE bit is 0, no error interrupt request is generated even if the EPIF bit in the CiEIFR register is set to 1.

When the EPIE bit is 1, an error interrupt request is generated if the EPIF bit is set to 1.

#### 24.1.19.4 BOEIE Bit

When the BOEIE bit is 0, no error interrupt request is generated even if the BOEIF bit in the CiEIFR register is set to 1.

When the BOEIE bit is 1, an error interrupt request is generated if the BOEIF bit is set to 1.

#### 24.1.19.5 BORIE Bit

When the BORIE bit is 0, an error interrupt request is not generated even if the BORIF bit in the CiEIFR register is set to 1.

When the BORIE bit is 1, an error interrupt request is generated if the BORIF bit is set to 1.

#### 24.1.19.6 ORIE Bit

When the ORIE bit is 0, no error interrupt request is generated even if the ORIF bit in the CiEIFR register is set to 1.

When the ORIE bit is 1, an error interrupt request is generated if the ORIF bit is set to 1.

#### 24.1.19.7 OLIE Bit

When the OLIE bit is 0, no error interrupt request is generated even if the OLIF bit in the CiEIFR register is set to 1.

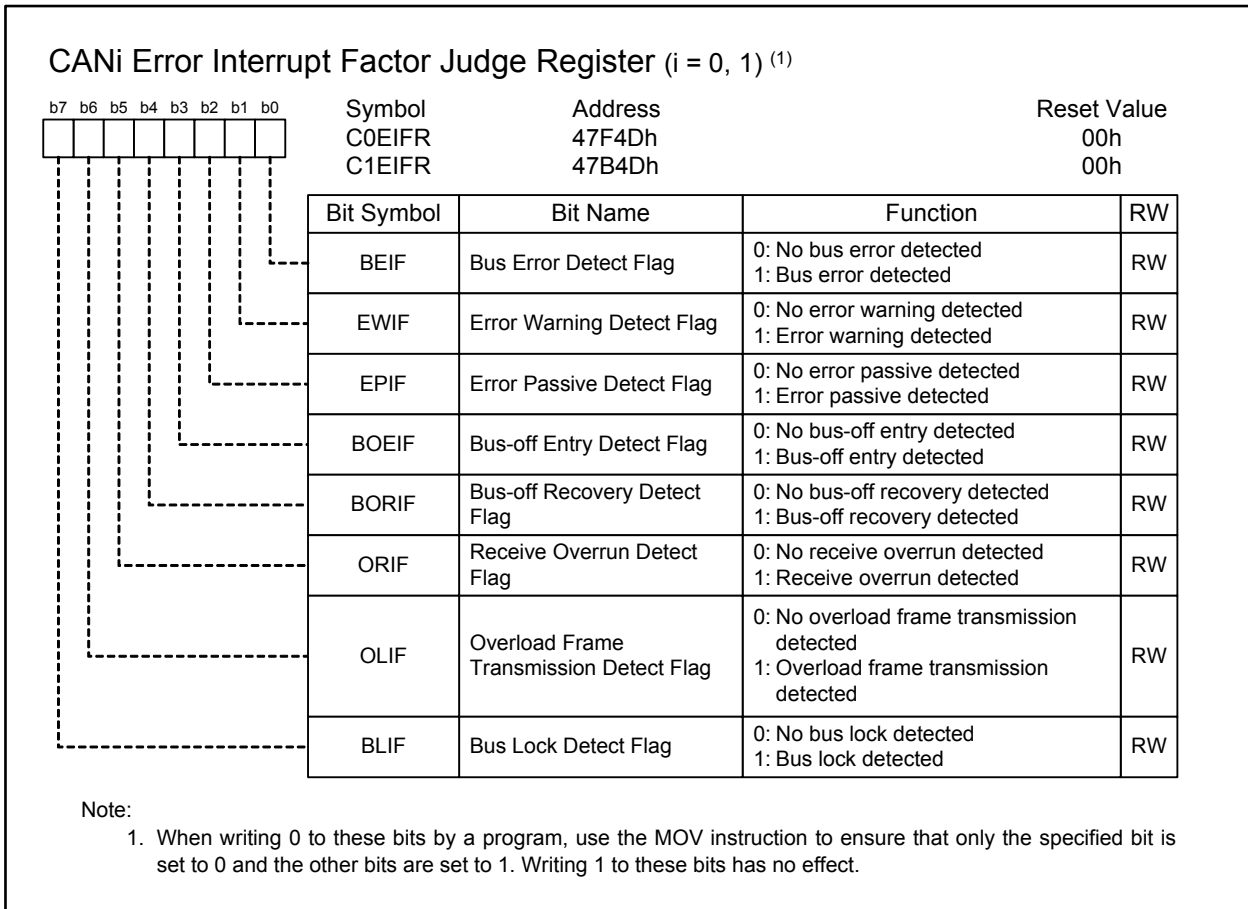
When the OLIE bit is 1, an error interrupt request is generated if the OLIF bit is set to 1.

#### 24.1.19.8 BLIE Bit

When the BLIE bit is 0, no error interrupt request is generated even if the BLIF bit in the CiEIFR register is set to 1.

When the BLIE bit is 1, an error interrupt request is generated if the BLIF bit is set to 1.

### 24.1.20 CAN<sub>i</sub> Error Interrupt Factor Judge Register (CiEIFR) (i = 0, 1)



**Figure 24.25 Registers C0EIFR and C1EIFR**

If an event corresponding to each bit occurs, the corresponding bit in the CiEIFR register is set to 1 regardless of the setting of the CiEIER register.

To set each bit to 0, write 0 by a program. If the set timing occurs simultaneously with the clear timing by the program, the bit becomes 1.

#### 24.1.20.1 BEIF Bit

The BEIF bit becomes 1 when a bus error is detected.

#### 24.1.20.2 EWIF Bit

The EWIF bit becomes 1 when the value of the receive error counter (REC) or transmit error counter (TEC) exceeds 95.

This bit becomes 1 only when the REC or TEC initially exceeds 95. Thus, if 0 is written to the EWIF bit by a program while the REC or TEC remains greater than 95, this bit does not become 1 until the REC and the TEC go below 95 and then exceed 95 again.

### 24.1.20.3 EPIF Bit

The EPIF bit becomes 1 when the CAN error state enters the error-passive state (the REC or TEC value exceeds 127).

This bit becomes 1 only when the REC or TEC initially exceeds 127. Thus, if 0 is written to the EPIF bit by a program while the REC or TEC remains greater than 127, this bit does not become 1 until the REC and the TEC go below 127 and then exceed 127 again.

### 24.1.20.4 BOEIF Bit

The BOEIF bit becomes 1 when the CAN error state enters the bus-off state (the TEC value exceeds 255).

This bit also becomes 1 when the BOM bit in the CiCTRL register is 01b (entry to CAN halt mode automatically at bus-off entry) and the CAN module enters the bus-off state (i = 0, 1).

### 24.1.20.5 BORIF Bit

The BORIF bit becomes 1 when the CAN module recovers from the bus-off state normally by detecting 11 consecutive bits 128 times in the following conditions:

- (1) When the BOM bit in the CiCTRL register is 00b
- (2) When the BOM bit is 10b
- (3) When the BOM bit is 11b

The BORIF bit does not become 1 if the CAN module recovers from the bus-off state in the following conditions:

- (1) When the CANM bit in the CiCTRL register is set to 01b (CAN reset mode)
- (2) When the RBOC bit in the CiCTRL register is set to 1 (forced recovery from bus-off)
- (3) When the BOM bit is 01b
- (4) When the BOM bit is 11b and the CANM bit is set to 10b (CAN halt mode) before normal recovery occurs

Table 24.8 lists the operation of bits BOEIF and BORIF according to BOM bit setting value.

**Table 24.8 Operation of Bits BOEIF and BORIF According to BOM Bit Setting Value**

BOM Bit	BOEIF Bit	BORIF Bit
00b	Becomes 1 when entering the bus-off state	Becomes 1 when recovering from the bus-off state
01b		Does not become 1
10b		Becomes 1 when recovering from the bus-off state
11b		Becomes 1 if normal bus-off recovery occurs before the CANM bit is set to 10b (CAN halt mode)

### 24.1.20.6 ORIF Bit

The ORIF bit becomes 1 when a receive overrun occurs.

This bit does not become 1 in overwrite mode. In overwrite mode, a reception complete interrupt request is generated if an overwrite condition occurs, thus this bit does not become 1.

In normal mailbox mode, if an overrun occurs in any mailbox from [0] to [31] in overrun mode, this bit is set to 1.

In FIFO mailbox mode, if an overrun occurs in any mailbox from [0] to [23] or the receive FIFO in overrun mode, this bit becomes 1.

**24.1.20.7 OLIF Bit**

The OLIF bit becomes 1 if the transmitting condition of an overload frame is detected when the CAN module performs transmission or reception.

**24.1.20.8 BLIF Bit**

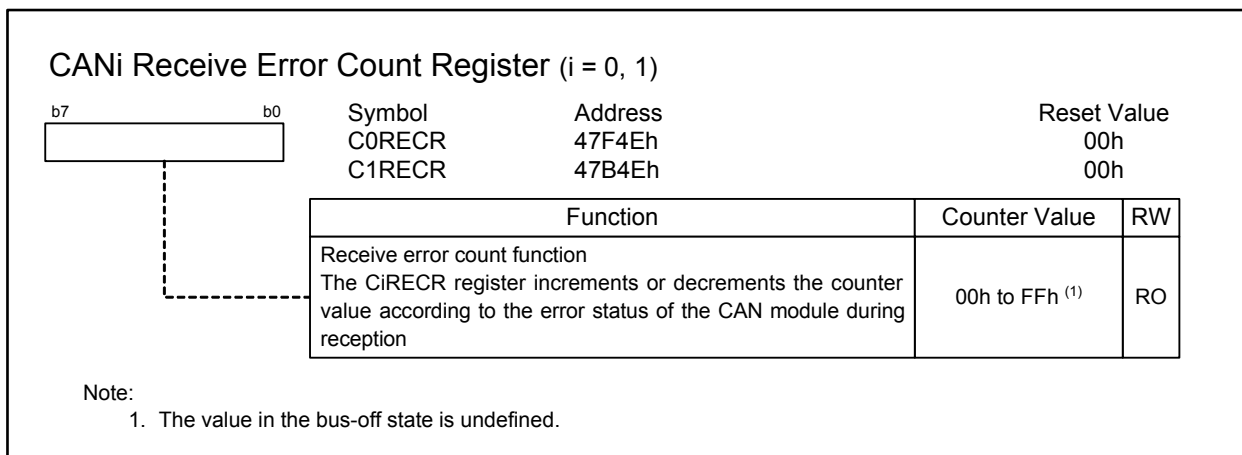
The BLIF bit becomes 1 if 32 consecutive dominant bits are detected on the CAN bus while the CAN module is in CAN operation mode.

After the BLIF bit becomes 1, 32 consecutive dominant bits are detected again under either of the following conditions:

- After this bit is set to 0 from 1, recessive bits are detected.
- After this bit is set to 0 from 1, the CAN module enters CAN reset mode or CAN halt mode and then enters CAN operation mode again.



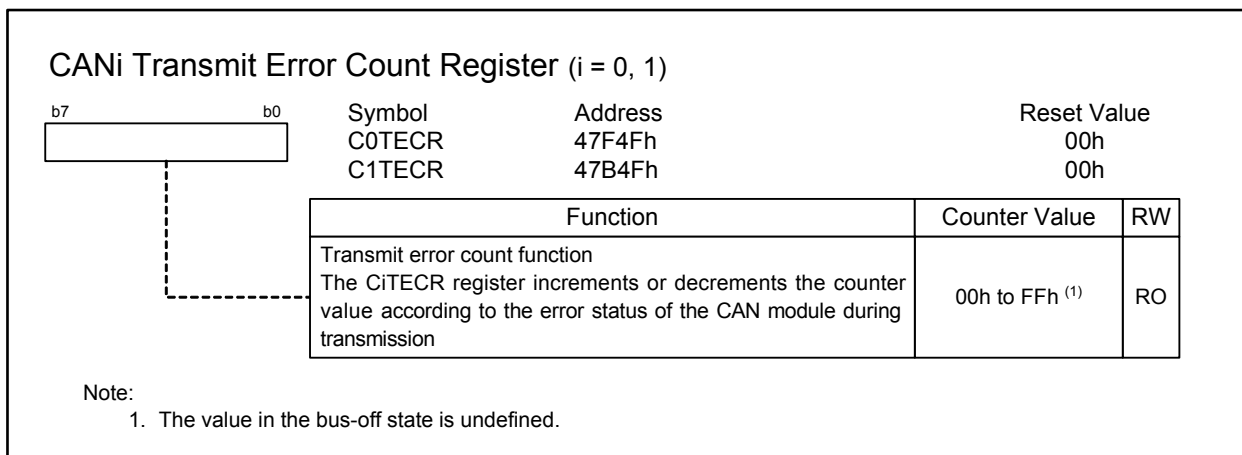
### 24.1.21 CAN<sub>i</sub> Receive Error Count Register (CiRECR) (i = 0, 1)



**Figure 24.26 Registers C0RECR and C1RECR**

The CiRECR register indicates the value of the receive error counter. Refer to the CAN Specification (ISO 11898-1) for the increment/decrement conditions of the receive error counter.

### 24.1.22 CANi Transmit Error Count Register (CiTECR) (i = 0, 1)

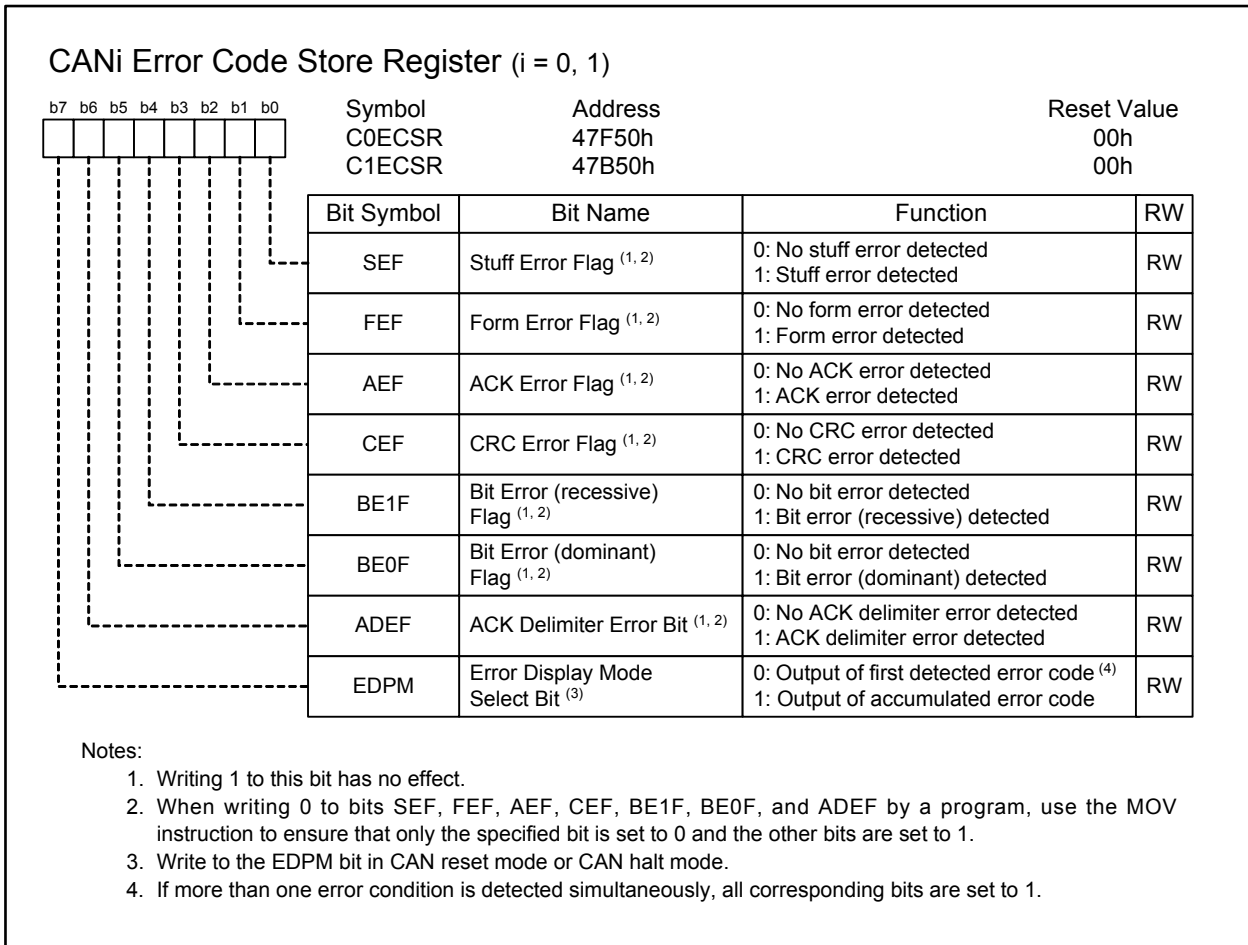


**Figure 24.27 Registers C0TECR and C1TECR**

The CiTECR register indicates the value of the TEC error counter.

Refer to the CAN Specification (ISO 11898-1) for the increment/decrement conditions of the transmit error counter.

### 24.1.23 CANi Error Code Store Register (CiECSR) (i = 0, 1)



**Figure 24.28 Registers C0ECSR and C1ECSR**

The CiECSR register can be used to monitor whether an error has occurred on the CAN bus. Refer to the CAN Specification (ISO 11898-1) to check the generation conditions of each error.

To set each bit except the EDPM bit to 0, write 0 by a program. If the timing at which each bit is set to 1 and the timing at which is written by a program are the same, the relevant bit is set to 1.

#### 24.1.23.1 SEF Bit

The SEF bit becomes 1 when a stuff error is detected.

#### 24.1.23.2 FEF Bit

The FEF bit becomes 1 when a form error is detected.

#### 24.1.23.3 AEF Bit

The AEF bit becomes 1 when an ACK error is detected.

**24.1.23.4 CEF Bit**

The CEF bit becomes 1 when a CRC error is detected.

**24.1.23.5 BE1F Bit**

The BE1F bit becomes 1 when a recessive bit error is detected.

**24.1.23.6 BE0F Bit**

The BE0F bit becomes 1 when a dominant bit error is detected.

**24.1.23.7 ADEF Bit**

The ADEF bit becomes 1 when a form error is detected with the ACK delimiter during transmission.

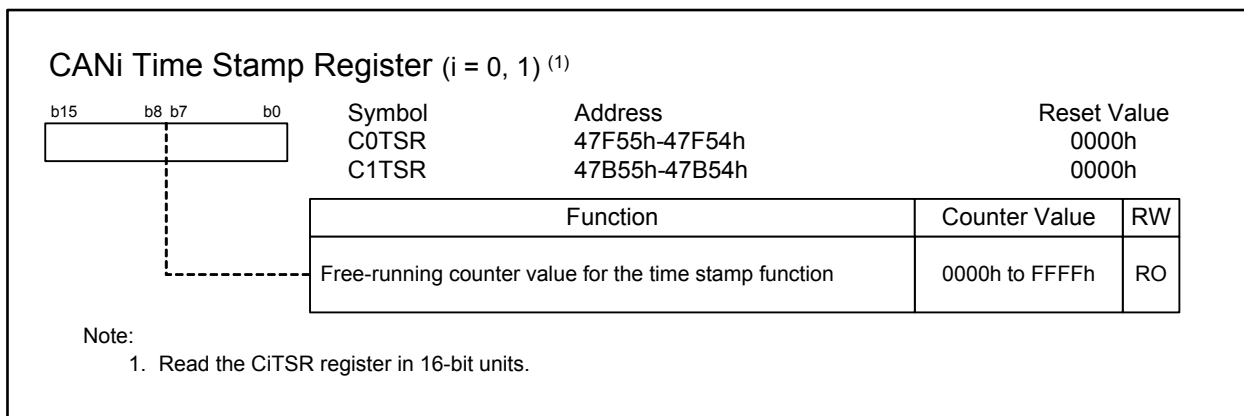
**24.1.23.8 EDPM Bit**

The EDPM bit selects the output mode of the CiECSR register ( $i = 0, 1$ ).

When this bit is set to 0, the CiECSR register outputs the first error code.

When this bit is set to 1, the CiECSR register outputs the accumulated error code.

### 24.1.24 CANi Time Stamp Register (CiTSR) (i = 0, 1)



**Figure 24.29 Registers C0TSR and C1TSR**

When the CiTSR register is read, the value of the time stamp counter (16-bit free-running counter) at that moment is read.

The value of the time stamp counter reference clock is a multiple of 1 bit time, as configured by the TSPS bit in the CiCTLR register.

The time stamp counter stops in CAN sleep mode and CAN halt mode, and is initialized in CAN reset mode.

The time stamp counter value is stored to TSL and TSH in the CiMBj register when a received message is stored in a receive mailbox (j = 0 to 31).

### 24.1.25 CAN<sub>i</sub> Test Control Register (CiTCR) (i = 0, 1)

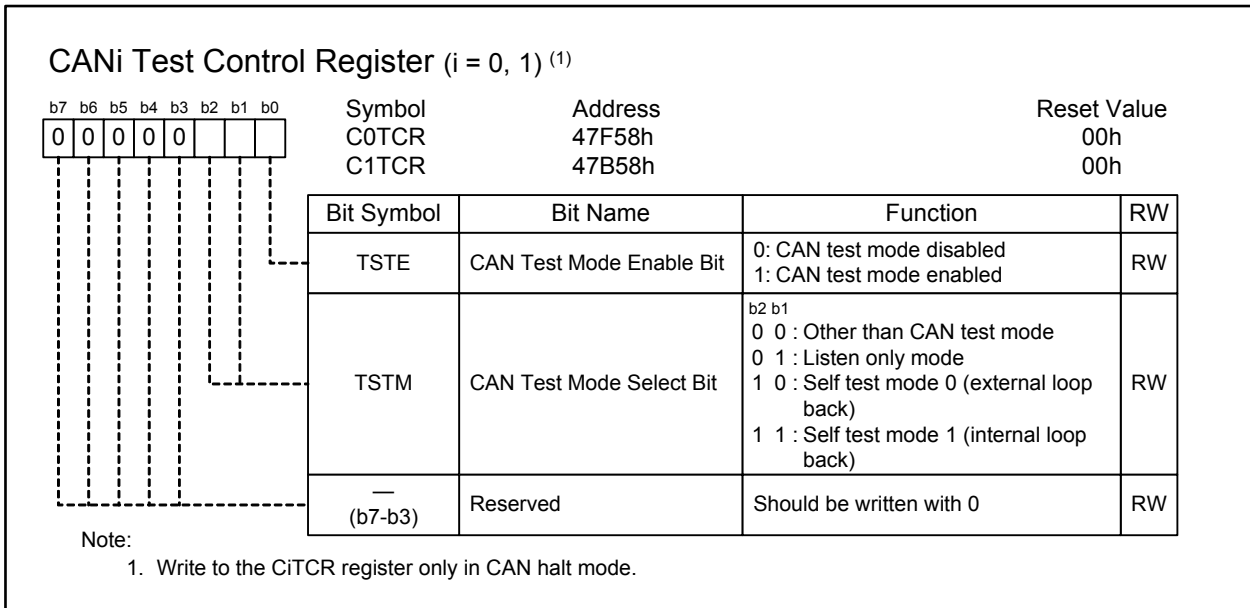


Figure 24.30 Registers C0TCR and C1TCR

#### 24.1.25.1 TSTE Bit

When the TSTE bit is set to 0, CAN test mode is disabled.  
When this bit is set to 1, CAN test mode is enabled.

#### 24.1.25.2 TSTM Bit

The TSTM bit selects the CAN test mode.  
The details of each CAN test mode are described below.

### 24.1.25.3 Listen Only Mode

The ISO 11898-1 recommends an optional bus monitoring mode. In listen only mode, the CAN node is able to receive valid data frames and valid remote frames. It sends only recessive bits on the CAN bus and the protocol controller is not required to send the ACK bit, overload flag, or active error flag.

Listen only mode can be used for baud rate detection.

Do not request transmission from any mailboxes in this mode.

Figure 24.31 shows the connection when listen only mode is selected.

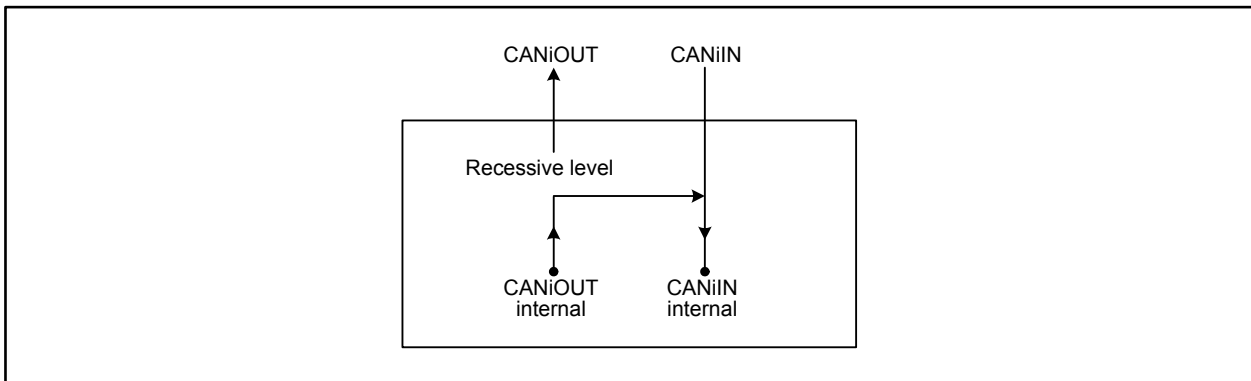


Figure 24.31 Connection when Listen Only Mode is Selected ( $i = 0, 1$ )

### 24.1.25.4 Self Test Mode 0 (External Loop Back)

Self test mode 0 is provided for CAN transceiver tests.

In this mode, the protocol controller treats its own transmitted messages as messages received via the CAN transceiver and stores them into a receive mailbox. To be independent from external stimulation, the protocol controller generates the ACK bit.

Connect the CANiOUT/CANiIN pins to the transceiver ( $i = 0, 1$ ).

Figure 24.32 shows the connection when self test mode 0 is selected.

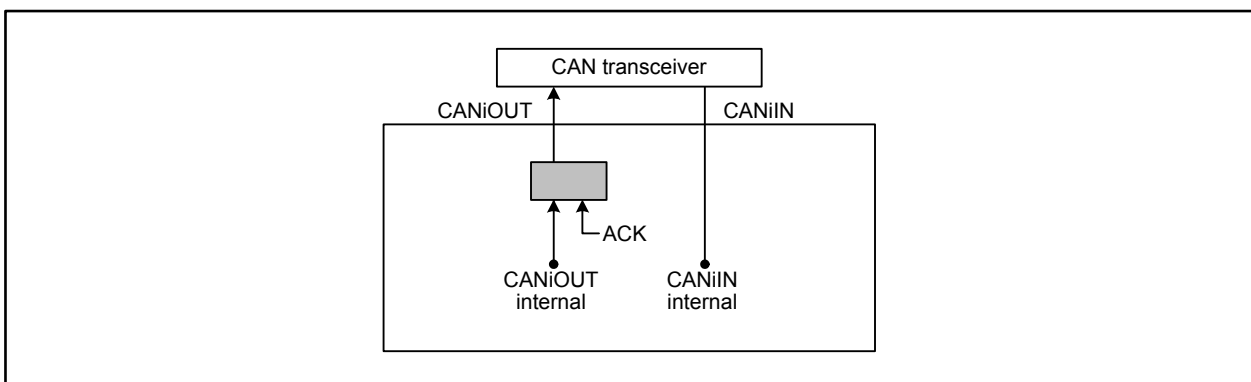


Figure 24.32 Connection when Self Test Mode 0 is Selected ( $i = 0, 1$ )

### 24.1.25.5 Self Test Mode 1 (Internal Loop Back)

Self test mode 1 is provided for self test functions.

In this mode, the protocol controller treats its transmitted messages as received messages and stores them into a receive mailbox. To be independent from external stimulation, the protocol controller generates the ACK bit.

In self test mode 1, the protocol controller performs an internal feedback from the internal CANiOUT pin to the internal CANiIN pin ( $i = 0, 1$ ). The input value of the external CANiIN pin is ignored. The external CANiOUT pin outputs only recessive bits. The CANiOUT/CANiIN pins do not need to be connected to the CAN bus or any external device.

Figure 24.33 shows the connection when self test mode 1 is selected.

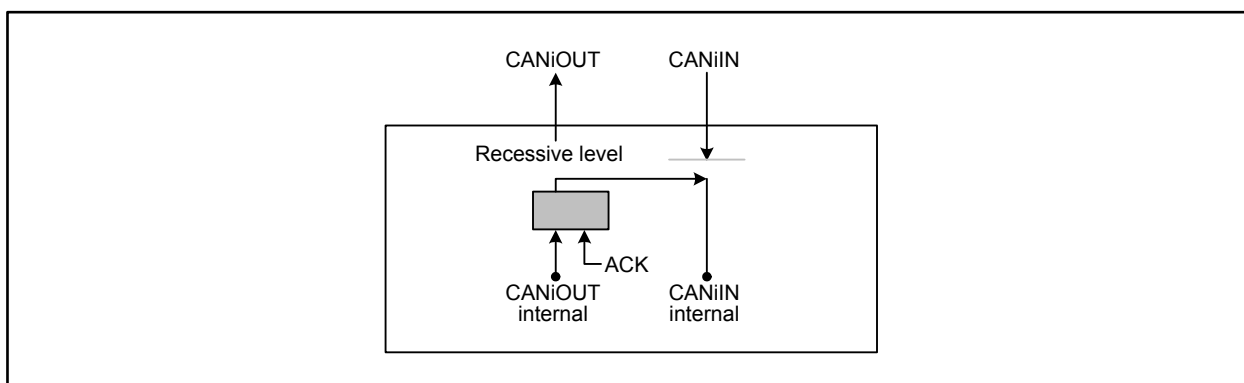


Figure 24.33 Connection when Self Test Mode 1 is Selected ( $i = 0, 1$ )

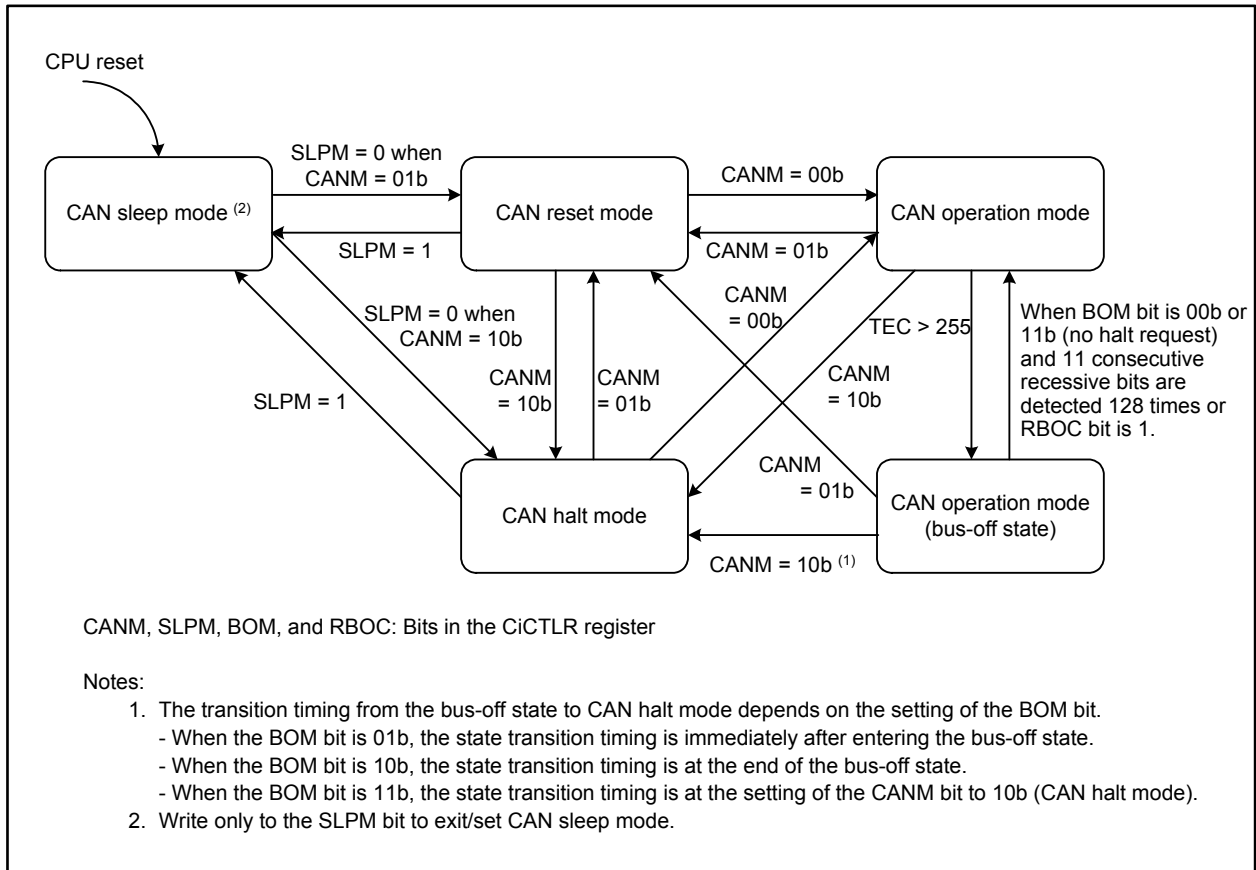


## 24.2 Operating Modes

The CAN module has the following four operating modes:

- CAN reset mode
- CAN halt mode
- CAN operation mode
- CAN sleep mode

Figure 24.34 shows the transition between CAN operating modes.



**Figure 24.34 Transition between CAN Operating Modes (i = 0, 1)**

### 24.2.1 CAN Reset Mode

CAN reset mode is provided for CAN communication configuration.

When the CANM bit in the CiCTLR register is set to 01b, the CAN module enters CAN reset (i = 0, 1). Then the RSTST bit in the CiSTR register becomes 1. Do not change the CANM bit until the RSTST bit becomes 1.

Configure the CiBCR register before exiting CAN reset mode and entering any other mode.

The following registers are initialized to their reset values after entering CAN reset mode and their initialized values are retained during CAN reset mode:

- CiMCTLj register (j = 0 to 31)
- CiSTR register (except bits SLPST and TFST)
- CiEIFR register
- CiRECR register
- CiTECR register
- CiTSR register
- CiMSSR register
- CiMSMR register
- CiRFCR register
- CiTFRCR register
- CiTCR register
- CiECSR register (except EDPM bit)

The previous values of the following registers are retained after entering CAN reset mode:

- CiCLKR register
- CiCTLR register
- CiSTR register (bits SLPST and TFST)
- CiMIER register
- CiEIER register
- CiBCR register
- CiCSSR register
- CiECSR register (EDPM bit only)
- CiMBj register
- Registers CiMKR0 to CiMKR7
- Registers CiFIDCR0 and CiFIDCR1
- CiMKIVLR register
- CiAFSR register
- CiRFPCR register
- CiTFPCR register

### 24.2.2 CAN Halt Mode

CAN halt mode is used for mailbox configuration and test mode setting.

When the CANM bit in the CiCTLR register is set to 10b, CAN halt mode is selected ( $i = 0, 1$ ). Then the HLTST bit in the CiSTR register becomes 1. Do not change the CANM bit until the HLTST bit becomes 1.

Refer to Table 24.9 “Operation in CAN Reset Mode and CAN Halt Mode” regarding the state transition conditions when transmitting or receiving.

All registers except bits RSTST, HLTST, and SLPST in the CiSTR register remain unchanged when the CAN module enters CAN halt mode.

Do not change registers CiCLKR, CiCTLR (except bits CANM and SLPM), and CiEIER in CAN halt mode. The CiBCR register can be changed in CAN halt mode only when listen only mode is selected to use with automatic bit rate detection.

**Table 24.9 Operation in CAN Reset Mode and CAN Halt Mode**

Mode	Receiver	Transmitter	Bus-off
CAN reset mode	CAN module enters CAN reset mode without waiting for the end of message reception	CAN module enters CAN reset mode after waiting for the end of message transmission (1, 4)	CAN module enters CAN reset mode without waiting for the end of bus-off recovery
CAN halt mode	CAN module enters CAN halt mode after waiting for the end of message reception (2, 3)	CAN module enters CAN halt mode after waiting for the end of message transmission (1, 4)	<ul style="list-style-type: none"> <li>- When the BOM bit is 00b A halt request from a program will be acknowledged only after bus-off recovery</li> <li>- When the BOM bit is 01b CAN module automatically enters CAN halt mode without waiting for the end of bus-off recovery (regardless of a halt request from a program)</li> <li>- When the BOM bit is 10b CAN module automatically enters CAN halt mode after waiting for the end of bus-off recovery (regardless of a halt request from a program)</li> <li>- When the BOM bit is 11b CAN module enters CAN halt mode (without waiting for the end of bus-off recovery) if a halt is requested by a program during bus-off</li> </ul>

BOM bit: Bit in the CiCTLR register ( $i = 0, 1$ )

Notes:

1. If several messages are requested to be transmitted, mode transition occurs after the completion of the first message transmission. When CAN reset mode is being requested during suspend transmission, mode transition occurs when the bus is idle, the next transmission ends, or the CAN module becomes a receiver.
2. If the CAN bus is locked at the dominant level, the program can detect this state by monitoring the BLIF bit in the CiEIFR register.
3. If a CAN bus error occurs during reception after CAN halt mode is requested, the CAN mode transits to CAN halt mode.
4. If a CAN bus error or arbitration lost occurs during transmission after CAN reset mode or CAN halt mode is requested, the CAN mode transits to the requested CAN mode.

### 24.2.3 CAN Sleep Mode

CAN sleep mode is used for reducing current consumption by stopping the clock supply to the CAN module. After a MCU reset, the CAN module starts from CAN sleep mode.

When the SLPM bit in the CiCTLR register is set to 1, the CAN module enters CAN sleep mode (i = 0, 1). Then the SLPST bit in the CiSTR register becomes 1. Do not change the value of the SLPM bit until the SLPST bit becomes 1. Other registers remain unchanged when the MCU enters CAN sleep mode.

Write to the SLPM bit in CAN reset mode and CAN halt mode. Do not change any other registers (except the SLPM bit) during CAN sleep mode. Read operations are still allowed.

When the SLPM bit is set to 0, the CAN module is released from CAN sleep mode. When the CAN module exits CAN sleep mode, the other registers remain unchanged.

### 24.2.4 CAN Operation Mode (Excluding Bus-off State)

CAN operation mode is used for CAN communication.

When the CANM bit in the CiCTLR register is set to 00b, the CAN module enters CAN operation mode ( $i = 0, 1$ ).

Then bits RSTST and HLTST in the CiSTR register become 0. Do not change the value of the CANM bit until these bits become 0.

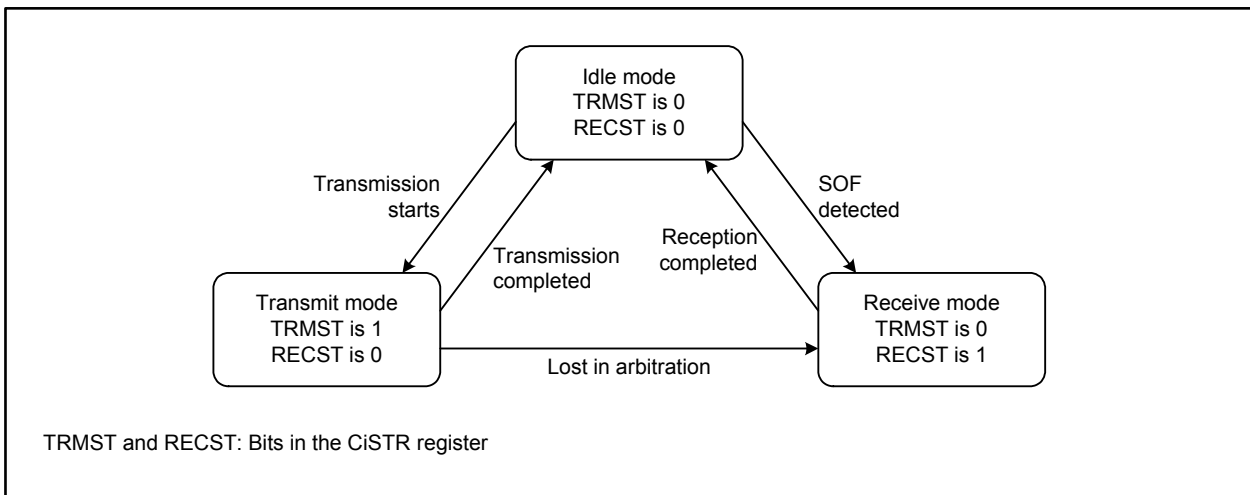
If 11 consecutive recessive bits are detected after entering CAN operation mode, the CAN module is in the following states:

- The CAN module becomes an active node on the network that enables transmission and reception of CAN messages.
- Error monitoring of the CAN bus, such as receive and transmit error counters, is performed.

During CAN operation mode, the CAN module may be in one of the following three submodes, depending on the status of the CAN bus:

- Idle mode: Transmission or reception is not being performed.
- Receive mode: A CAN message sent by another node is being received.
- Transmit mode: A CAN message is being transmitted. The CAN module may receive its own message simultaneously when self test mode 0 (TSTM bits in the CiTCR register are 10b) or self test mode 1 (TSTM bits are 11b) is selected.

Figure 24.35 shows the submode in CAN operation mode.



**Figure 24.35 Submode in CAN Operation Mode ( $i = 0, 1$ )**

### 24.2.5 CAN Operation Mode (Bus-off State)

The CAN module enters the bus-off state according to the increment/decrement rules for the transmit/error counters in the CAN Specifications.

The following cases apply when recovering from the bus-off state. When the CAN module is in the bus-off state, the values of the associated registers, except registers CiSTR, CiEIFR, CiRECR, CiTECR, and CiTSR, remain unchanged ( $i = 0, 1$ ).

- (1) When the BOM bit in the CiCTLR register is 00b (normal mode)

The CAN module enters the error-active state after it has completed the recovery from the bus-off state and CAN communication is enabled. The BORIF bit in the CiEIFR register becomes 1 (bus-off recovery detected) at this time.

- (2) When the RBOC bit in the CiCTLR register is set to 1 (forced recovery from bus-off)

The CAN module enters the error-active state when it is in the bus-off state and the RBOC bit is set to 1. CAN communication is enabled again after 11 consecutive recessive bits are detected. The BORIF bit does not become 1 at this time.

- (3) When the BOM bit is 01b (entry to CAN halt mode automatically at bus-off entry)

The CAN module enters CAN halt mode when it reaches the bus-off state. The BORIF bit does not become 1 at this time.

- (4) When the BOM bit is 10b (entry to CAN halt mode automatically at bus-off end)

The CAN module enters CAN halt mode when it has completed the recovery from bus-off. The BORIF bit becomes 1 at this time.

- (5) When the BOM bit is 11b (entry to CAN halt mode by a program) and the CANM bit in the CiCTLR register is set to 10b (CAN halt mode) during the bus-off state

The CAN module enters CAN halt mode when it is in the bus-off state and the CANM bit is set to 10b (CAN halt mode). The BORIF bit does not become 1 at this time.

If the CANM bit is not set to 10b during bus-off, the same behavior as (1) applies.

## 24.3 CAN Communication Speed Configuration

The following description explains about the CAN communication speed configuration.

### 24.3.1 CAN Clock Configuration

This group has a CAN clock selector.

The CAN clock can be configured by setting the CCLKS bit in the CiCLKR register and the BRP bit in the CiBCR register ( $i = 0, 1$ ).

Figure 24.36 shows the block diagram of CAN clock generator.

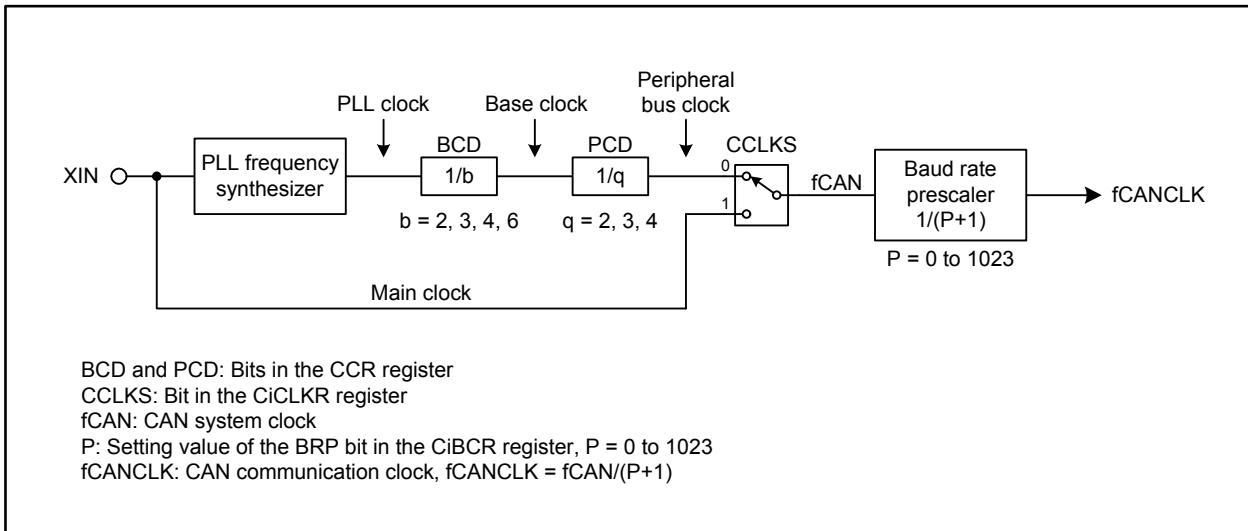


Figure 24.36 Block Diagram of the CAN Clock Generator ( $i = 0, 1$ )

### 24.3.2 Bit Timing Configuration

The bit time is a single bit time for transmitting/receiving a message and consists of the three segments in the figure below.

Figure 24.37 shows the bit timing.

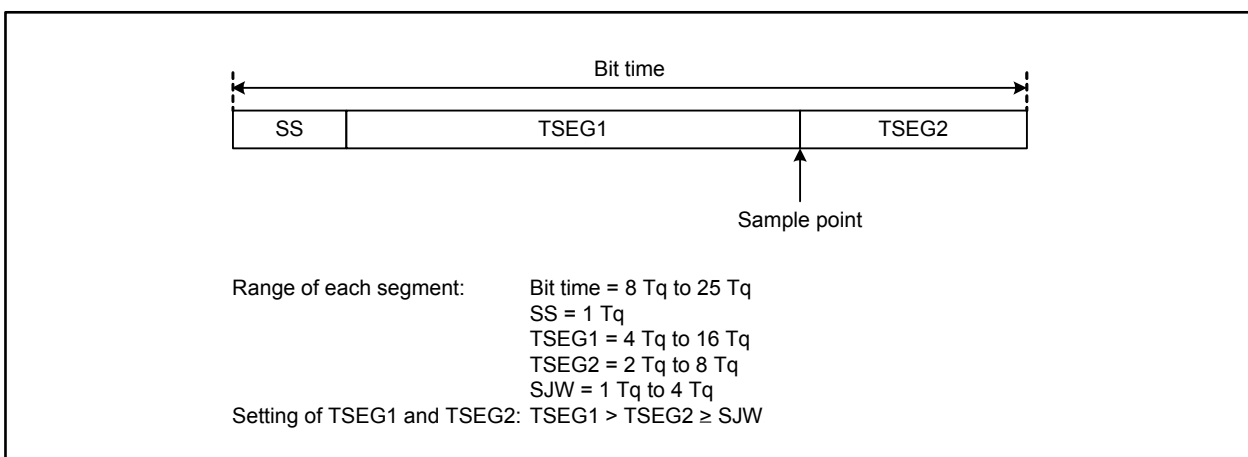


Figure 24.37 Bit Timing

### 24.3.3 Bit rate

The bit rate depends on the CAN clock ( $f_{CAN}$ ), the divisor of the baud rate prescaler, and the number of  $T_q$  of 1 bit time.

$$\text{Bit rate[bps]} = \frac{f_{CAN}}{\text{Baud rate prescaler division value}^{(1)} \times \text{number of } T_q \text{ of 1 bit time}} = \frac{f_{CANCLK}}{\text{Number of } T_q \text{ of 1 bit time}}$$

Note:

1. Divisor of the baud rate prescaler =  $P + 1$  ( $P = 0$  to 1023)  
P: Setting value of the BRP bit in the CiBCR register ( $i = 0, 1$ )

Table 24.10 lists bit rate examples.

**Table 24.10 Bit Rate Examples**

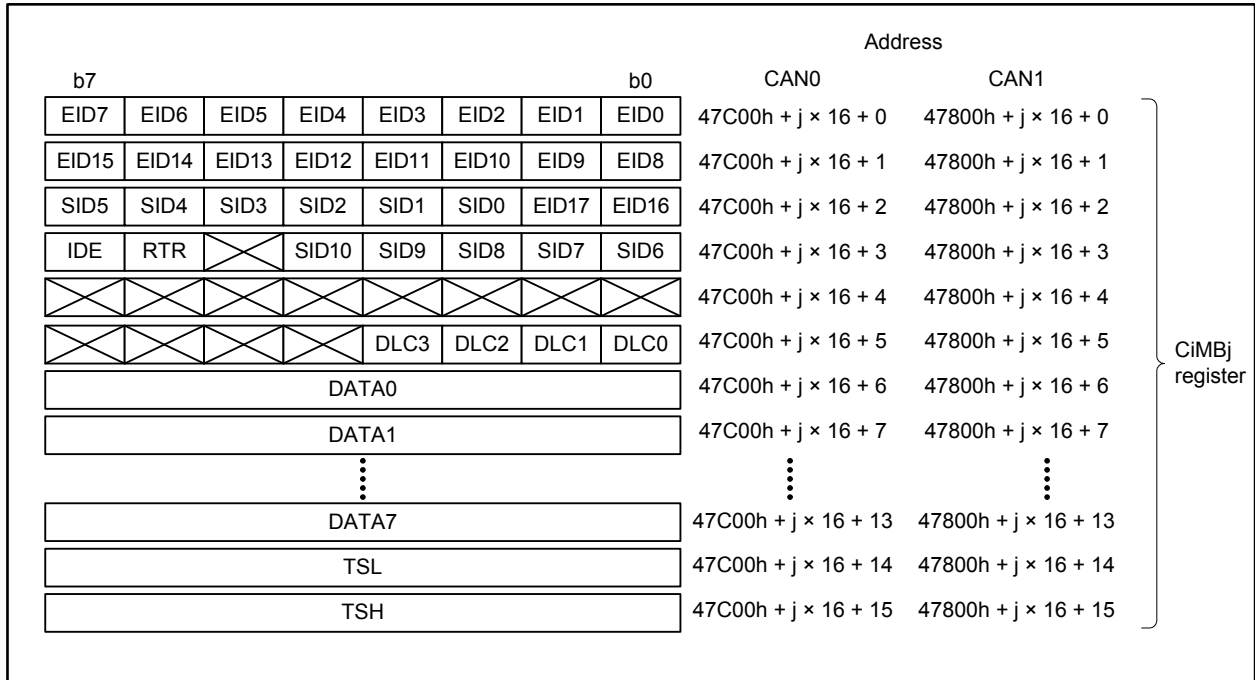
fCAN	32 MHz		24 MHz		20 MHz		16 MHz		8 MHz	
	No. of $T_q$	P+1	No. of $T_q$	P+1	No. of $T_q$	P+1	No. of $T_q$	P+1	No. of $T_q$	P+1
1 Mbps	8 $T_q$	4	8 $T_q$	3	10 $T_q$	2	8 $T_q$	2	8 $T_q$	1
	16 $T_q$	2			20 $T_q$	1	16 $T_q$	1		
500 kbps	8 $T_q$	8	8 $T_q$	6	10 $T_q$	4	8 $T_q$	4	8 $T_q$	2
	16 $T_q$	4	16 $T_q$	3	20 $T_q$	2	16 $T_q$	2	16 $T_q$	1
250 kbps	8 $T_q$	16	8 $T_q$	12	10 $T_q$	8	8 $T_q$	8	8 $T_q$	4
	16 $T_q$	8	16 $T_q$	6	20 $T_q$	4	16 $T_q$	4	16 $T_q$	2
83.3 kbps	8 $T_q$	48	8 $T_q$	36	8 $T_q$	30	8 $T_q$	24	8 $T_q$	12
	16 $T_q$	24	16 $T_q$	18	10 $T_q$	24	16 $T_q$	12	16 $T_q$	6
					16 $T_q$	15				
					20 $T_q$	12				
33.3 kbps	8 $T_q$	120	8 $T_q$	90	8 $T_q$	75	8 $T_q$	60	8 $T_q$	30
	10 $T_q$	96	10 $T_q$	72	10 $T_q$	60	10 $T_q$	48	10 $T_q$	24
	16 $T_q$	60	16 $T_q$	45	20 $T_q$	30	16 $T_q$	30	16 $T_q$	15
	20 $T_q$	48	20 $T_q$	36			20 $T_q$	24	20 $T_q$	12



## 24.4 Mailbox and Mask Register Structure

There are 32 mailboxes with the same structure.

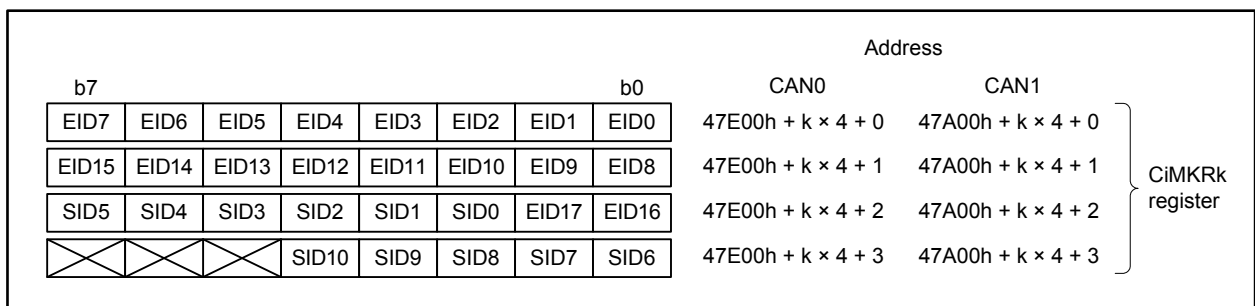
Figure 24.38 shows the structure of registers C0MBj to C1MBj (j = 0 to 31).



**Figure 24.38 Structure of Registers C0MBj to C1MBj (i = 0, 1; j = 0 to 31)**

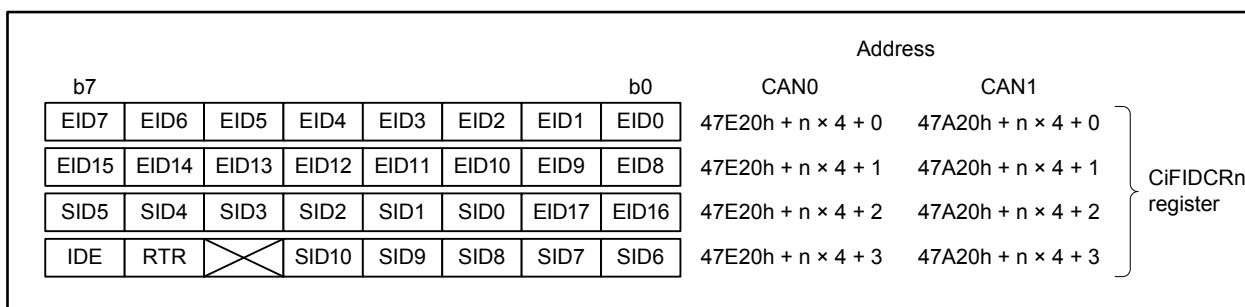
There are eight mask registers with the same structure.

Figure 24.39 shows the structure of registers C0MKRk to C1MKRk (k = 0 to 7).



**Figure 24.39 Structure of Registers C0MKRk to C1MKRk (i = 0, 1; k = 0 to 7)**

There are two FIFO received ID compare registers with the same structure.  
Figure 24.40 shows the structure of registers C0FIDCRn to C1FIDCRn (n = 0, 1).



**Figure 24.40 Structure of Registers C0FIDCRn to C1FIDCRn (i = 0, 1; n = 0, 1)**

## 24.5 Acceptance Filtering and Masking Function

Acceptance filtering allows the user to receive messages with a specified range of multiple IDs for mailboxes.

Registers CiMKR0 to CiMKR7 can perform masking of the standard ID and the extended ID of 29 bits ( $i = 0, 1$ ).

- The CiMKR0 register corresponds to mailboxes [0] to [3].
- The CiMKR1 register corresponds to mailboxes [4] to [7].
- The CiMKR2 register corresponds to mailboxes [8] to [11].
- The CiMKR3 register corresponds to mailboxes [12] to [15].
- The CiMKR4 register corresponds to mailboxes [16] to [19].
- The CiMKR5 register corresponds to mailboxes [20] to [23].
- The CiMKR6 register corresponds to mailboxes [24] to [27] in normal mailbox mode, and receive FIFO mailboxes [28] to [31] in FIFO mailbox mode.
- The CiMKR7 register corresponds to mailboxes [28] to [31] in normal mailbox mode, and receive FIFO mailboxes [28] to [31] in FIFO mailbox mode.

The CiMKIVLR register disables acceptance filtering individually for each mailbox.

The IDE bit in the CiMB $j$  register is enabled when the IDFM bit in the CiCTRL register is 10b (mixed ID mode) ( $j = 0$  to 31).

The RTR bit in the CiMB $j$  register selects a data frame or a remote frame.

In FIFO mailbox mode, normal mailboxes (mailboxes [0] to [23]) use the single corresponding register among registers CiMKR0 to CiMKR5 for acceptance filtering. Receive FIFO mailboxes (mailboxes [28] to [31]) use two registers CiMKR6 and CiMKR7 for acceptance filtering.

Also, the receive FIFO uses registers CiFIDCR0 and CiFIDCR1 for ID comparison. Bits EID, SID, RTR, and IDE in registers CiMB28 to CiMB31 for the receive FIFO are disabled. As acceptance filtering depends on the result of two ID-mask sets, two ranges of IDs can be received into the receive FIFO.

The CiMKIVLR register is disabled for the receive FIFO.

If both the settings for standard ID and extended ID are set in the IDE bits in registers CiFIDCR0 and CiFIDCR1 individually, both ID formats are received.

If both setting of data frame and remote frame are set in the RTR bits in registers CiFIDCR0 and CiFIDCR1 individually, both the data and remote frames are received.

When a combination of two ranges of IDs is not necessary, set the same mask value and the same ID into both of the FIFO ID/mask register sets.

Figure 24.41 shows the mask registers and their corresponding mailboxes, and Figure 24.42 shows acceptance filtering.

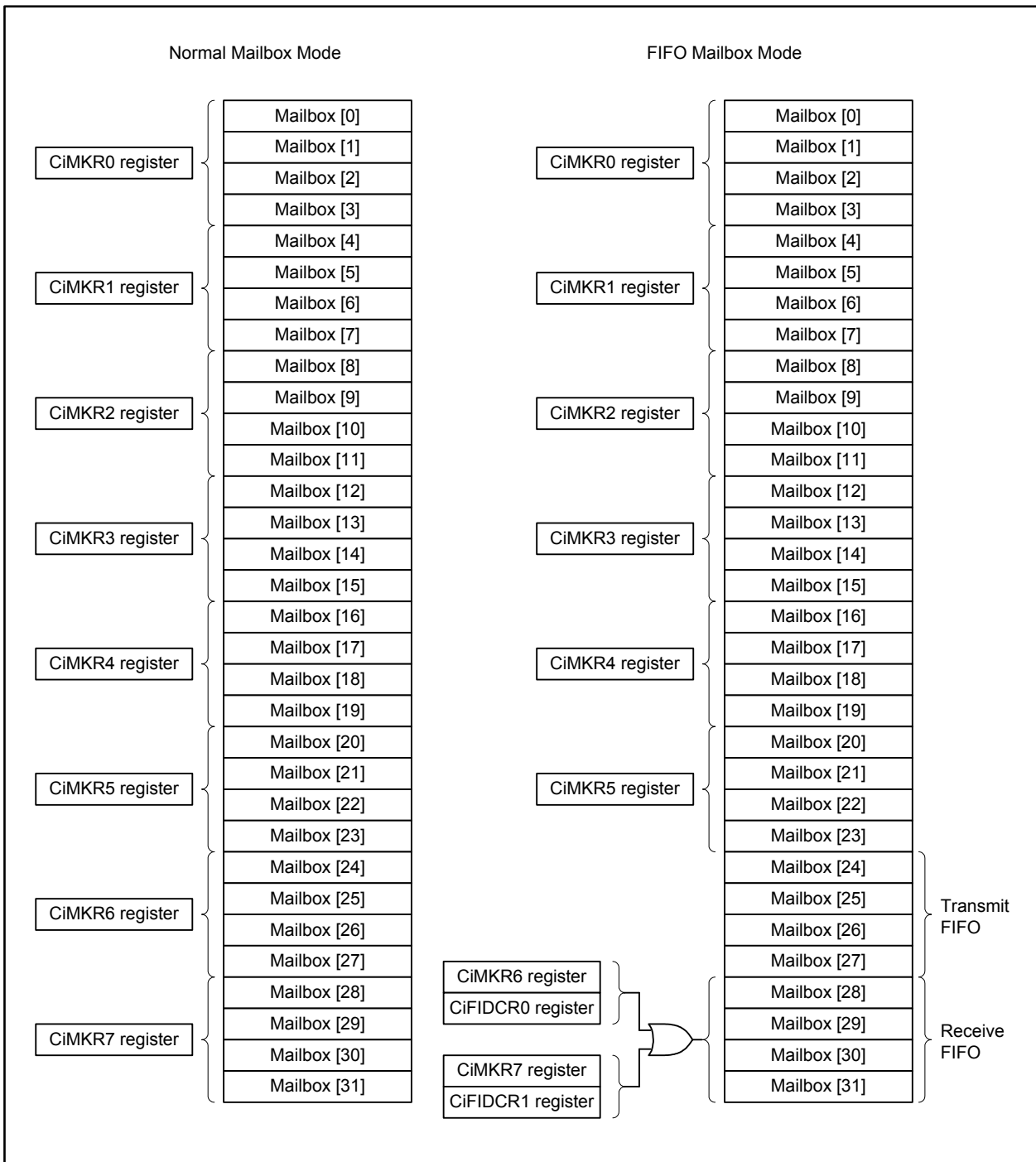
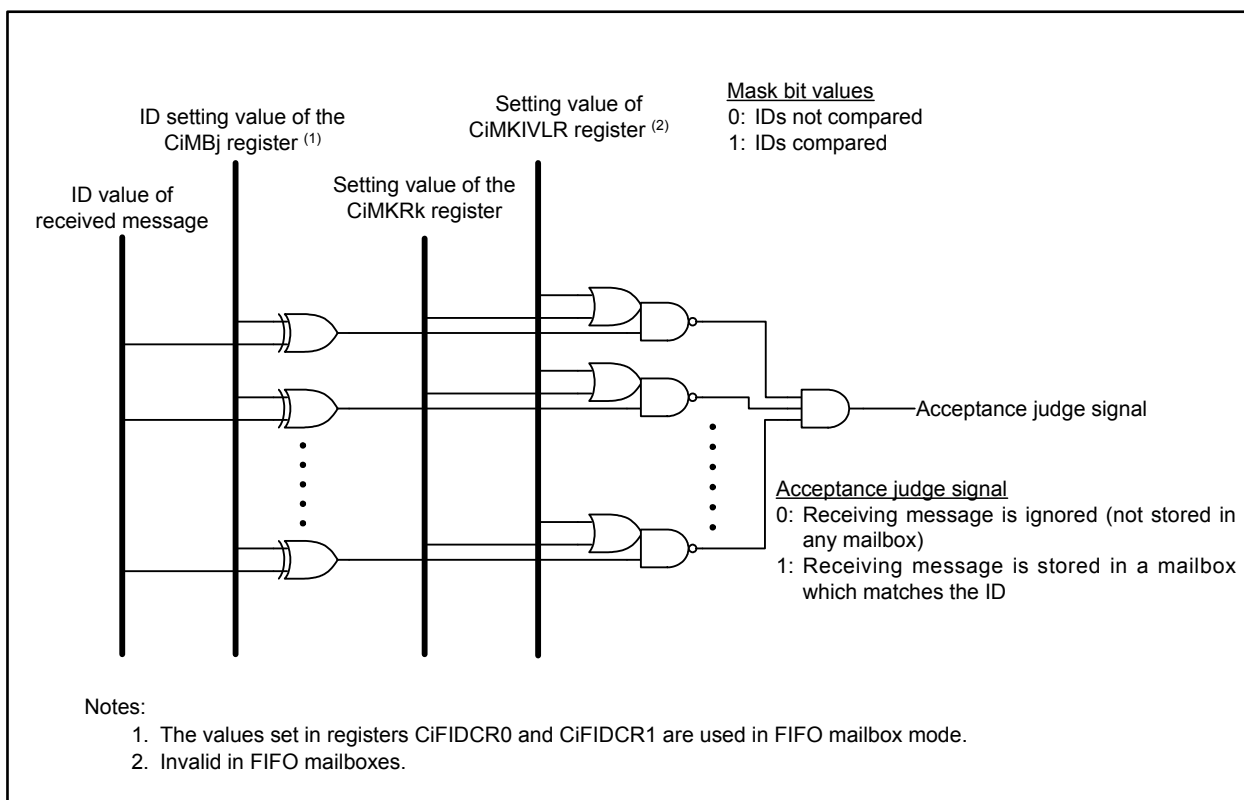


Figure 24.41 Mask Registers and Their Corresponding Mailboxes (i = 0, 1)



**Figure 24.42 Acceptance Filtering ( $i = 0, 1; j = 0$  to 31;  $k = 0$  to 7)**

## 24.6 Reception and Transmission

Table 24.11 lists the CAN communication mode configuration.

**Table 24.11 Configuration for CAN Reception Mode and Transmission Mode**

TRMREQ	RECREQ	ONESHOT	Communication Mode of Mailbox
0	0	0	Mailbox disabled or transmission being aborted
0	0	1	Configurable only when transmission or reception from a mailbox (programmed in one-shot mode) is aborted
0	1	0	Configured as a receive mailbox for a data frame or a remote frame
0	1	1	Configured as a one-shot receive mailbox for a data frame or a remote frame
1	0	0	Configured as a transmit mailbox for a data frame or a remote frame
1	0	1	Configured as a one-shot transmit mailbox for a data frame or a remote frame
1	1	0	Do not set
1	1	1	Do not set

TRMREQ, RECREQ, and ONESHOT: Bits in the CiMCTLj register (i = 0, 1; j = 0 to 31)

When a mailbox is configured as a receive mailbox or a one-shot receive mailbox, note the following:

- (1) Before a mailbox is configured as a receive mailbox or a one-shot receive mailbox, set the CiMCTLj register to 00h (i = 0, 1; j = 0 to 31).
- (2) A received message is stored into the first mailbox that matches the condition according to the result of receive mode configuration and acceptance filtering. Upon deciding which mailbox stores the received message, the mailbox with the smaller number has higher priority.
- (3) In CAN operation mode, when a CAN module transmits a message whose ID matches with the ID/mask set of a mailbox configured to receive messages, the CAN module never receives the transmitted data. In self test mode, however, the CAN module may receive its transmitted data. In this case, the CAN module sends an ACK.

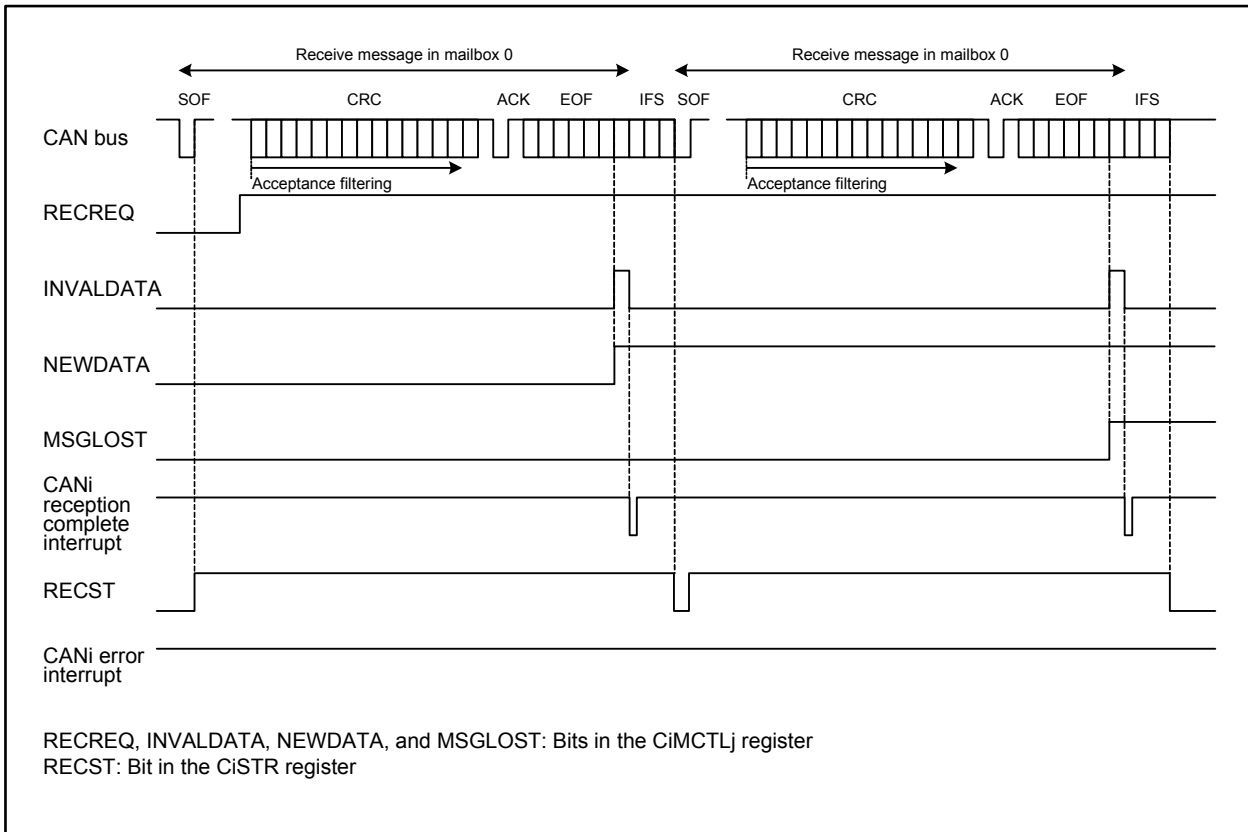
When a mailbox is configured as a transmit mailbox or a one-shot transmit mailbox, note the following:

- (1) Before a mailbox is configured as a transmit mailbox or one-shot transmit mailbox, ensure that the CiMCTLj register is 00h and that there is no pending abort process.

### 24.6.1 Reception

Figure 24.43 shows an operation example of data frame reception in overwrite mode.

This example shows the operation of overwriting the first message when the CAN module receives two consecutive CAN messages that match the receiving conditions of the CiMCTL0 register ( $i = 0, 1$ ).

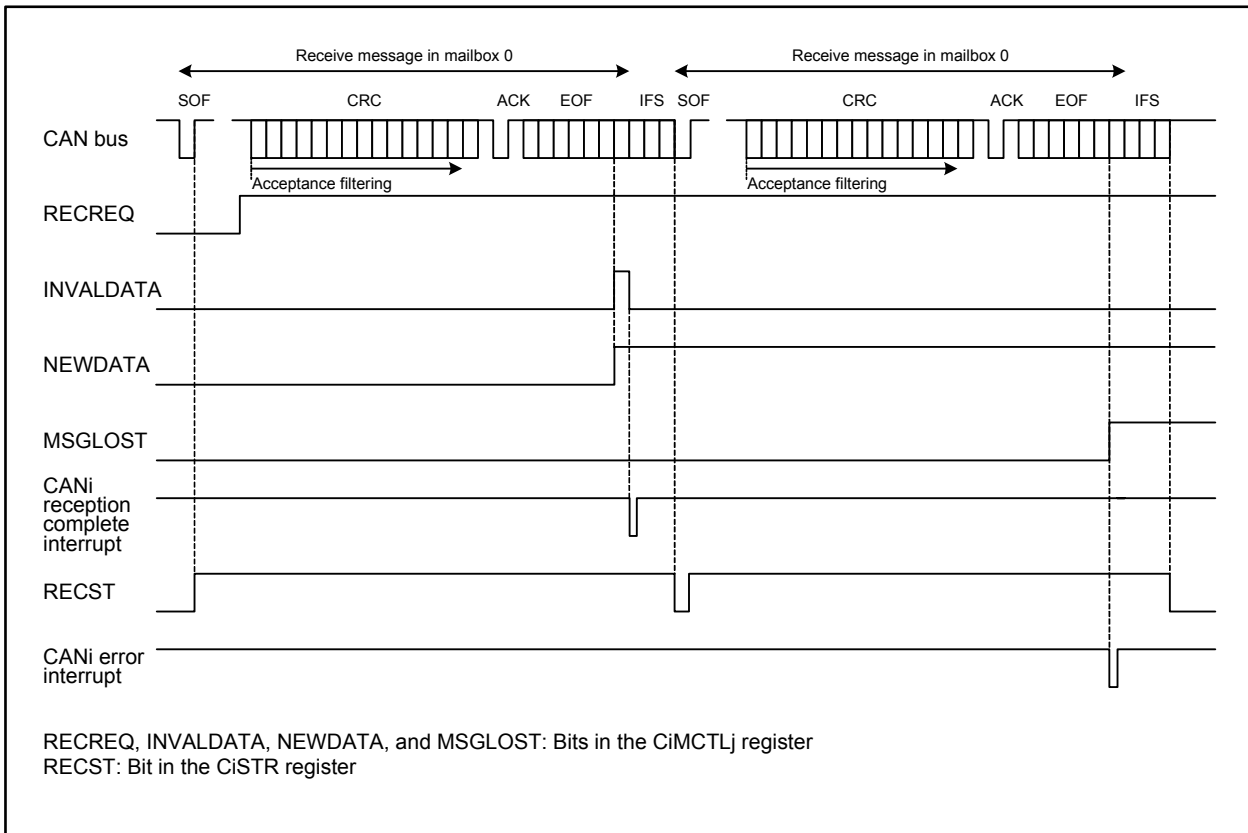


**Figure 24.43 Operation Example of Data Frame Reception in Overwrite Mode ( $i = 0, 1$ ;  $j = 0$  to 31)**

- (1) When an SOF is detected on the CAN bus, the RECST bit in the CiSTR register becomes 1 (reception in progress) if the CAN module has no message ready to start transmission.
- (2) The acceptance filter procedure starts at the beginning of the CRC field to select the receive mailbox.
- (3) After a message has been received, the NEWDATA bit in the CiMCTLj register for the receive mailbox becomes 1 (new data being updated/stored in the mailbox) ( $j = 0$  to 31). Simultaneously, the INVALIDDATA bit in the CiMCTLj register becomes 1 (message is being updated), and then the INVALIDDATA bit becomes 0 (message valid) again after the complete message is transferred to the mailbox.
- (4) When the interrupt enable bit in the CiMIER register for the receive mailbox is 1 (interrupt enabled), the CANi reception complete interrupt request is generated. This interrupt occurs when the INVALIDDATA bit becomes 0.
- (5) After reading the message from the mailbox, the NEWDATA bit needs to be set to 0 by a program.
- (6) In overwrite mode, if the next CAN message has been received into a mailbox whose NEWDATA bit is still set to 1, the MSGLOST bit in the CiMCTLj register becomes 1 (message has been overwritten). The new received message is transferred to the mailbox. The CANi reception complete interrupt request is generated the same as in (4).

Figure 24.44 shows an operation example of data frame reception in overrun mode.

This example shows the operation of overrunning the second message when the CAN module receives two consecutive CAN messages that match the receiving conditions of the CiMCTL0 register ( $i = 0, 1$ ).



**Figure 24.44 Operation Example of Data Frame Reception in Overrun Mode ( $i = 0, 1; j = 0$  to 31)**

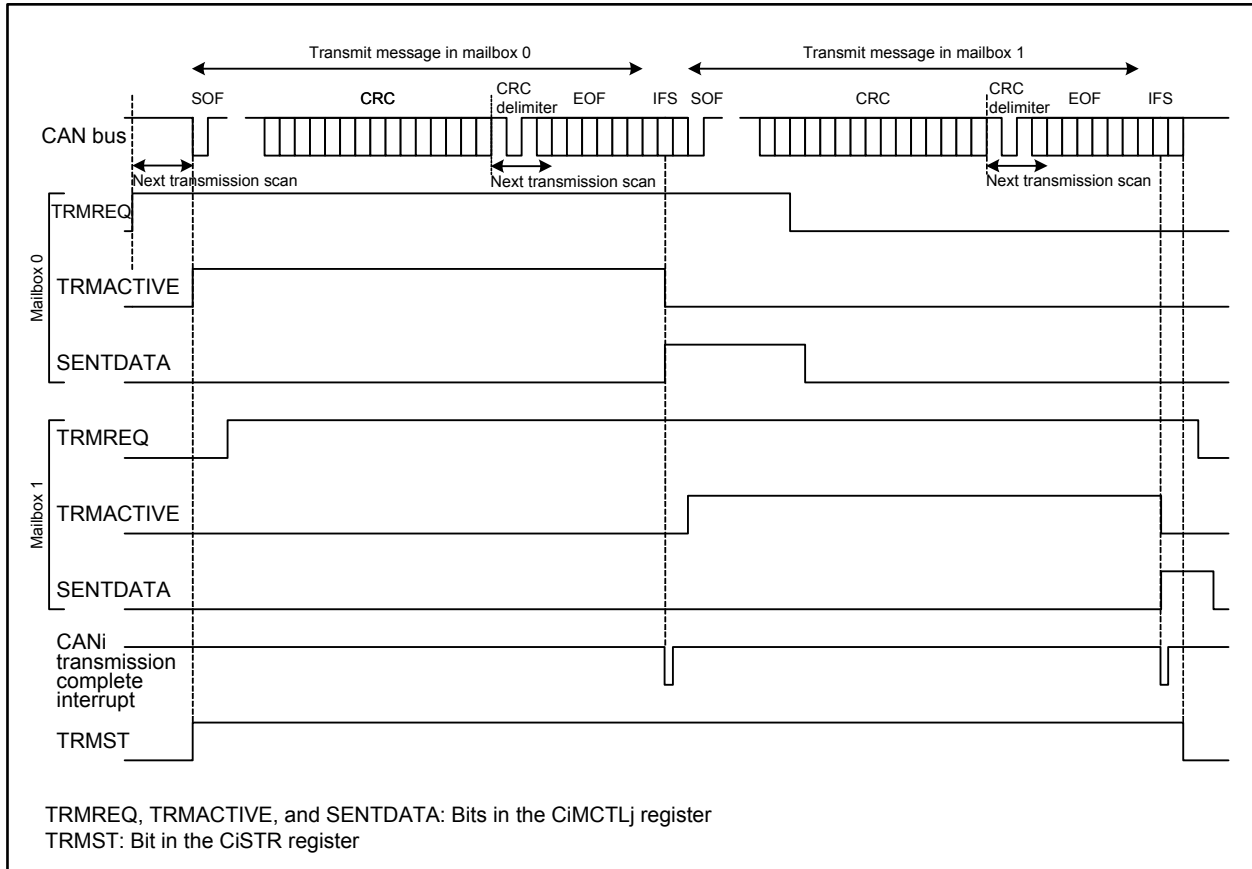
(1) to (5) are the same as overwrite mode.

(6) In overrun mode, if the next message has been received before the NEWDATA bit is set to 0, the MSGLOST bit in the CiMCTLj register becomes 1 (message has been overrun) ( $j = 0$  to 31). The new received message is discarded and a CANi error interrupt request is generated if the corresponding interrupt enable bit in the CiEIER register is 1 (interrupt enabled).



## 24.6.2 Transmission

Figure 24.45 shows an operation example of data frame transmission. This example shows an operation of transmitting messages that have been set in registers CiMCTL0 and CiMCTL1 ( $i = 0, 1$ ).



**Figure 24.45 Operation Example of Data Frame Transmission ( $i = 0, 1$ ;  $j = 0$  to 31)**

- (1) When the TRMREQ bit in the CiMCTLj register is set to 1 (transmit mailbox) in the bus-idle state, the mailbox scan procedure starts to decide the highest-priority mailbox for transmission ( $j = 0$  to 31). Once the transmit mailbox is decided, the TRMACTIVE bit in the CiMCTLj register becomes 1 (from when a transmission request is received until transmission is completed, or an error/arbitration lost has occurred), the TRMST bit in the CiSTR register becomes 1 (transmission in progress), and the CAN module starts transmission. <sup>(1)</sup>
- (2) If other TRMREQ bits are set, the transmission scan procedure starts with the CRC delimiter for the next transmission.
- (3) If transmission is completed without losing arbitration, the SENDDATA bit in the CiMCTLj register becomes 1 (transmission completed) and the TRMACTIVE bit becomes 0 (transmission is pending, or no transmission request). If the interrupt enable bit in the CiMIER register is 1 (interrupt enabled), the CANi transmission complete interrupt request is generated.
- (4) When requesting the next transmission from the same mailbox, set bits SENDDATA and TRMREQ to 0, then set the TRMREQ bit to 1 after checking that bits SENDDATA and TRMREQ have been set to 0.

Note:

1. If arbitration is lost after the CAN module starts transmission, the TRMACTIVE bit becomes 0. The transmission scan procedure is performed again to search for the highest-priority transmit mailbox from the beginning of the CRC delimiter. If an error occurs either during transmission or following the loss of arbitration, the transmission scan procedure is performed again from the start of the error delimiter to search for the highest-priority transmit mailbox.

## 24.7 CAN Interrupts

The CAN module provides the following CAN interrupts:

- CANi wakeup interrupt (i = 0, 1)
- CANi reception complete interrupt
- CANi transmission complete interrupt
- CANi receive FIFO interrupt
- CANi transmit FIFO interrupt
- CANi error interrupt

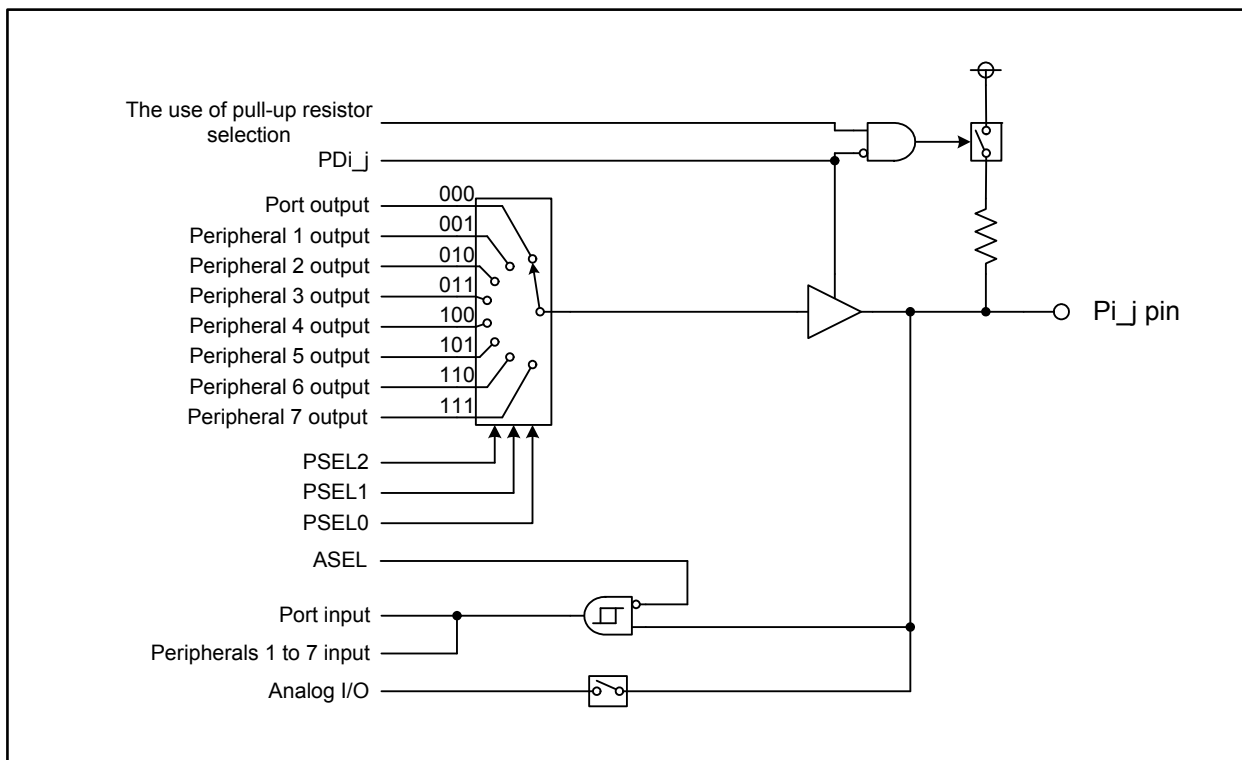
There are eight types of interrupt sources for the CANi error interrupts. These sources can be determined by checking the CiEIFR register.

- Bus error
- Error-warning
- Error-passive
- Bus-off entry
- Bus-off recovery
- Receive overrun
- Overload frame transmission
- Bus lock

## 25. I/O Pins

Each pin of the MCU functions as a programmable I/O port or an I/O pin for integrated peripherals. These functions can be switched by the function select registers. The pull-up resistors are enabled for every group of four pins. However, a pull-up resistor is separated from other peripheral functions even if it is enabled, when a pin functions as an output pin.

Figure 25.1 shows a block diagram of typical I/O pin.



**Figure 25.1** Typical I/O Pin Block Diagram ( $i = 0$  to  $9$ ;  $j = 0$  to  $7$ )

The registers to control I/O pins are as follows: port  $P_i$  direction register ( $PDI$  register), output function select register, and pull-up control register. The  $PDI$  register selects the input or output state of pins. The output function select register which selects an output function consists of bits  $PSEL2$  to  $PSEL0$ , and  $ASEL$ . Bits  $PSEL2$  to  $PSEL0$  select a function as a programmable I/O or peripheral output (except analog output). The  $ASEL$  bit prevents the increase in power consumption of input buffer caused by an intermediate potential when a pin functions as an analog I/O pin. The pull-up control register enables/disables the pull-up resistors. To use a pin as an analog I/O pin, the  $PDI_j$  bit should be set to 0 (input), and bits  $PSEL2$  to  $PSEL0$  should be set to 000b and the  $ASEL$  bit should be set to 1.

The input-only port  $P8\_5$  shares a pin with  $\overline{NMI}$  and has no function select register or bit 5 in the  $PD8$  register. Port  $P9\_1$  also functions as an input-only port. The function select register and bit 1 in the  $PD9$  register are reserved. Port  $P9$  is protected from unexpected write accesses by the  $PRC2$  bit in the  $PRCR$  register. Ports  $P3$ ,  $P7$ , and  $P8$  are protected from unexpected write accesses by the  $PRC30$  bit in the  $PRCR3$  register (refer to 9. "Protection").

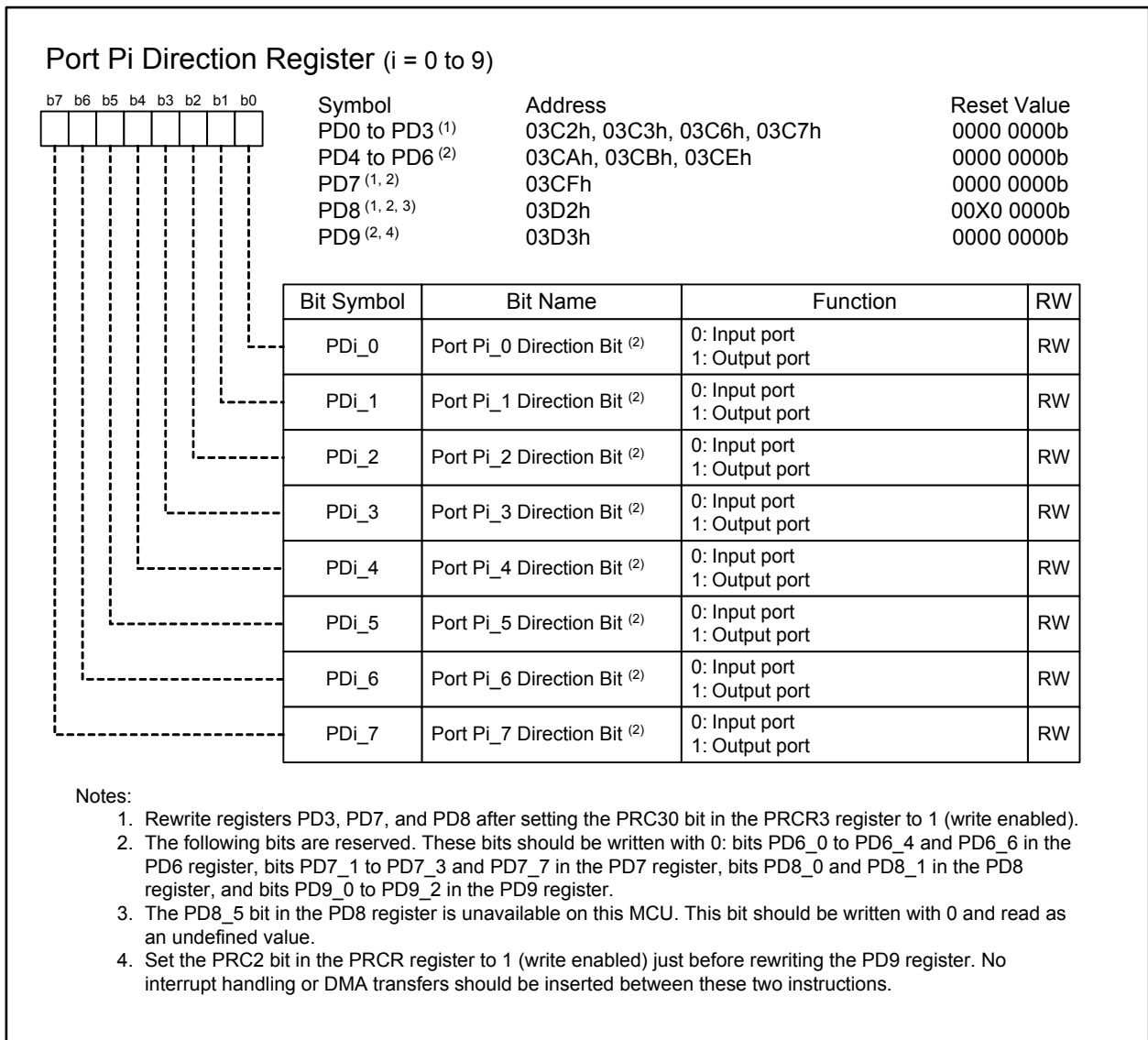
## 25.1 Port Pi Direction Register (PDi Register, i = 0 to 9)

The PDi register selects the input or output state of pins. Bits in this register correspond to respective pins.

Figure 25.2 shows the PDi register.

No register bit is provided for port P8\_5. For port P9\_1, a reserved bit is provided.

The PD9 register is protected from unexpected write accesses by setting the PRC2 bit in the PRCR register. Registers PD3, PD7, and PD8 are protected by the PRC30 bit in the PRCR3 register (refer to 9. "Protection").



**Figure 25.2 Registers PD0 to PD9**

## 25.2 Output Function Select Registers

When a programmable I/O port and peripheral output share a pin, this register selects the output function of the pin. Regardless of the register setting, a signal is input to all the connected peripherals.

The output function select register consists of bits PSEL2 to PSEL0, and ASEL. Bits PSEL2 to PSEL0 select a function as programmable I/O or peripheral output (except analog output). The ASEL bit prevents the increase in power consumption caused by an intermediate potential generated when a pin functions as an analog I/O pin.

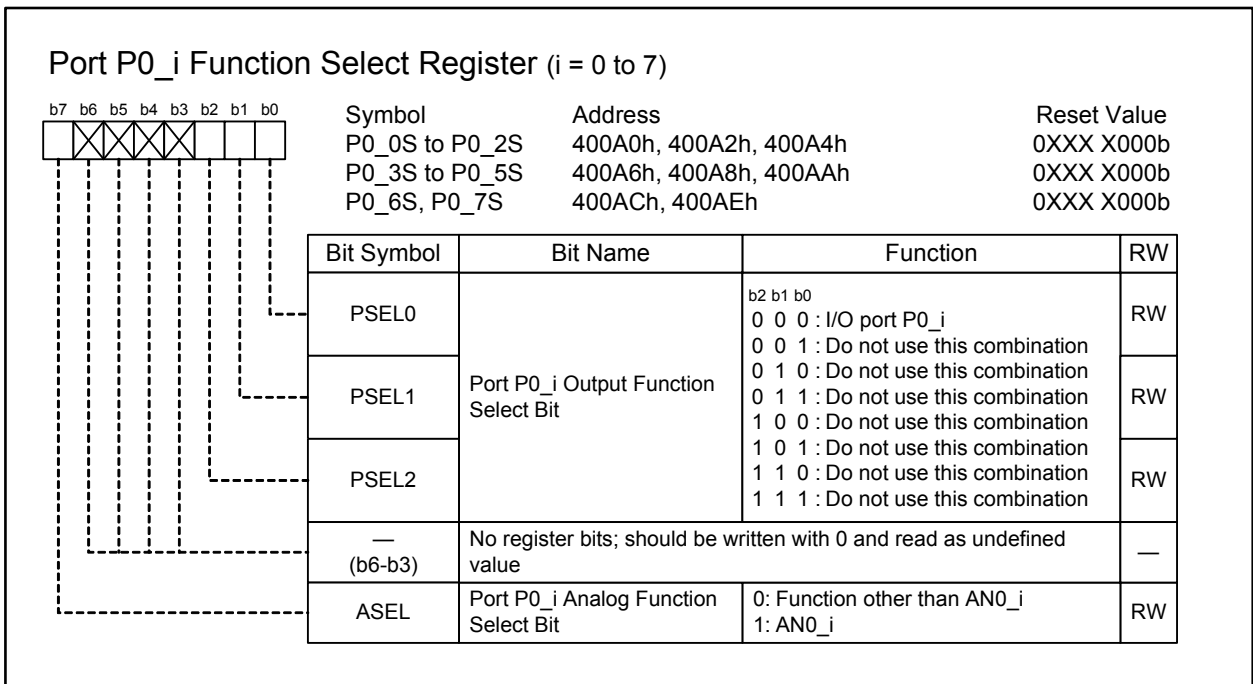
Table 25.1 shows the peripherals assigned to each PSEL2 to PSEL0 bit combination, and Figures 25.3 to 25.12 show the function select registers.

Note that ports P8\_5 and P9\_1 (input only) have no output function select registers.

The P9\_iS register is protected from unexpected write accesses by setting the PRC2 bit in the PRCR register and registers P3\_iS, P7\_iS, and P8\_iS are protected by the PRC30 bit in the PRCR3 register (refer to 9. "Protection").

**Table 25.1 Peripheral Assignment**

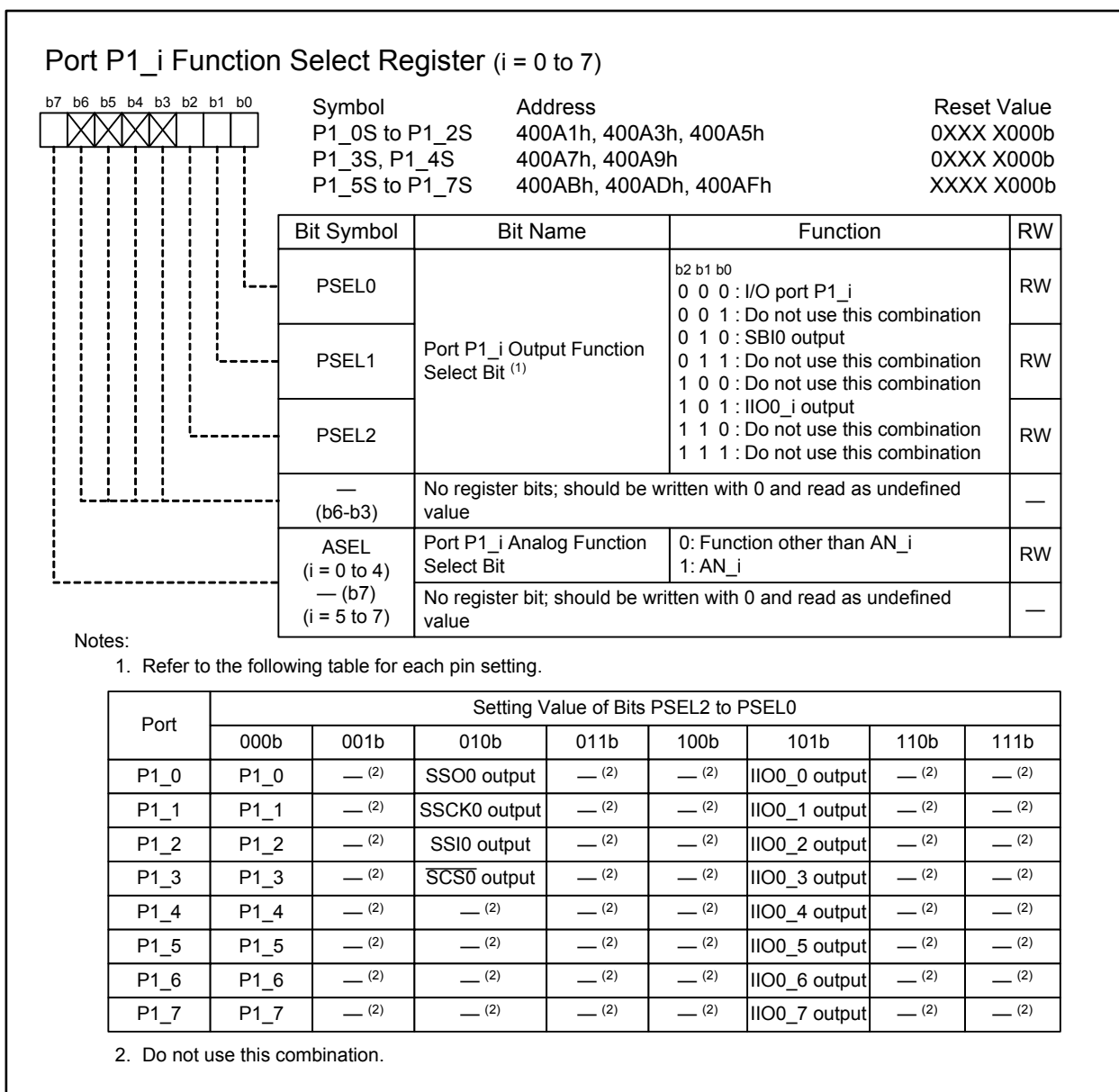
Bits PSEL2 to PSEL0	Peripherals
001b	Timer
010b	Three-phase motor control timers, Serial bus interface 0
011b	UART
100b	UART special function
101b	Intelligent I/O group 0, CAN channel 0, LIN module
110b	CAN channel 1
111b	



**Figure 25.3 Registers P0\_0S to P0\_7S**

Port P0<sub>i</sub> shares a pin with the AN0<sub>i</sub> input pin for the A/D converter (i = 0 to 7).

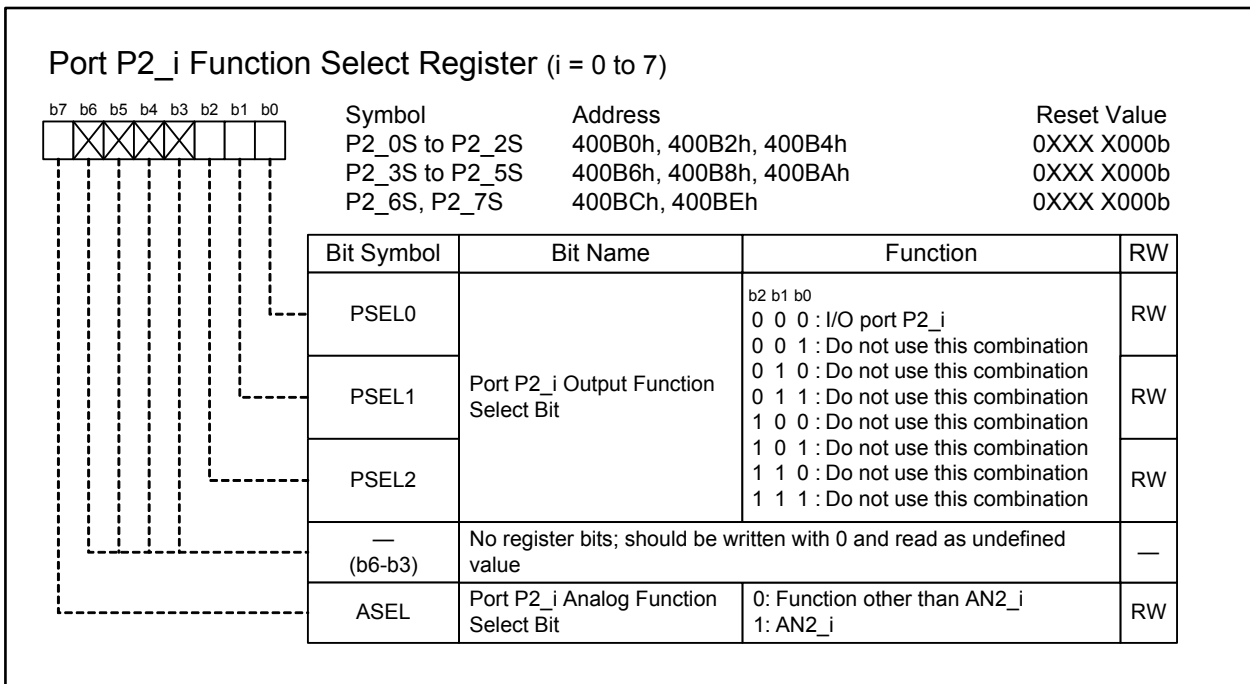
To use it as a programmable I/O port, the P0<sub>i</sub>S register should be set to 00h. To use it as an A/D converter input pin, this register should be set to 80h and the PD0<sub>i</sub> bit should be set to 0 (port P0<sub>i</sub> functions as an input port).



**Figure 25.4 Registers P1\_0S to P1\_7S**

Port P1<sub>i</sub> shares a pin with the serial bus interface (SBI0), AN<sub>i</sub> input pin for the A/D converter, intelligent I/O group 0 (IIO0), and external interrupt input pin (i = 0 to 7).

To use it as an output pin, the PD1<sub>i</sub> bit should be set to 1 (port P1<sub>i</sub> functions as an output port) and a function should be selected according to Figure 25.4. To use it as an input pin (except for the A/D converter), the PD1<sub>i</sub> bit should be set to 0 (port P1<sub>i</sub> functions as an input port). To use it as an A/D converter input pin, this register should be set to 80h and the PD1<sub>i</sub> bit should be set to 0 (port P1<sub>i</sub> functions as an input port).

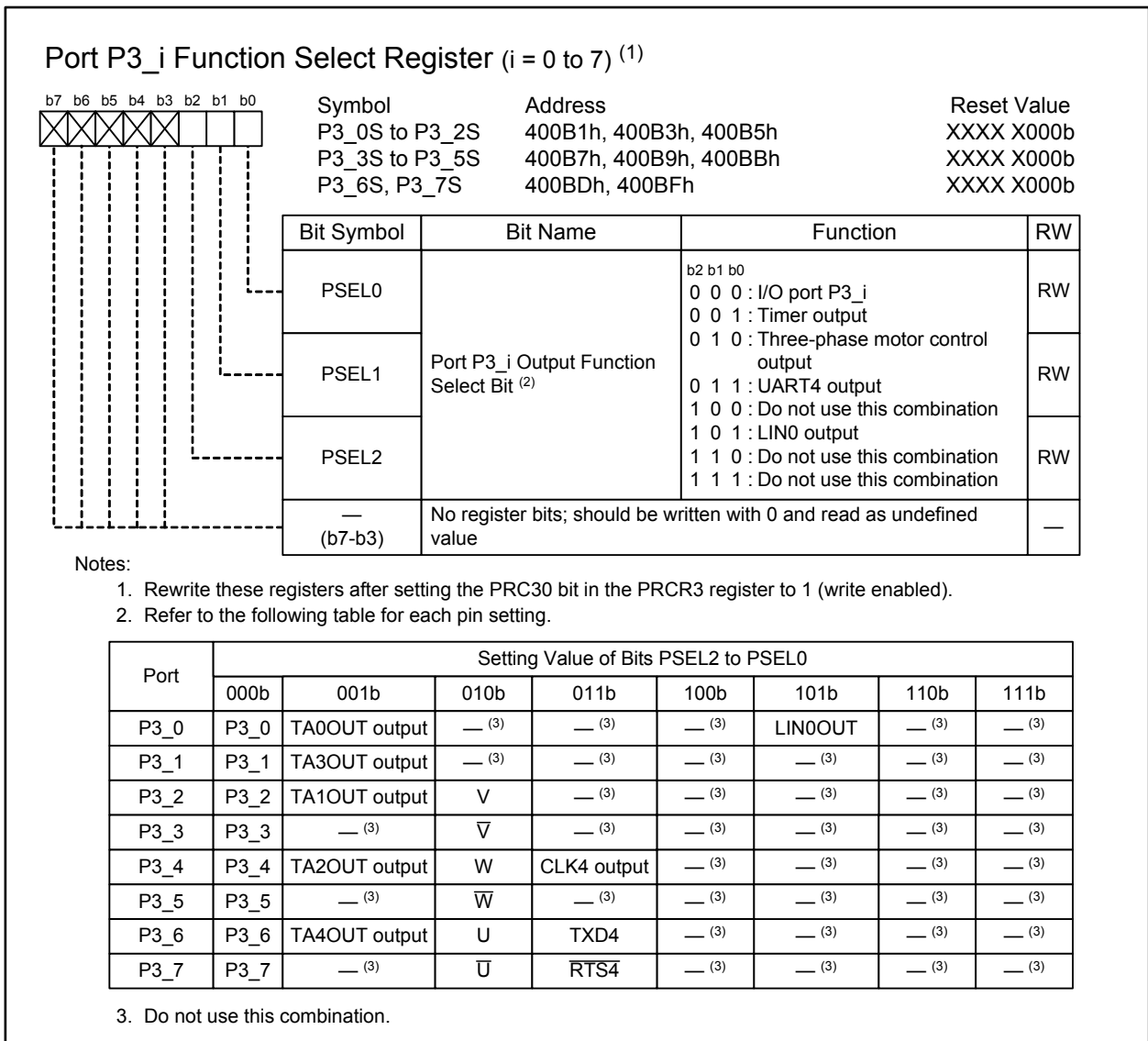


**Figure 25.5 Registers P2\_0S to P2\_7S**

Port P2<sub>i</sub> shares a pin with the AN2<sub>i</sub> pin for the A/D converter (i = 0 to 7).

To use it as a programmable I/O port, the P2<sub>i</sub>S register should be set to 00h. To use it as an A/D converter input pin, this register should be set to 80h and the PD2<sub>i</sub> bit should be set to 0 (port P2<sub>i</sub> functions as an input port).

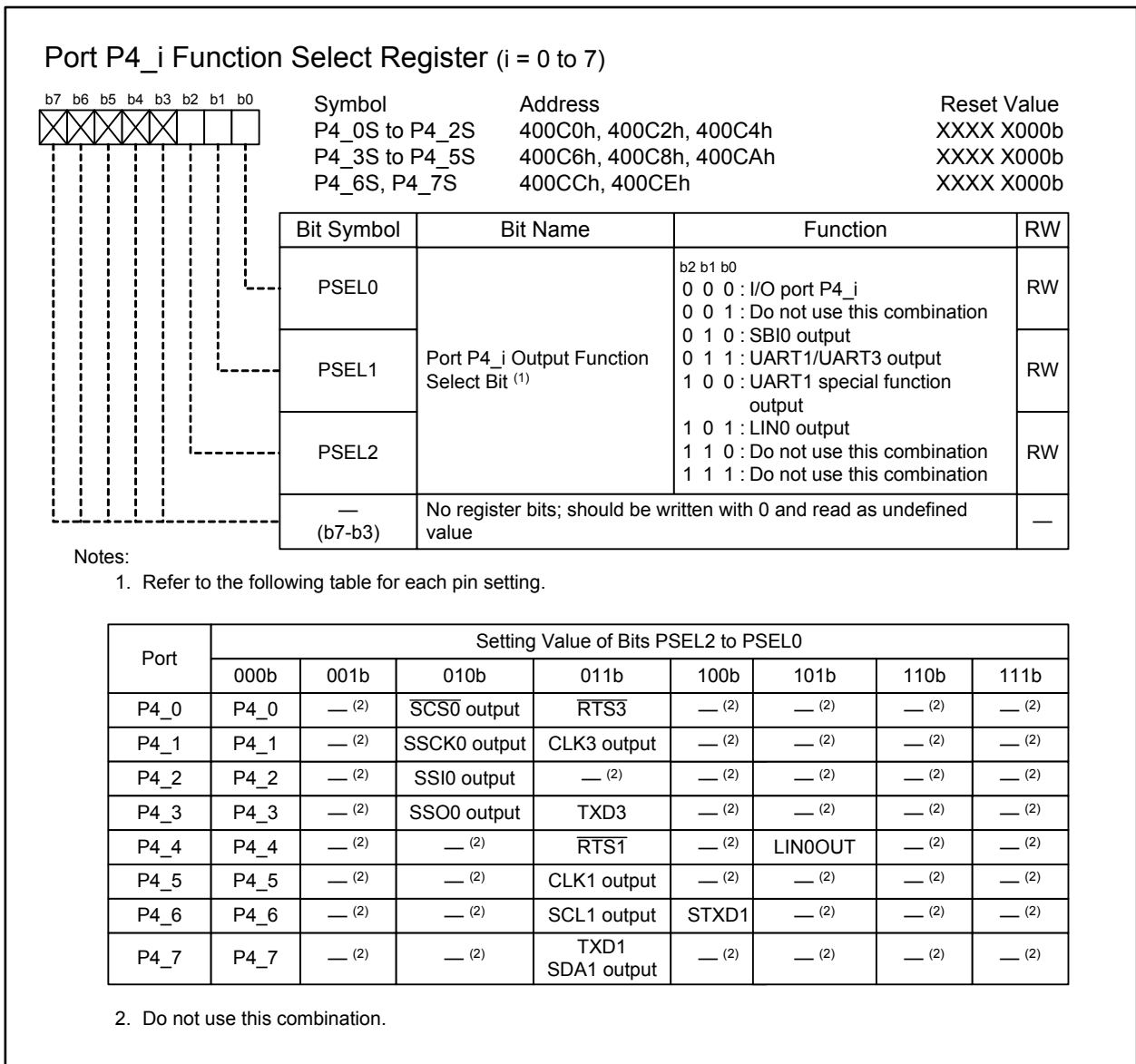




**Figure 25.6 Registers P3\_0S to P3\_7S**

Port P3<sub>i</sub> shares a pin with the timer output, three-phase motor control output, serial interface (UART4), and LIN module (i = 0 to 7).

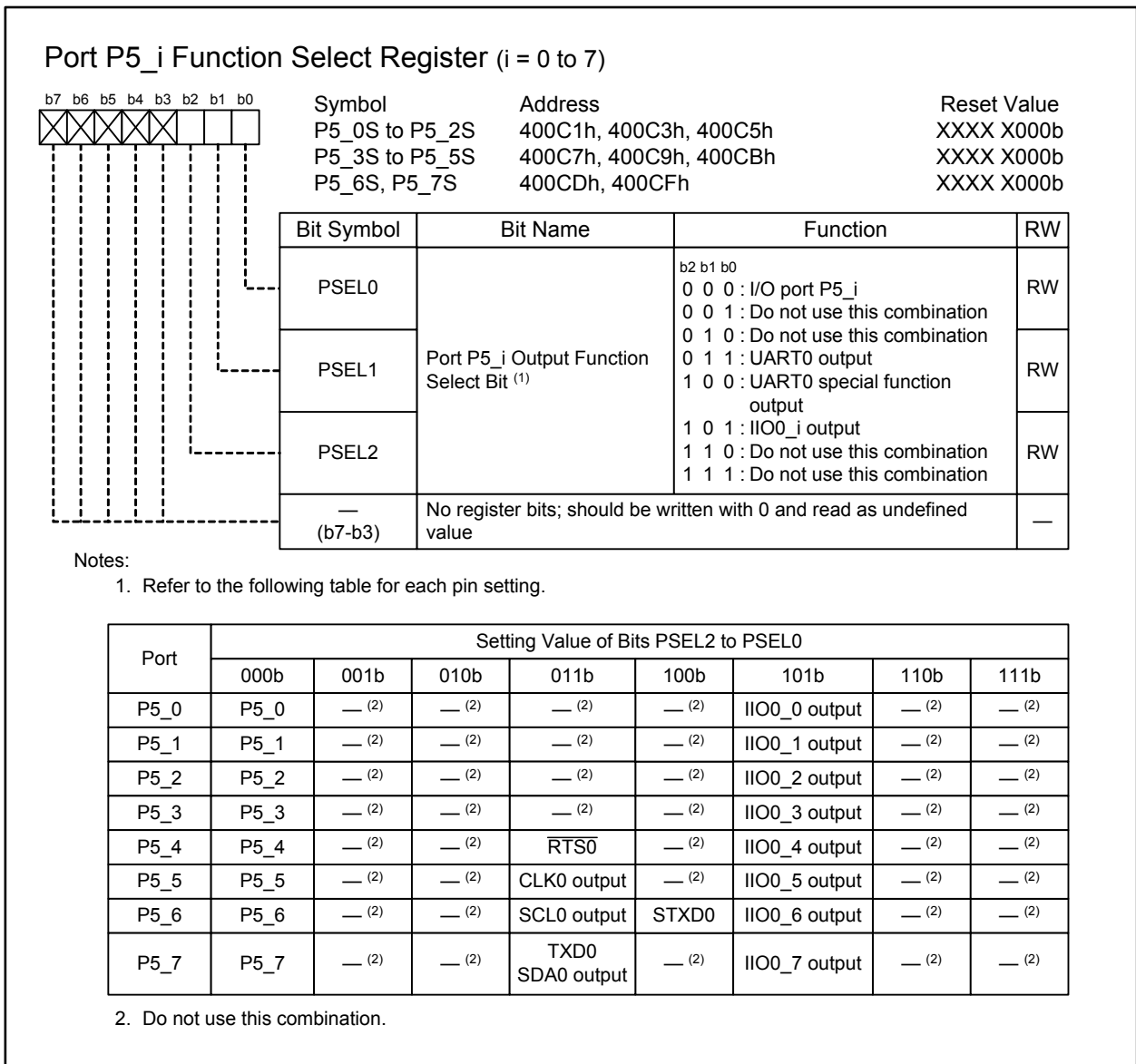
To use it as an output pin, the PD3<sub>i</sub> bit should be set to 1 (port P3<sub>i</sub> functions as an output port) and a function should be selected according to Figure 25.6. To use it as an input pin, the PD3<sub>i</sub> bit should be set to 0 (port P3<sub>i</sub> functions as an input port).



**Figure 25.7 Registers P4\_0S to P4\_7S**

Port P4<sub>i</sub> shares a pin with the serial interface (UART1 and UART3), serial bus interface (SBI0), and LIN module (i = 0 to 7).

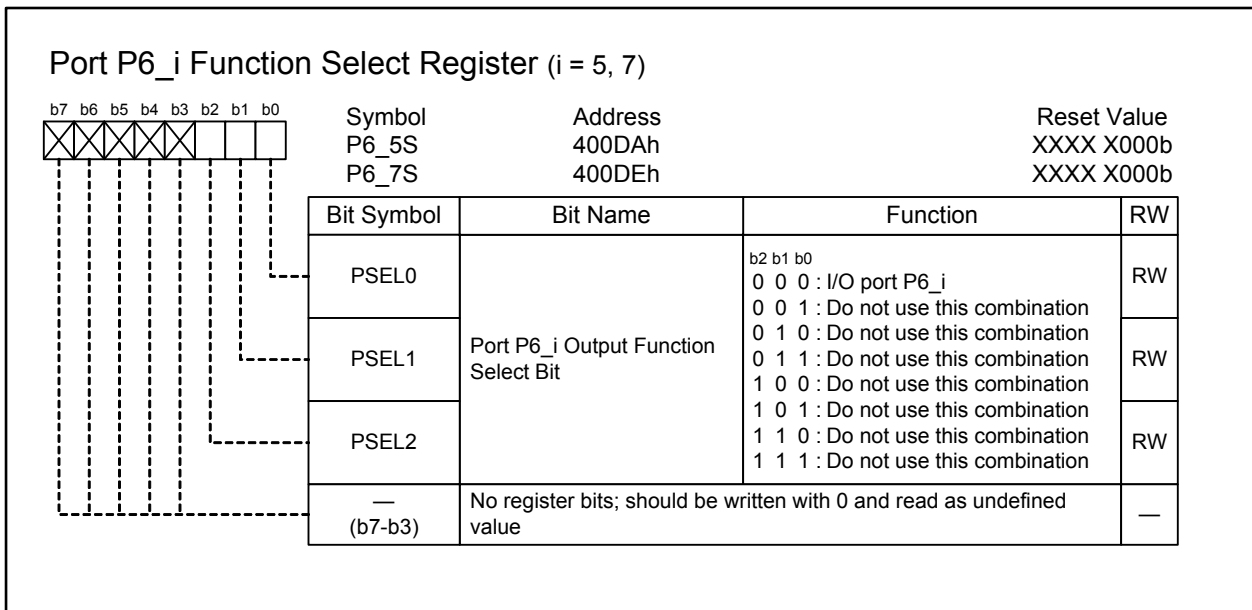
To use it as an output pin, the PD4<sub>i</sub> bit should be set to 1 (port P4<sub>i</sub> functions as an output port) and a function should be selected according to Figure 25.7. To use it as an input pin, the PD4<sub>i</sub> bit should be set to 0 (port P4<sub>i</sub> functions as an input port).



**Figure 25.8 Registers P5\_0S to P5\_7S**

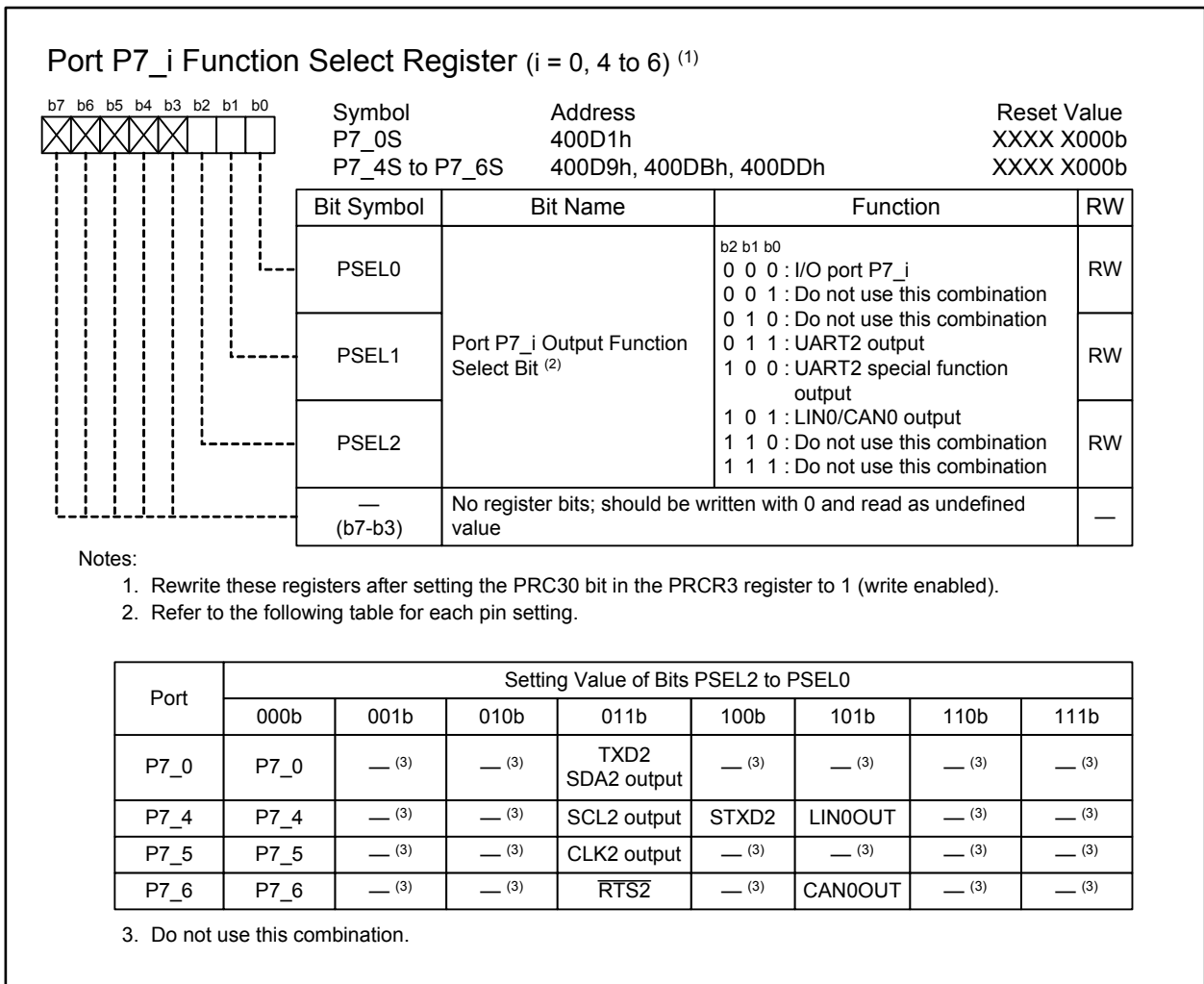
Port P5<sub>i</sub> shares a pin with intelligent I/O group 0 (IIO0) and the serial interface (UART0) (i = 0 to 7).

To use it as an output pin, the PD5<sub>i</sub> bit should be set to 1 (port P5<sub>i</sub> functions as an output port) and a function should be selected according to Figure 25.8. To use it as an input pin, the PD5<sub>i</sub> bit should be set to 0 (port P5<sub>i</sub> functions as an input port).



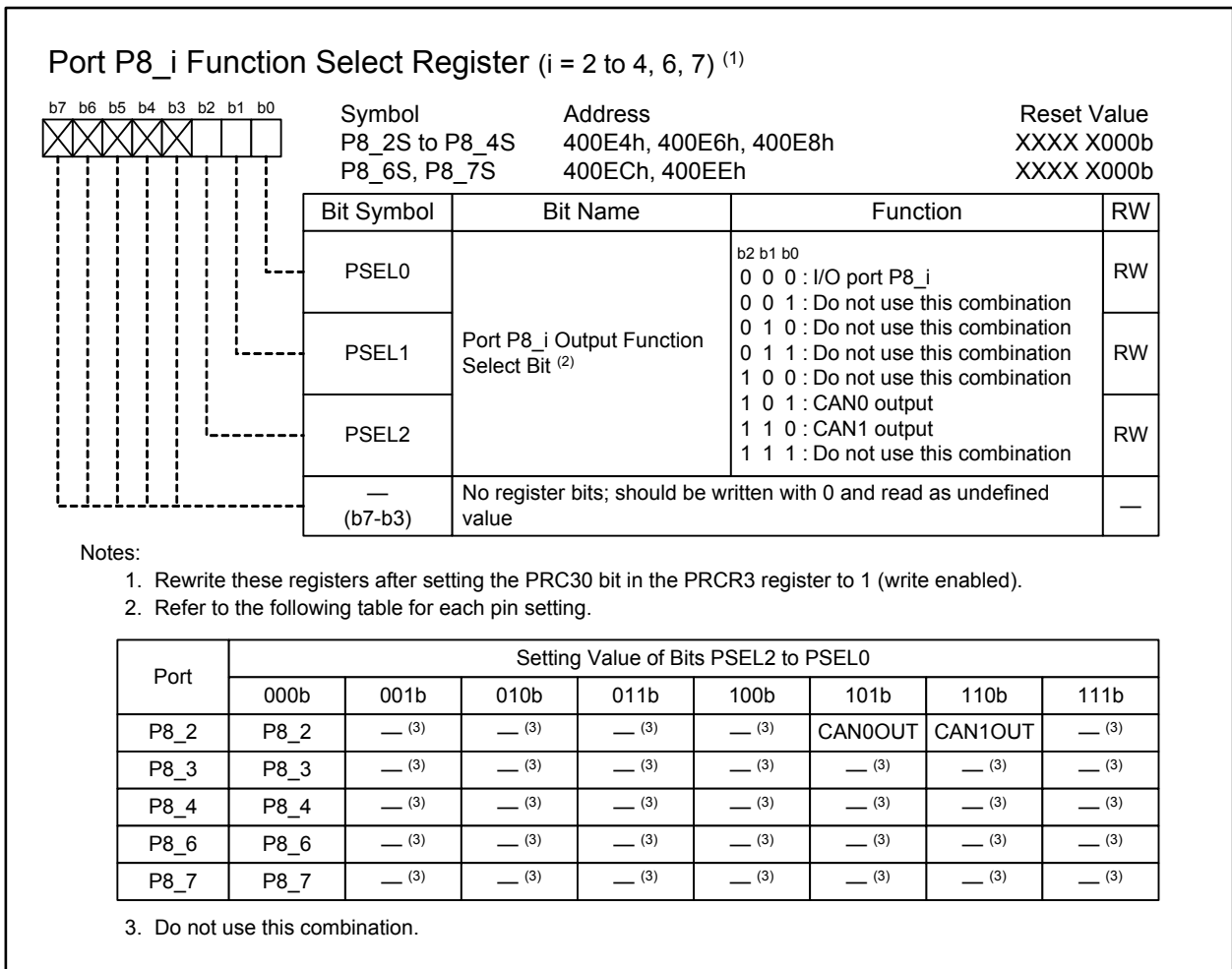
**Figure 25.9 Registers P6\_5S and P6\_7S**

Port P6<sub>i</sub> does not share a pin with any functions. The P6<sub>i</sub>S register should be set to 00h (I/O port) (i = 5 and 7).



**Figure 25.10 Registers P7\_0S and P7\_4S to P7\_6S**

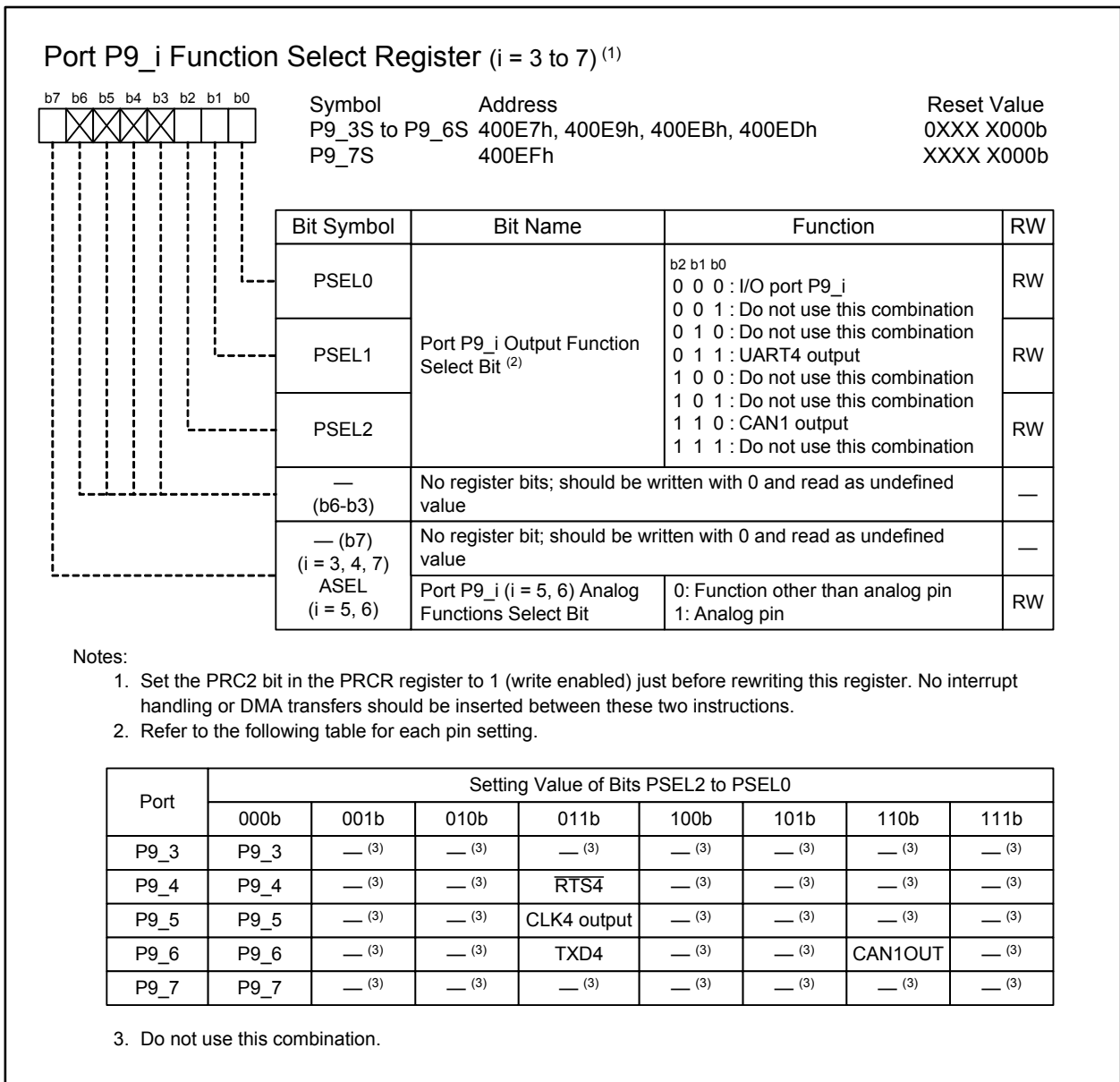
Port P7<sub>i</sub> shares a pin with the serial interface (UART2), LIN module, and CAN module (i = 0 to 7). To use it as an output pin, the PD7<sub>i</sub> bit should be set to 1 (port P7<sub>i</sub> functions as an output port) and a function should be selected according to Figure 25.10. To use it as an input pin, the PD7<sub>i</sub> bit should be set to 0 (port P7<sub>i</sub> functions as an input port).



**Figure 25.11 Registers P8\_2S to P8\_4S, P8\_6S, and P8\_7S**

Port P8<sub>i</sub> shares a pin with the CAN module and external interrupt input pin (i = 2 to 4, 6, 7).

To use it as an output pin, the PD8<sub>i</sub> bit should be set to 1 (port P8<sub>i</sub> functions as an output port) and a function should be selected according to Figure 25.11. To use it as an input pin, the PD8<sub>i</sub> bit should be set to 0 (port P8<sub>i</sub> functions as an input port).



**Figure 25.12 Registers P9\_3S to P9\_7S**

Port P9<sub>i</sub> shares a pin with the serial interface (UART4) and CAN module (i = 3 to 7). In particular, port P9<sub>i</sub> also shares a pin with the A/D converter I/O (ANEX0 and ANEX1) pin (i = 5, 6).

To use it as the A/D converter pin, the P9<sub>i</sub>S register should be set to 80h and the PD9<sub>i</sub> bit should be set to 0 (port P9<sub>i</sub> functions as an input port) irrespective of the I/O state.

To use it as an output pin for functions other than the A/D converter, the PD9<sub>i</sub> bit should be set to 1 (port P9<sub>i</sub> functions as an output port) and a function should be selected according to Figure 25.12. To use it as an input pin of functions other than the A/D converter, the PD9<sub>i</sub> bit should be set to 0 (port P9<sub>i</sub> functions as an input port).

### 25.3 Input Function Select Registers

When a peripheral input is assigned to multiple pins, these registers select which input pin should be connected to the peripheral.

Figures 25.13 to 25.17 show the input function select registers.

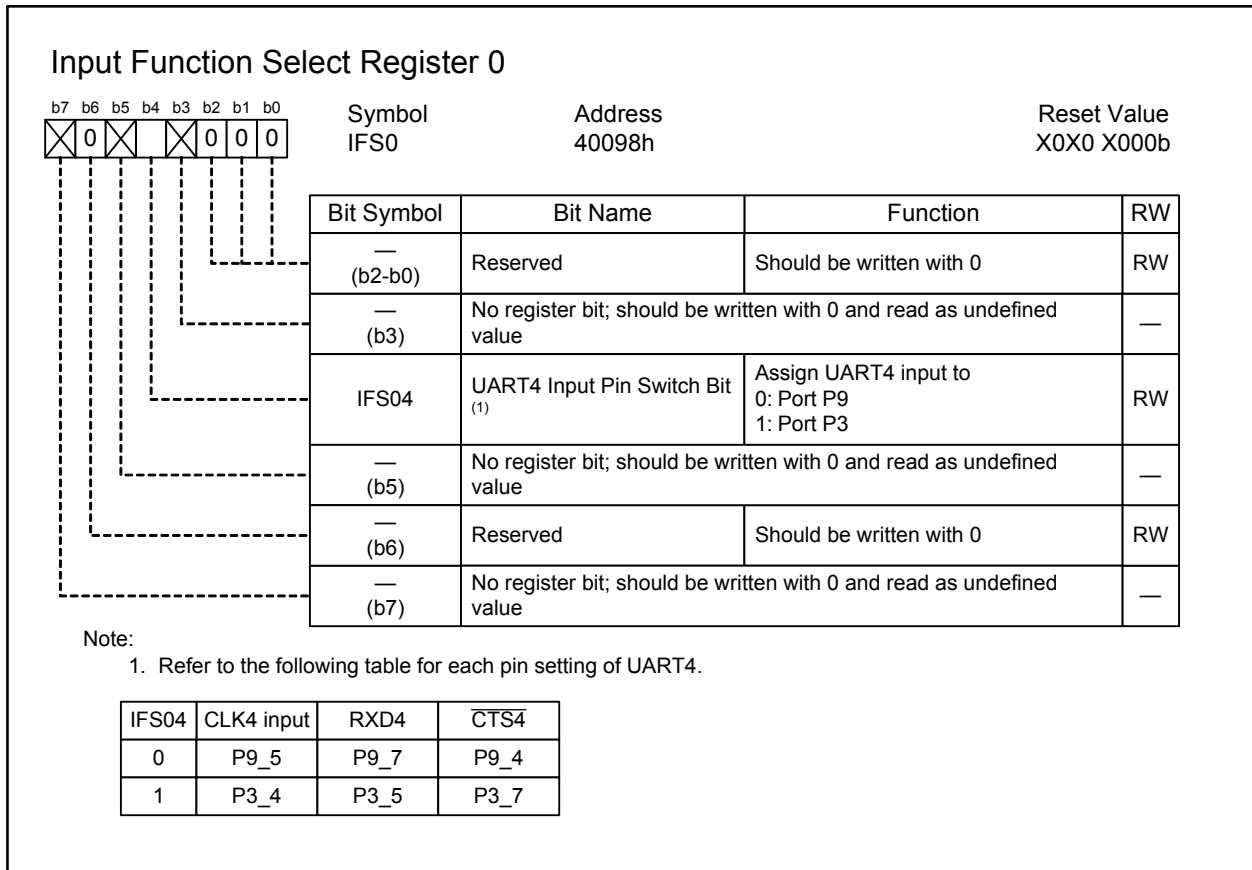


Figure 25.13 IFS0 Register

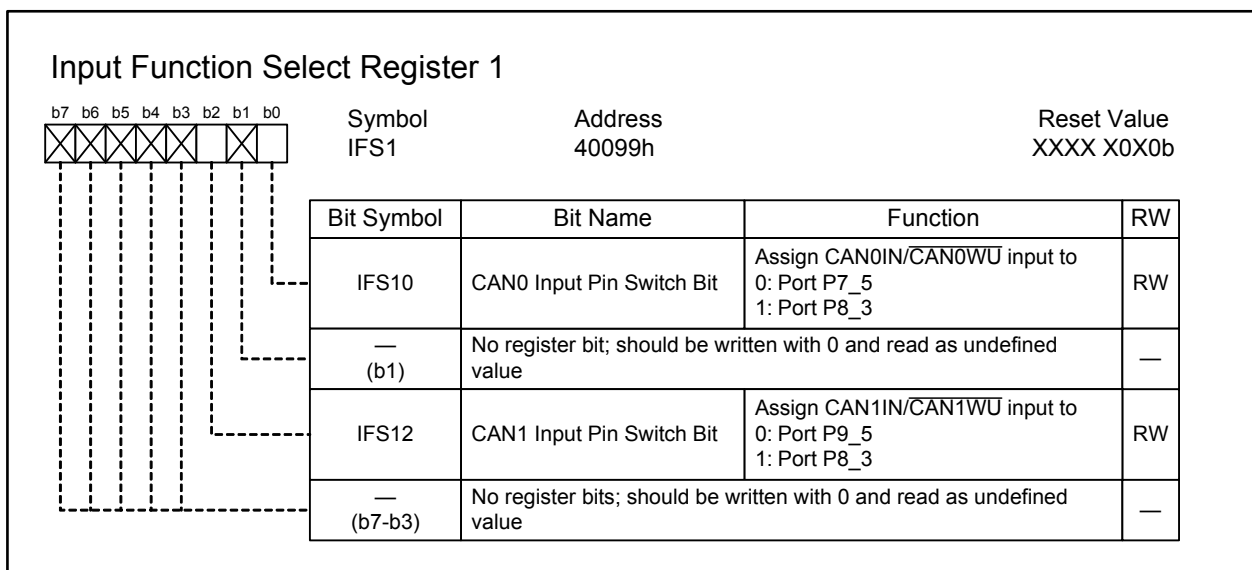


Figure 25.14 IFS1 Register



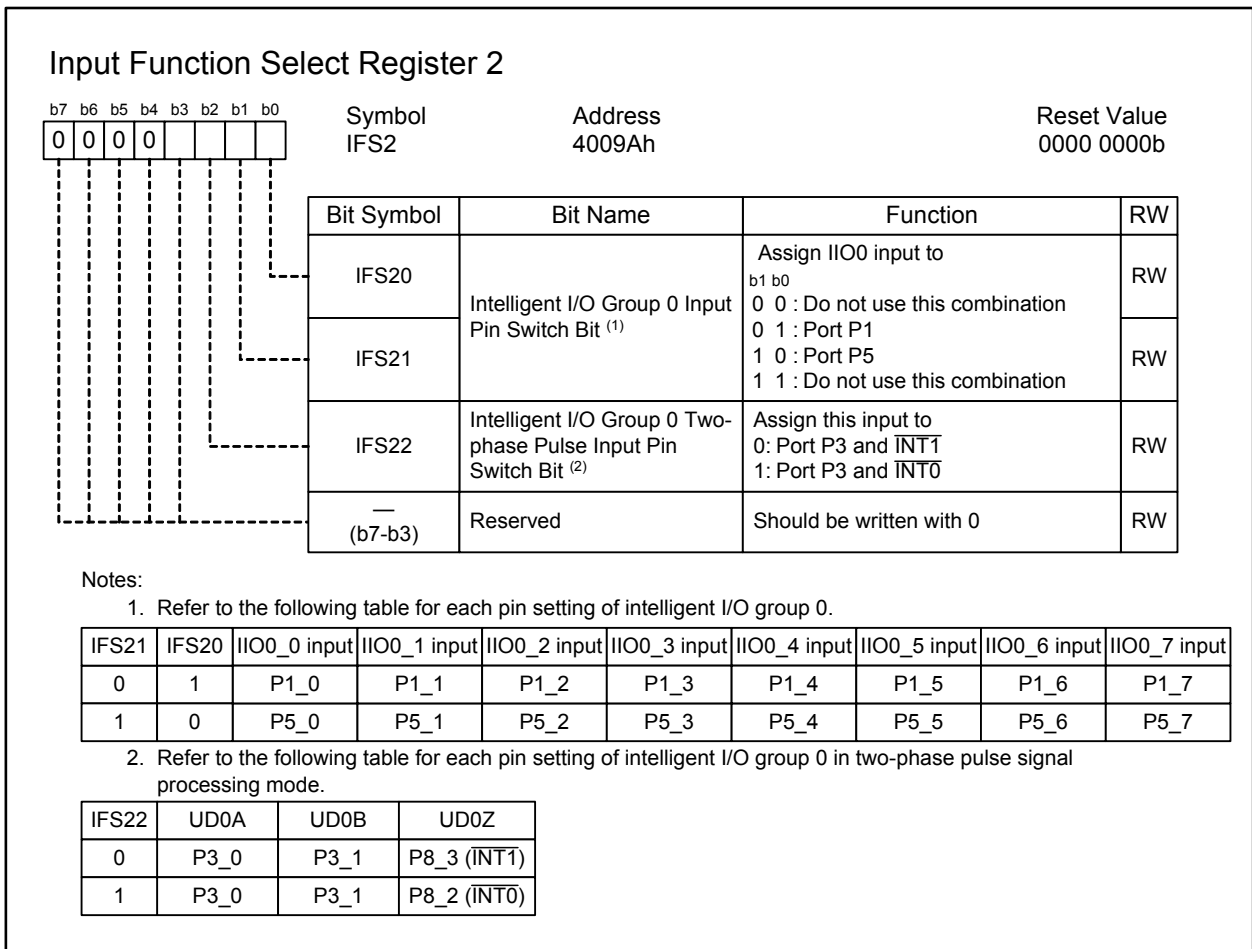
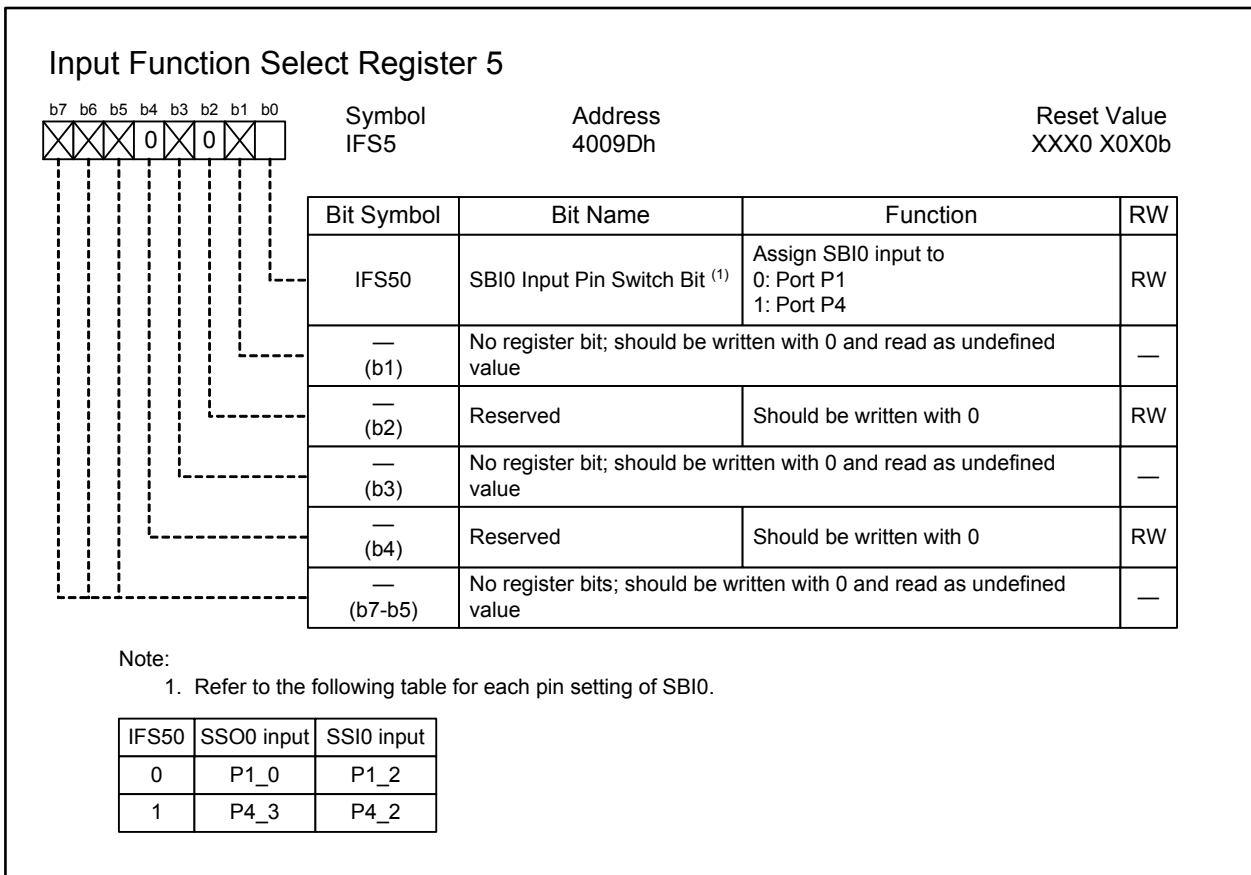
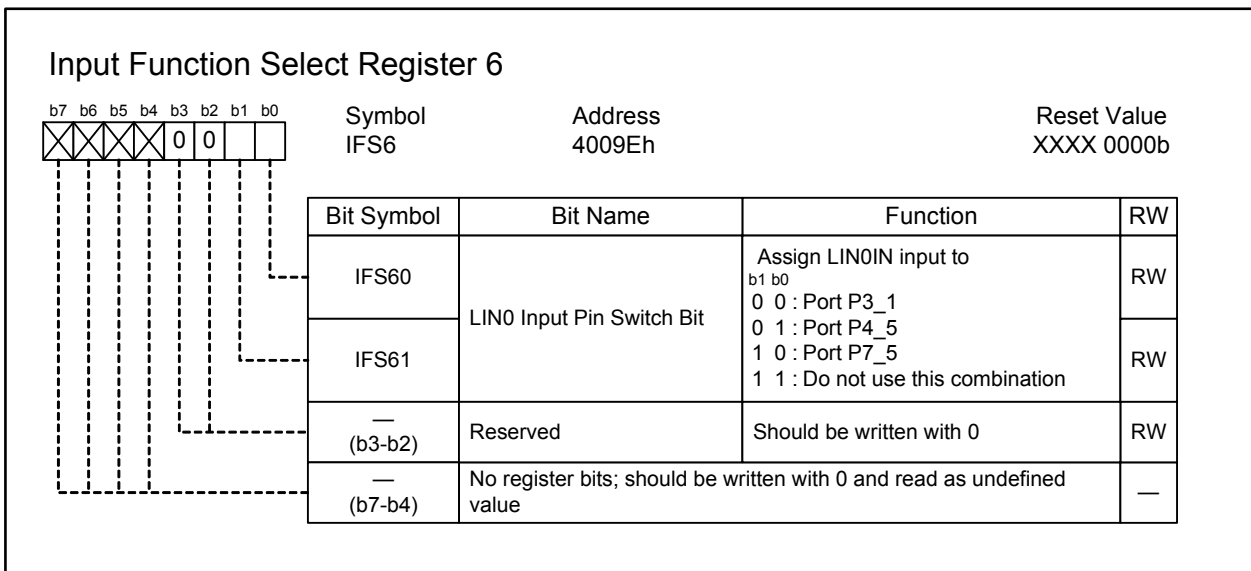


Figure 25.15 IFS2 Register



**Figure 25.16 IFS5 Register**



**Figure 25.17 IFS6 Register**

### 25.4 Pull-up Control Registers 0 to 2 (Registers PUR0 to PUR2)

Figures 25.18 to 25.20 show registers PUR0 to PUR2.

These registers enable/disable the pull-up resistors for every group of four pins. To enable the pull-up resistors, the corresponding bits in registers PUR0 to PUR2 should be set to 1 (pull-up resistor enabled) and the respective bits in the direction register should be set to 0 (input).

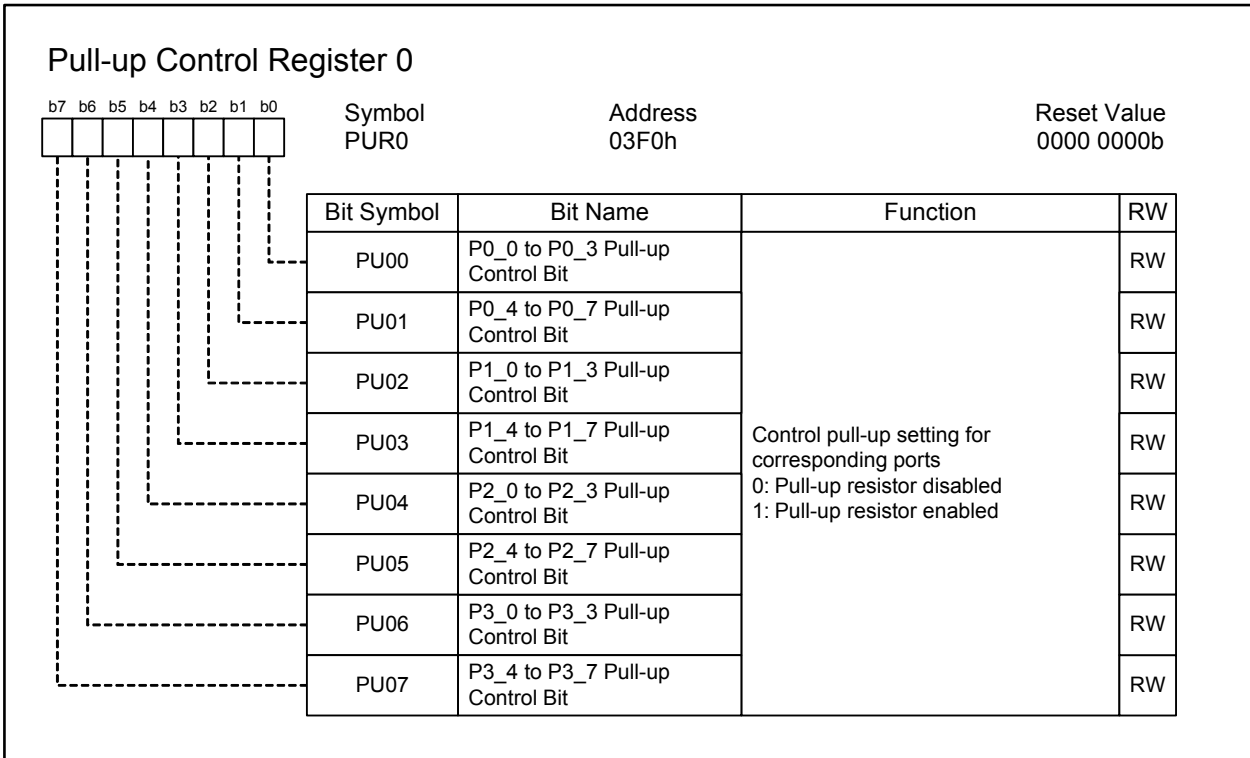


Figure 25.18 PUR0 Register

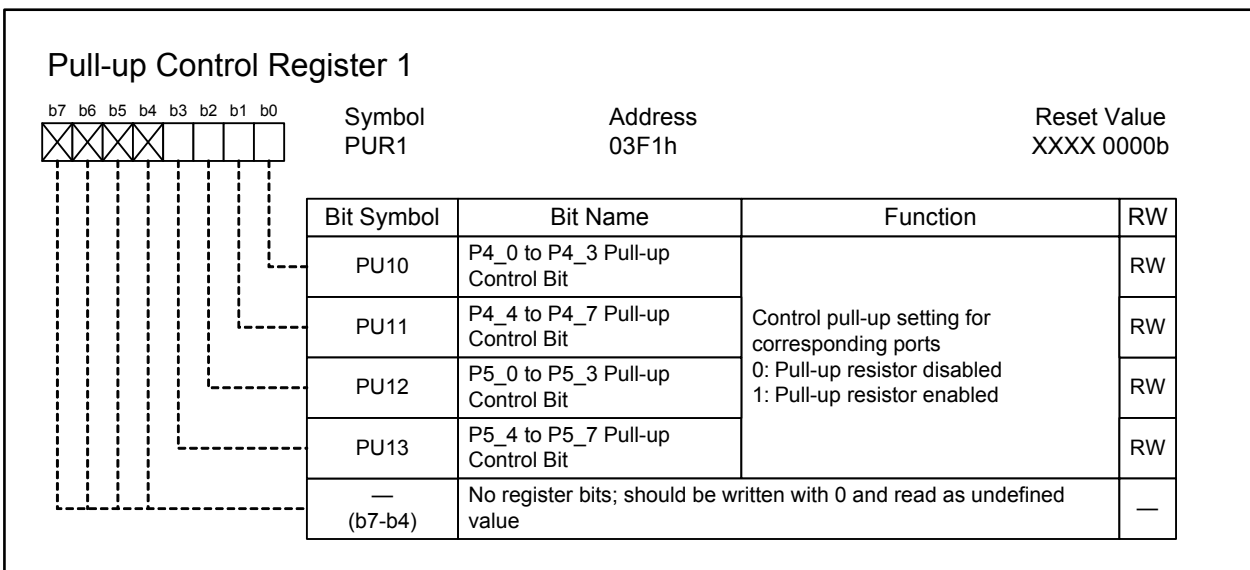
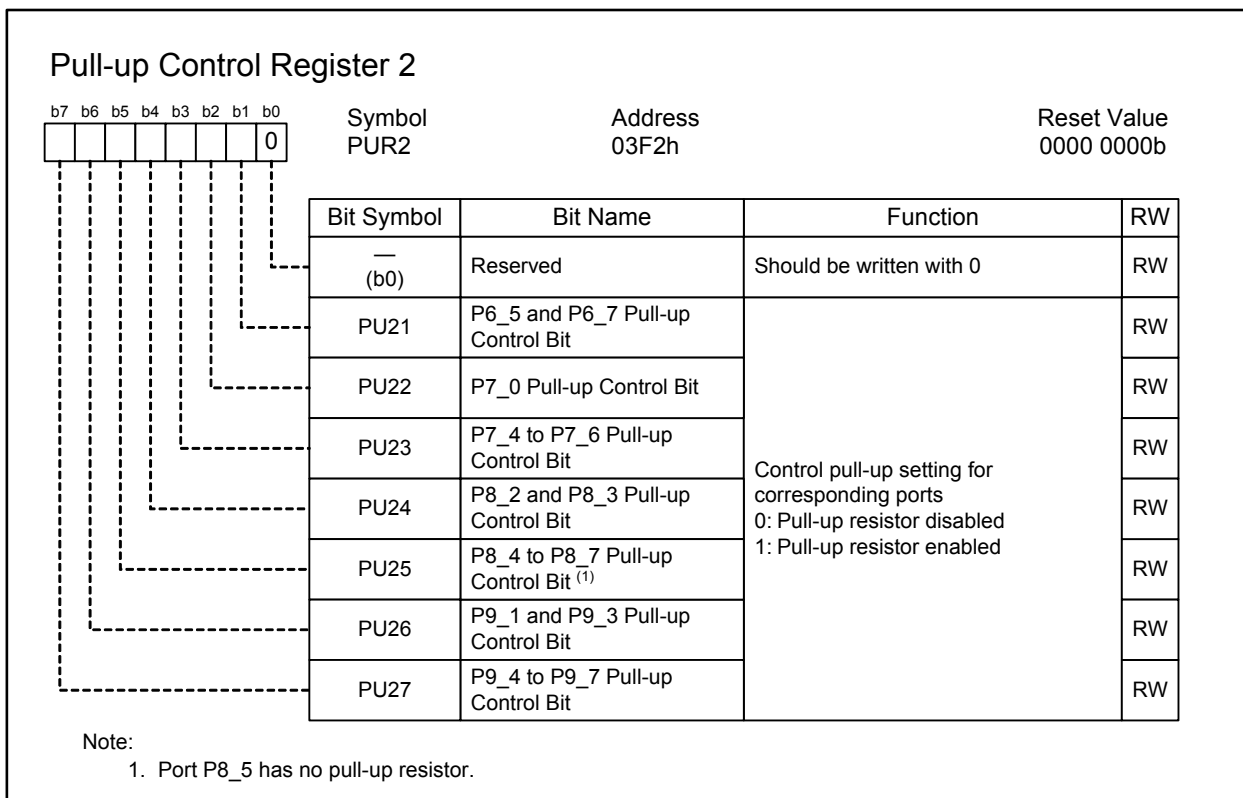


Figure 25.19 PUR1 Register



**Figure 25.20 PUR2 Register**

### 25.5 Port Control Register (PCR Register)

Figure 25.21 shows the PCR register.

This register selects an output mode for port P1 between push-pull output and pseudo-N-channel open drain output. When the PCR0 bit is set to 1, the P-channel transistor in the output buffer is turned off. Note that port P1 cannot be a perfect open drain output due to remaining parasitic diode. The absolute maximum rating of the input voltage is, therefore, -0.3 V to VCC + 0.3 V (refer to Figure 25.22).

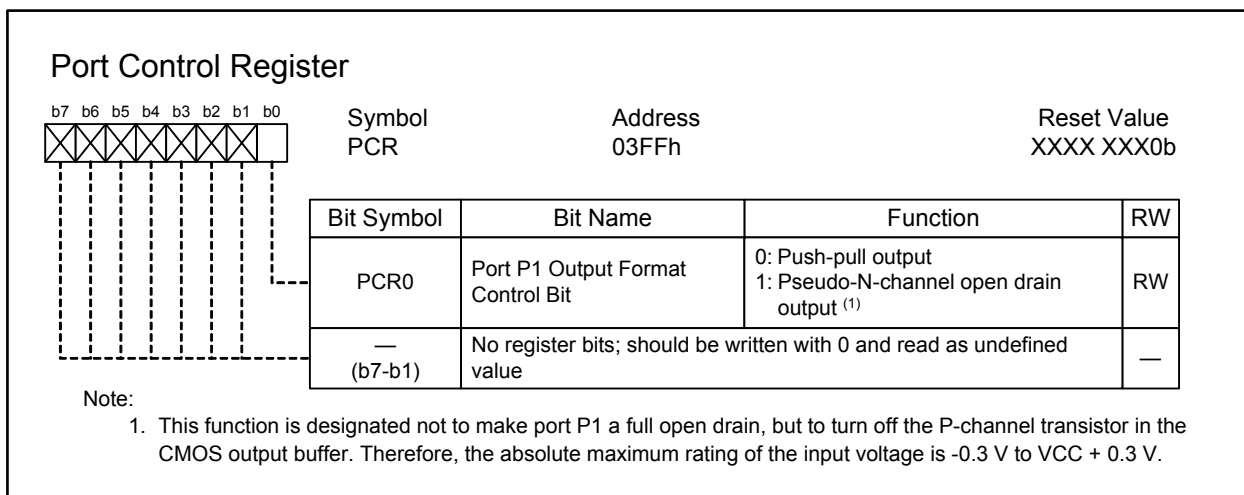


Figure 25.21 PCR Register

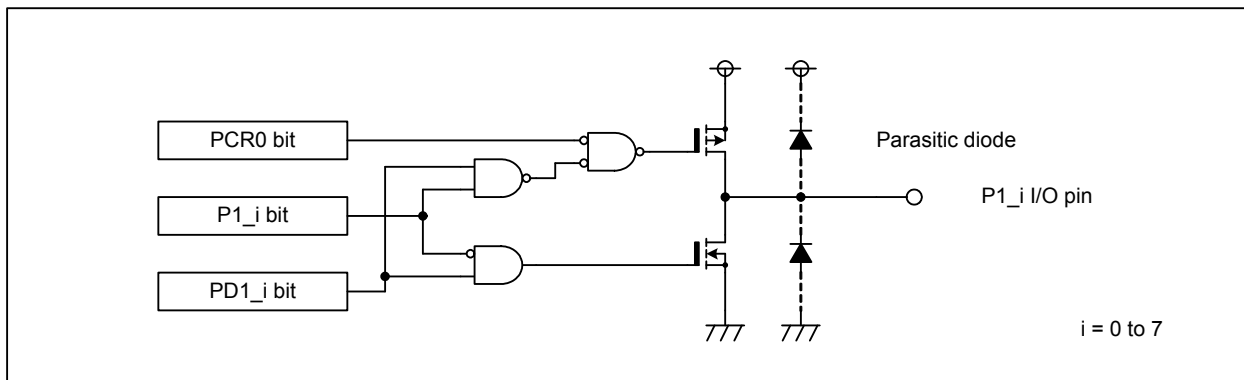


Figure 25.22 Port P1 Output Buffer Configuration

## 25.6 Configuring Unused Pins

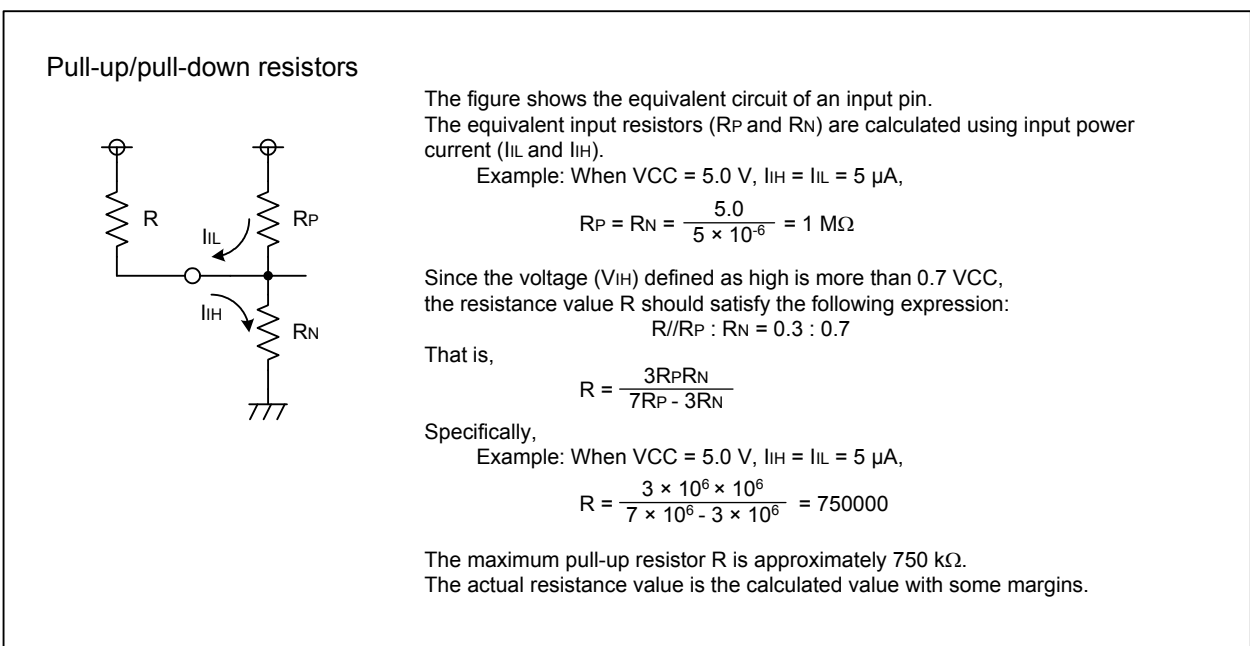
Table 25.2 and Figure 25.24 show examples of configuring unused pins on the board.

**Table 25.2 Unused Pin Configuration in Single-chip Mode (1)**

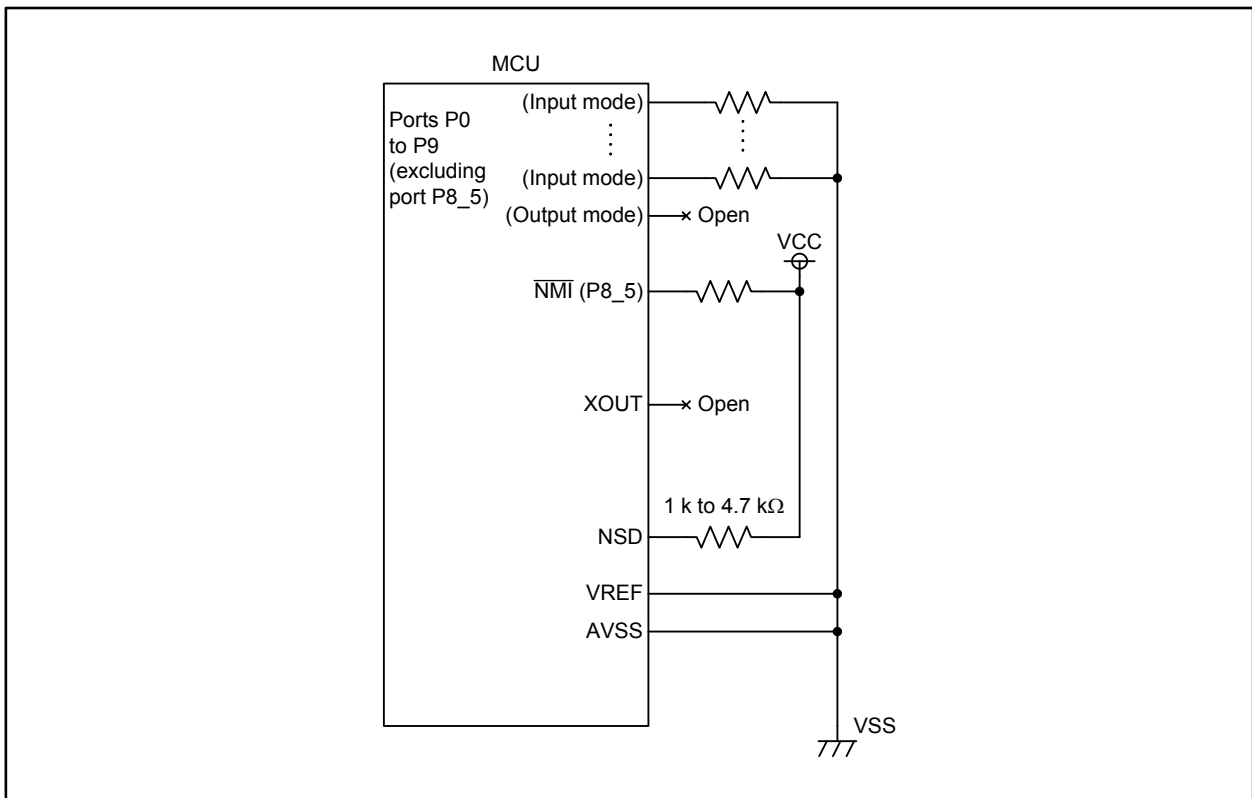
Pin Name	Setting
Ports P0 to P9 (excluding ports P8_5 and P9_1) (2)	Configure as input ports so that each pin is connected to VSS via its own resistor; (3) or configure as output ports to leave the pins open
P9_1	Connect the pin to VSS via a resistor (3)
XOUT (4)	Leave pin open
NMI (P8_5)	Connect the pin to VCC via a resistor (3)
AVCC	Connect the pin to VCC
AVSS, VREF	Connect the pin to VSS
NSD	Connect the pin to VCC via a resistor of 1 to 4.7 kΩ

Notes:

- Unused pins should be wired within 2 cm of the MCU.
- When configuring the pins as output ports to leave them open, note that ports as inputs remain unchanged from when the reset is released until the mode transition is completed. During this transition, the power supply current may increase due to an undefined voltage level of the pins. In addition, the direction register value may change due to noise or program runaway caused by the noise. To avoid these situations, reconfigure the direction register regularly by software, which may achieve higher program reliability.
- Select a resistance value that is appropriate for the system. A range from 10 to 100 kΩ is recommended
- This setting is applicable when an external clock is applied to the XIN pin.



**Figure 25.23 Pull-up/Pull-down Resistors**

**Figure 25.24 Unused Pin Configuration**

## 26. Flash Memory

### 26.1 Overview

The flash memory can be programmed in the following three modes: CPU rewrite mode, standard serial I/O mode, and parallel I/O mode.

Table 26.1 lists specifications of the flash memory and Table 26.2 shows the overview of each rewrite mode.

**Table 26.1 Flash Memory Specifications**

Item	Specification
Rewrite modes	CPU rewrite mode, standard serial I/O mode, parallel I/O mode
Structure	Block architecture. Refer to Figure 26.1
Program operation	8-byte basis
Erase operation	1-block basis
Program and erase control method	Software commands
Protection types	Lock bit protect, ROM code protect, ID code protect
Software commands	9

**Table 26.2 Flash Memory Rewrite Mode Overview**

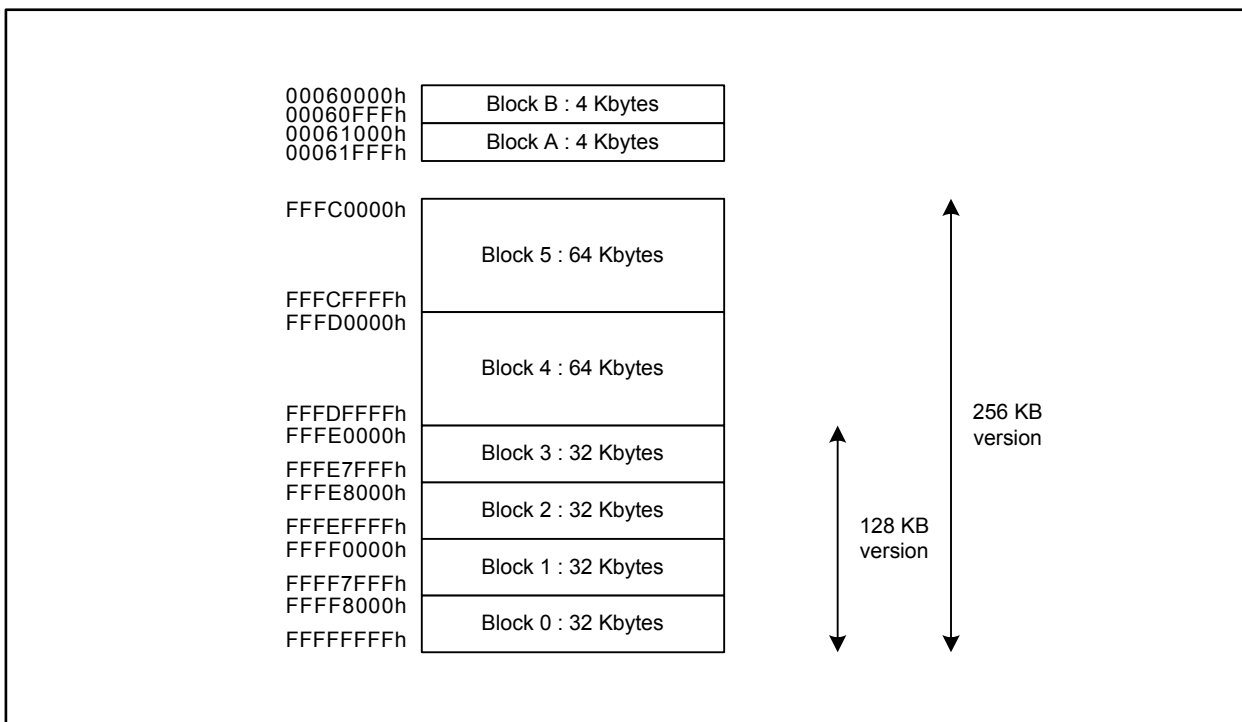
Rewrite Mode	CPU Rewrite Mode	Standard Serial I/O Mode	Parallel I/O Mode
Function	CPU executes a software command to rewrite the flash memory EW0 mode: Rewritable in areas other than the on-chip flash memory EW1 mode: Rewritable in areas other than specified blocks to be rewritten	A dedicated serial programmer rewrites the flash memory Standard serial I/O mode 1: Synchronous serial I/O selected Standard serial I/O mode 2: UART selected	A dedicated parallel programmer rewrites the flash memory
CPU operating mode	Single-chip mode	Standard serial I/O mode	Parallel I/O mode
Programmer	—	Serial programmer	Parallel programmer
On-board rewriting	Supported	Supported	Not supported

Figure 26.1 shows the on-chip flash memory structure.

The on-chip flash memory contains program area to store user programs, and data area/data flash to store the result of user programs. The program area consists of blocks 0 to 5, and data area/data flash consists of blocks A and B.

Each block can be individually protected (locked) from programming or erasing by setting the lock bit.





**Figure 26.1 On-chip Flash Memory Block Diagram**

## 26.2 Flash Memory Protection

There are three types of protection as shown in Table 26.3. Lock bit protection is intended to prevent accidental write or erase by program runaway. ROM code protection and ID code protection are intended to prevent read or write by a third party.

**Table 26.3 Protection Types and Characteristics**

Protection Type	Lock Bit Protection	ROM Code Protection	ID Code Protection
Protected operations	Erase, write	Read, write	Read, erase, write
Protection available in	CPU rewrite mode Standard serial I/O mode Parallel I/O mode	Parallel I/O mode	Standard serial I/O mode
Protection available for	Individual blocks	Entire flash memory	Entire flash memory
Protection settings	Setting 0 to the lock bit of block to be protected	Setting the protect bit of any block to 0	Writing the program which has set an ID code to specified address
Protection disabled by	Setting the LBD bit in the FMR register to 1 (lock bit protection disabled), or by erasing the blocks whose lock bits are set to 0 to permanently disable the protection	Erasing all blocks whose protect bits are set to 0	Sending a proper ID code from the serial programmer

### 26.2.1 Lock Bit Protection

This protection can be used in all three rewrite modes. When the lock bit protection is enabled, all blocks whose lock bits are set to 0 (locked) are protected against programming and erasing.

To set the lock bit to 0, the lock bit program command must be issued.

To temporarily disable the protection of all protected blocks, disable the lock bit protection itself by setting the LBD bit in the FMR1 register to 1 (lock bit protection disabled). The protection of a protected block is disabled permanently and its lock bit becomes 1 (unlocked) if the block is erased.

### 26.2.2 ROM Code Protection

This protection can only be used in parallel I/O mode. When the ROM code protection is enabled, the entire flash memory is protected against reading and writing.

To disable the protection, erase all the blocks whose protect bits are set to 0 (protected).

Each block has two protect bits. Setting any protect bit to 0 by a software command enables the protection for the entire flash memory. Table 26.4 lists protect bit addresses.

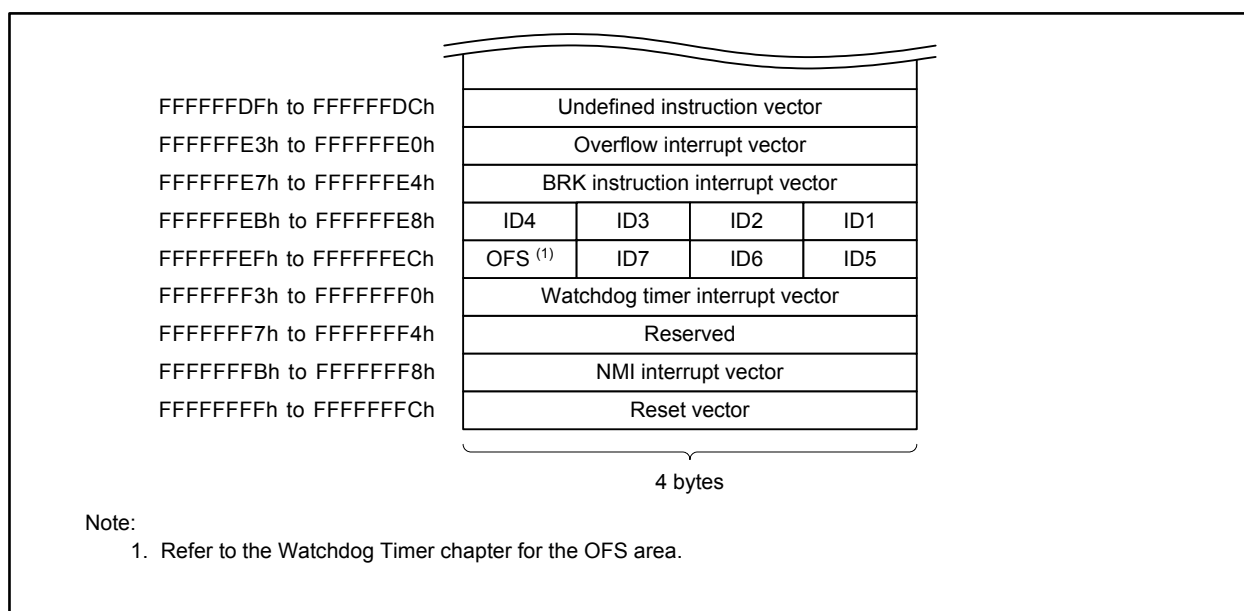
**Table 26.4 Protect Bit Addresses**

Block	Protect Bit 0	Protect Bit 1
Block B	00060100h	00060300h
Block A	00061100h	00061300h
Block 5	FFFC0100h	FFFC0300h
Block 4	FFFD0100h	FFFD0300h
Block 3	FFFE0100h	FFFE0300h
Block 2	FFFE8100h	FFFE8300h
Block 1	FFFF0100h	FFFF0300h
Block 0	FFFF8100h	FFFF8300h

### 26.2.3 ID Code Protection

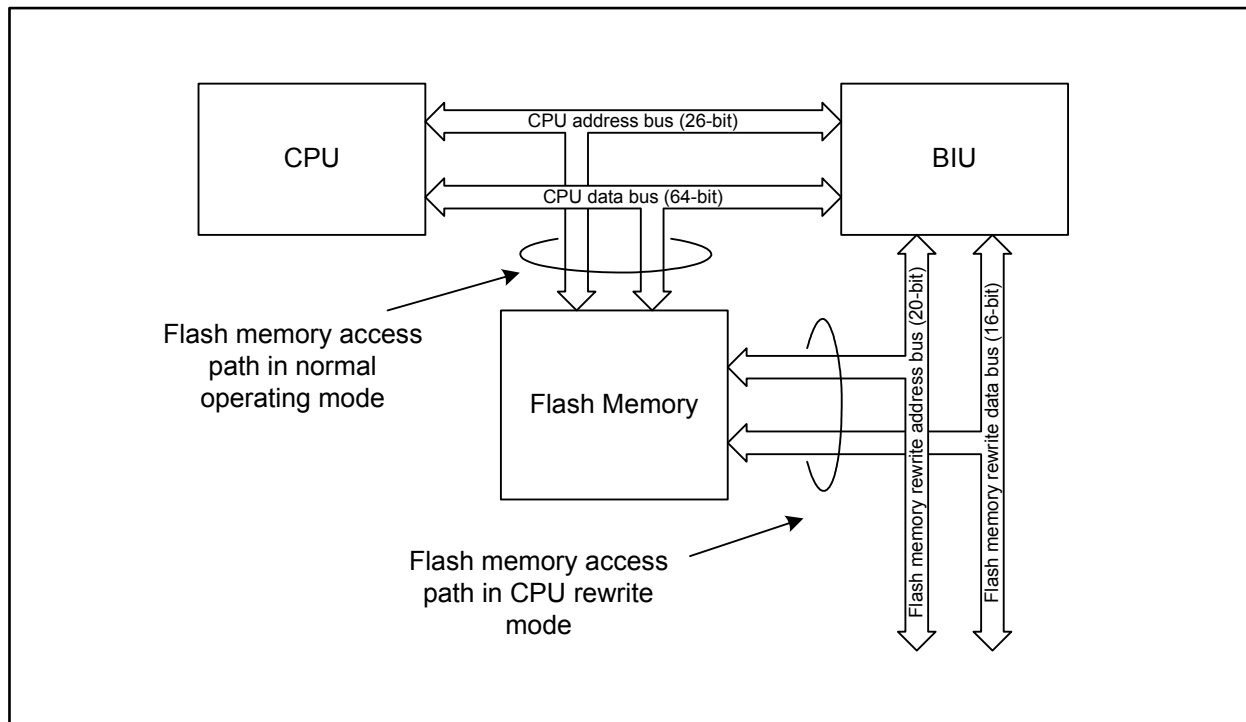
This protection can only be used in standard serial I/O mode. A command from the serial programmer is to be accepted when the 7-byte ID code sent from the serial programmer matches the ID code programmed in the flash memory. However, when the reset vector is FFFFFFFFh, the ID code check is skipped because the flash memory is considered to be blank. When the reset vector is FFFFFFFFh and the ROM code protection is enabled, only the block erase command is accepted.

The ID codes sent from the serial programmer are consecutively numbered as ID1, ID2, ..., and ID7. ID codes programmed in the flash memory, also numbered as ID1, ID2, ..., and ID7, are assigned to addresses FFFFFFFE8h, FFFFFFFE9h, ..., and FFFFFFFEh as shown in Figure 26.2. The ID code protection is enabled when a program which has an ID code set in the corresponding address is written to the flash memory.

**Figure 26.2 Addresses for ID Code Stored**

### 26.3 CPU Rewrite Mode

In CPU rewrite mode, the CPU executes software commands to rewrite the flash memory. The CPU accesses the flash memory not via the CPU buses, but via the dedicated flash memory rewrite buses (refer to Figure 26.3).



**Figure 26.3 Flash Memory Access Path in CPU Rewrite Mode**

Bus setting for flash memory rewrite should be performed by the FEBC register. Refer to 26.3.1 “Flash Memory Rewrite Bus Timing” and 28. “Electrical Characteristics” for the appropriate bus setting.

The CPU rewrite mode contains modes EW0 and EW1 as shown in Table 26.5.

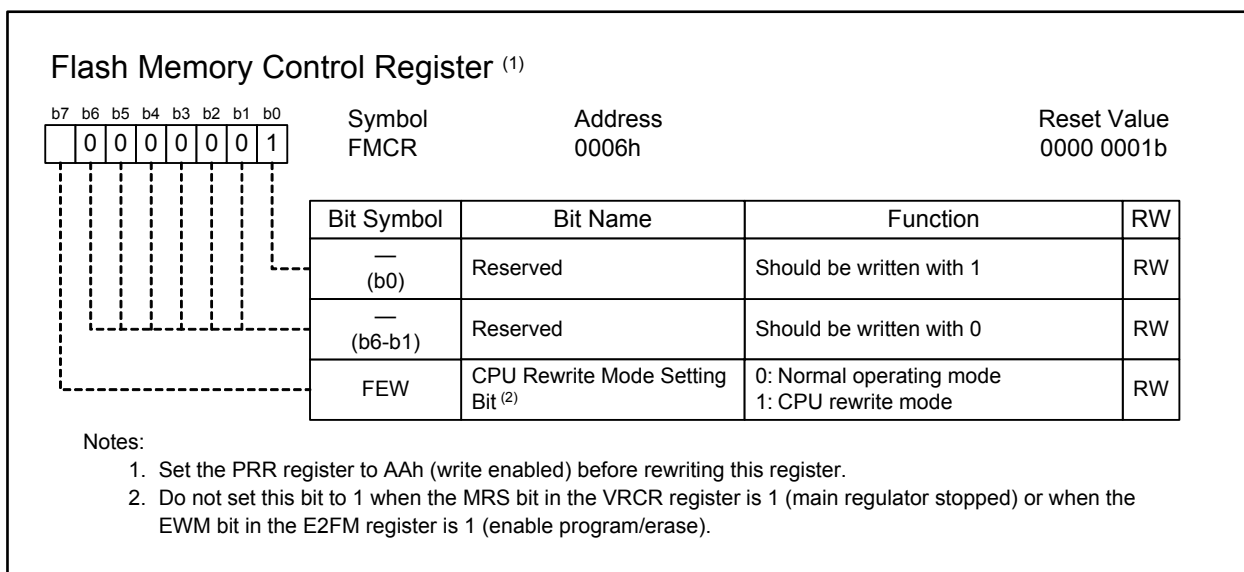
**Table 26.5 EW0 and EW1 Modes**

Item	EW0 Mode	EW1 Mode
Rewrite program executable spaces	Spaces other than the on-chip flash memory	Internal spaces other than specified blocks to be rewritten, internal RAM
Restrictions on software commands	None	<ul style="list-style-type: none"> <li>• Do not execute either the program command or the block erase command for blocks where the rewrite control programs are written to</li> <li>• Do not execute the enter read status register mode command</li> <li>• Execute the enter read lock bit status mode command in RAM</li> <li>• Execute the enter read protect bit status mode command in RAM</li> </ul>
Mode after program/erase operation	Read status register mode	Read array mode
CPU state during program/erase operation	Operating	In a hold state (I/O ports maintain the state before the command was executed)
Flash memory state detection by	<ul style="list-style-type: none"> <li>• Reading the FMSR0 register by a program</li> <li>• Executing the enter read status register mode command to read data</li> </ul>	<ul style="list-style-type: none"> <li>• Reading the FMSR0 register by a program</li> </ul>
Other restrictions	None	<ul style="list-style-type: none"> <li>• The watchdog timer is disabled in count source protect mode</li> <li>• Disable interrupts (except NMI) and DMA transfer during program/erase operation</li> </ul>

To select CPU rewrite mode, the FEW bit in the FMCR register should be set to 1. Then, EW0 mode/EW1 mode can be selected by setting the EWM bit in the FMR0 register.

Registers FMCR and FMR0 are protected by registers PRR and FPR0, respectively.

Figures 26.4 to 26.11 show associated registers.

**Figure 26.4 FMCR Register**

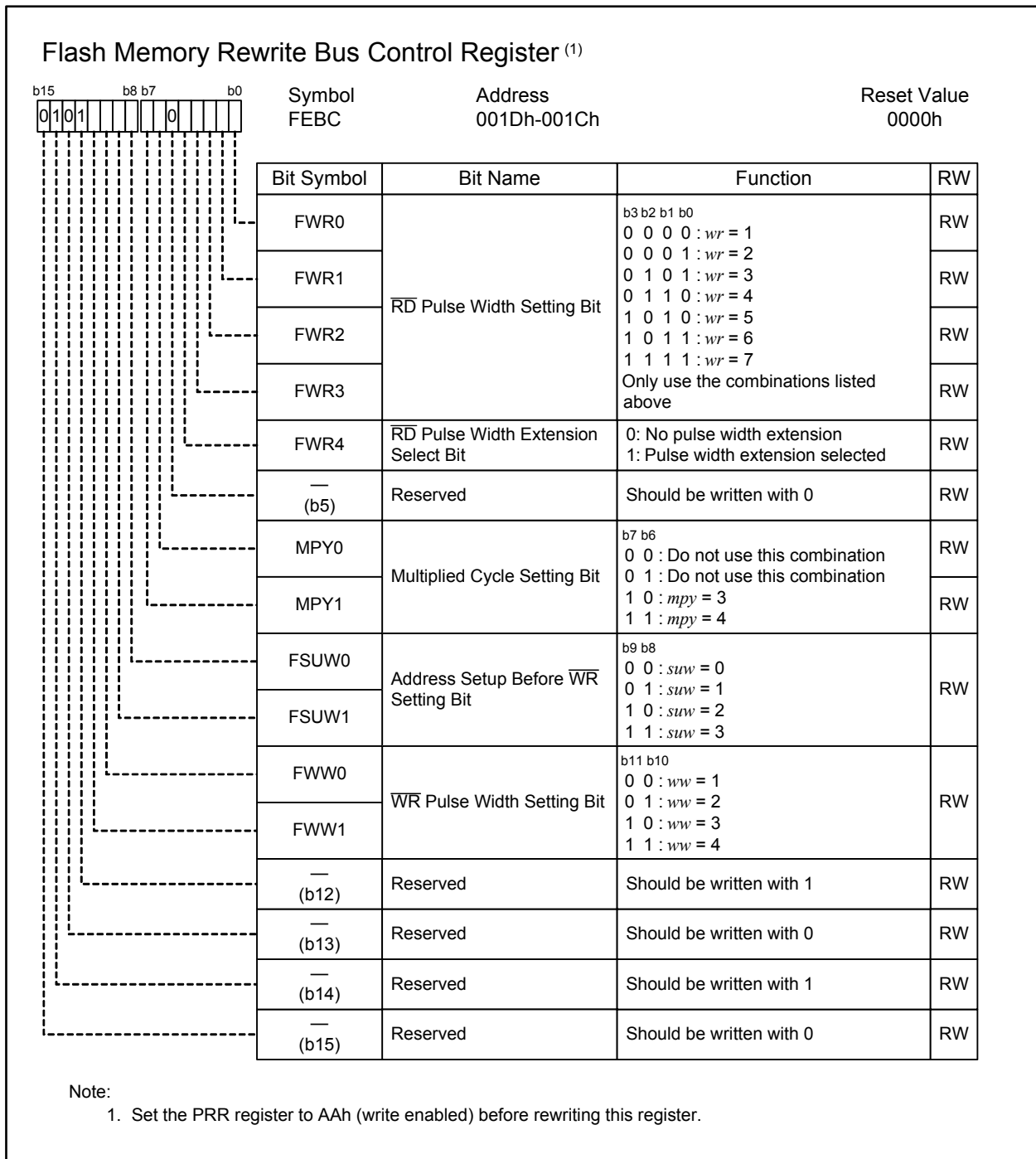


Figure 26.5 FEBC Register

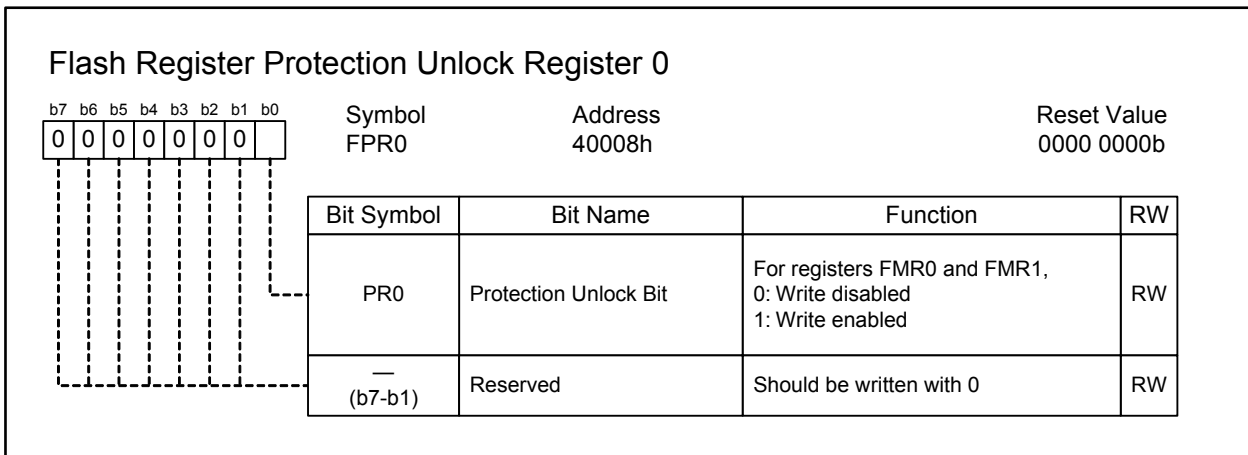


Figure 26.6 FPR0 Register

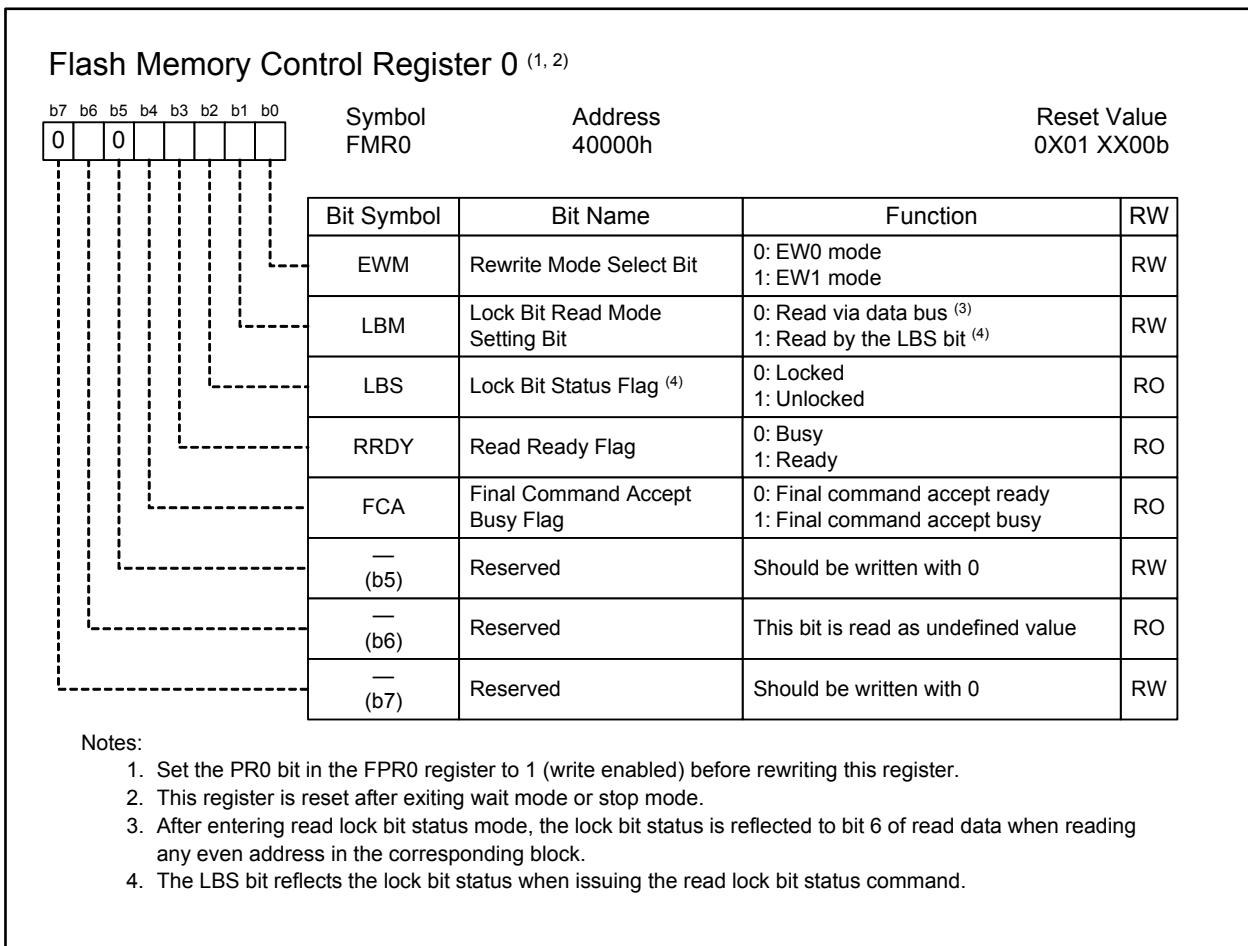


Figure 26.7 FMR0 Register



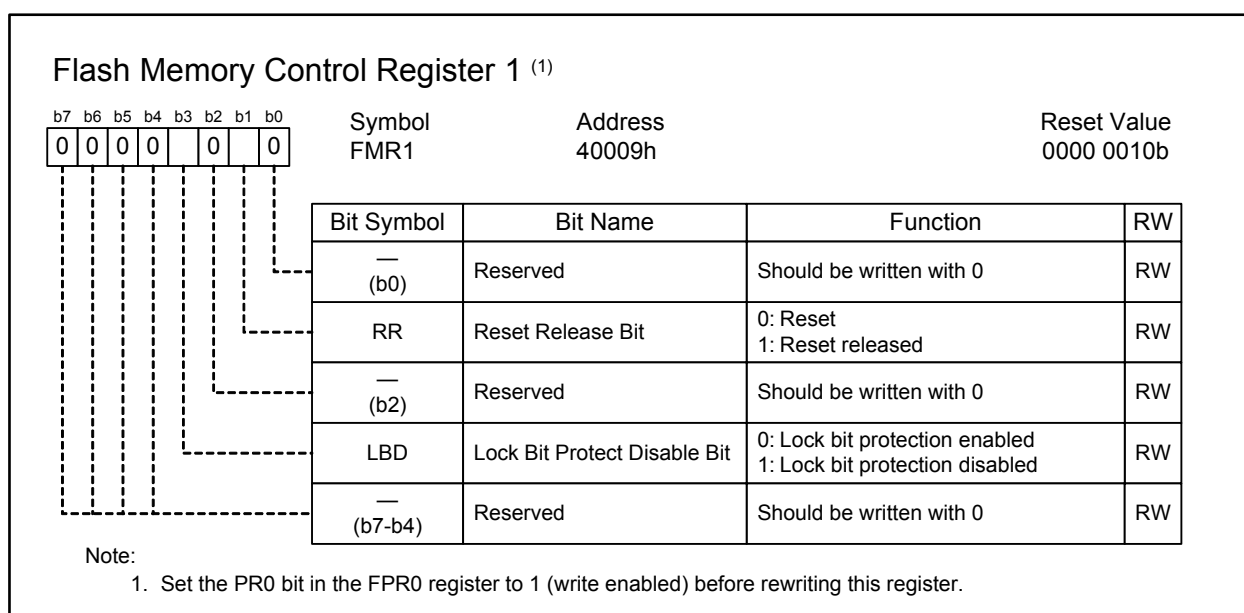


Figure 26.8 FMR1 Register

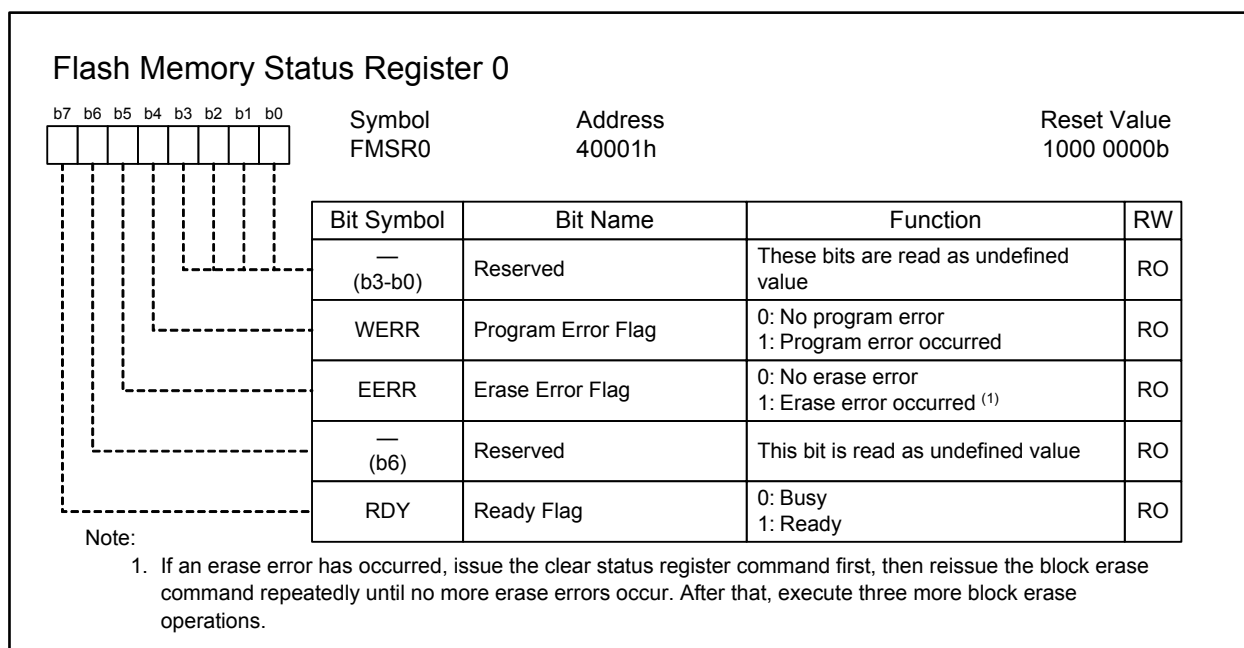


Figure 26.9 FMSR0 Register

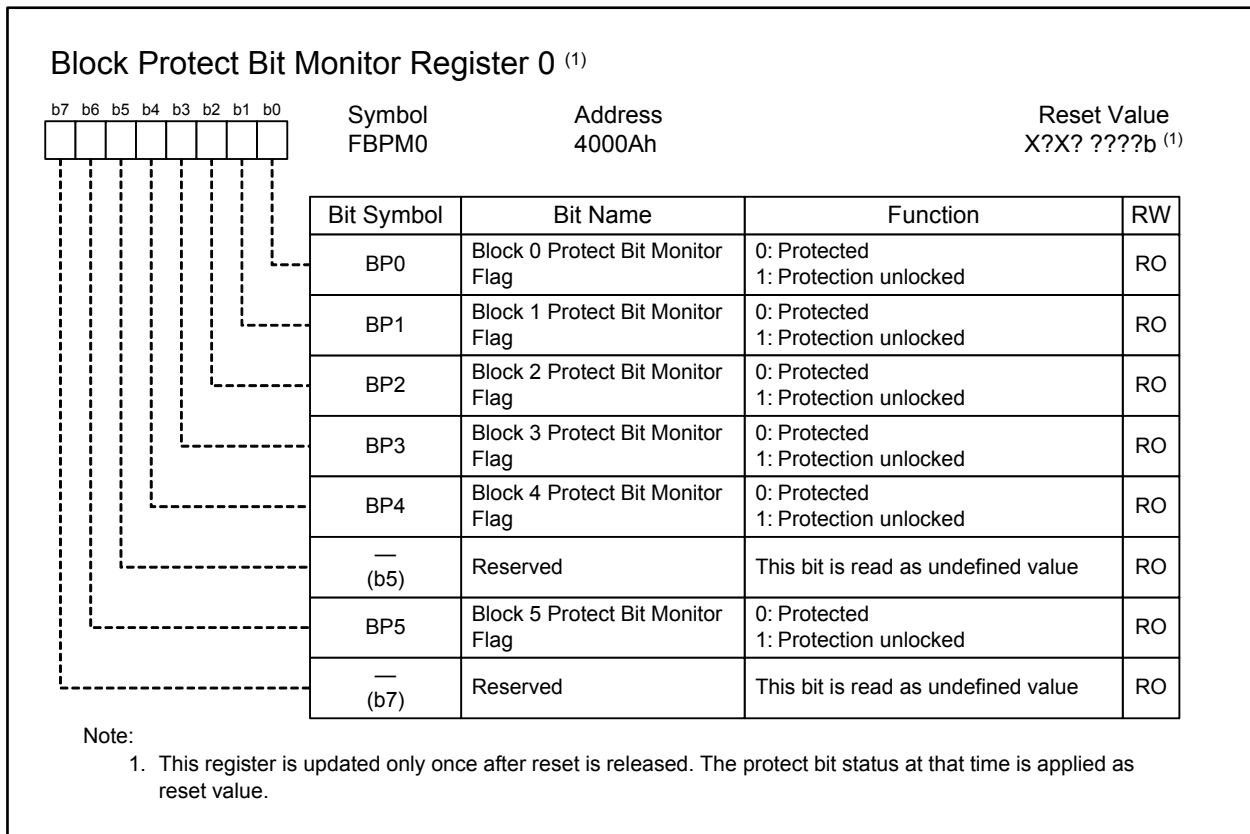


Figure 26.10 FBPM0 Register

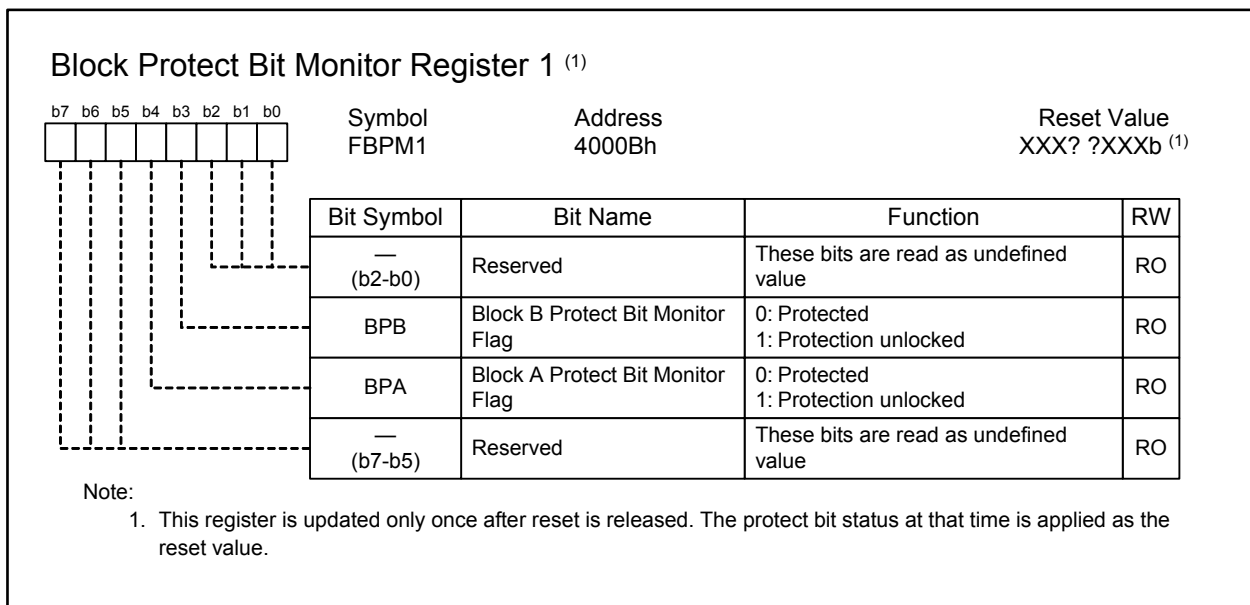


Figure 26.11 FBPM1 Register

### 26.3.1 Flash Memory Rewrite Bus Timing

The bus setting for the flash memory rewrite is performed by setting the FEBC register. This section specifically describes the setting of the FEBC register.

The reference clock is the base clock set with bits BCD1 and BCD0 in the CCR register. Time duration including  $t_{su}$ ,  $t_w$ ,  $t_c$ , and  $t_h$  are specified by the number of base clock cycles.

Tables 26.6 to 26.8 show the correlation of the read cycle and setting of bits MPY1, MPY0, and FWR4 to FWR0, according to peripheral bus clock divide ratios. Tables 26.9 to 26.11 show the correlation of the write cycle and setting of bits MPY1, MPY0, FSUW1, FSUW0, FWW1, and FWW0. Associated read/write timings are illustrated in Figures 26.12 and 26.13, respectively.

Read/write cycle timing is selected from the tables below to meet the timing requirements in the CPU rewrite mode described in the electrical characteristics.

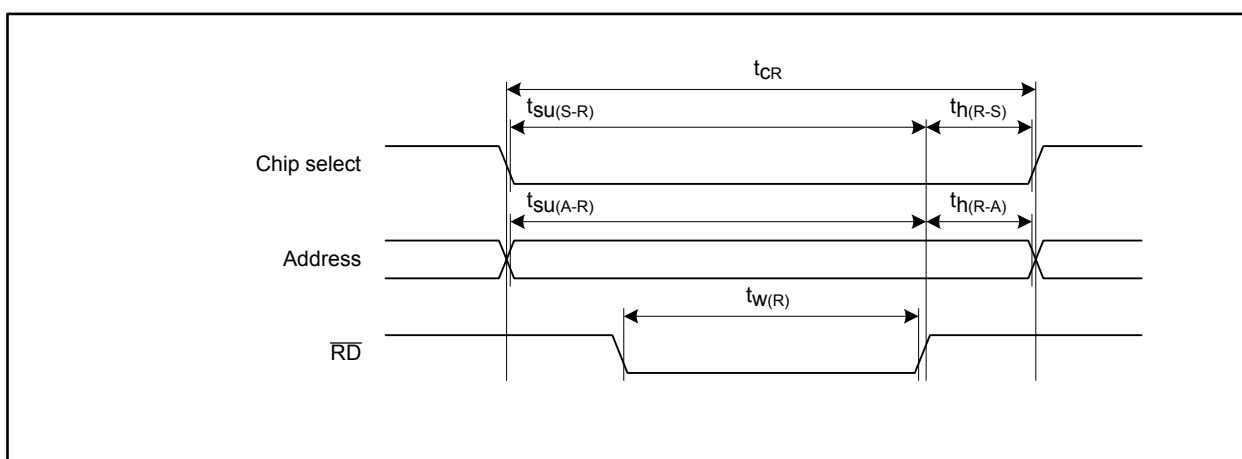


Figure 26.12 Read Timing

Table 26.6 Read Cycle and Bit Settings: MPY1, MPY0, and FWR4 to FWR0, When Peripheral Bus Clock is Divided by 2 (unit: cycles)

FWR3 to FWR0 Bit Settings		FWR4 Bit Settings	MPY1 and MPY0 Bit Settings							
			10b				11b			
			$mpy = 3$				$mpy = 4$			
			$t_{su(S-R)}$ , $t_{su(A-R)}$	$t_w(R)$	$t_{CR}$	$t_{th(R-S)}$ , $t_{th(R-A)}$	$t_{su(S-R)}$ , $t_{su(A-R)}$	$t_w(R)$	$t_{CR}$	$t_{th(R-S)}$ , $t_{th(R-A)}$
0000b	$wr = 1$	0	4	3	4	0	6	5	6	0
		1	6	5	6	0	6	5	6	0
0001b	$wr = 2$	0	8	7	8	0	10	9	10	0
		1	8	7	8	0	10	9	10	0
0101b	$wr = 3$	0	10	9	10	0	14	13	14	0
		1	12	11	12	0	14	13	14	0
0110b	$wr = 4$	0	14	13	14	0	18	17	18	0
		1	14	13	14	0	18	17	18	0
1010b	$wr = 5$	0	16	15	16	0	22	21	22	0
		1	18	17	18	0	22	21	22	0
1011b	$wr = 6$	0	20	19	20	0	26	25	26	0
		1	20	19	20	0	26	25	26	0
1111b	$wr = 7$	0	22	21	22	0	30	29	30	0
		1	24	23	24	0	30	29	30	0

**Table 26.7 Read Cycle and Bit Settings: MPY1, MPY0, and FWR4 to FWR0, When Peripheral Bus Clock is Divided by 3 (unit: cycles)**

FWR3 to FWR0 Bit Settings		FWR4 Bit Settings	MPY1 and MPY0 Bit Settings							
			10b				11b			
			<i>mpy = 3</i>				<i>mpy = 4</i>			
			tsu(S-R), tsu(A-R)	tw(R)	tCR	th(R-S), th(R-A)	tsu(S-R), tsu(A-R)	tw(R)	tCR	th(R-S), th(R-A)
0000b	<i>wr = 1</i>	0	6	4.5	6	0	6	4.5	6	0
		1	6	4.5	6	0	6	4.5	6	0
0001b	<i>wr = 2</i>	0	9	7.5	9	0	9	7.5	9	0
		1	9	7.5	9	0	12	10.5	12	0
0101b	<i>wr = 3</i>	0	12	10.5	12	0	15	13.5	15	0
		1	12	10.5	12	0	15	13.5	15	0
0110b	<i>wr = 4</i>	0	15	13.5	15	0	18	16.5	18	0
		1	15	13.5	15	0	18	16.5	18	0
1010b	<i>wr = 5</i>	0	18	16.5	18	0	21	19.5	21	0
		1	18	16.5	18	0	24	22.5	24	0
1011b	<i>wr = 6</i>	0	21	19.5	21	0	27	25.5	27	0
		1	21	19.5	21	0	27	25.5	27	0
1111b	<i>wr = 7</i>	0	24	22.5	24	0	30	28.5	30	0
		1	24	22.5	24	0	30	28.5	30	0

**Table 26.8 Read Cycle and Bit Settings: MPY1, MPY0, and FWR4 to FWR0, When Peripheral Bus Clock is Divided by 4 (unit: cycles)**

FWR3 to FWR0 Bit Settings		FWR4 Bit Settings	MPY1 and MPY0 Bit Settings							
			10b				11b			
			<i>mpy = 3</i>				<i>mpy = 4</i>			
			tsu(S-R), tsu(A-R)	tw(R)	tCR	th(R-S), th(R-A)	tsu(S-R), tsu(A-R)	tw(R)	tCR	th(R-S), th(R-A)
0000b	<i>wr = 1</i>	0	4	2	4	0	8	6	8	0
		1	8	6	8	0	8	6	8	0
0001b	<i>wr = 2</i>	0	8	6	8	0	12	10	12	0
		1	8	6	8	0	12	10	12	0
0101b	<i>wr = 3</i>	0	12	10	12	0	16	14	16	0
		1	12	10	12	0	16	14	16	0
0110b	<i>wr = 4</i>	0	16	14	16	0	20	18	20	0
		1	16	14	16	0	20	18	20	0
1010b	<i>wr = 5</i>	0	16	14	16	0	24	22	24	0
		1	20	18	20	0	24	22	24	0
1011b	<i>wr = 6</i>	0	20	18	20	0	28	26	28	0
		1	20	18	20	0	28	26	28	0
1111b	<i>wr = 7</i>	0	24	22	24	0	32	30	32	0
		1	24	22	24	0	32	30	32	0

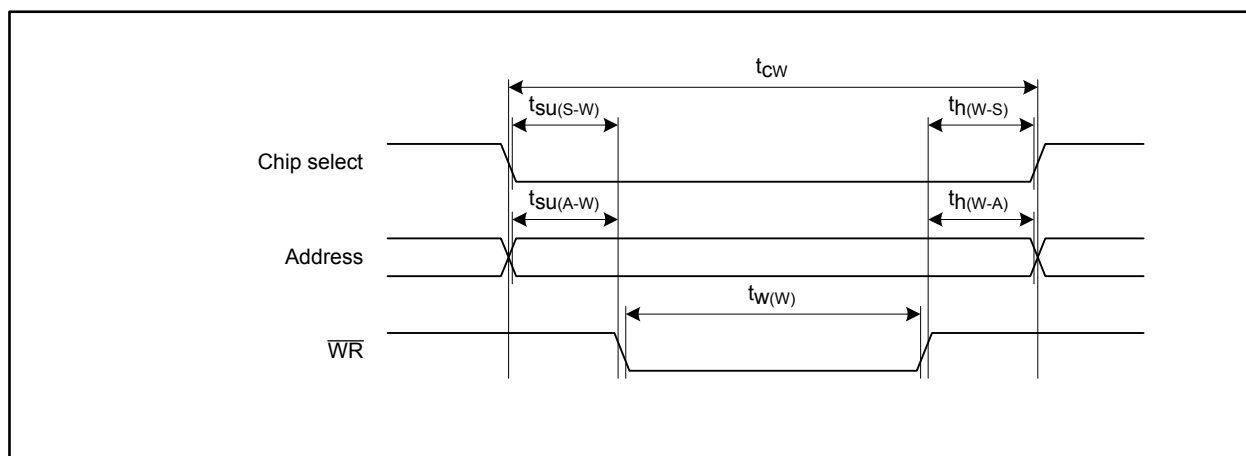


Figure 26.13 Write Timing

Table 26.9 Write Cycle and Bit Settings: MPY1, MPY0, FSUW1, FSUW0, FWW1, and FWW0, When Peripheral Bus Clock is Divided by 2 (unit: cycles)

FSUW1 and FSUW0 Bit Settings	FWW1 and FWW0 Bit Settings	MPY1 and MPY0 Bit Settings									
		10b					11b				
		<i>mpy</i> = 3					<i>mpy</i> = 4				
		tsu(S-W), tsu(A-W)	tw(W)	tcw	th(W-S), th(W-A)	tsu(S-W), tsu(A-W)	tw(W)	tcw	th(W-S), th(W-A)		
00b	<i>suw</i> = 0	00b	<i>ww</i> = 1	1	3	6	2	1	4	6	1
		01b	<i>ww</i> = 2	1	6	8	1	1	8	10	1
		10b	<i>ww</i> = 3	1	9	12	2	1	12	14	1
		11b	<i>ww</i> = 4	1	12	14	1	1	16	18	1
01b	<i>suw</i> = 1	00b	<i>ww</i> = 1	4	3	8	1	5	4	10	1
		01b	<i>ww</i> = 2	4	6	12	2	5	8	14	1
		10b	<i>ww</i> = 3	4	9	14	1	5	12	18	1
		11b	<i>ww</i> = 4	4	12	18	2	5	16	22	1
10b	<i>suw</i> = 2	00b	<i>ww</i> = 1	7	3	12	2	9	4	14	1
		01b	<i>ww</i> = 2	7	6	14	1	9	8	18	1
		10b	<i>ww</i> = 3	7	9	18	2	9	12	22	1
		11b	<i>ww</i> = 4	7	12	20	1	9	16	26	1
11b	<i>suw</i> = 3	00b	<i>ww</i> = 1	10	3	14	1	13	4	18	1
		01b	<i>ww</i> = 2	10	6	18	2	13	8	22	1
		10b	<i>ww</i> = 3	10	9	20	1	13	12	26	1
		11b	<i>ww</i> = 4	10	12	24	2	13	16	30	1

**Table 26.10 Write Cycle and Bit Settings: MPY1, MPY0, FSUW1, FSUW0, FWW1, and FWW0, When Peripheral Bus Clock is Divided by 3 (unit: cycles)**

FSUW1 and FSUW0 Bit Settings		FWW1 and FWW0 Bit Settings		MPY1 and MPY0 Bit Settings							
				10b				11b			
				<i>mpy = 3</i>				<i>mpy = 4</i>			
				tsu(S-W), tsu(A-W)	tw(W)	tcw	th(W-S), th(W-A)	tsu(S-W), tsu(A-W)	tw(W)	tcw	th(W-S), th(W-A)
00b	<i>suw = 0</i>	00b	<i>ww = 1</i>	1	3	6	2	1	4	6	1
		01b	<i>ww = 2</i>	1	6	9	2	1	8	12	3
		10b	<i>ww = 3</i>	1	9	12	2	1	12	15	2
		11b	<i>ww = 4</i>	1	12	15	2	1	16	18	1
01b	<i>suw = 1</i>	00b	<i>ww = 1</i>	4	3	9	2	6	3	12	3
		01b	<i>ww = 2</i>	4	6	12	2	6	7	15	2
		10b	<i>ww = 3</i>	4	9	15	2	6	11	18	1
		11b	<i>ww = 4</i>	4	12	18	2	6	15	24	3
10b	<i>suw = 2</i>	00b	<i>ww = 1</i>	7	3	12	2	9	4	15	2
		01b	<i>ww = 2</i>	7	6	15	2	9	8	18	1
		10b	<i>ww = 3</i>	7	9	18	2	9	12	24	3
		11b	<i>ww = 4</i>	7	12	21	2	9	16	27	2
11b	<i>suw = 3</i>	00b	<i>ww = 1</i>	10	3	15	2	13	4	18	1
		01b	<i>ww = 2</i>	10	6	18	2	13	8	24	3
		10b	<i>ww = 3</i>	10	9	21	2	13	12	27	2
		11b	<i>ww = 4</i>	10	12	24	2	13	16	30	1

**Table 26.11 Write Cycle and Bit Settings: MPY1, MPY0, FSUW1, FSUW0, FWW1, and FWW0, When Peripheral Bus Clock is Divided by 4 (unit: cycles)**

FSUW1 and FSUW0 Bit Settings		FWW1 and FWW0 Bit Settings		MPY1 and MPY0 Bit Settings							
				10b				11b			
				<i>mpy = 3</i>				<i>mpy = 4</i>			
				tsu(S-W), tsu(A-W)	tw(W)	tcw	th(W-S), th(W-A)	tsu(S-W), tsu(A-W)	tw(W)	tcw	th(W-S), th(W-A)
00b	<i>suw = 0</i>	00b	<i>ww = 1</i>	1	3	8	4	1	4	8	3
		01b	<i>ww = 2</i>	1	6	8	1	1	8	12	3
		10b	<i>ww = 3</i>	1	9	12	2	1	12	16	3
		11b	<i>ww = 4</i>	1	12	16	3	1	16	20	3
01b	<i>suw = 1</i>	00b	<i>ww = 1</i>	4	3	8	1	5	4	12	3
		01b	<i>ww = 2</i>	4	6	12	2	5	8	16	3
		10b	<i>ww = 3</i>	4	9	16	3	5	12	20	3
		11b	<i>ww = 4</i>	4	12	20	4	5	16	24	3
10b	<i>suw = 2</i>	00b	<i>ww = 1</i>	8	2	12	2	9	4	16	3
		01b	<i>ww = 2</i>	8	5	16	3	9	8	20	3
		10b	<i>ww = 3</i>	8	8	20	4	9	12	24	3
		11b	<i>ww = 4</i>	8	11	20	1	9	16	28	3
11b	<i>suw = 3</i>	00b	<i>ww = 1</i>	10	3	16	3	13	4	20	3
		01b	<i>ww = 2</i>	10	6	20	4	13	8	24	3
		10b	<i>ww = 3</i>	10	9	20	1	13	12	28	3
		11b	<i>ww = 4</i>	10	12	24	2	13	16	32	3

### 26.3.2 Software Commands

In CPU rewrite mode, software commands enable rewrite and erase operations for the flash memory. Writing commands and reading/writing data should be performed in 16-bit units.

Table 26.12 lists the software commands.

**Table 26.12 Software Commands**

Command	First Command Cycle		Second Command Cycle	
	Address	Data	Address	Data
Enter read array mode	FFFFFF800h	00FFh	—	—
Enter read status register mode <sup>(1)</sup>	FFFFFF800h	0070h	—	—
Clear status register	FFFFFF800h	0050h	—	—
Program <sup>(2)</sup>	FFFFFF800h	0043h	WA	WD
Block erase	FFFFFF800h	0020h	BA	00D0h
Lock bit program	FFFFFF800h	0077h	BA	00D0h
Read lock bit status	FFFFFF800h	0071h	BA	00D0h
Enter read lock bit status mode <sup>(3)</sup>	FFFFFF800h	0071h	—	—
Protect bit program	FFFFFF800h	0067h	PBA	00D0h
Enter read protect bit status mode <sup>(3)</sup>	FFFFFF800h	0061h	—	—

WA: Even address to be written

WD: 16-bit data to be written

BA: Even address within a specific block

PBA: Protect bit address (refer to Table 26.4)

**Notes:**

1. This command cannot be executed in EW1 mode.
2. The program is performed in 64-bit (4-word) units. A sequence of commands consists of commands from the second to fifth. The upper 29 bits of the address WA should be fixed and the lower 3 bits of respective commands from the second to fifth should be set to 000b, 010b, 100b, and 110b for the addresses 0h, 2h, 4h, and 6h, or 8h, Ah, Ch, and Eh.
3. This command should be executed in RAM.

### 26.3.3 Mode Transition

CPU rewrite mode supports four flash memory operating modes:

- Read array mode
- Read status register mode
- Read lock bit status mode
- Read protect bit status mode

When reading the flash memory in these modes, the memory content, the status register content, the state of the lock bit in the read block, and the state of the protect bit are individually read. Details are listed in Tables 26.13 to 26.15.

**Table 26.13 Status Register**

Bit	Bit Symbol	Bit Name	Definition	
			0	1
b15-b8	—	Disabled bit	—	—
b7	SR7	Sequencer status	BUSY	READY
b6	—	Reserved bit	—	—
b5	SR5	Erase status	Successfully completed	Error
b4	SR4	Program status	Successfully completed	Error
b3	—	Reserved bit	—	—
b2	—	Reserved bit	—	—
b1	—	Reserved bit	—	—
b0	—	Reserved bit	—	—

**Table 26.14 Lock Bit Status**

Bit	Bit Symbol	Bit Name	Definition	
			0	1
b15-b7	—	Disabled bit	—	—
b6	LBS	Lock bit status	Locked	Unlocked
b5-b0	—	Disabled bit	—	—

**Table 26.15 Protect Bit Status**

Bit	Bit Symbol	Bit Name	Definition	
			0	1
b15-b7	—	Disabled bit	—	—
b6	PBS	Protect bit status	Protected	Unprotected
b5-b0	—	Disabled bit	—	—

In these operating modes, program or erase operation can be performed by software commands. After an operation is completed, the flash memory module automatically enters read array mode (in EW1 mode) or read status register mode (in EW0 mode).



### 26.3.4 Issuing Software Commands

This section describes how to issue software commands.

These commands should be issued while the RDY bit in the FMSR0 register is 1 (ready).

#### 26.3.4.1 Enter Read Array Mode Command

Execute this command to enter read array mode.

When 00FFh is written to address FFFF800h, the flash memory enters read array mode. In this mode, the content stored to a given address in memory can be read.

In EW1 mode, the flash memory is always in read array mode.

#### 26.3.4.2 Enter Read Status Register Mode

Execute this command to enter read status register mode.

When 0070h is written to address FFFF800h, the status register content is read in any address of the flash memory.

Do not issue this command in EW1 mode.

#### 26.3.4.3 Clear Status Register

Execute this command to reset the status register in the flash memory.

When 0050h is written to address FFFF800h, bits SR5 and SR4 in the status register become 0 (successfully completed) (refer to Table 26.13). Consequently, bits EERR and WERR in the FMSR0 register become 0 (no errors).

### 26.3.4.4 Program Command

Execute this command to program the flash memory in 8-byte (4-word) units.

To start automatic programming (program and program-verify operations), write 0043h to address FFFF800h, then write data to addresses  $8n + 0$  to  $8n + 6$ . Verify that the FCA bit in the FMR0 register is 0 just before executing the final command.

To monitor the automatic program operation, read the RDY bit in the FMSR0 register. This bit becomes 0 (busy) when the operation is in progress and 1 (ready) when the operation is completed.

The operation result can be verified by the WERR bit in the FMSR0 register (refer to 26.3.5 “Status Check”).

Do not write additional data to an address that is already programmed.

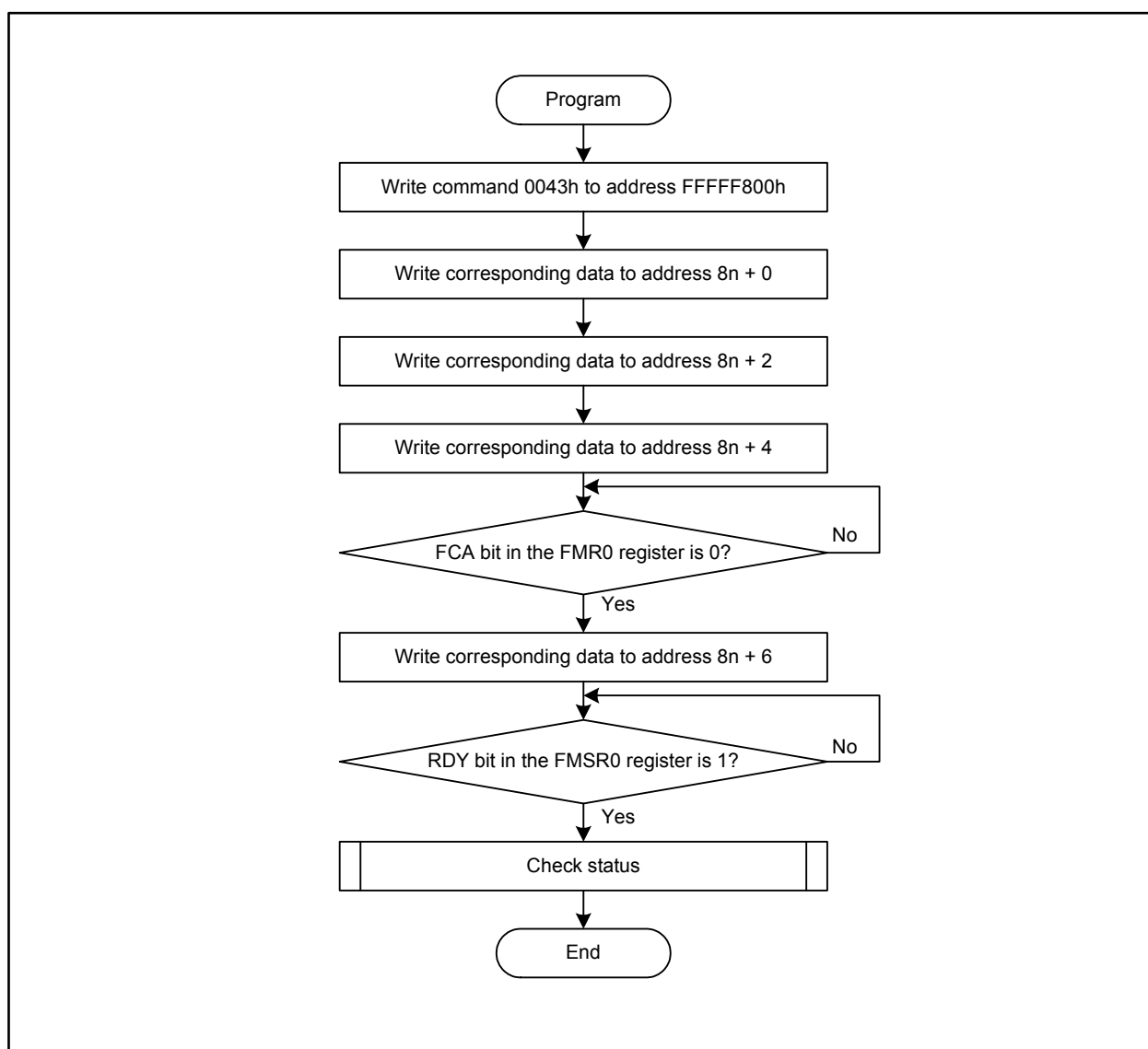


Figure 26.14 Program Command Execution Flowchart

### 26.3.4.5 Block Erase Command

Execute this command to erase a specified block in the flash memory.

To start automatic erasing of a specified block (erase and erase-verify operations), write 0020h to address FFFFF800h, verify that the FCA bit in the FMR0 register is 0, then write 00D0h to an even address in the corresponding block.

To monitor the automatic erase operation, read the RDY bit in the FMSR0 register. This bit becomes 0 (busy) when the operation is in progress and 1 (ready) when the operation is completed.

The operation result can be verified by the EERR bit in the FMSR0 register (refer to 26.3.5 “Status Check”).

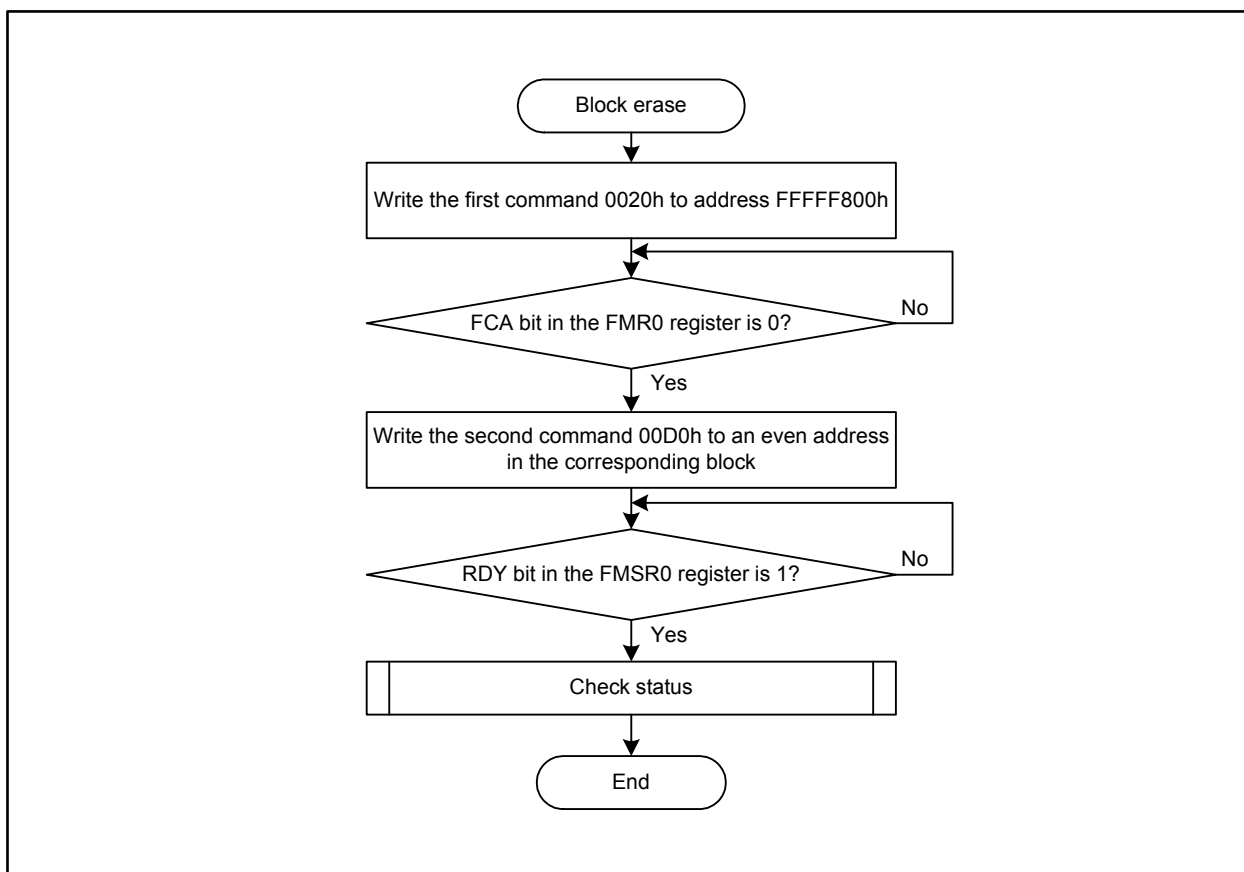


Figure 26.15 Block Erase Command Execution Flowchart

### 26.3.4.6 Lock Bit Program Command

Execute this command to lock a specified block in the flash memory.

To lock the block, write 0077h to address FFFF800h, verify that the FCA bit in the FMR0 register is 0, then write 00D0h to an even address in the corresponding block. Then the lock bit of the block becomes 0 (locked).

To monitor the lock bit program, read the RDY bit in the FMSR0 register. This bit becomes 0 (busy) when the operation is in progress and 1 (ready) when the operation is completed.

The state of the lock bit can be verified by the read lock bit status command if the LBM bit in the FMR0 register is 1 (read by the LBS bit) (refer to 26.3.4.7 "Read Lock Bit Status Command"). If the LBM bit is 0 (read via data bus), enter read lock bit status mode (refer to 26.3.4.8 "Enter Read Lock Bit Status Mode Command").

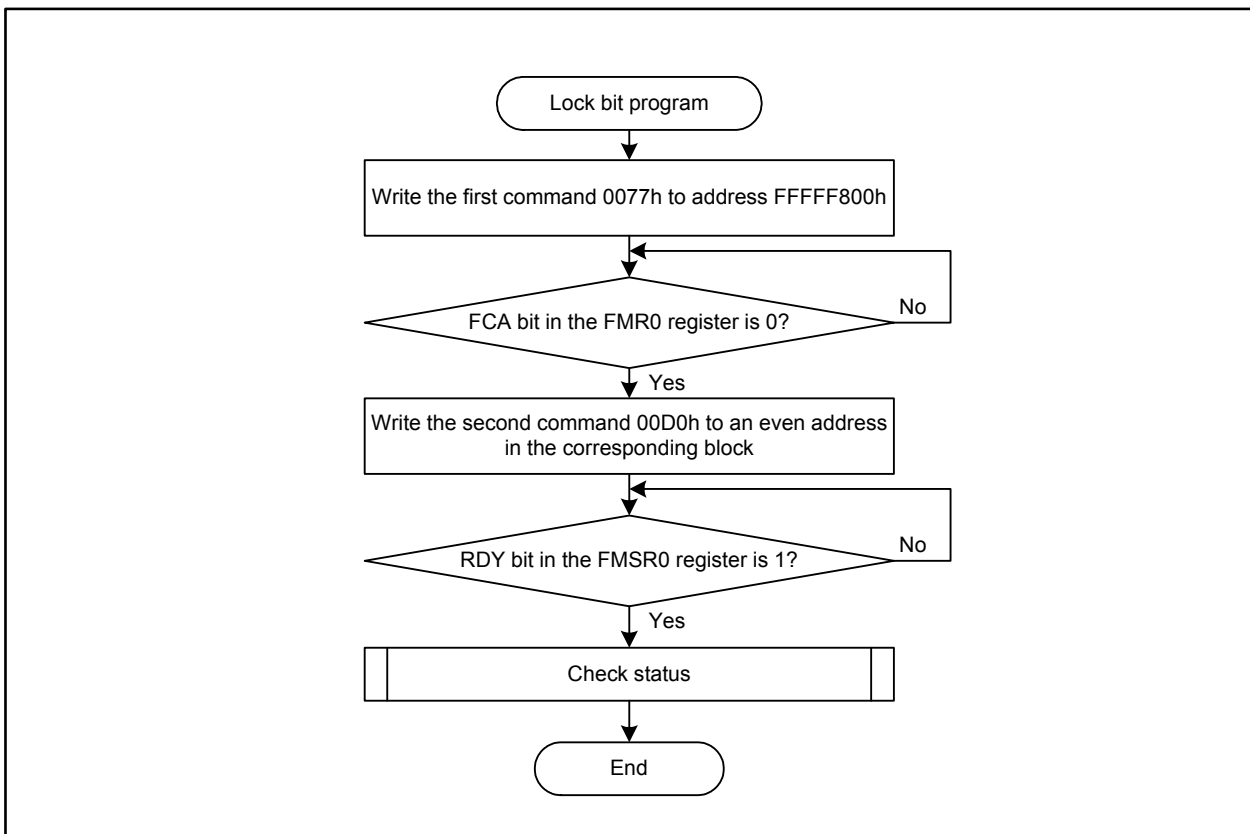


Figure 26.16 Lock Bit Program Command Execution Flowchart

### 26.3.4.7 Read Lock Bit Status Command

Execute this command to verify if a specified block in the flash memory is locked. This command can be used when the LBM bit in the FMR0 register is 1 (read by the LBS bit).

The LBS bit in the FMSR0 register reflects the lock bit status of the specified block when the following is performed: first write 0071h to address FFFFF800h and verify that the FCA bit in the FMR0 register becomes 0. Then write 00D0h to an even address of the corresponding block.

Read the LBS bit after the RDY bit in the FMSR0 register becomes 1 (ready).

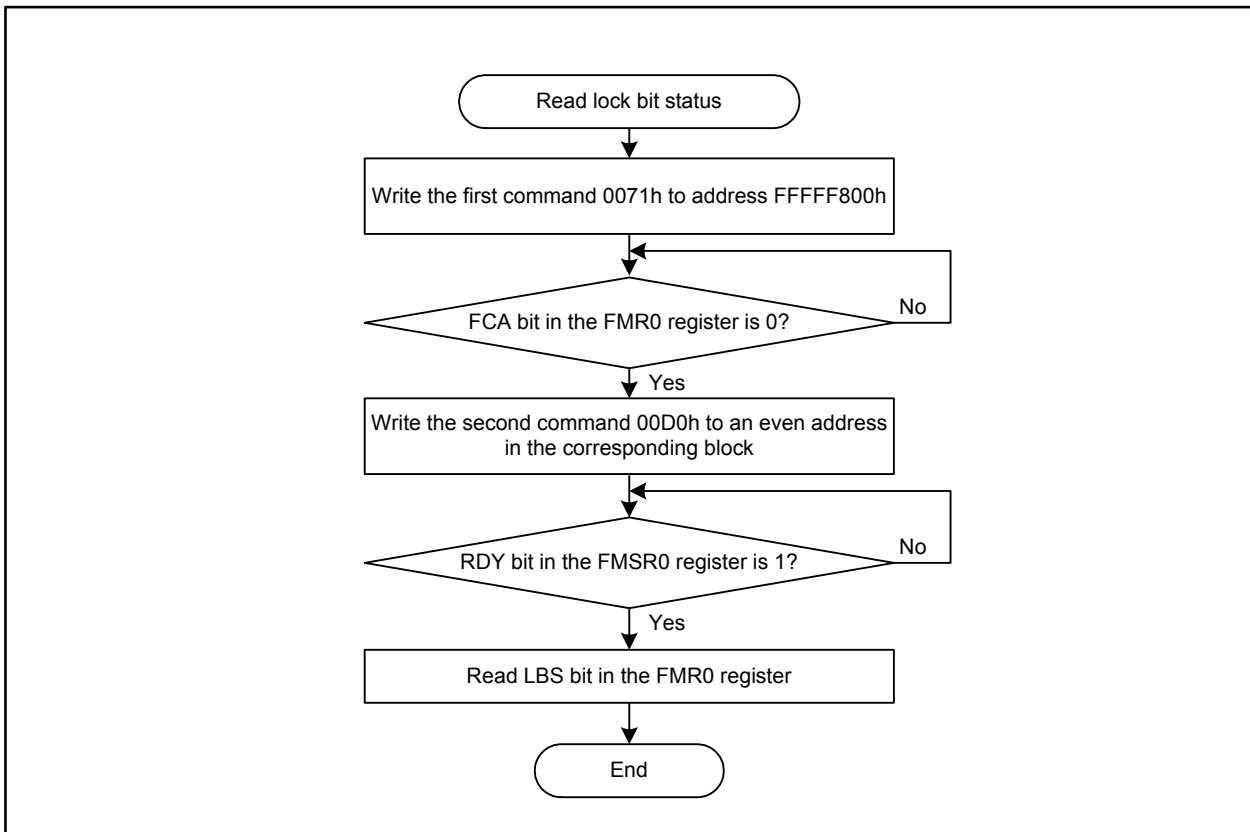


Figure 26.17 Read Lock Bit Status Command Execution Flowchart

### 26.3.4.8 Enter Read Lock Bit Status Mode Command

Execute this command to enter read lock bit status mode. This command is enabled when the LBM bit in the FMR0 register is 0 (read via data bus).

To read the lock bit status of the read block, write 0071h to address FFFFF800h (refer to Table 26.14).

The status is read in any address of the flash memory.

Execute this command in RAM.

### 26.3.4.9 Protect Bit Program Command

Execute this command to protect a specific block in the flash memory. ROM code protection is enabled by setting one of the protect bits of the block to 0.

To set the protect bit of the designated block to 0 (protected), write 0067h to address FFFFF800h, verify that the FCA bit in the FMR0 register is 0, and then write 00D0h to the protect bit of the corresponding block (refer to Table 26.4).

To monitor the protect bit program, read the RDY bit in the FMSR0 register. This bit becomes 0 (busy) when the operation is in progress and 1 (ready) when the operation is completed.

To verify the state of protect bit, enter read protect bit status mode (refer to 26.3.4.10 “Enter Read Protect Bit Status Mode Command”), then read the flash memory.

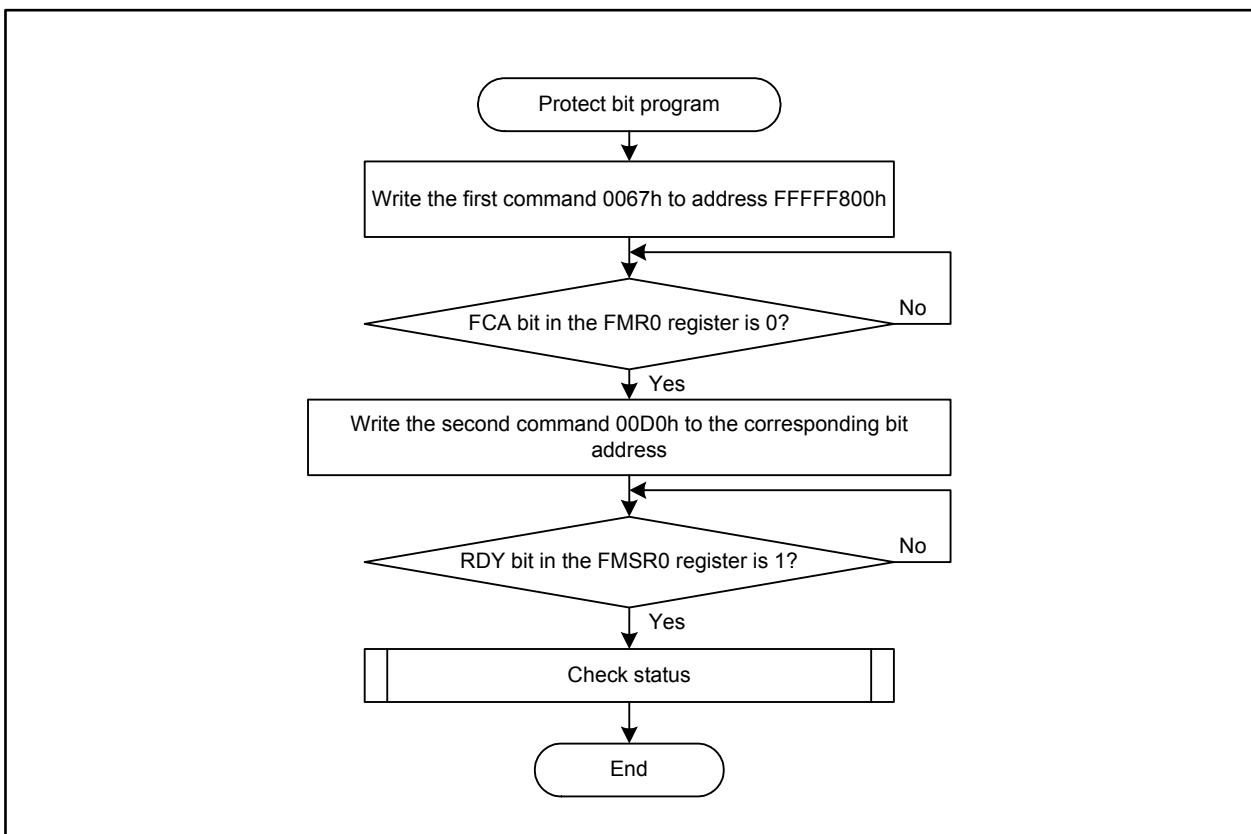


Figure 26.18 Protect Bit Program Command Execution Flowchart

### 26.3.4.10 Enter Read Protect Bit Status Mode Command

Execute this command to enter read protect bit status mode.

To read the protect bit status of the read block, write 0061h to address FFFFF800h (refer to Table 26.15). The status is read from any address in the flash memory.

Execute this command in RAM.

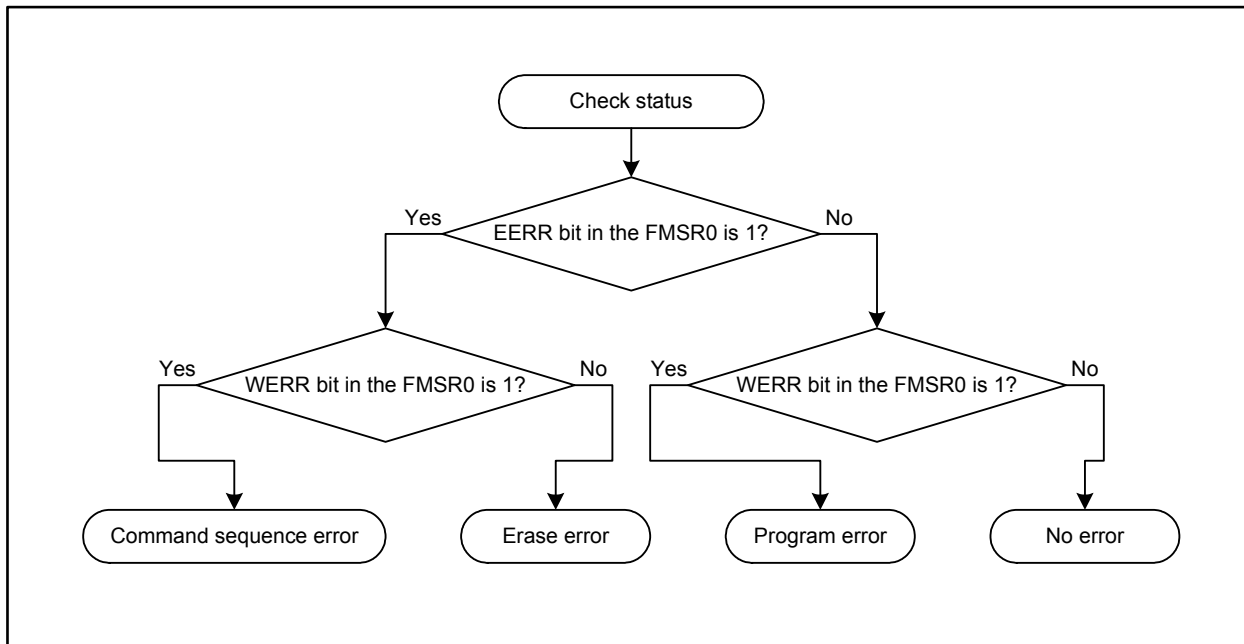
### 26.3.5 Status Check

To verify if a software command is successfully executed, read the EERR or WERR bit in the FMSR0 register, or the SR5 bit or SR4 bit in the status register.

Table 26.16 lists status and errors indicated by these bits and Figure 26.19 shows the flowchart of the status check.

**Table 26.16 Status and Errors**

FMSR0 Register (Status Register)		Error	Source of Error
EERR bit (SR5 bit)	WERR bit (SR4 bit)		
1	1	Command sequence error	<ul style="list-style-type: none"> <li>Data other than 00D0h or 00FFh (command to cancel) was written as the last command of two commands</li> <li>An unavailable address was specified by an address specifying command</li> </ul>
1	0	Erase error	<ul style="list-style-type: none"> <li>Attempted to erase a locked block</li> <li>Corresponding block was not erased properly</li> </ul>
0	1	Program error	<ul style="list-style-type: none"> <li>Attempted to program a locked block</li> <li>Data was not programmed properly</li> <li>Lock bit was not programmed properly</li> <li>Protect bit was not programmed properly</li> </ul>
0	0	No error	



**Figure 26.19 Status Check Flowchart**

When an error occurs, execute the clear status register command and then handle the error.

If erase errors or program errors occur frequently even though the program is correct, the corresponding block may be disabled.

## 26.4 Standard Serial I/O Mode

In standard serial I/O mode, an R32C/161 Group compatible serial programmer can be used to rewrite the flash memory while the MCU is mounted on a board.

For further information on the serial programmer, contact your serial programmer manufacturer and refer to the user's manual included with the serial programmer for instructions.

As shown in Table 26.17, this mode provides two types of transmit/receive mode: Standard serial I/O mode 1 which uses a synchronous serial interface, and standard serial I/O mode 2 which uses UART.

**Table 26.17 Standard Serial I/O Mode Specifications**

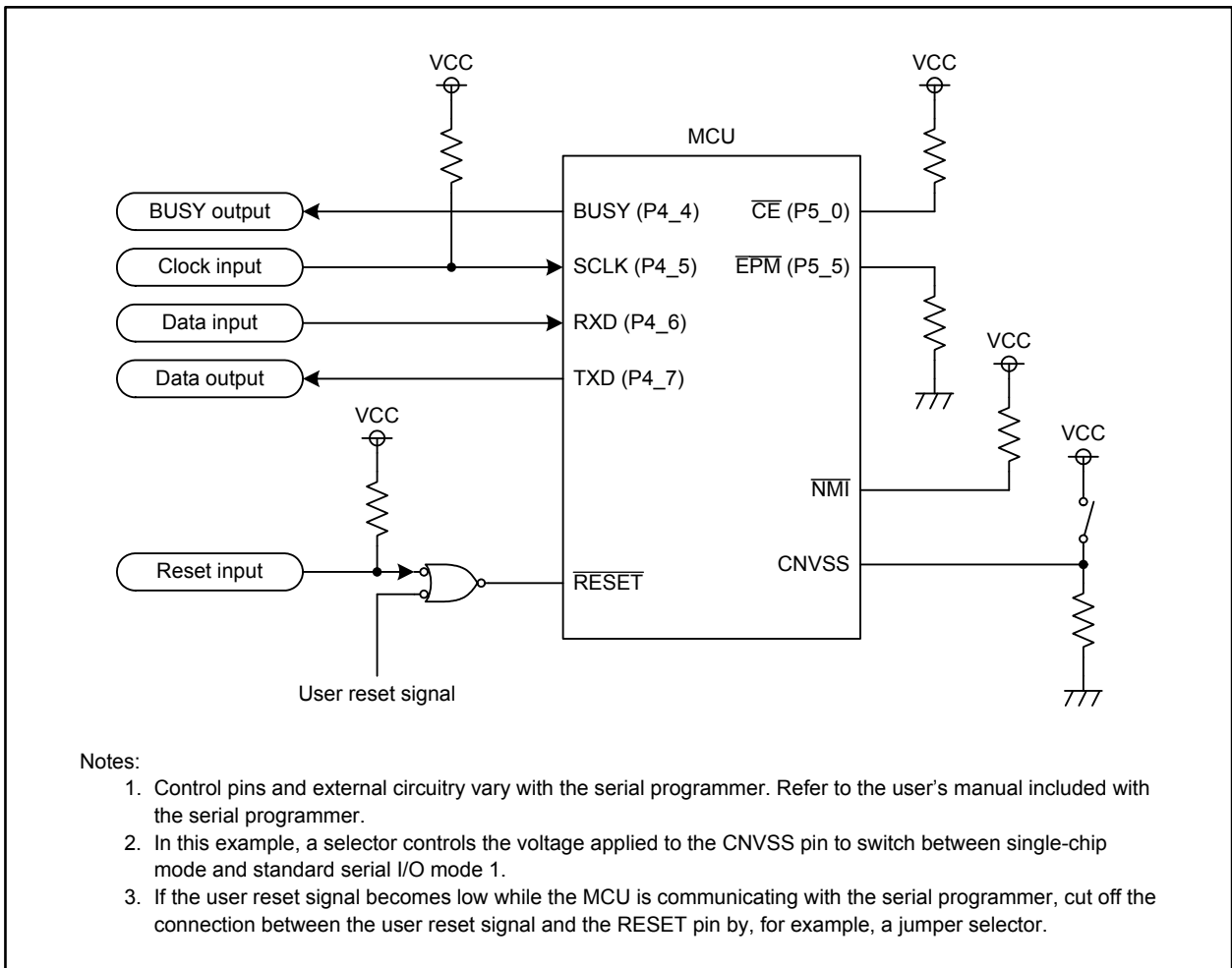
Item		Standard Serial I/O Mode 1	Standard Serial I/O Mode 2
Transmit/receive mode		Synchronous serial I/O	UART
Transmit/receive bit rate		High	Low
Serial interface to be used		UART1	UART1
Pin settings	CNVSS	High	High
	$\overline{\text{CE}}$ (P5_0)	High	High
	$\overline{\text{EPM}}$ (P5_5)	Low	Low
	SCLK (P4_5)	In reset: Low In transmission/reception: Transmit/receive clock	In reset: Low In transmission/reception: Unused
Pin functions	BUSY (P4_4)	BUSY signal	Monitor to check program operation
	RXD (P4_6)	Serial data input	Serial data input
	TXD (P4_7)	Serial data output	Serial data output

Table 26.18 lists the pin definitions and functions in standard serial I/O mode. Figures 26.20 and 26.21 show examples of a circuit application in standard serial I/O modes 1 and 2, respectively. Refer to the serial programmer user manual to handle pins controlled by the serial programmer.

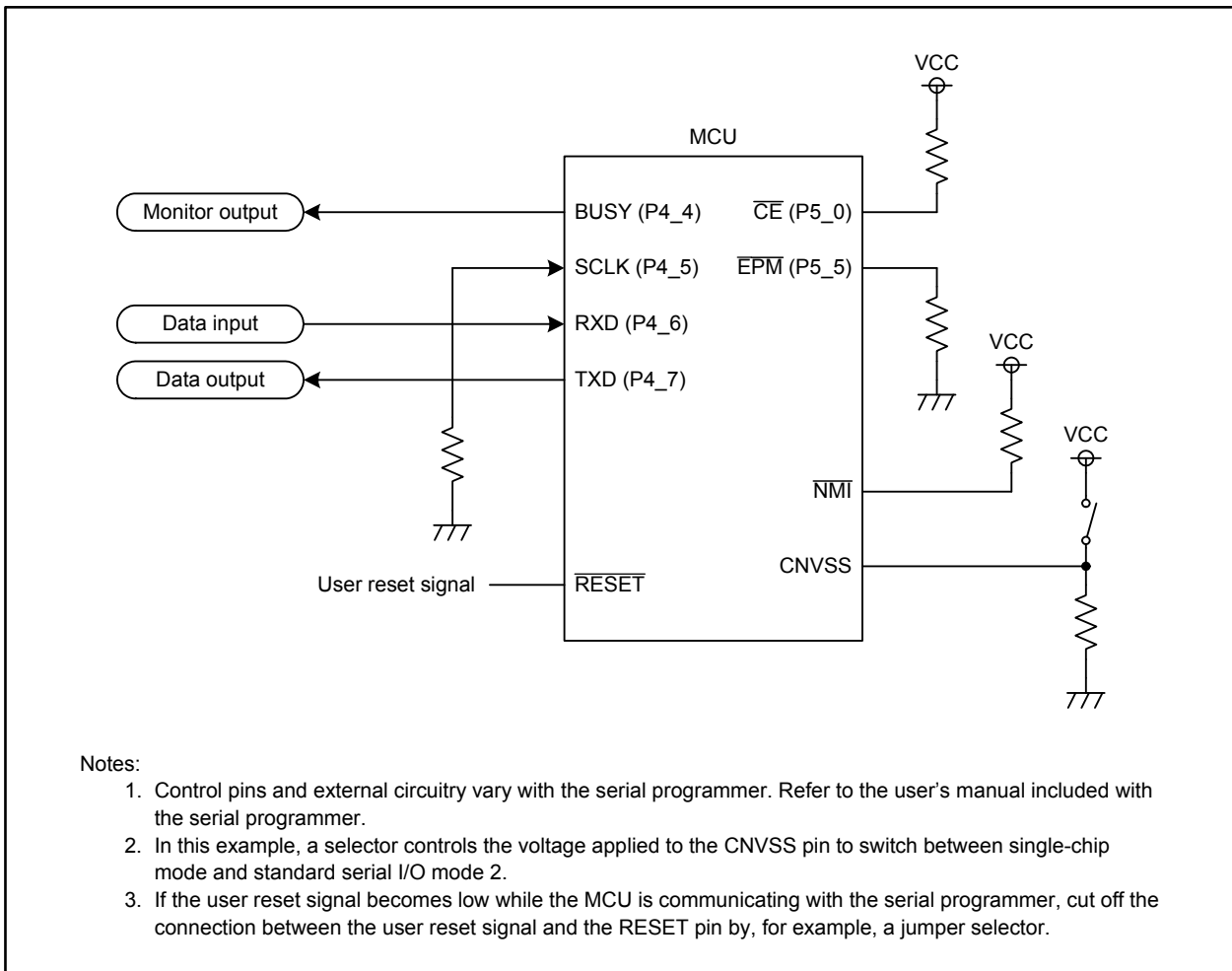


**Table 26.18 Pin Definitions and Functions in Standard Serial I/O Mode**

Pin Name	Function	I/O	Description
VCC, VSS	Power supply input	I	Applicable as follows: VCC = guaranteed voltage for program/erase operations, VSS = 0 V
VDC1, VDC0	Connecting pins for decoupling capacitor	—	A decoupling capacitor for internal voltage should be connected between VDC0 and VDC1
CNVSS	CNVSS	I	This pin should be connected to VCC via a resistor
RESET	Reset input	I	Reset input pin. While the RESET pin is driven low, a clock of 20 cycles or more should be input at the XIN pin
XIN	Main clock input	I	A ceramic resonator or a crystal oscillator should be connected between pins XIN and XOUT. An external clock should be input at XIN while leaving XOUT open
XOUT	Main clock output	O	
NSD	Debug port	I/O	This pin should be connected to VCC via a resistor of 1 to 4.7 kΩ
AVCC, AVSS	Analog power supply	I	AVCC and AVSS should be connected to VCC and VSS, respectively
VREF	Reference voltage input	I	Reference voltage input for the A/D converter and D/A converter
P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_3	Input port	I	High or low should be input, or the ports should be left open
P4_4	BUSY output	O	Standard serial I/O mode 1: BUSY output pin Standard serial I/O mode 2: Program operation monitor
P4_5	SCLK input	I	Standard serial I/O mode 1: Serial clock input pin Standard serial I/O mode 2: Low should be input
P4_6	Data input RXD	I	Serial data input pin
P4_7	Data output TXD	O	Serial data output pin
P5_0	CE input	I	High should be input
P5_1 to P5_4	Input port	I	High or low should be input, or the ports should be left open
P5_5	EPM input	I	Low should be input
P5_6, P5_7, P6_5, P6_7	Input port	I	High or low should be input, or the ports should be left open
P7_0, P7_4 to P7_6, P8_2 to P8_4	Input port	I	High or low should be input, or the ports should be left open
P8_5	NMI input	I	This pin should be connected to VCC via a resistor
P8_6, P8_7, P9_1, P9_3 to P9_7	Input port	I	High or low should be input, or the ports should be left open



**Figure 26.20 Circuit Application in Standard Serial I/O Mode 1**



**Figure 26.21 Circuit Application in Standard Serial I/O Mode 2**

## 26.5 Parallel I/O mode

In parallel I/O mode, an R32C/161 Group compatible parallel programmer can be used to rewrite the flash memory.

For further information on the parallel programmer, contact your parallel programmer manufacturer and refer to the user's manual included with your parallel programmer for instructions.

## 26.6 Notes on Flash Memory Rewriting

### 26.6.1 Note on Power Supply

- Keep the supply voltage constant within the range specified in the electrical characteristics while a rewrite operation on the flash memory is in progress. If the supply voltage goes beyond the guaranteed value, the device cannot be guaranteed.

### 26.6.2 Note on Hardware Reset

- Do not perform a hardware reset while a rewrite operation on the flash memory is in progress.

### 26.6.3 Note on Flash Memory Protection

- If an ID code written in an assigned address has an error, any read/write operation on the flash memory in standard serial I/O mode is disabled.

### 26.6.4 Notes on Programming

- Do not set the FEW bit in the FMCR register to 1 (CPU rewrite mode) in low speed mode or low power mode.
- When the EWM bit in the E2FM register is 1 (program or erase enabled), do not set the FEW bit in the FMCR register to 1.
- The program, block erase, lock bit program, and protect bit program are interrupted by an NMI, a watchdog timer interrupt, an oscillator stop detection interrupt, or a low voltage detection interrupt. If any of the software commands above are interrupted, erase the corresponding block and then execute the same command again. If the block erase command is interrupted, the lock bit and protect bit values become undefined. Therefore, disable the lock bit, and then execute the block erase command again.

### 26.6.5 Notes on Interrupts

- EW0 mode
  - To use interrupts assigned to the relocatable vector table, the vector table should be addressed in RAM space.
  - When an NMI, watchdog timer interrupt, oscillator stop detection interrupt, or low voltage detection interrupt occurs, the flash memory module automatically enters read array mode. Therefore, these interrupts are enabled even during a rewrite operation. However, the rewrite operation in progress is aborted by the interrupts and registers FMR0 and FRSR0 are reset. When the interrupt handler has ended, set the LBD bit in the FMR1 register to 1 (lock bit protection disabled) to re-execute the rewrite operation.
  - Instructions BRK, INTO, and UND, which refer to data on the flash memory, cannot be used in this mode.
- EW1 mode
  - Interrupts assigned to the relocatable vector table should not be accepted during program or block erase operation.
  - The watchdog timer should not be used in count source protect mode. The watchdog timer interrupt should not be generated.
  - When an NMI, watchdog timer interrupt, oscillator stop detection interrupt, or low voltage detection interrupt occurs, the flash memory module automatically enters read array mode. Therefore, these interrupts are enabled even during a rewrite operation. However, the rewrite operation in progress is aborted by the interrupts and registers FMR0 and FRSR0 are reset. When the interrupt handler has ended, set the EWM bit in the FMR0 register to 1 (EW1 mode) and the LBD bit in the FMR1 register to 1 (lock bit protection disabled) to re-execute the rewrite operation.

### 26.6.6 Notes on Rewrite Control Program

- EW0 mode
  - If the supply voltage drops during the rewrite operation of blocks having the rewrite control program, the rewrite control program may not be successfully rewritten, and the rewrite operation itself may not be performed. In this case, perform the rewrite operation by serial programmer or parallel programmer.
- EW1 mode
  - Do not rewrite blocks having the rewrite control program.

### 26.6.7 Notes on Number of Program/Erase Cycles and Software Command Execution Time

- The time to execute software commands (program, block erase, lock bit program, and protect bit program) increases as the number of program/erase cycles increases. If the number of program/erase cycles exceeds the endurance value specified in the electrical characteristics, it may take an unpredictable amount of time to execute the software commands. The wait time for executing software commands should be set much longer than the execution time specified in the electrical characteristics.

### 26.6.8 Other Notes

- The minimum values of program/erase cycles specified in the electrical characteristics are the maximum values that can guarantee the initial performance of the flash memory. The program/erase operation may still be performed even if the number of program/erase cycles exceeds the guaranteed values.
- Chips repeatedly programmed and erased for debugging should not be used for commercial products.

## 27. E<sup>2</sup>PROM Emulation Data Flash

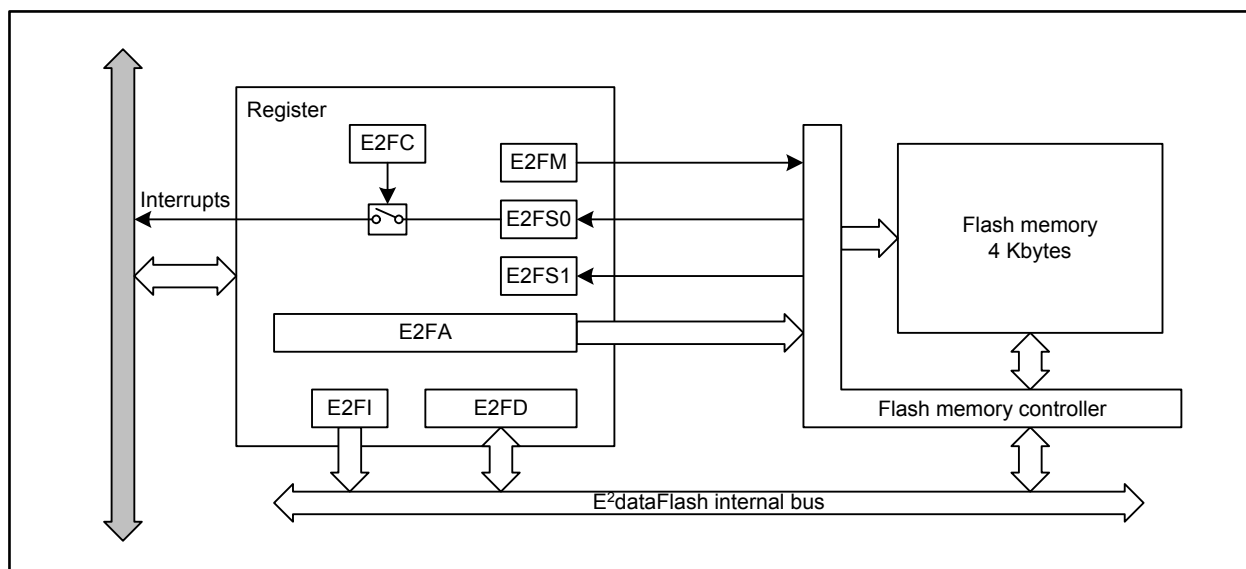
### 27.1 Overview

E<sup>2</sup>PROM emulation data flash (hereafter referred to as E<sup>2</sup>dataFlash) is an on-chip data flash which has features of serial E<sup>2</sup>PROMs. It is erasable in much smaller blocks than the other data flash and can be programmed and erased without keeping the CPU away from its operation.

Table 27.1 lists the specifications and Figure 27.1 shows the block diagram of the E<sup>2</sup>dataFlash.

**Table 27.1 E<sup>2</sup>dataFlash Specifications**

Item	Specification	
	ECC disabled	ECC enabled
Memory size	4 Kbytes	2 Kbytes
Block size	32 bytes	16 bytes
Number of blocks	128	
Unit to be programmed	2 bytes	1 byte
Unit to be erased	1 block	
Program and erase control method	By software commands	
Software commands	4	
Error correction	None	1-bit error correctable



**Figure 27.1 E<sup>2</sup>dataFlash Block Diagram**

The CPU indirectly accesses the E<sup>2</sup>dataFlash via registers E2FA, E2FI, and E2FD which are set for SFR space.

Figures 27.2 to 27.8 show E<sup>2</sup>dataFlash associated registers.

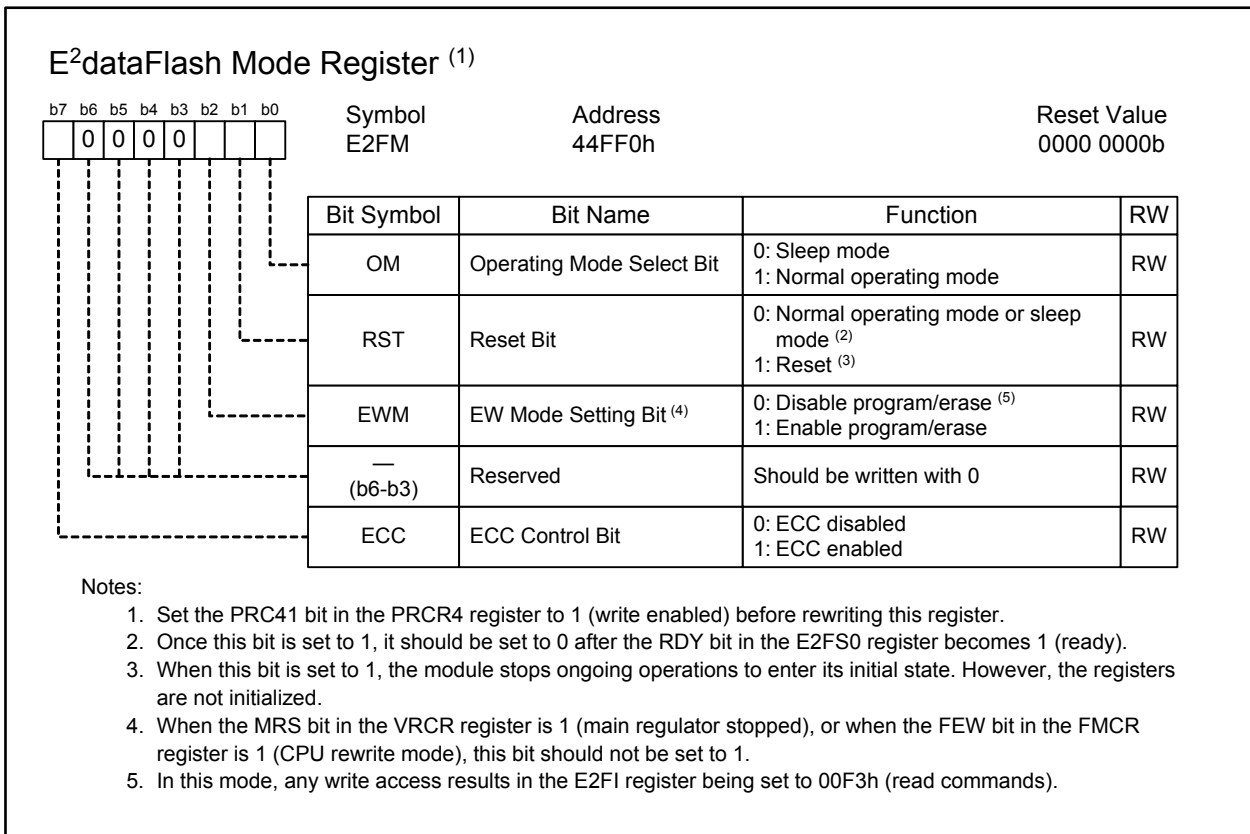


Figure 27.2 E2FM Register

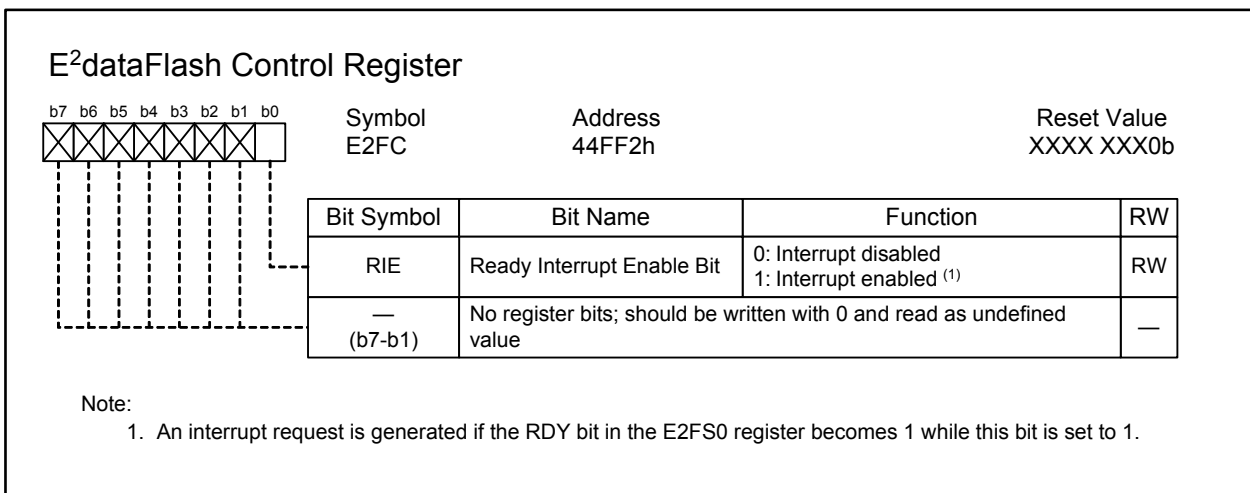


Figure 27.3 E2FC Register

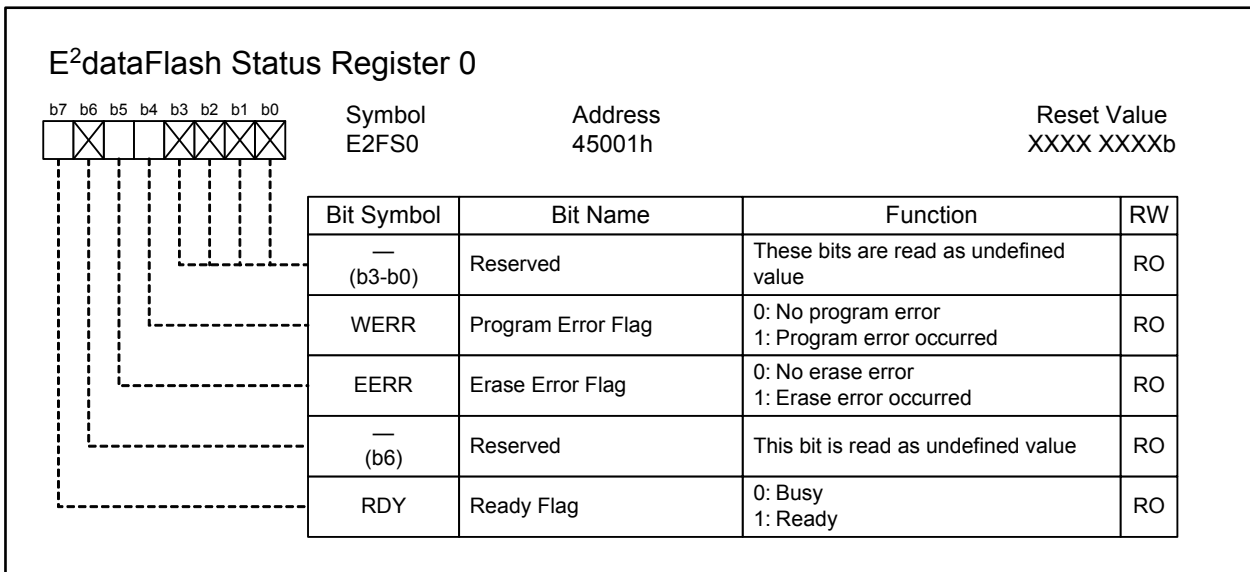


Figure 27.4 E2FS0 Register

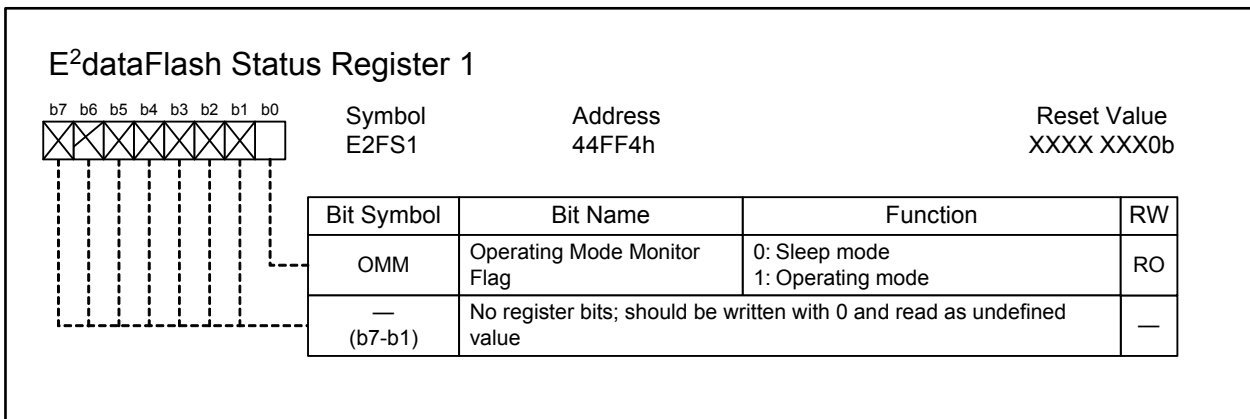


Figure 27.5 E2FS1 Register

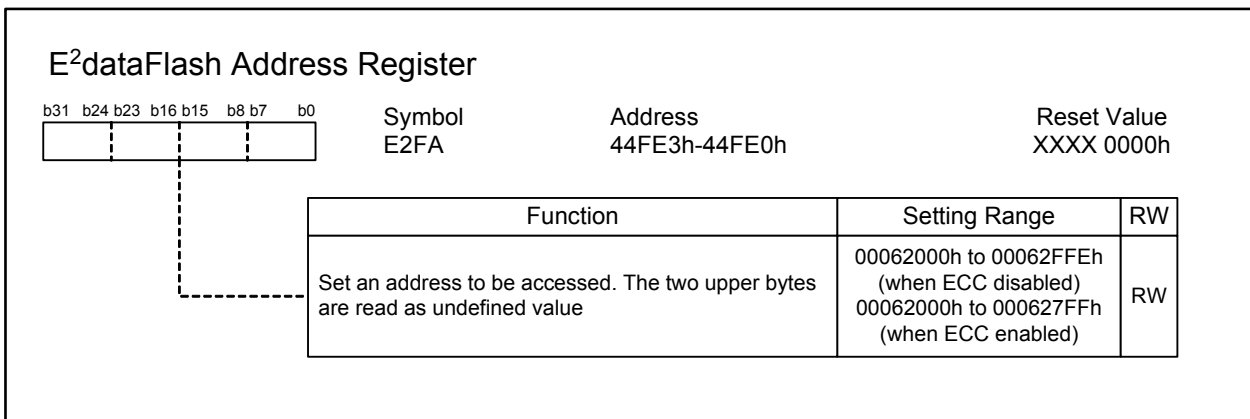


Figure 27.6 E2FA Register



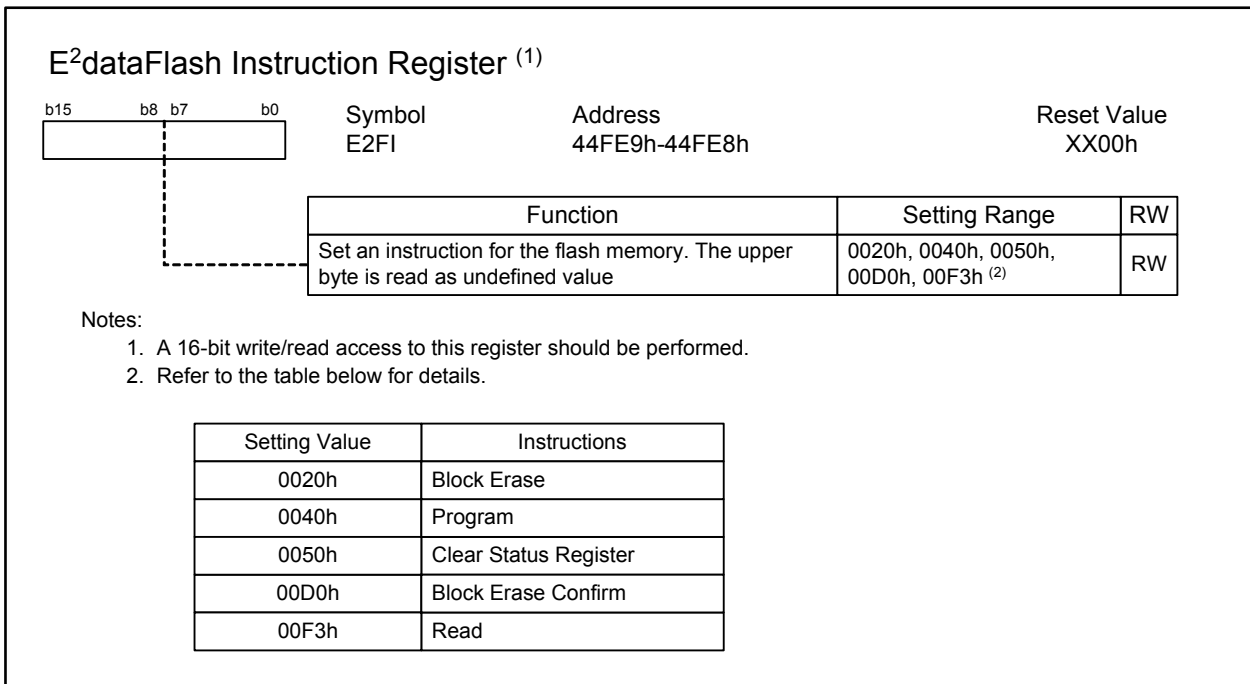


Figure 27.7 E2FI Register

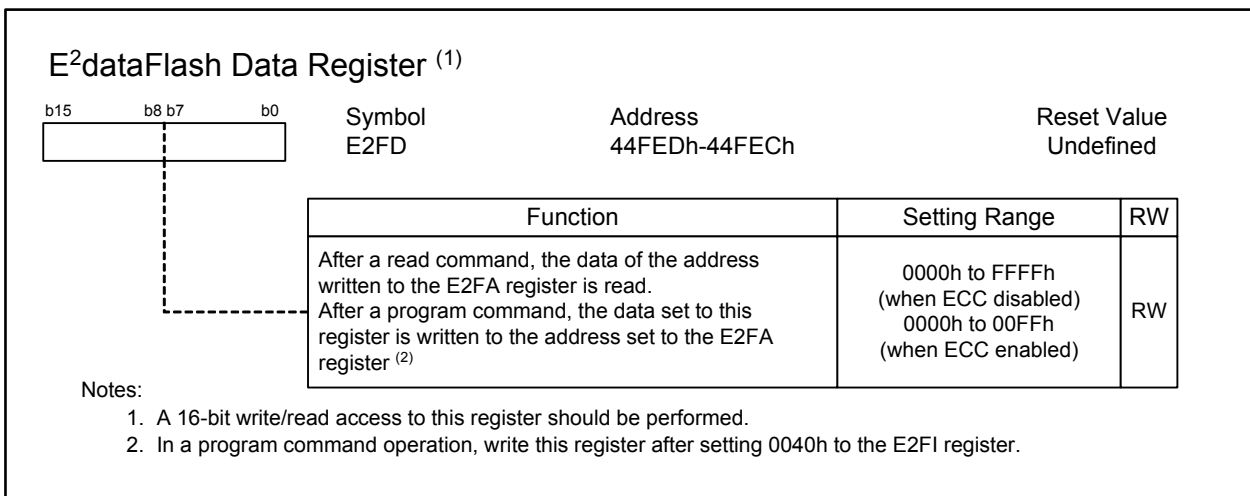


Figure 27.8 E2FD Register

### 27.2 Block Configuration

The E<sup>2</sup>dataFlash consists of 32 bytes x 128 blocks of flash memory when ECC is disabled, and 16 bytes x 128 blocks when ECC is enabled (refer to Figure 27.9 for details).

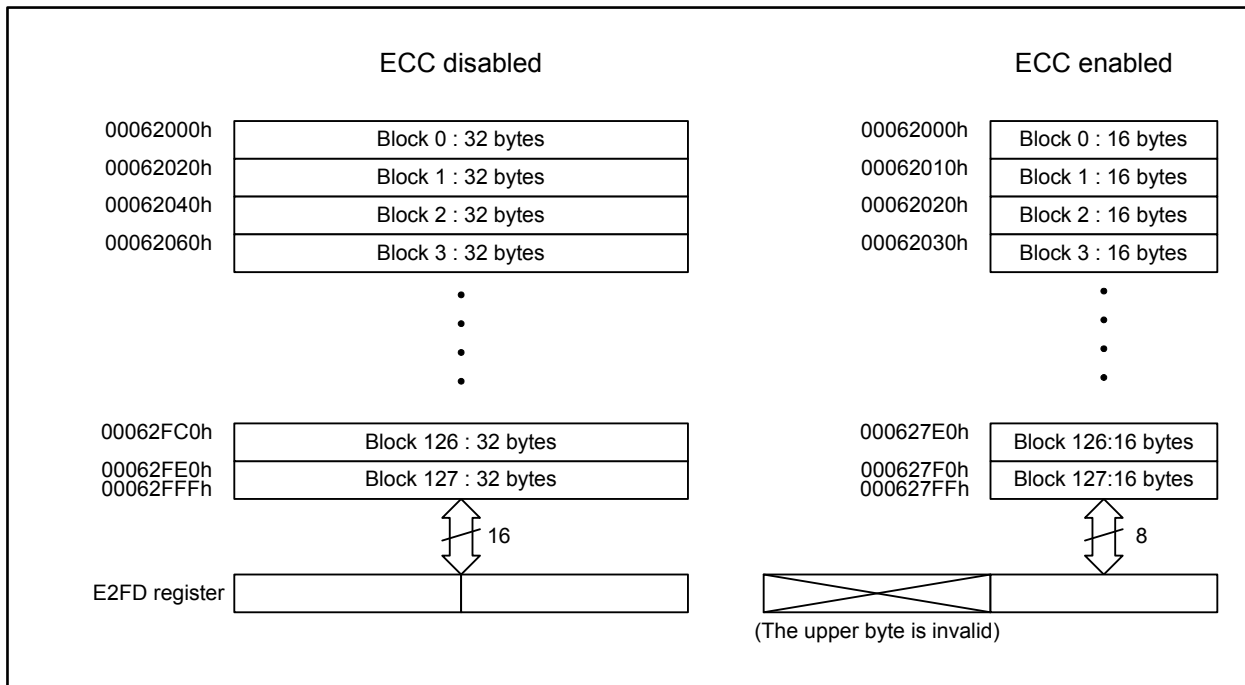


Figure 27.9 E<sup>2</sup>dataFlash Memory Configuration

### 27.3 Operational Procedures

Figures 27.10 to 27.13 show the read, program, block erase, and clear status operational procedures. Follow the procedures to operate each.

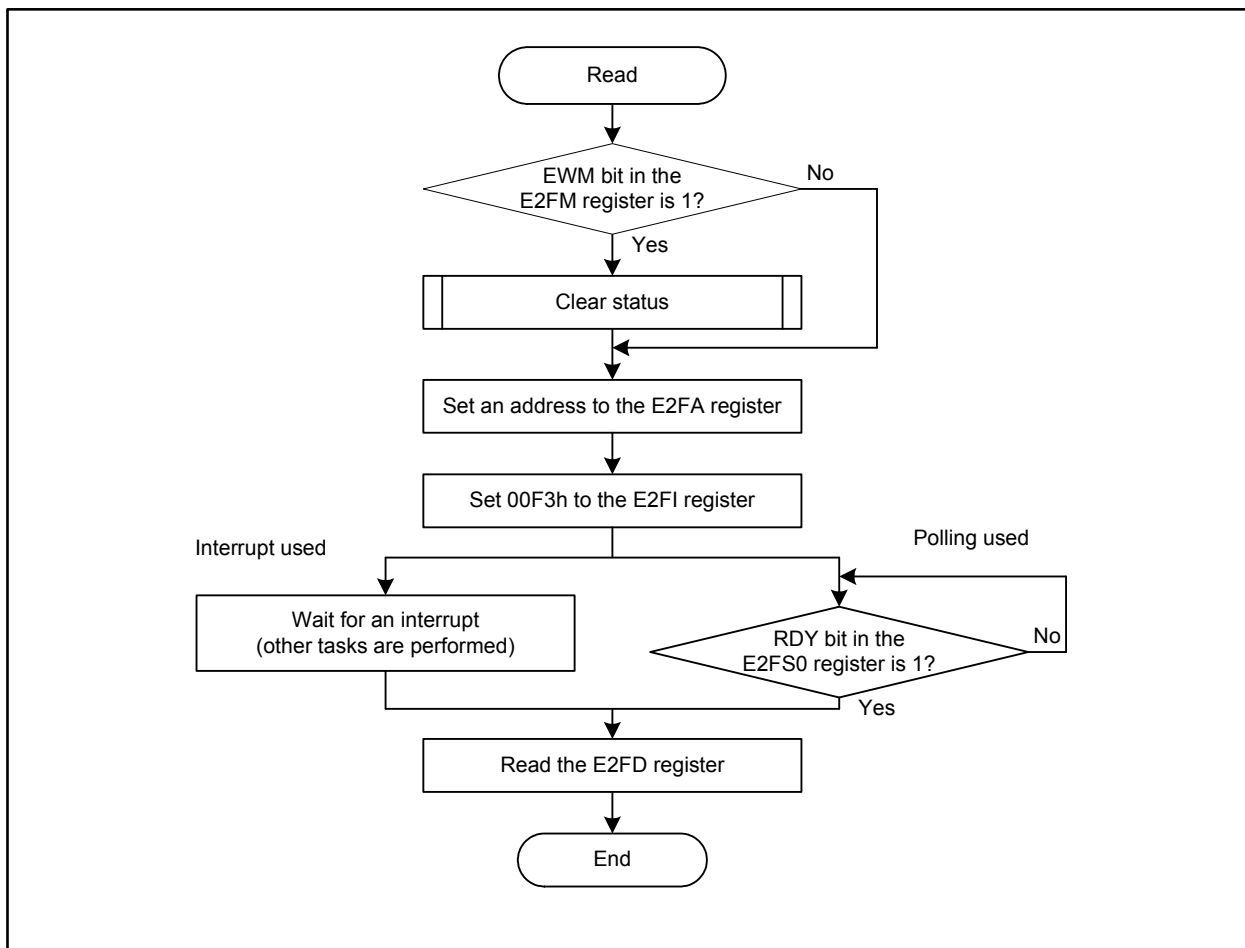


Figure 27.10 Read Operation

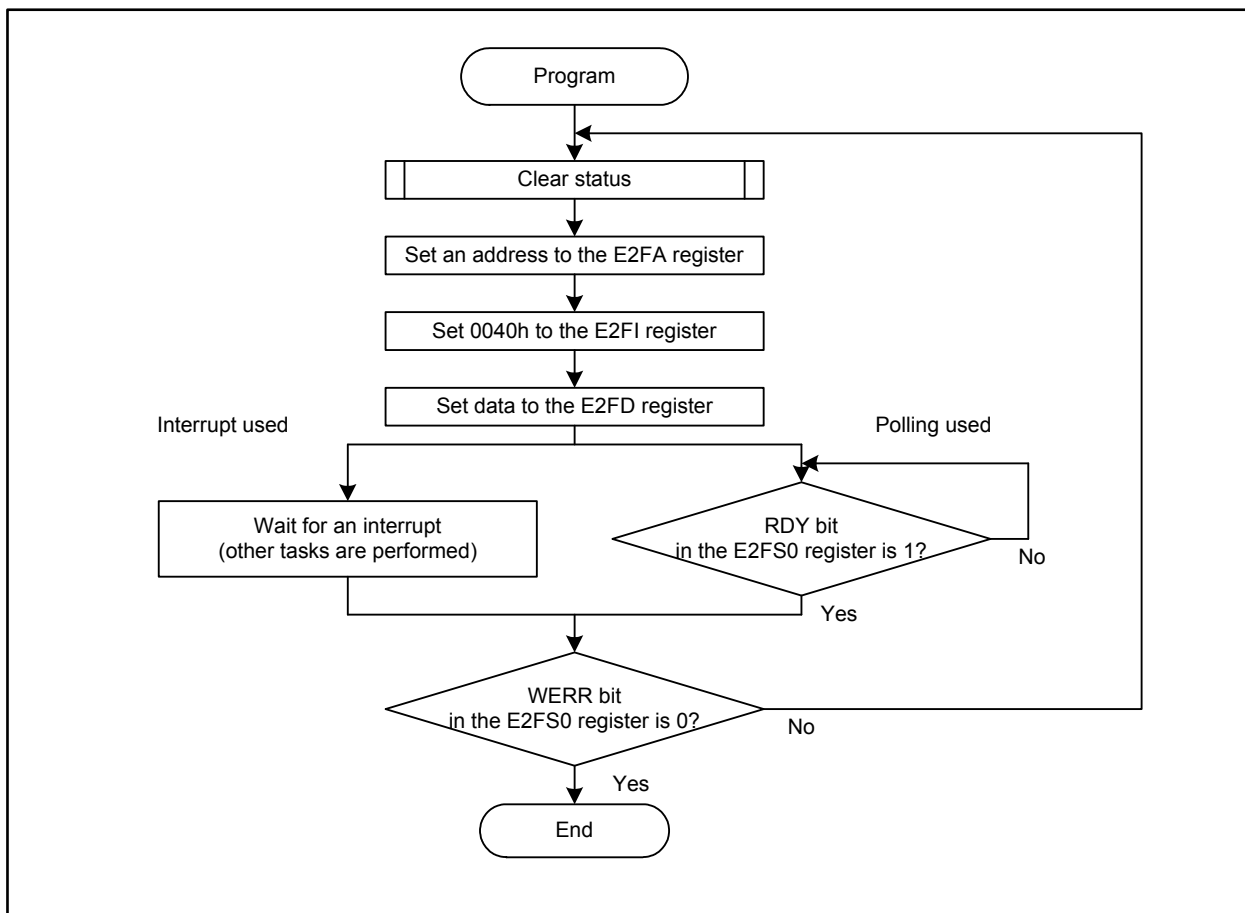


Figure 27.11 Program Operation

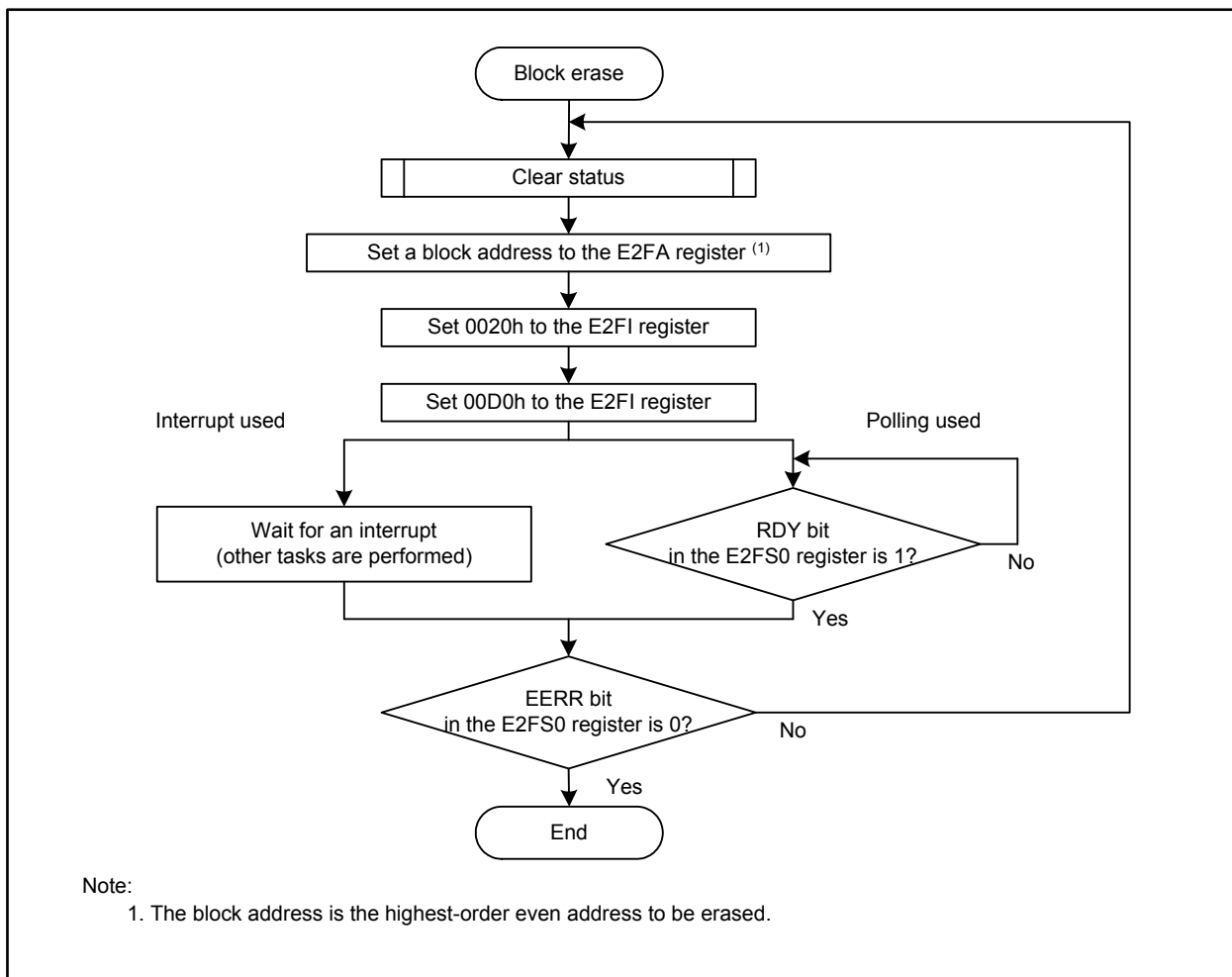


Figure 27.12 Block Erase Operation

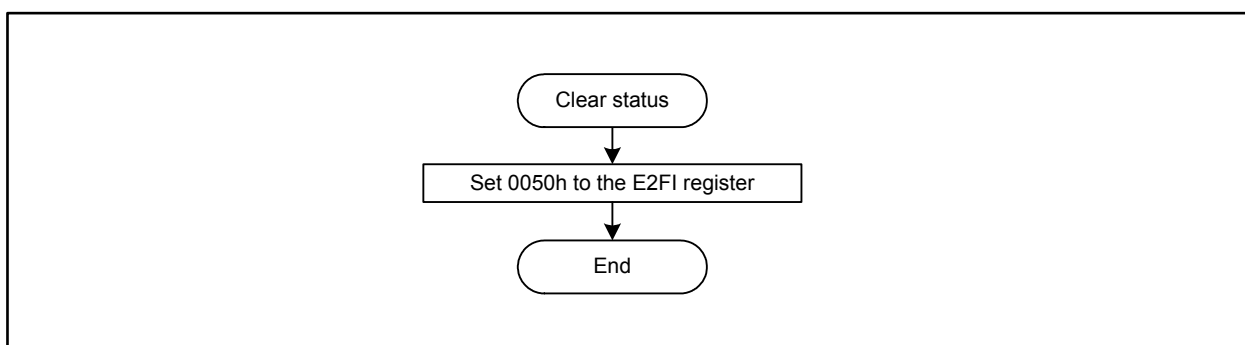


Figure 27.13 Clear Status Operation

#### 27.4 Note on E<sup>2</sup>dataFlash

When the FEW bit in the FMCR register is 1 (CPU rewrite mode), do not set the EWM bit in the E2FM register to 1 (program or erase enabled).

## 28. Electrical Characteristics

**Table 28.1 Absolute Maximum Ratings (1)**

Symbol	Characteristic		Condition	Value	Unit
$V_{CC}$	Supply voltage		$V_{CC} = AV_{CC}$	-0.3 to 6.0	V
$AV_{CC}$	Analog supply voltage		$V_{CC} = AV_{CC}$	-0.3 to 6.0	V
$V_I$	Input voltage	XIN, $\overline{RESET}$ , CNVSS, NSD, $V_{REF}$ , P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_7, P5_0 to P5_7, P6_5, P6_7, P7_0, P7_4 to P7_6, P8_2 to P8_7, P9_1, P9_3 to P9_7		-0.3 to $V_{CC} + 0.3$	V
$V_O$	Output voltage	XOUT, P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_7, P5_0 to P5_7, P6_5, P6_7, P7_0, P7_4 to P7_6, P8_2 to P8_4, P8_6, P8_7, P9_3 to P9_7		-0.3 to $V_{CC} + 0.3$	V
$P_d$	Power consumption		$T_a = 25^\circ\text{C}$	500	mW
—	Operating temperature range			-40 to 125	$^\circ\text{C}$
$T_{stg}$	Storage temperature range			-65 to 150	$^\circ\text{C}$

Note:

1. Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

**Table 28.2 Operating Conditions (1/6) (1)**

Symbol	Characteristic		Value			Unit
			Min.	Typ.	Max.	
V <sub>CC</sub>	Digital supply voltage		3.0	5.0	5.5	V
AV <sub>CC</sub>	Analog supply voltage			V <sub>CC</sub>		V
V <sub>REF</sub>	Reference voltage		3.0		V <sub>CC</sub>	V
V <sub>SS</sub>	Digital ground voltage			0		V
AV <sub>SS</sub>	Analog ground voltage			0		V
dV <sub>CC</sub> /dt	V <sub>CC</sub> ramp up rate (V <sub>CC</sub> < 2.0 V)		0.05			V/ms
V <sub>IH</sub>	High level input voltage	XIN, $\overline{\text{RESET}}$ , CNVSS, NSD	0.8 × V <sub>CC</sub>		V <sub>CC</sub>	V
		P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_7, P5_0 to P5_7, P6_5, P6_7, P7_0, P7_4 to P7_6, P8_2 to P8_7 (2), P9_1, P9_3 to P9_7	0.7 × V <sub>CC</sub>		V <sub>CC</sub>	V
V <sub>IL</sub>	Low level input voltage	XIN, $\overline{\text{RESET}}$ , CNVSS, NSD	0		0.2 × V <sub>CC</sub>	V
		P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_7, P5_0 to P5_7, P6_5, P6_7, P7_0, P7_4 to P7_6, P8_2 to P8_7 (2), P9_1, P9_3 to P9_7	0		0.3 × V <sub>CC</sub>	V
T <sub>opr</sub>	Operating temperature range	J version	-40		85	°C
		L version	-40		105	°C
		K version	-40		125	°C

## Notes:

1. The device is operationally guaranteed under these operating conditions.
2. V<sub>IH</sub> and V<sub>IL</sub> for P8\_7 are specified for P8\_7 as a programmable port. These values are not applicable for P8\_7 as XCIN.



**Table 28.3 Operating Conditions (2/6)**  
**( $V_{CC} = 3.0$  to  $5.5$  V,  $V_{SS} = 0$  V, and  $T_a = T_{opr}$ , unless otherwise noted) (1)**

Symbol	Characteristic		Value (2)			Unit
			Min.	Typ.	Max.	
$C_{VDC}$	Decoupling capacitance for voltage regulator	Inter-pin voltage: 1.5 V	2.4		10.0	$\mu\text{F}$

## Notes:

1. The device is operationally guaranteed under these operating conditions.
2. This value should be met with due consideration to the following conditions: operating temperature, DC bias, aging, etc.

**Table 28.4 Operating Conditions (3/6)**  
**( $V_{CC} = 3.0$  to  $5.5$  V,  $V_{SS} = 0$  V, and  $T_a = T_{opr}$ , unless otherwise noted) (1)**

Symbol	Characteristic		Value			Unit
			Min.	Typ.	Max.	
$I_{OH(peak)}$	High level peak output current (2)	P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_7, P5_0 to P5_7, P6_5, P6_7, P7_0, P7_4 to P7_6, P8_2 to P8_4, P8_6, P8_7, P9_3 to P9_7			-10.0	mA
$I_{OH(avg)}$	High level average output current (3)	P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_7, P5_0 to P5_7, P6_5, P6_7, P7_0, P7_4 to P7_6, P8_2 to P8_4, P8_6, P8_7, P9_3 to P9_7			-5.0	mA
$I_{OL(peak)}$	Low level peak output current (2)	P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_7, P5_0 to P5_7, P6_5, P6_7, P7_0, P7_4 to P7_6, P8_2 to P8_4, P8_6, P8_7, P9_3 to P9_7			10.0	mA
$I_{OL(avg)}$	Low level average output current (3)	P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_7, P5_0 to P5_7, P6_5, P6_7, P7_0, P7_4 to P7_6, P8_2 to P8_4, P8_6, P8_7, P9_3 to P9_7			5.0	mA

## Notes:

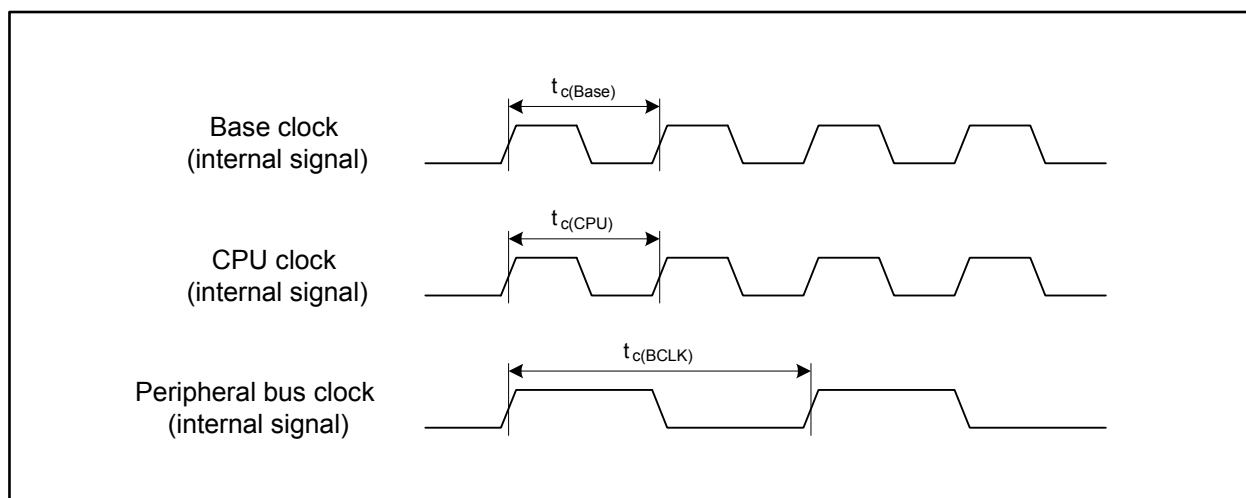
- The device is operationally guaranteed under these operating conditions.
- The following conditions should be satisfied:
  - The sum of  $I_{OL(peak)}$  of ports P0, P1, P2, P8\_6, P8\_7, and P9 is 80 mA or less.
  - The sum of  $I_{OL(peak)}$  of ports P3, P4, P5, P6, P7, and P8\_2 to P8\_4 is 80 mA or less.
  - The sum of  $I_{OH(peak)}$  of ports P1 and P2 is -40 mA or less.
  - The sum of  $I_{OH(peak)}$  of ports P0, P9\_6, and P9\_7 is -40 mA or less.
  - The sum of  $I_{OH(peak)}$  of ports P3, P4, P5, and P6 is -40 mA or less.
  - The sum of  $I_{OH(peak)}$  of ports P7, P8, and P9\_3 to P9\_5 is -40 mA or less.
- Average value within 100 ms.

**Table 28.5 Operating Conditions (4/6)**  
 ( $V_{CC} = 3.0$  to  $5.5$  V,  $V_{SS} = 0$  V, and  $T_a = T_{opr}$ , unless otherwise noted) (1)

Symbol	Characteristic	Value			Unit
		Min.	Typ.	Max.	
$f_{(XIN)}$	Main clock oscillator frequency	4		8	MHz
$f_{(XRef)}$	Reference clock frequency	2		4	MHz
$f_{(PLL)}$	PLL clock oscillator frequency	96		144	MHz
$f_{(Base)}$	Base clock frequency			48	MHz
$t_{c(Base)}$	Base clock cycle time	20.83			ns
$f_{(CPU)}$	CPU operating frequency			48	MHz
$t_{c(CPU)}$	CPU clock cycle time	20.83			ns
$f_{(BCLK)}$	Peripheral bus clock operating frequency			24	MHz
$t_{c(BCLK)}$	Peripheral bus clock cycle time	41.67			ns
$f_{(PER)}$	Peripheral clock source frequency			24	MHz
$f_{(XCIN)}$	Sub clock oscillator frequency		32.768	50	kHz

Note:

1. The device is operationally guaranteed under these operating conditions.



**Figure 28.1 Clock Cycle Time**

**Table 28.6 Operating Conditions (5/6)****( $V_{CC} = 3.0$  to  $5.5$  V,  $V_{SS} = 0$  V, and  $T_a = T_{opr}$ , unless otherwise noted) (1, 2)**

Symbol	Characteristic		Measurement Condition	Value			Unit
				Min.	Typ.	Max.	
$I_{IC(H)}$	High input injection current	P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_7, P5_0 to P5_7, P6_5, P6_7, P7_0, P7_4 to P7_6, P8_2 to P8_5, P9_3 to P9_7	$V_I > V_{CC}$			2	mA
$I_{IC(L)}$	Low input injection current	P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_7, P5_0 to P5_7, P6_5, P6_7, P7_0, P7_4 to P7_6, P8_2 to P8_5, P9_3 to P9_7	$V_I < V_{SS}$			-2	mA
$\Sigma I_{IC} $	Total injection current					20	mA

## Notes:

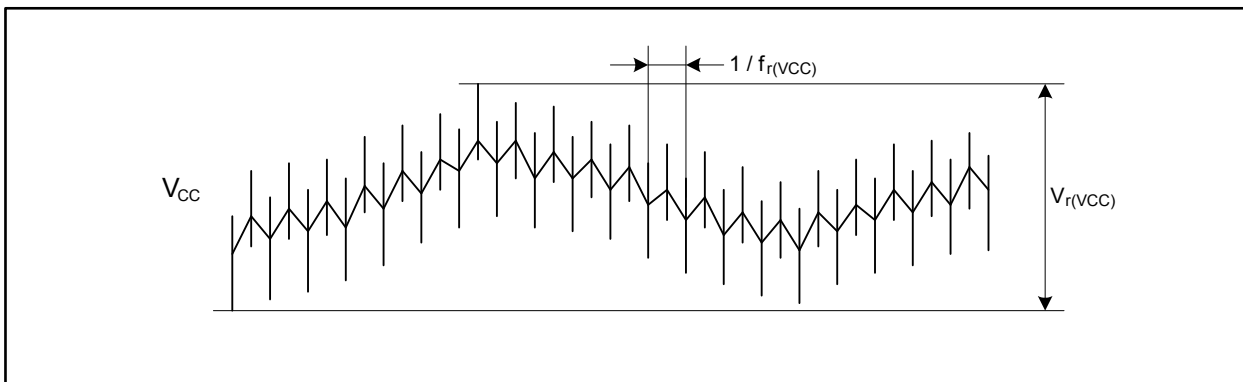
1. The device is operationally guaranteed under these operating conditions.
2. These conditions are applicable when each port is designated as input.

**Table 28.7 Operating Conditions (6/6)**  
 ( $V_{CC} = 3.0$  to  $5.5$  V,  $V_{SS} = 0$  V, and  $T_a = T_{opr}$ , unless otherwise noted) (1)

Symbol	Characteristic	Value			Unit
		Min.	Typ.	Max.	
$V_{r(VCC)}$	Allowable ripple voltage	$V_{CC} = 5.0$ V		0.5	Vp-p
		$V_{CC} = 3.0$ V		0.3	Vp-p
$dV_{r(VCC)}/dt$	Ripple voltage gradient	$V_{CC} = 5.0$ V		$\pm 0.3$	V/ms
		$V_{CC} = 3.0$ V		$\pm 0.3$	V/ms
$f_{r(VCC)}$	Allowable ripple frequency			10	kHz

Note:

1. The device is operationally guaranteed under these operating conditions.



**Figure 28.2 Ripple Waveform**

**Table 28.8 Electrical Characteristics of Flash Memory**  
**( $V_{CC} = 3.0$  to  $5.5$  V,  $V_{SS} = 0$  V, and  $T_a = T_{opr}$ , unless otherwise noted)**

Symbol	Characteristic	Value			Unit	
		Min.	Typ.	Max.		
—	Program/erase cycles (1)	Program area	1000		Cycles	
		Data area	10000		Cycles	
—	4-word program time	Program area		150	900	$\mu$ s
		Data area		300	1700	$\mu$ s
—	Lock bit program time	Program area		70	500	$\mu$ s
		Data area		140	1000	$\mu$ s
—	Block erasure time	4-Kbyte block		0.12	3.0	s
		32-Kbyte block		0.17	3.0	s
		64-Kbyte block		0.20	3.0	s
—	Data retention (2)	$T_a = 55^\circ\text{C}$ (3, 4)	20			Years

## Notes:

- Program/erase definition  
 This value represents the number of erasures per block.  
 When the number of program/erase cycles is n, each block can be erased n times.  
 For example, if a 4-word write is performed in 512 different addresses in the 4-Kbyte block A and then the block is erased, this is counted as a single program/erase operation.  
 However, the same address cannot be written to more than once per erasure (overwrite disabled).
- Data retention includes periods when no supply voltage is applied and no clock is provided.
- This data retention includes 3000 hours in  $T_a = 125^\circ\text{C}$  and 7000 hours in  $T_a = 85^\circ\text{C}$ .
- Contact a Renesas Electronics sales office for data retention times other than the above condition.

**Table 28.9 Electrical Characteristics of E<sup>2</sup>dataFlash**  
( $V_{CC} = 3.0$  to  $5.5$  V,  $V_{SS} = 0$  V, and  $T_a = T_{opr}$ , unless otherwise noted)

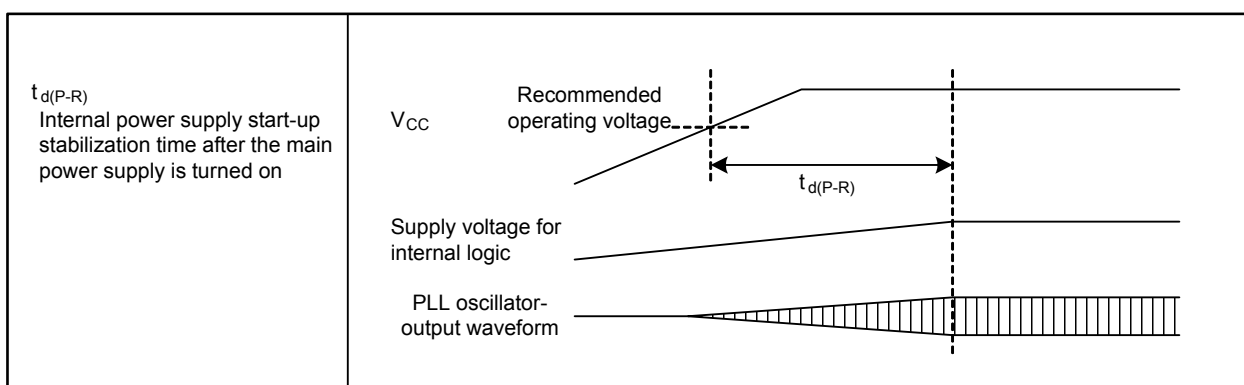
Symbol	Characteristic		Value			Unit
			Min.	Typ.	Max.	
—	Program/erase cycles (1)		100000			Cycles
—	Word program time			100	2000	$\mu$ s
—	Block erasure time	32 byte block		15	200	ms
$t_{PS}$	Flash memory circuit start-up stabilization time			35	50	$\mu$ s
—	Data retention (2)	$T_a = 55^\circ\text{C}$ (3, 4)	20			Years

## Notes:

1. Program/erase definition  
This value represents the number of erasure per block.  
When the number of program/erase cycles is n, each block can be erased n times.  
For example, if a word write is performed in 16 different addresses in a block and then the block is erased, this is counted as a single program/erase operation. However, the same address cannot be written to more than once per erasure (overwrite disabled).
2. Data retention includes periods when no supply voltage is applied and no clock is provided.
3. This data retention includes 3000 hours in  $T_a = 125^\circ\text{C}$  and 7000 hours in  $T_a = 85^\circ\text{C}$ .
4. Contact a Renesas Electronics sales office for data retention times other than the above condition.

**Table 28.10 Power Supply Circuit Timing Characteristics**  
( $V_{CC} = 3.0$  to  $5.5$  V,  $V_{SS} = 0$  V, and  $T_a = T_{opr}$ , unless otherwise noted)

Symbol	Characteristic	Measurement Condition	Value			Unit
			Min.	Typ.	Max.	
$t_{d(P-R)}$	Internal power supply start-up stabilization time after the main power supply is turned on				2	ms



**Figure 28.3 Power Supply Circuit Timing**

**Table 28.11 Electrical Characteristics of Voltage Regulator for Internal Logic**  
 ( $V_{CC} = 3.0$  to  $5.5$  V,  $V_{SS} = 0$  V, and  $T_a = T_{opr}$ , unless otherwise noted)

Symbol	Characteristics	Measurement Condition	Value			Unit
			Min.	Typ.	Max.	
$V_{VDC1}$	Output voltage			1.5		V

**Table 28.12 Electrical Characteristics of Low Voltage Detector**  
 ( $V_{CC} = 4.2$  to  $5.5$  V,  $V_{SS} = 0$  V, and  $T_a = T_{opr}$ , unless otherwise noted)

Symbol	Characteristics	Measurement Condition	Value			Unit
			Min.	Typ.	Max.	
$\Delta V_{det}$	Detected voltage error				$\pm 0.2$	V
$V_{det(R)} - V_{det(F)}$	Hysteresis width		0			V
—	Self-consuming current	$V_{CC} = 5.0$ V, low voltage detector enabled		4		$\mu$ A
$t_{d(E-A)}$	Operation start time of low voltage detector				150	$\mu$ s

**Table 28.13 Electrical Characteristics of Oscillator**  
 ( $V_{CC} = 3.0$  to  $5.5$  V,  $V_{SS} = 0$  V, and  $T_a = T_{opr}$ , unless otherwise noted)

Symbol	Characteristics	Measurement Condition	Value			Unit
			Min.	Typ.	Max.	
$f_{SO(PLL)}$	PLL clock self-oscillation frequency		35	50	80	MHz
$ \Delta f_{LOCK} $	Lock detection (1)				2	%
$ \Delta f_{UNLOCK} $	Unlock detection (1)		2			%
$t_{LOCK(PLL)}$	PLL lock time (2, 3)	$f_{(XRef)} = 4$ MHz			1	ms
$t_{jitter(p-p)}$	PLL jitter period (p-p)				2.0	ns
$f_{(OCO)}$	On-chip oscillator frequency		94	125	156	kHz

Notes:

1. This value is the deviation from target frequency.
2. This value is applicable only when the main clock oscillation is stable.
3. This value is the time until the PLF1 bit in the PLS register becomes 1.

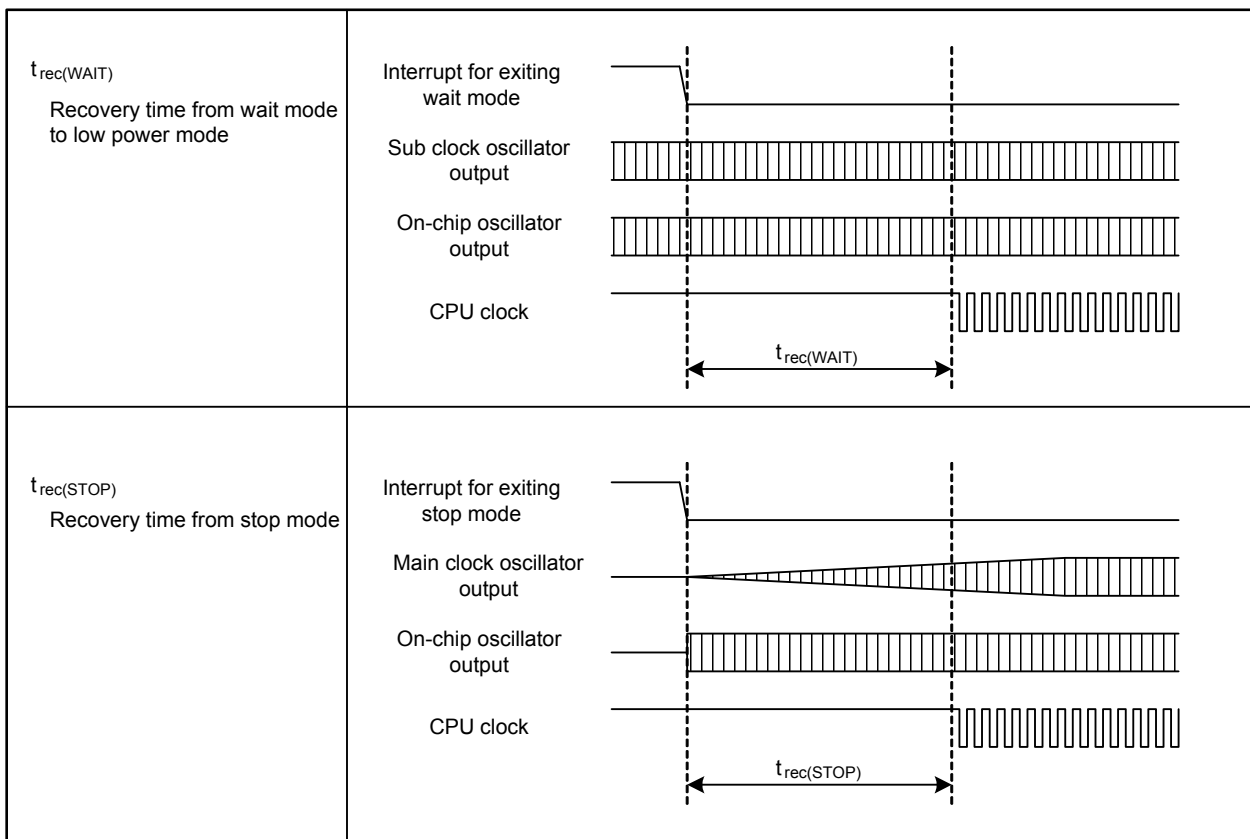


**Table 28.14 Electrical Characteristics of Clock Circuitry**  
 ( $V_{CC} = 3.0$  to  $5.5$  V,  $V_{SS} = 0$  V, and  $T_a = T_{opr}$ , unless otherwise noted)

Symbol	Characteristics	Measurement Condition	Value			Unit
			Min.	Typ.	Max.	
$t_{rec(WAIT)}$	Recovery time from wait mode to low power mode				225	$\mu s$
$t_{rec(STOP)}$	Recovery time from stop mode (1)				225	$\mu s$

Note:

- The stop mode recovery time does not include the main clock oscillation stabilization time. The CPU starts operating before the oscillator is stabilized.

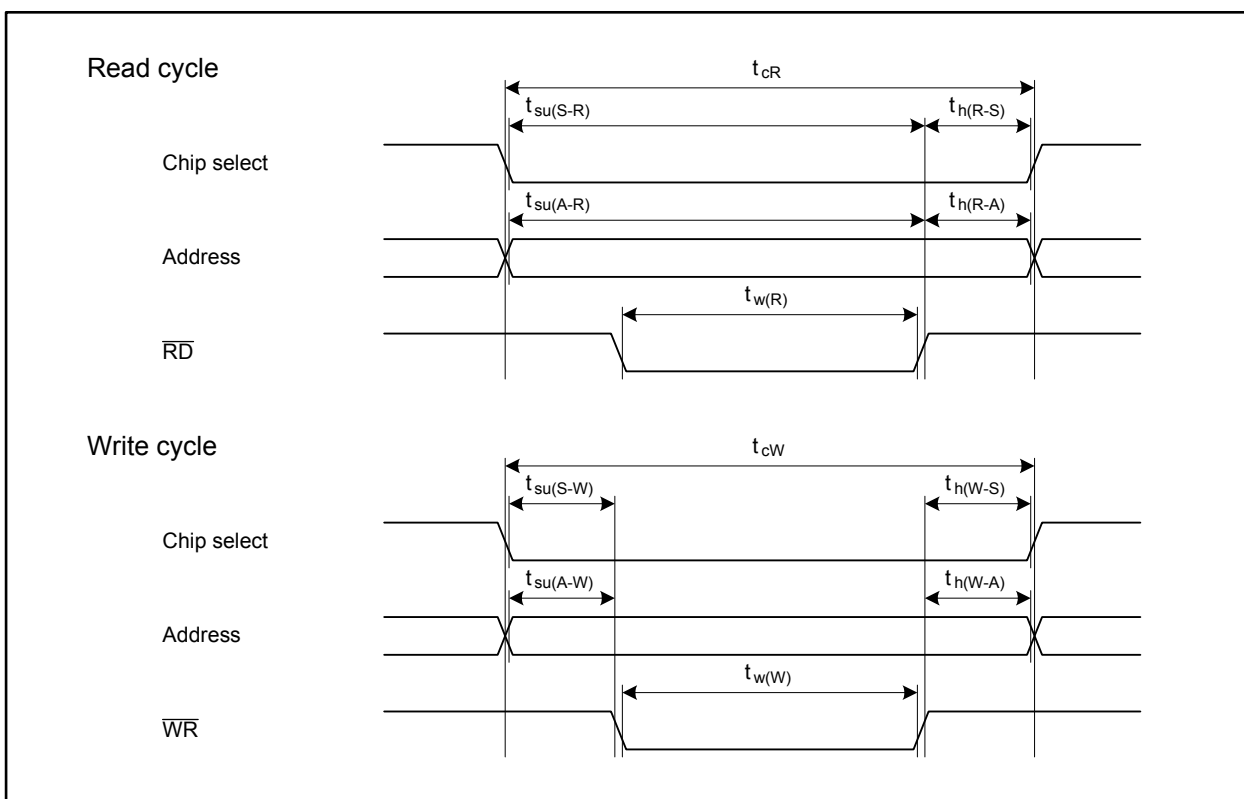


**Figure 28.4 Clock Circuit Timing**

Timing Requirements ( $V_{CC} = 3.0$  to  $5.5$  V,  $V_{SS} = 0$  V, and  $T_a = T_{opr}$ , unless otherwise noted)

**Table 28.15 Flash Memory CPU Rewrite Mode Timing**

Symbol	Characteristics	Value		Unit
		Min.	Max.	
$t_{cR}$	Read cycle time	200		ns
$t_{su(S-R)}$	Chip-select setup time before read	200		ns
$t_{h(R-S)}$	Chip-select hold time after read	0		ns
$t_{su(A-R)}$	Address setup time before read	200		ns
$t_{h(R-A)}$	Address hold time after read	0		ns
$t_{w(R)}$	Read pulse width	100		ns
$t_{cW}$	Write cycle time	200		ns
$t_{su(S-W)}$	Chip-select setup time before write	0		ns
$t_{h(W-S)}$	Chip-select hold time after write	30		ns
$t_{su(A-W)}$	Address setup time before write	0		ns
$t_{h(W-A)}$	Address hold time after write	30		ns
$t_{w(W)}$	Write pulse width	50		ns



**Figure 28.5 Flash Memory CPU Rewrite Mode Timing**

$$V_{CC} = 5 \text{ V}$$

**Table 28.16 Electrical Characteristics (1/3)**

( $V_{CC} = 4.2$  to  $5.5 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ ,  $T_a = T_{opr}$ , and  $f_{(CPU)} = 48 \text{ MHz}$ , unless otherwise noted)

Symbol	Characteristic		Measurement Condition	Value			Unit
				Min.	Typ.	Max.	
V <sub>OH</sub>	High level output voltage	P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_7, P5_0 to P5_7, P6_5, P6_7, P7_0, P7_4 to P7_6, P8_2 to P8_4, P8_6, P8_7, P9_3 to P9_7	I <sub>OH</sub> = -5 mA	V <sub>CC</sub> - 2.0		V <sub>CC</sub>	V
		P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_7, P5_0 to P5_7, P6_5, P6_7, P7_0, P7_4 to P7_6, P8_2 to P8_4, P8_6, P8_7, P9_3 to P9_7	I <sub>OH</sub> = -200 μA	V <sub>CC</sub> - 0.3		V <sub>CC</sub>	V
V <sub>OL</sub>	Low level output voltage	P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_7, P5_0 to P5_7, P6_5, P6_7, P7_0, P7_4 to P7_6, P8_2 to P8_4, P8_6, P8_7, P9_3 to P9_7	I <sub>OL</sub> = 5 mA			2.0	V
		P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_7, P5_0 to P5_7, P6_5, P6_7, P7_0, P7_4 to P7_6, P8_2 to P8_4, P8_6, P8_7, P9_3 to P9_7	I <sub>OL</sub> = 200 μA			0.45	V

$$V_{CC} = 5 \text{ V}$$

Table 28.17 Electrical Characteristics (2/3)

 $(V_{CC} = 4.2 \text{ to } 5.5 \text{ V}, V_{SS} = 0 \text{ V}, T_a = T_{opr}, \text{ and } f_{(CPU)} = 48 \text{ MHz, unless otherwise noted})$ 

Symbol	Characteristic	Measurement Condition	Value			Unit
			Min.	Typ.	Max.	
$V_{T+} - V_{T-}$	Hysteresis	NMI, $\overline{\text{INT0}}$ to $\overline{\text{INT5}}$ , TA0IN to TA4IN, TA0OUT to TA4OUT, TB0IN to TB5IN, $\overline{\text{CTS0}}$ to $\overline{\text{CTS4}}$ , CLK0 to CLK4, RXD0 to RXD4, SCL0 to SCL2, SDA0 to SDA2, $\overline{\text{SS0}}$ to $\overline{\text{SS2}}$ , SRXD0 to SRXD2, $\overline{\text{ADTRG}}$ , IIO0_0 to IIO0_7, UD0A, UD0B, $\overline{\text{SCS0}}$ , SSCK0, SSI0, SSO0, LIN0IN, CAN0IN, CAN1IN, CAN0WU, CAN1WU	0.2		1.0	V
			$\overline{\text{RESET}}$	0.2		1.8
$I_{IH}$	High level input current	XIN, $\overline{\text{RESET}}$ , CNVSS, NSD, P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_7, P5_0 to P5_7, P6_5, P6_7, P7_0, P7_4 to P7_6, P8_2 to P8_7, P9_1, P9_3 to P9_7	$V_I = 5 \text{ V}$			1.0 $\mu\text{A}$
$I_{IL}$	Low level input current	XIN, $\overline{\text{RESET}}$ , CNVSS, NSD, P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_7, P5_0 to P5_7, P6_5, P6_7, P7_0, P7_4 to P7_6, P8_2 to P8_7, P9_1, P9_3 to P9_7	$V_I = 0 \text{ V}$			-1.0 $\mu\text{A}$
$R_{PULLUP}$	Pull-up resistor	P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_7, P5_0 to P5_7, P6_5, P6_7, P7_0, P7_4 to P7_6, P8_2 to P8_4, P8_6, P8_7, P9_1, P9_3 to P9_7	$V_I = 0 \text{ V}$			30 50 170 $\text{k}\Omega$
$R_{fXIN}$	Feedback resistor	XIN				1.5 $\text{M}\Omega$
$R_{fXCIN}$	Feedback resistor	XCIN				15 $\text{M}\Omega$

$$V_{CC} = 5 V$$

Table 28.18 Electrical Characteristics (3/3)

 $(V_{CC} = 4.2 \text{ to } 5.5 V, V_{SS} = 0 V, \text{ and } T_a = T_{opr}, \text{ unless otherwise noted})$ 

Symbol	Characteristic	Measurement Condition	Value			Unit	
			Min.	Typ.	Max.		
I <sub>CC</sub>	Power supply current	In single-chip mode, output pins are left open and others are connected to V <sub>SS</sub>	f <sub>(CPU)</sub> = 48 MHz, f <sub>(BCLK)</sub> = 24 MHz, f <sub>(XIN)</sub> = 8 MHz, Active: XIN, PLL, Stopped: XCIN, OCO		31	48	mA
		XIN-XOUT Drive strength: low	f <sub>(CPU)</sub> = f <sub>SO(PLL)</sub> /24 MHz, Active: PLL (self-oscillation), Stopped: XIN, XCIN, OCO		7		mA
		XCIN-XCOUT Drive strength: low	f <sub>(CPU)</sub> = f <sub>(BCLK)</sub> = f <sub>(XIN)</sub> /256 MHz, f <sub>(XIN)</sub> = 8 MHz, Active: XIN, Stopped: PLL, XCIN, OCO		1.2		mA
			f <sub>(CPU)</sub> = f <sub>(BCLK)</sub> = 32.768 kHz, Active: XCIN, Stopped: XIN, PLL, OCO, Main regulator: shutdown		220		μA
			f <sub>(CPU)</sub> = f <sub>(BCLK)</sub> = f <sub>(OCO)</sub> /4 kHz, Active: OCO, Stopped: XIN, PLL, XCIN, Main regulator: shutdown		230		μA
			f <sub>(CPU)</sub> = f <sub>(BCLK)</sub> = f <sub>(XIN)</sub> /256 MHz, f <sub>(XIN)</sub> = 8 MHz, Active: XIN, Stopped: PLL, XCIN, OCO, T <sub>a</sub> = 25°C, Wait mode		960	1600	μA
			f <sub>(CPU)</sub> = f <sub>(BCLK)</sub> = 32.768 kHz, Active: XCIN, Stopped: XIN, PLL, OCO, Main regulator: shutdown, T <sub>a</sub> = 25°C, Wait mode		8	140	μA
			f <sub>(CPU)</sub> = f <sub>(BCLK)</sub> = f <sub>(OCO)</sub> /4 kHz, Active: OCO, Stopped: XIN, PLL, XCIN, Main regulator: shutdown, T <sub>a</sub> = 25°C, Wait mode		10	150	μA
			Stopped: all clocks, Main regulator: shutdown, T <sub>a</sub> = 25°C		5	70	μA
			Stopped: all clocks, Main regulator: shutdown, T <sub>a</sub> = 85°C			400	μA
	Stopped: all clocks, Main regulator: shutdown, T <sub>a</sub> = 105°C			1200	μA		
	Stopped: all clocks, Main regulator: shutdown, T <sub>a</sub> = 125°C			2000	μA		

$$V_{CC} = 5 \text{ V}$$

**Table 28.19 A/D Conversion Characteristics ( $V_{CC} = AV_{CC} = V_{REF} = 4.2$  to  $5.5 \text{ V}$ ,  $V_{SS} = AV_{SS} = 0 \text{ V}$ ,  $T_a = T_{opr}$ , and  $f_{(BCLK)} = 24 \text{ MHz}$ , unless otherwise noted)**

Symbol	Characteristic	Measurement Condition	Value			Unit
			Min.	Typ.	Max.	
—	Resolution	$V_{REF} = V_{CC}$			10	Bits
—	Absolute error	$V_{REF} = V_{CC} = 5 \text{ V}$ AN_0 to AN_4, AN0_0 to AN0_7, AN2_0 to AN2_7, ANEX0, ANEX1			$\pm 3$	LSB
					$\pm 7$	LSB
INL	Integral non-linearity error	$V_{REF} = V_{CC} = 5 \text{ V}$ AN_0 to AN_4, AN0_0 to AN0_7, AN2_0 to AN2_7, ANEX0, ANEX1			$\pm 3$	LSB
					$\pm 7$	LSB
DNL	Differential non-linearity error			$\pm 1$	LSB	
—	Offset error			$\pm 3$	LSB	
—	Gain error			$\pm 3$	LSB	
$R_{LADDER}$	Resistor ladder	$V_{REF} = V_{CC}$	4		20	$k\Omega$
$t_{CONV}$	Conversion time (10 bits)	$\phi_{AD} = 16 \text{ MHz}$ , with sample and hold function	2.06			$\mu\text{s}$
		$\phi_{AD} = 16 \text{ MHz}$ , without sample and hold function	3.69			$\mu\text{s}$
$t_{CONV}$	Conversion time (8 bits)	$\phi_{AD} = 16 \text{ MHz}$ , with sample and hold function	1.75			$\mu\text{s}$
		$\phi_{AD} = 16 \text{ MHz}$ , without sample and hold function	3.06			$\mu\text{s}$
$t_{SAMP}$	Sampling time	$\phi_{AD} = 16 \text{ MHz}$	0.188			$\mu\text{s}$
$V_{IA}$	Analog input voltage		0		$V_{REF}$	V
$\phi_{AD}$	Operating clock frequency	Without sample and hold function	0.25		16	MHz
		With sample and hold function	1		16	MHz
$R_{PU(AST)}$	Pull-up resistor for open-circuit detection		5	10	15	$k\Omega$
$R_{PD(AST)}$	Pull-down resistor for open-circuit detection		5	10	15	$k\Omega$

$$V_{CC} = 5 \text{ V}$$

Timing Requirements ( $V_{CC} = 4.2$  to  $5.5 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ , and  $T_a = T_{opr}$ , unless otherwise noted)

**Table 28.20 External Clock Input**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$t_{C(X)}$	External clock input period	125	250	ns
$t_{W(XH)}$	External clock input high level pulse width	50		ns
$t_{W(XL)}$	External clock input low level pulse width	50		ns
$t_{R(X)}$	External clock input rise time		5	ns
$t_{F(X)}$	External clock input fall time		5	ns
$t_W / t_C$	External clock input duty	40	60	%

$$V_{CC} = 5 \text{ V}$$

Timing Requirements ( $V_{CC} = 4.2$  to  $5.5 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ , and  $T_a = T_{opr}$ , unless otherwise noted)

**Table 28.21 Timer A Input (counting input in event counter mode)**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$t_{C(TA)}$	TAiIN input clock cycle time	200		ns
$t_{W(TAH)}$	TAiIN input high level pulse width	80		ns
$t_{W(TAL)}$	TAiIN input low level pulse width	80		ns

**Table 28.22 Timer A Input (gating input in timer mode)**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$t_{C(TA)}$	TAiIN input clock cycle time	400		ns
$t_{W(TAH)}$	TAiIN input high level pulse width	180		ns
$t_{W(TAL)}$	TAiIN input low level pulse width	180		ns

**Table 28.23 Timer A Input (external trigger input in one-shot timer mode)**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$t_{C(TA)}$	TAiIN input clock cycle time	200		ns
$t_{W(TAH)}$	TAiIN input high level pulse width	80		ns
$t_{W(TAL)}$	TAiIN input low level pulse width	80		ns

**Table 28.24 Timer A Input (external trigger input in pulse-width modulation mode)**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$t_{W(TAH)}$	TAiIN input high level pulse width	80		ns
$t_{W(TAL)}$	TAiIN input low level pulse width	80		ns

**Table 28.25 Timer A Input (increment/decrement switching input in event counter mode)**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$t_{C(UP)}$	TAiOUT input clock cycle time	2000		ns
$t_{W(UPH)}$	TAiOUT input high level pulse width	1000		ns
$t_{W(UPL)}$	TAiOUT input low level pulse width	1000		ns
$t_{Su(UP-TIN)}$	TAiOUT input setup time	400		ns
$t_h(TIN-UP)$	TAiOUT input hold time	400		ns



$$V_{CC} = 5 \text{ V}$$

Timing Requirements ( $V_{CC} = 4.2$  to  $5.5 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ , and  $T_a = T_{opr}$ , unless otherwise noted)

**Table 28.26 Timer B Input (counting input in event counter mode)**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$t_{C(TB)}$	TBiIN input clock cycle time (one edge counting)	200		ns
$t_{W(TBH)}$	TBiIN input high level pulse width (one edge counting)	80		ns
$t_{W(TBL)}$	TBiIN input low level pulse width (one edge counting)	80		ns
$t_{C(TB)}$	TBiIN input clock cycle time (both edges counting)	200		ns
$t_{W(TBH)}$	TBiIN input high level pulse width (both edges counting)	80		ns
$t_{W(TBL)}$	TBiIN input low level pulse width (both edges counting)	80		ns

**Table 28.27 Timer B Input (pulse period measure mode)**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$t_{C(TB)}$	TBiIN input clock cycle time	400		ns
$t_{W(TBH)}$	TBiIN input high level pulse width	180		ns
$t_{W(TBL)}$	TBiIN input low level pulse width	180		ns

**Table 28.28 Timer B Input (pulse-width measure mode)**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$t_{C(TB)}$	TBiIN input clock cycle time	400		ns
$t_{W(TBH)}$	TBiIN input high level pulse width	180		ns
$t_{W(TBL)}$	TBiIN input low level pulse width	180		ns

$$V_{CC} = 5 \text{ V}$$

Timing Requirements ( $V_{CC} = 4.2$  to  $5.5 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ , and  $T_a = T_{opr}$ , unless otherwise noted)

**Table 28.29 Serial Interface**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$t_{C(CK)}$	CLKi input clock cycle time	200		ns
$t_{W(CKH)}$	CLKi input high level pulse width	80		ns
$t_{W(CKL)}$	CLKi input low level pulse width	80		ns
$t_{su(D-C)}$	RXD <sub>i</sub> input setup time	80		ns
$t_{h(C-D)}$	RXD <sub>i</sub> input hold time	90		ns

**Table 28.30 A/D Trigger Input**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$t_{W(ADH)}$	ADTRG <sub>i</sub> input high level pulse width Hardware trigger input high level pulse width	$\frac{3}{\phi_{AD}}$		ns
$t_{W(ADL)}$	ADTRG <sub>i</sub> input low level pulse width Hardware trigger input high level pulse width	125		ns

**Table 28.31 External Interrupt  $\overline{INT}_i$  Input**

Symbol	Characteristic		Value		Unit
			Min.	Max.	
$t_{W(INH)}$	$\overline{INT}_i$ input high level pulse width (1)	Edge sensitive	250		ns
		Level sensitive	$t_{C(CPU)} + 200$		ns
$t_{W(INL)}$	$\overline{INT}_i$ input low level pulse width (1)	Edge sensitive	250		ns
		Level sensitive	$t_{C(CPU)} + 200$		ns

Note:

1. The values are applied in case the filtering function is disabled.

$$V_{CC} = 5 \text{ V}$$

Timing Requirements ( $V_{CC} = 4.2$  to  $5.5 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ , and  $T_a = T_{opr}$ , unless otherwise noted)

**Table 28.32 Serial Bus Interface**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$f_{(SSCK)}$	SSCKi frequency		4	MHz
$t_{c(SSCK)}$	SSCKi clock cycle time	250		ns
$t_{w(SSCKH)}$	SSCKi input high level pulse width	$0.35 \times t_{c(SSCK)}$	$0.6 \times t_{c(SSCK)}$	ns
$t_{w(SSCKL)}$	SSCKi input low level pulse width	$0.35 \times t_{c(SSCK)}$	$0.6 \times t_{c(SSCK)}$	ns
$t_{r(SSCK)}$	SSCKi input rising time		1	$\mu\text{s}$
$t_{f(SSCK)}$	SSCKi input falling time		1	$\mu\text{s}$
$t_{su(SCS-SSCK)}$	SCSi input setup time	$t_{c(BCLK)} + 50$		ns
$t_{h(SSCK-SCS)}$	SCSi input hold time	$t_{c(BCLK)} + 50$		ns
$t_{su(SSI-SSCK)}$	SSI input setup time	80		ns
$t_{h(SSCK-SSI)}$	SSI input hold time	10		ns
$t_{su(SSO-SSCK)}$	SSO input setup time	80		ns
$t_{h(SSCK-SSO)}$	SSO input hold time	20		ns

$$V_{CC} = 5 \text{ V}$$

Switching Characteristics ( $V_{CC} = 4.2$  to  $5.5 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ , and  $T_a = T_{opr}$ , unless otherwise noted)

**Table 28.33 Serial Interface**

Symbol	Characteristic	Measurement Condition	Value		Unit
			Min.	Max.	
$t_{d(C-Q)}$	TXDi output delay time	Refer to Figure 28.6		80	ns
$t_{h(C-Q)}$	TXDi output hold time		0		ns

**Table 28.34 Serial Bus Interface**

Symbol	Characteristic	Measurement Condition	Value		Unit
			Min.	Max.	
$t_{w(SSCKH)}$	SSCKi output high level pulse width	Refer to Figure 28.6	$0.35 \times t_{c(SSCK)}$	$0.6 \times t_{c(SSCK)}$	ns
$t_{w(SSCKL)}$	SSCKi output low level pulse width		$0.35 \times t_{c(SSCK)}$	$0.6 \times t_{c(SSCK)}$	ns
$t_{r(SSCK)}$	SSCKi output rising time			20	ns
$t_{f(SSCK)}$	SSCKi output falling time			20	ns
$t_{d(SCS-SSCK)}$	SSCKi output delay time for SCSi			$0.5 \times t_{c(SSCK)} + 20$	ns
$t_{d(SSCK-SCS)}$	SCSi output delay time for SSCKi			$0.5 \times t_{c(SSCK)} - 20$	ns
$t_{en(SCS-SSO)}$	SSOi output enable time			$1.5 \times t_{c(BCLK)} + 100$	ns
$t_{dis(SCS-SSO)}$	SSOi output disable time			$1.5 \times t_{c(BCLK)} + 100$	ns
$t_{en(SCS-SSI)}$	SSLi output enable time			$1.5 \times t_{c(BCLK)} + 100$	ns
$t_{dis(SCS-SSI)}$	SSLi output disable time			$1.5 \times t_{c(BCLK)} + 100$	ns
$t_{d(SSCK-SSO)}$	SSOi output delay time for SSCKi			30	ns
$t_{d(SSCK-SSI)}$	SSLi output delay time for SSCKi			85	ns
$t_{rec(SCS)}$	SCSi output high level period in continuous transmission			$0.625 \times t_{c(SSCK)}$	ns

$$V_{CC} = 3.3 \text{ V}$$

**Table 28.35 Electrical Characteristics (1/3) ( $V_{CC} = 3.0$  to  $3.6 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ ,  $T_a = T_{opr}$ , and  $f_{(CPU)} = 48 \text{ MHz}$ , unless otherwise noted)**

Symbol	Characteristic		Measurement Condition	Value			Unit
				Min.	Typ.	Max.	
$V_{OH}$	High level output voltage	P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_7, P5_0 to P5_7, P6_5, P6_7, P7_0, P7_4 to P7_6, P8_2 to P8_4, P8_6, P8_7, P9_3 to P9_7	$I_{OH} = -1 \text{ mA}$	$V_{CC} - 0.6$		$V_{CC}$	V
$V_{OL}$	Low level output voltage	P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_7, P5_0 to P5_7, P6_5, P6_7, P7_0, P7_4 to P7_6, P8_2 to P8_4, P8_6, P8_7, P9_3 to P9_7	$I_{OL} = 1 \text{ mA}$			0.5	V

$$V_{CC} = 3.3 \text{ V}$$

**Table 28.36 Electrical Characteristics (2/3) ( $V_{CC} = 3.0$  to  $3.6 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ ,  $T_a = T_{opr}$ , and  $f_{(CPU)} = 48 \text{ MHz}$ , unless otherwise noted)**

Symbol	Characteristic		Measurement Condition	Value			Unit
				Min.	Typ.	Max.	
$V_{T+} - V_{T-}$	Hysteresis	NMI, $\overline{\text{INT0}}$ to $\overline{\text{INT5}}$ , TA0IN to TA4IN, TA0OUT to TA4OUT, TB0IN to TB5IN, $\overline{\text{CTS0}}$ to $\overline{\text{CTS4}}$ , CLK0 to CLK4, RXD0 to RXD4, SCL0 to SCL2, SDA0 to SDA2, $\overline{\text{SS0}}$ to $\overline{\text{SS2}}$ , SRXD0 to SRXD2, $\overline{\text{ADTRG}}$ , IIO0_0 to IIO0_7, UD0A, UD0B, $\overline{\text{SCS0}}$ , SSCK0, SSI0, SSO0, LIN0IN, CAN0IN, CAN1IN, $\overline{\text{CAN0WU}}$ , $\overline{\text{CAN1WU}}$		0.2		1.0	V
		$\overline{\text{RESET}}$		0.2		1.8	V
$I_{IH}$	High level input current	XIN, $\overline{\text{RESET}}$ , CNVSS, NSD, P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_7, P5_0 to P5_7, P6_5, P6_7, P7_0, P7_4 to P7_6, P8_2 to P8_7, P9_1, P9_3 to P9_7	$V_I = 3.3 \text{ V}$			1.0	$\mu\text{A}$
$I_{IL}$	Low level input current	XIN, $\overline{\text{RESET}}$ , CNVSS, NSD, P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_7, P5_0 to P5_7, P6_5, P6_7, P7_0, P7_4 to P7_6, P8_2 to P8_7, P9_1, P9_3 to P9_7	$V_I = 0 \text{ V}$			-1.0	$\mu\text{A}$
$R_{PULLUP}$	Pull-up resistor	P0_0 to P0_7, P1_0 to P1_7, P2_0 to P2_7, P3_0 to P3_7, P4_0 to P4_7, P5_0 to P5_7, P6_5, P6_7, P7_0, P7_4 to P7_6, P8_2 to P8_4, P8_6, P8_7, P9_1, P9_3 to P9_7	$V_I = 0 \text{ V}$	50	100	500	$\text{k}\Omega$
$R_{fXIN}$	Feedback resistor	XIN			3		$\text{M}\Omega$
$R_{fXCIN}$	Feedback resistor	XCIN			25		$\text{M}\Omega$

$$V_{CC} = 3.3 \text{ V}$$

Table 28.37 Electrical Characteristics (3/3)

( $V_{CC} = 3.0$  to  $3.6 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ , and  $T_a = T_{opr}$ , unless otherwise noted)

Symbol	Characteristic	Measurement Condition	Value			Unit	
			Min.	Typ.	Max.		
$I_{CC}$	Power supply current	In single-chip mode, output pins are left open and others are connected to $V_{SS}$	$f_{(CPU)} = 48 \text{ MHz}$ , $f_{(BCLK)} = 24 \text{ MHz}$ , $f_{(XIN)} = 8 \text{ MHz}$ , Active: XIN, PLL, Stopped: XCIN, OCO		31	48	mA
		XIN-XOUT Drive strength: low	$f_{(CPU)} = f_{SO(PLL)}/24 \text{ MHz}$ , Active: PLL (self-oscillation), Stopped: XIN, XCIN, OCO		7		mA
		XCIN-XCOUT Drive strength: low	$f_{(CPU)} = f_{(BCLK)} = f_{(XIN)}/256 \text{ MHz}$ , $f_{(XIN)} = 8 \text{ MHz}$ , Active: XIN, Stopped: PLL, XCIN, OCO		670		$\mu\text{A}$
			$f_{(CPU)} = f_{(BCLK)} = 32.768 \text{ kHz}$ , Active: XCIN, Stopped: XIN, PLL, OCO, Main regulator: shutdown		180		$\mu\text{A}$
			$f_{(CPU)} = f_{(BCLK)} = f_{(OCO)}/4 \text{ kHz}$ , Active: OCO, Stopped: XIN, PLL, XCIN, Main regulator: shutdown		190		$\mu\text{A}$
			$f_{(CPU)} = f_{(BCLK)} = f_{(XIN)}/256 \text{ MHz}$ , $f_{(XIN)} = 8 \text{ MHz}$ , Active: XIN, Stopped: PLL, XCIN, OCO, $T_a = 25^\circ\text{C}$ , Wait mode		500	900	$\mu\text{A}$
			$f_{(CPU)} = f_{(BCLK)} = 32.768 \text{ kHz}$ , Active: XCIN, Stopped: XIN, PLL, OCO, Main regulator: shutdown, $T_a = 25^\circ\text{C}$ , Wait mode		8	140	$\mu\text{A}$
			$f_{(CPU)} = f_{(BCLK)} = f_{(OCO)}/4 \text{ kHz}$ , Active: OCO, Stopped: XIN, PLL, XCIN, Main regulator: shutdown, $T_a = 25^\circ\text{C}$ , Wait mode		10	150	$\mu\text{A}$
			Stopped: all clocks, Main regulator: shutdown, $T_a = 25^\circ\text{C}$		5	70	$\mu\text{A}$
			Stopped: all clocks, Main regulator: shutdown, $T_a = 85^\circ\text{C}$			400	$\mu\text{A}$
			Stopped: all clocks, Main regulator: shutdown, $T_a = 105^\circ\text{C}$			1200	$\mu\text{A}$
	Stopped: all clocks, Main regulator: shutdown, $T_a = 125^\circ\text{C}$			2000	$\mu\text{A}$		

$$V_{CC} = 3.3 \text{ V}$$

**Table 28.38 A/D Conversion Characteristics ( $V_{CC} = AV_{CC} = V_{REF} = 3.0$  to  $3.6 \text{ V}$ ,  $V_{SS} = AV_{SS} = 0 \text{ V}$ ,  $T_a = T_{opr}$ , and  $f_{(BCLK)} = 24 \text{ MHz}$ , unless otherwise noted)**

Symbol	Characteristic	Measurement Condition	Value			Unit	
			Min.	Typ.	Max.		
—	Resolution	$V_{REF} = V_{CC}$			10	Bits	
—	Absolute error	$V_{REF} = V_{CC} = 3.3 \text{ V}$	AN_0 to AN_4, AN0_0 to AN0_7, AN2_0 to AN2_7, ANEX0, ANEX1			$\pm 5$	LSB
			External op-amp connection mode			$\pm 7$	LSB
INL	Integral non-linearity error	$V_{REF} = V_{CC} = 3.3 \text{ V}$	AN_0 to AN_4, AN0_0 to AN0_7, AN2_0 to AN2_7, ANEX0, ANEX1			$\pm 5$	LSB
			External op-amp connection mode			$\pm 7$	LSB
DNL	Differential non- linearity error	$V_{REF} = V_{CC} = 3.3 \text{ V}$			$\pm 1$	LSB	
—	Offset error				$\pm 3$	LSB	
—	Gain error				$\pm 3$	LSB	
$R_{LADDER}$	Resistor ladder	$V_{REF} = V_{CC}$	4		20	$k\Omega$	
$t_{CONV}$	Conversion time (10 bits)	$\phi_{AD} = 10 \text{ MHz}$ , with sample and hold function	3.3			$\mu\text{s}$	
$t_{CONV}$	Conversion time (8 bits)	$\phi_{AD} = 10 \text{ MHz}$ , with sample and hold function	2.8			$\mu\text{s}$	
$t_{SAMP}$	Sampling time	$\phi_{AD} = 10 \text{ MHz}$	0.3			$\mu\text{s}$	
$V_{IA}$	Analog input voltage		0		$V_{REF}$	V	
$\phi_{AD}$	Operating clock frequency	Without sample and hold function	0.25		10	MHz	
		With sample and hold function	1		10	MHz	
$R_{PU(AST)}$	Pull-up resistor for open-circuit detection		5	10	15	$k\Omega$	
$R_{PD(AST)}$	Pull-down resistor for open-circuit detection		5	10	15	$k\Omega$	



$$V_{CC} = 3.3 \text{ V}$$

Timing Requirements ( $V_{CC} = 3.0$  to  $3.6 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ , and  $T_a = T_{opr}$ , unless otherwise noted)

**Table 28.39 External Clock Input**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$t_{C(X)}$	External clock input period	125	250	ns
$t_{W(XH)}$	External clock input high level pulse width	50		ns
$t_{W(XL)}$	External clock input low level pulse width	50		ns
$t_{r(X)}$	External clock input rise time		5	ns
$t_{f(X)}$	External clock input fall time		5	ns
$t_W / t_C$	External clock input duty	40	60	%

$$V_{CC} = 3.3 \text{ V}$$

Timing Requirements ( $V_{CC} = 3.0$  to  $3.6 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ , and  $T_a = T_{opr}$ , unless otherwise noted)

**Table 28.40 Timer A Input (counting input in event counter mode)**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$t_{c(TA)}$	TAiIN input clock cycle time	200		ns
$t_{w(TAH)}$	TAiIN input high level pulse width	80		ns
$t_{w(TAL)}$	TAiIN input low level pulse width	80		ns

**Table 28.41 Timer A Input (gating input in timer mode)**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$t_{c(TA)}$	TAiIN input clock cycle time	400		ns
$t_{w(TAH)}$	TAiIN input high level pulse width	180		ns
$t_{w(TAL)}$	TAiIN input low level pulse width	180		ns

**Table 28.42 Timer A Input (external trigger input in one-shot timer mode)**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$t_{c(TA)}$	TAiIN input clock cycle time	200		ns
$t_{w(TAH)}$	TAiIN input high level pulse width	80		ns
$t_{w(TAL)}$	TAiIN input low level pulse width	80		ns

**Table 28.43 Timer A Input (external trigger input in pulse-width modulation mode)**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$t_{w(TAH)}$	TAiIN input high level pulse width	80		ns
$t_{w(TAL)}$	TAiIN input low level pulse width	80		ns

**Table 28.44 Timer A Input (increment/decrement switching input in event counter mode)**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$t_{c(UP)}$	TAiOUT input clock cycle time	2000		ns
$t_{w(UPH)}$	TAiOUT input high level pulse width	1000		ns
$t_{w(UPL)}$	TAiOUT input low level pulse width	1000		ns
$t_{su(UP-TIN)}$	TAiOUT input setup time	400		ns
$t_h(TIN-UP)$	TAiOUT input hold time	400		ns

$$V_{CC} = 3.3 \text{ V}$$

Timing Requirements ( $V_{CC} = 3.0$  to  $3.6 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ , and  $T_a = T_{opr}$ , unless otherwise noted)

**Table 28.45 Timer B Input (counting input in event counter mode)**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$t_{c(TB)}$	TBiIN input clock cycle time (one edge counting)	200		ns
$t_{w(TBH)}$	TBiIN input high level pulse width (one edge counting)	80		ns
$t_{w(TBL)}$	TBiIN input low level pulse width (one edge counting)	80		ns
$t_{c(TB)}$	TBiIN input clock cycle time (both edges counting)	200		ns
$t_{w(TBH)}$	TBiIN input high level pulse width (both edges counting)	80		ns
$t_{w(TBL)}$	TBiIN input low level pulse width (both edges counting)	80		ns

**Table 28.46 Timer B Input (pulse period measure mode)**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$t_{c(TB)}$	TBiIN input clock cycle time	400		ns
$t_{w(TBH)}$	TBiIN input high level pulse width	180		ns
$t_{w(TBL)}$	TBiIN input low level pulse width	180		ns

**Table 28.47 Timer B Input (pulse-width measure mode)**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$t_{c(TB)}$	TBiIN input clock cycle time	400		ns
$t_{w(TBH)}$	TBiIN input high level pulse width	180		ns
$t_{w(TBL)}$	TBiIN input low level pulse width	180		ns

$$V_{CC} = 3.3 \text{ V}$$

Timing Requirements ( $V_{CC} = 3.0$  to  $3.6 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ , and  $T_a = T_{opr}$ , unless otherwise noted)

**Table 28.48 Serial Interface**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$t_{c(CK)}$	CLKi input clock cycle time	200		ns
$t_{w(CKH)}$	CLKi input high level pulse width	80		ns
$t_{w(CKL)}$	CLKi input low level pulse width	80		ns
$t_{su(D-C)}$	RXD <sub>i</sub> input setup time	80		ns
$t_{h(C-D)}$	RXD <sub>i</sub> input hold time	90		ns

**Table 28.49 A/D Trigger Input**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$t_{w(ADH)}$	ADTRG input high level pulse width Hardware trigger input high level pulse width	$\frac{3}{\phi_{AD}}$		ns
$t_{w(ADL)}$	ADTRG input low level pulse width Hardware trigger input high level pulse width	125		ns

**Table 28.50 External Interrupt  $\overline{INT}_i$  Input**

Symbol	Characteristic		Value		Unit
			Min.	Max.	
$t_{w(INH)}$	$\overline{INT}_i$ input high level pulse width (1)	Edge sensitive	250		ns
		Level sensitive	$t_{c(CPU)} + 200$		ns
$t_{w(INL)}$	$\overline{INT}_i$ input low level pulse width (1)	Edge sensitive	250		ns
		Level sensitive	$t_{c(CPU)} + 200$		ns

Note:

1. The values are applied in case filtering function is disabled.

$$V_{CC} = 3.3 \text{ V}$$

Timing Requirements ( $V_{CC} = 3.0$  to  $3.6 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ , and  $T_a = T_{opr}$ , unless otherwise noted)

**Table 28.51 Serial Bus Interface**

Symbol	Characteristic	Value		Unit
		Min.	Max.	
$f_{(SSCK)}$	SSCKi frequency		4	MHz
$t_{c(SSCK)}$	SSCKi clock cycle time	250		ns
$t_{w(SSCKH)}$	SSCKi input high level pulse width	$0.35 \times t_{c(SSCK)}$	$0.6 \times t_{c(SSCK)}$	ns
$t_{w(SSCKL)}$	SSCKi input low level pulse width	$0.35 \times t_{c(SSCK)}$	$0.6 \times t_{c(SSCK)}$	ns
$t_{r(SSCK)}$	SSCKi input rising time		1	$\mu\text{s}$
$t_{f(SSCK)}$	SSCKi input falling time		1	$\mu\text{s}$
$t_{su(SCS-SSCK)}$	SCSi input setup time	$t_{c(BCLK)} + 50$		ns
$t_{h(SSCK-SCS)}$	SCSi input hold time	$t_{c(BCLK)} + 50$		ns
$t_{su(SSI-SSCK)}$	SSI input setup time	100		ns
$t_{h(SSCK-SSI)}$	SSI input hold time	10		ns
$t_{su(SSO-SSCK)}$	SSO input setup time	100		ns
$t_{h(SSCK-SSO)}$	SSO input hold time	20		ns

$$V_{CC} = 3.3 \text{ V}$$

Switching Characteristics ( $V_{CC} = 3.0$  to  $3.6 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ , and  $T_a = T_{opr}$ , unless otherwise noted)

**Table 28.52 Serial Interface**

Symbol	Characteristic	Measurement Condition	Value		Unit
			Min.	Max.	
$t_{d(C-Q)}$	TXDi output delay time	Refer to Figure 28.6		80	ns
$t_{h(C-Q)}$	TXDi output hold time		0		ns

**Table 28.53 Serial Bus Interface**

Symbol	Characteristic	Measurement Condition	Value		Unit
			Min.	Max.	
$t_{w(SSCKH)}$	SSCKi output high level pulse width	Refer to Figure 28.6	$0.35 \times t_{c(SSCK)}$	$0.6 \times t_{c(SSCK)}$	ns
$t_{w(SSCKL)}$	SSCKi output low level pulse width		$0.35 \times t_{c(SSCK)}$	$0.6 \times t_{c(SSCK)}$	ns
$t_{r(SSCK)}$	SSCKi output rising time			35	ns
$t_{f(SSCK)}$	SSCKi output falling time			35	ns
$t_{d(SCS-SSCK)}$	SSCKi output delay time for SCSi			$0.5 \times t_{c(SSCK)} + 40$	ns
$t_{d(SSCK-SCS)}$	SCSi output delay time for SSCKi			$0.5 \times t_{c(SSCK)} - 40$	ns
$t_{en(SCS-SSO)}$	SSOi output enable time			$1.5 \times t_{c(BCLK)} + 100$	ns
$t_{dis(SCS-SSO)}$	SSOi output disable time			$1.5 \times t_{c(BCLK)} + 100$	ns
$t_{en(SCS-SSI)}$	SSLi output enable time			$1.5 \times t_{c(BCLK)} + 100$	ns
$t_{dis(SCS-SSI)}$	SSLi output disable time			$1.5 \times t_{c(BCLK)} + 100$	ns
$t_{d(SSCK-SSO)}$	SSOi output delay time for SSCKi			50	ns
$t_{d(SSCK-SSI)}$	SSLi output delay time for SSCKi			120	ns
$t_{rec(SCS)}$	SCSi output high level period in continuous transmission			$0.625 \times t_{c(SSCK)}$	ns

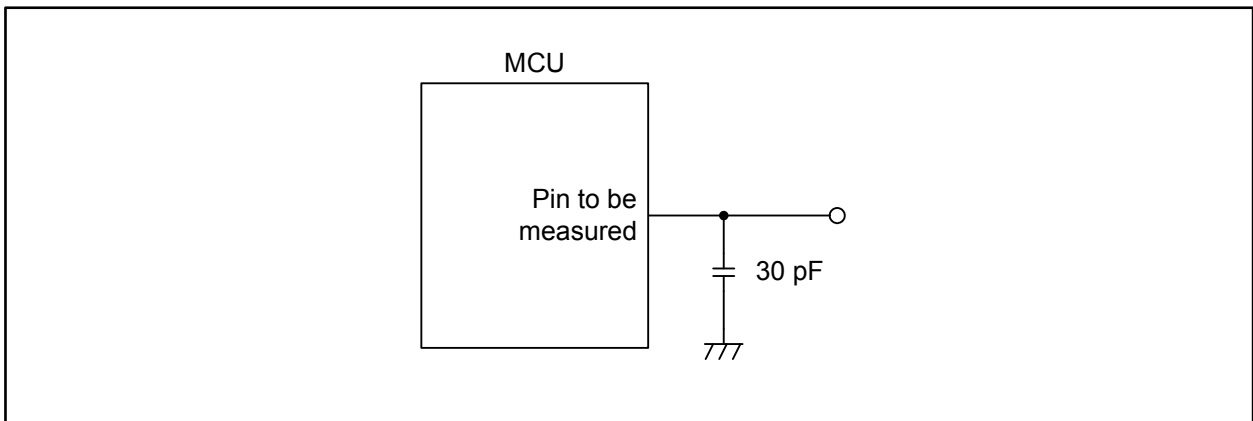


Figure 28.6 Switching Characteristic Measurement Circuit

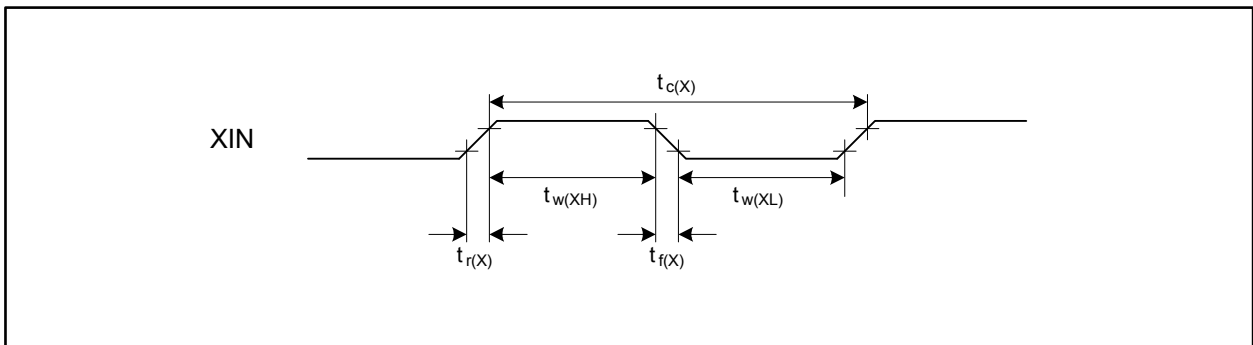


Figure 28.7 External Clock Input Timing

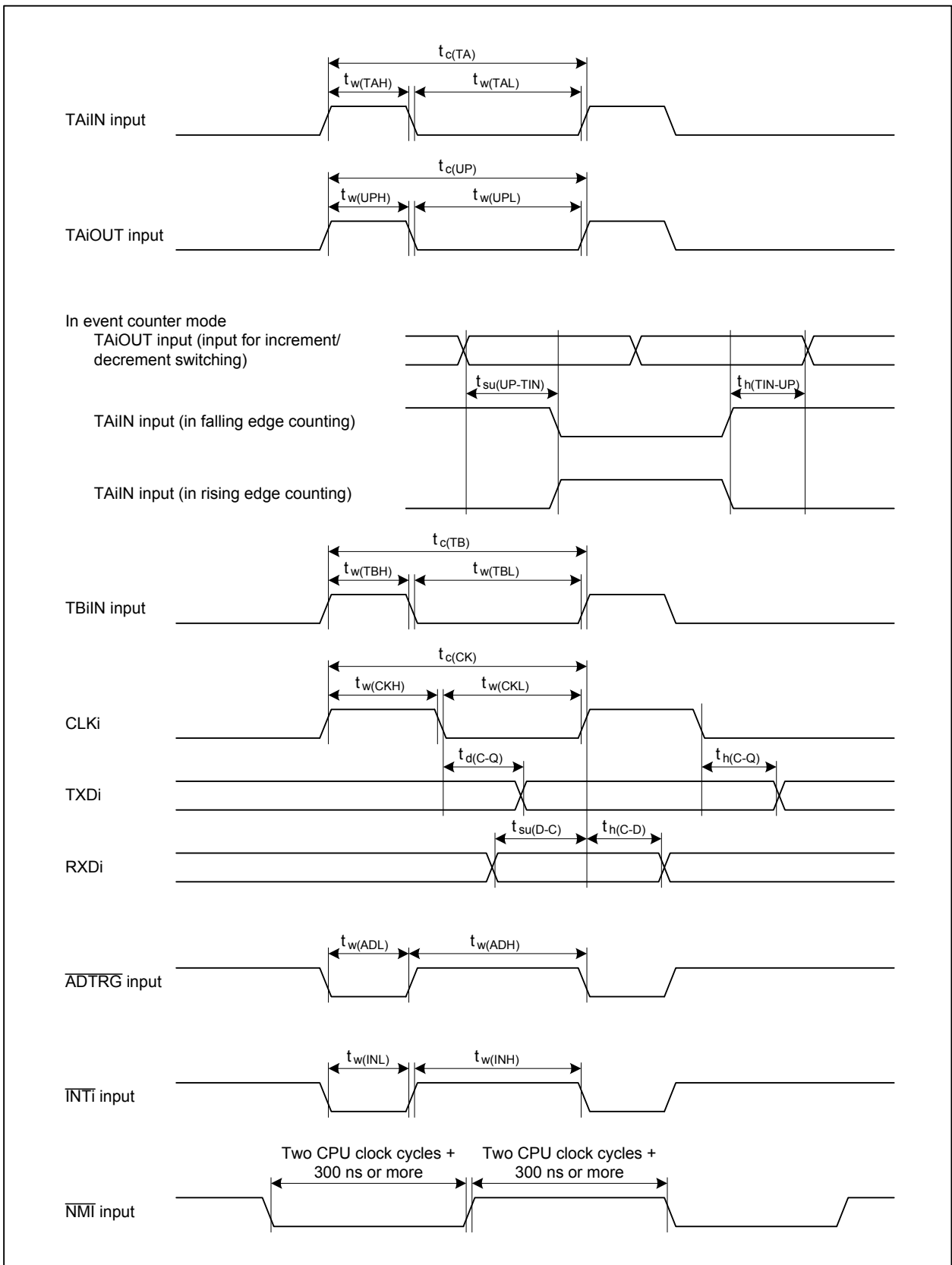


Figure 28.8 Timing of Peripherals



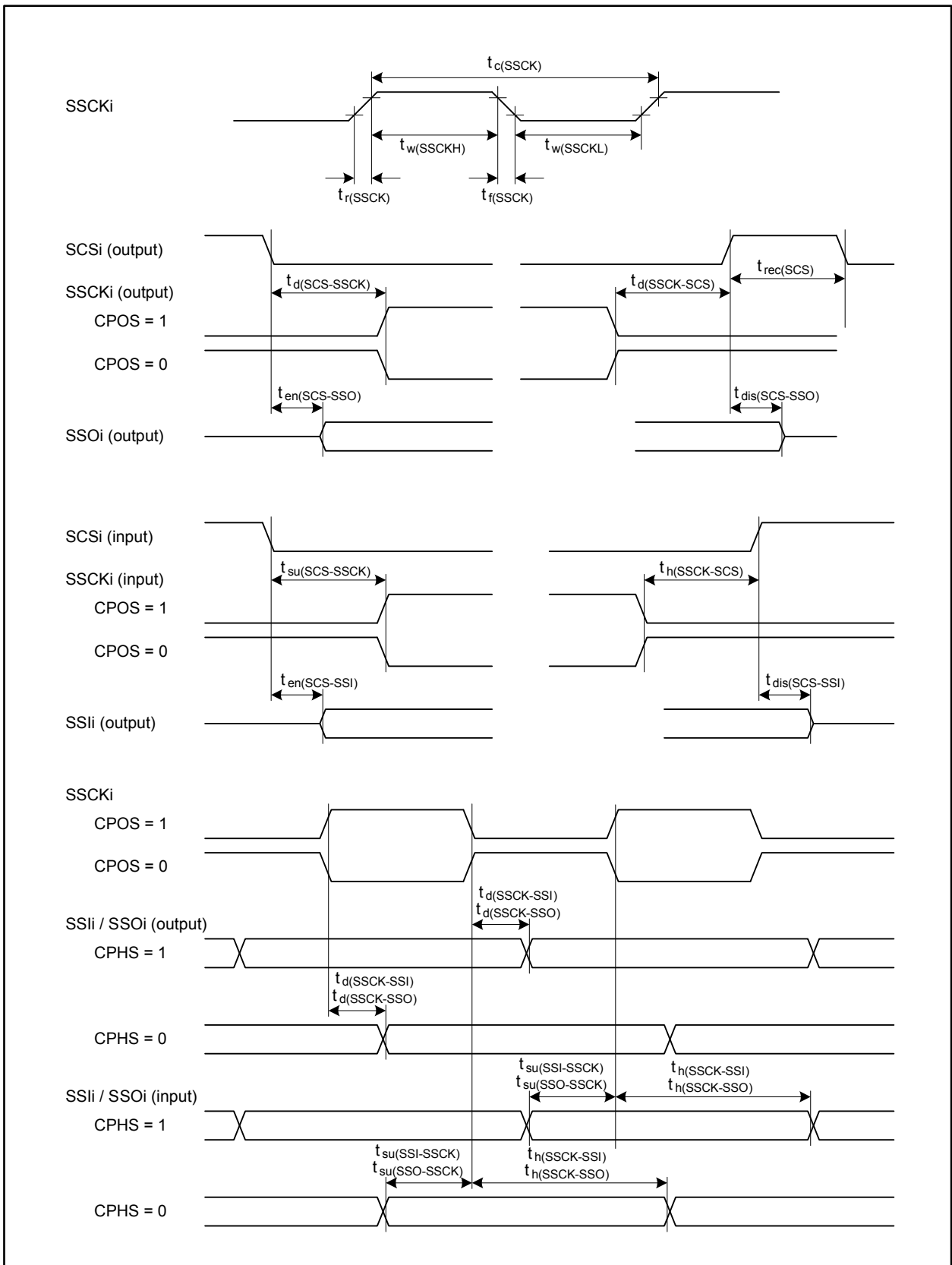


Figure 28.9 Timing of Serial Bus Interface

## 29. Usage Notes

### 29.1 Notes on Board Designing

#### 29.1.1 Power Supply Pins

The board should be designed so there is no potential difference between pins with the same name. Note the following points:

- Connect all VSS pins to the same GND. Traces for the pins should be as wide as physically possible so the same voltage can be applied to every VSS pin.
- Connect all VCC pins to the same power supply. Traces for the pins should be as wide as physically possible so the same voltage can be applied to every VCC pin.

Insert a capacitor between each VCC pin and the VSS pin to prevent operation errors due to noise. The capacitor should be beneficially effective at high and low frequencies and should have a capacitance of approximately 0.1  $\mu\text{F}$ . The traces for the capacitor and the power supply pins should be as short and wide as physically possible.

#### 29.1.2 Supply Voltage

The device is operationally guaranteed under operating conditions specified in electrical characteristics.

Drive the RESET pin low before the supply voltage becomes lower than the recommended value.

## 29.2 Notes on Register Setting

### 29.2.1 Registers with Write-only Bits

Read-modify-write instructions cannot be used when setting a register containing write-only bits. Read-modify-write instructions read a value of an address, modify the value, and write the modified value to the same address. Table 29.1 lists read-modify-write instructions, and Table 29.2 lists registers containing write-only bits. To set a new value by modifying the previous one, write the previous value into RAM as well as to the register, change the contents of the RAM and then transfer the new value to the register by the MOV instruction.

**Table 29.1 Read-modify-write Instructions**

Function	Mnemonic
Transfer	MOV <i>Dir</i>
Bit processing	BCLR, BMC <i>nd</i> , BNOT, BSET, BTSTC, and BTSTS
Shifting	ROLC, RORC, ROT, SHA, and SHL
Arithmetic operation	ABS, ADC, ADCF, ADD, ADSF, DEC, DIV, DIVU, DIVX, EXTS, EXTZ, INC, MUL, MULU, NEG, SBB, and SUB
Decimal operation	DADC, DADD, DSBB, and DSUB
Floating-point operation	ADDF, DIVF, MULF, and SUBF
Logical operation	AND, NOT, OR, and XOR

**Table 29.2 Registers with Write-only Bits**

Module	Register	Symbol	Address
Watchdog timer	Watchdog timer start register	WDTS	04404Eh
Timer A	Timer A0 register <sup>(1)</sup>	TA0	0347h-0346h
	Timer A1 register <sup>(1)</sup>	TA1	0349h-0348h
	Timer A2 register <sup>(1)</sup>	TA2	034Bh-034Ah
	Timer A3 register <sup>(1)</sup>	TA3	034Dh-034Ch
	Timer A4 register <sup>(1)</sup>	TA4	034Fh-034Eh
	Increment/decrement select register	UDF	0344h
Three-phase motor control timers	Timer B2 interrupt generating frequency set counter	ICTB2	030Dh
	Timer A1-1 register	TA11	0303h-0302h
	Timer A2-1 register	TA21	0305h-0304h
	Timer A4-1 register	TA41	0307h-0306h
	Timer A1 mirror register	TA1M	0221h-0220h
	Timer A2 mirror register	TA2M	0225h-0224h
	Timer A4 mirror register	TA4M	0229h-0228h
	Timer A1-1 mirror register	TA11M	0223h-0222h
	Timer A2-1 mirror register	TA21M	0227h-0226h
	Timer A4-1 mirror register	TA41M	022Bh-022Ah
	Dead time timer	DTT	030Ch
	Serial interface	UART0 bit rate register	U0BRG
UART1 bit rate register		U1BRG	02E9h
UART2 bit rate register		U2BRG	0339h
UART3 bit rate register		U3BRG	01E1h
UART4 bit rate register		U4BRG	01E9h
UART0 transmit buffer register		U0TB	036Bh-036Ah
UART1 transmit buffer register		U1TB	02EBh-02EAh
UART2 transmit buffer register		U2TB	033Bh-033Ah
UART3 transmit buffer register		U3TB	01E3h-01E2h
UART4 transmit buffer register		U4TB	01EBh-01EAh
CAN module	CAN0 receive FIFO pointer control register	C0RFPCR	047F49h
	CAN0 transmit FIFO pointer control register	C0TFPCR	047F4Bh
	CAN1 receive FIFO pointer control register	C1RFPCR	047B49h
	CAN1 transmit FIFO pointer control register	C1TFPCR	047B4Bh

Note:

1. The register has write-only bits in one-shot timer mode and pulse-width modulation mode.

## 29.3 Notes on Clock Generator

### 29.3.1 Sub Clock

#### 29.3.1.1 Oscillator Constant Matching

The constant matching of the sub clock oscillator should be evaluated in both cases when the drive strength is high and low.

Contact the oscillator manufacturer for details on the oscillation circuit constant matching.

### 29.3.2 Power Control

Do not switch the base clock source until the oscillation of the clock to be used has stabilized. However, this does not apply to the on-chip oscillator since it starts running immediately after the CM31 bit in the CM3 register is set to 1.

To switch the base clock source from the PLL clock to a low speed clock, use the MOV.L or OR.L instruction to set the BCS bit in the CCR register to 1.

- Program example in assembly language

```
OR.L    #80h, 0004h
```

- Program example in C language

```
asm("OR.L #80h, 0004h");
```

#### 29.3.2.1 Stop Mode

- To exit stop mode using a reset, apply a low signal to the  $\overline{\text{RESET}}$  pin until the main clock oscillation stabilizes.

#### 29.3.2.2 Suggestions for Power Saving

The following are suggestions to reduce power consumption when programming or designing systems.

- I/O pins:

If inputs are floating, both transistors may be conducting. Set unassigned pins to input mode and connect each of them to VSS via a resistor, or set them to output mode and leave them open.

- A/D converter:

When not performing the A/D conversion, set the VCUT bit in the AD0CON1 register to 0 (VREF disconnected). To perform the A/D conversion, set the VCUT bit to 1 (VREF connected) and wait at least 1  $\mu\text{s}$  before starting conversion.

- Peripheral clock stop:

When entering wait mode, power consumption can be reduced by setting the CM02 bit in the CM0 register to 1 to stop the peripheral clock source. However, this setting does not stop the fC32.

## 29.4 Notes on Interrupts

### 29.4.1 ISP Setting

The interrupt stack pointer (ISP) is initialized to 00000000h after a reset. Set a value to the ISP before an interrupt is accepted, otherwise the program may go out of control. A multiple of 4 should be set to the ISP, which enables faster interrupt sequence due to less memory access.

When using NMI, in particular, since this interrupt cannot be disabled, set the PM24 bit in the PM2 register to 1 (NMI enabled) after setting the ISP at the beginning of the program.

### 29.4.2 NMI

- NMI cannot be disabled once the PM24 bit in the PM2 register is set to 1 (NMI enabled). This bit setting should be done only when using NMI.
- When the PM24 bit in the PM2 register is 1 (NMI enabled), the P8\_5 bit in the P8 register is enabled just for monitoring the  $\overline{\text{NMI}}$  pin state. It is not enabled as a general port.

### 29.4.3 External Interrupts

- The input signal to the  $\overline{\text{INTi}}$  pin requires the pulse width specified in the electrical characteristics (i = 0 to 5). If the pulse width is narrower than the specification, an external interrupt may not be accepted.
- When the effective level or edge of the  $\overline{\text{INTi}}$  pin (i = 0 to 5) is changed by the following bits: bits POL, LVS in the INTiIC register, the IFSR0i bit (i = 0 to 5) in the IFSR0 register, the corresponding IR bit may become 1 (interrupt requested). When setting the above mentioned bits, preset bits ILVL2 to ILVL0 in the INTiIC register to 000b (interrupt disabled). After setting the above mentioned bits, set the corresponding IR bit to 0 (no interrupt requested), then rewrite bits ILVL2 to ILVL0.

## 29.5 Notes on DMAC

### 29.5.1 DMAC-associated Register Settings

- Set DMAC-associated registers while bits MDi1 and MDi0 in the DMDi register are 00b (DMA transfer disabled) (i = 0 to 3). Then, set bits MDi1 and MDi0 to 01b (single transfer) or 11b (repeat transfer) at the end of the setup procedure. This procedure also applies when rewriting bits UDAi, USAi, and BWi1 and BWi0 in the DMDi register.
- When rewriting the DMAC-associated registers while DMA transfer is enabled, stop the peripherals that can be DMA triggers so that no DMA transfer request is generated, then set bits MDi1 and MDi0 in the DMDi register of the corresponding channel to 00b (DMA transfer disabled).
- Once a DMA transfer request is accepted, DMA transfer cannot be disabled even if setting bits MDi1 and MDi0 in the DMDi register to 00b (DMA transfer disabled). Do not change the settings of any DMAC-associated registers other than bits MDi1 and MDi0 until the DMA transfer is completed.
- After setting registers DMiSL and DMiSL2, wait at least six peripheral bus clocks to set bits MDi1 and MDi0 in the DMDi register to 01b (single transfer) or 11b (repeat transfer).

### 29.5.2 Reading DMAC-associated Registers

- Use the following read order to sequentially read registers DMiSL and DMiSL2:  
DM0SL, DM1SL, DM2SL, and DM3SL  
DM0SL2, DM1SL2, DM2SL2, and DM3SL2

## 29.6 Notes on Timers

### 29.6.1 Timer A and Timer B

All timers are stopped after a reset. To restart timers, configure parameters such as operating mode, count source, and counter value, then set the TAI<sub>S</sub> bit or TB<sub>J</sub>S bit in the TABSR or TBSR register to 1 (count starts) (i = 0 to 4; j = 0 to 5).

The following registers and bits should be set while the TAI<sub>S</sub> bit or TB<sub>J</sub>S bit is 0 (count stops):

- Registers TAI<sub>M</sub>R and TB<sub>J</sub>M<sub>R</sub>
- UDF register
- Bits TAZIE, TA0TGL, and TA0TGH in the ONSF register
- TRGSR register

### 29.6.2 Timer A

#### 29.6.2.1 Timer Mode

- While the timer counter is running, the TAI register indicates a counter value at any given time. However, FFFFh is read while reloading is in progress. A set value is read if the TAI register is set while the timer counter is stopped.

#### 29.6.2.2 Event Counter Mode

- While the timer counter is running, the TAI register indicates a counter value at any given time. However, FFFFh is read if the timer counter underflows or 0000h if overflows while reloading is in progress. A set value is read if the TAI register is set while the timer counter is stopped.

#### 29.6.2.3 One-shot Timer Mode

- If the TAI<sub>S</sub> bit in the TABSR register is set to 0 (count stops) while the timer counter is running, the following operations are performed:
  - The timer counter stops and the setting value of the TAI register is reloaded.
  - A low signal is output at the TAI<sub>OUT</sub> pin.
  - The IR bit in the TAI<sub>IC</sub> register becomes 1 (interrupts requested) after one CPU clock cycle.
- The one-shot timer is operated by an internal count source. When the trigger is an input to the TAI<sub>IN</sub> pin, the signal is output with a maximum one count source clock delay after a trigger input to the TAI<sub>IN</sub> pin.
- The IR bit becomes 1 by any of the settings below. To use the timer Ai interrupt, set the IR bit to 0 after one of the settings below is done:
  - Select one-shot timer mode after a reset.
  - Switch operating modes from timer mode to one-shot timer mode.
  - Switch operating modes from event counter mode to one-shot timer mode.
- If a retrigger occurs while counting, the timer counter decrements by one, reloads the setting value of the TAI register, and then continues counting. To generate a retrigger while counting, wait one or more count source cycles after the last trigger is generated.
- When an external trigger input is selected to start counting in timer A one-shot mode, do not provide an external retrigger for 300 ns before the timer counter reaches 0000h. Otherwise, it may stop counting.



#### 29.6.2.4 Pulse-width Modulation Mode

- The IR bit becomes 1 by any of the settings below. To use the timer Ai interrupt, set the IR bit to 0 after one of the settings below is done (i = 0 to 4):
  - Select pulse-width modulation mode after a reset.
  - Switch operating modes from timer mode to pulse-width modulation mode.
  - Switch operating modes from event counter mode to pulse-width modulation mode.
  
- If the TAI<sub>S</sub> bit in the TABSR register is set to 0 (count stops) while PWM pulse is output, the following operations are performed:
  - The timer counter stops.
  - The output level at the TAI<sub>OUT</sub> pin changes from high to low. The IR bit becomes 1.
  - When a low signal is output at the TAI<sub>OUT</sub> pin, it does not change. The IR bit does not change, either.

## 29.6.3 Timer B

### 29.6.3.1 Timer Mode and Event Counter Mode

- While the timer counter is running, the TBj register indicates a counter value at any given time (j = 0 to 5). However, FFFFh is read while reloading is in progress. When a value is set to the TBj register while the timer counter is stopped, if the TBj register is read before the count starts, the set value is read.

### 29.6.3.2 Pulse Period/Pulse-width Measure Mode

- While the TBjS bit in the TABSR or TBSR register is 1 (start counter), after the MR3 bit becomes 1 (overflow) and at least one count source cycle has elapsed, a write operation to the TBjMR register sets the MR3 bit to 0 (no overflow).
- Use the IR bit in the TBjIC register to detect overflow. The MR3 bit is used only to determine an interrupt request source within the interrupt handler.
- The counter value is undefined when the timer counter starts. Therefore, the timer counter may overflow before a measured pulse is applied on the initial valid edge and cause a timer Bj interrupt request to be generated.
- When the measured pulse is applied on the initial valid edge after the timer counter starts, an undefined value is transferred to the reload register. At this time, a timer Bj interrupt request is not generated.
- The IR bit may become 1 (interrupt requested) by changing bits MR1 and MR0 in the TBjMR register after the timer counter starts. However, if the same value is rewritten to bits MR1 and MR0, the IR bit does not change.
- Pulse width is continuously measured in pulse-width measure mode. Whether the measurement result is high-level width or not is determined by a program.
- When an overflow occurs at the same time a pulse is applied on the valid edge, this pulse is not recognized since an interrupt request is generated only once. Do not let an overflow occur in pulse period measure mode.
- In pulse-width measure mode, determine whether an interrupt source is a pulse applied on the valid edge or an overflow by reading the port level in the timer Bj interrupt handler.

## 29.7 Notes on Three-phase Motor Control Timers

### 29.7.1 Shutdown

- When a low signal is applied to the  $\overline{\text{NMI}}$  pin with the following bit settings, pins TA1OUT, TA2OUT, and TA4OUT become high-impedance: the PM24 bit in the PM2 register is 1 (NMI enabled), the SDE bit in the IOBC register is 1 (shutdown enabled), the INV02 bit in the INVC0 register is 1 (three-phase motor control timers used), and the INV03 bit is 1 (three-phase motor control timer output enabled).

### 29.7.2 Register Setting

- Do not write to the TAI1 register before and after timer B2 underflows ( $i = 1, 2, 4$ ). Before writing to the TAI1 register, read the TB2 register to verify that sufficient time remains until timer B2 underflows. Then, immediately write to the TAI1 register so no interrupt handling is performed during this write procedure. If the TB2 register indicates little time remains until the underflow, write to the TAI1 register after timer B2 underflows.

## 29.8 Notes on Serial Interface

### 29.8.1 Changing the UiBRG Register (i = 0 to 4)

- Set the UiBRG register after setting bits CLK1 and CLK0 in the UiC0 register. When these bits are changed, the UiBRG register must be set again.
- When a clock is input immediately after the UiBRG register is set to 00h, the counter may become FFh. In this case, it requires extra 256 clocks to reload 00h to the register. Once 00h is reloaded, the counter performs the operation without dividing the count source according to the setting.

### 29.8.2 Pin Output

When the NCH bit in the U2C0 register is set to 1, pins assigned for TXD2/SDA2 and SCL2 function as N-channel open drain output even if the UART output function is not selected in output function select register.

When the NODC bit in the U2SMR3 register is set to 1, CLK2 pins function as N-channel open drain output even if the UART output function is not selected in output function select register.

### 29.8.3 Synchronous Serial Interface Mode

#### 29.8.3.1 Selecting an External Clock

- If an external clock is selected, the following conditions must be met while the external clock is held high when the CKPOL bit in the UiC0 register is 0 (transmit data output on the falling edge of the transmit/receive clock and receive data input on the rising edge), or while the external clock is held low when the CKPOL bit is 1 (transmit data output on the rising edge of the transmit/receive clock and receive data input on the falling edge) (i = 0 to 4):
  - The TE bit in the UiC1 register is 1 (transmission enabled).
  - The RE bit in the UiC1 register is 1 (reception enabled). This bit setting is not required when only transmitting.
  - The TI bit in the UiC1 register is 0 (data held in the UiTB register).

#### 29.8.3.2 Receive Operation

- In synchronous serial interface mode, the transmit/receive clock is controlled by the transmit control circuit. Set UARTi-associated registers for a transmit operation, even if the MCU is used only for receive operation (i = 0 to 4). Dummy data is output from the TXDi pin while receiving when the TXDi pin is set to output mode.
- When data is received continuously, an overrun error occurs when the RI bit in the UiC1 register is 1 (data held in the UiRB register) and the seventh bit of the next data is received in the UARTi receive shift register. Then, the OER bit in the UiRB register becomes 1 (overrun error occurred). In this case, the UiRB register becomes undefined. If an overrun error occurs, the IR bit in the SiRIC register does not change to 1.

### 29.8.4 Special Mode 1 (I<sup>2</sup>C Mode)

- To generate a start condition, stop condition, or restart condition, set the STSPSEL bit in the UiSMR4 register to 0 (i = 0 to 2). Then, wait a half or more clock cycles of the transmit/receive clock to change the condition generate bits (STAREQ, RSTAREQ, or STPREQ bit) from 0 to 1.

### 29.8.5 Reset Procedure on Communication Error

Operations which result in communication errors such as rewriting function select registers during transmission/reception should not be performed. Follow the procedure below to reset the internal circuit once the communication error occurs in the following cases: when the operation above is performed by a receiver or transmitter or when a bit slip is caused by noise.

#### A. Synchronous Serial Interface Mode

- (1) Set the TE bit in the UiC1 register to 0 (transmission disabled) and the RE bit to 0 (reception disabled) (i = 0 to 4).
- (2) Set bits SMD2 to SMD0 in the UiMR register to 000b (serial interface disabled).
- (3) Set bits SMD2 to SMD0 in the UiMR register to 001b (synchronous serial interface mode).
- (4) Set the TE bit in the UiC1 register to 1 (transmission enabled) and the RE bit to 1 (reception enabled) if necessary.

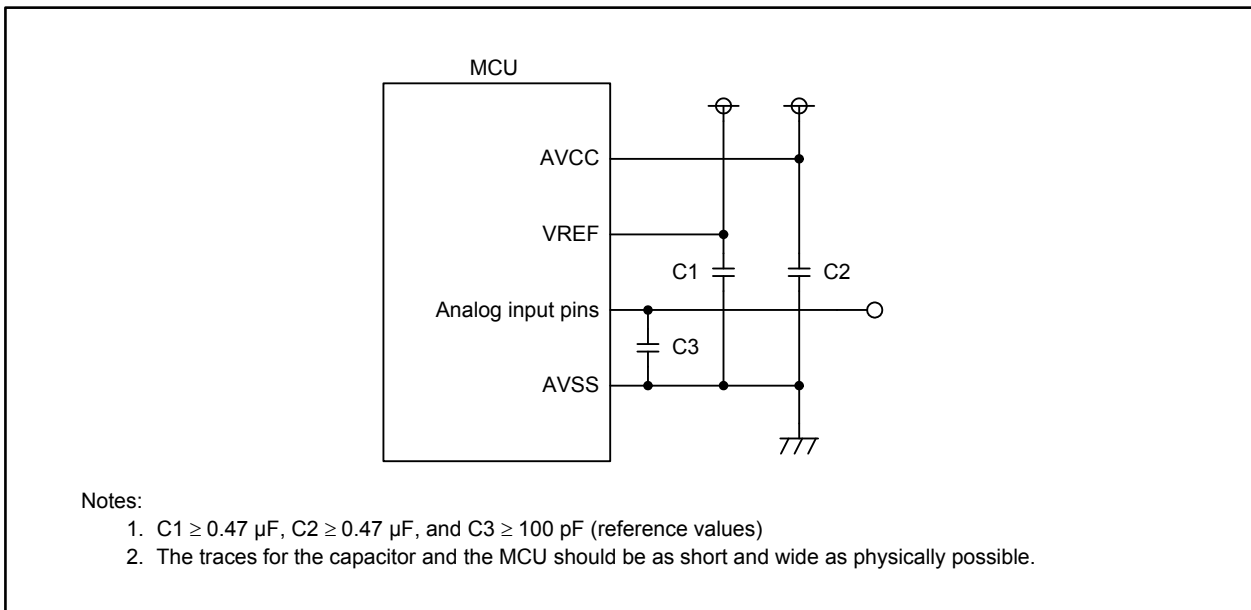
#### B. UART Mode

- (1) Set the TE bit in the UiC1 register to 0 (transmission disabled) and the RE bit to 0 (reception disabled).
- (2) Set bits SMD2 to SMD0 in the UiMR register to 000b (serial interface disabled).
- (3) Set bits SMD2 to SMD0 in the UiMR register to 100b (UART mode, 7-bit character length), 101b (UART mode, 8-bit character length), or 110b (UART mode, 9-bit character length).
- (4) Set the TE bit in the UiC1 register to 1 (transmission enabled) and the RE bit to 1 (reception enabled) if necessary.

## 29.9 Notes on A/D Converter

### 29.9.1 Notes on Designing Boards

- Three capacitors should be placed between the AVSS pin and pins such as AVCC, VREF, and analog inputs (AN\_0 to AN\_4, AN0\_0 to AN0\_7, and AN2\_0 to AN2\_7) to avoid erroneous operations caused by noise or latchup, and to reduce conversion errors. Figure 29.1 shows an example of pin configuration for A/D converter.



**Figure 29.1 Pin Configuration for the A/D Converter**

- When  $AVCC = VREF = VCC$ , A/D input voltage for pins AN\_0 to AN\_4, AN0\_0 to AN0\_7, AN2\_0 to AN2\_7, ANEX0, and ANEX1 should be VCC or lower.

## 29.9.2 Notes on Programming

- The following registers should be written while A/D conversion is stopped. That is, before a trigger occurs: AD0CON0 (except the ADST bit), AD0CON1, AD0CON2, AD0CON3, and AD0CON5.
- When the VCUT bit in the AD0CON1 register is changed from 0 (VREF connected) to 1 (VREF disconnected), wait for at least 1  $\mu$ s before starting A/D conversion. When not performing A/D conversion, set the VCUT bit to 0 to reduce power consumption.
- Set the port direction bit for the pin to be used as an analog input pin to 0 (input). Set the ASEL bit of the corresponding port function select register to 1 (port is used as A/D input).
- When the TRG bit in the AD0CON0 register is 1 (external trigger or hardware trigger), set the corresponding port direction bit (PD9\_7 bit) for the  $\overline{\text{ADTRG}}$  pin to 0 (input).
- The  $\phi_{\text{AD}}$  frequency should be 16 MHz or lower when VCC is 4.2 to 5.5 V, and 10 MHz or lower when VCC is 3.0 to 4.2 V. It should be 1 MHz or higher when the sample and hold function is enabled. If not, it should be 250 kHz or higher.
- When A/D operating mode (bits MD1 and MD0 in the AD0CON0 register or the MD2 bit in the AD0CON1 register) has been changed, reselect analog input pins by setting bits CH2 to CH0 in the AD0CON0 register or bits SCAN1 and SCAN0 in the AD0CON1 register.
- If the AD0i register is read when the A/D converted result is stored to the register, the stored value may have an error ( $i = 0$  to 7). Read the AD0i register after A/D conversion is completed. In one-shot mode or single sweep mode, read the AD0i register after the IR bit in the AD0IC register becomes 1 (interrupt requested). In repeat mode, repeat sweep mode 0, or repeat sweep mode 1, an interrupt request can be generated each time A/D conversion is completed when the DUS bit in the AD0CON3 register is 1 (DMAC operating mode enabled). Similar to the other modes above, read the AD00 register after the IR bit in the AD0IC register becomes 1 (interrupt requested).
- When an A/D conversion is halted by setting the ADST bit in the AD0CON0 register to 0, the converted result is undefined. In addition, the unconverted AD0i register may also become undefined. Consequently, the AD0i register should not be used just after A/D conversion is halted.
- External triggers cannot be used in DMAC operating mode. When the DMAC is configured to transfer converted results, do not read the AD00 register by a program.
- While in single sweep mode, if A/D conversion is halted by setting the ADST bit in the AD0CON0 register to 0 (A/D conversion is stopped), an interrupt request may be generated even though the sweep is not completed. To halt A/D conversion, disable interrupts before setting the ADST bit to 0.

## 29.10 Note on Serial Bus Interface

### 29.10.1 Note on Synchronous Serial Communication Mode and 4-wire Serial Bus Mode

#### 29.10.1.1 Accessing SBI Associated Registers

To read the SBI associated registers (44F06h to 44F0Fh), insert at least three instructions after writing to the registers.

- An example of inserting at least three instructions

```
Program example    MOV.B  #00h, 44F0Bh ; Set the SS0ER register to 00h.
                   NOP
                   NOP
                   NOP
                   MOV.B  44F0Bh, R0L
```



## 29.11 Notes on Flash Memory Rewriting

### 29.11.1 Note on Power Supply

- Keep the supply voltage constant within the range specified in the electrical characteristics while a rewrite operation on the flash memory is in progress. If the supply voltage goes beyond the guaranteed value, the device cannot be guaranteed.

### 29.11.2 Note on Hardware Reset

- Do not perform a hardware reset while a rewrite operation on the flash memory is in progress.

### 29.11.3 Note on Flash Memory Protection

- If an ID code written in an assigned address has an error, any read/write operation on the flash memory in standard serial I/O mode is disabled.

### 29.11.4 Notes on Programming

- Do not set the FEW bit in the FMCR register to 1 (CPU rewrite mode) in low speed mode or low power mode.
- When the EWM bit in the E2FM register is 1 (program or erase enabled), do not set the FEW bit in the FMCR register to 1.
- The program, block erase, lock bit program, and protect bit program are interrupted by an NMI, a watchdog timer interrupt, an oscillator stop detection interrupt, or a low voltage detection interrupt. If any of the software commands above are interrupted, erase the corresponding block and then execute the same command again. If the block erase command is interrupted, the lock bit and protect bit values become undefined. Therefore, disable the lock bit, and then execute the block erase command again.

### 29.11.5 Notes on Interrupts

- EW0 mode
  - To use interrupts assigned to the relocatable vector table, the vector table should be addressed in RAM space.
  - When an NMI, watchdog timer interrupt, oscillator stop detection interrupt, or low voltage detection interrupt occurs, the flash memory module automatically enters read array mode. Therefore, these interrupts are enabled even during a rewrite operation. However, the rewrite operation in progress is aborted by the interrupts and registers FMR0 and FRSR0 are reset. When the interrupt handler has ended, set the LBD bit in the FMR1 register to 1 (lock bit protection disabled) to re-execute the rewrite operation.
  - Instructions BRK, INTO, and UND, which refer to data on the flash memory, cannot be used in this mode.
- EW1 mode
  - Interrupts assigned to the relocatable vector table should not be accepted during program or block erase operation.
  - The watchdog timer should not be used in count source protect mode. The watchdog timer interrupt should not be generated.
  - When an NMI, watchdog timer interrupt, oscillator stop detection interrupt, or low voltage detection interrupt occurs, the flash memory module automatically enters read array mode. Therefore, these interrupts are enabled even during a rewrite operation. However, the rewrite operation in progress is aborted by the interrupts and registers FMR0 and FRSR0 are reset. When the interrupt handler has ended, set the EWM bit in the FMR0 register to 1 (EW1 mode) and the LBD bit in the FMR1 register to 1 (lock bit protection disabled) to re-execute the rewrite operation.

### 29.11.6 Notes on Rewrite Control Program

- EW0 mode
  - If the supply voltage drops during the rewrite operation of blocks having the rewrite control program, the rewrite control program may not be successfully rewritten, and the rewrite operation itself may not be performed. In this case, perform the rewrite operation by serial programmer or parallel programmer.
- EW1 mode
  - Do not rewrite blocks having the rewrite control program.

### 29.11.7 Notes on Number of Program/Erase Cycles and Software Command Execution Time

- The time to execute software commands (program, block erase, lock bit program, and protect bit program) increases as the number of program/erase cycles increases. If the number of program/erase cycles exceeds the endurance value specified in the electrical characteristics, it may take an unpredictable amount of time to execute the software commands. The wait time for executing software commands should be set much longer than the execution time specified in the electrical characteristics.

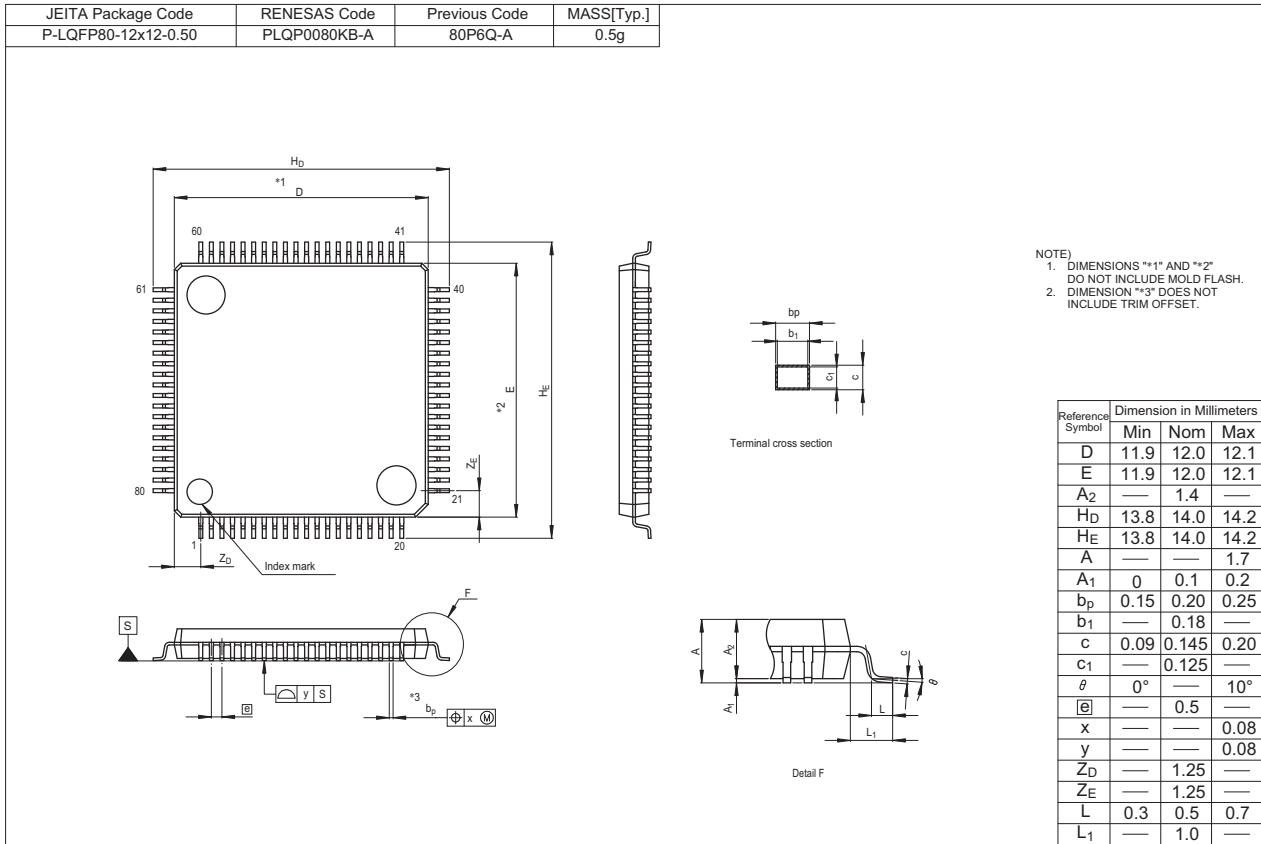
### 29.11.8 Other Notes

- The minimum values of program/erase cycles specified in the electrical characteristics are the maximum values that can guarantee the initial performance of the flash memory. The program/erase operation may still be performed even if the number of program/erase cycles exceeds the guaranteed values.
- Chips repeatedly programmed and erased for debugging should not be used for commercial products.

**29.12 Note on E<sup>2</sup>dataFlash**

When the FEW bit in the FMCR register is 1 (CPU rewrite mode), do not set the EWM bit in the E2FM register to 1 (program or erase enabled).

# Appendix 1. Package Dimensions



## INDEX

### Numerics

4-wire Serial Bus Mode ..... 363

### A

A0 ..... 13  
 A1 ..... 13  
 A2 ..... 13  
 A3 ..... 13  
 AD00 to AD07 ..... 282  
 AD0CON0 ..... 278  
 AD0CON1 ..... 279  
 AD0CON2 ..... 280  
 AD0CON3 ..... 281  
 AD0CON5 ..... 282  
 AD0IC ..... 128  
 Address Register ..... 13

### B

B Flag ..... 14  
 BCN0IC to BCN2IC ..... 128  
 BRK Instruction Interrupt ..... 119  
 BRK2 Instruction Interrupt ..... 119

### C

C Flag ..... 13  
 C0AFSR, C1AFSR ..... 442  
 C0BCR, C1BCR ..... 414  
 C0CLKR, C1CLKR ..... 413  
 C0CSSR, C1CSSR ..... 441  
 C0CTLR, C1CTLR ..... 409  
 C0ECSR, C1ECSR ..... 450  
 C0EIC, C1EIC ..... 128  
 C0EIER, C1EIER ..... 443  
 C0EIFR, C1EIFR ..... 445  
 C0FIDCR0, C0FIDCR1 ..... 417  
 C0FRIC, C1FRIC ..... 128  
 C0FTIC, C1FTIC ..... 128  
 C0MB0 to C0MB31 ..... 421  
 C0MCTL0 to C0MCTL31 ..... 425  
 C0MIER, C1MIER ..... 424  
 C0MKIVLR, C1MKIVLR ..... 419  
 C0MKR0 to C0MKR7 ..... 416  
 C0MSMR, C1MSMR ..... 438  
 C0MSSR, C1MSSR ..... 439  
 C0RECR, C1RECR ..... 448  
 C0RFCR, C1RFCR ..... 428

C0RFPCR, C1RFPCR ..... 431  
 C0RIC, C1RIC ..... 128  
 C0STR, C1STR ..... 435  
 C0TCR, C1TCR ..... 453  
 C0TECR, C1TECR ..... 449  
 C0TFCR, C1TFCR ..... 432  
 C0TFPCR, C1TFPCR ..... 434  
 C0TIC, C1TIC ..... 128  
 C0TSR, C1TSR ..... 452  
 C0WIC, C1WIC ..... 128  
 C1FIDCR0, C1FIDCR1 ..... 417  
 C1MB0 to C1MB31 ..... 421  
 C1MCTL0 to C1MCTL31 ..... 425  
 C1MKR0 to C1MKR7 ..... 416  
 Carry Flag ..... 13  
 CCR ..... 83  
 CM0 ..... 84  
 CM1 ..... 85  
 CM2 ..... 85  
 CM3 ..... 86  
 CPSRF ..... 87  
 CRCD ..... 295  
 CRCIN ..... 296

### D

D Flag ..... 13  
 Data Register ..... 13  
 DCR0 to DCR3 ..... 15, 154  
 DCT0 to DCT3 ..... 15, 153  
 DDA0 to DDA3 ..... 15, 155  
 DDR0 to DDR3 ..... 15, 155  
 Debug Flag ..... 13  
 DM0IC to DM3IC ..... 128  
 DM0SL to DM3SL ..... 149  
 DM0SL2 to DM3SL2 ..... 150  
 DMA Destination Address Register ..... 15  
 DMA Destination Address Reload Register ..... 15  
 DMA Mode Register ..... 15  
 DMA Source Address Register ..... 15  
 DMA Source Address Reload Register ..... 15  
 DMA Terminal Count Register ..... 15  
 DMA Terminal Count Reload Register ..... 15  
 DMD0 to DMD3 ..... 15, 153  
 DP Bit ..... 14  
 DSA0 to DSA3 ..... 15, 154  
 DSR0 to DSR3 ..... 15, 155  
 DTT ..... 222  
 DVCR ..... 78

### E

E2FA ..... 527

E2FC .....	526
E2FD .....	528
E2FI .....	528
E2FIC .....	128
E2FM .....	526
E2FS0 .....	527
E2FS1 .....	527

**F**

Fast Interrupt .....	121
FB .....	13
FBPM0 .....	505
FBPM1 .....	505
FEBC .....	502
Fixed-point Designation Bit .....	14
Flag Register .....	13
FLG .....	13
Floating-point Overflow Flag .....	14
Floating-point Rounding Mode .....	14
Floating-point Underflow Flag .....	14
FMCR .....	501
FMR0 .....	503
FMR1 .....	504
FMSR0 .....	504
FO Flag .....	14
FPR0 .....	503
Frame Base Register .....	13
FU Flag .....	14

**G**

G0BCR0 .....	305
G0BCR1 .....	306
G0BT .....	304
G0FE .....	313
G0FS .....	312
G0PO0 to G0PO7 .....	310
G0POCR0 to G0POCR7 .....	309
G0PSCR .....	311
G0SDR .....	310
G0TM0 to G0TM7 .....	308
G0TMCR0 to G0TMCR7 .....	307
G0TPR6, G0TPR7 .....	307

**H**

Hardware Interrupt .....	120
--------------------------	-----

**I**

I Flag .....	14
--------------	----

IC07DDR .....	334
ICTB2 .....	213
IDB0, IDB1 .....	212
IFS0 .....	487
IFS1 .....	487
IFS2 .....	488
IFS5 .....	489
IFS6 .....	489
IFSR0 .....	137
IIO0IC to IIO11IC .....	128
IIO0IE to IIO11IE .....	141
IIO0IR to IIO11IR .....	140
INT Instruction Interrupt .....	119
INT0IC to INT5IC .....	129
INTB .....	13
Interrupt Control Register .....	128
Interrupt Enable Flag .....	14
Interrupt request level .....	129
Interrupt Response Time .....	133
Interrupt Sequence .....	132
Interrupt Stack Pointer .....	13
Interrupt Types .....	118
Interrupt Vector Table Base Register .....	13
INTF0 .....	138
INTF1 .....	138
INVC0 .....	210
INVC1 .....	211
IOBC .....	212
IPL .....	14, 127
ISP .....	13

**L**

L0IC .....	128
LBRG .....	375
LBRK .....	380
LBRP0 .....	376
LBRP1 .....	376
LCW .....	375
LDB1 to LDB8 .....	386
LEST .....	386
LIDB .....	383
LLDIC .....	128
LMD0 .....	377
LMD1 .....	378
Low Voltage Detection Interrupt .....	79, 120
Low Voltage Detector .....	76
LRFC .....	382
LSC .....	383
LSPC .....	381
LST .....	385
LTC .....	384
LVDC .....	77

LWUP ..... 379

## M

Maskable Interrupt ..... 118

MOD ..... 166

## N

NMI (Non Maskable Interrupt) ..... 120

Non-maskable Interrupt ..... 118

## O

O Flag ..... 14

OFS ..... 146

ONSF ..... 180

Oscillator Stop Detection Interrupt ..... 120

Overflow Flag ..... 14

Overflow Interrupt ..... 119

## P

P0 to P9 ..... 173

P0\_0S to P0\_7S ..... 477

P1\_0S to P1\_7S ..... 478

P2\_0S to P2\_7S ..... 479

P3\_0S to P3\_7S ..... 480

P4\_0S to P4\_7S ..... 481

P5\_0S to P5\_7S ..... 482

P6\_5S, P6\_7S ..... 483

P7\_0S, P7\_4S to P7\_6S ..... 484

P8\_2S to P8\_4S, P8\_6S, P8\_7S ..... 485

P9\_3S to P9\_7S ..... 486

PBC ..... 113

PC ..... 13

PCR ..... 492

PD0 to PD9 ..... 475

Peripheral Interrupt ..... 120

PLC0 ..... 92

PLC1 ..... 93

PLS ..... 93

PM0 ..... 72

PM2 ..... 88

PM3 ..... 89

PRCR ..... 114

PRCR2 ..... 115

PRCR3 ..... 115

PRCR4 ..... 116

Processor Interrupt Priority Level ..... 14, 127

Program Counter ..... 13

PRR ..... 117

PUR0 ..... 490

PUR1 ..... 490

PUR2 ..... 491

## R

R2R0 ..... 13

R3R1 ..... 13

R6R4 ..... 13

R7R5 ..... 13

Register Bank Select Flag ..... 14

Register Saving ..... 134

RIPL1, RIPL2 ..... 131, 163

RND ..... 14

## S

S Flag ..... 13

S0RIC to S4RIC ..... 128

S0TIC to S4TIC ..... 128

Save Flag Register ..... 15

Save PC Register ..... 15

SB ..... 13

Serial Bus Interface ..... 336

Sign Flag ..... 13

Single-step Interrupt ..... 120

Software Interrupt ..... 119

SP ..... 13

Special Interrupt ..... 120

SS0CRH ..... 341

SS0CRL ..... 342

SS0ER ..... 345

SS0IC ..... 128

SS0MR ..... 343, 344

SS0MR2 ..... 347

SS0RDR ..... 348

SS0SR ..... 346

SS0TDR ..... 348

Stack Pointer ..... 13

Stack Pointer Select Flag ..... 14

Static Base Register ..... 13

SVF ..... 15

SVP ..... 15

Synchronous Serial Communication Mode 356

## T

TA0 to TA4 ..... 177

TA0IC to TA4IC ..... 128

TA0MR to TA4MR .. 178, 184, 187, 190, 192

TA1, TA2, TA4, TA11, TA21, TA41 ..... 217

TA1M, TA11M, TA2M, TA21M, TA4M, TA41M

..... 218

TA1MR, TA2MR, TA4MR .....	219
TABSR .....	178, 196, 220
TB0 to TB5 .....	195
TB0IC to TB5IC .....	128
TB0MR to TB5MR .....	195, 198, 200, 203
TB2 .....	215
TB2MR .....	215
TB2SC .....	216
TBECKS .....	201
TBSR .....	196
TCSPR .....	86, 182
TRGSR .....	181, 220

XYC .....	298
-----------	-----

**Y**

Y0R to Y15R .....	299
-------------------	-----

**Z**

Z Flag .....	13
Zero Flag .....	13

**U**

U Flag .....	14
U0BRG to U4BRG .....	237
U0C0 to U2C0 .....	231
U0C1 to U2C1 .....	233
U0MR to U2MR .....	229
U0RB to U2RB .....	238
U0SMR to U2SMR .....	234
U0SMR2 to U2SMR2 .....	235
U0SMR3 to U2SMR3 .....	236
U0SMR4 to U2SMR4 .....	237
U0TB to U4TB .....	238
U34CON .....	234
U3C0, U4C0 .....	232
U3C1, U4C1 .....	233
U3MR, U4MR .....	230
U3RB, U4RB .....	239
UDF .....	179
Undefined Instruction Interrupt .....	119
User Stack Pointer .....	13
USP .....	13

**V**

VCT .....	15
Vector Register .....	15
VRCR .....	74

**W**

Watchdog Timer Interrupt .....	120
WDC .....	144
WDK .....	145
WDTs .....	145

**X**

X0R to X15R .....	298
-------------------	-----



<b>REVISION HISTORY</b>	<b>R32C/161 Group User's Manual: Hardware</b>
-------------------------	---

Rev.	Date	Description	
		Page	Summary
0.51	Dec 24, 2008	—	Initial release
1.02	Feb 26, 2010	—	Second edition released
		—	<p><b>This manual in general</b></p> <ul style="list-style-type: none"> <li>• Changed the following expressions: “erase/program” to “program/erase” (under Chapters 1, 26, 27, and 29); “ID code check” to “ID code protect” (under Chapters 1 and 26); “sample &amp; hold” to “sample and hold” (under Chapters 1, 18, and 28); “reset operation” to “reset” (under Chapters 7, 8, 10, 11, and 29); and “Start/Stop Condition” to “Start Condition/Stop Condition” (under Chapters 4, 10, and 17)</li> <li>• Added a question mark to the sentence in diamond-shaped boxes of flowcharts (under Chapters 22, 26, and 27)</li> <li>• Modified descriptions for notes in some figures (under Chapters 6, 7, 8, 11, 12, 17, and 26)</li> </ul>
		—	<p><b>About This Manual</b></p> <ul style="list-style-type: none"> <li>• Made minor text modifications to this section</li> <li>• Deleted “Shortsheet” row from <b>1. Purpose and Target User</b></li> <li>• Revised illustration in <b>3. Registers</b></li> </ul>
		3	<p><b>Chapter 1. Overview</b></p> <ul style="list-style-type: none"> <li>• Changed the following expressions in <b>Table 1.2</b>: “Read protection” to “Security protection” and “on-board flash reprogramming” to “on-board flash programming”</li> </ul>
		4	<ul style="list-style-type: none"> <li>• Completed “under development” phase of part number R5F64610JFP in <b>Table 1.3</b></li> </ul>
		—	<p><b>Chapter 2. Central Processing Unit</b></p> <ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> </ul>
		—	<p><b>Chapter 3. Memory</b></p> <ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> </ul>
		17	<p><b>Chapter 4. Special Function Registers</b></p> <ul style="list-style-type: none"> <li>• Changed hexadecimal format of reset values for registers CCR and FMCR in <b>Table 4.1</b> to binary</li> </ul>
		24	<ul style="list-style-type: none"> <li>• Changed reset values “XXXX XXXXb” and “XXXX 000Xb” for registers U3RB and U4RB in <b>Table 4.8</b> to “XXXXh”</li> </ul>
		26	<ul style="list-style-type: none"> <li>• Changed expression of register name “Xi Register Yi Register” and symbol “XiR, YiR” in <b>Table 4.10</b> to “Xi Register/Yi Register” and to “XiR/YiR”, respectively</li> </ul>
		35, 36	<ul style="list-style-type: none"> <li>• Modified register name “Port Pi_j Port Function Select Register” in <b>Table 4.19 and 4.20</b> to “Port Pi_j Function Select Register”</li> </ul>
		38	<ul style="list-style-type: none"> <li>• Modified register name “DMAi Request Source Select Register 1” in <b>Table 4.22</b> to “DMAi Request Source Select Register”</li> </ul>
		39	<ul style="list-style-type: none"> <li>• Modified “X” in reset values for unassigned bits to “0” for the following registers in <b>Table 4.23</b>: LCW, LBRG, LMD0, LBRK, LSPC, LRFC, LSC, LTC, LST, and LEST</li> </ul>
		55, 69	<ul style="list-style-type: none"> <li>• Modified reset value “00h” for C1CLKR in <b>Table 4.39</b> and C0CLKR in <b>Table 4.53</b> to “000X 0000b”</li> </ul>

REVISION HISTORY	R32C/161 Group User's Manual: Hardware
------------------	--

Rev.	Date	Description	
		Page	Summary
			<b>Chapter 5. Resets</b>
		—	<ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> </ul>
		71	<ul style="list-style-type: none"> <li>• Deleted the arrow indicating period from when the <math>\overline{\text{RESET}}</math> signal becomes high to when address data is output from <b>Figure 5.2</b></li> </ul>
		73	<ul style="list-style-type: none"> <li>• Modified <b>Figure 5.5</b></li> </ul>
		76	<b>Chapter 6. Power Management</b> <ul style="list-style-type: none"> <li>• Corrected the following bit symbols “LV DEN” and “LVEIEN” in <b>Figure 6.3</b> to “VDEN” and “LVDIEN”, respectively; Corrected register symbol “LVDCR” to “LVDC”</li> </ul>
			<b>Chapter 7. Clock Generator</b>
		—	<ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> <li>• Modified expression: “peripheral clock” to “peripheral clock source” when it means a peripheral clock source</li> </ul>
		81	<ul style="list-style-type: none"> <li>• Modified reference figure number “Figure 7.3” in <b>7.1</b> to “Figure 7.2”</li> </ul>
		82	<ul style="list-style-type: none"> <li>• Changed the following bit symbols in <b>Figure 7.1</b>: “CM06 to CM00” to “CM00 to CM02, CM04, and CM05”, “CM27 to CM20” to “CM20”, “CM31 and CM30” to “CM30 and CM31”, and “PM27 to PM20” to “PM26”</li> </ul>
		83	<ul style="list-style-type: none"> <li>• Corrected a typo “Aah” in Note 1 for CCR register of <b>Figure 7.2</b> to “AAh”</li> </ul>
		84	<ul style="list-style-type: none"> <li>• Added “and f2n whose clock source is the main clock” to Note 3 for CM0 register in <b>Figure 7.3</b></li> </ul>
		85	<ul style="list-style-type: none"> <li>• Modified description for Note 3 of CM1 register in <b>Figure 7.4</b></li> </ul>
		88	<ul style="list-style-type: none"> <li>• Modified “Bus clock” in “Function” of PM22 bit in <b>Figure 7.9</b> to “Peripheral bus clock”</li> </ul>
		92	<ul style="list-style-type: none"> <li>• Added register name “PLC1” for SEO bit in <b>Figure 7.13</b></li> </ul>
		92, 93	<ul style="list-style-type: none"> <li>• Added description associated with interrupt and DMA transfer to Note 1 for registers PLC0 and PLC1 in <b>Figures 7.14 and 7.15</b></li> </ul>
		94	<ul style="list-style-type: none"> <li>• Changed frequency of PLL frequency synthesizer “50MHz” in line 3 of the third paragraph above <b>Table 7.2</b> to “t<sub>so</sub>(PLL)”; Deleted description associated with frequency from line 14; Modified “t<sub>ocs</sub>(PLL)” to “t<sub>Lock</sub>(PLL)”</li> </ul>
		96	<ul style="list-style-type: none"> <li>• Deleted description “with the frequency of approximately 8MHz” from line 2 of <b>7.3</b>; Corrected a typo “fc” in line 4 to “fC”</li> </ul>
		97	<ul style="list-style-type: none"> <li>• Deleted description associated with frequency from line 2 of <b>7.4</b>;</li> <li>Added “the serial bus interface” to line 6</li> <li>• Added “whose clock source is the peripheral clock source” to line 5 of (1) in <b>7.5</b></li> </ul>
		100-102	<ul style="list-style-type: none"> <li>• Added definitions for the following bit symbols in <b>Figures 7.18 to 7.20</b>: BCS, CM04, CM05, CM10, CM20, CM30, and CM31</li> </ul>
		104, 107	<ul style="list-style-type: none"> <li>• Revised procedures and descriptions for entering wait mode in <b>7.7.2.2</b> and for entering stop mode in <b>7.7.3.1</b></li> </ul>
		105	<ul style="list-style-type: none"> <li>• Changed “the peripheral clock” in line 9 of <b>7.7.2.4</b>, to “f1, f8, f32, f2n (when the clock source is the peripheral clock source), or fAD”; Added description “with the software interrupt numbers 0 to 63” to line 11 to specify peripheral clocks</li> </ul>

<b>REVISION HISTORY</b>	<b>R32C/161 Group User's Manual: Hardware</b>
-------------------------	---

Rev.	Date	Description	
		Page	Summary
		106	<ul style="list-style-type: none"> <li>• Added "Watchdog timer interrupt" and the related specifications to <b>Table 7.5</b></li> </ul>
		108	<ul style="list-style-type: none"> <li>• Moved <b>Table 7.6</b> to under <b>7.7.3.3</b> as <b>Table 7.7</b> and previous <b>Table 7.7</b> becomes <b>Table 7.6</b>; Made related modifications due to the changes</li> </ul>
		110	<ul style="list-style-type: none"> <li>• Deleted entire section of "Wait Mode" from <b>7.9.2.1</b> and the first bullet point for "Stop Mode" from previous <b>7.9.2.2</b> (current <b>7.9.2.1</b>)</li> </ul>
		—	<p><b>Chapter 8. Bus</b></p> <ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> </ul>
		112	<ul style="list-style-type: none"> <li>• Modified bus width "16-bit" in <b>8.2</b> to "16-/32-bit"; Added description for maximum operating frequency to lines 1 and 2</li> </ul>
		—	<p><b>Chapter 9. Protection</b></p> <ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> </ul>
		113	<ul style="list-style-type: none"> <li>• Added IOBC to PRC1 bit as writable register in <b>Figure 9.1</b></li> </ul>
		—	<p><b>Chapter 10. Interrupts</b></p> <ul style="list-style-type: none"> <li>• Made major text modifications to this chapter</li> </ul>
		117	<ul style="list-style-type: none"> <li>• Added "MULX" as instruction to (2) of <b>10.2</b></li> </ul>
		118	<ul style="list-style-type: none"> <li>• Modified number of interrupts "seven" in <b>10.3.1</b> to "five"</li> </ul>
		121-124	<ul style="list-style-type: none"> <li>• Added reference chapter and section numbers to "Reference" in <b>Tables 10.2 to 10.5</b></li> </ul>
		126, 127	<ul style="list-style-type: none"> <li>• Changed IR bit name "Interrupt Request Bit" in <b>Figures 10.3 and 10.4</b> to "Interrupt Request Flag"</li> </ul>
		131	<ul style="list-style-type: none"> <li>• Modified number of cycles for some interrupts in <b>Table 10.7</b></li> </ul>
		133	<ul style="list-style-type: none"> <li>• Changed priority order of hardware interrupts in <b>10.8</b></li> </ul>
		134	<ul style="list-style-type: none"> <li>• Modified "Bits RLVL02 to RLVL00" and "Bits RLVL12 to RLVL10" in <b>Figure 10.8</b> to "Bits RLVL2 to RLVL0 in the RIPL1 register" and "Bits RLVL2 to RLVL0 in the RIPL2 register", respectively</li> </ul>
		137	<ul style="list-style-type: none"> <li>• Moved "(i = 0 to 11)" in <b>Figure 10.12</b> to the title</li> </ul>
		138, 139	<ul style="list-style-type: none"> <li>• Corrected the following register names: "Intelligent I/O Interrupt Request Register" in <b>Figure 10.13</b> to "Intelligent I/O Interrupt Request Register i (i = 0 to 11)" and "Intelligent I/O Interrupt Enable Register" in <b>Figure 10.14</b> to "Intelligent I/O Interrupt Enable Register i (i = 0 to 11)"; Changed variable "j" for definitions of bits to "y" in <b>Figures 10.13 and 10.14</b>; Added expression "channel" to descriptions for TM0yR, PO0yR, TM0yE, and PO0yE</li> </ul>
		140	<ul style="list-style-type: none"> <li>• Corrected typos "PR2 register" in <b>10.13.1</b> and "INTi (i = 0 to 5) pin" in <b>10.13.3</b> to "PM2 register" and "INTi pin (i = 0 to 5)", respectively</li> </ul>
		—	<p><b>Chapter 11. Watchdog Timer</b></p> <ul style="list-style-type: none"> <li>• Revised this chapter entirely</li> </ul>
		142	<ul style="list-style-type: none"> <li>• Modified expression "bus clock" to "peripheral bus clock"</li> </ul>
		143	<ul style="list-style-type: none"> <li>• Changed upper prescaler section in <b>Figure 11.1</b>; Added definitions to the following bit symbols: PM22, WDK2, and WDK3</li> </ul>
		143	<ul style="list-style-type: none"> <li>• Changed bit name "Count Source Protect Mode Prescaler Select Bit" for WDK2 and WDK3 in <b>Figure 11.3</b> to "On-chip Oscillator Prescaler Select Bit"</li> </ul>

REVISION HISTORY	R32C/161 Group User's Manual: Hardware
------------------	--

Rev.	Date	Description	
		Page	Summary
		144	<ul style="list-style-type: none"> <li>• Changed bit name “Watchdog Timer Prescaler Select Bit” for WPSC1 and WPSC0 in <b>Figure 11.5</b> to “Watchdog Timer On-chip Oscillator Prescaler Select Bit”; Added note symbol “(2)” to WDTON bit; Changed “RW” section for each bit to “(1)” as note symbol; Added description “in the PM2 register” to Note 2</li> </ul>
		—	<b>Chapter 12. DMAC</b>
		146	<ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> <li>• Changed the following expressions in <b>Table 12.1</b>: “transfer unit” to “transfer size”, “destination address” to “addressing mode”, “forward” to “incrementing addressing”, and “fixed” to “non-incrementing addressing”</li> </ul>
		147	<ul style="list-style-type: none"> <li>• Modified description “registers DMiSL and DMiSL2” in line 3 of a paragraph above <b>Figure 12.2</b> to “DMiSL register, and in bits DSEL24 to DSEL20 in the DMiSL2 register”</li> </ul>
		148	<ul style="list-style-type: none"> <li>• Modified register name “DMiSL Register 2 (i = 0 to 3) Function” for DSEL20 to DSEL24 in <b>Figure 12.3</b> to “DMiSL2 Register (i = 0 to 3) Function”; Modified “DSEL4 to DSEL0” in Note 1 to “DSEL24 to DSEL20”</li> </ul>
		151	<ul style="list-style-type: none"> <li>• Changed bit names for USAi and UDAi of DMDi register in <b>Figure 12.4</b> and modified descriptions for their functions; Deleted the second sentence of Note 2; Added Note 3; Changed the following expressions of function descriptions: “Not updated” to “Non-incrementing addressing” and “Updated with each transfer” to “Incrementing addressing”</li> </ul>
		154	<ul style="list-style-type: none"> <li>• Modified description for Note 2 of DCTi register in <b>Figure 12.5</b>; Deleted Note 3</li> </ul>
		159	<ul style="list-style-type: none"> <li>• Added “(i = 0 to 3)” to DSAi in line 3 of <b>12.1</b></li> <li>• Modified description “channel i” in line 1 of the first bullet point of <b>12.4.1</b> to “DMDi register”; Deleted whole description of the second and third bullet points; Added two new paragraphs</li> <li>• Modified description “peripheral clocks” in the fourth bullet point of <b>12.4.2</b> to “peripheral bus clocks”</li> </ul>
		—	<b>Chapter 13. DMAC II</b>
		—	<ul style="list-style-type: none"> <li>• Revised this chapter entirely</li> <li>• Changed the following expressions: “transfer data” to “transfer type”, “transfer data unit” to “transfer size”, “transfer space” to “transfer memory space”, “transfer direction” to “addressing mode”, “fixed address” to “non-incrementing/constant addressing”, “forward address” to “incrementing addressing”, “end-of-transfer interrupt” to “DMA II transfer complete interrupt”, “chained transfer address” for CADR to “chained transfer base address”, “transfer source address” for SADR to “source address”, “transfer destination address” for DADR to “destination address”</li> </ul>

REVISION HISTORY	R32C/161 Group User's Manual: Hardware
------------------	--

Rev.	Date	Description	
		Page	Summary
		160	<ul style="list-style-type: none"> <li>• Corrected typos “64-Kbyte space” and “0000000h to 01FFFFFFh” in <b>Table 13.1</b> to “64-Mbyte space” and “00000000h to 01FFFFFFh”, respectively; Added “multiple transfer” and its description to “transfer modes”; Deleted “multiple transfer” from item column</li> <li>• Modified description “The relocatable vector table” in line 1 of <b>13.1</b> to “The relocatable vector”</li> </ul>
		162	<ul style="list-style-type: none"> <li>• Corrected a typo “BASE + 6” in <b>Figure 13.2</b> to “BASE + 8”</li> </ul>
		164	<ul style="list-style-type: none"> <li>• Revised <b>Figure 13.3</b>; Modified function “No register bits” for b14 to b8 to “Reserved”</li> </ul>
		165	<ul style="list-style-type: none"> <li>• Corrected a typo “FE0000000h” in line 3 of <b>13.3</b> to “FE000000h”</li> </ul>
		167	<ul style="list-style-type: none"> <li>• Modified descriptions in <b>Figure 13.4</b></li> </ul>
			<b>Chapter 15. Timers</b>
		—	<ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> </ul>
		172	<ul style="list-style-type: none"> <li>• Corrected the following typos: “TTA0TGL”, “TAiGH”, and “TAiGL” in <b>Figure 15.1</b> to “TA0TGL”, “TAiTGH”, and “TAiTGL”, respectively</li> </ul>
		174	<ul style="list-style-type: none"> <li>• Corrected a typo “TBiS bit” in <b>Figure 15.3</b> to “TAiS”; Added description “Event/trigger selection”</li> </ul>
		189	<ul style="list-style-type: none"> <li>• Corrected a typo “FEh” as a value of m for “8-bit PWM” in <b>Table 15.5</b> to “FFh”</li> </ul>
		190	<ul style="list-style-type: none"> <li>• Modified reset value “0000 000b” for TAI MR register in <b>Figure 15.16</b> to “0000 0000b”</li> </ul>
		192	<ul style="list-style-type: none"> <li>• Changed “TBiS bit” in <b>Figure 15.19</b> to “TBiS”</li> </ul>
		193	<ul style="list-style-type: none"> <li>• Corrected a typo “TM5MR” in the title of <b>Figure 15.21</b> to “TB5MR”</li> </ul>
		201	<ul style="list-style-type: none"> <li>• Moved note symbol “(1)” from “Function” in <b>Figure 15.27</b> to “Bit Name”</li> </ul>
		203, 204	<ul style="list-style-type: none"> <li>• Deleted “(i = 0 to 4)” from line 1 of the first bullet point of <b>15.3.2.3</b> and the second bullet point of <b>15.3.2.4</b></li> </ul>
		204	<ul style="list-style-type: none"> <li>• Added “(i = 0 to 4)” to line 1 of the first bullet point of <b>15.3.2.4</b></li> </ul>
		205	<ul style="list-style-type: none"> <li>• Replaced variable “i” with “j” in <b>15.3.3.1</b> and <b>15.3.3.2</b>; Added “(j = 0 to 5)” to the first bullet point of <b>15.3.3.1</b></li> </ul>
			<b>Chapter 16. Three-phase Motor Control Timers</b>
		—	<ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> <li>• Changed expression “timer B2 counter” to “timer B2”</li> </ul>
		208	<ul style="list-style-type: none"> <li>• Modified “RW” for INV05 bit in <b>Figure 16.2</b> to “RO”</li> </ul>
		210	<ul style="list-style-type: none"> <li>• Modified reset value “0011 111b” for IDBi register in <b>Figure 16.5</b> to “XX11 1111b”; Changed status of bits 7 and 6 from reserved to unassigned</li> </ul>
		213	<ul style="list-style-type: none"> <li>• Modified reset value “0010 0000b” for TB2MR register in <b>Figure 16.8</b> to “00XX 0000b”</li> </ul>
		214, 218	<ul style="list-style-type: none"> <li>• Changed titles of <b>Figures 16.9</b> and <b>16.14</b></li> </ul>
		220	<ul style="list-style-type: none"> <li>• Changed expression “high side- and low side- transistors” in <b>Figure 16.16</b> to “high-side and low-side transistors”</li> </ul>
		221	<ul style="list-style-type: none"> <li>• Corrected a typo “TA4-1” in <b>Figure 16.18</b> to “TA41”; Changed expression “Refer to Figure 16.1” in Note 1 to “Refer to the block diagram of three-phase motor control timers”</li> </ul>
		222	<ul style="list-style-type: none"> <li>• Modified description “The three motor control timer is applicable” in <b>Figure 16.19</b> to “This bit setting is applicable”</li> </ul>

REVISION HISTORY	R32C/161 Group User's Manual: Hardware
------------------	--

Rev.	Date	Description	
		Page	Summary
			<b>Chapter 17. Serial Interface</b>
		—	<ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> <li>• Changed expression “UART transmit/UART receive interrupt” to “transmit/receive interrupt”</li> </ul>
		225, 226	<ul style="list-style-type: none"> <li>• Deleted “CRS” from list of bits in UiC0 register in <b>Figures 17.1 and 17.2</b></li> </ul>
		228	<ul style="list-style-type: none"> <li>• Modified expression of bit symbol “SMD1” (the second one of the two) for UiMR register in <b>Figure 17.4</b> to “SMD2”; Changed expression “i-bit transfer data” to “i-bit character length”</li> </ul>
		230	<ul style="list-style-type: none"> <li>• Corrected a typo “synchronous serialb interface mode” in Note 1 for UiC0 register in <b>Figure 17.6</b> to “synchronous serial interface mode”</li> </ul>
		233, 235	<ul style="list-style-type: none"> <li>• Modified descriptions “SCL” and “SDA” in <b>Figure 17.11 and 17.13</b> to “SCLi” and “SDAi”, respectively</li> </ul>
		234	<ul style="list-style-type: none"> <li>• Modified description “To set the SS” for Note 2 of UiSMR3 in <b>Figure 17.12</b> to “To use the SS function”; Corrected a typo “UiCO register” to “UiC0 register”</li> </ul>
		236	<ul style="list-style-type: none"> <li>• Deleted description “and read as undefined value” from “b15-b9” of UiTB register in <b>Figure 17.15</b></li> <li>• Added addresses “033Fh-033Eh” to UiRB register in <b>Figure 17.16</b></li> </ul>
		238	<ul style="list-style-type: none"> <li>• Modified description “(i = 0 to 2)” for “Transmit/receive clock” in <b>Table 17.2</b> to “(i = 0 to 4)”</li> </ul>
		239	<ul style="list-style-type: none"> <li>• Corrected typos “000h” for bits “7 to 4” and “000h” for bits “2 to 0” for UiSM3 register in <b>Table 17.3</b> to “0000b” and “000b”, respectively</li> </ul>
		243, 252	<ul style="list-style-type: none"> <li>• Deleted description “(i = 0 to 4)” from subsection B in <b>17.1.1 and 17.2.2</b></li> </ul>
		243, 244	<ul style="list-style-type: none"> <li>• Changed variable “j” for descriptions of bits in <b>Figures 17.20 and 17.21</b> to “i”; Deleted “(j = 0 to 6)” from Notes 2 and 4 in <b>Figure 17.20</b> and from Notes 1 and 2 in <b>Figure 17.21</b></li> </ul>
		246	<ul style="list-style-type: none"> <li>• Specified number of stop bits for “Overrun error” in <b>Table 17.5</b>; Changed expression “n stop bit(s)” for “Error detection” to “n stop bit length”</li> </ul>
		248	<ul style="list-style-type: none"> <li>• Corrected a typo “SUM0” for UiRB register in <b>Table 17.7</b> to “SUM”</li> </ul>
		256	<ul style="list-style-type: none"> <li>• Added bits ACKC and ACKD and their definitions to <b>Figure 17.29</b></li> </ul>
		257	<ul style="list-style-type: none"> <li>• Changed bits “7 to 4” for UiC1 register in <b>Table 17.10</b> to “7 to 5” and corrected “0000b” to “000b”; Added UiIRS bit and its function to UiC1 register; Changed description “fix a low output” for bits SWC and SWC9 to “hold a low output”</li> </ul>
		258	<ul style="list-style-type: none"> <li>• Added description “ to the line of “Default output value at the SDAi pin” in <b>Table 17.11</b> to “by output function select registers”</li> </ul>
		268	<ul style="list-style-type: none"> <li>• Corrected a typo “SS pin” in title of <b>Figure 17.35</b> to “SSi pin”</li> </ul>
		271	<ul style="list-style-type: none"> <li>• Deleted whole description from the third bullet point of previous 17.5.2.2 (current <b>17.5.3.2</b>)</li> </ul>
			<b>Chapter 18. A/D Converter</b>
		—	<ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> <li>• Modified numbers of sections and figures due to the deletes throughout the chapter</li> </ul>

<b>REVISION HISTORY</b>	<b>R32C/161 Group User's Manual: Hardware</b>
-------------------------	---

Rev.	Date	Description	
		Page	Summary
		276	<ul style="list-style-type: none"> <li>• Deleted second and third sentences from Note 3 for AD0CON1 register in <b>Figure 18.3</b>; Added Note 4</li> </ul>
		278	<ul style="list-style-type: none"> <li>• Modified description for Note 4 of AD0CON3 register in <b>Figure 18.5</b></li> </ul>
		279	<ul style="list-style-type: none"> <li>• Modified register name "AD0 Control Register" in <b>Figure 18.6</b> to "A/D0 Control Register"; Changed "RO" for unassigned bits b5 and b4 to "RW" and changed function for unassigned bits b7 and b6, and b3</li> </ul>
		280-284	<ul style="list-style-type: none"> <li>• Added bit symbol boxes for AD0i register in <b>Figure 18.7</b>; Changed function for unassigned bits b15 to b10; Modified description for Notes 1 and 4</li> </ul>
		282-284	<ul style="list-style-type: none"> <li>• Revised description for <b>Tables 18.2 to 18.6</b>; Corrected a typo "A/D00" for "Reading of A/D conversion result" to "AD00"</li> </ul>
		285	<ul style="list-style-type: none"> <li>• Added Note 1 to <b>Tables 18.4 to 18.6</b></li> </ul>
		286	<ul style="list-style-type: none"> <li>• Changed expression "AD0j" in <b>18.2.1</b> to "AD0i"; Modified description "TRG0 bit in the AD0CON2 register" in <b>18.2.3</b> to "bits TRG1 and TRG0 in the AD0CON2 register"</li> </ul>
		291	<ul style="list-style-type: none"> <li>• Added description of reference table to line 2 of <b>18.2.5</b></li> </ul>
		—	<p><b>Chapter 19. CRC Calculator</b></p> <ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> </ul>
		—	<p><b>Chapter 20. X-Y Conversion</b></p> <ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> </ul>
		—	<p><b>Chapter 21. Intelligent I/O</b></p> <ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> </ul>
		300	<ul style="list-style-type: none"> <li>• Moved "(j = 0 to 7)" in <b>Figure 21.1</b> to the title; Added definition to BTOR bit</li> </ul>
		317	<ul style="list-style-type: none"> <li>• Moved "(j = 0 to 7)" for "Trigger input polarity" in <b>Table 21.4</b> to the title; Changed expression "IIO0_j pin function" to "IIO0_j input pin function"</li> </ul>
		317, 318	<ul style="list-style-type: none"> <li>• Moved "j = 0 to 7; k = 6, 7" below <b>Table 21.5</b> to the title</li> </ul>
		319	<ul style="list-style-type: none"> <li>• Moved "j = 0 to 7" in <b>Figures 21.18 and 21.19</b> to the titles</li> </ul>
		320	<ul style="list-style-type: none"> <li>• Moved "j = 6, 7" in <b>Figure 21.20</b> to the title</li> </ul>
		321	<ul style="list-style-type: none"> <li>• Moved "j = 0 to 7" in <b>Table 21.6</b> to the title</li> </ul>
		321	<ul style="list-style-type: none"> <li>• Changed expression "IIO0_j pin function" in <b>Table 21.7</b> to "IIO0_j output pin function"</li> </ul>
		323, 325	<ul style="list-style-type: none"> <li>• Changed expression "IIO0_j pin function (output)" in <b>Tables 21.8 and 21.9</b> to "IIO0_j output pin function"</li> </ul>
		330	<ul style="list-style-type: none"> <li>• Deleted "j = 0 to 7" from Case 2 in <b>Figure 21.25</b></li> </ul>
		332	<ul style="list-style-type: none"> <li>• Modified expressions "read out" and "low pulse long" in <b>Figure 21.27</b> to "read" and "low pulse width", respectively</li> </ul>
		—	<p><b>Chapter 22. Serial Bus Interface</b></p> <ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> </ul>
		339	<ul style="list-style-type: none"> <li>• Corrected a typo "SS0L" in <b>Figure 22.4</b> to "SS0"; Modified description "Output pin level remains changed" for "Function" of SOLP bit to "Output pin level remains unchanged"</li> </ul>

<b>REVISION HISTORY</b>	<b>R32C/161 Group User's Manual: Hardware</b>
-------------------------	---

Rev.	Date	Description	
		Page	Summary
		343	<ul style="list-style-type: none"> <li>• Modified expression “undefined value” for b1 in <b>Figure 22.8</b> to “0”;</li> <li>Moved note symbol “(3)” for ORE bit from function description to bit name; Switched function of “0” and “1” for TDRE bit</li> </ul>
		350	<ul style="list-style-type: none"> <li>• Modified figure number to refer “Figure 22.15” in last line of <b>22.1.3</b> to “Figure 22.14”</li> </ul>
		351	<ul style="list-style-type: none"> <li>• Deleted “Abbreviation” column from <b>Table 22.3</b> and added the abbreviations to each interrupt request source; Added note symbols “(1)” and “(2)” to “Generating Condition for Interrupt Requests”, and “CE” bit to Note 2</li> <li>• Modified table number to refer “Table 22.4” in line 1 of the second paragraph in <b>22.1.4</b> to “Table 22.3”</li> </ul>
		352	<ul style="list-style-type: none"> <li>• Corrected a typo “SBIMS” in <b>Table 22.4</b> to “SSUMS”</li> </ul>
		353	<ul style="list-style-type: none"> <li>• Modified figure number to refer “Figure 22.16” in line 1 of <b>22.1.6.1</b> to “Figure 22.15”</li> </ul>
		359	<ul style="list-style-type: none"> <li>• Added “RE” bit to (5) in <b>Figure 22.20</b></li> </ul>
		366	<ul style="list-style-type: none"> <li>• Modified section number for “<math>\overline{\text{SCSi}}</math> Pin Control and Arbitration” <b>22.1.8</b> to <b>22.1.7.4</b></li> </ul>
		367	<ul style="list-style-type: none"> <li>• Modified addresses of program example in <b>22.2.1.1</b></li> </ul>
		—	<p><b>Chapter 23. LIN Module</b></p> <ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> <li>• Modified read values for unassigned bits from “undefined value” to “0”, and “X” in reset values for unassigned bits to “0”</li> </ul>
		372	<ul style="list-style-type: none"> <li>• Modified bit symbol “FLSRC” for LBRG register in <b>Figure 23.4</b> to “FLNS”</li> </ul>
		379	<ul style="list-style-type: none"> <li>• Modified register name “LIN Status Control Command Register” in <b>Figure 23.14</b> to “LIN Status Control Register”</li> </ul>
		382	<ul style="list-style-type: none"> <li>• Corrected a typo: “LDB0” in the title of <b>Figure 23.18</b> to “LDB1”</li> </ul>
		386, 387	<ul style="list-style-type: none"> <li>• Corrected the following typos in (4) and (6) of <b>Table 23.4</b> and (5) of <b>Table 23.5</b>: “LiEST register” to “LEST register” and “LiTC register”s to “LTC register”s, respectively</li> </ul>
		388	<ul style="list-style-type: none"> <li>• Added Note 3 to <b>Figure 23.23</b></li> </ul>
		394	<ul style="list-style-type: none"> <li>• Corrected a typo “LiEST” in line 7 of <b>23.7.1.1</b> to “LEST”</li> </ul>
		396	<ul style="list-style-type: none"> <li>• Revised whole section of <b>23.8</b> including <b>Figure 23.30</b></li> </ul>
		400	<ul style="list-style-type: none"> <li>• Modified descriptions “Frame transmit completion flag” and “Frame receive completion flag” in <b>Figure 23.32</b> to “Frame/wake-up transmit completion flag” and “Frame/wake-up receive completion flag”, respectively</li> </ul>
		—	<p><b>Chapter 24. CAN Module</b></p> <ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> <li>• Modified read values for unassigned bits from “undefined value” to “0”</li> </ul>
		403	<ul style="list-style-type: none"> <li>• Changed expression “XIN” in <b>Figure 24.1</b> to “Main clock”</li> </ul>
		409	<ul style="list-style-type: none"> <li>• Modified reset value “00h” for CiCKLR register in <b>Figure 24.3</b> to “000X 0000b”; Changed b4 to undefined value</li> </ul>
		410	<ul style="list-style-type: none"> <li>• Corrected a typo “b23-22” for CiBCR register in <b>Figure 24.4</b> to “b23-b22”; Added description “from CAN reset mode” to Note 1</li> </ul>



<b>REVISION HISTORY</b>	<b>R32C/161 Group User's Manual: Hardware</b>
-------------------------	---

Rev.	Date	Description	
		Page	Summary
		412	<ul style="list-style-type: none"> <li>• Corrected a typo “47A13h-47E10h” for address of C1MKR4 in <b>Figure 24.5</b> to “47A13h-47A10h”</li> </ul>
		412, 413	<ul style="list-style-type: none"> <li>• Changed function for b31 to b29 for CiMKRk in <b>Figure 24.5</b> and b29 for CiFIDCRn in <b>Figure 24.6</b> to “Reserved”</li> </ul>
		416	<ul style="list-style-type: none"> <li>• Changed “*” as multiplication sign in <b>Table 24.4</b> to “x”</li> </ul>
		417	<ul style="list-style-type: none"> <li>• Changed functions of b29, b32 to b39, and b44 to b47 for CiMBj in <b>Figure 24.8</b> to “Reserved”; Added “b28”, “b18”, and “b17” to respective bit field box; Corrected a typo “00” for setting values of “DATA0 to DATA7”, “TSL”, and “TSH” to “00h”</li> </ul>
		419	<ul style="list-style-type: none"> <li>• Changed expression “-” in <b>Table 24.6</b> to “X”</li> </ul>
		421	<ul style="list-style-type: none"> <li>• Modified register name “CANi Message Control Register” in the title of <b>24.1.9</b> to “CANi Message Control Register j”</li> </ul>
		423	<ul style="list-style-type: none"> <li>• Deleted a part of description in lines 7 and 8 of <b>24.1.9.9</b></li> </ul>
		428	<ul style="list-style-type: none"> <li>• Changed “0” for bit field b4 of CiTFPCR in <b>Figure 24.14</b> to “X”</li> <li>• Modified a part of description in <b>24.1.12.2</b></li> </ul>
		430	<ul style="list-style-type: none"> <li>• Corrected a typo “TFE bit in the CiTFPCR register” in line 3 of <b>24.1.13</b> to “TFE bit in the CiTFPCR register”</li> </ul>
		437	<ul style="list-style-type: none"> <li>• Modified “0” for bit fields b3 and b4 of 4th read of CiMSSR register in <b>Figure 24.21</b> to “X”</li> </ul>
		438	<ul style="list-style-type: none"> <li>• Modified register name “CAFSR” in line 2 of <b>24.1.18</b> to “CiAFSR”</li> </ul>
		444, 445	<ul style="list-style-type: none"> <li>• Deleted “(8 bits)” from function description for CiRECR and CiTECR in <b>Figures 24.26 and 24.27</b></li> </ul>
		452	<ul style="list-style-type: none"> <li>• Added description “set” to Note 2 for <b>Figure 24.34</b></li> </ul>
		458	<ul style="list-style-type: none"> <li>• Deleted BPR bit from <b>Figure 24.36</b>; Added “P = 0 to 1023” to definition of setting value P</li> </ul>
		469	<ul style="list-style-type: none"> <li>• Moved “CANi wakeup interrupt” from interrupt sources to list of CAN interrupts in <b>24.7</b></li> </ul>
		—	<p><b>Chapter 25. I/O Pins</b></p> <ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> </ul>
		472	<ul style="list-style-type: none"> <li>• Added “b” to all values in <b>Table 25.1</b></li> </ul>
		474, 477	<ul style="list-style-type: none"> <li>• Modified description “Serial bus k output” for bits PSEL2 to PSEL0 in <b>Figures 25.4 and 25.7</b> to “SBk output”</li> </ul>
		474, 478	<ul style="list-style-type: none"> <li>• Modified description “IIOi_j output” for bits PSEL2 to PSEL0 in <b>Figures 25.4 and 25.8</b> to “IIOi output”</li> </ul>
		482	<ul style="list-style-type: none"> <li>• Corrected a typo: “PD_9i bit” in line 7 below <b>Figure 25.12</b> to “PD9_i bit”</li> </ul>
		485	<ul style="list-style-type: none"> <li>• Changed bit name “Serial Bus i Input Pin Switch Bit” in <b>Figure 25.16</b> to “SBli Input Pin Switch Bit” and the description for Note 1</li> <li>• Modified description “Do not use this communication” in <b>Figure 25.17</b> to “Do not use this combination”</li> </ul>
		—	<p><b>Chapter 26. Flash Memory</b></p> <ul style="list-style-type: none"> <li>• Made major text modifications to this chapter</li> <li>• Changed expressions “write” and “rewrite” to “program” when this word is used in combination with “erase”</li> <li>• Corrected a typo “read allay” to “read array”</li> </ul>

<b>REVISION HISTORY</b>	<b>R32C/161 Group User's Manual: Hardware</b>
-------------------------	---

Rev.	Date	Description	
		Page	Summary
		491	<ul style="list-style-type: none"> <li>• Changed the following expressions in <b>Table 26.1</b>: “Operating modes” to “Rewrite modes”, “Block formation” to “Structure”, “Unit to be programmed” to “Program operation”, “Unit to be erased” to “Erase operation”, “How to control program and erase” to “Program/erase controlled by”, and “Protect types” to “Protection types”; Modified specifications; Specified protection types; Deleted row “ROM code protection”</li> </ul>
		493, 494	<ul style="list-style-type: none"> <li>• Revised <b>sections 26.2 through 26.2.3 and Table 26.3</b></li> </ul>
		495	<ul style="list-style-type: none"> <li>• Changed expressions “Flash rewrite address bus” and “Flash rewrite data bus” in <b>Figure 26.3</b> to “Flash memory rewrite address bus” and “Flash memory rewrite data bus”, respectively</li> </ul>
		496	<ul style="list-style-type: none"> <li>• Revised <b>Table 26.5</b></li> </ul>
		497	<ul style="list-style-type: none"> <li>• Added note symbol “(2)” to FEW bit in <b>Figure 26.4</b>; Changed description “erase/write enable mode” in Note 2 to “program/erase enabled”</li> </ul>
		499	<ul style="list-style-type: none"> <li>• Changed FCA bit names and their function descriptions in <b>Figure 26.7</b></li> </ul>
		500	<ul style="list-style-type: none"> <li>• Corrected a typo “b7-4” for FMR1 register in <b>Figure 26.8</b> to “b7-b4”</li> </ul>
		506	<ul style="list-style-type: none"> <li>• Added note symbol “(3)” to “Enter read protect bit status mode” in <b>Table 26.12</b></li> </ul>
		507	<ul style="list-style-type: none"> <li>• Modified descriptions “Read lock bit mode” and “Read protect bit mode” in <b>26.3.3</b> to “Read lock bit status mode” and “Read protect bit status mode”, respectively</li> <li>• Corrected a typo “b5-0” in <b>Tables 26.14 and 26.15</b> to “b5-b0”</li> </ul>
		509	<ul style="list-style-type: none"> <li>• Changed decision timing of FCA bit in <b>Figure 26.14</b></li> </ul>
		509-513	<ul style="list-style-type: none"> <li>• Changed logical value “1” for decision in <b>Figures 26.14 to 26.18</b> to “0”</li> <li>• Added description for FCA bit to <b>26.3.4.4 to 26.3.4.7 and 26.3.4.9</b></li> </ul>
		510	<ul style="list-style-type: none"> <li>• Corrected a typo “FFFF800h” in line 3 of <b>26.3.4.5</b> to “FFFFFF800h”</li> </ul>
		511	<ul style="list-style-type: none"> <li>• Corrected a typo “00D0h” in line 2 of <b>26.3.4.6</b> to “0077h”; Specified “FFFFFF800h” as even address</li> </ul>
		514	<ul style="list-style-type: none"> <li>• Modified expressions “Status/Error” and “Status check” in <b>Table 26.16 and Figure 26.19</b> to “Error” and “Check status”, respectively</li> </ul>
		516	<ul style="list-style-type: none"> <li>• Deleted column “Power Supply” from <b>Table 26.18</b>; Modified “I/O port” for P0_0 to P4_3 to “Input port”; Modified description “P9_0 to P9_7” as input ports to “P9_1, P9_3 to P9_7”</li> </ul>
		519	<ul style="list-style-type: none"> <li>• Changed description “erase/write enable mode” in the second bullet point of <b>26.6.4</b> to “program or erase enabled”</li> <li>• Modified description “erase operation” in the first bullet point of EW1 mode in <b>26.6.5</b> to “block erase operation”</li> <li>• Corrected a typo “FMR1 register” in line 5 of the third bullet point of EW1 mode to “FMR0 register”</li> </ul>
		—	<p><b>Chapter 27. E<sup>2</sup>PROM Emulation data Flash</b></p> <ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> <li>• Changed expressions “write” and “rewrite” to “program” when this word is used in combination with “erase”</li> </ul>

REVISION HISTORY	R32C/161 Group User's Manual: Hardware
------------------	--

Rev.	Date	Description	
		Page	Summary
		523	<ul style="list-style-type: none"> <li>• Modified setting range “00062000h to 00062FFFh” in <b>Figure 27.6</b> to “00062000h to 00062FFEh”</li> </ul>
		524	<ul style="list-style-type: none"> <li>• Corrected a typo “F2FI Register” in the title of <b>Figure 27.7</b> to “E2FI Register”</li> </ul>
		526-528	<ul style="list-style-type: none"> <li>• Changed <b>Figures 27.10 through 27.13</b></li> </ul>
		529	<ul style="list-style-type: none"> <li>• Changed description “erase/write enable mode” in line 2 of <b>27.4</b> to “program or erase enabled”</li> </ul>
			<b>Chapter 28. Electrical Characteristics</b>
		—	<ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> </ul>
		532	<ul style="list-style-type: none"> <li>• Added description about values as Note 2 and its note symbol to “Value” in <b>Table 28.3</b></li> </ul>
		533	<ul style="list-style-type: none"> <li>• Corrected a typo “pots” in the first bullet point in Note 2 for <b>Table 28.4</b> to “ports”</li> </ul>
		535	<ul style="list-style-type: none"> <li>• Modified “P9_6” in <b>Table 28.6</b> to “P9_7”</li> </ul>
		538	<ul style="list-style-type: none"> <li>• Corrected a typo “32 Kbyte block” in <b>Table 28.9</b> to “32 byte block”</li> <li>• Deleted description “the following 10000 hours” from Note 3 for <b>Table 28.9</b></li> <li>• Deleted “V<sub>CC</sub> = 3.0 to 5.5 V” from “measurement condition” in <b>Table 28.10</b></li> </ul>
		539	<ul style="list-style-type: none"> <li>• Changed maximum value for f<sub>SO(PLL)</sub> in <b>Table 28.13</b> from “65” to “80”; Added two measurement conditions</li> </ul>
		540	<ul style="list-style-type: none"> <li>• Changed the order of description of t<sub>rec(STOP)</sub> and t<sub>rec(WAIT)</sub> in <b>Table 28.14 and Figure 28.4</b></li> </ul>
		544, 554	<ul style="list-style-type: none"> <li>• Added “XIN” as “Active” to the first, third, and sixth rows of <b>Tables 28.18 and 28.37</b></li> </ul>
		545	<ul style="list-style-type: none"> <li>• Modified expression “Sample time” for “Characteristics” in <b>Table 28.19</b> to “Sampling time”</li> </ul>
		549, 559	<ul style="list-style-type: none"> <li>• Changed minimum value for “t<sub>W(ADH)</sub>” “2/φ<sub>AD</sub>” in <b>Table 28.30 and 28.49</b> to “3/φ<sub>AD</sub>”</li> </ul>
		549, 560	<ul style="list-style-type: none"> <li>• Corrected a typo “SSCKI input falling time” in <b>Tables 28.33 and 28.51</b> to “SSCKi input falling time”</li> </ul>
		551, 561	<ul style="list-style-type: none"> <li>• Modified expression “TXDi hold time” in <b>Tables 28.33 and 28.52</b> to “TXDi output hold time”</li> </ul>
		555	<ul style="list-style-type: none"> <li>• Modified expression “Differential non-linearity” in <b>Table 28.38</b> to “Differential non-linearity error”</li> </ul>
		556	<ul style="list-style-type: none"> <li>• Corrected the following typos in <b>Table 28.39</b>: “t<sub>W(H)</sub>”, “t<sub>W(L)</sub>”, “t<sub>r</sub>”, and “t<sub>f</sub>” to “t<sub>W(XH)</sub>”, “t<sub>W(XL)</sub>”, “t<sub>r(X)</sub>”, and “t<sub>f(X)</sub>”, respectively</li> </ul>
			<b>Chapter 29. Usage Notes</b>
		—	<ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> </ul>
		566	<ul style="list-style-type: none"> <li>• Modified description for <b>29.2</b> as <b>29.2.1</b></li> </ul>
		567	<ul style="list-style-type: none"> <li>• Modified addresses of registers from TA1M to TA41M in <b>Table 29.2</b>;</li> <li>Deleted part of the last rows for C0CSSR and C1CSSR for CAN module</li> </ul>

<b>REVISION HISTORY</b>	<b>R32C/161 Group User's Manual: Hardware</b>
-------------------------	---

Rev.	Date	Description	
		Page	Summary
		568	<ul style="list-style-type: none"> <li>• Deleted entire section of “Wait Mode” from <b>29.3.2.1</b> and the first bullet point for “Stop Mode” from previous 29.3.2.2 (current <b>29.3.2.1</b>)</li> </ul>
		569	<ul style="list-style-type: none"> <li>• Corrected typos “PR2 register” in <b>29.4.1</b> and “INTi (i = 0 to 5) pin” in <b>29.4.3</b> to “PM2 register” and “<math>\overline{\text{INT}}_i</math> pin (i = 0 to 5)”, respectively</li> </ul>
		570	<ul style="list-style-type: none"> <li>• Modified description “channel i” in line 1 of the first bullet point of <b>29.5.1</b> to “DMDi register”; Deleted whole description of the second and third bullet points; Added two new paragraphs</li> </ul>
		571, 572	<ul style="list-style-type: none"> <li>• Modified description “peripheral clocks” in the fourth bullet point of <b>29.5.2</b> to “peripheral bus clocks”</li> </ul>
		572	<ul style="list-style-type: none"> <li>• Deleted “(i = 0 to 4)” from line 1 of the first bullet point of <b>29.6.2.3</b> and the second bullet point of <b>29.6.2.4</b></li> </ul>
		572	<ul style="list-style-type: none"> <li>• Added “(i = 0 to 4)” to line 1 of the first bullet point of <b>29.6.2.4</b></li> </ul>
		573	<ul style="list-style-type: none"> <li>• Replaced variable “i” with “j” in <b>29.6.3.1</b> and <b>29.6.3.2</b>; Added “(j = 0 to 5)” to the first bullet point of <b>29.6.3.1</b></li> </ul>
		575	<ul style="list-style-type: none"> <li>• Deleted whole description from the third bullet point of <b>29.8.3.2</b></li> </ul>
		577	<ul style="list-style-type: none"> <li>• Changed description “AD0i register” in last line of the seventh bullet point in <b>29.9.2</b> to “AD00 register”; Deleted “(i=0 to 7)” from line 2 of the eighth bullet point; Added description to the ninth bullet point</li> </ul>
		578	<ul style="list-style-type: none"> <li>• Modified addresses of program example in <b>29.10.1.1</b></li> </ul>
		579	<ul style="list-style-type: none"> <li>• Changed description “erase/write enable mode” in the second bullet point of <b>29.11.4</b> to “program or erase enabled”</li> <li>• Modified description “erase operation” in the first bullet point of EW1 mode in <b>29.11.5</b> to “block erase operation”</li> <li>• Corrected a typo “FMR1 register” in line 5 of the third bullet point of EW1 mode to “FMR0 register”</li> </ul>
		581	<ul style="list-style-type: none"> <li>• Changed description “erase/write enable mode” in line 2 of <b>29.12</b> to “program or erase enabled”</li> </ul>
1.20	Jul 04, 2012	—	Third edition released
		—	<p>This manual in general</p> <ul style="list-style-type: none"> <li>• Applied new Renesas templates and formats to the manual</li> <li>• Changed company name to “Renesas Electronics Corporation” and changed related descriptions due to business merger of Renesas Technology Corporation and NEC Electronic Corporation</li> <li>• Changed document number “REJ09B0517-0102” to “R01UH0226EJ0120”</li> <li>• Modified the following expressions: “version J”, “version L”, and “version K” to “J version”, “L version”, and “K version”, respectively (under Chapters 1 and 28)</li> <li>• Modified expressions “a write” to “a write operation”, “calculation transfer” to “calculation result transfer”, and “chained transfer” to “chain transfer”</li> </ul>
		—	<p><b>Chapter 1. Overview</b></p> <ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> </ul>

REVISION HISTORY	R32C/161 Group User's Manual: Hardware
------------------	--

Rev.	Date	Description	
		Page	Summary
		2	<ul style="list-style-type: none"> <li>• Modified expressions “Main clock oscillator stop/re-oscillation detection”, “calculation transfer”, “chained transfer”, and “inputs/ outputs” in <b>Table 1.1</b> to “Main clock oscillator stop/restart detection”, “calculation result transfer”, “chain transfer”, and “I/O ports”, respectively</li> </ul>
		4	<ul style="list-style-type: none"> <li>• Completed all “under development” phases in <b>Table 1.3</b></li> </ul>
		7	<ul style="list-style-type: none"> <li>• Changed order of signals in <b>Figure 1.3</b></li> </ul>
		9	<ul style="list-style-type: none"> <li>• Changed order of timer pins “TA4IN/<math>\bar{U}</math>/TB1IN” in <b>Table 1.5</b> to “TA4IN/TB1IN/<math>\bar{U}</math>”</li> </ul>
		10	<ul style="list-style-type: none"> <li>• Modified expression “fC” in <b>Table 1.6</b> to “low speed clocks”</li> </ul>
		—	<p><b>Chapter 2. CPU</b></p> <ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> </ul>
		13	<ul style="list-style-type: none"> <li>• Corrected a typo “R3R0” in line 3 of <b>2.1.1</b> to “R3R1”</li> </ul>
		—	<p><b>Chapter 3. Memory</b></p> <ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> </ul>
		—	<p><b>Chapter 4. SFRs</b></p> <ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> </ul>
		23	<ul style="list-style-type: none"> <li>• Changed hexadecimal format of reset value for G0BCR0 register in <b>Table 4.7</b> to binary; Changed register name “Group 0 Timer Measurement Prescaler Register” to “Group 0 Time Measurement Prescaler Register”</li> </ul>
		26	<ul style="list-style-type: none"> <li>• Modified expression “XY Control Register” in <b>Table 4.10</b> to “X-Y Control Register”</li> </ul>
		28	<ul style="list-style-type: none"> <li>• Changed register names “UART2 Transmission/Receive Mode Register” and “Increment/Decrement Counting Select Register” in <b>Table 4.12</b> to “UART2 Transmit/Receive Mode Register” and “Increment/Decrement Select Register”, respectively; Changed hexadecimal format of reset values for registers TABSR, ONSF, and TRGSR to binary</li> </ul>
		35	<ul style="list-style-type: none"> <li>• Corrected reset value “X000 X000b” for IFS0 register in <b>Table 4.19</b> to “X0X0 X000b”</li> </ul>
		38	<ul style="list-style-type: none"> <li>• Changed register name “External Interrupt Source Select Register 0” in <b>Table 4.22</b> to “External Interrupt Request Source Select Register 0”</li> </ul>
		52, 53, 66, 67	<ul style="list-style-type: none"> <li>• Changed register name “CANi Acceptance Mask Register k” in <b>Tables 4.36, 4.37, 4.50, and 4.51</b> to “CANi Mask Register k”</li> </ul>
		55, 69	<ul style="list-style-type: none"> <li>• Changed register names “CANi Reception Error Count Register” and “CANi Transmission Error Count Register” in <b>Tables 4.39 and 4.53</b> to “CANi Receive Error Count Register” and “CANi Transmit Error Count Register”, respectively; Corrected reset value “XXXX XX00b” for CiMSMR register to “0000 0000b”</li> </ul>
		—	<p><b>Chapter 5. Resets</b></p> <ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> </ul>
		70	<ul style="list-style-type: none"> <li>• Changed expression “operating level” in (2) of B in <b>5.1</b> to “operating voltage”</li> </ul>
		—	<p><b>Chapter 6. Power Management</b></p> <ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> </ul>

REVISION HISTORY	R32C/161 Group User's Manual: Hardware
------------------	--

Rev.	Date	Description	
		Page	Summary
		74	<ul style="list-style-type: none"> <li>• Modified descriptions “main clock oscillator active” and “PLL clock oscillator active” in Note 2 of <b>Figure 6.2</b> to “main clock oscillator enabled” and “PLL oscillator enabled”</li> </ul>
		77	<ul style="list-style-type: none"> <li>• Modified VDEN bit name in <b>Figure 6.4</b> to “Low Voltage Detector Enable Bit”; Modified its function descriptions “low voltage detection disabled” and “low voltage detection enabled” to “low voltage detector disabled” and “low voltage detector enabled”, respectively</li> </ul>
		—	<p><b>Chapter 7. Clock Generator</b></p> <ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> <li>• Modified function descriptions of bits CM05 and CM10 to the following: “main clock oscillator enabled”, “main clock oscillator disabled”, “PLL oscillator enabled”, and “PLL oscillator disabled”</li> </ul>
		82	<ul style="list-style-type: none"> <li>• Modified expression “fC” in <b>Figure 7.1</b> to “Low speed clock”; Added BCS bit</li> </ul>
		83	<ul style="list-style-type: none"> <li>• Deleted the last sentence from Note 2 in <b>Figure 7.2</b>; Added description for bit settings to Note 6</li> </ul>
		84	<ul style="list-style-type: none"> <li>• Changed expression “fC” in <b>Figure 7.3</b> to “a low speed clock”; Added Note 7</li> </ul>
		85	<ul style="list-style-type: none"> <li>• Modified CM10 and CM15 bit names “PLL Clock Oscillator Stop Bit” and “XIN-XOUT Drive Power Select Bit” in <b>Figure 7.4</b> to “PLL Oscillator Stop Bit” and “XIN-XOUT Drive Strength Select Bit”; Modified description of Note 2; Added Note 4</li> <li>• Modified function descriptions of CM20 bit in <b>Figure 7.5</b> to “Disable oscillator stop detection” when it is 0 and “Enable oscillator stop detection” when it is 1; Corrected “CM02 bit” in Note 3 to “CM20 bit”</li> </ul>
		86	<ul style="list-style-type: none"> <li>• Modified description of Note 1 in <b>Figure 7.6</b></li> </ul>
		88	<ul style="list-style-type: none"> <li>• Modified the following descriptions in Note 3 in <b>Figure 7.9</b>: “start main clock oscillator running” for CM05 bit to “main clock oscillator enabled/disabled” and “start PLL clock oscillator running” for CM10 bit to “PLL oscillator enabled/disabled”; Added Note 6</li> </ul>
		89	<ul style="list-style-type: none"> <li>• Added description to Note 1 in <b>Figure 7.10</b></li> </ul>
		93	<ul style="list-style-type: none"> <li>• Changed SEO bit name and its function description in <b>Figure 7.15</b></li> <li>• Added Note 1 to <b>Figure 7.16</b></li> </ul>
		95	<ul style="list-style-type: none"> <li>• Modified the last sentence in <b>7.1.4</b></li> </ul>
		96	<ul style="list-style-type: none"> <li>• Deleted the last sentence in parenthesis in <b>7.2</b></li> <li>• Added description of stopping main clock to lines 6 to 7 of <b>7.2.1</b></li> </ul>
		98, 106, 109	<ul style="list-style-type: none"> <li>• Modified expression “fC” in line 1 of <b>7.6</b> to “Low speed clocks” and “fC” in <b>Tables 7.3, 7.4, and 7.6</b> to “a low speed clock”</li> </ul>
		99	<ul style="list-style-type: none"> <li>• Revised the entire paragraph of <b>7.7</b></li> </ul>
		100	<ul style="list-style-type: none"> <li>• Revised <b>7.7.1</b> entirely</li> </ul>
		102-104	<ul style="list-style-type: none"> <li>• Moved <b>Figures 7.18 to 7.20</b> into <b>7.7.1</b></li> </ul>
		102	<ul style="list-style-type: none"> <li>• Corrected a typo “f(XPLL)” in the third row of <b>Figure 7.18</b> to “f(PLL)”; Deleted Note 4</li> </ul>
		103	<ul style="list-style-type: none"> <li>• Corrected a typo “CM0 = 1” in the fifth row of <b>Figure 7.19</b> to “CM05 = 1”; Deleted Note 3</li> </ul>

REVISION HISTORY	R32C/161 Group User's Manual: Hardware
------------------	--

Rev.	Date	Description	
		Page	Summary
		104	<ul style="list-style-type: none"> <li>• Corrected “CM31 = 1” in the first row and “CM10 = 0” for “Low speed mode” in the second row of <b>Figure 7.20</b> to “CM31 = 0” and “CM10 = 1”, respectively; Deleted Note 3</li> </ul>
		105	<ul style="list-style-type: none"> <li>• Modified descriptions in lines 1 to 3 of <b>7.7.2</b></li> <li>• Added description “in wait mode” to line 3 of <b>7.7.2.1</b></li> </ul>
		106	<ul style="list-style-type: none"> <li>• Revised <b>7.7.2.4</b></li> </ul>
		108	<ul style="list-style-type: none"> <li>• Modified description of line 1 of <b>7.7.3</b></li> </ul>
		109	<ul style="list-style-type: none"> <li>• Added interrupt numbers from 0 to 63 to line 2 of <b>7.7.3.3</b> as factors of exiting stop mode</li> </ul>
		—	<p><b>Chapter 8. Bus</b></p> <ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> </ul>
		—	<p><b>Chapter 9. Protection</b></p> <ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> <li>• Changed title of each section in this chapter</li> </ul>
		—	<p><b>Chapter 10. Interrupts</b></p> <ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> </ul>
		118	<ul style="list-style-type: none"> <li>• Modified description of Note 1 in <b>Figure 10.1</b></li> </ul>
		119	<ul style="list-style-type: none"> <li>• Modified descriptions in the second paragraph in (5) of <b>10.2</b></li> </ul>
		121, 122, 129	<ul style="list-style-type: none"> <li>• Modified description of jump operation in <b>10.5, Table 10.1, and below Figure 10.4</b></li> </ul>
		132	<ul style="list-style-type: none"> <li>• Moved description of Note 1 to (2) of <b>10.6.4</b></li> </ul>
		133	<ul style="list-style-type: none"> <li>• Modified description of Note 1 of <b>Table 10.7</b></li> </ul>
		136	<ul style="list-style-type: none"> <li>• Changed expression “Serial bus 0” in <b>Figure 10.8</b> to “Serial bus interface 0”; Deleted “Bits RLVL2 to RLVL0 in the RIPL2 register” and associated signal lines; Changed expression “DMAC II” to “DMA II transfer complete”</li> </ul>
		139	<ul style="list-style-type: none"> <li>• Corrected register symbol “IIOiE” in <b>10.12</b> to “IIOiE”</li> </ul>
		140	<ul style="list-style-type: none"> <li>• Changed function description of b0 in <b>Figure 10.13</b>; Modified description in Note 3</li> </ul>
		—	<p><b>Chapter 11. Watchdog Timer</b></p> <ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> </ul>
		143	<ul style="list-style-type: none"> <li>• Modified description of lines 4 to 5 in <b>11. Watchdog Timer</b>; Corrected register symbol “WKD” in line 9 to “WDK”; Modified CPU clock frequency to “48 MHz” and watchdog timer period to “21.8 ms”</li> </ul>
		144	<ul style="list-style-type: none"> <li>• Added Notes 1 and 2 to <b>Figure 11.2</b></li> </ul>
		145	<ul style="list-style-type: none"> <li>• Added Note 2 to <b>Figure 11.3</b></li> </ul>
		146	<ul style="list-style-type: none"> <li>• Added Note 3 to <b>Figure 11.5</b></li> </ul>
		—	<p><b>Chapter 12. DMAC</b></p> <ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> </ul>
		148	<ul style="list-style-type: none"> <li>• Modified descriptions of timer- and UART-associated interrupt requests in “DMA request sources” in <b>Table 12.1</b>; Modified expression “DMA transfer startup” to “DMA transfer start-up”; Modified description “more than 00000001h” in “DMA transfer start-up” to “other than 00000000h”</li> </ul>
		156	<ul style="list-style-type: none"> <li>• Modified descriptions in <b>12.1</b></li> </ul>

REVISION HISTORY	R32C/161 Group User's Manual: Hardware
------------------	--

Rev.	Date	Description	
		Page	Summary
			<b>Chapter 13. DMAC II</b>
		— 162	<ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> <li>• Corrected source address “FFFFFFFh” in Note 1 of <b>Table 13.1</b> to “FFFFFFFFh”</li> </ul>
		164, 165	<ul style="list-style-type: none"> <li>• Corrected bit symbol “IIRLT” in the fifth bullet point of <b>13.1</b> to “IRLT”</li> <li>• Changed expression “DMA II transfer complete interrupt vector address” in lines 3 to 4 and the seventh bullet point of <b>13.1.2</b> and <b>Figure 13.2</b> to “jump address for the DMA II transfer complete interrupt handler”</li> </ul>
		164, 167	<ul style="list-style-type: none"> <li>• Modified expression “interrupt vector” in <b>Figure 13.2</b> and line 1 of <b>13.1.4</b> to “interrupt vector space”</li> </ul>
		165	<ul style="list-style-type: none"> <li>• Changed description “jump address” in the seventh bullet point of <b>13.1.2</b> to “start address”</li> </ul>
		166	<ul style="list-style-type: none"> <li>• Changed bit names of OPER bit and bits CNT2 to CNT0 in <b>Figure 13.3</b> to “Calculation Result Transfer Select Bit” and “Number of Transfers Setting Bit”, respectively</li> </ul>
		170	<ul style="list-style-type: none"> <li>• Modified descriptions in <b>Figure 13.5</b></li> </ul>
		—	<b>Chapter 14. Programmable I/O Ports</b>
		—	<ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> </ul>
		—	<b>Chapter 15. Timers</b>
		175	<ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> <li>• Separated signal for overflow or underflow from interrupt signal in <b>Figure 15.2</b></li> </ul>
		179	<ul style="list-style-type: none"> <li>• Deleted “Counting” from UDF register name and bit names of bits TA4UD to TA0UD in <b>Figure 15.7</b></li> </ul>
		187	<ul style="list-style-type: none"> <li>• Changed MR2 bit name “Increment/Decrement Count Switching Source Select Bit” in <b>Figure 15.12</b> to “Increment/Decrement Switching Source Select Bit”; Corrected bit symbols “TAiTGH and TAIiTGL” in Note 5 to “TAjTGH and TAJiTGL”</li> </ul>
		189 207	<ul style="list-style-type: none"> <li>• Corrected a typo “TA4NR” in line 3 of <b>15.1.3</b> to “TA4MR”</li> <li>• Modified description in <b>15.3.3.1</b></li> <li>• Modified description “TBjS bit” in the first bullet point of <b>15.3.3.2</b> to “TBjS bit in the TABSR or TBSR register”; Modified “TBj” in the Eighth bullet point to “timer Bj”</li> </ul>
		—	<b>Chapter 16. Three-phase Motor Control Timers</b>
		211	<ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> <li>• Changed function descriptions “Timer A reload control signal is 0” and “Timer A reload control signal is 1” for INV13 bit in <b>Figure 16.3</b> to “Timer A1 reload control signal is 0” and “Timer A1 reload control signal is 1”, respectively; Modified Note 1</li> </ul>
		215	<ul style="list-style-type: none"> <li>• Changed bit functions of bits MR2 and MR3 for TB2MR register in <b>Figure 16.8</b></li> </ul>
		216	<ul style="list-style-type: none"> <li>• Changed function description of PWCON bit for TB2SC register in <b>Figure 16.9</b></li> </ul>
		217	<ul style="list-style-type: none"> <li>• Deleted description in lines 8 to 9 of <b>16.3</b>; Added “TA4M” to line 10</li> </ul>
		222	<ul style="list-style-type: none"> <li>• Corrected bit symbol “INV06” in Note 3 of <b>Figure 16.16</b> to “INV16”</li> </ul>



<b>REVISION HISTORY</b>	<b>R32C/161 Group User's Manual: Hardware</b>
-------------------------	---

Rev.	Date	Description	
		Page	Summary
		223	<ul style="list-style-type: none"> <li>• Corrected register symbol “INV1” in Note 2 of <b>Figure 16.18</b> to “INVC1”</li> </ul>
		225	<ul style="list-style-type: none"> <li>• Added description of SDE bit to <b>16.6.1</b>; Modified text representation</li> <li>• Modified “overflow” in <b>16.6.2</b> to “underflow”</li> </ul>
		—	<b>Chapter 17. Serial Interface</b>
		231, 232	<ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> <li>• Modified CRD bit name in <b>Figures 17.5 and 17.6</b> to “CTS Function Disable Bit”; Modified their function descriptions</li> </ul>
		231	<ul style="list-style-type: none"> <li>• Deleted Note 1 from <b>Figure 17.5</b></li> </ul>
		233	<ul style="list-style-type: none"> <li>• Modified bit description of UiIRS bit when it is 0 in <b>Figure 17.7</b> to “Transmit buffer is empty (TI = 1)”; Modified bit name of UiLCH bit to “Logic Inversion Select Bit”</li> </ul>
		235	<ul style="list-style-type: none"> <li>• Modified CSC bit name in <b>Figure 17.11</b> to “Clock Synchronization Bit”; Deleted “of the SCLi” from function description of SWC bit</li> </ul>
		236	<ul style="list-style-type: none"> <li>• Corrected “UiBRG count source” in function description of bits DL2 to DL0 in <b>Figure 17.12</b> to “baud rate generator count source”</li> </ul>
		237	<ul style="list-style-type: none"> <li>• Deleted “of the SCLi” from function description of SWC9 bit in <b>Figure 17.13</b></li> </ul>
		237, 263	<ul style="list-style-type: none"> <li>• Corrected a typo “STARREQ” in Note 3 of <b>Figure 17.13</b> and in line 1 of <b>17.3.2</b> to “STAREQ”</li> </ul>
		243	<ul style="list-style-type: none"> <li>• Modified “TXEPT flag” in <b>Figure 17.18</b> to “TXEPT bit”; Corrected bit symbol “UiRS” in the fourth dash to “UiIRS”</li> </ul>
		251, 252	<ul style="list-style-type: none"> <li>• Corrected bit functions of UiIRS bit in the fourth dash in <b>Figures 17.23 and 17.24</b></li> </ul>
		255, 256	<ul style="list-style-type: none"> <li>• Changed expression “Transmit/receive clock” in <b>Figures 17.27 and 17.28</b> to “CLKi”</li> </ul>
		260, 261	<ul style="list-style-type: none"> <li>• Divided Table 17.11 into <b>Tables 17.11 and 17.12</b></li> </ul>
		273	<ul style="list-style-type: none"> <li>• Moved description in the fourth dash in <b>17.5.3.1</b> to the second dash</li> </ul>
		274	<ul style="list-style-type: none"> <li>• Added a new paragraph for “Reset Procedure on Communication Error” as <b>17.5.5</b></li> </ul>
		—	<b>Chapter 18. A/D Converter</b>
		280	<ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> <li>• Modified expression “sample &amp; hold function” in function description of SMP bit in <b>Figure 18.4</b> to “sample and hold function”</li> </ul>
		282	<ul style="list-style-type: none"> <li>• Modified “DMA” in Note 4 of <b>Figure 18.7</b> to “DMAC”</li> </ul>
		284, 286, 287	<ul style="list-style-type: none"> <li>• Modified “DMA” in the first bullet point in “Reading A/D converted result” of <b>Tables 18.3, 18.5, and 18.6</b> to “DMAC”</li> </ul>
		287	<ul style="list-style-type: none"> <li>• Modified description of the number of prioritized pins in line 1 of <b>18.1.5</b> and “Function” in <b>Table 18.6</b></li> </ul>
		289	<ul style="list-style-type: none"> <li>• Deleted description “(AN0 to AN7, ANEX0, ANEX1 as analog input port)” from line 10 of <b>18.2.6</b></li> </ul>
		294	<ul style="list-style-type: none"> <li>• Corrected “AD0i” in the ninth bullet point of <b>18.3.2</b> to “AD00”</li> </ul>
		—	<b>Chapter 19. CRC Calculator</b>
		295	<ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> <li>• Corrected a typo “CRC_CCITT” in line 2 of <b>19. CRC Calculator</b> to “CRC-CCITT”</li> </ul>

REVISION HISTORY	R32C/161 Group User's Manual: Hardware
------------------	--

Rev.	Date	Description	
		Page	Summary
		295	<ul style="list-style-type: none"> <li>• Modified <b>Figure 19.1</b></li> </ul>
		—	<p><b>Chapter 20. X-Y Conversion</b></p> <ul style="list-style-type: none"> <li>• Made minor text modifications to this chapter</li> </ul>
		—	<p><b>Chapter 21. Intelligent I/O</b></p> <ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> </ul>
		303	<ul style="list-style-type: none"> <li>• Changed expression “the <math>\overline{\text{INT0}}</math> pin” in <b>Figure 21.1</b> to “the <math>\overline{\text{INT0}}</math> pin or the <math>\overline{\text{INT1}}</math> pin”</li> </ul>
		306	<ul style="list-style-type: none"> <li>• Changed expression “<math>\overline{\text{INT0}}</math> pin” in <b>Figure 21.4</b> to “<math>\overline{\text{INT0}}/\overline{\text{INT1}}</math> pin”; Changed bit name of bits UD1 and UD0 “Increment/Decrement Counting Control Bit” to “Increment/Decrement Control Bit”; Added Note 2</li> </ul>
		307	<ul style="list-style-type: none"> <li>• Deleted Note 3 from <b>Figure 21.5</b></li> </ul>
		314	<ul style="list-style-type: none"> <li>• Modified descriptions of “Reset Conditions” in <b>Table 21.2</b>; Changed description of the timer counter to start decrementing in the first bullet point of “Other functions”</li> </ul>
		315	<ul style="list-style-type: none"> <li>• Changed expression “<math>\overline{\text{INT0}}</math> pin” in <b>Figure 21.14</b> to “<math>\overline{\text{INT0}}/\overline{\text{INT1}}</math> pin”</li> </ul>
		318	<ul style="list-style-type: none"> <li>• Changed expression “<math>\overline{\text{INT0}}</math>” in <b>Figure 21.17</b> to “<math>\overline{\text{INT0}}/\overline{\text{INT1}}</math>”</li> </ul>
		322	<ul style="list-style-type: none"> <li>• Moved “k = 4, 5” in <b>Figure 21.20</b> to its figure title</li> </ul>
		325, 327, 330	<ul style="list-style-type: none"> <li>• Modified “Input to the IIO0_j pin” in <b>Figures 21.21 to 21.23</b> to “IIO0_j pin”</li> </ul>
		328, 329	<ul style="list-style-type: none"> <li>• Divided Table 21.9 into <b>Tables 21.9 and 21.10</b></li> </ul>
		—	<p><b>Chapter 22. Serial Bus Interface</b></p> <ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> </ul>
		336	<ul style="list-style-type: none"> <li>• Corrected a typo “SBUMS bit” in <b>22. Serial Bus Interface</b> to “SSUMS bit”</li> </ul>
		337, 339	<ul style="list-style-type: none"> <li>• Corrected typos “SS0CK pin” and “SS0RDR register” in <b>Tables 22.1 and 22.2</b> to “SSCK0 pin” and “SS0TDR register”, respectively</li> </ul>
		339	<ul style="list-style-type: none"> <li>• Modified “SSI0 (I): Data input pin” and “SSO0 (O): Data output pin” in <b>Table 22.2</b> to “SSI0 (I/O): Data I/O pin” and “SSO0 (I/O): Data I/O pin”, respectively</li> </ul>
		341	<ul style="list-style-type: none"> <li>• Corrected “b6 b5 b4” in “Function” of bits CKS2 to CKS0 in <b>Figure 22.3</b> to “b2 b1 b0”</li> </ul>
		343	<ul style="list-style-type: none"> <li>• Added description “Bits left during transmission/reception” to “Function” of bits BC2 to BC0 in <b>Figure 22.5</b></li> </ul>
		347	<ul style="list-style-type: none"> <li>• Modified expressions “input/output pin” and “Input/output pin” in <b>Figure 22.9</b> to “I/O pin”</li> </ul>
		363	<ul style="list-style-type: none"> <li>• Corrected bit symbol “MSL” in line 14 of <b>22.1.7</b> to “MLS”</li> <li>• Modified “flags RDRF and ORER” in line 5 of <b>22.1.7.1</b> to “bits RDRF and ORER”</li> </ul>
		—	<p><b>Chapter 23. LIN Module</b></p> <ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> </ul>
		371	<ul style="list-style-type: none"> <li>• Modified descriptions “Break dominant” and “Break delimiter” in <b>Table 23.1</b> to “Transmit break length” and “Transmit break delimiter length”, respectively</li> </ul>
		372	<ul style="list-style-type: none"> <li>• Modified pin names “LINOUT” and “LININ” in <b>Figure 23.1</b> to “LIN0OUT” and “LIN0IN”, respectively</li> </ul>

<b>REVISION HISTORY</b>	<b>R32C/161 Group User's Manual: Hardware</b>
-------------------------	---

Rev.	Date	Description	
		Page	Summary
		375	<ul style="list-style-type: none"> <li>• Added Note 1 to <b>Figure 23.3</b></li> </ul>
		377	<ul style="list-style-type: none"> <li>• Changed Note 4 in <b>Figure 23.7</b></li> </ul>
		380	<ul style="list-style-type: none"> <li>• Modified bit names for bits BLT3 to BLT0 and bits BDT1 and BDT0 in <b>Figure 23.10</b> to “Transmit Break (Low) Length Setting Bit” and “Transmit Break Delimiter (High) Length Setting Bit”, respectively</li> </ul>
		383	<ul style="list-style-type: none"> <li>• Corrected reset value of LSC register in <b>Figure 23.14</b> to “0000 0000b”; Changed bit name of bits OM1 and OM0 to “Operating Mode Select Bit”; Modified function description of b7 to b2</li> </ul>
		385	<ul style="list-style-type: none"> <li>• Changed bit name of bits OMM1 and OMM0 in <b>Figure 23.16</b> to “Operating Mode Monitor Flag”; Added description to Note 1; Added Note 3</li> </ul>
		386	<ul style="list-style-type: none"> <li>• Added description to Note 1 in <b>Figure 23.17</b></li> </ul>
		389	<ul style="list-style-type: none"> <li>• Modified description of the third bullet point in <b>Table 23.3</b>; Modified bit symbols “BFTL3 to BFTL0” and “BFTD1 and BFTD0” to “BLT3 to BLT0” and “BDT1 and BDT0”, respectively</li> </ul>
		390, 391	<ul style="list-style-type: none"> <li>• Changed (4) for “LIN Module Processing” in <b>Tables 23.4 and 23.5</b></li> </ul>
		393	<ul style="list-style-type: none"> <li>• Corrected a typo “BPR0” in Note 1 of <b>Table 23.6</b> to “BRP0”</li> </ul>
		400	<ul style="list-style-type: none"> <li>• Revised <b>Figure 23.30</b></li> </ul>
		401	<ul style="list-style-type: none"> <li>• Added detecting condition to “Input signal low detection” in <b>Table 23.8</b></li> </ul>
		404	<ul style="list-style-type: none"> <li>• Added description to lines 9 to 10 in <b>23.11</b></li> <li>• Modified <b>Figures 23.32 and 23.33</b></li> </ul>
		—	<p><b>Chapter 24. CAN Module</b></p> <ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> <li>• Changed expression “8/3 encoder” to “8-to-3 priority encoder”</li> </ul>
		405	<ul style="list-style-type: none"> <li>• Modified “Table 24.1” in line 5 of <b>24. CAN Module</b> to “Tables 24.1 and 24.2”</li> </ul>
		409	<ul style="list-style-type: none"> <li>• Modified RBOC bit name in <b>Figure 24.2</b> to “Forced Recovery From Bus-off Bit”; Modified its function description</li> </ul>
		417	<ul style="list-style-type: none"> <li>• Modified Note 2 in <b>Figure 24.6</b></li> </ul>
		421	<ul style="list-style-type: none"> <li>• Modified Note 4 in <b>Figure 24.8</b></li> </ul>
		426	<ul style="list-style-type: none"> <li>• Corrected “fCAN (CAN system clock)” in line 4 of <b>24.1.9.5</b> to “the peripheral bus clock”</li> </ul>
		428	<ul style="list-style-type: none"> <li>• Changed description of Note 2 in <b>Figure 24.11</b></li> </ul>
		429	<ul style="list-style-type: none"> <li>• Corrected “fCAN” in line 5 of <b>24.1.10.3</b> to “the peripheral bus clock”</li> </ul>
		435	<ul style="list-style-type: none"> <li>• Changed function description of b7 in <b>Figure 24.17</b> to “No register bit; this bit is read as 0”</li> </ul>
		442	<ul style="list-style-type: none"> <li>• Changed expression “3/8 decoder” in <b>Figure 24.23</b> to “3-to-8 decoder”; Moved “(j = 0 to 31)” to its figure title</li> </ul>
		450	<ul style="list-style-type: none"> <li>• Moved footnote “(4)” from “Bit Name” to “Function” in <b>Figure 24.28</b></li> </ul>
		459	<ul style="list-style-type: none"> <li>• Changed expression “MCU hardware reset or software reset” in line 2 of <b>24.2.3</b> to “a MCU reset”</li> </ul>
		460	<ul style="list-style-type: none"> <li>• Corrected a typo “CiSTR” in the third bullet point of the third paragraph of <b>24.2.4</b> to “CiTCR”</li> </ul>
		462	<ul style="list-style-type: none"> <li>• Corrected q value in <b>Figure 24.36</b> to “q = 2, 3, 4”</li> </ul>

REVISION HISTORY	R32C/161 Group User's Manual: Hardware
------------------	--

Rev.	Date	Description	
		Page	Summary
		468	<ul style="list-style-type: none"> <li>• Moved “(j = 0 to 31)” and “(k = 0 to 7)” in <b>Figure 24.42</b> to its figure title</li> </ul>
		470-472	<ul style="list-style-type: none"> <li>• Moved “(j = 0 to 31)” in <b>Figures 24.43 to 24.45</b> to their figure titles</li> </ul>
		—	<b>Chapter 25. I/O Pins</b>
		474	<ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> <li>• Deleted ASEL from a factor of pull-up resistor being separated from peripheral functions in <b>25. I/O Pins</b> and <b>Figure 25.1</b></li> <li>• Modified descriptions “has neither the bit 5 of the function select register nor the PDi register” in lines 10 to 11 and “The bit 1 of the function select register and the PDi register” in lines 11 to 12 below <b>Figure 25.1</b> to “has no function select register or bit 5 in the PD8 register” and “The function select register and bit 1 in the PD9 register”, respectively</li> </ul>
		478	<ul style="list-style-type: none"> <li>• Corrected reset values of registers P1_0S to P1_4S “XXXX X000b” in <b>Figure 25.4</b> to “0XXX X000b”</li> </ul>
		478, 482	<ul style="list-style-type: none"> <li>• Changed expression “IIO0 output” in <b>Figures 25.4 and 25.8</b> to “IIO0_i output”</li> </ul>
		480	<ul style="list-style-type: none"> <li>• Corrected description “PD3_i register” in line 4 below <b>Figure 25.6</b> to “PD3_i bit”</li> </ul>
		484	<ul style="list-style-type: none"> <li>• Corrected addresses of registers P7_4S to P7_6S in <b>Figure 25.10</b> to “400D9h”, “400DBh”, “400DDh”</li> </ul>
		489	<ul style="list-style-type: none"> <li>• Modified expression “serial bus 0” in function description for IFS50 bit in <b>Figure 25.16</b> to “SBI0”</li> </ul>
		—	<b>Chapter 26. Flash Memory</b>
		497	<ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> <li>• Deleted descriptions “erase” and “by using the serial programmer” from “ROM Code Protection” in <b>Table 26.3</b>; Added “erase” to “ID Code Protection”</li> <li>• Deleted description “use the serial programmer” from line 3 of <b>26.2.2</b></li> </ul>
		498	<ul style="list-style-type: none"> <li>• Assigned “OFS” to the reserved ID code area in address FFFFFFFFh in <b>Figure 26.2</b>; Added Note 1</li> </ul>
		500	<ul style="list-style-type: none"> <li>• Changed the following expressions in <b>Table 26.5</b>: “the program or the block erase command” to “the program command or the block erase command” and “the read status register command” and “the ready status register command” to “the enter read status register mode command”</li> </ul>
		506, 508	<ul style="list-style-type: none"> <li>• Changed expressions “<math>\overline{CS0}</math>” and “A23 to A0, <math>\overline{BC3}</math> to <math>\overline{BC0}</math>” in <b>Figures 26.12 and 26.13</b> to “Chip select” and “Address”, respectively</li> </ul>
		524	<ul style="list-style-type: none"> <li>• Modified descriptions of the first bullet points in <b>26.6.7 and 26.6.8</b></li> </ul>
		—	<b>Chapter 27. E<sup>2</sup>PROM Emulation Data Flash</b>
		526	<ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> <li>• Added description to Note 3 of <b>Figure 27.2</b>; Deleted description from Note 5</li> </ul>
		532	<ul style="list-style-type: none"> <li>• Modified description for EERR bit setting in <b>Figure 27.12</b> to “EERR bit in the E2FS0 register is 0?”</li> </ul>

REVISION HISTORY	R32C/161 Group User's Manual: Hardware
------------------	--

Rev.	Date	Description	
		Page	Summary
		—	<b>Chapter 28. Electrical Characteristics</b> <ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> <li>• Changed expression “clock period” to “clock cycle time”</li> </ul>
		541, 542	<ul style="list-style-type: none"> <li>• Changed expression “Programming and erasure endurance” in <b>Tables 28.8 and 28.9</b> to “Program/erase cycles”; Changed its unit “times” to “Cycles”</li> </ul>
		545	<ul style="list-style-type: none"> <li>• Changed expressions “<math>\overline{CS0}</math>” and “A23 to A0, <math>\overline{BC0}</math> to <math>\overline{BC3}</math>” in <b>Figure 28.5</b> to “Chip select” and “Address”, respectively</li> </ul>
		547, 557	<ul style="list-style-type: none"> <li>• Modified “LININ” in <b>Tables 28.17 and 28.36</b> to “LIN0IN”</li> </ul>
		553, 563	<ul style="list-style-type: none"> <li>• Corrected “INTi” in the title of <b>Tables 28.31 and 28.50</b> to “<math>\overline{INTi}</math>”</li> </ul>
		—	<b>Chapter 29. Usage Notes</b> <ul style="list-style-type: none"> <li>• Modified wording and enhanced description in this chapter</li> </ul>
		570, 571	<ul style="list-style-type: none"> <li>• Changed orders of <b>Tables 29.1 and 29.2</b></li> </ul>
		577	<ul style="list-style-type: none"> <li>• Modified description in <b>29.6.3.1</b></li> <li>• Modified description “TBjS bit” in the first bullet point of <b>29.6.3.2</b> to “TBjS bit in the TABSR or TBSR register”; Modified “TBj” in the Eighth bullet point to “timer Bj”</li> </ul>
		578	<ul style="list-style-type: none"> <li>• Added description of SDE bit to <b>29.7.1</b>; Modified text representation</li> <li>• Modified “overflow” in <b>29.7.2</b> to “underflow”</li> </ul>
		579	<ul style="list-style-type: none"> <li>• Moved description the fourth dash in <b>29.8.3.1</b> to the second dash</li> </ul>
		580	<ul style="list-style-type: none"> <li>• Added a new paragraph for “Reset Procedure on Communication Error” as <b>29.8.5</b></li> </ul>
		582	<ul style="list-style-type: none"> <li>• Corrected “AD0i” in the ninth bullet point of <b>29.9.2</b> to “AD00”</li> </ul>
		585	<ul style="list-style-type: none"> <li>• Modified descriptions of the first bullet points in <b>29.11.7 and 29.11.8</b></li> </ul>
		—	<b>Appendix 1. Package Dimensions</b> <ul style="list-style-type: none"> <li>• Added a seating plane to the drawing of package dimension</li> </ul>
		587	<ul style="list-style-type: none"> <li>• Added a seating plane to the drawing of package dimension</li> </ul>

---

R32C/161 Group User's Manual: Hardware

Publication Date: Rev.0.51 Dec 24, 2008  
Rev.1.20 Jul 04, 2012

Published by: Renesas Electronics Corporation

---

**SALES OFFICES**

Renesas Electronics Corporation

<http://www.renesas.com>Refer to "<http://www.renesas.com/>" for the latest and detailed information.

**Renesas Electronics America Inc.**  
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.  
Tel: +1-408-588-6000, Fax: +1-408-588-6130

**Renesas Electronics Canada Limited**  
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada  
Tel: +1-905-898-5441, Fax: +1-905-898-3220

**Renesas Electronics Europe Limited**  
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K  
Tel: +44-1628-585-100, Fax: +44-1628-585-900

**Renesas Electronics Europe GmbH**  
Arcadiastrasse 10, 40472 Düsseldorf, Germany  
Tel: +49-211-65030, Fax: +49-211-6503-1327

**Renesas Electronics (China) Co., Ltd.**  
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China  
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

**Renesas Electronics (Shanghai) Co., Ltd.**  
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China  
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

**Renesas Electronics Hong Kong Limited**  
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong  
Tel: +852-2886-9318, Fax: +852-2886-9022/9044

**Renesas Electronics Taiwan Co., Ltd.**  
13F, No. 363, Fu Shing North Road, Taipei, Taiwan  
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

**Renesas Electronics Singapore Pte. Ltd.**  
1 harbourFront Avenue, #06-10, keppel Bay Tower, Singapore 098632  
Tel: +65-6213-0200, Fax: +65-6278-8001

**Renesas Electronics Malaysia Sdn.Bhd.**  
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia  
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

**Renesas Electronics Korea Co., Ltd.**  
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea  
Tel: +82-2-558-3737, Fax: +82-2-558-5141

R32C/161 Group



Renesas Electronics Corporation

R01UH0226EJ0120  
(Previous Number: REJ09B0517-0102)