User's Manual

16

RL78/G21

User's Manual: Hardware

16-Bit Single-Chip Microcontrollers

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).

Renesas Electronics
www.renesas.com

Rev.1.00 May 2026

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 or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving 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, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. 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 shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended 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; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or 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 damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, 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 and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall 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 shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.
(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

© 2026 Renesas Electronics Corporation. All rights reserved.

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Precaution against Electrostatic Discharge (ESD)
   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on
   The state of the product is undefined at the time 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 time 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 time 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 time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state
   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. Handling of unused pins
   Handle unused pins in accordance 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 the 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.
5. Clock signals
   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is 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. Additionally, 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.
6. Voltage application waveform at input pin
   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses
   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products
   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

How to Use This Manual

Readers

This manual is intended for user engineers who wish to understand the functions of the RL78/G21 and design and develop application systems and programs for these devices.
The target products are as follows.

- 20-pin: R7F103G6x (x = G, J)
- 32-pin: R7F103GBx (x = G, J)
- 44-pin: R7F103GFx (x = G, J)
- 48-pin: R7F103GGx (x = G, J)
- 64-pin: R7F103GLx (x = G, J)

Purpose

This manual is intended to give users an understanding of the functions described in the **Organization** below.

Organization

The RL78/G21 manual is separated into two parts: this manual and User's Manual: Software (common to the RL78 family).

RL78/G21 User's Manual: Hardware (This Manual)	**RL78 Family User's Manual: Software**
• Pin functions • Internal block functions • Interrupts • Other on-chip peripheral functions • Electrical characteristics	• CPU functions • Instruction set • Explanation of each instruction

How to Read This Manual

It is assumed that the readers of this manual have general knowledge of electrical engineering, logic circuits, and microcontrollers.

- To gain a general understanding of functions:
  - → Read this manual in the order of the **CONTENTS**. The mark “<R>” shows major revised points. The revised points can be easily searched by copying an “<R>” in the PDF file and specifying it in the “Find what:” field.
- How to interpret the register format:
  - → For a bit number enclosed in angle brackets, the bit name is defined as a reserved word in the assembler, and is defined as an sfr variable using the #pragma sfr directive in the compiler.
- To know details of the RL78/G21 Microcontroller instructions:
  - → Refer to the separate document **RL78 Family User’s Manual: Software (R01US0015E).**

Conventions

Data significance: Higher digits on the left and lower digits on the right

Active low representations: $\overline{\times\times\times}$ (overscore over pin and signal name)

Note: Footnote for item marked with Note in the text

Caution: Information requiring particular attention

Remark: Supplementary information

Numerical representations: Binary numbers are represented as xxxx or xxxxB, where each x is 0 or 1.

Decimal numbers are represented as xxxx, where each x is a numeral from 0 to 9.

Hexadecimal numbers are represented as xxxxH, where each x is a number from 0 to 9 or a letter from A to F.

Related Documents

Documents Related to Devices

Document Name	Document No.
RL78/G21 User's Manual: Hardware	This manual
RL78 Family User's Manual: Software	R01US0015E

Documents Related to Flash Memory Programming and On-chip Debugging

Document Name	Document No.
PG-FP6 Flash Memory Programmer User's Manual	**Note 1**
E2 Emulator User's Manual	R20UT3538E
E2 Emulator Lite RTE0T0002LKCE00000R User's Manual	R20UT3240E
Renesas Flash Programmer Flash Memory Programming Software User's Manual	**Note 2**

Note 1. For a list of the documents relevant to the PG-FP6, visit the Web page below.
https://www.renesas.com/us/en/software-tool/pg-fp6

Note 2. For a list of the documents relevant to the Renesas Flash Programmer, visit the Web page below.
https://www.renesas.com/us/en/software-tool/renesas-flash-programmer-programming-gui

Caution The related documents listed above are subject to change without notice. Be sure to use the latest version of each document when designing.

Other Documents

Document Name	Document No.
Renesas Microcontrollers RL78 Family	R01CP0003E
Semiconductor Package Mount Manual	R50ZZ0003E
Semiconductor Reliability Handbook	R51ZZ0001E

Caution The related documents listed above are subject to change without notice. Be sure to use the latest version of each document when designing.

All trademarks and registered trademarks are the property of their respective owners.

EEPROM is a trademark of Renesas Electronics Corporation.

SuperFlash is a registered trademark of Silicon Storage Technology, Inc. in several countries including the United States and Japan.

Caution: This product uses SuperFlash® technology licensed from Silicon Storage Technology, Inc.

CONTENTS

1. Outline 1
1.1 Features 1
1.2 List of Part Numbers 4
1.3 Pin Configuration (Top View) 6
1.3.1 20-pin products 6
1.3.2 32-pin products 8
1.3.3 44-pin products 10
1.3.4 48-pin products 13
1.3.5 64-pin products 16
1.4 Pin Identification 19
1.5 Block Diagram 20
1.6 Outline of Functions 21

2. Pin Functions 24
2.1 Functions of Port Pins 24
2.1.1 20-pin products 25
2.1.2 32-pin products 26
2.1.3 44-pin products 28
2.1.4 48-pin products 30
2.1.5 64-pin products 32
2.2 Pin Functions Other than Port Pin Functions 34
2.2.1 Functions for each product 34
2.2.2 Description of pin functions 37
2.3 Connection of Unused Pins 39
2.4 Block Diagrams of Pins 40

3. CPU Architecture 54
3.1 Memory Space 55
3.1.1 Internal program memory space 58
3.1.2 Mirror area 62
3.1.3 Internal data memory space 64
3.1.4 Special function register (SFR) area 65
3.1.5 Extended special function register (2nd SFR: 2nd special function register) area 65
3.1.6 Data memory addressing 66
3.2 Processor Registers 67
3.2.1 Control registers 67
3.2.2 General-purpose registers 69
3.2.3 ES and CS registers 70
3.2.4 Special function registers (SFRs) 71
3.2.5 Extended special function registers (2nd SFRs) 75
3.3 Instruction Address Addressing 82
3.3.1 Relative addressing 82
3.3.2 Immediate addressing 82
3.3.3 Table indirect addressing 83
3.3.4 Register indirect addressing 84

3.4 Types of Addressing for Processing Data Addresses 85
3.4.1 Implied addressing 85
3.4.2 Register addressing 86
3.4.3 Direct addressing 87
3.4.4 Short direct addressing 88
3.4.5 SFR addressing 89
3.4.6 Register indirect addressing 90
3.4.7 Based addressing 91
3.4.8 Based indexed addressing 94
3.4.9 Stack addressing 95

4. Port Functions 98
4.1 Port Functions 98
4.2 Port Configuration 98
4.2.1 Port 0 99
4.2.2 Port 1 99
4.2.3 Port 2 100
4.2.4 Port 3 100
4.2.5 Port 4 100
4.2.6 Port 5 101
4.2.7 Port 6 101
4.2.8 Port 7 101
4.2.9 Port 12 102
4.2.10 Port 13 102
4.2.11 Port 14 103
4.3 Registers for Controlling the Port Function 104
4.3.1 Port mode registers (PMxx) 108
4.3.2 Port registers (Pxx) 109
4.3.3 Pull-up resistor option registers (PUxx) 110
4.3.4 Port input mode registers (PIMxx) 111
4.3.5 Port output mode registers (POMxx) 112
4.3.6 Port digital input disable registers (PDIDISxx) 113
4.3.7 Port mode control A registers (PMCAxx) 114
4.3.8 Peripheral I/O redirection registers (PIORx) 115
4.3.9 Global digital input disable register (GDIDIS) 117
4.3.10 Port mode select register (PMS) 117
4.4 Port Function Operations 118
4.4.1 Writing to I/O port 118
4.4.2 Reading from I/O port 118
4.4.3 Operations on I/O port 118
4.4.4 Communications with devices operating at a different voltage (1.8 V, 2.5 V, or 3 V) by switching the EV_{DD} voltage 118
4.4.5 Communications with devices operating at a different voltage (1.8 V, 2.5 V, or 3 V) by switching I/O buffers 119
4.5 Register Settings When Using Alternate Function 122
4.5.1 Basic concept when using alternate function 122
4.5.2 Register settings for alternate function whose output function is not used 124
4.5.3 Register settings and port pin state 125
4.5.4 Examples of register settings for port and alternate functions 125

4.6 Cautions When Using Port Function 140
4.6.1 Cautions on 1-bit memory manipulation instruction for port register (Pxx) 140
4.6.2 Notes on specifying the pin settings 141

5. Operation State Control 142
5.1 Configuration of Operation State Control 142
5.2 Registers for Controlling the Operation State Control 144
5.2.1 Flash operating mode select register (FLMODE) 144
5.2.2 Flash operating mode protect register (FLMWRP) 146
5.3 Initial Setting of Operating Modes of the Flash Memory 147
5.4 Transitions between Operating Modes of the Flash Memory 148
5.5 Details of Operating Modes of the Flash Memory 149
5.5.1 Details of HS (high-speed main) mode 149
5.5.2 Details of LS (low-speed main) mode 150
5.5.3 Details of LP (low-power main) mode 151
5.5.4 Details on SUB mode 152

6. Clock Generator 153
6.1 Functions of Clock Generator 153
6.2 Configuration of Clock Generator 155
6.3 Registers for Controlling the Clock Generator 157
6.3.1 Clock operation mode control register (CMC) 158
6.3.2 System clock control register (CKC) 159
6.3.3 Clock operation status control register (CSC) 160
6.3.4 Oscillation stabilization time counter status register (OSTC) 161
6.3.5 Oscillation stabilization time select register (OSTS) 163
6.3.6 Peripheral enable registers 0, 1 (PER0, PER1) 165
6.3.7 Subsystem clock supply mode control register (OSMC) 168
6.3.8 Subsystem clock select register (CKSEL) 169
6.3.9 High-speed on-chip oscillator frequency select register (HOCODIV) 170
6.3.10 High-speed system clock division register (MOSCDIV) 171
6.3.11 High-speed on-chip oscillator trimming register (HIOTRM) 172
6.4 System Clock Oscillator 173
6.4.1 X1 oscillator 173
6.4.2 High-speed on-chip oscillator 176
6.4.3 Low-speed on-chip oscillator 176
6.5 Operations of the Clock Generator 177
6.6 Controlling Clocks 179
6.6.1 Example of setting the high-speed on-chip oscillator 179
6.6.2 Example of setting the X1 oscillator clock 180
6.6.3 State transitions of the CPU clock 181
6.6.4 Conditions before changing the CPU clock and processing after changing the CPU clock 185
6.6.5 Time required for switchover of the CPU clock and main system clock 186
6.6.6 Conditions before clock oscillation is stopped 187
6.7 Resonator and Oscillator Constants 188

7. Timer Array Unit (TAU) 189
7.1 Functions of Timer Array Unit 191

7.1.1 Independent channel operation function ... 191
7.1.2 Simultaneous channel operation function ... 193
7.1.3 8-bit timer function (only for channels 1 and 3) ... 194
7.1.4 LIN-bus function (only for channel 7) ... 194
7.2 Configuration of Timer Array Unit ... 195
7.2.1 Timer counter register mn (TCRmn) ... 200
7.2.2 Timer data register mn (TDRmn) ... 202
7.3 Registers for Controlling the Timer Array Unit ... 203
7.3.1 Peripheral enable register 0 (PER0) ... 204
7.3.2 Peripheral reset control register 0 (PRR0) ... 205
7.3.3 Timer clock select register m (TPSm) ... 206
7.3.4 Timer mode register mn (TMRmn) ... 210
7.3.5 Timer status register mn (TSRmn) ... 214
7.3.6 Timer channel enable status register m (TEm) ... 215
7.3.7 Timer channel start register m (TSm) ... 216
7.3.8 Timer channel stop register m (TTm) ... 217
7.3.9 Timer input select register 0 (TIS0) ... 218
7.3.10 Timer output enable register m (TOEm) ... 219
7.3.11 Timer output register m (TOm) ... 220
7.3.12 Timer output level register m (TOLm) ... 221
7.3.13 Timer output mode register m (TOMm) ... 222
7.3.14 Input switch control register (ISC) ... 223
7.3.15 Noise filter enable register 1 (NFEN1) ... 224
7.3.16 Registers for controlling the port functions multiplexed with the inputs and outputs of timers ... 226
7.4 Basic Specifications of the Timer Array Unit ... 227
7.4.1 Basic specifications of the simultaneous channel operation function ... 227
7.4.2 Basic specifications of the 8-bit timer function (only for channels 1 and 3) ... 229
7.5 Operations of Counters ... 230
7.5.1 Counter clock (f_{TCLK}) ... 230
7.5.2 Timing of the start of counting ... 232
7.5.3 Operations of counters ... 233
7.6 Channel Output (TOmn Pin) Control ... 238
7.6.1 TOmn pin output circuit configuration ... 238
7.6.2 TOmn pin output setting ... 239
7.6.3 Cautions on channel output operation ... 240
7.6.4 Collective manipulation of TOmn bit ... 245
7.6.5 Timer interrupts and TOmn outputs when counting is started ... 246
7.7 Timer Input (TImn) Control ... 247
7.7.1 TImn input circuit configuration ... 247
7.7.2 Noise filter ... 247
7.7.3 Cautions on channel input operation ... 248
7.8 Independent Channel Operation Function of Timer Array Unit ... 249
7.8.1 Operation as an interval timer or for square wave output ... 249
7.8.2 Operation as an external event counter ... 253
7.8.3 Operation as a frequency divider (only for channel 0) ... 257
7.8.4 Operation for input pulse interval measurement ... 262
7.8.5 Operation for input signal low- and high-level width measurement ... 267
7.8.6 Operation as a delay counter ... 271

7.9 Simultaneous Channel Operation Function of Timer Array Unit 275
7.9.1 Operation for the one-shot pulse output function 275
7.9.2 Operation for the PWM function 282
7.9.3 Operation for the multiple PWM output function 289
7.10 Cautions When Using Timer Array Unit 297
7.10.1 Cautions when using timer output 297

8. 32-bit Interval Timer (TML32) 298
8.1 Overview 298
8.2 Registers for Controlling the 32-bit Interval Timer 301
8.2.1 Peripheral enable register 1 (PER1) 302
8.2.2 Peripheral reset control register 1 (PRR1) 303
8.2.3 Interval timer compare registers 0mn (ITLCMP0mn) 304
8.2.4 Interval timer compare registers 0n (ITLCMP0n) 304
8.2.5 Interval timer capture register 00 (ITLCAP00) 305
8.2.6 Interval timer control register (ITLCTL0) 306
8.2.7 Interval timer clock select register 0 (ITLCSEL0) 308
8.2.8 Interval timer frequency division register 0 (ITLFDIV00) 309
8.2.9 Interval timer frequency division register 1 (ITLFDIV01) 310
8.2.10 Interval timer capture control register 0 (ITLCC0) 311
8.2.11 Interval timer status register (ITLS0) 313
8.2.12 Interval timer match detection mask register (ITLMKF0) 315
8.3 Operation 316
8.3.1 Counter mode settings 316
8.3.2 Capture mode settings 319
8.3.3 Timer operation 320
8.3.4 Capture operation 321
8.3.5 Interrupt 323
8.3.6 Interval timer setting procedures 325

9. Watchdog Timer (WDT) 328
9.1 Functions of Watchdog Timer 328
9.2 Configuration of Watchdog Timer 329
9.3 Register for Controlling the Watchdog Timer 330
9.3.1 Watchdog timer enable register (WDTE) 330
9.4 Operation of Watchdog Timer 331
9.4.1 Controlling operation of watchdog timer 331
9.4.2 Setting overflow time of watchdog timer 332
9.4.3 Setting window open period of watchdog timer 333
9.4.4 Setting watchdog timer interval interrupt 334

10. A/D Converter (ADC) 335
10.1 Function of A/D Converter 335
10.2 Configuration of A/D Converter 338
10.3 Registers for Controlling the A/D Converter 340
10.3.1 Peripheral enable register 0 (PER0) 341
10.3.2 Peripheral reset control register 0 (PRR0) 342
10.3.3 A/D converter mode register 0 (ADM0) 343
10.3.4 A/D converter mode register 1 (ADM1) 351

10.3.5 A/D converter mode register 2 (ADM2) 352
10.3.6 10-bit A/D conversion result register (ADCR) 355
10.3.7 8-bit A/D conversion result register (ADCRH) 356
10.3.8 Analog input channel specification register (ADS) 357
10.3.9 Conversion result comparison upper limit setting register (ADUL) 359
10.3.10 Conversion result comparison lower limit setting register (ADLL) 359
10.3.11 A/D test register (ADTES) 360
10.3.12 Registers for controlling the port functions multiplexed with the analog inputs of the A/D converter 360
10.4 A/D Converter Conversion Operations 361
10.5 Input Voltage and Conversion Results 363
10.6 Operating Modes of the A/D Converter 364
10.6.1 Software trigger mode: Select mode and sequential conversion mode 364
10.6.2 Software trigger mode: Select mode and one-shot conversion mode 365
10.6.3 Software trigger mode: Scan mode and sequential conversion mode 366
10.6.4 Software trigger mode: Scan mode and one-shot conversion mode 367
10.6.5 Hardware trigger no-wait mode: Select mode and sequential conversion mode 368
10.6.6 Hardware trigger no-wait mode: Select mode and one-shot conversion mode 369
10.6.7 Hardware trigger no-wait mode: Scan mode and sequential conversion mode 370
10.6.8 Hardware trigger no-wait mode: Scan mode and one-shot conversion mode 371
10.6.9 Hardware trigger wait mode: Select mode and sequential conversion mode 372
10.6.10 Hardware trigger wait mode: Select mode and one-shot conversion mode 373
10.6.11 Hardware trigger wait mode: Scan mode and sequential conversion mode 374
10.6.12 Hardware trigger wait mode: Scan mode and one-shot conversion mode 375
10.7 Flows of Setting the A/D Converter Modes 376
10.7.1 Settings in software trigger mode 376
10.7.2 Settings in hardware trigger no-wait mode 377
10.7.3 Settings in hardware trigger wait mode 378
10.7.4 Example of settings when temperature sensor output voltage or internal reference voltage is selected in the software trigger mode and one-shot conversion mode 379
10.7.5 Settings in test mode 380
10.8 SNOOZE Mode Function 381
10.9 How to Read A/D Converter Characteristics Table 385
10.10 Points for Caution When the A/D Converter Is to be Used 389

11. Serial Array Unit (SAU) 393
11.1 Functions of Serial Array Unit 395
11.1.1 Simplified SPI (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) 395
11.1.2 UART (UART0 to UART2) 396
11.1.3 Simplified I^2C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) 397
11.2 Configuration of Serial Array Unit 398
11.2.1 Shift register 401
11.2.2 Lower 8 or 9 bits of the serial data register mn (SDRmn) 401
11.3 Registers for Controlling the Serial Array Unit 403
11.3.1 Peripheral enable register 0 (PER0) 404
11.3.2 Peripheral reset control register 0 (PRR0) 405
11.3.3 Serial clock select register m (SPSm) 406
11.3.4 Serial mode register mn (SMRmn) 407
11.3.5 Serial communication operation setting register mn (SCRmn) 409

11.3.6 Serial data register mn (SDRmn) 411
11.3.7 Serial flag clear trigger register mn (SIRmn) 413
11.3.8 Serial status register mn (SSRmn) 414
11.3.9 Serial channel start register m (SSm) 416
11.3.10 Serial channel stop register m (STm) 417
11.3.11 Serial channel enable status register m (SEm) 418
11.3.12 Serial output enable register m (SOEm) 419
11.3.13 Serial output register m (SOm) 420
11.3.14 Serial output level register m (SOLm) 421
11.3.15 Serial standby control register m (SSCm) 423
11.3.16 Input switch control register (ISC) 425
11.3.17 Noise filter enable register 0 (NFEN0) 427
11.3.18 Registers for controlling the port functions multiplexed with the inputs and outputs of the serial array unit 428
11.3.19 UART loopback select register (ULBS) 429
11.4 Operation Stop Mode 430
11.4.1 Stopping the operation by units 430
11.4.2 Stopping the operation by channels 431
11.5 Operation of Simplified SPI (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) Communication 432
11.5.1 Master transmission 435
11.5.2 Master reception 444
11.5.3 Master transmission/reception 454
11.5.4 Slave transmission 464
11.5.5 Slave reception 473
11.5.6 Slave transmission/reception 481
11.5.7 SNOOZE mode 491
11.5.8 Calculating transfer clock frequency 496
11.5.9 Procedure for processing errors that occurred during simplified SPI (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) communication 498
11.6 Operation of UART (UART0 to UART2) Communication 499
11.6.1 UART transmission 502
11.6.2 UART reception 512
11.6.3 SNOOZE mode 519
11.6.4 Calculating baud rate 526
11.6.5 Procedure for processing errors that occurred during UART (UART0 to UART2) communication 530
11.7 LIN Communication Operation 531
11.7.1 LIN transmission 531
11.7.2 LIN reception 534
11.8 Operation of Simplified I^2C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) Communication 541
11.8.1 Address field transmission 544
11.8.2 Data transmission 550
11.8.3 Data reception 553
11.8.4 Stop condition generation 557
11.8.5 Calculating transfer rate 559
11.8.6 Procedure for processing errors that occurred during simplified I^2C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication 562

12. Serial Interface IICA (IICA) ... 563
12.1 Functions of Serial Interface IICA ... 563
12.2 Configuration of Serial Interface IICA ... 566
12.3 Registers for Controlling Serial Interface IICA ... 569
12.3.1 Peripheral enable register 0 (PER0) ... 570
12.3.2 Peripheral reset control register 0 (PRR0) ... 571
12.3.3 IICA control register n0 (IICCTLn0) ... 572
12.3.4 IICA status register n (IICSn) ... 576
12.3.5 IICA flag register n (IICFn) ... 580
12.3.6 IICA control register n1 (IICCTLn1) ... 582
12.3.7 IICA low-level width setting register n (IICWLn) ... 585
12.3.8 IICA high-level width setting register n (IICWHn) ... 585
12.3.9 Registers for controlling the port functions multiplexed with the IICA serial interface inputs and outputs ... 586
12.4 I^2C Bus Mode Functions ... 587
12.4.1 Pin configuration ... 587
12.4.2 Setting transfer clock by using IICWLn and IICWHn registers ... 588
12.5 I^2C Bus Definitions and Control Methods ... 590
12.5.1 Start conditions ... 590
12.5.2 Address ... 591
12.5.3 Transfer direction specification ... 591
12.5.4 Acknowledge (ACK) ... 592
12.5.5 Stop condition ... 593
12.5.6 Clock stretching ... 594
12.5.7 Release from clock stretching ... 596
12.5.8 Timing of generation of the interrupt request signal (INTIICAn) and control of clock stretching ... 597
12.5.9 Address match detection method ... 598
12.5.10 Error detection ... 598
12.5.11 Extension code ... 599
12.5.12 Arbitration ... 600
12.5.13 Wakeup function ... 602
12.5.14 Communication reservation ... 605
12.5.15 Cautions ... 609
12.5.16 Communication operations ... 610
12.5.17 Timing of I^2C interrupt request signal (INTIICAn) occurrence ... 618
12.6 Timing Charts ... 638

13. Serial Interface UARTA (UARTA) ... 653
13.1 Overview ... 653
13.2 Registers for Controlling the Serial Interface UARTA ... 655
13.2.1 Peripheral enable register 1 (PER1) ... 656
13.2.2 Transmit buffer register n (TXBAn) ... 657
13.2.3 Receive buffer register n (RXBAn) ... 658
13.2.4 Operation mode setting register n0 (ASIMAn0) ... 659
13.2.5 Operation mode setting register n1 (ASIMAn1) ... 661
13.2.6 Baud rate generator control register n (BRGCAn) ... 662
13.2.7 Status register n (ASISAn) ... 663
13.2.8 Status clear trigger register n (ASCTAn) ... 665

13.2.9 UARTA clock select register 0 (UTA0CK) 666
13.2.10 UART loopback select register (ULBS) 667
13.2.11 Registers for controlling the port functions multiplexed with the inputs and outputs of the UARTA serial interface 668
13.3 Operation 669
13.3.1 Operation stop mode 669
13.3.2 UART mode 669
13.3.3 Receive data noise filter 683
13.3.4 Baud rate generator 684
13.4 Points for Caution when the Serial Interface UARTA is to be Used 690
13.4.1 Port setting for the RxDAn pin 690

14. Interrupt Functions 691
14.1 Interrupt Function Types 691
14.2 Interrupt Sources and Configuration 691
14.3 Registers for Controlling the Interrupt Functions 697
14.3.1 Interrupt request flag registers (IF0L, IF0H, IF1L, IF1H, IF2L, IF2H, IF3L) 701
14.3.2 Interrupt mask flag registers (MK0L, MK0H, MK1L, MK1H, MK2L, MK2H, MK3L) 703
14.3.3 Priority specification flag registers (PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR03L, PR10L, PR10H, PR11L, PR11H, PR12L, PR12H, PR13L) 705
14.3.4 External interrupt rising edge enable registers (EGP0, EGP1), External interrupt falling edge enable registers (EGN0, EGN1) 708
14.3.5 Program status word (PSW) 710
14.3.6 Registers for controlling the port functions multiplexed with the interrupt inputs 711
14.4 Interrupt Servicing Operations 712
14.4.1 Maskable interrupt request acknowledgment 712
14.4.2 Software interrupt request acknowledgment 715
14.4.3 Multiple interrupt servicing 715
14.4.4 Interrupt request held pending 719

15. Key Interrupt Function 720
15.1 Functions of the Key Interrupt 720
15.2 Configuration of the Key Interrupt 720
15.3 Registers for Controlling the Key Interrupt 722
15.3.1 Key return control register (KRCTL) 722
15.3.2 Key return mode register 0 (KRM0) 723
15.3.3 Key return flag register (KRF) 724
15.3.4 Registers for controlling the port functions multiplexed with the key interrupt inputs 725

16. Standby Function 726
16.1 Standby Function 726
16.2 Registers for Controlling the Standby Function 727
16.3 Standby Operation 728
16.3.1 HALT mode 728
16.3.2 STOP mode 734
16.3.3 SNOOZE mode 739

17. Reset Function 742
17.1 Timing of Reset Operation 744

17.2 Registers for Controlling the Reset Function 748
17.2.1 Reset control flag register (RESF) 748
17.2.2 Power-on-reset status register (PORSR) 751
17.2.3 Peripheral reset control register 0 (PRR0) 752
17.2.4 Peripheral reset control register 1 (PRR1) 753

18. Power-on-reset Circuit (POR) 754
18.1 Functions of Power-on-reset Circuit 754
18.2 Configuration of Power-on-reset Circuit 755
18.3 Operation of Power-on-reset Circuit 756

19. Voltage Detector (LVD) 758
19.1 Functions of Voltage Detector 758
19.2 Configuration of Voltage Detector 759
19.3 Registers for Controlling the Voltage Detector 760
19.3.1 Voltage detection register (LVIM) 760
19.3.2 LVD detection flag clearing register (LVDFCLR) 761
19.3.3 Voltage detection level register (LVIS) 762
19.4 Operation of Voltage Detector 764
19.4.1 Setting for use in the reset mode 764
19.4.2 Setting for use in the interrupt mode 767
19.5 Points for Caution when the Voltage Detector Is to be Used 770

20. Safety Functions 773
20.1 Overview of Safety Functions 773
20.2 Registers for Controlling the Safety Functions 774
20.3 Operation of Safety Functions 774
20.3.1 Flash memory CRC operation (high-speed CRC) 774
20.3.1.1 Flash memory CRC control register (CRC0CTL) 775
20.3.1.2 Flash memory CRC operation result register (PGCRCL) 776
20.3.2 CRC operation (general-purpose CRC) 778
20.3.2.1 CRC input register (CRCIN) 778
20.3.2.2 CRC data register (CRCD) 779
20.3.3 Flash memory guard function 780
20.3.3.1 Code flash memory guard register (GFLASH0) 780
20.3.3.2 Data flash memory guard register (GFLASH1) 781
20.3.3.3 Flash security area guard register (GFLASH2) 782
20.3.4 RAM parity error detection 783
20.3.4.1 RAM parity error control register (RPECTL) 783
20.3.5 RAM guard function 785
20.3.5.1 Invalid memory access detection control register (IAWCTL) 785
20.3.6 SFR guard function 786
20.3.6.1 Invalid memory access detection control register (IAWCTL) 786
20.3.7 Illicit memory access detection 787
20.3.7.1 Invalid memory access detection control register (IAWCTL) 788
20.3.8 Guard function of invalid memory access detection control register 789
20.3.8.1 Guard register of IAWCTL register (GIAWCTL) 789
20.3.9 Frequency detection 790
20.3.9.1 Timer input select register 0 (TIS0) 791

20.3.10 Testing of the A/D converter ... 792
20.3.10.1 A/D test register (ADTES) ... 794
20.3.10.2 Analog input channel specification register (ADS) ... 795
20.3.11 Detection of the digital output signal level of the I/O pins ... 797
20.3.11.1 Port mode select register (PMS) ... 797
20.3.12 UART loopback ... 798
20.3.12.1 UART loopback select register (ULBS) ... 799

21. Security Functions ... 800
21.1 Flash Read Protection ... 800
21.1.1 Function of flash read protection ... 800
21.1.2 Setting of flash read protection ... 800
21.1.3 Operation ... 802
21.2 Unique ID ... 803
21.2.1 Function of a unique ID ... 803
21.2.2 ASCII codes representing the product name ... 803

22. Regulator ... 804
22.1 Overview ... 804

23. Option Bytes ... 805
23.1 Functions of Option Bytes ... 805
23.1.1 User option bytes (000C0H to 000C2H or 040C0H to 040C2H) ... 805
23.1.2 On-chip debug option byte (000C3H or 040C3H) ... 806
23.2 Format of User Option Bytes ... 807
23.3 Format of On-chip Debug Option Byte ... 811
23.4 Setting of Option Bytes ... 812

24. Flash Memory ... 813
24.1 Serial Programming Using Flash Memory Programmer ... 814
24.1.1 Programming environment ... 816
24.1.2 Communications mode ... 816
24.2 Serial Programming Using External Device (that Incorporates UART) ... 817
24.2.1 Programming environment ... 817
24.2.2 Communications mode ... 818
24.3 Handling of Pins on the Board ... 819
24.3.1 P40/TOOL0 pin ... 819
24.3.2 $\overline{\text{RESET}}$ pin ... 820
24.3.3 Port pins ... 821
24.3.4 REGC pin ... 821
24.3.5 Power supply ... 821
24.4 Programming Method ... 822
24.4.1 Serial programming procedure ... 822
24.4.2 Flash memory programming mode ... 823
24.4.3 Selecting communications mode ... 824
24.4.4 Communications commands ... 825
24.5 Processing Times for Commands When the Dedicated Flash Memory Programmer Is in Use (Reference Values) ... 827
24.6 Self-programming ... 828

24.6.1 Self-programming procedure ... 829
24.6.2 Registers for controlling the flash memory ... 830
24.6.2.1 Flash address pointer registers H and L (FLAPH, FLAPL) ... 831
24.6.2.2 Flash end address pointer registers H and L (FLSEDH, FLSEDL) ... 832
24.6.2.3 Flash write buffer registers H and L (FLWH, FLWL) ... 834
24.6.2.4 Flash protect command register (PFCMD) ... 835
24.6.2.5 Flash status register (PFS) ... 836
24.6.2.6 Flash programming mode control register (FLPMC) ... 837
24.6.2.7 Flash area selection register (FLARS) ... 838
24.6.2.8 Flash memory sequencer initial setting register (FSSET) ... 839
24.6.2.9 Flash memory sequencer control register (FSSQ) ... 841
24.6.2.10 Flash extra area sequencer control register (FSSE) ... 843
24.6.2.11 Flash registers initialization register (FLRST) ... 845
24.6.2.12 Flash memory sequencer status registers H and L (FSASTH, FSASTL) ... 846
24.6.2.13 Flash security flag monitoring register (FLSEC) ... 848
24.6.2.14 Flash FSW monitoring register E (FLFSWE) ... 849
24.6.2.15 Flash FSW monitoring register S (FLFSWS) ... 850
24.6.2.16 Data flash control register (DFLCTL) ... 851
24.6.2.17 Interrupt vector jump enable register (VECTCTRL) ... 852
24.6.2.18 Interrupt vector change registers 0 and 1 (FLSIVC0, FLSIVC1) ... 853
24.6.3 Setting the flash memory control mode ... 854
24.6.4 Initializing the registers for use with the flash memory sequencer ... 856
24.6.5 Setting the operating frequency of the flash memory sequencer ... 856
24.6.6 Rewriting the flash memory ... 857
24.6.7 Interrupts in code flash memory programming mode ... 868
24.6.8 Example of executing the commands to rewrite the flash memory areas ... 870
24.6.9 Notes on self-programming ... 873
24.7 Boot Swap Function ... 874
24.8 Flash Shield Window Function ... 876
24.9 Security Settings ... 877
24.10 Data Flash Memory ... 880
24.10.1 Overview of the data flash memory ... 880
24.10.2 Procedure for accessing the data flash memory ... 881

25. On-chip Debugging ... 882
25.1 Connection between the E2 or E2 Lite On-chip Debugging Emulator and This MCU ... 882
25.2 Connection between the External Device that Incorporates UART and This MCU ... 884
25.3 Security Settings for On-chip Debugging ... 884
25.4 Allocation of Memory Spaces to User Resources ... 885

26. BCD Correction Circuit ... 887
26.1 BCD Correction Circuit Function ... 887
26.2 Register for Controlling the BCD Correction Circuit ... 887
26.2.1 BCD correction result register (BCDADJ) ... 887
26.3 BCD Correction Circuit Operation ... 888

27. Instruction Set ... 890
27.1 Conventions Used in Operation List ... 891
27.1.1 Operand identifiers and description methods ... 891

27.1.2 Description of operation column 892
27.1.3 Description of flag operation column 893
27.1.4 PREFIX instruction 893
27.2 Operation List 894

28. Electrical Characteristics 912
28.1 Absolute Maximum Ratings 913
28.2 Characteristics of the Oscillators 915
28.2.1 Characteristics of the X1 oscillator 915
28.2.2 Characteristics of the on-chip oscillators 915
28.3 DC Characteristics 916
28.3.1 Characteristics of pins 916
28.3.2 Characteristics of the supply current 923
28.4 AC Characteristics 929
28.5 Characteristics of the Peripheral Functions 934
28.5.1 Serial array unit 934
28.5.2 Serial interface UARTA 963
28.5.3 Serial interface IICA 964
28.6 Characteristics of the Analog Circuits 967
28.6.1 Characteristics of the A/D converter for T_A = –40 to 85°C 967
28.6.2 Characteristics of the A/D converter for T_A = –40 to 105°C 972
28.6.3 Characteristics of the temperature sensor and internal reference voltage 976
28.6.4 Characteristics of the POR circuit 976
28.6.5 Characteristics of the LVD circuit 977
28.6.6 Characteristics of the rising slope of the power supply voltage 979
28.7 Characteristics of Retention of RAM Data 980
28.8 Characteristics of Flash Memory Programming 980
28.9 Dedicated Flash Memory Programmer Communication (UART) 982
28.10 Timing of Entry to Flash Memory Programming Modes 982

29. Package Drawings 983
29.1 20-pin Products 983
29.2 32-pin Products 984
29.3 44-pin Products 985
29.4 48-pin Products 986
29.5 64-pin Products 987

APPENDIX A Revision History 989
A.1 Major Revisions in This Edition 989
A.2 Major Revisions in This Edition 991

Section 1 Outline

1.1 Features

Ultra-low power consumption technology

- Operation from a single power supply (V_{DD}) within the 1.6-V to 5.5-V range
- HALT mode
- STOP mode
- SNOOZE mode

RL78 CPU core

- CISC architecture with 3-stage pipeline
- The minimum instruction execution time can be changed from high to ultra-low speed.
  - High speed: 0.03125 µs at 32-MHz operation with the high-speed on-chip oscillator clock
  - Ultra-low speed: 30.5 µs at 32.768-kHz operation with the subsystem clock
- Multiply/divide/multiply & accumulate instructions are supported.
- Address space: 1 MB
- General-purpose registers: (8-bit register × 8) × 4 banks
- On-chip RAM: 8 KB

Code flash memory

- Code flash memory: 128 or 256 KB
- Block size: 2 KB
- Security function: Prohibition of block erase and rewriting
- On-chip debugging
- Self-programming with boot swapping and flash shield window

Data flash memory

- Data flash memory: 4 KB
- Background operation (BGO): Instructions can be executed from the program memory while rewriting the data flash memory.
- Number of rewrites: 1,000,000 times (typ.)

High-speed on-chip oscillator

- Selectable from among 32 MHz, 24 MHz, 16 MHz, 12 MHz, 8 MHz, 6 MHz, 4 MHz, 3 MHz, 2 MHz, and 1 MHz
- High accuracy: ±1.0% (V_{DD} = 1.6 to 5.5 V, T_A = -40 to +105°C)

Low-speed on-chip oscillator

- 32.768 kHz (typ.)

Operating ambient temperature

- T_A = -40 to +105°C (3Z: Industrial and consumer applications)

Power management and reset function

- Power-on-reset circuit (POR)
- Voltage detectors (LVD0 and LVD1)

Serial interface

- Simplified SPI (CSI[Note 1]): 2 to 6 channels
- UART/UART (LIN-bus supported)/UARTA: 4 channels
- I^2C/Simplified I^2C: 2 to 7 channels

Timers

- 16-bit timer: 8 channels
- 32-bit interval timer: 1 channel in 32-bit counter mode
  2 channels in 16-bit counter mode
  4 channels in 8-bit counter mode
- Watchdog timer: 1 channel (operates with the low-speed on-chip oscillator clock)

A/D converter

- 8-/10-bit resolution
- Analog input: 6 to 12 channels
- Internal reference voltage (1.48 V (typ.)) and temperature sensor

Input/output port pins

- Number of port pins: 17 to 59
  N-ch open-drain I/O (withstand voltage of 6 V): 0 to 2
  N-ch open-drain I/O (withstand voltage of V_{DD}[Note 2]/withstand voltage of EV_{DD}[Note 3]): 5 to 19
- Can be set to N-ch open drain or TTL input buffer, and use of an on-chip pull-up resistor can be specified.
- Connectable to a device with different voltage (1.8, 2.5, or 3 V)

Others

- Key interrupt input
- Binary-coded decimal (BCD) correction circuit

Note 1. Although the CSI function is generally called SPI, it is also called CSI in this product, so it is referred to as such in this manual.

Note 2. This applies to the 20- to 48-pin products.

Note 3. This applies to the 64-pin products.

Remark The functions mounted depend on the product. For details, see **1.6 Outline of Functions**.

O ROM, RAM capacities

Code flash memory	Data flash memory	RAM	RL78/G21				
			20 pins	32 pins	44 pins	48 pins	64 pins
256 KB	4 KB	8 KB	R7F103G6J	R7F103GBJ	R7F103GFJ	R7F103GGJ	R7F103GLJ
128 KB	4 KB	8 KB	R7F103G6G	R7F103GBG	R7F103GFG	R7F103GGG	R7F103GLG

1.2 List of Part Numbers

Figure 1 - 1 Part Number, Memory Size, and Package of RL78/G21

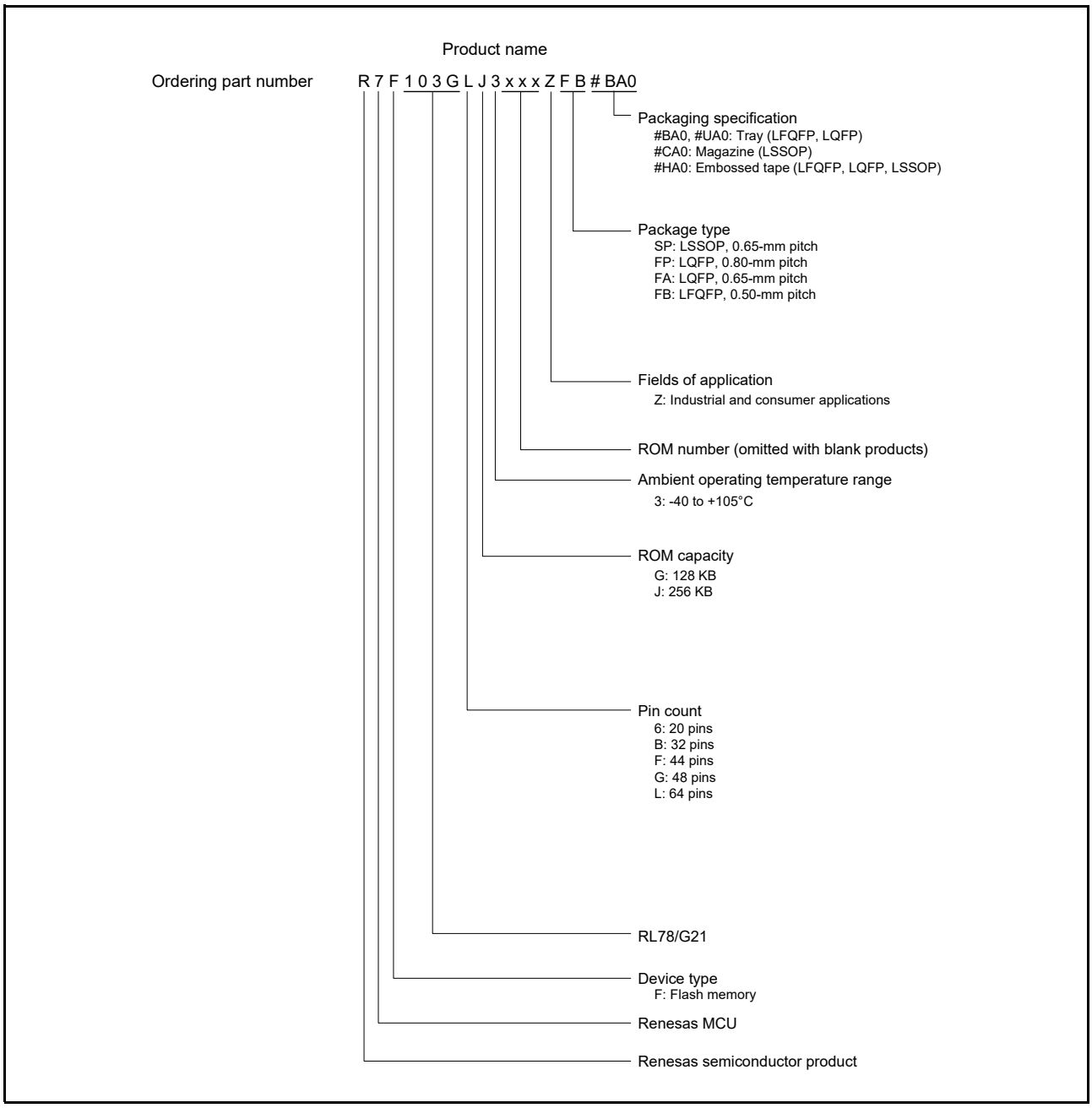


Table 1 - 1 List of Ordering Part Numbers

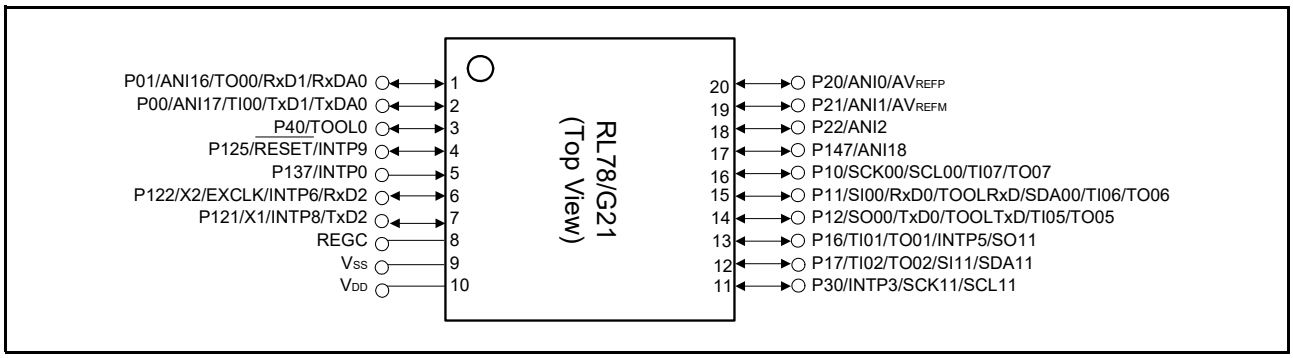
Pin Count	Package	Fields of Application Note	Ordering Part Number		Renesas Code
			Product Name	Packaging Specification	
20	20-pin plastic LSSOP (4.4 × 6.5 mm, 0.65-mm pitch)	Z	R7F103G6J3ZSP, R7F103G6G3ZSP	#CA0, #HA0	PLSP0020JB-A
32	32-pin plastic LQFP (7 × 7 mm, 0.80-mm pitch)	Z	R7F103GBJ3ZFP, R7F103GBG3ZFP	#BA0, #UA0, #HA0	PLQP0032GB-A
44	44-pin plastic LQFP (10 × 10 mm, 0.80-mm pitch)	Z	R7F103GFJ3ZFP, R7F103GFG3ZFP	#BA0, #UA0, #HA0	PLQP0044GF-A
48	48-pin plastic LFQFP (7 × 7 mm, 0.50-mm pitch)	Z	R7F103GGJ3ZFB, R7F103GGG3ZFB	#BA0, #UA0, #HA0	PLQP0048KB-B
64	64-pin plastic LQFP (12 × 12 mm, 0.65-mm pitch)	Z	R7F103GLJ3ZFA, R7F103GLG3ZFA	#BA0, #UA0, #HA0	PLQP0064JB-A
	64-pin plastic LFQFP (10 × 10 mm, 0.50-mm pitch)	Z	R7F103GLJ3ZFB, R7F103GLG3ZFB	#BA0, #UA0, #HA0	PLQP0064KB-C

Note For the fields of application, see **Figure 1 - 1 Part Number, Memory Size, and Package of RL78/G21**.

1.3 Pin Configuration (Top View)

1.3.1 20-pin products

- 20-pin plastic LSSOP (4.4 × 6.5 mm, 0.65-mm pitch)



Caution **Connect the REGC pin to Vss via a capacitor (0.47 to 1 µF).**

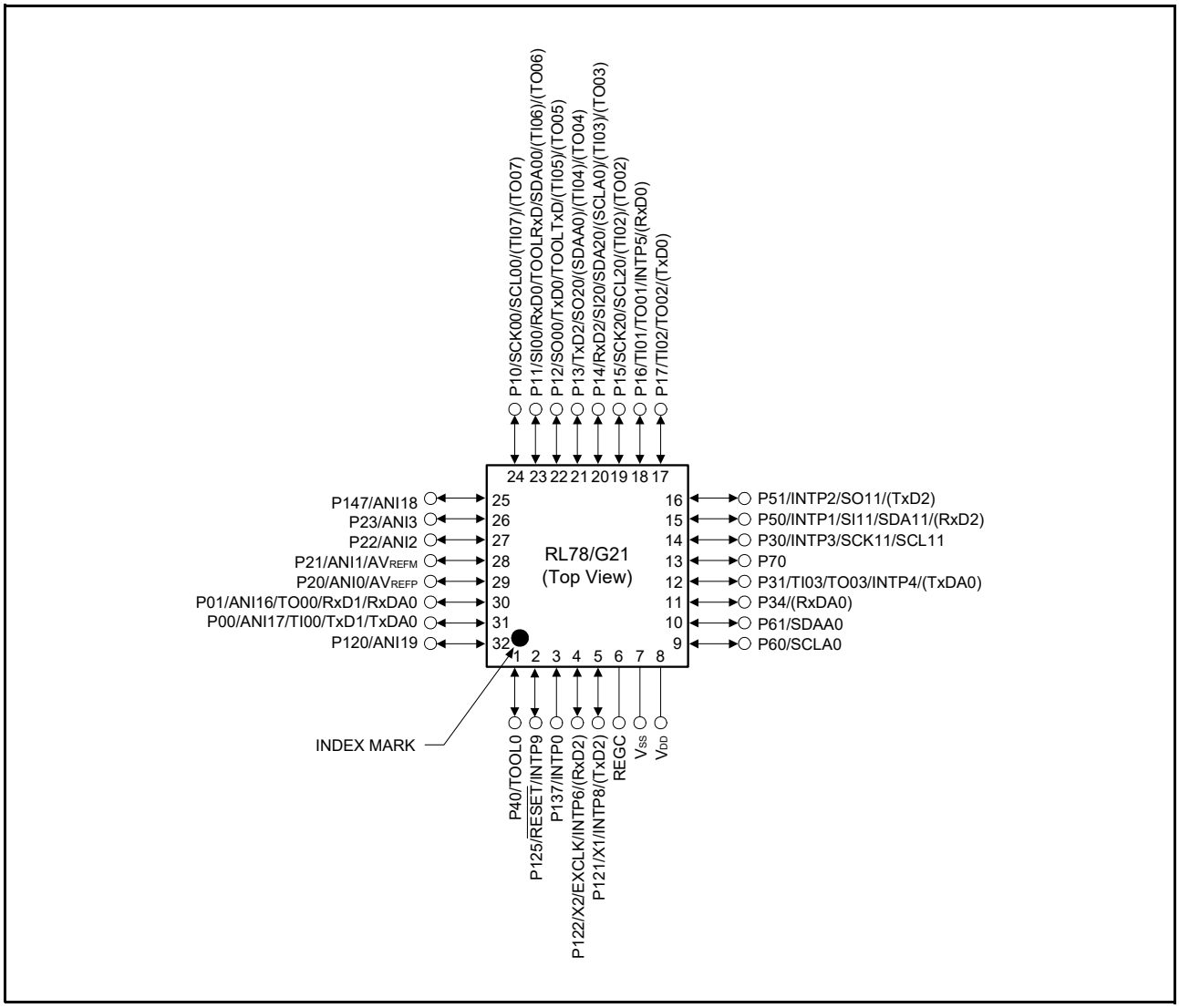
Remark For pin identification, see **1.4 Pin Identification**.

Table 1 - 2 Multiplexed Pin Functions of the 20-pin Products

Pin Number	I/O	Power supply, system clock, and debugging	Analog Circuit	HMI		Timer	Communications Interfaces		
20LSSOP	Digital port		A/D converter (ADC)	Interrupt	Key interrupt	Timer array unit (TAU)	Serial array unit (SAU)	Serial interface IICA (IICA)	Serial interface UARTA (UARTA)
1	P01	—	ANI16	—	—	TO00	RxD1	—	RxDA0
2	P00	—	ANI17	—	—	TI00	TxD1	—	TxDA0
3	P40	TOOL0	—	—	—	—	—	—	—
4	P125	$\overline{\text{RESET}}$	—	INTP9	—	—	—	—	—
5	P137	—	—	INTP0	—	—	—	—	—
6	P122	X2/EXCLK	—	INTP6	—	—	RxD2	—	—
7	P121	X1	—	INTP8	—	—	TxD2	—	—
8	—	REGC	—	—	—	—	—	—	—
9	—	V_{SS}	—	—	—	—	—	—	—
10	—	V_{DD}	—	—	—	—	—	—	—
11	P30	—	—	INTP3	—	—	SCK11/SCL11	—	—
12	P17	—	—	—	—	TI02/TO02	SI11/SDA11	—	—
13	P16	—	—	INTP5	—	TI01/TO01	SO11	—	—
14	P12	TOOLTxD	—	—	—	TI05/TO05	SO00/TxD0	—	—
15	P11	TOOLRxD	—	—	—	TI06/TO06	SI00/RxD0/ SDA00	—	—
16	P10	—	—	—	—	TI07/TO07	SCK00/SCL00	—	—
17	P147	—	ANI18	—	—	—	—	—	—
18	P22	—	ANI2	—	—	—	—	—	—
19	P21	—	ANI1/AV_{REFM}	—	—	—	—	—	—
20	P20	—	ANI0/AV_{REFP}	—	—	—	—	—	—

1.3.2 32-pin products

- 32-pin plastic LQFP (7 × 7 mm, 0.80-mm pitch)



Caution Connect the REGC pin to Vss via a capacitor (0.47 to 1 µF).

Remark 1. For pin identification, see **1.4 Pin Identification**.

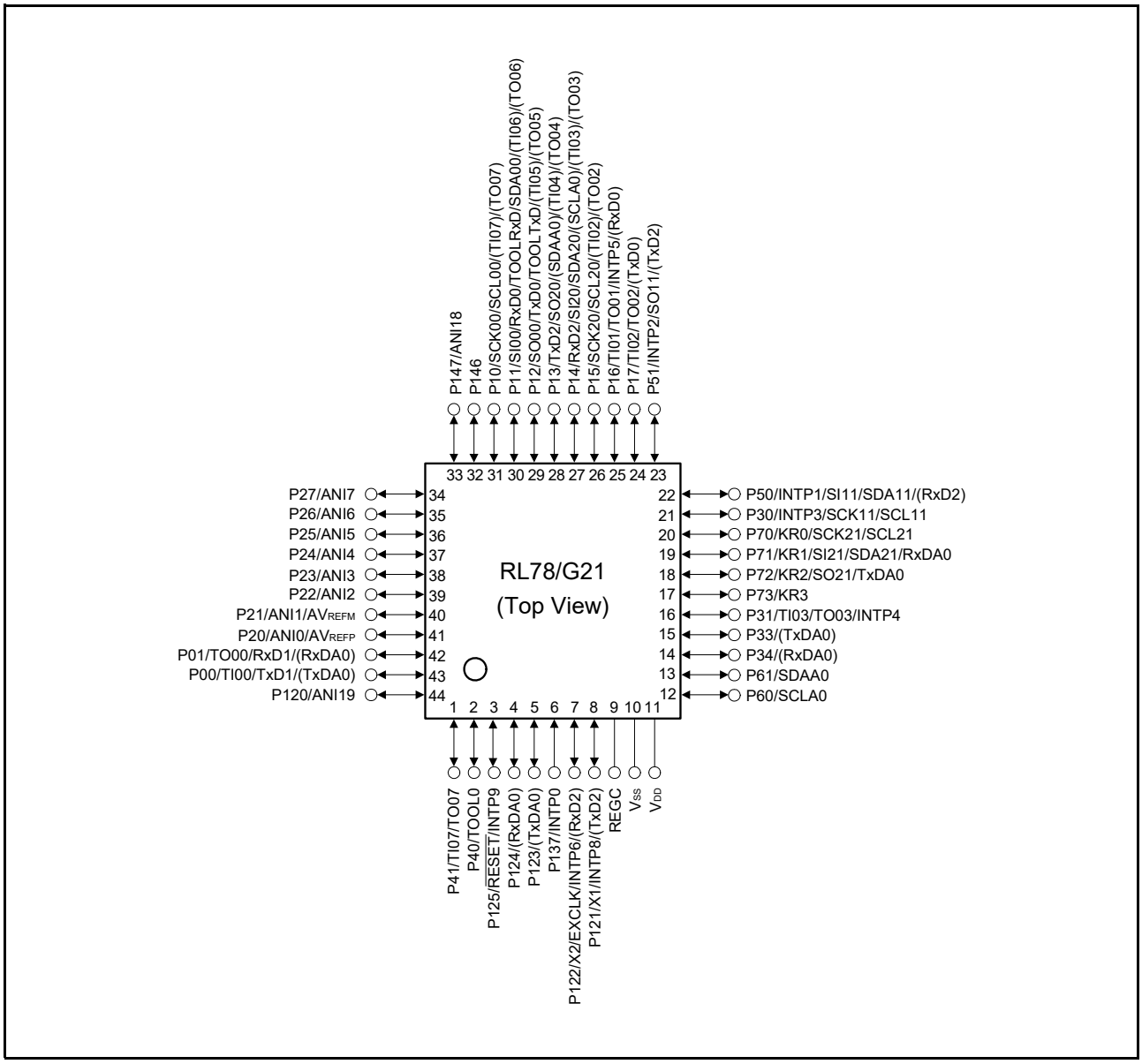
Remark 2. Functions in parentheses in the above figure can be assigned via settings in the corresponding peripheral I/O redirection register (PIORx). For details, see **4.3.8 Peripheral I/O redirection registers (PIORx)**.

Table 1 - 3 Multiplexed Pin Functions of the 32-pin Products

Pin Number	I/O	Power supply, system clock, and debugging	Analog Circuit	HMI		Timer	Communications Interfaces		
32LQFP	Digital port		A/D converter (ADC)	Interrupt	Key interrupt	Timer array unit (TAU)	Serial array unit (SAU)	Serial interface IICA (IICA)	Serial interface UARTA (UARTA)
1	P40	TOOL0	—	—	—	—	—	—	—
2	P125	$\overline{\text{RESET}}$	—	INTP9	—	—	—	—	—
3	P137	—	—	INTP0	—	—	—	—	—
4	P122	X2/EXCLK	—	INTP6	—	—	(RxD2)	—	—
5	P121	X1	—	INTP8	—	—	(TxD2)	—	—
6	—	REGC	—	—	—	—	—	—	—
7	—	V_{SS}	—	—	—	—	—	—	—
8	—	V_{DD}	—	—	—	—	—	—	—
9	P60	—	—	—	—	—	—	SCLA0	—
10	P61	—	—	—	—	—	—	SDAA0	—
11	P34	—	—	—	—	—	—	—	(RxDA0)
12	P31	—	—	INTP4	—	TI03/TO03	—	—	(TxDA0)
13	P70	—	—	—	—	—	—	—	—
14	P30	—	—	INTP3	—	—	SCK11/SCL11	—	—
15	P50	—	—	INTP1	—	—	SI11/SDA11/ (RxD2)	—	—
16	P51	—	—	INTP2	—	—	SO11/(TxD2)	—	—
17	P17	—	—	—	—	TI02/TO02	(TxD0)	—	—
18	P16	—	—	INTP5	—	TI01/TO01	(RxD0)	—	—
19	P15	—	—	—	—	(TI02)/(TO02)	SCK20/SCL20	—	—
20	P14	—	—	—	—	(TI03)/(TO03)	SI20/RxD2/ SDA20	(SCLA0)	—
21	P13	—	—	—	—	(TI04)/(TO04)	SO20/TxD2	(SDAA0)	—
22	P12	TOOLTxD	—	—	—	(TI05)/(TO05)	SO00/TxD0	—	—
23	P11	TOOLRxD	—	—	—	(TI06)/(TO06)	SI00/RxD0/ SDA00	—	—
24	P10	—	—	—	—	(TI07)/(TO07)	SCK00/SCL00	—	—
25	P147	—	ANI18	—	—	—	—	—	—
26	P23	—	ANI3	—	—	—	—	—	—
27	P22	—	ANI2	—	—	—	—	—	—
28	P21	—	ANI1/AV_{REFM}	—	—	—	—	—	—
29	P20	—	ANI0/AV_{REFP}	—	—	—	—	—	—
30	P01	—	ANI16	—	—	TO00	RxD1	—	RxDA0
31	P00	—	ANI17	—	—	TI00	TxD1	—	TxDA0
32	P120	—	ANI19	—	—	—	—	—	—

1.3.3 44-pin products

- 44-pin plastic LQFP (10 × 10 mm, 0.80-mm pitch)



Caution Connect the REGC pin to Vss via a capacitor (0.47 to 1 μF).

Remark 1. For pin identification, see **1.4 Pin Identification**.

Remark 2. Functions in parentheses in the above figure can be assigned via settings in the corresponding peripheral I/O redirection register (PIORx). For details, see **4.3.8 Peripheral I/O redirection registers (PIORx)**.

Table 1 - 4 Multiplexed Pin Functions of the 44-pin Products (1/2)

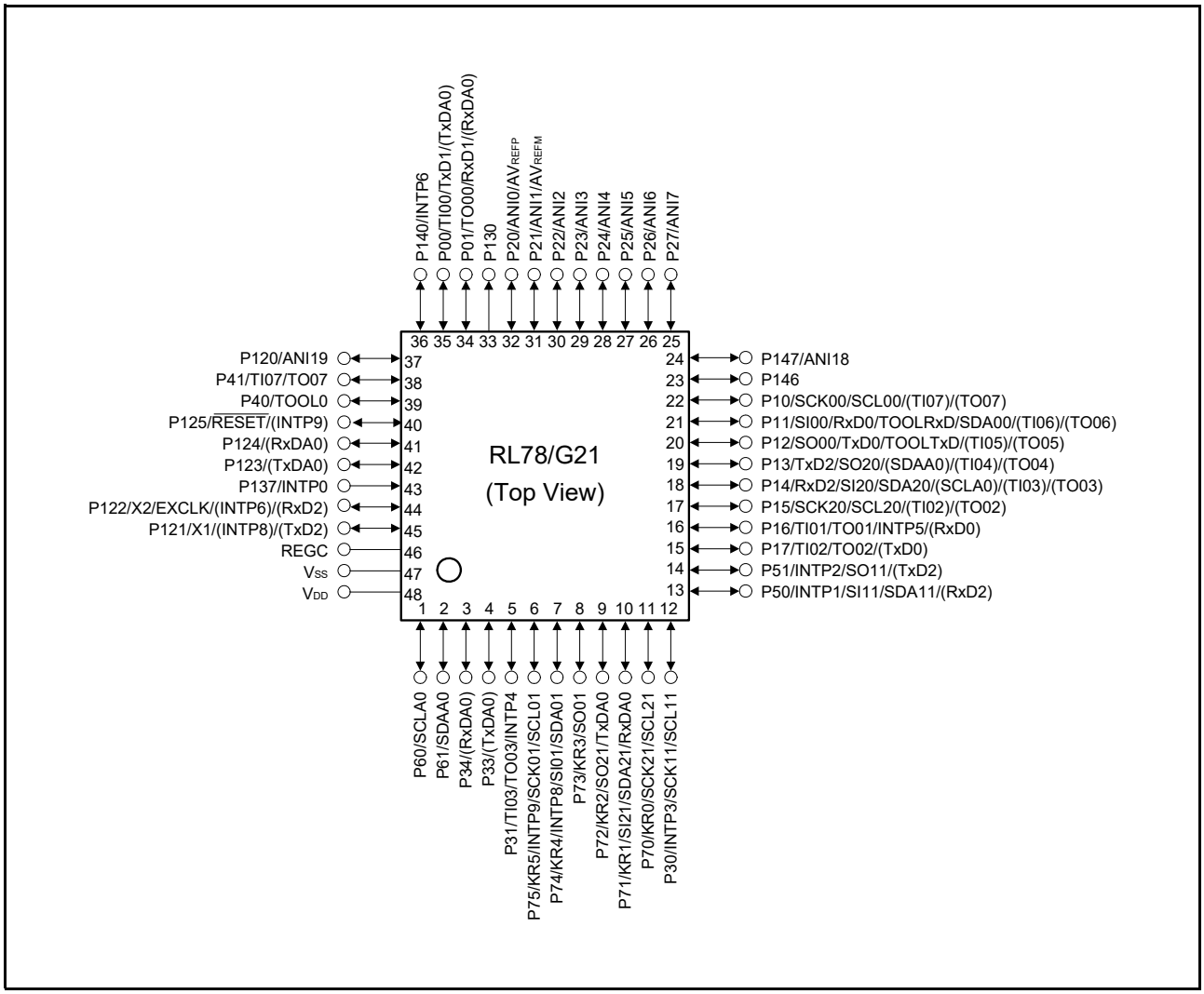
Pin Number	I/O	Power supply, system clock, and debugging	Analog Circuit	HMI		Timer	Communications Interfaces		
44LQFP	Digital port		A/D converter (ADC)	Interrupt	Key interrupt	Timer array unit (TAU)	Serial array unit (SAU)	Serial interface IICA (IICA)	Serial interface UARTA (UARTA)
1	P41	—	—	—	—	TI07/TO07	—	—	—
2	P40	TOOL0	—	—	—	—	—	—	—
3	P125	$\overline{\text{RESET}}$	—	INTP9	—	—	—	—	—
4	P124	—	—	—	—	—	—	—	(RxDA0)
5	P123	—	—	—	—	—	—	—	(TxDA0)
6	P137	—	—	INTP0	—	—	—	—	—
7	P122	X2/EXCLK	—	INTP6	—	—	(RxD2)	—	—
8	P121	X1	—	INTP8	—	—	(TxD2)	—	—
9	—	REGC	—	—	—	—	—	—	—
10	—	V_{SS}	—	—	—	—	—	—	—
11	—	V_{DD}	—	—	—	—	—	—	—
12	P60	—	—	—	—	—	—	SCLA0	—
13	P61	—	—	—	—	—	—	SDAA0	—
14	P34	—	—	—	—	—	—	—	(RxDA0)
15	P33	—	—	—	—	—	—	—	(TxDA0)
16	P31	—	—	INTP4	—	TI03/TO03	—	—	—
17	P73	—	—	—	KR3	—	—	—	—
18	P72	—	—	—	KR2	—	SO21	—	TxDA0
19	P71	—	—	—	KR1	—	SI21/SDA21	—	RxDA0
20	P70	—	—	—	KR0	—	SCK21/SCL21	—	—
21	P30	—	—	INTP3	—	—	SCK11/SCL11	—	—
22	P50	—	—	INTP1	—	—	SI11/SDA11/ (RxD2)	—	—
23	P51	—	—	INTP2	—	—	SO11/(TxD2)	—	—
24	P17	—	—	—	—	TI02/TO02	(TxD0)	—	—
25	P16	—	—	INTP5	—	TI01/TO01	(RxD0)	—	—
26	P15	—	—	—	—	(TI02)/(TO02)	SCK20/SCL20	—	—
27	P14	—	—	—	—	(TI03)/(TO03)	SI20/RxD2/ SDA20	(SCLA0)	—
28	P13	—	—	—	—	(TI04)/(TO04)	SO20/TxD2	(SDAA0)	—
29	P12	TOOLTxD	—	—	—	(TI05)/(TO05)	SO00/TxD0	—	—
30	P11	TOOLRxD	—	—	—	(TI06)/(TO06)	SI00/RxD0/ SDA00	—	—
31	P10	—	—	—	—	(TI07)/(TO07)	SCK00/SCL00	—	—
32	P146	—	—	—	—	—	—	—	—
33	P147	—	ANI18	—	—	—	—	—	—
34	P27	—	ANI7	—	—	—	—	—	—
35	P26	—	ANI6	—	—	—	—	—	—
36	P25	—	ANI5	—	—	—	—	—	—

Table 1 - 4 Multiplexed Pin Functions of the 44-pin Products (2/2)

Pin Number	I/O	Power supply, system clock, and debugging	Analog Circuit	HMI		Timer	Communications Interfaces		
44LQFP	Digital port		A/D converter (ADC)	Interrupt	Key interrupt	Timer array unit (TAU)	Serial array unit (SAU)	Serial interface IICA (IICA)	Serial interface UARTA (UARTA)
37	P24	—	ANI4	—	—	—	—	—	—
38	P23	—	ANI3	—	—	—	—	—	—
39	P22	—	ANI2	—	—	—	—	—	—
40	P21	—	ANI1/AVREFM	—	—	—	—	—	—
41	P20	—	ANI0/AVREFP	—	—	—	—	—	—
42	P01	—	—	—	—	TO00	RxD1	—	(RxDA0)
43	P00	—	—	—	—	TI00	TxD1	—	(TxDA0)
44	P120	—	ANI19	—	—	—	—	—	—

1.3.4 48-pin products

- 48-pin plastic LFQFP (7 × 7 mm, 0.50-mm pitch)



Caution Connect the REGC pin to Vss via a capacitor (0.47 to 1 µF).

Remark 1. For pin identification, see **1.4 Pin Identification**.

Remark 2. Functions in parentheses in the above figure can be assigned via settings in the corresponding peripheral I/O redirection register (PIORx). For details, see **4.3.8 Peripheral I/O redirection registers (PIORx)**.

Table 1 - 5 Multiplexed Pin Functions of the 48-pin Products (1/2)

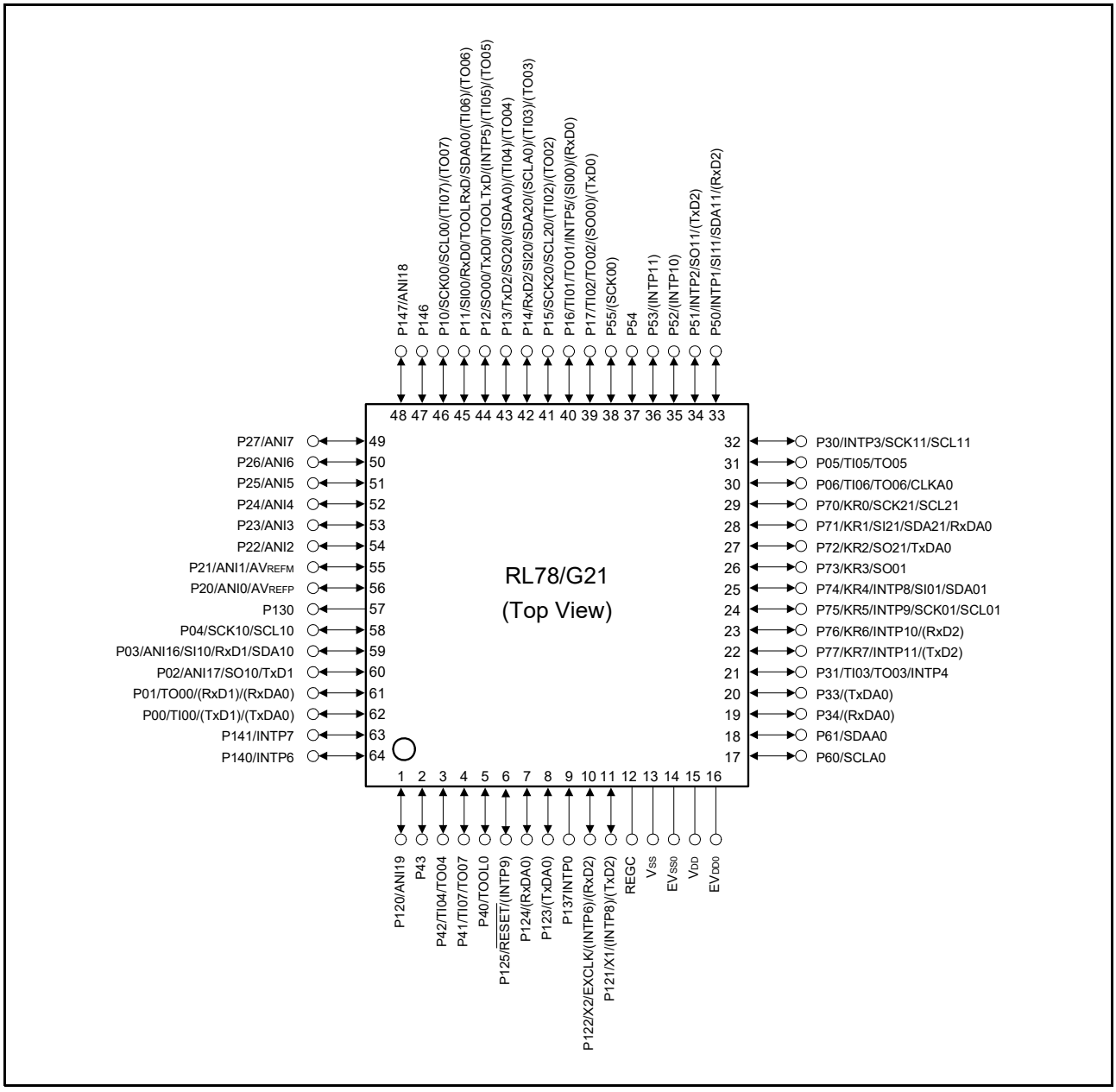
Pin Number	I/O	Power supply, system clock, and debugging	Analog Circuit	HMI		Timer	Communications Interfaces		
48LFQFP	Digital port		A/D converter (ADC)	Interrupt	Key interrupt	Timer array unit (TAU)	Serial array unit (SAU)	Serial interface IICA (IICA)	Serial interface UARTA (UARTA)
1	P60	—	—	—	—	—	—	SCLA0	—
2	P61	—	—	—	—	—	—	SDAA0	—
3	P34	—	—	—	—	—	—	—	(RxDA0)
4	P33	—	—	—	—	—	—	—	(TxDA0)
5	P31	—	—	INTP4	—	TI03/TO03	—	—	—
6	P75	—	—	INTP9	KR5	—	SCK01/SCL01	—	—
7	P74	—	—	INTP8	KR4	—	SI01/SDA01	—	—
8	P73	—	—	—	KR3	—	SO01	—	—
9	P72	—	—	—	KR2	—	SO21	—	TxDA0
10	P71	—	—	—	KR1	—	SI21/SDA21	—	RxDA0
11	P70	—	—	—	KR0	—	SCK21/SCL21	—	—
12	P30	—	—	INTP3	—	—	SCK11/SCL11	—	—
13	P50	—	—	INTP1	—	—	SI11/SDA11/ (RxD2)	—	—
14	P51	—	—	INTP2	—	—	SO11/(TxD2)	—	—
15	P17	—	—	—	—	TI02/TO02	(TxD0)	—	—
16	P16	—	—	INTP5	—	TI01/TO01	(RxD0)	—	—
17	P15	—	—	—	—	(TI02)/(TO02)	SCK20/SCL20	—	—
18	P14	—	—	—	—	(TI03)/(TO03)	SI20/RxD2/ SDA20	(SCLA0)	—
19	P13	—	—	—	—	(TI04)/(TO04)	SO20/TxD2	(SDAA0)	—
20	P12	TOOLTxD	—	—	—	(TI05)/(TO05)	SO00/TxD0	—	—
21	P11	TOOLRxD	—	—	—	(TI06)/(TO06)	SI00/RxD0/ SDA00	—	—
22	P10	—	—	—	—	(TI07)/(TO07)	SCK00/SCL00	—	—
23	P146	—	—	—	—	—	—	—	—
24	P147	—	ANI18	—	—	—	—	—	—
25	P27	—	ANI7	—	—	—	—	—	—
26	P26	—	ANI6	—	—	—	—	—	—
27	P25	—	ANI5	—	—	—	—	—	—
28	P24	—	ANI4	—	—	—	—	—	—
29	P23	—	ANI3	—	—	—	—	—	—
30	P22	—	ANI2	—	—	—	—	—	—
31	P21	—	ANI1/AVREFM	—	—	—	—	—	—
32	P20	—	ANI0/AVREFP	—	—	—	—	—	—
33	P130	—	—	—	—	—	—	—	—
34	P01	—	—	—	—	TO00	RxD1	—	(RxDA0)
35	P00	—	—	—	—	TI00	TxD1	—	(TxDA0)
36	P140	—	—	INTP6	—	—	—	—	—

Table 1 - 5 Multiplexed Pin Functions of the 48-pin Products (2/2)

Pin Number	I/O	Power supply, system clock, and debugging	Analog Circuit	HMI		Timer	Communications Interfaces		
48LFQFP	Digital port		A/D converter (ADC)	Interrupt	Key interrupt	Timer array unit (TAU)	Serial array unit (SAU)	Serial interface IICA (IICA)	Serial interface UARTA (UARTA)
37	P120	—	ANI19	—	—	—	—	—	—
38	P41	—	—	—	—	TI07/TO07	—	—	—
39	P40	TOOL0	—	—	—	—	—	—	—
40	P125	$\overline{\text{RESET}}$	—	(INTP9)	—	—	—	—	—
41	P124	—	—	—	—	—	—	—	(RxDA0)
42	P123	—	—	—	—	—	—	—	(TxDA0)
43	P137	—	—	INTP0	—	—	—	—	—
44	P122	X2/EXCLK	—	(INTP6)	—	—	(RxD2)	—	—
45	P121	X1	—	(INTP8)	—	—	(TxD2)	—	—
46	—	REGC	—	—	—	—	—	—	—
47	—	V_{SS}	—	—	—	—	—	—	—
48	—	V_{DD}	—	—	—	—	—	—	—

1.3.5 64-pin products

- 64-pin plastic LQFP (12 × 12 mm, 0.65-mm pitch)
- 64-pin plastic LFQFP (10 × 10 mm, 0.50-mm pitch)



Caution 1. Connect the EVSS0 pin to the same ground as the VSS pin.

Caution 2. Make sure that the voltage on the VDD pin is no less than that on the EVDD0 pin.

Caution 3. Connect the REGC pin to VSS via a capacitor (0.47 to 1 μF).

Remark 1. For pin identification, see **1.4 Pin Identification**.

Remark 2. When using the microcontroller for an application where the noise generated inside the microcontroller must be reduced, it is recommended to supply separate powers to the VDD and EVDD0 pins and connect the VSS and EVSS0 pins to separate ground lines.

Remark 3. Functions in parentheses in the above figure can be assigned via settings in the corresponding peripheral I/O redirection register (PIORx). For details, see **4.3.8 Peripheral I/O redirection registers (PIORx)**.

Table 1 - 6 Multiplexed Pin Functions of the 64-pin Products (1/2)

Pin Number	I/O	Power supply, system clock, and debugging	Analog Circuit	HMI		Timer	Communications Interfaces		
64LFQFP, 64LQFP	Digital port		A/D converter (ADC)	Interrupt	Key interrupt	Timer array unit (TAU)	Serial array unit (SAU)	Serial interface IICA (IICA)	Serial interface UARTA (UARTA)
1	P120	—	ANI19	—	—	—	—	—	—
2	P43	—	—	—	—	—	—	—	—
3	P42	—	—	—	—	TI04/TO04	—	—	—
4	P41	—	—	—	—	TI07/TO07	—	—	—
5	P40	TOOL0	—	—	—	—	—	—	—
6	P125	$\overline{\text{RESET}}$	—	(INTP9)	—	—	—	—	—
7	P124	—	—	—	—	—	—	—	(RxDA0)
8	P123	—	—	—	—	—	—	—	(TxDA0)
9	P137	—	—	INTP0	—	—	—	—	—
10	P122	X2/EXCLK	—	(INTP6)	—	—	(RxD2)	—	—
11	P121	X1	—	(INTP8)	—	—	(TxD2)	—	—
12	—	REGC	—	—	—	—	—	—	—
13	—	V_{SS}	—	—	—	—	—	—	—
14	—	EV_{SS0}	—	—	—	—	—	—	—
15	—	V_{DD}	—	—	—	—	—	—	—
16	—	EV_{DD0}	—	—	—	—	—	—	—
17	P60	—	—	—	—	—	—	SCLA0	—
18	P61	—	—	—	—	—	—	SDAA0	—
19	P34	—	—	—	—	—	—	—	(RxDA0)
20	P33	—	—	—	—	—	—	—	(TxDA0)
21	P31	—	—	INTP4	—	TI03/TO03	—	—	—
22	P77	—	—	INTP11	KR7	—	(TxD2)	—	—
23	P76	—	—	INTP10	KR6	—	(RxD2)	—	—
24	P75	—	—	INTP9	KR5	—	SCK01/SCL01	—	—
25	P74	—	—	INTP8	KR4	—	SI01/SDA01	—	—
26	P73	—	—	—	KR3	—	SO01	—	—
27	P72	—	—	—	KR2	—	SO21	—	TxDA0
28	P71	—	—	—	KR1	—	SI21/SDA21	—	RxDA0
29	P70	—	—	—	KR0	—	SCK21/SCL21	—	—
30	P06	—	—	—	—	TI06/TO06	—	—	CLKA0
31	P05	—	—	—	—	TI05/TO05	—	—	—
32	P30	—	—	INTP3	—	—	SCK11/SCL11	—	—
33	P50	—	—	INTP1	—	—	SI11/SDA11/ (RxD2)	—	—
34	P51	—	—	INTP2	—	—	SO11/(TxD2)	—	—
35	P52	—	—	(INTP10)	—	—	—	—	—
36	P53	—	—	(INTP11)	—	—	—	—	—
37	P54	—	—	—	—	—	—	—	—

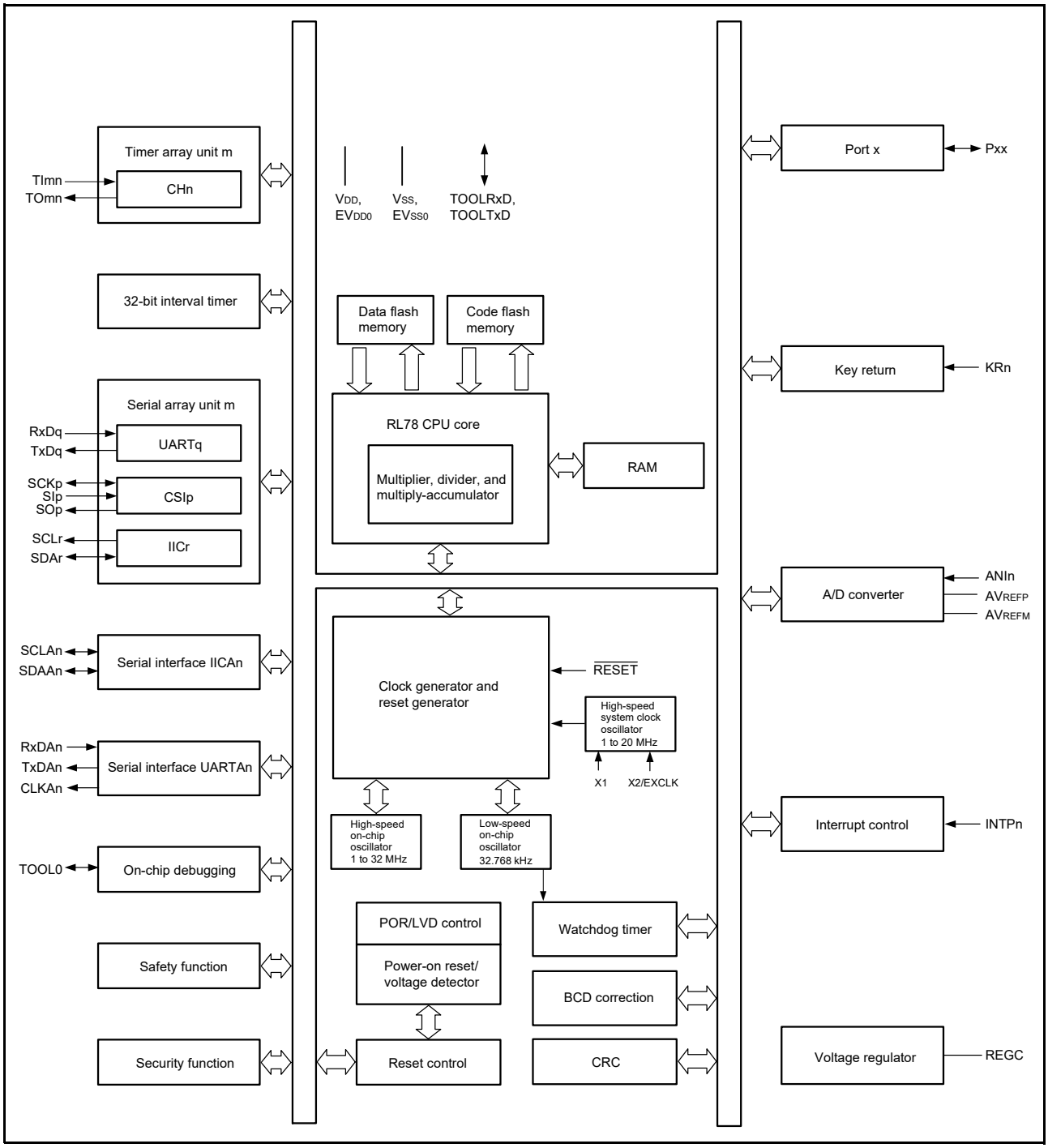
Table 1 - 6 Multiplexed Pin Functions of the 64-pin Products (2/2)

Pin Number	I/O	Power supply, system clock, and debugging	Analog Circuit	HMI		Timer	Communications Interfaces		
64LFQFP, 64LQFP	Digital port		A/D converter (ADC)	Interrupt	Key interrupt	Timer array unit (TAU)	Serial array unit (SAU)	Serial interface IICA (IICA)	Serial interface UARTA (UARTA)
38	P55	—	—	—	—	—	(SCK00)	—	—
39	P17	—	—	—	—	TI02/TO02	(SO00)/(TxD0)	—	—
40	P16	—	—	INTP5	—	TI01/TO01	(SI00)/(RxD0)	—	—
41	P15	—	—	—	—	(TI02)/(TO02)	SCK20/SCL20	—	—
42	P14	—	—	—	—	(TI03)/(TO03)	SI20/RxD2/ SDA20	(SCLA0)	—
43	P13	—	—	—	—	(TI04)/(TO04)	SO20/TxD2	(SDAA0)	—
44	P12	TOOLTxD	—	(INTP5)	—	(TI05)/(TO05)	SO00/TxD0	—	—
45	P11	TOOLRxD	—	—	—	(TI06)/(TO06)	SI00/RxD0/ SDA00	—	—
46	P10	—	—	—	—	(TI07)/(TO07)	SCK00/SCL00	—	—
47	P146	—	—	—	—	—	—	—	—
48	P147	—	ANI18	—	—	—	—	—	—
49	P27	—	ANI7	—	—	—	—	—	—
50	P26	—	ANI6	—	—	—	—	—	—
51	P25	—	ANI5	—	—	—	—	—	—
52	P24	—	ANI4	—	—	—	—	—	—
53	P23	—	ANI3	—	—	—	—	—	—
54	P22	—	ANI2	—	—	—	—	—	—
55	P21	—	ANI1/AVREFM	—	—	—	—	—	—
56	P20	—	ANI0/AVREFP	—	—	—	—	—	—
57	P130	—	—	—	—	—	—	—	—
58	P04	—	—	—	—	—	SCK10/SCL10	—	—
59	P03	—	ANI16	—	—	—	SI10/RxD1/ SDA10	—	—
60	P02	—	ANI17	—	—	—	SO10/TxD1	—	—
61	P01	—	—	—	—	TO00	(RxD1)	—	(RxDA0)
62	P00	—	—	—	—	TI00	(TxD1)	—	(TxDA0)
63	P141	—	—	INTP7	—	—	—	—	—
64	P140	—	—	INTP6	—	—	—	—	—

1.4 Pin Identification

ANI0 to ANI7,		$\overline{\text{RESET}}$:	Reset
ANI16 to ANI19:	Analog input	RxD0 to RxD2, RxDA0:	Receive data
AV_{REFM}:	Analog reference voltage minus	SCLA0,	
AV_{REFP}:	Analog reference voltage plus	SCK00, SCK01, SCK10,	
CLKA0:	Asynchronous serial clock output	SCK11, SCK20, SCK21:	Serial clock input/output
EV_{DD0}:	Power supply for port	SCL00, SCL01, SCL10,	
EV_{SS0}:	Ground for port	SCL11, SCL20, SCL21:	Serial clock output
EXCLK:	External clock input	SDAA0, SDA00, SDA01,	
	(main system clock)	SDA10, SDA11,	
INTP0 to INTP11:	Interrupt request from peripheral	SDA20, SDA21:	Serial data input/output
KR0 to KR7:	Key return input	SI00, SI01, SI10, SI11,	
P00 to P06:	Port 0	SI20, SI21:	Serial data input
P10 to P17:	Port 1	SO00, SO01, SO10,	
P20 to P27:	Port 2	SO11, SO20, SO21:	Serial data output
P30, P31, P33, P34:	Port 3	TI00 to TI07:	Timer input
P40 to P43:	Port 4	TO00 to TO07:	Timer output
P50 to P55:	Port 5	TOOL0:	Data input/output for tool
P60, P61:	Port 6	TOOLRxD, TOOLTxD:	Data input/output for external device
P70 to P77:	Port 7	TxD0 to TxD2, TxDA0:	Transmit data
P120 to P125:	Port 12	V_{DD}:	Power supply
P130, P137:	Port 13	V_{SS}:	Ground
P140, P141, P146, P147:	Port 14	X1, X2:	Crystal oscillator (main system clock)
REGC:	Regulator capacitance		

1.5 Block Diagram



Caution **44- to 64-pin products incorporate the key return function.**

Remark m: Unit number, n: Channel number, p: Simplified SPI (CSI) number, q: UART number, r: Simplified I2C number, xx: Port number

1.6 Outline of Functions

[20- to 64-pin products]

Caution This outline describes the functions at the time when the corresponding peripheral I/O redirection register (PIORx) is set to 00H.

(1/3)

Item		20-pin	32-pin	44-pin	48-pin	64-pin
		R7F103G6x	R7F103GBx	R7F103GFx	R7F103GGx	R7F103GLx
Code flash memory		128 or 256 KB				
Data flash memory		4 KB				
RAM		8 KB				
Address space		1 MB				
CPU/peripheral hardware clock frequency (f_{CLK})	Main system clock	HS (high-speed main) mode: 1 to 32 MHz (V_{DD} = 1.8 to 5.5 V) HS (high-speed main) mode: 1 to 4 MHz**Note 1** (V_{DD} = 1.6 to 5.5 V) LS (low-speed main) mode: 1 to 24 MHz (V_{DD} = 1.8 to 5.5 V) LS (low-speed main) mode: 1 to 4 MHz**Note 1** (V_{DD} = 1.6 to 5.5 V) LP (low-power main) mode: 1 to 2 MHz**Note 2** (V_{DD} = 1.6 to 5.5 V)				
	Subsystem clock	SUB mode: 32.768 kHz (V_{DD} = 1.6 to 5.5 V)				
Main system clock	High-speed system clock (f_{MX})	1 to 20 MHz				
	High-speed on-chip oscillator clock (f_{IH})	1 MHz, 2 MHz, 3 MHz, 4 MHz, 6 MHz, 8 MHz, 12 MHz, 16 MHz, 24 MHz, 32 MHz				
Subsystem clock	Low-speed on-chip oscillator clock (f_{IL})	32.768 kHz (typ.)				
General-purpose registers		8 bits × 32 registers (8 bits × 8 registers × 4 banks)				
Minimum instruction execution time		0.03125 μs (at the 32-MHz operation with the high-speed on-chip oscillator clock (f_{IH}))				
Instruction set		• Data transfer (8/16 bits) • Adder and subtractor/logical operation (8/16 bits) • Multiplication (8 bits × 8 bits, 16 bits × 16 bits), division (16 bits ÷ 16 bits, 32 bits ÷ 32 bits) • Multiplication and accumulation (16 bits × 16 bits + 32 bits) • Rotate, barrel shift, and bit manipulation (set, reset, test, and Boolean operation), etc.				
I/O port	Total number of pins	17	29	41	45	59
	CMOS I/O	16 (N-ch open-drain I/O [withstand voltage of V_{DD}]: 5)	26 (N-ch open-drain I/O [withstand voltage of V_{DD}]: 11)	38 (N-ch open-drain I/O [withstand voltage of V_{DD}]: 14)	41 (N-ch open-drain I/O [withstand voltage of V_{DD}]: 15)	55 (N-ch open-drain I/O [withstand voltage of V_{DD}]: 19)
	CMOS input	1				
	CMOS output	0	0	0	1	1
	N-ch open-drain I/O (withstand voltage: 6 V)	0	2	2	2	2

(2/3)

Item		20-pin	32-pin	44-pin	48-pin	64-pin
		R7F103G6x	R7F103GBx	R7F103GFx	R7F103GGx	R7F103GLx
Timers	16-bit timer	8 channels				
	Watchdog timer	1 channel				
	32-bit interval timer (TML32)	1 channel in 32-bit counter mode, 2 channels in 16-bit counter mode, 4 channels in 8-bit counter mode				
	Timer output	6 channels (PWM outputs: 5Note 3)	4 channels (PWM outputs: 3Note 3), 8 channels (PWM outputs: 7Note 3)Note 4	5 channels (PWM outputs: 3Note 3), 8 channels (PWM outputs: 7Note 3) Note 4		8 channels (PWM outputs: 7Note 3)
8-/10-bit resolution A/D converter		6 channels	8 channels	10 channels	10 channels	12 channels
Serial interface		[20-pin products] • Single simplified SPI (CSI) or simplified I2C channel, or single UART • Single simplified SPI (CSI) or simplified I2C channel, or single UART • Single UART, and not usable for simplified SPI (CSI) or simplified I2C [32-pin products] • Single simplified SPI (CSI) or simplified I2C channel, or single UART • Single simplified SPI (CSI) or simplified I2C channel, or single UART • Single simplified SPI (CSI) or simplified I2C channel, or single UART also supporting the LIN bus [44-pin products] • Single simplified SPI (CSI) or simplified I2C channel, or single UART • Single simplified SPI (CSI) or simplified I2C channel, or single UART • Two simplified SPI (CSI) or simplified I2C channels, or single UART also supporting the LIN bus [48-pin products] • Two simplified SPI (CSI) or simplified I2C channels, or single UART • Single simplified SPI (CSI) or simplified I2C channel, or single UART • Two simplified SPI (CSI) or simplified I2C channels, or single UART also supporting the LIN bus [64-pin products] • Two simplified SPI (CSI) or simplified I2C channels, or single UART • Two simplified SPI (CSI) or simplified I2C channels, or single UART • Two simplified SPI (CSI) or simplified I2C channels, or single UART also supporting the LIN bus				
	UARTA	1 channel				
	I2C bus	—	1 channel			
Vectored interrupt sources	Internal	23	26	26	26	26
	External	6	9	10	10	13
Key interrupt		—		4	6	8
Reset		• Reset by $\overline{\text{RESET}}$ pin • Internal reset by watchdog timer • Internal reset by power-on-reset • Internal reset by voltage detectors (LVD0 and LVD1) • Internal reset by illegal instruction executionNote 5 • Internal reset by RAM parity error • Internal reset by illegal-memory access				
Power-on-reset circuit		Detection voltage • 1.50 V (typ.)				

(3/3)

Item		20-pin	32-pin	44-pin	48-pin	64-pin
		R7F103G6x	R7F103GBx	R7F103GFx	R7F103GGx	R7F103GLx
Voltage detector	LVD0	Detection voltage • Rising edge: 1.69 V to 3.96 V (6 stages) • Falling edge: 1.65 V to 3.88 V (6 stages)				
	LVD1	Detection voltage • Rising edge: 1.67 V to 4.16 V (18 stages) • Falling edge: 1.63 V to 4.08 V (18 stages)				
On-chip debugging		Available (tracing supported)				
Power supply voltage		V_{DD} = 1.6 to 5.5 V				
Operating ambient temperature		T_A = -40 to +105°C (3Z: Industrial and consumer applications)				

Note 1. Ensure that the operating voltage is at least 1.8 V during overwriting of the flash memory.

Note 2. When the flash memory is to be overwritten, switch to high-speed main (HS) mode or low-speed main (LS) mode.

Note 3. The number of PWM outputs varies depending on the setting of channels in use (the number of masters and slaves). For details, see **7.9.3 Operation for the multiple PWM output function**.

Note 4. This applies when the corresponding bit in the peripheral I/O redirection register (PIOR0) is set to 1.

Note 5. In normal operation, executing the instruction code FFH triggers an internal reset, but this is not the case during emulation by the on-chip debugging emulator.

Section 2 Pin Functions

2.1 Functions of Port Pins

Pin I/O buffer power supplies depend on the product. The relationship between these power supplies and the pins is shown below.

Table 2 - 1 Pin I/O Buffer Power Supplies

1. 20- to 48-pin products

Power Supply	Corresponding Pins
V_{DD}	All pins

2. 64-pin products

Power Supply	Corresponding Pins
EV_{DD0}	• Port pins other than P20 to P27, P121 to P125, and P137
V_{DD}	• P20 to P27, P121 to P125, and P137 • $\overline{RESET}$, REGC

2.1.1 20-pin products

Function Name	Pin Type	I/O	After Reset Release	Alternate Function	Function
P00	7-3-5	I/O	Analog function	ANI17/TI00/TxD1/TxDA0	Port 0. 2-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. A TTL input buffer can be set for P01 as an input. N-ch open-drain output (withstand voltage of VDD) can be set for P00 as an output. P00 and P01 can be set for the analog pin functions**Note**.
P01	8-3-3			ANI16/TO00/RxD1/RxDA0	
P10	8-1-10	I/O	Input port	SCK00/SCL00/TI07/TO07	Port 1. 5-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. TTL input buffers can be set for P10, P11, P16, and P17 as inputs. N-ch open-drain output (withstand voltage of VDD) can be set for P10 to P12 and P17 as outputs.
P11				SI00/RxD0/TOOLRxD/ SDA00/TI06/TO06	
P12	7-1-11			SO00/TxD0/TOOLTxD/ TI05/TO05	
P16	8-1-3			TI01/TO01/INTP5/SO11	
P17	8-1-10			TI02/TO02/SI11/SDA11	
P20	4-3-5	I/O	Analog function	ANI0/AVREFP	Port 2. 3-bit I/O port. Input or output can be specified in 1-bit units. P20 to P22 can be set for the analog pin functions**Note**.
P21				ANI1/AVREFM	
P22				ANI2	
P30	7-1-3	I/O	Input port	INTP3/SCK11/SCL11	Port 3. 1-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port.
P40				TOOL0	Port 4. 1-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port.
P121	7-2-3	I/O	Input port	X1/INTP8/TxD2	Port 12. 3-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. P125 is multiplexed with the input pin function for the external reset ($\overline{\text{RESET}}$). When P125 is to be used for the external reset, set the PORTSELB bit of the user option byte (000C1H) to 1.
P122				X2/EXCLK/INTP6/RxD2	
P125	3-2-2			$\overline{\text{RESET}}$/INTP9	
P137	3-1-2	Input	Input port	INTP0	Port 13. 1-bit input-only port. Use of an on-chip pull-up resistor can be specified by a software setting at input port.
P147	7-3-3	I/O	Analog function	ANI18	Port 14. 1-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. P147 can be set for the analog pin function**Note**.

Note Digital or analog can be selected per pin (in 1-bit units) with the port mode control A register (PMCAxx).

2.1.2 32-pin products

(1/2)

Function Name	Pin Type	I/O	After Reset Release	Alternate Function	Function
P00	7-3-5	I/O	Analog function	ANI17/TI00/TxD1/TxDA0	Port 0. 2-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. A TTL input buffer can be set for P01 as an input. N-ch open-drain output (withstand voltage of VDD) can be set for P00 as an output. P00 and P01 can be set for the analog pin functions**Note**.
P01	8-3-3			ANI16/TO00/RxD1/RxDA0	
P10	8-1-10	I/O	Input port	SCK00/SCL00/(TI07)/(TO07)	Port 1. 8-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. TTL input buffers can be set for P10, P11, and P13 to P17 as inputs. N-ch open-drain output (withstand voltage of VDD) can be set for P10 to P15 and P17 as outputs.
P11				SI00/RxD0/TOOLRxD/SDA00/(TI06)/(TO06)	
P12	7-1-11			SO00/TxD0/TOOLTxD/(TI05)/(TO05)	
P13	8-1-10			TxD2/SO20/(SDAA0)/(TI04)/(TO04)	
P14				RxD2/SI20/SDA20/(SCLA0)/(TI03)/(TO03)	
P15				SCK20/SCL20/(TI02)/(TO02)	
P16	8-1-3			TI01/TO01/INTP5/(RxD0)	
P17	8-1-10			TI02/TO02/(TxD0)	
P20	4-3-5	I/O	Analog function	ANI0/AVREFP	Port 2. 4-bit I/O port. Input or output can be specified in 1-bit units. P20 to P23 can be set for the analog pin functions**Note**.
P21				ANI1/AVREFM	
P22				ANI2	
P23				ANI3	
P30	7-1-3	I/O	Input port	INTP3/SCK11/SCL11	Port 3. 3-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. N-ch open-drain output (withstand voltage of VDD) can be set for P31 as an output.
P31	7-1-11			TI03/TO03/INTP4/(TxDA0)	
P34	8-1-3			(RxDA0)	
P40	7-1-3	I/O	Input port	TOOL0	Port 4. 1-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port.
P50	8-1-10	I/O	Input port	INTP1/SI11/SDA11/(RxD2)	Port 5. 2-bit I/O port. N-ch open-drain output (withstand voltage of VDD) can be set for P50 and P51 as outputs. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. A TTL input buffer can be set for P50 as an input.
P51	7-1-11			INTP2/SO11/(TxD2)	

(2/2)

Function Name	Pin Type	I/O	After Reset Release	Alternate Function	Function
P60	12-1-3	I/O	Input port	SCLA0	Port 6. 2-bit I/O port. Input or output can be specified in 1-bit units. The outputs of P60 and P61 are N-ch open-drain (withstand voltage of 6 V).
P61				SDAA0	
P70	7-1-3	I/O	Input port	—	Port 7. 1-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port.
P120	7-3-3	I/O	Analog function	ANI19	Port 12. 4-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. P120 can be set for the analog pin function**Note** . P125 is multiplexed with the input pin function for the external reset ($\overline{\text{RESET}}$). When P125 is to be used for the external reset, set the PORTSELB bit of the user option byte (000C1H) to 1.
P121	7-2-3		Input port	X1/INTP8/(TxD2)	
P122				X2/EXCLK/INTP6/(RxD2)	
P125	3-2-2			$\overline{\text{RESET}}$/INTP9	
P137	3-1-2	Input	Input port	INTP0	Port 13. 1-bit input-only port. Use of an on-chip pull-up resistor can be specified by a software setting at input port.
P147	7-3-3	I/O	Analog function	ANI18	Port 14. 1-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. P147 can be set for the analog pin function**Note**.

Note Digital or analog can be selected per pin (in 1-bit units) with the port mode control A register (PMCAxx).

Remark Pin functions in parentheses in the above table can be assigned via settings in the corresponding peripheral I/O redirection register (PIORx). For details, see **4.3.8 Peripheral I/O redirection registers (PIORx)**.

2.1.3 44-pin products

(1/2)

Function Name	Pin Type	I/O	After Reset Release	Alternate Function	Function
P00	7-1-11	I/O	Input port	TI00/TxD1/(TxDA0)	Port 0. 2-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. A TTL input buffer can be set for P01 as an input. N-ch open-drain output (withstand voltage of VDD) can be set for P00 as an output.
P01	8-1-3			TO00/RxD1/(RxDA0)	
P10	8-1-10	I/O	Input port	SCK00/SCL00/(TI07)/(TO07)	Port 1. 8-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. TTL input buffers can be set for P10, P11, and P13 to P17 as inputs. N-ch open-drain output (withstand voltage of VDD) can be set for P10 to P15 and P17 as outputs.
P11				SI00/RxD0/TOOLRxD/SDA00/(TI06)/(TO06)	
P12	7-1-11			SO00/TxD0/TOOLTxD/(TI05)/(TO05)	
P13	8-1-10			TxD2/SO20/(SDAA0)/(TI04)/(TO04)	
P14				RxD2/SI20/SDA20/(SCLA0)/(TI03)/(TO03)	
P15				SCK20/SCL20/(TI02)/(TO02)	
P16	8-1-3			TI01/TO01/INTP5/(RxD0)	
P17	8-1-10			TI02/TO02/(TxD0)	
P20	4-3-5	I/O	Analog function	ANI0/AVREFP	Port 2. 8-bit I/O port. Input or output can be specified in 1-bit units. P20 to P27 can be set for the analog pin functions**Note** .
P21				ANI1/AVREFM	
P22				ANI2	
P23				ANI3	
P24				ANI4	
P25				ANI5	
P26				ANI6	
P27				ANI7	
P30	7-1-3	I/O	Input port	INTP3/SCK11/SCL11	Port 3. 4-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. N-ch open-drain output (withstand voltage of VDD) can be set for P31 and P33 as outputs. A TTL input buffer can be set for P34 as an input.
P31	7-1-11			TI03/TO03/INTP4	
P33				(TxDA0)	
P34	8-1-3			(RxDA0)	
P40	7-1-3	I/O	Input port	TOOL0	Port 4. 2-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port.
P41				TI07/TO07	

(2/2)

Function Name	Pin Type	I/O	After Reset Release	Alternate Function	Function
P50	8-1-10	I/O	Input port	INTP1/SI11/SDA11/(RxD2)	Port 5. 2-bit I/O port. N-ch open-drain output (withstand voltage of VDD) can be set for P50 and P51 as outputs. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. A TTL input buffer can be set for P50 as an input.
P51	7-1-11			INTP2/SO11/(TxD2)	
P60	12-1-3	I/O	Input port	SCLA0	Port 6. 2-bit I/O port. Input or output can be specified in 1-bit units. The outputs of P60 and P61 are N-ch open-drain (withstand voltage of 6 V).
P61				SDAA0	
P70	7-1-3	I/O	Input port	KR0/SCK21/SCL21	Port 7. 4-bit I/O port. A TTL input buffer can be set for P71 as an input. N-ch open-drain output (withstand voltage of VDD) can be set for P71 and P72 as outputs. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port.
P71	8-1-10			KR1/SI21/SDA21/RxDA0	
P72	7-1-11			KR2/SO21/TxDA0	
P73	7-1-3			KR3	
P120	7-3-3	I/O	Analog function	ANI19	Port 12. 6-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. P120 can be set for the analog pin function[Note] . P125 is multiplexed with the input pin function for the external reset ($\overline{\text{RESET}}$). When P125 is to be used for the external reset, set the PORTSELB bit of the user option byte (000C1H) to 1.
P121	7-2-3		Input port	X1/INTP8/(TxD2)	
P122				X2/EXCLK/INTP6/(RxD2)	
P123	7-1-3			(TxDA0)	
P124				(RxDA0)	
P125	3-2-2		Input port	$\overline{\text{RESET}}$/INTP9	
P137	3-1-2	Input	Input port	INTP0	Port 13. 1-bit input-only port. Use of an on-chip pull-up resistor can be specified by a software setting at input port.
P146	7-1-3	I/O	Input port	—	Port 14. 2-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. P147 can be set for the analog pin function[Note] .
P147	7-3-3		Analog function	ANI18	

Note Digital or analog can be selected per pin (in 1-bit units) with the port mode control A register (PMCAxx).

Remark Pin functions in parentheses in the above table can be assigned via settings in the corresponding peripheral I/O redirection register (PIORx). For details, see **4.3.8 Peripheral I/O redirection registers (PIORx)**.

2.1.4 48-pin products

(1/2)

Function Name	Pin Type	I/O	After Reset Release	Alternate Function	Function
P00	7-1-11	I/O	Input port	TI00/TxD1/(TxDA0)	Port 0. 2-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. A TTL input buffer can be set for P01 as an input. N-ch open-drain output (withstand voltage of VDD) can be set for P00 as an output.
P01	8-1-3			TO00/RxD1/(RxDA0)	
P10	8-1-10	I/O	Input port	SCK00/SCL00/(TI07)/(TO07)	Port 1. 8-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. TTL input buffers can be set for P10, P11, and P13 to P17 as inputs. N-ch open-drain output (withstand voltage of VDD) can be set for P10 to P15 and P17 as outputs.
P11				SI00/RxD0/TOOLRxD/SDA00/(TI06)/(TO06)	
P12	7-1-11			SO00/TxD0/TOOLTxD/(TI05)/(TO05)	
P13	8-1-10			TxD2/SO20/(SDAA0)/(TI04)/(TO04)	
P14				RxD2/SI20/SDA20/(SCLA0)/(TI03)/(TO03)	
P15				SCK20/SCL20/(TI02)/(TO02)	
P16	8-1-3			TI01/TO01/INTP5/(RxD0)	
P17	8-1-10			TI02/TO02/(TxD0)	
P20	4-3-5	I/O	Analog function	ANI0/AVREFP	Port 2. 8-bit I/O port. Input or output can be specified in 1-bit units. P20 to P27 can be set for the analog pin functions**Note** .
P21				ANI1/AVREFM	
P22				ANI2	
P23				ANI3	
P24				ANI4	
P25				ANI5	
P26				ANI6	
P27				ANI7	
P30	7-1-3	I/O	Input port	INTP3/SCK11/SCL11	Port 3. 4-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. N-ch open-drain output (withstand voltage of VDD) can be set for P31 and P33 as outputs. A TTL input buffer can be set for P34 as an input.
P31	7-1-11			TI03/TO03/INTP4	
P33				(TxDA0)	
P34	8-1-3			(RxDA0)	
P40	7-1-3	I/O	Input port	TOOL0	Port 4. 2-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port.
P41				TI07/TO07	

(2/2)

Function Name	Pin Type	I/O	After Reset Release	Alternate Function	Function
P50	8-1-10	I/O	Input port	INTP1/SI11/SDA11/(RxD2)	Port 5. 2-bit I/O port. N-ch open-drain output (withstand voltage of VDD) can be set for P50 and P51 as outputs. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. A TTL input buffer can be set for P50 as an input.
P51	7-1-11			INTP2/SO11/(TxD2)	
P60	12-1-3	I/O	Input port	SCLA0	Port 6. 2-bit I/O port. Input or output can be specified in 1-bit units. The outputs of P60 and P61 are N-ch open-drain (withstand voltage of 6 V).
P61				SDAA0	
P70	7-1-3	I/O	Input port	KR0/SCK21/SCL21	Port 7. 6-bit I/O port. A TTL input buffer can be set for P71 as an input. N-ch open-drain output (withstand voltage of VDD) can be set for P71, P72, and P74 as outputs. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port.
P71	8-1-10			KR1/SI21/SDA21/RxDA0	
P72	7-1-11			KR2/SO21/TxDA0	
P73	7-1-3			KR3/SO01	
P74	7-1-11			KR4/INTP8/SI01/SDA01	
P75	7-1-3			KR5/INTP9/SCK01/SCL01	
P120	7-3-3	I/O	Analog function	ANI19	Port 12. 6-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. P120 can be set for the analog pin function**Note** . P125 is multiplexed with the input pin function for the external reset ($\overline{\text{RESET}}$). When P125 is to be used for the external reset, set the PORTSELB bit of the user option byte (000C1H) to 1.
P121	7-2-3		Input port	X1/INTP8/(TxD2)	
P122				X2/EXCLK/INTP6/(RxD2)	
P123	7-1-3			(TxDA0)	
P124				(RxDA0)	
P125	3-2-2			$\overline{\text{RESET}}$/INTP9	
P130	1-1-1	Output	Output port	—	Port 13. 1-bit output-only port and 1-bit input-only port. Use of an on-chip pull-up resistor can be specified by a software setting at input port.
P137	3-1-2	Input	Input port	INTP0	
P140	7-1-3	I/O	Input port	INTP6	Port 14. 3-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. P147 can be set for the analog pin function**Note** .
P146				—	
P147	7-3-3		Analog function	ANI18	

Note Digital or analog can be selected per pin (in 1-bit units) with the port mode control A register (PMCAxx).

Remark Pin functions in parentheses in the above table can be assigned via settings in the corresponding peripheral I/O redirection register (PIORx). For details, see **4.3.8 Peripheral I/O redirection registers (PIORx)**.

2.1.5 64-pin products

Function Name	Pin Type	I/O	After Reset Release	Alternate Function	Function
P00	7-1-11	I/O	Input port	TI00/(TxD1)/(TxDA0)	Port 0. 7-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. TTL input buffers can be set for P01, P03, and P04 as inputs. N-ch open-drain output (withstand voltage of EV_{DD}) can be set for P00 and P02 to P04 as outputs. P02 and P03 can be set for the analog pin functions[Note] .
P01	8-1-3			TO00/(RxD1)/(RxDA0)	
P02	7-3-5		Analog function	ANI17/SO10/TxD1	
P03	8-3-6			ANI16/SI10/RxD1/SDA10	
P04	8-1-10		Input port	SCK10/SCL10	
P05	7-1-3			TI05/TO05	
P06				TI06/TO06/CLKA0	
P10	8-1-10	I/O	Input port	SCK00/SCL00/(TI07)/(TO07)	Port 1. 8-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. TTL input buffers can be set for P10, P11, and P13 to P17 as inputs. N-ch open-drain output (withstand voltage of EV_{DD}) can be set for P10 to P15 and P17 as outputs.
P11				SI00/RxD0/TOOLRxD/SDA00/(TI06)/(TO06)	
P12	7-1-11			SO00/TxD0/TOOLTxD/(INTP5)/(TI05)/(TO05)	
P13	8-1-10			TxD2/SO20/(SDAA0)/(TI04)/(TO04)	
P14				RxD2/SI20/SDA20/(SCLA0)/(TI03)/(TO03)	
P15				SCK20/SCL20/(TI02)/(TO02)	
P16	8-1-3			TI01/TO01/INTP5/(SI00)/(RxD0)	
P17	8-1-10			TI02/TO02/(SO00)/(TxD0)	
P20	4-3-5	I/O	Analog function	ANI0/AV_{REFP}	Port 2. 8-bit I/O port. Input or output can be specified in 1-bit units. P20 to P27 can be set for the analog pin functions[Note] .
P21				ANI1/AV_{REFM}	
P22				ANI2	
P23				ANI3	
P24				ANI4	
P25				ANI5	
P26				ANI6	
P27				ANI7	
P30	7-1-3	I/O	Input port	INTP3/SCK11/SCL11	Port 3. 4-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. N-ch open-drain output (withstand voltage of V_{DD}) can be set for P31 and P33 as outputs. A TTL input buffer can be set for P34 as an input.
P31	7-1-11			TI03/TO03/INTP4	
P33				(TxDA0)	
P34	8-1-3			(RxDA0)	
P40	7-1-3	I/O	Input port	TOOL0	Port 4. 4-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port.
P41				TI07/TO07	
P42				TI04/TO04	
P43				—	

Function Name	Pin Type	I/O	After Reset Release	Alternate Function	Function
P50	8-1-10	I/O	Input port	INTP1/SI11/SDA11/(RxD2)	Port 5. 6-bit I/O port. TTL input buffers can be set for P50 and P55 as inputs. N-ch open-drain output (withstand voltage of EV_{DD}) can be set for P50, P51, and P55 as outputs. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port.
P51	7-1-11			INTP2/SO11/(TxD2)	
P52	7-1-3			(INTP10)	
P53				(INTP11)	
P54				—	
P55	8-1-10			(SCK00)	
P60	12-1-3	I/O	Input port	SCLA0	Port 6. 2-bit I/O port. Input or output can be specified in 1-bit units. The outputs of P60 and P61 are N-ch open-drain (withstand voltage of 6 V).
P61				SDAA0	
P70	7-1-3	I/O	Input port	KR0/SCK21/SCL21	Port 7. 8-bit I/O port. A TTL input buffer can be set for P71 as an input. N-ch open-drain output (withstand voltage of EV_{DD}) can be set for P71, P72, and P74 as outputs. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port.
P71	8-1-10			KR1/SI21/SDA21/RxDA0	
P72	7-1-11			KR2/SO21/TxDA0	
P73	7-1-3			KR3/SO01	
P74	7-1-11			KR4/INTP8/SI01/SDA01	
P75	7-1-3			KR5/INTP9/SCK01/SCL01	
P76				KR6/INTP10/(RxD2)	
P77				KR7/INTP11/(TxD2)	
P120	7-3-3	I/O	Analog function	ANI19	Port 12. 6-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. P120 can be set for the analog pin function**Note**. P125 is multiplexed with the input pin function for the external reset ($\overline{\text{RESET}}$). When P125 is to be used for the external reset, set the PORTSELB bit of the user option byte (000C1H) to 1.
P121	7-2-3		Input port	X1/INTP8/(TxD2)	
P122				X2/EXCLK/INTP6/(RxD2)	
P123	7-1-3			(TxDA0)	
P124				(RxDA0)	
P125	3-2-2			$\overline{\text{RESET}}$/INTP9	
P130	1-1-1	Output	Output port	—	Port 13. 1-bit output-only port and 1-bit input-only port. Use of an on-chip pull-up resistor can be specified by a software setting at input port.
P137	3-1-2	Input	Input port	INTP0	
P140	7-1-3	I/O	Input port	INTP6	Port 14. 4-bit I/O port. Input or output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port. P147 can be set for the analog pin function**Note**.
P141				INTP7	
P146				—	
P147	7-3-3		Analog function	ANI18	

Note Digital or analog can be selected per pin (in 1-bit units) with the port mode control A register (PMCAxx).

Remark Pin functions in parentheses in the above table can be assigned via settings in the corresponding peripheral I/O redirection register (PIORx). For details, see **4.3.8 Peripheral I/O redirection registers (PIORx)**.

2.2 Pin Functions Other than Port Pin Functions

2.2.1 Functions for each product

(1/3)

Function Name	64-pin	48-pin	44-pin	32-pin	20-pin
ANI0	✓	✓	✓	✓	✓
ANI1	✓	✓	✓	✓	✓
ANI2	✓	✓	✓	✓	✓
ANI3	✓	✓	✓	✓	—
ANI4	✓	✓	✓	—	—
ANI5	✓	✓	✓	—	—
ANI6	✓	✓	✓	—	—
ANI7	✓	✓	✓	—	—
ANI16	✓	—	—	✓	✓
ANI17	✓	—	—	✓	✓
ANI18	✓	✓	✓	—	—
ANI19	✓	✓	✓	—	—
INTP0	✓	✓	✓	✓	✓
INTP1	✓	✓	✓	✓	—
INTP2	✓	✓	✓	✓	—
INTP3	✓	✓	✓	✓	✓
INTP4	✓	✓	✓	✓	—
INTP5	✓	✓	✓	✓	✓
INTP6	✓	✓	✓	✓	✓
INTP7	✓	—	—	—	—
INTP8	✓	✓	✓	✓	✓
INTP9	✓	✓	✓	✓	✓
INTP10	✓	—	—	—	—
INTP11	✓	—	—	—	—
KR0	✓	✓	✓	—	—
KR1	✓	✓	✓	—	—
KR2	✓	✓	✓	—	—
KR3	✓	✓	✓	—	—
KR4	✓	✓	—	—	—
KR5	✓	✓	—	—	—
KR6	✓	—	—	—	—
KR7	✓	—	—	—	—
$\overline{\text{RESET}}$	✓	✓	✓	✓	✓
RxD0	✓	✓	✓	✓	✓
RxD1	✓	✓	✓	✓	✓
RxD2	✓	✓	✓	✓	✓

(2/3)

Function Name	64-pin	48-pin	44-pin	32-pin	20-pin
TxD0	✓	✓	✓	✓	✓
TxD1	✓	✓	✓	✓	✓
TxD2	✓	✓	✓	✓	✓
RxDA0	✓	✓	✓	✓	✓
TxDA0	✓	✓	✓	✓	✓
CLKA0	✓	—	—	—	—
SCK00	✓	✓	✓	✓	✓
SCK01	✓	✓	—	—	—
SCK10	✓	—	—	—	—
SCK11	✓	✓	✓	✓	✓
SCK20	✓	✓	✓	✓	—
SCK21	✓	✓	✓	—	—
SCL00	✓	✓	✓	✓	✓
SCL01	✓	✓	—	—	—
SCL10	✓	—	—	—	—
SCL11	✓	✓	✓	✓	✓
SCL20	✓	✓	✓	✓	—
SCL21	✓	✓	✓	—	—
SDA00	✓	✓	✓	✓	✓
SDA01	✓	✓	—	—	—
SDA10	✓	—	—	—	—
SDA11	✓	✓	✓	✓	✓
SDA20	✓	✓	✓	✓	—
SDA21	✓	✓	✓	—	—
SI00	✓	✓	✓	✓	✓
SI01	✓	✓	—	—	—
SI10	✓	—	—	—	—
SI11	✓	✓	✓	✓	✓
SI20	✓	✓	✓	✓	—
SI21	✓	✓	✓	—	—
SO00	✓	✓	✓	✓	✓
SO01	✓	✓	—	—	—
SO10	✓	—	—	—	—
SO11	✓	✓	✓	✓	✓
SO20	✓	✓	✓	✓	—
SO21	✓	✓	✓	—	—
SCLA0	✓	✓	✓	✓	—
SDAA0	✓	✓	✓	✓	—
TI00	✓	✓	✓	✓	✓
TI01	✓	✓	✓	✓	✓

(3/3)

Function Name	64-pin	48-pin	44-pin	32-pin	20-pin
TI02	✓	✓	✓	✓	✓
TI03	✓	✓	✓	✓	—
TI04	✓	(✓)	(✓)	(✓)	—
TI05	✓	(✓)	(✓)	(✓)	✓
TI06	✓	(✓)	(✓)	(✓)	✓
TI07	✓	✓	✓	(✓)	✓
TO00	✓	✓	✓	✓	✓
TO01	✓	✓	✓	✓	✓
TO02	✓	✓	✓	✓	✓
TO03	✓	✓	✓	✓	—
TO04	✓	(✓)	(✓)	(✓)	—
TO05	✓	(✓)	(✓)	(✓)	✓
TO06	✓	(✓)	(✓)	(✓)	✓
TO07	✓	✓	✓	(✓)	✓
X1	✓	✓	✓	✓	✓
X2	✓	✓	✓	✓	✓
EXCLK	✓	✓	✓	✓	✓
V_{DD}	✓	✓	✓	✓	✓
EV_{DD0}	✓	—	—	—	—
AV_{REFP}	✓	✓	✓	✓	✓
AV_{REFM}	✓	✓	✓	✓	✓
V_{SS}	✓	✓	✓	✓	✓
EV_{SS0}	✓	—	—	—	—
TOOLRxD	✓	✓	✓	✓	✓
TOOLTxD	✓	✓	✓	✓	✓
TOOL0	✓	✓	✓	✓	✓

Remark Parentheses indicate that the pin can be assigned via settings in the peripheral I/O redirection register (PIORx).

2.2.2 Description of pin functions

(1/2)

Function Name	I/O	Function
ANI0 to ANI7, ANI16 to ANI19	Input	Analog voltage inputs for the A/D converter (see **Figure 10 - 45 Connections of VDD, AVREFP, and Analog Input Pins**)
INTP0 to INTP11	Input	External interrupt request input pins for which the valid edge (rising edge, falling edge, or both rising and falling edges) can be specified.
KR0 to KR7	Input	Key interrupt inputs
REGC	—	Pin for connecting a regulator output stabilization capacitor for internal operation. Connect this pin to Vss via a capacitor (0.47 to 1 μF). Also, use a capacitor with good characteristics, since it is used to stabilize internal voltage.
$\overline{\text{RESET}}$	Input	This is the active-low system reset input pin.
RxD0 to RxD2	Input	Serial data input pins for serial interfaces UART0 to UART2
TxD0 to TxD2	Output	Serial data output pins for serial interfaces UART0 to UART2
RxDA0	Input	Serial data input pin for serial interface UARTA0
TxDA0	Output	Serial data output pin for serial interface UARTA0
CLKA0	Output	Clock output pin for serial interface UARTA0
SCK00, SCK01, SCK10, SCK11, SCK20, SCK21	I/O	Serial clock I/O pins for serial interfaces CSI00, CSI01, CSI10, CSI11, CSI20, and CSI21
SCL00, SCL01, SCL10, SCL11, SCL20, SCL21	Output	Serial clock output pins for serial interfaces IIC00, IIC01, IIC10, IIC11, IIC20, and IIC21
SDA00, SDA01, SDA10, SDA11, SDA20, SDA21	I/O	Serial data I/O pins for serial interfaces IIC00, IIC01, IIC10, IIC11, IIC20, and IIC21
SI00, SI01, SI10, SI11, SI20, SI21	Input	Serial data input pins for serial interfaces CSI00, CSI01, CSI10, CSI11, CSI20, and CSI21
SO00, SO01, SO10, SO11, SO20, SO21	Output	Serial data output pins for serial interfaces CSI00, CSI01, CSI10, CSI11, CSI20, and CSI21
SCLA0	I/O	Clock I/O pin for serial interface IICA0
SDAA0	I/O	Serial data I/O pin for serial interface IICA0
TI00 to TI07	Input	Pins for inputting an external counting clock/capture trigger to 16-bit timers 00 to 07
TO00 to TO07	Output	Timer output pins for 16-bit timers 00 to 07
X1, X2	—	Resonator connection for the main system clock
EXCLK	Input	External clock input for the main system clock
VDD	—	<20- to 48-pin products> Positive power supply for all pins <64-pin products> Positive power supply for P20 to P27, P121 to P125, P137, and pins in use for functions other than port functions
EVDD0	—	Positive power supply for port pins other than P20 to P27, P121 to P125, and P137
AVREFP	Input	Positive reference voltage input for the A/D converter
AVREFM	Input	Negative reference voltage input for the A/D converter
Vss	—	<20- to 48-pin products> Ground voltage for all pins <64-pin products> Ground voltage for P20 to P27, P121 to P125, P137, and pins in use for functions other than port functions
EVSS0	—	Ground voltage for port pins other than P20 to P27, P121 to P125, and P137
TOOLRxD	Input	UART reception pin for the external device connection used during flash memory programming

(2/2)

Function Name	I/O	Function
TOOLTxD	Output	UART transmission pin for the external device connection used during flash memory programming
TOOL0	I/O	Data I/O for flash memory programmer or debugger

Caution The relationship between the voltage on P40/TOOL0 and the operating mode after release from the reset state is as follows.

P40/TOOL0	Operating Mode
EV_{DD}	**Normal operating mode**
0 V	**Flash memory programming mode**

For details, see 24.4 Programming Method.

Remark Use bypass capacitors (about 0.1 μF) as noise and latch up countermeasures with relatively thick wires at the shortest distance to V_{DD} to V_{SS} and EV_{DD0} to EV_{SS0} lines.

2.3 Connection of Unused Pins

Table 2 - 2 shows the connections of unused pins.

Remark The pins mounted depend on the product. Refer to **1.3 Pin Configuration (Top View)** and **2.1 Functions of Port Pins**.

Table 2 - 2 Connections of Unused Pins

Pin Name	I/O	Recommended Connection of Unused Pins
P00 to P06	I/O	Input: Independently connect the pins to EV_{DD0} or EV_{SS0} via resistors. Output: Leave the pins open-circuit.
P10 to P17		
P20 to P27		Input: Independently connect the pins to V_{DD} or V_{SS} via resistors. Output: Leave the pins open-circuit.
P30, P31, P33, P34		Input: Independently connect the pins to EV_{DD0} or EV_{SS0} via resistors. Output: Leave the pins open-circuit.
P40/TOOL0		Input: Independently connect the pin to EV_{DD0} via a resistor or leave the pin open-circuit. Output: Leave the pin open-circuit.
P41 to P43		Input: Independently connect the pins to EV_{DD0} or EV_{SS0} via resistors. Output: Leave the pins open-circuit.
P50 to P55		
P60, P61		Input: Independently connect the pins to EV_{DD0} or EV_{SS0} via resistors. Output: Set the port's output latch to 0 and leave the pins open-circuit, or set the port's output latch to 1 and independently connect the pins to EV_{DD0} or EV_{SS0} via resistors.
P70 to P77		Input: Independently connect the pins to EV_{DD0} or EV_{SS0} via resistors. Output: Leave the pins open-circuit.
P120		
P121, P122		Input: Independently connect the pins to V_{DD} or V_{SS} via resistors. Output: Leave the pins open-circuit.
P123, P124		Input: Independently connect the pins to V_{DD} or V_{SS} via resistors. Output: Leave the pins open-circuit.
P125/$\overline{\text{RESET}}$		PORTSELB of the user option byte (000C1H) = 0: Input: Independently connect the pin to V_{DD} via a resistor. Output: Leave the pin open-circuit. PORTSELB of the user option byte (000C1H) = 1: Leave the pin open-circuit or connect the pin to V_{DD}.
P130	Output	Leave the pin open-circuit.
P137	Input	Set the PDIDIS137 bit in the port digital input disable register (PDIDIS13) to 1, and leave the pin open-circuit. Alternatively, provide the pin with an independent connection to V_{DD} or V_{SS} via a resistor.
P140, P141, P146, P147	I/O	Input: Independently connect the pins to EV_{DD0} or EV_{SS0} via resistors. Output: Leave the pins open-circuit.
REGC	—	Connect the pin to V_{SS} via a capacitor (0.47 to 1 μF).

Remark For products that do not have an EV_{DD0} or EV_{SS0}, read EV_{DD0} as V_{DD}, and EV_{SS0} as V_{SS}.

2.4 Block Diagrams of Pins

Figures 2 - 1 to **2 - 14** show the block diagrams of the pins described in **2.1.1 20-pin products** to **2.1.5 64-pin products**.

Figure 2 - 1 Pin Block Diagram for Pin Type 1-1-1

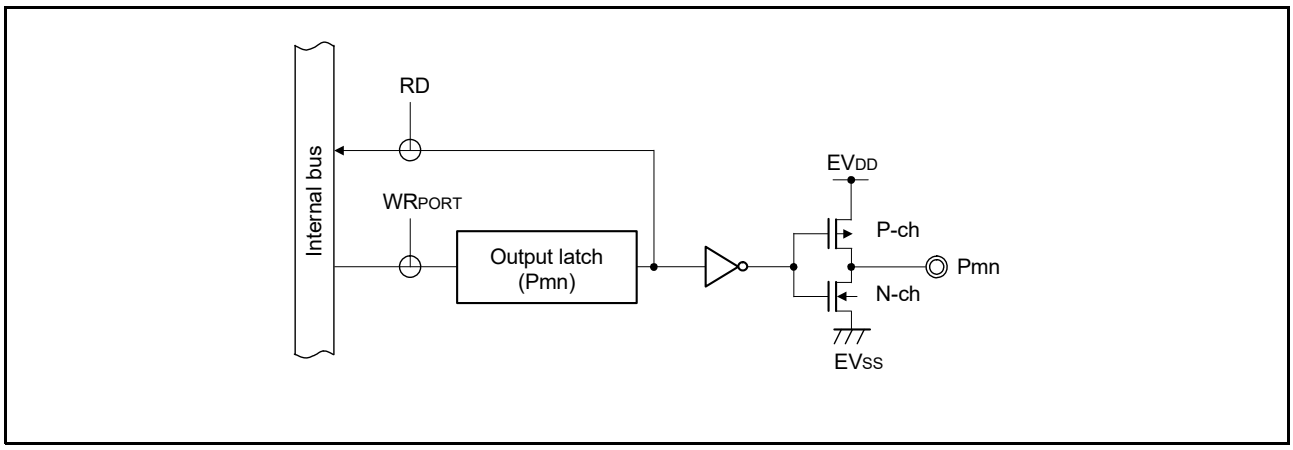
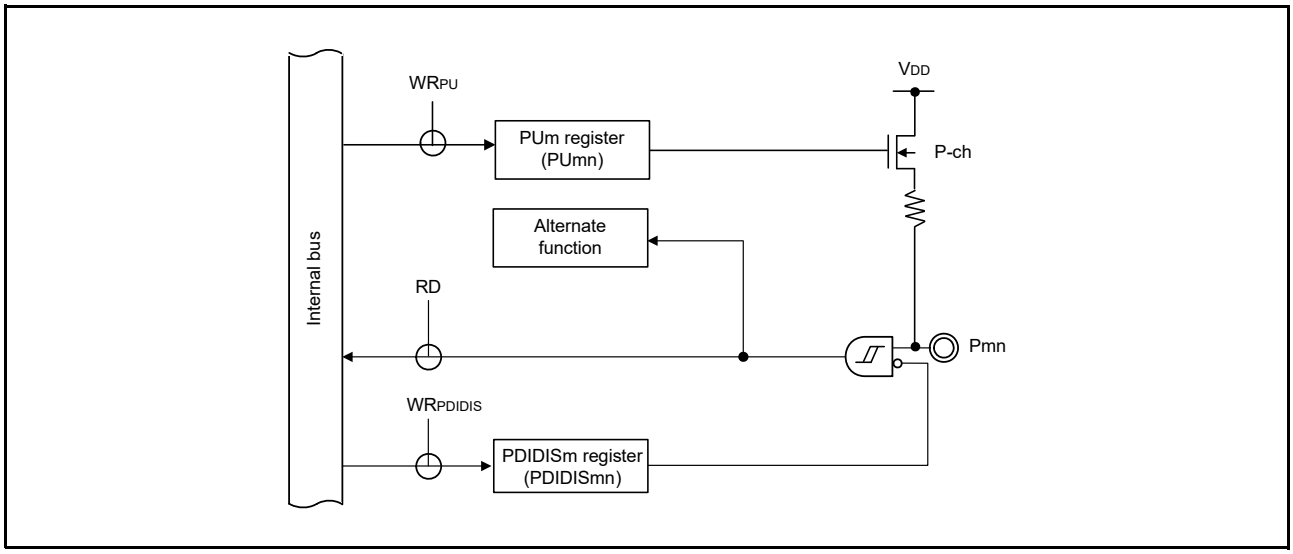
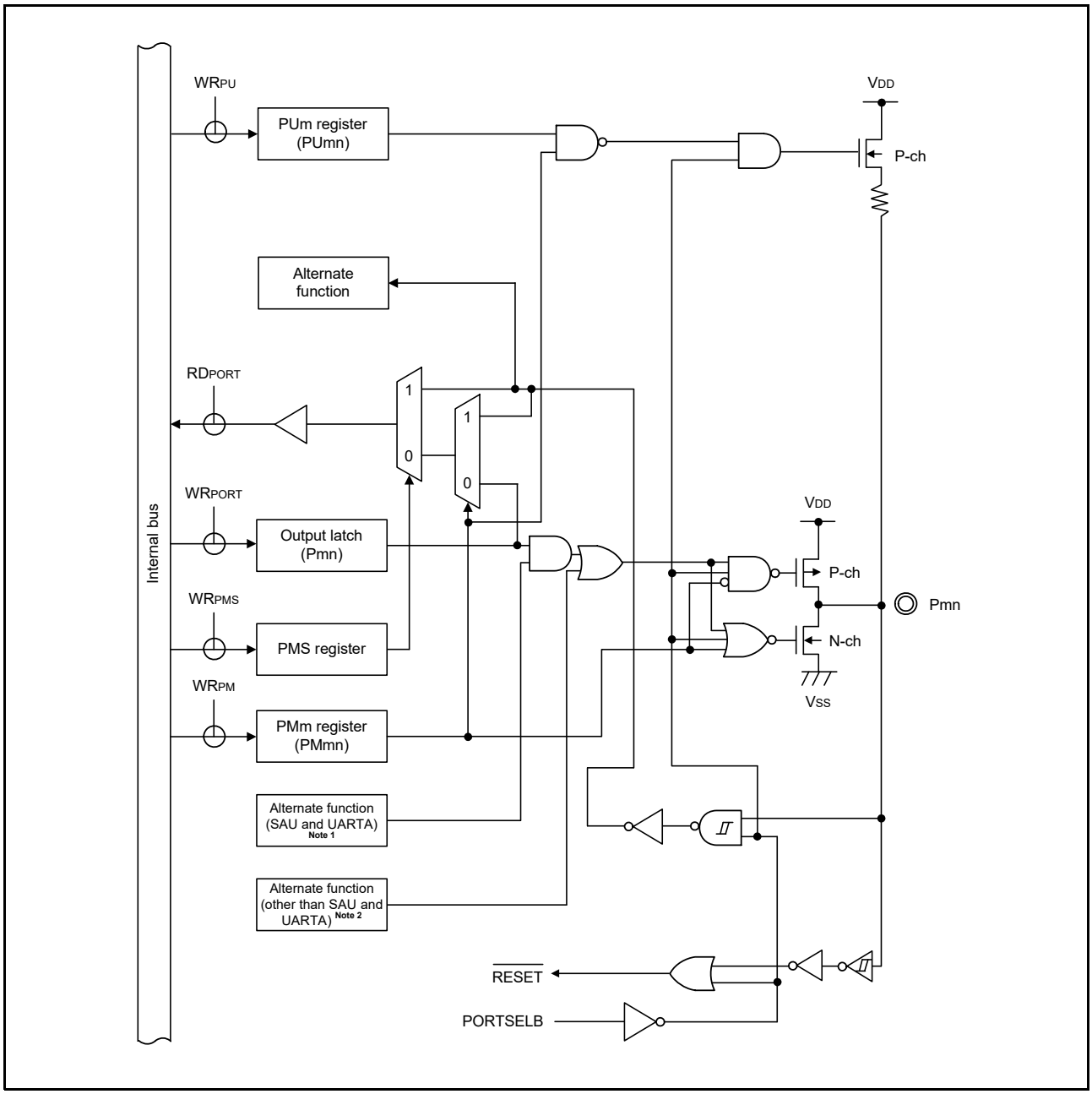


Figure 2 - 2 Pin Block Diagram for Pin Type 3-1-2



Remark For alternate functions, see **2.1 Functions of Port Pins**.

Figure 2 - 3 Pin Block Diagram for Pin Type 3-2-2



Note 1. This excludes the clock output from the UARTA.

Note 2. This includes the clock output from the UARTA.

Remark 1. For alternate functions, see **2.1 Functions of Port Pins**.

Remark 2. SAU: Serial array unit

Figure 2 - 4 Pin Block Diagram for Pin Type 4-3-5

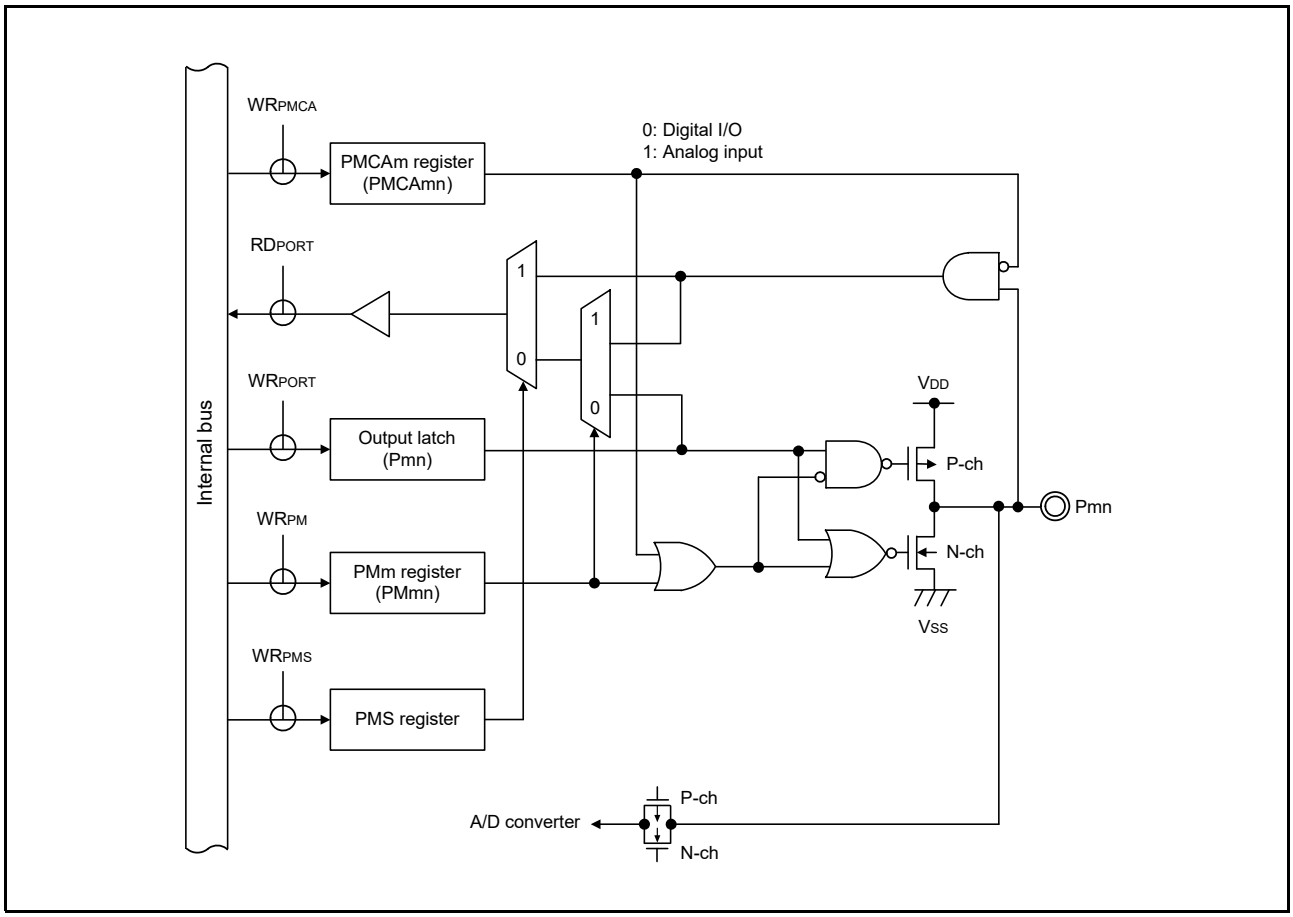
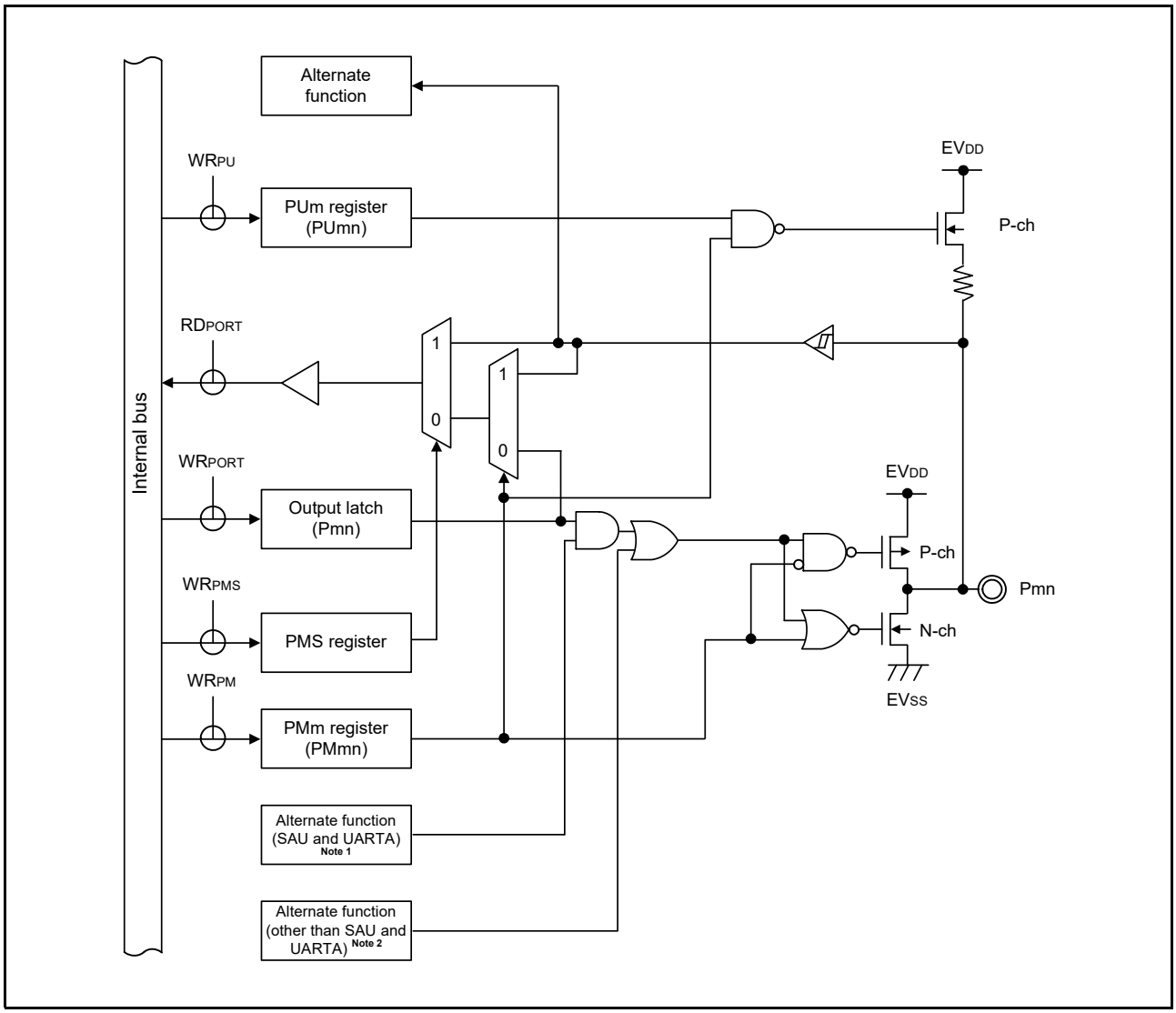


Figure 2 - 5 Pin Block Diagram for Pin Type 7-1-3



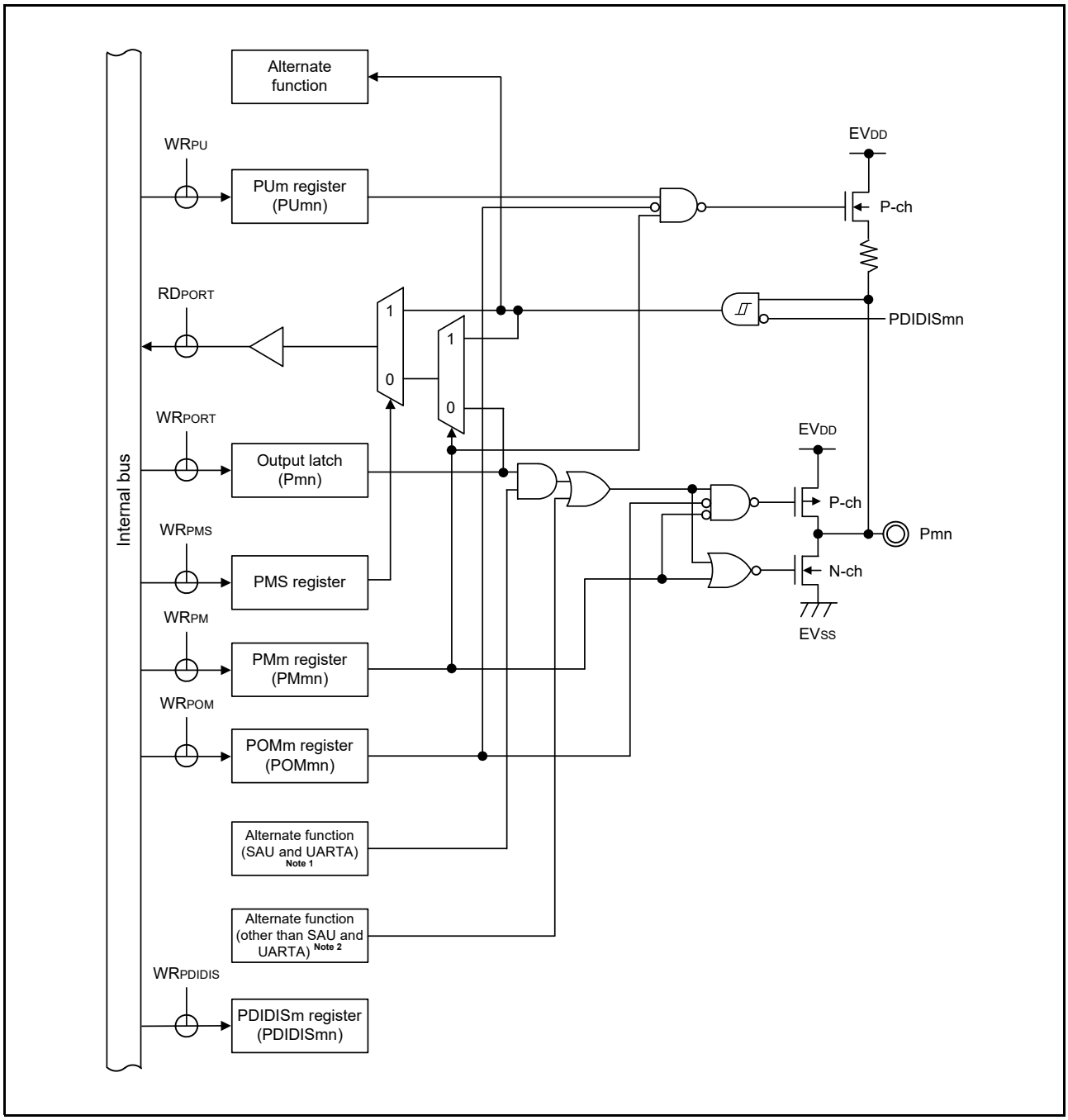
Note 1. This excludes the clock output from the UARTA.

Note 2. This includes the clock output from the UARTA.

Remark 1. For alternate functions, see **2.1 Functions of Port Pins**.

Remark 2. SAU: Serial array unit

Figure 2 - 6 Pin Block Diagram for Pin Type 7-1-11



Note 1. This excludes the clock output from the UARTA.

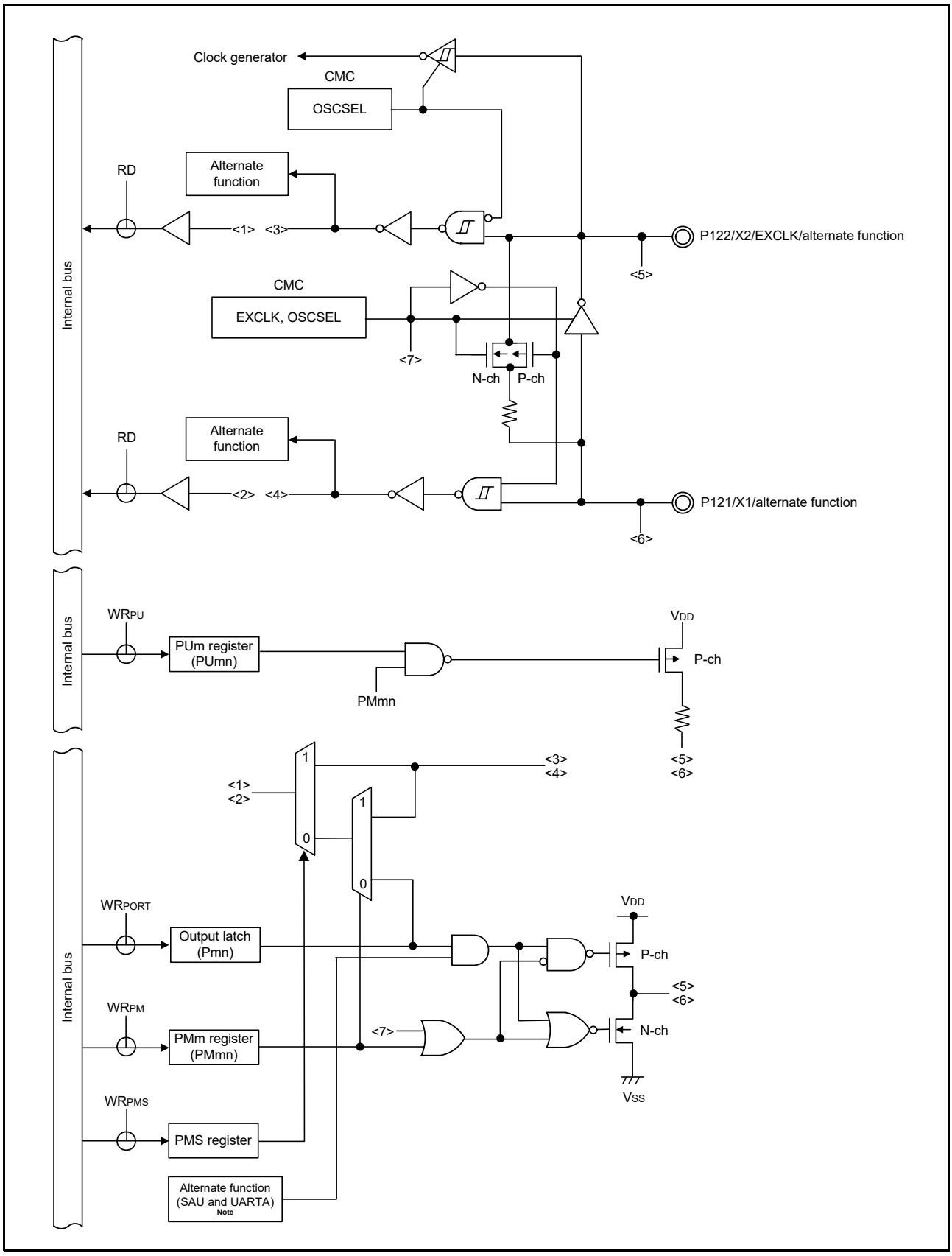
Note 2. This includes the clock output from the UARTA.

Caution **The input buffer is enabled even if the type 7-1-11 pin is operating as an output when the N-ch open-drain output mode is selected by the corresponding bit in the port output mode register (POMm). This may lead to a through current flowing through the type 7-1-11 pin when the voltage level on this pin is intermediate. However, setting the corresponding bit of the given PDIDISm register to 1 prevents the flow of a through current.**

Remark 1. For alternate functions, see **2.1 Functions of Port Pins**.

Remark 2. SAU: Serial array unit

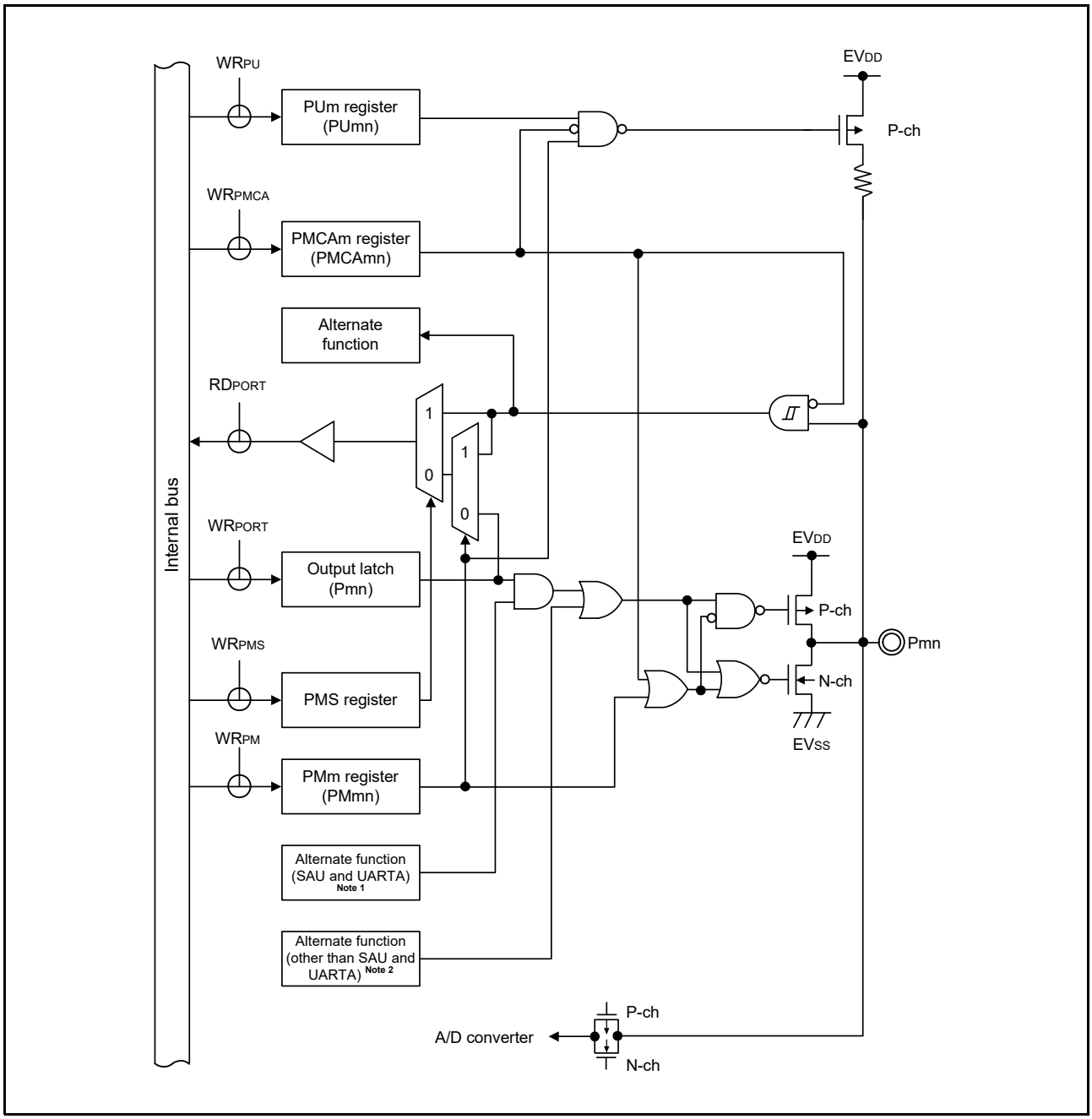
Figure 2 - 7 Pin Block Diagram for Pin Type 7-2-3



Note This excludes the clock output from the UARTA.

Remark For alternate functions, see **2.1 Functions of Port Pins**.

Figure 2 - 8 Pin Block Diagram for Pin Type 7-3-3



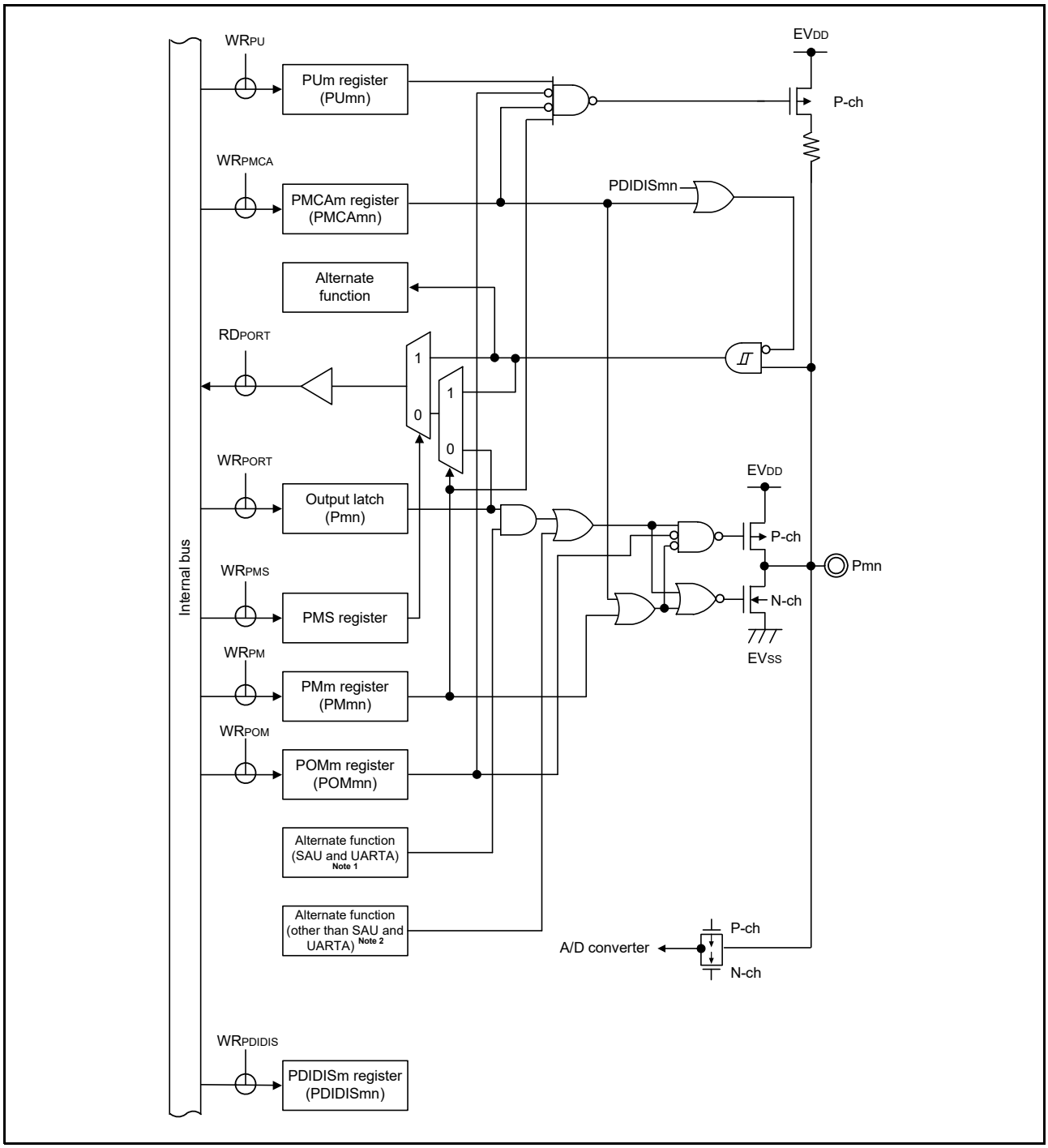
Note 1. This excludes the clock output from the UARTA.

Note 2. This includes the clock output from the UARTA.

Remark 1. For alternate functions, see **2.1 Functions of Port Pins**.

Remark 2. SAU: Serial array unit

Figure 2 - 9 Pin Block Diagram for Pin Type 7-3-5



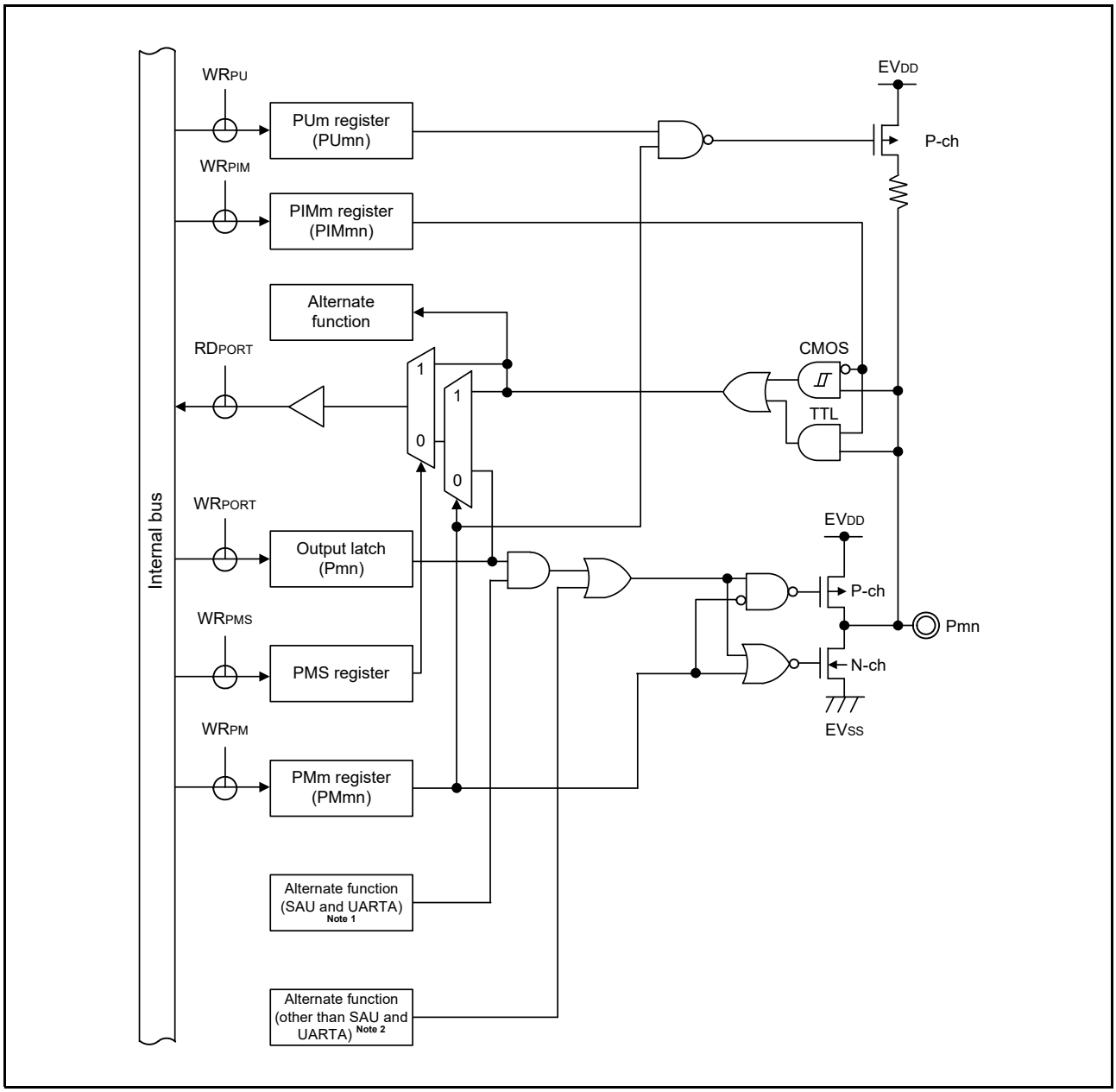
Note 1. This excludes the clock output from the UARTA.

Note 2. This includes the clock output from the UARTA.

Remark 1. For alternate functions, see **2.1 Functions of Port Pins**.

Remark 2. SAU: Serial array unit

Figure 2 - 10 Pin Block Diagram for Pin Type 8-1-3



Note 1. This excludes the clock output from the UARTA.

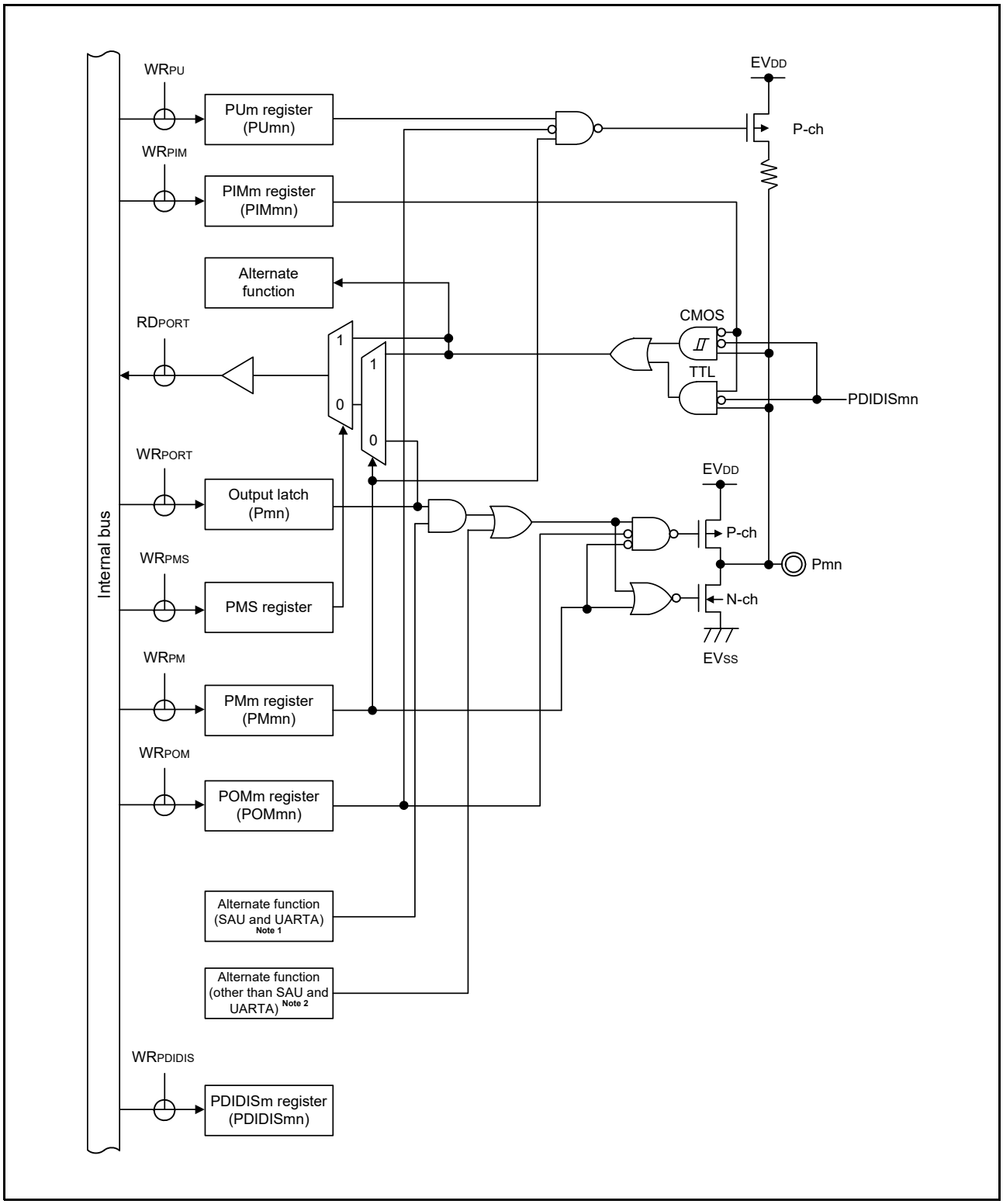
Note 2. This includes the clock output from the UARTA.

Caution **When the type 8-1-3 pin is set to TTL input buffer by the corresponding bit in the port input mode register (PIMm) and is driven high, a through current may flow through the type 8-1-3 pin due to the configuration of the TTL input buffer. Drive the type 8-1-3 pin low to prevent the through current.**

Remark 1. For alternate functions, see **2.1 Functions of Port Pins**.

Remark 2. SAU: Serial array unit

Figure 2 - 11 Pin Block Diagram for Pin Type 8-1-10



Note 1. This excludes the clock output from the UARTA.

Note 2. This includes the clock output from the UARTA.

(Cautions and Remarks are listed on the next page.)

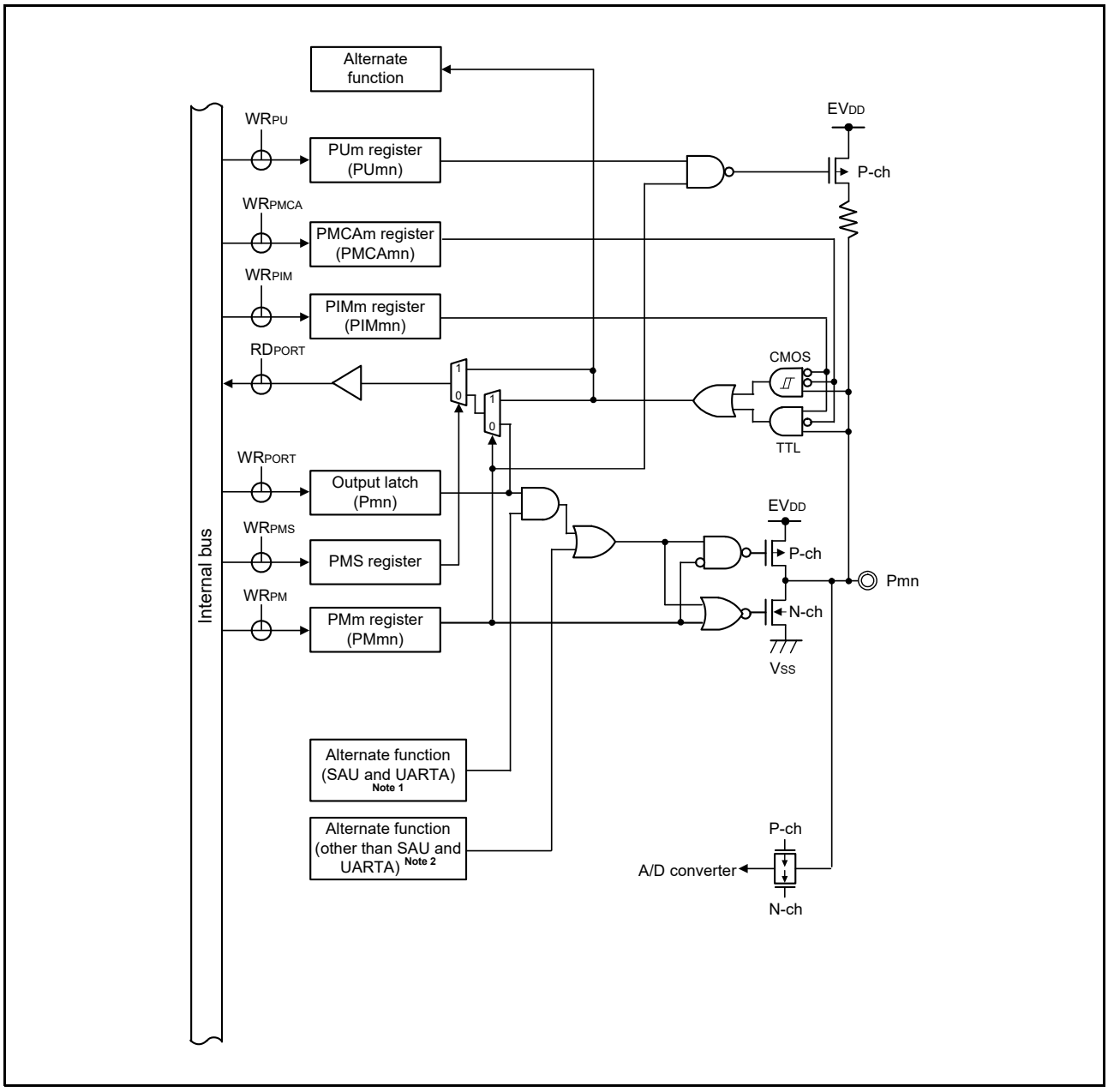
Caution 1. **The input buffer is enabled even if the type 8-1-10 pin is operating as an output when the N-ch open-drain output mode is selected by the corresponding bit in the port output mode register (POMm). This may lead to a through current flowing through the type 8-1-10 pin when the voltage level on this pin is intermediate. However, setting the corresponding bit of the given PDIDISm register to 1 prevents the flow of a through current.**

Caution 2. **When the type 8-1-10 pin is set to TTL input buffer by the corresponding bit in the port input mode register (PIMm) and is driven high, a through current may flow through the type 8-1-10 pin due to the configuration of the TTL input buffer. Drive the type 8-1-10 pin low to prevent the through current.**

Remark 1. For alternate functions, see **2.1 Functions of Port Pins**.

Remark 2. SAU: Serial array unit

Figure 2 - 12 Pin Block Diagram for Pin Type 8-3-3



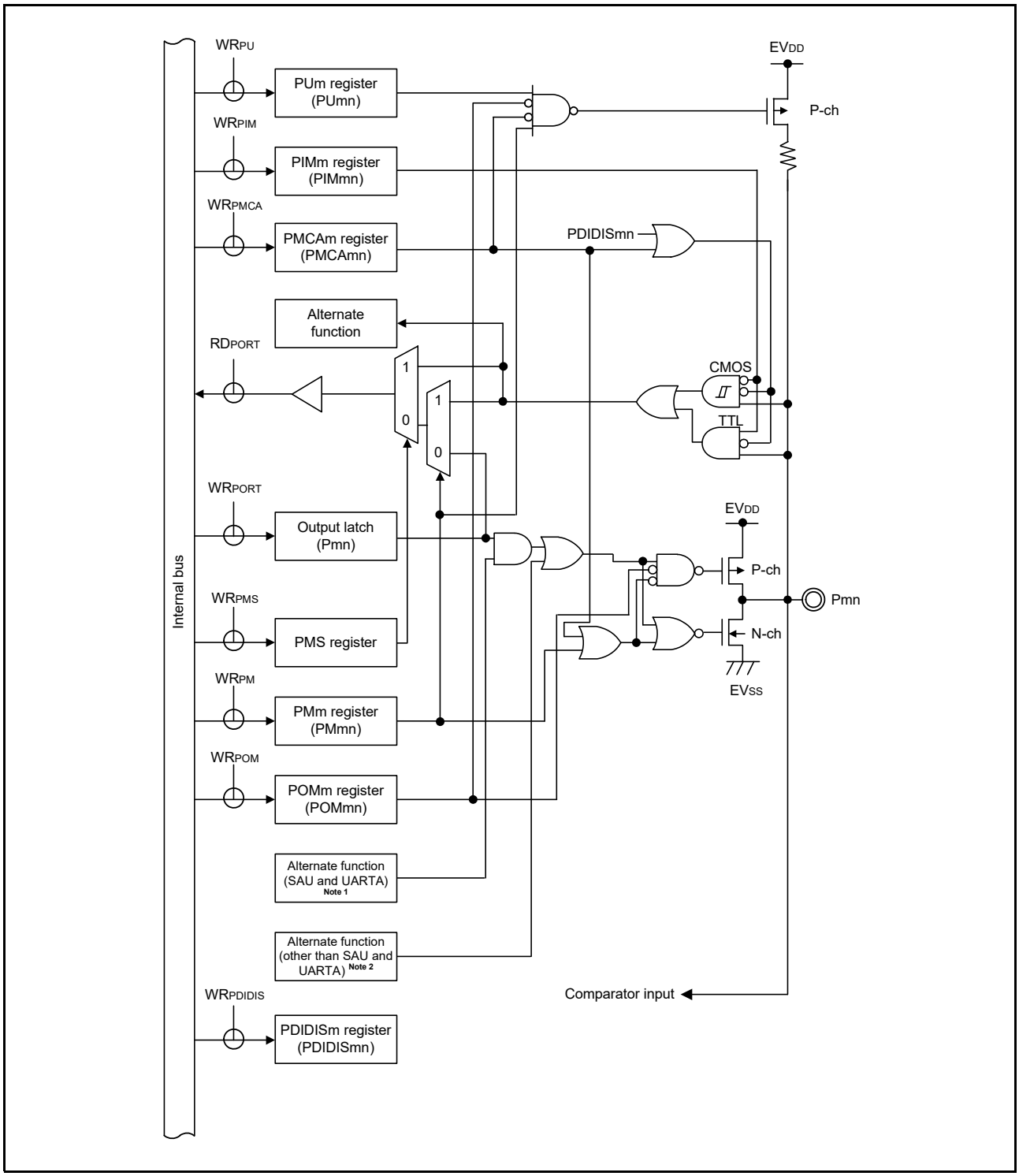
Note 1. This excludes the clock output from the UARTA.

Note 2. This includes the clock output from the UARTA.

Remark 1. For alternate functions, see **2.1 Functions of Port Pins**.

Remark 2. SAU: Serial array unit

Figure 2 - 13 Pin Block Diagram for Pin Type 8-3-6



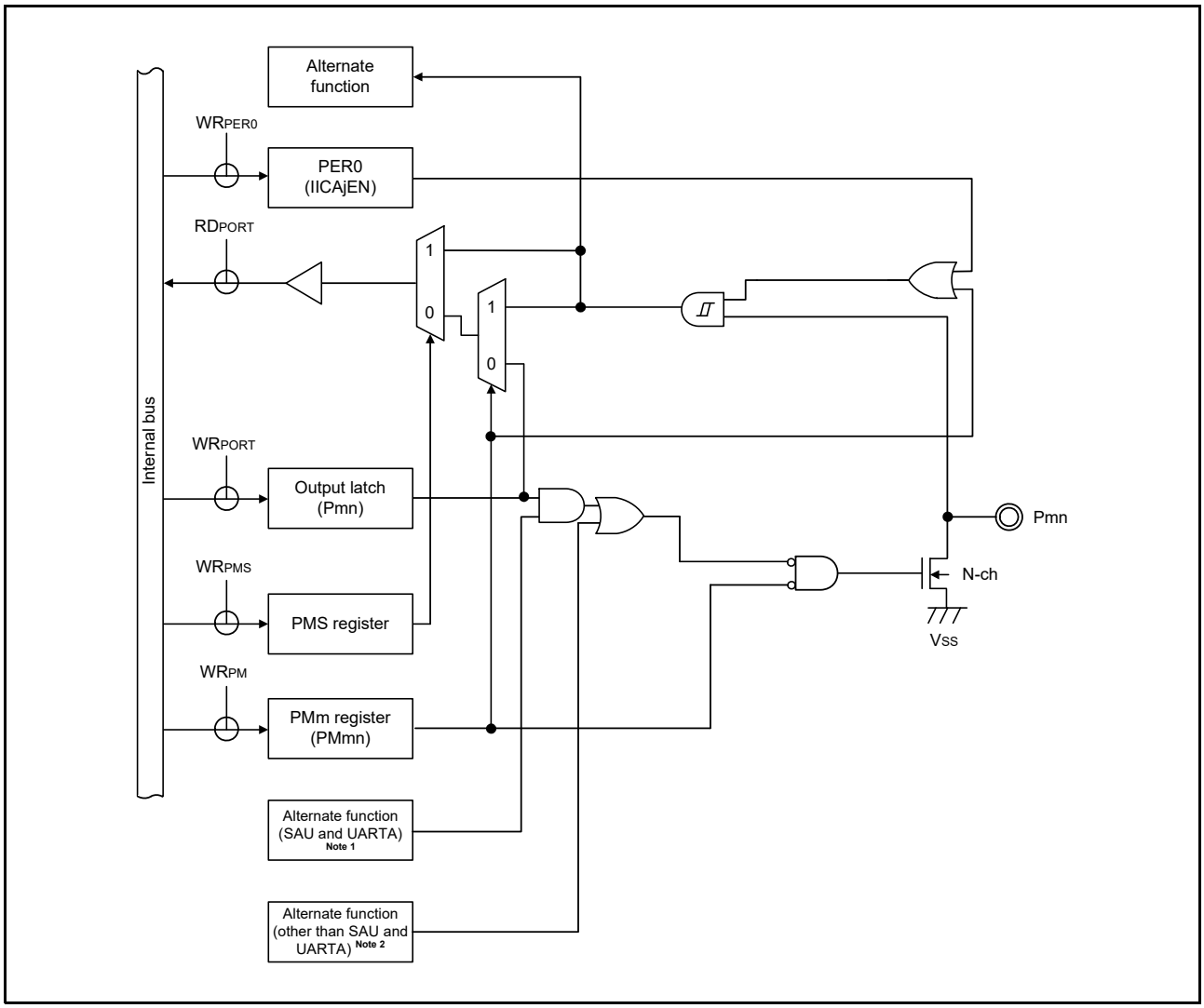
Note 1. This excludes the clock output from the UARTA.

Note 2. This includes the clock output from the UARTA.

Remark 1. For alternate functions, see **2.1 Functions of Port Pins**.

Remark 2. SAU: Serial array unit

Figure 2 - 14 Pin Block Diagram for Pin Type 12-1-3



Note 1. This excludes the clock output from the UARTA.

Note 2. This includes the clock output from the UARTA.

Caution **The input buffer is enabled when using the IICA with the IICAjEN bit in the PER0 register being set to 1. This may lead to a through current flowing through the type 12-1-3 pin when the voltage level on this pin is intermediate.**

Remark 1. For alternate functions, see **2.1 Functions of Port Pins**.

Remark 2. SAU: Serial array unit

Section 3 CPU Architecture

This MCU features the RL78-S3 CPU core. The CPU core in the RL78-S3 employs a Harvard architecture which has an independent instruction-fetching bus, address bus and data bus. In addition, through the adoption of three-stage pipeline control over fetching, decoding, and memory access, the operation efficiency is markedly improved over that of the previous CPU core. The CPU core features high performance and highly functional instruction processing, and is suitable for use in various applications that require high-speed and high-functionality processing.

- 3-stage pipeline CISC architecture
- Address space: 1 Mbyte
- Minimum instruction execution time: One instruction per clock cycle
- General-purpose registers: Eight 8-bit registers
- Type of instruction: 81

The following multiply/divide instructions are available only in the RL78-S3 CPU core.
MULHU (unsigned 16-bit multiplication)
MULH (signed 16-bit multiplication)
DIVHU (unsigned 16-bit division)
DIVWU (unsigned 32-bit division)
MACHU (unsigned multiplication/accumulation (16 bits × 16 bits) + 32 bits)
MACH (signed multiplication/accumulation (16 bits × 16 bits) + 32 bits)

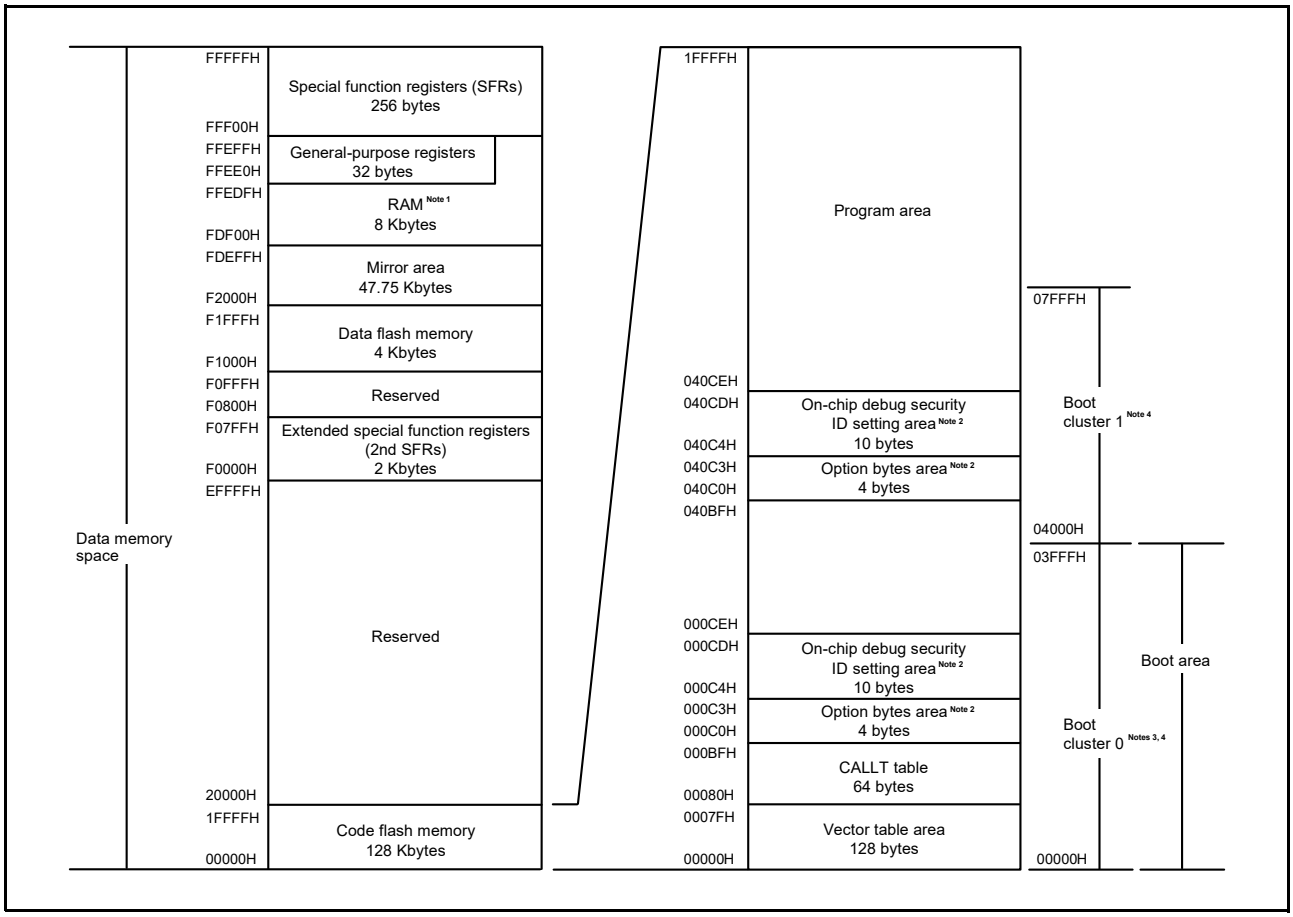
- Data allocation: Little endian

This MCU supports an OCD trace function.

3.1 Memory Space

A 1-Mbyte address space is accessible by this MCU. **Figure 3 - 1** and **Figure 3 - 2** show the memory maps.

Figure 3 - 1 Memory Map (R7F103GxG (x = 6, B, F, G, L))



Note 1. Instructions can be executed from the RAM area excluding the general-purpose register area.

Note 2. When boot swapping is not to be used, that is, when the value of the BTFLG bit in the FLSEC register is 1, set the option bytes to 000C0H to 000C3H, and the on-chip debug security IDs to 000C4H to 000CDH.
When boot swapping is to be used or the value of the BTFLG bit in the FLSEC register is 0, set the option bytes to 000C0H to 000C3H and 040C0H to 040C3H, and the on-chip debug security IDs to 000C4H to 000CDH and 040C4H to 040CDH.

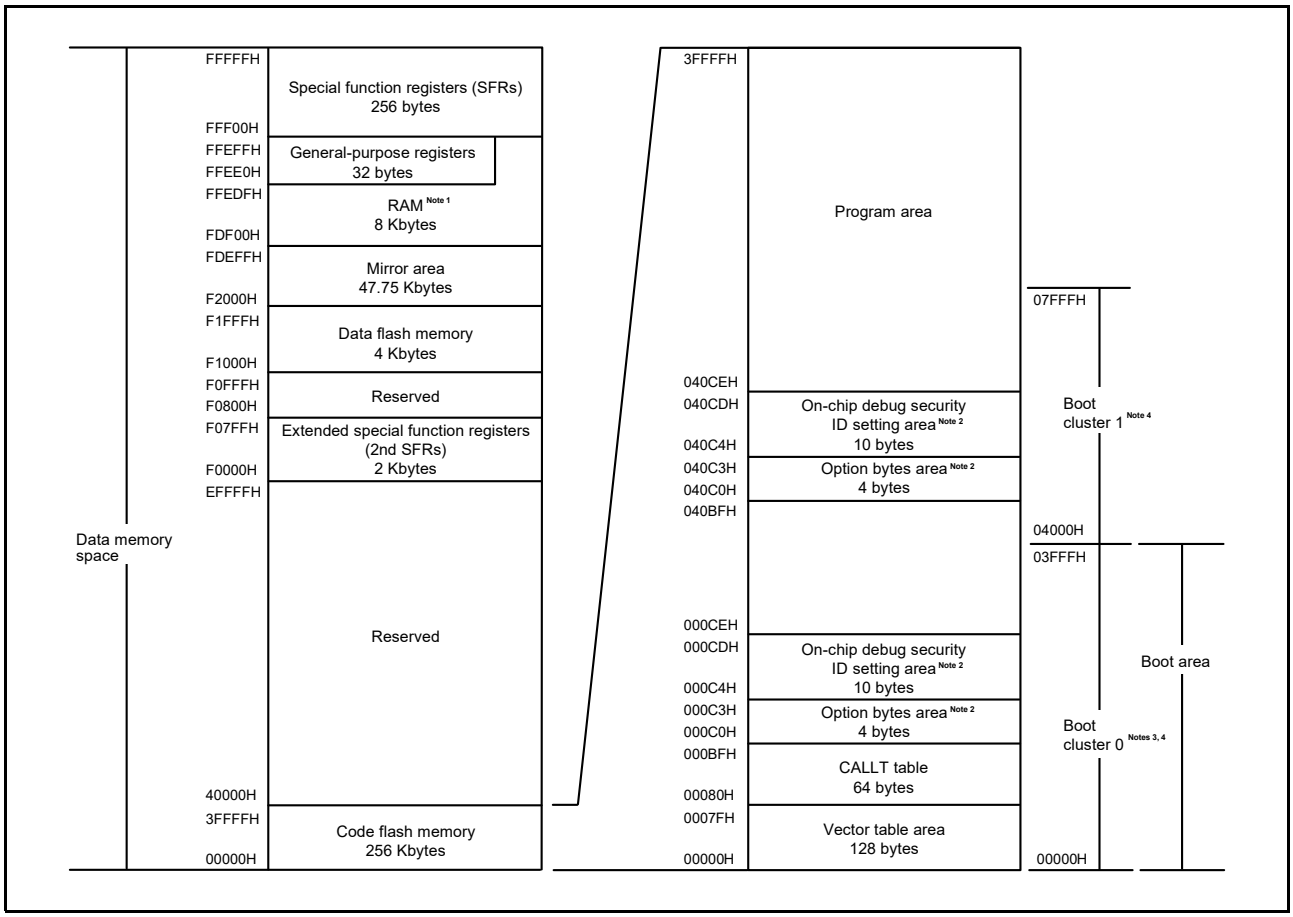
Note 3. Rewriting of the boot area can be prohibited by a security setting. See **24.9 Security Settings**.

Note 4. Boot cluster 0 is selected as the boot area at the time of shipment. When boot swapping is applied, the boot area is swapped between boot cluster 0 and boot cluster 1. See **24.7 Boot Swap Function**.

Caution **While RAM parity error resets are enabled (RPERDIS = 0), be sure to initialize RAM areas where data access is to proceed and the RAM area + 10 bytes when instructions are fetched from RAM areas, respectively.**
RAM parity error resets become enabled (RPERDIS = 0) following a reset. For details, see 20.3.4 RAM parity error detection.

Remark The code flash memory area is divided into blocks, with each block being 2 Kbytes. For the correspondence between addresses and block numbers, see **Table 3 - 1 Correspondence between Addresses and Block Numbers in Flash Memory**.

Figure 3 - 2 Memory Map (R7F103GxJ (x = 6, B, F, G, L))



Note 1. Instructions can be executed from the RAM area excluding the general-purpose register area.

Note 2. When boot swapping is not to be used, that is, when the value of the BTFLG bit in the FLSEC register is 1, set the option bytes to 000C0H to 000C3H, and the on-chip debug security IDs to 000C4H to 000CDH.
When boot swapping is to be used or the value of the BTFLG bit in the FLSEC register is 0, set the option bytes to 000C0H to 000C3H and 040C0H to 040C3H, and the on-chip debug security IDs to 000C4H to 000CDH and 040C4H to 040CDH.

Note 3. Rewriting of the boot area can be prohibited by a security setting. See **24.9 Security Settings**.

Note 4. Boot cluster 0 is selected as the boot area at the time of shipment. When boot swapping is applied, the boot area is swapped between boot cluster 0 and boot cluster 1. See **24.7 Boot Swap Function**.

Caution **While RAM parity error resets are enabled (RPERDIS = 0), be sure to initialize RAM areas where data access is to proceed and the RAM area + 10 bytes when instructions are fetched from RAM areas, respectively.**
RAM parity error resets become enabled (RPERDIS = 0) following a reset. For details, see 20.3.4 RAM parity error detection.

Remark The code flash memory area is divided into blocks, with each block being 2 Kbytes. For the correspondence between addresses and block numbers, see **Table 3 - 1 Correspondence between Addresses and Block Numbers in Flash Memory**.

Table 3 - 1 shows the correspondence between the addresses and block numbers in the flash memory.

Table 3 - 1 Correspondence between Addresses and Block Numbers in Flash Memory

Address	Block Number	Address	Block Number	Address	Block Number	Address	Block Number
00000H to 007FFH	000H	10000H to 107FFH	020H	20000H to 207FFH	040H	30000H to 307FFH	060H
00800H to 00FFFH	001H	10800H to 10FFFH	021H	20800H to 20FFFH	041H	30800H to 30FFFH	061H
01000H to 017FFH	002H	11000H to 117FFH	022H	21000H to 217FFH	042H	31000H to 317FFH	062H
01800H to 01FFFH	003H	11800H to 11FFFH	023H	21800H to 21FFFH	043H	31800H to 31FFFH	063H
02000H to 027FFH	004H	12000H to 127FFH	024H	22000H to 227FFH	044H	32000H to 327FFH	064H
02800H to 02FFFH	005H	12800H to 12FFFH	025H	22800H to 22FFFH	045H	32800H to 32FFFH	065H
03000H to 037FFH	006H	13000H to 137FFH	026H	23000H to 237FFH	046H	33000H to 337FFH	066H
03800H to 03FFFH	007H	13800H to 13FFFH	027H	23800H to 23FFFH	047H	33800H to 33FFFH	067H
04000H to 047FFH	008H	14000H to 147FFH	028H	24000H to 247FFH	048H	34000H to 347FFH	068H
04800H to 04FFFH	009H	14800H to 14FFFH	029H	24800H to 24FFFH	049H	34800H to 34FFFH	069H
05000H to 057FFH	00AH	15000H to 157FFH	02AH	25000H to 257FFH	04AH	35000H to 357FFH	06AH
05800H to 05FFFH	00BH	15800H to 15FFFH	02BH	25800H to 25FFFH	04BH	35800H to 35FFFH	06BH
06000H to 067FFH	00CH	16000H to 167FFH	02CH	26000H to 267FFH	04CH	36000H to 367FFH	06CH
06800H to 06FFFH	00DH	16800H to 16FFFH	02DH	26800H to 26FFFH	04DH	36800H to 36FFFH	06DH
07000H to 077FFH	00EH	17000H to 177FFH	02EH	27000H to 277FFH	04EH	37000H to 377FFH	06EH
07800H to 07FFFH	00FH	17800H to 17FFFH	02FH	27800H to 27FFFH	04FH	37800H to 37FFFH	06FH
08000H to 087FFH	010H	18000H to 187FFH	030H	28000H to 287FFH	050H	38000H to 387FFH	070H
08800H to 08FFFH	011H	18800H to 18FFFH	031H	28800H to 28FFFH	051H	38800H to 38FFFH	071H
09000H to 097FFH	012H	19000H to 197FFH	032H	29000H to 297FFH	052H	39000H to 397FFH	072H
09800H to 09FFFH	013H	19800H to 19FFFH	033H	29800H to 29FFFH	053H	39800H to 39FFFH	073H
0A000H to 0A7FFH	014H	1A000H to 1A7FFH	034H	2A000H to 2A7FFH	054H	3A000H to 3A7FFH	074H
0A800H to 0AFFFH	015H	1A800H to 1AFFFH	035H	2A800H to 2AFFFH	055H	3A800H to 3AFFFH	075H
0B000H to 0B7FFH	016H	1B000H to 1B7FFH	036H	2B000H to 2B7FFH	056H	3B000H to 3B7FFH	076H
0B800H to 0BFFFH	017H	1B800H to 1BFFFH	037H	2B800H to 2BFFFH	057H	3B800H to 3BFFFH	077H
0C000H to 0C7FFH	018H	1C000H to 1C7FFH	038H	2C000H to 2C7FFH	058H	3C000H to 3C7FFH	078H
0C800H to 0CFFFH	019H	1C800H to 1CFFFH	039H	2C800H to 2CFFFH	059H	3C800H to 3CFFFH	079H
0D000H to 0D7FFH	01AH	1D000H to 1D7FFH	03AH	2D000H to 2D7FFH	05AH	3D000H to 3D7FFH	07AH
0D800H to 0DFFFH	01BH	1D800H to 1DFFFH	03BH	2D800H to 2DFFFH	05BH	3D800H to 3DFFFH	07BH
0E000H to 0E7FFH	01CH	1E000H to 1E7FFH	03CH	2E000H to 2E7FFH	05CH	3E000H to 3E7FFH	07CH
0E800H to 0EFFFH	01DH	1E800H to 1EFFFH	03DH	2E800H to 2EFFFH	05DH	3E800H to 3EFFFH	07DH
0F000H to 0F7FFH	01EH	1F000H to 1F7FFH	03EH	2F000H to 2F7FFH	05EH	3F000H to 3F7FFH	07EH
0F800H to 0FFFFH	01FH	1F800H to 1FFFFH	03FH	2F800H to 2FFFFH	05FH	3F800H to 3FFFFH	07FH

Remark R7F103GxG (x = 6, B, F, G, L): Block numbers 000H to 03FH
R7F103GxJ (x = 6, B, F, G, L): Block numbers 000H to 07FH

3.1.1 Internal program memory space

The internal program memory holds the program and table data. This MCU incorporates the internal ROM (flash memory) capacities listed in **Table 3 - 2**.

Table 3 - 2 Internal ROM Capacity

Part Number	Internal ROM	
	Structure	Capacity
R7F103GxG (x = 6, B, F, G, L)	Flash memory	131072 × 8 bits (00000H to 1FFFFH)
R7F103GxJ (x = 6, B, F, G, L)		262144 × 8 bits (00000H to 3FFFFH)

The internal program memory space is divided into the following areas.

1. Vector table area

   The 128-byte area 00000H to 0007FH is reserved as a vector table area. The program start addresses for branch upon reset or generation of each interrupt request are stored in the vector table area. Furthermore, the interrupt jump address is a 64 K address of 00000H to 0FFFFH, because the vector code is assumed to be 2 bytes. Of the 16-bit address, the lower 8 bits are stored at even addresses and the higher 8 bits are stored at odd addresses. To use the boot swap function, set a vector table also at 04000H to 0407FH. **Table 3 - 3** lists the vector table. "✓" indicates an interrupt source which is supported. "—" indicates an interrupt source which is not supported. The vector table address can be changed to an address in RAM when self-programming is to proceed. For details, see **24.6.2.18 Interrupt vector change registers 0 and 1 (FLSIVC0, FLSIVC1)**.

Table 3 - 3 Vector Table (1/2)

Vector Table Address	Interrupt Source	64-pin	48-pin	44-pin	32-pin	20-pin
00000H	RESET, POR, LVD, WDT, TRAP, IAW, RPE	✓	✓	✓	✓	✓
00004H	INTWDTI	✓	✓	✓	✓	✓
00006H	INTLVI	✓	✓	✓	✓	✓
00008H	INTP0	✓	✓	✓	✓	✓
0000AH	INTP1	✓	✓	✓	✓	—
0000CH	INTP2	✓	✓	✓	✓	—
0000EH	INTP3	✓	✓	✓	✓	✓
00010H	INTP4	✓	✓	✓	✓	—
00012H	INTP5	✓	✓	✓	✓	✓
00014H	INTST2/INTCSI20/INTIIC20	✓	✓	✓	✓	Note 1
00016H	INTSR2/INTCSI21/INTIIC21	✓	✓	✓	Note 2	Note 2
00018H	INTSRE2	✓	✓	✓	✓	✓
0001AH	—	—	—	—	—	—
0001CH	—	—	—	—	—	—
0001EH	INTST0/INTCSI00/INTIIC00	✓	✓	✓	✓	✓
00020H	INTTM00	✓	✓	✓	✓	✓
00022H	INTSRE0	✓	✓	✓	✓	✓
	INTTM01H	✓	✓	✓	✓	✓
00024H	INTST1/INTCSI10/INTIIC10	✓	Note 3	Note 3	Note 3	Note 3
00026H	INTSR1/INTCSI11/INTIIC11	✓	✓	✓	✓	✓
00028H	INTSRE1	✓	✓	✓	✓	✓
	INTTM03H	✓	✓	✓	✓	✓
0002AH	INTIICA0	✓	✓	✓	✓	—
0002CH	INTSR0/INTCSI01/INTIIC01	✓	✓	Note 4	Note 4	Note 4
0002EH	INTTM01	✓	✓	✓	✓	✓
00030H	INTTM02	✓	✓	✓	✓	✓
00032H	INTTM03	✓	✓	✓	✓	✓
00034H	INTAD	✓	✓	✓	✓	✓
00036H	—	—	—	—	—	—
00038H	INTITL	✓	✓	✓	✓	✓
0003AH	INTKR	✓	✓	✓	—	—
0003CH	—	—	—	—	—	—
0003EH	—	—	—	—	—	—
00040H	—	—	—	—	—	—
00042H	INTTM04	✓	✓	✓	✓	✓
00044H	INTTM05	✓	✓	✓	✓	✓
00046H	INTTM06	✓	✓	✓	✓	✓
00048H	INTTM07	✓	✓	✓	✓	✓

Table 3 - 3 Vector Table (2/2)

Vector Table Address	Interrupt Source	64-pin	48-pin	44-pin	32-pin	20-pin
0004AH	INTP6	✓	✓	✓	✓	✓
0004CH	INTP7	✓	—	—	—	—
0004EH	INTP8	✓	✓	✓	✓	✓
00050H	INTP9	✓	✓	✓	✓	✓
00052H	INTFL	✓	✓	✓	✓	✓
00054H	INTP10	✓	—	—	—	—
00056H	INTP11	✓	—	—	—	—
00058H	INTURE0	✓	✓	✓	✓	✓
0005AH	—	—	—	—	—	—
0005CH	—	—	—	—	—	—
0005EH	—	—	—	—	—	—
00060H	—	—	—	—	—	—
00062H	—	—	—	—	—	—
00064H	—	—	—	—	—	—
00066H	—	—	—	—	—	—
00068H	—	—	—	—	—	—
0006AH	INTUT0	✓	✓	✓	✓	✓
0006CH	INTUR0	✓	✓	✓	✓	✓
0006EH	—	—	—	—	—	—
00070H	—	—	—	—	—	—
00072H	—	—	—	—	—	—
00074H	—	—	—	—	—	—
00076H	—	—	—	—	—	—
00078H	—	—	—	—	—	—
0007AH	—	—	—	—	—	—
0007CH	—	—	—	—	—	—
0007EH	BRK	✓	✓	✓	✓	✓

Note 1. INTST2 is only present in this product.
Note 2. INTSR2 is only present in this product.
Note 3. INTST1 is only present in this product.
Note 4. INTSR0 is only present in this product.

2. CALLT instruction table area
   The 64-byte area 00080H to 000BFH can store the subroutine entry address of a 2-byte call instruction (CALLT). Set the subroutine entry address to a value in a range of 00000H to 0FFFFH (because an address code is 2 bytes). To use the boot swap function, set a CALLT instruction table also at 04080H to 040BFH.

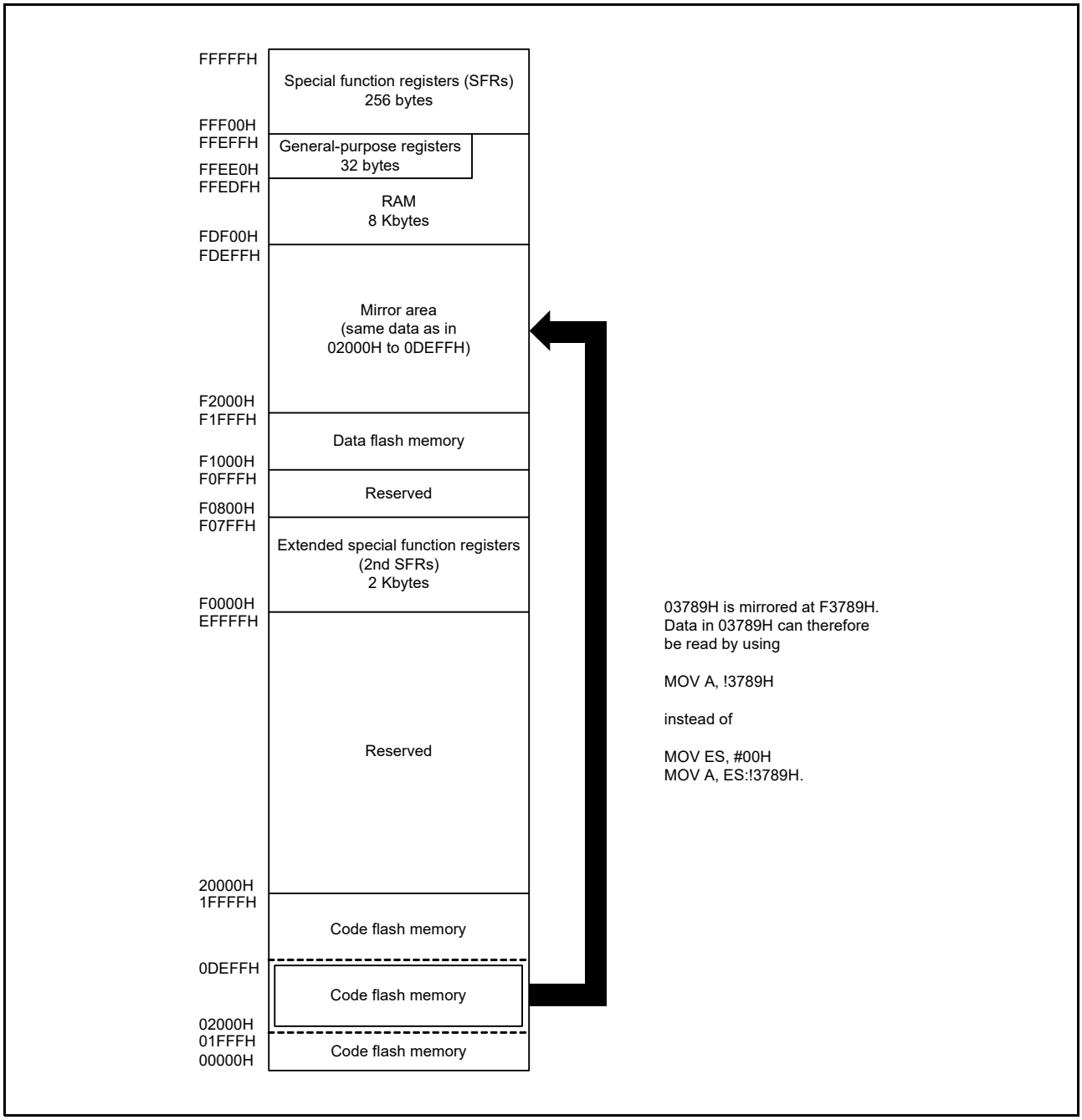
3. Option bytes area
   A 4-byte area of 000C0H to 000C3H can be used as an option bytes area. Set the option byte at 040C0H to 040C3H when boot swapping is to be used or the value of the BTFLG bit in the FLSEC register is 0. For details, see **Section 23 Option Bytes**.

4. On-chip debug security ID setting area
   A 10-byte area of 000C4H to 000CDH and 040C4H to 040CDH can be used as an on-chip debug security ID setting area. Set the 10-byte security ID for on-chip debugging at 000C4H to 000CDH when boot swapping is not to be used, that is, the value of the BTFLG bit in the FLSEC register is 1, and at both 000C4H to 000CDH and 040C4H to 040CDH when boot swapping is to be used or the value of the BTFLG bit in the FLSEC register is 0. For details, see **Section 25 On-chip Debugging**.

3.1.2 Mirror area

In this MCU, the code flash memory area from 00000H to 0FFFFH or 10000H to 1FFFFH is mirrored at F0000H to FFFFFH. The setting of the effective bit of the processor mode control register (PMC) determines which range of the code flash memory area is to be mirrored. By reading data from F0000H to FFFFFH, an instruction that does not have the ES register as an operand can be used, and thus the contents of the code flash memory can be read with the shorter code. However, the code flash memory area is not mirrored to the special function registers (SFRs), extended special function registers (2nd SFRs), RAM, data flash memory, and use prohibited areas. See **3.1 Memory Space** for the mirror area of each product. The mirror area can only be read and no instruction can be fetched from this area. The following show examples.

Example R7F103GxG (x = 6, B, F, G, L) (Flash memory: 128 Kbytes, RAM: 8 Kbytes)



The PMC register is described below.

- Processor mode control register (PMC)
  This register sets the flash memory space for mirroring to area from F0000H to FFFFFH. The PMC register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 3 - 3 Format of Processor Mode Control Register (PMC)

Address: FFFFEH
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	<0>
PMC	0	0	0	0	0	0	0	MAA

MAA	Selection of flash memory space for mirroring to area from F0000H to FFFFFH
0	00000H to 0FFFFH is mirrored to F0000H to FFFFFH
1	10000H to 1FFFFH is mirrored to F0000H to FFFFFH

Caution 1. After setting the PMC register, wait for at least one instruction and access the mirror area.
Caution 2. When boot swapping is executed while the MAA bit is 0, data at 02000H to 07FFFH are mirrored at F2000H to F7FFFH after boot swapping.

3.1.3 Internal data memory space

This MCU incorporates RAM with the capacities listed in **Table 3 - 4**.

Table 3 - 4 Internal RAM Capacity

Part Number	Internal RAM
R7F103GxG (x = 6, B, F, G, L)	8192 × 8 bits (FDF00H to FFEFFH)
R7F103GxJ (x = 6, B, F, G, L)	8192 × 8 bits (FDF00H to FFEFFH)

The internal RAM can be used as a data area and a program area where instructions are fetched (the space to which the general-purpose registers are allocated cannot be used for instruction fetching). Four general-purpose register banks consisting of eight 8-bit registers per bank are assigned to the 32-byte area of FFEE0H to FFEFFH of the internal RAM area. The internal RAM is used as stack memory.

Caution 1. The space to which the general-purpose registers are allocated (FFEE0H to FFEFFH) cannot be used for instruction fetching or the stack.

Caution 2. The use of the RAM area in the following products is prohibited when the trace function of on-chip debugging is in use.
R7F103GxG (x = 6, B, F, G, L): FE300H to FE6FFH
R7F103GxJ (x = 6, B, F, G, L): FE300H to FE6FFH

3.1.4 Special function register (SFR) area

On-chip peripheral hardware special function registers (SFRs) are allocated in the area FFF00H to FFFFFH (see **Table 3 - 5 List of Special Function Registers (SFRs)** in **3.2.4 Special function registers (SFRs)**).

Caution Do not access addresses to which SFRs are not assigned.

3.1.5 Extended special function register (2nd SFR: 2nd special function register) area

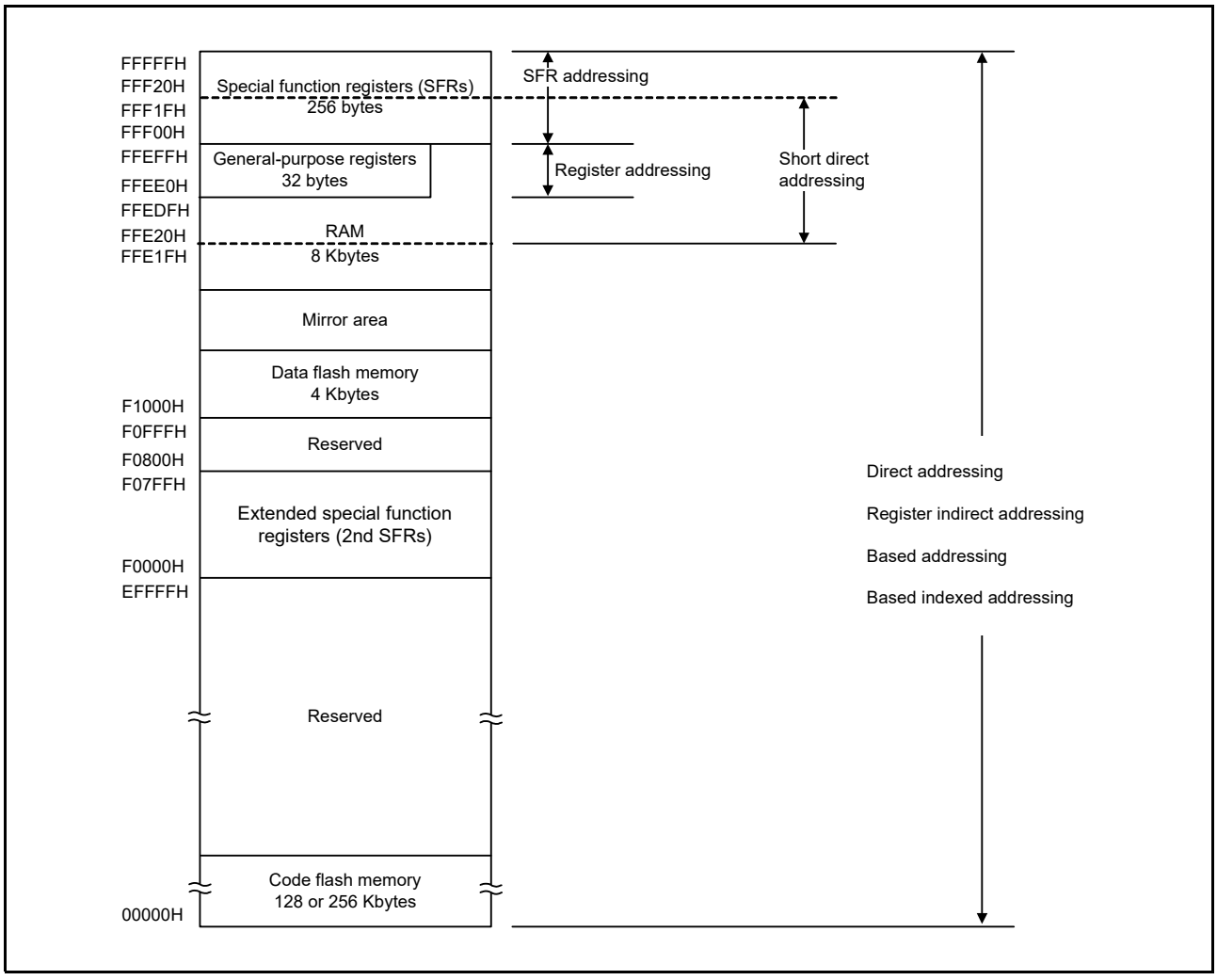
On-chip peripheral hardware special function registers (2nd SFRs) are allocated in the area F0000H to F07FFH (see **Table 3 - 6 List of Extended Special Function Registers (2nd SFRs)** in **3.2.5 Extended special function registers (2nd SFRs)**).

Caution Do not access addresses to which 2nd SFRs are not assigned.

3.1.6 Data memory addressing

Addressing refers to the method of specifying the addresses of next instructions to be executed or the addresses of the registers or memory locations relevant to the execution of the instructions. This MCU provides a wide range of addressing modes to ease programming. For areas of memory containing data in particular, special addressing methods designed to apply the functions of special function registers (SFRs) and general-purpose registers are available for use. **Figure 3 - 4** shows the correspondence between data memory and types of addressing. For details of each type of addressing, see **3.4 Types of Addressing for Processing Data Addresses**.

Figure 3 - 4 Correspondence Between Data Memory and Type of Addressing



3.2 Processor Registers

Products of this group incorporate the following processor registers.

3.2.1 Control registers

The control registers control the program sequence, state, and stack memory. The control registers consist of a program counter (PC), a program status word (PSW) and a stack pointer (SP).

1. Program counter (PC)
   The program counter is a 20-bit register that holds the address information of the next program to be executed. In normal operation, PC is automatically incremented according to the number of bytes of the instruction to be fetched. When a branch instruction is executed, immediate data and register contents are set. The values of the reset vector table at addresses 00000H and 00001H are set in the program counter following a reset.

Figure 3 - 5 Format of Program Counter

Symbol	19																			0
PC																				

2. Program status word (PSW)
   The program status word is an 8-bit register consisting of various flags set/reset by instruction execution. Program status word contents are stored in the stack area upon vectored interrupt request is acknowledged or PUSH PSW instruction execution and are restored upon execution of the RETB, RETI and POP PSW instructions. The value of the PSW is 06H following a reset.

Figure 3 - 6 Format of Program Status Word

Symbol	7							0
PSW	IE	Z	RBS1	AC	RBS0	ISP1	ISP0	CY

a) Interrupt enable flag (IE)
   This flag controls the interrupt request acknowledge operations of the CPU. When 0, the IE flag is set to the interrupt disabled (DI) state, and all maskable interrupt requests are disabled. When 1, the IE flag is set to the interrupt enabled (EI) state and interrupt request acknowledgment is controlled with an in-service priority flag (ISP1, ISP0), an interrupt mask flag for various interrupt sources, and a priority specification flag. The IE flag is reset to 0 upon DI instruction execution or interrupt acknowledgment and is set to 1 upon EI instruction execution.

b) Zero flag (Z)
   When the operation result is zero, this flag is set to 1. It is reset to 0 in all other cases.

c) Register bank select flags (RBS1, RBS0)
   These are 2-bit flags to select one of the four register banks. In these flags, the 2-bit information that indicates the register bank selected by SEL RBn instruction execution is stored.

d) Auxiliary carry flag (AC)
   If the operation result has a carry from bit 3 or a borrow at bit 3, this flag is set to 1. It is reset to 0 in all other cases.

e) In-service priority flags (ISP1, ISP0)
This flag manages the priority of acknowledgeable maskable vectored interrupts. Vectored interrupt requests specified lower than the value of ISP1 and ISP0 flags by the priority specification flag registers (PRn0L, PRn0H, PRn1L, PRn1H, PRn2L, PRn2H, PRn3L, and PRn3H; see **14.3.3 Priority specification flag registers**) cannot be acknowledged. Actual request acknowledgment is controlled by the interrupt enable flag (IE).

Remark n = 0, 1

f) Carry flag (CY)
This flag stores overflow and underflow upon add/subtract instruction execution. It stores the shift-out value upon rotate instruction execution and functions as a bit accumulator during bit operation instruction execution.

3. Stack pointer (SP)
This is a 16-bit register to hold the start address of the memory stack area. Only the internal RAM area can be set as the stack area.

Figure 3 - 7 Format of Stack Pointer

Symbol	15	14	13	12	11	10	9	8
SP	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8

7	6	5	4	3	2	1	0
SP7	SP6	SP5	SP4	SP3	SP2	SP1	0

In stack addressing through a stack pointer, the SP is decremented ahead of write (save) to the stack memory and is incremented after read (restore) from the stack memory.

Caution 1. Since the contents of the SP become undefined following a reset, be sure to initialize the SP before using the stack.

Caution 2. The space to which the general-purpose registers are allocated (FFEE0H to FFEFFH) cannot be used for instruction fetching or the stack.

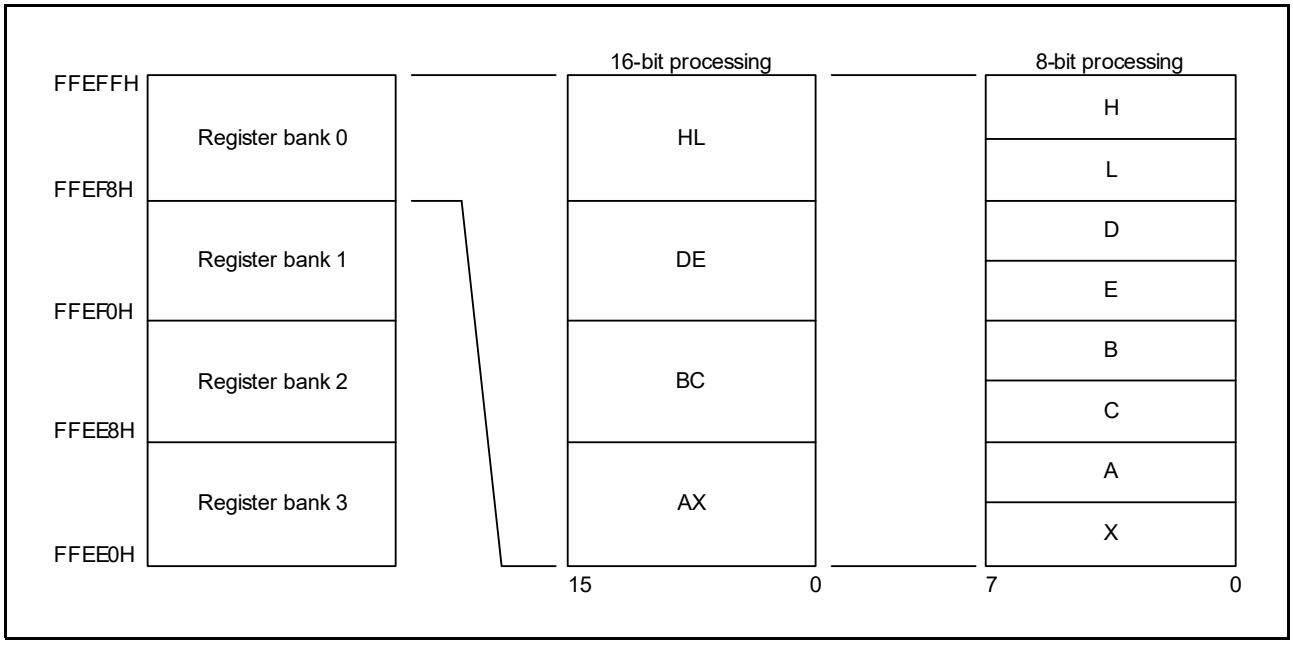
Caution 3. The internal RAM area in the following products cannot be used as stack memory when using the on-chip debugging trace function.
R7F103GxG (x = 6, B, F, G, L): FE300H to FE6FFH
R7F103GxJ (x = 6, B, F, G, L): FE300H to FE6FFH

3.2.2 General-purpose registers

General-purpose registers are mapped at particular addresses (FFEE0H to FFEFFH) of the data memory. The general-purpose registers consist of 4 banks, each bank consisting of eight 8-bit registers (X, A, C, B, E, D, L, and H). Each register can be used as an 8-bit register, and two 8-bit registers can also be used in a pair as a 16-bit register (AX, BC, DE, and HL). Register banks to be used for instruction execution are set by the CPU control instruction (SEL RBn). Because of the 4-register bank configuration, an efficient program can be created by switching between a register for normal processing and a register for interrupt processing for each bank.

Caution The space to which the general-purpose registers are allocated (FFEE0H to FFEFFH) cannot be used for instruction fetching or the stack.

Figure 3 - 8 Configuration of General-purpose Registers (Function Name)



3.2.3 ES and CS registers

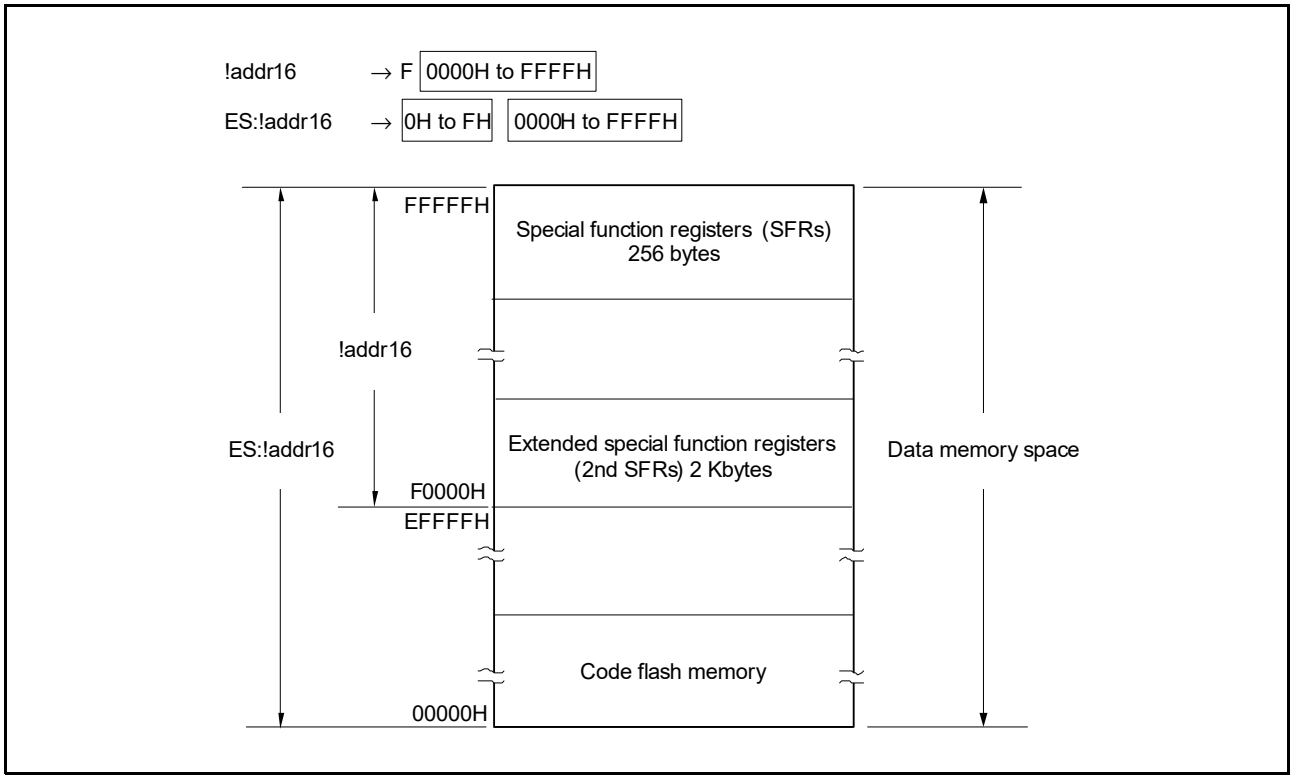
The ES register and CS register are used to specify the higher address for data access and when a branch instruction is executed (register indirect addressing), respectively. The default value of the ES register after reset is 0FH, and that of the CS register is 00H.

Figure 3 - 9 Configuration of ES and CS Registers

Symbol	7	6	5	4	3	2	1	0
ES	0	0	0	0	ES3	ES2	ES1	ES0
Symbol	7	6	5	4	3	2	1	0
CS	0	0	0	0	CS3	CS2	CS1	CS0

Though the data area which can be accessed with 16-bit addresses is the 64 Kbytes from F0000H to FFFFFH, using the ES register as well extends this to the 1 Mbyte from 00000H to FFFFFH.

Figure 3 - 10 Extension of Data Area Which Can Be Accessed



3.2.4 Special function registers (SFRs)

Unlike a general-purpose register, each special function register (SFR) has a special function. SFRs are allocated to the FFF00H to FFFFFH area. SFRs can be manipulated like general-purpose registers, using operation, transfer, and bit memory manipulation instructions. The manipulable bit units, 1, 8, and 16, depend on the SFR type. Each manipulation bit unit can be specified as follows.

- 1-bit memory manipulation
  Describe as follows for the 1-bit manipulation instruction operand (sfr.bit).
  When the bit name is defined: <Bit name>
  When the bit name is not defined: <Register name>, <Bit number> or <Address>, <Bit number>
- 8-bit memory manipulation
  Describe the symbol defined by the assembler for the 8-bit manipulation instruction operand (sfr). This manipulation can also be specified with an address.
- 16-bit memory manipulation
  Describe the symbol defined by the assembler for the 16-bit manipulation instruction operand (sfrp). When specifying an address, describe an even address.

Table 3 - 5 gives lists of the SFRs. The meanings of items in the table are as follows.

- Symbol
  This item indicates the address of a special function register. It is a reserved word in the assembler, and is defined as an sfr variable using the #pragma sfr directive in the compiler. When using the assembler, debugger, and simulator, symbols can be written as an instruction operand.
- R/W
  This item indicates whether the corresponding SFR can be read or written.
  R/W: Read/write enable
  R: Read only
  W: Write only
- Manipulable bit units
  "✓" indicates the manipulable bit unit (1, 8, or 16). "—" indicates a bit unit for which manipulation is not possible.
- After reset
  Items in this column indicate the states (values) of each of the registers after generation of a reset signal.

Caution **Do not access addresses to which SFRs are not assigned.**

Remark For 2nd SFRs, see **3.2.5 Extended special function registers (2nd SFRs)**.

Table 3 - 5 List of Special Function Registers (SFRs) (1/3)

Address	Special Function Register (SFR) Name	Symbol		R/W	Manipulable Bit Range			After Reset
					1 bit	8 bits	16 bits	
FFF00H	Port register 0	P0		R/W	✓	✓	—	00H
FFF01H	Port register 1	P1		R/W	✓	✓	—	00H
FFF02H	Port register 2	P2		R/W	✓	✓	—	00H
FFF03H	Port register 3	P3		R/W	✓	✓	—	00H
FFF04H	Port register 4	P4		R/W	✓	✓	—	00H
FFF05H	Port register 5	P5		R/W	✓	✓	—	00H
FFF06H	Port register 6	P6		R/W	✓	✓	—	00H
FFF07H	Port register 7	P7		R/W	✓	✓	—	00H
FFF0CH	Port register 12	P12		R/W	✓	✓	—	Undefined
FFF0DH	Port register 13	P13		R/W	✓	✓	—	Undefined
FFF0EH	Port register 14	P14		R/W	✓	✓	—	00H
FFF10H	Serial data register 00	TXD0/ SIO00	SDR00	R/W	—	✓	✓	0000H
FFF11H		—			—	—		
FFF12H	Serial data register 01	RXD0/ SIO01	SDR01	R/W	—	✓	✓	0000H
FFF13H		—			—	—		
FFF18H	Timer data register 00	TDR00		R/W	—	—	✓	0000H
FFF19H								
FFF1AH	Timer data register 01	TDR01L	TDR01	R/W	—	✓	✓	00H
FFF1BH		TDR01H			—	✓		00H
FFF1EH	10-bit A/D conversion result register	ADCR		R	—	—	✓	0000H
FFF1FH	8-bit A/D conversion result register	ADCRH		R	—	✓	—	00H
FFF20H	Port mode register 0	PM0		R/W	✓	✓	—	FFH
FFF21H	Port mode register 1	PM1		R/W	✓	✓	—	FFH
FFF22H	Port mode register 2	PM2		R/W	✓	✓	—	FFH
FFF23H	Port mode register 3	PM3		R/W	✓	✓	—	FFH
FFF24H	Port mode register 4	PM4		R/W	✓	✓	—	FFH
FFF25H	Port mode register 5	PM5		R/W	✓	✓	—	FFH
FFF26H	Port mode register 6	PM6		R/W	✓	✓	—	FFH
FFF27H	Port mode register 7	PM7		R/W	✓	✓	—	FFH
FFF2CH	Port mode register 12	PM12		R/W	✓	✓	—	FFH
FFF2EH	Port mode register 14	PM14		R/W	✓	✓	—	FFH
FFF30H	A/D converter mode register 0	ADM0		R/W	✓	✓	—	00H
FFF31H	Analog input channel specification register	ADS		R/W	✓	✓	—	00H
FFF32H	A/D converter mode register 1	ADM1		R/W	✓	✓	—	00H
FFF34H	Key return control register	KRCTL		R/W	✓	✓	—	00H
FFF35H	Key return flag register	KRF		R/W	—	✓	—	00H
FFF37H	Key return mode register 0	KRM0		R/W	✓	✓	—	00H
FFF38H	External interrupt rising edge enable register 0	EGP0		R/W	✓	✓	—	00H
FFF39H	External interrupt falling edge enable register 0	EGN0		R/W	✓	✓	—	00H
FFF3AH	External interrupt rising edge enable register 1	EGP1		R/W	✓	✓	—	00H
FFF3BH	External interrupt falling edge enable register 1	EGN1		R/W	✓	✓	—	00H
FFF44H	Serial data register 02	TXD1/ SIO10	SDR02	R/W	—	✓	✓	0000H
FFF45H		—			—	—		
FFF46H	Serial data register 03	RXD1/ SIO11	SDR03	R/W	—	✓	✓	0000H
FFF47H		—			—	—		

Table 3 - 5 List of Special Function Registers (SFRs) (2/3)

Address	Special Function Register (SFR) Name	Symbol		R/W	Manipulable Bit Range			After Reset
					1 bit	8 bits	16 bits	
FFF48H	Serial data register 10	TXD2/ SIO20	SDR10	R/W	—	✓	✓	0000H
FFF49H		—			—	—		
FFF4AH	Serial data register 11	RXD2/ SIO21	SDR11	R/W	—	✓	✓	0000H
FFF4BH		—			—	—		
FFF50H	IICA shift register 0	IICA0		R/W	—	✓	—	00H
FFF51H	IICA status register 0	IICS0		R	✓	✓	—	00H
FFF52H	IICA flag register 0	IICF0		R/W	✓	✓	—	00H
FFF64H	Timer data register 02	TDR02		R/W	—	—	✓	0000H
FFF65H								
FFF66H	Timer data register 03	TDR03L	TDR03	R/W	—	✓	✓	00H
FFF67H		TDR03H			—	✓		00H
FFF68H	Timer data register 04	TDR04		R/W	—	—	✓	0000H
FFF69H								
FFF6AH	Timer data register 05	TDR05		R/W	—	—	✓	0000H
FFF6BH								
FFF6CH	Timer data register 06	TDR06		R/W	—	—	✓	0000H
FFF6DH								
FFF6EH	Timer data register 07	TDR07		R/W	—	—	✓	0000H
FFF6FH								
FFFA0H	Clock operation mode control register	CMC		R/W	—	✓	—	00H
FFFA1H	Clock operation status control register	CSC		R/W	✓	✓	—	C0H
FFFA2H	Oscillation stabilization time counter status register	OSTC		R	✓	✓	—	00H
FFFA3H	Oscillation stabilization time select register	OSTS		R/W	—	✓	—	07H
FFFA4H	System clock control register	CKC		R/W	✓	✓	—	00H
FFFA7H	Subsystem clock select register	CKSEL		R/W	✓	✓	—	00H
FFFA8H	Reset control flag register	RESF		R	—	✓	—	Undefined **Note 1**
FFFA9H	Voltage detection register	LVIM		R/W	✓	✓	—	00H**Note 2**
FFFAAH	Voltage detection level register	LVIS		R/W	✓	✓	—	19H
FFFABH	Watchdog timer enable register	WDTE		R/W	—	✓	—	9AH/ 1AH**Note 3**
FFFACH	CRC input register	CRCIN		R/W	—	✓	—	00H
FFFD0H	Interrupt request flag register 2	IF2L	IF2	R/W	✓	✓	✓	00H
FFFD1H		IF2H		R/W	✓	✓		00H
FFFD2H	Interrupt request flag register 3	IF3L	IF3	R/W	✓	✓	✓	00H
FFFD3H		—			—	—		00H
FFFD4H	Interrupt mask flag register 2	MK2L	MK2	R/W	✓	✓	✓	FFH
FFFD5H		MK2H		R/W	✓	✓		FFH
FFFD6H	Interrupt mask flag register 3	MK3L	MK3	R/W	✓	✓	✓	FFH
FFFD7H		—			—	—		00H
FFFD8H	Priority specification flag register 02	PR02L	PR02	R/W	✓	✓	✓	FFH
FFFD9H		PR02H		R/W	✓	✓		FFH
FFFDAH	Priority specification flag register 03	PR03L	PR03	R/W	✓	✓	✓	FFH
FFFDBH		—			—	—		00H
FFFDCH	Priority specification flag register 12	PR12L	PR12	R/W	✓	✓	✓	FFH
FFFDDH		PR12H		R/W	✓	✓		FFH

Table 3 - 5 List of Special Function Registers (SFRs) (3/3)

Address	Special Function Register (SFR) Name	Symbol		R/W	Manipulable Bit Range			After Reset
					1 bit	8 bits	16 bits	
FFFDEH	Priority specification flag register 13	PR13L	PR13	R/W	✓	✓	✓	FFH
FFFDFH		—			—	—		00H
FFFE0H	Interrupt request flag register 0	IF0L	IF0	R/W	✓	✓	✓	00H
FFFE1H		IF0H		R/W	✓	✓		00H
FFFE2H	Interrupt request flag register 1	IF1L	IF1	R/W	✓	✓	✓	00H
FFFE3H		IF1H		R/W	✓	✓		00H
FFFE4H	Interrupt mask flag register 0	MK0L	MK0	R/W	✓	✓	✓	FFH
FFFE5H		MK0H		R/W	✓	✓		FFH
FFFE6H	Interrupt mask flag register 1	MK1L	MK1	R/W	✓	✓	✓	FFH
FFFE7H		MK1H		R/W	✓	✓		FFH
FFFE8H	Priority specification flag register 00	PR00L	PR00	R/W	✓	✓	✓	FFH
FFFE9H		PR00H		R/W	✓	✓		FFH
FFFEAH	Priority specification flag register 01	PR01L	PR01	R/W	✓	✓	✓	FFH
FFFEBH		PR01H		R/W	✓	✓		FFH
FFFECH	Priority specification flag register 10	PR10L	PR10	R/W	✓	✓	✓	FFH
FFFEDH		PR10H		R/W	✓	✓		FFH
FFFEEH	Priority specification flag register 11	PR11L	PR11	R/W	✓	✓	✓	FFH
FFFEFH		PR11H		R/W	✓	✓		FFH
FFFF0H	Multiplication and accumulation register (L)	MACRL		R/W	—	—	✓	0000H
FFFF1H								
FFFF2H	Multiplication and accumulation register (H)	MACRH		R/W	—	—	✓	0000H
FFFF3H								
FFFFEH	Processor mode control register	PMC		R/W	✓	✓	—	00H

Note 1. For the reset resources, see **Section 17 Reset Function**.

Note 2. The initial value depends on the source of the reset. For details, see **19.3.1 Voltage detection register (LVIM)**.

Note 3. The reset value of the WDTE register is determined by the setting of the option byte.

Remark For 2nd SFRs, see **Table 3 - 6 List of Extended Special Function Registers (2nd SFRs)**.

3.2.5 Extended special function registers (2nd SFRs)

Unlike a general-purpose register, each extended special function register (2nd SFR) has a special function. 2nd SFRs are allocated to the F0000H to F07FFH area. SFRs other than those in the SFR area (FFF00H to FFFFFH) are allocated to this area. An instruction that accesses the 2nd SFR area, however, is 1 byte longer than an instruction that accesses the SFR area. 2nd SFRs can be manipulated like general-purpose registers, using operation, transfer, and bit memory manipulation instructions. The manipulable bit units, 1, 8, and 16, depend on the 2nd SFR type. Each manipulation bit unit can be specified as follows.

- 1-bit memory manipulation
  Describe as follows for the 1-bit manipulation instruction operand (!addr16.bit).
  When the bit name is defined: <Bit name>
  When the bit name is not defined: <Register name>, <Bit number> or <Address>, <Bit number>
- 8-bit memory manipulation
  Describe the symbol defined by the assembler for the 8-bit manipulation instruction operand (!addr16). This manipulation can also be specified with an address.
- 16-bit memory manipulation
  Describe the symbol defined by the assembler for the 16-bit manipulation instruction operand (!addr16). When specifying an address, describe an even address.

Table 3 - 6 gives lists of the 2nd SFRs. The meanings of items in the table are as follows.

- Symbol
  This item indicates the address of a 2nd SFR. It is a reserved word in the assembler, and is defined as an sfr variable using the #pragma sfr directive in the compiler. When using the assembler, debugger, and simulator, symbols can be written as an instruction operand.
- R/W
  This item indicates whether the corresponding 2nd SFR can be read or written.
  R/W: Read/write enable
  R: Read only
  W: Write only
- Manipulable bit units
  "✓" indicates the manipulable bit unit (1, 8, or 16). "—" indicates a bit unit for which manipulation is not possible.
- After reset
  Items in this column indicate the states (values) of each of the registers after generation of a reset signal.

Caution **Do not access addresses to which 2nd SFRs are not assigned.**

Remark For SFRs in the SFR area, see **3.1.4 Special function register (SFR) area**.

Table 3 - 6 List of Extended Special Function Registers (2nd SFRs) (1/6)

Address	Extended Special Function Register (2nd SFR) Name	Symbol	R/W	Manipulable Bit Range			After Reset
				1-bit	8-bit	16-bit	
F0010H	A/D converter mode register 2	ADM2	R/W	✓	✓	—	00H
F0011H	Conversion result comparison upper limit setting register	ADUL	R/W	—	✓	—	FFH
F0012H	Conversion result comparison lower limit setting register	ADLL	R/W	—	✓	—	00H
F0013H	A/D test register	ADTES	R/W	—	✓	—	00H
F0030H	Pull-up resistor option register 0	PU0	R/W	✓	✓	—	00H
F0031H	Pull-up resistor option register 1	PU1	R/W	✓	✓	—	00H
F0033H	Pull-up resistor option register 3	PU3	R/W	✓	✓	—	00H
F0034H	Pull-up resistor option register 4	PU4	R/W	✓	✓	—	01H
F0035H	Pull-up resistor option register 5	PU5	R/W	✓	✓	—	00H
F0037H	Pull-up resistor option register 7	PU7	R/W	✓	✓	—	00H
F003CH	Pull-up resistor option register 12	PU12	R/W	✓	✓	—	00H
F003DH	Pull-up resistor option register 13	PU13	R/W	✓	✓	—	00H
F003EH	Pull-up resistor option register 14	PU14	R/W	✓	✓	—	00H
F0040H	Port input mode register 0	PIM0	R/W	✓	✓	—	00H
F0041H	Port input mode register 1	PIM1	R/W	✓	✓	—	00H
F0043H	Port input mode register 3	PIM3	R/W	✓	✓	—	00H
F0045H	Port input mode register 5	PIM5	R/W	✓	✓	—	00H
F0047H	Port input mode register 7	PIM7	R/W	✓	✓	—	00H
F0050H	Port output mode register 0	POM0	R/W	✓	✓	—	00H
F0051H	Port output mode register 1	POM1	R/W	✓	✓	—	00H
F0053H	Port output mode register 3	POM3	R/W	✓	✓	—	00H
F0055H	Port output mode register 5	POM5	R/W	✓	✓	—	00H
F0057H	Port output mode register 7	POM7	R/W	✓	✓	—	00H
F0060H	Port mode control A register 0	PMCA0	R/W	✓	✓	—	FFH
F0062H	Port mode control A register 2	PMCA2	R/W	✓	✓	—	FFH
F006CH	Port mode control A register 12	PMCA12	R/W	✓	✓	—	FFH
F006EH	Port mode control A register 14	PMCA14	R/W	✓	✓	—	FFH
F0070H	Noise filter enable register 0	NFEN0	R/W	✓	✓	—	00H
F0071H	Noise filter enable register 1	NFEN1	R/W	✓	✓	—	00H
F0073H	Input switch control register	ISC	R/W	✓	✓	—	00H
F0074H	Timer input select register 0	TIS0	R/W	—	✓	—	00H
F0077H	Peripheral I/O redirection register 0	PIOR0	R/W	—	✓	—	00H
F0078H	Invalid memory access detection control register	IAWCTL	R/W	—	✓	—	00H
F0079H	UART loop back select register	ULBS	R/W	✓	✓	—	00H
F007AH	Peripheral I/O redirection register 1	PIOR1	R/W	—	✓	—	00H
F007BH	Port mode select register	PMS	R/W	✓	✓	—	00H
F007DH	Global digital input disable register	GDIDIS	R/W	✓	✓	—	00H
F0090H	Data flash control register	DFLCTL	R/W	✓	✓	—	00H
F00A0H	High-speed on-chip oscillator trimming register	HIOTRM	R/W	—	✓	—	**Note 1**
F00A8H	High-speed on-chip oscillator frequency select register	HOCODIV	R/W	—	✓	—	**Note 2**
F00AAH	Flash operating mode select register	FLMODE	R/W	✓	✓	—	40H/80H/C0H **Note 3**
F00ABH	Flash operating mode protect register	FLMWRP	R/W	✓	✓	—	00H
F00B0H	Flash security flag monitoring register	FLSEC	R	—	—	✓	Undefined
F00B2H	Flash FSW monitoring register S	FLFSWS	R	—	—	✓	Undefined

Table 3 - 6 List of Extended Special Function Registers (2nd SFRs) (2/6)

Address	Extended Special Function Register (2nd SFR) Name	Symbol		R/W	Manipulable Bit Range			After Reset
					1-bit	8-bit	16-bit	
F00B4H	Flash FSW monitoring register E	FLFSWE		R	—	—	✓	Undefined
F00B6H	Flash memory sequencer initial setting register	FSSET		R/W	—	✓	—	00H
F00B7H	Flash extra area sequencer control register	FSSE		R/W	✓	✓	—	00H
F00C0H	Flash protect command register	PFCMD		W	—	✓	—	Undefined
F00C1H	Flash status register	PFS		R	✓	✓	—	00H
F00F0H	Peripheral enable register 0	PER0		R/W	✓	✓	—	00H
F00F1H	Peripheral reset control register 0	PRR0		R/W	✓	✓	—	00H
F00F3H	Subsystem clock supply mode control register	OSMC		R/W	✓	✓	—	Undefined **Note 4**
F00F5H	RAM parity error control register	RPECTL		R/W	✓	✓	—	00H
F00F9H	Power-on-reset status register	PORSR		R/W	✓	✓	—	00H
F00FAH	Peripheral enable register 1	PER1		R/W	✓	✓	—	00H
F00FBH	Peripheral reset control register 1	PRR1		R/W	✓	✓	—	00H
F00FEH	BCD correction result register	BCDADJ		R	—	✓	—	Undefined
F00FFH	Interrupt vector jump enable register	VECTCTRL		R/W	—	✓	—	00H
F0100H	Serial status register 00	SSR00L	SSR00	R	—	✓	✓	0000H
F0101H		—			—	—		
F0102H	Serial status register 01	SSR01L	SSR01	R	—	✓	✓	0000H
F0103H		—			—	—		
F0104H	Serial status register 02	SSR02L	SSR02	R	—	✓	✓	0000H
F0105H		—			—	—		
F0106H	Serial status register 03	SSR03L	SSR03	R	—	✓	✓	0000H
F0107H		—			—	—		
F0108H	Serial flag clear trigger register 00	SIR00L	SIR00	R/W	—	✓	✓	0000H
F0109H		—			—	—		
F010AH	Serial flag clear trigger register 01	SIR01L	SIR01	R/W	—	✓	✓	0000H
F010BH		—			—	—		
F010CH	Serial flag clear trigger register 02	SIR02L	SIR02	R/W	—	✓	✓	0000H
F010DH		—			—	—		
F010EH	Serial flag clear trigger register 03	SIR03L	SIR03	R/W	—	✓	✓	0000H
F010FH		—			—	—		
F0110H	Serial mode register 00	SMR00		R/W	—	—	✓	0020H
F0111H								
F0112H	Serial mode register 01	SMR01		R/W	—	—	✓	0020H
F0113H								
F0114H	Serial mode register 02	SMR02		R/W	—	—	✓	0020H
F0115H								
F0116H	Serial mode register 03	SMR03		R/W	—	—	✓	0020H
F0117H								
F0118H	Serial communication operation setting register 00	SCR00		R/W	—	—	✓	0087H
F0119H								
F011AH	Serial communication operation setting register 01	SCR01		R/W	—	—	✓	0087H
F011BH								
F011CH	Serial communication operation setting register 02	SCR02		R/W	—	—	✓	0087H
F011DH								
F011EH	Serial communication operation setting register 03	SCR03		R/W	—	—	✓	0087H
F011FH								
F0120H	Serial channel enable status register 0	SE0L	SE0	R	✓	✓	✓	0000H
F0121H		—			—	—		

Table 3 - 6 List of Extended Special Function Registers (2nd SFRs) (3/6)

Address	Extended Special Function Register (2nd SFR) Name	Symbol		R/W	Manipulable Bit Range			After Reset
					1-bit	8-bit	16-bit	
F0122H	Serial channel start register 0	SS0L	SS0	R/W	✓	✓	✓	0000H
F0123H		—			—	—		
F0124H	Serial channel stop register 0	ST0L	ST0	R/W	✓	✓	✓	0000H
F0125H		—			—	—		
F0126H	Serial clock select register 0	SPS0L	SPS0	R/W	—	✓	✓	0000H
F0127H		—			—	—		
F0128H	Serial output register 0	SO0		R/W	—	—	✓	0F0FH
F0129H								
F012AH	Serial output enable register 0	SOE0L	SOE0	R/W	✓	✓	✓	0000H
F012BH		—			—	—		
F0134H	Serial output level register 0	SOL0L	SOL0	R/W	—	✓	✓	0000H
F0135H		—			—	—		
F0138H	Serial standby control register 0	SSC0L	SSC0	R/W	—	✓	✓	0000H
F0139H		—			—	—		
F0140H	Serial status register 10	SSR10L	SSR10	R	—	✓	✓	0000H
F0141H		—			—	—		
F0142H	Serial status register 11	SSR11L	SSR11	R	—	✓	✓	0000H
F0143H		—			—	—		
F0148H	Serial flag clear trigger register 10	SIR10L	SIR10	R/W	—	✓	✓	0000H
F0149H		—			—	—		
F014AH	Serial flag clear trigger register 11	SIR11L	SIR11	R/W	—	✓	✓	0000H
F014BH		—			—	—		
F0150H	Serial mode register 10	SMR10		R/W	—	—	✓	0020H
F0151H								
F0152H	Serial mode register 11	SMR11		R/W	—	—	✓	0020H
F0153H								
F0158H	Serial communication operation setting register 10	SCR10		R/W	—	—	✓	0087H
F0159H								
F015AH	Serial communication operation setting register 11	SCR11		R/W	—	—	✓	0087H
F015BH								
F0160H	Serial channel enable status register 1	SE1L	SE1	R	✓	✓	✓	0000H
F0161H		—			—	—		
F0162H	Serial channel start register 1	SS1L	SS1	R/W	✓	✓	✓	0000H
F0163H		—			—	—		
F0164H	Serial channel stop register 1	ST1L	ST1	R/W	✓	✓	✓	0000H
F0165H		—			—	—		
F0166H	Serial clock select register 1	SPS1L	SPS1	R/W	—	✓	✓	0000H
F0167H		—			—	—		
F0168H	Serial output register 1	SO1		R/W	—	—	✓	0F0FH
F0169H								
F016AH	Serial output enable register 1	SOE1L	SOE1	R/W	✓	✓	✓	0000H
F016BH		—			—	—		
F0174H	Serial output level register 1	SOL1L	SOL1	R/W	—	✓	✓	0000H
F0175H		—			—	—		
F0178H	Serial standby control register 1	SSC1L	SSC1	R/W	—	✓	✓	0000H
F0179H		—			—	—		
F0180H	Timer counter register 00	TCR00		R	—	—	✓	FFFFH
F0181H								

Table 3 - 6 List of Extended Special Function Registers (2nd SFRs) (4/6)

Address	Extended Special Function Register (2nd SFR) Name	Symbol		R/W	Manipulable Bit Range			After Reset
					1-bit	8-bit	16-bit	
F0182H	Timer counter register 01	TCR01		R	—	—	✓	FFFFH
F0183H								
F0184H	Timer counter register 02	TCR02		R	—	—	✓	FFFFH
F0185H								
F0186H	Timer counter register 03	TCR03		R	—	—	✓	FFFFH
F0187H								
F0188H	Timer counter register 04	TCR04		R	—	—	✓	FFFFH
F0189H								
F018AH	Timer counter register 05	TCR05		R	—	—	✓	FFFFH
F018BH								
F018CH	Timer counter register 06	TCR06		R	—	—	✓	FFFFH
F018DH								
F018EH	Timer counter register 07	TCR07		R	—	—	✓	FFFFH
F018FH								
F0190H	Timer mode register 00	TMR00		R/W	—	—	✓	0000H
F0191H								
F0192H	Timer mode register 01	TMR01		R/W	—	—	✓	0000H
F0193H								
F0194H	Timer mode register 02	TMR02		R/W	—	—	✓	0000H
F0195H								
F0196H	Timer mode register 03	TMR03		R/W	—	—	✓	0000H
F0197H								
F0198H	Timer mode register 04	TMR04		R/W	—	—	✓	0000H
F0199H								
F019AH	Timer mode register 05	TMR05		R/W	—	—	✓	0000H
F019BH								
F019CH	Timer mode register 06	TMR06		R/W	—	—	✓	0000H
F019DH								
F019EH	Timer mode register 07	TMR07		R/W	—	—	✓	0000H
F019FH								
F01A0H	Timer status register 00	TSR00L	TSR00	R	—	✓	✓	0000H
F01A1H		—			—	—		
F01A2H	Timer status register 01	TSR01L	TSR01	R	—	✓	✓	0000H
F01A3H		—			—	—		
F01A4H	Timer status register 02	TSR02L	TSR02	R	—	✓	✓	0000H
F01A5H		—			—	—		
F01A6H	Timer status register 03	TSR03L	TSR03	R	—	✓	✓	0000H
F01A7H		—			—	—		
F01A8H	Timer status register 04	TSR04L	TSR04	R	—	✓	✓	0000H
F01A9H		—			—	—		
F01AAH	Timer status register 05	TSR05L	TSR05	R	—	✓	✓	0000H
F01ABH		—			—	—		
F01ACH	Timer status register 06	TSR06L	TSR06	R	—	✓	✓	0000H
F01ADH		—			—	—		
F01AEH	Timer status register 07	TSR07L	TSR07	R	—	✓	✓	0000H
F01AFH		—			—	—		
F01B0H	Timer channel enable status register 0	TE0L	TE0	R	✓	✓	✓	0000H
F01B1H		—			—	—		

Table 3 - 6 List of Extended Special Function Registers (2nd SFRs) (5/6)

Address	Extended Special Function Register (2nd SFR) Name	Symbol		R/W	Manipulable Bit Range			After Reset
					1-bit	8-bit	16-bit	
F01B2H	Timer channel start register 0	TS0L	TS0	R/W	✓	✓	✓	0000H
F01B3H		—			—	—		
F01B4H	Timer channel stop register 0	TT0L	TT0	R/W	✓	✓	✓	0000H
F01B5H		—			—	—		
F01B6H	Timer clock select register 0	TPS0		R/W	—	—	✓	0000H
F01B7H								
F01B8H	Timer output register 0	TO0L	TO0	R/W	—	✓	✓	0000H
F01B9H		—			—	—		
F01BAH	Timer output enable register 0	TOE0L	TOE0	R/W	✓	✓	✓	0000H
F01BBH		—			—	—		
F01BCH	Timer output level register 0	TOL0L	TOL0	R/W	—	✓	✓	0000H
F01BDH		—			—	—		
F01BEH	Timer output mode register 0	TOM0L	TOM0	R/W	—	✓	✓	0000H
F01BFH		—			—	—		
F0214H	High-speed system clock division register	MOSCDIV		R/W	—	✓	—	00H
F0218H	LVD detection flag clearing register	LVDFCLR		R/W	✓	✓	—	00H
F0230H	IICA control register 00	IICCTL00		R/W	✓	✓	—	00H
F0231H	IICA control register 01	IICCTL01		R/W	✓	✓	—	00H
F0232H	IICA low-level width setting register 0	IICWL0		R/W	—	✓	—	FFH
F0233H	IICA high-level width setting register 0	IICWH0		R/W	—	✓	—	FFH
F0234H	Slave address register 0	SVA0		R/W	—	✓	—	00H
F02B0H	Port digital input disable register 0	PDIDIS0		R/W	✓	✓	—	00H
F02B1H	Port digital input disable register 1	PDIDIS1		R/W	✓	✓	—	00H
F02B3H	Port digital input disable register 3	PDIDIS3		R/W	✓	✓	—	00H
F02B5H	Port digital input disable register 5	PDIDIS5		R/W	✓	✓	—	00H
F02B7H	Port digital input disable register 7	PDIDIS7		R/W	✓	✓	—	00H
F02BDH	Port digital input disable register 13	PDIDIS13		R/W	✓	✓	—	00H
F02C0H	Flash programming mode control register	FLPMC		R/W	—	✓	—	08H
F02C1H	Flash area selection register	FLARS		R/W	✓	✓	—	00H
F02C2H	Flash address pointer register L	FLAPL		R/W	—	—	✓	0000H
F02C4H	Flash address pointer register H	FLAPH		R/W	—	✓	—	00H
F02C5H	Flash memory sequencer control register	FSSQ		R/W	✓	✓	—	00H
F02C6H	Flash end address pointer register L	FLSEDL		R/W	—	—	✓	0000H
F02C8H	Flash end address pointer register H	FLSEDH		R/W	—	✓	—	00H
F02C9H	Flash registers initialization register	FLRST		R/W	✓	✓	—	00H
F02CAH	Flash memory sequencer status register L	FSASTL		R	✓	✓	—	00H/80H
F02CBH	Flash memory sequencer status register H	FSASTH		R	✓	✓	—	00H/04H
F02CCH	Flash write buffer register L	FLWL		R/W	—	—	✓	0000H
F02CEH	Flash write buffer register H	FLWH		R/W	—	—	✓	0000H
F02F0H	Flash memory CRC control register	CRC0CTL		R/W	✓	✓	—	00H
F02F2H	Flash memory CRC operation result register	PGCRCL		R/W	—	—	✓	0000H
F02FAH	CRC data register	CRCD		R/W	—	—	✓	0000H
F0300H	Transmit buffer register 0	TXBA0		R/W	—	✓	—	FFH
F0301H	Receive buffer register 0	RXBA0		R	—	✓	—	FFH
F0302H	Operation mode setting register 00	ASIMA00		R/W	✓	✓	—	01H
F0303H	Operation mode setting register 01	ASIMA01		R/W	✓	✓	—	1AH
F0304H	Baud rate generator control register 0	BRGCA0		R/W	—	✓	—	FFH
F0305H	Status register 0	ASISA0		R	—	✓	—	00H

Table 3 - 6 List of Extended Special Function Registers (2nd SFRs) (6/6)

Address	Extended Special Function Register (2nd SFR) Name	Symbol		R/W	Manipulable Bit Range			After Reset
					1-bit	8-bit	16-bit	
F0306H	Status clear trigger register 0	ASCTA0		R/W	✓	✓	—	00H
F0310H	UARTA clock select register 0	UTA0CK		R/W	✓	✓	—	00H
F0360H	Interval timer compare register 00	ITLCMP000	ITLCMP00	R/W	—	✓	✓	FFFFH
F0361H		ITLCMP001			—	✓		
F0362H	Interval timer compare register 01	ITLCMP012	ITLCMP01	R/W	—	✓	✓	FFFFH
F0363H		ITLCMP013			—	✓		
F0364H	Interval timer capture register 00	ITLCAP00		R	—	—	✓	0000H
F0365H								
F0366H	Interval timer control register	ITLCTL0		R/W	✓	✓	—	00H
F0367H	Interval timer clock select register 0	ITLCSEL0		R/W	—	✓	—	00H
F0368H	Interval timer frequency division register 0	ITLFDIV00		R/W	—	✓	—	00H
F0369H	Interval timer frequency division register 1	ITLFDIV01		R/W	—	✓	—	00H
F036AH	Interval timer capture control register 0	ITLCC0		R/W	✓	✓	—	00H
F036BH	Interval timer status register	ITLS0		R/W	—	✓	—	00H
F036CH	Interval timer match detection mask register	ITLMKF0		R/W	—	✓	—	00H
F0480H	Interrupt vector change register 0	FLSIVC0		R/W	—	—	✓	0000H
F0481H								
F0482H	Interrupt vector change register 1	FLSIVC1		R/W	—	—	✓	000FH
F0483H								
F0488H	Code flash memory guard register	GFLASH0		R/W	—	—	✓	0000H
F0489H								
F048AH	Data flash memory guard register	GFLASH1		R/W	—	—	✓	0000H
F048BH								
F048CH	Flash security area guard register	GFLASH2		R/W	—	—	✓	0000H
F048DH								
F048EH	Guard register of IAWCTL register	GIAWCTL		R/W	—	—	✓	0000H
F048FH								

Note 1. The value after a reset is adjusted at the time of shipment.

Note 2. The value after a reset is a value set in the FRQSEL2 to FRQSEL0 bits of the user option byte (000C2H).

Note 3. The initial value of the FLMODE register is set to the value of the MODE1 and MODE0 bits updated with the set value of the CMODE1 and CMODE0 bits in the option byte at address 000C2H.

Note 4. In the initial value of the OSMC register, the value of the WUTMMCK0 bit is 0 and that of the HIPREC bit is 1.

Remark For SFRs in the SFR area, see **Table 3 - 5 List of Special Function Registers (SFRs)**.

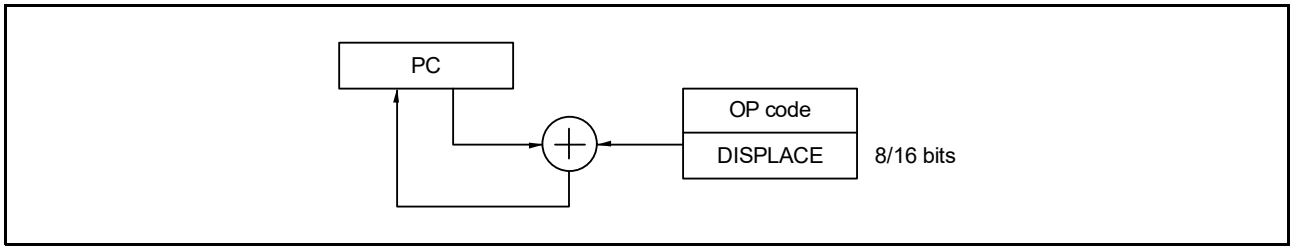
3.3 Instruction Address Addressing

3.3.1 Relative addressing

[Function]

Relative addressing stores in the program counter (PC) the result of adding a displacement value included in the instruction word (signed complement data: –128 to +127 or –32768 to +32767) to the program counter (PC)'s value (the start address of the next instruction), and specifies the program address to be used as the branch destination. Relative addressing is applied only to branch instructions.

Figure 3 - 11 Outline of Relative Addressing



3.3.2 Immediate addressing

[Function]

Immediate addressing stores immediate data of the instruction word in the program counter, and specifies the program address to be used as the branch destination. For immediate addressing, CALL !!addr20 or BR !!addr20 is used to specify 20-bit addresses and CALL !addr16 or BR !addr16 is used to specify 16-bit addresses. 0000 is set to the higher 4 bits when specifying 16-bit addresses.

Figure 3 - 12 Example of CALL !!addr20/BR !!addr20

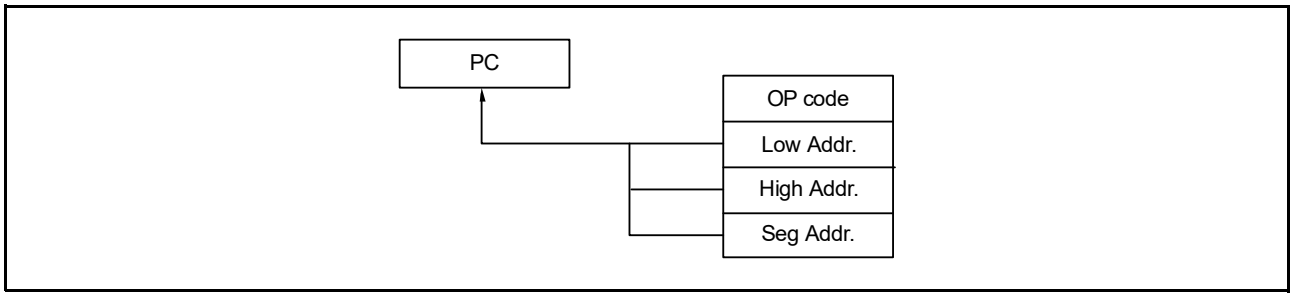
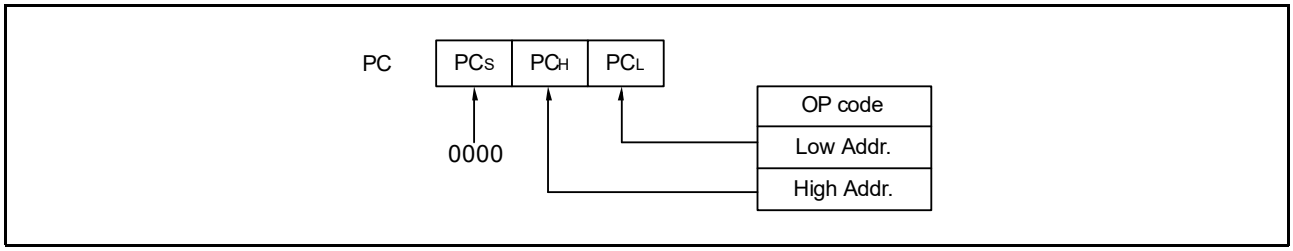


Figure 3 - 13 Example of CALL !addr16/BR !addr16

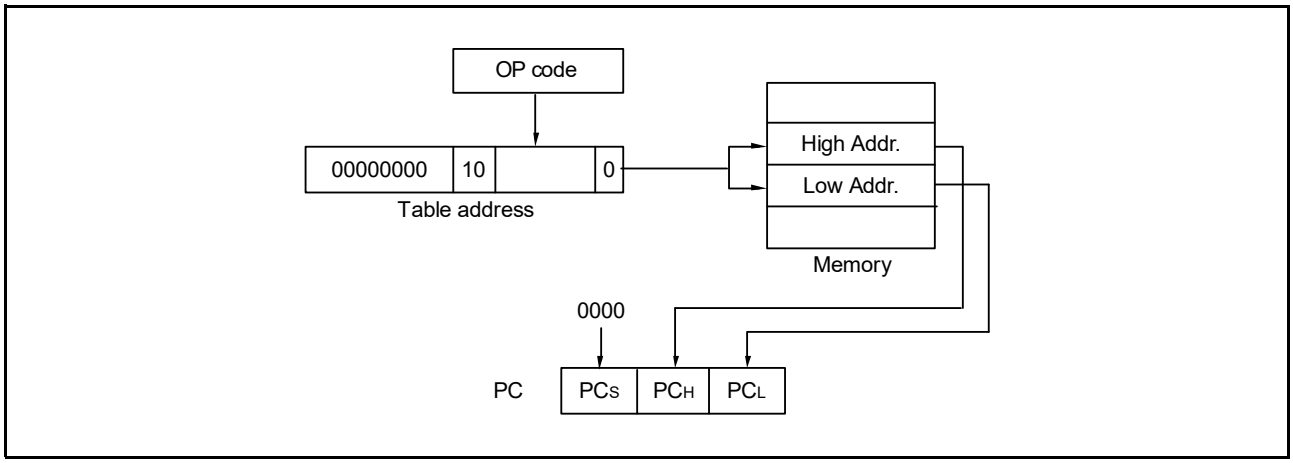


3.3.3 Table indirect addressing

[Function]

Table indirect addressing specifies a table address in the CALLT table area (0080H to 00BFH) with the 5-bit immediate data in the instruction word, stores the contents at that table address and the next address in the program counter (PC) as 16-bit data, and specifies the program address. Table indirect addressing is applied only for CALLT instructions. In the RL78 microcontrollers, branching is enabled only to the 64-Kbyte space from 00000H to 0FFFFH.

Figure 3 - 14 Outline of Table Indirect Addressing

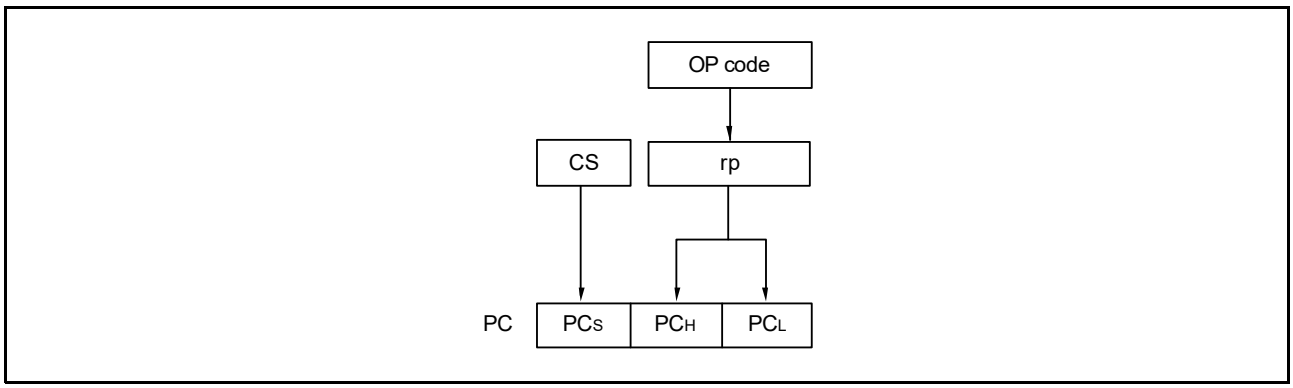


3.3.4 Register indirect addressing

[Function]

Register indirect addressing stores in the program counter (PC) the contents of a general-purpose register pair (AX/BC/DE/HL) and CS register of the current register bank specified with the instruction word as 20-bit data, and specifies the program address. Register indirect addressing can be applied only to the CALL AX, BC, DE, HL, and BR AX instructions.

Figure 3 - 15 Outline of Register Indirect Addressing



3.4 Types of Addressing for Processing Data Addresses

3.4.1 Implied addressing

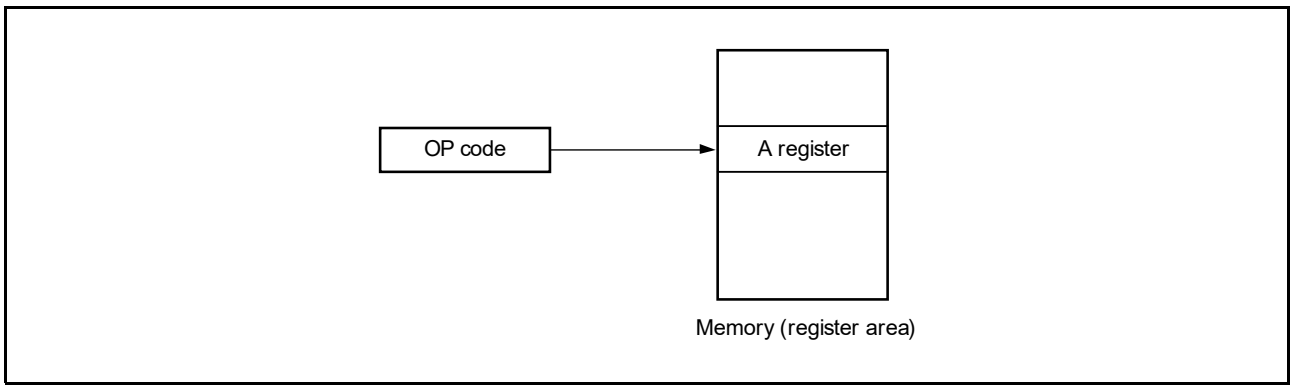
[Function]

Instructions for accessing registers (such as accumulators) that have special functions are directly specified with the instruction word, without using any register specification field in the instruction word.

[Operand format]

Implied addressing can be applied only to MULU X.

Figure 3 - 16 Outline of Implied Addressing



3.4.2 Register addressing

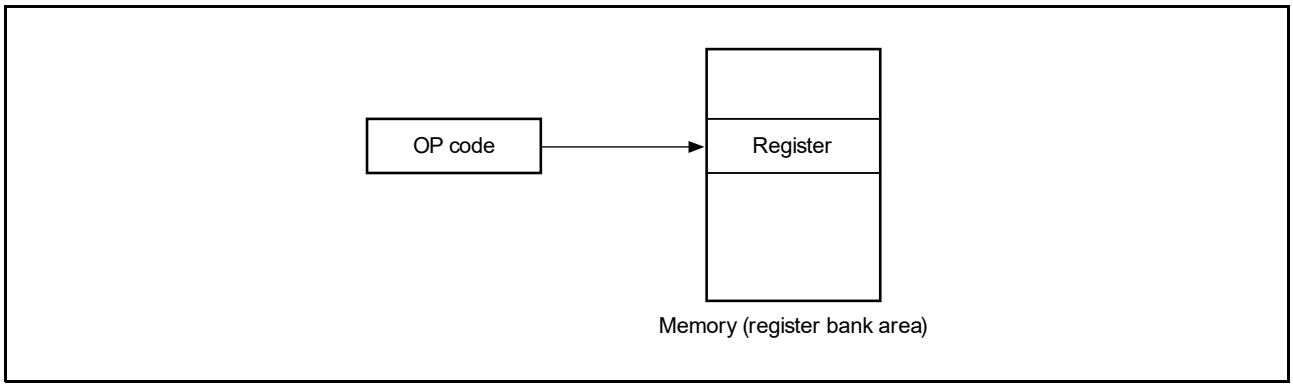
[Function]

Register addressing accesses a general-purpose register as an operand. The instruction word of 3-bit long is used to select an 8-bit register and the instruction word of 2-bit long is used to select a 16-bit register.

[Operand format]

Identifier	Description
r	X, A, C, B, E, D, L, H
rp	AX, BC, DE, HL

Figure 3 - 17 Outline of Register Addressing



3.4.3 Direct addressing

[Function]

Direct addressing uses immediate data in the instruction word as an operand address to directly specify the target address.

[Operand format]

Identifier	Description
!addr16	Label or 16-bit immediate data (only the space from F0000H to FFFFFH is specifiable)
ES:!addr16	Label or 16-bit immediate data (higher 4-bit addresses are specified by the ES register)

Figure 3 - 18 Example of !addr16

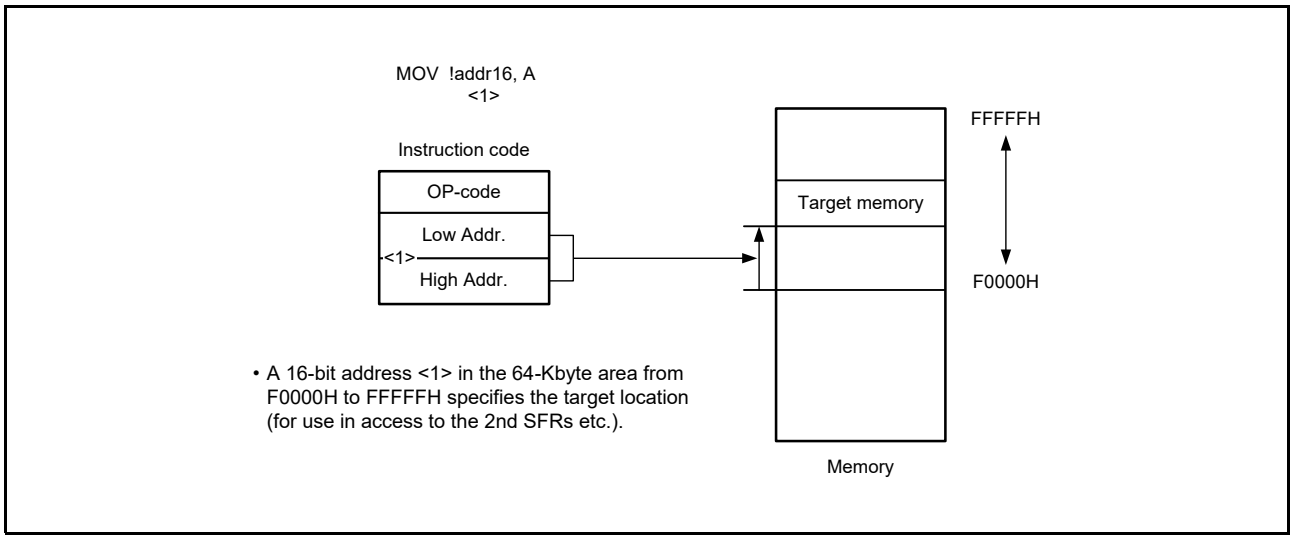
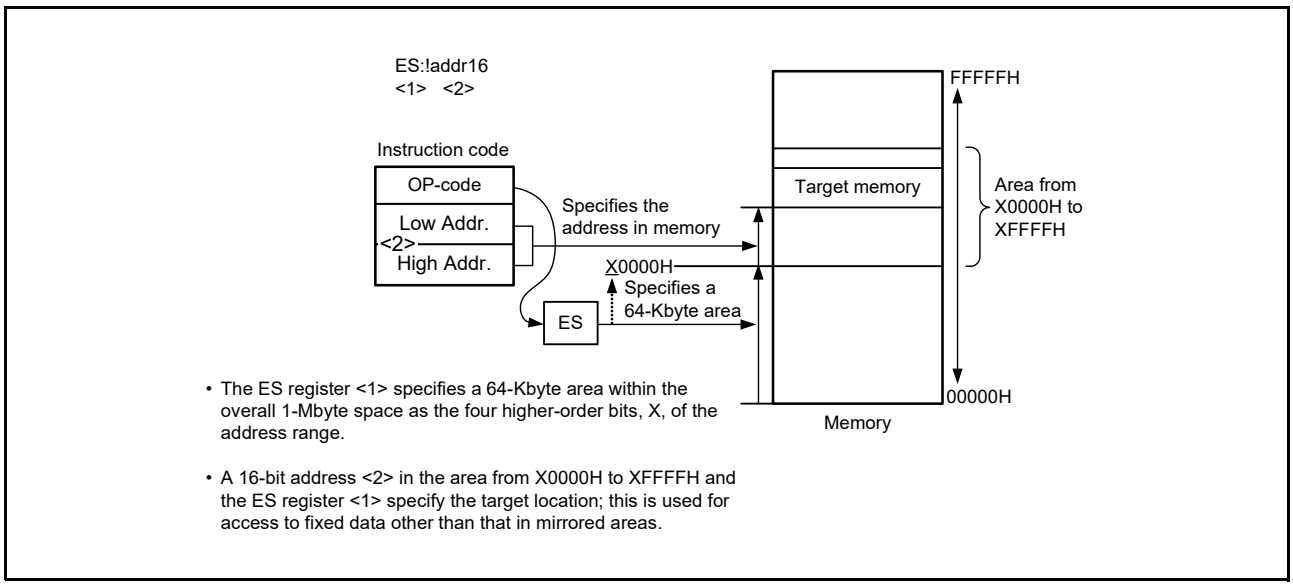


Figure 3 - 19 Example of ES:!addr16



3.4.4 Short direct addressing

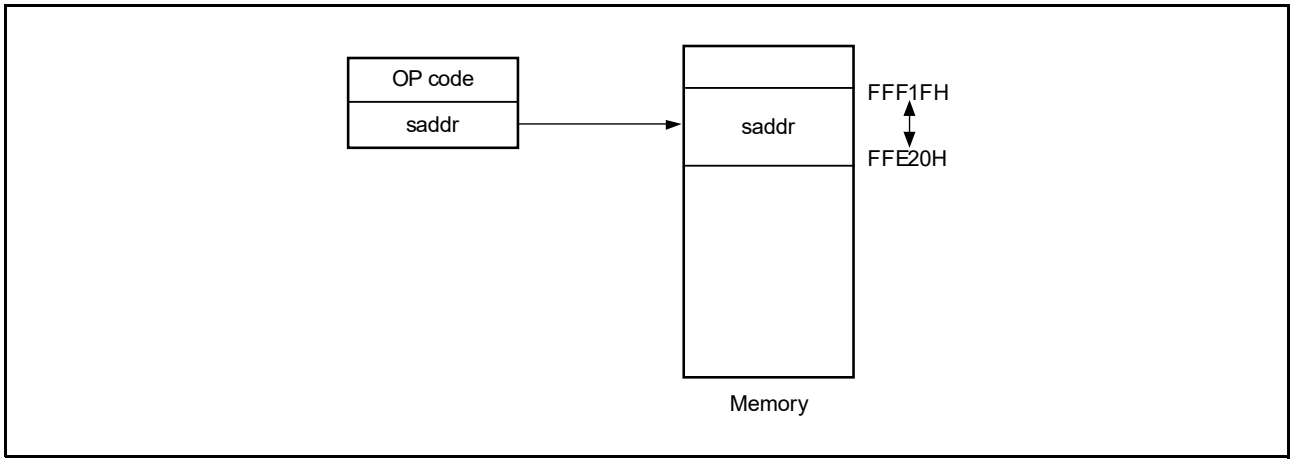
[Function]

Short direct addressing directly specifies the target addresses using 8-bit data in the instruction word. This type of addressing is applied only to the space from FFE20H to FFF1FH.

[Operand format]

Identifier	Description
SADDR	Label, FFE20H to FFF1FH immediate data, or 0FE20H to 0FF1FH immediate data (only the space from FFE20H to FFF1FH is specifiable)
SADDRP	Label, FFE20H to FFF1FH immediate data, or 0FE20H to 0FF1FH immediate data (even address only) (only the space from FFE20H to FFF1FH is specifiable)

Figure 3 - 20 Outline of Short Direct Addressing



Remark SADDR and SADDRP are used to describe the values of addresses FE20H to FF1FH with 16-bit immediate data (higher 4 bits of actual address are omitted), and the values of addresses FFE20H to FFF1FH with 20-bit immediate data. Regardless of whether SADDR or SADDRP is used, addresses within the space from FFE20H to FFF1FH are specified for the memory.

3.4.5 SFR addressing

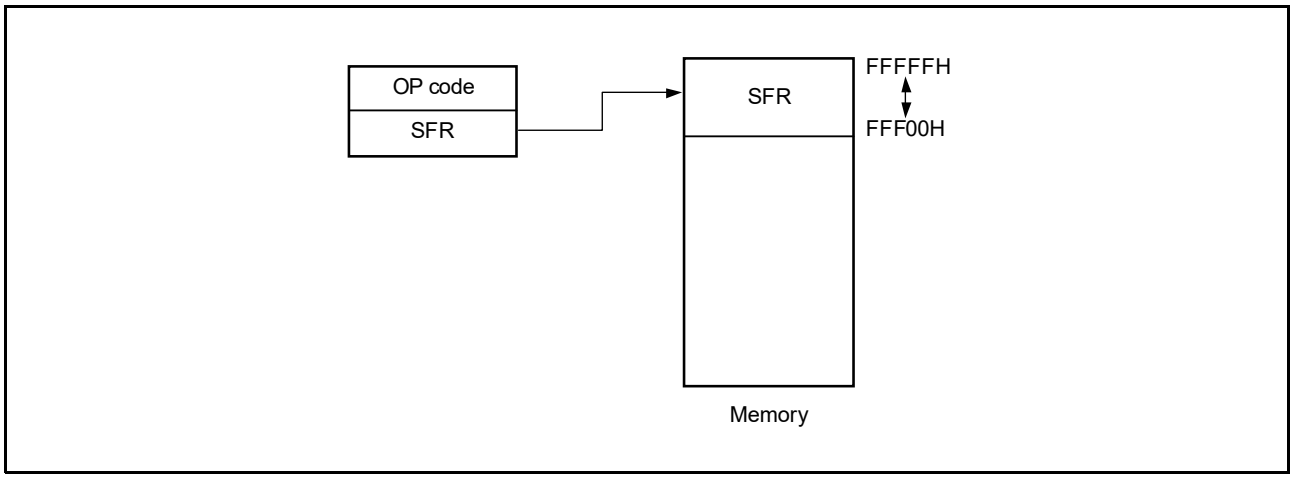
[Function]

SFR addressing directly specifies the target SFR addresses using 8-bit data in the instruction word. This type of addressing is applied only to the space from FFF00H to FFFFFH.

[Operand format]

Identifier	Description
SFR	SFR name
SFRP	16-bit-manipulatable SFR name (even address)

Figure 3 - 21 Outline of SFR Addressing



3.4.6 Register indirect addressing

[Function]

Register indirect addressing directly specifies the target addresses using the contents of the register pair specified with the instruction word as an operand address.

[Operand format]

Identifier	Description
—	[DE], [HL] (only the space from F0000H to FFFFFH is specifiable)
—	ES:[DE], ES:[HL] (higher 4-bit addresses are specified by the ES register)

Figure 3 - 22 Example of [DE], [HL]

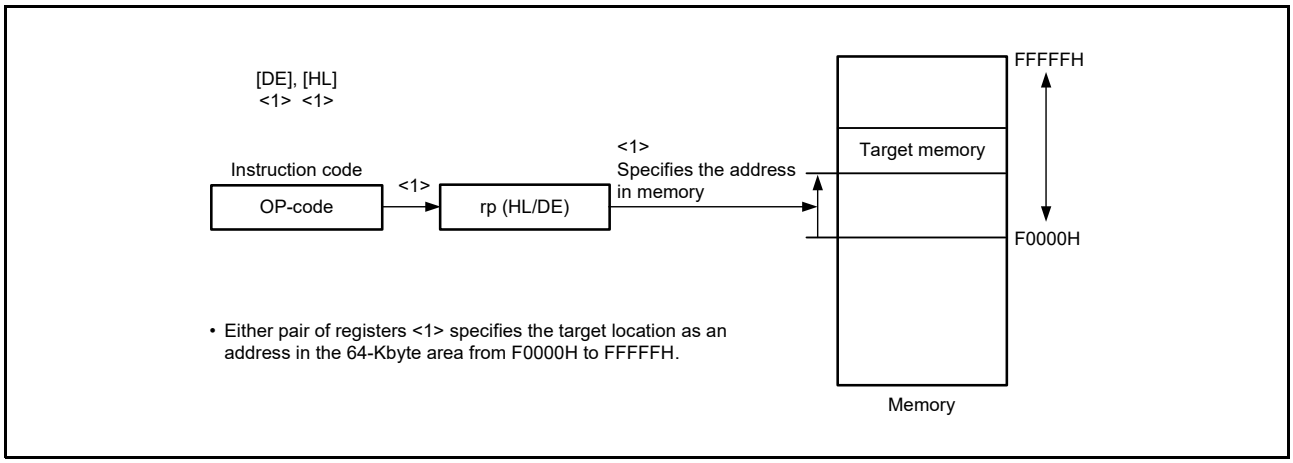
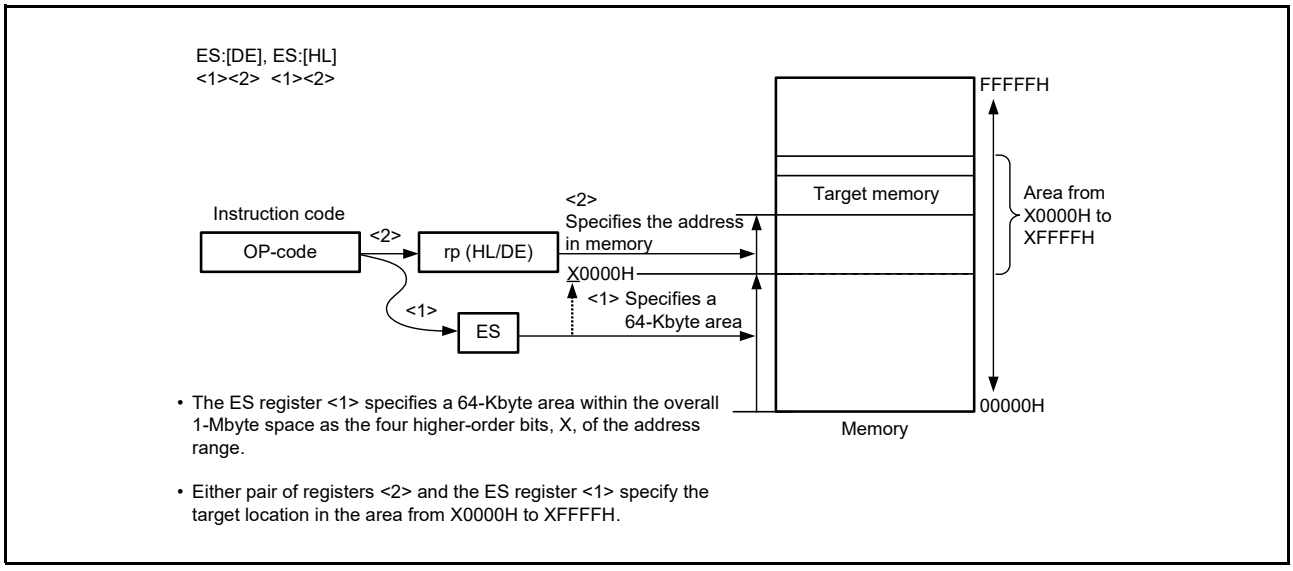


Figure 3 - 23 Example of ES:[DE], ES:[HL]



3.4.7 Based addressing

[Function]

Based addressing uses the contents of a register pair specified with the instruction word or 16-bit immediate data as a base address, and 8-bit immediate data or 16-bit immediate data as offset data. The sum of these values is used to specify the target address.

[Operand format]

Identifier	Description
—	[HL+byte], [DE+byte], [SP+byte] (only the space from F0000H to FFFFFH is specifiable)
—	word[B], word[C] (only the space from F0000H to FFFFFH is specifiable)
—	word[BC] (only the space from F0000H to FFFFFH is specifiable)
—	ES:[HL+byte], ES:[DE+byte] (higher 4-bit addresses are specified by the ES register)
—	ES:word[B], ES:word[C] (higher 4-bit addresses are specified by the ES register)
—	ES:word[BC] (higher 4-bit addresses are specified by the ES register)

Figure 3 - 24 Example of [SP+byte]

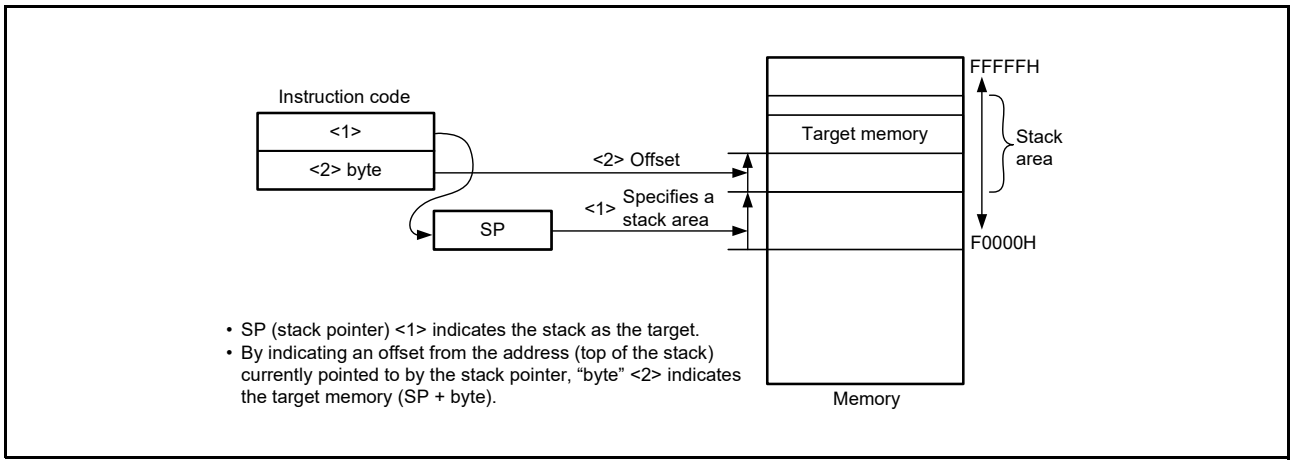


Figure 3 - 25 Example of [HL+byte], [DE+byte]

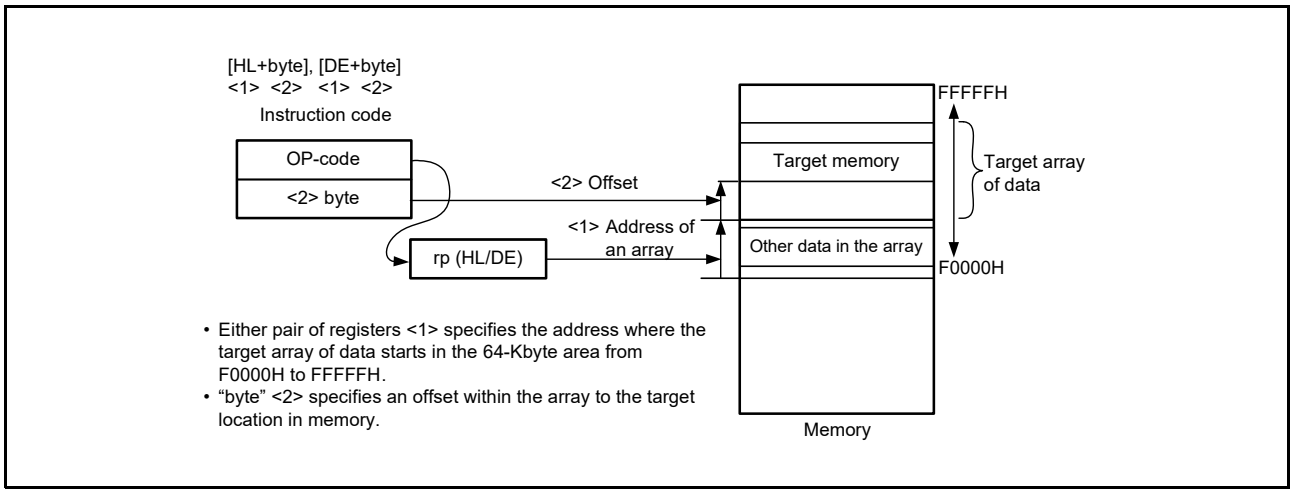


Figure 3 - 26 Example of word[B], word[C]

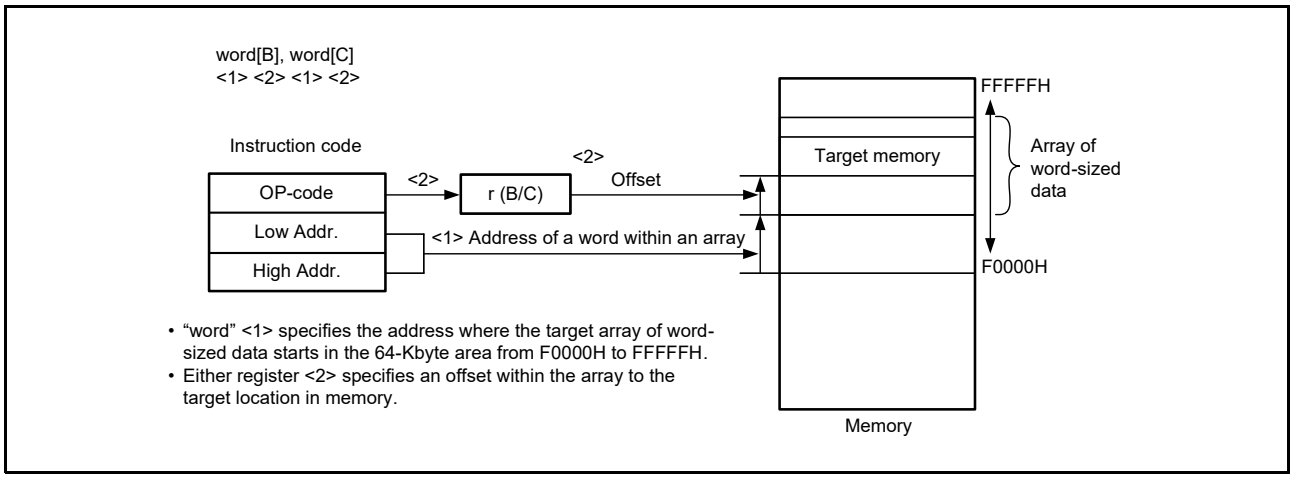


Figure 3 - 27 Example of word[BC]

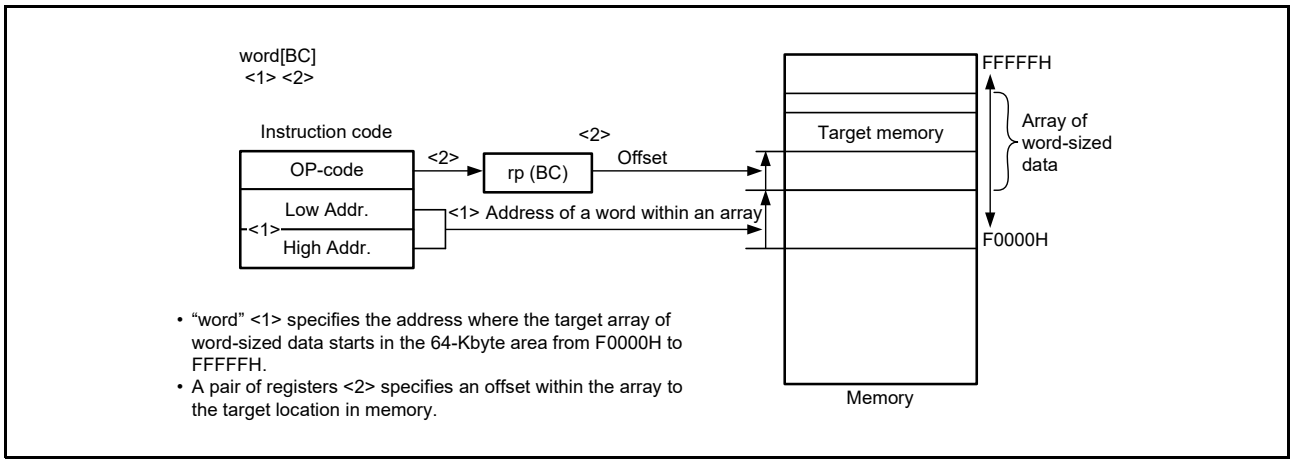


Figure 3 - 28 Example of ES:[HL+byte], ES:[DE+byte]

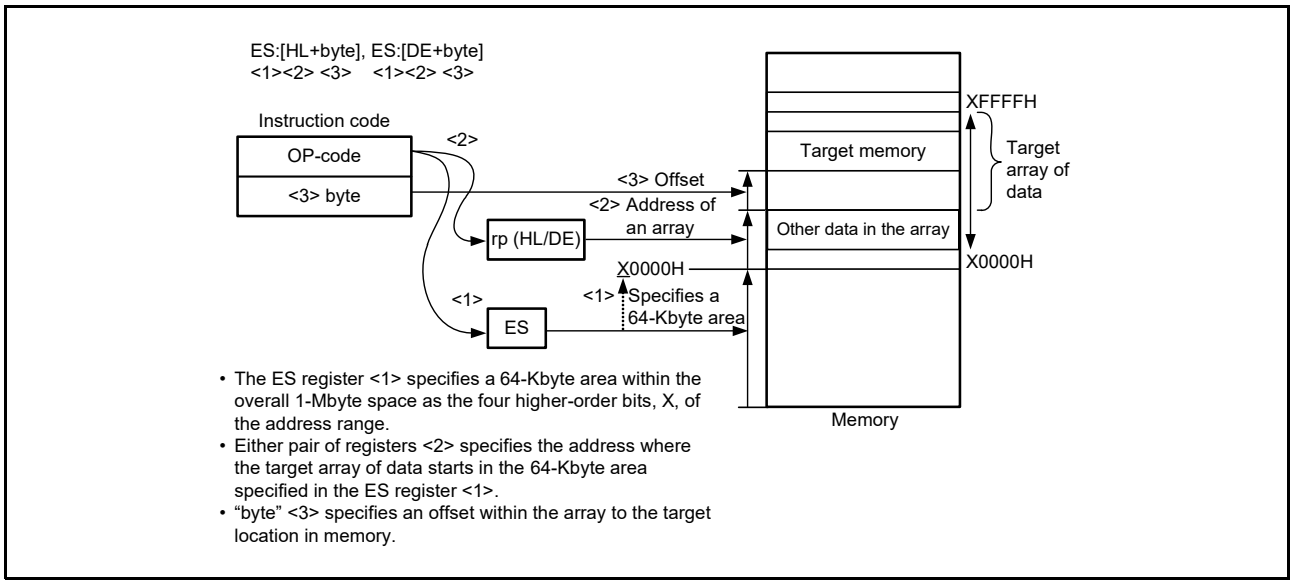
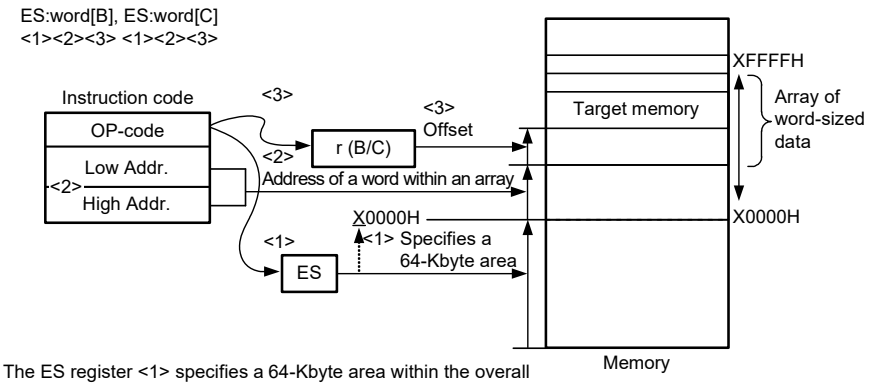
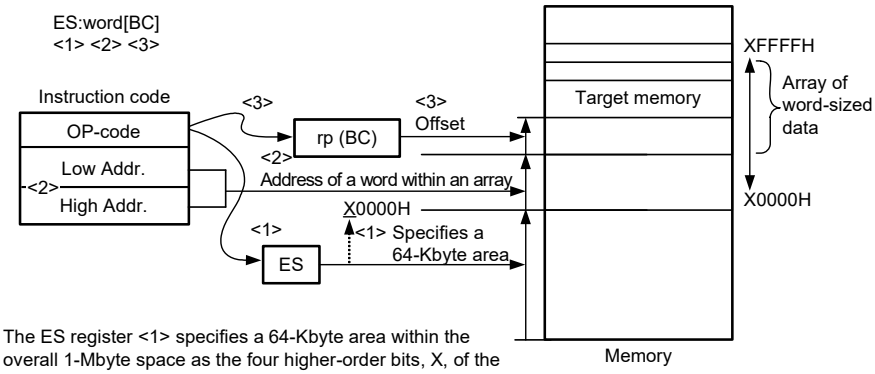


Figure 3 - 29 Example of ES:word[B], ES:word[C]



- The ES register <1> specifies a 64-Kbyte area within the overall 1-Mbyte space as the four higher-order bits, X, of the address range.
- "word" <2> specifies the address where the target array of word-sized data starts in the 64-Kbyte area specified in the ES register <1>.
- Either register <3> specifies an offset within the array to the target location in memory.

Figure 3 - 30 Example of ES:word[BC]



- The ES register <1> specifies a 64-Kbyte area within the overall 1-Mbyte space as the four higher-order bits, X, of the address range.
- "word" <2> specifies the address where the target array of word-sized data starts in the 64-Kbyte area specified in the ES register <1>.
- A pair of registers <3> specifies an offset within the array to the target location in memory.

3.4.8 Based indexed addressing

[Function]

Based indexed addressing uses the contents of a register pair specified with the instruction word as the base address, and the content of the B register or C register similarly specified with the instruction word as offset address. The sum of these values is used to specify the target address.

[Operand format]

Identifier	Description
—	[HL+B], [HL+C] (only the space from F0000H to FFFFFH is specifiable)
—	ES:[HL+B], ES:[HL+C] (higher 4-bit addresses are specified by the ES register)

Figure 3 - 31 Example of [HL+B], [HL+C]

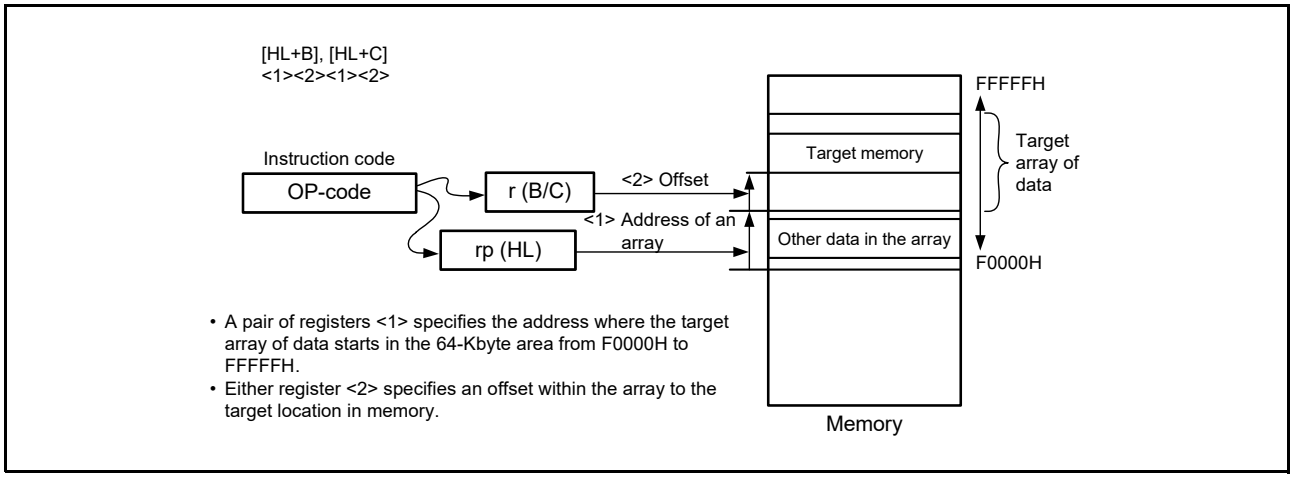
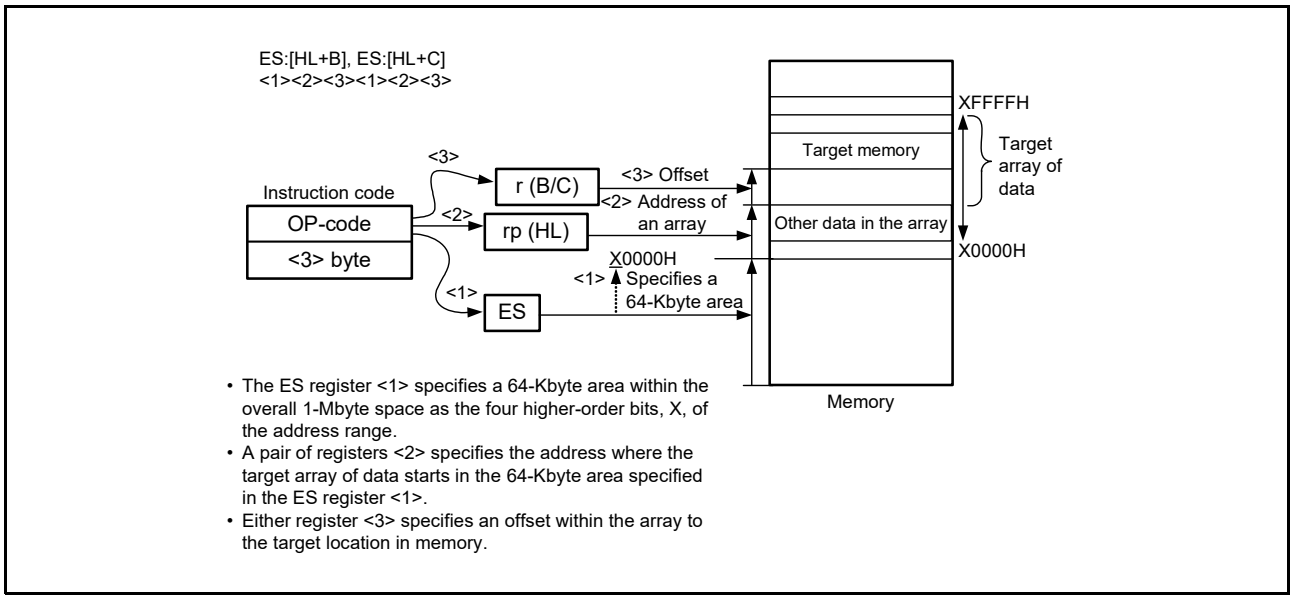


Figure 3 - 32 Example of ES:[HL+B], ES:[HL+C]



3.4.9 Stack addressing

[Function]

The stack area is indirectly addressed with the stack pointer (SP) values. This addressing is automatically employed when the PUSH, POP, subroutine call, and return instructions are executed or the register is saved/restored upon generation of an interrupt request. Only the internal RAM area can be set as the stack area.

[Operand format]

Identifier	Description
—	PUSH PSW AX/BC/DE/HL POP PSW AX/BC/DE/HL CALL/CALLT RET BRK RETB (Interrupt request generated) RETI

The data to be saved/restored by each stack operation is shown in **Figure 3 - 33** to **Figure 3 - 38**.

Figure 3 - 33 Example of PUSH rp

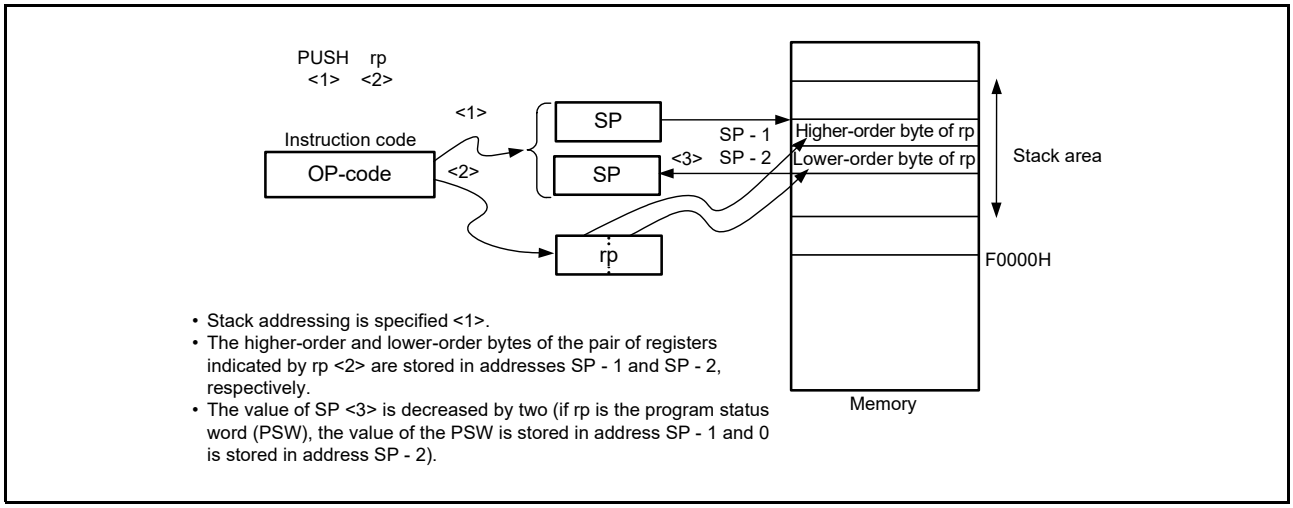


Figure 3 - 34 Example of POP

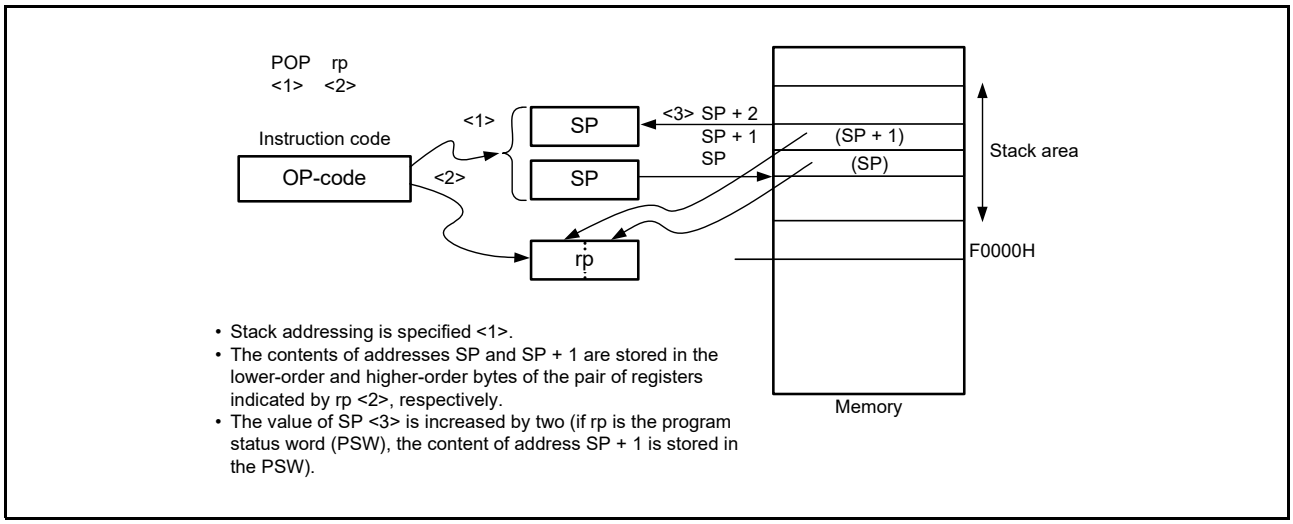


Figure 3 - 35 Example of CALL, CALLT

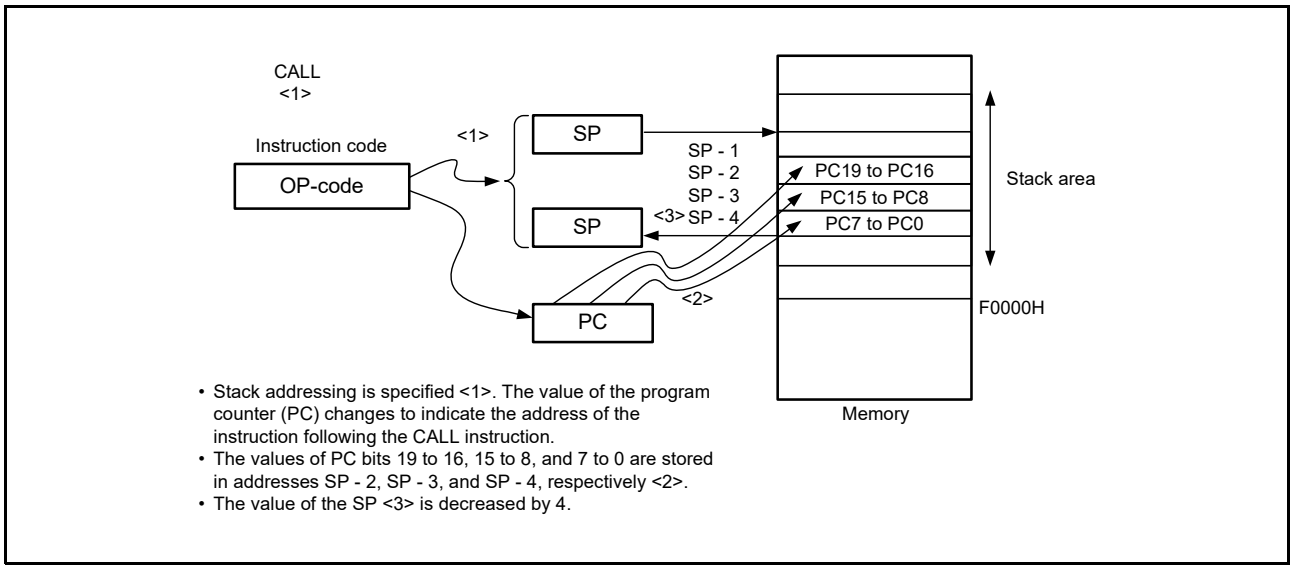


Figure 3 - 36 Example of RET

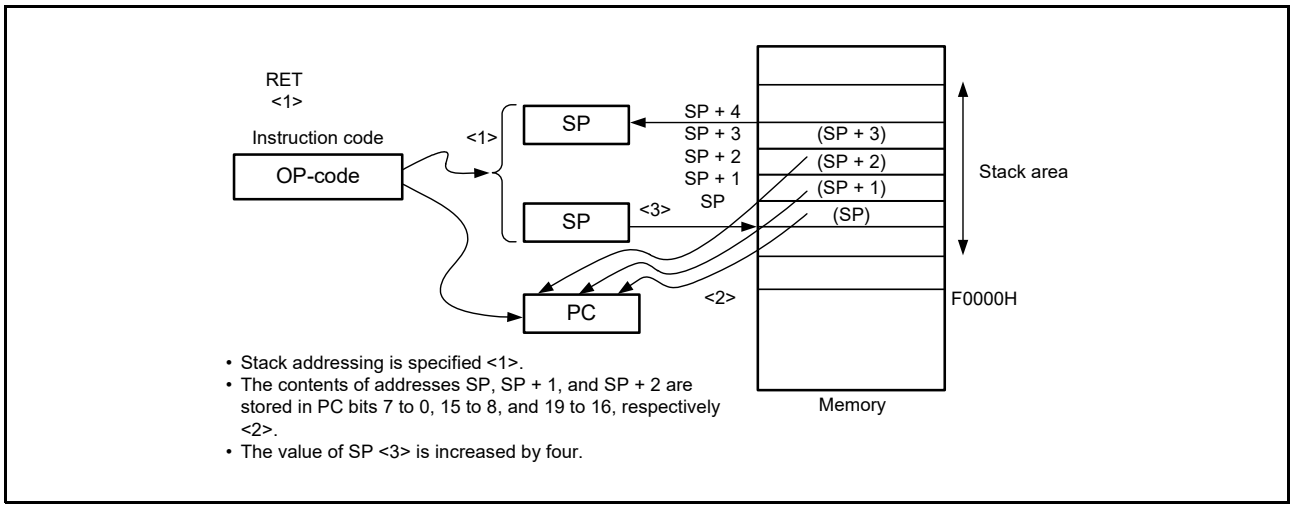
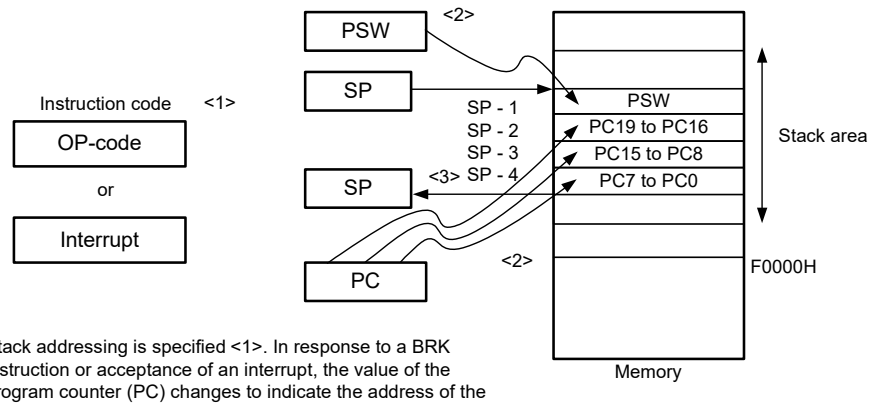
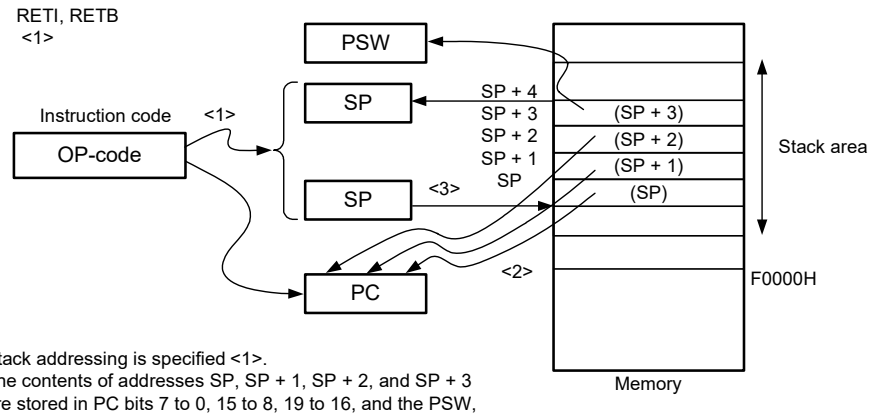


Figure 3 - 37 Example of Interrupt, BRK



- Stack addressing is specified <1>. In response to a BRK instruction or acceptance of an interrupt, the value of the program counter (PC) changes to indicate the address of the next instruction.
- The values of the PSW, PC bits 19 to 16, 15 to 8, and 7 to 0 are stored in addresses SP - 1, SP - 2, SP - 3, and SP - 4, respectively <2>.
- The value of the SP <3> is decreased by 4.

Figure 3 - 38 Example of RETI, RETB



- Stack addressing is specified <1>.
- The contents of addresses SP, SP + 1, SP + 2, and SP + 3 are stored in PC bits 7 to 0, 15 to 8, 19 to 16, and the PSW, respectively <2>.
- The value of SP <3> is increased by four.

Section 4 Port Functions

4.1 Port Functions

Products of this group are provided with digital I/O ports, which enable variety of control operations. In addition to the function as digital I/O ports, these ports have several alternate functions. For details of the alternate functions, see **Section 2 Pin Functions**.

4.2 Port Configuration

Ports include the following hardware.

Table 4 - 1 Port Configuration

Item	Configuration
Control registers	• Port mode registers (PMxx) (xx = 0 to 7, 12, 14) • Port registers (Pxx) (xx = 0 to 7, 12 to 14) • Pull-up resistor option registers (PUxx) (xx = 0, 1, 3 to 5, 7, 12 to 14) • Port input mode registers (PIMxx) (xx = 0, 1, 3, 5, 7) • Port output mode registers (POMxx) (xx = 0, 1, 3, 5, 7) • Port digital input disable registers (PDIDISxx) (xx = 0, 1, 3, 5, 7, 13) • Port mode control A registers (PMCAxx) (xx = 0, 2, 12, 14) • Peripheral I/O redirection registers (PIORx) (x = 0, 1) • Global digital input disable register (GDIDIS) • Port mode select register (PMS)
Port	• 20-pin products Total: 17 (CMOS I/O: 16 (N-ch open-drain I/O [withstand voltage of V_{DD}]: 5), CMOS input: 1) • 32-pin products Total: 29 (CMOS I/O: 26 (N-ch open-drain I/O [withstand voltage of V_{DD}]: 11), CMOS input: 1, N-ch open-drain I/O [withstand voltage of 6 V]: 2) • 44-pin products Total: 41 (CMOS I/O: 38 (N-ch open-drain I/O [withstand voltage of V_{DD}]: 14), CMOS input: 1, N-ch open-drain I/O [withstand voltage of 6 V]: 2) • 48-pin products Total: 45 (CMOS I/O: 41 (N-ch open-drain I/O [withstand voltage of V_{DD}]: 15), CMOS input: 1, CMOS output: 1, N-ch open-drain I/O [withstand voltage of 6 V]: 2) • 64-pin products Total: 59 (CMOS I/O: 55 (N-ch open-drain I/O [withstand voltage of EV_{DD}]: 19), CMOS input: 1, CMOS output: 1, N-ch open-drain I/O [withstand voltage of 6 V]: 2)
Pull-up resistor	• 20-pin products Total: 14 • 32-pin products Total: 23 • 44-pin products Total: 31 • 48-pin products Total: 34 • 64-pin products Total: 48

4.2.1 Port 0

Port 0 is an I/O port with an output latch. Port 0 can be set to the input mode or output mode in 1-bit units by the port mode register (PM0). When the P00 to P06 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by the pull-up resistor option register (PU0). Input to the P01, P03 and P04 pins can be specified through a normal input buffer or a TTL input buffer in 1-bit units by the port input mode register (PIM0). Output from the P00 and P02 to P04 pins can be set to the N-ch open-drain output (withstand voltage of V_{DD}**Note 1**/withstand voltage of EV_{DD}**Note 2**) mode in 1-bit units by the port output mode register (POM0). This port can also be used for timer I/O, A/D converter analog input, and serial interface data I/O and clock I/O. Use the registers shown in **4.3 Registers for Controlling the Port Function** to specify the states of each of the pins. For the correspondence between register settings and pin state, see **Table 4 - 4 Correspondence between Register Settings and Port Pin State**. P00 to P06 are set to the modes listed below following a reset.

- P00 and P01 pins of the 20- and 32-pin products: Analog input mode
- P00, P01 and P04 to P06 pins of the 44- to 64-pin products: Input mode
- P02 and P03 pins of the 64-pin products: Analog input mode

Note 1. For 20- to 48-pin products
Note 2. For 64-pin products

4.2.2 Port 1

Port 1 is an I/O port with an output latch. Port 1 can be set to the input mode or output mode in 1-bit units by the port mode register (PM1). When the P10 to P17 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by the pull-up resistor option register (PU1). Input to the P10, P11, and P13 to P17 pins can be specified through a normal input buffer or a TTL input buffer in 1-bit units by the port input mode register (PIM1). Output from the P10 to P15 and P17 pins can be set to the N-ch open-drain output (withstand voltage of V_{DD}**Note 1**/withstand voltage of EV_{DD}**Note 2**) mode in 1-bit units by the port output mode register (POM1). This port can also be used for serial interface data I/O and clock I/O, UART data transmission and reception for external device connection when programming flash memory, timer I/O, and external interrupt request input. Use the registers shown in **4.3 Registers for Controlling the Port Function** to specify the states of each of the pins. For the correspondence between register settings and pin state, see **Table 4 - 4 Correspondence between Register Settings and Port Pin State.** P10 to P17 are set to input mode following a reset.

Note 1. For 20 - to 48-pin-pin products
Note 2. For 64-pin products

4.2.3 Port 2

Port 2 is an I/O port with an output latch. Port 2 can be set to the input mode or output mode in 1-bit units by the port mode register (PM2). This port can also be used for A/D converter analog input, and positive and negative reference voltage inputs of the A/D converter. Use the registers shown in **4.3 Registers for Controlling the Port Function** to specify the states of each of the pins. For the correspondence between register settings and pin state, see **Table 4 - 4 Correspondence between Register Settings and Port Pin State**. P20 to P27 are set to analog input mode following a reset.

4.2.4 Port 3

Port 3 is an I/O port with an output latch. Port 3 can be set to the input mode or output mode in 1-bit units by the port mode register (PM3). When the P30, P31, P33, and P34 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by the pull-up resistor option register (PU3). Input to the P34 pin can be specified through a normal input buffer or a TTL input buffer by the port input mode register (PIM3). Output from the P31 and P33 pins can be set to the N-ch open-drain output (withstand voltage of V_{DD}**Note 1**/withstand voltage of EV_{DD}**Note 2**) mode by the port output mode register (POM3). This port can also be used for external interrupt request input, timer I/O, and serial interface data I/O and clock I/O. Use the registers shown in **4.3 Registers for Controlling the Port Function** to specify the states of each of the pins. For the correspondence between register settings and pin state, see **Table 4 - 4 Correspondence between Register Settings and Port Pin State**. P30, P31, P33, and P34 are set to input mode following a reset.

Note 1. For 20- to 48-pin products
Note 2. For 64-pin products

4.2.5 Port 4

Port 4 is an I/O port with an output latch. Port 4 can be set to the input mode or output mode in 1-bit units by the port mode register (PM4). When the P40 to P43 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by the pull-up resistor option register (PU4). This port can also be used for data I/O for a flash memory programmer/debugger and timer I/O. Use the registers shown in **4.3 Registers for Controlling the Port Function** to specify the states of each of the pins. For the correspondence between register settings and pin state, see **Table 4 - 4 Correspondence between Register Settings and Port Pin State**. P40 to P43 are set to input mode following a reset.

4.2.6 Port 5

Port 5 is an I/O port with an output latch. Port 5 can be set to the input mode or output mode in 1-bit units by the port mode register (PM5). When the P50 to P55 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by the pull-up resistor option register (PU5). Input to the P50 and P55 pins can be specified through a normal input buffer or a TTL input buffer in 1-bit units by the port input mode register (PIM5). Output from the P50, P51, and P55 pins can be set to the N-ch open-drain output (withstand voltage of V_{DD}**Note 1**/withstand voltage of EV_{DD}**Note 2**) mode in 1-bit units by the port output mode register (POM5). This port can also be used for external interrupt request input, and serial interface data I/O and clock I/O. Use the registers shown in **4.3 Registers for Controlling the Port Function** to specify the states of each of the pins. For the correspondence between register settings and pin state, see **Table 4 - 4 Correspondence between Register Settings and Port Pin State**. P50 to P55 are set to input mode following a reset.

Note 1. For 32- to 48-pin products
Note 2. For 64-pin products

4.2.7 Port 6

Port 6 is an I/O port with an output latch. Port 6 can be set to the input mode or output mode in 1-bit units by the port mode register (PM6). Output from the P60 and P61 pins is N-ch open-drain output (withstand voltage of 6 V). This port can also be used for serial interface data I/O and clock I/O. Use the registers shown in **4.3 Registers for Controlling the Port Function** to specify the states of each of the pins. For the correspondence between register settings and pin state, see **Table 4 - 4 Correspondence between Register Settings and Port Pin State**. P60 and P61 are set to input mode following a reset.

4.2.8 Port 7

Port 7 is an I/O port with an output latch. Port 7 can be set to the input mode or output mode in 1-bit units by the port mode register (PM7). When this port is used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by the pull-up resistor option register (PU7). Input to the P71 pin can be specified through a normal input buffer or a TTL input buffer by the port input mode register (PIM7). Output from the P71, P72, and P74 pins can be set to the N-ch open-drain output (withstand voltage of V_{DD}**Note 1**/withstand voltage of EV_{DD}**Note 2**) mode in 1-bit units by the port output mode register (POM7). This port can also be used for key interrupt input, serial interface data I/O and clock I/O, and external interrupt request input. Use the registers shown in **4.3 Registers for Controlling the Port Function** to specify the states of each of the pins. For the correspondence between register settings and pin state, see **Table 4 - 4 Correspondence between Register Settings and Port Pin State**. P70 to P77 are set to input mode following a reset.

Note 1. For 32- to 48-pin products
Note 2. For 64-pin products

4.2.9 Port 12

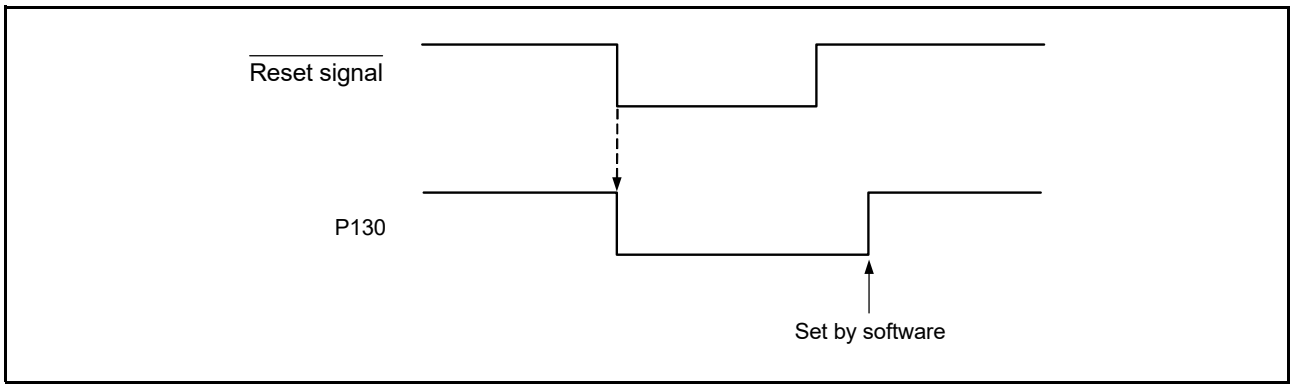
Port 12 is an I/O port with an output latch. Port 12 can be set to the input mode or output mode in 1-bit units by the port mode register (PM12). When this port is used as an input port, use of an on-chip pull-up resistor can be specified by the pull-up resistor option register (PU12). When P125 is set for use as the $\overline{\text{RESET}}$ function (PORTSELB = 1) by the user option byte (000C1H), the on-chip pull-up resistor is always enabled. This port can also be used for A/D converter analog input, connection of a resonator for the main system clock, external interrupt request input, reset input, and serial interface data I/O. Use the registers shown in **4.3 Registers for Controlling the Port Function** to specify the states of each of the pins. For the correspondence between register settings and pin state, see **Table 4 - 4 Correspondence between Register Settings and Port Pin State**. P120 is set to analog input mode and P121 to P125 are set to input mode following a reset.

Caution **After power is supplied, P125 functions as the $\overline{\text{RESET}}$ input. The MCU remains in the reset state as long as a low-level signal is applied to P125 even if the internal reset signal is released by the power-on reset (POR). When the pin is to be used for P125 or INTP9, select the port function (PORTSELB = 0) in the user option byte (000C1H), and release all reset sources.**

4.2.10 Port 13

P130 is a 1-bit output-only port with an output latch. P137 is a 1-bit input-only port. P130 is fixed to output mode, and P137 is fixed to input mode. When the P137 pin is used as an input port, use of an on-chip pull-up resistor can be specified by the pull-up resistor option register (PU13). This port can also be used for external interrupt request input. Use the registers shown in **4.3 Registers for Controlling the Port Function** to specify the states of each of the pins. For the correspondence between register settings and pin state, see **Table 4 - 4 Correspondence between Register Settings and Port Pin State**.

Remark When a reset signal is applied, P130 outputs a low-level signal. If P130 is set to a mode for outputting a high-level signal before a reset signal is applied, the P130 signal can be used to indicate a CPU reset.



4.2.11 Port 14

Port 14 is an I/O port with an output latch. Port 14 can be set to the input mode or output mode in 1-bit units by the port mode register (PM14). When the P140, P141, P146, and P147 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by the pull-up resistor option register (PU14). This port can also be used for external interrupt request input and A/D converter analog input. Use the registers shown in **4.3 Registers for Controlling the Port Function** to specify the states of each of the pins. For the correspondence between register settings and pin state, see **Table 4 - 4 Correspondence between Register Settings and Port Pin State**. P140, P141, and P146 are set to input mode and P147 is set to analog input mode following a reset.

4.3 Registers for Controlling the Port Function

The following registers are used to control the port functions.

- Port mode registers (PMxx)
- Port registers (Pxx)
- Pull-up resistor option registers (PUxx)
- Port input mode registers (PIMxx)
- Port output mode registers (POMxx)
- Port digital input disable registers (PDIDISxx)
- Port mode control A registers (PMCAxx)
- Peripheral I/O redirection registers (PIORx)
- Global digital input disable register (GDIDIS)
- Port mode select register (PMS)

Caution The inclusion of registers and bit functions depends on the product. For details on registers and bit functions included in each product, see Table 4 - 2. Be sure to set bits to which no functions are allocated to their initial values.

Table 4 - 2 PMxx, Pxx, PUxx, PIMxx, POMxx, PDIDISxx, and PMCAxx Registers and the Bits Implemented in Each Product (1/3)

Port		Bit name							64 pins	48 pins	44 pins	32 pins	20 pins
		PMxx register	Pxx register	PUxx register	PIMxx register	POMxx register	PDIDISxx register	PMCAxx register					
Port 0	0	PM00	P00	PU00	—	POM00	PDIDIS00	PMCA00 **Note**	✓	✓	✓	✓	✓
	1	PM01	P01	PU01	PIM01	—	—	PMCA01 **Note**	✓	✓	✓	✓	✓
	2	PM02	P02	PU02	—	POM02	PDIDIS02	PMCA02	✓	—	—	—	—
	3	PM03	P03	PU03	PIM03	POM03	PDIDIS03	PMCA03	✓	—	—	—	—
	4	PM04	P04	PU04	PIM04	POM04	PDIDIS04	—	✓	—	—	—	—
	5	PM05	P05	PU05	—	—	—	—	✓	—	—	—	—
	6	PM06	P06	PU06	—	—	—	—	✓	—	—	—	—
	7	—	—	—	—	—	—	—	—	—	—	—	—
Port 1	0	PM10	P10	PU10	PIM10	POM10	PDIDIS10	—	✓	✓	✓	✓	✓
	1	PM11	P11	PU11	PIM11	POM11	PDIDIS11	—	✓	✓	✓	✓	✓
	2	PM12	P12	PU12	—	POM12	PDIDIS12	—	✓	✓	✓	✓	✓
	3	PM13	P13	PU13	PIM13	POM13	PDIDIS13	—	✓	✓	✓	✓	—
	4	PM14	P14	PU14	PIM14	POM14	PDIDIS14	—	✓	✓	✓	✓	—
	5	PM15	P15	PU15	PIM15	POM15	PDIDIS15	—	✓	✓	✓	✓	—
	6	PM16	P16	PU16	PIM16	—	—	—	✓	✓	✓	✓	✓
	7	PM17	P17	PU17	PIM17	POM17	PDIDIS17	—	✓	✓	✓	✓	✓
Port 2	0	PM20	P20	—	—	—	—	PMCA20	✓	✓	✓	✓	✓
	1	PM21	P21	—	—	—	—	PMCA21	✓	✓	✓	✓	✓
	2	PM22	P22	—	—	—	—	PMCA22	✓	✓	✓	✓	✓
	3	PM23	P23	—	—	—	—	PMCA23	✓	✓	✓	✓	—
	4	PM24	P24	—	—	—	—	PMCA24	✓	✓	✓	—	—
	5	PM25	P25	—	—	—	—	PMCA25	✓	✓	✓	—	—
	6	PM26	P26	—	—	—	—	PMCA26	✓	✓	✓	—	—
	7	PM27	P27	—	—	—	—	PMCA27	✓	✓	✓	—	—
Port 3	0	PM30	P30	PU30	—	—	—	—	✓	✓	✓	✓	✓
	1	PM31	P31	PU31	—	POM31	PDIDIS31	—	✓	✓	✓	✓	—
	2	—	—	—	—	—	—	—	—	—	—	—	—
	3	PM33	P33	PU33	—	POM33	PDIDIS33	—	✓	✓	✓	—	—
	4	PM34	P34	PU34	PIM34	—	—	—	✓	✓	✓	✓	—
	5	—	—	—	—	—	—	—	—	—	—	—	—
	6	—	—	—	—	—	—	—	—	—	—	—	—
	7	—	—	—	—	—	—	—	—	—	—	—	—

Table 4 - 2 PMxx, Pxx, PUxx, PIMxx, POMxx, PDIDISxx, and PMCAxx Registers and the Bits Implemented in Each Product (2/3)

Port		Bit name							64 pins	48 pins	44 pins	32 pins	20 pins
		PMxx register	Pxx register	PUxx register	PIMxx register	POMxx register	PDIDISxx register	PMCAxx register					
Port 4	0	PM40	P40	PU40	—	—	—	—	✓	✓	✓	✓	✓
	1	PM41	P41	PU41	—	—	—	—	✓	✓	✓	—	—
	2	PM42	P42	PU42	—	—	—	—	✓	—	—	—	—
	3	PM43	P43	PU43	—	—	—	—	✓	—	—	—	—
	4	—	—	—	—	—	—	—	—	—	—	—	—
	5	—	—	—	—	—	—	—	—	—	—	—	—
	6	—	—	—	—	—	—	—	—	—	—	—	—
	7	—	—	—	—	—	—	—	—	—	—	—	—
Port 5	0	PM50	P50	PU50	PIM50	POM50	PDIDIS50	—	✓	✓	✓	✓	—
	1	PM51	P51	PU51	—	POM51	PDIDIS51	—	✓	✓	✓	✓	—
	2	PM52	P52	PU52	—	—	—	—	✓	—	—	—	—
	3	PM53	P53	PU53	—	—	—	—	✓	—	—	—	—
	4	PM54	P54	PU54	—	—	—	—	✓	—	—	—	—
	5	PM55	P55	PU55	PIM55	POM55	PDIDIS55	—	✓	—	—	—	—
	6	—	—	—	—	—	—	—	—	—	—	—	—
	7	—	—	—	—	—	—	—	—	—	—	—	—
Port 6	0	PM60	P60	—	—	—	—	—	✓	✓	✓	✓	—
	1	PM61	P61	—	—	—	—	—	✓	✓	✓	✓	—
	2	—	—	—	—	—	—	—	—	—	—	—	—
	3	—	—	—	—	—	—	—	—	—	—	—	—
	4	—	—	—	—	—	—	—	—	—	—	—	—
	5	—	—	—	—	—	—	—	—	—	—	—	—
	6	—	—	—	—	—	—	—	—	—	—	—	—
	7	—	—	—	—	—	—	—	—	—	—	—	—
Port 7	0	PM70	P70	PU70	—	—	—	—	✓	✓	✓	✓	—
	1	PM71	P71	PU71	PIM71	POM71	PDIDIS71	—	✓	✓	✓	—	—
	2	PM72	P72	PU72	—	POM72	PDIDIS72	—	✓	✓	✓	—	—
	3	PM73	P73	PU73	—	—	—	—	✓	✓	✓	—	—
	4	PM74	P74	PU74	—	POM74	PDIDIS74	—	✓	✓	—	—	—
	5	PM75	P75	PU75	—	—	—	—	✓	✓	—	—	—
	6	PM76	P76	PU76	—	—	—	—	✓	—	—	—	—
	7	PM77	P77	PU77	—	—	—	—	✓	—	—	—	—

Table 4 - 2 PMxx, Pxx, PUxx, PIMxx, POMxx, PDIDISxx, and PMCAxx Registers and the Bits Implemented in Each Product (3/3)

Port		Bit name							64 pins	48 pins	44 pins	32 pins	20 pins
		PMxx register	Pxx register	PUxx register	PIMxx register	POMxx register	PDIDISxx register	PMCAxx register					
Port 12	0	PM120	P120	PU120	—	—	—	PMCA120	✓	✓	✓	✓	—
	1	PM121	P121	PU121	—	—	—	—	✓	✓	✓	✓	✓
	2	PM122	P122	PU122	—	—	—	—	✓	✓	✓	✓	✓
	3	PM123	P123	PU123	—	—	—	—	✓	✓	✓	—	—
	4	PM124	P124	PU124	—	—	—	—	✓	✓	✓	—	—
	5	PM125	P125	PU125	—	—	—	—	✓	✓	✓	✓	✓
	6	—	—	—	—	—	—	—	—	—	—	—	—
	7	—	—	—	—	—	—	—	—	—	—	—	—
Port 13	0	—	P130	—	—	—	—	—	✓	✓	—	—	—
	1	—	—	—	—	—	—	—	—	—	—	—	—
	2	—	—	—	—	—	—	—	—	—	—	—	—
	3	—	—	—	—	—	—	—	—	—	—	—	—
	4	—	—	—	—	—	—	—	—	—	—	—	—
	5	—	—	—	—	—	—	—	—	—	—	—	—
	6	—	—	—	—	—	—	—	—	—	—	—	—
	7	—	P137	PU137	—	—	PDIDIS137	—	✓	✓	✓	✓	✓
Port 14	0	PM140	P140	PU140	—	—	—	—	✓	✓	—	—	—
	1	PM141	P141	PU141	—	—	—	—	✓	—	—	—	—
	2	—	—	—	—	—	—	—	—	—	—	—	—
	3	—	—	—	—	—	—	—	—	—	—	—	—
	4	—	—	—	—	—	—	—	—	—	—	—	—
	5	—	—	—	—	—	—	—	—	—	—	—	—
	6	PM146	P146	PU146	—	—	—	—	✓	✓	✓	—	—
	7	PM147	P147	PU147	—	—	—	PMCA147	✓	✓	✓	✓	✓

Note This bit is only present in the 20- and 32-pin products.

4.3.1 Port mode registers (PMxx)

The PMxx registers specify input or output mode for the ports in 1-bit units. These registers can be set by a 1-bit or 8-bit memory manipulation instruction. The value of each PMxx register following a reset is FFH. To use an alternate function of a port pin, set the port mode register by referencing **4.5 Register Settings When Using Alternate Function**.

Figure 4 - 1 Format of Port Mode Registers (PMxx)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PM0	1	PM06	PM05	PM04	PM03	PM02	PM01	PM00	FFF20H	FFH	R/W
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10	FFF21H	FFH	R/W
PM2	PM27	PM26	PM25	PM24	PM23	PM22	PM21	PM20	FFF22H	FFH	R/W
PM3	1	1	1	PM34	PM33	1	PM31	PM30	FFF23H	FFH	R/W
PM4	1	1	1	1	PM43	PM42	PM41	PM40	FFF24H	FFH	R/W
PM5	1	1	PM55	PM54	PM53	PM52	PM51	PM50	FFF25H	FFH	R/W
PM6	1	1	1	1	1	1	PM61	PM60	FFF26H	FFH	R/W
PM7	PM77	PM76	PM75	PM74	PM73	PM72	PM71	PM70	FFF27H	FFH	R/W
PM12	1	1	PM125	PM124	PM123	PM122	PM121	PM120	FFF2CH	FFH	R/W
PM14	PM147	PM146	1	1	1	1	PM141	PM140	FFF2EH	FFH	R/W

PMmn	Pmn pin I/O mode selection (m = 0 to 7, 12, 14; n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

Caution Be sure to set bits that are not implemented to their initial values.

4.3.2 Port registers (Pxx)

The Pxx registers set the output latch values of ports. If data is read in input mode, the pin level is read. If data is read in output mode, the output latch value of the port is read**Note**. The Pxx registers can be set by a 1-bit or 8-bit memory manipulation instruction. The value of each Pxx register following a reset is 00H.

Note If P00 to P03, P20 to P27, P120, or P147 is set up as an analog function port, when the port is read in the input mode, 0 is always returned instead of the pin level. When the P125/$\overline{\text{RESET}}$/INTP9 pin is set for use as the $\overline{\text{RESET}}$ input (PORTSELB = 1) in the user option byte (000C1H), P125 is always read as 1.

Figure 4 - 2 Format of Port Registers (Pxx)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
P0	0	P06	P05	P04	P03	P02	P01	P00	FFF00H	00H (output latch)	R/W
P1	P17	P16	P15	P14	P13	P12	P11	P10	FFF01H	00H (output latch)	R/W
P2	P27	P26	P25	P24	P23	P22	P21	P20	FFF02H	00H (output latch)	R/W
P3	0	0	0	P34	P33	0	P31	P30	FFF03H	00H (output latch)	R/W
P4	0	0	0	0	P43	P42	P41	P40	FFF04H	00H (output latch)	R/W
P5	0	0	P55	P54	P53	P52	P51	P50	FFF05H	00H (output latch)	R/W
P6	0	0	0	0	0	0	P61	P60	FFF06H	00H (output latch)	R/W
P7	P77	P76	P75	P74	P73	P72	P71	P70	FFF07H	00H (output latch)	R/W
P12	0	0	P125	P124	P123	P122	P121	P120	FFF0CH	00H (output latch)	R/W
P13	P137	0	0	0	0	0	0	P130	FFF0DH	**Note 1**	R/W**Note 2**
P14	P147	P146	0	0	0	0	P141	P140	FFF0EH	00H (output latch)	R/W

Pmn	Output data control (in output mode)	Input data read (in input mode)
0	Output 0	Input low level
1	Output 1	Input high level

Note 1. P137: Undefined
P130: 0 (output latch)

Note 2. P137 is only readable.

Caution **Be sure to set bits that are not implemented to their initial values.**

Remark m = 0 to 7, 12 to 14; n = 0 to 7

4.3.3 Pull-up resistor option registers (PUxx)

The PUxx registers specify whether to use the on-chip pull-up resistors. On-chip pull-up resistors can be used in 1-bit units only for the bits set to both normal output mode (POMmn = 0) and input mode (PMmn = 1) for the pins to which the use of an on-chip pull-up resistor has been specified in these registers. On-chip pull-up resistors are not connected to bits set to output mode and bits used as alternate-function output pins, regardless of the settings of these registers.
The PUxx registers can be set by a 1-bit or 8-bit memory manipulation instruction. The value of PU4 following a reset is 01H, that of PU12 is 20H, and the other PUxx registers have the value 00H.

Caution When data is input from a device operating at a different voltage to the TTL buffer for a port with the PIMxx register, set PUmn = 0 and pull up to the power supply of the device operating at a different voltage via an external resistor.

Figure 4 - 3 Format of Pull-up Resistor Option Registers (PUxx)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PU0	0	PU06	PU05	PU04	PU03	PU02	PU01	PU00	F0030H	00H	R/W
PU1	PU17	PU16	PU15	PU14	PU13	PU12	PU11	PU10	F0031H	00H	R/W
PU3	0	0	0	PU34	PU33	0	PU31	PU30	F0033H	00H	R/W
PU4	0	0	0	0	PU43	PU42	PU41	PU40	F0034H	01H	R/W
PU5	0	0	PU55	PU54	PU53	PU52	PU51	PU50	F0035H	00H	R/W
PU7	PU77	PU76	PU75	PU74	PU73	PU72	PU71	PU70	F0037H	00H	R/W
PU12	0	0	PU125	PU124	PU123	PU122	PU121	PU120	F003CH	20H	R/W
PU13	PU137	0	0	0	0	0	0	0	F003DH	00H	R/W
PU14	PU147	PU146	0	0	0	0	PU141	PU140	F003EH	00H	R/W

PUmn	Pmn pin on-chip pull-up resistor selection (m = 0, 1, 3 to 5, 7, 12 to 14; n = 0 to 7)
0	On-chip pull-up resistor not connected
1	On-chip pull-up resistor connected

Caution 1. Be sure to set bits that are not implemented to their initial values.

Caution 2. PU125 can only be used when P125/INTP9 is selected (PORTSELB = 0) in the user option byte (000C1H). When the $\overline{\text{RESET}}$ function is selected (PORTSELB = 1) in the user option byte (000C1H), the on-chip pull-up resistor is always enabled.

4.3.4 Port input mode registers (PIMxx)

The PIMxx registers set the input buffer in 1-bit units. TTL input buffer can be selected during serial communication with an external device operating at a different voltage. The PIMxx registers can be set by a 1-bit or 8-bit memory manipulation instruction. The value of each PIMxx register following a reset is 00H.

Figure 4 - 4 Format of Port Input Mode Registers (PIMxx)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PIM0	0	0	0	PIM04	PIM03	0	PIM01	0	F0040H	00H	R/W
PIM1	PIM17	PIM16	PIM15	PIM14	PIM13	0	PIM11	PIM10	F0041H	00H	R/W
PIM3	0	0	0	PIM34	0	0	0	0	F0043H	00H	R/W
PIM5	0	0	PIM55	0	0	0	0	PIM50	F0045H	00H	R/W
PIM7	0	0	0	0	0	0	PIM71	0	F0047H	00H	R/W

PIMmn	Pmn pin input buffer selection (m = 0, 1, 3, 5, 7; n = 0 to 7)
0	Normal input buffer
1	TTL input buffer

Caution Be sure to set bits that are not implemented to their initial values.

4.3.5 Port output mode registers (POMxx)

The POMxx registers set the output mode in 1-bit units. N-ch open-drain output (withstand voltage of V_{DD}[Note 1]/ withstand voltage of EV_{DD}[Note 2]) mode can be selected during serial communication with an external device operating at a different voltage, and for the SDA00, SDA01, SDA10, SDA11, SDA20, and SDA21 pins during simplified I^2C communication with an external device operating at the same voltage. In addition, POMxx registers are used in combination with PUxx registers to specify whether to use on-chip pull-up resistors. The POMxx registers can be set by a 1-bit or 8-bit memory manipulation instruction. The value of each POMxx register following a reset is 00H.

Caution An on-chip pull-up resistor is not connected to a bit for which N-ch open-drain output (withstand voltage of V_{DD}[Note 1]/withstand voltage of EV_{DD}[Note 2]) mode (POMmn = 1) is set.

Note 1. This applies to the 20- to 48-pin products.
Note 2. This applies to the 64-pin products.

Figure 4 - 5 Format of Port Output Mode Registers (POMxx)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
POM0	0	0	0	POM04	POM03	POM02	0	POM00	F0050H	00H	R/W
POM1	POM17	0	POM15	POM14	POM13	POM12	POM11	POM10	F0051H	00H	R/W
POM3	0	0	0	0	POM33	0	POM31	0	F0053H	00H	R/W
POM5	0	0	POM55	0	0	0	POM51	POM50	F0055H	00H	R/W
POM7	0	0	0	POM74	0	POM72	POM71	0	F0057H	00H	R/W

POMmn	Pmn pin output mode selection (m = 0, 1, 3, 5, 7; n = 0 to 7)
0	Normal output mode
1	N-ch open-drain output (withstand voltage of V_{DD}[Note 1]/withstand voltage of EV_{DD}[Note 2]) mode

Note 1. This applies to the 20- to 48-pin products.
Note 2. This applies to the 64-pin products.

Caution Be sure to set bits that are not implemented to their initial values.

4.3.6 Port digital input disable registers (PDIDISxx)

The PDIDISxx registers are used to prevent flows of through-current into input buffers. When N-ch open-drain output is selected for serial communications with an external device operating at a different voltage or an input port is not to be used, power consumption can be lowered by setting the corresponding bit in the given PDIDISxx register to 1. The PDIDISxx registers can be set by a 1-bit or 8-bit memory manipulation instruction. The value of each PDIDISxx register following a reset is 00H.

Figure 4 - 6 Format of Port Digital Input Disable Registers (PDIDISxx)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PDIDIS0	0	0	0	PDIDIS04	PDIDIS03	PDIDIS02	0	PDIDIS00	F02B0H	00H	R/W
PDIDIS1	PDIDIS17	0	PDIDIS15	PDIDIS14	PDIDIS13	PDIDIS12	PDIDIS11	PDIDIS10	F02B1H	00H	R/W
PDIDIS3	0	0	0	0	PDIDIS33	0	PDIDIS31	0	F02B3H	00H	R/W
PDIDIS5	0	0	PDIDIS55	0	0	0	PDIDIS51	PDIDIS50	F02B5H	00H	R/W
PDIDIS7	0	0	0	PDIDIS74	0	PDIDIS72	PDIDIS71	0	F02B7H	00H	R/W
PDIDIS13	PDIDIS137	0	0	0	0	0	0	0	F02BDH	00H	R/W

PDIDISmn	Setting of input buffers (m = 0, 1, 3, 5, 7, 13; n = 0 to 7)
0	Input to the input buffers is enabled.
1	Input to the input buffers is disabled. This prevents flows of through-current into the input buffers.

Caution Be sure to set bits that are not implemented to their initial values.

4.3.7 Port mode control A registers (PMCAxx)

The PMCAxx registers specify the digital I/O or analog function in 1-bit units. These registers can be set by a 1-bit or 8-bit memory manipulation instruction. The value of each PMCAxx register following a reset is FFH.

Figure 4 - 7 Format of Port Mode Control A Registers (PMCAxx)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PMCA0	1	1	1	1	PMCA03	PMCA02	PMCA01	PMCA00	F0060H	FFH	R/W
PMCA2	PMCA27	PMCA26	PMCA25	PMCA24	PMCA23	PMCA22	PMCA21	PMCA20	F0062H	FFH	R/W
PMCA12	1	1	1	1	1	1	1	PMCA120	F006CH	FFH	R/W
PMCA14	PMCA147	1	1	1	1	1	1	1	F006EH	FFH	R/W

PMCAmn	Selection of digital I/O or analog function for Pmn pin (m = 0, 2, 12, 14; n = 0 to 7)
0	Digital I/O
1	Analog function

Caution 1. Select the input mode by using the port mode register (PM0, PM2, PM12, or PM14) for a port pin which is set as an analog function by the corresponding PMCAxx register.

Caution 2. Do not set the pin that is specified as digital I/O by the PMCAxx register to the analog function by the analog input channel specification register (ADS).

Caution 3. Be sure to set bits that are not implemented to their initial values.

4.3.8 Peripheral I/O redirection registers (PIORx)

The PIORx registers are used to specify whether to enable or disable the peripheral I/O redirection function. This function is used to switch alternate functions that are multiplexed with port pins to replace the port functions of the pins. Use the PIORx registers to assign the target alternate function for redirection to a selected port pin before enabling the function. Note that the redirection setting cannot be changed once the operation of a function has been enabled. The PIORx registers can be set by an 8-bit memory manipulation instruction. The value of each PIORx register following a reset is 00H.

Figure 4 - 8 Format of Peripheral I/O Redirection Register (PIOR0)

Address: F0077H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
PIOR0	0	0	0	PIOR04	0	PIOR02	PIOR01	PIOR00

Bit	Alternative function	64-pin		48-pin		44-pin		32-pin	
		Setting value		Setting value		Setting value		Setting value	
		0	1	0	1	0	1	0	1
PIOR04	INTP5	P16	P12	These functions are not available for use. Set this bit to 0 (default value).					
PIOR02	SCLA0	P60	P14	P60	P14	P60	P14	P60	P14
	SDAA0	P61	P13	P61	P13	P61	P13	P61	P13
PIOR01	INTP10	P76	P52	—	—	—	—	—	—
	INTP11	P77	P53	—	—	—	—	—	—
	TxD0	P12	P17	P12	P17	P12	P17	P12	P17
	RxD0	P11	P16	P11	P16	P11	P16	P11	P16
	SCL00	P10	—	P10	—	P10	—	P10	—
	SDA00	P11	—	P11	—	P11	—	P11	—
	SI00	P11	P16	P11	—	P11	—	P11	—
	SO00	P12	P17	P12	—	P12	—	P12	—
	SCK00	P10	P55	P10	—	P10	—	P10	—
PIOR00	TI02/TO02	P17	P15	P17	P15	P17	P15	P17	P15
	TI03/TO03	P31	P14	P31	P14	P31	P14	P31	P14
	TI04/TO04	P42	P13	—	P13	—	P13	—	P13
	TI05/TO05	P05	P12	—	P12	—	P12	—	P12
	TI06/TO06	P06	P11	—	P11	—	P11	—	P11
	TI07/TO07	P41	P10	P41	P10	P41	P10	—	P10

Caution **The PIOR0 register is not present in the 20-pin products.**

Remark —: These functions are not available for use.

Figure 4 - 9 Format of Peripheral I/O Redirection Register (PIOR1)

Address: F007AH
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
PIOR1	PIOR17	PIOR16	PIOR15	PIOR14	PIOR13	PIOR12	PIOR11	PIOR10

Bit	Alternative function	64-pin				48-pin				44-pin				32-pin			
		Setting value				Setting value				Setting value				Setting value			
		00	01	10	11	00	01	10	11	00	01	10	11	00	01	10	11
PIOR17, PIOR16	RxDA0	P71	P124	P01	P34	P71	P124	P01	P34	P71	P124	P01	P34	P01	P34	These functions are not available for use. Set this bit to 0 (default value).	
	TxDA0	P72	P123	P00	P33	P72	P123	P00	P33	P72	P123	P00	P33	P00	P31		
PIOR15, PIOR14	TxD2	P13	P121	P51	P77	P13	P121	P51	—	P13	P121	P51	—	P13	P121	P51	—
	RxD2	P14	P122	P50	P76	P14	P122	P50	—	P14	P122	P50	—	P14	P122	P50	—
	SCL20	P15	—	—	—	P15	—	—	—	P15	—	—	—	P15	—	—	—
	SDA20	P14	—	—	—	P14	—	—	—	P14	—	—	—	P14	—	—	—
	SI20	P14	—	—	—	P14	—	—	—	P14	—	—	—	P14	—	—	—
	SO20	P13	—	—	—	P13	—	—	—	P13	—	—	—	P13	—	—	—
	SCK20	P15	—	—	—	P15	—	—	—	P15	—	—	—	P15	—	—	—

Bit	Alternative function	64-pin		48-pin		44-pin		32-pin	
		Setting value		Setting value		Setting value		Setting value	
		0	1	0	1	0	1	0	1
PIOR13	TxD1	P02	P00	These functions are not available for use. Set this bit to 0 (default value).					
	RxD1	P03	P01						
	SCL10	P04	—						
	SDA10	P03	—						
	SI10	P03	—						
	SO10	P02	—						
	SCK10	P04	—						
PIOR12	INTP9	P75	P125	P75	P125				
PIOR11	INTP8	P74	P121	P74	P121				
PIOR10	INTP6	P140	P122	P140	P122				

Caution **The PIOR1 register is not present in the 20-pin products.**

Remark —: These functions are not available for use.

4.3.9 Global digital input disable register (GDIDIS)

The GDIDIS register is used to prevent flows of through-current into the input buffers of input ports which use EV_{DD} as the power supply when the EV_{DD} power supply is turned off. When none of the I/O ports using EV_{DD} as the power supply are in use, power consumption can be lowered by setting the GDIDIS0 bit to 1 and then turning the EV_{DD} power supply off. Setting the GDIDIS0 bit to 1 prohibits input to all input buffers using EV_{DD} as the power supply, preventing flows of through-current while the EV_{DD} power supply is turned off. The GDIDIS register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Remark The GDIDIS register is equipped with the 64-pin products.

Figure 4 - 10 Format of Global Digital Input Disable Register (GDIDIS)

Address: F007DH
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
GDIDIS	0	0	0	0	0	0	0	GDIDIS0

GDIDIS0	Setting of input buffers using EV_{DD} power supply
0	Input to the input buffers is enabled.
1	Input to the input buffers is disabled. This prevents flows of through-current into the input buffers.

Remark Even when input to input buffers is prohibited (GDIDIS0 = 1), peripheral functions which do not use port functions having EV_{DD} as the power supply can be used.

Turn off the EV_{DD} power supply with the following procedure.

<1> Disable input to input buffers (set GDIDIS0 = 1).
<2> Turn off the EV_{DD} power supply.

Turn on again the EV_{DD} power supply with the following procedure.

<1> Turn on the EV_{DD} power supply.
<2> Enable input to input buffers (set GDIDIS0 = 0).

Caution 1. Do not input an input voltage equal to or greater than EV_{DD} to an input port that uses EV_{DD} as the power supply.

Caution 2. When input to input buffers is prohibited (GDIDIS0 = 1), the value read from the port register (Pxx) of a port that uses EV_{DD} as the power supply is 1. When 1 is set in a port output mode register (POMxx) (N-ch open-drain output (withstand voltage of EV_{DD}) mode), the value read from the corresponding port register (Pxx) is 0.

4.3.10 Port mode select register (PMS)

The PMS register is used to specify whether the value in the output latch for a port is read or the output level on a port pin is read when the pin is in output mode (the PMmn bit of the corresponding port mode register (PMxx) is 0). For details, see **20.3.11.1 Port mode select register (PMS)**.

4.4 Port Function Operations

Port operations differ depending on whether the input or output mode is selected, as shown below.

4.4.1 Writing to I/O port

1. Output mode
   A value is written to the output latch by a transfer instruction, and the output latch contents are output from the pin. Once data is written to the output latch, it is retained until data is written to the output latch again. The data of the output latch is cleared when a reset signal is generated.
2. Input mode
   A value is written to the output latch by a transfer instruction, but since the output buffer is off, the pin state does not change. Therefore, byte data can be written to the ports used for both input and output. Once data is written to the output latch, it is retained until data is written to the output latch again. The data of the output latch is cleared when a reset signal is generated.

4.4.2 Reading from I/O port

1. Output mode
   The output latch contents are read by a transfer instruction. The output latch contents do not change.
2. Input mode
   The pin state is read by a transfer instruction. The output latch contents do not change.

4.4.3 Operations on I/O port

1. Output mode
   An operation is performed on the output latch contents, and the result is written to the output latch. The output latch contents are output from the pins. Once data is written to the output latch, it is retained until data is written to the output latch again. The data of the output latch is cleared when a reset signal is generated.
2. Input mode
   The pin level is read and an operation is performed on the read value. The result of the operation is written to the output latch, but since the output buffer is off, the pin state does not change. Therefore, byte data can be written to the ports used for both input and output. The data of the output latch is cleared when a reset signal is generated.

4.4.4 Communications with devices operating at a different voltage (1.8 V, 2.5 V, or 3 V) by switching the EV_{DD} voltage

Communications with external devices operating at a different voltage (1.8 V, 2.5 V, or 3 V) to this device is possible by switching EV_{DD}**Note** through the general-purpose I/O pins, in accord with the power supply of the device to be connected.

Note $EV_{DD} \leq V_{DD}$ must be satisfied.

4.4.5 Communications with devices operating at a different voltage (1.8 V, 2.5 V, or 3 V) by switching I/O buffers

The port input mode registers (PIMxx) and port output mode registers (POMxx) can be used to switch the I/O buffers to enable communications with external devices that have different operating voltages (1.8 V, 2.5 V, or 3 V). To receive input from external devices operating at different voltages, set the relevant bits of the port input mode registers (PIM0, PIM1, PIM3, PIM5, and PIM7) in 1-bit units to switch from normal input (CMOS) to the TTL input buffers. To output data to external devices operating at different voltages, set the relevant bits of the port output mode registers (POM0, POM1, POM3, POM5, and POM7) in 1-bit units to switch from normal output (CMOS) to N-ch open-drain output (withstand voltage of V_{DD}**Note 1**/withstand voltage of EV_{DD}**Note 2**) mode. Setting the port digital input disable registers (PDIDIS0, PDIDIS1, PDIDIS3, PDIDIS5, PDIDIS7, and PDIDIS13) bit-by-bit can prevent the flow of through currents to the corresponding input buffers. The following describes connection through a serial interface to an external device operating at a different voltage.

Note 1. This applies to the 20- to 48-pin products.
Note 2. This applies to the 64-pin products.

1. Procedure for setting input pins of UART0 to UART2, UARTA0, CSI00, CSI10, and CSI20 for use with the TTL input buffers
   Use the following port pin or pins for each interface.
   - P11 (P16) for UART0
   - P03 (P01) for UART1
   - P14 (P50) for UART2
   - P71 (P01 and P34) for UARTA0
   - P10 and P11 (P55 and P16) for CSI00
   - P03 and P04 for CSI10
   - P14 and P15 for CSI20

   Remark Pin functions in parentheses can be assigned via settings in the corresponding peripheral I/O redirection registers (PIORx). For details, see **4.3.8 Peripheral I/O redirection registers (PIORx)**.

   <1> Pull up the input pin to be used to the voltage of the target device via an external resistor. The on-chip pull-up resistor cannot be used for this purpose.
   <2> Set the corresponding bit of the PIM0, PIM1, PIM3, PIM5, and PIM7 registers to 1 to switch to the TTL input buffer. For V_{IH} and V_{IL}, refer to the DC characteristics when the TTL input buffer is selected.
   <3> Enable the operation of the serial array unit and set the mode to the UART/simplified SPI (CSI**Note**) mode.

   Note Although the CSI function is generally called SPI, it is also called CSI in this product, so it is referred to as such in this manual.

2. Procedure for setting output pins of UART0 to UART2, UARTA0, CSI00, CSI10, and CSI20 for use with the N-ch open-drain output mode
   - P12 (P17) for UART0
   - P02 (P00) for UART1
   - P13 (P51) for UART2
   - P72 (P00 and P33) for UARTA0
   - P10 and P12 (P55 and P17) for CSI00
   - P02 and P04 for CSI10
   - P13 and P15 for CSI20

   Remark Pin functions in parentheses can be assigned via settings in the corresponding peripheral I/O redirection registers (PIORx). For details, see **4.3.8 Peripheral I/O redirection registers (PIORx)**.

   <1> Pull up the input pin to be used to the voltage of the target device via an external resistor. The on-chip pull-up resistor cannot be used for this purpose.
   <2> The port pins are set for input (Hi-Z) after the reset state is released.
   <3> Set the corresponding bit of the PDIDIS0, PDIDIS1, PDIDIS3, PDIDIS5, and PDIDIS7 registers to 1 to disable input to the input buffer.
   <4> Set the output latch of the corresponding port to 1.
   <5> Set the corresponding bit of the POM0, POM1, POM3, POM5, and POM7 registers to 1 to set the N-ch open-drain output (withstand voltage of V_{DD}**Note 1**/withstand voltage of EV_{DD}**Note 2**) mode.
   <6> Enable the operation of the serial array unit and set the mode to the UART/simplified SPI (CSI**Note 3**) mode.
   <7> Set the corresponding bit of the PM0, PM1, PM3, PM5, and PM7 registers to the output mode. At this time, the output data is high level, so the pin is in the Hi-Z state.

   Note 1. This applies to the 20- to 48-pin products.
   Note 2. This applies to the 64-pin products.
   Note 3. Although the CSI function is generally called SPI, it is also called CSI in this product, so it is referred to as such in this manual.

3. Procedure for setting I/O pins of IIC00, IIC10, and IIC20 for use in connection with a device operating at a different voltage (1.8 V, 2.5 V, or 3 V)
   - P10 and P11 for IIC00
   - P03 and P04 for IIC10
   - P14 and P15 for IIC20

   Remark Pin functions in parentheses can be assigned via settings in the corresponding peripheral I/O redirection registers (PIORx). For details, see **4.3.8 Peripheral I/O redirection registers (PIORx)**.

   <1> Pull up the input pin to be used to the voltage of the target device via an external resistor. The on-chip pull-up resistor cannot be used for this purpose.
   <2> The port pins are set for input (Hi-Z) after the reset state is released.
   <3> Set the output latch of the corresponding port to 1.
   <4> Set the corresponding bit of the POM0 and POM1 registers to 1 to set the N-ch open-drain output (withstand voltage of V_{DD}**Note 1**/withstand voltage of EV_{DD}**Note 2**) mode.
   <5> Set the corresponding bit of the PIM0 and PIM1 registers to 1 to switch to the TTL input buffer. For V_{IH} and V_{IL}, refer to the DC characteristics when the TTL input buffer is selected.
   <6> Enable the operation of the serial array unit and set the mode to the simplified I^2C mode.
   <7> Set the corresponding bit of the PM0 and PM1 registers to the output mode (data I/O is possible in the output mode). At this time, the output data is high level, so the pin is in the Hi-Z state.

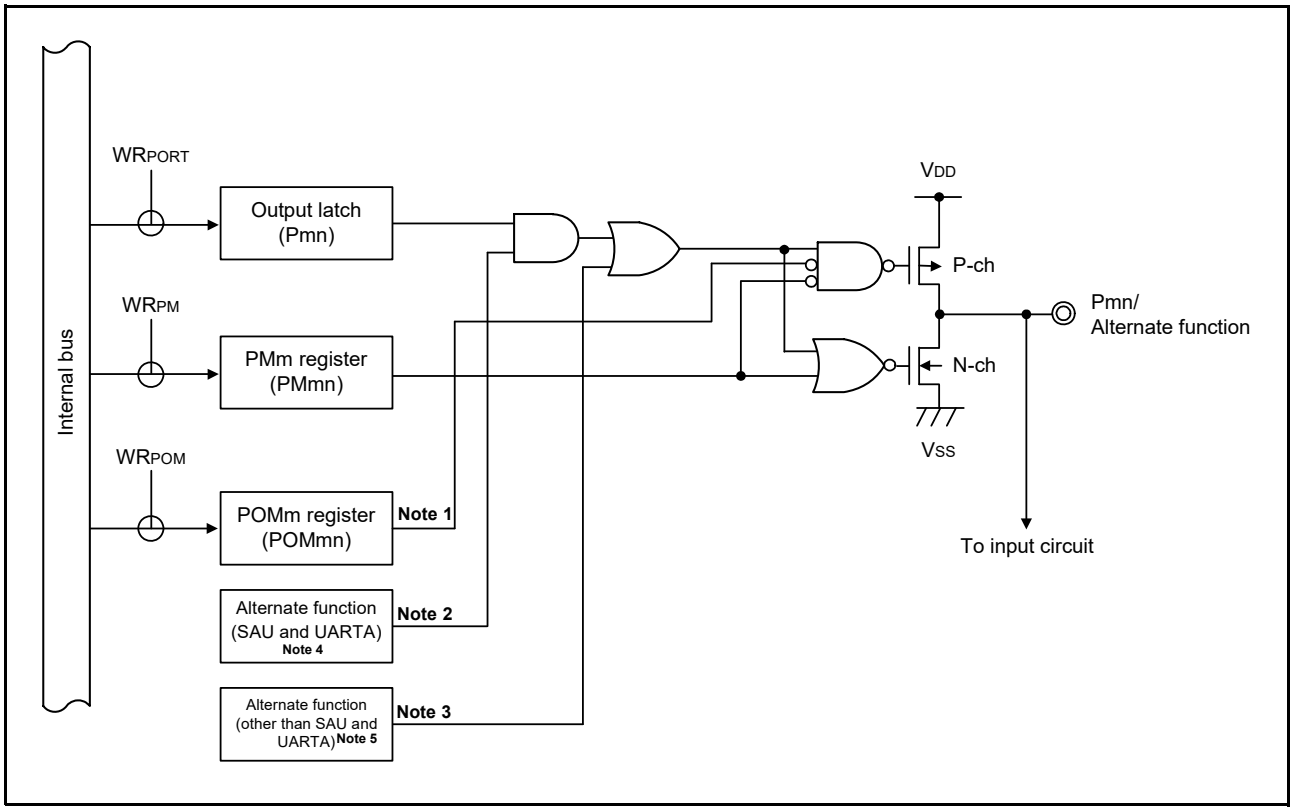
   Note 1. This applies to the 20- to 48-pin products.
   Note 2. This applies to the 64-pin products.

4.5 Register Settings When Using Alternate Function

4.5.1 Basic concept when using alternate function

In the beginning, for a pin that is also assigned to the analog function, use the corresponding port mode control A register (PMCAxx) to specify whether to use the pin for the analog function or digital input/output. **Figure 4 - 11** shows the basic configuration of the output circuit for a pin used for digital input/output. The outputs from multiplexed SAU and UARTA functions (excluding the clock output from the UARTA) and the outputs from the output latches for the port-pin functions are input to an AND gate. The output of the AND gate is input to an OR gate. The outputs of multiplexed functions other than the SAU and UARTA (timer array unit, IICA, etc.) are connected to other input pins of the OR gate. When such a pin is used for the port function or an alternate function, the unused alternate function must not hinder the output of the function to be used. **Table 4 - 3** shows a concept of basic settings for this case.

Figure 4 - 11 Basic Configuration of Output Circuit for a Pin



Note 1. For pins that do not have corresponding bits in the POMm registers, consider this signal to be 0.
Note 2. For pins that do not have multiplexed functions of the indicated types, consider this signal to be 1.
Note 3. For pins that do not have multiplexed functions of the indicated types, consider this signal to be 0.
Note 4. The UARTA clock output is not included.
Note 5. The UARTA clock output is included.

Remark m: Port number (m = 0 to 7, 12 to 14); n: Bit number (n = 0 to 7)

Table 4 - 3 Concept of Basic Settings

Output Function of Used Pin	Output Settings of Unused Alternate Function		
	Port-pins Output Function	Output Functions of the SAU and UARTA[Note 1]	Output Functions other than those of the SAU and UARTA[Note 2]
Port-pins output function	—	Output at the high level	Output at the low level
Output functions of the SAU and UARTA[Note 1]	High level	Output at the high level	Output at the low level
Output functions other than those of the SAU and UARTA[Note 2]	Low level	Don't care	Output at the low level[Note 3]

Note 1. The UARTA clock output is not included.

Note 2. The UARTA clock output is included.

Note 3. The output of the multiplexed functions which are not in use must be set to the low level because two or more output functions other than those of the SAU and UARTA may be multiplexed on the same pin. The output of the multiplexed functions which are not in use must be set to the high level because two or more output functions of the SAU and UARTA may be multiplexed on the same pin. For details on the setting method, see **4.5.2 Register settings for alternate function whose output function is not used**.

4.5.2 Register settings for alternate function whose output function is not used

When the output of an alternate function of the pin is not used, the following settings should be made. Note that when the peripheral I/O redirection function is applicable to an alternate function, the output can be switched to another pin by setting the corresponding peripheral I/O redirection register (PIORx). This allows usage of the port function or other alternate function assigned to the target pin.

1. SOp = 1, TxDq = 1 (settings when the serial output (SOp/TxDq) of SAU is not used)
   When the serial output (SOp/TxDq) is not used, such as a case in which only the serial input of SAU is used, set the bit in serial output enable register m (SOEm) which corresponds to the unused output to 0 (output disabled) and set the SOmn bit in serial output register m (SOm) to 1 (high-level output). These are the same settings as the initial state.

2. SCKp = 1, SDAr = 1, SCLr = 1 (settings when channel n in SAU is not used)
   When SAU is not used, set bit n (SEmn) in serial channel enable status register m (SEm) to 0 (operation stopped state), set the bit in serial output enable register m (SOEm) which corresponds to the unused output to 0 (output disabled), and set the SOmn and CKOmn bits in serial output register m (SOm) to 1 (high-level output). These are the same settings as the initial state.

3. TOmn = 0 (settings when the output of channel n in TAU is not used)
   When the TOmn output of TAU is not used, set the bit in timer output enable register 0 (TOE0) which corresponds to the unused output to 0 (output disabled) and set the corresponding bit in timer output register 0 (TO0) to 0 (low-level output). These are the same settings as the initial state.

4. SDAAn = 0, SCLAn = 0 (setting when IICA is not used)
   When IICA is not used, set the IICEn bit in IICA control register n0 (IICCTLn0) to 0 (operation stopped). This is the same setting as the initial state.

5. TxDAn = 1, CLKAn = 0 (settings when the UARTA is not used)
   When the UARTA is not used, set the UARTAENn, TXEAn, and RXEAn bits in operation mode setting register n0 (ASIMAn0) to 0 (operation disabled). These are the same settings as the initial state.

Remark p: CSI number (p = 00, 01, 10, 11, 20, 21), q: UART number (q = 0 to 2),
r: IIC number (r = 00, 01, 10, 11, 20, 21), n: UARTA number (n = 0)

4.5.3 Register settings and port pin state

Table 4 - 4 shows the correspondence between register settings and port pin state.

Table 4 - 4 Correspondence between Register Settings and Port Pin State

PMCAxx	PMxx	Pxx	PUxx	Pin State
1	×	×	×	Analog input/output
0	1	×	1	Pulled up
0	1	×	0	Hi-Z
0	0	1	×	Port output (high level)
0	0	0	×	Port output (low level)

Remark ×: Don’t care

4.5.4 Examples of register settings for port and alternate functions

Table 4 - 5 shows examples of register settings for port and alternate functions. The registers used to control the port functions should be set as shown in **Table 4 - 5**. See the following remark for legends.

Caution **In examples of register settings, the port digital input disable register (PDIDISxx) is set for input to the input buffer being enabled.**

Remark —: Not supported
×: Don’t care
PIORx: Peripheral I/O redirection registers
POMxx: Port output mode registers
PMCAxx: Port mode control A registers
PMxx: Port mode registers
Pxx: Port output latch
Functions in parentheses can be assigned via settings in the corresponding peripheral I/O redirection registers (PIORx).

Table 4 - 5 Examples of Register and Output Latch Settings for Alternate Functions (1/17)

Pin Name	Function Used		PIORx	POMxx	PMCAxx	PMxx	Pxx	Alternate Function Output		20-pin	32-pin	44-pin	48-pin	64-pin
	Function Name	I/O						SAU and UARTA (excluding clock output from the UARTA)	Other than SAU and UARTA (including clock output from the UARTA)					
P00	P00	Input	—	×	0 Note 1	1	×	×	—	✓	✓	✓	✓	✓
		Output	—	0	0 Note 1	0	0/1	TxD1 = 1 (TxD1) = 1 Note 3 TxDA0 = 1 (TxDA0) = 1 Note 4	—					
		N-ch open-drain output	—	1	0 Note 1	0	0/1							
	ANI17	Analog input	—	×	1	1	×	×	—	✓	✓	—	—	—
	TI00	Input	—	×	0 Note 1	1	×	×	—	✓	✓	✓	✓	✓
	TxD1	Output	—	0/1	0 Note 1	0	1	TxDA0 = 1 Note 1 (TxDA0) = 1 Note 5	—	✓	✓	✓	✓	—
	(TxD1)	Output	PIOR13 = 1	0/1	0 Note 1	0	1	(TxDA0) = 1 Note 3	—	—	—	—	—	✓
	TxDA0	Output	PIOR17 = 0 PIOR16 = 0 Note 2	0/1	0 Note 1	0	1	TxD1 = 1 Note 1	—	✓	✓	—	—	—
	(TxDA0)	Output	PIOR17 = 1 PIOR16 = 0	0/1	0 Note 1	0	1	TxD1 = 1 Note 5 (TxD1) = 1 Note 3	—	—	—	✓	✓	✓
P01	P01	Input	—	—	0 Note 1	1	×	—	×	✓	✓	✓	✓	✓
		Output	—	—	0 Note 1	0	0/1	—	TO00 = 0					
	ANI16	Analog input	—	—	1	1	×	—	×	✓	✓	—	—	—
	TO00	Output	—	—	0 Note 1	0	0	—	×	✓	✓	✓	✓	✓
	RxD1	Input	—	—	0 Note 1	1	×	—	×	✓	✓	✓	✓	—
	(RxD1)	Input	PIOR13 = 1	—	0 Note 1	1	×	—	×	—	—	—	—	✓
	RxDA0	Input	PIOR17 = 0 PIOR16 = 0 Note 2	—	0 Note 1	1	×	—	×	✓	✓	—	—	—
	(RxDA0)	Input	PIOR17 = 1 PIOR16 = 0	—	0 Note 1	1	×	—	×	—	—	✓	✓	✓
P02	P02	Input	—	×	0	1	×	×	—	—	—	—	—	✓
		Output	—	0	0	0	0/1	TxD1/ SO10 = 1	—					
		N-ch open-drain output	—	1	0	0	0/1							
	ANI17	Analog input	—	×	1	1	×	×	—	—	—	—	—	✓
	TxD1	Output	PIOR13 = 0	0/1	0	0	1	×	—	—	—	—	—	✓
	SO10	Output	PIOR13 = 0	0/1	0	0	1	×	—	—	—	—	—	✓
P03	P03	Input	—	×	0	1	×	×	—	—	—	—	—	✓
		Output	—	0	0	0	0/1	SDA10 = 1	—					
		N-ch open-drain output	—	1	0	0	0/1							
	ANI16	Analog input	—	×	1	1	×	×	—	—	—	—	—	✓
	SI10	Input	PIOR13 = 0	×	0	1	×	×	—	—	—	—	—	✓
	RxD1	Input	PIOR13 = 0	×	0	1	×	×	—	—	—	—	—	✓
	SDA10	I/O	PIOR13 = 0	1	0	0	1	×	—	—	—	—	—	✓

Note 1. This setting is only applicable in the 20- and 32-pin products.

Note 2. This setting is only applicable in the 32-pin products.

Note 3. This setting is only applicable in the 64-pin products.

Note 4. This setting is only applicable in the 44- to 64-pin products.

Note 5. This setting is only applicable in the 44- and 48-pin products.

Table 4 - 5 Examples of Register and Output Latch Settings for Alternate Functions (2/17)

Pin Name	Function Used		PIORx	POMxx	PMCAxx	PMxx	Pxx	Alternate Function Output		20-pin	32-pin	44-pin	48-pin	64-pin
	Function Name	I/O						SAU and UARTA (excluding clock output from the UARTA)	Other than SAU and UARTA (including clock output from the UARTA)					
P04	P04	Input	—	×	—	1	×	×	—	—	—	—	—	✓
		Output	—	0	—	0	0/1	SCK10/ SCL10 = 1	—					
		N-ch open-drain output	—	1	—	0	0/1							
	SCK10	Input	PIOR13 = 0	×	—	1	×	×	—	—	—	—	—	✓
		Output	PIOR13 = 0	0/1	—	0	1	×	—					
	SCL10	Output	PIOR13 = 0	0/1	—	0	1	×	—	—	—	—	—	✓
P05	P05	Input	—	—	—	1	×	—	×	—	—	—	—	✓
		Output	—	—	—	0	0/1	—	TO05 = 0					
	TI05	Input	PIOR00 = 0	—	—	1	×	—	×	—	—	—	—	✓
	TO05	Output	PIOR00 = 0	—	—	0	0	—	×	—	—	—	—	✓
P06	P06	Input	—	—	—	1	×	—	×	—	—	—	—	✓
		Output	—	—	—	0	0/1	—	TO06 = 0 CLKA0 = 0					
	TI06	Input	PIOR00 = 0	—	—	1	×	—	×	—	—	—	—	✓
	TO06	Output	PIOR00 = 0	—	—	0	0	—	CLKA0 = 0	—	—	—	—	✓
	CLKA0	Output	—	—	—	0	0	—	TO06 = 0	—	—	—	—	✓

Table 4 - 5 Examples of Register and Output Latch Settings for Alternate Functions (3/17)

Pin Name	Function Used		PIORx	POMxx	PMxx	Pxx	Alternate Function Output		20-pin	32-pin	44-pin	48-pin	64-pin
	Function Name	I/O					SAU and UARTA (excluding clock output from the UARTA)	Other than SAU and UARTA (including clock output from the UARTA)					
P10	P10	Input	—	×	1	×	×	×	✓	✓	✓	✓	✓
		Output	—	0	0	0/1	SCK00/SCL00 = 1	(TO07) = 0Note 1 TO07 = 0Note 2					
		N-ch open-drain output	—	1	0	0/1							
	SCK00	Input	PIOR01 = 0Note 1	×	1	×	×	×	✓	✓	✓	✓	✓
		Output	PIOR01 = 0Note 1	0/1	0	1	×	(TO07) = 0Note 1 TO07 = 0Note 2					
	SCL00	Output	PIOR01 = 0Note 1	0/1	0	1	×	(TO07) = 0Note 1 TO07 = 0Note 2	✓	✓	✓	✓	✓
	(TI07)	Input	PIOR00 = 1	×	1	×	×	×	—	✓	✓	✓	✓
	(TO07)	Output	PIOR00 = 1	0	0	0	×	×	—	✓	✓	✓	✓
	TI07	Input	—	×	1	×	×	×	✓	—	—	—	—
	TO07	Output	—	0	0	0	×	×	✓	—	—	—	—
P11	P11	Input	—	×	1	×	×	×	✓	✓	✓	✓	✓
		Output	—	0	0	0/1	SDA00 = 1	(TO06) = 0Note 1 TO06 = 0Note 2					
		N-ch open-drain output	—	1	0	0/1							
	SI00	Input	PIOR01 = 0Note 1	×	1	×	×	×	✓	✓	✓	✓	✓
	RxD0	Input	PIOR01 = 0Note 1	×	1	×	×	×	✓	✓	✓	✓	✓
	SDA00	I/O	PIOR01 = 0Note 1	1	0	1	×	(TO06) = 0Note 1 TO06 = 0Note 2	✓	✓	✓	✓	✓
	(TI06)	Input	PIOR00 = 1	×	1	×	×	×	—	✓	✓	✓	✓
	(TO06)	Output	PIOR00 = 1	0	0	0	×	×	—	✓	✓	✓	✓
	TI06	Input	—	×	1	×	×	×	✓	—	—	—	—
	TO06	Output	—	0	0	0	×	×	✓	—	—	—	—
P12	P12	Input	—	×	1	×	×	×	✓	✓	✓	✓	✓
		Output	—	0	0	0/1	SO00/TxD0 = 1	(TO05) = 0Note 1 TO05 = 0Note 2					
		N-ch open-drain output	—	1	0	0/1							
	SO00	Output	PIOR01 = 0Note 1	0/1	0	1	×	(TO05) = 0Note 1 TO05 = 0Note 2	✓	✓	✓	✓	✓
	TxD0	Output	PIOR01 = 0Note 1	0/1	0	1	×	(TO05) = 0Note 1 TO05 = 0Note 2	✓	✓	✓	✓	✓
	(INTP5)	Input	PIOR04 = 1	×	1	×	×	×	—	—	—	—	✓
	(TI05)	Input	PIOR00 = 1	×	1	×	×	×	—	✓	✓	✓	✓
	(TO05)	Output	PIOR00 = 1	0	0	0	×	×	—	✓	✓	✓	✓
	TI05	Input	—	×	1	×	×	×	✓	—	—	—	—
	TO05	Output	—	0	0	0	×	×	✓	—	—	—	—

Note 1. This setting is only applicable in the 32- to 64-pin products.

Note 2. This setting is only applicable in the 20-pin products.

Table 4 - 5 Examples of Register and Output Latch Settings for Alternate Functions (4/17)

Pin Name	Function Used		PIORx	POMxx	PMxx	Pxx	Alternate Function Output		20-pin	32-pin	44-pin	48-pin	64-pin
	Function Name	I/O					SAU and UARTA (excluding clock output from the UARTA)	Other than SAU and UARTA (including clock output from the UARTA)					
P13	P13	Input	—	×	1	×	×	×	—	✓	✓	✓	✓
		Output	—	0	0	0/1	TxD2/SO20 = 1	(TO04) = 0 (SDAA0) = 0					
		N-ch open-drain output	—	1	0	0/1							
	TxD2	Output	PIOR15 = 0 PIOR14 = 0	0/1	0	1	×	(TO04) = 0 (SDAA0) = 0	—	✓	✓	✓	✓
	SO20	Output	PIOR15 = 0 PIOR14 = 0	0/1	0	1	×	(TO04) = 0 (SDAA0) = 0	—	✓	✓	✓	✓
	(SDAA0)	I/O	PIOR02 = 1	1	0	0	×	(TO04) = 0	—	✓	✓	✓	✓
	(TI04)	Input	PIOR00 = 1	×	1	×	×	×	—	✓	✓	✓	✓
	(TO04)	Output	PIOR00 = 1	0	0	0	×	(SDAA0) = 0	—	✓	✓	✓	✓
P14	P14	Input	—	×	1	×	×	×	—	✓	✓	✓	✓
		Output	—	0	0	0/1	SDA20 = 1	(TO03) = 0 (SCLA0) = 0					
		N-ch open-drain output	—	1	0	0/1							
	RxD2	Input	PIOR15 = 0 PIOR14 = 0	×	1	×	×	×	—	✓	✓	✓	✓
	SI20	Input	PIOR15 = 0 PIOR14 = 0	×	1	×	×	×	—	✓	✓	✓	✓
	SDA20	I/O	PIOR15 = 0 PIOR14 = 0	1	0	1	×	(TO03) = 0 (SCLA0) = 0	—	✓	✓	✓	✓
	(SCLA0)	I/O	PIOR02 = 1	1	0	0	×	(TO03) = 0	—	✓	✓	✓	✓
	(TI03)	Input	PIOR00 = 1	×	1	×	×	×	—	✓	✓	✓	✓
	(TO03)	Output	PIOR00 = 1	0	0	0	×	(SCLA0) = 0	—	✓	✓	✓	✓
P15	P15	Input	—	×	1	×	×	×	—	✓	✓	✓	✓
		Output	—	0	0	0/1	SCK20/SCL20 = 1	(TO02) = 0					
		N-ch open-drain output	—	1	0	0/1							
	SCK20	Input	PIOR15 = 0 PIOR14 = 0	×	1	×	×	×	—	✓	✓	✓	✓
		Output	PIOR15 = 0 PIOR14 = 0	0/1	0	1	×	(TO02) = 0					
	SCL20	Output	PIOR15 = 0 PIOR14 = 0	0/1	0	1	×	(TO02) = 0	—	✓	✓	✓	✓
	(TI02)	Input	PIOR00 = 1	×	1	×	×	×	—	✓	✓	✓	✓
	(TO02)	Output	PIOR00 = 1	0	0	0	×	×	—	✓	✓	✓	✓

Table 4 - 5 Examples of Register and Output Latch Settings for Alternate Functions (5/17)

Pin Name	Function Used		PIORx	POMxx	PMxx	Pxx	Alternate Function Output		20-pin	32-pin	44-pin	48-pin	64-pin
	Function Name	I/O					SAU and UARTA (excluding clock output from the UARTA)	Other than SAU and UARTA (including clock output from the UARTA)					
P16	P16	Input	—	—	1	×	×	×	✓	✓	✓	✓	✓
		Output	—	—	0	0/1	SO11 = 1 Note 2	TO01 = 0					
	TI01	Input	—	—	1	×	×	×	✓	✓	✓	✓	✓
	TO01	Output	—	—	0	0	×	×	✓	✓	✓	✓	✓
	INTP5	Input	PIOR04 = 0 Note 1	—	1	×	×	×	✓	✓	✓	✓	✓
	(SI00)	Input	PIOR01 = 1	—	1	×	×	×	—	—	—	—	✓
	(RxD0)	Input	PIOR01 = 1	—	1	×	×	×	—	✓	✓	✓	✓
	SO11	Output	—	—	0	1	×	TO01 = 0 Note 2	✓	—	—	—	—
P17	P17	Input	—	×	×	×	×	×	✓	✓	✓	✓	✓
		Output	—	0	0	0/1	(TxD0) = 1 (SO00) = 1 Note 1 SDA11 = 1 Note 2	TO02 = 0					
		N-ch open-drain output	—	1	0	0/1							
	TI02	Input	PIOR00 = 0	×	1	×	×	×	✓	✓	✓	✓	✓
	TO02	Output	PIOR00 = 0	0	0	0	×	×	✓	✓	✓	✓	✓
	(TxD0)	Output	PIOR01 = 1	0/1	0	1	×	TO02 = 0 Note 3	—	✓	✓	✓	✓
	(SO00)	Output	PIOR01 = 1	0/1	0	1	×	TO02 = 0 Note 1	—	—	—	—	✓
	SI11	Input	—	×	1	×	×	×	✓	—	—	—	—
	SDA11	I/O	—	×	0	1	×	TO02 = 0 Note 2	✓	—	—	—	—

Note 1. This setting is only applicable in the 64-pin products.

Note 2. This setting is only applicable in the 20-pin products.

Note 3. This setting is only applicable in the 32- to 64-pin products.

Table 4 - 5 Examples of Register and Output Latch Settings for Alternate Functions (6/17)

Pin Name	Function Used		PMCAxx	PMxx	Pxx	20-pin	32-pin	44-pin	48-pin	64-pin
	Function Name	I/O								
P20	P20	Input	0	1	×	✓	✓	✓	✓	✓
		Output	0	0	0/1					
	ANI0	Analog input	1	1	×	✓	✓	✓	✓	✓
	AVREFP	Reference voltage	1	1	×	✓	✓	✓	✓	✓
P21	P21	Input	0	1	×	✓	✓	✓	✓	✓
		Output	0	0	0/1					
	ANI1	Analog input	1	1	×	✓	✓	✓	✓	✓
	AVREFM	Reference voltage	1	1	×	✓	✓	✓	✓	✓
P22	P22	Input	0	1	×	✓	✓	✓	✓	✓
		Output	0	0	0/1					
	ANI2	Analog input	1	1	×	✓	✓	✓	✓	✓
P23	P23	Input	0	1	×	—	✓	✓	✓	✓
		Output	0	0	0/1					
	ANI3	Analog input	1	1	×	—	✓	✓	✓	✓
P24	P24	Input	0	1	×	—	—	✓	✓	✓
		Output	0	0	0/1					
	ANI4	Analog input	1	1	×	—	—	✓	✓	✓
P25	P25	Input	0	1	×	—	—	✓	✓	✓
		Output	0	0	0/1					
	ANI5	Analog input	1	1	×	—	—	✓	✓	✓
P26	P26	Input	0	1	×	—	—	✓	✓	✓
		Output	0	0	0/1					
	ANI6	Analog input	1	1	×	—	—	✓	✓	✓
P27	P27	Input	0	1	×	—	—	✓	✓	✓
		Output	0	0	0/1					
	ANI7	Analog input	1	1	×	—	—	✓	✓	✓

Table 4 - 5 Examples of Register and Output Latch Settings for Alternate Functions (7/17)

Pin Name	Function Used		PIORx	POMxx	PMxx	Pxx	Alternate Function Output		20-pin	32-pin	44-pin	48-pin	64-pin
	Function Name	I/O					SAU and UARTA (excluding clock output from the UARTA)	Other than SAU and UARTA (including clock output from the UARTA)					
P30	P30	Input	—	—	1	×	×	×	✓	✓	✓	✓	✓
		Output	—	—	0	0/1	SCK11/SCL11 = 1	×					
	INTP3	Input	—	—	1	×	×	×	✓	✓	✓	✓	✓
	SCK11	Input	—	—	1	×	×	×	✓	✓	✓	✓	✓
		Output	—	—	0	1	×	×					
	SCL11	Output	—	—	0	1	×	×	✓	✓	✓	✓	✓
P31	P31	Input	—	×	1	×	—	×	—	✓	✓	✓	✓
		Output	—	0	0	0/1	(TxDA0) =1Note 2	TO03 = 0					
		N-ch open-drain output	—	1	0	0/1							
	TI03	Input	PIOR00 = 0	×	1	×	×	×	—	✓	✓	✓	✓
	TO03	Output	PIOR00 = 0	0	0	0	×	×	—	✓	✓	✓	✓
	INTP4	Input	—	×	1	×	×	×	—	✓	✓	✓	✓
	(TxDA0)	Output	PIOR17 = 0 PIOR16 = 1	0/1	0	1	×	TO03 = 0Note 2	—	✓	—	—	—
P33	P33	Input	—	×	1	×	×	×	—	—	✓	✓	✓
		Output	—	0	0	0/1	(TxDA0) = 1	×					
		N-ch open-drain output	—	1	0	0/1							
	(TxDA0)	Output	PIOR17 = 1 PIOR16 = 1	0/1	0	1	×	×	—	—	✓	✓	✓
P34	P34	Input	—	—	1	×	—	—	—	✓	✓	✓	✓
		Output	—	—	0	0/1	—	—					
	(RxDA0)	Input	PIOR17 = 1 PIOR16 = 1Note 1 PIOR17 = 0 PIOR16 = 1Note 2	—	1	×	—	—	—	✓	✓	✓	✓

Note 1. This setting is only applicable in the 44- to 64-pin products.

Note 2. This setting is only applicable in the 32-pin products.

Table 4 - 5 Examples of Register and Output Latch Settings for Alternate Functions (8/17)

Pin Name	Function Used		PIORx	PMxx	Pxx	Alternate Function Output		20-pin	32-pin	44-pin	48-pin	64-pin
	Function Name	I/O				SAU and UARTA (excluding clock output from the UARTA)	Other than SAU and UARTA (including clock output from the UARTA)					
P40	P40	Input	—	1	×	—	—	✓	✓	✓	✓	✓
		Output	—	0	0/1	—	—					
P41	P41	Input	—	1	×	—	×	—	—	✓	✓	✓
		Output	—	0	0/1	—	TO07 = 0					
	TI07	Input	PIOR00 = 0	1	×	—	×	—	—	✓	✓	✓
	TO07	Output	PIOR00 = 0	0	0	—	×	—	—	✓	✓	✓
P42	P42	Input	—	1	×	—	×	—	—	—	—	✓
		Output	—	0	0/1	—	TO04 = 0					
	TI04	Input	PIOR00 = 0	1	×	—	×	—	—	—	—	✓
	TO04	Output	PIOR00 = 0	0	0	—	×	—	—	—	—	✓
P43	P43	Input	—	1	×	—	—	—	—	—	—	✓
		Output	—	0	0/1	—	—					

Table 4 - 5 Examples of Register and Output Latch Settings for Alternate Functions (9/17)

Pin Name	Function Used		PIORx	POMxx	PMxx	Pxx	Alternate Function Output		20-pin	32-pin	44-pin	48-pin	64-pin
	Function Name	I/O					SAU and UARTA (excluding clock output from the UARTA)	Other than SAU and UARTA (including clock output from the UARTA)					
P50	P50	Input	—	×	1	×	×	—	—	✓	✓	✓	✓
		Output	—	0	0	0/1	SDA11 = 1	—					
		N-ch open-drain output	—	1	0	0/1							
	INTP1	Input	—	×	1	×	×	—	—	✓	✓	✓	✓
	SI11	Input	—	×	1	×	×	—	—	✓	✓	✓	✓
	SDA11	I/O	—	1	0	1	×	—	—	✓	✓	✓	✓
	(RxD2)	Input	PIOR15 = 1 PIOR14 = 0	×	1	×	×	—	—	✓	✓	✓	✓
P51	P51	Input	—	×	1	×	×	—	—	✓	✓	✓	✓
		Output	—	0	0	0/1	SO11 = 1 (TxD2) = 1	—					
		N-ch open-drain output	—	1	0	0/1							
	INTP2	Input	—	×	1	×	×	—	—	✓	✓	✓	✓
	SO11	Output	—	0/1	0	1	(TxD2) = 1	—	—	✓	✓	✓	✓
	(TxD2)	Output	PIOR15 = 1 PIOR14 = 0	0/1	0	1	SO11 = 1	—	—	✓	✓	✓	✓
P52	P52	Input	—	—	1	×	—	—	—	—	—	—	✓
		Output	—	—	0	0/1	—	—					
	(INTP10)	Input	PIOR01 = 1	—	1	×	—	—	—	—	—	—	✓
P53	P53	Input	—	—	1	×	—	—	—	—	—	—	✓
		Output	—	—	0	0/1	—	—					
	(INTP11)	Input	PIOR01 = 1	—	1	×	—	—	—	—	—	—	✓
P54	P54	Input	—	—	1	×	—	—	—	—	—	—	✓
		Output	—	—	0	0/1	—	—					
P55	P55	Input	—	×	1	×	×	×	—	—	—	—	✓
		Output	—	0	0	0/1	(SCK00) = 1	×					
		N-ch open-drain output	—	1	0	0/1							
	(SCK00)	Input	PIOR01 = 1	×	1	×	×	×	—	—	—	—	✓
		Output	PIOR01 = 1	0/1	0	1	×	×					

Table 4 - 5 Examples of Register and Output Latch Settings for Alternate Functions (10/17)

Pin Name	Function Used		PIORx	PMxx	Pxx	Alternate Function Output		20-pin	32-pin	44-pin	48-pin	64-pin
	Function Name	I/O				SAU and UARTA (excluding clock output from the UARTA)	Other than SAU and UARTA (including clock output from the UARTA)					
P60	P60	Input	—	1	×	—	×	—	✓	✓	✓	✓
		N-ch open-drain output (withstand voltage of 6 V)	—	0	0/1	—	SCLA0 = 0					
	SCLA0	I/O	PIOR02 = 0	0	0	—	×	—	✓	✓	✓	✓
P61	P61	Input	—	1	×	—	×	—	✓	✓	✓	✓
		N-ch open-drain output (withstand voltage of 6 V)	—	0	0/1	—	SDAA0 = 0					
	SDAA0	I/O	PIOR02 = 0	0	0	—	×	—	✓	✓	✓	✓

Table 4 - 5 Examples of Register and Output Latch Settings for Alternate Functions (11/17)

Pin Name	Function Used		PIORx	POMxx	PMxx	Pxx	Alternate Function Output		20-pin	32-pin	44-pin	48-pin	64-pin
	Function Name	I/O					SAU and UARTA (excluding clock output from the UARTA)	Other than SAU and UARTA (including clock output from the UARTA)					
P70	P70	Input	—	—	1	×	×	—	—	✓	✓	✓	✓
		Output	—	—	0	0/1	SCK21/SCL21 = 1 Note 1	—					
	KR0	Input	—	—	1	×	×	—	—	—	✓	✓	✓
	SCK21	Input	—	—	1	×	×	—	—	—	✓	✓	✓
		Output	—	—	0	1	×	—					
	SCL21	Output	—	—	0	1	×	—	—	—	✓	✓	✓
P71	P71	Input	—	×	1	×	×	—	—	—	✓	✓	✓
		Output	—	0	0	0/1	SDA21 = 1	—					
		N-ch open-drain output	—	1	0	0/1							
	KR1	Input	—	×	1	×	×	—	—	—	✓	✓	✓
	SI21	Input	—	×	1	×	×	—	—	—	✓	✓	✓
	SDA21	I/O	—	1	0	1	×	—	—	—	✓	✓	✓
	RxDA0	Input	PIOR17 = 0 PIOR16 = 0	×	1	×	×	—	—	—	✓	✓	✓
P72	P72	Input	—	×	1	×	×	—	—	—	✓	✓	✓
		Output	—	0	0	0/1	SO21 = 1 TxDA0 = 1	—					
		N-ch open-drain output	—	1	0	0/1							
	KR2	Input	—	×	1	×	×	—	—	—	✓	✓	✓
	SO21	Output	—	0/1	0	1	TxDA0 = 1	—	—	—	✓	✓	✓
	TxDA0	Output	PIOR17 = 0 PIOR16 = 0	0/1	0	1	SO21 = 1	—	—	—	✓	✓	✓
P73	P73	Input	—	—	1	×	×	—	—	—	✓	✓	✓
		Output	—	—	0	0/1	SO01 = 1 Note 2	—					
	KR3	Input	—	—	1	×	×	—	—	—	✓	✓	✓
	SO01	Output	—	—	0	1	×	—	—	—	—	✓	✓
P74	P74	Input	—	×	1	×	×	—	—	—	—	✓	✓
		Output	—	0	0	0/1	SDA01 = 1	—					
		N-ch open-drain output	—	1	0	0/1							
	KR4	Input	—	×	1	×	×	—	—	—	—	✓	✓
	INTP8	Input	PIOR11 = 0	×	1	×	×	—	—	—	—	✓	✓
	SI01	Input	—	×	1	×	×	—	—	—	—	✓	✓
	SDA01	I/O	—	1	0	1	×	—	—	—	—	✓	✓

Note 1. This setting is only applicable in the 44- to 64-pin products.

Note 2. This setting is only applicable in the 48- and 64-pin products.

Table 4 - 5 Examples of Register and Output Latch Settings for Alternate Functions (12/17)

Pin Name	Function Used		PIORx	POMxx	PMxx	Pxx	Alternate Function Output		20-pin	32-pin	44-pin	48-pin	64-pin
	Function Name	I/O					SAU and UARTA (excluding clock output from the UARTA)	Other than SAU and UARTA (including clock output from the UARTA)					
P75	P75	Input	—	—	1	×	SCK01/SCL01 = 1	—	—	—	—	✓	✓
		Output	—	—	0	0/1	SCK01/SCL01 = 1	—					
	KR5	Input	—	—	1	×	×	—	—	—	—	✓	✓
	INTP9	Input	PIOR12 = 0	—	1	×	×	—	—	—	—	✓	✓
	SCK01	Input	—	—	1	×	×	—	—	—	—	✓	✓
		Output	—	—	0	1	×	—					
	SCL01	Output	—	—	0	1	×	—	—	—	—	✓	✓
P76	P76	Input	—	—	1	×	—	—	—	—	—	—	✓
		Output	—	—	0	0/1	—	—					
	KR6	Input	—	—	1	×	—	—	—	—	—	—	✓
	INTP10	Input	PIOR01 = 0	—	1	×	—	—	—	—	—	—	✓
	(RxD2)	Input	PIOR15 = 1 PIOR14 = 1	—	1	×	—	—	—	—	—	—	✓
P77	P77	Input	—	—	1	×	×	—	—	—	—	—	✓
		Output	—	—	0	0/1	(TxD2) = 1	—					
	KR7	Input	—	—	1	×	×	—	—	—	—	—	✓
	INTP11	Input	PIOR01 = 0	—	1	×	×	—	—	—	—	—	✓
	(TxD2)	Output	PIOR15 = 1 PIOR14 = 1	—	0	1	×	—	—	—	—	—	✓

Table 4 - 5 Examples of Register and Output Latch Settings for Alternate Functions (13/17)

Pin Name	Function Used		PMCAxx	PMxx	Pxx	Alternate Function Output		20-pin	32-pin	44-pin	48-pin	64-pin
	Function Name	I/O				SAU and UARTA (excluding clock output from the UARTA)	Other than SAU and UARTA (including clock output from the UARTA)					
P120	P120	Input	0	1	×	—	—	—	✓	✓	✓	✓
		Output	0	0	0/1	—	—					
	ANI19	Analog input	1	1	×	—	—	—	✓	✓	✓	✓

Table 4 - 5 Examples of Register and Output Latch Settings for Alternate Functions (14/17)

Pin Name	Function Used		CMC	PIORx	PMxx	Pxx	Alternate Function Output		20-pin	32-pin	44-pin	48-pin	64-pin
	Function Name	I/O	EXCLK, OSCSEL				SAU and UARTA (excluding clock output from the UARTA)	Other than SAU and UARTA (including clock output from the UARTA)					
P121	P121	Input	00B/10B/11B	—	1	×	×	—	✓	✓	✓	✓	✓
		Output		—	0	0/1	TxD2 = 1 Note 1 (TxD2) = 1 Note 2	—					
	X1	—	01B	—	1	×	×	—	✓	✓	✓	✓	✓
	(INTP8)	Input	00B/10B/11B	PIOR11 = 1	1	×	×	—	—	—	—	✓	✓
	INTP8	Input	00B/10B/11B	—	1	×	×	—	✓	✓	✓	—	—
	(TxD2)	Output	00B/10B/11B	PIOR15 = 0 PIOR14 = 1	0	1	×	—	—	✓	✓	✓	✓
	TxD2	Output	00B/10B/11B	—	0	1	×	—	✓	—	—	—	—
P122	P122	Input	00B/10B	—	1	×	—	—	✓	✓	✓	✓	✓
		Output		—	0	0/1	—	—					
	X2	—	01B	—	1	×	—	—	✓	✓	✓	✓	✓
	EXCLK	Input	11B	—	1	×	—	—	✓	✓	✓	✓	✓
	(INTP6)	Input	00B/10B	PIOR10 = 1	1	×	—	—	—	—	—	✓	✓
	INTP6	Input	00B/10B	—	1	×	—	—	✓	✓	✓	—	—
	(RxD2)	Input	00B/10B	PIOR15 = 0 PIOR14 = 1	1	×	—	—	✓	✓	✓	✓	✓
	RxD2	Input	00B/10B	—	1	×	—	—	✓	—	—	—	—

Note 1. This setting is only applicable in the 20-pin products.

Note 2. This setting is only applicable in the 32- to 64-pin products.

Table 4 - 5 Examples of Register and Output Latch Settings for Alternate Functions (15/17)

Pin Name	Function Used		PIORx	PMxx	Pxx	Alternate Function Output		20-pin	32-pin	44-pin	48-pin	64-pin
	Function Name	I/O				SAU and UARTA (excluding clock output from the UARTA)	Other than SAU and UARTA (including clock output from the UARTA)					
P123	P123	Input	—	1	×	×	—	—	—	✓	✓	✓
		Output	—	0	0/1	(TxDA0) = 1	—					
	(TxDA0)	Output	PIOR17 = 0 PIOR16 = 1	1	1	×	—	—	—	✓	✓	✓
P124	P124	Input	—	1	×	—	—	—	—	✓	✓	✓
		Output	—	0	0/1	—	—					
	(RxDA0)	Input	PIOR17 = 0 PIOR16 = 1	1	×	—	—	—	—	✓	✓	✓
P125	P125	Input	—	1	×	—	—	✓	✓	✓	✓	✓
		Output	—	0	0/1	—	—					
	$\overline{\text{RESET}}$	Input	—	1	×	—	—	✓	✓	✓	✓	✓
	INTP9	Input	—	1	×	—	—	✓	✓	✓	—	—
	(INTP9)	Input	PIOR12 = 1	1	×	—	—	—	—	—	✓	✓

Table 4 - 5 Examples of Register and Output Latch Settings for Alternate Functions (16/17)

Pin Name	Function Used		PMxx	Pxx	20-pin	32-pin	44-pin	48-pin	64-pin
	Function Name	I/O							
P130	P130	Output	—	0/1	—	—	—	✓	✓
P137	P137	Input	—	×	✓	✓	✓	✓	✓
	INTP0	Input	—	×	✓	✓	✓	✓	✓

Table 4 - 5 Examples of Register and Output Latch Settings for Alternate Functions (17/17)

Pin Name	Function Used		PIORx	PMCAxx	PMxx	Pxx	Alternate Function Output		20-pin	32-pin	44-pin	48-pin	64-pin
	Function Name	I/O					SAU and UARTA (excluding clock output from the UARTA)	Other than SAU and UARTA (including clock output from the UARTA)					
P140	P140	Input	—	—	1	×	—	—	—	—	—	✓	✓
		Output	—	—	0	0/1	—	—					
	INTP6	Input	PIOR10 = 0	—	1	×	—	—	—	—	—	✓	✓
P141	P141	Input	—	—	1	×	—	—	—	—	—	—	✓
		Output	—	—	0	0/1	—	—					
	INTP7	Input	—	—	1	×	—	—	—	—	—	—	✓
P146	P146	Input	—	—	1	×	—	—	—	—	✓	✓	✓
		Output	—	—	0	0/1	—	—					
P147	P147	Input	—	0	1	×	—	—	✓	✓	✓	✓	✓
		Output	—	0	0	0/1	—	—					
	ANI18	Analog input	—	1	1	×	—	—	✓	✓	✓	✓	✓

4.6 Cautions When Using Port Function

4.6.1 Cautions on 1-bit memory manipulation instruction for port register (Pxx)

When a 1-bit memory manipulation instruction is executed on a port that provides both input and output functions, the output latch value of an input port that is not subject to manipulation may be written in addition to the targeted bit. Therefore, it is recommended to rewrite the output latch before switching a port from input mode to output mode.

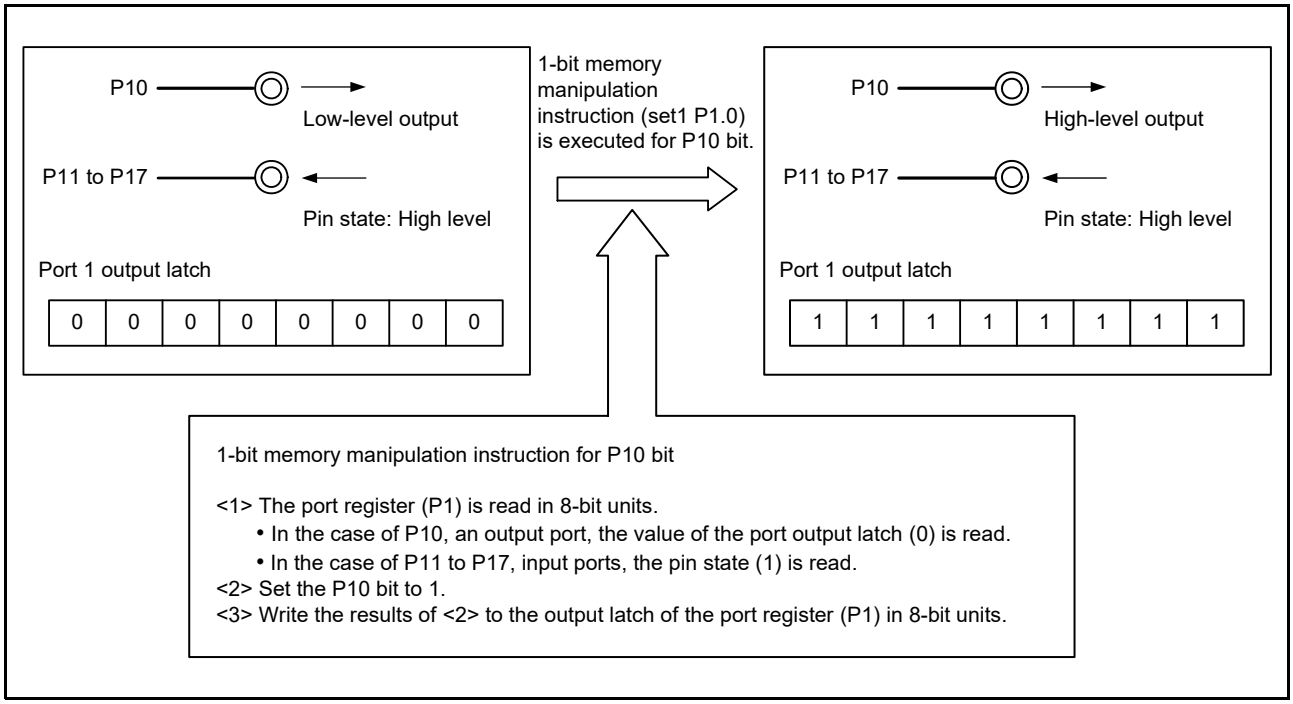
(Example) When P10 is an output port, P11 to P17 are input ports (all pins are at the high level), and the output latch value of port 1 is 00H, if the output of output port P10 is changed from low level to high level via a 1-bit memory manipulation instruction, the output latch value of port 1 becomes FFH.

Explanation: The targets of writing to and reading from the Pxx register of a port whose PMmn bit is 1 are the output latch and pin state, respectively. A 1-bit memory manipulation instruction is executed in the following order.

<1> The Pxx register is read in 8-bit units.
<2> The targeted one bit is manipulated.
<3> The Pxx register is written in 8-bit units.

In step <1>, the output latch value (0) of P10, which is an output port, is read, while the states of P11 to P17, which are input ports, are read. If the states of P11 to P17 are high level at this time, the read value is FEH. The value is changed to FFH by the manipulation in <2>. FFH is written to the output latch by the manipulation in <3>.

Figure 4 - 12 One-bit Memory Manipulation Instruction (P10)



4.6.2 Notes on specifying the pin settings

For an output pin to which multiple functions are assigned, the output of the unused alternate functions must be set to the initial state of the pin so as to prevent conflicting outputs. This also applies to the functions assigned by using the corresponding peripheral I/O redirection registers (PIORx). For details about the alternate function output, see **4.5 Register Settings When Using Alternate Function**. No specific setting is required for input pins because the output of their alternate functions is disabled (the buffer output is Hi-Z).

Disabling the unused functions, including blocks that are only used for input or do not have I/O, is recommended for lower power consumption.

Section 5 Operation State Control

The operating voltage, operating timing, and operating current of the internal circuit are optimized using operating modes of the flash memory. Select an appropriate operating mode of the flash memory in accord with the operating voltage range and clock frequencies of the MCU. The operating mode of the flash memory set by the option byte is selected for operation immediately after a reset is released. The mode can be changed by setting of the relevant register.

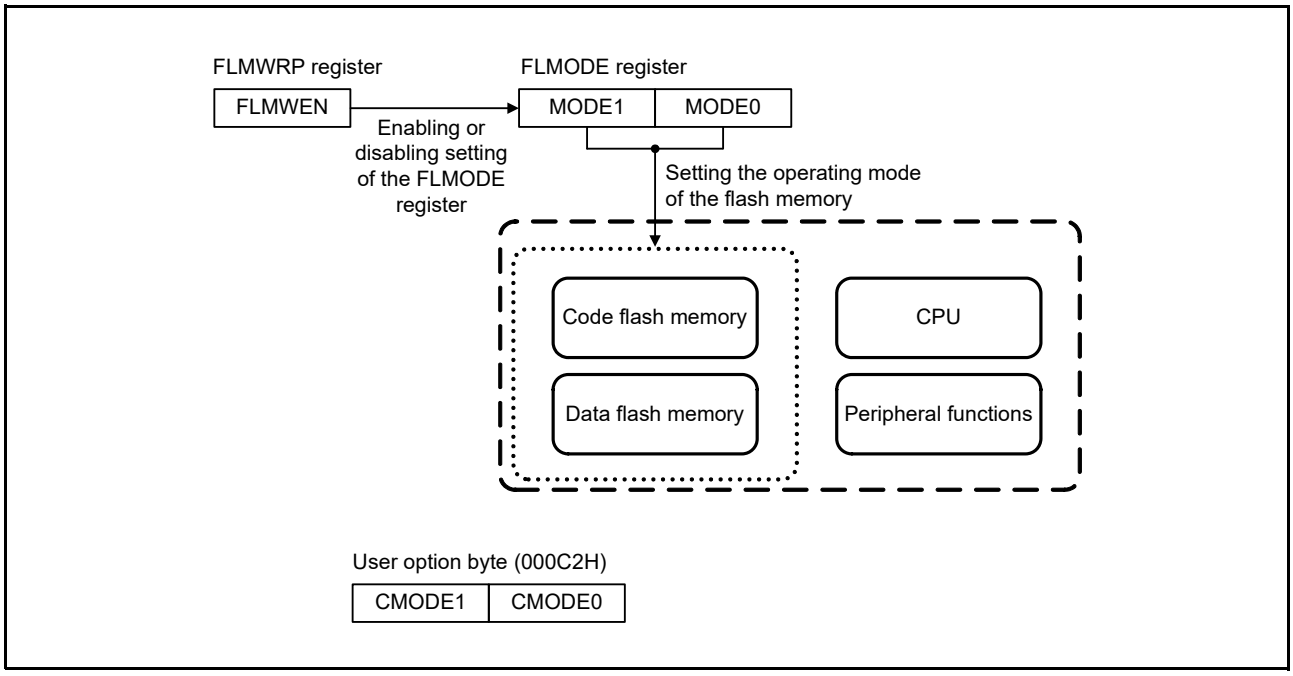
5.1 Configuration of Operation State Control

Operation state control is supported by the following hardware blocks.

Table 5 - 1 Configuration of Operation State Control

Item	Configuration
Option byte	• Address of the user option byte: 000C2H
Control registers	• Flash operating mode select register (FLMODE) • Flash operating mode protect register (FLMWRP)

Figure 5 - 1 Block Diagram of Operation State Control



There are the following four operating modes of the flash memory.

- HS (high-speed main) mode
- LS (low-speed main) mode
- LP (low-power main) mode
- SUB mode

The MCU can be operated efficiently by setting these operating modes of the flash memory according to MCU operating conditions. **Table 5 - 2** lists the features of each operating mode of the flash memory.

Table 5 - 2 Features of Each Operating Mode of the Flash Memory

Operating Mode of the Flash Memory	Recommended Operating Range		Description
HS (high-speed main) mode	1.6 to 1.8 V	1 to 4 MHz (Rewriting of the flash memory is not possible.)	High-speed CPU operation (at 32 MHz (max.)) is possible in this mode. Suitable when CPU processing capacity is required.
	1.8 to 5.5 V	1 to 32 MHz	
LS (low-speed main) mode	1.6 to 1.8 V	1 to 4 MHz (Rewriting of the flash memory is not possible.)	The operating current and CPU operation processing (at 24 MHz (max.)) are well-balanced in this mode.
	1.8 to 5.5 V	1 to 24 MHz	
LP (low-power main) mode**Note 1**	1.6 to 5.5 V	1 to 2 MHz (Rewriting of the flash memory is not possible.)	The CPU operates at 1 to 2 MHz in this mode. Low operating current is realized at 1 to 2 MHz.
SUB mode	1.6 to 5.5 V	32.768 kHz (Rewriting of the flash memory is not possible.)	CPU operation is driven by the subsystem clock.**Note 2** This mode enables low-power operation.

Note 1. In LP mode, only the high-speed on-chip oscillator clock (f_{IH}) can be selected as the CPU/peripheral hardware clock (f_{CLK}).

Note 2. The subsystem clock can be derived from the low-speed on-chip oscillator clock (f_{IL}).

5.2 Registers for Controlling the Operation State Control

The following registers are used to control the operation state control.

- Flash operating mode select register (FLMODE)
- Flash operating mode protect register (FLMWRP)

5.2.1 Flash operating mode select register (FLMODE)

The FLMODE is an 8-bit register used to control operating modes of the flash memory. This register can be set by a 1-bit or 8-bit memory manipulation instruction. Note that the value of this register cannot be changed when the FLMWEN bit in the flash operating mode protect register (FLMWRP) is 0. The settings of the MODE1 and MODE0 bits are updated to those of the CMODE1 and CMODE0 bits in the user option byte at address 000C2H following a reset.

Figure 5 - 2 Format of Flash Operating Mode Select Register (FLMODE)

Address: F00AAH
After reset: 40H/80H/C0H**Note 1**
R/W: R/W

Symbol	<7>	<6>	5	4	3	2	1	0
FLMODE	MODE1	MODE0	0	0	0	0	0	0

MODE1	MODE0	Selection of operating mode of the flash memory
0	0	Setting prohibited
0	1	LP (low-power main) mode**Note 2** (Selectable when 1 MHz ≤ f_{CLK} ≤ 2 MHz in LS mode)
1	0	LS (low-speed main) mode (Selectable when 1 MHz ≤ f_{CLK} ≤ 24 MHz in HS mode or in LP mode)
1	1	HS (high-speed main) mode (Selectable when in LS mode)

Note 1. The initial value of the FLMODE register is set to the value of the MODE1 and MODE0 bits updated with the set value of the CMODE1 and CMODE0 bits in the user option byte (address: 000C2H).

Note 2. In LP mode, only the high-speed on-chip oscillator clock (f_{IH}) can be selected as the CPU/peripheral hardware clock (f_{CLK}).

Caution 1. The value of the FLMODE register can be changed when the FLMWEN bit in the flash operating mode protect register (FLMWRP) is 1. After the value of the FLMODE register is changed, set the FLMWEN bit to 0.

Caution 2. Operation is in SUB mode when the setting of the CSS bit in the system clock control register (CKC) is 1 (operation of the CPU and peripheral functions is driven by the subsystem clock) regardless of the settings of the MODE1 and MODE0 bits.

Caution 3. When changing the operating mode of the flash memory, make sure that operation is possible within the voltage range and operating frequency range in the changed operating mode of the flash memory before changing the mode.

(Cautions continue on the next page.)

Caution 4. When the operating mode of the flash memory is changed by the MODE1 and MODE0 bits, the CPU is placed in the wait state for the following time until the mode changes. Interrupt requests are held pending during this wait period.

Times for switching between the operating modes of the flash memory

Switching between the Operating Modes of the Flash Memory	Transition Time
LS (low-speed main) mode → HS (high-speed main) mode	225 cycles[Note]
LP (low-power main) mode → LS (low-speed main) mode	10 cycles[Note]
LS (low-speed main) mode → LP (low-power main) mode	10 cycles[Note]
HS (high-speed main) mode → LS (low-speed main) mode	30 cycles[Note]

Note The cycle of the CPU/peripheral hardware clock (f_{CLK})

Caution 5. When rewriting the FLMODE register, insert at least one clock cycle of the CPU/peripheral hardware clock (f_{CLK}) after having rewritten the FLMODE register. Do not write to the FLMODE register successively.

Caution 6. Do not change the FLMODE register while rewriting the flash memory.

Caution 7. Before changing the operating mode of the flash memory, set the DFLEN bit in the data flash control register (DFLCTL) to 1 to enable access to the data flash memory.

Caution 8. Before rewriting the contents of the code flash memory or data flash memory area by self-programming, be sure to place the MCU in HS (high-speed main) or LS (low-speed main) mode.

5.2.2 Flash operating mode protect register (FLMWRP)

The FLMWRP is an 8-bit register used to control access to the flash operating mode select register. This register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 5 - 3 Format of Flash Operating Mode Protect Register (FLMWRP)

Address: F00ABH
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	<0>
FLMWRP	0	0	0	0	0	0	0	FLMWEN

FLMWEN	Control of flash operating mode select register (FLMODE)
0	Rewriting the FLMODE register is disabled.
1	Rewriting the FLMODE register is enabled.

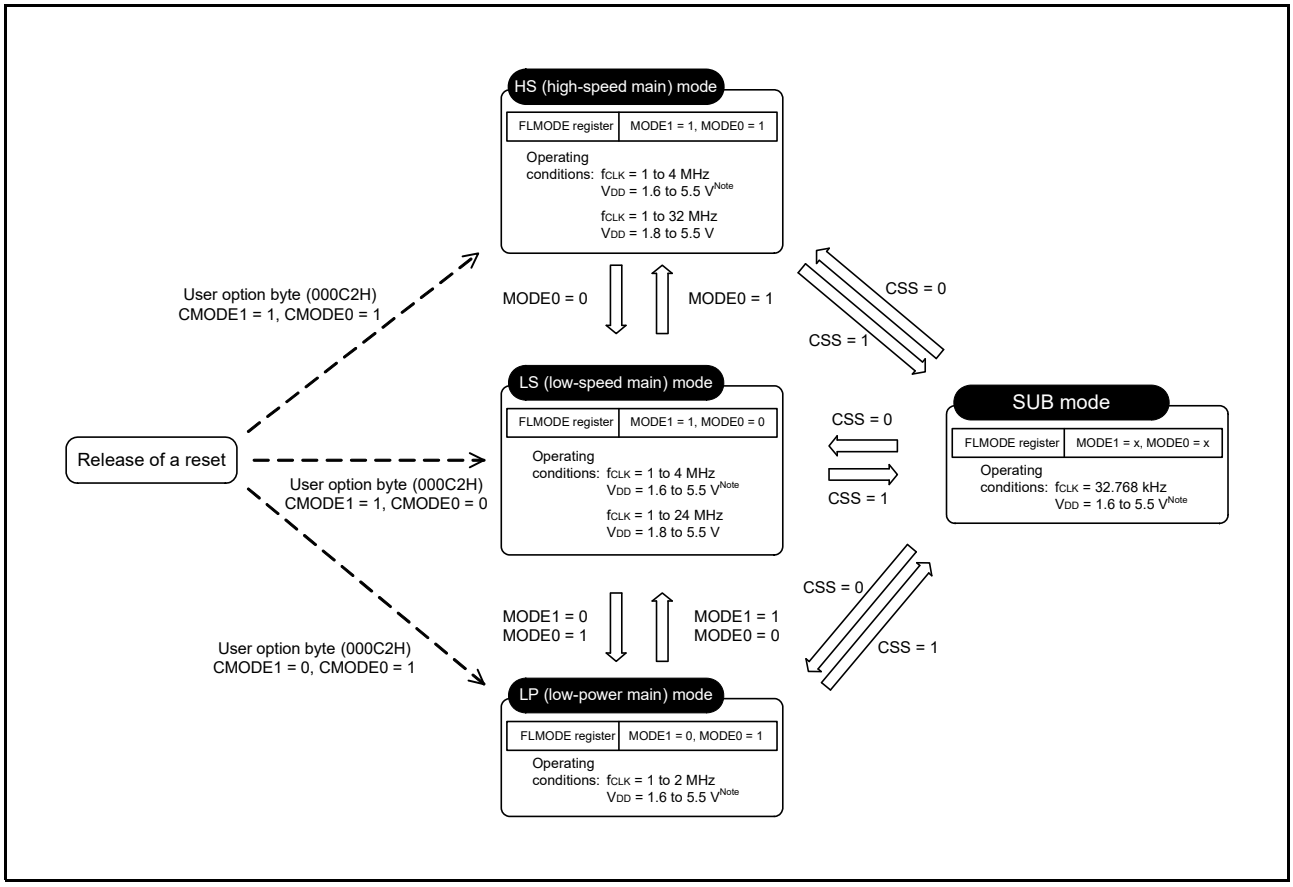
5.3 Initial Setting of Operating Modes of the Flash Memory

The user option byte (000C2H) is used to set the initial state of the operating mode of the flash memory and the high-speed on-chip oscillator after a reset is released. Set an appropriate operating mode of the flash memory according to the V_{DD} voltage and the high-speed on-chip oscillator frequency at a reset release. When a reset is released, the values of the CMODE1 and CMODE0 bits are reflected in the MODE1 and MODE0 bits in the flash operating mode select register (FLMODE) and the values of the FRQSEL3 to FRQSEL0 bits are reflected in the high-speed on-chip oscillator frequency select register (HOCODIV). For details on the user option byte (000C2H), see **Section 23 Option Bytes**.

5.4 Transitions between Operating Modes of the Flash Memory

The settings of the CMODE1 and CMODE0 bits in the user option byte (000C2H) determine the initial operating mode of the flash memory immediately after a reset is released. The initial state can be selected from among the HS (high-speed main) mode, LS (low-speed main) mode, or LP (low-power main) mode. The values of the CMODE1 and CMODE0 bits are reflected in the MODE1 and MODE0 bits in the flash operating mode select register (FLMODE). After that, the operating mode of the flash memory can be changed by changing the value of the FLMODE register during CPU operation. Setting the CSS bit in the CKC register to 1 enables automatically placing the MCU in SUB mode.

Figure 5 - 4 State Transitions between Operating Modes of the Flash Memory



Note Rewriting of the flash memory is not possible.

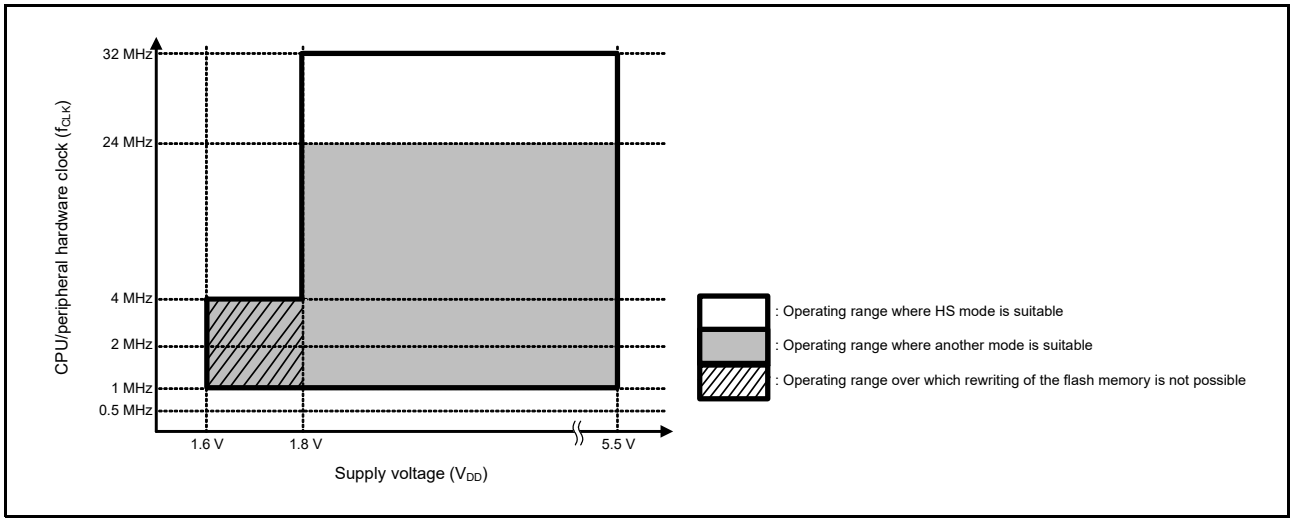
Caution **When a reset is applied while the MCU operates, operation always starts in the operating mode of the flash memory set by the option byte after a reset release. Therefore, make sure that operation does not start outside the operating voltage range when a reset is released by setting the LVD detection voltage to at least the operating voltage range of the operating mode of the flash memory set in the option byte.**

5.5 Details of Operating Modes of the Flash Memory

5.5.1 Details of HS (high-speed main) mode

HS (high-speed main) mode is suitable for applications that require CPU high-speed processing. The CPU can run in HS mode immediately after release from the reset state. Transition to this mode from LS (low-speed main) mode is possible. The suitable operating range for HS mode is when the supply voltage satisfies 1.8 V ≤ V_{DD} ≤ 5.5 V and the operating frequency satisfies 24 MHz < f_{CLK} ≤ 32 MHz. When 24 MHz or a lower frequency is used for operation, another mode can be used as the suitable operating mode of the flash memory.

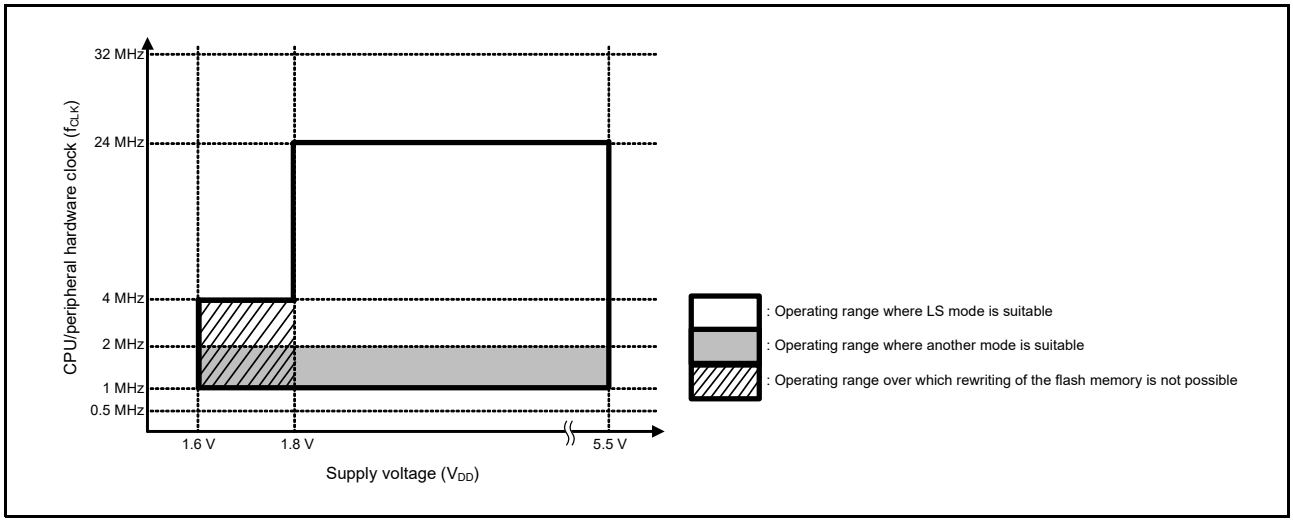
Figure 5 - 5 Operating Range in HS Mode



5.5.2 Details of LS (low-speed main) mode

LS (low-speed main) mode supports performance in terms of both CPU processing capacity and operating voltage, and is suitable for applications that require low-power consumption and are capable of running at 2 to 24 MHz. The CPU can run in LS mode immediately after release from the reset state. Transition to this mode from HS (high-speed main) mode or LP (low-power main) mode is possible. For a transition from HS mode to LS mode, make sure that the operating frequency satisfies 1 MHz ≤ f_{CLK} ≤ 24 MHz. The suitable operating range for LS mode is when the supply voltage satisfies 1.8 V ≤ V_{DD} ≤ 5.5 V and the operating frequency satisfies 2 MHz < f_{CLK} ≤ 24 MHz, or when the supply voltage satisfies 1.6 V ≤ V_{DD} < 1.8 V and the operating frequency satisfies 2 MHz < f_{CLK} ≤ 4 MHz.

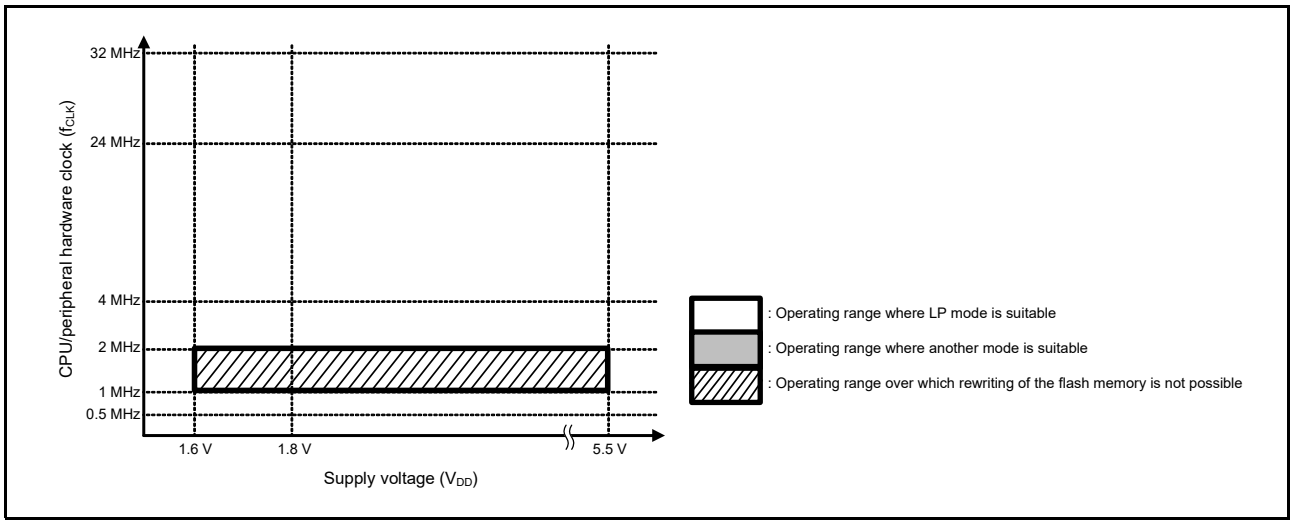
Figure 5 - 6 Operating Range in LS Mode



5.5.3 Details of LP (low-power main) mode

In LP (low-power main) mode, the CPU operates with a low voltage and at a frequency from 1 to 2 MHz. The CPU can run in LP mode immediately after release from the reset state. Transition to this mode from LS (low-speed main) mode is possible. For a transition from LS mode to LP mode, make sure that the operating frequency satisfies 1 MHz ≤ f_{CLK} ≤ 2 MHz. The suitable operating range for LP mode is when the supply voltage satisfies 1.6 V ≤ V_{DD} ≤ 5.5 V and the operating frequency satisfies 1 MHz ≤ f_{CLK} ≤ 2 MHz. In LP mode, only the high-speed on-chip oscillator clock (f_{IH}) can be selected as the CPU/peripheral hardware clock (f_{CLK}). When rewriting of flash memory is required, place the CPU in LS mode.

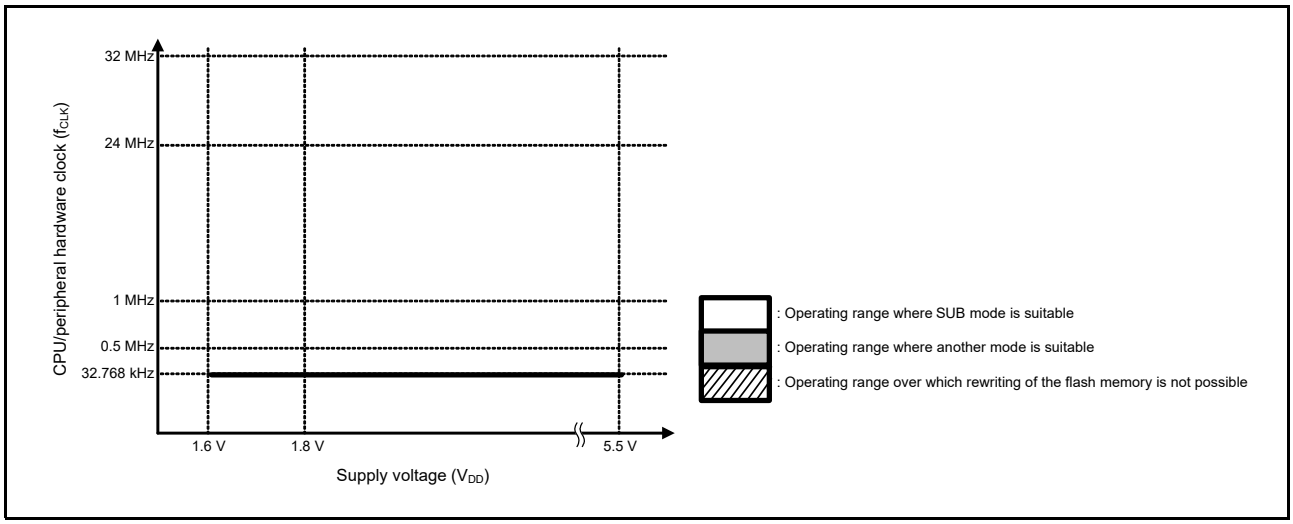
Figure 5 - 7 Operating Range in LP Mode



5.5.4 Details on SUB mode

In SUB mode, the CPU operates at a frequency of 32.768 kHz. Transition to SUB mode from HS (high-speed main) mode, LS (low-speed main) mode, or LP (low-power main) mode is possible. Setting the CSS bit in the system clock control register (CKC) to 1 places the CPU in SUB mode. Rewriting of the flash memory is not possible in SUB mode. When rewriting of flash memory is required, place the CPU in HS mode or LS mode.

Figure 5 - 8 Operating Range in SUB Mode



Section 6 Clock Generator

6.1 Functions of Clock Generator

The clock generator generates clocks to be supplied to the CPU and peripheral hardware. The following kinds of system clocks and clock oscillators are selectable.

1. Main system clock
   <1> X1 oscillator
   This circuit oscillates a clock of f_X = 1 to 20 MHz by connecting a resonator to the X1 and X2 pins. Oscillation can be stopped by executing a STOP instruction or setting of the MSTOP bit of the clock operation status control register (CSC).
   <2> High-speed on-chip oscillator
   The frequency of oscillation for f_{IH} can be selected from among 32, 24, 16, 12, 8, 6, 4, 3, 2, and 1 MHz (typ.) by using a user option byte (000C2H). After release from the reset state, the CPU always starts operating with this high-speed on-chip oscillator clock. Oscillation can be stopped by executing a STOP instruction or setting of the HIOSTOP bit of the CSC register. The frequency specified by using the user option byte can be changed by using the high-speed on-chip oscillator frequency select register (HOCODIV). For details about the frequency, see **Figure 6 - 11 Format of High-speed On-chip Oscillator Frequency Select Register (HOCODIV)**. The frequencies that can be specified for the high-speed on-chip oscillator by using the user option byte and the high-speed on-chip oscillator frequency select register (HOCODIV) are shown below.

Power Supply Voltage	Oscillation Frequency (MHz)									
	1	2	3	4	6	8	12	16	24	32
1.8 V ≤ V_{DD} ≤ 5.5 V	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
1.6 V ≤ V_{DD} ≤ 5.5 V	✓	✓	✓	✓	—	—	—	—	—	—

An external main system clock (f_{EX} = 1 to 20 MHz) can also be supplied from the EXCLK/X2/P122 pin. External main system clock input can be disabled by executing a STOP instruction or setting of the MSTOP bit.

As the main system clock, a high-speed system clock (X1 clock or external main system clock) or a high-speed on-chip oscillator clock can be selected by setting of the MCM0 bit of the system clock control register (CKC).

Note that the usable frequency range of the CPU/peripheral hardware clock depends on the settings for the operating mode of the flash memory and V_{DD} power supply voltage. When the main system clock is to be used as the CPU/peripheral hardware clock, select the frequency of the main system clock to suit the operating mode of the flash memory specified in the CMODE1 and CMODE0 bits of a user option byte (000C2H) (see **Section 23 Option Bytes**) or the flash operating mode select register (FLMODE).

2. Subsystem clock

<1> Low-speed on-chip oscillator

This circuit oscillates a clock of f_{IL} = 32.768 kHz (typ.).

The low-speed on-chip oscillator operates when the following condition is met.

- The value of one or more of the following bits is 1: WDTON bit of a user option byte (000C0H), WUTMMCK0 bit of the subsystem clock supply mode control register (OSMC), and SELLOSC bit of the subsystem clock select register (CKSEL)
  However, if a HALT or STOP instruction is executed when WDTON = 1, WUTMMCK0 = 0, SELLOSC = 0 and the WDSTBYON bit of the user option byte (000C0H) is 0, this oscillator stops oscillating.

Remark f_X: X1 clock oscillation frequency
f_{IH}: High-speed on-chip oscillator clock frequency (32 MHz max.)
f_{EX}: External main system clock frequency
f_{IL}: Low-speed on-chip oscillator clock frequency

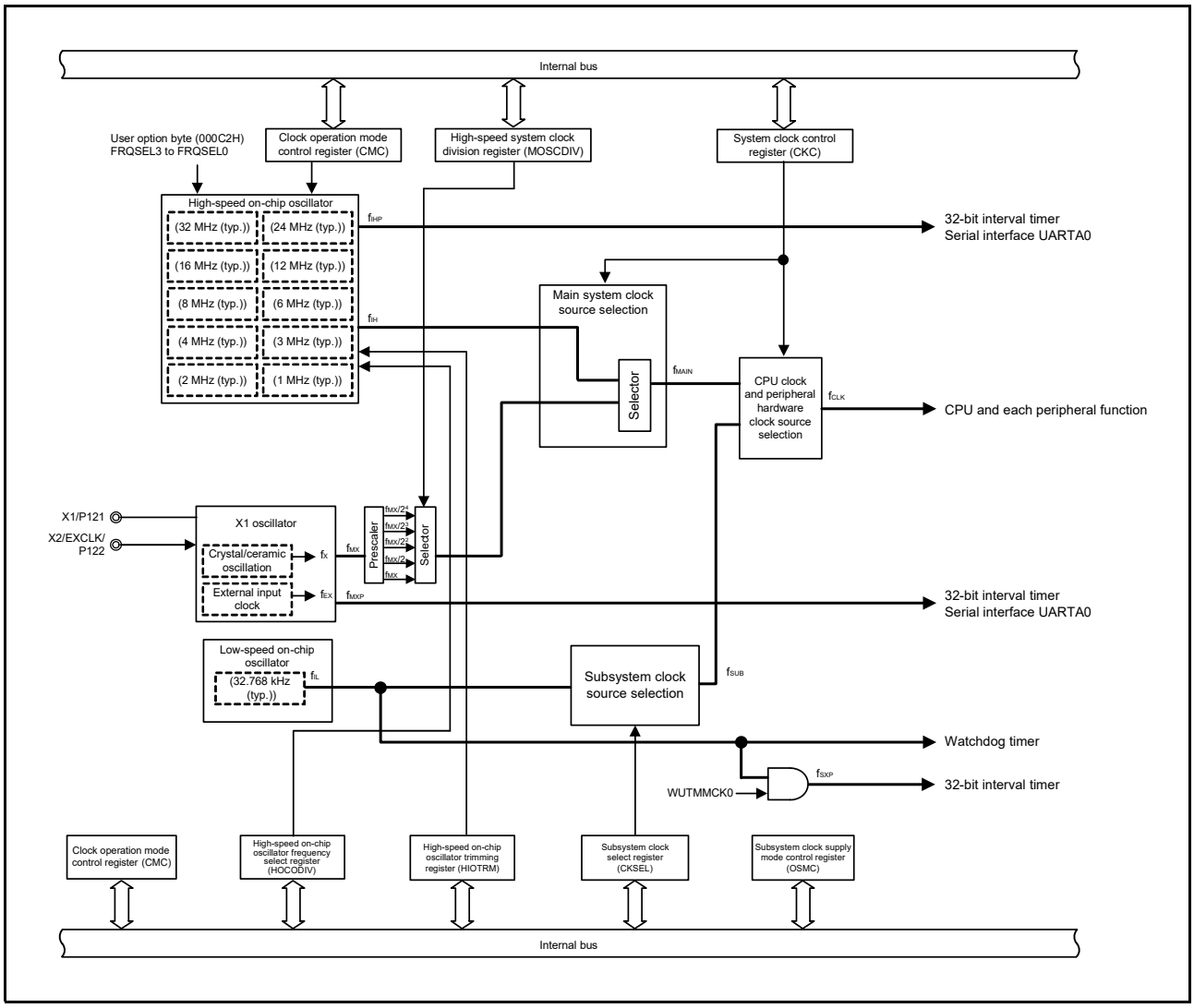
6.2 Configuration of Clock Generator

The clock generator includes the following hardware.

Table 6 - 1 Configuration of Clock Generator

Item	Configuration
Control registers	• Clock operation mode control register (CMC) • System clock control register (CKC) • Clock operation status control register (CSC) • Oscillation stabilization time counter status register (OSTC) • Oscillation stabilization time select register (OSTS) • Peripheral enable registers 0, 1 (PER0, PER1) • Subsystem clock supply mode control register (OSMC) • Subsystem clock select register (CKSEL) • High-speed on-chip oscillator frequency select register (HOCODIV) • High-speed system clock division register (MOSCDIV) • High-speed on-chip oscillator trimming register (HIOTRM)
Oscillators	• X1 oscillator • High-speed on-chip oscillator • Low-speed on-chip oscillator

Figure 6 - 1 Block Diagram of Clock Generator



Remark fX: X1 clock oscillation frequency
fEX: External main system clock frequency
fIH: High-speed on-chip oscillator clock frequency (32 MHz max.)
fIHP: High-speed on-chip oscillator peripheral clock frequency
(32 MHz when FRQSEL3 = 1, 24 MHz when FRQSEL3 = 0)
fMX: High-speed system clock frequency
fMXP: High-speed peripheral clock frequency
fMAIN: Main system clock frequency
fSXP: Low-speed peripheral clock frequency
fSUB: Subsystem clock frequency
fCLK: CPU/peripheral hardware clock frequency
fIL: Low-speed on-chip oscillator clock frequency

6.3 Registers for Controlling the Clock Generator

The following registers are used to control the clock generator.

- Clock operation mode control register (CMC)
- System clock control register (CKC)
- Clock operation status control register (CSC)
- Oscillation stabilization time counter status register (OSTC)
- Oscillation stabilization time select register (OSTS)
- Peripheral enable registers 0, 1 (PER0, PER1)
- Subsystem clock supply mode control register (OSMC)
- Subsystem clock select register (CKSEL)
- High-speed on-chip oscillator frequency select register (HOCODIV)
- High-speed system clock division register (MOSCDIV)
- High-speed on-chip oscillator trimming register (HIOTRM)

Caution The inclusion of registers and bit functions depends on the product. Be sure to set bits to which no functions are allocated to their initial values.

6.3.1 Clock operation mode control register (CMC)

The CMC register is used to set the operation mode of the X1/P121 and X2/EXCLK/P122 pins, and to select the gain of the oscillator. The CMC register can be written only once by an 8-bit memory manipulation instruction after release from the reset state. This register can be read by an 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 6 - 2 Format of Clock Operation Mode Control Register (CMC)

Address: FFFA0H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
CMC	EXCLK	OSCSEL	0	0	0	0	0	AMPH

EXCLK	OSCSEL	High-speed system clock pin operation mode	X1/P121 pin	X2/EXCLK/P122 pin
0	0	Port mode	Port	Port
0	1	X1 oscillation mode	Crystal/ceramic resonator connection	
1	0	Port mode	Port	Port
1	1	External clock input mode	Port	External clock input

AMPH	Control of the X1 clock oscillation frequency
0	1 MHz ≤ fx ≤ 10 MHz
1	10 MHz < fx ≤ 20 MHz

Caution 1. Be sure to leave bits 5 to 1 set to 0.

Caution 2. The CMC register can be written only once by an 8-bit memory manipulation instruction after release from the reset state. Even if you intend to use the CMC register with its initial value (00H), be sure to write 00H to the register after release from the reset state as a precaution against malfunctions (since returning the value to 00H after erroneously having written a value other than 00H to it is not possible).

Caution 3. After release from the reset state, set the CMC register before X1 oscillation is started by setting the clock operation status control register (CSC).

Caution 4. Be sure to set the AMPH bit to 1 if the X1 clock oscillation frequency exceeds 10 MHz.

Caution 5. Make the setting of the AMPH bit while f_{IH} is selected as f_{CLK} after release from the reset state (before f_{CLK} is switched to f_{MX}).

Remark fx: X1 clock oscillation frequency

6.3.2 System clock control register (CKC)

The CKC register is used to select a CPU/peripheral hardware clock and a main system clock. The CKC register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 6 - 3 Format of System Clock Control Register (CKC)

Address: FFFA4H
After reset: 00H
R/W: R/W**Note 1**

Symbol	<7>	<6>	<5>	<4>	3	2	1	0
CKC	CLS	CSS	MCS	MCM0	0	0	0	0

CLS	State of the CPU/peripheral hardware clock (f_{CLK})
0	Main system clock (f_{MAIN})
1	Subsystem clock (f_{SUB})

CSS**Note 2**	Selection of the CPU/peripheral hardware clock (f_{CLK})
0	Main system clock (f_{MAIN})
1	Subsystem clock (f_{SUB})

MCS	State of the main system clock (f_{MAIN})
0	High-speed on-chip oscillator clock (f_{IH})
1	High-speed system clock (f_{MX})

MCM0 **Note 2**	Main system clock (f_{MAIN}) operation control
0	Selects the high-speed on-chip oscillator clock (f_{IH}) as the main system clock (f_{MAIN})
1	Selects the high-speed system clock (f_{MX}) as the main system clock (f_{MAIN})

Note 1. Bits 7 and 5 are only readable.

<R> **Note 2.** Changing the value of the MCM0 bit is prohibited while the CSS bit is set to 1. Also, do not set the CSS bit to 1 while the setting of the SELLOSC bit in the CKSEL register is 0.

Caution 1. Be sure to leave bits 3 to 0 set to 0.

Caution 2. The clock set by the CSS bit is supplied to the CPU and peripheral hardware. If the CPU clock is changed, therefore, the clock supplied to peripheral hardware (except the 32-bit interval timer and watchdog timer) is also changed at the same time. Consequently, stop each peripheral function when changing the CPU/peripheral hardware clock.

Caution 3. If the subsystem clock is used as the peripheral hardware clock, the operations of the A/D converter and serial interface IICA are not guaranteed. For the operating characteristics of the peripheral hardware, refer to the sections describing each item of peripheral hardware as well as Section 28 Electrical Characteristics.

Caution 4. When switching the CPU clock between the main system clock and subsystem clock, accessing the data flash memory is prohibited until the CLS bit changes after a clock is selected by the CSS bit.

Caution 5. When switching the CPU clock between the main system clock and subsystem clock, executing the HALT or STOP instruction is prohibited until the CLS bit changes after a clock is selected by the CSS bit.

Remark f_{IH}: High-speed on-chip oscillator clock frequency (32 MHz max.)
f_{MX}: High-speed system clock frequency
f_{MAIN}: Main system clock frequency
f_{SUB}: Subsystem clock frequency

6.3.3 Clock operation status control register (CSC)

The CSC register is used to control the operations of the high-speed system clock and high-speed on-chip oscillator clock. The CSC register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is C0H.

Figure 6 - 4 Format of Clock Operation Status Control Register (CSC)

Address: FFFA1H
After reset: C0H
R/W: R/W

Symbol	<7>	6	5	4	3	2	1	<0>
CSC	MSTOP	1	0	0	0	0	0	HIOSTOP

MSTOP	High-speed system clock operation control		
	X1 oscillation mode	External clock input mode	Port mode
0	The X1 oscillator runs.	An external clock signal on the EXCLK pin is effective.	I/O port
1	The X1 oscillator is stopped.	An external clock signal on the EXCLK pin is ineffective.	

HIOSTOP	High-speed on-chip oscillator clock operation control
0	The high-speed on-chip oscillator runs.
1	The high-speed on-chip oscillator is stopped.

Caution 1. After release from the reset state, set the clock operation mode control register (CMC) before setting the CSC register.

Caution 2. Set the oscillation stabilization time select register (OSTS) before setting the MSTOP bit to 0 after release from the reset state. Note that if the OSTS register is being used with its default settings, the OSTS register is not required to be set here.

Caution 3. When starting X1 oscillation by using the MSTOP bit, check the oscillation stabilization time of the X1 clock by using the oscillation stabilization time counter status register (OSTC).

Caution 4. Do not stop the clock selected for the CPU/peripheral hardware clock (f_{CLK}) with the CSC register.

Caution 5. When stopping the clock, confirm the conditions before clock oscillation is stopped. For details on how to stop the clock, see Table 6 - 7 Conditions Before the Clock Oscillation Is Stopped and Flag Settings.

6.3.4 Oscillation stabilization time counter status register (OSTC)

The OSTC register indicates the counter value by the X1 clock oscillation stabilization time counter. The X1 clock oscillation stabilization time can be checked in the following cases:

- If the X1 clock starts oscillation while the high-speed on-chip oscillator clock or subsystem clock is in use as the CPU clock.
- If entry to and then release from the STOP mode proceed while the high-speed on-chip oscillator clock is in use as the CPU clock and the X1 clock is oscillating.

The OSTC register can be read by a 1-bit or 8-bit memory manipulation instruction. The value of this register is 00H following a reset, STOP instruction, or the MSTOP bit of the clock operation status control register (CSC) being set to 1.

Remark The oscillation stabilization time counter starts counting in the following cases.

- When oscillation of the X1 clock starts (EXCLK = 0 and OSCSEL = 1 in the CMC register → MSTOP = 0)
- When the STOP mode is released

Figure 6 - 5 Format of Oscillation Stabilization Time Counter Status Register (OSTC)

Address: FFFA2H
After reset: 00H
R/W: R

Symbol	7	6	5	4	3	2	1	0
OSTC	MOST8	MOST9	MOST10	MOST11	MOST13	MOST15	MOST17	MOST18

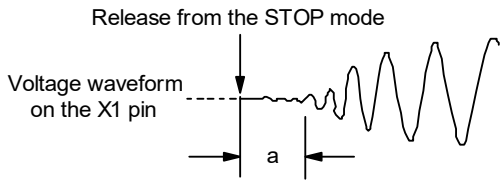
MOST8	MOST9	MOST10	MOST11	MOST13	MOST15	MOST17	MOST18	State of the oscillation stabilization time		
									fx = 10 MHz	fx = 20 MHz
0	0	0	0	0	0	0	0	Less than 2^8/fx	Less than 25.6 μs	Less than 12.8 μs
1	0	0	0	0	0	0	0	2^8/fx min.	25.6 μs min.	12.8 μs min.
1	1	0	0	0	0	0	0	2^9/fx min.	51.2 μs min.	25.6 μs min.
1	1	1	0	0	0	0	0	2^{10}/fx min.	102 μs min.	51.2 μs min.
1	1	1	1	0	0	0	0	2^{11}/fx min.	204 μs min.	102 μs min.
1	1	1	1	1	0	0	0	2^{13}/fx min.	819 μs min.	409 μs min.
1	1	1	1	1	1	0	0	2^{15}/fx min.	3.27 ms min.	1.63 ms min.
1	1	1	1	1	1	1	0	2^{17}/fx min.	13.1 ms min.	6.55 ms min.
1	1	1	1	1	1	1	1	2^{18}/fx min.	26.2 ms min.	13.1 ms min.

Caution 1. After the above time has elapsed, the bits are set to 1 in order from the MOST8 bit and remain 1.

Caution 2. The value counted by the OSTC register will only have reached the oscillation stabilization time setting in the oscillation stabilization time select register (OSTS). In the following cases, set the oscillation stabilization time of the OSTS register to the value greater than the counter value which is to be checked by using the OSTC register.

- **If the X1 clock starts oscillation while the high-speed on-chip oscillator clock or subsystem clock is in use as the CPU clock.**
- **If entry to and then release from the STOP mode proceed while the high-speed on-chip oscillator clock is in use as the CPU clock and the X1 clock is oscillating.**
  Therefore, note that the value counted by the OSTC register will only have reached the oscillation stabilization time setting in the OSTS register after release from the STOP mode.

Caution 3. The X1 clock oscillation stabilization time does not include the time until clock oscillation starts ("a" below).



Remark fx: X1 clock oscillation frequency

6.3.5 Oscillation stabilization time select register (OSTS)

The OSTS register is used to select the X1 clock oscillation stabilization time. When the X1 clock is made to oscillate by clearing the MSTOP bit to start operation of the X1 oscillator, actual operation is automatically delayed for the time set in the OSTS register. Use the oscillation stabilization time counter status register (OSTC) to confirm that the specified oscillation stabilization time has elapsed when the CPU clock is switched from the high-speed on-chip oscillator clock or the subsystem clock to the X1 clock or entry to and then release from the STOP mode proceed while the high-speed on-chip oscillator clock is in use as the CPU clock and the X1 clock is oscillating. The OSTC register can be used to check the counter value when counting has reached the time set beforehand in the OSTS register. The OSTS register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 07H.

Figure 6 - 6 Format of Oscillation Stabilization Time Select Register (OSTS)

Address: FFFA3H
After reset: 07H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0

OSTS2	OSTS1	OSTS0	Selection of the oscillation stabilization time		
				fx = 10 MHz	fx = 20 MHz
0	0	0	2^8/fx	25.6 μs	12.8 μs
0	0	1	2^9/fx	51.2 μs	25.6 μs
0	1	0	2^{10}/fx	102 μs	51.2 μs
0	1	1	2^{11}/fx	204 μs	102 μs
1	0	0	2^{13}/fx	819 μs	409 μs
1	0	1	2^{15}/fx	3.27 ms	1.63 ms
1	1	0	2^{17}/fx	13.1 ms	6.55 ms
1	1	1	2^{18}/fx	26.2 ms	13.1 ms

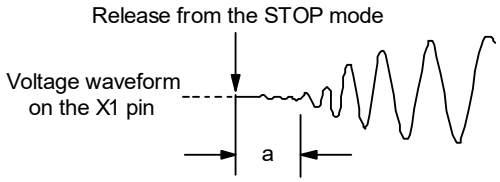
Caution 1. Change the setting of the OSTS register before setting the MSTOP bit of the clock operation status control register (CSC) to 0.

Caution 2. The value counted by the OSTC register will only have reached the oscillation stabilization time setting in the OSTS register. In the following cases, set the oscillation stabilization time of the OSTS register to the value greater than the counter value which is to be checked by using the OSTC register after the oscillation starts.

- **If the X1 clock starts oscillation while the high-speed on-chip oscillator clock or subsystem clock is in use as the CPU clock.**
- **If entry to and then release from the STOP mode proceed while the high-speed on-chip oscillator clock is in use as the CPU clock and the X1 clock is oscillating.**
  Therefore, note that the value counted by the OSTC register will only have reached the oscillation stabilization time setting in the OSTS register after release from the STOP mode.

(Caution and Remark are listed on the next page.)

Caution 3. The X1 clock oscillation stabilization time does not include the time until clock oscillation starts ("a" below).



Remark fx: X1 clock oscillation frequency

6.3.6 Peripheral enable registers 0, 1 (PER0, PER1)

The PER0 and PER1 registers are used to enable or disable the supply of a clock signal to various on-chip peripheral modules. Clock supply to an on-chip peripheral module that is not to be used can be stopped to decrease power consumption and noise. To use any of the on-chip peripheral modules listed below, the clock supplies to which are controlled by these registers, set the corresponding bit to 1 before making the initial settings of the on-chip peripheral module.

- Serial interface IICA
- A/D converter
- Serial array unit
- Timer array unit
- 32-bit interval timer
- Serial interface UARTA

The PER0 and PER1 registers can be set by a 1-bit or 8-bit memory manipulation instruction. The value of each of these registers following a reset is 00H.

Figure 6 - 7 Format of Peripheral Enable Register 0 (PER0)

Address: F00F0H
After reset: 00H
R/W: R/W

Symbol	7	6	<5>	<4>	<3>	<2>	1	<0>
PER0	0	0	ADCEN	IICA0EN Note 1	SAU1EN	SAU0EN	0	TAU0EN

ADCEN	Control of supply of an input clock to the A/D converter
0	Stops supply of an input clock. • The SFRs used by the A/D converter cannot be written.Note 2
1	Enables supply of an input clock. • The SFRs used by the A/D converter can be read and written.

IICA0EN	Control of supply of an input clock to the IICA0 serial interface
0	Stops supply of an input clock. • The SFRs used by the IICA0 serial interface cannot be written.Note 2
1	Enables supply of an input clock. • The SFRs used by the IICA0 serial interface can be read and written.

SAU1EN	Control of supply of an input clock to serial array unit 1
0	Stops supply of an input clock. • The SFRs used by serial array unit 1 cannot be written.Note 2
1	Enables supply of an input clock. • The SFRs used by serial array unit 1 can be read and written.

SAU0EN	Control of supply of an input clock to serial array unit 0
0	Stops supply of an input clock. • The SFRs used by serial array unit 0 cannot be written.Note 2
1	Enables supply of an input clock. • The SFRs used by serial array unit 0 can be read and written.

TAU0EN	Control of supply of an input clock to timer array unit 0
0	Stops supply of an input clock. • The SFRs used by timer array unit 0 cannot be written.Note 2
1	Enables supply of an input clock. • The SFRs used by timer array unit 0 can be read and written.

Note 1. This bit is only present in the 32- to 64-pin products.
Note 2. When an SFR is read, the value returned is 00H or 0000H regardless of its setting.

Caution 1. Be sure to leave bits 7, 6, and 1 set to 0.
Caution 2. Be sure to leave bit 4 in the 20-pin products set to 0.
Caution 3. Do not change the value of a bit of the PER0 register while operation of the corresponding on-chip peripheral module is enabled. Only change a value while the corresponding on-chip peripheral module is stopped.

Figure 6 - 8 Format of Peripheral Enable Register 1 (PER1)

Address: F00FAH
After reset: 00H
R/W: R/W

Symbol	7	6	5	<4>	3	<2>	1	0
PER1	0	0	0	TML32EN	0	UTAEN	0	0

TML32EN	Control of supply of an input clock to the 32-bit interval timer
0	Stops supply of an input clock. • The SFRs used by the 32-bit interval timer cannot be written.**Note**
1	Enables supply of an input clock. • The SFRs used by the 32-bit interval timer can be read and written.

UTAEN	Control of supply of an input clock to the UARTAn serial interface (n = 0)
0	Stops supply of an input clock. • The SFRs used by the UARTAn serial interface cannot be written.**Note**
1	Enables supply of an input clock. • The SFRs used by the UARTAn serial interface can be read and written.

Note When an SFR is read, the value returned is 00H or 0000H regardless of its setting.

Caution 1. Be sure to leave bits 7 to 5, 3, 1, and 0 set to 0.

Caution 2. Do not change the value of a bit of the PER1 register while operation of the corresponding on-chip peripheral module is enabled. Only change a value while the corresponding on-chip peripheral module is stopped.

6.3.7 Subsystem clock supply mode control register (OSMC)

The OSMC register can be used to select the operating clock for the 32-bit interval timer. The OSMC register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is undefined[Note].

Note The values of the WUTMMCK0 and HIPREC bits are respectively 0 and 1.

Figure 6 - 9 Format of Subsystem Clock Supply Mode Control Register (OSMC)

Address: F00F3H
After reset: Undefined[Note 1]
R/W: R/W[Note 2]

Symbol	7	6	5	<4>	3	2	1	<0>
OSMC	0	0	0	WUTMMCK0	×	×	0	HIPREC

WUTMMCK0	Selection of the operating clock for the 32-bit interval timer
0	Nothing is selected.
1	Low-speed on-chip oscillator clock[Note 3]

HIPREC	State of the high-speed on-chip oscillator clock[Note 4]
0	The high-speed on-chip oscillator clock is stopped or waiting for the precision of its oscillation to become stable is in progress.
1	The high-speed on-chip oscillator clock is operating with high precision.

Note 1. The values of the WUTMMCK0 and HIPREC bits are respectively 0 and 1.
Note 2. Bits 3, 2, and 0 are only readable. Writing to these bits is ignored.
Note 3. The low-speed on-chip oscillator clock is only selectable by setting the WUTMMCK0 bit while the 32-bit interval timer is stopped.
<R> **Note 4.** For frequency accuracy of the high-speed on-chip oscillator clock, see **Section 28 Electrical Characteristics**.

Caution 1. Be sure to leave bits 7 to 5 and 1 set to 0.
Caution 2. Do not execute a STOP instruction when HIPREC = 0.

Remark ×: Don't care

6.3.8 Subsystem clock select register (CKSEL)

The CKSEL register is used to select the low-speed on-chip oscillator clock as the subsystem clock. The CKSEL register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 6 - 10 Format of Subsystem Clock Select Register (CKSEL)

Address: FFFA7H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	<0>
CKSEL	0	0	0	0	0	0	0	SELLOSC

SELLOSC	Selection of the low-speed on-chip oscillator clock
0	Nothing is selected.
1	Low-speed on-chip oscillator clock

Caution When changing the value of the SELLOSC bit, be sure to set the CSS bit to 0 (selecting f_{MAIN}) and confirm that the value of the CLS bit is 0 before doing so.

6.3.9 High-speed on-chip oscillator frequency select register (HOCODIV)

The HOCODIV register is used to change the frequency of the high-speed on-chip oscillator set in a user option byte (000C2H). Note that the selectable frequencies depend on the value of the FRQSEL3 bit of the user option byte (000C2H). The HOCODIV register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is that set in the FRQSEL2 to FRQSEL0 bits of the user option byte (000C2H).

Figure 6 - 11 Format of High-speed On-chip Oscillator Frequency Select Register (HOCODIV)

Address: F00A8H
After reset: The value set in the FRQSEL2 to FRQSEL0 bits of a user option byte (000C2H)
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
HOCODIV	0	0	0	0	0	HOCODIV2	HOCODIV1	HOCODIV0

HOCODIV2	HOCODIV1	HOCODIV0	Selection of the high-speed on-chip oscillator clock frequency	
			FRQSEL3 = 0	FRQSEL3 = 1
0	0	0	f_{IH} = 24 MHz	f_{IH} = 32 MHz
0	0	1	f_{IH} = 12 MHz	f_{IH} = 16 MHz
0	1	0	f_{IH} = 6 MHz	f_{IH} = 8 MHz
0	1	1	f_{IH} = 3 MHz	f_{IH} = 4 MHz
1	0	0	Setting prohibited	f_{IH} = 2 MHz
1	0	1	Setting prohibited	f_{IH} = 1 MHz
Other than above			Setting prohibited	

Caution 1. Set the HOCODIV register while ensuring that the voltage is within the usable range for the operating mode of the flash memory set in the flash operating mode select register (FLMODE) both before and after the frequency change. For details about the FLMODE register, see 5.2.1 Flash operating mode select register (FLMODE).

Caution 2. Set the HOCODIV register with the high-speed on-chip oscillator clock (f_{IH}) selected as the CPU/peripheral hardware clock (f_{CLK}).

Caution 3. After use of the HOCODIV register to change the frequency, actually switching of the frequency only proceeds after the following transition times have elapsed.

- **The CPU/peripheral hardware clock continues to operate at the frequency before the change for up to three cycles.**
- **Up to three cycles of waiting are required for the CPU/peripheral hardware clock to be at the post-change frequency.**

Note that even if a value set in the HOCODIV register is the same as its current value, up to three cycles of waiting following the setting are required for stable operation of the CPU/peripheral hardware clock.

6.3.10 High-speed system clock division register (MOSCDIV)

The MOSCDIV register is used to select the division ratio of the high-speed system clock. The MOSCDIV register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 6 - 12 Format of High-speed System Clock Division Register (MOSCDIV)

Address: F0214H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
MOSCDIV	0	0	0	0	0	MOSCDIV2	MOSCDIV1	MOSCDIV0

MOSCDIV2	MOSCDIV1	MOSCDIV0	Selected division ratio for the high-speed system clock	f_{MX} = 20 MHz
0	0	0	f_{MX}	20 MHz
0	0	1	f_{MX}/2	10 MHz
0	1	0	f_{MX}/4	5 MHz
0	1	1	f_{MX}/8	2.5 MHz
1	0	0	f_{MX}/16	1.25 MHz
Other than above			Setting prohibited	

Caution Set the MOSCDIV register while ensuring that the voltage is within the usable range for the operating mode of the flash memory set in the flash operating mode select register (FLMODE) both before and after the frequency change.

6.3.11 High-speed on-chip oscillator trimming register (HIOTRM)

The HIOTRM register is used to adjust the accuracy of the high-speed on-chip oscillator. The accuracy of the high-speed on-chip oscillator frequency can be adjusted through self-measurement of the frequency by using a timer with high accuracy (timer array unit or 32-bit interval timer) for external clock input or in other ways. The HIOTRM register can be set by an 8-bit memory manipulation instruction.

Caution The frequency will vary if the temperature and V_{DD} pin voltage change after accuracy adjustment. When the temperature and V_{DD} voltage change, accuracy adjustment must be executed regularly or before the frequency accuracy is required.

Figure 6 - 13 Format of High-speed On-chip Oscillator Trimming Register (HIOTRM)

Address: F00A0H
After reset: **Note**
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
HIOTRM	0	0	HIOTRM5	HIOTRM4	HIOTRM3	HIOTRM2	HIOTRM1	HIOTRM0

HIOTRM5	HIOTRM4	HIOTRM3	HIOTRM2	HIOTRM1	HIOTRM0	High-speed on-chip oscillator
0	0	0	0	0	0	Minimum speed
0	0	0	0	0	1	
0	0	0	0	1	0	
0	0	0	0	1	1	
0	0	0	1	0	0	
• • •						
1	1	1	1	1	0	
1	1	1	1	1	1	Maximum speed

Note The value of this register following a reset is that adjusted at shipment.

Remark 1. The HIOTRM register holds a six-bit value used to adjust the high-speed on-chip oscillator with an increment of 1 corresponding to an increase of frequency by about 0.05%.

Remark 2. For the usage example of the HIOTRM register, see the **Application Note for RL78/G23 High-speed On-chip Oscillator (HOCO) Clock Frequency Correction (R01AN5555)**.

6.4 System Clock Oscillator

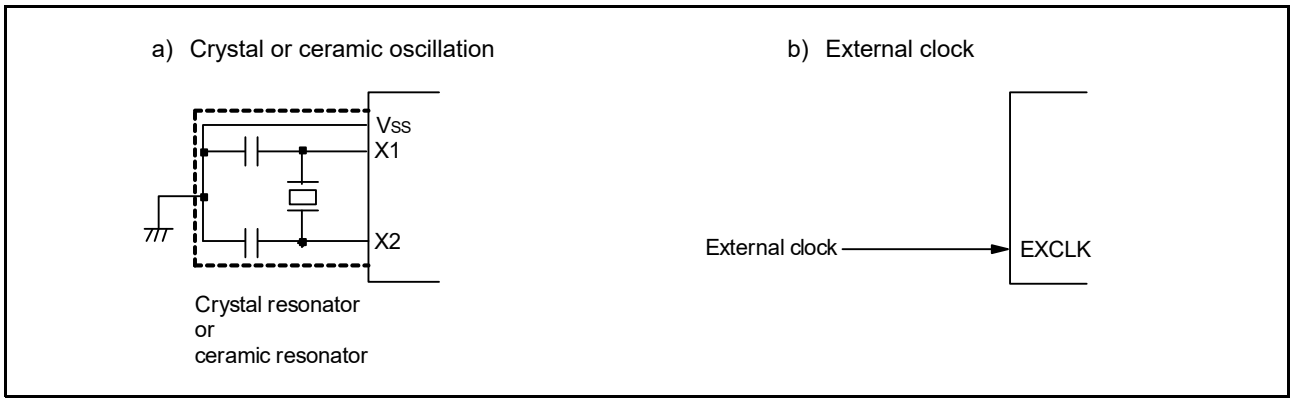
6.4.1 X1 oscillator

The X1 oscillator oscillates with a crystal resonator or ceramic resonator (1 to 20 MHz) connected to the X1 and X2 pins. An external clock can also be input. In this case, input the clock signal to the EXCLK pin. To use the X1 oscillator, set the EXCLK and OSCSEL bits of the clock operation mode control register (CMC) as follows.

- Crystal or ceramic oscillation: EXCLK and OSCSEL = 01B
- External clock input: EXCLK and OSCSEL = 11B

When the X1 oscillator is not used, set the port mode (EXCLK and OSCSEL = 00B). When the pins are not used as port pins, either, see **Table 2 - 2 Connections of Unused Pins**. **Figure 6 - 14** shows the examples of external circuits for the X1 oscillator.

Figure 6 - 14 Examples of External Circuits for the X1 Oscillator



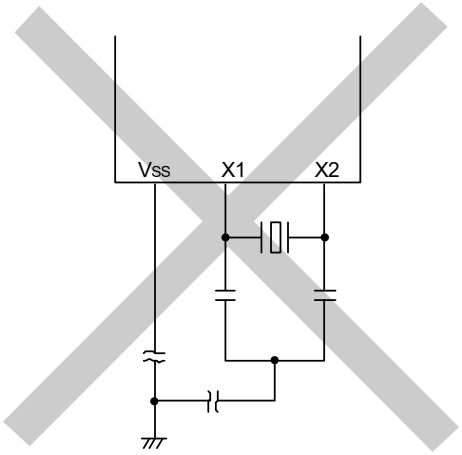
Caution When using the X1 oscillator, wire as follows in the area enclosed in broken lines in Figure 6 - 14 to avoid an adverse effect from wiring capacitance.

- **Keep the wiring length as short as possible.**
- **Do not cross the wiring with the other signal lines. Do not route the wiring near a signal line through which a strong fluctuating current flows.**
- **Always apply the same voltage to the ground point for capacitors in the oscillator as that on Vss. Do not ground the capacitors to a ground pattern through which a strong current flows.**
- **Do not bring a signal line out from the oscillator.**

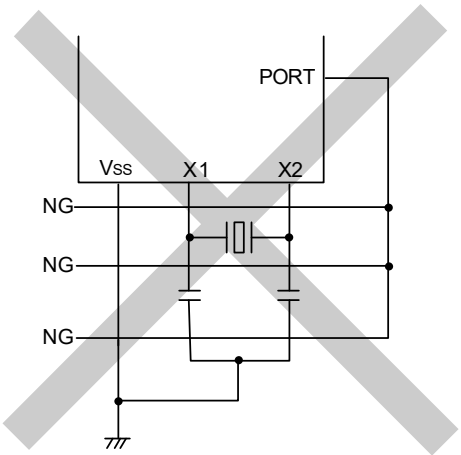
Figure 6 - 15 shows examples of incorrect resonator connection.

Figure 6 - 15 Examples of Incorrect Resonator Connection (1/2)

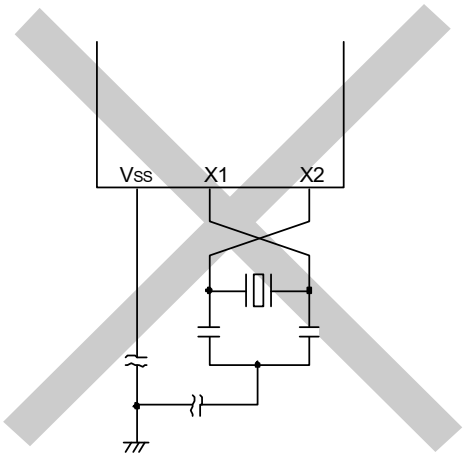
a) The wiring runs of the oscillator are too long.



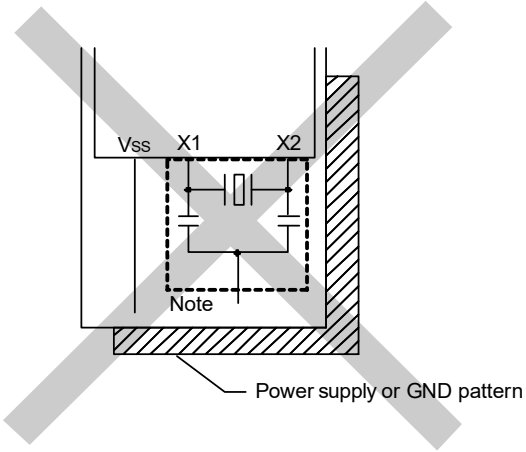
b) The wiring runs of the oscillator cross signal lines.



c) The wiring runs for the X1 and X2 signals cross each other.



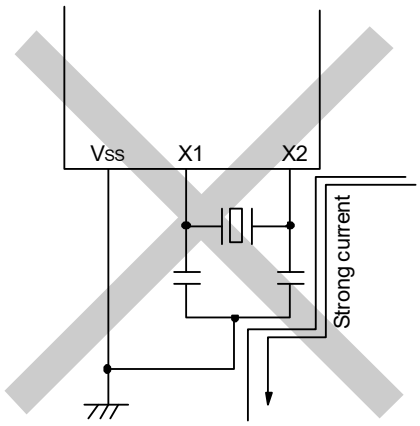
d) The power supply or GND pattern is under the wiring runs for the X1 and X2 signals.



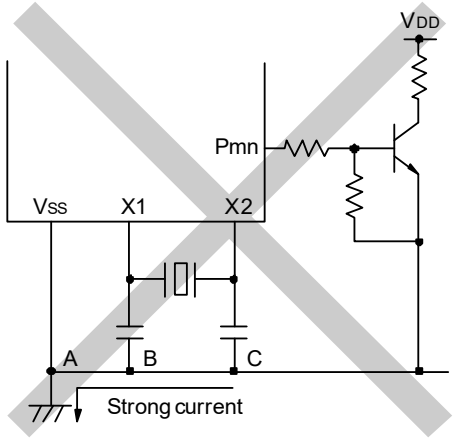
Note Do not place a power supply or GND pattern under the wiring section (section enclosed in broken lines in the figure) of the X1 and X2 pins and the resonator in a multi-layer board or double-sided board. Do not configure a layout that will cause capacitance elements and affect the oscillation characteristics.

Figure 6 - 15 Examples of Incorrect Resonator Connection (2/2)

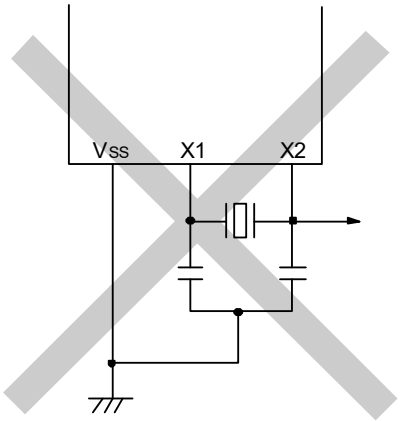
e) A signal line for the oscillator and a signal line through which a strong fluctuating current flows are close to each other.



f) Current flows through the ground line of the oscillator (making the voltage at points A, B, and C fluctuate).



g) A signal line is brought out.



Caution When the wiring runs for the X2 and X1 signals are in parallel, crosstalk noise from the X2 signal line may be imposed on the X1 signal, resulting in malfunctions.

6.4.2 High-speed on-chip oscillator

Each product incorporates a high-speed on-chip oscillator. The frequency can be selected from among 32, 24, 16, 12, 8, 6, 4, 3, 2, and 1 MHz by using the user option byte (000C2H). Oscillation can be controlled by the HIOSTOP bit of the clock operation status control register (CSC). The high-speed on-chip oscillator automatically starts oscillating after release from the reset state.

6.4.3 Low-speed on-chip oscillator

Each product incorporates a low-speed on-chip oscillator. The low-speed on-chip oscillator operates when any of the following conditions is met.

- The watchdog timer is operating.
- The value of one or both of the WUTMMCK0 bit of the subsystem clock supply mode control register (OSMC) and the SELLOSC bit of the subsystem clock select register (CKSEL) is 1.

The low-speed on-chip oscillator is stopped when the watchdog timer is stopped and both WUTMMCK0 and SELLOSC are set to 0.

6.5 Operations of the Clock Generator

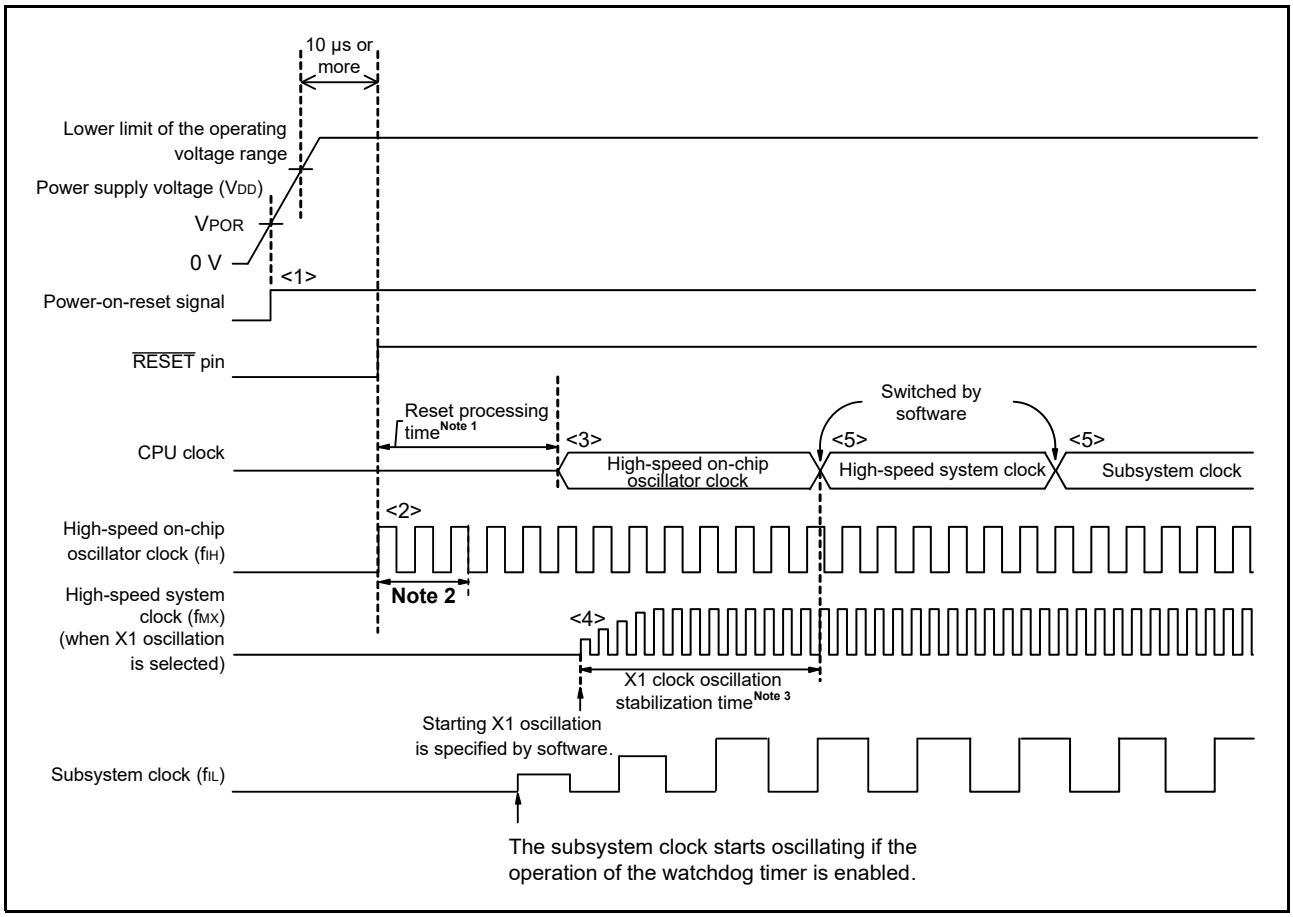
The clock generator generates the following clocks and controls the operation modes of the CPU, such as standby mode (see **Figure 6 - 1 Block Diagram of Clock Generator**).

- Main system clocks (f_{MAIN})
  - High-speed system clocks (f_{MX})
    X1 clock (f_X)
    External main system clock (f_{EX})
  - High-speed on-chip oscillator clock (f_{IH})
- Subsystem clock (f_{SUB})
  - Low-speed on-chip oscillator clock (f_{IL})
- CPU/peripheral hardware clock (f_{CLK})
- Peripheral clocks
  - High-speed on-chip oscillator peripheral clock (f_{IHP})
  - High-speed peripheral clock (f_{MXP})
  - Low-speed peripheral clock (f_{SXP})

The CPU starts operation when the high-speed on-chip oscillator starts outputting after release from the reset state in products of this group.

Figure 6 - 16 shows the clock generator operation when power supply voltage is turned on.

Figure 6 - 16 Clock Generator Operation When Power Supply Voltage Is Turned On



<1> When the power is turned on, an internal reset signal is generated by the power-on-reset circuit. Note that the reset state is maintained after a reset by the voltage detection circuit or an external reset until the voltage reaches the range of operating voltage described in **28.4 AC Characteristics** (the above figure is an example when the external reset is in use).

<2> On release from the reset state, the high-speed on-chip oscillator automatically starts oscillating. If the operation of the watchdog timer is enabled, the low-speed on-chip oscillator also automatically starts oscillating.

<3> The CPU starts operation with the high-speed on-chip oscillator clock after waiting for the voltage to become stable and a reset processing have been performed after release from the reset state.

<4> Set the start of oscillation of the X1 clock via software (see **6.6.2 Example of setting the X1 oscillator clock**).

<5> When switching the CPU clock to the X1 clock, wait for the clock oscillation to become stable, and then make the setting for switching via software (see **6.6.2 Example of setting the X1 oscillator clock**).

Note 1. For the reset processing time, see **Section 18 Power-on-reset Circuit (POR)**.

Note 2. The internal reset processing time includes the oscillation accuracy stabilization time of the high-speed on-chip oscillator clock.

Note 3. When releasing a reset, confirm the oscillation stabilization time for the X1 clock using the oscillation stabilization time counter status register (OSTC).

Caution Waiting for the oscillation stabilization time is not required when external clock input through the EXCLK pin is used.

6.6 Controlling Clocks

6.6.1 Example of setting the high-speed on-chip oscillator

After release from the reset state, the CPU/peripheral hardware clock (f_{CLK}) always starts operating with the high-speed on-chip oscillator clock as the source. The frequency of the high-speed on-chip oscillator can be selected from among 32, 24, 16, 12, 8, 6, 4, 3, 2, and 1 MHz by using the FRQSEL3 to FRQSEL0 bits of a user option byte (000C2H). In addition, the frequency can be changed by the high-speed on-chip oscillator frequency select register (HOCODIV).

[User option byte setting]

Address: 000C2H

	7	6	5	4	3	2	1	0
User option byte (000C2H)	CMODE1 0/1	CMODE0 0/1	1	0	FRQSEL3 0/1	FRQSEL2 0/1	FRQSEL1 0/1	FRQSEL0 0/1

FRQSEL3	FRQSEL2	FRQSEL1	FRQSEL0	Frequency of the high-speed on-chip oscillator
1	0	0	0	32 MHz
0	0	0	0	24 MHz
1	0	0	1	16 MHz
0	0	0	1	12 MHz
1	0	1	0	8 MHz
0	0	1	0	6 MHz
1	0	1	1	4 MHz
0	0	1	1	3 MHz
1	1	0	0	2 MHz
1	1	0	1	1 MHz
Other than above				Setting prohibited

[High-speed on-chip oscillator frequency select register (HOCODIV) setting]

Address: F00A8H

Symbol	7	6	5	4	3	2	1	0
HOCODIV	0	0	0	0	0	HOCODIV2	HOCODIV1	HOCODIV0

HOCODIV2	HOCODIV1	HOCODIV0	Selection of the high-speed on-chip oscillator clock frequency	
			FRQSEL3 = 0	FRQSEL3 = 1
0	0	0	f_{IH} = 24 MHz	f_{IH} = 32 MHz
0	0	1	f_{IH} = 12 MHz	f_{IH} = 16 MHz
0	1	0	f_{IH} = 6 MHz	f_{IH} = 8 MHz
0	1	1	f_{IH} = 3 MHz	f_{IH} = 4 MHz
1	0	0	Setting prohibited	f_{IH} = 2 MHz
1	0	1	Setting prohibited	f_{IH} = 1 MHz
Other than above			Setting prohibited	

6.6.2 Example of setting the X1 oscillator clock

After release from the reset state, the CPU/peripheral hardware clock (f_{CLK}) always starts operating with the high-speed on-chip oscillator clock as the source. To subsequently change the source clock to the X1 oscillator clock, set and start the oscillator by using the oscillation stabilization time select register (OSTS), clock operation mode control register (CMC), and clock operation status control register (CSC), and wait for the oscillation to become stable by using the oscillation stabilization time counter status register (OSTC). After the oscillation becomes stable, set the X1 oscillator clock as the source of f_{CLK} by using the system clock control register (CKC).

[Register settings] Set the registers in the order of <1> to <5> below.

<1> Set the OSCSEL bit of the CMC register to 1 to make the X1 oscillator operate. Also set the AMPH bit to 1 if f_X is more than 10 MHz.

	7	6	5	4	3	2	1	0
CMC	EXCLK 0	OSCSEL 1	0	0	0	0	0	AMPH 0/1

<2> Use the OSTS register to select the oscillation stabilization time for the X1 oscillator after release from the STOP mode.
Example: Set the register as shown below if waiting is to be for at least 102 μs with a 10-MHz resonator.

	7	6	5	4	3	2	1	0
OSTS	0	0	0	0	0	OSTS2 0	OSTS1 1	OSTS0 0

<3> Clear the MSTOP bit of the CSC register to 0 to start X1 oscillation.

	7	6	5	4	3	2	1	0
CSC	MSTOP 0	1	0	0	0	0	0	HIOSTOP 0

<4> Use the OSTC register to wait for oscillation of the X1 oscillator to become stable.
Example: Wait until counting has reached the following value if waiting is to be for at least 102 μs with a 10-MHz resonator.

	7	6	5	4	3	2	1	0
OSTC	MOST8 1	MOST9 1	MOST10 1	MOST11 0	MOST13 0	MOST15 0	MOST17 0	MOST18 0

<5> Use the MCM0 bit of the CKC register to specify the X1 oscillator clock as the source of the CPU/peripheral hardware clock.

	7	6	5	4	3	2	1	0
CKC	CLS 0	CSS 0	MCS 0	MCM0 1	0	0	0	0

Caution When using the system clock control register (CKC) to change the main system clock (f_{MAIN}), do so while the voltage is within the usable range for the operating mode of the flash memory set in the user option byte (000C2H) and in the flash operating mode select register (FLMODE) both before and after changing the clock. For details about the FLMODE register, see 5.2.1 Flash operating mode select register (FLMODE).

6.6.3 State transitions of the CPU clock

Figure 6 - 17 shows the state transitions of the CPU clock.

Figure 6 - 17 State Transitions of the CPU Clock

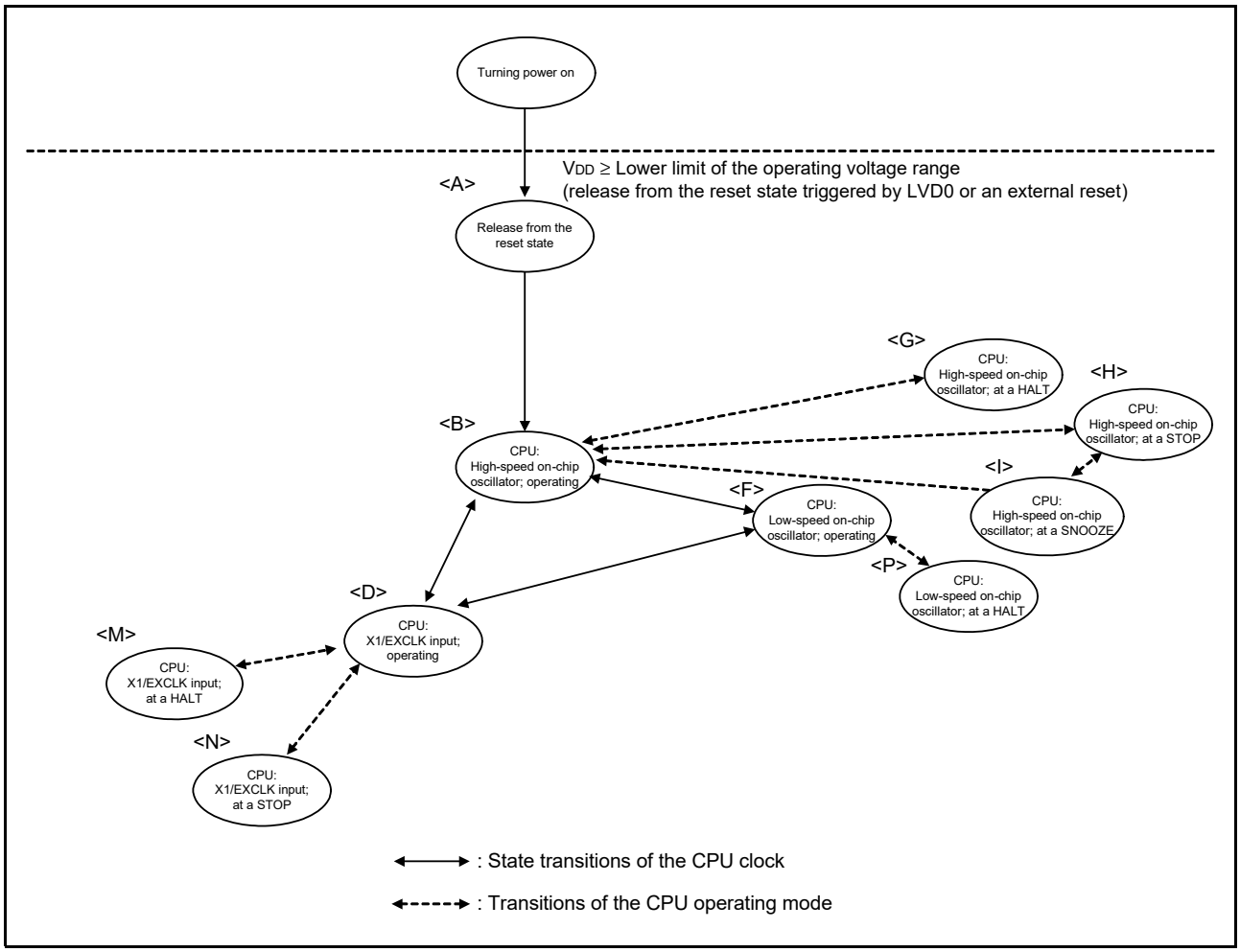


Table 6 - 2 shows examples of transitions of the CPU clock and SFR settings.

Table 6 - 2 Examples of Transitions of the CPU Clock and SFR Settings (1/3)

1. Transition to state <B> where the CPU is operating with the high-speed on-chip oscillator clock after release from the reset state <A>
   Scope of state transitions: <A> → <B>

Clock after Transition	SFR Setting
High-speed on-chip oscillator	SFR setting not required (default state after release from the reset state)

2. Transitions to state <B> where the CPU is operating with the high-speed on-chip oscillator clock
   Scope of state transitions: <D> → <B>, <F> → <B>

(Sequence of setting the SFRs) →

SFR to be Set / Clock after Transition	CSC Register	Waiting for Oscillation Stabilization	CKC Register	
	HIOSTOP		CSS	MCM0
High-speed on-chip oscillator	0	5 μs	0	0

Unnecessary if the high-speed on-chip oscillator clock is already running

Remark <A> to <P> in **Table 6 - 2** correspond to <A> to <P> in **Figure 6 - 17 State Transitions of the CPU Clock**.

Table 6 - 2 Examples of Transitions of the CPU Clock and SFR Settings (2/3)

3. Transitions to state <D> where the CPU is operating with the high-speed system clock
   Scope of state transitions: <B> → <D>, <F> → <D>

(Sequence of setting the SFRs) →

SFR to be Set / Clock after Transition	CMC Register[Note 1]			OSTS Register	CSC Register	OSTC Register	CKC Register	
	EXCLK	OSCSEL	AMPH		MSTOP		CSS	MCM0
X1 clock: 1 MHz ≤ fx ≤ 10 MHz	0	1	0	Note 2	0	Must be checked	0	1
X1 clock: 10 MHz < fx ≤ 20 MHz	0	1	1	Note 2	0	Must be checked	0	1
External main clock	1	1	×	Note 2	0	Need not be checked	0	1

Unnecessary if these bits are already set

Unnecessary if the high-speed system clock is already running

Note 1. The clock operation mode control register (CMC) can be changed only once after release from the reset state. This register setting is not necessary if it has already been set.

Note 2. Set the oscillation stabilization time as follows.

- Desired oscillation stabilization time counted by the oscillation stabilization time counter status register (OSTC) ≤ Oscillation stabilization time set by the oscillation stabilization time select register (OSTS)

Caution Set a clock after the supply voltage has reached the operating voltage range for the clock to be set (see Section 28 Electrical Characteristics).

Remark 1. ×: Don't care.

Remark 2. <A> to <P> in **Table 6 - 2** correspond to <A> to <P> in **Figure 6 - 17 State Transitions of the CPU Clock**.

Table 6 - 2 Examples of Transitions of the CPU Clock and SFR Settings (3/3)

4. Transitions to state <F> where the CPU is operating with the low-speed on-chip oscillator clock
   Scope of state transitions: <B> → <F>, <D> → <F>

(Sequence of setting the SFRs) →

SFR to be Set / Clock after Transition	CKSEL	Waiting for Oscillation Accuracy Stabilization	CKC Register
	SELLOSC		CSS
Low-speed on-chip oscillator	1	80 μs	1

Unnecessary if the low-speed on-chip oscillator clock is already running

5. Transitions from the CPU operating mode <B>, <D>, or <F> to the HALT mode <G>, <M>, or <P>
   Scope of state transitions: <B> → <G>, <D> → <M>, <F> → <P>

Mode after Transition	Description
HALT mode	Executing a HALT instruction

6. Transitions from the CPU operating mode <B> or <D> to the STOP mode <H> or <N>
   Scope of state transitions: <B> → <H>, <D> → <N>

(Setting sequence) →

Mode after Transition	Description		
STOP mode	Stopping peripheral functions that cannot operate in STOP mode	Setting the OSTS register	Executing a STOP instruction

Only necessary if the CPU is shifting from the state of operating with the high-speed system clock to the STOP mode

7. Transitions between a STOP mode <H> and SNOOZE mode <I>
   For details about the setting for switching from the STOP mode to the SNOOZE mode, see **Section 16 Standby Function** and descriptions of the SNOOZE mode functions of the on-chip peripheral modules.

Remark <A> to <P> in **Table 6 - 2** correspond to <A> to <P> in **Figure 6 - 17 State Transitions of the CPU Clock**.

6.6.4 Conditions before changing the CPU clock and processing after changing the CPU clock

Table 6 - 3 shows the conditions before changing the CPU clock and processing after changing the CPU clock.

Table 6 - 3 Changing the CPU Clock

CPU Clock		Conditions before Change	Processing after Change
Before Change	After Change		
High-speed on-chip oscillator clock	X1 clock	X1 oscillation has become stable. • OSCSEL = 1, EXCLK = 0, MSTOP = 0 • The oscillation stabilization time has elapsed.	Operating current can be reduced by stopping the high-speed on-chip oscillator (HIOSTOP = 1) after checking that the CPU clock is changed.
	External main system clock	External clock input from the EXCLK pin is enabled. • OSCSEL = 1, EXCLK = 1, MSTOP = 0	
	Low-speed on-chip oscillator clock	The low-speed on-chip oscillator is selected. • SELLOSC = 1	
X1 clock	High-speed on-chip oscillator clock	The high-speed on-chip oscillator is enabled. • HIOSTOP = 0 • The oscillation stabilization time has elapsed.	X1 oscillation can be stopped (MSTOP = 1) after checking that the CPU clock is changed.
	External main system clock	Transition is not possible.	—
	Low-speed on-chip oscillator clock	The low-speed on-chip oscillator is selected. • SELLOSC = 1	X1 oscillation can be stopped (MSTOP = 1) after checking that the CPU clock is changed.
External main system clock	High-speed on-chip oscillator clock	The high-speed on-chip oscillator is enabled. • HIOSTOP = 0 • The oscillation stabilization time has elapsed.	External main system clock input can be disabled (MSTOP = 1) after checking that the CPU clock is changed.
	X1 clock	Transition is not possible.	—
	Low-speed on-chip oscillator clock	The low-speed on-chip oscillator is selected. • SELLOSC = 1	External main system clock input can be disabled (MSTOP = 1) after checking that the CPU clock is changed.
Low-speed on-chip oscillator clock	High-speed on-chip oscillator clock	The high-speed on-chip oscillator is operating and the high-speed on-chip oscillator clock is selected as the main system clock. • HIOSTOP = 0, MCS = 0	—
	X1 clock	X1 oscillation has become stable and the high-speed system clock is selected as the main system clock. • OSCSEL = 1, EXCLK = 0, MSTOP = 0 • The oscillation stabilization time has elapsed. • MCS = 1	
	External main system clock	External clock input from the EXCLK pin is enabled and the high-speed system clock is selected as the main system clock. • OSCSEL = 1, EXCLK = 1, MSTOP = 0 • MCS = 1	

6.6.5 Time required for switchover of the CPU clock and main system clock

By setting the CSS and MCM0 bits of the system clock control register (CKC), the CPU clock can be switched between the main system clock and subsystem clock and the main system clock can be switched between the on-chip oscillator clock and high-speed system clock. In actual operation, the clock is not switched immediately after writing to the CKC register; the CPU continues to operate with the prior clock for several clock cycles after writing proceeds (see **Table 6 - 4** to **Table 6 - 6**). Whether the source of the CPU clock is the main system clock or subsystem clock can be ascertained from the CLS bit of the CKC register. Whether the source of the main system clock is the high-speed system clock or high-speed on-chip oscillator clock can be ascertained from the MCS bit of the CKC register. When the CPU clock is switched, the peripheral hardware clock is also switched.

Table 6 - 4 Maximum Time Required for Main System Clock Switchover

Clock A	Switching Directions	Clock B	Remark
f_{IH}	◀⟷▶	f_{MX}	See **Table 6 - 5**.
f_{MAIN}	◀⟷▶	f_{SUB}	See **Table 6 - 6**.

Table 6 - 5 Maximum Number of Clock Cycles Required for f_{IH} ↔ f_{MX}

Set Value before Switchover		Set Value after Switchover	
MCM0		MCM0	
		0 ($f_{MAIN} = f_{IH}$)	1 ($f_{MAIN} = f_{MX}$)
0 ($f_{MAIN} = f_{IH}$)	$f_{MX} \geq f_{IH}$		2 cycles
	$f_{MX} < f_{IH}$		2 f_{IH}/f_{MX} cycles
1 ($f_{MAIN} = f_{MX}$)	$f_{MX} \geq f_{IH}$	2 f_{MX}/f_{IH} cycles	
	$f_{MX} < f_{IH}$	2 cycles	

Table 6 - 6 Maximum Number of Clock Cycles Required for f_{MAIN} ↔ f_{SUB}

Set Value before Switchover	Set Value after Switchover	
CSS	CSS	
	0 ($f_{CLK} = f_{MAIN}$)	1 ($f_{CLK} = f_{SUB}$)
0 ($f_{CLK} = f_{MAIN}$)		1 + 2 f_{MAIN}/f_{SUB} cycles
1 ($f_{CLK} = f_{SUB}$)	3 cycles	

Remark 1. The number of clock cycles listed in **Table 6 - 5** and **Table 6 - 6** is the number of cycles of the CPU clock before switchover.

Remark 2. Calculate the number of clock cycles in **Table 6 - 5** and **Table 6 - 6** by rounding up the number after the decimal position.

Example: When switching the main system clock from the high-speed on-chip oscillator clock (8 MHz selected) to the high-speed system clock (in this case, f_{IH} = 8 MHz, f_{MX} = 10 MHz)

1 + f_{IH}/f_{MX} = 1 + 8/10 = 1 + 0.8 = 1.8 → 2 cycles

6.6.6 Conditions before clock oscillation is stopped

The following lists the register flag settings for stopping the clock oscillation (disabling external clock input) and conditions before the clock oscillation is stopped. When stopping the clock, confirm the conditions before clock oscillation is stopped.

Table 6 - 7 Conditions Before the Clock Oscillation Is Stopped and Flag Settings

Clock	Conditions Before Clock Oscillation is Stopped (External Clock Input is Disabled)	Flag Settings of the SFRs
High-speed on-chip oscillator clock	MCS = 1 or CLS = 1 (The CPU is operating with a clock other than the high-speed on-chip oscillator clock.)	HIOSTOP = 1
X1 clock	MCS = 0 or CLS = 1 (The CPU is operating with a clock other than the high-speed system clock.)	MSTOP = 1
External main system clock		
Low-speed on-chip oscillator clock**Note**	CLS = 0 (The CPU is operating with a clock other than the low-speed on-chip oscillator clock.)	SELLOSC = 0 WUTMMCK0 = 0

Note The low-speed on-chip oscillator clock is not stopped while the WDT is operating.

6.7 Resonator and Oscillator Constants

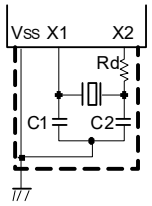
For the resonators for which operation has been verified and their oscillator constants (reference values), see the target product page on the Renesas Web site.

Caution 1. The constants for these oscillator circuits are reference values based on specific environments set up for evaluation by the manufacturers. For actual applications, request evaluation by the manufacturer of the oscillator circuit mounted on a board. Furthermore, if you are switching from a different product to this microcontroller, and whenever you change the board, again request evaluation by the manufacturer of the oscillator circuit mounted on the new board.

Caution 2. The oscillation voltage and oscillation frequency only indicate the oscillator characteristic. Use products of this group so that the internal operation conditions are within the specifications of the DC and AC characteristics.

Figure 6 - 18 Examples of External Circuits

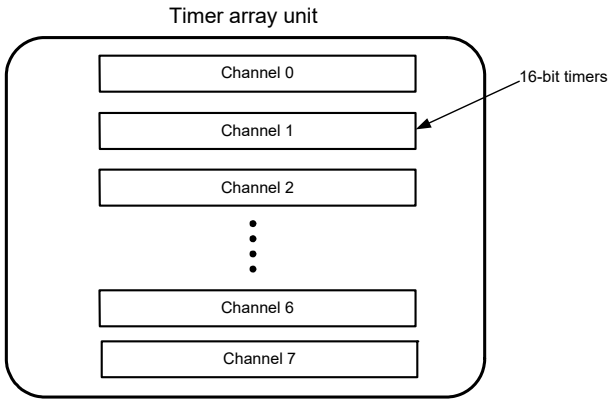
a) X1 oscillation



Section 7 Timer Array Unit (TAU)

<R> This product has a single timer array unit, which includes eight timer channels. The presence or absence of timer I/O pins depends on the product. See **Table 7 - 2 Timer I/O Pins Provided in Each Product** for details. Most of the following descriptions in this section use the 64-pin products as an example.

The timer array unit has eight 16-bit timers. Each 16-bit timer is called a channel and can be used as an independent timer. In addition, two or more channels can be used to create a high-accuracy timer.



For details about each function, see the table below.

Independent channel operation function	Simultaneous channel operation function
• Interval timer (see **7.8.1**) • Square wave output (see **7.8.1**) • External event counter (see **7.8.2**) • Divider**Note** (see **7.8.3**) • Input pulse interval measurement (see **7.8.4**) • Measurement of low- or high-level width of input signal (see **7.8.5**) • Delay counter (see **7.8.6**)	• One-shot pulse output (see **7.9.1**) • PWM output (see **7.9.2**) • Multiple PWM output (see **7.9.3**)

Note Only channel 0 supports this function.

Each of the 16-bit timer channels 1 and 3 can be used as two 8-bit timers, represented by the higher- and lower-order bytes. The following functions can be realized by using channels 1 and 3 as 8-bit timers:

- Interval timer (upper or lower 8-bit timer)/square wave output (only for lower 8-bit timer)
- External event counter (only for lower 8-bit timer)
- Delay counter (only for lower 8-bit timer)

Channel 7 can be used to realize LIN-bus communication operating in combination with UART2 of the serial array unit.

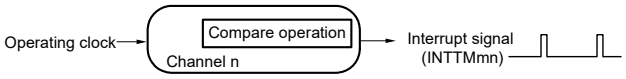
7.1 Functions of Timer Array Unit

Timer array unit has the following functions.

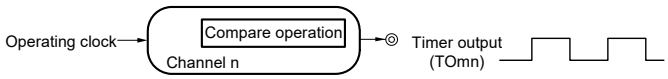
7.1.1 Independent channel operation function

By operating a channel independently, it can be used for the following seven purposes without being affected by the operation mode of other channels.

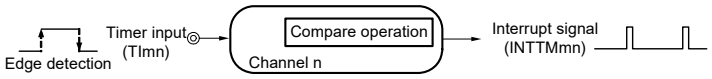
1. Interval timer
   Each timer of a unit can be used as a reference timer that generates an interrupt (INTTMmn) at fixed intervals.



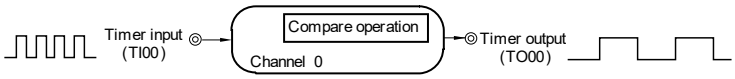
2. Square wave output
   A toggle operation is performed each time INTTMmn interrupt is generated and a square wave with a duty factor of 50% is output from a timer output pin (TOmn).



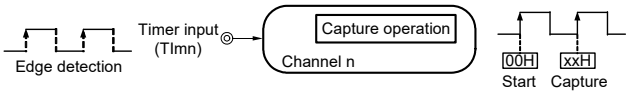
3. External event counter
   Each timer of a unit can be used as an event counter that generates an interrupt when the number of the valid edges of a signal input to the timer input pin (TImn) has reached a specific value.



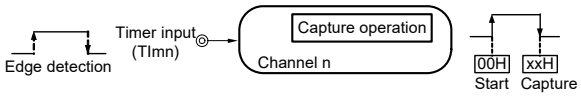
4. Divider function (only for channel 0)
   A clock input from a timer input pin (TI00) is divided and output from an output pin (TO00).



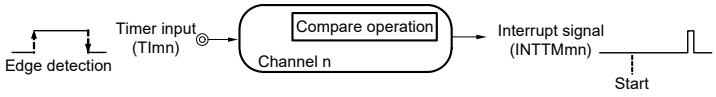
5. Input pulse interval measurement
   Counting is started by the valid edge of a pulse signal input to a timer input pin (TImn). The count value of the timer is captured at the valid edge of the next pulse. In this way, the interval of the input pulse can be measured.



6. Measurement of low- or high-level width of input signal
   Counting is started by a single edge of the signal input to the timer input pin (TImn), and the count value is captured at the other edge. In this way, the low- or high-level width of the input signal can be measured.



7. Delay counter
   Counting is started at the valid edge of the signal input to the timer input pin (TImn), and an interrupt is generated after any delay period.



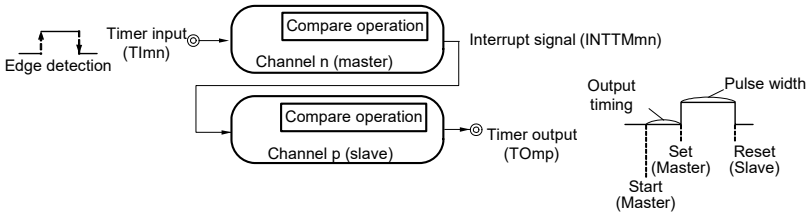
Remark 1. m: Unit number (m = 0), n: Channel number (n = 0 to 7)

Remark 2. The presence or absence of timer I/O pins of channel 0 to 7 depends on the product. See **Table 7 - 2 Timer I/O Pins Provided in Each Product** for details.

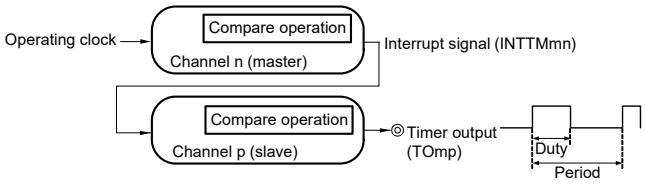
7.1.2 Simultaneous channel operation function

By using the combination of a master channel (a reference timer mainly controlling the cycle) and slave channels (timers operating according to the master channel), channels can be used for the following three purposes.

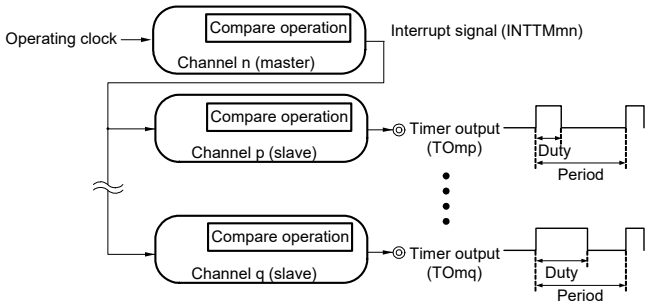
1. One-shot pulse output
   Two channels are used as a set to generate a one-shot pulse with a specified output timing and a specified pulse width.



2. PWM (Pulse Width Modulation) output
   Two channels are used as a set to generate a pulse with a specified period and a specified duty factor.



3. Multiple PWM (Pulse Width Modulation) output
   By extending the PWM function and using one master channel and two or more slave channels, up to seven types of PWM signals that have a specific period and a specified duty factor can be generated.



Caution **For details about the rules of simultaneous channel operation function, see 7.4.1 Basic specifications of the simultaneous channel operation function.**

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7),
p, q: Slave channel number (n < p < q ≤ 7)

7.1.3 8-bit timer function (only for channels 1 and 3)

The 8-bit timer function allows the use of a 16-bit timer configured by cascade-connecting two 8-bit timer channels. This function can only be used with channels 1 and 3.

Caution **There are several rules for using the 8-bit timer function. For details, see 7.4.2 Basic specifications of the 8-bit timer function (only for channels 1 and 3).**

7.1.4 LIN-bus function (only for channel 7)

Timer array unit is used to check whether signals received in LIN-bus communication match the LIN-bus communication format.

1. Detection of wakeup signal
   The timer starts counting at the falling edge of a signal input to the serial data input pin (RxD2) of UART2 and the count value of the timer is captured at the rising edge. In this way, a low-level width can be measured. If the low-level width is greater than a specific value, it is recognized as a wakeup signal.
2. Detection of break field
   The timer starts counting at the falling edge of a signal input to the serial data input pin (RxD2) of UART2 after a wakeup signal is detected, and the count value of the timer is captured at the rising edge. In this way, a low-level width is measured. If the low-level width is greater than a specific value, it is recognized as a break field.
3. Measurement of pulse width of sync field
   After a break field is detected, the low-level width and high-level width of the signal input to the serial data input pin (RxD2) of UART2 are measured. From the bit interval of the sync field measured in this way, a baud rate is calculated.

Remark For details about setting up the operations used to implement the LIN bus, see **7.3.14 Input switch control register (ISC)** and **7.8.5 Operation for input signal low- and high-level width measurement**.

7.2 Configuration of Timer Array Unit

Timer array unit includes the following hardware.

Table 7 - 1 Configuration of Timer Array Unit

Item	Configuration
Timer/counter	Timer counter register mn (TCRmn)
Register	Timer data register mn (TDRmn)
Timer input	TI00 to TI07[Note 1], RxD2 pin (for the LIN bus)
Timer output	TO00 to TO07 pins[Note 1], output controller
Control registers	<Registers of unit setting block> • Peripheral enable register 0 (PER0) • Peripheral reset control register 0 (PRR0) • Timer clock select register m (TPSm) • Timer channel enable status register m (TEm) • Timer channel start register m (TSm) • Timer channel stop register m (TTm) • Timer input select register 0 (TIS0) • Timer output enable register m (TOEm) • Timer output register m (TOm) • Timer output level register m (TOLm) • Timer output mode register m (TOMm)
	<Registers of each channel> • Timer mode register mn (TMRmn) • Timer status register mn (TSRmn) • Input switch control register (ISC) • Noise filter enable register 1 (NFEN1) • Port mode registers (PMxx)[Note 2] • Port registers (Pxx)[Note 2] • Port output mode registers (POMxx)[Note 2] • Port mode control A registers (PMCAxx)[Note 2]

Note 1. The presence or absence of timer I/O pins of channel 0 to 7 depends on the product. See **Table 7 - 2 Timer I/O Pins Provided in Each Product** for details.

Note 2. The port mode registers (PMxx), port registers (Pxx), port output mode registers (POMxx), and port mode control A registers (PMCAxx) to be set differ depending on the product. For details, see **4.5.4 Examples of register settings for port and alternate functions**.

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

The presence or absence of timer I/O pins in each timer array unit channel depends on the product.

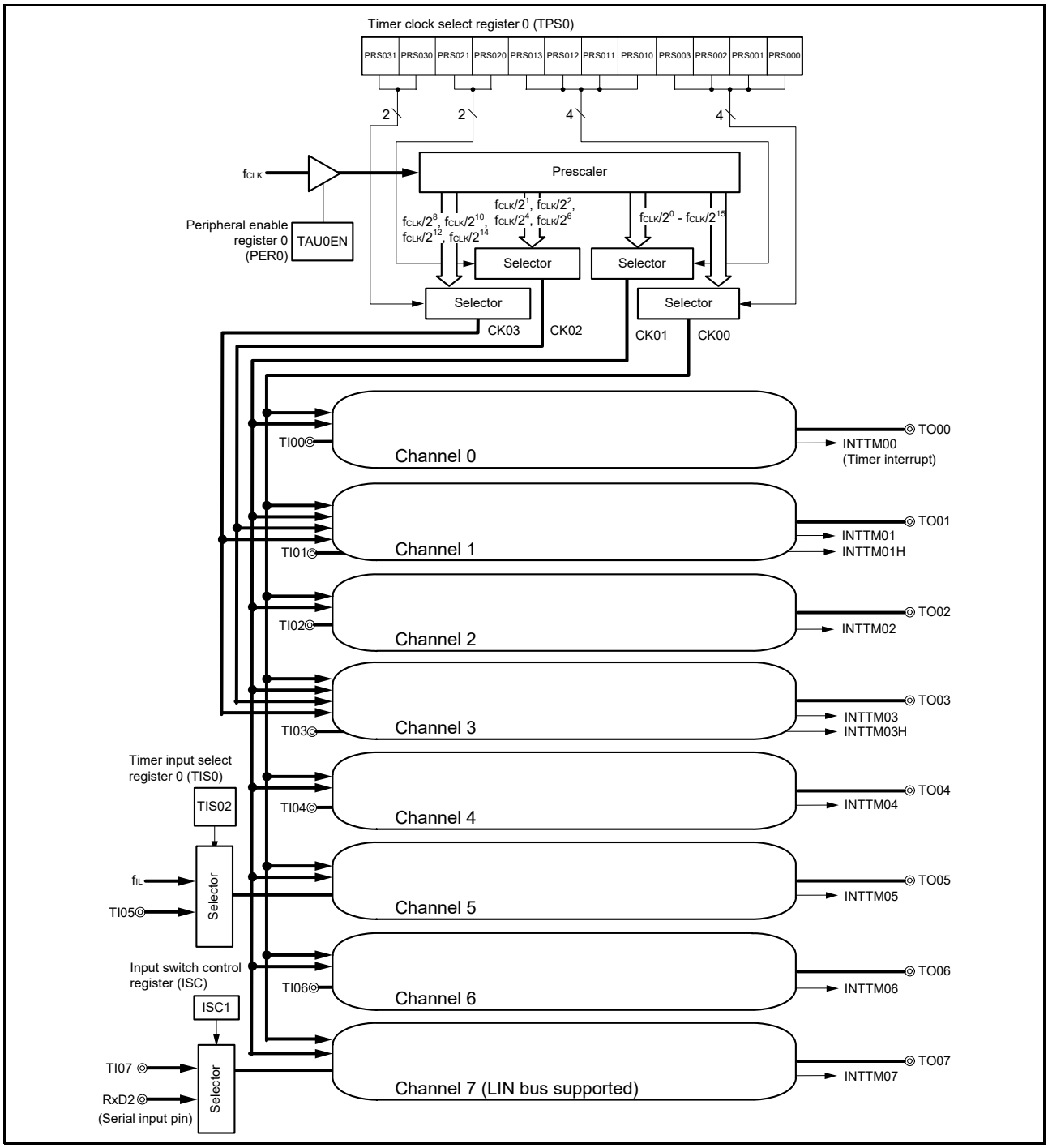
Table 7 - 2 Timer I/O Pins Provided in Each Product

Timer array unit channels		I/O Pins of Each Product	
		20-pin	32- to 64-pin
Unit 0	Channel 0	TI00, TO00	TI00, TO00
	Channel 1	TI01/TO01	TI01/TO01
	Channel 2	TI02/TO02	TI02/TO02
	Channel 3	—	(TI03/TO03)
	Channel 4	—	(TI04/TO04)
	Channel 5	TI05/TO05	(TI05/TO05)
	Channel 6	TI06/TO06	(TI06/TO06)
	Channel 7	TI07/TO07	(TI07/TO07)

Remark 1. When timer input and timer output are shared by the same pin, either only timer input or only timer output can be used.
Remark 2. —: The channel is not available.
Remark 3. Pins in parentheses are assigned when bit 0 of the peripheral I/O redirection register (PIOR0) is set to 1.

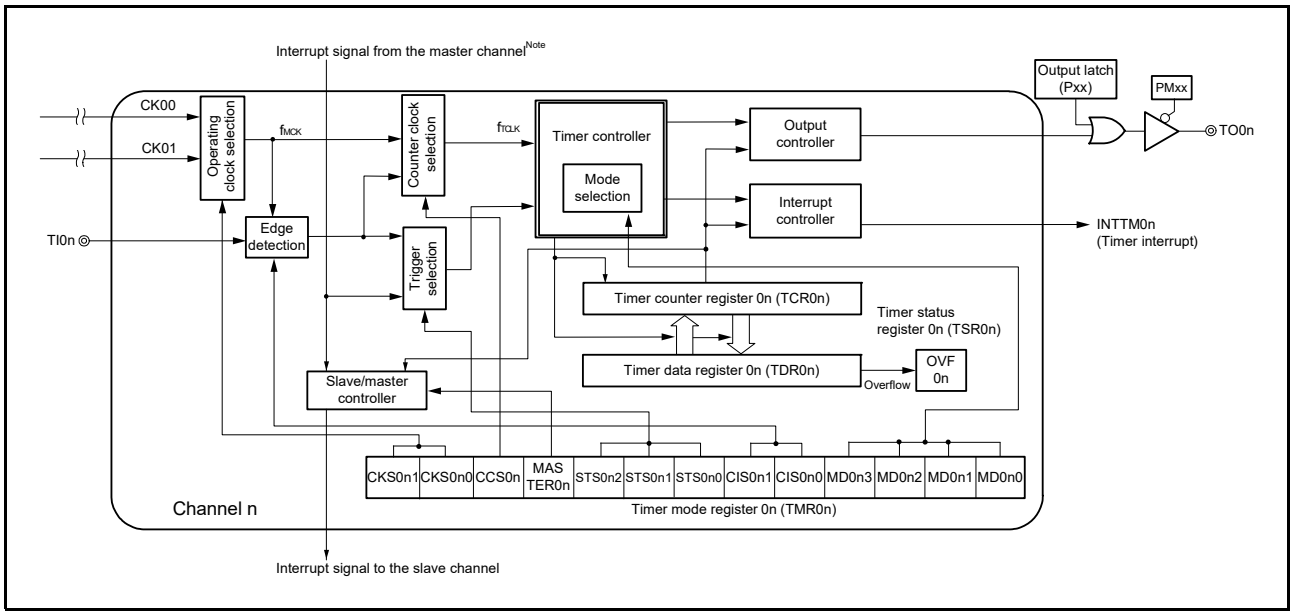
Figure 7 - 1 shows the block diagram of timer array unit 0.

Figure 7 - 1 Block Diagram of Timer Array Unit 0 for the 64-pin Products as an Example



Remark fIL: Low-speed on-chip oscillator clock frequency

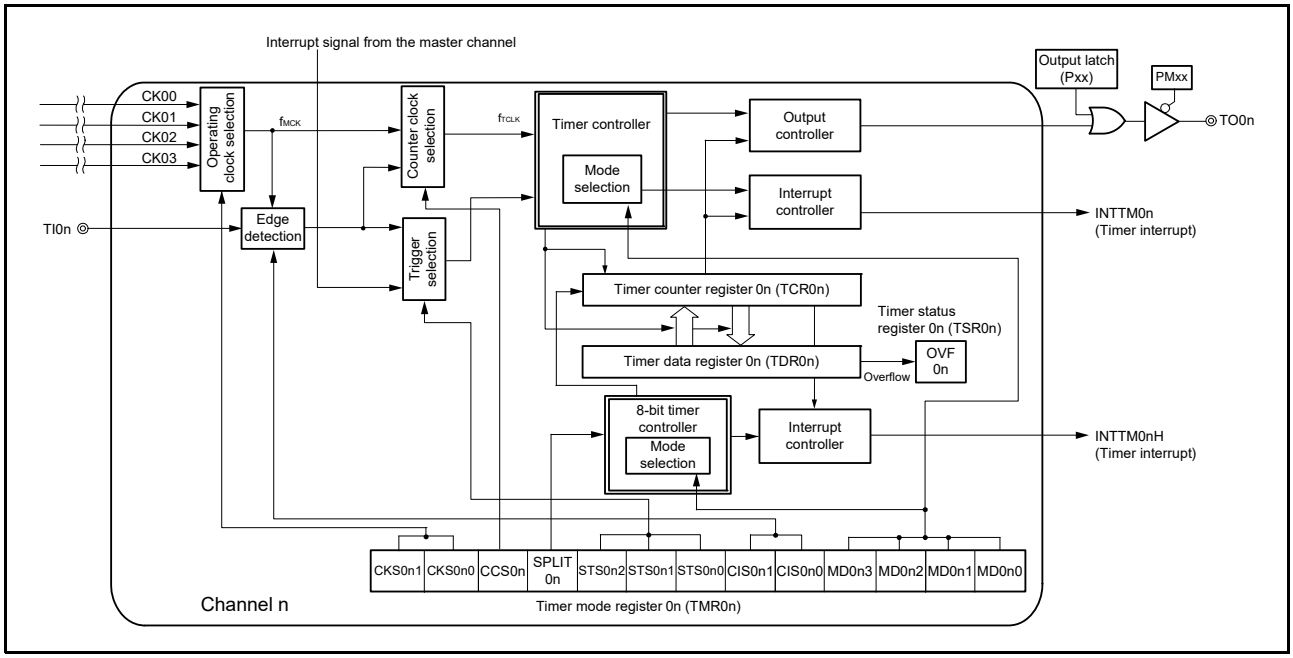
Figure 7 - 2 Internal Block Diagram of Channels 0, 2, 4, and 6 of Timer Array Unit 0



Note Only for channels 2, 4, and 6

Remark n = 0, 2, 4, 6

Figure 7 - 3 Internal Block Diagram of Channels 1 and 3 of Timer Array Unit 0



Remark n = 1, 3

Figure 7 - 4 Internal Block Diagram of Channel 5 of Timer Array Unit 0

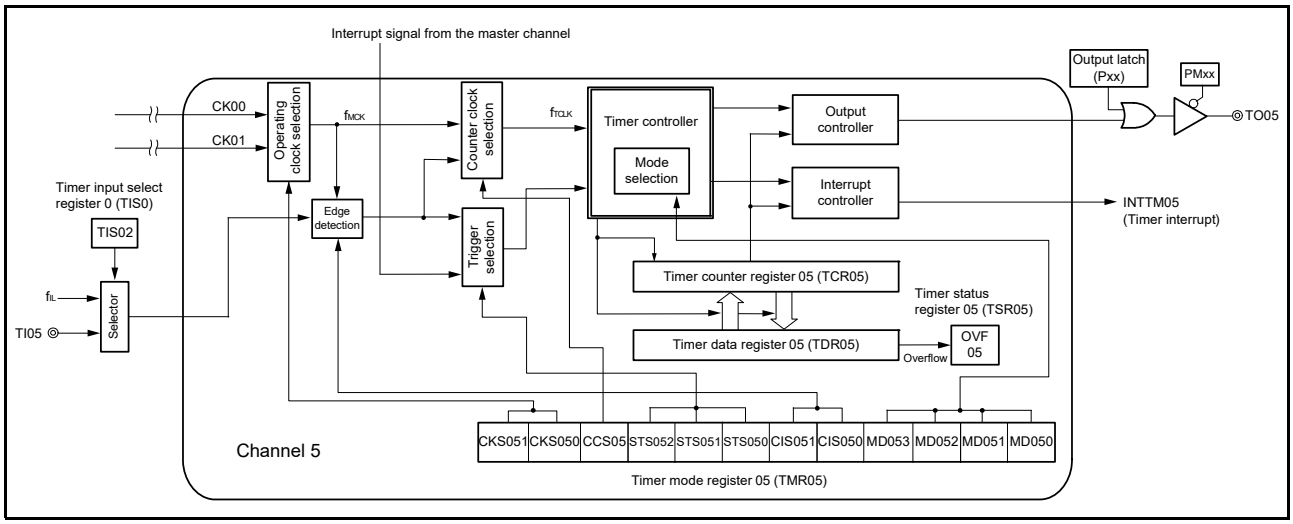
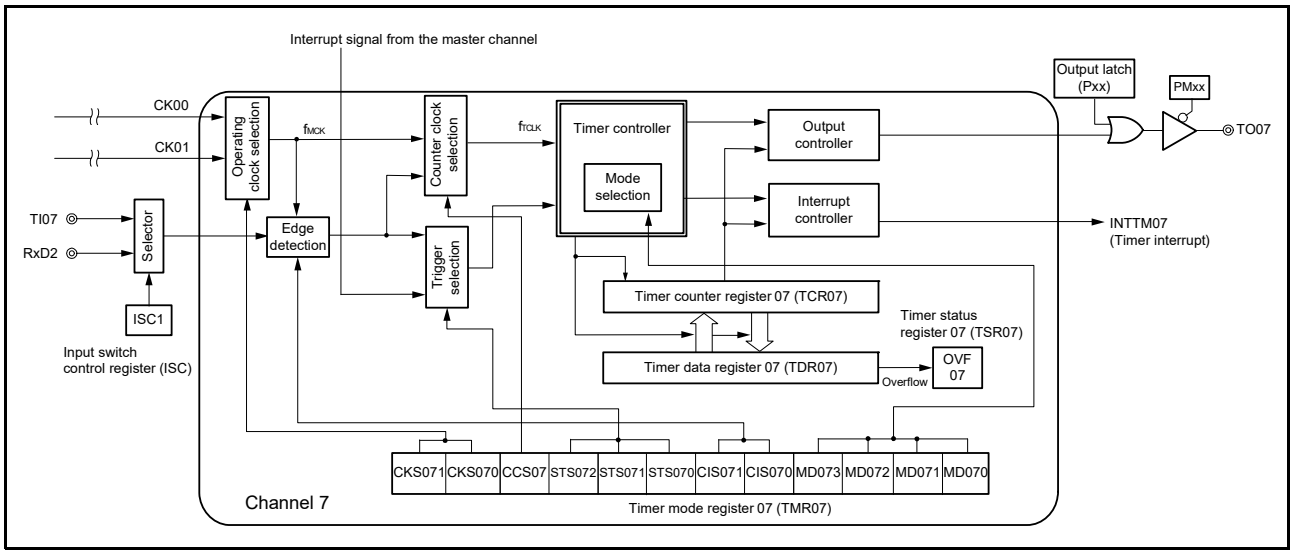


Figure 7 - 5 Internal Block Diagram of Channel 7 of Timer Array Unit 0

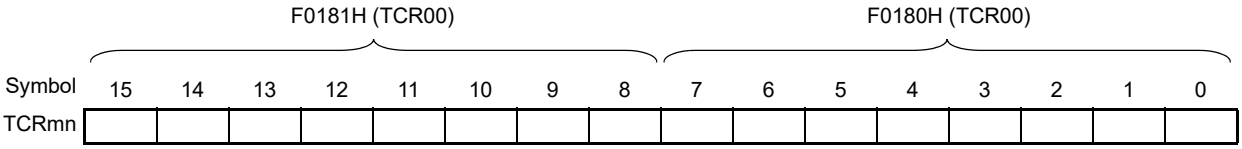


7.2.1 Timer counter register mn (TCRmn)

The TCRmn register is a 16-bit read-only register and is used to count the number of input clock cycles. The value of this counter is incremented or decremented in synchronization with the rising edge of each cycle of the counter clock. Whether the counter is incremented or decremented depends on the operation mode that is selected by the MDmn3 to MDmn0 bits of timer mode register mn (TMRmn) (see **7.3.4 Timer mode register mn (TMRmn)**).

Figure 7 - 6 Format of Timer Counter Register mn (TCRmn)

Address: F0180H, F0181H (TCR00) to F018EH, F018FH (TCR07)
After reset: FFFFH
R/W: R



Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

The count value can be read by reading timer counter register mn (TCRmn). The count value is set to FFFFH in the following cases.

- When the reset signal is generated
- When the TAUmRES bit of peripheral reset control register 0 (PRR0) is cleared
- When counting of the slave channel has been completed in the PWM output mode
- When counting of the slave channel has been completed in the delay counter mode
- When counting of the master/slave channel has been completed in the one-shot pulse output mode
- When counting of the slave channel has been completed in the multiple PWM output mode

The count value is cleared to 0000H in the following cases.

- When the start trigger is input in the capture mode
- When capturing has been completed in the capture mode

Caution The count value is not captured to timer data register mn (TDRmn) even when the TCRmn register is read.

The value read from the TCRmn register varies depending on the change to the operation mode and operating state as shown in the table below.

Table 7 - 3 Timer Counter Register mn (TCRmn) Read Value in Various Operation Modes

Operation Mode	Count Mode	Value Read from the Timer Counter Register mn (TCRmn)Note			
		Value when the operation mode is changed after releasing reset	Value when count operation is temporarily stopped (TTmn = 1)	Value when the operation mode is changed after count operation was temporarily stopped (TTmn = 1)	Value when waiting for a start trigger after one count
Interval timer mode	Countdown	FFFFH	Value when counting is stopped	Undefined	—
Capture mode	Count-up	0000H	Value when counting is stopped	Undefined	—
Event counter mode	Countdown	FFFFH	Value when counting is stopped	Undefined	—
One-count mode	Countdown	FFFFH	Value when counting is stopped	Undefined	FFFFH
Capture & one-count mode	Count-up	0000H	Value when counting is stopped	Undefined	Captured value of TDRmn register + 1

Note This indicates the value read from the TCRmn register when channel n has stopped operating as a timer (TEmn = 0) and has been enabled to operate as a counter (TSmn = 1). The read value is held in the TCRmn register until the count operation starts.

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

7.2.2 Timer data register mn (TDRmn)

This is a 16-bit register from which a capture function and a compare function can be selected. The capture or compare function can be switched by selecting an operation mode by using the MDmn3 to MDmn0 bits of timer mode register mn (TMRmn). The value of the TDRmn register can be changed at any time. This register can be read or written in 16-bit units. In addition, for the TDRm1 and TDRm3 registers, while in the 8-bit timer mode (when the SPLITm1, SPLITm3 bits of timer mode registers m1 and m3 (TMRm1, TMRm3) are 1), it is possible to read and write the data in 8-bit units, with TDRm1H and TDRm3H used as the higher 8 bits, and TDRm1L and TDRm3L used as the lower 8 bits. The value of this register following a reset is 0000H.

Figure 7 - 7 Format of Timer Data Register mn (TDRmn) (n = 0, 2, 4 to 7)

Address: FFF18H, FFF19H (TDR00), FFF64H, FFF65H (TDR02), FFF68H, FFF69H (TDR04) to FFF6EH, FFF6FH (TDR07)
After reset: 0000H
R/W: R/W

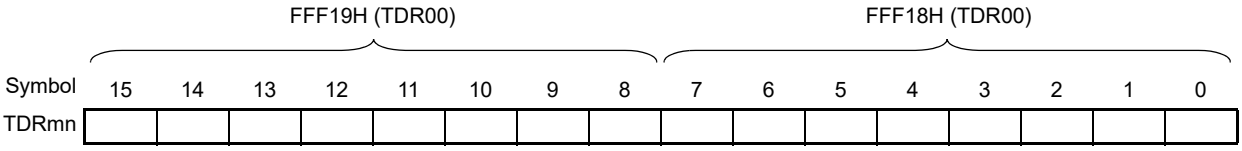
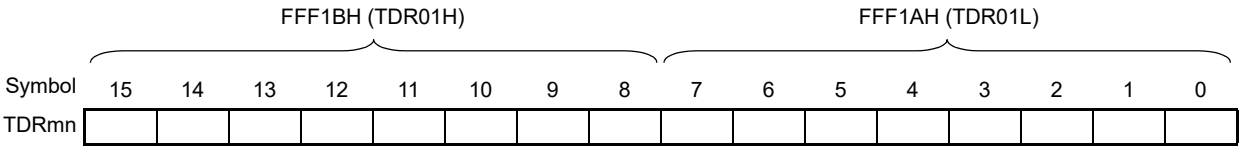


Figure 7 - 8 Format of Timer Data Register mn (TDRmn) (n = 1, 3)

Address: FFF1AH, FFF1BH (TDR01), FFF66H, FFF67H (TDR03)
After reset: 0000H
R/W: R/W



i) When timer data register mn (TDRmn) is used as compare register
Counting down is started from the value set to the TDRmn register. When the count value reaches 0000H, an interrupt signal (INTTMmn) is generated. The TDRmn register holds its value until it is rewritten.

ii) When timer data register mn (TDRmn) is used as capture register
The count value of timer counter register mn (TCRmn) is captured to the TDRmn register when the capture trigger is input. A valid edge of the TImn pin can be selected as the capture trigger. This selection is made by timer mode register mn (TMRmn).

Caution **The TDRmn register does not perform a capture operation even if a capture trigger is input, when it is set to the compare function.**

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

7.3 Registers for Controlling the Timer Array Unit

The following registers are used to control the timer array unit.

- Peripheral enable register 0 (PER0)
- Peripheral reset control register 0 (PRR0)
- Timer clock select register m (TPSm)
- Timer mode register mn (TMRmn)
- Timer status register mn (TSRmn)
- Timer channel enable status register m (TEm)
- Timer channel start register m (TSm)
- Timer channel stop register m (TTm)
- Timer input select register 0 (TIS0)
- Timer output enable register m (TOEm)
- Timer output register m (TOm)
- Timer output level register m (TOLm)
- Timer output mode register m (TOMm)
- Input switch control register (ISC)
- Noise filter enable register 1 (NFEN1)
- Port mode registers (PMxx)
- Port registers (Pxx)
- Port output mode registers (POMxx)
- Port mode control A registers (PMCAxx)

Caution **Which registers and bits are included depends on the product. Be sure to set bits that are not mounted to their initial values.**

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

7.3.1 Peripheral enable register 0 (PER0)

The PER0 register is used to enable or disable the supply of a clock signal to various on-chip peripheral modules. Clock supply to an on-chip peripheral module that is not to be used can be stopped to decrease power consumption and noise. If timer array unit 0 is to be used, be sure to set the TAU0EN bit of this register to 1. The PER0 register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 7 - 9 Format of Peripheral Enable Register 0 (PER0)

Address: F00F0H
After reset: 00H
R/W: R/W

Symbol	7	6	<5>	<4>	<3>	<2>	1	<0>
PER0	0	0	ADCEN	IICA0EN Note 1	SAU1EN	SAU0EN	0	TAU0EN

TAU0EN	Control of supply of an input clock to timer array unit 0
0	Stops supply of an input clock. • The SFRs used by timer array unit 0 cannot be written.Note 2
1	Enables supply of an input clock. • The SFRs used by timer array unit 0 can be read and written.

Note 1. This bit is only present in the 32- to 64-pin products.

Note 2. When an SFR used by a timer array unit is read, the value returned is 00H or 0000H regardless of the setting value.

Caution 1. Be sure to clear bits 7, 6, and 1 to 0.

Caution 2. Be sure to clear bit 4 in the 20-pin products to 0.

Caution 3. When setting a timer array unit, start by setting the following registers while the setting of the TAUmEN bit is 1.

- **Timer clock select register m (TPSm)**
- **Timer mode register mn (TMRmn)**
- **Timer status register mn (TSRmn)**
- **Timer channel enable status register m (TEm)**
- **Timer channel start register m (TSm)**
- **Timer channel stop register m (TTm)**
- **Timer output enable register m (TOEm)**
- **Timer output register m (TOm)**
- **Timer output level register m (TOLm)**
- **Timer output mode register m (TOMm)**

If the setting of the TAUmEN bit is 0, each register which controls the timer array unit is restored to 00H and writing to any of those registers is ignored. Note, however, writing to the following registers is valid.

- **Timer input select register 0 (TIS0)**
- **Input switch control register (ISC)**
- **Noise filter enable register 1 (NFEN1)**
- **Port mode control A registers (PMCA0, PMCA1)**
- **Port mode registers (PM0, PM1, PM3, PM4)**
- **Port registers (P0, P1, P3, P4)**

7.3.2 Peripheral reset control register 0 (PRR0)

The PRR0 register is used to control resetting of the on-chip peripheral modules. Each bit in this register controls resetting and release from the reset state of the corresponding on-chip peripheral module. Set the TAU0RES bit of this register to 1 to place timer array unit 0 in the reset state. The PRR0 register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 7 - 10 Format of Peripheral Reset Control Register 0 (PRR0)

Address: F00F1H
After reset: 00H
R/W: R/W

Symbol	7	6	<5>	<4>	<3>	<2>	1	<0>
PRR0	0	0	ADCRES	IICA0RES **Note**	SAU1RES	SAU0RES	0	TAU0RES

TAU0RES	Control resetting of timer array unit 0
0	Timer array unit 0 is released from the reset state.
1	Timer array unit 0 is in the reset state. • The SFRs for use with timer array unit 0 are initialized.

Note This bit is only present in the 32- to 64-pin products.

Caution 1. Be sure to clear bits 7, 6, and 1 to 0.
Caution 2. Be sure to clear bit 4 in the 20-pin products to 0.

7.3.3 Timer clock select register m (TPSm)

The TPSm register is a 16-bit register used to select two types or four types of operating clocks (CKm0, CKm1, CKm2, CKm3) that are commonly supplied to each channel. CKm0 is selected by using bits 3 to 0 of the TPSm register, and CKm1 is selected by using bits 7 to 4 of the TPSm register. In addition, only for channels 1 and 3, CKm2 and CKm3 can be also selected. CKm2 is selected by using bits 9 and 8 of the TPSm register, and CKm3 is selected by using bits 13 and 12 of the TPSm register. Rewriting of the TPSm register during timer operation is possible only in the following cases.

If the PRSm00 to PRSm03 bits can be rewritten (n = 0 to 7):
All channels for which CKm0 is selected as the operating clock (CKSmn1, CKSmn0 = 00B) are stopped (TEmn = 0).
If the PRSm10 to PRSm13 bits can be rewritten (n = 0 to 7):
All channels for which CKm1 is selected as the operating clock (CKSmn1, CKSmn0 = 10B) are stopped (TEmn = 0).
If the PRSm20 and PRSm21 bits can be rewritten (n = 1, 3):
All channels for which CKm2 is selected as the operating clock (CKSmn1, CKSmn0 = 01B) are stopped (TEmn = 0).
If the PRSm30 and PRSm31 bits can be rewritten (n = 1, 3):
All channels for which CKm3 is selected as the operating clock (CKSmn1, CKSmn0 = 11B) are stopped (TEmn = 0).

The TPSm register can be set by a 16-bit memory manipulation instruction. The value of this register following a reset is 0000H.

Figure 7 - 11 Format of Timer Clock Select Register m (TPSm) (1/2)

Address: F01B6H, F01B7H (TPS0)
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
TPSm	0	0	PRSm31	PRSm30	0	0	PRSm21	PRSm20

7	6	5	4	3	2	1	0
PRSm13	PRSm12	PRSm11	PRSm10	PRSm03	PRSm02	PRSm01	PRSm00

PRS mk3	PRS mk2	PRS mk1	PRS mk0	Selection of operating clock (CKmk)Note (k = 0, 1)				
					f_{CLK} = 2 MHz	f_{CLK} = 5 MHz	f_{CLK} = 10 MHz	f_{CLK} = 20 MHz	f_{CLK} = 32 MHz
0	0	0	0	f_{CLK}	2 MHz	5 MHz	10 MHz	20 MHz	32 MHz
0	0	0	1	$f_{CLK}/2$	1 MHz	2.5 MHz	5 MHz	10 MHz	16 MHz
0	0	1	0	$f_{CLK}/2^2$	500 kHz	1.25 MHz	2.5 MHz	5 MHz	8 MHz
0	0	1	1	$f_{CLK}/2^3$	250 kHz	625 kHz	1.25 MHz	2.5 MHz	4 MHz
0	1	0	0	$f_{CLK}/2^4$	125 kHz	313 kHz	625 kHz	1.25 MHz	2 MHz
0	1	0	1	$f_{CLK}/2^5$	62.5 kHz	156 kHz	313 kHz	625 kHz	1 MHz
0	1	1	0	$f_{CLK}/2^6$	31.3 kHz	78.1 kHz	156 kHz	313 kHz	500 kHz
0	1	1	1	$f_{CLK}/2^7$	15.6 kHz	39.1 kHz	78.1 kHz	156 kHz	250 kHz
1	0	0	0	$f_{CLK}/2^8$	7.81 kHz	19.5 kHz	39.1 kHz	78.1 kHz	125 kHz
1	0	0	1	$f_{CLK}/2^9$	3.91 kHz	9.77 kHz	19.5 kHz	39.1 kHz	62.5 kHz
1	0	1	0	$f_{CLK}/2^{10}$	1.95 kHz	4.88 kHz	9.77 kHz	19.5 kHz	31.3 kHz
1	0	1	1	$f_{CLK}/2^{11}$	977 Hz	2.44 kHz	4.88 kHz	9.77 kHz	15.6 kHz
1	1	0	0	$f_{CLK}/2^{12}$	488 Hz	1.22 kHz	2.44 kHz	4.88 kHz	7.81 kHz
1	1	0	1	$f_{CLK}/2^{13}$	244 Hz	610 Hz	1.22 kHz	2.44 kHz	3.91 kHz
1	1	1	0	$f_{CLK}/2^{14}$	122 Hz	305 Hz	610 Hz	1.22 kHz	1.95 kHz
1	1	1	1	$f_{CLK}/2^{15}$	61.0 Hz	153 Hz	305 Hz	610 Hz	977 Hz

Note When changing the clock selected for f_{CLK} (by changing the system clock control register (CKC) value), stop timer array unit (TTm = 00FFH). The timer array unit must also be stopped if the operating clock (f_{MCK}) or the valid edge of the signal input from the TImn pin is selected.

Caution 1. Be sure to clear bits 15, 14, 11, 10 to 0.

Caution 2. If f_{CLK} (no division) is selected as the operating clock (CKmk) and TDRmn is set to 0000H (m = 0, n = 0 to 7), interrupt requests output from timer array units cannot be used.

Remark 1. f_{CLK}: CPU/peripheral hardware clock frequency

Remark 2. Waveform of the clock to be selected in the TPSm register which becomes high level for one period of f_{CLK} from its rising edge (m = 0). For details, see **7.5.1 Counter clock (f_{TCLK})**.

Figure 7 - 11 Format of Timer Clock Select Register m (TPSm) (2/2)

PRS m21	PRS m20	Selection of operating clock (CKm2)Note					
			f_{CLK} = 2 MHz	f_{CLK} = 5 MHz	f_{CLK} = 10 MHz	f_{CLK} = 20 MHz	f_{CLK} = 32 MHz
0	0	$f_{CLK}/2$	1 MHz	2.5 MHz	5 MHz	10 MHz	16 MHz
0	1	$f_{CLK}/2^2$	500 kHz	1.25 MHz	2.5 MHz	5 MHz	8 MHz
1	0	$f_{CLK}/2^4$	125 kHz	313 kHz	625 kHz	1.25 MHz	2 MHz
1	1	$f_{CLK}/2^6$	31.3 kHz	78.1 kHz	156 kHz	313 kHz	500 kHz

PRS m31	PRS m30	Selection of operating clock (CKm3)Note					
			f_{CLK} = 2 MHz	f_{CLK} = 5 MHz	f_{CLK} = 10 MHz	f_{CLK} = 20 MHz	f_{CLK} = 32 MHz
0	0	$f_{CLK}/2^8$	7.81 kHz	19.5 kHz	39.1 kHz	78.1 kHz	125 kHz
0	1	$f_{CLK}/2^{10}$	1.95 kHz	4.88 kHz	9.77 kHz	19.5 kHz	31.3 kHz
1	0	$f_{CLK}/2^{12}$	488 Hz	1.22 kHz	2.44 kHz	4.88 kHz	7.81 kHz
1	1	$f_{CLK}/2^{14}$	122 Hz	305 Hz	610 Hz	1.22 kHz	1.95 kHz

Note When changing the clock selected for f_{CLK} (by changing the system clock control register (CKC) value), stop timer array unit (TTm = 00FFH). The timer array unit must also be stopped if the operating clock (f_{MCK}) or the valid edge of the signal input from the TImn pin is selected.

Caution Be sure to clear bits 15, 14, 11, 10 to 0.

By using channels 1 and 3 in the 8-bit timer mode and specifying CKm2 or CKm3 as the operating clock, the interval times shown in **Table 7 - 4** can be achieved by using the interval timer function.

Table 7 - 4 Interval Times Available for Operating Clock CKSm2 or CKSm3

Clock		Interval time[Note] (f_{CLK} = 32 MHz)			
		10 μs	100 μs	1 ms	10 ms
CKm2	$f_{CLK}/2$	✓	—	—	—
	$f_{CLK}/2^2$	✓	—	—	—
	$f_{CLK}/2^4$	✓	✓	—	—
	$f_{CLK}/2^6$	✓	✓	—	—
CKm3	$f_{CLK}/2^8$	—	✓	✓	—
	$f_{CLK}/2^{10}$	—	✓	✓	—
	$f_{CLK}/2^{12}$	—	—	✓	✓
	$f_{CLK}/2^{14}$	—	—	✓	✓

Note The margin is within 5%.

Remark 1. f_{CLK}: CPU/peripheral hardware clock frequency

Remark 2. For details of a signal of $f_{CLK}/2^j$ (j = 2 to 15) selected with the TPSm register, see **7.5.1 Counter clock (f_{TCLK})**.

7.3.4 Timer mode register mn (TMRmn)

The TMRmn register sets an operation mode of channel n. This register is used to select the operating clock (f_{MCK}), select the counter clock, select the master/slave, select the 16 or 8-bit timer (only for channels 1 and 3), specify the start trigger and capture trigger, select the valid edge of the timer input, and specify the operation mode (interval, capture, event counter, one-count, or capture and one-count). Rewriting the TMRmn register is prohibited when the register is in operation (when TEmn = 1). However, bits 7 and 6 (CISmn1, CISmn0) can be rewritten even while the register is operating with some functions (when TEmn = 1). For details, see **7.8 Independent Channel Operation Function of Timer Array Unit** and **7.9 Simultaneous Channel Operation Function of Timer Array Unit**. The TMRmn register can be set by a 16-bit memory manipulation instruction. The value of this register following a reset is 0000H.

Caution The bit function assigned to bit 11 of the TMRmn register depends on the channel.
TMRm2, TMRm4, TMRm6: MASTERmn bit (n = 2, 4, 6)
TMRm1, TMRm3: SPLITmn bit (n = 1, 3)
TMRm0, TMRm5, TMRm7: Fixed to 0

Figure 7 - 12 Format of Timer Mode Register mn (TMRmn) (1/3)

Address: F0190H, F0191H (TMR00) to F019EH, F019FH (TMR07)
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
TMRmn (n = 2, 4, 6)	CKSmn1	CKSmn0	0	CCSmn	MASTERmn	STSmn2	STSmn1	STSmn0

7	6	5	4	3	2	1	0
CISmn1	CISmn0	0	0	MDmn3	MDmn2	MDmn1	MDmn0

Symbol	15	14	13	12	11	10	9	8
TMRmn (n = 1, 3)	CKSmn1	CKSmn0	0	CCSmn	SPLITmn	STSmn2	STSmn1	STSmn0

7	6	5	4	3	2	1	0
CISmn1	CISmn0	0	0	MDmn3	MDmn2	MDmn1	MDmn0

Symbol	15	14	13	12	11	10	9	8
TMRmn (n = 0, 5, 7)	CKSmn1	CKSmn0	0	CCSmn	0 Note 1	STSmn2	STSmn1	STSmn0

7	6	5	4	3	2	1	0
CISmn1	CISmn0	0	0	MDmn3	MDmn2	MDmn1	MDmn0

CKSmn1	CKSmn0	Selection of operating clock (f_{MCK}) of channel n
0	0	Operating clock CKm0 set by timer clock select register m (TPSm)
0	1	Operating clock CKm2 set by timer clock select register m (TPSm)
1	0	Operating clock CKm1 set by timer clock select register m (TPSm)
1	1	Operating clock CKm3 set by timer clock select register m (TPSm)
Operating clock (f_{MCK}) is used by the edge detector. A counter clock (f_{TCLK}) and a sampling clock are generated depending on the setting of the CCSmn bit. The operating clocks CKm2 and CKm3 can only be selected for channels 1 and 3.		

CCSmn	Selection of counter clock (f_{TCLK}) of channel n
0	Operating clock (f_{MCK}) specified by the CKSmn1 and CKSmn0 bits
1	Valid edge of input signal input from the TImn pin In channel 5, valid edge of input signal selected by TIS0 In channel 7, valid edge of input signal selected by ISC
Counter clock (f_{TCLK}) is used for the counter, output controller, and interrupt controller.	

Figure 7 - 12 Format of Timer Mode Register mn (TMRmn) (2/3)

Bit 11 of TMRmn (n = 2, 4, 6)

MASTERmn	Selection between using channel n independently or simultaneously with another channel (as a slave or master)
0	Operates in independent channel operation function or as slave channel in simultaneous channel operation function.
1	Operates as master channel in simultaneous channel operation function.
Only the channel 2, 4, 6 can be set as a master channel (MASTERmn = 1). Be sure to use channel 0, 5, 7 are fixed to 0 (Regardless of the bit setting, channel 0 operates as master, because it is the highest channel). Clear the MASTERmn bit to 0 for a channel that is used with the independent channel operation function.	

Bit 11 of TMRmn (n = 1, 3)

SPLITmn	Selection of 8 or 16-bit timer operation for channels 1 and 3
0	Operates as 16-bit timer. (Operates in independent channel operation function or as slave channel in simultaneous channel operation function.)
1	Operates as 8-bit timer.

STSmn2	STSmn1	STSmn0	Setting of start trigger or capture trigger of channel n
0	0	0	Only software trigger start is valid (other trigger sources are unselected).
0	0	1	Valid edge of the TImn pin input is used as both the start trigger and capture trigger.
0	1	0	Both the edges of the TImn pin input are used as a start trigger and a capture trigger.
1	0	0	Interrupt signal of the master channel is used (when the channel is used as a slave channel with the simultaneous channel operation function).
Other than above			Setting prohibited

CISmn1	CISmn0	Selection of TImn pin input valid edge
0	0	Falling edge
0	1	Rising edge
1	0	Both edges (when low-level width is measured) Start trigger: Falling edge, Capture trigger: Rising edge
1	1	Both edges (when high-level width is measured) Start trigger: Rising edge, Capture trigger: Falling edge
If both the edges are specified when the value of STSmn2 to STSmn0 is other than 010B, set CISmn1 and CISmn0 to 10B.		

Figure 7 - 12 Format of Timer Mode Register mn (TMRmn) (3/3)

MD mn3	MD mn2	MD mn1	Operation mode of channel n	Corresponding function	Count operation of TCR
0	0	0	Interval timer mode	Interval timer/Square wave output/Divider function/PWM output (master)	Counting down
0	1	0	Capture mode	Input pulse interval measurement	Counting up
0	1	1	Event counter mode	External event counter	Counting down
1	0	0	One-count mode	Delay counter/One-shot pulse output/PWM output (slave)	Counting down
1	1	0	Capture & one-count mode	Measurement of low- or high-level width of input signal	Counting up
Other than above			Setting prohibited		
The operation in each mode varies depending on the MDmn0 bit. See the table below for details.					

Operation mode set by the MDmn3 to MDmn1 bits (see the table above)	MD mn0	Setting of starting counting and interrupt
• Interval timer mode (000B) • Capture mode (010B)	0	Timer interrupt is not generated when counting is started (timer output does not change, either).
	1	Timer interrupt is generated when counting is started (timer output also changes).
• Event counter mode (011B)	0	Timer interrupt is not generated when counting is started (timer output does not change, either).
• One-count mode[Note 2] (100B)	0	Start trigger is invalid during counting operation. At that time, interrupt is not generated.
	1	Start trigger is valid during counting operation[Note 3]. At that time, interrupt is not generated.
• Capture & one-count mode (110B)	0	Timer interrupt is not generated when counting is started (timer output does not change, either). Start trigger is invalid during counting operation. At that time interrupt is not generated.
Other than above		Setting prohibited

Note 1. Bit 11 is a read-only bit and fixed to 0. Writing to this bit is ignored.

Note 2. In one-count mode, interrupt output (INTTMmn) when starting a count operation and TOmn output are not controlled.

Note 3. If the start trigger (TSmn = 1) is issued during operation, the counter is initialized, and recounting is started (does not occur the interrupt request).

Caution 1. Be sure to clear bits 13, 5, and 4 to 0.

Caution 2. The timer array unit must be stopped (TTm = 00FFH) if the clock selected for f_{CLK} is changed (by changing the value of the system clock control register (CKC)), even if the operating clock specified by using the CKSmn1 and CKSmn0 bits (f_{MCK}) or the valid edge of the signal input from the TImn pin is selected as the counter clock (f_{TCLK}).

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

7.3.5 Timer status register mn (TSRmn)

The TSRmn register indicates the overflow state of the counter of channel n. The TSRmn register is valid only in the capture mode (MDmn3 to MDmn1 = 010B) and capture & one-count mode (MDmn3 to MDmn1 = 110B). See **Table 7 - 5** for the operation of the OVF bit in each operation mode and set/clear conditions. The TSRmn register can be read by a 16-bit memory manipulation instruction. The 8 lower-order bits of a TSRmn register can be handled as the TSRmnL register, which can be read by an 8-bit memory manipulation instruction. The value of this register following a reset is 0000H.

Figure 7 - 13 Format of Timer Status Register mn (TSRmn)

Address: F01A0H, F01A1H (TSR00) to F01AEH, F01AFH (TSR07)
After reset: 0000H
R/W: R

Symbol	15	14	13	12	11	10	9	8
TSRmn	0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	OVF

OVF	Counter overflow state of channel n
0	Overflow does not occur.
1	Overflow occurs.
When OVF = 1, this flag is cleared (OVF = 0) when the next value is captured without overflow.	

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

Table 7 - 5 OVF Bit Operation and Set/Clear Conditions in Each Operation Mode

Timer operation mode	OVF bit	Set/clear conditions
• Capture mode • Capture & one-count mode	clear	When no overflow has occurred upon capturing
	set	When an overflow has occurred upon capturing
• Interval timer mode • Event counter mode • One-count mode	clear	— (Use prohibited)
	set	

Remark The OVF bit does not change immediately after the counter has overflowed, but changes upon the subsequent capture.

7.3.6 Timer channel enable status register m (TEm)

The TEm register is used to enable or stop the timer operation of each channel. Each bit of the TEm register corresponds to each bit of the timer channel start register m (TSm) and the timer channel stop register m (TTm). When a bit of the TSm register is set to 1, the corresponding bit of this register is set to 1. When a bit of the TTm register is set to 1, the corresponding bit of this register is cleared to 0. The TEm register can be read by a 16-bit memory manipulation instruction. The 8 lower-order bits of a TEm register can be handled as the TEmL register, which can be read by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 0000H.

Figure 7 - 14 Format of Timer Channel Enable Status Register m (TEm)

Address: F01B0H, F01B1H (TE0)
After reset: 0000H
R/W: R

Symbol	15	14	13	12	11	10	9	8
TEm	0	0	0	0	TEHm3	0	TEHm1	0

7	6	5	4	3	2	1	0
TEm7	TEm6	TEm5	TEm4	TEm3	TEm2	TEm1	TEm0

TEHm3	Indication of whether operation of the higher 8-bit timer is enabled or stopped when channel 3 is in the 8-bit timer mode
0	Operation is stopped.
1	Operation is enabled.

TEHm1	Indication of whether operation of the higher 8-bit timer is enabled or stopped when channel 1 is in the 8-bit timer mode
0	Operation is stopped.
1	Operation is enabled.

TEmn	Indication of operation enabled or stopped state of channel n
0	Operation is stopped.
1	Operation is enabled.
This bit displays whether operation of the lower 8-bit timer for TEm1 and TEm3 is enabled or stopped when channel 1 or 3 is in the 8-bit timer mode.	

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

7.3.7 Timer channel start register m (TSm)

The TSm register is a trigger register that is used to initialize timer counter register mn (TCRmn) and start the counting operation of each channel. When a bit of this register is set to 1, the corresponding bit of timer channel enable status register m (TEm) is set to 1. The TSmn, TSHm1, TSHm3 bits are immediately cleared when operation is enabled (TEmn, TEHm1, TEHm3 = 1), because they are trigger bits. The TSm register can be set by a 16-bit memory manipulation instruction. The 8 lower-order bits of a TSm register can be handled as the TSmL register, which can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 0000H.

Figure 7 - 15 Format of Timer Channel Start Register m (TSm)

Address: F01B2H, F01B3H (TS0)
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
TSm	0	0	0	0	TSHm3	0	TSHm1	0

7	6	5	4	3	2	1	0
TSm7	TSm6	TSm5	TSm4	TSm3	TSm2	TSm1	TSm0

TSHm3	Trigger to enable operation (start operation) of the higher 8-bit timer when channel 3 is in the 8-bit timer mode
0	No trigger operation
1	The TEHm3 bit is set to 1 and the count operation becomes enabled. The TCRm3 register count operation start in the interval timer mode in the count operation enabled state (see **Table 7 - 6** in **7.5.2 Timing of the start of counting**).

TSHm1	Trigger to enable operation (start operation) of the higher 8-bit timer when channel 1 is in the 8-bit timer mode
0	No trigger operation
1	The TEHm1 bit is set to 1 and the count operation becomes enabled. The TCRm1 register count operation start in the interval timer mode in the count operation enabled state (see **Table 7 - 6** in **7.5.2 Timing of the start of counting**).

TSmn	Operation enable (start) trigger of channel n
0	No trigger operation
1	The TEmn bit is set to 1 and the count operation becomes enabled. The TCRmn register count operation start in the count operation enabled state varies depending on each operation mode (see **Table 7 - 6** in **7.5.2 Timing of the start of counting**). This bit is the trigger to enable operation (start operation) of the lower 8-bit timer for TSm1 and TSm3 when channel 1 or 3 is in the 8-bit timer mode.

Caution 1. Be sure to clear bits 15 to 12, 10, and 8 to 0.

Caution 2. When switching from a function that does not use TImn pin input to one that does, the following wait period is required from when timer mode register mn (TMRmn) is set until the TSmn (TSHm1, TSHm3) bit is set to 1.
When the TImn pin noise filter is enabled (TNFENmn = 1): Four cycles of the operating clock (f_{MCK})
When the TImn pin noise filter is disabled (TNFENmn = 0): Two cycles of the operating clock (f_{MCK})

Remark 1. When the TSm register is read, 0 is always read.

Remark 2. m: Unit number (m = 0), n: Channel number (n = 0 to 7)

7.3.8 Timer channel stop register m (TTm)

The TTm register is a trigger register that is used to stop the counting operation of each channel. When a bit of this register is set to 1, the corresponding bit of timer channel enable status register m (TEm) is cleared to 0. The TTmn, TTHm1, TTHm3 bits are immediately cleared when operation is stopped (TEmn, TEHm1, TEHm3 = 0), because they are trigger bits. The TTm register can be set by a 16-bit memory manipulation instruction. The 8 lower-order bits of a TTm register can be handled as the TTmL register, which can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 0000H.

Figure 7 - 16 Format of Timer Channel Stop Register m (TTm)

Address: F01B4H, F01B5H (TT0)
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
TTm	0	0	0	0	TTHm3	0	TTHm1	0

7	6	5	4	3	2	1	0
TTm7	TTm6	TTm5	TTm4	TTm3	TTm2	TTm1	TTm0

TTHm3	Trigger to stop operation of the higher 8-bit timer when channel 3 is in the 8-bit timer mode
0	No trigger operation
1	TEHm3 bit is cleared to 0 and the count operation is stopped.

TTHm1	Trigger to stop operation of the higher 8-bit timer when channel 1 is in the 8-bit timer mode
0	No trigger operation
1	TEHm1 bit is cleared to 0 and the count operation is stopped.

TTmn	Operation stop trigger of channel n
0	No trigger operation
1	TEmn bit is cleared to 0 and the count operation is stopped. This bit is the trigger to stop operation of the lower 8-bit timer for TTm1 and TTm3 when channel 1 or 3 is in the 8-bit timer mode.

Caution Be sure to clear bits 15 to 12, 10, and 8 to 0.

Remark 1. When the TTm register is read, 0 is always read.
Remark 2. m: Unit number (m = 0), n: Channel number (n = 0 to 7)

7.3.9 Timer input select register 0 (TIS0)

The TIS0 register is used to select the timer input of channel 5. The TIS0 register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 7 - 17 Format of Timer Input Select Register 0 (TIS0)

Address: F0074H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
TIS0	0	0	0	0	0	TIS02	0	0

TIS02	Selection of timer input used with channel 5
0	Input signal of timer input pin (TI05)
1	Low-speed on-chip oscillator clock (f_{IL})

Caution Make sure that both the low-level and high-level widths of timer input to be selected are no less than $1/f_{MCK}$ + 10 ns.

7.3.10 Timer output enable register m (TOEm)

The TOEm register is used to enable or disable timer output of each channel. Channel n for which timer output has been enabled becomes unable to rewrite the value of the TOmn bit of timer output register m (TOm) described later by software, and the value reflecting the setting of the timer output function through the count operation is output from the timer output pin (TOmn). The TOEm register can be set by a 16-bit memory manipulation instruction. The 8 lower-order bits of a TOEm register can be handled as the TOEmL register, which can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 0000H.

Figure 7 - 18 Format of Timer Output Enable Register m (TOEm)

Address: F01BAH, F01BBH (TOE0)
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
TOEm	0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
TOEm7	TOEm6	TOEm5	TOEm4	TOEm3	TOEm2	TOEm1	TOEm0

TOEmn	Enabling/disabling timer output for channel n
0	Disables timer output. The corresponding TOmn bit does not reflect timer operation with this setting, so the output level of a TOmn bit is fixed to the level written to the TOm register. Writing to the TOmn bit is enabled and the level set in the TOmn bit is output from the TOmn pin.
1	Enables timer output. The corresponding TOmn bit reflects timer operation with this setting, so the output waveform is generated. Writing to the TOmn bit is ignored.

Caution **Be sure to clear bits 15 to 8 to 0.**

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

7.3.11 Timer output register m (TOm)

The TOm register is a buffer register of timer output of each channel. The value of each bit in this register is output from the timer output pin (TOmn) of each channel. The TOmn bit on this register can be rewritten by software only when timer output is disabled (TOEmn = 0). When timer output is enabled (TOEmn = 1), rewriting this register by software is ignored, and the value is changed only by the timer operation. To use the P00/TI00, P01/TO00, P16/TI01/TO01, P17/TI02/TO02, P31/TI03/TO03, P42/TI04/TO04, P05/TI05/TO05, P06/TI06/TO06, or P41/TI07/TO07 pin as a port function pin, set the corresponding TOmn bit to 0. The TOm register can be set by a 16-bit memory manipulation instruction. The 8 lower-order bits of a TOm register can be handled as the TOmL register, which can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 0000H.

Figure 7 - 19 Format of Timer Output Register m (TOm)

Address: F01B8H, F01B9H (TO0)
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
TOm	0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
TOm7	TOm6	TOm5	TOm4	TOm3	TOm2	TOm1	TOm0

TOmn	Timer output of channel n
0	Timer output value is 0.
1	Timer output value is 1.

Caution **Be sure to clear bits 15 to 8 to 0.**

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

7.3.12 Timer output level register m (TOLm)

The TOLm register controls the timer output level of each channel. The setting of the inverted output of channel n by this register is reflected at the timing of set or reset of the timer output signal while the timer output is enabled (TOEmn = 1) in the Slave channel output mode (TOMmn = 1). In the master channel output mode (TOMmn = 0), this register setting is invalid. The TOLm register can be set by a 16-bit memory manipulation instruction. The 8 lower-order bits of a TOLm register can be handled as the TOLmL register, which can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 0000H.

Figure 7 - 20 Format of Timer Output Level Register m (TOLm)

Address: F01BCH, F01BDH (TOL0)
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
TOLm	0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
TOLm7	TOLm6	TOLm5	TOLm4	TOLm3	TOLm2	TOLm1	0

TOLmn	Control of timer output level of channel n
0	Positive logic output (active-high)
1	Negative logic output (active-low)

Caution Be sure to clear bits 15 to 8, and 0 to 0.

Remark 1. If the value of this register is rewritten during timer operation, the timer output logic is inverted when the timer output signal changes next, instead of immediately after the register value is rewritten.

Remark 2. m: Unit number (m = 0), n: Channel number (n = 0 to 7)

7.3.13 Timer output mode register m (TOMm)

The TOMm register is used to control the timer output mode of each channel. When a channel is used for the independent channel operation function, set the corresponding bit of the channel to be used to 0. When a channel is used for the simultaneous channel operation function (PWM output, one-shot pulse output, or multiple PWM output), set the corresponding bit of the master channel to 0 and the corresponding bit of the slave channel to 1. The setting of each channel n by this register is reflected at the timing when the timer output signal is set or reset while the timer output is enabled (TOEmn = 1). The TOMm register can be set by a 16-bit memory manipulation instruction. The 8 lower-order bits of a TOMm register can be handled as the TOMmL register, which can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 0000H.

Figure 7 - 21 Format of Timer Output Mode Register m (TOMm)

Address: F01BEH, F01BFH (TOM0)
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
TOMm	0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
TOMm7	TOMm6	TOMm5	TOMm4	TOMm3	TOMm2	TOMm1	0

TOMmn	Control of timer output mode of channel n
0	Master channel output mode (to produce toggled output by timer interrupt request signal (INTTMmn))
1	Slave channel output mode (output is set by the timer interrupt request signal (INTTMmn) of the master channel, and reset by the timer interrupt request signal (INTTM0p) of the slave channel)

Caution **Be sure to clear bits 15 to 8, and 0 to 0.**

Remark m: Unit number (m = 0)
n: Channel number
n = 0 to 7 (n = 0, 2, 4, 6 for master channel)
p: Slave channel number
n < p ≤ 7
(For details of the relation between the master channel and slave channel, see **7.4.1 Basic specifications of the simultaneous channel operation function**.)

7.3.14 Input switch control register (ISC)

The ISC1 and ISC0 bits of the ISC register are used to implement LIN-bus communication operation by using channel 7 in association with the serial array unit. When the ISC1 bit is set to 1, the input signal of the serial data input pin (RxD2) is selected as a timer input signal. The ISC register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 7 - 22 Format of Input Switch Control Register (ISC)

Address: F0073H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
ISC	ISC7	ISC6	0	ISC4	ISC3	0	ISC1	ISC0

ISC1	Switching channel 7 input of timer array unit
0	Uses the input signal of the TI07 pin as a timer input (normal operation).
1	Input signal of the RxD2 pin is used as timer input (detects the wakeup signal and measures the low-level width of the break field and the pulse width of the sync field).

ISC0	Switching external interrupt (INTP0) input
0	Uses the input signal of the INTP0 pin as an external interrupt (normal operation).
1	Uses the input signal of the RxD2 pin as an external interrupt (wakeup signal detection).

Caution **Be sure to clear bits 5 and 2 to 0.**

Remark When the LIN-bus communication function is used, select the input signal of the RxD2 pin by setting ISC1 to 1.

7.3.15 Noise filter enable register 1 (NFEN1)

The NFEN1 register is used to set whether the noise filter can be used for the timer input signal to each channel. Enable the noise filter by setting the corresponding bits to 1 on the pins in need of noise removal. When the noise filter is enabled, after synchronization with the operating clock (f_{MCK}) for the target channel, whether the signal keeps the same value for two clock cycles is detected. When the noise filter is disabled, the input signal is only synchronized with the operating clock (f_{MCK}) for the target channel**Note**. The NFEN1 register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of each of these registers following a reset is 00H.

Note For details, see the following sections.

- **7.5.1 Counter clock (f_{TCLK}), 2.When valid edge of input signal via the TImn pin is selected (CCSmn = 1)**
- **7.5.2 Timing of the start of counting**
- **7.7 Timer Input (TImn) Control**

Figure 7 - 23 Format of Noise Filter Enable Register 1 (NFEN1)

Address: F0071H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
NFEN1	TNFEN07	TNFEN06	TNFEN05	TNFEN04	TNFEN03	TNFEN02	TNFEN01	TNFEN00

TNFEN07	Enabling/disabling use of the noise filter for the TI07 pin**Note**
0	Disables use of the noise filter for the TI07 pin.
1	Enables use of the noise filter for the TI07 pin.

TNFEN06	Enabling/disabling use of the noise filter for the TI06 pin
0	Disables use of the noise filter for the TI06 pin.
1	Enables use of the noise filter for the TI06 pin.

TNFEN05	Enabling/disabling use of the noise filter for the TI05 pin
0	Disables use of the noise filter for the TI05 pin.
1	Enables use of the noise filter for the TI05 pin.

TNFEN04	Enabling/disabling use of the noise filter for the TI04 pin
0	Disables use of the noise filter for the TI04 pin.
1	Enables use of the noise filter for the TI04 pin.

TNFEN03	Enabling/disabling use of the noise filter for the TI03 pin
0	Disables use of the noise filter for the TI03 pin.
1	Enables use of the noise filter for the TI03 pin.

TNFEN02	Enabling/disabling use of the noise filter for the TI02 pin
0	Disables use of the noise filter for the TI02 pin.
1	Enables use of the noise filter for the TI02 pin.

TNFEN01	Enabling/disabling use of the noise filter for the TI01 pin
0	Disables use of the noise filter for the TI01 pin.
1	Enables use of the noise filter for the TI01 pin.

TNFEN00	Enabling/disabling use of the noise filter for the TI00 pin
0	Disables use of the noise filter for the TI00 pin.
1	Enables use of the noise filter for the TI00 pin.

Note The applicable pin can be switched by setting the ISC1 bit of the ISC register.
ISC1 = 0: Whether or not to use the noise filter of the TI07 pin can be selected.
ISC1 = 1: Whether or not to use the noise filter of the RxD2 pin can be selected.

Remark The presence or absence of timer I/O pins of channel 0 to 7 depends on the product. See **Table 7 - 2 Timer I/O Pins Provided in Each Product** for details.

7.3.16 Registers for controlling the port functions multiplexed with the inputs and outputs of timers

Set the following registers to control the port functions multiplexed with the inputs and outputs of timer array units.

- Port mode registers (PMxx)
- Port registers (Pxx)
- Port output mode registers (POMxx)
- Port mode control A registers (PMCAxx)

For details, see the following sections.

- **4.3.1 Port mode registers (PMxx)**
- **4.3.2 Port registers (Pxx)**
- **4.3.5 Port output mode registers (POMxx)**
- **4.3.7 Port mode control A registers (PMCAxx)**

For examples of register settings for controlling port functions, see **4.5.4 Examples of register settings for port and alternate functions**.

7.4 Basic Specifications of the Timer Array Unit

7.4.1 Basic specifications of the simultaneous channel operation function

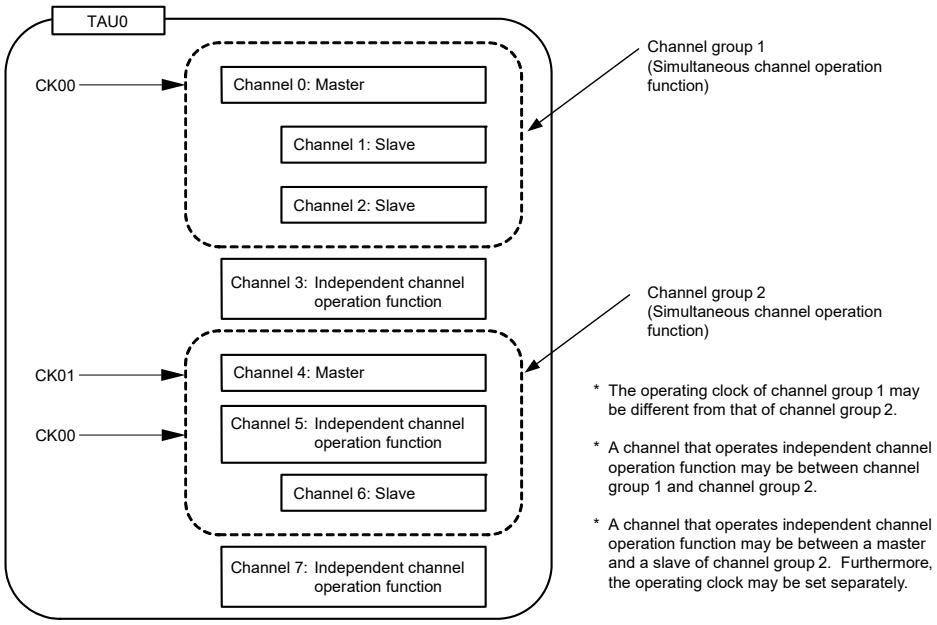
When simultaneously using multiple channels, namely, a combination of a master channel (a reference timer mainly counting the cycle) and slave channels (timers operating according to the master channel). The following lists the basic specifications of this function.

1. Only an even channel (channels 0, 2, 4, and 6) can be set as a master channel.
2. Any channel, except channel 0, can be set as a slave channel.
3. The slave channel must be lower than the master channel. For example, if channel 2 is set as a master channel, channel 3 and those that follow (i.e., channels 3, 4, 5, etc.) can be set as a slave channel.
4. Two or more slave channels can be set for one master channel.
5. When two or more master channels are to be used, slave channels with a master channel between them may not be set. For example, if channels 0 and 4 are set as master channels, channels 1 to 3 can be set as the slave channels of master channel 0, whereas channels 5 to 7 cannot be set as the slave channels of master channel 0.
6. The operating clock for a slave channel in combination with a master channel must be the same as that of the master channel. The CKSmn1 and CKSmn0 bits (bits 15 and 14 of timer mode register mn (TMRmn)) of the slave channel that operates in combination with the master channel must be the same value as that of the master channel.
7. A master channel can transmit INTTMmn (interrupt), start software trigger, and counter clock to the lower channels.
8. A slave channel can use INTTMmn (interrupt), a start software trigger, or the counter clock of the master channel as a source clock, but cannot transmit its own INTTMmn (interrupt), start software trigger, or counter clock to channels with lower channel numbers.
9. A master channel cannot use INTTMmn (interrupt), a start software trigger, or the counter clock from the other higher master channel as a source clock.
10. To simultaneously start channels that operate in combination, the operation enable (start) trigger bit (TSmn) of the channels in combination must be set at the same time.
11. During the counting operation, a TSmn bit of a master channel or TSmn bits of all channels which are operating simultaneously can be set. It cannot be applied to TSmn bits of slave channels alone.
12. To stop the channels in combination simultaneously, the operation stop trigger bit (TTmn) of the channels in combination must be set at the same time.
13. CKm2/CKm3 cannot be selected while channels are operating simultaneously, because the operating clocks of master channels and slave channels have to be synchronized.
14. Timer mode register m0 (TMRm0) has no master bit (it is fixed to 0). However, as channel 0 is the highest channel, it can be used as a master channel during simultaneous operation.

The specifications of the simultaneous channel operation function are applied in a channel group (a master channel and slave channels forming one simultaneous channel operation function). If two or more channel groups that do not operate in combination are specified, the basic specifications of the simultaneous channel operation function in **7.4.1 Basic specifications of the simultaneous channel operation function** do not apply to the channel groups.

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

Example


TAU0
CK00
Channel 0: Master
Channel 1: Slave
Channel 2: Slave
Channel 3: Independent channel operation function
CK01
Channel 4: Master
CK00
Channel 5: Independent channel operation function
Channel 6: Slave
Channel 7: Independent channel operation function
Channel group 1 (Simultaneous channel operation function)
Channel group 2 (Simultaneous channel operation function)
* The operating clock of channel group 1 may be different from that of channel group 2.
* A channel that operates independent channel operation function may be between channel group 1 and channel group 2.
* A channel that operates independent channel operation function may be between a master and a slave of channel group 2. Furthermore, the operating clock may be set separately.

7.4.2 Basic specifications of the 8-bit timer function (only for channels 1 and 3)

The 8-bit timer function allows the use of a 16-bit timer configured by cascade-connecting two 8-bit timer channels. This function can only be used with channels 1 and 3. The following lists the basic specifications of this function.

1. This 8-bit timer function (cascade connection to form a 16-bit timer) can only be used with channels 1 and 3.
2. To use the two channels as 8-bit timers, set the SPLITmn bits of timer mode registers mn (TMRmn) to 1.
3. The higher-numbered 8-bit timer can be used as an interval timer.
4. INTTMm1H and INTTMm3H interrupts are output at the start of operation by the higher-numbered 8-bit timer (which is the same as the operation when MDmn0 is set to 1).
5. Selection of the operating clock for the higher-numbered 8-bit timer is in accord with the setting of the CKSmn1 and CKSmn0 bits of the TMRmn register for the lower-numbered 8-bit timer.
6. Set the TSHm1 and TSHm3 bits to start operation of the higher-numbered 8-bit timer as an 8-bit timer, and the TTHm1 and TTHm3 bits to stop operation. The channel state can be checked by reading the TEHm1 and TEHm3 bits.
7. Operation of the lower-numbered 8-bit timer is in accord with the setting of the TMRmn register, through which any of the following three functions is selectable.
   - Interval timer square wave output
   - External event counter
   - Delay counter
8. Set the TSm1 and TSm3 bits to start operation of the lower-numbered 8-bit timer as an 8-bit timer, and the TTm1 and TTm3 bits to stop operation. The channel state can be checked by reading the TEm1 and TEm3 bits.
9. When the channels are to be used as a 16-bit timer, setting the TSHm1, TSHm3, TTHm1, and TTHm3 bits has no effect. Set the TSm1, TSm3, TTm1, and TTm3 bits to start operation by both channels 1 and 3. These trigger bits do not affect the TEHm3 and TEHm1 bits.
10. The simultaneous operations (one-shot pulse, PWM, and multiple PWM) cannot be used when the 8-bit timer functions are selected by the SPLITmn bits.

Remark m: Unit number (m = 0), n: Channel number (n = 1, 3)

7.5 Operations of Counters

7.5.1 Counter clock (fTCLK)

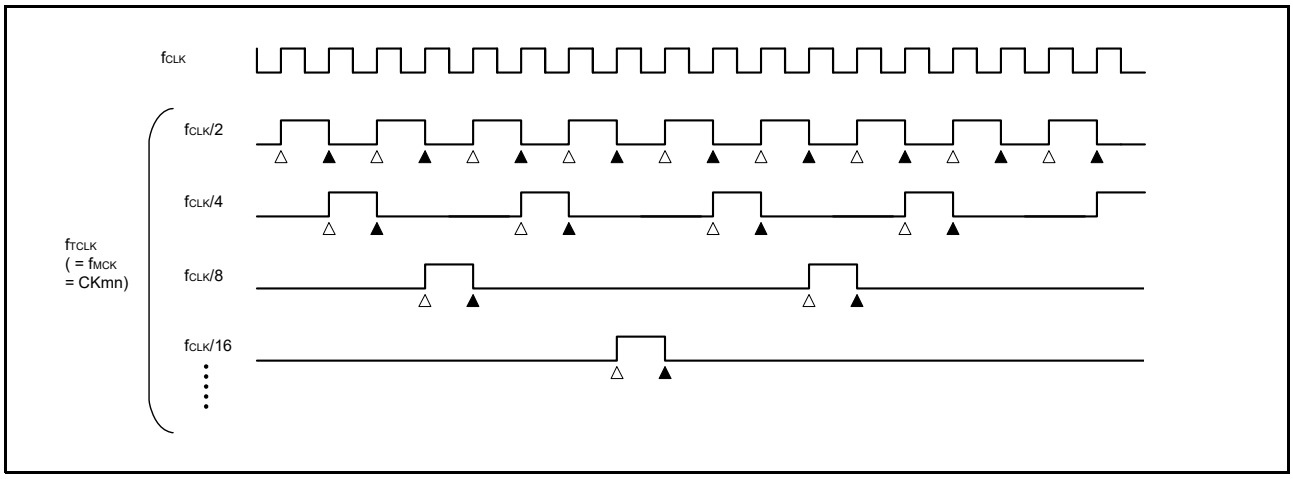
The counter clock (fTCLK) of the timer array unit can be selected between following by CCSmn bit of timer mode register mn (TMRmn).

- Operating clock (fMCK) specified by the CKSmn1 and CKSmn0 bits
- Valid edge of input signal input from the TImn pin

Because the timer array unit is designed to operate in synchronization with fCLK, the timings of the counter clock (fTCLK) are shown below.

1. When operating clock (fMCK) specified by the CKSmn1 and CKSmn0 bits is selected (CCSmn = 0)
   The counter clock (fTCLK) is between fCLK to $f_{CLK}/2^{15}$ by setting of timer clock select register m (TPSm). When a divided fCLK is selected, however, the clock selected in TPSmn register, but a signal which becomes high level for one period of fCLK from its rising edge. When a fCLK is selected, fixed to high level. Counting of timer counter register mn (TCRmn) delayed by one period of fCLK from rising edge of the counter clock, because of synchronization with fCLK. But, this is described as “counting at rising edge of the counter clock”, as a matter of convenience.

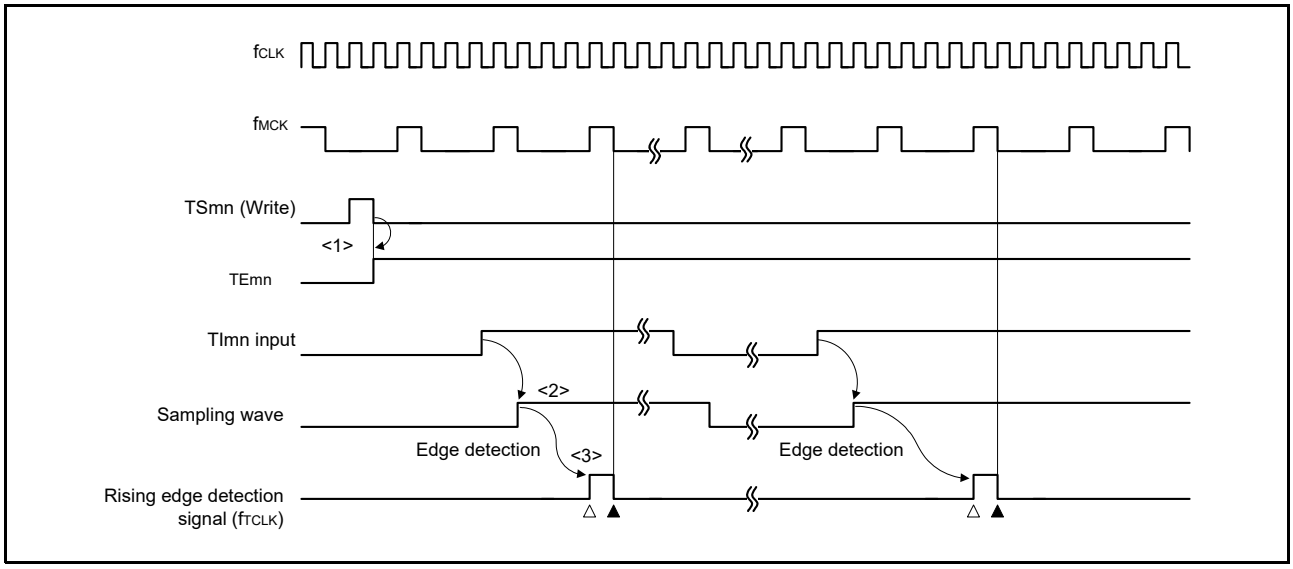
Figure 7 - 24 Timing of fCLK and Counter Clock (fTCLK) (When CCSmn = 0)



Remark 1. △ : Rising edge of the counter clock
▲ : Synchronization, increment/decrement of counter
Remark 2. fCLK: CPU/peripheral hardware clock

2. When valid edge of input signal via the TImn pin is selected (CCSmn = 1)
   The counter clock (fTCLK) becomes the signal that detects valid edge of input signal via the TImn pin and synchronizes next rising fMCK. The counter clock (fTCLK) is delayed for 1 to 2 period of fMCK from the input signal via the TImn pin (when a noise filter is used, the delay becomes 3 to 4 clock). Counting of timer counter register mn (TCRmn) delayed by one period of fCLK from rising edge of the counter clock, because of synchronization with fCLK. But, this is described as “counting at valid edge of input signal via the TImn pin”, as a matter of convenience.

Figure 7 - 25 Timing of fCLK and Counter Clock (fTCLK) (When CCSmn = 1, Noise Filter Unused)



<1> Setting TSmn bit to 1 enables the timer to be started and to become wait state for valid edge of input signal via the TImn pin.

<2> The rise of input signal via the TImn pin is sampled by fMCK.

<3> The edge is detected by the rising of the sampled signal and the detection signal (counter clock) is output.

Remark 1. △ : Rising edge of the counter clock
▲ : Synchronization, increment/decrement of counter

Remark 2. fCLK: CPU/peripheral hardware clock
fMCK: Operating clock of channel n

Remark 3. The waveforms of the input signals via TImn pin of the input pulse interval measurement, the measurement of low- and high-level width of input signal, the delay counter, and the one-shot pulse output are the same.

7.5.2 Timing of the start of counting

Timer counter register mn (TCRmn) becomes enabled to operation by setting of TSmn bit of timer channel start register m (TSm). **Table 7 - 6** shows operations from count operation enabled state to timer counter register mn (TCRmn) count start.

Table 7 - 6 Operations from the Count Operation Enabled State to the Start of Counting by a Timer Counter Register mn (TCRmn)

Timer operation mode	Operation when TSmn = 1 is set
Interval timer mode	No operation is carried out from start trigger detection (TSmn=1) until counter clock generation. The first counter clock loads the value of the TDRmn register to the TCRmn register and the subsequent counter clock performs count down operation (see **7.5.3 Operations of counters, 1. Operation in interval timer mode**).
Event counter mode	Writing 1 to the TSmn bit loads the value of the TDRmn register to the TCRmn register. If detect edge of TImn input. The subsequent counter clock performs count down operation (see **7.5.3 Operations of counters, 2. Operation in event counter mode**).
Capture mode	No operation is carried out from start trigger detection (TSmn = 1) until counter clock generation. The first counter clock loads 0000H to the TCRmn register and the subsequent counter clock performs count up operation (see **7.5.3 Operations of counters**, **3. Operation in capture mode (input pulse interval measurement)**).
One-count mode	The waiting-for-start-trigger state is entered by writing 1 to the TSmn bit while the timer is stopped (TEmn = 0). No operation is carried out from start trigger detection until counter clock generation. The first counter clock loads the value of the TDRmn register to the TCRmn register and the subsequent counter clock performs count down operation (see **7.5.3 Operations of counters, 4. Operation in one-count mode**).
Capture & one-count mode	The waiting-for-start-trigger state is entered by writing 1 to the TSmn bit while the timer is stopped (TEmn = 0). No operation is carried out from start trigger detection until counter clock generation. The first counter clock loads 0000H to the TCRmn register and the subsequent counter clock performs count up operation (see **7.5.3 Operations of counters, 5. Operation in capture & one-count mode (high-level width measurement)**).

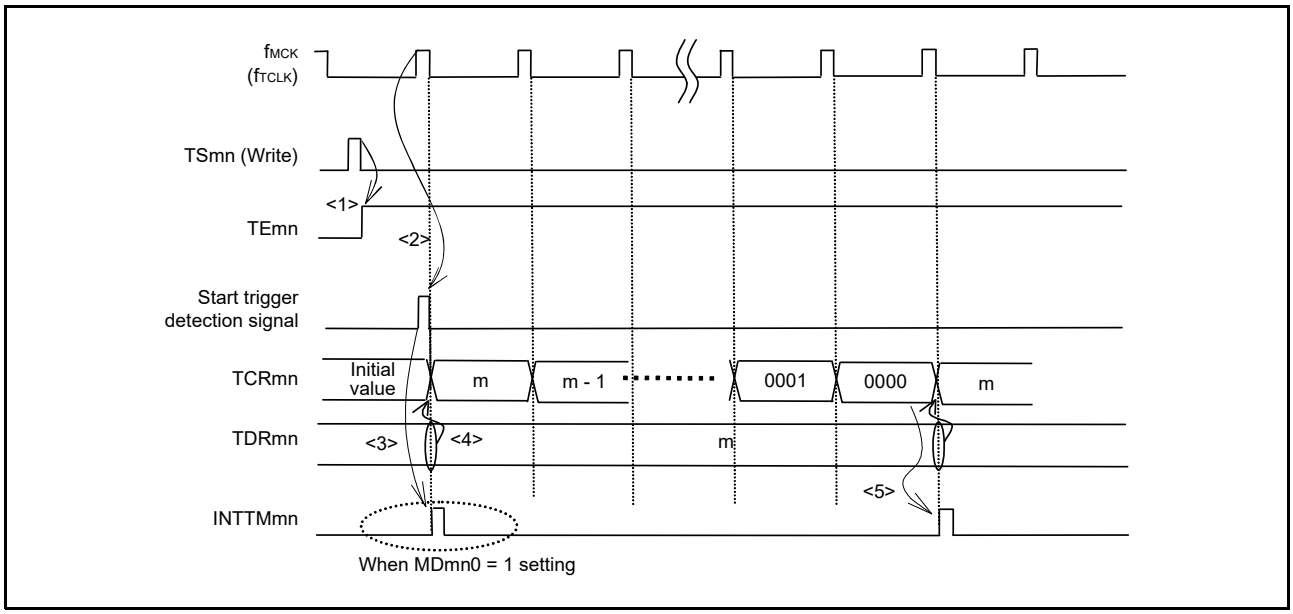
7.5.3 Operations of counters

Here, the counter operation in each mode is explained.

1. Operation in interval timer mode

<1> Operation is enabled (TEmn = 1) by writing 1 to the TSmn bit. Timer counter register mn (TCRmn) holds the initial value until counter clock generation.
<2> A start trigger is generated at the first counter clock after operation is enabled.
<3> When the MDmn0 bit is set to 1, INTTMmn is generated by the start trigger.
<4> By the first counter clock after the operation enable, the value of timer data register mn (TDRmn) is loaded to the TCRmn register and counting starts in the interval timer mode.
<5> When the TCRmn register counts down and its count value is 0000H, INTTMmn is generated and the value of timer data register mn (TDRmn) is loaded to the TCRmn register and counting keeps on.

Figure 7 - 26 Timing during Operation in Interval Timer Mode



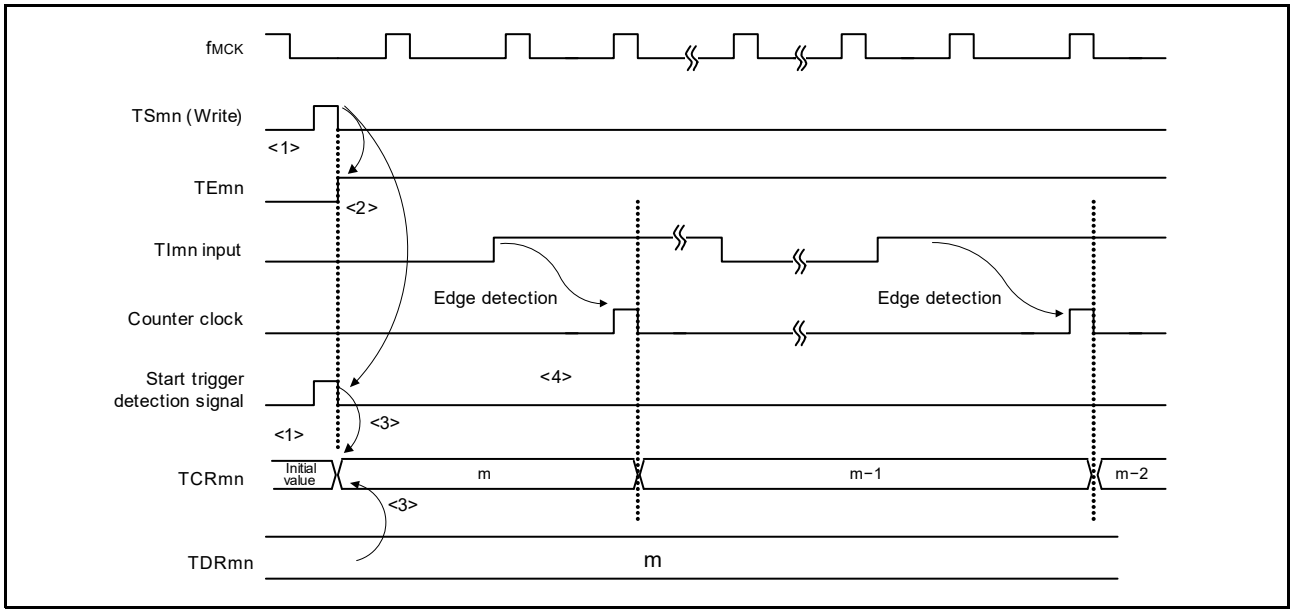
Caution **In the first cycle operation of counter clock after writing the TSmn bit, an error at a maximum of one clock is generated since count start delays until counter clock has been generated. When the information on count start timing is necessary, an interrupt can be generated at count start by setting MDmn0 = 1.**

Remark fMCK, the start trigger detection signal, and INTTMmn become active between one clock in synchronization with fCLK.

2. Operation in event counter mode

<1> Timer counter register mn (TCRmn) holds its initial value while operation is stopped (TEmn = 0).
<2> Operation is enabled (TEmn = 1) by writing 1 to the TSmn bit.
<3> As soon as 1 has been written to the TSmn bit and 1 has been set to the TEmn bit, the value of timer data register mn (TDRmn) is loaded to the TCRmn register to start counting.
<4> After that, the TCRmn register value is counted down according to the counter clock of the valid edge of the TImn input.

Figure 7 - 27 Timing during Operation in Event Counter Mode

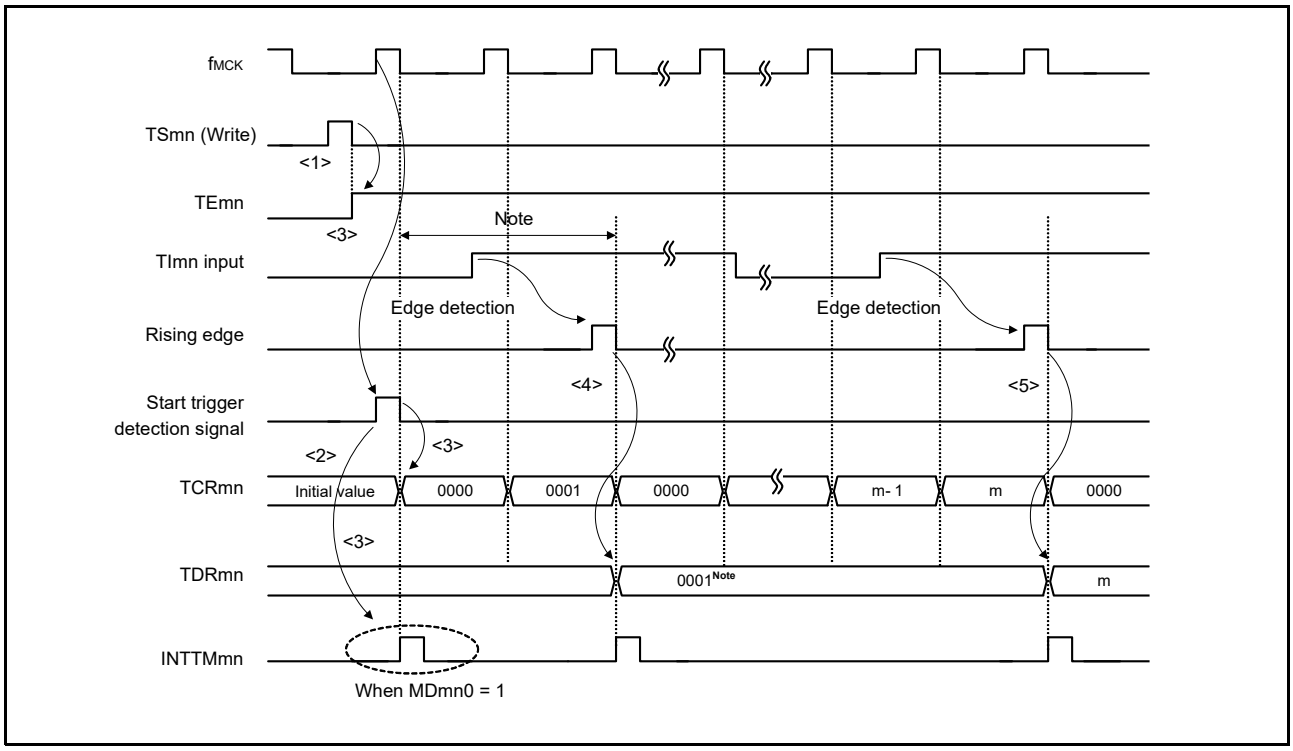


Remark The above figure shows the timing when the noise filter is not used. By making the noise filter on-state, the edge detection becomes 2 fMCK cycles (it sums up to 3 to 4 cycles) later than the normal cycle of TImn input. The error per one period occurs be the asynchronous between the period of the TImn input and that of the counter clock (fMCK).

3. Operation in capture mode (input pulse interval measurement)

<1> Operation is enabled (TEmn = 1) by writing 1 to the TSmn bit.
<2> Timer counter register mn (TCRmn) holds the initial value until counter clock generation.
<3> A start trigger is generated at the first counter clock after operation is enabled. And the value of 0000H is loaded to the TCRmn register and counting starts in the capture mode. (When the MDmn0 bit is set to 1, INTTMmn is generated by the start trigger.)
<4> On detection of the valid edge of the TImn input, the value of the TCRmn register is captured to timer data register mn (TDRmn) and INTTMmn is generated. However, this captured value is meaningless. The TCRmn register keeps on counting from 0000H.
<5> On next detection of the valid edge of the TImn input, the value of the TCRmn register is captured to timer data register mn (TDRmn) and INTTMmn is generated.

Figure 7 - 28 Timing during Operation in Capture Mode (Input Pulse Interval Measurement)



Note If a clock has been input to TImn (the trigger exists) when capturing starts, counting starts when a trigger is detected, even if no edge is detected. Therefore, the first captured value (<4>) does not determine a pulse interval (in the above figure, 0001 just indicates two clock cycles but does not determine the pulse interval) and so the user can ignore it.

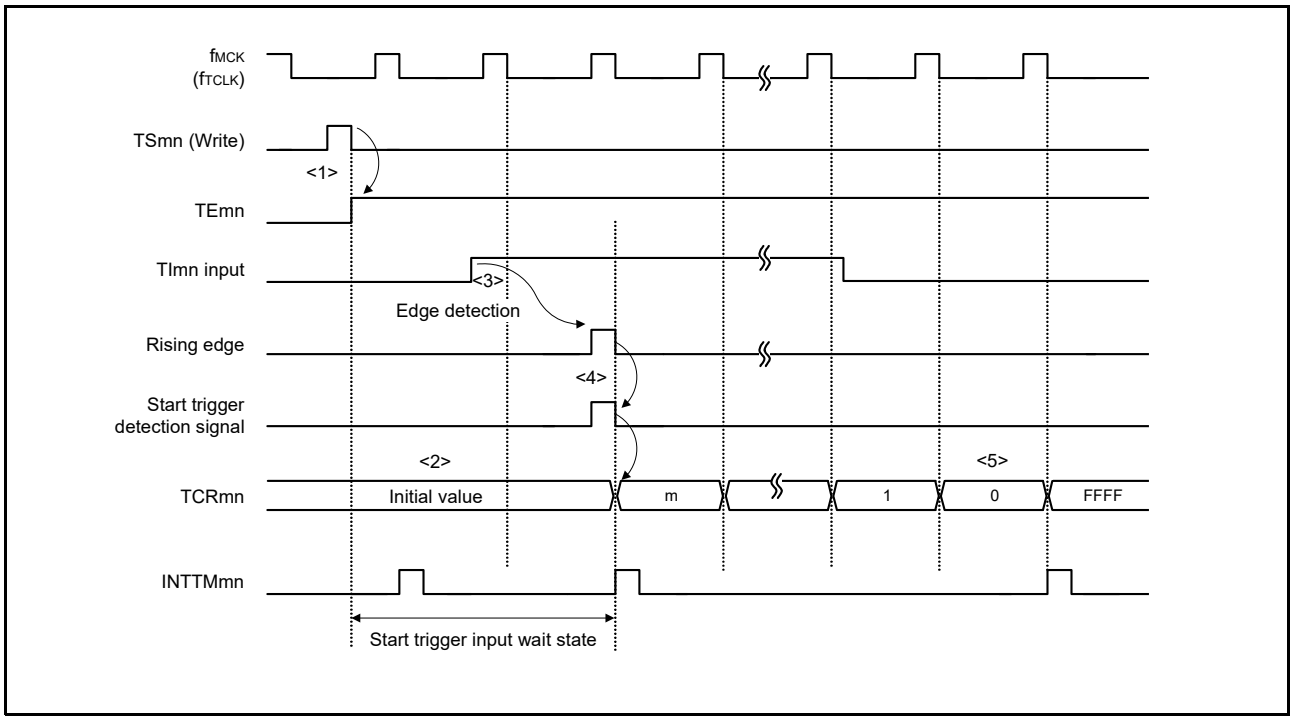
Caution **In the first cycle operation of counter clock after writing the TSmn bit, an error at a maximum of one clock is generated since count start delays until counter clock has been generated. When the information on count start timing is necessary, an interrupt can be generated at count start by setting MDmn0 = 1.**

Remark The above figure shows the timing when the noise filter is not used. By making the noise filter on-state, the edge detection becomes 2 fMCK cycles (it sums up to 3 to 4 cycles) later than the normal cycle of TImn input. The error per one period occurs be the asynchronous between the period of the TImn input and that of the counter clock (fMCK).

4. Operation in one-count mode

<1> Operation is enabled (TEmn = 1) by writing 1 to the TSmn bit.
<2> Timer counter register mn (TCRmn) holds the initial value until start trigger generation.
<3> Rising edge of the TImn input is detected.
<4> On start trigger detection, the value of timer data register mn (TDRmn) is loaded to the TCRmn register and count starts.
<5> When the TCRmn register counts down and its count value is 0000H, INTTMmn is generated and the value of the TCRmn register becomes FFFFH and counting stops.

Figure 7 - 29 Timing during Operation in One-count Mode

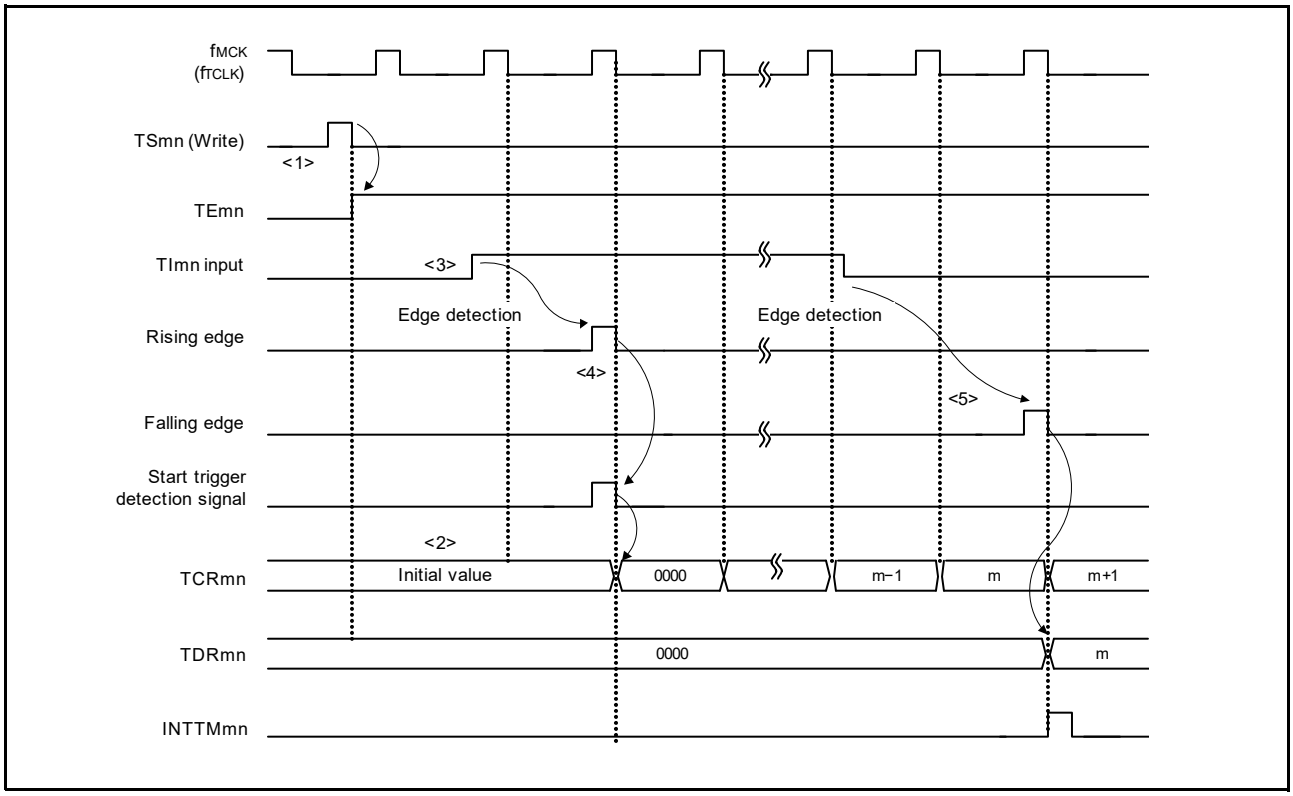


Remark The above figure shows the timing when the noise filter is not used. By making the noise filter on-state, the edge detection becomes 2 fMCK cycles (it sums up to 3 to 4 cycles) later than the normal cycle of TImn input. The error per one period occurs be the asynchronous between the period of the TImn input and that of the counter clock (fMCK).

5. Operation in capture & one-count mode (high-level width measurement)

<1> Operation is enabled (TEmn = 1) by writing 1 to the TSmn bit of timer channel start register m (TSm).
<2> Timer counter register mn (TCRmn) holds the initial value until start trigger generation.
<3> Rising edge of the TImn input is detected.
<4> On start trigger detection, the value of 0000H is loaded to the TCRmn register and count starts.
<5> On detection of the falling edge of the TImn input, the value of the TCRmn register is captured to timer data register mn (TDRmn) and INTTMmn is generated.

Figure 7 - 30 Timing during Operation in Capture & One-count Mode (High-level Width Measurement)

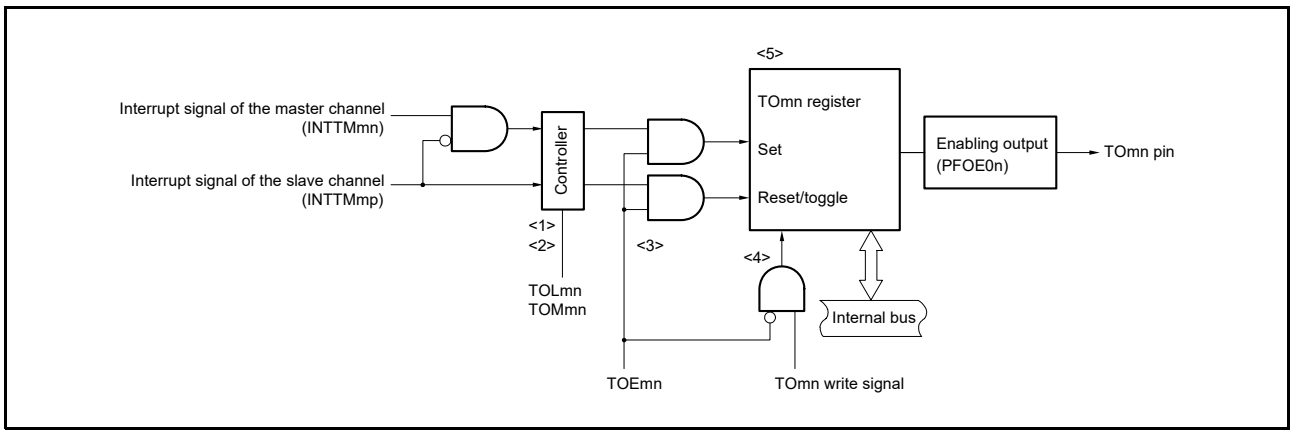


Remark The above figure shows the timing when the noise filter is not used. By making the noise filter on-state, the edge detection becomes 2 fMCK cycles (it sums up to 3 to 4 cycles) later than the normal cycle of TImn input. The error per one period occurs be the asynchronous between the period of the TImn input and that of the counter clock (fMCK).

7.6 Channel Output (TOmn Pin) Control

7.6.1 TOmn pin output circuit configuration

Figure 7 - 31 Output Circuit Configuration



The following describes the TOmn pin output circuit.

<1> When TOMmn = 0 (master channel output mode), the set value of timer output level register m (TOLm) is ignored and only INTTM0p (slave channel timer interrupt) is transmitted to timer output register m (TOm).

<2> When TOMmn = 1 (slave channel output mode), both INTTMmn (master channel timer interrupt) and INTTM0p (slave channel timer interrupt) are transmitted to the TOm register. At this time, the TOLm register becomes valid and the signals are controlled as follows:

- When TOLmn = 0: Positive logic output (INTTMmn → set, INTTM0p → reset)
- When TOLmn = 1: Negative logic output (INTTMmn → reset, INTTM0p → set)

When INTTMmn and INTTM0p are simultaneously generated, (0% output of PWM), INTTM0p (reset signal) takes priority, and INTTMmn (set signal) is masked.

<3> While timer output is enabled (TOEmn = 1), INTTMmn (master channel timer interrupt) and INTTM0p (slave channel timer interrupt) are transmitted to the TOm register. Writing to the TOm register (TOmn write signal) becomes invalid. When TOEmn = 1, the TOmn pin output never changes with signals other than interrupt signals. To initialize the TOmn pin output level, it is necessary to set timer operation is stopped (TOEmn = 0) and to write a value to the TOm register.

<4> While timer output is disabled (TOEmn = 0), writing to the TOmn bit to the target channel (TOmn write signal) becomes valid. When timer output is disabled (TOEmn = 0), neither INTTMmn (master channel timer interrupt) nor INTTM0p (slave channel timer interrupt) is transmitted to the TOm register.

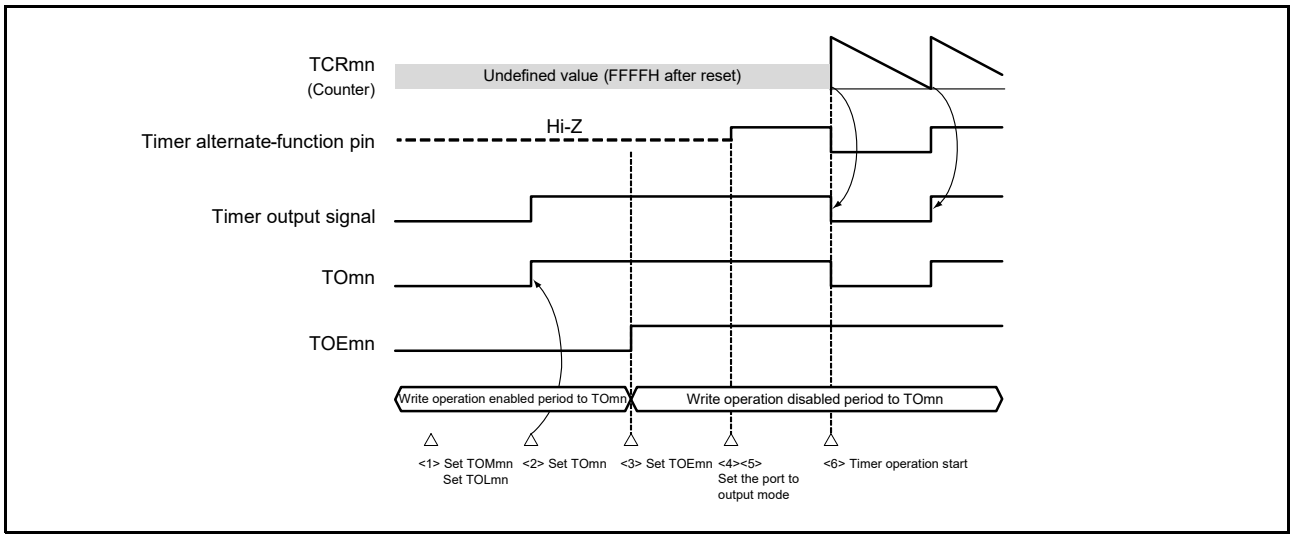
<5> The TOm register can always be read, and the TOmn pin output level can be checked.

Remark m: Unit number (m = 0)
n: Channel number
n = 0 to 7 (n = 0, 2, 4, 6 for master channel)
p: Slave channel number
n < p ≤ 7

7.6.2 TOmn pin output setting

The following figure shows the procedure and state transitions from the initial settings of a TOmn output pin to the start of timer operation.

Figure 7 - 32 State Transitions from the Settings for Timer Output to the Start of Timer Operation



<1> The operation mode of timer output is set.

- TOMmn bit (0: Master channel output mode, 1: Slave channel output mode)
- TOLmn bit (0: Positive logic output, 1: Negative logic output)

<2> The timer output signal is set to the initial state by setting timer output register m (TOm).

<3> The timer output operation is enabled by writing 1 to the TOEmn bit (writing to the TOm register is disabled).

<4> The port is set to digital I/O by the port mode control A register (PMCAxx) (see **7.3.16 Registers for controlling the port functions multiplexed with the inputs and outputs of timers**).

<5> The port I/O setting is set to output (see **7.3.16 Registers for controlling the port functions multiplexed with the inputs and outputs of timers**).

<6> The timer operation is enabled (TSmn = 1).

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

7.6.3 Cautions on channel output operation

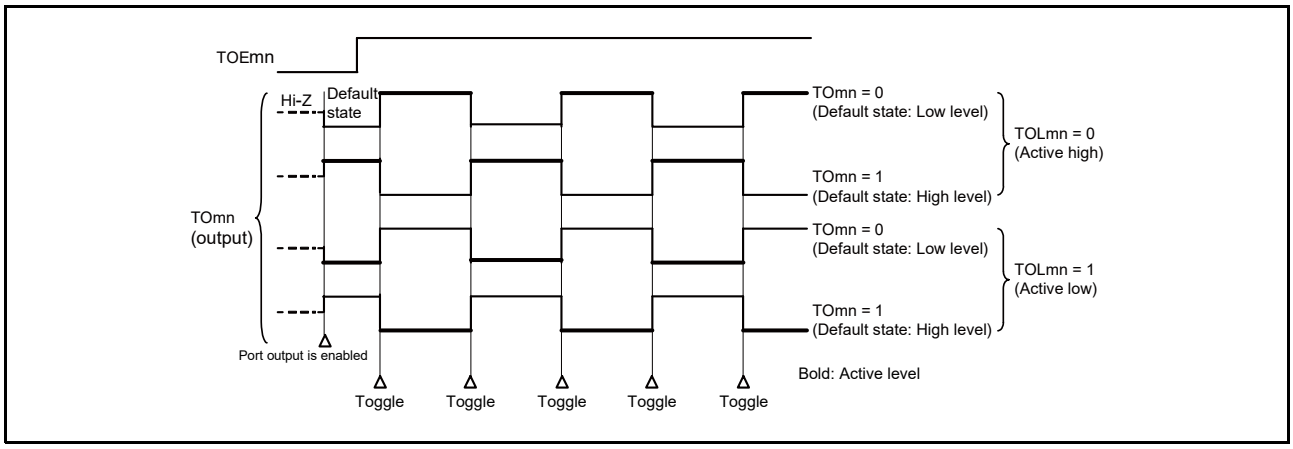
1. Changing values set in the registers TOm, TOEm, and TOLm during timer operation
   Since the timer operations (operations of timer counter register mn (TCRmn) and timer data register mn (TDRmn)) are independent of the TOmn output circuit and changing the values set in timer output register m (TOm), timer output enable register m (TOEm), and timer output level register m (TOLm) does not affect the timer operation, the values can be changed during timer operation. To output an expected waveform from the TOmn pin by timer operation, however, set the TOm, TOEm, TOLm, and TOMm registers to the values stated in the register setting example of each operation shown in **7.7 Timer Input (TImn) Control** and **7.8 Independent Channel Operation Function of Timer Array Unit**. When the values set to the TOEm, and TOMm registers (but not the TOm register) are changed close to the occurrence of the timer interrupt (INTTMmn) of each channel, the waveform output to the TOmn pin might differ, depending on whether the values are changed immediately before or immediately after the timer interrupt (INTTMmn) occurs.

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

2. Default level of TOmn pin and output level after timer operation start
   The change in the output level of the TOmn pin when timer output register m (TOm) is written while timer output is disabled (TOEmn = 0), the initial level is changed, and then timer output is enabled (TOEmn = 1) before port output is enabled, is shown below.

   a) When operation starts with master channel output mode (TOMmn = 0) setting
      The setting of timer output level register m (TOLm) is invalid when master channel output mode (TOMmn = 0).
      When the timer operation starts after setting the default level, the toggle signal is generated and the output level of the TOmn pin is inverted.

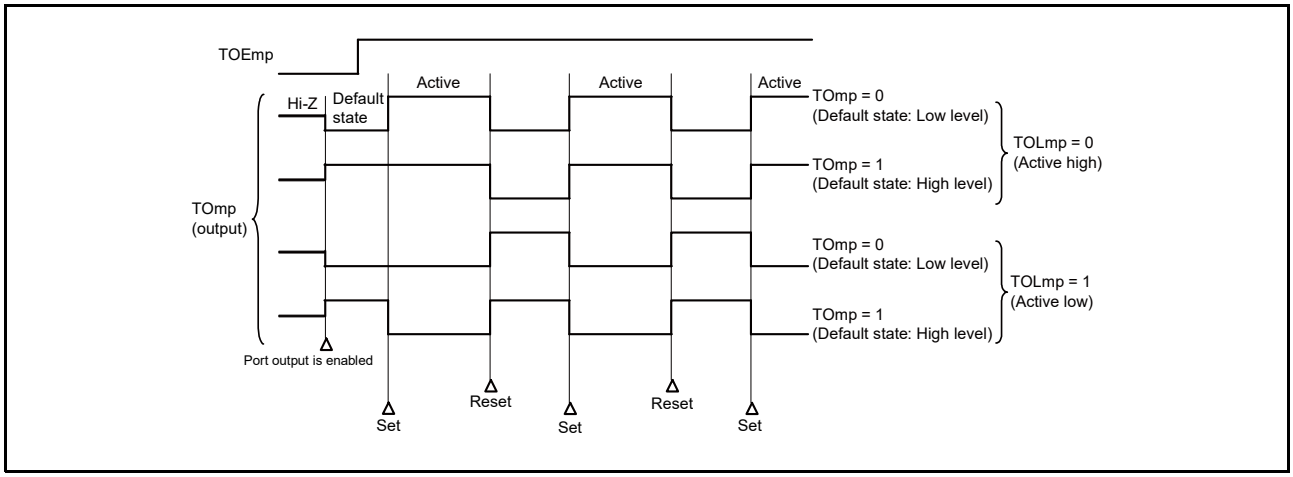
Figure 7 - 33 TOmn Pin Output States with Toggled Output (TOMmn = 0)



Remark 1. Toggle: Toggle signal to invert the output on the TOmn pin
Remark 2. m: Unit number (m = 0), n: Channel number (n = 0 to 7)

b) When operation starts with slave channel output mode (TOMmp = 1) setting (PWM output)
When slave channel output mode (TOMmp = 1), the active level is determined by timer output level register m (TOLm) setting.

Figure 7 - 34 TOmp Pin Output States with PWM Output (TOMmp = 1)



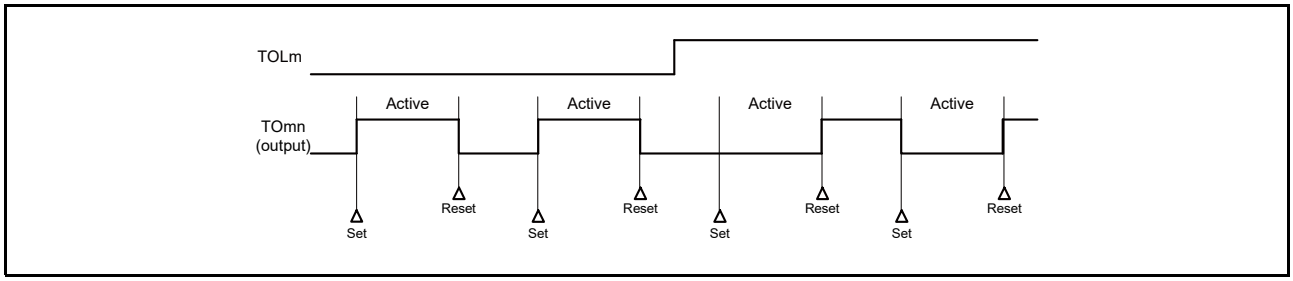
Remark 1. Set: The output signal of the TOmp pin changes from inactive level to active level.
Reset: The output signal of the TOmp pin changes from active level to inactive level.

Remark 2. m: Unit number (m = 0), p: Channel number (p = 1 to 7)

3. Operation of TOmn pin in slave channel output mode (TOMmn = 1)

a) When the relevant bit of timer output level register m (TOLm) is changed during timer operation
When the TOLm register setting has been changed during timer operation, the setting becomes valid at the generation timing of the TOmn pin change condition. Rewriting the TOLm register does not change the output level of the TOmn pin. The operation when TOMmn is set to 1 and the value of the TOLm register is changed while the timer is operating (TEmn = 1) is shown below.

Figure 7 - 35 Operation When the Relevant Bit of the TOLm Register Is Changed during Timer Operation



Remark 1. Set: The output signal of the TOmn pin changes from inactive level to active level.
Reset: The output signal of the TOmn pin changes from active level to inactive level.

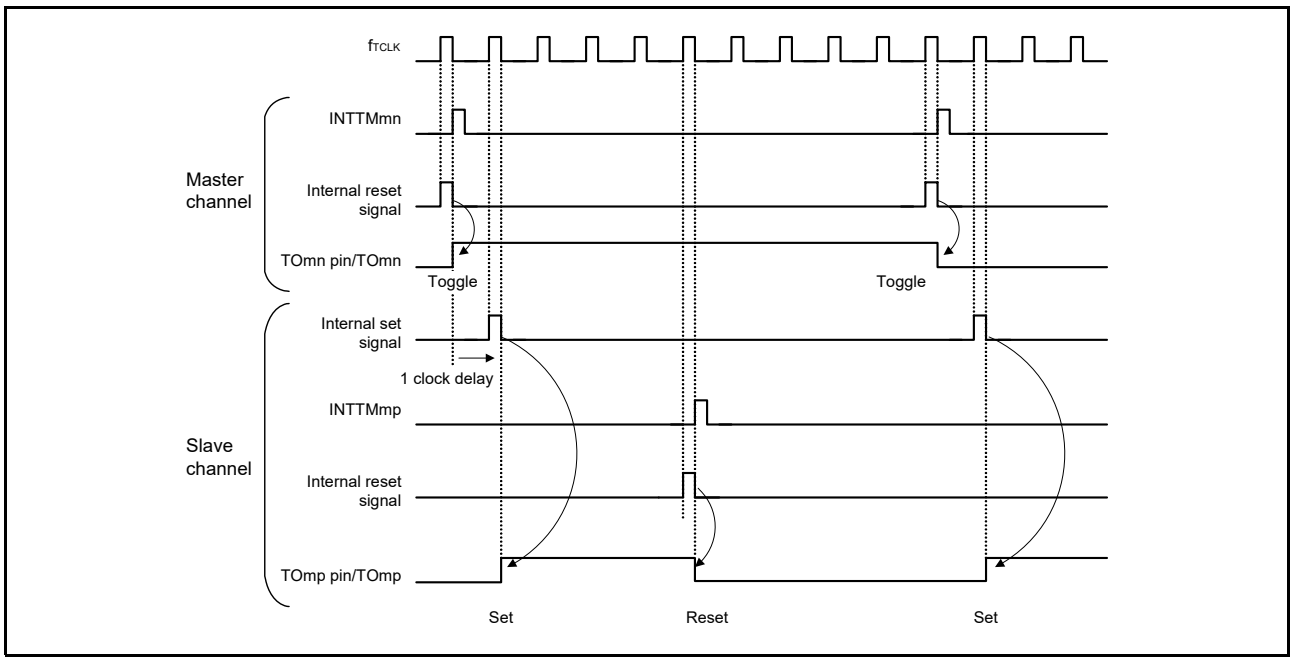
Remark 2. m: Unit number (m = 0), n: Channel number (n = 0 to 7)

b) Set/reset timing
To realize 0%/100% output at PWM output, the TOmn pin/TOmn bit set timing at master channel timer interrupt (INTTMmn) generation is delayed by 1 counter clock by the slave channel. If the set condition and reset condition are generated at the same time, a higher priority is given to the latter. **Figure 7 - 36** shows the states of operation following set and reset signals when the master and slave channels are set as follows.

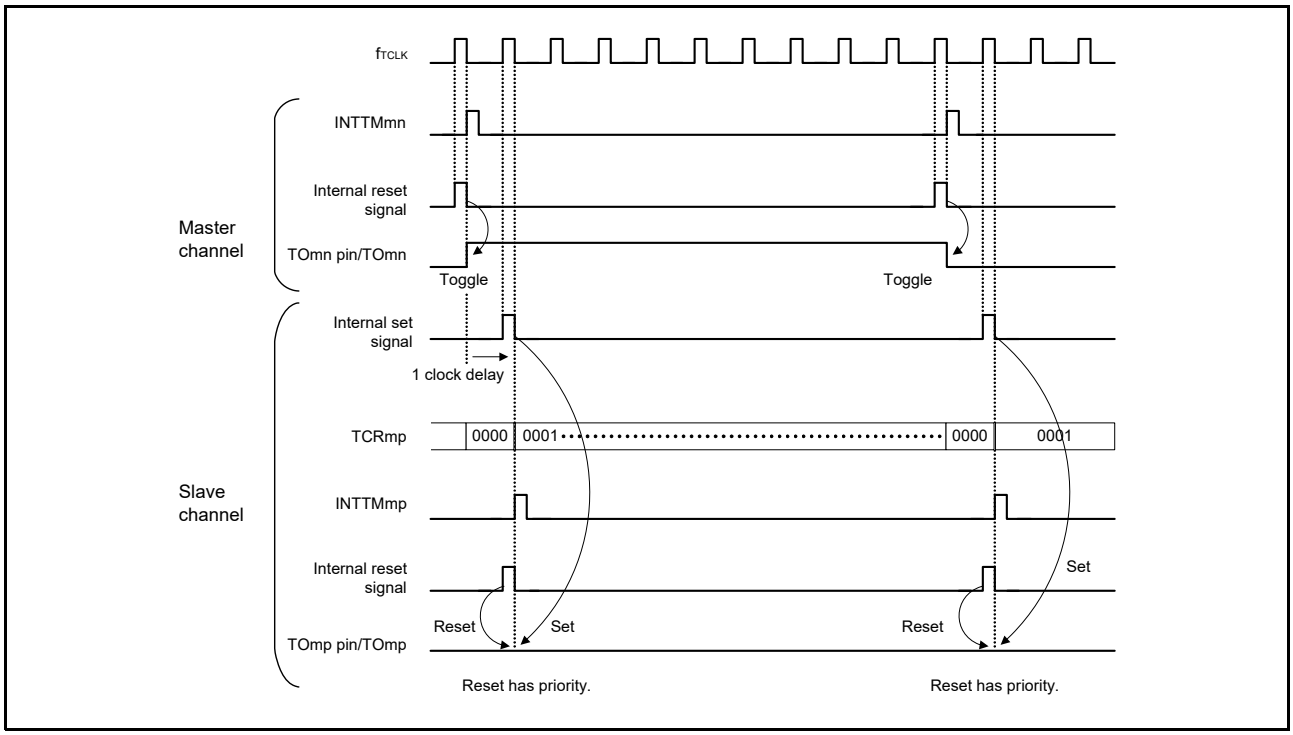
- Master channel: TOEmn = 1, TOMmn = 0, TOLmn = 0
- Slave channel: TOEmp = 1, TOMmp = 1, TOLmp = 0

Figure 7 - 36 States of Operation following Set and Reset Signals

1. Basic timing during operation



2. Timing during operation with the duty cycle set to 0%



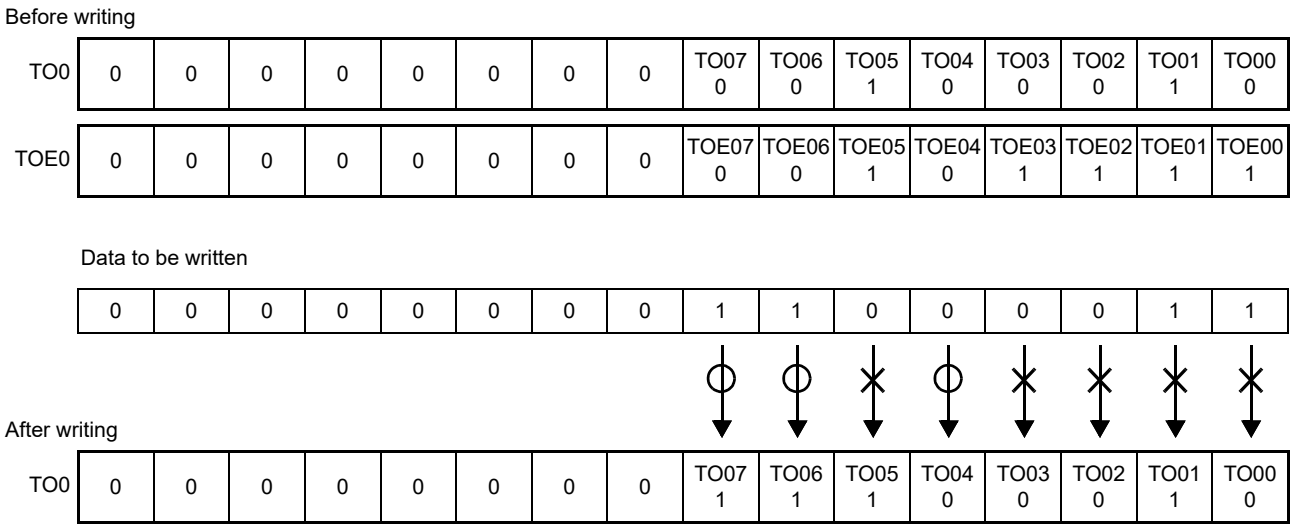
Remark 1. Internal reset signal: TOmn pin reset/toggle signal
Internal set signal: TOmn pin set signal

Remark 2. m: Unit number (m = 0, 1)
n: Channel number
n = 0 to 7 (n = 0, 2, 4, 6 for master channel)
p: Slave channel number
n < p ≤ 7

7.6.4 Collective manipulation of TOmn bit

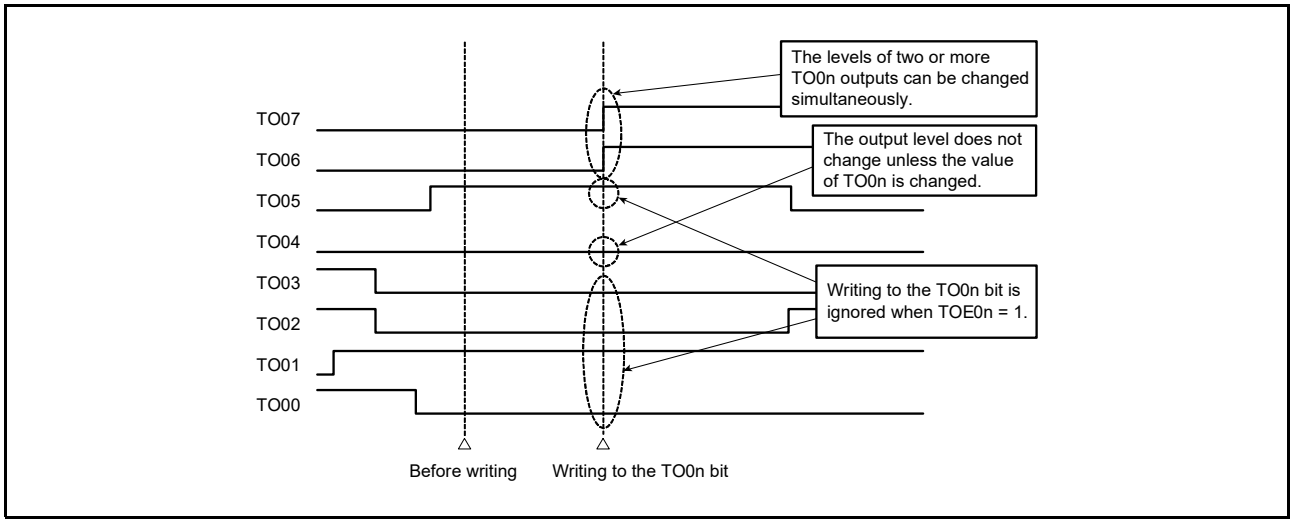
In timer output register m (TOm), the setting bits for all the channels are located in one register in the same way as timer channel start register m (TSm). Therefore, the TOmn bit of all the channels can be manipulated collectively. Only the desired bits can also be manipulated by enabling writing only to the TOmn bits (TOEmn = 0) that correspond to the relevant bits of the channel used to perform output (TOmn).

Figure 7 - 37 Example of TO0n Bit Collective Manipulation



Writing is done only to the TOmn bit with TOEmn = 0, and writing to the TOmn bit with TOEmn = 1 is ignored. TOmn (channel output) to which TOEmn = 1 is set is not affected by the write operation. Even if the write operation is done to the TOmn bit, it is ignored and the output change by timer operation is normally done.

Figure 7 - 38 TO0n Pin States by Collective Manipulation of TO0n Bits



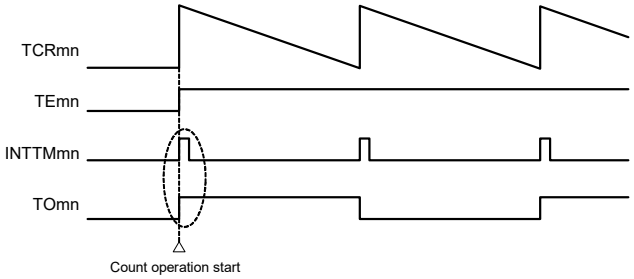
Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

7.6.5 Timer interrupts and TOmn outputs when counting is started

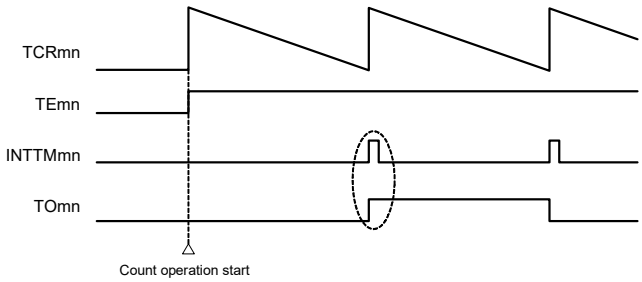
In the interval timer mode or capture mode, the MDmn0 bit in timer mode register mn (TMRmn) sets whether or not to generate a timer interrupt at count start. When MDmn0 is set to 1, the count operation start timing can be known by the timer interrupt (INTTMmn) generation. In the other modes, neither timer interrupt at count operation start nor TOmn output is controlled. The following figure shows operation examples when the interval timer mode (TOEmn = 1, TOMmn = 0) is set.

Figure 7 - 39 Examples of the Operation of Timer Interrupts and TOmn Outputs When Counting Is Started

a) When MDmn0 is set to 1



b) When MDmn0 is set to 0



When MDmn0 is set to 1, a timer interrupt (INTTMmn) is output at count operation start, and TOmn performs a toggle operation. When MDmn0 is set to 0, a timer interrupt (INTTMmn) is not output at count operation start, and TOmn does not change either. After counting one cycle, INTTMmn is output and TOmn performs a toggle operation.

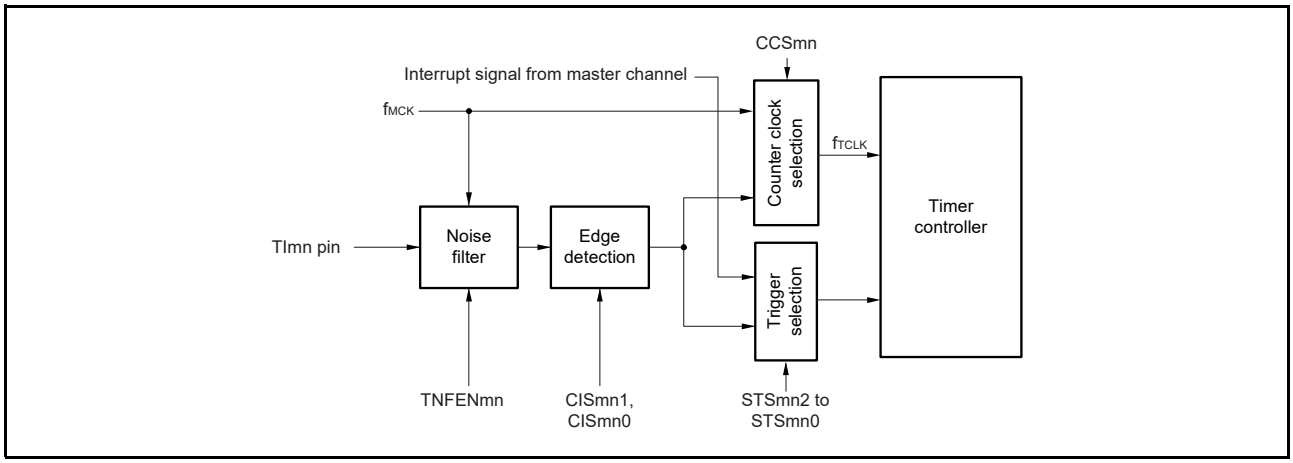
Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

7.7 Timer Input (TImn) Control

7.7.1 TImn input circuit configuration

A signal is input from a timer input pin, goes through a noise filter and an edge detector, and is sent to a timer controller. Enable the noise filter for the pin in need of noise removal. The following shows the configuration of the input circuit.

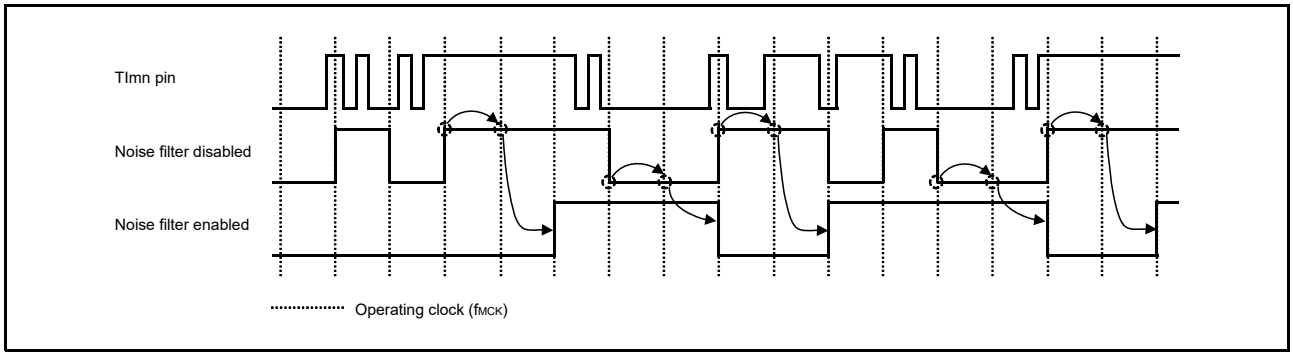
Figure 7 - 40 Input Circuit Configuration



7.7.2 Noise filter

When the noise filter is disabled, the input signal is only synchronized with the operating clock (fMCK) for channel n. When the noise filter is enabled, after synchronization with the operating clock (fMCK) for channel n, whether the signal keeps the same value for two clock cycles is detected. The following shows differences in waveforms output from the noise filter between when the noise filter is enabled and disabled.

Figure 7 - 41 Sampling Waveforms through TImn Input Pin with Noise Filter Enabled and Disabled



7.7.3 Cautions on channel input operation

When a timer input pin is set as unused, the operating clock is not supplied to the noise filter. Therefore, after settings are made to use the timer input pin, the following wait time is necessary before a trigger is specified to enable operation of the channel corresponding to the timer input pin.

1. Noise filter is disabled
   When the CCSmn, STSmn1, and STSmn0 bits in the timer mode register mn (TMRmn) are all 0 and then one of them is set to 1, wait for at least two cycles of the operating clock (f_{MCK}), and then set the operation enable trigger bit in the timer channel start register (TSm).

2. Noise filter is enabled
   When the CCSmn, STSmn1, and STSmn0 bits in the timer mode register mn (TMRmn) are all 0 and then one of them is set to 1, wait for at least four cycles of the operating clock (f_{MCK}), and then set the operation enable trigger bit in the timer channel start register (TSm).

7.8 Independent Channel Operation Function of Timer Array Unit

7.8.1 Operation as an interval timer or for square wave output

1. Interval timer

   The timer array unit can be used as a reference timer that generates INTTMmn (timer interrupt) at fixed intervals. The interrupt generation period can be calculated by the following expression.

Generation period of INTTMmn (timer interrupt) = Period of counter clock × (Set value of TDRmn + 1)

2. Operation for square wave output

   TOmn performs a toggle operation as soon as INTTMmn has been generated, and outputs a square wave with a duty factor of 50%. The period and frequency for outputting a square wave from TOmn can be calculated by the following expressions.

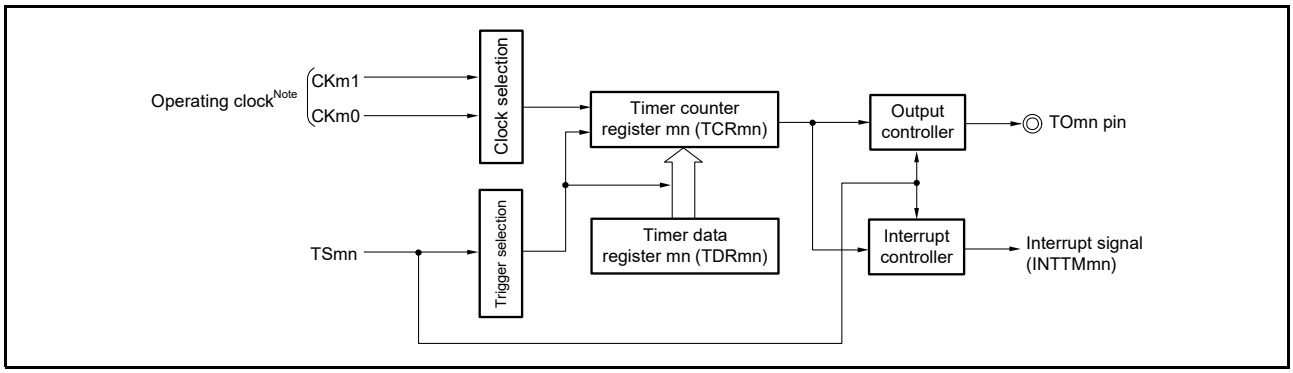
Period of square wave output from TOmn = Period of counter clock × (Set value of TDRmn + 1) × 2

Frequency of square wave output from TOmn = Frequency of counter clock/{(Set value of TDRmn + 1) × 2}

   Timer counter register mn (TCRmn) operates as a down counter in the interval timer mode. The TCRmn register loads the value of timer data register mn (TDRmn) at the first counter clock after any of the operation enable (start) trigger bits (TSmn, TSHm1, TSHm3) of timer channel start register m (TSm) is set to 1. If the MDmn0 bit of timer mode register mn (TMRmn) is 0 at this time, INTTMmn is not output and the output on TOmn is not toggled. If the MDmn0 bit of the TMRmn register is 1, INTTMmn is output and the output on TOmn is toggled. After that, the TCRmn register count down in synchronization with the counter clock. When TCRmn = 0000H, INTTMmn is output and the output on TOmn is toggled at the next counter clock. At the same time, the TCRmn register loads the value of the TDRmn register again. After that, the same operation is repeated. The TDRmn register can be rewritten at any time. The new value of the TDRmn register becomes valid from the next period.

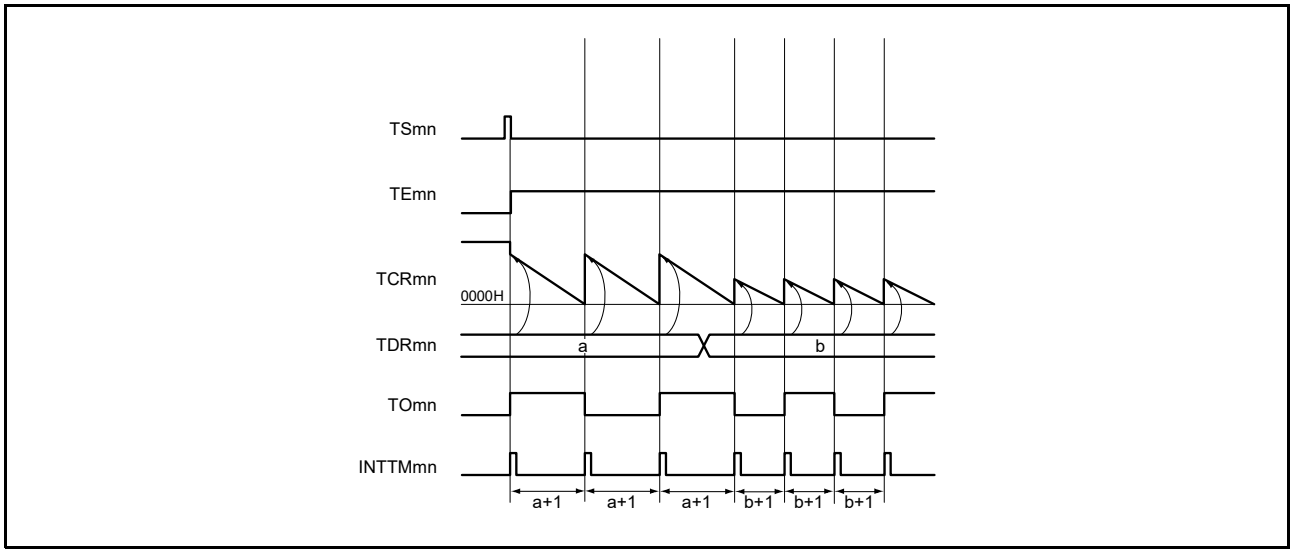
Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

Figure 7 - 42 Block Diagram for Operation as an Interval Timer or for Square Wave Output



Note For channels 1 and 3, the clock can be selected as CKm0, CKm1, CKm2, or CKm3.

Figure 7 - 43 Example of Basic Timing during Operation as an Interval Timer or for Square Wave Output (MDmn0 = 1)

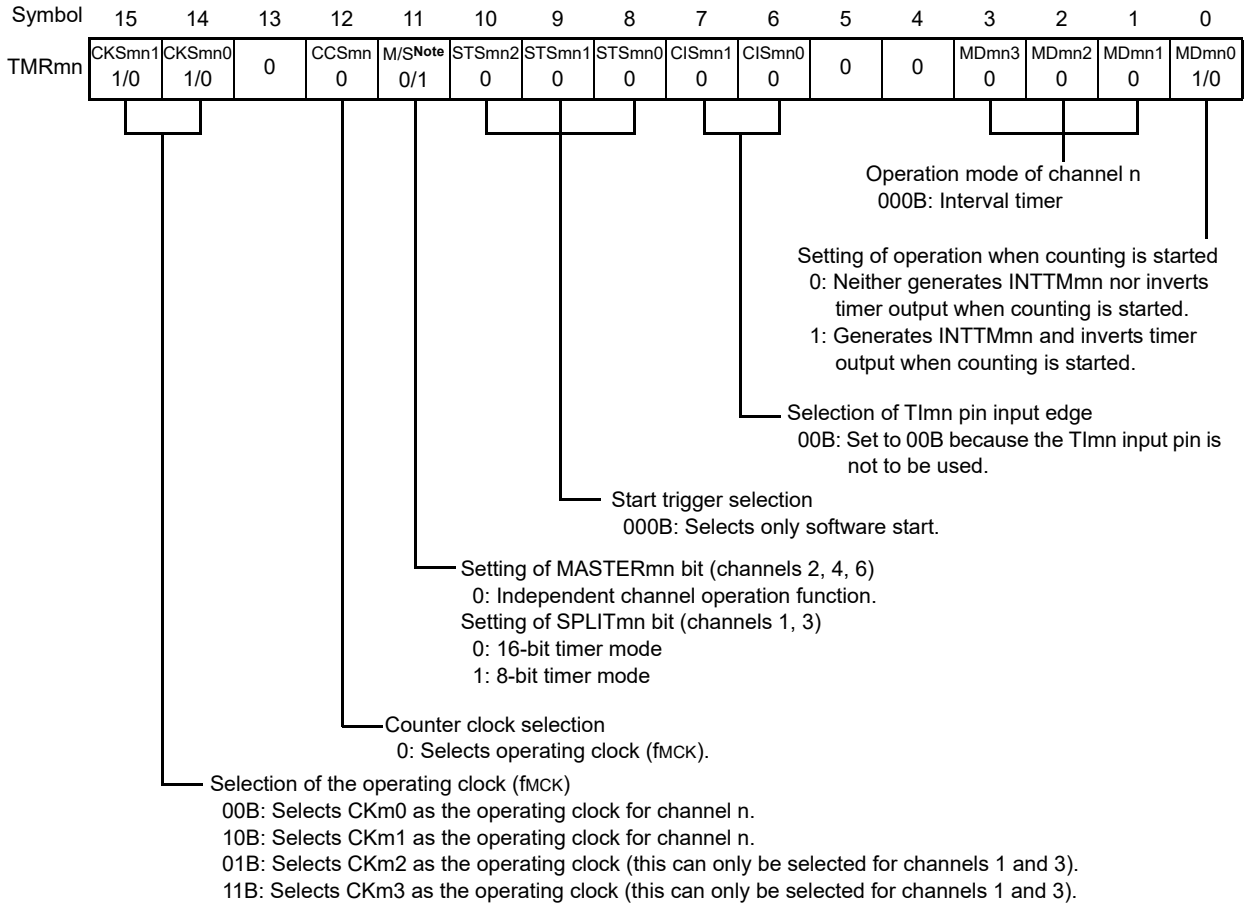


Remark 1. m: Unit number (m = 0), n: Channel number (n = 0 to 7)

Remark 2. TSmn: Bit n of timer channel start register m (TSm)
TEmn: Bit n of timer channel enable status register m (TEm)
TCRmn: Timer counter register mn (TCRmn)
TDRmn: Timer data register mn (TDRmn)
TOmn: TOmn pin output signal

Figure 7 - 44 Example of Register Settings for Operation as an Interval Timer or for Square Wave Output

a) Timer mode register mn (TMRmn)



b) Timer output register m (TOm)

	Bit n	
TOm	TOmn 1/0	0: Outputs 0 from TOmn. 1: Outputs 1 from TOmn.

c) Timer output enable register m (TOEm)

	Bit n	
TOEm	TOEmn 1/0	0: Stops the TOmn output operation by counting operation. 1: Enables the TOmn output operation by counting operation.

d) Timer output level register m (TOLm)

	Bit n	
TOLm	TOLmn 0	0: Set this bit to 0 when TOMmn = 0 (master channel output mode)

e) Timer output mode register m (TOMm)

	Bit n	
TOMm	TOMmn 0	0: Sets master channel output mode.

Note TMRm2, TMRm4, TMRm6: MASTERmn bit
TMRm1, TMRm3: SPLITmn bit
TMRm0, TMRm5, TMRm7: Fixed to 0

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

Figure 7 - 45 Procedure for Operations When the Interval Timer or Square Wave Output Function Is to Be Used

	Software Operation	Hardware State
TAU default setting		Power-off state (Clock supply is stopped and writing to each register is disabled.)
	Sets the TAUmEN bit of peripheral enable register 0 (PER0) to 1. →	Power-on state. Each channel stops operating. (Clock supply is started and writing to each register is enabled.)
	Sets timer clock select register m (TPSm). Determines clock frequencies of CKm0 to CKm3.	
Channel default setting	Sets timer mode register mn (TMRmn) (determines operation mode of channel). Sets interval (period) value to timer data register mn (TDRmn).	Channel stops operating. (Clock is supplied and some power is consumed.)
	To use the TOmn output Clears the TOMmn bit of timer output mode register m (TOMm) to 0 (master channel output mode). Clears the TOLmn bit to 0. Sets the TOmn bit and determines default level of the TOmn output. →	The TOmn pin goes into Hi-Z output state. The TOmn default setting level is output when the port mode register is in the output mode and the port register is 0.
	Sets the TOEmn bit to 1 and enables operation of TOmn. →	TOmn does not change because channel stops operating.
	Clears the port register and port mode register to 0. →	The TOmn pin outputs the TOmn set level.
Operation start	(Sets the TOEmn bit to 1 only if using TOmn output and resuming operation.). Sets the TSmn (TSHm1, TSHm3) bit to 1. → The TSmn (TSHm1, TSHm3) bit automatically returns to 0 because it is a trigger bit.	TEmn (TEHm1, TEHm3) = 1, and count operation starts. Value of the TDRmn register is loaded to timer counter register mn (TCRmn). INTTMmn is generated and TOmn performs toggle operation if the MDmn0 bit of the TMRmn register is 1.
During operation	Set values of the TMRmn register, TOMmn, and TOLmn bits cannot be changed. Set value of the TDRmn register can be changed. The TCRmn register can always be read. The TSRmn register is not used. Set values of the TOm and TOEm registers can be changed.	Counter (TCRmn) counts down. When count value reaches 0000H, the value of the TDRmn register is loaded to the TCRmn register again and the count operation is continued. By detecting TCRmn = 0000H, INTTMmn is generated and TOmn performs toggle operation. After that, the above operation is repeated.
Operation stop	The TTmn (TTHm1, TTHm3) bit is set to 1. → The TTmn (TTHm1, TTHm3) bit automatically returns to 0 because it is a trigger bit.	TEmn (TEHm1, TEHm3) = 0, and count operation stops. The TCRmn register holds count value and stops. The TOmn output is not initialized and retains its current state.
	The TOEmn bit is cleared to 0 and value is set to the TOmn bit. →	The TOmn pin outputs the TOmn bit set level.
TAU stop	To hold the TOmn pin output level Clears the TOmn bit to 0 after the value to be held is set to the port register. →	The TOmn pin output level is held by port function.
	When holding the TOmn pin output level is not necessary Setting not required.	
	The TAUmEN bit of the PER0 register is cleared to 0. → Set the TAUmRES bit of the PRR0 register to 1 to initialize all circuits of the timer array unit. →	This stops supply of the input clock to timer array unit m. All circuits are initialized and SFR of each channel is also initialized. (The TOmn bit is cleared to 0 and the TOmn pin is set to port mode.)

Operation is resumed.

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

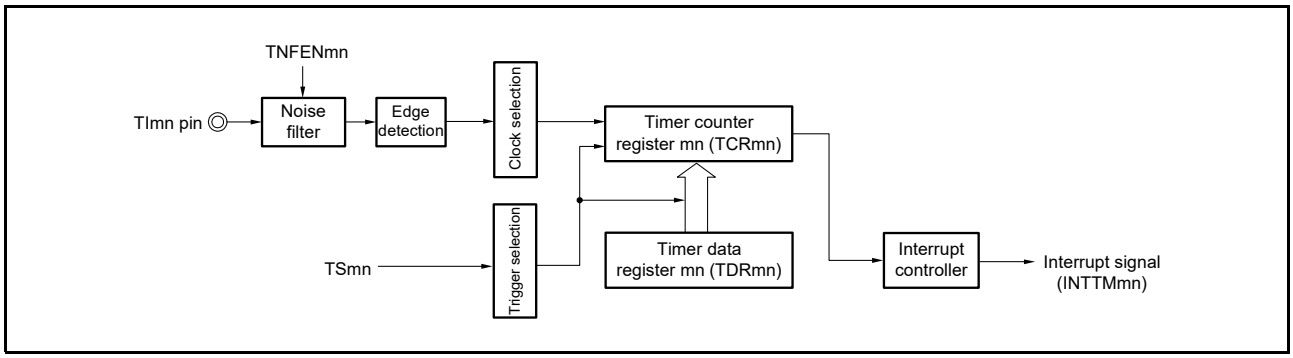
7.8.2 Operation as an external event counter

The timer array unit can be used as an external event counter that counts the number of times the valid input edge (external event) is detected in the TImn pin. When a specified count value is reached, the event counter generates an interrupt. The specified number of counts can be calculated by the following expression.

Specified number of counts = Set value of TDRmn + 1

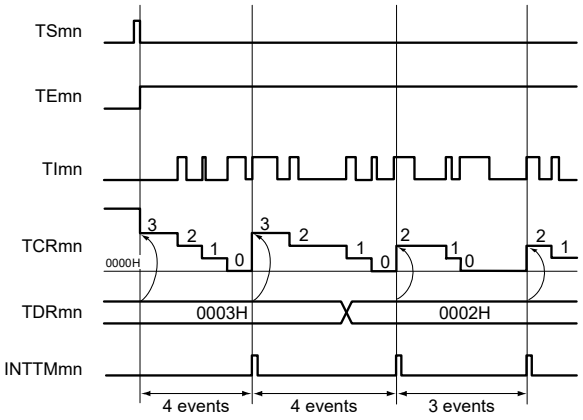
Timer counter register mn (TCRmn) operates as a down counter in the event counter mode. The TCRmn register loads the value of timer data register mn (TDRmn) by setting any operation enable (start) trigger bit (TSmn) of timer channel start register m (TSm) to 1. The TCRmn register counts down each time the valid input edge of the TImn pin has been detected. When TCRmn = 0000H, the TCRmn register loads the value of the TDRmn register again, and outputs INTTMmn. After that, the above operation is repeated. An irregular waveform that depends on external events is output from the TOmn pin. Stop the output by setting the TOEmn bit of timer output enable register m (TOEm) to 0. The TDRmn register can be rewritten at any time. The new value of the TDRmn register becomes valid during the next count period. Instead of using the TImn pin input, a channel specified for the external event counter function can also use the timer input selected in the TIS0 register as its input source to drive counting.

Figure 7 - 46 Block Diagram for Operation as an External Event Counter



Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

Figure 7 - 47 Example of Basic Timing during Operation as an External Event Counter

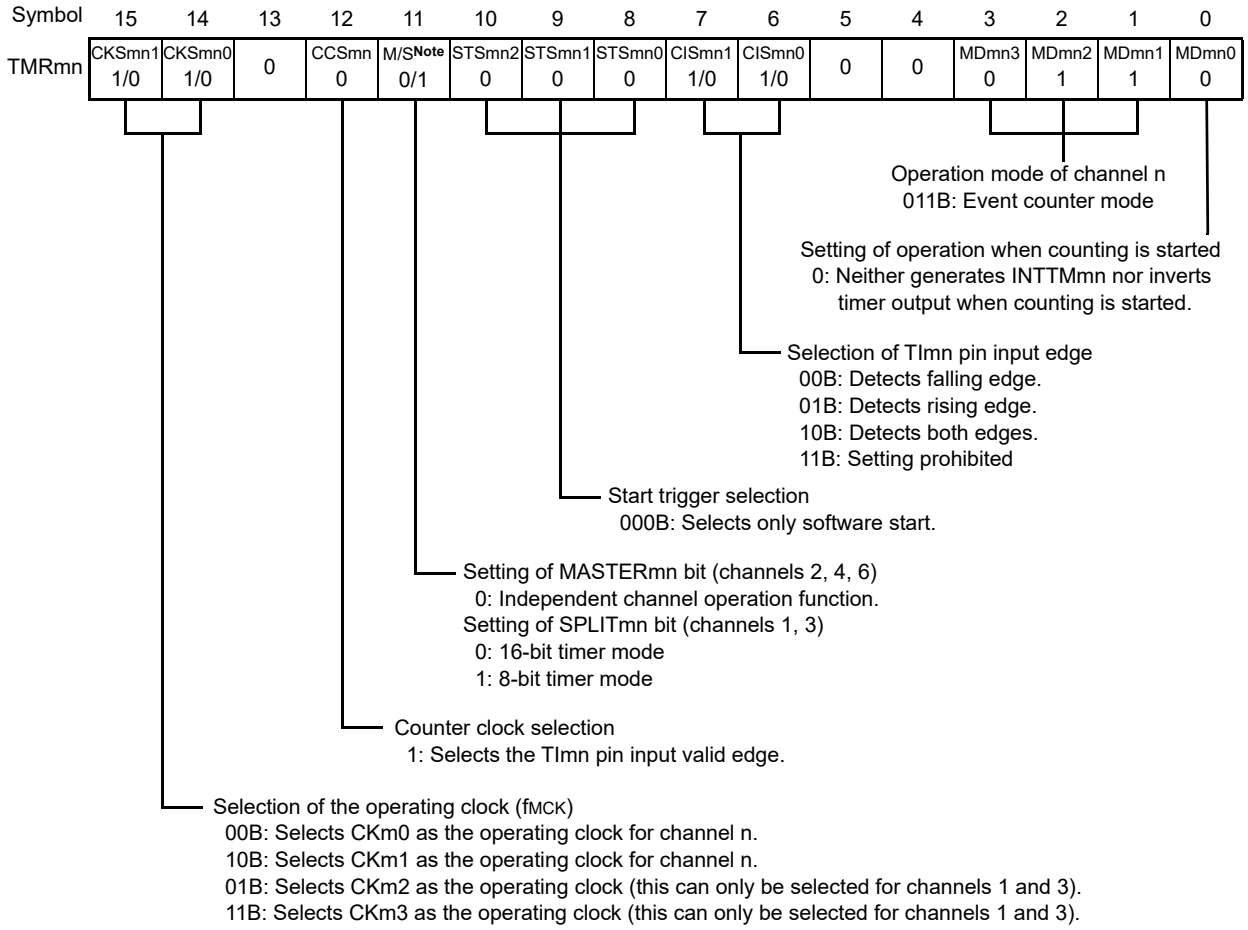


Remark 1. m: Unit number (m = 0), n: Channel number (n = 0 to 7)

Remark 2. TSmn: Bit n of timer channel start register m (TSm)
TEmn: Bit n of timer channel enable status register m (TEm)
TImn: TImn pin input signal
TCRmn: Timer counter register mn (TCRmn)
TDRmn: Timer data register mn (TDRmn)

Figure 7 - 48 Example of Register Settings in External Event Counter Mode

a) Timer mode register mn (TMRmn)



b) Timer output register m (TOm)

	Bit n	
TOm	TOmn 0	0: Outputs 0 from TOmn.

c) Timer output enable register m (TOEm)

	Bit n	
TOEm	TOEmn 0	0: Stops the TOmn output operation by counting operation.

d) Timer output level register m (TOLm)

	Bit n	
TOLm	TOLmn 0	0: Set this bit to 0 when TOMmn = 0 (master channel output mode).

e) Timer output mode register m (TOMm)

	Bit n	
TOMm	TOMmn 0	0: Sets master channel output mode.

Note TMRm2, TMRm4, TMRm6: MASTERmn bit
TMRm1, TMRm3: SPLITmn bit
TMRm0, TMRm5, TMRm7: Fixed to 0

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

Figure 7 - 49 Procedure for Operations When the External Event Counter Function Is to Be Used

	Software Operation	Hardware State
TAU default setting		Power-off state (Clock supply is stopped and writing to each register is disabled.)
	Sets the TAUmEN bit of peripheral enable register 0 (PER0) to 1. →	Power-on state. Each channel stops operating. (Clock supply is started and writing to each register is enabled.)
	Sets timer clock select register m (TPSm). Determines clock frequencies of CKm0 to CKm3.	
Channel default setting	Sets the corresponding bit of the noise filter enable register 1 (NFEN1) to 0 (off) or 1 (on). Sets timer mode register mn (TMRmn) (determines operation mode of channel). Sets number of counts to timer data register mn (TDRmn). Clears the TOEmn bit of timer output enable register m (TOEm) to 0.	Channel stops operating. (Clock is supplied and some power is consumed.)
Operation start	Sets the TSmn bit to 1. → The TSmn bit automatically returns to 0 because it is a trigger bit.	TEmn = 1, and count operation starts. Value of the TDRmn register is loaded to timer counter register mn (TCRmn) and detection of the TImn pin input edge is awaited.
During operation	Set value of the TDRmn register can be changed. The TCRmn register can always be read. The TSRmn register is not used. Set values of the TMRmn register, TOMmn, TOLmn, TOmn, and TOEmn bits cannot be changed.	Counter (TCRmn) counts down each time input edge of the TImn pin has been detected. When count value reaches 0000H, the value of the TDRmn register is loaded to the TCRmn register again, and the count operation is continued. By detecting TCRmn = 0000H, the INTTMmn output is generated. After that, the above operation is repeated.
Operation stop	The TTmn bit is set to 1. → The TTmn bit automatically returns to 0 because it is a trigger bit.	TEmn = 0, and count operation stops. The TCRmn register holds count value and stops.
TAU stop	The TAUmEN bit of the PER0 register is cleared to 0. → Set the TAUmRES bit of the PRR0 register to 1 to initialize all circuits of the timer array unit. →	This stops supply of the input clock to timer array unit m. All circuits are initialized and SFR of each channel is also initialized.

Operation is resumed

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

7.8.3 Operation as a frequency divider (only for channel 0)

The timer array unit can be used as a frequency divider that divides a clock input to the TI00 pin and outputs the result from the TO00 pin. The divided clock frequency output from TO00 can be calculated by the following expression.

- When rising edge/falling edge is selected:
  Divided clock frequency = Input clock frequency/{(Set value of TDR00 + 1) × 2}
- When both edges are selected:
  Divided clock frequency ≈ Input clock frequency/(Set value of TDR00 + 1)

Timer counter register 00 (TCR00) operates as a down counter in the interval timer mode. After the operation enable (start) trigger bit (TS00) of timer channel start register 0 (TS0) is set to 1, the TCR00 register loads the value of timer data register 00 (TDR00) when the TI00 valid edge is detected. If the MD000 bit of timer mode register 00 (TMR00) is 0 at this time, INTTM00 is not output and the output on TO00 is not toggled. If the MD000 bit of timer mode register 00 (TMR00) is 1, INTTM00 is output and the output on TO00 is toggled. After that, the TCR00 register counts down at the valid edge of the TI00 pin. When TCR00 = 0000H, it toggles the output on TO00. At the same time, the TCR00 register loads the value of the TDR00 register again, and continues counting. If detection of both the edges of the TI00 pin is selected, the duty factor error of the input clock affects the divided clock period of the TO00 output. The period of the TO00 output clock includes a sampling error of one period of the operating clock.

Clock period of TO00 output = Ideal TO00 output clock period ± Operating clock period (error)

The TDR00 register can be rewritten at any time. The new value of the TDR00 register becomes valid during the next count period.

Figure 7 - 50 Block Diagram for Operation as a Frequency Divider

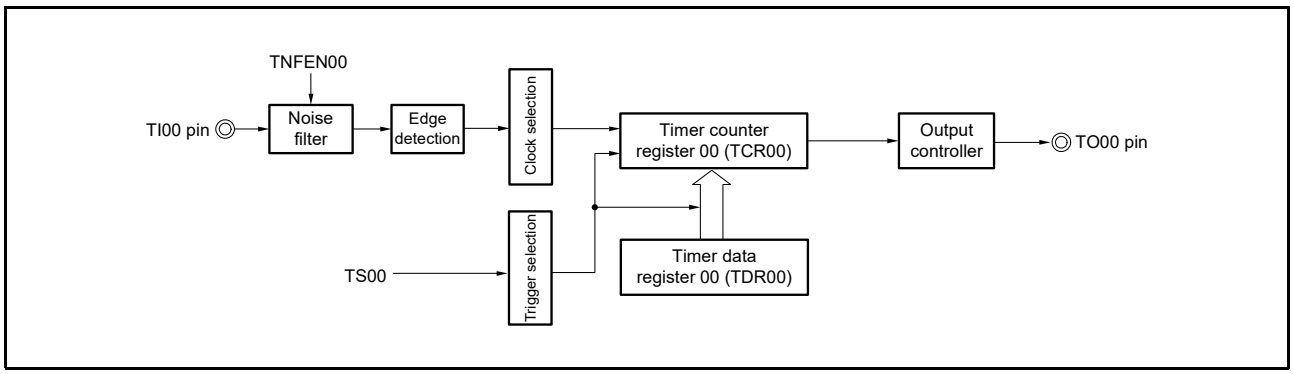
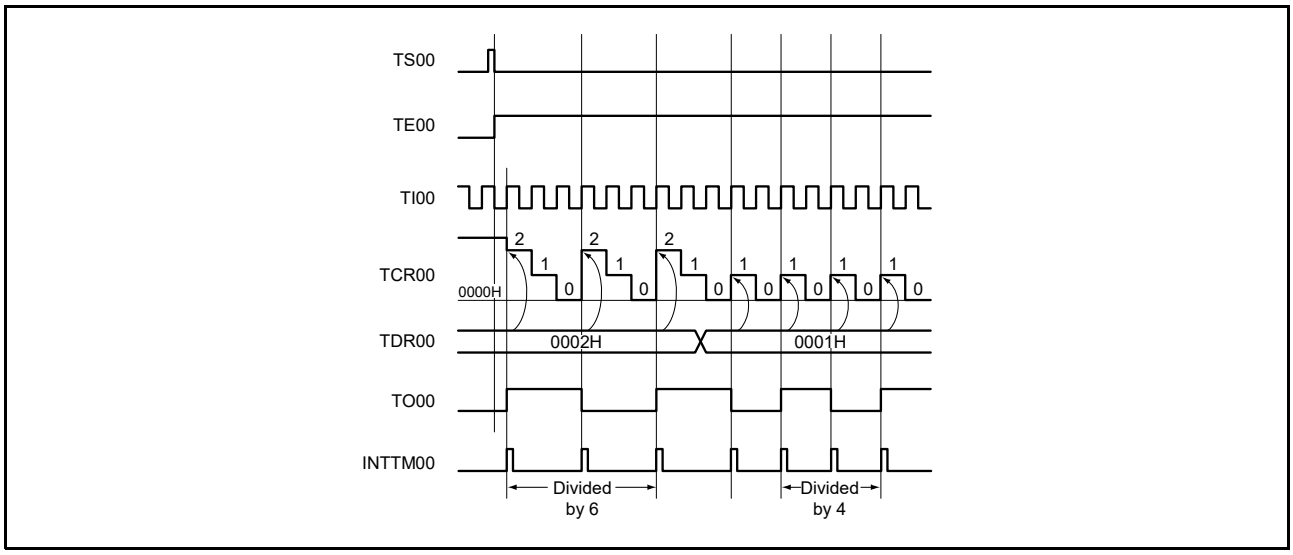


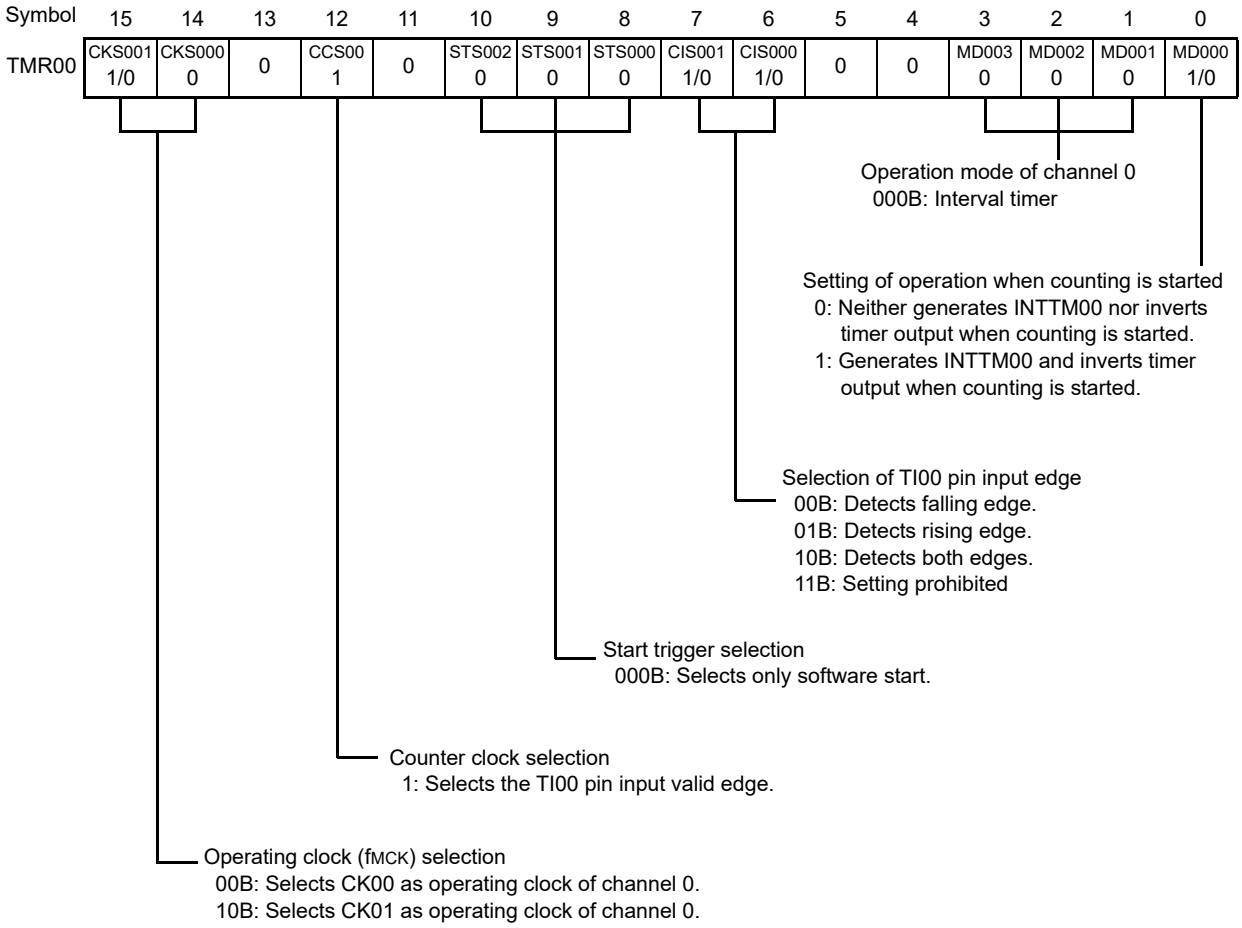
Figure 7 - 51 Example of Basic Timing during Operation as a Frequency Divider (MD000 = 1)



Remark TS00: Bit 0 of timer channel start register 0 (TS0)
TE00: Bit 0 of timer channel enable status register 0 (TE0)
TI00: TI00 pin input signal
TCR00: Timer counter register 00 (TCR00)
TDR00: Timer data register 00 (TDR00)
TO00: TO00 pin output signal

Figure 7 - 52 Example of Register Settings for Operation as a Frequency Divider

a) Timer mode register 00 (TMR00)



b) Timer output register 0 (TO0)

	Bit 0	
TO0	TO00 1/0	0: Outputs 0 from TO00. 1: Outputs 1 from TO00.

c) Timer output enable register 0 (TOE0)

	Bit 0	
TOE0	TOE00 1/0	0: Stops the TO00 output operation by counting operation. 1: Enables the TO00 output operation by counting operation.

d) Timer output level register 0 (TOL0)

	Bit 0	
TOL0	TOL00 0	0: Set this bit to 0 when master channel output mode (TOM00 = 0)

e) Timer output mode register 0 (TOM0)

	Bit 0	
TOM0	TOM00 0	0: Sets master channel output mode.

Figure 7 - 53 Procedure for Operations When the Frequency Divider Function Is to Be Used (1/2)

	Software Operation	Hardware State
TAU default setting		Power-off state (Clock supply is stopped and writing to each register is disabled.)
	Sets the TAU0EN bit of peripheral enable register 0 (PER0) to 1. →	Power-on state. Each channel stops operating. (Clock supply is started and writing to each register is enabled.)
	Sets timer clock select register 0 (TPS0). Determines clock frequencies of CK00 to CK03.	
Channel default setting	Sets the corresponding bit of the noise filter enable register 1 (NFEN1) to 0 (off) or 1 (on). Sets timer mode register 0n (TMR0n) (determines operation mode of channel and selects the detection edge). Sets interval (period) value to timer data register 00 (TDR00).	Channel stops operating. (Clock is supplied and some power is consumed.)
	Clears the TOM00 bit of timer output mode register 0 (TOM0) to 0 (master channel output mode). Clears the TOL00 bit to 0. Sets the TO00 bit and determines default level of the TO00 output. →	The TO00 pin goes into Hi-Z output state. The TO00 default setting level is output when the port mode register is in output mode and the port register is 0.
	Sets the TOE00 bit to 1 and enables operation of TO00. →	TO00 does not change because channel stops operating.
	Clears the port register and port mode register to 0. →	The TO00 pin outputs the TO00 set level.

Figure 7 - 53 Procedure for Operations When the Frequency Divider Function Is to Be Used (2/2)

	Software Operation	Hardware State
Operation start	Sets the TOE00 bit to 1 (only when operation is resumed). Sets the TS00 bit to 1. The TS00 bit automatically returns to 0 because it is a trigger bit.	TE00 = 1, and count operation starts. Value of the TDR00 register is loaded to timer counter register 00 (TCR00). INTTM00 is generated and TO00 performs toggle operation if the MD000 bit of the TMR00 register is 1.
During operation	Set value of the TDR00 register can be changed. The TCR00 register can always be read. The TSR00 register is not used. Set values of the TO0 and TOE0 registers can be changed. Set values of the TMR00 register, TOM00, and TOL00 bits cannot be changed.	Counter (TCR00) counts down. When count value reaches 0000H, the value of the TDR00 register is loaded to the TCR00 register again, and the count operation is continued. By detecting TCR00 = 0000H, INTTM00 is generated and TO00 performs toggle operation. After that, the above operation is repeated.
Operation stop	The TT00 bit is set to 1. The TT00 bit automatically returns to 0 because it is a trigger bit.	TE00 = 0, and count operation stops. The TCR00 register holds count value and stops. The TO00 output is not initialized and retains its current state.
	The TOE00 bit is cleared to 0 and value is set to the TO00 bit.	The TO00 pin outputs the TO00 set level.
TAU stop	To hold the TO00 pin output level Clears the TO00 bit to 0 after the value to be held is set to the port register. When holding the TO00 pin output level is not necessary Setting not required.	The TO00 pin output level is held by port function.
	The TAU0EN bit of the PER0 register is cleared to 0. Set the TAUmRES bit of the PRR0 register to 1 to initialize all circuits of the timer array unit.	This stops supply of the input clock to timer array unit m. All circuits are initialized and SFR of each channel is also initialized. (The TO00 bit is cleared to 0 and the TO00 pin is set to port mode).

Operation is resumed

7.8.4 Operation for input pulse interval measurement

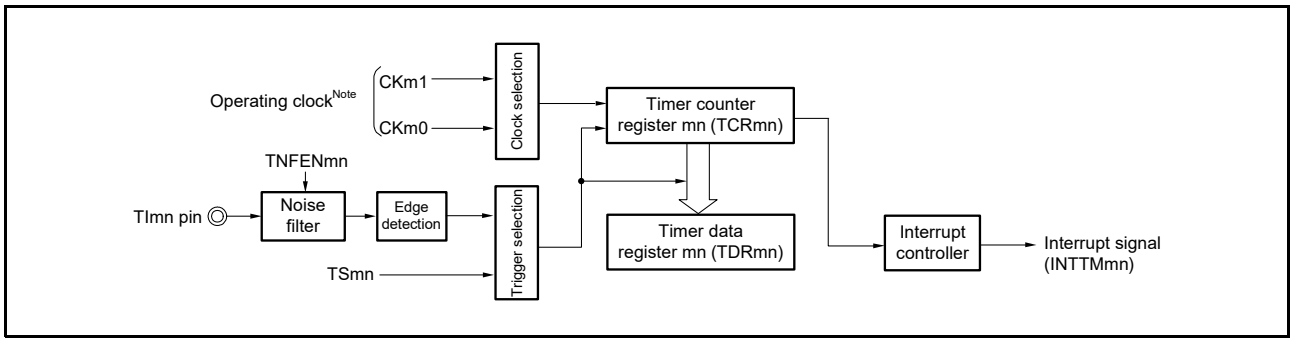
The count value can be captured at the TImn valid edge and the interval of the pulse input to TImn can be measured. In addition, the count value can be captured by using software operation (TSmn = 1) as a capture trigger while the TEmn bit is set to 1. The pulse interval can be calculated by the following expression.

TImn input pulse interval = Period of counter clock × ((10000H × TSRmn.OVF) + (Captured value of TDRmn + 1))

Caution The TImn pin input is sampled using the operating clock selected with the CKSmn bit of timer mode register mn (TMRmn), so an error equivalent to one cycle of the operating clock occurs.

Timer counter register mn (TCRmn) operates as an up counter in the capture mode. When the operation enable (start) trigger bit (TSmn) of timer channel start register m (TSm) is set to 1, the TCRmn register counts up from 0000H in synchronization with the counter clock. When the TImn pin input valid edge is detected, the count value of the TCRmn register is transferred (captured) to timer data register mn (TDRmn) and, at the same time, the TCRmn register is cleared to 0000H, and the INTTMmn is output. If the counter overflows at this time, the OVF bit of timer status register mn (TSRmn) is set to 1. If the counter does not overflow, the OVF bit is cleared. After that, the above operation is repeated. As soon as the count value has been captured to the TDRmn register, the OVF bit of the TSRmn register is updated depending on whether the counter overflows during the measurement period. Therefore, the overflow state of the captured value can be checked. If the counter reaches a full count for two or more periods, it is judged to be an overflow occurrence, and the OVF bit of the TSRmn register is set to 1. However, a normal interval value cannot be measured for the OVF bit, if two or more overflows occur. Set the STSmn2 to STSmn0 bits of the TMRmn register to 001B to use the valid edges of TImn as a start trigger and a capture trigger. Instead of using the TImn pin input, an input pulse interval can also be measured by using the timer input selected in the TIS0 register or the software operation (TSmn = 1) as a start trigger and a capture trigger.

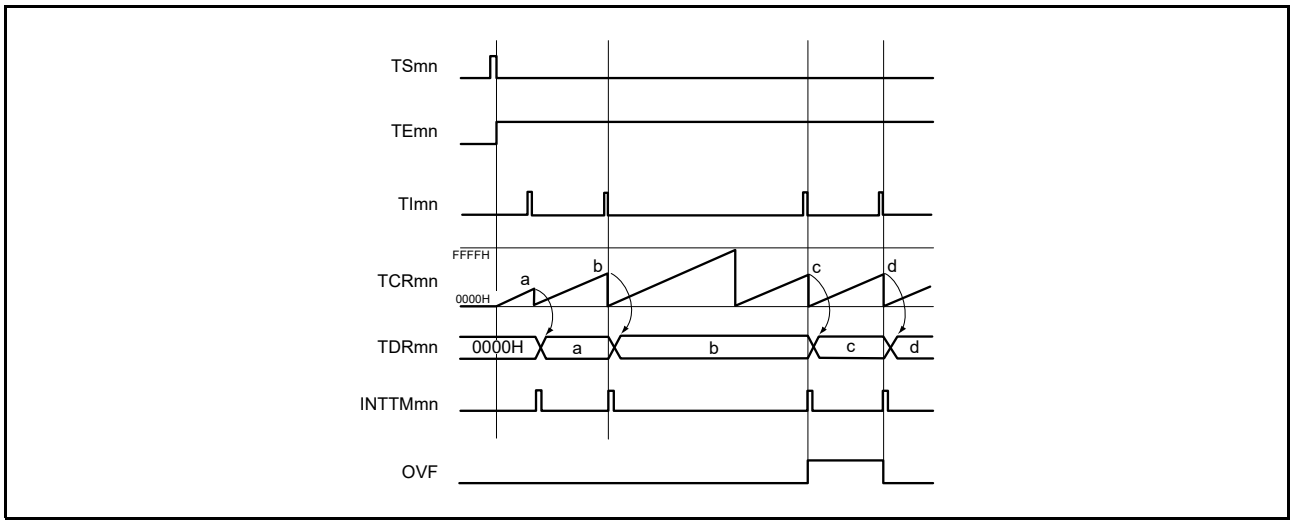
Figure 7 - 54 Block Diagram for Operation for Input Pulse Interval Measurement



Note For channels 1 and 3, the clock can be selected as CKm0, CKm1, CKm2, or CKm3.

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

Figure 7 - 55 Example of Basic Timing during Operation for Input Pulse Interval Measurement (MDmn0 = 0)

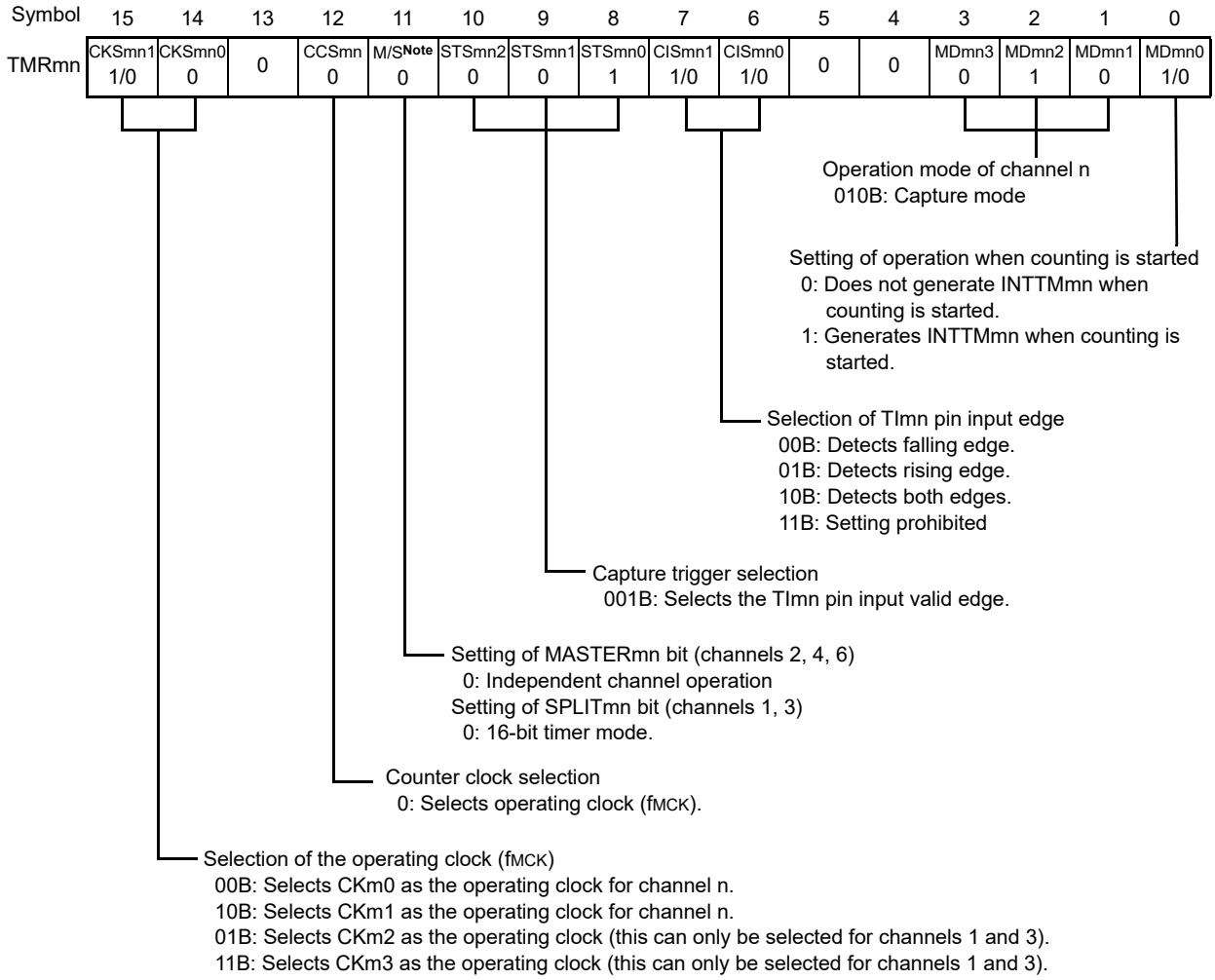


Remark 1. m: Unit number (m = 0), n: Channel number (n = 0 to 7)

Remark 2. TSmn: Bit n of timer channel start register m (TSm)
TEmn: Bit n of timer channel enable status register m (TEm)
TImn: TImn pi n input signal
TCRmn: Timer counter register mn (TCRmn)
TDRmn: Timer data register mn (TDRmn)
OVF: Bit 0 of timer status register mn (TSRmn)

Figure 7 - 56 Example of Register Settings for Operation for Input Pulse Interval Measurement

a) Timer mode register mn (TMRmn)



b) Timer output register m (TOm)

	Bit n	
TOm	TOmn 0	0: Outputs 0 from TOmn.

c) Timer output enable register m (TOEm)

	Bit n	
TOEm	TOEmn 0	0: Stops TOmn output operation by counting operation.

d) Timer output level register m (TOLm)

	Bit n	
TOLm	TOLmn 0	0: Set this bit to 0 when TOMmn = 0 (master channel output mode).

e) Timer output mode register m (TOMm)

	Bit n	
TOMm	TOMmn 0	0: Sets master channel output mode.

(Note and Remark are listed on the next page.)

Note TMRm2, TMRm4, TMRm6: MASTERmn bit
TMRm1, TMRm3: SPLITmn bit
TMRm0, TMRm5, TMRm7: Fixed to 0

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

Figure 7 - 57 Procedure for Operations When the Input Pulse Interval Measurement Function Is to Be Used

	Software Operation	Hardware State
TAU default setting		Power-off state (Clock supply is stopped and writing to each register is disabled.)
	Sets the TAUmEN bit of peripheral enable register 0 (PER0) to 1. →	Power-on state. Each channel stops operating. (Clock supply is started and writing to each register is enabled.)
	Sets timer clock select register m (TPSm). Determines clock frequencies of CKm0 to CKm3.	
Channel default setting	Sets the corresponding bit of the noise filter enable register 1 (NFEN1) to 0 (off) or 1 (on). Sets timer mode register mn (TMRmn) (determines operation mode of channel).	Channel stops operating. (Clock is supplied and some power is consumed.)
Operation start	Sets TSmn bit to 1. → The TSmn bit automatically returns to 0 because it is a trigger bit.	TEmn = 1, and count operation starts. Timer counter register mn (TCRmn) is cleared to 0000H. When the MDmn0 bit of the TMRmn register is 1, INTTMmn is generated.
During operation	Set values of only the CISmn1 and CISmn0 bits of the TMRmn register can be changed. The TDRmn register can always be read. The TCRmn register can always be read. The TSRmn register can always be read. Set values of the TOMmn, TOLmn, TOmn, and TOEmn bits cannot be changed.	Counter (TCRmn) counts up from 0000H. When the valid edge of the TImn pin input is detected or the TSmn bit is set to 1, the count value is transferred (captured) to timer data register mn (TDRmn). At the same time, the TCRmn register is cleared to 0000H, and the INTTMmn signal is generated. If an overflow occurs at this time, the OVF bit of timer status register mn (TSRmn) is set; if an overflow does not occur, the OVF bit is cleared. After that, the above operation is repeated.
Operation stop	The TTmn bit is set to 1. → The TTmn bit automatically returns to 0 because it is a trigger bit.	TEmn = 0, and count operation stops. The TCRmn register holds count value and stops. The OVF bit of the TSRmn register is also held.
TAU stop	The TAUmEN bit of the PER0 register is cleared to 0. → Set the TAUmRES bit of the PRR0 register to 1 to initialize all circuits of the timer array unit. →	This stops supply of the input clock to timer array unit m. All circuits are initialized and SFR of each channel is also initialized.

Operation is resumed.

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

7.8.5 Operation for input signal low- and high-level width measurement

Caution When using a channel to implement the LIN bus, set the ISC1 bit of the input switch control register (ISC) to 1. In the following descriptions, read TImn as RxD2.

By starting counting at one edge of the TImn pin input and capturing the number of counts at another edge, the signal width (low and high levels) of TImn can be measured. The signal width of TImn can be calculated by the following expression.

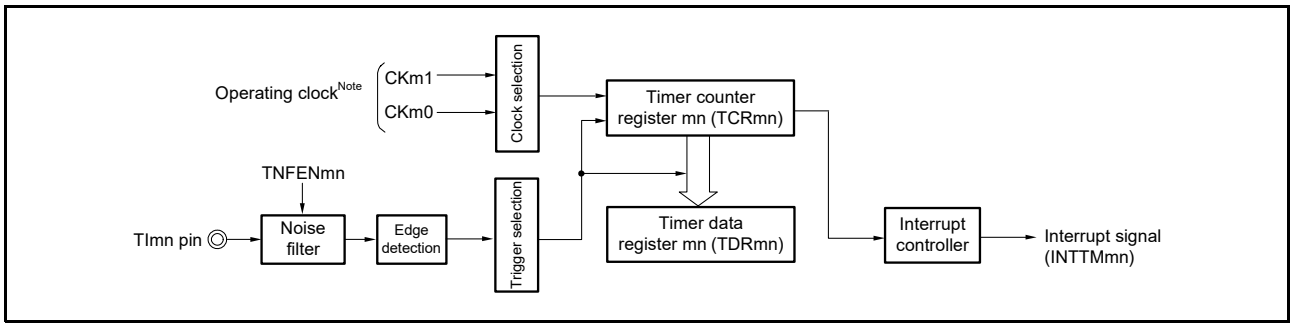
Signal width of TImn input = Period of counter clock × ((10000H × TSRmn.OVF) + (Captured value of TDRmn + 1))

Caution The TImn pin input is sampled using the operating clock selected with the CKSmn bit of timer mode register mn (TMRmn), so an error equivalent to one cycle of the operating clock occurs.

Timer counter register mn (TCRmn) operates as an up counter in the capture & one-count mode. When the operation enable (start) trigger bit (TSmn) of timer channel start register m (TSm) is set to 1, the TEmn bit is set to 1 and the TImn pin start edge detection wait state is set. When the TImn pin input start edge (rising edge of the TImn pin input when the high-level width is to be measured) is detected, the counter counts up from 0000H in synchronization with the counter clock. When the valid capture edge (falling edge of the TImn pin input when the high-level width is to be measured) is detected later, the count value is transferred to timer data register mn (TDRmn) and, at the same time, INTTMmn is output. If the counter overflows at this time, the OVF bit of timer status register mn (TSRmn) is set to 1. If the counter does not overflow, the OVF bit is cleared. The TCRmn register stops at the value “value transferred to the TDRmn register + 1”, and the TImn pin start edge detection wait state is set. After that, the above operation is repeated. As soon as the count value has been captured to the TDRmn register, the OVF bit of the TSRmn register is updated depending on whether the counter overflows during the measurement period. Therefore, the overflow state of the captured value can be checked. If the counter reaches a full count for two or more periods, it is judged to be an overflow occurrence, and the OVF bit of the TSRmn register is set to 1. However, a normal interval value cannot be measured for the OVF bit, if two or more overflows occur. Whether the low-level width or high-level width of the TImn pin is to be measured can be selected by using the CISmn1 and CISmn0 bits of the TMRmn register. Because this function is used to measure the signal width of the TImn pin input, the TSmn bit cannot be set to 1 while the TEmn bit is 1. Instead of the TImn pin input, the timer input selected in the TIS0 register can also be used as a start edge and a capture edge.

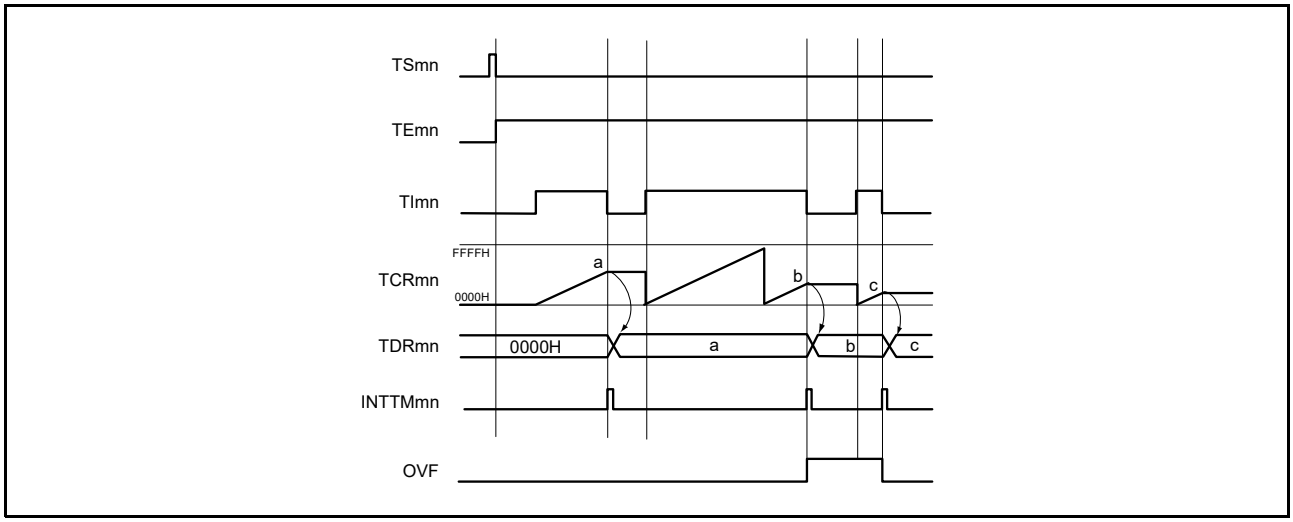
- CISmn1 and CISmn0 bits of the TMRmn register = 10B: Low-level width is measured.
- CISmn1 and CISmn0 bits of the TMRmn register = 11B: High-level width is measured.

Figure 7 - 58 Block Diagram for Operation for Input Signal Low- and High-level Width Measurement



Note For channels 1 and 3, the clock can be selected as CKm0, CKm1, CKm2, or CKm3.

Figure 7 - 59 Example of Basic Timing during Operation for Input Signal Low- and High-level Width Measurement

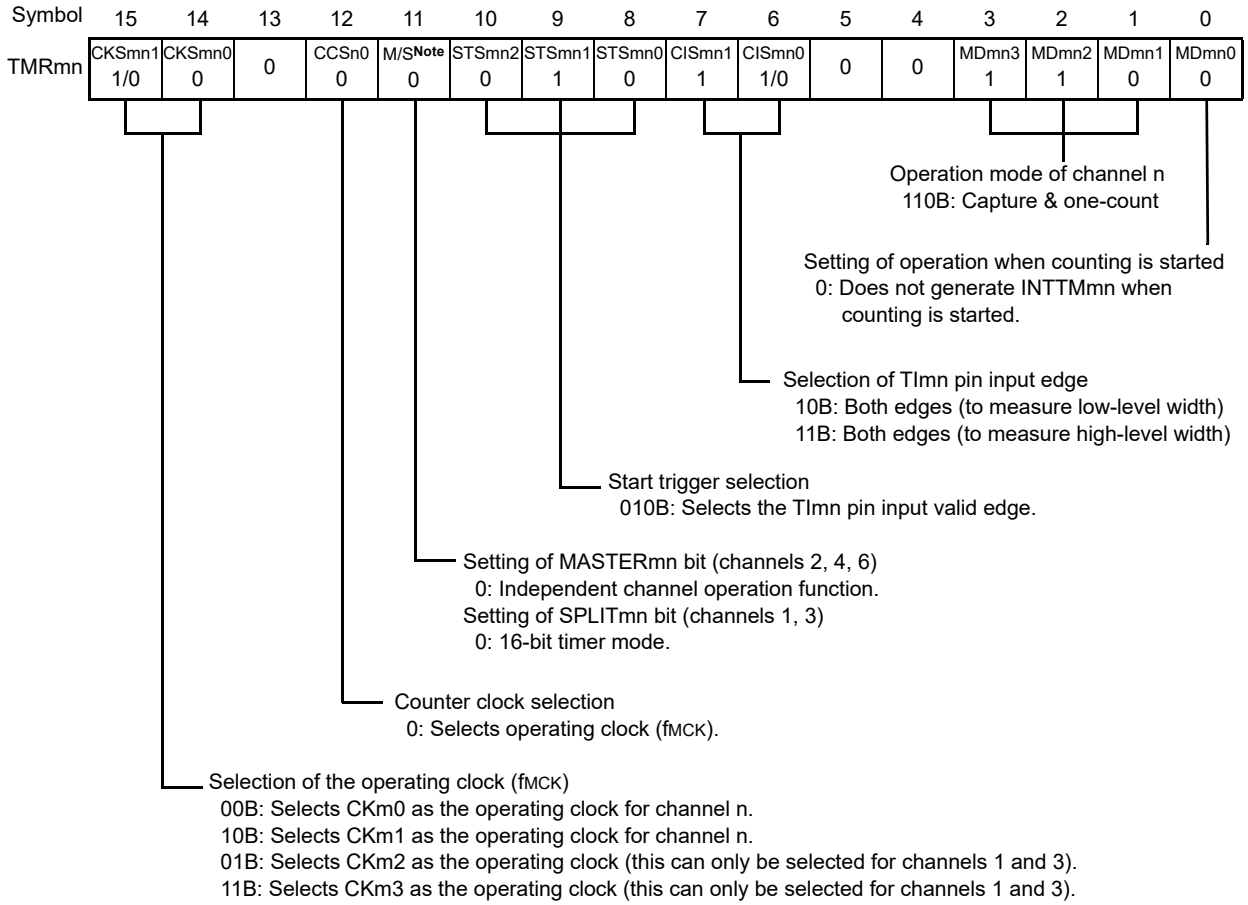


Remark 1. m: Unit number (m = 0), n: Channel number (n = 0 to 7)

Remark 2. TSmn: Bit n of timer channel start register m (TSm)
TEmn: Bit n of timer channel enable status register m (TEm)
TImn: TImn pin input signal
TCRmn: Timer counter register mn (TCRmn)
TDRmn: Timer data register mn (TDRmn)
OVF: Bit 0 of timer status register mn (TSRmn)

Figure 7 - 60 Example of Register Settings for Operation for Input Signal Low- and High-level Width Measurement

a) Timer mode register mn (TMRmn)



b) Timer output register m (TOm)

	Bit n	
TOm	TOmn 0	0: Outputs 0 from TOmn.

c) Timer output enable register m (TOEm)

	Bit n	
TOEm	TOEmn 0	0: Stops the TOmn output operation by counting operation.

d) Timer output level register m (TOLm)

	Bit n	
TOLm	TOLmn 0	0: Set this bit to 0 when TOMmn = 0 (master channel output mode).

e) Timer output mode register m (TOMm)

	Bit n	
TOMm	TOMmn 0	0: Sets master channel output mode.

Note TMRm2, TMRm4, TMRm6: MASTERmn bit
TMRm1, TMRm3: SPLITmn bit
TMRm0, TMRm5, TMRm7: Fixed to 0

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

Figure 7 - 61 Procedure for Operations When the Input Signal Low- and High-level Width Measurement Function Is to Be Used

	Software Operation	Hardware State
TAU default setting		Power-off state (Clock supply is stopped and writing to each register is disabled.)
	Sets the TAUmEN bit of peripheral enable register 0 (PER0) to 1. →	Power-on state. Each channel stops operating. (Clock supply is started and writing to each register is enabled.)
	Sets timer clock select register m (TPSm). Determines clock frequencies of CKm0 to CKm3.	
Channel default setting	Sets the corresponding bit of the noise filter enable register 1 (NFEN1) to 0 (off) or 1 (on). Sets timer mode register mn (TMRmn) (determines operation mode of channel). Clears the TOEmn bit to 0 and stops operation of TOmn.	Channel stops operating. (Clock is supplied and some power is consumed.)
Operation start	Sets the TSmn bit to 1. → The TSmn bit automatically returns to 0 because it is a trigger bit.	TEmn = 1, and the TImn pin start edge detection wait state is set.
	Detects the TImn pin input count start valid edge. →	Clears timer counter register mn (TCRmn) to 0000H and starts counting up.
During operation	Set value of the TDRmn register can always be read. The TCRmn register can always be read. The TSRmn register can always be read. Set values of the TMRmn register, TOMmn, TOLmn, TOmn, and TOEmn bits cannot be changed.	When the TImn pin start edge is detected, the counter (TCRmn) counts up from 0000H. If a capture edge of the TImn pin is detected, the count value is transferred to timer data register mn (TDRmn) and INTTMmn is generated. If an overflow occurs at this time, the OVF bit of timer status register mn (TSRmn) is set; if an overflow does not occur, the OVF bit is cleared. The TCRmn register stops the count operation until the next TImn pin start edge is detected.
Operation stop	The TTmn bit is set to 1. → The TTmn bit automatically returns to 0 because it is a trigger bit.	TEmn = 0, and count operation stops. The TCRmn register holds count value and stops. The OVF bit of the TSRmn register is also held.
TAU stop	The TAUmEN bit of the PER0 register is cleared to 0. → Set the TAUmRES bit of the PRR0 register to 1 to initialize all circuits of the timer array unit. →	This stops supply of the input clock to timer array unit m. All circuits are initialized and SFR of each channel is also initialized.

Operation is resumed.

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

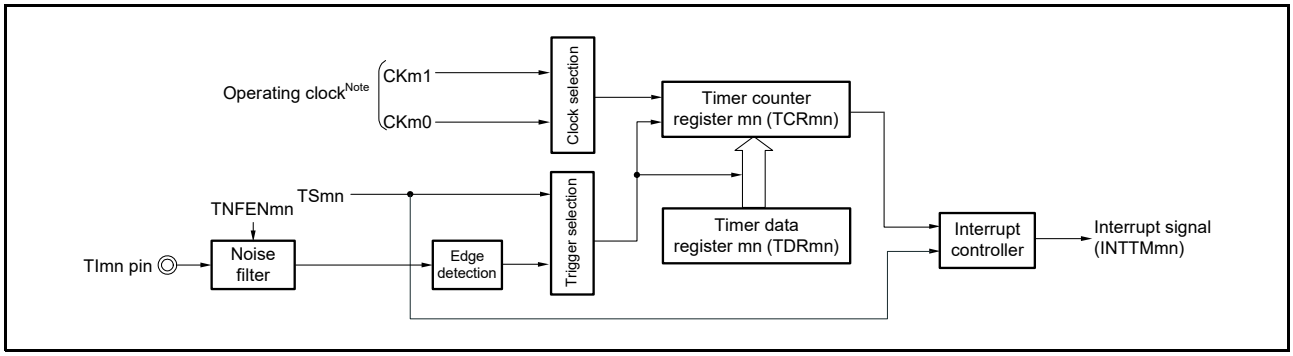
7.8.6 Operation as a delay counter

It is possible to start counting down when the valid edge of the TImn pin input is detected (an external event), and then generate INTTMmn (a timer interrupt) after any specified interval. It is also possible to start counting down and generate INTTMmn (timer interrupt) at any interval by setting TSmn to 1 by software while TEmn = 1. The interrupt generation period can be calculated by the following expression.

Generation period of INTTMmn (timer interrupt) = Period of counter clock × (Set value of TDRmn + 1)

Timer counter register mn (TCRmn) operates as a down counter in the one-count mode. When the operation enable (start) trigger bit (TSmn, TSHm1, TSHm3) of timer channel start register m (TSm) is set to 1, the TEmn, TEHm1, TEHm3 bits are set to 1 and the TImn pin input valid edge detection wait state is set. Timer counter register mn (TCRmn) starts operating upon TImn pin input valid edge detection and loads the value of timer data register mn (TDRmn). The TCRmn register counts down from the value of the TDRmn register it has loaded, in synchronization with the counter clock. When TCRmn = 0000H, it outputs INTTMmn and stops counting until the next TImn pin input valid edge is detected. The TDRmn register can be rewritten at any time. The new value of the TDRmn register becomes valid from the next period. Instead of using the TImn pin input, a channel specified for the delay counter function can also use the timer input selected in the TIS0 register or the software operation (TSmn = 1) as a start trigger for the function.

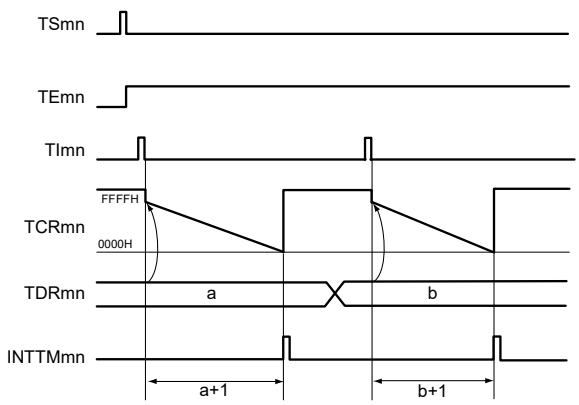
Figure 7 - 62 Block Diagram for Operation as a Delay Counter



Note For channels 1 and 3, the clock can be selected as CKm0, CKm1, CKm2, or CKm3.

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

Figure 7 - 63 Example of Basic Timing during Operation as a Delay Counter

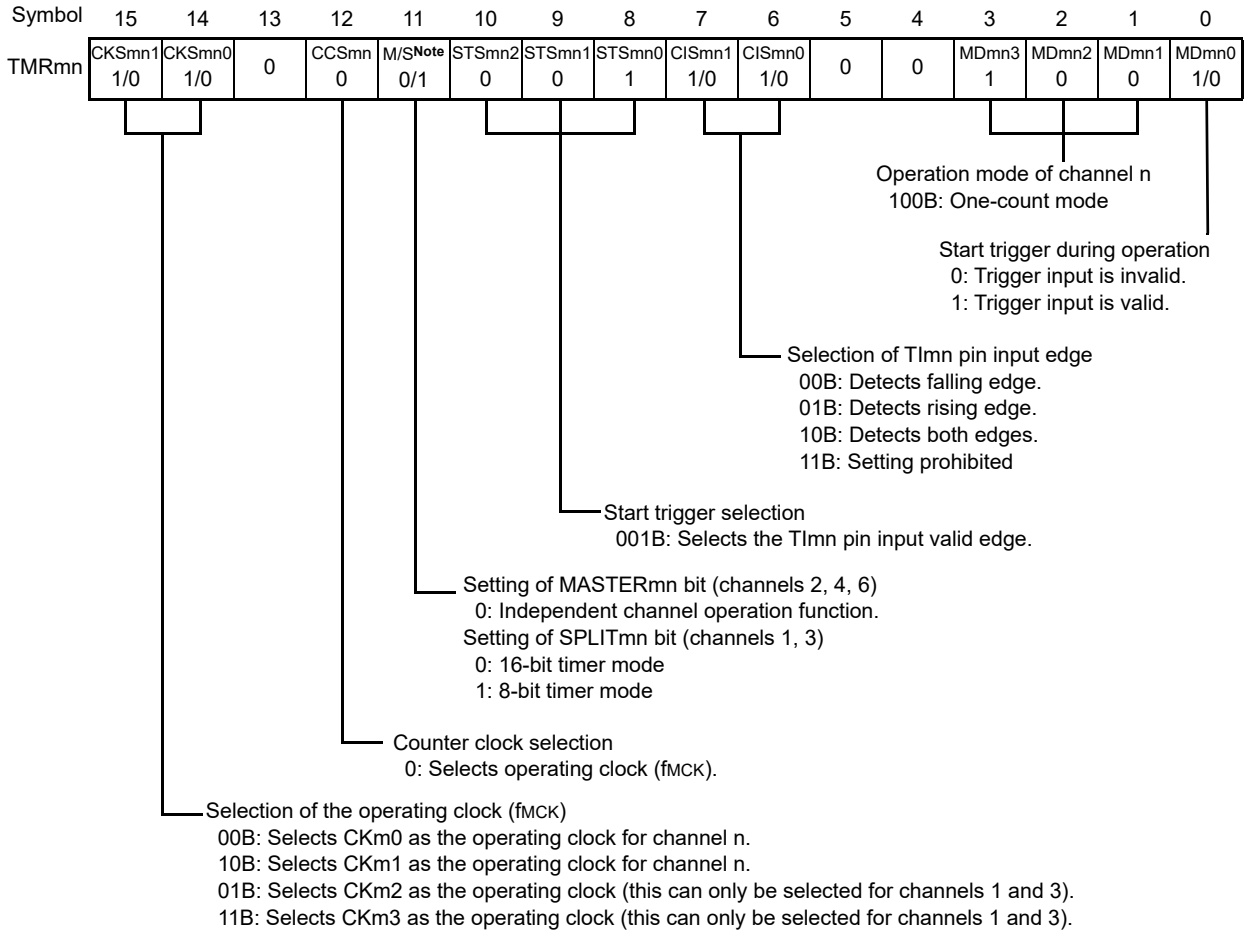


Remark 1. m: Unit number (m = 0), n: Channel number (n = 0 to 7)

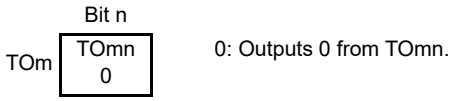
Remark 2. TSmn: Bit n of timer channel start register m (TSm)
TEmn: Bit n of timer channel enable status register m (TEm)
TImn: TImn pin input signal
TCRmn: Timer counter register mn (TCRmn)
TDRmn: Timer data register mn (TDRmn)

Figure 7 - 64 Example of Register Settings for Operation as a Delay Counter

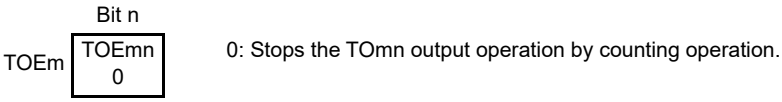
a) Timer mode register mn (TMRmn)



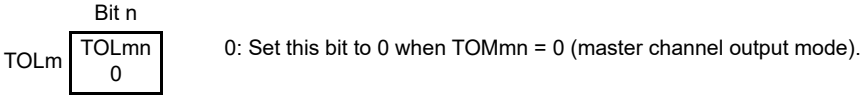
b) Timer output register m (TOm)



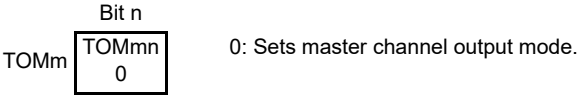
c) Timer output enable register m (TOEm)



d) Timer output level register m (TOLm)



e) Timer output mode register m (TOMm)



Note TMRm2, TMRm4, TMRm6: MASTERmn bit
TMRm1, TMRm3: SPLITmn bit
TMRm0, TMRm5, TMRm7: Fixed to 0

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

Figure 7 - 65 Procedure for Operations When the Delay Counter Function Is to Be Used

	Software Operation	Hardware State
TAU default setting		Power-off state (Clock supply is stopped and writing to each register is disabled.)
	Sets the TAUmEN bit of peripheral enable register 0 (PER0) to 1. →	Power-on state. Each channel stops operating. (Clock supply is started and writing to each register is enabled.)
	Sets timer clock select register m (TPSm). Determines clock frequencies of CKm0 to CKm3.	
Channel default setting	Sets the corresponding bit of the noise filter enable register 1 (NFEN1) to 0 (off) or 1 (on). Sets timer mode register mn (TMRmn) (determines operation mode of channel). INTTMmn output delay is set to timer data register mn (TDRmn). Clears the TOEmn bit to 0 and stops operation of TOmn.	Channel stops operating. (Clock is supplied and some power is consumed.)
Operation start	Sets the TSmn bit to 1. → The TSmn bit automatically returns to 0 because it is a trigger bit.	TEmn = 1, and the start trigger detection (the valid edge of the TImn pin input is detected or the TSmn bit is set to 1) wait state is set.
	The counter starts counting down by the next start trigger detection. • Detects the TImn pin input valid edge. → • Sets the TSmn bit to 1 by the software.	Value of the TDRmn register is loaded to the timer counter register mn (TCRmn).
During operation	Set value of the TDRmn register can be changed. The TCRmn register can always be read. The TSRmn register is not used.	The counter (TCRmn) counts down. When the count value of TCRmn reaches 0000H, the INTTMmn output is generated, and the count operation stops until the next start trigger detection (the valid edge of the TImn pin input is detected or the TSmn bit is set to 1).
Operation stop	The TTmn bit is set to 1. → The TTmn bit automatically returns to 0 because it is a trigger bit.	TEmn = 0, and count operation stops. The TCRmn register holds count value and stops.
TAU stop	The TAUmEN bit of the PER0 register is cleared to 0. → Set the TAUmRES bit of the PRR0 register to 1 to initialize all circuits of the timer array unit. →	This stops supply of the input clock to timer array unit m. All circuits are initialized and SFR of each channel is also initialized.

Operation is resumed.

Remark m: Unit number (m = 0), n: Channel number (n = 0 to 7)

7.9 Simultaneous Channel Operation Function of Timer Array Unit

7.9.1 Operation for the one-shot pulse output function

By using two channels as a set, a one-shot pulse having any delay pulse width can be generated from the signal input to the TImn pin. The delay time and pulse width can be calculated by the following expressions.

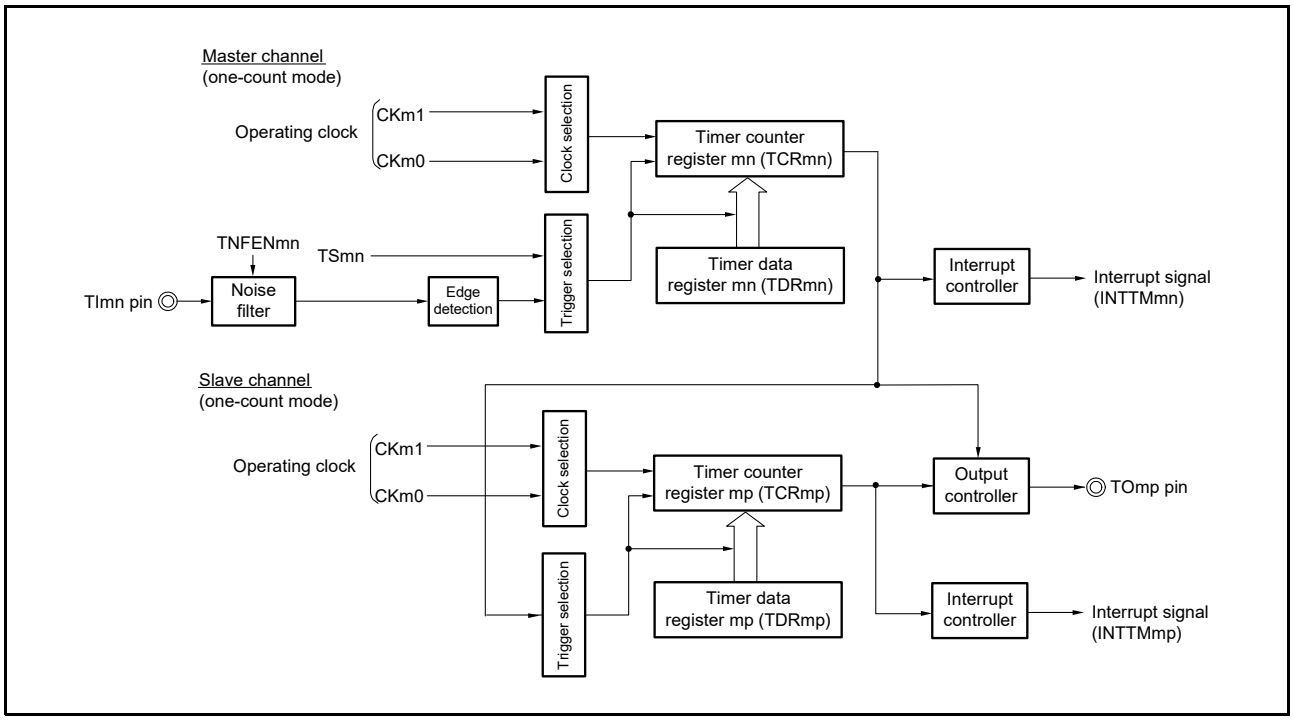
Delay time = {Set value of TDRmn (master) + 2} × Counter clock period
Pulse width = {Set value of TDRmp (slave)} × Counter clock period

The master channel operates in the one-count mode and counts the delays. Timer counter register mn (TCRmn) of the master channel starts operating upon start trigger detection and loads the value of timer data register mn (TDRmn). The TCRmn register counts down from the value of the TDRmn register it has loaded, in synchronization with the counter clock. When TCRmn = 0000H, it outputs INTTMmn and stops counting until the next start trigger is detected. The slave channel operates in the one-count mode and counts the pulse width. The TCRmp register of the slave channel starts operation using INTTMmn of the master channel as a start trigger, and loads the value of the TDRmp register. The TCRmp register counts down from the value of The TDRmp register it has loaded, in synchronization with the count value. When count value = 0000H, it outputs INTTMmp and stops counting until the next start trigger (INTTMmn of the master channel) is detected. The output level of TOmp becomes active one counter clock after generation of INTTMmn from the master channel, and inactive when TCRmp = 0000H. Instead of using the TImn pin input, a one-shot pulse can also be output using the software operation (TSmn = 1) as a start trigger.

Caution **The timing of loading of timer data register mn (TDRmn) of the master channel is different from that of the TDRmp register of the slave channel. If the TDRmn and TDRmp registers are rewritten during operation, therefore, an illegal waveform is output. Rewrite the TDRmn register after INTTMmn is generated and the TDRmp register after INTTMmp is generated.**

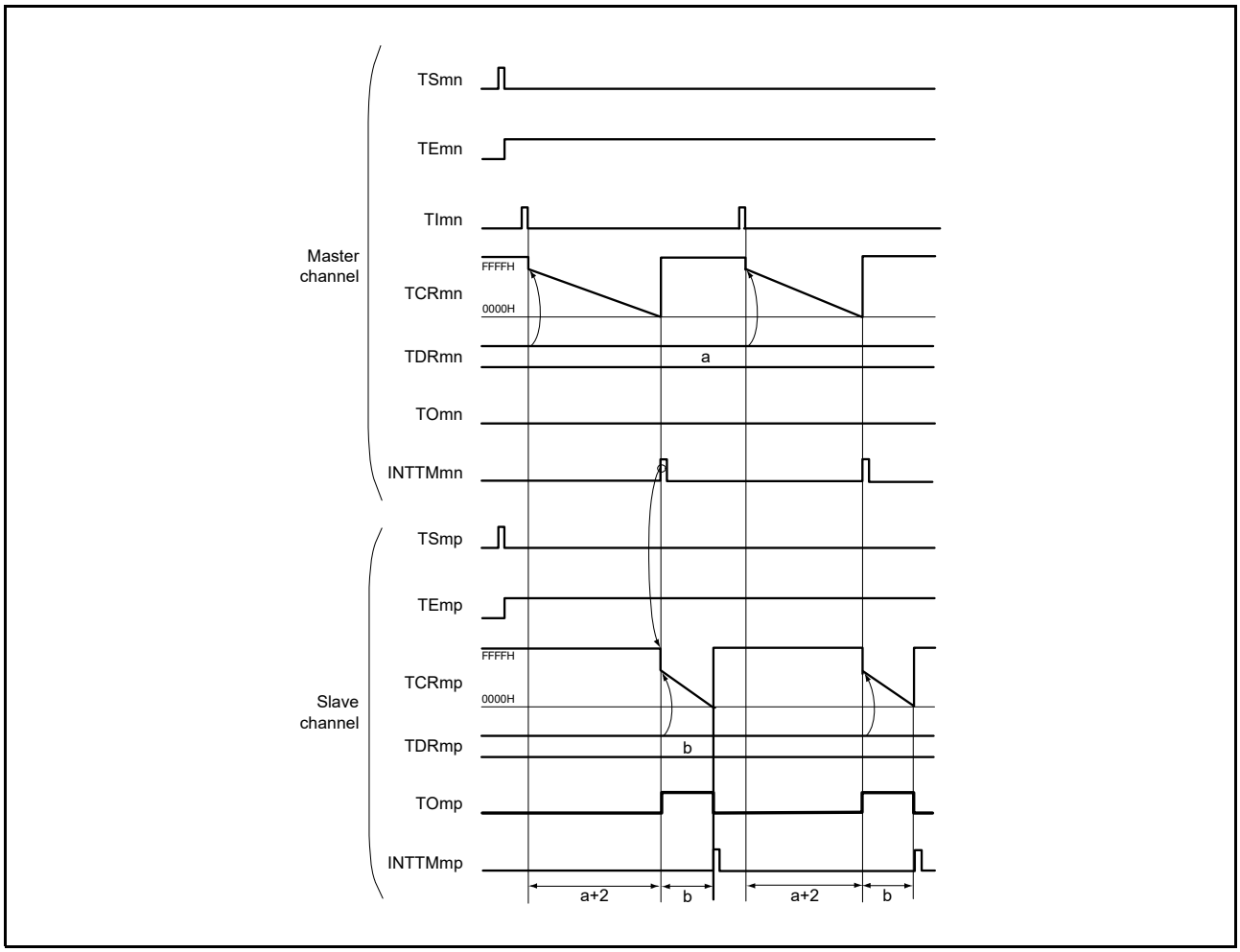
Remark m: Unit number (m = 0), n: Master channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)

Figure 7 - 66 Block Diagram for Operation for the One-shot Pulse Output Function



Remark m: Unit number (m = 0), n: Master channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)

Figure 7 - 67 Example of Basic Timing during Operation for the One-shot Pulse Output Function

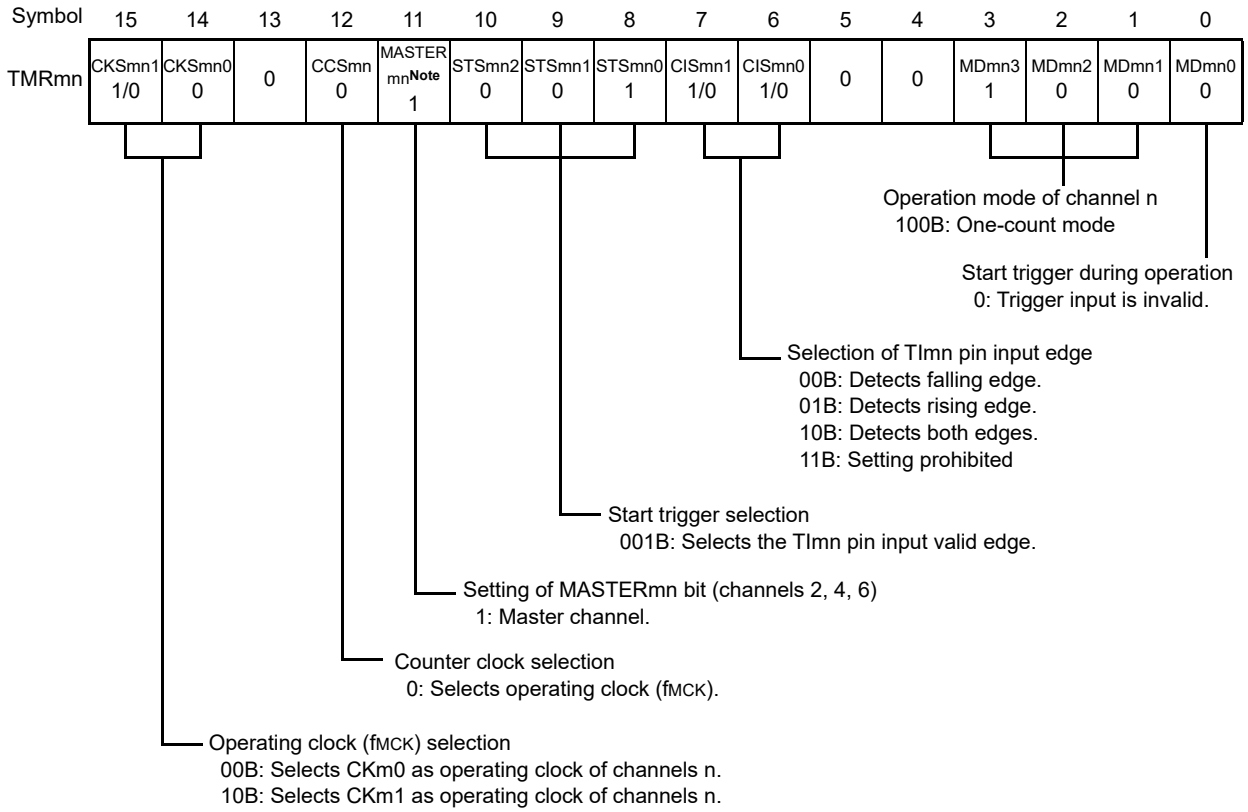


Remark 1. m: Unit number (m = 0), n: Master channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)

Remark 2. TSmn, TSmp: Bits n and p of timer channel start register m (TSm)
TEmn, TEmp: Bits n and p of timer channel enable status register m (TEm)
TImn, TImp: Signals on the TImn and TImp input pins
TCRmn, TCRmp: Timer counter registers mn, mp (TCRmn, TCRmp)
TDRmn, TDRmp: Timer data registers mn, mp (TDRmn, TDRmp)
TOmn, TOmp: Signals on the TOmn and TOmp output pins

Figure 7 - 68 Example of Register Settings for the Master Channel When the One-shot Pulse Output Function Is to Be Used

a) Timer mode register mn (TMRmn)



b) Timer output register m (TOm)

	Bit n
TOm	TOmn 0

0: Outputs 0 from TOmn.

c) Timer output enable register m (TOEm)

	Bit n
TOEm	TOEmn 0

0: Stops the TOmn output operation by counting operation.

d) Timer output level register m (TOLm)

	Bit n
TOLm	TOLmn 0

0: Set this bit to 0 when TOMmn = 0 (master channel output mode).

e) Timer output mode register m (TOMm)

	Bit n
TOMm	TOMmn 0

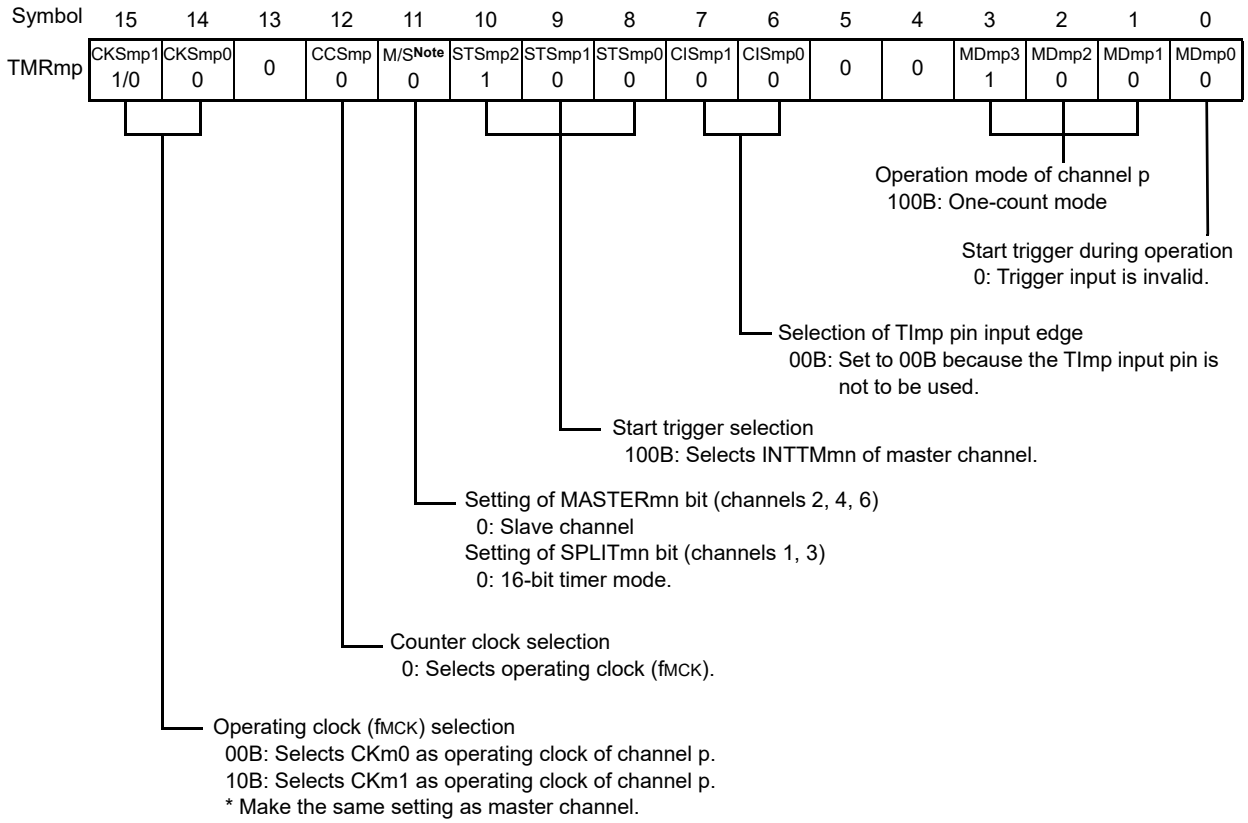
0: Sets master channel output mode.

Note TMRm2, TMRm4, TMRm6: MASTERmn = 1
TMRm0: Fixed to 0

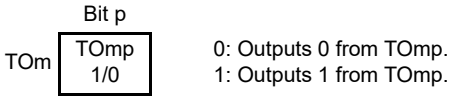
Remark m: Unit number (m = 0), n: Master channel number (n = 0, 2, 4, 6)

Figure 7 - 69 Example of Register Settings for the Slave Channel When the One-shot Pulse Output Function Is to Be Used

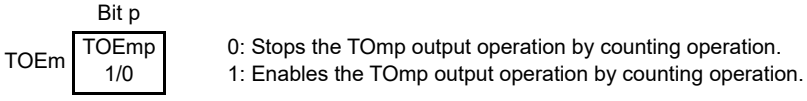
a) Timer mode register mp (TMRmp)



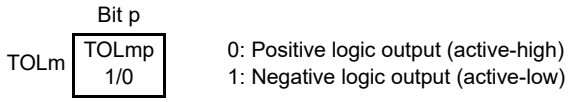
b) Timer output register m (TOm)



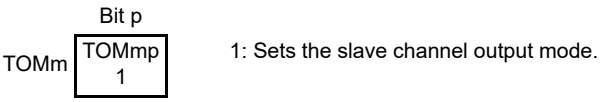
c) Timer output enable register m (TOEm)



d) Timer output level register m (TOLm)



e) Timer output mode register m (TOMm)



Note TMRm2, TMRm4, TMRm6: MASTERmn bit
TMRm1, TMRm3: SPLITmp bit
TMRm5, TMRm7: Fixed to 0

Remark m: Unit number (m = 0), n: Master channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)

Figure 7 - 70 Procedure for Operations When the One-shot Pulse Output Function Is to Be Used (1/2)

	Software Operation	Hardware State
TAU default setting		Power-off state (Clock supply is stopped and writing to each register is disabled.)
	Sets the TAUmEN bit of peripheral enable registers 0 (PER0) to 1. →	Power-on state. Each channel stops operating. (Clock supply is started and writing to each register is enabled.)
	Sets timer clock select register m (TPSm). Determines clock frequencies of CKm0 and CKm1.	
Channel default setting	Sets the corresponding bit of the noise filter enable register 1 (NFEN1) to 1. Sets timer mode register mn, mp (TMRmn, TMRmp) of two channels to be used (determines operation mode of channels). An output delay is set to timer data register mn (TDRmn) of the master channel, and a pulse width is set to the TDRmp register of the slave channel.	Channel stops operating. (Clock is supplied and some power is consumed.)
	Sets slave channel. The TOMmp bit of timer output mode register m (TOMm) is set to 1 (slave channel output mode). Sets the TOLmp bit. Sets the TOmp bit and determines default level of the TOmp output. → Sets the TOEmp bit to 1 and enables operation of TOmp. → Clears the port register and port mode register to 0. →	The TOmp pin goes into Hi-Z output state. The TOmp default setting level is output when the port mode register is in output mode and the port register is 0. TOmp does not change because channel stops operating. The TOmp pin outputs the TOmp set level.

Figure 7 - 70 Procedure for Operations When the One-shot Pulse Output Function Is to Be Used (2/2)

<R>

	Software Operation	Hardware State
Operation start	Sets the TOEmp bit (slave) to 1 (only when operation is resumed). The TSmn (master) and TSmp (slave) bits of timer channel start register m (TSm) are set to 1 at the same time. → The TSmn and TSmp bits automatically return to 0 because they are trigger bits.	The TEmn and TEmp bits are set to 1 and the master channel enters the start trigger detection (the valid edge of the TImn pin input is detected or the TSmn bit of the master channel is set to 1) wait state. Counter stops operating.
	Count operation of the master channel is started by start trigger detection of the master channel. • Detects the TImn pin input valid edge. • Sets the TSmn bit of the master channel to 1 by software[Note].	Master channel starts counting.
During operation	Set values of only the CISmn1 and CISmn0 bits of the TMRmn register can be changed. Set values of the TMRmp, TDRmn, TDRmp registers, TOMmn, TOMmp, TOLmn, and TOLmp bits cannot be changed. The TCRmn and TCRmp registers can always be read. The TSRmn and TSRmp registers are not used. Set values of the TOm and TOEm registers by slave channel can be changed.	Master channel loads the value of the TDRmn register to timer counter register mn (TCRmn) by the start trigger detection (the valid edge of the TImn pin input is detected or the TSmn bit of the master channel is set to 1), and the counter starts counting down. When the count value reaches TCRmn = 0000H, the INTTMmn output is generated, and the counter stops until the next start trigger detection. The slave channel, triggered by INTTMmn of the master channel, loads the value of the TDRmp register to the TCRmp register, and the counter starts counting down. The output level of TOmp becomes active one counter clock after generation of INTTMmn from the master channel. It becomes inactive when TCRmp = 0000H, and the counting operation is stopped. After that, the above operation is repeated.
Operation stop	The TTmn (master) and TTmp (slave) bits are set to 1 at the same time. → The TTmn and TTmp bits automatically return to 0 because they are trigger bits.	TEmn, TEmp = 0, and count operation stops. The TCRmn and TCRmp registers hold count value and stop. The TOmp output is not initialized and retains its current state.
	The TOEmp bit of slave channel is cleared to 0 and value is set to the TOmp bit. →	The TOmp pin outputs the TOmp set level.
TAU stop	To hold the TOmp pin output level Clears the TOmp bit to 0 after the value to be held is set to the port register. → When holding the TOmp pin output level is not necessary Setting not required.	The TOmp pin output level is held by port function.
	The TAUmEN bit of the PER0 register is cleared to 0. → Set the TAUmRES bit of the PRR0 register to 1 to initialize all circuits of the timer array unit. →	This stops supply of the input clock to timer array unit m. All circuits are initialized and SFR of each channel is also initialized. (The TOmp bit is cleared to 0 and the TOmp pin is set to port mode.)

Operation is resumed.

Note Do not set the TSmn bit of the slave channel to 1.

Remark m: Unit number (m = 0), n: Master channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)

7.9.2 Operation for the PWM function

Two channels can be used as a set to generate a pulse of any period and duty factor. The period and duty factor of the output pulse can be calculated by the following expressions.

Pulse period = {Set value of TDRmn (master) + 1} × Counter clock period
Duty factor [%] = {Set value of TDRmp (slave)}/{Set value of TDRmn (master) + 1} × 100
0% output: Set value of TDRmp (slave) = 0000H
100% output: Set value of TDRmp (slave) ≥ {Set value of TDRmn (master) + 1}

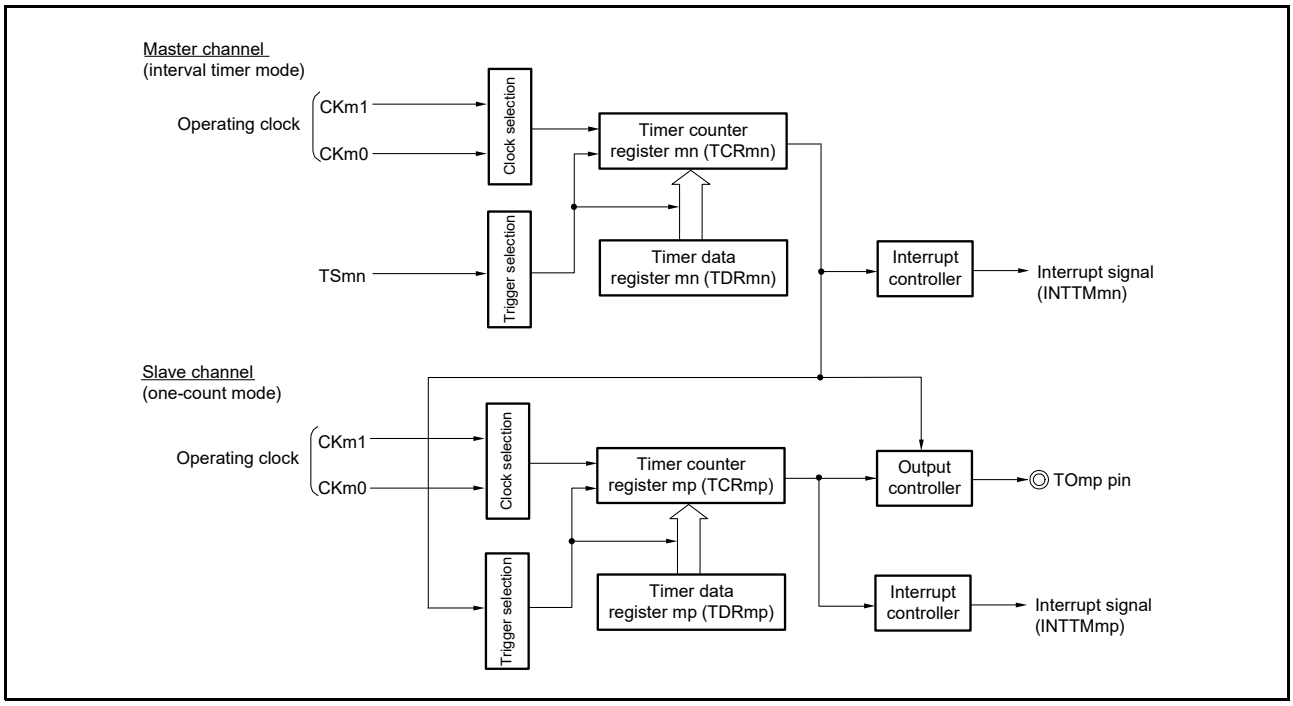
Remark The duty factor exceeds 100% if the set value of TDRmp (slave) > (set value of TDRmn (master) + 1), it summarizes to 100% output.

The master channel operates in the interval timer mode. If the operation enable (start) trigger bit (TSmn) of timer channel start register m (TSm) is set to 1, an interrupt (INTTMmn) is output, the value set to timer data register mn (TDRmn) is loaded to timer counter register mn (TCRmn), and the counter counts down in synchronization with the counter clock. When the counter reaches 0000H, INTTMmn is output, the value of the TDRmn register is loaded again to the TCRmn register, and the counter counts down. This operation is repeated until the operation stop trigger bit (TTmn) of timer channel stop register m (TTm) is set to 1. If two channels are used to output a PWM waveform, the period until the master channel counts down to 0000H is the PWM output (TOmp) cycle. The slave channel operates in one-count mode. By using INTTMmn from the master channel as a start trigger, the TCRmp register loads the value of the TDRmp register and the counter counts down to 0000H. When the counter reaches 0000H, it outputs INTTMmp and waits until the next start trigger (INTTMmn from the master channel) is generated. If two channels are used to output a PWM waveform, the period until the slave channel counts down to 0000H is the PWM output (TOmp) duty. PWM output (TOmp) goes to the active level one clock after the master channel generates INTTMmn and goes to the inactive level when the TCRmp register of the slave channel becomes 0000H.

Caution **To rewrite both timer data register mn (TDRmn) of the master channel and the TDRmp register of the slave channel, a write access is necessary two times. The timing at which the values of the TDRmn and TDRmp registers are loaded to the TCRmn and TCRmp registers is upon occurrence of INTTMmn of the master channel. Thus, when rewriting is performed split before and after occurrence of INTTMmn of the master channel, the TOmp pin cannot output the expected waveform. To rewrite both the TDRmn register of the master and the TDRmp register of the slave, therefore, be sure to rewrite both the registers immediately after INTTMmn is generated from the master channel.**

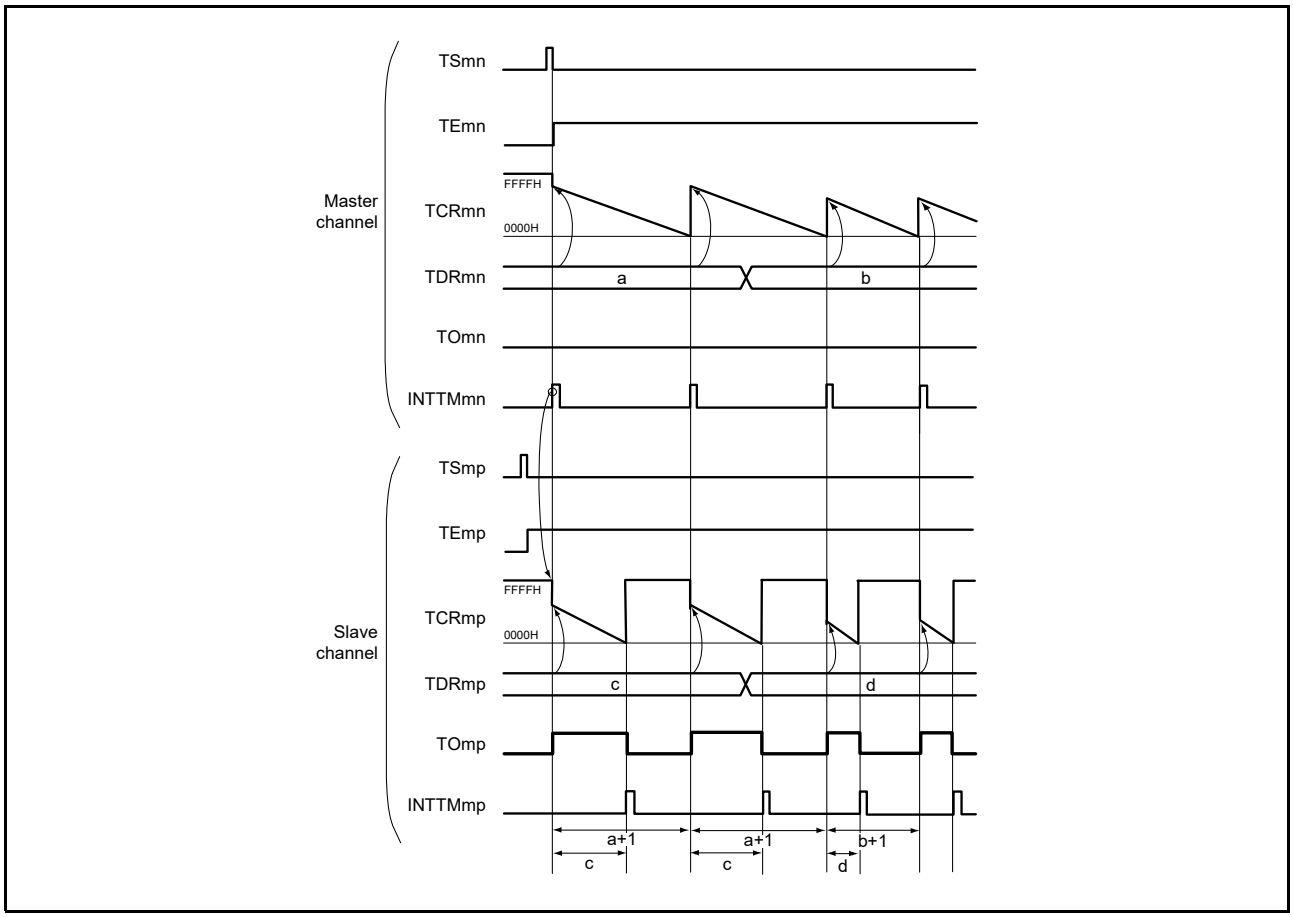
Remark m: Unit number (m = 0), n: Master channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)

Figure 7 - 71 Block Diagram for Operation for the PWM Function



Remark m: Unit number (m = 0), n: Master channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)

Figure 7 - 72 Example of Basic Timing during Operation for the PWM Function

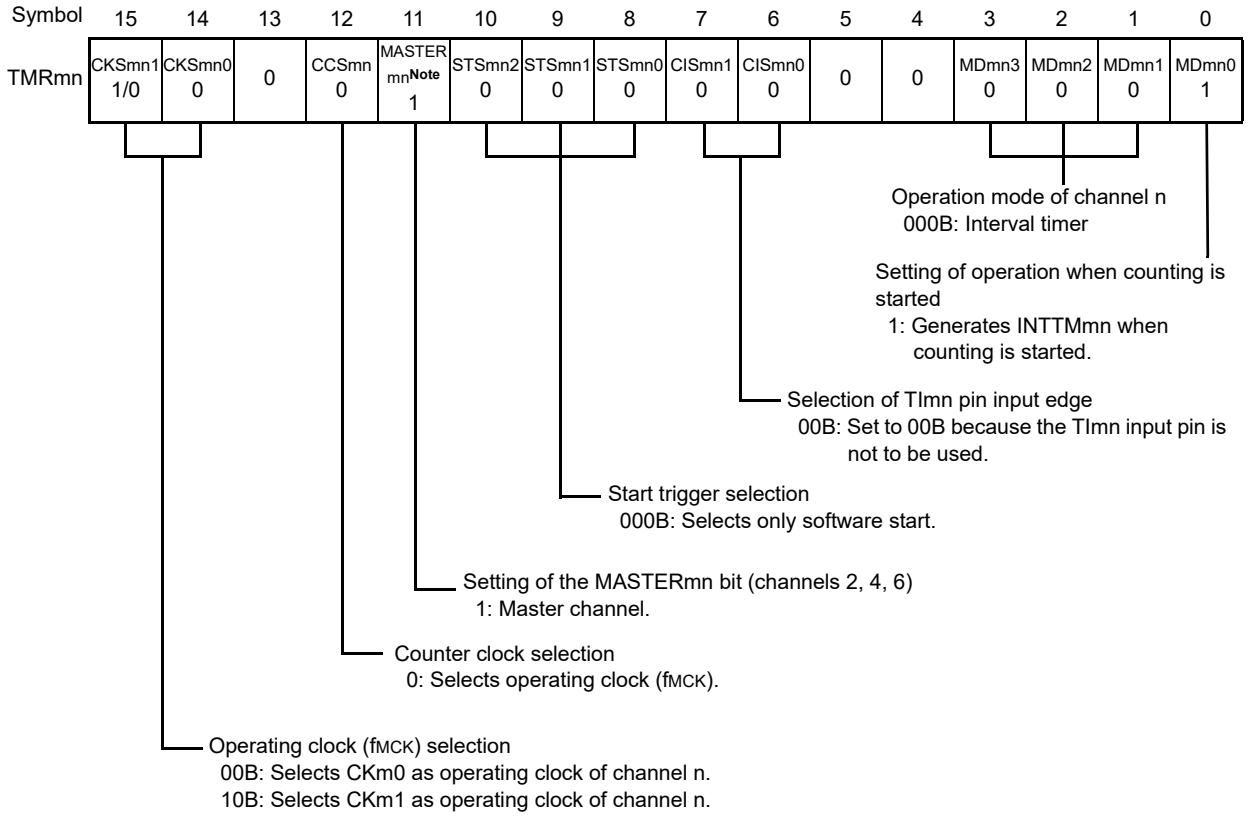


Remark 1. m: Unit number (m = 0), n: Master channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)

Remark 2. TSmn, TSmp: Bits n and p of timer channel start register m (TSm)
TEmn, TEmp: Bits n and p of timer channel enable status register m (TEm)
TCRmn, TCRmp: Timer counter registers mn, mp (TCRmn, TCRmp)
TDRmn, TDRmp: Timer data registers mn, mp (TDRmn, TDRmp)
TOmn, TOmp: Signals on the TOmn and TOmp output pins

Figure 7 - 73 Example of Register Settings for the Master Channel When the PWM Function Is to Be Used

a) Timer mode register mn (TMRmn)



b) Timer output register m (TOm)

	Bit n
TOm	TOmn 0

0: Outputs 0 from TOmn.

c) Timer output enable register m (TOEm)

	Bit n
TOEm	TOEmn 0

0: Stops the TOmn output operation by counting operation.

d) Timer output level register m (TOLm)

	Bit n
TOLm	TOLmn 0

0: Set this bit to 0 when TOMmn = 0 (master channel output mode).

e) Timer output mode register m (TOMm)

	Bit n
TOMm	TOMmn 0

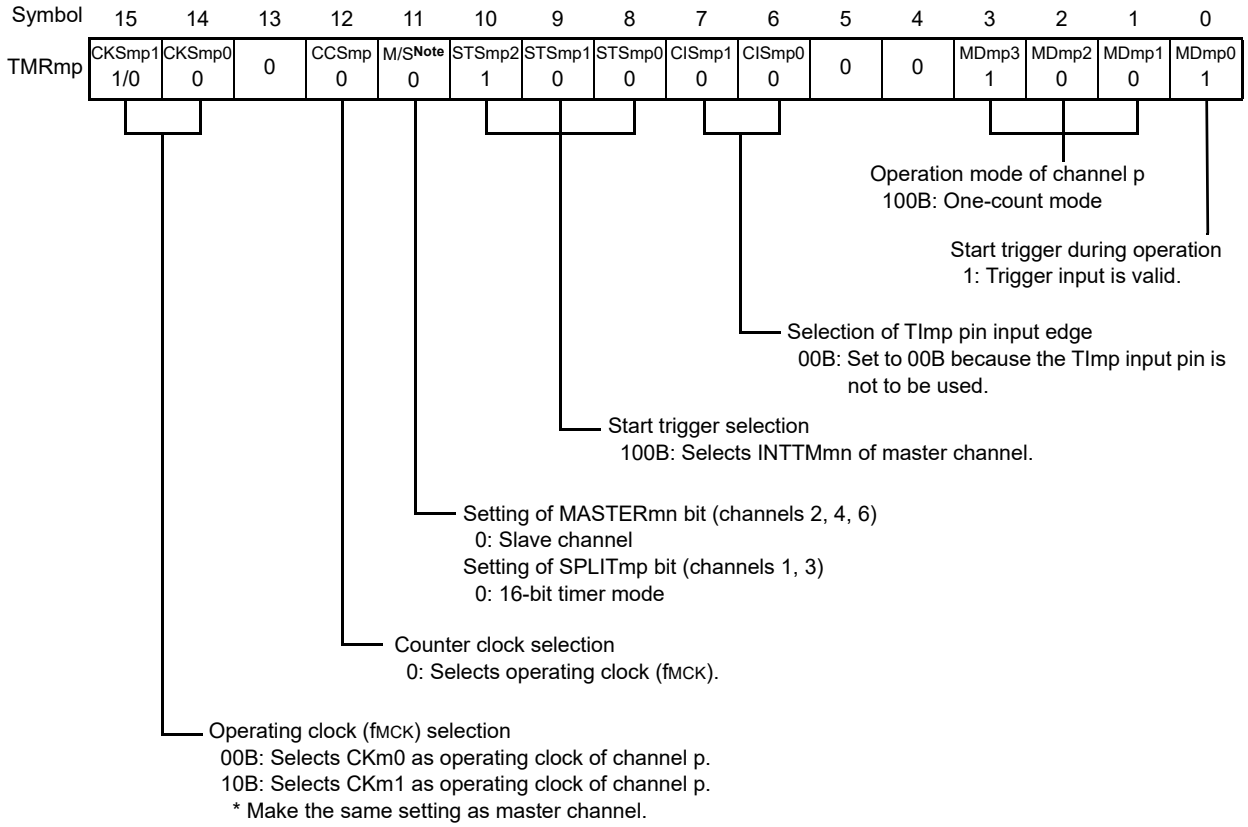
0: Sets master channel output mode.

Note TMRm2, TMRm4, TMRm6: MASTERmn = 1
TMRm0: Fixed to 0

Remark m: Unit number (m = 0), n: Master channel number (n = 0, 2, 4, 6)

Figure 7 - 74 Example of Register Settings for the Slave Channel When the PWM Function Is to Be Used

a) Timer mode register mp (TMRmp)



b) Timer output register m (TOm)

	Bit p	
TOm	TOmp 1/0	0: Outputs 0 from TOmp. 1: Outputs 1 from TOmp.

c) Timer output enable register m (TOEm)

	Bit p	
TOEm	TOEmp 1/0	0: Stops the TOmp output operation by counting operation. 1: Enables the TOmp output operation by counting operation.

d) Timer output level register m (TOLm)

	Bit p	
TOLm	TOLmp 1/0	0: Positive logic output (active-high) 1: Negative logic output (active-low)

e) Timer output mode register m (TOMm)

	Bit p	
TOMm	TOMmp 1	1: Sets the slave channel output mode.

Note TMRm2, TMRm4, TMRm6: MASTERmn bit
TMRm1, TMRm3: SPLITmp bit
TMRm5, TMRm7: Fixed to 0

Remark m: Unit number (m = 0), n: Master channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)

Figure 7 - 75 Procedure for Operations When the PWM Function Is to Be Used (1/2)

	Software Operation	Hardware State
TAU default setting		Power-off state (Clock supply is stopped and writing to each register is disabled.)
	Sets the TAUmEN bit of peripheral enable register 0 (PER0) to 1. →	Power-on state. Each channel stops operating. (Clock supply is started and writing to each register is enabled.)
	Sets timer clock select register m (TPSm). Determines clock frequencies of CKm0 and CKm1.	
Channel default setting	Sets timer mode registers mn, mp (TMRmn, TMRmp) of two channels to be used (determines operation mode of channels). An interval (period) value is set to timer data register mn (TDRmn) of the master channel, and a duty factor is set to the TDRmp register of the slave channel.	Channel stops operating. (Clock is supplied and some power is consumed.)
	Sets slave channel. The TOMmp bit of timer output mode register m (TOMm) is set to 1 (slave channel output mode). Sets the TOLmp bit. Sets the TOmp bit and determines default level of the TOmp output. →	The TOmp pin goes into Hi-Z output state. The TOmp default setting level is output when the port mode register is in output mode and the port register is 0.
	Sets the TOEmp bit to 1 and enables operation of TOmp. →	TOmp does not change because channel stops operating.
	Clears the port register and port mode register to 0. →	The TOmp pin outputs the TOmp set level.

Figure 7 - 75 Procedure for Operations When the PWM Function Is to Be Used (2/2)

	Software Operation	Hardware State
Operation start	Sets the TOEmp bit (slave) to 1 (only when operation is resumed). The TSmn (master) and TSmp (slave) bits of timer channel start register m (TSm) are set to 1 at the same time. → The TSmn and TSmp bits automatically return to 0 because they are trigger bits.	TEmn = 1, TEmp = 1 When the master channel starts counting, INTTMmn is generated. Triggered by this interrupt, the slave channel also starts counting.
During operation	Set values of the TMRmn and TMRmp registers, TOMmn, TOMmp, TOLmn, and TOLmp bits cannot be changed. Set values of the TDRmn and TDRmp registers can be changed after INTTMmn of the master channel is generated. The TCRmn and TCRmp registers can always be read. The TSRmn and TSRmp registers are not used.	The counter of the master channel loads the TDRmn register value to timer counter register mn (TCRmn), and counts down. When the count value reaches TCRmn = 0000H, INTTMmn output is generated. At the same time, the value of the TDRmn register is loaded to the TCRmn register, and the counter starts counting down again. At the slave channel, the value of the TDRmp register is loaded to the TCRmp register, triggered by INTTMmn of the master channel, and the counter starts counting down. The output level of TOmp becomes active one counter clock after generation of the INTTMmn output from the master channel. It becomes inactive when TCRmp = 0000H, and the counting operation is stopped. After that, the above operation is repeated.
Operation stop	The TTmn (master) and TTmp (slave) bits are set to 1 at the same time. → The TTmn and TTmp bits automatically return to 0 because they are trigger bits.	TEmn, TEmp = 0, and count operation stops. The TCRmn and TCRmp registers hold count value and stop. The TOmp output is not initialized and retains its current state.
	The TOEmp bit of slave channel is cleared to 0 and value is set to the TOmp bit. →	The TOmp pin outputs the TOmp set level.
TAU stop	To hold the TOmp pin output level Clears the TOmp bit to 0 after the value to be held is set to the port register. → When holding the TOmp pin output level is not necessary Setting not required.	The TOmp pin output level is held by port function.
	The TAUmEN bit of the PER0 register is cleared to 0. → Set the TAUmRES bit of the PRR0 register to 1 to initialize all circuits of the timer array unit. →	This stops supply of the input clock to timer array unit m. All circuits are initialized and SFR of each channel is also initialized. (The TOmp bit is cleared to 0 and the TOmp pin is set to port mode.)

Operation is resumed.

Remark m: Unit number (m = 0), n: Master channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)

7.9.3 Operation for the multiple PWM output function

By extending the PWM function and using multiple slave channels, many PWM waveforms with different duty values can be output. For example, when using two slave channels, the period and duty factor of an output pulse can be calculated by the following expressions.

Pulse period = {Set value of TDRmn (master) + 1} × Counter clock period
Duty factor 1 [%] = {Set value of TDRmp (slave 1)}/{Set value of TDRmn (master) + 1} × 100
Duty factor 2 [%] = {Set value of TDRmq (slave 2)}/{Set value of TDRmn (master) + 1} × 100

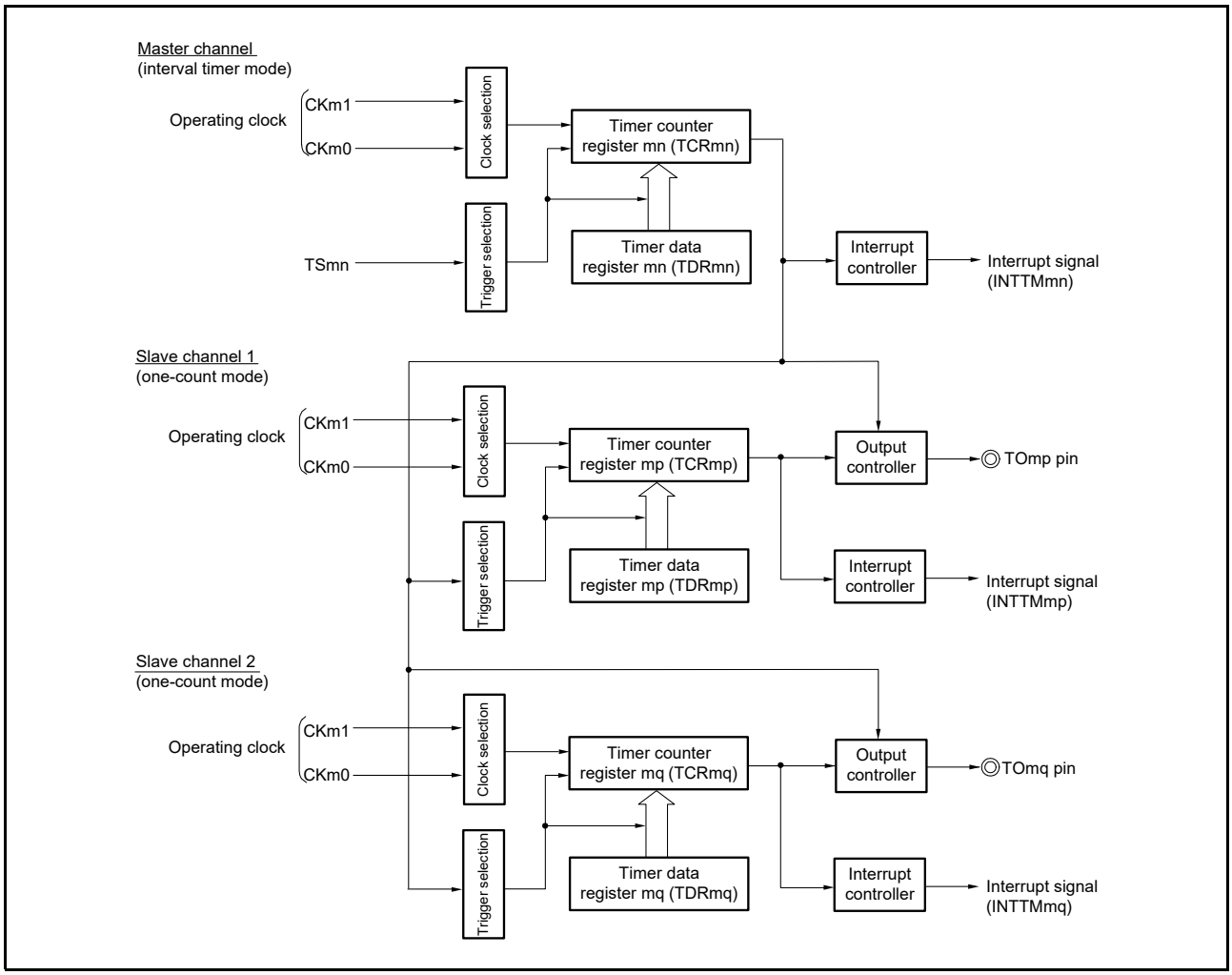
Remark Although the duty factor exceeds 100% if the set value of TDRmp (slave 1) > {set value of TDRmn (master) + 1} or if the {set value of TDRmq (slave 2)} > {set value of TDRmn (master) + 1}, it is summarized into 100% output.

Timer counter register mn (TCRmn) of the master channel operates in the interval timer mode and counts the periods. The TCRmp register of the slave channel 1 operates in one-count mode, counts the duty factor, and outputs a PWM waveform from the TOmp pin. The TCRmp register loads the value of timer data register mp (TDRmp), using INTTMmn of the master channel as a start trigger, and starts counting down. When TCRmp = 0000H, TCRmp outputs INTTMmp and stops counting until the next start trigger (INTTMmn of the master channel) has been input. The output level of TOmp becomes active one counter clock after generation of INTTMmn from the master channel, and inactive when TCRmp = 0000H. In the same way as the TCRmp register of the slave channel 1, the TCRmq register of the slave channel 2 operates in one-count mode, counts the duty factor, and outputs a PWM waveform from the TOmq pin. The TCRmq register loads the value of the TDRmq register, using INTTMmn of the master channel as a start trigger, and starts counting down. When TCRmq = 0000H, the TCRmq register outputs INTTMmq and stops counting until the next start trigger (INTTMmn of the master channel) has been input. The output level of TOmq becomes active one counter clock after generation of INTTMmn from the master channel, and inactive when TCRmq = 0000H. When channel 0 is used as the master channel as above, up to seven types of PWM signals can be output at the same time.

Caution **To rewrite both timer data register mn (TDRmn) of the master channel and the TDRmp register of the slave channel 1, write access is necessary at least twice. Since the values of the TDRmn and TDRmp registers are loaded to the TCRmn and TCRmp registers after INTTMmn is generated from the master channel, if rewriting is performed separately before and after generation of INTTMmn from the master channel, the TOmp pin cannot output the expected waveform. To rewrite both the TDRmn register of the master and the TDRmp register of the slave, be sure to rewrite both the registers immediately after INTTMmn is generated from the master channel (This applies also to the TDRmq register of the slave channel 2).**

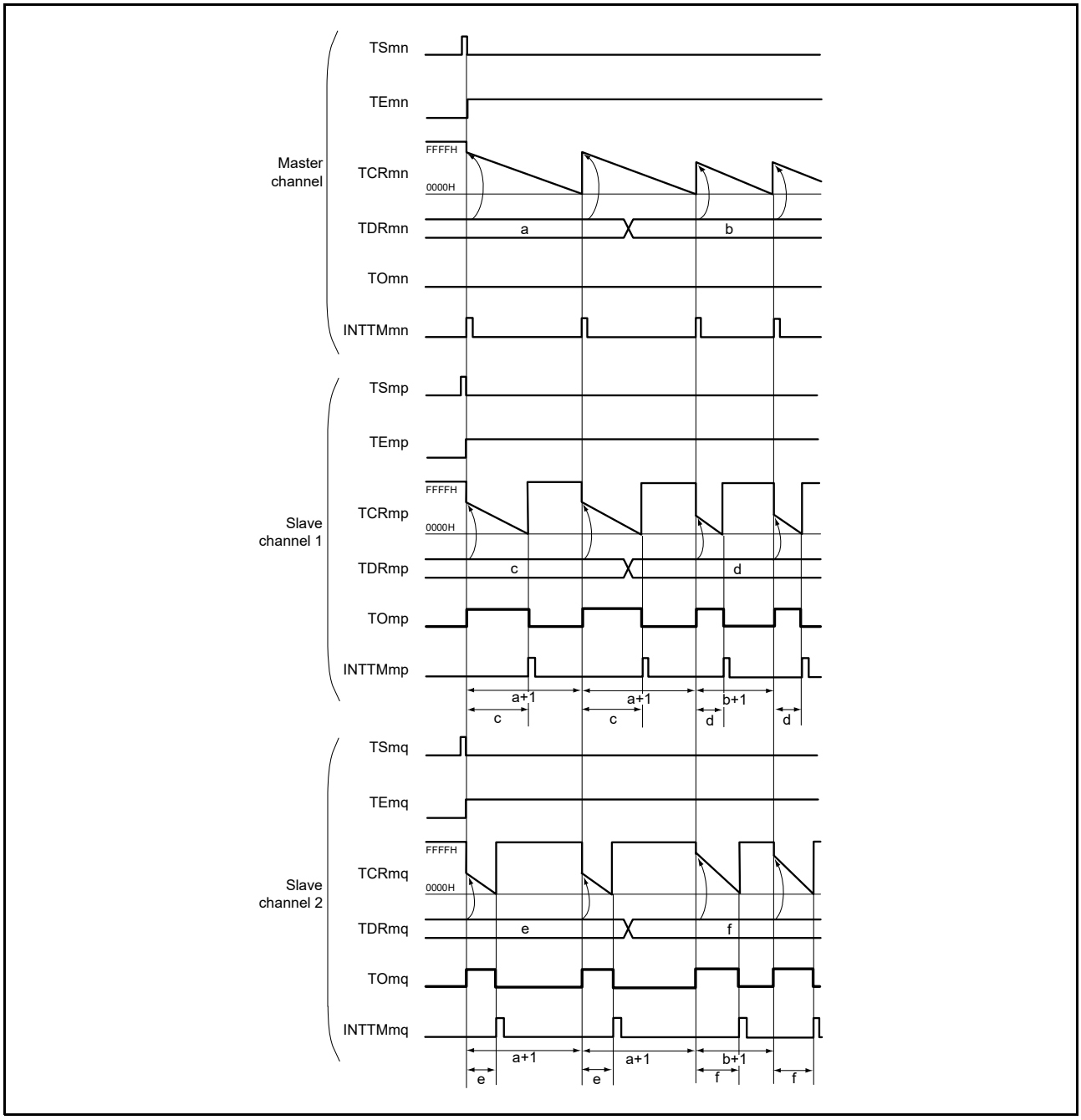
Remark m: Unit number (m = 0), n: Master channel number (n = 0, 2, 4)
p: Slave channel number, q: Slave channel number
n < p < q ≤ 7 (Where p and q are integers)

Figure 7 - 76 Block Diagram for Operation for the Multiple PWM Output Function (for Two Types of PWM Output)



Remark m: Unit number (m = 0), n: Master channel number (n = 0, 2, 4)
p: Slave channel number, q: Slave channel number
n < p < q ≤ 7 (Where p and q are integers)

Figure 7 - 77 Example of Basic Timing during Operation for the Multiple PWM Output Function (for Two Types of PWM Output)

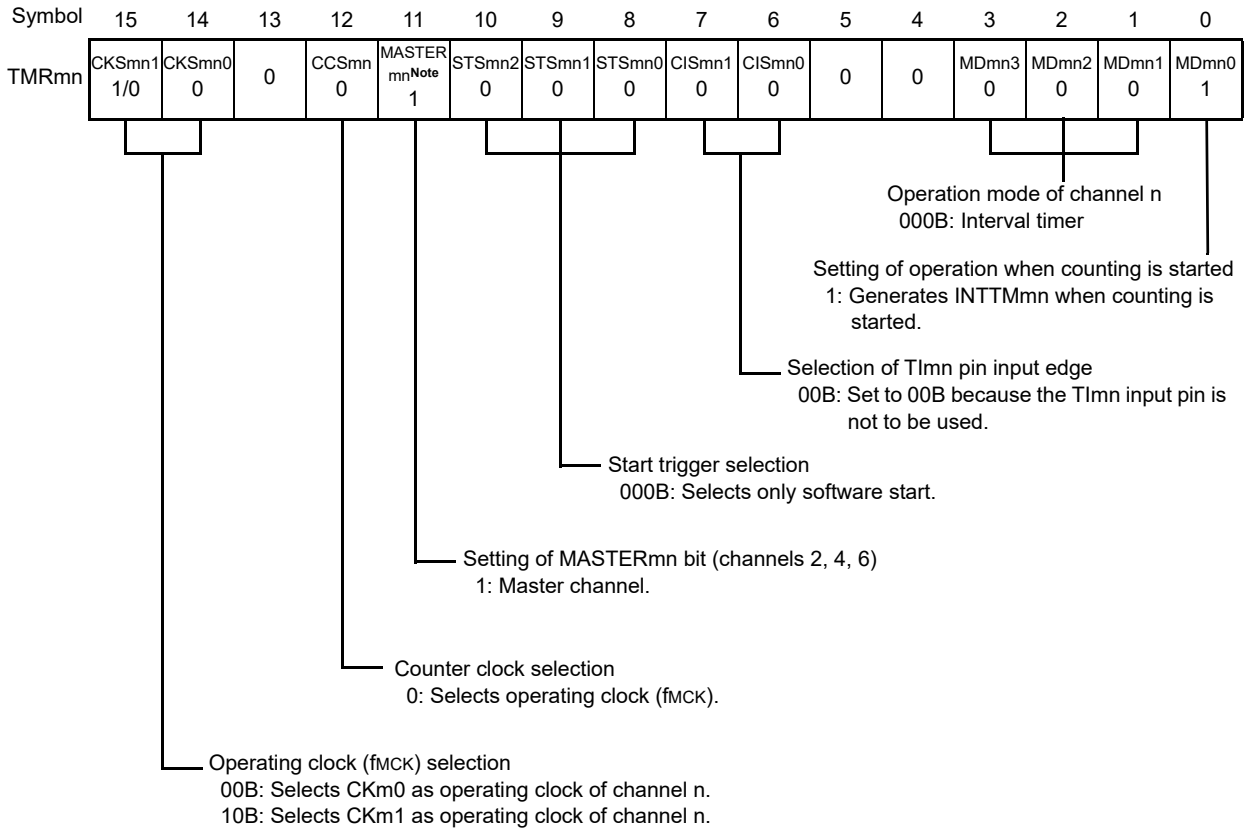


Remark 1. m: Unit number (m = 0), n: Master channel number (n = 0, 2, 4)
p: Slave channel number, q: Slave channel number
n < p < q ≤ 7 (Where p and q are integers)

Remark 2. TSmn, TSmp, TSmq: Bits n, p, and q of timer channel start register m (TSm)
TEmn, TEmp, TEmq: Bits n, p, and q of timer channel enable status register m (TEm)
TCRmn, TCRmp, TCRmq: Timer counter registers mn, mp, mq (TCRmn, TCRmp, TCRmq)
TDRmn, TDRmp, TDRmq: Timer data registers mn, mp, mq (TDRmn, TDRmp, TDRmq)
TOmn, TOmp, TOmq: Signals on the TOmn, TOmp, and TOmq output pins

Figure 7 - 78 Example of Register Settings for the Master Channel When the Multiple PWM Output Function Is to Be Used

a) Timer mode register mn (TMRmn)



b) Timer output register m (TOm)

	Bit n	
TOm	TOmn 0	0: Outputs 0 from TOmn.

c) Timer output enable register m (TOEm)

	Bit n	
TOEm	TOEmn 0	0: Stops the TOmn output operation by counting operation.

d) Timer output level register m (TOLm)

	Bit n	
TOLm	TOLmn 0	0: Set this bit to 0 when TOMmn = 0 (master channel output mode).

e) Timer output mode register m (TOMm)

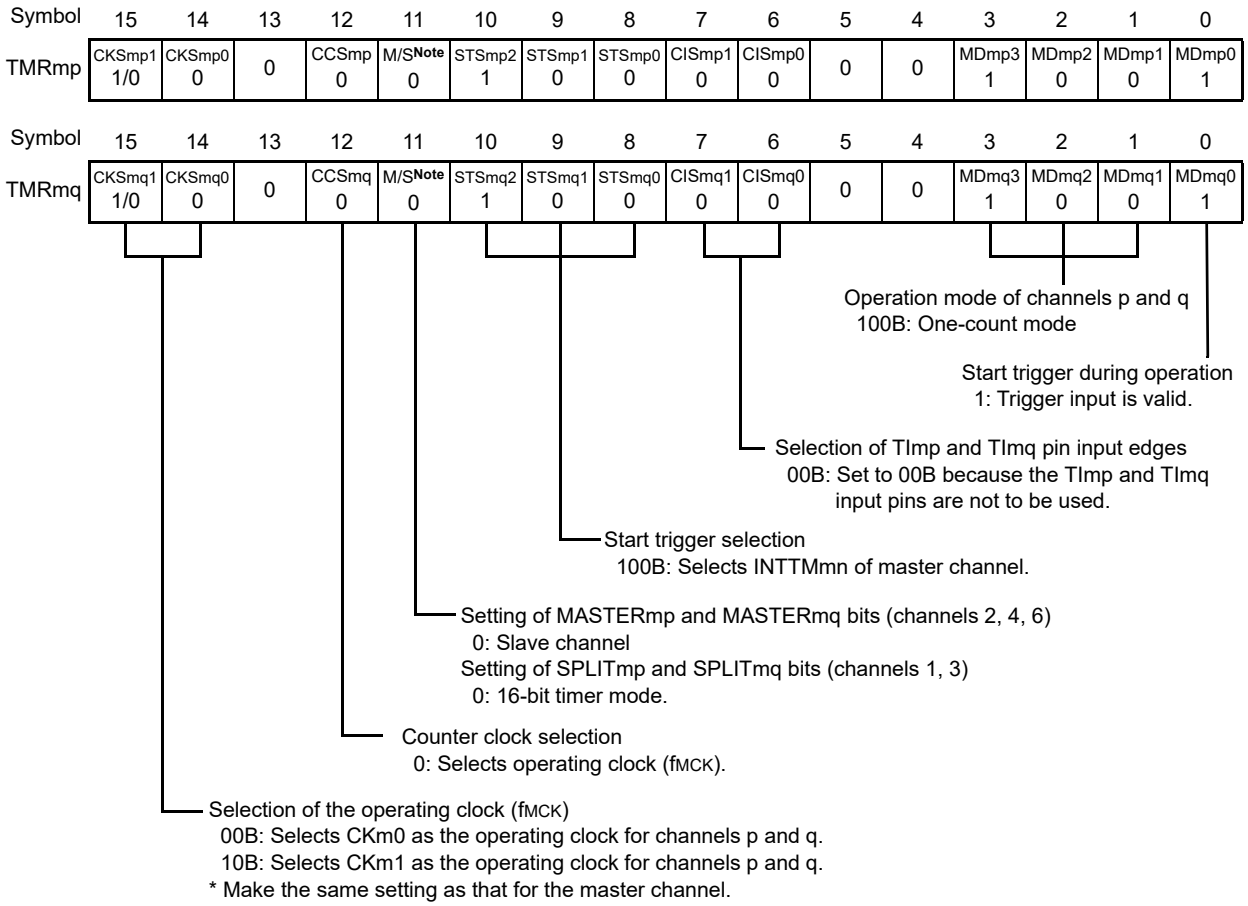
	Bit n	
TOMm	TOMmn 0	0: Sets master channel output mode.

Note TMRm2, TMRm4, TMRm6: MASTERmn = 1
TMRm0: Fixed to 0

Remark m: Unit number (m = 0), n: Master channel number (n = 0, 2, 4)

Figure 7 - 79 Example of Register Settings for the Slave Channel When the Multiple PWM Output Function Is to Be Used (for Two Types of PWM Output)

a) Timer mode registers mp and mq (TMRmp and TMRmq)



b) Timer output register m (TOm)

	Bit q	Bit p	
TOm	TOmq 1/0	TOmp 1/0	0: Outputs 0 from TOmp or TOmq. 1: Outputs 1 from TOmp or TOmq.

c) Timer output enable register m (TOEm)

	Bit q	Bit p	
TOEm	TOEmq 1/0	TOEmp 1/0	0: Stops the TOmp or TOmq output operation by counting operation. 1: Enables the TOmp or TOmq output operation by counting operation.

d) Timer output level register m (TOLm)

	Bit q	Bit p	
TOLm	TOLmq 1/0	TOLmp 1/0	0: Positive logic output (active-high) 1: Negative logic output (active-low)

e) Timer output mode register m (TOMm)

	Bit q	Bit p	
TOMm	TOMmq 1	TOMmp 1	1: Sets the slave channel output mode.

(Note and Remark are listed on the next page.)

Note TMRm2, TMRm4, TMRm6: MASTERmp and MASTERmq bits
TMRm1, TMRm3: SPLITmp and SPLITmq bits
TMRm5, TMRm7: Fixed to 0

Remark m: Unit number (m = 0), n: Master channel number (n = 0, 2, 4)
p: Slave channel number, q: Slave channel number
n < p < q ≤ 7 (Where p and q are integers)

Figure 7 - 80 Procedure for Operations When the Multiple PWM Output Function Is to Be Used (for Two Types of PWM Output) (1/2)

	Software Operation	Hardware State
TAU default setting		Power-off state (Clock supply is stopped and writing to each register is disabled.)
	Sets the TAUmEN bit of peripheral enable register 0 (PER0) to 1. →	Power-on state. Each channel stops operating. (Clock supply is started and writing to each register is enabled.)
	Sets timer clock select register m (TPSm). Determines clock frequencies of CKm0 and CKm1.	
Channel default setting	Sets timer mode registers mn, mp, 0q (TMRmn, TMRmp, TMRmq) of each channel to be used (determines operation mode of channels). An interval (period) value is set to timer data register mn (TDRmn) of the master channel, and a duty factor is set to the TDRmp and TDRmq registers of the slave channels.	Channel stops operating. (Clock is supplied and some power is consumed.)
	Sets slave channels. The TOMmp and TOMmq bits of timer output mode register m (TOMm) are set to 1 (slave channel output mode). Sets the TOLmp and TOLmq bits. Sets the TOmp and TOmq bits and determines default level of the TOmp and TOmq outputs. →	The TOmp and TOmq pins go into Hi-Z output state. The TOmp and TOmq default setting levels are output when the port mode register is in output mode and the port register is 0.
	Sets the TOEmp and TOEmq bits to 1 and enables operation of TOmp and TOmq. →	TOmp and TOmq do not change because channels stop operating.
	Clears the port register and port mode register to 0. →	The TOmp and TOmq pins output the TOmp and TOmq set levels.

Figure 7 - 80 Procedure for Operations When the Multiple PWM Output Function Is to Be Used (for Two Types of PWM Output) (2/2)

Operation is resumed.

	Software Operation	Hardware State
Operation start	(Sets the TOEmp and TOEmq (slave) bits to 1 only when resuming operation.) The TSmn bit (master), and TSmp and TSmq (slave) bits of timer channel start register m (TSm) are set to 1 at the same time. → The TSmn, TSmp, and TSmq bits automatically return to 0 because they are trigger bits.	TEmn = 1, TEmp, TEmq = 1 When the master channel starts counting, INTTMmn is generated. Triggered by this interrupt, the slave channel also starts counting.
During operation	Set values of the TMRmn, TMRmp, TMRmq registers, TOMmn, TOMmp, TOMmq, TOLmn, TOLmp, and TOLmq bits cannot be changed. Set values of the TDRmn, TDRmp, and TDRmq registers can be changed after INTTMmn of the master channel is generated. The TCRmn, TCRmp, and TCRmq registers can always be read. The TSRmn, TSRmp, and TSR0q registers are not used.	The counter of the master channel loads the TDRmn register value to timer counter register mn (TCRmn) and counts down. When the count value reaches TCRmn = 0000H, INTTMmn output is generated. At the same time, the value of the TDRmn register is loaded to the TCRmn register, and the counter starts counting down again. At the slave channel 1, the values of the TDRmp register are transferred to the TCRmp register, triggered by INTTMmn of the master channel, and the counter starts counting down. The output levels of TOmp become active one counter clock after generation of the INTTMmn output from the master channel. It becomes inactive when TCRmp = 0000H, and the counting operation is stopped. At the slave channel 2, the values of the TDRmq register are transferred to TCRmq register, triggered by INTTMmn of the master channel, and the counter starts counting down. The output levels of TOmq become active one counter clock after generation of the INTTMmn output from the master channel. It becomes inactive when TCRmq = 0000H, and the counting operation is stopped. After that, the above operation is repeated.
Operation stop	The TTmn bit (master), TTmp, and TTmq (slave) bits are set to 1 at the same time. → The TTmn, TTmp, and TTmq bits automatically return to 0 because they are trigger bits.	TEmn, TEmp, TEmq = 0, and count operation stops. The TCRmn, TCRmp, and TCRmq registers hold count value and stop. The TOmp and TOmq outputs are not initialized and retain their current states.
	The TOEmp and TOEmq bits of slave channels are cleared to 0 and value is set to the TOmp and TOmq bits. →	The TOmp and TOmq pins output the TOmp and TOmq set levels.
TAU stop	To hold the TOmp and TOmq pin output levels Clears the TOmp and TOmq bits to 0 after the value to be held is set to the port register. → When holding the TOmp and TOmq pin output levels are not necessary Setting not required	The TOmp and TOmq pin output levels are held by port function.
	The TAUmEN bit of the PER0 register is cleared to 0. → Set the TAUmRES bit of the PRR0 register to 1 to initialize all circuits of the timer array unit. →	This stops supply of the input clock to timer array unit m. All circuits are initialized and SFR of each channel is also initialized. (The TOmp and TOmq bits are cleared to 0 and the TOmp and TOmq pins are set to port mode.)

Remark m: Unit number (m = 0), n: Master channel number (n = 0, 2, 4)
p: Slave channel number, q: Slave channel number
n < p < q ≤ 7 (Where p and q are integers)

7.10 Cautions When Using Timer Array Unit

7.10.1 Cautions when using timer output

Pins may be assigned multiplexed timer output and other alternate functions. The assignment depends on the product. If you intend to use a timer output, set the outputs from all other multiplexed pin functions to their initial values. For details, see **4.5 Register Settings When Using Alternate Function**.

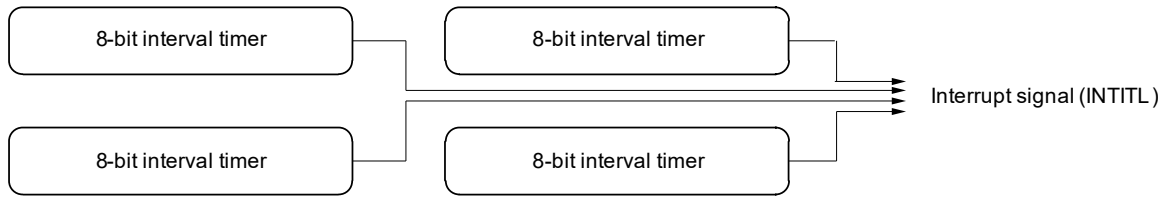
Section 8 32-bit Interval Timer (TML32)

The 32-bit interval timer consists of four 8-bit interval timers (referred to as channels 0 to 3). Each is capable of operating independently and in that case they all have the same functions. Two 8-bit interval timer channels can be connected to operate as a 16-bit interval timer. Four 8-bit interval timer channels can be connected to operate as a 32-bit interval timer.

8.1 Overview

The 32-bit interval timer operates with the f_{MXP}, f_{SXP}, or f_{IHP} clock, which is asynchronous with the CPU operation. The 32-bit interval timer has the following functions.

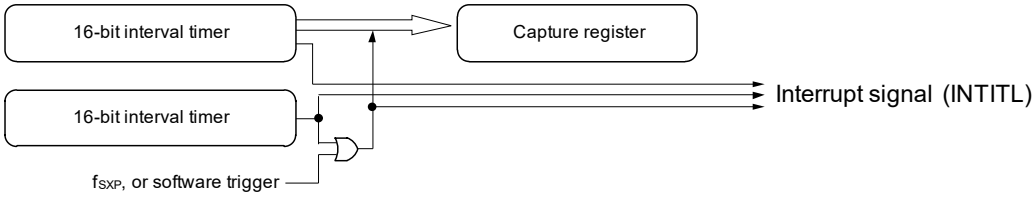
1. 8-bit counter mode
   The 8-bit timers are usable as four 8-bit interval timers that generate interrupts (INTITL) at fixed intervals.



2. 16-bit counter mode
   The 8-bit timers are usable as two 16-bit interval timers that generate interrupts (INTITL) at fixed intervals.



3. 16-bit capture mode
   The 8-bit timers are usable as two 16-bit interval timers. One is for generating the interrupts (INTITL) at fixed intervals. The other is for generating a trigger. The value of one of the 16-bit interval timers can also be stored in the capture register in response to a selected capture trigger.



4. 32-bit counter mode
   The 8-bit timers are usable as a 32-bit interval timer that generates interrupts (INTITL) at fixed intervals.

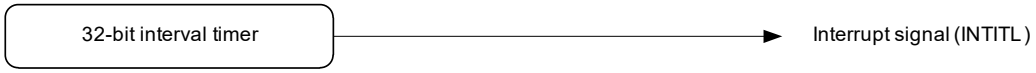


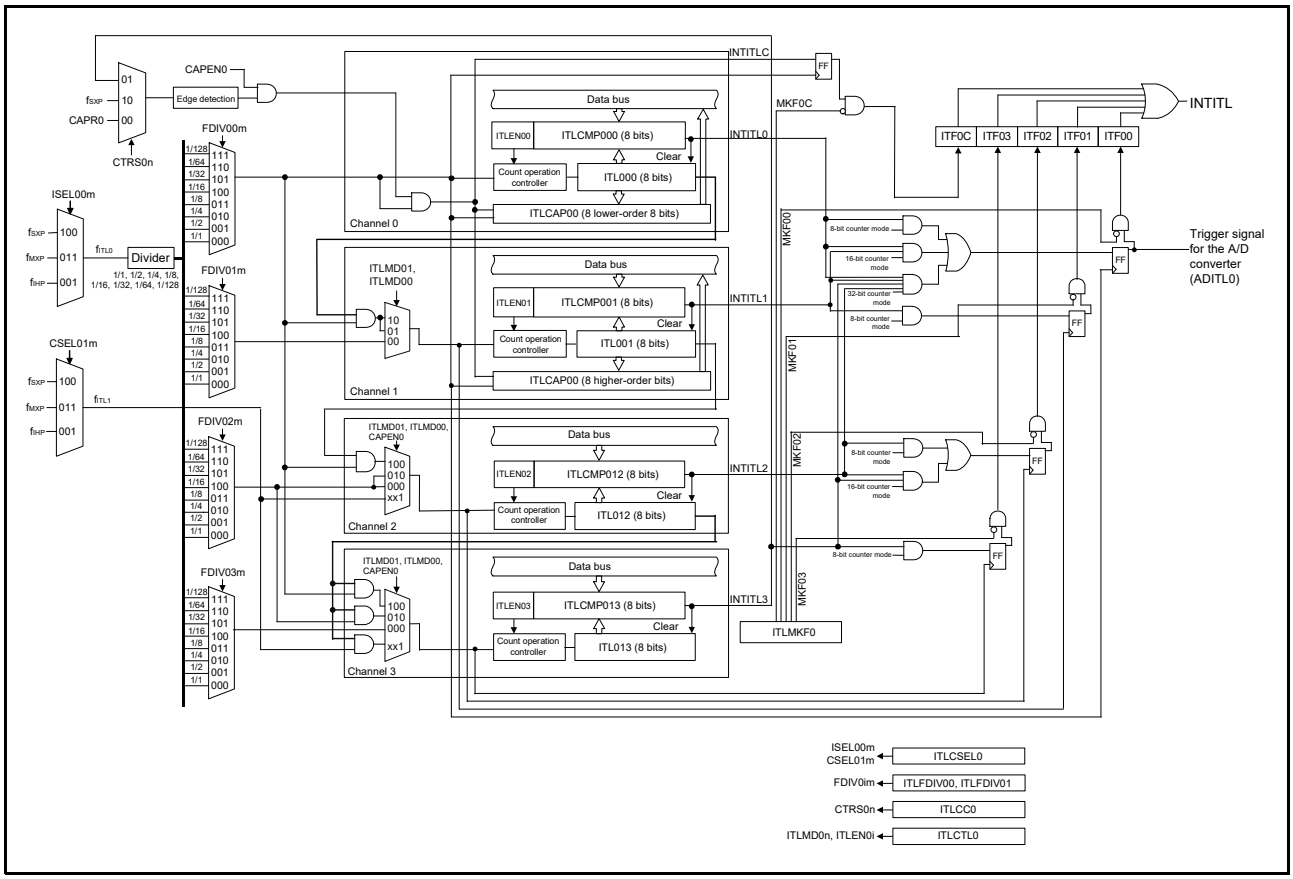
Table 8 - 1 lists the specifications of the 32-bit interval timer operations and **Figure 8 - 1** shows a block diagram of the 32-bit interval timer.

Table 8 - 1 Specifications of 32-bit Interval Timer

Item	Description
Count source (operating clock)	• f_{MXP} • f_{SXP} • f_{IHP}
Capture clock (Selectable sources for counting by the timer which can generate a capture trigger)	• f_{MXP} • f_{SXP} • f_{IHP}
Frequency division ratio	• 1/1, 1/2, 1/4, 1/8, 1/16, 1/32, 1/64, 1/128
Operating mode	• 8-bit counter mode Channels 0 to 3 independently operate as 8-bit counters. • 16-bit counter mode The combinations of channels 0 and 1 and channels 2 and 3 are cascade-connectable to operate as two 16-bit counters. • 32-bit counter mode Channels 0 to 3 are connected to operate as a 32-bit counter. • 16-bit capture mode Channels 0 and 1 are connected to operate as a 16-bit counter using the count source, channels 2 and 3 are connected to operate as a 16-bit counter using the capture clock, and the connected counters are used for capture operation.
Interrupt	• Five interrupt sources are integrated and output as the single interrupt signal (INTITL). - Output when the counter value in any of channels 0 to 3 matches the compare value. - Output when the capturing of the counter value is completed in the capture mode.

Remark f_{MXP}: High-speed peripheral clock frequency
f_{SXP}: Low-speed peripheral clock frequency
f_{IHP}: High-speed on-chip oscillator peripheral clock frequency

Figure 8 - 1 Block Diagram of 32-bit Interval Timer



Remark 1. ITL000, ITL001, ITL012, ITL013: 8-bit counters

In the 16-bit counter mode, the counters in channels 0 and 1 are connected (ITL000 + ITL001) and the counters in channels 2 and 3 are connected (ITL012 + ITL013). In the 32-bit counter mode, the counters in channels 0 to 3 are connected (ITL000 + ITL001 + ITL012 + ITL013).

ISEL00m, CSEL01m: Bits in the ITLCSEL0 register

FDIV0im: Bits in the ITLFDIV0n register

CTRS0n: Bits in the ITLCC0 register

ITLMD0n, ITLEN0i: Bits in the ITLCTL0 register

Remark 2. n = 0, 1; m = 0 to 2; i = 0 to 3

8.2 Registers for Controlling the 32-bit Interval Timer

The following registers are used to control the 32-bit interval timer.

- Peripheral enable register 1 (PER1)
- Peripheral reset control register 1 (PRR1)
- Interval timer compare registers 0mn (ITLCMP0mn)
- Interval timer compare registers 0n (ITLCMP0n)
- Interval timer capture register 00 (ITLCAP00)
- Interval timer control register (ITLCTL0)
- Interval timer clock select register 0 (ITLCSEL0)
- Interval timer frequency division register 0 (ITLFDIV00)
- Interval timer frequency division register 1 (ITLFDIV01)
- Interval timer capture control register 0 (ITLCC0)
- Interval timer status register (ITLS0)
- Interval timer match detection mask register (ITLMKF0)

8.2.1 Peripheral enable register 1 (PER1)

This register is used to enable or disable the supply of a clock signal to various on-chip peripheral modules. Clock supply to an on-chip peripheral module that is not to be used can be stopped to decrease power consumption and noise. If the 32-bit interval timer is to be used, be sure to set the TML32EN bit of this register to 1. The PER1 register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 8 - 2 Format of Peripheral Enable Register 1 (PER1)

Address: F00FAH
After reset: 00H
R/W: R/W

Symbol	7	6	5	<4>	3	<2>	1	0
PER1	0	0	0	TML32EN	0	UTAEN **Note**	0	0

TML32EN	Control of supply of an input clock to the 32-bit interval timer
0	Stops supply of an input clock. • The SFRs used by the 32-bit interval timer cannot be written.**Note**
1	Enables supply of an input clock. • The SFRs used by the 32-bit interval timer can be read and written.

Note When an SFR is read, the value returned is 00H or 0000H regardless of its setting.

Caution Be sure to set bits 7 to 5, 3, 1, and 0 to 0.

8.2.2 Peripheral reset control register 1 (PRR1)

The PRR1 register is used to control resetting of the on-chip peripheral modules. Each bit in this register controls resetting and release from the reset state of the corresponding on-chip peripheral module. Set the TML32RES bit of this register to 1 to reset the 32-bit interval timer. The PRR1 register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 8 - 3 Format of Peripheral Reset Control Register 1 (PRR1)

Address: F00FBH
After reset: 00H
R/W: R/W

Symbol	7	6	5	<4>	3	2	1	0
PRR1	0	0	0	TML32RES	0	0	0	0

TML32RES	Control resetting of the 32-bit interval timer
0	The 32-bit interval timer is released from the reset state.
1	The 32-bit interval timer is in the reset state. • The SFRs for use with the 32-bit interval timer are initialized.

Caution Be sure to set bits 7 to 5 and 3 to 0 to 0.

8.2.3 Interval timer compare registers 0mn (ITLCMP0mn)

These are compare value registers used in the 8-bit counter mode. These registers can be set by an 8-bit memory manipulation instruction. The value of each ITLCMP0mn register following a reset is FFH. A value from 01H to FFH can be specified. Setting of 00H is prohibited. These registers hold values to be compared with the ITL000 to ITL013 counter values.

Figure 8 - 4 Format of Interval Timer Compare Registers 0mn (ITLCMP0mn)

Address: F0360H (ITLCMP000), F0361H (ITLCMP001), F0362H (ITLCMP012), F0363H (ITLCMP013)
After reset: FFH
R/W: R/W**Note**

Symbol	7	6	5	4	3	2	1	0
ITLCMP0mn								

Note Write to the ITLCMP000 to ITLCMP013 registers while the settings of the ITLEN00 to ITLEN03 bits in the ITLCTL0 register are 0, respectively.

Remark mn = 00, 01, 12, 13

8.2.4 Interval timer compare registers 0n (ITLCMP0n)

These are compare value registers used in the 16-bit or 32-bit counter mode. These registers can be set by a 16-bit memory manipulation instruction. The value of each ITLCMP0n register following a reset is FFFFH. A value from 0001H to FFFFH can be specified. Setting of 0000H is prohibited. These registers hold values to be compared with the ITL0n counter values. When the setting of the ITLMD01 and ITLMD00 bits in the ITLCTL0 register is 10B, these registers are used as compare registers in the 32-bit counter mode; specify the upper 16-bit compare value in the ITLCMP01 register and the lower 16-bit compare value in the ITLCMP00 register.

Figure 8 - 5 Format of Interval Timer Compare Registers 0n (ITLCMP0n)

Address: F0360H, F0361H (ITLCMP00), F0362H, F0363H (ITLCMP01)
After reset: FFFFH
R/W: R/W**Note**

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ITLCMP0n																

Note Write to the ITLCMP00 register while the ITLEN00 bit in the ITLCTL0 register is 0. Write to the ITLCMP01 register while the ITLEN02 bit in the ITLCTL0 register is 0 in the 16-bit counter mode or while the ITLEN00 bit in the ITLCTL0 register is 0 in the 32-bit counter mode.

Remark n = 0, 1

8.2.5 Interval timer capture register 00 (ITLCAP00)

This register holds 16-bit captured values when the interval timers are operating in the 16-bit counter mode. The ITLCAP00 register can be read by a 16-bit memory manipulation instruction. The value of this register following a reset is 0000H. The values of the 16-bit counters (ITL000 + ITL001) are stored in the ITLCAP00 register in response to the capture trigger selected in the ITLCC0 register when the CAPEN0 bit in the ITLCC0 register is 1. When an interrupt on compare match with the ITLCMP01 register is to be used, select the counter clock in the ITLCSEL0 register and set the comparison value in the ITLCMP01 register.

Figure 8 - 6 Format of Interval Timer Capture Register 00 (ITLCAP00)

Address: F0364H, F0365H
After reset: 0000H
R/W: R

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ITLCAP00																

8.2.6 Interval timer control register (ITLCTL0)

This register is used to start or stop counting by the interval timer and to select the 8-bit, 16-bit, or 32-bit counter mode. This register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 8 - 7 Format of Interval Timer Control Register (ITLCTL0) (1/2)

Address: F0366H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	<3>	<2>	<1>	<0>
ITLCTL0	ITLMD01	ITLMD00	0	0	ITLEN03	ITLEN02	ITLEN01	ITLEN00

ITLMD01	ITLMD00	Selection of 8-bit, 16-bit, or 32-bit counter mode[Note 1]
0	0	The interval timer operates in the 8-bit counter mode.
0	1	The interval timer operates in the 16-bit counter mode (channel 0 is connected with channel 1 and channel 2 is connected with channel 3).
1	0	The interval timer operates in the 32-bit counter mode (channels 0 to 3 are connected).
1	1	Setting prohibited.

ITLEN03	8-bit counter mode: ITL013 count enable[Note 2]
0	Counting stops.
1	Counting begins.
In the 8-bit counter mode, writing 1 to this bit starts up-counting in the ITL013 counter and writing 0 stops it. In the 16-bit counter mode, set this bit to 0. In the 32-bit counter mode, set this bit to 0.	

ITLEN02	8-bit counter mode: ITL012 count enable[Note 2] 16-bit counter mode: ITL012 + ITL013 count enable[Note 2]
0	Counting stops.
1	Counting begins.
In the 8-bit counter mode, writing 1 to this bit starts up-counting in the ITL012 counter and writing 0 stops it. In the 16-bit counter mode, writing 1 to this bit starts up-counting in the ITL012 + ITL013 counter and writing 0 stops it. In the 32-bit counter mode, set this bit to 0.	

ITLEN01	8-bit counter mode: ITL001 count enable[Note 2]
0	Counting stops.
1	Counting begins.
In the 8-bit counter mode, writing 1 to this bit starts up-counting in the ITL001 counter and writing 0 stops it. In the 16-bit counter mode, set this bit to 0. In the 32-bit counter mode, set this bit to 0.	

Figure 8 - 7 Format of Interval Timer Control Register (ITLCTL0) (2/2)

ITLEN00	8-bit counter mode: ITL000 count enable[Note 2] 16-bit counter mode: ITL000 + ITL001 count enable[Note 2] 32-bit counter mode: ITL000 + ITL001 + ITL012 + ITL013 count enable[Note 2]
0	Counting stops.
1	Counting begins.
In the 8-bit counter mode, writing 1 to this bit starts up-counting in the ITL000 counter and writing 0 stops it. In the 16-bit counter mode, writing 1 to this bit starts up-counting in the ITL000 + ITL001 counter and writing 0 stops it. In the 32-bit counter mode, writing 1 to this bit starts up-counting in the ITL000 + ITL001 + ITL012 + ITL013 counter and writing 0 stops it.	

Note 1. To change the counter mode specified in the ITLMD01 and ITLMD00 bits, be sure to only write to the given bits while the ITLEN03 to ITLEN00 bits are all 0.

Note 2. When any of the ITLEN03 to ITLEN00 bits is cleared to 0, the corresponding counter is cleared to 0 without synchronization with the counter clock.

Mode	ITLMD01	ITLMD00	ITLEN03	ITLEN02	ITLEN01	ITLEN00	Target Counter
8-bit counter mode[Note 1]	0	0				✓	ITL000
					✓		ITL001
				✓			ITL012
			✓				ITL013
16-bit counter mode[Note 2]	0	1	Always set to 0.		Always set to 0.	✓	ITL000+ITL001
			Always set to 0.	✓	Always set to 0.		ITL012+ITL013
32-bit counter mode	1	0	Always set to 0.	Always set to 0.	Always set to 0.	✓	ITL000+ITL001+ITL012+ITL013

Note 1. Two or more bits of ITLEN03 to ITLEN00 can be set to 1 or 0 at the same time.

Note 2. The ITLEN02 and ITLEN00 bits can be set to 1 or 0 at the same time.

Remark ✓: Enables counting in the target counter.

8.2.7 Interval timer clock select register 0 (ITLCSEL0)

This register is used to select the count source for the interval timer. This register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 8 - 8 Format of Interval Timer Clock Select Register 0 (ITLCSEL0)

Address: F0367H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
ITLCSEL0	0	CSEL012	CSEL011	CSEL010	0	ISEL002	ISEL001	ISEL000

CSEL012	CSEL011	CSEL010	Selection of interval timer count clock for capturing (f_{ITL1})Note
0	0	0	Counting stops.
0	0	1	f_{IHP}
0	1	1	f_{MXP}
1	0	0	f_{SXP}
Others			Setting prohibited.

ISEL002	ISEL001	ISEL000	Selection of interval timer count clock (f_{ITL0})Note
0	0	0	Counting stops.
0	0	1	f_{IHP}
0	1	1	f_{MXP}
1	0	0	f_{SXP}
Others			Setting prohibited.

Note Be sure to only write to the CSEL012 to CSEL010 bits and ISEL002 to ISEL000 bits while the ITLEN03 to ITLEN00 bits in the ITLCTL0 register are all 0.

Remark f_{MXP}: High-speed peripheral clock frequency
f_{SXP}: Low-speed peripheral clock frequency
f_{IHP}: High-speed on-chip oscillator peripheral clock frequency

8.2.8 Interval timer frequency division register 0 (ITLFDIV00)

This register is used to select the counter clock for the interval timer. This register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 8 - 9 Format of Interval Timer Frequency Division Register 0 (ITLFDIV00)

Address: F0368H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
ITLFDIV00	0	FDIV012	FDIV011	FDIV010	0	FDIV002	FDIV001	FDIV000

FDIV012	FDIV011	FDIV010	8-bit counter mode: Counter clock for ITL001 Note 1
0	0	0	f_{ITL0}
0	0	1	$f_{ITL0}/2$
0	1	0	$f_{ITL0}/4$
0	1	1	$f_{ITL0}/8$
1	0	0	$f_{ITL0}/16$
1	0	1	$f_{ITL0}/32$
1	1	0	$f_{ITL0}/64$
1	1	1	$f_{ITL0}/128$
In the 8-bit counter mode, ITL001 counts cycles of the counter clock specified in the FDIV012 to FDIV010 bits. In the 16-bit counter mode, set these bits to 000B. In the 32-bit counter mode, set these bits to 000B.			

FDIV002	FDIV001	FDIV000	8-bit counter mode: Counter clock for ITL000 Note 2 16-bit counter mode: Counter clock for ITL000 + ITL001 Note 2 32-bit counter mode: Counter clock for ITL000 + ITL001 + ITL012 + ITL013 Note 2
0	0	0	f_{ITL0}
0	0	1	$f_{ITL0}/2$
0	1	0	$f_{ITL0}/4$
0	1	1	$f_{ITL0}/8$
1	0	0	$f_{ITL0}/16$
1	0	1	$f_{ITL0}/32$
1	1	0	$f_{ITL0}/64$
1	1	1	$f_{ITL0}/128$
In the 8-bit counter mode, ITL000 counts cycles of the counter clock specified in the FDIV002 to FDIV000 bits. In the 16-bit counter mode, ITL000 + ITL001 counts cycles of the counter clock specified in the FDIV002 to FDIV000 bits. In the 32-bit counter mode, ITL000 +ITL001 + ITL012 + ITL013 counts cycles of the counter clock specified in the FDIV002 to FDIV000 bits.			

Note 1. In the 8-bit counter mode, be sure to only write to the FDIV012 to FDIV010 bits while the ITLEN01 bit in the ITLCTL0 register is 0.

Note 2. Be sure to only write to the FDIV002 to FDIV000 bits while the ITLEN00 bit in the ITLCTL0 register is 0.

8.2.9 Interval timer frequency division register 1 (ITLFDIV01)

This register is used to select the counter clock for the interval timer. This register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 8 - 10 Format of Interval Timer Frequency Division Register 1 (ITLFDIV01)

Address: F0369H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
ITLFDIV01	0	FDIV032	FDIV031	FDIV030	0	FDIV022	FDIV021	FDIV020

FDIV032	FDIV031	FDIV030	8-bit counter mode: Counter clock for ITL013Note 1
0	0	0	f_{ITL0}
0	0	1	$f_{ITL0}/2$
0	1	0	$f_{ITL0}/4$
0	1	1	$f_{ITL0}/8$
1	0	0	$f_{ITL0}/16$
1	0	1	$f_{ITL0}/32$
1	1	0	$f_{ITL0}/64$
1	1	1	$f_{ITL0}/128$
In the 8-bit counter mode, ITL013 counts cycles of the counter clock specified in the FDIV032 to FDIV030 bits. In the 16-bit counter mode, set these bits to 000B. In the 32-bit counter mode, set these bits to 000B.			

FDIV022	FDIV021	FDIV020	8-bit counter mode: Counter clock for ITL012Note 2 16-bit counter mode: Counter clock for ITL012 and ITL013Note 2
0	0	0	f_{ITL0}
0	0	1	$f_{ITL0}/2$
0	1	0	$f_{ITL0}/4$
0	1	1	$f_{ITL0}/8$
1	0	0	$f_{ITL0}/16$
1	0	1	$f_{ITL0}/32$
1	1	0	$f_{ITL0}/64$
1	1	1	$f_{ITL0}/128$
In the 8-bit counter mode, ITL012 counts cycles of the counter clock specified in the FDIV022 to FDIV020 bits. In the 16-bit counter mode, ITL012 + ITL013 counts cycles of the counter clock specified in the FDIV022 to FDIV020 bits. In the 32-bit counter mode, these bits are not used; write 000B to them.			

Note 1. In the 8-bit counter mode, be sure to only write to the FDIV032 to FDIV030 bits while the ITLEN03 bit in the ITLCTL0 register is 0.

Note 2. In the 8-bit or 16-bit counter mode, be sure to only write to the FDIV022 to FDIV020 bits while the ITLEN02 bit in the ITLCTL0 register is 0.

8.2.10 Interval timer capture control register 0 (ITLCC0)

This register is used to enable or disable the capture function of the interval timer, specify whether to hold or clear the capture completion flag, set up the software trigger, and select the capture trigger. This register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 8 - 11 Format of Interval Timer Capture Control Register 0 (ITLCC0)

Address: F036AH
After reset: 00H
R/W: R/W

Symbol	7	<6>	5	<4>	3	2	1	0
ITLCC0	CAPEN0	CAPF0CR	CAPF0	CAPR0	CAPC0CR	0	CTRS01	CTRS00

CAPEN0	Capture enable[Note 1]
0	Capturing is disabled.
1	Capturing is enabled.

CAPF0CR	Capture completion flag clear[Note 2]
0	The value of the capture completion flag CAPF0 is held.
1	The value of the capture completion flag CAPF0 is cleared.

CAPF0	Capture completion flag[Note 3]
0	Capturing has not been completed.
1	Capturing has been completed. This flag is set to 1 after generation of the capture trigger selected in the CTRS01 and CTRS00 bits and capture of the data in the ITLCAP00 register. Writing 1 to the CAPF0CR bit clears this flag to 0.

CAPR0	Software capture trigger[Notes 4, 5]
0	Trigger operation does not proceed.
1	A software trigger for capturing is generated.

CAPC0CR	Selection of the 16-bit counter (ITL000 + ITL001) clearing after capturing[Note 6]
0	The 16-bit counter (ITL000 + ITL001) is retained after the completion of capturing.
1	The 16-bit counter (ITL000 + ITL001) is cleared after the completion of capturing.

CTRS01	CTRS00	Selection of capture trigger[Notes 5, 7]
0	0	Software trigger
0	1	Interrupt on compare match with the ITLCMP01 register[Note 8]
1	0	fSXP (rising edge)
1	1	Setting prohibited

(Notes are listed on the next page.)

Note 1. Be sure to only write to the CAPEN0 bit while the ITLEN03 to ITLEN00 bits in the ITLCTL0 register are all 0.

Note 2. The CAPF0CR bit is always read as 0.

Note 3. Only bit 5 is readable.

Note 4. The CAPR0 bit is always read as 0.

Note 5. In the capture operation, the interval at which the capture trigger is generated should be two or more cycles of the counter clock.

Note 6. Be sure to only write to the CAPC0CR bit while the ITLEN03 to ITLEN00 bits in the ITLCTL0 register are all 0.

Note 7. Be sure to only write to the CTRS01 and CTRS00 bits while the ITLEN03 to ITLEN00 bits in the ITLCTL0 register are all 0.

Note 8. When the interrupt on compare match with ITLCMP01 is selected as a capture trigger, the compare match detection flag for channel 2 (ITF02) and capture detection flag (ITF0C) are set on capture of the counter value. When only using the capture detection flag, set the ITLMKF0 register to mask the compare match detection flag for channel 2.

8.2.11 Interval timer status register (ITLS0)

This is a status register for the interval timer. This register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 00H. When the value of the ITL0mn register (mn = 00, 01, 12, 13) matches the value specified in the ITLCMP0mn, ITLCMP00, and ITLCMP01 registers, the compare-match flag for the corresponding channel is set. When a capture trigger is generated while the CAPEN0 bit in the ITLCC0 register is 1, the capture detection flag is set after the value of the ITL0n counter is stored in the ITLCAP00 register. The values of the ITF0C and ITF03 to ITF00 bits in this register are ORed and output as the interrupt request signal (INTITL). **Table 8 - 2** shows the conditions for setting the status flags in each counter mode selected by the ITLMD01 and ITLMD00 bits.

Figure 8 - 12 Format of Interval Timer Status Register (ITLS0)

Address: F036BH
After reset: 00H
R/W: R/W**Note**

Symbol	7	6	5	4	3	2	1	0
ITLS0	0	0	0	ITF0C	ITF03	ITF02	ITF01	ITF00

ITF0C	Capture detection flag
0	Completion of capturing has not been detected.
1	Completion of capturing has been detected.

ITF03	Compare match detection flag for channel 3
0	A compare match signal has not been detected in channel 3.
1	A compare match signal has been detected in channel 3.

ITF02	Compare match detection flag for channel 2
0	A compare match signal has not been detected in channel 2.
1	A compare match signal has been detected in channel 2.

ITF01	Compare match detection flag for channel 1
0	A compare match signal has not been detected in channel 1.
1	A compare match signal has been detected in channel 1.

ITF00	Compare match detection flag for channel 0
0	A compare match signal has not been detected in channel 0.
1	A compare match signal has been detected in channel 0.

Note Writing 1 to each bit is ignored. To clear the ITF0C or ITF0i bit (i = 0 to 3), write 0 to the desired bit and 1 to the other bits by using an 8-bit memory manipulation instruction.

Caution 1. If clearing any of the ITF0C, ITF03, ITF02, ITF01, ITF00 flag bits to 0 does not lead to the value of the ITLS0 register becoming 00H, an interrupt request (INTITL) is generated and the interrupt request flag (ITLIF) is set to 1.

Caution 2. To clear a flag bit in the ITLS0 register to 0, only write 0 to a bit that has the setting 1. This is because writing 0 to a bit that has the setting 0 may make detecting a compare match signal or capture detection signal generated at the same time as the writing of 0 impossible. For example, when the ITF01 flag bit is set to 1, write 00011101B to the ITLS0 register to clear the ITF01 flag bit.

Table 8 - 2 Conditions for Setting the Status Flags in Each Counter Mode

Mode	ITLMD01	ITLMD00	CAPEN0	Status Flag	Conditions for Setting Status Flag
8-bit counter mode	0	0	×	ITF00	The next rising edge of the counter clock following a match between the ITLCMP000 and ITL000 values
			×	ITF01	The next rising edge of the counter clock following a match between the ITLCMP001 and ITL001 values
			×	ITF02	The next rising edge of the counter clock following a match between the ITLCMP012 and ITL012 values
			×	ITF03	The next rising edge of the counter clock following a match between the ITLCMP013 and ITL013 values
16-bit counter mode	0	1	×	ITF00	The next rising edge of the counter clock following a match between the ITLCMP00 and ITL000 + ITL001 values
			×	ITF02	The next rising edge of the counter clock following a match between the ITLCMP01 and ITL012 + ITL013 values
			1	ITF0C	The ITL000 + ITL001 value is stored in ITLCAP00 after a capture trigger is generated.
32-bit counter mode	1	0	×	ITF00	The next rising edge of the counter clock following a match between the ITLCMP00 + ITLCMP01 and ITL000 + ITL001 + ITL012 + ITL013 values

Remark ×: Don’t care

8.2.12 Interval timer match detection mask register (ITLMKF0)

This register is used to enable or disable setting of each effective bit in the interval timer status register (ITLS0) to 1. This register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 00H. Setting an MKF0C or MKF03 to MKF00 bit to 1 masks the corresponding status flag (ITF0C, ITF03 to ITF00), after which the given flag is not set to 1 even if a compare match with a compare register or capture completion is detected. Since the status flag will not be set to 1, masking also prevents generation of the interval detection interrupt (INTITL).

Figure 8 - 13 Format of Interval Timer Match Detection Mask Register (ITLMKF0)

Address: F036CH
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
ITLMKF0	0	0	0	MKF0C	MKF03	MKF02	MKF01	MKF00

MKF0C	Mask for capture detection status flag[Note]
0	The ITF0C flag is not masked.
1	The ITF0C flag is masked.

MKF03	Mask for compare match status flag for channel 3[Note]
0	The ITF03 flag is not masked.
1	The ITF03 flag is masked.

MKF02	Mask for compare match status flag for channel 2[Note]
0	The ITF02 flag is not masked.
1	The ITF02 flag is masked.

MKF01	Mask for compare match status flag for channel 1[Note]
0	The ITF01 flag is not masked.
1	The ITF01 flag is masked.

MKF00	Mask for compare match status flag for channel 0[Note]
0	The ITF00 flag is not masked.
1	The ITF00 flag is masked.

Note Setting all functional bits to 1 for masking prevents setting of the corresponding bits in the ITLS0 register. This in turn prevents software detection of compare matches and completion of capture. When compare match for any of channels 0 to 3 is to be used, be sure to set the bit corresponding to the given status flag to 0 so that the flag is not masked. For the state of completion of capture, on the other hand, the CAPF0 flag in the interval timer capture control register 0 (ITLCC0) can be used to detect this even when the MKF0C bit is set to 1 to mask the ITF0C flag.

8.3 Operation

8.3.1 Counter mode settings

The 32-bit interval timer has three counter modes: the 8-bit counter mode, 16-bit counter mode, and 32-bit counter mode. **Table 8 - 3** to **Table 8 - 5** show the registers and settings for use in the 8-bit counter mode, 16-bit counter mode, and 32-bit counter mode, respectively.

Table 8 - 3 Registers and Settings Used in the 8-bit Counter Mode

Register	Bit	Setting
Interval timer compare registers 0mn (ITLCMP0mn)	Bits 7 to 0	Specify 8-bit compare values for channels 0 to 3.
Interval timer control register 0 (ITLCTL0)	ITLEN00	Specify whether to start or stop counting in channel 0.
	ITLEN01	Specify whether to start or stop counting in channel 1.
	ITLEN02	Specify whether to start or stop counting in channel 2.
	ITLEN03	Specify whether to start or stop counting in channel 3.
	ITLMD00	Set to 0.
	ITLMD01	Set to 0.
Interval timer frequency division registers n (ITLFDIV0n)	FDIV002 to FDIV000	Select the count clock for channel 0.
	FDIV012 to FDIV010	Select the count clock for channel 1.
	FDIV022 to FDIV020	Select the count clock for channel 2.
	FDIV032 to FDIV030	Select the count clock for channel 3.
Interval timer clock select register 0 (ITLCSEL0)	ISEL002 to ISEL000	Select the count clock for the interval timer.
	CSEL012 to CSEL010	Set to 000B.
Interval timer capture control register 0 (ITLCC0)	Bits 7 to 0	Set to 0.

Remark mn = 00, 01, 12, 13; n = 0, 1

Table 8 - 4 Registers and Settings Used in the 16-bit Counter Mode

Register	Bit	Setting
Interval timer compare registers 0n (ITLCMP0n)	Bits 15 to 0	Specify 16-bit compare values for channels 0 and 1, and channels 2 and 3.
Interval timer control register 0 (ITLCTL0)	ITLEN00	Specify whether to start or stop counting in channels 0 and 1.
	ITLEN01	Set to 0.
	ITLEN02	Specify whether to start or stop counting in channels 2 and 3.
	ITLEN03	Set to 0.
	ITLMD00	Set to 1.
	ITLMD01	Set to 0.
Interval timer frequency division registers n (ITLFDIV0n)	FDIV002 to FDIV000	Select the count clock for channels 0 and 1.
	FDIV012 to FDIV010	Set to 000B.
	FDIV022 to FDIV020	Select the count clock for channels 2 and 3.
	FDIV032 to FDIV030	Set to 000B.
Interval timer clock select register 0 (ITLCSEL0)	ISEL002 to ISEL000	Select the count clock for the interval timer.
	CSEL012 to CSEL010	Set to 000B.
Interval timer capture control register 0 (ITLCC0)	Bits 7 to 0	Set to 0.

Remark n = 0, 1

Table 8 - 5 Registers and Settings Used in the 32-bit Counter Mode

Register	Bit	Setting
Interval timer compare registers 0n (ITLCMP0n)	Bits 15 to 0	Specify a compare value in the 32-bit counter mode. Specify the lower 16 bits of the compare value in channels 0 and 1 (ITLCMP00) and the upper 16 bits of the compare value in channels 2 and 3 (ITLCMP01).
Interval timer control register 0 (ITLCTL0)	ITLEN00	Specify whether to start or stop counting in channels 0 to 3.
	ITLEN01	Set to 0.
	ITLEN02	Set to 0.
	ITLEN03	Set to 0.
	ITLMD00	Set to 0.
	ITLMD01	Set to 1.
Interval timer frequency division registers n (ITLFDIV0n)	FDIV002 to FDIV000	Select the count clock for channels 0 to 3.
	FDIV012 to FDIV010	Set to 000B.
	FDIV022 to FDIV020	Set to 000B.
	FDIV032 to FDIV030	Set to 000B.
Interval timer clock select register 0 (ITLCSEL0)	ISEL002 to ISEL000	Select the count clock for the interval timer.
	CSEL012 to CSEL010	Set to 000B.
Interval timer capture control register 0 (ITLCC0)	Bits 7 to 0	Set to 0.

Remark n = 0, 1

8.3.2 Capture mode settings

When the 16-bit capture mode is to be used for channels 0 and 1, the counter value is stored in interval timer capture register 00 (ITLCAP00) in response to a selected capture trigger.

Table 8 - 6 Registers and Settings Used in the 16-bit Capture Mode

Register	Bit	Setting
Interval timer compare register 00 (ITLCMP00)	Bits 15 to 0	Specify 16-bit compare values for channels 0 and 1.
Interval timer compare register 01 (ITLCMP01)**Note**	Bits 15 to 0	Specify 16-bit compare values for channels 2 and 3.
Interval timer control register 0 (ITLCTL0)	ITLEN00	Specify whether to start or stop counting in channels 0 and 1.
	ITLEN01	Set to 0.
	ITLEN02	Specify whether to start or stop counting in channels 2 and 3.
	ITLEN03	Set to 0.
	ITLMD00	Set to 1.
	ITLMD01	Set to 0.
Interval timer frequency division registers n (ITLFDIV0n)	FDIV002 to FDIV000	Select the count clock for channel 0.
	FDIV012 to FDIV010	Set to 000B.
	FDIV022 to FDIV020	Set to 000B.
	FDIV032 to FDIV030	Set to 000B.
Interval timer clock select register 0 (ITLCSEL0)	ISEL002 to ISEL000	Select the count clock for the interval timer in channels 0 and 1.
	CSEL012 to CSEL010	Select the count clock for the interval timer for capturing in channels 2 and 3.
Interval timer capture control register 0 (ITLCC0)	CAPEN0	Set to 1.
	CAPC0CR	Specify whether to clear or hold the counter value in channels 0 and 1 after the completion of capturing.
	CTRS01 and CTRS00	Select a capture trigger.

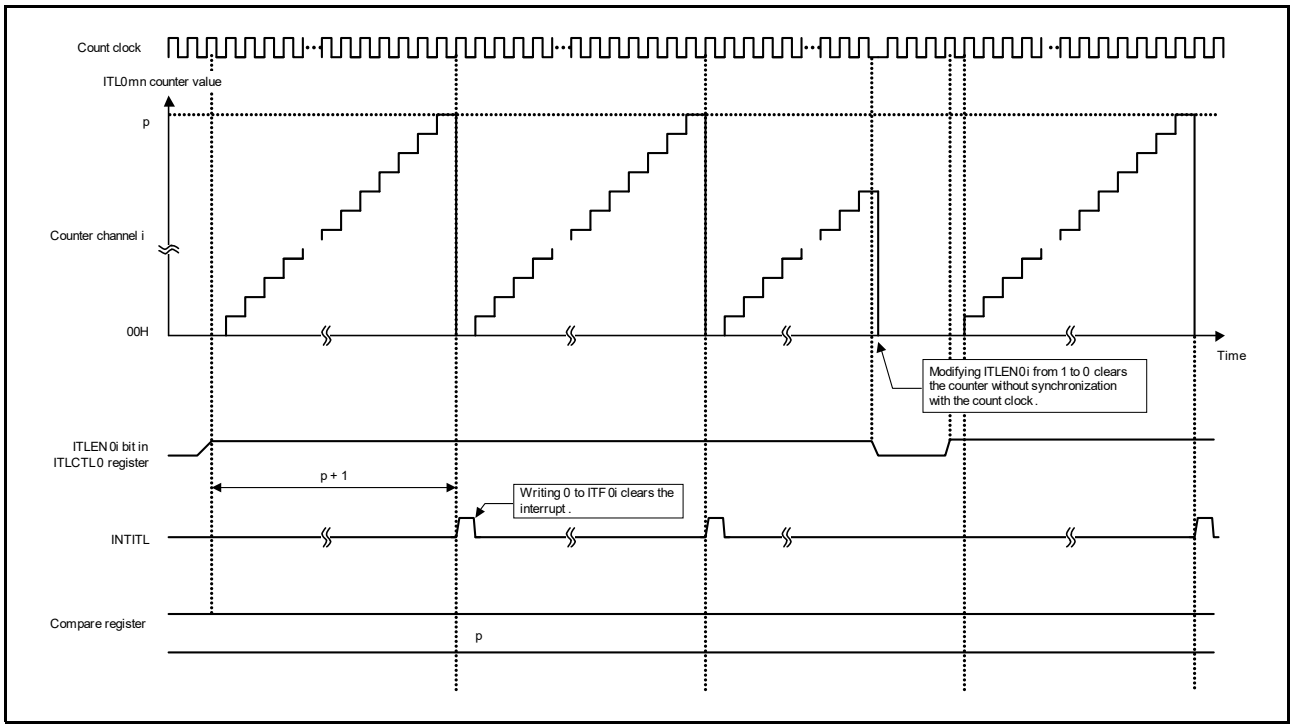
Note Channels 2 and 3 can only be used in the 16-bit counter mode when an interrupt on compare match with ITLCMP01 is not to be used as a capture trigger.

Remark n = 0, 1

8.3.3 Timer operation

The ITL0mn counter counts up cycles of the counting clock specified in the interval timer frequency division registers 0 and 1 (ITLFDIV00 and ITLFDIV01). An interrupt request signal (INTITL) is generated on the counting of the next clock cycle after the value of the counter matches the comparison value. The interrupt request signal (INTITL) remains high until the value of the ITLS0 register becomes 00H. While the interrupt request signal (INTITL) is high, neither the generation of a further interrupt request (INTITL) nor setting of the interrupt request flag (ITLIF) proceeds even if a compare match or capture completion is detected for an operating channel. Clearing the ITLEN03 to ITLEN00 bits to 0 clears the counter value.

Figure 8 - 14 Example of Timer Operation in the 8-bit Counter Mode



Remark 1. mn = 00, 01, 12, 13; i = 0 to 3

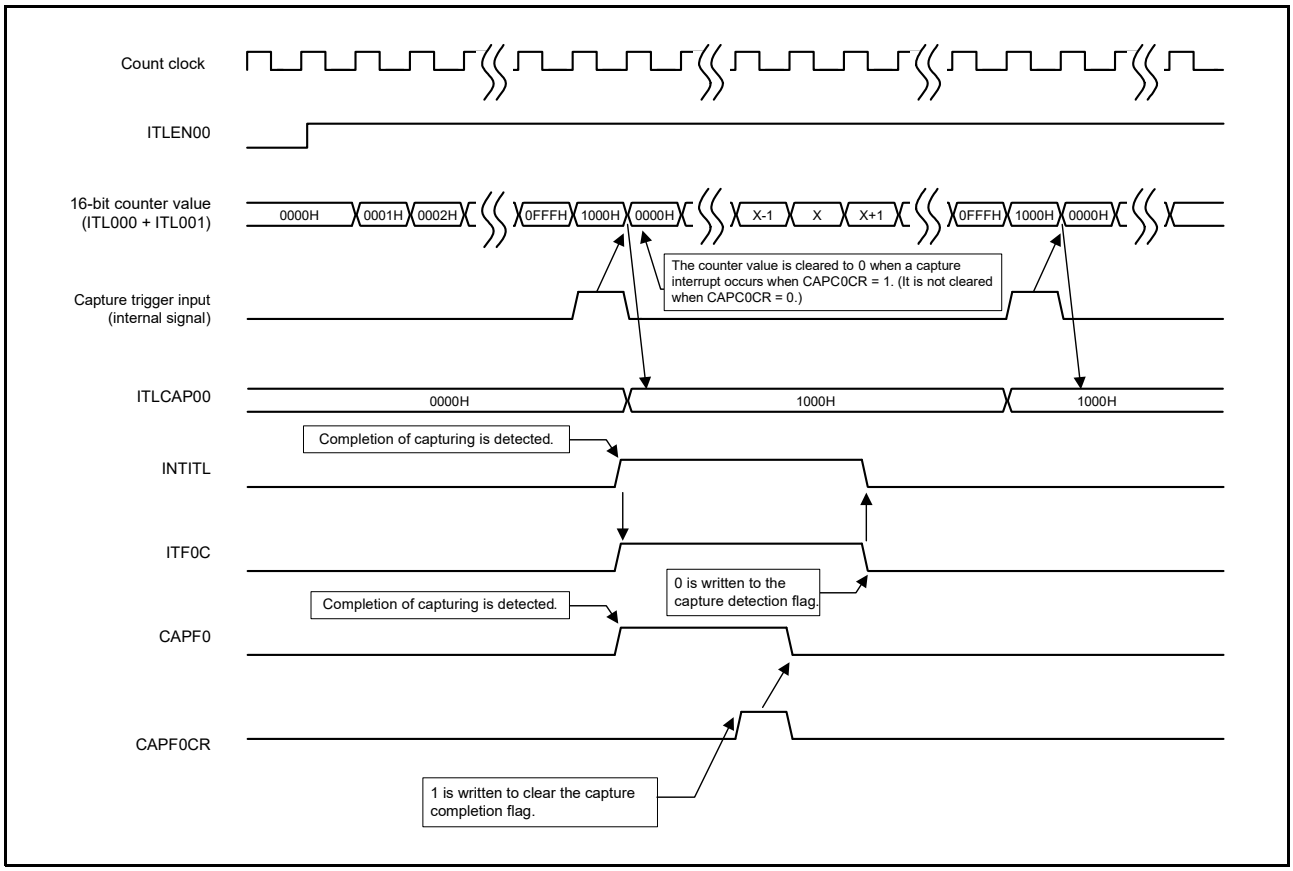
Remark 2. p: Set value of the ITLCMP0mn register

8.3.4 Capture operation

When the setting of the CAPEN0 bit in the interval timer capture control register 0 (ITLCC0) is 1, the values in the 16-bit counters (ITL000 and ITL001) are stored in interval timer capture register 00 (ITLCAP00) in response to the capture trigger specified in the ITLCC0 register. The capture trigger is selectable from among the interrupt on compare match with the ITLCMP01 register, f_{SXP}, and a software trigger (setting the CAPR0 bit to 1). To use the interrupt on compare match with the ITLCMP01 register as the capture trigger, set interval timer clock select register 0 (ITLCSEL0) to select the clock for counting, and set interval timer compare register 01 (ITLCMP01) to specify the comparison value. When using f_{SXP} or a software trigger (setting the CAPR0 bit to 1) as a capture trigger, channels 2 and 3 can be used in the 16-bit counter mode. After a capture trigger is input and the counter value is stored in interval timer capture register 00, the interrupt request signal (INTITL) is output, the capture completion flag (CAPF0) and capture detection flag (ITF0C) are set to 1, and the flag values are retained until they are explicitly cleared **Note**. The CAPF0 flag can be cleared by setting the CAPF0CR bit to 1. The ITF0C flag in the ITLS0 register can be cleared by writing 0 to it. Since capture operations operate with the counter clock, the interval at which the capture trigger is generated should be at least five cycles of the counter clock. If a capture trigger is generated again within two cycles of the counter clock after an earlier capture trigger was generated, the CAPF0 bit may not be set.

Note If the value of the ITLS0 register is other than 00H, neither interrupt operation nor setting of the interrupt request flag (ITLIF) will proceed even when the ITF0C flag is set to 1 because the interrupt request signal (INTITL) is kept at the high level.

Figure 8 - 15 Example of Capture Operation



When the counter value matches the comparison value while the CAPC0CR bit in the ITLCC0 register is set to 1 (the mode where the 16-bit counter (ITL000 + ITL001) is cleared following the completion of capture), counting of the next clock cycle clears the counter value. Note that the ITF00 flag is set when the 16-bit counter (ITL000 + ITL001) matches the comparison value before a capture trigger is input. The counter value is not cleared in this way if the CAPC0CR bit is set to 0 (the mode where the 16-bit counter (ITL000 + ITL001) retains its value after the completion of capture). The ITF00 flag is set when the 16-bit counter (ITL000 + ITL001) matches the comparison value.

8.3.5 Interrupt

Table 8 - 7 shows the interrupt sources in the 8-bit, 16-bit, and 32-bit counter modes. The ITF0C and ITF03 to ITF00 bits are interrupt status flags in the ITLS0 register. When any of the interrupt status flag is set, an interrupt request signal (INTITL) is output.

Table 8 - 7 Interrupt Sources in the 8-bit, 16-bit, and 32-bit Counter Modes

Interrupt Source	Interrupt Condition in the 8-bit Counter Mode	Interrupt Condition in the 16-bit Counter Mode	Interrupt Condition in the 32-bit Counter Mode
ITF00	Next rising edge of the counter clock after a compare match in channel 0	Next rising edge of the counter clock after a compare match in channels 0 and 1	Next rising edge of the counter clock after a compare match
ITF01	Next rising edge of the counter clock after a compare match in channel 1	Not generated	Not generated
ITF02	Next rising edge of the counter clock after a compare match in channel 2	Next rising edge of the counter clock after a compare match in channels 2 and 3	Not generated
ITF03	Next rising edge of the counter clock after a compare match in channel 3	Not generated	Not generated
ITF0C	Not generated; this is the case when the setting of ITLCC0 is 00H.	Timing of storing the counter value in the capture register after a capture trigger is input	Not generated; this is the case when the setting of ITLCC0 is 00H.

If the value of the ITLS0 register is other than 00H, the interrupt request signal (INTITL) is kept at the high level. Accordingly, neither the generation of a further interrupt request (INTITL) nor setting of the interrupt request flag (ITLIF) will proceed, even when a compare match or completion of capture is detected for an operating channel. However, if the value of the ITLS0 register is not 00H after any bit in the ITLS0 register is set to 0 by an 8-bit memory manipulation instruction, a low-level pulse signal is output on the INTITL pin and the interrupt request flag (ITLIF) is set to 1. Accordingly, clearing a status flag in the ITLS0 register to 0 while ITLIF = 0 during vector interrupt processing or other processing enables the detection of an interrupt in response to another status bit having the setting 1. **Figure 8 - 16** shows the relationship between clearing of the detection flags and the interval detection interrupt signal.

The following describes the operation shown in **Figure 8 - 16**. When a compare match in channel 1 is detected while the value of the ITLS0 register is 00H, the ITF01 flag is set to 1 and the interval detection interrupt signal (INTITL) is driven high. While the interval detection interrupt signal (INTITL) is kept at the high level, neither the generation of a further interrupt request (INTITL) nor setting of the interrupt request flag (ITLIF) will proceed, even when a compare match or completion of capture is detected for an operating channel. Note that if another detection flag is set to 1 immediately before clearing the ITF0x (x = 0 to 3, C) flag bit to 0, the output of the INTITL pin temporarily goes to the low level after clearing of the given ITF0x flag bit, leading to setting of the interrupt request flag (ITLIF) to 1.

<1> The ITF01 flag is set to 1 in response to a compare match in channel 1 and the interval detection interrupt signal (INTITL) and interrupt request flag (ITLIF) are driven high. The interval detection interrupt processing is executed after clearing of the interrupt request flag (ITLIF) to 0.

<2> Check which detection flag in the ITLS0 register is set to 1 from within the interval detection interrupt processing. In the case shown in **Figure 8 - 16**, the ITF01 and ITF00 flags being set to 1 can be confirmed.

<3> Clear the ITF01 and ITF00 flags detected in step 2 by using an 8-bit memory manipulation instruction to write 00011100B to the ITLS0 register so that its value becomes 00H.**Note**

<4> The respective processing sequences in response to the ITF01 and ITF00 flags being set to 1 are then executed.**Note**

<5> The ITF01 flag is set to 1 in response to a further compare match in channel 1 and the interval detection interrupt signal (INTITL) and interrupt request flag (ITLIF) are driven high. The interval detection interrupt processing is executed after clearing of the interrupt request flag (ITLIF) to 0.

<6> Check which detection flag in the ITLS0 register is set to 1 from within the interval detection interrupt processing. In the case shown in **Figure 8 - 16**, the ITF01 flag being set to 1 can be confirmed.

<7> Clear the ITF01 flag detected in step 6 by using an 8-bit memory manipulation instruction to write 00011101B to the ITLS0 register so that its value becomes 00H. Though the ITF00 flag is also set to 1 in response to the compare match in channel 0 at this time, the ITF00 flag is not cleared because the processing for the flag does not proceed.

<8> As the ITF00 flag is set to 1 at the time the ITF01 flag is cleared to 0 in step 7, the INTITL signal is temporarily driven low and the interrupt request flag (ITLIF) is set to 1. If the interrupt enable flag (IE) is not cleared to 0 at this time, this interrupt request is held pending.

<9> The processing in response to the ITF01 flag being set to 1 is then executed.

<10> As the interrupt request flag (ITLIF) is still set to 1 after return from the processing in response to the ITF01 flag being set to 1, clear the interrupt request flag (ITLIF) to 0, after which the interval detection interrupt processing that was held pending proceeds.

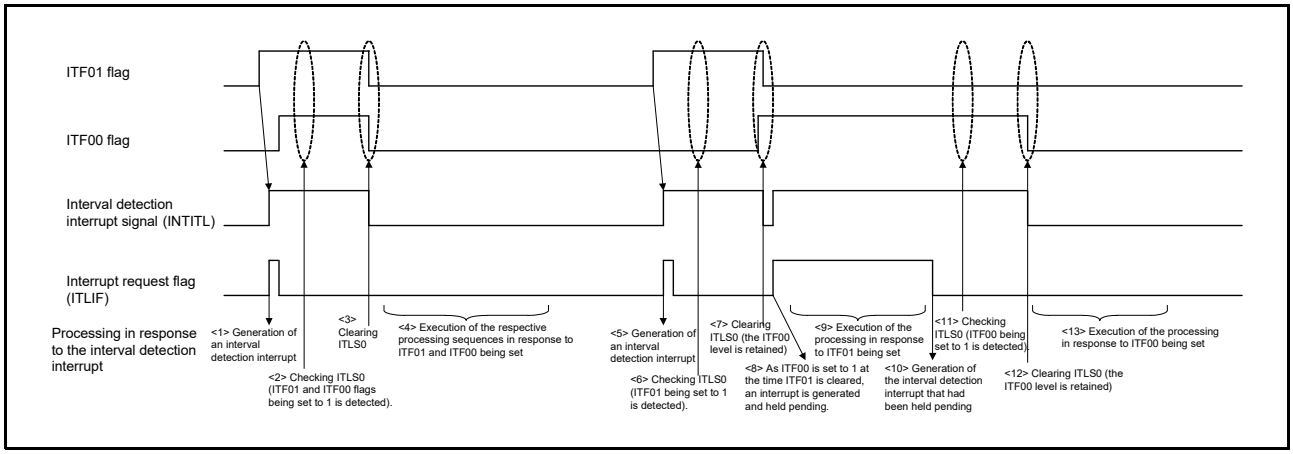
<11> Check which detection flag in the ITLS0 register is set to 1 from within the interval detection interrupt processing. In the case shown in **Figure 8 - 16**, the ITF00 flag being set to 1 can be confirmed.

<12> Clear the ITF00 flag detected in step 11 by using an 8-bit memory manipulation instruction to write 00011101B to the ITLS0 register so that its value becomes 00H.

<13> The processing in response to the ITF00 flag being set to 1 is then executed.

Note Missing an interrupt source can also be prevented by repeating the processing for clearing an interrupt source per flag.

Figure 8 - 16 Example of Clearing the Detected Flags



8.3.6 Interval timer setting procedures

The following shows the procedures for setting up the 32-bit interval timer.

Figure 8 - 17 Procedure for Starting the 32-bit Interval Timer

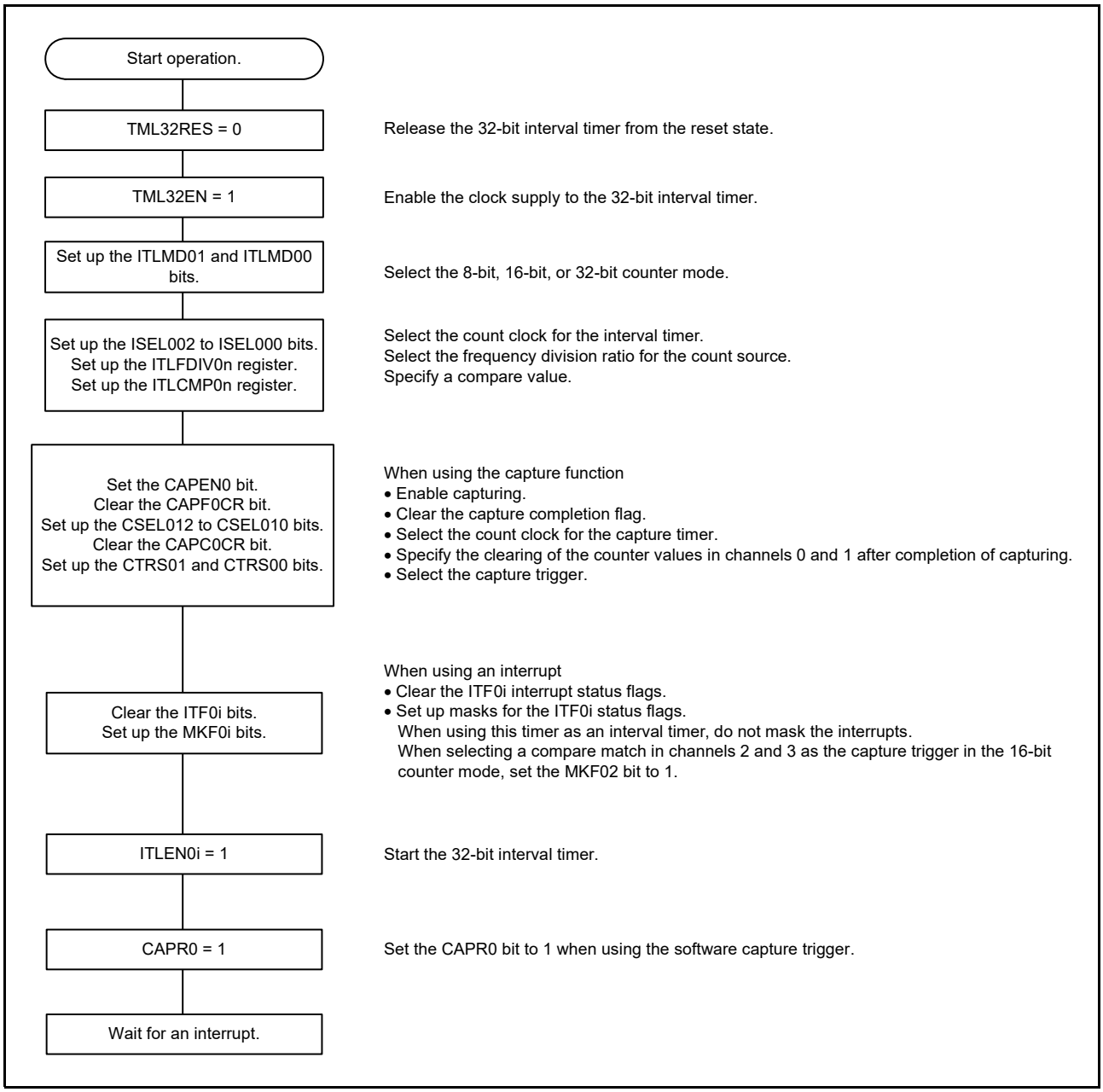


Figure 8 - 18 Procedure for Stopping the 32-bit Interval Timer

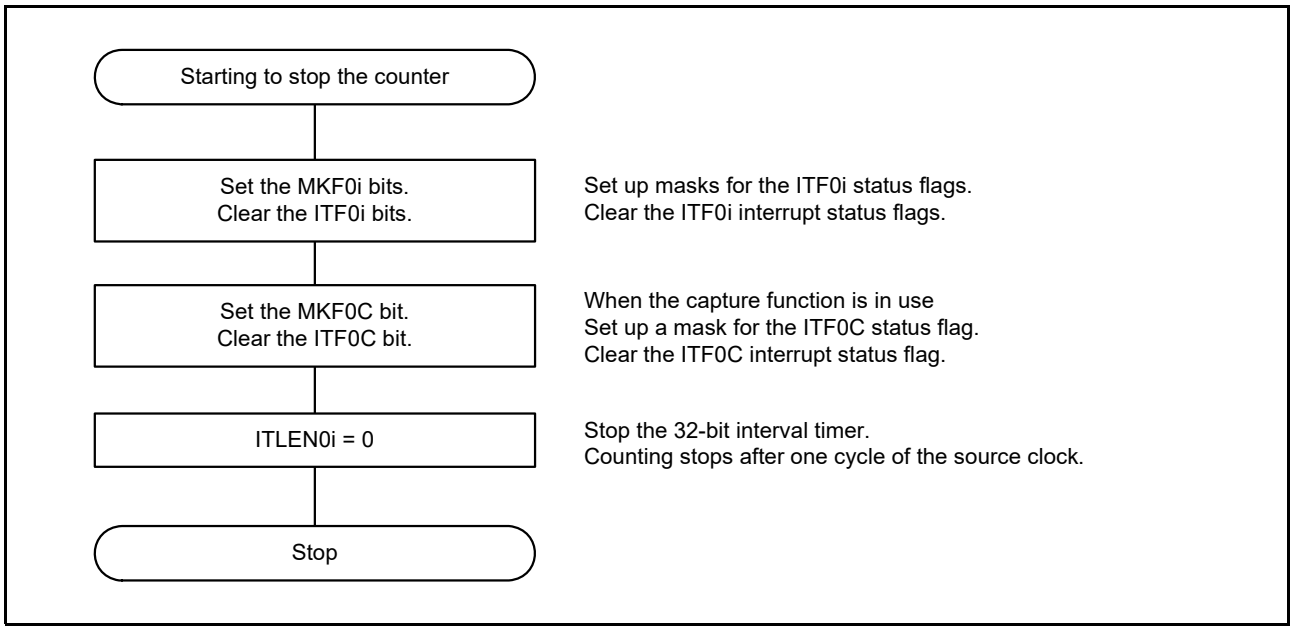


Figure 8 - 19 Procedure for Changing the Operating Mode of the 32-bit Interval Timer

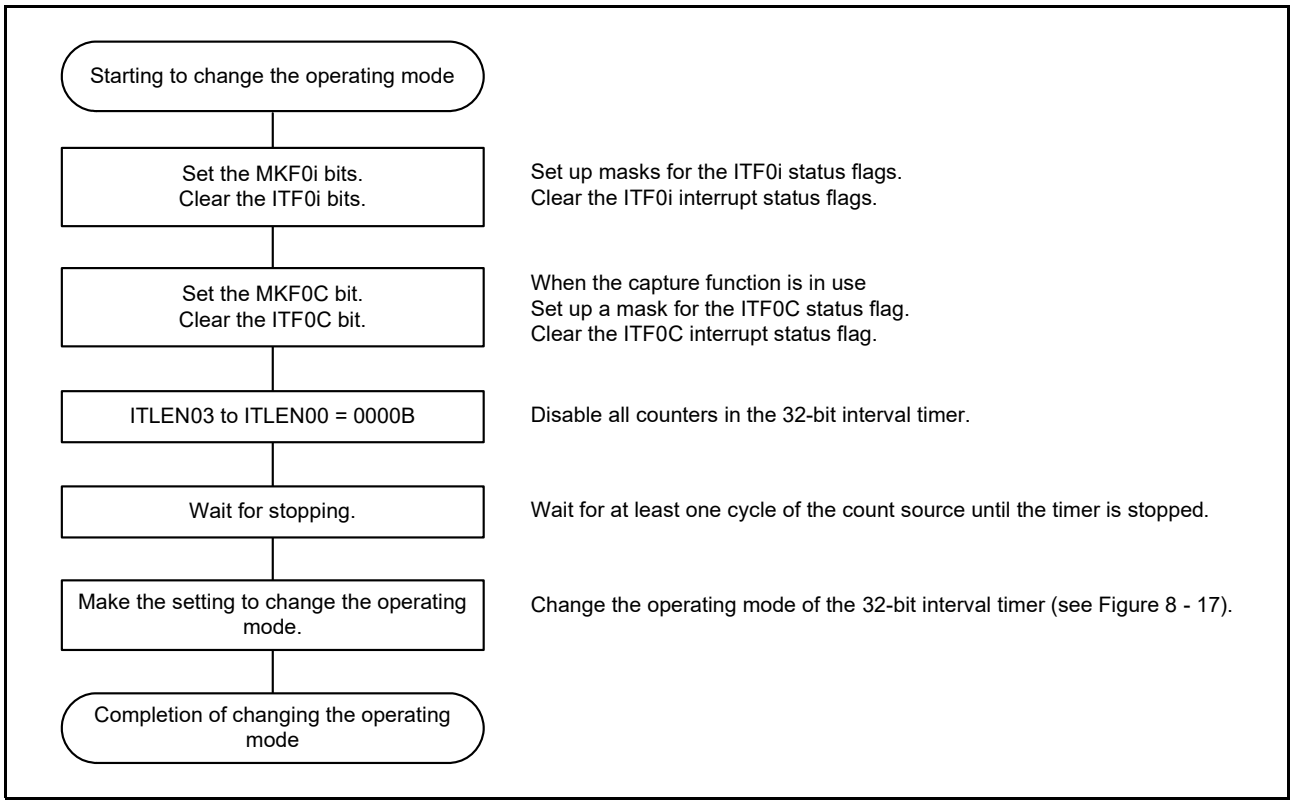
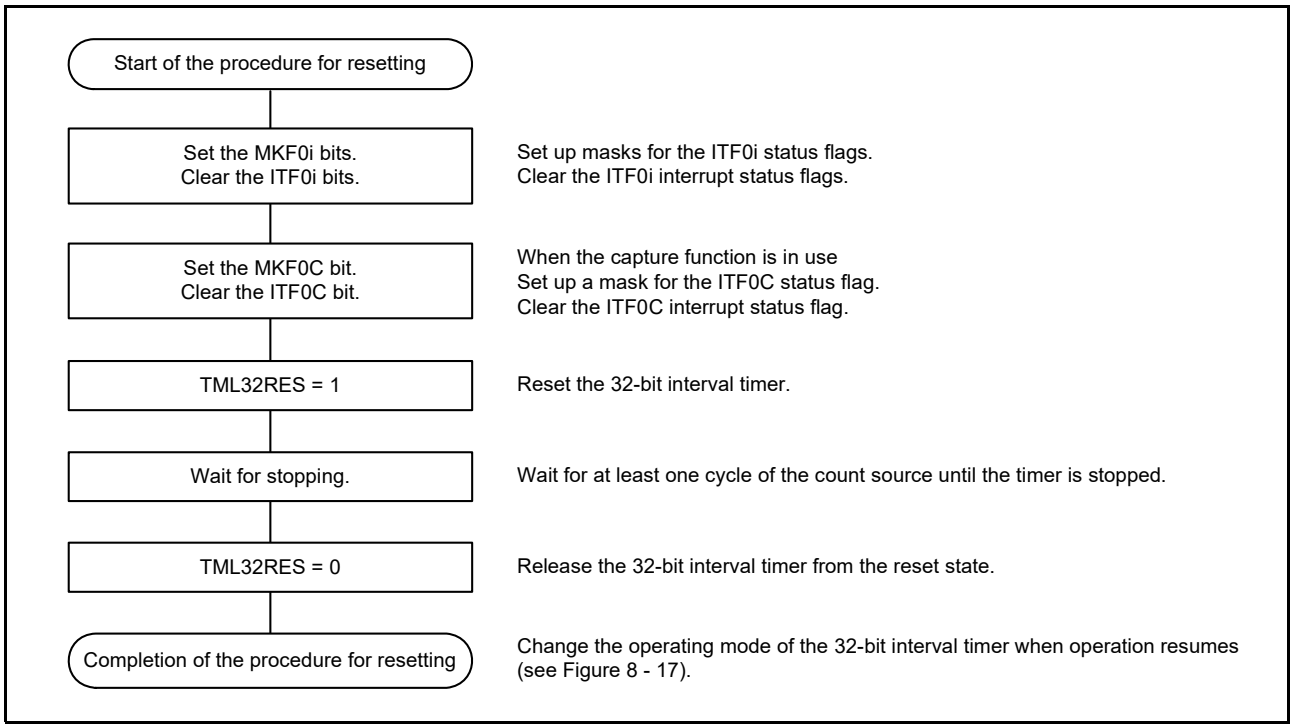


Figure 8 - 20 Procedure for Resetting the 32-bit Interval Timer



Section 9 Watchdog Timer (WDT)

9.1 Functions of Watchdog Timer

Counting by the watchdog timer is set by the user option byte (000C0H).
The watchdog timer operates with the low-speed on-chip oscillator clock (f_{IL}) divided by 2 (1/2 f_{IL}).
The watchdog timer is used to detect program malfunctions. If a malfunction is detected, an internal reset signal is generated. Any among the following cases is considered a program malfunction.

- The watchdog timer counter overflows.
- A 1-bit memory manipulation instruction is used to write to the watchdog timer enable register (WDTE).
- A value other than ACH is written to the WDTE register.
- Writing to the WDTE register proceeds while the window is closed.

When a reset occurs due to the watchdog timer, the WDTRF bit of the reset control flag register (RESF) is set to 1. For details of the RESF register, see **Section 17 Reset Function**.
When 75% of the overflow time + 1/4 f_{IL} is reached, an interval interrupt can be generated.

9.2 Configuration of Watchdog Timer

The watchdog timer includes the following hardware.

Table 9 - 1 Configuration of Watchdog Timer

Item	Configuration
Counter	Internal counter (17 bits)
Control register	Watchdog timer enable register (WDTE)

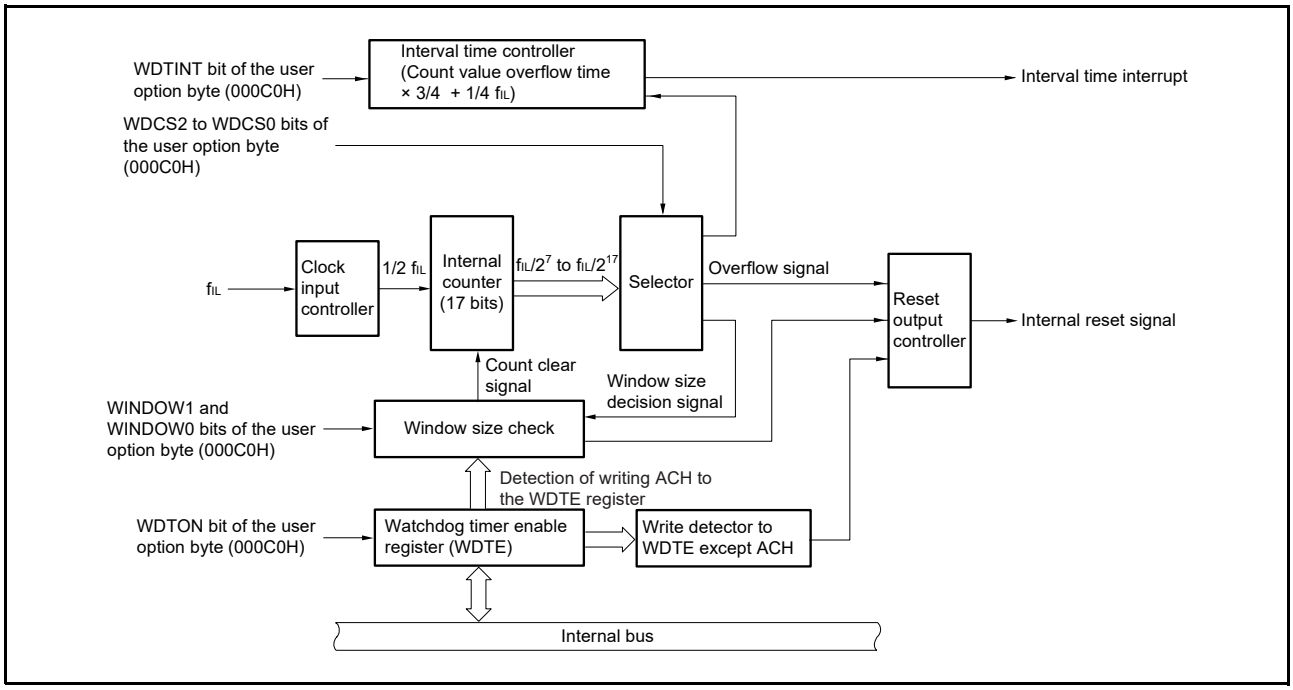
How the counter operation is controlled, overflow time, window open period, and interval interrupt are set by the user option byte.

Table 9 - 2 Setting of the User Option Bytes and Watchdog Timer

Setting of Watchdog Timer	User Option Byte (000C0H)
Watchdog timer interval interrupt	Bit 7 (WDTINT)
Window open period	Bits 6 and 5 (WINDOW1, WINDOW0)
Controlling counter operation of watchdog timer	Bit 4 (WDTON)
Overflow time of watchdog timer	Bits 3 to 1 (WDCS2 to WDCS0)
Controlling counter operation of watchdog timer (in HALT/STOP mode)	Bit 0 (WDSTBYON)

Remark For the user option byte, see **Section 23 Option Bytes**.

Figure 9 - 1 Block Diagram of Watchdog Timer



Remark fIL: Low-speed on-chip oscillator clock

9.3 Register for Controlling the Watchdog Timer

The following register is used to control the watchdog timer.

- Watchdog timer enable register (WDTE)

9.3.1 Watchdog timer enable register (WDTE)

Writing ACH to the WDTE register clears the watchdog timer counter and starts counting again.
This register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 9AH or 1AH[Note].

Note The value of the WDTE register following a reset depends on the setting of the WDTON bit of the user option byte (000C0H). To enable counting by the watchdog timer, set the WDTON bit to 1.

Figure 9 - 2 Format of Watchdog Timer Enable Register (WDTE)

Address: FFFABH
After reset: 9AH/1AH[Note]
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
WDTE								

WDTON Bit Setting Value	Value of the WDTE Register following a Reset
0 (Watchdog timer counting disabled)	1AH
1 (Watchdog timer counting enabled)	9AH

Note The value of the WDTE register following a reset depends on the setting of the WDTON bit of the user option byte (000C0H). To enable counting by the watchdog timer, set the WDTON bit to 1.

Caution 1. If a value other than ACH is written to the WDTE register, an internal reset signal is generated.
Caution 2. If a 1-bit memory manipulation instruction is executed for the WDTE register, an internal reset signal is generated.
Caution 3. The value read from the WDTE register is 9AH or 1AH. Note that the written value (ACH) is not read.

Remark When the setting of the WDTON bit in the user option byte (000C0H) is 1 (watchdog timer operation enabled), the illicit memory access detection is enabled even if the setting of the IAWEN bit in the IAWCTL register is 0 (the detection of illicit memory accesses disabled).

9.4 Operation of Watchdog Timer

9.4.1 Controlling operation of watchdog timer

1. When the watchdog timer is used, its operation is specified by the user option byte (000C0H).
   - Enable counting by the watchdog timer by setting bit 4 (WDTON) of the user option byte (000C0H) to 1 (the counter starts operating after a reset release). For details, see **Section 23 Option Bytes**.

WDTON	Watchdog Timer Counter
0	Counter operation disabled (counting stopped after reset)
1	Counter operation enabled (counting started after reset)

   - Set the time at which the counter is to overflow by using bits 3 to 1 (WDCS2 to WDCS0) of a user option byte (000C0H). For details, see **9.4.2 Setting overflow time of watchdog timer** and **Section 23 Option Bytes**.
   - Set a window open period by using bits 6 and 5 (WINDOW1 and WINDOW0) of the user option byte (000C0H). For details, see **9.4.3 Setting window open period of watchdog timer** and **Section 23 Option Bytes**.
2. After a reset release, the watchdog timer starts counting.
3. By writing ACH to the watchdog timer enable register (WDTE) after the watchdog timer starts counting and before the overflow time set by the user option byte, the watchdog timer is cleared and starts counting again.
4. After that, writing to the WDTE register the second and subsequent times must proceed while the window is open. If the WDTE register is written during a window close period, an internal reset signal is generated.
5. If the time at which an overflow is to occur elapses without ACH having been written to the WDTE register, an internal reset signal is generated. An internal reset signal is also generated in the following cases.
   - If a 1-bit memory manipulation instruction is executed on the WDTE register
   - If data other than ACH is written to the WDTE register

Caution 1. When data is written to the watchdog timer enable register (WDTE) for the first time after reset release, the watchdog timer is cleared in any timing regardless of the window open time, as long as the register is written before the overflow time, and the watchdog timer starts counting again.

Caution 2. After ACH is written to the WDTE register, an error of up to 4 cycles of the clock at f_{IL} may occur before the watchdog timer is cleared.

Caution 3. The watchdog timer can be cleared immediately before the counter value overflows.

Caution 4. The operation of the watchdog timer in the HALT, STOP, and SNOOZE modes differs as follows depending on the set value of bit 0 (WDSTBYON) of the user option byte (000C0H).

	WDSTBYON = 0	WDSTBYON = 1
In HALT mode	Watchdog timer operation stops.	Watchdog timer operation continues.
In STOP mode		
In SNOOZE mode		

If WDSTBYON = 0, the watchdog timer resumes counting after the HALT or STOP mode is released. At this time, the counter is cleared to 0 and counting starts. When operating with the X1 oscillation clock after releasing the STOP mode, the CPU starts operating after the oscillation stabilization time has elapsed. Therefore, if the period between the STOP mode release and the watchdog timer overflow is short, an overflow occurs during the oscillation stabilization time, causing a reset. Consequently, set the overflow time in consideration of the oscillation stabilization time when operating with the X1 oscillation clock and when the watchdog timer is to be cleared after the STOP mode release by an interval interrupt.

9.4.2 Setting overflow time of watchdog timer

Set the overflow time of the watchdog timer by using bits 3 to 1 (WDCS2 to WDCS0) of the user option byte (000C0H). If an overflow occurs, an internal reset signal is generated. The present count is cleared and the watchdog timer starts counting again by writing ACH to the watchdog timer enable register (WDTE) during the window open period before the overflow time.

The following overflow times can be set.

Table 9 - 3 Setting of Overflow Time of Watchdog Timer

WDCS2	WDCS1	WDCS0	Overflow Time of Watchdog Timer (f_{IL} = 37.683 kHz (max.))
0	0	0	$2^7/f_{IL}$ (3.39 ms)
0	0	1	$2^8/f_{IL}$ (6.79 ms)
0	1	0	$2^9/f_{IL}$ (13.58 ms)
0	1	1	$2^{10}/f_{IL}$ (27.17 ms)
1	0	0	$2^{12}/f_{IL}$ (108.69 ms)
1	0	1	$2^{14}/f_{IL}$ (434.78 ms)**Note**
1	1	0	$2^{15}/f_{IL}$ (869.56 ms)**Note**
1	1	1	$2^{17}/f_{IL}$ (3478.26 ms)**Note**

Note Using the watchdog timer under the following conditions may lead to the generation of an interval interrupt (INTWDTI) after one cycle of the watchdog timer clock once the watchdog timer counter has been cleared:

- the watchdog timer interval interrupt is in use, and
- ACH is written to the WDTE register (FFFABH) when the watchdog timer counter has reached or exceeded 75% of the overflow time.

This interrupt can be masked by clearing the watchdog timer counter through steps 1 to 5 below.

1. Set the WDTIMK bit of the interrupt mask flag register 0 (MK0L) to 1 before clearing the watchdog timer counter.
2. Clear the watchdog timer counter.
3. Wait for at least 80 μs.
4. Clear the WDTIIF bit of the interrupt request flag register 0 (IF0L) to 0.
5. Clear the WDTIMK bit of the interrupt mask flag register 0 (MK0L) to 0.

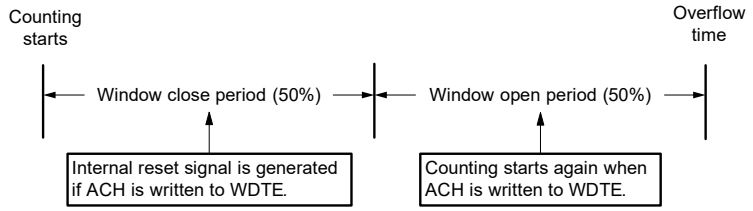
Remark f_{IL}: Low-speed on-chip oscillator clock frequency

9.4.3 Setting window open period of watchdog timer

Set the window open period of the watchdog timer by using bits 6 and 5 (WINDOW1 and WINDOW0) of the user option byte (000C0H). The outline of the window is as follows.

- If ACH is written to the watchdog timer enable register (WDTE) during the window open period, the watchdog timer is cleared and starts counting again.
- Even if ACH is written to the WDTE register during the window close period, an abnormality is detected and an internal reset signal is generated.

Example: The window open period is 50%.



Caution When data is written to the WDTE register for the first time after reset release, the watchdog timer is cleared in any timing regardless of the window open time, as long as the register is written before the overflow time, and the watchdog timer starts counting again.

The window open period can be set is as follows.

Table 9 - 4 Setting Window Open Period of Watchdog Timer

WINDOW1	WINDOW0	Window Open Period of Watchdog Timer
0	1	50%
1	1	100%
Others		Setting prohibited

Remark If the overflow time is set to $2^{10}/f_{IL}$, the times over which the window is open and closed are as follows.

	Setting of Window Open Period	
	50%	100%
Window close time	0 to 18.38 ms	None
Window open time	18.38 to 27.17 ms	0 to 27.17 ms

<Window open period: 50%>

- Overflow time:
  $2^{10}/f_{IL}$ (max.) = 2^{10}/37.683 kHz = 27.17 ms
- Window close time:
  0 to $2^{10}/f_{IL}$ (min.) × (1 - 0.5) = 0 to 2^{10}/27.852 kHz × 0.5 = 0 to 18.38 ms
- Window open time:
  $2^{10}/f_{IL}$ (min.) × (1 - 0.5) to $2^{10}/f_{IL}$ (max.) = 2^{10}/27.853 kHz × 0.5 to 2^{10}/37.683 kHz = 18.38 to 27.17 ms

9.4.4 Setting watchdog timer interval interrupt

Depending on the setting of bit 7 (WDTINT) of a user option byte (000C0H), an interval interrupt (INTWDTI) can be generated when 75% of the overflow time + 1/4 f_{IL} is reached.

Table 9 - 5 Setting of Watchdog Timer Interval Interrupt

WDTINT	Use of Watchdog Timer Interval Interrupt
0	Interval interrupt is not used.
1	Interval interrupt is generated when 75% of the overflow time + 1/4 f_{IL} is reached.

Caution **When operating with the X1 oscillation clock after releasing the STOP mode, the CPU starts operating after the oscillation stabilization time has elapsed. Therefore, if the period between the STOP mode release and the watchdog timer overflow is short, an overflow occurs during the oscillation stabilization time, causing a reset. Consequently, set the overflow time in consideration of the oscillation stabilization time when operating with the X1 oscillation clock and when the watchdog timer is to be cleared after the STOP mode release by an interval interrupt.**

Remark The watchdog timer continues counting even after INTWDTI is generated until ACH is written to the watchdog timer enable register (WDTE). If ACH is not written to the WDTE register before the overflow time, an internal reset signal is generated.

Section 10 A/D Converter (ADC)

The number of analog input channels of the A/D converter differs, depending on the product.

	20-pin	32-pin	44- and 48-pin	64-pin
Number of the analog input channels	6 (ANI0 to ANI2, ANI16 to ANI18)	8 (ANI0 to ANI3, ANI16 to ANI19)	10 (ANI0 to ANI7, ANI18, ANI19)	12 (ANI0 to ANI7, ANI16 to ANI19)

10.1 Function of A/D Converter

The A/D converter is used to convert analog input signals into digital values, and is configured to control analog inputs, including up to 12 channels of A/D converter analog inputs (ANI0 to ANI7 and ANI16 to ANI19). 10-bit or 8-bit resolution can be selected by the ADTYP bit of A/D converter mode register 2 (ADM2). The A/D converter has the following function.

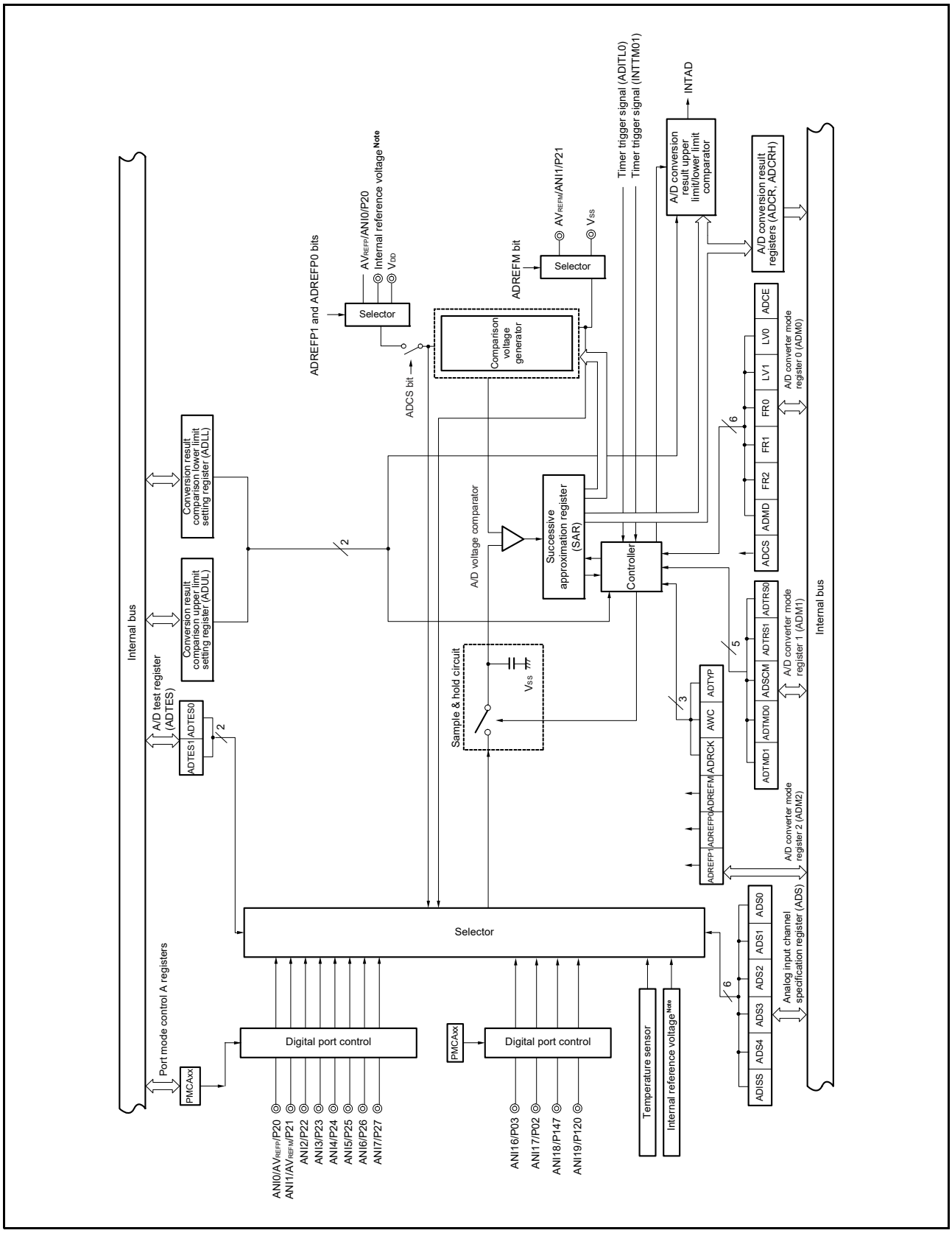
- 10-bit/8-bit resolution A/D conversion
  10-bit or 8-bit resolution A/D conversion proceeds repeatedly for one analog input channel selected from ANI0 to ANI7 and ANI16 to ANI19. Each time an A/D conversion operation is complete, an interrupt request signal (INTAD) is generated in the select mode.

Various A/D conversion modes can be specified by using the mode combinations below.

Trigger mode	Software trigger mode	Conversion is started by software.
	Hardware trigger no-wait mode	Conversion is started by detecting a hardware trigger.
	Hardware trigger wait mode	The power to the A/D converter is turned on by detecting a hardware trigger while the power is off and the A/D converter is in the conversion standby state, and conversion is then started automatically after the A/D power supply stabilization wait time has passed. When using the SNOOZE mode function, specify the hardware trigger wait mode.
Channel selection mode	Select mode	A/D conversion is performed on the analog input of one selected channel.
	Scan mode	A/D conversion is performed on the analog input of four channels in order. Four consecutive channels can be selected from ANI0 to ANI7 as analog input channels.
Conversion operation mode	One-shot conversion mode	A/D conversion is performed on the selected channel once.
	Sequential conversion mode	A/D conversion is sequentially performed on the selected channels until it is stopped by software.
Operation voltage mode	Normal 1 or normal 2 mode	Conversion operations are possible within the operating voltage range 2.7 V ≤ V_{DD} ≤ 5.5 V.
	Low voltage 1 or low voltage 2 mode	Conversion operations are possible within the operating voltage range 1.6 V ≤ V_{DD} ≤ 5.5 V. Select this mode for conversion at a low voltage. Because the operation voltage is low, it is internally boosted during conversion.
Sampling time selection	Sampling clock cycles: 7 f_{AD}	The sampling time in normal 1 or low voltage 1 mode is seven cycles of the conversion clock (f_{AD}). Select this mode when the output impedance of the analog input source is high and the sampling time should be long.
	Sampling clock cycles: 5 f_{AD}	The sampling time in normal 2 or low voltage 2 mode is five cycles of the conversion clock (f_{AD}). Select this mode when enough sampling time is ensured (for example, when the output impedance of the analog input source is low).

Caution The operation mode that can be selected differs depending on the analog input channel, V_{DD} voltage, AV_{REFP} voltage, trigger mode, and f_{CLK}. See Table 10 - 3 A/D Conversion Time Selection for details.

Figure 10 - 1 Block Diagram of A/D Converter



Note For the internal reference voltage, see **Section 28 Electrical Characteristics**.

Remark Analog input pins in this figure are for a 64-pin product.

10.2 Configuration of A/D Converter

The A/D converter includes the following hardware.

1. ANI0 to ANI7 and ANI16 to ANI19 pins
   These are the analog input pins of the 12 channels of the A/D converter. They input analog signals to be converted into digital signals. Pins other than the one selected as the analog input pin can be used as I/O port pins.
2. Sample & hold (S&H) circuit
   The sample & hold circuit samples each of the analog input voltages sequentially sent from the input circuit, and sends them to the A/D voltage comparator. This circuit also holds the sampled analog input voltage during A/D conversion.
3. A/D voltage comparator
   This A/D voltage comparator compares the voltage generated from the voltage tap of the comparison voltage generator with the analog input voltage. If the analog input voltage is found to be greater than the reference voltage (1/2 AV_{REF}) as a result of the comparison, the most significant bit (MSB) of the successive approximation register (SAR) is set. If the analog input voltage is less than the reference voltage (1/2 AV_{REF}), the MSB bit of the SAR is reset. After that, bit 8 of the SAR register is automatically set, and the next comparison is made. The voltage tap of the comparison voltage generator is selected by the value of bit 9, to which the result has been already set.

   - Bit 9 = 0: (1/4 AV_{REF})
   - Bit 9 = 1: (3/4 AV_{REF})

   The voltage tap of the comparison voltage generator and the analog input voltage are compared and bit 8 of the SAR register is manipulated according to the result of the comparison.

   - Analog input voltage ≥ Voltage tap of comparison voltage generator: Bit 8 = 1
   - Analog input voltage ≤ Voltage tap of comparison voltage generator: Bit 8 = 0

   Comparison is continued like this to bit 0 of the SAR register.
   When performing A/D conversion at a resolution of 8 bits, the comparison continues until bit 2 of the SAR register.
4. Comparison voltage generator
   The comparison voltage generator generates the voltage to be compared with the input from an analog input pin.
5. Successive approximation register (SAR)
   The SAR register is used to set voltage tap data whose values from the comparison voltage generator match the voltage values of the analog input pins, 1 bit at a time starting from the most significant bit (MSB). If data is set in the SAR register all the way to the least significant bit (LSB) (end of A/D conversion), the contents of the SAR register (conversion results) are held in the A/D conversion result register (ADCR). When all the specified A/D conversion operations have ended, an A/D conversion end interrupt request signal (INTAD) is generated.
6. 10-bit A/D conversion result register (ADCR)
   The A/D conversion result is loaded from the successive approximation register to this register each time A/D conversion is completed, and the ADCR register holds the A/D conversion result in its higher 10 bits (the lower 6 bits are fixed to 0).
7. 8-bit A/D conversion result register (ADCRH)
   The A/D conversion result is loaded from the successive approximation register to this register each time A/D conversion is completed, and the ADCRH register holds the higher 8 bits of the A/D conversion result.

8. Controller
   This circuit controls the conversion time of an input analog signal that is to be converted into a digital signal, as well as starting and stopping of the conversion operation. When A/D conversion has been completed, this controller generates INTAD through the A/D conversion result upper limit/lower limit comparator.
9. AVREFP pin
   This pin is used to input an external reference voltage (AVREFP). To use AVREFP as the positive reference voltage of the A/D converter, set the ADREFP1 and ADREFP0 bits of A/D converter mode register 2 (ADM2) to 0 and 1, respectively. The analog signals input to ANI2 to ANI7 and ANI16 to ANI19 are converted to digital signals based on the voltage applied between AVREFP and the negative reference voltage (AVREFM/VSS). Selecting VDD or the internal reference voltage as the positive reference voltage of the A/D converter is also possible.
10. AVREFM pin
    This pin is used to input an external reference voltage (AVREFM). To use AVREFM as the negative reference voltage of the A/D converter, set the ADREFM bit of the ADM2 register to 1. Selecting VSS as the negative reference voltage of the A/D converter is also possible.

Remark AVREF: The positive reference voltage of the A/D converter. This can be selected from AVREFP, the internal reference voltage, and VDD. For the internal reference voltage, see **Section 28 Electrical Characteristics**.

10.3 Registers for Controlling the A/D Converter

The following registers are used to control the A/D converter.

- Peripheral enable register 0 (PER0)
- Peripheral reset control register 0 (PRR0)
- A/D converter mode register 0 (ADM0)
- A/D converter mode register 1 (ADM1)
- A/D converter mode register 2 (ADM2)
- 10-bit A/D conversion result register (ADCR)
- 8-bit A/D conversion result register (ADCRH)
- Analog input channel specification register (ADS)
- Conversion result comparison upper limit setting register (ADUL)
- Conversion result comparison lower limit setting register (ADLL)
- A/D test register (ADTES)
- Port mode registers (PMxx)
- Port mode control A registers (PMCAxx)

10.3.1 Peripheral enable register 0 (PER0)

The PER0 register is used to enable or disable the supply of a clock signal to various on-chip peripheral modules. Clock supply to an on-chip peripheral module that is not to be used can be stopped to decrease power consumption and noise. If the A/D converter is to be used, be sure to set the ADCEN bit of this register to 1. The PER0 register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 10 - 2 Format of Peripheral Enable Register 0 (PER0)

Address: F00F0H
After reset: 00H
R/W: R/W

Symbol	7	6	<5>	<4>	<3>	<2>	1	<0>
PER0	0	0	ADCEN	IICA0EN Note 1	SAU1EN	SAU0EN	0	TAU0EN

ADCEN	Control of supply of an input clock to the A/D converter
0	Stops supply of an input clock. • The SFRs used by the A/D converter cannot be written.Note 2
1	Enables supply of an input clock. • The SFRs used by the A/D converter can be read and written.

Note 1. This bit is only present in the 32- to 64-pin products.
Note 2. When an SFR is read, the value returned is 00H or 0000H regardless of its setting.

Caution 1. Be sure to leave bits 7, 6, and 1 set to 0.
Caution 2. Be sure to leave bit 4 in the 20-pin products set to 0.
Caution 3. When setting the A/D converter, start by setting the following registers while the setting of the ADCEN bit is 1.
- **A/D converter mode register 0 (ADM0)**
- **A/D converter mode register 1 (ADM1)**
- **A/D converter mode register 2 (ADM2)**
- **10-bit A/D conversion result register (ADCR)**
- **8-bit A/D conversion result register (ADCRH)**
- **Analog input channel specification register (ADS)**
- **Conversion result comparison upper limit setting register (ADUL)**
- **Conversion result comparison lower limit setting register (ADLL)**
- **A/D test register (ADTES)**

If the setting of the ADCEN bit is 0, each register which controls the A/D converter is restored to 00H and writing to any of those registers is ignored. Note, however, writing to the following registers is valid.
- **Port mode registers (PM0, PM2, PM12, and PM14)**
- **Port mode control A registers (PMCA0, PMCA2, PMCA12, and PMCA14)**

10.3.2 Peripheral reset control register 0 (PRR0)

The PRR0 register is used to control resetting of the on-chip peripheral modules. Each bit in this register controls resetting and release from the reset state of the corresponding on-chip peripheral module. To place the A/D converter in the reset state, be sure to set the ADCRES bit of this register to 1. The PRR0 register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 10 - 3 Format of Peripheral Reset Control Register 0 (PRR0)

Address: F00F1H
After reset: 00H
R/W: R/W

Symbol	7	6	<5>	<4>	<3>	<2>	1	<0>
PRR0	0	0	ADCRES	IICA0RES **Note**	SAU1RES	SAU0RES	0	TAU0RES

ADCRES	Control resetting of the A/D converter
0	The A/D converter is released from the reset state.
1	The A/D converter is in the reset state. • The SFRs for use with the A/D converter are initialized.

Note This bit is only present in the 32- to 64-pin products.

Caution 1. Be sure to leave bits 7, 6, and 1 set to 0.
Caution 2. Be sure to leave bit 4 in the 20-pin products set to 0.

10.3.3 A/D converter mode register 0 (ADM0)

The ADM0 register sets the time for converting analog input to digital data, and starts and stops conversion. The ADM0 register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 10 - 4 Format of A/D Converter Mode Register 0 (ADM0)

Address: FFF30H
After reset: 00H
R/W: R/W

Symbol	<7>	6	5	4	3	2	1	<0>
ADM0	ADCS	ADMD	FR2Note 1	FR1Note 1	FR0Note 1	LV1Note 1	LV0Note 1	ADCE

ADCS	A/D conversion operation control
0	Stops conversion operation. [When read] Conversion is stopped or in standby.
1	Enables conversion operation. [When read] While in the software trigger mode: Conversion in progress While in the hardware trigger wait mode: A/D power supply stabilization wait state or conversion in progress

ADMD	Specification of the A/D conversion channel selection mode
0	Select mode
1	Scan mode

ADCE	A/D voltage comparator operation controlNote 2
0	Stops A/D voltage comparator operation.
1	Enables A/D voltage comparator operation.

Note 1. For details of the FR2 to FR0, LV1, LV0 bits, and A/D conversion, see **Table 10 - 3 A/D Conversion Time Selection**.

<R> **Note 2.** While in the software trigger mode or hardware trigger no-wait mode, the operation of the A/D voltage comparator is controlled by the ADCS and ADCE bits, and it takes 1 μs from the start of operation for the operation to become stable. Therefore, after having set the ADCE bit to 1, wait for at least 1 μs before setting the ADCS bit to 1; this ensures that the results of conversion will be valid from the very first conversion. If the ADCS bit is set to 1 without the wait, ignore the data from the first conversion.

Caution 1. Change the ADMD, FR2 to FR0, LV1, and LV0 bits while conversion is stopped (ADCS = 0, ADCE = 0).

<R> **Caution 2. Do not set the ADCS bit to 1 and the ADCE bit to 0 at the same time.**

Caution 3. Do not change the ADCS and ADCE bits from 0 to 1 at the same time by using an 8-bit memory manipulation instruction. Be sure to set these bits in the order described in 10.7 Flows of Setting the A/D Converter Modes.

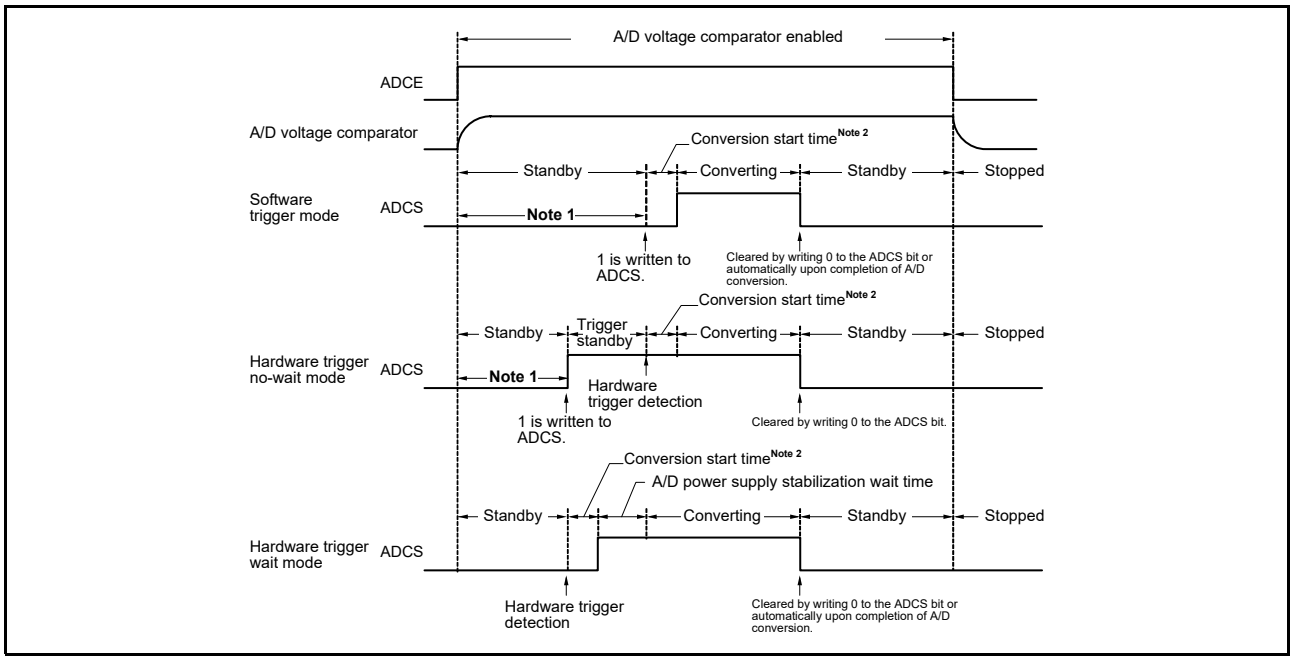
Table 10 - 1 Settings of ADCS and ADCE Bits

ADCS	ADCE	A/D Conversion State
0	0	Stopped
0	1	Standby
1	0	Setting prohibited
1	1	Conversion in progress

Table 10 - 2 Setting and Clearing Conditions for ADCS Bit

A/D Conversion Mode			Set Conditions	Clear Conditions
Software trigger mode	Select mode	Sequential conversion mode	1 is written to ADCS.	• 0 is written to ADCS.
		One-shot conversion mode		• 0 is written to ADCS. • Completion of A/D conversion
	Scan mode	Sequential conversion mode		• 0 is written to ADCS.
		One-shot conversion mode		• 0 is written to ADCS. • Completion of conversion on the specified four channels
Hardware trigger no-wait mode	Select mode	Sequential conversion mode		• 0 is written to ADCS.
		One-shot conversion mode		• 0 is written to ADCS.
	Scan mode	Sequential conversion mode		• 0 is written to ADCS.
		One-shot conversion mode		• 0 is written to ADCS.
Hardware trigger wait mode	Select mode	Sequential conversion mode	A hardware trigger is input.	• 0 is written to ADCS.
		One-shot conversion mode		• 0 is written to ADCS. • Completion of A/D conversion
	Scan mode	Sequential conversion mode		• 0 is written to ADCS.
		One-shot conversion mode		• 0 is written to ADCS. • Completion of conversion on the specified four channels

Figure 10 - 5 Timing of Operations When the A/D Voltage Comparator Is Used



Note 1. While in the software trigger mode or hardware trigger no-wait mode, the interval between setting the ADCE bit to 1 and setting the ADCS bit to 1 must be at least 1 μs to allow stabilization of the internal circuit.

Note 2. The following shows the maximum time to start conversion.

ADM0			Conversion Clock (f_{AD})	Conversion Start Time (Number of f_{CLK} Clock Cycles)	
FR2	FR1	FR0		Software Trigger Mode/ Hardware Trigger No-wait Mode	Hardware Trigger Wait Mode
0	0	0	f_{CLK}/64	63	1
0	0	1	f_{CLK}/32	31	1
0	1	0	f_{CLK}/16	15	1
0	1	1	f_{CLK}/8	7	1
1	0	0	f_{CLK}/6	5	1
1	0	1	f_{CLK}/5	4	1
1	1	0	f_{CLK}/4	3	1
1	1	1	f_{CLK}/2	1	1

However, for the second and subsequent conversion in sequential conversion mode and for conversion of the channel specified by scan 1, 2, and 3 in scan mode, the conversion start time and stabilization wait time for A/D power supply do not occur after a hardware trigger is detected.

Caution 1. If using the hardware trigger wait mode, setting the ADCS bit to 1 is prohibited (but the bit is automatically switched to 1 when the hardware trigger signal is detected). However, it is possible to clear the ADCS bit to 0 to specify the A/D conversion standby state.

Caution 2. While in the one-shot conversion mode of the hardware trigger no-wait mode, the ADCS bit is not automatically cleared to 0 when A/D conversion is complete. Instead, 1 is retained.

Caution 3. Only rewrite the value of the ADCE bit when ADCS = 0 (while in the conversion stopped/conversion standby state).

<R> **Caution 4. To complete A/D conversion, specify at least the following time as the hardware trigger interval:**
Hardware trigger no-wait mode: 2 f_{CLK} clock cycles + conversion start time + A/D conversion time
Hardware trigger wait mode: 2 f_{CLK} clock cycles + conversion start time + A/D power supply stabilization wait time + A/D conversion time

Remark f_{CLK}: CPU/peripheral hardware clock frequency

Table 10 - 3 A/D Conversion Time Selection (1/4)

1. When there is no A/D power supply stabilization wait time
   Normal modes 1 and 2 in software trigger mode and hardware trigger no-wait mode

A/D Converter Mode Register 0 (ADM0)					Mode	Conversion Clock (f_{AD})	Number of Clock Cycles for Conversion **Note**	Conversion Time	A/D Conversion Time at 10-bit Resolution				
									2.7 V ≤ V_{DD} ≤ 5.5 V				
FR2	FR1	FR0	LV1	LV0					f_{CLK} = 1 MHz	f_{CLK} = 4 MHz	f_{CLK} = 8 MHz	f_{CLK} = 16 MHz	f_{CLK} = 32 MHz
0	0	0	0	0	Normal 1	f_{CLK}/64	19 f_{AD} (7 f_{AD} for sampling)	1216/f_{CLK}	Setting prohibited	Setting prohibited	Setting prohibited	76 μs	38 μs
0	0	1				f_{CLK}/32		608/f_{CLK}	Setting prohibited	Setting prohibited	76 μs	38 μs	19 μs
0	1	0				f_{CLK}/16		304/f_{CLK}	Setting prohibited	76 μs	38 μs	19 μs	9.5 μs
0	1	1				f_{CLK}/8		152/f_{CLK}	Setting prohibited	38 μs	19 μs	9.5 μs	4.75 μs
1	0	0				f_{CLK}/6		114/f_{CLK}	Setting prohibited	28.5 μs	14.25 μs	7.125 μs	3.5625 μs
1	0	1				f_{CLK}/5		95/f_{CLK}	95 μs	23.75 μs	11.875 μs	5.938 μs	2.9688 μs
1	1	0				f_{CLK}/4		76/f_{CLK}	76 μs	19 μs	9.5 μs	4.75 μs	2.375 μs
1	1	1				f_{CLK}/2		38/f_{CLK}	38 μs	9.5 μs	4.75 μs	2.375 μs	Setting prohibited
0	0	0	0	1	Normal 2	f_{CLK}/64	17 f_{AD} (5 f_{AD} for sampling)	1088/f_{CLK}	Setting prohibited	Setting prohibited	Setting prohibited	68 μs	34 μs
0	0	1				f_{CLK}/32		544/f_{CLK}	Setting prohibited	Setting prohibited	68 μs	34 μs	17 μs
0	1	0				f_{CLK}/16		272/f_{CLK}	Setting prohibited	68 μs	34 μs	17 μs	8.5 μs
0	1	1				f_{CLK}/8		136/f_{CLK}	Setting prohibited	34 μs	17 μs	8.5 μs	4.25 μs
1	0	0				f_{CLK}/6		102/f_{CLK}	Setting prohibited	25.5 μs	12.75 μs	6.375 μs	3.1875 μs
1	0	1				f_{CLK}/5		85/f_{CLK}	85 μs	21.25 μs	10.625 μs	5.3125 μs	2.6563 μs
1	1	0				f_{CLK}/4		68/f_{CLK}	68 μs	17 μs	8.5 μs	4.25 μs	2.125 μs
1	1	1				f_{CLK}/2		34/f_{CLK}	34 μs	8.5 μs	4.25 μs	2.125 μs	Setting prohibited

Note These are the numbers of clock cycles when conversion is with 10-bit resolution. When eight-bit resolution is selected, the values are shorter by two cycles of the conversion clock (f_{AD}).

Caution 1. The A/D conversion time must also be set within the relevant range of conversion times (t_{CONV}) described in 28.6.1 Characteristics of the A/D converter for T_A = –40 to 85°C.

Caution 2. Rewrite the FR2 to FR0, LV1 and LV0 bits to other than the same data while conversion is stopped (ADCS = 0, ADCE = 0).

Caution 3. The conversion times listed above do not include the conversion start time. Add the conversion start time to obtain the time for the first conversion. The conversion times also do not take clock frequency errors into account. Consider clock frequency errors when selecting conversion times.

Remark f_{CLK}: CPU/peripheral hardware clock frequency

Table 10 - 3 A/D Conversion Time Selection (2/4)

2. When there is no A/D power supply stabilization wait time

Low-voltage modes 1 and 2 in software trigger mode and hardware trigger no-wait mode

A/D Converter Mode Register 0 (ADM0)					Mode	Conversion Clock (f_{AD})	Number of Clock Cycles for Conversion Note	Conversion Time	A/D Conversion Time at 10-bit Resolution				
									1.6 V ≤ V_{DD} ≤ 5.5 V		1.8 V ≤ V_{DD} ≤ 5.5 V	2.4 V ≤ V_{DD} ≤ 5.5 V	2.7 V ≤ V_{DD} ≤ 5.5 V
FR2	FR1	FR0	LV1	LV0					f_{CLK} = 1 MHz	f_{CLK} = 4 MHz	f_{CLK} = 8 MHz	f_{CLK} = 16 MHz	f_{CLK} = 32 MHz
0	0	0	1	0	Low voltage 1	f_{CLK}/64	19 f_{AD} (7 f_{AD} for sampling)	1216/f_{CLK}	Setting prohibited	Setting prohibited	Setting prohibited	76 µs	38 µs
0	0	1				f_{CLK}/32		608/f_{CLK}	Setting prohibited	Setting prohibited	76 µs	38 µs	19 µs
0	1	0				f_{CLK}/16		304/f_{CLK}	Setting prohibited	76 µs	38 µs	19 µs	9.5 µs
0	1	1				f_{CLK}/8		152/f_{CLK}	Setting prohibited	38 µs	19 µs	9.5 µs	4.75 µs
1	0	0				f_{CLK}/6		114/f_{CLK}	Setting prohibited	28.5 µs	14.25 µs	7.125 µs	3.5625 µs
1	0	1				f_{CLK}/5		95/f_{CLK}	95 µs	23.75 µs	11.875 µs	5.938 µs	2.9688 µs
1	1	0				f_{CLK}/4		76/f_{CLK}	76 µs	19 µs	9.5 µs	4.75 µs	2.375 µs
1	1	1				f_{CLK}/2		38/f_{CLK}	38 µs	9.5 µs	4.75 µs	2.375 µs	Setting prohibited
0	0	0	1	1	Low voltage 2	f_{CLK}/64	17 f_{AD} (5 f_{AD} for sampling)	1088/f_{CLK}	Setting prohibited	Setting prohibited	Setting prohibited	68 µs	34 µs
0	0	1				f_{CLK}/32		544/f_{CLK}	Setting prohibited	Setting prohibited	68 µs	34 µs	17 µs
0	1	0				f_{CLK}/16		272/f_{CLK}	Setting prohibited	68 µs	34 µs	17 µs	8.5 µs
0	1	1				f_{CLK}/8		136/f_{CLK}	Setting prohibited	34 µs	17 µs	8.5 µs	4.25 µs
1	0	0				f_{CLK}/6		102/f_{CLK}	Setting prohibited	25.5 µs	12.75 µs	6.375 µs	3.1875 µs
1	0	1				f_{CLK}/5		85/f_{CLK}	85 µs	21.25 µs	10.625 µs	5.3125 µs	2.6563 µs
1	1	0				f_{CLK}/4		68/f_{CLK}	68 µs	17 µs	8.5 µs	4.25 µs	2.125 µs
1	1	1				f_{CLK}/2		34/f_{CLK}	34 µs	8.5 µs	4.25 µs	2.125 µs	Setting prohibited

Note These are the numbers of clock cycles when conversion is with 10-bit resolution. When eight-bit resolution is selected, the values are shorter by two cycles of the conversion clock (f_{AD}).

Caution 1. The A/D conversion time must also be set within the relevant range of conversion times (t_{CONV}) described in 28.6.1 Characteristics of the A/D converter for T_A = –40 to 85°C.

Caution 2. Rewrite the FR2 to FR0, LV1 and LV0 bits to other than the same data while conversion is stopped (ADCS = 0, ADCE = 0).

Caution 3. The conversion times listed above do not include the conversion start time. Add the conversion start time to obtain the time for the first conversion. The conversion times also do not take clock frequency errors into account. Consider clock frequency errors when selecting conversion times.

Remark f_{CLK}: CPU/peripheral hardware clock frequency

Table 10 - 3 A/D Conversion Time Selection (3/4)

<R> 3. When there is an A/D power supply stabilization wait time

Normal modes 1 and 2 in hardware trigger wait mode**Note 1**

A/D Converter Mode Register 0 (ADM0)					Mode	Conversion Clock (f_{AD})	Number of Clock Cycles for A/D Power Supply Stabilization Wait	Number of Clock Cycles for Conversion **Note 2**	A/D Power Supply Stabilization Wait Time + Conversion Time	A/D Power Supply Stabilization Wait Time + A/D Conversion Time at 10-bit Resolution				
										2.7 V ≤ V_{DD} ≤ 5.5 V				
FR2	FR1	FR0	LV1	LV0						f_{CLK} = 1 MHz	f_{CLK} = 4 MHz	f_{CLK} = 8 MHz	f_{CLK} = 16 MHz	f_{CLK} = 32 MHz
0	0	0	0	0	Normal 1	f_{CLK}/64	8 f_{AD}	19 f_{AD} (7 f_{AD} for sampling)	1728/f_{CLK}	Setting prohibited	Setting prohibited	Setting prohibited	108 μs	54 μs
0	0	1				f_{CLK}/32			864/f_{CLK}	Setting prohibited	Setting prohibited	108 μs	54 μs	27 μs
0	1	0				f_{CLK}/16			432/f_{CLK}	Setting prohibited	108 μs	54 μs	27 μs	13.5 μs
0	1	1				f_{CLK}/8			216/f_{CLK}	Setting prohibited	54 μs	27 μs	13.5 μs	6.75 μs
1	0	0				f_{CLK}/6			162/f_{CLK}	Setting prohibited	40.5 μs	20.25 μs	10.125 μs	5.0625 μs
1	0	1				f_{CLK}/5			135/f_{CLK}	135 μs	33.75 μs	16.875 μs	8.4375 μs	4.21875 μs
1	1	0				f_{CLK}/4			108/f_{CLK}	108 μs	27 μs	13.5 μs	6.75 μs	3.375 μs
1	1	1				f_{CLK}/2			54/f_{CLK}	54 μs	13.5 μs	6.75 μs	3.375 μs	Setting prohibited
0	0	0	0	1	Normal 2	f_{CLK}/64	8 f_{AD}	17 f_{AD} (5 f_{AD} for sampling)	1600/f_{CLK}	Setting prohibited	Setting prohibited	Setting prohibited	100 μs	50 μs
0	0	1				f_{CLK}/32			800/f_{CLK}	Setting prohibited	Setting prohibited	100 μs	50 μs	25 μs
0	1	0				f_{CLK}/16			400/f_{CLK}	Setting prohibited	100 μs	50 μs	25 μs	12.5 μs
0	1	1				f_{CLK}/8			200/f_{CLK}	Setting prohibited	50 μs	25 μs	12.5 μs	6.25 μs
1	0	0				f_{CLK}/6			150/f_{CLK}	Setting prohibited	37.5 μs	18.75 μs	9.375 μs	4.6875 μs
1	0	1				f_{CLK}/5			125/f_{CLK}	125 μs	31.25 μs	15.625 μs	7.8125 μs	3.90625 μs
1	1	0				f_{CLK}/4			100/f_{CLK}	100 μs	25 μs	12.5 μs	6.25 μs	3.125 μs
1	1	1				f_{CLK}/2			50/f_{CLK}	50 μs	12.5 μs	6.25 μs	3.125 μs	Setting prohibited

Note 1. For the second and subsequent conversion in sequential conversion mode and for conversion of the channel specified by scan 1, 2, and 3 in scan mode, the conversion start time and stabilization wait time for A/D power supply do not occur after a hardware trigger is detected (see **Table 10 - 3 A/D Conversion Time Selection (1/4)**).

Note 2. These are the numbers of clock cycles when conversion is with 10-bit resolution. When eight-bit resolution is selected, the values are shorter by two cycles of the conversion clock (f_{AD}).

Caution 1. The A/D conversion time must also be set within the relevant range of conversion times (t_{CONV}) described in 28.6.1 Characteristics of the A/D converter for T_A = –40 to 85°C. Note that the conversion time (t_{CONV}) does not include the A/D power supply stabilization wait time.

Caution 2. Rewrite the FR2 to FR0, LV1 and LV0 bits to other than the same data while conversion is stopped (ADCS = 0, ADCE = 0).

Caution 3. The conversion times listed above do not include the conversion start time. Add the conversion start time to obtain the time for the first conversion. The conversion times also do not take clock frequency errors into account. Consider clock frequency errors when selecting conversion times.

Caution 4. The conversion times in hardware trigger wait mode include the A/D power supply stabilization wait time from the time the hardware trigger is detected.

Remark f_{CLK}: CPU/peripheral hardware clock frequency

Table 10 - 3 A/D Conversion Time Selection (4/4)

<R> 4. When there is an A/D power supply stabilization wait time

Low-voltage modes 1 and 2 in hardware trigger wait mode[Note 1]

A/D Converter Mode Register 0 (ADM0)					Mode	Conversion Clock (f_{AD})	Number of Clock Cycles for A/D Power Supply Stabilization Wait	Number of Clock Cycles for Conversion [Note 2]	A/D Power Supply Stabilization Wait Time + Conversion Time	A/D Power Supply Stabilization Wait Time + A/D Conversion Time at 10-bit Resolution				
										1.6 V ≤ V_{DD} ≤ 5.5 V		1.8 V ≤ V_{DD} ≤ 5.5 V	2.4 V ≤ V_{DD} ≤ 5.5 V	2.7 V ≤ V_{DD} ≤ 5.5 V
FR2	FR1	FR0	LV1	LV0						f_{CLK} = 1 MHz	f_{CLK} = 4 MHz	f_{CLK} = 8 MHz	f_{CLK} = 16 MHz	f_{CLK} = 32 MHz
0	0	0	1	0	Low voltage 1	f_{CLK}/64	2 f_{AD}	19 f_{AD} (7 f_{AD} for sampling)	1344/f_{CLK}	Setting prohibited	Setting prohibited	Setting prohibited	84 µs	42 µs
0	0	1				f_{CLK}/32			672/f_{CLK}	Setting prohibited	Setting prohibited	84 µs	42 µs	21 µs
0	1	0				f_{CLK}/16			336/f_{CLK}	Setting prohibited	84 µs	42 µs	21 µs	10.5 µs
0	1	1				f_{CLK}/8			168/f_{CLK}	Setting prohibited	42 µs	21 µs	10.5 µs	5.25 µs
1	0	0				f_{CLK}/6			126/f_{CLK}	Setting prohibited	31.5 µs	15.75 µs	7.875 µs	3.9375 µs
1	0	1				f_{CLK}/5			105/f_{CLK}	105 µs	26.25 µs	13.125 µs	6.5625 µs	3.238125 µs
1	1	0				f_{CLK}/4			84/f_{CLK}	84 µs	21 µs	10.5 µs	5.25 µs	2.625 µs
1	1	1				f_{CLK}/2			42/f_{CLK}	42 µs	10.5 µs	5.25 µs	2.625 µs	Setting prohibited
0	0	0	1	1	Low voltage 2	f_{CLK}/64	2 f_{AD}	17 f_{AD} (5 f_{AD} for sampling)	1216/f_{CLK}	Setting prohibited	Setting prohibited	Setting prohibited	76 µs	38 µs
0	0	1				f_{CLK}/32			608/f_{CLK}	Setting prohibited	Setting prohibited	76 µs	38 µs	19 µs
0	1	0				f_{CLK}/16			304/f_{CLK}	Setting prohibited	76 µs	38 µs	19 µs	9.5 µs
0	1	1				f_{CLK}/8			152/f_{CLK}	Setting prohibited	38 µs	19 µs	9.5 µs	4.75 µs
1	0	0				f_{CLK}/6			114/f_{CLK}	Setting prohibited	28.5 µs	14.25 µs	7.125 µs	3.5625 µs
1	0	1				f_{CLK}/5			95/f_{CLK}	96 µs	23.75 µs	11.88 µs	5.938 µs	2.9688 µs
1	1	0				f_{CLK}/4			76/f_{CLK}	76 µs	19 µs	9.5 µs	4.75 µs	2.375 µs
1	1	1				f_{CLK}/2			38/f_{CLK}	38 µs	9.5 µs	4.75 µs	2.375 µs	Setting prohibited

Note 1. For the second and subsequent conversion in sequential conversion mode and for conversion of the channel specified by scan 1, 2, and 3 in scan mode, the conversion start time and stabilization wait time for A/D power supply do not occur after a hardware trigger is detected (see **Table 10 - 3 A/D Conversion Time Selection (2/4)**).

Note 2. These are the numbers of clock cycles when conversion is with 10-bit resolution. When eight-bit resolution is selected, the values are shorter by two cycles of the conversion clock (f_{AD}).

Caution 1. The A/D conversion time must also be set within the relevant range of conversion times (t_{CONV}) described in 28.6.1 Characteristics of the A/D converter for T_A = –40 to 85°C. Note that the conversion time (t_{CONV}) does not include the A/D power supply stabilization wait time.

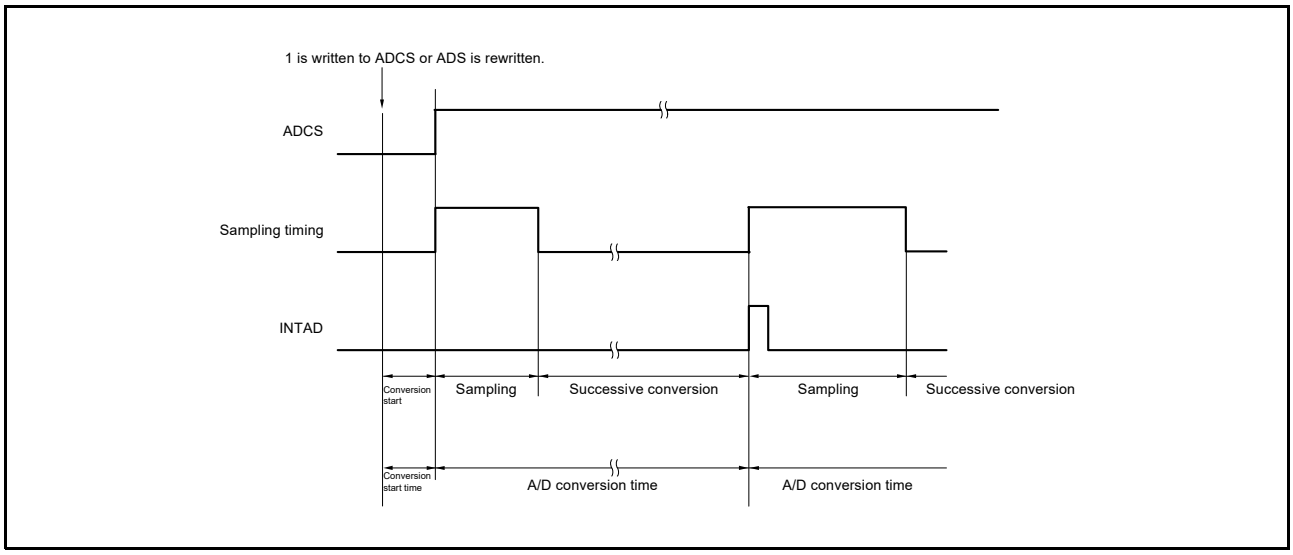
Caution 2. Rewrite the FR2 to FR0, LV1 and LV0 bits to other than the same data while conversion is stopped (ADCS = 0, ADCE = 0).

Caution 3. The conversion times listed above do not include the conversion start time. Add the conversion start time to obtain the time for the first conversion. The conversion times also do not take clock frequency errors into account. Consider clock frequency errors when selecting conversion times.

Caution 4. The conversion times in hardware trigger wait mode include the A/D power supply stabilization wait time from the time the hardware trigger is detected.

Remark f_{CLK}: CPU/peripheral hardware clock frequency

Figure 10 - 6 A/D Converter Sampling and A/D Conversion Timing for Software Trigger Mode



10.3.4 A/D converter mode register 1 (ADM1)

The ADM1 register is used to specify the A/D conversion trigger, conversion mode, and hardware trigger signal. The ADM1 register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 10 - 7 Format of A/D Converter Mode Register 1 (ADM1)

Address: FFF32H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
ADM1	ADTMD1	ADTMD0	ADSCM	0	0	0	ADTRS1	ADTRS0

ADTMD1	ADTMD0	Selection of the A/D conversion trigger mode
0	×	Software trigger mode
1	0	Hardware trigger no-wait mode
1	1	Hardware trigger wait mode

ADSCM	Specification of the A/D conversion mode
0	Sequential conversion mode
1	One-shot conversion mode

ADTRS1	ADTRS0	Selection of the hardware trigger signal
0	0	Timer array unit 0 channel 1 count or capture end interrupt signal (INTTM01)
1	1	Trigger signal (ADITL0) in response to a compare match of the 32-bit interval timer
Other than the above		Setting prohibited

Caution 1. Rewrite the value of the ADM1 register while conversion is stopped (ADCS = 0, ADCE = 0).

Caution 2. To complete A/D conversion, specify at least the following time as the hardware trigger interval:
Hardware trigger no-wait mode: 2 f_{CLK} clock cycles + conversion start time + A/D conversion time
Hardware trigger wait mode: 2 f_{CLK} clock cycles + conversion start time + A/D power supply stabilization wait time + A/D conversion time

Remark 1. ×: Don't care

Remark 2. f_{CLK}: CPU/peripheral hardware clock frequency

10.3.5 A/D converter mode register 2 (ADM2)

The ADM2 register is used to select the positive and negative reference voltages of the A/D converter, select the range of the A/D conversion results that causes interrupt generation, select the resolution, and specify whether to use the SNOOZE mode. The ADM2 register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 10 - 8 Format of A/D Converter Mode Register 2 (ADM2) (1/2)

Address: F0010H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	<3>	<2>	1	<0>
ADM2	ADREFP1	ADREFP0	ADREFM	0	ADRCK	AWC	0	ADTYP

ADREFP1	ADREFP0	Selection of the positive reference voltage source of the A/D converter
0	0	Supplied from V_{DD}
0	1	Supplied from P20/AV_{REFP}/ANI0
1	0	Supplied from the internal reference voltage**Note 1**
1	1	Setting prohibited

- Use the following procedures to rewrite the ADREFP1 and ADREFP0 bits.
  <1> Set ADCE = 0.
  <2> Change the value of the ADREFP1 and ADREFP0 bits.
  <3> Wait for reference voltage stabilization time A.
  <4> Set ADCE = 1.
  <5> Wait for reference voltage stabilization time B.

  Reference voltage stabilization times A and B
  Stabilization of the reference voltage will require the following waiting times; the duration of time A depends on the settings of ADREFP1 and ADREFP0.
  - ADREFP1 and ADREFP0 = 10B
    A = 5 μs
    B = 1 μs
  - ADREFP1 and ADREFP0 = 00B or 01B
    A: Not required.
    B = 1 μs

  After waiting for reference voltage stabilization time B, start the A/D conversion.
- When ADREFP1 and ADREFP0 are set to 10B, the temperature sensor output voltage and internal reference voltage**Note 1** cannot be A/D converted. Be sure to proceed with A/D conversion while ADISS = 0.

ADREFM	Selection of the negative reference voltage of the A/D converter
0	Supplied from V_{SS}
1	Supplied from P21/AV_{REFM}/ANI1

Figure 10 - 8 Format of A/D Converter Mode Register 2 (ADM2) (2/2)

ADRCK	Control of the range of the A/D conversion results that causes interrupt generation
0	The interrupt signal (INTAD) is generated when the result of A/D conversion falls within area 1, where ADLL register ≤ ADCR register ≤ ADUL register.
1	The interrupt signal (INTAD) is generated when the result of A/D conversion falls within area 2 or 3, which are defined as follows. Area 2: ADCR register < ADLL register Area 3: ADUL register < ADCR register
Figure 10 - 9 shows the ranges (areas 1 to 3) in which the interrupt signal (INTAD) is generated.	

AWC	Specification of the SNOOZE mode
0	Do not use the SNOOZE mode function.
1	Use the SNOOZE mode function.
When there is a hardware trigger signal in the STOP mode, the STOP mode is exited, and A/D conversion is performed without operating the CPU (the SNOOZE mode). • The SNOOZE mode function can only be specified when the high-speed on-chip oscillator clock is selected for the CPU/peripheral hardware clock (f_{CLK}). If any other clock is selected, specifying this mode is prohibited. • Using the SNOOZE mode function in the software trigger mode or hardware trigger no-wait mode is prohibited. • Using the SNOOZE mode function in the sequential conversion mode is prohibited. • When using the SNOOZE mode function, specify a hardware trigger interval of at least “shift time to SNOOZE mode**Note 2** + conversion start time + A/D power supply stabilization wait time + A/D conversion time + 2 f_{CLK} clock cycles”. • Even when using the SNOOZE mode function, be sure to set the AWC bit to 0 in normal operation and change it to 1 just before shifting to STOP mode. Also, be sure to change the AWC bit to 0 after returning from STOP mode to normal operation. If the AWC bit is left set to 1, A/D conversion will not start normally in spite of the subsequent SNOOZE mode or normal operation.	

ADTYP	Selection of the A/D conversion resolution
0	10-bit resolution
1	8-bit resolution

Note 1. For the internal reference voltage, see **Section 28 Electrical Characteristics**.

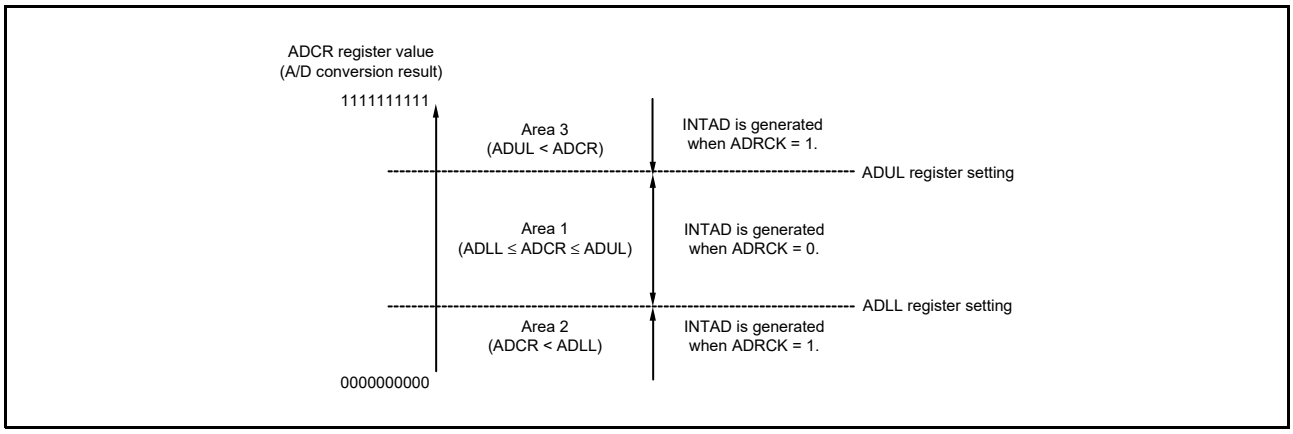
Note 2. Refer to “Transition time from STOP mode to SNOOZE mode” in **16.3.3 SNOOZE mode**.

Caution 1. Rewrite the value of the ADM2 register while conversion is stopped (ADCS = 0, ADCE = 0).

Caution 2. Do not set the ADREFP1 bit to 1 when shifting to STOP mode, or to HALT mode while the CPU is operating on the subsystem clock. When the internal reference voltage is selected (ADREFP1, ADREFP0 = 10B), the A/D converter reference voltage current (I_{ADREF}) indicated in 28.3.2 Characteristics of the supply current will be added.

Caution 3. When using AV_{REFP} and AV_{REFM}, specify ANI0 and ANI1 as the analog input channels and specify input mode by using a port mode register (PMxx).

Figure 10 - 9 Range of Interrupt Signals Generated under Control by the ADRCK Bit



Remark The result of A/D conversion is only stored in the ADCR and ADCRH registers when INTAD is generated.

10.3.6 10-bit A/D conversion result register (ADCR)

The ADCR register is a 16-bit register that holds the A/D conversion result. The lower 6 bits are fixed to 0. Each time A/D conversion is complete, the conversion result is loaded from the successive approximation register (SAR). The higher 8 bits of the conversion result are stored in FFF1FH and the lower 2 bits are stored in the higher 2 bits of FFF1EH**Note**. The ADCR register can be read by a 16-bit memory manipulation instruction. The value of this register following a reset is 0000H.

Note If the A/D conversion result is outside the range specified by using the A/D conversion result comparison function (the value specified by the ADM2.ADRCK bit and ADUL and ADLL registers), the result is not stored. For details, see **Figure 10 - 9**.

Figure 10 - 10 Format of 10-bit A/D Conversion Result Register (ADCR)

Address: FFF1EH, FFF1FH
After reset: 0000H
R/W: R

	FFF1FH								FFF1EH							
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCR											0	0	0	0	0	0

Caution 1. When 8-bit resolution A/D conversion is selected (ADM2.ADTYP = 1) and the ADCR register is read, 0 is read from the bits other than the higher 8 bits.

Caution 2. When the ADCR register is accessed in 16-bit units, and A/D conversion with 10-bit resolution is selected, the higher 10 bits of the conversion result are read in order starting at bit 15 of the ADCR register.

Caution 3. The contents of the ADCR register may become undefined when writing to any of the following registers.

- **A/D converter mode register 0 (ADM0)**
- **Analog input channel specification register (ADS)**
- **Port mode control A registers (PMCA0, PMCA2, PMCA12, PMCA14)**

Read the conversion result following conversion completion before writing to any of these registers. Otherwise, the correct conversion result may not be obtained.

10.3.7 8-bit A/D conversion result register (ADCRH)

The ADCRH register is an 8-bit register that holds the A/D conversion result. The higher 8 bits of 10-bit resolution are stored[Note]. The ADCRH register can be read by an 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Note If the A/D conversion result is outside the range specified by using the A/D conversion result comparison function (the value specified by the ADM2.ADRCK bit and ADUL and ADLL registers), the result is not stored. For details, see **Figure 10 - 9**.

Figure 10 - 11 Format of 8-bit A/D Conversion Result Register (ADCRH)

Address: FFF1FH
After reset: 00H
R/W: R

Symbol	7	6	5	4	3	2	1	0
ADCRH								

Caution The contents of the ADCRH register may become undefined when writing to any of the following registers.
- **A/D converter mode register 0 (ADM0)**
- **Analog input channel specification register (ADS)**
- **Port mode control A registers (PMCA0, PMCA2, PMCA12, PMCA14)**

Read the conversion result following conversion completion before writing to any of these registers. Otherwise, the correct conversion result may not be obtained.

10.3.8 Analog input channel specification register (ADS)

The ADS register specifies the input channel of the analog voltage to be A/D converted. The ADS register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 10 - 12 Format of Analog Input Channel Specification Register (ADS)

Address: FFF31H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
ADS	ADISS	0	0	ADS4	ADS3	ADS2	ADS1	ADS0

<Select mode (ADM0.ADMD = 0)>

ADISS	ADS4	ADS3	ADS2	ADS1	ADS0	Analog input channel	Input source
0	0	0	0	0	0	ANI0	P20/ANI0/AVREFP pin
0	0	0	0	0	1	ANI1	P21/ANI1/AVREFM pin
0	0	0	0	1	0	ANI2	P22/ANI2 pin
0	0	0	0	1	1	ANI3	P23/ANI3 pin
0	0	0	1	0	0	ANI4	P24/ANI4 pin
0	0	0	1	0	1	ANI5	P25/ANI5 pin
0	0	0	1	1	0	ANI6	P26/ANI6 pin
0	0	0	1	1	1	ANI7	P27/ANI7 pin
0	1	0	0	0	0	ANI16	P03/ANI16 pin**Note 1**
0	1	0	0	0	1	ANI17	P02/ANI17 pin**Note 2**
0	1	0	0	1	0	ANI18	P147/ANI18 pin
0	1	0	0	1	1	ANI19	P120/ANI19 pin
1	0	0	0	0	0	—	Temperature sensor output voltage
1	0	0	0	0	1	—	Internal reference voltage**Note 3**
Other than the above						Setting prohibited	

Note 1. P01/ANI16 pin in the 20- and 32-pin products
Note 2. P00/ANI17 pin in the 20- and 32-pin products
Note 3. For the internal reference voltage, see **Section 28 Electrical Characteristics**.

<Scan mode (ADM0.ADMD = 1)>

ADISS	ADS4	ADS3	ADS2	ADS1	ADS0	Analog input channel			
						Scan 0	Scan 1	Scan 2	Scan 3
0	0	0	0	0	0	ANI0	ANI1	ANI2	ANI3
0	0	0	0	0	1	ANI1	ANI2	ANI3	ANI4
0	0	0	0	1	0	ANI2	ANI3	ANI4	ANI5
0	0	0	0	1	1	ANI3	ANI4	ANI5	ANI6
0	0	0	1	0	0	ANI4	ANI5	ANI6	ANI7
Other than the above						Setting prohibited			

Caution 1. Be sure to leave bits 6 and 5 set to 0.

Caution 2. Set the port that is specified as the analog input by a PMCAxx register to the input mode by using a port mode register (PM0, PM2, PM12, or PM14).

Caution 3. When specifying an input channel by the ADS register, do not select the pin that is specified as digital I/O by a port mode control A register (PMCA0, PMCA2, PMCA12, or PMCA14).

Caution 4. Rewrite the value of the ADISS bit while conversion is stopped (ADCS = 0, ADCE = 0).

Caution 5. If using AV_{REFP} as the positive reference voltage of the A/D converter, do not select ANI0 as an A/D conversion channel.

Caution 6. If using AV_{REFM} as the negative reference voltage of the A/D converter, do not select ANI1 as an A/D conversion channel.

Caution 7. If the ADISS bit is set to 1, the internal reference voltage cannot be used for the positive reference voltage. After the ADISS bit is set to 1, the initial conversion result cannot be used. For the setting flow, see 10.7.4 Example of settings when temperature sensor output voltage or internal reference voltage is selected in the software trigger mode and one-shot conversion mode. For details about the internal reference voltage, see Section 28 Electrical Characteristics.

Caution 8. Do not set the ADISS bit to 1 when shifting to STOP mode, or to HALT mode while the CPU is operating on the subsystem clock. When the ADISS bit is set to 1, the A/D converter reference voltage current (I_{ADREF}) indicated in 28.3.2 Characteristics of the supply current will be added.

10.3.9 Conversion result comparison upper limit setting register (ADUL)

The ADUL register sets the upper limit value of the range of A/D conversion results that causes interrupt generation. The A/D conversion results and ADUL register value are compared, and generation of the interrupt signal (INTAD) is controlled in the range specified for the ADRCK bit of A/D converter mode register 2 (ADM2). For details, see **Figure 10 - 9 Range of Interrupt Signals Generated under Control by the ADRCK Bit**. The ADUL register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is FFH.

Figure 10 - 13 Format of Conversion Result Comparison Upper Limit Setting Register (ADUL)

Address: F0011H
After reset: FFH
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
ADUL	ADUL7	ADUL6	ADUL5	ADUL4	ADUL3	ADUL2	ADUL1	ADUL0

Caution 1. When A/D conversion with 10-bit resolution is selected, the eight higher-order bits of the 10-bit A/D conversion result register (ADCR) are compared with the value in the ADUL register.
Caution 2. Only write a new value to the ADUL register while conversion is stopped (ADCS = 0, ADCE = 0).
Caution 3. The setting of the ADUL register must be greater than that of the ADLL register.

10.3.10 Conversion result comparison lower limit setting register (ADLL)

The ADLL register sets the lower limit value of the range of A/D conversion results that causes interrupt generation. The A/D conversion results and ADLL register value are compared, and generation of the interrupt signal (INTAD) is controlled in the range specified for the ADRCK bit of A/D converter mode register 2 (ADM2). For details, see **Figure 10 - 9 Range of Interrupt Signals Generated under Control by the ADRCK Bit**. The ADLL register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 10 - 14 Format of Conversion Result Comparison Lower Limit Setting Register (ADLL)

Address: F0012H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
ADLL	ADLL7	ADLL6	ADLL5	ADLL4	ADLL3	ADLL2	ADLL1	ADLL0

Caution 1. When A/D conversion with 10-bit resolution is selected, the eight higher-order bits of the 10-bit A/D conversion result register (ADCR) are compared with the value in the ADLL register.
Caution 2. Only write a new value to the ADLL register while conversion is stopped (ADCS = 0, ADCE = 0).
Caution 3. The setting of the ADLL register must be less than that of the ADUL register.

10.3.11 A/D test register (ADTES)

The ADTES register is used to select the positive or negative reference voltage for the converter, an analog input channel (ANIxx), the temperature sensor output voltage, or the internal reference voltage[Note] as the target for A/D conversion. When using this register to test the converter, set as follows.

- For zero-scale measurement, select the negative reference voltage as the target for conversion.
- For full-scale measurement, select the positive reference voltage as the target for conversion.

The ADTES register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Note For the internal reference voltage, see **Section 28 Electrical Characteristics**.

Figure 10 - 15 Format of A/D Test Register (ADTES)

Address: F0013H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
ADTES	0	0	0	0	0	0	ADTES1	ADTES0

ADTES1	ADTES0	A/D conversion target
0	0	ANIxx, temperature sensor output voltage, or internal reference voltage[Note] (set by the analog input channel specification register (ADS))
1	0	The negative reference voltage (selected by the ADREFM bit of the ADM2 register)
1	1	The positive reference voltage (selected by the ADREFP1 and ADREFP0 bits of the ADM2 register)
Other than the above		Setting prohibited

Note For the internal reference voltage, see **Section 28 Electrical Characteristics**.

Caution Be sure to leave bits 7 to 2 set to 0.

10.3.12 Registers for controlling the port functions multiplexed with the analog inputs of the A/D converter

Set the following registers to control the port functions multiplexed with the analog inputs of the A/D converter.

- Port mode registers (PMxx)
- Port mode control A registers (PMCAxx)

For details, see **4.3.1 Port mode registers (PMxx)** and **4.3.7 Port mode control A registers (PMCAxx)**.

For examples of register settings for controlling port functions, see **4.5.4 Examples of register settings for port and alternate functions**.

10.4 A/D Converter Conversion Operations

The A/D converter conversion operations are described below.

<1> The voltage input to the selected analog input channel is sampled by the sample & hold (S&H) circuit.

<2> When sampling has been done for a certain time, the S&H circuit is placed in the hold state and the sampled voltage is held until the A/D conversion operation has ended.

<3> Bit 9 of the successive approximation register (SAR) is set. The series resistor string voltage tap is set to (1/2) × AV_{REF} by the tap selector.

<4> The voltage difference between the series resistor string voltage tap and sampled voltage is compared by the voltage comparator. If the analog input is greater than (1/2) × AV_{REF}, the MSB bit of the SAR register remains set to 1. If the analog input is smaller than (1/2) × AV_{REF}, the MSB bit is reset to 0.

<5> Next, bit 8 of the SAR register is automatically set to 1, and the operation proceeds to the next comparison. The series resistor string voltage tap is selected according to the value of bit 9 already compared, as described below.

- Bit 9 = 1: (3/4) × AV_{REF}
- Bit 9 = 0: (1/4) × AV_{REF}

The voltage tap and sampled voltage are compared and bit 8 of the SAR register is manipulated as follows.

- Sampled voltage ≥ Voltage tap: Bit 8 = 1
- Sampled voltage < Voltage tap: Bit 8 = 0

<6> Comparison is continued in this way up to bit 0 of the SAR register.

<7> When the comparison of 10 bits is completed, a series of comparison results remains in the SAR register, and its value is transferred to the A/D conversion result registers (ADCR, ADCRH) and then latched**Note 1**. At the same time, the A/D conversion end interrupt request signal (INTAD) can also be generated**Note 1**.

<8> Repeat steps <1> to <7>, until the ADCS bit is cleared to 0**Note 2**. To stop the A/D converter, clear the ADCS bit to 0.

Note 1. If the A/D conversion result is outside the A/D conversion result range specified by the ADM2.ADRCK bit and the ADUL and ADLL registers, the A/D conversion end interrupt request signal (INTAD) is not generated and no A/D conversion results are stored in the ADCR and ADCRH registers. For details, see **Figure 10 - 9 Range of Interrupt Signals Generated under Control by the ADRCK Bit**.

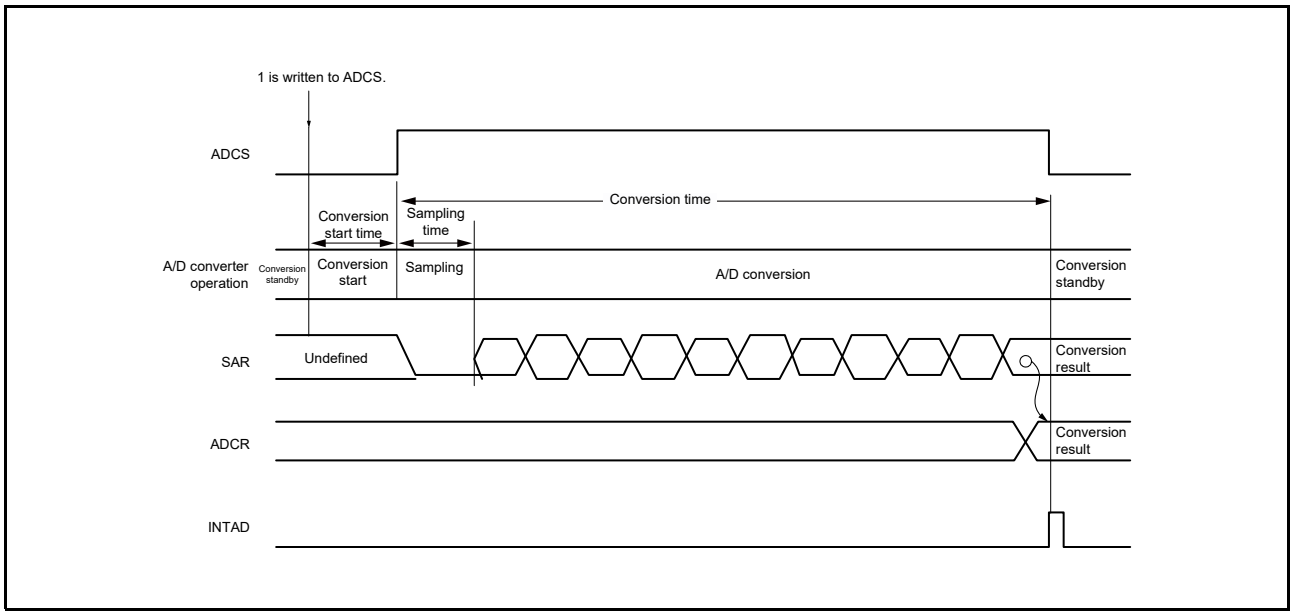
Note 2. While in the sequential conversion mode, the ADCS flag is not automatically cleared to 0. This flag is not automatically cleared to 0 while in the one-shot conversion mode of the hardware trigger no-wait mode, either. Instead, 1 is retained.

Remark 1. Two types of the A/D conversion result registers are available.

- ADCR register (16 bits) to hold 10-bit A/D conversion value
- ADCRH register (8 bits) to hold 8-bit A/D conversion value

Remark 2. AV_{REF}: The positive reference voltage of the A/D converter. This can be selected from AV_{REFP}, the internal reference voltage, and V_{DD}. For the internal reference voltage, see **Section 28 Electrical Characteristics**.

Figure 10 - 16 Operation of the A/D Converter in the Software Trigger Mode



In one-shot conversion mode, the ADCS bit is automatically cleared to 0 after completion of A/D conversion. In sequential conversion mode, A/D conversion operations proceed continuously until the software clears the ADCS bit of A/D converter mode register 0 (ADM0) to 0. Writing to the analog input channel specification register (ADS) during A/D conversion aborts the current A/D conversion and starts a new conversion for the analog input newly specified by the ADS register. Any data from the aborted conversion are discarded. The values of the A/D conversion result registers (ADCR and ADCRH) following a reset are 0000H and 00H, respectively.

10.5 Input Voltage and Conversion Results

The relationship between the analog input voltage input to the analog input pins (ANI0 to ANI7, ANI16 to ANI19) and the theoretical A/D conversion result (stored in the 10-bit A/D conversion result register (ADCR)) is shown by the following expression.

$$SAR = INT\left(\frac{V_{AIN}}{AV_{REF}} \times 1024 + 0.5\right)$$

$$ADCR = SAR \times 64$$

or

<R>

$$\left(\frac{ADCR}{64} - 0.5\right) \times \frac{AV_{REF}}{1024} \leq V_{AIN} < \left(\frac{ADCR}{64} + 0.5\right) \times \frac{AV_{REF}}{1024}$$

where, INT( ): Function which returns integer part of value in parentheses

V_{AIN}: Analog input voltage

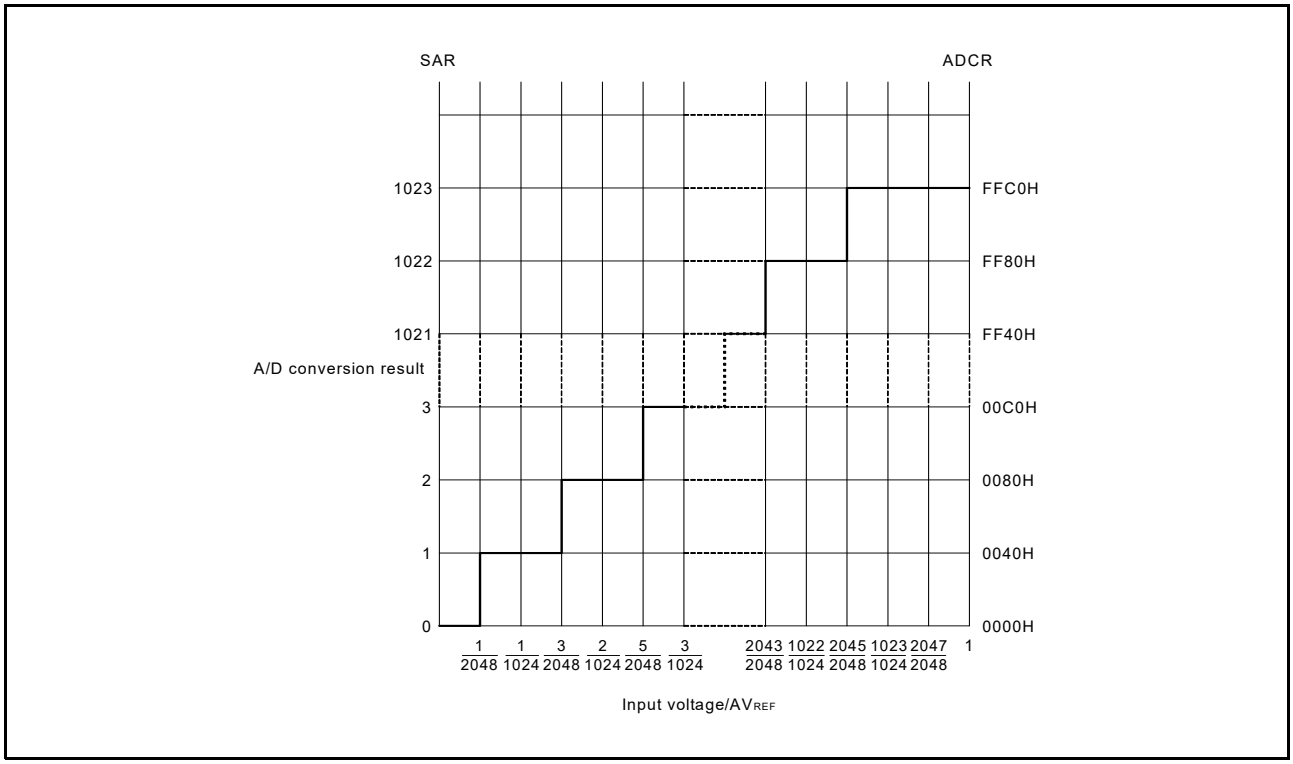
AV_{REF}: AV_{REF} pin voltage

ADCR: 10-bit A/D conversion result register (ADCR) value

SAR: Successive approximation register

Figure 10 - 17 shows the relationship between the analog input voltage and the A/D conversion result.

Figure 10 - 17 Relationship Between Analog Input Voltage and A/D Conversion Result



Remark AV_{REF}: The positive reference voltage of the A/D converter. This can be selected from AV_{REFP}, the internal reference voltage, and V_{DD}. For the internal reference voltage, see **Section 28 Electrical Characteristics**.

10.6 Operating Modes of the A/D Converter

This section describes the operation of the A/D converter modes. For the procedures to set each of the modes, see **10.7 Flows of Setting the A/D Converter Modes**.

10.6.1 Software trigger mode: Select mode and sequential conversion mode

<1> Setting the ADCE bit of A/D converter mode register 0 (ADM0) to 1 while conversion is stopped places the A/D converter in the standby state.

<2> To allow stabilization of the internal circuit, apply software control for a 1-μs wait, then set the ADCS bit in the ADM0 register to 1 to start A/D conversion of the analog input specified by the analog input channel specification register (ADS).

<3> Once A/D conversion is complete, the result of conversion is stored in the A/D conversion result registers (ADCR and ADCRH), and an A/D conversion end interrupt request signal (INTAD) is generated. The next A/D conversion starts at the same time.

<4> Overwriting the ADCS bit with 1 during conversion aborts the current A/D conversion and starts a new conversion. Any data from the aborted conversion are discarded.

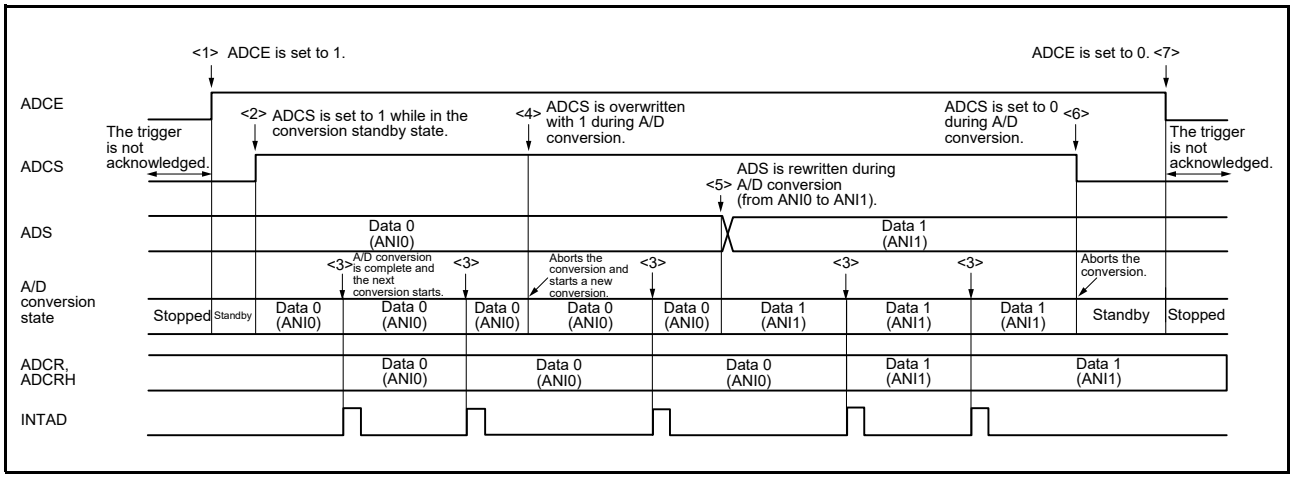
<5> Writing to the ADS register during conversion aborts the current A/D conversion and starts a new conversion for the analog input newly specified by the ADS register. Any data from the aborted conversion are discarded.

<6> Setting the ADCS bit to 0 during conversion aborts the current A/D conversion, and places the A/D converter in the standby state.

<7> Setting the ADCE bit to 0 while the A/D converter is in the standby state places the converter in the stopped state. While the ADCE bit is 0, setting the ADCS bit to 1 is ignored, and A/D conversion does not start in response.

Caution In software trigger mode, a hardware trigger does not start A/D conversion.

Figure 10 - 18 Example of Timing of Operations in the Software Trigger Mode: Select Mode and Sequential Conversion Mode



10.6.2 Software trigger mode: Select mode and one-shot conversion mode

<1> Setting the ADCE bit of A/D converter mode register 0 (ADM0) to 1 while conversion is stopped places the A/D converter in the standby state.

<2> To allow stabilization of the internal circuit, apply software control for a 1-μs wait, then set the ADCS bit in the ADM0 register to 1 to start A/D conversion of the analog input specified by the analog input channel specification register (ADS).

<3> Once A/D conversion is complete, the result of conversion is stored in the A/D conversion result registers (ADCR and ADCRH), and an A/D conversion end interrupt request signal (INTAD) is generated.

<4> Once A/D conversion is complete, the ADCS bit is automatically cleared to 0, and the converter is placed in the standby state.

<5> Overwriting the ADCS bit with 1 during conversion aborts the current A/D conversion and starts a new conversion. Any data from the aborted conversion are discarded.

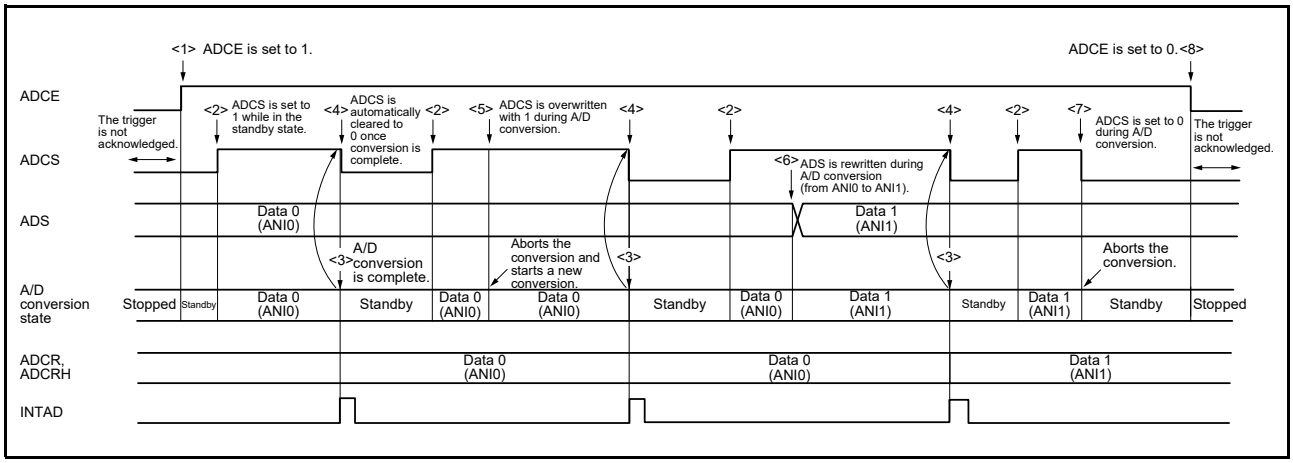
<6> Writing to the ADS register during conversion aborts the current A/D conversion and starts a new conversion for the analog input newly specified by the ADS register. Any data from the aborted conversion are discarded.

<7> Setting the ADCS bit to 0 during conversion aborts the current A/D conversion, and places the A/D converter in the standby state.

<8> Setting the ADCE bit to 0 while the A/D converter is in the standby state places the converter in the stopped state. While the ADCE bit is 0, setting the ADCS bit to 1 is ignored, and A/D conversion does not start in response.

Caution In software trigger mode, a hardware trigger does not start A/D conversion.

Figure 10 - 19 Example of Timing of Operations in the Software Trigger Mode: Select Mode and One-shot Conversion Mode



10.6.3 Software trigger mode: Scan mode and sequential conversion mode

<1> Setting the ADCE bit of A/D converter mode register 0 (ADM0) to 1 while conversion is stopped places the A/D converter in the standby state.

<2> To allow stabilization of the internal circuit, apply software control for a 1-μs wait, then set the ADCS bit in the ADM0 register to 1 to start A/D conversion on the four analog input channels specified as scan 0 to scan 3 by the analog input channel specification register (ADS). A/D conversion proceeds sequentially, starting with the channel specified as scan 0.

<3> The four analog input channels are sequentially converted. Each time a conversion is completed, the result of conversion is stored in the A/D conversion result registers (ADCR and ADCRH), and an A/D conversion end interrupt request signal (INTAD) is generated. On completion of conversion for all four channels, the next A/D conversion of the specified channels is automatically started.

<4> Overwriting the ADCS bit with 1 during conversion aborts the current A/D conversion and starts a new conversion from the first channel. Any data from the aborted conversion are discarded.

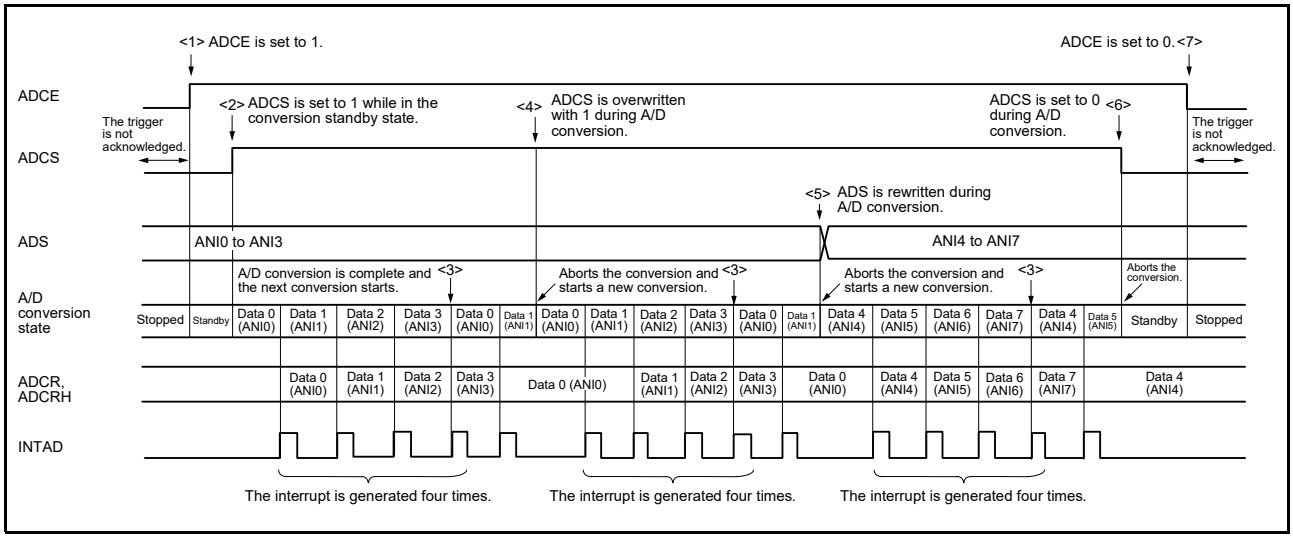
<5> Writing to the ADS register during conversion aborts the current A/D conversion and starts a new conversion from the first channel newly specified by the ADS register. Any data from the aborted conversion are discarded.

<6> Setting the ADCS bit to 0 during conversion aborts the current A/D conversion, and places the A/D converter in the standby state.

<7> Setting the ADCE bit to 0 while the A/D converter is in the standby state places the converter in the stopped state. While the ADCE bit is 0, setting the ADCS bit to 1 is ignored, and A/D conversion does not start in response.

Caution In software trigger mode, a hardware trigger does not start A/D conversion.

Figure 10 - 20 Example of Timing of Operations in the Software Trigger Mode: Scan Mode and Sequential Conversion Mode



10.6.4 Software trigger mode: Scan mode and one-shot conversion mode

<1> Setting the ADCE bit of A/D converter mode register 0 (ADM0) to 1 while conversion is stopped places the A/D converter in the standby state.

<2> To allow stabilization of the internal circuit, apply software control for a 1-μs wait, then set the ADCS bit in the ADM0 register to 1 to start A/D conversion on the four analog input channels specified as scan 0 to scan 3 by the analog input channel specification register (ADS). A/D conversion proceeds sequentially, starting with the channel specified as scan 0.

<3> The four analog input channels are sequentially converted. Each time a conversion is completed, the result of conversion is stored in the A/D conversion result registers (ADCR and ADCRH), and an A/D conversion end interrupt request signal (INTAD) is generated.

<4> Once A/D conversion for all four channels is complete, the ADCS bit is automatically cleared to 0, and the converter is placed in the standby state.

<5> Overwriting the ADCS bit with 1 during conversion aborts the current A/D conversion and starts a new conversion from the first channel. Any data from the aborted conversion are discarded.

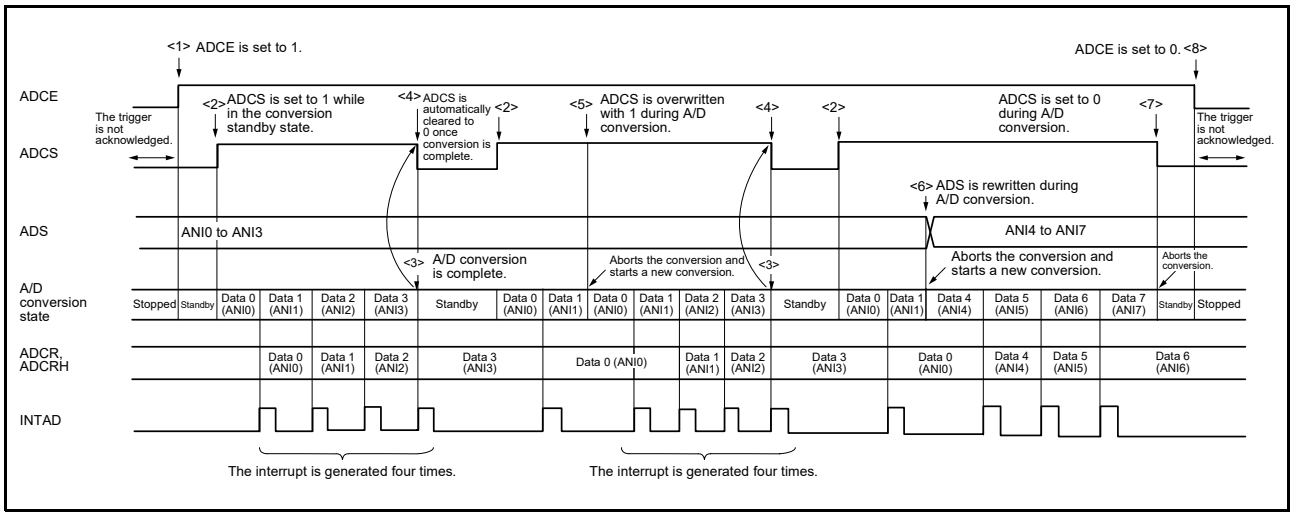
<6> Writing to the ADS register during conversion aborts the current A/D conversion and starts a new conversion from the first channel newly specified by the ADS register. Any data from the aborted conversion are discarded.

<7> Setting the ADCS bit to 0 during conversion aborts the current A/D conversion, and places the A/D converter in the standby state.

<8> Setting the ADCE bit to 0 while the A/D converter is in the standby state places the converter in the stopped state. While the ADCE bit is 0, setting the ADCS bit to 1 is ignored, and A/D conversion does not start in response.

Caution In software trigger mode, a hardware trigger does not start A/D conversion.

Figure 10 - 21 Example of Timing of Operations in the Software Trigger Mode: Scan Mode and One-shot Conversion Mode



10.6.5 Hardware trigger no-wait mode: Select mode and sequential conversion mode

<1> Setting the ADCE bit of A/D converter mode register 0 (ADM0) to 1 while conversion is stopped places the A/D converter in the standby state.

<2> To allow stabilization of the internal circuit, apply software control for a 1-μs wait, then set the ADCS bit in the ADM0 register to 1 to place the A/D converter in the hardware trigger standby state. Note that, while in this state, A/D conversion does not start even if the ADCS bit is set to 1.

<3> Input of a hardware trigger while the ADCS bit is 1 starts A/D conversion of the analog input specified by the analog input channel specification register (ADS).

<4> Once A/D conversion is complete, the result of conversion is stored in the A/D conversion result registers (ADCR and ADCRH), and an A/D conversion end interrupt request signal (INTAD) is generated. The next A/D conversion starts at the same time.

<5> Input of a hardware trigger during conversion aborts the current A/D conversion and starts a new conversion. Any data from the aborted conversion are discarded.

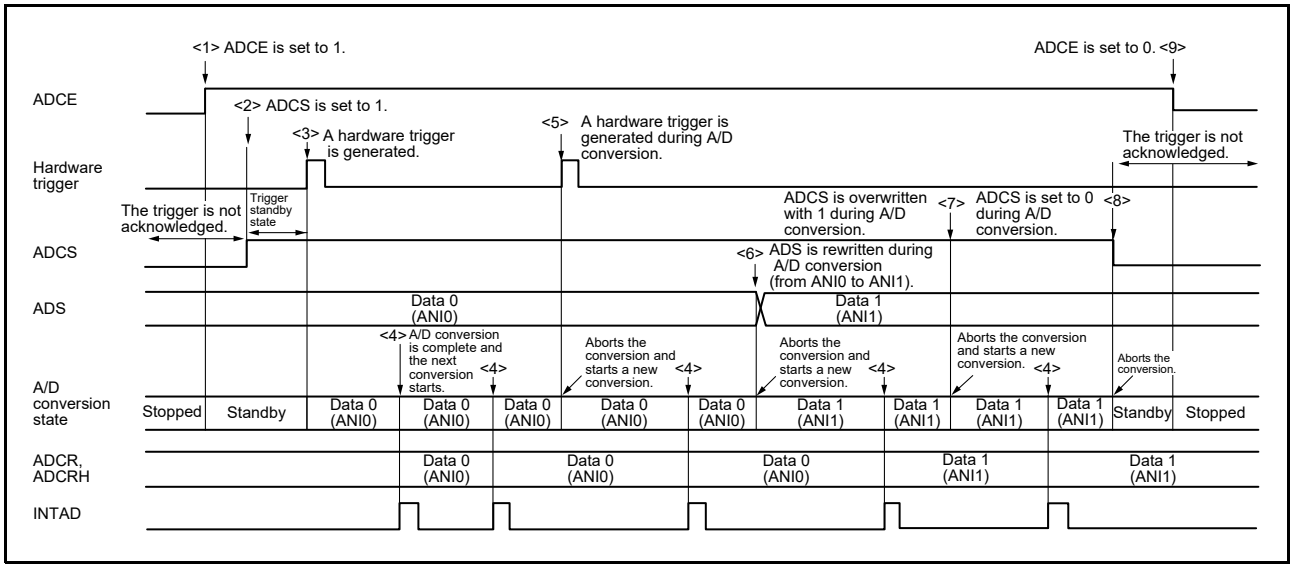
<6> Writing to the ADS register during conversion aborts the current A/D conversion and starts a new conversion for the analog input newly specified by the ADS register. Any data from the aborted conversion are discarded.

<7> Overwriting the ADCS bit with 1 during conversion aborts the current A/D conversion and starts a new conversion. Any data from the aborted conversion are discarded.

<8> Setting the ADCS bit to 0 during conversion aborts the current A/D conversion, and places the A/D converter in the standby state. While the ADCS bit is 0, input of a hardware trigger is ignored, and A/D conversion does not start in response.

<9> Setting the ADCE bit to 0 while the A/D converter is in the standby state places the converter in the stopped state.

Figure 10 - 22 Example of Timing of Operations in the Hardware Trigger No-wait Mode: Select Mode and Sequential Conversion Mode



10.6.6 Hardware trigger no-wait mode: Select mode and one-shot conversion mode

<1> Setting the ADCE bit of A/D converter mode register 0 (ADM0) to 1 while conversion is stopped places the A/D converter in the standby state.

<2> To allow stabilization of the internal circuit, apply software control for a 1-μs wait, then set the ADCS bit in the ADM0 register to 1 to place the A/D converter in the hardware trigger standby state. Note that, while in this state, A/D conversion does not start even if the ADCS bit is set to 1.

<3> Input of a hardware trigger while the ADCS bit is 1 starts A/D conversion of the analog input specified by the analog input channel specification register (ADS).

<4> Once A/D conversion is complete, the result of conversion is stored in the A/D conversion result registers (ADCR and ADCRH), and an A/D conversion end interrupt request signal (INTAD) is generated.

<5> Once A/D conversion is complete, the converter is placed in the standby state with the ADCS bit remaining set to 1.

<6> Input of a hardware trigger during conversion aborts the current A/D conversion and starts a new conversion. Any data from the aborted conversion are discarded.

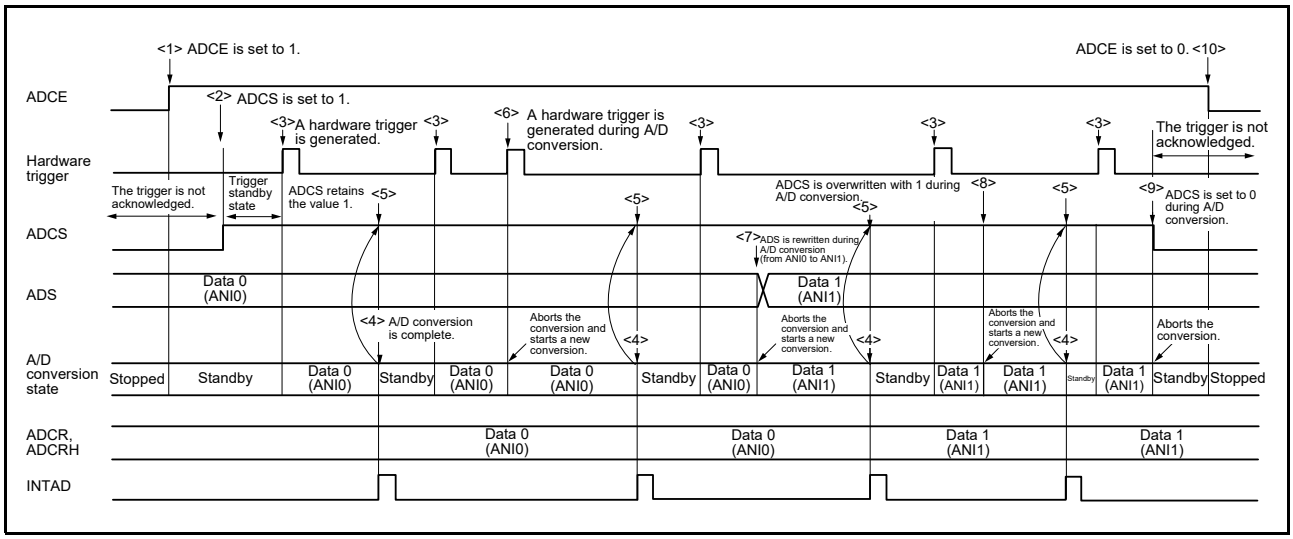
<7> Writing to the ADS register during conversion aborts the current A/D conversion and starts a new conversion for the analog input newly specified by the ADS register. Any data from the aborted conversion are discarded.

<8> Overwriting the ADCS bit with 1 during conversion aborts the current A/D conversion and starts a new conversion. Any data from the aborted conversion are discarded.

<9> Setting the ADCS bit to 0 during conversion aborts the current A/D conversion, and places the A/D converter in the standby state. While the ADCS bit is 0, input of a hardware trigger is ignored, and A/D conversion does not start in response.

<10> Setting the ADCE bit to 0 while the A/D converter is in the standby state places the converter in the stopped state.

Figure 10 - 23 Example of Timing of Operations in the Hardware Trigger No-wait Mode: Select Mode and One-shot Conversion Mode



10.6.7 Hardware trigger no-wait mode: Scan mode and sequential conversion mode

<1> Setting the ADCE bit of A/D converter mode register 0 (ADM0) to 1 while conversion is stopped places the A/D converter in the standby state.

<2> To allow stabilization of the internal circuit, apply software control for a 1-μs wait, then set the ADCS bit in the ADM0 register to 1 to place the A/D converter in the hardware trigger standby state. Note that, while in this state, A/D conversion does not start even if the ADCS bit is set to 1.

<3> Input of a hardware trigger while the ADCS bit is 1 starts A/D conversion on the four analog input channels specified as scan 0 to scan 3 by the analog input channel specification register (ADS). A/D conversion proceeds sequentially, starting with the channel specified as scan 0.

<4> The four analog input channels are sequentially converted. Each time a conversion is completed, the result of conversion is stored in the A/D conversion result registers (ADCR and ADCRH), and an A/D conversion end interrupt request signal (INTAD) is generated. On completion of conversion for all four channels, the next A/D conversion of the specified channels is automatically started.

<5> Input of a hardware trigger during conversion aborts the current A/D conversion and starts a new conversion from the first channel. Any data from the aborted conversion are discarded.

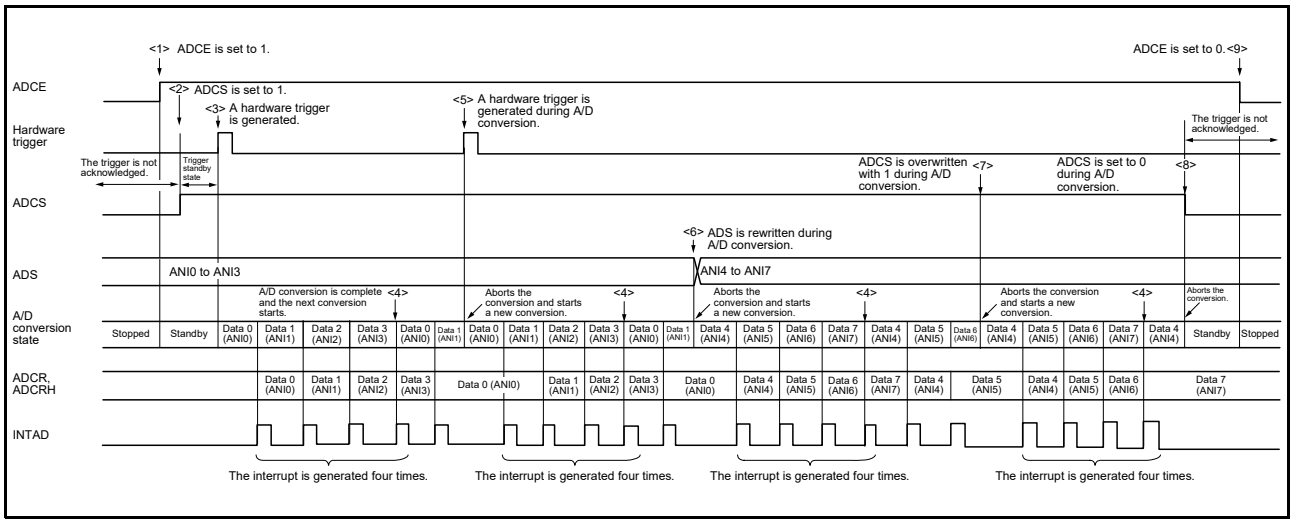
<6> Writing to the ADS register during conversion aborts the current A/D conversion and starts a new conversion from the first channel newly specified by the ADS register. Any data from the aborted conversion are discarded.

<7> Overwriting the ADCS bit with 1 during conversion aborts the current A/D conversion and starts a new conversion. Any data from the aborted conversion are discarded.

<8> Setting the ADCS bit to 0 during conversion aborts the current A/D conversion, and places the A/D converter in the standby state. While the ADCS bit is 0, input of a hardware trigger is ignored, and A/D conversion does not start in response.

<9> Setting the ADCE bit to 0 while the A/D converter is in the standby state places the converter in the stopped state.

Figure 10 - 24 Example of Timing of Operations in the Hardware Trigger No-wait Mode: Scan Mode and Sequential Conversion Mode



10.6.8 Hardware trigger no-wait mode: Scan mode and one-shot conversion mode

<1> Setting the ADCE bit of A/D converter mode register 0 (ADM0) to 1 while conversion is stopped places the A/D converter in the standby state.

<2> To allow stabilization of the internal circuit, apply software control for a 1-μs wait, then set the ADCS bit in the ADM0 register to 1 to place the A/D converter in the hardware trigger standby state. Note that, while in this state, A/D conversion does not start even if the ADCS bit is set to 1.

<3> Input of a hardware trigger while the ADCS bit is 1 starts A/D conversion on the four analog input channels specified as scan 0 to scan 3 by the analog input channel specification register (ADS). A/D conversion proceeds sequentially, starting with the channel specified as scan 0.

<4> The four analog input channels are sequentially converted. Each time a conversion is completed, the result of conversion is stored in the A/D conversion result registers (ADCR and ADCRH), and an A/D conversion end interrupt request signal (INTAD) is generated.

<5> Once A/D conversion for all four channels is complete, the converter is placed in the standby state with the ADCS bit remaining set to 1.

<6> Input of a hardware trigger during conversion aborts the current A/D conversion and starts a new conversion from the first channel. Any data from the aborted conversion are discarded.

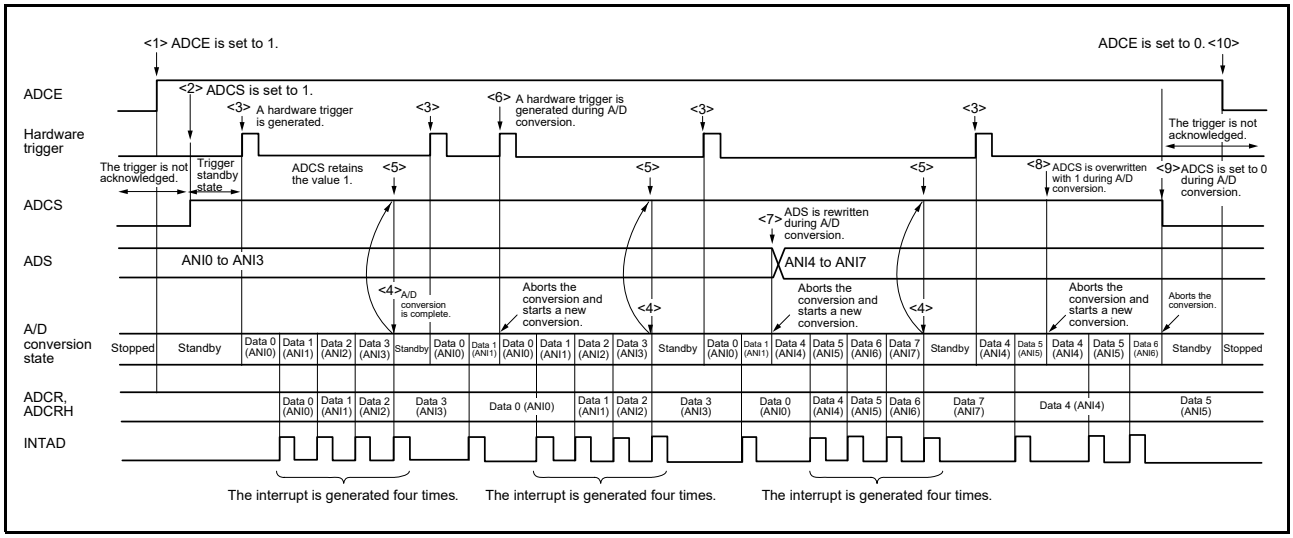
<7> Writing to the ADS register during conversion aborts the current A/D conversion and starts a new conversion from the first channel newly specified by the ADS register. Any data from the aborted conversion are discarded.

<8> Overwriting the ADCS bit with 1 during conversion aborts the current A/D conversion and starts a new conversion from the first channel. Any data from the aborted conversion are discarded.

<9> Setting the ADCS bit to 0 during conversion aborts the current A/D conversion, and places the A/D converter in the standby state. While the ADCS bit is 0, input of a hardware trigger is ignored, and A/D conversion does not start in response.

<10> Setting the ADCE bit to 0 while the A/D converter is in the standby state places the converter in the stopped state.

Figure 10 - 25 Example of Timing of Operations in the Hardware Trigger No-wait Mode: Scan Mode and One-shot Conversion Mode



10.6.9 Hardware trigger wait mode: Select mode and sequential conversion mode

<1> Setting the ADCE bit of A/D converter mode register 0 (ADM0) to 1 while conversion is stopped places the A/D converter in the hardware trigger standby state.

<2> Input of a hardware trigger while in the hardware trigger standby state starts A/D conversion of the analog input specified by the analog input channel specification register (ADS). The ADCS bit of the ADM0 register is automatically set to 1 in response to the input of a hardware trigger.

<3> Once A/D conversion is complete, the conversion result is stored in the A/D conversion result registers (ADCR and ADCRH), an A/D conversion end interrupt request signal (INTAD) is generated, and the next A/D conversion automatically starts without a hardware trigger, simultaneously.

<4> Input of a hardware trigger during conversion aborts the current A/D conversion and starts a new conversion. Any data from the aborted conversion are discarded.

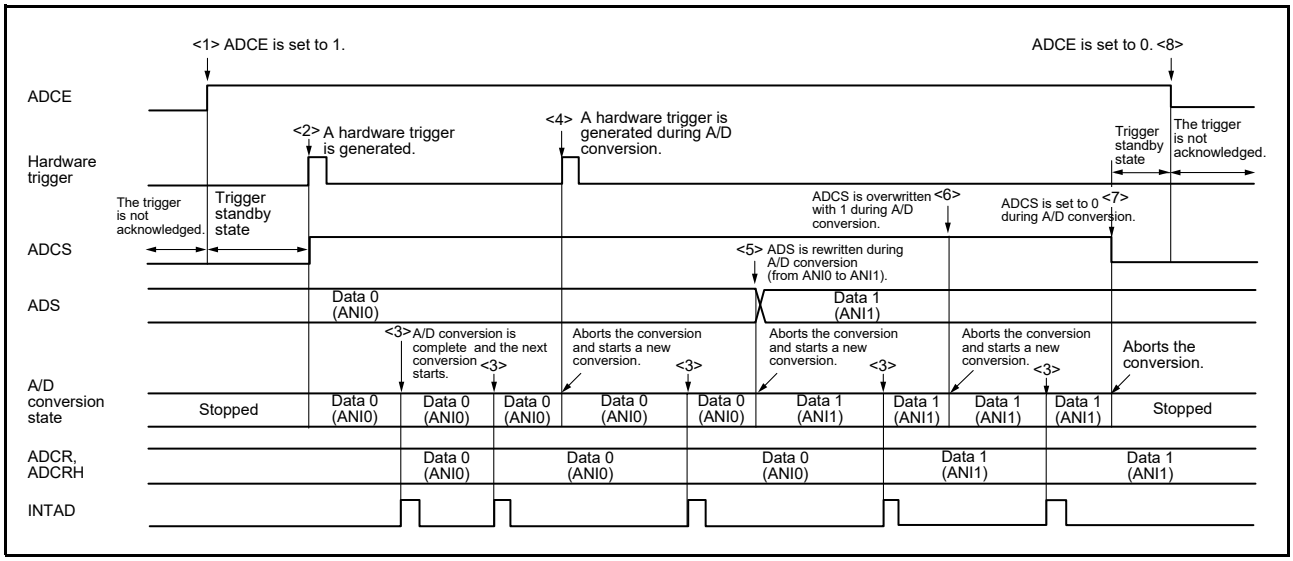
<5> Writing to the ADS register during conversion aborts the current A/D conversion and starts a new conversion for the analog input newly specified by the ADS register. Any data from the aborted conversion are discarded.

<6> Overwriting the ADCS bit with 1 during conversion aborts the current A/D conversion and starts a new conversion. Any data from the aborted conversion are discarded.

<7> Setting the ADCS bit to 0 during conversion aborts the current A/D conversion, and places the A/D converter in the hardware trigger standby state.

<8> Setting the ADCE bit to 0 while the A/D converter is in the hardware trigger standby state places the converter in the stopped state. While the ADCE bit is 0, input of a hardware trigger is ignored, and A/D conversion does not start in response.

Figure 10 - 26 Example of Timing of Operations in the Hardware Trigger Wait Mode: Select Mode and Sequential Conversion Mode



10.6.10 Hardware trigger wait mode: Select mode and one-shot conversion mode

<1> Setting the ADCE bit of A/D converter mode register 0 (ADM0) to 1 while conversion is stopped places the A/D converter in the hardware trigger standby state.

<2> Input of a hardware trigger while in the hardware trigger standby state starts A/D conversion of the analog input specified by the analog input channel specification register (ADS). The ADCS bit of the ADM0 register is automatically set to 1 in response to the input of a hardware trigger.

<3> Once A/D conversion is complete, the conversion result is stored in the A/D conversion result registers (ADCR and ADCRH), and an A/D conversion end interrupt request signal (INTAD) is generated.

<4> Once A/D conversion is complete, the ADCS bit is automatically cleared to 0, and the converter is placed in the standby state.

<5> Input of a hardware trigger during conversion aborts the current A/D conversion and starts a new conversion. Any data from the aborted conversion are discarded.

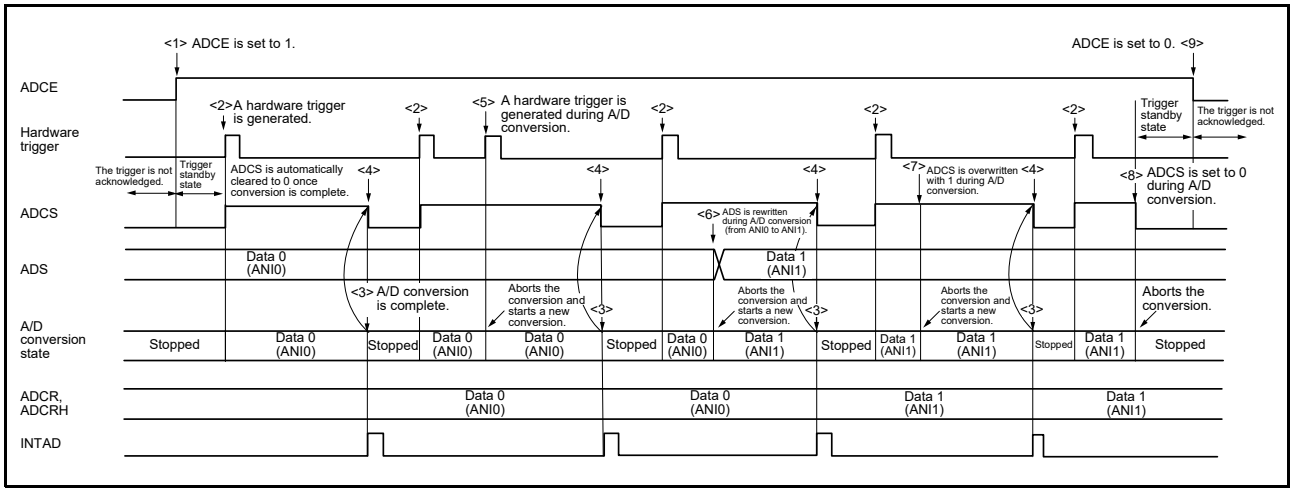
<6> Writing to the ADS register during conversion aborts the current A/D conversion and starts a new conversion for the analog input newly specified by the ADS register. Any data from the aborted conversion are discarded.

<7> Overwriting the ADCS bit with 1 during conversion aborts the current A/D conversion and starts a new conversion. Any data from the aborted conversion are discarded.

<8> Setting the ADCS bit to 0 during conversion aborts the current A/D conversion, and places the A/D converter in the hardware trigger standby state.

<9> Setting the ADCE bit to 0 while the A/D converter is in the hardware trigger standby state places the converter in the stopped state. While the ADCE bit is 0, input of a hardware trigger is ignored, and A/D conversion does not start in response.

Figure 10 - 27 Example of Timing of Operations in the Hardware Trigger Wait Mode: Select Mode and One-shot Conversion Mode



10.6.11 Hardware trigger wait mode: Scan mode and sequential conversion mode

<1> Setting the ADCE bit of A/D converter mode register 0 (ADM0) to 1 while conversion is stopped places the A/D converter in the hardware trigger standby state.

<2> Input of a hardware trigger while in the hardware trigger standby state starts A/D conversion on the four analog input channels specified as scan 0 to scan 3 by the analog input channel specification register (ADS). The ADCS bit of the ADM0 register is automatically set to 1 in response to the input of a hardware trigger. A/D conversion proceeds sequentially, starting with the channel specified as scan 0.

<3> The four analog input channels are sequentially converted. Each time a conversion is completed, the result of conversion is stored in the A/D conversion result registers (ADCR and ADCRH), and an A/D conversion end interrupt request signal (INTAD) is generated. On completion of conversion for all four channels, the next A/D conversion of the specified channels is automatically started.

<4> Input of a hardware trigger during conversion aborts the current A/D conversion and starts a new conversion from the first channel. Any data from the aborted conversion are discarded.

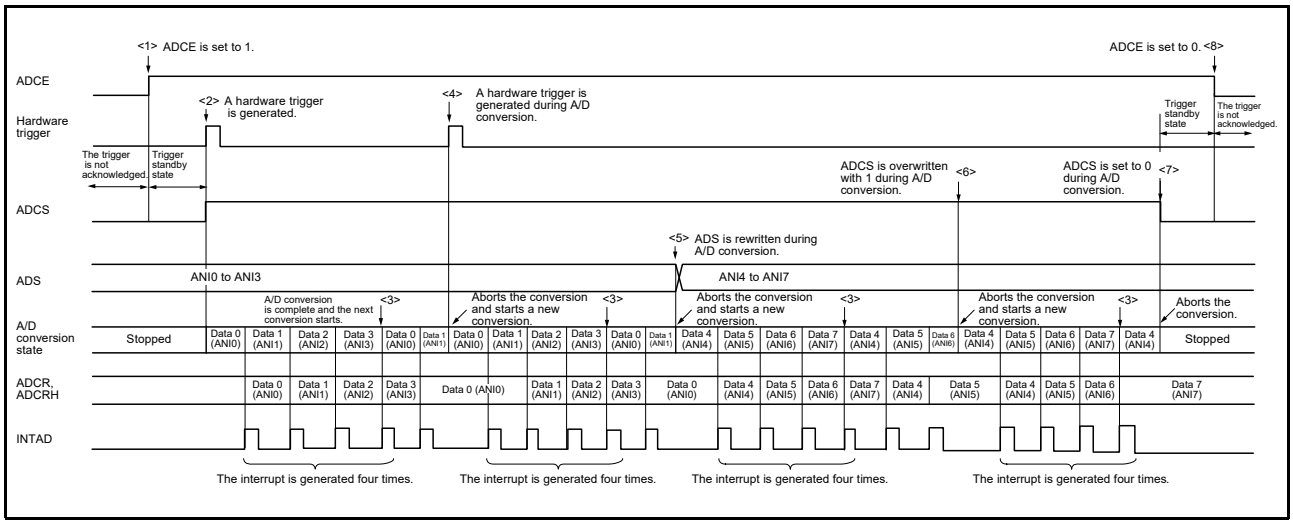
<5> Writing to the ADS register during conversion aborts the current A/D conversion and starts a new conversion from the first channel newly specified by the ADS register. Any data from the aborted conversion are discarded.

<6> Overwriting the ADCS bit with 1 during conversion aborts the current A/D conversion and starts a new conversion. Any data from the aborted conversion are discarded.

<7> Setting the ADCS bit to 0 during conversion aborts the current A/D conversion, and places the A/D converter in the hardware trigger standby state.

<8> Setting the ADCE bit to 0 while the A/D converter is in the hardware trigger standby state places the converter in the stopped state. While the ADCE bit is 0, input of a hardware trigger is ignored, and A/D conversion does not start in response.

Figure 10 - 28 Example of Timing of Operations in the Hardware Trigger Wait Mode: Scan Mode and Sequential Conversion Mode



10.6.12 Hardware trigger wait mode: Scan mode and one-shot conversion mode

<1> Setting the ADCE bit of A/D converter mode register 0 (ADM0) to 1 while conversion is stopped places the A/D converter in the hardware trigger standby state.

<2> Input of a hardware trigger while in the hardware trigger standby state starts A/D conversion on the four analog input channels specified as scan 0 to scan 3 by the analog input channel specification register (ADS). The ADCS bit of the ADM0 register is automatically set to 1 in response to the input of a hardware trigger. A/D conversion proceeds sequentially, starting with the channel specified as scan 0.

<3> The four analog input channels are sequentially converted. Each time a conversion is completed, the result of conversion is stored in the A/D conversion result registers (ADCR and ADCRH), and an A/D conversion end interrupt request signal (INTAD) is generated.

<4> Once A/D conversion is complete, the ADCS bit is automatically cleared to 0, and the converter is placed in the standby state.

<5> Input of a hardware trigger during conversion aborts the current A/D conversion and starts a new conversion from the first channel. Any data from the aborted conversion are discarded.

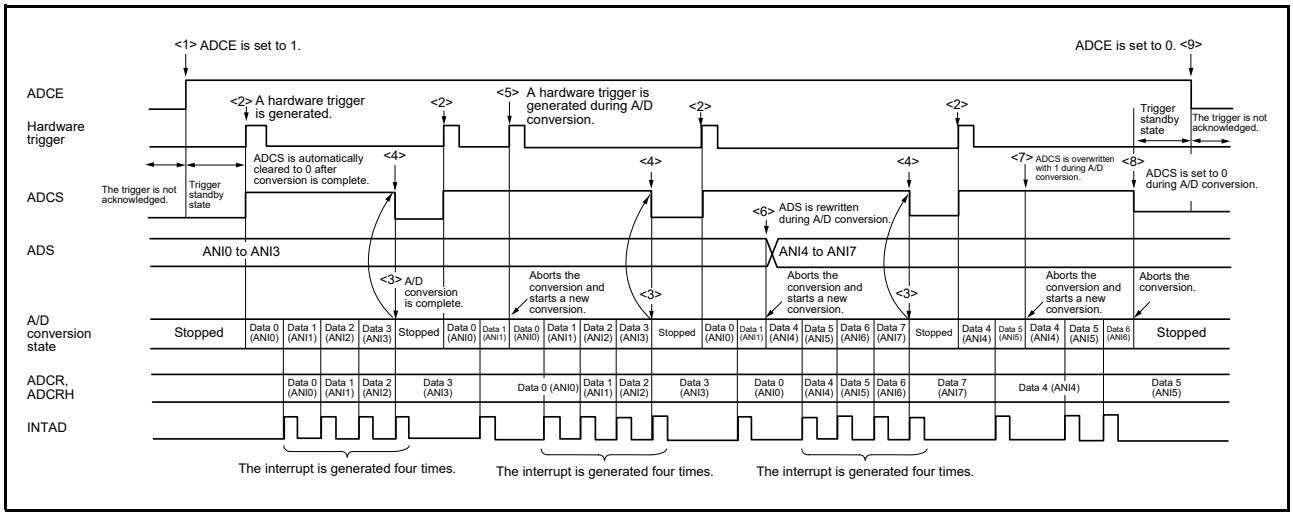
<6> Writing to the ADS register during conversion aborts the current A/D conversion and starts a new conversion from the first channel newly specified by the ADS register. Any data from the aborted conversion are discarded.

<7> Overwriting the ADCS bit with 1 during conversion aborts the current A/D conversion and starts a new conversion. Any data from the aborted conversion are discarded.

<8> Setting the ADCS bit to 0 during conversion aborts the current A/D conversion, and places the A/D converter in the hardware trigger standby state.

<9> Setting the ADCE bit to 0 while the A/D converter is in the hardware trigger standby state places the converter in the stopped state. While the ADCE bit is 0, input of a hardware trigger is ignored, and A/D conversion does not start in response.

Figure 10 - 29 Example of Timing of Operations in the Hardware Trigger Wait Mode: Scan Mode and One-shot Conversion Mode

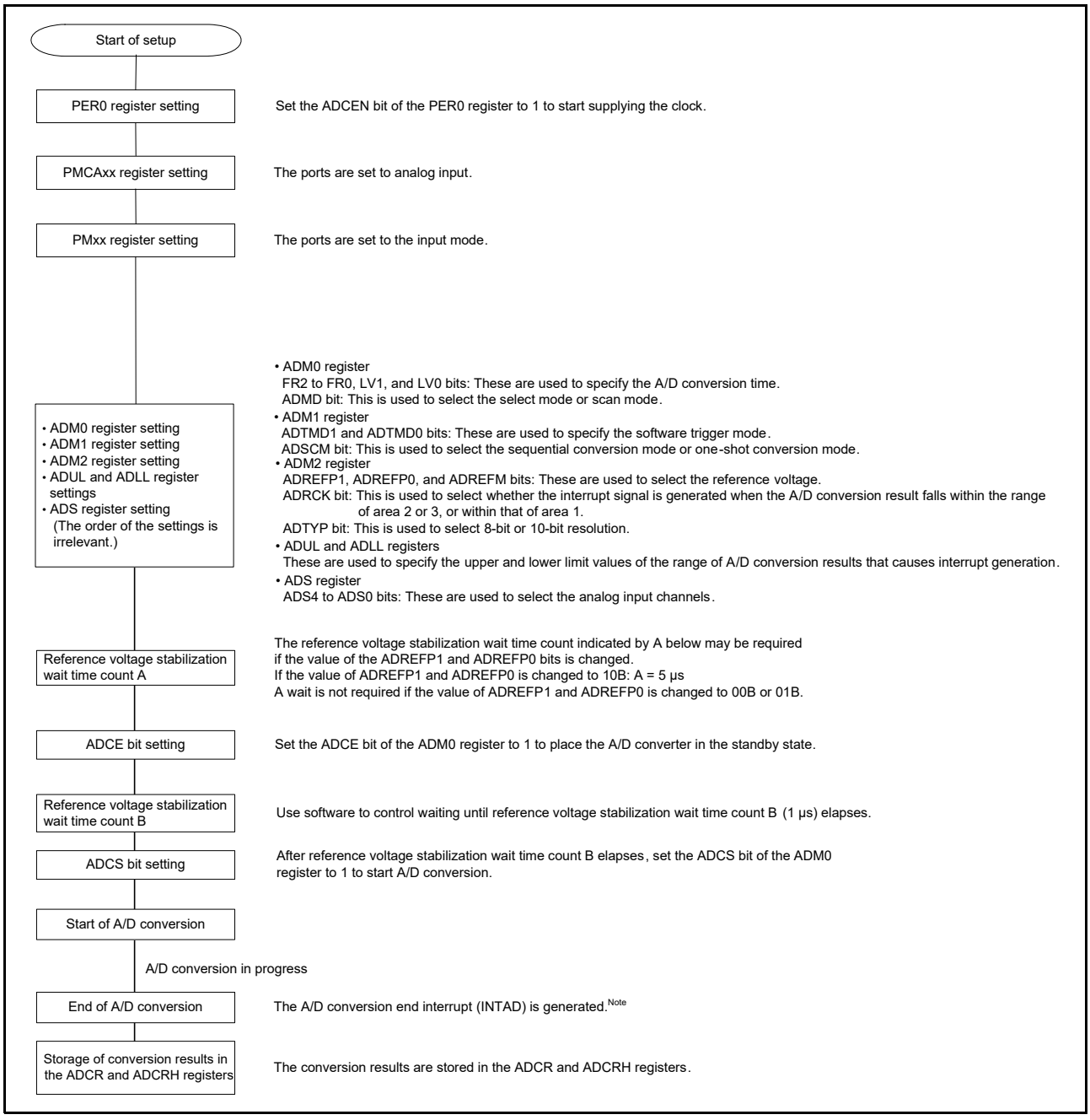


10.7 Flows of Setting the A/D Converter Modes

This section describes the flows of setting each of the A/D converter modes.

10.7.1 Settings in software trigger mode

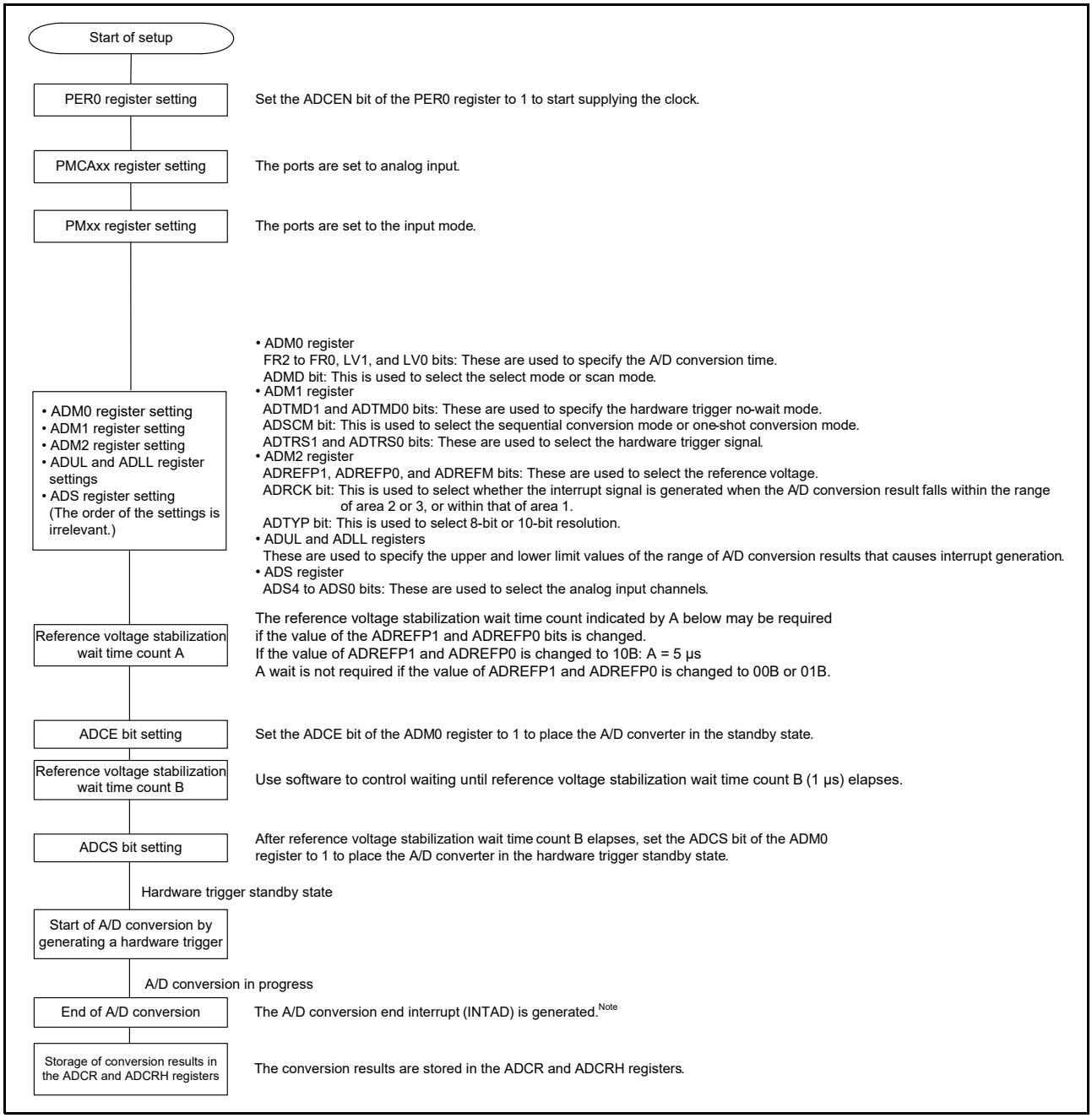
Figure 10 - 30 Settings in Software Trigger Mode



Note Depending on the settings of the ADRCK bit and ADUL and ADLL registers, the interrupt signal may not be generated. In such cases, the results of A/D conversion are not stored in the ADCR and ADCRH registers.

10.7.2 Settings in hardware trigger no-wait mode

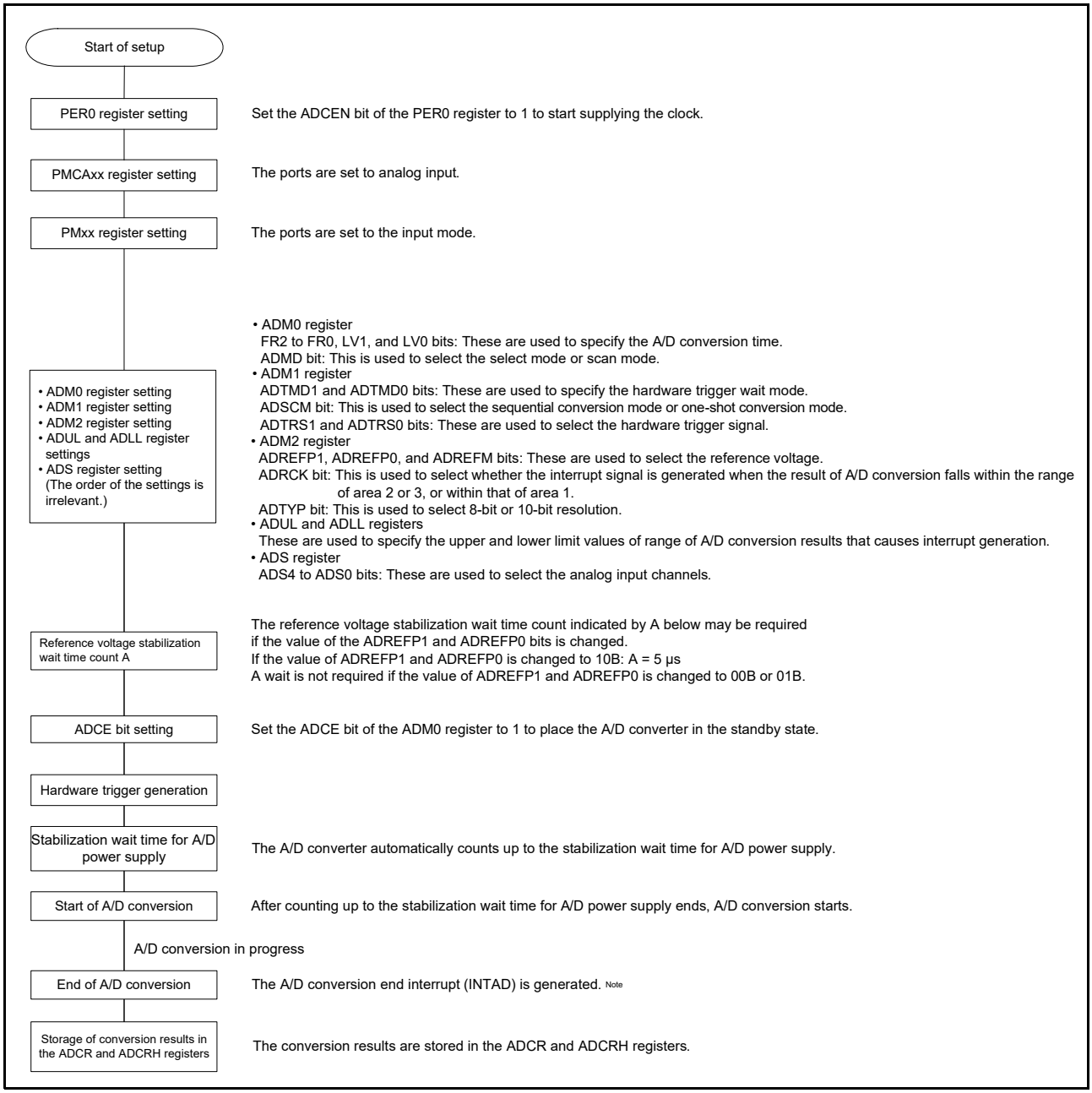
Figure 10 - 31 Settings in Hardware Trigger No-wait Mode



Note Depending on the settings of the ADRCK bit and ADUL and ADLL registers, the interrupt signal may not be generated. In such cases, the results of A/D conversion are not stored in the ADCR and ADCRH registers.

10.7.3 Settings in hardware trigger wait mode

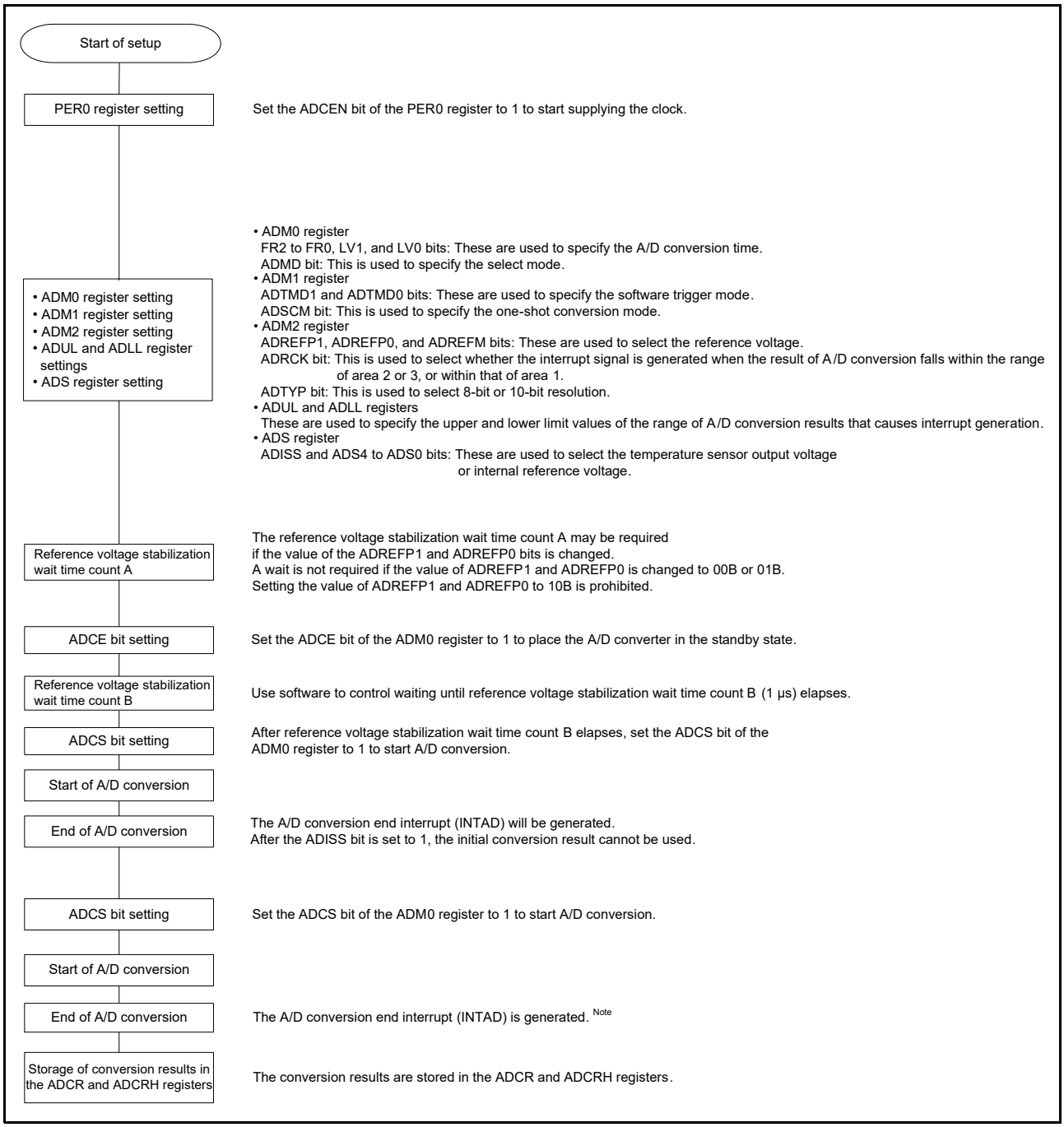
Figure 10 - 32 Settings in Hardware Trigger Wait Mode



Note Depending on the settings of the ADRCK bit and ADUL and ADLL registers, the interrupt signal may not be generated. In such cases, the results of A/D conversion are not stored in the ADCR and ADCRH registers.

10.7.4 Example of settings when temperature sensor output voltage or internal reference voltage is selected in the software trigger mode and one-shot conversion mode

Figure 10 - 33 Settings When Temperature Sensor Output Voltage or Internal Reference Voltage is Selected

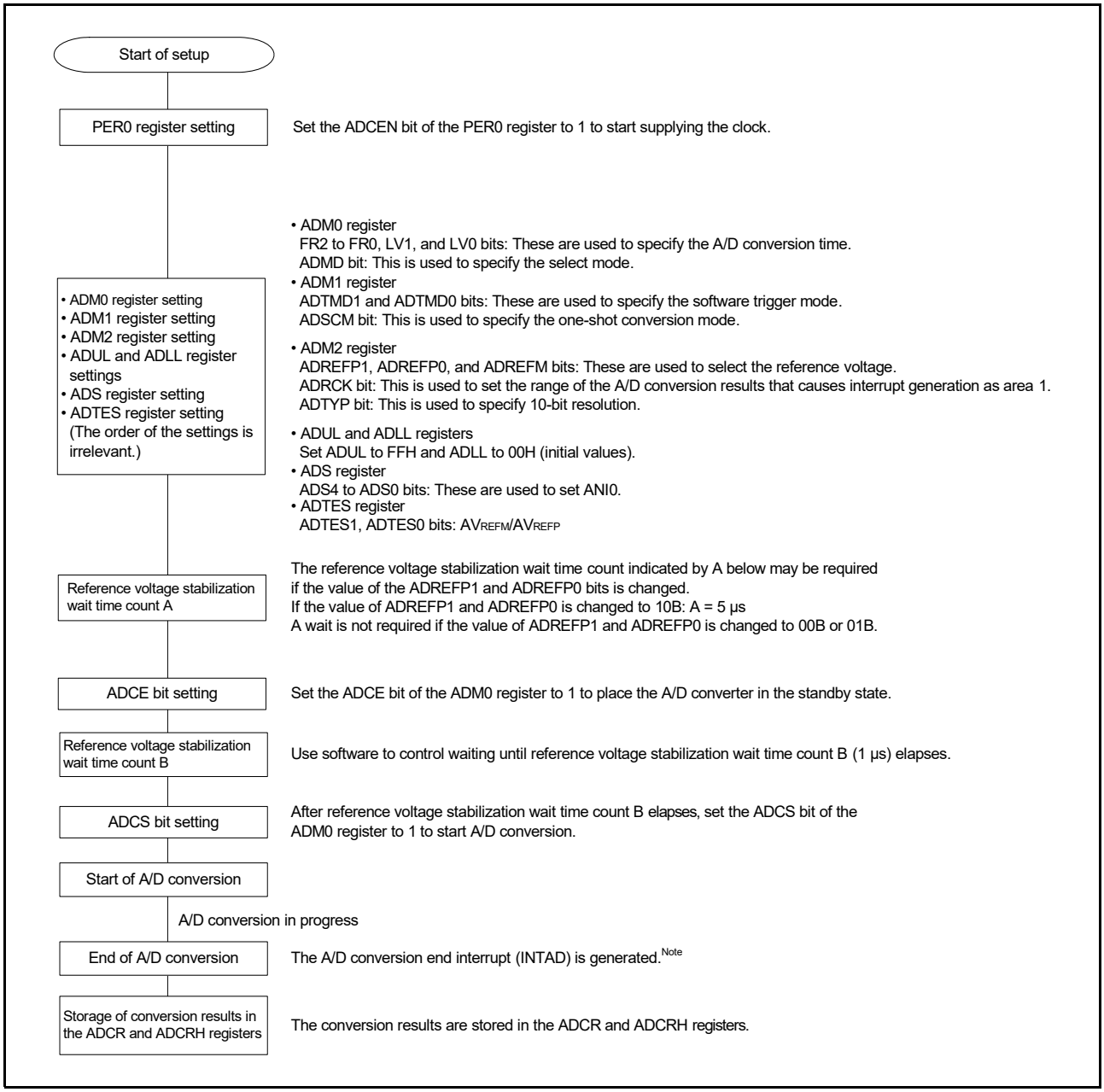


Note Depending on the settings of the ADRCK bit and ADUL and ADLL registers, the interrupt signal may not be generated. In such cases, the results of A/D conversion are not stored in the ADCR and ADCRH registers.

10.7.5 Settings in test mode

Figure 10 - 34 Settings in Test Mode

<R>



Note Depending on the settings of the ADRCK bit and ADUL and ADLL registers, the interrupt signal may not be generated. In such cases, the results of A/D conversion are not stored in the ADCR and ADCRH registers.

Caution For the procedure for testing the A/D converter, see 20.3.10 Testing of the A/D converter.

10.8 SNOOZE Mode Function

In the SNOOZE mode, A/D conversion is triggered by inputting a hardware trigger in the STOP mode. Normally, A/D conversion is stopped while in the STOP mode, but, by using the SNOOZE mode function, A/D conversion can be performed without operating the CPU. This is effective for reducing the operating current.

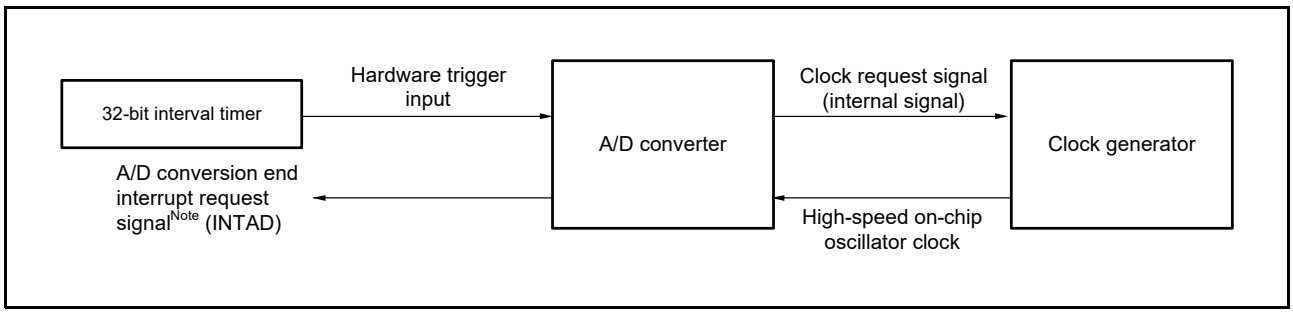
In the SNOOZE mode, the hardware trigger wait mode can only be used, with the following conversion modes:

- Select mode with one-shot conversion
- Scan mode with one-shot conversion

If the A/D conversion result range is specified using the ADUL and ADLL registers, A/D conversion results can be judged at a certain interval of time. Using this function enables power supply voltage monitoring and input key judgment based on A/D inputs.

Caution The SNOOZE mode can only be specified when the high-speed on-chip oscillator clock is selected for f_{CLK}.

Figure 10 - 35 Block Diagram When Using SNOOZE Mode Function



When using the SNOOZE mode function, the initial setting of each register is specified before switching to the STOP mode. For details about these settings, see **Figure 10 - 38 Flow of Setting up the SNOOZE Mode**. Just before move to STOP mode, the AWC bit of A/D converter mode register 2 (ADM2) is set to 1. After the initial settings are specified, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1. If a hardware trigger is input after switching to the STOP mode, the high-speed on-chip oscillator clock is supplied to the A/D converter. After supplying this clock, the A/D converter automatically counts up to the A/D power supply stabilization wait time, and then A/D conversion starts. The SNOOZE mode operation after A/D conversion is complete differs depending on whether an interrupt signal is generated**Note**.

Note Depending on the setting of the A/D conversion result comparison function (ADM2.ADRCK bit, ADUL and ADLL registers), the interrupt signal may not be generated.

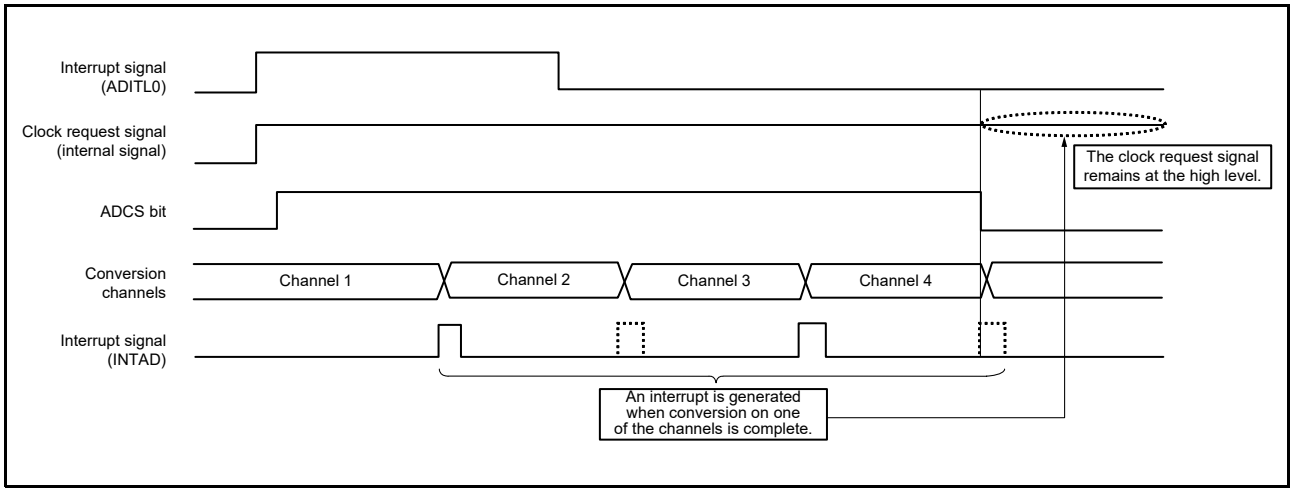
Caution Select the trigger signal (ADITL0) in response to a compare match of the 32-bit interval timer as the hardware trigger signal.

1. If an interrupt is generated after A/D conversion is complete
   If the A/D conversion result value is inside the range of values specified by the A/D conversion result comparison function (which is set up by using the ADRCK bit and ADUL and ADLL registers), the A/D conversion end interrupt request signal (INTAD) is generated.

   - While in the select mode
     When A/D conversion is complete and an A/D conversion end interrupt request signal (INTAD) is generated, the A/D converter returns to normal operation mode from SNOOZE mode. At this time, be sure to set the AWC bit of A/D converter mode register 2 (ADM2) to 0. If the AWC bit is left set to 1, A/D conversion will not start normally in the subsequent SNOOZE or normal operation mode.

   - While in the scan mode
     If even one value of the A/D conversion results of the four channels falls within the range specified by the A/D conversion result comparison function, and A/D conversion end interrupt request signal (INTAD) is generated, the A/D converter switches from the SNOOZE mode to the normal operation mode. At this time, be sure to set the AWC bit of A/D converter mode register 2 (ADM2) to 0. If the AWC bit is left set to 1, A/D conversion will not start normally in the subsequent SNOOZE or normal operation mode.

Figure 10 - 36 Example of Operation When Interrupt Is Generated Once A/D Conversion Is Complete in the Scan Mode



2. If no interrupt is generated after A/D conversion is complete
   If the A/D conversion result value is outside the range of values specified by the A/D conversion result comparison function (which is set up by using the ADRCK bit and ADUL and ADLL registers), the A/D conversion end interrupt request signal (INTAD) is not generated.

   - While in the select mode
     If the A/D conversion end interrupt request signal (INTAD) is not generated after A/D conversion is complete, the clock request signal (an internal signal) is automatically set to the low level, and supplying the high-speed on-chip oscillator clock stops. If a hardware trigger is input later, A/D conversion work is again performed in the SNOOZE mode.

   - While in the scan mode
     If the A/D conversion result values of the four channels do not fall within the range specified by the A/D conversion result comparison function even once, and the A/D conversion end interrupt request signal (INTAD) is not generated, the clock request signal (an internal signal) is automatically set to the low level after A/D conversion of the four channels is complete, and supplying the high-speed on-chip oscillator clock stops. If a hardware trigger is input later, A/D conversion work is again performed in the SNOOZE mode.

Figure 10 - 37 Example of Operation When No Interrupt Is Generated Once A/D Conversion Is Complete in the Scan Mode

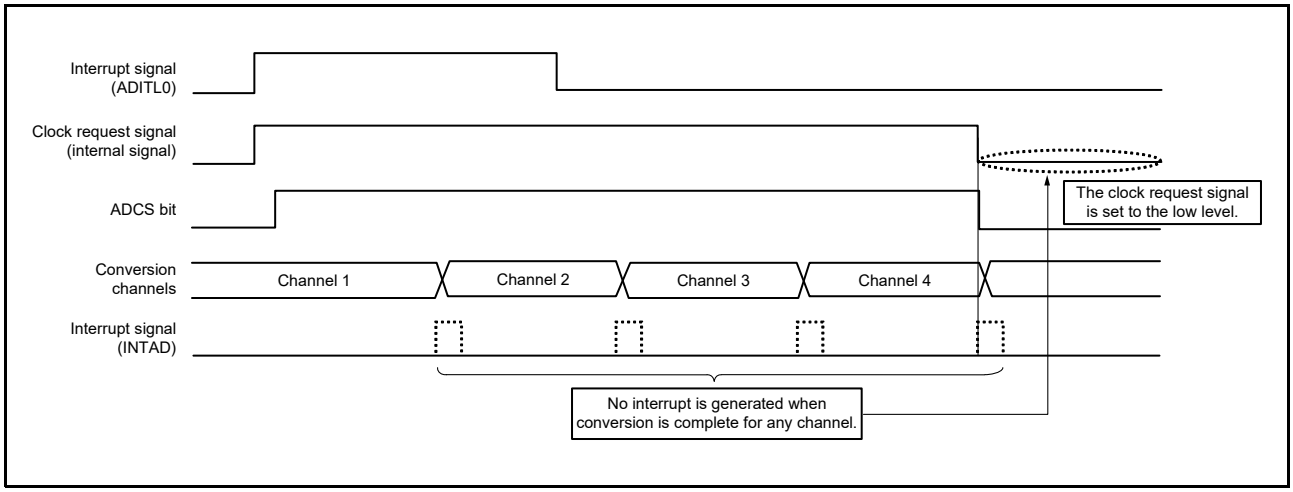
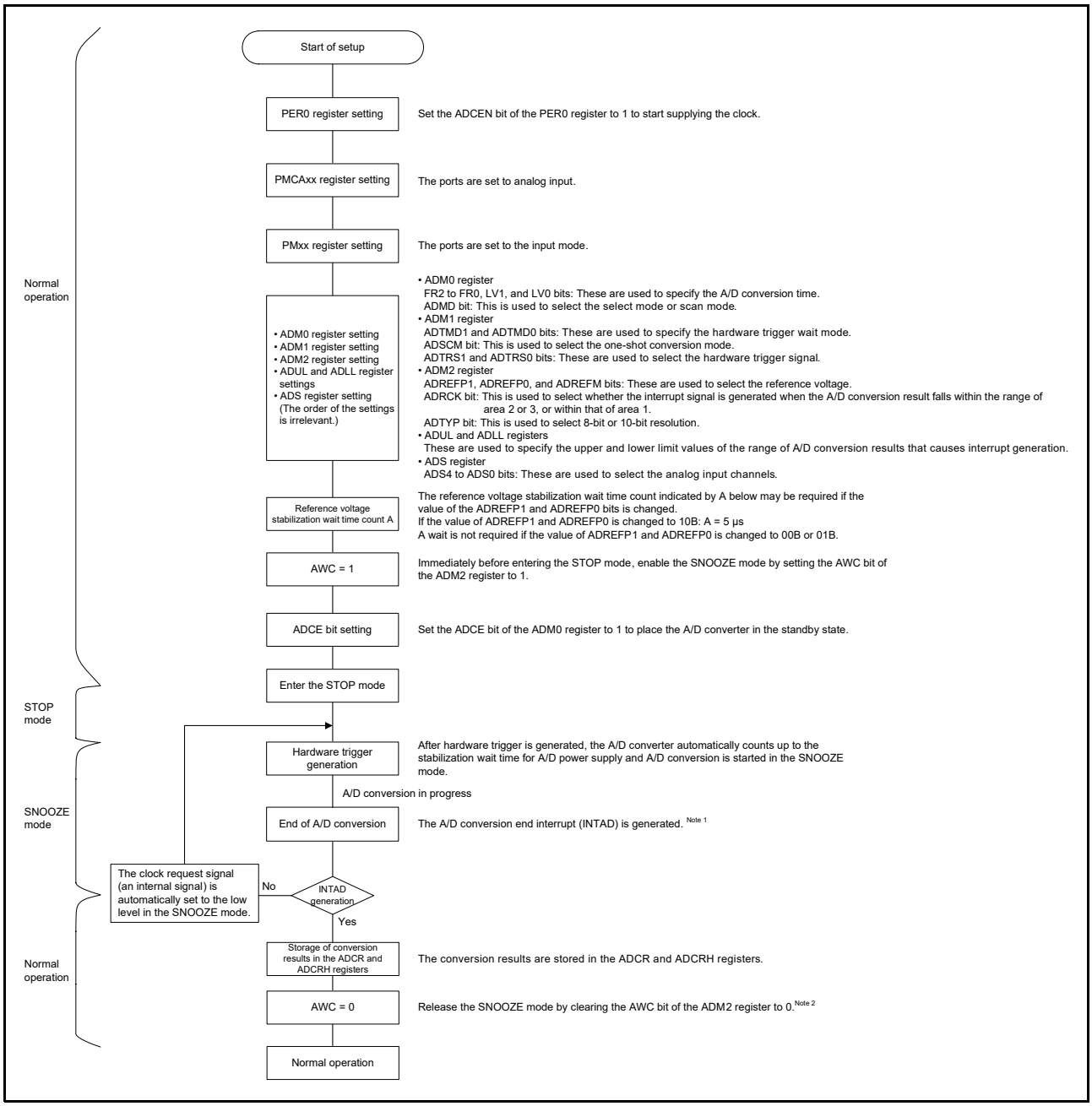


Figure 10 - 38 Flow of Setting up the SNOOZE Mode

<R>



Note 1. If the A/D conversion end interrupt request signal (INTAD) is not generated by setting ADRCK bit and ADUL and ADLL registers, the result is not stored in the ADCR and ADCRH registers. The A/D converter enters the STOP mode again. If a hardware trigger is input later, A/D conversion operation is again performed in the SNOOZE mode.

Note 2. If the AWC bit is left set to 1, A/D conversion will not start normally in spite of the subsequent SNOOZE or normal operation mode. Be sure to set the AWC bit to 0.

10.9 How to Read A/D Converter Characteristics Table

Here, special terms unique to the A/D converter are explained.

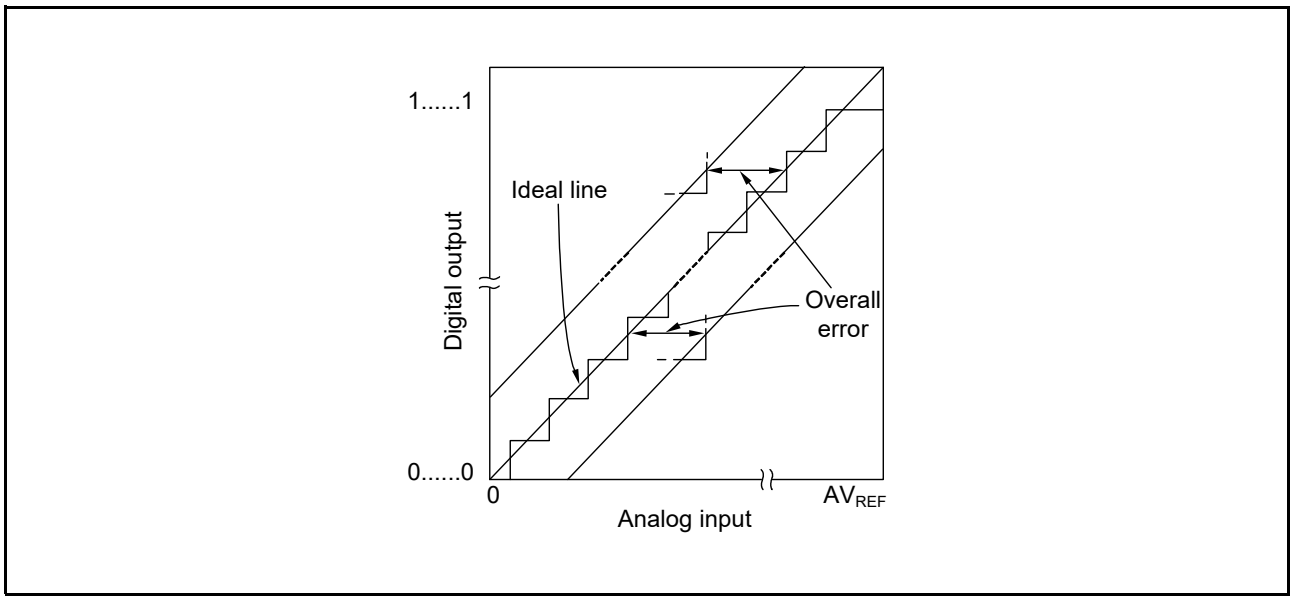
1. Resolution
   This is the minimum analog input voltage that can be identified. That is, the percentage of the analog input voltage per bit of digital output is called 1 LSB (Least Significant Bit). The percentage of 1 LSB with respect to the full scale is expressed by %FSR (Full Scale Range).

   1 LSB is as follows when the resolution is 10 bits.

   $$1\ LSB = 1/2^{10} = 1/1024$$
   $$\approx 0.098\%FSR$$

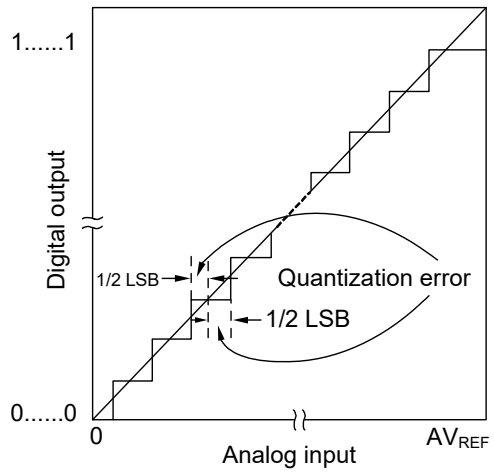
   Accuracy has no relation to resolution, but is determined by overall error.
2. Overall error
   This shows the maximum error value between the actual measured value and the theoretical value. Zero-scale error, full-scale error, integral linearity error, and differential linearity error, and combinations of these express the overall error. Note that the quantization error is not included in the overall error in the characteristics table.

Figure 10 - 39 Overall Error



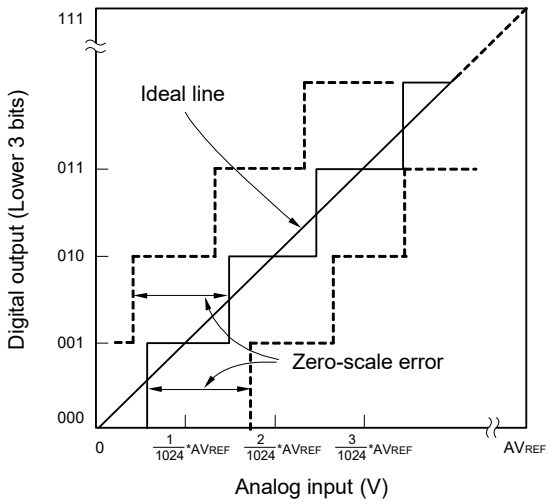
3. Quantization error
   When analog values are converted to digital values, a ±1/2 LSB error naturally occurs. In an A/D converter, an analog input voltage in a range of ±1/2 LSB is converted to the same digital code, so a quantization error cannot be avoided. Note that the quantization error is not included in the overall error, zero-scale error, full-scale error, integral linearity error, and differential linearity error in the characteristics table.

Figure 10 - 40 Quantization Error



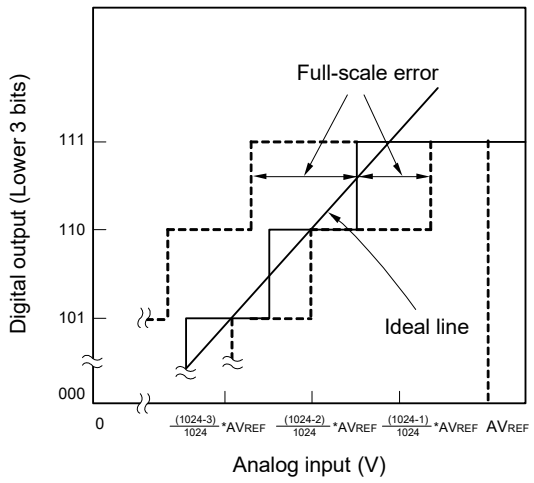
4. Zero-scale error
   This shows the difference between the actual measurement value of the analog input voltage and the theoretical value (1/2 LSB) when the digital output changes from 0.....000 to 0.....001. If the actual measurement value is greater than the theoretical value, it shows the difference between the actual measurement value of the analog input voltage and the theoretical value (3/2 LSB) when the digital output changes from 0……001 to 0……010.

Figure 10 - 41 Zero-scale Error



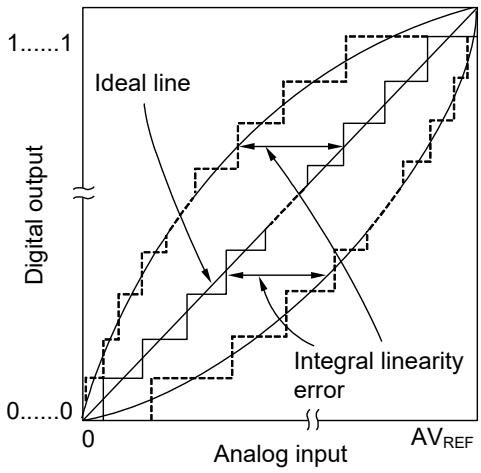
5. Full-scale error
   This shows the difference between the actual measurement value of the analog input voltage and the theoretical value (Full-scale - 3/2 LSB) when the digital output changes from 1.....110 to 1.....111.

Figure 10 - 42 Full-scale Error



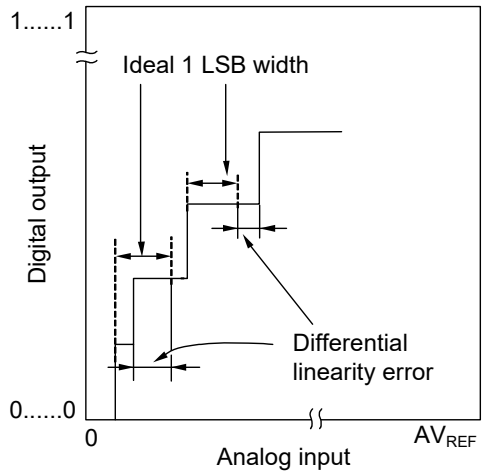
6. Integral linearity error
   This shows the degree to which the conversion characteristics deviate from the ideal linear relationship. It expresses the maximum value of the difference between the actual measurement value and the ideal straight line when the zero-scale error and full-scale error are 0.

Figure 10 - 43 Integral Linearity Error

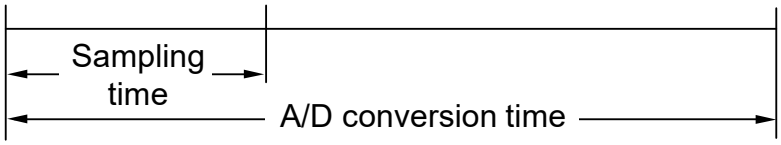


7. Differential linearity error
   While the ideal width of code output is 1 LSB, this indicates the difference between the actual measurement value and the ideal value.

Figure 10 - 44 Differential Linearity Error



8. Conversion time
   This expresses the time from the start of sampling to when the digital output is obtained. The sampling time is included in the conversion time in the characteristics table.
9. Sampling time
   This is the time the analog switch is turned on for the analog voltage to be sampled by the sample & hold circuit.



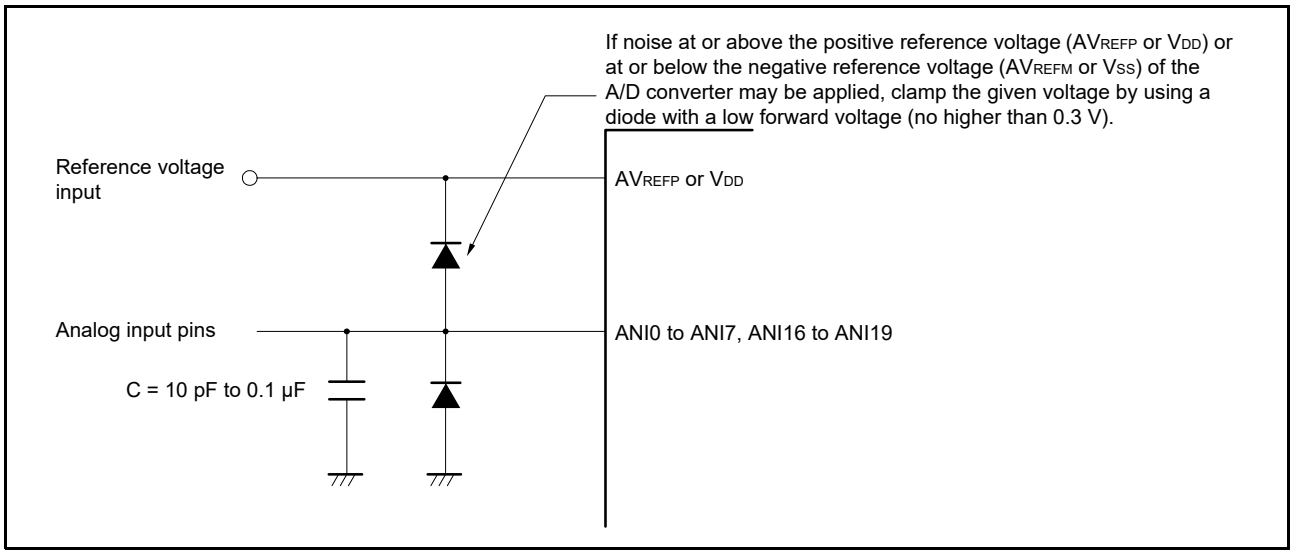
10.10 Points for Caution When the A/D Converter Is to be Used

1. Operating current in STOP mode
   Shift to STOP mode after stopping the A/D converter (by setting the ADCS bit of A/D converter mode register 0 (ADM0) to 0). The operating current can be reduced by setting the ADCE bit of the ADM0 register to 0 at the same time. To restart from the standby state, clear the ADIF bit of interrupt request flag register 1H (IF1H) to 0 and start operation.
2. Input range of ANI0 to ANI7 and ANI16 to ANI19 pins
   Observe the rated range of the ANI0 to ANI7 and ANI16 to ANI19 pins input voltage. If a voltage exceeding V_{DD} and AV_{REFP} or a voltage lower than V_{SS} and AV_{REFM} (even in the range of absolute maximum ratings) is input to an analog input channel, the converted value of that channel becomes undefined. In addition, the converted values of the other channels may also be affected. When internal reference voltage is selected as the positive reference voltage of the A/D converter, do not input a voltage equal to or higher than the internal reference voltage to a pin selected by the ADS register. However, it is no problem that a voltage equal to or higher than the internal reference voltage is input to a pin not selected by the ADS register.

   Caution For details about the internal reference voltage, see Section 28 Electrical Characteristics.

3. Conflicting operations
   <1> Conflict between the write access to the A/D conversion result register (ADCR or ADCRH) upon the end of conversion and the read access to the ADCR or ADCRH register with an instruction
   Reading from the ADCR or ADCRH register has the higher priority. After the read operation, the new conversion result is written to the ADCR or ADCRH register.
   <2> Conflict between the write access to the ADCR or ADCRH register for storage of the result of A/D conversion upon the end of conversion, the write access to A/D converter mode register 0 (ADM0) with an instruction, and the write access to the analog input channel specification register (ADS)
   Writing to the ADM0 and ADS registers has the higher priority. Writing to the ADCR and ADCRH registers is not performed, nor is the conversion end interrupt signal (INTAD) generated.
4. Noise countermeasures
   To maintain the 10-bit resolution, attention must be paid to noise input to the AV_{REFP}, V_{DD}, ANI0 to ANI7, and ANI16 to ANI19 pins.
   <1> Connect a capacitor with a low equivalent resistance and a good frequency response (capacitance of about 0.01 μF) via the shortest possible run of relatively thick wiring to the V_{DD} and AV_{REFP} pins.
   <2> The higher the output impedance of the analog input source, the greater the influence. To reduce the noise, connecting an external capacitor as shown in **Figure 10 - 45** is recommended.
   <3> Do not switch these pins with other pins during conversion.
   <4> The accuracy is improved if the HALT mode is set immediately after the start of conversion.

Figure 10 - 45 Connections of V_{DD}, AV_{REFP}, and Analog Input Pins



5. Analog input (ANIxx) pins

<1> The analog input pins (ANI0 to ANI7 and ANI16 to ANI19) are also used as input port pins (P02, P03, P20 to P27, P120, and P147). Do not change the output values for the port-pin functions P02, P03, P20 to P27, P120, and P147 while A/D conversion of the signals on the ANI0 to ANI7 or ANI16 to ANI19 pins is selected and conversion is in progress, since doing so may lower the precision of the results of conversion.

<2> If a pin adjacent to a pin that is being A/D converted is used as a digital I/O port pin, the A/D conversion result might differ from the expected value due to a coupling noise. Be sure to prevent such a pulse from being input or output. Be sure to avoid the input or output of digital signals and signals with similarly sharp transitions during conversion.

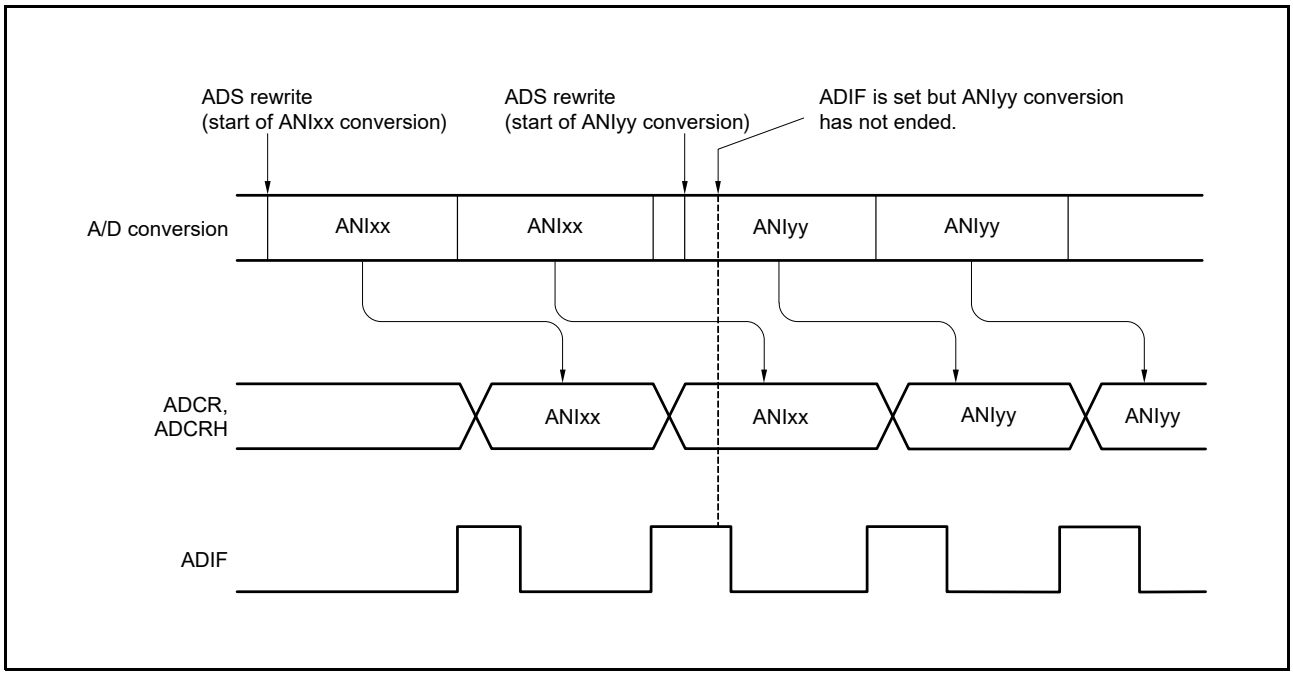
6. Input impedance of analog input (ANIxx) pins

This A/D converter charges a sampling capacitor for sampling during sampling time. Therefore, only a leakage current flows when sampling is not in progress, and a current that charges the capacitor flows during sampling. Consequently, the input impedance fluctuates depending on whether sampling is in progress, and on the other states. To make sure that sampling is effective, however, we recommend using the converter with analog input sources that have output impedances no greater than 1 kΩ. If a source has a higher output impedance, lengthen the sampling time or connect a larger capacitor (with a value of about 0.1 µF) to the pin from among ANI0 to ANI7 and ANI16 to ANI19 to which the source is connected (see **Figure 10 - 45**). The sampling capacitor may be being charged while the setting of the ADCS bit is 0 and immediately after sampling is restarted and so is not defined at these times. Accordingly, the state of conversion is undefined after charging starts in the next round of conversion

<R> after the value of the ADCS bit has been 0 or when conversion is repeated. Thus, to secure full charging regardless of the size of fluctuations in the analog signal, ensure that the output impedances of the sources of analog inputs are low or secure sufficient time for the completion of conversion.

7. Interrupt request flag (ADIF)
   The interrupt request flag (ADIF) is not cleared to 0 even if the analog input channel specification register (ADS) is changed. Therefore, if an analog input pin is changed during A/D conversion, the A/D conversion result and ADIF flag for the pre-change analog input may be set just before the ADS register rewrite. Caution is therefore required since, at this time, when ADIF flag is read immediately after the ADS register rewrite, ADIF flag is set despite the fact A/D conversion for the post-change analog input has not ended. When A/D conversion is stopped and then resumed, clear ADIF flag to 0 before the A/D conversion operation is resumed.

Figure 10 - 46 Timing of A/D Conversion End Interrupt Request Generation



8. Conversion results just after A/D conversion start
   While in the software trigger mode or hardware trigger no-wait mode, the first A/D conversion value immediately after A/D conversion starts may not fall within the rating range if the ADCS bit is set to 1 within 1 μs after the ADCE bit was set to 1. Take measures such as polling the A/D conversion end interrupt request signal (INTAD) and removing the first conversion result.
9. A/D conversion result register (ADCR, ADCRH) read operation
   When a write operation is performed to A/D converter mode register 0 (ADM0), analog input channel specification register (ADS), or port mode control A register (PMCAxx), the contents of the ADCR and ADCRH registers may become undefined. Read the conversion result following conversion completion before writing to the ADM0, ADS, or PMCAxx register. Using a timing other than the above may cause an incorrect conversion result to be read.

10. Internal equivalent circuit

The equivalent circuit of the analog input block is shown below.

Figure 10 - 47 Internal Equivalent Circuit of ANIxx Pin

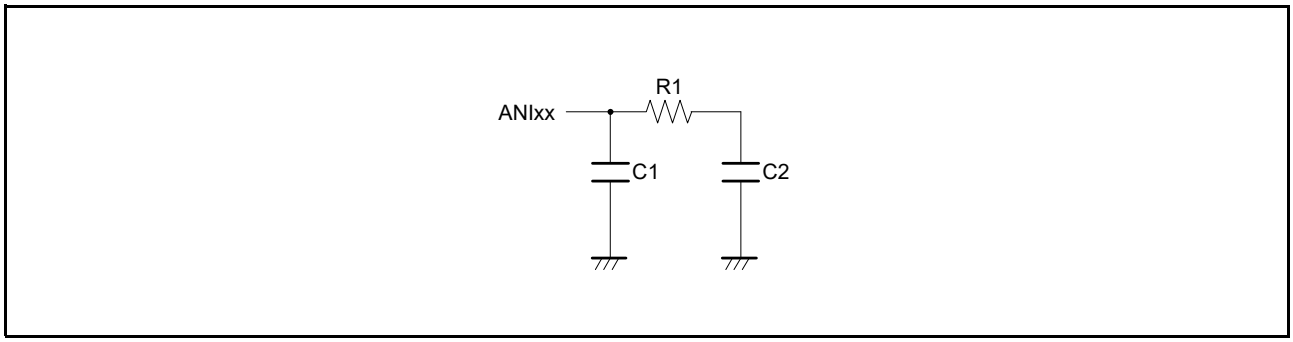


Table 10 - 4 Resistance and Capacitance Values of Equivalent Circuit (Reference Values)

AV_{REFP}, V_{DD}	ANIxx Pins	R1 [kΩ]	C1 [pF]	C2 [pF]
3.6 V ≤ V_{DD} ≤ 5.5 V	ANI0 to ANI7	14	8	2.5
	ANI16 to ANI19	18	8	7
2.7 V ≤ V_{DD} ≤ 3.6 V	ANI0 to ANI7	39	8	2.5
	ANI16 to ANI19	53	8	7
1.8 V ≤ V_{DD} ≤ 2.7 V	ANI0 to ANI7	231	8	2.5
	ANI16 to ANI19	321	8	7
1.6 V ≤ V_{DD} < 2.7 V	ANI0 to ANI7	632	8	2.5
	ANI16 to ANI19	902	8	7

Remark The resistance and capacitance values are not guaranteed values.

11. Starting the A/D converter

Start the A/D converter after the AV_{REFP} and V_{DD} voltages stabilize.

Section 11 Serial Array Unit (SAU)

Each product of this group has two serial array units. Units 0 and 1 respectively has up to four and two serial channels. Each channel can achieve 3-wire serial (simplified SPI or CSI[Note]), UART, and simplified I2C communication. The following tables show the assignment of functions to each channel in a product of this group.

Note Although the CSI function is generally called SPI, it is also called CSI in this product, so it is referred to as such in this manual.

20-pin products

Unit	Channel	Used as Simplified SPI (CSI)	Used as UART	Used as Simplified I2C
0	0	CSI00	UART0	IIC00
	1	—		—
	2	—	UART1	—
	3	CSI11		IIC11
1	0	—	UART2 (supporting the LIN bus)	—
	1	—		—

32-pin products

Unit	Channel	Used as Simplified SPI (CSI)	Used as UART	Used as Simplified I2C
0	0	CSI00	UART0	IIC00
	1	—		—
	2	—	UART1	—
	3	CSI11		IIC11
1	0	CSI20	UART2 (supporting the LIN bus)	IIC20
	1	—		—

44-pin products

Unit	Channel	Used as Simplified SPI (CSI)	Used as UART	Used as Simplified I2C
0	0	CSI00	UART0	IIC00
	1	—		—
	2	—	UART1	—
	3	CSI11		IIC11
1	0	CSI20	UART2 (supporting the LIN bus)	IIC20
	1	CSI21		IIC21

48-pin products

Unit	Channel	Used as Simplified SPI (CSI)	Used as UART	Used as Simplified I²C
0	0	CSI00	UART0	IIC00
	1	CSI01		IIC01
	2	—	UART1	—
	3	CSI11		IIC11
1	0	CSI20	UART2 (supporting the LIN bus)	IIC20
	1	CSI21		IIC21

64-pin products

Unit	Channel	Used as Simplified SPI (CSI)	Used as UART	Used as Simplified I²C
0	0	CSI00	UART0	IIC00
	1	CSI01		IIC01
	2	CSI10	UART1	IIC10
	3	CSI11		IIC11
1	0	CSI20	UART2 (supporting the LIN bus)	IIC20
	1	CSI21		IIC21

When UART0 is used for channels 0 and 1 of the unit 0, CSI00 and CSI01 cannot be used, but CSI10, UART1, or IIC10 in channels 1 and 3 can be used.

Caution Most of the following descriptions in this section use the units and channels of the 64-pin products as an example.

11.1 Functions of Serial Array Unit

Each serial interface supported by a product of this group has the following features.

11.1.1 Simplified SPI (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21)

Data is transmitted or received in synchronization with the serial clock (SCK) output from the master. 3-wire serial communication is clocked communication performed by using three communication lines: one for the serial clock (SCK), one for transmitting serial data (SO), one for receiving serial data (SI). For details about the settings, see **11.5 Operation of Simplified SPI (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) Communication**.

[Data transmission/reception]

- Data length of 7 or 8 bits
- Phase control of transmit/receive data
- MSB/LSB first selectable

[Clock control]

- Master/slave selection
- Phase control of I/O clock
- Setting of transfer period by prescaler and internal counter of each channel
- Maximum transfer rate**Note** During master communication: Max. f_{CLK}/2 (CSI00 only)
  Max. f_{CLK}/4
  During slave communication: Max. f_{MCK}/6

[Interrupt function]

- Transfer end interrupt/buffer empty interrupt

[Error detection flag]

- Overrun error

In addition, the simplified SPI (CSI00) supports the SNOOZE mode. In the SNOOZE mode, data can be received without CPU processing upon detecting SCK input in the STOP mode.

Note Set up the transfer rate within a range satisfying the SCK cycle time (t_{KCY}). For details, see **Section 28 Electrical Characteristics**.

Caution Use a general-purpose port pin to send a chip select signal when required.

11.1.2 UART (UART0 to UART2)

This is a start-stop synchronization communication function using two lines: serial data transmission (TxD) and serial data reception (RxD) lines. By using these two communication lines, each data frame, which consist of a start bit, data, parity bit, and stop bit, is transferred asynchronously (using the internal baud rate) between the microcontroller and the other communication party. Full-duplex UART communication can be performed by using a channel dedicated to transmission (even-numbered channel) and a channel dedicated to reception (odd-numbered channel). The LIN bus can be implemented by using timer array unit with an external interrupt (INTP0). For details about the settings, see **11.6 Operation of UART (UART0 to UART2) Communication**.

[Data transmission/reception]

- Data length of 7, 8, or 9 bits **Note**
- MSB/LSB first selectable
- Level setting of transmit/receive data and select of reverse
- Parity bit appending and parity check functions
- Stop bit appending

[Interrupt function]

- Transfer end interrupt/buffer empty interrupt
- Error interrupt in case of framing error, parity error, or overrun error

[Error detection flag]

- Framing error, parity error, or overrun error

In addition, the UART0 reception supports the SNOOZE mode. In the SNOOZE mode, data can be received without CPU processing upon detecting RxD input in the STOP mode.

UART2 (channels 0 and 1 of unit 1) supports the LIN bus.

[LIN-bus functions]

- Wakeup signal detection
- Break field (BF) detection
- Sync field measurement, baud rate calculation

} Using the external interrupt (INTP0) and timer array unit

Note Only UART0 supports the 9-bit data length.

11.1.3 Simplified I2C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21)

This is a clocked communication function to communicate with two or more devices by using two lines: serial clock (SCL) and serial data (SDA). This simplified I2C is designed for single communication with a device such as EEPROM, flash memory, or A/D converter, and therefore, it functions only as a master. Make sure by using software, as well as operating the control registers, that the AC specifications of the start and stop conditions are observed. For details about the settings, see **11.8 Operation of Simplified I2C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) Communication**.

[Data transmission/reception]

- Master transmission, master reception (only master function with a single master)
- ACK output function **Note** and ACK detection function
- Data length of 8 bits (When an address is transmitted, the address is specified by the higher 7 bits, and the least significant bit is used for R/W control.)
- Manual generation of start condition and stop condition

[Interrupt function]

- Transfer end interrupt

[Error detection flag]

- ACK error or overrun error

* [Functions not supported by simplified I2C]

- Slave transmission, slave reception
- Arbitration loss detection function
- Clock stretch detection

Note When receiving the last data, ACK will not be output if 0 is written to the SOEmn bit (serial output enable register m (SOEm)) and serial communication data output is stopped. See **11.8.3 Data reception**, **2. Processing flow** for details.

Remark 1. To use an I2C bus of full function, see **Section 12 Serial Interface IICA (IICA)**.

Remark 2. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

11.2 Configuration of Serial Array Unit

The serial array unit includes the following registers, and input and output pins.

Table 11 - 1 Configuration of Serial Array Unit

Item	Configuration
Shift register	8 or 9 bits[Note 1]
Buffer registers	Lower 8 bits or 9 bits of serial data register mn (SDRmn)[Notes 1, 2]
Serial clock I/O pins	SCK00, SCK01, SCK10, SCK11, SCK20, and SCK21 (for simplified SPI), SCL00, SCL01, SCL10, SCL11, SCL20, and SCL21 (for simplified I^2C)
Serial data input pins	SI00, SI01, SI10, SI11, SI20, and SI21 (for simplified SPI), RxD0, and RxD1 (for UART), RxD2 pin (for UART supporting the LIN bus)
Serial data output pins	SO00, SO01, SO10, SO11, SO20, and SO21 (for simplified SPI), TxD0 and TxD1 (for UART), TxD2 (for UART supporting the LIN bus)
Serial data I/O pins	SDA00, SDA01, SDA10, SDA11, SDA20, and SDA21 (for simplified I^2C)
Control registers	<Registers of unit setting block> • Peripheral enable register 0 (PER0) • Peripheral reset control register 0 (PRR0) • Serial clock select register m (SPSm) • Serial channel enable status register m (SEm) • Serial channel start register m (SSm) • Serial channel stop register m (STm) • Serial output enable register m (SOEm) • Serial output register m (SOm) • Serial output level register m (SOLm) • Serial standby control register m (SSCm) • Input switch control register (ISC) • Noise filter enable register 0 (NFEN0)
	<Registers of each channel> • Serial data register mn (SDRmn) • Serial mode register mn (SMRmn) • Serial communication operation setting register mn (SCRmn) • Serial status register mn (SSRmn) • Serial flag clear trigger register mn (SIRmn)
	• Port input mode registers (PIMxx) • Port output mode registers (POMxx) • Port mode control A registers (PMCAxx) • Port mode registers (PMxx) • Port registers (Pxx) • UART loopback select register (ULBS)

Note 1. The number of bits used as the shift register and buffer register differs depending on the unit and channel.
- mn = 00, 01: Lower 9 bits
- Other than above: Lower 8 bits

Note 2. The lower 8 bits of serial data register mn (SDRmn) can be read or written as the following SFR, depending on the communication mode.
- CSIp communication: SIOp (CSIp data register)
- UARTq reception: RxDq (UARTq receive data register)
- UARTq transmission: TxDq (UARTq transmit data register)
- IICr communication: SIOr (IICr data register)

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21)
q: UART number (q = 0 to 2), r: IIC number (r = 00, 01, 10, 11, 20, 21)

Figure 11 - 1 shows the block diagram of serial array unit 0.

Figure 11 - 1 Block Diagram of Serial Array Unit 0

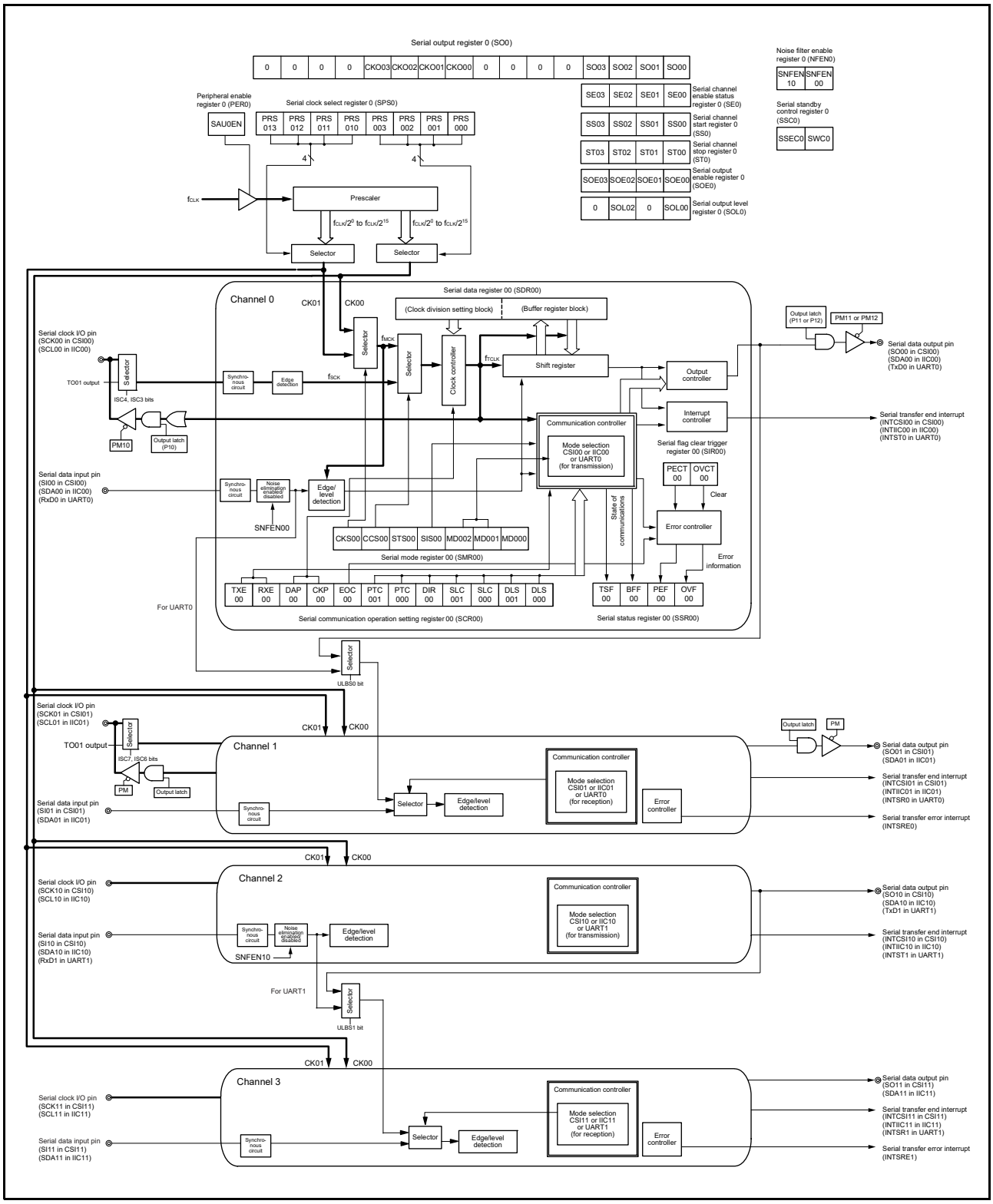
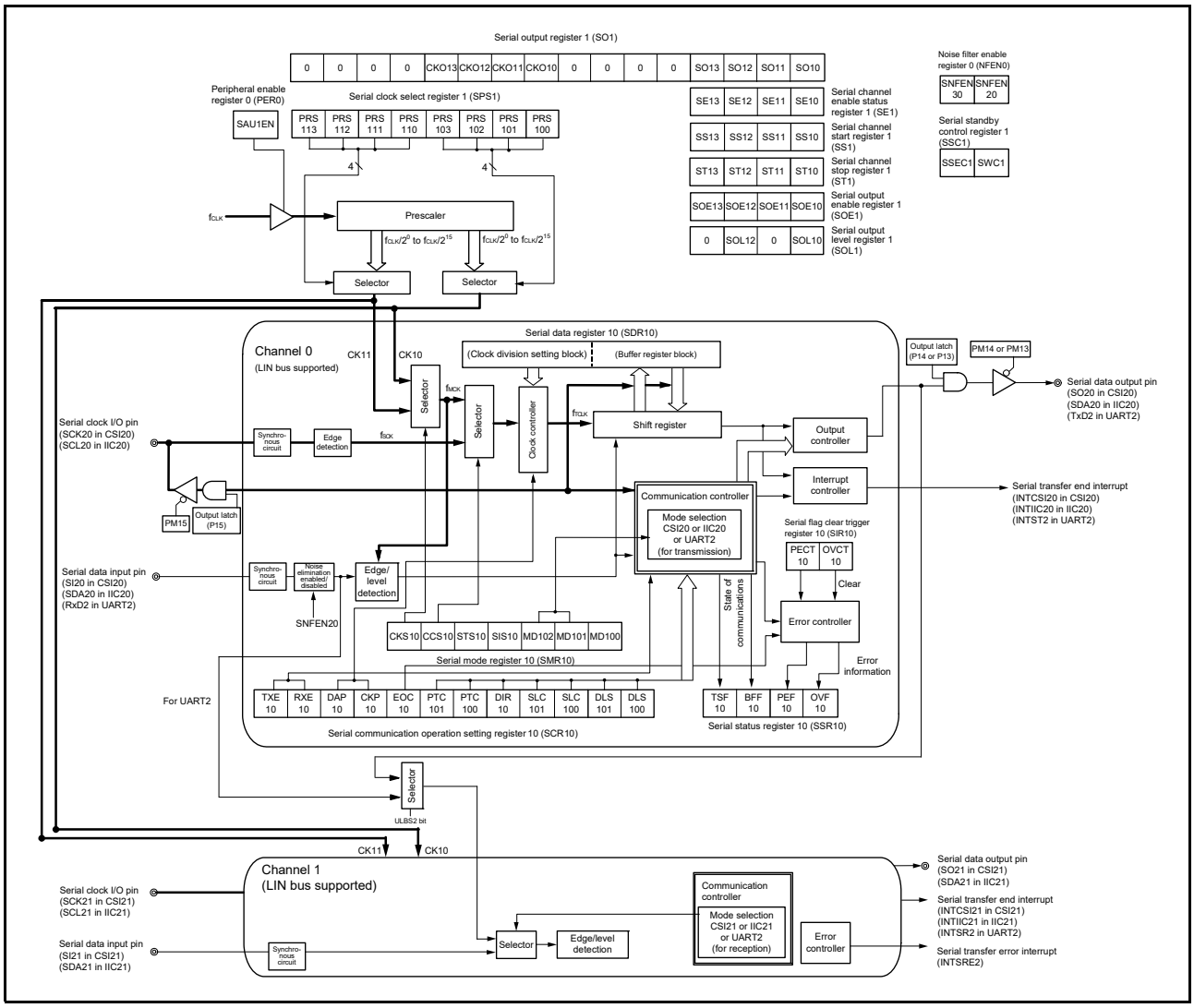


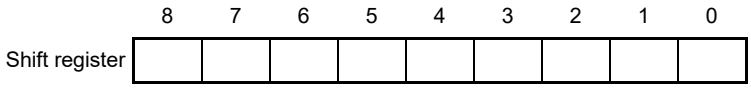
Figure 11 - 2 shows the block diagram of serial array unit 1.

Figure 11 - 2 Block Diagram of Serial Array Unit 1



11.2.1 Shift register

This is a 9-bit register that converts parallel data into serial data or vice versa. In case of the UART communication of nine bits of data, nine bits (bits 8 to 0) are used[Note]. During reception, it converts data input to the serial pin into parallel data. When data is transmitted, the value set to this register is output as serial data from the serial output pin. The shift register cannot be directly manipulated by program. To read or write the shift register, use the lower 8 or 9 bits of serial data register mn (SDRmn).



Note Only UART0 supports the 9-bit data length.

11.2.2 Lower 8 or 9 bits of the serial data register mn (SDRmn)

The SDRmn is the transmit/receive data register (16 bits) of channel n. Bits 8 to 0 (lower 9 bits)[Note 1] or bits 7 to 0 (lower 8 bits) function as a transmit/receive buffer register, and bits 15 to 9 are used as a register that sets the division ratio of the operation clock (f_{MCK}). When data is received, parallel data converted by the shift register is stored in the lower 8 or 9 bits. When data is to be transmitted, set transmit data to be transferred to the shift register to the lower 8 or 9 bits. The data stored in the lower 8 or 9 bits of this register is as follows, depending on the setting of bits 0 and 1 (DLSmn0, DLSmn1) of serial communication operation setting register mn (SCRmn), regardless of the output sequence of the data.

- 7-bit data length (stored in bits 0 to 6 of the SDRmn register)
- 8-bit data length (stored in bits 0 to 7 of the SDRmn register)
- 9-bit data length (stored in bits 0 to 8 of the SDRmn register)[Note 1]

The SDRmn register can be read or written in 16-bit units. The lower 8 or 9 bits of the SDRmn register can be read or written[Note 2] as the following SFR, depending on the communication mode.

- CSIp communication: SIOp (CSIp data register)
- UARTq reception: RXDq (UARTq receive data register)
- UARTq transmission: TXDq (UARTq transmit data register)
- IICr communication: SIOr (IICr data register)

The value of each SDRmn register following a reset is 0000H.

Note 1. Only UART0 supports the 9-bit data length.

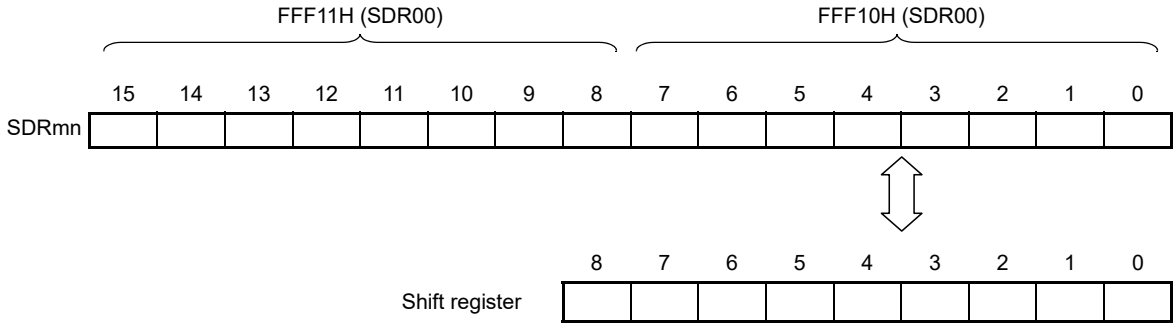
Note 2. When operation is stopped (SEmn = 0), do not rewrite the SDRmn[7:0] bits by an 8-bit memory manipulation instruction (SDRmn[15:9] are all cleared to 0).

Remark 1. After data is received, 0 is stored in bits 0 to 8 in bit portions that exceed the data length.

Remark 2. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21)
q: UART number (q = 0 to 2), r: IIC number (r = 00, 01, 10, 11, 20, 21)

Figure 11 - 3 Format of Serial Data Register mn (SDRmn) (mn = 00, 01)

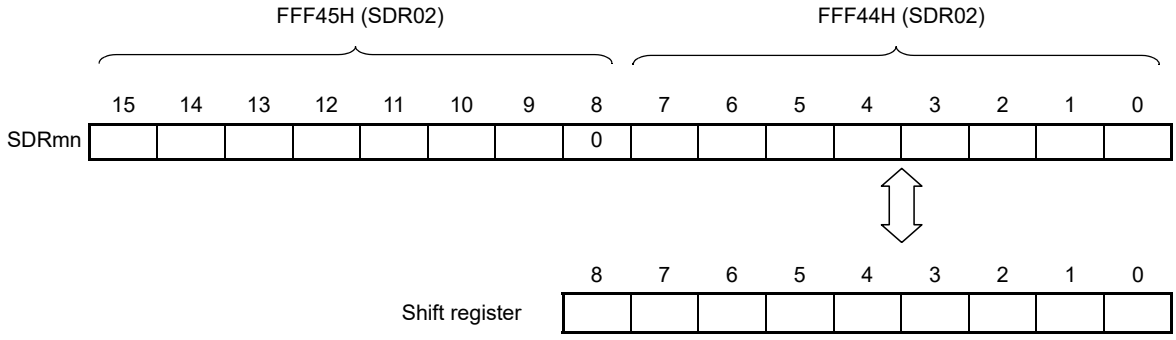
Address: FFF10H, FFF11H (SDR00), FFF12H, FFF13H (SDR01)
After reset: 0000H
R/W: R/W



Remark For the function of the higher 7 bits of the SDRmn register, see **11.3 Registers for Controlling the Serial Array Unit**.

Figure 11 - 4 Format of Serial Data Register mn (SDRmn) (mn = 02, 03, 10, 11)

Address: FFF44H, FFF45H (SDR02), FFF46H, FFF47H (SDR03),
FFF48H, FFF49H (SDR10), FFF4AH, FFF4BH (SDR11)
After reset: 0000H
R/W: R/W



Caution **Be sure to set bit 8 to 0.**

Remark For the function of the higher 7 bits of the SDRmn register, see **11.3 Registers for Controlling the Serial Array Unit**.

11.3 Registers for Controlling the Serial Array Unit

The following registers are used to control the serial array unit.

- Peripheral enable register 0 (PER0)
- Peripheral reset control register 0 (PRR0)
- Serial clock select register m (SPSm)
- Serial mode register mn (SMRmn)
- Serial communication operation setting register mn (SCRmn)
- Serial data register mn (SDRmn)
- Serial flag clear trigger register mn (SIRmn)
- Serial status register mn (SSRmn)
- Serial channel start register m (SSm)
- Serial channel stop register m (STm)
- Serial channel enable status register m (SEm)
- Serial output enable register m (SOEm)
- Serial output register m (SOm)
- Serial output level register m (SOLm)
- Serial standby control register m (SSCm)
- Input switch control register (ISC)
- Noise filter enable register 0 (NFEN0)
- Port mode registers (PMxx)
- Port registers (Pxx)
- Port input mode registers (PIMxx)
- Port output mode registers (POMxx)
- Port mode control A registers (PMCAxx)
- UART loopback select register (ULBS)

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

11.3.1 Peripheral enable register 0 (PER0)

The PER0 register is used to enable or disable the supply of a clock signal to various on-chip peripheral modules. Clock supply to an on-chip peripheral module that is not to be used can be stopped to decrease power consumption and noise. If serial array unit 0 is to be used, be sure to set the SAU0EN bit of this register to 1. If serial array unit 1 is to be used, be sure to set the SAU1EN bit of this register to 1. The PER0 register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of the PER0 register following a reset is 00H.

Figure 11 - 5 Format of Peripheral Enable Register 0 (PER0)

Address: F00F0H
After reset: 00H
R/W: R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
PER0	0	0	ADCEN	IICA0EN Note 1	SAU1EN	SAU0EN	0	TAU0EN

SAUmEN	Control of supply of an input clock to serial array unit m
0	Stops supply of an input clock. • The SFRs used by serial array unit m cannot be written.Note 2
1	Enables supply of an input clock. • The SFRs used by serial array unit m can be read and written.

Note 1. This bit is only present in the 32- to 64-pin products.

Note 2. When an SFR used by serial array unit m is read, the value returned is 00H or 0000H regardless of the setting value.

Caution 1. Be sure to set bits 7, 6, and 1 to 0.

Caution 2. Be sure to set bit 4 in the 20-pin products to 0.

Caution 3. When setting serial array unit m, start by setting the following registers while the setting of the SAUmEN bit is 1.

- **Serial clock select register m (SPSm)**
- **Serial mode register mn (SMRmn)**
- **Serial communication operation setting register mn (SCRmn)**
- **Serial data register mn (SDRmn)**
- **Serial flag clear trigger register mn (SIRmn)**
- **Serial status register mn (SSRmn)**
- **Serial channel start register m (SSm)**
- **Serial channel stop register m (STm)**
- **Serial channel enable status register m (SEm)**
- **Serial output enable register m (SOEm)**
- **Serial output level register m (SOLm)**
- **Serial output register m (SOm)**
- **Serial standby control register m (SSCm)**

If the setting of the SAUmEN bit is 0, writing to any of the registers which control serial array unit m is ignored. Note, however, writing to the following registers is valid.

- **Input switch control register (ISC)**
- **Noise filter enable register 0 (NFEN0)**
- **Port input mode registers (PIM0, PIM1, PIM5, PIM7)**
- **Port output mode registers (POM0, POM1, POM5, POM7)**
- **Port mode control A register (PMCA0)**
- **Port mode registers (PM0, PM1, PM3, PM5, PM7)**
- **Port registers (P0, P1, P3, P5, P7)**

11.3.2 Peripheral reset control register 0 (PRR0)

The PRR0 register is used to control resetting of the on-chip peripheral modules. Each bit in this register controls resetting and release from the reset state of the corresponding on-chip peripheral module. Set the SAU0RES and SAU1RES bits to 1 to reset serial array units 0 and 1, respectively. The PRR0 register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of the PRR0 register following a reset is 00H.

Figure 11 - 6 Format of Peripheral Reset Control Register 0 (PRR0)

Address: F00F1H
After reset: 00H
R/W: R/W

Symbol	7	<6>	<5>	<4>	<3>	<2>	<1>	<0>
PRR0	0	0	ADCRES	IICA0RES **Note**	SAU1RES	SAU0RES	TAU1RES	TAU0RES

SAUmRES	Control resetting of serial array unit m
0	Serial array unit m is released from the reset state.
1	Serial array unit m is in the reset state. • The SFRs for use with serial array unit m are initialized.

Note This bit is only present in the 32- to 64-pin products.

Caution 1. Be sure to set bits 7, 6, and 1 to 0.
Caution 2. Be sure to set bit 4 in the 20-pin products to 0.

11.3.3 Serial clock select register m (SPSm)

The SPSm is a 16-bit register that is used to select two types of operation clocks (CKm0, CKm1) that are commonly supplied to each channel. CKm1 is selected by bits 7 to 4 of the SPSm register, and CKm0 is selected by bits 3 to 0. Rewriting the SPSm register is prohibited when the register is in operation (when SEmn = 1). The SPSm register can be set by a 16-bit memory manipulation instruction. The 8 lower-order bits of an SPSm register can be handled as the SPSmL register, which can be set by an 8-bit memory manipulation instruction. The value of each SPSm register following a reset is 0000H.

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

Figure 11 - 7 Format of Serial Clock Select Register m (SPSm)

Address: F0126H, F0127H (SPS0), F0166H, F0167H (SPS1)**Note**
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
SPSm	0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
PRSm13	PRSm12	PRSm11	PRSm10	PRSm03	PRSm02	PRSm01	PRSm00

PRS mk3	PRS mk2	PRS mk1	PRS mk0	Selection of operation clock (CKmk)**Note**				
					f_{CLK} = 2 MHz	f_{CLK} = 5 MHz	f_{CLK} = 10 MHz	f_{CLK} = 20 MHz	f_{CLK} = 32 MHz
0	0	0	0	f_{CLK}	2 MHz	5 MHz	10 MHz	20 MHz	32 MHz
0	0	0	1	$f_{CLK}/2$	1 MHz	2.5 MHz	5 MHz	10 MHz	16 MHz
0	0	1	0	$f_{CLK}/2^2$	500 kHz	1.25 MHz	2.5 MHz	5 MHz	8 MHz
0	0	1	1	$f_{CLK}/2^3$	250 kHz	625 kHz	1.25 MHz	2.5 MHz	4 MHz
0	1	0	0	$f_{CLK}/2^4$	125 kHz	313 kHz	625 kHz	1.25 MHz	2 MHz
0	1	0	1	$f_{CLK}/2^5$	62.5 kHz	156 kHz	313 kHz	625 kHz	1 MHz
0	1	1	0	$f_{CLK}/2^6$	31.3 kHz	78.1 kHz	156 kHz	313 kHz	500 kHz
0	1	1	1	$f_{CLK}/2^7$	15.6 kHz	39.1 kHz	78.1 kHz	156 kHz	250 kHz
1	0	0	0	$f_{CLK}/2^8$	7.81 kHz	19.5 kHz	39.1 kHz	78.1 kHz	125 kHz
1	0	0	1	$f_{CLK}/2^9$	3.91 kHz	9.77 kHz	19.5 kHz	39.1 kHz	62.5 kHz
1	0	1	0	$f_{CLK}/2^{10}$	1.95 kHz	4.88 kHz	9.77 kHz	19.5 kHz	31.3 kHz
1	0	1	1	$f_{CLK}/2^{11}$	977 Hz	2.44 kHz	4.88 kHz	9.77 kHz	15.6 kHz
1	1	0	0	$f_{CLK}/2^{12}$	488 Hz	1.22 kHz	2.44 kHz	4.88 kHz	7.81 kHz
1	1	0	1	$f_{CLK}/2^{13}$	244 Hz	610 Hz	1.22 kHz	2.44 kHz	3.91 kHz
1	1	1	0	$f_{CLK}/2^{14}$	122 Hz	305 Hz	610 Hz	1.22 kHz	1.95 kHz
1	1	1	1	$f_{CLK}/2^{15}$	61 Hz	153 Hz	305 Hz	610 Hz	977 Hz

Note When changing the clock selected for f_{CLK} (by changing the system clock control register (CKC) value), do so after having stopped (serial channel stop register m (STm) = 000FH) the operation of the serial array unit (SAU).

Caution Be sure to set bits 15 to 8 to 0.

Remark 1. f_{CLK}: CPU/peripheral hardware clock frequency

Remark 2. m: Unit number (m = 0, 1)

Remark 3. k = 0, 1

11.3.4 Serial mode register mn (SMRmn)

The SMRmn register is used to set an operation mode of channel n. It is also used to select an operation clock (f_{MCK}), specify whether the serial clock (f_{SCK}) may be input or not, set a start trigger, the operating mode (as simplified SPI or CSI, UART, or simplified I^2C), and an interrupt source. This register is also used to invert the level of the receive data only in the UART mode. Rewriting the SMRmn register is prohibited when the register is in operation (when SEmn = 1). However, the MDmn0 bit can be rewritten during operation. The SMRmn register can be set by a 16-bit memory manipulation instruction. The value of each SMRmn register following a reset is 0020H.

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

Figure 11 - 8 Format of Serial Mode Register mn (SMRmn) (1/2)

Address: F0110H, F0111H (SMR00) to F0116H, F0117H (SMR03),
F0150H, F0151H (SMR10), F0152H, F0153H (SMR11)

After reset: 0020H

R/W: R/W

Symbol	15	14	13	12	11	10	9	8
SMRmn	CKSmn	CCSmn	0	0	0	0	0	STSmn **Note**

7	6	5	4	3	2	1	0
0	SISmn0 **Note**	1	0	0	MDmn2	MDmn1	MDmn0

CKSmn	Selection of operation clock (f_{MCK}) of channel n
0	Operation clock CKm0 set by the SPSm register
1	Operation clock CKm1 set by the SPSm register
Operation clock (f_{MCK}) is used by the edge detector. In addition, depending on the setting of the CCSmn bit and the higher 7 bits of the SDRmn register, a transfer clock (f_{TCLK}) is generated.	

CCSmn	Selection of transfer clock (f_{TCLK}) of channel n
0	Divided operation clock f_{MCK} specified by the CKSmn bit
1	Clock input f_{SCK} from the SCKp pin (slave transfer in simplified SPI or CSI mode)
Transfer clock f_{TCLK} is used for the shift register, communication controller, output controller, interrupt controller, and error controller. When CCSmn = 0, the division ratio of operation clock (f_{MCK}) is set by the higher 7 bits of the SDRmn register.	

STSmn	Selection of start trigger source
0	Only software trigger is valid (selected for simplified SPI or CSI, UART transmission, and simplified I^2C).
1	Valid edge of the RxDq pin (selected for UART reception)
Transfer is started when the above source is satisfied after 1 is set to the SSm register.	

Figure 11 - 8 Format of Serial Mode Register mn (SMRmn) (2/2)

SISmn0	Controls inversion of level of receive data of channel n in UART mode
0	Falling edge is detected as the start bit. The input communication data is captured as is.
1	Rising edge is detected as the start bit. The input communication data is inverted and captured.

MDmn2	MDmn1	Setting of operation mode of channel n
0	0	Simplified SPI (CSI) mode
0	1	UART mode
1	0	Simplified I^2C mode
1	1	Setting prohibited

MDmn0	Selection of interrupt source of channel n
0	Transfer end interrupt
1	Buffer empty interrupt (Occurs when data is transferred from the SDRmn register to the shift register.)
For continuous transmission, set this bit to 1 and write the next transmit data when SDRmn data has run out.	

Note This bit is only present in the SMR01, SMR03, and SMR11 registers.

Caution **Be sure to set bits 13 to 9, 7, 4, and 3 (or bits 13 to 6, 4, and 3 for the SMR00, SMR02, or SMR10 register) to 0. Be sure to set bit 5 to 1.**

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), q: UART number (q = 0 to 2), r: IIC number (r = 00, 01, 10, 11, 20, 21)

11.3.5 Serial communication operation setting register mn (SCRmn)

The SCRmn is a communication operation setting register of channel n. It is used to set a data transmission/reception mode, phase of data and clock, whether an error signal is to be masked or not, parity bit, start bit, stop bit, and data length. Rewriting the SCRmn register is prohibited when the register is in operation (when SEmn = 1). The SCRmn register can be set by a 16-bit memory manipulation instruction. The value of each SCRmn register following a reset is 0087H.

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

Figure 11 - 9 Format of Serial Communication Operation Setting Register mn (SCRmn) (1/2)

Address: F0118H, F0119H (SCR00) to F011EH, F011FH (SCR03),
F0158H, F0159H (SCR10), F015AH, F015BH (SCR11)

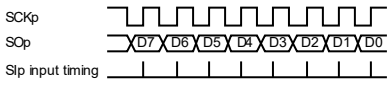
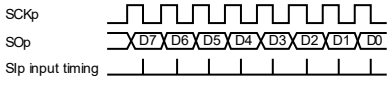
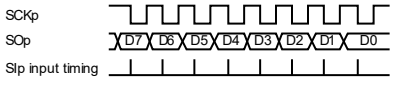
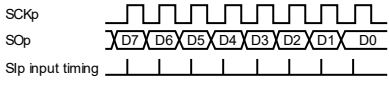
After reset: 0087H

R/W: R/W

Symbol	15	14	13	12	11	10	9	8
SCRmn	TXEmn	RXEmn	DAPmn	CKPmn	0	EOCmn	PTCmn1	PTCmn0

7	6	5	4	3	2	1	0
DIRmn	0	SLCmn1 **Note 1**	SLCmn0	0	1	DLSmn1 **Note 2**	DLSmn0

TXEmn	RXEmn	Setting of operation mode of channel n
0	0	Disable communication.
0	1	Reception only
1	0	Transmission only
1	1	Transmission/reception

DAPmn	CKPmn	Selection of data and clock phase in simplified SPI (CSI) mode	Type
0	0		1
0	1		2
1	0		3
1	1		4
Set DAPmn and CKPmn to 00B in the UART mode and simplified I^2C mode.			

EOCmn	Mask control of error interrupt signal (INTSREx (x = 0 to 3))
0	Disables generation of error interrupt INTSREx (INTSRx is generated).
1	Enables generation of error interrupt INTSREx (INTSRx is not generated if an error occurs).
Set EOCmn to 0 in the simplified SPI (CSI) mode, simplified I^2C mode, and during UART transmission **Note 3**.	

Figure 11 - 9 Format of Serial Communication Operation Setting Register mn (SCRmn) (2/2)

PTCmn1	PTCmn0	Setting of parity bit in UART mode	
		Transmission	Reception
0	0	Does not output the parity bit.	Receives without parity
0	1	Outputs 0 parity**Note 4**.	No parity judgment
1	0	Outputs even parity.	Judged as even parity.
1	1	Outputs odd parity.	Judges as odd parity.
Set PTCmn1 and PTCmn to 00B in the simplified SPI (CSI) mode and simplified I^2C mode.			

DIRmn	Selection of data transfer sequence in simplified SPI (CSI) and UART modes
0	Inputs/outputs data with MSB first.
1	Inputs/outputs data with LSB first.
Set DIRmn to 0 in the simplified I^2C mode.	

SLCmn1	SLCmn0	Setting of stop bit in UART mode
0	0	No stop bit
0	1	Stop bit length = 1 bit
1	0	Stop bit length = 2 bits (mn = 00, 02, 10 only)
1	1	Setting prohibited
When the transfer end interrupt is selected, the interrupt is generated when all stop bits have been completely transferred. Set 1 bit (SLCmn1, SLCmn0 = 01B) during UART reception and in the simplified I^2C mode. Set no stop bit (SLCmn1, SLCmn0 = 00B) in the simplified SPI (CSI) mode. Set 1 bit (SLCmn1, SLCmn0 = 01B) or 2 bits (SLCmn1, SLCmn0 = 10B) during UART transmission.		

DLSmn1	DLSmn0	Setting of data length in simplified SPI (CSI) and UART modes
0	1	9-bit data length (stored in bits 0 to 8 of the SDRmn register) (settable in UART mode only)
1	0	7-bit data length (stored in bits 0 to 6 of the SDRmn register)
1	1	8-bit data length (stored in bits 0 to 7 of the SDRmn register)
Other than above		Setting prohibited
Set DLSmn1 and DLSmn0 to 11B in the simplified I^2C mode.		

Note 1. This bit is only present in the SCR00, SCR02, and SCR10 registers.

Note 2. This bit is only present in the SCR00 and SCR01 registers, and is fixed to 1 for the other registers.

Note 3. An INTSREx error interrupt might occur unless the setting of EOCmn in CSImn is 0.

Note 4. 0 is appended to the data value regardless of the contents.

Caution **Be sure to set bits 11, 6, and 3 to 0. Also set bit 5 of the SCR01, SCR03, or SCR11 register to 0. Be sure to set bit 2 to 1.**

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21)

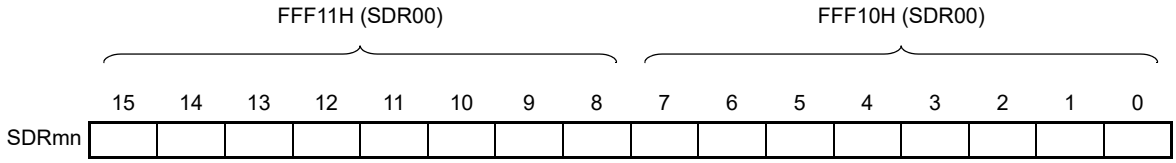
11.3.6 Serial data register mn (SDRmn)

The SDRmn is the transmit/receive data register (16 bits) of channel n. Bits 8 to 0 (lower 9 bits) of the SDR00 and SDR01 registers or bits 7 to 0 (lower 8 bits) of the SDR02, SDR03, SDR10, and SDR11 registers function as a transmit/receive buffer register, and bits 15 to 9 (higher 7 bits) are used as a register that sets the division ratio of the operation clock (f_{MCK}). If the CCSmn bit of serial mode register mn (SMRmn) is cleared to 0, the clock set by dividing the operation clock by bits 15 to 9 (higher 7 bits) of the SDRmn register is used as the transfer clock. If the CCSmn bit of serial mode register mn (SMRmn) is set to 1, set bits 15 to 9 (upper 7 bits) of the SDR00 and SDR01 registers to 0000000B. The input clock f_{SCK} (slave transfer in simplified SPI or CSI mode) from the SCKp pin is used as the transfer clock. The lower 8 or 9 bits of the SDRmn register function as a transmit/receive buffer register. During reception, the parallel data converted by the shift register is stored in the lower 8 or 9 bits, and during transmission, the data to be transmitted to the shift register is set to the lower 8 or 9 bits. The SDRmn register can be read or written in 16-bit units. However, the higher 7 bits can only be written or read when the operation is stopped (SEmn = 0). During operation (SEmn = 1), a value is written only to the lower 8 or 9 bits of the SDRmn register. When the SDRmn register is read during operation, the higher 7 bits are always read as 0. The value of each SDRmn register following a reset is 0000H.

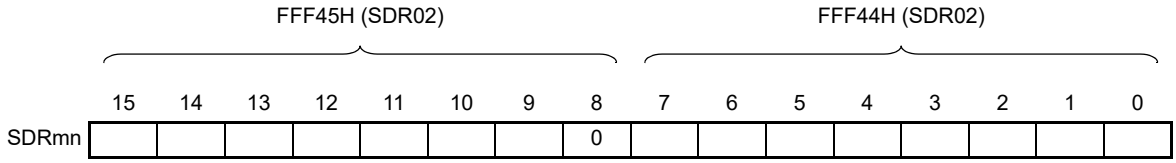
Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

Figure 11 - 10 Format of Serial Data Register mn (SDRmn)

Address: FFF10H, FFF11H (SDR00), FFF12H, FFF13H (SDR01)
After reset: 0000H
R/W: R/W



Address: FFF44H, FFF45H (SDR02), FFF46H, FFF47H (SDR03)
FFF48H, FFF49H (SDR10), FFF4AH, FFF4BH (SDR11)
After reset: 0000H
R/W: R/W



SDRmn[15:9]							Transfer clock setting by dividing the operation clock
0	0	0	0	0	0	0	f_{MCK}/2
0	0	0	0	0	0	1	f_{MCK}/4
0	0	0	0	0	1	0	f_{MCK}/6
0	0	0	0	0	1	1	f_{MCK}/8
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
1	1	1	1	1	1	0	f_{MCK}/254
1	1	1	1	1	1	1	f_{MCK}/256

Caution 1. Be sure to set bit 8 of the SDR02, SDR03, SDR10, and SDR11 registers to 0.

Caution 2. Setting SDRmn[15:9] to 0000000B or 0000001B is prohibited when UART is used.

Caution 3. Setting SDRmn[15:9] to 0000000B is prohibited when simplified I^2C is used. Set SDRmn[15:9] to 0000001B or greater.

Caution 4. When operation is stopped (SEmn = 0), do not rewrite the SDRmn[7:0] bits by an 8-bit memory manipulation instruction (the SDRmn[15:9] bits are all cleared to 0).

Remark 1. For the function of the lower 8 or 9 bits of the SDRmn register, see **11.2 Configuration of Serial Array Unit**.

Remark 2. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

11.3.7 Serial flag clear trigger register mn (SIRmn)

The SIRmn is a trigger register that is used to clear each error flag of channel n. When each bit (FECTmn, PECTmn, OVCTmn) of this register is set to 1, the corresponding bit (FEFmn, PEFmn, OVFmn) of serial status register mn is cleared to 0. Because the SIRmn is a trigger register, it is cleared immediately when the corresponding bit of the SSRmn register is cleared. The SIRmn register can be set by a 16-bit memory manipulation instruction. The 8 lower-order bits of an SIRmn register can be handled as the SIRmnL register, which can be set by an 8-bit memory manipulation instruction. The value of each SIRmn register following a reset is 0000H.

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

Figure 11 - 11 Format of Serial Flag Clear Trigger Register mn (SIRmn)

Address: F0108H, F0109H (SIR00) to F010EH, F010FH (SIR03),
F0148H, F0149H (SIR10), F014AH, F014BH (SIR11)

After reset: 0000H

R/W: R/W

Symbol	15	14	13	12	11	10	9	8
SIRmn	0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
0	0	0	0	0	FECTmn **Note**	PECTmn	OVCTmn

FECTmn	Clear trigger of framing error flag of channel n
0	Not cleared
1	Clears the FEFmn bit of the SSRmn register to 0.

PECTmn	Clear trigger of parity error flag of channel n
0	Not cleared
1	Clears the PEFmn bit of the SSRmn register to 0.

OVCTmn	Clear trigger of overrun error flag of channel n
0	Not cleared
1	Clears the OVFmn bit of the SSRmn register to 0.

Note This bit is only present in the SIR01, SIR03, and SIR11 registers.

Caution Be sure to set bits 15 to 3 (or bits 15 to 2 for the SIR00, SIR02, or SIR10 register) to 0.

Remark 1. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

Remark 2. When the SIRmn register is read, 0000H is always read.

11.3.8 Serial status register mn (SSRmn)

The SSRmn register indicates the state of communications and occurrence of errors for channel n. The errors indicated by this register are a framing error, parity error, and overrun error. The SSRmn register can be read by a 16-bit memory manipulation instruction. The 8 lower-order bits of an SSRmn register can be handled as the SSRmnL register, which can be read by an 8-bit memory manipulation instruction. The value of each SSRmn register following a reset is 0000H.

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

Figure 11 - 12 Format of Serial Status Register mn (SSRmn) (1/2)

Address: F0100H, F0101H (SSR00) to F0106H, F0107H (SSR03),
F0140H, F0141H (SSR10) , F0142H, F0143H (SSR11)

After reset: 0000H

R/W: R

Symbol	15	14	13	12	11	10	9	8
SSRmn	0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
0	TSFmn	BFFmn	0	0	FEFmn **Note**	PEFmn	OVFmn

TSFmn	Flag indicating the state of communications for channel n
0	Communication is stopped or suspended.
1	Communication is in progress.

<Clear conditions>
- The STmn bit of the STm register is set to 1 (communication is stopped) or the SSmn bit of the SSm register is set to 1 (communication is suspended).
- Communication ends.

<Set condition>
- Communication starts.

BFFmn	Flag indicating the state of the buffer register for channel n
0	Valid data is not stored in the SDRmn register.
1	Valid data is stored in the SDRmn register.

<Clear conditions>
- Transferring transmit data from the SDRmn register to the shift register ends during transmission.
- Reading receive data from the SDRmn register ends during reception.
- The STmn bit of the STm register is set to 1 (communication is stopped) or the SSmn bit of the SSm register is set to 1 (communication is enabled).

<Set condition>
- Transmit data is written to the SDRmn register while the TXEmn bit of the SCRmn register is set to 1 (transmission or transmission/reception mode in each communication mode).
- Receive data is stored in the SDRmn register while the RXEmn bit of the SCRmn register is set to 1 (reception or transmission/reception mode in each communication mode).
- A reception error occurs.

Figure 11 - 12 Format of Serial Status Register mn (SSRmn) (2/2)

FEFmn	Framing error detection flag of channel n
0	No error occurs.
1	An error occurs (during UART reception).
<Clear conditions> • 1 is written to the FECTmn bit of the SIRmn register. <Set condition> • A stop bit is not detected when UART reception ends.	

PEFmn	Parity/ACK error detection flag of channel n
0	No error occurs.
1	Parity error occurs (during UART reception) or ACK is not detected (during I^2C transmission).
<Clear conditions> • 1 is written to the PECTmn bit of the SIRmn register. <Set condition> • The parity of the transmit data and the parity bit do not match when UART reception ends (parity error). • No ACK signal is returned from the slave at the ACK reception timing during I^2C transmission (ACK is not detected).	

OVFmn	Overrun error detection flag of channel n
0	No error occurs.
1	An error occurs
<Clear conditions> • 1 is written to the OVCTmn bit of the SIRmn register. <Set condition> • Even though receive data is stored in the SDRmn register, that data is not read and transmit data or the next receive data is written while the RXEmn bit of the SCRmn register is set to 1 (reception or transmission/reception mode in each communication mode). • Transmit data is not ready for slave transmission or transmission/reception in simplified SPI (CSI) mode.	

Note This bit is only present in the SSR01, SSR03, and SSR11 registers.

Caution 1. If data is written to the SDRmn register when BFFmn = 1, the transmit/receive data stored in the register is discarded and an overrun error (OVFmn = 1) is detected.

Caution 2. When the simplified SPI (CSI) is handling reception in the SNOOZE mode (SWCm = 1), the OVFmn and BFFmn flags will not change.

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

11.3.9 Serial channel start register m (SSm)

The SSm is a trigger register that is used to enable starting communication/count by each channel. When 1 is written to a bit (SSmn) of this register, the corresponding bit (SEmn) of serial channel enable status register m (SEm) is set to 1 (operation is enabled). Because the SSmn bit is a trigger bit, it is cleared immediately when SEmn = 1. The SSm register can be set by a 16-bit memory manipulation instruction. The 8 lower-order bits of an SSm register can be handled as the SSmL register, which can be set by a 1-bit or 8-bit memory manipulation instruction. The value of each SSm register following a reset is 0000H.

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

Figure 11 - 13 Format of Serial Channel Start Register m (SSm)

Address: F0122H, F0123H (SS0)
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
SS0	0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
0	0	0	0	SS03	SS02	SS01	SS00

Address: F0162H, F0163H (SS1)[Note]
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
SS1	0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
0	0	0	0	0	0	SS11	SS10

SSmn	Operation start trigger of channel n
0	No trigger operation
1	Set the SEmn bit to 1 to place the channel in the communications waiting state.[Note]

Note Setting an SSmn bit to 1 during communications stops communications through channel n and places the channel in the waiting state. At this time, the values of the control registers and shift register, the states of the SCKmn and SOmn pins, and the values of the FEFmn, PEFmn, and OVFmn flags are retained.

Caution 1. Be sure to set the following bits to 0.
Bits 15 to 4 of the SS0 register
Bits 15 to 2 of the SS1 register
Caution 2. For the UART reception, set the RXEmn bit of SCRmn register to 1, and then be sure to set SSmn to 1 after at least 4 fMCK clock cycles have elapsed.

Remark 1. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)
Remark 2. When the SSm register is read, 0000H is always read.

11.3.10 Serial channel stop register m (STm)

The STm is a trigger register that is used to enable stopping communication/count by each channel. When 1 is written to a bit (STmn) of this register, the corresponding bit (SEmn) of serial channel enable status register m (SEm) is cleared to 0 (operation is stopped). Because the STmn bit is a trigger bit, it is cleared immediately when SEmn = 0. The STm register can be set by a 16-bit memory manipulation instruction. The 8 lower-order bits of an STm register can be handled as the STmL register, which can be set by a 1-bit or 8-bit memory manipulation instruction. The value of each STm register following a reset is 0000H.

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

Figure 11 - 14 Format of Serial Channel Stop Register m (STm)

Address: F0124H, F0125H (ST0)
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
ST0	0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
0	0	0	0	ST03	ST02	ST01	ST00

Address: F0164H, F0165H (ST1)[Note]
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
ST1	0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
0	0	0	0	0	0	ST11	ST10

STmn	Operation stop trigger of channel n
0	No trigger operation
1	Clears the SEmn bit to 0 and stops the communication operation[Note].

Note The values of the control registers and shift register, the states of the SCKmn and SOmn pins, and the values of the FEFmn, PEFmn, and OVFmn flags are retained.

Caution **Be sure to set the following bits to 0.**
Bits 15 to 4 of the ST0 register
Bits 15 to 2 of the ST1 register

Remark 1. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)
Remark 2. When the STm register is read, 0000H is always read.

11.3.11 Serial channel enable status register m (SEm)

The SEm register indicates whether data transmission/reception operation of each channel is enabled or stopped. When 1 is written to a bit of serial channel start register m (SSm), the corresponding bit of this register is set to 1. When 1 is written to a bit of serial channel stop register m (STm), the corresponding bit is cleared to 0. For channel n whose operation is enabled, the value of the CKOmn bit of serial output register m (SOm) to be described later cannot be rewritten by software, and a value reflected by a communication operation is output from the serial clock pin. For channel n whose operation is stopped, the value of the CKOmn bit of the SOm register can be set by software and is output from the serial clock pin. In this way, any waveform, such as that of a start condition/stop condition, can be created by software. The SEm register can be read by a 16-bit memory manipulation instruction. The 8 lower-order bits of an SEm register can be handled as the SEmL register, which can be read by a 1-bit or 8-bit memory manipulation instruction. The value of each SEm register following a reset is 0000H.

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

Figure 11 - 15 Format of Serial Channel Enable Status Register m (SEm)

Address: F0120H, F0121H (SE0)
After reset: 0000H
R/W: R

Symbol	15	14	13	12	11	10	9	8
SE0	0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
0	0	0	0	SE03	SE02	SE01	SE00

Address: F0160H, F0161H (SE1)
After reset: 0000H
R/W: R

Symbol	15	14	13	12	11	10	9	8
SE1	0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
0	0	0	0	0	0	SE11	SE10

SEmn	Indication of whether operation of channel n is enabled or stopped.
0	Operation stops
1	Operation is enabled.

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

11.3.12 Serial output enable register m (SOEm)

The SOEm register is used to enable or stop output of the serial communication operation of each channel. For channel n whose serial output is enabled, the value of the SOmn bit of serial output register m (SOm) to be described later cannot be rewritten by software, and a value reflected by a communication operation is output from the serial data output pin. For channel n, whose serial output is stopped, the SOmn bit value of the SOm register can be set by software, and that value can be output from the serial data output pin. In this way, any waveform, such as that of a start condition/stop condition, can be created by software. The SOEm register can be set by a 16-bit memory manipulation instruction. The 8 lower-order bits of an SOEm register can be handled as the SOEmL register, which can be set by a 1-bit or 8-bit memory manipulation instruction. The value of each SOEm register following a reset is 0000H.

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

Figure 11 - 16 Format of Serial Output Enable Register m (SOEm)

Address: F012AH, F012BH (SOE0)
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
SOE0	0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
0	0	0	0	SOE03	SOE02	SOE01	SOE00

Address: F016AH, F016BH (SOE1)
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
SOE1	0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
0	0	0	0	0	0	SOE11	SOE10

SOEmn	Serial output enable/stop of channel n
0	Stops output by serial communication operation.
1	Enables output by serial communication operation.

Caution **Be sure to set the following bits to 0.**
Bits 15 to 4 of the SOE0 register
Bits 15 to 2 of the SOE1 register

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

11.3.13 Serial output register m (SOm)

The SOm is a buffer register for serial output of each channel. The value of the SOmn bit of this register is output from the serial data output pin of channel n. The value of the CKOmn bit of this register is output from the serial clock output pin of channel n. The SOmn bit of this register can be rewritten by software only when serial output is disabled (SOEmn = 0). When serial output is enabled (SOEmn = 1), rewriting by software is ignored, and the value of the register can be changed only by a serial communication operation. The CKOmn bit of this register can be rewritten by software only when the channel operation is stopped (SEmn = 0). While channel operation is enabled (SEmn = 1), rewriting by software is ignored, and the value of the CKOmn bit can be changed only by a serial communication operation. To use the pin for serial interface as a port function pin, set the corresponding CKOmn and SOmn bits to 1. The SOm register can be set by a 16-bit memory manipulation instruction. The values of the SO0 and SO1 registers following a reset are respectively 0F0FH and 0303H.

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

Figure 11 - 17 Format of Serial Output Register m (SOm)

Address: F0128H, F0129H (SO0)
After reset: 0F0FH
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
SO0	0	0	0	0	CKO03	CKO02	CKO01	CKO00

7	6	5	4	3	2	1	0
0	0	0	0	SO03	SO02	SO01	SO00

Address: F0168H, F0169H (SO1)
After reset: 0303H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
SO1	0	0	0	0	0	0	CKO11	CKO10

7	6	5	4	3	2	1	0
0	0	0	0	0	0	SO11	SO10

CKOmn	Serial clock output of channel n
0	Serial clock output value is 0.
1	Serial clock output value is 1.

SOmn	Serial data output of channel n
0	Serial data output value is 0.
1	Serial data output value is 1.

Caution **Be sure to set the following bits to 0.**
Bits 15 to 12 and 7 to 4 of the SO0 register
Bits 15 to 10 and 7 to 2 of the SO1 register

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

11.3.14 Serial output level register m (SOLm)

The SOLm register is used to set inversion of the data output level of each channel. This register can be set only in the UART mode. Be sure to set 0 for the bit corresponding the channel used in the simplified SPI (CSI) mode or simplified I^2C mode. Inverting channel n by using this register is reflected on pin output only when serial output is enabled (SOEmn = 1). When serial output is disabled (SOEmn = 0), the value of the SOmn bit is output as is. Rewriting the SOLm register is prohibited when the register is in operation (when SEmn = 1). The SOLm register can be set by a 16-bit memory manipulation instruction. The 8 lower-order bits of an SOLm register can be handled as the SOLmL register, which can be set by an 8-bit memory manipulation instruction. The value of each SOLm register following a reset is 0000H.

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

Figure 11 - 18 Format of Serial Output Level Register m (SOLm)

Address: F0134H, F0135H (SOL0)
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
SOL0	0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
0	0	0	0	0	SOL02	0	SOL00

Address: F0174H, F0175H (SOL1)
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
SOL1	0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	SOL10

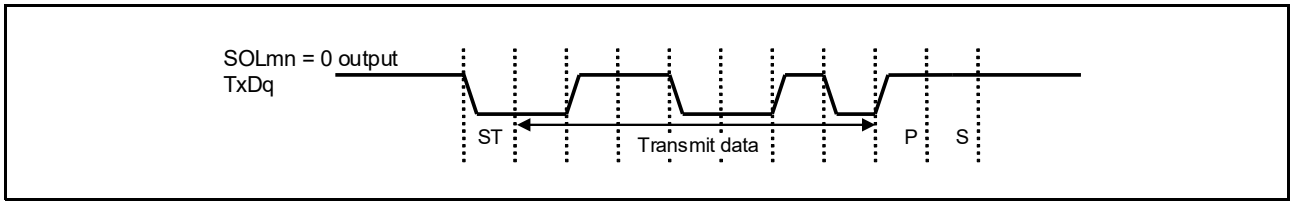
SOLmn	Selects inversion of the level of the transmit data of channel n in UART mode
0	Communication data is output as is.
1	Communication data is inverted and output.

Caution Be sure to set the following bits to 0.
Bits 15 to 3, and 1 of the SOL0 register
Bits 15 to 1 of the SOL1 register

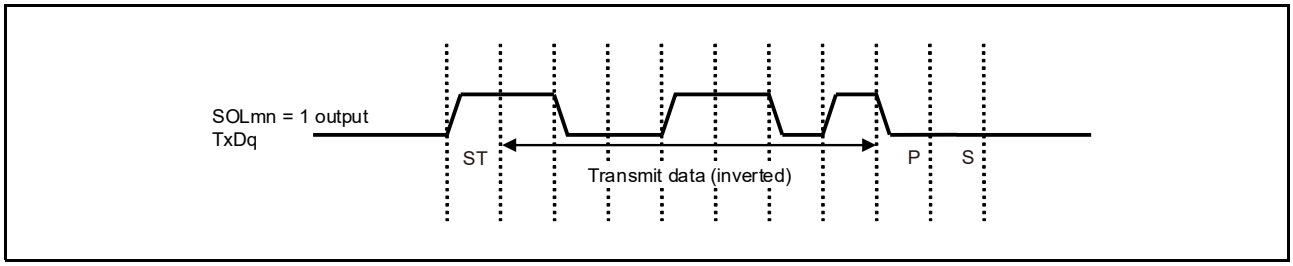
Figure 11 - 19 shows examples in which the level of transmit data is reversed during UART transmission.

Figure 11 - 19 Examples of Reverse Transmit Data

a) Non-reverse Output (SOLmn = 0)



b) Reverse Output (SOLmn = 1)



Remark m: Unit number (m = 0, 1), n: Channel number (n = 0, 2)

11.3.15 Serial standby control register m (SSCm)

The SSC0 register is used to control the startup of reception (the SNOOZE mode) while in the STOP mode when receiving CSI00 or UART0 serial data. The SSCm register can be set by a 16-bit memory manipulation instruction. The 8 lower-order bits of an SSCm register can be handled as the SSCmL register, which can be set by an 8-bit memory manipulation instruction. The value of each SSCm register following a reset is 0000H.

Caution The maximum transfer rate in the SNOOZE mode is as follows.

- **Up to 1 Mbps in CSI00**
- **Up to 9.6 kbps in UART0 with f_{CLK} = f_{IH} (32 MHz)**

Remark m: Unit number (m = 0)

Figure 11 - 20 Format of Serial Standby Control Register m (SSCm)

Address: F0138H, F0139H (SSC0)
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
SSCm	0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
0	0	0	0	0	0	SSECm	SWCm

SSECm	Selection of whether to enable or disable the generation of communication error interrupts in the SNOOZE mode
0	Enable the generation of error interrupts (INTSRE0/INTSRE2).
1	Disable the generation of error interrupts (INTSRE0/INTSRE2).
• The SSECm bit can be set to 1 or 0 only when both the SWCm and EOCmn bits are set to 1 during UART reception in the SNOOZE mode. In other cases, set the SSECm bit to 0. • Setting SSECm and SWCm to 10B is prohibited.	

SWCm	Setting of the SNOOZE mode
0	Do not use the SNOOZE mode function.
1	Use the SNOOZE mode function.
• When there is a hardware trigger signal in the STOP mode, the STOP mode is exited, and simplified SPI (CSI) or UART reception is performed without operating the CPU (the SNOOZE mode). • The SNOOZE mode function can only be specified when the high-speed on-chip oscillator clock or medium-speed on-chip oscillator clock is selected for the CPU/peripheral hardware clock (f_{CLK}). If any other clock is selected, specifying this mode is prohibited. • Even when using SNOOZE mode, be sure to set the SWCm bit to 0 in normal operation mode and change it to 1 just before shifting to STOP mode. Also, be sure to change the SWCm bit to 0 after returning from STOP mode to normal operation mode.	

Remark m: Unit number (m = 0), n: Channel number (n = 1)

Table 11 - 2 Interrupt in UART Reception Operation in SNOOZE Mode

EOCmn Bit	SSECm Bit	Reception Ended Successfully	Reception Ended in an Error
0	0	INTSRx is generated.	INTSRx is generated.
0	1	INTSRx is generated.	INTSRx is generated.
1	0	INTSRx is generated.	INTSREx is generated.
1	1	INTSRx is generated.	No interrupt is generated.

Remark m: Unit number (m = 0), n: Channel number (n = 1)

11.3.16 Input switch control register (ISC)

<R> The ISC1 and ISC0 bits of the ISC register are used to realize a LIN-bus communication by UART2 in coordination with an external interrupt and the timer array unit. The ISC4 and ISC3 bits are used to select the serial clock input source of
<R> CSI00. The ISC7 and ISC6 bits are used to select the serial clock input source of CSI01. When bit 0 is set to 1, the input signal of the serial data input (RxD2) pin is selected as an external interrupt (INTP0) that can be used to detect a wakeup signal. When bit 1 is set to 1, the input signal of the serial data input (RxD2) pin is selected as a timer input, so that wake up signal can be detected, the low-level width of the break field, and the pulse width of the sync field can be measured by the timer. Bits 3 and 4 can be used to select the SCK00 pin input or TO01 output signal as the serial clock input source of CSI00. Bits 6 and 7 can be used to select the SCK01 pin input or TO01 output signal as the serial clock input source of CSI01. The ISC register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of the ISC register following a reset is 00H.

Figure 11 - 21 Format of Input Switch Control Register (ISC)

Address: F0073H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
ISC	ISC7	ISC6	0	ISC4	ISC3	0	ISC1	ISC0

ISC7	ISC6	Switch of the serial clock input source of CSI01Note 1
0	0	Input signal of the SCK01 pin (normal operation)
0	1	Setting prohibited
1	0	TO01 output signal
1	1	Setting prohibited

ISC4	ISC3	Switch of the serial clock input source of CSI00Note 2
0	0	Input signal of the SCK00 pin (normal operation)
0	1	Setting prohibited
1	0	TO01 output signal
1	1	Setting prohibited

ISC1	Switching channel 7 input of timer array unit
0	Uses the input signal of the TI07 pin as a timer input (normal operation).
1	Input signal of the RxD2 pin is used as timer input (detects the wakeup signal and measures the low-level width of the break field and the pulse width of the sync field).

ISC0	Switching external interrupt (INTP0) input
0	Uses the input signal of the INTP0 pin as an external interrupt (normal operation).
1	Uses the input signal of the RxD2 pin as an external interrupt (wakeup signal detection).

Note 1. When UART mode or simplified I^2C mode is to be selected for channel 1, set the ISC7 and ISC6 bits to 0.

Note 2. When UART mode or simplified I^2C mode is to be selected for channel 0, set the ISC4 and ISC3 bits to 0.

Caution Be sure to set bits 5 and 2 to 0.

11.3.17 Noise filter enable register 0 (NFEN0)

The NFEN0 register is used to set whether the noise filter can be used for the input signal from the serial data input pin to each channel. Disable the noise filter of the pin used for simplified SPI (CSI) or simplified I^2C communication, by clearing the corresponding bit of this register to 0. Enable the noise filter of the pin used for UART communication, by setting the corresponding bit of this register to 1. When the noise filter is enabled, after synchronization is performed with the operation clock (f_{MCK}) of the target channel, 2-clock match detection is performed. When the noise filter is disabled, only synchronization is performed with the operation clock (f_{MCK}) of the target channel. The NFEN0 register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of the NFEN0 register following a reset is 00H.

Figure 11 - 22 Format of Noise Filter Enable Register 0 (NFEN0)

Address: F0070H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
NFEN0	0	0	0	SNFEN20	0	SNFEN10	0	SNFEN00

SNFEN20	Use of noise filter of RxD2 pin
0	Noise filter OFF
1	Noise filter ON
Set SNFEN20 to 1 to use the RxD2 pin. Set SNFEN20 to 0 to use the other than RxD2 pin.	

SNFEN10	Use of noise filter of RxD1 pin
0	Noise filter OFF
1	Noise filter ON
Set the SNFEN10 bit to 1 to use the RxD1 pin. Set SNFEN10 to 0 to use the other than RxD1 pin.	

SNFEN00	Use of noise filter of RxD0 pin
0	Noise filter OFF
1	Noise filter ON
Set the SNFEN00 bit to 1 to use the RxD0 pin. Set SNFEN00 to 0 to use the other than RxD0 pin.	

Caution Be sure to set bits 7 to 5, 3, and 1 to 0.

11.3.18 Registers for controlling the port functions multiplexed with the inputs and outputs of the serial array unit

Set the following registers to control the port functions multiplexed with the inputs and outputs of the serial array unit.

- Port mode registers (PMxx)
- Port registers (Pxx)
- Port input mode registers (PIMxx)
- Port output mode registers (POMxx)
- Port mode control A registers (PMCAxx)

For details, see the following sections.

- **4.3.1 Port mode registers (PMxx)**
- **4.3.2 Port registers (Pxx)**
- **4.3.4 Port input mode registers (PIMxx)**
- **4.3.5 Port output mode registers (POMxx)**
- **4.3.7 Port mode control A registers (PMCAxx)**

For examples of register settings for controlling port functions, see **4.5.4 Examples of register settings for port and alternate functions**. To communicate with an external device operating at a different voltage (1.8 V, 2.5 V or 3 V), see **4.4.5 Communications with devices operating at a different voltage (1.8 V, 2.5 V, or 3 V) by switching I/O buffers**.

11.3.19 UART loopback select register (ULBS)

The ULBS register is used to enable the UART loopback function. This register has bits to individually control UART channels. When the bit corresponding to each channel is set to 1, the UART loopback function is selected, and output from the transmission shift register is looped back to the reception shift register. The ULBS register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of the ULBS register following a reset is 00H.

Figure 11 - 23 Format of UART Loopback Select Register (ULBS)

Address: F0079H
After reset: 00H
R/W: R/W

Symbol	7	6	<5>	<4>	<3>	<2>	<1>	<0>
ULBS	0	0	0	ULBS4	0	ULBS2	ULBS1	ULBS0

ULBSn	Selection of the UART loopback function
0	Inputs the states of the RxD0 to RxD2 pins of serial array unit UART0 to UART2 to the reception shift register.
1	Loops back output from the transmission shift register to the reception shift register.

Caution **Be sure to set bits 7 to 5 and 3 to 0.**

Remark n = 0 to 2

11.4 Operation Stop Mode

Each serial interface of serial array unit has the operation stop mode. In this mode, serial communication cannot be executed, thus reducing the power consumption. In addition, the pin for serial interface can be used as port function pins in this mode.

11.4.1 Stopping the operation by units

The stopping of the operation by units is set by using peripheral enable register 0 (PER0). The PER0 register is used to enable or disable the supply of a clock signal to various on-chip peripheral modules. Clock supply to an on-chip peripheral module that is not to be used can be stopped to decrease power consumption and noise. To stop the operation of serial array unit 0, be sure to set the SAU0EN bit to 0. To stop the operation of serial array unit 1, be sure to set the SAU1EN bit to 0.

Figure 11 - 24 Peripheral Enable Register 0 (PER0) Setting When Stopping the Operation by Units

a) Peripheral enable register 0 (PER0): Only set the bit corresponding to SAUm to be stopped to 0.

	7	6	5	4	3	2	1	0
PER0	0	0	ADCEN ×	IICA0EN[Note] ×	SAU1EN 0/1	SAU0EN 0/1	0	TAU0EN ×

Control of SAUm input clock
0: Stops supply of input clock
1: Supplies input clock

Note This bit is only present in the 32- to 64-pin products.

Caution 1. Be sure to set bits 7, 6, and 1 to 0.
Caution 2. Be sure to set bit 4 in the 20-pin products to 0.
Caution 3. If SAUmEN = 0, writing to the registers which control serial array unit m is ignored, and, even if the register is read, only the initial value is read. Note that this does not apply to the following registers.
- **Input switch control register (ISC)**
- **Noise filter enable register 0 (NFEN0)**
- **Port input mode registers (PIM0, PIM1, PIM5, PIM7)**
- **Port output mode registers (POM0, POM1, POM5, POM7)**
- **Port mode control A register (PMCA0)**
- **Port mode registers (PM0, PM1, PM3, PM5, PM7)**
- **Port registers (P0, P1, P3, P5, P7)**

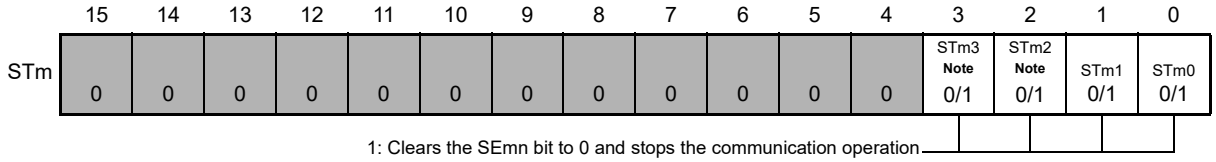
Remark ×: Bits not used with serial array units (depending on the settings of other peripheral functions)
0/1: Set to 0 or 1 depending on the user's application.

11.4.2 Stopping the operation by channels

The stopping of the operation by channels is set using each of the following registers.

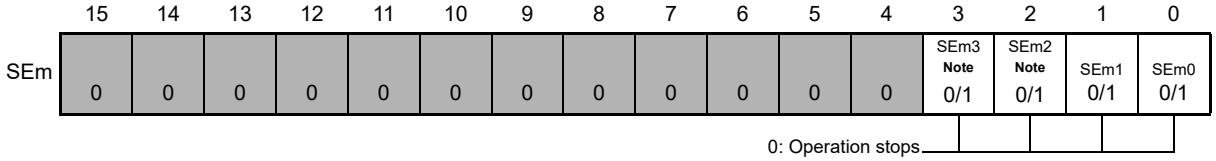
Figure 11 - 25 Each Register Setting When Stopping the Operation by Channels

a) Serial channel stop register m (STm): The STm is a trigger register that is used to enable stopping communication/count by each channel.



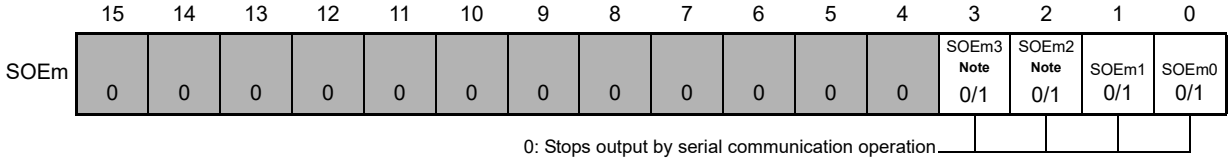
* Because the STmn bit is a trigger bit, it is cleared immediately when SEmn = 0.

b) Serial channel enable status register m (SEm): This register indicates whether data transmission/reception operation of each channel is enabled or stopped.



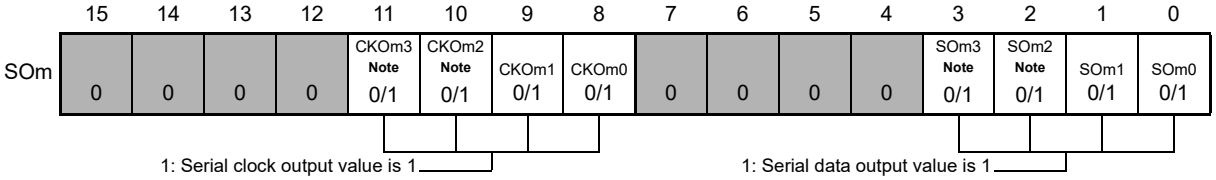
* The SEm is a read-only status register, whose operation is stopped by using the STm register.
With a channel whose operation is stopped, the value of the CKOmn bit of the SOm register can be set by software.

c) Serial output enable register m (SOEm): This register is used to enable or stop output of the serial communication operation of each channel.



* For channel n, whose serial output is stopped, the SOmn bit value of the SOm register can be set by software.

d) Serial output register m (SOm): The SOm is a buffer register for serial output of each channel.



* When using pins corresponding to each channel as port function pins, set the corresponding CKOmn, SOmn bits to 1.

Note These bits are only present in products with 80 to 128 pins of serial array unit 1.

Remark 1. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

Remark 2. ■: Setting disabled (fixed by hardware), 0/1: Set to 0 or 1 depending on the user's application.

11.5 Operation of Simplified SPI (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) Communication

The simplified SPI allows clocked communications using three lines: serial clock (SCK), serial input (SI), and serial output (SO).

[Data transmission/reception]

- Data length of 7 or 8 bits
- Phase control of transmit/receive data
- MSB/LSB first selectable

[Clock control]

- Master/slave selection
- Phase control of I/O clock
- Setting of transfer period by prescaler and internal counter of each channel
- Maximum transfer rate**Note** During master communication: Max. f_{CLK}/2 (CSI00 only)
  Max. f_{CLK}/4
  During slave communication: Max. f_{MCK}/6

[Interrupt function]

- Transfer end interrupt/buffer empty interrupt

[Error detection flag]

- Overrun error

In addition, the simplified SPI (CSI00) supports the SNOOZE mode. In the SNOOZE mode, data can be received without CPU processing upon detecting SCK input in the STOP mode.

Note Set up the transfer rate within a range satisfying the SCK cycle time (t_{KCY}). For details, see **Section 28 Electrical Characteristics**.

Caution Use a general-purpose port pin to send a chip select signal when required.

The channels supporting simplified SPI (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) are channels 0 to 3 of SAU0 and channels 0 and 1 of SAU1.

20-pin products

Unit	Channel	Used as Simplified SPI (CSI)	Used as UART	Used as Simplified I²C
0	0	CSI00	UART0	IIC00
	1	—		—
	2	—	UART1	—
	3	CSI11		IIC11
1	0	—	UART2 (supporting the LIN bus)	—
	1	—		—

32-pin products

Unit	Channel	Used as Simplified SPI (CSI)	Used as UART	Used as Simplified I²C
0	0	CSI00	UART0	IIC00
	1	—		—
	2	—	UART1	—
	3	CSI11		IIC11
1	0	CSI20	UART2 (supporting the LIN bus)	IIC20
	1	—		—

44-pin products

Unit	Channel	Used as Simplified SPI (CSI)	Used as UART	Used as Simplified I²C
0	0	CSI00	UART0	IIC00
	1	—		—
	2	—	UART1	—
	3	CSI11		IIC11
1	0	CSI20	UART2 (supporting the LIN bus)	IIC20
	1	CSI21		IIC21

48-pin products

Unit	Channel	Used as Simplified SPI (CSI)	Used as UART	Used as Simplified I²C
0	0	CSI00	UART0	IIC00
	1	CSI01		IIC01
	2	—	UART1	—
	3	CSI11		IIC11
1	0	CSI20	UART2 (supporting the LIN bus)	IIC20
	1	CSI21		IIC21

64-pin products

Unit	Channel	Used as Simplified SPI (CSI)	Used as UART	Used as Simplified I²C
0	0	CSI00	UART0	IIC00
	1	CSI01		IIC01
	2	CSI10	UART1	IIC10
	3	CSI11		IIC11
1	0	CSI20	UART2 (supporting the LIN bus)	IIC20
	1	CSI21		IIC21

Simplified SPI (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) performs the following seven types of communication operations.

- Master transmission (See **11.5.1**.)
- Master reception (See **11.5.2**.)
- Master transmission/reception (See **11.5.3**.)
- Slave transmission (See **11.5.4**.)
- Slave reception (See **11.5.5**.)
- Slave transmission/reception (See **11.5.6**.)
- SNOOZE mode function (See **11.5.7**.)

11.5.1 Master transmission

Master transmission is that this MCU outputs a transfer clock and transmits data to another device.

Simplified SPI	CSI00	CSI01	CSI10	CSI11	CSI20	CSI21
Target channel	Channel 0 of SAU0	Channel 1 of SAU0	Channel 2 of SAU0	Channel 3 of SAU0	Channel 0 of SAU1	Channel 1 of SAU1
Pins used	SCK00, SO00	SCK01, SO01	SCK10, SO10	SCK11, SO11	SCK20, SO20	SCK21, SO21
Interrupt	INTCSI00	INTCSI01	INTCSI10	INTCSI11	INTCSI20	INTCSI21
	Transfer end interrupt (in single-transfer mode) or buffer empty interrupt (in continuous transfer mode) can be selected.					
Error detection flag	None					
Transfer data length	7 or 8 bits					
Transfer rate**Note**	Max. $f_{CLK}/2$ [Hz] (CSI00 only), $f_{CLK}/4$ [Hz] Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [Hz] f_{CLK}: CPU/peripheral hardware clock frequency					
Data phase	Selectable by the DAPmn bit of the SCRmn register • DAPmn = 0: Data output starts from the start of the operation of the serial clock. • DAPmn = 1: Data output starts half a clock cycle before the start of the serial clock operation.					
Clock phase	Selectable by the CKPmn bit of the SCRmn register • CKPmn = 0: Non-reverse • CKPmn = 1: Reverse					
Data direction	MSB or LSB first					

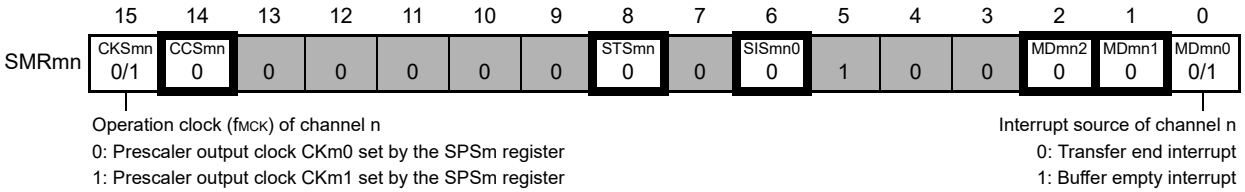
Note Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics specified in the electrical characteristics. For details, see **Section 28 Electrical Characteristics**.

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

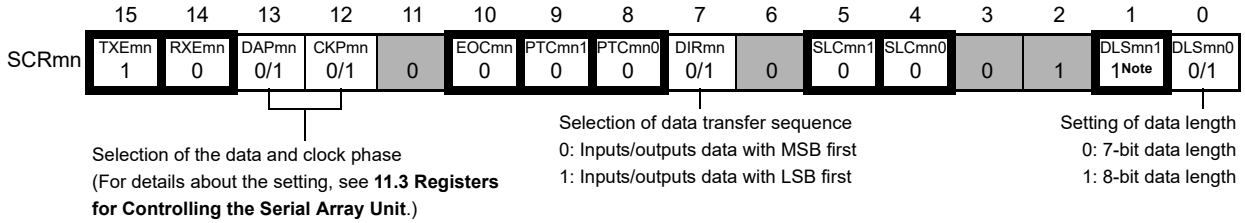
1. Register setting

Figure 11 - 26 Example of Contents of Registers for Master Transmission of Simplified SPI (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21)

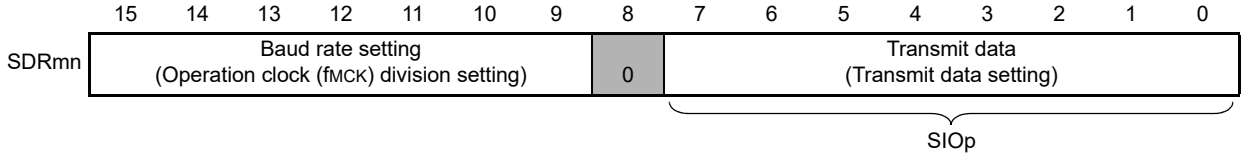
a) Serial mode register mn (SMRmn)



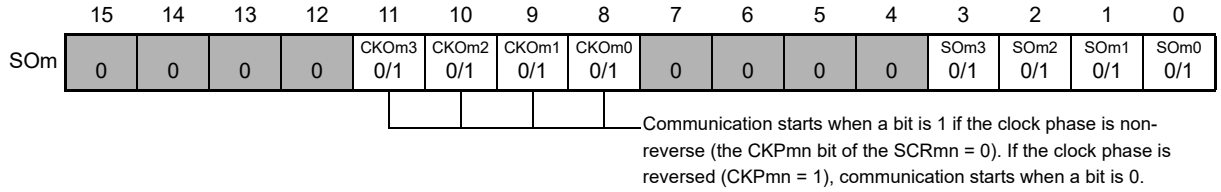
b) Serial communication operation setting register mn (SCRmn)



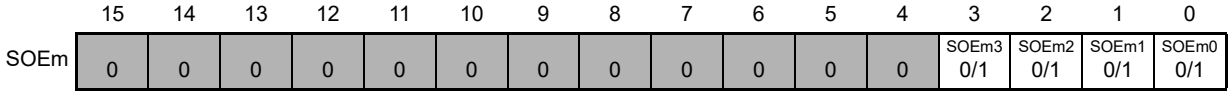
c) Serial data register mn (SDRmn) (lower 8 bits: SIOp)



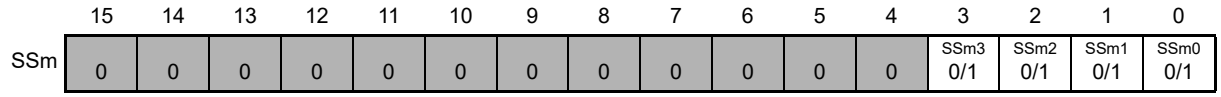
d) Serial output register m (SOm): Set only the bit of the target channel.



e) Serial output enable register m (SOEm): Set only the bit of the target channel to 1.



f) Serial channel start register m (SSm): Set only the bit of the target channel to 1.



Note This bit is only present in the SCR00 and SCR01 registers, and is fixed to 1 for the other registers.

(Remarks are listed on the next page.)

Remark 1. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

Remark 2. ■: Setting is fixed in the simplified SPI (CSI) master transmission mode, ■: Setting disabled (set to the initial value)
0/1: Set to 0 or 1 depending on the user's application.

2. Operation procedure

Figure 11 - 27 Initial Setting Procedure for Master Transmission

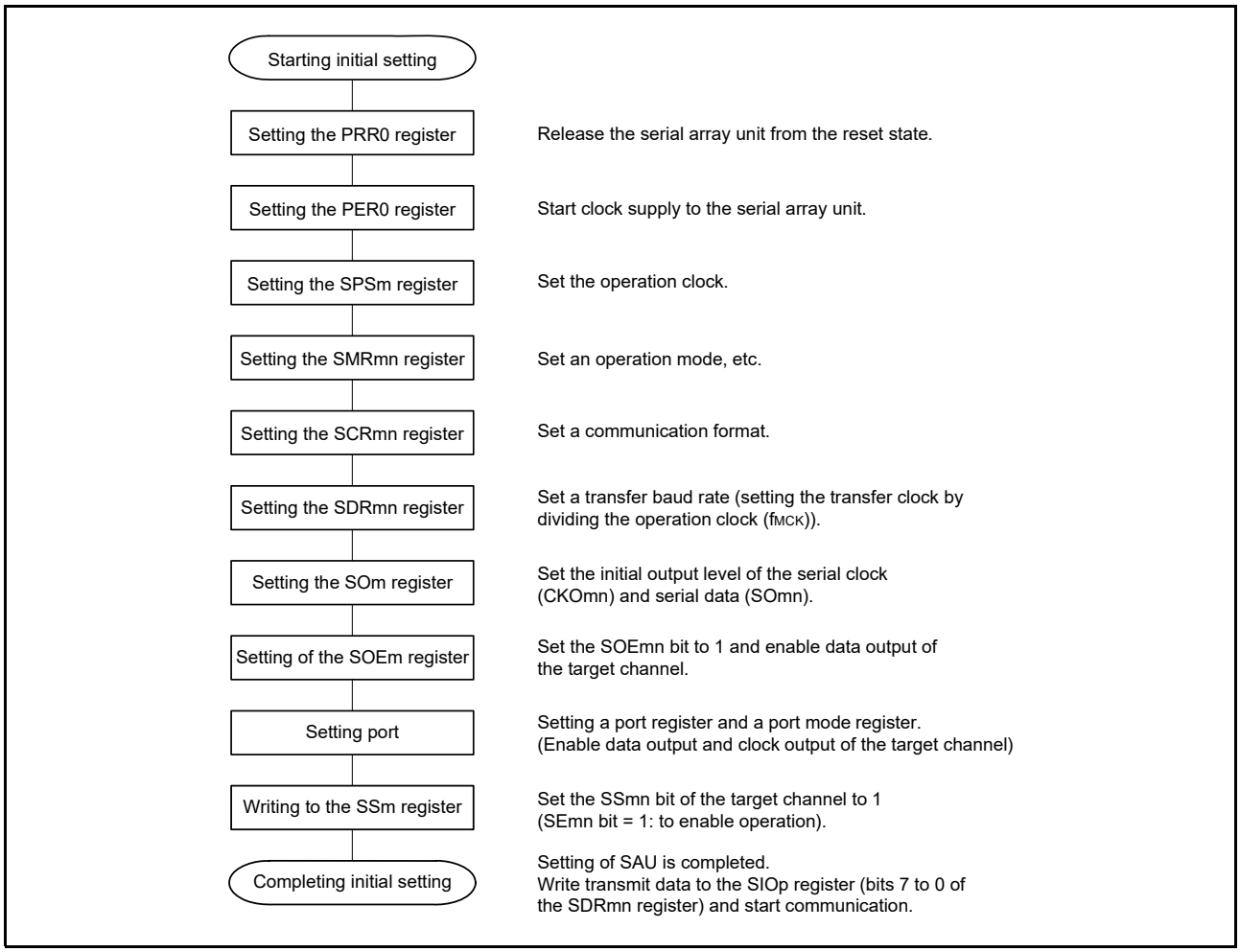


Figure 11 - 28 Procedure for Stopping Master Transmission

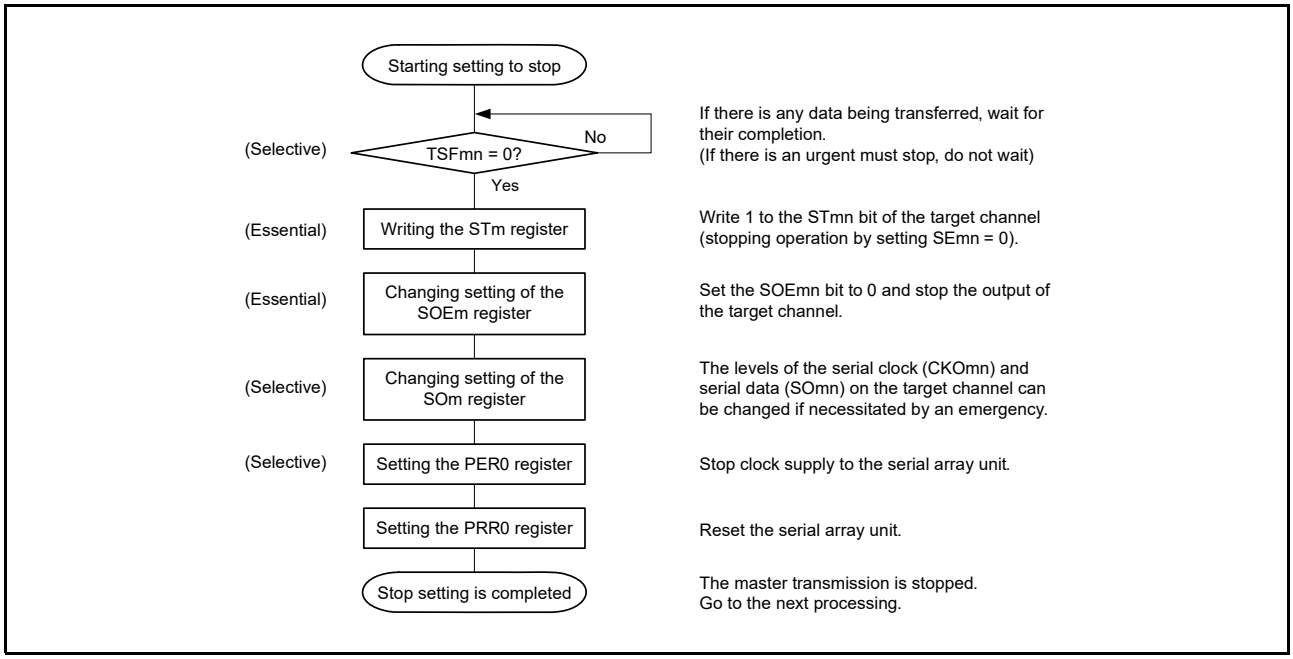
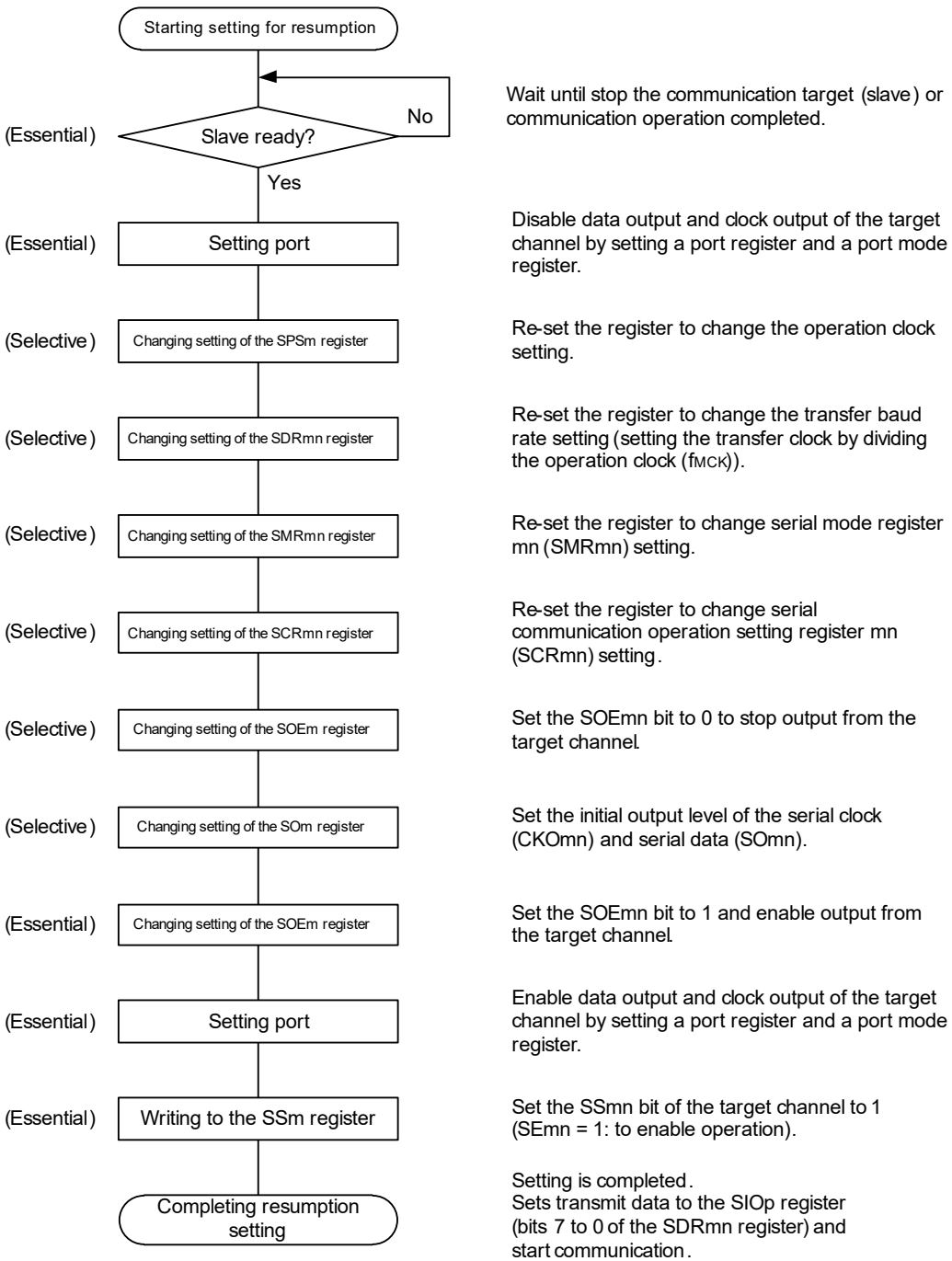


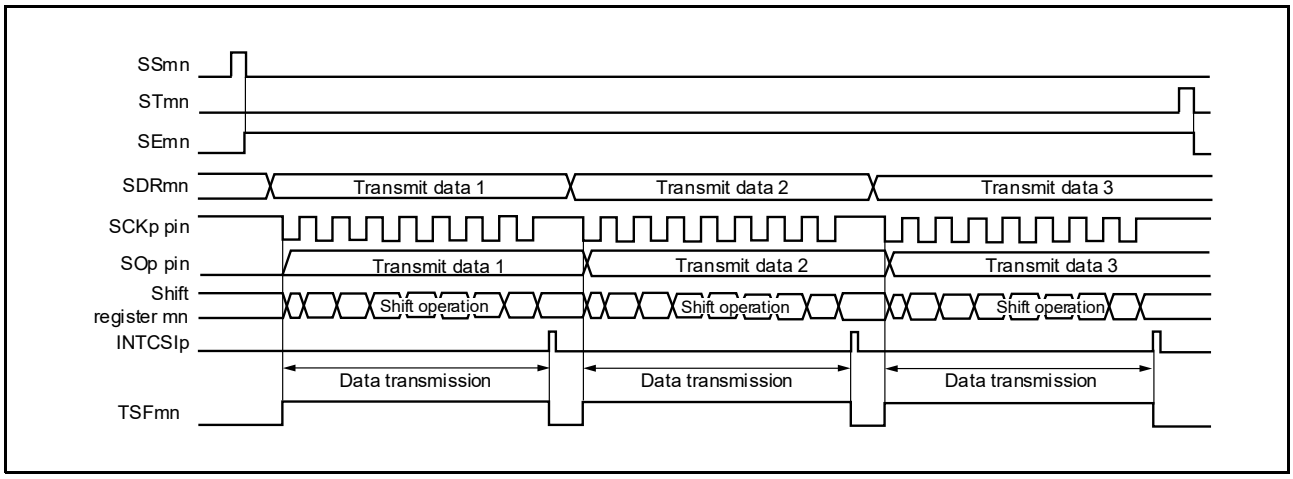
Figure 11 - 29 Procedure for Resuming Master Transmission



Remark If the PRR0 register is rewritten while stopping the communication to reset the serial array unit, wait until the communication target (slave) stops or communication finishes, and then perform initialization instead of restarting the communication.

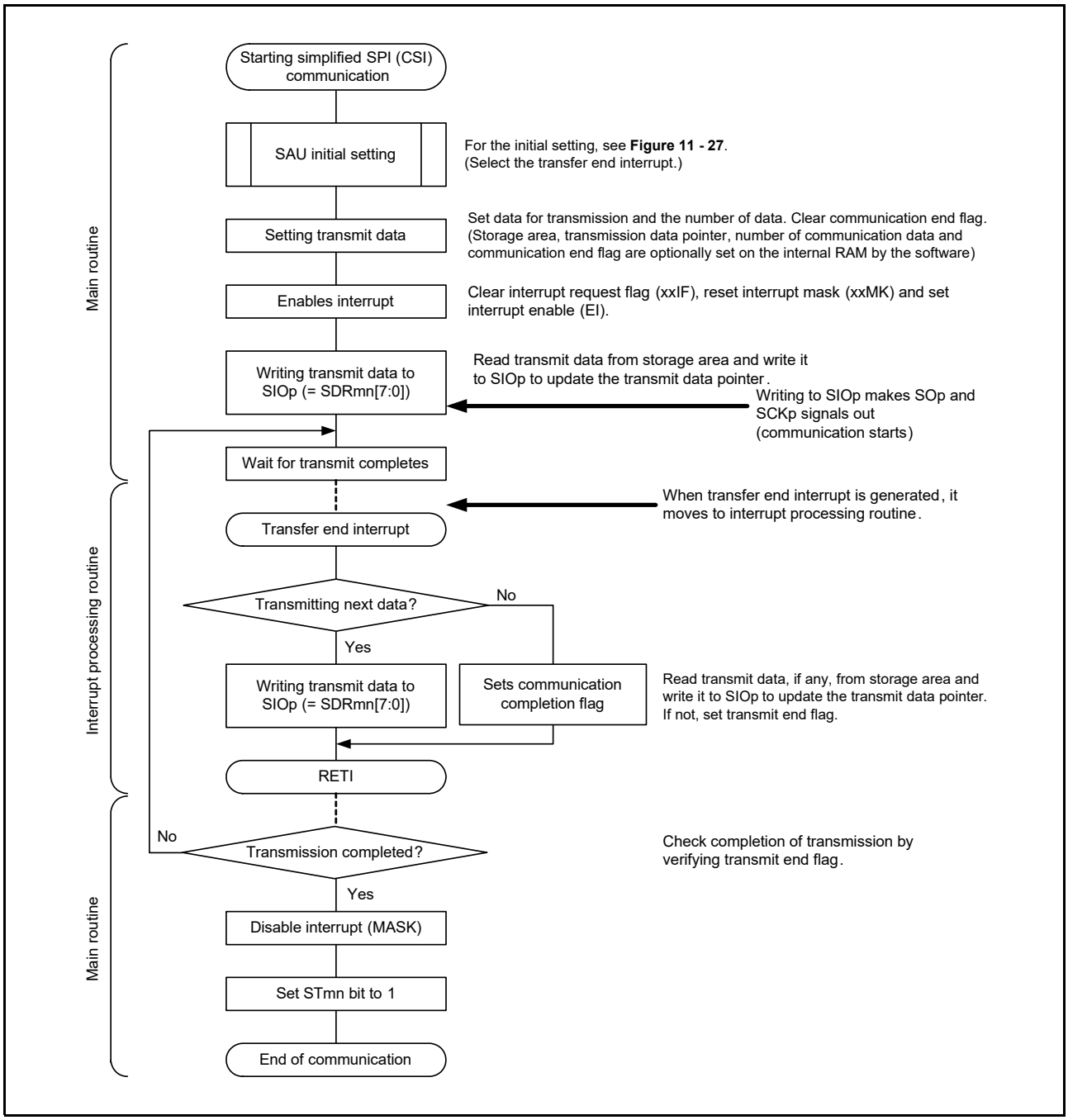
3. Processing flow (in single-transmission mode)

Figure 11 - 30 Timing Chart of Master Transmission (in Single-transmission Mode) (Type 1: DAPmn = 0, CKPmn = 0)



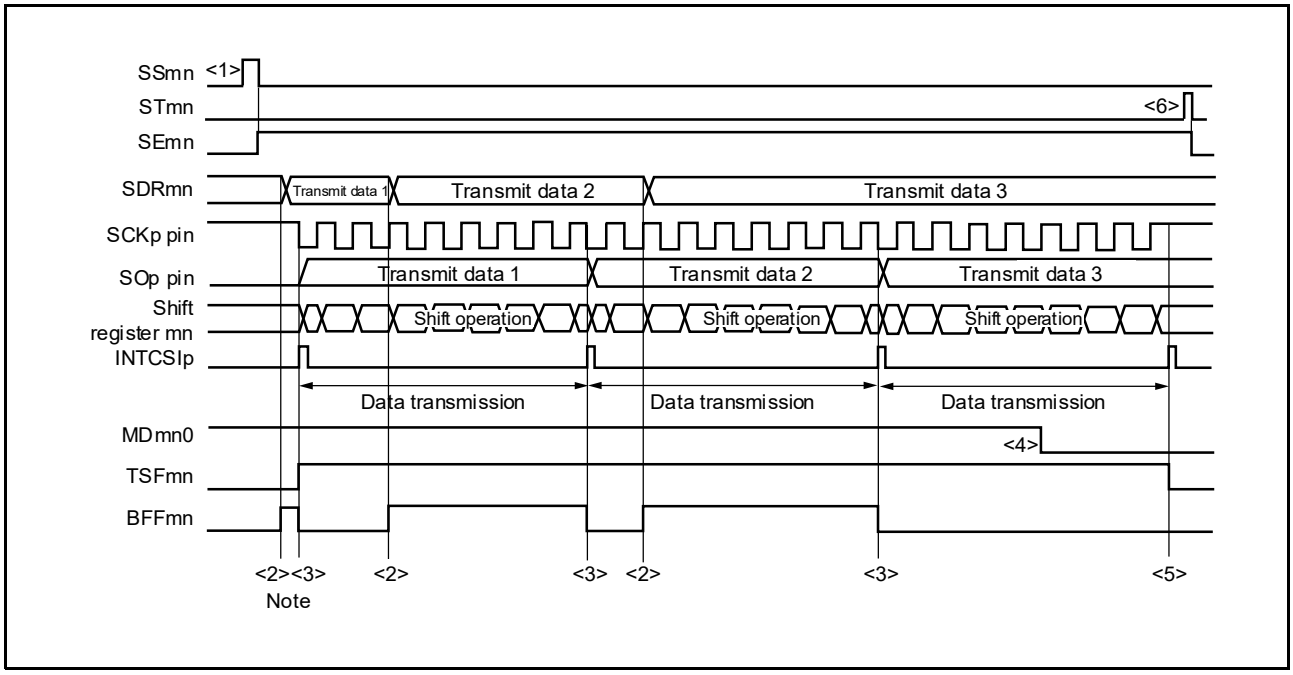
Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

Figure 11 - 31 Flowchart of Master Transmission (in Single-transmission Mode)



4. Processing flow (in continuous transmission mode)

Figure 11 - 32 Timing Chart of Master Transmission (in Continuous Transmission Mode)
(Type 1: DAPmn = 0, CKPmn = 0)

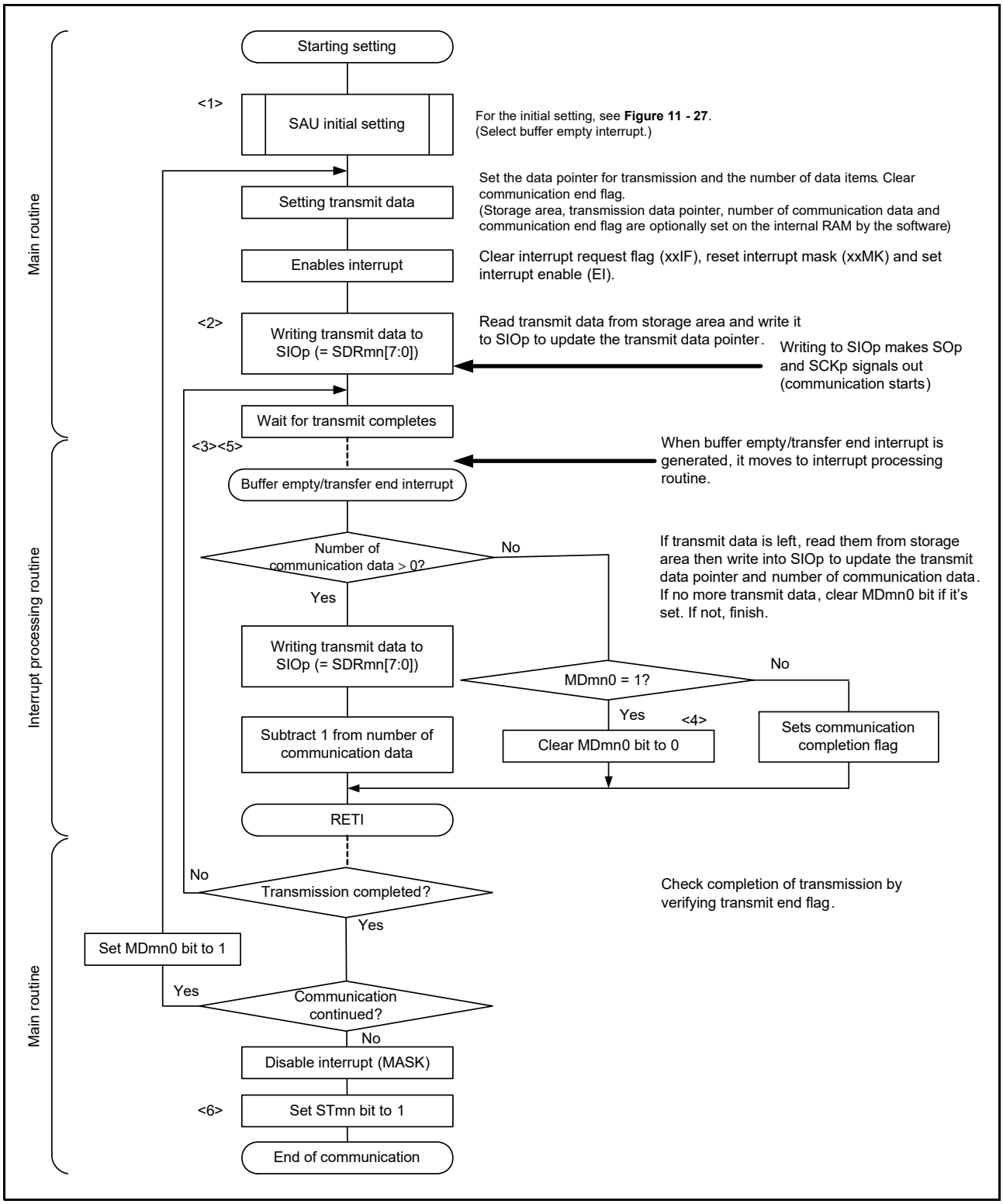


Note If transmit data is written to the SDRmn register while the BFFmn bit of serial status register mn (SSRmn) is 1 (valid data is stored in serial data register mn (SDRmn)), the transmit data is overwritten.

Caution **The MDmn0 bit of serial mode register mn (SSRmn) can be rewritten even during operation. However, rewrite it before transfer of the last bit is started, so that it has been rewritten before the transfer end interrupt of the last transmit data.**

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

Figure 11 - 33 Flowchart of Master Transmission (in Continuous Transmission Mode)



Remark <1> to <6> in the figure correspond to <1> to <6> in **Figure 11 - 32 Timing Chart of Master Transmission (in Continuous Transmission Mode) (Type 1: DAPmn = 0, CKPmn = 0)**.

11.5.2 Master reception

Master reception is that this MCU outputs a transfer clock and receives data from another device.

Simplified SPI	CSI00	CSI01	CSI10	CSI11	CSI20	CSI21
Target channel	Channel 0 of SAU0	Channel 1 of SAU0	Channel 2 of SAU0	Channel 3 of SAU0	Channel 0 of SAU1	Channel 1 of SAU1
Pins used	SCK00, SI00	SCK01, SI01	SCK10, SI10	SCK11, SI11	SCK20, SI20	SCK21, SI21
Interrupt	INTCSI00	INTCSI01	INTCSI10	INTCSI11	INTCSI20	INTCSI21
	Transfer end interrupt (in single-transfer mode) or buffer empty interrupt (in continuous transfer mode) can be selected.					
Error detection flag	Overrun error detection flag (OVFmn) only					
Transfer data length	7 or 8 bits					
Transfer rate**Note**	Max. $f_{CLK}/2$ [Hz] (CSI00 only), $f_{CLK}/4$ [Hz] Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [Hz] f_{CLK}: CPU/peripheral hardware clock frequency					
Data phase	Selectable by the DAPmn bit of the SCRmn register • DAPmn = 0: Data input starts from the start of the operation of the serial clock. • DAPmn = 1: Data input starts half a clock cycle before the start of the serial clock operation.					
Clock phase	Selectable by the CKPmn bit of the SCRmn register • CKPmn = 0: Non-reverse • CKPmn = 1: Reverse					
Data direction	MSB or LSB first					

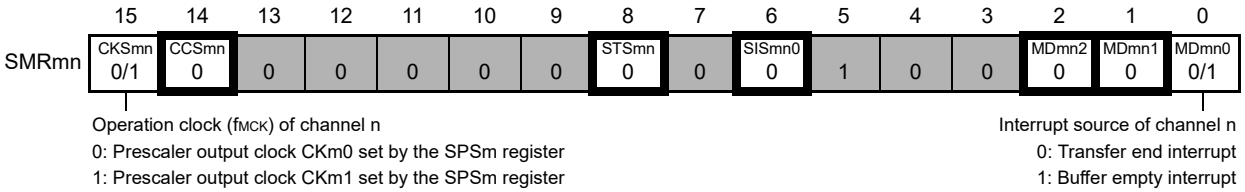
Note Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics specified in the electrical characteristics. For details, see **Section 28 Electrical Characteristics**.

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

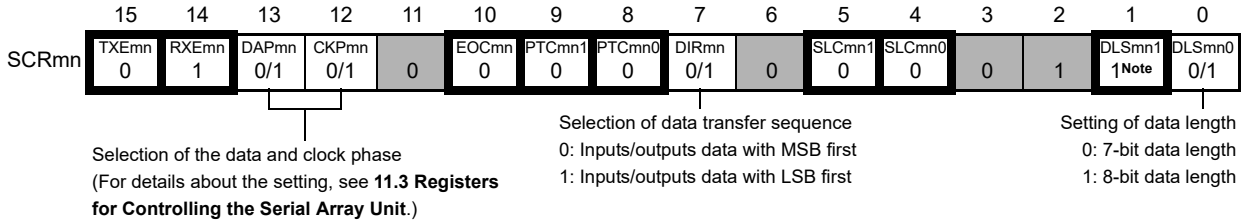
1. Register setting

Figure 11 - 34 Example of Contents of Registers for Master Reception of Simplified SPI (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21)

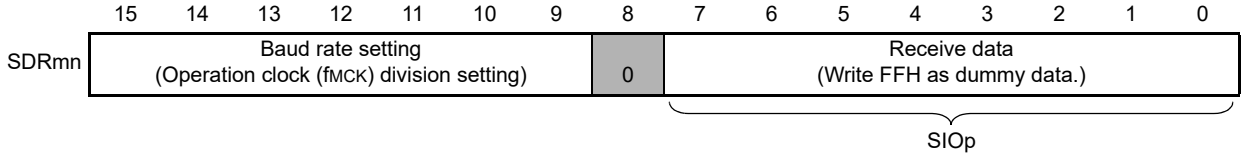
a) Serial mode register mn (SMRmn)



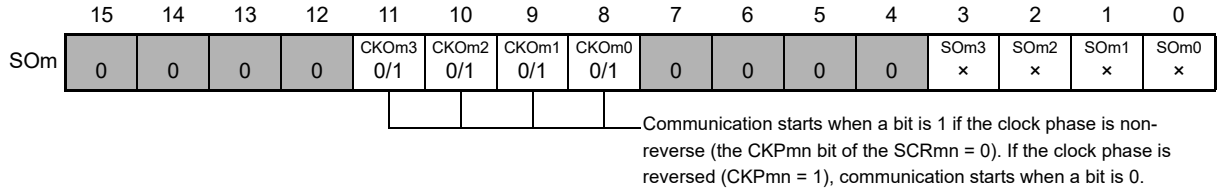
b) Serial communication operation setting register mn (SCRmn)



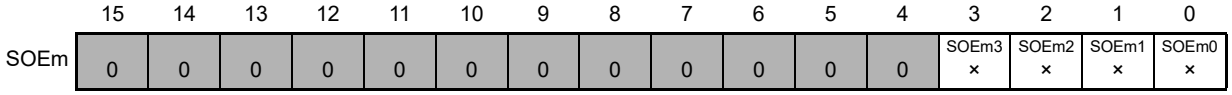
c) Serial data register mn (SDRmn) (lower 8 bits: SIOp)



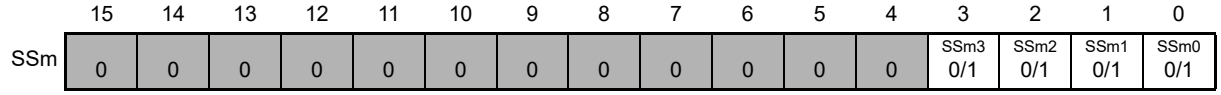
d) Serial output register m (SOm): Set only the bit of the target channel.



e) Serial output enable register m (SOEm): This register is not used in this mode.



f) Serial channel start register m (SSm): Set only the bit of the target channel to 1.



Note This bit is only present in the SCR00 and SCR01 registers, and is fixed to 1 for the other registers.

(Remarks are listed on the next page.)

Remark 1. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

Remark 2. ■: Setting is fixed in the simplified SPI (CSI) master reception mode, ■: Setting disabled (set to the initial value)
×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)
0/1: Set to 0 or 1 depending on the user's application.

2. Operation procedure

Figure 11 - 35 Initial Setting Procedure for Master Reception

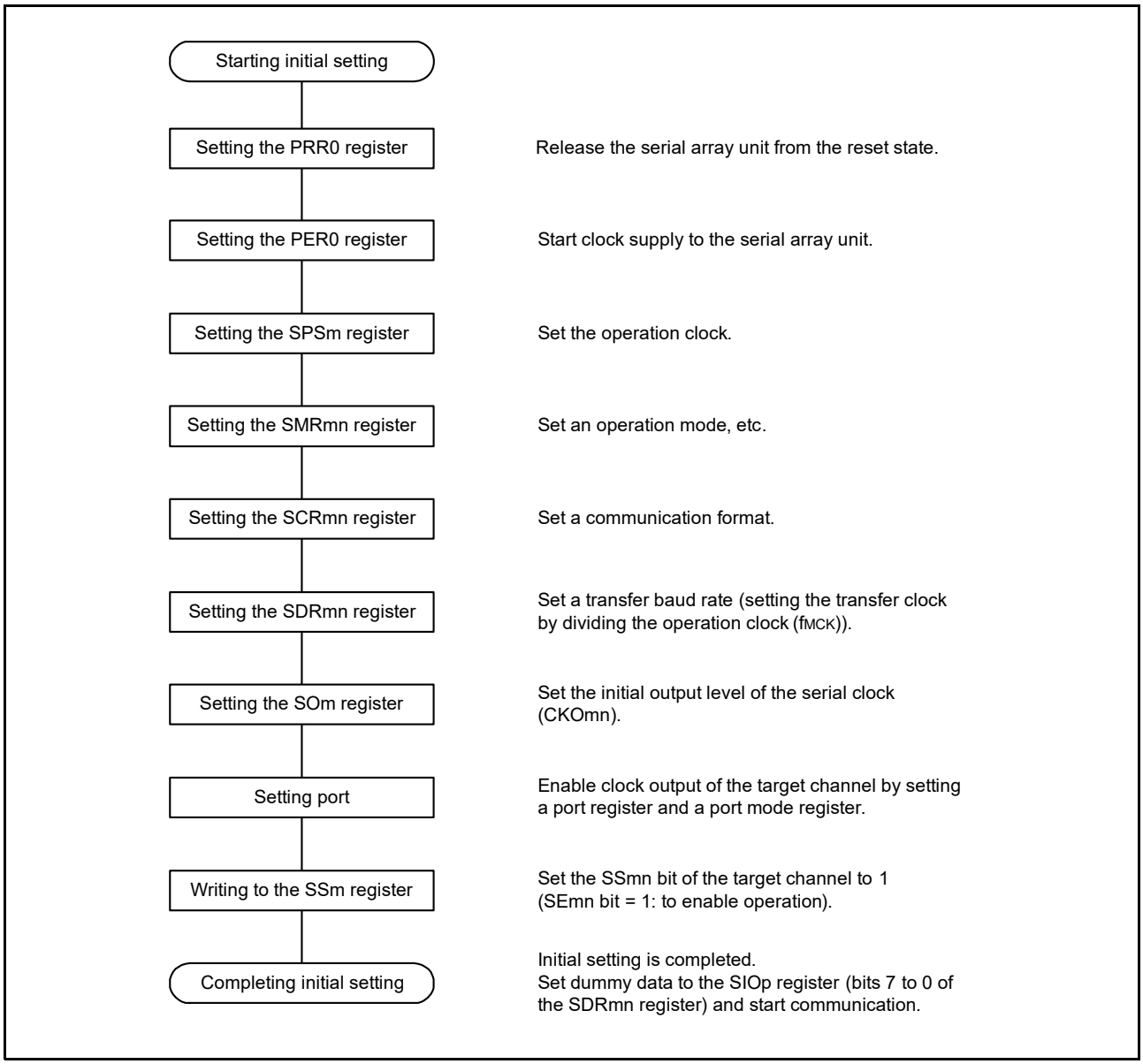


Figure 11 - 36 Procedure for Stopping Master Reception

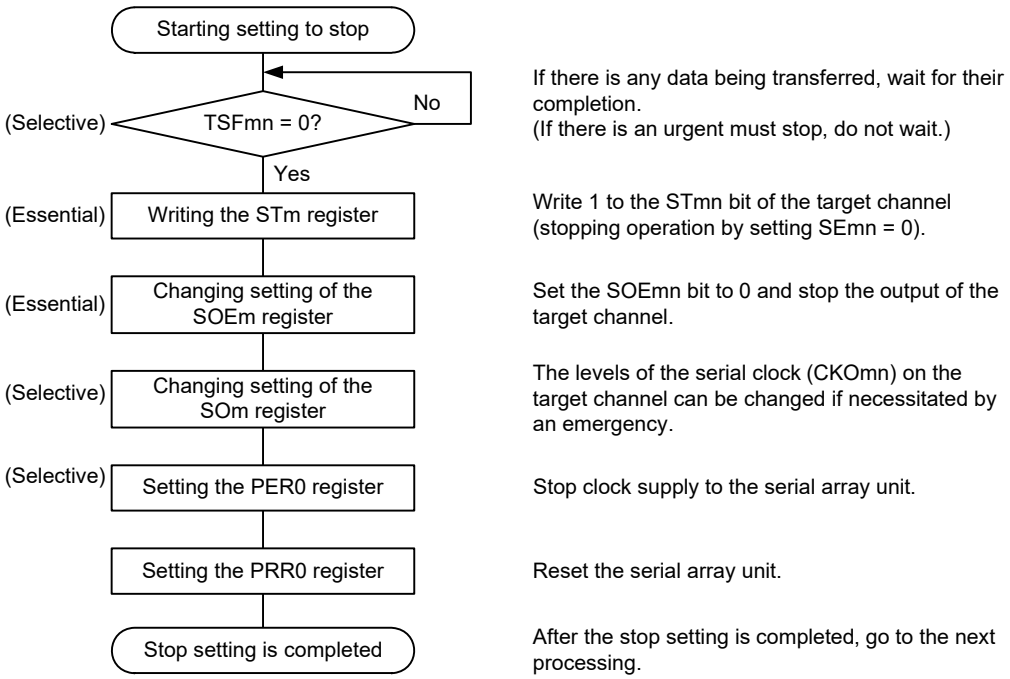
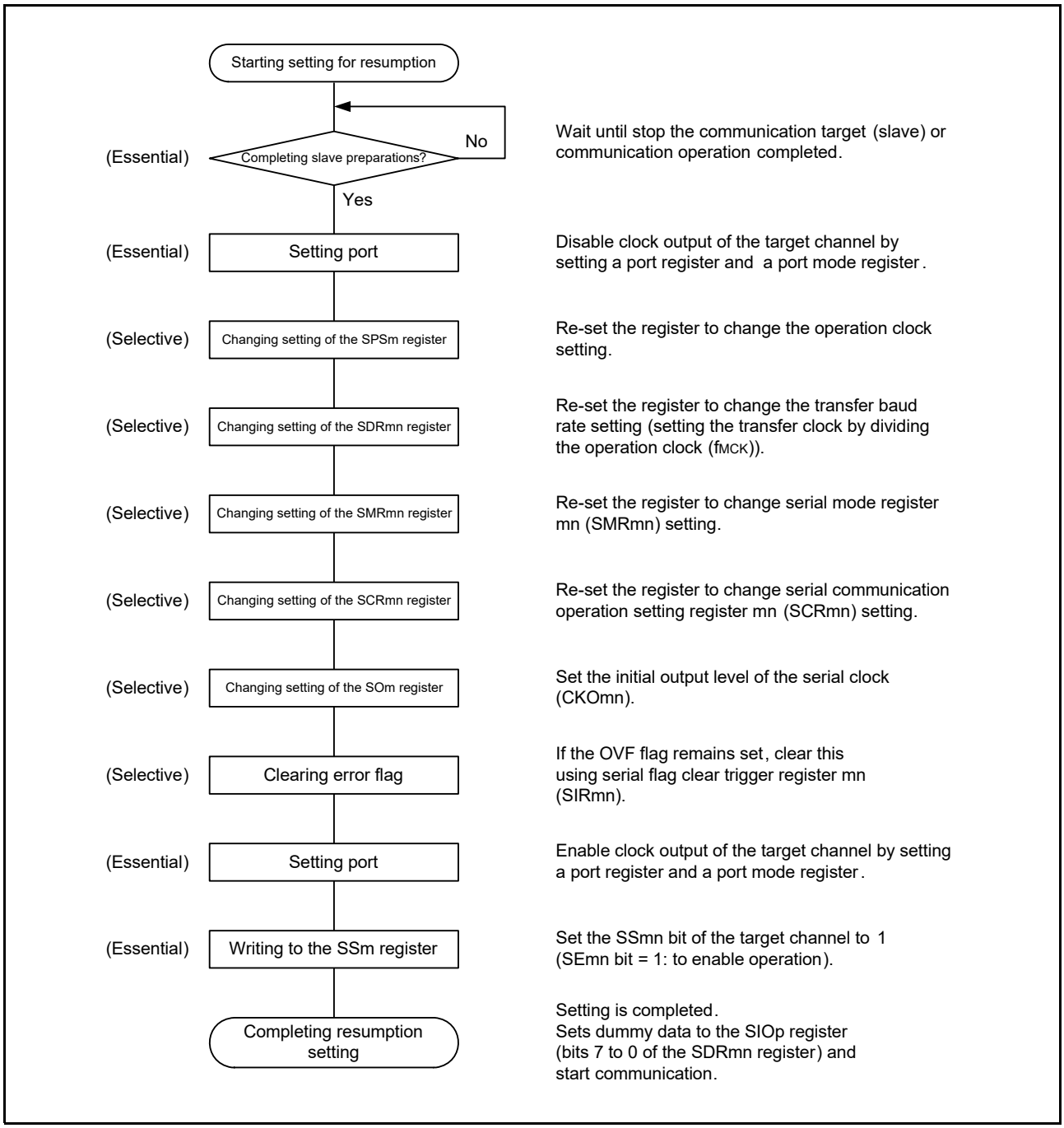


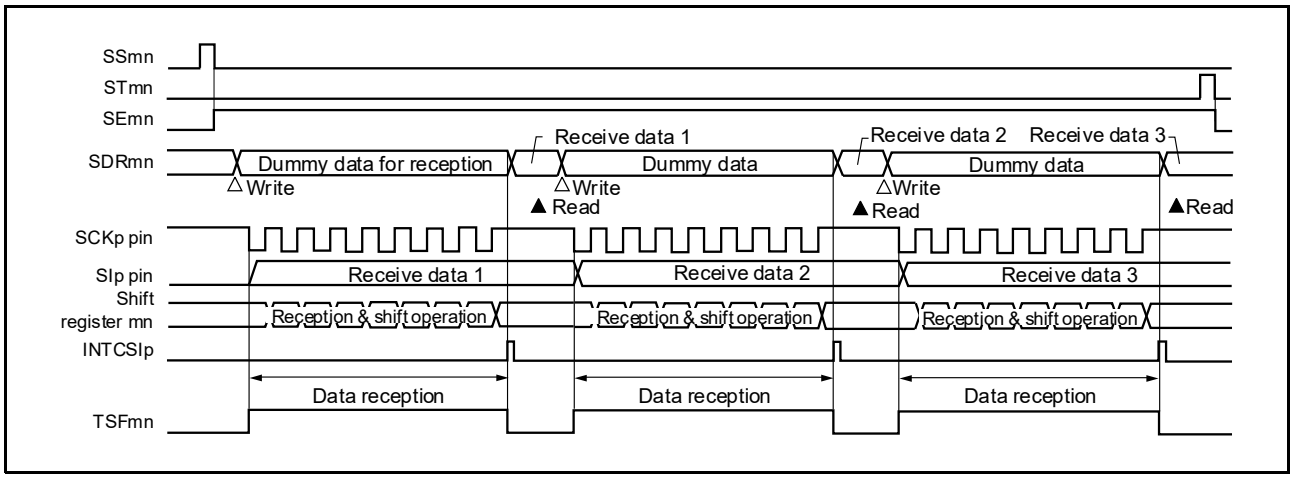
Figure 11 - 37 Procedure for Resuming Master Reception



Remark If the PRR0 register is rewritten while stopping the communication to reset the serial array unit, wait until the communication target (slave) stops or communication finishes, and then perform initialization instead of restarting the communication.

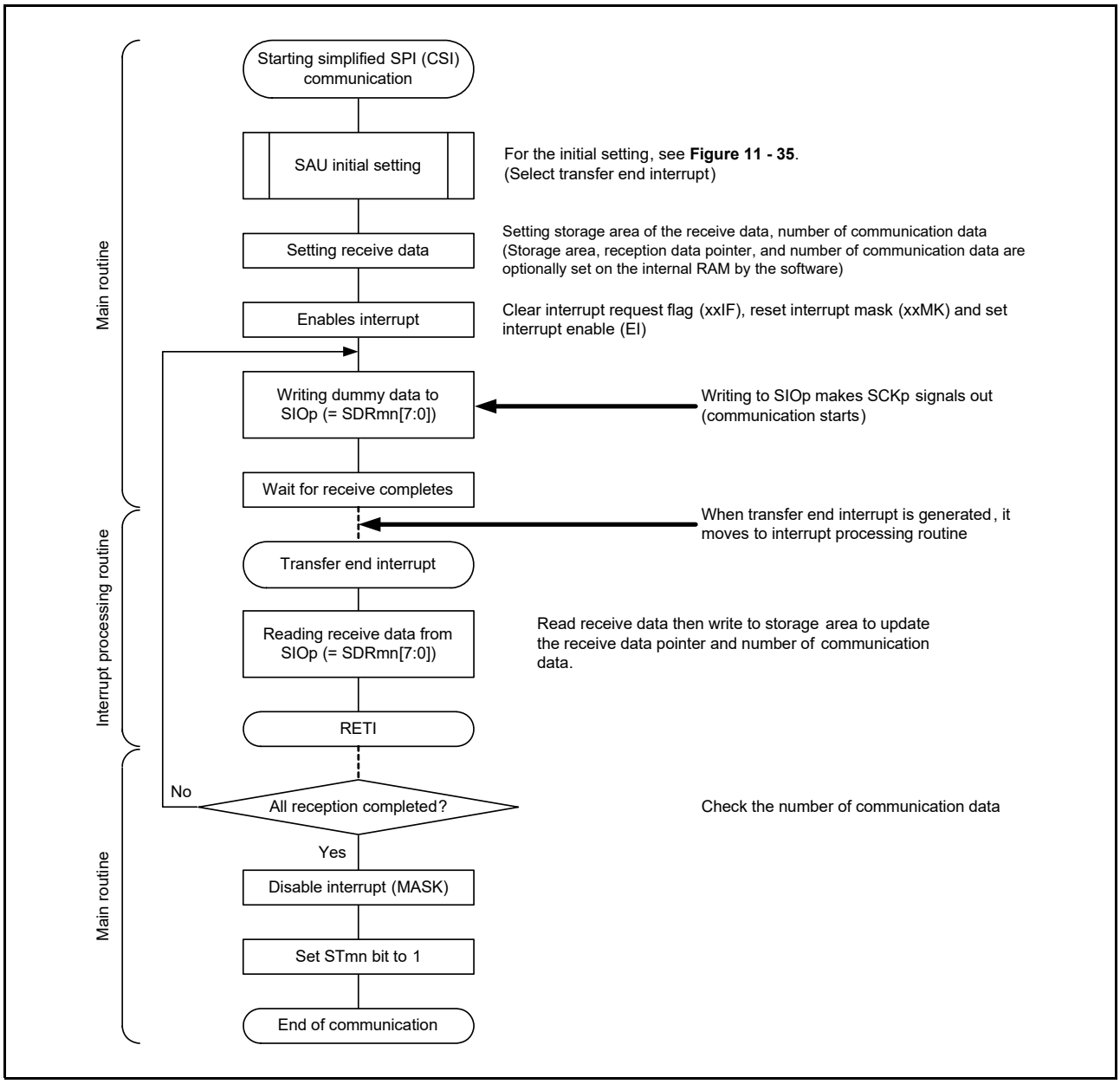
3. Processing flow (in single-reception mode)

Figure 11 - 38 Timing Chart of Master Reception (in Single-reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)



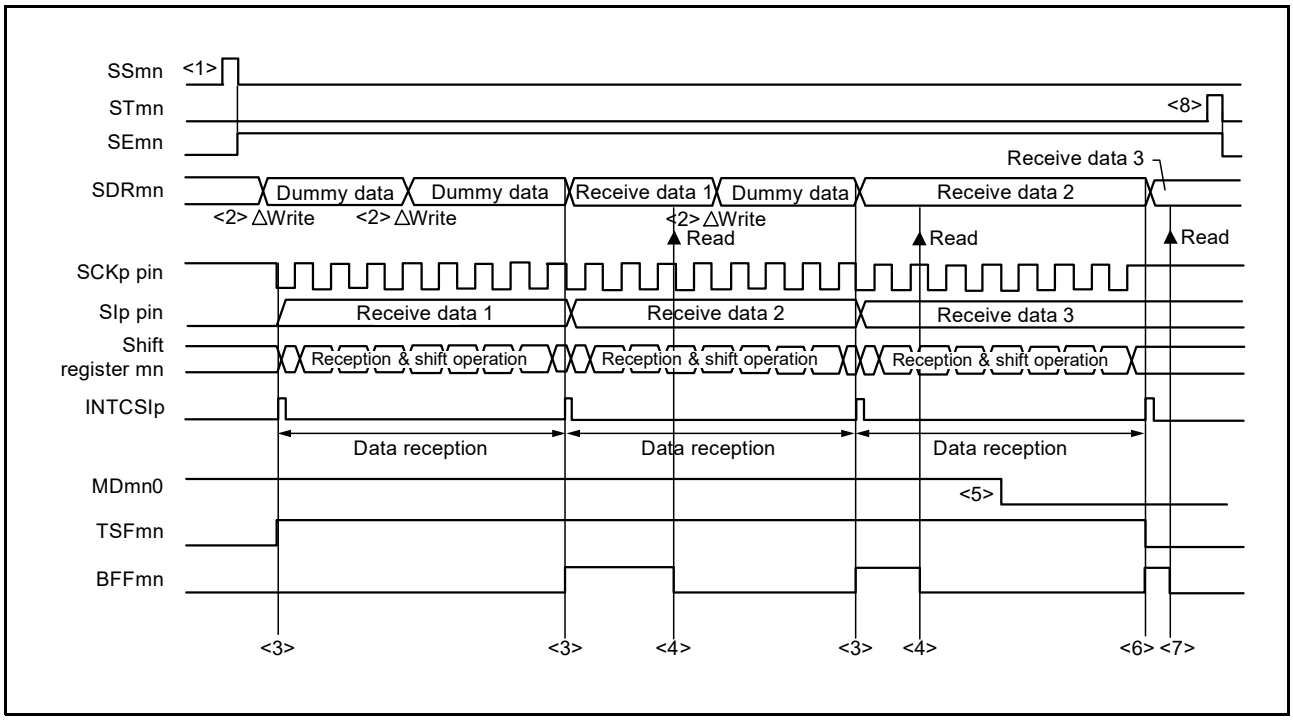
Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

Figure 11 - 39 Flowchart of Master Reception (in Single-reception Mode)



4. Processing flow (in continuous reception mode)

Figure 11 - 40 Timing Chart of Master Reception (in Continuous Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)

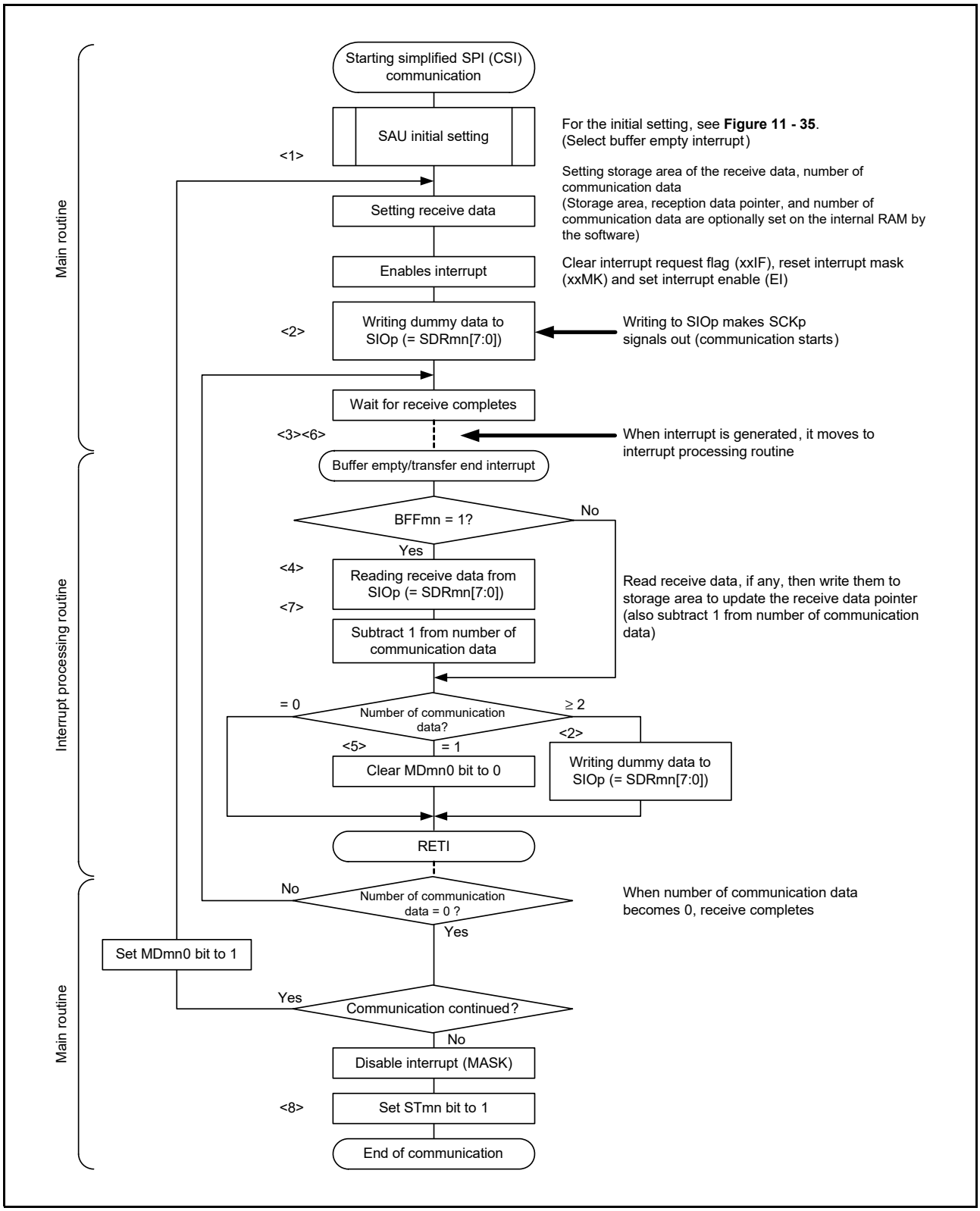


Caution **The MDmn0 bit can be rewritten even during operation. However, rewrite it before receive of the last bit is started, so that it has been rewritten before the transfer end interrupt of the last receive data.**

Remark 1. <1> to <8> in the figure correspond to <1> to <8> in **Figure 11 - 41 Flowchart of Master Reception (in Continuous Reception Mode)**.

Remark 2. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

Figure 11 - 41 Flowchart of Master Reception (in Continuous Reception Mode)



Remark <1> to <8> in the figure correspond to <1> to <8> in **Figure 11 - 40 Timing Chart of Master Reception (in Continuous Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)**.

11.5.3 Master transmission/reception

Master transmission/reception is that this MCU outputs a transfer clock and transmits/receives data to/from other device.

Simplified SPI	CSI00	CSI01	CSI10	CSI11	CSI20	CSI21
Target channel	Channel 0 of SAU0	Channel 1 of SAU0	Channel 2 of SAU0	Channel 3 of SAU0	Channel 0 of SAU1	Channel 1 of SAU1
Pins used	SCK00, SI00, SO00	SCK01, SI01, SO01	SCK10, SI10, SO10	SCK11, SI11, SO11	SCK20, SI20, SO20	SCK21, SI21, SO21
Interrupt	INTCSI00	INTCSI01	INTCSI10	INTCSI11	INTCSI20	INTCSI21
	Transfer end interrupt (in single-transfer mode) or buffer empty interrupt (in continuous transfer mode) can be selected.					
Error detection flag	Overrun error detection flag (OVFmn) only					
Transfer data length	7 or 8 bits					
Transfer rate**Note**	Max. $f_{CLK}/2$ [Hz] (CSI00 only), $f_{CLK}/4$ [Hz] Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [Hz] f_{CLK}: CPU/peripheral hardware clock frequency					
Data phase	Selectable by the DAPmn bit of the SCRmn register • DAPmn = 0: Data I/O starts at the start of the operation of the serial clock. • DAPmn = 1: Data I/O starts half a clock cycle before the start of the serial clock operation.					
Clock phase	Selectable by the CKPmn bit of the SCRmn register • CKPmn = 0: Non-reverse • CKPmn = 1: Reverse					
Data direction	MSB or LSB first					

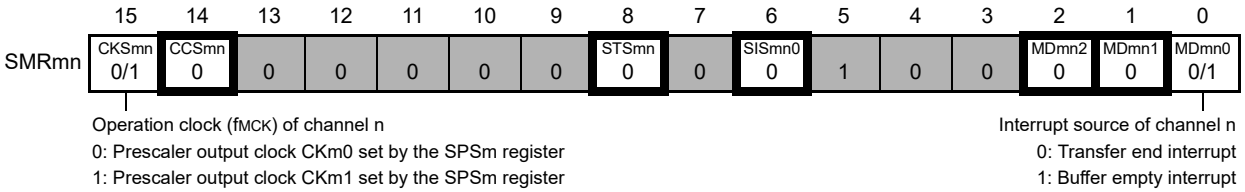
Note Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics specified in the electrical characteristics. For details, see **Section 28 Electrical Characteristics**.

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 13

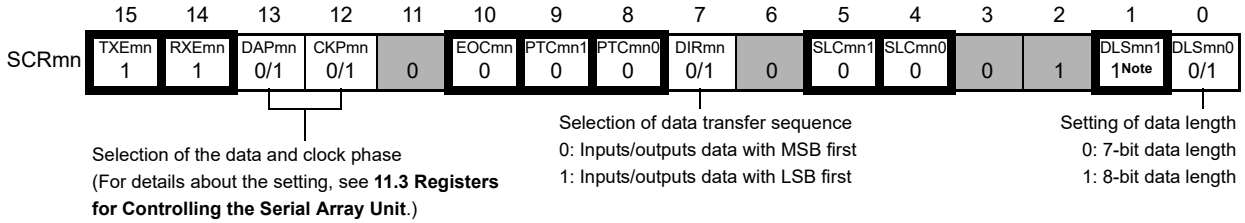
1. Register setting

Figure 11 - 42 Example of Contents of Registers for Master Transmission/Reception of Simplified SPI (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21)

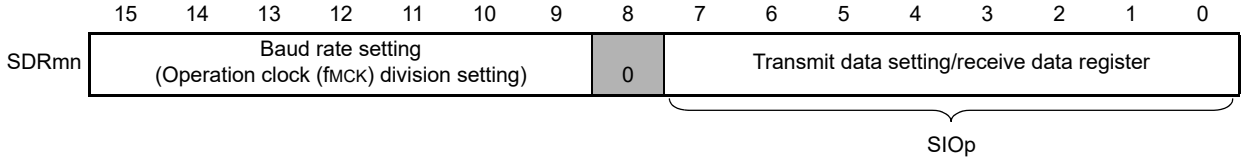
a) Serial mode register mn (SMRmn)



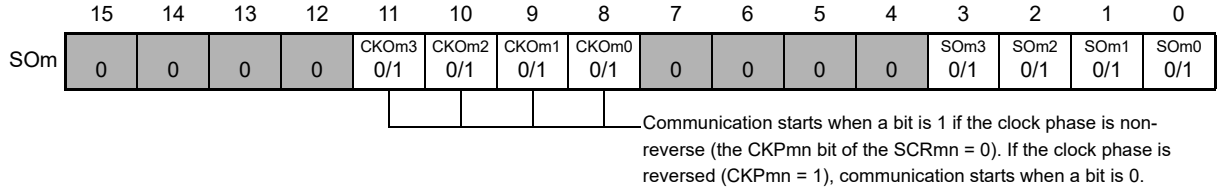
b) Serial communication operation setting register mn (SCRmn)



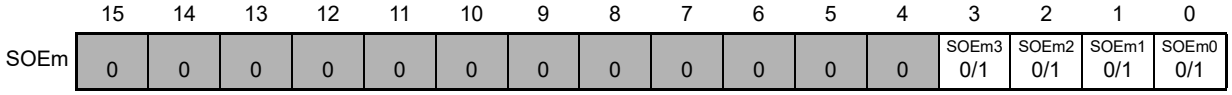
c) Serial data register mn (SDRmn) (lower 8 bits: SIOp)



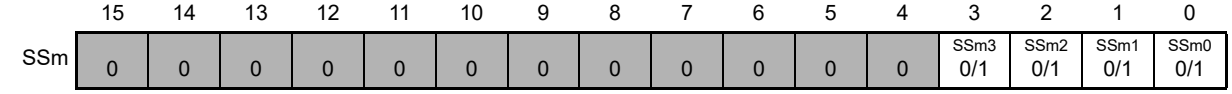
d) Serial output register m (SOm): Set only the bit of the target channel.



e) Serial output enable register m (SOEm): Set only the bit of the target channel to 1.



f) Serial channel start register m (SSm): Set only the bit of the target channel to 1.



Note This bit is only present in the SCR00 and SCR01 registers, and is fixed to 1 for the other registers.

(Remarks are listed on the next page.)

Remark 1. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

Remark 2. ■: Setting is fixed in the simplified SPI (CSI) master transmission/reception mode, ■: Setting disabled (set to the initial value)
0/1: Set to 0 or 1 depending on the user's application.

2. Operation procedure

Figure 11 - 43 Initial Setting Procedure for Master Transmission/Reception

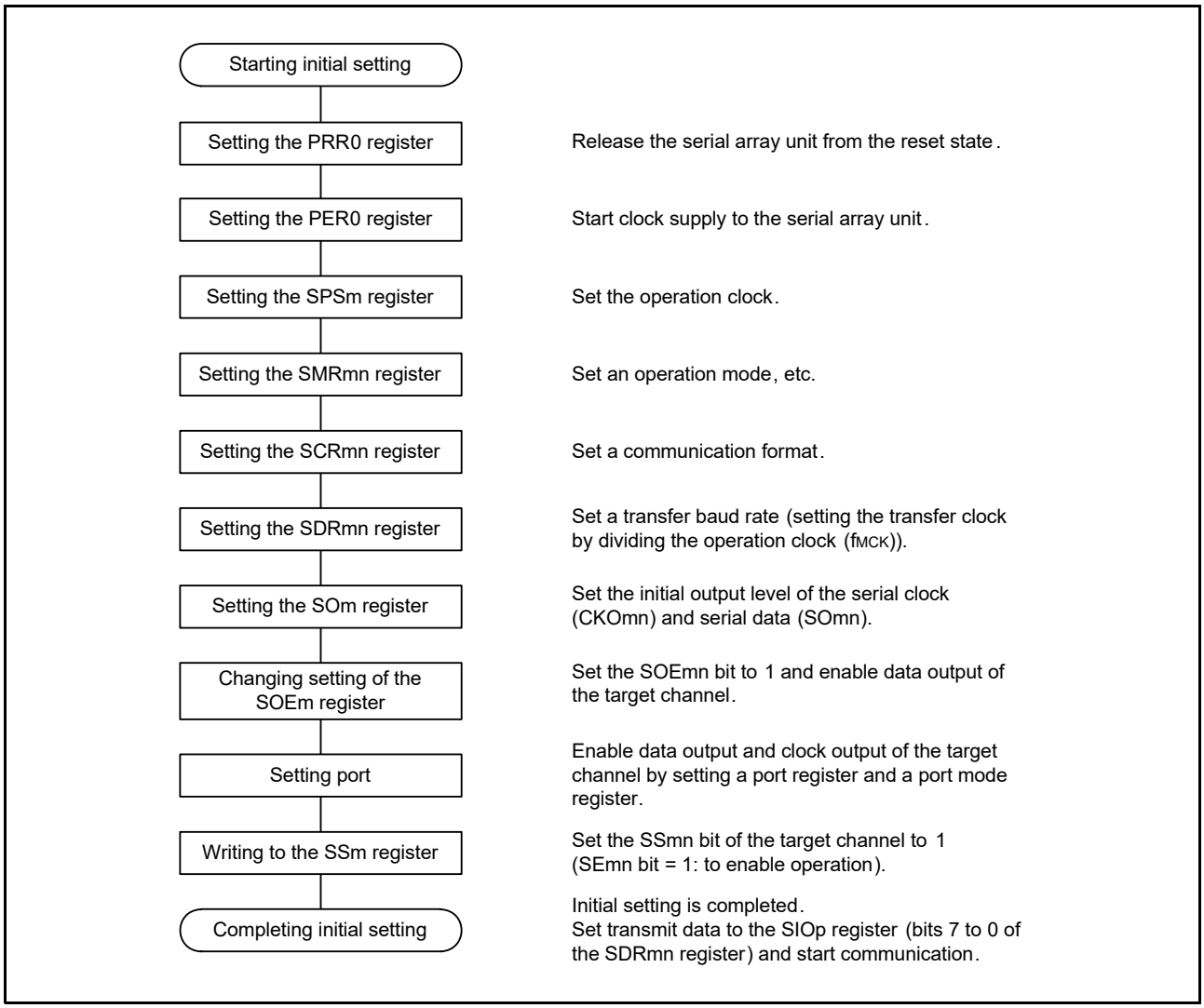


Figure 11 - 44 Procedure for Stopping Master Transmission/Reception

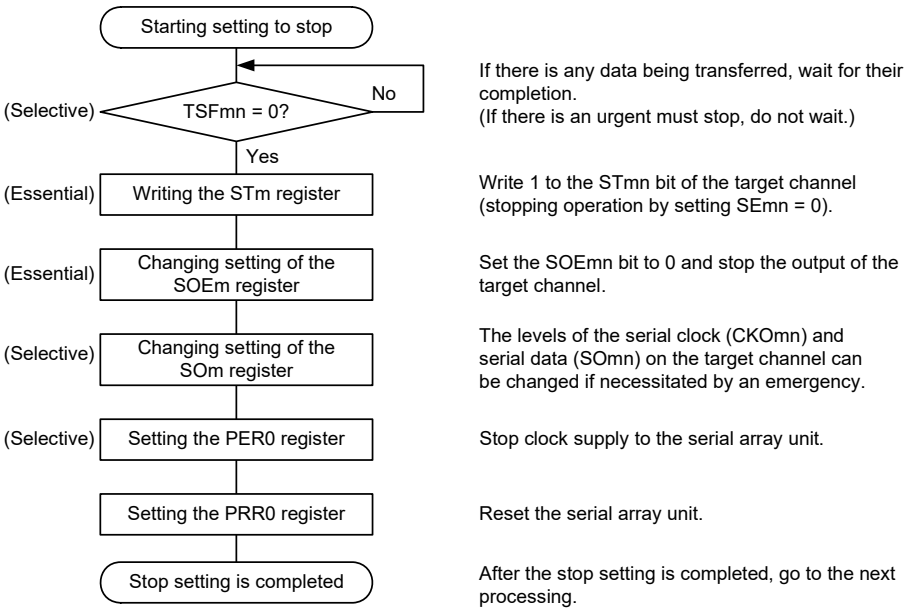
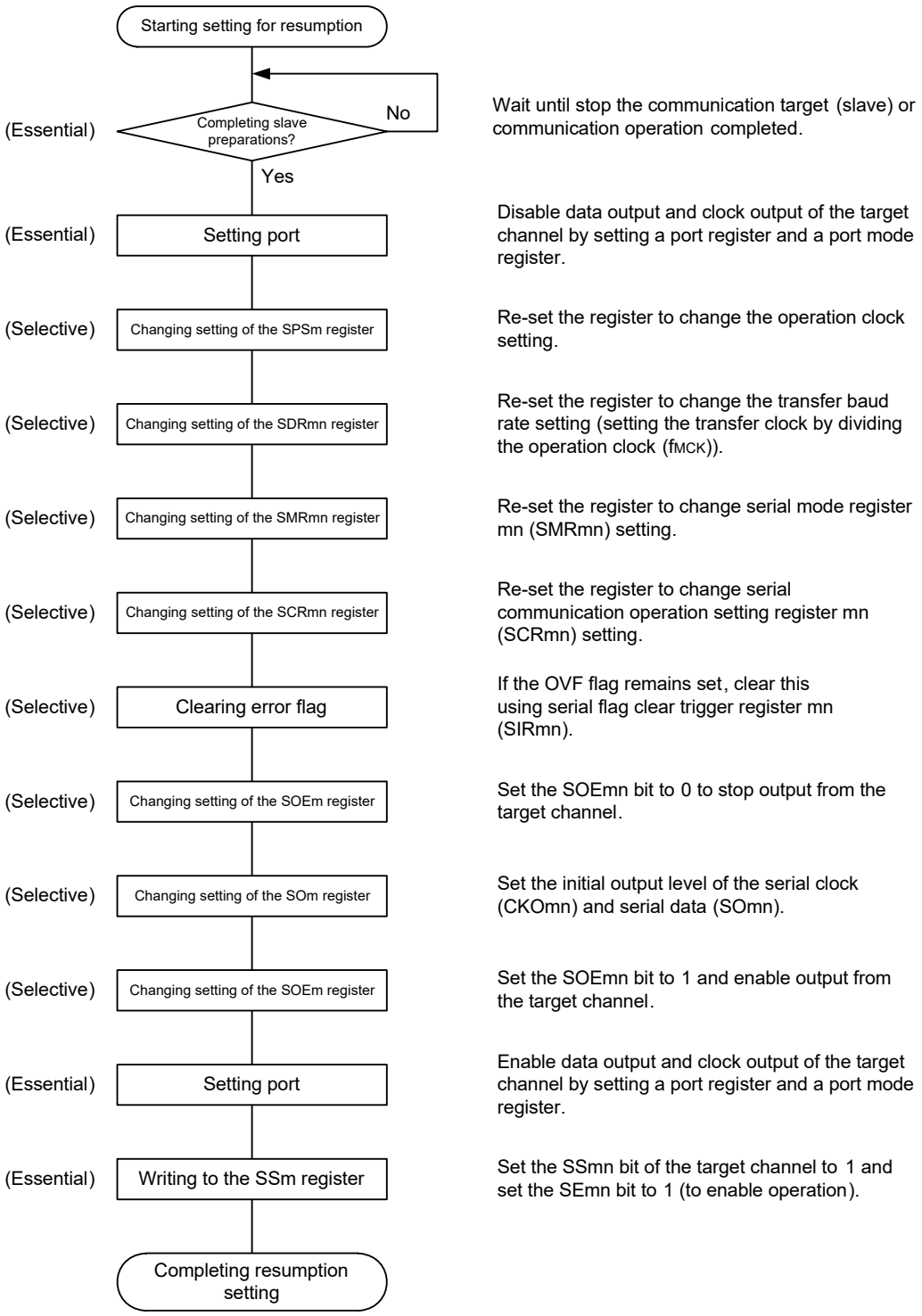
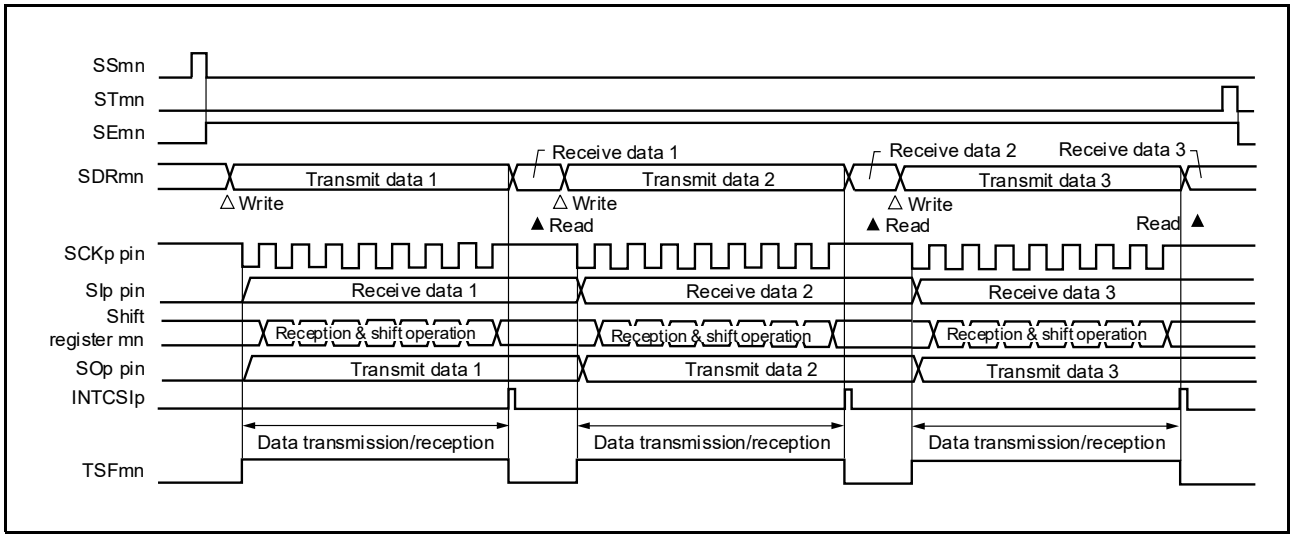


Figure 11 - 45 Procedure for Resuming Master Transmission/Reception



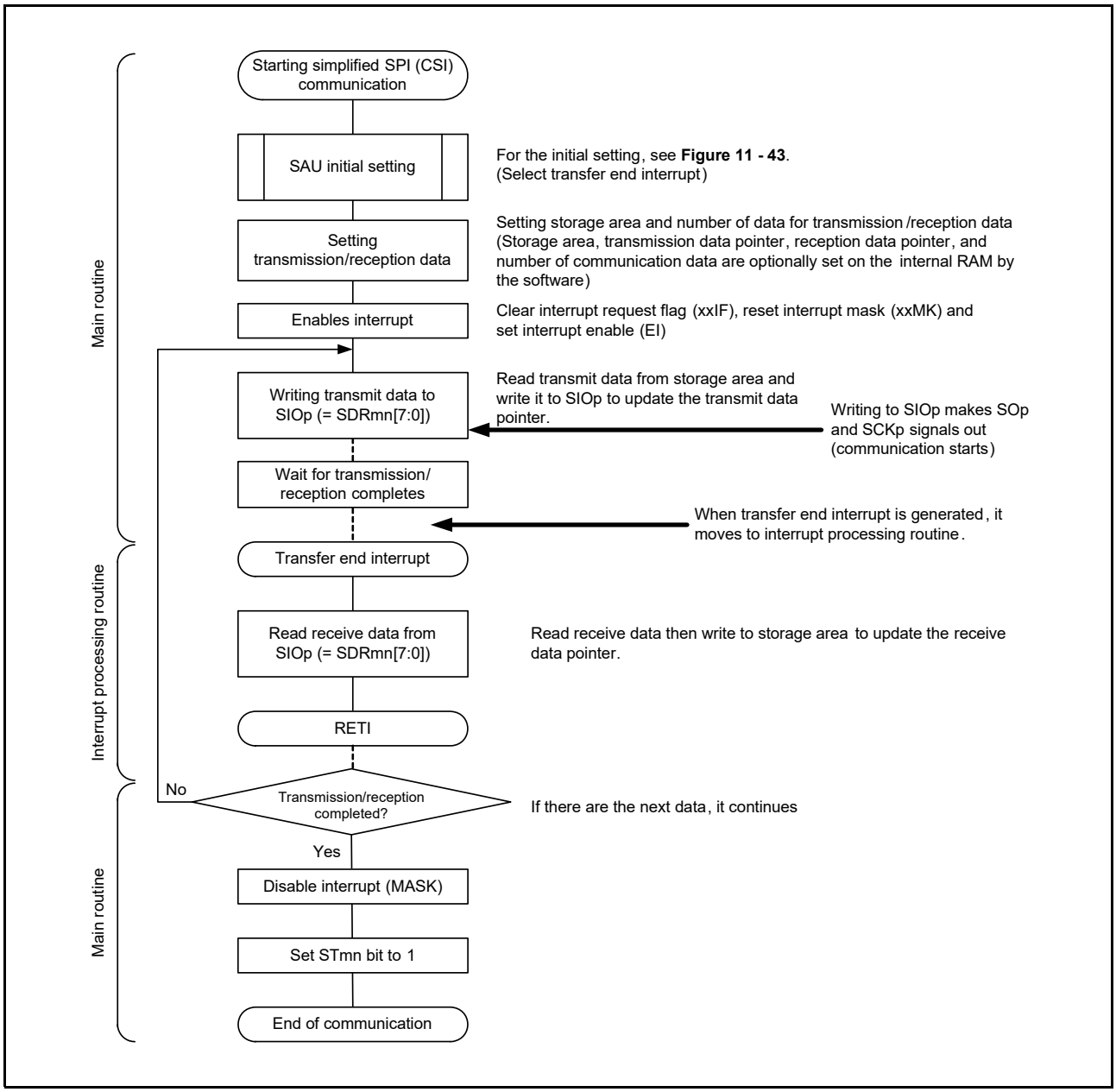
3. Processing flow (in single-transmission/reception mode)

Figure 11 - 46 Timing Chart of Master Transmission/Reception (in Single-transmission/reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)



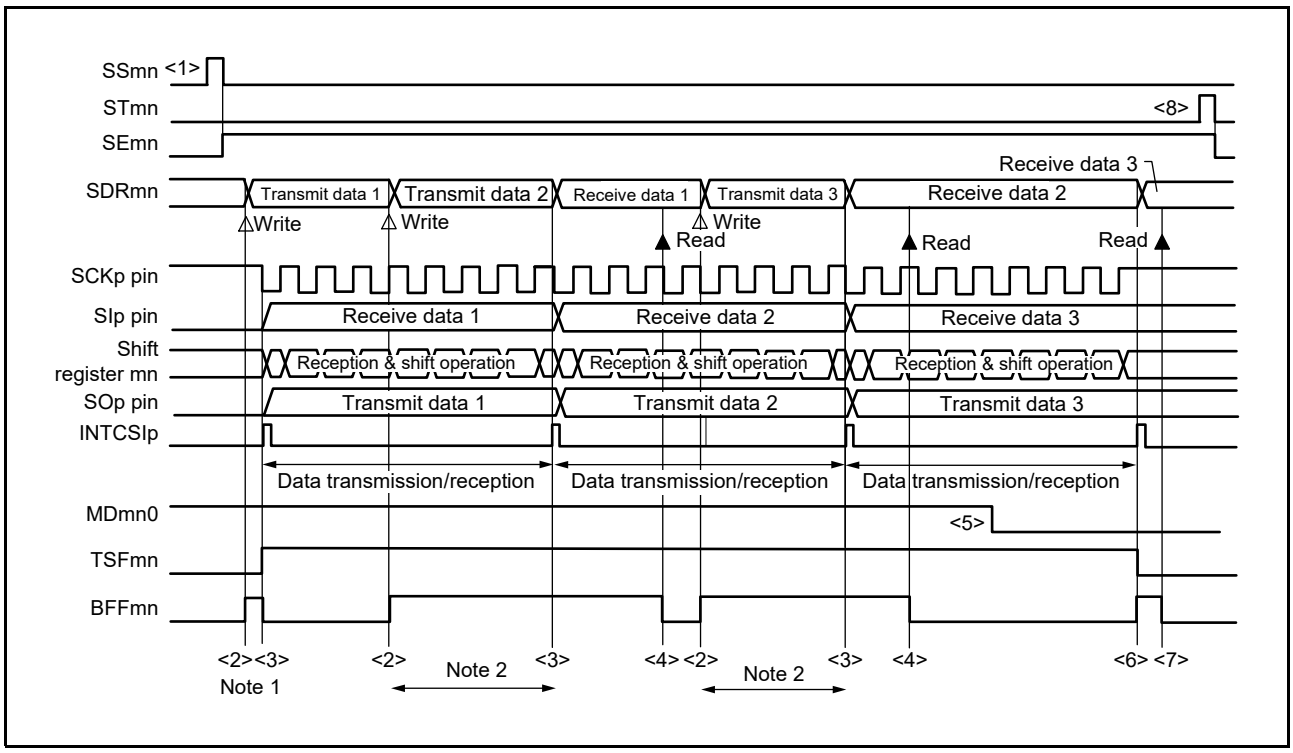
Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

Figure 11 - 47 Flowchart of Master Transmission/Reception (in Single-transmission/reception Mode)



4. Processing flow (in continuous transmission/reception mode)

Figure 11 - 48 Timing Chart of Master Transmission/Reception (in Continuous Transmission/Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)



Note 1. If transmit data is written to the SDRmn register while the BFFmn bit of serial status register mn (SSRmn) is 1 (valid data is stored in serial data register mn (SDRmn)), the transmit data is overwritten.

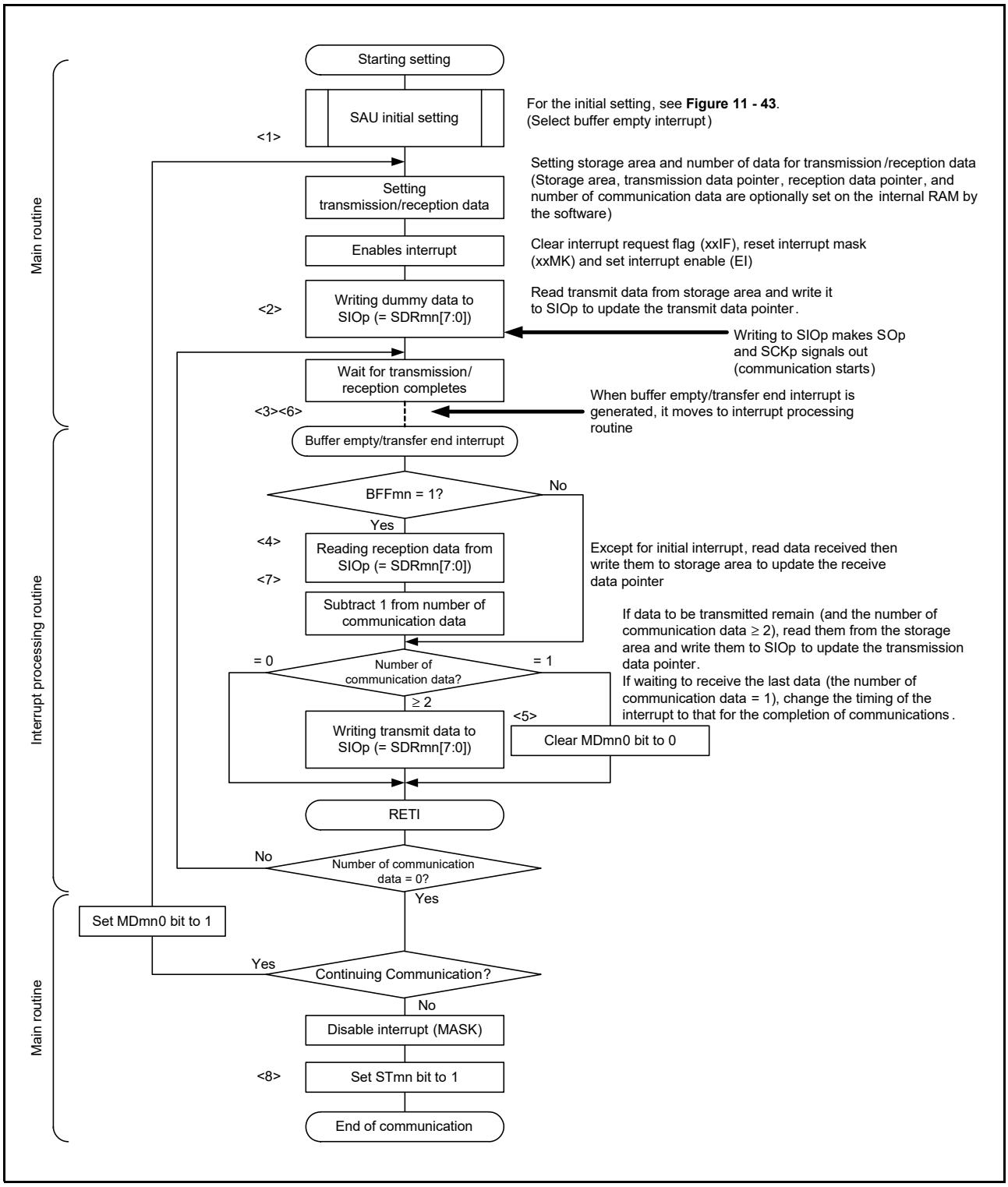
Note 2. The transmit data can be read by reading the SDRmn register during this period. Reading this register does not affect the transfer operation.

Caution **The MDmn0 bit of serial mode register mn (SSRmn) can be rewritten even during operation. However, rewrite it before transfer of the last bit is started, so that it has been rewritten before the transfer end interrupt of the last transmit data.**

Remark 1. <1> to <8> in the figure correspond to <1> to <8> in **Figure 11 - 49 Flowchart of Master Transmission/Reception (in Continuous Transmission/Reception Mode)**.

Remark 2. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

Figure 11 - 49 Flowchart of Master Transmission/Reception (in Continuous Transmission/Reception Mode)



Remark <1> to <8> in the figure correspond to <1> to <8> in **Figure 11 - 48 Timing Chart of Master Transmission/Reception (in Continuous Transmission/Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)**.

11.5.4 Slave transmission

Slave transmission is that this MCU transmits data to another device in the state of a transfer clock being input from another device.

Simplified SPI	CSI00	CSI01	CSI10	CSI11	CSI20	CSI21
Target channel	Channel 0 of SAU0	Channel 1 of SAU0	Channel 2 of SAU0	Channel 3 of SAU0	Channel 0 of SAU1	Channel 1 of SAU1
Pins used	SCK00, SO00	SCK01, SO01	SCK10, SO10	SCK11, SO11	SCK20, SO20	SCK21, SO21
Interrupt	INTCSI00	INTCSI01	INTCSI10	INTCSI11	INTCSI20	INTCSI21
	Transfer end interrupt (in single-transfer mode) or buffer empty interrupt (in continuous transfer mode) can be selected.					
Error detection flag	Overrun error detection flag (OVFmn) only					
Transfer data length	7 or 8 bits					
Transfer rate	Max. f_{MCK}/6 [Hz]**Notes 1, 2**					
Data phase	Selectable by the DAPmn bit of the SCRmn register • DAPmn = 0: Data output starts from the start of the operation of the serial clock. • DAPmn = 1: Data output starts half a clock cycle before the start of the serial clock operation.					
Clock phase	Selectable by the CKPmn bit of the SCRmn register • CKPmn = 0: Non-reverse • CKPmn = 1: Reverse					
Data direction	MSB or LSB first					

Note 1. Because the external serial clock input to the SCK00, SCK01, SCK10, SCK11, SCK20, and SCK21 pins is sampled internally and used, the fastest transfer rate is f_{MCK}/6 [Hz].

Note 2. Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics specified in the electrical characteristics. For details, see **Section 28 Electrical Characteristics**.

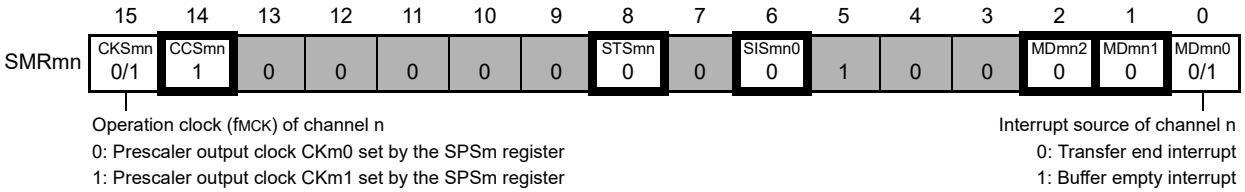
Remark 1. f_{MCK}: Operation clock frequency of target channel
f_{SCK}: Serial clock frequency

Remark 2. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

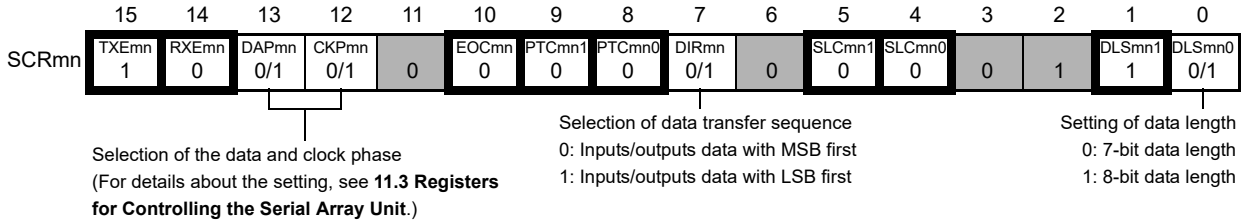
1. Register setting

Figure 11 - 50 Example of Contents of Registers for Slave Transmission of Simplified SPI (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21)

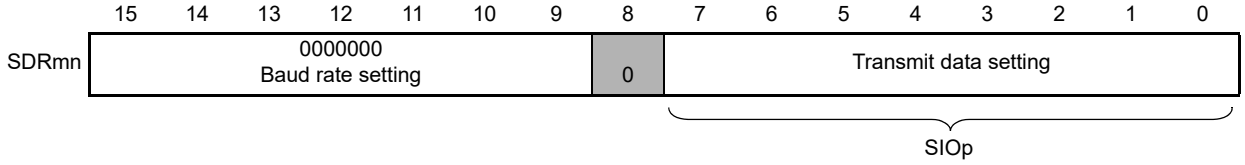
a) Serial mode register mn (SMRmn)



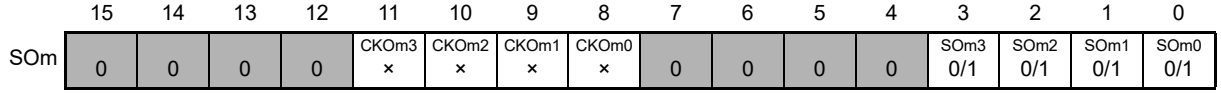
b) Serial communication operation setting register mn (SCRmn)



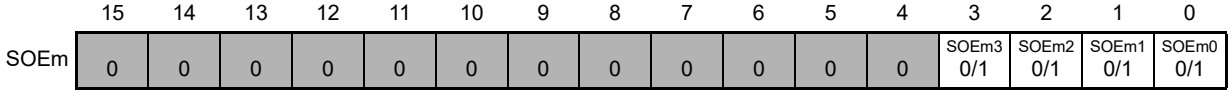
c) Serial data register mn (SDRmn) (lower 8 bits: SIOp)



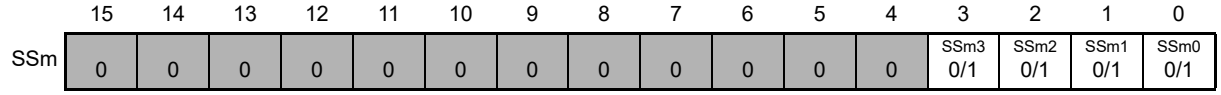
d) Serial output register m (SOm): Set only the bit of the target channel.



e) Serial output enable register m (SOEm): Set only the bit of the target channel to 1.



f) Serial channel start register m (SSm): Set only the bit of the target channel to 1.



Remark 1. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

Remark 2. : Setting is fixed in the simplified SPI (CSI) slave transmission mode, : Setting disabled (set to the initial value)
×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)
0/1: Set to 0 or 1 depending on the user's application.

2. Operation procedure

Figure 11 - 51 Initial Setting Procedure for Slave Transmission

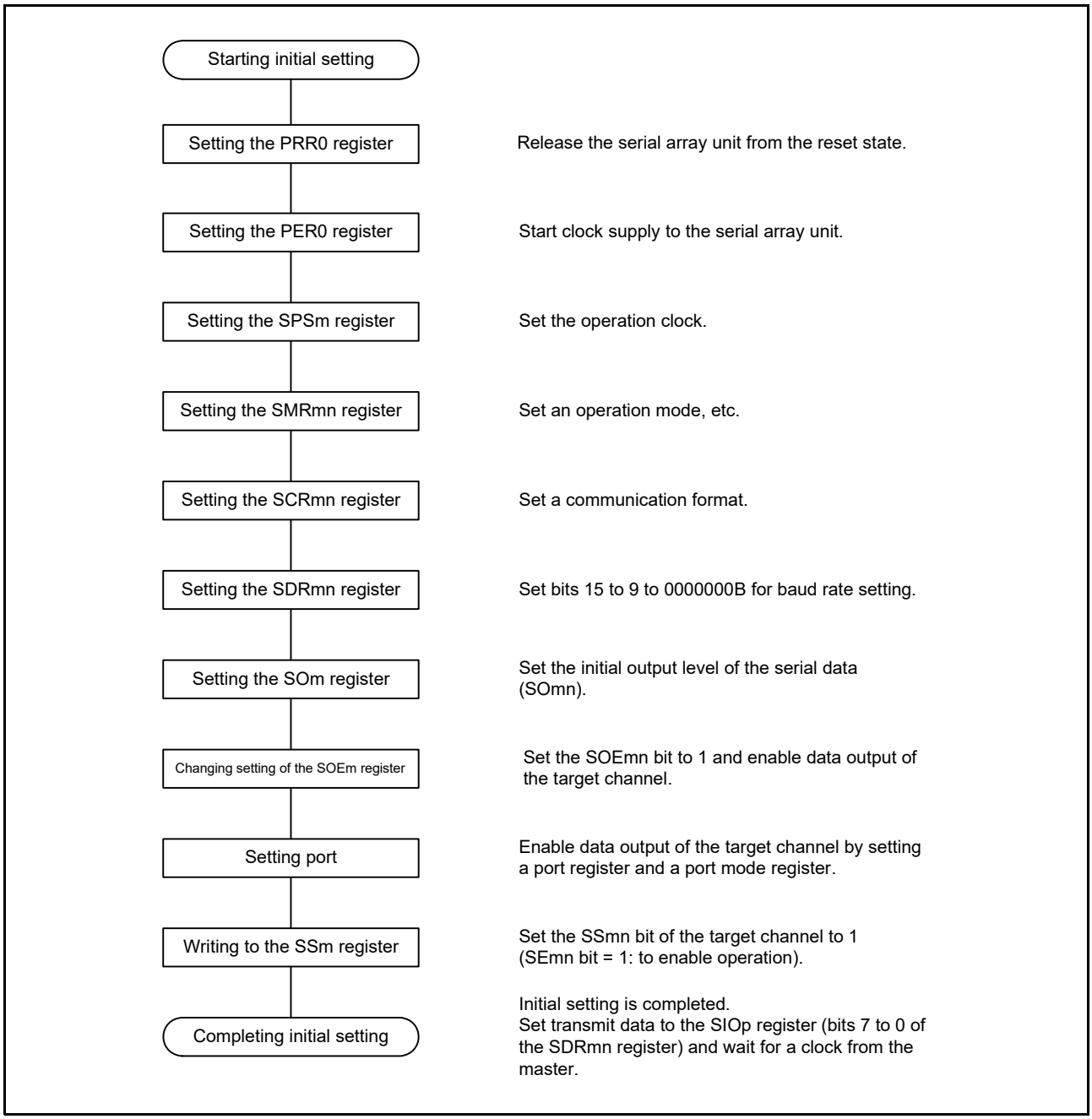


Figure 11 - 52 Procedure for Stopping Slave Transmission

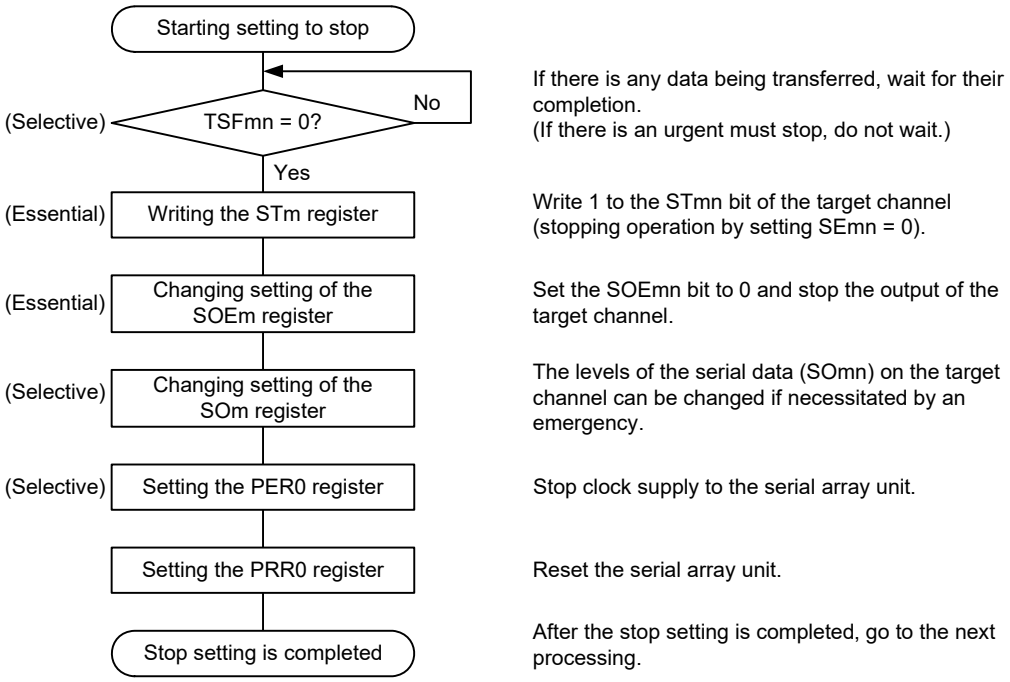
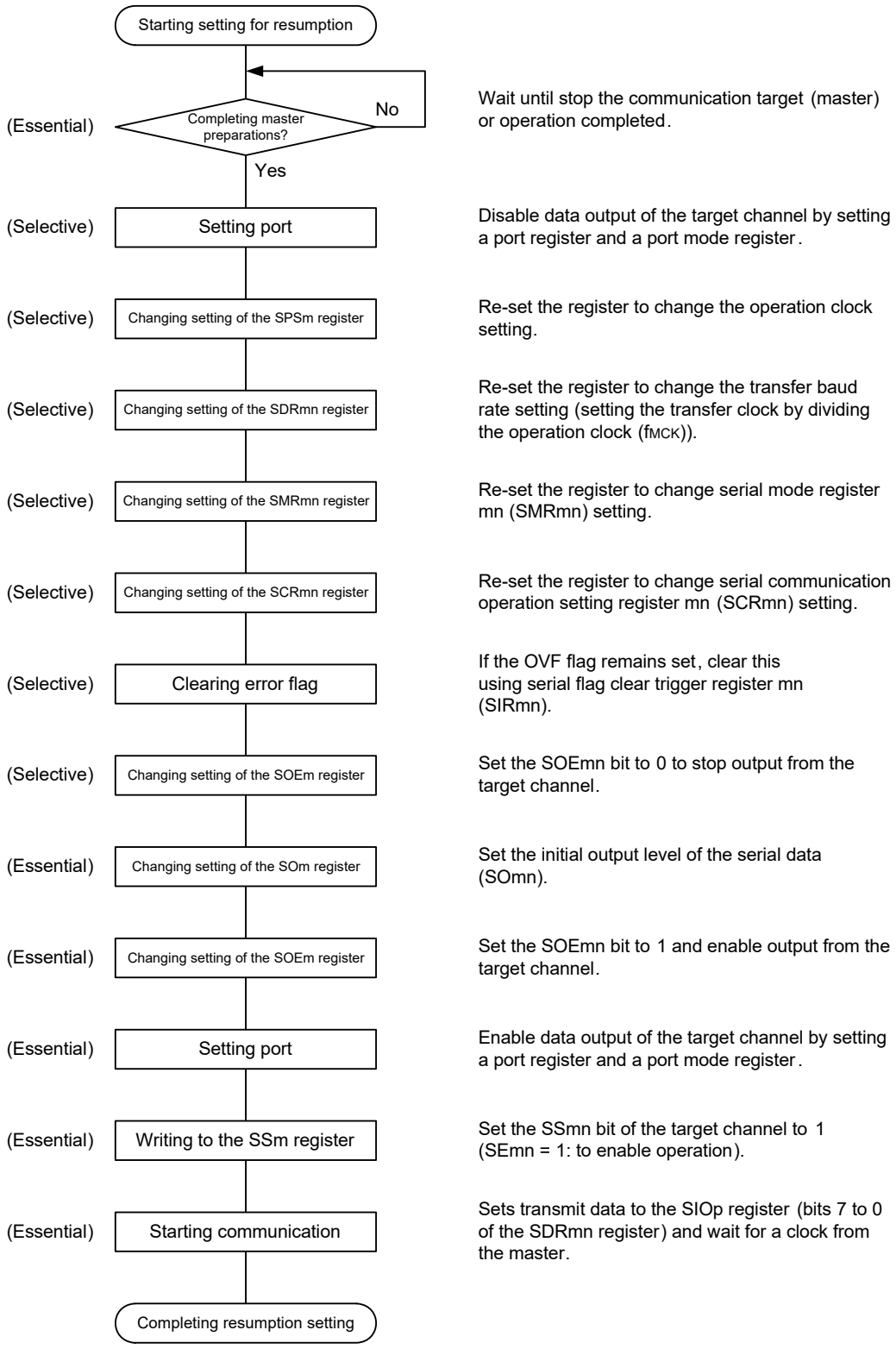


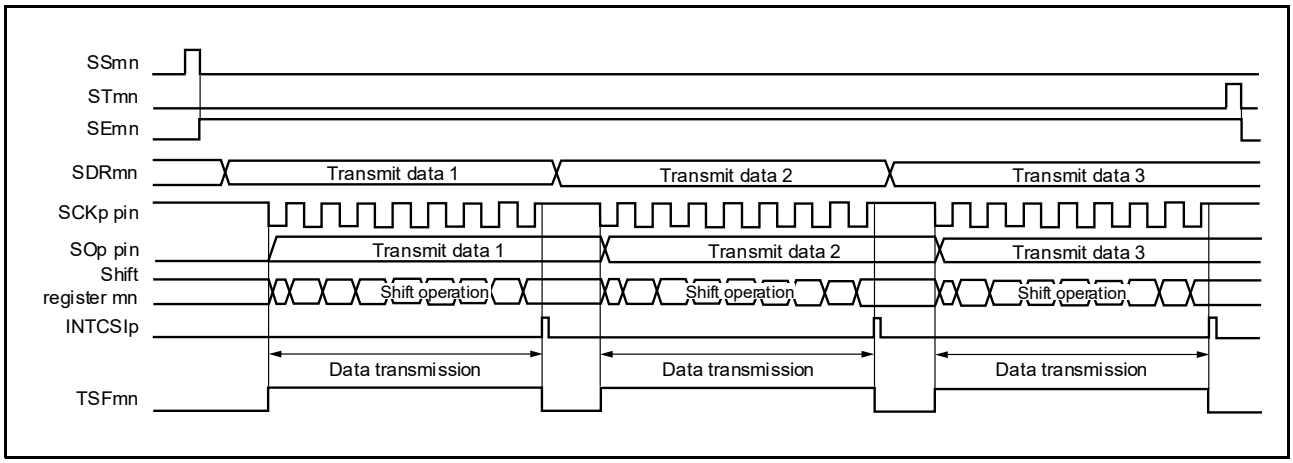
Figure 11 - 53 Procedure for Resuming Slave Transmission



Remark If the PRR0 register is rewritten while stopping the communication to reset the serial array unit, wait until the communication target (master) stops or communication finishes, and then perform initialization instead of restarting the communication.

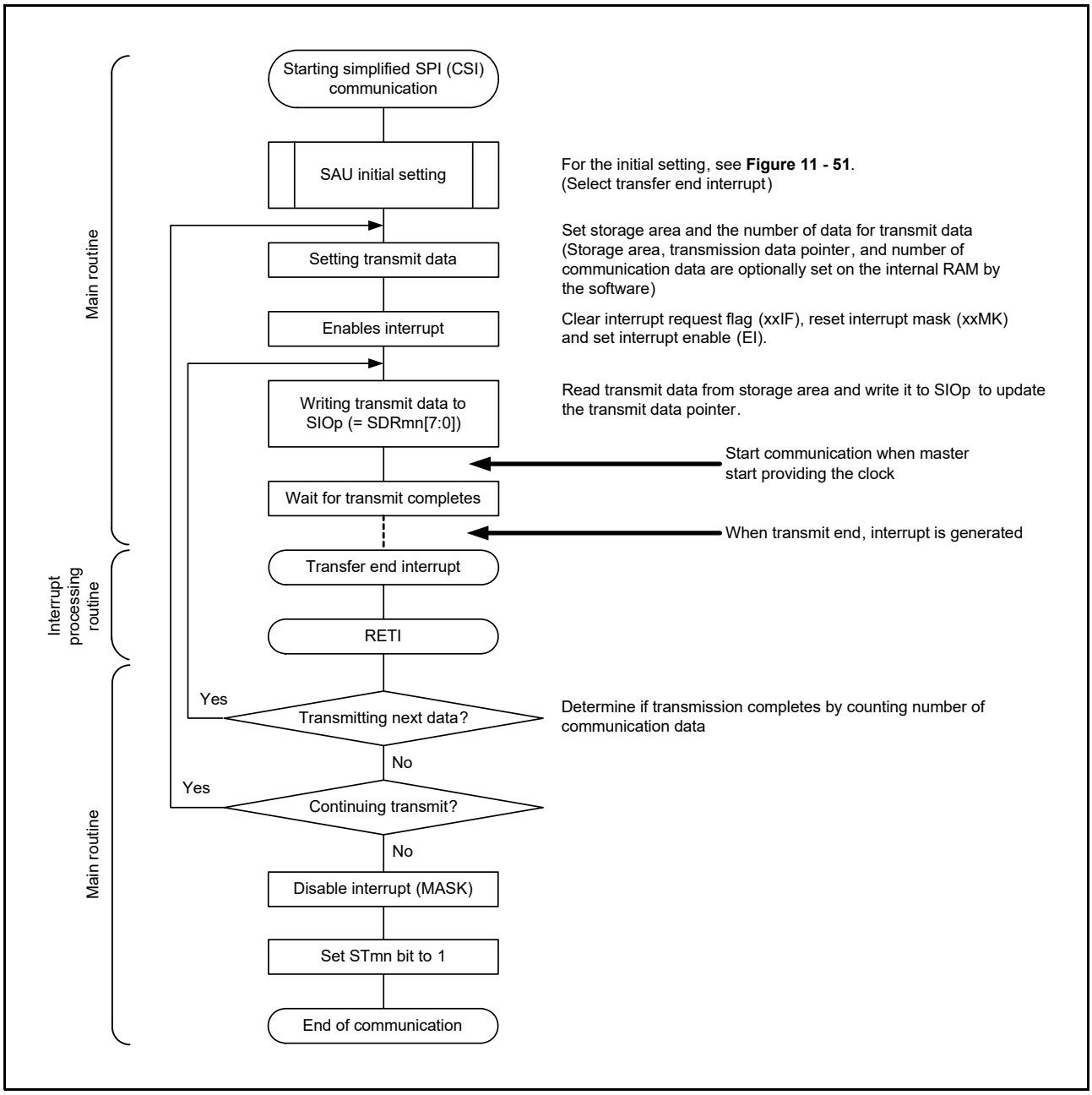
3. Processing flow (in single-transmission mode)

Figure 11 - 54 Timing Chart of Slave Transmission (in Single-transmission Mode) (Type 1: DAPmn = 0, CKPmn = 0)



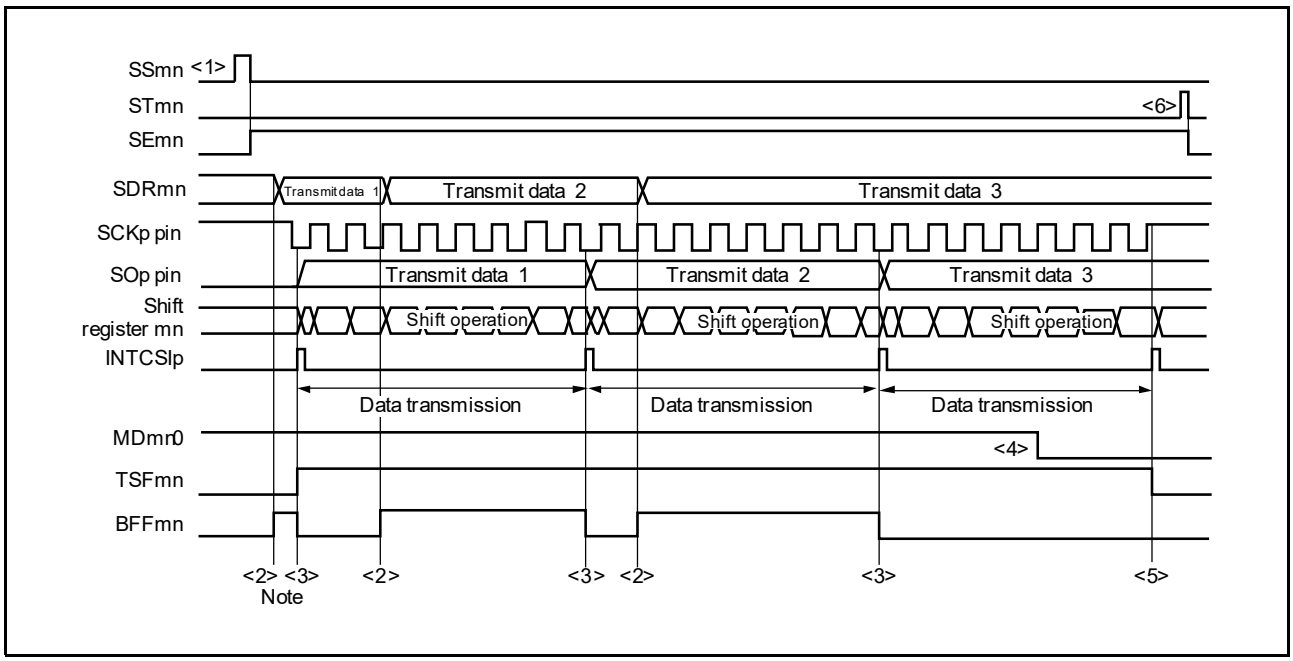
Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

Figure 11 - 55 Flowchart of Slave Transmission (in Single-transmission Mode)



4. Processing flow (in continuous transmission mode)

Figure 11 - 56 Timing Chart of Slave Transmission (in Continuous Transmission Mode)
(Type 1: DAPmn = 0, CKPmn = 0)

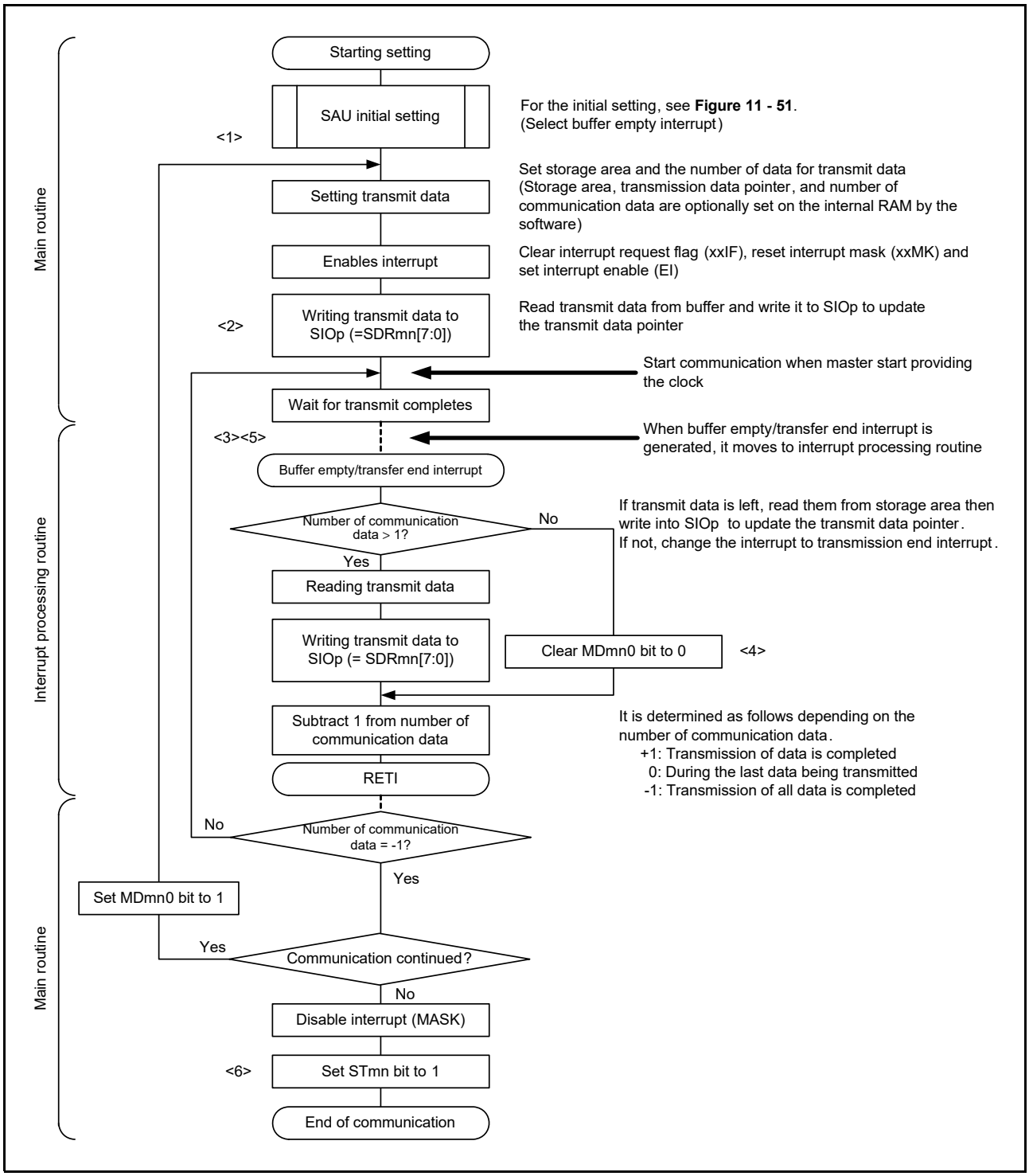


Note If transmit data is written to the SDRmn register while the BFFmn bit of serial status register mn (SSRmn) is 1 (valid data is stored in serial data register mn (SDRmn)), the transmit data is overwritten.

Caution **The MDmn0 bit of serial mode register mn (SSRmn) can be rewritten even during operation. However, rewrite it before transfer of the last bit is started.**

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

Figure 11 - 57 Flowchart of Slave Transmission (in Continuous Transmission Mode)



Remark <1> to <6> in the figure correspond to <1> to <6> in **Figure 11 - 56 Timing Chart of Slave Transmission (in Continuous Transmission Mode) (Type 1: DAPmn = 0, CKPmn = 0)**.

11.5.5 Slave reception

Slave reception is that this MCU receives data from another device in the state of a transfer clock being input from another device.

Simplified SPI	CSI00	CSI01	CSI10	CSI11	CSI20	CSI21
Target channel	Channel 0 of SAU0	Channel 1 of SAU0	Channel 2 of SAU0	Channel 3 of SAU0	Channel 0 of SAU1	Channel 1 of SAU1
Pins used	SCK00, SI00	SCK01, SI01	SCK10, SI10	SCK11, SI11	SCK20, SI20	SCK21, SI21
Interrupt	INTCSI00	INTCSI01	INTCSI10	INTCSI11	INTCSI20	INTCSI21
	Transfer end interrupt only (Setting the buffer empty interrupt is prohibited.)					
Error detection flag	Overrun error detection flag (OVFmn) only					
Transfer data length	7 or 8 bits					
Transfer rate	Max. f_{MCK}/6 [Hz]**Notes 1, 2**					
Data phase	Selectable by the DAPmn bit of the SCRmn register • DAPmn = 0: Data input starts from the start of the operation of the serial clock. • DAPmn = 1: Data input starts half a clock cycle before the start of the serial clock operation.					
Clock phase	Selectable by the CKPmn bit of the SCRmn register • CKPmn = 0: Non-reverse • CKPmn = 1: Reverse					
Data direction	MSB or LSB first					

Note 1. Because the external serial clock input to the SCK00, SCK01, SCK10, SCK11, SCK20, and SCK21 pins is sampled internally and used, the fastest transfer rate is f_{MCK}/6 [Hz].

Note 2. Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics specified in the electrical characteristics. For details, see **Section 28 Electrical Characteristics**.

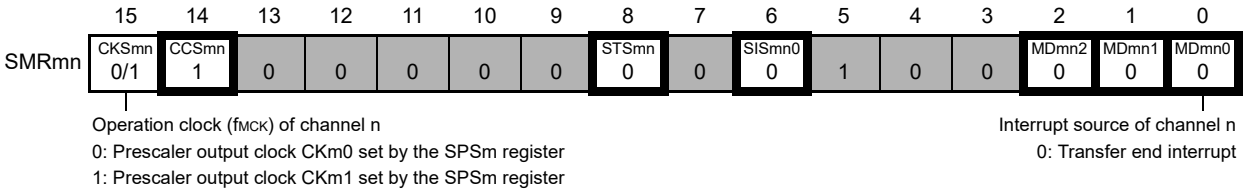
Remark 1. f_{MCK}: Operation clock frequency of target channel
f_{SCK}: Serial clock frequency

Remark 2. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

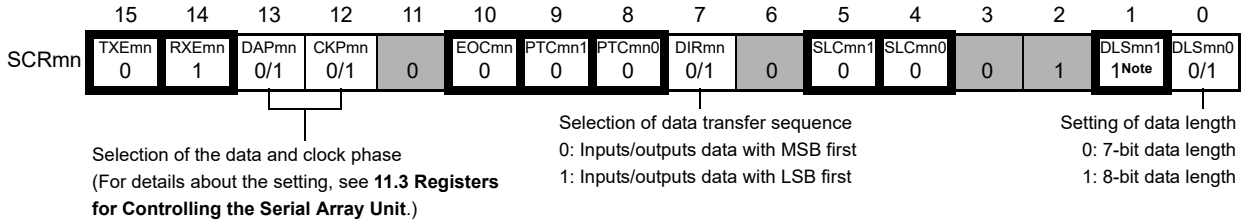
1. Register setting

Figure 11 - 58 Example of Contents of Registers for Slave Reception of Simplified SPI (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21)

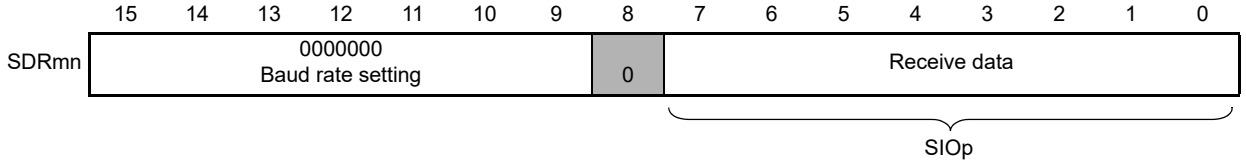
a) Serial mode register mn (SMRmn)



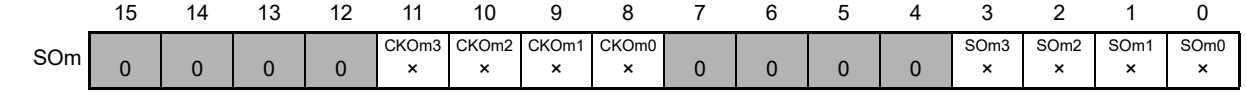
b) Serial communication operation setting register mn (SCRmn)



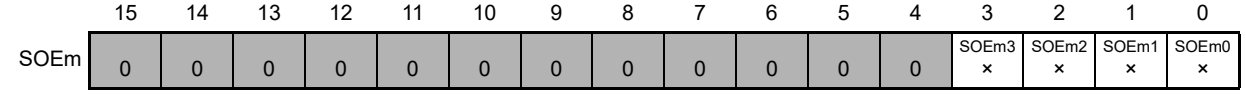
c) Serial data register mn (SDRmn) (lower 8 bits: SIOp)



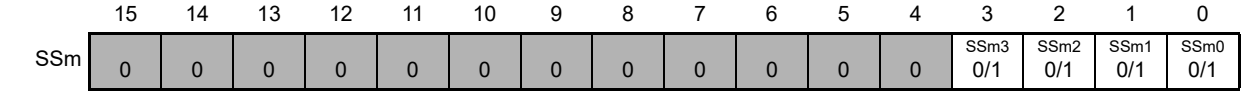
d) Serial output register m (SOm): This register is not used in this mode.



e) Serial output enable register m (SOEm): This register is not used in this mode.



f) Serial channel start register m (SSm): Set only the bit of the target channel to 1.



Note This bit is only present in the SCR00 and SCR01 registers, and is fixed to 1 for the other registers.

(Remarks are listed on the next page.)

Remark 1. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

Remark 2. ■: Setting is fixed in the slave reception mode of the simplified SPI (CSI), ■: Setting disabled (set to the initial value)
×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)
0/1: Set to 0 or 1 depending on the user's application.

2. Operation procedure

Figure 11 - 59 Initial Setting Procedure for Slave Reception

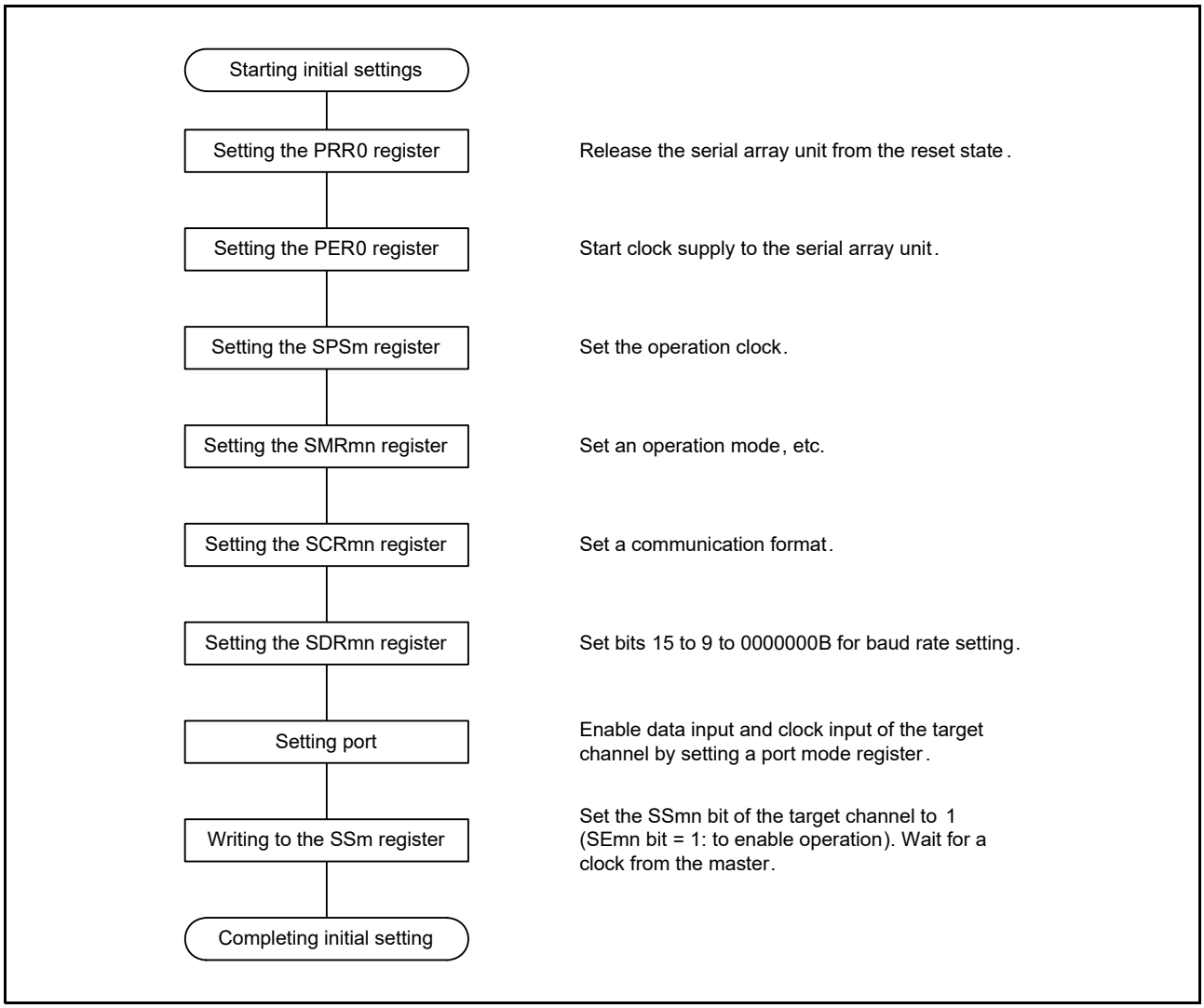


Figure 11 - 60 Procedure for Stopping Slave Reception

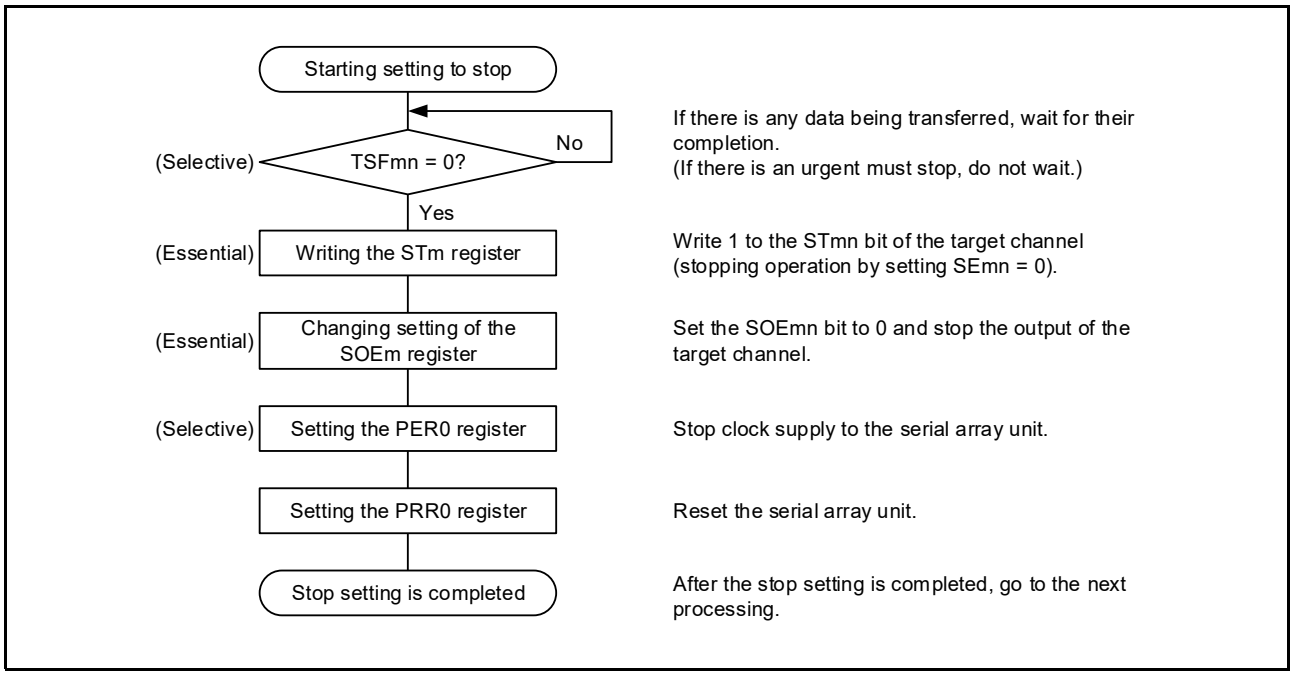
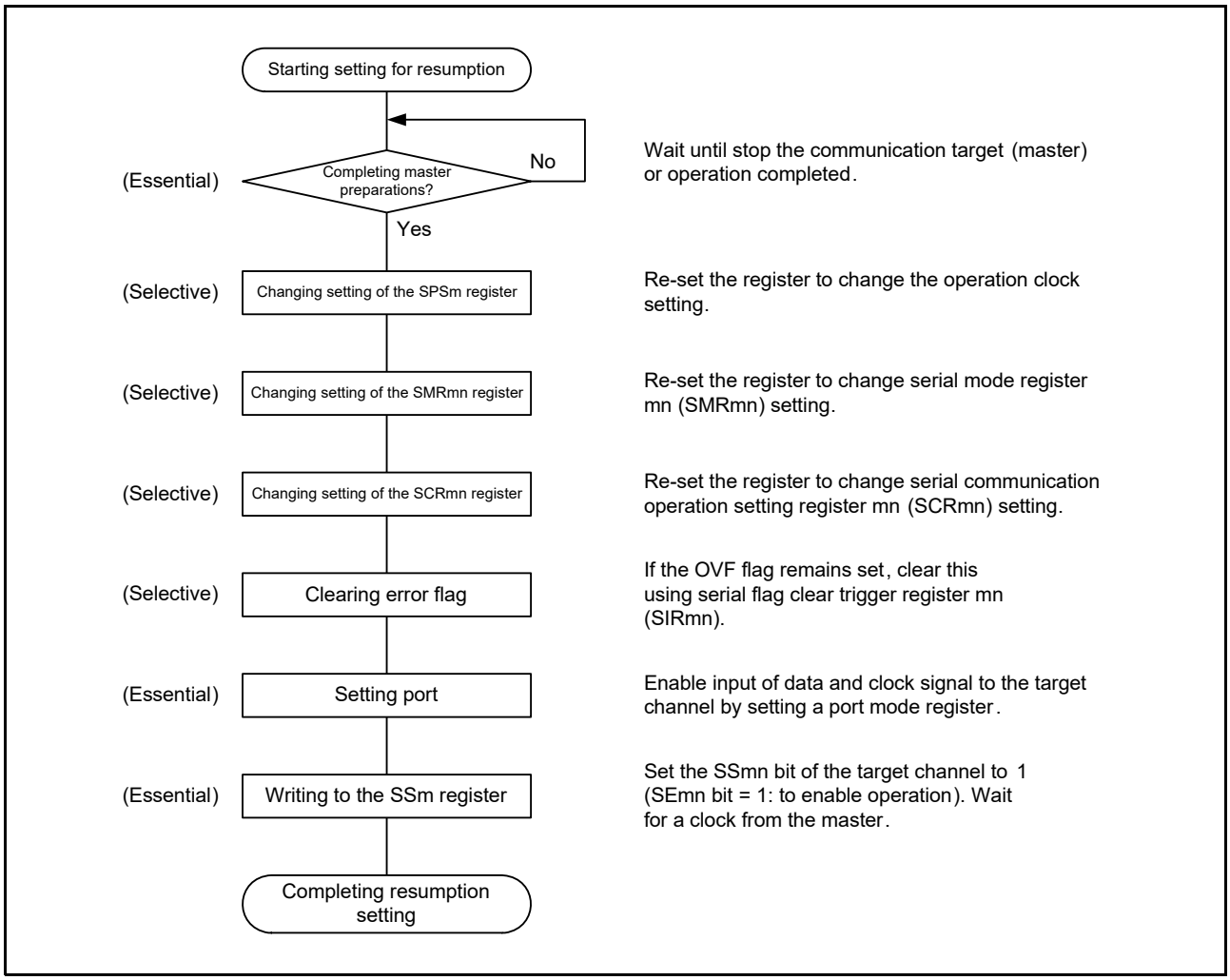


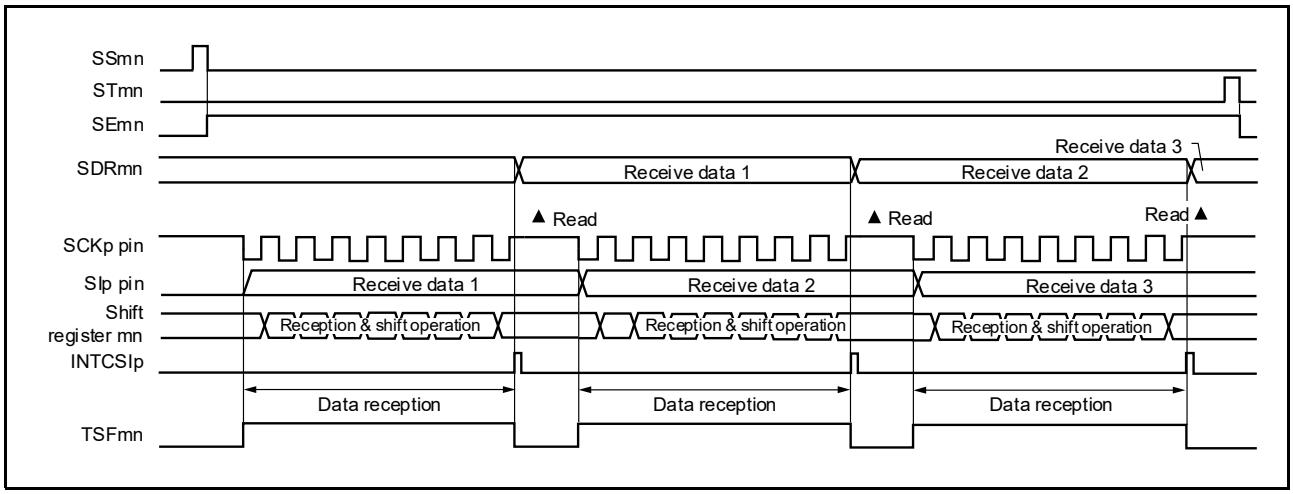
Figure 11 - 61 Procedure for Resuming Slave Reception



Remark If the PRR0 register is rewritten while stopping the communication to reset the serial array unit, wait until the communication target (master) stops or communication finishes, and then perform initialization instead of restarting the communication.

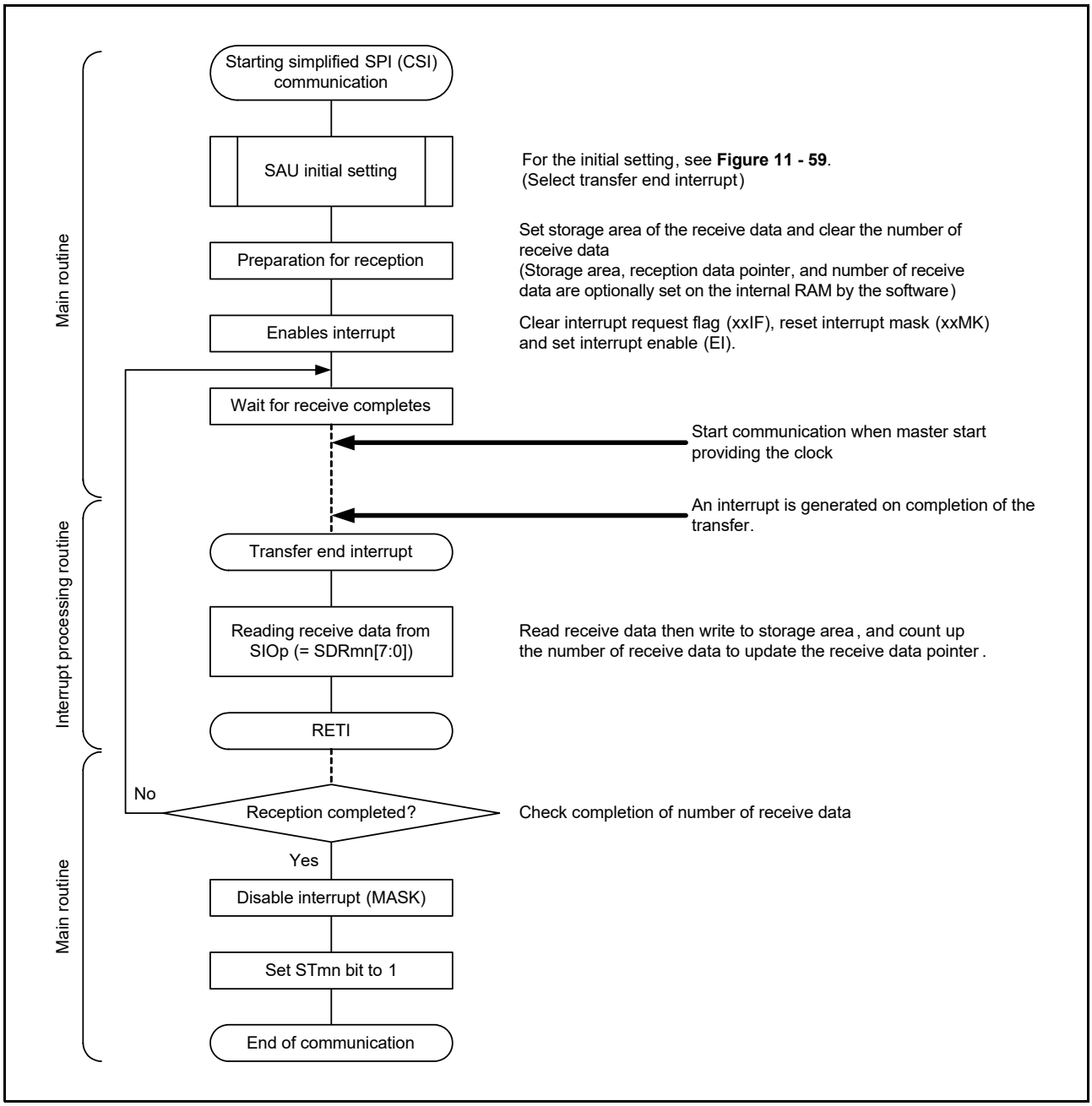
3. Processing flow (in single-reception mode)

Figure 11 - 62 Timing Chart of Slave Reception (in Single-reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)



Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

Figure 11 - 63 Flowchart of Slave Reception (in Single-reception Mode)



11.5.6 Slave transmission/reception

Slave transmission/reception is that this MCU transmits/receives data to/from another device in the state of a transfer clock being input from another device.

Simplified SPI	CSI00	CSI01	CSI10	CSI11	CSI20	CSI21
Target channel	Channel 0 of SAU0	Channel 1 of SAU0	Channel 2 of SAU0	Channel 3 of SAU0	Channel 0 of SAU1	Channel 1 of SAU1
Pins used	SCK00, SI00, SO00	SCK01, SI01, SO01	SCK10, SI10, SO10	SCK11, SI11, SO11	SCK20, SI20, SO20	SCK21, SI21, SO21
Interrupt	INTCSI00	INTCSI01	INTCSI10	INTCSI11	INTCSI20	INTCSI21
	Transfer end interrupt (in single-transfer mode) or buffer empty interrupt (in continuous transfer mode) can be selected.					
Error detection flag	Overrun error detection flag (OVFmn) only					
Transfer data length	7 or 8 bits					
Transfer rate	Max. f_{MCK}/6 [Hz]**Notes 1, 2**					
Data phase	Selectable by the DAPmn bit of the SCRmn register • DAPmn = 0: Data I/O starts at the start of the operation of the serial clock. • DAPmn = 1: Data I/O starts half a clock cycle before the start of the serial clock operation.					
Clock phase	Selectable by the CKPmn bit of the SCRmn register • CKPmn = 0: Non-reverse • CKPmn = 1: Reverse					
Data direction	MSB or LSB first					

Note 1. Because the external serial clock input to the SCK00, SCK01, SCK10, SCK11, SCK20, and SCK21 pins is sampled internally and used, the fastest transfer rate is f_{MCK}/6 [Hz].

Note 2. Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics specified in the electrical characteristics. For details, see **Section 28 Electrical Characteristics**.

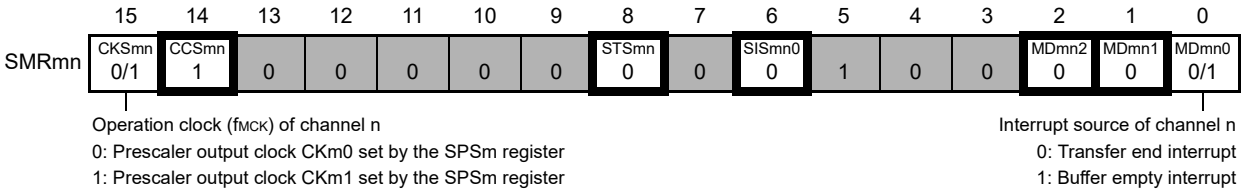
Remark 1. f_{MCK}: Operation clock frequency of target channel
f_{SCK}: Serial clock frequency

Remark 2. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

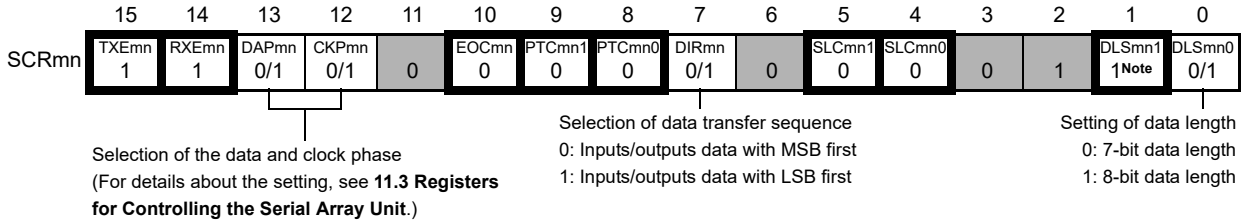
1. Register setting

Figure 11 - 64 Example of Contents of Registers for Slave Transmission/Reception of Simplified SPI (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21)

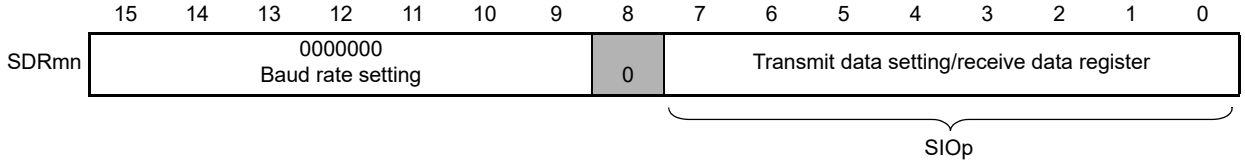
a) Serial mode register mn (SMRmn)



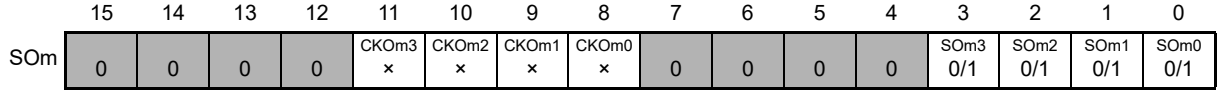
b) Serial communication operation setting register mn (SCRmn)



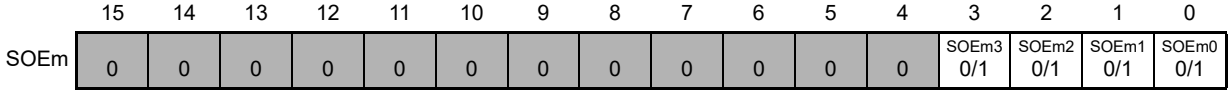
c) Serial data register mn (SDRmn) (lower 8 bits: SIOp)



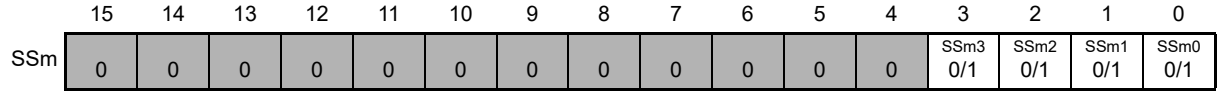
d) Serial output register m (SOm): Set only the bit of the target channel.



e) Serial output enable register m (SOEm): Set only the bit of the target channel to 1.



f) Serial channel start register m (SSm): Set only the bit of the target channel to 1.



Note This bit is only present in the SCR00 and SCR01 registers, and is fixed to 1 for the other registers.

Caution Be sure to set transmit data to the SIOp register before the clock from the master is started.

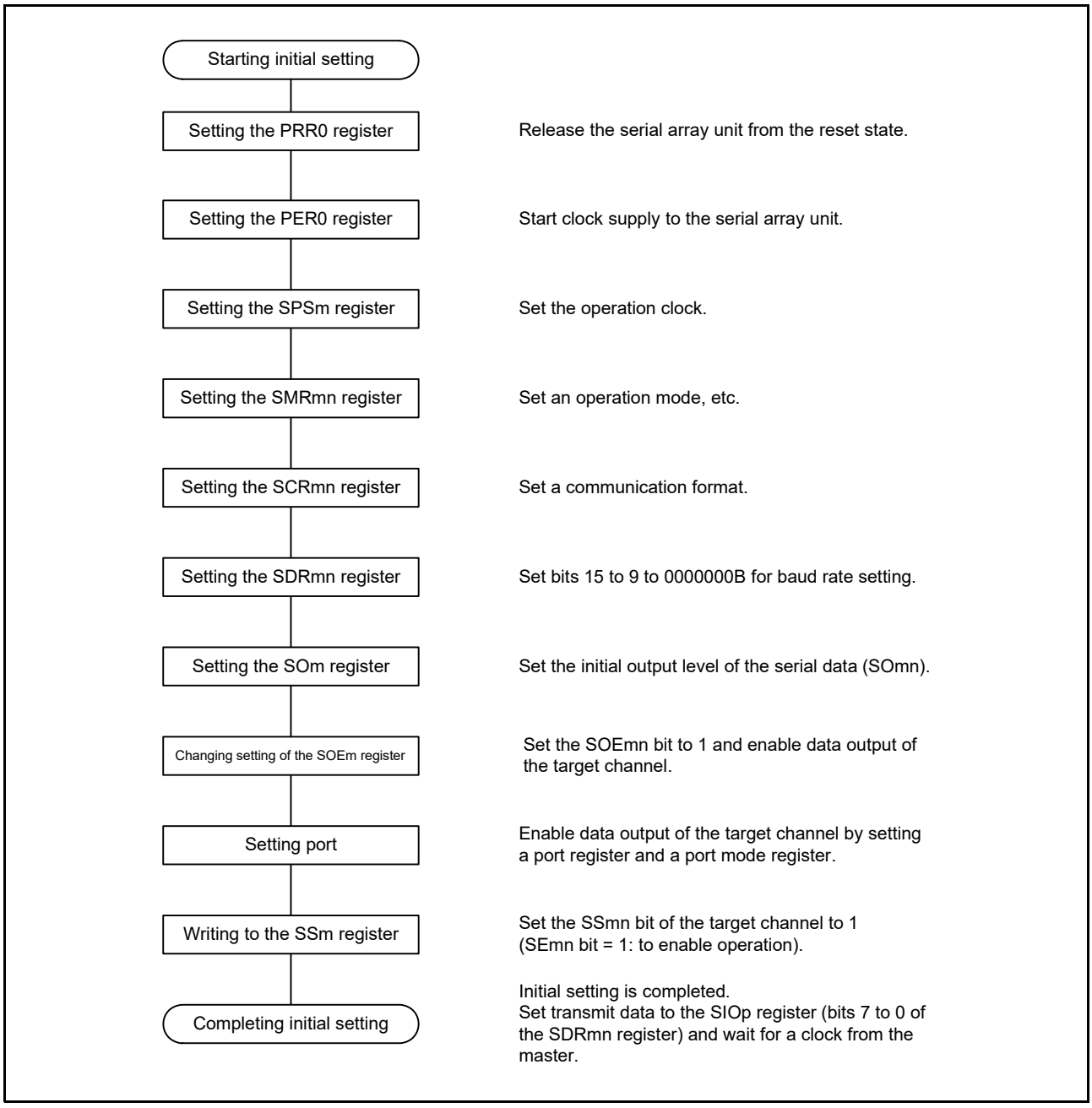
(Remarks are listed on the next page.)

Remark 1. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

Remark 2. ■: Setting is fixed in the simplified SPI (CSI) slave transmission/reception mode, ■: Setting disabled (set to the initial value)
×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)
0/1: Set to 0 or 1 depending on the user's application.

2. Operation procedure

Figure 11 - 65 Initial Setting Procedure for Slave Transmission/Reception



Caution Be sure to set transmit data to the SIOp register before the clock from the master is started.

Figure 11 - 66 Procedure for Stopping Slave Transmission/Reception

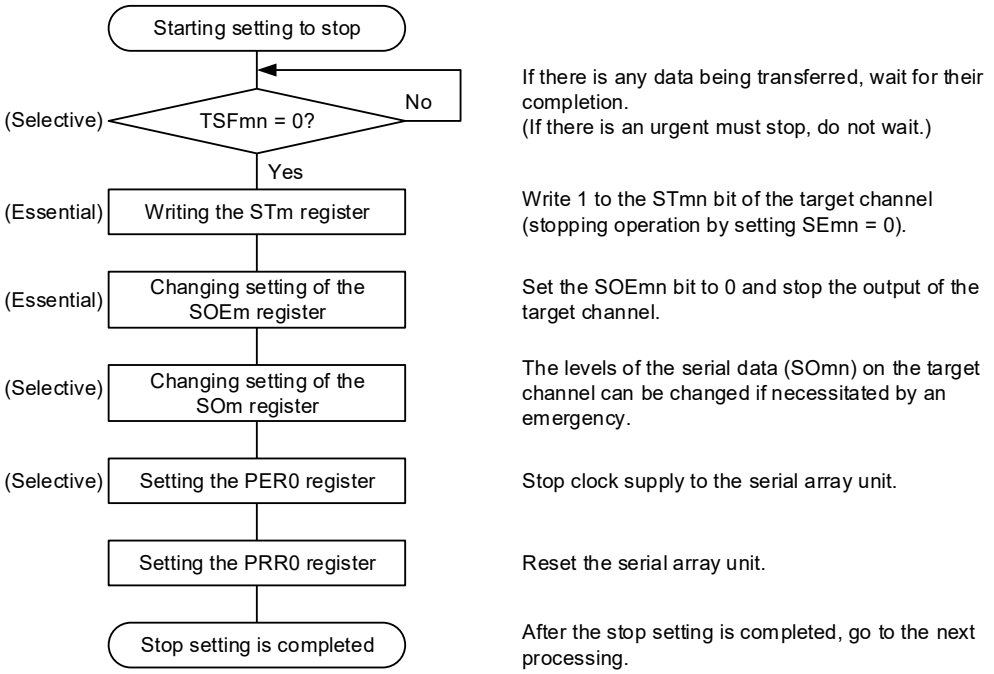
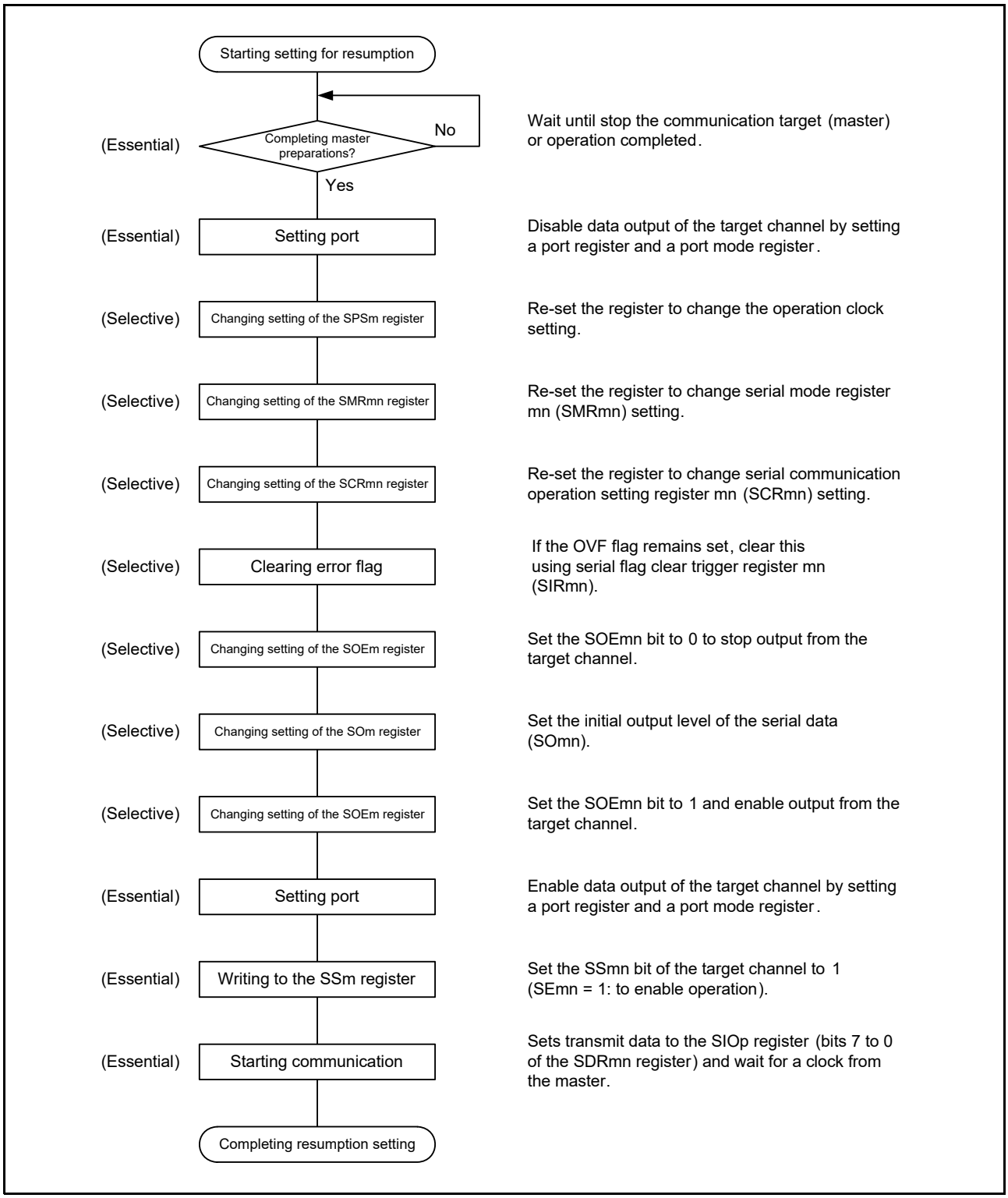


Figure 11 - 67 Procedure for Resuming Slave Transmission/Reception

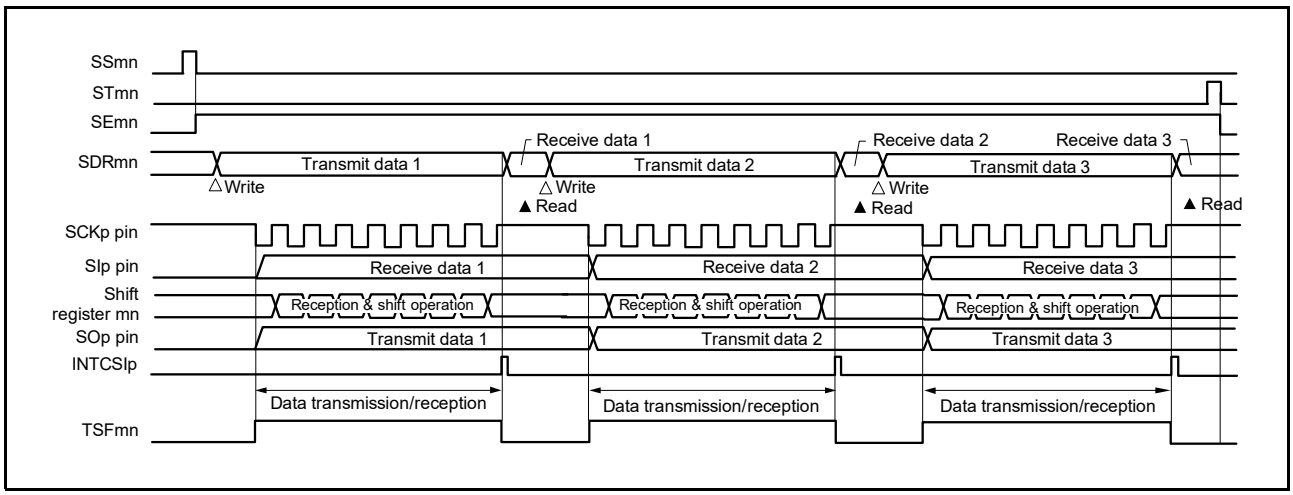


Caution 1. Be sure to set transmit data to the SIOp register before the clock from the master is started.

Caution 2. If the PRR0 register is rewritten while stopping the communication to reset the serial array unit, wait until the communication target (master) stops or communication finishes, and then perform initialization instead of restarting the communication.

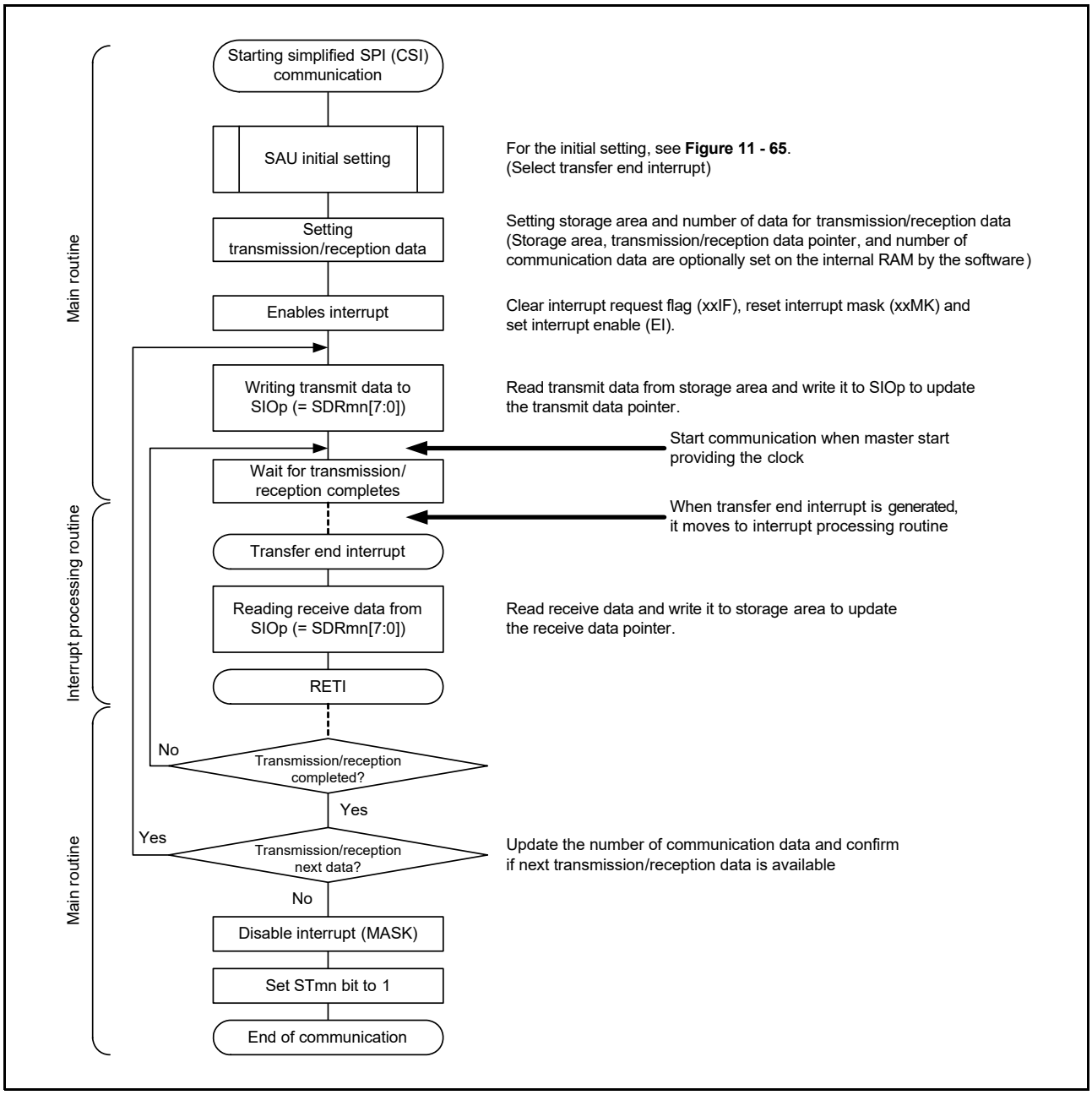
3. Processing flow (in single-transmission/reception mode)

Figure 11 - 68 Timing Chart of Slave Transmission/Reception (in Single-transmission/reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)



Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

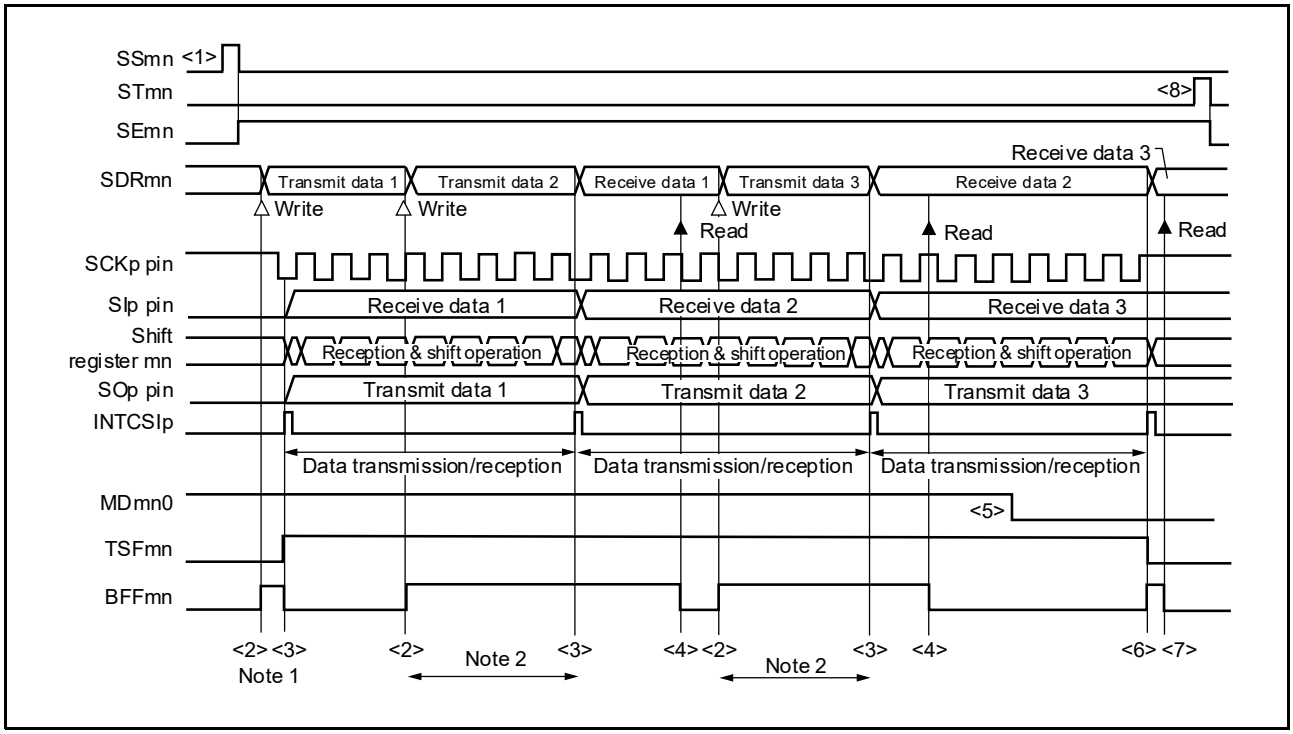
Figure 11 - 69 Flowchart of Slave Transmission/Reception (in Single-transmission/reception Mode)



Caution Be sure to set transmit data to the SIOp register before the clock from the master is started.

4. Processing flow (in continuous transmission/reception mode)

Figure 11 - 70 Timing Chart of Slave Transmission/Reception (in Continuous Transmission/Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)



Note 1. If transmit data is written to the SDRmn register while the BFFmn bit of serial status register mn (SSRmn) is 1 (valid data is stored in serial data register mn (SDRmn)), the transmit data is overwritten.

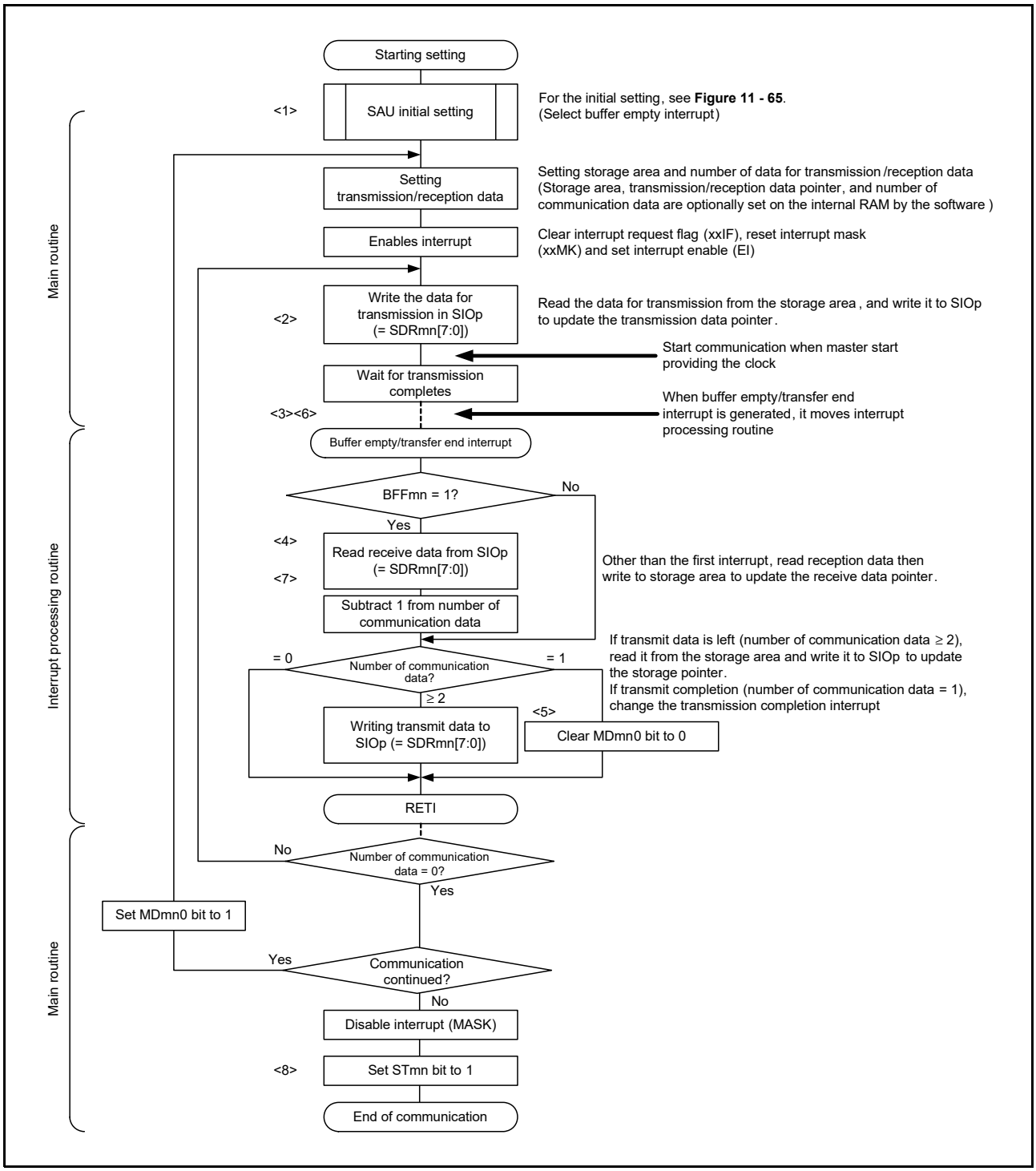
Note 2. The transmit data can be read by reading the SDRmn register during this period. At this time, the transfer operation is not affected.

Caution **The MDmn0 bit of serial mode register mn (SSRmn) can be rewritten even during operation. However, rewrite it before transfer of the last bit is started, so that it has been rewritten before the transfer end interrupt of the last transmit data.**

Remark 1. <1> to <8> in the figure correspond to <1> to <8> in **Figure 11 - 71 Flowchart of Slave Transmission/Reception (in Continuous Transmission/Reception Mode)**.

Remark 2. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

Figure 11 - 71 Flowchart of Slave Transmission/Reception (in Continuous Transmission/Reception Mode)



Caution **Be sure to set transmit data to the SIOp register before the clock from the master is started.**

Remark <1> to <8> in the figure correspond to <1> to <8> in **Figure 11 - 70 Timing Chart of Slave Transmission/Reception (in Continuous Transmission/Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)**.

11.5.7 SNOOZE mode

The SNOOZE mode makes the simplified SPI (CSI) perform reception operations upon SCKp pin input detection while in the STOP mode. Normally the simplified SPI (CSI) stops communication in the STOP mode. However, using the SNOOZE mode enables the simplified SPI (CSI) to perform reception operations without CPU operation upon detection of the SCKp pin input. Only the simplified SPI (CSI00) supports the SNOOZE mode.

When using the simplified SPI (CSI) in SNOOZE mode, make the following setting before switching to the STOP mode. See **Figure 11 - 73 Flowchart of SNOOZE Mode Operation (once startup)** and **Figure 11 - 75 Flowchart of SNOOZE Mode Operation (continuous startup)**.

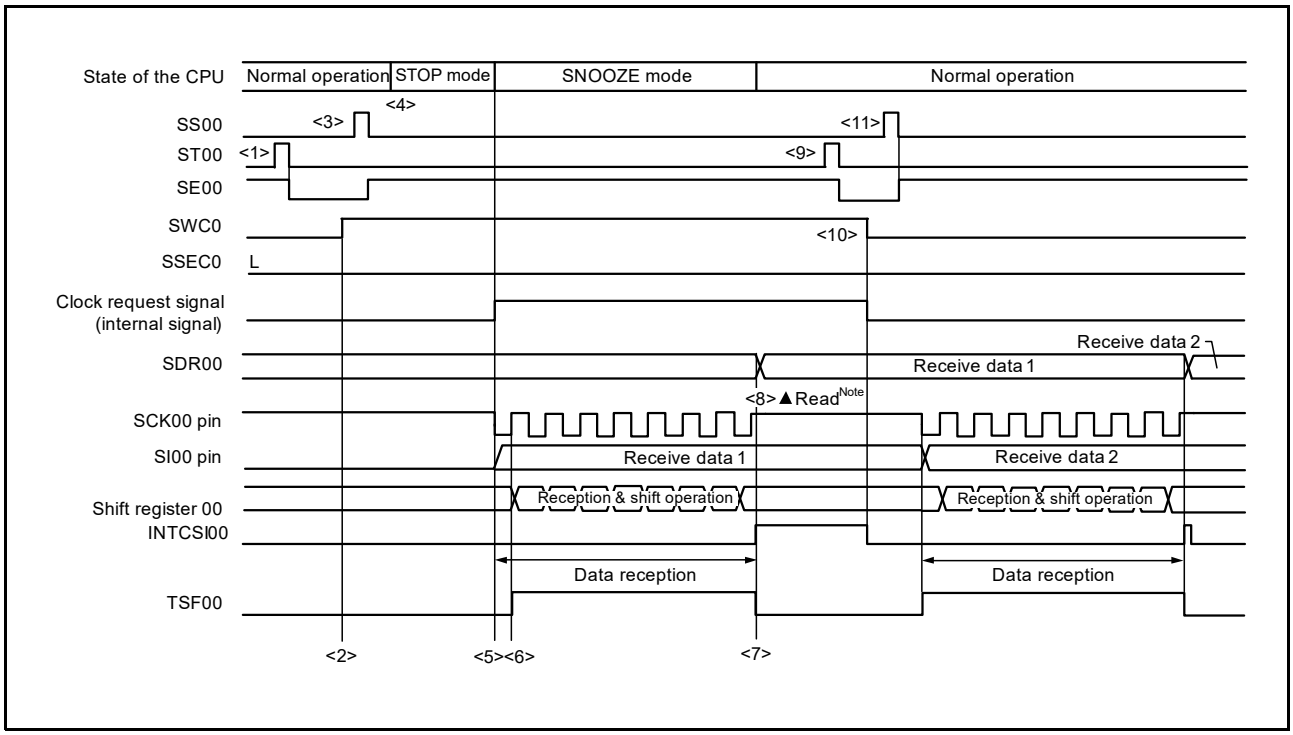
- When using the SNOOZE mode function, set the SWCm bit of serial standby control register m (SSCm) to 1 just before switching to the STOP mode. After the initial setting has been completed, set the SSm0 bit of serial channel start register m (SSm) to 1.
- The CPU shifts to the SNOOZE mode on detecting the valid edge of the SCKp signal following a transition to the STOP mode. A CSIp starts reception on detecting input of the serial clock on the SCKp pin.

Caution 1. The SNOOZE mode can only be specified when the high-speed on-chip oscillator clock is selected for f_{CLK}.

Caution 2. The maximum transfer rate when using CSIp in the SNOOZE mode is 1 Mbps.

1. SNOOZE mode operation (once startup)

Figure 11 - 72 Timing Chart of SNOOZE Mode Operation (once startup) (Type 1: DAPmn = 0, CKPmn = 0)



Note Only read received data while SWCm = 1 and before the next valid edge of the SCKp pin input is detected.

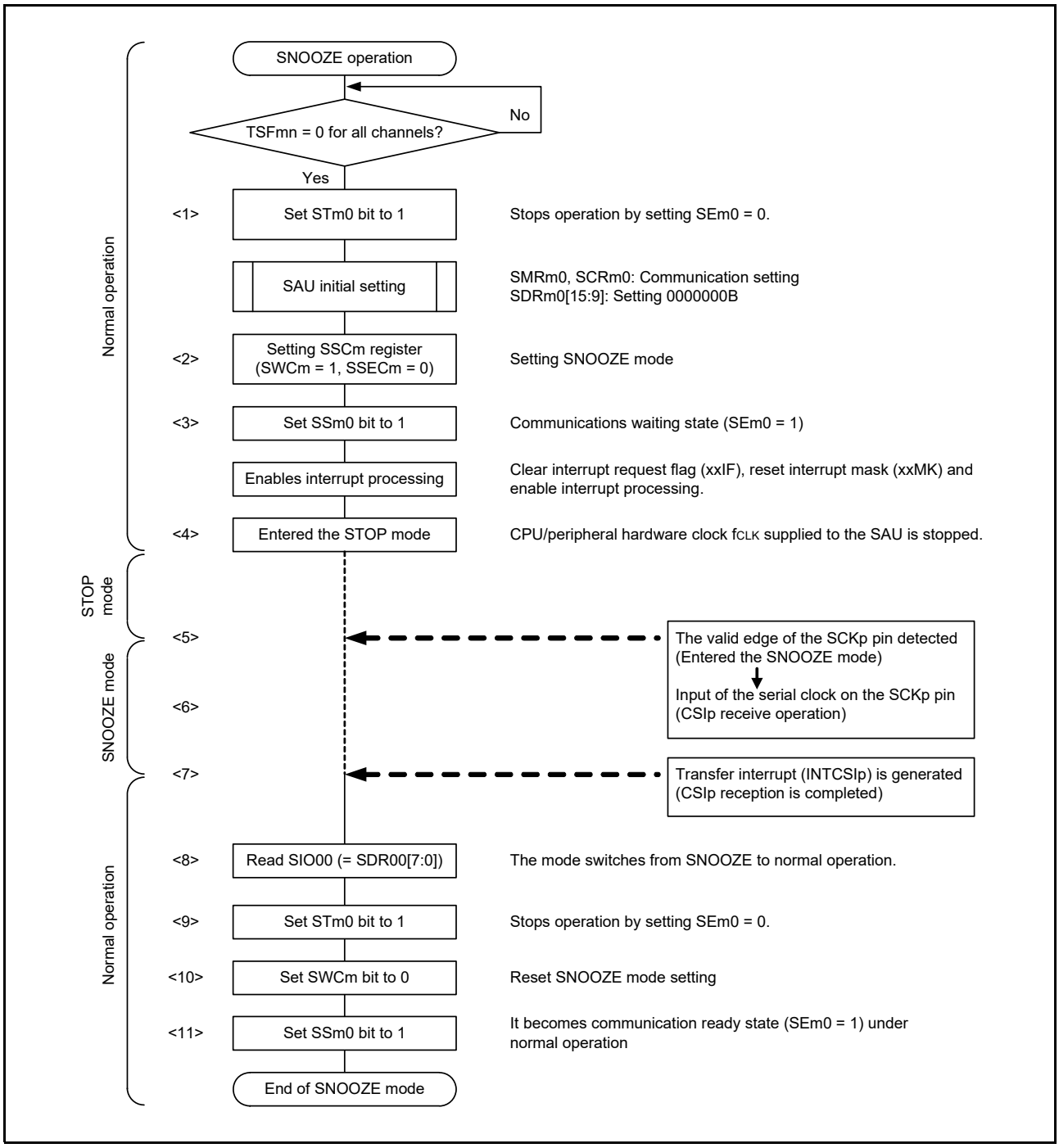
Caution 1. Before switching to the SNOOZE mode or after reception operation in the SNOOZE mode finishes, set the STm0 bit to 1 (the SEm0 bit is cleared and the operation stops). After the receive operation completes, also clear the SWCm bit to 0 (SNOOZE mode release).

Caution 2. When SWCm = 1, the BFFm0 and OVFm0 flags will not change.

Remark 1. <1> to <11> in the figure correspond to <1> to <11> in **Figure 11 - 73 Flowchart of SNOOZE Mode Operation (once startup)**.

Remark 2. m = 0, p = 00

Figure 11 - 73 Flowchart of SNOOZE Mode Operation (once startup)

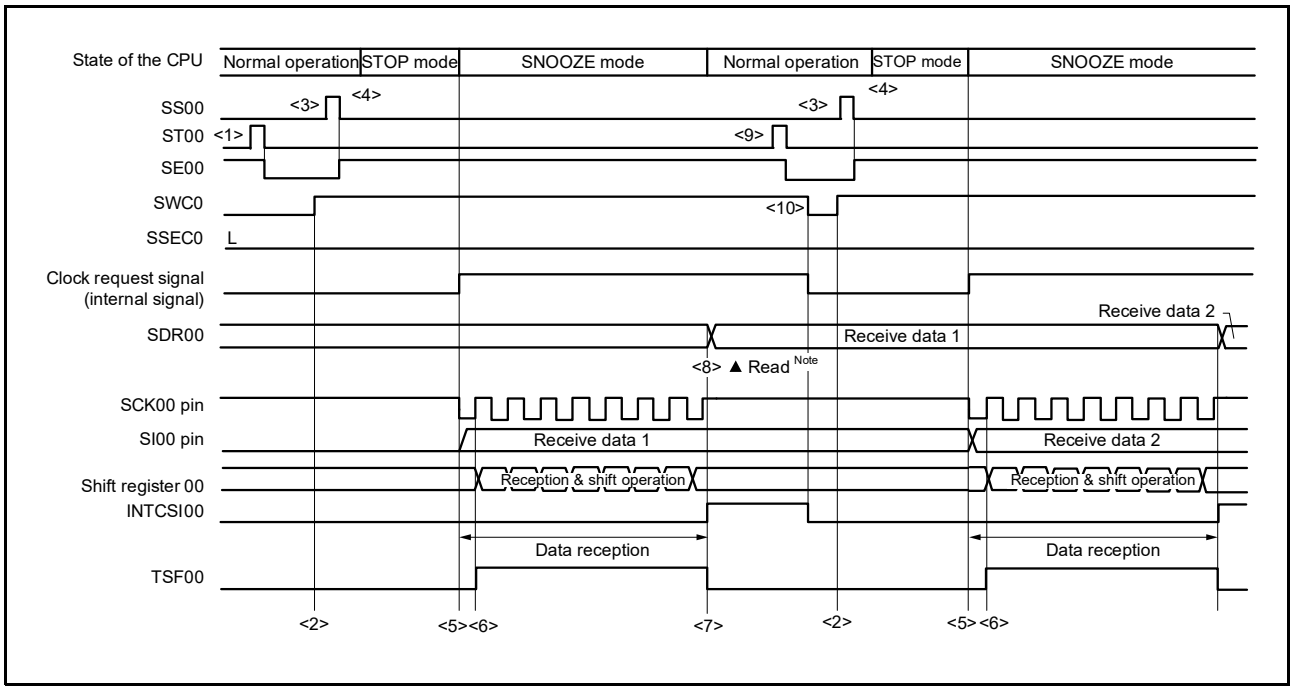


Remark 1. <1> to <11> in the figure correspond to <1> to <11> in **Figure 11 - 72 Timing Chart of SNOOZE Mode Operation (once startup) (Type 1: DAPmn = 0, CKPmn = 0)**.

Remark 2. m = 0, p = 00

2. SNOOZE mode operation (continuous startup)

Figure 11 - 74 Timing Chart of SNOOZE Mode Operation (continuous startup) (Type 1: DAPmn = 0, CKPmn = 0)



Note Only read received data while SWCm = 1 and before the next valid edge of the SCKp pin input is detected.

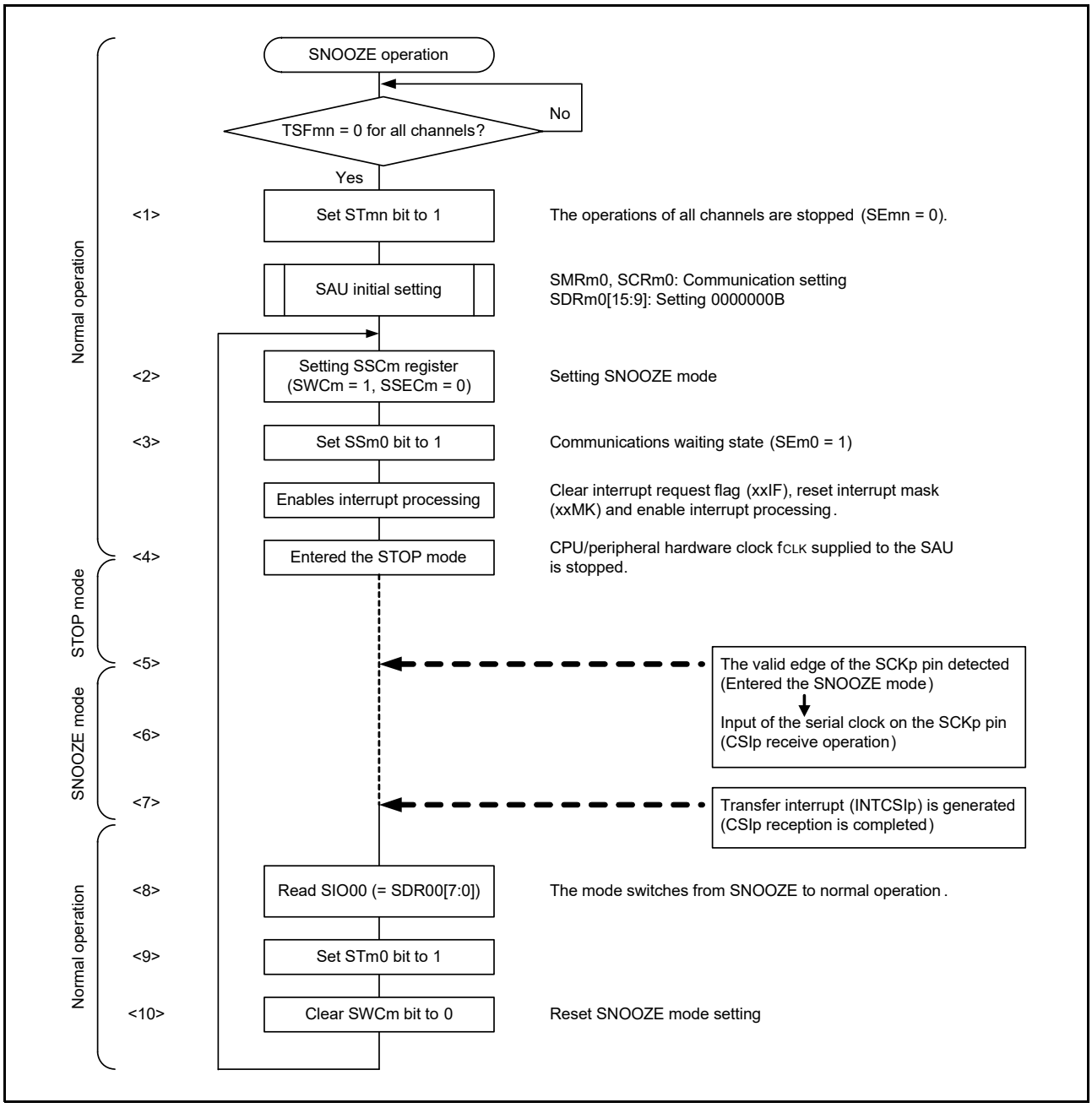
Caution 1. Before switching to the SNOOZE mode or after reception operation in the SNOOZE mode finishes, set the STm0 bit to 1 (the SEm0 bit is cleared and the operation stops). After the receive operation completes, also clear the SWCm bit to 0 (SNOOZE mode release).

Caution 2. When SWCm = 1, the BFFm0 and OVFm0 flags will not change.

Remark 1. <1> to <10> in the figure correspond to <1> to <10> in **Figure 11 - 75 Flowchart of SNOOZE Mode Operation (continuous startup)**.

Remark 2. m = 0, p = 00

Figure 11 - 75 Flowchart of SNOOZE Mode Operation (continuous startup)



Remark 1. <1> to <10> in the figure correspond to <1> to <10> in **Figure 11 - 74 Timing Chart of SNOOZE Mode Operation (continuous startup) (Type 1: DAPmn = 0, CKPmn = 0)**.

Remark 2. m = 0, p = 00

11.5.8 Calculating transfer clock frequency

The transfer clock frequency for simplified SPI (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) communication can be calculated by the following expressions.

1. Master

(Transfer clock frequency) = {Operation clock (f_{MCK}) frequency of target channel} ÷ (SDRmn[15:9] + 1) ÷ 2 [Hz]

2. Slave

(Transfer clock frequency) = {Frequency of serial clock (SCK) supplied by master} **Note** [Hz]

Note The permissible maximum transfer clock frequency is $f_{MCK}/6$.

Remark The value of SDRmn[15:9] is the value of bits 15 to 9 of serial data register mn (SDRmn) (0000000B to 1111111B) and therefore is 0 to 127.

The operation clock (f_{MCK}) is determined by serial clock select register m (SPSm) and the CKSmn bit of serial mode register mn (SMRmn).

Table 11 - 3 Selection of Operation Clock For Simplified SPI (1/2)

SMRmn Register	SPSm Register								Operation Clock (f_{MCK})**Note**	
CKSmn	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00		f_{CLK} = 32 MHz
0	×	×	×	×	0	0	0	0	f_{CLK}	32 MHz
	×	×	×	×	0	0	0	1	$f_{CLK}/2$	16 MHz
	×	×	×	×	0	0	1	0	$f_{CLK}/2^2$	8 MHz
	×	×	×	×	0	0	1	1	$f_{CLK}/2^3$	4 MHz
	×	×	×	×	0	1	0	0	$f_{CLK}/2^4$	2 MHz
	×	×	×	×	0	1	0	1	$f_{CLK}/2^5$	1 MHz
	×	×	×	×	0	1	1	0	$f_{CLK}/2^6$	500 kHz
	×	×	×	×	0	1	1	1	$f_{CLK}/2^7$	250 kHz
	×	×	×	×	1	0	0	0	$f_{CLK}/2^8$	125 kHz
	×	×	×	×	1	0	0	1	$f_{CLK}/2^9$	62.5 kHz
	×	×	×	×	1	0	1	0	$f_{CLK}/2^{10}$	31.25 kHz
	×	×	×	×	1	0	1	1	$f_{CLK}/2^{11}$	15.63 kHz
	×	×	×	×	1	1	0	0	$f_{CLK}/2^{12}$	7.81 kHz
	×	×	×	×	1	1	0	1	$f_{CLK}/2^{13}$	3.91 kHz
	×	×	×	×	1	1	1	0	$f_{CLK}/2^{14}$	1.95 kHz
	×	×	×	×	1	1	1	1	$f_{CLK}/2^{15}$	977 Hz

Table 11 - 3 Selection of Operation Clock For Simplified SPI (2/2)

SMRmn Register	SPSm Register								Operation Clock (f_{MCK})[Note]	
CKSmn	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00		f_{CLK} = 32 MHz
1	0	0	0	0	×	×	×	×	f_{CLK}	32 MHz
	0	0	0	1	×	×	×	×	$f_{CLK}/2$	16 MHz
	0	0	1	0	×	×	×	×	$f_{CLK}/2^2$	8 MHz
	0	0	1	1	×	×	×	×	$f_{CLK}/2^3$	4 MHz
	0	1	0	0	×	×	×	×	$f_{CLK}/2^4$	2 MHz
	0	1	0	1	×	×	×	×	$f_{CLK}/2^5$	1 MHz
	0	1	1	0	×	×	×	×	$f_{CLK}/2^6$	500 kHz
	0	1	1	1	×	×	×	×	$f_{CLK}/2^7$	250 kHz
	1	0	0	0	×	×	×	×	$f_{CLK}/2^8$	125 kHz
	1	0	0	1	×	×	×	×	$f_{CLK}/2^9$	62.5 kHz
	1	0	1	0	×	×	×	×	$f_{CLK}/2^{10}$	31.25 kHz
	1	0	1	1	×	×	×	×	$f_{CLK}/2^{11}$	15.63 kHz
	1	1	0	0	×	×	×	×	$f_{CLK}/2^{12}$	7.81 kHz
	1	1	0	1	×	×	×	×	$f_{CLK}/2^{13}$	3.91 kHz
	1	1	1	0	×	×	×	×	$f_{CLK}/2^{14}$	1.95 kHz
	1	1	1	1	×	×	×	×	$f_{CLK}/2^{15}$	977 Hz
Other than above									Setting prohibited	

Note When changing the clock selected for f_{CLK} (by changing the system clock control register (CKC) value), do so after having stopped (serial channel stop register m (STm) = 000FH) the operation of the serial array unit (SAU).

Remark 1. ×: Don't care

Remark 2. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

11.5.9 Procedure for processing errors that occurred during simplified SPI (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) communication

Table 11 - 4 describes the procedure for processing errors that occurred during simplified SPI (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) communication.

Table 11 - 4 Processing Procedure in Case of Overrun Error

Software Manipulation	State of the Hardware	Remark
Reads serial data register mn (SDRmn).	The BFFmn bit of the SSRmn register is set to 0 and channel n is enabled to receive data	This is to prevent an overrun error if the next reception is completed during error processing.
Reads serial status register mn (SSRmn).		The error type is identified and the read value is used to clear the error flag.
Writes 1 to serial flag clear trigger register mn (SIRmn).	The error flag is cleared.	The error only during reading can be cleared, by writing the value read from the SSRmn register to the SIRmn register without modification.

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

11.6 Operation of UART (UART0 to UART2) Communication

This is a start-stop synchronization communication function using two lines: serial data transmission (TxD) and serial data reception (RxD) lines. By using these two communication lines, each data frame, which consist of a start bit, data, parity bit, and stop bit, is transferred asynchronously (using the internal baud rate) between the microcontroller and the other communication party. Full-duplex asynchronous communication UART communication can be performed by using a channel dedicated to transmission (even-numbered channel) and a channel dedicated to reception (odd-numbered channel). The LIN bus can be implemented by using UART2, timer array unit 0 (channel 7), and an external interrupt (INTP0).

[Data transmission/reception]

- Data length of 7, 8, or 9 bits **Note**
- MSB/LSB first selectable
- Level setting of transmit/receive data (selecting whether to reverse the level)
- Parity bit appending and parity check functions
- Stop bit appending, stop bit check function

[Interrupt function]

- Transfer end interrupt/buffer empty interrupt
- Error interrupt in case of framing error, parity error, or overrun error

[Error detection flag]

- Framing error, parity error, or overrun error

In addition, the UART 0 reception supports the SNOOZE mode. In the SNOOZE mode, data can be received without CPU processing upon detecting RxD input in the STOP mode.

UART2 (channels 0 and 1 of unit 1) supports the LIN bus.

[LIN-bus functions]

- Wakeup signal detection
- Break field (BF) detection
- Sync field measurement, baud rate calculation

} Using the external interrupt (INTP0) and the timer array unit 0 (channel 7)

Note Only UART0 supports the 9-bit data length.

When a serial array unit is to be used as UART, the low-speed on-chip oscillator clock (f_{IL}) cannot be selected for f_{CLK}.

- UART0 uses channels 0 and 1 of SAU0.
- UART1 uses channels 2 and 3 of SAU0.
- UART2 uses channels 0 and 1 of SAU1.

20-pin products

Unit	Channel	Used as Simplified SPI (CSI)	Used as UART	Used as Simplified I²C
0	0	CSI00	UART0	IIC00
	1	—		—
	2	—	UART1	—
	3	CSI11		IIC11
1	0	—	UART2 (supporting the LIN bus)	—
	1	—		—

32-pin products

Unit	Channel	Used as Simplified SPI (CSI)	Used as UART	Used as Simplified I²C
0	0	CSI00	UART0	IIC00
	1	—		—
	2	—	UART1	—
	3	CSI11		IIC11
1	0	CSI20	UART2 (supporting the LIN bus)	IIC20
	1	—		—

44-pin products

Unit	Channel	Used as Simplified SPI (CSI)	Used as UART	Used as Simplified I²C
0	0	CSI00	UART0	IIC00
	1	—		—
	2	—	UART1	—
	3	CSI11		IIC11
1	0	CSI20	UART2 (supporting the LIN bus)	IIC20
	1	CSI21		IIC21

48-pin products

Unit	Channel	Used as Simplified SPI (CSI)	Used as UART	Used as Simplified I²C
0	0	CSI00	UART0	IIC00
	1	CSI01		IIC01
	2	—	UART1	—
	3	CSI11		IIC11
1	0	CSI20	UART2 (supporting the LIN bus)	IIC20
	1	CSI21		IIC21

64-pin products

Unit	Channel	Used as Simplified SPI (CSI)	Used as UART	Used as Simplified I²C
0	0	CSI00	UART0	IIC00
	1	CSI01		IIC01
	2	CSI10	UART1	IIC10
	3	CSI11		IIC11
1	0	CSI20	UART2 (supporting the LIN bus)	IIC20
	1	CSI21		IIC21

Select a single function for each channel. Only the selected function is possible. If UART0 is selected for channels 0 and 1 of unit 0, for example, the CSI00 and CSI01 functions cannot be used. At this time, however, channel 2 or 3 of the same unit can be used for a function other than UART0, such as CSI10, UART1, and IIC10.

Caution When using a serial array unit for UART, both the transmitter side (even-numbered channel) and the receiver side (odd-numbered channel) can only be used for UART.

UART performs the following four types of communication operations.

- UART transmission (See **11.6.1**.)
- UART reception (See **11.6.2**.)
- LIN transmission (UART2 only) (See **11.7.1**.)
- LIN reception (UART2 only) (See **11.7.2**.)

11.6.1 UART transmission

UART transmission is an operation to transmit data from this MCU to another device asynchronously (start-stop synchronization). Of two channels used for UART, the even channel is used for UART transmission.

UART	UART0	UART1	UART2
Target channel	Channel 0 of SAU0	Channel 2 of SAU0	Channel 0 of SAU1
Pins used	TxD0	TxD1	TxD2
Interrupt	INTST0	INTST1	INTST2
	Transfer end interrupt (in single-transfer mode) or buffer empty interrupt (in continuous transfer mode) can be selected.		
Error detection flag	None		
Transfer data length	7, 8, or 9 bits**Note 1**		
Transfer rate**Note 2**	Max. f_{MCK}/6 [bps] (SDRmn[15:9] = 2 or more), Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [bps]		
Data phase	Non-reverse output (default: high level) Reverse output (default: low level)		
Parity bit	The following selectable • No parity bit • Appending 0 parity • Appending even parity • Appending odd parity		
Stop bit	The following selectable • Appending 1 bit • Appending 2 bit		
Data direction	MSB or LSB first		

Note 1. Only UART0 supports the 9-bit data length.

Note 2. Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics specified in the electrical characteristics. For details, see **Section 28 Electrical Characteristics**.

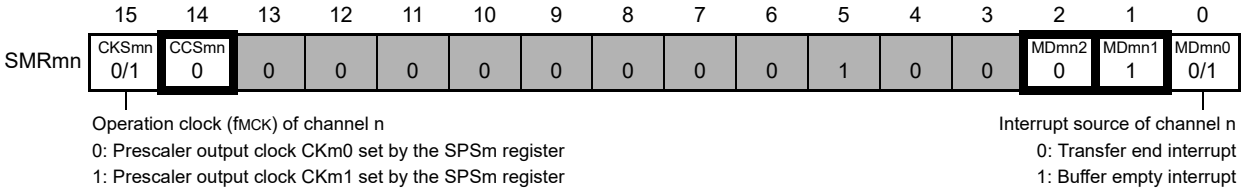
Remark 1. f_{MCK}: Operation clock frequency of target channel
f_{CLK}: CPU/peripheral hardware clock frequency

Remark 2. m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), mn = 00, 02, 10

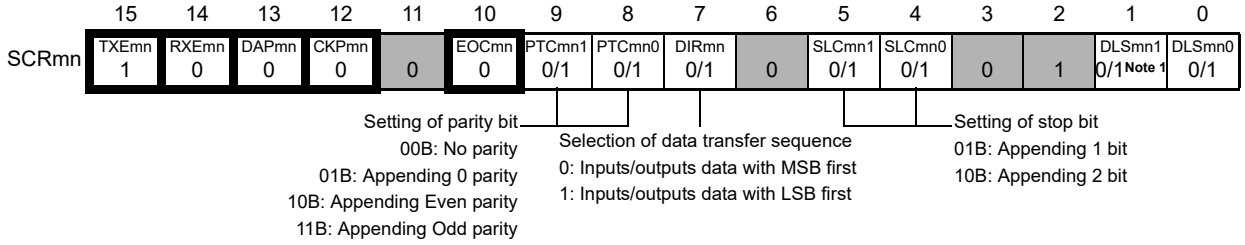
1. Register setting

Figure 11 - 76 Example of Contents of Registers for UART Transmission of UART (UART0 to UART2)

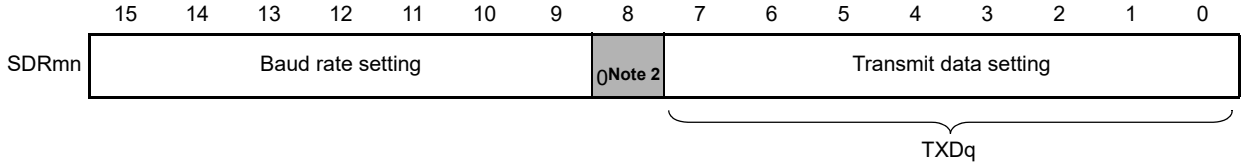
a) Serial mode register mn (SMRmn)



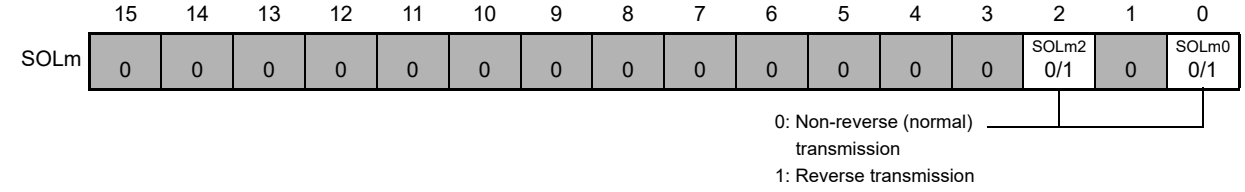
b) Serial communication operation setting register mn (SCRmn)



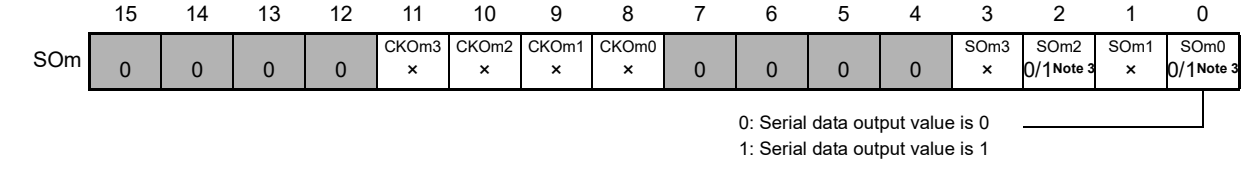
c) Serial data register mn (SDRmn) (lower 8 bits: TXDq)



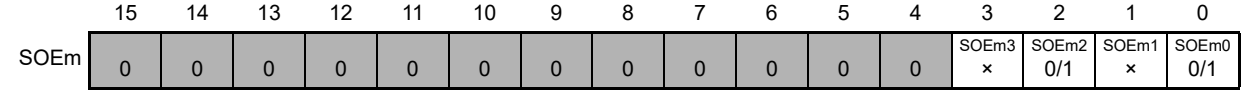
d) Serial output level register m (SOLm): Set only the bit of the target channel.



e) Serial output register m (SOm): Set only the bit of the target channel.



f) Serial output enable register m (SOEm): Set only the bit of the target channel to 1.



(Notes and Remarks are listed on the next page.)

g) Serial channel start register m (SSm): Set only the bit of the target channel to 1.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	SSm3 ×	SSm2 0/1	SSm1 ×	SSm0 0/1

Note 1. This bit is only present in the SCR00 register, and is fixed to 1 for the other registers.

Note 2. When UART performs 9-bit communication, bits 0 to 8 of the SDRm0 register are used as the transmission data specification area. Only UART0 supports the 9-bit data length.

Note 3. Before transmission is started, be sure to set this bit to 1 when the SOLmn bit of the target channel is set to 0, and set this bit to 0 when the SOLmn bit of the target channel is set to 1. The value varies depending on the communication data during communication operation.

Remark 1. m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), q: UART number (q = 0 to 2), mn = 00, 02, 10

Remark 2. ■: Setting is fixed in the UART transmission mode, ■: Setting disabled (set to the initial value)
0/1: Set to 0 or 1 depending on the user's application.

2. Operation procedure

Figure 11 - 77 Initial Setting Procedure for UART Transmission

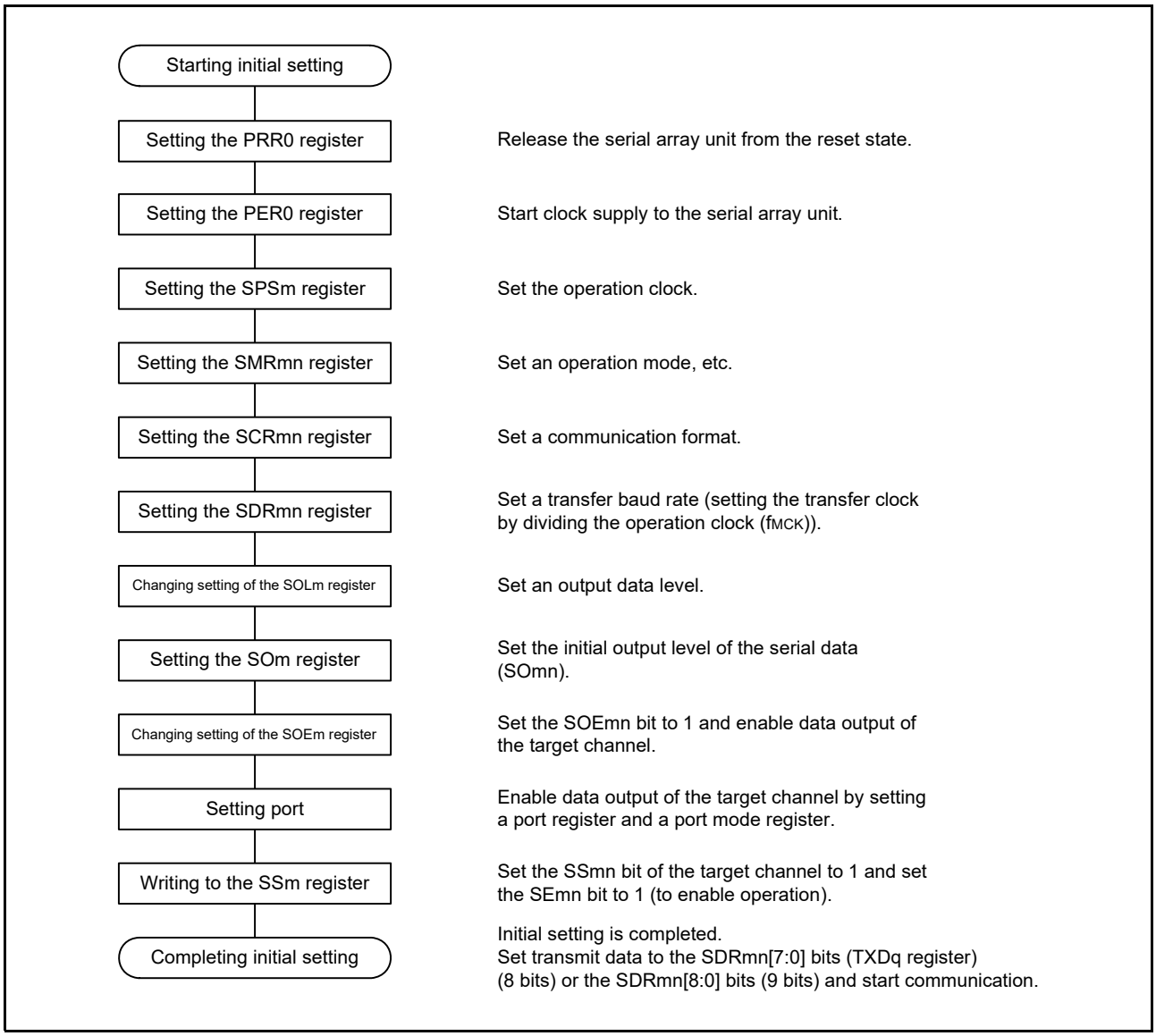


Figure 11 - 78 Procedure for Stopping UART Transmission

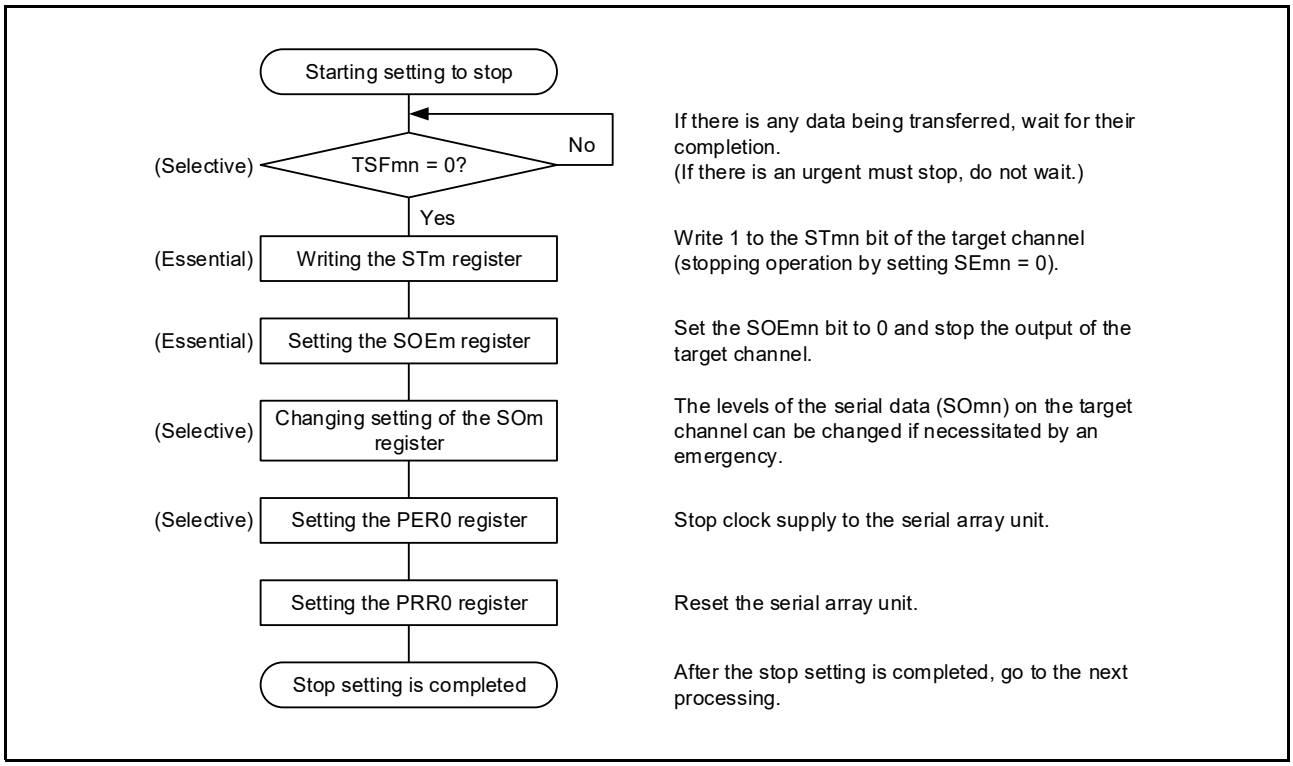
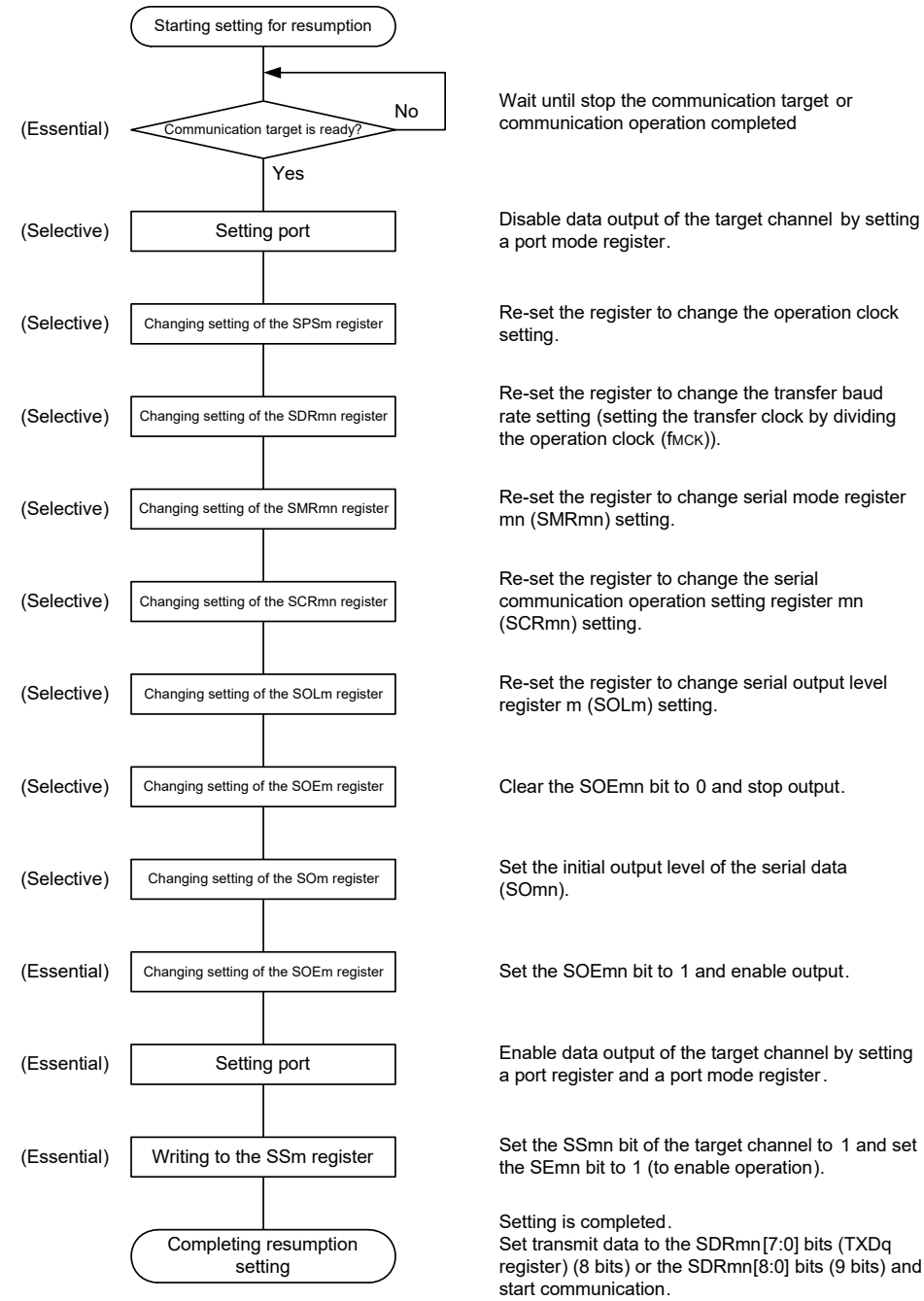


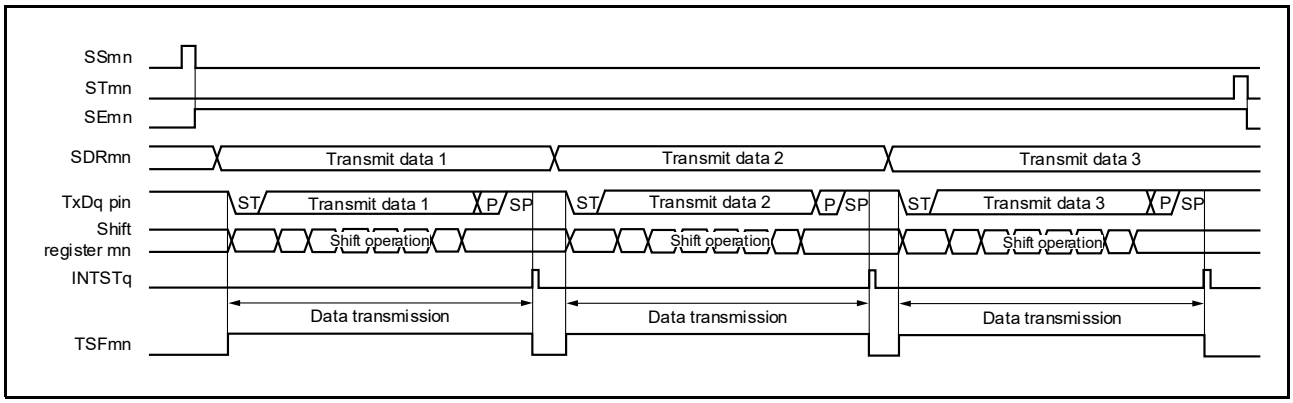
Figure 11 - 79 Procedure for Resuming UART Transmission



Remark If the PRR0 register is rewritten while stopping the communication to reset the serial array unit, wait until the communication target stops or communication finishes, and then perform initialization instead of restarting the communication.

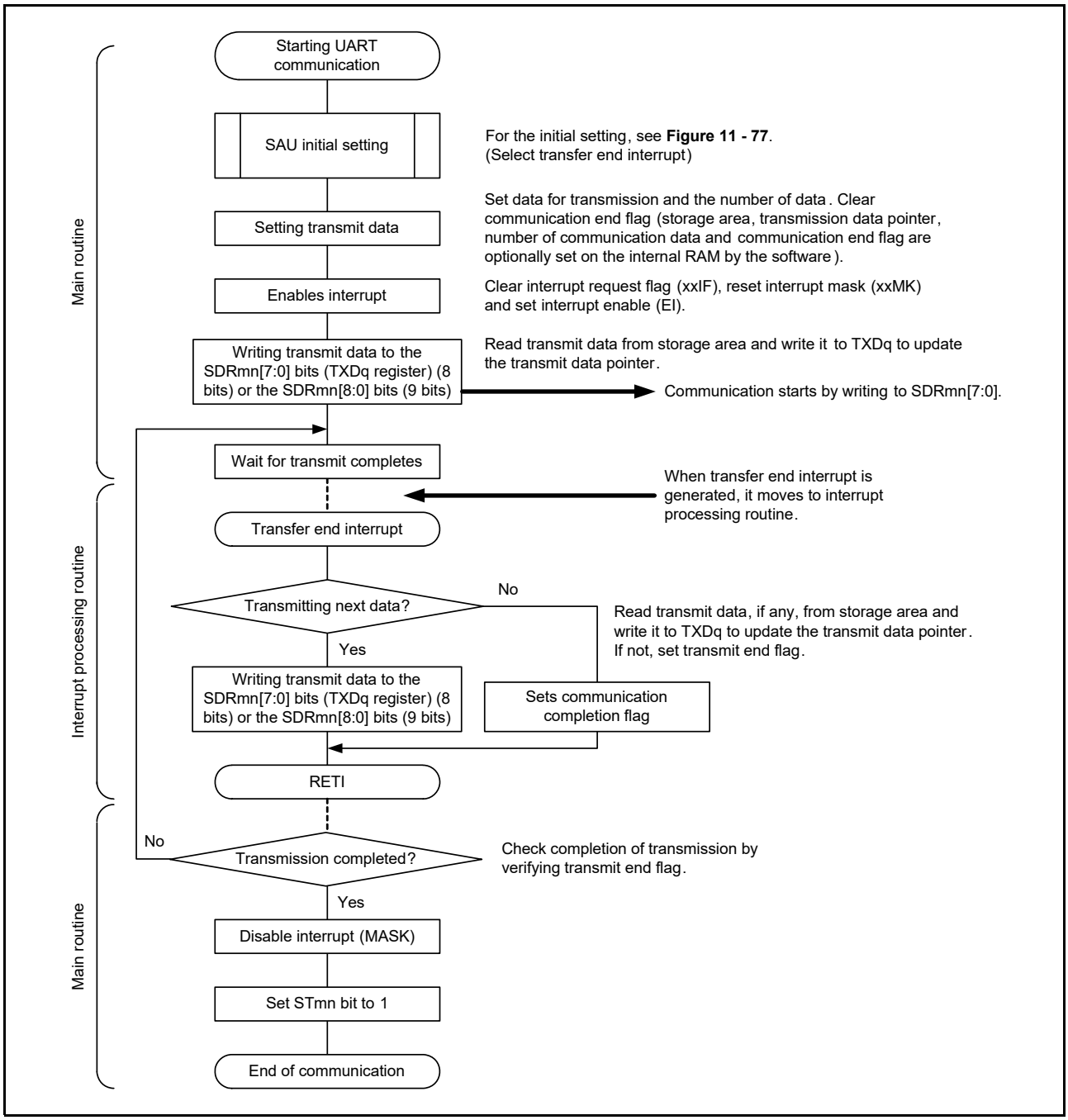
3. Processing flow (in single-transmission mode)

Figure 11 - 80 Timing Chart of UART Transmission (in Single-transmission Mode)



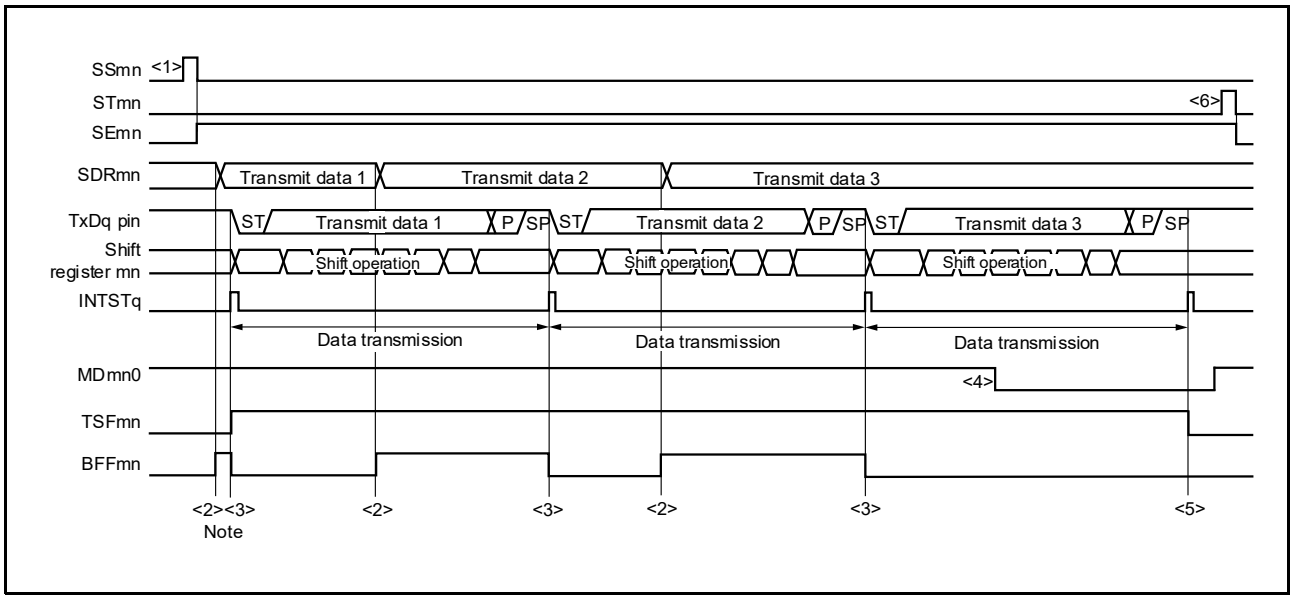
Remark m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), q: UART number (q = 0 to 2), mn = 00, 02, 10

Figure 11 - 81 Flowchart of UART Transmission (in Single-transmission Mode)



4. Processing flow (in continuous transmission mode)

Figure 11 - 82 Timing Chart of UART Transmission (in Continuous Transmission Mode)

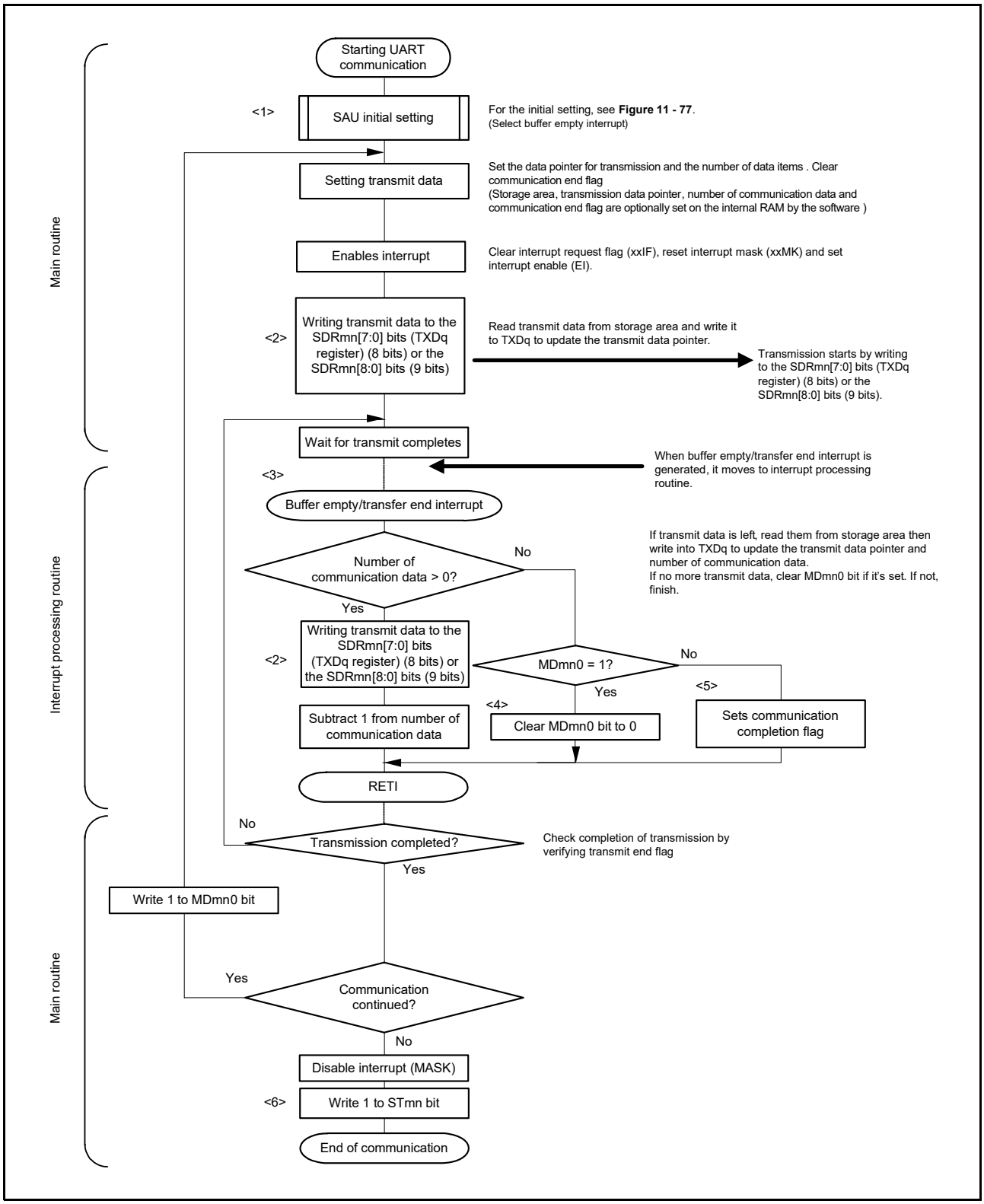


Note If transmit data is written to the SDRmn register while the BFFmn bit of serial status register mn (SSRmn) is 1 (valid data is stored in serial data register mn (SDRmn)), the transmit data is overwritten.

Caution **The MDmn0 bit of serial mode register mn (SSRmn) can be rewritten even during operation. However, rewrite it before transfer of the last bit is started, so that it has been rewritten before the transfer end interrupt of the last transmit data.**

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), q: UART number (q = 0 to 2),
mn = 00, 02, 10

Figure 11 - 83 Flowchart of UART Transmission (in Continuous Transmission Mode)



Remark <1> to <6> in the figure correspond to <1> to <6> in **Figure 11 - 82 Timing Chart of UART Transmission (in Continuous Transmission Mode)**.

11.6.2 UART reception

UART reception is an operation wherein this MCU asynchronously receives data from another device (start-stop synchronization). For UART reception, the odd-number channel of the two channels used for UART is used. The SMR register of both the odd- and even-numbered channels must be set.

UART	UART0	UART1	UART2
Target channel	Channel 1 of SAU0	Channel 3 of SAU0	Channel 1 of SAU1
Pins used	RxD0	RxD1	RxD2
Interrupt	INTSR0	INTSR1	INTSR2
	Transfer end interrupt only (Setting the buffer empty interrupt is prohibited.)		
Error interrupt	INTSRE0	INTSRE1	INTSRE2
Error detection flag	• Framing error detection flag (FEFmn) • Parity error detection flag (PEFmn) • Overrun error detection flag (OVFmn)		
Transfer data length	7, 8, or 9 bits**Note 1**		
Transfer rate**Note 2**	Max. f_{MCK}/6 [bps] (SDRmn[15:9] = 2 or more), Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [bps]		
Data phase	Non-reverse output (default: high level) Reverse output (default: low level)		
Parity bit	The following selectable • No parity bit (no parity check) • No parity judgment (0 parity) • Even parity check • Odd parity check		
Stop bit	Appending 1 bit		
Data direction	MSB or LSB first		

Note 1. Only UART0 supports the 9-bit data length.

Note 2. Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics specified in the electrical characteristics. For details, see **Section 28 Electrical Characteristics**.

Remark 1. f_{MCK}: Operation clock frequency of target channel
f_{CLK}: CPU/peripheral hardware clock frequency

Remark 2. m: Unit number (m = 0, 1), n: Channel number (n = 1, 3), mn = 01, 03, 11

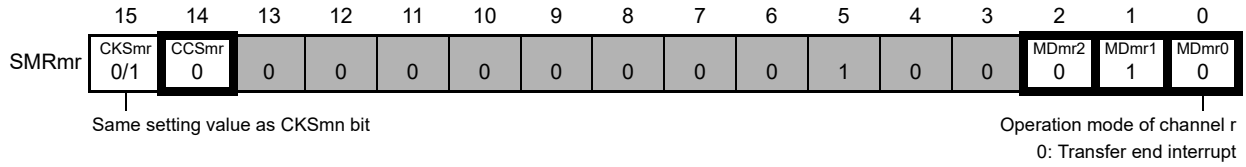
1. Register setting

Figure 11 - 84 Example of Contents of Registers for UART Reception of UART (UART0 to UART3)

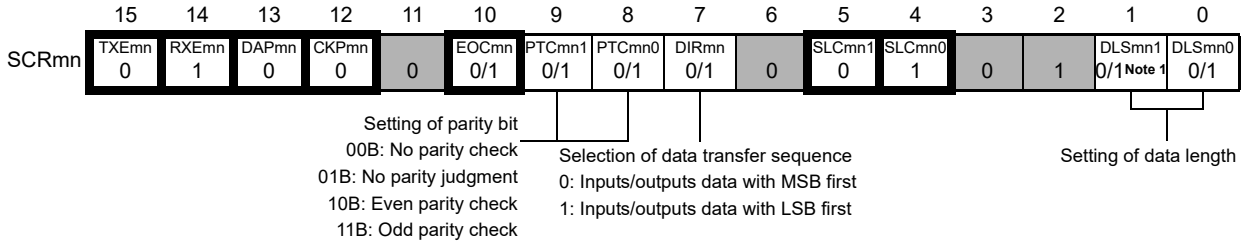
a) Serial mode register mn (SMRmn)



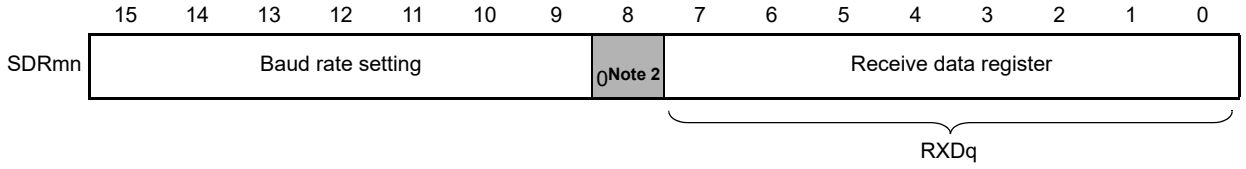
b) Serial mode register mr (SMRmr)



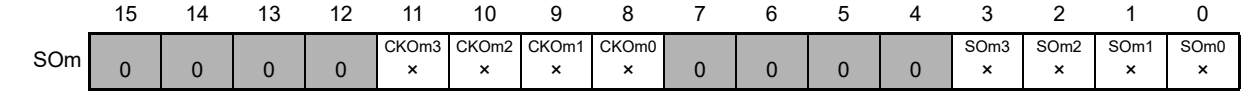
c) Serial communication operation setting register mn (SCRmn)



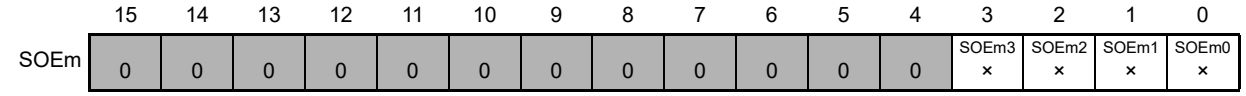
d) Serial data register mn (SDRmn) (lower 8 bits: RXDq)



e) Serial output register m (SOm): This register is not used in this mode.



f) Serial output enable register m (SOEm): This register is not used in this mode.



(Notes, Caution, and Remarks are listed on the next page.)

g) Serial channel start register m (SSm): Set only the bit of the target channel to 1.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	SSm3 0/1	SSm2 ×	SSm1 0/1	SSm0 ×

Note 1. This bit is only present in the SCR01 register, and is fixed to 1 for the other registers.

Note 2. When UART performs 9-bit communication, bits 0 to 8 of the SDRm1 register are used as the reception data specification area. Only UART0 supports the 9-bit data length.

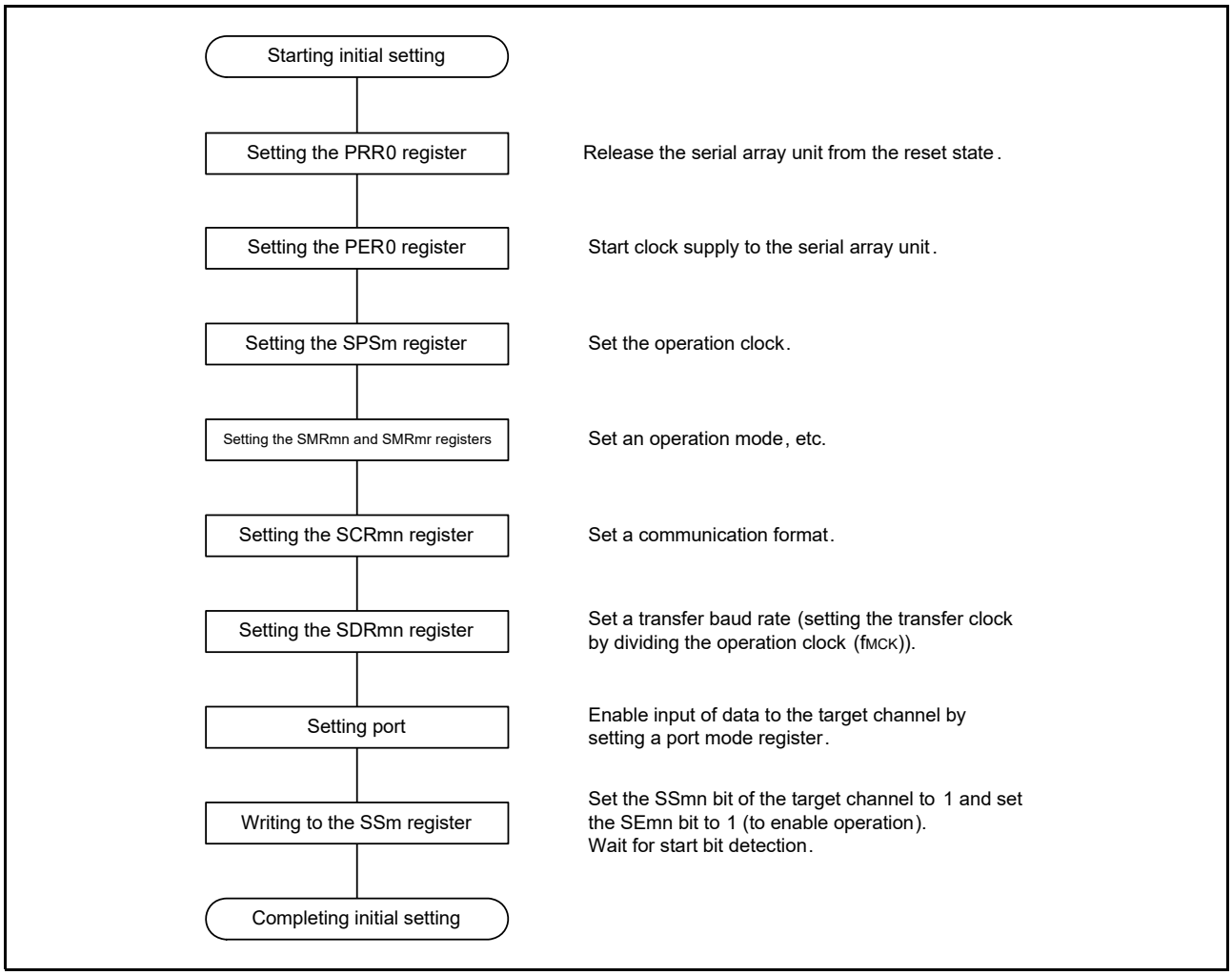
Caution **For the UART reception, be sure to set the SMRmr register of channel r to UART transmission mode that is to be paired with channel n.**

Remark 1. m: Unit number (m = 0, 1), n: Channel number (n = 1, 3), r: Channel number (r = n – 1), q: UART number (q = 0 to 2)
mn = 01, 03, 11

Remark 2. ■: Setting is fixed in the UART reception mode, ■: Setting disabled (set to the initial value)
×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)
0/1: Set to 0 or 1 depending on the user's application.

2. Operation procedure

Figure 11 - 85 Initial Setting Procedure for UART Reception



Caution Set the RXEmn bit of SCRmn register to 1, and then be sure to set SSmn to 1 after at least 4 fMCK clock cycles have elapsed.

Figure 11 - 86 Procedure for Stopping UART Reception

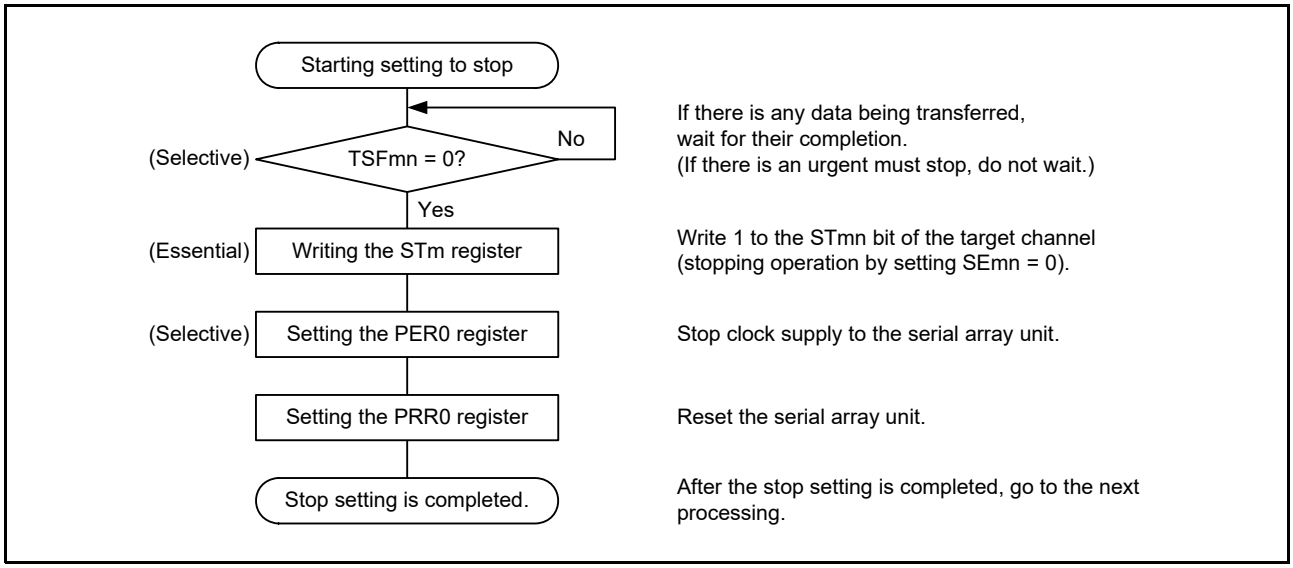
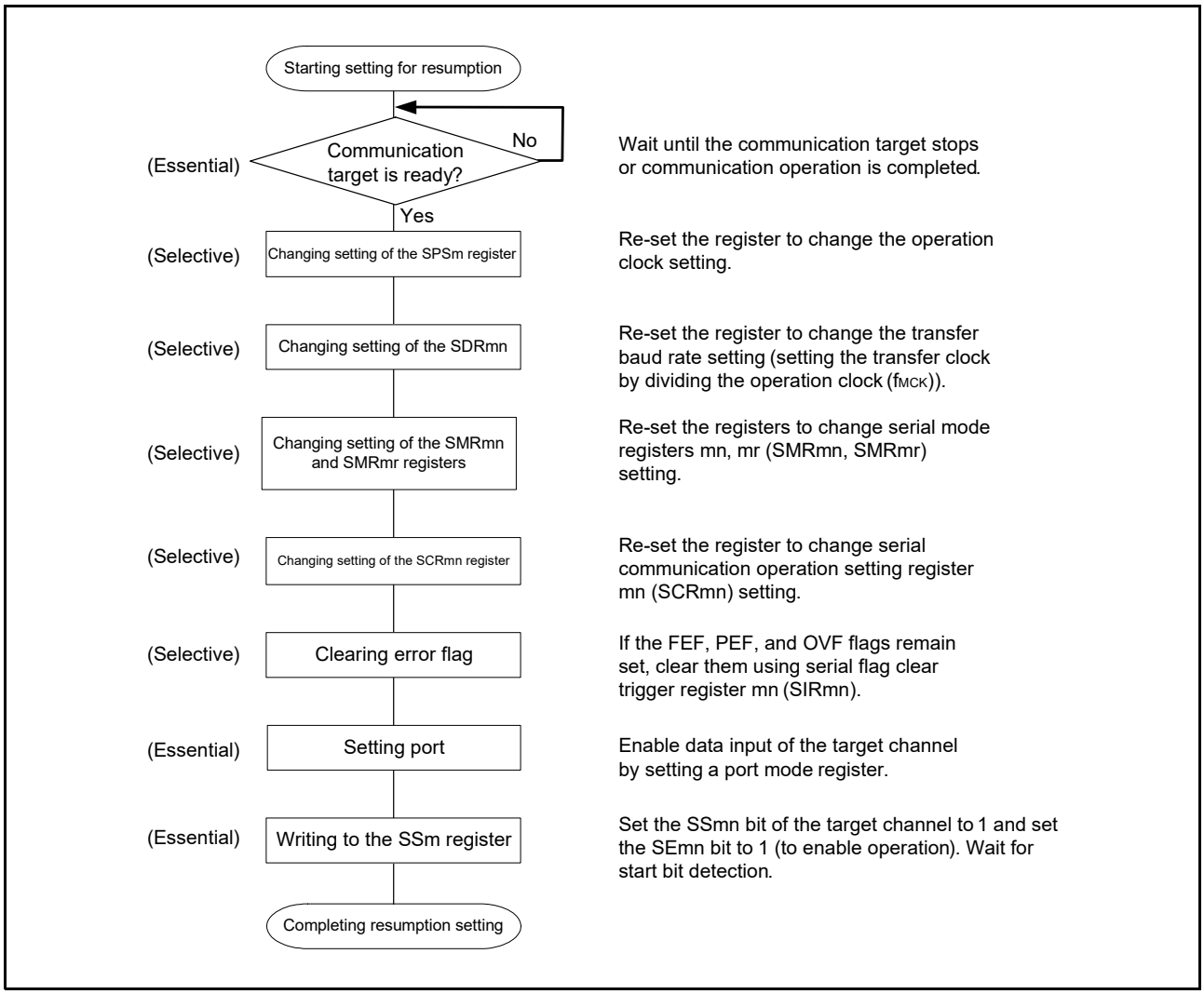


Figure 11 - 87 Procedure for Resuming UART Reception

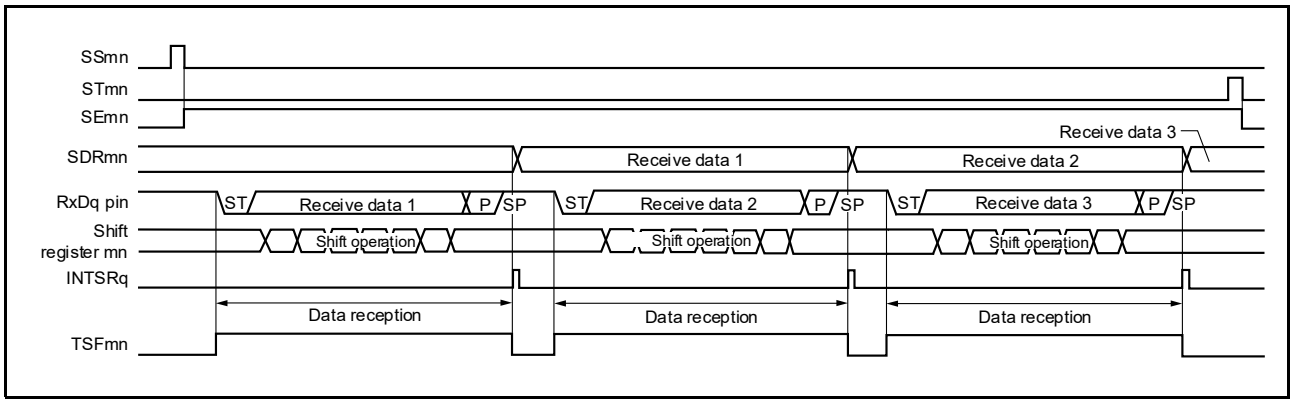


Caution **Set the RXEmn bit of SCRmn register to 1, and then be sure to set SSmn to 1 after at least 4 fMCK clocks have elapsed.**

Remark If the PRR0 register is rewritten while stopping the communication to reset the serial array unit, wait until the communication target stops or communication finishes, and then perform initialization instead of restarting the communication.

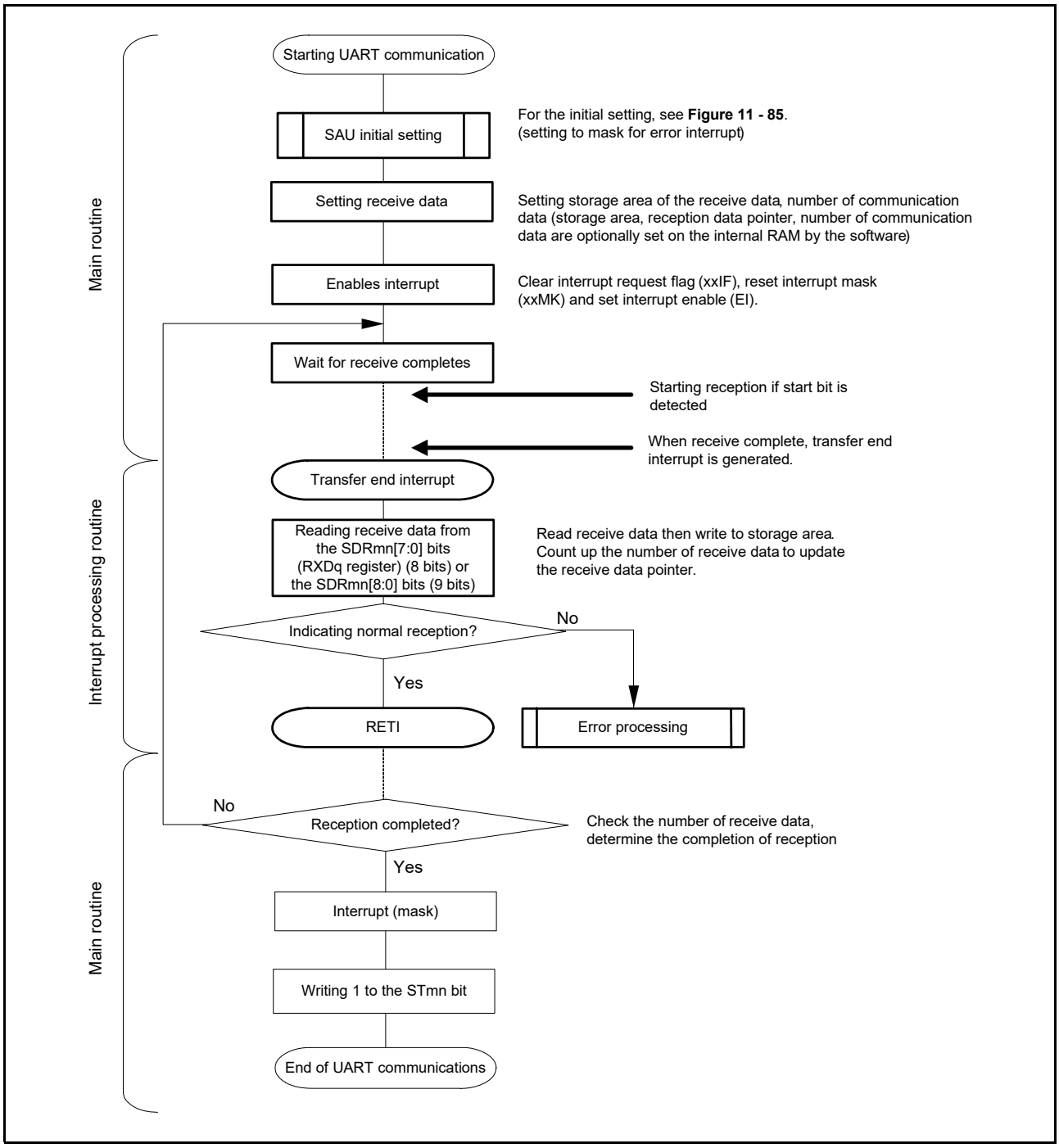
3. Processing flow

Figure 11 - 88 Timing Chart of UART Reception



Remark m: Unit number (m = 0, 1), n: Channel number (n = 1, 3), r: Channel number (r = n – 1), q: UART number (q = 0 to 2)
mn = 01, 03, 11

Figure 11 - 89 Flowchart of UART Reception



11.6.3 SNOOZE mode

The SNOOZE mode makes the UART perform reception operations upon RxDq pin input detection while in the STOP mode. Normally the UART stops communication in the STOP mode. However, using the SNOOZE mode enables the
<R> UART to perform reception operations without CPU operation on detection of input on the RxDq pin. The SNOOZE mode can only be set in the UART0 reception.

When using UARTq in the SNOOZE mode, make the following settings before entering the STOP mode. See **Figure 11 - 92 Flowchart of SNOOZE Mode Operation (EOCm1 = 0, SSECm = 0/1 or EOCm1 = 1, SSECm = 0)** and **Figure 11 - 94 Flowchart of SNOOZE Mode Operation (EOCm1 = 1, SSECm = 1)**.

- In the SNOOZE mode, the baud rate setting for UART reception needs to be changed to a value different from that in normal operation. Set the SPSm register and bits 15 to 9 of the SDRmn register with reference to **Table 11 - 5**.
- Set the EOCmn and SSECm bits. This is for enabling or stopping generation of an error interrupt (INTSREq) when a communication error occurs.
- When using the SNOOZE mode function, set the SWCm bit of serial standby control register m (SSCm) to 1 just before switching to the STOP mode. After the initial setting has been completed, set the SSm1 bit of serial channel start register m (SSm) to 1.
- A UARTq starts reception in SNOOZE mode on detecting input of the start bit on the RxDq pin following a transition of the CPU to the STOP mode.

Caution 1. The SNOOZE mode can only be used when the high-speed on-chip oscillator clock is selected for f_{CLK}.

Caution 2. The maximum transfer rate in the SNOOZE mode is 9.6 kbps when $f_{CLK} = f_{IH}$ (32 MHz).

Caution 3. When SWCm = 1, UARTq can be used only when the reception operation is started in the STOP mode. When used simultaneously with another SNOOZE mode function or interrupt, if the reception operation is started in a state other than the STOP mode, such as those given below, data may not be received correctly and a framing error or parity error may be generated.

- **When after the SWCm bit has been set to 1, the reception operation is started before the STOP mode is entered**
- **When the reception operation is started while another function is in the SNOOZE mode**
- **When after returning from the STOP mode to normal operation due to an interrupt or other cause, the reception operation is started before the SWCm bit is returned to 0**

Caution 4. If a parity error, framing error, or overrun error occurs while the SSECm bit is set to 1, the PEFmn, FEFmn, or OVFmn flag is not set and an error interrupt (INTSREq) is not generated. Therefore, when the setting of SSECm = 1 is made, clear the PEFmn, FEFmn, and OVFmn flags before setting the SWCm bit to 1 and read the value in bits 7 to 0 (RxDq register) of the SDRm1 register.

Caution 5. The CPU shifts from the STOP mode to the SNOOZE mode on detecting the valid edge of the RxDq signal. Note, however, that reception through the UART channel may not start and the CPU may remain in the SNOOZE mode if an input pulse on the RxDq pin is too short to be detected as a start bit. In such cases, data may not be received correctly, and this may lead to a framing error or parity error in the next UART reception.

Table 11 - 5 Baud Rate Setting for UART Reception in SNOOZE Mode

Baud Rate	High-speed On-chip Oscillator (f_{IH})	Operating Clock (f_{MCK})	SDRmn[15:9]	Maximum Permissible Value	Minimum Permissible Value
4800 bps	32 MHz ±1%**Note**	$f_{CLK}/2^5$	106	1.45%	-1.67%
	24 MHz ±1%**Note**	$f_{CLK}/2^5$	79	1.77%	-1.37%
9600 bps	32 MHz ±1%**Note**	$f_{CLK}/2^4$	106	1.45%	-1.67%
	24 MHz ±1%**Note**	$f_{CLK}/2^4$	79	1.77%	-1.37%

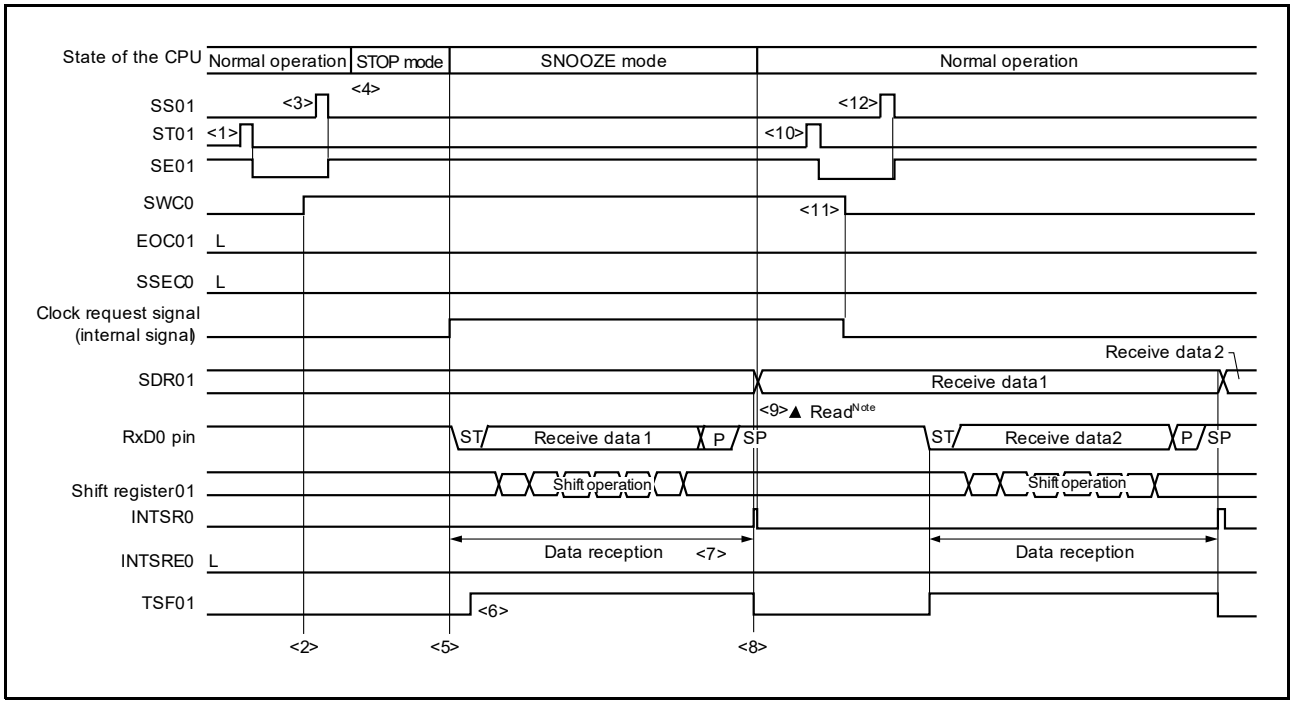
Note When the accuracy of the clock frequency of the high-speed on-chip oscillator is ±1.5% or ±2.0%, the permissible range becomes smaller as shown below.

- In the case of f_{IH} ±1.5%, perform (Maximum permissible value – 0.5%) and (Minimum permissible value + 0.5%) to the values in the above table.
- In the case of f_{IH} ±2.0%, perform (Maximum permissible value – 1.0%) and (Minimum permissible value + 1.0%) to the values in the above table.

Remark The maximum permissible value and minimum permissible value are permissible values for the baud rate in UART reception.

1. SNOOZE mode operation (EOCm1 = 0, SSECm = 0/1)
   Because of the setting of EOCm1 = 0, even though a communication error occurs, an error interrupt (INTSREq) is not generated, regardless of the setting of the SSECm bit. A transfer end interrupt (INTSRq) will be generated.

Figure 11 - 90 Timing Chart of SNOOZE Mode Operation (EOCm1 = 0, SSECm = 0/1)



Note Read the received data when SWCm = 1.

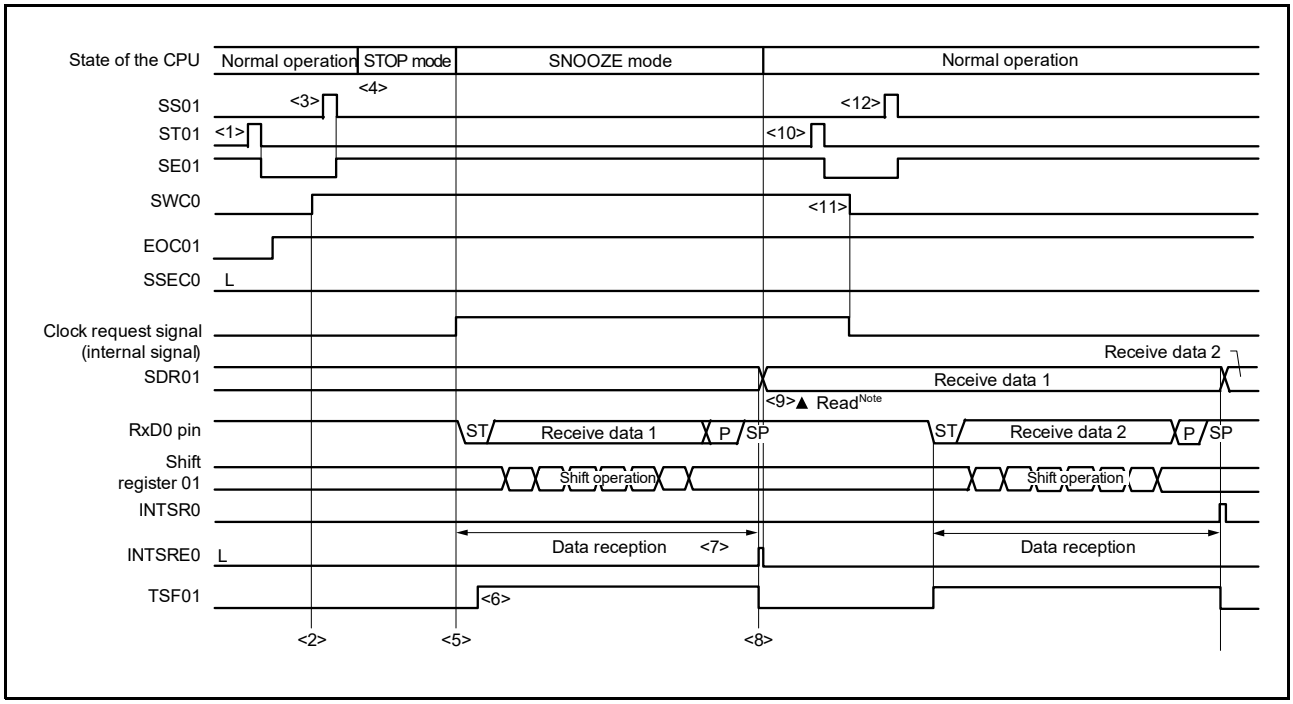
Caution **Before switching to the SNOOZE mode or after reception operation in the SNOOZE mode finishes, set the STm1 bit to 1 (the SEm1 bit is cleared and the operation stops). After the receive operation completes, also clear the SWCm bit to 0 (SNOOZE mode release).**

Remark 1. <1> to <12> in the figure correspond to <1> to <12> in **Figure 11 - 92 Flowchart of SNOOZE Mode Operation (EOCm1 = 0, SSECm = 0/1 or EOCm1 = 1, SSECm = 0)**.

Remark 2. m = 0, q = 0

2. SNOOZE mode operation (EOCm1 = 1, SSECm = 0: Error interrupt (INTSREq) generation is enabled)
   Because EOCm1 = 1 and SSECm = 0, an error interrupt (INTSREq) is generated when a communication error occurs.

Figure 11 - 91 Timing Chart of SNOOZE Mode Operation (EOCm1 = 1, SSECm = 0)



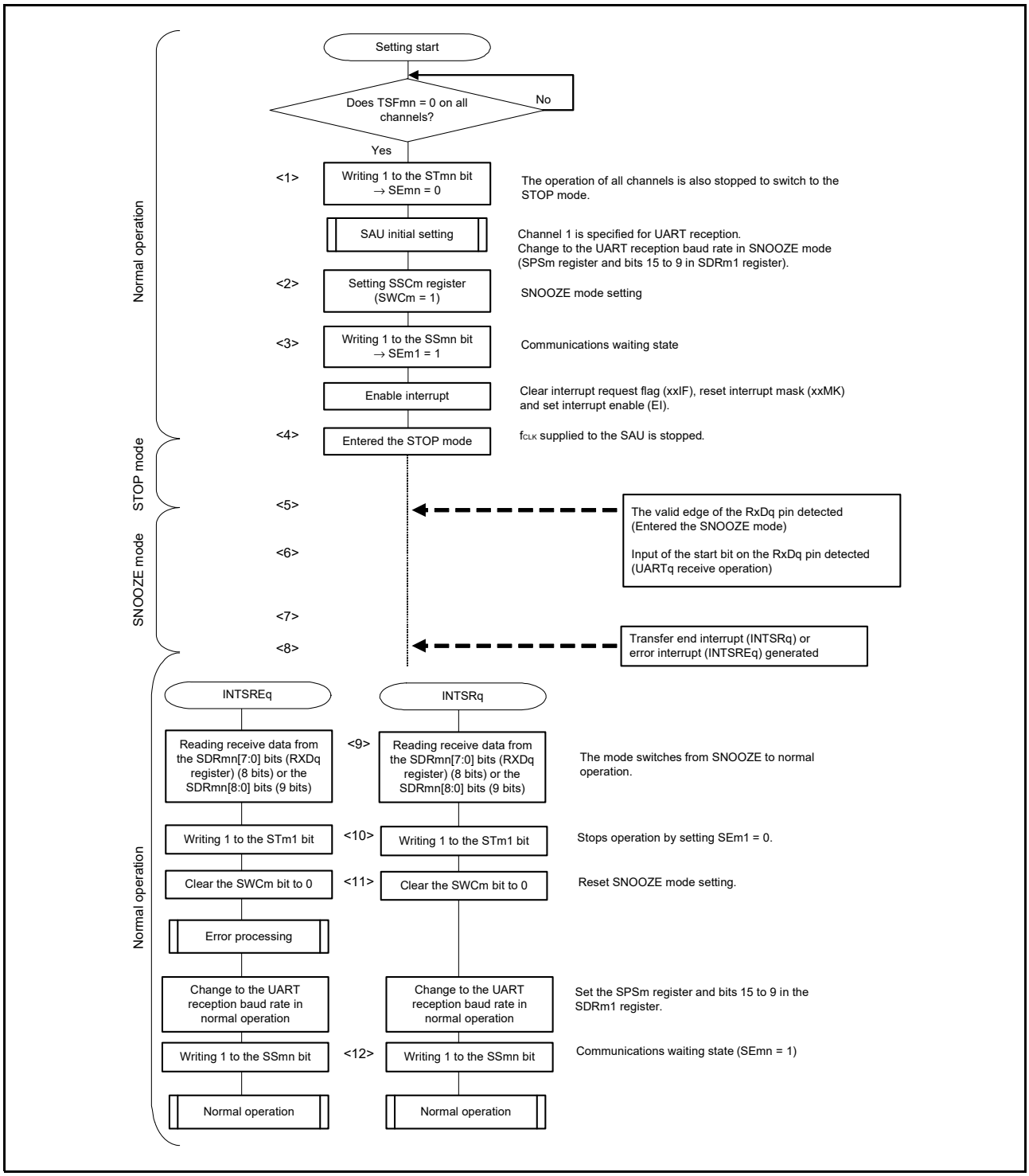
Note Read the received data when SWCm = 1.

Caution **Before switching to the SNOOZE mode or after reception operation in the SNOOZE mode finishes, set the STm1 bit to 1 (the SEm1 bit is cleared and the operation stops). After the receive operation completes, also clear the SWCm bit to 0 (SNOOZE mode release).**

Remark 1. <1> to <12> in the figure correspond to <1> to <12> in **Figure 11 - 92 Flowchart of SNOOZE Mode Operation (EOCm1 = 0, SSECm = 0/1 or EOCm1 = 1, SSECm = 0)**.

Remark 2. m = 0, q = 0

Figure 11 - 92 Flowchart of SNOOZE Mode Operation (EOCm1 = 0, SSECm = 0/1 or EOCm1 = 1, SSECm = 0)

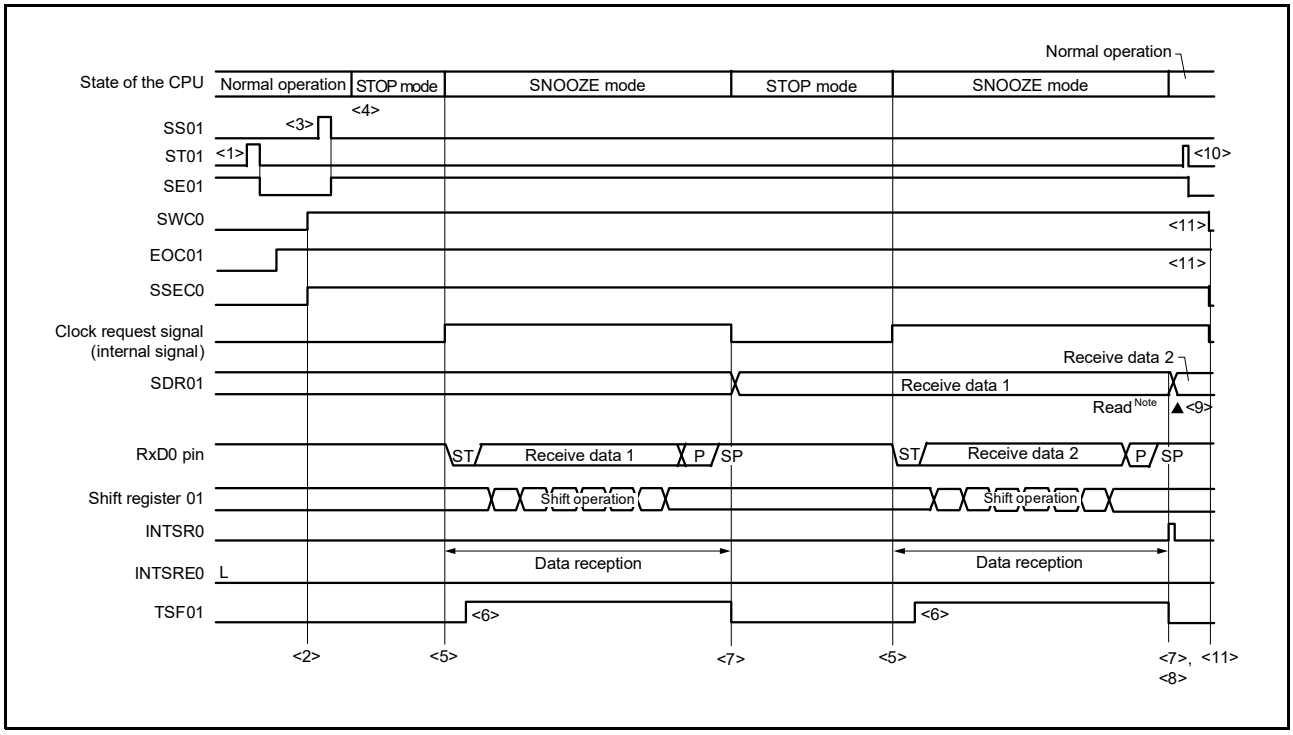


Remark 1. <1> to <12> in the figure correspond to <1> to <12> in **Figure 11 - 90 Timing Chart of SNOOZE Mode Operation (EOCm1 = 0, SSECm = 0/1)** and **Figure 11 - 91 Timing Chart of SNOOZE Mode Operation (EOCm1 = 1, SSECm = 0)**.

Remark 2. m = 0, q = 0, n = 0 to 3

3. SNOOZE mode operation (EOCm1 = 1, SSECm = 1: Error interrupt (INTSREq) generation is stopped)
   Because EOCm1 = 1 and SSECm = 1, an error interrupt (INTSREq) is not generated when a communication error occurs.

Figure 11 - 93 Timing Chart of SNOOZE Mode Operation (EOCm1 = 1, SSECm = 1)



Note Read the received data when SWCm = 1.

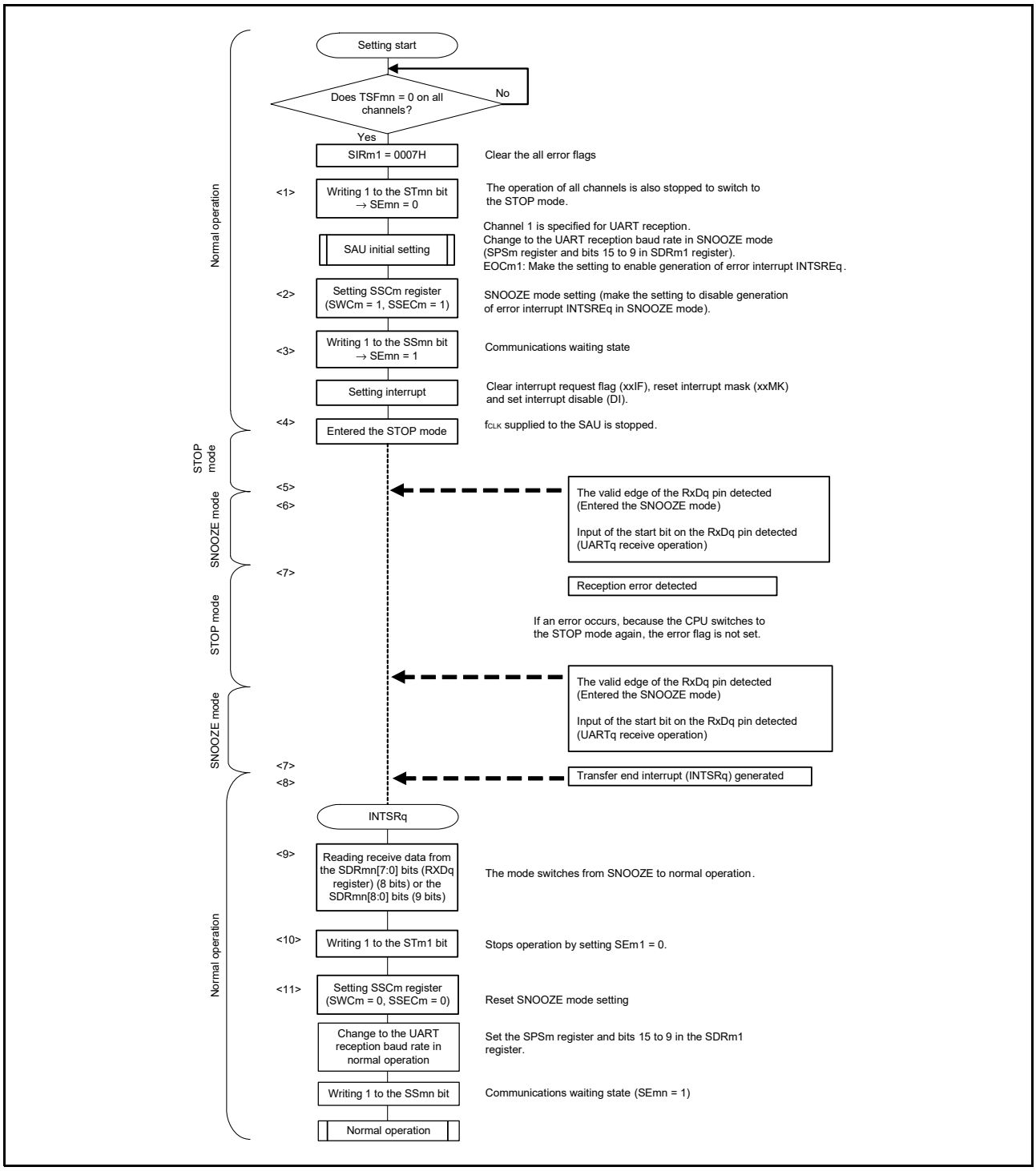
Caution 1. **Before switching to the SNOOZE mode or after reception operation in the SNOOZE mode finishes, set the STm1 bit to 1 (the SEm1 bit is cleared and the operation stops). After the receive operation completes, also clear the SWCm bit to 0 (SNOOZE mode release).**

Caution 2. **If a parity error, framing error, or overrun error occurs while the SSECm bit is set to 1, the PEFm1, FEFm1, or OVFm1 flag is not set and an error interrupt (INTSREq) is not generated. Therefore, when the setting of SSECm = 1 is made, clear the PEFm1, FEFm1, and OVFm1 flags before setting the SWCm bit to 1 and read the value in SDRm1[7:0] (RxDq register) (8 bits) or SDRm1[8:0] (9 bits).**

Remark 1. <1> to <11> in the figure correspond to <1> to <11> in **Figure 11 - 94 Flowchart of SNOOZE Mode Operation (EOCm1 = 1, SSECm = 1)**.

Remark 2. m = 0, q = 0

Figure 11 - 94 Flowchart of SNOOZE Mode Operation (EOCm1 = 1, SSECm = 1)



Caution **If a parity error, framing error, or overrun error occurs while the SSECm bit is set to 1, the PEFm1, FEFm1, or OVFm1 flag is not set and an error interrupt (INTSREq) is not generated. Therefore, when the setting of SSECm = 1 is made, clear the PEFm1, FEFm1, and OVFm1 flags before setting the SWCm bit to 1 and read the value in SDRm1[7:0] (RxDq register) (8 bits) or SDRm1[8:0] (9 bits).**

Remark 1. <1> to <11> in the figure correspond to <1> to <11> in **Figure 11 - 93 Timing Chart of SNOOZE Mode Operation (EOCm1 = 1, SSECm = 1)**.

Remark 2. m = 0, q = 0, n = 0 to 3

11.6.4 Calculating baud rate

1. Baud rate calculation expression

The baud rate for UART (UART0 to UART2) communication can be calculated by the following expressions.

(Baud rate) = {Operation clock (f_{MCK}) frequency of target channel} ÷ (SDRmn[15:9]+1) ÷ 2 [bps]

Caution Setting serial data register mn (SDRmn) SDRmn[15:9] = (0000000B, 0000001B) is prohibited.

Remark 1. When UART is used, the value of SDRmn[15:9] is the value of bits 15 to 9 of the SDRmn register (0000010B to 1111111B) and therefore is 2 to 127.

Remark 2. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

The operation clock (f_{MCK}) is determined by serial clock select register m (SPSm) and the CKSmn bit of serial mode register mn (SMRmn).

Table 11 - 6 Selection of Operation Clock For UART

SMRmn Register	SPSm Register								Operation Clock (f_{MCK})[Note]	
CKSmn	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00		f_{CLK} = 32 MHz
0	×	×	×	×	0	0	0	0	f_{CLK}	32 MHz
	×	×	×	×	0	0	0	1	$f_{CLK}/2$	16 MHz
	×	×	×	×	0	0	1	0	$f_{CLK}/2^2$	8 MHz
	×	×	×	×	0	0	1	1	$f_{CLK}/2^3$	4 MHz
	×	×	×	×	0	1	0	0	$f_{CLK}/2^4$	2 MHz
	×	×	×	×	0	1	0	1	$f_{CLK}/2^5$	1 MHz
	×	×	×	×	0	1	1	0	$f_{CLK}/2^6$	500 kHz
	×	×	×	×	0	1	1	1	$f_{CLK}/2^7$	250 kHz
	×	×	×	×	1	0	0	0	$f_{CLK}/2^8$	125 kHz
	×	×	×	×	1	0	0	1	$f_{CLK}/2^9$	62.5 kHz
	×	×	×	×	1	0	1	0	$f_{CLK}/2^{10}$	31.25 kHz
	×	×	×	×	1	0	1	1	$f_{CLK}/2^{11}$	15.63 kHz
	×	×	×	×	1	1	0	0	$f_{CLK}/2^{12}$	7.81 kHz
	×	×	×	×	1	1	0	1	$f_{CLK}/2^{13}$	3.91 kHz
	×	×	×	×	1	1	1	0	$f_{CLK}/2^{14}$	1.95 kHz
	×	×	×	×	1	1	1	1	$f_{CLK}/2^{15}$	977 Hz
1	0	0	0	0	×	×	×	×	f_{CLK}	32 MHz
	0	0	0	1	×	×	×	×	$f_{CLK}/2$	16 MHz
	0	0	1	0	×	×	×	×	$f_{CLK}/2^2$	8 MHz
	0	0	1	1	×	×	×	×	$f_{CLK}/2^3$	4 MHz
	0	1	0	0	×	×	×	×	$f_{CLK}/2^4$	2 MHz
	0	1	0	1	×	×	×	×	$f_{CLK}/2^5$	1 MHz
	0	1	1	0	×	×	×	×	$f_{CLK}/2^6$	500 kHz
	0	1	1	1	×	×	×	×	$f_{CLK}/2^7$	250 kHz
	1	0	0	0	×	×	×	×	$f_{CLK}/2^8$	125 kHz
	1	0	0	1	×	×	×	×	$f_{CLK}/2^9$	62.5 kHz
	1	0	1	0	×	×	×	×	$f_{CLK}/2^{10}$	31.25 kHz
	1	0	1	1	×	×	×	×	$f_{CLK}/2^{11}$	15.63 kHz
	1	1	0	0	×	×	×	×	$f_{CLK}/2^{12}$	7.81 kHz
	1	1	0	1	×	×	×	×	$f_{CLK}/2^{13}$	3.91 kHz
	1	1	1	0	×	×	×	×	$f_{CLK}/2^{14}$	1.95 kHz
	1	1	1	1	×	×	×	×	$f_{CLK}/2^{15}$	977 Hz
Other than above									Setting prohibited	

Note When changing the clock selected for f_{CLK} (by changing the system clock control register (CKC) value), do so after having stopped (serial channel stop register m (STm) = 000FH) the operation of the serial array unit (SAU).

Remark 1. ×: Don't care

Remark 2. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10,11

2. Baud rate error during transmission

The baud rate error of UART (UART0 to UART2) communication during transmission can be calculated by the following expression. Make sure that the baud rate at the transmission side is within the permissible baud rate range at the reception side.

(Baud rate error) = (Calculated baud rate value) ÷ (Target baud rate) × 100 - 100 [%]

Here is an example of setting a UART baud rate at f_{CLK} = 32 MHz.

UART Baud Rate (Target Baud Rate)	f_{CLK} = 32 MHz			
	Operation Clock (f_{MCK})	SDRmn[15:9]	Calculated Baud Rate	Error from Target Baud Rate
300 bps	$f_{CLK}/2^9$	103	300.48 bps	+0.16%
600 bps	$f_{CLK}/2^8$	103	600.96 bps	+0.16%
1200 bps	$f_{CLK}/2^7$	103	1201.92 bps	+0.16%
2400 bps	$f_{CLK}/2^6$	103	2403.85 bps	+0.16%
4800 bps	$f_{CLK}/2^5$	103	4807.69 bps	+0.16%
9600 bps	$f_{CLK}/2^4$	103	9615.38 bps	+0.16%
19200 bps	$f_{CLK}/2^3$	103	19230.8 bps	+0.16%
31250 bps	$f_{CLK}/2^3$	63	31250.0 bps	±0.0%
38400 bps	$f_{CLK}/2^2$	103	38461.5 bps	+0.16%
76800 bps	$f_{CLK}/2$	103	76923.1 bps	+0.16%
153600 bps	f_{CLK}	103	153846 bps	+0.16%
312500 bps	f_{CLK}	50	313725.5 bps	+0.39%

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), mn = 00, 02, 10

3. Permissible baud rate range for reception

The permissible baud rate range for reception during UART (UART0 to UART2) communication can be calculated by the following expression. Make sure that the baud rate at the transmission side is within the permissible baud rate range at the reception side.

$$\text{(Maximum receivable baud rate)} = \frac{2 \times k \times Nfr}{2 \times k \times Nfr - k + 2} \times Brate$$

$$\text{(Minimum receivable baud rate)} = \frac{2 \times k \times (Nfr - 1)}{2 \times k \times Nfr - k - 2} \times Brate$$

Brate: Calculated baud rate value at the reception side (See **11.6.4 Calculating baud rate**, **1. Baud rate calculation expression**.)

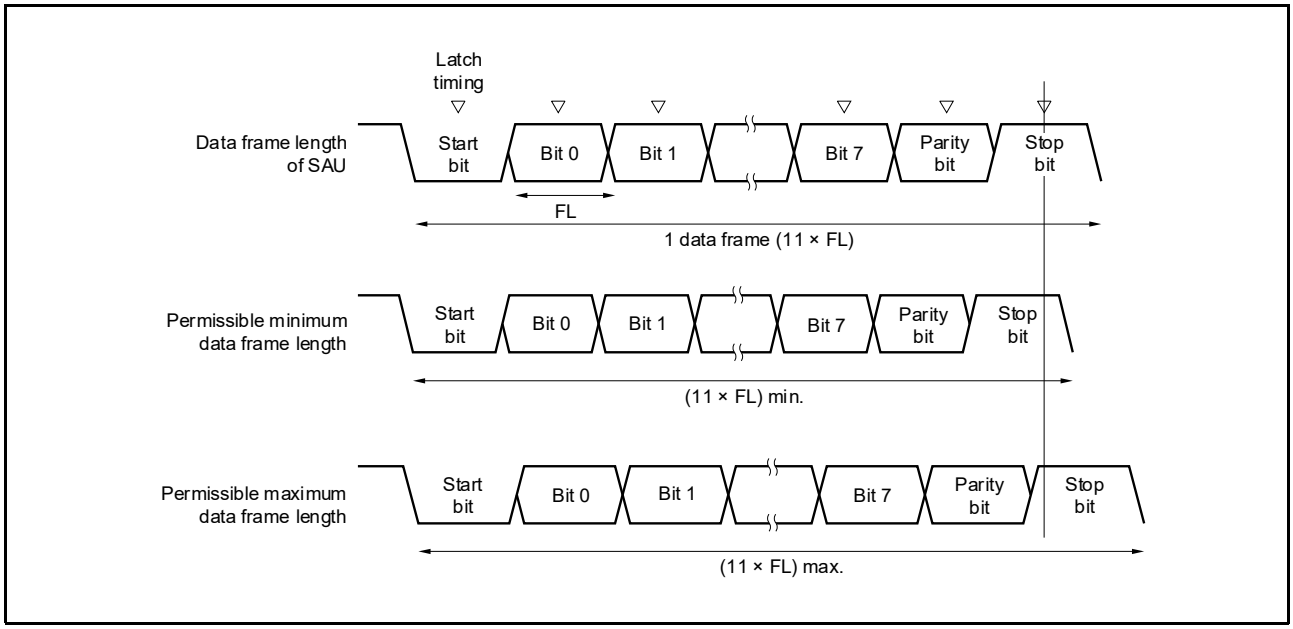
k: SDRmn[15:9] + 1

Nfr: 1 data frame length [bits]

= (Start bit) + (Data length) + (Parity bit) + (Stop bit)

Remark m: Unit number (m = 0, 1), n: Channel number (n = 1, 3), mn = 01, 03, 11

Figure 11 - 95 Permissible Baud Rate Range for Reception (1 Data Frame Length = 11 Bits)



The timing of latching receive data is determined by the division ratio set by bits 15 to 9 of serial data register mn (SDRmn) after the start bit is detected. If the last data (stop bit) is received before this latch timing, the data can be correctly received.

11.6.5 Procedure for processing errors that occurred during UART (UART0 to UART2) communication

Table 11 - 7 and **Table 11 - 8** describe the procedure for processing errors that occurred during UART (UART0 to UART2) communication.

Table 11 - 7 Processing Procedure in Case of Parity Error or Overrun Error

Software Manipulation	State of the Hardware	Remark
Reads serial data register mn (SDRmn). →	The BFFmn bit of the SSRmn register is set to 0 and channel n is enabled to receive data	This is to prevent an overrun error if the next reception is completed during error processing.
Reads serial status register mn (SSRmn).		The error type is identified and the read value is used to clear the error flag.
Writes 1 to serial flag clear trigger register mn (SIRmn). →	The error flag is cleared.	Only the error generated during reading can be cleared, by writing the value read from the SSRmn register to the SIRmn register without modification.

Table 11 - 8 Processing Procedure in Case of Framing Error

Software Manipulation	State of the Hardware	Remark
Reads serial data register mn (SDRmn). →	The BFFmn bit of the SSRmn register is set to 0 and channel n is enabled to receive data	This is to prevent an overrun error if the next reception is completed during error processing.
Reads serial status register mn (SSRmn).		The error type is identified and the read value is used to clear the error flag.
Writes serial flag clear trigger register mn (SIRmn). →	The error flag is cleared.	Only the error generated during reading can be cleared, by writing the value read from the SSRmn register to the SIRmn register without modification.
Sets the STmn bit of serial channel stop register m (STm) to 1. →	The SEmn bit of serial channel enable status register m (SEm) is set to 0 and channel n stops operation.	
Synchronization with other party of communication		Synchronization with the other party of communication is re-established and communication is resumed because it is considered that a framing error has occurred because the start bit has been shifted.
Sets the SSmn bit of serial channel start register m (SSm) to 1. →	The SEmn bit of serial channel enable status register m (SEm) is set to 1 and channel n is enabled to operate.	

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

11.7 LIN Communication Operation

11.7.1 LIN transmission

Of UART transmission, UART2 supports LIN communication. For LIN transmission, channel 0 of unit 1 is used.

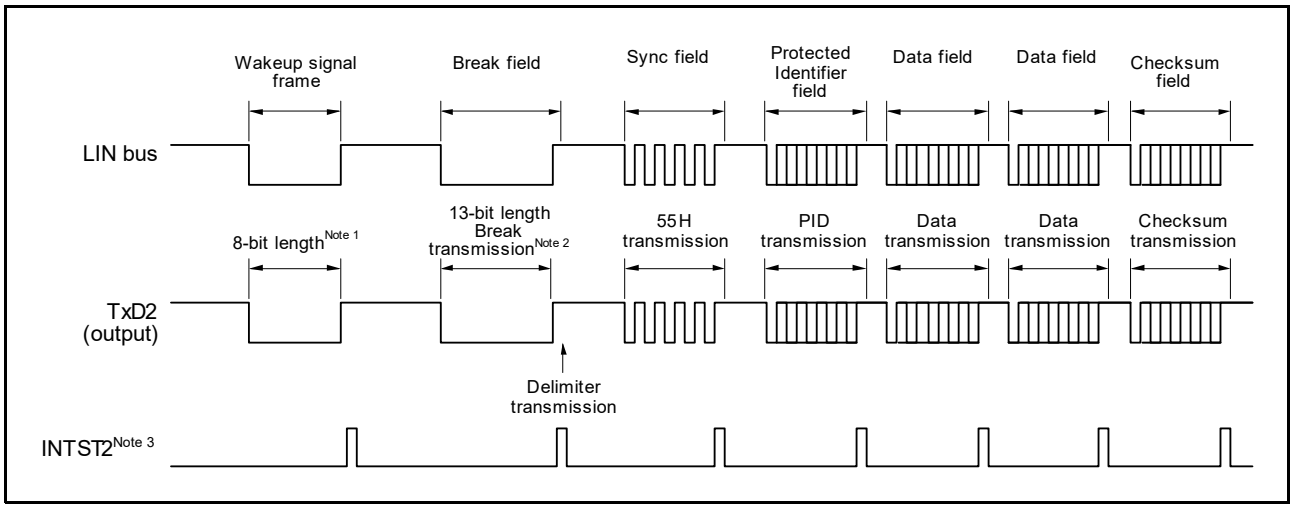
UART	UART0	UART1	UART2
Support of LIN communication	Not supported	Not supported	Supported
Target channel	—	—	Channel 0 of SAU1
Pins used	—	—	TxD2
Interrupt	—	—	INTST2
	Transfer end interrupt (in single-transfer mode) or buffer empty interrupt (in continuous transfer mode) can be selected.		
Error detection flag	None		
Transfer data length	8 bits		
Transfer rate**Note**	Max. f_{MCK}/6 [bps] (SDR10[15:9] = 2 or more), Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [bps]		
Data phase	Non-reverse output (default: high level) Reverse output (default: low level)		
Parity bit	No parity bit		
Stop bit	Appending 1 bit		
Data direction	LSB first		

Note Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics specified in the electrical characteristics. For details, see **Section 28 Electrical Characteristics**. In general, 2.4, 9.6, or 19.2 kbps is often used in LIN communication.

Remark f_{MCK}: Operation clock frequency of target channel
f_{CLK}: CPU/peripheral hardware clock frequency

LIN stands for Local Interconnect Network and is a low-speed (1 to 20 kbps) serial communication protocol designed to reduce the cost of an automobile network. Communication of LIN is single-master communication and up to 15 slaves can be connected to one master. The slaves are used to control switches, actuators, and sensors, which are connected to the master via LIN. Usually, the master is connected to a network such as CAN (Controller Area Network). A LIN bus is a single-wire bus to which nodes are connected via transceiver conforming to ISO9141. According to the protocol of LIN, the master transmits a frame by attaching baud rate information to it. A slave receives this frame and corrects a baud rate error from the master. If the baud rate error of a slave is within ±15%, communication can be established.
Figure 11 - 96 outlines a transmission operation of LIN.

Figure 11 - 96 Transmission Operation of LIN



Note 1. Set the baud rate in accordance with the wakeup signal regulations and transmit data of 80H.

Note 2. A break field is defined to have a width of 13 bits and output a low level. Where the baud rate for main transfer is N [bps], therefore, the baud rate of the break field is calculated as follows.

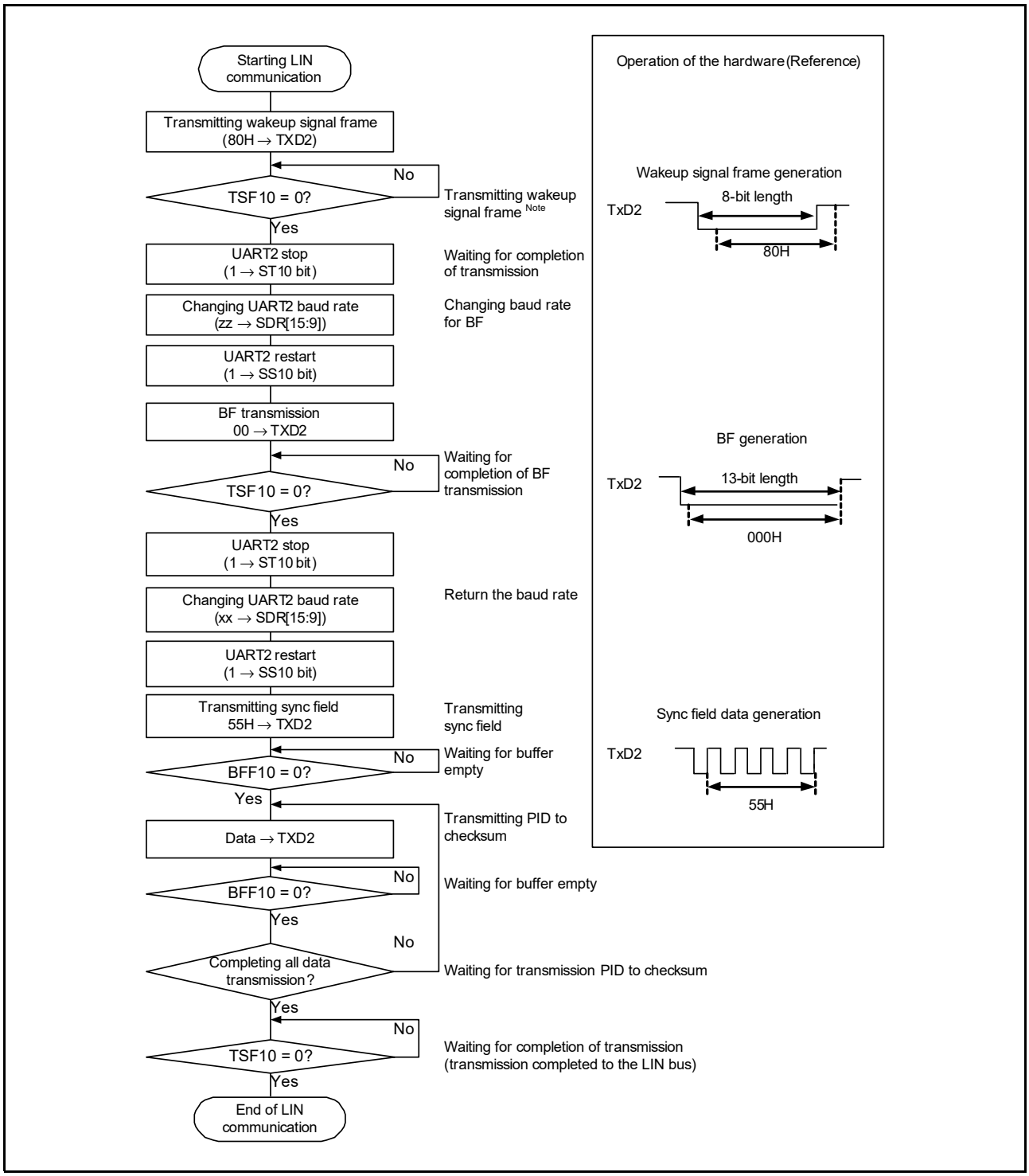
(Baud rate of break field) = 9/13 × N

By transmitting data of 00H at this baud rate, a break field is generated.

Note 3. INTST2 is output upon completion of transmission. INTST2 is also output at BF transmission.

Remark The interval between fields is controlled by software.

Figure 11 - 97 Flowchart for LIN Transmission



Note This is only required if the LIN bus is being started from sleep mode.

Remark This flow assumes that the initial setting of the UART is completed and transmission is enabled.

11.7.2 LIN reception

Of UART reception, UART2 supports LIN communication. For LIN reception, channel 1 of unit 1 is used.

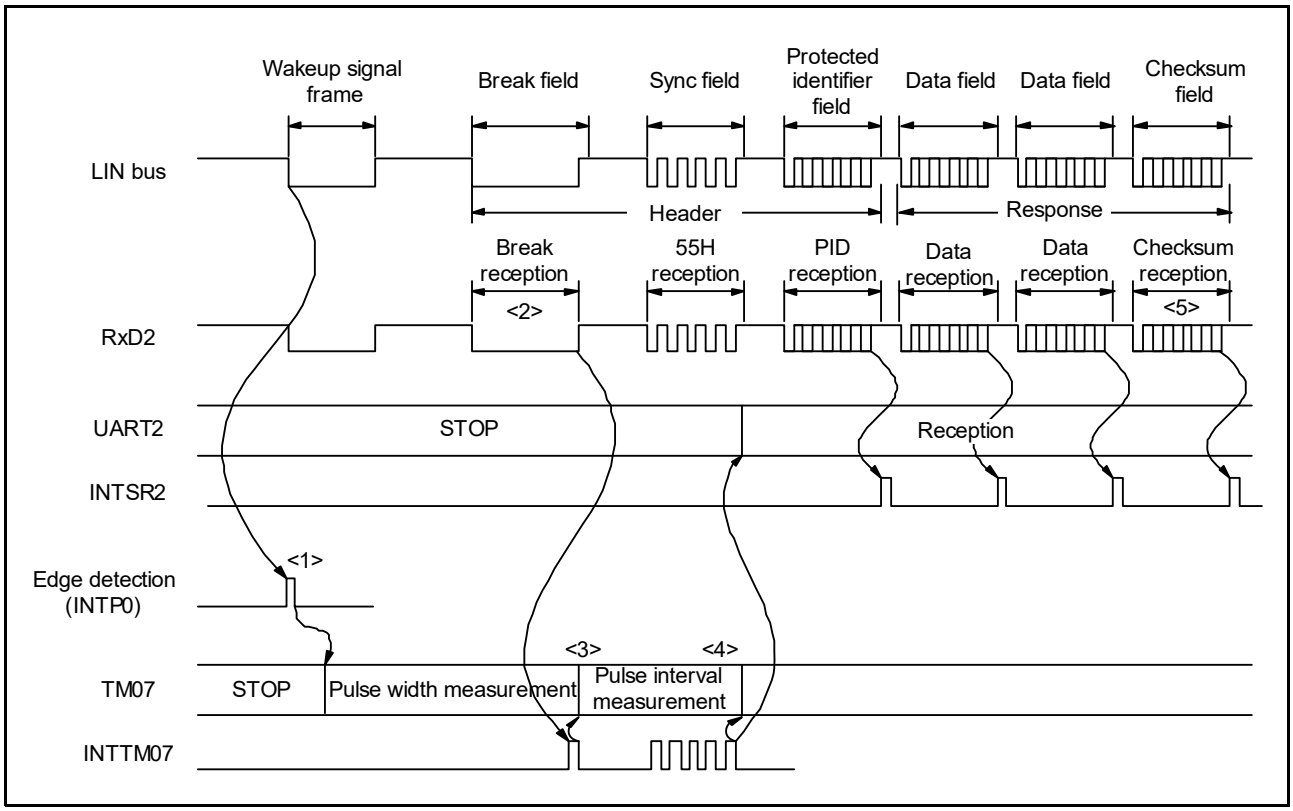
UART	UART0	UART1	UART2
Support of LIN communication	Not supported	Not supported	Supported
Target channel	—	—	Channel 1 of SAU1
Pins used	—	—	RxD2
Interrupt	—	—	INTSR2
	Transfer end interrupt only (Setting the buffer empty interrupt is prohibited.)		
Error interrupt	—	—	INTSRE2
Error detection flag	• Framing Error detection flag (FEF11) • Overrun Error detection flag (OVF11)		
Transfer data length	8 bits		
Transfer rate**Note**	Max. f_{MCK}/6 [bps] (SDR11[15:9] = 2 or more), Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [bps]		
Data phase	Non-reverse output (default: high level) Reverse output (default: low level)		
Parity bit	No parity bit (The parity bit is not checked.)		
Stop bit	Check the first bit		
Data direction	LSB first		

Note Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics specified in the electrical characteristics. For details, see **Section 28 Electrical Characteristics**.

Remark f_{MCK}: Operation clock frequency of target channel
f_{CLK}: CPU/peripheral hardware clock frequency

Figure 11 - 98 outlines a reception operation of LIN.

Figure 11 - 98 Reception Operation of LIN



Here is the flow of reception processing.

<1> The wakeup signal is detected by detecting an edge on the INTP0 interrupt pin. When the wakeup signal is detected, set TM07 to the pulse width measurement function to measure the low-level width of the BF signal. Then wait for BF signal reception.

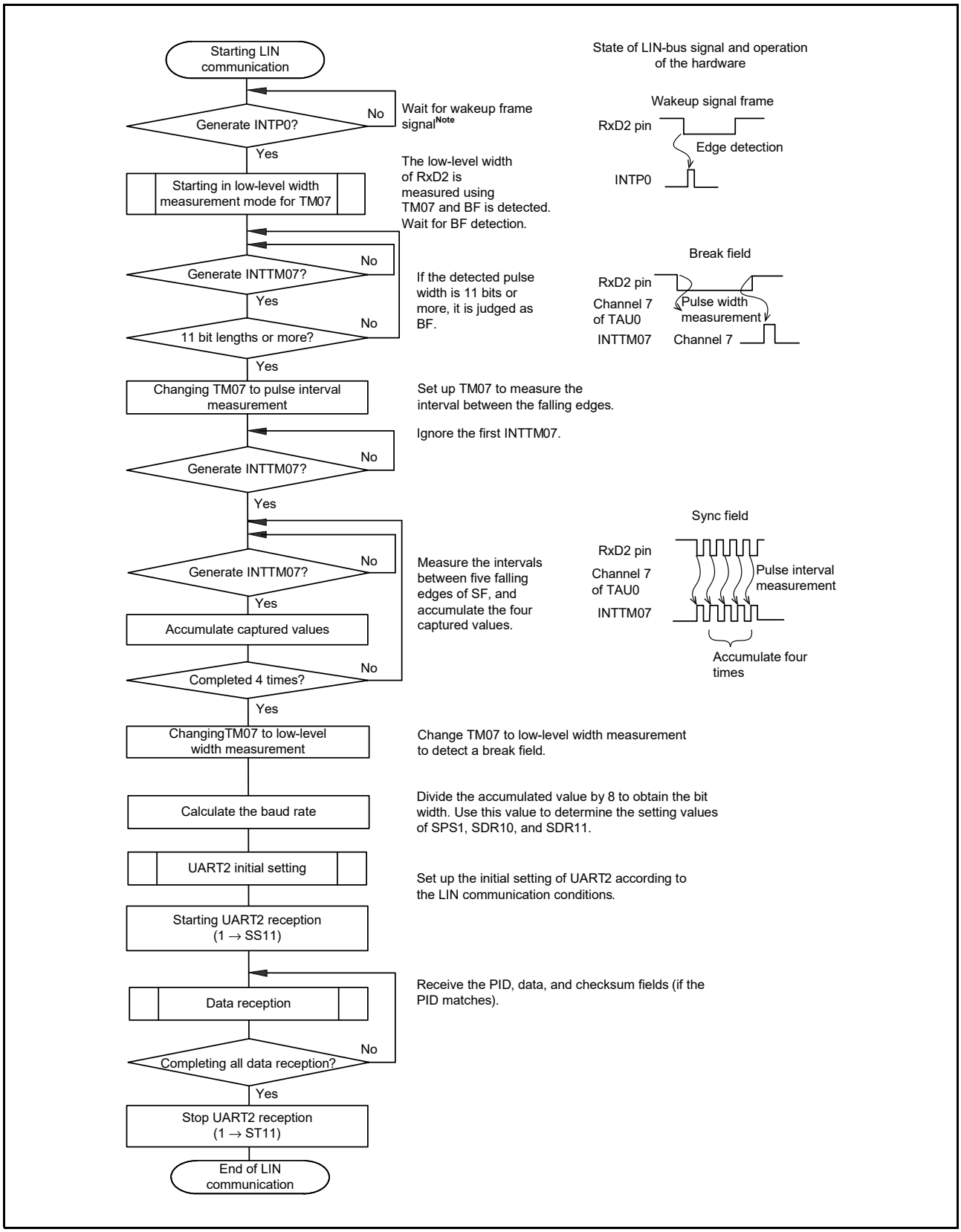
<2> TM07 starts measuring the low-level width upon detection of the falling edge of the BF signal, and then captures the data upon detection of the rising edge of the BF signal. The captured data is used to judge whether it is the BF signal.

<3> When the BF signal has been received normally, change TM07 to pulse interval measurement and measure the interval between the falling edges of the RxD2 signal in the Sync field four times. (see **7.8.4 Operation for input pulse interval measurement**.)

<4> Calculate a baud rate error from the bit interval of sync field (SF). Stop UART2 once and adjust (re-set) the baud rate.

<5> The checksum field should be distinguished by software. In addition, processing to initialize UART2 after the checksum field is received and to wait for reception of BF should also be performed by software.

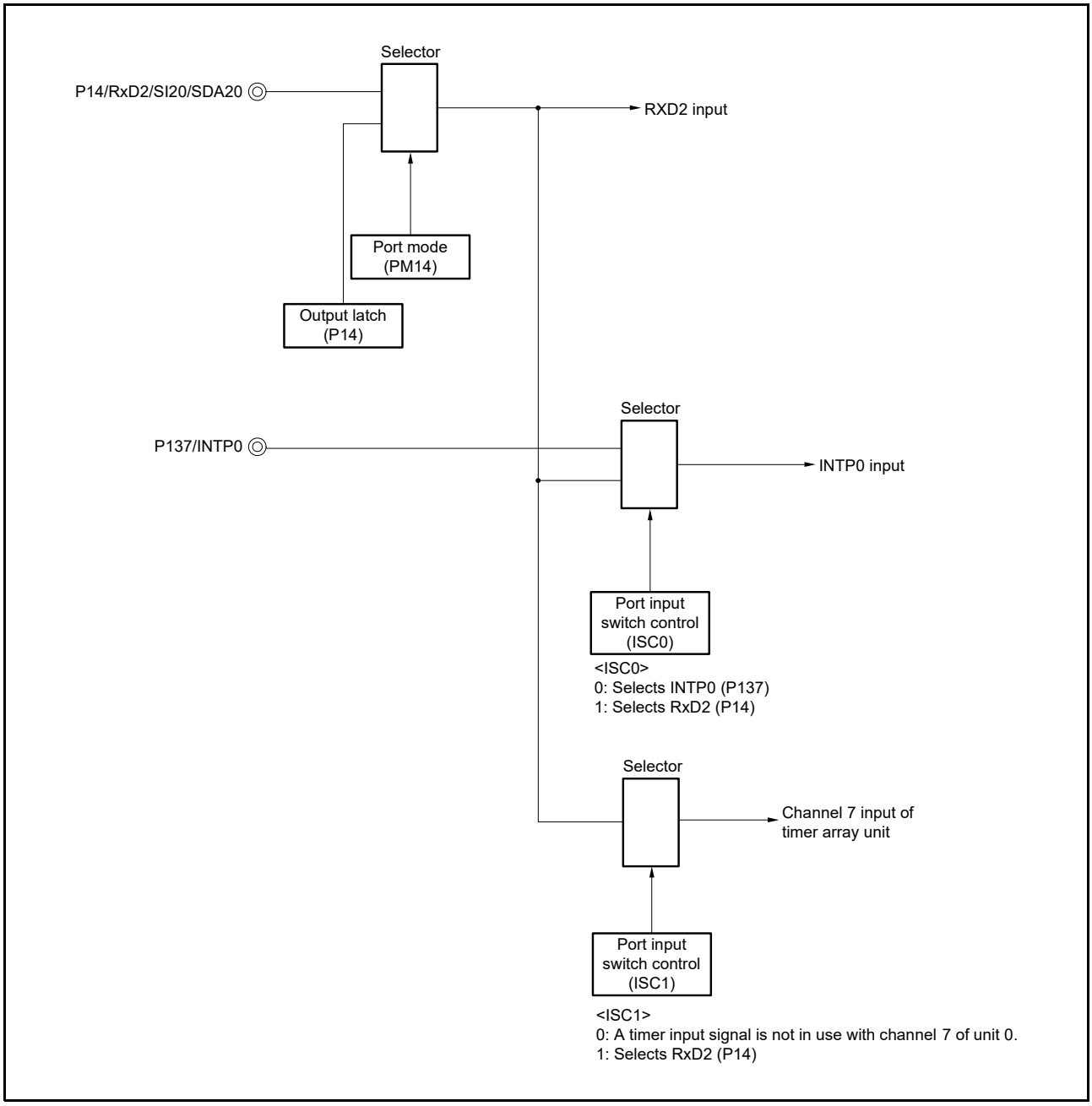
Figure 11 - 99 Flowchart of LIN Reception



Note This is only required if the LIN bus is in sleep mode.

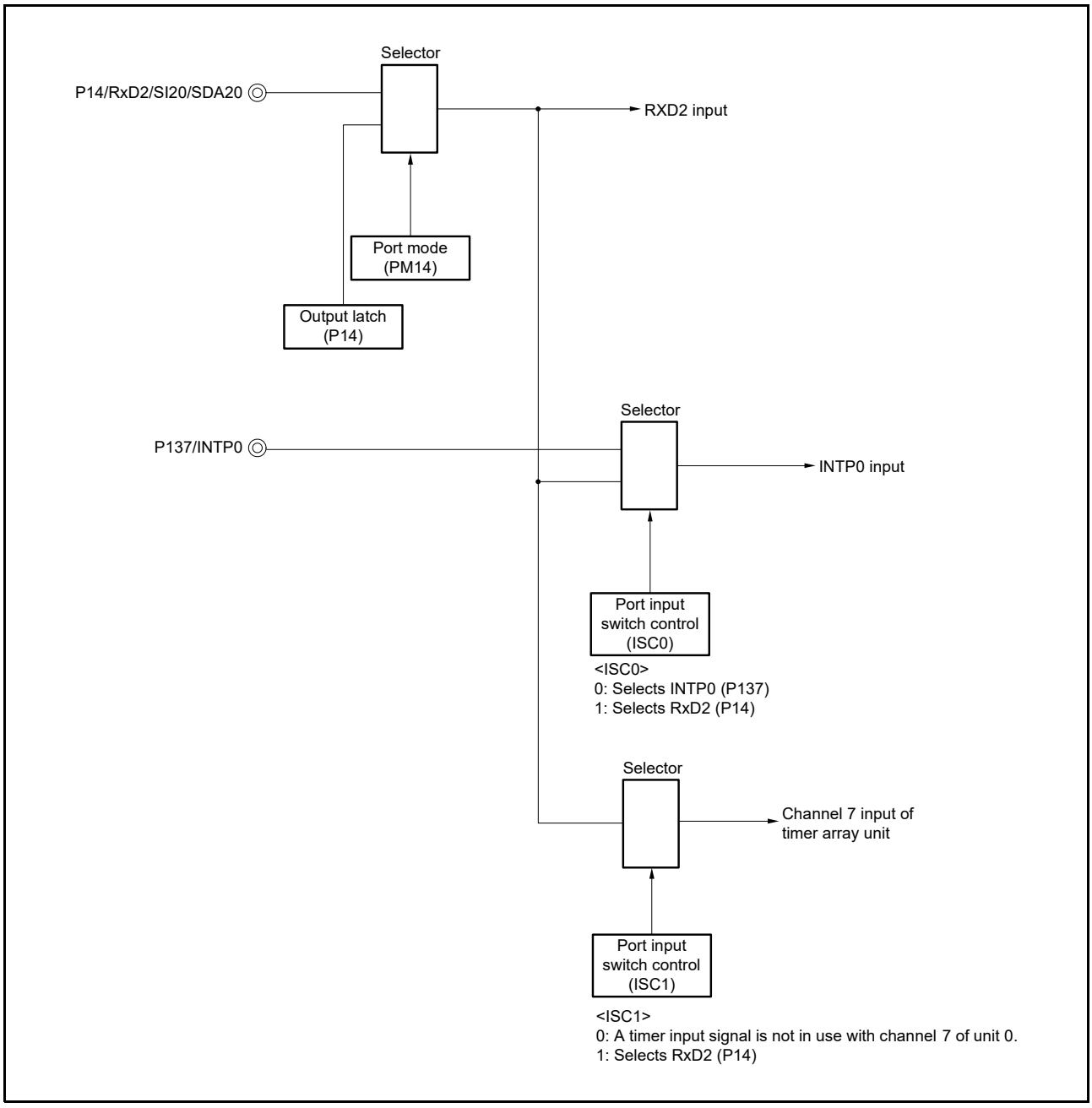
Figure 11 - 100 to **Figure 11 - 102** show the configuration of ports used for LIN reception. The wakeup signal transmitted from the master of LIN is received by detecting an edge of an external interrupt (INTP0). The length of the sync field transmitted from the master can be measured by using the external event capture operation of the timer array unit 0 to calculate a baud-rate error. By using the port input switching control (the ISC0 and ISC1 bits), the signal input to the reception port (RxD2) can be used as an external interrupt (INTP0) or sent to the timer array unit without additional external connections.

Figure 11 - 100 Port Configuration for Reception of LIN in 20-pin Products



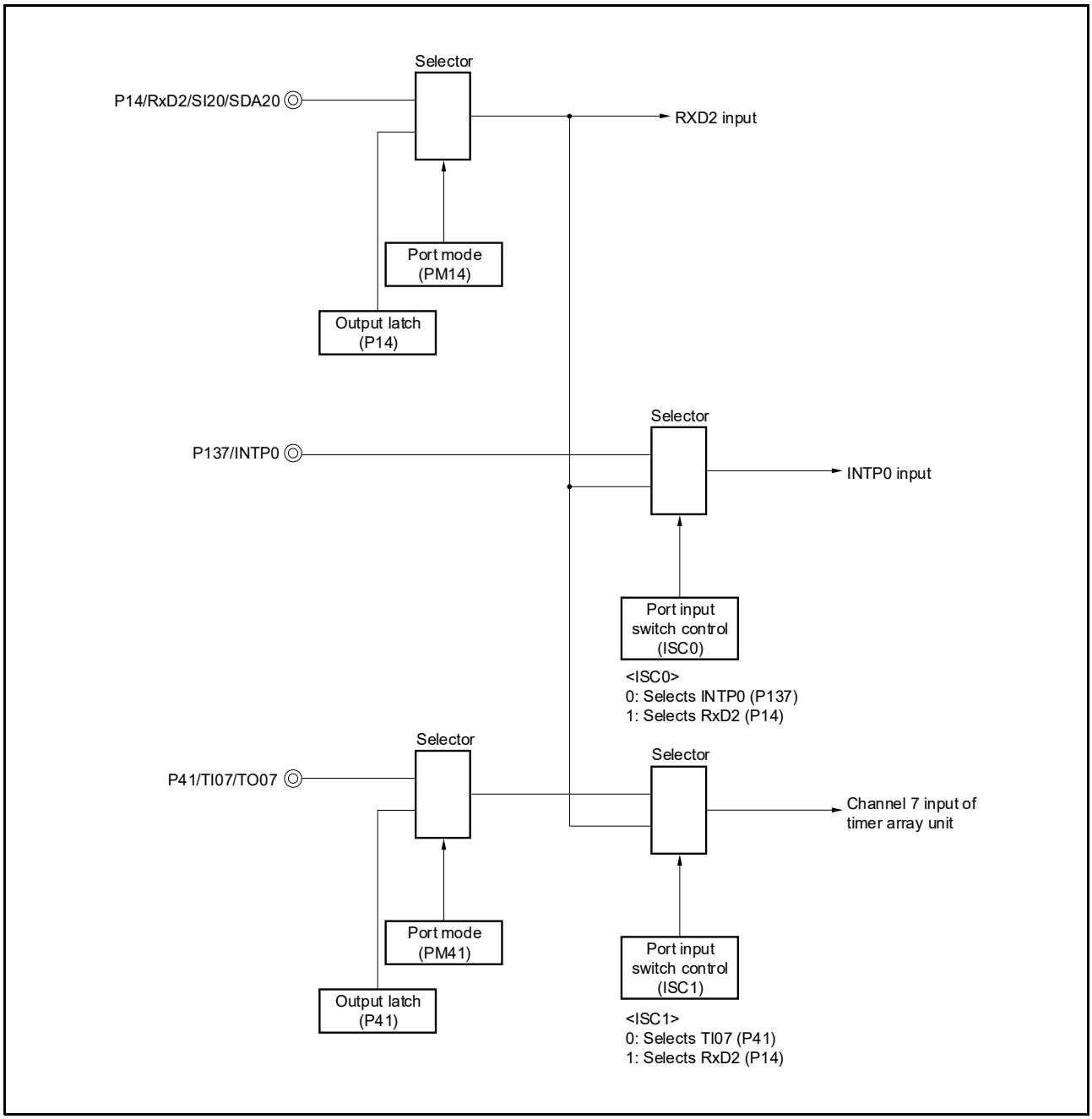
Remark ISC0, ISC1: Bits 0 and 1 of the input switch control register (ISC) (See **Figure 11 - 21**.)

Figure 11 - 101 Port Configuration for Reception of LIN in 32-pin Products



Remark ISC0, ISC1: Bits 0 and 1 of the input switch control register (ISC) (See **Figure 11 - 21**.)

Figure 11 - 102 Port Configuration for Reception of LIN in 44- to 64-pin Products



Remark ISC0, ISC1: Bits 0 and 1 of the input switch control register (ISC) (See **Figure 11 - 21**.)

The peripheral functions used for the LIN communication operation are as follows.

<Peripheral functions used>

- External interrupt (INTP0); Wakeup signal detection
  Usage: To detect an edge of the wakeup signal and the start of communication
- Channel 7 of timer array unit; Baud rate error detection, break field detection.
  Usage: To detect the length of the sync field (SF) and divide it by the number of bits in order to detect a baud rate error. (The interval of the edge input to RxD2 is measured in the capture mode.)
  To measure the low-level width to detect the break field (BF).
- Channels 0 and 1 (UART2) of serial array unit 1 (SAU1)

11.8 Operation of Simplified I2C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) Communication

This is a clocked communication function to communicate with two or more devices by using two lines: serial clock (SCL) and serial data (SDA). This simplified I2C is designed for single communication with a device such as EEPROM, flash memory, or A/D converter, and therefore, it functions only as a master. Operate the control registers by software for setting the start and stop conditions while observing the specifications of the I2C bus line.

[Data transmission/reception]

- Master transmission, master reception (only master function with a single master)
- ACK output function **Note** and ACK detection function
- Data length of 8 bits (When an address is transmitted, the address is specified by the higher 7 bits, and the least significant bit is used for R/W control.)
- Generation of start condition and stop condition for software

[Interrupt function]

- Transfer end interrupt

[Error detection flag]

- Overrun error
- ACK error

* [Functions not supported by simplified I2C]

- Slave transmission, slave reception
- Multi-master function (arbitration loss detection function)
- Clock stretch detection

Note When receiving the last data, ACK will not be output if 0 is written to the SOEmn (SOEm register) bit and serial communication data output is stopped. See **11.8.3 Data reception**, **2. Processing flow** for details.

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

The channels supporting simplified I2C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) are channels 0 to 3 of SAU0 and channels 0 and 1 of SAU1.

20-pin products

Unit	Channel	Used as Simplified SPI (CSI)	Used as UART	Used as Simplified I2C
0	0	CSI00	UART0	IIC00
	1	—		—
	2	—	UART1	—
	3	CSI11		IIC11
1	0	—	UART2 (supporting the LIN bus)	—
	1	—		—

32-pin products

Unit	Channel	Used as Simplified SPI (CSI)	Used as UART	Used as Simplified I2C
0	0	CSI00	UART0	IIC00
	1	—		—
	2	—	UART1	—
	3	CSI11		IIC11
1	0	CSI20	UART2 (supporting the LIN bus)	IIC20
	1	—		—

44-pin products

Unit	Channel	Used as Simplified SPI (CSI)	Used as UART	Used as Simplified I2C
0	0	CSI00	UART0	IIC00
	1	—		—
	2	—	UART1	—
	3	CSI11		IIC11
1	0	CSI20	UART2 (supporting the LIN bus)	IIC20
	1	CSI21		IIC21

48-pin products

Unit	Channel	Used as Simplified SPI (CSI)	Used as UART	Used as Simplified I2C
0	0	CSI00	UART0	IIC00
	1	CSI01		IIC01
	2	—	UART1	—
	3	CSI11		IIC11
1	0	CSI20	UART2 (supporting the LIN bus)	IIC20
	1	CSI21		IIC21

64-pin products

Unit	Channel	Used as Simplified SPI (CSI)	Used as UART	Used as Simplified I²C
0	0	CSI00	UART0	IIC00
	1	CSI01		IIC01
	2	CSI10	UART1	IIC10
	3	CSI11		IIC11
1	0	CSI20	UART2 (supporting the LIN bus)	IIC20
	1	CSI21		IIC21

Simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) performs the following four types of communication operations.

- Address field transmission (See **11.8.1**.)
- Data transmission (See **11.8.2**.)
- Data reception (See **11.8.3**.)
- Stop condition generation (See **11.8.4**.)

11.8.1 Address field transmission

Address field transmission is a transmission operation that first executes in I^2C communication to identify the target for transfer (slave). After a start condition is generated, an address (7 bits) and a transfer direction (1 bit) are transmitted in one frame.

Simplified I^2C	IIC00	IIC01	IIC10	IIC11	IIC20	IIC21
Target channel	Channel 0 of SAU0	Channel 1 of SAU0	Channel 2 of SAU0	Channel 3 of SAU0	Channel 0 of SAU1	Channel 1 of SAU1
Pins used	SCL00, SDA00**Note 1**	SCL01, SDA01**Note 1**	SCL10, SDA10**Note 1**	SCL11, SDA11**Note 1**	SCL20, SDA20**Note 1**	SCL21, SDA21**Note 1**
Interrupt	INTIIC00	INTIIC01	INTIIC10	INTIIC11	INTIIC20	INTIIC21
	Transfer end interrupt only (Setting the buffer empty interrupt is prohibited.)					
Error detection flag	ACK error detection flag (PEFmn)					
Transfer data length	8 bits (transmitted with specifying the higher 7 bits as address and the least significant bit as R/W control)					
Transfer rate**Note 2**	Max. $f_{MCK}/4$ [Hz] (SDRmn[15:9] = 1 or more) f_{MCK}: Operation clock frequency of target channel However, the following condition must be satisfied in each mode of I^2C. • Max. 1 MHz (fast mode plus) • Max. 400 kHz (fast mode) • Max. 100 kHz (standard mode)					
Data level	Non-reverse output (default: high level)					
Parity bit	No parity bit					
Stop bit	Appending 1 bit (for ACK transmission/reception timing)					
Data direction	MSB first					

Note 1. To perform communication via simplified I^2C, set the N-ch open-drain output [withstand voltage of V_{DD} (20- to 48-pin products)/withstand voltage of EV_{DD} (64-pin products)] mode (POMxx = 1) with the port output mode register (POMxx). See **4.3 Registers for Controlling the Port Function** and **4.5 Register Settings When Using Alternate Function** for details.
To communicate with an external device operating at a different voltage through IIC00, IIC10, and IIC20, set the N-ch open-drain output [withstand voltage of V_{DD} (20- to 48-pin products)/withstand voltage of EV_{DD} (64-pin products)] mode (POMxx = 1) also for the clock input/output pins (SCL00, SCL10, SCL20). See **4.4.5 Communications with devices operating at a different voltage (1.8 V, 2.5 V, or 3 V) by switching I/O buffers** for details.

Note 2. Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics specified in the electrical characteristics. For details, see **Section 28 Electrical Characteristics**.

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

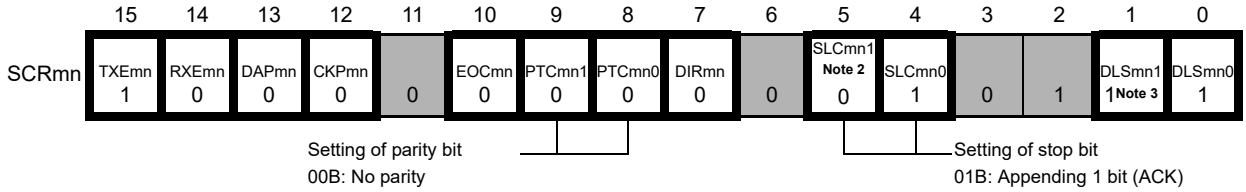
1. Register setting

Figure 11 - 103 Example of Contents of Registers for Address Field Transmission of Simplified I2C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21)

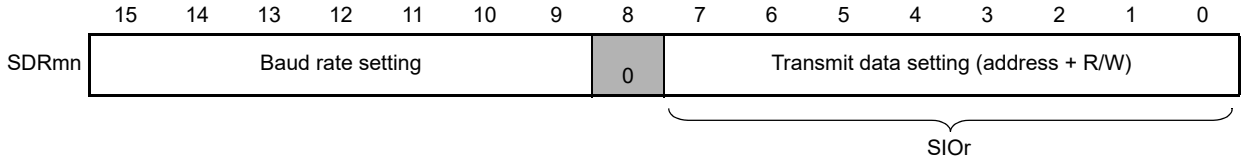
a) Serial mode register mn (SMRmn)



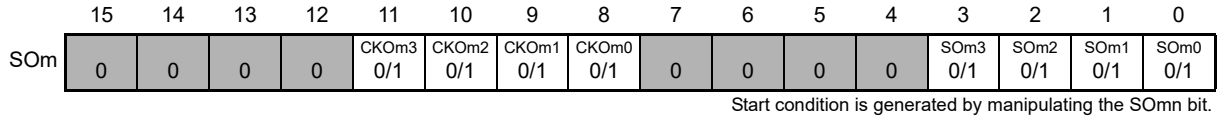
b) Serial communication operation setting register mn (SCRmn)



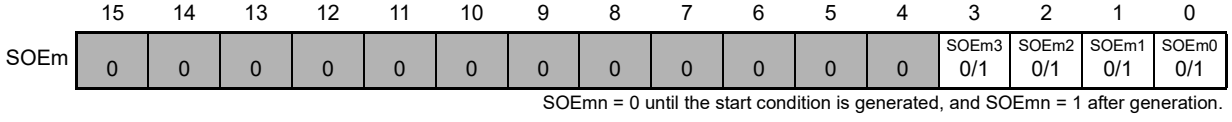
c) Serial data register mn (SDRmn) (lower 8 bits: SIOr)



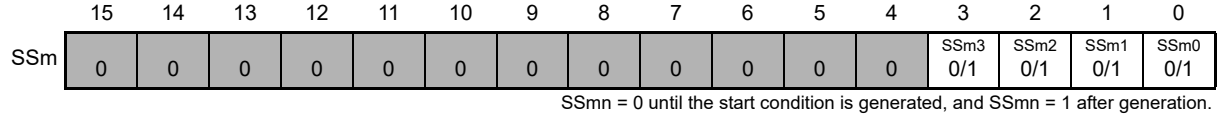
d) Serial output register m (SOm)



e) Serial output enable register m (SOEm)



f) Serial channel start register m (SSm): Set only the bit of the target channel to 1.



Note 1. This bit is only present in the SMR00, SMR03, and SMR11 registers.
Note 2. This bit is only present in the SCR00, SCR02, and SCR10 registers.
Note 3. This bit is only present in the SCR00 and SCR01 registers, and is fixed to 1 for the other registers.

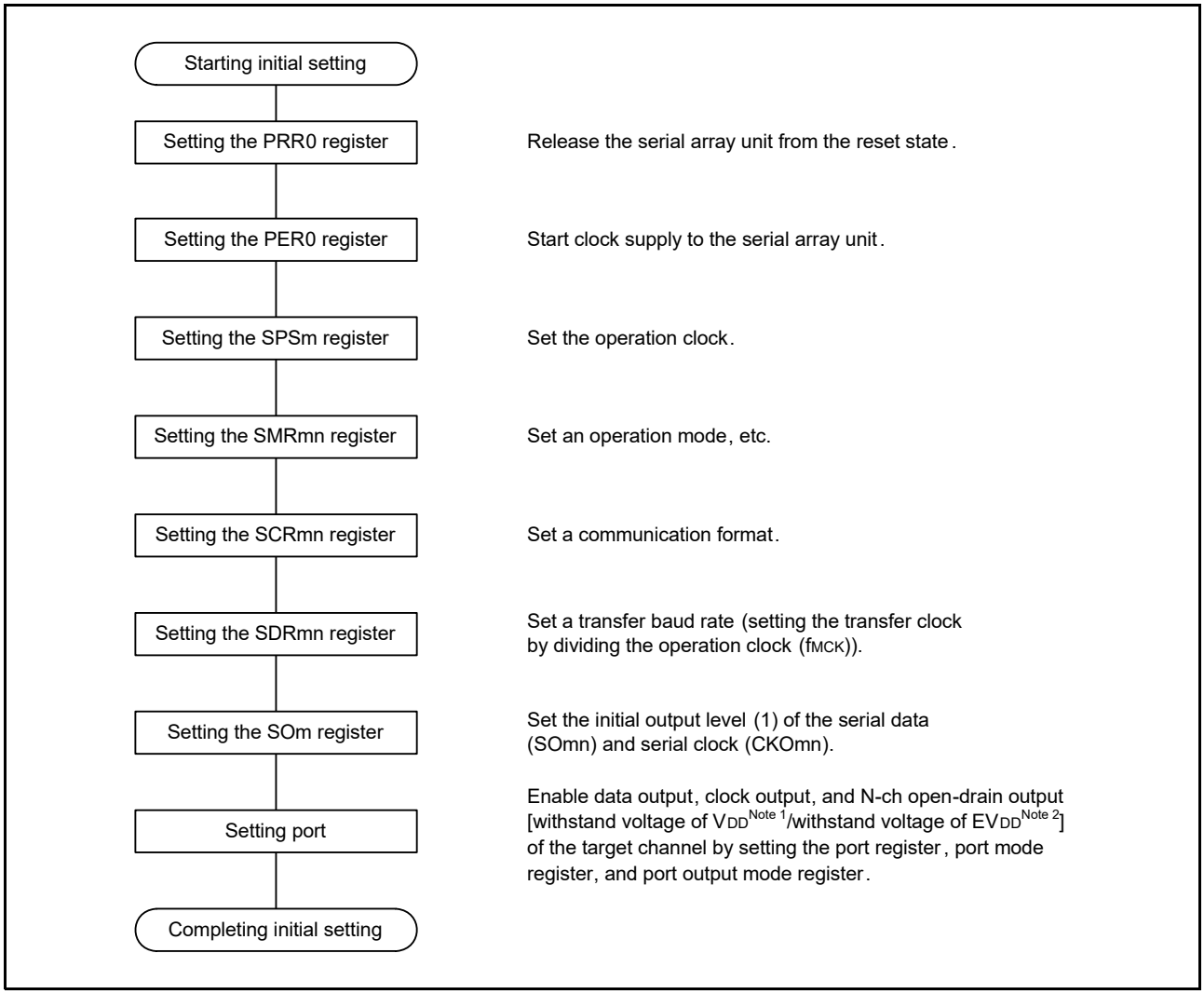
(Remarks are listed on the next page.)

Remark 1. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), r: IIC number (r = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

Remark 2. ■: Setting is fixed in the IIC mode, ■: Setting disabled (set to the initial value)
0/1: Set to 0 or 1 depending on the user's application.

2. Operation procedure

Figure 11 - 104 Initial Setting Procedure for Simplified I2C Address Field Transmission

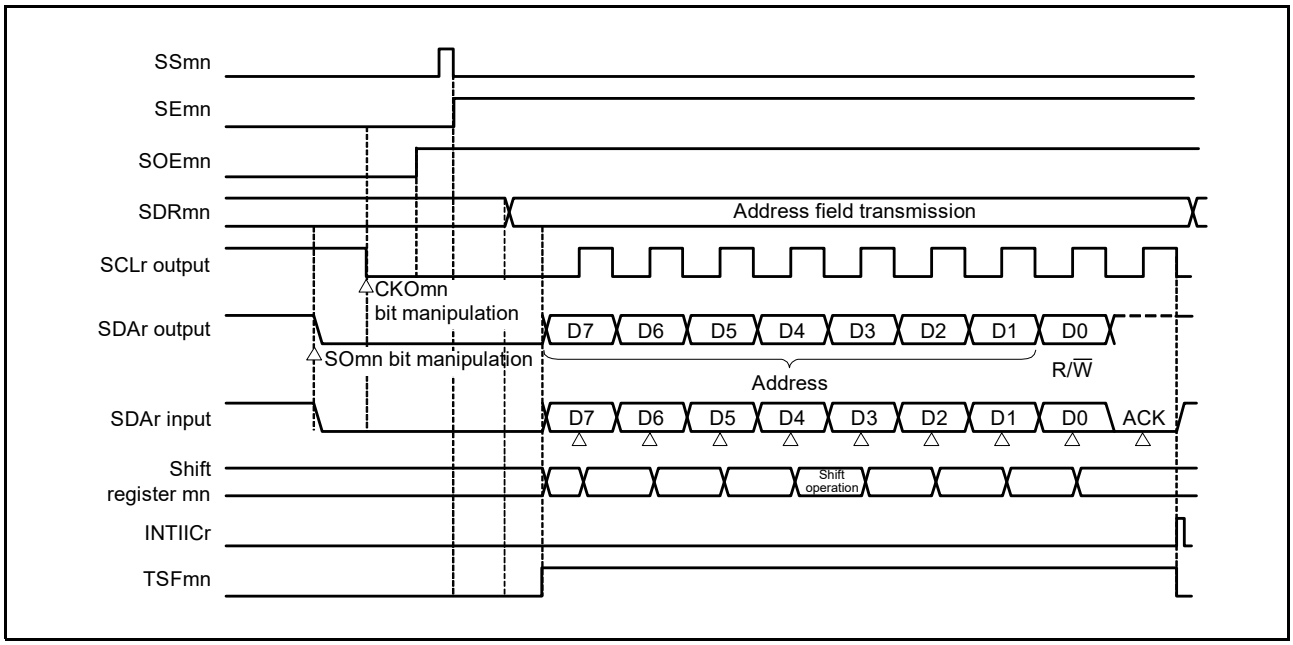


Note 1. 20- to 48-pin products

Note 2. 64-pin products

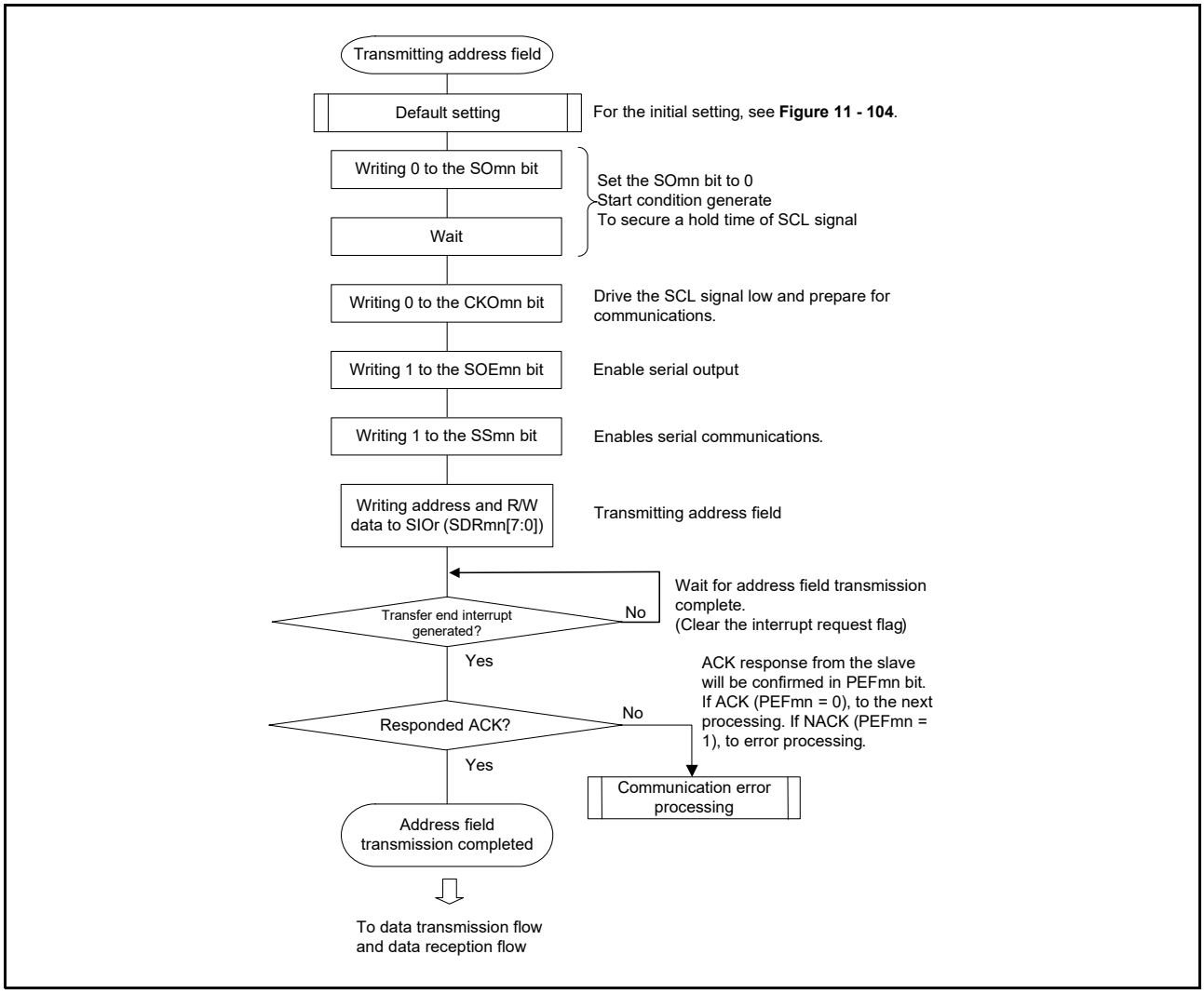
3. Processing flow

Figure 11 - 105 Timing Chart of Address Field Transmission



Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), r: IIC number (r = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

Figure 11 - 106 Flowchart of Simplified I2C Address Field Transmission



11.8.2 Data transmission

Data transmission is an operation to transmit data to the target for transfer (slave) after transmission of an address field. After all data are transmitted to the slave, a stop condition is generated and the bus is released.

Simplified I²C	IIC00	IIC01	IIC10	IIC11	IIC20	IIC21
Target channel	Channel 0 of SAU0	Channel 1 of SAU0	Channel 2 of SAU0	Channel 3 of SAU0	Channel 0 of SAU1	Channel 1 of SAU1
Pins used	SCL00, SDA00Note 1	SCL01, SDA01Note 1	SCL10, SDA10Note 1	SCL11, SDA11Note 1	SCL20, SDA20Note 1	SCL21, SDA21Note 1
Interrupt	INTIIC00	INTIIC01	INTIIC10	INTIIC11	INTIIC20	INTIIC21
	Transfer end interrupt only (Setting the buffer empty interrupt is prohibited.)					
Error detection flag	ACK error flag (PEFmn)					
Transfer data length	8 bits					
Transfer rateNote 2	Max. fMCK/4 [Hz] (SDRmn[15:9] = 1 or more) fMCK: Operation clock frequency of target channel However, the following condition must be satisfied in each mode of I²C. • Max. 1 MHz (fast mode plus) • Max. 400 kHz (fast mode) • Max. 100 kHz (standard mode)					
Data level	Non-reverse output (default: high level)					
Parity bit	No parity bit					
Stop bit	Appending 1 bit (for ACK reception timing)					
Data direction	MSB first					

Note 1. To perform communication via simplified I²C, set the N-ch open-drain output [withstand voltage of VDD (20- to 48-pin products)/withstand voltage of EVDD (64-pin products)] mode (POMxx = 1) with the port output mode register (POMxx). For details, see **4.3 Registers for Controlling the Port Function** and **4.5 Register Settings When Using Alternate Function**. To communicate with an external device operating at a different voltage through IIC00, IIC10, and IIC20, set the N-ch open-drain output [withstand voltage of VDD (20- to 48-pin products)/withstand voltage of EVDD (64- to 128-pin products)] mode (POMxx = 1) also for the clock input/output pins (SCL00, SCL10, SCL20, SCL30, SCL31). For details, see **4.4.5 Communications with devices operating at a different voltage (1.8 V, 2.5 V, or 3 V) by switching I/O buffers**.

Note 2. Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics specified in the electrical characteristics. For details, see **Section 28 Electrical Characteristics**.

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

1. Register setting

Figure 11 - 107 Example of Contents of Registers for Data Transmission of Simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21)

a) Serial mode register mn (SMRmn): Do not manipulate this register during data transmission/reception.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn 0/1	CCSmn 0	0	0	0	0	0	STSmn 0Note 1	0	SISmn0 0Note 1	1	0	0	MDmn2 1	MDmn1 0	MDmn0 0

b) Serial communication operation setting register mn (SCRmn): Do not manipulate the bits of this register, except the TXEmn and RXEmn bits, during data transmission/reception.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn 1	RXEmn 0	DAPmn 0	CKPmn 0	0	EOCmn 0	PTCmn1 0	PTCmn0 0	DIRmn 0	0	SLCmn1 0Note 2	SLCmn0 1	0	1	DLSmn1 1Note 3	DLSmn0 1

c) Serial data register mn (SDRmn) (lower 8 bits: SIOr): During data transmission/reception, valid only lower 8-bits (SIOr)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	Baud rate settingNote 4							0	Transmit data setting							

SIOr

d) Serial output register m (SOm): Do not manipulate this register during data transmission/reception.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOm	0	0	0	0	CKOm3 0/1 Note 5	CKOm2 0/1 Note 5	CKOm1 0/1 Note 5	CKOm0 0/1 Note 5	0	0	0	0	SOm3 0/1 Note 5	SOm2 0/1 Note 5	SOm1 0/1 Note 5	SOm0 0/1 Note 5

e) Serial output enable register m (SOEm): Do not manipulate this register during data transmission/reception.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm	0	0	0	0	0	0	0	0	0	0	0	0	SOEm3 1	SOEm2 1	SOEm1 1	SOEm0 1

f) Serial channel start register m (SSm): Do not manipulate this register during data transmission/reception.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	SSm3 0/1	SSm2 0/1	SSm1 0/1	SSm0 0/1

Note 1. This bit is only present in the SMR01, SMR03, and SMR11 registers.
Note 2. This bit is only present in the SCR00, SCR02, and SCR10 registers.
Note 3. This bit is only present in the SCR00 and SCR01 registers, and is fixed to 1 for the other registers.
Note 4. Because the setting is completed by address field transmission, setting is not required.
Note 5. The value varies depending on the communication data during communication operation.

Remark 1. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), r: IIC number (r = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

Remark 2. ■: Setting is fixed in the IIC mode, ■: Setting disabled (set to the initial value)
0/1: Set to 0 or 1 depending on the user's application.

2. Processing flow

Figure 11 - 108 Timing Chart of Data Transmission

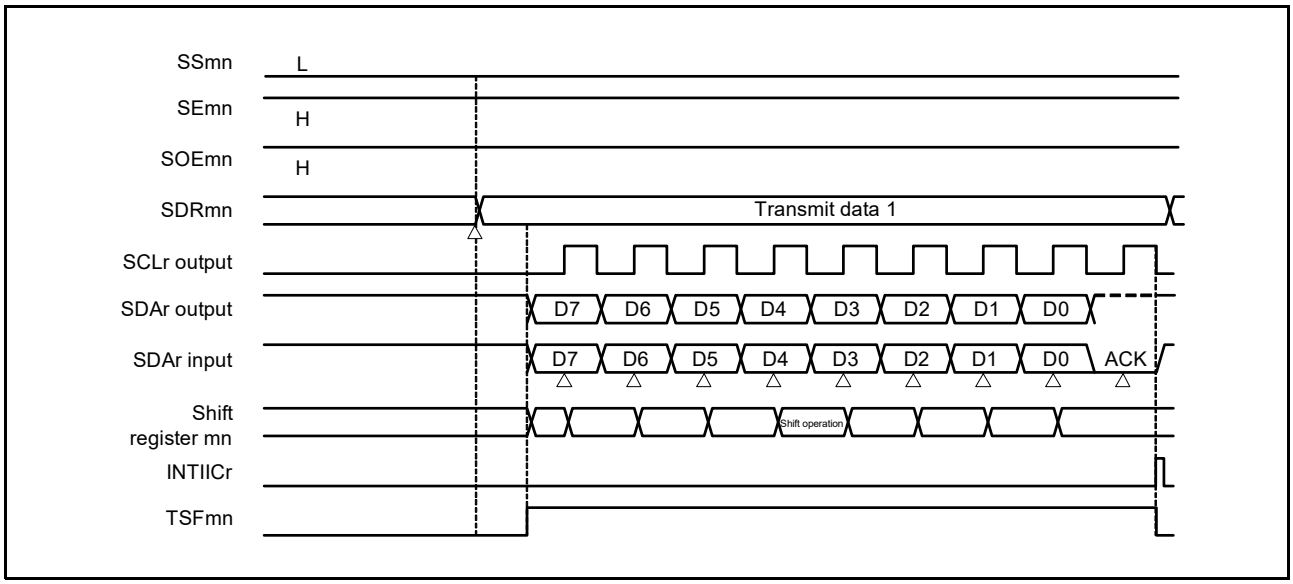
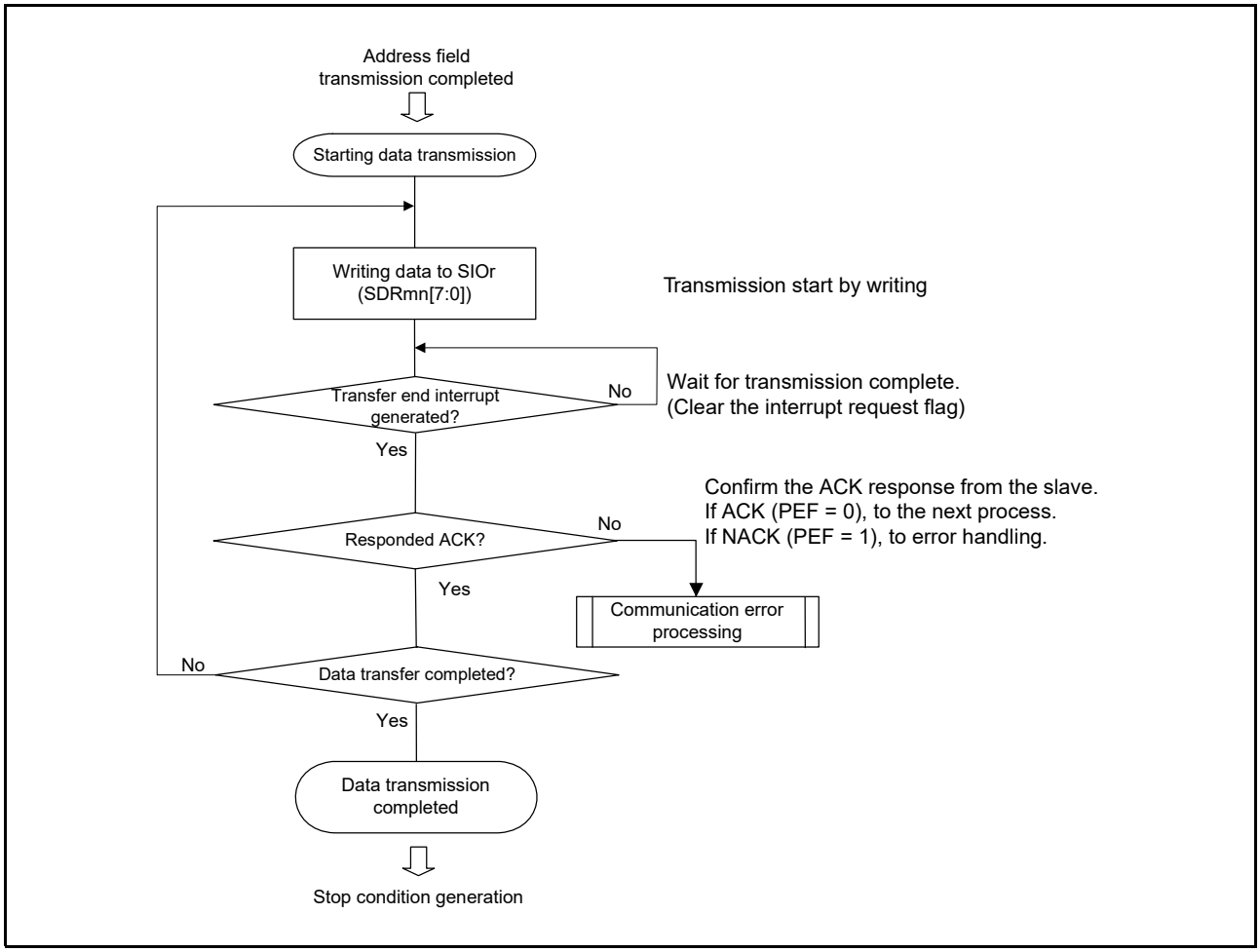


Figure 11 - 109 Flowchart of Simplified I2C Data Transmission



11.8.3 Data reception

Data reception is an operation to receive data from the target for transfer (slave) after transmission of an address field. After all data are received from the slave, a stop condition is generated and the bus is released.

Simplified I²C	IIC00	IIC01	IIC10	IIC11	IIC20	IIC21
Target channel	Channel 0 of SAU0	Channel 1 of SAU0	Channel 2 of SAU0	Channel 3 of SAU0	Channel 0 of SAU1	Channel 1 of SAU1
Pins used	SCL00, SDA00Note 1	SCL01, SDA01Note 1	SCL10, SDA10Note 1	SCL11, SDA11Note 1	SCL20, SDA20Note 1	SCL21, SDA21Note 1
Interrupt	INTIIC00	INTIIC01	INTIIC10	INTIIC11	INTIIC20	INTIIC21
	Transfer end interrupt only (Setting the buffer empty interrupt is prohibited.)					
Error detection flag	Overrun error detection flag (OVFmn) only					
Transfer data length	8 bits					
Transfer rateNote 2	Max. fMCK/4 [Hz] (SDRmn[15:9] = 1 or more) fMCK: Operation clock frequency of target channel However, the following condition must be satisfied in each mode of I²C. • Max. 1 MHz (fast mode plus) • Max. 400 kHz (fast mode) • Max. 100 kHz (standard mode)					
Data level	Non-reverse output (default: high level)					
Parity bit	No parity bit					
Stop bit	Appending 1 bit (ACK transmission)					
Data direction	MSB first					

Note 1. To perform communication via simplified I²C, set the N-ch open-drain output [withstand voltage of VDD (20- to 48-pin products)/withstand voltage of EVDD (64-pin products)] mode (POMxx = 1) with the port output mode register (POMxx). For details, see **4.3 Registers for Controlling the Port Function** and **4.5 Register Settings When Using Alternate Function**. To communicate with an external device operating at a different voltage through IIC00, IIC10, and IIC20, set the N-ch open-drain output [withstand voltage of VDD (20- to 48-pin products)/withstand voltage of EVDD (64-pin products)] mode (POMxx = 1) also for the clock input/output pins (SCL00, SCL10, SCL20, SCL30, SCL31). For details, see **4.4.5 Communications with devices operating at a different voltage (1.8 V, 2.5 V, or 3 V) by switching I/O buffers**.

Note 2. Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics specified in the electrical characteristics. For details, see **Section 28 Electrical Characteristics**.

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

1. Register setting

Figure 11 - 110 Example of Contents of Registers for Data Reception of Simplified I2C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21)

a) Serial mode register mn (SMRmn): Do not manipulate this register during data transmission/reception.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn 0/1	CCSmn 0	0	0	0	0	0	STSmn 0Note 1	0	SISmn0 0Note 1	1	0	0	MDmn2 1	MDmn1 0	MDmn0 0

b) Serial communication operation setting register mn (SCRmn): Do not manipulate the bits of this register, except the TXEmn and RXEmn bits, during data transmission/reception.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn 0	RXEmn 1	DAPmn 0	CKPmn 0	0	EOCmn 0	PTCmn1 0	PTCmn0 0	DIRmn 0	0	SLCmn1 0Note 2	SLCmn0 1	0	1	DLSmn1 1Note 3	DLSmn0 1

c) Serial data register mn (SDRmn) (lower 8 bits: SIOr): The lower 8 bits (SIOr) are only valid during data transmission/reception.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	Baud rate settingNote 4							0	Dummy transmit data setting (FFH)							
									SIOr							

d) Serial output register m (SOm): Do not manipulate this register during data transmission/reception.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOm	0	0	0	0	CKOm3 0/1 Note 5	CKOm2 0/1 Note 5	CKOm1 0/1 Note 5	CKOm0 0/1 Note 5	0	0	0	0	SOm3 0/1 Note 5	SOm2 0/1 Note 5	SOm1 0/1 Note 5	SOm0 0/1 Note 5

e) Serial output enable register m (SOEm): Do not manipulate this register during data transmission/reception.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm	0	0	0	0	0	0	0	0	0	0	0	0	SOEm3 0/1	SOEm2 0/1	SOEm1 0/1	SOEm0 0/1

f) Serial channel start register m (SSm): Do not manipulate this register during data transmission/reception.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	SSm3 0/1	SSm2 0/1	SSm1 0/1	SSm0 0/1

Note 1. This bit is only present in the SMR01, SMR03, and SMR11 registers.

Note 2. This bit is only present in the SCR00, SCR02, and SCR10 registers.

Note 3. This bit is only present in the SCR00 and SCR01 registers, and is fixed to 1 for the other registers.

Note 4. Because the setting is completed by address field transmission, setting is not required.

Note 5. The value varies depending on the communication data during communication operation.

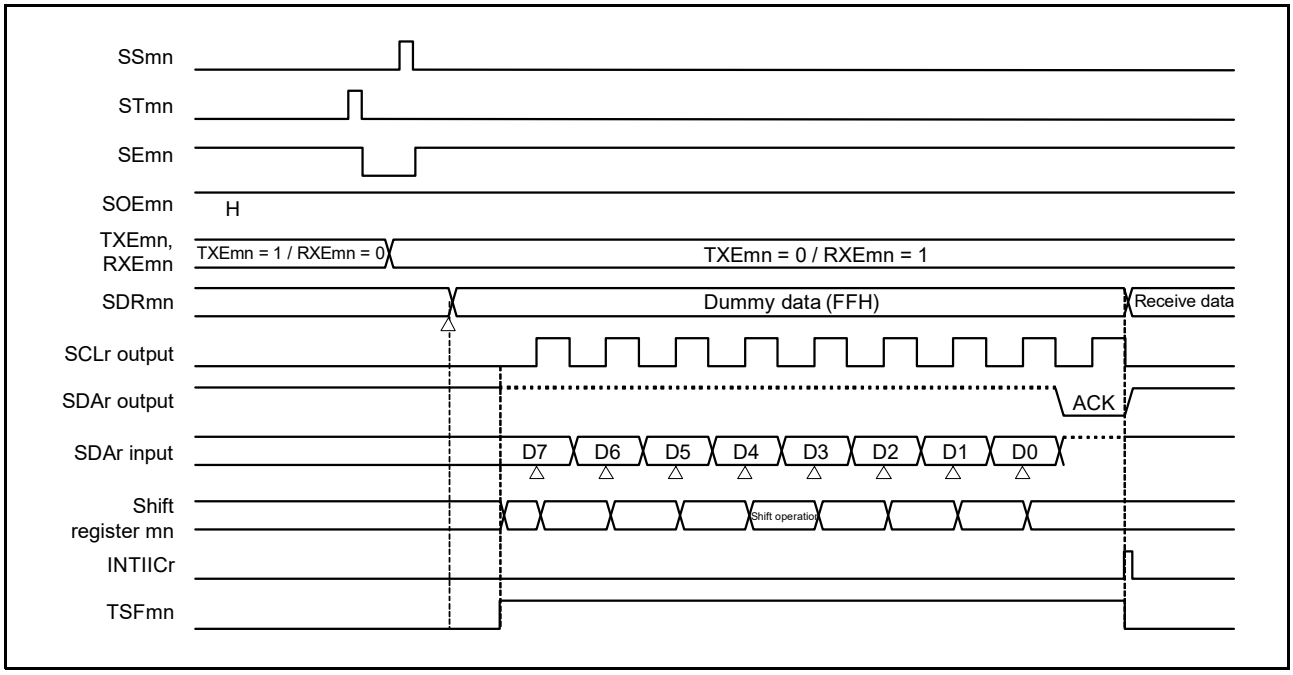
Remark 1. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), r: IIC number (r = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

Remark 2. ■: Setting is fixed in the IIC mode, ■: Setting disabled (set to the initial value)
0/1: Set to 0 or 1 depending on the user's application.

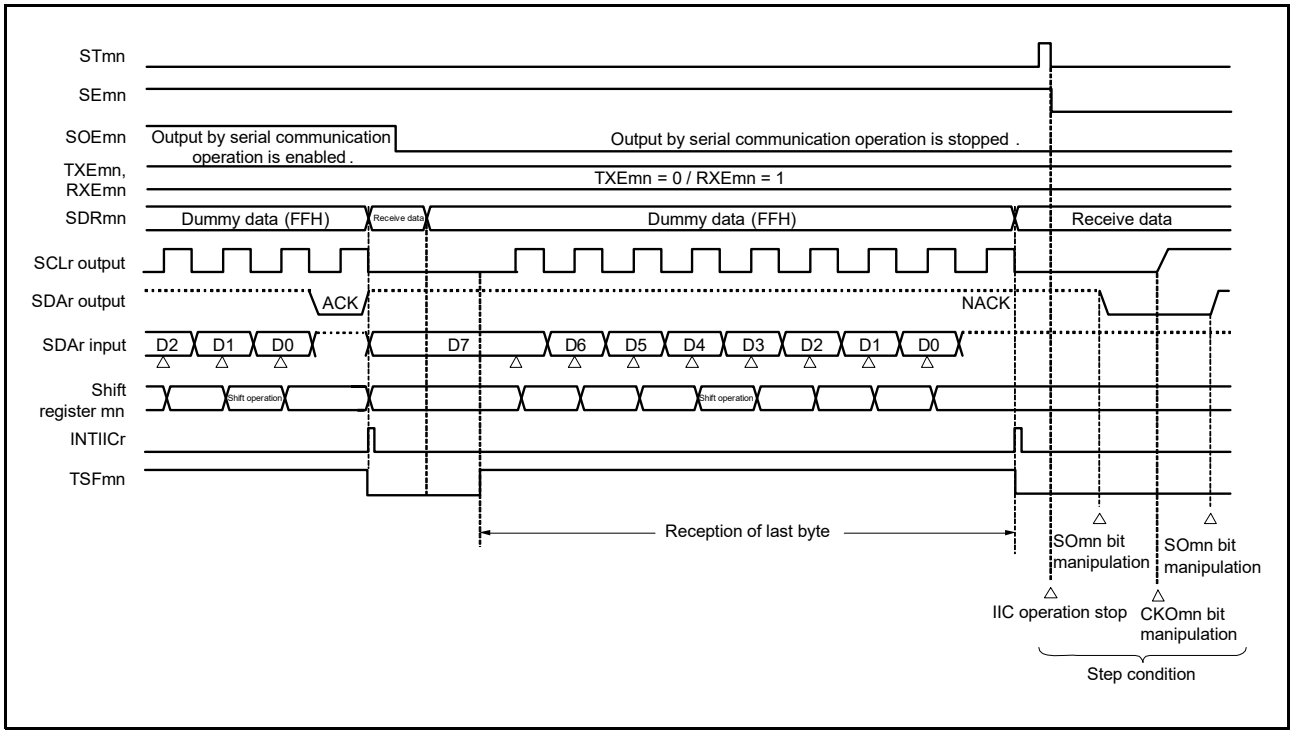
2. Processing flow

Figure 11 - 111 Timing Chart of Data Reception

a) When starting data reception

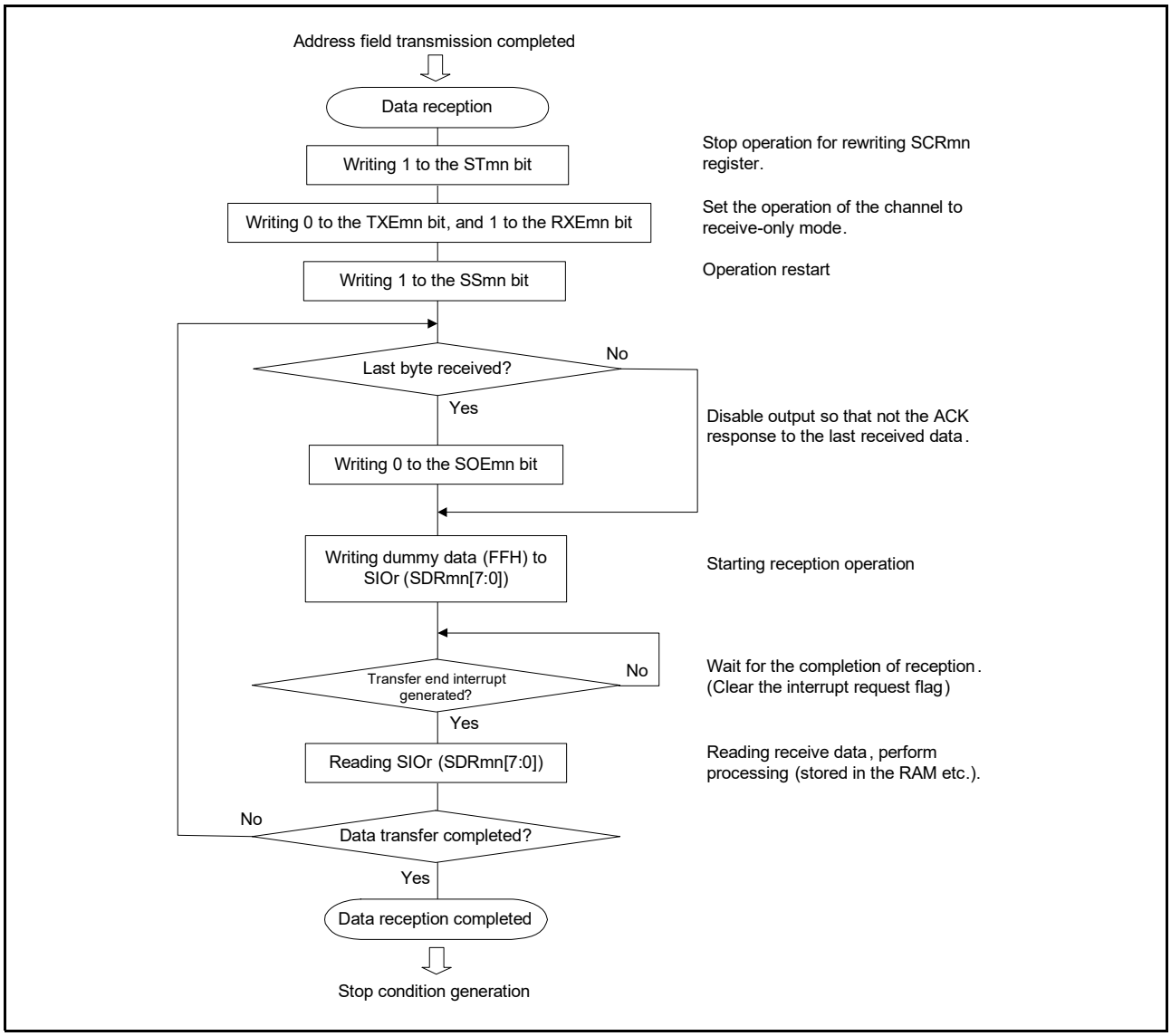


b) When receiving last data



Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), r: IIC number (r = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

Figure 11 - 112 Flowchart of Data Reception



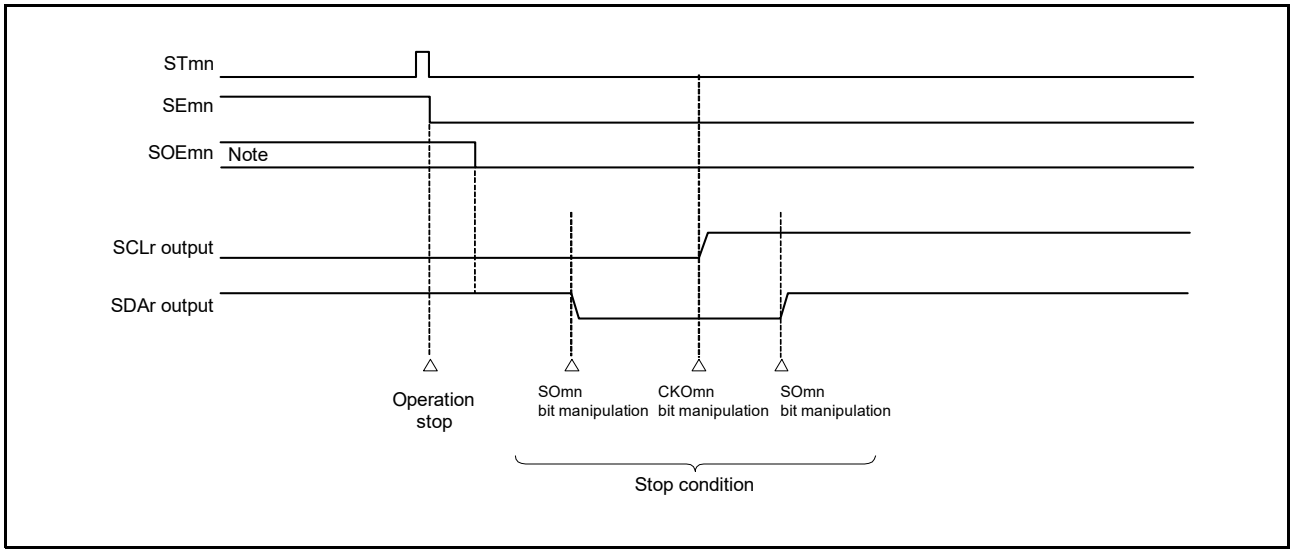
Caution **ACK is not output when the last data is received (NACK). Communication is then completed by setting 1 in the STmn bit of serial channel stop register m (STm) to stop operation and generating a stop condition.**

11.8.4 Stop condition generation

After all data are transmitted to or received from the target slave, a stop condition is generated and the bus is released.

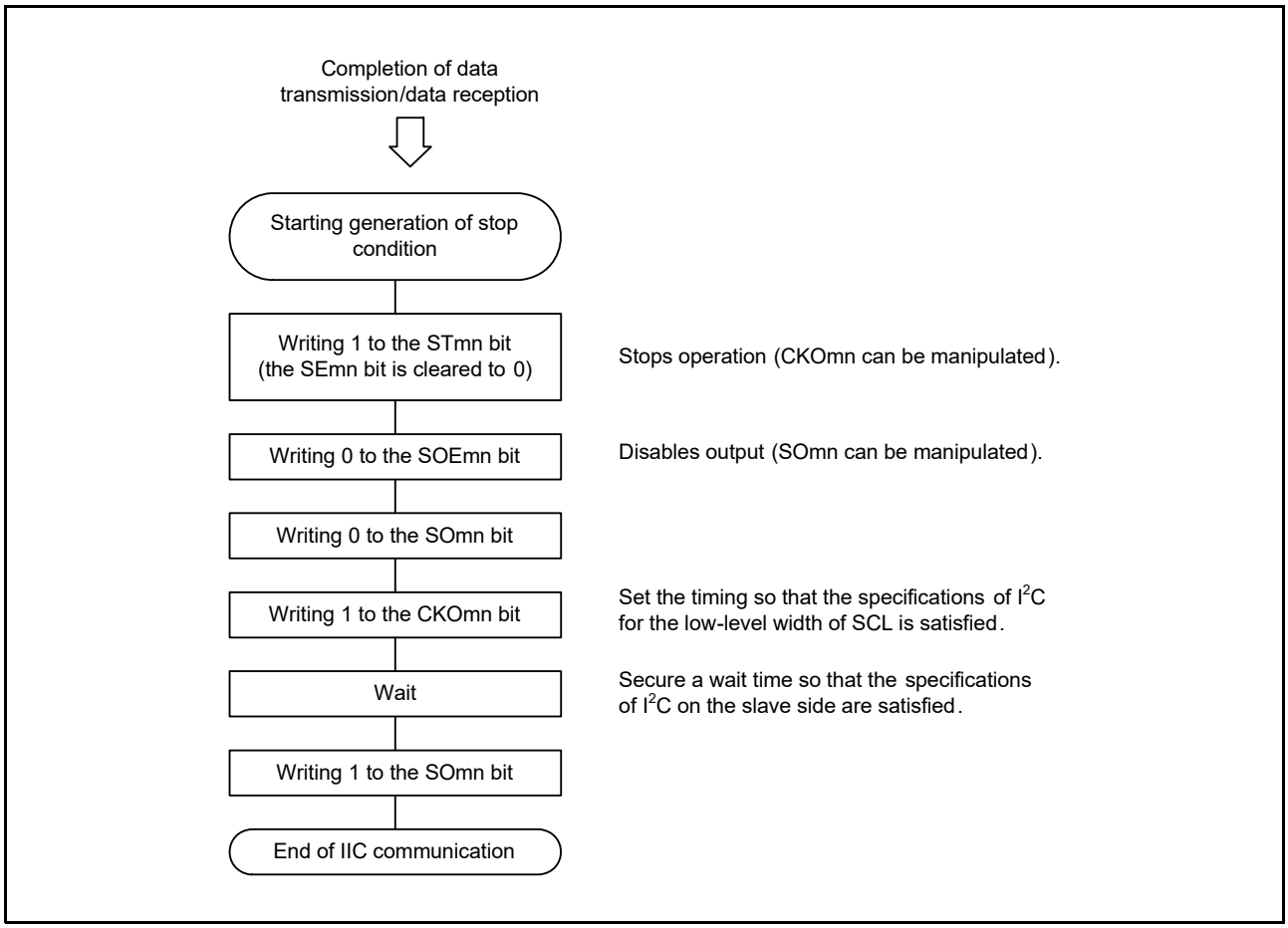
1. Processing flow

Figure 11 - 113 Timing Chart of Stop Condition Generation



Note During a receive operation, the SOEmn bit of serial output enable register m (SOEm) is cleared to 0 before receiving the last data.

Figure 11 - 114 Flowchart of Stop Condition Generation



11.8.5 Calculating transfer rate

The transfer rate for simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication can be calculated by the following expressions.

(Transfer rate) = {Operation clock (f_{MCK}) frequency of target channel} ÷ (SDRmn[15:9] + 1) ÷ 2

<R> **Caution SDRmn[15:9] must not be set to 0000000B. Set SDRmn[15:9] to 0000001B or greater. The duty ratio of the SCL signal output by the simplified I²C is 50%. The I²C bus specifications define that the low-level width of the SCL signal is longer than the high-level width. If 400 kbps (fast mode) or 1 Mbps (fast mode plus) is specified, therefore, the low-level width of the SCL output signal becomes shorter than the value specified in the I²C bus specifications. Make sure that the SDRmn[15:9] value satisfies the I²C bus specifications.**

Remark 1. The value of SDRmn[15:9] is the value of bits 15 to 9 of the SDRmn register (0000001B to 1111111B) and therefore is 1 to 127.

Remark 2. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

The operation clock (f_{MCK}) is determined by serial clock select register m (SPSm) and the CKSmn bit of serial mode register mn (SMRmn).

Table 11 - 9 Selection of Operation Clock For Simplified I²C

SMRmn Register	SPSm Register								Operation Clock (f_{MCK})[Note]	
CKSmn	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00		f_{CLK} = 32 MHz
0	×	×	×	×	0	0	0	0	f_{CLK}	32 MHz
	×	×	×	×	0	0	0	1	$f_{CLK}/2$	16 MHz
	×	×	×	×	0	0	1	0	$f_{CLK}/2^2$	8 MHz
	×	×	×	×	0	0	1	1	$f_{CLK}/2^3$	4 MHz
	×	×	×	×	0	1	0	0	$f_{CLK}/2^4$	2 MHz
	×	×	×	×	0	1	0	1	$f_{CLK}/2^5$	1 MHz
	×	×	×	×	0	1	1	0	$f_{CLK}/2^6$	500 kHz
	×	×	×	×	0	1	1	1	$f_{CLK}/2^7$	250 kHz
	×	×	×	×	1	0	0	0	$f_{CLK}/2^8$	125 kHz
	×	×	×	×	1	0	0	1	$f_{CLK}/2^9$	62.5 kHz
	×	×	×	×	1	0	1	0	$f_{CLK}/2^{10}$	31.25 kHz
	×	×	×	×	1	0	1	1	$f_{CLK}/2^{11}$	15.63 kHz
1	0	0	0	0	×	×	×	×	f_{CLK}	32 MHz
	0	0	0	1	×	×	×	×	$f_{CLK}/2$	16 MHz
	0	0	1	0	×	×	×	×	$f_{CLK}/2^2$	8 MHz
	0	0	1	1	×	×	×	×	$f_{CLK}/2^3$	4 MHz
	0	1	0	0	×	×	×	×	$f_{CLK}/2^4$	2 MHz
	0	1	0	1	×	×	×	×	$f_{CLK}/2^5$	1 MHz
	0	1	1	0	×	×	×	×	$f_{CLK}/2^6$	500 kHz
	0	1	1	1	×	×	×	×	$f_{CLK}/2^7$	250 kHz
	1	0	0	0	×	×	×	×	$f_{CLK}/2^8$	125 kHz
	1	0	0	1	×	×	×	×	$f_{CLK}/2^9$	62.5 kHz
	1	0	1	0	×	×	×	×	$f_{CLK}/2^{10}$	31.25 kHz
	1	0	1	1	×	×	×	×	$f_{CLK}/2^{11}$	15.63 kHz
Other than above									Setting prohibited	

Note When changing the clock selected for f_{CLK} (by changing the system clock control register (CKC) value), do so after having stopped (serial channel stop register m (STm) = 000FH) the operation of the serial array unit (SAU).

Remark 1. ×: Don't care

Remark 2. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

Here is an example of setting an I^2C transfer rate where $f_{MCK} = f_{CLK} = 32$ MHz.

I^2C Transfer Mode (Desired Transfer Rate)	f_{CLK} = 32 MHz			
	Operation Clock (f_{MCK})	SDRmn[15:9]	Calculated Transfer Rate	Error from Desired Transfer Rate
100 kHz	$f_{CLK}/2$	79	100 kHz	0.0%
400 kHz	f_{CLK}	41	380 kHz	5.0%Note
1 MHz	f_{CLK}	18	0.84 MHz	16.0%Note

Note The error cannot be set to about 0% because the duty ratio of the SCL signal is 50%.

11.8.6 Procedure for processing errors that occurred during simplified I2C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication

Table 11 - 10 and **Table 11 - 11** describe the procedure for processing errors that occurred during simplified I2C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication.

Table 11 - 10 Processing Procedure in Case of Overrun Error

Software Manipulation	State of the Hardware	Remark
Reads serial data register mn (SDRmn).	The BFFmn bit of the SSRmn register is set to 0 and channel n is enabled to receive data	This is to prevent an overrun error if the next reception is completed during error processing.
Reads serial status register mn (SSRmn).		The error type is identified and the read value is used to clear the error flag.
Writes 1 to serial flag clear trigger register mn (SIRmn).	The error flag is cleared.	Only the error during reading can be cleared, by writing the value read from the SSRmn register to the SIRmn register without modification.

Table 11 - 11 Processing Procedure in Case of ACK Error in Simplified I2C Mode

Software Manipulation	State of the Hardware	Remark
Reads serial status register mn (SSRmn).		The error type is identified and the read value is used to clear the error flag.
Writes serial flag clear trigger register mn (SIRmn).	The error flag is cleared.	Only the error during reading can be cleared, by writing the value read from the SSRmn register to the SIRmn register without modification.
Sets the STmn bit of serial channel stop register m (STm) to 1.	The SEmn bit of serial channel enable status register m (SEm) is set to 0 and channel n stops operation.	The slave is not ready for reception because ACK is not returned. Therefore, a stop condition is created, the bus is released, and communication is started again from the start condition. Or, a restart condition is generated and transmission can be redone from address transmission.
Creates a stop condition.		
Creates a start condition.		
Sets the SSmn bit of serial channel start register m (SSm) to 1.	The SEmn bit of serial channel enable status register m (SEm) is set to 1 and channel n is enabled to operate.	

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), r: IIC number (r = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

Section 12 Serial Interface IICA (IICA)

The number of the serial Interface IICA channels depends on the product.

	20-pin Products	32- to 64-pin Products
Number of channels	—	1

12.1 Functions of Serial Interface IICA

The serial interface IICA has the following three modes.

1. Operation stop mode
   This mode is used when serial transfers are not performed. The operating power can be reduced in this mode.

2. I^2C bus mode (multi-master supported)
   This mode is used for 8-bit data transfers with several devices via two lines: a serial clock (SCLAn) line and a serial data bus (SDAAn) line. This mode complies with the I^2C bus format and the master device can send start conditions, addresses, transfer directions, acknowledges (ACK), data, and stop conditions to the slave devices, via the serial data bus. The slave device automatically detects these states and data by hardware. This function can simplify the part of application program that controls the I^2C bus. Since the SCLAn and SDAAn pins are used for open drain outputs, serial interface IICA requires pull-up resistors for the serial clock line and the serial data bus line.

3. Wakeup mode
   The STOP mode can be released by generating an interrupt request signal (INTIICAn) when an extension code from the master device or the local address has been received while in STOP mode. This can be set by using the WUPn bit of IICA control register n1 (IICCTLn1). The all address match function is enabled by setting the SVADISn bit of the IICCTLn1 register to 1, allowing any received address is to be determined as a matched address.

Remark n = 0

Figure 12 - 1 shows a block diagram of serial interface IICA0.

Figure 12 - 1 Block Diagram of Serial Interface IICA0

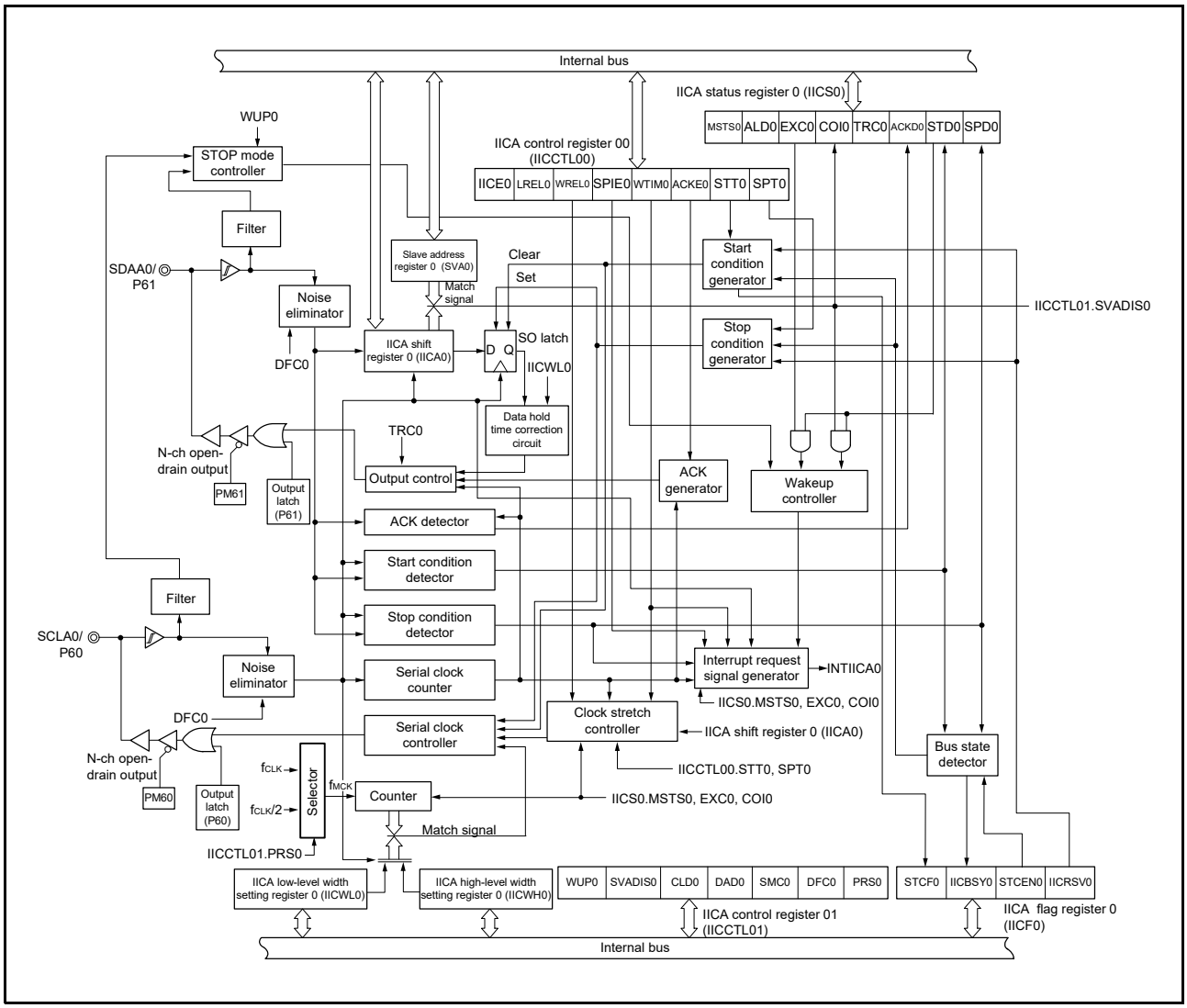
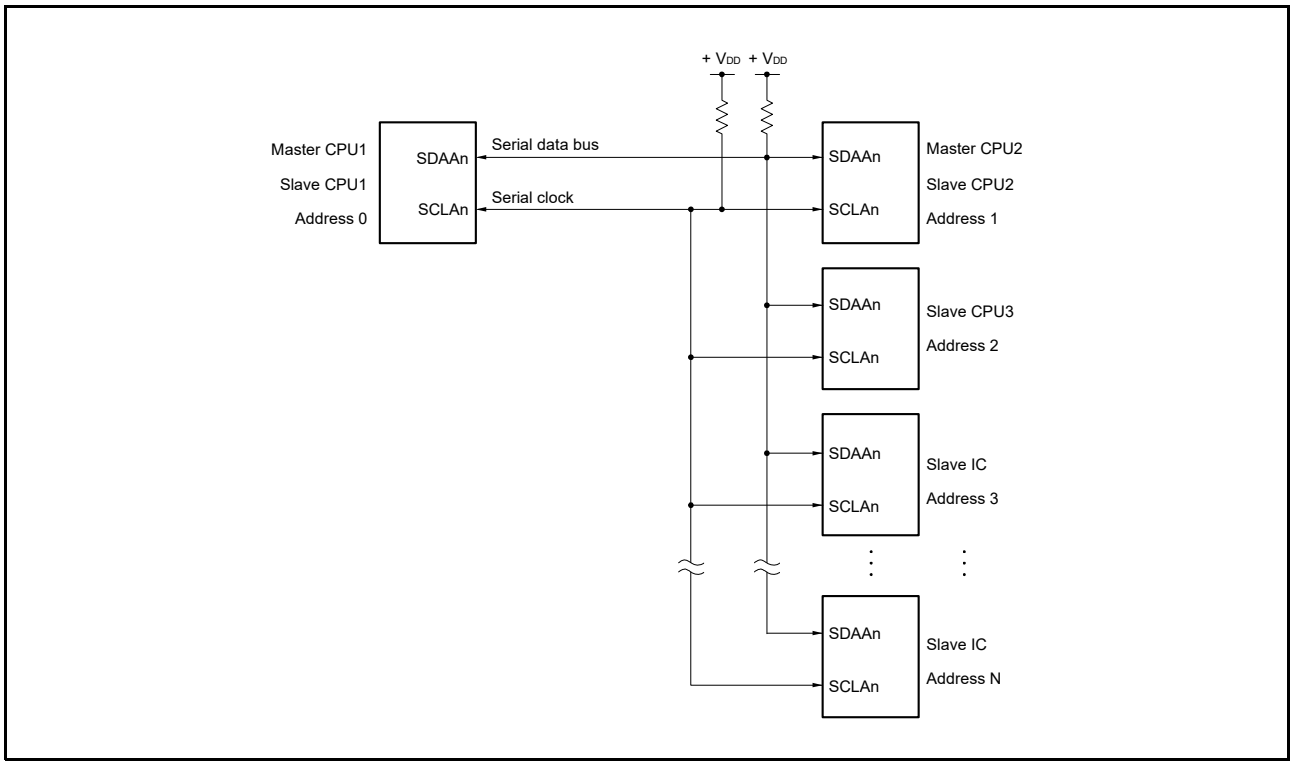


Figure 12 - 2 shows an example of the serial bus configuration using the I2C bus.

Figure 12 - 2 Example of the Serial Bus Configuration Using the I2C Bus



Remark n = 0

12.2 Configuration of Serial Interface IICA

Serial interface IICA includes the following hardware.

Table 12 - 1 Configuration of Serial Interface IICA

Item	Configuration
Registers	• IICA shift register n (IICAn) • Slave address register n (SVAn)
Control registers	• Peripheral enable register 0 (PER0) • Peripheral reset control register 0 (PRR0) • IICA control register n0 (IICCTLn0) • IICA status register n (IICSn) • IICA flag register n (IICFn) • IICA control register n1 (IICCTLn1) • IICA low-level width setting register n (IICWLn) • IICA high-level width setting register n (IICWHn) • Port mode registers (PMxx) • Port registers (Pxx)

Remark n = 0

1. IICA shift register n (IICAn)
   The IICAn register is used to convert 8-bit serial data to 8-bit parallel data and vice versa in synchronization with the serial clock. The IICAn register can be used for both transmission and reception. The actual transmit and receive operations can be controlled by writing to and reading from the IICAn register. Release serial interface IICA from the clock stretch state and start data transfer by writing data to the IICAn register during the clock stretch period. The IICAn register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 12 - 3 Format of IICA Shift Register n (IICAn)

Address: FFF50H (IICA0)
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
IICAn								

Caution 1. Do not write data to the IICAn register during data transfer.
Caution 2. Write to or read from the IICAn register only during the clock stretch period. Accessing the IICAn register in a communication state other than during the clock stretch period is prohibited. When the device serves as the master, however, the IICAn register can be written only once after the communication trigger bit (IICCTLn0.STTn) is set to 1.
Caution 3. When communication is reserved, write data to the IICAn register after the interrupt triggered by a stop condition is detected.

Remark n = 0

2. Slave address register n (SVAn)
   This register holds seven bits (A6, A5, A4, A3, A2, A1, and A0) of the local address when in slave mode. The SVAn register can be set by an 8-bit memory manipulation instruction. However, rewriting to this register is prohibited while STDn = 1 (while the start condition is detected). The value of this register following a reset is 00H.

Figure 12 - 4 Format of Slave Address Register n (SVAn)

Address: F0234H (SVA0)
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
SVAn	A6	A5	A4	A3	A2	A1	A0	0 **Note**

Note Bit 0 is fixed to 0.

Remark n = 0

3. SO latch
   The SO latch is used to retain the SDAAn pin's output level.

4. Wakeup controller
   This circuit generates an interrupt request signal (INTIICAn) when the received address matches the address value set to the slave address register n (SVAn), when any address is received while the all address match function is enabled, or when an extension code is received.

5. Serial clock counter
   This counter counts the serial clock cycles that are output or input during transmit/receive operations and is used to verify that 8-bit data was transmitted or received.

6. Interrupt request signal generator
   This circuit controls the generation of interrupt request signals (INTIICAn). An I^2C interrupt request is generated by the following two triggers.
   - Falling edge of eighth or ninth clock of the serial clock (set by the IICCTLn0.WTIMn bit)
   - Interrupt request generated when a stop condition is detected (set by the IICCTLn0.SPIEn bit)

7. Serial clock controller
   In master mode, this circuit generates the serial clock, which is output via the SCLAn pin.

8. Clock stretch controller
   This circuit controls the timing of clock stretching.

9. ACK generator, stop condition detector, start condition detector, and ACK detector
   These circuits generate or detect each state.

10. Data hold time correction circuit
    This circuit generates the hold time for data after the falling edge of the serial clock.

11. Start condition generator
    This circuit generates a start condition when the IICCTLn0.STTn bit is set to 1. However, while communication reservations are disabled (IICFn.IICRSVn = 1) and the bus is busy (IICFn.IICBSYn = 1), start condition requests are ignored and the IICFn.STCFn flag is set to 1.

12. Stop condition generator
    This circuit generates a stop condition when the IICCTLn0.SPTn bit is set to 1.

13. Bus state detector
    This circuit detects whether or not the bus is released by detecting start conditions and stop conditions. Note that the bus state cannot be detected immediately after the IICA operation has been enabled. Set the initial state of the bus state detector by the IICFn.STCENn bit.

Remark n = 0

12.3 Registers for Controlling Serial Interface IICA

The following registers are used to control serial interface IICA.

- Peripheral enable register 0 (PER0)
- Peripheral reset control register 0 (PRR0)
- IICA control register n0 (IICCTLn0)
- IICA status register n (IICSn)
- IICA flag register n (IICFn)
- IICA control register n1 (IICCTLn1)
- IICA low-level width setting register n (IICWLn)
- IICA high-level width setting register n (IICWHn)
- Port mode registers (PMxx)
- Port registers (Pxx)
- Port output mode registers (POMxx)
- Port mode control A registers (PMCAxx)

Remark n = 0

12.3.1 Peripheral enable register 0 (PER0)

The PER0 register is used to enable or disable the supply of a clock signal to various on-chip peripheral modules. Clock supply to an on-chip peripheral module that is not to be used can be stopped to decrease power consumption and noise. When serial interface IICA0 is to be used, be sure to set the IICA0EN bit of this register to 1. The PER0 register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 12 - 5 Format of Peripheral Enable Register 0 (PER0)

Address: F00F0H
After reset: 00H
R/W: R/W

Symbol	7	6	<5>	<4>	<3>	<2>	1	<0>
PER0	0	0	ADCEN	IICA0EN Note 1	SAU1EN	SAU0EN	0	TAU0EN

IICAnEN	Control of supply of an input clock to the serial interface IICAn
0	Stops supply of an input clock. • The SFRs used by the serial interface IICAn cannot be written.Note 2
1	Enables supply of an input clock. • The SFRs used by the serial interface IICAn can be read and written.

Note 1. This bit is only present in the 32- to 64-pin products.

Note 2. When an SFR used by the serial interface IICAn is read, the value returned is 00H or 0000H regardless of the setting value.

Caution 1. Be sure to clear bits 7, 6, and 1 to 0.

Caution 2. Be sure to clear bit 4 in the 20-pin products to 0.

Caution 3. When setting a serial interface IICA, start by setting the following registers while the setting of the IICAnEN bit is 1.

- **IICA control register n0 (IICCTLn0)**
- **IICA flag register n (IICFn)**
- **IICA status register n (IICSn)**
- **IICA control register n1 (IICCTLn1)**
- **IICA low-level width setting register n (IICWLn)**
- **IICA high-level width setting register n (IICWHn)**
- **IICA shift register n (IICAn)**
- **Slave address register n (SVAn)**

If the setting of the IICAnEN bit is 0, each register which controls the serial interface IICA is restored to its initial value and writing to any of those registers is ignored. Note, however, writing to the following registers is valid.

- **Port mode register (PM6)**
- **Port register (P6)**

Caution 4. While the operation of a peripheral function is enabled, do not switch the setting of the corresponding bit in the PER0 register. When switching the setting by the PER0 register, each peripheral function assigned to the PER0 register must be stopped.

Remark n = 0

12.3.2 Peripheral reset control register 0 (PRR0)

The PRR0 register is used to control resetting of the on-chip peripheral modules. Use this register to control reset or release from the reset for the target on-chip peripheral modules. The PRR0 register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 12 - 6 Format of Peripheral Reset Control Register 0 (PRR0)

Address: F00F1H
After reset: 00H
R/W: R/W

Symbol	7	6	<5>	<4>	<3>	<2>	1	<0>
PRR0	0	0	ADCRES	IICA0RES **Note**	SAU1RES	SAU0RES	0	TAU0RES

IICAnRES	Control resetting of the serial interface IICAn
0	Serial interface IICAn is released from the reset state.
1	Serial interface IICAn is in the reset state. • The SFRs for use with the serial interface IICAn are initialized.

Note This bit is only present in the 32- to 64-pin products.

Caution 1. Be sure to clear bits 7, 6, and 1 to 0.
Caution 2. Be sure to clear bit 4 in the 20-pin products to 0.

Remark n = 0

12.3.3 IICA control register n0 (IICCTLn0)

This register is used to enable or disable the I2C operations, set the timing of clock stretching, and set other I2C operations. The IICCTLn0 register can be set by a 1-bit or 8-bit memory manipulation instruction. Note that bits SPIEn, WTIMn, and ACKEn must be set while the setting of IICEn is 0 or this module is in the clock stretch state. These bits can be set at the same time as setting the IICEn bit 1. The value of this register following a reset is 00H.

Remark n = 0

Figure 12 - 7 Format of IICA Control Register n0 (IICCTLn0) (1/4)

Address: F0230H (IICCTL00)
After reset: 00H
R/W: R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IICCTLn0	IICEn	LRELn	WRELn	SPIEn	WTIMn	ACKEn	STTn	SPTn

IICEn	I2C operation enable
0	Stop operation. Reset the IICA status register n (IICSn).Note 1 Stop internal operation.
1	Enable operation.
Be sure to set this bit to 1 while the SCLAn and SDAAn lines are at high level.	
Condition for clearing (IICEn = 0)	Condition for setting (IICEn = 1)
• Cleared by instruction • Reset	• Set by instruction

LRELn Notes 2, 3	Exit from communications
0	Normal operation
1	IICA exits from the current communications and is in the standby mode. This bit is automatically cleared to 0 after having been set to 1. Its uses include cases in which a locally irrelevant extension code has been received. The SCLAn and SDAAn lines are set to high impedance. The following flags of IICA control register n0 (IICCTLn0) and the IICA status register n (IICSn) are cleared to 0. • STTn, SPTn, MSTSn, EXCn, COIn, TRCn, ACKDn, and STDn
The standby mode following exit from communications remains in effect until the following communications entry conditions are met. • After a stop condition is detected, restart is in master mode. • An address match, extension code reception, or address reception with the all address match function enabled occurs after the start condition.	
Condition for clearing (LRELn = 0)	Condition for setting (LRELn = 1)
• Automatically cleared after execution • Reset	• Set by instruction

Figure 12 - 7 Format of IICA Control Register n0 (IICCTLn0) (2/4)

WRELn **Notes 2, 3**	Release from the clock stretch state
0	The interface is not released from the clock stretch state.
1	The interface is released from the clock stretch state. After release from the clock stretch state, this bit is automatically cleared to 0.
When the WRELn bit is set (for release from the clock stretch state) during the clock stretch period at the ninth clock pulse in the transmission state (IICSn.TRCn = 1), the SDAAn line goes into the high impedance state (IICSn.TRCn = 0).	
Condition for clearing (WRELn = 0)	Condition for setting (WRELn = 1)
• Automatically cleared after execution • Reset	• Set by instruction

SPIEn**Note 2**	Enable/disable generation of interrupt request when stop condition is detected
0	Disable
1	Enable
If IICCTLn1.WUPn = 1, no stop condition interrupt will be generated even if SPIEn = 1.	
Condition for clearing (SPIEn = 0)	Condition for setting (SPIEn = 1)
• Cleared by instruction • Reset	• Set by instruction

WTIMn **Note 4**	Control of clock stretching and interrupt request generation
0	An interrupt request is generated on the falling edge of the eighth clock cycle. • Master mode: After the output of eight clock pulses, the clock output is set to the low level and clock stretching is set. • Slave mode: After the input of eight clock pulses, the clock is set to the low level and clock stretching is set for the master device.
1	An interrupt request is generated on the falling edge of the ninth clock cycle. • Master mode: After the output of nine clock pulses, the clock output is set to the low level and clock stretching is set. • Slave mode: After the input of nine clock pulses, the clock is set to the low level and clock stretching is set for the master device.
An interrupt is generated on the falling edge of the ninth clock cycle during address transfer independently of the setting of this bit. The setting of this bit is valid when the address transfer is completed. In master mode, clock stretching is inserted at the falling edge of the ninth clock cycle during address transfer. For a slave device that has received a local address, clock stretching is inserted at the falling edge of the ninth clock cycle after an acknowledge (ACK) is issued. However, if the slave device has received an extension code, clock stretching is inserted at the falling edge of the eighth clock cycle. When an address is received while the all address match function is enabled, clock stretching is inserted at the falling edge of the eighth clock cycle.	
Condition for clearing (WTIMn = 0)	Condition for setting (WTIMn = 1)
• Cleared by instruction • Reset	• Set by instruction

Figure 12 - 7 Format of IICA Control Register n0 (IICCTLn0) (3/4)

ACKEn **Notes 4, 5**	Acknowledgment control
0	Disable acknowledgment.
1	Enable acknowledgment. During the ninth clock period, the SDAAn line is set to low level.

Condition for clearing (ACKEn = 0)	Condition for setting (ACKEn = 1)
• Cleared by instruction • Reset	• Set by instruction

STTn **Notes 2, 6**	Start condition trigger
0	Do not generate a start condition.
1	When bus is released (in standby state, when IICFn.IICBSYn = 0): If this bit is set to 1, a start condition is generated (startup as the master). When a third party is communicating: • When communication reservation function is enabled (IICFn.IICRSVn = 0) Functions as the start condition reservation flag. When set to 1, automatically generates a start condition after the bus is released. • When communication reservation function is disabled (IICFn.IICRSVn = 1) Even if this bit is set to 1, the STTn bit is cleared and the STTn clear flag (IICFn.STCFn) is set to 1. No start condition is generated. In the clock stretch state (for a master device): Generates a restart condition after release from the clock stretch state.

Cautions concerning set timing
- For master reception: Cannot be set to 1 during transfer. Can be set to 1 only during the clock stretch period when the ACKEn bit has been cleared to 0 and slave has been notified of final reception.
- For master transmission: A start condition cannot be generated normally during the acknowledge period. Set to 1 during the clock stretch period that follows output of the ninth clock.
- Cannot be set to 1 at the same time as stop condition trigger (SPTn).
- Once STTn is set to 1, setting it to 1 again before the clear condition is met is not allowed.

Condition for clearing (STTn = 0)	Condition for setting (STTn = 1)
• Cleared by setting the STTn bit to 1 while communication reservation is prohibited. • Cleared by loss in arbitration • Cleared after start condition is generated by master device • Cleared by LRELn = 1 (exit from communications) • When IICEn = 0 (operation stop) • Reset	• Set by instruction

Figure 12 - 7 Format of IICA Control Register n0 (IICCTLn0) (4/4)

SPTn[Note 7]	Stop condition trigger
0	Stop condition is not generated.
1	Stop condition is generated (termination of master device's transfer).

Cautions concerning set timing
- For master reception: Cannot be set to 1 during transfer. Can be set to 1 only during the clock stretch period when the ACKEn bit has been cleared to 0 and slave has been notified of final reception.
- For master transmission: A stop condition cannot be generated normally during the acknowledge period. Therefore, set it during the clock stretch period that follows output of the ninth clock.
- Cannot be set to 1 at the same time as start condition trigger (STTn).
- The SPTn bit can be set to 1 only when in master mode.
- When the WTIMn bit has been cleared to 0, if the SPTn bit is set to 1 during the clock stretch period that follows output of eight clock pulses, note that a stop condition will be generated during the high-level period of the ninth clock after release from the clock stretch state. The WTIMn bit should be changed from 0 to 1 during the clock stretch period following the output of eight clock pulses, and the SPTn bit should be set to 1 during the clock stretch period that follows the output of the ninth clock.
- Once SPTn is set to 1, setting it to 1 again before the clear condition is met is not allowed.

Condition for clearing (SPTn = 0)	Condition for setting (SPTn = 1)
• Cleared by loss in arbitration • Automatically cleared after stop condition is detected • Cleared by LRELn = 1 (exit from communications) • When IICEn = 0 (operation stop) • Reset	• Set by instruction

Note 1. The IICA status register n (IICSn), the STCFn and IICBSYn flags of the IICA flag register n (IICFn), and the CLDn and DADn bits of IICA control register n1 (IICCTLn1) are reset.

Note 2. The setting of this bit has no effect while the setting of IICEn is 0.

Note 3. Reading the LRELn and WRELn bits always returns 0.

Note 4. The signal of this bit is invalid while IICEn is 0. Set this bit during that period.

Note 5. The set value is invalid during address transfer and if the code is not an extension code, and the all address match function is disabled. When the device serves as a slave and the addresses match, an acknowledgment is generated regardless of the set value.

Note 6. The STTn bit is always read as 0.

Note 7. The SPTn bit is always read as 0.

Caution 1. If the operation of I^2C is enabled (IICEn = 1) when the SCLAn line is high level, the SDAAn line is low level, and the digital filter is turned on (IICCTLn1.DFCn = 1), a start condition will be inadvertently detected immediately. In this case, set the LRELn bit to 1 by using a 1-bit memory manipulation instruction immediately after enabling operation of I^2C (IICEn = 1).

Caution 2. When the IICSn.TRCn bit 1 is set to 1 (transmission state), the WRELn bit of the IICA control register n0 is set to 1 during the ninth clock and the interface is released from the clock stretch state, after which the TRCn bit is cleared (reception state) and the SDAAn line is set to the high impedance state. Release the interface from the clock stretch state while the TRCn bit is 1 (transmission state) by writing to the IICA shift register n.

Remark 1. The SPTn bit becomes 0 when it is read after data setting.

Remark 2. n = 0

12.3.4 IICA status register n (IICSn)

This register indicates the state of the I^2C. The IICSn register can only be read by a 1-bit or 8-bit memory manipulation instruction while the setting of IICCTLn0.STTn is 1 or this module is in the clock stretch state. The value of this register following a reset is 00H.

Caution **Reading the IICSn register while the address match wakeup function is enabled (IICCTLn1.WUPn = 1) in STOP mode is prohibited. When the WUPn bit is changed from 1 to 0 (stopping operation of wakeup function), regardless of the INTIICAn interrupt request signal, a change in the state is not reflected until the next start condition or stop condition is detected. To use the wakeup function, therefore, enable (IICCTLn0.SPIEn = 1) the interrupt generated by detecting a stop condition and read the IICSn register after the interrupt has been detected.**

Remark n = 0

Figure 12 - 8 Format of IICA Status Register n (IICSn) (1/4)

Address: FFF51H (IICS0)
After reset: 00H
R/W: R

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IICSn	MSTSn	ALDn	EXCn	COIn	TRCn	ACKDn	STDn	SPDn

MSTSn	Master status check flag
0	Slave state or communications standby state
1	Master communications state

Condition for clearing (MSTSn = 0)	Condition for setting (MSTSn = 1)
• When a stop condition is detected • When ALDn = 1 (arbitration loss) • Cleared by IICCTLn0.LRELn = 1 (exit from communications) • When IICCTLn0.IICEn changes from 1 to 0 (operation stop) • Reset	• When a start condition is generated

ALDn	Detection of arbitration loss
0	This value indicates either that arbitration is not in progress or that the result of arbitration was a win.
1	This value indicates that the result of arbitration was a loss. The MSTSn flag is cleared when the ALDn bit has this value.

Condition for clearing (ALDn = 0)	Condition for setting (ALDn = 1)
• Automatically cleared after the IICSn register is read**Note 1** • When IICCTLn0.IICEn changes from 1 to 0 (operation stop) • Reset	• When the arbitration result is a “loss”.

Figure 12 - 8 Format of IICA Status Register n (IICSn) (2/4)

EXCn	Detection of extension code reception
0	Extension code was not received.
1	Extension code was received. Or, the all address match function is enabled.

Condition for clearing (EXCn = 0)	Condition for setting (EXCn = 1)
• When a start condition is detected • When a stop condition is detected • Cleared by IICCTLn0.LRELn = 1 (exit from communications) • When IICCTLn0.IICEn bit changes from 1 to 0 (operation stop) • Reset	• When the higher four bits of the received address data is either 0000 or 1111 (set at the rising edge of the eighth clock). • When an address is received while the all address match function is enabled (IICCTLn1.SVADISn = 1) (set at the rising edge of the eighth clock).

COIn	Detection of matching addresses
0	Addresses do not match.
1	Addresses match. Or, the all address match function is enabled.

Condition for clearing (COIn = 0)	Condition for setting (COIn = 1)
• When a start condition is detected • When a stop condition is detected • Cleared by IICCTLn0.LRELn = 1 (exit from communications) • When IICCTLn0.IICEn changes from 1 to 0 (operation stop) • Reset	• When the received address matches the local address (slave address register n (SVAn)) (set at the rising edge of the eighth clock). • When an address is received while the all address match function is enabled (IICCTLn1.SVADISn = 1) (set at the rising edge of the eighth clock).

Figure 12 - 8 Format of IICA Status Register n (IICSn) (3/4)

TRCn	Detection of transmission/reception state
0	Reception state (non-transmission state). The SDAAn line is set for high impedance.
1	Transmission state. The value in the SOn latch is enabled for output to the SDAAn line (valid starting at the falling edge of the first byte's ninth clock).

Condition for clearing (TRCn = 0)	Condition for setting (TRCn = 1)
<Both master and slave> • When a stop condition is detected • Cleared by IICCTLn0.LRELn = 1 (exit from communications) • When IICCTLn0.IICEn changes from 1 to 0 (operation stop) • Cleared by IICCTLn0.WRELn = 1 **Note 2** (release from the clock stretch state) • When the ALDn bit changes from 0 to 1 (arbitration loss) • Reset • When not used for communication (MSTSn, EXCn, COIn = 0) <Master> • When 1 is output to the first byte's LSB (transfer direction specification bit) <Slave> • When a start condition is detected • When 0 is input to the first byte's LSB (transfer direction specification bit)	<Master> • When a start condition is generated • When 0 (master transmission) is output to the LSB (transfer direction specification bit) of the first byte (during address transfer) <Slave> • When 1 (slave transmission) is input to the LSB (transfer direction specification bit) of the first byte from the master (during address transfer)

ACKDn	Detection of acknowledge (ACK)
0	Acknowledge was not detected.
1	Acknowledge was detected.

Condition for clearing (ACKDn = 0)	Condition for setting (ACKDn = 1)
• When a stop condition is detected • At the rising edge of the next byte's first clock • Cleared by IICCTLn0.LRELn = 1 (exit from communications) • When IICCTLn0.IICEn changes from 1 to 0 (operation stop) • Reset	• After the SDAAn line is set to low level at the rising edge of SCLAn line's ninth clock

Figure 12 - 8 Format of IICA Status Register n (IICSn) (4/4)

STDn	Detection of start condition
0	Start condition was not detected.
1	Start condition was detected. This indicates that the address transfer period is in effect.

Condition for clearing (STDn = 0)	Condition for setting (STDn = 1)
• When a stop condition is detected • At the rising edge of the next byte's first clock following address transfer • Cleared by IICCTLn0.LRELn = 1 (exit from communications) • When IICCTLn0.IICEn changes from 1 to 0 (operation stop) • Reset	• When a start condition is detected

SPDn	Detection of stop condition
0	Stop condition was not detected.
1	Stop condition was detected. The master device's communication is terminated and the bus is released.

Condition for clearing (SPDn = 0)	Condition for setting (SPDn = 1)
• At the rising edge of the address transfer byte's first clock following setting of this bit and detection of a start condition • When IICCTLn1.WUPn changes from 1 to 0 • When IICCTLn0.IICEn changes from 1 to 0 (operation stop) • Reset	• When a stop condition is detected

Note 1. The ALDn bit is also cleared when a 1-bit memory manipulation instruction is executed for another bit in the IICSn register. Therefore, when using the ALDn bit, read the data of this bit before the data of the other bits.

Note 2. When the TRCn flag of the IICA status register n (IICSn) is set to 1 (transmission state), setting the WRELn bit of the IICA control register n0 (IICCTLn0) to 1 at the ninth clock releases the interface from the clock stretch state, after which the TRCn flag is cleared (reception state) and the SDAAn line is set to the high impedance state. Release the interface from the clock stretch state while the TRCn bit is 1 (transmission state) by writing to the IICA shift register n.

Remark n = 0

12.3.5 IICA flag register n (IICFn)

This register sets the operation mode of I²C and indicates the state of the I²C bus. The IICFn register can be set by a 1-bit or 8-bit memory manipulation instruction. However, the STTn clear flag (STCFn) and I²C bus status flag (IICBSYn) bits are read-only. The IICRSVn bit can be used to enable or disable the communication reservation. The STCENn bit can be used to set the initial value of the IICBSYn flag. The IICRSVn and STCENn bits can be written only when the operation of I²C is disabled (IICCTLn0.IICEn = 0). The IICFn register is read-only while the operation of the I²C is enabled. The value of this register following a reset is 00H.

Remark n = 0

Figure 12 - 9 Format of IICA Flag Register n (IICFn) (1/2)

Address: FFF52H (IICF0)
After reset: 00H
R/W: R/W**Note**

Symbol	<7>	<6>	5	4	3	2	<1>	<0>
IICFn	STCFn	IICBSYn	0	0	0	0	STCENn	IICRSVn

STCFn	STTn clear flag
0	Generate start condition
1	Start condition generation unsuccessful: clear the STTn flag

Condition for clearing (STCFn = 0)	Condition for setting (STCFn = 1)
• Cleared by IICCTLn0.STTn = 1 • When IICCTLn0.IICEn = 0 (operation stop) • Reset	• Generating start condition unsuccessful and the IICCTLn0.STTn bit cleared to 0 when communication reservation is disabled (IICRSVn = 1).

IICBSYn	I²C bus status flag
0	Bus released state (initial communications state when STCENn = 1)
1	Bus communications state (initial communications state when STCENn = 0)

Condition for clearing (IICBSYn = 0)	Condition for setting (IICBSYn = 1)
• When a stop condition is detected • When IICCTLn0.IICEn = 0 (operation stop) • Reset	• When a start condition is detected • Setting of the IICCTLn0.IICEn bit when STCENn = 0

STCENn	Initial start enable trigger
0	After operation is enabled (IICCTLn0.IICEn = 1), enable generation of a start condition upon detection of a stop condition.
1	After operation is enabled (IICCTLn0.IICEn = 1), enable generation of a start condition without detecting a stop condition.

Condition for clearing (STCENn = 0)	Condition for setting (STCENn = 1)
• Cleared by instruction • When a start condition is detected • Reset	• Set by instruction

Figure 12 - 9 Format of IICA Flag Register n (IICFn) (2/2)

IICRSVn	Communication reservation function disable bit
0	Enable communication reservation
1	Disable communication reservation

Condition for clearing (IICRSVn = 0)	Condition for setting (IICRSVn = 1)
• Cleared by instruction • Reset	• Set by instruction

Note Bits 7 and 6 are read-only.

Caution 1. Write to the STCENn bit only when the operation is stopped (IICCTLn0.IICEn = 0).

Caution 2. As the bus is recognized as being in the released state (IICBSYn = 0) regardless of its actual state when STCENn = 1. When generating the first start condition (IICCTLn0.STTn = 1), it is necessary to verify that no third party communications are in progress in order to prevent such communications from being destroyed.

Caution 3. Write to the IICRSVn bit only when the operation is stopped (IICCTLn0.IICEn = 0).

Remark n = 0

12.3.6 IICA control register n1 (IICCTLn1)

This register is used to set the operation mode of I2C and detect the states of the SCLAn and SDAAn pins. The IICCTLn1 register can be set by a 1-bit or 8-bit memory manipulation instruction. However, the CLDn and DADn bits are read-only. Set the IICCTLn1 register, except the WUPn bit, while operation of I2C is disabled (IICCTLn0.IICEn = 0). The value of this register following a reset is 00H.

Remark n = 0

Figure 12 - 10 Format of IICA Control Register n1 (IICCTLn1) (1/2)

Address: F0231H (IICCTL01)
After reset: 00H
R/W: R/W**Note 1**

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	1	<0>
IICCTLn1	WUPn	SVADISn	CLDn	DADn	SMCn	DFCn	0	PRSn

WUPn	Control of address match wakeup
0	Stops operation of address match wakeup function in STOP mode.
1	Enables operation of address match wakeup function in STOP mode.
To shift to STOP mode when WUPn = 1, execute the STOP instruction at least three cycles of fMCK after setting the WUPn bit to 1. See **Figure 12 - 22 Flow for Setting WUPn = 1**. Clear the WUPn bit to 0 after the address has matched, an address has been received while the all address match function is enabled, or an extension code has been received. The subsequent communication can be entered by clearing the WUPn bit to 0. The interface must be released from the clock stretch state and transmit data must be written after the WUPn bit has been cleared to 0. The interrupt timing when the address has matched, when an address has been received while the all address match function is enabled, or when an extension code has been received, while WUPn = 1, is identical to the interrupt timing when WUPn = 0. A delay of the difference of sampling by the clock will occur. Furthermore, when WUPn = 1, a stop condition interrupt is not generated even if IICCTLn0.SPIEn is set to 1.	
Condition for clearing (WUPn = 0)	Condition for setting (WUPn = 1)
• Cleared by instruction (after address match, address reception with the all address match function enabled, or extension code reception)	• Set by instruction (when the MSTSn, EXCn, and COIn bits are 0, and the STDn bit also 0 (communication not entered))**Note 2**

SVADISn	Address match disabling flag
0	Disables the all address match function.
1	Enables the all address match function.
When SVADISn = 1, IICA considers any address as address match, and performs the same operation as that when an extension code is received. Therefore, IICSn.COIn is set to 1, and IICSn.EXCn is set to 1. For details on extension code reception, see **12.5.11 Extension code**.	

Figure 12 - 10 Format of IICA Control Register n1 (IICCTLn1) (2/2)

CLDn	Detection of SCLAn pin level (valid only when IICCTLn0.IICEn = 1)
0	The SCLAn pin was detected at low level.
1	The SCLAn pin was detected at high level.

Condition for clearing (CLDn = 0)	Condition for setting (CLDn = 1)
• When the SCLAn pin is at low level • When IICCTLn0.IICEn = 0 (operation stop) • Reset	• When the SCLAn pin is at high level

DADn	Detection of SDAAn pin level (valid only when IICCTLn0.IICEn = 1)
0	The SDAAn pin was detected at low level.
1	The SDAAn pin was detected at high level.

Condition for clearing (DADn = 0)	Condition for setting (DADn = 1)
• When the SDAAn pin is at low level • When IICCTLn0.IICEn = 0 (operation stop) • Reset	• When the SDAAn pin is at high level

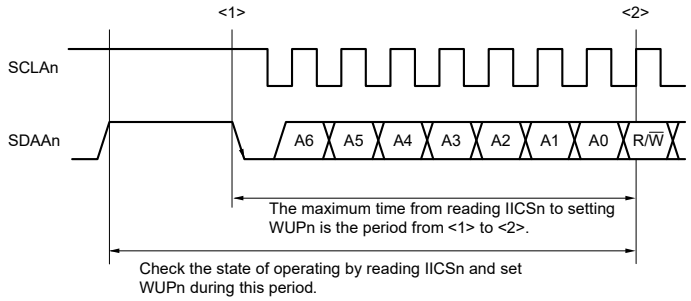
SMCn	Operation mode switching
0	Operates in standard mode (fastest transfer rate: 100 kbps).
1	Operates in fast mode (fastest transfer rate: 400 kbps) or fast mode plus (fastest transfer rate: 1 Mbps).

DFCn	Digital filter operation control
0	Digital filter off.
1	Digital filter on.
Use the digital filter only in fast mode and fast mode plus. The digital filter is used for noise elimination. The transfer clock does not vary, regardless of the DFCn bit being set to 1 or cleared to 0.	

PRSn	Control of IICA operation clock (f_{MCK})
0	Selects f_{CLK} ($1\ MHz \leq f_{CLK} \leq 20\ MHz$).
1	Selects $f_{CLK}/2$ ($20\ MHz < f_{CLK}$).

Note 1. Bits 5 and 4 are read-only.

Note 2. The state of the IICA status register n (IICSn) must be checked and the WUPn bit must be set during the period shown below.



(Cautions and Remark are listed on the next page.)

Caution 1. **The fastest operation frequency of the IICA operation clock (f_{MCK}) is 20 MHz (max.). Only set the PRSn bit to 1 when f_{CLK} exceeds 20 MHz.**

Caution 2. **Note the minimum f_{CLK} operation frequency when setting the transfer clock. The minimum f_{CLK} operation frequency for serial interface IICA is determined according to the mode.**
Fast mode: f_{CLK} = 3.5 MHz (min.)
Fast mode plus: f_{CLK} = 10 MHz (min.)
Normal mode: f_{CLK} = 1 MHz (min.)

Remark n = 0

12.3.7 IICA low-level width setting register n (IICWLn)

This register is used to set the low-level width (t_{LOW}) of the SCLAn pin signal that is output by serial interface IICA and to control the SDAAn pin signal. The IICWLn register can be set by an 8-bit memory manipulation instruction. Set the IICWLn register while operation of I^2C is disabled (IICCTLn0.IICEn = 0). For details on setting the IICWLn register, see **12.4.2 Setting transfer clock by using IICWLn and IICWHn registers**. The data hold time is one-quarter of the time set by the IICWLn register. The value of this register following a reset is FFH.

Figure 12 - 11 Format of IICA Low-level Width Setting Register n (IICWLn)

Address: F0232H (IICWL0)
After reset: FFH
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
IICWLn								

Remark 1. The minimum cycle of the serial clock is (IICWLn + 1) + (IICWHn + 1).
Remark 2. n = 0

12.3.8 IICA high-level width setting register n (IICWHn)

This register is used to set the high-level width of the SCLAn pin signal that is output by serial interface IICA and to control the SDAAn pin signal. The IICWHn register can be set by an 8-bit memory manipulation instruction. Set the IICWHn register while operation of I^2C is disabled (IICCTLn0.IICEn = 0). For details on setting the IICWHn register, see **12.4.2 Setting transfer clock by using IICWLn and IICWHn registers**. The value of this register following a reset is FFH.

Figure 12 - 12 Format of IICA High-level Width Setting Register n (IICWHn)

Address: F0233H (IICWH0)
After reset: FFH
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
IICWHn								

Remark 1. The minimum cycle of the serial clock is (IICWLn + 1) + (IICWHn + 1).
Remark 2. n = 0

12.3.9 Registers for controlling the port functions multiplexed with the IICA serial interface inputs and outputs

Set the following registers to control the port functions multiplexed with the serial interface IICA inputs and outputs.

- Port mode registers (PMxx)
- Port registers (Pxx)
- Port output mode registers (POMxx)
- Port mode control A registers (PMCAxx)

For details, see the following sections.

- **4.3.1 Port mode registers (PMxx)**
- **4.3.2 Port registers (Pxx)**
- **4.3.5 Port output mode registers (POMxx)**
- **4.3.7 Port mode control A registers (PMCAxx)**

For examples of register settings for controlling port functions, see **4.5.4 Examples of register settings for port and alternate functions**.

12.4 I2C Bus Mode Functions

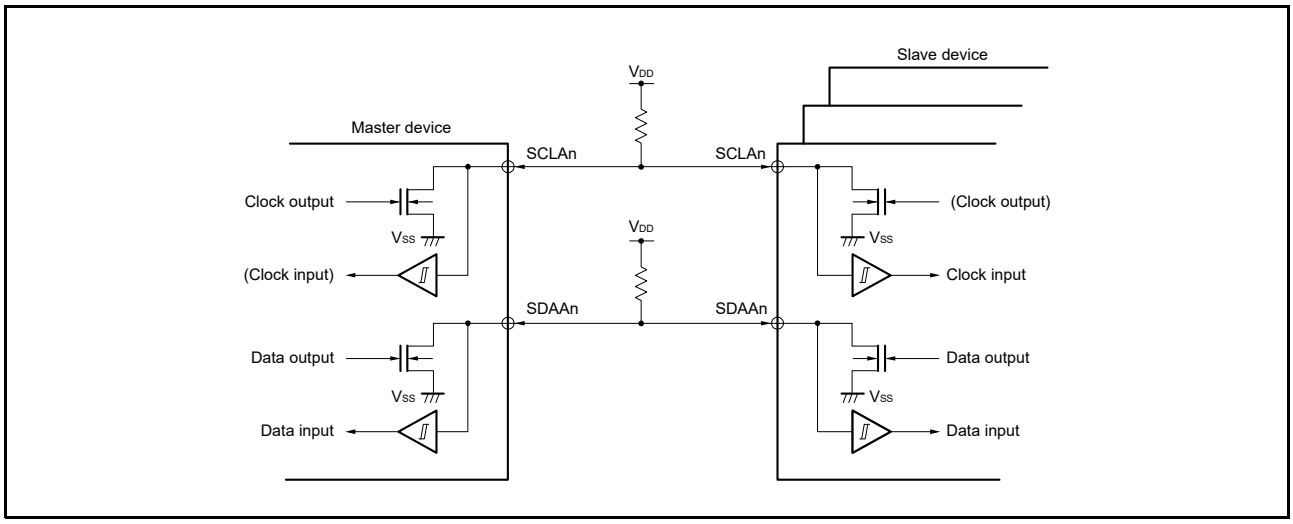
12.4.1 Pin configuration

The serial clock pin (SCLAn) and the serial data bus pin (SDAAn) are configured as follows.

1. SCLAn: This pin is used for serial clock input and output. This pin is an N-ch open-drain output for both master and slave devices. Input is Schmitt input.
2. SDAAn: This pin is used for serial data input and output. This pin is an N-ch open-drain output for both master and slave devices. Input is Schmitt input.

Since outputs from the serial clock line and the serial data bus line are N-ch open-drain outputs, an external pull-up resistor is required.

Figure 12 - 13 Pin Configuration Diagram



Remark n = 0

12.4.2 Setting transfer clock by using IICWLn and IICWHn registers

1. Setting transfer clock on master side

$$\text{Transfer clock} = \frac{f_{MCK}}{IICWL + IICWH + f_{MCK}(t_R + t_F)}$$

At this time, the optimal setting values of the IICWLn and IICWHn registers are as follows.
(The fractional parts of all setting values are rounded up.)

- When the fast mode

$$IICWLn = \frac{0.52}{\text{Transfer clock}} \times f_{MCK}$$

$$IICWHn = \left(\frac{0.48}{\text{Transfer clock}} - t_R - t_F\right) \times f_{MCK}$$

- When the normal mode

$$IICWLn = \frac{0.47}{\text{Transfer clock}} \times f_{MCK}$$

$$IICWHn = \left(\frac{0.53}{\text{Transfer clock}} - t_R - t_F\right) \times f_{MCK}$$

- When the fast mode plus

$$IICWLn = \frac{0.50}{\text{Transfer clock}} \times f_{MCK}$$

$$IICWHn = \left(\frac{0.50}{\text{Transfer clock}} - t_R - t_F\right) \times f_{MCK}$$

2. Setting IICWLn and IICWHn registers on slave side
(The fractional parts of all setting values are rounded up.)
   - When the fast mode
     $IICWLn = 1.3\ \mu s \times f_{MCK}$
     $IICWHn = (1.2\ \mu s - t_R - t_F) \times f_{MCK}$
   - When the normal mode
     $IICWLn = 4.7\ \mu s \times f_{MCK}$
     $IICWHn = (5.3\ \mu s - t_R - t_F) \times f_{MCK}$
   - When the fast mode plus
     $IICWLn = 0.50\ \mu s \times f_{MCK}$
     $IICWHn = (0.50\ \mu s - t_R - t_F) \times f_{MCK}$

Caution 1. The fastest operation frequency of the IICA operation clock (f_{MCK}) is 20 MHz (max.). Set the PRSn bit of the IICA control register n1 (IICCTLn1) to 1 only when f_{CLK} exceeds 20 MHz.

Caution 2. Note the minimum f_{CLK} operation frequency when setting the transfer clock. The minimum f_{CLK} operation frequency for serial interface IICA is determined according to the mode.
Fast mode: f_{CLK} = 3.5 MHz (min.)
Fast mode plus: f_{CLK} = 10 MHz (min.)
Normal mode: f_{CLK} = 1 MHz (min.)

Caution 3. The minimum cycle of the serial clock is (IICWLn + 1) + (IICWHn + 1). Determine the values to be set in the IICWLn and IICWHn registers, taking into account the rising time (t_R) and falling time (t_F) of the SDAAn and SCLAn signals.

(Remarks are listed on the next page.)

Remark 1. Calculate the rise time (t_R) and fall time (t_F) of the SDAAn and SCLAn signals separately, because they differ depending on the pull-up resistor and wire load.

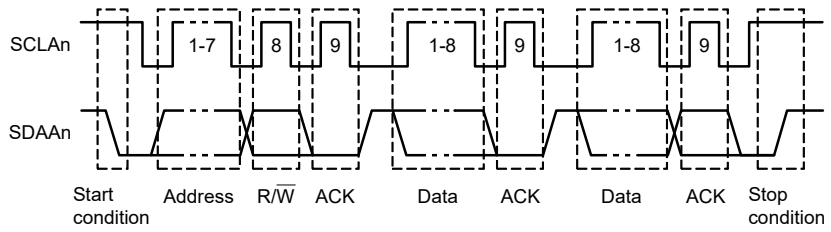
Remark 2. IICWLn: IICA low-level width setting register n
IICWHn: IICA high-level width setting register n
t_F: SDAAn and SCLAn signal falling times
t_R: SDAAn and SCLAn signal rising times
f_{MCK}: IICA operation clock frequency

Remark 3. n = 0

12.5 I2C Bus Definitions and Control Methods

The following section describes the I2C bus's serial data communication format and the signals used by the I2C bus. **Figure 12 - 14** shows the transfer timing for the “start condition”, “address”, “data”, and “stop condition” output via the I2C bus's serial data bus.

Figure 12 - 14 I2C Bus Serial Data Transfer Timing

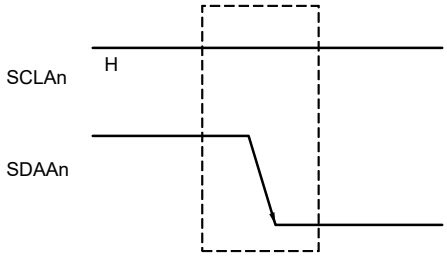


The master device generates the start condition, slave address, and stop condition. The acknowledge (ACK) can be generated by either the master or slave device (normally, it is output by the device that receives 8-bit data). The serial clock (SCLAn) is continuously output by the master device. However, for the slave device, the period over which the SCLAn pin is at the low level can be extended and clock stretching can be inserted.

12.5.1 Start conditions

<R> When the SCLAn pin is at high level, changing the SDAAn pin from the high level to the low level generates a start condition. A start condition is a signal that the master device generates to the slave device when starting a serial transfer. When the device is used as a slave, start conditions can be detected.

Figure 12 - 15 Start Conditions



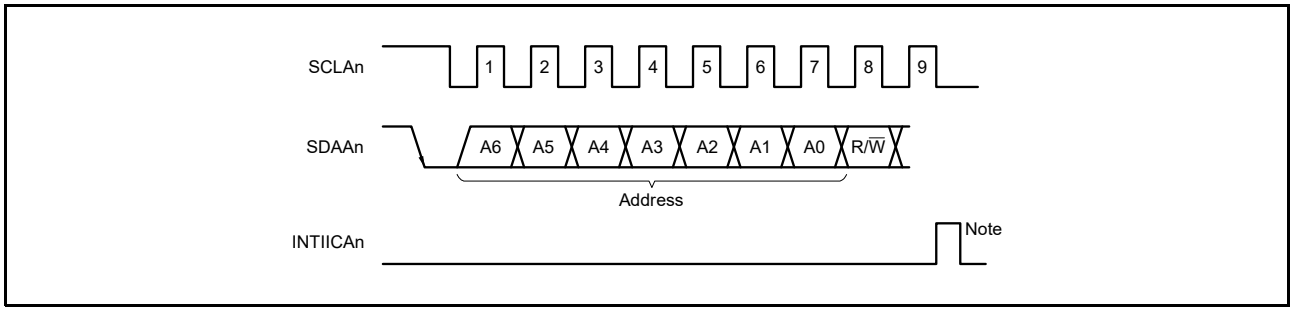
A start condition is output when the STTn bit of the IICA control register n0 (IICCTLn0) is set to 1 after a stop condition has been detected (IICSn.SPDn = 1). When a start condition is detected, the STDn bit of the IICSn register is set to 1.

Remark n = 0

12.5.2 Address

The address is defined by the 7 bits of data that follow the start condition. An address is a 7-bit data segment that is output in order to select one of the slave devices that are connected to the master device via the bus lines. Therefore, each slave device connected via the bus lines must have a unique address. The slave devices include hardware that detects the start condition and checks whether or not the 7-bit address data matches the data values stored in the slave address register n (SVAn). If the address data matches the SVAn register values, the slave device is selected and communicates with the master device until the master device generates a start condition or stop condition.

Figure 12 - 16 Addresses



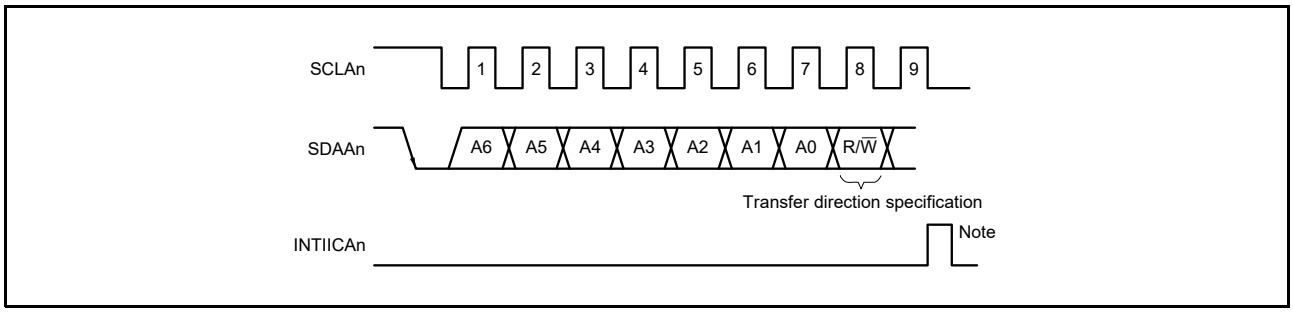
Note INTIICAn is not issued if data other than a local address or extension code is received while the all address match function is disabled during slave device operation.

Addresses are output when a total of 8 bits consisting of the slave address and the transfer direction described in **12.5.3 Transfer direction specification** are written to the IICA shift register n (IICAn). The received addresses are written to the IICAn register. The slave address is assigned to the higher 7 bits of the IICAn register.

12.5.3 Transfer direction specification

In addition to the 7-bit address data, the master device sends 1 bit that specifies the transfer direction. When this transfer direction specification bit has a value of 0, it indicates that the master device is transmitting data to a slave device. When the transfer direction specification bit has a value of 1, it indicates that the master device is receiving data from a slave device.

Figure 12 - 17 Transfer Direction Specification



Note INTIICAn is not issued if data other than a local address or extension code is received while the all address match function is disabled during slave device operation.

Remark n = 0

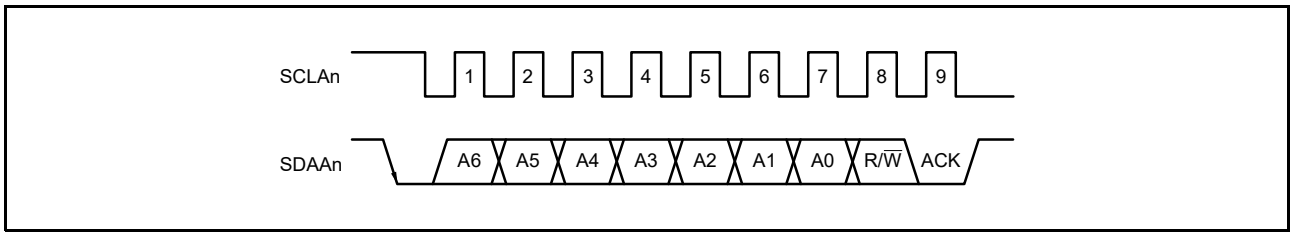
12.5.4 Acknowledge (ACK)

ACK is used to check the state of serial data at the transmission and reception sides. The reception side returns ACK each time it has received 8-bit data. The transmission side usually receives ACK after transmitting 8-bit data. When ACK is returned from the reception side, it is assumed that reception has been correctly performed and processing is continued. Whether ACK has been detected can be checked by using the ACKDn bit of the IICA status register n (IICSn). When the master receives the last data item, it does not return ACK and instead generates a stop condition. If a slave does not return ACK after receiving data, the master outputs a stop condition or restart condition and stops transmission. If ACK is not returned, the possible causes are as follows.

<1> Reception was not performed normally.
<2> The final data item was received.
<3> The reception side specified by the address does not exist.

To generate ACK, the reception side makes the SDAAn line low at the ninth clock (indicating normal reception). Automatic generation of ACK is enabled by setting the ACKEn bit of the IICA control register n0 (IICCTLn0) to 1. The TRCn bit of the IICSn register is set to the value of the eighth bit that follows 7-bit address information. Usually, set the ACKEn bit to 1 for reception (TRCn = 0). If a slave can receive no more data during reception (TRCn = 0) or does not require the next data item, then the slave must inform the master, by clearing the ACKEn bit to 0, that it will not receive any more data. When the master does not require the next data item during reception (TRCn = 0), it must clear the ACKEn bit to 0 so that ACK is not generated. In this way, the master informs a slave at the transmission side that it does not require any more data (transmission will be stopped).

Figure 12 - 18 ACK



When the local address is received, ACK is automatically generated, regardless of the value of the ACKEn bit. When an address other than that of the local address is received, ACK is not generated (NACK). When an extension code is received, or when an address is received while the all address match function is enabled, ACK is generated if the ACKEn bit is set to 1 in advance. How ACK is generated when data is received depends on the setting of the timing of clock stretching as follows.

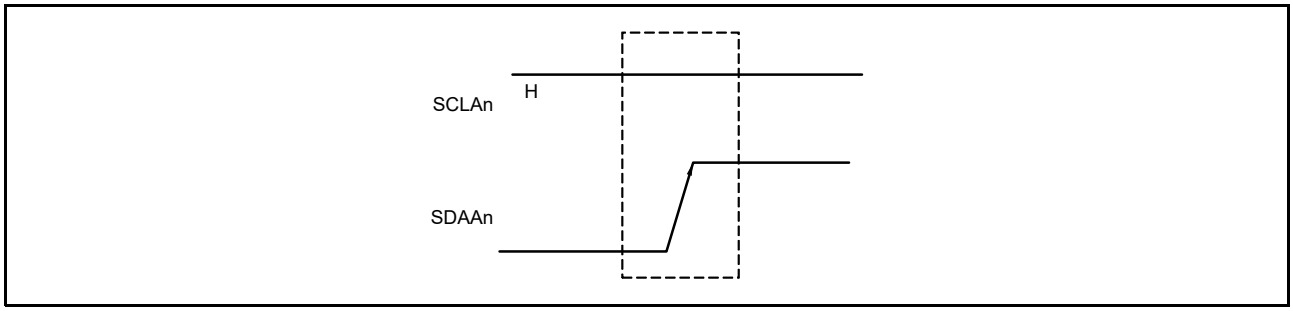
- When 8th cycle clock stretching is selected (IICCTLn0.WTIMn = 0):
  By setting the ACKEn bit to 1 before release from the clock stretch state, ACK is generated at the falling edge of the eighth clock cycle of the SCLAn pin.
- When 9th cycle clock stretching is selected (IICCTLn0.WTIMn = 1):
  ACK is generated if the ACKEn bit is set to 1 in advance.

Remark n = 0

12.5.5 Stop condition

When the SCLAn pin is at high level, changing the SDAAn pin from low level to high level generates a stop condition. A stop condition is a signal that the master device generates to the slave device when serial transfer has been completed. When the device is used as a slave, stop conditions can be detected.

Figure 12 - 19 Stop Condition



A stop condition is generated when the SPTn bit of the IICA control register n0 (IICCTLn0) is set to 1. When the stop condition is detected, the SPTn bit of the IICA status register n (IICSn) is set to 1 and INTIICAn is generated when the SPIEn bit of the IICCTLn0 register is set to 1.

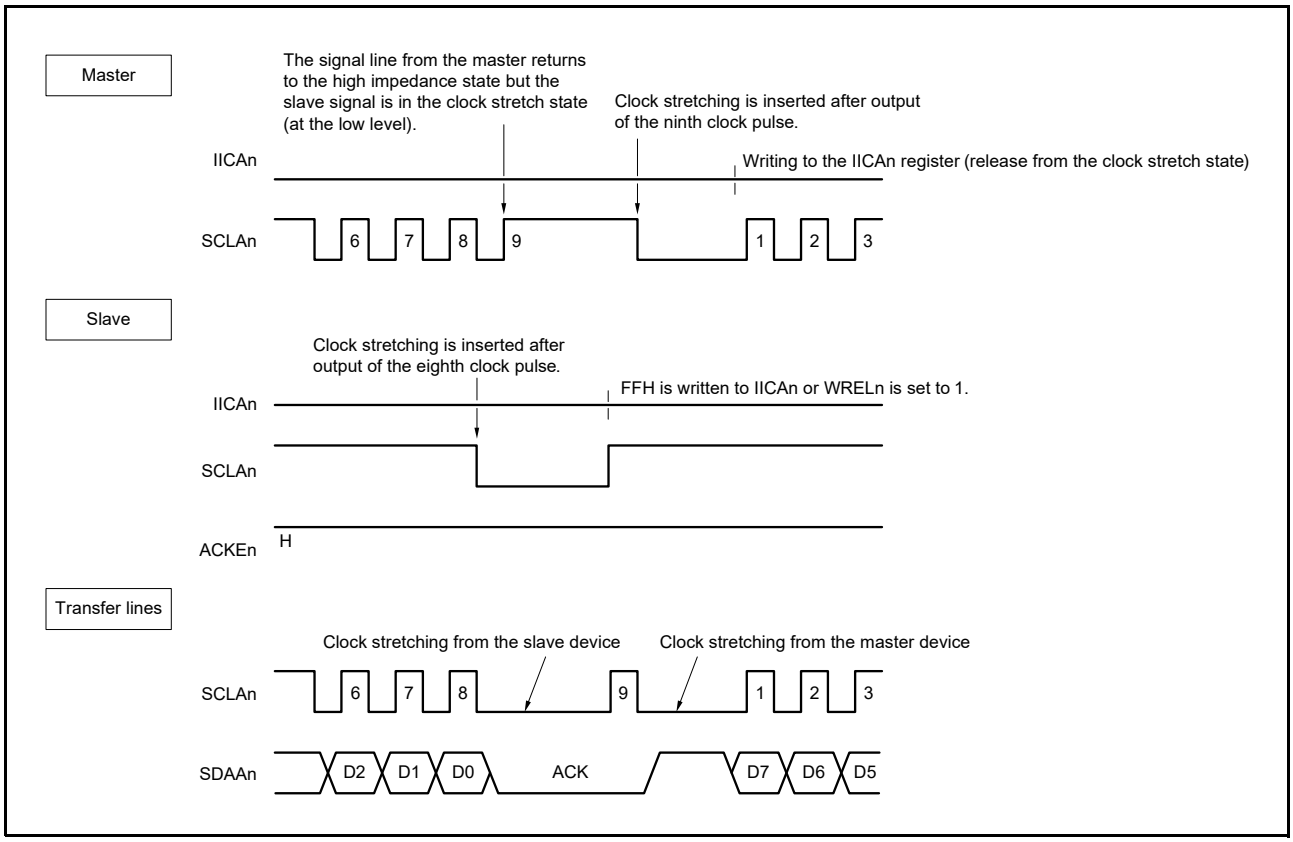
Remark n = 0

12.5.6 Clock stretching

Clock stretching is used to notify the other party in communications that a device (master or slave) is preparing to transmit or receive data (i.e., the interface is in the clock stretch state). Setting the SCLAn pin to the low level indicates the clock stretch state to the other party. When clock stretching is released for both the master and slave devices, the next data transfer can start.

Figure 12 - 20 Clock Stretching (1/2)

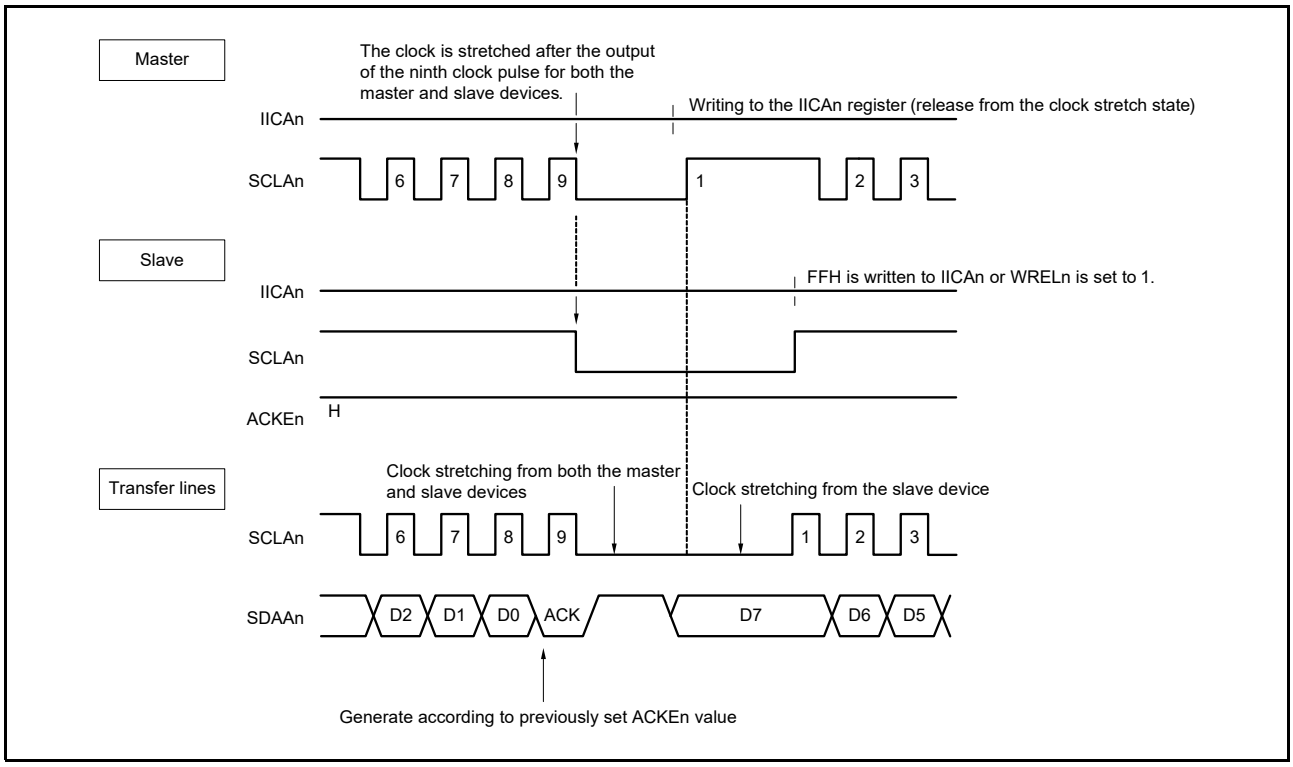
1. When clock stretching is set for the ninth and eighth clock cycles for the master and slave devices, respectively (master: transmission, slave: reception, and ACKEn = 1)



Remark n = 0

Figure 12 - 20 Clock Stretching (2/2)

2. When clock stretching is set for the ninth clock cycle for both the master and slave devices (master: transmission, slave: reception, and ACKEn = 1)



Remark ACKEn: Bit 2 of IICA control register n0 (IICCTLn0)
WRELn: Bit 5 of IICA control register n0 (IICCTLn0)

Clock stretching is automatically generated depending on the setting of the WTIMn bit of the IICA control register n0 (IICCTLn0). Normally, the receiving side releases the clock stretch state when IICCTLn0.WRELn is set to 1 or when FFH is written to the IICA shift register n (IICAn), and the transmitting side releases the clock stretch state when data is written to the IICAn register. The master device can also releases the clock stretch state via either of the following methods.

- By setting IICCTLn0.STTn to 1
- By setting IICCTLn0.SPTn to 1

Remark n = 0

12.5.7 Release from clock stretching

The I^2C interface usually releases the clock stretch state by the following processing.

- Writing data to the IICA shift register n (IICAn)
- Setting the WRELn bit of the IICA control register n0 (IICCTLn0) (release from the clock stretch state)
- Setting the STTn bit of the IICCTLn0 register (generating start condition)**Note**
- Setting the SPTn bit of the IICCTLn0 register (generating stop condition)**Note**

Executing the above processing for release from clock stretching leads to IICA releasing the clock stretch state after which communications are resumed. To release the clock stretch state and transmit data (including addresses), write the data to the IICAn register. To receive data after release from the clock stretch state, or to complete data transmission, set the WRELn bit of the IICCTLn0 register to 1. To generate a restart condition after release from the clock stretch state, set the STTn bit of the IICCTLn0 register to 1. To generate a stop condition after release from the clock stretch state, set the SPTn bit of the IICCTLn0 register to 1. Execute the processing for release only once for each period in the clock stretch state. If, for example, data is written to the IICAn register after release from the clock stretch state by setting the WRELn bit to 1, an incorrect value may be output to SDAAn line because the timing for changing the SDAAn line conflicts with the timing for writing the IICAn register.
In addition to the above, communications are stopped if the IICEn bit is cleared to 0 when communications have been aborted, so that the clock stretch state can be released. If the I^2C bus has deadlocked due to noise, the device can exit from communications by setting the LRELn bit of the IICCTLn0 register to 1, so that the clock stretch state can be released.

Note This setting is only required in the master device.

Caution **If the processing for release from clock stretching is executed when WUPn = 1, the clock stretch state will not be released.**

Remark n = 0

12.5.8 Timing of generation of the interrupt request signal (INTIICAn) and control of clock stretching

The setting of the WTIMn bit of the IICA control register n0 (IICCTLn0) determines the timing by which INTIICAn is generated and controls clock stretching, as shown in **Table 12 - 2**.

Table 12 - 2 INTIICAn Generation Timing and Control of Clock Stretching

WTIMn	During Slave Device Operation			During Master Device Operation		
	Address	Data Reception	Data Transmission	Address	Data Reception	Data Transmission
0	9**Notes 1, 2**	8**Note 2**	8**Note 2**	9	8	8
1	9**Notes 1, 2, 3**	9**Note 2**	9**Note 2**	9	9	9

Note 1. The slave device's INTIICAn signal and clock stretching occur at the falling edge of the ninth clock cycle only when there is a match with the address set to the slave address register n (SVAn). At this point, ACK is generated regardless of the value set to the ACKEn bit of the IICCTLn0 register. For a slave device that has received an extension code, or has received an address while the all address match function is enabled, INTIICAn occurs at the falling edge of the eighth clock. However, if the address does not match after restart, INTIICAn is generated at the falling edge of the ninth clock cycle, but clock stretching does not occur.

Note 2. If the received address does not match the contents of the slave address register n (SVAn), the all address match function is disabled, and extension code is not received, neither INTIICAn nor clock stretching occurs.

Note 3. When the setting of the WTIMn bit is 1, for a slave device that has received an extension code, or has received an address while the all address match function is enabled, INTIICAn and clock stretching occur at the falling edge of the eighth clock as well as on that of the ninth clock.

Remark The numbers in the table indicate the pulses of the serial clock signal. Interrupt requests and control of clock stretching are both synchronized with the falling edge of these clock pulses.

1. During address transmission/reception
   - Slave device operation: The timing of the interrupt and clock stretching depends on the conditions described in **Notes 1** to **3** above.
   - Master device operation: The interrupt and clock stretching occur at the falling edge of the ninth clock cycle, regardless of the setting of the WTIMn bit.
2. During data reception
   - Master/slave device operation: The timing of the interrupt and clock stretching depends on the setting of the WTIMn bit.
3. During data transmission
   - Master/slave device operation: The timing of the interrupt and clock stretching depends on the setting of the WTIMn bit.
4. Release from clock stretching

   The four types of processing for release from clock stretching are as follows.
   - Writing data to the IICA shift register n (IICAn)
   - Setting the WRELn bit of the IICA control register n0 (IICCTLn0) (release from the clock stretch state)
   - Setting the STTn bit of the IICCTLn0 register (generating start condition)**Note**
   - Setting the SPTn bit of the IICCTLn0 register (generating stop condition)**Note**

   When 8th cycle clock stretching has been selected (WTIMn = 0), the presence/absence of ACK generation must be determined before release from the clock stretch state.
5. Detection of stop condition

   INTIICAn is generated when a stop condition is detected (only when SPIEn = 1).

Note This setting is only required in the master device.

Remark n = 0

12.5.9 Address match detection method

In I2C bus mode, the master device can select a particular slave device by transmitting the corresponding slave address. Address match can be detected automatically by hardware. An interrupt request signal (INTIICAn) occurs only when the address set to the slave address register n (SVAn) matches the slave address sent by the master device, when an address is received while the all address match function is enabled (IICCTLn1.SVADISn = 1), or when an extension code has been received.

12.5.10 Error detection

In I2C bus mode, the state of the serial data bus (SDAAn) during data transmission is captured by the IICA shift register n (IICAn) of the transmitting device, so the IICA data prior to transmission can be compared with the transmitted IICA data to enable detection of transmission errors. A transmission error is judged as having occurred when the compared data values do not match.

Remark n = 0

12.5.11 Extension code

1. When the higher 4 bits of the receive address are either 0000 or 1111, the detection of extension code reception flag (EXCn) is set to 1 for extension code reception and an interrupt request signal (INTIICAn) is issued at the falling edge of the eighth clock. When an address is received while the all address match function is enabled, it is also determined that an extension code has been received. The local address stored in the slave address register n (SVAn) is not affected.
2. The settings below are specified if 11110xx0 is transferred from the master by using a 10-bit address transfer while the SVAn register is set to 11110xx0 or if an address is received while the all address match function is enabled. Note that INTIICAn occurs at the falling edge of the eighth clock.
   - Higher four bits of data match or the all address match function is enabled: IICSn.EXCn = 1
   - Seven bits of data match or the all address match function is enabled: IICSn.COIn = 1
3. Since the processing after the interrupt request occurs differs according to the data that follows the extension code, such processing is performed by software. If the extension code is received or an address is received with the all address match function enabled during operation as a slave, then the slave device is participating in communication even if its address does not match. For example, after the extension code is received, if you do not wish to operate the target device as a slave device, set IICCTLn0.LRELn to 1 to set the standby mode for the next communication operation.

Table 12 - 3 Bit Definitions of Major Extension Codes

Slave Address	R/W Bit	Description
0 0 0 0 0 0 0	0	General call address
1 1 1 1 0 x x	0	10-bit slave address specification (during address authentication)
1 1 1 1 0 x x	1	10-bit slave address specification (after address match, when read command is issued)

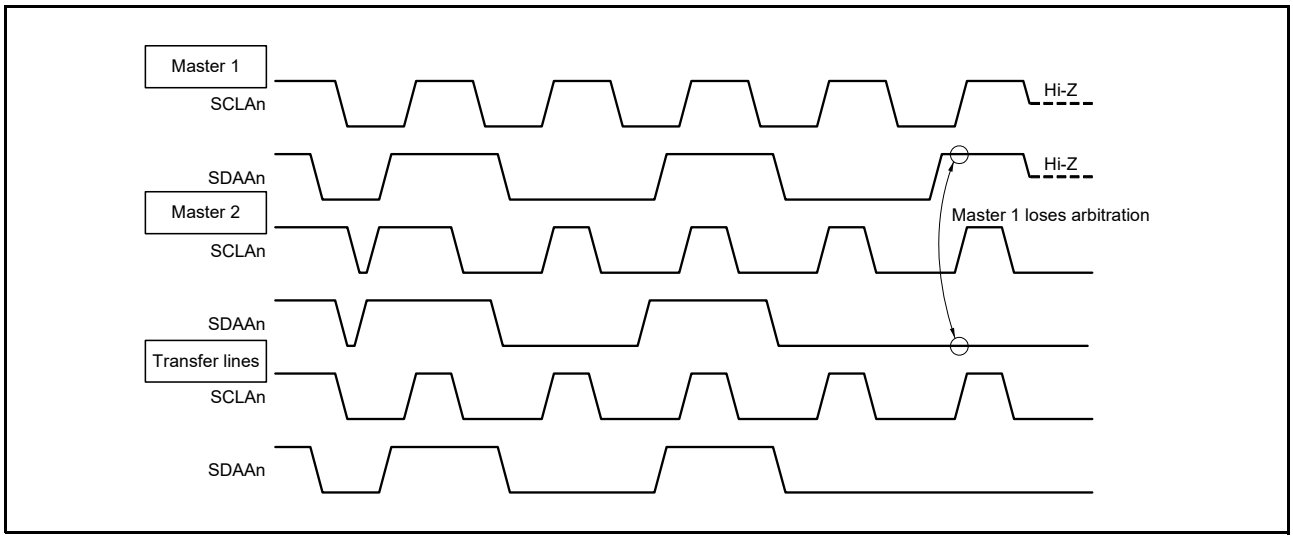
Remark 1. See the I^2C bus specifications issued by NXP Semiconductors for details of extension codes other than those described above.

Remark 2. n = 0

12.5.12 Arbitration

When several master devices simultaneously generate a start condition (when IICCTLn0.STTn is set to 1 before IICSn.STDn is set to 1), communication among the master devices is performed as the number of clocks are adjusted until the data differs. This kind of operation is called arbitration. When one of the master devices loses in arbitration, an arbitration loss flag (ALDn) in the IICA status register n (IICSn) is set to 1 via the timing by which the arbitration loss occurred, and the SCLAn and SDAAn lines are both set to high impedance, which releases the bus. The arbitration loss is detected by checking ALDn = 1 by software at the timing of the next interrupt request (the eighth or ninth clock cycle, when a stop condition is detected, etc.). For details of interrupt request timing, see **12.5.8 Timing of generation of the interrupt request signal (INTIICAn) and control of clock stretching**.

Figure 12 - 21 Arbitration Timing Example



Remark n = 0

Table 12 - 4 State during Arbitration and Interrupt Request Generation Timing

State during Arbitration	Interrupt Request Generation Timing
During address transmission	At falling edge of eighth or ninth clock following byte transfer[Note 1]
Read/write data after address transmission	
During extension code transmission	
Read/write data after extension code transmission	
During data transmission	
During ACK transfer period after data transmission	
When restart condition is detected during data transfer	
When stop condition is detected during data transfer	When stop condition is generated (when SPIEn = 1)[Note 2]
When data is at low level while attempting to generate a restart condition	At falling edge of eighth or ninth clock following byte transfer[Note 1]
When stop condition is detected while attempting to generate a restart condition	When stop condition is generated (when SPIEn = 1)[Note 2]
When data is at low level while attempting to generate a stop condition	At falling edge of eighth or ninth clock following byte transfer[Note 1]
When SCLAn is at low level while attempting to generate a restart condition	

Note 1. When IICCTLn0.WTIMn = 1, an interrupt request occurs at the falling edge of the ninth clock. When WTIMn = 0, the extension code's slave address is received, and an address is received while the all address match function is enabled, an interrupt request occurs at the falling edge of the eighth clock.

Note 2. When there is a chance that arbitration will occur, set SPIEn = 1 for master device operation.

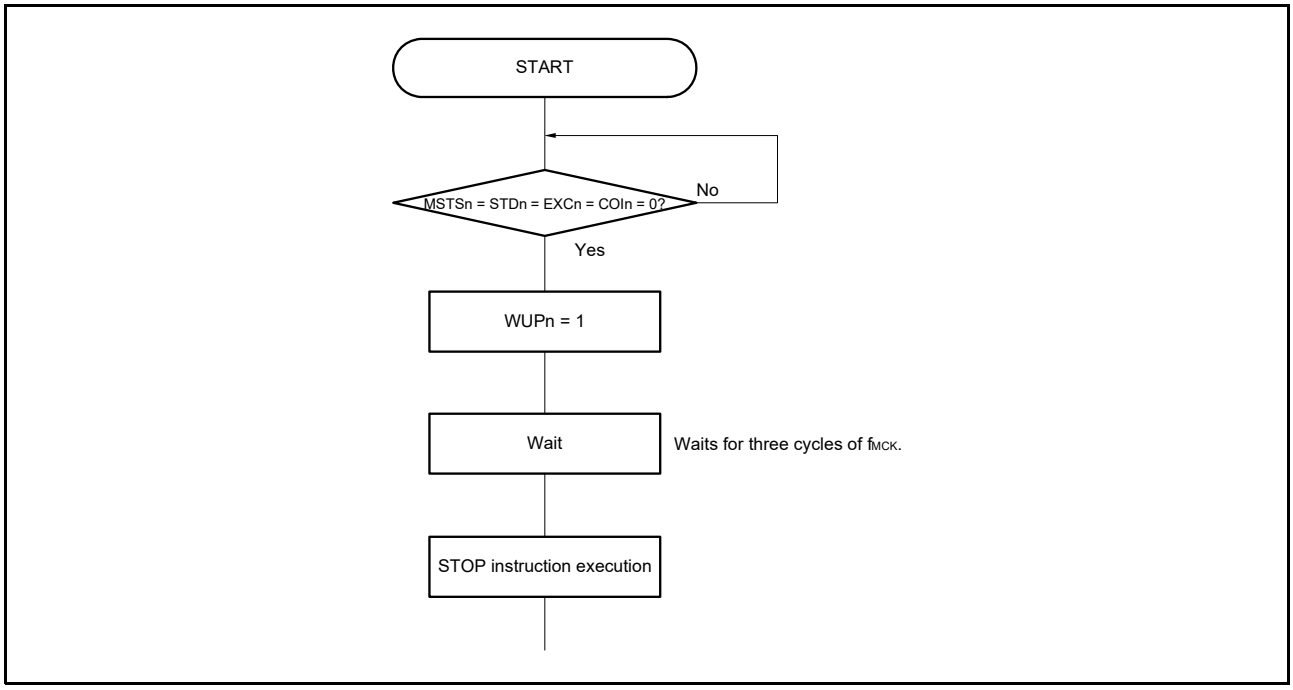
Remark 1. SPIEn: Bit 4 of IICA control register n0 (IICCTLn0)

Remark 2. n = 0

12.5.13 Wakeup function

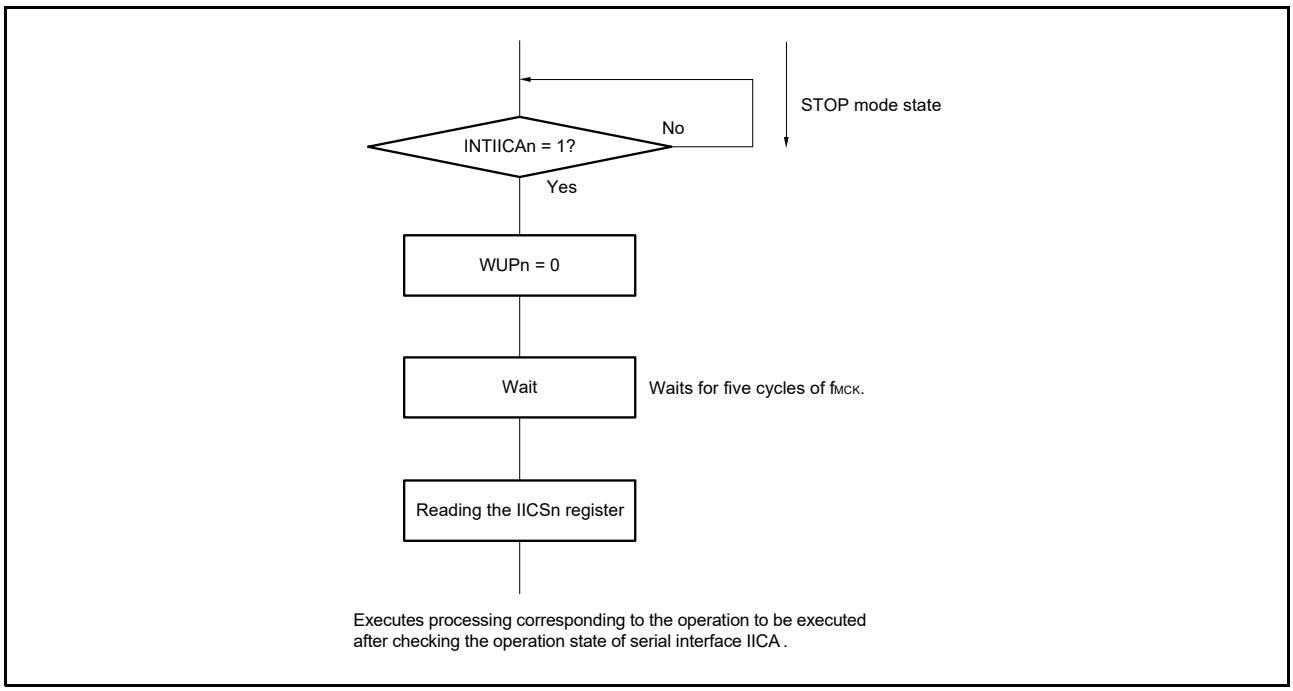
The I^2C bus slave function is a function that generates an interrupt request signal (INTIICAn) when the local address is received, an address is received while the all address match function is enabled, or an extension code is received. This function makes processing more efficient by preventing unnecessary INTIICAn signal from occurring when addresses do not match while the all address match function is disabled. When a start condition is detected, wakeup standby mode is set. Even a master that has generated a start condition enters the wakeup standby state while transmitting an address because the master may become a slave due to an arbitration loss. To use the wakeup function in the STOP mode, set the WUPn bit to 1. Addresses can be received regardless of the operation clock. An interrupt request signal (INTIICAn) is also generated when the local address is received, an address is received while the all address match function is enabled, or an extension code is received. Operation returns to normal operation by using an instruction to clear the WUPn bit to 0 after this interrupt has been generated. **Figure 12 - 22** shows the flow for setting WUPn = 1 and **Figure 12 - 23** shows the flow for setting WUPn = 0 upon an address match (or when the all address match function is enabled) including extension code reception.

Figure 12 - 22 Flow for Setting WUPn = 1



Remark n = 0

Figure 12 - 23 Flow When Setting WUPn = 0 upon Address Match (or When the All Address Match Function is Enabled) (Including Extension Code Reception)



Use the following flows to perform the processing to release the STOP mode other than by an interrupt request signal (INTIICAn) generated from serial interface IICA.

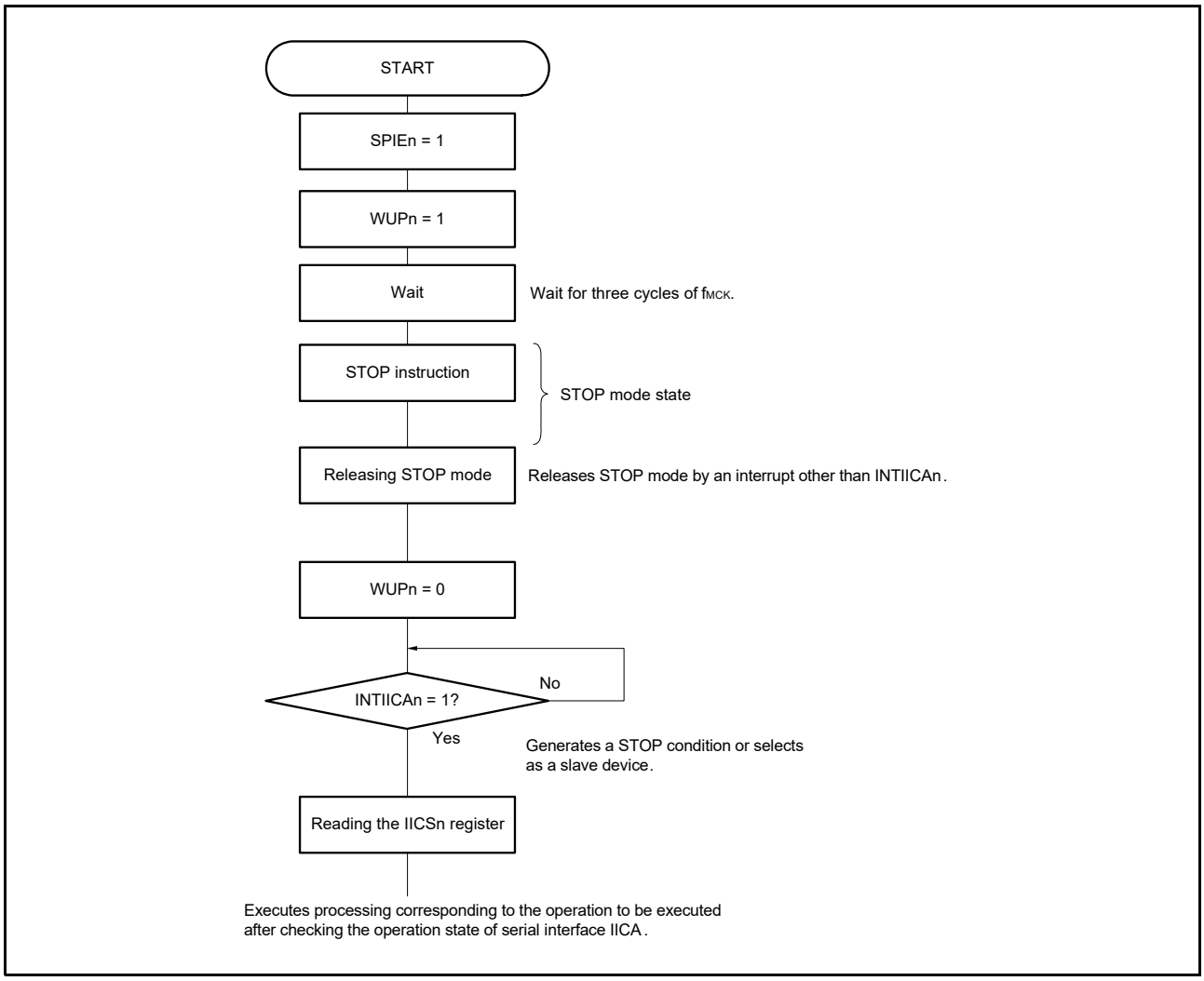
- When operating next IIC communication as master: Flow shown in **Figure 12 - 24**.
- When operating next IIC communication as slave:

  When released by INTIICAn interrupt: Same as the flow in **Figure 12 - 23**.

  When released by other than INTIICAn interrupt: Wait for INTIICAn interrupt with WUPn left set to 1.

Remark n = 0

Figure 12 - 24 When Operating as Master Device after Releasing STOP Mode other than by INTIICAn



Remark n = 0

12.5.14 Communication reservation

1. When communication reservation function is enabled (IICFn.IICRSVn = 0)
   To start master device communications when not currently using a bus, a communication reservation can be made to enable transmission of a start condition when the bus is released. There are two modes under which the bus is not used.

   - When arbitration results in neither master nor slave operation
   - While the all address match function is disabled, when an extension code is received and slave operation is disabled (ACK is not returned and the bus was released by setting IICCTLn0.LRELn to 1 and exiting from communication)

   If the STTn bit of the IICCTLn0 register is set to 1 while the bus is not used, a start condition is automatically generated and wait state is entered after the bus is released (after a stop condition is detected). If an address is written to the IICA shift register n (IICAn) after the SPIEn bit of the IICCTLn0 register was set to 1, and it was detected by generation of an interrupt request signal (INTIICAn) that the bus was released (detection of the stop condition), then the device automatically starts communication as the master. Data written to the IICAn register before the stop condition is detected is invalid. When the STTn bit has been set to 1, the operation mode (as start condition or as communication reservation) is determined according to the state of the bus.

   - If the bus has been released: A start condition is generated
   - If the bus has not been released (standby mode): Communication reservation

   Check whether the communication reservation operates or not by using the MSTSn flag (bit 7 of the IICA status register n (IICSn)) after the STTn bit is set to 1 and the wait time elapses. Use software to secure the wait time calculated by the following expression.

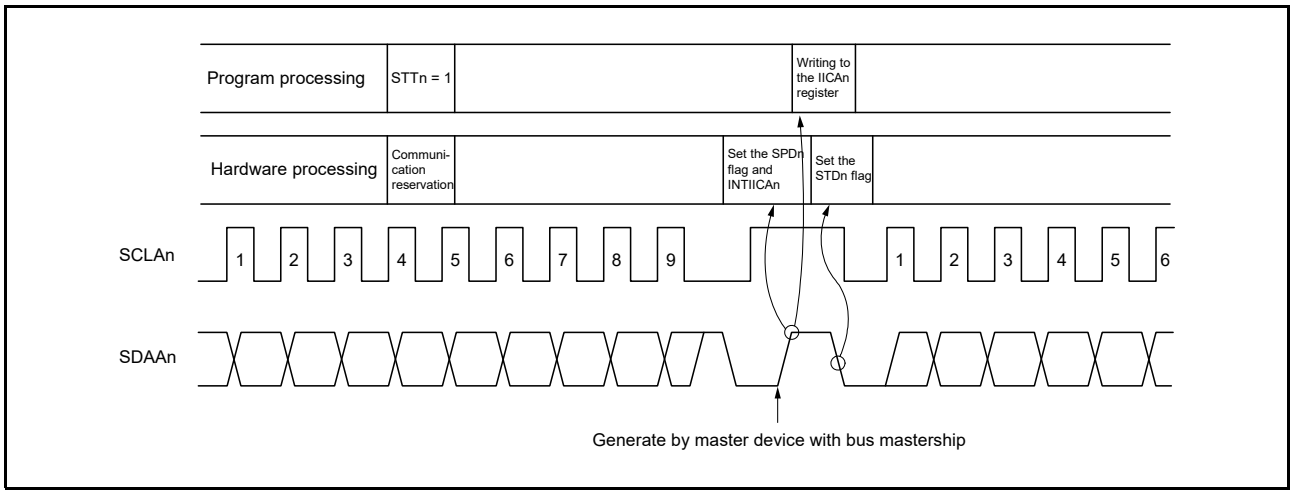
Wait time from setting STTn = 1 to checking the MSTSn flag:
(IICWLn setting value + IICWHn setting value + 4)/f_{MCK} + t_F × 2

   Remark 1. IICWLn: IICA low-level width setting register n
   IICWHn: IICA high-level width setting register n
   t_F: SDAAn and SCLAn signal falling times
   f_{MCK}: IICA operation clock frequency

   Remark 2. n = 0

Figure 12 - 25 shows the communication reservation timing.

Figure 12 - 25 Communication Reservation Timing

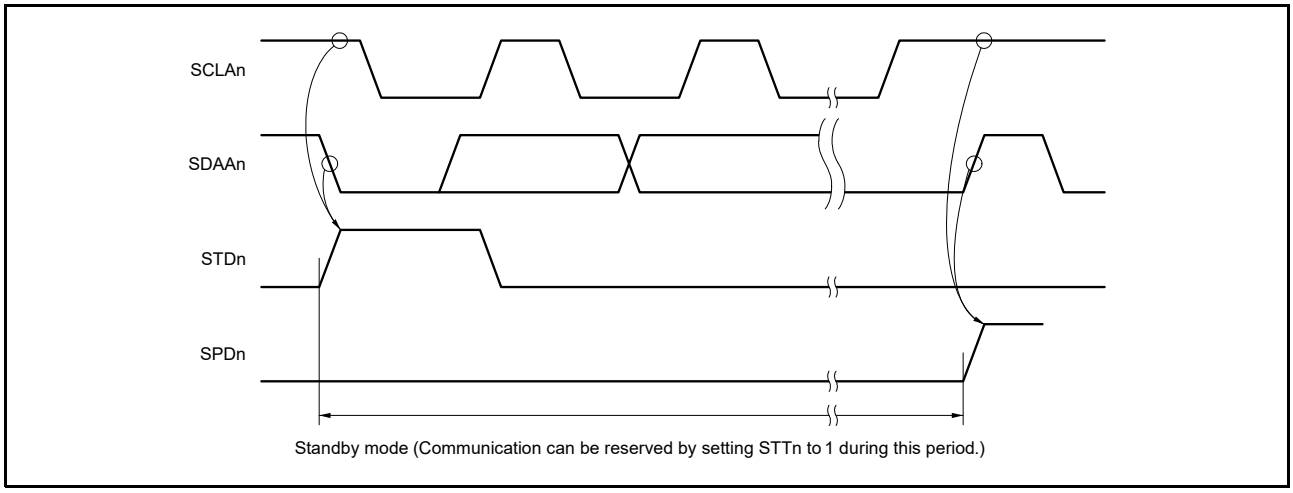


Remark 1. IICAn: IICA shift register n
STTn: Bit 1 of IICA control register n0 (IICCTLn0)
STDn: Bit 1 of IICA status register n (IICSn)
SPDn: Bit 0 of IICA status register n (IICSn)

Remark 2. n = 0

Communication reservations are accepted via the timing shown in **Figure 12 - 26**. After IICSn.STDn is set to 1, a communication reservation can be made by setting IICCTLn0.STTn to 1 before a stop condition is detected.

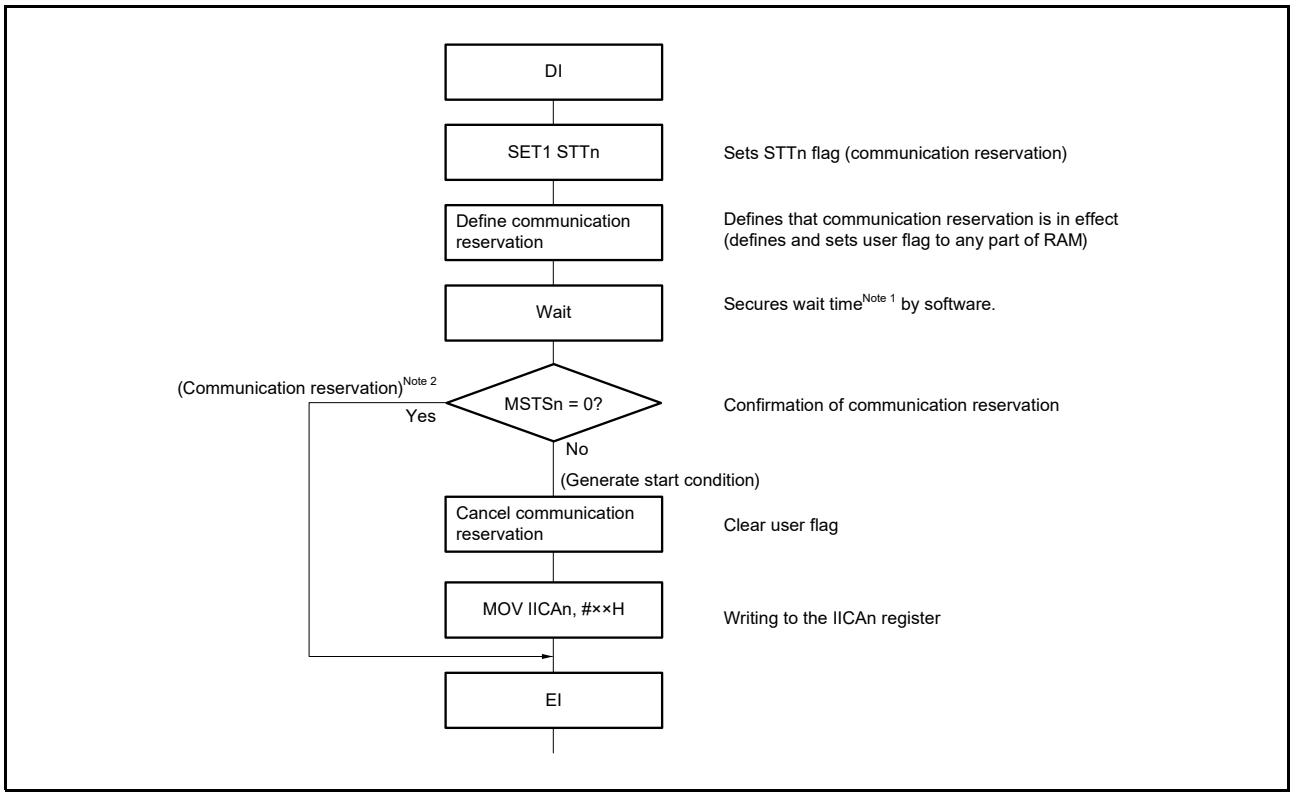
Figure 12 - 26 Timing for Accepting Communication Reservations



Remark n = 0

Figure 12 - 27 shows the communication reservation protocol.

Figure 12 - 27 Communication Reservation Protocol



Note 1. The wait time is calculated as follows.
(IICWLn setting value + IICWHn setting value + 4)/f_{MCK} + t_F × 2

Note 2. The communication reservation operation executes a write to the IICA shift register n (IICAn) when a stop condition interrupt request occurs.

Remark 1. STTn: Bit 1 of IICA control register n0 (IICCTLn0)
MSTSn: Bit 7 of IICA status register n (IICSn)
IICAn: IICA shift register n
IICWLn: IICA low-level width setting register n
IICWHn: IICA high-level width setting register n
t_F: SDAAn and SCLAn signal falling times
f_{MCK}: IICA operation clock frequency

Remark 2. n = 0

2. When communication reservation function is disabled (IICFn.IICRSVn = 1)
   When the STTn bit of the IICA control register n0 (IICCTLn0) is set to 1 when the bus is not used in a communication during bus communication, this request is rejected and a start condition is not generated. The states where the bus is not in use consist of the following two states.

   - When arbitration results in neither master nor slave operation
   - While the all address match function is disabled, when an extension code is received and slave operation is disabled (ACK is not returned and the bus was released by setting IICCTLn0.LRELn to 1 and exiting from communication)

   To confirm whether the start condition was generated or request was rejected, check the STCFn flag of the IICFn register. It takes up to 5 cycles of f_{MCK} until the STCFn flag is set to 1 after setting STTn = 1. Therefore, secure the time by software.

   Remark n = 0

12.5.15 Cautions

1. When STCENn = 0
   Immediately after I^2C operation is enabled (IICEn = 1), the bus is recognized as being in a communications state (IICBSYn = 1) regardless of its actual state. When changing from a mode in which no stop condition has been detected to a master device communication mode, first generate a stop condition to release the bus, then perform master device communication. When using multiple masters, it is not possible to perform master device communication when the bus has not been released (when a stop condition has not been detected). Use the following sequence for generating a stop condition.
   <1> Set IICA control register n1 (IICCTLn1).
   <2> Set the IICEn bit of the IICA control register n0 (IICCTLn0) to 1.
   <3> Set the SPTn bit of the IICCTLn0 register to 1.

2. When STCENn = 1
   Immediately after I^2C operation is enabled (IICEn = 1), the bus is recognized as being in the released state (IICBSYn = 0) regardless of its actual state. To generate the first start condition (STTn = 1), it is necessary to confirm that the bus has been released, so as to not disturb other communications.

3. If other I^2C communications are already in progress
   If I^2C operation is enabled and the device participates in communication already in progress when the SDAAn pin is low and the SCLAn pin is high, the IICA recognizes that the SDAAn pin has gone low (detects a start condition). If the value on the bus at this time can be recognized as an extension code or the all address match function is enabled, ACK is returned, but this interferes with other I^2C communications. To avoid this, start the IICA in the following sequence.
   <1> Clear the SPIEn bit of the IICCTLn0 register to 0 to disable generation of an interrupt request signal (INTIICAn) when the stop condition is detected.
   <2> Set the IICEn bit of the IICCTLn0 register to 1 to enable the operation of the IICA.
   <3> Wait for detection of the start condition.
   <4> Set the LRELn bit of the IICCTLn0 register to 1 before ACK is returned (4 to 72 cycles of f_{MCK} after setting the IICEn bit to 1), to forcibly disable detection.

4. Setting the STTn and SPTn bits (bits 1 and 0 of the IICCTLn0 register) again after they are set and before they are cleared to 0 is prohibited.

5. When transmission is reserved, set the SPIEn bit (bit 4 of the IICCTLn0 register) to 1 so that an interrupt request is generated when the stop condition is detected. Transfer is started when communication data is written to the IICA shift register n (IICAn) after the interrupt request is generated. Unless the interrupt is generated when the stop condition is detected, the device stops in the wait state because the interrupt request is not generated when communication is started. However, it is not necessary to set the SPIEn bit to 1 when the MSTSn flag of the IICA status register n (IICSn) is detected by software.

Remark n = 0

12.5.16 Communication operations

The following shows three operation procedures with the flowchart.

1. Master operation in single-master system
   The flowchart when using a product of this group as the master in a single master system is shown below. This flowchart is broadly divided into the initial settings and communication processing. Execute the initial settings at startup. If communication with the slave is required, prepare the communication and then execute communication processing.

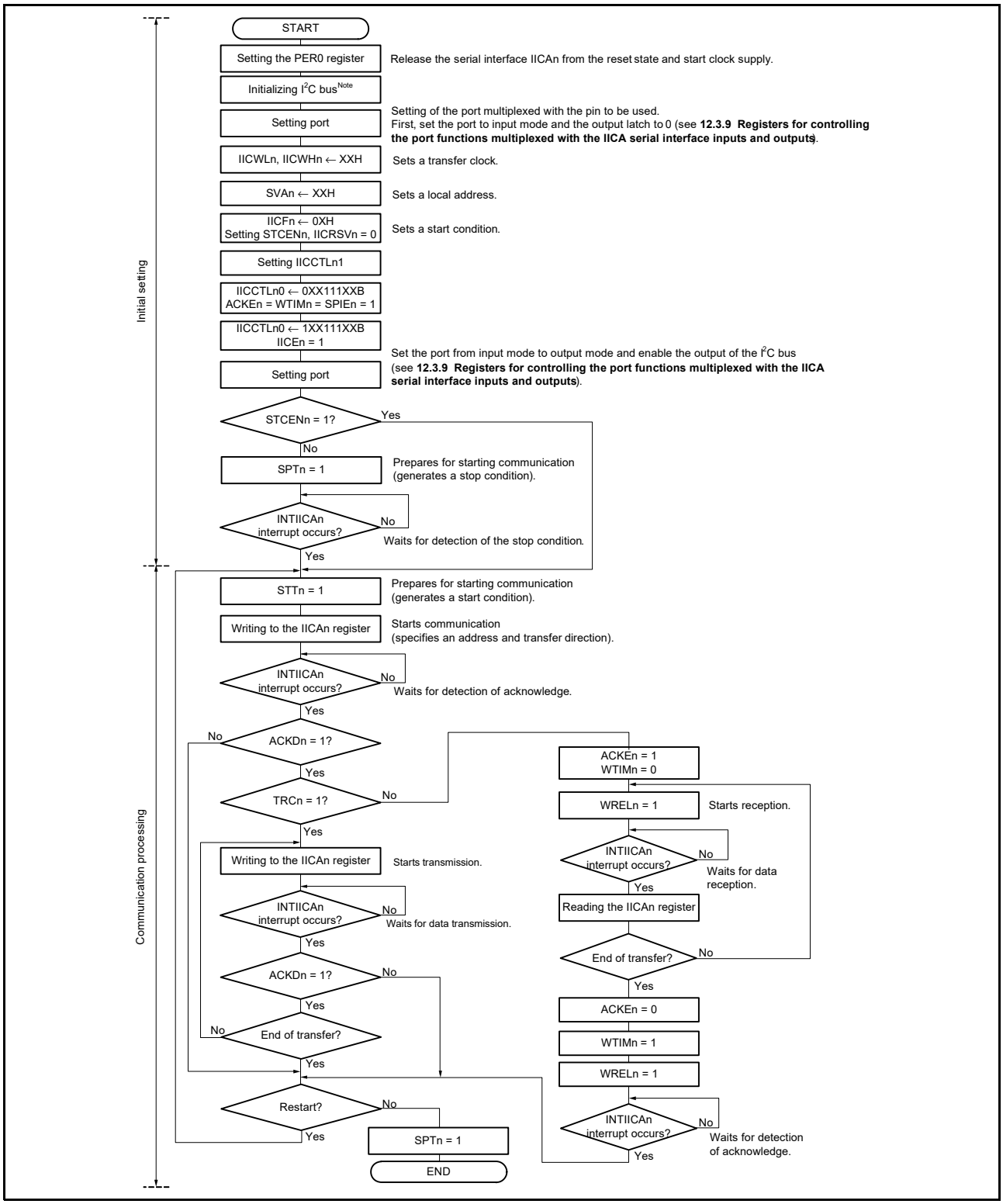
2. Master operation in multi-master system
   In the I^2C bus multi-master system, whether the bus is released or used cannot be judged by the I^2C bus specifications when a device takes part in a communication. Here, when data and clock are at a high level for a certain period (1 frame), a product of this group takes part in a communication with bus released state. This flowchart is broadly divided into the initial settings, communication waiting, and communication processing. The processing when the product loses in arbitration and is specified as the slave is omitted here, and only the processing as the master is shown. Execute the initial settings at startup to take part in a communication. Then, wait for the communication request as the master or wait for the specification as the slave. The actual communication is performed in the communication processing, and it supports the transmission/reception with the slave and the arbitration with other masters.

3. Slave operation
   An example of when a product of this group is used as the I^2C bus slave is shown below.
   When used as the slave, operation is started by an interrupt. Execute the initial settings at startup, then wait for the INTIICAn interrupt occurrence (communication waiting). When an INTIICAn interrupt occurs, the communications state is judged and the result is passed as a flag over to the main processing. By checking the flags, necessary communication processing is performed.

Remark n = 0

1. Master operation in single-master system

Figure 12 - 28 Master Operation in Single-master System



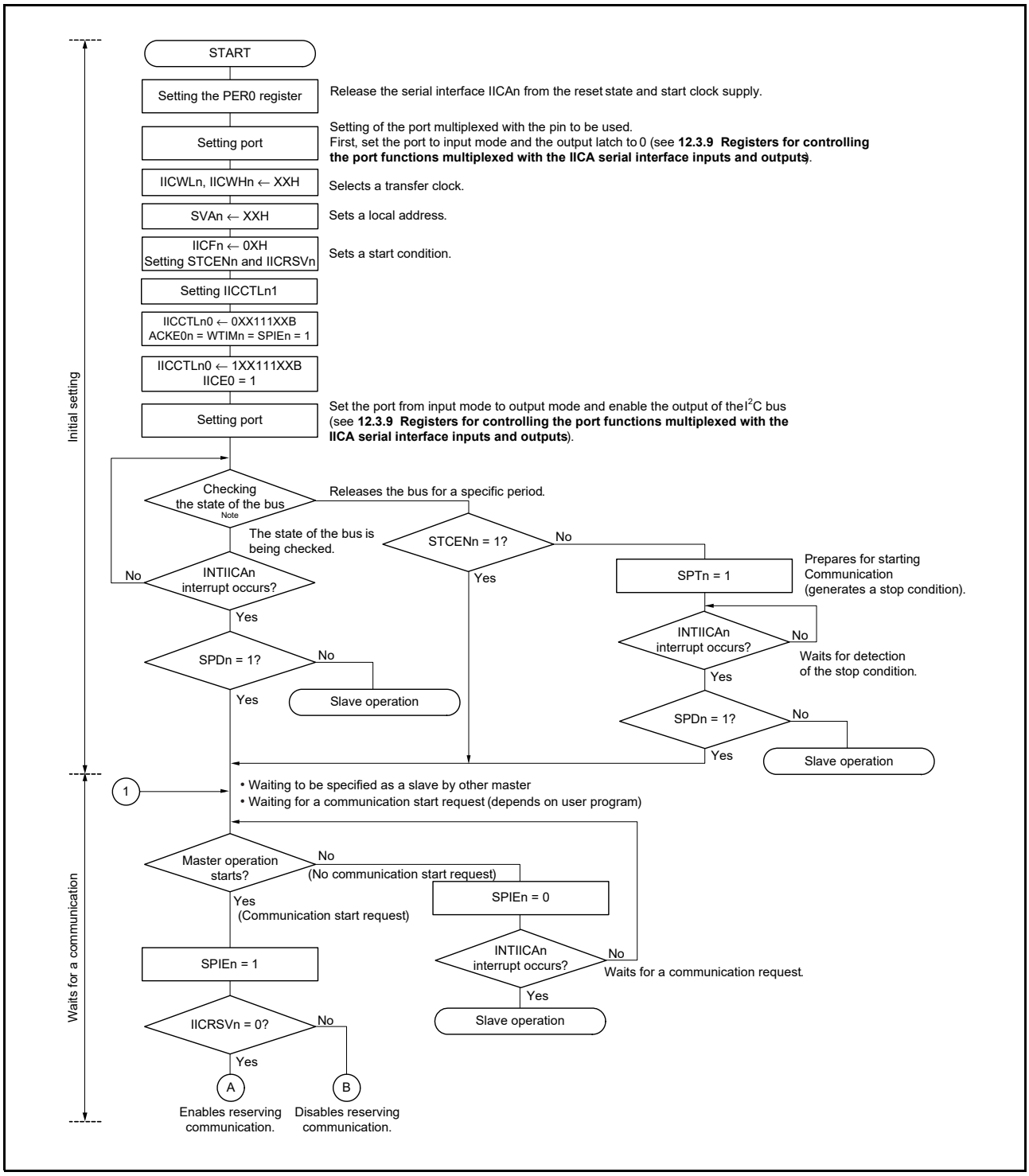
Note Release (SCLAn and SDAAn pins = high level) the I2C bus in conformance with the specifications of the product that is communicating. If EEPROM is outputting a low level to the SDAAn pin, for example, set the SCLAn pin in the output port mode, and output a clock pulse from the output port until the SDAAn pin is constantly at high level.

Remark 1. Conform to the specifications of the product that is communicating, with respect to the transmission and reception formats.

Remark 2. n = 0

2. Master operation in multi-master system

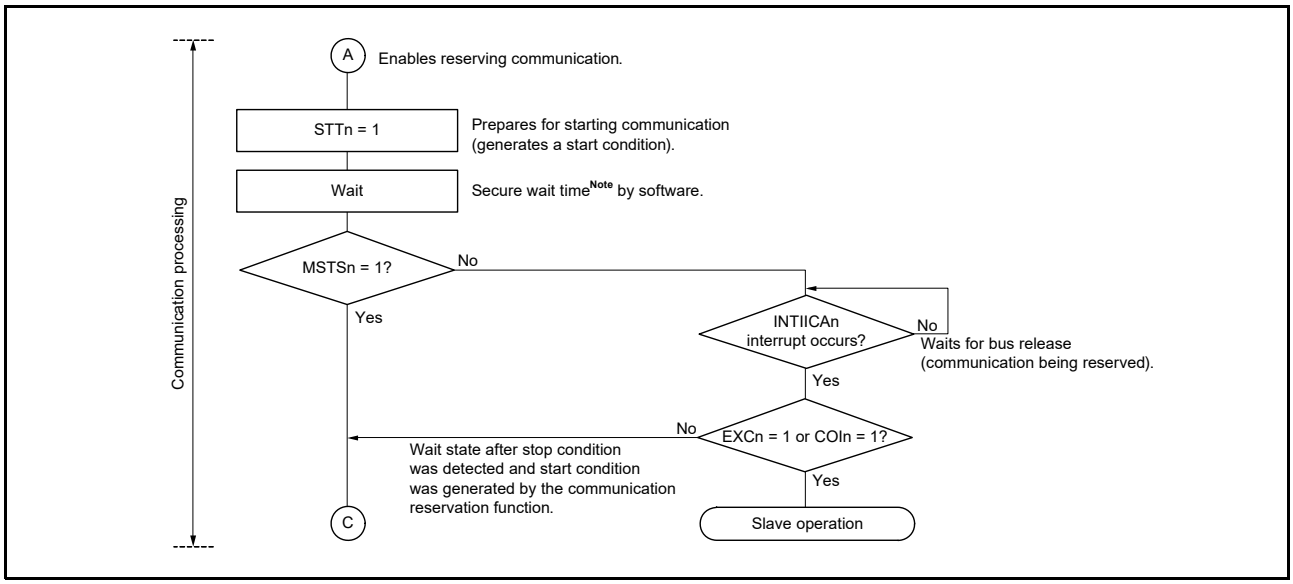
Figure 12 - 29 Master Operation in Multi-master System (1/3)



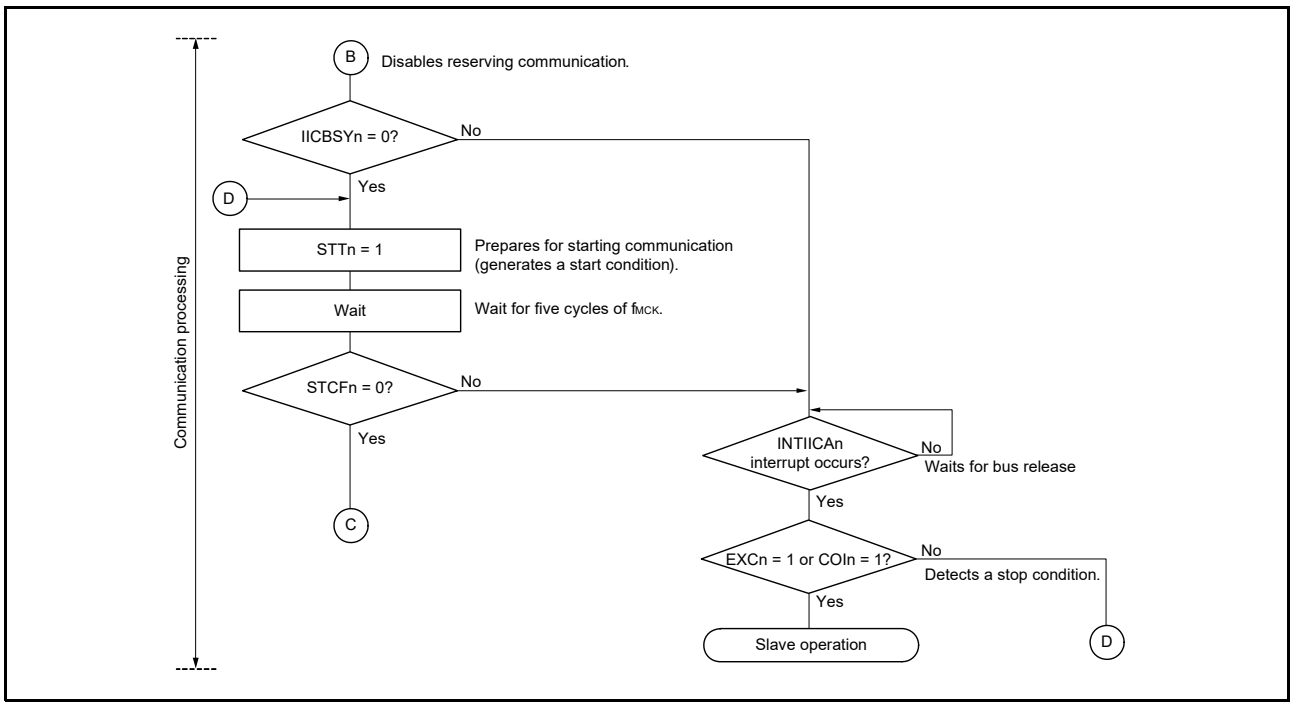
Note Confirm that the bus is released (CLDn = 1, DADn = 1) for a specific period (for example, for a period of one frame). If the SDAAn pin is constantly at low level, decide whether to release the I^2C bus (SCLAn and SDAAn pins = high level) in conformance with the specifications of the product that is communicating.

Remark n = 0

Figure 12 - 29 Master Operation in Multi-master System (2/3)



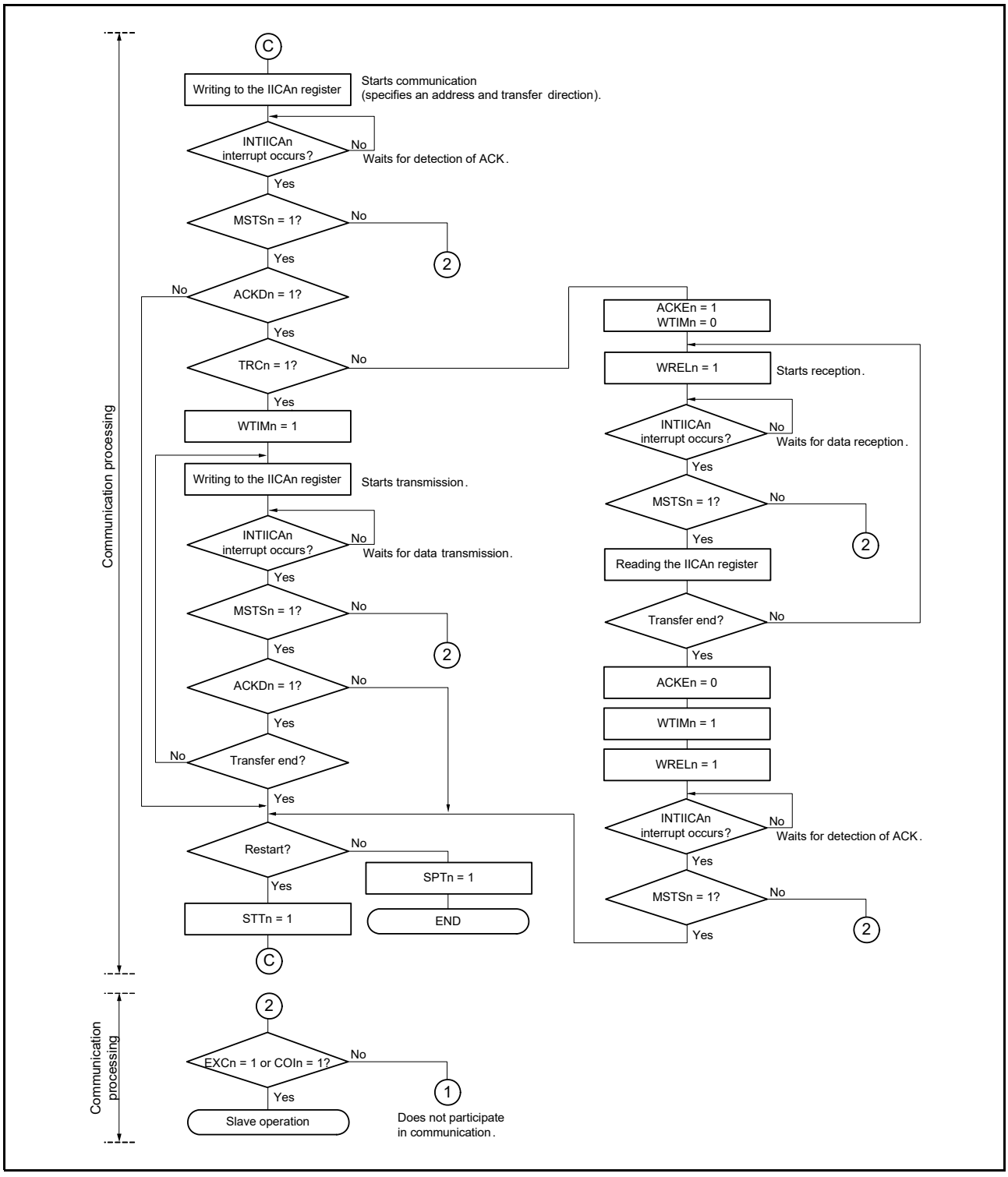
Note The wait time is calculated as follows.
(IICWLn setting value + IICWHn setting value + 4)/f_{MCK} + t_F × 2



Remark 1. IICWLn: IICA low-level width setting register n
IICWHn: IICA high-level width setting register n
t_F: SDAAn and SCLAn signal falling times
f_{MCK}: IICA operation clock frequency

Remark 2. n = 0

Figure 12 - 29 Master Operation in Multi-master System (3/3)



Remark 1. Conform to the specifications of the product that is communicating, with respect to the transmission and reception formats.

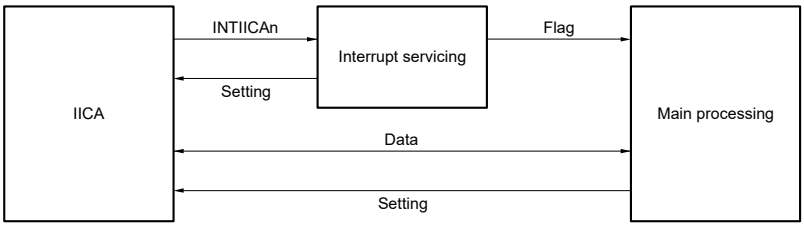
Remark 2. To use the device as a master in a multi-master system, read the MSTSn flag each time interrupt INTIICAn has occurred to check the arbitration result.

Remark 3. To use the device as a slave in a multi-master system, check the state by using the IICA status register n (IICSn) and IICA flag register n (IICFn) each time interrupt INTIICAn has occurred, and determine the processing to be performed next.

Remark 4. n = 0

3. Slave operation

The processing procedure of the slave operation is as follows. Basically, the slave operation is event-driven. Therefore, processing by the INTIICAn interrupt (processing that must substantially change the state of operating such as detection of a stop condition during communication) is necessary. In the following explanation, it is assumed that the all address match function is disabled and the extension code is not supported for data communication. It is also assumed that the INTIICAn interrupt servicing routine only handles state transition processing, and that the main processing handles actual data communications.



Therefore, data communication processing is performed by preparing the following three flags and passing them to the main processing instead of INTIICAn.

<1> Communication mode flag

This flag indicates the following two communications states.

- Clear mode: State in which data communications are not in progress
- Communication mode: State in which data communications are in progress (from valid address detection to stop condition detection, no detection of ACK from master, address mismatch)

<2> Ready flag

This flag indicates that data communication is enabled. Its function is the same as the INTIICAn interrupt for ordinary data communication. This flag is set by interrupt servicing and cleared by the main processing. Clear this flag by interrupt servicing when communication is started. However, the ready flag is not set by interrupt servicing when the first data is transmitted. Therefore, the first data is transmitted without the flag being cleared (an address match is interpreted as a request for the next data).

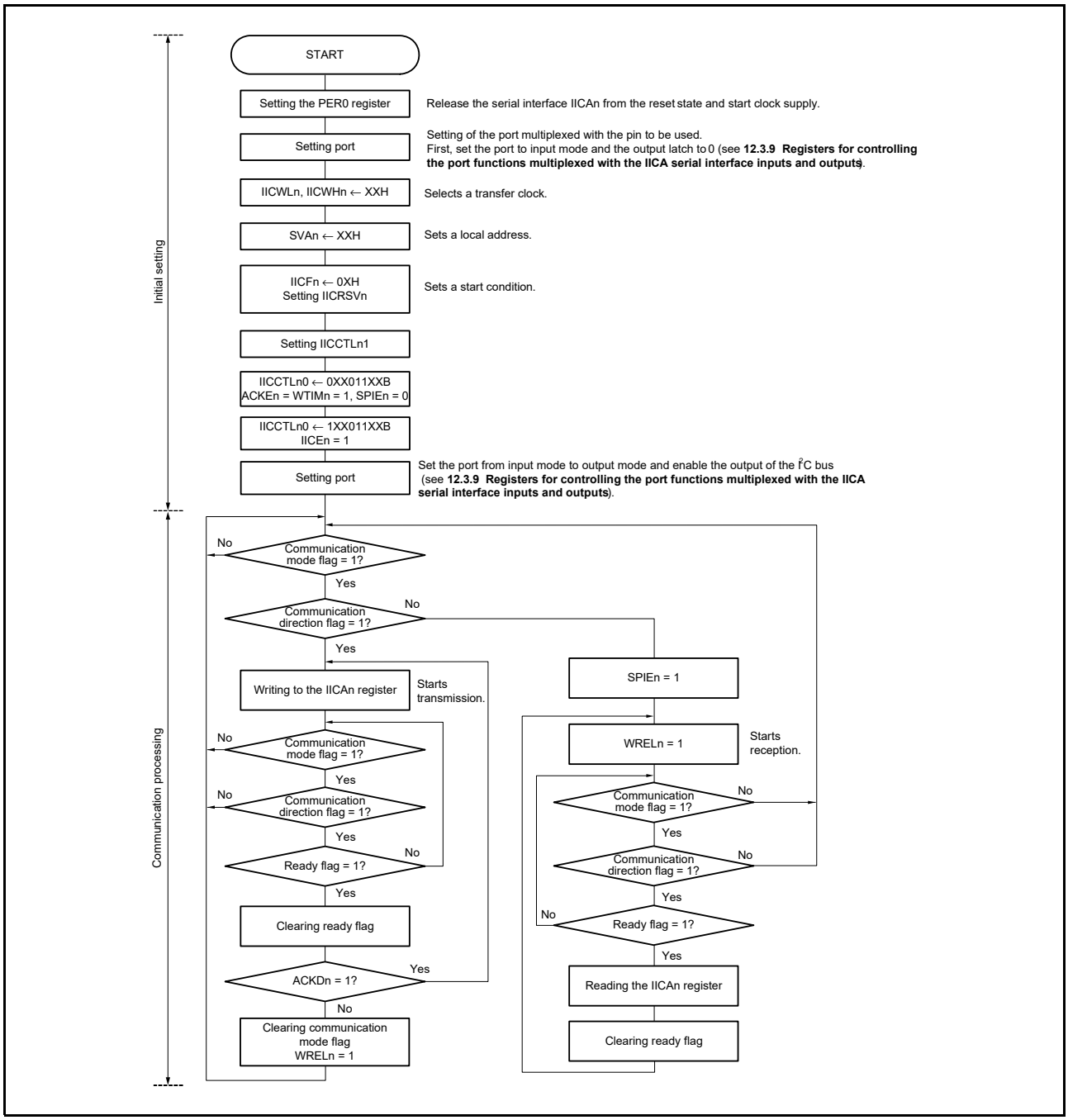
<3> Communication direction flag

This flag indicates the direction of communication. Its value is the same as the TRCn flag.

Remark n = 0

The main processing of the slave operation is explained next. Start serial interface IICA and wait until communication is enabled. When communication is enabled, execute communication by using the communication mode flag and ready flag (processing of the stop condition and start condition is performed by an interrupt. Here, check the state by using the flags). The transmission operation is repeated until the master no longer returns ACK. If ACK is not returned from the master, communication is completed. For reception, the necessary amount of data is received. When communication is completed, ACK is not returned as the next data. After that, the master generates a stop condition or restart condition. Exit from the communications state occurs in this way.

Figure 12 - 30 Slave Operation Flowchart (1)



Remark 1. Conform to the specifications of the product that is communicating, with respect to the transmission and reception formats.

Remark 2. n = 0

An example of the processing procedure of the slave with the INTIICAn interrupt is explained below (processing is performed assuming that the all address match function is disabled and no extension code is used). The INTIICAn interrupt checks the state, and the following operations are performed.

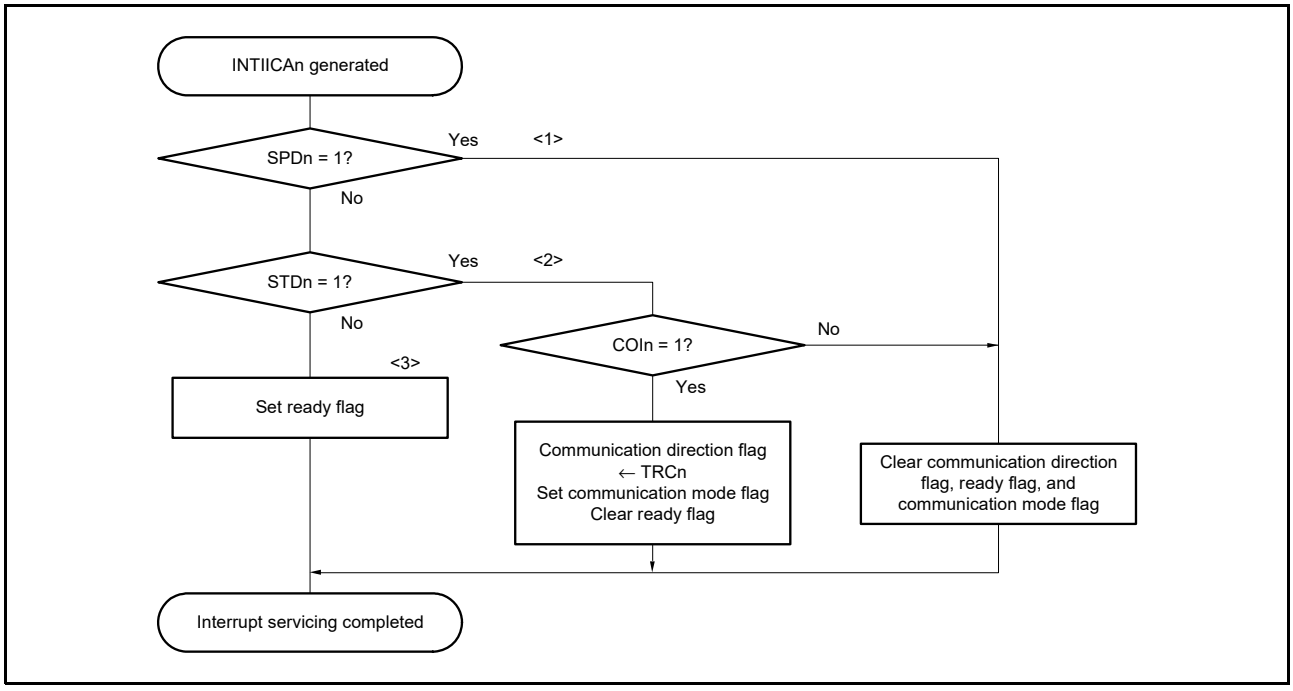
<1> Communication is stopped if the stop condition is issued.

<2> If the start condition is issued, the address is checked and communication is completed if the address does not match. If the address matches, the communication mode is set, wait is canceled, and processing returns from the interrupt (the ready flag is cleared).

<3> For data transmit/receive, only the ready flag is set. Processing returns from the interrupt with the I^2C bus remaining in the wait state.

Remark <1> to <3> above correspond to <1> to <3> in **Figure 12 - 31 Slave Operation Flowchart (2)**.

Figure 12 - 31 Slave Operation Flowchart (2)



Remark n = 0

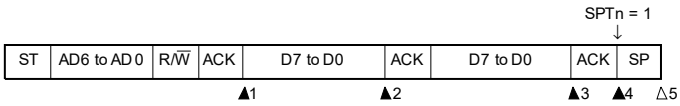
12.5.17 Timing of I²C interrupt request signal (INTIICAn) occurrence

The timing of transmitting or receiving data and generation of interrupt request signal INTIICAn, and the value of the IICA status register n (IICSn) when the INTIICAn signal is generated are shown below.

Remark 1. ST: Start condition
AD6 to AD0: Address
R/$\overline{W}$: Transfer direction specification
ACK: Acknowledge
D7 to D0: Data
SP: Stop condition

Remark 2. n = 0

1. Master device operation
   a) Start ~ Address ~ Data ~ Data ~ Stop (transmission/reception)
      i) When WTIMn = 0

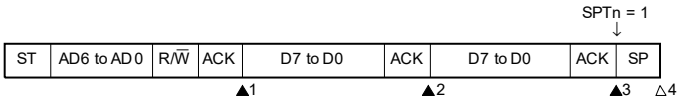


▲1: IICSn = 1000×110B
▲2: IICSn = 1000×000B
▲3: IICSn = 1000×000B (Sets the WTIMn bit to 1)**Note**
▲4: IICSn = 1000××00B (Sets the SPTn bit to 1)
△5: IICSn = 00000001B

Note To generate a stop condition, set the WTIMn bit to 1 and change the timing for generating the INTIICAn interrupt request signal.

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

      ii) When WTIMn = 1



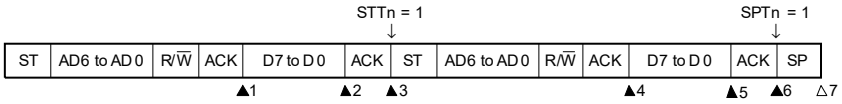
▲1: IICSn = 1000×110B
▲2: IICSn = 1000×100B
▲3: IICSn = 1000××00B (Sets the SPTn bit to 1)
△4: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

Remark n = 0

b) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop (restart)

i) When WTIMn = 0



▲1: IICSn = 1000×110B

▲2: IICSn = 1000×000B (Sets the WTIMn bit to 1)**Note 1**

▲3: IICSn = 1000××00B (Clears the WTIMn bit to 0**Note 2**, sets the STTn bit to 1)

▲4: IICSn = 1000×110B

▲5: IICSn = 1000×000B (Sets the WTIMn bit to 1)**Note 3**

▲6: IICSn = 1000××00B (Sets the SPTn bit to 1)

△7: IICSn = 00000001B

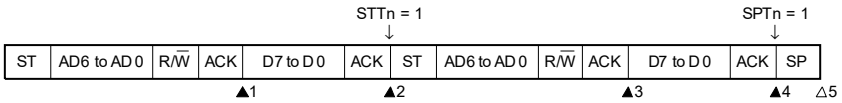
Note 1. To generate a start condition, set the WTIMn bit to 1 and change the timing for generating the INTIICAn interrupt request signal.

Note 2. Clear the WTIMn bit to 0 to restore the original setting.

Note 3. To generate a stop condition, set the WTIMn bit to 1 and change the timing for generating the INTIICAn interrupt request signal.

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

ii) When WTIMn = 1



▲1: IICSn = 1000×110B

▲2: IICSn = 1000××00B (Sets the STTn bit to 1)

▲3: IICSn = 1000×110B

▲4: IICSn = 1000××00B (Sets the SPTn bit to 1)

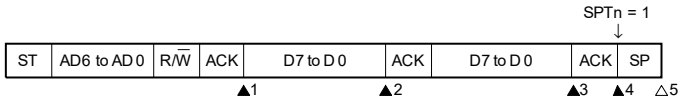
△5: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

Remark n = 0

c) Start ~ Code ~ Data ~ Data ~ Stop (extension code transmission)

i) When WTIMn = 0



▲1: IICSn = 1010×110B

▲2: IICSn = 1010×000B

▲3: IICSn = 1010×000B (Sets the WTIMn bit to 1)**Note**

▲4: IICSn = 1010××00B (Sets the SPTn bit to 1)

△5: IICSn = 00000001B

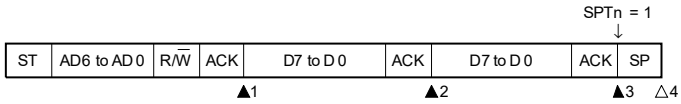
Note To generate a stop condition, set the WTIMn bit to 1 and change the timing for generating the INTIICAn interrupt request signal.

Remark ▲: Always generated

△: Generated only when SPIEn = 1

×: Don't care

ii) When WTIMn = 1



▲1: IICSn = 1010×110B

▲2: IICSn = 1010×100B

▲3: IICSn = 1010××00B (Sets the SPTn bit to 1)

△4: IICSn = 00000001B

Remark ▲: Always generated

△: Generated only when SPIEn = 1

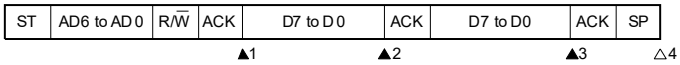
×: Don't care

Remark n = 0

2. Slave device operation (slave address data reception)

a) Start ~ Address ~ Data ~ Data ~ Stop

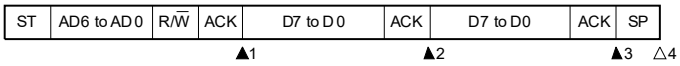
i) When WTIMn = 0



▲1: IICSn = 0001×110B
▲2: IICSn = 0001×000B
▲3: IICSn = 0001×000B
△4: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

ii) When WTIMn = 1



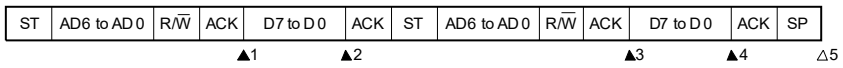
▲1: IICSn = 0001×110B
▲2: IICSn = 0001×100B
▲3: IICSn = 0001××00B
△4: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

Remark n = 0

b) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

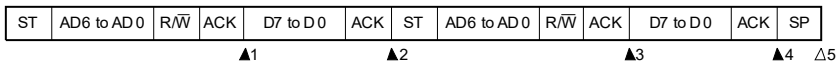
i) When WTIMn = 0 (after restart, matches with SVAn, the all address match function is disabled)



▲1: IICSn = 0001×110B
▲2: IICSn = 0001×000B
▲3: IICSn = 0001×110B
▲4: IICSn = 0001×000B
△5: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

ii) When WTIMn = 1 (after restart, matches with SVAn, the all address match function is disabled)



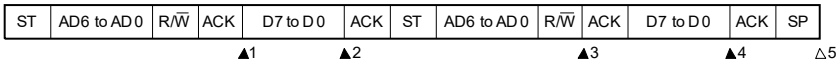
▲1: IICSn = 0001×110B
▲2: IICSn = 0001××00B
▲3: IICSn = 0001×110B
▲4: IICSn = 0001××00B
△5: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

Remark n = 0

c) Start ~ Address ~ Data ~ Start ~ Code ~ Data ~ Stop

i) When WTIMn = 0 (after restart, does not match address (= extension code, the all address match function is disabled))



▲1: IICSn = 0001×110B
▲2: IICSn = 0001×000B
▲3: IICSn = 0010×010B
▲4: IICSn = 0010×000B
△5: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

ii) When WTIMn = 1 (after restart, does not match address (= extension code, the all address match function is disabled))



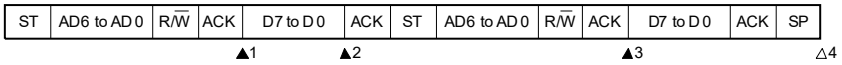
▲1: IICSn = 0001×110B
▲2: IICSn = 0001××00B
▲3: IICSn = 0010×010B
▲4: IICSn = 0010×110B
▲5: IICSn = 0010××00B
△6: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

Remark n = 0

d) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

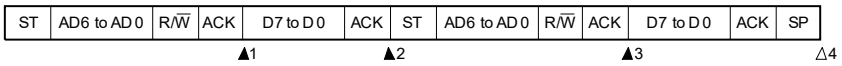
i) When WTIMn = 0 (after restart, does not match address (= not extension code, the all address match function is disabled))



▲1: IICSn = 0001×110B
▲2: IICSn = 0001×000B
▲3: IICSn = 00000×10B
△4: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

ii) When WTIMn = 1 (after restart, does not match address (= not extension code, the all address match function is disabled))

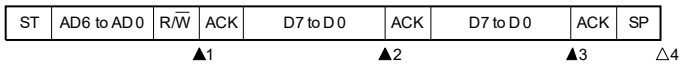


▲1: IICSn = 0001×110B
▲2: IICSn = 0001××00B
▲3: IICSn = 00000×10B
△4: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

Remark n = 0

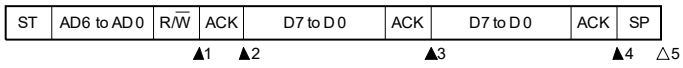
3. Slave device operation (when receiving extension code and the all address match function is disabled)
   The device is always participating in communication when it receives an extension code.
   a) Start ~ Code ~ Data ~ Data ~ Stop
      i) When WTIMn = 0



▲1: IICSn = 0010×010B
▲2: IICSn = 0010×000B
▲3: IICSn = 0010×000B
△4: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

      ii) When WTIMn = 1



▲1: IICSn = 0010×010B
▲2: IICSn = 0010×110B
▲3: IICSn = 0010×100B
▲4: IICSn = 0010××00B
△5: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

Remark n = 0

b) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

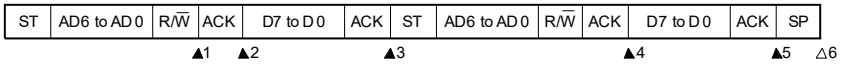
i) When WTIMn = 0 (after restart, matches with SVAn, the all address match function is disabled)



▲1: IICSn = 0010×010B
▲2: IICSn = 0010×000B
▲3: IICSn = 0001×110B
▲4: IICSn = 0001×000B
△5: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

ii) When WTIMn = 1 (after restart, matches with SVAn, the all address match function is disabled)



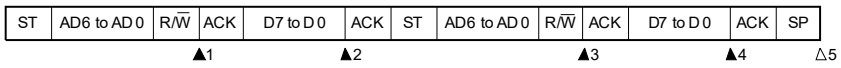
▲1: IICSn = 0010×010B
▲2: IICSn = 0010×110B
▲3: IICSn = 0010××00B
▲4: IICSn = 0001×110B
▲5: IICSn = 0001××00B
△6: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

Remark n = 0

c) Start ~ Code ~ Data ~ Start ~ Code ~ Data ~ Stop

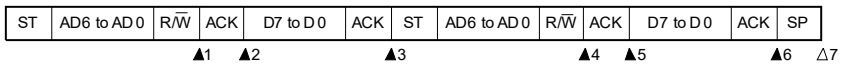
i) When WTIMn = 0 (after restart, extension code reception, the all address match function is disabled)



▲1: IICSn = 0010×010B
▲2: IICSn = 0010×000B
▲3: IICSn = 0010×010B
▲4: IICSn = 0010×000B
△5: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

ii) When WTIMn = 1 (after restart, extension code reception, the all address match function is disabled)



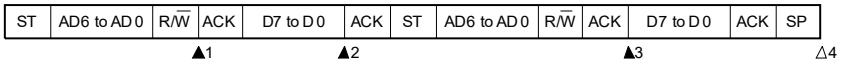
▲1: IICSn = 0010×010B
▲2: IICSn = 0010×110B
▲3: IICSn = 0010××00B
▲4: IICSn = 0010×010B
▲5: IICSn = 0010×110B
▲6: IICSn = 0010××00B
△7: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

Remark n = 0

d) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

i) When WTIMn = 0 (after restart, does not match address (= not extension code, the all address match function is disabled))



▲1: IICSn = 0010×010B
▲2: IICSn = 0010×000B
▲3: IICSn = 00000×10B
△4: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

ii) When WTIMn = 1 (after restart, does not match address (= not extension code, the all address match function is disabled))



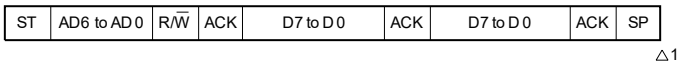
▲1: IICSn = 0010×010B
▲2: IICSn = 0010×110B
▲3: IICSn = 0010××00B
▲4: IICSn = 00000×10B
△5: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

Remark n = 0

4. Operation without communication

a) Start ~ Code ~ Data ~ Data ~ Stop



△1: IICSn = 00000001B

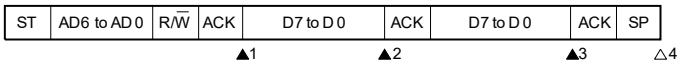
Remark △: Generated only when SPIEn = 1

5. Arbitration loss operation (operation as slave after arbitration loss)

When the device is used as a master in a multi-master system, read the MSTSn flag each time interrupt request signal INTIICAn has occurred to check the arbitration result.

a) When arbitration loss occurs during transmission of slave address data

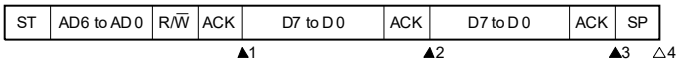
i) When WTIMn = 0



▲1: IICSn = 0101×110B
▲2: IICSn = 0001×000B
▲3: IICSn = 0001×000B
△4: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

ii) When WTIMn = 1



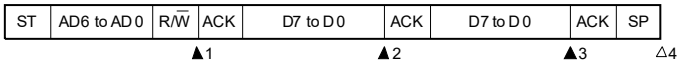
▲1: IICSn = 0101×110B
▲2: IICSn = 0001×100B
▲3: IICSn = 0001××00B
△4: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

Remark n = 0

b) When arbitration loss occurs during transmission of extension code (the all address match function is disabled)

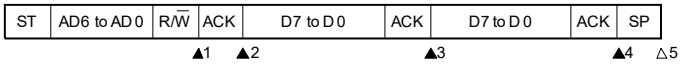
i) When WTIMn = 0



▲1: IICSn = 0110×010B
▲2: IICSn = 0010×000B
▲3: IICSn = 0010×000B
△4: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

ii) When WTIMn = 1

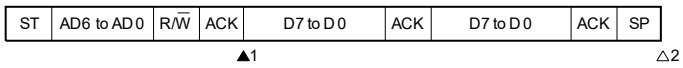


▲1: IICSn = 0110×010B
▲2: IICSn = 0010×110B
▲3: IICSn = 0010×100B
▲4: IICSn = 0010××00B
△5: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

Remark n = 0

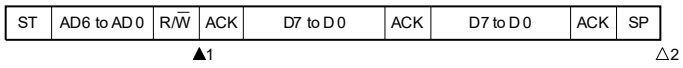
6. Operation when arbitration loss occurs (no communication after arbitration loss)
   When the device is used as a master in a multi-master system, read the MSTSn flag each time interrupt request signal INTIICAn has occurred to check the arbitration result.
   a) When arbitration loss occurs during transmission of slave address data (when WTIMn = 1)



▲1: IICSn = 01000110B
△2: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1

   b) When arbitration loss occurs during transmission of extension code (the all address match function is disabled)



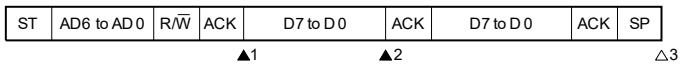
▲1: IICSn = 0110×010B
Sets LRELn = 1 by software
△2: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

Remark n = 0

c) When arbitration loss occurs during transmission of data

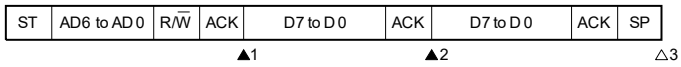
i) When WTIMn = 0



▲1: IICSn = 10001110B
▲2: IICSn = 01000000B
△3: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1

ii) When WTIMn = 1



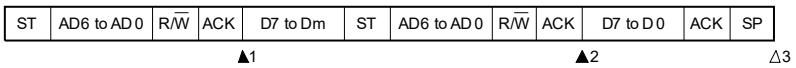
▲1: IICSn = 10001110B
▲2: IICSn = 01000100B
△3: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1

Remark n = 0

d) When loss occurs due to restart condition during data transfer

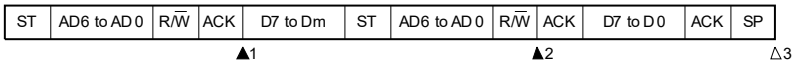
i) Not extension code (Example: unmatches with SVAn, the all address match function is disabled)



▲1: IICSn = 1000×110B
▲2: IICSn = 01000110B
△3: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care
m = 6 to 0

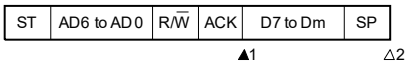
ii) Extension code (the all address match function is disabled)



▲1: IICSn = 1000×110B
▲2: IICSn = 01100010B
Sets LRELn = 1 by software
△3: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care
m = 6 to 0

e) When loss occurs due to stop condition during data transfer



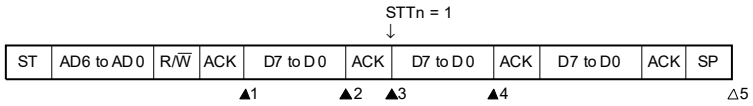
▲1: IICSn = 10000110B
△2: IICSn = 01000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care
m = 6 to 0

Remark n = 0

f) When arbitration loss occurs due to low-level data when attempting to generate a restart condition

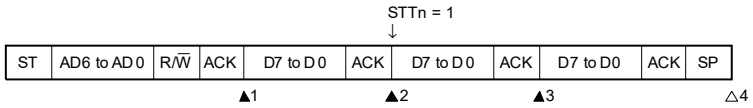
i) When WTIMn = 0



▲1: IICSn = 1000×110B
▲2: IICSn = 1000×000B (Sets the WTIMn bit to 1)
▲3: IICSn = 1000×100B (Clears the WTIMn bit to 0)
▲4: IICSn = 01000000B
△5: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

ii) When WTIMn = 1



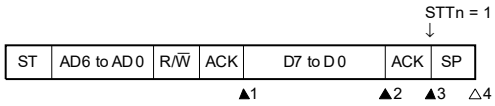
▲1: IICSn = 1000×110B
▲2: IICSn = 1000×100B (Sets the STTn bit to 1)
▲3: IICSn = 01000100B
△4: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

Remark n = 0

g) When arbitration loss occurs due to a stop condition when attempting to generate a restart condition

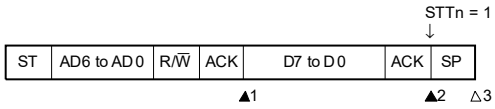
i) When WTIMn = 0



▲1: IICSn = 1000×110B
▲2: IICSn = 1000×000B (Sets the WTIMn bit to 1)
▲3: IICSn = 1000××00B (Sets the STTn bit to 1)
△4: IICSn = 01000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

ii) When WTIMn = 1



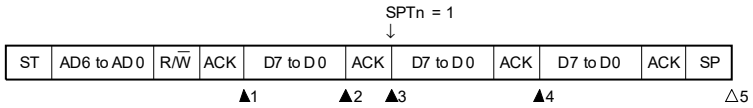
▲1: IICSn = 1000×110B
▲2: IICSn = 1000××00B (Sets the STTn bit to 1)
△3: IICSn = 01000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

Remark n = 0

h) When arbitration loss occurs due to low-level data when attempting to generate a stop condition

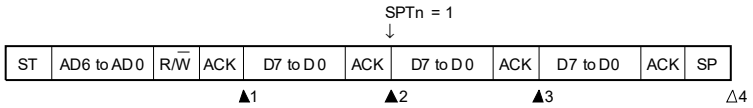
i) When WTIMn = 0



▲1: IICSn = 1000×110B
▲2: IICSn = 1000×000B (Sets the WTIMn bit to 1)
▲3: IICSn = 1000×100B (Clears the WTIMn bit to 0)
▲4: IICSn = 01000100B
△5: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

ii) When WTIMn = 1



▲1: IICSn = 1000×110B
▲2: IICSn = 1000×100B (Sets the SPTn bit to 1)
▲3: IICSn = 01000100B
△4: IICSn = 00000001B

Remark ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

Remark n = 0

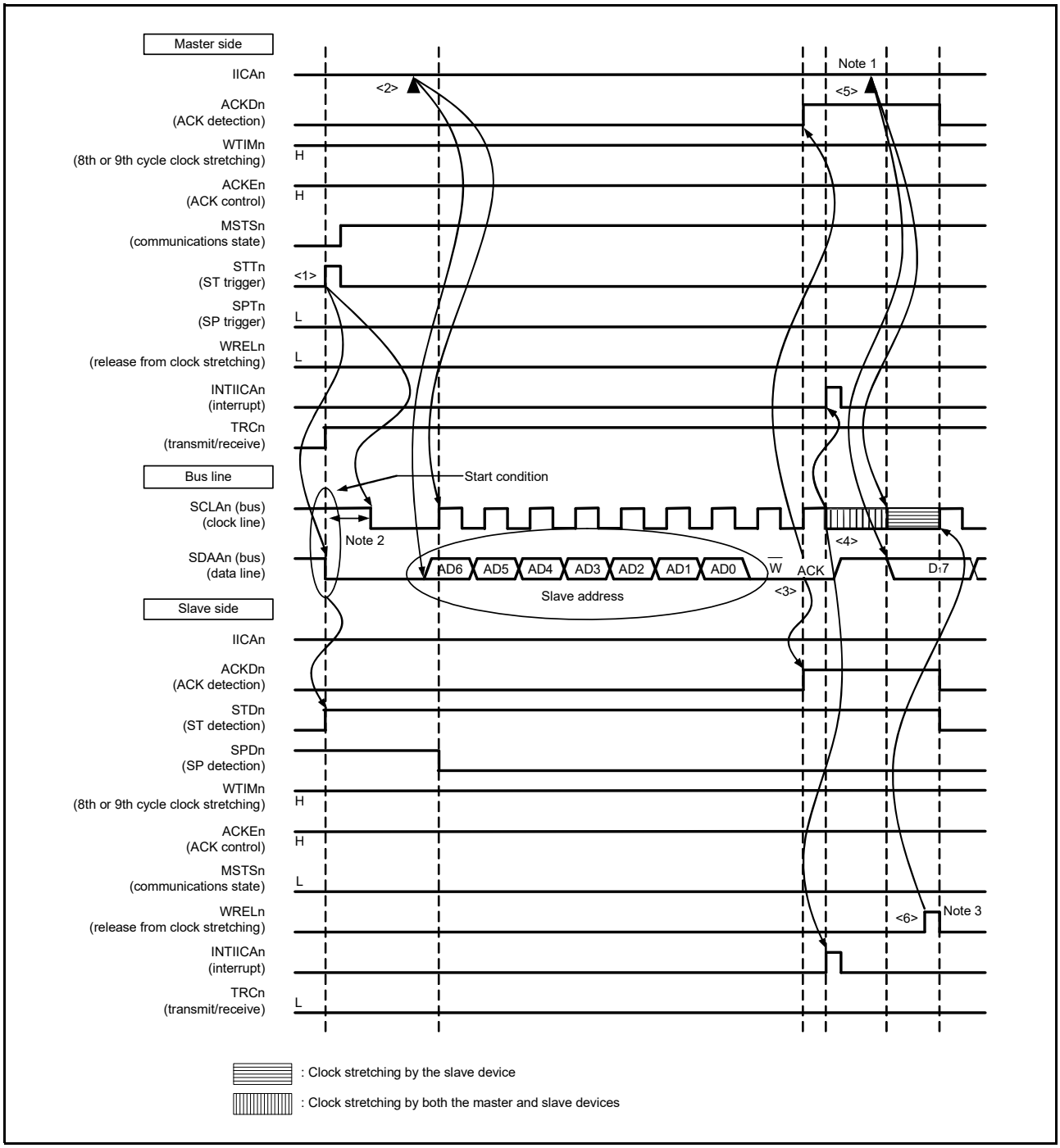
12.6 Timing Charts

When using the I^2C bus mode, the master device outputs an address via the serial bus to select one of several slave devices as its communication partner. After outputting the slave address, the master device transmits the TRCn flag of the IICA status register n (IICSn), which specifies the data transfer direction, and then starts serial communication with the slave device. **Figure 12 - 32** and **Figure 12 - 33** show timing charts of the data communication. The IICA shift register n (IICAn)'s shift operation is synchronized with the falling edge of the serial clock (SCLAn). The transmit data is transferred to the SO latch and is output (MSB first) via the SDAAn pin. Data input via the SDAAn pin is captured into IICAn at the rising edge of SCLAn. In the timing charts described in this section, it is assumed that the all address match function is disabled.

Remark n = 0

Figure 12 - 32 Example of Master to Slave Communications
(9th Cycle Clock Stretching Is Selected for Both the Master and Slave) (1/4)

1. Start condition to address to data



Note 1. For release from the clock stretch state during transmission by a master device, write data to the IICAn register instead of setting the WRELn bit.

Note 2. Make sure that the time between the fall of the SDAAn pin signal and the fall of the SCLAn pin signal is at least 4.0 μs when specifying standard mode and at least 0.6 μs when specifying fast mode.

Note 3. For release from the clock stretch state during reception by a slave device, write FFH to IICAn or set the WRELn bit.

Remark n = 0

The meanings of <1> to <6> in **1. Start condition to address to data** in **Figure 12 - 32** are explained below.

<1> The start condition trigger is set by the master device (STTn = 1) and a start condition (i.e. SCLAn = 1 and SDAAn changes from 1 to 0) is generated once the bus data line goes low (SDAAn). When the start condition is subsequently detected, the master device enters the master device communications state (MSTSn = 1). The master device is ready to communicate once the bus clock line goes low (SCLAn = 0) after the hold time has elapsed.

<2> The master device writes the address + W (transmission) to the IICA shift register n (IICAn) and transmits the slave address.

<3> In the slave device if the address received matches the address (SVAn value) of a slave device**Note**, that slave device sends an ACK by hardware to the master device. The ACK is detected by the master device (ACKDn = 1) at the rising edge of the 9th clock.

<4> The master device issues an interrupt (INTIICAn: end of address transmission) at the falling edge of the 9th clock. The slave device with the address matching the transmitted slave address sets the clock stretch state (SCLAn = 0) and issues an interrupt (INTIICAn: address match)**Note**.

<5> The master device writes the data to transmit to the IICAn register and releases the clock stretch state set by the master device.

<6> If the slave device releases the clock stretch state (WRELn = 1), the master device starts transferring data to the slave device.

Note If the transmitted address does not match the address of the slave device, the slave device does not return an ACK to the master device (NACK: SDAAn = 1). The slave device also does not issue the INTIICAn interrupt (address match) and does not set the clock stretch state. The master device, however, issues the INTIICAn interrupt (end of address transmission) regardless of whether it receives an ACK or NACK.

Remark 1. <1> to <15> in **Figure 12 - 32** represent the entire procedure for communicating data using the I^2C bus.

1. Start condition to address to data in **Figure 12 - 32**: Steps <1> to <6>

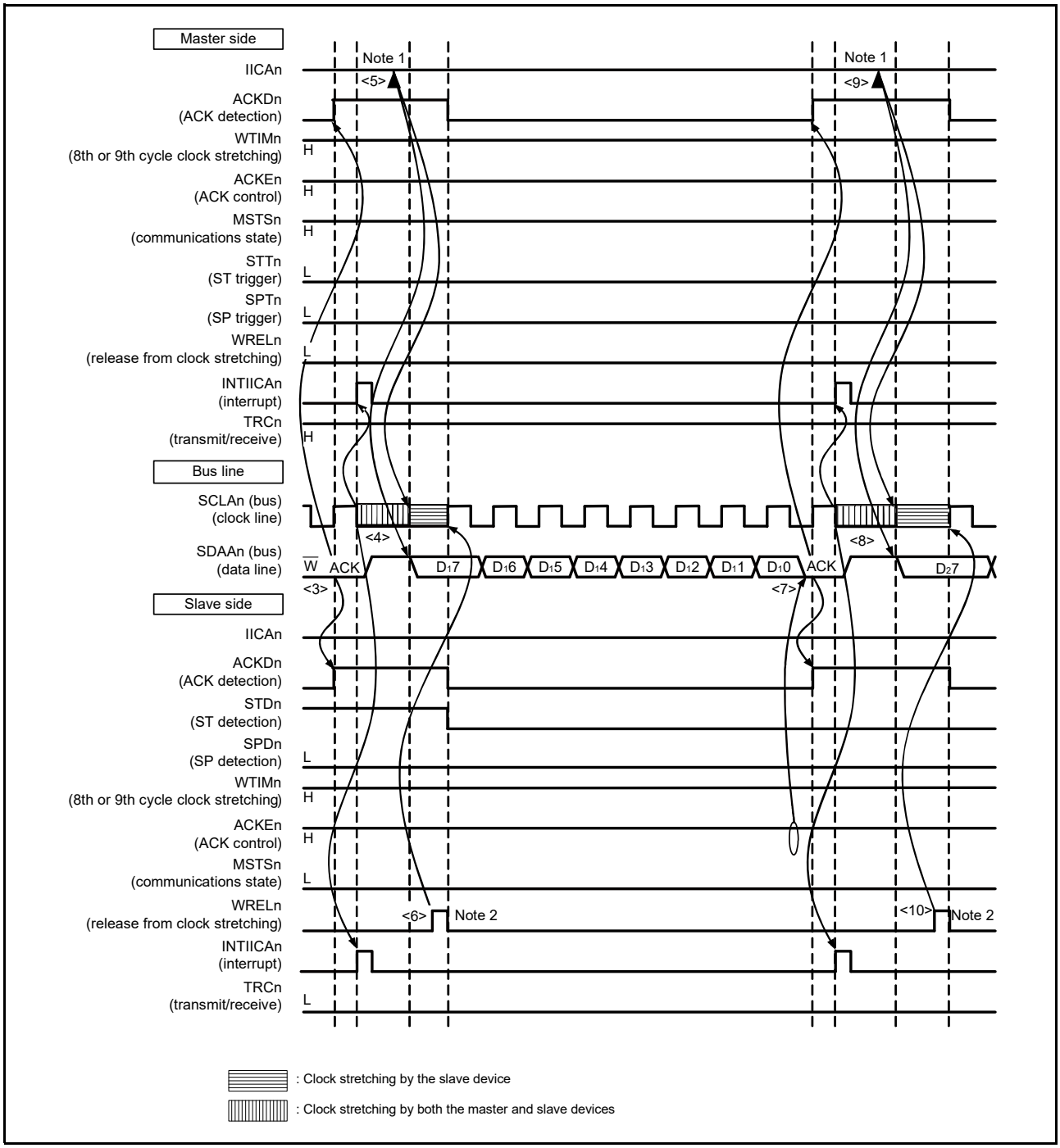
2. Address to data to data in **Figure 12 - 32**: Steps <3> to <10>

3. Data to data to stop condition in **Figure 12 - 32**: Steps <7> to <15>

Remark 2. n = 0

Figure 12 - 32 Example of Master to Slave Communications
(9th Cycle Clock Stretching Is Selected for Both the Master and Slave) (2/4)

2. Address to data to data



Note 1. For release from the clock stretch state during transmission by a master device, write data to the IICAn register instead of setting the WRELn bit.

Note 2. For release from the clock stretch state during reception by a slave device, write FFH to IICAn or set the WRELn bit.

Remark n = 0

The meanings of <3> to <10> in **2. Address to data to data** in **Figure 12 - 32** are explained below.

<3> In the slave device if the address received matches the address (SVAn value) of a slave device**Note**, that slave device sends an ACK by hardware to the master device. The ACK is detected by the master device (ACKDn = 1) at the rising edge of the 9th clock.

<4> The master device issues an interrupt (INTIICAn: end of address transmission) at the falling edge of the 9th clock. The slave device with the address matching the transmitted slave address sets the clock stretch state (SCLAn = 0) and issues an interrupt (INTIICAn: address match)**Note**.

<5> The master device writes the data to transmit to the IICA shift register n (IICAn) and releases the clock stretch state set by the master device.

<6> If the slave device releases the clock stretch state (WRELn = 1), the master device starts transferring data to the slave device.

<7> After data transfer is completed, because of ACKEn = 1, the slave device sends an ACK by hardware to the master device. The ACK is detected by the master device (ACKDn = 1) at the rising edge of the 9th clock.

<8> The master device and slave device set the clock stretch state (SCLAn = 0) at the falling edge of the 9th clock, and both the master device and slave device issue an interrupt (INTIICAn: end of transfer).

<9> The master device writes the data to transmit to the IICAn register and releases the clock stretch state set by the master device.

<10> The slave device reads the received data and releases the clock stretch state (WRELn = 1). The master device then starts transferring data to the slave device.

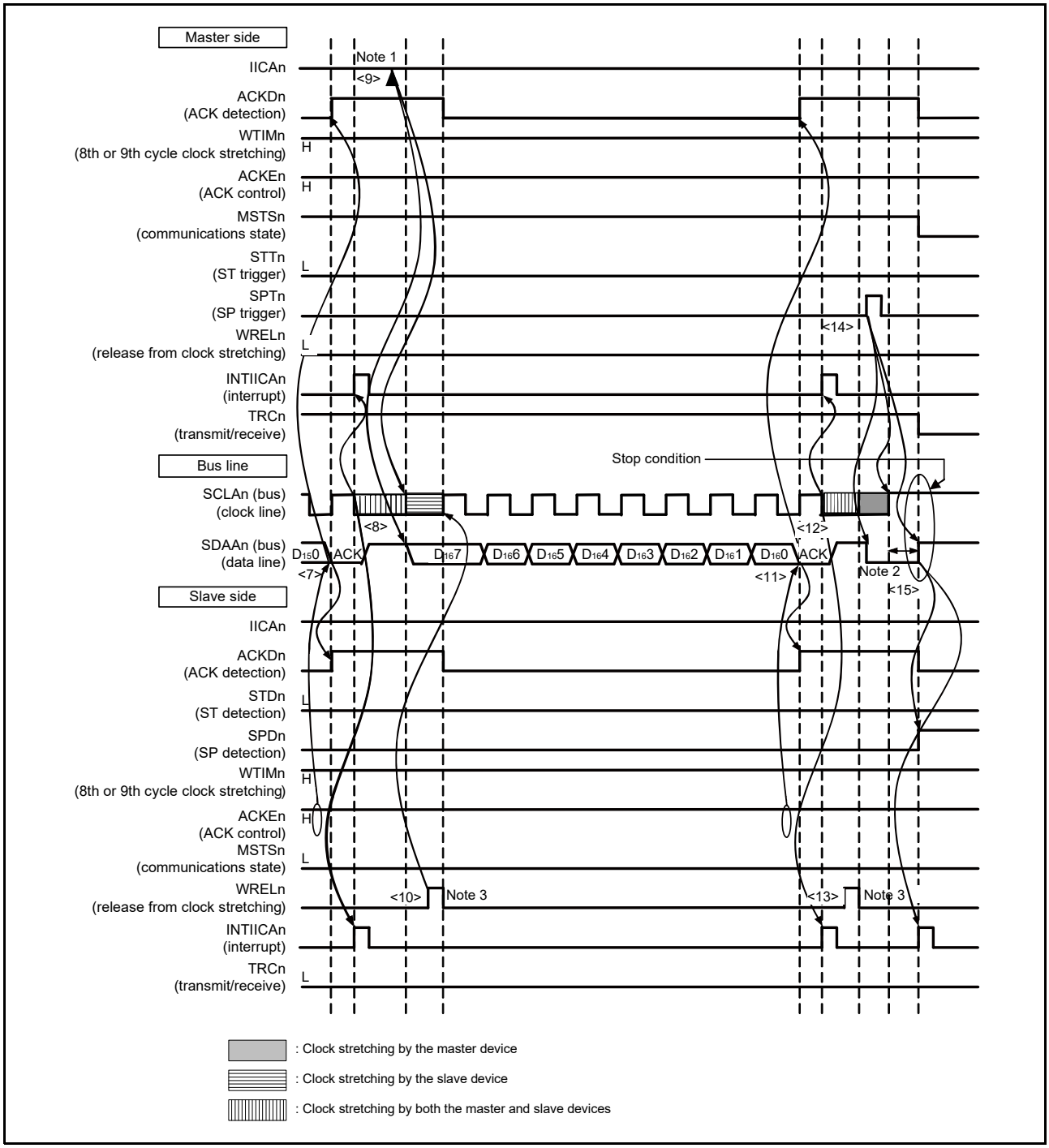
Note If the transmitted address does not match the address of the slave device, the slave device does not return an ACK to the master device (NACK: SDAAn = 1). The slave device also does not issue the INTIICAn interrupt (address match) and does not set the clock stretch state. The master device, however, issues the INTIICAn interrupt (end of address transmission) regardless of whether it receives an ACK or NACK.

Remark 1. <1> to <15> in **Figure 12 - 32** represent the entire procedure for communicating data using the I^2C bus.
1. Start condition to address to data in **Figure 12 - 32**: Steps <1> to <6>
2. Address to data to data in **Figure 12 - 32**: Steps <3> to <10>
3. Data to data to stop condition in **Figure 12 - 32**: Steps <7> to <15>

Remark 2. n = 0

Figure 12 - 32 Example of Master to Slave Communications
(9th Cycle Clock Stretching Is Selected for Both the Master and Slave) (3/4)

3. Data to data to stop condition



Note 1. For release from the clock stretch state during transmission by a master device, write data to the IICAn register instead of setting the WRELn bit.

Note 2. Make sure that the time between the rise of the SCLAn pin signal and the generation of the stop condition after a stop condition has been issued is at least 4.0 μs when specifying standard mode and at least 0.6 μs when specifying fast mode.

Note 3. For release from the clock stretch state during reception by a slave device, write FFH to IICAn or set the WRELn bit.

Remark n = 0

The meanings of <7> to <15> in **3. Data to data to stop condition** in **Figure 12 - 32** are explained below.

<7> After data transfer is completed, because of ACKEn = 1, the slave device sends an ACK by hardware to the master device. The ACK is detected by the master device (ACKDn = 1) at the rising edge of the 9th clock.

<8> The master device and slave device set the clock stretch state (SCLAn = 0) at the falling edge of the 9th clock, and both the master device and slave device issue an interrupt (INTIICAn: end of transfer).

<9> The master device writes the data to transmit to the IICA shift register n (IICAn) and releases the clock stretch state set by the master device.

<10> The slave device reads the received data and releases the clock stretch state (WRELn = 1). The master device then starts transferring data to the slave device.

<11> When data transfer is complete, the slave device (ACKEn =1) sends an ACK by hardware to the master device. The ACK is detected by the master device (ACKDn = 1) at the rising edge of the 9th clock.

<12> The master device and slave device set the clock stretch state (SCLAn = 0) at the falling edge of the 9th clock, and both the master device and slave device issue an interrupt (INTIICAn: end of transfer).

<13> The slave device reads the received data and releases the clock stretch state (WRELn = 1).

<14> By the master device setting a stop condition trigger (SPTn = 1), the bus data line is cleared (SDAAn = 0) and the bus clock line is set (SCLAn = 1). After the stop condition setup time has elapsed, by setting the bus data line (SDAAn = 1), the stop condition is then generated (i.e. SCLAn =1 and SDAAn changes from 0 to 1).

<15> When a stop condition is generated, the slave device detects the stop condition and issues an interrupt (INTIICAn: stop condition).

Remark 1. <1> to <15> in **Figure 12 - 32** represent the entire procedure for communicating data using the I^2C bus.

1. Start condition to address to data in **Figure 12 - 32**: Steps <1> to <6>

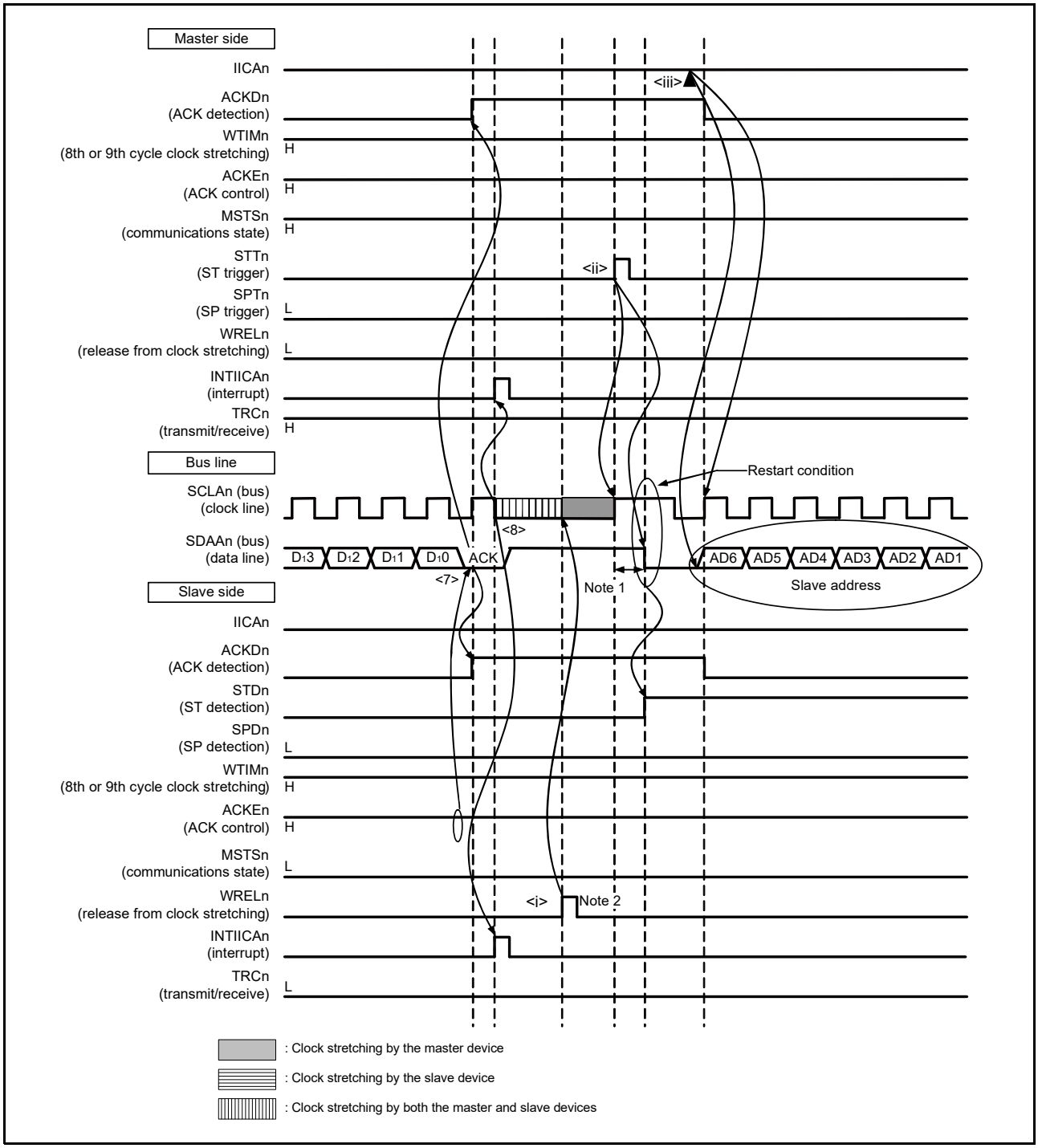
2. Address to data to data in **Figure 12 - 32**: Steps <3> to <10>

3. Data to data to stop condition in **Figure 12 - 32**: Steps <7> to <15>

Remark 2. n = 0

Figure 12 - 32 Example of Master to Slave Communications
(9th Cycle Clock Stretching Is Selected for Both the Master and Slave) (4/4)

4. Data to restart condition to address



Note 1. Make sure that the time between the rise of the SCLAn pin signal and the generation of the start condition after a restart condition has been issued is at least 4.7 μs when specifying standard mode and at least 0.6 μs when specifying fast mode.

Note 2. For release from the clock stretch state during reception by a slave device, write FFH to IICAn or set the WRELn bit.

Remark n = 0

The following describes the operations in **4. Data to restart condition to address** in **Figure 12 - 32**. After the operations in steps <7> and <8>, the operations in steps <i> to <iii> are performed. These steps return the processing to step <3>, the data transmission step.

<7> After data transfer is completed, because of ACKEn = 1, the slave device sends an ACK by hardware to the master device. The ACK is detected by the master device (ACKDn = 1) at the rising edge of the 9th clock.

<8> The master device and slave device set the clock stretch state (SCLAn = 0) at the falling edge of the 9th clock, and both the master device and slave device issue an interrupt (INTIICAn: end of transfer).

<i> The slave device reads the received data and releases the clock stretch state (WRELn = 1).

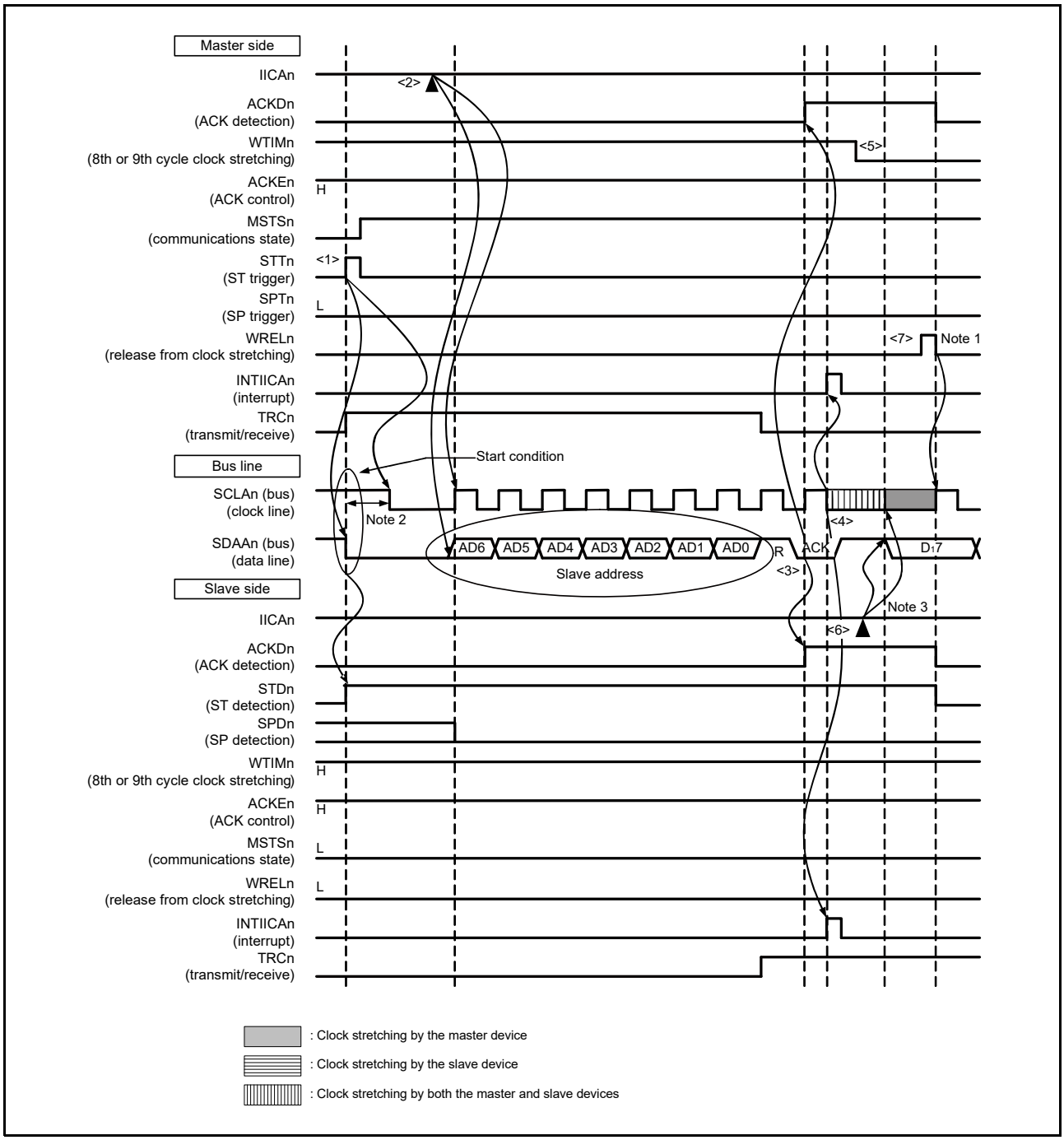
<ii> The start condition trigger is set again by the master device (STTn = 1) and a start condition (i.e. SCLAn =1 and SDAAn changes from 1 to 0) is generated once the bus clock line goes high (SCLAn = 1) and the bus data line goes low (SDAAn = 0) after the restart condition setup time has elapsed. When the start condition is subsequently detected, the master device is ready to communicate once the bus clock line goes low (SCLAn = 0) after the hold time has elapsed.

<iii> The master device writing the address + R/W (transmission) to the IICA shift register (IICAn) enables the slave address to be transmitted.

Remark n = 0

Figure 12 - 33 Example of Slave to Master Communications
(8th Cycle Clock Stretching Is Selected for the Master and 9th Cycle Clock Stretching Is Selected for the Slave) (1/3)

1. Start condition to address to data



Note 1. For release from the clock stretch state during reception by a master device, write FFH to IICAn or set the WRELn bit.

Note 2. Make sure that the time between the fall of the SDAAn pin signal and the fall of the SCLAn pin signal is at least 4.0 μs when specifying standard mode and at least 0.6 μs when specifying fast mode.

Note 3. For release from the clock stretch state during transmission by a slave device, write data to the IICAn register instead of setting the WRELn bit.

Remark n = 0

The meanings of <1> to <7> in **1. Start condition to address to data** in **Figure 12 - 33** are explained below.

<1> The start condition trigger is set by the master device (STTn = 1) and a start condition (i.e. SCLAn = 1 and SDAAn changes from 1 to 0) is generated once the bus data line goes low (SDAAn). When the start condition is subsequently detected, the master device enters the master device communications state (MSTSn = 1). The master device is ready to communicate once the bus clock line goes low (SCLAn = 0) after the hold time has elapsed.

<2> The master device writes the address + R (reception) to the IICA shift register n (IICAn) and transmits the slave address.

<3> In the slave device if the address received matches the address (SVAn value) of a slave device**Note**, that slave device sends an ACK by hardware to the master device. The ACK is detected by the master device (ACKDn = 1) at the rising edge of the 9th clock.

<4> The master device issues an interrupt (INTIICAn: end of address transmission) at the falling edge of the 9th clock. The slave device with the address matching the transmitted slave address sets the clock stretch state (SCLAn = 0) and issues an interrupt (INTIICAn: address match)**Note**.

<5> The timing at which the master device sets the clock stretch state changes to the 8th clock (WTIMn = 0).

<6> The slave device writes the data to transmit to the IICAn register and releases the clock stretch state set by the slave device.

<7> The master device releases the clock stretch state (WRELn = 1) and starts transferring data from the slave device to the master device.

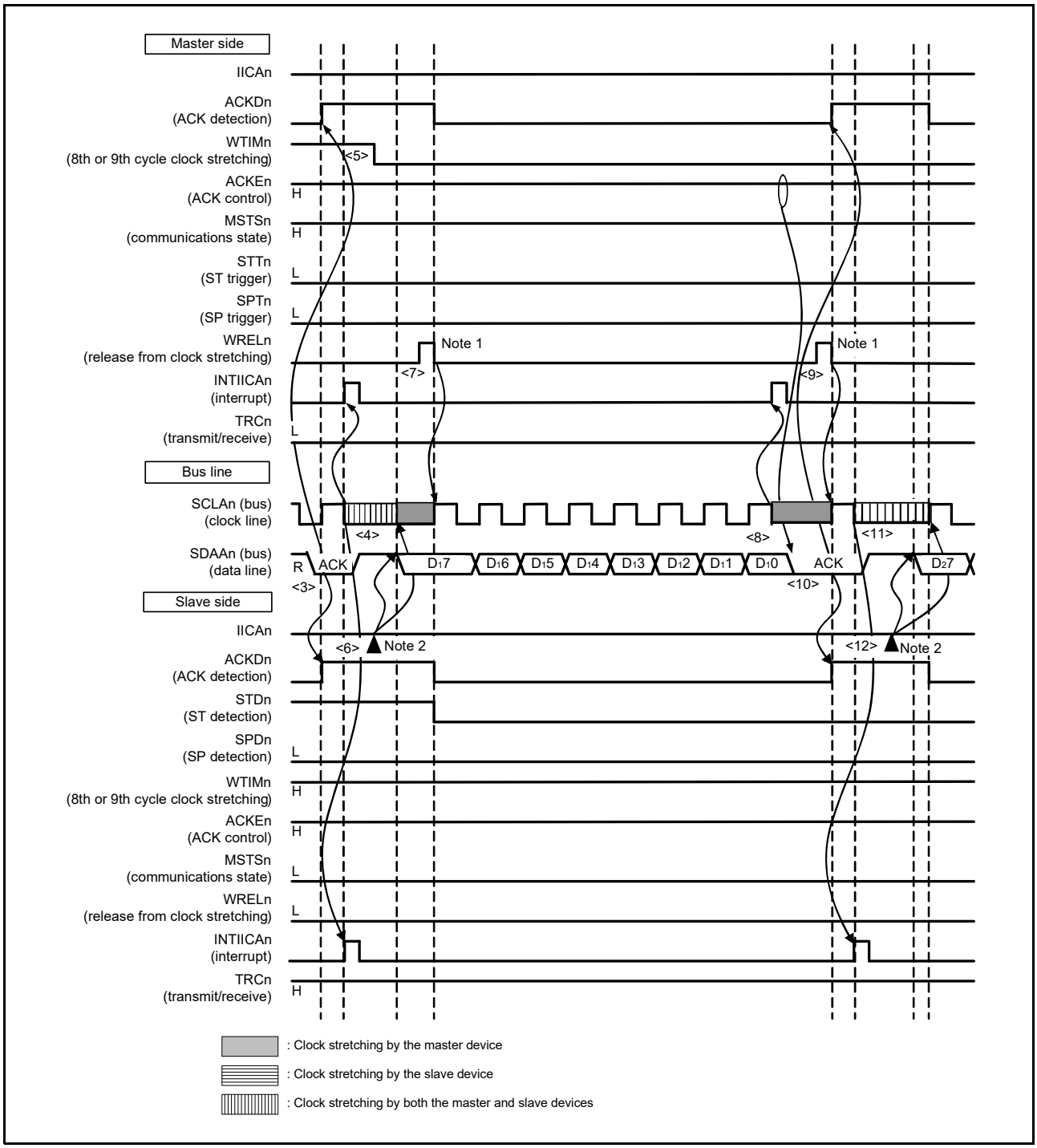
Note If the transmitted address does not match the address of the slave device, the slave device does not return an ACK to the master device (NACK: SDAAn = 1). The slave device also does not issue the INTIICAn interrupt (address match) and does not set the clock stretch state. The master device, however, issues the INTIICAn interrupt (end of address transmission) regardless of whether it receives an ACK or NACK.

Remark 1. <1> to <19> in **Figure 12 - 33** represent the entire procedure for communicating data using the I^2C bus.
1. Start condition to address to data in **Figure 12 - 33**: Steps <1> to <7>
2. Address to data to data in **Figure 12 - 33**: Steps <3> to <12>
3. Data to data to stop condition in **Figure 12 - 33**: Steps <8> to <19>

Remark 2. n = 0

Figure 12 - 33 Example of Slave to Master Communications
(8th Cycle Clock Stretching Is Selected for the Master and 9th Cycle Clock Stretching Is Selected for the Slave) (2/3)

2. Address to data to data



Note 1. For release from the clock stretch state during reception by a master device, write FFH to IICAn or set the WRELn bit.

Note 2. For release from the clock stretch state during transmission by a slave device, write data to the IICAn register instead of setting the WRELn bit.

Remark n = 0

The meanings of <3> to <12> in **2. Address to data to data** in **Figure 12 - 33** are explained below.

<3> In the slave device if the address received matches the address (SVAn value) of a slave device**Note**, that slave device sends an ACK by hardware to the master device. The ACK is detected by the master device (ACKDn = 1) at the rising edge of the 9th clock.

<4> The master device issues an interrupt (INTIICAn: end of address transmission) at the falling edge of the 9th clock. The slave device with the address matching the transmitted slave address sets the clock stretch state (SCLAn = 0) and issues an interrupt (INTIICAn: address match)**Note**.

<5> The master device changes the timing of clock stretching to the 8th clock (WTIMn = 0).

<6> The slave device writes the data to transmit to the IICA shift register n (IICAn) and releases the clock stretch state set by the slave device.

<7> The master device releases the clock stretch state (WRELn = 1) and starts transferring data from the slave device to the master device.

<8> The master device sets the clock stretch state (SCLAn = 0) at the falling edge of the 8th clock, and issues an interrupt (INTIICAn: end of transfer). Because of ACKEn = 1 in the master device, the master device then sends an ACK by hardware to the slave device.

<9> The master device reads the received data and releases the clock stretch state (WRELn = 1).

<10> The ACK is detected by the slave device (ACKDn = 1) at the rising edge of the 9th clock.

<11> The slave device sets the clock stretch state (SCLAn = 0) at the falling edge of the 9th clock, and the slave device issue an interrupt (INTIICAn: end of transfer).

<12> By the slave device writing the data to transmit to the IICAn register, the clock stretch state set by the slave device is released. The slave device then starts transferring data to the master device.

Note If the transmitted address does not match the address of the slave device, the slave device does not return an ACK to the master device (NACK: SDAAn = 1). The slave device also does not issue the INTIICAn interrupt (address match) and does not set the clock stretch state. The master device, however, issues the INTIICAn interrupt (end of address transmission) regardless of whether it receives an ACK or NACK.

Remark 1. <1> to <19> in **Figure 12 - 33** represent the entire procedure for communicating data using the I^2C bus.

1. Start condition to address to data in **Figure 12 - 33**: Steps <1> to <7>

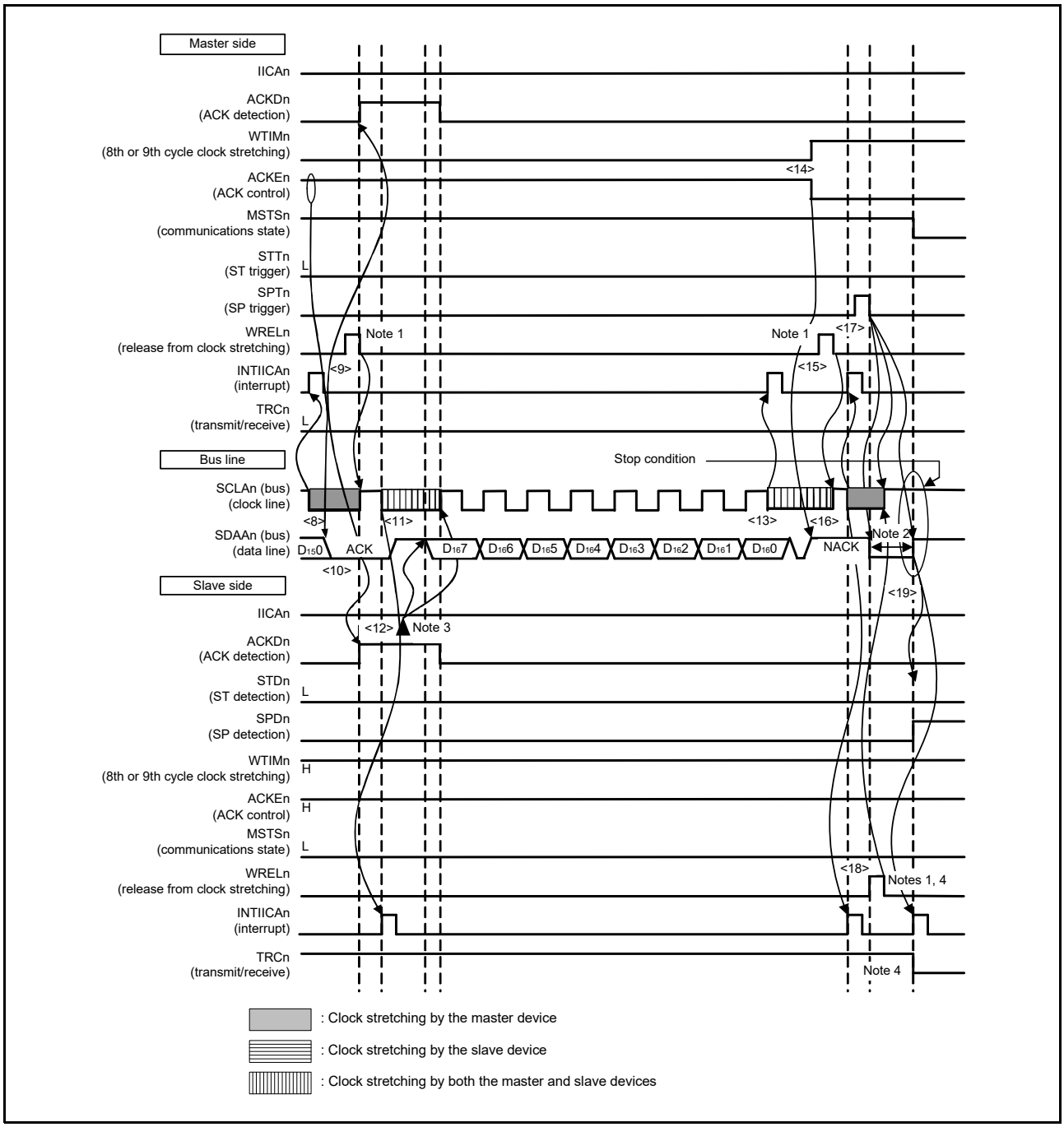
2. Address to data to data in **Figure 12 - 33**: Steps <3> to <12>

3. Data to data to stop condition in **Figure 12 - 33**: Steps <8> to <19>

Remark 2. n = 0

Figure 12 - 33 Example of Slave to Master Communications
(8th Cycle Clock Stretching Is Changed to 9th Cycle Clock Stretching for the Master and 9th Cycle Clock Stretching Is Selected for the Slave) (3/3)

3. Data to data to stop condition



Note 1. For release from the clock stretch state, write FFH to IICAn or set the WRELn bit.

Note 2. Make sure that the time between the rise of the SCLAn pin signal and the generation of the stop condition after a stop condition has been issued is at least 4.0 μs when specifying standard mode and at least 0.6 μs when specifying fast mode.

Note 3. For release from the clock stretch state during transmission by a slave device, write data to the IICAn register instead of setting the WRELn bit.

Note 4. If the clock stretch state during transmission by a slave device is released by setting the WRELn bit, the TRCn flag will be cleared.

Remark n = 0

The meanings of <8> to <19> in **3. Data to data to stop condition** in **Figure 12 - 33** are explained below.

<8> The master device sets the clock stretch state (SCLAn = 0) at the falling edge of the 8th clock, and issues an interrupt (INTIICAn: end of transfer). Because of ACKEn = 0 in the master device, the master device then sends an ACK by hardware to the slave device.

<9> The master device reads the received data and releases the clock stretch state (WRELn = 1).

<10> The ACK is detected by the slave device (ACKDn = 1) at the rising edge of the 9th clock.

<11> The slave device sets the clock stretch state (SCLAn = 0) at the falling edge of the 9th clock, and the slave device issue an interrupt (INTIICAn: end of transfer).

<12> By the slave device writing the data to transmit to the IICA register, the clock stretch state set by the slave device is released. The slave device then starts transferring data to the master device.

<13> The master device issues an interrupt (INTIICAn: end of transfer) at the falling edge of the 8th clock, and sets the clock stretch state (SCLAn = 0). Because ACK control (ACKEn = 1) is performed, the bus data line is at the low level (SDAAn = 0) at this stage.

<14> The master device sets NACK as the response (ACKEn = 0) and changes the timing at which it sets the clock stretch state to the 9th clock (WTIMn = 1).

<15> If the master device releases the clock stretch state (WRELn = 1), the slave device detects the NACK (ACK = 0) at the rising edge of the 9th clock.

<16> The master device and slave device set the clock stretch state (SCLAn = 0) at the falling edge of the 9th clock, and both the master device and slave device issue an interrupt (INTIICAn: end of transfer).

<17> When the master device issues a stop condition (SPTn = 1), the bus data line is cleared (SDAAn = 0) and the master device releases the clock stretch state. The master device then waits until the bus clock line is set (SCLAn = 1).

<18> The slave device acknowledges the NACK, halts transmission, and releases the clock stretch state (WRELn = 1) to end communication. Once the slave device releases the clock stretch state, the bus clock line is set (SCLAn = 1).

<19> Once the master device recognizes that the bus clock line is set (SCLAn = 1) and after the stop condition setup time has elapsed, the master device sets the bus data line (SDAAn = 1) and issues a stop condition (i.e. SCLAn =1 and SDAAn changes from 0 to 1). The slave device detects the generated stop condition and slave device issue an interrupt (INTIICAn: stop condition).

Remark 1. <1> to <19> in **Figure 12 - 33** represent the entire procedure for communicating data using the I^2C bus.

1. Start condition to address to data in **Figure 12 - 33**: Steps <1> to <7>

2. Address to data to data in **Figure 12 - 33**: Steps <3> to <12>

3. Data to data to stop condition in **Figure 12 - 33**: Steps <8> to <19>

Remark 2. n = 0

Section 13 Serial Interface UARTA (UARTA)

Each product of this group has a single serial interface UARTA.

13.1 Overview

The serial interface UARTAn supports the following two modes.

1. Operation stop mode
   This mode is used when serial communication is not executed; power consumption can be reduced.
2. UART mode
   This is a UART mode that supports continuous transmission. The list below outlines the features:

The serial interface UARTAn performs an asynchronous communication. It has the following functions.

- Maximum transfer rate: 153.6 kbps
- Transmission and reception using two pins
  TxDAn: Transmit data output pin
  RxDAn: Receive data input pin
- Character length of transfer data selectable from 5, 7, and 8 bits
- Baud rate arbitrarily settable with the dedicated internal 8-bit baud rate generator
- Transmission and reception independent of each other (full-duplex communication)
- MSB or LSB first transfer selectable
- Inversion control of communication logic level provided
- Transmit clock output is available (CLKAn)
- Operating clock independent of the CPU/peripheral hardware clock (f_{CLK}) selectable

Remark n: Unit number (n = 0)

Figure 13 - 1 shows a block diagram of the serial interface UARTAn and **Table 13 - 1** shows the pin configuration of the serial interface UARTAn.

Figure 13 - 1 Block Diagram of Serial Interface UARTAn

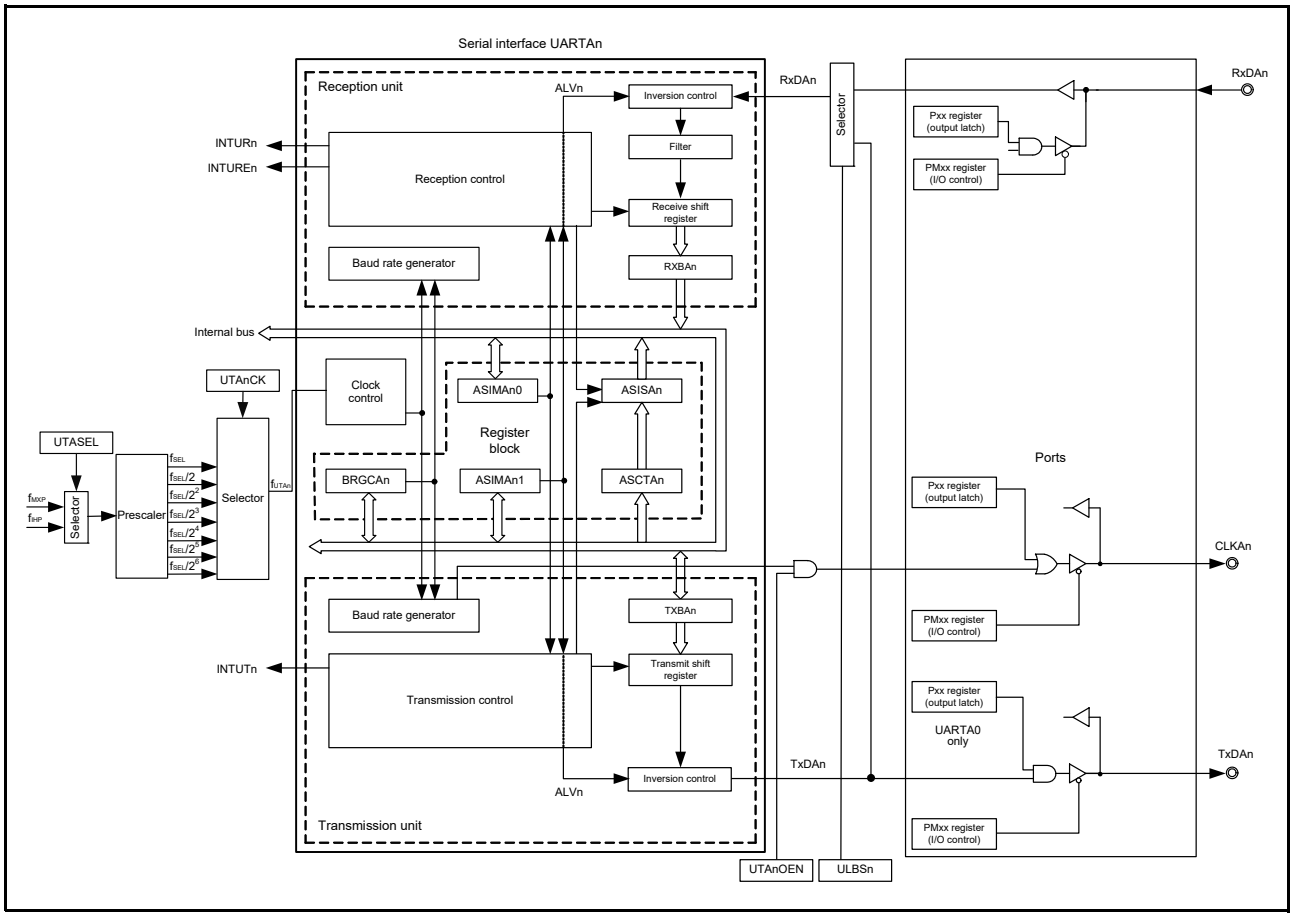


Table 13 - 1 Pin Configuration of Serial Interface UARTAn

Name	I/O	Function
RxDAn	Input	Serial data input signal
TxDAn	Output	Serial data output signal
CLKAn	Output	Serial clock output signal

Remark 1. f_{MXP}: High-speed peripheral clock frequency
f_{IHP}: High-speed on-chip oscillator peripheral clock frequency
f_{SEL}: Selected clock to be divided for the UARTA
f_{UTAn}: Serial interface UARTAn operation clock

Remark 2. n = 0

13.2 Registers for Controlling the Serial Interface UARTA

The following registers are used to control the serial interface UARTA.

- Peripheral enable register 1 (PER1)
- Transmit buffer register n (TXBAn)
- Receive buffer register n (RXBAn)
- Operation mode setting register n0 (ASIMAn0)
- Operation mode setting register n1 (ASIMAn1)
- Baud rate generator control register n (BRGCAn)
- Status register n (ASISAn)
- Status clear trigger register n (ASCTAn)
- UARTA clock select register 0 (UTA0CK)
- UART loopback select register (ULBS)
- Port mode registers (PMxx)
- Port registers (Pxx)
- Port input mode registers (PIMxx)
- Port output mode registers (POMxx)
- Port mode control A registers (PMCAxx)

Remark n: Unit number (n = 0)

13.2.1 Peripheral enable register 1 (PER1)

The PER1 register is used to enable or disable the supply of a clock signal to various on-chip peripheral modules. Clock supply to an on-chip peripheral module that is not to be used can be stopped to decrease power consumption and noise. To use the serial interface UARTAn, be sure to set the UTAEN bit to 1. The PER1 register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 13 - 2 Format of Peripheral Enable Register 1 (PER1)

Address: F00FAH
After reset: 00H
R/W: R/W

Symbol	7	6	5	<4>	3	<2>	1	0
PER1	0	0	0	TML32EN	0	UTAEN **Note**	0	0

UTAEN	Control of supply of an input clock to the serial interface UARTAn
0	Stops supply of an input clock. • The SFRs used by the serial interface UARTAn cannot be written.**Note**
1	Enables supply of an input clock. • The SFRs used by the serial interface UARTAn can be read and written.

Note When an SFR is read, 00H is returned regardless of its setting.

Caution 1. Be sure to leave bits 7 to 5, 3, 1, and 0 set to 0.

Caution 2. Make sure that the setting of the UTAEN bit is 1 before using the serial interface UARTAn. If UTAEN = 0, writing to the registers which control the serial interface UARTAn is ignored.

Remark n: Unit number (n = 0)

13.2.2 Transmit buffer register n (TXBAn)

TXBAn is a buffer register for setting transmit data. Transmission starts by writing data for transmission to the TXBAn register. The TXBAn register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is FFH.

Figure 13 - 3 Format of Transmit Buffer Register n (TXBAn)

Address: F0300H (TXBA0)
After reset: FFH
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
TXBAn								

Bits 7 to 0	Function
—	• When a character length of 8 bits is specified: Data in bits 7 to 0 of TXBAn are transferred. • When a character length of 7 bits is specified: Data in bits 6 to 0 of TXBAn are transferred in either MSB- or LSB-first mode; bit 7 is invalid. • When a character length of 5 bits is specified: Data in bits 4 to 0 of TXBAn are transferred in either MSB- or LSB-first mode; bits 7 to 5 are invalid.

Caution 1. When the TXBFAn flag of the ASISAn register is 1, do not write data for transmission to the TXBAn register.

Caution 2. After setting the TXEAn bit of the ASIMAn0 register to 1, wait for the period of at least one cycle of the serial interface UARTAn operation clock (f_{UTAn}) before setting the first data for transmission in the TXBAn register. If data for transmission is set within one cycle of the serial interface UARTAn operation clock after the TXEAn bit is set to 1, the start of transmission is delayed by one cycle of the serial interface UARTAn operation clock.

Remark 1. Data is transferred from the TXBAn register to this register, and is then transmitted as serial data through the TxDAn pin. In the first transmission, data is transferred from the TXBAn register to this register immediately after data is written to the TXBAn register. In continuous transmission, data is transferred after transmission of one frame and just before generation of the transfer completion interrupt. The transmit shift register cannot be manipulated directly by a program.

Remark 2. n: Unit number (n = 0)

13.2.3 Receive buffer register n (RXBAn)

The RXBAn register holds the parallel data converted by the receive shift register. Every time one byte of data is received, the next receive data is transferred from the receive shift register**Note** to this register. The RXBAn register can be read by an 8-bit memory manipulation instruction. The value of this register following a reset is FFH.

Note The receive shift register converts the serial data that is input through the RxDAn pin to parallel data. The receive shift register cannot be manipulated directly by a program.

Figure 13 - 4 Format of Receive Buffer Register n (RXBAn)

Address: F0301H (RXBA0)
After reset: FFH
R/W: R

Symbol	7	6	5	4	3	2	1	0
RXBAn								

Bits 7 to 0	Function
—	• When a character length of 8 bits is specified: Receive data is transferred to bits 7 to 0 of this register. • When a character length of 7 bits is specified: Receive data is transferred to bits 6 to 0 of this register in either MSB- or LSB-first mode; bit 7 is always 0. • When a character length of 5 bits is specified: Receive data is transferred to bits 4 to 0 of this register in either MSB- or LSB-first mode; bits 7 to 5 are always 0.

Caution **If an overrun error (OVEAn) occurs, the data received at that time are not stored in the RXBAn register.**

Remark n: Unit number (n = 0)

13.2.4 Operation mode setting register n0 (ASIMAn0)

The ASIMAn0 register is an 8-bit register that controls serial communication of the serial interface UARTAn. The ASIMAn0 register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 01H.

Figure 13 - 5 Format of Operation Mode Setting Register n0 (ASIMAn0)

Address: F0302H (ASIMA00)
After reset: 01H
R/W: R/W

Symbol	<7>	<6>	<5>	4	3	2	<1>	<0>
ASIMAn0	UARTAENn	TXEAn	RXEAn	0	0	0	ISSMAn	ISRMAn

UARTAENn Note 1	UART operation enable
0	Disables the UART operation clock. Resets the internal circuits.Note 2
1	Enables the UART operation clock.

TXEAn	Transmission enable
0	Disables transmission. (Resets the transmission circuit.)
1	Enables transmission.

RXEAn	Reception enable
0	Disables reception. (Resets the reception circuit.)
1	Enables reception.

ISSMAn	Transmit interrupt mode select
0	The INTUTn interrupt is generated on completion of transmission.
1	The INTUTn interrupt is generated when the transmit buffer becomes empty. (for continuous transmission)

ISRMAn	Receive interrupt mode select
0	The INTUREn interrupt is generated when a reception error occurs. (INTURn is not generated.)
1	The INTURn interrupt is generated when a reception error occurs. (INTUREn is not generated.)

Note 1. When UARTAENn = 0, the level being output from the TxDAn pin and the level being input from the RxDAn pin are determined according to the setting of the ALVn bit as described below.
- When ALVn = 0, output from the TxDAn pin is high.
- When ALVn = 1, output from the TxDAn pin is low.

Note 2. The ASISAn and RXBAn registers are reset by clearing the UARTAENn bit to 0.

Caution 1. Be sure to leave bits 4 to 2 set to 0.

Caution 2. To start transmission, set the UARTAENn bit to 1 and then set the TXEAn bit to 1. To stop transmission, clear the TXEAn bit to 0 and then clear the UARTAENn bit to 0.

Caution 3. To start reception, set the UARTAENn bit to 1 and then set the RXEAn bit to 1. To stop reception, clear the RXEAn bit to 0 and then clear the UARTAENn bit to 0.

(Cautions and Remark are listed on the next page.)

Caution 4. Follow the procedure below when setting the UARTAENn bit to 1 and then setting the RXEAn bit to 1.
- **When ALVn = 0, the setting must be made while the level being input to the RxDAn pin is high. Otherwise, reception will start at that point and a framing error may occur.**
- **When ALVn = 1, the setting must be made while the level being input to the RxDAn pin is low. Otherwise, reception will start at that point and a framing error may occur.**

Caution 5. The TXEAn and RXEAn bits are synchronized with the serial interface UARTAn operation clock (f_{UTAn}). To enable transmission or reception again, set the TXEAn or RXEAn bit to 1 at least two cycles of the serial interface UARTAn operation clock after clearing the TXEAn or RXEAn bit to 0. If the bit is set to 1 within two cycles of the serial interface UARTAn operation clock after the clearing, the transmission or reception circuit may not be able to be initialized.

Caution 6. After setting TXEAn bit to 1, wait for at least one cycle of the serial interface UARTAn operation clock (f_{UTAn}) before setting the transmit data in the TXBAn register.

Caution 7. Clear the RXEAn bit to 0 before modifying the ISRMAn bit.

Remark n: Unit number (n = 0)

13.2.5 Operation mode setting register n1 (ASIMAn1)

The ASIMAn1 register is an 8-bit register that controls serial communication of the serial interface UARTAn. The ASIMAn1 register can be set by a 1-bit or 8-bit memory manipulation instruction. The ASIMAn1 register must be modified while TXEAn = 0 and RXEAn = 0. The value of this register following a reset is 1AH.

Figure 13 - 6 Format of Operation Mode Setting Register n1 (ASIMAn1)

Address: F0303H (ASIMA01)
After reset: 1AH
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
ASIMAn1	0	PSn1	PSn0	CLn1	CLn0	SLn	DIRn	ALVn

PSn1	PSn0	Transmission/reception parity bit setting 1, 0	
		Transmission	Reception
0	0	No parity bit is output.	Data is received without parity.
0	1	0 parity is output.	Data is received with 0 parity.Note
1	0	Odd parity is output.	Check is made for odd parity.
1	1	Even parity is output.	Check is made for even parity.

CLn1	CLn0	Transmission/reception character length setting 1, 0
0	0	Character length of data = 5 bits
0	1	Setting prohibited
1	0	Character length of data = 7 bits
1	1	Character length of data = 8 bits

SLn	Transmission stop bit length setting
0	Stop bit length = 1 bit
1	Stop bit length = 2 bits

DIRn	Transmission/reception order setting
0	MSB first
1	LSB first

ALVn	Transmission/reception level setting
0	Positive logic (wait state = high level, start bit = low level, stop bit = high level)
1	Negative logic (wait state = low level, start bit = high level, stop bit = low level)

Note When “Data is received with 0 parity” is set, parity check is not performed. Accordingly the PEAn bit of the ASISAn register is not set: no reception error interrupts are generated.

Caution 1. Clear both the TXEAn and RXEAn bits to 0 before modifying the ASIMAn1 register.

Caution 2. Reception is always handled as including a stop bit. The setting of the SLn bit does not affect reception.

Remark n: Unit number (n = 0)

13.2.6 Baud rate generator control register n (BRGCAn)

The BRGCAn register sets the frequency divisor for the 8-bit counter in the serial interface UARTAn. The BRGCAn register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is FFH.

Figure 13 - 7 Format of Baud Rate Generator Control Register n (BRGCAn)

Address: F0304H (BRGCA0)
After reset: FFH
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
BRGCAn								

Bits 7 to 0	Function
—	Controls the UART baud rate (serial transfer speed). For details on the settings, see **Table 13 - 2**.

Caution Modify the BRGCAn bits while the TXEAn and RXEAn bits are 0 (in the transmission/reception stopped state).

Table 13 - 2 BRGCAn Settings

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	k	Selection of 8-bit counter output clock
0	0	0	0	0	0	0	×	×	Setting prohibited
0	0	0	0	0	0	1	0	2	$f_{UTAn}/2$
0	0	0	0	0	0	1	1	3	$f_{UTAn}/3$
:	:	:	:	:	:	:	:	:	:
1	1	1	1	1	1	0	0	252	$f_{UTAn}/252$
1	1	1	1	1	1	0	1	253	$f_{UTAn}/253$
1	1	1	1	1	1	1	0	254	$f_{UTAn}/254$
1	1	1	1	1	1	1	1	255	$f_{UTAn}/255$

Caution The baud rate is one half the frequency of the output clock signal from the 8-bit counter.

Remark 1. k: The value set with the BRGCAn register (k = 2, 3, 4, …, 255)
Remark 2. ×: Don't care.
For an example of the baud rate setting, see **13.3.4 Baud rate generator, 3. Baud rate calculation, c) Baud rate setting example**.
Remark 3. n: Unit number (n = 0)

13.2.7 Status register n (ASISAn)

The ASISAn register indicates the states in terms of errors and transmission on completion of reception by the serial interface UARTAn. It consists of three error flag bits (PEAn, FEAn, and OVEAn) and two transmission status flag bits (TXBFAn and TXSFAn). The ASISAn register is read-only and can be read by an 8-bit memory manipulation instruction. The value of this register following a reset is 00H. The PEAn, FEAn, and OVEAn bits are initialized by clearing the UARTAENn or RXEAn bit to 0. These bits are also cleared by writing to the corresponding bit of the ASCTAn register. The TXBFAn and TXSFAn bits are initialized by clearing the UARTAENn or TXEAn bit to 0.

Figure 13 - 8 Format of Status Register n (ASISAn) (1/2)

Address: F0305H (ASISA0)
After reset: 00H
R/W: R

Symbol	7	6	5	4	3	2	1	0
ASISAn	0	0	TXBFAn	TXSFAn	0	PEAn	FEAn	OVEAn

TXBFAn	Transmit buffer data flag
—	[Setting condition] • Data is written to the TXBAn register. (Data exists in the TXBAn register.) [Clearing conditions] • The UARTAENn or TXEAn bit is cleared to 0. • Data is transferred to the transmit shift register.

TXSFAn	Transmit shift register data flag
—	[Setting condition] • Data is transferred from the TXBAn register. (Data is being transmitted.) [Clearing conditions] • The UARTAENn or TXEAn bit is cleared to 0. • Data is transferred from the transmit shift register and then no subsequent data is transferred from the TXBAn register.

PEAn	Parity error flag
—	[Setting condition] • The parity of the received data does not match the parity bit. [Clearing conditions] • The UARTAENn or RXEAn bit is cleared to 0. • 1 is written to the PECTAn bit.

FEAn	Framing error flag
—	[Setting condition] • A stop bit is not detected when receiving data. [Clearing conditions] • The UARTAENn or RXEAn bit is cleared to 0. • 1 is written to the FECTAn bit.

Figure 13 - 8 Format of Status Register n (ASISAn) (2/2)

OVEAn	Overrun error flag
—	[Setting condition] • The next reception is completed before the receive data in the RXBAn register is read. [Clearing conditions] • The UARTAENn or RXEAn bit is cleared to 0. • 1 is written to the OVECTAn bit.

Caution 1. For continuous transmission, be sure to check that the TXBFAn flag is 0 after writing the first transmit data (the first byte) to the TXBAn register and then write the next transmit data (the second byte) to the TXBAn register. Otherwise, the transmit data become undefined. However, the TXBFAn flag need not be checked when continuous transmission is performed by using the buffer empty interrupt (ISSMAn bit = 1).

Caution 2. When initializing the transmission unit (TXEAn = 0) after completion of continuous transmission, be sure to check that the TXSFAn flag is 0 after the transfer completion interrupt is generated, and then initialize the unit. Otherwise, the transmit data become undefined.

Caution 3. The operation of the PEAn bit depends on the setting of the PSn1 and PSn0 bits of the ASIMAn1 register.

Caution 4. For the receive data, only the first 1 bit of the stop bits is checked regardless of the stop bit length.

Caution 5. When an overrun error occurs, the next receive data is not written to the RXBAn register and discarded.

Remark n: Unit number (n = 0)

13.2.8 Status clear trigger register n (ASCTAn)

The ASCTAn register sets the trigger to clear the states in terms of errors on completion of reception of the serial interface UARTAn. It contains 3 bits of the error clear trigger flags (PECTAn, FECTAn, and OVECTAn). The ASCTAn register can be written by an 8-bit or 1-bit memory manipulation instruction. When the ASCTAn register is read, 00H is always read. The value of this register following a reset is 00H. Writing 1 to the PECTAn, FECTAn, and OVECTAn bits clears the PEAn, FEAn, and OVEAn bits of the ASISAn register, respectively. When writing 0, the corresponding error flags are not cleared.

Figure 13 - 9 Format of Status Clear Trigger Register n (ASCTAn)

Address: F0306H (ASCTA0)
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	<2>	<1>	<0>
ASCTAn	0	0	0	0	0	PECTAn	FECTAn	OVECTAn

PECTAn **Note**	Parity error flag clear trigger
0	Does not clear the PEAn flag (the flag is retained).
1	Clears the PEAn flag.

FECTAn **Note**	Framing error flag clear trigger
0	Does not clear the FEAn flag (the flag is retained).
1	Clears the FEAn flag.

OVECTAn **Note**	Overrun error flag clear trigger
0	Does not clear the OVEAn flag (the flag is retained).
1	Clears the OVEAn flag.

Note Reading this bit returns 0.

Caution **After writing 1 to the trigger bit, the corresponding error flag is cleared on the next rising edge of the operating clock (f_{UTAn}). Accordingly, if reading the ASISAn register immediately after writing 1 to the trigger bit, the corresponding error flag may not have been cleared yet.**

Remark n: Unit number (n = 0)

13.2.9 UARTA clock select register 0 (UTA0CK)

The UTA0CK register selects the operating clock of the serial interface UARTA0. The UTASEL1 and UTASEL0 bits select the clock source, f_{SEL}, for the serial interface UARTA0 from f_{MXP} and f_{IH}. The bits from UTA0CK3 to UTA0CK0 select the operating clock for the serial interface UARTA0 from $f_{SEL}/1$ to $f_{SEL}/64$. Setting the UTA0OEN bit to 1 selects the UARTA0 clock output mode, and the operating clock for UARTA0 is output from the CLKA0 pin. The UTA0CK register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 13 - 10 Format of UARTA Clock Select Register 0 (UTA0CK)

Address: F0310H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
UTA0CK	UTA0OEN	0	UTASEL1	UTASEL0	UTA0CK3	UTA0CK2	UTA0CK1	UTA0CK0

UTA0OEN **Note**	Serial interface UARTA0 clock output enable
0	Disables CLKA0 output.
1	Enables CLKA0 output.

UTASEL1	UTASEL0	f_{SEL} clock select
0	0	Stop
0	1	f_{MXP}
1	0	f_{IHP}
1	1	Setting prohibited

UTA0CK3	UTA0CK2	UTA0CK1	UTA0CK0	Serial interface UARTA0 operation clock select (f_{UTA0})
0	0	0	0	f_{SEL}
0	0	0	1	$f_{SEL}/2$
0	0	1	0	$f_{SEL}/4$
0	0	1	1	$f_{SEL}/8$
0	1	0	0	$f_{SEL}/16$
0	1	0	1	$f_{SEL}/32$
0	1	1	0	$f_{SEL}/64$
Other than above				Setting prohibited

Note Set this bit to 0 in the 20- to 48-pin products because the CLKA0 output pin is not present in the given products.

Caution **This register should be read or written when the TXEA0 and RXEA0 bits are 0 (in the transmission/reception stopped state).**

Remark f_{MXP}: High-speed peripheral clock
f_{IHP}: High-speed on-chip oscillator peripheral clock
f_{SEL}: Selected clock to be divided for the UARTA

13.2.10 UART loopback select register (ULBS)

The ULBS register is used to enable the UART loopback function. This register has a bit to control loopback for each UART. Setting a bit to 1 enables the loopback function for the given UART, with the output from the transmit shift register being looped back to the receive shift register. The ULBS register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 13 - 11 Format of the UART Loopback Select Register (ULBS)

Address: F0079H
After reset: 00H
R/W: R/W

Symbol	7	6	5	<4>	3	<2>	<1>	<0>
ULBS	0	0	0	ULBS4	0	ULBS2	ULBS1	ULBS0

ULBS4	Selection of the UART loopback function
0	The state of the RxDA0 pin of the serial interface UARTA0 is input to the receive shift register.
1	The output from the transmit shift register is looped back to the receive shift register.

Caution Be sure to leave bits 7 to 5 and 3 set to 0.

13.2.11 Registers for controlling the port functions multiplexed with the inputs and outputs of the UARTA serial interface

Set the following registers to control the port functions multiplexed with the inputs and outputs of the UARTA serial interface.

- Port mode registers (PMxx)
- Port registers (Pxx)
- Port input mode registers (PIMxx)
- Port output mode registers (POMxx)
- Port mode control A registers (PMCAxx)

For details, see the following sections.

- **4.3.1 Port mode registers (PMxx)**
- **4.3.2 Port registers (Pxx)**
- **4.3.4 Port input mode registers (PIMxx)**
- **4.3.5 Port output mode registers (POMxx)**
- **4.3.7 Port mode control A registers (PMCAxx)**

For examples of register settings for controlling port functions, see **4.5.4 Examples of register settings for port and alternate functions**. When connecting an external device operating at a different voltage (1.8 V, 2.5 V, or 3 V), see **4.4.5 Communications with devices operating at a different voltage (1.8 V, 2.5 V, or 3 V) by switching I/O buffers**.

13.3 Operation

The serial interface UARTAn operates in either of the following two modes.

- Operation stop mode
- UART mode

13.3.1 Operation stop mode

In the operation stop mode, serial communication is not performed, and thus the power consumption can be reduced. In addition, in this mode, the pins can be used as ordinary port pins. To set the operation stop mode, clear bits 7 to 5 (UARTAENn, TXEAn, RXEAn) of the ASIMAn0 register to 0.
The bus clock is not stopped by the above setting. To completely stop operation, clear the UTAEN bit of the PER1 register to 0 after the above setting.

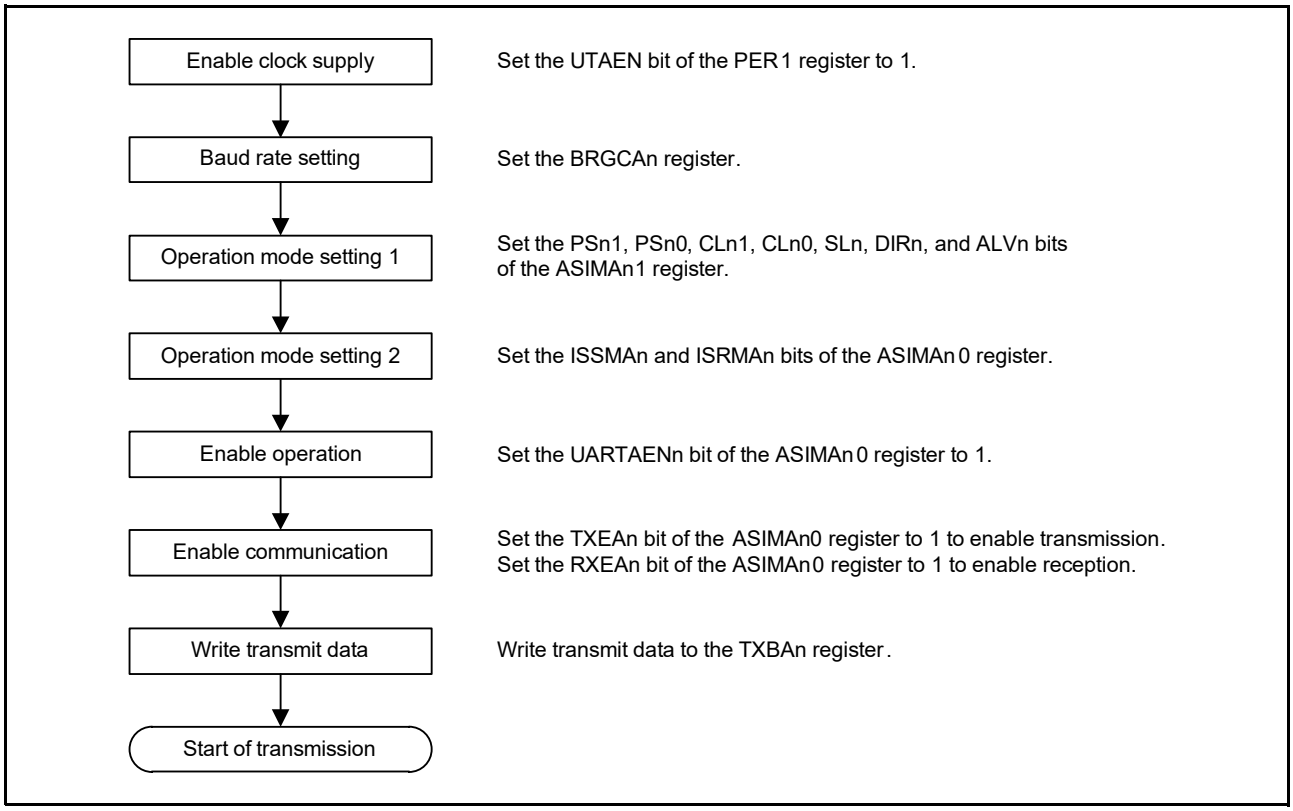
13.3.2 UART mode

In this mode, one byte of data is transmitted and one byte is received following the start bit. That is, operation is full duplex. A dedicated UART baud rate generator is incorporated, so that communication can be executed at a wide range of baud rates.

Remark n: Unit number (n = 0)

1. Communication procedure
   Figure 13 - 12 shows the flowchart of communication procedure.

Figure 13 - 12 Flowchart of Communication Procedure



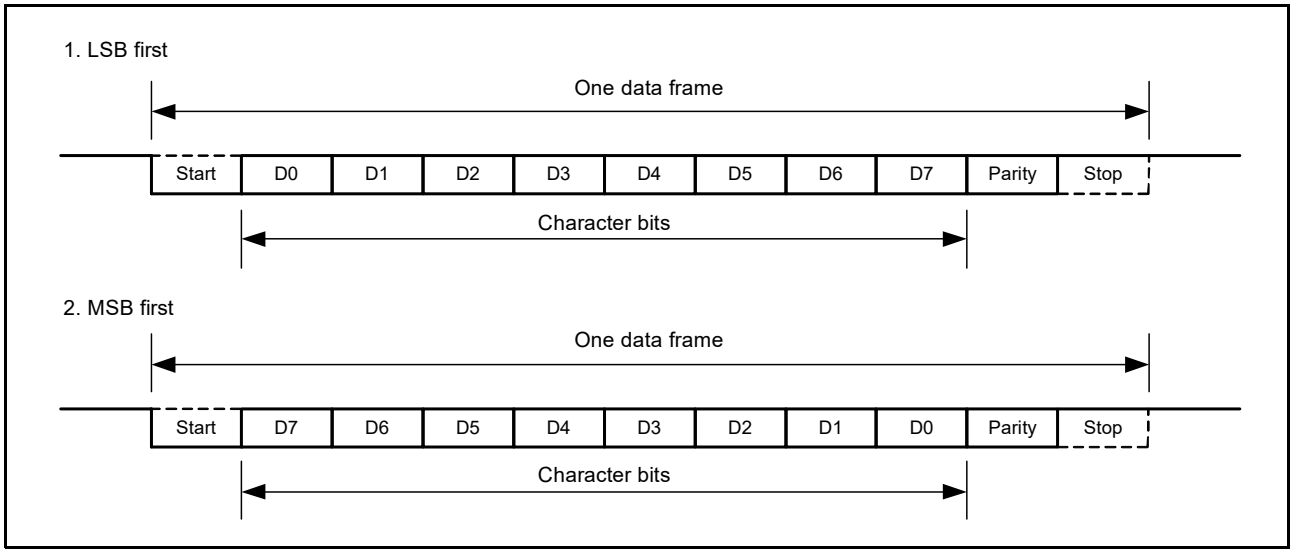
Caution **When a pin is to be used for UART reception, use the port-related register to set the given pin to input mode. When a pin is to be used for UART transmission, use the port register (Pxx) for the given pin to set the initial output level and then use the port-related register to set the pin to output mode. Make these settings before enabling transmission. The port-related registers to be used for the settings stated above differ with the channel and product to be used. For further details on the settings, see 4.5 Register Settings When Using Alternate Function. When connecting an external device operating at a different voltage (1.8 V, 2.5 V or 3 V), see 4.4.5 Communications with devices operating at a different voltage (1.8 V, 2.5 V, or 3 V) by switching I/O buffers.**

Remark n: Unit number (n = 0)

2. Format and waveform example of transmit/receive data

The following describes the communication data format of the serial interface UARTAn. **Figure 13 - 13** shows the data format.

Figure 13 - 13 Transmit/Receive Data Format



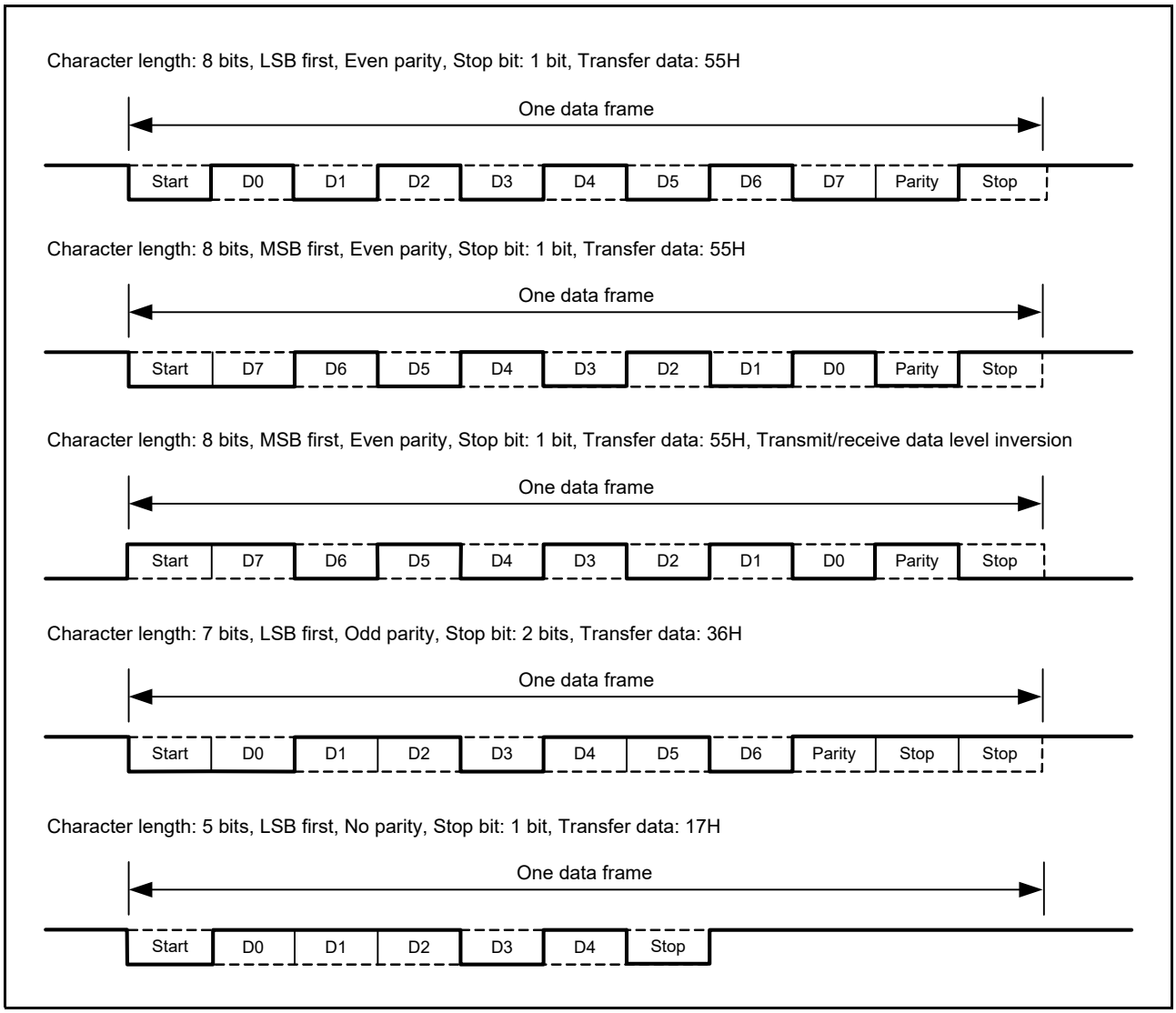
One data frame consists of the following bits.

- Start bit: 1 bit
- Character bits: 5, 7 or 8 bits
- Parity bit: Even parity, odd parity, 0 parity, or no parity
- Stop bit: 1 or 2 bits

The character bit length, the parity, the stop bit length, the transfer direction (LSB/MSB first), and the TxDAn pin output (direct/inverted) in one data frame are specified by the ASIMAn1 register.

Figure 13 - 14 shows the examples of transmit/receive data waveforms.

Figure 13 - 14 Example of Transmit/Receive Data Waveform



3. Parity types and operation

The parity bit is used to detect a bit error in communication data. Usually, the same type of parity bit is used on both the transmitting and reception sides. With even/odd parity, a 1-bit (odd number) error can be detected. With zero/no parity, an error cannot be detected.

a) Even parity

- In transmission

  Data for transmission, including the parity bit, are controlled so that an even number of bits have the value 1. The value of the parity bit is set as follows.

  If the data for transmission have an odd number of bits with the value 1: 1

  If the data for transmission have an even number of bits with the value 1: 0

- In reception

  In the data for reception, including the parity bit, the number of bits with the value 1, is counted. If it is odd, a parity error occurs.

b) Odd parity

- In transmission

  Unlike even parity, data for transmission, including the parity bit, are controlled so that an odd number of bits have the value 1.

  If the data for transmission have an odd number of bits with the value 1: 0

  If the data for transmission have an even number of bits with the value 1: 1

- In reception

  In the data for reception, including the parity bit, the number of bits with the value 1, is counted. If it is even, a parity error occurs.

c) 0 parity

The parity bit is cleared to 0 when data is transmitted, regardless of the transmit data. The parity bit is not detected when the data is received. Therefore, a parity error does not occur regardless of whether the parity bit is 0 or 1.

d) No parity

No parity bit is appended to the transmit data. Reception is performed assuming that there is no parity bit. A parity error does not occur, because there is no parity bit.

4. Normal transmission

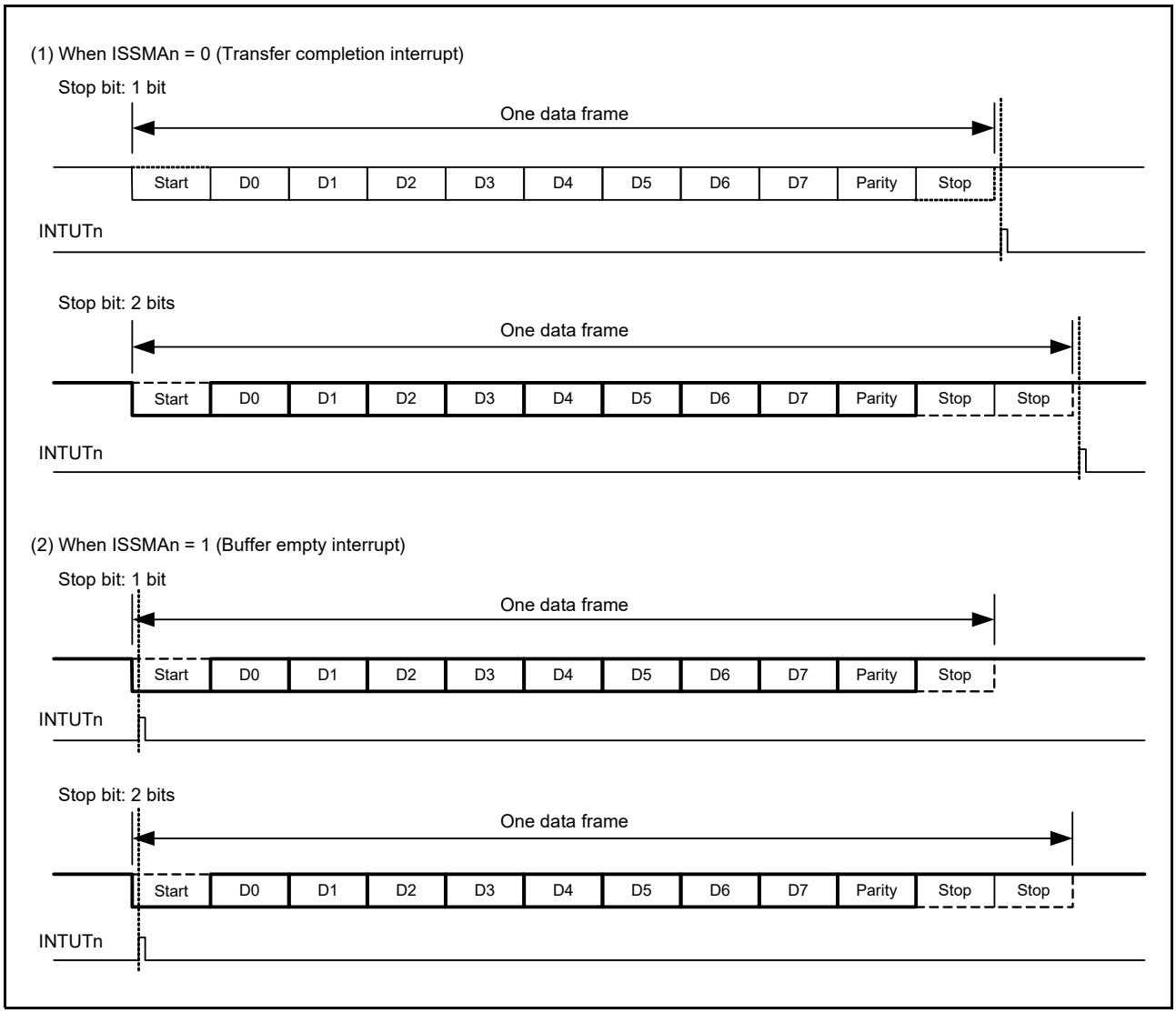
Transmission is enabled by setting the UARTAENn bit of the operation mode setting register n0 (ASIMAn0) to 1 and then setting the TXEAn bit of the ASIMAn0 register to 1. Transmission can be started by writing the data for transmission to the transmit buffer register n (TXBAn). The start bit, parity bit, and stop bit are automatically appended to the data. When transmission is started, the data in the TXBAn register are transferred to the transmit shift register. After that, the transmit data are sequentially output from the transmit shift register to the TxDAn pin in the specified transfer direction. When transmission is completed, the parity and stop bits which are set by the ASIMAn0 register are appended and a transfer completion interrupt request signal (INTUTn) is generated. Transmission is suspended until the next transmit data is written to the TXBAn register.

Figure 13 - 15 shows the timing of the transfer completion interrupt request signal (INTUTn). INTUTn is issued at the following timing.

a) When ISSMAn = 0 (INTUTn functions as a transfer completion interrupt.)
INTUTn is issued after the output of the last stop bit.

b) ISSMAn = 1 (INTUTn functions as a buffer empty interrupt.)
INTUTn is issued when the start bit is output.

Figure 13 - 15 Interrupt Output Timing



5. Continuous transmission

The serial interface UARTAn has two separate registers for continuous transmission: the transmit buffer register n (TXBAn) and the transmit shift register. At the moment the transmit shift register starts a shift operation, the next transmit data can be written to the transmit buffer register n (TXBAn). This operation enables continuous transmission, thereby improving communication rate. Note that continuous transmission is not achieved when writing to the TXBAn register is not completed within the maximum number of clock cycles defined below from generation of the buffer empty interrupt.

Maximum number of clock cycles = Data transfer length × 2k - (2k + 3)
k: The value set with the BRGCAn bits (k = 2, 3, 4, …, 255)

An example of calculating the maximum number of clock cycles is described below.
When the BRGCAn register = 02H (k = 2), start bit = 1 bit, character length = 8 bits, parity used, and stop bit = 1 bit:
The maximum number of clock cycles = Transfer length × 2k - (2k + 3) = 11 × 2 × 2 - (2 × 2 + 3) = 37
(Writing must be completed within 37 cycles of the serial interface UARTAn operating clock (f_{UTAn}).)

Continuous transmission is achieved by the following two methods.

a) Continuous transmission by polling

Continuous transmission is achieved by polling the transmit buffer data flag (TXBFAn bit) and the transmit shift register data flag (TXSFAn bit) of the status register n (ASISAn). When using this method, clear the ISSMAn bit of the operation mode setting register n0 (ASIMAn0) to 0.

- At the start of and during continuous transmission

  At the start of continuous transmission, write the first byte of data to the TXBAn register, check that the transmit buffer data flag (TXBFAn) is 0, and then write the second byte of data. In a similar way, check that the TXBFAn flag is 0 and then write the subsequent data to the TXBAn register.

TXBFAn	Determination flag indicating that writing to TXBAn is enabled or disabled at the start of continuous transmission
0	Writing is enabled.
1	Writing is disabled.

Caution To determine if continuous transmission is enabled or disabled, only check the TXBFAn flag. The TXSFAn flag must not be used for the determination in combination with this flag.

- Completion of continuous transmission

  In continuous transmission, when data in the transmit shift register and the TXBAn register are transmitted after the required number of transmit data are written to the TXBAn register, the continuous transmission is completed. To confirm the completion, check the setting of the transmit shift register data flag (TXSFAn).

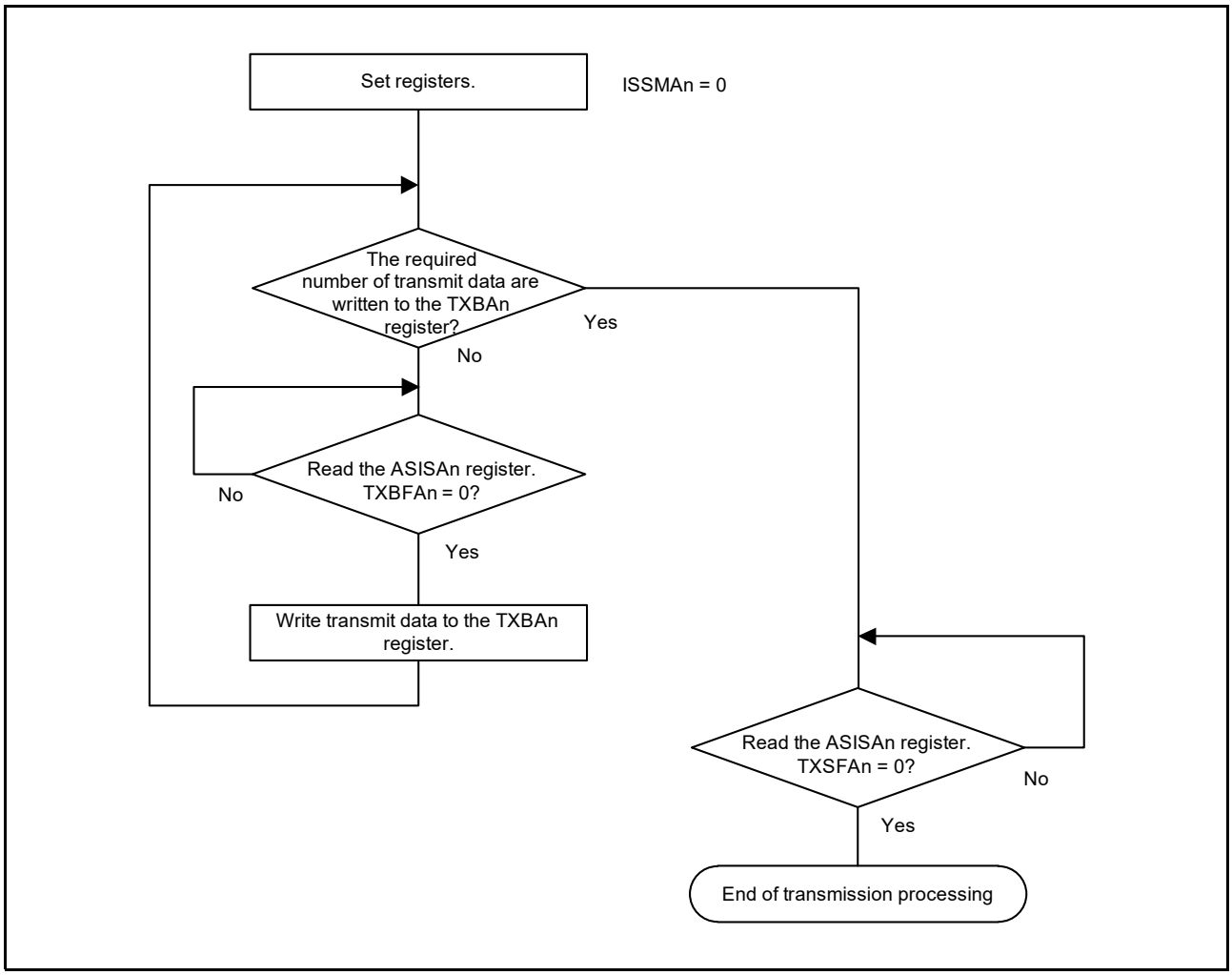
TXSFAn	Confirmation flag indicating whether transmission is in progress or not
0	Transmission is completed.
1	Transmission is in progress.

Caution 1. When initializing the transmission unit after completion of continuous transmission, check that the TXSFAn flag is 0 after the transfer completion interrupt is generated, and then initialize the unit.

Caution 2. During continuous transmission, after transmission of one data frame, the subsequent transmission may be completed before execution of the INTUTn interrupt processing. This can be detected by incorporating the program that counts the number of transmit data and by referencing the TXSFAn flag.

Figure 13 - 16 shows a flow example of continuous transmission processing by polling.

Figure 13 - 16 Flow Example of Continuous Transmission Processing by Polling



b) Continuous transfer by using an interrupt

Continuous transmission is achieved by using the interrupt (INTUTn). An interrupt can be generated when data in the transmit buffer register n (TXBAn) are transferred to the transmit shift register by setting the ISSMAn bit to 1 in the operation mode setting register n0 (ASIMAn0). With this setting, continuous transmission is enabled by writing data to the TXBAn register on occurrence of the buffer empty interrupt. In addition, the transfer completion interrupt can be generated on completion of continuous transmission by clearing the ISSMAn bit to 0 after writing the last transmit data to the TXBAn register.

Figure 13 - 17 shows a flow example of continuous transmission using interrupt.

Figure 13 - 17 Flow Example of Continuous Transmission Using Interrupt

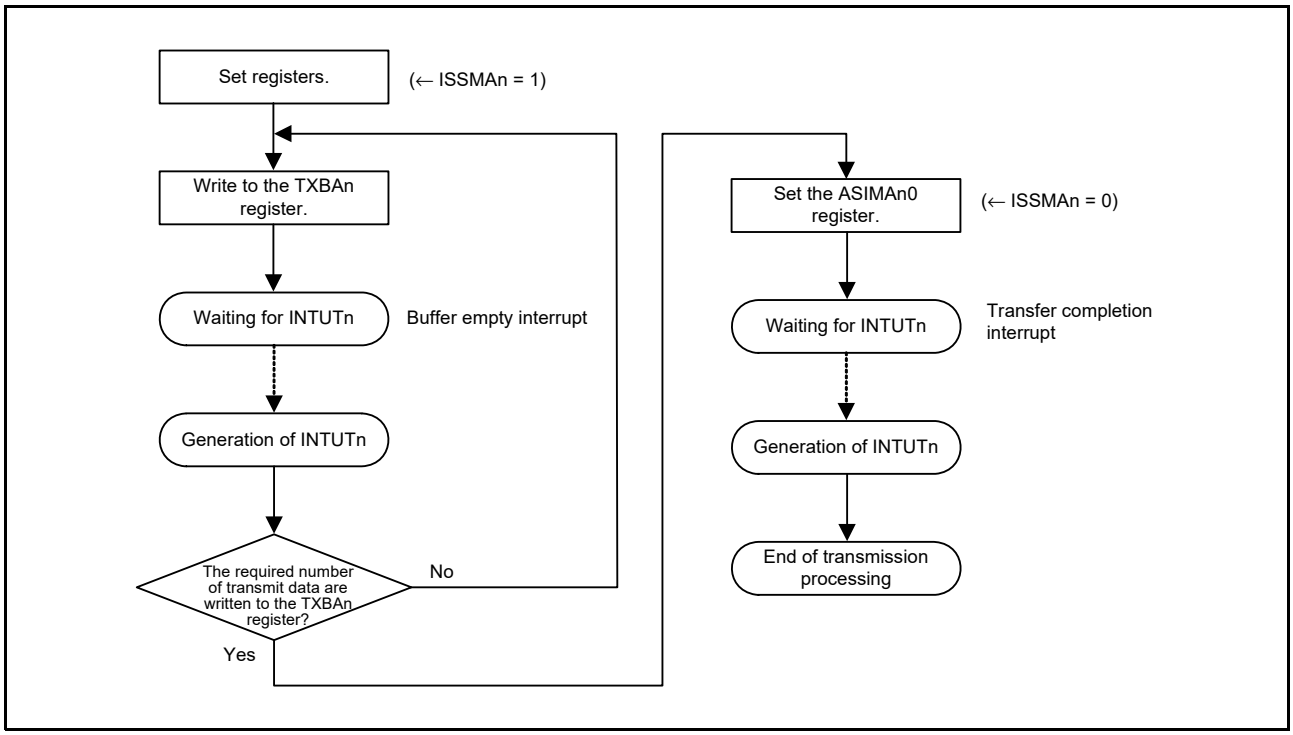
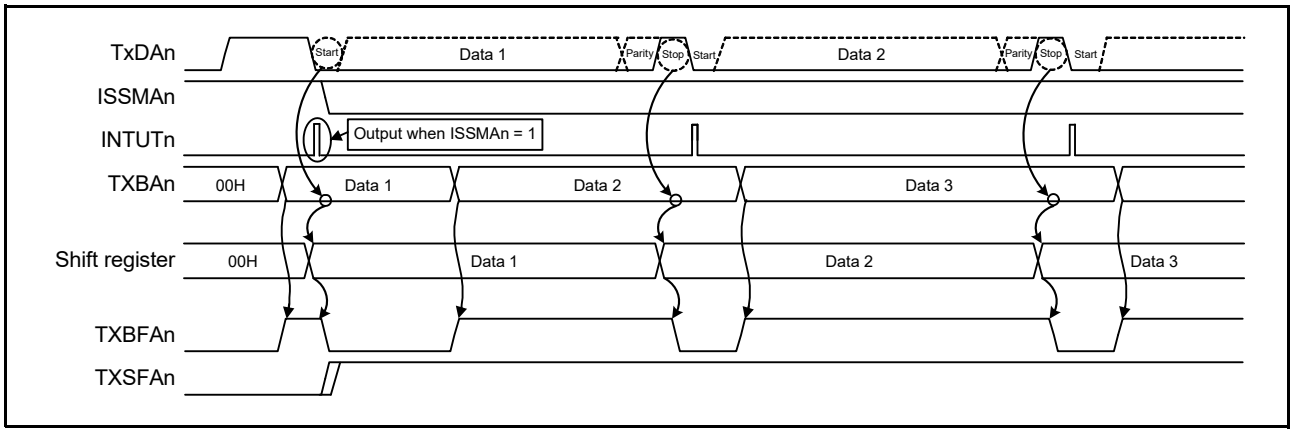


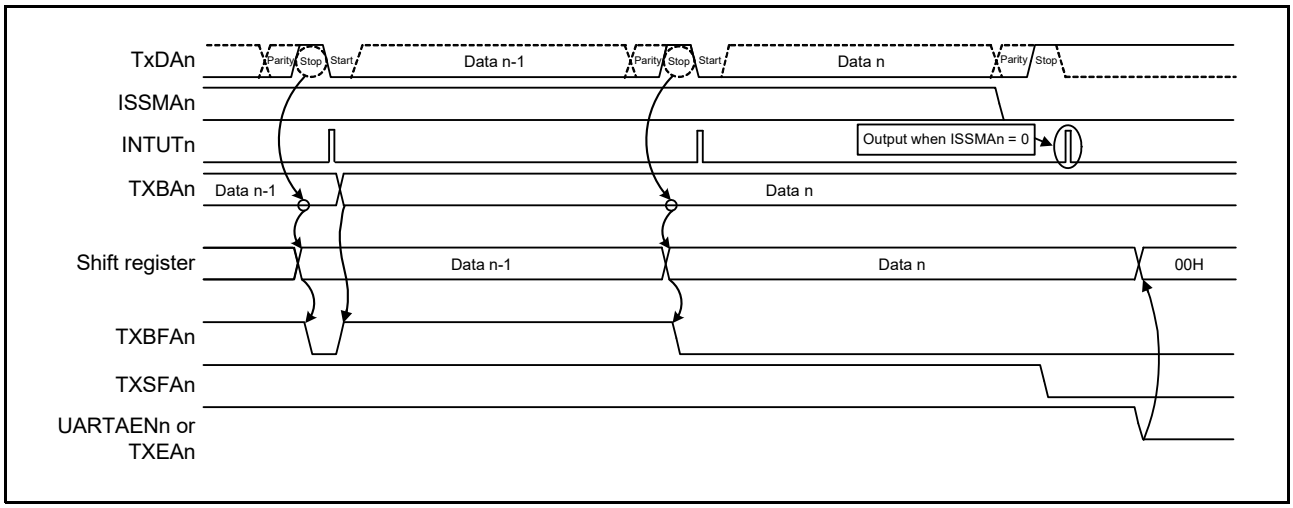
Figure 13 - 18 and **Figure 13 - 19** show the timing charts when the continuous transmission is started and completed, respectively.

Figure 13 - 18 Timing Chart When Continuous Transmission Is Started



Caution When the ASISAn register is read, the value of the TXBFAn and TXSFAn flags is 11B over certain periods. Accordingly, use only the TXBFAn flag to determine if writing is enabled or disabled.

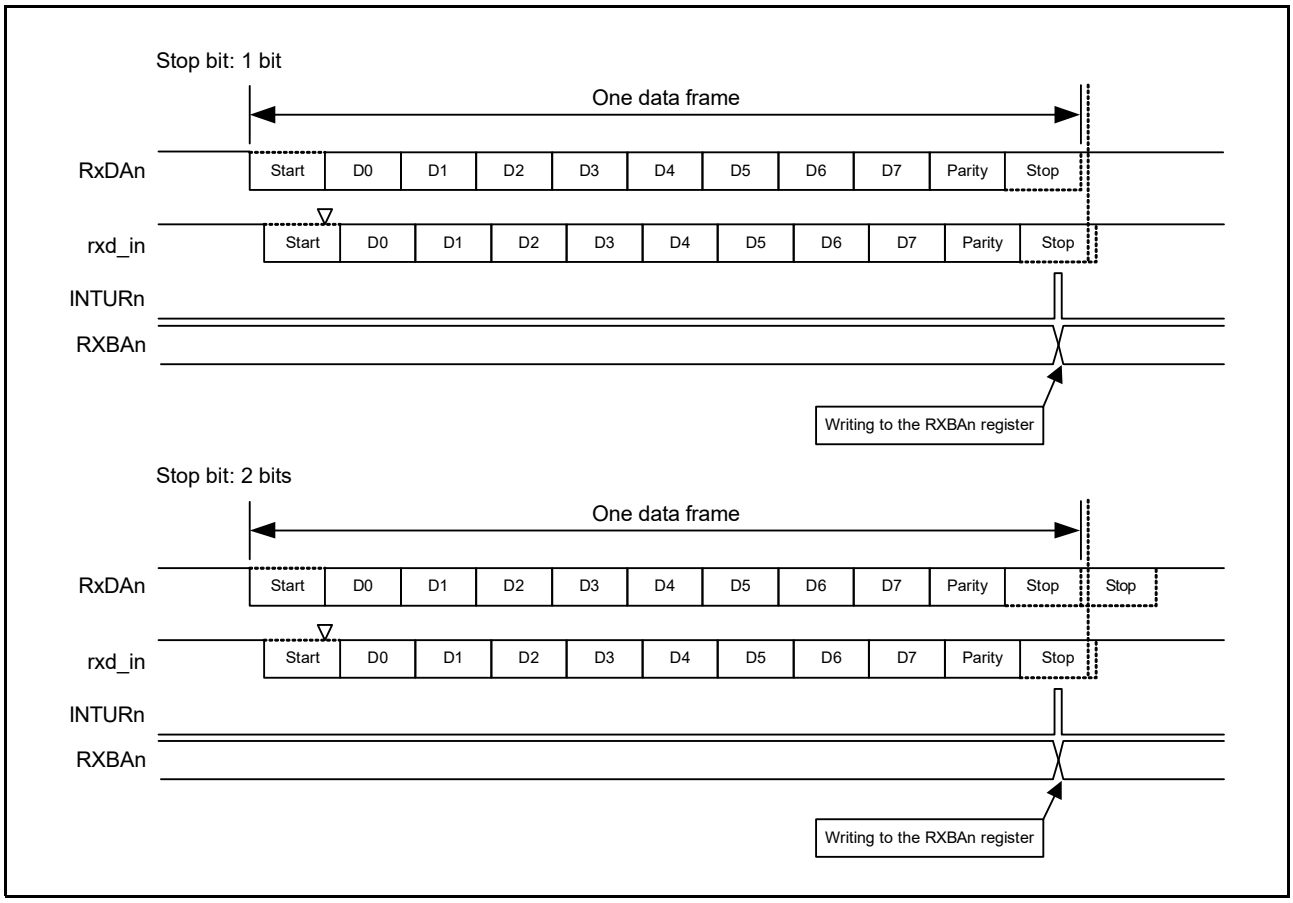
Figure 13 - 19 Timing Chart When Continuous Transmission Is Completed



6. Normal reception

When setting the UARTAENn bit of the operation mode setting register n0 (ASIMAn0) to 1 and then setting the RXEAn bit of the ASIMAn0 register to 1, reception is enabled, and sampling of the input to the RxDAn pin is performed. When the ALVn bit is 0, the 8-bit counter of the baud rate generator starts counting on detection of the falling edge on the RxDAn pin. When the counter reaches the set value of the baud rate generator control register n (BRGCAn), the input to the RxDAn pin is sampled again (at the point indicated with ∇ in **Figure 13 - 20**). If the RxDAn pin is low, it is regarded as a start bit. When the ALVn bit is 1, the 8-bit counter of the baud rate generator starts counting on detection of the rising edge on the RxDAn pin. When the counter reaches the set value of the baud rate generator control register n (BRGCAn), the input to the RxDAn pin is sampled again (at the point indicated with ∇ in **Figure 13 - 20**). If the RxDAn pin is high, it is regarded as a start bit. **Figure 13 - 20** shows the timing chart of receive operation. On detection of a start bit, receive operation is started: serial data is sequentially stored in the receive shift register at a specified baud rate. On reception of a stop bit, the transfer completion interrupt (INTURn) is generated, and at the same time, the data in the receive shift register is written to the receive buffer register n (RXBAn). Note that when an overrun error (OVEAn) occurs, the data received on occurrence of the error is not written to the RXBAn register. When a parity error (PEAn) or a framing error (FEAn) occurs during reception, reception continues until a stop bit is received. After completion of the reception, the reception error interrupt (INTURn/INTUREn) set in the ISRMAn bit is generated. When a reception error occurs, read the status register n (ASISAn) and then read the receive buffer register n (RXBAn) to clear the error flag. If the receive buffer register n (RXBAn) is not read, an overrun error will occur when the next data is received: the reception error state will continue. Reception is always handled as including a stop bit. Accordingly, the second stop bit is ignored.

Figure 13 - 20 Timing of UART Receive Operation



Remark 1. rxd_in is the internal signal generated by latching RxDAn with a noise filter. rxd_in is delayed relative to RxDAn by maximum of 3 cycles of the UART operation clock.

Remark 2. The INTUR output timing in the figure is just an example. The timing relative to RxDAn varies according to the setting of the BRGCAn register.

7. Reception error

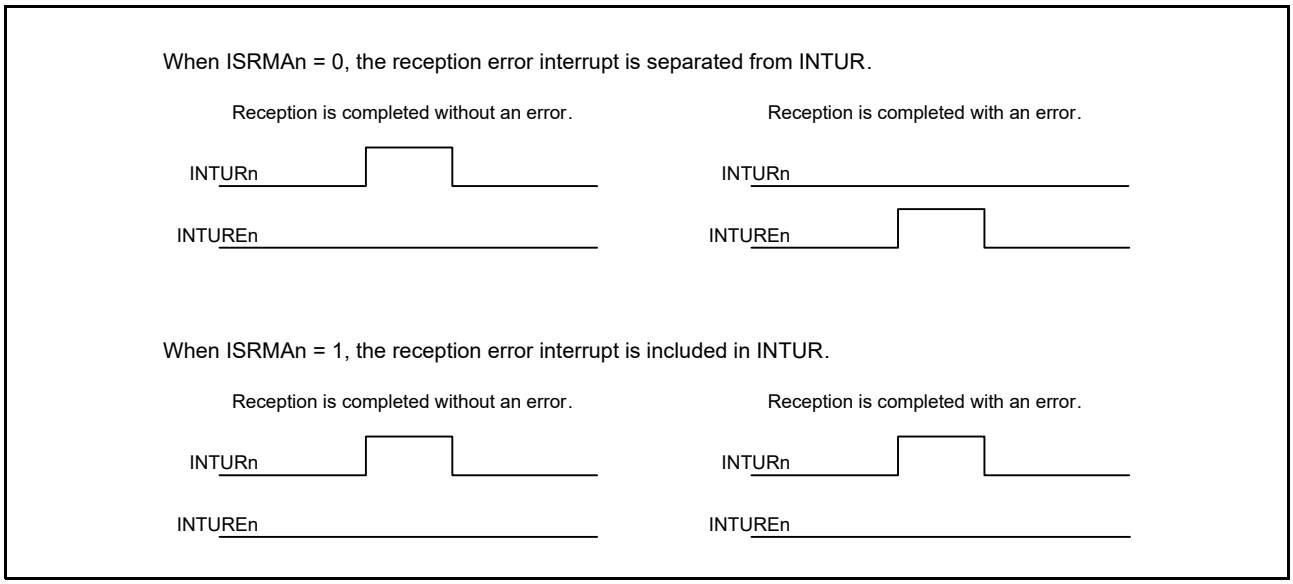
Three types of errors may occur during reception; parity error, framing error, and overrun error. When these errors occur, the corresponding error flag in the status register n (ASISAn) is set, and the reception error interrupt request signal (INTURn or INTUREn) is generated. The type of the reception error can be identified by the reception error interrupt processing routine, which reads and checks the contents of the status register n (ASISAn). The contents of the ASISAn register are cleared to 0 by setting the corresponding bit of the status clear trigger register n (ASCTAn) to 1. **Table 13 - 3** shows the causes of the reception errors.

Table 13 - 3 Causes of Reception Errors

Error Flag	Reception Error	Cause
PEAn	Parity error	The parity specified for reception does not match the parity of receive data.
FEAn	Framing error	No stop bit is detected.
OVEAn	Overrun error	Before the receive data is read from the receive buffer, the next data reception is completed.

Setting the ISRMAn bit of the operation mode setting register n0 (ASIMAn0) to 0 allows the reception error interrupt to be separated from INTURn and allows it to be generated as INTUREn. **Figure 13 - 21** shows the interrupt output waveform which varies depending on the setting of the ISRMAn bit.

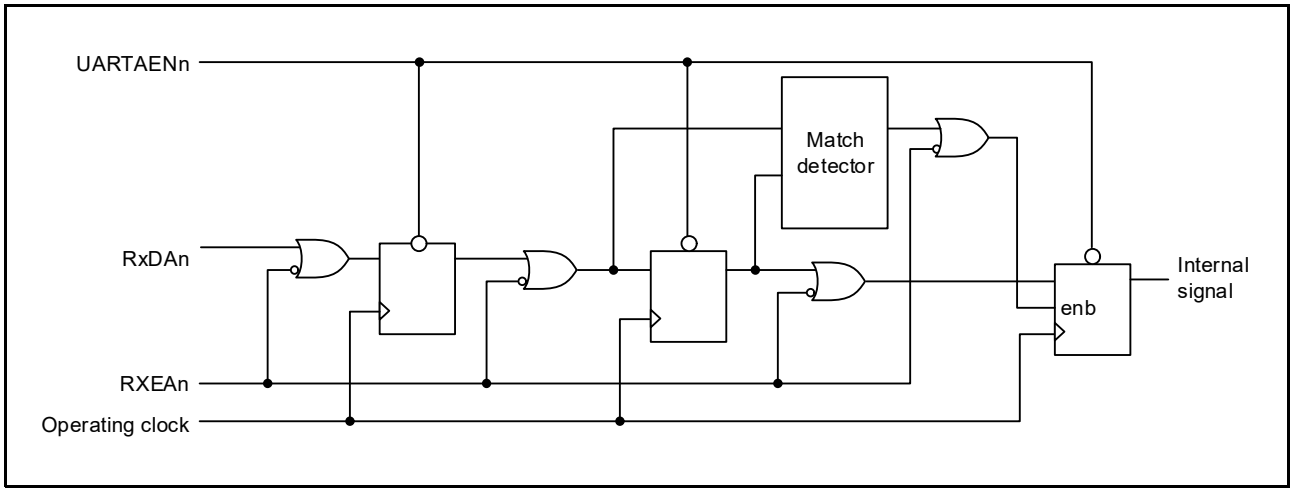
Figure 13 - 21 Various Interrupt Output Waveforms Depending on ISRMAn Setting



13.3.3 Receive data noise filter

This filter samples the receive data (RxDAn), and determines the level when the same level is sampled twice. The receive data is delayed by maximum of three cycles of the operating clock because of the circuit configuration. **Figure 13 - 22** shows the noise filter circuit.

Figure 13 - 22 Noise Filter



Caution 1. When ALVn = 0 (wait state = high level; start bit = low level), the initial value of the receive data (RxDAn) needs to be "high".

Caution 2. When ALVn = 1 (wait state = low level; start bit = high level), the initial value of the receive data (RxDAn) needs to be "low".

Remark n: Unit number (n = 0)

13.3.4 Baud rate generator

The baud rate generator consists of 8-bit programmable counters, and generates a serial clock for transmission/reception of the serial interface UARTAn. An 8-bit counter is provided each for transmission and reception.

1. Configuration of baud rate generator
   a) Serial interface UARTAn operation clock
      When the UARTAENn bit is 1 in the operation mode setting register n0 (ASIMAn0), the serial interface UARTAn operation clock (f_{UTAn}) is supplied to each module. When the UARTAENn bit is 0, the serial interface UARTAn operation clock is fixed to low level.

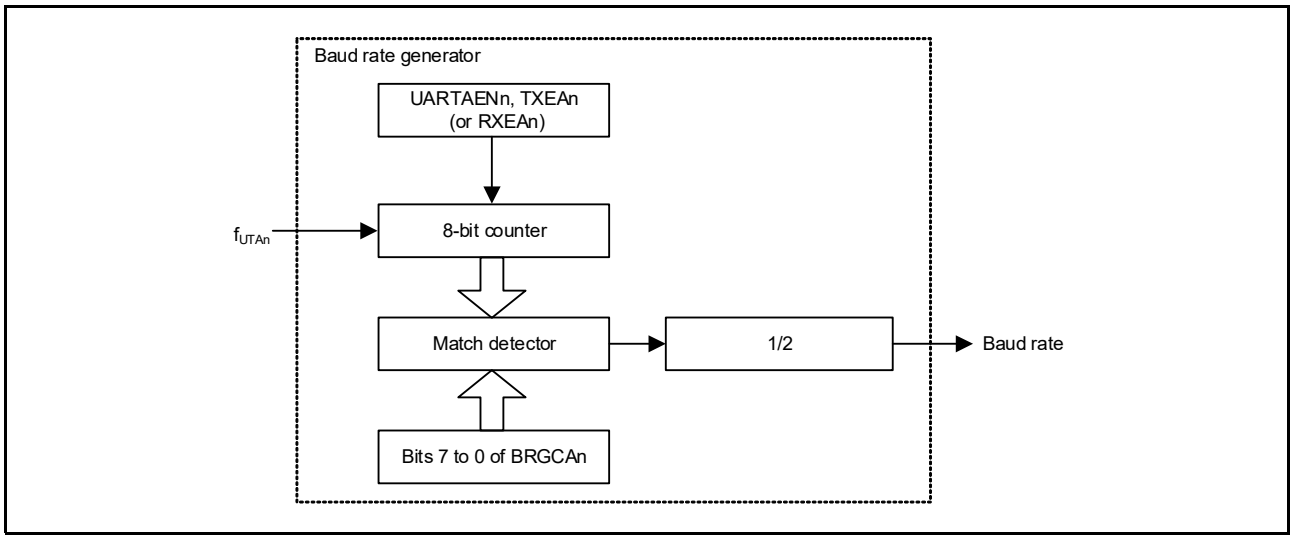
   b) Transmission counter
      This counter is cleared to 0 and stops when the UARTAENn bit is 0 or the TXEAn bit is 0 in the operation mode setting register n0 (ASIMAn0). It starts counting when the UARTAENn bit is 1 and the TXEAn bit is 1. The counter is cleared to 0 when the first transmit data is written to the transmit buffer register n (TXBAn). When continuous transmission is performed, the counter is cleared to 0 again when transmission of one frame of data has been completed. If there is no data to be transmitted next, the counter is not cleared to 0 and continues counting until the UARTAENn or TXEAn bit is cleared to 0. When the UARTAENn bit is 0 or the TXEAn bit is 0 in the ASIMAn0 register, the counter stops at 00H.

   c) Reception counter
      This counter is cleared to 0 and stops when the UARTAENn bit is 0 or the RXEAn bit is 0 in the operation mode setting register n0 (ASIMAn0). It starts counting when the start bit is detected. The counter stops operation after one frame has been received, until the next start bit is detected. When the UARTAENn bit is 0 or the RXEAn bit is 0 in the ASIMAn0 register, the counter stops at 00H.

Remark n: Unit number (n = 0)

Figure 13 - 23 shows the configuration of the baud rate generator.

Figure 13 - 23 Configuration of Baud Rate Generator



2. Generation of serial clock
   A serial clock to be generated can be specified by using the baud rate generator control register n (BRGCAn). The baud rate generator divides the frequency of the input clock signal to the 8-bit counter (f_{UTAn}) by the divisor set by the BRGCAn register. The result of this division is further divided by 2 to produce the serial clock.

3. Baud rate calculation

   a) Baud rate calculation expression
   The baud rate can be calculated by the following expression.
   Baud rate = $f_{UTAn} \div (2 \times k)$ [bps]

   f_{UTAn}: Frequency of operating clock
   k: Value set by bits 7 to 0 of the BRGCAn register (k = 2, 3, 4, …, 255)

   b) Baud rate error
   The baud rate error can be calculated by the following expression.

$$\text{Error} = \left(\frac{\text{Actual baud rate (baud rate with error)}}{\text{Desired baud rate (correct baud rate)}} - 1 \right) \times 100\ [\%]$$

Caution 1. Keep the baud rate error during transmission to within the permissible error range on the reception side.

Caution 2. Make sure that the baud rate error during reception satisfies the permissible baud rate error range during reception. Permissible baud rate error during reception is described in 13.3.4 Baud rate generator, 3. Baud rate calculation, d) Permissible baud rate range during reception.

c) Baud rate setting example

Table 13 - 4 Set Data of Baud Rate Generator (1/2)

Desired baud rate	In operation with fIHP = 32 MHz (UTASEL1 and UTASEL0 = 10B)															
	No division (UTAnCK3 to UTAnCK0 = 0000B)		×1/2 (UTAnCK3 to UTAnCK0 = 0001B)		×1/4 (UTAnCK3 to UTAnCK0 = 0010B)		×1/8 (UTAnCK3 to UTAnCK0 = 0011B)		×1/16 (UTAnCK3 to UTAnCK0 = 0100B)		×1/32 (UTAnCK3 to UTAnCK0 = 0101B)		×1/64 (UTAnCK3 to UTAnCK0 = 0110B)	
	k	Error from the desired baud rate	k	Error from the desired baud rate	k	Error from the desired baud rate	k	Error from the desired baud rate	k	Error from the desired baud rate	k	Error from the desired baud rate	k	Error from the desired baud rate
200 bps	Disabled		Disabled		Disabled		Disabled		Disabled		Disabled		Disabled	
300 bps	Disabled		Disabled		Disabled		Disabled		Disabled		Disabled		Disabled	
600 bps	Disabled		Disabled		Disabled		Disabled		Disabled		Disabled		Disabled	
1200 bps	Disabled		Disabled		Disabled		Disabled		Disabled		Disabled		208	0.16%
2400 bps	Disabled		Disabled		Disabled		Disabled		Disabled		208	0.16%	104	0.16%
4800 bps	Disabled		Disabled		Disabled		Disabled		208	0.16%	104	0.16%	52	0.16%
9600 bps	Disabled		Disabled		Disabled		208	0.16%	104	0.16%	52	0.16%	26	0.16%
19200 bps	Disabled		Disabled		208	0.16%	104	0.16%	52	0.16%	26	0.16%	13	0.16%
38400 bps	Disabled		208	0.16%	104	0.16%	52	0.16%	26	0.16%	13	0.16%	Disabled	
76800 bps	208	0.16%	104	0.16%	52	0.16%	26	0.16%	13	0.16%	Disabled		Disabled	
115200 bps	139	–0.08%	69	0.64%	35	–0.79%	17	2.12%	Disabled		Disabled		Disabled	
153600 bps	104	0.16%	52	0.16%	26	0.16%	13	0.16%	Disabled		Disabled		Disabled	

Remark k: Value set by bits 7 to 0 of the baud rate generator control register n (BRGCAn) (k = 2, 3, 4, ..., 255)
n: Unit number (n = 0)

Table 13 - 4 Set Data of Baud Rate Generator (2/2)

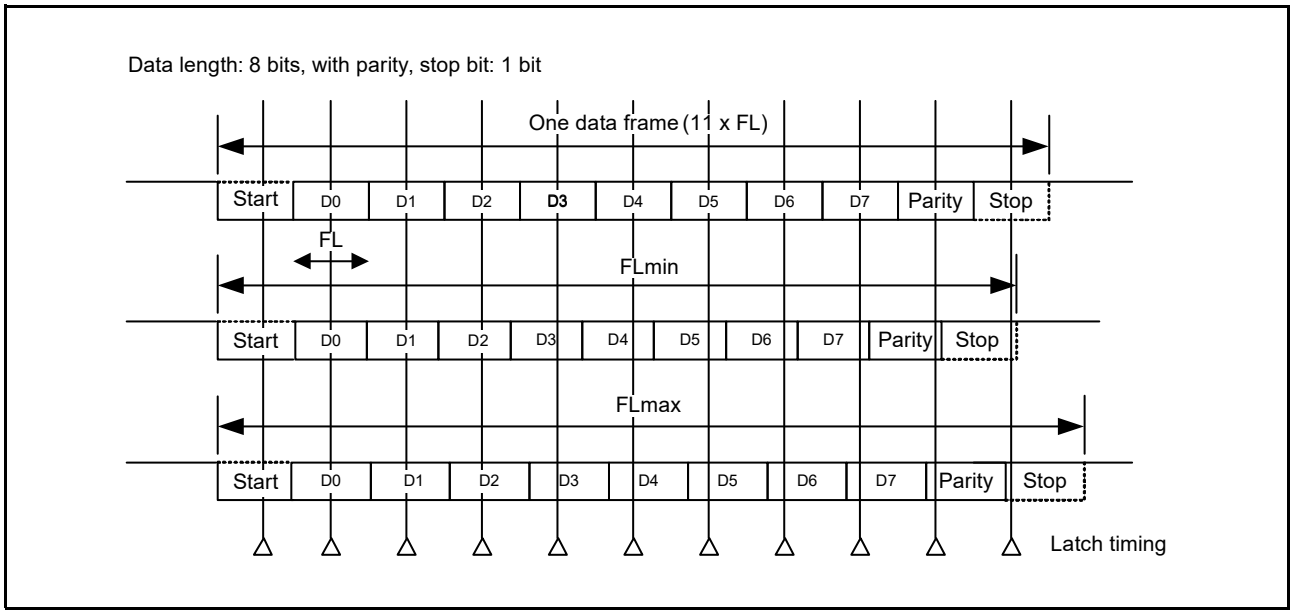
Desired baud rate	In operation with f_{MXP} = 20 MHz (UTASEL1 and UTASEL0 = 01B)															
	No division (UTAnCK3 to UTAnCK0 = 0000B)		×1/2 (UTAnCK3 to UTAnCK0 = 0001B)		×1/4 (UTAnCK3 to UTAnCK0 = 0010B)		×1/8 (UTAnCK3 to UTAnCK0 = 0011B)		×1/16 (UTAnCK3 to UTAnCK0 = 0100B)		×1/32 (UTAnCK3 to UTAnCK0 = 0101B)		×1/64 (UTAnCK3 to UTAnCK0 = 0110B)	
	k	Error from the desired baud rate	k	Error from the desired baud rate	k	Error from the desired baud rate	k	Error from the desired baud rate	k	Error from the desired baud rate	k	Error from the desired baud rate	k	Error from the desired baud rate
200 bps	Disabled		Disabled		Disabled		Disabled		Disabled		Disabled		Disabled	
300 bps	Disabled		Disabled		Disabled		Disabled		Disabled		Disabled		Disabled	
600 bps	Disabled		Disabled		Disabled		Disabled		Disabled		Disabled		255	2.12%
1200 bps	Disabled		Disabled		Disabled		Disabled		Disabled		255	2.12%	130	0.16%
2400 bps	Disabled		Disabled		Disabled		Disabled		255	2.12%	130	0.16%	65	0.16%
4800 bps	Disabled		Disabled		Disabled		255	2.12%	130	0.16%	65	0.16%	33	–1.36%
9600 bps	Disabled		Disabled		255	2.12%	130	0.16%	65	0.16%	33	–1.36%	16	1.73%
19200 bps	Disabled		255	2.12%	130	0.16%	65	0.16%	33	–1.36%	16	1.73%	8	1.73%
38400 bps	255	2.12%	130	0.16%	65	0.16%	33	–1.36%	16	1.73%	8	1.73%	4	1.73%
76800 bps	130	0.16%	65	0.16%	33	–1.36%	16	1.73%	8	1.73%	4	1.73%	Disabled	
115200 bps	87	–0.22%	43	0.94%	22	–1.36%	11	–1.36%	Disabled		Disabled		Disabled	
153600 bps	65	0.16%	33	–1.36%	16	1.73%	8	1.73%	4	1.73%	Disabled		Disabled	

Remark k: Value set by bits 7 to 0 of the baud rate generator control register n (BRGCAn) (k = 2, 3, 4, ..., 255)
n: Unit number (n = 0)

d) Permissible baud rate range during reception

Figure 13 - 24 shows the permissible error from the baud rate on the transmitting side during reception.

Figure 13 - 24 Permissible Baud Rate Range during Reception



Caution Be sure to make settings so that the baud rate error during reception is within the permissible error range. Use the calculation expression below to check if the error is within the permissible range.

After the start bit is detected, the latch timing of receive data is determined by the counter specified with the baud rate generator control register n (BRGCAn). If the whole frame including the stop bit has been received before this latching, reception can proceed correctly. Assuming that 11 bits of data are received, the theoretical values can be calculated as follows.

- The relation between 1-bit data length and baud rate
  FL = (Brate) - 1
  Brate: Baud rate of UART
  k: Set value of the BRGCAn register
  FL: 1-bit data length
  Margin of latch timing: 2 clock cycles

- Minimum permissible data frame length (FLmin)

$$k = 3 \text{ to } 255: FLmin = 11 \times FL - \frac{k - 2}{2k} \times FL = \frac{21k + 2}{2k} FL$$

- Maximum permissible baud rate for reception on the transmitting side (BRmax)

$$k = 2: BRmax = Brate + \frac{1}{22k} Brate$$

$$k = 3 \text{ to } 255: BRmax = (FLmin/11)^{-1} = \frac{22k}{21k + 2} Brate$$

- Maximum permissible data frame length (FLmax)

$$k = 3 \text{ to } 255: FLmax = \frac{21k + 2}{20k} FL \times 11$$

- Minimum permissible baud rate for reception on the transmitting side (BRmin)

$$k = 2: BRmin = Brate - \frac{1}{22k} Brate$$

$$k = 3 \text{ to } 255: BRmin = (FLmax/11)^{-1} = \frac{20k}{21k - 2} Brate$$

Table 13 - 5 shows the permissible baud rate errors between the UART and the transmitting side that can be calculated from the above expressions for the minimum and maximum baud rates.

Table 13 - 5 Maximum/Minimum Permissible Baud Rate Error

Division ratio (k)	Maximum permissible baud rate error	Minimum permissible baud rate error
2	+2.27%	–2.27%
4	+2.33%	–2.44%
8	+3.53%	–3.61%
20	+4.27%	–4.31%
50	+4.56%	–4.58%
100	+4.66%	–4.67%
255	+4.72%	–4.73%

Remark 1. The permissible error of reception depends on the number of bits in one frame, input clock frequency, and division ratio (k). The higher the input clock frequency and the division ratio (k), the higher the permissible error.

Remark 2. k: Set value of the BRGCAn register
n: Unit number (n = 0)

13.4 Points for Caution when the Serial Interface UARTA is to be Used

13.4.1 Port setting for the RxDAn pin

When ALVn = 0 (wait state = high level, start bit = low level), the initial value of receive data (RxDAn) must be high.
When ALVn = 1 (wait state = low level, start bit = high level), the initial value of receive data (RxDAn) must be low.
Accordingly, port setting is required for the RxDAn pin before setting UARTAENn = 1.

Section 14 Interrupt Functions

The interrupt function switches the program execution to other processing. When the branch processing is finished, the program returns to the interrupted processing. The number of interrupt sources differs, depending on the product.

		20-pin	32-pin	44- and 48-pin	64-pin
Maskable interrupts	External	6	9	10	13
	Internal	23	26	26	26

14.1 Interrupt Function Types

The following two types of interrupt functions are used.

1. Maskable interrupts
   These interrupts undergo mask control. Maskable interrupts can be divided into four priority groups by setting the priority specification flag registers (PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR03L, PR10L, PR10H, PR11L, PR11H, PR12L, PR12H, PR13L). Multiple interrupt servicing can be applied to low-priority interrupts when high-priority interrupts are generated. If two or more interrupt requests, each having the same priority, are simultaneously generated, then they are processed according to the default priority of vectored interrupt servicing. For default priority, see **Table 14 - 1**. A standby release signal is generated and STOP, HALT, and SNOOZE modes are released. External interrupt requests and internal interrupt requests are provided as maskable interrupts.
2. Software interrupt
   This is a vectored interrupt generated by executing the BRK instruction. It is acknowledged even when interrupts are disabled. The software interrupt does not undergo interrupt priority control.

14.2 Interrupt Sources and Configuration

Interrupt sources include maskable interrupts and a software interrupt. In addition, they also have up to seven reset sources (see **Table 14 - 1**). The vector codes that store the program start address when branching due to the generation of a reset or various interrupt requests are two bytes each, so interrupts jump to a 64 K address of 00000H to 0FFFFH.

Table 14 - 1 Interrupt Source List (1/3)

Interrupt Type	Default Priority[Note 1]	Interrupt Source		Internal/External	Vector Table Address	Basic Configuration Type[Note 2]	64-pin	48-pin	44-pin	32-pin	20-pin
		Name	Trigger								
Maskable	0	INTWDTI	Watchdog timer interval[Note 3] (75% of overflow time + 1/4fIL)	Internal	00004H	<a>	✓	✓	✓	✓	✓
	1	INTLVI	Voltage detection[Note 4]		00006H		✓	✓	✓	✓	✓
	2	INTP0	Pin input edge detection	External	00008H	<b>	✓	✓	✓	✓	✓
	3	INTP1			0000AH		✓	✓	✓	✓	—
	4	INTP2			0000CH		✓	✓	✓	✓	—
	5	INTP3			0000EH		✓	✓	✓	✓	✓
	6	INTP4			00010H		✓	✓	✓	✓	—
	7	INTP5			00012H		✓	✓	✓	✓	✓
	8	INTST2/ INTCSI20/ INTIIC20	UART2 transmission transfer end or buffer empty interrupt/CSI20 transfer end or buffer empty interrupt/IIC20 transfer end	Internal	00014H	<a>	✓	✓	✓	✓	Note 5
	9	INTSR2/ INTCSI21/ INTIIC21	UART2 reception transfer end/CSI21 transfer end or buffer empty interrupt/IIC21 transfer end		00016H		✓	✓	✓	Note 6	Note 6
	10	INTSRE2	UART2 reception communication error occurrence		00018H		✓	✓	✓	✓	✓
	13	INTST0/ INTCSI00/ INTIIC00	UART0 transmission transfer end or buffer empty interrupt/CSI00 transfer end or buffer empty interrupt/IIC00 transfer end		0001EH		✓	✓	✓	✓	✓
	14	INTTM00	End of counting or capturing by channel 0 of timer array unit 0		00020H		✓	✓	✓	✓	✓
	15	INTSRE0	UART0 reception communication error occurrence		00022H		✓	✓	✓	✓	✓
		INTTM01H	End of counting or capturing by channel 1 of timer array unit 0 (at higher 8-bit timer operation)				✓	✓	✓	✓	✓
	16	INTST1/ INTCSI10/ INTIIC10	UART1 transmission transfer end or buffer empty interrupt/CSI10 transfer end or buffer empty interrupt/IIC10 transfer end		00024H		✓	Note 7	Note 7	Note 7	Note 7
	17	INTSR1/ INTCSI11/ INTIIC11	UART1 reception transfer end/CSI11 transfer end or buffer empty interrupt/IIC11 transfer end		00026H		✓	✓	✓	✓	✓
	18	INTSRE1	UART1 reception communication error occurrence		00028H		✓	✓	✓	✓	✓
		INTTM03H	End of counting or capturing by channel 3 of timer array unit 0 (at higher 8-bit timer operation)				✓	✓	✓	✓	—
	19	INTIICA0	End of IICA0 communication		0002AH		✓	✓	✓	✓	—

Table 14 - 1 Interrupt Source List (2/3)

Interrupt Type	Default Priority Note 1	Interrupt Source		Internal/External	Vector Table Address	Basic Configuration Type Note 2	64-pin	48-pin	44-pin	32-pin	20-pin
		Name	Trigger								
Maskable	20	INTSR0/ INTCSI01/ INTIIC01	UART0 reception transfer end/CSI01 transfer end or buffer empty interrupt/IIC01 transfer end	Internal	0002CH	<a>	✓	✓	Note 8	Note 8	Note 8
	21	INTTM01	End of counting or capturing by channel 1 of timer array unit 0 (at 16-bit/lower 8-bit timer operation)		0002EH		✓	✓	✓	✓	✓
	22	INTTM02	End of counting or capturing by channel 2 of timer array unit 0		00030H		✓	✓	✓	✓	✓
	23	INTTM03	End of counting or capturing by channel 3 of timer array unit 0 (at 16-bit/lower 8-bit timer operation)		00032H		✓	✓	✓	✓	—
	24	INTAD	End of A/D conversion		00034H		✓	✓	✓	✓	✓
	26	INTITL	Interval signal of 32-bit interval timer detection		00038H		✓	✓	✓	✓	✓
	27	INTKR	Key return signal detection	External	0003AH	<c>	✓	✓	✓	—	—
	31	INTTM04	End of counting or capturing by channel 4 of timer array unit 0	Internal	00042H	<a>	✓	✓	✓	✓	—
	32	INTTM05	End of counting or capturing by channel 5 of timer array unit 0		00044H		✓	✓	✓	✓	✓
	33	INTTM06	End of counting or capturing by channel 6 of timer array unit 0		00046H		✓	✓	✓	✓	✓
	34	INTTM07	End of counting or capturing by channel 7 of timer array unit 0		00048H		✓	✓	✓	✓	✓
	35	INTP6	Pin input edge detection	External	0004AH	<b>	✓	✓	✓	✓	✓
	36	INTP7			0004CH		✓	—	—	—	—
	37	INTP8			0004EH		✓	✓	✓	✓	✓
	38	INTP9			00050H		✓	✓	✓	✓	✓
	39	INTFL	Reserved	Internal	00052H	<b>	✓	✓	✓	✓	✓
	40	INTP10	Pin input edge detection	External	00054H	<b>	✓	—	—	—	—
	41	INTP11	Pin input edge detection		00056H		✓	—	—	—	—
	42	INTURE0	UARTA0 reception communication error occurrence	Internal	00058H	<a>	✓	✓	✓	✓	✓
	51	INTUT0	UARTA0 transmission transfer end or buffer empty interrupt		0006AH		✓	✓	✓	✓	✓
	52	INTUR0	UARTA0 reception transfer end		0006CH		✓	✓	✓	✓	✓

Table 14 - 1 Interrupt Source List (3/3)

Interrupt Type	Default Priority Note 1	Interrupt Source		Internal/External	Vector Table Address	Basic Configuration Type Note 2	64-pin	48-pin	44-pin	32-pin	20-pin
		Name	Trigger								
Software	—	BRK	Execution of BRK instruction	—	0007EH	<d>	✓	✓	✓	✓	✓
Reset	—	RESET	$\overline{\text{RESET}}$ pin input	—	00000H	—	✓	✓	✓	✓	✓
		POR	Power-on-reset				✓	✓	✓	✓	✓
		LVD	Voltage detection				✓	✓	✓	✓	✓
		WDT	Overflow of watchdog timer				✓	✓	✓	✓	✓
		TRAP	Execution of illegal instruction				✓	✓	✓	✓	✓
		IAW	Illegal-memory access				✓	✓	✓	✓	✓
		RPE	RAM parity error				✓	✓	✓	✓	✓

Note 1. The default priority determines the sequence of interrupts if two or more maskable interrupts occur simultaneously. Zero indicates the highest priority and 52 indicates the lowest priority.

Note 2. Basic configuration types <a> to <d> correspond to <a> to <d> in **Figure 14 - 1 Basic Configuration of Interrupt Function**.

Note 3. This is applicable when the value of bit 7 (WDTINT) in the user option byte (000C0H) is 1.

Note 4. This is applicable when the value of bit 6 (LVD1SEL) in the voltage detection level register (LVIS) is 0 or the value of bit 6 (LVD0SEL) in the user option byte (000C1H) is 0.

Note 5. INTST2 is only present in products of this group.

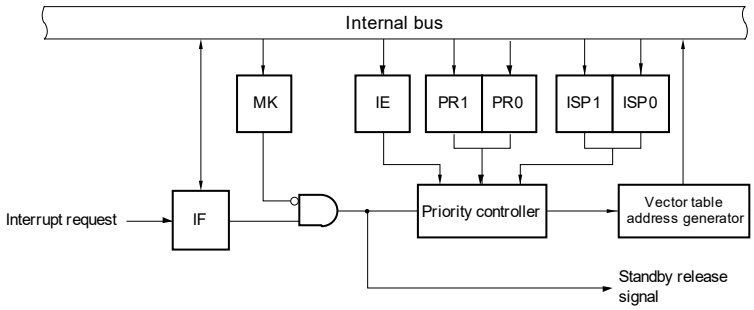
Note 6. INTSR2 is only present in products of this group.

Note 7. INTST1 is only present in products of this group.

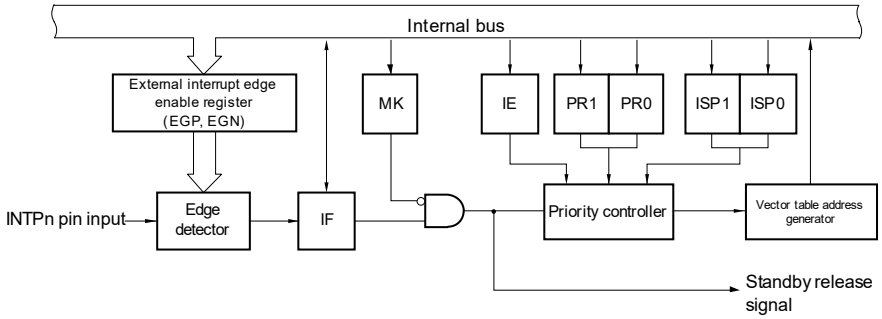
Note 8. INTSR0 is only present in products of this group.

Figure 14 - 1 Basic Configuration of Interrupt Function (1/2)

<a> Internal maskable interrupt



<b> External maskable interrupt (INTPn)

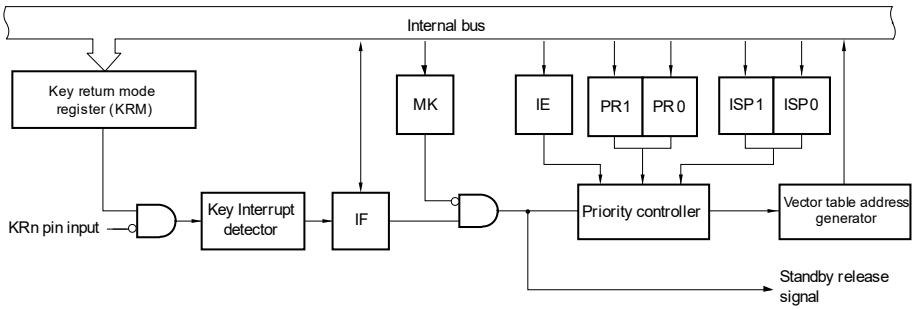


Remark 1. IF: Interrupt request flag
IE: Interrupt enable flag
ISP0: In-service priority flag 0
ISP1: In-service priority flag 1
MK: Interrupt mask flag
PR0: Priority specification flag 0
PR1: Priority specification flag 1

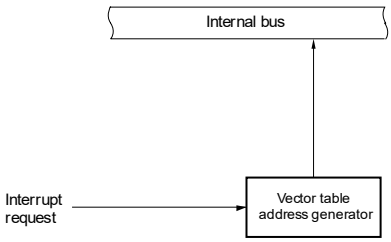
Remark 2. 20-pin products: n = 0, 3, 5, 6, 8, 9
32- to 48-pin products: n = 0 to 6, 8, 9
64-pin products: n = 0 to 11

Figure 14 - 1 Basic Configuration of Interrupt Function (2/2)

<c> External maskable interrupt (INTKR)



<d> Software interrupt



Remark 1. IF: Interrupt request flag
IE: Interrupt enable flag
ISP0: In-service priority flag 0
ISP1: In-service priority flag 1
MK: Interrupt mask flag
PR0: Priority specification flag 0
PR1: Priority specification flag 1

Remark 2. 44-pin products: n = 0 to 3
48-pin products: n = 0 to 5
64-pin products: n = 0 to 7

14.3 Registers for Controlling the Interrupt Functions

The following registers are used to control the interrupt functions.

- Interrupt request flag registers (IF0L, IF0H, IF1L, IF1H, IF2L, IF2H, IF3L)
- Interrupt mask flag registers (MK0L, MK0H, MK1L, MK1H, MK2L, MK2H, MK3L)
- Priority specification flag registers (PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR03L, PR10L, PR10H, PR11L, PR11H, PR12L, PR12H, PR13L)
- External interrupt rising edge enable registers (EGP0, EGP1), External interrupt falling edge enable registers (EGN0, EGN1)
- Program status word (PSW)
- Port mode registers (PMxx)

Table 14 - 2 shows a list of interrupt request flags, interrupt mask flags, and priority specification flags corresponding to interrupt request sources.

Table 14 - 2 Flags Corresponding to Interrupt Request Sources (1/4)

Interrupt Source	Interrupt Request Flag		Interrupt Mask Flag		Priority Specification Flag		64-pin	48-pin	44-pin	32-pin	20-pin
		Register		Register		Register					
INTWDTI	WDTIIF	IF0L	WDTIMK	MK0L	WDTIPR0, WDTIPR1	PR00L, PR10L	✓	✓	✓	✓	✓
INTLVI**Note**	LVIIF**Note**		LVIMK**Note**		LVIPR0, LVIPR1**Note**		✓	✓	✓	✓	✓
INTP0	PIF0		PMK0		PPR00, PPR10		✓	✓	✓	✓	✓
INTP1	PIF1		PMK1		PPR01, PPR11		✓	✓	✓	✓	—
INTP2	PIF2		PMK2		PPR02, PPR12		✓	✓	✓	✓	—
INTP3	PIF3		PMK3		PPR03, PPR13		✓	✓	✓	✓	✓
INTP4	PIF4		PMK4		PPR04, PPR14		✓	✓	✓	✓	—
INTP5	PIF5		PMK5		PPR05, PPR15		✓	✓	✓	✓	✓

Note The DLVD0F and DLVD1F flags of the LVIM register can be used to confirm which voltage detector has issued the given interrupt, LVD0 or LVD1. For details, see **19.3.1 Voltage detection register (LVIM)**.

Table 14 - 2 Flags Corresponding to Interrupt Request Sources (2/4)

Interrupt Source	Interrupt Request Flag		Interrupt Mask Flag		Priority Specification Flag		64-pin	48-pin	44-pin	32-pin	20-pin
		Register		Register		Register					
INTST2Note 1	STIF2Note 1	IF0H	STMK2Note 1	MK0H	STPR02, STPR12Note 1	PR00H, PR10H	✓	✓	✓	✓	✓
INTCSI20Note 1	CSIIF20Note 1		CSIMK20Note 1		CSIPR020, CSIPR120Note 1		✓	✓	✓	✓	—
INTIIC20Note 1	IICIF20Note 1		IICMK20Note 1		IICPR020, IICPR120Note 1		✓	✓	✓	✓	—
INTSR2Note 2	SRIF2Note 2		SRMK2Note 2		SRPR02, SRPR12Note 2		✓	✓	✓	✓	✓
INTCSI21Note 2	CSIIF21Note 2		CSIMK21Note 2		CSIPR021, CSIPR121Note 2		✓	✓	✓	—	—
INTIIC21Note 2	IICIF21Note 2		IICMK21Note 2		IICPR021, IICPR121Note 2		✓	✓	✓	—	—
INTSRE2	SREIF2		SREMK2		SREPR02, SREPR12		✓	✓	✓	✓	✓
INTST0Note 3	STIF0Note 3		STMK0Note 3		STPR00, STPR10Note 3		✓	✓	✓	✓	✓
INTCSI00Note 3	CSIIF00Note 3		CSIMK00Note 3		CSIPR000, CSIPR100Note 3		✓	✓	✓	✓	✓
INTIIC00Note 3	IICIF00Note 3		IICMK00Note 3		IICPR000, IICPR100Note 3		✓	✓	✓	✓	✓
INTTM00	TMIF00		TMMK00		TMPR000, TMPR100		✓	✓	✓	✓	✓
INTSRE0Note 4	SREIF0Note 4		SREMK0Note 4		SREPR00, SREPR10Note 4		✓	✓	✓	✓	✓
INTTM01HNote 4	TMIF01HNote 4		TMMK01HNote 4		TMPR001H, TMPR101HNote 4		✓	✓	✓	✓	✓

Note 1. If any of the interrupt sources INTST2, INTCSI20, and INTIIC20 is generated, bit 0 of the IF0H register is set to 1. Bit 0 of the MK0H, PR00H, and PR10H registers supports these three interrupt sources.

Note 2. If any of the interrupt sources INTSR2, INTCSI21, and INTIIC21 is generated, bit 1 of the IF0H register is set to 1. Bit 1 of the MK0H, PR00H, and PR10H registers supports these three interrupt sources.

Note 3. If any of the interrupt sources INTST0, INTCSI00, and INTIIC00 is generated, bit 5 of the IF0H register is set to 1. Bit 5 of the MK0H, PR00H, and PR10H registers supports these three interrupt sources.

Note 4. Do not use a UART0 reception error interrupt and an interrupt of channel 1 of TAU0 (at higher 8-bit timer operation) at the same time because they share flags for the interrupt request sources. If the UART0 reception error interrupt is not used (EOC01 = 0), UART0 and channel 1 of TAU0 (at higher 8-bit timer operation) can be used at the same time. If either of the interrupt sources INTSRE0 or INTTM01H is generated, bit 7 of the IF0H register is set to 1. Bit 7 of the MK0H, PR00H, and PR10H registers supports these two interrupt sources.

Table 14 - 2 Flags Corresponding to Interrupt Request Sources (3/4)

Interrupt Source	Interrupt Request Flag		Interrupt Mask Flag		Priority Specification Flag		64-pin	48-pin	44-pin	32-pin	20-pin
		Register		Register		Register					
INTST1Note 1	STIF1Note 1	IF1L	STMK1Note 1	MK1L	STPR01, STPR11Note 1	PR01L, PR11L	✓	✓	✓	✓	✓
INTCSI10Note 1	CSIIF10Note 1		CSIMK10Note 1		CSIPR010,CSIPR110Note 1		✓	—	—	—	—
INTIIC10Note 1	IICIF10Note 1		IICMK10Note 1		IICPR010, IICPR110Note 1		✓	—	—	—	—
INTSR1Note 2	SRIF1Note 2		SRMK1Note 2		SRPR01, SRPR11Note 2		✓	✓	✓	✓	✓
INTCSI11Note 2	CSIIF11Note 2		CSIMK11Note 2		CSIPR011, CSIPR111Note 2		✓	✓	✓	✓	✓
INTIIC11Note 2	IICIF11Note 2		IICMK11Note 2		IICPR011, IICPR111Note 2		✓	✓	✓	✓	✓
INTSRE1Note 3	SREIF1Note 3		SREMK1Note 3		SREPR01, SREPR11Note 3		✓	✓	✓	✓	✓
INTTM03HNote 3	TMIF03HNote 3		TMMK03HNote 3		TMPR003H,TMPR103HNote 3		✓	✓	✓	✓	—
INTIICA0	IICAIF0		IICAMK0		IICAPR00, IICAPR10		✓	✓	✓	✓	—
INTSR0Note 4	SRIF0Note 4		SRMK0Note 4		SRPR00, SRPR10Note 4		✓	✓	✓	✓	✓
INTCSI01Note 4	CSIIF01Note 4		CSIMK01Note 4		CSIPR001, CSIPR101Note 4		✓	✓	—	—	—
INTIIC01Note 4	IICIF01Note 4		IICMK01Note 4		IICPR001, IICPR101Note 4		✓	✓	—	—	—
INTTM01	TMIF01		TMMK01		TMPR001, TMPR101		✓	✓	✓	✓	✓
INTTM02	TMIF02		TMMK02		TMPR002, TMPR102		✓	✓	✓	✓	✓
INTTM03	TMIF03		TMMK03		TMPR003, TMPR103		✓	✓	✓	✓	—
INTAD	ADIF	IF1H	ADMK	MK1H	ADPR0, ADPR1	PR01H, PR11H	✓	✓	✓	✓	✓
INTITL	ITLIF		ITLMK		ITLPR0, ITLPR1		✓	✓	✓	✓	✓
INTKR	KRIF		KRMK		KRPR0, KRPR1		✓	✓	✓	—	—
INTTM04	TMIF04		TMMK04		TMPR004, TMPR104		✓	✓	✓	✓	—

Note 1. If any of the interrupt sources INTST1, INTCSI10, and INTIIC10 is generated, bit 0 of the IF1L register is set to 1. Bit 0 of the MK1L, PR01L, and PR11L registers supports these three interrupt sources.

Note 2. If any of the interrupt sources INTSR1, INTCSI11, and INTIIC11 is generated, bit 1 of the IF1L register is set to 1. Bit 1 of the MK1L, PR01L, and PR11L registers supports these three interrupt sources.

Note 3. Do not use a UART1 reception error interrupt and an interrupt of channel 3 of TAU0 (at higher 8-bit timer operation) at the same time because they share flags for the interrupt request sources. If the UART1 reception error interrupt is not used (EOC03 = 0), UART1 and channel 3 of TAU0 (at higher 8-bit timer operation) can be used at the same time. If either of the interrupt sources INTSRE1 or INTTM03H is generated, bit 2 of the IF1L register is set to 1. Bit 2 of the MK1L, PR01L, and PR11L registers supports these two interrupt sources.

Note 4. If any of the interrupt sources INTSR0, INTCSI01, and INTIIC01 is generated, bit 4 of the IF1L register is set to 1. Bit 4 of the MK1L, PR01L, and PR11L registers supports these three interrupt sources.

Table 14 - 2 Flags Corresponding to Interrupt Request Sources (4/4)

Interrupt Source	Interrupt Request Flag		Interrupt Mask Flag		Priority Specification Flag		64-pin	48-pin	44-pin	32-pin	20-pin
		Register		Register		Register					
INTTM05	TMIF05	IF2L	TMMK05	MK2L	TMPR005, TMPR105	PR02L, PR12L	✓	✓	✓	✓	✓
INTTM06	TMIF06		TMMK06		TMPR006, TMPR106		✓	✓	✓	✓	✓
INTTM07	TMIF07		TMMK07		TMPR007, TMPR107		✓	✓	✓	✓	✓
INTP6	PIF6		PMK6		PPR06, PPR16		✓	✓	✓	✓	✓
INTP7	PIF7		PMK7		PPR07, PPR17		✓	—	—	—	—
INTP8	PIF8		PMK8		PPR08, PPR18		✓	✓	✓	✓	✓
INTP9	PIF9		PMK9		PPR09, PPR19		✓	✓	✓	✓	✓
INTFL	FLIF		FLMK		FLPR0, FLPR1		✓	✓	✓	✓	✓
INTP10	PIF10	IF2H	PMK10	MK2H	PPR010, PPR110	PR02H, PR12H	✓	—	—	—	—
INTP11	PIF11		PMK11		PPR011, PPR111		✓	—	—	—	—
INTURE0	UREIF0		UREMK0		UREPR00, UREPR10		✓	✓	✓	✓	✓
INTUT0	UTIF0	IF3L	UTMK0	MK3L	UTPR00, UTPR10	PR03L, PR13L	✓	✓	✓	✓	✓
INTUR0	URIF0		URMK0		URPR00, URPR10		✓	✓	✓	✓	✓

14.3.1 Interrupt request flag registers (IF0L, IF0H, IF1L, IF1H, IF2L, IF2H, IF3L)

The interrupt request flags are set to 1 when the corresponding interrupt request is generated or an instruction is executed. They are cleared to 0 when the given interrupt request is acknowledged, a reset signal is generated, or an instruction is executed. When an interrupt is acknowledged, the interrupt request flag is automatically cleared and then the interrupt routine is entered. The IF0L, IF0H, IF1L, IF1H, IF2L, IF2H, and IF3L registers can be set by a 1-bit or 8-bit memory manipulation instruction. When the IF0L and IF0H registers, the IF1L and IF1H registers, and the IF2L and IF2H registers are combined to form 16-bit registers IF0, IF1, and IF2, they can be set by a 16-bit memory manipulation instruction. The value of each register following a reset is 00H.

Remark If an instruction to write to these registers is executed, the number of clock cycles executing the instruction takes is increased by two.

Figure 14 - 2 Format of Interrupt Request Flag Registers (IF0L, IF0H, IF1L, IF1H, IF2L, IF2H, IF3L) (1/2)

Address: FFFE0H
After reset: 00H
R/W: R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IF0L	PIF5	PIF4	PIF3	PIF2	PIF1	PIF0	LVIIF	WDTIIF

Address: FFFE1H
After reset: 00H
R/W: R/W

Symbol	<7>	<6>	<5>	4	3	<2>	<1>	<0>
IF0H	SREIF0 TMIF01H	TMIF00	STIF0 CSIIF00 IICIF00	0	0	SREIF2	SRIF2 CSIIF21 IICIF21	STIF2 CSIIF20 IICIF20

Address: FFFE2H
After reset: 00H
R/W: R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IF1L	TMIF03	TMIF02	TMIF01	SRIF0 CSIIF01 IICIF01	IICAIF0	SREIF1 TMIF03H	SRIF1 CSIIF11 IICIF11	STIF1 CSIIF10 IICIF10

Address: FFFE3H
After reset: 00H
R/W: R/W

Symbol	<7>	6	5	4	<3>	<2>	1	<0>
IF1H	TMIF04	0	0	0	KRIF	ITLIF	0	ADIF

Figure 14 - 2 Format of Interrupt Request Flag Registers (IF0L, IF0H, IF1L, IF1H, IF2L, IF2H, IF3L) (2/2)

Address: FFFD0H
After reset: 00H
R/W: R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IF2L	FLIF	PIF9	PIF8	PIF7	PIF6	TMIF07	TMIF06	TMIF05

Address: FFFD1H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	<2>	<1>	<0>
IF2H	0	0	0	0	0	UREIF0	PIF11	PIF10

Address: FFFD2H
After reset: 00H
R/W: R/W

Symbol	7	6	5	<4>	<3>	2	1	0
IF3L	0	0	0	URIF0	UTIF0	0	0	0

xxIFx	Interrupt request flag
0	No interrupt request signal is generated
1	Indicates the generation of the interrupt request signal and the interrupt request currently being in the active state.

Caution 1. The available registers and bits differ depending on the product. For details about the registers and bits available for each product, see Table 14 - 2 Flags Corresponding to Interrupt Request Sources. Be sure to set bits that are not available to the initial value.

Caution 2. When manipulating a flag of the interrupt request flag register, use a 1-bit memory manipulation instruction (CLR1). When describing in C language, use a bit memory manipulation instruction such as "IF0L.0 = 0;" or "_asm("clr1 IF0L.0");" because the compiled assembler must be a 1-bit memory manipulation instruction (CLR1). If a program is described in C language using an 8-bit memory manipulation instruction such as "IF0L & = 0xfe;" and compiled, it becomes the assembler of three instructions.

```
mov a, IF0L
and a, #0FEH
mov IF0L, a
```

In this case, even if the request flag of the another bit of the same interrupt request flag register (IF0L) is set to 1 at the timing between "mov a, IF0L" and "mov IF0L, a", the flag is cleared to 0 at "mov IF0L, a". Therefore, care must be exercised when using an 8-bit memory manipulation instruction in C language.

14.3.2 Interrupt mask flag registers (MK0L, MK0H, MK1L, MK1H, MK2L, MK2H, MK3L)

The interrupt mask flags are used to enable/disable the corresponding maskable interrupt. The MK0L, MK0H, MK1L, MK1H, MK2L, MK2H, and MK3L registers can be set by a 1-bit or 8-bit memory manipulation instruction. When the MK0L and MK0H registers, the MK1L and MK1H registers, and the MK2L and MK2H registers are combined to form 16-bit registers MK0, MK1, and MK2, they can be set by a 16-bit memory manipulation instruction. The value of each register following a reset is FFH.

Remark If an instruction to write to these registers is executed, the number of clock cycles executing the instruction takes is increased by two.

Figure 14 - 3 Format of Interrupt Mask Flag Registers (MK0L, MK0H, MK1L, MK1H, MK2L, MK2H, MK3L) (1/2)

Address: FFFE4H
After reset: FFH
R/W: R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
MK0L	PMK5	PMK4	PMK3	PMK2	PMK1	PMK0	LVIMK	WDTIMK

Address: FFFE5H
After reset: FFH
R/W: R/W

Symbol	<7>	<6>	<5>	4	3	<2>	<1>	<0>
MK0H	SREMK0 TMMK01H	TMMK00	STMK0 CSIMK00 IICMK00	1	1	SREMK2 TMMK11H	SRMK2 CSIMK21 IICMK21	STMK2 CSIMK20 IICMK20

Address: FFFE6H
After reset: FFH
R/W: R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
MK1L	TMMK03	TMMK02	TMMK01	SRMK0 CSIMK01 IICMK01	IICAMK0	SREMK1 TMMK03H	SRMK1 CSIMK11 IICMK11	STMK1 CSIMK10 IICMK10

Address: FFFE7H
After reset: FFH
R/W: R/W

Symbol	<7>	6	5	4	<3>	<2>	1	<0>
MK1H	TMMK04	1	1	1	KRMK	ITLMK	1	ADMK

Figure 14 - 3 Format of Interrupt Mask Flag Registers (MK0L, MK0H, MK1L, MK1H, MK2L, MK2H, MK3L) (2/2)

Address: FFFD4H
After reset: FFH
R/W: R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
MK2L	FLMK	PMK9	PMK8	PMK7	PMK6	TMMK07	TMMK06	TMMK05

Address: FFFD5H
After reset: FFH
R/W: R/W

Symbol	7	6	5	4	3	<2>	<1>	<0>
MK2H	1	1	1	1	1	UREMK0	PMK11	PMK10

Address: FFFD6H
After reset: FFH
R/W: R/W

Symbol	7	6	5	<4>	<3>	2	1	0
MK3L	1	1	1	URMK0	UTMK0	1	1	1

xxMKx	Interrupt servicing control
0	Interrupt servicing enabled
1	Interrupt servicing disabled

Caution The available registers and bits differ depending on the product. For details about the registers and bits available for each product, see Table 14 - 2 Flags Corresponding to Interrupt Request Sources. Be sure to set bits that are not available to the initial value.

14.3.3 Priority specification flag registers (PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR03L, PR10L, PR10H, PR11L, PR11H, PR12L, PR12H, PR13L)

The priority specification flag registers are used to set the corresponding maskable interrupt priority level. A priority level is set by using the PR0xy and PR1xy registers in combination (xy = 0L, 0H, 1L, 1H, 2L, 2H, or 3L). The PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR03L, PR10L, PR10H, PR11L, PR11H, PR12L, PR12H, and PR13L registers can be set by a 1-bit or 8-bit memory manipulation instruction. If the PR00L and PR00H registers, the PR01L and PR01H registers, the PR02L and PR02H registers, the PR10L and PR10H registers, the PR11L and PR11H registers, and the PR12L and PR12H registers are combined to form 16-bit registers PR00, PR01, PR02, PR10, PR11, and PR12, they can be set by a 16-bit memory manipulation instruction. The value of each register following a reset is FFH.

Remark If an instruction to write to these registers is executed, the number of clock cycles executing the instruction takes is increased by two.

Figure 14 - 4 Format of Priority Specification Flag Registers (PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR03L, PR10L, PR10H, PR11L, PR11H, PR12L, PR12H, PR13L) (1/3)

Address: FFFE8H
After reset: FFH
R/W: R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
PR00L	PPR05	PPR04	PPR03	PPR02	PPR01	PPR00	LVIPR0	WDTIPR0

Address: FFFECH
After reset: FFH
R/W: R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
PR10L	PPR15	PPR14	PPR13	PPR12	PPR11	PPR10	LVIPR1	WDTIPR1

Address: FFFE9H
After reset: FFH
R/W: R/W

Symbol	<7>	<6>	<5>	4	3	<2>	<1>	<0>
PR00H	SREPR00 TMPR001H	TMPR000	STPR00 CSIPR000 IICPR000	1	1	SREPR02 TMPR011H	SRPR02 CSIPR021 IICPR021	STPR02 CSIPR020 IICPR020

Address: FFFEDH
After reset: FFH
R/W: R/W

Symbol	<7>	<6>	<5>	4	3	<2>	<1>	<0>
PR10H	SREPR10 TMPR101H	TMPR100	STPR10 CSIPR100 IICPR100	1	1	SREPR12 TMPR111H	SRPR12 CSIPR121 IICPR121	STPR12 CSIPR120 IICPR120

Figure 14 - 4 Format of Priority Specification Flag Registers (PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR03L, PR10L, PR10H, PR11L, PR11H, PR12L, PR12H, PR13L) (2/3)

Address: FFFEAH
After reset: FFH
R/W: R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
PR01L	TMPR003	TMPR002	TMPR001	SRPR00 CSIPR001 IICPR001	IICAPR00	SREPR01 TMPR003H	SRPR01 CSIPR011 IICPR011	STPR01 CSIPR010 IICPR010

Address: FFFEEH
After reset: FFH
R/W: R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
PR11L	TMPR103	TMPR102	TMPR101	SRPR10 CSIPR101 IICPR101	IICAPR10	SREPR11 TMPR103H	SRPR11 CSIPR111 IICPR111	STPR11 CSIPR110 IICPR110

Address: FFFEBH
After reset: FFH
R/W: R/W

Symbol	<7>	6	5	4	<3>	<2>	1	<0>
PR01H	TMPR004	1	1	1	KRPR0	ITLPR0	1	ADPR0

Address: FFFEFH
After reset: FFH
R/W: R/W

Symbol	<7>	6	5	4	<3>	<2>	1	<0>
PR11H	TMPR104	1	1	1	KRPR1	ITLPR1	1	ADPR1

Address: FFFD8H
After reset: FFH
R/W: R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
PR02L	FLPR0	PPR09	PPR08	PPR07	PPR06	TMPR007	TMPR006	TMPR005

Address: FFFDCH
After reset: FFH
R/W: R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
PR12L	FLPR1	PPR19	PPR18	PPR17	PPR16	TMPR107	TMPR106	TMPR105

Figure 14 - 4 Format of Priority Specification Flag Registers (PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR03L, PR10L, PR10H, PR11L, PR11H, PR12L, PR12H, PR13L) (3/3)

Address: FFFD9H
After reset: FFH
R/W: R/W

Symbol	7	6	5	4	3	<2>	<1>	<0>
PR02H	1	1	1	1	1	UREPR00	PPR011	PPR010

Address: FFFDDH
After reset: FFH
R/W: R/W

Symbol	7	6	5	4	3	<2>	<1>	<0>
PR12H	1	1	1	1	1	UREPR10	PPR111	PPR110

Address: FFFDAH
After reset: FFH
R/W: R/W

Symbol	7	6	5	<4>	<3>	2	1	0
PR03L	1	1	1	URPR00	UTPR00	1	1	1

Address: FFFDEH
After reset: FFH
R/W: R/W

Symbol	7	6	5	<4>	<3>	2	1	0
PR13L	1	1	1	URPR10	UTPR10	1	1	1

xxPR1x	xxPR0x	Priority level selection
0	0	Specify level 0 (high priority level)
0	1	Specify level 1
1	0	Specify level 2
1	1	Specify level 3 (low priority level)

Caution The available registers and bits differ depending on the product. For details about the registers and bits available for each product, see Table 14 - 2 Flags Corresponding to Interrupt Request Sources. Be sure to set bits that are not available to the initial value.

14.3.4 External interrupt rising edge enable registers (EGP0, EGP1), External interrupt falling edge enable registers (EGN0, EGN1)

These registers specify the valid edge for INTP0 to INTP11. The EGP0, EGP1, EGN0, and EGN1 registers can be set by a 1-bit or 8-bit memory manipulation instruction. The value of each register following a reset is 00H.

Figure 14 - 5 Format of External Interrupt Rising Edge Enable Registers (EGP0, EGP1) and External Interrupt Falling Edge Enable Registers (EGN0, EGN1)

Address: FFF38H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
EGP0	EGP7	EGP6	EGP5	EGP4	EGP3	EGP2	EGP1	EGP0

Address: FFF39H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
EGN0	EGN7	EGN6	EGN5	EGN4	EGN3	EGN2	EGN1	EGN0

Address: FFF3AH
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
EGP1	0	0	0	0	EGP11	EGP10	EGP9	EGP8

Address: FFF3BH
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
EGN1	0	0	0	0	EGN11	EGN10	EGN9	EGN8

EGPn	EGNn	INTPn pin valid edge selection
0	0	Edge detection disabled
0	1	Falling edge
1	0	Rising edge
1	1	Both rising and falling edges

Remark n = 0 to 11

Table 14 - 3 shows the port pins (INTPx) corresponding to the EGPn and EGNn bits.

Table 14 - 3 Port Pins (INTPx) Corresponding to EGPn and EGNn Bits

Detection Enable Bit		Port Pins (INTPx)	64-pin	32- to 48-pin	20-pin
EGP0	EGN0	INTP0	✓	✓	✓
EGP1	EGN1	INTP1	✓	✓	—
EGP2	EGN2	INTP2	✓	✓	—
EGP3	EGN3	INTP3	✓	✓	✓
EGP4	EGN4	INTP4	✓	✓	—
EGP5	EGN5	INTP5	✓	✓	✓
EGP6	EGN6	INTP6	✓	✓	✓
EGP7	EGN7	INTP7	✓	—	—
EGP8	EGN8	INTP8	✓	✓	✓
EGP9	EGN9	INTP9	✓	✓	✓
EGP10	EGN10	INTP10	✓	—	—
EGP11	EGN11	INTP11	✓	—	—

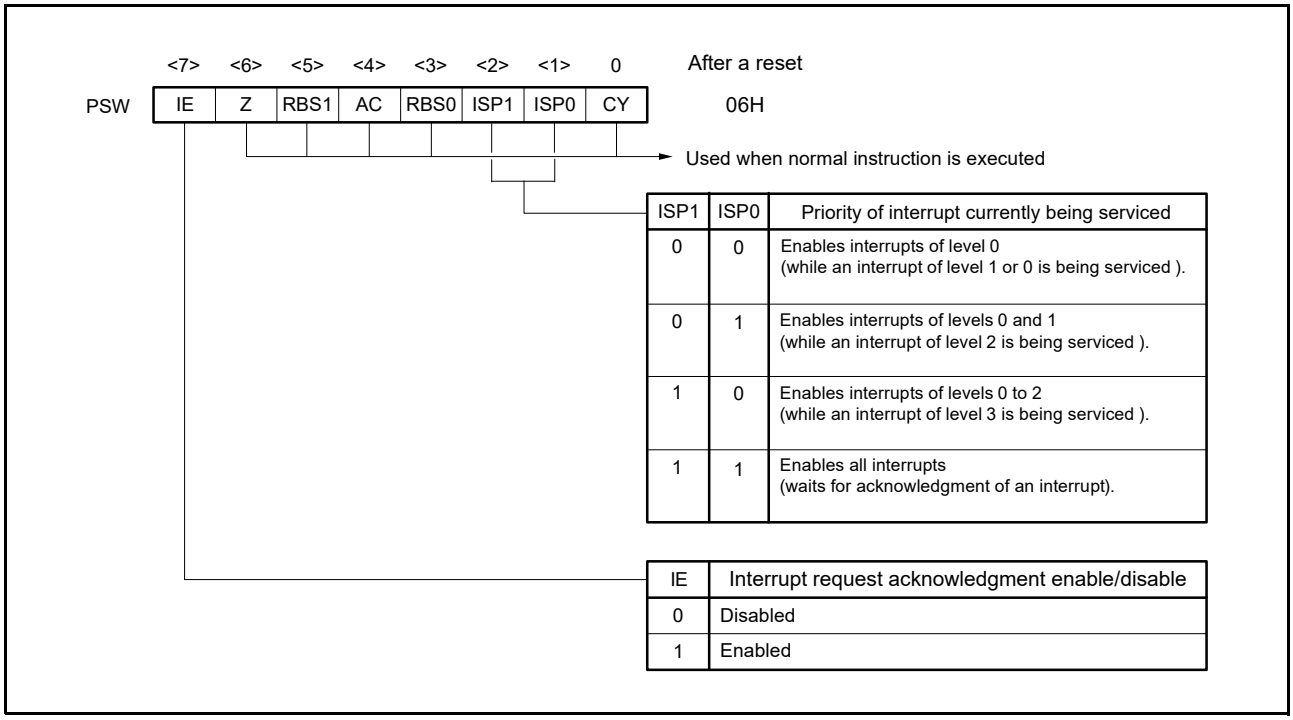
Caution **When the input port pins used for the external interrupt functions are switched to the output mode, the INTPn interrupt might be generated upon detection of a valid edge. When switching the input port pins to the output mode, set the corresponding port mode register (PMxx) to 0 after disabling the edge detection (by setting EGPn and EGNn to 0).**

Remark 1. For edge detection ports, see **2.1 Functions of Port Pins**.
Remark 2. n = 0 to 11

14.3.5 Program status word (PSW)

The program status word (PSW) is a register that holds the results of instruction execution and the current state for interrupts. The IE flag that sets maskable interrupt enable/disable and the ISP1 and ISP0 flags that control multiple interrupt servicing are mapped to the PSW. Besides 8-bit read/write, this register can carry out operations using bit memory manipulation instructions and dedicated instructions (EI and DI). When a vectored interrupt request is acknowledged, if the BRK instruction is executed, the contents of the PSW are automatically saved into a stack and the IE flag is reset to 0. Upon acknowledgment of a maskable interrupt request, if the value of the priority specification flag register of the acknowledged interrupt is not 00, its value minus 1 is transferred to the ISP1 and ISP0 flags. The PSW contents are also saved into the stack with the PUSH PSW instruction. They are restored from the stack with the RETI, RETB, and POP PSW instructions. The value of the PSW following a reset is 06H.

Figure 14 - 6 Configuration of Program Status Word



ISP1	ISP0	Priority of interrupt currently being serviced
0	0	Enables interrupts of level 0 (while an interrupt of level 1 or 0 is being serviced).
0	1	Enables interrupts of levels 0 and 1 (while an interrupt of level 2 is being serviced).
1	0	Enables interrupts of levels 0 to 2 (while an interrupt of level 3 is being serviced).
1	1	Enables all interrupts (waits for acknowledgment of an interrupt).

IE	Interrupt request acknowledgment enable/disable
0	Disabled
1	Enabled

14.3.6 Registers for controlling the port functions multiplexed with the interrupt inputs

Set the following registers to control the port functions multiplexed with the interrupt inputs.

- Port mode registers (PMxx)

For details, see **4.3.1 Port mode registers (PMxx)**. For examples of register settings for controlling port functions, see **4.5.4 Examples of register settings for port and alternate functions**.

14.4 Interrupt Servicing Operations

14.4.1 Maskable interrupt request acknowledgment

A maskable interrupt request becomes acknowledgeable when the interrupt request flag is set to 1 and the mask (MK) flag corresponding to that interrupt request is cleared to 0. A vectored interrupt request is acknowledged if interrupts are in the interrupt enabled state (when the IE flag is set to 1). However, a low-priority interrupt request is not acknowledged during servicing of a higher priority interrupt request. **Table 14 - 4** shows the times required from acceptance of an interrupt request in the form of the generation of a maskable interrupt request until vectored interrupt servicing proceeds. For the interrupt request acknowledgment timing, see **Figure 14 - 8** and **Figure 14 - 9**.

Table 14 - 4 Times from Generation of Maskable Interrupt Request until Vectored Interrupt Servicing

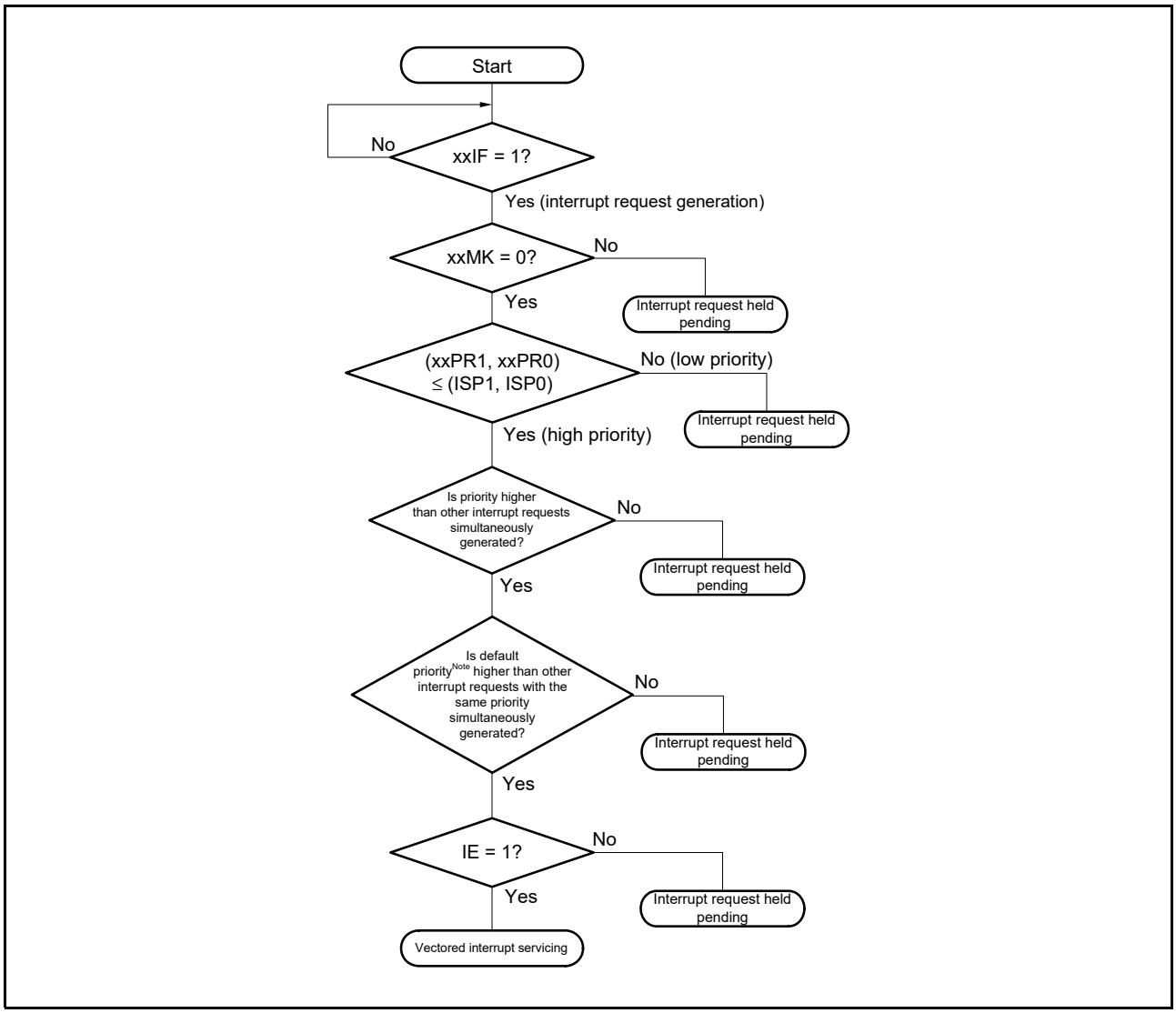
	Minimum Time	Maximum Time[Note]
Times required for interrupt request acceptance	9 clock cycles	16 clock cycles

Note Maximum time does not apply when an instruction from the internal RAM area is executed.

Remark One clock cycle is equivalent to 1/f_{CLK}, where f_{CLK} is the frequency of the CPU clock.

If two or more maskable interrupt requests are generated simultaneously, the request with a higher priority level specified in the priority specification flag is acknowledged first. If two or more interrupt requests have the same priority level, the request with the highest default priority is acknowledged first. An interrupt request that is held pending is acknowledged when it becomes acknowledgeable. **Figure 14 - 7** shows the interrupt request acknowledgment algorithm. If a maskable interrupt request is acknowledged, the contents are saved into the stacks in the order of PSW, then PC, the IE flag is reset to 0, and the contents of the priority specification flag corresponding to the acknowledged interrupt are transferred to the ISP1 and ISP0 flags. The vector table data determined for each interrupt request is loaded into the PC and program control branches to the specified servicing. Restoring from an interrupt is possible by using the RETI instruction.

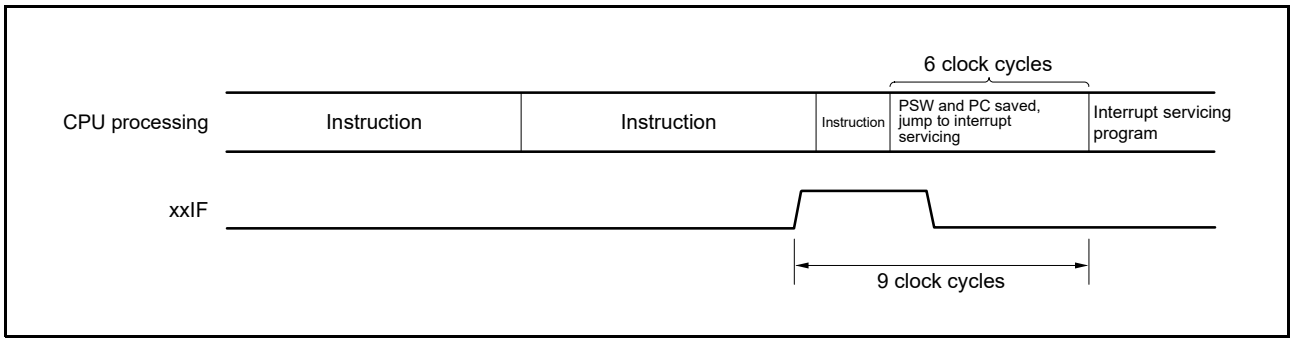
Figure 14 - 7 Interrupt Request Acknowledgment Processing Algorithm



Note For the default priority, refer to **Table 14 - 1 Interrupt Source List**.

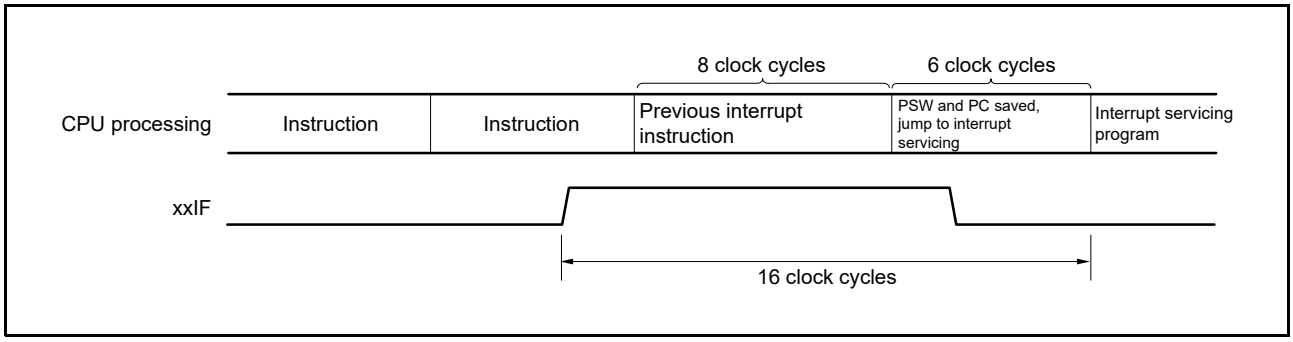
Remark xxIF: Interrupt request flag
xxMK: Interrupt mask flag
xxPR0: Priority specification flag 0
xxPR1: Priority specification flag 1
IE: Flag that controls acknowledgment of maskable interrupt request (1 = Enable, 0 = Disable)
ISP1, ISP0: Flags that indicate the priority level of the interrupt currently being serviced (see **Figure 14 - 6 Configuration of Program Status Word**)

Figure 14 - 8 Minimum Time Taken until Acknowledgment of an Interrupt Request



Remark One clock cycle is equivalent to 1/f_{CLK}, where f_{CLK} is the frequency of the CPU clock.

Figure 14 - 9 Maximum Time Taken until Acknowledgment of an Interrupt Request



Remark One clock cycle is equivalent to 1/f_{CLK}, where f_{CLK} is the frequency of the CPU clock.

14.4.2 Software interrupt request acknowledgment

A software interrupt request is acknowledged by BRK instruction execution. Software interrupts cannot be disabled. If a software interrupt request is acknowledged, the contents are saved into the stacks in the order of the program status word (PSW), then program counter (PC), the IE flag is reset to 0, the contents of the vector table (0007EH, 0007FH) are loaded into the PC, and program control branches to the specified servicing. Restoring from a software interrupt is possible by using the RETB instruction.

Caution The RETI instruction cannot be used for restoring from the software interrupt.

14.4.3 Multiple interrupt servicing

Multiple interrupt servicing occurs when another interrupt request is acknowledged during execution of an interrupt. Multiple interrupt servicing does not occur unless the interrupt request acknowledgment enabled state is selected (IE = 1). When an interrupt request is acknowledged, interrupt request acknowledgment becomes disabled (IE = 0). Therefore, to enable multiple interrupt servicing, it is necessary to set the IE flag to 1 with the EI instruction during interrupt servicing to enable interrupt acknowledgment. Moreover, even if interrupts are enabled, multiple interrupt servicing may not be enabled, this being subject to interrupt priority control. Two types of priority control are available: default priority control and programmable priority control. Programmable priority control is used for multiple interrupt servicing. In the interrupt enabled state, if an interrupt request with a priority higher than that of the interrupt currently being serviced is generated, it is acknowledged for multiple interrupt servicing. If an interrupt with a priority equal to or lower than that of the interrupt currently being serviced is generated during interrupt servicing, it is not acknowledged for multiple interrupt servicing. However, when setting the IE flag to 1 during the interruption at level 0, other level 0 interruptions can be allowed. Interrupt requests that are not enabled because interrupts are in the interrupt disabled state or because they have a lower priority are held pending. When servicing of the current interrupt ends, the pending interrupt request is acknowledged following execution of at least one main processing instruction.

Table 14 - 5 shows relationship between interrupt requests enabled for multiple interrupt servicing and **Figure 14 - 10** shows multiple interrupt servicing examples.

Table 14 - 5 Relationship between Interrupt Requests Enabled for Multiple Interrupt Servicing during Interrupt Servicing

Multiple Interrupt Request / Maskable Interrupt		Maskable Interrupt Request								Software Interrupt Request
		Priority Level 0 (PR = 00)		Priority Level 1 (PR = 01)		Priority Level 2 (PR = 10)		Priority Level 3 (PR = 11)		
		IE = 1	IE = 0	IE = 1	IE = 0	IE = 1	IE = 0	IE = 1	IE = 0	
Processing in progress	ISP1 and ISP0 = 00B	✓	—	—	—	—	—	—	—	✓
	ISP1 and ISP0 = 01B	✓	—	✓	—	—	—	—	—	✓
	ISP1 and ISP0 = 10B	✓	—	✓	—	✓	—	—	—	✓
Waiting for reception	ISP1 and ISP0 = 11B	✓	—	✓	—	✓	—	✓	—	✓

Remark 1. ✓: Multiple interrupt servicing enabled

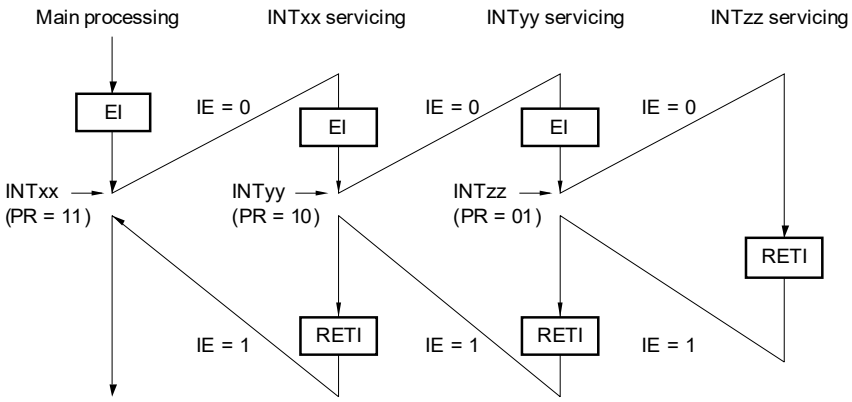
Remark 2. —: Multiple interrupt servicing disabled

Remark 3. ISP1, ISP0, and IE are flags contained in the PSW.
ISP1 and ISP0 = 00B: An interrupt of level 1 or level 0 is being serviced.
ISP1 and ISP0 = 01B: An interrupt of level 2 is being serviced.
ISP1 and ISP0 = 10B: An interrupt of level 3 is being serviced.
ISP1 and ISP0 = 11B: Wait for an interrupt acknowledgment (all interrupts are enabled).
IE = 0: Interrupt request acknowledgment is disabled.
IE = 1: Interrupt request acknowledgment is enabled.

Remark 4. PR is a flag contained in the PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR03L, PR10L, PR10H, PR11L, PR11H, PR12L, PR12H, and PR13L registers.
PR = 00: Specify level 0 with xxPR1x = 0, xxPR0x = 0 (higher priority level)
PR = 01: Specify level 1 with xxPR1x = 0, xxPR0x = 1
PR = 10: Specify level 2 with xxPR1x = 1, xxPR0x = 0
PR = 11: Specify level 3 with xxPR1x = 1, xxPR0x = 1 (lower priority level)

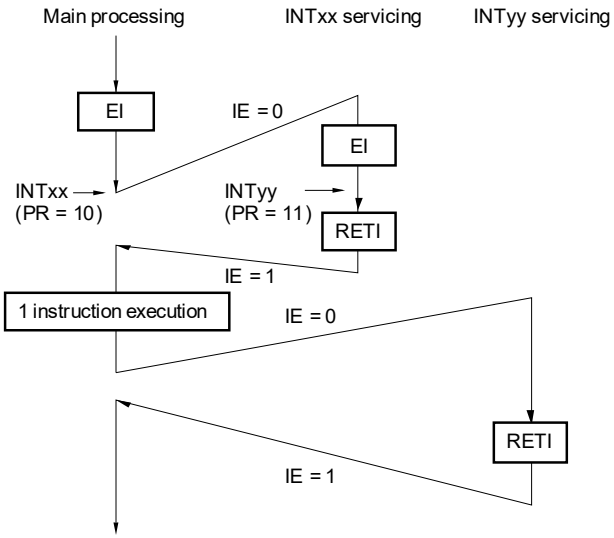
Figure 14 - 10 Examples of Multiple Interrupt Servicing (1/2)

Example 1. Multiple interrupt servicing occurs twice



During servicing of interrupt INTxx, two interrupt requests, INTyy and INTzz, are acknowledged, and multiple interrupt servicing takes place. Before each interrupt request is acknowledged, the EI instruction must always be issued to enable interrupt request acknowledgment.

Example 2. Multiple interrupt servicing does not occur due to priority control

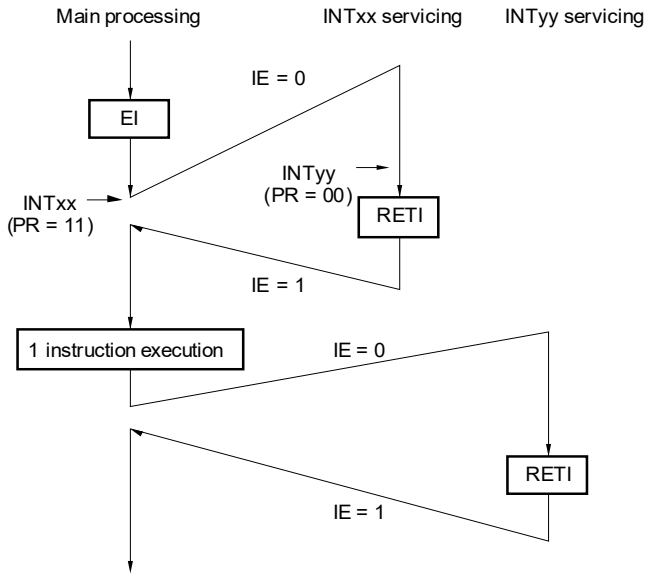


Interrupt request INTyy issued during servicing of interrupt INTxx is not acknowledged because its priority is lower than that of INTxx, and multiple interrupt servicing does not take place. The INTyy interrupt request is held pending, and is acknowledged following execution of one main processing instruction.

Remark PR = 00: Specify level 0 with xxPR1x = 0, xxPR0x = 0 (higher priority level)
PR = 01: Specify level 1 with xxPR1x = 0, xxPR0x = 1
PR = 10: Specify level 2 with xxPR1x = 1, xxPR0x = 0
PR = 11: Specify level 3 with xxPR1x = 1, xxPR0x = 1 (lower priority level)
IE = 0: Interrupt request acknowledgment is disabled.
IE = 1: Interrupt request acknowledgment is enabled.

Figure 14 - 10 Examples of Multiple Interrupt Servicing (2/2)

Example 3. Multiple interrupt servicing does not occur because interrupts are not enabled



Interrupts are not enabled during servicing of interrupt INTxx (EI instruction is not issued), therefore, interrupt request INTyy is not acknowledged and multiple interrupt servicing does not take place. The INTyy interrupt request is held pending, and is acknowledged following execution of one main processing instruction.

Remark PR = 00: Specify level 0 with xxPR1x = 0, xxPR0x = 0 (higher priority level)
PR = 01: Specify level 1 with xxPR1x = 0, xxPR0x = 1
PR = 10: Specify level 2 with xxPR1x = 1, xxPR0x = 0
PR = 11: Specify level 3 with xxPR1x = 1, xxPR0x = 1 (lower priority level)
IE = 0: Interrupt request acknowledgment is disabled
IE = 1: Interrupt request acknowledgment is enabled.

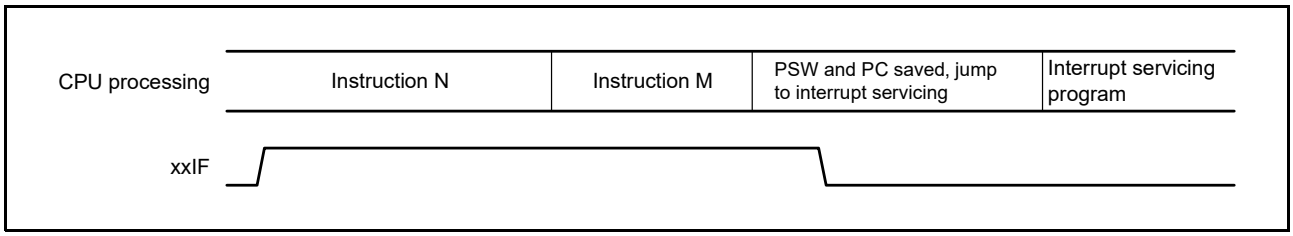
14.4.4 Interrupt request held pending

There are instructions where, even if an interrupt request is issued while the instructions are being executed, interrupt request acknowledgment is held pending until the end of execution of the next instruction. These instructions (instructions that hold interrupt requests pending) are listed below.

- MOV PSW, #byte
- MOV PSW, A
- MOV1 PSW. bit, CY
- SET1 PSW. bit
- CLR1 PSW. bit
- RETB
- RETI
- POP PSW
- BTCLR PSW. bit, $addr20
- EI
- DI
- SKC
- SKNC
- SKZ
- SKNZ
- SKH
- SKNH
- Write instructions for the IF0L, IF0H, IF1L, IF1H, IF2L, IF2H, IF3L, MK0L, MK0H, MK1L, MK1H, MK2L, MK2H, MK3L, PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR03L, PR10L, PR10H, PR11L, PR11H, PR12L, PR12H, and PR13L registers

Figure 14 - 11 shows the timing at which interrupt requests are held pending.

Figure 14 - 11 Holding Interrupt Requests Pending



Remark 1. Instruction N: Instruction to hold interrupt requests pending
Remark 2. Instruction M: Instruction other than the instruction to hold interrupt requests pending

Section 15 Key Interrupt Function

The number of key interrupt input channels differs, depending on the product.

	20- and 32-pin products	44-pin products	48-pin products	64-pin products
Number of the key interrupt input channels	—	4	6	8

15.1 Functions of the Key Interrupt

A key interrupt (INTKR) can be generated by inputting a rising edge/falling edge to the key interrupt input pins (KR0 to KR7).

Table 15 - 1 Assignment of the Key Interrupt Detection Pins

Key interrupt pins	Key return mode register 0 (KRM0)
KR0	KRM00
KR1	KRM01
KR2	KRM02
KR3	KRM03
KR4	KRM04
KR5	KRM05
KR6	KRM06
KR7	KRM07

Remark Pins KR0 to KR3 are only present in the 44-pin products.
Pins KR0 to KR5 are only present in the 48-pin products.
Pins KR0 to KR7 are only present in the 64-pin products.

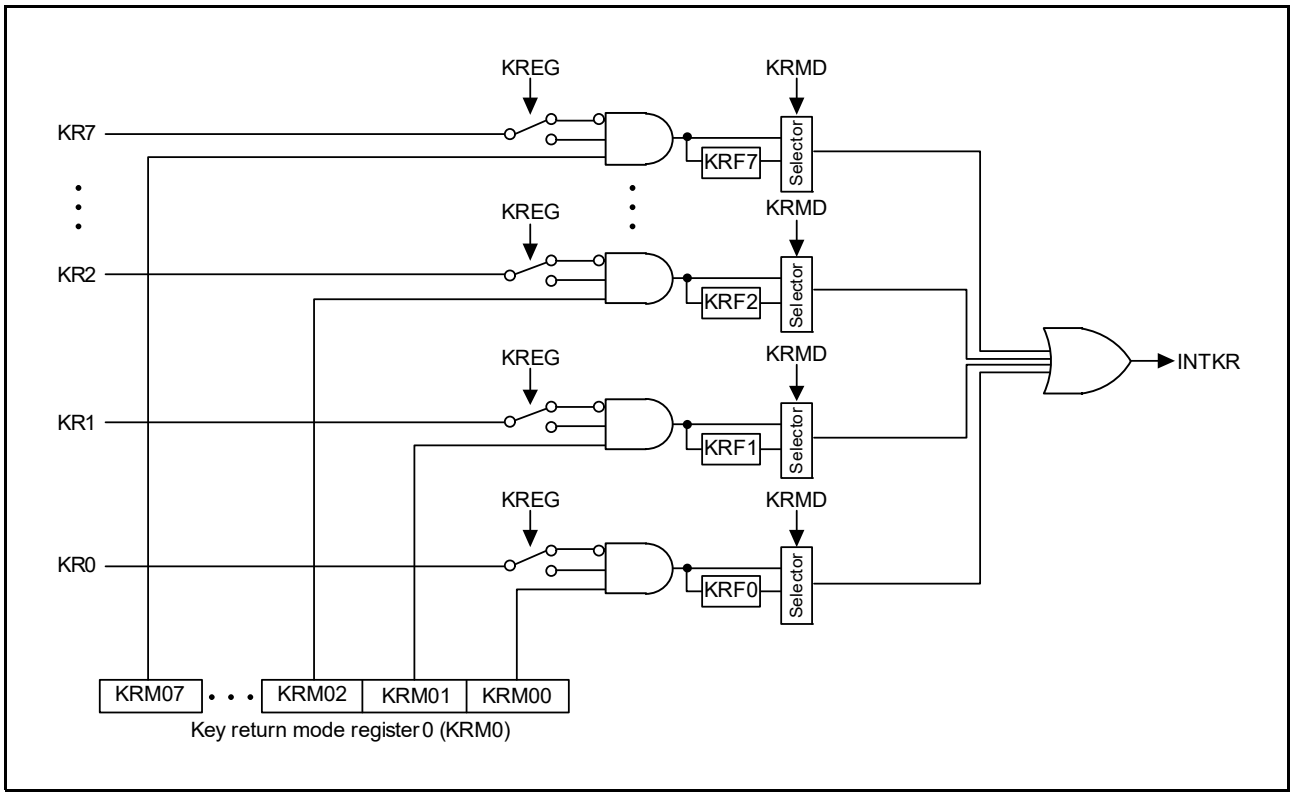
15.2 Configuration of the Key Interrupt

The key interrupt includes the following hardware blocks.

Table 15 - 2 Configuration of the Key Interrupt

Item	Configuration
Control registers	• Key return control register (KRCTL) • Key return mode register 0 (KRM0) • Key return flag register (KRF) • Port mode registers (PMxx)

Figure 15 - 1 Block Diagram of the Key Interrupt



Remark Pins KR0 to KR3 are only present in the 44-pin products.
Pins KR0 to KR5 are only present in the 48-pin products.
Pins KR0 to KR7 are only present in the 64-pin products.

15.3 Registers for Controlling the Key Interrupt

The following registers are used to control the key interrupt.

- Key return control register (KRCTL)
- Key return mode register 0 (KRM0)
- Key return flag register (KRF)
- Port mode registers (PMxx)

15.3.1 Key return control register (KRCTL)

The KRCTL register controls the usage of the key return flags (KRF0 to KRF7) and sets the detection edge. The KRCTL register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 15 - 2 Format of Key Return Control Register (KRCTL)

Address: FFF34H
After reset: 00H
R/W: R/W

Symbol	<7>	6	5	4	3	2	1	<0>
KRCTL	KRMD	0	0	0	0	0	0	KREG

KRMD	Usage of key return flags (KRF0 to KRF7)
0	Does not use key return flags
1	Uses key return flags

KREG	Selection of detection edge (KR0 to KR7)
0	Falling edge
1	Rising edge

15.3.2 Key return mode register 0 (KRM0)

The KRM0 register controls the KR0 to KR7 signals. The KRM0 register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 15 - 3 Format of Key Return Mode Register 0 (KRM0)

Address: FFF37H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
KRM0	KRM07	KRM06	KRM05	KRM04	KRM03	KRM02	KRM01	KRM00

KRM0n	Key interrupt mode control
0	Does not detect key interrupt signal
1	Detects key interrupt signal

Caution 1. Be sure to leave the following bits set to 0.
Bits 7 to 0 in the 20- and 32-pin products
Bits 7 to 4 in the 44-pin products
Bits 7 and 6 in the 48-pin products

Caution 2. The on-chip pull-up resistor can be applied by setting the corresponding key interrupt input pins (bits) in pull-up resistor option register (PU7) to 1.

Caution 3. An interrupt will be generated if the target bit of the KRM0 register is set while a low level (the KREG bit is set to 0) or a high level (KREG is set to 1) is being input to the key interrupt input pin. To ignore this interrupt, set the KRM0 register after disabling interrupt servicing by using the interrupt mask flag. Once the key interrupt input high- and low-level widths elapse, clear the interrupt request flag and enable interrupt servicing. For the key interrupt input high- and low-level widths, see 28.4 AC Characteristics.

Caution 4. The pins not used in the key interrupt mode can be used as normal port pins.

Remark n = 0 to 7

15.3.3 Key return flag register (KRF)

The KRF register controls the key interrupt flags (KRF0 to KRF7). The KRF register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 15 - 4 Format of Key Return Flag Register (KRF)

Address: FFF35H
After reset: 00H
R/W: R/W**Note**

Symbol	7	6	5	4	3	2	1	0
KRF	KRF7	KRF6	KRF5	KRF4	KRF3	KRF2	KRF1	KRF0

KRFn	Key interrupt flag
0	No key interrupt signal has been detected.
1	A key interrupt signal has been detected.

Note Writing 1 has no effect. To clear the KRFn bit, write 0 to the corresponding bit and 1 to the other bits using an 8-bit memory manipulation instruction.

Remark n = 0 to 7

15.3.4 Registers for controlling the port functions multiplexed with the key interrupt inputs

Set the following registers to control the port functions multiplexed with the key interrupt inputs.

- Port mode registers (PMxx)

For details, see **4.3.1 Port mode registers (PMxx)**. For examples of register settings for controlling port functions, see **4.5.4 Examples of register settings for port and alternate functions**.

Section 16 Standby Function

16.1 Standby Function

The standby function reduces the operating current of the system, and the following three modes are available.

1. HALT mode
   Executing a HALT instruction places this LSI chip in the HALT mode. In the HALT mode, the CPU operating clock is stopped. If the high-speed system clock oscillator, high-speed on-chip oscillator, or subsystem clock oscillator is operating before the HALT mode is set, oscillation of each clock continues. In this mode, the operating current is not decreased as much as in the STOP mode, but the HALT mode is effective for restarting operation immediately upon interrupt request generation and carrying out intermittent operations frequently.

2. STOP mode
   Executing a STOP instruction places this LSI chip in the STOP mode. In the STOP mode, the high-speed system clock oscillator, and high-speed on-chip oscillator stop, stopping the whole system, thereby considerably reducing the CPU operating current. Because this mode can be released by an interrupt request, it enables intermittent operations. However, because a wait time is required to secure the oscillation stabilization time after the STOP mode is released when the X1 clock is selected, select the HALT mode if it is necessary to start processing immediately upon interrupt request generation.

3. SNOOZE mode
   In SNOOZE mode, this MCU is released from STOP mode and the following peripheral modules can operate without CPU intervention. For details, see the sections on the individual modules.
   - **Section 10 A/D Converter (ADC)**
   - **Section 11 Serial Array Unit (SAU)**

   This can only be specified when the high-speed on-chip oscillator is selected for the CPU/peripheral hardware clock (f_{CLK}).

In any of these modes, registers, flags and data memory retain all values they had immediately before the transition to the standby mode. The states of the output latches and the output buffers for I/O port pins are also retained.

Caution 1. Shifting to the STOP mode is only enabled when the CPU is operating with the main system clock. Do not execute the STOP instruction while the CPU operates with the subsystem clock. Shifting to the HALT mode is enabled whether the CPU is operating with the main system clock or the subsystem clock.

Caution 2. When shifting to the STOP mode, be sure to stop the peripheral modules operating with the main system clock before executing the STOP instruction. Note that this does not apply to the peripheral modules to be used in the SNOOZE mode; that is, operations of these modules should not be stopped.

Caution 3. It can be selected by the WDTON bit of the option byte or the WUTMMCK0 bit of the subsystem clock supply mode control register (OSMC) whether the low-speed on-chip oscillator continues oscillating or stops in the HALT or STOP mode. For details, see 6.1 Functions of Clock Generator, 2. Subsystem clock, <1> Low-speed on-chip oscillator.

16.2 Registers for Controlling the Standby Function

The following registers are used to control the standby function.

- Subsystem clock supply mode control register (OSMC)
- Oscillation stabilization time counter status register (OSTC)
- Oscillation stabilization time select register (OSTS)

Remark For details of the OSMC, OSTC, and OSTS registers, see **Section 6 Clock Generator**. For registers which control the SNOOZE mode, see the following sections.

- **Section 10 A/D Converter (ADC)**
- **Section 11 Serial Array Unit (SAU)**

16.3 Standby Operation

16.3.1 HALT mode

1. Setting of and states in the HALT mode
   The HALT mode is set by executing the HALT instruction. HALT mode can be set regardless of whether the CPU clock before the setting was the high-speed system clock, high-speed on-chip oscillator clock, or subsystem clock.

   Caution The interrupt request signal is used for release from the HALT mode, so if the interrupt mask flag is 0 (enabling the interrupt processing) and the interrupt request flag is 1 (the interrupt request signal is being generated), the HALT mode is not entered if these are the settings even when a HALT instruction is executed.

Table 16 - 1 and **Table 16 - 3** show the operating states in the HALT mode.

Table 16 - 1 Operating States in HALT Mode When the CPU Clock Is Used as the Main System Clock (1/2)

HALT Mode Setting / Item			When a HALT Instruction is Executed While the CPU is Operating with the Main System Clock		
			When the CPU is Operating with the High-speed On-chip Oscillator Clock (f_{IH})	When the CPU is Operating with the X1 Clock (f_X)	When the CPU is Operating with the External Main System Clock (f_{EX})
System clock			Clock supply to the CPU is stopped		
	Main system clock	f_{IH}	Operation continues (stopping this clock signal is not possible)	Operation disabled	
		f_X	Operation disabled	Operation continues (stopping this clock signal is not possible)	Cannot operate
		f_{EX}		Cannot operate	Operation continues (stopping this clock signal is not possible)
	Subsystem clock	f_{IL}	Set by the WDSTBYON and WDTON bits of the user option byte (000C0H), and the WUTMMCK0 bit of the subsystem clock supply mode control register (OSMC) WUTMMCK0 = 1 or SELLOSC = 1: Oscillates WUTMMCK0 = 0, SELLOSC = 0, and WDTON = 0: Stopped WUTMMCK0 = 0, SELLOSC = 0, WDTON = 1, and WDSTBYON = 1: Oscillates WUTMMCK0 = 0, SELLOSC = 0, WDTON = 1, and WDSTBYON = 0: Stopped		
CPU			Operation stopped		
Code flash memory					
Data flash memory					
RAM					
Port (latch)			Retains the state before the transition to HALT mode		
Timer array unit			Operation enabled		
32-bit interval timer					
Watchdog timer			See **Section 9 Watchdog Timer (WDT)**.		
A/D converter			Operation enabled		
Serial array unit					
Serial interface IICA					
Serial interface UARTA					
Power-on-reset					
Voltage detection					
External interrupt					
Key interrupt					

Table 16 - 1 Operating States in HALT Mode When the CPU Clock Is Used as the Main System Clock (2/2)

HALT Mode Setting / Item		When a HALT Instruction is Executed While the CPU is Operating with the Main System Clock		
		When the CPU is Operating with the High-speed On-chip Oscillator Clock (f_{IH})	When the CPU is Operating with the X1 Clock (fx)	When the CPU is Operating with the External Main System Clock (f_{EX})
CRC operation	High-speed CRC	Operation enabled		
	General-purpose CRC	Operation stopped		
Illicit memory access detection				
RAM parity error detection				
RAM guard function				
SFR guard function				

Remark 1. Operation stopped: Operation is automatically stopped at the time of switching to the HALT mode.
Operation disabled: Stop operation before switching to the HALT mode.

Remark 2. f_{IH}: High-speed on-chip oscillator clock f_{IL}: Low-speed on-chip oscillator clock
f_{EX}: External main system clock fx: X1 clock

Table 16 - 2 Operating States in HALT Mode When the CPU Clock Is Used as the Subsystem Clock

Item \ HALT Mode Setting			When a HALT Instruction is Executed While the CPU is Operating with the Subsystem Clock
			When the CPU is Operating with the Low-speed On-chip Oscillator Clock (f_{IL})
System clock			Clock supply to the CPU is stopped
	Main system clock	f_{IH}	Operation disabled
		f_{IM}	
		f_X	
		f_{EX}	
	Subsystem clock	f_{IL}	Operation continues (stopping this clock signal is not possible)
CPU			Operation stopped
Code flash memory			
Data flash memory			
RAM			
Port (latch)			Retains the state before the transition to HALT mode
Timer array unit			Operation enabled
32-bit interval timer			
Watchdog timer			See **Section 9 Watchdog Timer (WDT)**.
A/D converter			Operation disabled
Serial array unit			Operation enabled
Serial interface IICA			Operation disabled
Serial interface UARTA			
Power-on-reset			Operation enabled
Voltage detection			
External interrupt			
Key interrupt			
CRC operation		High-speed CRC	Operation disabled
		General-purpose CRC	Operation stopped
Illicit memory access detection			
RAM parity error detection			
RAM guard function			
SFR guard function			

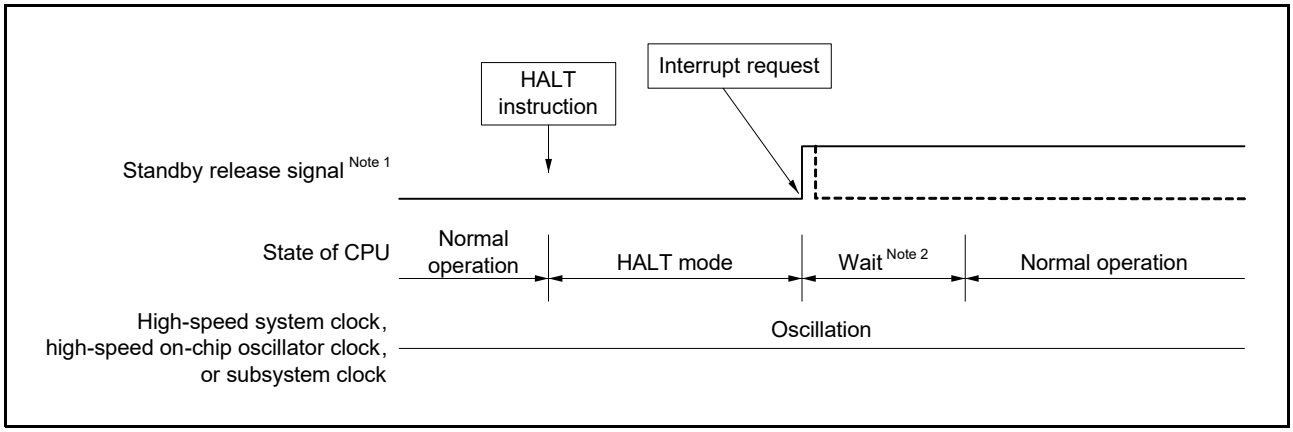
Remark 1. Operation stopped: Operation is automatically stopped at the time of switching to the HALT mode.
Operation disabled: Stop operation before switching to the HALT mode.

Remark 2. f_{IH}: High-speed on-chip oscillator clock f_{IL}: Low-speed on-chip oscillator clock
f_{EX}: External main system clock f_X: X1 clock

2. HALT mode release
   The HALT mode can be released by the following two sources.

a) Release by a non-masked interrupt request
   When a non-masked interrupt request is generated, the HALT mode is released. If interrupt acknowledgment is enabled, vectored interrupt servicing proceeds. If interrupt acknowledgment is disabled, the instruction at the next address is executed.

Figure 16 - 1 HALT Mode Release by Interrupt Request Generation



Note 1. For details of the standby release signal, see **Figure 14 - 1 Basic Configuration of Interrupt Function**.

Note 2. Wait time for HALT mode release
- When vectored interrupt servicing proceeds
  - Main system clock: 15 to 16 clock cycles
  - Subsystem clock: 10 to 11 clock cycles
- When vectored interrupt servicing does not proceed
  - Main system clock: 9 to 10 clock cycles
  - Subsystem clock: 4 to 5 clock cycles

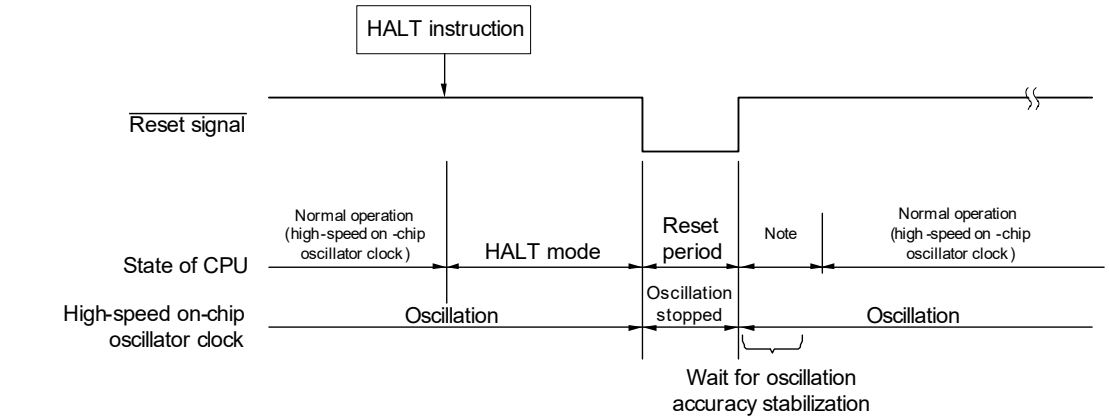
Remark The broken lines indicate the case when the interrupt request which has released the standby mode is acknowledged.

b) Release by reset signal generation

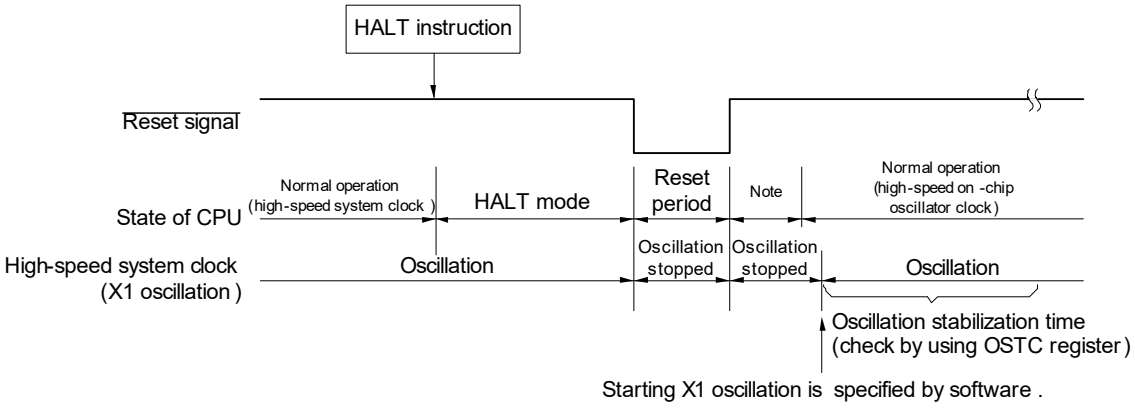
When the reset signal is generated, HALT mode is released, and then, as in the case with a normal reset operation, the program is executed after branching to the reset vector address.

Figure 16 - 2 HALT Mode Release by Reset (1/2)

<1> When high-speed on-chip oscillator clock is used as CPU clock

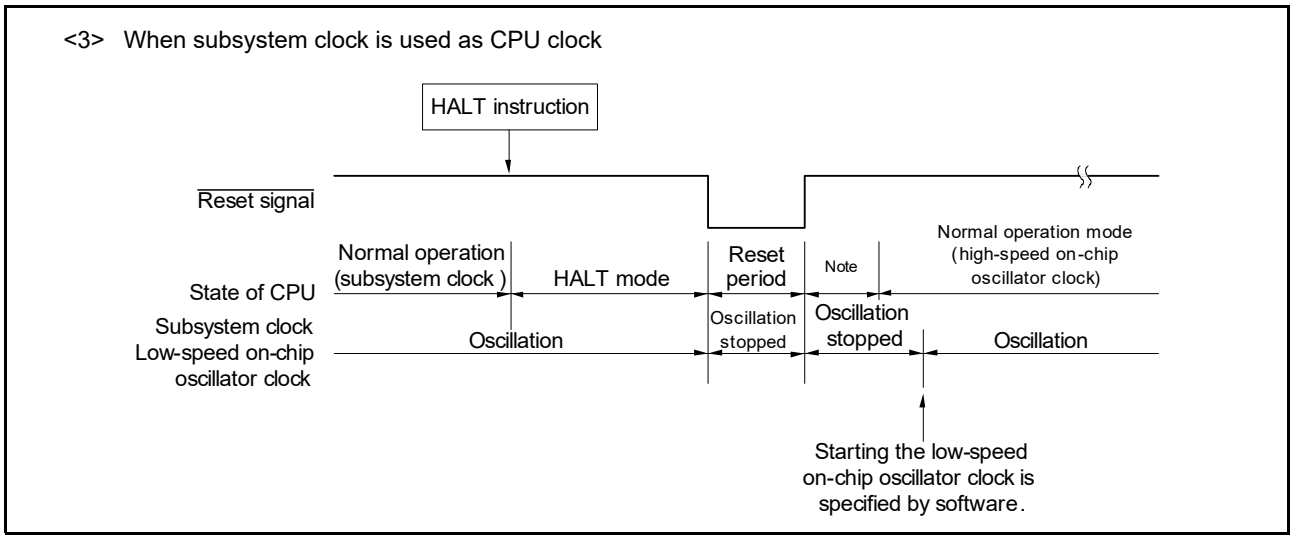


<2> When high-speed system clock is used as CPU clock



Note For the reset processing time, see **Section 17 Reset Function**. For the reset processing time of the power-on-reset circuit (POR) and voltage detectors (LVD0 and LVD1), see **Section 18 Power-on-reset Circuit (POR)**.

Figure 16 - 2 HALT Mode Release by Reset (2/2)



Note For the reset processing time, see **Section 17 Reset Function**. For the reset processing time of the power-on-reset circuit (POR) and voltage detectors (LVD0 and LVD1), see **Section 18 Power-on-reset Circuit (POR)**.

16.3.2 STOP mode

1. STOP mode setting and operating states
   The STOP mode is set by executing the STOP instruction, and it can be set only when the CPU clock before the setting was the main system clock.

Caution The interrupt request signal is used for release from the STOP mode, so if the interrupt mask flag is 0 (enabling the interrupt processing) and the interrupt request flag is 1 (the interrupt request signal is being generated), release from the STOP mode immediately proceeds if these are the settings when a STOP instruction is executed. Accordingly, once the STOP instruction is executed, the system returns to its normal operating mode after the elapse of release time from the STOP mode.

Table 16 - 3 shows the operating states in the STOP mode.

Table 16 - 3 Operating States in STOP Mode (1/2)

Item \ STOP Mode Setting			When a STOP Instruction is Executed While the CPU is Operating with the Main System Clock		
			When the CPU is Operating with the High-speed On-chip Oscillator Clock (f_{IH})	When the CPU is Operating with the X1 Clock (fx)	When the CPU is Operating with the External Main System Clock (f_{EX})
System clock			Clock supply to the CPU is stopped		
	Main system clock	f_{IH}	Stopped		
		f_X	Stopped		
		f_{EX}			
		f_{IL}	Set by the WDSTBYON and WDTON bits of the user option byte (000C0H), and the WUTMMCK0 bit of the subsystem clock supply mode control register (OSMC) WUTMMCK0 = 1 or SELLOSC = 1: Oscillates WUTMMCK0 = 0, SELLOSC = 0, and WDTON = 0: Stopped WUTMMCK0 = 0, SELLOSC = 0, WDTON = 1, and WDSTBYON = 1: Oscillates WUTMMCK0 = 0, SELLOSC = 0, WDTON = 1, and WDSTBYON = 0: Stopped		
CPU			Operation disabled		
Code flash memory					
Data flash memory					
RAM					
Port (latch)			Retains the state before the transition to STOP mode		
Timer array unit			Operation stopped		
32-bit interval timer			Capable of operation when f_{SXP} is selected		
Watchdog timer			See **Section 9 Watchdog Timer (WDT)**.		
A/D converter			Wakeup operation is enabled (switching to SNOOZE mode)		
Serial array unit			Wakeup operation is enabled only for CSIp and UARTq (switching to SNOOZE mode) Operation of the types of modules other than CSIp and UARTq is disabled.		
Serial interface IICA			Capable of waking up in response to address matching		
Serial interface UARTA			Operation disabled		
Power-on-reset			Operation enabled		
Voltage detection					
External interrupt					
Key interrupt					

Table 16 - 3 Operating States in STOP Mode (2/2)

Item \ STOP Mode Setting		When a STOP Instruction is Executed While the CPU is Operating with the Main System Clock		
		When the CPU is Operating with the High-speed On-chip Oscillator Clock (f_{IH})	When the CPU is Operating with the X1 Clock (f_X)	When the CPU is Operating with the External Main System Clock (f_{EX})
CRC operation	High-speed CRC	Operation stopped		
	General-purpose CRC			
Illicit memory access detection				
RAM parity error detection				
RAM guard function				
SFR guard function				

Remark 1. Operation stopped: Operation is automatically stopped at the time of switching to the STOP mode.
Operation disabled: Stop operation before switching to the STOP mode.

Remark 2. f_{IH}: High-speed on-chip oscillator clock f_X: X1 clock
f_{EX}: External main system clock f_{SXP}: Low-speed peripheral clock
f_{IL}: Low-speed on-chip oscillator clock

Remark 3. 20- to 64-pin products: p = 00, q = 0

2. STOP mode release

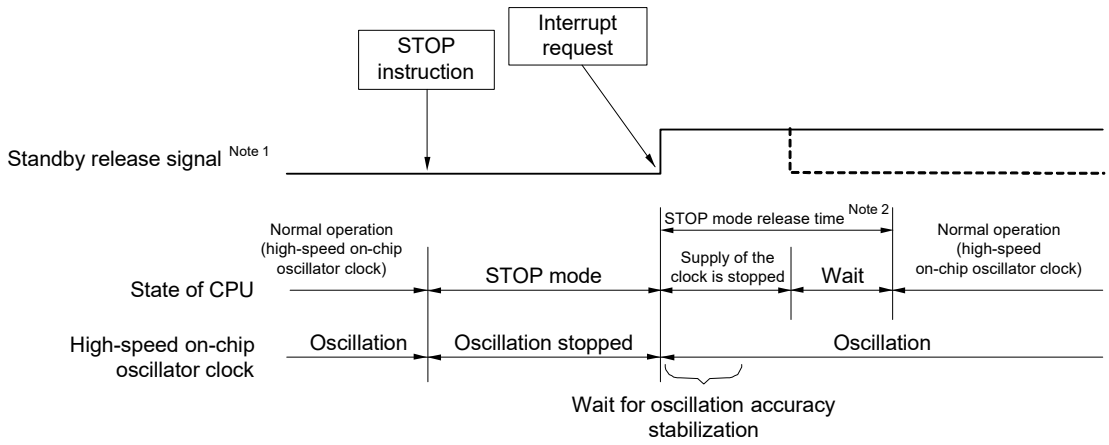
The STOP mode can be released by the following two sources.

a) Release by a non-masked interrupt request

When a non-masked interrupt request is generated, the STOP mode is released. After the oscillation stabilization time has elapsed, if interrupt acknowledgment is enabled, vectored interrupt servicing proceeds. If interrupt acknowledgment is disabled, the instruction at the next address is executed.

Figure 16 - 3 STOP Mode Release by Interrupt Request Generation (1/3)

<1> When high-speed on-chip oscillator clock is used as CPU clock



Note 1. For details of the standby release signal, see **Figure 14 - 1 Basic Configuration of Interrupt Function**.

Note 2. STOP mode release time

- Supply of the clock is stopped: 4.4 to 5.4 μs + 3 to 4 clock cycles
- Wait:
  - When vectored interrupt servicing proceeds: 7 clock cycles
  - When vectored interrupt servicing does not proceed: 1 clock cycle

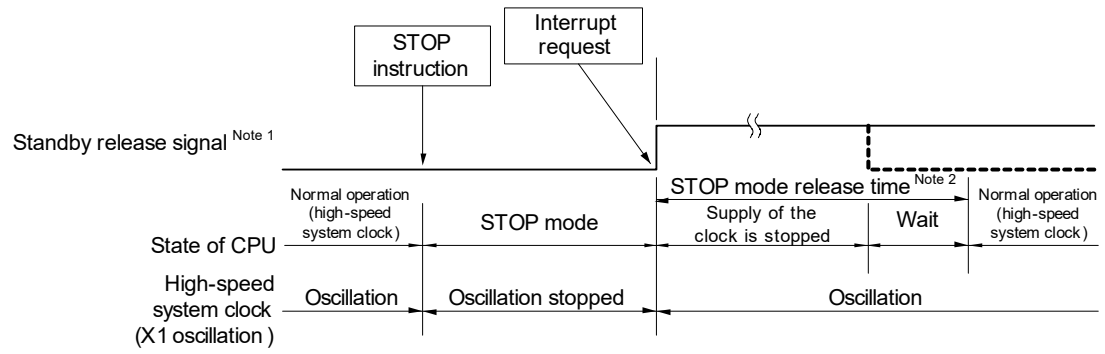
Caution **To shorten oscillation stabilization time after the STOP mode is released when the CPU operates with the high-speed system clock (X1 oscillation), temporarily switch the CPU clock to the high-speed on-chip oscillator clock before the execution of the STOP instruction.**

Remark 1. The clock supply stop time varies depending on the temperature conditions and STOP mode period.

Remark 2. The broken lines indicate the case when the interrupt request that has released the standby mode is acknowledged.

Figure 16 - 3 STOP Mode Release by Interrupt Request Generation (2/3)

<2> When high-speed system clock (X1 oscillation) is used as CPU clock



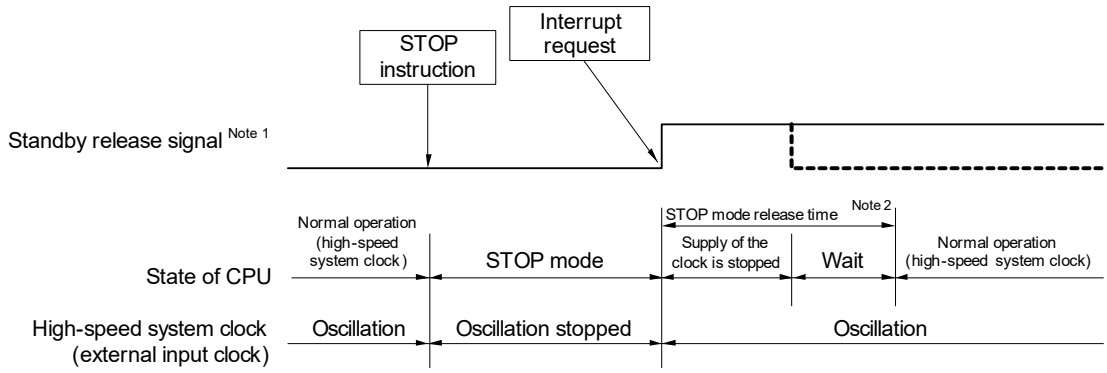
Note 1. For details of the standby release signal, see **Figure 14 - 1 Basic Configuration of Interrupt Function**.

Note 2. STOP mode release time

- Supply of the clock is stopped: Oscillation stabilization time (set by the OSTS register) + 3 to 4 clock cycles
- Wait:
  - When vectored interrupt servicing proceeds: 7 clock cycles
  - When vectored interrupt servicing does not proceed: 1 clock cycle

Figure 16 - 3 STOP Mode Release by Interrupt Request Generation (3/3)

<3> When high-speed system clock (external clock input) is used as CPU clock



Note 1. For details of the standby release signal, see **Figure 14 - 1 Basic Configuration of Interrupt Function**.

Note 2. STOP mode release time

- Supply of the clock is stopped: 50 to 51 cycles of the external clock
- Wait:
  - When vectored interrupt servicing proceeds: 7 clock cycles
  - When vectored interrupt servicing does not proceed: 1 clock cycle

Caution To reduce the oscillation stabilization time after release from the STOP mode while the CPU operates with the high-speed system clock (X1 oscillation), switch the clock to the high-speed on-chip oscillator clock temporarily before executing the STOP instruction.

Remark 1. The clock supply stop time varies depending on the temperature conditions and STOP mode period.

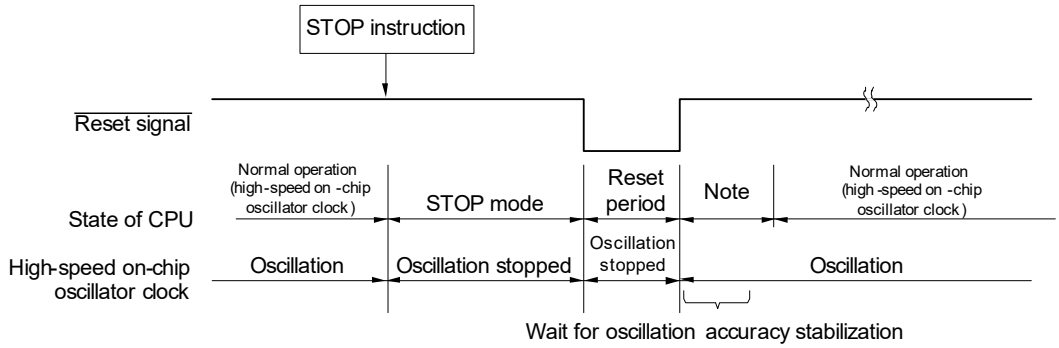
Remark 2. The broken lines indicate the case when the interrupt request that has released the standby mode is acknowledged.

b) Release by reset signal generation

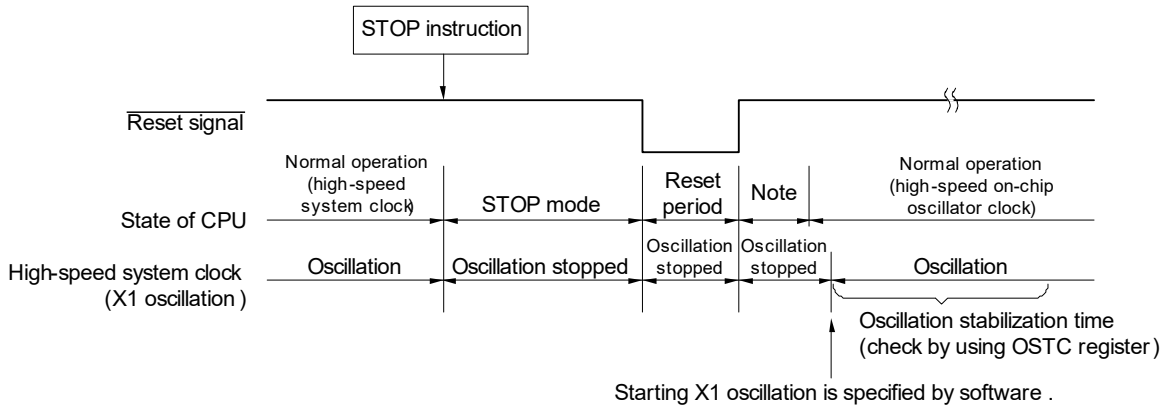
When the reset signal is generated, STOP mode is released, and then, as in the case with a normal reset operation, the program is executed after branching to the reset vector address.

Figure 16 - 4 STOP Mode Release by Reset

<1> When high-speed on-chip oscillator clock is used as CPU clock



<2> When high-speed system clock is used as CPU clock



Note For the reset processing time, see **Section 17 Reset Function**. For the reset processing time of the power-on-reset circuit (POR) and voltage detectors (LVD0 and LVD1), see **Section 18 Power-on-reset Circuit (POR)**.

16.3.3 SNOOZE mode

1. SNOOZE mode setting and operating states
   This MCU can be placed in SNOOZE mode, in which operation of the following peripheral modules is selectable. For details, see the sections on the individual modules.
   - **Section 10 A/D Converter (ADC)**
   - **Section 11 Serial Array Unit (SAU)**

   This MCU can also be placed in SNOOZE mode if the CPU clock before entry to SNOOZE mode is the high-speed on-chip oscillator clock.

   For transitions to SNOOZE mode, the following intervals of waiting are inserted.

<R>
   - Transition time from STOP mode to SNOOZE mode: 4.4 to 5.4 μs

   - Transition time from SNOOZE mode to normal operation:
     – When vectored interrupt servicing proceeds: 0.3 to 0.4 μs + 10 to 11 clock cycles
     – When vectored interrupt servicing does not proceed: 0.3 to 0.4 μs + 4 to 5 clock cycles

Caution The transition time from STOP mode to SNOOZE mode varies depending on the temperature conditions and the STOP mode period.

Table 16 - 4 shows the operating states in the SNOOZE mode.

Table 16 - 4 Operating States in SNOOZE Mode

Item \ STOP Mode Setting			Generation of the Source Conditions Which Lead to Transitions to SNOOZE Mode during STOP Mode
			When the CPU is Operating with the High-speed On-chip Oscillator Clock (f_{IH})
System clock			Clock supply to the CPU is stopped
	Main system clock	f_{IH}	Operation started
		f_X	Stopped
		f_{EX}	
		f_{IL}	Set by the WDSTBYON and WDTON bits of the user option byte (000C0H), and the WUTMMCK0 bit of the subsystem clock supply mode control register (OSMC) WUTMMCK0 = 1 or SELLOSC = 1: Oscillates WUTMMCK0 = 0, SELLOSC = 0, and WDTON = 0: Stopped WUTMMCK0 = 0, SELLOSC = 0, WDTON = 1, and WDSTBYON = 1: Oscillates WUTMMCK0 = 0, SELLOSC = 0, WDTON = 1, and WDSTBYON = 0: Stopped
CPU			Operation stopped
Code flash memory			
Data flash memory			
RAM			
Port (latch)			Retains the state before the transition to SNOOZE mode
Timer array unit			Operation disabled
32-bit interval timer			Capable of operation when f_{SXP} is selected
Watchdog timer			See **Section 9 Watchdog Timer (WDT)**.
A/D converter			Operation enabled
Serial array unit			Only CSIp and UARTq are capable of operation. Operation of the types of modules other than CSIp and UARTq is disabled.
Serial interface IICA			Capable of waking up in response to address matching
Serial interface UARTA			Operation disabled
Power-on-reset			Operation enabled
Voltage detection			
External interrupt			
Key interrupt			
CRC operation		High-speed CRC	Operation stopped
		General-purpose CRC	
Illicit memory access detection			Operation stopped
RAM parity error detection			
RAM guard function			
SFR guard function			

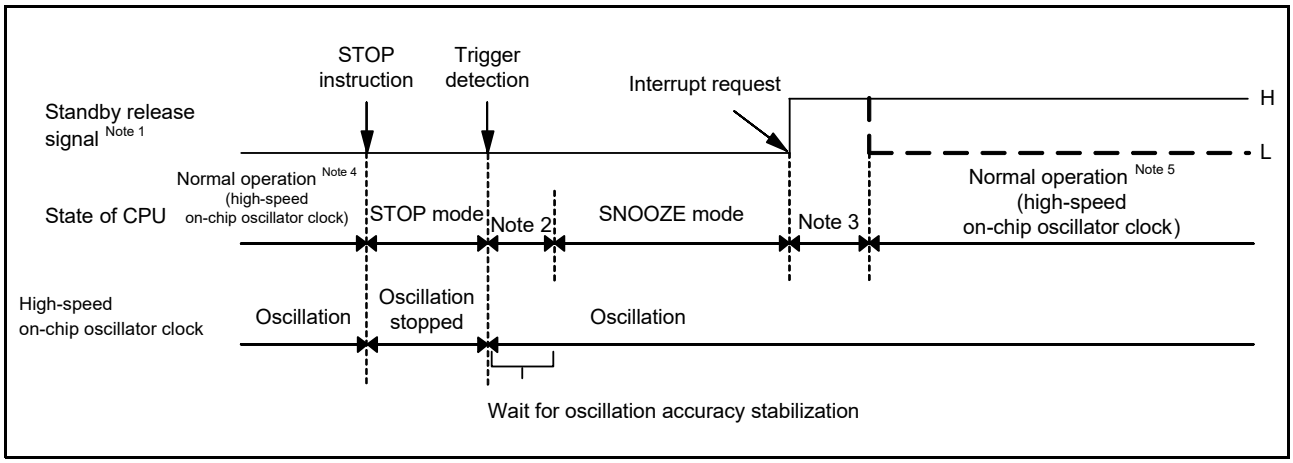
Remark 1. Operation stopped: Operation is automatically stopped at the time of switching to the STOP mode.
Operation disabled: Stop operation before switching to the STOP mode.

Remark 2. f_{IH}: High-speed on-chip oscillator clock f_X: X1 clock
f_{EX}: External main system clock f_{SXP}: Low-speed peripheral clock
f_{IL}: Low-speed on-chip oscillator clock

Remark 3. 20- to 64-pin products: p = 00, q = 0

2. Timing diagram when the interrupt request signal is generated in the SNOOZE mode

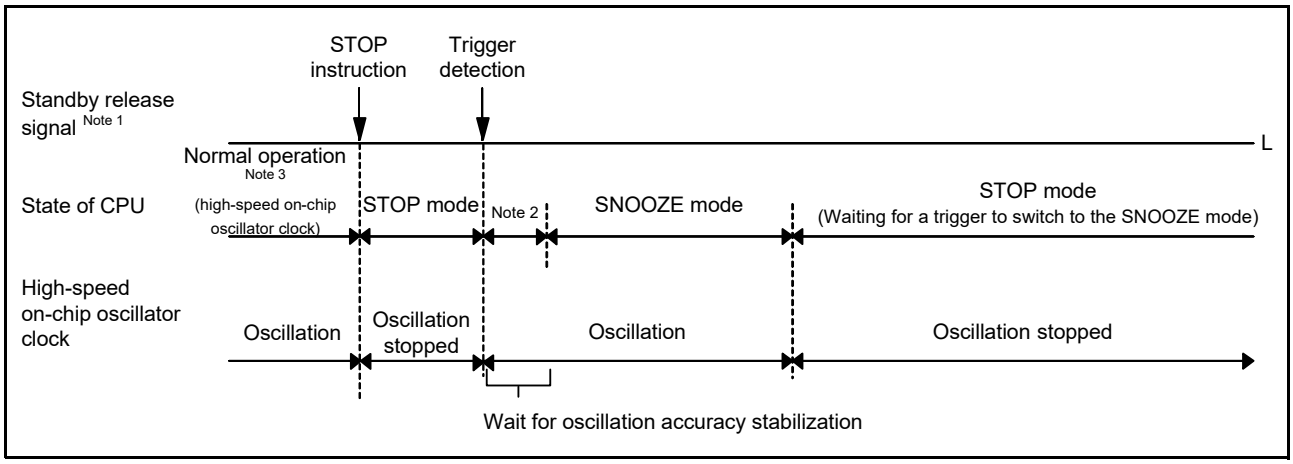
Figure 16 - 5 When the Interrupt Request Signal is Generated in the SNOOZE Mode



Note 1. For details of the standby release signal, see **Figure 14 - 1 Basic Configuration of Interrupt Function**.
Note 2. Transition time from STOP mode to SNOOZE mode
Note 3. Transition time from SNOOZE mode to normal operation
Note 4. Enable the SNOOZE mode immediately before switching to the STOP mode.
Note 5. Be sure to release the SNOOZE mode immediately after return to the normal operation.

3. Timing diagram when the interrupt request signal is not generated in the SNOOZE mode

Figure 16 - 6 When the Interrupt Request Signal is Not Generated in the SNOOZE Mode



Note 1. For details of the standby release signal, see **Figure 14 - 1 Basic Configuration of Interrupt Function**.
Note 2. Transition time from STOP mode to SNOOZE mode
Note 3. Enable the SNOOZE mode immediately before switching to the STOP mode.

Remark For details on the SNOOZE mode, see the following sections.
- **Section 10 A/D Converter (ADC)**
- **Section 11 Serial Array Unit (SAU)**

Section 17 Reset Function

A reset is triggered by any of the following events.

1. External reset input via the $\overline{\text{RESET}}$ pin
2. Internal reset due to detection of a program malfunction by the watchdog timer
3. Internal reset by comparison of supply voltage and detection voltage of the power-on-reset circuit (POR)
4. Internal reset by comparison of supply voltage and detection voltage of the voltage detectors (LVD0 and LVD1)
5. Internal reset due to execution of an illegal instruction**Note**
6. Internal reset due to a RAM parity error
7. Internal reset due to illegal-memory access

External and internal resets start program execution from the address at 00000H and 00001H when the reset signal is generated. A reset is applied when a low level is input to the $\overline{\text{RESET}}$ pin or the watchdog timer overflows, or on voltage detection by the power-on reset circuit, LVD0, or LVD1, execution of illegal instruction**Note**, generation of a RAM parity error, or illegal-memory access, and each module is set to the state shown in **Table 17 - 1 Operating States during a Reset**.

Note In normal operation, executing the instruction code FFH triggers an internal reset, but this is not the case during emulation by the on-chip debugging emulator.

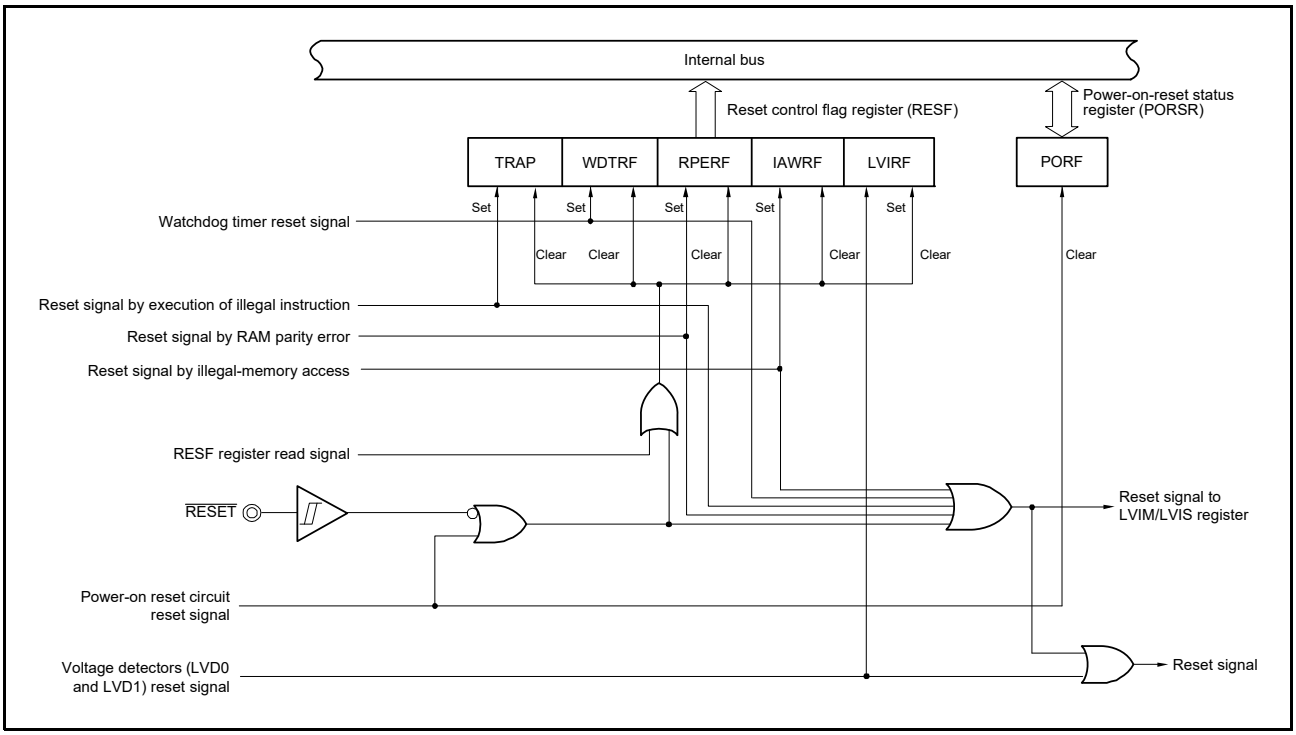
Caution 1. For an external reset, set the PORTSELB bit of the user option byte (000C1H) to 0 to select P125 as the $\overline{\text{RESET}}$ pin, and input a low level for at least 10 μs to it. To perform an external reset upon power application, input a low level to the $\overline{\text{RESET}}$ pin, turn power on, continue to input a low level to the pin for at least 10 μs within the operating voltage range shown in 28.4 AC Characteristics, and then input a high level to the pin.

Caution 2. During generation of a reset signal, the X1 clock, high-speed on-chip oscillator clock, and low-speed on-chip oscillator clock stop oscillating. The external main system clock input becomes invalid.

Caution 3. The SFRs and 2nd SFRs are initialized following a reset, which causes the port pins in the states below.

- **P40: High-impedance during the external reset period or reset period by the POR. High level during other types of reset and after the reset is released (connected to the on-chip pull-up resistor).**
- **P125: Low level during the external reset period (the low-level signal input to the $\overline{\text{RESET}}$ pin). High level during other types of reset and after the reset is released (connected to the on-chip pull-up resistor).**
- **P130: Low-level output during a reset and after the reset is released.**
- **Port pins other than P40, P125, and P130: High-impedance during a reset and after the reset is released.**

Figure 17 - 1 Block Diagram of Reset Function

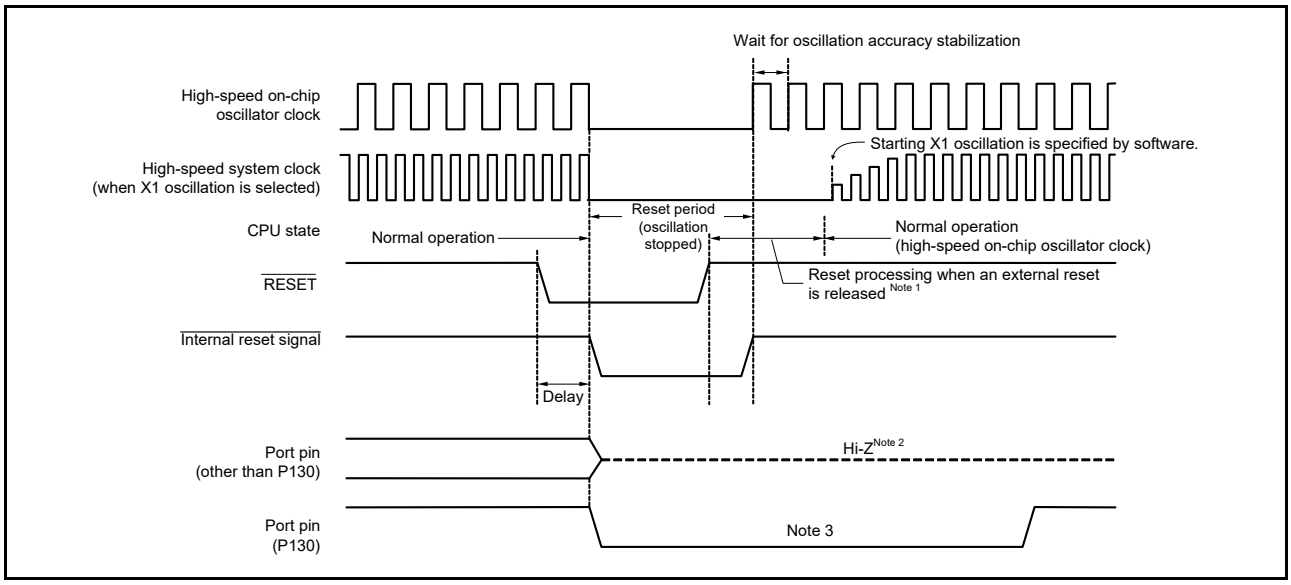


Caution An internal reset triggered by LVD0 does not reset LVD0 itself.

17.1 Timing of Reset Operation

This MCU is reset by input of the low level on the $\overline{\text{RESET}}$ pin and released from the reset state by input of the high level on the $\overline{\text{RESET}}$ pin. After reset processing, execution of the program with the high-speed on-chip oscillator clock as the operating clock starts.

Figure 17 - 2 Timing of Reset by $\overline{\text{RESET}}$ Input



Note 1. Reset times (times for release from the external reset state)
The first external reset following release from the POR state:
- When the LVD is in use: 0.506 ms (typ.), 0.694 ms (max.)
- When the LVD is not in use: 0.201 ms (typ.), 0.335 ms (max.)

The second or subsequent external reset following release from the POR state:
- When the LVD is in use: 0.476 ms (typ.), 0.616 ms (max.)
- When the LVD is not in use: 0.170 ms (typ.), 0.257 ms (max.)

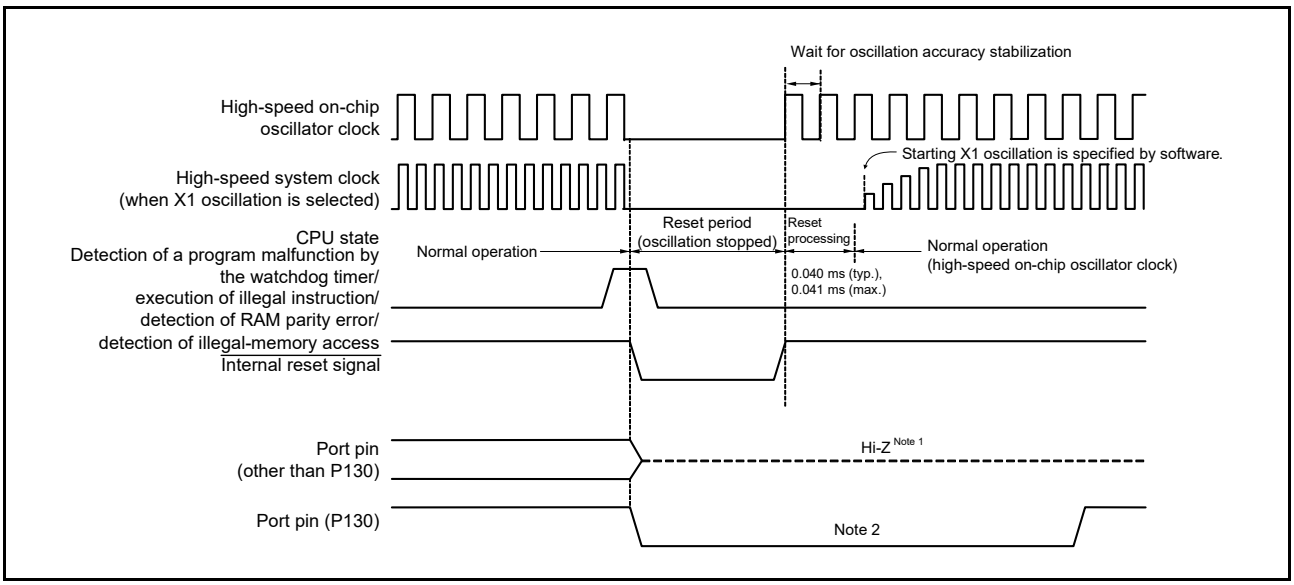
After power is supplied, the following voltage stabilization waiting time is required before reset processing starts after release from the external reset state.
- 4.0 ms (typ.), 9.9 ms (max.)

Note 2. The state of the P40 pin is as follows.
- High-impedance during the external reset period or reset period by the POR.
- High level during other types of reset and after the reset is released (connected to the on-chip pull-up resistor).

Note 3. The P130 pin outputs a low-level signal following the application of a reset. If this pin was in use as a high-level output before the application of a reset, the output signal from the P130 pin can effectively be used as the reset signal for an external device that has an active-low reset signal. To de-assert the reset signal to the external device, set the P130 pin for high-level output by software.

Release from the reset state proceeds automatically in the case of a reset due to detection of a program malfunction by the watchdog timer, execution of an illegal instruction, detection of a RAM parity error, or detection of illegal-memory access. After reset processing, program execution starts with the high-speed on-chip oscillator clock as the operating clock.

Figure 17 - 3 Timing of Reset Due to Detection of a Program Malfunction by the Watchdog Timer, Execution of Illegal Instruction, Detection of RAM Parity Error, or Detection of Illegal-memory Access



Note 1. The states of the P40 and P125 pins are as follows.
- High level during the reset period and after the reset is released (connected to the on-chip pull-up resistor).

Note 2. The P130 pin outputs a low-level signal following the application of a reset. If this pin was in use as a high-level output before the application of a reset, the output signal from the P130 pin can effectively be used as the reset signal for an external device that has an active-low reset signal. To de-assert the reset signal to the external device, set the P130 pin for high-level output by software.

A reset from the power-on reset circuit or by LVD0 voltage detection is released when $V_{DD} \geq V_{POR}$ or $V_{DD} \geq V_{LVD0}$ after the reset. After reset processing, execution of the program starts with the high-speed on-chip oscillator clock as the operating clock. For details, see **Section 18 Power-on-reset Circuit (POR)** or **Section 19 Voltage Detector (LVD)**.

Remark V_{POR}: POR power supply rise detection voltage
V_{LVD0}: LVD0 detection voltage

Table 17 - 1 Operating States during a Reset

Item			Operating State during a Reset
System clock			Clock supply to the CPU is stopped.
	Main system clock	f_{IH}	Operation stopped
		f_{IM}	
		f_X	Operation stopped (the X1 and X2 pins are input port mode)
		f_{EX}	Clock input invalid (the pin is input port mode)
	Subsystem clock	f_{IL}	Operation stopped
CPU			Operation stopped
Code flash memory			Operation stopped
Data flash memory			Operation stopped
RAM			Operation stopped
Port (latch)			High-impedance[Note]
Timer array unit			Operation stopped
32-bit interval timer			Operation stopped
Watchdog timer			
A/D converter			
Serial array unit			
Serial interface IICA			
Serial interface UARTA			
Power-on-reset function			Detection operation possible
Voltage detection function			LVD0: LVD0 operation is possible following an LVD0 reset but is stopped following other types of reset. LVD1: Operation stopped
External interrupt			Operation stopped
Key interrupt function			
CRC operation function		High-speed CRC	
		General-purpose CRC	
Illegal-memory access detection function			
RAM parity error detection function			
RAM guard function			
SFR guard function			

Note The states of the P40, P125, and P130 pin are as follows.

- P40: High-impedance during the external reset period or reset period by the POR. High level during other types of reset (connected to the on-chip pull-up resistor).
- P125: Low level during the external reset period (the low-level signal input to the $\overline{\text{RESET}}$ pin). High level during other types of reset and after the reset is released (connected to the on-chip pull-up resistor).
- P130: Low-level output during the reset period

Remark f_{IH}: High-speed on-chip oscillator clock f_X: X1 oscillation clock
f_{IL}: Low-speed on-chip oscillator clock f_{EX}: External main system clock

Table 17 - 2 States of the Hardware Blocks after a Reset is Released

Hardware		State after a Reset is Released
Program counter (PC)		The contents of the reset vector table (00000H, 00001H) are set.Note
Stack pointer (SP)		Undefined
Program status word (PSW)		06H
RAM	Data memory	Undefined
	General-purpose registers	Undefined

Note The contents of the PC are undefined during a reset and until the clock oscillation becomes stable after the reset is released.

Remark For the states of the special function registers (SFRs) after a reset is released, see **3.1.4 Special function register (SFR) area** and **3.1.5 Extended special function register (2nd SFR: 2nd special function register) area**.

17.2 Registers for Controlling the Reset Function

The following registers are used to control the reset function.

- Reset control flag register (RESF)
- Power-on-reset status register (PORSR)
- Peripheral reset control register 0 (PRR0)
- Peripheral reset control register 1 (PRR1)

17.2.1 Reset control flag register (RESF)

This MCU has multiple internal reset sources. The RESF register indicates which source has generated the reset request. The RESF register can be read by an 8-bit memory manipulation instruction. The flags TRAP, WDTRF, RPERF, IAWRF, and LVIRF are automatically cleared by any of the following event.

- Reset input via the $\overline{\text{RESET}}$ pin
- Reset by the power-on-reset circuit (POR)
- The RESF register is accessed.

Figure 17 - 4 Format of Reset Control Flag Register (RESF)

Address: FFFA8H
After reset: Undefined**Note 1**
R/W: R

Symbol	7	6	5	4	3	2	1	0
RESF	TRAP	0	0	WDTRF	0	RPERF	IAWRF	LVIRF

TRAP	Internal reset request by execution of illegal instruction**Note 2**
0	Internal reset request is not generated, or the RESF register is cleared.
1	Internal reset request is generated.

WDTRF	Internal reset request by watchdog timer (WDT)
0	Internal reset request is not generated, or the RESF register is cleared.
1	Internal reset request is generated.

RPERF	Internal reset request by RAM parity error
0	Internal reset request is not generated, or the RESF register is cleared.
1	Internal reset request is generated.

IAWRF	Internal reset request by illegal-memory access
0	Internal reset request is not generated, or the RESF register is cleared.
1	Internal reset request is generated.

LVIRF	Internal reset request by voltage detector (LVD0 or LVD1)
0	Internal reset request is not generated, or the RESF register is cleared.
1	Internal reset request is generated.

Note 1. The value after reset varies depending on the reset source. See **Table 17 - 3**.

(Note and Cautions continue on the next page.)

Note 2. In normal operation, executing the instruction code FFH triggers an internal reset, but this is not the case during emulation by the on-chip debugging emulator.

Caution 1. Do not read data by a 1-bit memory manipulation instruction.

Caution 2. When enabling RAM parity error resets (RPECTL.RPERDIS = 0), be sure to initialize the used RAM area at data access or the used RAM area + 10 bytes at execution of instruction from the RAM area. Reset generation enables RAM parity error resets (RPERDIS = 0). For details, see 20.3.4 RAM parity error detection.

Table 17 - 3 shows the state of the RESF register when a reset request is generated.

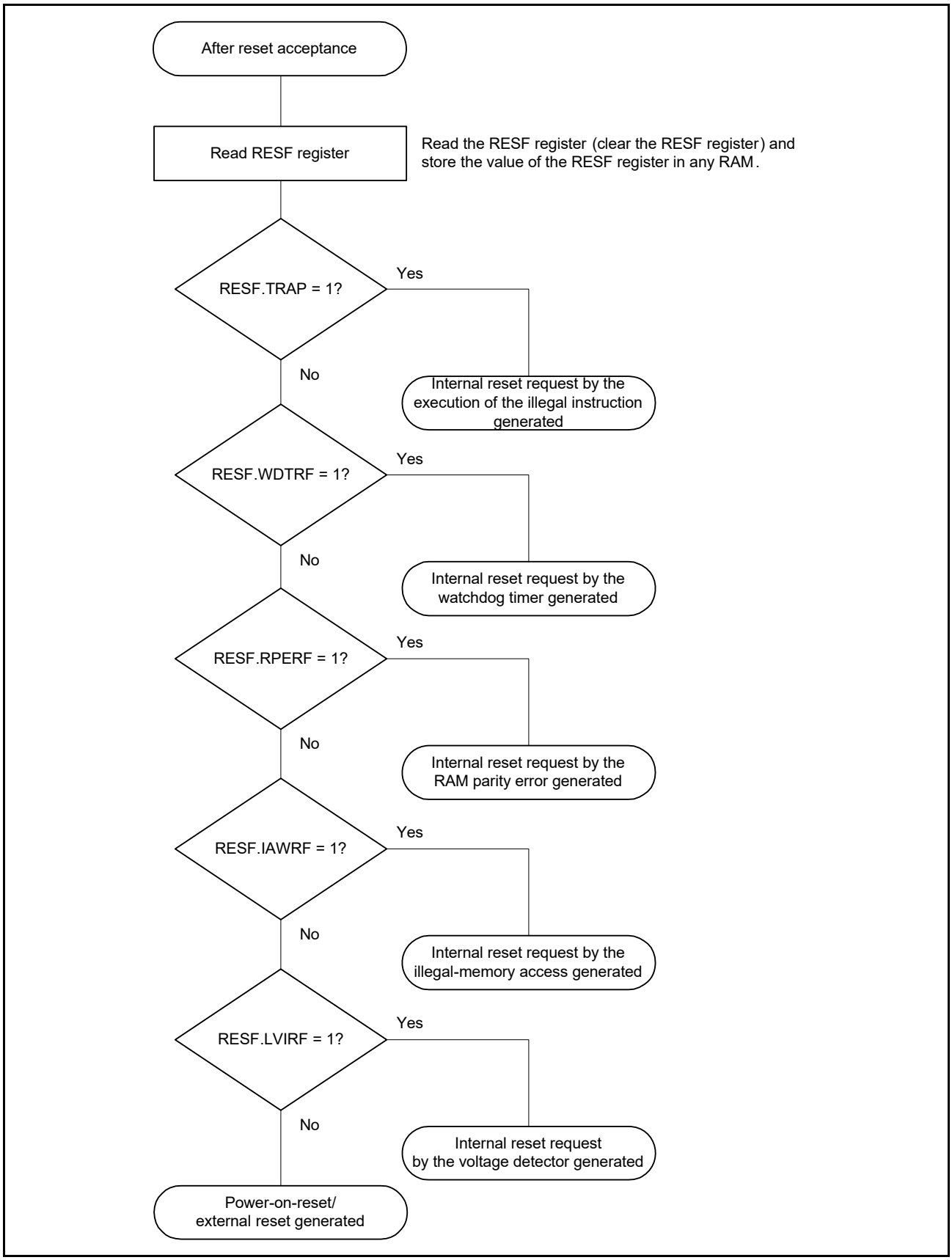
Table 17 - 3 State of the RESF Register when Reset Request is Generated

Reset Source / Flag	$\overline{\text{RESET}}$ Input	Reset by POR	Reset by Execution of Illegal Instruction	Reset by WDT	Reset by RAM Parity Error	Reset by Illegal-memory Access	Reset by LVD0 or LVD1
TRAP	Cleared to 0	Cleared to 0	Set to 1	Retained	Retained	Retained	Retained
WDTRF			Retained	Set to 1			
RPERF				Retained	Set to 1		
IAWRF					Retained	Set to 1	
LVIRF						Retained	Set to 1

The RESF register is automatically cleared after it is read by using an 8-bit memory manipulation instruction.

Figure 17 - 5 shows an example of the procedure for checking a reset source.

Figure 17 - 5 Example of the Procedure for Checking Reset Source



The above flow is an example of the procedure for checking a reset source.

17.2.2 Power-on-reset status register (PORSR)

The PORSR register is used to check the occurrence of a power-on reset. Only writing 1 to bit 0 (PORF) of the PORSR register is effective. Writing 0 to the bit has no effect. Write 1 to the PORF bit in advance to enable checking of the occurrence of a power-on reset. The PORSR register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a power-on reset is 00H.

Caution 1. The PORSR register is reset only by a power-on reset; it retains the value when a reset caused by another source occurs.

Caution 2. The PORF bit being set to 1 guarantees that a power-on reset has not occurred, but does not guarantee that the values in RAM have been retained.

Figure 17 - 6 Format of Power-on-Reset Status Register (PORSR)

Address: F00F9H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	<0>
PORSR	0	0	0	0	0	0	0	PORF

PORF	Checking occurrence of power-on reset
0	A value 1 has not been written, or a power-on reset has occurred.
1	No power-on reset has occurred.

17.2.3 Peripheral reset control register 0 (PRR0)

The PRR0 register is used to control resetting of the on-chip peripheral modules. Each bit in this register controls resetting and release from the reset state of the corresponding on-chip peripheral module. The PRR0 register can be set by a 1-bit or 8-bit memory manipulation instruction The value of this register following a reset is 00H.

Figure 17 - 7 Format of Peripheral Reset Control Register 0 (PRR0)

Address: F00F1H
After reset: 00H
R/W: R/W

Symbol	7	6	<5>	<4>	<3>	<2>	1	<0>
PRR0	0	0	ADCRES	IICA0RES **Note**	SAU1RES	SAU0RES	0	TAU0RES

IICA0RES	Control resetting of the serial interface IICA0
0	The serial interface IICA0 is released from the reset state.
1	The serial interface IICA0 is in the reset state. • The SFRs for use with the serial interface IICA0 are initialized.

ADCRES	Control resetting of the A/D converter
0	The A/D converter is released from the reset state.
1	The A/D converter is in the reset state. • The SFRs for use with the A/D converter are initialized.

SAUnRES	Control resetting of serial array unit n
0	Serial array unit n is released from the reset state.
1	Serial array unit n is in the reset state. • The SFRs for use with serial array unit n are initialized.

TAU0RES	Control resetting of timer array unit 0
0	Timer array unit 0 is released from the reset state.
1	Timer array unit 0 is in the reset state. • The SFRs for use with timer array unit 0 are initialized.

Note This bit is only present in the 32- to 64-pin products.

Caution 1. Be sure to leave bits 7, 6, and 1 set to 0.
Caution 2. Be sure to leave bit 4 in the 20-pin products set to 0.

Remark n = 0, 1

17.2.4 Peripheral reset control register 1 (PRR1)

The PRR1 register is used to control resetting of the on-chip peripheral modules. Each bit in this register controls resetting and release from the reset state of the corresponding on-chip peripheral module. The PRR1 register can be set by a 1-bit or 8-bit memory manipulation instruction The value of this register following a reset is 00H.

Figure 17 - 8 Format of Peripheral Reset Control Register 1 (PRR1)

Address: F00FBH
After reset: 00H
R/W: R/W

Symbol	7	6	5	<4>	3	2	1	0
PRR1	0	0	0	TML32RES	0	0	0	0

TML32RES	Control resetting of the 32-bit interval timer
0	The 32-bit interval timer is released from the reset state.
1	The 32-bit interval timer is in the reset state. • The SFRs for use with the 32-bit interval timer are initialized.

Caution Be sure to leave bits 7 to 5 and 3 set to 0.

Section 18 Power-on-reset Circuit (POR)

18.1 Functions of Power-on-reset Circuit

The power-on-reset circuit (POR) has the following functions.

- Generates internal reset signal at power on.
  The reset signal is released when the supply voltage (V_{DD}) exceeds the detection voltage (V_{POR}). Note that the reset state must be retained until the operating voltage becomes in the range defined in **28.4 AC Characteristics**. This is done by utilizing LVD0 or controlling the externally input reset signal.
- Compares supply voltage (V_{DD}) and detection voltage (V_{PDR}), generates internal reset signal when $V_{DD} < V_{PDR}$.
  Note that, after power supply is turned off, this LSI should be placed in the STOP mode, or in the reset state by utilizing LVD0 or externally input reset signal, before the operating voltage falls below the range defined in **28.4 AC Characteristics**. When restarting the operation, make sure that the operating voltage has returned within the range of operation.

Caution **If an internal reset signal is generated by the power-on-reset circuit, the reset control flag register (RESF) and power-on-reset status register (PORSR) are cleared to 00H.**

Remark 1. This MCU incorporates multiple hardware functions that generate an internal reset signal. A flag that indicates the reset source is located in the reset control flag register (RESF) for when an internal reset signal is generated by the watchdog timer (WDT), voltage detector (LVD), illegal instruction execution, RAM parity error, or illegal-memory access. The RESF register is not cleared to 00H and the flag is set to 1 when an internal reset signal is generated by the watchdog timer (WDT), voltage detector (LVD), illegal instruction execution, RAM parity error, or illegal-memory access. For details of the RESF register, see **Section 17 Reset Function**.

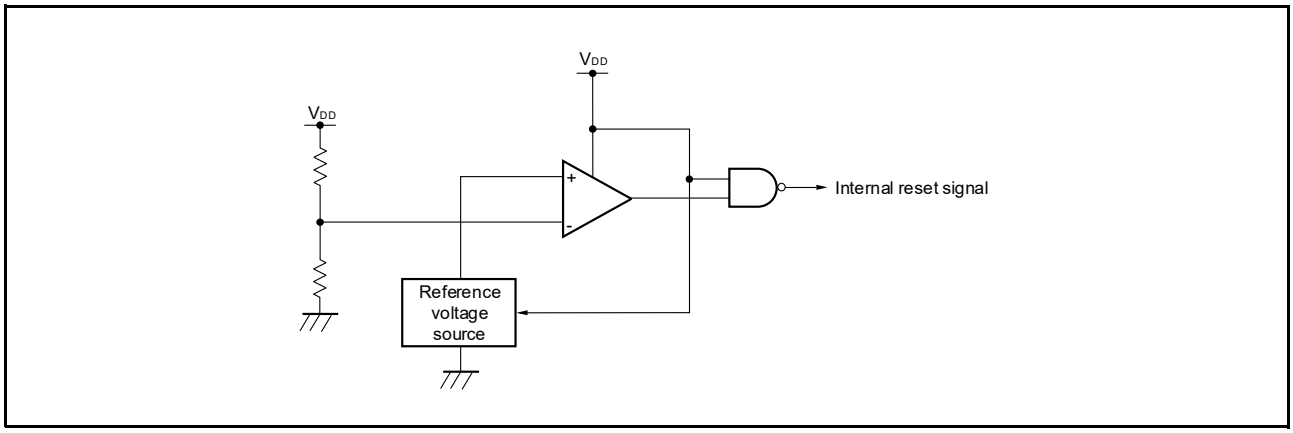
Remark 2. The power-on-reset status register (PORSR) is used to check the occurrence of an internal reset from the power-on-reset circuit. For details of the PORSR register, see **Section 17 Reset Function**.

Remark 3. V_{POR}: POR power supply rise detection voltage
V_{PDR}: POR power supply fall detection voltage
For details, see **28.6.4 Characteristics of the POR circuit**.

18.2 Configuration of Power-on-reset Circuit

Figure 18 - 1 shows the block diagram of the power-on-reset circuit.

Figure 18 - 1 Block Diagram of Power-on-reset Circuit

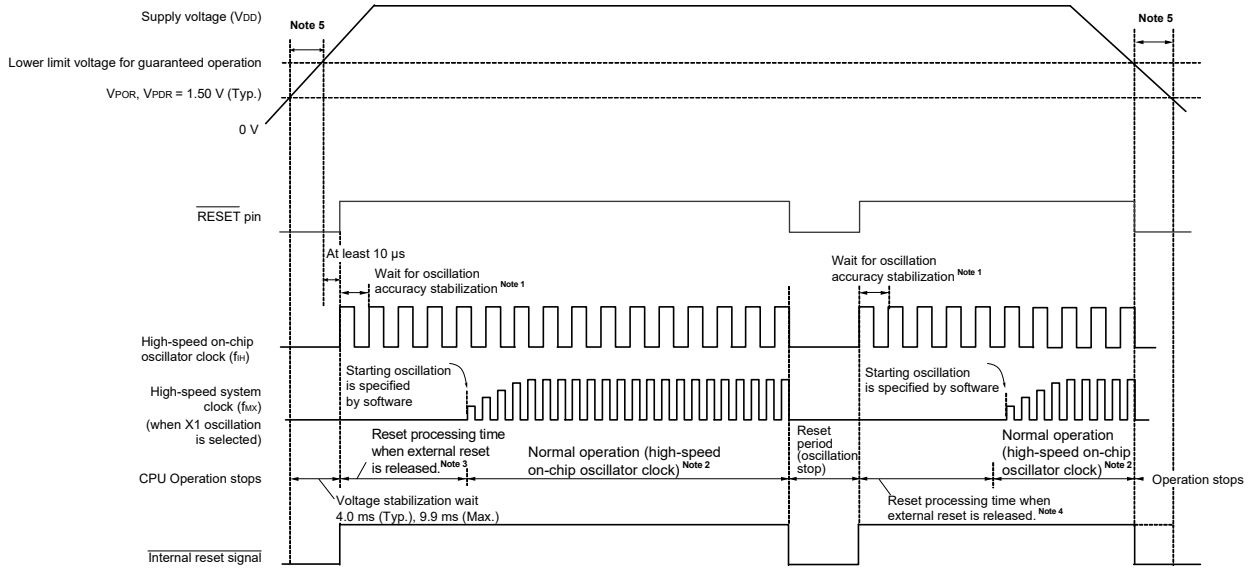


18.3 Operation of Power-on-reset Circuit

Figure 18 - 2 shows the timing of generation of the internal reset signal by the power-on-reset circuit and voltage detector.

Figure 18 - 2 Timing of Generation of Internal Reset Signal by Power-on-reset Circuit and Voltage Detector (1/2)

<1> When the externally input reset signal on the $\overline{\text{RESET}}$ pin is used



Note 1. After power is supplied, the reset state must be retained until the operating voltage becomes in the range defined in **28.4 AC Characteristics**. This is done by controlling the externally input reset signal. After power supply is turned off, this LSI should be placed in the STOP mode, or in the reset state by utilizing the voltage detection circuit or externally input reset signal, before the voltage falls below the range of operation. When restarting the operation, make sure that the operating voltage has returned within the range of operation.

Note 2. The internal reset processing time includes the oscillation accuracy stabilization time of the high-speed on-chip oscillator clock.

Note 3. The time until normal operation starts includes the following reset processing time when the external reset is released (release from the first external reset following release from the POR state) after the $\overline{\text{RESET}}$ signal is driven high as well as the voltage stabilization wait time after VPOR (1.50 V, typ.) is reached.
With the LVD circuit in use: 0.506 ms (typ.), 0.694 ms (max.)
With the LVD circuit not in use: 0.201 ms (typ.), 0.335 ms (max.)

Note 4. The CPU clock can be switched from the high-speed on-chip oscillator clock to the high-speed system clock or low-speed on-chip oscillator. To use the X1 clock, use the oscillation stabilization time counter status register (OSTC) to confirm the lapse of the oscillation stabilization time.

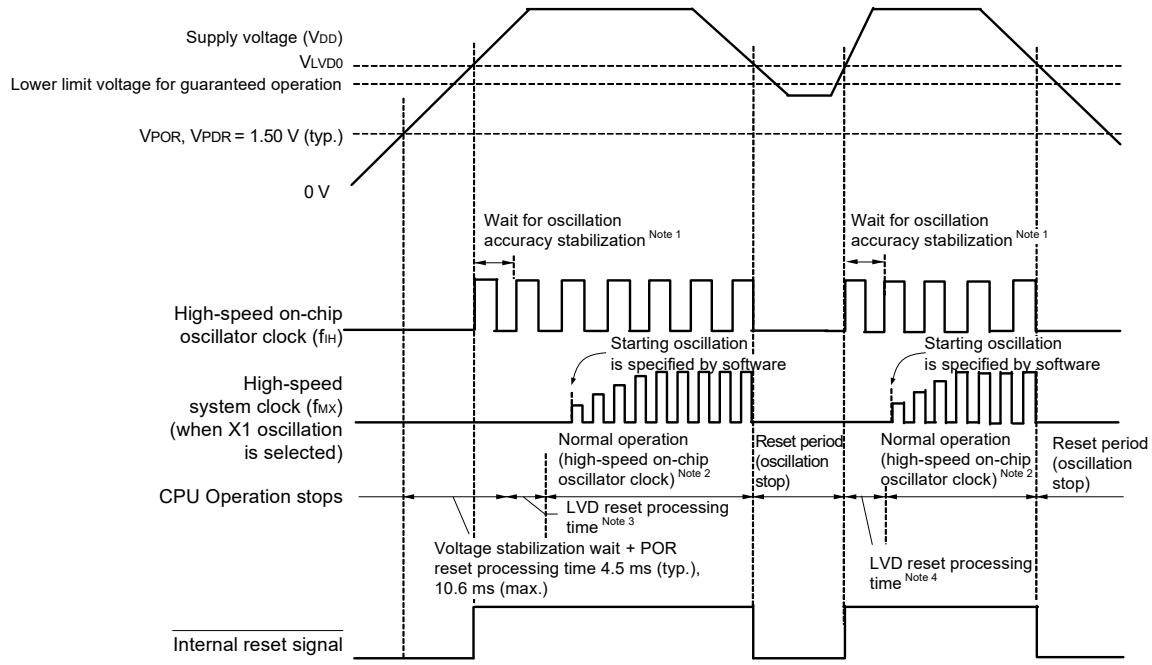
Note 5. The reset processing times in the case of the second or subsequent external reset following release from the POR state are listed below.
With the LVD circuit in use: 0.476 ms (typ.), 0.616 ms (max.)
With the LVD circuit not in use: 0.170 ms (typ.), 0.257 ms (max.)

Caution **For power-on reset, be sure to use the externally input reset signal on the $\overline{\text{RESET}}$ pin when LVD0 is off. For details, see Section 19 Voltage Detector (LVD).**

Remark VPOR: POR power supply rise detection voltage
VPDR: POR power supply fall detection voltage

Figure 18 - 2 Timing of Generation of Internal Reset Signal by Power-on-reset Circuit and Voltage Detector (2/2)

<2> LVD0 reset mode (LVD0SEL of the user option byte (000C1H) = 1)



Note 1. The internal reset processing time includes the oscillation accuracy stabilization time of the high-speed on-chip oscillator clock.

Note 2. The CPU clock can be switched from the high-speed on-chip oscillator clock to the high-speed system clock or low-speed on-chip oscillator. To use the X1 clock, use the oscillation stabilization time counter status register (OSTC) to confirm the lapse of the oscillation stabilization time.

<R> **Note 3.** The time until normal operation starts includes the following LVD reset processing time after the LVD0 detection level (V_{LVD0}) is reached as well as the voltage stabilization wait + POR reset processing time after the V_{POR} (1.50 V, typ.) is reached.
LVD reset processing time: 0 to 0.049 ms (max.)

<R> **Note 4.** When the power supply voltage is below the lower limit for operation and the power supply voltage is then restored after an internal reset is generated only by the voltage detector (LVD0), the following LVD reset processing time is required after the LVD0 detection level (V_{LVD0}) is reached.
LVD reset processing time: 0.045 ms (typ.), 0.049 ms (max.)

Remark 1. V_{LVD0}: LVD detection voltage
V_{POR}: POR power supply rise detection voltage
V_{PDR}: POR power supply fall detection voltage

Remark 2. When the LVD0 interrupt mode is selected (LVD0SEL of the user option byte (000C1H) = 0), the time until normal operation starts after power is turned on is the same as the time specified in **Note 3**.

Remark 3. Operation of LVD1 is stopped when power is initially supplied. LVD1 is also stopped by an internal reset.

Section 19 Voltage Detector (LVD)

19.1 Functions of Voltage Detector

Enabling, selecting the operation mode, and setting the detection voltage (V_{LVD0}) for voltage detector 0 (LVD0) are done by using a user option byte (000C1H). On the other hand, enabling, selecting the operation mode, and setting the detection voltage (V_{LVD1}) for voltage detector 1 (LVD1) are done by using the voltage detection level register. The voltage detectors have the following functions.

- LVD0 and LVD1 compare the supply voltage (V_{DD}) with the detection voltage (V_{LVD0}, V_{LVD1}), and generate an internal reset or internal interrupt signal.
- The user option byte is used to select the detection voltage (V_{LVD0}) for LVD0 from among 6 voltages (for details, see **Section 23 Option Bytes**).
- The voltage detection level register is used to select the detection voltage (V_{LVD1}) for LVD1 from among 18 voltages.
- Operable in STOP mode.
- After power is supplied, the reset state must be retained until the operating voltage becomes in the range defined in **28.4 AC Characteristics**. This is done by utilizing LVD0 or controlling the externally input reset signal. After the power supply is turned off, this LSI should be placed in the STOP mode, or placed in the reset state by utilizing LVD0 or controlling the externally input reset signal before the voltage falls below the operating range.

The internal reset and internal interrupt signals are generated in each mode as follows.

Reset Mode LVD0	Reset Mode LVD1	Interrupt Mode LVD0	Interrupt Mode LVD1
Deasserts an internal reset signal on detecting $V_{DD} \geq V_{LVD0}$. Generates an internal reset on detecting $V_{DD} < V_{LVD0}$ and retains the reset state until $V_{DD} \geq V_{LVD0}$ is detected.	Generates an internal reset on detecting $V_{DD} < V_{LVD1}$ after LVD1 operation has been enabled.	Retains the state of an internal reset by the LVD immediately after a reset until $V_{DD} \geq V_{LVD0}$. Releases the LVD internal reset by detecting $V_{DD} \geq V_{LVD0}$. Generates an interrupt request signal (INTLVI) by detecting $V_{DD} < V_{LVD0}$ or $V_{DD} \geq V_{LVD0}$ after the LVD internal reset is released.	Generates an interrupt request signal (INTLVI) on detecting $V_{DD} < V_{LVD1}$ after LVD1 operation has been enabled. After the first detection, generates an interrupt request signal (INTLVI) on detecting $V_{DD} < V_{LVD1}$ or $V_{DD} \geq V_{LVD1}$.

While LVD0 or LVD1 is operating, whether the supply voltage is no less than or less than the detection level can be checked by reading the voltage detection flag (LVDnF: bits 0 and 1 of the voltage detection register (LVIM)). The LVIRF bit of the reset control flag register (RESF) is set to 1 if reset occurs. For details of the RESF register, see **Section 17 Reset Function**.

19.2 Configuration of Voltage Detector

Figures 19 - 1 and **19 - 2** are block diagrams of the voltage detectors.

Figure 19 - 1 Block Diagram of LVD0

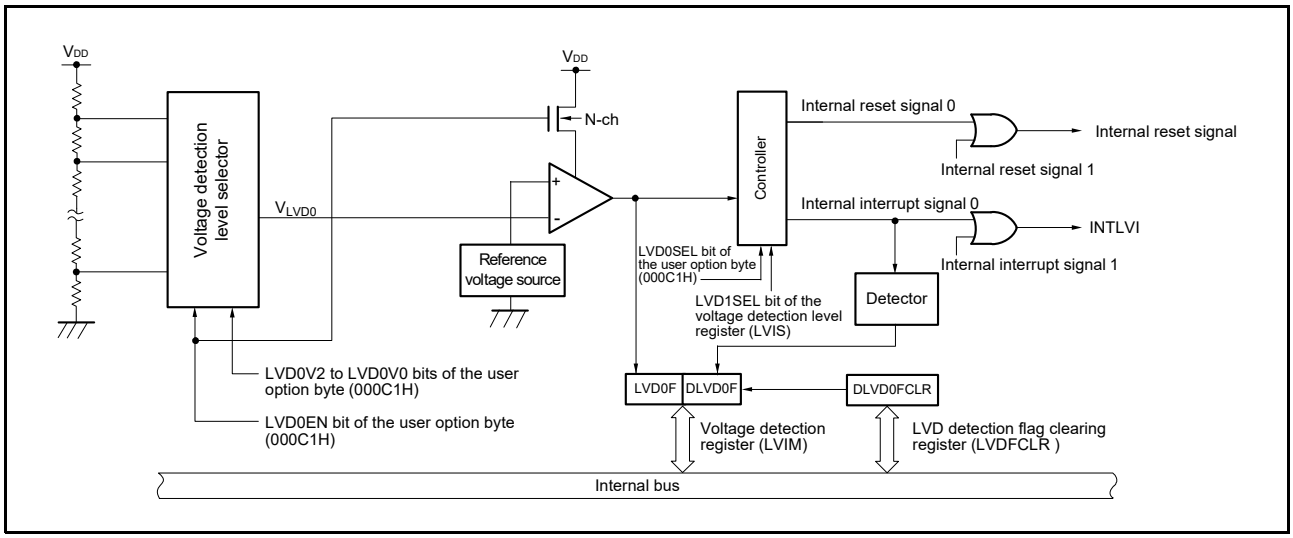
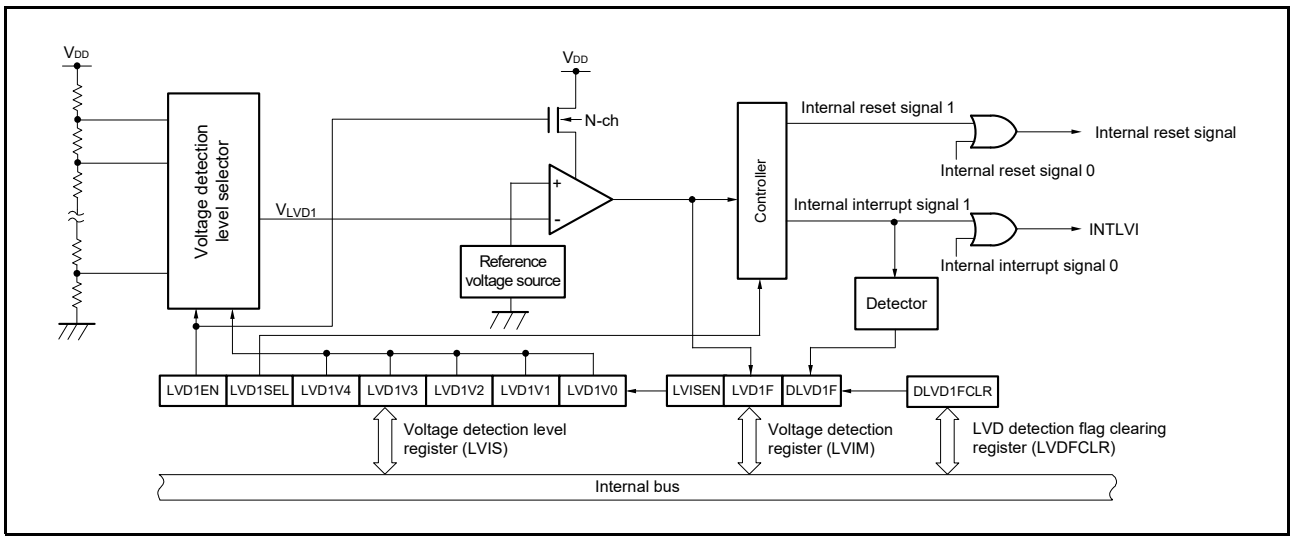


Figure 19 - 2 Block Diagram of LVD1



19.3 Registers for Controlling the Voltage Detector

The following registers are used to control the voltage detector.

- User option byte (000C1H/040C1H): See **Section 23 Option Bytes**.
- Voltage detection register (LVIM)
- LVD detection flag clearing register (LVDFCLR)
- Voltage detection level register (LVIS)

19.3.1 Voltage detection register (LVIM)

This register is used to specify whether to enable or disable rewriting the voltage detection level register (LVIS), as well as to check the states of LVD0 and LVD1. This register can be set by a 1-bit or 8-bit memory manipulation instruction.
The value of this register following a reset is 00H.

Figure 19 - 3 Format of Voltage Detection Register (LVIM)

Address: FFFA9H
After reset: 00H**Note 1**
R/W: R/W**Notes 2, 3**

Symbol	<7>	6	5	4	<3>	<2>	<1>	<0>
LVIM	LVISEN	0	0	0	DLVD1F	DLVD0F	LVD1F	LVD0F

LVISEN	Specification of whether to enable or disable rewriting the voltage detection level register (LVIS)
0	Rewriting of the LVIS register is disabled.
1**Note 4**	Rewriting of the LVIS register is enabled (reset and interrupt generation by LVD1 are masked).

DLVDnF	LVDn detection interrupt flag
0	The given LVDn interrupt has not been detected.
1	The given LVDn interrupt has been detected.

LVDnF	Voltage detection flag
0	Supply voltage (V_{DD}) ≥ detection voltage (V_{LVDn}), or when LVD is off
1	Supply voltage (V_{DD}) < detection voltage (V_{LVDn})

Note 1. The value after a reset is 01H when LVD0 operation is enabled and the power supply voltage (V_{DD}) is less than the detection voltage (V_{LVD0}).
Note 2. Bits 0 and 1 are read-only.
Note 3. Bits 2 and 3 are read-only. These bits are cleared by using the LVD detection flag clearing register (LVDFCLR).
Note 4. While the LVISEN bit is 1, the reset and interrupt generation by LVD1 are masked. Therefore, clear the LVISEN bit to 0 after having written a new value to the LVIS register.

Remark n = 0, 1

19.3.2 LVD detection flag clearing register (LVDFCLR)

This register is used to clear the interrupt detection flags (DLVD0F and DLVD1F) of the voltage detection register (LVIM). The LVDFCLR register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 19 - 4 Format of LVD Detection Flag Clearing Register (LVDFCLR)

Address: F0218H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	<3>	<2>	1	0
LVDFCLR	0	0	0	0	DLVD1FCLR	DLVD0FCLR	0	0

DLVD1FCLR **Note**	Clearing DLVD1F
0	No effect
1	Writing 1 to this bit clears the DLVD1F flag.

DLVD0FCLR **Note**	Clearing DLVD0F
0	No effect
1	Writing 1 to this bit clears the DLVD0F flag.

Note Only 1 can be written to this bit. Writing 0 has no effect. The bit is read as 0 even after 1 has been written to it.

19.3.3 Voltage detection level register (LVIS)

This register is used to select the voltage detection level for LVD1. This register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 19H.

Figure 19 - 5 Format of Voltage Detection Level Register (LVIS)

Address: FFFAAH
After reset: 19H
R/W: R/W

Symbol	<7>	<6>	5	4	3	2	1	0
LVIS	LVD1EN	LVD1SEL	0	LVD1V4	LVD1V3	LVD1V2	LVD1V1	LVD1V0

LVD1EN	Enabling operation of LVD1
0	Operation stopped
1	Operation enabled

LVD1SEL	Operation mode of LVD1
0	Interrupt mode
1	Reset mode

LVD1V4	LVD1V3	LVD1V2	LVD1V1	LVD1V0	Detection voltages for LVD1 Notes 1, 2, 3		
						Rising edge	Falling edge
1	1	1	1	1	V_{LVD117}	1.67 V Note 4	1.63 V Note 4
1	1	1	1	0	V_{LVD116}	1.78 V Note 4	1.74 V Note 4
1	1	1	0	1	V_{LVD115}	1.88 V Note 4	1.84 V Note 4
1	1	1	0	0	V_{LVD114}	1.98 V	1.94 V
1	1	0	1	1	V_{LVD113}	2.09 V	2.04 V
1	1	0	1	0	V_{LVD112}	2.20 V	2.15 V
1	1	0	0	1	V_{LVD111}	2.30 V	2.25 V
1	1	0	0	0	V_{LVD110}	2.40 V	2.35 V
1	0	1	1	1	V_{LVD19}	2.50 V	2.45 V
1	0	1	1	0	V_{LVD18}	2.66 V	2.60 V
1	0	1	0	1	V_{LVD17}	2.82 V	2.76 V
1	0	1	0	0	V_{LVD16}	2.97 V	2.91 V
1	0	0	1	1	V_{LVD15}	3.13 V	3.06 V
1	0	0	1	0	V_{LVD14}	3.35 V	3.27 V
1	0	0	0	1	V_{LVD13}	3.55 V	3.47 V
1	0	0	0	0	V_{LVD12}	3.75 V	3.67 V
0	1	1	1	1	V_{LVD11}	3.96 V	3.88 V
0	1	1	1	0	V_{LVD10}	4.16 V	4.08 V

Note 1. The LVD1V4 to LVD1V0 bits can only be rewritten once after release from the reset state.
Note 2. When setting LVD0 to reset mode, set the detection voltage of LVD1 higher than the detection voltage of LVD0.

(Note and Caution continue on the next page.)

Note 3. If LVD0 is set to interrupt mode and the LVD0 detection voltage is greater than the LVD1 detection voltage, LVD0 becomes undefined after the LVD1 setting following release from the reset state.

Note 4. This setting can only be used when LVD0 is off.

Caution When the values in the LVIS register are to be changed, do so according to the procedure described in Figures 19 - 10 and 19 - 11.

19.4 Operation of Voltage Detector

19.4.1 Setting for use in the reset mode

Enabling, selecting the operation mode (reset mode: LVD0SEL = 1), and setting the detection voltage (V_{LVD0}) for LVD0 are done by using a user option byte (000C1H). On the other hand, enabling, selecting the operation mode (reset mode: LVD1SEL = 1), and setting the detection voltage (V_{LVD1}) for LVD1 are done by using the voltage detection level register (LVIS).

- Operation in LVD reset mode
  When LVD0 is set for the reset mode (the value of the LVD0SEL bit in the user option byte is 1), the state of the internal reset being applied by LVD0 is retained until the power supply voltage (V_{DD}) exceeds the rising voltage detection level (V_{LVD0}) after power has been supplied. The internal reset is released when the supply voltage (V_{DD}) exceeds the rising voltage detection level (V_{LVD0}). At the fall of the operating voltage, an internal reset by LVD0 is generated when the power supply voltage (V_{DD}) falls below the falling voltage detection level (V_{LVD0}). Operation of LVD1 is stopped when power is initially supplied. When LVD1 operation is enabled, it generates an internal reset when the power supply voltage (V_{DD}) falls below the voltage detection level (V_{LVD1}). If operation of LVD1 is enabled while the power supply voltage (V_{DD}) is lower than the voltage detection level (V_{LVD1}), it generates an internal reset at the time its operation is enabled. If LVD1 is set for reset mode, LVD0 is placed in interrupt mode. In addition, the generation of an internal reset by LVD1 places LVD0 in reset mode. LVD1 detection voltage can only be set once after release from the reset state.

Figure 19 - 6 and **Figure 19 - 7** show the timing of the internal reset signal generated in LVD0 and LVD1, respectively.

Figure 19 - 6 Timing of the Internal Reset Signal Generated in LVD0

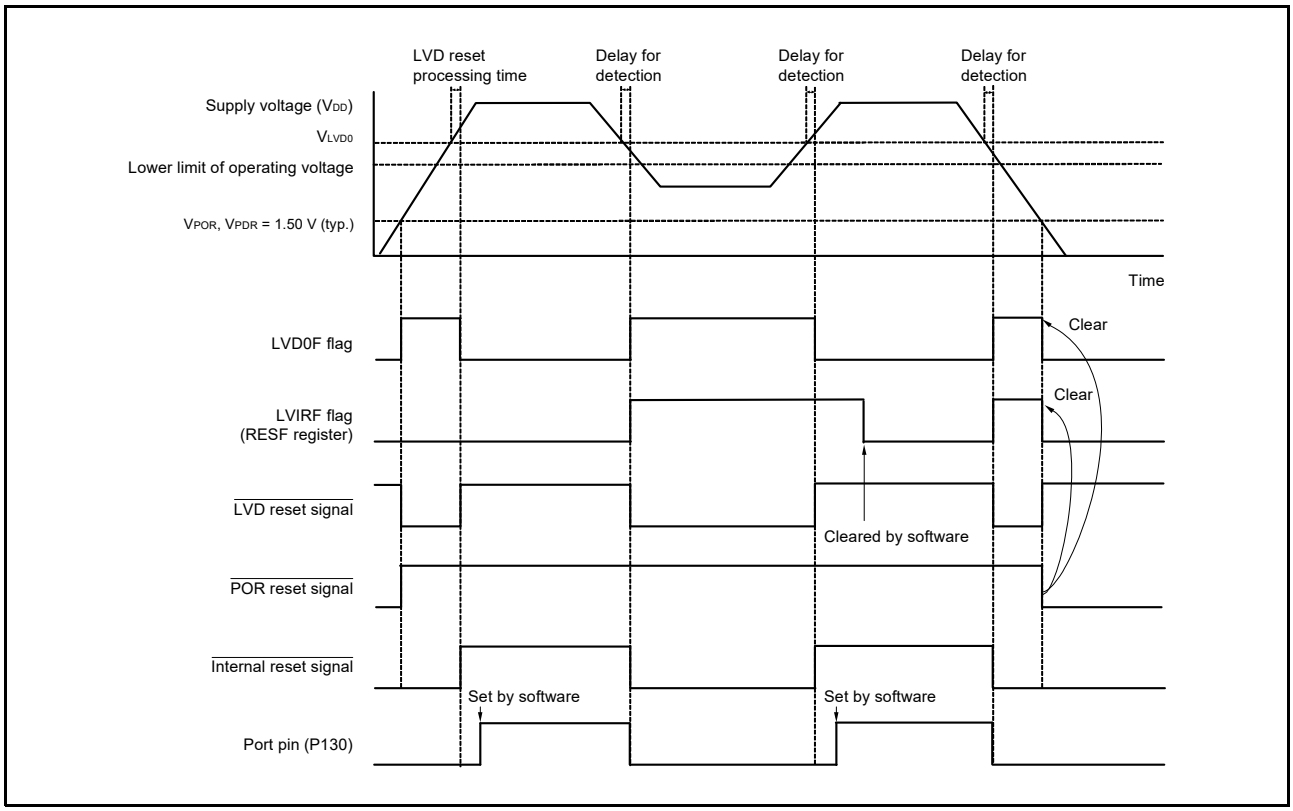
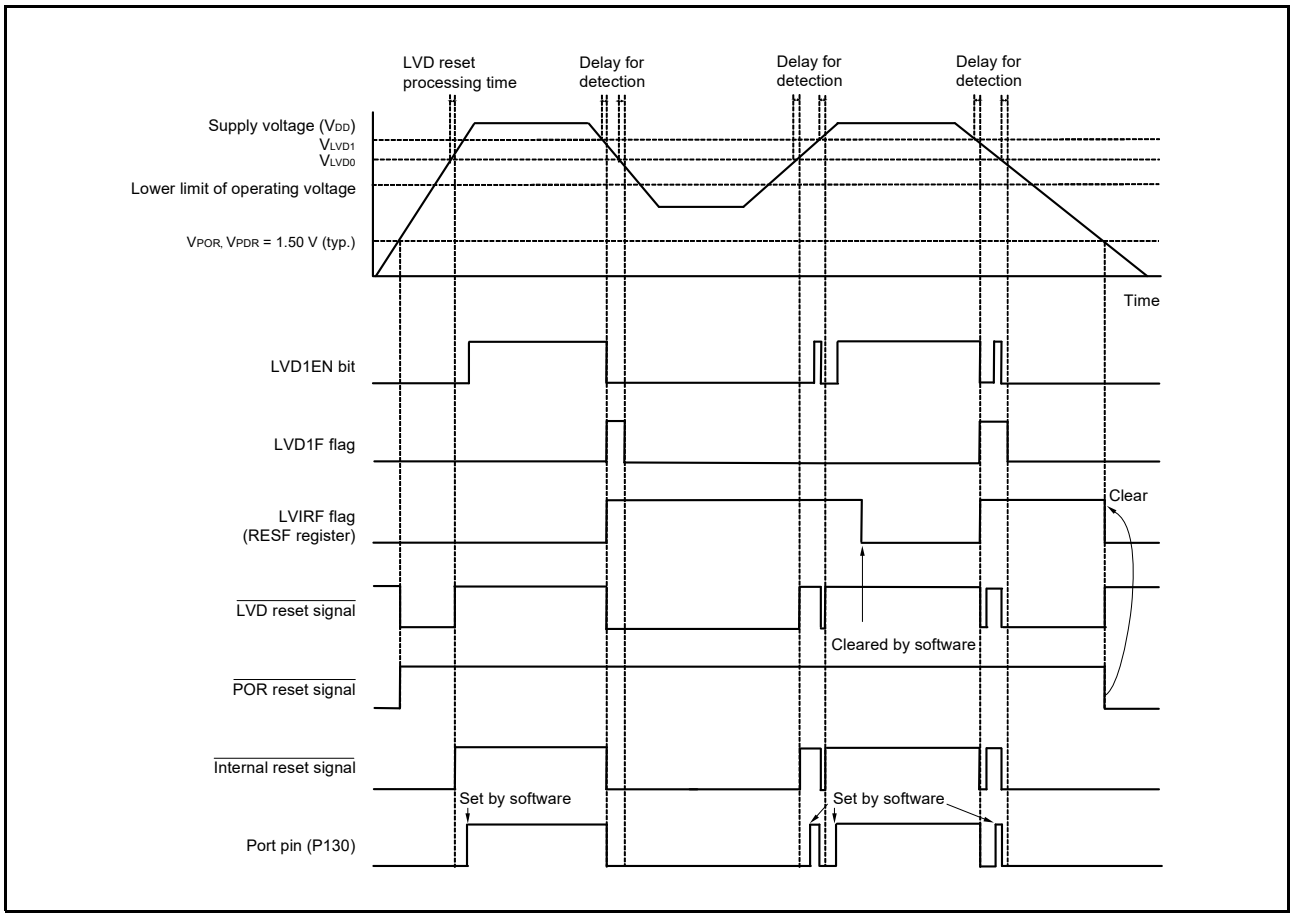


Figure 19 - 7 Timing of the Internal Reset Signal Generated in LVD1



Remark LVD0: Reset mode

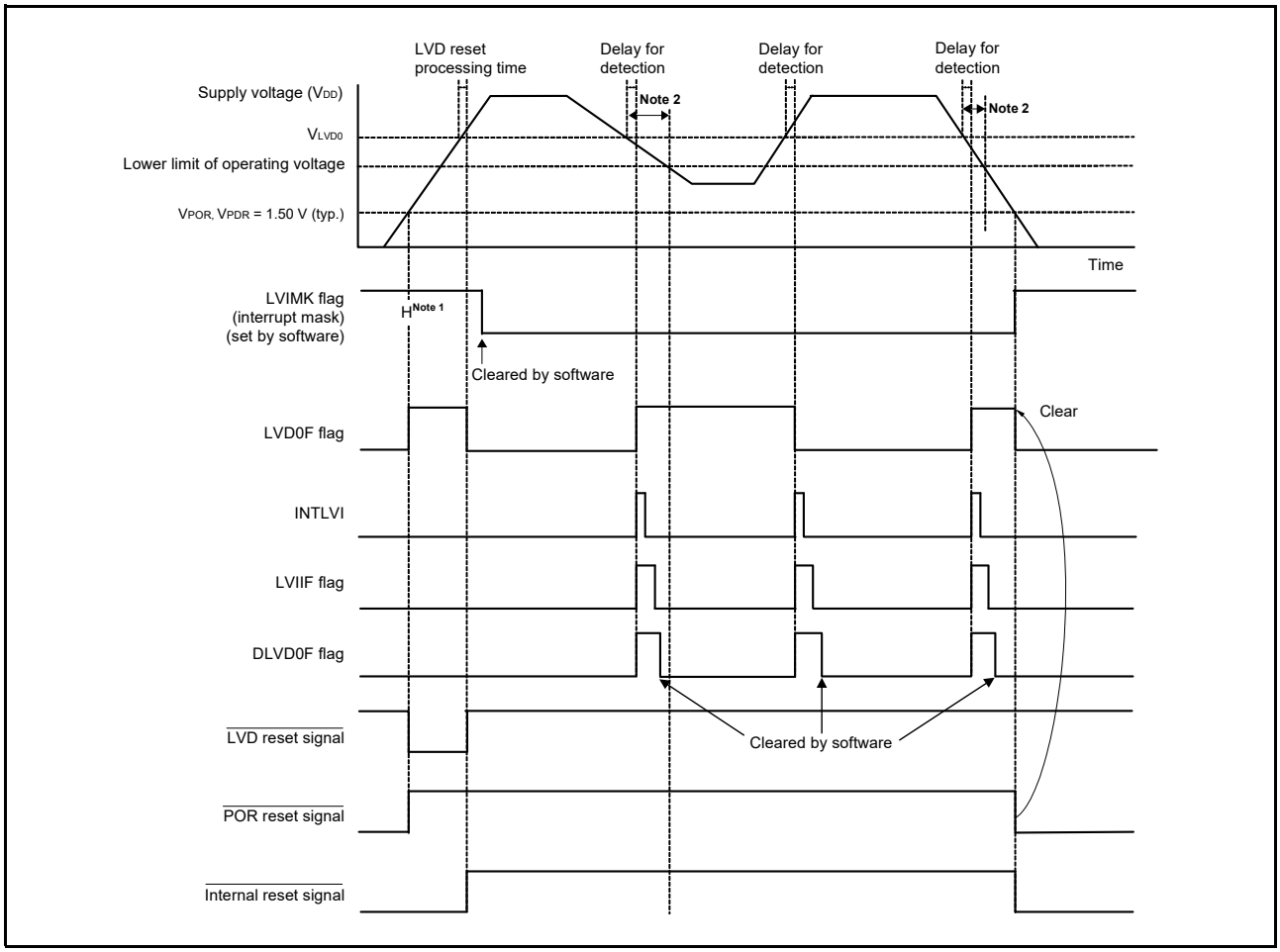
19.4.2 Setting for use in the interrupt mode

Enabling, selecting the operation mode (interrupt mode: LVD0SEL = 0), and setting the detection voltage (V_{LVD0}) for LVD0 are done by using a user option byte (000C1H). On the other hand, enabling, selecting the operation mode (interrupt mode: LVD1SEL = 0), and setting the detection voltage (V_{LVD1}) for LVD1 are done by using the voltage detection level register (LVIS).

- Operation in LVD interrupt mode
  When LVD0 is set for the interrupt mode (the value of the LVD0SEL bit in the option byte is 0), the state of the internal reset being applied by LVD0 is retained until the power supply voltage (V_{DD}) exceeds the rising voltage detection level (V_{LVD0}) immediately after a reset has been generated. The internal reset is released when the power supply voltage (V_{DD}) exceeds the rising voltage detection level (V_{LVD0}). After the internal reset signal has been deasserted, LVD0 generates an interrupt request signal (INTLVI) if the power supply voltage (V_{DD}) falls below the voltage detection level (V_{LVD0}). Similarly, when the power supply voltage (V_{DD}) rises above the voltage detection level (V_{LVD0}), LVD0 also generates an interrupt request signal (INTLVI). When the voltage falls, this LSI should be placed in the STOP mode, or placed in the reset state by controlling the externally input reset signal, before the voltage falls below the operating voltage range defined in **28.4 AC Characteristics**. When restarting the operation, make sure that the power supply voltage has returned within the operating voltage range. Operation of LVD1 is stopped when power is initially supplied. When LVD1 operation is enabled, it generates an interrupt request signal (INTLVI) when the power supply voltage (V_{DD}) falls below the voltage detection level (V_{LVD1}). Similarly, when the power supply voltage (V_{DD}) rises above the voltage detection level (V_{LVD1}), LVD1 also generates an interrupt request signal (INTLVI). Note that if operation of LVD1 is enabled while the power supply voltage (V_{DD}) is lower than the voltage detection level (V_{LVD1}), it generates an interrupt request signal (INTLVI) at the time its operation is enabled. LVD1 detection voltage can only be set once after release from the reset state.

Figure 19 - 8 and **Figure 19 - 9** show the timing of the interrupt request signal generated in LVD0 and LVD1, respectively.

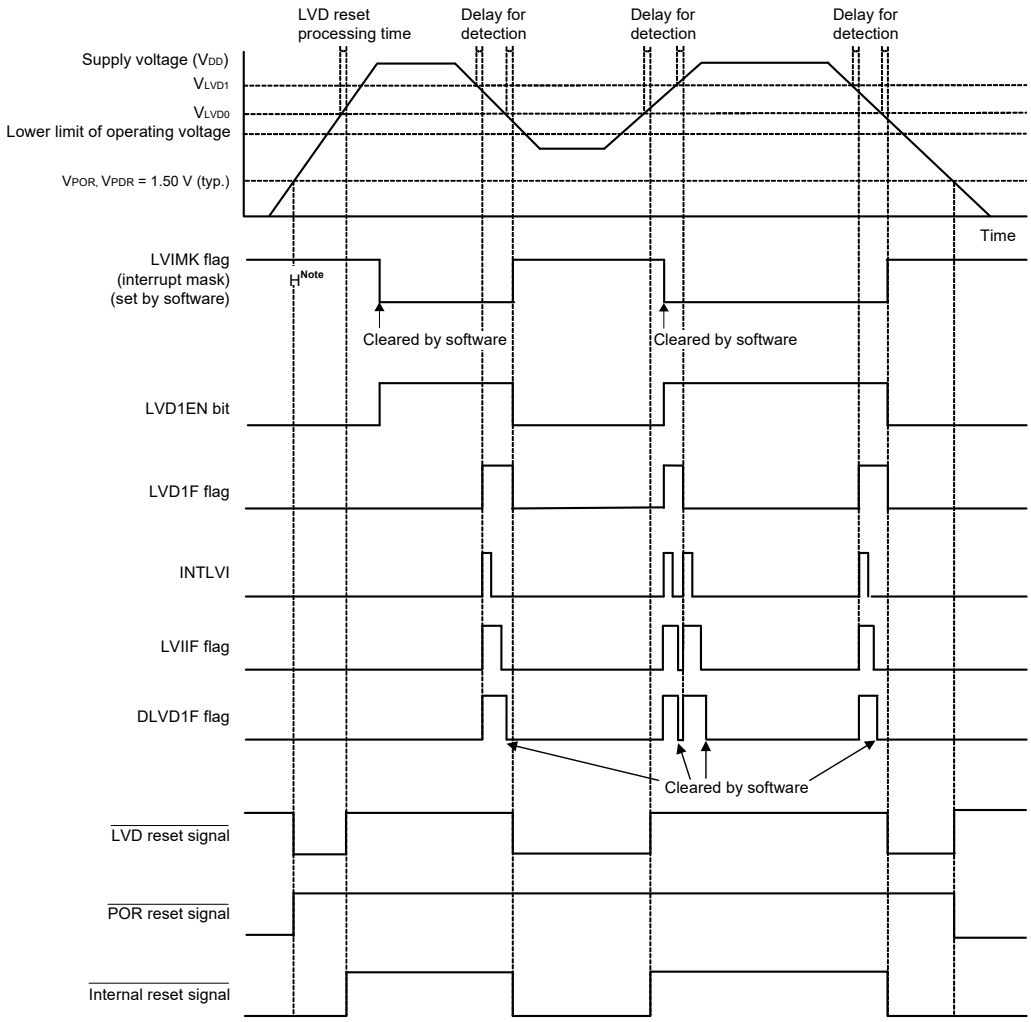
Figure 19 - 8 Timing of the Interrupt Request Signal Generated in LVD0



Note 1. The LVIMK flag is set to 1 by reset signal generation.

Note 2. When the voltage falls, this LSI should be placed in the STOP mode, or placed in the reset state by controlling the externally input reset signal, before the voltage falls below the operating voltage range defined in **28.4 AC Characteristics**. When restarting the operation, make sure that the operating voltage has returned within the operating voltage range.

Figure 19 - 9 Timing of the Interrupt Request Signal Generated in LVD1



Note The LVIMK flag is set to 1 by reset signal generation.

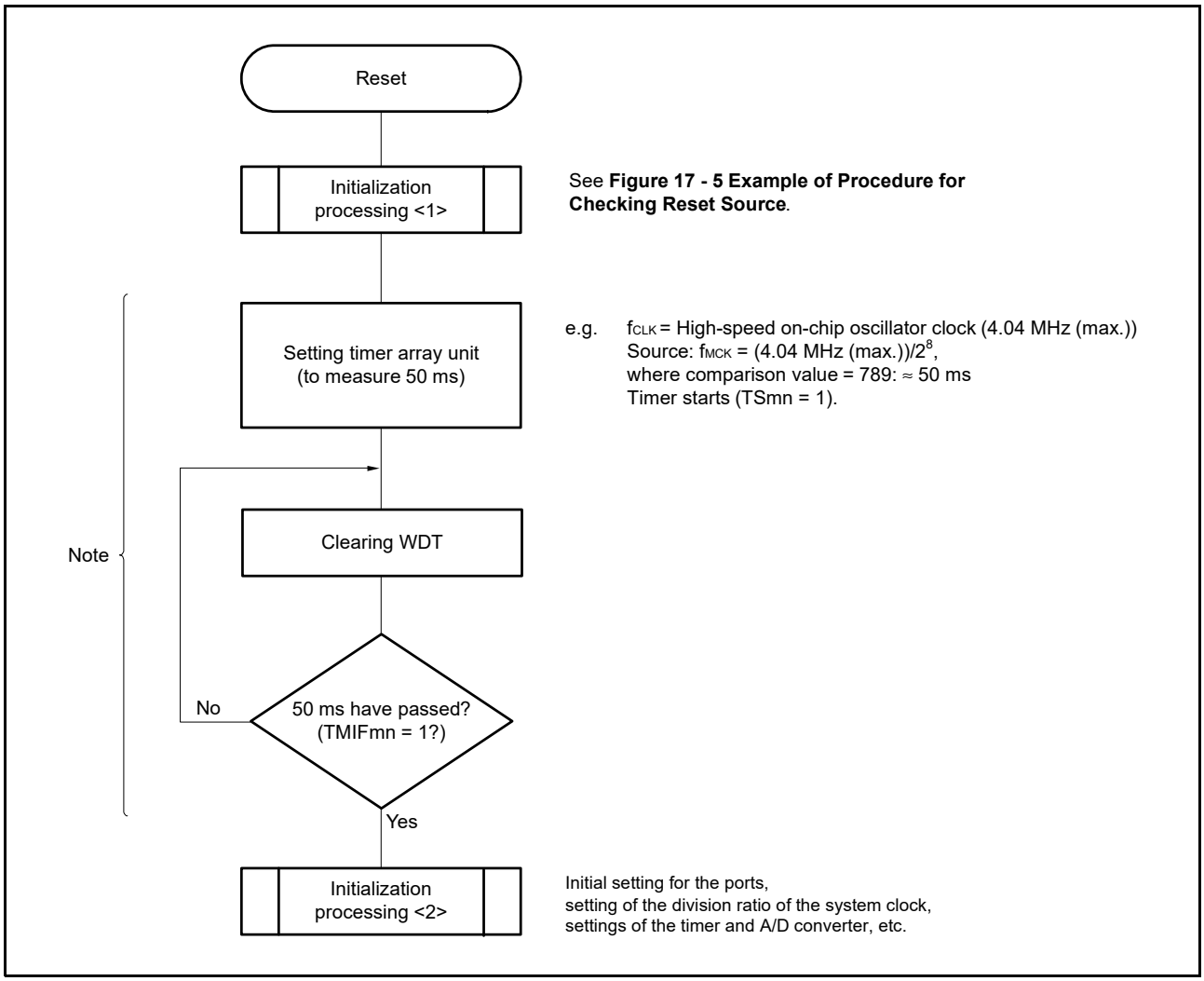
Remark LVD0: Reset mode

19.5 Points for Caution when the Voltage Detector Is to be Used

1. Voltage fluctuation when power is supplied
   In a system where the supply voltage (V_{DD}) fluctuates for a certain period in the vicinity of the detection voltages for LVD0 or LVD1, the system may be repeatedly reset and released from the reset state. In this case, the time from release of reset to the start of the operation of the microcontroller can be arbitrarily set by taking the following action.

   <Action>
   After releasing the reset signal, wait for the supply voltage fluctuation period of each system by means of a software counter that uses a timer, and then initialize the ports.

Figure 19 - 10 Example of Software Processing If Supply Voltage Fluctuation Is 50 ms or Less in Vicinity of the Detection Voltages for LVD0 or LVD1

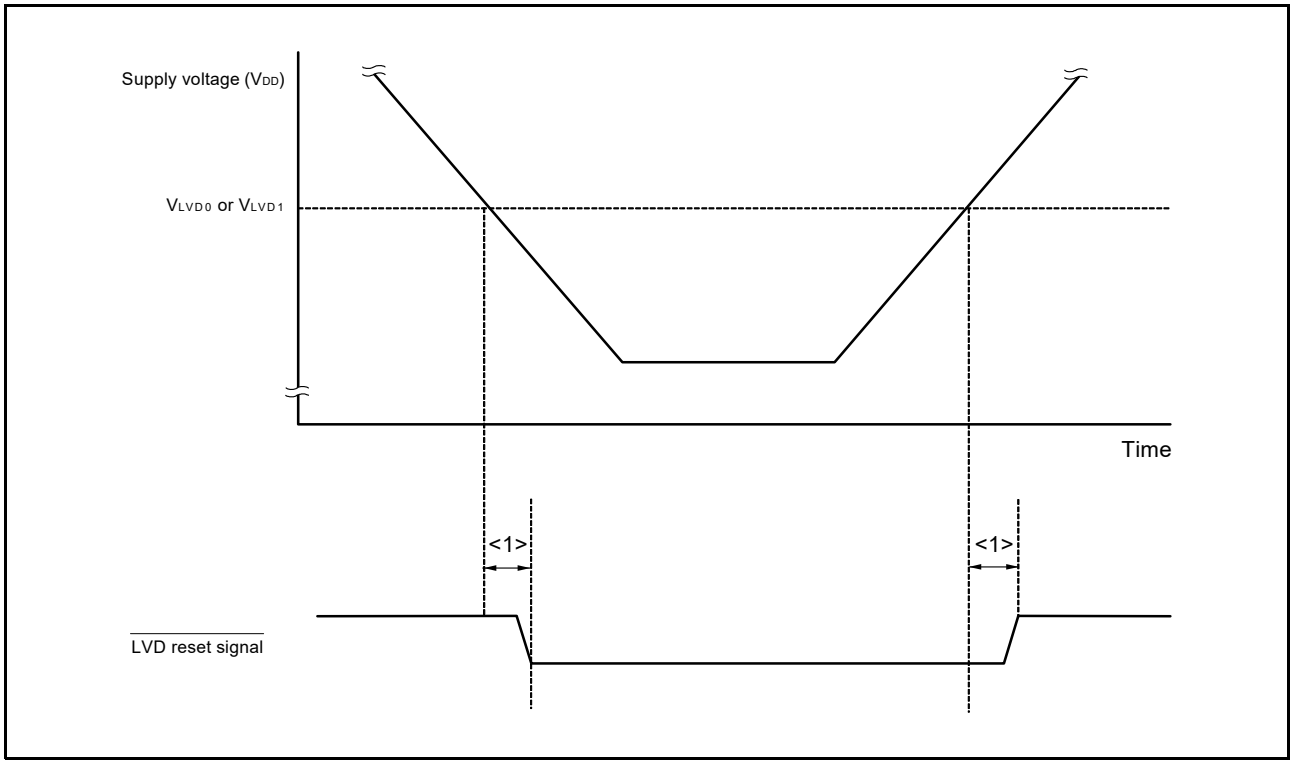


Note If reset is generated again during this period, initialization processing <2> is not started.

Remark m = 0
n = 0 to 7

2. Delays from the time an LVD0 or LVD1 reset source condition is satisfied until an LVD0 or LVD1 reset has been generated and deasserted
   The delay is from the time the power supply voltage (V_{DD}) becomes less than the LVD0 or LVD1 falling detection voltage (V_{LVD0} or V_{LVD1}) until the LVD0 or LVD1 reset is generated. In the same way, the delay is from the time the LVD0 or LVD1 rising detection voltage (V_{LVD0} or V_{LVD1}) becomes no greater than the power supply voltage (V_{DD}) until the LVD0 or LVD1 reset is deasserted. See **Figure 19 - 11**.

Figure 19 - 11 Delays from the Time an LVD0 or LVD1 Reset Source Condition is Satisfied until an LVD0 or LVD1 Reset has been Generated and Deasserted



Remark <1>: Delay for detection (500 μs (max.))

3. Turning power on when LVD0 is off
   Use the external rest input via the $\overline{\text{RESET}}$ pin when LVD0 is off.
   For an external reset, input a low level for 10 μs or more to the $\overline{\text{RESET}}$ pin. To perform an external reset upon power application, input a low level to the $\overline{\text{RESET}}$ pin, turn power on, continue to input a low level to the pin for 10 μs or more within the operating voltage range shown in **28.4 AC Characteristics**, and then input a high level to the pin.

4. Operating voltage fall when LVD0 is off or the interrupt mode is selected
   When the operating voltage falls with LVD0 off or with the interrupt mode selected, this LSI should be placed in the STOP mode, or placed in the reset state by controlling the externally input reset signal before the voltage falls below the operating voltage range defined in **28.4 AC Characteristics**. When restarting the operation, make sure that the operating voltage has returned within the operating voltage range.

5. Procedure for setting the LVD1 detection voltage
   Follow the procedure below to set the LVD1 detection voltage. After step <3>, LVD1 is enabled after the stabilization waiting time (at least 500 μs) has elapsed.
   <1> Set the LVISEN bit of the LVIM register to 1.
   <2> Set the LVD1EN bit of the LVIS register to 1 and change the setting of the LVD1V4 to LVD1V0 bits.
   <3> Set the LVISEN bit of the LVIM register to 0.

Section 20 Safety Functions

20.1 Overview of Safety Functions

Products of this group provide the following safety functions to comply with the IEC60730 safety standard. These safety functions enable the microcontroller to self-diagnose abnormalities and safely stop operating if an abnormality is detected.

1. Flash memory CRC operation (high-speed CRC, general-purpose CRC)
   This detects data errors in the flash memory by performing CRC operations. The following two CRC functions can be used depending on the application or purpose of use.

   - High-speed CRC: The CPU can be stopped and a high-speed check executed on its entire code flash memory area during the initialization routine.
   - General-purpose CRC: This can be used for checking various data in addition to the code flash memory area while the CPU is running.

2. Flash memory guard function
   This prevents rewriting of data in the flash memory due to incorrect CPU operations.
3. RAM parity error detection
   This detects parity errors when reading RAM data.
4. RAM guard function
   This prevents rewriting of data in RAM due to incorrect CPU operations.
5. SFR guard function
   This prevents rewriting of data in the SFRs due to incorrect CPU operations.
6. Illicit memory access detection
   This detects illicit accesses to invalid memory areas (such as areas where no memory is allocated and areas to which access is restricted).
7. Guard function of invalid memory access detection control register (IAWCTL)
   This prevents rewriting of the invalid memory access detection control register due to incorrect CPU operations.
8. Frequency detection
   This function allows a self-check of the CPU/peripheral hardware clock frequencies using the timer array unit.
9. Testing of the A/D converter
   This test checks whether or not the A/D converter is operating normally by converting the A/D converter's positive and negative reference voltages, analog input channels (ANIxx), temperature sensor output voltage, and internal reference voltage.
10. Detection of the digital output signal level of the I/O pins
    This is used to read the output level of an I/O pin when the pin is in the output mode.
11. UART loopback
    This is used to confirm that the transmit data is output normally by shutting off the TxDn and RxDn pins of UARTn and the TxDAm and RxDAm pins of UARTAm from the outside and connecting them within the MCU to loop back the output from the transmission shift register to the reception shift register.

Remark 1. m = 0, n = 0 to 2

Remark 2. For usage examples of the safety functions complying with the IEC60730 safety standard, refer to the **application notes IEC60730/60335 Self Test Library for RL78 MCU**.

20.2 Registers for Controlling the Safety Functions

The following registers are used to control the safety functions.

- Flash memory CRC control register (CRC0CTL)
- Flash memory CRC operation result register (PGCRCL)
- CRC input register (CRCIN)
- CRC data register (CRCD)
- Code flash memory guard register (GFLASH0)
- Data flash memory guard register (GFLASH1)
- Flash security area guard register (GFLASH2)
- RAM parity error control register (RPECTL)
- Invalid memory access detection control register (IAWCTL)
  - RAM guard function
  - SFR guard function
  - Illicit memory access detection
- Guard register of IAWCTL register (GIAWCTL)
- Timer input select register 0 (TIS0)
- A/D test register (ADTES)
- Analog input channel specification register (ADS)
- Port mode select register (PMS)
- UART loopback select register (ULBS)

The content of each register is described in **20.3 Operation of Safety Functions**.

20.3 Operation of Safety Functions

20.3.1 Flash memory CRC operation (high-speed CRC)

The IEC60730 standard mandates the checking of data in the flash memory, and recommends using a CRC to do it. The high-speed CRC provided in products of this group can be used to check the entire code flash memory area during the initialization routine. The high-speed CRC can be executed only when the program is allocated on the RAM and in the HALT mode of the main system clock. The high-speed CRC performs an operation by reading 32-bit data per clock from the flash memory while stopping the CPU. This function therefore can finish a check in a shorter time (for example, 512 µs@32 MHz with 64-Kbyte flash memory). The CRC generator polynomial used complies with "$X^{16} + X^{12} + X^5 + 1$" of CRC-16-CCITT. The high-speed CRC operates in MSB first order from bit 31 to bit 0.

Caution **The CRC operation result might differ during on-chip debugging because the monitor program is allocated.**

Remark The operation result is different between the high-speed CRC and the general-purpose CRC, because the general-purpose CRC operates in LSB first order.

20.3.1.1 Flash memory CRC control register (CRC0CTL)

This register is used to control the operation of the high-speed CRC ALU, as well as to specify the operation range. The CRC0CTL register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 20 - 1 Format of Flash Memory CRC Control Register (CRC0CTL)

Address: F02F0H
After reset: 00H
R/W: R/W

Symbol	<7>	6	5	4	3	2	1	0
CRC0CTL	CRC0EN	0	FEA5	FEA4	FEA3	FEA2	FEA1	FEA0

CRC0EN	Control of CRC ALU operation
0	Stops the operation.
1	Starts the operation according to HALT instruction execution.

FEA5	FEA4	FEA3	FEA2	FEA1	FEA0	High-speed CRC operation range
0	0	0	0	0	0	0000H to 03FFBH (16 Kbytes - 4 bytes)
0	0	0	0	0	1	0000H to 07FFBH (32 Kbytes - 4 bytes)
0	0	0	0	1	0	0000H to 0BFFBH (48 Kbytes - 4 bytes)
0	0	0	0	1	1	0000H to 0FFFBH (64 Kbytes - 4 bytes)
0	0	0	1	0	0	0000H to 13FFBH (80 Kbytes - 4 bytes)
0	0	0	1	0	1	0000H to 17FFBH (96 Kbytes - 4 bytes)
0	0	0	1	1	0	0000H to 1BFFBH (112 Kbytes - 4 bytes)
0	0	0	1	1	1	0000H to 1FFFBH (128 Kbytes - 4 bytes)
0	0	1	0	0	0	0000H to 23FFBH (144 Kbytes - 4 bytes)
0	0	1	0	0	1	0000H to 27FFBH (160 Kbytes - 4 bytes)
0	0	1	0	1	0	0000H to 2BFFBH (176 Kbytes - 4 bytes)
0	0	1	0	1	1	0000H to 2FFFBH (192 Kbytes - 4 bytes)
0	0	1	1	0	0	0000H to 33FFBH (208 Kbytes - 4 bytes)
0	0	1	1	0	1	0000H to 37FFBH (224 Kbytes - 4 bytes)
0	0	1	1	1	0	0000H to 3BFFBH (240 Kbytes - 4 bytes)
0	0	1	1	1	1	0000H to 3FFFBH (256 Kbytes - 4 bytes)
Other than the above						Setting prohibited

Caution **Be sure to clear bit 6 to 0.**

Remark Input the expected CRC operation result value to be used for comparison in the lowest 4 bytes of the flash memory. Note that the operation range will thereby be reduced by 4 bytes.

20.3.1.2 Flash memory CRC operation result register (PGCRCL)

This register holds the high-speed CRC operation results. The PGCRCL register can be set by a 16-bit memory manipulation instruction. The value of this register following a reset is 0000H.

Figure 20 - 2 Format of Flash Memory CRC Operation Result Register (PGCRCL)

Address: F02F2H, F02F3H
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
PGCRCL	PGCRC15	PGCRC14	PGCRC13	PGCRC12	PGCRC11	PGCRC10	PGCRC9	PGCRC8

7	6	5	4	3	2	1	0
PGCRC7	PGCRC6	PGCRC5	PGCRC4	PGCRC3	PGCRC2	PGCRC1	PGCRC0

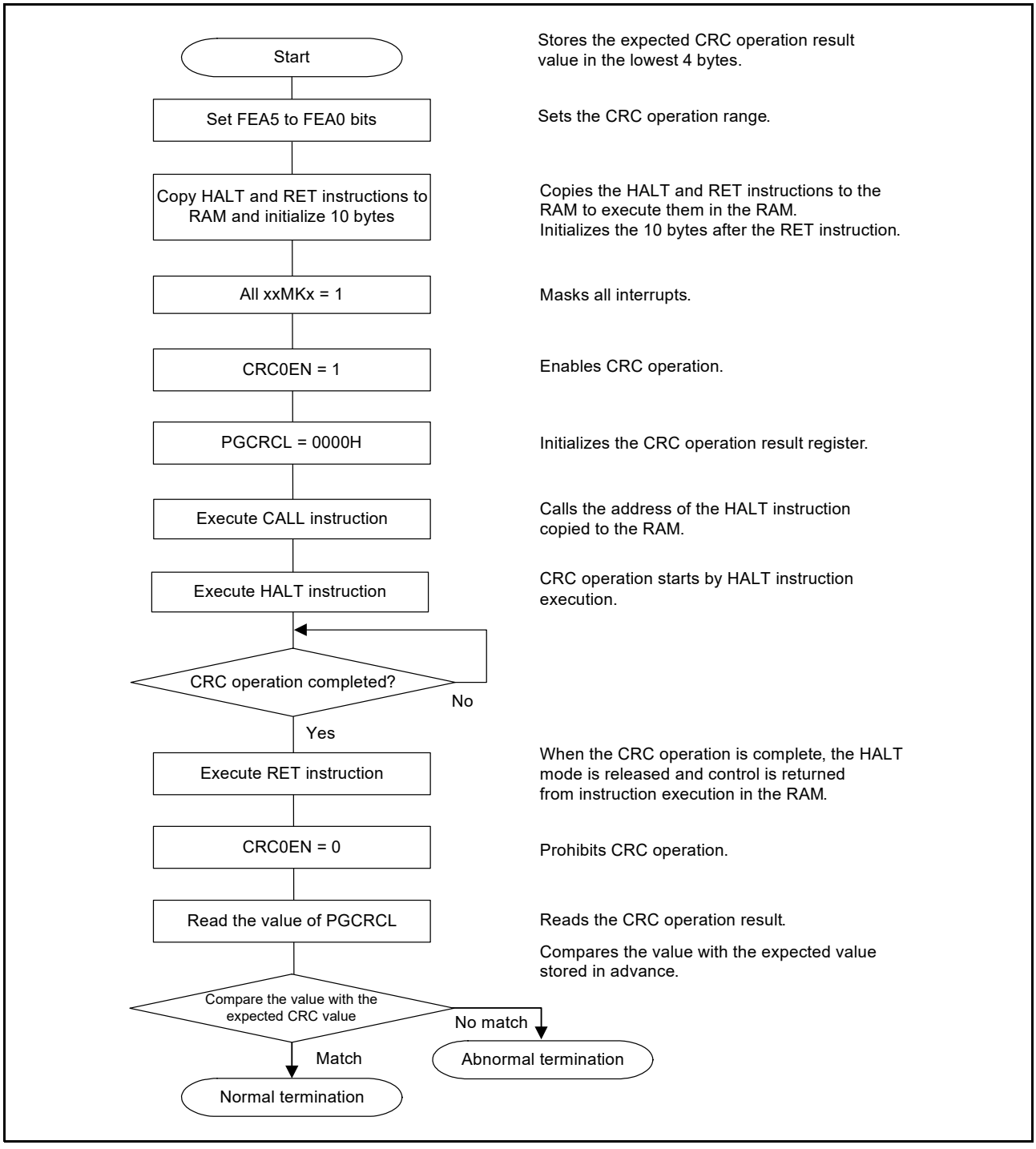
PGCRC15 to PGCRC0	High-speed CRC operation results
0000H to FFFFH	Holds the high-speed CRC operation results.

Caution This register can only be written if the CRC0EN bit of the CRC0CTL register is 1.

Figure 20 - 3 shows the flowchart of flash memory CRC operation (high-speed CRC).

<Operation flow>

Figure 20 - 3 Flowchart of Flash Memory CRC Operation (High-speed CRC)



Caution 1. The CRC operation is executed only on the code flash memory.

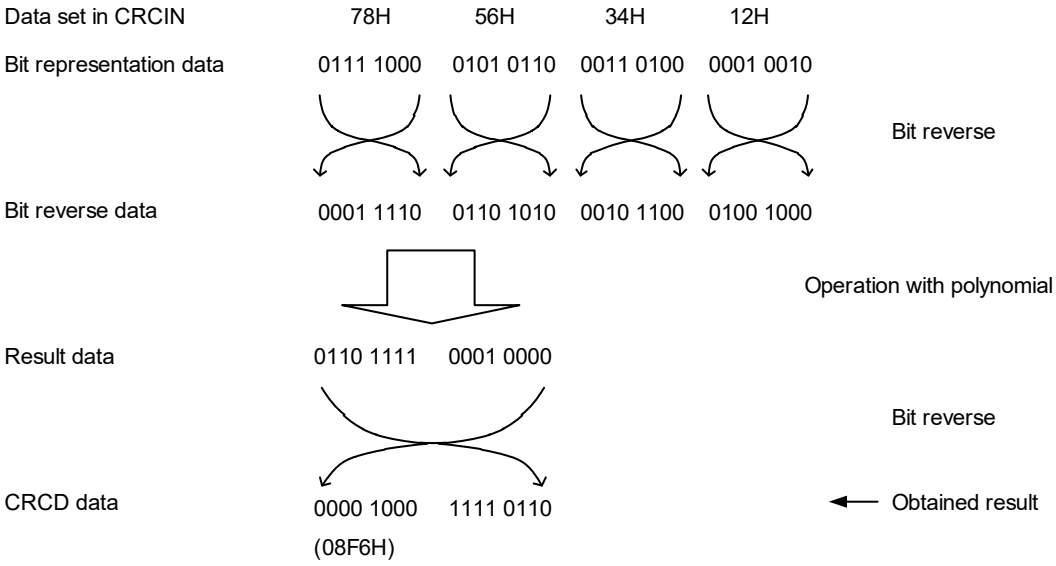
Caution 2. Store the expected CRC operation value in the area below the operation range in the code flash memory.

Caution 3. The CRC operation is enabled by executing the HALT instruction in the RAM area. Be sure to execute the HALT instruction in the RAM area.

The expected CRC value can be calculated by using the CS+ integrated development environment. See the CS+ Integrated Development Environment user's manual for details.

20.3.2 CRC operation (general-purpose CRC)

The general-purpose CRC handles CRC operation as a peripheral module while the CPU is operating. The general-purpose CRC can be used for checking various data in addition to the code flash memory area. The data to be checked can be specified by using software (a user-created program). The CRC calculation cannot be used in the HALT mode. The general-purpose CRC can operate either in the main system clock operation mode or in the subsystem clock operation mode. The CRC generator polynomial used is "$X^{16} + X^{12} + X^5 + 1$" of CRC-16-CCITT. The data to be input is inverted in bit order and then calculated to allow for LSB-first communication. For example, if the data 12345678H is sent from the LSB, values are written to the CRCIN register in the order of 78H, 56H, 34H, and 12H, enabling a value of 08F6H to be obtained from the CRCD register. This is the result obtained by executing a CRC operation on the bit rows shown below, which consist of the data 12345678H inverted in bit order.



Caution Because the debugger rewrites the software break setting line to a break instruction during program execution, the CRC operation result differs if a software break is set in the CRC operation target area.

20.3.2.1 CRC input register (CRCIN)

This register is an 8-bit register to set the CRC operation data of the general-purpose CRC. The possible setting range is 00H to FFH. The CRCIN register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 20 - 4 Format of CRC Input Register (CRCIN)

Address: FFFACH
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
CRCIN								

Bits 7 to 0	Function
00H to FFH	Data input

20.3.2.2 CRC data register (CRCD)

This register holds the CRC operation result of the general-purpose CRC. The setting range is 0000H to FFFFH. After one clock cycle of the CPU/peripheral hardware clock (f_{CLK}) has elapsed from the time CRCIN register was written to, the CRC operation result is stored in the CRCD register. The CRCD register can be set by a 16-bit memory manipulation instruction. The value of this register following a reset is 0000H.

Figure 20 - 5 Format of CRC Data Register (CRCD)

Address: F02FAH, F02FBH
After reset: 0000H
R/W: R/W

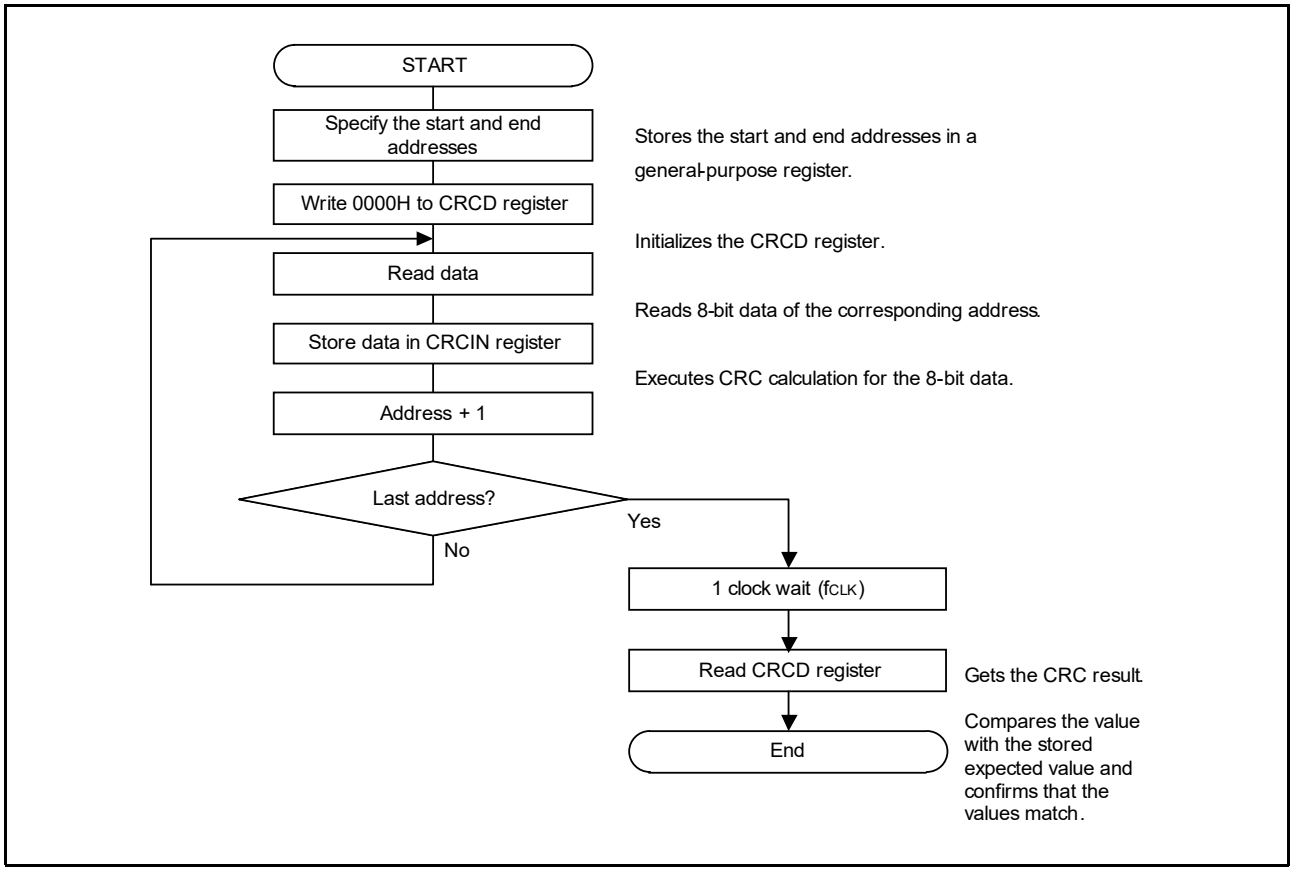
Symbol	15	14	13	12	11	10	9	8
CRCD								

7	6	5	4	3	2	1	0

Caution 1. Read the value written to the CRCD register before writing to the CRCIN register.
Caution 2. If conflict between writing a value and storing the operation result in the CRCD register occurs, the writing is ignored.

<Operation flow>

Figure 20 - 6 CRC Operation (General-purpose CRC)



20.3.3 Flash memory guard function

To ensure safe operation, the IEC60730 standard requires protecting flash memory from rewriting of its data due to incorrect CPU operations. Products of this group provide functionality to protect data in the code flash memory, data flash memory, and the security area in the flash memory. Enabling this function disables writing to the protected area of the flash memory. Reading from the protected area is possible.

20.3.3.1 Code flash memory guard register (GFLASH0)

This register is used to protect the code flash memory against being rewritten. To allow rewriting of the code flash memory, set the GFLASH0.GFL0 bit to 0. To protect the code flash memory, set the GFLASH0.GFL0 bit to 1. Be sure to confirm that the value has been set to 1 before reading from the code flash memory. The GFLASH0 register can be set by a 16-bit memory manipulation instruction. The value of this register following a reset is 0000H.

Figure 20 - 7 Format of Code Flash Memory Guard Register (GFLASH0)

Address: F0488H, F0489H
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
GFLASH0	RKEY[7:0]							

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	GFL0

RKEY[7:0]	Function
Bits 7 to 0	The RKEY[7:0] bits contain the key code to control rewriting of this register. When rewriting the GFL0 bit, set RKEY[7:0] to 30H. The RKEY[7:0] bits return 00H when read.

GFL0	Control of rewriting the code flash memory
0	Disables protection of the code flash memory (rewriting is allowed).
1	Enables protection of the code flash memory (rewriting is not allowed).

20.3.3.2 Data flash memory guard register (GFLASH1)

This register is used to protect the data flash memory against being rewritten. To allow rewriting of the data flash memory, set the GFLASH1.GFL1 bit to 0. To protect the data flash memory, set the GFLASH1.GFL1 bit to 1. Be sure to confirm that the value has been set to 1 before reading from the data flash memory. The GFLASH1 register can be set by a 16-bit memory manipulation instruction. The value of this register following a reset is 0000H.

Figure 20 - 8 Format of Data Flash Memory Guard Register (GFLASH1)

Address: F048AH, F048BH
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
GFLASH1	RKEY[7:0]							

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	GFL1

RKEY[7:0]	Function
Bits 7 to 0	The RKEY[7:0] bits contain the key code to control rewriting of this register. When rewriting the GFL1 bit, set RKEY[7:0] to C5H. The RKEY[7:0] bits return 00H when read.

GFL1	Control of rewriting the data flash memory
0	Disables protection of the data flash memory (rewriting is allowed).
1	Enables protection of the data flash memory (rewriting is not allowed).

20.3.3.3 Flash security area guard register (GFLASH2)

This register is used to protect the flash memory's security area, which holds the security settings. To allow rewriting of the flash memory's security area, set the GFLASH2.GFL2 bit to 0. The GFLASH2 register can be set by a 16-bit memory manipulation instruction. The value of this register following a reset is 0000H.

Figure 20 - 9 Format of Flash Security Area Guard Register (GFLASH2)

Address: F048CH, F048DH
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
GFLASH2	RKEY[7:0]							

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	GFL2

RKEY[7:0]	Function
Bits 7 to 0	The RKEY[7:0] bits contain the key code to control rewriting of this register. When rewriting the GFL2 bit, set RKEY[7:0] to 9AH. The RKEY[7:0] bits return 00H when read.

GFL2	Control of rewriting the flash memory's security area
0	Disables protection of the security area in the flash memory (rewriting is allowed).
1	Enables protection of the security area in the flash memory (rewriting is not allowed).

20.3.4 RAM parity error detection

The IEC60730 standard mandates the checking of RAM data. A single-bit parity bit is therefore added to all 8-bit data in the RAM of each product of this group. By using this RAM parity error detection, the parity bit is appended when data is written, and the parity is checked when the data is read. This function can also be used to trigger a reset when a parity error occurs.

20.3.4.1 RAM parity error control register (RPECTL)

This register indicates the presence of a parity error and controls the reset triggered by a parity error. The RPECTL register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 20 - 10 Format of RAM Parity Error Control Register (RPECTL)

Address: F00F5H
After reset: 00H
R/W: R/W

Symbol	<7>	6	5	4	3	2	1	<0>
RPECTL	RPERDIS	0	0	0	0	0	0	RPEF

RPERDIS	Control of the reset triggered by a parity error
0	Enables parity error resets.
1	Disables parity error resets.

RPEF	Parity error status flag
0	No parity error has occurred.
1	A parity error has occurred.

Caution **The parity bit is appended when data is written, and the parity is checked when the data is read. Therefore, while RAM parity error resets are enabled (RPERDIS = 0), be sure to initialize RAM areas where data access is to proceed before reading data. The RL78's CPU executes look-ahead due to the pipeline operation, the CPU might read an uninitialized RAM area that is allocated beyond the RAM used, which causes a RAM parity error. Therefore, while RAM parity error resets are enabled (RPERDIS = 0), be sure to initialize the RAM area + 10 bytes when instructions are fetched from RAM areas.**

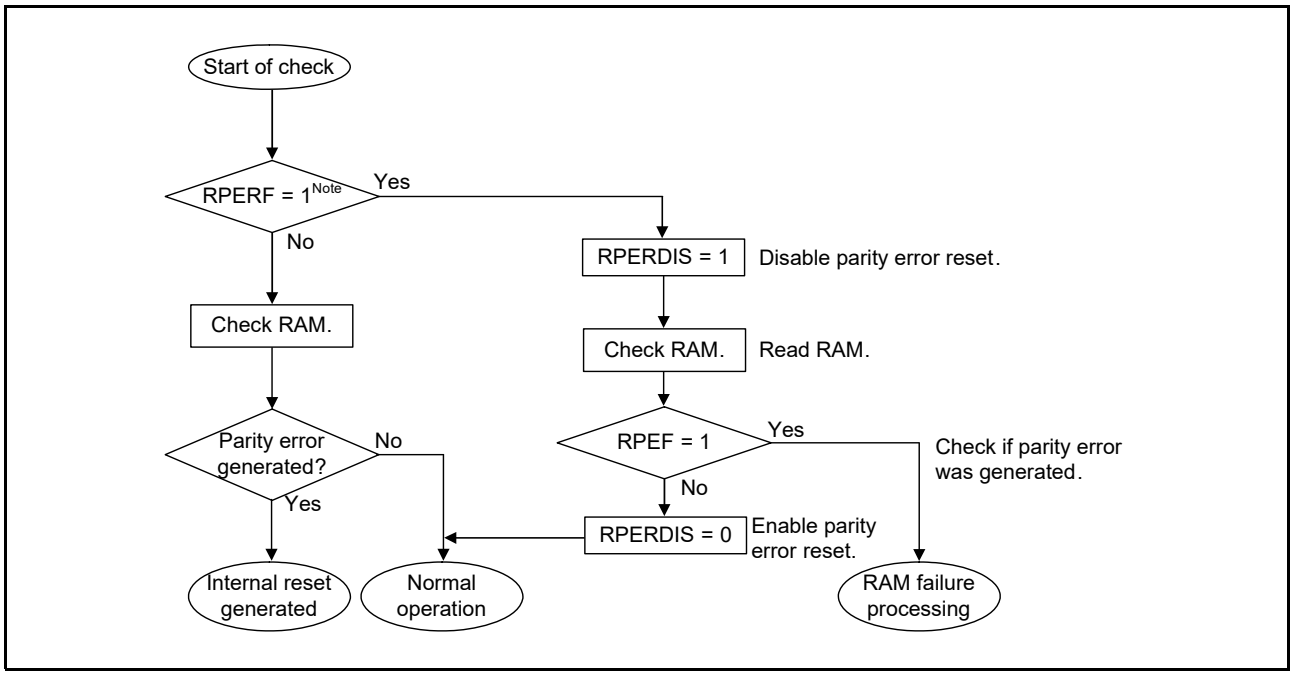
Remark 1. The parity error reset is enabled by default (RPERDIS = 0).

Remark 2. Even if the parity error reset is disabled (RPERDIS = 1), the RPEF flag will be set to 1 if a parity error occurs. If parity error resets are enabled (RPERDIS = 0) with the RPEF flag set to 1, a parity error reset is generated when the RPERDIS bit is cleared to 0.

Remark 3. The RPEF flag is set to 1 when the RAM parity error occurs and cleared to 0 by writing 0 to it or by any reset source. When the RPEF flag is 1, the value is retained even if RAM for which no parity error has occurred is read.

Remark 4. The general registers are not included for RAM parity error detection.

Figure 20 - 11 Flow for Detecting the RAM Parity Error



Note To check the state following an internal reset triggered by a RAM parity error, see **Section 17 Reset Function**.

20.3.5 RAM guard function

Products of this group provide functionality to protect data in the specified memory space. Enabling this function disables writing to the specified area of the RAM. Reading from the specified area is possible.

20.3.5.1 Invalid memory access detection control register (IAWCTL)

This register is used to control the detection of illicit memory accesses, and protection of the RAM and SFRs. Use the GRAM1 and GRAM0 bits to protect the RAM. The IAWCTL register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 20 - 12 Format of Invalid Memory Access Detection Control Register (IAWCTL)

Address: F0078H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
IAWCTL	IAWEN	0	GRAM1	GRAM0	0	GPORT	GINT	GCSC

GRAM1	GRAM0	Protected area in the RAM **Note**
0	0	Disabled. Writing to the RAM is allowed.
0	1	128 bytes from the base address of the RAM
1	0	256 bytes from the base address of the RAM
1	1	512 bytes from the base address of the RAM

Note The base address of the RAM differs depending on the size of the RAM in the product.

20.3.6 SFR guard function

Products of this group provide functionality to protect the data in the control registers for use with the ports, interrupts, clock control, voltage detection, and RAM parity error detection. Enabling this function disables writing to the protected area of the SFRs. Reading from the protected area is possible.

20.3.6.1 Invalid memory access detection control register (IAWCTL)

This register is used to control the detection of illicit memory accesses and the protection of the RAM and SFRs. Use the GPORT, GINT, and GCSC bits to protect the SFRs. The IAWCTL register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 20 - 13 Format of Invalid Memory Access Detection Control Register (IAWCTL)

Address: F0078H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
IAWCTL	IAWEN	0	GRAM1	GRAM0	0	GPORT	GINT	GCSC

GPORT	Protection of the port control registers
0	Disabled. Reading from and writing to the port control registers are allowed.
1	Enabled. Writing to the port control registers is not allowed. Reading from them is allowed. [Protected SFRs] PMxx, PUxx, PIMxx, POMxx, PMCAxx, PDIDISxx Note that the port registers (Pxx) are not protected.

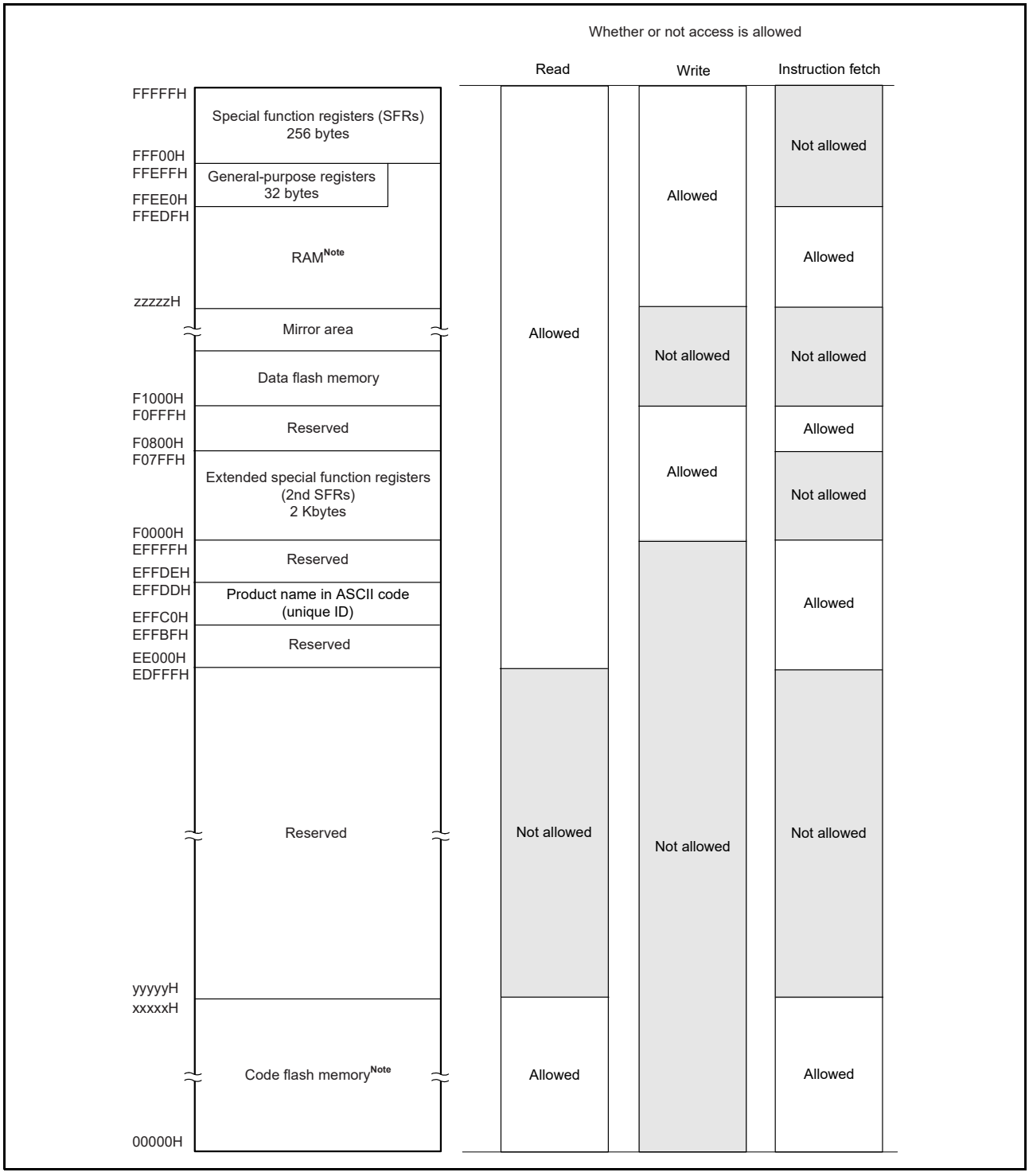
GINT	Protection of the interrupt control registers
0	Disabled. Reading from and writing to the interrupt control registers are allowed.
1	Enabled. Writing to the interrupt control registers is not allowed. Reading from them is allowed. [Protected SFRs] IFxx, MKxx, PRxx, EGPx, EGNx

GCSC	Protection of the clock, voltage detector, and RAM parity error detection control registers
0	Disabled. Reading from and writing to the clock, voltage detector, and RAM parity error detection control registers are allowed.
1	Enabled. Writing to the clock, voltage detector, and RAM parity error detection control registers is not allowed. Reading from them is allowed. [Protected SFRs] CMC, CSC, OSTS, CKC, PERx, OSMC, LVIM, LVIS, RPECTL, CKSEL, PRRx

20.3.7 Illicit memory access detection

The IEC60730 standard mandates checking that the CPU and interrupts are operating correctly. Products of this group provide functionality to trigger a reset when an invalid memory area is accessed. Access to the areas indicated as “Not allowed” in **Figure 20 - 14** is detected as illicit.

Figure 20 - 14 Illicit Access Areas



Note The following table lists the capacity and address of the code flash memory and RAM, and the lowest address of the area to be detected as illicit when accessed of each product.

Products	Code flash memory (00000H to xxxxxH)	RAM (zzzzzH to FFEFFH)	Lowest address of the area to be detected as illicit when accessed for reading or instruction fetching (yyyyyH)
R7F103GxG (x = 6, B, F, G, L)	131072 × 8 bits (00000H to 1FFFFH)	8192 × 8 bits (FDF00H to FFEFFH)	20000H
R7F103GxJ (x = 6, B, F, G, L)	262144 × 8 bits (00000H to 3FFFFH)	8192 × 8 bits (FDF00H to FFEFFH)	40000H

20.3.7.1 Invalid memory access detection control register (IAWCTL)

This register is used to control the detection of illicit memory accesses and the protection of the RAM and SFRs. Use the IAWEN bit to protect the SFRs. The IAWCTL register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 20 - 15 Format of Invalid Memory Access Detection Control Register (IAWCTL)

Address: F0078H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
IAWCTL	IAWEN	0	GRAM1	GRAM0	0	GPORT	GINT	GCSC

IAWEN[Note]	Control of illicit memory access detection
0	Disables the detection of illicit memory accesses.
1	Enables the detection of illicit memory accesses.

Note Only writing 1 to the IAWEN bit has an effect. Writing 0 to it has no effect after the IAWEN bit has been set to 1.

Remark When the setting of the WDTON bit in the user option byte (000C0H) is 1 (watchdog timer operation enabled), the illicit memory access detection is enabled even if the IAWEN bit is 0.

20.3.8 Guard function of invalid memory access detection control register

To ensure safe operation, the IEC60730 standard requires the setting for enabling or disabling illicit memory access detection to be protected from being rewritten even if a CPU malfunction occurs. Products of this group provide functionality to protect the invalid memory access detection control register (IAWCTL) against being rewritten. Enabling the protection of the invalid memory access detection control register disables writing to the given register. Reading from the protected register is possible.

20.3.8.1 Guard register of IAWCTL register (GIAWCTL)

This register is used to protect the setting for enabling or disabling the illicit memory access detection. To allow rewriting of the invalid memory access detection control register (IAWCTL), set the GIAWCTL.GIA bit to 0 to disable protection of the IAWCTL register. The GIAWCTL register can be set by a 16-bit memory manipulation instruction. The value of this register following a reset is 0000H.

Figure 20 - 16 Format of Guard Register of IAWCTL Register (GIAWCTL)

Address: F048EH, F048FH
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
GIAWCTL	RKEY[7:0]							

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	GIA

RKEY[7:0]	Function
Bits 7 to 0	The RKEY[7:0] bits contain the key code to control rewriting of this register. When rewriting the GIA bit, set RKEY[7:0] to A4H. The RKEY[7:0] bits return 00H when read.

GIA	Control of rewriting the IAWCTL register
0	Disables protection of the IAWCTL register (rewriting is allowed).
1	Enables protection of the IAWCTL register (rewriting is not allowed).

20.3.9 Frequency detection

The IEC60730 standard mandates checking that the oscillation frequency is correct. By using the CPU/peripheral hardware clock frequency (f_{CLK}) to measure the pulse width of the input signal to channel 5 of the timer array unit 0 (TAU0), whether the proportional relationship between the two clock frequencies is correct can be determined. Note that, however, if one or both clock operations are completely stopped, the proportional relationship between the clocks cannot be determined.

<Clock frequencies to be compared>

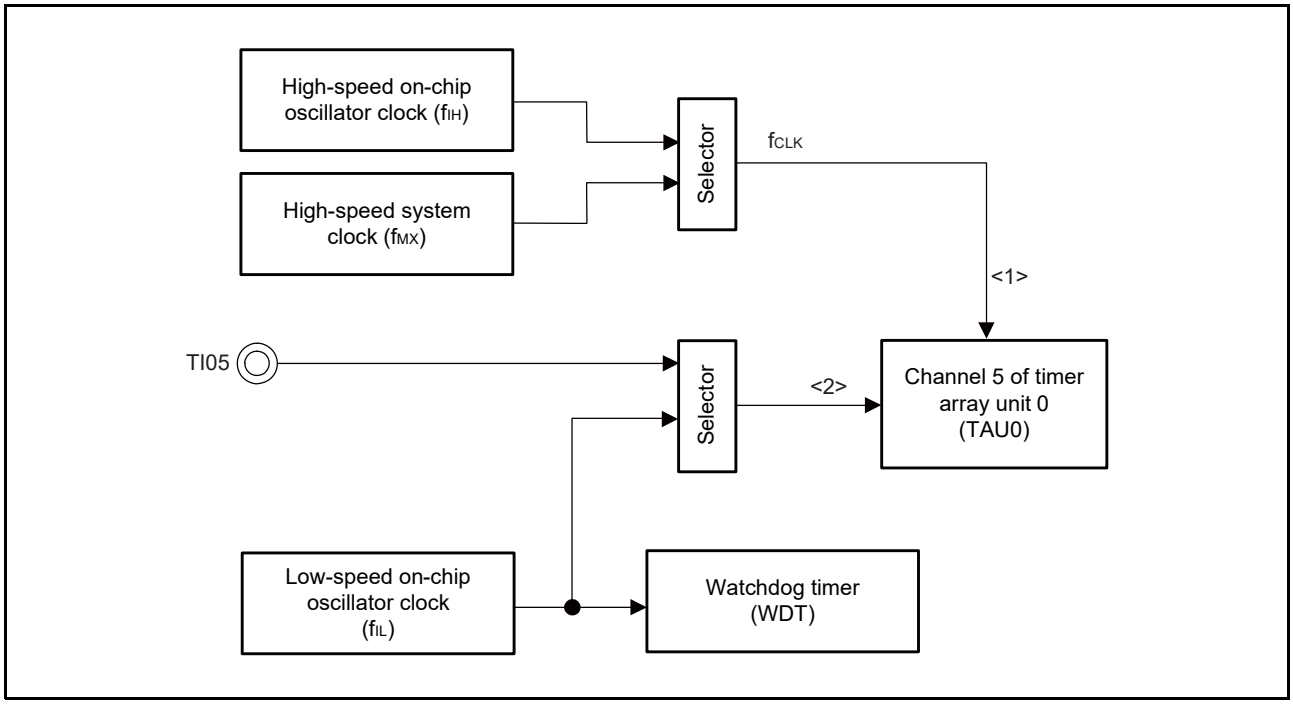
<1> CPU/peripheral hardware clock frequency (f_{CLK}):

- High-speed on-chip oscillator clock (f_{IH})
- High-speed system clock (f_{MX})

<2> Input to channel 5 of the timer array unit 0:

- Timer input to channel 5 (TI05)
- Low-speed on-chip oscillator clock (f_{IL})

Figure 20 - 17 Configuration of Frequency Detection Function



If the results of input pulse interval measurement are abnormal, the clock frequency is considered to be abnormal. For details on the input pulse interval measurement, see **7.8.4 Operation for input pulse interval measurement**.

20.3.9.1 Timer input select register 0 (TIS0)

The TIS0 register is used to select the timer input of channel 5 of the timer array unit 0 (TAU0). The TIS0 register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 20 - 18 Format of Timer Input Select Register 0 (TIS0)

Address: F0074H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
TIS0	0	0	0	0	0	TIS02	TIS01	TIS00

TIS02	Selection of timer input used with channel 5
0	Input signal of timer input pin (TI05)
1	Low-speed on-chip oscillator clock (f_{IL})

20.3.10 Testing of the A/D converter

The IEC60730 standard mandates testing of the A/D converter. This test checks whether or not the A/D converter is operating normally by converting the A/D converter's positive and negative reference voltages, analog input channels (ANIxx), temperature sensor output voltage, and internal reference voltage. For details on the method of checking, refer to the **application note (R01AN5607) Safety Function (A/D test)**.

Use the following procedure to check the analog multiplexer.

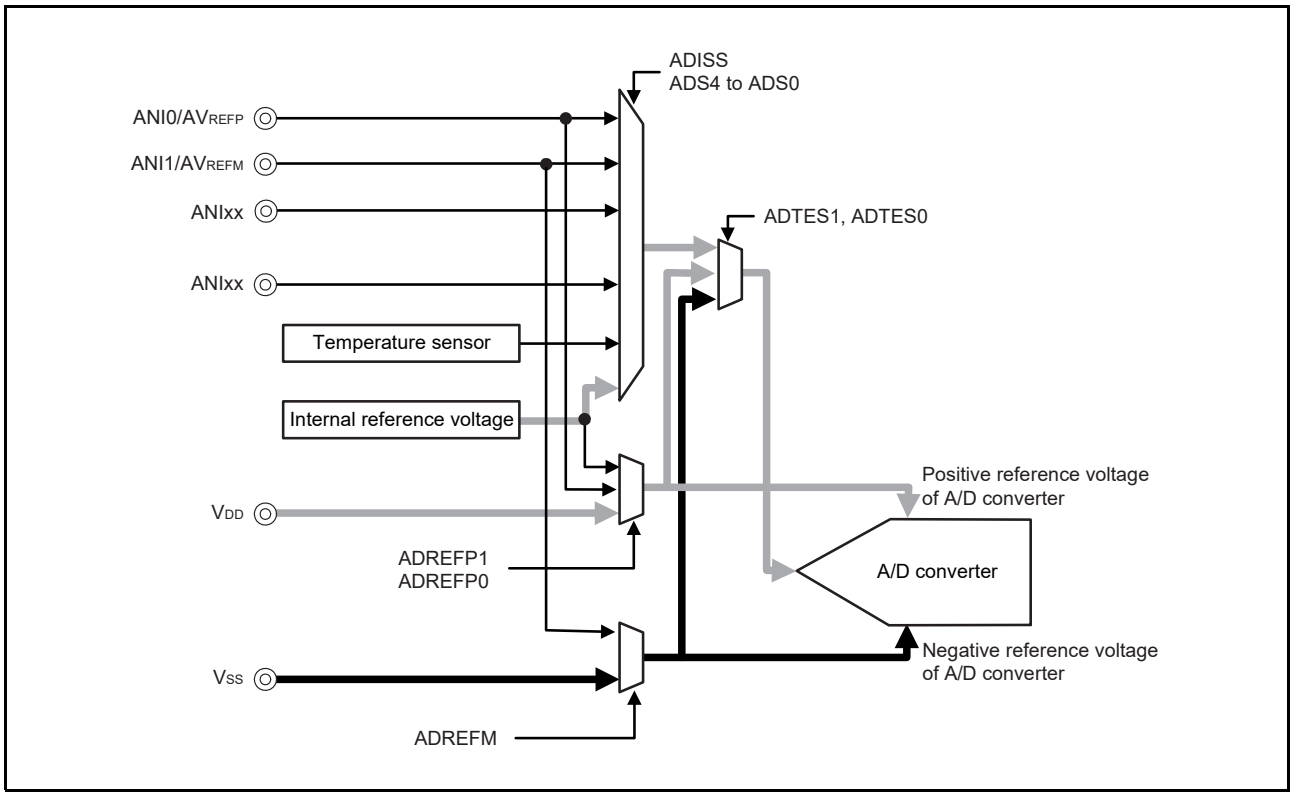
<1> Select the ANIxx pin for A/D conversion using the ADTES register (ADTES1, ADTES0 = 00B).
<2> Perform A/D conversion for the ANIxx pin (conversion result 1-1).
<3> Select the A/D converter's negative reference voltage for A/D conversion using the ADTES register (ADTES1, ADTES0 = 10B)
<4> Perform A/D conversion of the negative reference voltage of the A/D converter (conversion result 2-1).
<5> Select the ANIxx pin for A/D conversion using the ADTES register (ADTES1, ADTES0 = 00B).
<6> Perform A/D conversion for the ANIxx pin (conversion result 1-2).
<7> Select the A/D converter's positive reference voltage for A/D conversion using the ADTES register (ADTES1, ADTES0 = 11B)
<8> Perform A/D conversion of the positive reference voltage of the A/D converter (conversion result 2-2).
<9> Select the ANIxx pin for A/D conversion using the ADTES register (ADTES1, ADTES0 = 00B).
<10> Perform A/D conversion for the ANIxx pin (conversion result 1-3).
<11> Check that the conversion results 1-1, 1-2, and 1-3 are equal.
<12> Check that the A/D conversion result 2-1 is all zero and the A/D conversion result 2-2 is all one.

Using the procedure above can confirm that the analog multiplexer is selected and all wiring is connected.

Remark 1. If the analog input voltage is variable during A/D conversion in steps <1> to <10> above, use another method to check the analog multiplexer.

Remark 2. The results of conversion might include an error. Consider an appropriate level of error in comparison of the results of conversion.

Figure 20 - 19 Configuration of Testing of the A/D Converter



20.3.10.1 A/D test register (ADTES)

This register is used to select the A/D converter's positive reference voltage, A/D converter's negative reference voltage, analog input channels (ANIxx), temperature sensor output voltage, or internal reference voltage as the target of A/D conversion. When testing the A/D converter, specify the following settings:

- Select the negative reference voltage as the target of A/D conversion for zero-scale measurement.
- Select the positive reference voltage as the target of A/D conversion for full-scale measurement.

The ADTES register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 20 - 20 Format of A/D Test Register (ADTES)

Address: F0013H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
ADTES	0	0	0	0	0	0	ADTES1	ADTES0

ADTES1	ADTES0	A/D conversion target
0	0	ANIxx/temperature sensor output voltage/internal reference voltage[Note] (This is specified using the analog input channel specification register (ADS).)
1	0	Negative reference voltage (selected with the ADREFM bit in the ADM2 register)
1	1	Positive reference voltage (selected with the ADREFP1 or ADREFP0 bit in the ADM2 register)
Other than the above		Setting prohibited

Note For the internal reference voltage, see **Section 28 Electrical Characteristics**.

Caution Be sure to clear bits 7 to 2 to 0.

20.3.10.2 Analog input channel specification register (ADS)

This register specifies the input channel of the analog voltage to be A/D converted. Set the A/D test register (ADTES) to 00H when measuring ANIxx, temperature sensor output voltage, or internal reference voltage. The ADS register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 20 - 21 Format of Analog Input Channel Specification Register (ADS)

Address: FFF31H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
ADS	ADISS	0	0	ADS4	ADS3	ADS2	ADS1	ADS0

Select mode (ADM0.ADMD = 0)

ADISS	ADS4	ADS3	ADS2	ADS1	ADS0	Analog input channel	Input source
0	0	0	0	0	0	ANI0	P20/ANI0/AVREFP pin
0	0	0	0	0	1	ANI1	P21/ANI1/AVREFM pin
0	0	0	0	1	0	ANI2	P22/ANI2 pin
0	0	0	0	1	1	ANI3	P23/ANI3 pin
0	0	0	1	0	0	ANI4	P24/ANI4 pin
0	0	0	1	0	1	ANI5	P25/ANI5 pin
0	0	0	1	1	0	ANI6	P26/ANI6 pin
0	0	0	1	1	1	ANI7	P27/ANI7 pin
0	1	0	0	0	0	ANI16	P03/ANI16 pin Note 1
0	1	0	0	0	1	ANI17	P02/ANI17 pin Note 2
0	1	0	0	1	0	ANI18	P147/ANI18 pin
0	1	0	0	1	1	ANI19	P120/ANI19 pin
1	0	0	0	0	0	—	Temperature sensor output voltage
1	0	0	0	0	1	—	Internal reference voltage
Other than the above						Setting prohibited	

Note 1. The P01/ANI16 pin is specified in the 20-pin and 32-pin products.

Note 2. The P00/ANI17 pin is specified in the 20-pin and 32-pin products.

Caution 1. Be sure to clear bits 6 and 5 to 0.

Caution 2. Select input mode for the ports which are set to analog input with the PMCAxx registers, using the port mode registers (PM0, PM2, PM12, and PM14).

Caution 3. Do not use the ADS register to set the pins which should be set as digital I/O with the PMCAxx registers (PMCA0, PMCA2, PMCA12, and PMCA14).

Caution 4. Only rewrite the value of the ADISS bit while the conversion operation is stopped (ADCS = 0, ADCE = 0).

Caution 5. If using AVREFP as the positive reference voltage source of the A/D converter, do not select ANI0 as an A/D conversion channel.

Caution 6. If using AVREFM as the negative reference voltage source of the A/D converter, do not select ANI1 as an A/D conversion channel.

(Cautions are listed on the next page.)

Caution 7. When the ADISS bit is set to 1, the internal reference voltage cannot be used for the positive reference voltage. In addition, the first conversion result obtained after setting the ADISS bit to 1 is not available. For a detailed setting flow, see 10.7.4 Example of settings when temperature sensor output voltage or internal reference voltage is selected in the software trigger mode and one-shot conversion mode.

Caution 8. If a transition is made to STOP mode or a transition is made to HALT mode during CPU operation with subsystem clock, do not set the ADISS bit to 1. When the ADISS bit is 1, the A/D converter reference voltage current (I_{ADREF}) shown in 28.3.2 Characteristics of the supply current is added.

20.3.11 Detection of the digital output signal level of the I/O pins

The IEC60730 standard mandates checking that the I/O function is operating correctly. Products of this group provide functionality to read the output level of an I/O pin when the pin is in the output mode.

20.3.11.1 Port mode select register (PMS)

This register is used to select whether to read the output latch value of a port or read the output level of a pin when the pin is in the output mode (PMmn bit of port mode register (PMm) is 0). The PMS register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 20 - 22 Format of Port Mode Select Register (PMS)

Address: F007BH
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
PMS	0	0	0	0	0	0	0	PMS0

PMS0	Selection of data to be read when pin is in output mode
0	Reads the value of the Pmn register.
1	Reads the digital output level of the pin.

Caution 1. Be sure to clear bits 7 to 1 to 0.

Caution 2. Do not rewrite a port register (Pxx) with a 1-bit memory manipulation instruction when the setting of the PMS0 bit in the PMS register is 1. Use an 8-bit memory manipulation instruction to rewrite a port register (Pxx).

Caution 3. The setting of this register has no effect on the input port pin (P137) and output port pin (P130).

Remark m = 0 to 7, 12, 14
n = 0 to 7

Caution 4. The setting of this register has no effect when the settings of the PMCAmn and PMCTmn bits are both 1. To read the digital output level of a pin with the PMCAmn bit and PMCTmn bit both set to 1, first clear the PMCAmn and PMCTmn bits to 0 and then set the PMS0 bit to 1.

Remark m = 0, 2, 12, 14
n = 0 to 7

Caution 5. The setting of this register has no effect when the setting of the PDIDISmn bit is 1. To read the digital output level of a pin with the PDIDISmn bit set to 1, first clear the PDIDISmn bit to 0 and then set the PMS0 bit to 1.

Remark m = 0, 1, 3, 5, 7
n = 0 to 7

20.3.12 UART loopback

The IEC60730 standard recommends to diagnose abnormalities of external interfaces (communications). The UART loopback is used to confirm the normal output of the UART transmit data by shutting off the RxD pin from the outside and connecting it within the MCU to loop back the output from the transmission shift register to the reception shift register.

When the UART loopback is selected, the transmit data from the TxD pin is controlled by the port functions so that it does not affect its communication partner.

- Communication with negative logic
  The port pin multiplexed with the TxD pin is set to the input mode (PMxx = 1) and an on-chip pull-up resistor is connected (PUxx = 1) to retain the setting of 1.
- Communication with positive logic
  0 (PMxx = 0, Pxx = 0) is output from the port that also uses the TxD pin.

20.3.12.1 UART loopback select register (ULBS)

The ULBS register is used to enable the UART loopback. This register has respective bits for independently controlling each UART channel. Setting the bit corresponding to each channel to 1 will select the UART loopback and loop back the output of the transmission shift register to the reception shift register. The ULBS register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 20 - 23 Format of UART Loopback Select Register (ULBS)

Address: F0079H
After reset: 00H
R/W: R/W

Symbol	7	6	5	<4>	3	<2>	<1>	<0>
ULBS	0	0	0	ULBS4	0	ULBS2	ULBS1	ULBS0

ULBS4	UART loopback selection
0	Inputs the state of the RxDA0 pin of serial interface UARTA0 to the reception shift register.
1	Loops back the output of the transmission shift register to the reception shift register.

ULBS2	UART loopback selection
0	Inputs the state of the RxD2 pin of serial array unit UART2 to the reception shift register.
1	Loops back the output of the transmission shift register to the reception shift register.

ULBS1	UART loopback selection
0	Inputs the state of the RxD1 pin of serial array unit UART1 to the reception shift register.
1	Loops back the output of the transmission shift register to the reception shift register.

ULBS0	UART loopback selection
0	Inputs the state of the RxD0 pin of serial array unit UART0 to the reception shift register.
1	Loops back the output of the transmission shift register to the reception shift register.

Caution Be sure to clear bits 7 to 5 and 3 to 0.

Section 21 Security Functions

21.1 Flash Read Protection

21.1.1 Function of flash read protection

The flash read protection can be used to protect a specified range of the code flash memory area against read access by the CPU. Note that fetching of instructions in the specified range by the CPU is still possible.
In addition, verifying the code flash memory area protected against read access by the flash read protection is possible in the serial programming mode.

21.1.2 Setting of flash read protection

The settings for flash read protection are made through serial programming by using a flash memory programmer or through self-programming in the extra area. Read access to the whole range of the code flash memory is enabled by the default setting at the time of shipment. Set a range in the code flash memory to be protected against read access by specifying the flash read protection start block and end block, and disable changing of the flash read protection settings. This makes read access to addresses in the specified range of blocks in the code flash memory impossible. In addition, changing of the flash read protection settings can be disabled, which makes the settings for the start and end blocks fixed, thus making changes to the blocks where flash read protection starts and ends impossible. However, disabling changing of the flash read protection settings can be released by using a flash memory programmer. Note that release from disabling of changes to the flash read protection settings requires deletion of the contents of the full range of the code flash memory. Accordingly, reading the data written in the range protected against read access is impossible after release from disabling of changes to the flash read protection settings. **Table 21 - 1** describes the settings for flash read protection and their functions. **Table 21 - 2** describes the method of setting flash read protection.

Table 21 - 1 Settings for Flash Read Protection and their Functions

Item to Be Set	Function
Block where flash read protection starts	Specifies the number of the block where the read-access disabled area starts. The specifiable values are in the range from the block numbered 001H to the number of the block at the highest-order address in the code flash memory. The block for which the number is set as the block where protection starts is part of the read-access disabled area. Setting 000H as the block where protection starts is prohibited. The initial setting is 1FFH.
Block where flash read protection ends	Specifies the number of the block where the read-access disabled area ends. The specifiable values are in the range from the number of the block where the protection starts to the number of the block at the highest-order address in the code flash memory. The block for which the number is set as the block where protection ends is part of the read-access disabled area. The initial setting is 1FFH.
Disabling changing of the flash read protection settings	Fixes the settings of the blocks where flash read protection starts and ends. When the “enabled” setting is made, changes to the blocks where protection starts and ends are not possible.

For details on the relationship between the addresses and block numbers, refer to **Table 3 - 1 Correspondence between Addresses and Block Numbers in Flash Memory**.

Table 21 - 2 Method of Setting Flash Read Protection

Item to Be Set	Method of Setting	Method of Changing
Block where flash read protection starts	Using a flash memory programmer or self-programming.	Using a flash memory programmer or self-programming. Note that the block where protection starts is not adjustable while changing of the flash read protection settings is disabled.
Block where flash read protection ends	Using a flash memory programmer or self-programming.	Using a flash memory programmer or self-programming. Note that the block where protection ends is not adjustable while changing of the flash read protection settings is disabled.
Disabling changing of the flash read protection settings	Using a flash memory programmer or self-programming.	Fixing of the flash read protection settings can be released by using a flash memory programmer.Note If you do so, the values for the start and end blocks are initialized.

Note Release from the fixed setting is only possible when erasure of blocks is not prohibited, rewriting of boot area is not prohibited, and the code and data flash memory areas are blank.

Caution 1. The settings for flash read protection in the extra area are not readable. To confirm that the settings for flash read protection are in place, read from the read-access disabled area and confirm that FFH is returned.

Caution 2. To specify the read-access disabled area for flash read protection, be sure to specify the numbers of both the block where protection starts and the block where it ends.

Caution 3. Reading from the read-access disabled area by using an on-chip debugger is also impossible. This means that program code allocated to the read access-disabled area cannot be debugged by using the on-chip debugger. Therefore, only make the settings for flash read protection after having debugged the program code in the protected areas.

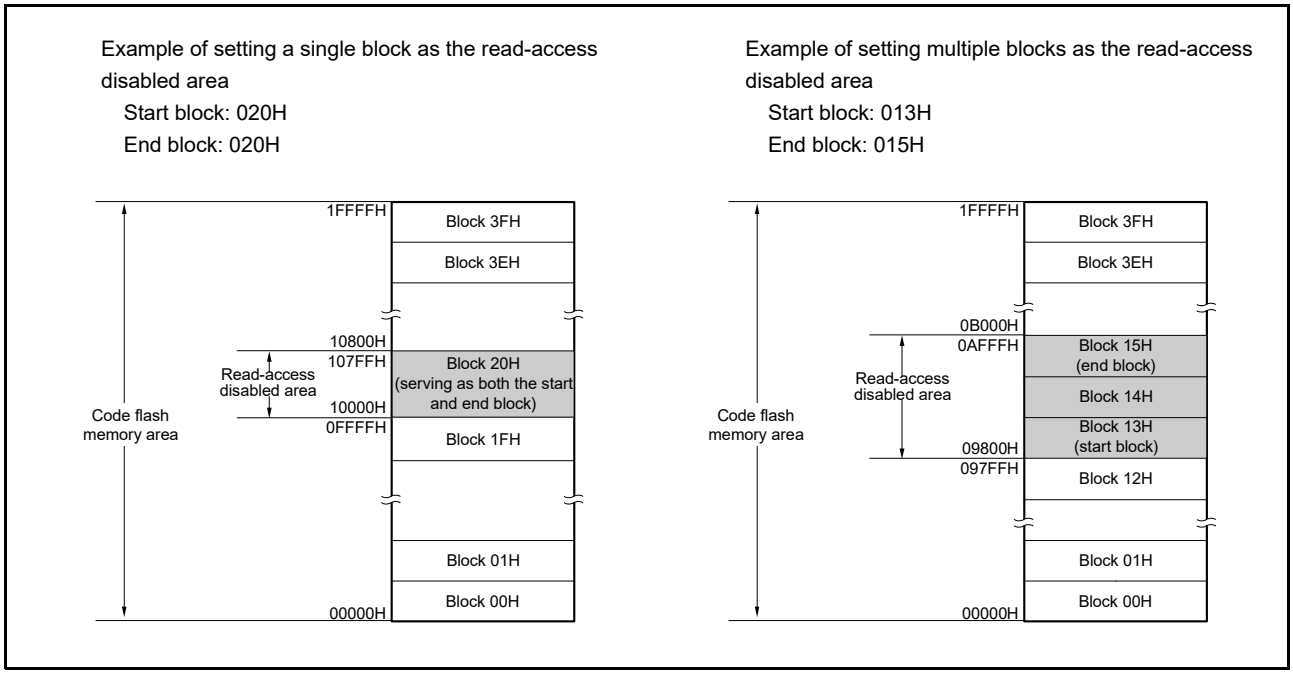
Caution 4. When a part of boot cluster 0 or boot cluster 1 is to be set as a part of the read-access disabled area, boot swapping may cause data in the read-access disabled area to be swapped with data in the read access-enabled area. To prevent this, when setting a part of boot cluster 0 or boot cluster 1 as part of the read-access disabled area, make the setting for prohibiting the rewriting of boot area so as to prohibit boot swapping itself.

Caution 5. When settings for flash read protection have been made through self-programming, the settings become enabled after the MCU is reset and then released from the reset state.

21.1.3 Operation

Reading by the CPU from the area to which read access has been disabled with the use of flash read protection always returns FFH. Fetching of instructions in the read-access disabled area by the CPU is still possible. Note that even if program code is to be executed from the read-access disabled area, it is unable to read data that have been placed in the read-access disabled area. Place data for use with code to be executed from the read-access disabled area in areas that are not protected. **Figure 21 - 1** shows examples of setting read-access disabled areas for the application of flash read protection.

Figure 21 - 1 Examples of Setting Read-access Disabled Areas for the Application of Flash Read Protection



To enable read access to the full range of the code flash memory again after having set a range to be protected against read access, set the flash read protection start block and end block as 1FFH while prohibition of changing the flash read protection settings is disabled.

21.2 Unique ID

21.2.1 Function of a unique ID

A unique ID is a unique value that is allocated to an individual product and stored in the extra area. A unique ID is entered for each product at the time of manufacturing the MCUs. Changing the entered ID is not possible. The data length is 16 bytes (128 bits).

Caution The value of a unique ID is not a random number.

21.2.2 ASCII codes representing the product name

Product names are also stored as strings of ASCII codes in the extra area. Since the product names include indicators of the number of pins and capacity of the flash memory for the given product, conditional branch processing on the basis of the product name is possible. **Table 21 - 3** shows the places of the unique ID, basic product name, and indicators of the number of pins and capacity of the flash memory in the memory map.

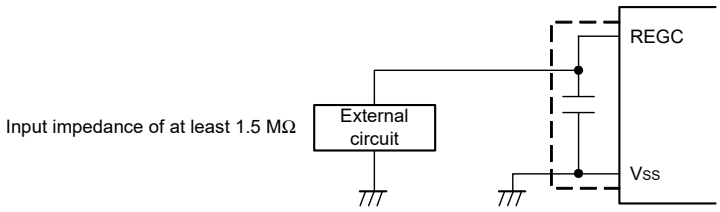
Table 21 - 3 Memory Map of the Unique ID and Product Name

Address	Item Name	Value to Be Entered
EFFC0H to EFFCFH	Unique ID	The unique value allocated to an individual product
EFFD5H	ASCII codes representing the product name	52H: “R”
EFFD6H		37H: “7”
EFFD7H		46H: “F”
EFFD8H		31H: “1”
EFFD9H		30H: “0”
EFFDAH		33H: “3”
EFFDBH		47H: “G”
EFFDCH		36H: “6” (20 pins) 42H: “B” (32 pins) 46H: “F” (44 pins) 47H: “G” (48 pins) 4CH: “L” (64 pins)
EFFDDH		47H: “G” (128-Kbyte flash memory) 4AH: “J” (256-Kbyte flash memory)

Section 22 Regulator

22.1 Overview

Each product of this group incorporates a circuit for constant voltage operation. To stabilize the output voltage from the regulator, connect the REGC pin to Vss via a capacitor (0.47 to 1 μF). Use a capacitor with good characteristics, since it is for stabilizing the internal voltage. The REGC pin can be used as a reference voltage for an external circuit. An external circuit for connection to the REGC pin for this purpose must have an input impedance of at least 1.5 MΩ. The voltage on the REGC pin is in the range from 1.38 to 1.60 V, and the typical value is 1.5 V.



Caution The length of the wiring within the broken lines in the above figure should be as short as possible.

The regulator outputs a 1.5-V voltage.

Section 23 Option Bytes

23.1 Functions of Option Bytes

Addresses 000C0H to 000C3H of the flash memory in each product of this group form an option byte area. Option bytes consist of user option bytes (000C0H to 000C2H) and on-chip debug option byte (000C3H). Upon power application or resetting and starting, an option byte is automatically referenced and a specified function is set. When using the product, be sure to set the following functions by using the option bytes. For the bits to which no function is allocated, do not change their initial values. To use the boot swap operation during self-programming, 000C0H to 000C3H are replaced by 040C0H to 040C3H. Therefore, set the same values as 000C0H to 000C3H to 040C0H to 040C3H.

Caution The option bytes should always be set regardless of whether each function is used.

23.1.1 User option bytes (000C0H to 000C2H or 040C0H to 040C2H)

1. 000C0H or 040C0H
   - Setting of watchdog timer operation
     - Enabling or disabling of counter operation
     - Enabling or stopping of counter operation in the HALT or STOP mode
   - Setting of overflow time of watchdog timer
   - Setting of window open period of watchdog timer
   - Setting of interval interrupt of watchdog timer
     - Interval interrupt is used or not used

Caution When boot swapping is to be used, 000C0H is replaced with 040C0H. Therefore, set the same value as the setting in 000C0H in 040C0H.

2. 000C1H or 040C1H
   - Setting of the operating mode of LVD0
     - Reset mode
     - Interrupt mode
     - LVD0 off (by controlling the externally input reset signal on the $\overline{\text{RESET}}$ pin)
   - Setting of LVD0 detection level (V_{LVD0})
   - Control of the P125/$\overline{\text{RESET}}$/(INTP9) pin
     - Port function
     - $\overline{\text{RESET}}$ input

Caution 1. After power is supplied, the reset state must be retained until the operating voltage becomes in the range defined in 28.4 AC Characteristics. This is done by utilizing the voltage detection circuit or controlling the externally input reset signal. After the power supply is turned off, this LSI should be placed in the STOP mode, or placed in the reset state by utilizing the voltage detection circuit or controlling the externally input reset signal, before the voltage falls below the operating range. The range of operating voltage varies with the setting of the user option byte (000C2H or 040C2H).

Caution 2. When boot swapping is to be used, 000C1H is replaced with 040C1H. Therefore, set the same value as the setting in 000C1H in 040C1H.

3. 000C2H or 040C2H
   - Setting of the operating mode of the flash memory
     Make the setting depending on the main system clock frequency (f_{MAIN}) and power supply voltage (V_{DD}) to be used.
     – LS (low-speed main) mode
     – HS (high-speed main) mode
     – LP (low-power main) mode
   - Setting of the frequency of the high-speed on-chip oscillator
     – Select from 1 to 32 MHz.

Caution When boot swapping is to be used, 000C2H is replaced with 040C2H. Therefore, set the same value as the setting in 000C2H in 040C2H.

23.1.2 On-chip debug option byte (000C3H or 040C3H)

- Control of on-chip debug operation
  – On-chip debug operation is disabled or enabled.
- Handling of data of flash memory in case of failure in on-chip debug security ID authentication
  – Data of flash memory is erased or not erased in case of failure in on-chip debug security ID authentication.

Caution When boot swapping is to be used, 000C3H is replaced with 040C3H. Therefore, set the same value as the setting in 000C3H in 040C3H.

23.2 Format of User Option Bytes

Figure 23 - 1 Format of User Option Byte (000C0H or 040C0H)

Address: 000C0H or 040C0H[Note 1]

Symbol	7	6	5	4	3	2	1	0
	WDTINT	WINDOW1	WINDOW0	WDTON	WDCS2	WDCS1	WDCS0	WDSTBYON

WDTINT	Use of interval interrupt of watchdog timer
0	Interval interrupt is not used.
1	Interval interrupt is generated when 75% of the overflow time + 1/4 f_{IL} is reached.

WINDOW1	WINDOW0	Watchdog timer window open period[Note 2]
0	0	Setting prohibited
0	1	50%
1	0	Setting prohibited
1	1	100%

WDTON	Operation control of watchdog timer counter
0	Counter operation disabled (counting stopped after reset)
1	Counter operation enabled (counting started after reset)

WDCS2	WDCS1	WDCS0	Watchdog timer overflow time (f_{IL} = 37.683 kHz (max.))
0	0	0	$2^7/f_{IL}$ (3.39 ms)
0	0	1	$2^8/f_{IL}$ (6.79 ms)
0	1	0	$2^9/f_{IL}$ (13.58 ms)
0	1	1	$2^{10}/f_{IL}$ (27.17 ms)
1	0	0	$2^{12}/f_{IL}$ (108.69 ms)
1	0	1	$2^{14}/f_{IL}$ (434.78 ms)
1	1	0	$2^{15}/f_{IL}$ (869.56 ms)
1	1	1	$2^{17}/f_{IL}$ (3478.26 ms)

WDSTBYON	Operation control of watchdog timer counter (HALT/STOP mode)
0	Counter operation stopped in HALT/STOP mode[Note 2]
1	Counter operation enabled in HALT/STOP mode

Note 1. When boot swapping is to be used, 000C0H is replaced with 040C0H. Therefore, set the same value as the setting in 000C0H in 040C0H.

Note 2. The window open period is 100% when WDSTBYON = 0, regardless of the value of the WINDOW1 and WINDOW0 bits.

Remark When the setting of the WDTON bit is 1, the illicit memory access detection remains enabled regardless of the setting of the IAWEN bit in the IAWCTL register.

Figure 23 - 2 Format of User Option Byte (000C1H or 040C1H)

Address: 000C1H or 040C1H[Note]

Symbol	7	6	5	4	3	2	1	0
	LVD0EN	LVD0SEL	1	PORTSELB	1	LVD0V2	LVD0V1	LVD0V0

LVD0 setting (reset mode)

Detection Voltage		Option Byte Setting Value				
V_{LVD0}		LVD0EN	Mode setting LVD0SEL	LVD0V2	LVD0V1	LVD0V0
Rising edge	Falling edge					
1.69 V	1.65 V	1	1	1	1	1
1.90 V	1.86 V			1	1	0
2.38 V	2.33 V			1	0	1
2.67 V	2.62 V			1	0	0
2.97 V	2.91 V			0	1	1
3.96 V	3.88 V			0	1	0
—		Settings other than the above are prohibited.				

LVD0 setting (interrupt mode)

Detection Voltage		Option Byte Setting Value				
V_{LVD0}		LVD0EN	Mode setting LVD0SEL	LVD0V2	LVD0V1	LVD0V0
Rising edge	Falling edge					
1.69 V	1.65 V	1	0	1	1	1
1.90 V	1.86 V			1	1	0
2.38 V	2.33 V			1	0	1
2.67 V	2.62 V			1	0	0
2.97 V	2.91 V			0	1	1
3.96 V	3.88 V			0	1	0
—		Settings other than the above are prohibited.				

LVD0 off setting (external reset input from the $\overline{\text{RESET}}$ pin is used)

Detection Voltage		Option Byte Setting Value				
V_{LVD0}		LVD0EN	Mode setting LVD0SEL	LVD0V2	LVD0V1	LVD0V0
Rising edge	Falling edge					
—	—	0	×	0	1	0
—		Settings other than the above are prohibited.				

Control of the P125/$\overline{\text{RESET}}$/(INTP9) pin

PORTSELB	Control of the P125/$\overline{\text{RESET}}$/(INTP9) pin
0	Port function P125/(INTP9)
1	$\overline{\text{RESET}}$ input (The on-chip pull-up resistor is always enabled.)

(Note, Cautions, and Remarks are listed on the next page.)

Note When boot swapping is to be used, 000C1H is replaced with 040C1H. Therefore, set the same value as the setting in 000C1H in 040C1H.

Caution 1. Be sure to set bits 5 and 3 to 1.

Caution 2. After power is supplied, the reset state must be retained until the operating voltage becomes in the range defined in 28.4 AC Characteristics. This is done by utilizing the voltage detection circuit or controlling the externally input reset signal. After the power supply is turned off, this LSI should be placed in the STOP mode, or placed in the reset state by utilizing the voltage detection circuit or controlling the externally input reset signal, before the voltage falls below the operating range. The range of operating voltage varies with the setting of the user option byte (000C2H or 040C2H).

Caution 3. After power is supplied, P125 functions as the $\overline{\text{RESET}}$ input. The MCU remains in the reset state as long as a low-level signal is applied to P125 even if the internal reset signal is released by the power-on reset (POR). When the pin is to be used for P125 or INTP9, select the port function (PORTSELB = 0) in the user option byte (000C1H/010C1H), and release all reset sources.

Remark 1. ×: Don't care.

Remark 2. For details on LVD0, see **Section 19 Voltage Detector (LVD)**.

Remark 3. The detection voltage is a typical value. For details, see **28.6.5 Characteristics of the LVD circuit**.

Figure 23 - 3 Format of User Option Byte (000C2H or 040C2H)

Address: 000C2H or 040C2H[Note 1]

Symbol	7	6	5	4	3	2	1	0
	CMODE1	CMODE0	1	0	FRQSEL3	FRQSEL2	FRQSEL1	FRQSEL0

CMODE1	CMODE0	Operating Mode of the Flash Memory	Operating Frequency Range	Operating Voltage Range
0	1	LP (low-power main) mode[Note 2]	1 to 2 MHz (Rewriting of flash memory is not possible.)	1.6 to 5.5 V
1	0	LS (low-speed main) mode	1 to 4 MHz (Rewriting of flash memory is not possible.)	1.6 to 5.5 V
			1 to 24 MHz	1.8 to 5.5 V
1	1	HS (high-speed main) mode	1 to 4 MHz (Rewriting of flash memory is not possible.)	1.6 to 5.5 V
			1 to 32 MHz	1.8 to 5.5 V
Other than above		Setting prohibited		

FRQSEL3	FRQSEL2	FRQSEL1	FRQSEL0	Frequency of the High-speed On-chip Oscillator Clock
1	0	0	0	32 MHz
0	0	0	0	24 MHz
1	0	0	1	16 MHz
0	0	0	1	12 MHz
1	0	1	0	8 MHz
0	0	1	0	6 MHz
1	0	1	1	4 MHz
0	0	1	1	3 MHz
1	1	0	0	2 MHz
1	1	0	1	1 MHz
Other than above				Setting prohibited

Note 1. When boot swapping is to be used, 000C2H is replaced with 040C2H. Therefore, set the same value as the setting in 000C2H in 040C2H.

Note 2. In LP mode, the high-speed on-chip oscillator clock (f_{IH}) can only be selected as the CPU/peripheral hardware clock (f_{CLK}).

Caution 1. Be sure to set bits 5 and 4 to 10B.

Caution 2. The operating frequency range and operating voltage range vary depending on the operating mode of the flash memory. For details, see 28.4 AC Characteristics.

23.3 Format of On-chip Debug Option Byte

The format of on-chip debug option byte is shown below.

Figure 23 - 4 Format of On-chip Debug Option Byte (000C3H or 040C3H)

Address: 000C3H or 040C3H[Note]

Symbol	7	6	5	4	3	2	1	0
	OCDENSET	0	0	0	0	1	0	OCDERSD

OCDENSET	OCDERSD	Control of on-chip debug operation
0	0	Disables on-chip debugging.
0	1	Setting prohibited
1	0	Enables on-chip debugging. Erases data of flash memory in case of failures in authenticating on-chip debug security ID.
1	1	Enables on-chip debugging. Does not erase data of flash memory in case of failures in authenticating on-chip debug security ID.

Note When boot swapping is to be used, 000C3H is replaced with 040C3H. Therefore, set the same value as the setting in 000C3H in 040C3H.

Caution **Only bits 7 and 0 (OCDENSET and OCDERSD) are specifiable.**
Be sure to set bits 6 to 1 to 000010B.

Remark The use of on-chip debugging changes the values of bits 3 to 1. Accordingly, the values after such settings are made become undefined. However, note that when setting the option byte, be sure to set bits 3 to 1 to their default values (010B).

23.4 Setting of Option Bytes

The user option bytes and on-chip debug option byte can also be set by using linker options instead of statements in the source code. In such cases, the settings made by using linker options are given priority over the statements in the source code as shown below. An example of the statements in relation to the option byte settings in software is shown below.

```
        .CSEG   OPT_BYTE
        .DB     0x36    ; Does not use the interval interrupt of the watchdog timer.
                        ; Enables watchdog timer operation.
                        ; The window open period of the watchdog timer is 50%.
                        ; The overflow time of the watchdog timer is 2^10/fIL.
                        ; Stops watchdog timer operation during HALT/STOP mode.
        .DB     0xBF    ; Selects VLVD0 as 1.69 V on rising edges and 1.65 V on falling edges.
                        ; Select the interrupt mode as the operating mode of LVD0.
                        ; Uses the RESET input.
        .DB     0x6D    ; Select the LP (low-power main) mode as the operating mode of the flash memory
                        ; and 1 MHz as the frequency of the high-speed on-chip oscillator clock.
        .DB     0x85    ; Enables on-chip debug operation, does not erase flash memory data when
                        ; security ID authentication fails.
```

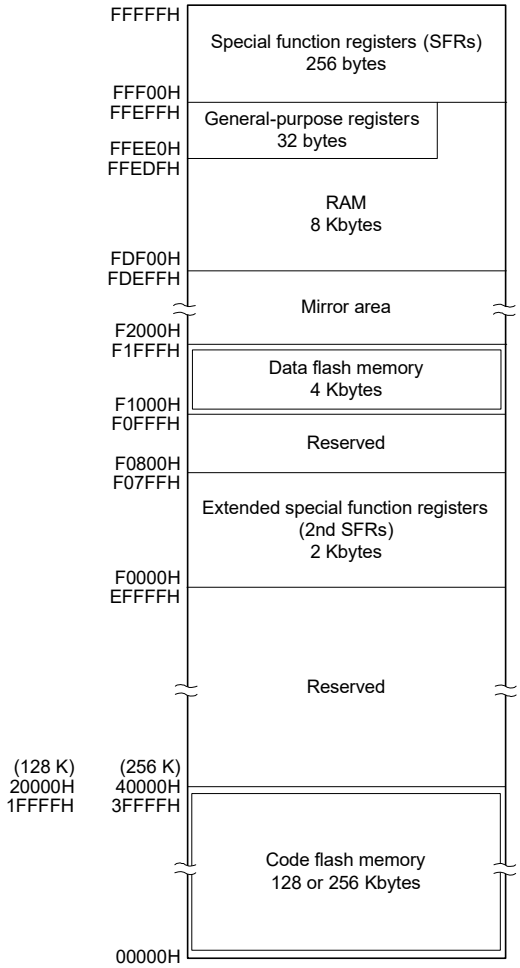
When boot swapping is to be used during self-programming, 000C0H to 000C3H are replaced with 040C0H to 040C3H. Therefore, set the same values as the settings in 000C0H to 000C3H in 040C0H to 040C3H as shown below.

```
OPT2    .CSEG   AT      0x040C0
        .DB             0x36    ; Does not use the interval interrupt of the watchdog timer.
                                ; Enables watchdog timer operation.
                                ; The window open period of the watchdog timer is 50%.
                                ; The overflow time of the watchdog timer is 2^10/fIL.
                                ; Stops watchdog timer operation during HALT/STOP mode.
        .DB             0xBF    ; Selects VLVD0 as 1.69 V on rising edges and 1.65 V on falling edges.
                                ; Select the interrupt mode as the operating mode of LVD0.
                                ; Uses the RESET input.
        .DB             0x6D    ; Select the LP (low-power main) mode as the operating mode of the flash memory
                                ; and 1 MHz as the frequency of the high-speed on-chip oscillator clock.
        .DB             0x85    ; Enables on-chip debug operation, does not erase flash memory data when
                                ; security ID authentication fails.
```

Caution To specify the option byte by using assembly language, use OPT_BYTE as the relocation attribute name of the CSEG pseudo instruction. To specify 040C0H to 040C3H for use as the option bytes in the case of boot swapping, use the relocation attribute AT to specify absolute addresses.

Section 24 Flash Memory

This MCU incorporates flash memory that allows the writing, erasure, and overwriting of programs. The flash memory consists of a code flash memory area, from which programs can be executed, a data flash memory area, which is suitable for storing data, and an extra area, which is for storing flash memory operation and security settings.



The following methods for programming the flash memory are available. The contents of code flash memory can be rewritten through serial programming by using a flash memory programmer or other external device (via UART communications), or through self-programming.

- **24.1 Serial Programming Using Flash Memory Programmer**
  Data can be written to the flash memory on-board or off-board by using a dedicated flash memory programmer.
- **24.2 Serial Programming Using External Device (that Incorporates UART)**
  Data can be written to the flash memory on-board through UART communications with an external device (a microcontroller or ASIC).
- **24.6 Self-programming**
  The user application can execute self-programming of the code flash memory.

Caution When rewriting the flash memory, do not change the flash operating mode select register (FLMODE).

The data flash memory can be rewritten to by using self-programming during user program execution (background operation). For access and writing to the data flash memory, see **24.6 Self-programming** and **24.10 Data Flash Memory**. The code flash memory and the data flash memory have a function to protect them from rewriting. For details, see **20.3.3 Flash memory guard function**.

24.1 Serial Programming Using Flash Memory Programmer

The following dedicated flash memory programmers can be used to write data to the internal flash memory of this MCU.

- PG-FP6
- E2 or E2 Lite on-chip debugging emulator

Data can be written to the flash memory on-board or off-board, by using a dedicated flash memory programmer.

1. On-board programming
   The contents of the flash memory can be rewritten after this MCU has been mounted on the target system. Mount a connector for the dedicated flash memory programmer on the target system.
2. Off-board programming
   Data can be written to the flash memory with a dedicated program adapter before this MCU is mounted on the target system.

Table 24 - 1 Wiring between This MCU and Dedicated Flash Memory Programmer

Pin Configuration of Dedicated Flash Memory Programmer				Pin Name	Pin No.				
					20-pin	32-pin	44-pin	48-pin	64-pin
Signal Name		I/O	Pin Function		LSSOP (4.4 × 6.5)	LQFP (7 × 7)	LQFP (10 × 10)	LFQFP (7 × 7)	LQFP (12 × 12) LFQFP (10 × 10)
PG-FP6	E2 or E2 Lite On-chip Debugging Emulator								
—	TOOL0	I/O	Transmit/ receive signal	TOOL0/ P40	3	1	2	39	5
SI/RxD	—	I/O							
—	RESET_OUT	Output	Reset signal	$\overline{\text{RESET}}$	4	2	3	40	6
$\overline{\text{RESET}}$	—	Output							
V_{CC}	V_{DD}	I/O	V_{DD} voltage generation/ power monitoring	V_{DD}	10	8	11	48	15
GND		—	Ground	V_{SS}	9	7	10	47	13
				REGC **Note**	8	6	9	46	12
FLMD1	EMV_{DD}	—	Driving power for TOOL0 pin	V_{DD}	10	8	11	48	15

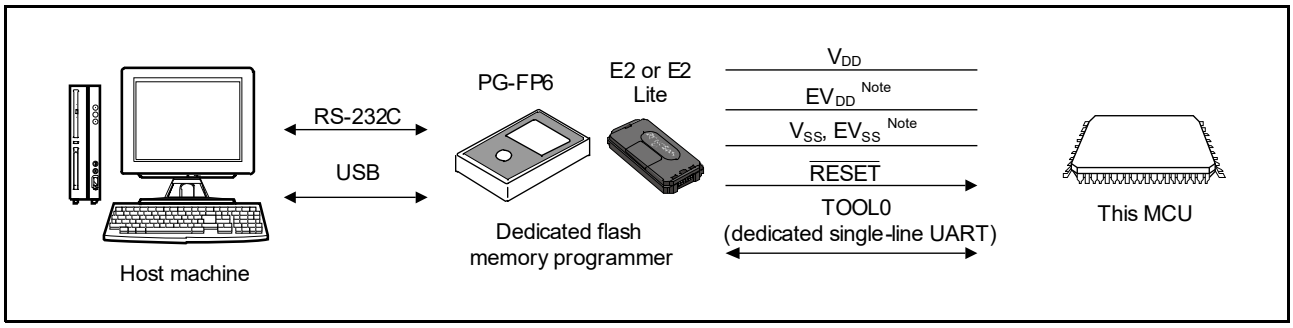
Note Connect the REGC pin to the ground via a capacitor (0.47 to 1 μF).

Remark Pins that are not indicated in the above table can be left open-circuit when using the flash memory programmer for flash memory programming.

24.1.1 Programming environment

Figure 24 - 1 shows the environment required for writing a program to the flash memory of this MCU.

Figure 24 - 1 Environment for Writing Program to Flash Memory



Note This pin is only present in the 64-pin products.

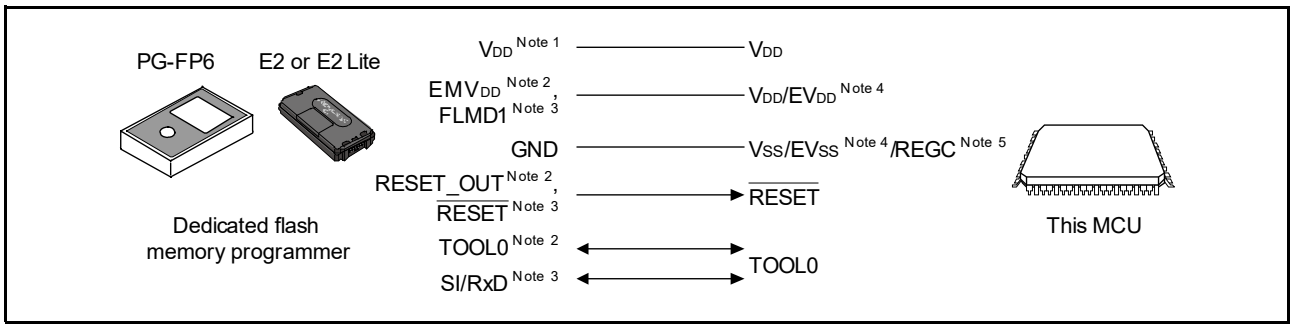
A host machine that controls the dedicated flash memory programmer is necessary. For the interface between the dedicated flash memory programmer and this MCU, the TOOL0 pin of the this MCU is used for writing and erasure via the dedicated single-line UART.

24.1.2 Communications mode

The TOOL0 pin of this MCU is used for communications between the dedicated flash memory programmer and this MCU through serial communications by using the dedicated single-line UART.

Transfer rate: 1 Mbps, 500 kbps, 250 kbps, 115.2 kbps

Figure 24 - 2 Communications with Dedicated Flash Memory Programmer



Note 1. The signal name for the PG-FP6 is VCC.
Note 2. This connection applies when an E2 or E2 Lite on-chip debugging emulator is in use.
Note 3. This connection applies when a PG-FP6 is in use.
Note 4. This pin is only present in the 64-pin products.
Note 5. Connect the REGC pin to the ground via a capacitor (0.47 to 1 µF).

The dedicated flash memory programmer generates the following signals for this MCU. See the manual for the PG-FP6 or, E2 or E2 Lite on-chip debugging emulator for details.

Table 24 - 2 Pin Connection

Dedicated Flash Memory Programmer				This MCU
Signal Name		I/O	Pin Function	Pin Name Note 1
PG-FP6	E2 or E2 Lite On-chip Debugging Emulator			
VCC	VDD	I/O	VDD voltage generation/power monitoring	VDD
GND		—	Ground	VSS, EVSS, REGC Note 2
FLMD1	EMVDD	—	Driving power for the TOOL0 pin	VDD, EVDD
$\overline{\text{RESET}}$	—	Output	Reset signal	$\overline{\text{RESET}}$
—	RESET_OUT	Output		
—	TOOL0	I/O	Transmit/receive signal	TOOL0
SI/RxD	—	I/O	Transmit/receive signal	

Note 1. Pins to be connected differ with the product. For details, see **Table 24 - 1**.

Note 2. Connect the REGC pin to the ground via a capacitor (0.47 to 1 μF).

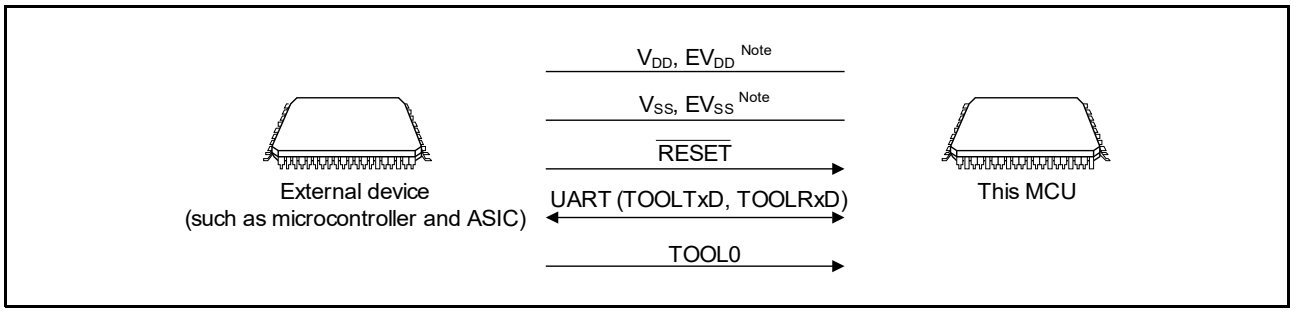
24.2 Serial Programming Using External Device (that Incorporates UART)

On-board data writing to the flash memory of this MCU is possible by using an external device (a microcontroller or ASIC) that is connected to this MCU through a UART. On the development of flash memory programmer by user, refer to the application note, **RL78 Microcontroller (RL78 Protocol C) Serial Programming Guide (R01AN5756)**.

24.2.1 Programming environment

Figure 24 - 3 shows the environment required for writing a program to the flash memory of this MCU.

Figure 24 - 3 Environment for Writing Program to Flash Memory



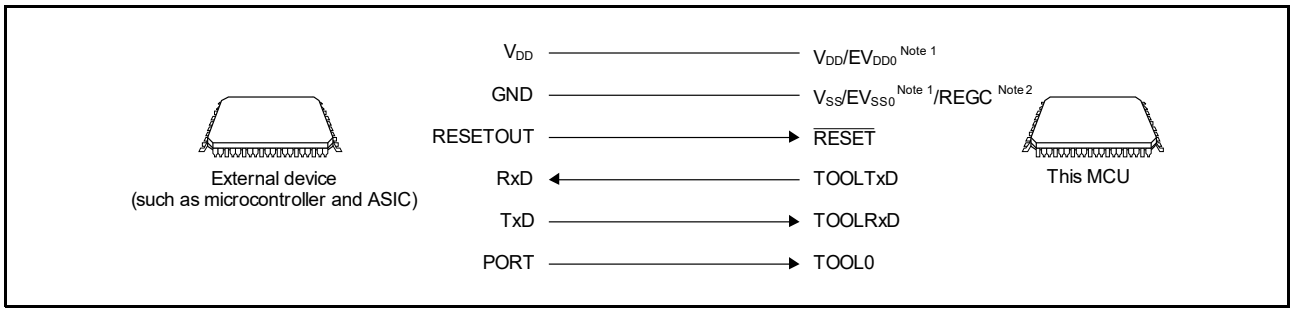
Note This pin is only present in the 64-pin products.

24.2.2 Communications mode

The TOOLTxD and TOOLRxD pins of this MCU are used for communications between the external device and this MCU through serial communications by using the dedicated UART.

Transfer rate: 1 Mbps, 500 kbps, 250 kbps, 115.2 kbps

Figure 24 - 4 Communications with External Device



Note 1. This pin is only present in the 64-pin products.

Note 2. Connect the REGC pin to the ground via a capacitor (0.47 to 1 µF).

The external device generates the following signals for this MCU.

Table 24 - 3 Pin Connection

External Device			This MCU
Signal Name	I/O	Pin Function	Pin Name
VDD	I/O	VDD voltage generation/power monitoring	VDD, EVDD**Note 1**
GND	—	Ground	VSS, REGC**Note 2**, EVSS**Note 1**
RESETOUT	Output	Reset signal output	$\overline{\text{RESET}}$
RxD	Input	Receive signal	TOOLTxD
TxD	Output	Transmit signal	TOOLRxD
PORT	Output	Mode signal	TOOL0

Note 1. This pin is only present in the 64-pin products.

Note 2. Connect the REGC pin to the ground via a capacitor (0.47 to 1 µF).

24.3 Handling of Pins on the Board

To write to the flash memory on board by using a flash memory programmer, mount a connector for the dedicated flash memory programmer on the target system. In addition, include a function for mode switching from normal operation to flash memory programming in the board design. Entry to flash memory programming mode places all pins that are not to be used in programming the flash memory in the same states as those immediately after a reset. If the given states do not suit the operation of external devices, apply appropriate handling to the pins.

Remark For details on flash memory programming mode, refer to **24.4.2 Flash memory programming mode**.

24.3.1 P40/TOOL0 pin

In the flash memory programming mode, connect this pin to the dedicated flash memory programmer via an external 1-kΩ pull-up resistor. When this pin is used as the port pin, use that by the following method.

When used as an input pin: Input of the low level is prohibited for t_{HD} period after external reset release. However, when this pin is used via a pull-down resistor, use a resistor with a value of 500 kΩ or more.
When used as an output pin: When this pin is used via a pull-down resistor, use a resistor with a value of 500 kΩ or more.

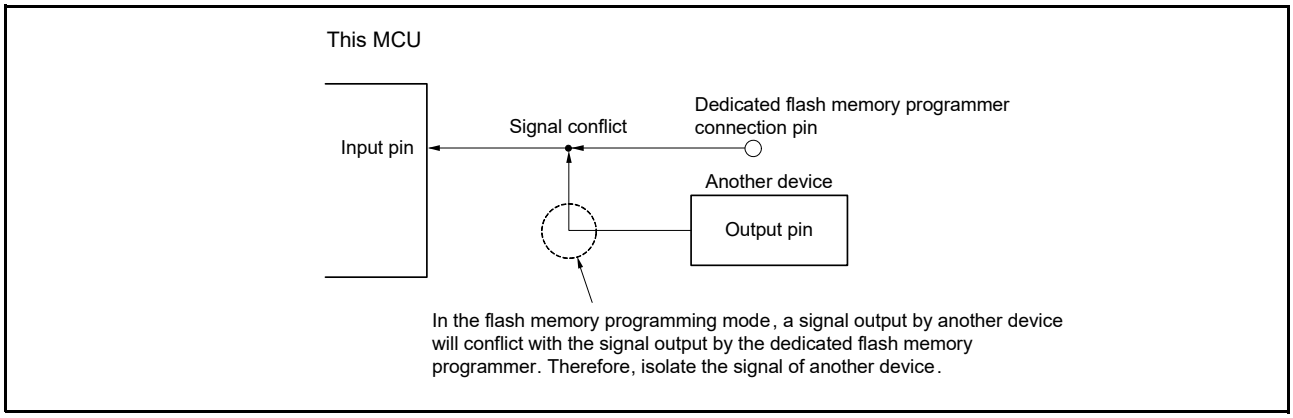
Remark 1. t_{HD}: This is the time over which the TOOL0 pin must be kept at the low level following the end of the external and internal resets for setting of the flash memory programming mode. For details, see **28.10 Timing of Entry to Flash Memory Programming Modes**.

Remark 2. The SAU and IICA pins are not used for communications between this MCU and dedicated flash memory programmer, because the single-line UART (TOOL0 pin) is used.

24.3.2 $\overline{\text{RESET}}$ pin

Signal conflict will occur if the reset signal of the dedicated flash memory programmer and external device is connected to the $\overline{\text{RESET}}$ pin that is connected to the reset signal generator on the board. To prevent this conflict, isolate the connection with the reset signal generator. The flash memory will not be correctly programmed if the reset signal is input from the user system while the flash memory programming mode is set. Do not input any signal other than the reset signal of the dedicated flash memory programmer and external device.

Figure 24 - 5 Signal Conflict ($\overline{\text{RESET}}$ Pin)



24.3.3 Port pins

Entry to flash memory programming mode places all pins that are not to be used in programming the flash memory in the same states as those immediately after a reset. If the given states of the port pins do not suit the operation of external devices that are connected to individual port pins, apply appropriate handling to the pins such as connection to VDD or VSS via resistors.

24.3.4 REGC pin

Connect the REGC pin to GND via a capacitor having excellent characteristics (0.47 to 1 μF) in the same manner as during normal operation. Also, use a capacitor with good characteristics, since it is used to stabilize internal voltage.

24.3.5 Power supply

To use the supply voltage output of the flash memory programmer, connect the VDD pin to VDD[Note] of the flash memory programmer, and the VSS pin to GND of the flash memory programmer. To use the on-board supply voltage, connect the power supply in the same manner as during normal operation. However, when writing to the flash memory by using the flash memory programmer and using the on-board supply voltage, be sure to connect the VDD and VSS pins to VDD[Note] and GND of the flash memory programmer to use the power monitor function by the flash memory programmer.

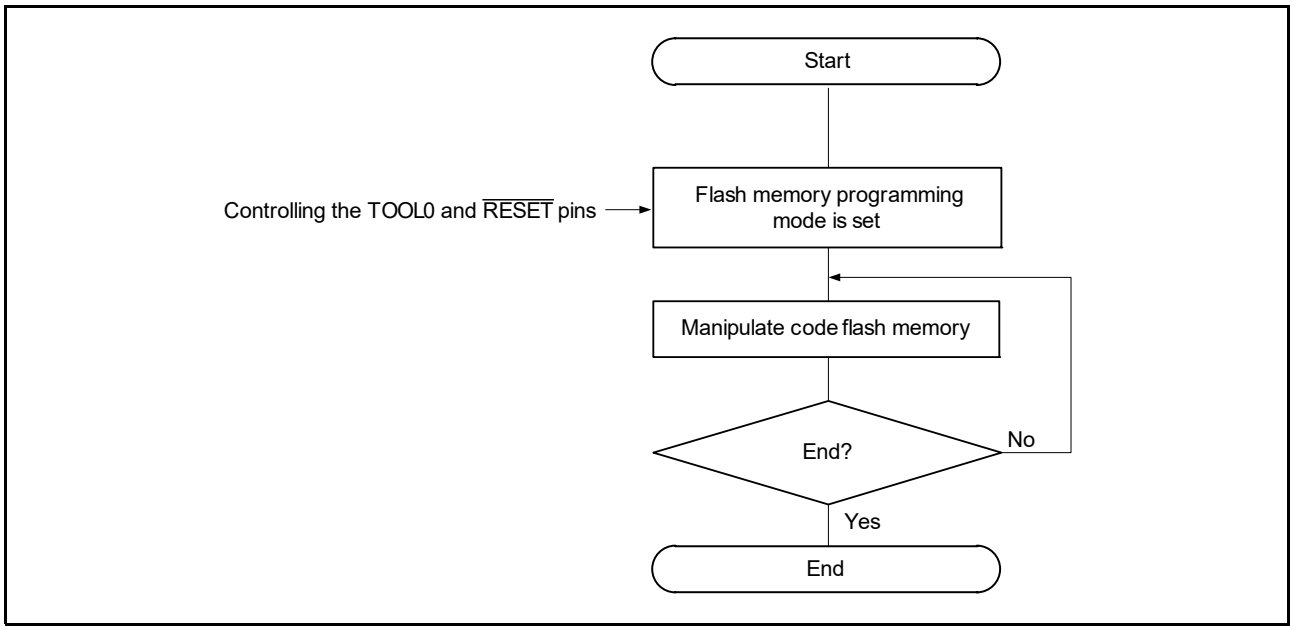
Note The signal name for the PG-FP6 is VCC.

24.4 Programming Method

24.4.1 Serial programming procedure

The following figure illustrates a flow for rewriting the code flash memory through serial programming.

Figure 24 - 6 Code Flash Memory Manipulation Procedure



24.4.2 Flash memory programming mode

To rewrite the contents of the code flash memory through serial programming, specify the flash memory programming mode. The following describes how to enter the flash memory programming mode.

<For serial programming by using the dedicated flash memory programmer>

Connect this MCU to a dedicated flash memory programmer. Starting communications with the dedicated flash memory programmer automatically places the RL78 LSI chip in the flash memory programming mode.

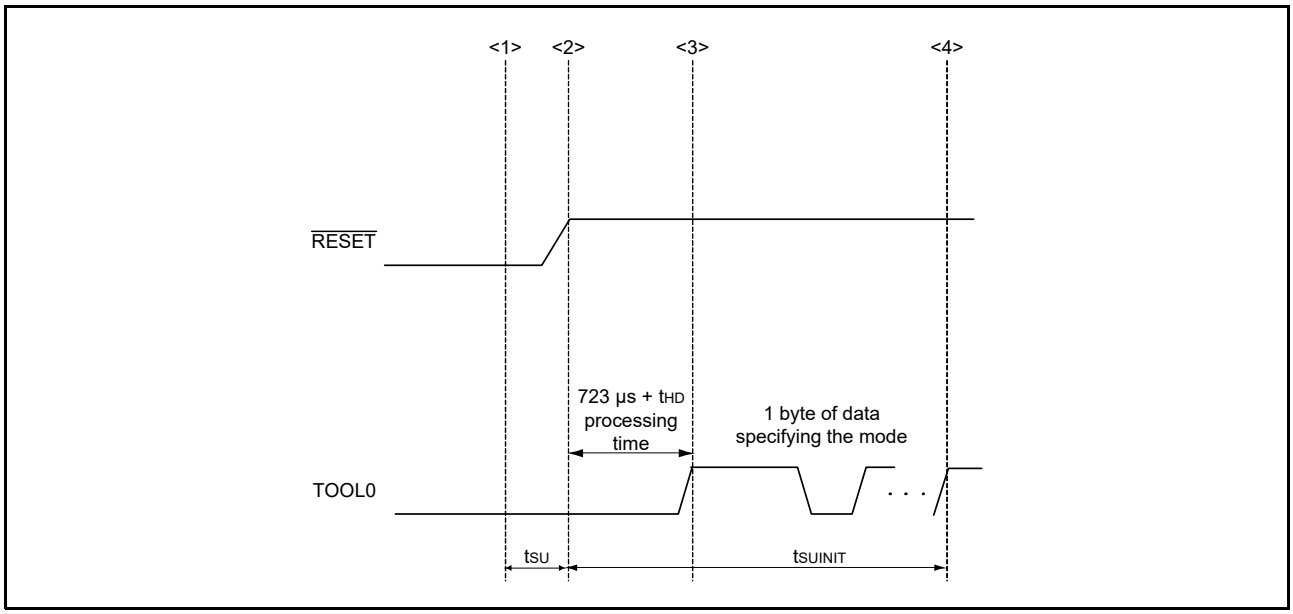
<For serial programming by using an external device (UART communications)>

Set the TOOL0 pin to the low level, and then release the MCU from the reset state (see **Table 24 - 4**). After that, enter flash memory programming mode according to steps <1> to <4> shown in **Figure 24 - 7**. For details, refer to the application note, **RL78 Microcontroller (RL78 Protocol C) Serial Programming Guide (R01AN5756)**.

Table 24 - 4 Relationship between the Voltage Applied to the TOOL0 Pin and the Operating Mode after Release from the Reset State

TOOL0	Operating Mode
EV_{DD}	Normal operation mode
0 V	Flash memory programming mode

Figure 24 - 7 Entry to Flash Memory Programming Mode



<1> The low level is input to the TOOL0 pin.

<2> The external reset is de-asserted (release from a POR or LVD reset must precede this).

<3> The TOOL0 pin is set to the high level.

<4> The bit rate setting is based on the UART reception.

Remark t_{SUINIT} : The section is the up to 100 ms from the end of the external reset within which specifying the initial communications settings must be finished.

t_{SU} : This is the time from the TOOL0 pin being placed at the low level until the end of the external reset.

t_{HD} : This is the time over which the TOOL0 pin must be kept at the low level following the end of an external reset (excluding the processing time of the firmware to control the flash memory).

For details, see **28.10 Timing of Entry to Flash Memory Programming Modes**.

24.4.3 Selecting communications mode

Figure 24 - 5 shows the communications modes of this MCU.

Table 24 - 5 Communications Modes

Communications Mode	Standard Setting[Note 1]				Pins Used
	Port	Speed[Note 2]	Frequency	Multiply Rate	
Single-line UART (when a flash memory programmer is used, or when an external device is used)	UART	115200 bps, 250000 bps, 500000 bps, 1 Mbps	—	—	TOOL0
Dedicated UART (when an external device is used)	UART	115200 bps, 250000 bps, 500000 bps, 1 Mbps	—	—	TOOLTxD, TOOLRxD

Note 1. Selection items for standard settings on GUI of the flash memory programmer.

Note 2. As factors other than bit rate errors, such as dullness of the signal waveform, may also affect UART communications, conduct extensive evaluation with a selected bit rate before attempting to use it.

24.4.4 Communications commands

This MCU executes serial programming through the commands listed in **Table 24 - 6**. The signals sent from the dedicated flash memory programmer or external device to this MCU are called commands, and programming functions corresponding to the commands are executed. For details, refer to the application note, **RL78 Microcontroller (RL78 Protocol C) Serial Programming Guide (R01AN5756)**.

Table 24 - 6 Flash Memory Control Commands

Classification	Command Name	Function
Verification	Verify	Compares the contents of a specified area of the flash memory with data transmitted from the programmer.
Erasure	Block Erase	Erases a specified area in the flash memory.
Blank checking	Block Blank Check	Checks if a specified block in the flash memory has been correctly erased.
Writing	Programming	Writes data to a specified area in the flash memory**Note**.
Getting information	Silicon Signature	Gets the RL78 microcontroller information (such as the device name, flash memory configuration, and programming firmware version).
	Checksum	Gets the checksum data for a specified area.
Security	Security Set	Sets security information.
	Security Get	Gets security information.
	Security Release	Releases the setting to prohibit writing.
Others	Reset	Used to detect the state of synchronization during communications.
	Baud Rate Set	Sets the bit rate when UART communications mode is selected.

Note Confirm that no data have been written to the write area. If data in the area has not been erased, do not attempt further writing of data because data cannot be erased after the setting to prohibit block erasure has been made.

The product information (such as device name and firmware version) can be obtained by executing the Silicon Signature command. **Tables 24 - 7** and **24 - 8** show the signature data list and an example of signature data, respectively.

Table 24 - 7 Signature Data List

Field Name	Description	Number of Bytes of Transmit Data
Device code	The serial number assigned to the device	3 bytes
Device name	Device name (ASCII code)	10 bytes
Code flash memory area end address	End address of the code flash memory area (Sent from the lower-order byte of the address. Example: 00000H to 3FFFFH (256 Kbytes) → FFH, FFH, 03H)	3 bytes
Data flash memory area end address	End address of the data flash memory area (Sent from the lower-order byte of the address. Example: F1000H to F1FFFH (4 Kbytes) → FFH, 1FH, 0FH)	3 bytes
Firmware version	Version information of firmware for programming (Sent from the byte equivalent to the highest-order digit of the version number. Example: Ver. 1.02 → 01H, 00H, 02H)	3 bytes

Table 24 - 8 Example of Signature Data

Field Name	Description	Number of Bytes of Transmit Data	Data (Hexadecimal)
Device code	RL78 protocol C	3 bytes	10 00 0A
Device name	R7F103GLJ	10 bytes	52 = “R” 37 = “7” 46 = “F” 31 = “1” 30 = “0” 33 = “3” 47 = “G” 4C = “L” 4A = “J” 20 = “ ”
Code flash memory area end address	Code flash memory area 00000H to 3FFFFH (256 Kbytes)	3 bytes	FF FF 03
Data flash memory area end address	Data flash memory area F1000H to F1FFFH (4 Kbytes)	3 bytes	FF 1F 0F
Firmware version	Ver. 1.02	3 bytes	01 00 02

24.5 Processing Times for Commands When the Dedicated Flash Memory Programmer Is in Use (Reference Values)

The following shows the processing times for each command (reference value) when the PG-FP6 is used as a dedicated flash memory programmer.

Table 24 - 9 Processing Times for Commands When the PG-FP6 Is in Use (Reference Values)

PG-FP6 Command	Code Flash Memory	
	128 Kbytes	256 Kbytes
Erasure	1.3 s	2.1 s
Writing	2.9 s	5.2 s
Verification	2.2 s	3.9 s
Writing after erasure	4 s	6.9 s

Remark The command processing times (reference values) shown in the table are typical values under the following conditions.
Port: TOOL0 (single-line UART)
Speed: 1,000,000 bps

24.6 Self-programming

This MCU supports a self-programming function that can be used to rewrite the flash memory via a user program. Because this function allows a user application to rewrite the flash memory, it can be used to update the program in the field.

For details, refer to the **RL78 Family Renesas Flash Driver RL78 Type01 User's Manual (R20UT4830EJ)**.

Caution 1. The self-programming function cannot be used when the CPU operates with the subsystem clock (f_{SUB}).

Caution 2. To prohibit an interrupt during self-programming, in the same way as in the normal operation mode, execute self-programming in the state where the IE flag is cleared to 0 by the DI instruction. To enable an interrupt, clear the interrupt mask flag for an interrupt to be accepted to 0 in the state where the IE flag is set to 1 by the EI instruction, and then execute self-programming.

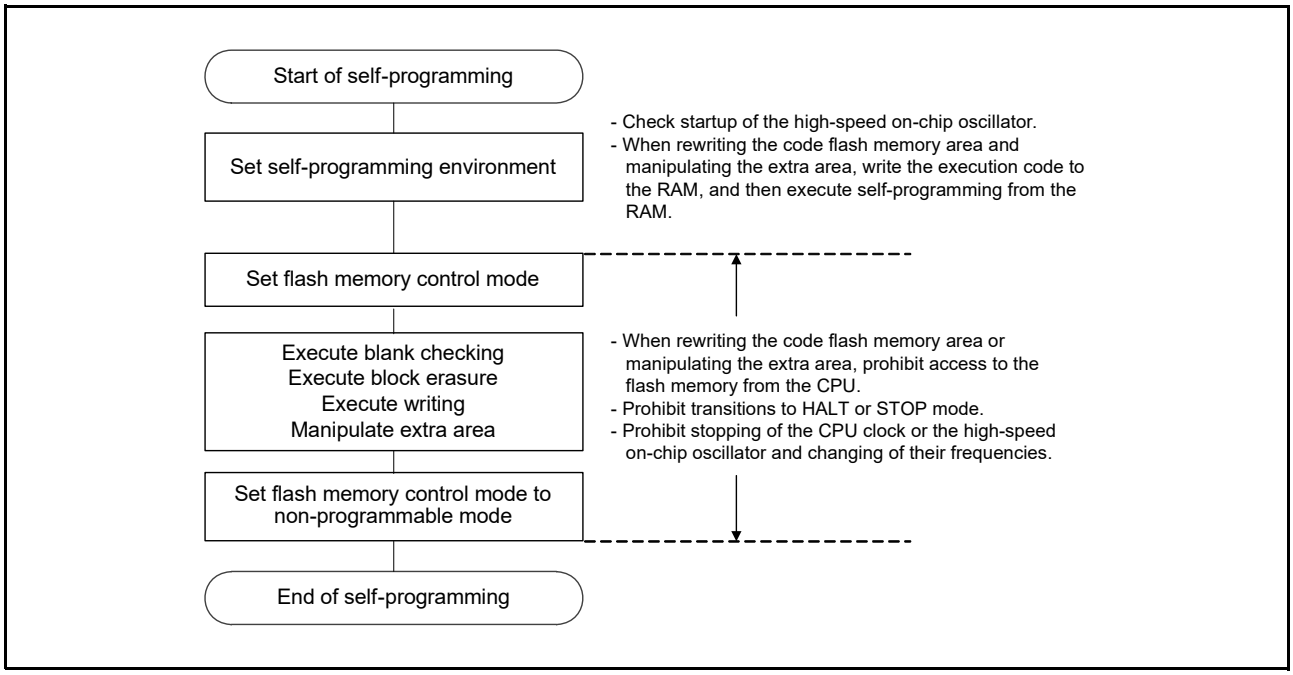
Caution 3. The high-speed on-chip oscillator should be kept operating during self-programming. If it is stopped, be sure to start operation (HIOSTOP = 0) and ensure it is operating with high precision (HIPREC = 1) before executing self-programming.

Caution 4. Do not change the flash operating mode select register (FLMODE) while the flash memory is being rewritten.

24.6.1 Self-programming procedure

The following figure illustrates a flow for rewriting the flash memory by using self-programming. For details on registers for use in self-programming, see **24.6.2 Registers for controlling the flash memory**.

Figure 24 - 8 Flow of Self-programming (Rewriting the Flash Memory)



24.6.2 Registers for controlling the flash memory

The following registers are used to control the flash memory.

- Flash address pointer registers H and L (FLAPH, FLAPL)
- Flash end address pointer registers H and L (FLSEDH, FLSEDL)
- Flash write buffer registers H and L (FLWH, FLWL)
- Flash protect command register (PFCMD)
- Flash status register (PFS)
- Flash programming mode control register (FLPMC)
- Flash area selection register (FLARS)
- Flash memory sequencer initial setting register (FSSET)
- Flash memory sequencer control register (FSSQ)
- Flash extra area sequencer control register (FSSE)
- Flash registers initialization register (FLRST)
- Flash memory sequencer status registers H and L (FSASTH, FSASTL)
- Flash security flag monitoring register (FLSEC)
- Flash FSW monitoring register E (FLFSWE)
- Flash FSW monitoring register S (FLFSWS)
- Data flash control register (DFLCTL)
- Interrupt vector jump enable register (VECTCTRL)
- Interrupt vector change registers 0 and 1 (FLSIVC0, FLSIVC1)

24.6.2.1 Flash address pointer registers H and L (FLAPH, FLAPL)

The FLAPH and FLAPL registers specify the address where programming the flash memory is to start. The FLAPH and FLAPL registers can be set by 8-bit and 16-bit memory manipulation instructions, respectively. The values of the FLAPH and FLAPL registers are 00H and 0000H, respectively, under either of the following conditions.

- Following a reset
- The value of the FLRST bit of the FLRST register is 1.

Figure 24 - 9 Format of Flash Address Pointer Registers H and L (FLAPH, FLAPL)

Address: F02C4H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
FLAPH	0	0	0	0	FLAP19	FLAP18	FLAP17	FLAP16

Address: F02C2H, F02C3H
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
FLAPL	FLAP15	FLAP14	FLAP13	FLAP12	FLAP11	FLAP10	FLAP9	FLAP8

7	6	5	4	3	2	1	0
FLAP7	FLAP6	FLAP5	FLAP4	FLAP3	FLAP2	FLAP1	FLAP0

Caution 1. The FLAPH and FLAPL registers can be rewritten under either of the following conditions.
- **The FLSPM bit in the FLPMC register is 1 (code flash memory area: programming mode).**
- **The EEEMD bit in the FLPMC register is 1 (data flash memory area: programming mode).**

Caution 2. Rewrite or read these registers when the extra area sequencer and the code/data flash memory area sequencer are stopped (SQEND = 0, ESQEND = 0 in the FSASTH register, SQST = 0 in the FSSQ register, and ESQST = 0 in the FSSE register).

24.6.2.2 Flash end address pointer registers H and L (FLSEDH, FLSEDL)

The FLSEDH and FLSEDL registers specify the address where programming of the flash memory is to end. The FLSEDH and FLSEDL registers can be set by 8-bit and 16-bit memory manipulation instructions, respectively. The values of the FLSEDH and FLSEDL registers are 00H and 0000H, respectively, under either of the following conditions.

- Following a reset
- The value of the FLRST bit of the FLRST register is 1.

Figure 24 - 10 Format of Flash End Address Pointer Registers H and L (FLSEDH, FLSEDL)

Address: F02C8H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
FLSEDH	0	0	0	0	EWA19	EWA18	EWA17	EWA16

Address: F02C6H, F02C7H
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
FLSEDL	EWA15	EWA14	EWA13	EWA12	EWA11	EWA10	EWA9	EWA8

7	6	5	4	3	2	1	0
EWA7	EWA6	EWA5	EWA4	EWA3	EWA2	EWA1	EWA0

Caution 1. The FLSEDH and FLSEDL registers can be rewritten under either of the following conditions.
• The FLSPM bit in the FLPMC register is 1 (code flash memory area: programming mode).
• The EEEMD bit in the FLPMC register is 1 (data flash memory area: programming mode).

Caution 2. Rewrite or read these registers when the extra area sequencer and the code/data flash memory area sequencer are stopped (SQEND = 0, ESQEND = 0 in the FSASTH register, SQST = 0 in the FSSQ register, and ESQST = 0 in the FSSE register).

Caution 3. The settings of the EWA1 and EWA0 bits are meaningless during programming of the code flash memory.

Table 24 - 10 Method of Setting the FLAPH, FLAPL, FLSEDH, and FLSEDL Registers

Commands Exclusively for Use with the Code/Data Flash Memory Area Sequencer		Settings of the FLAP and FLSED Registers	
FSSQ	Writing	FLAPH, FLAPL: Address from which writing is to proceed FLSEDH, FLSEDL: All 0s	
	Blank checking	For one word: Settings of FLAPH and FLAPL = settings of FLSEDH and FLSEDL For two or more words: Settings of FLAPH and FLAPL < settings of FLSEDH and FLSEDL	
	Block erasure**Note**	Code flash memory	FLAPH, FLAPL: Set the start address and 0s in the FLAP19 to FLAP11 bits and the FLAP10 to FLAP2 bits, respectively.
			FLSEDH, FLSEDL: Set the end address and 1s in the EWA19 to EWA11 bits and the EWA10 to EWA2 bits, respectively.
		Data flash memory	FLAPH, FLAPL: Set the start address and 0s in the FLAP19 to FLAP8 bits and the FLAP7 to FLAP0 bits, respectively.
			FLSEDH, FLSEDL: Set the end address and 1s in the EWA19 to EWA8 bits and the EWA7 to EWA0 bits, respectively.
FSSE	All commands	Settings of FLAPH and FLAPL: All 0s Settings of FLSEDH and FLSEDL: All 0s	

Note Set the FLAPH, FLAPL, FLSEDH, and FLSEDL registers so that the following condition is met.
Combined settings of FLAPH and FLAPL ≤ combined settings of FLSEDH and FLSEDL

24.6.2.3 Flash write buffer registers H and L (FLWH, FLWL)

The FLWH and FLWL registers hold data to be written during programming of the flash memory. The FLWH and FLWL registers can be set by a 16-bit memory manipulation instruction. The value of each of the FLWH and FLWL registers is 0000H under any of the following conditions.

- Following a reset
- The value of the FLRST bit of the FLRST register is 1.
- The flash memory sequencer has finished operating.

Set data to be written to the data flash memory in the 8 lower-order bits of the FLWL register.

Figure 24 - 11 Format of Flash Write Buffer Registers H and L (FLWH, FLWL)

Address: F02CEH, F02CFH
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
FLWH	FLW31	FLW30	FLW29	FLW28	FLW27	FLW26	FLW25	FLW24

7	6	5	4	3	2	1	0
FLW23	FLW22	FLW21	FLW20	FLW19	FLW18	FLW17	FLW16

Address: F02CCH, F02CDH
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
FLWL	FLW15	FLW14	FLW13	FLW12	FLW11	FLW10	FLW9	FLW8

7	6	5	4	3	2	1	0
FLW7	FLW6	FLW5	FLW4	FLW3	FLW2	FLW1	FLW0

Caution 1. The FLWH and FLWL registers can be rewritten under either of the following conditions.
• The FLSPM bit in the FLPMC register is 1 (code flash memory area: programming mode).
• The EEEMD bit in the FLPMC register is 1 (data flash memory area: programming mode).

Caution 2. Rewrite or read these registers when the extra area sequencer and the code/data flash memory area sequencer are stopped (SQEND = 0, ESQEND = 0 in the FSASTH register, SQST = 0 in the FSSQ register, and ESQST = 0 in the FSSE register).

Caution 3. When writing to the data flash memory, set write data in the 8 lower-order bits of the FLWL register. Set other bits to 0.

24.6.2.4 Flash protect command register (PFCMD)

The PFCMD register protects the flash programming mode control register (FLPMC) against write access. To enable write access to the FLPMC register, write A5H according to the specific sequence. For details on the procedure for handling the specific sequence, see **24.6.3 Setting the flash memory control mode**. The PFCMD register can be set by an 8-bit memory manipulation instruction.

Figure 24 - 12 Format of Flash Protect Command Register (PFCMD)

Address: F00C0H
After reset: Undefined
R/W: W

Symbol	7	6	5	4	3	2	1	0
PFCMD	REG7	REG6	REG5	REG4	REG3	REG2	REG1	REG0

24.6.2.5 Flash status register (PFS)

The PFS register indicates whether or not a protection error has occurred during write access to the flash programming mode control register (FLPMC). For details on the conditions for setting and clearing the FPRERR bit, see **24.6.3.1 Procedure for executing the specific sequence**. The PFS register can be read by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 24 - 13 Format of Flash Status Register (PFS)

Address: F00C1H
After reset: 00H
R/W: R

Symbol	7	6	5	4	3	2	1	0
PFS	0	0	0	0	0	0	0	FPRERR

FPRERR	Protection error flag
0	No error has occurred.
1	An error has occurred.

24.6.2.6 Flash programming mode control register (FLPMC)

The FLPMC register disables or enables writing to the flash memory and selects the programming mode. Enabling write access to the FLPMC register requires the execution of a specific sequence. For details on the procedure for handling the specific sequence, see **24.6.3 Setting the flash memory control mode**. The FLPMC register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 08H.

Figure 24 - 14 Format of Flash Programming Mode Control Register (FLPMC)

Address: F02C0H
After reset: 08H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
FLPMC	0	0	0	EEEMD	FWEDIS	0	FLSPM	0

EEEMD	Selection of the programming mode for the data flash memory
0	Non-programmable mode
1	Programming mode

FWEDIS	Software control over enabling or disabling erasure and programming of the code flash memory[Note]
0	Enables erasure and programming.
1	Disables erasure and programming.

FLSPM	Selection of the programming mode for the code flash memory
0	Non-programmable mode
1	Programming mode

Note Be sure to keep the value of this bit at 0 until erasure or programming of the code flash memory is completed.

Caution **When the extra area sequencer and the code/data flash memory area sequencer are stopped (SQEND = 0, ESQEND = 0 in the FSASTH register, SQST = 0 in the FSSQ register, and ESQST = 0 in the FSSE register), rewriting to the FLPMC register is enabled.**

24.6.2.7 Flash area selection register (FLARS)

The FLARS register selects the area of the flash memory for self-programming. The FLARS register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of the FLARS register is 00H under either of the following conditions.

- Following a reset
- The value of the FLRST bit of the FLRST register is 1.

Figure 24 - 15 Format of Flash Area Selection Register (FLARS)

Address: F02C1H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
FLARS	0	0	0	0	0	0	0	EXA

EXA	Selection of the area of the flash memory for self-programming
0	Code/data flash memory areas
1	Extra area

Caution The FLARS register can be rewritten under either of the following conditions.
• The FLSPM bit in the FLPMC register is 1 (code flash memory area: programming mode).
• The EEEMD bit in the FLPMC register is 1 (data flash memory area: programming mode).

24.6.2.8 Flash memory sequencer initial setting register (FSSET)

The FSSET register sets the operating frequency of the flash memory sequencer and makes the initial settings for boot swapping. The FSSET register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 24 - 16 Format of Flash Memory Sequencer Initial Setting Register (FSSET)

Address: F00B6H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
FSSET	TMSPMD	TMBTSEL	0	FSET4	FSET3	FSET2	FSET1	FSET0

TMSPMD	Selection of boot area setting **Note**
0	Specifies the boot area according to the setting of EX bit 8 (BTFLG) in the security flag and boot swap function setting area of the extra area. BTFLG = 0: Boot cluster 1 as the boot area BTFLG = 1: Boot cluster 0 as the boot area
1	Specifies the boot area according to the setting of the TMBTSEL bit.

TMBTSEL	Specification of the boot area when TMSPMD = 1
0	Specifies boot cluster 0 as the boot area.
1	Specifies boot cluster 1 as the boot area.

FSET4 to FSET0	Setting of the operating frequency of the flash memory sequencer
—	Sets the operating frequency of the flash memory sequencer. For the correspondence between the operating frequency of the flash memory sequencer and the setting of the FSET4 to FSET0 bits, see **Table 24 - 11**.

Note Setting the TMSPMD and TMBTSEL bits is not possible while the BTPR bit in FLSEC is 0 (rewriting of the boot area is disabled).

Caution 1. The FSSET register can be rewritten under either of the following conditions.
- **The FLSPM bit in the FLPMC register is 1 (code flash memory area: programming mode).**
- **The EEEMD bit in the FLPMC register is 1 (data flash memory area: programming mode).**

Caution 2. Setting of the boot area by the BTFLG bit is immediately reflected. To change the boot area after a reset is released, read the MBTSEL bit in FSASTL while TMSPMD = 0 and set the same value to the TMBTSEL bit. After that, set the TMSPMD bit to 1, and then specify the boot cluster to be started as the boot area at release from the reset state in the BTFLG bit using the extra area sequencer. The boot cluster set by the BTFLG bit is activated as the boot area at the next reset release.

Table 24 - 11 Correspondence between the Operating Frequency of the Flash Memory Sequencer and the Setting of the FSET4 to FSET0 Bits

Operating Frequency (MHz)	Setting of the FSET4 to FSET0 Bits	Operating Frequency (MHz)	Setting of the FSET4 to FSET0 Bits	Operating Frequency (MHz)	Setting of the FSET4 to FSET0 Bits
32	11111B	31	11110B	30	11101B
29	11100B	28	11011B	27	11010B
26	11001B	25	11000B	24	10111B
23	10110B	22	10101B	21	10100B
20	10011B	19	10010B	18	10001B
17	10000B	16	01111B	15	01110B
14	01101B	13	01100B	12	01011B
11	01010B	10	01001B	9	01000B
8	00111B	7	00110B	6	00101B
5	00100B	4	00011B	3	00010B
2	00001B	1	00000B	—	—

Caution **Set the value corresponding to that obtained by rounding the CPU operating frequency up to the nearest whole number in the FSET[4:0] bits. For example, when the CPU operating frequency is 4.5 MHz, set the bits to 00100B for 5 MHz. Note that frequencies that are not whole numbers, such as 1.5 MHz, are not available as CPU operating frequencies below 4 MHz.**

24.6.2.9 Flash memory sequencer control register (FSSQ)

The FSSQ register specifies operation control and commands for use with the code/data flash memory area sequencer. When the SQST bit in this register is set to 1, the code/data flash memory area sequencer executes the command set in the MDCH, SQMD2, SQMD1, and SQMD0 bits. The FSSQ register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of the FSSQ register is 00H under either of the following conditions.

- Following a reset
- The value of the FLRST bit of the FLRST register is 1.

Figure 24 - 17 Format of Flash Memory Sequencer Control Register (FSSQ)

Address: F02C5H
After reset: 00H
R/W: R/W

Symbol	<7>	<6>	5	4	3	2	1	0
FSSQ	SQST	FSSTP	0	0	MDCH	SQMD2	SQMD1	SQMD0

SQST	Operation control of the code/data flash memory area sequencer
0	The code/data flash memory area sequencer is stopped.Note 1
1	The code/data flash memory area sequencer is started.

FSSTP	Forcible termination control of the code/data flash memory area sequencer
0	The code/data flash memory area sequencer is not forcibly terminated.
1	The code/data flash memory area sequencer is forcibly terminated.

MDCH	SQMD2	SQMD1	SQMD0	Commands for Use with the Code/Data Flash Memory Area Sequencer
0	0	0	1	• Writing Writes data stored in the FLWH and FLWL registers to the address specified in the FLAPH and FLAPL registers. Note 2 Writes 4-byte data when the code flash memory area address is specified. Writes the 1-byte data stored in the eight lower-order bits (FLW7 to FLW0) in FLWL to the specified address when the data flash memory area address is specified.
0	0	1	1	• Blank checking of the code flash memory area Checks whether the value of the code flash memory area from the address specified in the FLAPH and FLAPL registers to the address specified in the FLSEDH and FLSEDL registers is 1. Note 3
1	0	1	1	• Blank checking of the data flash memory area Checks whether the value of the data flash memory area from the address specified in the FLAPH and FLAPL registers to the address specified in the FLSEDH and FLSEDL registers is 1.
0	1	0	0	• Block erasure Erases blocks in the range from the block start address specified in the FLAPH and FLAPL registers to the block end address specified in the FLSEDH and FLSEDL registers.Note 4
Other than above				Setting prohibited

(Notes and Caution are listed on the next page.)

Note 1. Check that the SQEND bit in the FSASTH register is 1 (the code/data flash memory area sequencer being stopped), and then set the SQST bit to 0 to stop the code/data flash memory area sequencer.

<R> **Note 2.** Four-byte data can be written to the code flash memory area. Accordingly, set the two lower-order bits of the FLAPL register to 00B to be a multiple of 4. For details, see **24.6.6.4 Operations for rewriting the code flash memory area**.

<R> **Note 3.** Specify a start address (at intervals of four bytes) for blank checking of the code flash memory area. Accordingly, set the two lower-order bits of the FLAPL register to 00B to be a multiple of 4. For details, see **24.6.6.4 Operations for rewriting the code flash memory area**.

Note 4. The code flash memory area blocks can be erased in units of 2 Kbytes. The data flash memory blocks can be erased in units of 256 bytes. Specify the erase addresses (start address and end address) so that all blocks to be erased are included. For details, see **24.6.6.4 Operations for rewriting the code flash memory area** and **24.6.6.5 Operations for rewriting the data flash memory area**. For the relationship between the address and block number, see **Table 3 - 1 Correspondence between Addresses and Block Numbers in Flash Memory**.

Caution The FSSQ register can be rewritten under either of the following conditions.

- **The FLSPM bit in the FLPMC register is 1 (code flash memory area: programming mode) and the FWEDIS bit is 0 (enabling erasure and programming of the code flash memory).**
- **The EEEMD bit in the FLPMC register is 1 (data flash memory area: programming mode).**

24.6.2.10 Flash extra area sequencer control register (FSSE)

The FSSE register specifies operation control and commands for use with the extra area sequencer. When the ESQST bit of the FSSE register is set to 1, the extra area sequencer executes the command set by the ESQMD3 to ESQMD0 bits. The FSSE register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of the FSSE register is 00H under either of the following conditions.

- Following a reset
- The value of the FLRST bit of the FLRST register is 1.

Figure 24 - 18 Format of Flash Extra Area Sequencer Control Register (FSSE)

Address: F00B7H
After reset: 00H
R/W: R/W

Symbol	<7>	6	5	4	3	2	1	0
FSSE	ESQST	0	0	0	ESQMD3	ESQMD2	ESQMD1	ESQMD0

ESQST	Operation control of the extra area sequencer
0	The extra area sequencer is stopped.**Note**
1	The extra area sequencer is started.

ESQMD3	ESQMD2	ESQMD1	ESQMD0	Commands for Use with the extra area sequencer
0	0	0	1	• Write to the flash shield window setting area Writes the 4-byte data specified in the FLWH and FLWL registers to the flash shield window setting area of the extra area to control flash shield window mode and set the start block and end block. Also, if the setting of EX bit 15 (FSPR) in the flash shield window setting area is 0, no value can be written to the area. Attempted writing leads to setting of the extra area sequencer error flag (ESEQER) to 1.
0	1	1	0	• Write to the flash read protection setting area Writes the 4-byte data specified in the FLWH and FLWL registers to the flash read protection setting area of the extra area to disable changing of the flash read protection setting and set the start block and end block. Also, if the setting of EX bit 31 (SWPR) in the flash read protection setting area is 0, no value can be written to the area. Attempted writing leads to setting of the extra area sequencer error flag (ESEQER) to 1.
0	1	1	1	• Write to the security flag and boot swap function setting area Writes the 4-byte data specified in the FLWH and FLWL registers to the flash memory security flag and boot swap function setting area of the extra area to disable block erasure, writing, and rewriting of the boot area and set selection of the boot area. Also, if the setting of EX bit 9 (BTPR) in the security flag and boot swap function setting area is 0, no value can be written to the area. Attempted writing leads to setting of the extra area sequencer error flag (ESEQER) to 1.
Other than above				Setting prohibited

(Note and Cautions are listed on the next page.)

Note Check that the ESQEND bit in the FSASTH register is 1 (the extra area sequencer being stopped), and then set the ESQST bit to 0 to stop the extra area sequencer.

Caution 1. The FSSE register can be rewritten when the following conditions are both satisfied.
- **The FLSPM bit in the FLPMC register is 1 (code flash memory area: programming mode)**
- **The FWEDIS bit is 0 (enabling erasure and programming of the code flash memory)**

Caution 2. To write to the extra area, set the EXA bit of the FLARS register to 1 and set the data to be written in the FLWH and FLWL registers before activating the extra area sequencer.

Caution 3. Rewrite the ESQMD3 to ESQMD0 bits while the extra area sequencer and the code/data flash memory area sequencer are stopped (SQEND = 0, ESQEND = 0 in the FSASTH register, SQST = 0 in the FSSQ register, and ESQST = 0 in the FSSE register).

24.6.2.11 Flash registers initialization register (FLRST)

The FLRST register initializes all registers related to the sequencer whether it is to be used with the extra area or code/data flash memory areas. The FLRST register can be set by a 1-bit or 8-bit memory manipulation instruction. For details about how to manipulate the FLRST register, see **24.6.4 Initializing the registers for use with the flash memory sequencer**. The value of this register following a reset is 00H.

Figure 24 - 19 Format of Flash Registers Initialization Register (FLRST)

Address: F02C9H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
FLRST	0	0	0	0	0	0	0	FLRST

FLRST	Control of initializing the registers
0	The registers are not reset.
1	The FLAPH, FLAPL, FLSEDH, FLSEDL, FLWH, FLWL, FLARS, FSSQ, and FSSE registers are reset.

Caution 1. Registers can be initialized by setting the FLRST bit to 1 only when the extra area sequencer and the code/data flash memory area sequencer are stopped (SQEND = 0, ESQEND = 0 in the FSASTH register, SQST = 0 in the FSSQ register, and ESQST = 0 in the FSSE register).

Caution 2. When using the sequencer, be sure to set the FLRST bit to 0 before setting the FLAPH, FLAPL, FLSEDH, FLSEDL, FLWH, FLWL, FLARS, FSSQ, and FSSE registers. Do not set the FLRST bit to 1 during operation of the sequencer.

24.6.2.12 Flash memory sequencer status registers H and L (FSASTH, FSASTL)

The FSASTH and FSASTL registers indicate the results of the respective operations of the flash memory sequencer when it has been used with the extra area or code/data flash memory areas. The FSASTH and FSASTL registers can be read by a 1-bit or 8-bit memory manipulation instruction.

Figure 24 - 20 Format of Flash Memory Sequencer Status Registers H and L (FSASTH, FSASTL) (1/2)

Address: F02CBH
After reset: 00H/04H
R/W: R

Symbol	7	6	5	4	3	2	1	0
FSASTH	ESQEND	SQEND	0	0	0	×	0	0

Address: F02CAH
After reset: 00H/80H Note 1
R/W: R

Symbol	7	6	5	4	3	2	1	0
FSASTL	MBTSEL	MOPEN	ESEQER	SEQER	BLER	0	WRER	ERER

ESQEND	Extra area sequencer operation end status flag
0	The extra area sequencer is operating or stopped by setting the ESQST bit to 0.
1	The extra area sequencer is stopped.

SQEND	Code/data flash memory area sequencer operation end status flag
0	The code/data flash memory area sequencer is operating or stopped by setting the SQST bit to 0.
1	The code/data flash memory area sequencer is stopped.

MBTSEL	Boot flag monitoring bit Note 2
0	BTFLG = 1 (boot area: boot cluster 0)
1	BTFLG = 0 (boot area: boot cluster 1)

MOPEN	Code/data flash memory area sequencer operation status flag
0	The code/data flash memory area sequencer is stopped.
1	The code/data flash memory area sequencer is operating.

ESEQER	Error flag of the extra area sequencer
0	No error has occurred.
1	An error has occurred.
This flag is cleared to 0 when the extra area sequencer is activated.	

Figure 24 - 20 Format of Flash Memory Sequencer Status Registers H and L (FSASTH, FSASTL) (2/2)

SEQER	Error flag of the flash memory sequencer
0	No error has occurred.
1	An error has occurred.
This flag is cleared to 0 when the extra area sequencer or the code/data flash memory area sequencer is activated.	

BLER	Error flag for the blank check command
0	No error has occurred.
1	An error has occurred.
This flag is cleared to 0 when the extra area sequencer or the code/data flash memory area sequencer is activated.	

WRER	Error flag for the write command
0	No error has occurred.
1	An error has occurred.
This flag is cleared to 0 when the extra area sequencer or the code/data flash memory area sequencer is activated. The read value of this bit becomes undefined if the command is forcibly terminated during writing.	

ERER	Error flag for the block erase command
0	No error has occurred.
1	An error has occurred.
This flag is cleared to 0 when the extra area sequencer or the code/data flash memory area sequencer is activated. The read value of this bit becomes undefined if the command is forcibly terminated during block erasure.	

Note 1. The initial value of the MBTSEL bit is undefined since it depends on the value of the BTFLG bit (boot area switching flag) stored in the extra area.

Note 2. This bit indicates the inverse of the value of the BTFLG bit (boot area switching flag) stored in the extra area.

24.6.2.13 Flash security flag monitoring register (FLSEC)

The FLSEC register monitors the information on the security flag and boot swap function settings in the extra area. The FLSEC register can be read by a 16-bit memory manipulation instruction.

Figure 24 - 21 Format of Flash Security Flag Monitoring Register (FLSEC)

Address: F00B0H, F00B1H
After reset: Undefined
R/W: R

Symbol	15	14	13	12	11	10	9	8
FLSEC	0	0	0	WRPR	0	SEPR	BTPR	BTFLG

7	6	5	4	3	2	1	0
0	0	0	0	SWPR	0	IFPR	IDEN

WRPR	Write-disabled flag
0	Writing is disabled.
1	Writing is enabled.

SEPR	Block erase-disabled flag
0	Block erasure is disabled.
1	Block erasure is enabled.

BTPR	Boot area rewrite-disabled flag
0	Rewriting of the boot area is disabled.
1	Rewriting of the boot area is enabled.

BTFLG	Boot area switching flag
0	The boot area is boot cluster 1.
1	The boot area is boot cluster 0.

SWPR	Changing of the flash memory read protection setting disabled flag
0	Changing of the flash memory read protection setting is disabled.
1	Changing of the flash memory read protection setting is enabled.

IFPR	Connection to the programmer and on-chip debugger disabled flag
0	Connection to the programmer and on-chip debugger is disabled.
1	Connection to the programmer and on-chip debugger is enabled.

IDEN	Programmer connection ID authentication enabled flag
0	ID authentication is enabled.
1	ID authentication is disabled.

24.6.2.14 Flash FSW monitoring register E (FLFSWE)

The FLFSWE register monitors the end block number specified for the flash memory shield area and whether the shield area is inside or outside the window range. After a reset or writing to the extra area, the values in the extra area are reflected in the FLFSWE register. For details on the flash shield window function, see **24.8 Flash Shield Window Function**. The FLFSWE register can be read by a 16-bit memory manipulation instruction.

Figure 24 - 22 Format of Flash FSW Monitoring Register E (FLFSWE)

Address: F00B4H, F00B5H
After reset: Undefined
R/W: R

Symbol	15	14	13	12	11	10	9	8
FLFSWE	FSWC	0	0	0	0	0	0	FSWE8

7	6	5	4	3	2	1	0
FSWE7	FSWE6	FSWE5	FSWE4	FSWE3	FSWE2	FSWE1	FSWE0

FSWC	Setting of the flash memory shield area
0	Inside shield mode The flash memory shield area is set inside the window range.
1	Outside shield mode The flash memory shield area is set outside the window range.

FSWE8 to FSWE0	End block number of the flash memory shield area
—	End block number + 1 **Note**

Note These bits show the value set in the extra area. The actual end block number is (the value of the FSWE8 to FSWE0 bits - 1). Though the end block number is specified for serial programming, (end block number + 1) is set in the extra area. For details, see **Table 24 - 12**.

24.6.2.15 Flash FSW monitoring register S (FLFSWS)

The FLFSWS register monitors the start block number of the flash memory shield area and whether rewriting of the flash shield window setting is disabled or enabled. After a reset or writing to the extra area, the values in the extra area are reflected in the FLFSWS register. For details on the flash shield window function, see **24.8 Flash Shield Window Function**. The FLFSWS register can be read by a 16-bit memory manipulation instruction.

Figure 24 - 23 Format of Flash FSW Monitoring Register S (FLFSWS)

Address: F00B2H, F00B3H
After reset: Undefined
R/W: R

Symbol	15	14	13	12	11	10	9	8
FLFSWS	FSPR	0	0	0	0	0	0	FSWS8

7	6	5	4	3	2	1	0
FSWS7	FSWS6	FSWS5	FSWS4	FSWS3	FSWS2	FSWS1	FSWS0

FSPR	Changing of the flash shield window setting disabled flag
0	Changing of the flash shield window setting is disabled.
1	Changing of the flash shield window setting is enabled.

FSWS8 to FSWS0	Start block number of the flash memory shield area
—	Start block number

24.6.2.16 Data flash control register (DFLCTL)

The DFLCTL register enables or disables access to the data flash memory area. The DFLCTL register can be set by a 1-bit or 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 24 - 24 Format of Data Flash Control Register (DFLCTL)

Address: F0090H
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	<0>
DFLCTL	0	0	0	0	0	0	0	DFLEN

DFLEN	Data flash memory area access control
0	Access to the data flash memory area is disabled.
1	Access to the data flash memory area is enabled.

24.6.2.17 Interrupt vector jump enable register (VECTCTRL)

The VECTCTRL register specifies the destination address of the jump in response to any interrupt which occurs during self-programming. The VECTCTRL register can be set by an 8-bit memory manipulation instruction. The value of this register following a reset is 00H.

Figure 24 - 25 Format of Interrupt Vector Jump Enable Register (VECTCTRL)

Address: F00FFH
After reset: 00H
R/W: R/W

Symbol	7	6	5	4	3	2	1	0
VECTCTRL	0	0	0	0	0	0	0	VECTCTRL

VECTCTRL	Setting the interrupt branch destinations
0	Interrupt vector addresses in the ROM
1	Specified addresses in the RAM[Note]

Note A destination address in the RAM is specified by the FLSIVC1 and FLSIVC0 registers. For details, see **24.6.2.18 Interrupt vector change registers 0 and 1 (FLSIVC0, FLSIVC1)**.

24.6.2.18 Interrupt vector change registers 0 and 1 (FLSIVC0, FLSIVC1)

The FLSIVC0 and FLSIVC1 registers specify the destination address of the jump in response to any interrupt which occurs during self-programming. For details on how to handle an interrupt during self-programming, see **24.6.7 Interrupts in code flash memory programming mode**. The FLSIVC0 and FLSIVC1 registers can be set by a 16-bit memory manipulation instruction. The values of the FLSIVC0 and FLSIVC1 registers following a reset are 0000H and 000FH, respectively.

Figure 24 - 26 Format of Interrupt Vector Change Registers 0 and 1 (FLSIVC0, FLSIVC1)

Address: F0480H, F0481H
After reset: 0000H
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
FLSIVC0	FLSIV15	FLSIV14	FLSIV13	FLSIV12	FLSIV11	FLSIV10	FLSIV9	FLSIV8

7	6	5	4	3	2	1	0
FLSIV7	FLSIV6	FLSIV5	FLSIV4	FLSIV3	FLSIV2	FLSIV1	FLSIV0

Address: F0482H, F0483H
After reset: 000FH
R/W: R/W

Symbol	15	14	13	12	11	10	9	8
FLSIVC1	0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
0	0	0	0	FLSIV19	FLSIV18	FLSIV17	FLSIV16

Caution Set the values of the 4 higher-order bits and 16 lower-order bits of the branch destination address in the FLSIVC1 and FLSIVC0 registers, respectively.

24.6.3 Setting the flash memory control mode

The flash memory has the following flash memory control modes.

- Code flash memory programming mode
  The code flash memory area and the extra area can be rewritten.
- Data flash memory programming mode
  The data flash memory area can be rewritten.
- Non-programmable mode
  The flash memory (code flash memory area, data flash memory area, and extra area) cannot be rewritten.

To rewrite the flash memory, set the flash memory control mode to code flash memory programming mode or data flash memory programming mode. Setting each of the flash memory control modes requires executing the specific sequence for setting the flash protect command register (PFCMD) and flash programming mode control register (FLPMC).

Caution For handling of the data flash memory area, follow the procedure while access to the data flash memory is enabled (the value of the DFLEN bit of the DFLCTL register is 1).

24.6.3.1 Procedure for executing the specific sequence

Writing the required values to the flash protect command register (PFCMD) and the flash programming mode control register (FLPMC) by following steps 1 to 4 below enables the transitions to each of the flash memory control modes.

<1> Write A5H to the PFCMD register.
<2> Write the value to be set to the FLPMC register.
<3> Write the inverse of the value to be set to the FLPMC register.
<4> Write the value to be set to the FLPMC register.

- The specific sequence can only be executed while the value of the FLRST bit of the FLRST register is 0 and the flash memory sequencer is stopped.
- If writing to any other memory area or register is attempted in the intervals between steps 1 to 4 during execution of the specific sequence, a protection error occurs, writing to the specified register does not proceed, and the FPRERR flag of the flash status register (PFS) is set to 1. The FPRERR flag is cleared following a reset or when execution of the specific sequence is re-started.

24.6.3.2 Procedure for entry to the code flash memory programming mode

The procedure for entry to the code flash memory programming mode is given below.

<1> Write A5H to the PFCMD register.
<2> Write 02H to the FLPMC register (EEEMD = 0, FWEDIS = 0, FLSPM = 1).
<3> Write FDH to the FLPMC register (inverse of 02H).
<4> Write 02H to the FLPMC register (EEEMD = 0, FWEDIS = 0, FLSPM = 1).

24.6.3.3 Procedure for entry to the data flash memory programming mode

The procedure for entry to the data flash memory programming mode is given below.

<1> Write A5H to the PFCMD register.
<2> Write 10H to the FLPMC register (EEEMD = 1, FWEDIS = 0, FLSPM = 0).
<3> Write EFH to the FLPMC register (inverse of 10H).
<4> Write 10H to the FLPMC register (EEEMD = 1, FWEDIS = 0, FLSPM = 0).

24.6.3.4 Procedure for entry to the non-programmable mode

After executing the procedure for entry to the non-programmable mode from the code flash memory programming mode or data flash memory programming mode and waiting**Note**, reading from the target flash memory for the programming mode before the mode transition becomes possible.

Note Wait time: 10 μs

<When the interrupt vector has not been changed to addresses in the RAM>

The procedure for the transition in cases where branch destinations in response to interrupts are interrupt vector addresses in the ROM is given below.

<1> Write A5H to the PFCMD register.
<2> Write 08H to the FLPMC register (EEEMD = 0, FWEDIS = 1, FLSPM = 0).
<3> Write F7H to the FLPMC register (inverse of 08H).
<4> Write 08H to the FLPMC register (EEEMD = 0, FWEDIS = 1, FLSPM = 0).
<5> Reading from the target flash memory area becomes possible after waiting for 10 μs.

<When the interrupt vectors have been changed to addresses in the RAM>

The procedure for the transition in cases where branch destinations in response to interrupts are changed to specified addresses in the RAM is given below.

<1> Write A5H to the PFCMD register.
<2> Write 00H to the FLPMC register (EEEMD = 0, FWEDIS = 0, FLSPM = 0).
<3> Write FFH to the FLPMC register (inverse of 00H).
<4> Write 00H to the FLPMC register (EEEMD = 0, FWEDIS = 0, FLSPM = 0).
<5> Reading from the target flash memory area becomes possible after waiting for 10 μs.

24.6.4 Initializing the registers for use with the flash memory sequencer

The following registers can be initialized by setting the FLRST bit of the flash registers initialization register (FLRST) to 1.
Target registers: FLAPH, FLAPL, FLSEDH, FLSEDL, FLWH, FLWL, FLARS, FSSQ, FSSE

The procedure for clearing the target registers is given below.

<1> Set the FLRST bit.
<2> Use software code to wait for at least one cycle of the CPU clock.
<3> Clear the FLRST bit.

24.6.5 Setting the operating frequency of the flash memory sequencer

Set the value corresponding to that obtained by rounding the CPU operating frequency (1 MHz to 32 MHz) up to the nearest whole number in the FSET[4:0] bits. For example, when the CPU operating frequency is 4.5 MHz, set the bits to 00100B for 5 MHz.

How to set the operating frequency of the flash memory sequencer is described below.

<1> Enter the code flash memory programming mode or data flash memory programming mode. For the procedures for entry to each of these flash memory programming modes, see **24.6.3.1 Procedure for executing the specific sequence**, **24.6.3.2 Procedure for entry to the code flash memory programming mode**, and **24.6.3.3 Procedure for entry to the data flash memory programming mode**.
<2> After reading from the flash memory sequencer initial setting register (FSSET), set the TMSPMD and TMBTSEL bits to the same values as those read from the FSSET register and the FSET4 to FSET0 bits to the value corresponding to the CPU operating frequency, respectively.

Caution When using the code/data flash memory area sequencer and the extra area sequencer to rewrite the code or data flash memory or the extra area, set the value corresponding to the CPU operating frequency in the FSET4 to FSET0 bits of the FSSET register before doing so. Note that if rewriting of any of these areas is attempted while the value corresponding to the CPU operating frequency is not correct, operation is undefined and written data are not guaranteed. Even if the values in the flash memory are as expected immediately after rewriting, retaining the values for any specified period is not guaranteed.

24.6.6 Rewriting the flash memory

24.6.6.1 Overview

The flash memory sequencer serves as a code/data flash memory area sequencer or an extra area sequencer. In the former role, it is used to rewrite the code flash memory area or data flash memory area, while in the latter role it is used to rewrite the extra area. To rewrite a given area, execute the corresponding commands for use with the sequencer.

24.6.6.2 Selecting the area to be rewritten

Select the code flash memory area, data flash memory area, or extra area for rewriting by using the flash area selection register (FLARS).

24.6.6.3 Commands for use with the code/data flash memory area sequencer

Use the commands exclusively used for the code/data flash memory area sequencer to rewrite the code or data flash memory area. To execute a command, set the target command in the SQMD2 to SQMD0 bits of the flash memory sequencer control register (FSSQ) and set the SQST bit to 1. The SQMD2 to SQMD0 bits and the SQST bit can be set simultaneously. For the commands exclusively for use with the code/data flash memory area sequencer, see **24.6.2.9 Flash memory sequencer control register (FSSQ)**.

24.6.6.4 Operations for rewriting the code flash memory area

To rewrite the code flash memory area, execute commands for use with the code/data flash memory area sequencer. Before starting to execute a command, set the data required for execution, such as specifying an address, addresses, or data, in the corresponding registers. Allocate the processing software to rewrite the code flash memory area in the RAM and execute it from the RAM.

Units for block erasure and for writing in rewriting of the code flash memory area

- Unit for blocks to be erased: 2 Kbytes
- Unit for writing: 4 bytes

<Handling the commands>

<1> Enter the code flash memory programming mode. For the procedure for entry to the code flash memory programming mode, see **24.6.3.1 Procedure for executing the specific sequence** and **24.6.3.2 Procedure for entry to the code flash memory programming mode**.

<2> Set the EXA bit of the FLARS register to 0 (code/data flash memory areas).

<3> Before executing each command, set the address data, write data, and command in the corresponding registers.

- Block erasure
  Set the block start address[Note 1] (example: 002000H) of the code flash memory to be erased in the FLAPH and FLAPL registers. Set the block end address[Note 1] (example: 0027FFH) of the code flash memory to be erased in the FLSEDH and FLSEDL registers.
- Writing
  Set the start address[Note 2] (example: 002000H) of the flash memory to be written in the FLAPH and FLAPL registers. Set 4-byte write data in the FLWH and FLWL registers.
- Blank checking
  Set the start address[Note 2] (example: 002000H) of the flash memory to be blank-checked in the FLAPH and FLAPL registers. Set the end address (example: 0027FFH) of the flash memory to be blank-checked in the FLSEDH and FLSEDL registers. When blank checking is only to be applied to one word (four bytes), set the FLSEDH and FLSEDL registers to the same values as those in the FLAPH and FLAPL registers.

<4> When the value of the command to be executed is set in the MDCH and SQMD2 to SQMD0 bits of the FSSQ register and the SQST bit is set to 1, the code/data flash memory area sequencer executes the specified command. The MDCH, SQST, and SQMD2 to SQMD0 bits can be set simultaneously. If these bits are set simultaneously, the FSSQ register is set to the following values.

- Block erasure: 84H
- Writing: 81H
- Blank checking of the code flash memory area: 83H

<5> Wait until the command for use with the code/data flash memory area sequencer is complete. For details on the procedure for waiting for the completion of command execution, see the section titled “Procedure for checking the completion of commands for use with the code/data flash memory area sequencer” in **24.6.6.9 Procedures for checking completion of the commands for use with the flash memory sequencer in the respective areas**.

<6> Processing after executing a command

<Continuing command processing>

The same command or another command can be executed by continuously updating the address data and write data in step 3 with the state remaining in code flash memory programming mode.

<Completing command processing>

Switch to the non-programmable mode. For the procedure for switching to the non-programmable mode, see **24.6.3.1 Procedure for executing the specific sequence** and **24.6.3.4 Procedure for entry to the non-programmable mode**.

(Notes are listed on the next page.)

Note 1. The code flash memory area blocks can be erased in units of 2 Kbytes. Specify the erase addresses (start address and end address) so that all blocks to be erased are included. For the relationship between addresses and block numbers, see **Table 3 - 1 Correspondence between Addresses and Block Numbers in Flash Memory**.

Note 2. The code flash memory area can be written and blank-checked in units of 4 bytes. Therefore, set the two lower-order bits of the FLAPL register for specifying the address to 00B (an integer of 4).

24.6.6.5 Operations for rewriting the data flash memory area

To rewrite the data flash memory area, execute commands for use with the code/data flash memory area sequencer. Before starting to execute a command, set the data required for execution, such as specifying an address, addresses, or data, in the corresponding registers.

Units for block erasure and for writing in rewriting of the data flash memory area

- Unit for blocks to be erased: 256 bytes
- Unit for writing: 1 byte

<Handling the commands>

<1> Enter the data flash memory programming mode. For the procedure for entry to the data flash memory programming mode, see **24.6.3.1 Procedure for executing the specific sequence** and **24.6.3.3 Procedure for entry to the data flash memory programming mode**.

<2> Set the EXA bit of the FLARS register to 0 (code/data flash memory areas).

<3> Before executing each command, set the address data, write data, and command in the corresponding registers.

- Block erasure
  Set the block start address[Note] (example: 0F1100H) of the data flash memory to be erased in the FLAPH and FLAPL registers. Set the block end address[Note] (example: 0F11FFH) of the data flash memory to be erased in the FLSEDH and FLSEDL registers.
- Writing
  Set the start address (example: 0F1101H) of the flash memory to be written in the FLAPH and FLAPL registers. Set the write data in the 8 lower-order bits of the FLWL register.
- Blank checking:
  Set the start address (example: 0F1100H) of the flash memory to be blank-checked in the FLAPH and FLAPL registers. Set the end address (example: 0F11FFH) of the flash memory to be blank-checked in the FLSEDH and FLSEDL registers. When blank checking is only to be applied to one byte, set the FLSEDH and FLSEDL registers to the same values as those in the FLAPH and FLAPL registers.

<4> When the value of the command to be executed is set in the MDCH and SQMD2 to SQMD0 bits of the FSSQ register and the SQST bit is set to 1, the code/data flash memory area sequencer executes the specified command. The MDCH, SQST, and SQMD2 to SQMD0 bits can be set simultaneously. If these bits are set simultaneously, the FSSQ register is set to the following values.

- Block erasure: 84H
- Writing: 81H
- Blank checking of the data flash memory area: 8BH

<5> Wait until the command for use with the code/data flash memory area sequencer is complete. For details on the procedure for waiting for the completion of command execution, see the section titled “Procedure for checking the completion of commands for use with the code/data flash memory area sequencer” in **24.6.6.9 Procedures for checking completion of the commands for use with the flash memory sequencer in the respective areas**.

<6> Processing after executing a command

<Continuing command processing>

The same command or another command can be executed by continuously updating the address data and write data in step 3 with the state remaining in data flash memory programming mode.

<Completing command processing>

Switch to the non-programmable mode. For the procedure for switching to the non-programmable mode, see **24.6.3.1 Procedure for executing the specific sequence** and **24.6.3.4 Procedure for entry to the non-programmable mode**.

(Note is listed on the next page.)

Note The data flash memory area blocks can be erased in units of 256 bytes. Therefore, set the 8 lower-order bits of the FLAPL register for specifying the start address to 0000 0000B (an integer of 256). Also set the 8 lower-order bits of the FLSEDL register for specifying the end address to 1111 1111B.

24.6.6.6 Commands for use with the extra area sequencer

Use the commands for use with the extra area sequencer to rewrite the settings of the flash shield window, flash read protection, flash security, and boot swap function in the extra area. To execute a command, set the command to be executed in the ESQMD3 to ESQMD0 bits of the flash extra area sequencer control register (FSSE), and then set the ESQST bit to 1. The ESQMD3 to ESQMD0 bits and the ESQST bit can be set simultaneously. Allocate the extra area sequencer command processing software in the RAM and execute it from the RAM.

24.6.6.7 Operations for rewriting the extra area

To rewrite the extra area, enter the code flash memory programming mode and then execute commands for use with the extra area sequencer. Before starting to execute a command, set the data required for executing each command in the corresponding registers.

<Handling the commands>

<1> Enter the code flash memory programming mode. For the procedure for entry to the code flash memory programming mode, see **24.6.3.1 Procedure for executing the specific sequence** and **24.6.3.2 Procedure for entry to the code flash memory programming mode**.

<2> Set the EXA bit of the FLARS register to 1 (extra area).

<3> Before executing a command, set 4-byte data in the FLWH and FLWL registers. Each bit of the combined FLW31 to FLW0 bits of the FLWH and FLWL registers corresponds to EX bits 31 to 0 of the target extra area data. For details on data to be set for each command, see **24.6.6.8 Data to be set for the commands for use with the extra area sequencer**.

<4> When the value of the command to be executed is set in the ESQMD3 to ESQMD0 bits of the FSSE register and the ESQST bit is set to 1, the extra area sequencer executes the specified command. The ESQMD3 to ESQMD0 bits and the ESQST bit can be set simultaneously. If these bits are set simultaneously, the FSSE register is set to the following values.

- Write data to the flash shield window setting area: 81H
- Write data to the flash read protection setting area: 86H
- Write data to the security flag and boot swap function setting area: 87H

<5> Wait until the command for use with the extra area sequencer is complete. For details on the procedure for waiting for the completion of command execution, see the section titled “Procedure for checking the completion of commands for use with the extra area sequencer” in **24.6.6.9 Procedures for checking completion of the commands for use with the flash memory sequencer in the respective areas**.

<6> Processing after executing a command

<Continuing command processing>

The same command or another command can be executed by continuously updating the FLWH and FLWL register data to be set in the extra area in step 3 with the state remaining in code flash memory programming mode.

<Completing command processing>

Switch to the non-programmable mode. For the procedure for switching to the non-programmable mode, see **24.6.3.1 Procedure for executing the specific sequence** and **24.6.3.4 Procedure for entry to the non-programmable mode**.

24.6.6.8 Data to be set for the commands for use with the extra area sequencer

Writing to the extra area proceeds per 4 bytes. Each command for use with the extra area sequencer writes the data set in the combined FLW31 to FLW0 bits of the FLWH and FLWL registers to EX bits 31 to 0 in the extra area corresponding to the given command.

1. Write to the flash shield window setting area
   Set the data in the FLWH and FLWL registers to the flash shield window setting area.

EX bit 31	EX bit 30	EX bit 29	EX bit 28	EX bit 27	EX bit 26	EX bit 25	EX bit 24
FSWC	1	1	1	1	1	1	FSWE8

EX bit 23	EX bit 22	EX bit 21	EX bit 20	EX bit 19	EX bit 18	EX bit 17	EX bit 16
FSWE7	FSWE6	FSWE5	FSWE4	FSWE3	FSWE2	FSWE1	FSWE0

EX bit 15	EX bit 14	EX bit 13	EX bit 12	EX bit 11	EX bit 10	EX bit 9	EX bit 8
FSPR	1	1	1	1	1	1	FSWS8

EX bit 7	EX bit 6	EX bit 5	EX bit 4	EX bit 3	EX bit 2	EX bit 1	EX bit 0
FSWS7	FSWS6	FSWS5	FSWS4	FSWS3	FSWS2	FSWS1	FSWS0

Bit Name	Setting
FSWC	Specifies the range of the flash memory shield area. 0: Flash memory shield area: Inside the window range 1: Flash memory shield area: Outside the window range (default)
FSPR	Specifies whether to enable or disable changing of the flash shield window setting area. 0: Changing of the flash shield window setting area is disabled. 1: Changing of the flash shield window setting area is enabled (default).
FSWE8 to FSWE0	Flash shield window end block setting area Specify the block number (end block number + 1).**Note**
FSWS8 to FSWS0	Flash shield window start block setting area Specify the start block number.**Note**

Note For the relationship between addresses and block numbers, see **Table 3 - 1 Correspondence between Addresses and Block Numbers in Flash Memory**.

Caution **The value of the FSPR bit can be changed from 0 (disabling) to 1 (enabling) by executing the chip erase command of the dedicated flash memory programmer or the Security Release command for use in serial programming mode with the entire flash memory erased. Note that if either of the disabling settings listed below is made, executing the chip erase command of the dedicated flash memory programmer or the Security Release command for use in serial programming mode is not possible.**

- **SEPR = 0 (block erasure is disabled)**
- **BTPR = 0 (rewriting of the boot area is disabled)**

Moreover, setting the FSPR bit to 1 (enabling) is not possible because a command cannot be transmitted when connection in serial programming mode is not available due to the setting for disabling connection to the programmer and on-chip debugger or for enabling programmer connection ID authentication having been made.

2. Write to the flash read protection setting area
   Set the data in the FLWH and FLWL registers to the flash read protection setting area.

EX bit 31	EX bit 30	EX bit 29	EX bit 28	EX bit 27	EX bit 26	EX bit 25	EX bit 24
SWPR	1	1	1	1	1	1	UPAddr8

EX bit 23	EX bit 22	EX bit 21	EX bit 20	EX bit 19	EX bit 18	EX bit 17	EX bit 16
UPAddr7	UPAddr6	UPAddr5	UPAddr4	UPAddr3	UPAddr2	UPAddr1	UPAddr0

EX bit 15	EX bit 14	EX bit 13	EX bit 12	EX bit 11	EX bit 10	EX bit 9	EX bit 8
1	1	1	1	1	1	1	LOWAddr8

EX bit 7	EX bit 6	EX bit 5	EX bit 4	EX bit 3	EX bit 2	EX bit 1	EX bit 0
LOWAddr7	LOWAddr6	LOWAddr5	LOWAddr4	LOWAddr3	LOWAddr2	LOWAddr1	LOWAddr0

Bit Name	Setting
SWPR	Specifies whether to enable or disable changing of the flash read protection setting area. 0: Changing of the flash read protection setting area is disabled. 1: Changing of the flash read protection setting area is enabled (default).
UPAddr8 to UPAddr0	Flash read protection end block setting area Specify the end block number.**Note**
LOWAddr8 to LOWAddr0	Flash read protection start block setting area Specify the start block number.**Note**

Note For the relationship between addresses and block numbers, see **Table 3 - 1 Correspondence between Addresses and Block Numbers in Flash Memory**. The flash read protection setting area cannot be read after a reset is released.

Caution **The value of the SWPR bit can be changed from 0 (disabling) to 1 (enabling) by executing the chip erase command of the dedicated flash memory programmer or the Security Release command for use in serial programming mode with the entire flash memory erased. Note that if either of the disabling settings listed below is made, executing the chip erase command of the dedicated flash memory programmer or the Security Release command for use in serial programming mode is not possible.**
- **SEPR = 0 (block erasure is disabled)**
- **BTPR = 0 (rewriting of the boot area is disabled)**

Moreover, setting the SWPR bit to 1 (enabling) is not possible because a command cannot be transmitted when connection in serial programming mode is not available due to the setting for disabling connection to the programmer and on-chip debugger or for enabling programmer connection ID authentication having been made.

3. Write to the security flag and boot swap function setting area
   Set the data in the FLWH and FLWL registers to the security flag and boot swap function setting area. For details of the security settings, see **24.9 Security Settings**.

EX bit 31	EX bit 30	EX bit 29	EX bit 28	EX bit 27	EX bit 26	EX bit 25	EX bit 24
1	1	1	1	1	1	1	1

EX bit 23	EX bit 22	EX bit 21	EX bit 20	EX bit 19	EX bit 18	EX bit 17	EX bit 16
1	1	1	1	1	1	1	1

EX bit 15	EX bit 14	EX bit 13	EX bit 12	EX bit 11	EX bit 10	EX bit 9	EX bit 8
1	1	1	WRPR	1	SEPR	BTPR	BTFLG

EX bit 7	EX bit 6	EX bit 5	EX bit 4	EX bit 3	EX bit 2	EX bit 1	EX bit 0
1	1	1	1	1	IFPR	1	IDEN

Bit Name	Setting
WRPR	Specifies whether to enable or disable writing in serial programming mode. Writing in serial programming mode is disabled. 0: Writing in serial programming mode is disabled. 1: Writing in serial programming mode is enabled (default).
SEPR	Specifies whether to enable or disable block erasure. Block erasure in serial programming mode is disabled. 0: Block erasure in serial programming mode is disabled. 1: Block erasure in serial programming mode is enabled (default).
BTPR	Specifies whether to enable or disable rewriting of the boot area. Rewriting of the boot area and boot swapping are disabled. 0: Rewriting of the boot area and boot swapping are disabled. 1: Rewriting of the boot area and boot swapping are enabled (default).
BTFLG	Specifies the boot area when the TMSPMD bit in the FSSET register is 0. 0: Boot area: Boot cluster 1 1: Boot area: Boot cluster 0 (default)
IFPR	Specifies whether to enable or disable connection to the programmer and on-chip debugger. The serial programming mode and connection to the on-chip debugger are disabled. 0: Serial programming mode and connection to the on-chip debugger are disabled. 1: Serial programming mode and connection to the on-chip debugger are enabled (default).
IDEN	Specifies whether to enable or disable programmer connection ID authentication. ID authentication for connection in serial programming mode proceeds. 0: ID authentication for connection in serial programming mode is enabled. 1: ID authentication for connection in serial programming mode is disabled (default).

(Cautions are listed on the next page.)

Caution 1. When changing the value of the BTFLG bit, set all other bits to 1.

Caution 2. When changing the values of security flags other than BTFLG to 0 (disabling), read the register first and set the BTFLG bit to the same value as was read, and set the other bits to 1.

Caution 3. The value of the WRPR bit can be changed from 0 (disabling) to 1 (enabling) by executing the chip erase command of the dedicated flash memory programmer or the Security Release command for use in serial programming mode with the entire flash memory erased. Note that if either of the disabling settings listed below is made, executing the chip erase command of the dedicated flash memory programmer or the Security Release command for use in serial programming mode is not possible.

- **SEPR = 0 (block erasure is disabled)**
- **BTPR = 0 (rewriting of the boot area is disabled)**

Moreover, setting the WRPR bit to 1 (enabling) is not possible because a command cannot be transmitted when connection in serial programming mode is not available due to the setting for disabling connection to the programmer and on-chip debugger or for enabling programmer connection ID authentication having been made.

Caution 4. Restoring the value of any among the SEPR, BTPR, IFPR, and IDEN bits to 1 after having set it to 0 is not possible.

24.6.6.9 Procedures for checking completion of the commands for use with the flash memory sequencer in the respective areas

Perform the following procedure for checking the completion of commands when terminating an activated command for use with the code/data flash memory area sequencer and the extra area sequencer.

- Procedure for checking the completion of commands for use with the code/data flash memory area sequencer
  - <1> Activate a command for use with the code/data flash memory area sequencer, and then wait until the SQEND flag of the FSASTH register is set to 1.
  - <2> Confirm that the SQEND flag of the FSASTH register is set to 1, and then clear the SQST bit of the FSSQ register.
  - <3> Wait until the SQEND flag of the FSASTH register is cleared. When the SQEND flag is cleared, the command is completed and the sequencer stops.

- Procedure for checking the completion of commands for use with the extra area sequencer
  - <1> Activate a command for use with the extra area sequencer, and then wait until the ESQEND flag of the FSASTH register is set to 1.
  - <2> Confirm that the ESQEND flag of the FSASTH register is set to 1, and then clear the ESQST bit of the FSSE register.
  - <3> Wait until the ESQEND flag of the FSASTH register is cleared. When the ESQEND flag is cleared, the command is completed and the sequencer stops.

24.6.6.10 Procedure for forcibly terminating a command for use with the code/data flash memory area sequencer

A command for use with the code/data flash memory area sequencer can be forcibly terminated while it is in progress. Note that no command for use with the extra area sequencer can be forcibly terminated while it is in progress.

<Procedure for forcible termination>

<1> Set the FSSTP bit of the FSSQ register to 1 after a command for use with the code/data flash memory area sequencer is activated until the SQST bit of the FSSQ register in <2> is cleared.

<2> Confirm that the SQEND flag of the FSASTH register is set to 1, and then clear the SQST and FSSTP bits of the FSSQ register.

<3> Wait until the SQEND flag of the FSASTH register is automatically cleared. When the SQEND flag is cleared, the forcible termination is completed.

24.6.7 Interrupts in code flash memory programming mode

24.6.7.1 Overview

When an interrupt occurs, reference to the interrupt vector in the ROM proceeds, and execution branches to the interrupt processing that is allocated in the up to 64-Kbyte ROM space in accord with the address in the interrupt vector (16 bits), from which the interrupt processing is executed. However, in code flash memory programming mode, where the code flash memory and extra area can be rewritten, referring to the ROM is not possible. As a result, the interrupt processing cannot be executed in this way. Note that even when referring to the ROM is not possible, the interrupt processing can be executed by changing the branch destinations of the interrupts. Specifically, changing the branch destinations of all interrupts to specified addresses in the RAM enables the execution of interrupt processing from the RAM instead of by using the interrupt vector and interrupt processing in the ROM.

24.6.7.2 Operation to change the branch destinations of the interrupts

To change the branch destinations of the interrupts, set the interrupt vector change registers 1 and 0 (FLSIVC1 and FLSIVC0) and interrupt vector jump enable register (VECTCTRL) so that execution branches to an address in the RAM following an interrupt. This enables execution of the interrupt processing from the RAM without reference to the interrupt vector in the ROM, even when an interrupt occurs in code flash memory programming mode. The FLSIVC1 and FLSIVC0 registers are used to specify the branch destination address in response to interrupts which occur during rewriting of the code flash memory or extra area. Set the values of the 16 lower-order bits and 4 higher-order bits of the branch destination address in the FLSIVC0 and FLSIVC1 registers, respectively.

Settings for control of the branch destination of interrupts that occur during self-programming are as follows.

- For branching to vector addresses in the ROM: VECTCTRL = 0 or FWEDIS of FLPMC = 1
- For branching to a RAM address: VECTCTRL = 1 (with FWEDIS of FLPMC = 0)

Caution 1. The user must determine the type of interrupt by checking the interrupt flags. Therefore, the interrupt flags are not automatically cleared after the VECTCTRL register has been set.

Caution 2. A changed interrupt branch destination in the ROM is not specifiable.

Caution 3. The change to the interrupt branch destination made by the VECTCTRL register is only effective during self-programming.

Caution 4. Disable interrupts while changing the interrupt branch destination to an address in the RAM.

24.6.7.3 Operation to change the interrupt branch destination

To specify interrupt processing from the RAM, update the FLSIVC1, FLSIVC0, and VECTCTRL registers while the value of the FWEDIS bit of the flash programming mode control register (FLPMC) is 0. Execute the specific sequence to handle the FWEDIS bit of the FLPMC register, and set the FLSIVC1, FLSIVC0, and VECTCTRL registers. The interrupt branch destination is thus changed to an address in RAM.

<When changing the interrupt branch destinations to an address in RAM >

The following describes operation for changing all interrupt branch destinations to a specified address in the RAM.

<1> Save the setting for enabling or disabling interrupts that was in place before starting this procedure and make the setting to disable interrupts.

<2> Execute the specific sequence and set the FWEDIS bit of the FLPMC register to 0.

1. Write A5H to the PFCMD register.
2. Write 00H to the FLPMC register (EEEMD = 0, FWEDIS = 0, FLSPM = 0).
3. Write FFH to the FLPMC register (inverse of 00H).
4. Write 00H to the FLPMC register (EEEMD = 0, FWEDIS = 0, FLSPM = 0).

<3> Specify an address in RAM in the FLSIVC1 and FLSIVC0 registers.

<4> Set the VECTCTRL register to 01H to set the RAM address as the branch destination for interrupts.

<5> Restore the saved setting for enabling or disabling interrupts.

Caution 1. Retain the value 0 in the FWEDIS bit as long as interrupt processing from the RAM remains specified.

Caution 2. Do not set the interrupt branch destination in the saddr space at addresses from FFE20H to FFEFFH.

Caution 3. When instructions are being executed from the RAM area and RAM parity error resets are enabled (RPERDIS = 0), initialize the RAM area where data access is to proceed + 10 bytes.

<When returning the interrupt branch destinations from the RAM address to the vectors in the ROM>

The following describes operation for returning the interrupt branch destinations to the addresses indicated by the interrupt vectors in the ROM (default state).

<1> Save the setting for enabling or disabling interrupts that was in place before starting this procedure and make the setting to disable interrupts.

<2> Execute the specific sequence and set the FWEDIS bit of the FLPMC register to 1.

1. Write A5H to the PFCMD register.
2. Write 08H to the FLPMC register (EEEMD = 0, FWEDIS = 1, FLSPM = 0).
3. Write F7H to the FLPMC register (inverse of 08H).
4. Write 08H to the FLPMC register (EEEMD = 0, FWEDIS = 1, FLSPM = 0).

<3> Set the VECTCTRL register to 00H to set the vector addresses in the ROM as the interrupt branch destinations.

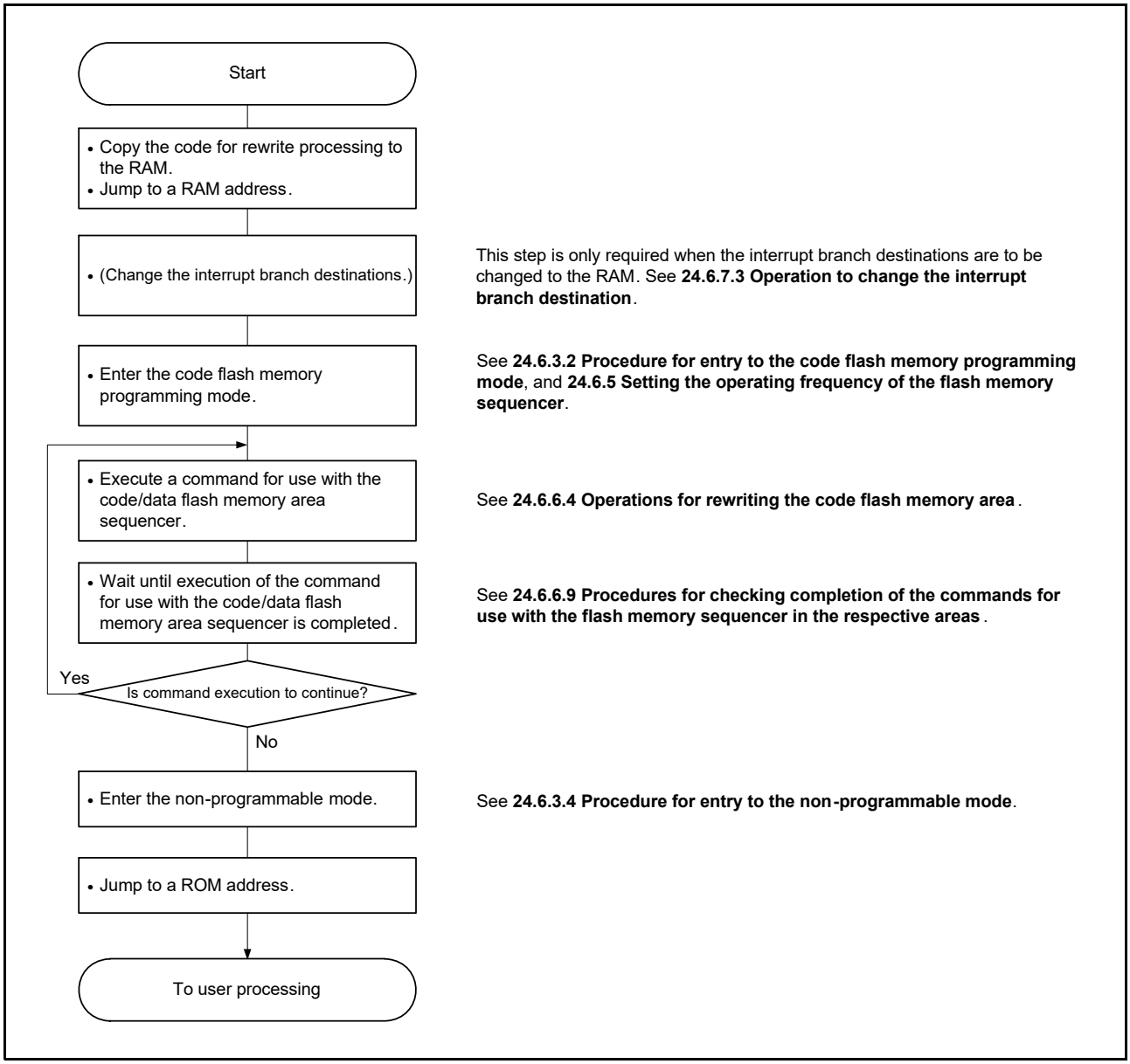
<4> Restore the saved setting for enabling or disabling interrupts.

24.6.8 Example of executing the commands to rewrite the flash memory areas

24.6.8.1 Example of executing the commands to rewrite the code flash memory area

Figure 24 - 27 shows the flow of executing the commands to rewrite the code flash memory area.

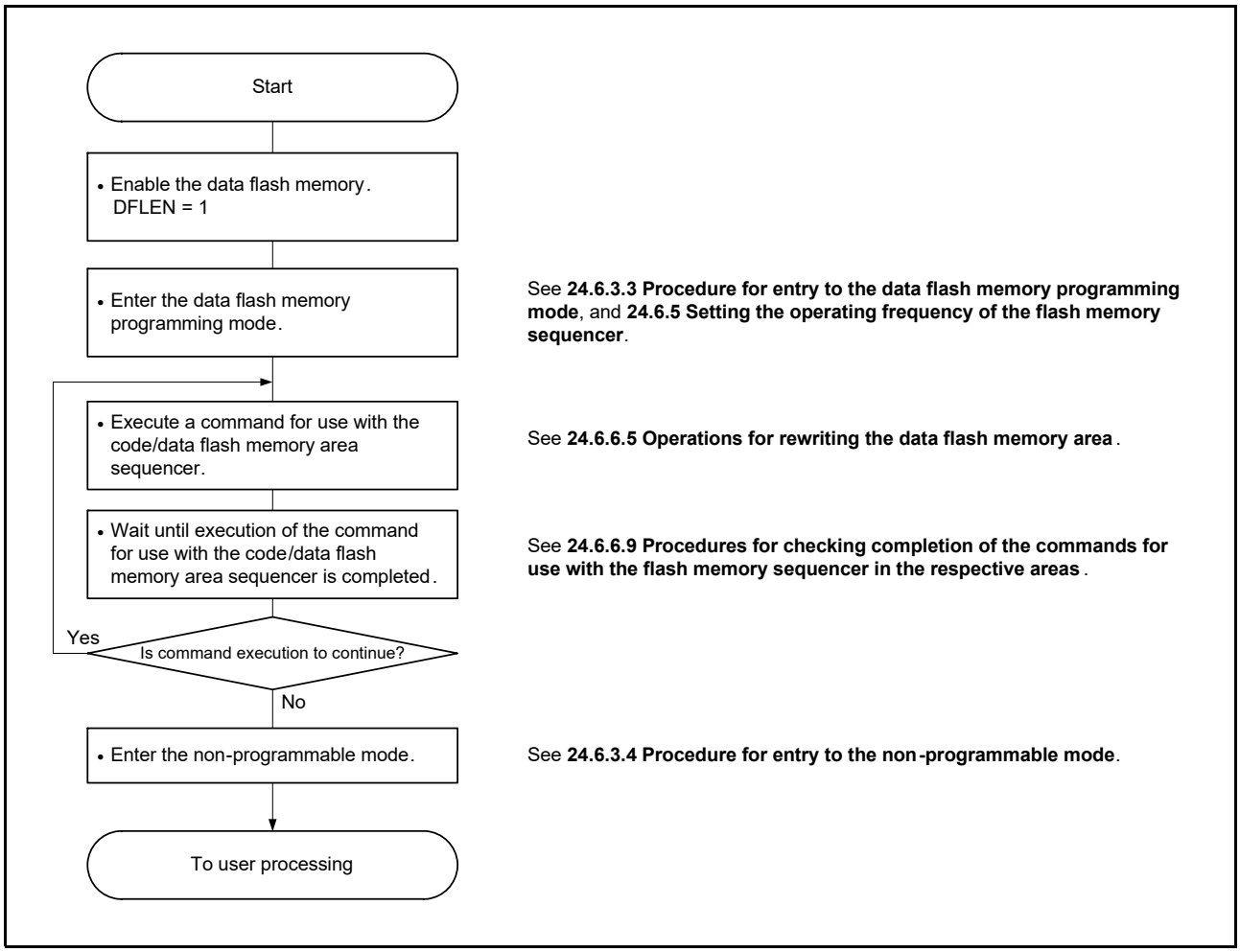
Figure 24 - 27 Flow of Executing the Commands to Rewrite the Code Flash Memory Area



24.6.8.2 Example of executing the commands to rewrite the data flash memory area

Figure 24 - 28 shows the flow of executing the commands to rewrite the data flash memory area.

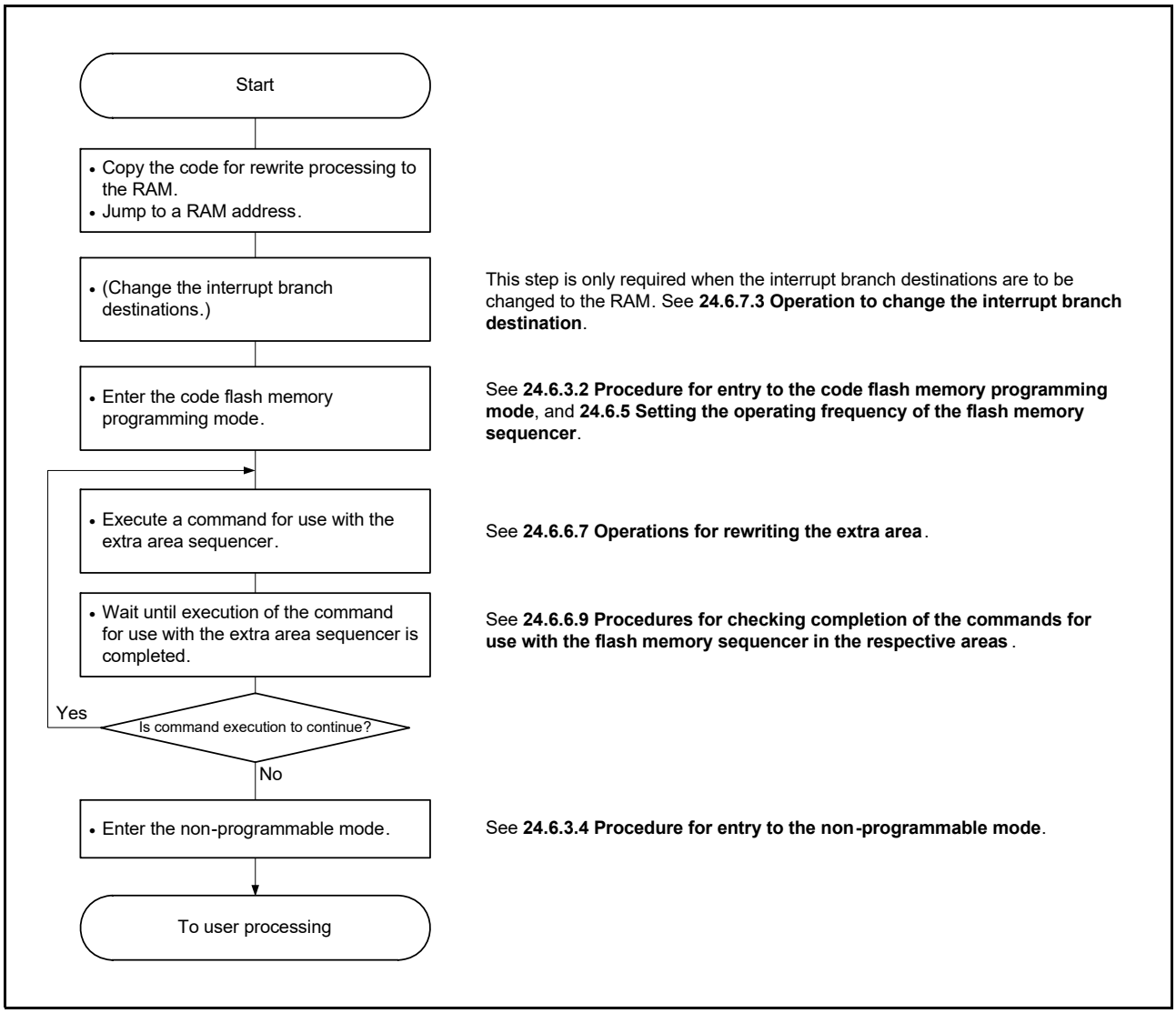
Figure 24 - 28 Flow of Executing the Commands to Rewrite the Data Flash Memory Area



24.6.8.3 Example of executing the commands to rewrite the extra area

Figure 24 - 29 shows the flow of executing the commands to rewrite the extra area.

Figure 24 - 29 Flow of Executing the Commands to Rewrite the Extra Area



24.6.9 Notes on self-programming

1. Rewriting the code flash memory or extra area
   To rewrite the code flash memory or extra area, place the code or values in the RAM.

2. Precondition for manipulating the data flash memory area
   Before manipulating the data flash memory area, set the DFLEN bit of the data flash control register (DFLCTL) to 1 (enabling access to the data flash memory).

3. Execution of programs during rewriting of the flash memory
   The flash memory sequencer is used to control rewriting of the flash memory during self-programming. In the flash memory control modes where rewriting of the flash memory is enabled, reference to the flash memory to be manipulated is not possible.
   - In code flash memory programming mode, reference to the code flash memory is not possible. Accordingly, in code flash memory programming mode, copy the user program that is to be executed from the ROM (code flash memory) and its data for reference to the RAM in advance so that the program can be executed and reference to the data in the RAM is possible.
   - In data flash memory programming mode, reference to the data flash memory is not possible. Accordingly, in data flash memory programming mode, copy data that are for reference to the RAM in advance so that reference to the data in the RAM is possible.

4. Specifying the range of unavailable area
   Specify the range of blank checking and block erasure within the range of code flash memory area or data flash memory area. Do not specify any unavailable area or both the code flash memory area and data flash memory area including an unavailable area.

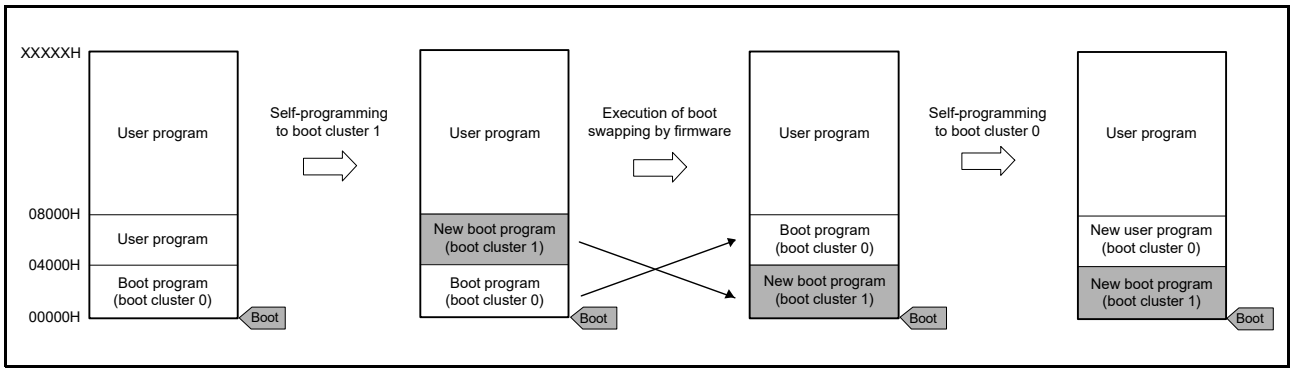
24.7 Boot Swap Function

The boot area consists of the vector table area, CALLT table area, option bytes area, setting areas for the on-chip debug security ID and programmer connection ID, and program area. The required settings and information respectively for starting the program and for connecting an on-chip debugger and programmer are stored in these areas. Thus, if rewriting in the boot area through self-programming failed by temporary power failure or other reasons, restarting a program by resetting or overwriting is disabled due to data destruction in the boot area.
The boot swap function is used to avoid this problem. When the boot area is set to boot cluster 0, write a new boot program to boot cluster 1 through self-programming before erasing boot cluster 0**Note**. After the boot program has been successfully written to boot cluster 1, change the boot area from boot cluster 0 to boot cluster 1 by self-programming to use boot cluster 1 as the boot area. After that, erase the data in boot cluster 0, and then write data to boot cluster 0. As a result, even if a power failure occurs while the area is being rewritten, the program is executed correctly because it is booted from boot cluster 1 when the program is reset and started next. A boot cluster is a 16-Kbyte area.

Note Allocate the addresses in the new boot program as follows.

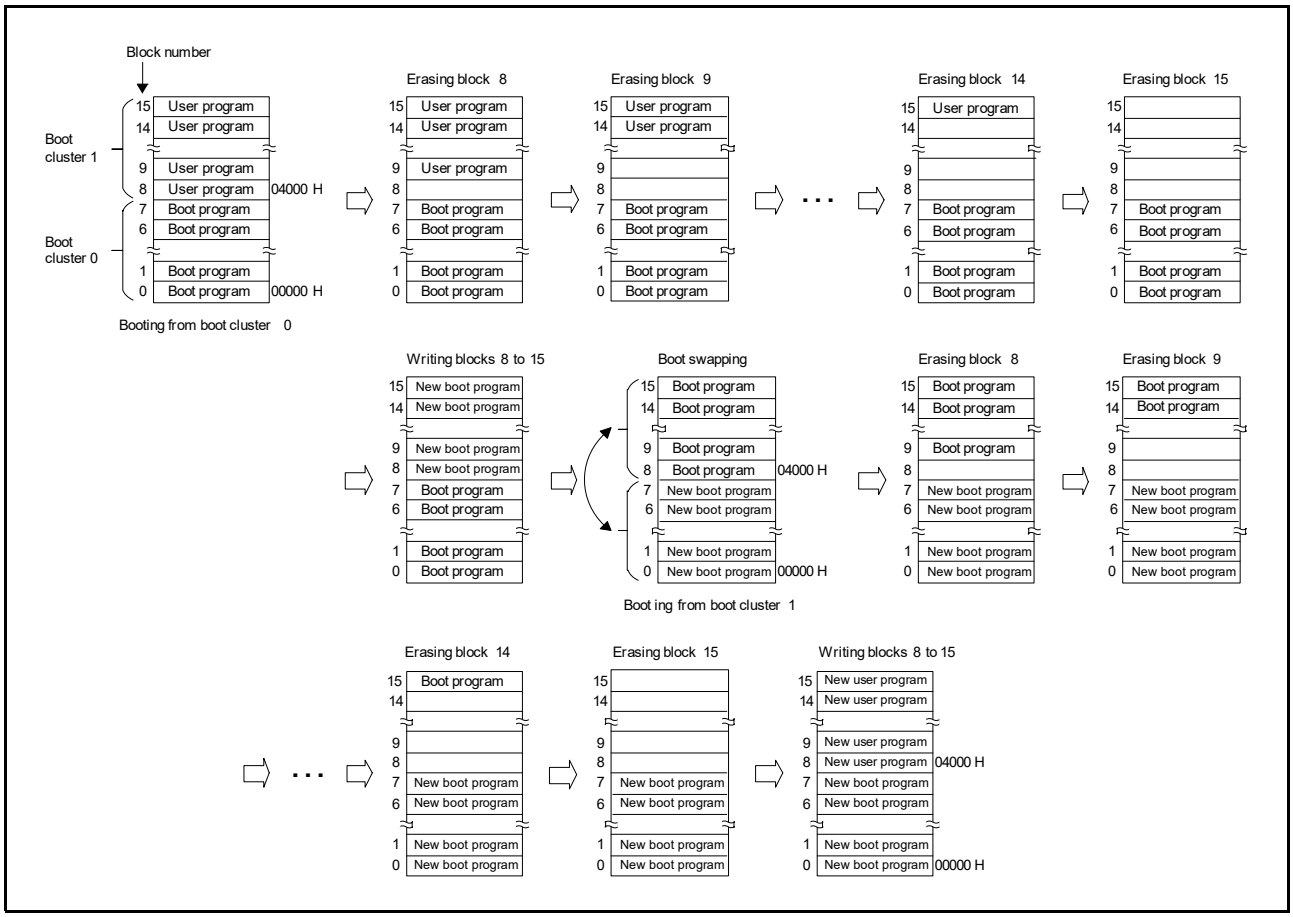
- 04000H to 0407FH (128 bytes): Vector table area
- 04080H to 040BFH (64 bytes): CALLT table area
- 040C0H to 040C3H (4 bytes): Option bytes area
- 040C4H to 040CDH (10 bytes): On-chip debug security ID setting area

Figure 24 - 30 Boot Swap Function



In the example of the above figure, the boot areas before and after boot swapping are as follows.
Boot cluster 0: Boot area before boot swapping
Boot cluster 1: Boot area after boot swapping

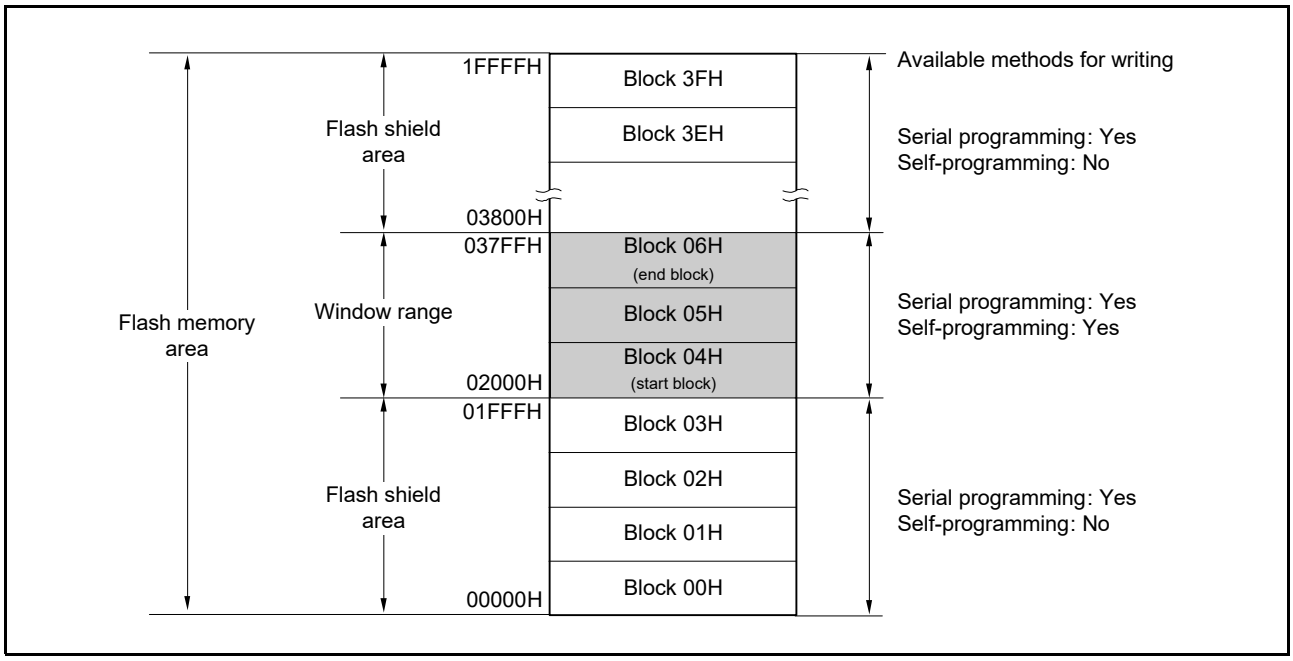
Figure 24 - 31 Example of Executing Boot Swapping



24.8 Flash Shield Window Function

The flash shield window function is provided as a security function which disables writing to and erasing of the selected flash memory shield area. This function is only effective for self-programming. The flash memory shield area is selectable as either the area inside or areas outside the range specified as the window. The window range is set by specifying the blocks where it starts and ends. The flash memory shield area can be set or changed during both serial programming and self-programming. Writing to and erasing of the flash memory shield area are disabled during self-programming. During serial programming, however, the flash memory shield area can also be written and erased.

Figure 24 - 32 Flash Shield Window Setting Example
(Target Devices: R7F103GLJ, Start Block Number: 04H, End Block Number: 06H, FSWC: 1)



Caution 1. If the non-programmable area of the boot area overlaps with the flash shield window range, disabling of rewriting the boot area takes priority.

Caution 2. The flash shield window can only be used for the code flash memory (and is not supported for the data flash memory).

Table 24 - 12 Relationship between Flash Shield Window Function Setting/Change Methods and Commands

Programming Condition	Window Range Setting/Change Method	Command to be Executed	
		Block erase	Write
Self-programming	Specify the window start block number and the (end block number + 1) block number (following the end block) in the flash shield window setting area using the self-programming.	Block erasure is not possible inside the flash memory shield area.	Writing is not possible inside the flash memory shield area.
Serial programming	Specify the start block number in the window range and the end block number in the window range on GUI of dedicated flash memory programmer, etc.	Block erasure is also possible inside the flash memory shield area.	Writing is also possible inside the flash memory shield area.

24.9 Security Settings

This MCU supports a security function that prohibits rewriting the user program written to the internal flash memory, so that the program cannot be changed by an unauthorized person. The operations shown below can be performed by serial programming or self-programming.

- Disabling block erasure
  Execution of the block erase command for a specific block in the code flash memory and data flash memory is prohibited during serial programming. However, blocks can be erased by means of self-programming.
- Disabling writing
  Execution of the write command for entire blocks in the code flash memory and data flash memory is prohibited during serial programming. However, blocks can be written by means of self-programming. After the setting to prohibit writing has been made, releasing the setting by the Security Release command is enabled by a reset.
- Disabling rewriting the boot area
  Execution of the block erase command and write command for the boot area (00000H to 03FFFH) in the code flash memory is prohibited.
- Disabling connection to the programmer and on-chip debugger
  Connection to a dedicated flash memory programmer and on-chip debugger is prohibited. A dedicated flash memory programmer and on-chip debugger cannot be used to manipulate the flash memory.
- Enabling programmer connection ID authentication
  Authentication for an arbitrary 10-byte ID code is enabled when connecting to a dedicated flash memory programmer. The 10-byte ID area is 000C4H to 000CDH[Note]. If the ID does not match when using serial programming, the dedicated flash memory programmer cannot be used to manipulate the flash memory.

Block erasure, writing, and rewriting the boot area are enabled by the default setting at the time of shipment. Security can be set by serial programming and self-programming. Each security setting can be used in combination. **Table 24 - 13** shows the relationship between the erase and write commands when the security function of this MCU is enabled.

Note The 10-byte ID code area for the programmer connection ID is shared with the security ID code for on-chip debugging.

Caution **The security function of the dedicated flash programmer does not support self-programming.**

Remark To prohibit writing and erasure during self-programming, use the flash shield window function. For details, see **24.8 Flash Shield Window Function**.

Table 24 - 13 Relationship between Enabling the Security Function and Commands

1. During serial programming

Valid Security	Command to be Executed	
	Block Erase	Write
Prohibition of block erasure	Blocks cannot be erased.	Data can be written.Note
Prohibition of writing	Blocks can be erased.	Data cannot be written.
Prohibition of rewriting the boot area	The boot area cannot be erased	The boot area cannot be written.
Prohibition of connection to the programmer and on-chip debugger	Blocks cannot be erased.	Data cannot be written.
Success in authentication with programmer connection ID authentication enabled	Blocks can be erased.	Data can be written.
Failure in authentication with programmer connection ID authentication enabled	Blocks cannot be erased.	Data cannot be written.

Note Confirm that no data have been written to the write area. If data in the area has not been erased, do not attempt further writing of data because data cannot be erased after the setting to prohibit block erasure has been made.

2. During self-programming

Valid Security	Command to be Executed	
	Block Erase	Write
Prohibition of block erasure	Blocks can be erased.	Data can be written.
Prohibition of writing		
Prohibition of rewriting the boot area	The boot area cannot be erased.	The boot area cannot be written.
Prohibition of connection to the programmer and on-chip debugger	Blocks can be erased.	Data can be written.
Programmer connection ID authentication enabled	Blocks can be erased.	Data can be written.

Remark To prohibit writing and erasure during self-programming, use the flash shield window function. For details, see **24.8 Flash Shield Window Function**.

Table 24 - 14 Setting Security in Each Programming Mode

1. During serial programming

Security	Security Setting	Disabling the Security Setting
Prohibition of block erasure	Set via GUI of dedicated flash memory programmer, etc.	Disabling the setting is not possible.
Prohibition of writing		Set via GUI of dedicated flash memory programmer, etc.
Prohibition of rewriting the boot area		Disabling the setting is not possible.
Prohibition of connection to the programmer and on-chip debugger		
Enabling programmer connection ID authentication		

Caution **The setting to prohibit writing can only be released when the settings to prohibit erasing blocks and rewriting the boot area are not made and the code and data flash memory areas are blank. However, if connection for serial programming is prohibited due to the setting to prohibit connection to the programmer and on-chip debugger or to enable programmer connection ID authentication, releasing the setting to prohibit writing is not possible because serial programming cannot be executed.**

2. During self-programming

Security	Security Setting	Disabling the Security Setting
Prohibition of block erasure	Set by using self-programming.	Disabling the setting is not possible.
Prohibition of writing		Disabling the setting is not possible during self-programming. Set via GUI of dedicated flash memory programmer, etc.
Prohibition of rewriting the boot area		Disabling the setting is not possible.
Prohibition of connection to the programmer and on-chip debugger		
Enabling programmer connection ID authentication		

24.10 Data Flash Memory

24.10.1 Overview of the data flash memory

An overview of the data flash memory is provided below.

- The user program can rewrite the data flash memory by self-programming. For details, refer to the **RL78 Family Renesas Flash Driver RL78 Type 01 User's Manual (R20UT4830EJ)**.
- The data flash memory can also be rewritten to through serial programming using a dedicated flash memory programmer or an external device.
- The data flash memory can be erased in 1-block (256-byte) units.
- The data flash memory can be accessed only in 8-bit units.
- The data flash memory can be directly read by CPU instructions.
- Instructions can be executed from the code flash memory while rewriting the data flash memory (that is, background operation (BGO) is supported).
- The data flash memory area is exclusively to be used for data, so executing instructions from the data flash memory is prohibited.
- Accessing the data flash memory is prohibited while rewriting the code flash memory (during self-programming).
- Manipulating the DFLCTL register is prohibited while rewriting the data flash memory.
- Transition to the HALT or STOP mode is prohibited while rewriting the data flash memory.

Caution 1. The data flash memory is stopped after release from a reset. The data flash control register (DFLCTL) must be set up in order to use the data flash memory.

Caution 2. The high-speed on-chip oscillator should be kept operating during rewriting of the data flash memory area. If it is stopped, be sure to start operation (HIOSTOP = 0) and ensure it is operating with high precision (HIPREC = 1) before executing self-programming.

Remark For rewriting the code flash memory via a user program, see **24.6 Self-programming**.

24.10.2 Procedure for accessing the data flash memory

The data flash memory is stopped after release from a reset. For access to this memory, follow the procedure below to make the required initial setting.

<1> Set the DFLEN bit of the data flash control register (DFLCTL) to 1.
<2> Wait for the setup time by using a software timer, etc.
The setup time depends on the operating mode of the flash memory in terms of the main clock selection.
<Setup time for each operating mode of the flash memory>
- HS (high-speed main) mode: 250 ns
- LS (low-speed main) mode: 250 ns
- LP (low-power main) mode: 0 ns

<3> After the wait, the data flash memory can be accessed.

Caution 1. Accessing the data flash memory is prohibited during the setup time.
Caution 2. Transition to the STOP mode is prohibited during the setup time. To enter the STOP mode during the setup time, clear DFLEN to 0 and then execute the STOP instruction.
Caution 3. The high-speed on-chip oscillator should be kept operating during rewriting of the data flash memory area. If it is stopped, be sure to start operation (CSC.HIOSTOP = 0) and ensure it is operating with high precision (HIPREC = 1) before executing self-programming.
Caution 4. When switching the CPU clock between the main system clock and sub system clock, accessing the data flash memory is prohibited until the CLS bit changes after a clock is selected by the CSS bit.

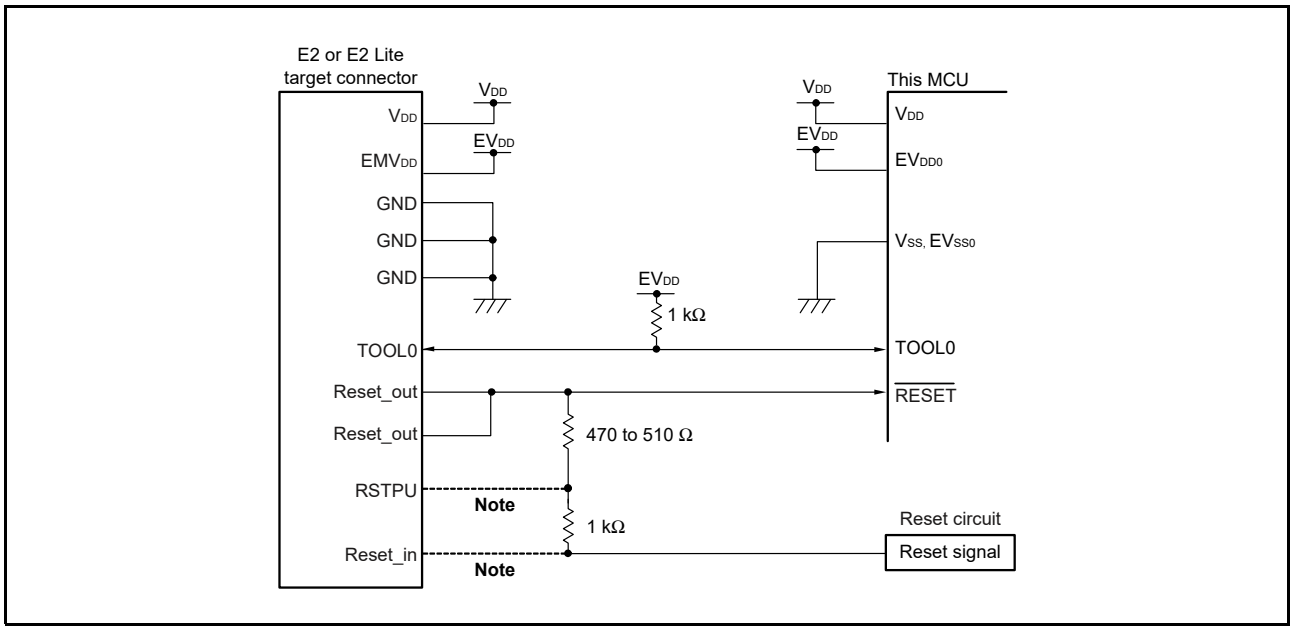
Section 25 On-chip Debugging

25.1 Connection between the E2 or E2 Lite On-chip Debugging Emulator and This MCU

On-chip debugging is handled by connecting this MCU and host machine through the E2 or E2 Lite on-chip debugging emulator. Pins V_{DD}, $\overline{RESET}$, TOOL0, and V_{SS} are used for connection to the emulator. Serial communications are handled through the TOOL0 pin as a single-line UART connection. For details and usage notes on the circuit to make the connection, refer to the **E1/E20/E2 Emulator, E2 Emulator Lite Additional Document for User's Manual (Notes on Connection of RL78) (R20UT1994)**.

Caution **This MCU has on-chip debugging functionality for use in the development and evaluation of user systems. Do not use on-chip debugging with products designated as part of mass production, because using this function may cause the guaranteed number of times the flash memory is rewritten to be exceeded, and product reliability therefore cannot be guaranteed. Renesas Electronics is not liable for problems occurring when on-chip debugging is used with products designated as part of mass production.**

Figure 25 - 1 Example of Connection between the E2 or E2 Lite On-chip Debugging Emulator and This MCU

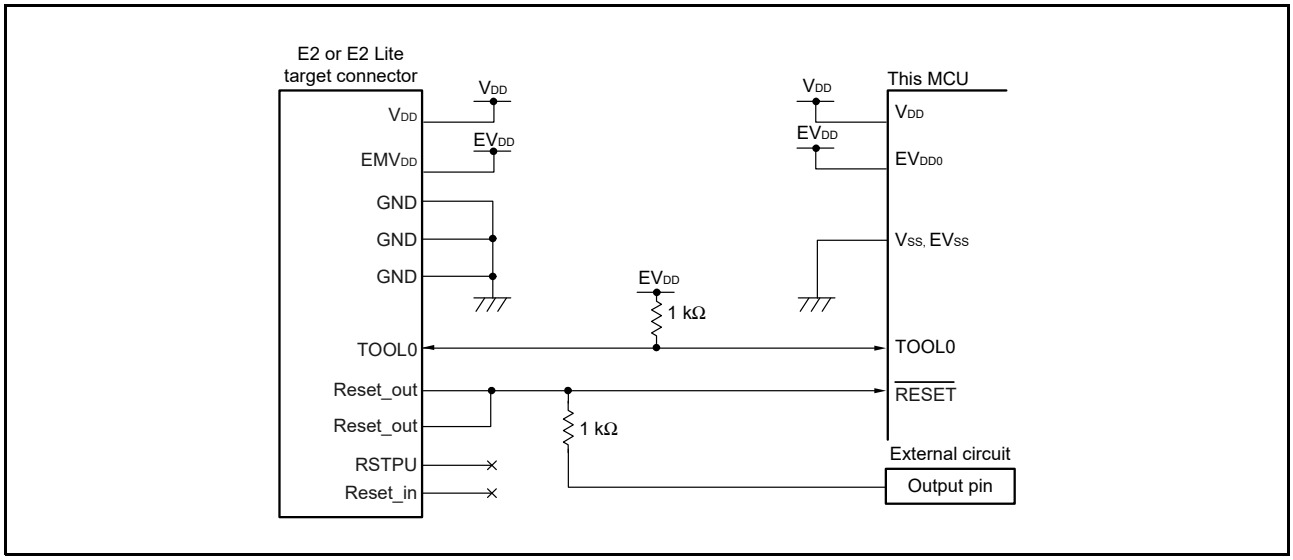


Note The connection shown as a broken line is not required for serial programming.

Remark Pins EV_{DD0} and EV_{SS0} are not present in some products. In such cases, use the V_{DD} and V_{SS} pins as the alternative.

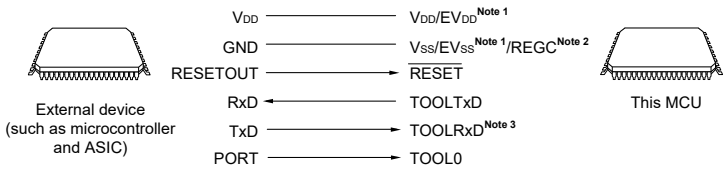
Note that in target systems where the $\overline{\text{RESET}}$ pin is used as a port function (P125), isolate the connection between the external circuit and the Reset_out pin of the E2 or E2 Lite target connector.

Figure 25 - 2 Example of Connection between the E2 or E2 Lite On-chip Debugging Emulator and This MCU when the $\overline{\text{RESET}}$ Pin Is Used as a Port Function (P125)



25.2 Connection between the External Device that Incorporates UART and This MCU

On-board communications between an external device (a microcontroller or ASIC) that is connected to this MCU via a UART and the host machine is possible. Pins V_{DD}, $\overline{RESET}$, TOOL0, V_{SS}, TOOLTxD, and TOOLRxD are used for the communications. Communications between the external device and this MCU is established by serial communications using the TOOLTxD and TOOLRxD pins via the dedicated UART of this MCU. For details and usage notes on the circuit to make the connection, refer to the **RL78 Debugging Functions Using the Serial Port (R20AN0632EJ)**.



Note 1. This pin is only present in the 64-pin products.

Note 2. Connect the REGC pin to the ground via a capacitor (0.47 to 1 µF).

Note 3. Set the port pin with which TOOLRxD is multiplexed as an input. The input to the input buffer must also be enabled by using the PDIDISxx register.

25.3 Security Settings for On-chip Debugging

To protect against third parties reading the contents of memory, on-chip debugging includes the following functionality.

- Disabling of connection between this MCU and the programmer or on-chip debugger (see **24.9 Security Settings** in **Section 24 Flash Memory**).
- On-chip debugging control bits in the flash memory at 000C3H (see **Section 23 Option Bytes**)
- An area in the range from 000C4H to 000CDH**Note** to hold the security ID code for on-chip debugging.

Note The area to hold the security ID code for use in on-chip debugging is also used to hold the ID code for the programmer connection ID authentication when a programmer is to be used.

Table 25 - 1 On-chip Debug Security ID

Address	Security ID Code for On-chip Debugging
000C4H to 000CDH	Any 10-byte ID code**Notes 1, 2**
040C4H to 040CDH	

Note 1. The setting FFFFFFFFFFFFFFFFFFFFH is not allowed.

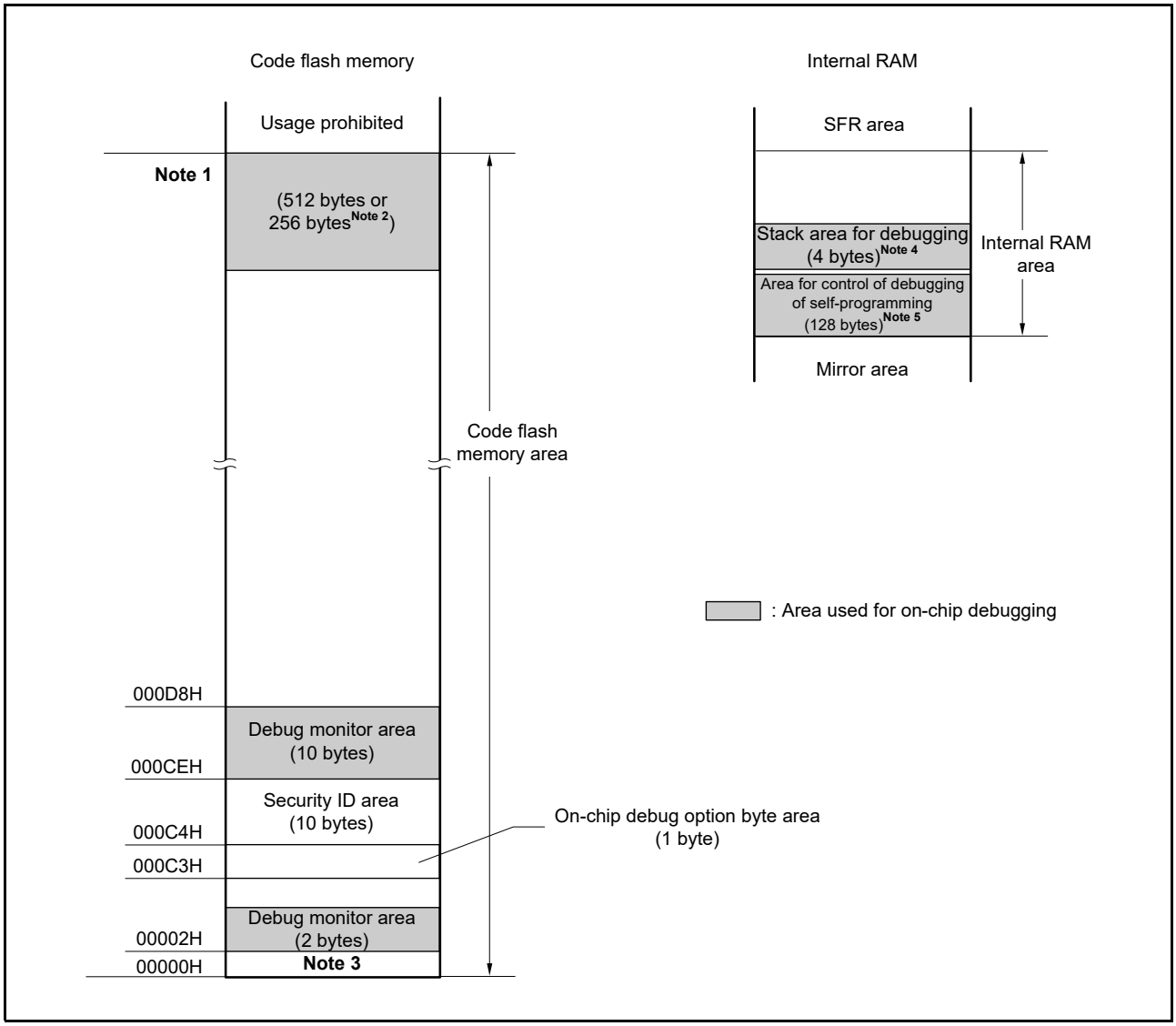
Note 2. Set the 10-byte security ID for on-chip debugging at both 000C4H to 000CDH and 040C4H to 040CDH when boot swapping is to be used or the value of the BTFLG bit in the FLSEC register is 0.

25.4 Allocation of Memory Spaces to User Resources

Allocation of memory spaces to user resources is required before communications between this MCU and E2 or E2 Lite on-chip debugging emulator, and on-chip debugging, can proceed. If you are using an assembler or compiler from Renesas Electronics, you can use linker options to allocate the memory spaces.

1. Allocation of memory spaces to the user program
   The shaded areas in **Figure 25 - 3** are reserved for the monitor program for debugging, and user programs and data cannot be allocated to these areas. When using on-chip debugging, ensure that nothing is allocated to these areas so that they can be secured for on-chip debugging. Also ensure that the contents of these areas are not modified by the user program.

Figure 25 - 3 Memory Spaces Allocated for Use by the Monitor Program for Debugging



Note 1. The address depends on the products as shown below.

Products	Address of **Note 1**
R7F103GxG (x = 6, B, F, G, L)	1FFFFH
R7F103GxJ (x = 6, B, F, G, L)	3FFFFH

Note 2. When the realtime RAM monitor (RRM) and dynamic memory modification (DMM) are not to be used, the size of this area is 256 bytes.

Note 3. During debugging, the reset vector is relocated to the address of the monitor program.

Note 4. Since this area is allocated immediately below the portion of the main stack area that is currently in use, the address range of this area depends on the amount of the stack in use other than for debugging. Accordingly, four additional bytes are required for the entire stack area. In the case of self-programming, this is a 12-byte area, so 12 additional bytes are required.

Note 5. The on-chip debugger uses the 128-byte RAM areas of the products listed in the table below for breaks in self-programming. The correspondences between target products and RAM areas are given in the table.

Products	RAM Area to be Used (128 bytes)
R7F103GxG (x = 6, B, F, G, L)	FDF00H to FDF7FH
R7F103GxJ (x = 6, B, F, G, L)	FDF00H to FDF7FH

The given address range will not be in use unless the setting to enable debugging during self-programming are made by the on-chip debugger. For the setting of debugging of self-programming, refer to the user's manual for the given integrated development environment.

Section 26 BCD Correction Circuit

26.1 BCD Correction Circuit Function

The result of addition/subtraction of the BCD (binary-coded decimal) code and BCD code can be obtained as BCD code with this circuit. The decimal correction operation result is obtained by performing addition/subtraction having the A register as the operand and then adding/subtracting the BCD correction result register (BCDADJ).

26.2 Register for Controlling the BCD Correction Circuit

The following register is used to control the BCD correction circuit.

- BCD correction result register (BCDADJ)

26.2.1 BCD correction result register (BCDADJ)

The BCDADJ register holds correction values for obtaining the add/subtract result as BCD code through add/subtract instructions using the A register as the operand. The value read from the BCDADJ register varies depending on the value of the A register when it is read and those of the CY and AC flags. The BCDADJ register is read by an 8-bit memory manipulation instruction. The value of this register following a reset is undefined.

Figure 26 - 1 Format of BCD Correction Result Register (BCDADJ)

Address: F00FEH
After reset: Undefined
R/W: R

Symbol	7	6	5	4	3	2	1	0
BCDADJ								

26.3 BCD Correction Circuit Operation

The basic operation of the BCD correction circuit is as follows.

1. Addition: Obtaining the result of the binary addition of BCD values and BCD values as a BCD value

<1> The BCD code value to which addition is performed is stored in the A register.

<2> By adding the value of the A register and the second operand (value of one more BCD code to be added) as are in binary, the binary operation result is stored in the A register and the correction value is stored in the BCD correction result register (BCDADJ).

<3> Decimal correction is performed by adding in binary the values of the A register (addition result in binary) and the BCDADJ register (correction value), and the correction result is stored in the A register and CY flag.

Caution The value read from the BCDADJ register varies depending on the value of the A register when it is read and those of the CY and AC flags. Therefore, execute the instruction <3> after the instruction <2> instead of executing any other instructions. To perform BCD correction in the interrupt enabled state, saving and restoring the A register is required within the interrupt function. PSW (CY and AC flags) is restored by the RETI instruction.

Examples are shown below.

Example 1: 99 + 89 = 188

Instruction	A Register	CY Flag	AC Flag	BCDADJ Register
MOV A, #99H ; <1>	99H	—	—	—
ADD A, #89H ; <2>	22H	1	1	66H
ADD A, !BCDADJ ; <3>	88H	1	0	—

Example 2: 85 + 15 = 100

Instruction	A Register	CY Flag	AC Flag	BCDADJ Register
MOV A, #85H ; <1>	85H	—	—	—
ADD A, #15H ; <2>	9AH	0	0	66H
ADD A, !BCDADJ ; <3>	00H	1	1	—

Example 3: 80 + 80 = 160

Instruction	A Register	CY Flag	AC Flag	BCDADJ Register
MOV A, #80H ; <1>	80H	—	—	—
ADD A, #80H ; <2>	00H	1	0	60H
ADD A, !BCDADJ ; <3>	60H	1	0	—

2. Subtraction: Obtaining the result of the binary subtraction of BCD values and BCD values as a BCD value

<1> The BCD code value from which subtraction is performed is stored in the A register.

<2> By subtracting the value of the second operand (value of BCD code to be subtracted) from the A register as is in binary, the calculation result in binary is stored in the A register, and the correction value is stored in the BCD correction result register (BCDADJ).

<3> Decimal correction is performed by subtracting the value of the BCDADJ register (correction value) from the A register (subtraction result in binary) in binary, and the correction result is stored in the A register and CY flag.

Caution The value read from the BCDADJ register varies depending on the value of the A register when it is read and those of the CY and AC flags. Therefore, execute the instruction <3> after the instruction <2> instead of executing any other instructions. To perform BCD correction in the interrupt enabled state, saving and restoring the A register is required within the interrupt function. PSW (CY and AC flags) is restored by the RETI instruction.

An example is shown below.

Example: 91 - 52 = 39

Instruction	A Register	CY Flag	AC Flag	BCDADJ Register
MOV A, #91H ; <1>	91H	—	—	—
SUB A, #52H ; <2>	3FH	0	1	06H
SUB A, !BCDADJ ; <3>	39H	0	0	—

Section 27 Instruction Set

This section lists the instructions in the RL78 microcontroller instruction set. For details of each operation and operation code, refer to the separate document **RL78 Family User's Manual Software (R01US0015)**.

27.1 Conventions Used in Operation List

27.1.1 Operand identifiers and description methods

Operands are described in the “Operand” column of each instruction in accordance with the description method of the instruction operand identifier (refer to the assembler specifications for details). When there are two or more description methods, select one of them. Alphabetic letters in capitals and the symbols, #, !, !!, $, $!, [ ], and ES: are keywords and are described as they are. Each symbol has the following meaning.

- #: Immediate data specification
- !: 16-bit absolute address specification
- !!: 20-bit absolute address specification
- $: 8-bit relative address specification
- $!: 16-bit relative address specification
- [ ]: Indirect address specification
- ES: Extension address specification

In the case of immediate data, describe an appropriate numeric value or a label. When using a label, be sure to describe the #, !, !!, $, $!, [ ], and ES: symbols. For operand register identifiers, r and rp, either function names (X, A, C, etc.) or absolute names (names in parentheses in the table below, R0, R1, R2, etc.) can be used for description.

Table 27 - 1 Operand Identifiers and Description Methods

Identifier	Description Method
r rp sfr sfrp	X (R0), A (R1), C (R2), B (R3), E (R4), D (R5), L (R6), H (R7) AX (RP0), BC (RP1), DE (RP2), HL (RP3) Special-function register symbol (SFR symbol) FFF00H to FFFFFH Special-function register symbols (16-bit manipulatable SFR symbol. Even addresses only**Note**) FFF00H to FFFFFH
saddr saddrp	FFE20H to FFF1FH Immediate data or labels FFE20H to FF1FH Immediate data or labels (even addresses only**Note**)
addr20 addr16 addr5	00000H to FFFFFH Immediate data or labels 0000H to FFFFH Immediate data or labels (only even addresses for 16-bit data transfer instructions**Note**) 0080H to 00BFH Immediate data or labels (even addresses only)
word byte bit	16-bit immediate data or label 8-bit immediate data or label 3-bit immediate data or label
RBn	RB0 to RB3

Note Bit 0 = 0 when an odd address is specified.

Remark The special function registers can be described to operand sfr as symbols. See **Table 3 - 5 List of Special Function Registers (SFRs)** for the symbols of the special function registers. The extended special function registers can be described to operand !addr16 as symbols. See **Table 3 - 6 List of Extended Special Function Registers (2nd SFRs)** for the symbols of the extended special function registers.

27.1.2 Description of operation column

The operation when the instruction is executed is shown in the “Operation” column using the following symbols.

Table 27 - 2 Symbols in “Operation” Column

Symbol	Function
A	A register; 8-bit accumulator
X	X register
B	B register
C	C register
D	D register
E	E register
H	H register
L	L register
ES	ES register
CS	CS register
AX	AX register pair; 16-bit accumulator
BC	BC register pair
DE	DE register pair
HL	HL register pair
PC	Program counter
SP	Stack pointer
PSW	Program status word
CY	Carry flag
AC	Auxiliary carry flag
Z	Zero flag
RBS	Register bank select flag
IE	Interrupt request enable flag
()	Memory contents indicated by address or register contents in parentheses
X_H, X_L X_S, X_H, X_L	16-bit registers: X_H = higher 8 bits, X_L = lower 8 bits 20-bit registers: X_S = (bits 19 to 16), X_H = (bits 15 to 8), X_L = (bits 7 to 0)
∧	Logical product (AND)
∨	Logical sum (OR)
∨̲	Exclusive logical sum (exclusive OR)
—	Inverted data
addr5	16-bit immediate data (even addresses only in 0080H to 00BFH)
addr16	16-bit immediate data
addr20	20-bit immediate data
jdisp8	Signed 8-bit data (displacement value)
jdisp16	Signed 16-bit data (displacement value)

27.1.3 Description of flag operation column

The change of the flag value when the instruction is executed is shown in the “Flag” column using the following symbols.

Table 27 - 3 Symbols in “Flag” Column

Symbol	Change of Flag Value
(Blank)	Unchanged
0	Cleared to 0
1	Set to 1
×	Set or cleared according to the result
R	Previously saved value is restored

27.1.4 PREFIX instruction

Instructions with “ES:” have a PREFIX operation code as a prefix to extend the accessible data area to the 1 MB space (00000H to FFFFFH), by adding the ES register value to the 64 KB space from F0000H to FFFFFH. When a PREFIX operation code is attached as a prefix to the target instruction, only one instruction immediately after the PREFIX operation code is executed as the addresses with the ES register value added. An interrupt is not acknowledged between a PREFIX instruction code and the instruction immediately after.

Table 27 - 4 Use Example of PREFIX Operation Code

Instruction	Opcode				
	1	2	3	4	5
MOV !addr16, #byte	CFH	!addr16		#byte	—
MOV ES:!addr16, #byte	11H	CFH	!addr16		#byte
MOV A, [HL]	8BH	—	—	—	—
MOV A, ES:[HL]	11H	8BH	—	—	—

Caution Set the ES register value with MOV ES, A, etc., before executing the PREFIX instruction.

27.2 Operation List

Table 27 - 5 Operation List (1/18)

Instruction Group	Mnemonic	Operands	Bytes	Clock Cycles		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit data transfer	MOV	r, #byte	2	1	—	r ← byte			
		PSW, #byte	3	3	—	PSW ← byte	×	×	×
		CS, #byte	3	1	—	CS ← byte			
		ES, #byte	2	1	—	ES ← byte			
		!addr16, #byte	4	1	—	(addr16) ← byte			
		ES:!addr16, #byte	5	2	—	(ES, addr16) ← byte			
		saddr, #byte	3	1	—	(saddr) ← byte			
		sfr, #byte	3	1	—	sfr ← byte			
		[DE+byte], #byte	3	1	—	(DE + byte) ← byte			
		ES:[DE+byte], #byte	4	2	—	((ES, DE) + byte) ← byte			
		[HL+byte], #byte	3	1	—	(HL + byte) ← byte			
		ES:[HL+byte], #byte	4	2	—	((ES, HL) + byte) ← byte			
		[SP+byte], #byte	3	1	—	(SP + byte) ← byte			
		word[B], #byte	4	1	—	(B + word) ← byte			
		ES:word[B], #byte	5	2	—	((ES, B) + word) ← byte			
		word[C], #byte	4	1	—	(C+word) ← byte			
		ES:word[C], #byte	5	2	—	((ES, C) + word) ← byte			
		word[BC], #byte	4	1	—	(BC+word) ← byte			
		ES:word[BC], #byte	5	2	—	((ES, BC) + word) ← byte			
		A, r**Note 3**	1	1	—	A ← r			
		r**Note 3**, A	1	1	—	r ← A			
		A, PSW	2	1	—	A ← PSW			
		PSW, A	2	3	—	PSW ← A	×	×	×
		A, CS	2	1	—	A ← CS			
		CS, A	2	1	—	CS ← A			
		A, ES	2	1	—	A ← ES			
		ES, A	2	1	—	ES ← A			
		A, !addr16	3	1	4	A ← (addr16)			
		A, ES:!addr16	4	2	5	A ← (ES, addr16)			
		!addr16, A	3	1	—	(addr16) ← A			
		ES:!addr16, A	4	2	—	(ES, addr16) ← A			
		A, saddr	2	1	—	A ← (saddr)			
		saddr, A	2	1	—	(saddr) ← A			

Note 1. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the internal RAM area, SFR area, or 2nd SFR area, or when the instruction does not involve data access.

Note 2. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the code flash memory area, or of access to the data flash memory area by an 8-bit instruction.

Note 3. Except when r = A.

Remark The listed numbers of clock cycles are for when the program is present in the internal ROM (flash memory) area. If the instruction is fetched from the internal RAM area, the number becomes up to double the listed number plus 3 clock cycles.

Table 27 - 5 Operation List (2/18)

Instruction Group	Mnemonic	Operands	Bytes	Clock Cycles		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit data transfer	MOV	A, sfr	2	1	—	A ← sfr			
		sfr, A	2	1	—	sfr ← A			
		A, [DE]	1	1	4	A ← (DE)			
		[DE], A	1	1	—	(DE) ← A			
		A, ES:[DE]	2	2	5	A ← (ES, DE)			
		ES:[DE], A	2	2	—	(ES, DE) ← A			
		A, [HL]	1	1	4	A ← (HL)			
		[HL], A	1	1	—	(HL) ← A			
		A, ES:[HL]	2	2	5	A ← (ES, HL)			
		ES:[HL], A	2	2	—	(ES, HL) ← A			
		A, [DE+byte]	2	1	4	A ← (DE + byte)			
		[DE+byte], A	2	1	—	(DE + byte) ← A			
		A, ES:[DE+byte]	3	2	5	A ← ((ES, DE) + byte)			
		ES:[DE+byte], A	3	2	—	((ES, DE) + byte) ← A			
		A, [HL+byte]	2	1	4	A ← (HL + byte)			
		[HL+byte], A	2	1	—	(HL + byte) ← A			
		A, ES:[HL+byte]	3	2	5	A ← ((ES, HL) + byte)			
		ES:[HL+byte], A	3	2	—	((ES, HL) + byte) ← A			
		A, [SP+byte]	2	1	—	A ← (SP + byte)			
		[SP+byte], A	2	1	—	(SP + byte) ← A			
		A, word[B]	3	1	4	A ← (B + word)			
		word[B], A	3	1	—	(B + word) ← A			
		A, ES:word[B]	4	2	5	A ← ((ES, B) + word)			
		ES:word[B], A	4	2	—	((ES, B) + word) ← A			
		A, word[C]	3	1	4	A ← (C + word)			
		word[C], A	3	1	—	(C + word) ← A			
		A, ES:word[C]	4	2	5	A ← ((ES, C) + word)			
		ES:word[C], A	4	2	—	((ES, C) + word) ← A			
		A, word[BC]	3	1	4	A ← (BC + word)			
		word[BC], A	3	1	—	(BC + word) ← A			
		A, ES:word[BC]	4	2	5	A ← ((ES, BC) + word)			
		ES:word[BC], A	4	2	—	((ES, BC) + word) ← A			

Note 1. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the internal RAM area, SFR area, or 2nd SFR area, or when the instruction does not involve data access.

Note 2. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the code flash memory area, or of access to the data flash memory area by an 8-bit instruction.

Remark The listed numbers of clock cycles are for when the program is present in the internal ROM (flash memory) area. If the instruction is fetched from the internal RAM area, the number becomes up to double the listed number plus 3 clock cycles.

Table 27 - 5 Operation List (3/18)

Instruction Group	Mnemonic	Operands	Bytes	Clock Cycles		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit data transfer	MOV	A, [HL+B]	2	1	4	A ← (HL + B)			
		[HL+B], A	2	1	—	(HL + B) ← A			
		A, ES:[HL+B]	3	2	5	A ← ((ES, HL) + B)			
		ES:[HL+B], A	3	2	—	((ES, HL) + B) ← A			
		A, [HL+C]	2	1	4	A ← (HL + C)			
		[HL+C], A	2	1	—	(HL + C) ← A			
		A, ES:[HL+C]	3	2	5	A ← ((ES, HL) + C)			
		ES:[HL+C], A	3	2	—	((ES, HL) + C) ← A			
		X, !addr16	3	1	4	X ← (addr16)			
		X, ES:!addr16	4	2	5	X ← (ES, addr16)			
		X, saddr	2	1	—	X ← (saddr)			
		B, !addr16	3	1	4	B ← (addr16)			
		B, ES:!addr16	4	2	5	B ← (ES, addr16)			
		B, saddr	2	1	—	B ← (saddr)			
		C, !addr16	3	1	4	C ← (addr16)			
		C, ES:!addr16	4	2	5	C ← (ES, addr16)			
		C, saddr	2	1	—	C ← (saddr)			
		ES, saddr	3	1	—	ES ← (saddr)			
	XCH	A, r[Note 3]	1 (r = X) 2 (other than r = X)	1	—	A ←→ r			
		A, !addr16	4	2	—	A ←→ (addr16)			
		A, ES:!addr16	5	3	—	A ←→ (ES, addr16)			
		A, saddr	3	2	—	A ←→ (saddr)			
		A, sfr	3	2	—	A ←→ sfr			
		A, [DE]	2	2	—	A ←→ (DE)			
		A, ES:[DE]	3	3	—	A ←→ (ES, DE)			
		A, [HL]	2	2	—	A ←→ (HL)			
		A, ES:[HL]	3	3	—	A ←→ (ES, HL)			
		A, [DE+byte]	3	2	—	A ←→ (DE + byte)			
		A, ES:[DE+byte]	4	3	—	A ←→ ((ES, DE) + byte)			
		A, [HL+byte]	3	2	—	A ←→ (HL + byte)			
		A, ES:[HL+byte]	4	3	—	A ←→ ((ES, HL) + byte)			

Note 1. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the internal RAM area, SFR area, or 2nd SFR area, or when the instruction does not involve data access.

Note 2. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the code flash memory area, or of access to the data flash memory area by an 8-bit instruction.

Note 3. Except when r = A.

Remark The listed numbers of clock cycles are for when the program is present in the internal ROM (flash memory) area. If the instruction is fetched from the internal RAM area, the number becomes up to double the listed number plus 3 clock cycles.

Table 27 - 5 Operation List (4/18)

Instruction Group	Mnemonic	Operands	Bytes	Clock Cycles		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit data transfer	XCH	A, [HL+B]	2	2	—	A ←→ (HL + B)			
		A, ES:[HL+B]	3	3	—	A ←→ ((ES, HL) + B)			
		A, [HL+C]	2	2	—	A ←→ (HL + C)			
		A, ES:[HL+C]	3	3	—	A ←→ ((ES, HL) + C)			
	ONEB	A	1	1	—	A ← 01H			
		X	1	1	—	X ← 01H			
		B	1	1	—	B ← 01H			
		C	1	1	—	C ← 01H			
		!addr16	3	1	—	(addr16) ← 01H			
		ES:!addr16	4	2	—	(ES, addr16) ← 01H			
		saddr	2	1	—	(saddr) ← 01H			
	CLRB	A	1	1	—	A ← 00H			
		X	1	1	—	X ← 00H			
		B	1	1	—	B ← 00H			
		C	1	1	—	C ← 00H			
		!addr16	3	1	—	(addr16) ← 00H			
		ES:!addr16	4	2	—	(ES, addr16) ← 00H			
		saddr	2	1	—	(saddr) ← 00H			
	MOVS	[HL+byte], X	3	1	—	(HL + byte) ← X	×		×
		ES:[HL+byte], X	4	2	—	(ES, HL + byte) ← X	×		×
16-bit data transfer	MOVW	rp, #word	3	1	—	rp ← word			
		saddrp, #word	4	1	—	(saddrp) ← word			
		sfrp, #word	4	1	—	sfrp ← word			
		AX, rp**Note 3**	1	1	—	AX ← rp			
		rp**Note 3**, AX	1	1	—	rp ← AX			
		AX, !addr16	3	1	4	AX ← (addr16)			
		!addr16, AX	3	1	—	(addr16) ← AX			
		AX, ES:!addr16	4	2	5	AX ← (ES, addr16)			
		ES:!addr16, AX	4	2	—	(ES, addr16) ← AX			
		AX, saddrp	2	1	—	AX ← (saddrp)			
		saddrp, AX	2	1	—	(saddrp) ← AX			
		AX, sfrp	2	1	—	AX ← sfrp			
		sfrp, AX	2	1	—	sfrp ← AX			

Note 1. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the internal RAM area, SFR area, or 2nd SFR area, or when the instruction does not involve data access.

Note 2. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the code flash memory area, or of access to the data flash memory area by an 8-bit instruction.

Note 3. Except when rp = AX.

Remark The listed numbers of clock cycles are for when the program is present in the internal ROM (flash memory) area. If the instruction is fetched from the internal RAM area, the number becomes up to double the listed number plus 3 clock cycles.

Table 27 - 5 Operation List (5/18)

Instruction Group	Mnemonic	Operands	Bytes	Clock Cycles		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
16-bit data transfer	MOVW	AX, [DE]	1	1	4	AX ← (DE)			
		[DE], AX	1	1	—	(DE) ← AX			
		AX, ES:[DE]	2	2	5	AX ← (ES, DE)			
		ES:[DE], AX	2	2	—	(ES, DE) ← AX			
		AX, [HL]	1	1	4	AX ← (HL)			
		[HL], AX	1	1	—	(HL) ← AX			
		AX, ES:[HL]	2	2	5	AX ← (ES, HL)			
		ES:[HL], AX	2	2	—	(ES, HL) ← AX			
		AX, [DE+byte]	2	1	4	AX ← (DE + byte)			
		[DE+byte], AX	2	1	—	(DE + byte) ← AX			
		AX, ES:[DE+byte]	3	2	5	AX ← ((ES, DE) + byte)			
		ES:[DE+byte], AX	3	2	—	((ES, DE) + byte) ← AX			
		AX, [HL+byte]	2	1	4	AX ← (HL + byte)			
		[HL+byte], AX	2	1	—	(HL + byte) ← AX			
		AX, ES:[HL+byte]	3	2	5	AX ← ((ES, HL) + byte)			
		ES:[HL+byte], AX	3	2	—	((ES, HL) + byte) ← AX			
		AX, [SP+byte]	2	1	—	AX ← (SP + byte)			
		[SP+byte], AX	2	1	—	(SP + byte) ← AX			
		AX, word[B]	3	1	4	AX ← (B + word)			
		word[B], AX	3	1	—	(B + word) ← AX			
		AX, ES:word[B]	4	2	5	AX ← ((ES, B) + word)			
		ES:word[B], AX	4	2	—	((ES, B) + word) ← AX			
		AX, word[C]	3	1	4	AX ← (C + word)			
		word[C], AX	3	1	—	(C + word) ← AX			
		AX, ES:word[C]	4	2	5	AX ← ((ES, C) + word)			
		ES:word[C], AX	4	2	—	((ES, C) + word) ← AX			
		AX, word[BC]	3	1	4	AX ← (BC + word)			
		word[BC], AX	3	1	—	(BC + word) ← AX			
		AX, ES:word[BC]	4	2	5	AX ← ((ES, BC) + word)			
		ES:word[BC], AX	4	2	—	((ES, BC) + word) ← AX			

Note 1. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the internal RAM area, SFR area, or 2nd SFR area, or when the instruction does not involve data access.

Note 2. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the code flash memory area, or of access to the data flash memory area by an 8-bit instruction.

Remark The listed numbers of clock cycles are for when the program is present in the internal ROM (flash memory) area. If the instruction is fetched from the internal RAM area, the number becomes up to double the listed number plus 3 clock cycles.

Table 27 - 5 Operation List (6/18)

Instruction Group	Mnemonic	Operands	Bytes	Clock Cycles		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
16-bit data transfer	MOVW	BC, !addr16	3	1	4	BC ← (addr16)			
		BC, ES:!addr16	4	2	5	BC ← (ES, addr16)			
		DE, !addr16	3	1	4	DE ← (addr16)			
		DE, ES:!addr16	4	2	5	DE ← (ES, addr16)			
		HL, !addr16	3	1	4	HL ← (addr16)			
		HL, ES:!addr16	4	2	5	HL ← (ES, addr16)			
		BC, saddrp	2	1	—	BC ← (saddrp)			
		DE, saddrp	2	1	—	DE ← (saddrp)			
		HL, saddrp	2	1	—	HL ← (saddrp)			
	XCHW	AX, rp Note 3	1	1	—	AX ←→ rp			
	ONEW	AX	1	1	—	AX ← 0001H			
		BC	1	1	—	BC ← 0001H			
	CLRW	AX	1	1	—	AX ← 0000H			
		BC	1	1	—	BC ← 0000H			
8-bit operation	ADD	A, #byte	2	1	—	A, CY ← A + byte	×	×	×
		saddr, #byte	3	2	—	(saddr), CY ← (saddr) + byte	×	×	×
		A, r Note 4	2	1	—	A, CY ← A + r	×	×	×
		r, A	2	1	—	r, CY ← r + A	×	×	×
		A, !addr16	3	1	4	A, CY ← A + (addr16)	×	×	×
		A, ES:!addr16	4	2	5	A, CY ← A + (ES, addr16)	×	×	×
		A, saddr	2	1	—	A, CY ← A + (saddr)	×	×	×
		A, [HL]	1	1	4	A, CY ← A + (HL)	×	×	×
		A, ES:[HL]	2	2	5	A, CY ← A + (ES, HL)	×	×	×
		A, [HL+byte]	2	1	4	A, CY ← A + (HL + byte)	×	×	×
		A, ES:[HL+byte]	3	2	5	A, CY ← A + ((ES, HL) + byte)	×	×	×
		A, [HL+B]	2	1	4	A, CY ← A + (HL + B)	×	×	×
		A, ES:[HL+B]	3	2	5	A, CY ← A + ((ES, HL) + B)	×	×	×
		A, [HL+C]	2	1	4	A, CY ← A + (HL + C)	×	×	×
		A, ES:[HL+C]	3	2	5	A, CY ← A + ((ES, HL) + C)	×	×	×

Note 1. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the internal RAM area, SFR area, or 2nd SFR area, or when the instruction does not involve data access.

Note 2. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the code flash memory area, or of access to the data flash memory area by an 8-bit instruction.

Note 3. Except when rp = AX.

Note 4. Except when r = A.

Remark The listed numbers of clock cycles are for when the program is present in the internal ROM (flash memory) area. If the instruction is fetched from the internal RAM area, the number becomes up to double the listed number plus 3 clock cycles.

Table 27 - 5 Operation List (7/18)

Instruction Group	Mnemonic	Operands	Bytes	Clock Cycles		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit operation	ADDC	A, #byte	2	1	—	A, CY ← A + byte + CY	×	×	×
		saddr, #byte	3	2	—	(saddr), CY ← (saddr) + byte + CY	×	×	×
		A, r Note 3	2	1	—	A, CY ← A + r + CY	×	×	×
		r, A	2	1	—	r, CY ← r + A + CY	×	×	×
		A, !addr16	3	1	4	A, CY ← A + (addr16) + CY	×	×	×
		A, ES:!addr16	4	2	5	A, CY ← A + (ES, addr16) + CY	×	×	×
		A, saddr	2	1	—	A, CY ← A + (saddr) + CY	×	×	×
		A, [HL]	1	1	4	A, CY ← A + (HL) + CY	×	×	×
		A, ES:[HL]	2	2	5	A, CY ← A + (ES, HL) + CY	×	×	×
		A, [HL+byte]	2	1	4	A, CY ← A + (HL + byte) + CY	×	×	×
		A, ES:[HL+byte]	3	2	5	A, CY ← A + ((ES, HL) + byte) + CY	×	×	×
		A, [HL+B]	2	1	4	A, CY ← A + (HL + B) + CY	×	×	×
		A, ES:[HL+B]	3	2	5	A, CY ← A + ((ES, HL) + B) + CY	×	×	×
		A, [HL+C]	2	1	4	A, CY ← A + (HL + C) + CY	×	×	×
		A, ES:[HL+C]	3	2	5	A, CY ← A + ((ES, HL) + C) + CY	×	×	×
	SUB	A, #byte	2	1	—	A, CY ← A - byte	×	×	×
		saddr, #byte	3	2	—	(saddr), CY ← (saddr) - byte	×	×	×
		A, r Note 3	2	1	—	A, CY ← A - r	×	×	×
		r, A	2	1	—	r, CY ← r - A	×	×	×
		A, !addr16	3	1	4	A, CY ← A - (addr16)	×	×	×
		A, ES:!addr16	4	2	5	A, CY ← A - (ES, addr16)	×	×	×
		A, saddr	2	1	—	A, CY ← A - (saddr)	×	×	×
		A, [HL]	1	1	4	A, CY ← A - (HL)	×	×	×
		A, ES:[HL]	2	2	5	A, CY ← A - (ES, HL)	×	×	×
		A, [HL+byte]	2	1	4	A, CY ← A - (HL + byte)	×	×	×
		A, ES:[HL+byte]	3	2	5	A, CY ← A - ((ES, HL) + byte)	×	×	×
		A, [HL+B]	2	1	4	A, CY ← A - (HL + B)	×	×	×
		A, ES:[HL+B]	3	2	5	A, CY ← A - ((ES, HL) + B)	×	×	×
		A, [HL+C]	2	1	4	A, CY ← A - (HL + C)	×	×	×
		A, ES:[HL+C]	3	2	5	A, CY ← A - ((ES, HL) + C)	×	×	×

Note 1. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the internal RAM area, SFR area, or 2nd SFR area, or when the instruction does not involve data access.

Note 2. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the code flash memory area, or of access to the data flash memory area by an 8-bit instruction.

Note 3. Except when r = A.

Remark The listed numbers of clock cycles are for when the program is present in the internal ROM (flash memory) area. If the instruction is fetched from the internal RAM area, the number becomes up to double the listed number plus 3 clock cycles.

Table 27 - 5 Operation List (8/18)

Instruction Group	Mnemonic	Operands	Bytes	Clock Cycles		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit operation	SUBC	A, #byte	2	1	—	A, CY ← A - byte - CY	×	×	×
		saddr, #byte	3	2	—	(saddr), CY ← (saddr) - byte - CY	×	×	×
		A, r Note 3	2	1	—	A, CY ← A - r - CY	×	×	×
		r, A	2	1	—	r, CY ← r - A - CY	×	×	×
		A, !addr16	3	1	4	A, CY ← A - (addr16) - CY	×	×	×
		A, ES:!addr16	4	2	5	A, CY ← A - (ES, addr16) - CY	×	×	×
		A, saddr	2	1	—	A, CY ← A - (saddr) - CY	×	×	×
		A, [HL]	1	1	4	A, CY ← A - (HL) - CY	×	×	×
		A, ES:[HL]	2	2	5	A, CY ← A - (ES, HL) - CY	×	×	×
		A, [HL+byte]	2	1	4	A, CY ← A - (HL + byte) - CY	×	×	×
		A, ES:[HL+byte]	3	2	5	A, CY ← A - ((ES, HL) + byte) - CY	×	×	×
		A, [HL+B]	2	1	4	A, CY ← A - (HL + B) - CY	×	×	×
		A, ES:[HL+B]	3	2	5	A, CY ← A - ((ES, HL) + B) - CY	×	×	×
		A, [HL+C]	2	1	4	A, CY ← A - (HL + C) - CY	×	×	×
		A, ES:[HL+C]	3	2	5	A, CY ← A - ((ES:HL) + C) - CY	×	×	×
	AND	A, #byte	2	1	—	A ← A ∧ byte	×		
		saddr, #byte	3	2	—	(saddr) ← (saddr) ∧ byte	×		
		A, r Note 3	2	1	—	A ← A ∧ r	×		
		r, A	2	1	—	r ← r ∧ A	×		
		A, !addr16	3	1	4	A ← A ∧ (addr16)	×		
		A, ES:!addr16	4	2	5	A ← A ∧ (ES:addr16)	×		
		A, saddr	2	1	—	A ← A ∧ (saddr)	×		
		A, [HL]	1	1	4	A ← A ∧ (HL)	×		
		A, ES:[HL]	2	2	5	A ← A ∧ (ES:HL)	×		
		A, [HL+byte]	2	1	4	A ← A ∧ (HL + byte)	×		
		A, ES:[HL+byte]	3	2	5	A ← A ∧ ((ES:HL) + byte)	×		
		A, [HL+B]	2	1	4	A ← A ∧ (HL + B)	×		
		A, ES:[HL+B]	3	2	5	A ← A ∧ ((ES:HL) + B)	×		
		A, [HL+C]	2	1	4	A ← A ∧ (HL + C)	×		
		A, ES:[HL+C]	3	2	5	A ← A ∧ ((ES:HL) + C)	×		

Note 1. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the internal RAM area, SFR area, or 2nd SFR area, or when the instruction does not involve data access.

Note 2. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the code flash memory area, or of access to the data flash memory area by an 8-bit instruction.

Note 3. Except when r = A.

Remark The listed numbers of clock cycles are for when the program is present in the internal ROM (flash memory) area. If the instruction is fetched from the internal RAM area, the number becomes up to double the listed number plus 3 clock cycles.

Table 27 - 5 Operation List (9/18)

Instruction Group	Mnemonic	Operands	Bytes	Clock Cycles		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit operation	OR	A, #byte	2	1	—	A ← A ∨ byte	×		
		saddr, #byte	3	2	—	(saddr) ← (saddr) ∨ byte	×		
		A, r Note 3	2	1	—	A ← A ∨ r	×		
		r, A	2	1	—	r ← r ∨ A	×		
		A, !addr16	3	1	4	A ← A ∨ (addr16)	×		
		A, ES:!addr16	4	2	5	A ← A ∨ (ES:addr16)	×		
		A, saddr	2	1	—	A ← A ∨ (saddr)	×		
		A, [HL]	1	1	4	A ← A ∨ (HL)	×		
		A, ES:[HL]	2	2	5	A ← A ∨ (ES:HL)	×		
		A, [HL+byte]	2	1	4	A ← A ∨ (HL + byte)	×		
		A, ES:[HL+byte]	3	2	5	A ← A ∨ ((ES:HL) + byte)	×		
		A, [HL+B]	2	1	4	A ← A ∨ (HL + B)	×		
		A, ES:[HL+B]	3	2	5	A ← A ∨ ((ES:HL) + B)	×		
		A, [HL+C]	2	1	4	A ← A ∨ (HL + C)	×		
		A, ES:[HL+C]	3	2	5	A ← A ∨ ((ES:HL) + C)	×		
	XOR	A, #byte	2	1	—	A ← A ⩛ byte	×		
		saddr, #byte	3	2	—	(saddr) ← (saddr) ⩛ byte	×		
		A, r Note 3	2	1	—	A ← A ⩛ r	×		
		r, A	2	1	—	r ← r ⩛ A	×		
		A, !addr16	3	1	4	A ← A ⩛ (addr16)	×		
		A, ES:!addr16	4	2	5	A ← A ⩛ (ES:addr16)	×		
		A, saddr	2	1	—	A ← A ⩛ (saddr)	×		
		A, [HL]	1	1	4	A ← A ⩛ (HL)	×		
		A, ES:[HL]	2	2	5	A ← A ⩛ (ES:HL)	×		
		A, [HL+byte]	2	1	4	A ← A ⩛ (HL + byte)	×		
		A, ES:[HL+byte]	3	2	5	A ← A ⩛ ((ES:HL) + byte)	×		
		A, [HL+B]	2	1	4	A ← A ⩛ (HL + B)	×		
		A, ES:[HL+B]	3	2	5	A ← A ⩛ ((ES:HL) + B)	×		
		A, [HL+C]	2	1	4	A ← A ⩛ (HL + C)	×		
		A, ES:[HL+C]	3	2	5	A ← A ⩛ ((ES:HL) + C)	×		

Note 1. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the internal RAM area, SFR area, or 2nd SFR area, or when the instruction does not involve data access.

Note 2. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the code flash memory area, or of access to the data flash memory area by an 8-bit instruction.

Note 3. Except when r = A.

Remark The listed numbers of clock cycles are for when the program is present in the internal ROM (flash memory) area. If the instruction is fetched from the internal RAM area, the number becomes up to double the listed number plus 3 clock cycles.

Table 27 - 5 Operation List (10/18)

Instruction Group	Mnemonic	Operands	Bytes	Clock Cycles		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit operation	CMP	A, #byte	2	1	—	A - byte	×	×	×
		!addr16, #byte	4	1	4	(addr16) - byte	×	×	×
		ES:!addr16, #byte	5	2	5	(ES:addr16) - byte	×	×	×
		saddr, #byte	3	1	—	(saddr) - byte	×	×	×
		A, r Note 3	2	1	—	A - r	×	×	×
		r, A	2	1	—	r - A	×	×	×
		A, !addr16	3	1	4	A - (addr16)	×	×	×
		A, ES:!addr16	4	2	5	A - (ES:addr16)	×	×	×
		A, saddr	2	1	—	A - (saddr)	×	×	×
		A, [HL]	1	1	4	A - (HL)	×	×	×
		A, ES:[HL]	2	2	5	A - (ES:HL)	×	×	×
		A, [HL+byte]	2	1	4	A - (HL + byte)	×	×	×
		A, ES:[HL+byte]	3	2	5	A - ((ES:HL) + byte)	×	×	×
		A, [HL+B]	2	1	4	A - (HL + B)	×	×	×
		A, ES:[HL+B]	3	2	5	A - ((ES:HL) + B)	×	×	×
		A, [HL+C]	2	1	4	A - (HL + C)	×	×	×
		A, ES:[HL+C]	3	2	5	A - ((ES:HL) + C)	×	×	×
	CMP0	A	1	1	—	A - 00H	×	0	0
		X	1	1	—	X - 00H	×	0	0
		B	1	1	—	B - 00H	×	0	0
		C	1	1	—	C - 00H	×	0	0
		!addr16	3	1	4	(addr16) - 00H	×	0	0
		ES:!addr16	4	2	5	(ES:addr16) - 00H	×	0	0
		saddr	2	1	—	(saddr) - 00H	×	0	0
	CMPS	X, [HL+byte]	3	1	4	X - (HL + byte)	×	×	×
		X, ES:[HL+byte]	4	2	5	X - ((ES:HL) + byte)	×	×	×

Note 1. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the internal RAM area, SFR area, or 2nd SFR area, or when the instruction does not involve data access.

Note 2. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the code flash memory area, or of access to the data flash memory area by an 8-bit instruction.

Note 3. Except when r = A.

Remark The listed numbers of clock cycles are for when the program is present in the internal ROM (flash memory) area. If the instruction is fetched from the internal RAM area, the number becomes up to double the listed number plus 3 clock cycles.

Table 27 - 5 Operation List (11/18)

Instruction Group	Mnemonic	Operands	Bytes	Clock Cycles		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
16-bit operation	ADDW	AX, #word	3	1	—	AX, CY ← AX + word	×	×	×
		AX, AX	1	1	—	AX, CY ← AX + AX	×	×	×
		AX, BC	1	1	—	AX, CY ← AX + BC	×	×	×
		AX, DE	1	1	—	AX, CY ← AX + DE	×	×	×
		AX, HL	1	1	—	AX, CY ← AX + HL	×	×	×
		AX, !addr16	3	1	4	AX, CY ← AX + (addr16)	×	×	×
		AX, ES:!addr16	4	2	5	AX, CY ← AX + (ES:addr16)	×	×	×
		AX, saddrp	2	1	—	AX, CY ← AX + (saddrp)	×	×	×
		AX, [HL+byte]	3	1	4	AX, CY ← AX + (HL + byte)	×	×	×
		AX, ES:[HL+byte]	4	2	5	AX, CY ← AX + ((ES:HL) + byte)	×	×	×
	SUBW	AX, #word	3	1	—	AX, CY ← AX - word	×	×	×
		AX, BC	1	1	—	AX, CY ← AX - BC	×	×	×
		AX, DE	1	1	—	AX, CY ← AX - DE	×	×	×
		AX, HL	1	1	—	AX, CY ← AX - HL	×	×	×
		AX, !addr16	3	1	4	AX, CY ← AX - (addr16)	×	×	×
		AX, ES:!addr16	4	2	5	AX, CY ← AX - (ES:addr16)	×	×	×
		AX, saddrp	2	1	—	AX, CY ← AX - (saddrp)	×	×	×
		AX, [HL+byte]	3	1	4	AX, CY ← AX - (HL + byte)	×	×	×
		AX, ES:[HL+byte]	4	2	5	AX, CY ← AX - ((ES:HL) + byte)	×	×	×
	CMPW	AX, #word	3	1	—	AX - word	×	×	×
		AX, BC	1	1	—	AX - BC	×	×	×
		AX, DE	1	1	—	AX - DE	×	×	×
		AX, HL	1	1	—	AX - HL	×	×	×
		AX, !addr16	3	1	4	AX - (addr16)	×	×	×
		AX, ES:!addr16	4	2	5	AX - (ES:addr16)	×	×	×
		AX, saddrp	2	1	—	AX - (saddrp)	×	×	×
		AX, [HL+byte]	3	1	4	AX - (HL + byte)	×	×	×
		AX, ES:[HL+byte]	4	2	5	AX - ((ES:HL) + byte)	×	×	×

Note 1. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the internal RAM area, SFR area, or 2nd SFR area, or when the instruction does not involve data access.

Note 2. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the code flash memory area, or of access to the data flash memory area by an 8-bit instruction.

Remark The listed numbers of clock cycles are for when the program is present in the internal ROM (flash memory) area. If the instruction is fetched from the internal RAM area, the number becomes up to double the listed number plus 3 clock cycles.

Table 27 - 5 Operation List (12/18)

Instruction Group	Mnemonic	Operands	Bytes	Clock Cycles		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
Multiply, Divide, Multiply & accumulate	MULU	X	1	1	—	AX ← A × X			
	MULHU		3	2	—	BCAX ← AX × BC (unsigned)			
	MULH		3	2	—	BCAX ← AX × BC (signed)			
	DIVHU		3	9	—	AX (quotient), DE (remainder) ← AX ÷ DE (unsigned)			
	DIVWU		3	17	—	BCAX (quotient), HLDE (remainder) ← BCAX ÷ HLDE (unsigned)			
	MACHU		3	3	—	MACR ← MACR + AX × BC (unsigned)		×	×
	MACH		3	3	—	MACR ← MACR + AX × BC (signed)		×	×

Note 1. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the internal RAM area, SFR area, or 2nd SFR area, or when the instruction does not involve data access.

Note 2. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the code flash memory area, or of access to the data flash memory area by an 8-bit instruction.

Caution Disable interrupts when executing the DIVHU or DIVWU instruction in an interrupt servicing routine. Alternatively, unless they are executed in the RAM area, note that execution of a DIVHU or DIVWU instruction is possible even with interrupts enabled as long as a NOP instruction is added immediately after the DIVHU or DIVWU instruction in the assembly language source code. The following compilers automatically add a NOP instruction immediately after any DIVHU or DIVWU instruction output during the build process.

- **V. 1.01.00 and later versions of CC-RL (Renesas Electronics compiler), for both C and assembly language source code**
- **Service pack 1.40.3 and later versions of the EWRL78 (IAR compiler), for C language source code**
- **LLVM RL78 (CyberTHOR compiler), for both C and C++ language source code**
- **GNURL78 (CyberTHOR compiler), for both C and C++ language source code**

Remark 1. The listed numbers of clock cycles are for when the program is present in the internal ROM (flash memory) area. If the instruction is fetched from the internal RAM area, the number becomes up to double the listed number plus 3 clock cycles.

Remark 2. MACR indicates the multiplication and accumulation register (MACRH, MACRL).

Table 27 - 5 Operation List (13/18)

Instruction Group	Mnemonic	Operands	Bytes	Clock Cycles		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
Increment/ decrement	INC	r	1	1	—	r ← r + 1	×	×	
		!addr16	3	2	—	(addr16) ← (addr16) + 1	×	×	
		ES:!addr16	4	3	—	(ES, addr16) ← (ES, addr16) + 1	×	×	
		saddr	2	2	—	(saddr) ← (saddr) + 1	×	×	
		[HL+byte]	3	2	—	(HL + byte) ← (HL + byte) + 1	×	×	
		ES:[HL+byte]	4	3	—	((ES:HL) + byte) ← ((ES:HL) + byte) + 1	×	×	
	DEC	r	1	1	—	r ← r - 1	×	×	
		!addr16	3	2	—	(addr16) ← (addr16) - 1	×	×	
		ES:!addr16	4	3	—	(ES, addr16) ← (ES, addr16) - 1	×	×	
		saddr	2	2	—	(saddr) ← (saddr) - 1	×	×	
		[HL+byte]	3	2	—	(HL + byte) ← (HL + byte) - 1	×	×	
		ES:[HL+byte]	4	3	—	((ES:HL) + byte) ← ((ES:HL) + byte) - 1	×	×	
	INCW	rp	1	1	—	rp ← rp + 1			
		!addr16	3	2	—	(addr16) ← (addr16) + 1			
		ES:!addr16	4	3	—	(ES, addr16) ← (ES, addr16) + 1			
		saddrp	2	2	—	(saddrp) ← (saddrp) + 1			
		[HL+byte]	3	2	—	(HL + byte) ← (HL + byte) + 1			
		ES:[HL+byte]	4	3	—	((ES:HL) + byte) ← ((ES:HL) + byte) + 1			
	DECW	rp	1	1	—	rp ← rp - 1			
		!addr16	3	2	—	(addr16) ← (addr16) - 1			
		ES:!addr16	4	3	—	(ES, addr16) ← (ES, addr16) - 1			
		saddrp	2	2	—	(saddrp) ← (saddrp) - 1			
		[HL+byte]	3	2	—	(HL + byte) ← (HL + byte) - 1			
		ES:[HL+byte]	4	3	—	((ES:HL) + byte) ← ((ES:HL) + byte) - 1			
Shift	SHR	A, cnt	2	1	—	(CY ← A_0, A_{m-1} ← A_m, A_7 ← 0) × cnt			×
	SHRW	AX, cnt	2	1	—	(CY ← AX_0, AX_{m-1} ← AX_m, AX_{15} ← 0) × cnt			×
	SHL	A, cnt	2	1	—	(CY ← A_7, A_m ← A_{m-1}, A_0 ← 0) × cnt			×
		B, cnt	2	1	—	(CY ← B_7, B_m ← B_{m-1}, B_0 ← 0) × cnt			×
		C, cnt	2	1	—	(CY ← C_7, C_m ← C_{m-1}, C_0 ← 0) × cnt			×
	SHLW	AX, cnt	2	1	—	(CY ← AX_{15}, AX_m ← AX_{m-1}, AX_0 ← 0) × cnt			×
		BC, cnt	2	1	—	(CY ← BC_{15}, BC_m ← BC_{m-1}, BC_0 ← 0) × cnt			×
	SAR	A, cnt	2	1	—	(CY ← A_0, A_{m-1} ← A_m, A_7 ← A_7) × cnt			×
	SARW	AX, cnt	2	1	—	(CY ← AX_0, AX_{m-1} ← AX_m, AX_{15} ← AX_{15}) × cnt			×

Note 1. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the internal RAM area, SFR area, or 2nd SFR area, or when the instruction does not involve data access.

Note 2. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the code flash memory area, or of access to the data flash memory area by an 8-bit instruction.

Remark 1. The listed numbers of clock cycles are for when the program is present in the internal ROM (flash memory) area. If the instruction is fetched from the internal RAM area, the number becomes up to double the listed number plus 3 clock cycles.

Remark 2. cnt indicates the bit shift count.

Table 27 - 5 Operation List (14/18)

Instruction Group	Mnemonic	Operands	Bytes	Clock Cycles		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
Rotate	ROR	A, 1	2	1	—	(CY, $A_7 \leftarrow A_0$, $A_{m-1} \leftarrow A_m$) × 1			×
	ROL	A, 1	2	1	—	(CY, $A_0 \leftarrow A_7$, $A_{m+1} \leftarrow A_m$) × 1			×
	RORC	A, 1	2	1	—	(CY $\leftarrow A_0$, $A_7 \leftarrow$ CY, $A_{m-1} \leftarrow A_m$) × 1			×
	ROLC	A, 1	2	1	—	(CY $\leftarrow A_7$, $A_0 \leftarrow$ CY, $A_{m+1} \leftarrow A_m$) × 1			×
	ROLWC	AX, 1	2	1	—	(CY $\leftarrow AX_{15}$, $AX_0 \leftarrow$ CY, $AX_{m+1} \leftarrow AX_m$) × 1			×
		BC, 1	2	1	—	(CY $\leftarrow BC_{15}$, $BC_0 \leftarrow$ CY, $BC_{m+1} \leftarrow BC_m$) × 1			×
Bit manipulate	MOV1	CY, A.bit	2	1	—	CY ← A.bit			×
		A.bit, CY	2	1	—	A.bit ← CY			
		CY, PSW.bit	3	1	—	CY ← PSW.bit			×
		PSW.bit, CY	3	4	—	PSW.bit ← CY	×	×	
		CY, saddr.bit	3	1	—	CY ← (saddr).bit			×
		saddr.bit, CY	3	2	—	(saddr).bit ← CY			
		CY, sfr.bit	3	1	—	CY ← sfr.bit			×
		sfr.bit, CY	3	2	—	sfr.bit ← CY			
		CY, [HL].bit	2	1	4	CY ← (HL).bit			×
		[HL].bit, CY	2	2	—	(HL).bit ← CY			
		CY, ES:[HL].bit	3	2	5	CY ← (ES, HL).bit			×
		ES:[HL].bit, CY	3	3	—	(ES, HL).bit ← CY			
	AND1	CY, A.bit	2	1	—	CY ← CY ∧ A.bit			×
		CY, PSW.bit	3	1	—	CY ← CY ∧ PSW.bit			×
		CY, saddr.bit	3	1	—	CY ← CY ∧ (saddr).bit			×
		CY, sfr.bit	3	1	—	CY ← CY ∧ sfr.bit			×
		CY, [HL].bit	2	1	4	CY ← CY ∧ (HL).bit			×
		CY, ES:[HL].bit	3	2	5	CY ← CY ∧ (ES, HL).bit			×
	OR1	CY, A.bit	2	1	—	CY ← CY ∨ A.bit			×
		CY, PSW.bit	3	1	—	CY ← CY ∨ PSW.bit			×
		CY, saddr.bit	3	1	—	CY ← CY ∨ (saddr).bit			×
		CY, sfr.bit	3	1	—	CY ← CY ∨ sfr.bit			×
		CY, [HL].bit	2	1	4	CY ← CY ∨ (HL).bit			×
		CY, ES:[HL].bit	3	2	5	CY ← CY ∨ (ES, HL).bit			×

Note 1. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the internal RAM area, SFR area, or 2nd SFR area, or when the instruction does not involve data access.

Note 2. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the code flash memory area, or of access to the data flash memory area by an 8-bit instruction.

Remark The listed numbers of clock cycles are for when the program is present in the internal ROM (flash memory) area. If the instruction is fetched from the internal RAM area, the number becomes up to double the listed number plus 3 clock cycles.

Table 27 - 5 Operation List (15/18)

Instruction Group	Mnemonic	Operands	Bytes	Clock Cycles		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
Bit manipulate	XOR1	CY, A.bit	2	1	—	CY ← CY ⨯ A.bit			×
		CY, PSW.bit	3	1	—	CY ← CY ⨯ PSW.bit			×
		CY, saddr.bit	3	1	—	CY ← CY ⨯ (saddr).bit			×
		CY, sfr.bit	3	1	—	CY ← CY ⨯ sfr.bit			×
		CY, [HL].bit	2	1	4	CY ← CY ⨯ (HL).bit			×
		CY, ES:[HL].bit	3	2	5	CY ← CY ⨯ (ES, HL).bit			×
	SET1	A.bit	2	1	—	A.bit ← 1			
		PSW.bit	3	4	—	PSW.bit ← 1	×	×	×
		!addr16.bit	4	2	—	(addr16).bit ← 1			
		ES:!addr16.bit	5	3	—	(ES, addr16).bit ← 1			
		saddr.bit	3	2	—	(saddr).bit ← 1			
		sfr.bit	3	2	—	sfr.bit ← 1			
		[HL].bit	2	2	—	(HL).bit ← 1			
		ES:[HL].bit	3	3	—	(ES, HL).bit ← 1			
	CLR1	A.bit	2	1	—	A.bit ← 0			
		PSW.bit	3	4	—	PSW.bit ← 0	×	×	×
		!addr16.bit	4	2	—	(addr16).bit ← 0			
		ES:!addr16.bit	5	3	—	(ES, addr16).bit ← 0			
		saddr.bit	3	2	—	(saddr).bit ← 0			
		sfr.bit	3	2	—	sfr.bit ← 0			
		[HL].bit	2	2	—	(HL).bit ← 0			
		ES:[HL].bit	3	3	—	(ES, HL).bit ← 0			
	SET1	CY	2	1	—	CY ← 1			1
	CLR1	CY	2	1	—	CY ← 0			0
	NOT1	CY	2	1	—	CY ← $\overline{CY}$			×

Note 1. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the internal RAM area, SFR area, or 2nd SFR area, or when the instruction does not involve data access.

Note 2. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the code flash memory area, or of access to the data flash memory area by an 8-bit instruction.

Remark The listed numbers of clock cycles are for when the program is present in the internal ROM (flash memory) area. If the instruction is fetched from the internal RAM area, the number becomes up to double the listed number plus 3 clock cycles.

Table 27 - 5 Operation List (16/18)

Instruction Group	Mnemonic	Operands	Bytes	Clock Cycles		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
Call/return	CALL	rp	2	3	—	(SP - 2) ← (PC + 2)S, (SP - 3) ← (PC + 2)H, (SP - 4) ← (PC + 2)L, PC ← CS, rp, SP ← SP - 4			
		$!addr20	3	3	—	(SP - 2) ← (PC + 3)S, (SP - 3) ← (PC + 3)H, (SP - 4) ← (PC + 3)L, PC ← PC + 3 + jdisp16, SP ← SP - 4			
		!addr16	3	3	—	(SP - 2) ← (PC + 3)S, (SP - 3) ← (PC + 3)H, (SP - 4) ← (PC + 3)L, PC ← 0000, addr16, SP ← SP - 4			
		!!addr20	4	3	—	(SP - 2) ← (PC + 4)S, (SP - 3) ← (PC + 4)H, (SP - 4) ← (PC + 4)L, PC ← addr20, SP ← SP - 4			
	CALLT	[addr5]	2	5	—	(SP - 2) ← (PC + 2)S, (SP - 3) ← (PC + 2)H, (SP - 4) ← (PC + 2)L , PCS ← 0000, PCH ← (0000, addr5 + 1), PCL ← (0000, addr5), SP ← SP - 4			
	BRK	—	2	5	—	(SP - 1) ← PSW, (SP - 2) ← (PC + 2)S, (SP - 3) ← (PC + 2)H, (SP - 4) ← (PC + 2)L, PCS ← 0000, PCH ← (0007FH), PCL ← (0007EH), SP ← SP - 4, IE ← 0			
	RET	—	1	6	—	PCL ← (SP), PCH ← (SP + 1), PCS ← (SP + 2), SP ← SP + 4			
	RETI	—	2	6	—	PCL ← (SP), PCH ← (SP + 1), PCS ← (SP + 2), PSW ← (SP + 3), SP ← SP + 4	R	R	R
	RETB	—	2	6	—	PCL ← (SP), PCH ← (SP + 1), PCS ← (SP + 2), PSW ← (SP + 3), SP ← SP + 4	R	R	R

Note 1. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the internal RAM area, SFR area, or 2nd SFR area, or when the instruction does not involve data access.

Note 2. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the code flash memory area, or of access to the data flash memory area by an 8-bit instruction.

Remark The listed numbers of clock cycles are for when the program is present in the internal ROM (flash memory) area. If the instruction is fetched from the internal RAM area, the number becomes up to double the listed number plus 3 clock cycles.

Table 27 - 5 Operation List (17/18)

Instruction Group	Mnemonic	Operands	Bytes	Clock Cycles		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
Stack manipulate	PUSH	PSW	2	1	—	(SP - 1) ← PSW, (SP - 2) ← 00H, SP ← SP - 2			
		rp	1	1	—	(SP - 1) ← rp_H, (SP - 2) ← rp_L, SP ← SP - 2			
	POP	PSW	2	3	—	PSW ← (SP + 1), SP ← SP + 2	R	R	R
		rp	1	1	—	rpL ← (SP), rp_H ← (SP + 1), SP ← SP + 2			
	MOVW	SP, #word	4	1	—	SP ← word			
		SP, AX	2	1	—	SP ← AX			
		AX, SP	2	1	—	AX ← SP			
		HL, SP	3	1	—	HL ← SP			
		BC, SP	3	1	—	BC ← SP			
		DE, SP	3	1	—	DE ← SP			
	ADDW	SP, #byte	2	1	—	SP ← SP + byte			
	SUBW	SP, #byte	2	1	—	SP ← SP - byte			
Unconditional branch	BR	AX	2	3	—	PC ← CS, AX			
		$addr20	2	3	—	PC ← PC + 2 + jdisp8			
		$!addr20	3	3	—	PC ← PC + 3 + jdisp16			
		!addr16	3	3	—	PC ← 0000, addr16			
		!!addr20	4	3	—	PC ← addr20			
Conditional branch	BC	$addr20	2	2/4Note 3	—	PC ← PC + 2 + jdisp8 if CY = 1			
	BNC	$addr20	2	2/4Note 3	—	PC ← PC + 2 + jdisp8 if CY = 0			
	BZ	$addr20	2	2/4Note 3	—	PC ← PC + 2 + jdisp8 if Z = 1			
	BNZ	$addr20	2	2/4Note 3	—	PC ← PC + 2 + jdisp8 if Z = 0			
	BH	$addr20	3	2/4Note 3	—	PC ← PC + 3 + jdisp8 if (Z ∨ CY) = 0			
	BNH	$addr20	3	2/4Note 3	—	PC ← PC + 3 + jdisp8 if (Z ∨ CY) = 1			
	BT	saddr.bit, $addr20	4	3/5Note 3	—	PC ← PC + 4 + jdisp8 if (saddr).bit = 1			
		sfr.bit, $addr20	4	3/5Note 3	—	PC ← PC + 4 + jdisp8 if sfr.bit = 1			
		A.bit, $addr20	3	3/5Note 3	—	PC ← PC + 3 + jdisp8 if A.bit = 1			
		PSW.bit, $addr20	4	3/5Note 3	—	PC ← PC + 4 + jdisp8 if PSW.bit = 1			
		[HL].bit, $addr20	3	3/5Note 3	6/7	PC ← PC + 3 + jdisp8 if (HL).bit = 1			
		ES:[HL].bit, $addr20	4	4/6Note 3	7/8	PC ← PC + 4 + jdisp8 if (ES, HL).bit = 1			

Note 1. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the internal RAM area, SFR area, or 2nd SFR area, or when the instruction does not involve data access.

Note 2. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the code flash memory area, or of access to the data flash memory area by an 8-bit instruction.

Note 3. These indicate the numbers of clock cycles when the condition is and is not met.

Remark The listed numbers of clock cycles are for when the program is present in the internal ROM (flash memory) area. If the instruction is fetched from the internal RAM area, the number becomes up to double the listed number plus 3 clock cycles.

Table 27 - 5 Operation List (18/18)

Instruction Group	Mnemonic	Operands	Bytes	Clock Cycles		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
Conditional branch	BF	saddr.bit, $addr20	4	3/5Note 3	—	PC ← PC + 4 + jdisp8 if (saddr).bit = 0			
		sfr.bit, $addr20	4	3/5Note 3	—	PC ← PC + 4 + jdisp8 if sfr.bit = 0			
		A.bit, $addr20	3	3/5Note 3	—	PC ← PC + 3 + jdisp8 if A.bit = 0			
		PSW.bit, $addr20	4	3/5Note 3	—	PC ← PC + 4 + jdisp8 if PSW.bit = 0			
		[HL].bit, $addr20	3	3/5Note 3	6/7	PC ← PC + 3 + jdisp8 if (HL).bit = 0			
		ES:[HL].bit, $addr20	4	4/6Note 3	7/8	PC ← PC + 4 + jdisp8 if (ES, HL).bit = 0			
	BTCLR	saddr.bit, $addr20	4	3/5Note 3	—	PC ← PC + 4 + jdisp8 if (saddr).bit = 1 then reset (saddr).bit			
		sfr.bit, $addr20	4	3/5Note 3	—	PC ← PC + 4 + jdisp8 if sfr.bit = 1 then reset sfr.bit			
		A.bit, $addr20	3	3/5Note 3	—	PC ← PC + 3 + jdisp8 if A.bit = 1 then reset A.bit			
		PSW.bit, $addr20	4	3/5Note 3	—	PC ← PC + 4 + jdisp8 if PSW.bit = 1 then reset PSW.bit	×	×	×
		[HL].bit, $addr20	3	3/5Note 3	—	PC ← PC + 3 + jdisp8 if (HL).bit = 1 then reset (HL).bit			
		ES:[HL].bit, $addr20	4	4/6Note 3	—	PC ← PC + 4 + jdisp8 if (ES, HL).bit = 1 then reset (ES, HL).bit			
Conditional skip	SKC	—	2	1	—	Next instruction skip if CY = 1			
	SKNC	—	2	1	—	Next instruction skip if CY = 0			
	SKZ	—	2	1	—	Next instruction skip if Z = 1			
	SKNZ	—	2	1	—	Next instruction skip if Z = 0			
	SKH	—	2	1	—	Next instruction skip if (Z ∨ CY) = 0			
	SKNH	—	2	1	—	Next instruction skip if (Z ∨ CY) = 1			
CPU control	SEL	RBnNote 4	2	1	—	RBS1, RBS0 ← n			
	NOP	—	1	1	—	No Operation			
	EI	—	3	4	—	IE ← 1 (Enable Interrupt)			
	DI	—	3	4	—	IE ← 0 (Disable Interrupt)			
	HALT	—	2	3	—	Set HALT Mode			
	STOP	—	2	3	—	Set STOP Mode			

Note 1. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the internal RAM area, SFR area, or 2nd SFR area, or when the instruction does not involve data access.

Note 2. These are the numbers of CPU clock cycles (f_{CLK}) in the case of access to the code flash memory area, or of access to the data flash memory area by an 8-bit instruction.

Note 3. These indicate the numbers of clock cycles when the condition is and is not met.

Note 4. n indicates the number of register banks (n = 0 to 3).

Remark The listed numbers of clock cycles are for when the program is present in the internal ROM (flash memory) area. If the instruction is fetched from the internal RAM area, the number becomes up to double the listed number plus 3 clock cycles.

Section 28 Electrical Characteristics

This section describes the electrical characteristics of the following products.

- 3Z: Industrial and consumer applications, T_A = -40 to +105°C
  R7F103Gxx3Zxx

Caution 1. This MCU has on-chip debugging functionality for use in the development and evaluation of user systems. Do not use on-chip debugging with products designated as part of mass production, because using this function may cause the guaranteed number of times the flash memory is rewritten to be exceeded, and product reliability therefore cannot be guaranteed. Renesas Electronics is not liable for problems occurring when on-chip debugging is used with products designated as part of mass production.

Caution 2. For products that do not have an EV_{DD0} or EV_{SS0} pin, read EV_{DD0} as V_{DD}, and EV_{SS0} as V_{SS}.

Caution 3. The present pins differ depending on the products. For details, see section 2.1 Functions of Port Pins through section 2.2.1 Functions for each product.

28.1 Absolute Maximum Ratings

(1/2)

Item	Symbols	Conditions	Ratings	Unit
Supply voltage	V_{DD}		-0.5 to +6.5	V
	EV_{DD0}		-0.5 to +6.5	V
	EV_{SS0}		-0.5 to +0.3	V
REGC pin input voltage	V_{IREGC}	REGC	-0.3 to +2.1 and -0.3 to V_{DD} + 0.3 **Note 1**	V
Input voltage	V_{I1}	P00 to P06, P10 to P17, P30, P31, P33, P34, P40 to P43, P50 to P55, P70 to P77, P120, P140, P141, P146, P147	-0.3 to EV_{DD0} + 0.3 and -0.3 to V_{DD} + 0.3 **Note 2**	V
	V_{I2}	P60, P61 (N-ch open-drain)	-0.3 to +6.5	V
	V_{I3}	P20 to P27, P121 to P125, P137, EXCLK, $\overline{\text{RESET}}$	-0.3 to V_{DD} + 0.3 **Note 2**	V
Output voltage	V_{O1}	P00 to P06, P10 to P17, P30, P31, P33, P34, P40 to P43, P50 to P55, P70 to P77, P120, P130, P140, P141, P146, P147	-0.3 to EV_{DD0} + 0.3 and -0.3 to V_{DD} + 0.3 **Note 2**	V
	V_{O2}	P20 to P27, P121 to P125	-0.3 to V_{DD} + 0.3 **Note 2**	V
Analog input voltage	V_{AI1}	ANI16 to ANI19	-0.3 to EV_{DD0} + 0.3 and -0.3 to AV_{REFP} + 0.3 **Notes 2, 3**	V
	V_{AI2}	ANI0 to ANI7	-0.3 to V_{DD} + 0.3 and -0.3 to AV_{REFP} + 0.3 **Notes 2, 3**	V

Note 1. Connect the REGC pin to V_{SS} via a capacitor (0.47 to 1 μF). The listed value is the absolute maximum rating of the REGC pin. Only use the capacitor connection. Do not apply a specific voltage to this pin.

Note 2. This voltage must be no higher than 6.5 V.

Note 3. The voltage on a pin in use for A/D conversion must not exceed AV_{REFP} + 0.3.

Caution **Product quality may suffer if the absolute maximum rating is exceeded even momentarily for any parameter. That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore the product must be used under conditions that ensure that the absolute maximum ratings are not exceeded.**

Remark 1. The characteristics of functions multiplexed on a given pin are the same as those for the port pin unless otherwise specified.

Remark 2. AV_{REFP} refers to the positive reference voltage of the A/D converter.

Remark 3. The reference voltage is V_{SS}.

(2/2)

Item	Symbols	Conditions			Ratings	Unit
High-level output current	I_{OH1}	Per pin	P00 to P06, P10 to P17, P30, P31, P33, P34, P40 to P43, P50 to P55, P70 to P77, P120, P130, P140, P141, P146, P147		-40	mA
		Total of all pins -170 mA	P00 to P04, P40 to P43, P120, P130, P140, P141		-70	mA
			P05, P06, P10 to P17, P30, P31, P33, P34, P50 to P55, P70 to P77, P146, P147		-100	mA
	I_{OH2}	Per pin	P20 to P27, P121 to P125		-5	mA
		Total of all pins			-20	mA
Low-level output current	I_{OL1}	Per pin	P00 to P06, P10 to P17, P30, P31, P33, P34, P40 to P43, P50 to P55, P60, P61, P70 to P77, P120, P130, P140, P141, P146, P147		40	mA
		Total of all pins 170 mA	P00 to P04, P40 to P43, P120, P130, P140, P141		70	mA
			P05, P06, P10 to P17, P30, P31, P33, P34, P50 to P55, P60, P61, P70 to P77, P146, P147		100	mA
	I_{OL2}	Per pin	P20 to P27, P121 to P125		10	mA
		Total of all pins			20	mA
Ambient operating temperature	T_A	In normal operation mode		3Z: Industrial and consumer applications	-40 to +105	°C
		In flash memory programming mode		3Z: Industrial and consumer applications	-40 to +105	
Storage temperature	T_{stg}				-65 to +150	°C

Caution **Product quality may suffer if the absolute maximum rating is exceeded even momentarily for any parameter. That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore the product must be used under conditions that ensure that the absolute maximum ratings are not exceeded.**

Remark The characteristics of functions multiplexed on a given pin are the same as those for the port pin unless otherwise specified.

28.2 Characteristics of the Oscillators

28.2.1 Characteristics of the X1 oscillator

(T_A = -40 to +105°C, 1.6 V ≤ V_{DD} ≤ 5.5 V, V_{SS} = 0 V)

Item	Resonator	Conditions	Min.	Typ.	Max.	Unit
X1 clock oscillation allowable input cycle time**Note**	Ceramic resonator/ crystal resonator		0.05		1	μs

Note The listed time and frequency indicate permissible ranges of the oscillator. For actual applications, request evaluation by the manufacturer of the oscillator circuit mounted on a board so you can use appropriate values. Refer to **28.4 AC Characteristics** for instruction execution time.

Caution Since the CPU is started by the high-speed on-chip oscillator clock after release from the reset state, the user should use the oscillation stabilization time counter status register (OSTC) to check the X1 clock oscillation stabilization time. Specify the values for the oscillation stabilization time in the OSTC register and the oscillation stabilization time select register (OSTS) after having sufficiently evaluated the oscillation stabilization time with the resonator to be used.

28.2.2 Characteristics of the on-chip oscillators

(T_A = -40 to +105°C, 1.6 V ≤ V_{DD} ≤ 5.5 V, V_{SS} = 0 V)

Item	Symbol	Conditions		Min.	Typ.	Max.	Unit
High-speed on-chip oscillator clock frequency	f_{IH}			1		32	MHz
<R> High-speed on-chip oscillator clock frequency accuracy (HIPREC = 1)		-40 to +105°C	1.6 V ≤ V_{DD} ≤ 5.5 V	-1		+1	%
High-speed on-chip oscillator clock correction resolution					0.05		%
Low-speed on-chip oscillator clock frequency**Note**	f_{IL}				32.768		kHz
Low-speed on-chip oscillator clock frequency accuracy				-15		+15	%

Note The listed values only indicate the characteristics of the oscillators. Refer to **28.4 AC Characteristics** for instruction execution time.

28.3 DC Characteristics

28.3.1 Characteristics of pins

(T_A = -40 to +105°C, 1.6 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (1/6)

Item	Symbol	Conditions		Min.	Typ.	Max.	Unit
Allowable high-level output current**Note 1**	I_{OH1}	Per pin for P00 to P06, P10 to P17, P30, P31, P33, P34, P40 to P43, P50 to P55, P70 to P77, P120, P130, P140, P141, P146, P147	1.6 V ≤ EV_{DD0} ≤ 5.5 V			-10.0 **Note 2**	mA
		Total of P00 to P04, P40 to P43, P120, P130, P140, P141 (when duty ≤ 70%**Note 3**)	4.0 V ≤ EV_{DD0} ≤ 5.5 V			-55.0 **Note 4**	mA
			2.7 V ≤ EV_{DD0} < 4.0 V			-10.0	mA
			1.8 V ≤ EV_{DD0} < 2.7 V			-5.0	mA
			1.6 V ≤ EV_{DD0} < 1.8 V			-2.5	mA
		Total of P05, P06, P10 to P17, P30, P31, P33, P34, P50 to P55, P70 to P77, P146, P147 (when duty ≤ 70%**Note 3**)	4.0 V ≤ EV_{DD0} ≤ 5.5 V			-80.0 **Note 5**	mA
			2.7 V ≤ EV_{DD0} < 4.0 V			-19.0	mA
			1.8 V ≤ EV_{DD0} < 2.7 V			-10.0	mA
			1.6 V ≤ EV_{DD0} < 1.8 V			-5.0	mA
		Total of all pins (when duty ≤ 70%**Note 3**)	1.6 V ≤ EV_{DD0} ≤ 5.5 V			-100.0 **Note 6**	mA
	I_{OH2}	Per pin for P20 to P27, P121 to P125	4.0 V ≤ V_{DD} ≤ 5.5 V			-3.0 **Note 2**	mA
			2.7 V ≤ V_{DD} < 4.0 V			-1.0 **Note 2**	mA
			1.8 V ≤ V_{DD} < 2.7 V			-1.0 **Note 2**	mA
			1.6 V ≤ V_{DD} < 1.8 V			-0.5 **Note 2**	mA
		Total of all pins (when duty ≤ 70%**Note 3**)	4.0 V ≤ V_{DD} ≤ 5.5 V			-20.0	mA
			2.7 V ≤ V_{DD} < 4.0 V			-10.0	mA
			1.8 V ≤ V_{DD} < 2.7 V			-5.0	mA
			1.6 V ≤ V_{DD} < 1.8 V			-5.0	mA

Note 1. Device operation is guaranteed at the listed currents even if current is flowing from the EV_{DD0} or V_{DD} pin to an output pin.

Note 2. The combination of these and other pins must also not exceed the value for maximum total current.

Note 3. The listed currents apply when the duty cycle is no greater than 70%. Use the following formula to calculate the output current when the duty cycle is greater than 70%, where n is the duty cycle.

- Total output current from the listed pins = (I_{OH} × 0.7)/(n × 0.01)

  Example when n = 80% and I_{OH} = -10.0 mA

  Total output current from the listed pins = (-10.0 × 0.7)/(80 × 0.01) ≈ -8.7 mA

Note that the duty cycle has no effect on the current that is allowed to flow into a single pin. A current higher than the absolute maximum rating must not flow into a single pin.

Note 4. The maximum value is -30 mA with an ambient operating temperature range of +85°C to +105°C.

Note 5. The maximum value is -50 mA with an ambient operating temperature range of +85°C to +105°C.

Note 6. The maximum value is -60 mA with an ambient operating temperature range of +85°C to +105°C.

(Caution and Remark continue on the next page.)

Caution **The following pins are not capable of the output of high-level signals in the N-ch open-drain output mode. P00, P02 to P04, P10 to P15, P17, P31, P33, P50, P51, P55, P71, P72, and P74**

Remark The characteristics of functions multiplexed on a given pin are the same as those for the port pin unless otherwise specified.

(T_A = -40 to +105°C, 1.6 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (2/6)

Item	Symbol	Conditions		Min.	Typ.	Max.	Unit
Allowable low-level output current Note 1	I_{OL1}	Per pin for P00 to P06, P10 to P17, P30, P31, P33, P34, P40 to P43, P50 to P55, P70 to P77, P120, P130, P140, P141, P146, P147				20.0 Note 2	mA
		Per pin for P60, P61				15.0 Note 2	mA
		Total of P00 to P04, P40 to P43, P120, P130, P140, P141 (when duty ≤ 70% Note 3)	4.0 V ≤ EV_{DD0} ≤ 5.5 V			70.0 Note 4	mA
			2.7 V ≤ EV_{DD0} < 4.0 V			15.0	mA
			1.8 V ≤ EV_{DD0} < 2.7 V			9.0	mA
			1.6 V ≤ EV_{DD0} < 1.8 V			4.5	mA
		Total of P05, P06, P10 to P17, P30, P31, P33, P34, P50 to P55, P60, P61, P70 to P77, P146, P147 (when duty ≤ 70% Note 3)	4.0 V ≤ EV_{DD0} ≤ 5.5 V			80.0 Note 4	mA
			2.7 V ≤ EV_{DD0} < 4.0 V			35.0	mA
			1.8 V ≤ EV_{DD0} < 2.7 V			20.0	mA
			1.6 V ≤ EV_{DD0} < 1.8 V			10.0	mA
		Total of all pins (when duty ≤ 70% Note 3)				150.0 Note 5	mA
	I_{OL2}	Per pin for P20 to P27, P121 to P125	4.0 V ≤ V_{DD} ≤ 5.5 V			8.5 Note 2	mA
			2.7 V ≤ V_{DD} < 4.0 V			1.5 Note 2	mA
			1.8 V ≤ V_{DD} < 2.7 V			0.6 Note 2	mA
			1.6 V ≤ V_{DD} < 1.8 V			0.4 Note 2	mA
		Total of all pins (when duty ≤ 70% Note 3)	4.0 V ≤ V_{DD} ≤ 5.5 V			20	mA
			2.7 V ≤ V_{DD} < 4.0 V			20	mA
			1.8 V ≤ V_{DD} < 2.7 V			15	mA
			1.6 V ≤ V_{DD} < 1.8 V			10	mA

Note 1. Device operation is guaranteed at the listed currents even if current is flowing from an output pin to the EV_{SS0} or V_{SS} pin.

Note 2. The combination of these and other pins must also not exceed the value for maximum total current.

Note 3. The listed currents apply when the duty cycle is no greater than 70%. Use the following formula to calculate the output current when the duty cycle is greater than 70%, where n is the duty cycle.

- Total output current from the listed pins = (I_{OL} × 0.7)/(n × 0.01)

  Example when n = 80% and I_{OL} = 10.0 mA

  Total output current from the listed pins = (10.0 × 0.7)/(80 × 0.01) ≈ 8.7 mA

Note that the duty cycle has no effect on the current that is allowed to flow into a single pin. A current higher than the absolute maximum rating must not flow into a single pin.

Note 4. The maximum value is 40 mA with an ambient operating temperature range of +85°C to +105°C.

Note 5. The maximum value is 80 mA with an ambient operating temperature range of +85°C to +105°C.

Remark The characteristics of functions multiplexed on a given pin are the same as those for the port pin unless otherwise specified.

(T_A = -40 to +105°C, 1.6 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (3/6)

Item	Symbol	Conditions		Min.	Typ.	Max.	Unit
Input voltage, high	V_{IH1}	P00 to P06, P10 to P17, P30, P31, P33, P34, P40 to P43, P50 to P55, P70 to P77, P120, P140, P141, P146, P147	Normal input buffer	0.8 EV_{DD0}		EV_{DD0}	V
	V_{IH2}	P01, P03, P04, P10, P11, P13 to P17, P34, P50, P55, P71	TTL input buffer 4.0 V ≤ EV_{DD0} ≤ 5.5 V	2.2		EV_{DD0}	V
			TTL input buffer 3.3 V ≤ EV_{DD0} < 4.0 V	2.0		EV_{DD0}	V
			TTL input buffer 1.6 V ≤ EV_{DD0} < 3.3 V	1.5		EV_{DD0}	V
	V_{IH3}	P20 to P27		0.7 V_{DD}		V_{DD}	V
	V_{IH4}	P60, P61		0.7 EV_{DD0}		6.0	V
	V_{IH5}	P121 to P125, P137, EXCLK		0.8 V_{DD}		V_{DD}	V
Input voltage, low	V_{IL1}	P00 to P06, P10 to P17, P30, P31, P33, P34, P40 to P43, P50 to P55, P70 to P77, P120, P140, P141, P146, P147	Normal input buffer	0		0.2 EV_{DD0}	V
	V_{IL2}	P01, P03, P04, P10, P11, P13 to P17, P34, P50, P55, P71	TTL input buffer 4.0 V ≤ EV_{DD0} ≤ 5.5 V	0		0.8	V
			TTL input buffer 3.3 V ≤ EV_{DD0} < 4.0 V	0		0.5	V
			TTL input buffer 1.6 V ≤ EV_{DD0} < 3.3 V	0		0.32	V
	V_{IL3}	P20 to P27		0		0.3 V_{DD}	V
	V_{IL4}	P60, P61		0		0.3 EV_{DD0}	V
	V_{IL5}	P121 to P125, P137, EXCLK		0		0.2 V_{DD}	V

Caution The maximum value of V_{IH} of pins P00, P02 to P04, P10 to P15, P17, P31, P33, P50, P51, P55, P71, P72, and P74 is EV_{DD0}, even in the N-ch open-drain output mode.

Remark The characteristics of functions multiplexed on a given pin are the same as those for the port pin unless otherwise specified.

(T_A = -40 to +105°C, 1.6 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (4/6)

Item	Symbol	Conditions		Min.	Typ.	Max.	Unit
Output voltage, high	V_{OH1}	P00 to P06, P10 to P17, P30, P31, P33, P34, P40 to P43, P50 to P55, P70 to P77, P120, P130, P140, P141, P146, P147	4.0 V ≤ EV_{DD0} ≤ 5.5 V, I_{OH1} = -10.0 mA	EV_{DD0} - 1.5			V
			4.0 V ≤ EV_{DD0} ≤ 5.5 V, I_{OH1} = -3.0 mA	EV_{DD0} - 0.7			V
			2.7 V ≤ EV_{DD0} ≤ 5.5 V, I_{OH1} = -2.0 mA	EV_{DD0} - 0.6			V
			1.8 V ≤ EV_{DD0} ≤ 5.5 V, I_{OH1} = -1.5 mA	EV_{DD0} - 0.5			V
			1.6 V ≤ EV_{DD0} < 5.5 V, I_{OH1} = -1.0 mA	EV_{DD0} - 0.5			V
	V_{OH2}	P20 to P27, P121 to P125	4.0 V ≤ V_{DD} ≤ 5.5 V, I_{OH2} = -3.0 mA	V_{DD} - 0.7			V
			2.7 V ≤ V_{DD} < 4.0 V, I_{OH2} = -1.0 mA	V_{DD} - 0.5			V
			1.8 V ≤ V_{DD} < 2.7 V, I_{OH2} = -1.0 mA	V_{DD} - 0.5			V
			1.6 V ≤ V_{DD} < 1.8 V, I_{OH2} = -0.5 mA	V_{DD} - 0.5			V

Caution **The following pins are not capable of the output of high-level signals in the N-ch open-drain output mode. P00, P02 to P04, P10 to P15, P17, P31, P33, P50, P51, P55, P71, P72, and P74**

Remark The characteristics of functions multiplexed on a given pin are the same as those for the port pin unless otherwise specified.

(T_A = -40 to +105°C, 1.6 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (5/6)

Item	Symbol	Conditions			Min.	Typ.	Max.	Unit
Output voltage, low	V_{OL1}	P00 to P06, P10 to P17, P30, P31, P33, P34, P40 to P43, P50 to P55, P70 to P77, P120, P130, P140, P141, P146, P147	4.0 V ≤ EV_{DD0} ≤ 5.5 V	I_{OL1} = 20.0 mA			1.3	V
			4.0 V ≤ EV_{DD0} ≤ 5.5 V	I_{OL1} = 8.5 mA			0.7	V
			2.7 V ≤ EV_{DD0} ≤ 5.5 V	I_{OL1} = 3.0 mA			0.6	V
			2.7 V ≤ EV_{DD0} ≤ 5.5 V	I_{OL1} = 1.5 mA			0.4	V
			1.8 V ≤ EV_{DD0} ≤ 5.5 V	I_{OL1} = 0.6 mA			0.4	V
			1.6 V ≤ EV_{DD0} ≤ 5.5 V	I_{OL1} = 0.3 mA			0.4	V
	V_{OL2}	P20 to P27, P121 to P125	4.0 V ≤ V_{DD} ≤ 5.5 V, I_{OL2} = 8.5 mA				0.7	V
			2.7 V ≤ V_{DD} < 4.0 V, I_{OL2} = 1.5 mA				0.5	V
			1.8 V ≤ V_{DD} < 2.7 V, I_{OL2} = 0.6 mA				0.4	V
			1.6 V ≤ V_{DD} < 1.8 V, I_{OL2} = 0.4 mA				0.4	V
	V_{OL3}	P60, P61	4.0 V ≤ EV_{DD0} ≤ 5.5 V, I_{OL3} = 15.0 mA				2.0	V
			4.0 V ≤ EV_{DD0} ≤ 5.5 V, I_{OL3} = 5.0 mA				0.4	V
			2.7 V ≤ EV_{DD0} ≤ 5.5 V, I_{OL3} = 3.0 mA				0.4	V
			1.8 V ≤ EV_{DD0} ≤ 5.5 V, I_{OL3} = 2.0 mA				0.4	V
			1.6 V ≤ EV_{DD0} ≤ 5.5 V, I_{OL3} = 1.0 mA				0.4	V

Remark The characteristics of functions multiplexed on a given pin are the same as those for the port pin unless otherwise specified.

(T_A = -40 to +105°C, 1.6 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (6/6)

Item	Symbol	Conditions		Min.	Typ.	Max.	Unit
Input leakage current, high	I_{LIH1}	P00 to P06, P10 to P17, P30, P31, P33, P34, P40 to P43, P50 to P55, P60, P61, P70 to P77, P120, P140, P141, P146, P147	V_I = EV_{DD0}			0.5	μA
	I_{LIH2}	P20 to P27, P137	V_I = V_{DD}			0.5	μA
	I_{LIH3}	P121 to P125 (X1, X2, EXCLK)	V_I = V_{DD}			0.5	μA
Input leakage current, low	I_{LIL1}	P00 to P06, P10 to P17, P30, P31, P33, P34, P40 to P43, P50 to P55, P60, P61, P70 to P77, P120, P140, P141, P146, P147	V_I = EV_{SS0}			-0.5	μA
	I_{LIL2}	P20 to P27, P137	V_I = V_{SS}			-0.5	μA
	I_{LIL3}	P121 to P125 (X1, X2, EXCLK)	V_I = V_{SS}			-0.5	μA
On-chip pull-up resistance	R_U	P00 to P06, P10 to P17, P30, P31, P33, P34, P40 to P43, P50 to P55, P70 to P77, P120 to P125, P140, P141, P146, P147	V_I = 0 V, in input port	10	20	100	kΩ
<R>	R_{U2}	P137	V_I = 0 V	70	100	140	kΩ

Remark The characteristics of functions multiplexed on a given pin are the same as those for the port pin unless otherwise specified.

28.3.2 Characteristics of the supply current

(T_A = -40 to +105°C, 1.6 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (1/4)

Item	Symbol	Conditions					Min.	Typ.	Max.	Unit
<R> Supply current **Note 1**	I_{DD1}	Operating mode	HS (high-speed main) mode	f_{IH} = 32 MHz **Note 2**	Basic operation	V_{DD} = 5.0 V		1.1	—	mA
						V_{DD} = 1.8 V		1.1	—	
					Normal operation	V_{DD} = 5.0 V		2.6	4.5	mA
						V_{DD} = 1.8 V		2.6	4.5	
			LS (low-speed main) mode	f_{IH} = 24 MHz **Note 2**	Normal operation	V_{DD} = 5.0 V		1.9	3.4	mA
						V_{DD} = 1.8 V		1.9	3.4	
				f_{IH} = 16 MHz **Note 2**	Normal operation	V_{DD} = 5.0 V		1.4	2.4	mA
						V_{DD} = 1.8 V		1.4	2.4	
				f_{IH} = 4 MHz **Note 2**	Normal operation	V_{DD} = 5.0 V		0.6	1.0	mA
						V_{DD} = 1.6 V		0.6	0.9	
			LP (low-power main) mode	f_{IH} = 2 MHz **Note 2**	Normal operation	V_{DD} = 5.0 V		435	625	µA
						V_{DD} = 1.6 V		432	622	
				f_{IH} = 1 MHz **Note 2**	Normal operation	V_{DD} = 5.0 V		358	489	µA
						V_{DD} = 1.6 V		355	485	
			HS (high-speed main) mode	f_{MX} = 20 MHz **Note 3**, square wave input	Normal operation	V_{DD} = 5.0 V		1.7	2.9	mA
						V_{DD} = 1.8 V		1.6	2.8	
			LS (low-speed main) mode	f_{MX} = 20 MHz **Note 3**, square wave input	Normal operation	V_{DD} = 5.0 V		1.5	2.7	mA
						V_{DD} = 1.8 V		1.5	2.6	
				f_{MX} = 20 MHz **Note 3**, resonator connection	Normal operation	V_{DD} = 5.0 V		1.7	2.9	mA
						V_{DD} = 1.8 V		1.7	2.9	
				f_{MX} = 10 MHz **Note 3**, square wave input	Normal operation	V_{DD} = 5.0 V		0.8	1.4	mA
						V_{DD} = 1.8 V		0.8	1.4	
				f_{MX} = 10 MHz **Note 3**, resonator connection	Normal operation	V_{DD} = 5.0 V		0.9	1.5	mA
						V_{DD} = 1.8 V		0.9	1.5	
				f_{MX} = 8 MHz **Note 3**, square wave input	Normal operation	V_{DD} = 5.0 V		0.7	1.1	mA
						V_{DD} = 1.8 V		0.6	1.1	
				f_{MX} = 8 MHz **Note 3**, resonator connection	Normal operation	V_{DD} = 5.0 V		0.8	1.3	mA
						V_{DD} = 1.8 V		0.7	1.2	

Note 1. The listed currents are the total currents flowing into V_{DD} and EV_{DD0}, including the input leakage currents flowing when the level of the input pin is fixed to V_{DD}, EV_{DD0} or V_{SS}, EV_{SS0}. The following points apply in the HS (high-speed main), LS (low-speed main), and LP (low-power main) modes.

- The currents in the “Typ.” column do not include the operating currents of the peripheral modules.
- The currents in the “Max.” column include the operating currents of the TAU, SAU, and IICA modules.

Note 2. The listed currents apply when the high-speed system clock and low-speed on-chip oscillator are stopped.

Note 3. The listed currents apply when the high-speed on-chip oscillator and low-speed on-chip oscillator are stopped.

Remark 1. f_{IH}: High-speed on-chip oscillator clock frequency

Remark 2. f_{MX}: High-speed system clock frequency (X1 clock oscillation frequency or external main system clock frequency)

Remark 3. The typical value for the ambient operating temperature (T_A) is +25°C unless otherwise specified.

(T_A = -40 to +105°C, 1.6 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (2/4)

Item	Symbol	Conditions					Min.	Typ.	Max.	Unit
<R> Supply current **Note 1**	I_{DD1}	Operating mode	Subsystem clock operation mode	f_{SUB} = 32.768 kHz**Note 2**, low-speed on-chip oscillator operation	Normal operation	T_A = -40°C		2.8	5.4	µA
						T_A = +25°C		3.1	5.7	
						T_A = +50°C		3.3	8.8	
						T_A = +70°C		3.6	14.0	
						T_A = +85°C		4.1	23.3	
						T_A = +105°C		5.4	48.3	

Note 1. The listed currents are the total currents flowing into V_{DD} and EV_{DD0}, including the input leakage currents flowing when the level of the input pin is fixed to V_{DD}, EV_{DD0} or V_{SS}, EV_{SS0}. In the subsystem clock operation mode, the currents in both the "Typ." and "Max." columns do not include the operating currents of the peripheral modules.

Note 2. The listed currents apply when the high-speed on-chip oscillator and high-speed system clock are stopped.

Remark 1. f_{IL}: Low-speed on-chip oscillator clock frequency

Remark 2. f_{SUB}: Subsystem clock frequency

(T_A = -40 to +105°C, 1.6 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (3/4)

Item	Symbol	Conditions				Min.	Typ.	Max.	Unit
<R> Supply current**Note 1**	I_{DD2} **Note 2**	HALT mode	HS (high-speed main) mode	f_{IH} = 32 MHz**Note 3**	V_{DD} = 5.0 V		0.46	1.85	mA
					V_{DD} = 1.8 V		0.45	1.84	
			LS (low-speed main) mode	f_{IH} = 24 MHz**Note 3**	V_{DD} = 5.0 V		0.39	1.44	mA
					V_{DD} = 1.8 V		0.38	1.43	
				f_{IH} = 16 MHz**Note 3**	V_{DD} = 5.0 V		0.39	1.13	mA
					V_{DD} = 1.8 V		0.38	1.11	
				f_{IH} = 4 MHz**Note 3**	V_{DD} = 5.0 V		0.34	0.58	mA
					V_{DD} = 1.6 V		0.33	0.57	
			LP (low-power main) mode	f_{IH} = 2 MHz**Note 3**	V_{DD} = 5.0 V		289	442	μA
					V_{DD} = 1.6 V		287	440	
				f_{IH} = 1 MHz**Note 3**	V_{DD} = 5.0 V		285	398	μA
					V_{DD} = 1.6 V		282	394	
			HS (high-speed main) mode	f_{MX} = 20 MHz**Note 4**, square wave input	V_{DD} = 5.0 V		0.19	1.03	mA
					V_{DD} = 1.8 V		0.16	0.99	
			LS (low-speed main) mode	f_{MX} = 20 MHz**Note 4**, square wave input	V_{DD} = 5.0 V		0.19	1.03	mA
					V_{DD} = 1.8 V		0.16	0.99	
				f_{MX} = 20 MHz**Note 4**, resonator connection	V_{DD} = 5.0 V		0.39	1.28	mA
					V_{DD} = 1.8 V		0.37	1.27	
				f_{MX} = 10 MHz**Note 4**, square wave input	V_{DD} = 5.0 V		0.12	0.54	mA
					V_{DD} = 1.8 V		0.10	0.52	
				f_{MX} = 10 MHz**Note 4**, resonator connection	V_{DD} = 5.0 V		0.23	0.68	mA
					V_{DD} = 1.8 V		0.22	0.67	
				f_{MX} = 8 MHz**Note 4**, square wave input	V_{DD} = 5.0 V		0.10	0.45	mA
					V_{DD} = 1.8 V		0.09	0.43	
				f_{MX} = 8 MHz**Note 4**, resonator connection	V_{DD} = 5.0 V		0.20	0.57	mA
					V_{DD} = 1.8 V		0.19	0.56	

Note 1. The listed currents are the total currents flowing into V_{DD} and EV_{DD0}, including the input leakage currents flowing when the level of the input pin is fixed to V_{DD}, EV_{DD0} or V_{SS}, EV_{SS0}. The following points apply in the HS (high-speed main), LS (low-speed main), and LP (low-power main) modes.

- The currents in the “Typ.” column do not include the operating currents of the peripheral modules.
- The currents in the “Max.” column include the operating currents of the TAU, SAU, and IICA modules.

Note 2. The listed currents apply when the HALT instruction has been fetched from the flash memory for execution.

Note 3. The listed currents apply when the high-speed system clock and low-speed on-chip oscillator are stopped.

Note 4. The listed currents apply when the high-speed on-chip oscillator and low-speed on-chip oscillator are stopped.

Remark 1. f_{IH}: High-speed on-chip oscillator clock frequency

Remark 2. f_{MX}: High-speed system clock frequency (X1 clock oscillation frequency or external main system clock frequency)

Remark 3. The typical value for the ambient operating temperature (T_A) is +25°C unless otherwise specified.

(TA = -40 to +105°C, 1.6 V ≤ EVDD0 ≤ VDD ≤ 5.5 V, VSS = EVSS0 = 0 V) (4/4)

Item	Symbol	Conditions				Min.	Typ.	Max.	Unit
<R> Supply current **Note 1**	IDD2 **Note 2**	HALT mode	Subsystem clock operation mode	fSUB = 32.768 kHz**Note 3**, low-speed on-chip oscillator operation	TA = -40°C		0.45	2.70	μA
					TA = +25°C		0.55	2.78	
					TA = +50°C		0.65	5.80	
					TA = +70°C		0.86	10.86	
					TA = +85°C		1.95	21.52	
					TA = +105°C		3.25	46.67	
	IDD3	STOP mode**Note 4**			TA = -40°C		0.16	2	μA
					TA = +25°C		0.23	2	
					TA = +50°C		0.50	5	
					TA = +70°C		0.85	10	
					TA = +85°C		1.50	19	
					TA = +105°C		2.90	44	

Note 1. The listed currents are the total currents flowing into VDD and EVDD0, including the input leakage currents flowing when the level of the input pin is fixed to VDD, EVDD0 or VSS, EVSS0. In the subsystem clock operation mode or the STOP mode, the currents in both the “Typ.” and “Max.” columns do not include the operating currents of the peripheral modules.

Note 2. The listed currents apply when the HALT instruction has been fetched from the flash memory for execution.

Note 3. The listed currents apply when the high-speed on-chip oscillator and high-speed system clock are stopped.

<R> **Note 4.** The listed currents apply when the low-speed on-chip oscillator is stopped. For the currents when operating the low-speed on-chip oscillator in the STOP mode, refer to the currents when operating the subsystem clock in the HALT mode.

Remark 1. fIL: Low-speed on-chip oscillator clock frequency

Remark 2. fSUB: Subsystem clock frequency

Peripheral Functions (Common to all products)

(T_A = -40 to +105°C, 1.6 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V)

Item	Symbol	Conditions			Min.	Typ.	Max.	Unit
High-speed on-chip oscillator operating current	I_{FIH} **Note 1**					320		μA
Low-speed on-chip oscillator operating current	I_{FIL} **Note 1**					0.2		μA
32-bit interval timer operating current	I_{IT} **Notes 1, 2, 3**					0.04		μA
Watchdog timer operating current	I_{WDT} **Notes 1, 2, 4**	f_{IL} = 32.768 kHz (typ.)				0.32		μA
A/D converter operating current	I_{ADC} **Notes 1, 5**	When conversion at maximum speed		Normal mode, AV_{REFP} = V_{DD} = 5.0 V		1.3	1.7	mA
				Low voltage mode, AV_{REFP} = V_{DD} = 3.0 V		0.5	0.7	mA
A/D converter internal reference voltage current	I_{ADREF} **Note 1**					100		μA
Temperature sensor operating current	I_{TMPS} **Note 1**					114		μA
LVD operating current	I_{LVD0} **Notes 1, 6**					0.02		μA
	I_{LVD1} **Notes 1, 6**					0.02		μA
Self-programming operating current	I_{FSP} **Notes 1, 7**					2.5	12.2	mA
Data flash memory rewrite operating current	I_{BGO} **Notes 1, 8**					2.5	12.2	mA
<R> SNOOZE operating current	I_{SNOZ} **Note 1**	f_{IH} = 32 MHz	ADC to be in use	The ADC is shifting from the STOP mode to the SNOOZE mode. **Note 9**		0.50	0.70	mA
				The ADC is operating in the low-voltage mode. AV_{REFP} = V_{DD} = 3.0 V		0.90	1.40	
			Simplified SPI (CSI)/UART to be in use			0.70	0.92	mA

Note 1. This current flows into V_{DD}.

Note 2. The listed currents apply when the high-speed on-chip oscillator and high-speed system clock are stopped.

Note 3. This current only flows to the 32-bit interval timer. It does not include the operating current of the low-speed on-chip oscillator. The supply current of this MCU is the sum of either I_{DD1} or I_{DD2}, and I_{IT}, when the 32-bit interval timer is operating or in the HALT mode. When the low-speed on-chip oscillator is selected, I_{FIL} should be included in the supply current.

Note 4. This current only flows to the watchdog timer. It includes the operating current of the low-speed on-chip oscillator. The supply current of this MCU is the sum of I_{DD1}, I_{DD2} or I_{DD3} and I_{WDT} when the watchdog timer is operating.

Note 5. This current only flows to the A/D converter. The supply current of this MCU is the sum of I_{DD1} or I_{DD2} and I_{ADC} when the A/D converter is operating or in the HALT mode.

Note 6. This current only flows to the LVD circuit. The supply current of this MCU is the sum of I_{DD1}, I_{DD2} or I_{DD3} and I_{LVD} when the LVD circuit is in operation.

Note 7. This current only flows during self programming.

Note 8. This current only flows while the data flash memory is being rewritten.

Note 9. For shift time to the SNOOZE mode, see **16.3.3 SNOOZE mode**.

(Remarks are listed on the next page.)

Remark 1. f_{IL}: Low-speed on-chip oscillator clock frequency

Remark 2. f_{CLK}: CPU/peripheral hardware clock frequency

Remark 3. The typical value for the ambient operating temperature (T_A) is +25°C unless otherwise specified.

28.4 AC Characteristics

(T_A = -40 to +105°C, 1.6 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V)

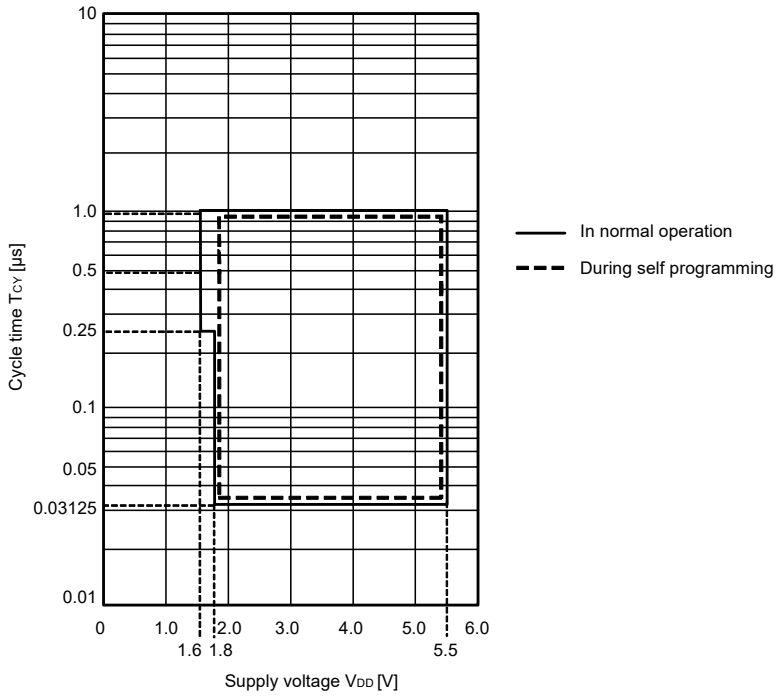
Item	Symbol	Conditions			Min.	Typ.	Max.	Unit
Instruction cycle (minimum instruction execution time)	T_{CY}	Main system clock (f_{MAIN}) operation	HS (high-speed main) mode	1.8 V ≤ V_{DD} ≤ 5.5 V	0.03125		1	μs
				1.6 V ≤ V_{DD} ≤ 1.8 V	0.25		1	μs
			LS (low-speed main) mode	1.8 V ≤ V_{DD} ≤ 5.5 V	0.04167		1	μs
				1.6 V ≤ V_{DD} ≤ 1.8 V	0.25		1	μs
			LP (low-power main) mode	1.6 V ≤ V_{DD} ≤ 5.5 V	0.5		1	μs
		Subsystem clock (f_{SUB}) operation		1.6 V ≤ V_{DD} ≤ 5.5 V		30.5		μs
		In the self programming mode	HS (high-speed main) mode	1.8 V ≤ V_{DD} ≤ 5.5 V	0.03125		1	μs
			LS (low-speed main) mode	1.8 V ≤ V_{DD} ≤ 5.5 V	0.04167		1	μs
External system clock frequency	f_{EX}	1.8 V ≤ V_{DD} ≤ 5.5 V			1.0		20.0	MHz
		1.6 V ≤ V_{DD} < 1.8 V			1.0		4.0	MHz
External system clock input high-level width, low-level width	t_{EXH}, t_{EXL}	1.8 V ≤ V_{DD} ≤ 5.5 V			24			ns
		1.6 V ≤ V_{DD} < 1.8 V			120			ns
TI00 to TI07 input high-level width, low-level width	t_{TIH}, t_{TIL}				1/f_{MCK} + 10			ns[Note]
TO00 to TO07 output frequency	f_{TO}	HS (high-speed main) mode LS (low-speed main) mode		4.0 V ≤ EV_{DD0} ≤ 5.5 V			16	MHz
				2.7 V ≤ EV_{DD0} < 4.0 V			8	MHz
				1.8 V ≤ EV_{DD0} < 2.7 V			4	MHz
				1.6 V ≤ EV_{DD0} < 1.8 V			2	MHz
		LP (low-power main) mode		1.6 V ≤ EV_{DD0} ≤ 5.5 V			2	MHz
Interrupt input high-level width, low-level width	t_{INTH}, t_{INTL}	INTP0 and INTPx multiplexed with P121, P122, or P125		1.6 V ≤ V_{DD} ≤ 5.5 V	1			μs
		INTP1 to INTP11 (however INTPx multiplexed with P121, P122, or P125 is not applicable)		1.6 V ≤ EV_{DD0} ≤ 5.5 V	1			μs
Key interrupt input high-level width, low-level width	t_{KRH}, t_{KRL}	KR0 to KR7		1.8 V ≤ EV_{DD0} ≤ 5.5 V	250			ns
				1.6 V ≤ EV_{DD0} < 1.8 V	1			μs
$\overline{\text{RESET}}$ low-level width	t_{RSL}				10			μs

Note The following conditions are required for low voltage interface when EV_{DD0} < V_{DD}.
1.8 V ≤ EV_{DD0} < 2.7 V: 125 ns min.
1.6 V ≤ EV_{DD0} < 1.8 V: 250 ns min.

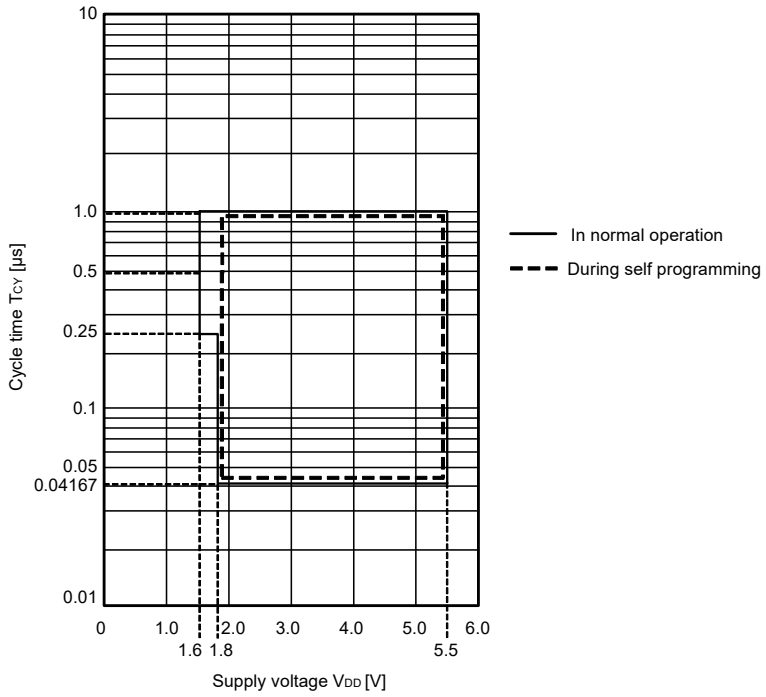
Remark f_{MCK}: Timer array unit operating clock frequency
(To set this operating clock, use the CKSmn0 and CKSmn1 bits of the timer mode register mn (TMRmn) (m: Unit number (m = 0), n: Channel number (n = 0 to 7).)

Minimum Instruction Execution Time during Main System Clock Operation

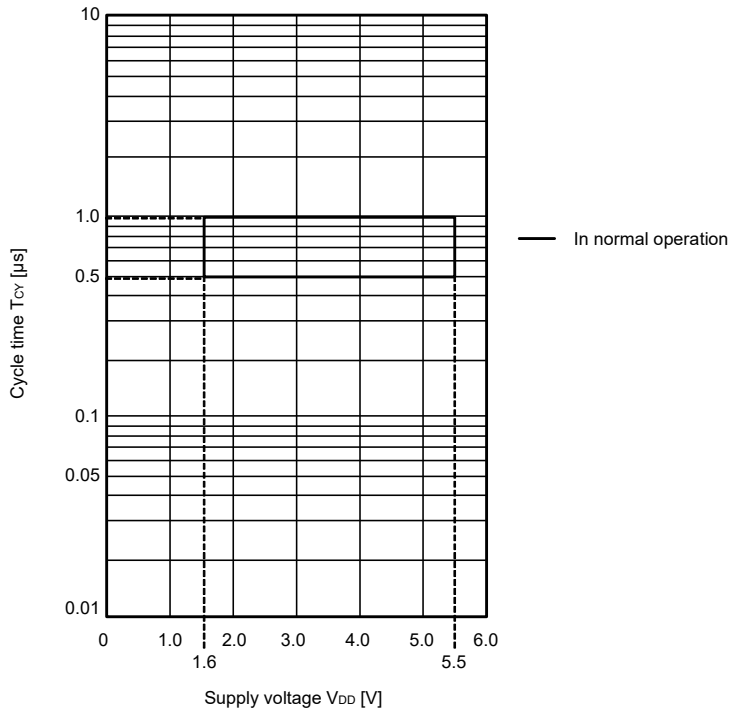
TCY vs VDD (HS (high-speed main) mode)



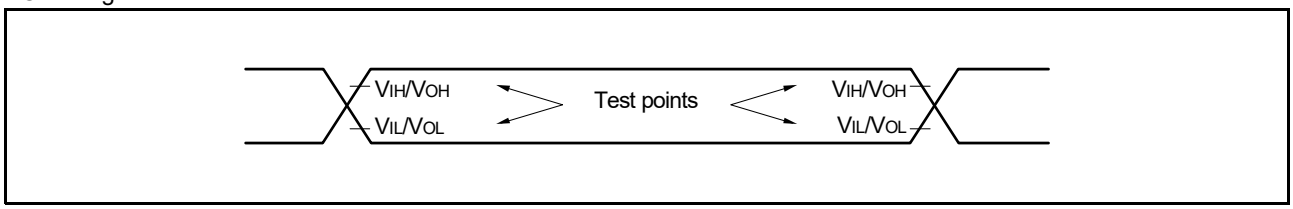
TCY vs VDD (LS (low-speed main) mode)



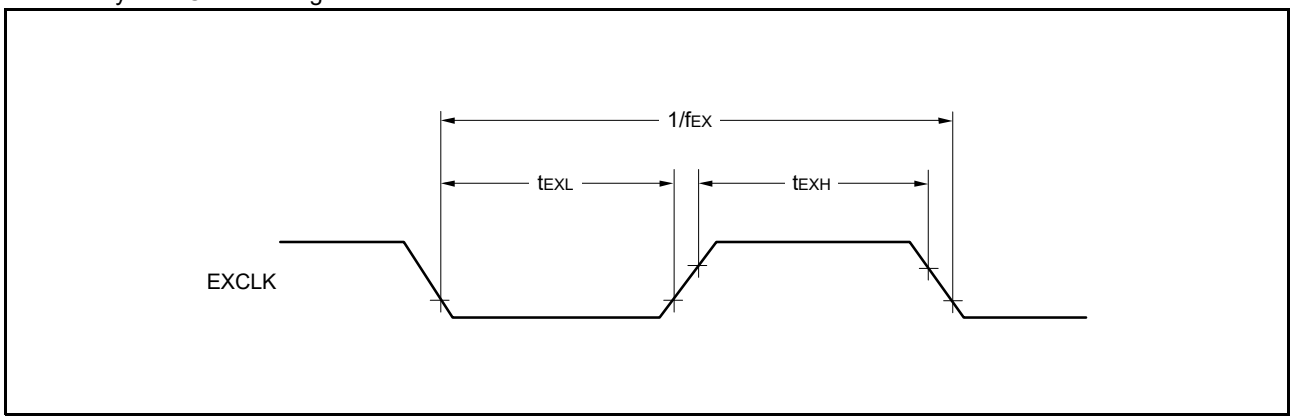
TCY vs VDD (LP (low-power main) mode)



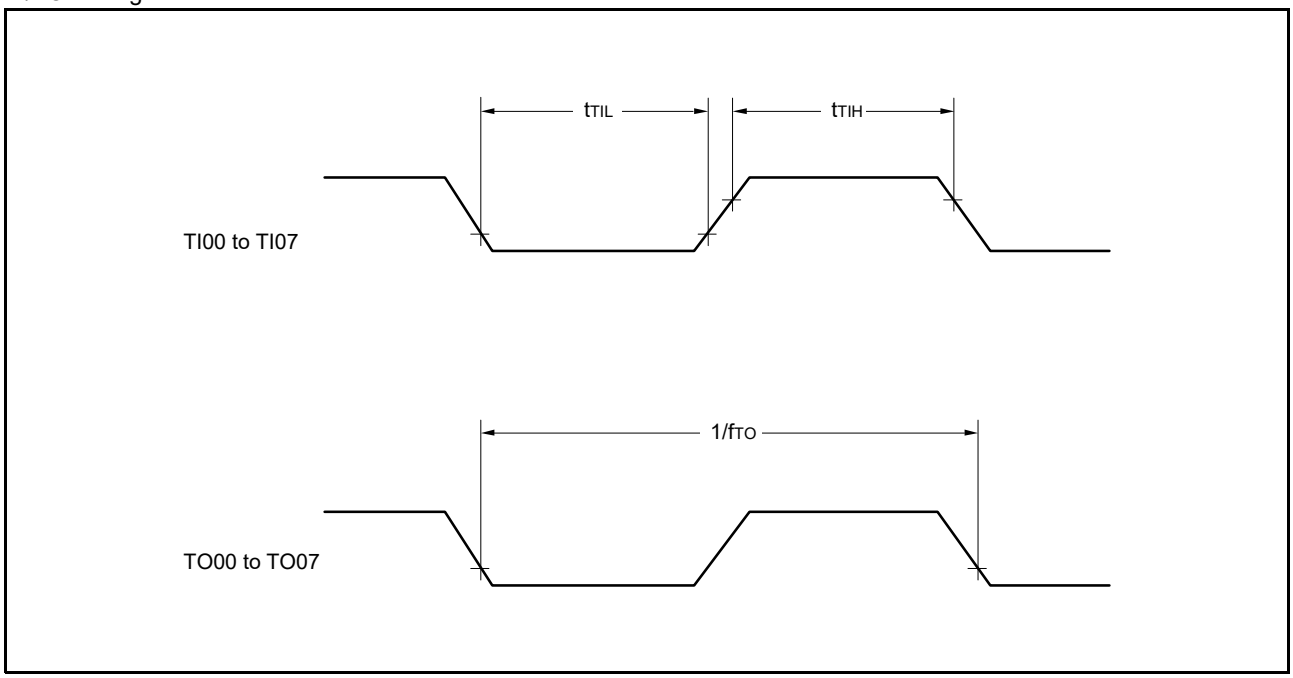
AC Timing Test Points



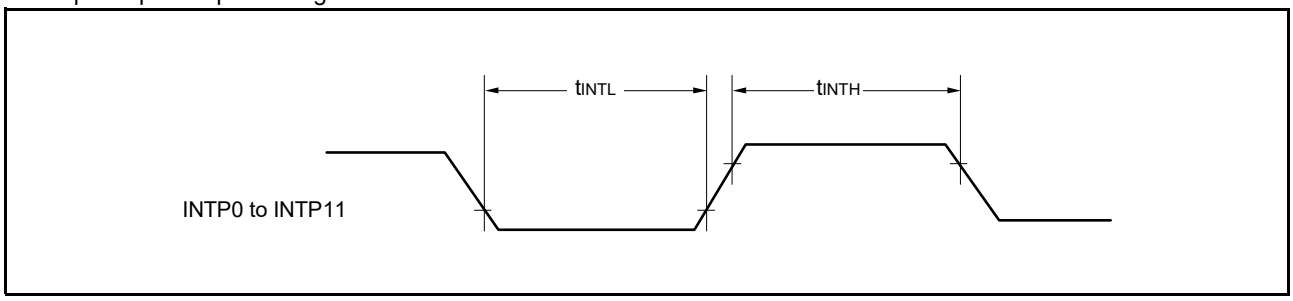
External System Clock Timing



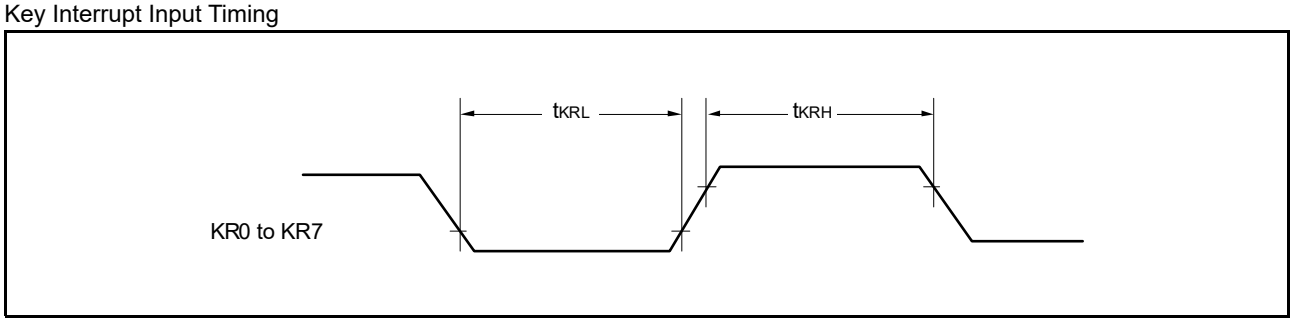
TI/TO Timing



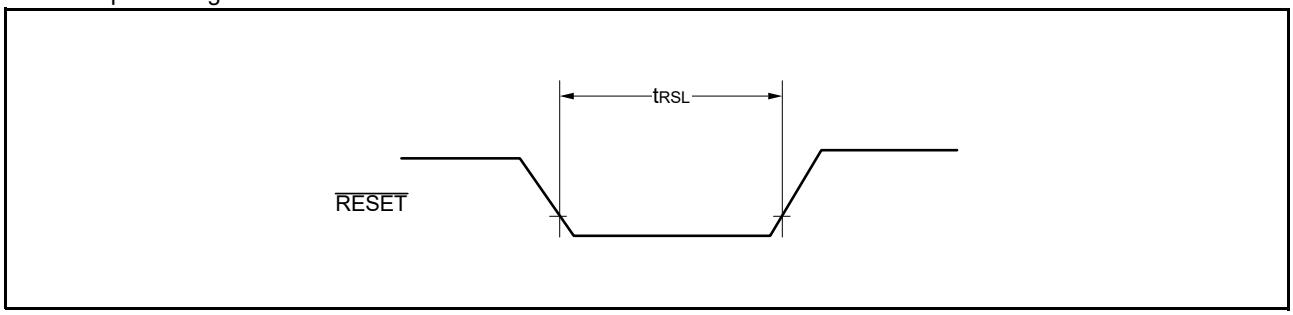
Interrupt Request Input Timing



Key Interrupt Input Timing

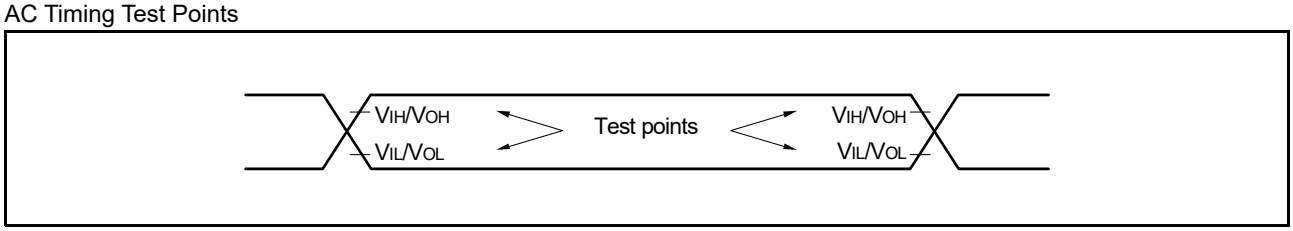


$\overline{\text{RESET}}$ Input Timing



28.5 Characteristics of the Peripheral Functions

AC Timing Test Points



28.5.1 Serial array unit

1. In UART communications with devices operating at same voltage levels

(T_A = -40 to +105°C, 1.6 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V)

Item	Symbol	Conditions	HS (High-speed Main) Mode		LS (Low-speed Main) Mode		LP (Low-power Main) Mode		Unit
			Min.	Max.	Min.	Max.	Min.	Max.	
Transfer rate **Note 1**		1.6 V ≤ EV_{DD0} ≤ 5.5 V		f_{MCK}/6 **Note 2**		f_{MCK}/6 **Note 2**		f_{MCK}/6	bps
		Theoretical value of the maximum transfer rate f_{MCK} = f_{CLK} **Note 3**		5.3		4		0.33	Mbps

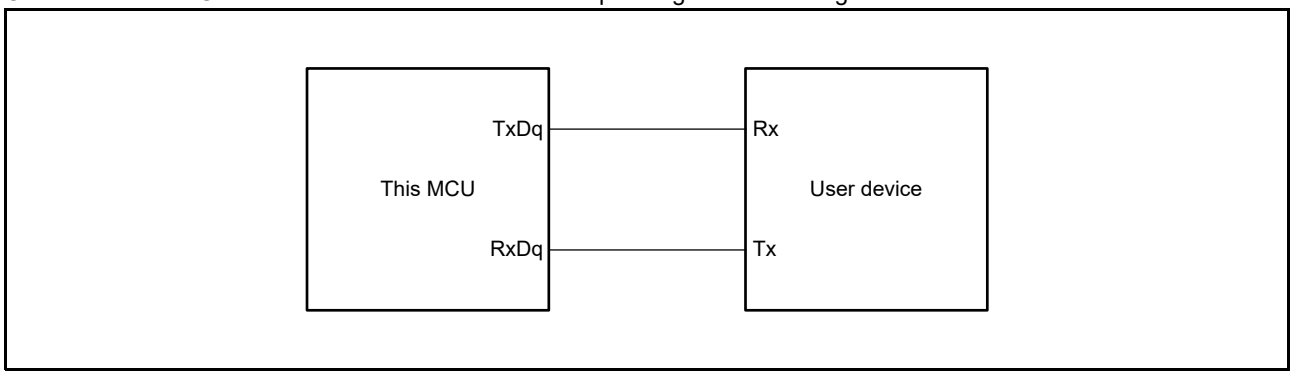
Note 1. The transfer rate in the SNOOZE mode is within the range from 4800 to 9600 bps.

Note 2. The following conditions are required for low voltage interface when EV_{DD0} < V_{DD}.
2.4 V ≤ EV_{DD0} < 2.7 V: 2.6 Mbps max.
1.8 V ≤ EV_{DD0} < 2.4 V: 1.3 Mbps max.
1.6 V ≤ EV_{DD0} < 1.8 V: 0.6 Mbps max.

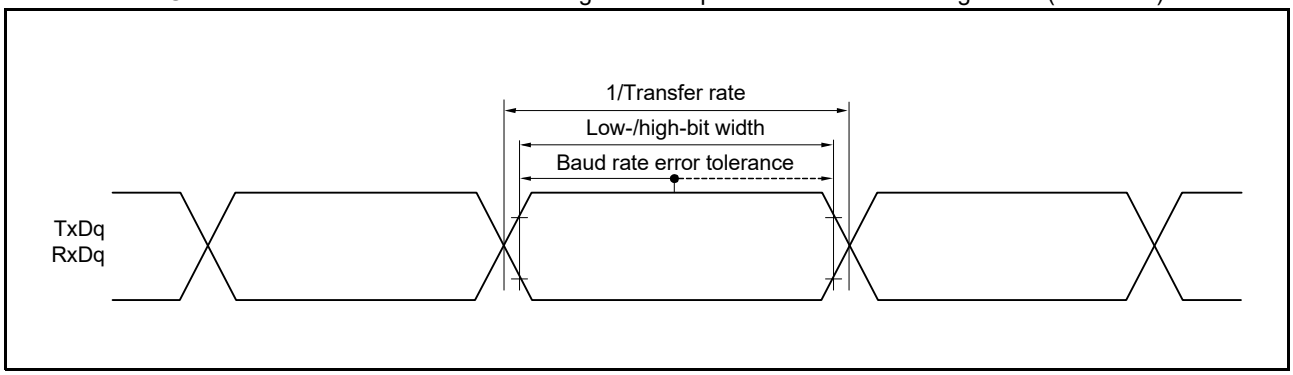
Note 3. The maximum operating frequencies of the CPU/peripheral hardware clock (f_{CLK}) are as follows.
HS (high-speed main) mode: 32 MHz (1.8 V ≤ V_{DD} ≤ 5.5 V)
4 MHz (1.6 V ≤ V_{DD} ≤ 5.5 V)
LS (low-speed main) mode: 24 MHz (1.8 V ≤ V_{DD} ≤ 5.5 V)
4 MHz (1.6 V ≤ V_{DD} ≤ 5.5 V)
LP (low-power main) mode: 2 MHz (1.6 V ≤ V_{DD} ≤ 5.5 V)

Caution Select the normal input buffer for the RxDq pin and the normal output mode for the TxDq pin by using a port input mode register (PIMg) and port output mode register (POMg).

Connection in the UART communications with devices operating at same voltage levels



Bit width in the UART communications when interfacing devices operate at the same voltage level (reference)



Remark 1. q: UART number (q = 0 to 2), g: PIM and POM number (g = 0, 1, 5)

Remark 2. f_{MCK}: Serial array unit operation clock frequency
(To set this operating clock, set the CKSmn bit in the serial mode register mn (SMRmn) (m: Unit number, n: Channel number (mn = 00 to 03, 10, 11).)

2. In simplified SPI (CSI) communications in the master mode with devices operating at same voltage levels with the internal SCKp clock (the ratings below are only applicable to CSI00)

(T_A = -40 to +85°C, 2.7 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V)

Item	Symbol	Conditions		HS (High-speed Main) Mode		LS (Low-speed Main) Mode		LP (Low-power Main) Mode		Unit
				Min.	Max.	Min.	Max.	Min.	Max.	
SCKp cycle time	t_{KCY1}	t_{KCY1} ≥ 2/f_{CLK}	4.0 V ≤ EV_{DD0} ≤ 5.5 V	62.5		83.3		1000		ns
			2.7 V ≤ EV_{DD0} ≤ 5.5 V	83.3		125		1000		ns
SCKp high-level width, low-level width	t_{KH1}, t_{KL1}	4.0 V ≤ EV_{DD0} ≤ 5.5 V		t_{KCY1}/2 - 7		t_{KCY1}/2 - 10		t_{KCY1}/2 - 50		ns
		2.7 V ≤ EV_{DD0} ≤ 5.5 V		t_{KCY1}/2 - 10		t_{KCY1}/2 - 15		t_{KCY1}/2 - 50		ns
SIp setup time (to SCKp↑)**Note 1**	t_{SIK1}	4.0 V ≤ EV_{DD0} ≤ 5.5 V		23		33		110		ns
		2.7 V ≤ EV_{DD0} ≤ 5.5 V		33		50		110		ns
SIp hold time (from SCKp↑) **Note 1**	t_{KSI1}	2.7 V ≤ EV_{DD0} ≤ 5.5 V		10		10		10		ns
Delay time from SCKp↓ to SOp output**Note 2**	t_{KSO1}	C = 20 pF**Note 3**			10		10		10	ns

Note 1. The setting applies when DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The setting for the SIp setup time becomes “to SCKp↓” and that for the SIp hold time becomes “from SCKp↓” when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

Note 2. This setting applies when DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The setting for the delay time to SOp output becomes “from SCKp↑” when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

Note 3. C is the load capacitance of the SCKp and SOp output lines.

Caution Select the normal input buffer for the SIp pin and the normal output mode for the SOp pin and SCKp pin by using a port input mode register (PIMg) and port output mode register (POMg).

Remark 1. The listed times are only valid when the peripheral I/O redirect function of CSI00 is not in use.

Remark 2. p: CSI number (p = 00), m: Unit number (m = 0), n: Channel number (n = 0), g: PIM and POM numbers (g = 1)

Remark 3. f_{MCK}: Serial array unit operation clock frequency
(To set this operating clock, use the CKSmn bit in the serial mode register mn (SMRmn) (m: Unit number, n: Channel number (mn = 00).)

3. In simplified SPI (CSI) communications in the master mode with devices operating at same voltage levels with the internal SCKp clock

(T_A = -40 to +105°C, 1.6 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V)

Item	Symbol	Conditions		HS (High-speed Main) Mode		LS (Low-speed Main) Mode		LP (Low-power Main) Mode		Unit
				Min.	Max.	Min.	Max.	Min.	Max.	
SCKp cycle time	t_{KCY1}	t_{KCY1} ≥ 4/f_{CLK}	2.7 V ≤ EV_{DD0} ≤ 5.5 V	125		166		2000		ns
			2.4 V ≤ EV_{DD0} ≤ 5.5 V	250		250		2000		ns
			1.8 V ≤ EV_{DD0} ≤ 5.5 V	500		500		2000		ns
			1.6 V ≤ EV_{DD0} ≤ 5.5 V	1000		1000		2000		ns
SCKp high-level width, low-level width	t_{KH1}, t_{KL1}	4.0 V ≤ EV_{DD0} ≤ 5.5 V		t_{KCY1}/2 - 12		t_{KCY1}/2 - 21		t_{KCY1}/2 - 50		ns
		2.7 V ≤ EV_{DD0} ≤ 5.5 V		t_{KCY1}/2 - 18		t_{KCY1}/2 - 25		t_{KCY1}/2 - 50		ns
		2.4 V ≤ EV_{DD0} ≤ 5.5 V		t_{KCY1}/2 - 38		t_{KCY1}/2 - 38		t_{KCY1}/2 - 50		ns
		1.8 V ≤ EV_{DD0} ≤ 5.5 V		t_{KCY1}/2 - 50		t_{KCY1}/2 - 50		t_{KCY1}/2 - 50		ns
		1.6 V ≤ EV_{DD0} ≤ 5.5 V		t_{KCY1}/2 - 100		t_{KCY1}/2 - 100		t_{KCY1}/2 - 100		ns
SIp setup time (to SCKp↑)**Note 1**	t_{SIK1}	4.0 V ≤ EV_{DD0} ≤ 5.5 V		44		54		110		ns
		2.7 V ≤ EV_{DD0} ≤ 5.5 V		44		54		110		ns
		2.4 V ≤ EV_{DD0} ≤ 5.5 V		75		75		110		ns
		1.8 V ≤ EV_{DD0} ≤ 5.5 V		110		110		110		ns
		1.6 V ≤ EV_{DD0} ≤ 5.5 V		220		220		220		ns
SIp hold time (from SCKp↑) **Note 1**	t_{KSI1}	1.6 V ≤ EV_{DD0} ≤ 5.5 V		19		19		19		ns
Delay time from SCKp↓ to SOp output**Note 2**	t_{KSO1}	1.6 V ≤ EV_{DD0} ≤ 5.5 V C = 30 pF**Note 3**			25		25		25	ns

Note 1. This setting applies when DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The setting for the SIp setup time becomes “to SCKp↓” and that for the SIp hold time becomes “from SCKp↓” when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

Note 2. This setting applies when DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The setting for the delay time to SOp output becomes “from SCKp↑” when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

Note 3. C is the load capacitance of the SCKp and SOp output lines.

Caution Select the normal input buffer for the SIp pin and the normal output mode for the SOp pin and SCKp pin by using a port input mode register (PIMg) and port output mode register (POMg).

Remark 1. p: CSI number (p = 00, 01, 10, 11, 20, 21), m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), g: PIM and POM numbers (g = 0, 1, 5, 7)

Remark 2. f_{MCK}: Serial array unit operation clock frequency
(To set this operating clock, use the CKSmn bit in the serial mode register mn (SMRmn) (m: Unit number, n: Channel number = 00 to 03, 10, 11).)

4. In simplified SPI (CSI) communications in the slave mode with devices operating at same voltage levels with the SCKp external clock

(T_A = -40 to +105°C, 1.6 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (1/2)

Item	Symbol	Conditions		HS (High-speed Main) Mode		LS (Low-speed Main) Mode		LP (Low-power Main) Mode		Unit
				Min.	Max.	Min.	Max.	Min.	Max.	
SCKp cycle time **Note 1**	t_{KCY2}	4.0 V ≤ EV_{DD0} ≤ 5.5 V	20 MHz < f_{MCK}	8/f_{MCK}		8/f_{MCK}		—		ns
			f_{MCK} ≤ 20 MHz	6/f_{MCK}		6/f_{MCK}		6/f_{MCK}		ns
		2.7 V ≤ EV_{DD0} ≤ 5.5 V	16 MHz < f_{MCK}	8/f_{MCK}		8/f_{MCK}		—		ns
			f_{MCK} ≤ 16 MHz	6/f_{MCK}		6/f_{MCK}		6/f_{MCK}		ns
		2.4 V ≤ EV_{DD0} ≤ 5.5 V		6/f_{MCK} and 500		6/f_{MCK} and 500		6/f_{MCK} and 500		ns
		1.8 V ≤ EV_{DD0} ≤ 5.5 V		6/f_{MCK} and 750		6/f_{MCK} and 750		6/f_{MCK} and 750		ns
		1.6 V ≤ EV_{DD0} ≤ 5.5 V		6/f_{MCK} and 1500		6/f_{MCK} and 1500		6/f_{MCK} and 1500		ns
SCKp high-level width, low-level width	t_{KH2}, t_{KL2}	4.0 V ≤ EV_{DD0} ≤ 5.5 V		t_{KCY2}/2 - 7		t_{KCY2}/2 - 7		t_{KCY2}/2 - 7		ns
		2.7 V ≤ EV_{DD0} ≤ 5.5 V		t_{KCY2}/2 - 8		t_{KCY2}/2 - 8		t_{KCY2}/2 - 8		ns
		1.8 V ≤ EV_{DD0} ≤ 5.5 V		t_{KCY2}/2 - 18		t_{KCY2}/2 - 18		t_{KCY2}/2 - 18		ns
		1.6 V ≤ EV_{DD0} ≤ 5.5 V		t_{KCY2}/2 - 66		t_{KCY2}/2 - 66		t_{KCY2}/2 - 66		ns

(Notes, Caution, and Remarks are listed on the next page.)

4. In simplified SPI (CSI) communications in the slave mode with devices operating at same voltage levels with the SCKp external clock

(T_A = -40 to +105°C, 1.6 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (2/2)

Item	Symbol	Conditions		HS (High-speed Main) Mode		LS (Low-speed Main) Mode		LP (Low-power Main) Mode		Unit
				Min.	Max.	Min.	Max.	Min.	Max.	
SIp setup time (to SCKp↑)**Note 2**	tSIK2	2.7 V ≤ EV_{DD0} ≤ 5.5 V		1/fMCK + 20		1/fMCK + 30		1/fMCK + 30		ns
		1.8 V ≤ EV_{DD0} ≤ 5.5 V		1/fMCK + 30		1/fMCK + 30		1/fMCK + 30		ns
		1.6 V ≤ EV_{DD0} ≤ 5.5 V		1/fMCK + 40		1/fMCK + 40		1/fMCK + 40		ns
SIp hold time (from SCKp↑)**Note 2**	tKSI2	1.8 V ≤ EV_{DD0} ≤ 5.5 V		1/fMCK + 31		1/fMCK + 31		1/fMCK + 31		ns
		1.6 V ≤ EV_{DD0} ≤ 5.5 V		1/fMCK + 250		1/fMCK + 250		1/fMCK + 250		ns
Delay time from SCKp↓ to SOp output **Note 3**	tKSO2	C = 30 pF **Note 4**	2.7 V ≤ EV_{DD0} ≤ 5.5 V		2/fMCK + 44		2/fMCK + 110		2/fMCK + 110	ns
			2.4 V ≤ EV_{DD0} ≤ 5.5 V		2/fMCK + 75		2/fMCK + 110		2/fMCK + 110	ns
			1.8 V ≤ EV_{DD0} ≤ 5.5 V		2/fMCK + 110		2/fMCK + 110		2/fMCK + 110	ns
			1.6 V ≤ EV_{DD0} ≤ 5.5 V		2/fMCK + 220		2/fMCK + 220		2/fMCK + 220	ns

Note 1. Transfer rate in the SNOOZE mode is 1 Mbps at the maximum.

Note 2. This setting applies when DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The setting for the SIp setup time becomes “to SCKp↓” and that for the SIp hold time becomes “from SCKp↓” when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

Note 3. This setting applies when DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The setting for the delay time to SOp output becomes “from SCKp↑” when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

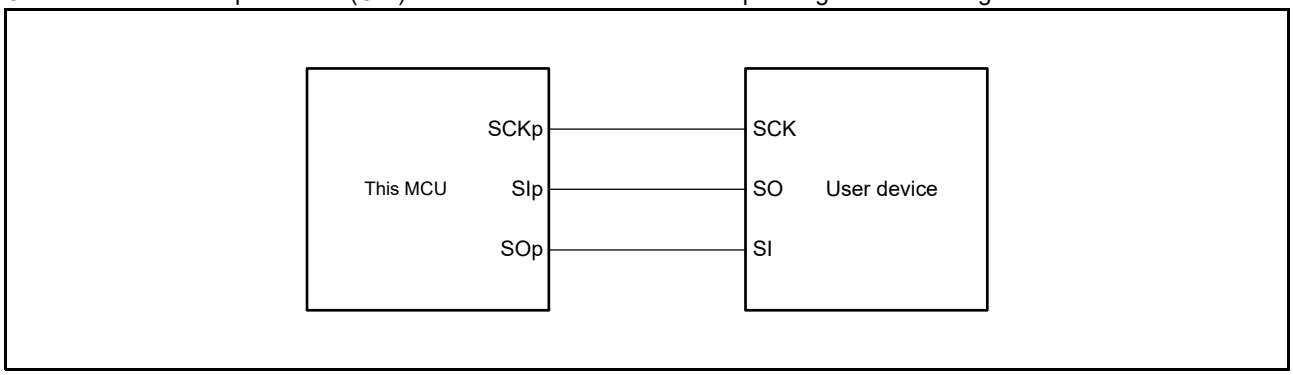
Note 4. C is the load capacitance of the SOp output line.

Caution Select the normal input buffer for the SIp pin and SCKp pin and the normal output mode for the SOp pin by using a port input mode register (PIMg) and port output mode register (POMg).

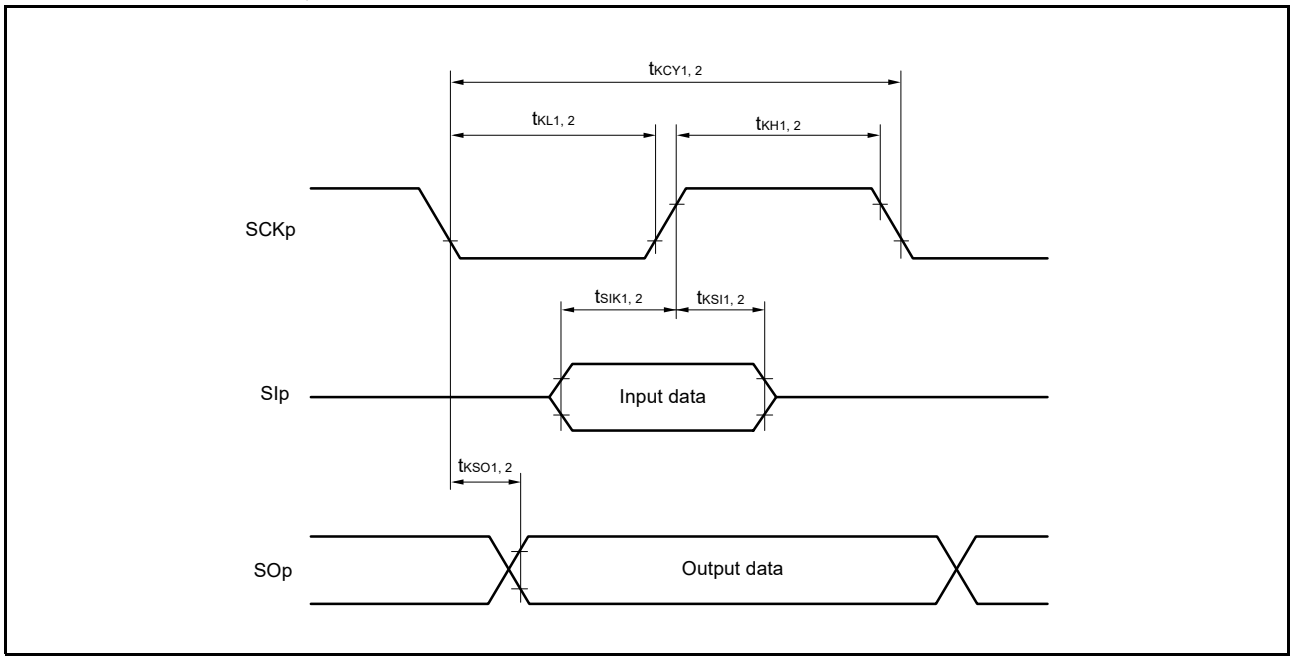
Remark 1. p: CSI number (p = 00, 01, 10, 11, 20, 21), m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), g: PIM and POM numbers (g = 0, 1, 5, 7)

Remark 2. fMCK: Serial array unit operation clock frequency
(To set this operating clock, use the CKSmn bit in the serial mode register mn (SMRmn) (m: Unit number, n: Channel number = 00 to 03, 10, 11).)

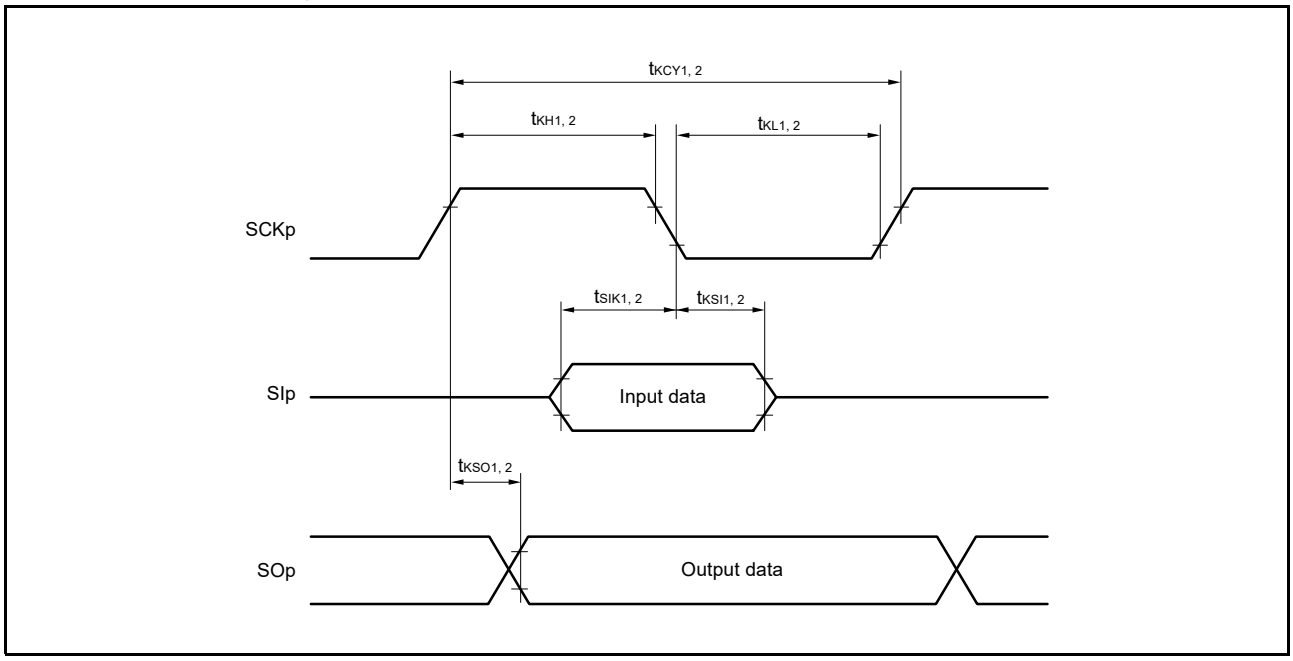
Connection in the simplified SPI (CSI) communications with devices operating at same voltage levels



Timing of serial transfer in the simplified SPI (CSI) communications with devices operating at same voltage levels when DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1



Timing of serial transfer in the simplified SPI (CSI) communications with devices operating at same voltage levels when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0



Remark 1. p: CSI number (p = 00, 01, 10, 11, 20, 21)

Remark 2. m: Unit number, n: Channel number (mn = 00 to 03, 10, 11)

5. In simplified I^2C communications with devices operating at same voltage levels

(T_A = -40 to +105°C, 1.6 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (1/2)

Item	Symbol	Conditions	HS (High-speed Main) Mode		LS (Low-speed Main) Mode		LP (Low-power Main) Mode		Unit
			Min.	Max.	Min.	Max.	Min.	Max.	
SCLr clock frequency	f_{SCL}	2.7 V ≤ EV_{DD0} ≤ 5.5 V, C_b = 50 pF, R_b = 2.7 kΩ		1000 Note 1		1000 Note 1		400 Note 1	kHz
		1.8 V ≤ EV_{DD0} ≤ 5.5 V, C_b = 100 pF, R_b = 3 kΩ		400 Note 1		400 Note 1		400 Note 1	kHz
		1.8 V ≤ EV_{DD0} < 2.7 V, C_b = 100 pF, R_b = 5 kΩ		300 Note 1		300 Note 1		300 Note 1	kHz
		1.6 V ≤ EV_{DD0} < 1.8 V, C_b = 100 pF, R_b = 5 kΩ		250 Note 1		250 Note 1		250 Note 1	kHz
Hold time when SCLr is low	t_{LOW}	2.7 V ≤ EV_{DD0} ≤ 5.5 V, C_b = 50 pF, R_b = 2.7 kΩ	475		475		1150		ns
		1.8 V ≤ EV_{DD0} ≤ 5.5 V, C_b = 100 pF, R_b = 3 kΩ	1150		1150		1150		ns
		1.8 V ≤ EV_{DD0} < 2.7 V, C_b = 100 pF, R_b = 5 kΩ	1550		1550		1550		ns
		1.6 V ≤ EV_{DD0} < 1.8 V, C_b = 100 pF, R_b = 5 kΩ	1850		1850		1850		ns
Hold time when SCLr is high	t_{HIGH}	2.7 V ≤ EV_{DD0} ≤ 5.5 V, C_b = 50 pF, R_b = 2.7 kΩ	475		475		1150		ns
		1.8 V ≤ EV_{DD0} ≤ 5.5 V, C_b = 100 pF, R_b = 3 kΩ	1150		1150		1150		ns
		1.8 V ≤ EV_{DD0} < 2.7 V, C_b = 100 pF, R_b = 5 kΩ	1550		1550		1550		ns
		1.6 V ≤ EV_{DD0} < 1.8 V, C_b = 100 pF, R_b = 5 kΩ	1850		1850		1850		ns

(Notes and Caution are listed on the next page, and Remarks are listed on page 944.)

5. In simplified I²C communications with devices operating at same voltage levels

(T_A = -40 to +105°C, 1.6 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (2/2)

Item	Symbol	Conditions	HS (High-speed Main) Mode		LS (Low-speed Main) Mode		LP (Low-power Main) Mode		Unit
			Min.	Max.	Min.	Max.	Min.	Max.	
Data setup time (reception)	$t_{SU:DAT}$	2.7 V ≤ EV_{DD0} ≤ 5.5 V, C_b = 50 pF, R_b = 2.7 kΩ	1/f_{MCK} + 85 **Note 2**		1/f_{MCK} + 85 **Note 2**		1/f_{MCK} + 145 **Note 2**		ns
		1.8 V ≤ EV_{DD0} ≤ 5.5 V, C_b = 100 pF, R_b = 3 kΩ	1/f_{MCK} + 145 **Note 2**		1/f_{MCK} + 145 **Note 2**		1/f_{MCK} + 145 **Note 2**		ns
		1.8 V ≤ EV_{DD0} < 2.7 V, C_b = 100 pF, R_b = 5 kΩ	1/f_{MCK} + 230 **Note 2**		1/f_{MCK} + 230 **Note 2**		1/f_{MCK} + 230 **Note 2**		ns
		1.6 V ≤ EV_{DD0} < 1.8 V, C_b = 100 pF, R_b = 5 kΩ	1/f_{MCK} + 290 **Note 2**		1/f_{MCK} + 290 **Note 2**		1/f_{MCK} + 290 **Note 2**		ns
Data hold time (transmission)	$t_{HD:DAT}$	2.7 V ≤ EV_{DD0} ≤ 5.5 V, C_b = 50 pF, R_b = 2.7 kΩ	0	305	0	305	0	305	ns
		1.8 V ≤ EV_{DD0} ≤ 5.5 V, C_b = 100 pF, R_b = 3 kΩ	0	355	0	355	0	355	ns
		1.8 V ≤ EV_{DD0} < 2.7 V, C_b = 100 pF, R_b = 5 kΩ	0	405	0	405	0	405	ns
		1.6 V ≤ EV_{DD0} < 1.8 V, C_b = 100 pF, R_b = 5 kΩ	0	405	0	405	0	405	ns

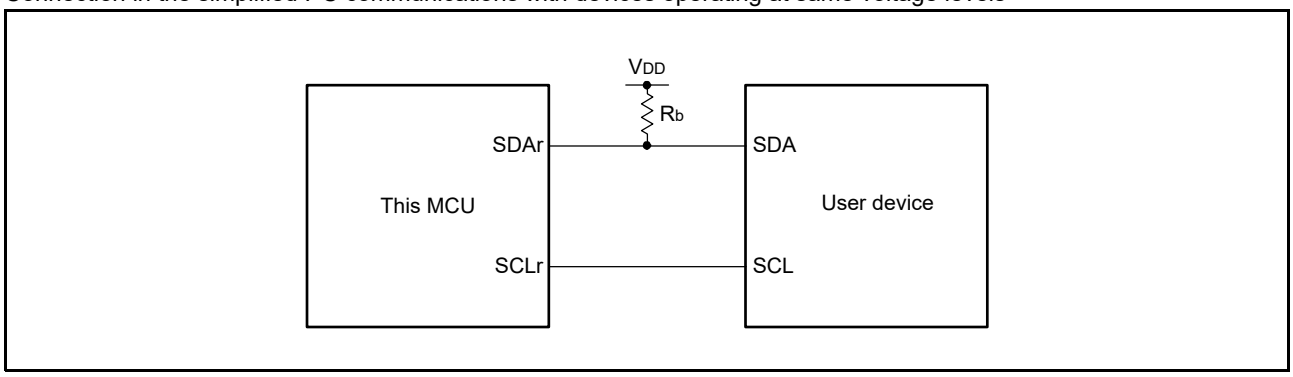
Note 1. The listed times must be no greater than f_{MCK}/4.

Note 2. Set f_{MCK} so that it will not exceed the hold time when SCLr is low or high.

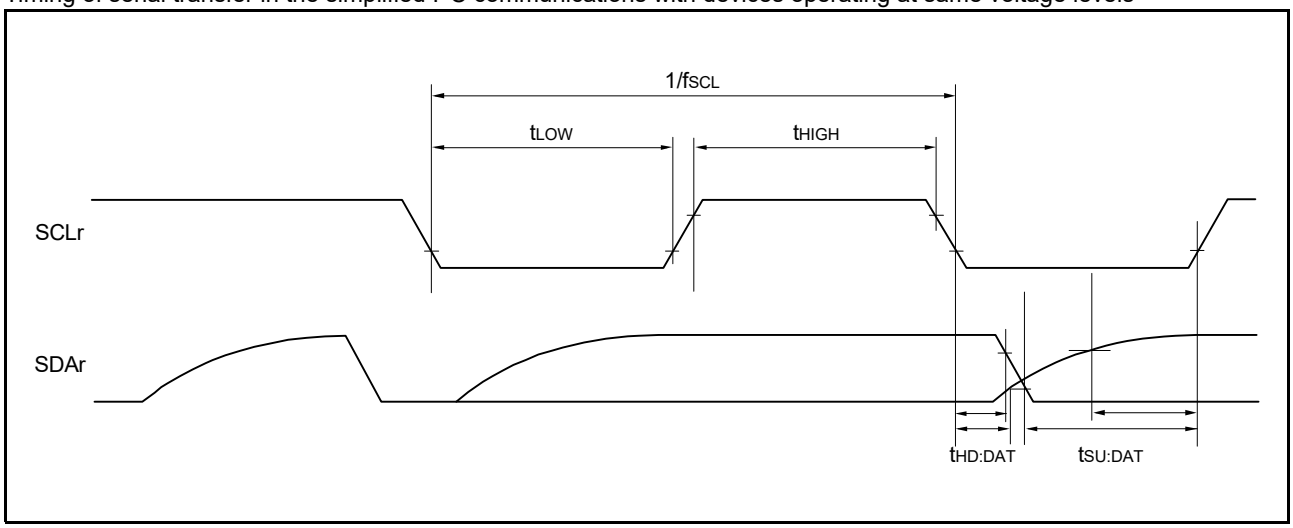
Caution Select the normal input buffer and the N-ch open drain output (withstand voltage of V_{DD} (for the 20- to 48-pin products)/withstand voltage of EV_{DD} (for the 64-pin products)) mode for the SDAr pin and the normal output mode for the SCLr pin by using a port input mode register (PIMg) and port output mode register (POMh).

(Remarks are listed on the next page.)

Connection in the simplified I²C communications with devices operating at same voltage levels



Timing of serial transfer in the simplified I²C communications with devices operating at same voltage levels



Remark 1. R_b[Ω]: Communication line (SDAr) pull-up resistance, C_b[F]: Communication line (SDAr, SCLr) load capacitance

Remark 2. r: IIC number (r = 00, 01, 10, 11, 20, 21), g: PIM number (g = 0, 1, 5, 7), h: POM number (g = 0, 1, 5, 7)

Remark 3. f_{MCK}: Serial array unit operation clock frequency
(To set this operating clock, use the CKSmn bit in the serial mode register mn (SMRmn) (m: Unit number, n: Channel number = 00 to 03, 10, 11).)

6. In UART communications with devices operating at different voltage levels (1.8 V, 2.5 V, 3 V)

(T_A = -40 to +105°C, 1.8 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (1/2)

Item	Symbol	Conditions		HS (High-speed Main) Mode		LS (Low-speed Main) Mode		LP (Low-power Main) Mode		Unit
				Min.	Max.	Min.	Max.	Min.	Max.	
Transfer rate		Reception	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V		f_{MCK}/6 **Note 1**		f_{MCK}/6 **Note 1**		f_{MCK}/6 **Note 1**	bps
			Theoretical value of the maximum transfer rate f_{MCK} = f_{CLK} **Note 2**		5.3		4		0.33	Mbps
			2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V		f_{MCK}/6 **Note 1**		f_{MCK}/6 **Note 1**		f_{MCK}/6 **Note 1**	bps
			Theoretical value of the maximum transfer rate f_{MCK} = f_{CLK} **Note 2**		5.3		4		0.33	Mbps
			1.8 V ≤ EV_{DD0} < 3.3 V, 1.6 V ≤ V_b ≤ 2.0 V		f_{MCK}/6 **Notes 1, 3, 4**		f_{MCK}/6 **Notes 1, 3**		f_{MCK}/6 **Notes 1, 3**	bps
			Theoretical value of the maximum transfer rate f_{MCK} = f_{CLK} **Note 2**		5.3		4		0.33	Mbps

Note 1. Transfer rate in the SNOOZE mode is within the range from 4800 to 9600 bps.

Note 2. The maximum operating frequencies of the CPU/peripheral hardware clock (f_{CLK}) are:
HS (high-speed main) mode: 32 MHz (1.8 V ≤ V_{DD} ≤ 5.5 V)
4 MHz (1.6 V ≤ V_{DD} ≤ 5.5 V)
LS (low-speed main) mode: 24 MHz (1.8 V ≤ V_{DD} ≤ 5.5 V)
4 MHz (1.6 V ≤ V_{DD} ≤ 5.5 V)
LP (low-power main) mode: 2 MHz (1.6 V ≤ V_{DD} ≤ 5.5 V)

Note 3. Use this rate with EV_{DD0} ≥ V_b.

Note 4. The following conditions are required for low voltage interface when EV_{DD0} < V_{DD}.
2.4 V ≤ EV_{DD0} < 2.7 V: 2.6 Mbps (max.)
1.8 V ≤ EV_{DD0} < 2.4 V: 1.3 Mbps (max.)

Caution Select the TTL input buffer for the RxDq pin and the N-ch open drain output (withstand voltage of V_{DD} (for the 20- to 48-pin products)/withstand voltage of EV_{DD} (for the 64-pin products)) mode for the TxDq pin by using a port input mode register (PIMg) and port output mode register (POMg). For V_{IH} and V_{IL}, see the DC characteristics with TTL input buffer selected.

Remark 1. V_b[V]: Communication line voltage

Remark 2. q: UART number (q = 0 to 2), g: PIM and POM number (g = 0, 1, 5)

Remark 3. f_{MCK}: Serial array unit operation clock frequency
(To set this operating clock, use the CKSmn bit in the serial mode register mn (SMRmn) (m: Unit number, n: Channel number = 00 to 03, 10, 11).)

Remark 4. Communications by using UART2 with devices operating at different voltage levels are not possible when the setting of the PIOR15 and PIOR14 bits of the peripheral I/O redirection register (PIOR1) is 01B or 11B.

6. In UART communications with devices operating at different voltage levels (1.8 V, 2.5 V, 3 V)

(T_A = -40 to +105°C, 1.8 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (2/2)

Item	Symbol	Conditions		HS (High-speed Main) Mode		LS (Low-speed Main) Mode		LP (Low-power Main) Mode		Unit
				Min.	Max.	Min.	Max.	Min.	Max.	
Transfer rate		Transmission	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V		**Note 1**		**Note 1**		**Note 1**	bps
			Theoretical value of the maximum transfer rate C_b = 50 pF, R_b = 1.4 kΩ, V_b = 2.7 V		2.8**Note 2**		2.8**Note 2**		2.8**Note 2**	Mbps
			2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V		**Note 3**		**Note 3**		**Note 3**	bps
			Theoretical value of the maximum transfer rate C_b = 50 pF, R_b = 2.7 kΩ, V_b = 2.3 V		1.2**Note 4**		1.2**Note 4**		1.2**Note 4**	Mbps
			1.8 V ≤ EV_{DD0} < 3.3 V, 1.6 V ≤ V_b ≤ 2.0 V		**Notes 5, 6**		**Notes 5, 6**		**Notes 5, 6**	bps
			Theoretical value of the maximum transfer rate C_b = 50 pF, R_b = 5.5 kΩ, V_b = 1.6 V		0.43 **Note 7**		0.43 **Note 7**		0.43 **Note 7**	Mbps

Note 1. The smaller maximum transfer rate derived by using f_{MCK}/6 or the following expression is the valid maximum transfer rate.
Expression for calculating the transfer rate when 4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V

$$\text{Maximum transfer rate} = \frac{1}{\{-C_b \times R_b \times \ln(1 - \frac{2.2}{V_b})\} \times 3} \text{ [bps]}$$

$$\text{Baud rate error (theoretical value)} = \frac{\frac{1}{\text{Transfer rate} \times 2} - \{-C_b \times R_b \times \ln(1 - \frac{2.2}{V_b})\}}{(\frac{1}{\text{Transfer rate}}) \times \text{Number of transferred bits}} \times 100 \text{ [\%]}$$

* This value is the theoretical value of the relative difference between the transmission and reception sides.

Note 2. This rate is calculated as an example when the conditions described in the “Conditions” column are met. See **Note 1** above to calculate the maximum transfer rate under conditions of the customer.

(Notes and Caution continue in the next page.)

Note 3. The smaller maximum transfer rate derived by using $f_{MCK}/6$ or the following expression is the valid maximum transfer rate.
Expression for calculating the transfer rate when 2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V

$$\text{Maximum transfer rate} = \frac{1}{\{-C_b \times R_b \times \ln(1 - \frac{2.0}{V_b})\} \times 3} \text{ [bps]}$$

$$\text{Baud rate error (theoretical value)} = \frac{\frac{1}{\text{Transfer rate} \times 2} - \{-C_b \times R_b \times \ln(1 - \frac{2.0}{V_b})\}}{(\frac{1}{\text{Transfer rate}}) \times \text{Number of transferred bits}} \times 100 \text{ [\%]}$$

* This value is the theoretical value of the relative difference between the transmission and reception sides.

Note 4. This rate is calculated as an example when the conditions described in the “Conditions” column are met. See **Note 3** above to calculate the maximum transfer rate under conditions of the customer.

Note 5. Use this rate with $EV_{DD0} \geq V_b$.

Note 6. The smaller maximum transfer rate derived by using $f_{MCK}/6$ or the following expression is the valid maximum transfer rate.
Expression for calculating the transfer rate when 1.8 V ≤ EV_{DD0} < 3.3 V, 1.6 V ≤ V_b ≤ 2.0 V

$$\text{Maximum transfer rate} = \frac{1}{\{-C_b \times R_b \times \ln(1 - \frac{1.5}{V_b})\} \times 3} \text{ [bps]}$$

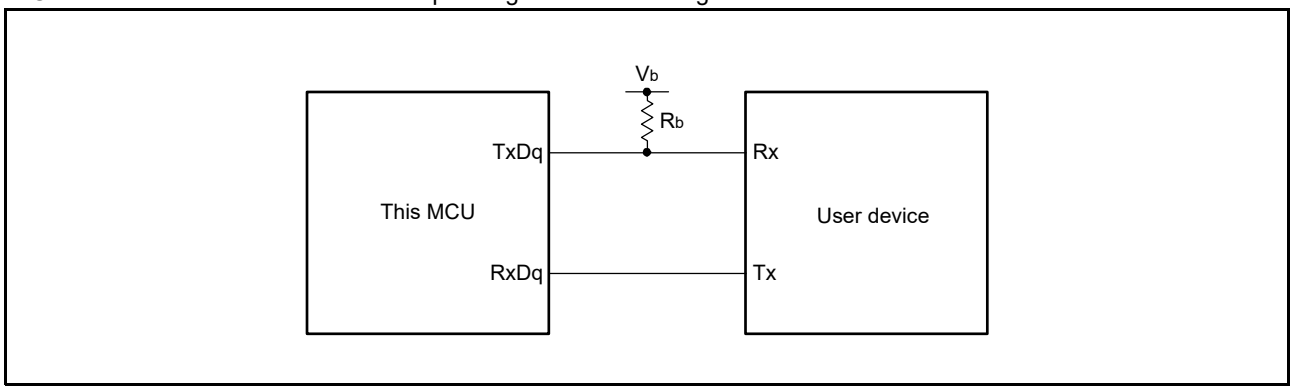
$$\text{Baud rate error (theoretical value)} = \frac{\frac{1}{\text{Transfer rate} \times 2} - \{-C_b \times R_b \times \ln(1 - \frac{1.5}{V_b})\}}{(\frac{1}{\text{Transfer rate}}) \times \text{Number of transferred bits}} \times 100 \text{ [\%]}$$

* This value is the theoretical value of the relative difference between the transmission and reception sides.

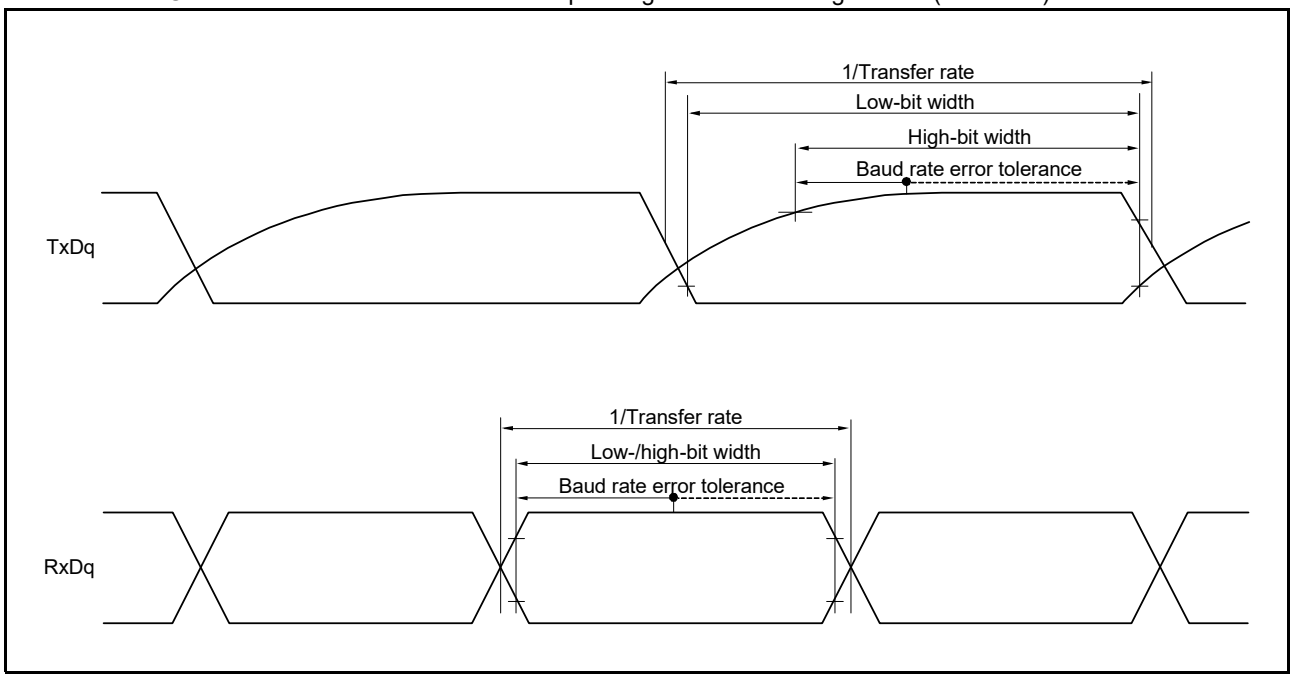
Note 7. This rate is calculated as an example when the conditions described in the “Conditions” column are met. Refer to **Note 6** above to calculate the maximum transfer rate under conditions of the customer.

Caution Select the TTL input buffer for the RxDq pin and the N-ch open drain output (withstand voltage of V_{DD} (for the 20- to 48-pin products)/withstand voltage of EV_{DD} (for the 64-pin products)) mode for the TxDq pin by using a port input mode register (PIMg) and port output mode register (POMg). For V_{IH} and V_{IL}, see the DC characteristics with TTL input buffer selected.

In UART communications with devices operating at different voltage levels



Bit width in the UART communications with devices operating at different voltage levels (reference)



Remark 1. R_b[Ω]: Communication line (TxDq) pull-up resistance, C_b[F]: Communication line (TxDq) load capacitance, V_b[V]: Communication line voltage

Remark 2. q: UART number (q = 0 to 2), g: PIM and POM number (g = 0, 1, 5)

Remark 3. f_{MCK}: Serial array unit operation clock frequency
(To set this operating clock, use the CKSmn bit in the serial mode register mn (SMRmn) (m: Unit number, n: Channel number = 00 to 03, 10, 11).)

Remark 4. Communications by using UART2 with devices operating at different voltage levels are not possible when the setting of the PIOR15 and PIOR14 bits of the peripheral I/O redirection register (PIOR1) is 01B or 11B.

7. In simplified SPI (CSI) communications in the master mode with devices operating at different voltage levels (2.5 V or 3 V) with the internal SCKp clock (the ratings below are only applicable to CSI00)

(T_A = -40 to +105°C, 2.7 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (1/2)

Item	Symbol	Conditions		HS (High-speed Main) Mode		LS (Low-speed Main) Mode		LP (Low-power Main) Mode		Unit
				Min.	Max.	Min.	Max.	Min.	Max.	
SCKp cycle time	t_{KCY1}	t_{KCY1} ≥ 2/f_{CLK}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 20 pF, R_b = 1.4 kΩ	200		200		2300		ns
			2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 20 pF, R_b = 2.7 kΩ	300		300		2300		ns
SCKp high-level width	t_{KH1}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 20 pF, R_b = 1.4 kΩ		t_{KCY1}/2 - 50		t_{KCY1}/2 - 50		t_{KCY1}/2 - 50		ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 20 pF, R_b = 2.7 kΩ		t_{KCY1}/2 - 120		t_{KCY1}/2 - 120		t_{KCY1}/2 - 120		ns
SCKp low-level width	t_{KL1}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 20 pF, R_b = 1.4 kΩ		t_{KCY1}/2 - 7		t_{KCY1}/2 - 7		t_{KCY1}/2 - 50		ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 20 pF, R_b = 2.7 kΩ		t_{KCY1}/2 - 10		t_{KCY1}/2 - 10		t_{KCY1}/2 - 50		ns
SIp setup time (to SCKp↑)**Note 1**	t_{SIK1}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 20 pF, R_b = 1.4 kΩ		58		58		479		ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 20 pF, R_b = 2.7 kΩ		121		121		479		ns
SIp hold time (from SCKp↑)**Note 1**	t_{KSI1}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 20 pF, R_b = 1.4 kΩ		10		10		10		ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 20 pF, R_b = 2.7 kΩ		10		10		10		ns
Delay time from SCKp↓ to SOp output**Note 1**	t_{KSO1}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 20 pF, R_b = 1.4 kΩ			60		60		60	ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 20 pF, R_b = 2.7 kΩ			130		130		130	ns

(Notes, Caution, and Remarks are listed on the next page.)

7. In simplified SPI (CSI) communications in the master mode with devices operating at different voltage levels (2.5 V or 3 V) with the internal SCKp clock (the ratings below are only applicable to CSI00)

(T_A = -40 to +105°C, 2.7 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (2/2)

Item	Symbol	Conditions	HS (High-speed Main) Mode		LS (Low-speed Main) Mode		LP (Low-power Main) Mode		Unit
			Min.	Max.	Min.	Max.	Min.	Max.	
SIp setup time (to SCKp↓)**Note 2**	t_{SIK1}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 20 pF, R_b = 1.4 kΩ	23		23		110		ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 20 pF, R_b = 2.7 kΩ	33		33		110		ns
SIp hold time (from SCKp↓)**Note 2**	t_{KSI1}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 20 pF, R_b = 1.4 kΩ	10		10		10		ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 20 pF, R_b = 2.7 kΩ	10		10		10		ns
Delay time from SCKp↑ to SOp output**Note 2**	t_{KSO1}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 20 pF, R_b = 1.4 kΩ		10		10		10	ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 20 pF, R_b = 2.7 kΩ		10		10		10	ns

Note 1. This setting applies when DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1.

Note 2. This setting applies when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

Caution Select the TTL input buffer for the SIp pin and the N-ch open drain output (withstand voltage of V_{DD} (for the 20- to 48-pin products)/withstand voltage of EV_{DD} (for the 64-pin products)) mode for the SOp pin and SCKp pin by using a port input mode register (PIMg) and port output mode register (POMg). For V_{IH} and V_{IL}, see the DC characteristics with TTL input buffer selected.

Remark 1. R_b[Ω]: Communication line (SCKp, SOp) pull-up resistance, C_b[F]: Communication line (SCKp, SOp) load capacitance, V_b[V]: Communication line voltage

Remark 2. p: CSI number (p = 00), m: Unit number (m = 0), n: Channel number (n = 0), g: PIM and POM numbers (g = 1)

Remark 3. f_{MCK}: Serial array unit operation clock frequency
(To set this operating clock, use the CKSmn bit in the serial mode register mn (SMRmn) (m: Unit number, n: Channel number = 00).)

Remark 4. The listed times are only valid when the peripheral I/O redirect function of CSI00 is not in use.

8. In simplified SPI (CSI) communications in the master mode with devices operating at different voltage levels (1.8 V, 2.5 V, or 3 V) with the internal SCKp clock

(T_A = -40 to +105°C, 1.8 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (1/3)

Item	Symbol	Conditions		HS (High-speed Main) Mode		LS (Low-speed Main) Mode		LP (Low-power Main) Mode		Unit
				Min.	Max.	Min.	Max.	Min.	Max.	
SCKp cycle time	t_{KCY1}	t_{KCY1} ≥ 4/f_{CLK}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 30 pF, R_b = 1.4 kΩ	300		300		2300		ns
			2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 30 pF, R_b = 2.7 kΩ	500		500		2300		ns
			1.8 V ≤ EV_{DD0} < 3.3 V, 1.6 V ≤ V_b ≤ 2.0 V**Note**, C_b = 30 pF, R_b = 5.5 kΩ	1150		1150		2300		ns
SCKp high-level width	t_{KH1}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 30 pF, R_b = 1.4 kΩ		t_{KCY1}/2 - 75		t_{KCY1}/2 - 75		t_{KCY1}/2 - 75		ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 30 pF, R_b = 2.7 kΩ		t_{KCY1}/2 - 170		t_{KCY1}/2 - 170		t_{KCY1}/2 - 170		ns
		1.8 V ≤ EV_{DD0} < 3.3 V, 1.6 V ≤ V_b ≤ 2.0 V**Note**, C_b = 30 pF, R_b = 5.5 kΩ		t_{KCY1}/2 - 458		t_{KCY1}/2 - 458		t_{KCY1}/2 - 458		ns
SCKp low-level width	t_{KL1}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 30 pF, R_b = 1.4 kΩ		t_{KCY1}/2 - 12		t_{KCY1}/2 - 12		t_{KCY1}/2 - 50		ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 30 pF, R_b = 2.7 kΩ		t_{KCY1}/2 - 18		t_{KCY1}/2 - 18		t_{KCY1}/2 - 50		ns
		1.8 V ≤ EV_{DD0} < 3.3 V, 1.6 V ≤ V_b ≤ 2.0 V**Note**, C_b = 30 pF, R_b = 5.5 kΩ		t_{KCY1}/2 - 50		t_{KCY1}/2 - 50		t_{KCY1}/2 - 50		ns

Note Use this setting with EV_{DD0} ≥ V_b.

Caution **Select the TTL input buffer for the SIp pin and the N-ch open drain output (withstand voltage of V_{DD} (for the 20- to 48-pin products)/withstand voltage of EV_{DD} (for the 64-pin products)) mode for the SOp pin and SCKp pin by using a port input mode register (PIMg) and port output mode register (POMg). For V_{IH} and V_{IL}, see the DC characteristics with TTL input buffer selected.**

(Remarks are listed on page 954.)

8. In simplified SPI (CSI) communications in the master mode with devices operating at different voltage levels (1.8 V, 2.5 V, or 3 V) with the internal SCKp clock

(T_A = -40 to +105°C, 1.8 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (2/3)

Item	Symbol	Conditions	HS (High-speed Main) Mode		LS (Low-speed Main) Mode		LP (Low-power Main) Mode		Unit
			Min.	Max.	Min.	Max.	Min.	Max.	
SIp setup time (to SCKp↑)Note 1	t_{SIK1}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 30 pF, R_b = 1.4 kΩ	81		81		479		ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 30 pF, R_b = 2.7 kΩ	177		177		479		ns
		1.8 V ≤ EV_{DD0} < 3.3 V, 1.6 V ≤ V_b ≤ 2.0 VNote 2, C_b = 30 pF, R_b = 5.5 kΩ	479		479		479		ns
SIp hold time (from SCKp↑)Note 1	t_{KSI1}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 30 pF, R_b = 1.4 kΩ	19		19		19		ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 30 pF, R_b = 2.7 kΩ	19		19		19		ns
		1.8 V ≤ EV_{DD0} < 3.3 V, 1.6 V ≤ V_b ≤ 2.0 VNote 2, C_b = 30 pF, R_b = 5.5 kΩ	19		19		19		ns
Delay time from SCKp↓ to SOp outputNote 1	t_{KSO1}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 30 pF, R_b = 1.4 kΩ		100		100		100	ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 30 pF, R_b = 2.7 kΩ		195		195		195	ns
		1.8 V ≤ EV_{DD0} < 3.3 V, 1.6 V ≤ V_b ≤ 2.0 VNote 2, C_b = 30 pF, R_b = 5.5 kΩ		483		483		483	ns

Note 1. This setting applies when DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1.

Note 2. Use this setting with EV_{DD0} ≥ V_b.

Caution Select the TTL input buffer for the SIp pin and the N-ch open drain output (withstand voltage of V_{DD} (for the 20- to 48-pin products)/withstand voltage of EV_{DD} (for the 64-pin products)) mode for the SOp pin and SCKp pin by using a port input mode register (PIMg) and port output mode register (POMg). For V_{IH} and V_{IL}, see the DC characteristics with TTL input buffer selected.

(Remarks are listed on page 954.)

8. In simplified SPI (CSI) communications in the master mode with devices operating at different voltage levels (1.8 V, 2.5 V, or 3 V) with the internal SCKp clock

(T_A = -40 to +105°C, 1.8 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (3/3)

Item	Symbol	Conditions	HS (High-speed Main) Mode		LS (Low-speed Main) Mode		LP (Low-power Main) Mode		Unit
			Min.	Max.	Min.	Max.	Min.	Max.	
SIp setup time (to SCKp↓)Note 1	t_{SIK1}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 30 pF, R_b = 1.4 kΩ	44		44		110		ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 30 pF, R_b = 2.7 kΩ	44		44		110		ns
		1.8 V ≤ EV_{DD0} < 3.3 V, 1.6 V ≤ V_b ≤ 2.0 VNote 2, C_b = 30 pF, R_b = 5.5 kΩ	110		110		110		ns
SIp hold time (from SCKp↓)Note 1	t_{KSI1}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 30 pF, R_b = 1.4 kΩ	19		19		19		ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 30 pF, R_b = 2.7 kΩ	19		19		19		ns
		1.8 V ≤ EV_{DD0} < 3.3 V, 1.6 V ≤ V_b ≤ 2.0 VNote 2, C_b = 30 pF, R_b = 5.5 kΩ	19		19		19		ns
Delay time from SCKp↑ to SOp outputNote 1	t_{KSO1}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 30 pF, R_b = 1.4 kΩ		25		25		25	ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 30 pF, R_b = 2.7 kΩ		25		25		25	ns
		1.8 V ≤ EV_{DD0} < 3.3 V, 1.6 V ≤ V_b ≤ 2.0 VNote 2, C_b = 30 pF, R_b = 5.5 kΩ		25		25		25	ns

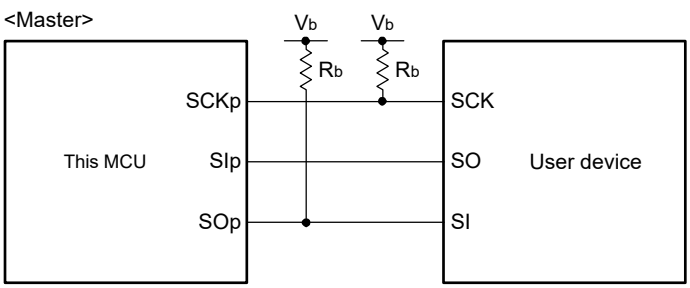
Note 1. This setting applies when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

Note 2. Use this setting with EV_{DD0} ≥ V_b.

Caution Select the TTL input buffer for the SIp pin and the N-ch open drain output (withstand voltage of V_{DD} (for the 20- to 48-pin products)/withstand voltage of EV_{DD} (for the 64-pin products)) mode for the SOp pin and SCKp pin by using a port input mode register (PIMg) and port output mode register (POMg). For V_{IH} and V_{IL}, see the DC characteristics with TTL input buffer selected.

(Remarks are listed on the next page.)

Connection in the simplified SPI (CSI) communications with devices operating at different voltage levels



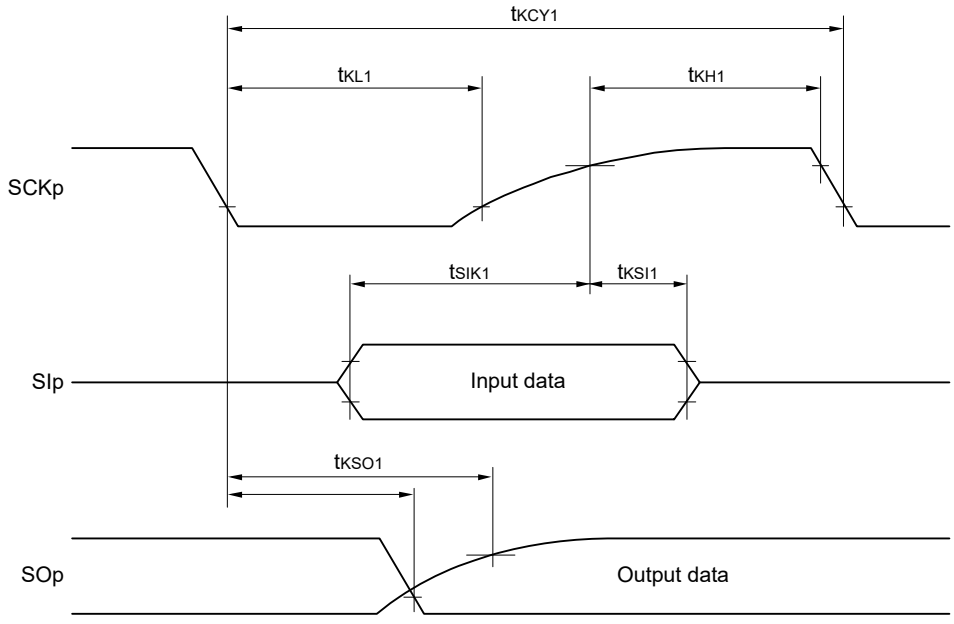
Remark 1. R_b[Ω]: Communication line (SCKp, SOp) pull-up resistance, C_b[F]: Communication line (SCKp, SOp) load capacitance, V_b[V]: Communication line voltage

Remark 2. p: CSI number (p = 00, 10, 20), m: Unit number, n: Channel number (mn = 00, 02, 10), g: PIM and POM number (g = 0, 1)

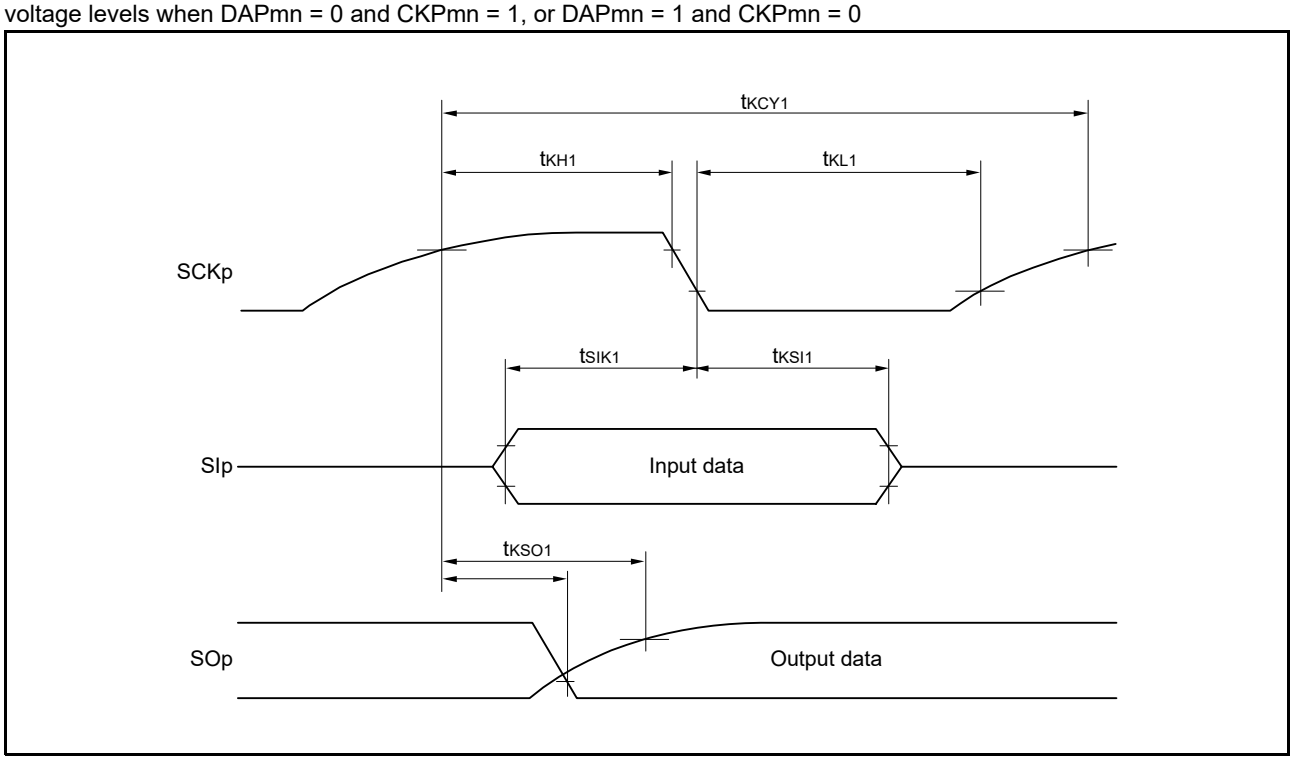
Remark 3. f_{MCK}: Serial array unit operation clock frequency
(To set this operating clock, use the CKSmn bit in the serial mode register mn (SMRmn) (m: Unit number, n: Channel number = 00, 02, 10).)

Remark 4. Communications by using CSI01, CSI11, and CSI21 with devices operating at different voltage levels are not possible. Use other CSI channels to handle such communications.

Timing of serial transfer in the simplified SPI (CSI) communications in the master mode with devices operating at different voltage levels when DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1



Timing of serial transfer in the simplified SPI (CSI) communications in the master mode with devices operating at different voltage levels when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0



Remark 1. p: CSI number (p = 00, 10, 20), m: Unit number, n: Channel number (mn = 00, 02, 10), g: PIM and POM number (g = 0, 1)

Remark 2. Communications by using CSI01, CSI11, and CSI21 with devices operating at different voltage levels are not possible. Use other CSI channels to handle such communications.

9. In simplified SPI (CSI) communications in the slave mode with devices operating at different voltage levels (1.8 V, 2.5 V, or 3 V) with the external SCKp clock

(T_A = -40 to +105°C, 1.8 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (1/2)

Item	Symbol	Conditions		HS (High-speed Main) Mode		LS (Low-speed Main) Mode		LP (Low-power Main) Mode		Unit
				Min.	Max.	Min.	Max.	Min.	Max.	
SCKp cycle time **Note 1**	t_{KCY2}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V	24 MHz < f_{MCK}	14/f_{MCK}		—		—		ns
			20 MHz < f_{MCK} ≤ 24 MHz	12/f_{MCK}		12/f_{MCK}		—		ns
			8 MHz < f_{MCK} ≤ 20 MHz	10/f_{MCK}		10/f_{MCK}		—		ns
			4 MHz < f_{MCK} ≤ 8 MHz	8/f_{MCK}		8/f_{MCK}		—		ns
			f_{MCK} ≤ 4 MHz	6/f_{MCK}		6/f_{MCK}		10/f_{MCK}		ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V,	24 MHz < f_{MCK}	20/f_{MCK}		—		—		ns
			20 MHz < f_{MCK} ≤ 24 MHz	16/f_{MCK}		16/f_{MCK}		—		ns
			16 MHz < f_{MCK} ≤ 20 MHz	14/f_{MCK}		14/f_{MCK}		—		ns
			8 MHz < f_{MCK} ≤ 16 MHz	12/f_{MCK}		12/f_{MCK}		—		ns
			4 MHz < f_{MCK} ≤ 8 MHz	8/f_{MCK}		8/f_{MCK}		—		ns
			f_{MCK} ≤ 4 MHz	6/f_{MCK}		6/f_{MCK}		10/f_{MCK}		ns
		1.8 V ≤ EV_{DD0} < 3.3 V, 1.6 V ≤ V_b ≤ 2.0 V **Note 2**	24 MHz < f_{MCK}	48/f_{MCK}		—		—		ns
			20 MHz < f_{MCK} ≤ 24 MHz	36/f_{MCK}		36/f_{MCK}		—		ns
			16 MHz < f_{MCK} ≤ 20 MHz	32/f_{MCK}		32/f_{MCK}		—		ns
			8 MHz < f_{MCK} ≤ 16 MHz	26/f_{MCK}		26/f_{MCK}		—		ns
			4 MHz < f_{MCK} ≤ 8 MHz	16/f_{MCK}		16/f_{MCK}		—		ns
			f_{MCK} ≤ 4 MHz	10/f_{MCK}		10/f_{MCK}		10/f_{MCK}		ns

(Notes and Caution are listed on the next page, and Remarks are listed on page 958.)

9. In simplified SPI (CSI) communications in the slave mode with devices operating at different voltage levels (1.8 V, 2.5 V, or 3 V) with the external SCKp clock

(T_A = -40 to +105°C, 1.8 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (2/2)

Item	Symbol	Conditions	HS (High-speed Main) Mode		LS (Low-speed Main) Mode		LP (Low-power Main) Mode		Unit
			Min.	Max.	Min.	Max.	Min.	Max.	
SCKp high-level width, low-level width	t_{KH2}, t_{KL2}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V	t_{KCY2}/2 - 12		t_{KCY2}/2 - 12		t_{KCY2}/2 - 50		ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V	t_{KCY2}/2 - 18		t_{KCY2}/2 - 18		t_{KCY2}/2 - 50		ns
		1.8 V ≤ EV_{DD0} < 3.3 V, 1.6 V ≤ V_b ≤ 2.0 V**Note 2**	t_{KCY2}/2 - 50		t_{KCY2}/2 - 50		t_{KCY2}/2 - 50		ns
SIp setup time (to SCKp↑)**Note 3**	t_{SIK2}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V	1/f_{MCK} + 20		1/f_{MCK} + 20		1/f_{MCK} + 30		ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V	1/f_{MCK} + 20		1/f_{MCK} + 20		1/f_{MCK} + 30		ns
		1.8 V ≤ EV_{DD0} < 3.3 V, 1.6 V ≤ V_b ≤ 2.0 V**Note 2**	1/f_{MCK} + 30		1/f_{MCK} + 30		1/f_{MCK} + 30		ns
SIp hold time (from SCKp↑)**Note 3**	t_{KSI2}		1/f_{MCK} + 31		1/f_{MCK} + 31		1/f_{MCK} + 31		ns
Delay time from SCKp↓ to SOp output**Note 4**	t_{KSO2}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 30 pF, R_b = 1.4 kΩ		2/f_{MCK} + 120		2/f_{MCK} + 120		2/f_{MCK} + 573	ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 30 pF, R_b = 2.7 kΩ		2/f_{MCK} + 214		2/f_{MCK} + 214		2/f_{MCK} + 573	ns
		1.8 V ≤ EV_{DD0} < 3.3 V, 1.6 V ≤ V_b ≤ 2.0 V**Note 2**, C_b = 30 pF, R_b = 5.5 kΩ		2/f_{MCK} + 573		2/f_{MCK} + 573		2/f_{MCK} + 573	ns

Note 1. Transfer rate in the SNOOZE mode is 1 Mbps at the maximum.

Note 2. Use this setting with EV_{DD0} ≥ V_b.

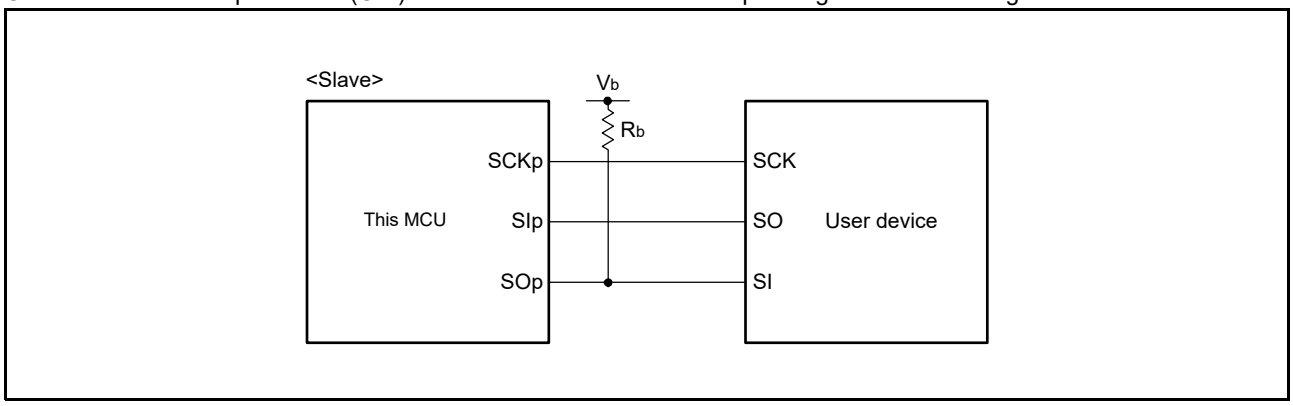
Note 3. This setting applies when DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The SIp setup time becomes "to SCKp↓" and SIp hold time becomes "from SCKp↓" when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

Note 4. This setting applies when DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The delay time to SOp output becomes "from SCKp↑" when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

Caution Select the TTL input buffer for the SIp pin and the N-ch open drain output (withstand voltage of V_{DD} (for the 20- to 48-pin products)/withstand voltage of EV_{DD} (for the 64-pin products)) mode for the SOp pin and SCKp pin by using a port input mode register (PIMg) and port output mode register (POMg). For V_{IH} and V_{IL}, see the DC characteristics with TTL input buffer selected.

(Remarks are listed on the next page.)

Connection in the simplified SPI (CSI) communications with devices operating at different voltage levels



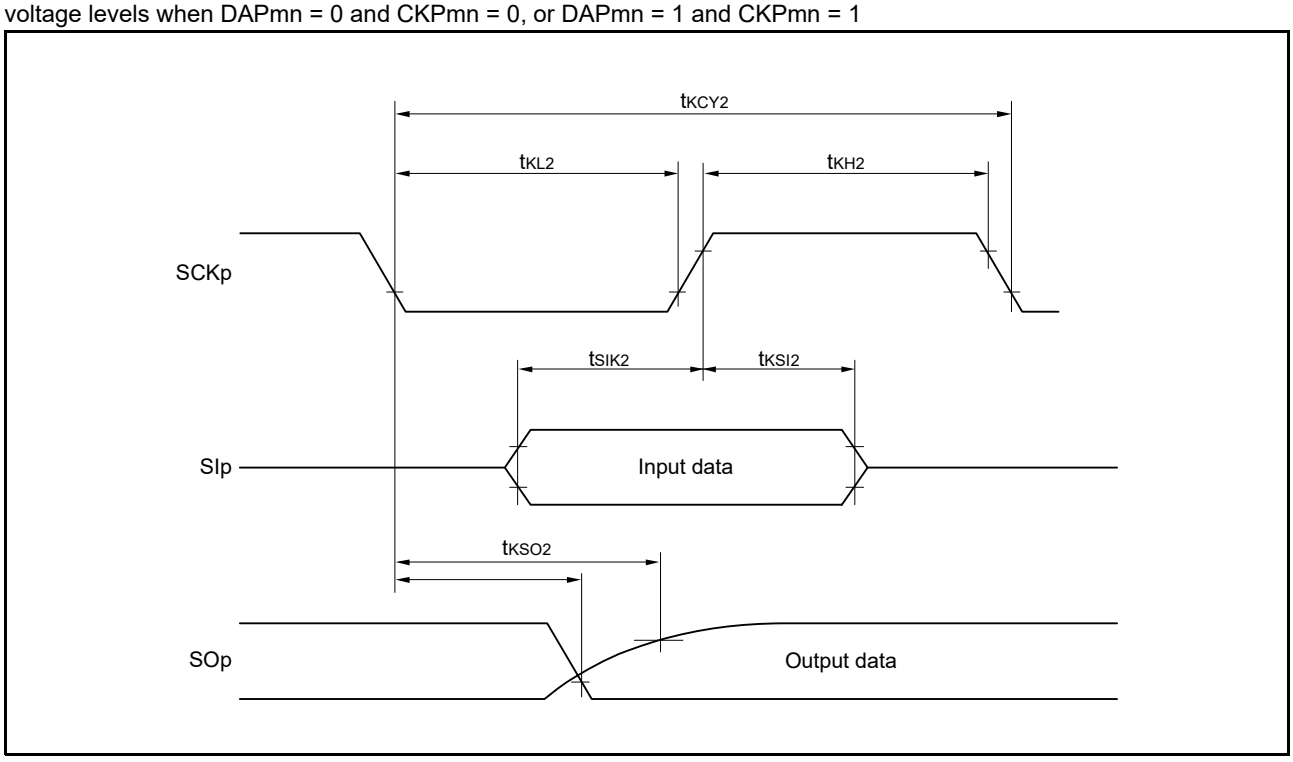
Remark 1. R_b[Ω]: Communication line (SOp) pull-up resistance, C_b[F]: Communication line (SOp) load capacitance, V_b[V]: Communication line voltage

Remark 2. p: CSI number (p = 00, 10, 20), m: Unit number, n: Channel number (mn = 00, 02, 10), g: PIM and POM number (g = 0, 1)

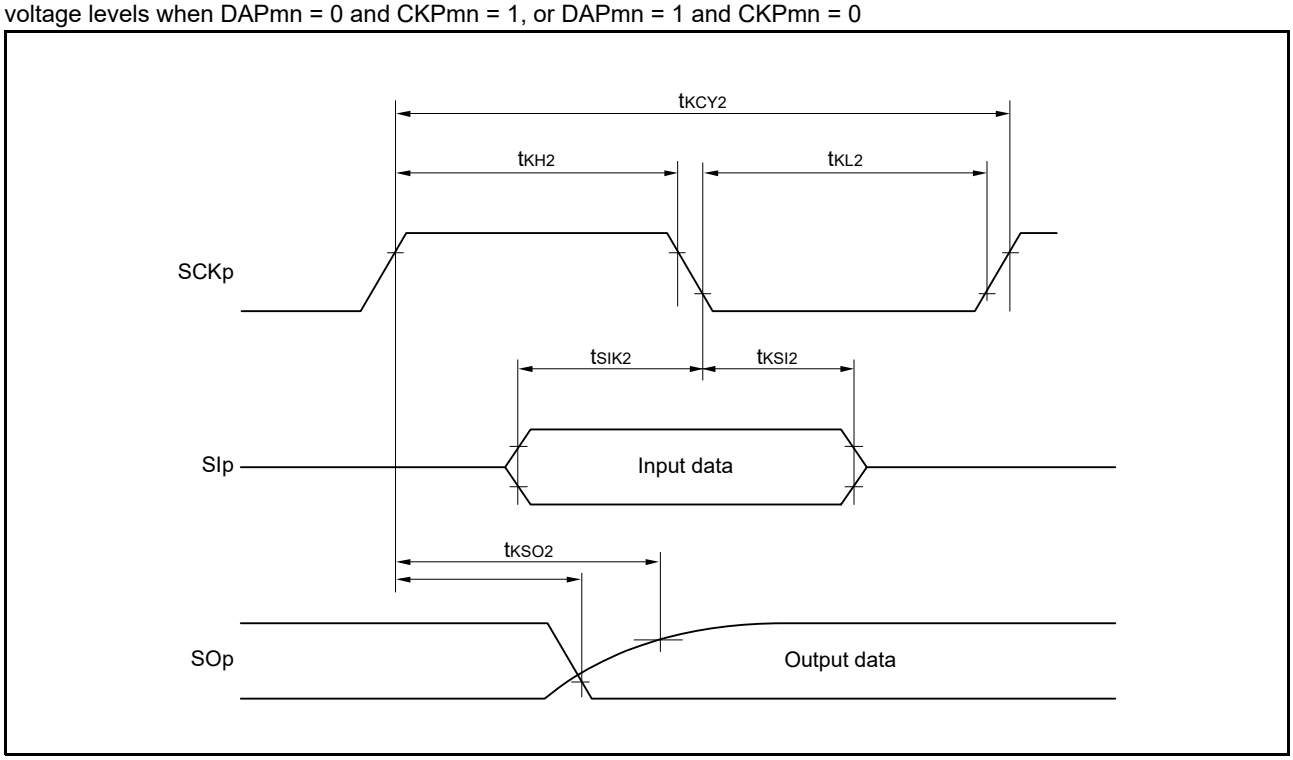
Remark 3. f_{MCK}: Serial array unit operation clock frequency
(To set this operating clock, use the CKSmn bit in the serial mode register mn (SMRmn) (m: Unit number, n: Channel number = 00, 02, 10).)

Remark 4. Communications by using CSI01, CSI11, and CSI21 with devices operating at different voltage levels are not possible. Use other CSI channels to handle such communications.

Timing of serial transfer in the simplified SPI (CSI) communications in the slave mode with devices operating at different voltage levels when DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1



Timing of serial transfer in the simplified SPI (CSI) communications in the slave mode with devices operating at different voltage levels when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0



Remark 1. p: CSI number (p = 00, 10, 20), m: Unit number, n: Channel number (mn = 00, 02, 10), g: PIM and POM number (g = 0, 1)

Remark 2. Communications by using CSI01, CSI11, and CSI21 with devices operating at different voltage levels are not possible. Use other CSI channels to handle such communications.

10. Simplified I²C communications with devices operating at different voltage levels (1.8 V, 2.5 V, or 3 V)

(T_A = -40 to +105°C, 1.8 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (1/2)

Item	Symbol	Conditions	HS (High-speed Main) Mode		LS (Low-speed Main) Mode		LP (Low-power Main) Mode		Unit
			Min.	Max.	Min.	Max.	Min.	Max.	
SCLr clock frequency	f_{SCL}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 50 pF, R_b = 2.7 kΩ		1000 **Note 1**		1000 **Note 1**		300 **Note 1**	kHz
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 50 pF, R_b = 2.7 kΩ		1000 **Note 1**		1000 **Note 1**		300 **Note 1**	kHz
		4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 100 pF, R_b = 2.8 kΩ		400 **Note 1**		400 **Note 1**		300 **Note 1**	kHz
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 100 pF, R_b = 2.7 kΩ		400 **Note 1**		400 **Note 1**		300 **Note 1**	kHz
		1.8 V ≤ EV_{DD0} < 3.3 V, 1.6 V ≤ V_b ≤ 2.0 V **Note 2**, C_b = 100 pF, R_b = 5.5 kΩ		300 **Note 1**		300 **Note 1**		300 **Note 1**	kHz
Hold time when SCLr is low	t_{LOW}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 50 pF, R_b = 2.7 kΩ	475		475		1550		ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 50 pF, R_b = 2.7 kΩ	475		475		1550		ns
		4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 100 pF, R_b = 2.8 kΩ	1150		1550		1550		ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 100 pF, R_b = 2.7 kΩ	1150		1550		1550		ns
		1.8 V ≤ EV_{DD0} < 3.3 V, 1.6 V ≤ V_b ≤ 2.0 V **Note 2**, C_b = 100 pF, R_b = 5.5 kΩ	1550		1550		1550		ns
Hold time when SCLr is high	t_{HIGH}	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 50 pF, R_b = 2.7 kΩ	245		245		610		ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 50 pF, R_b = 2.7 kΩ	200		200		610		ns
		4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 100 pF, R_b = 2.8 kΩ	675		675		610		ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 100 pF, R_b = 2.7 kΩ	600		600		610		ns
		1.8 V ≤ EV_{DD0} < 3.3 V, 1.6 V ≤ V_b ≤ 2.0 V **Note 2**, C_b = 100 pF, R_b = 5.5 kΩ	610		610		610		ns

10. Simplified I²C communications with devices operating at different voltage levels (1.8 V, 2.5 V, and 3 V)

(T_A = -40 to +105°C, 1.8 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V) (2/2)

Item	Symbol	Conditions	HS (High-speed Main) Mode		LS (Low-speed Main) Mode		LP (Low-power Main) Mode		Unit
			Min.	Max.	Min.	Max.	Min.	Max.	
Data setup time (reception)	$t_{SU:DAT}$	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 50 pF, R_b = 2.7 kΩ	1/f_{MCK} + 135 **Note 3**		1/f_{MCK} + 135 **Note 3**		1/f_{MCK} + 190 **Note 3**		ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 50 pF, R_b = 2.7 kΩ	1/f_{MCK} + 135 **Note 3**		1/f_{MCK} + 135 **Note 3**		1/f_{MCK} + 190 **Note 3**		ns
		4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 100 pF, R_b = 2.8 kΩ	1/f_{MCK} + 190 **Note 3**		1/f_{MCK} + 190 **Note 3**		1/f_{MCK} + 190 **Note 3**		ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 100 pF, R_b = 2.7 kΩ	1/f_{MCK} + 190 **Note 3**		1/f_{MCK} + 190 **Note 3**		1/f_{MCK} + 190 **Note 3**		ns
		1.8 V ≤ EV_{DD0} < 3.3 V, 1.6 V ≤ V_b ≤ 2.0 V **Note 2**, C_b = 100 pF, R_b = 5.5 kΩ	1/f_{MCK} + 190 **Note 3**		1/f_{MCK} + 190 **Note 3**		1/f_{MCK} + 190 **Note 3**		ns
Data hold time (transmission)	$t_{HD:DAT}$	4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 50 pF, R_b = 2.7 kΩ	0	305	0	305	0	305	ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 50 pF, R_b = 2.7 kΩ	0	305	0	305	0	305	ns
		4.0 V ≤ EV_{DD0} ≤ 5.5 V, 2.7 V ≤ V_b ≤ 4.0 V, C_b = 100 pF, R_b = 2.8 kΩ	0	355	0	355	0	355	ns
		2.7 V ≤ EV_{DD0} < 4.0 V, 2.3 V ≤ V_b ≤ 2.7 V, C_b = 100 pF, R_b = 2.7 kΩ	0	355	0	355	0	355	ns
		1.8 V ≤ EV_{DD0} < 3.3 V, 1.6 V ≤ V_b ≤ 2.0 V **Note 2**, C_b = 100 pF, R_b = 5.5 kΩ	0	405	0	405	0	405	ns

Note 1. The listed times must be no greater than f_{MCK}/4.

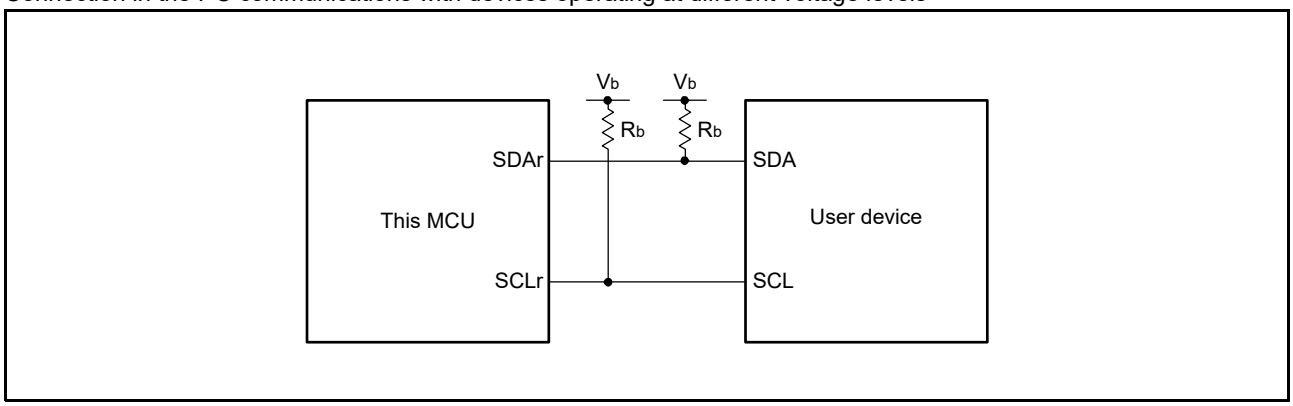
Note 2. Use this setting with EV_{DD0} ≥ V_b.

Note 3. Set f_{MCK} so that it will not exceed the hold time when SCLr is low or high.

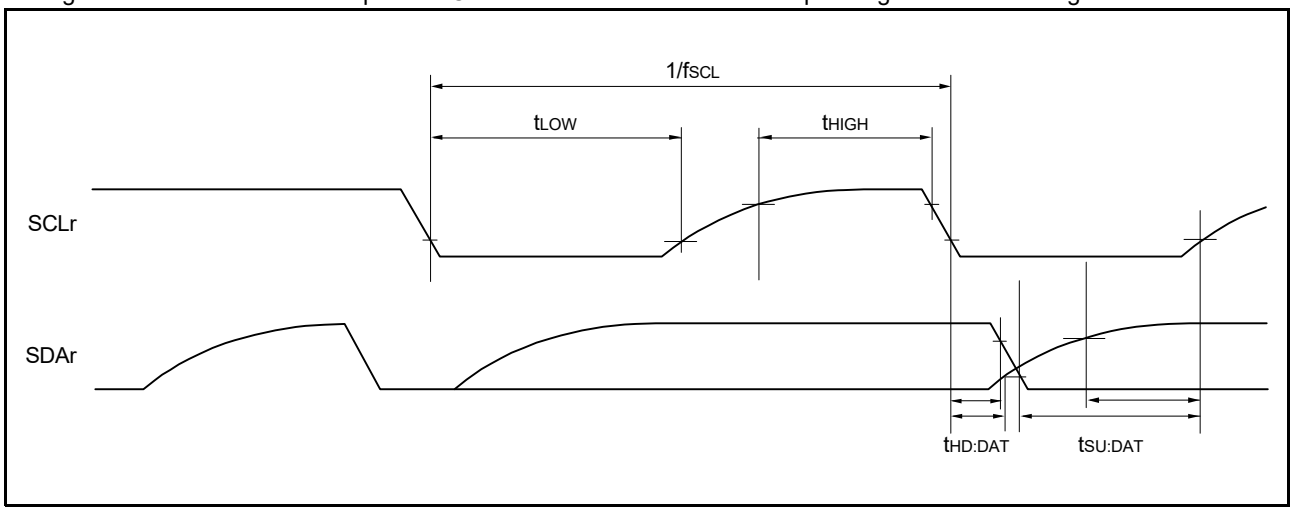
Caution Select the TTL input buffer and the N-ch open drain output (withstand voltage of V_{DD} (for the 20- to 48-pin products)/withstand voltage of EV_{DD} (for the 64-pin products)) mode for the SDAr pin and the N-ch open drain output (withstand voltage of V_{DD} (for the 20- to 48-pin products)/withstand voltage of EV_{DD} (for the 64-pin products)) mode for the SCLr pin by using a port input mode register (PIMg) and port output mode register (POMg). For V_{IH} and V_{IL}, see the DC characteristics with TTL input buffer selected.

(Remarks are listed on the next page.)

Connection in the I²C communications with devices operating at different voltage levels



Timing of serial transfer in the simplified I²C communications with devices operating at different voltage levels



Remark 1. R_b[Ω]: Communication line (SDAr, SCLr) pull-up resistance, C_b[F]: Communication line (SDAr, SCLr) load capacitance, V_b[V]: Communication line voltage

Remark 2. r: IIC number (r = 00, 10, 20), g: PIM and POM number (g = 0, 1)

Remark 3. f_{MCK}: Serial array unit operation clock frequency
(To set this operating clock, use the CKSmn bit in the serial mode register mn (SMRmn) (m: Unit number, n: Channel number = 00, 02, 10).)

28.5.2 Serial interface UARTA

(T_A = -40 to +105°C, 1.6 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V)

Item	Symbol	Conditions	Min.	Typ.	Max.	Unit
Transfer rate			200	0	153600	bps

Caution Select the normal input buffer for the RxDAn pin and the normal output mode for the TxDAn pin by using a port input mode register (PIMg) and port output mode register (POMg).

Remark 1. g: PIM number (g = 0, 3, 7), h: POM number (h = 0, 3, 7)
Remark 2. n: Unit number (n = 0)

28.5.3 Serial interface IICA

1. I^2C standard mode

(T_A = -40 to +105°C, 1.6 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V)

Item	Symbol	Conditions	Min.	Typ.	Max.	Unit
SCLA0 clock frequency	f_{SCL}	Standard mode: f_{CLK} ≥ 1 MHz	0		100	kHz
Setup time of restart condition	$t_{SU:STA}$		4.7			μs
Hold time[Note 1]	$t_{HD:STA}$		4.0			μs
Hold time when SCLA0 is low	t_{LOW}		4.7			μs
Hold time when SCLA0 is high	t_{HIGH}		4.0			μs
Data setup time (reception)	$t_{SU:DAT}$		250			ns
Data hold time (transmission)[Note 2]	$t_{HD:DAT}$		0		3.45	μs
Setup time of stop condition	$t_{SU:STO}$		4.0			μs
Bus-free time	t_{BUF}		4.7			μs

Note 1. The first clock pulse is generated after this period when the start or restart condition is detected.

Note 2. The maximum value of $t_{HD:DAT}$ applies to normal transfer. The clock stretching will be inserted on reception of an acknowledgment (ACK) signal.

Caution **The listed frequency and times apply even when the PIOR02 bit in the peripheral I/O redirection register (PIOR0) is 1. In such cases, the pin characteristics (I_{OH1}, I_{OL1}, V_{OH1}, V_{OL1}) must satisfy the values in the redirect destination.**

Remark The maximum value of communication line capacitance (C_b) and communication line pull-up resistor (R_b) are as follows.
C_b = 400 pF, R_b = 2.7 kΩ

2. I^2C fast mode

(T_A = -40 to +105°C, 1.6 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V)

Item	Symbol	Conditions	Min.	Typ.	Max.	Unit
SCLA0 clock frequency	f_{SCL}	Fast mode: f_{CLK} ≥ 3.5 MHz 1.8 V ≤ EV_{DD0} ≤ 5.5 V	0		400	kHz
Setup time of restart condition	$t_{SU:STA}$	1.8 V ≤ EV_{DD0} ≤ 5.5 V	0.6			µs
Hold time[Note 1]	$t_{HD:STA}$	1.8 V ≤ EV_{DD0} ≤ 5.5 V	0.6			µs
Hold time when SCLA0 is low	t_{LOW}	1.8 V ≤ EV_{DD0} ≤ 5.5 V	1.3			µs
Hold time when SCLA0 is high	t_{HIGH}	1.8 V ≤ EV_{DD0} ≤ 5.5 V	0.6			µs
Data setup time (reception)	$t_{SU:DAT}$	1.8 V ≤ EV_{DD0} ≤ 5.5 V	100			ns
Data hold time (transmission)[Note 2]	$t_{HD:DAT}$	1.8 V ≤ EV_{DD0} ≤ 5.5 V	0		0.9	µs
Setup time of stop condition	$t_{SU:STO}$	1.8 V ≤ EV_{DD0} ≤ 5.5 V	0.6			µs
Bus-free time	t_{BUF}	1.8 V ≤ EV_{DD0} ≤ 5.5 V	1.3			µs

Note 1. The first clock pulse is generated after this period when the start or restart condition is detected.

Note 2. The maximum value of $t_{HD:DAT}$ applies to normal transfer. The clock stretching will be inserted on reception of an acknowledgment (ACK) signal.

Caution **The values in the above table apply even when the PIOR02 bit in the peripheral I/O redirection register (PIOR0) is 1. In such cases, the pin characteristics (I_{OH1}, I_{OL1}, V_{OH1}, V_{OL1}) must satisfy the values in the redirect destination.**

Remark The maximum value of communication line capacitance (C_b) and communication line pull-up resistor (R_b) are as follows.
C_b = 320 pF, R_b = 1.1 kΩ

3. I^2C fast mode plus

(T_A = -40 to +105°C, 1.6 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V)

Item	Symbol	Conditions	Min.	Typ.	Max.	Unit
SCLA0 clock frequency	f_{SCL}	Fast mode plus: f_{CLK} ≥ 10 MHz 2.7 V ≤ EV_{DD0} ≤ 5.5 V	0		1000	kHz
Setup time of restart condition	$t_{SU:STA}$	2.7 V ≤ EV_{DD0} ≤ 5.5 V	0.26			µs
Hold time[Note 1]	$t_{HD:STA}$	2.7 V ≤ EV_{DD0} ≤ 5.5 V	0.26			µs
Hold time when SCLA0 is low	t_{LOW}	2.7 V ≤ EV_{DD0} ≤ 5.5 V	0.5			µs
Hold time when SCLA0 is high	t_{HIGH}	2.7 V ≤ EV_{DD0} ≤ 5.5 V	0.26			µs
Data setup time (reception)	$t_{SU:DAT}$	2.7 V ≤ EV_{DD0} ≤ 5.5 V	50			ns
Data hold time (transmission)[Note 2]	$t_{HD:DAT}$	2.7 V ≤ EV_{DD0} ≤ 5.5 V	0		0.45	µs
Setup time of stop condition	$t_{SU:STO}$	2.7 V ≤ EV_{DD0} ≤ 5.5 V	0.26			µs
Bus-free time	t_{BUF}	2.7 V ≤ EV_{DD0} ≤ 5.5 V	0.5			µs

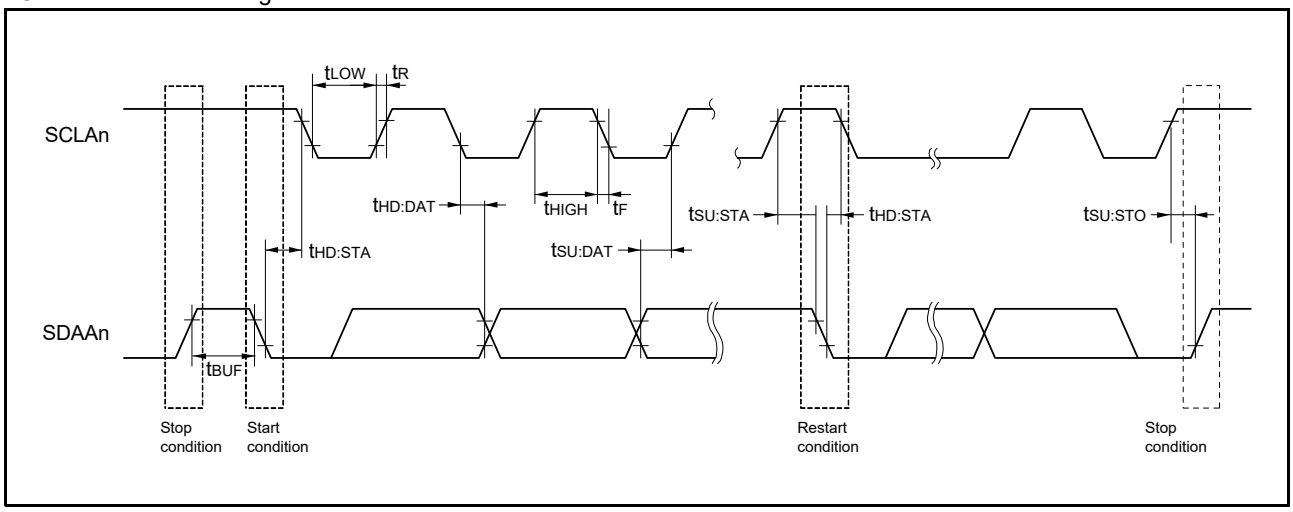
Note 1. The first clock pulse is generated after this period when the start or restart condition is detected.

Note 2. The maximum value of $t_{HD:DAT}$ applies to normal transfer. The clock stretching will be inserted on reception of an acknowledgment (ACK) signal.

Caution **The values in the above table apply even when the PIOR02 bit in the peripheral I/O redirection register (PIOR0) is 1. In such cases, the pin characteristics (I_{OH1}, I_{OL1}, V_{OH1}, V_{OL1}) must satisfy the values in the redirect destination.**

Remark The maximum value of communication line capacitance (C_b) and communication line pull-up resistor (R_b) are as follows.
C_b = 120 pF, R_b = 1.1 kΩ

IICA serial transfer timing



Remark n = 0

28.6 Characteristics of the Analog Circuits

28.6.1 Characteristics of the A/D converter for T_A = –40 to 85°C

Reference for the characteristics of the A/D converter

Reference voltage / Input Channel	Reference voltage (+) = AV_{REFP} / Reference voltage (–) = AV_{REFM}	Reference voltage (+) = V_{DD} / Reference voltage (–) = V_{SS}	Reference voltage (+) = V_{BGR} / Reference voltage (–) = AV_{REFM}
ANI0 to ANI7	See **28.6.1 (1)**	See **28.6.1 (3)**	See **28.6.1 (4)**
ANI16 to ANI19	See **28.6.1 (2)**		
Internal reference voltage, temperature sensor output voltage	See **28.6.1 (1)**		—

(1) Reference voltage (+) = AV_{REFP}/ANI0 (ADREFP1 = 0, ADREFP0 = 1), reference voltage (-) = AV_{REFM}/ANI1 (ADREFM = 1), conversion target: ANI2 to ANI7, internal reference voltage, temperature sensor output voltage

(T_A = –40 to +85°C, 1.6 V ≤ AV_{REFP} ≤ V_{DD} ≤ 5.5 V, V_{SS} = 0 V, reference voltage (+) = AV_{REFP}, reference voltage (–) = AV_{REFM} = 0 V)

Item	Symbol	Conditions		Min.	Typ.	Max.	Unit
Resolution	RES			8		10	bit
Overall error**Note 1**	AINL	10-bit resolution AV_{REFP} = V_{DD}**Note 2**	1.8 V ≤ AV_{REFP} ≤ 5.5 V		1.2	±3.5	LSB
			1.6 V ≤ AV_{REFP} ≤ 5.5 V**Note 3**		1.2	±7.0	LSB
Conversion time	t_{CONV}	10-bit resolution conversion target: ANI2 to ANI7	3.6 V ≤ V_{DD} ≤ 5.5 V	2.125		39	µs
			2.7 V ≤ V_{DD} ≤ 5.5 V	3.1875		39	µs
			1.8 V ≤ V_{DD} ≤ 5.5 V	17		39	µs
			1.6 V ≤ V_{DD} ≤ 5.5 V	57		95	µs
		10-bit resolution conversion target: Internal reference voltage, temperature sensor output voltage	3.6 V ≤ V_{DD} ≤ 5.5 V	2.375		39	µs
			2.7 V ≤ V_{DD} ≤ 5.5 V	3.5625		39	µs
			1.8 V ≤ V_{DD} ≤ 5.5 V	17		39	µs
Zero-scale error**Notes 1, 4**	E_{ZS}	10-bit resolution AV_{REFP} = V_{DD}**Note 2**	1.8 V ≤ AV_{REFP} ≤ 5.5 V			±0.25	%FSR
			1.6 V ≤ AV_{REFP} ≤ 5.5 V**Note 3**			±0.50	%FSR
Full-scale error**Notes 1, 4**	E_{FS}	10-bit resolution AV_{REFP} = V_{DD}**Note 2**	1.8 V ≤ AV_{REFP} ≤ 5.5 V			±0.25	%FSR
			1.6 V ≤ AV_{REFP} ≤ 5.5 V**Note 3**			±0.50	%FSR
Integral linearity error**Note 1**	ILE	10-bit resolution AV_{REFP} = V_{DD}**Note 2**	1.8 V ≤ AV_{REFP} ≤ 5.5 V			±2.5	LSB
			1.6 V ≤ AV_{REFP} ≤ 5.5 V**Note 3**			±5.0	LSB
Differential linearity error**Note 1**	DLE	10-bit resolution AV_{REFP} = V_{DD}**Note 2**	1.8 V ≤ AV_{REFP} ≤ 5.5 V			±1.5	LSB
			1.6 V ≤ AV_{REFP} ≤ 5.5 V**Note 3**			±2.0	LSB
Analog input voltage	V_{AIN}	ANI2 to ANI7		0		AV_{REFP}	V
		Internal reference voltage (1.8 V ≤ V_{DD} ≤ 5.5 V)		V_{BGR}**Note 5**			V
		Temperature sensor output voltage (1.8 V ≤ V_{DD} ≤ 5.5 V)		V_{TMPS25}**Note 5**			V

(Notes are listed on the next page.)

Note 1. This value does not include the quantization error (±1/2 LSB).

Note 2. When $AV_{REFP} < V_{DD}$, the maximum values are as follows.

- Overall error: Add ±1.0 LSB to the maximum value when $V_{DD} = AV_{REFP}$.
- Zero-scale/full-scale error: Add ±0.05%FSR to the maximum value when $V_{DD} = AV_{REFP}$.
- Integral linearity error and differential linearity error: Add ±0.5 LSB to the maximum value when $V_{DD} = AV_{REFP}$.

Note 3. The listed value applies when the settings of the maximum and minimum conversion time values are respectively 57 µs and 95 µs.

Note 4. This value is indicated as a ratio (%FSR) to the full-scale value.

Note 5. See **28.6.3 Characteristics of the temperature sensor and internal reference voltage**.

(2) Reference voltage (+) = AVREFP/ANI0 (ADREFP1 = 0, ADREFP0 = 1), reference voltage (-) = AVREFM/ANI1 (ADREFM = 1), conversion target: ANI16 to ANI19

(TA = –40 to +85°C, 1.6 V ≤ AVREFP ≤ VDD ≤ 5.5 V, VSS = 0 V, reference voltage (+) = AVREFP, reference voltage (–) = AVREFM = 0 V)

Item	Symbol	Conditions		Min.	Typ.	Max.	Unit
Resolution	RES			8		10	bit
Overall error**Note 1**	AINL	10-bit resolution AVREFP = VDD**Note 2**	1.8 V ≤ AVREFP ≤ 5.5 V		1.2	±5.0	LSB
			1.6 V ≤ AVREFP ≤ 5.5 V**Note 3**		1.2	±8.5	LSB
Conversion time	tCONV	10-bit resolution conversion target: ANI16 to ANI19	3.6 V ≤ VDD ≤ 5.5 V	2.125		39	μs
			2.7 V ≤ VDD ≤ 5.5 V	3.1875		39	μs
			1.8 V ≤ VDD ≤ 5.5 V	17		39	μs
			1.6 V ≤ VDD ≤ 5.5 V	57		95	μs
Zero-scale error**Notes 1, 4**	EZS	10-bit resolution AVREFP = VDD**Note 2**	1.8 V ≤ AVREFP ≤ 5.5 V			±0.35	%FSR
			1.6 V ≤ AVREFP ≤ 5.5 V**Note 3**			±0.60	%FSR
Full-scale error**Notes 1, 4**	EFS	10-bit resolution AVREFP = VDD**Note 2**	1.8 V ≤ AVREFP ≤ 5.5 V			±0.35	%FSR
			1.6 V ≤ AVREFP ≤ 5.5 V**Note 3**			±0.60	%FSR
Integral linearity error**Note 1**	ILE	10-bit resolution AVREFP = VDD**Note 2**	1.8 V ≤ AVREFP ≤ 5.5 V			±3.5	LSB
			1.6 V ≤ AVREFP ≤ 5.5 V**Note 3**			±6.0	LSB
Differential linearity error**Note 1**	DLE	10-bit resolution AVREFP = VDD**Note 2**	1.8 V ≤ AVREFP ≤ 5.5 V			±2.0	LSB
			1.6 V ≤ AVREFP ≤ 5.5 V**Note 3**			±2.5	LSB
Analog input voltage	VAIN	ANI16 to ANI19		0		AVREFP	V

Note 1. This value does not include the quantization error (±1/2 LSB).

Note 2. When AVREFP < VDD, the maximum values are as follows.
- Overall error: Add ±4.0 LSB to the maximum value when VDD = AVREFP.
- Zero-scale/full-scale error: Add ±0.20%FSR to the maximum value when VDD = AVREFP.
- Integral linearity error/differential linearity error: Add ±2.0 LSB to the maximum value when VDD = AVREFP.

Note 3. The listed value applies when the settings of the maximum and minimum conversion time values are respectively 57 μs and 95 μs.

Note 4. This value is indicated as a ratio (%FSR) to the full-scale value.

(3) Reference voltage (+) = V_{DD} (ADREFP1 = 0, ADREFP0 = 0), reference voltage (-) = Vss (ADREFM = 0), conversion target: ANI0 to ANI7, ANI16 to ANI19, internal reference voltage**Note 1**, temperature sensor output voltage**Note 1**

(T_A = –40 to +85°C, 1.6 V ≤ V_{DD} ≤ 5.5 V, V_{SS} = 0 V, reference voltage (+) = AV_{REFP}, reference voltage (–) = V_{SS})

Item	Symbol	Conditions		Min.	Typ.	Max.	Unit
Resolution	RES			8		10	bit
Overall error**Note 2**	AINL	10-bit resolution	1.8 V ≤ V_{DD} ≤ 5.5 V		1.2	±7.0	LSB
			1.6 V ≤ V_{DD} ≤ 5.5 V**Note 3**		1.2	±10.5	LSB
Conversion time	t_{CONV}	10-bit resolution conversion target: ANI0 to ANI7, ANI16 to ANI19	3.6 V ≤ V_{DD} ≤ 5.5 V	2.125		39	µs
			2.7 V ≤ V_{DD} ≤ 5.5 V	3.1875		39	µs
			1.8 V ≤ V_{DD} ≤ 5.5 V	17		39	µs
			1.6 V ≤ V_{DD} ≤ 5.5 V	57		95	µs
		10-bit resolution conversion target: Internal reference voltage, temperature sensor output voltage	3.6 V ≤ V_{DD} ≤ 5.5 V	2.375		39	µs
			2.7 V ≤ V_{DD} ≤ 5.5 V	3.5625		39	µs
			1.8 V ≤ V_{DD} ≤ 5.5 V	17		39	µs
Zero-scale error**Notes 2, 4**	E_{ZS}	10-bit resolution	1.8 V ≤ V_{DD} ≤ 5.5 V			±0.60	%FSR
			1.6 V ≤ V_{DD} ≤ 5.5 V**Note 3**			±0.85	%FSR
Full-scale error**Notes 2, 4**	E_{FS}	10-bit resolution	1.8 V ≤ V_{DD} ≤ 5.5 V			±0.60	%FSR
			1.6 V ≤ V_{DD} ≤ 5.5 V**Note 3**			±0.85	%FSR
Integral linearity error**Note 2**	ILE	10-bit resolution	1.8 V ≤ V_{DD} ≤ 5.5 V			±4.0	LSB
			1.6 V ≤ V_{DD} ≤ 5.5 V**Note 3**			±6.0	LSB
Differential linearity error**Note 2**	DLE	10-bit resolution	1.8 V ≤ V_{DD} ≤ 5.5 V			±2.0	LSB
			1.6 V ≤ V_{DD} ≤ 5.5 V**Note 3**			±2.5	LSB
Analog input voltage	V_{AIN}	ANI0 to ANI7, ANI16 to ANI19		0		V_{DD}	V
		Internal reference voltage (1.8 V ≤ V_{DD} ≤ 5.5 V)		V_{BGR}**Note 5**			V
		Temperature sensor output voltage (1.8 V ≤ V_{DD} ≤ 5.5 V)		V_{TMPS25}**Note 5**			V

Note 1. If the internal reference voltage or temperature sensor output voltage is to be A/D converted, V_{DD} must be at least 1.8 V.

Note 2. This value does not include the quantization error (±1/2 LSB).

Note 3. The listed value applies when the settings of the maximum and minimum conversion time values are respectively 57 µs and 95 µs.

Note 4. This value is indicated as a ratio (%FSR) to the full-scale value.

Note 5. See **28.6.3 Characteristics of the temperature sensor and internal reference voltage**.

(4) Reference voltage (+) = Internal reference voltage (ADREFP1 = 1, ADREFP0 = 1), reference voltage (-) = AVREFM/ANI1 (ADREFM = 1), conversion target: ANI0, ANI2 to ANI7, ANI16 to ANI19

(TA = –40 to +85°C, 1.8 V ≤ VDD ≤ 5.5 V, VSS = 0 V, reference voltage (+) = VBGR Note 1, reference voltage (–) = AVREFM Note 2 = 0 V)

Item	Symbol	Conditions	Min.	Typ.	Max.	Unit
Resolution	RES		8			bit
Conversion time	tCONV		17		39	μs
Zero-scale error Notes 3, 4	EZS				±0.60	%FSR
Integral linearity error Note 3	ILE				±2.0	LSB
Differential linearity error Note 3	DLE				±1.0	LSB
Analog input voltage	VAIN		0		VBGR Note 1	V

Note 1. See **28.6.3 Characteristics of the temperature sensor and internal reference voltage**.

Note 2. When reference voltage (–) = VSS, the maximum values are as follows.

- Zero-scale error: Add ±0.35%FSR to the maximum value when reference voltage (–) = AVREFM.
- Integral linearity error: Add ±0.5 LSB to the maximum value when reference voltage (–) = AVREFM.
- Differential linearity error: Add ±0.2 LSB to the maximum value when reference voltage (–) = AVREFM.

Note 3. This value does not include the quantization error (±1/2 LSB).

Note 4. This value is indicated as a ratio (%FSR) to the full-scale value.

28.6.2 Characteristics of the A/D converter for T_A = –40 to 105°C

Reference for the characteristics of the A/D converter

Reference voltage / Input Channel	Reference voltage (+) = AV_{REFP} / Reference voltage (–) = AV_{REFM}	Reference voltage (+) = V_{DD} / Reference voltage (–) = V_{SS}	Reference voltage (+) = V_{BGR} / Reference voltage (–) = AV_{REFM}
ANI0 to ANI7	See **28.6.2 (1)**	See **28.6.2 (3)**	See **28.6.2 (4)**
ANI16 to ANI19	See **28.6.2 (2)**		
Internal reference voltage, temperature sensor output voltage	See **28.6.2 (1)**		—

(1) Reference voltage (+) = AV_{REFP}/ANI0 (ADREFP1 = 0, ADREFP0 = 1), reference voltage (-) = AV_{REFM}/ANI1 (ADREFM = 1), conversion target: ANI2 to ANI7, internal reference voltage, temperature sensor output voltage

(T_A = –40 to +105°C, 2.4 V ≤ AV_{REFP} ≤ V_{DD} ≤ 5.5 V, V_{SS} = 0 V, reference voltage (+) = AV_{REFP}, reference voltage (–) = AV_{REFM} = 0 V)

Item	Symbol	Conditions		Min.	Typ.	Max.	Unit
Resolution	RES			8		10	bit
Overall error**Note 1**	AINL	10-bit resolution AV_{REFP} = V_{DD}**Note 2**	2.4 V ≤ AV_{REFP} ≤ 5.5 V		1.2	±3.5	LSB
Conversion time	t_{CONV}	10-bit resolution conversion target: ANI2 to ANI7	3.6 V ≤ V_{DD} ≤ 5.5 V	2.125		39	μs
			2.7 V ≤ V_{DD} ≤ 5.5 V	3.1875		39	μs
			2.4 V ≤ V_{DD} ≤ 5.5 V	17		39	μs
		10-bit resolution conversion target: Internal reference voltage, temperature sensor output voltage	3.6 V ≤ V_{DD} ≤ 5.5 V	2.375		39	μs
			2.7 V ≤ V_{DD} ≤ 5.5 V	3.5625		39	μs
			2.4 V ≤ V_{DD} ≤ 5.5 V	17		39	μs
Zero-scale error**Notes 1, 3**	E_{ZS}	10-bit resolution AV_{REFP} = V_{DD}**Note 2**	2.4 V ≤ AV_{REFP} ≤ 5.5 V			±0.25	%FSR
Full-scale error**Notes 1, 3**	E_{FS}	10-bit resolution AV_{REFP} = V_{DD}**Note 2**	2.4 V ≤ AV_{REFP} ≤ 5.5 V			±0.25	%FSR
Integral linearity error**Note 1**	ILE	10-bit resolution AV_{REFP} = V_{DD}**Note 2**	2.4 V ≤ AV_{REFP} ≤ 5.5 V			±2.5	LSB
Differential linearity error**Note 1**	DLE	10-bit resolution AV_{REFP} = V_{DD}**Note 2**	2.4 V ≤ AV_{REFP} ≤ 5.5 V			±1.5	LSB
Analog input voltage	V_{AIN}	ANI2 to ANI7		0		AV_{REFP}	V
		Internal reference voltage		V_{BGR}**Note 4**			V
		Temperature sensor output voltage		V_{TMPS25}**Note 4**			V

Note 1. This value does not include the quantization error (±1/2 LSB).

Note 2. When AV_{REFP} < V_{DD}, the maximum values are as follows.
- Overall error: Add ±1.0 LSB to the maximum value when V_{DD} = AV_{REFP}.
- Zero-scale/full-scale error: Add ±0.05%FSR to the maximum value when V_{DD} = AV_{REFP}.
- Integral linearity error and differential linearity error: Add ±0.5 LSB to the maximum value when V_{DD} = AV_{REFP}.

Note 3. This value is indicated as a ratio (%FSR) to the full-scale value.

Note 4. See **28.6.3 Characteristics of the temperature sensor and internal reference voltage**.

(2) Reference voltage (+) = AVREFP/ANI0 (ADREFP1 = 0, ADREFP0 = 1), reference voltage (-) = AVREFM/ANI1 (ADREFM = 1), conversion target: ANI16 to ANI19

(TA = –40 to +105°C, 2.4 V ≤ AVREFP ≤ VDD ≤ 5.5 V, VSS = 0 V, reference voltage (+) = AVREFP, reference voltage (–) = AVREFM = 0 V)

Item	Symbol	Conditions		Min.	Typ.	Max.	Unit
Resolution	RES			8		10	bit
Overall error**Note 1**	AINL	10-bit resolution AVREFP = VDD**Note 2**	2.4 V ≤ AVREFP ≤ 5.5 V		1.2	±5.0	LSB
Conversion time	tCONV	10-bit resolution conversion target: ANI16 to ANI19	3.6 V ≤ VDD ≤ 5.5 V	2.125		39	μs
			2.7 V ≤ VDD ≤ 5.5 V	3.1875		39	μs
			2.4 V ≤ VDD ≤ 5.5 V	17		39	μs
Zero-scale error**Notes 1, 3**	EZS	10-bit resolution AVREFP = VDD**Note 2**	2.4 V ≤ AVREFP ≤ 5.5 V			±0.35	%FSR
Full-scale error**Notes 1, 3**	EFS	10-bit resolution AVREFP = VDD**Note 2**	2.4 V ≤ AVREFP ≤ 5.5 V			±0.35	%FSR
Integral linearity error**Note 1**	ILE	10-bit resolution AVREFP = VDD**Note 2**	2.4 V ≤ AVREFP ≤ 5.5 V			±3.5	LSB
Differential linearity error**Note 1**	DLE	10-bit resolution AVREFP = VDD**Note 2**	2.4 V ≤ AVREFP ≤ 5.5 V			±2.0	LSB
Analog input voltage	VAIN	ANI16 to ANI19		0		AVREFP	V

Note 1. This value does not include the quantization error (±1/2 LSB).

Note 2. When AVREFP < VDD, the maximum values are as follows.

- Overall error: Add ±4.0 LSB to the maximum value when VDD = AVREFP.
- Zero-scale/full-scale error: Add ±0.20%FSR to the maximum value when VDD = AVREFP.
- Integral linearity error/differential linearity error: Add ±2.0 LSB to the maximum value when VDD = AVREFP.

Note 3. This value is indicated as a ratio (%FSR) to the full-scale value.

(3) Reference voltage (+) = V_{DD} (ADREFP1 = 0, ADREFP0 = 0), reference voltage (-) = Vss (ADREFM = 0), conversion target: ANI0 to ANI7, ANI16 to ANI19, internal reference voltage, temperature sensor output voltage

(T_A = –40 to +105°C, 2.4 V ≤ V_{DD} ≤ 5.5 V, V_{SS} = 0 V, reference voltage (+) = V_{DD}, reference voltage (–) = V_{SS})

Item	Symbol	Conditions		Min.	Typ.	Max.	Unit
Resolution	RES			8		10	bit
Overall error**Note 1**	AINL	10-bit resolution	2.4 V ≤ V_{DD} ≤ 5.5 V		1.2	±7.0	LSB
Conversion time	t_{CONV}	10-bit resolution conversion target: ANI0 to ANI7, ANI16 to ANI19	3.6 V ≤ V_{DD} ≤ 5.5 V	2.125		39	μs
			2.7 V ≤ V_{DD} ≤ 5.5 V	3.1875		39	μs
			2.4 V ≤ V_{DD} ≤ 5.5 V	17		39	μs
		10-bit resolution conversion target: Internal reference voltage, temperature sensor output voltage	3.6 V ≤ V_{DD} ≤ 5.5 V	2.375		39	μs
			2.7 V ≤ V_{DD} ≤ 5.5 V	3.5625		39	μs
			2.4 V ≤ V_{DD} ≤ 5.5 V	17		39	μs
Zero-scale error**Notes 1, 2**	E_{ZS}	10-bit resolution	2.4 V ≤ V_{DD} ≤ 5.5 V			±0.60	%FSR
Full-scale error**Notes 1, 2**	E_{FS}	10-bit resolution	2.4 V ≤ V_{DD} ≤ 5.5 V			±0.60	%FSR
Integral linearity error**Note 1**	ILE	10-bit resolution	2.4 V ≤ V_{DD} ≤ 5.5 V			±4.0	LSB
Differential linearity error**Note 1**	DLE	10-bit resolution	2.4 V ≤ V_{DD} ≤ 5.5 V			±2.0	LSB
Analog input voltage	V_{AIN}	ANI0 to ANI7, ANI16 to ANI19		0		V_{DD}	V
		Internal reference voltage		V_{BGR}**Note 3**			V
		Temperature sensor output voltage		V_{TMPS25}**Note 3**			V

Note 1. This value does not include the quantization error (±1/2 LSB).

Note 2. This value is indicated as a ratio (%FSR) to the full-scale value.

Note 3. See **28.6.3 Characteristics of the temperature sensor and internal reference voltage**.

(4) Reference voltage (+) = Internal reference voltage (ADREFP1 = 0, ADREFP0 = 1), reference voltage (-) = AVREFM/ANI1 (ADREFM = 1), conversion target: ANI0, ANI2 to ANI7, ANI16 to ANI19

(TA = –40 to +105°C, 2.4 V ≤ VDD ≤ 5.5 V, VSS = 0 V, reference voltage (+) = VBGR Note 1, reference voltage (–) = AVREFM Note 2 = 0 V)

Item	Symbol	Conditions	Min.	Typ.	Max.	Unit
Resolution	RES		8			bit
Conversion time	tCONV		17		39	µs
Zero-scale error Notes 3, 4	EZS				±0.60	%FSR
Integral linearity error Note 3	ILE				±2.0	LSB
Differential linearity error Note 3	DLE				±1.0	LSB
Analog input voltage	VAIN		0		VBGR Note 1	V

Note 1. See **28.6.3 Characteristics of the temperature sensor and internal reference voltage**.

Note 2. When reference voltage (–) = VSS, the maximum values are as follows.

- Zero-scale error: Add ±0.35%FSR to the maximum value when reference voltage (–) = AVREFM.
- Integral linearity error: Add ±0.5 LSB to the maximum value when reference voltage (–) = AVREFM.
- Differential linearity error: Add ±0.2 LSB to the maximum value when reference voltage (–) = AVREFM.

Note 3. This value does not include the quantization error (±1/2 LSB).

Note 4. This value is indicated as a ratio (%FSR) to the full-scale value.

28.6.3 Characteristics of the temperature sensor and internal reference voltage

(T_A = -40 to +105°C, 1.8 V ≤ V_{DD} ≤ 5.5 V, V_{SS} = 0 V)

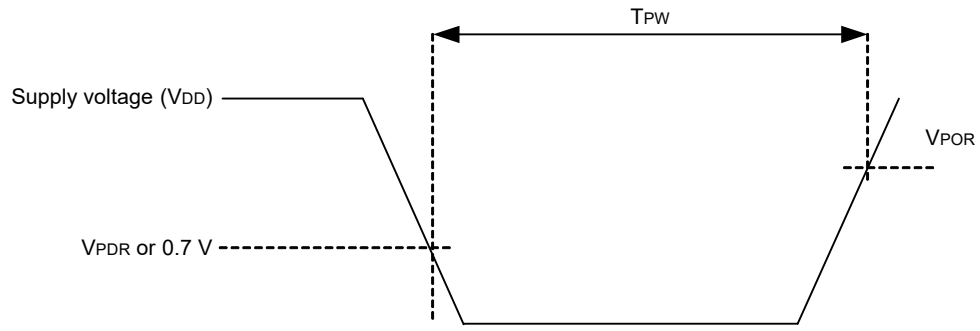
Item	Symbol	Conditions	Min.	Typ.	Max.	Unit
Temperature sensor output voltage	V_{TMPS25}	Setting ADS register = 80H, T_A = +25°C		1.05		V
Internal reference voltage	V_{BGR}	Setting ADS register = 81H	1.42	1.48	1.54	V
Temperature coefficient	F_{VTMPS}	Temperature dependency of the temperature sensor voltage		-3.3		mV/°C
Operation stabilization wait time	t_{AMP}		5			μs

28.6.4 Characteristics of the POR circuit

(T_A = -40 to +105°C, V_{SS} = 0 V)

Item	Symbol	Conditions	Min.	Typ.	Max.	Unit
Detection voltage	V_{POR}, V_{PDR}		1.43	1.50	1.57	V
Minimum pulse width**Note**	T_{PW}		300			μs

Note This width is the minimum time required for a POR reset when V_{DD} falls below V_{PDR}. This width is also the minimum time required for a POR reset from when V_{DD} falls below 0.7 V to when V_{DD} exceeds V_{POR} in the STOP mode or while the main system clock is stopped through setting bit 0 (HIOSTOP) and bit 7 (MSTOP) in the clock operation status control register (CSC).



28.6.5 Characteristics of the LVD circuit

1. LVD0 Detection Voltage in the Reset Mode and Interrupt Mode

(T_A = -40 to +105°C, $V_{PDR} \leq V_{DD} \leq 5.5$ V, V_{SS} = 0 V)

Item		Symbol	Conditions	Min.	Typ.	Max.	Unit
Detection voltage	Supply voltage level	V_{LVD00}	The power supply voltage is rising.	3.84	3.96	4.08	V
			The power supply voltage is falling.	3.76	3.88	4.00	V
		V_{LVD01}	The power supply voltage is rising.	2.88	2.97	3.06	V
			The power supply voltage is falling.	2.82	2.91	3.00	V
		V_{LVD02}	The power supply voltage is rising.	2.59	2.67	2.75	V
			The power supply voltage is falling.	2.54	2.62	2.70	V
		V_{LVD03}	The power supply voltage is rising.	2.31	2.38	2.45	V
			The power supply voltage is falling.	2.26	2.33	2.40	V
		V_{LVD04}	The power supply voltage is rising.	1.84	1.90	1.95	V
			The power supply voltage is falling.	1.80	1.86	1.91	V
		V_{LVD05}	The power supply voltage is rising.	1.64	1.69	1.74	V
			The power supply voltage is falling.	1.60	1.65	1.70	V
Minimum pulse width		t_{LW}		500			μs
Detection delay time						500	μs

2. LVD1 Detection Voltage in the Reset Mode and Interrupt Mode

(T_A = -40 to +105°C, $V_{PDR} \leq V_{DD} \leq 5.5$ V, V_{SS} = 0 V)

Item		Symbol	Conditions	Min.	Typ.	Max.	Unit
Detection voltage	Supply voltage level	V_{LVD10}	The power supply voltage is rising.	4.08	4.16	4.24	V
			The power supply voltage is falling.	4.00	4.08	4.16	V
		V_{LVD11}	The power supply voltage is rising.	3.88	3.96	4.04	V
			The power supply voltage is falling.	3.80	3.88	3.96	V
		V_{LVD12}	The power supply voltage is rising.	3.68	3.75	3.82	V
			The power supply voltage is falling.	3.60	3.67	3.74	V
		V_{LVD13}	The power supply voltage is rising.	3.48	3.55	3.62	V
			The power supply voltage is falling.	3.40	3.47	3.54	V
		V_{LVD14}	The power supply voltage is rising.	3.28	3.35	3.42	V
			The power supply voltage is falling.	3.20	3.27	3.34	V
		V_{LVD15}	The power supply voltage is rising.	3.07	3.13	3.19	V
			The power supply voltage is falling.	3.00	3.06	3.12	V
		V_{LVD16}	The power supply voltage is rising.	2.91	2.97	3.03	V
			The power supply voltage is falling.	2.85	2.91	2.97	V
		V_{LVD17}	The power supply voltage is rising.	2.76	2.82	2.87	V
			The power supply voltage is falling.	2.70	2.76	2.81	V
		V_{LVD18}	The power supply voltage is rising.	2.61	2.66	2.71	V
			The power supply voltage is falling.	2.55	2.60	2.65	V
		V_{LVD19}	The power supply voltage is rising.	2.45	2.50	2.55	V
			The power supply voltage is falling.	2.40	2.45	2.50	V
		V_{LVD110}	The power supply voltage is rising.	2.35	2.40	2.45	V
			The power supply voltage is falling.	2.30	2.35	2.40	V
		V_{LVD111}	The power supply voltage is rising.	2.25	2.30	2.34	V
			The power supply voltage is falling.	2.20	2.25	2.29	V
		V_{LVD112}	The power supply voltage is rising.	2.15	2.20	2.24	V
			The power supply voltage is falling.	2.10	2.15	2.19	V
		V_{LVD113}	The power supply voltage is rising.	2.05	2.09	2.13	V
			The power supply voltage is falling.	2.00	2.04	2.08	V
		V_{LVD114}	The power supply voltage is rising.	1.94	1.98	2.02	V
			The power supply voltage is falling.	1.90	1.94	1.98	V
		V_{LVD115} **Note**	The power supply voltage is rising.	1.84	1.88	1.91	V
			The power supply voltage is falling.	1.80	1.84	1.87	V
		V_{LVD116} **Note**	The power supply voltage is rising.	1.74	1.78	1.81	V
			The power supply voltage is falling.	1.70	1.74	1.77	V
		V_{LVD117} **Note**	The power supply voltage is rising.	1.64	1.67	1.70	V
			The power supply voltage is falling.	1.60	1.63	1.66	V
Minimum pulse width		t_{LW}		500			μs
Detection delay time						500	μs

Note This setting can only be used when LVD0 is disabled.

28.6.6 Characteristics of the rising slope of the power supply voltage

(T_A = -40 to +105°C, V_{SS} = 0 V)

Item	Symbol	Conditions	Min.	Typ.	Max.	Unit
Power supply voltage rising slope	S_{VDD}				54	V/ms

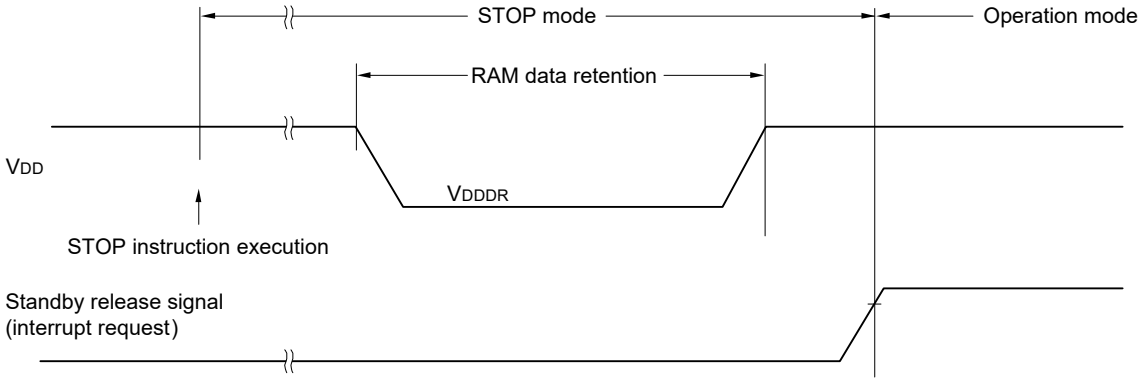
Caution Make sure to keep the internal reset state by LVD0 or an external reset until V_{DD} reaches the operating voltage range shown in AC characteristics.

28.7 Characteristics of Retention of RAM Data

(T_A = -40 to +105°C, V_{SS} = 0V)

Item	Symbol	Conditions	Min.	Typ.	Max.	Unit
Data retention supply voltage	V_{DDDR}		1.43[Note]		5.5	V

Note This voltage depends on the POR detection voltage. When the voltage drops, the data in RAM are retained until a POR is applied, but are not retained following a POR.



28.8 Characteristics of Flash Memory Programming

(T_A = -40 to +105°C, 1.8 V ≤ V_{DD} ≤ 5.5 V, V_{SS} = 0 V)

Item	Symbol	Conditions	Min.	Typ.	Max.	Unit
CPU/peripheral hardware clock frequency	f_{CLK}		1		32	MHz
Number of code flash memory rewrites[Notes 1, 2, 3]	C_{erwr}	Retained for 10 years T_A = +85°C	10,000			Times
		Retained for 20 years T_A = +85°C	1,000			
Number of data flash memory rewrites[Notes 1, 2, 3]		Retained for 1 year T_A = +25°C		1,000,000		
		Retained for 5 years T_A = +85°C	100,000			
		Retained for 20 years T_A = +85°C	10,000			

Note 1. 1 erase + 1 write after the erase is regarded as 1 rewrite. The retaining years are until next rewrite after the rewrite.

Note 2. The listed numbers of times apply when using flash memory programmer and Renesas Electronics self programming library.

Note 3. These are the characteristics of the flash memory and the results obtained from reliability testing by Renesas Electronics Corporation.

1. Code flash memory

(T_A = -40 to +105°C, 1.8 V ≤ V_{DD} ≤ 5.5 V, V_{SS} = 0 V)

Item		Symbol	f_{CLK} = 1 MHz			f_{CLK} = 2 MHz, 3 MHz			4 MHz ≤ f_{CLK} < 8 MHz			8 MHz ≤ f_{CLK} < 32 MHz			f_{CLK} = 32 MHz			Unit
			Min.	Typ.	Max.	Min.	Typ.	Max.	Min.	Typ.	Max.	Min.	Typ.	Max.	Min.	Typ.	Max.	
Programming time	4 bytes	t_{P4}	—	74.7	656.5	—	51.0	464.6	—	41.7	384.8	—	37.1	346.2	—	34.2	321.9	µs
Erasure time	2 Kbytes	t_{E2K}	—	10.4	312.2	—	7.7	258.5	—	6.4	231.8	—	5.8	218.4	—	5.6	214.4	ms
Blank checking time	4 bytes	t_{BC4}	—	—	38.4	—	—	19.2	—	—	13.1	—	—	10.2	—	—	8.3	µs
	2 Kbytes	t_{BC2K}	—	—	2618.9	—	—	1309.5	—	—	658.3	—	—	332.8	—	—	234.1	µs
Time taken to forcibly stop the erasure		t_{SED}	—	—	18.0	—	—	14.0	—	—	12.0	—	—	11.0	—	—	10.3	µs
Security setting time		t_{AWSSAS}	—	18.2	526.2	—	14.4	469.2	—	12.5	441.1	—	11.6	427.1	—	11.3	422.6	ms
Time until programming starts following cancellation of the STOP instruction		—	20	—	—	20	—	—	20	—	—	20	—	—	20	—	—	µs

Caution The listed values do not include the time until the operations of the flash memory start following execution of an instruction by software.

2. Data flash memory

(T_A = -40 to +105°C, 1.8 V ≤ V_{DD} ≤ 5.5 V, V_{SS} = 0 V)

Item		Symbol	f_{CLK} = 1 MHz			f_{CLK} = 2 MHz, 3 MHz			4 MHz ≤ f_{CLK} < 8 MHz			8 MHz ≤ f_{CLK} < 32 MHz			f_{CLK} = 32 MHz			Unit
			Min.	Typ.	Max.	Min.	Typ.	Max.	Min.	Typ.	Max.	Min.	Typ.	Max.	Min.	Typ.	Max.	
Programming time	1 byte	t_{P4}	—	74.7	656.5	—	51.0	464.6	—	41.7	384.8	—	37.1	346.2	—	34.2	321.9	µs
Erasure time	256 bytes	t_{E2K}	—	7.8	259.2	—	6.4	232.0	—	5.8	218.5	—	5.5	211.8	—	5.4	209.7	ms
Blank checking time	1 byte	t_{BC4}	—	—	38.4	—	—	19.2	—	—	13.1	—	—	10.2	—	—	8.3	µs
	256 bytes	t_{BC2K}	—	—	1326.1	—	—	663.1	—	—	335.1	—	—	171.2	—	—	121.0	µs
Time taken to forcibly stop the erasure		t_{SED}	—	—	18.0	—	—	14.0	—	—	12.0	—	—	11.0	—	—	10.3	µs
Time until programming starts following cancellation of the STOP instruction		—	20	—	—	20	—	—	20	—	—	20	—	—	20	—	—	µs
Time until reading starts following setting DFLCTL.DFLEN to 1		—	0.25	—	—	0.25	—	—	0.25	—	—	0.25	—	—	0.25	—	—	µs

Caution The listed values do not include the time until the operations of the flash memory start following execution of an instruction by software.

28.9 Dedicated Flash Memory Programmer Communication (UART)

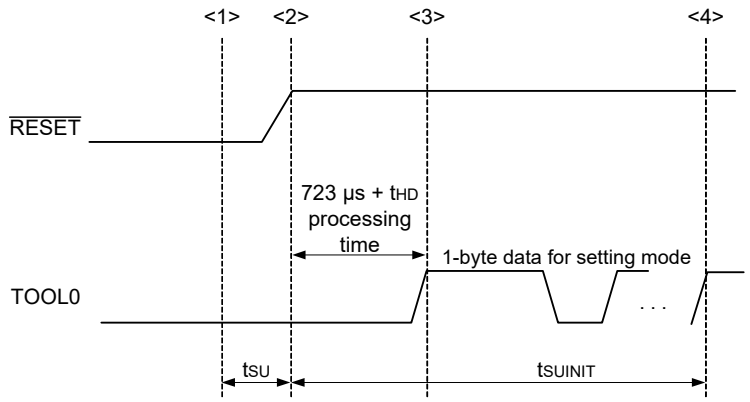
(T_A = -40 to +105°C, 1.8 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V)

Item	Symbol	Conditions	Min.	Typ.	Max.	Unit
Transfer rate		During serial programming	115,200		1,000,000	bps

28.10 Timing of Entry to Flash Memory Programming Modes

(T_A = -40 to +105°C, 1.8 V ≤ EV_{DD0} ≤ V_{DD} ≤ 5.5 V, V_{SS} = EV_{SS0} = 0 V)

Item	Symbol	Conditions	Min.	Typ.	Max.	Unit
Time to complete the communication for the initial setting after the external reset is released	t_{SUINIT}	POR and LVD reset must be released before the external reset is released.			100	ms
Time to release the external reset after the TOOL0 pin is set to the low level	t_{SU}	POR and LVD reset must be released before the external reset is released.	10			μs
Time to hold the TOOL0 pin at the low level after the external reset is released (the processing time of the firmware to control the flash memory is not included)	t_{HD}	POR and LVD reset must be released before the external reset is released.	1			ms



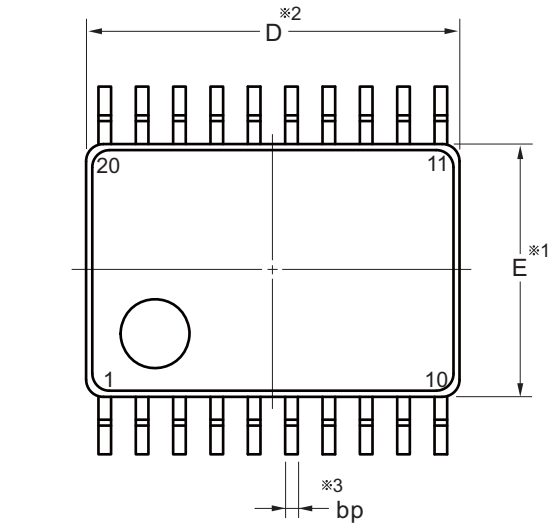
<1> The low level is input to the TOOL0 pin.
<2> The external reset is released. Note that the POR and LVD reset must be released before the external reset is released.
<3> The TOOL0 pin is set to the high level.
<4> Setting of the flash memory programming mode by UART reception and complete the baud rate setting.

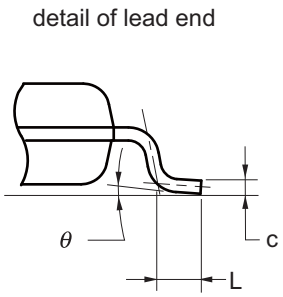
Remark t_{SUINIT}: The time during which the communications for the initial setting must be completed within 100 ms after the external reset is released.
t_{SU}: Time to release the external reset after the TOOL0 pin is set to the low level
t_{HD}: Time to hold the TOOL0 pin at the low level after the external reset is released. It does not include the processing time of the firmware to control the flash memory.

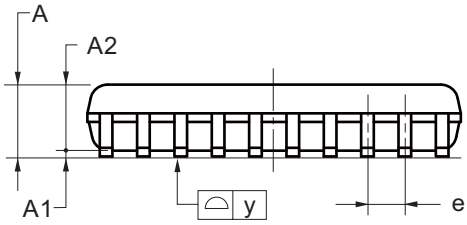
Section 29 Package Drawings

29.1 20-pin Products

JEITA Package Code	RENESAS Code	Previous Code	MASS (TYP.) [g]
P-LSSOP20-4.4x6.5-0.65	PLSP0020JB-A	P20MA-65-NAA-1	0.1









NOTE

1.Dimensions "※1" and "※2" do not include mold flash.

2.Dimension "※3" does not include trim offset.

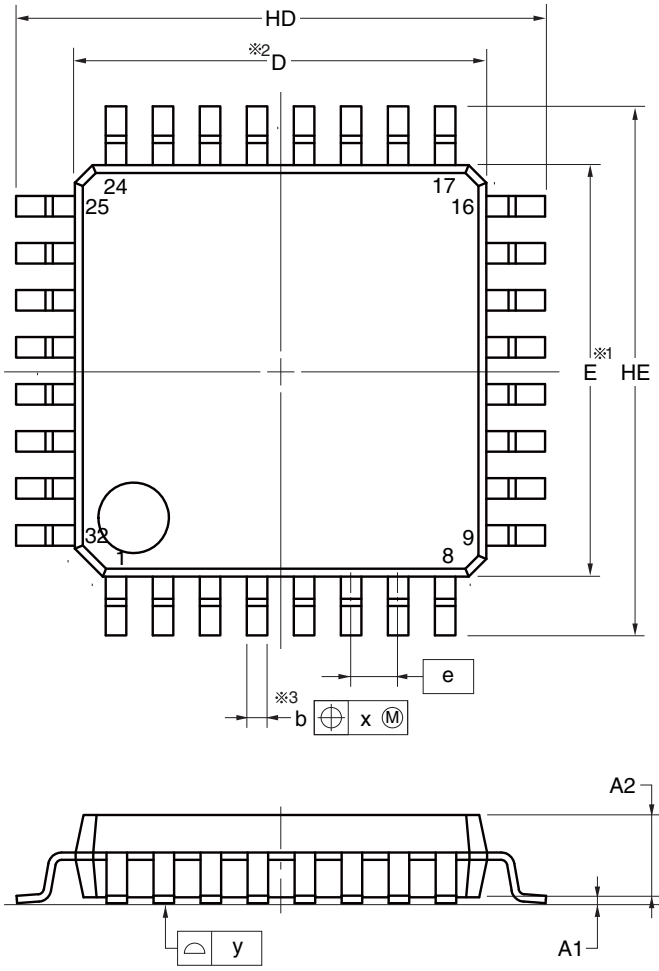
(UNIT:mm)

ITEM	DIMENSIONS
D	6.50±0.10
E	4.40±0.10
HE	6.40±0.20
A	1.45 MAX.
A1	0.10±0.10
A2	1.15
e	0.65±0.12
bp	$0.22^{+0.10}_{-0.05}$
c	$0.15^{+0.05}_{-0.02}$
L	0.50±0.20
y	0.10
θ	0° to 10°

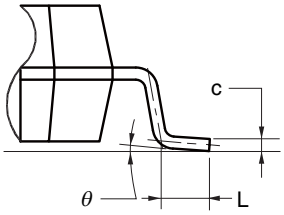
©2012 Renesas Electronics Corporation. All rights reserved.

29.2 32-pin Products

JEITA Package Code	RENESAS Code	Previous Code	MASS (TYP.) [g]
P-LQFP32-7x7-0.80	PLQP0032GB-A	P32GA-80-GBT-1	0.2



detail of lead end



(UNIT:mm)

ITEM	DIMENSIONS
D	7.00±0.10
E	7.00±0.10
HD	9.00±0.20
HE	9.00±0.20
A	1.70 MAX.
A1	0.10±0.10
A2	1.40
b	0.37±0.05
c	0.145±0.055
L	0.50±0.20
θ	0° to 8°
e	0.80
x	0.20
y	0.10

NOTE

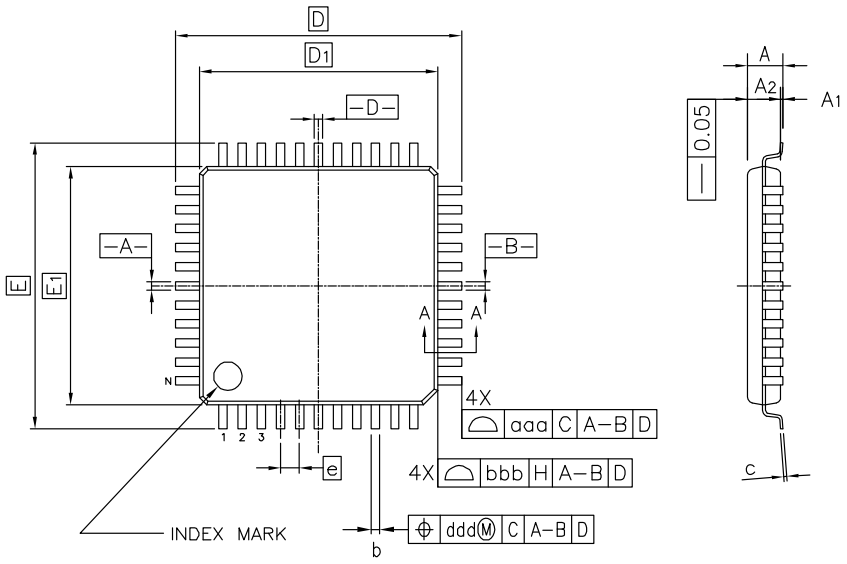
1.Dimensions "※1" and "※2" do not include mold flash.

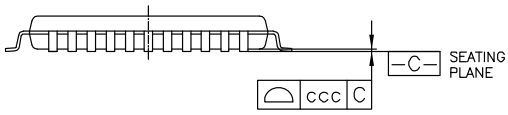
2.Dimension "※3" does not include trim offset.

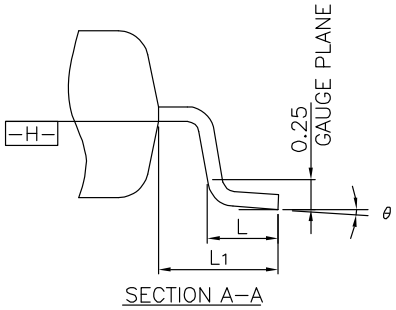
© 2012 Renesas Electronics Corporation. All rights reserved.

29.3 44-pin Products

JEITA Package code	RENESAS code	MASS(TYP.)[g]
P-LQFP44-10x10-0.80	PLQP0044GF-A	0.3



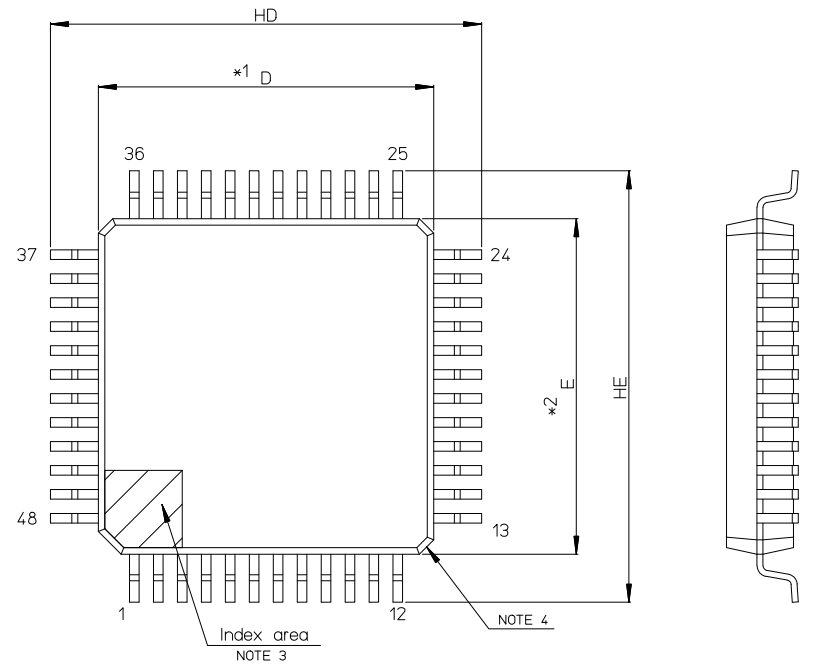


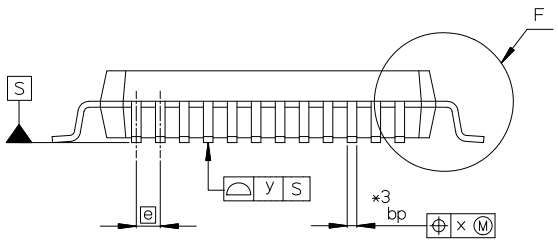


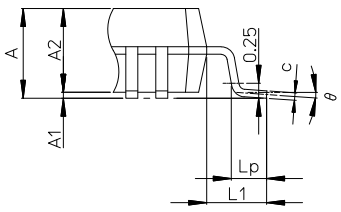
Reference Symbol	Dimension in Millimeters		
	Min.	Nom.	Max.
A	—	—	1.70
A_1	0.05	—	0.15
A_2	1.35	1.40	1.45
D	12.00 BSC.		
D_1	10.00 BSC.		
E	12.00 BSC.		
E_1	10.00 BSC.		
N	—	44	—
e	0.80 BSC.		
b	0.30	0.37	0.45
c	0.09	—	0.20
θ	0°	3.5°	8°
L	0.45	0.60	0.75
L_1	1.00 REF.		
aaa	—	—	0.20
bbb	—	—	0.20
ccc	—	—	0.10
ddd	—	—	0.20

29.4 48-pin Products

JEITA Package Code	RENESAS Code	Previous Code	MASS[Typ.]
P-LFQFP48-7x7-0.50	PLQP0048KB-B	——	0.2g







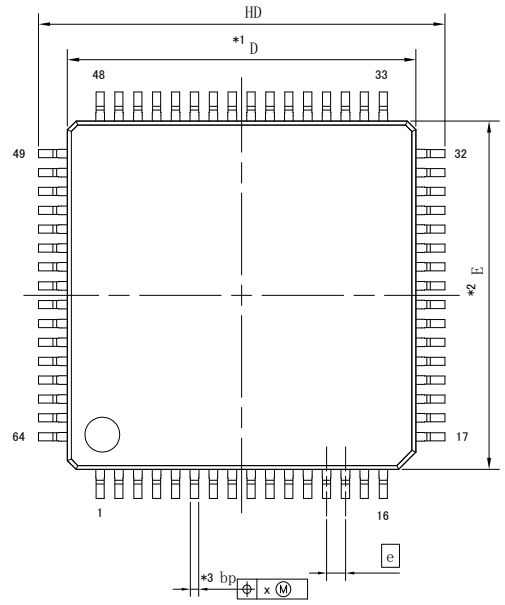
Detail F

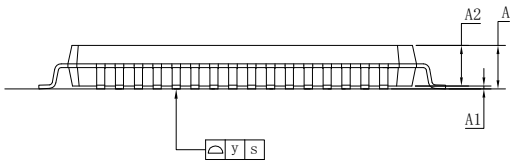
NOTE)
1. DIMENSIONS "*1" AND "*2" DO NOT INCLUDE MOLD FLASH.
2. DIMENSION "*3" DOES NOT INCLUDE TRIM OFFSET.
3. PIN 1 VISUAL INDEX FEATURE MAY VARY, BUT MUST BE LOCATED WITHIN THE HATCHED AREA.
4. CHAMFERS AT CORNERS ARE OPTIONAL; SIZE MAY VARY.

Reference Symbol	Dimension in Millimeters		
	Min	Nom	Max
D	6.9	7.0	7.1
E	6.9	7.0	7.1
A2	——	1.4	——
HD	8.8	9.0	9.2
HE	8.8	9.0	9.2
A	——	——	1.7
A1	0.05	——	0.15
bp	0.17	0.20	0.27
c	0.09	——	0.20
θ	0°	3.5°	8°
e	——	0.5	——
x	——	——	0.08
y	——	——	0.08
Lp	0.45	0.6	0.75
L1	——	1.0	——

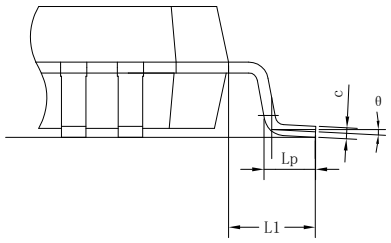
29.5 64-pin Products

JEITA Package code	RENESAS code	MASS(TYP.)[g]
P-LQFP64-12x12-0.65	PLQP0064JB-A	0.50





detail of lead end

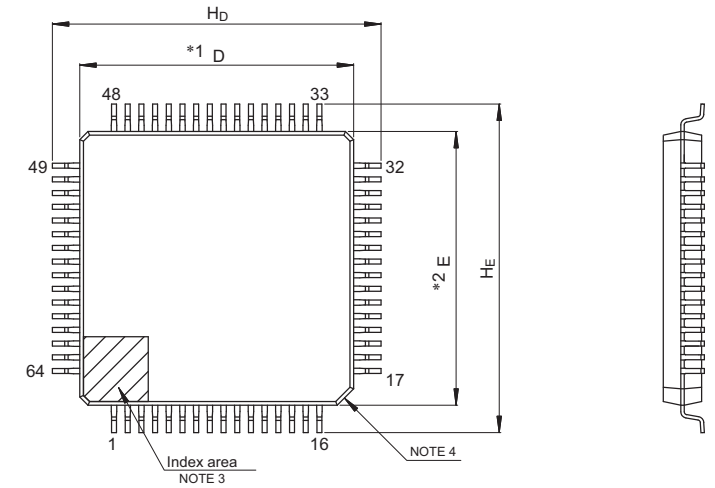


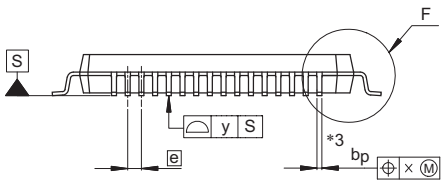
NOTE
1.DIMENSIONS "*1" AND "*2"DO NOT INCLUDE MOLD FLASH.
2.DIMENSION "*3" DOES NOT INCLUDE TRIM OFFSET.

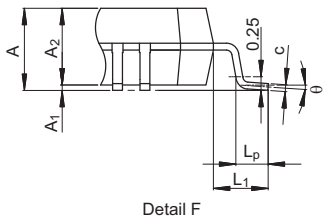
Reference Symbol	Dimension in Millimeters		
	Min.	Nom.	Max.
E	11.90	12.00	12.10
D	11.90	12.00	12.10
A_2	—	1.40	—
H_D	13.80	14.00	14.20
H_E	13.80	14.00	14.20
A	—	—	1.70
A_1	0.05	—	0.15
Lp	0.45	0.60	0.75
L1	—	1.00	—
b_p	0.27	0.32	0.37
c	0.09	—	0.20
e	—	0.65	—
θ	0.00	3.50	8.00
x	—	—	0.08
y	—	—	0.08

JEITA Package Code	RENESAS Code	Previous Code	MASS (Typ) [g]
P-LFQFP64-10x10-0.50	PLQP0064KB-C	—	0.3

Unit: mm







Detail F

NOTE)
1. DIMENSIONS "*1" AND "*2" DO NOT INCLUDE MOLD FLASH.
2. DIMENSION "*3" DOES NOT INCLUDE TRIM OFFSET.
3. PIN 1 VISUAL INDEX FEATURE MAY VARY, BUT MUST BE LOCATED WITHIN THE HATCHED AREA.
4. CHAMFERS AT CORNERS ARE OPTIONAL, SIZE MAY VARY.

Reference Symbol	Dimensions in millimeters		
	Min	Nom	Max
D	9.9	10.0	10.1
E	9.9	10.0	10.1
A_2	—	1.4	—
H_D	11.8	12.0	12.2
H_E	11.8	12.0	12.2
A	—	—	1.7
A_1	0.05	—	0.15
b_p	0.15	0.20	0.27
c	0.09	—	0.20
θ	0°	3.5°	8°
e	—	0.5	—
x	—	—	0.08
y	—	—	0.08
L_p	0.45	0.6	0.75
L_1	—	1.0	—

© 2015 Renesas Electronics Corporation. All rights reserved.

Appendix A Revision History

A.1 Major Revisions in This Edition

(1/2)

Page	Description	Classification
Section 6 Clock Generator		
p.159	Figure 6 - 3 Format of System Clock Control Register (CKC): Note 2 was modified.	(c)
p.168	Figure 6 - 9 Format of Subsystem Clock Supply Mode Control Register (OSMC): What used to be Note 4 was deleted.	(c)
Section 7 Timer Array Unit (TAU)		
p.189	Section 7 Timer Array Unit (TAU): The description was modified.	(c)
p.281	Figure 7 - 70 Procedure for Operations When the One-shot Pulse Output Function Is to Be Used (2/2) was modified.	(a)
Section 10 A/D Converter (ADC)		
p.343	Figure 10 - 4 Format of A/D Converter Mode Register 0 (ADM0): Note 2 and Caution 2 were modified.	(c)
p.345	Figure 10 - 5 Timing of Operations When the A/D Voltage Comparator Is Used: Caution 4 was modified.	(c)
p.348	Table 10 - 3 A/D Conversion Time Selection (3/4) was modified.	(c)
p.349	Table 10 - 3 A/D Conversion Time Selection (4/4) was modified.	(c)
p.363	10.5 Input Voltage and Conversion Results: The description was modified.	(a)
p.379	Figure 10 - 33 Settings When Temperature Sensor Output Voltage or Internal Reference Voltage is Selected was modified.	(c)
p.380	Figure 10 - 34 Settings in Test Mode was modified.	(a)
p.384	Figure 10 - 38 Flow of Setting up the SNOOZE Mode was modified.	(a)
p.390	10.10 Points for Caution When the A/D Converter Is to be Used: 6. Input impedance of analog input (ANIxx) pins was modified.	(a)
Section 11 Serial Array Unit (SAU)		
p.425	11.3.16 Input switch control register (ISC): The description was modified.	(a)
p.519	11.6.3 SNOOZE mode: The description was modified.	(a)
p.559	11.8.5 Calculating transfer rate: Caution was modified.	(a)
Section 12 Serial Interface IICA (IICA)		
p.590	12.5.1 Start conditions: The description was modified.	(a)
Section 16 Standby Function		
p.739	16.3.3 SNOOZE mode: 1. SNOOZE mode setting and operating states was modified.	(b)
Section 18 Power-on-reset Circuit (POR)		
p.757	Figure 18 - 2 Timing of Generation of Internal Reset Signal by Power-on-reset Circuit and Voltage Detector (2/2): Notes 3 and 4 were modified.	(b)

Remark “Classification” in the above table classifies revisions as follows.
(a): Error correction, (b): Addition/change of specifications, (c): Addition/change of description or note,
(d): Addition/change of package, part number, or management division, (e): Addition/change of related documents

(2/2)

Page	Description	Classification
Section 24 Flash Memory		
p.842	Figure 24 - 17 Format of Flash Memory Sequencer Control Register (FSSQ): Notes 2 and 3 were modified.	(a)
Section 28 Electrical Characteristics		
p.915	28.2.2 Characteristics of the on-chip oscillators was modified.	(b)
p.922	28.3.1 Characteristics of pins was modified.	(b)
p.923 to p.926	28.3.2 Characteristics of the supply current: Tables and Note 4 of Table (4/4) were modified.	(b)
p.927	28.3.2 Characteristics of the supply current: Peripheral Functions (Common to all products) was modified.	(b)

Remark “Classification” in the above table classifies revisions as follows.
(a): Error correction, (b): Addition/change of specifications, (c): Addition/change of description or note,
(d): Addition/change of package, part number, or management division, (e): Addition/change of related documents

A.2 Major Revisions in This Edition

Here is the revision history of the preceding editions. Section indicates the section of each edition.

Page	Description	Classification
Rev.0.50	First Edition issued.	Throughout

RL78/G21 User’s Manual: Hardware

Publication Date: Rev.0.50 Nov 10, 2025
Rev.1.00 May 20, 2026

Published by: Renesas Electronics Corporation

RL78/G21

Renesas Electronics Corporation

R01UH1177EJ0100