User's Manual

16

# RL78/I1B

## 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.2.10 Apr 2016

# Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
3. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The recommended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.

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

   “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.

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

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

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

(2012.4)

NOTES FOR CMOS DEVICES

(1) 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 VIL (MAX) and VIH (MIN) due to noise, etc., 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 VIL (MAX) and VIH (MIN).

(2) HANDLING OF UNUSED INPUT PINS: Unconnected CMOS device inputs can be cause of malfunction. If an input pin is unconnected, it is possible that an internal input level may be generated due to noise, etc., causing malfunction. CMOS devices behave differently than Bipolar or NMOS devices. Input levels of CMOS devices must be fixed high or low by using pull-up or pull-down circuitry. Each unused pin should be connected to VDD or GND via a resistor if there is a possibility that it will be an output pin. All handling related to unused pins must be judged separately for each device and according to related specifications governing the device.

(3) PRECAUTION AGAINST ESD: A strong electric field, when exposed to a MOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop generation of static electricity as much as possible, and quickly dissipate it when it has occurred. Environmental control must be adequate. When it is dry, a humidifier should be used. It is recommended to avoid using insulators that 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 should be grounded. The operator should be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions need to be taken for PW boards with mounted semiconductor devices.

(4) STATUS BEFORE INITIALIZATION: Power-on does not necessarily define the initial status of a MOS device. Immediately after the power source is turned ON, devices with reset functions have not yet been initialized. Hence, power-on does not guarantee output pin levels, I/O settings or contents of registers. A device is not initialized until the reset signal is received. A reset operation must be executed immediately after power-on for devices with reset functions.

(5) POWER ON/OFF SEQUENCE: In the case of a device that uses different power supplies for the internal operation and external interface, as a rule, switch on the external power supply after switching on the internal power supply. When switching the power supply off, as a rule, switch off the external power supply and then the internal power supply. Use of the reverse power on/off sequences may result in the application of an overvoltage to the internal elements of the device, causing malfunction and degradation of internal elements due to the passage of an abnormal current. The correct power on/off sequence must be judged separately for each device and according to related specifications governing the device.

(6) INPUT OF SIGNAL DURING POWER OFF STATE : Do not input signals or an I/O pull-up power supply while the device is not powered. 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. Input of signals during the power off state must be judged separately for each device and according to related specifications governing the device.

# How to Use This Manual

**Readers**

This manual is intended for user engineers who wish to understand the functions of the RL78/I1B and design and develop application systems and programs for these devices.
The target products are as follows.

- 80-pin: R5F10MME, R5F10MMG
- 100-pin: R5F10MPE, R5F10MPG

**Purpose**

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

**Organization**

The RL78/I1B manual is separated into two parts: this manual and the software edition (common to the RL78 Family).

| **RL78/I1B User's Manual Hardware** | **RL78 Microcontroller User's Manual Software** |
|---|---|
| • Pin functions<br>• Internal block functions<br>• Interrupts<br>• Other on-chip peripheral functions<br>• Electrical specifications | • CPU functions<br>• Instruction set<br>• 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/I1B Microcontroller instructions:
  → Refer to the separate document **RL78 Family Software User's Manual (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 | ···×××× or ××××B |
| | | Decimal | ···×××× |
| | | Hexadecimal | ···××××H |

**Related Documents** The related documents indicated in this publication may include preliminary versions. However, preliminary versions are not marked as such.

**Documents Related to Devices**

| Document Name | Document No. |
|---|---|
| RL78/F1E User's Manual: Hardware | R01UH0611E |
| RL78 Family User's Manual: Software | R01US0015E |

**Documents Related to Flash Memory Programming**

| Document Name | | Document No. |
|---|---|---|
| PG-FP5 Flash Memory Programmer User's Manual | | — |
| | RL78, 78K, V850, RX100, RX200, RX600 (Except RX64x), R8C, SH | R20UT2923E |
| | Common | R20UT2922E |
| | Setup Manual | R20UT0930E |

**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 MPUs & MCUs RL78 Family | R01CP0003E |
| Semiconductor Package Mount Manual | **Note** |
| Semiconductor Reliability Handbook | R51ZZ0001E |

**Note** See the “Semiconductor Package Mount Manual” website (http://www.renesas.com/products/package/index.jsp).

**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**


**CHAPTER 1 OUTLINE ..... 1**

**1.1 Features ..... 1**
**1.2 List of Part Numbers ..... 4**
**1.3 Pin Configuration (Top View) ..... 5**
1.3.1 80-pin products ..... 5
1.3.2 100-pin products ..... 6
**1.4 Pin Identification ..... 7**
**1.5 Block Diagram ..... 8**
1.5.1 80-pin products ..... 8
1.5.2 100-pin products ..... 9
**1.6 Outline of Functions ..... 10**

**CHAPTER 2 PIN FUNCTIONS ..... 12**

**2.1 Port Function List ..... 12**
2.1.1 80-pin products ..... 13
2.1.2 100-pin products ..... 15
**2.2 Functions Other than Port Pins ..... 18**
2.2.1 With functions for each product ..... 18
2.2.2 Description of Functions ..... 20
**2.3 Connection of Unused Pins ..... 22**
**2.4 Block Diagrams of Pins ..... 24**

**CHAPTER 3 CPU ARCHITECTURE ..... 38**

**3.1 Memory Space ..... 38**
3.1.1 Internal program memory space ..... 43
3.1.2 Mirror area ..... 46
3.1.3 Internal data memory space ..... 48
3.1.4 Special function register (SFR) area ..... 48
3.1.5 Extended special function register (2nd SFR: 2nd Special Function Register) area ..... 48
3.1.6 Data memory addressing ..... 49
**3.2 Processor Registers ..... 50**
3.2.1 Control registers ..... 50
3.2.2 General-purpose registers ..... 52
3.2.3 ES and CS registers ..... 53
3.2.4 Special function registers (SFRs) ..... 54
3.2.5 Extended special function registers (2nd SFRs: 2nd Special Function Registers) ..... 60
**3.3 Instruction Address Addressing ..... 69**

3.3.1 Relative addressing ..... 69
3.3.2 Immediate addressing ..... 69
3.3.3 Table indirect addressing ..... 70
3.3.4 Register direct addressing ..... 70
**3.4 Addressing for Processing Data Addresses ..... 71**
3.4.1 Implied addressing ..... 71
3.4.2 Register addressing ..... 71
3.4.3 Direct addressing ..... 72
3.4.4 Short direct addressing ..... 73
3.4.5 SFR addressing ..... 74
3.4.6 Register indirect addressing ..... 75
3.4.7 Based addressing ..... 76
3.4.8 Based indexed addressing ..... 80
3.4.9 Stack addressing ..... 81

**CHAPTER 4 PORT FUNCTIONS ..... 85**

**4.1 Port Functions ..... 85**
**4.2 Port Configuration ..... 85**
4.2.1 Port 0 ..... 86
4.2.2 Port 1 ..... 86
4.2.3 Port 2 ..... 87
4.2.4 Port 3 ..... 88
4.2.5 Port 4 ..... 88
4.2.6 Port 5 ..... 88
4.2.7 Port 6 ..... 88
4.2.8 Port 7 ..... 88
4.2.9 Port 8 ..... 89
4.2.10 Port 12 ..... 89
4.2.11 Port 13 ..... 89
**4.3 Registers Controlling Port Function ..... 90**
4.3.1 Port mode registers (PMxx) ..... 94
4.3.2 Port registers (Pxx) ..... 95
4.3.3 Pull-up resistor option registers (PUxx) ..... 96
4.3.4 Port input mode registers (PIMxx) ..... 97
4.3.5 Port output mode registers (POMxx) ..... 98
4.3.6 A/D port configuration register (ADPC) ..... 99
4.3.7 Global digital input disable register (GDIDIS) ..... 100
4.3.8 Peripheral I/O redirection register (PIOR) ..... 101
4.3.9 LCD port function registers 0 to 5 (PFSEG0 to PFSEG5) ..... 102
4.3.10 LCD input switch control register (ISCLCD) ..... 104
**4.4 Port Function Operations ..... 105**

4.4.1 Writing to I/O port ..... 105
4.4.2 Reading from I/O port ..... 105
4.4.3 Operations on I/O port ..... 105
4.4.4 Connecting to external device with different potential (1.8 V, 2.5 V, 3 V) ..... 106
4.4.5 Handling different potential (1.8 V, 2.5 V, 3 V) by using I/O buffers ..... 106
**4.5 Register Settings When Using Alternate Function ..... 108**
4.5.1 Basic concept when using alternate function ..... 108
4.5.2 Register settings for alternate function whose output function is not used ..... 109
4.5.3 Register setting examples for used port and alternate functions ..... 110
4.5.4 Operation of Ports That Alternately Function as SEGxx Pins ..... 118
4.5.5 Operation of Ports That Alternately Function as VL3, CAPL, CAPH Pins ..... 119
**4.6 Cautions When Using Port Function ..... 121**
4.6.1 Cautions on 1-Bit Manipulation Instruction for Port Register n (Pn) ..... 121
4.6.2 Notes on specifying the pin settings ..... 122

**CHAPTER 5 CLOCK GENERATOR ..... 123**

**5.1 Functions of Clock Generator ..... 123**
**5.2 Configuration of Clock Generator ..... 125**
**5.3 Registers Controlling Clock Generator ..... 127**
5.3.1 Clock operation mode control register (CMC) ..... 127
5.3.2 System clock control register (CKC) ..... 130
5.3.3 Clock operation status control register (CSC) ..... 132
5.3.4 Oscillation stabilization time counter status register (OSTC) ..... 133
5.3.5 Oscillation stabilization time select register (OSTS) ..... 135
5.3.6 Peripheral enable registers 0 and 1 (PER0, PER1) ..... 137
5.3.7 Subsystem clock supply mode control register (OSMC) ..... 140
5.3.8 High-speed on-chip oscillator frequency select register (HOCODIV) ..... 142
5.3.9 Peripheral clock control register (PCKC) ..... 143
**5.4 System Clock Oscillator ..... 144**
5.4.1 X1 oscillator ..... 144
5.4.2 XT1 oscillator ..... 144
5.4.3 High-speed on-chip oscillator ..... 148
5.4.4 Low-speed on-chip oscillator ..... 148
**5.5 Clock Generator Operation ..... 149**
**5.6 Controlling the Clock ..... 151**
5.6.1 Example of setting high-speed on-chip oscillator ..... 151
5.6.2 Example of setting X1 oscillation clock ..... 152
5.6.3 Example of setting XT1 oscillation clock ..... 154
5.6.4 CPU clock status transition diagram ..... 155
5.6.5 Conditions before changing the CPU clock and processing after changing CPU clock ..... 161
5.6.6 Time required for switching CPU clock and system clock ..... 163

5.6.7 Conditions before stopping clock oscillation ..... 164
**5.7 Resonator and Oscillator Constants ..... 165**

**CHAPTER 6 HIGH-SPEED ON-CHIP OSCILLATOR CLOCK FREQUENCY CORRECTION FUNCTION ..... 169**

**6.1 High-speed On-chip Oscillator Clock Frequency Correction Function ..... 169**
**6.2 Register ..... 170**
6.2.1 High-speed on-chip oscillator clock frequency correction control register (HOCOFC) ..... 170
**6.3 Operation ..... 171**
6.3.1 Operation overview ..... 171
6.3.2 Operation procedure ..... 174
**6.4 Usage Notes ..... 175**
6.4.1 SFR access ..... 175
6.4.2 Operation during standby state ..... 175
6.4.3 Changing high-speed on-chip oscillator frequency select register (HOCODIV) ..... 175

**CHAPTER 7 TIMER ARRAY UNIT ..... 176**

**7.1 Functions of Timer Array Unit ..... 177**
7.1.1 Independent channel operation function ..... 177
7.1.2 Simultaneous channel operation function ..... 178
7.1.3 8-bit timer operation function (channels 1 and 3 only) ..... 179
7.1.4 LIN-bus supporting function (channel 7 only) ..... 180
**7.2 Configuration of Timer Array Unit ..... 181**
7.2.1 Timer count register mn (TCRmn) ..... 185
7.2.2 Timer data register mn (TDRmn) ..... 187
**7.3 Registers Controlling Timer Array Unit ..... 188**
7.3.1 Peripheral enable register 0 (PER0) ..... 189
7.3.2 Timer clock select register m (TPSm) ..... 190
7.3.3 Timer mode register mn (TMRmn) ..... 193
7.3.4 Timer status register mn (TSRmn) ..... 198
7.3.5 Timer channel enable status register m (TEm) ..... 199
7.3.6 Timer channel start register m (TSm) ..... 200
7.3.7 Timer channel stop register m (TTm) ..... 201
7.3.8 Timer input select register 0 (TIS0) ..... 202
7.3.9 Timer output enable register m (TOEm) ..... 203
7.3.10 Timer output register m (TOm) ..... 204
7.3.11 Timer output level register m (TOLm) ..... 205
7.3.12 Timer output mode register m (TOMm) ..... 206
7.3.13 Input switch control register (ISC) ..... 207
7.3.14 Noise filter enable register 1 (NFEN1) ..... 208
7.3.15 Registers controlling port functions of pins to be used for timer I/O ..... 210

**7.4 Basic Rules of Timer Array Unit ..... 211**
7.4.1 Basic rules of simultaneous channel operation function ..... 211
7.4.2 Basic rules of 8-bit timer operation function (channels 1 and 3 only) ..... 213
**7.5 Operation of Counter ..... 214**
7.5.1 Count clock (fTCLK) ..... 214
7.5.2 Start timing of counter ..... 216
7.5.3 Operation of counter ..... 217
**7.6 Channel Output (TOmn Pin) Control ..... 222**
7.6.1 TOmn pin output circuit configuration ..... 222
7.6.2 TOmn Pin Output Setting ..... 223
7.6.3 Cautions on Channel Output Operation ..... 224
7.6.4 Collective manipulation of TOmn bit ..... 229
7.6.5 Timer Interrupt and TOmn Pin Output at Operation Start ..... 230
**7.7 Timer Input (TImn) Control ..... 231**
7.7.1 TImn input circuit configuration ..... 231
7.7.2 Noise filter ..... 231
7.7.3 Cautions on channel input operation ..... 232
**7.8 Independent Channel Operation Function of Timer Array Unit ..... 233**
7.8.1 Operation as interval timer/square wave output ..... 233
7.8.2 Operation as external event counter ..... 239
7.8.3 Operation as input pulse interval measurement ..... 244
7.8.4 Operation as input signal high-/low-level width measurement ..... 248
7.8.5 Operation as delay counter ..... 252
**7.9 Simultaneous Channel Operation Function of Timer Array Unit ..... 257**
7.9.1 Operation as one-shot pulse output function ..... 257
7.9.2 Operation as PWM function ..... 264
7.9.3 Operation as multiple PWM output function ..... 271
**7.10 Cautions When Using Timer Array Unit ..... 279**
7.10.1 Cautions When Using Timer output ..... 279

**CHAPTER 8 REAL-TIME CLOCK 2 ..... 280**

**8.1 Functions of Real-time Clock 2 ..... 280**
**8.2 Configuration of Real-time Clock 2 ..... 281**
**8.3 Registers Controlling Real-time Clock 2 ..... 283**
8.3.1 Peripheral enable register 0 (PER0) ..... 284
8.3.2 Peripheral enable register 1 (PER1) ..... 285
8.3.3 Subsystem clock supply mode control register (OSMC) ..... 286
8.3.4 Power-on-reset status register (PORSR) ..... 287
8.3.5 Real-time clock control register 0 (RTCC0) ..... 288
8.3.6 Real-time clock control register 1 (RTCC1) ..... 290
8.3.7 Second count register (SEC) ..... 293

8.3.8 Minute count register (MIN) ... 293
8.3.9 Hour count register (HOUR) ... 294
8.3.10 Date count register (DAY) ... 296
8.3.11 Day-of-week count register (WEEK) ... 297
8.3.12 Month count register (MONTH) ... 298
8.3.13 Year count register (YEAR) ... 298
8.3.14 Clock error correction register (SUBCUD) ... 299
8.3.15 Alarm minute register (ALARMWM) ... 302
8.3.16 Alarm hour register (ALARMWH) ... 302
8.3.17 Alarm day-of-week register (ALARMWW) ... 303
**8.4 Real-time Clock 2 Operation ... 304**
8.4.1 Starting operation of real-time clock 2 ... 304
8.4.2 Shifting to HALT/STOP mode after starting operation ... 305
8.4.3 Reading real-time clock 2 counter ... 306
8.4.4 Writing to real-time clock 2 counter ... 307
8.4.5 Setting alarm of real-time clock 2 ... 308
8.4.6 1 Hz output of real-time clock 2 ... 309
8.4.7 Clock error correction register setting procedure ... 310
8.4.8 Example of watch error correction of real-time clock 2 ... 311
8.4.9 High-accuracy 1 Hz output ... 314

**CHAPTER 9 SUBSYSTEM CLOCK FREQUENCY MEASUREMENT CIRCUIT ... 315**

**9.1 Subsystem Clock Frequency Measurement Circuit ... 315**
**9.2 Configuration of Subsystem Clock Frequency Measurement Circuit ... 315**
**9.3 Registers Controlling Subsystem Clock Frequency Measurement Circuit ... 316**
9.3.1 Peripheral enable register 1 (PER1) ... 317
9.3.2 Subsystem clock supply mode control register (OSMC) ... 318
9.3.3 Frequency measurement count register L (FMCRL) ... 319
9.3.4 Frequency measurement count register H (FMCRH) ... 319
9.3.5 Frequency measurement control register (FMCTL) ... 320
**9.4 Subsystem Clock Frequency Measurement Circuit Operation ... 321**
9.4.1 Setting subsystem clock frequency measurement circuit ... 321
9.4.2 Subsystem clock frequency measurement circuit operation timing ... 322

**CHAPTER 10 12-BIT INTERVAL TIMER ... 323**

**10.1 Functions of 12-bit Interval Timer ... 323**
**10.2 Configuration of 12-bit Interval Timer ... 323**
**10.3 Registers Controlling 12-bit Interval Timer ... 324**
10.3.1 Peripheral enable register 1 (PER1) ... 324
10.3.2 Subsystem clock supply mode control register (OSMC) ... 325
10.3.3 12-bit interval timer control register (ITMC) ... 326

**10.4 12-bit Interval Timer Operation ..... 327**
10.4.1 12-bit interval timer operation timing ..... 327
10.4.2 Start of count operation and re-enter to HALT/STOP mode after returned from HALT/STOP mode ..... 328

**CHAPTER 11 8-BIT INTERVAL TIMER ..... 329**

**11.1 Overview ..... 329**
**11.2 I/O Pins ..... 330**
**11.3 Registers ..... 330**
11.3.1 8-bit interval timer counter register ni (TRTni) (n = 0 or 1, i = 0 or 1) ..... 331
11.3.2 8-bit interval timer counter register n (TRTn) (n = 0 or 1) ..... 331
11.3.3 8-bit interval timer compare register ni (TRTCMPni) (n = 0 or 1, i = 0 or 1) ..... 332
11.3.4 8-bit interval timer compare register n (TRTCMPn) (n = 0 or 1) ..... 332
11.3.5 8-bit interval timer control register n (TRTCRn) (n = 0 or 1) ..... 333
11.3.6 8-bit interval timer division register n (TRTMDn) (n = 0 or 1) ..... 334
**11.4 Operation ..... 335**
11.4.1 Counter mode ..... 335
11.4.2 Timer operation ..... 336
11.4.3 Count start/stop timing ..... 338
11.4.4 Timing of updating compare register values ..... 341
**11.5 Notes on 8-bit Interval Timer ..... 342**
11.5.1 Changing the operating mode and clock settings ..... 342
11.5.2 Accessing compare registers ..... 342
11.5.3 8-bit interval timer setting procedure ..... 342

**CHAPTER 12 CLOCK OUTPUT/BUZZER OUTPUT CONTROLLER ..... 343**

**12.1 Functions of Clock Output/Buzzer Output Controller ..... 343**
**12.2 Configuration of Clock Output/Buzzer Output Controller ..... 345**
**12.3 Registers Controlling Clock Output/Buzzer Output Controller ..... 345**
12.3.1 Clock output select registers n (CKSn) ..... 345
12.3.2 Registers controlling port functions of pins to be used for clock or buzzer output ..... 347
**12.4 Operations of Clock Output/Buzzer Output Controller ..... 348**
12.4.1 Operation as output pin ..... 348
**12.5 Cautions of Clock Output/Buzzer Output Controller ..... 348**

**CHAPTER 13 WATCHDOG TIMER ..... 349**

**13.1 Functions of Watchdog Timer ..... 349**
**13.2 Configuration of Watchdog Timer ..... 350**
**13.3 Register Controlling Watchdog Timer ..... 351**
13.3.1 Watchdog timer enable register (WDTE) ..... 351

**13.4 Operation of Watchdog Timer ..... 352**
13.4.1 Controlling operation of watchdog timer ..... 352
13.4.2 Setting overflow time of watchdog timer ..... 353
13.4.3 Setting window open period of watchdog timer ..... 354
13.4.4 Setting watchdog timer interval interrupt ..... 355

**CHAPTER 14 A/D CONVERTER ..... 356**

**14.1 Function of A/D Converter ..... 356**
**14.2 Configuration of A/D Converter ..... 359**
**14.3 Registers Controlling A/D Converter ..... 361**
14.3.1 Peripheral enable register 0 (PER0) ..... 362
14.3.2 A/D converter mode register 0 (ADM0) ..... 363
14.3.3 A/D converter mode register 1 (ADM1) ..... 371
14.3.4 A/D converter mode register 2 (ADM2) ..... 372
14.3.5 10-bit A/D conversion result register (ADCR) ..... 375
14.3.6 8-bit A/D conversion result register (ADCRH) ..... 375
14.3.7 Analog input channel specification register (ADS) ..... 376
14.3.8 Conversion result comparison upper limit setting register (ADUL) ..... 377
14.3.9 Conversion result comparison lower limit setting register (ADLL) ..... 377
14.3.10 A/D test register (ADTES) ..... 378
14.3.11 Registers controlling port function of analog input pins ..... 378
**14.4 A/D Converter Conversion Operations ..... 379**
**14.5 Input Voltage and Conversion Results ..... 381**
**14.6 A/D Converter Operation Modes ..... 382**
14.6.1 Software trigger mode (select mode, sequential conversion mode) ..... 382
14.6.2 Software trigger mode (select mode, one-shot conversion mode) ..... 383
14.6.3 Software trigger mode (scan mode, sequential conversion mode) ..... 384
14.6.4 Software trigger mode (scan mode, one-shot conversion mode) ..... 385
14.6.5 Hardware trigger no-wait mode (select mode, sequential conversion mode) ..... 386
14.6.6 Hardware trigger no-wait mode (select mode, one-shot conversion mode) ..... 387
14.6.7 Hardware trigger no-wait mode (scan mode, sequential conversion mode) ..... 388
14.6.8 Hardware trigger no-wait mode (scan mode, one-shot conversion mode) ..... 389
14.6.9 Hardware trigger wait mode (select mode, sequential conversion mode) ..... 390
14.6.10 Hardware trigger wait mode (select mode, one-shot conversion mode) ..... 391
14.6.11 Hardware trigger wait mode (scan mode, sequential conversion mode) ..... 392
14.6.12 Hardware trigger wait mode (scan mode, one-shot conversion mode) ..... 393
**14.7 A/D Converter Setup Flowchart ..... 394**
14.7.1 Setting up software trigger mode ..... 394
14.7.2 Setting up hardware trigger no-wait mode ..... 395
14.7.3 Setting up hardware trigger wait mode ..... 396

14.7.4 Setup when temperature sensor output voltage/internal reference voltage is selected (example for software trigger mode and one-shot conversion mode) ... 397
14.7.5 Setting up test mode ... 398
**14.8 SNOOZE Mode Function ... 399**
**14.9 How to Read A/D Converter Characteristics Table ... 403**
**14.10 Cautions for A/D Converter ... 405**

**CHAPTER 15 TEMPERATURE SENSOR 2 ... 409**

**15.1 Functions of Temperature Sensor ... 409**
**15.2 Registers ... 410**
15.2.1 Temperature sensor control test register (TMPCTL) ... 410
**15.3 Setting Procedures ... 411**
15.3.1 A/D converter mode register 0 (ADM0) ... 411
15.3.2 Switching modes ... 412

**CHAPTER 16 24-BIT ΔΣ A/D CONVERTER ... 413**

**16.1 Functions of 24-bit ΔΣ A/D Converter ... 413**
16.1.1 I/O pins ... 416
16.1.2 Pre-amplifier ... 416
16.1.3 ΔΣ A/D converter ... 416
16.1.4 Reference voltage generator ... 416
16.1.5 Phase adjustment circuits (PHC0, PHC1) ... 417
16.1.6 Digital filter (DF) ... 417
16.1.7 High-pass filter (HPF) ... 417
**16.2 Registers ... 418**
16.2.1 ΔΣ A/D converter mode register (DSADMR) ... 419
16.2.2 ΔΣ A/D converter gain control register 0 (DSADGCR0) ... 421
16.2.3 ΔΣ A/D converter gain control register 1 (DSADGCR1) ... 422
16.2.4 ΔΣ A/D converter HPF control register (DSADHPFCR) ... 423
16.2.5 ΔΣ A/D converter phase control register 0 (DSADPHCR0) ... 424
16.2.6 ΔΣ A/D converter phase control register 1 (DSADPHCR1) ... 425
16.2.7 ΔΣ A/D converter conversion result register n (DSADCRnL, DSADCRnM, DSADCRnH) (n = 0, 1, 2, 3) ... 426
16.2.8 ΔΣ A/D converter conversion result register n (DSADCRn) (n = 0, 1, 2, 3) ... 428
16.2.9 Peripheral enable register 1 (PER1) ... 429
16.2.10 Peripheral clock control register (PCKC) ... 430
**16.3 Operation ... 431**
16.3.1 Operation of 24-bit ΔΣ A/D converter ... 432
16.3.2 Procedure for switching from normal operation mode to neutral missing mode ... 434
16.3.3 Interrupt operation ... 435
16.3.4 Operation in standby state ... 435

**16.4 Notes on Using 24-Bit ΔΣ A/D Converter ..... 436**
16.4.1 External pins ..... 436
16.4.2 SFR access ..... 436
16.4.3 Setting operating clock ..... 436
16.4.4 Phase adjustment for single-phase two-wire ..... 437

**CHAPTER 17 COMPARATOR ..... 438**

**17.1 Functions of Comparator ..... 438**
**17.2 Configuration of Comparator ..... 439**
**17.3 Registers Controlling Comparator ..... 440**
17.3.1 Peripheral enable register 1 (PER1) ..... 440
17.3.2 Comparator mode setting register (COMPMDR) ..... 441
17.3.3 Comparator filter control register (COMPFIR) ..... 443
17.3.4 Comparator output control register (COMPOCR) ..... 444
17.3.5 Registers controlling port functions of analog input pins ..... 445
**17.4 Operation ..... 446**
17.4.1 Comparator i digital filter (i = 0 or 1) ..... 448
17.4.2 Comparator i (i = 0 or 1) interrupts ..... 448
17.4.3 Comparator i Output (i = 0 or 1) ..... 449
17.4.4 Stopping or supplying comparator clock ..... 449

**CHAPTER 18 SERIAL ARRAY UNIT ..... 450**

**18.1 Functions of Serial Array Unit ..... 451**
18.1.1 3-wire serial I/O (CSI00) ..... 451
18.1.2 UART (UART0 to UART2) ..... 452
18.1.3 Simplified $I^2C$ (IIC00, IIC10) ..... 453
18.1.4 IrDA ..... 453
**18.2 Configuration of Serial Array Unit ..... 454**
18.2.1 Shift register ..... 458
18.2.2 Lower 8/9 bits of the serial data register mn (SDRmn) ..... 458
**18.3 Registers Controlling Serial Array Unit ..... 460**
18.3.1 Peripheral enable register 0 (PER0) ..... 461
18.3.2 Serial clock select register m (SPSm) ..... 462
18.3.3 Serial mode register mn (SMRmn) ..... 463
18.3.4 Serial communication operation setting register mn (SCRmn) ..... 464
18.3.5 Serial data register mn (SDRmn) ..... 467
18.3.6 Serial flag clear trigger register mn (SIRmn) ..... 469
18.3.7 Serial status register mn (SSRmn) ..... 470
18.3.8 Serial channel start register m (SSm) ..... 472
18.3.9 Serial channel stop register m (STm) ..... 473
18.3.10 Serial channel enable status register m (SEm) ..... 474

18.3.11 Serial output enable register m (SOEm) ... 475
18.3.12 Serial output register m (SOm) ... 476
18.3.13 Serial output level register m (SOLm) ... 477
18.3.14 Serial standby control register 0 (SSC0) ... 479
18.3.15 Input switch control register (ISC) ... 480
18.3.16 Noise filter enable register 0 (NFEN0) ... 481
18.3.17 Registers controlling port functions of serial input/output pins ... 482
**18.4 Operation Stop Mode ... 483**
18.4.1 Stopping the operation by units ... 484
18.4.2 Stopping the operation by channels ... 485
**18.5 Operation of 3-Wire Serial I/O (CSI00) Communication ... 486**
18.5.1 Master transmission ... 488
18.5.2 Master reception ... 498
18.5.3 Master transmission/reception ... 507
18.5.4 Slave transmission ... 517
18.5.5 Slave reception ... 527
18.5.6 Slave transmission/reception ... 534
18.5.7 SNOOZE mode function ... 544
18.5.8 Calculating transfer clock frequency ... 548
18.5.9 Procedure for processing errors that occurred during 3-wire serial I/O (CSI00) communication ... 550
**18.6 Operation of UART (UART0 to UART2) Communication ... 551**
18.6.1 UART transmission ... 553
18.6.2 UART reception ... 563
18.6.3 SNOOZE mode function ... 570
18.6.4 Calculating baud rate ... 578
18.6.5 Procedure for processing errors that occurred during UART (UART0 to UART2) communication ... 582
**18.7 LIN Communication Operation ... 583**
18.7.1 LIN transmission ... 583
18.7.2 LIN reception ... 586
**18.8 Operation of Simplified $I^2C$ (IIC00, IIC10) Communication ... 591**
18.8.1 Address field transmission ... 593
18.8.2 Data transmission ... 599
18.8.3 Data reception ... 603
18.8.4 Stop condition generation ... 608
18.8.5 Calculating transfer rate ... 609
18.8.6 Procedure for processing errors that occurred during simplified $I^2C$ (IIC00, IIC10) communication ... 611

**CHAPTER 19 SERIAL INTERFACE IICA ... 612**

**19.1 Functions of Serial Interface IICA ..... 612**
**19.2 Configuration of Serial Interface IICA ..... 615**
**19.3 Registers Controlling Serial Interface IICA ..... 618**
19.3.1 Peripheral enable register 0 (PER0) ..... 618
19.3.2 IICA control register n0 (IICCTLn0) ..... 619
19.3.3 IICA status register n (IICSn) ..... 624
19.3.4 IICA flag register n (IICFn) ..... 626
19.3.5 IICA control register n1 (IICCTLn1) ..... 628
19.3.6 IICA low-level width setting register n (IICWLn) ..... 630
19.3.7 IICA high-level width setting register n (IICWHn) ..... 630
19.3.8 Port mode register 6 (PM6) ..... 631
**19.4 $I^2C$ Bus Mode Functions ..... 632**
19.4.1 Pin configuration ..... 632
19.4.2 Setting transfer clock by using IICWLn and IICWHn registers ..... 633
**19.5 $I^2C$ Bus Definitions and Control Methods ..... 635**
19.5.1 Start conditions ..... 635
19.5.2 Addresses ..... 636
19.5.3 Transfer direction specification ..... 636
19.5.4 Acknowledge (ACK) ..... 637
19.5.5 Stop condition ..... 638
19.5.6 Wait ..... 639
19.5.7 Canceling wait ..... 641
19.5.8 Interrupt request (INTIICAn) generation timing and wait control ..... 642
19.5.9 Address match detection method ..... 643
19.5.10 Error detection ..... 643
19.5.11 Extension code ..... 643
19.5.12 Arbitration ..... 644
19.5.13 Wakeup function ..... 646
19.5.14 Communication reservation ..... 649
19.5.15 Cautions ..... 653
19.5.16 Communication operations ..... 654
19.5.17 Timing of $I^2C$ interrupt request (INTIICAn) occurrence ..... 661
**19.6 Timing Charts ..... 682**

**CHAPTER 20 IrDA ..... 697**

**20.1 Functions of IrDA ..... 697**
**20.2 Registers ..... 698**
20.2.1 Peripheral enable register 0 (PER0) ..... 698
20.2.2 IrDA control register (IRCR) ..... 699
**20.3 Operation ..... 700**
20.3.1 IrDA communication operation procedure ..... 700

20.3.2 Transmission ..... 701
20.3.3 Reception ..... 702
20.3.4 Selecting High-Level Pulse Width ..... 702
**20.4 Usage Notes on IrDA ..... 703**

**CHAPTER 21 LCD CONTROLLER/DRIVER ..... 704**

**21.1 Functions of LCD Controller/Driver ..... 705**
**21.2 Configuration of LCD Controller/Driver ..... 707**
**21.3 Registers Controlling LCD Controller/Driver ..... 709**
21.3.1 LCD mode register 0 (LCDM0) ..... 710
21.3.2 LCD mode register 1 (LCDM1) ..... 712
21.3.3 Subsystem clock supply mode control register (OSMC) ..... 714
21.3.4 LCD clock control register 0 (LCDC0) ..... 715
21.3.5 LCD boost level control register (VLCD) ..... 716
21.3.6 LCD input switch control register (ISCLCD) ..... 717
21.3.7 LCD port function registers 0 to 5 (PFSEG0 to PFSEG5) ..... 719
21.3.8 Port mode registers 0, 1, 3, 5, 7, 8 (PM0, PM1, PM3, PM5, PM7, PM8) ..... 722
**21.4 LCD Display Data Registers ..... 723**
**21.5 Selection of LCD Display Register ..... 727**
21.5.1 A-pattern area and B-pattern area data display ..... 728
21.5.2 Blinking display (Alternately displaying A-pattern and B-pattern area data) ..... 728
**21.6** Setting the LCD Controller/Driver **..... 729**
**21.7 Operation Stop Procedure ..... 732**
**21.8 Supplying LCD Drive Voltages VL1, VL2, VL3, and VL4 ..... 733**
21.8.1 External resistance division method ..... 733
21.8.2 Internal voltage boosting method ..... 735
21.8.3 Capacitor split method ..... 736
**21.9 Common and Segment Signals ..... 737**
21.9.1 Normal liquid crystal waveform ..... 737
**21.10 Display Modes ..... 745**
21.10.1 Static display example ..... 745
21.10.2 Two-time-slice display example ..... 748
21.10.3 Three-time-slice display example ..... 751
21.10.4 Four-time-slice display example ..... 755
21.10.5 Six-time-slice display example ..... 759
21.10.6 Eight-time-slice display example ..... 762

**CHAPTER 22 DATA TRANSFER CONTROLLER (DTC) ..... 766**

**22.1 Functions of DTC ..... 767**
**22.2 Configuration of DTC ..... 768**
**22.3 Registers Controlling DTC ..... 769**

22.3.1 Allocation of DTC control data area and DTC vector table area ... 770
22.3.2 Control data allocation ... 771
22.3.3 Vector table ... 773
22.3.4 Peripheral enable register 1 (PER1) ... 775
22.3.5 DTC control register j (DTCCRj) (j = 0 to 23) ... 776
22.3.6 DTC block size register j (DTBLSj) (j = 0 to 23) ... 777
22.3.7 DTC transfer count register j (DTCCTj) (j = 0 to 23) ... 777
22.3.8 DTC transfer count reload register j (DTRLDj) (j = 0 to 23) ... 778
22.3.9 DTC source address register j (DTSARj) (j = 0 to 23) ... 778
22.3.10 DTC destination address register j (DTDARj) (j = 0 to 23) ... 778
22.3.11 DTC activation enable register i (DTCENi) (i = 0 to 3) ... 779
22.3.12 DTC base address register (DTCBAR) ... 781
**22.4 DTC Operation ... 782**
22.4.1 Activation sources ... 782
22.4.2 Normal mode ... 783
22.4.3 Repeat mode ... 786
22.4.4 Chain transfers ... 789
**22.5 Notes on DTC ... 791**
22.5.1 Setting DTC control data and vector table ... 791
22.5.2 Allocation of DTC control data area and DTC vector table area ... 791
22.5.3 DTC pending instruction ... 791
22.5.4 Number of DTC execution clock cycles ... 792
22.5.5 DTC response time ... 793
22.5.6 DTC activation sources ... 793
22.5.7 Operation in standby mode status ... 794

**CHAPTER 23 INTERRUPT FUNCTIONS ... 795**

**23.1 Interrupt Function Types ... 795**
**23.2 Interrupt Sources and Configuration ... 795**
**23.3 Registers Controlling Interrupt Functions ... 800**
23.3.1 Interrupt request flag registers (IF0L, IF0H, IF1L, IF1H, IF2L, IF2H, IF3L) ... 803
23.3.2 Interrupt mask flag registers (MK0L, MK0H, MK1L, MK1H, MK2L, MK2H, MK3L) ... 805
23.3.3 Priority specification flag registers (PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR03L, PR10L, PR10H, PR11L, PR11H, PR12L, PR12H, PR13L) ... 806
23.3.4 External interrupt rising edge enable register (EGP0), external interrupt falling edge enable register (EGN0) ... 808
23.3.5 Program status word (PSW) ... 809
**23.4 Interrupt Servicing Operations ... 810**
23.4.1 Maskable interrupt request acknowledgment ... 810
23.4.2 Software interrupt request acknowledgment ... 813
23.4.3 Multiple interrupt servicing ... 813

23.4.4 Interrupt servicing during division instruction ..... 817
23.4.5 Interrupt request hold ..... 819

**CHAPTER 24 STANDBY FUNCTION ..... 820**

**24.1 Standby Function ..... 820**
**24.2 Registers Controlling Standby Function ..... 821**
**24.3 Standby Function Operation ..... 822**
24.3.1 HALT mode ..... 822
24.3.2 STOP mode ..... 828
24.3.3 SNOOZE mode ..... 834

**CHAPTER 25 RESET FUNCTION ..... 838**

**25.1 Timing of Reset Operation ..... 840**
**25.2 States of Operation During Reset Periods ..... 842**
**25.3 Register for Confirming Reset Source ..... 844**
25.3.1 Reset control flag register (RESF) ..... 844
25.3.2 Power-on-reset status register (PORSR) ..... 847

**CHAPTER 26 POWER-ON-RESET CIRCUIT ..... 848**

**26.1 Functions of Power-on-reset Circuit ..... 848**
**26.2 Configuration of Power-on-reset Circuit ..... 849**
**26.3 Operation of Power-on-reset Circuit ..... 849**

**CHAPTER 27 VOLTAGE DETECTOR ..... 854**

**27.1 Functions of Voltage Detector ..... 854**
**27.2 Configuration of Voltage Detector ..... 855**
**27.3 Registers Controlling Voltage Detector ..... 855**
27.3.1 Voltage detection register (LVIM) ..... 856
27.3.2 Voltage detection level register (LVIS) ..... 857
**27.4 Operation of Voltage Detector ..... 860**
27.4.1 When used as reset mode ..... 860
27.4.2 When used as interrupt mode ..... 862
27.4.3 When used as interrupt & reset mode ..... 864
**27.5 Cautions for Voltage Detector ..... 870**

**CHAPTER 28 BATTERY BACKUP FUNCTION ..... 872**

**28.1 Functions of Battery Backup ..... 872**
28.1.1 Pin configuration ..... 872
**28.2 Registers ..... 873**
28.2.1 Battery backup power switching control register 0 (BUPCTL0) ..... 873

28.2.2 Battery backup power switching control register 1 (BUPCTL1) ... 875
28.2.3 Global digital input disable register (GDIDIS) ... 875
**28.3 Operation ... 876**
28.3.1 Battery backup function ... 876
**28.4 Usage Notes ... 878**

**CHAPTER 29 OSCILLATION STOP DETECTOR ... 879**

**29.1 Functions of Oscillation Stop Detector ... 879**
**29.2 Configuration of Oscillation Stop Detector ... 880**
**29.3 Registers Used by Oscillation Stop Detector ... 881**
29.3.1 Peripheral enable register 1 (PER1) ... 881
29.3.2 Subsystem clock supply mode control register (OSMC) ... 882
29.3.3 Oscillation stop detection control register (OSDC) ... 883
**29.4 Operation of Oscillation Stop Detector ... 884**
29.4.1 How the oscillation stop detector operates ... 884
**29.5 Cautions on Using the Oscillation Stop Detector ... 885**

**CHAPTER 30 SAFETY FUNCTIONS ... 886**

**30.1 Overview of Safety Functions ... 886**
**30.2 Registers Used by Safety Functions ... 887**
**30.3 Operation of Safety Functions ... 887**
30.3.1 Flash memory CRC operation function (high-speed CRC) ... 887
30.3.1.1 Flash memory CRC control register (CRC0CTL) ... 888
30.3.1.2 Flash memory CRC operation result register (PGCRCL) ... 889
30.3.2 CRC operation function (general-purpose CRC) ... 891
30.3.2.1 CRC input register (CRCIN) ... 891
30.3.2.2 CRC data register (CRCD) ... 892
30.3.3 RAM parity error detection function ... 893
30.3.3.1 RAM parity error control register (RPECTL) ... 893
30.3.4 RAM guard function ... 895
30.3.4.1 Invalid memory access detection control register (IAWCTL) ... 895
30.3.5 SFR guard function ... 896
30.3.5.1 Invalid memory access detection control register (IAWCTL) ... 896
30.3.6 Invalid memory access detection function ... 897
30.3.6.1 Invalid memory access detection control register (IAWCTL) ... 898
30.3.7 Frequency detection function ... 899
30.3.7.1 Timer input select register 0 (TIS0) ... 900
30.3.8 A/D test function ... 901
30.3.8.1 A/D test register (ADTES) ... 902
30.3.8.2 Analog input channel specification register (ADS) ... 903
30.3.9 Digital output signal level detection function for I/O ports ... 904
30.3.9.1 Port mode select register (PMS) ... 904

**CHAPTER 31 REGULATOR ..... 905**

**31.1 Regulator Overview ..... 905**

**CHAPTER 32 OPTION BYTE ..... 906**

**32.1 Functions of Option Bytes ..... 906**

32.1.1 User option byte (000C0H to 000C2H/010C0H to 010C2H) ..... 906

32.1.2 On-chip debug option byte (000C3H/ 010C3H) ..... 907

**32.2 Format of User Option Byte ..... 908**

**32.3 Format of On-chip Debug Option Byte ..... 912**

**32.4 Setting of Option Byte ..... 913**

**CHAPTER 33 FLASH MEMORY ..... 914**

**33.1 Serial Programming Using Flash Memory Programmer ..... 916**

33.1.1 Programming environment ..... 917

33.1.2 Communication mode ..... 917

**33.2 Serial Programming Using External Device (That Incorporates UART) ..... 918**

33.2.1 Programming environment ..... 918

33.2.2 Communication mode ..... 919

**33.3 Connection of Pins on Board ..... 920**

33.3.1 P40/TOOL0 pin ..... 920

33.3.2 RESET pin ..... 920

33.3.3 Port pins ..... 921

33.3.4 REGC pin ..... 921

33.3.5 X1 and X2 pins ..... 921

33.3.6 Power supply ..... 921

**33.4 Serial Programming Method ..... 922**

33.4.1 Serial programming procedure ..... 922

33.4.2 Flash memory programming mode ..... 923

33.4.3 Selecting communication mode ..... 924

33.4.4 Communication commands ..... 925

**33.5 Self-Programming ..... 927**

33.5.1 Self-programming procedure ..... 928

33.5.2 Boot swap function ..... 929

33.5.3 Flash shield window function ..... 931

**33.6 Security Settings ..... 932**

**CHAPTER 34 ON-CHIP DEBUG FUNCTION ..... 934**

**34.1 Connecting E1 On-chip Debugging Emulator ..... 934**

**34.2 On-Chip Debug Security ID ..... 935**

**34.3 Securing of User Resources ..... 935**

**CHAPTER 35 BCD CORRECTION CIRCUIT ..... 937**

**35.1 BCD Correction Circuit Function ..... 937**

**35.2 Registers Used by BCD Correction Circuit ..... 937**

35.2.1 BCD correction result register (BCDADJ) ..... 937

**35.3 BCD Correction Circuit Operation ..... 938**

**CHAPTER 36 INSTRUCTION SET ..... 940**

**36.1 Conventions Used in Operation List ..... 940**

36.1.1 Operand identifiers and specification methods ..... 940

36.1.2 Description of operation column ..... 941

36.1.3 Description of flag operation column ..... 942

36.1.4 PREFIX instruction ..... 942

**36.2 Operation List ..... 943**

**CHAPTER 37 ELECTRICAL SPECIFICATIONS ..... 961**

**37.1 Absolute Maximum Ratings ..... 962**

**37.2 Oscillator Characteristics ..... 965**

37.2.1 X1, XT1 oscillator characteristics ..... 965

37.2.2 On-chip oscillator characteristics ..... 966

**37.3 DC Characteristics ..... 967**

37.3.1 Pin characteristics ..... 967

37.3.2 Supply current characteristics ..... 972

**37.4 AC Characteristics ..... 979**

**37.5 Peripheral Functions Characteristics ..... 982**

37.5.1 Serial array unit ..... 982

37.5.2 Serial interface IICA ..... 1005

**37.6 Analog Characteristics ..... 1008**

37.6.1 A/D converter characteristics ..... 1008

37.6.2 24-bit ΔΣ A/D converter characteristics ..... 1010

37.6.3 Temperature sensor 2 characteristics ..... 1012

37.6.4 Comparator ..... 1013

37.6.5 POR circuit characteristics ..... 1013

37.6.6 LVD circuit characteristics ..... 1014

37.6.7 Power supply voltage rising slope characteristics ..... 1015

**37.7 Battery Backup Function ..... 1016**

**37.8 LCD Characteristics ..... 1017**

37.8.1 Resistance division method ..... 1017

37.8.2 Internal voltage boosting method ..... 1018

37.8.3 Capacitor split method ..... 1020

**37.9 RAM Data Retention Characteristics ..... 1021**

**37.10 Flash Memory Programming Characteristics ..... 1021**
**37.11 Dedicated Flash Memory Programmer Communication (UART) ..... 1021**
**37.12 Timing Specs for Switching Flash Memory Programming Modes ..... 1022**

**CHAPTER 38 PACKAGE DRAWINGS ..... 1023**

**38.1 80-pin Products ..... 1023**
**38.2 100-pin Products ..... 1024**

**APPENDIX A REVISION HISTORY ..... 1025**

**A.1 Major Revisions in This Edition ..... 1025**
**A.2 Revision History of Preceding Editions ..... 1029**

# CHAPTER 1 OUTLINE

## 1.1 Features

Ultra-low power consumption technology

- $V_{DD}$ = single power supply voltage of 1.9 to 5.5 V
- HALT mode
- STOP mode
- SNOOZE mode

RL78 CPU core

- CISC architecture with 3-stage pipeline
- Minimum instruction execution time: Can be changed from high speed (0.04167 μs: @ 24 MHz operation with high-speed on-chip oscillator) to ultra-low speed (30.5 μs: @ 32.768 kHz operation with 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: 6 KB or 8 KB

Code flash memory

- Code flash memory: 64 KB or 128 KB
- Block size: 1 KB
- Prohibition of block erase and rewriting (security function)
- On-chip debug function
- Self-programming (with boot swap function/flash shield window function)

High-speed on-chip oscillator

- Select from 24 MHz (TYP.), 12 MHz (TYP.), 6 MHz (TYP.), and 3 MHz (TYP.)
- High accuracy: ±1.0 % ($V_{DD}$ = 1.9 to 5.5 V, $T_A$ = –20 to +85°C)
- On-chip high-speed on-chip oscillator clock frequency correction function

Operating ambient temperature

- $T_A$ = -40 to +85°C

Power management and reset function

- On-chip power-on-reset (POR) circuit
- On-chip voltage detector (LVD) (Select interrupt and reset from 11 levels)

Data transfer controller (DTC)

- Transfer mode: Normal mode, repeat mode, block mode
- Activation source: Start by interrupt sources (40 sources)
- Chain transfer function

Serial interface

- CSI: 1channel
- UART/UART (LIN-bus supported): 3 channels
- $I^2C$/Simplified $I^2C$ communication: 3 channels
- IrDA: 1 channel

Timer

- 16-bit timer: 8 channels
- 12-bit interval timer: 1 channel
- 8-bit interval timer: 4 channels
- Real-time clock 2: 1 channel (calendar for 99 years, alarm function, and clock correction function)
- Watchdog timer: 1 channel (operable with the dedicated low-speed on-chip oscillator)
- Oscillation stop detection circuit: 1 channel

LCD controller/driver

- Internal voltage boosting method, capacitor split method, and external resistance division method are switchable
- Segment signal output: 34 (30)Note 1 to 42 (38)Note 1
- Common signal output: 4 (8)Note 1

A/D converter

- 8/10-bit resolution A/D converter ($V_{DD}$ = 1.9 to 5.5 V): 4 or 6 channels
- 24-Bit ΔΣ A/D converter: 3 or 4 channels
- Internal reference voltage (1.45 V) and temperature sensorNote 2

Comparator

- 2 channels
- Operation mode: Comparator high-speed mode, comparator low-speed mode, or window mode
- External reference voltage and internal reference voltage are selectable

I/O port

- I/O port: 53 or 69 (N-ch open drain I/O [withstand voltage of 6 V]: 3,
  N-ch open drain I/O [$V_{DD}$ withstand voltage]: 13)
- Can be set to N-ch open drain, TTL input buffer, and on-chip pull-up resistor
- Different potential interface: Can connect to a 1.8/2.5/3 V device
- On-chip clock output/buzzer output controller

Others

- On-chip BCD (binary-coded decimal) correction circuit
- On-chip battery backup function

**Notes 1.** The values in parentheses are the number of signal outputs when 8 com is used.

**2.** Can be selected only in HS (high-speed main) mode

**Remark** The functions mounted depend on the product. See **1.6 Outline of Functions**.

O ROM, RAM capacities

| Flash ROM | Data flash | RAM | RL78/I1B | |
|---|---|---|---|---|
| | | | 80 pins | 100 pins |
| 128 KB | – | 8 KB[Note] | R5F10MMG | R5F10MPG |
| 64 KB | – | 6 KB | R5F10MME | R5F10MPE |

**Note** This is about 7 KB when the self-programming function is used. (For details, see **CHAPTER 3**)

## 1.2 List of Part Numbers

**Figure 1-1. Part Number, Memory Size, and Package of RL78/I1B**

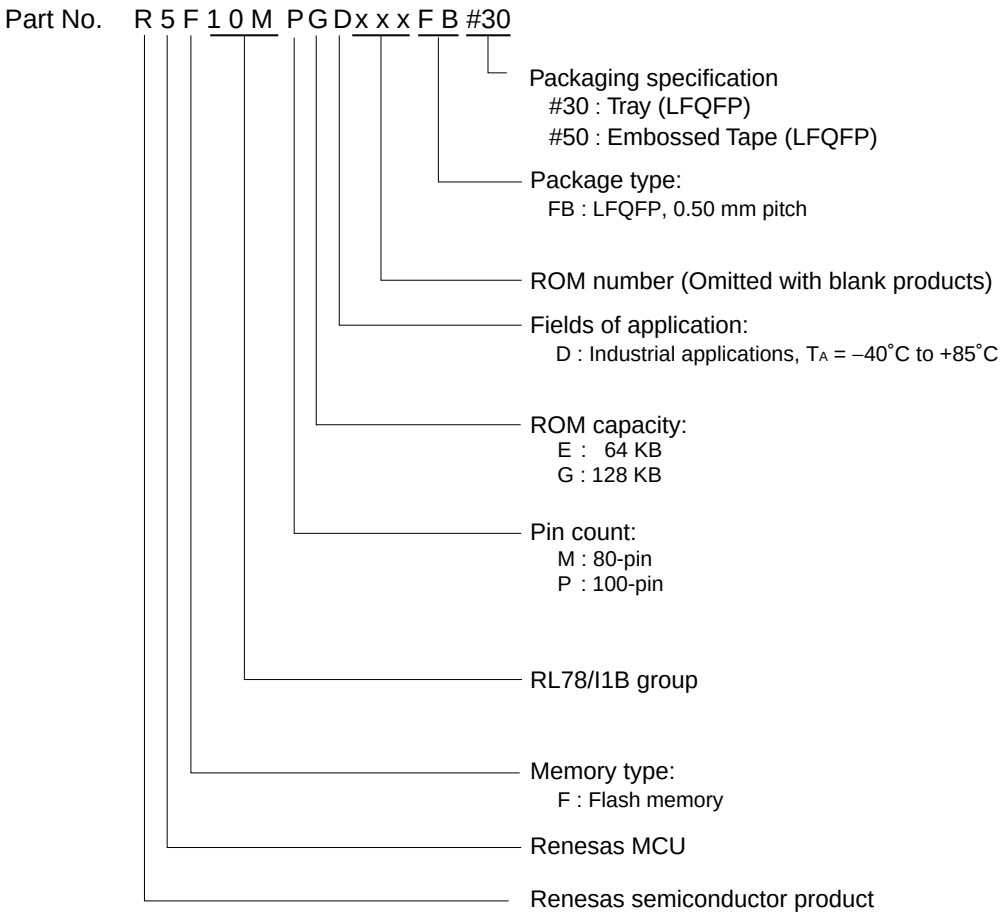



**Table 1-1. List of Ordering Part Numbers**

| Pin count | Package | Data flash | Fields of Application[Note] | Ordering Part Number |
|---|---|---|---|---|
| 80 pins | 80-pin plastic LFQFP (12 × 12 mm, 0.5 mm pitch) | – | D | R5F10MMEDFB#30, R5F10MMGDFB#30<br>R5F10MMEDFB#50, R5F10MMGDFB#50 |
| 100 pins | 100-pin plastic LFQFP (14 × 14 mm, 0.5 mm pitch) | – | D | R5F10MPEDFB#30, R5F10MPGDFB#30<br>R5F10MPEDFB#50, R5F10MPGDFB#50 |

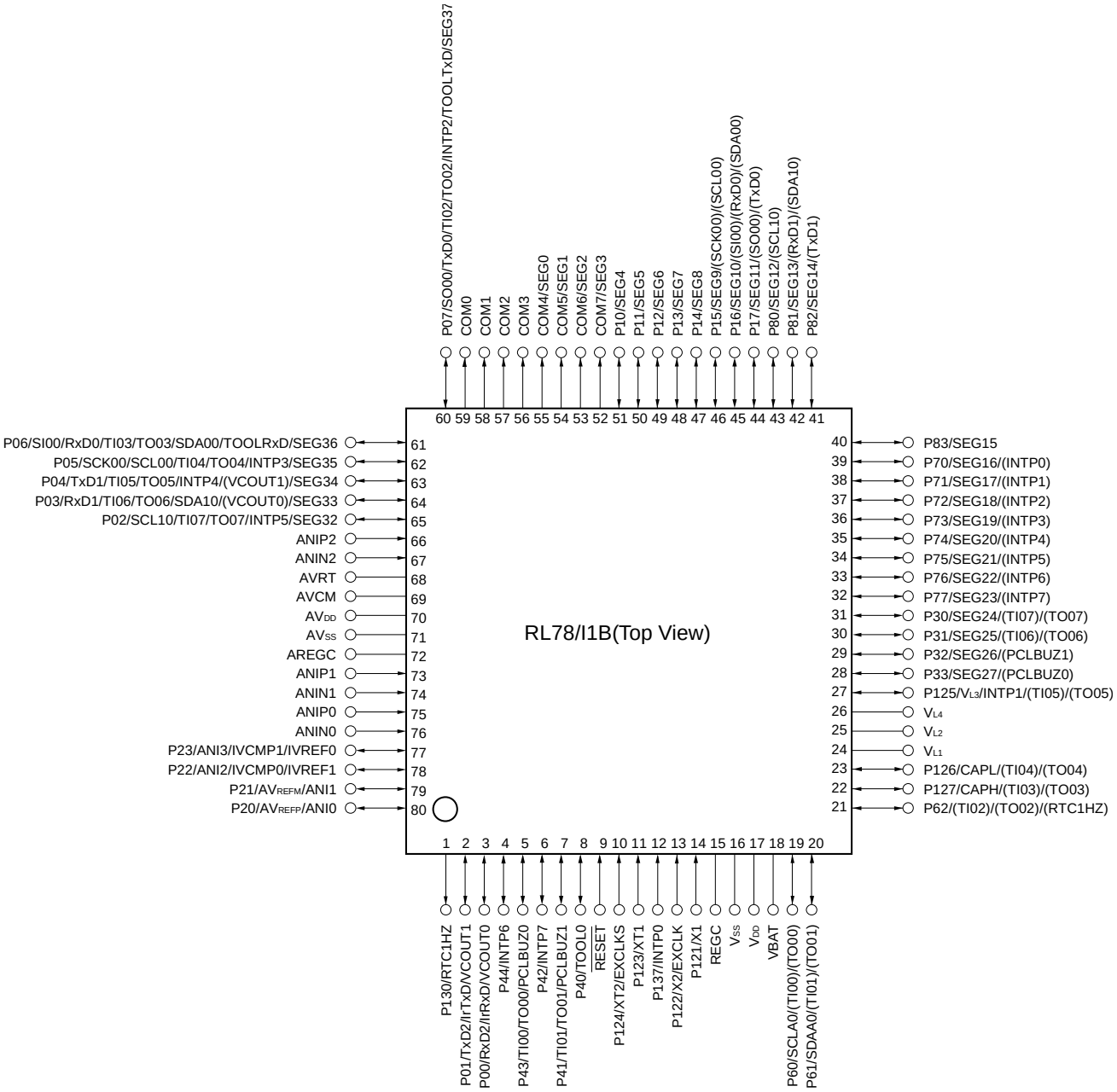
**Note** For the fields of application, see **Figure 1-1 Part Number, Memory Size, and Package of RL78/I1B**.

**Caution The ordering part numbers represent the numbers at the time of publication. For the latest ordering part numbers, refer to the target product page of the Renesas Electronics website.**

## 1.3 Pin Configuration (Top View)

### 1.3.1 80-pin products

<R> • 80-pin plastic LFQFP (12 × 12 mm, 0.5 mm pitch)



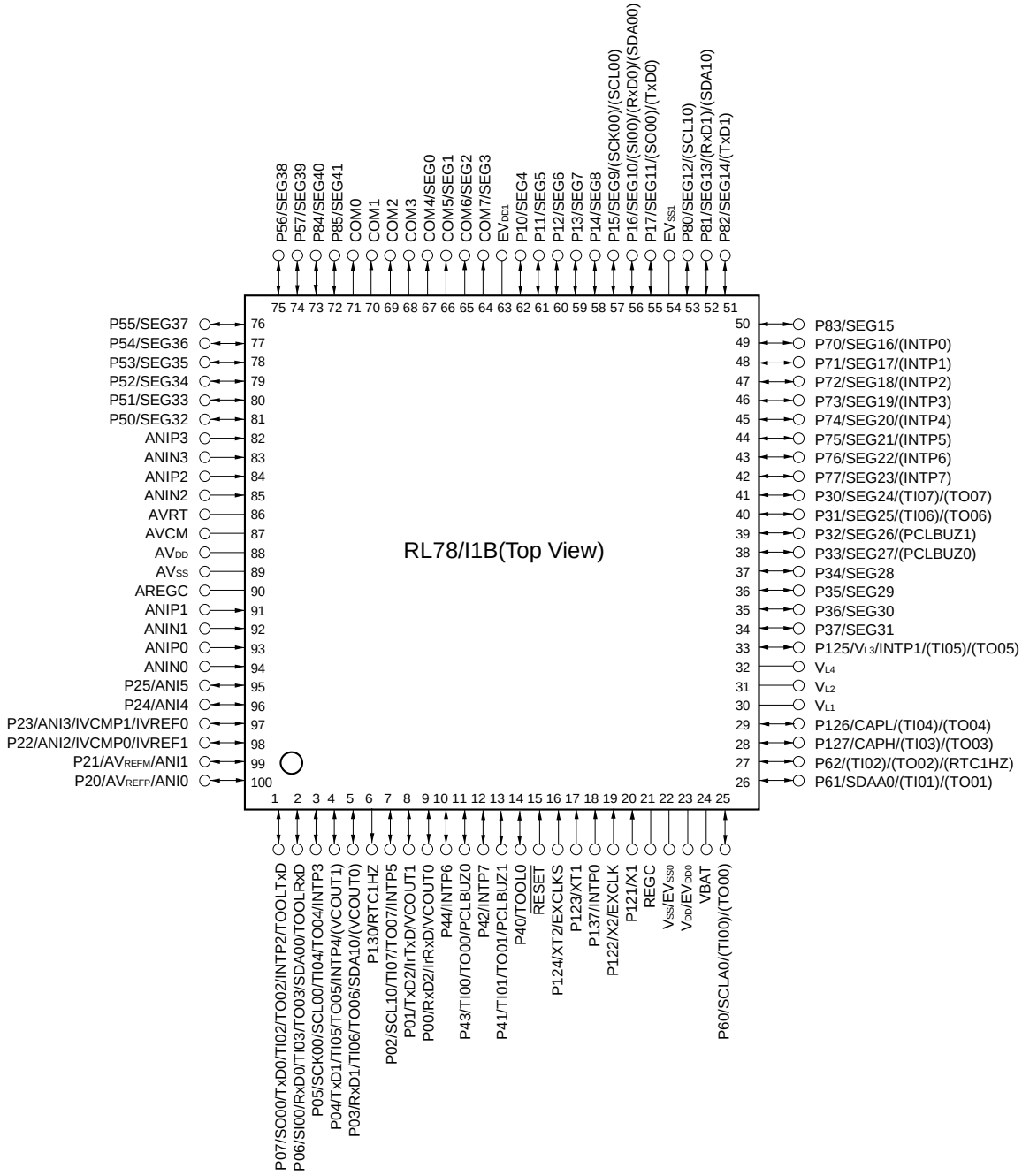

**Caution Connect the REGC pin to VSS via a capacitor (0.47 to 1 µF).**

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

**2.** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR).**

### 1.3.2 100-pin products

<R> • 100-pin plastic LFQFP (14 × 14 mm, 0.5 mm pitch)




**Cautions 1. Make $EV_{SS1}$ the same potential as $V_{SS}/EV_{SS0}$.**

**2. Make $EV_{DD1}$ the same potential as $V_{DD}/EV_{DD0}$.**

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

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

**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 $V_{DD}$ and $EV_{DD1}$ pins and connect the $V_{SS}$ and $EV_{SS1}$ pins to separate ground lines.

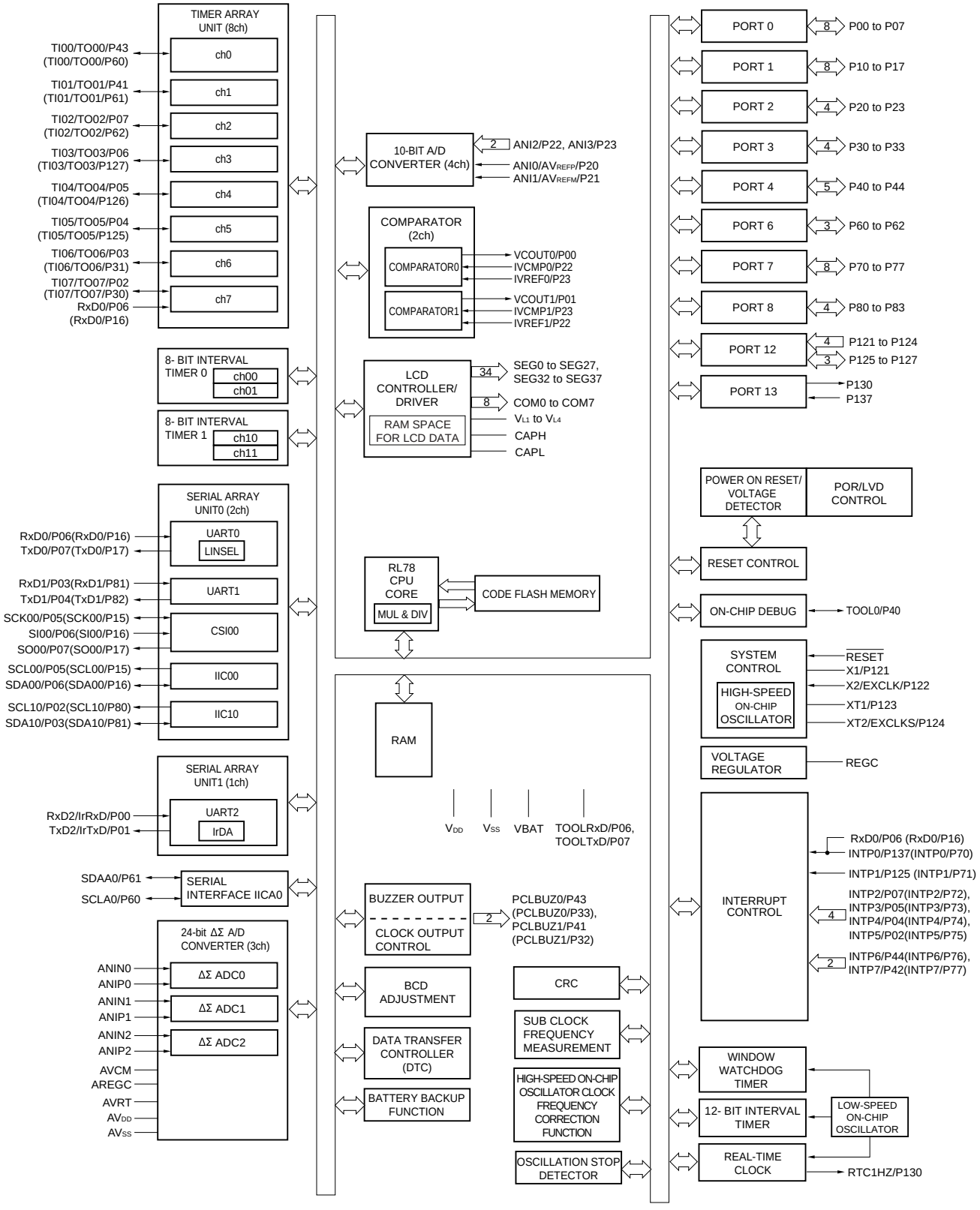
**3.** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR).**

## 1.4 Pin Identification

| | |
|---|---|
| ANI0 to ANI5: | Analog Input |
| ANIN0 to ANIN3, | |
| ANIP0 to ANIP3: | Analog Input for ΔΣ ADC |
| AREGC: | Regulator Capacitance for ΔΣ ADC |
| AVCM: | Control forΔΣADC |
| $AV_{DD}$: | Power Supply forΔΣADC |
| $AV_{REFM}$: | A/D Converter Reference Potential (– side) Input |
| $AV_{REFP}$: | A/D Converter Reference Potential (+ side) Input |
| AVRT: | Reference Potential for ΔΣ ADC |
| $AV_{SS}$: | Ground forΔΣADC |
| CAPH, CAPL: | Capacitor Connection for LCD Controller/Driver |
| COM0 to COM7: | Common Signal Output for LCD Controller/Driver |
| $EV_{DD0}$, $EV_{DD1}$: | Power Supply for Port |
| $EV_{SS0}$, $EV_{SS1}$: | Ground for Port |
| EXCLK: | External Clock Input (Main System Clock) |
| EXCLKS: | External Clock Input (Subsystem clock) |
| INTP0 to INTP7: | Interrupt Request From Peripheral |
| IrRxD: | Receive Data for IrDA |
| IrTxD: | Transmit Data for IrDA |
| IVCMP0, IVCMP1: | Comparator Input |
| IVREF0, IVREF1: | Comparator Reference Input |
| P00 to P07: | Port 0 |
| P10 to P17: | Port 1 |
| P20 to P25: | Port 2 |
| P30 to P37: | Port 3 |
| P40 to P44: | Port 4 |
| P50 to P57: | Port 5 |
| P60 to P62: | Port 6 |
| P70 to P77: | Port 7 |
| P80 to P85: | Port 8 |
| P121 to P127: | Port 12 |
| P130, P137: | Port 13 |
| PCLBUZ0, PCLBUZ1: | Programmable Clock Output/Buzzer Output |
| REGC: | Regulator Capacitance |
| RESET: | Reset |
| RTC1HZ: | Real-time Clock Correction Clock (1 Hz) Output |
| RxD0 to RxD2: | Receive Data for UART |
| SCK00: | Serial Clock Input/Output for CSI |
| SCLA0, SCL00, SCL10: | Serial Clock Input/Output for IIC |
| SDAA0, SDA00, SDA10: | Serial Data Input/Output for IIC |
| SEG0 to SEG41: | Segment Signal Output for LCD Controller/Driver |
| SI00: | Serial Data Input for CSI |
| SO00: | Serial Data Output for CSI |
| TI00 to TI07: | Timer Input |
| TO00 to TO07: | Timer Output |
| TOOL0: | Data Input/Output for Tool |
| TOOLRxD, TOOLTxD: | Data Input/Output for External Device |
| TxD0 to TxD2: | Transmit Data for UART |
| VBAT: | Battery Backup Power Supply |
| VCOUT0, VCOUT1: | Comparator Output |
| $V_{DD}$: | Power Supply |
| $V_{L1}$ to $V_{L4}$: | Voltage for Driving LCD |
| $V_{SS}$: | Ground |
| X1, X2: | Crystal Oscillator (Main System Clock) |
| XT1, XT2: | Crystal Oscillator (Subsystem Clock) |

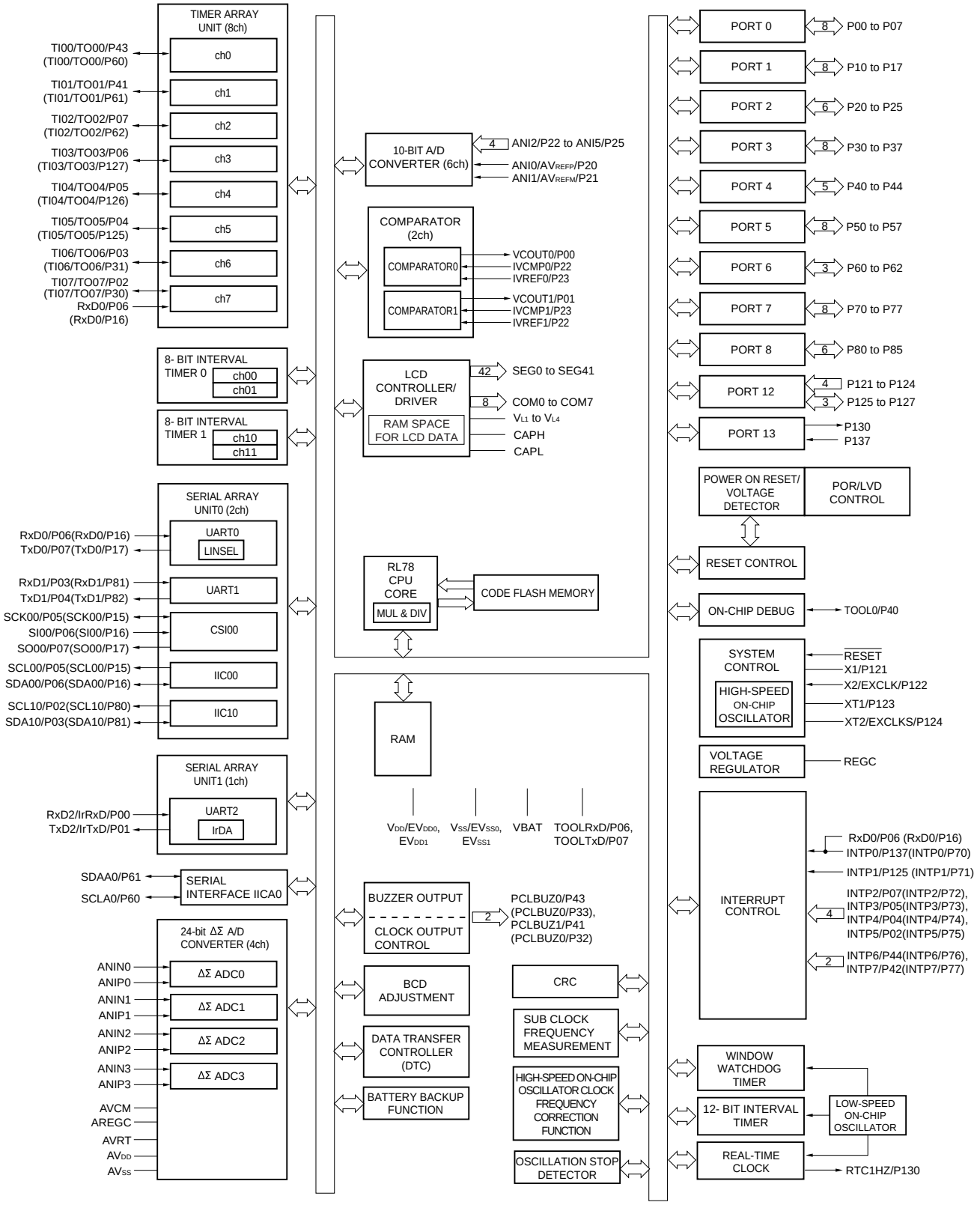
## 1.5 Block Diagram

### 1.5.1 80-pin products




**Remark** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR)**.

### 1.5.2 100-pin products




**Remark** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR).**

## 1.6 Outline of Functions

(1/2)

| Item | | 80-pin | | 100-pin | |
|---|---|---|---|---|---|
| | | R5F10MMEDFB | R5F10MMGDFB | R5F10MPEDFB | R5F10MPGDFB |
| Code flash memory (KB) | | 64 | 128 | 64 | 128 |
| Data flash memory (KB) | | – | | | |
| RAM (KB) | | 6 | 8 Note 1 | 6 | 8 Note 1 |
| Address space | | 1 MB | | | |
| <R> Main system clock | High-speed system clock | X1 (crystal/ceramic) oscillation, external main system clock input (EXCLK)<br>HS (High-speed main) mode: 1 to 20 MHz ($V_{DD}$ = 2.7 to 5.5 V),<br>HS (High-speed main) mode: 1 to 16 MHz ($V_{DD}$ = 2.4 to 5.5 V),<br>LS (Low-speed main) mode: 1 to 8 MHz ($V_{DD}$ = 1.9 to 5.5 V) | | | |
| | High-speed on-chip oscillator clock | HS (High-speed main) mode: 24/12/6/3 MHz ($V_{DD}$ = 2.7 to 5.5 V),<br>HS (High-speed main) mode: 12/6/3 MHz ($V_{DD}$ = 2.4 to 5.5 V),<br>LS (Low-speed main) mode: 6/3 MHz ($V_{DD}$ = 1.9 to 5.5 V) | | | |
| Subsystem clock | | XT1 (crystal) oscillation, external subsystem clock input (EXCLKS)<br>32.768 kHz (TYP.): $V_{DD}$ = 1.9 to 5.5 V | | | |
| High-speed on-chip oscillator clock frequency correction function | | Correct the frequency of the high-speed on-chip oscillator clock by the subsystem clock. | | | |
| Low-speed on-chip oscillator | | 15 kHz (TYP.): $V_{DD}$ = 1.9 to 5.5 V | | | |
| General-purpose register | | 8 bits × 8 registers × 4 banks | | | |
| Minimum instruction execution time | | 0.04167 μs (High-speed on-chip oscillator: $f_{IH}$ = 24 MHz operation) | | | |
| | | 0.05 μs (High-speed system clock: $f_{MX}$ = 20 MHz operation) | | | |
| | | 30.5 μs (Subsystem clock: $f_{SUB}$ = 32.768 kHz operation) | | | |
| Instruction set | | • Data transfer (8/16 bits)<br>• Adder and subtractor/logical operation (8/16 bits)<br>• Multiplication (16 bits × 16 bits), division (32 bits ÷ 32 bits)<br>• Multiplication and accumulation (16 bits × 16 bits + 32 bits)<br>• Rotate, barrel shift, and bit manipulation (set, reset, test, and boolean operation), etc. | | | |
| I/O port | Total | 53 | | 69 | |
| | CMOS I/O | 44 | | 60 | |
| | CMOS input | 5 | | 5 | |
| | CMOS output | 1 | | 1 | |
| | N-ch O.D I/O (6 V tolerance) | 3 | | 3 | |
| Timer | 16-bit timer TAU | 8 channels | | | |
| | Watchdog timer | 1 channel | | | |
| | 12-bit interval timer | 1 channel | | | |
| | 8-bit interval timer | 4 channels | | | |
| | Real-time clock 2 | 1 channel | | | |
| | Oscillation stop detection circuit | 1 channel | | | |
| | Timer output | Timer outputs: 8 channels<br>PWM outputs: 7 Note 2 | | | |
| | RTC output | 1 channel<br>• 1 Hz (subsystem clock: $f_{SUB}$ = 32.768 kHz) | | | |

**Notes 1.** In the case of the 8 KB, this is about 7 KB when the self-programming function is used.

**2.** The number of outputs varies, depending on the setting of channels in use and the number of the master (see **7.9.3 Operation as multiple PWM output function**).

(2/2)

| Item | | | 80-pin | | 100-pin | |
|---|---|---|---|---|---|---|
| | | | R5F10MMEDFB | R5F10MMGDFB | R5F10MPEDFB | R5F10MPGDFB |
| Clock output/buzzer output | | | 2 | | | |
| | | | • 2.44 kHz, 4.88 kHz, 9.76 kHz, 1.25 MHz, 2.5 MHz, 5 MHz, 10 MHz (Main system clock: $f_{MAIN}$ = 20 MHz operation)<br>• 256 Hz, 512 Hz, 1.024 kHz, 2.048 kHz, 4.096 kHz, 8.192 kHz, 16.384 kHz, 32.768 kHz (Subsystem clock: $f_{SUB}$ = 32.768 kHz operation) | | | |
| 10-bit resolution A/D converter | | | 4 channels | | 6 channels | |
| 24-Bit ΔΣ A/D Converter | | | 3 channels | | 4 channels | |
| | SNDR | | Typ. 80 dB (gain ×1)<br>Min. 69 dB (gain ×16)<br>Min. 65 dB (gain ×32) | | | |
| | Sampling frequency | | 3.906 kHz/1.953 kHz | | | |
| | PGA | | Current ch: ×1, ×2, ×4, ×8, ×16, ×32<br>Voltage ch: ×1, ×2, ×4, ×8, ×16 | | | |
| Comparator | | | 2 channels | | | |
| Serial interface | | | • CSI/UART/simplified $I^2C$: 1 channel<br>• UART/simplified $I^2C$: 1 channel<br>• UART/IrDA: 1 channel | | | |
| | $I^2C$ bus | | 1 channel | | | |
| Data transfer controller (DTC) | | | 30 sources | | | |
| LCD controller/driver | | | Internal voltage boosting method, capacitor split method, and external resistance division method are switchable. | | | |
| | Segment signal output | | 34 (30)[Note 1] | | 42 (38)[Note 1] | |
| | Common signal output | | 4 (8)[Note 1] | | | |
| Vectored interrupt sources | | Internal | 34 | | | |
| | | External | 10 | | | |
| Reset | | | • Reset by $\overline{\text{RESET}}$ pin<br>• Internal reset by watchdog timer<br>• Internal reset by power-on-reset<br>• Internal reset by voltage detector<br>• Internal reset by illegal instruction execution[Note 2]<br>• Internal reset by RAM parity error<br>• Internal reset by illegal-memory access | | | |
| Power-on-reset circuit | | | • Power-on-reset: 1.51 V (TYP.)<br>• Power-down-reset: 1.50 V (TYP.) | | | |
| Voltage detector | | | • Rising edge : 1.98 V to 4.06 V (11 stages)<br>• Falling edge : 1.94 V to 3.98 V (11 stages) | | | |
| Battery backup function | | | Provided | | | |
| On-chip debug function | | | Provided | | | |
| Power supply voltage | | | $V_{DD}$ = 1.9 to 5.5 V | | | |
| Operating ambient temperature | | | $T_A$ = –40 to +85 °C | | | |

**Notes 1.** The values in parentheses are the number of signal outputs when 8 com is used.

**2.** This reset occurs when instruction code FFH is executed.
This reset does not occur during emulation using an in-circuit emulator or an on-chip debugging emulator.

# CHAPTER 2 PIN FUNCTIONS

## 2.1 Port Function List

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) 80-pin products**

| Power Supply | Corresponding Pins |
|---|---|
| $V_{DD}$ | Port pins other than P20 to P23, P121 to P124, and P137[Note 1] |
| $V_{DD}$ or VBAT[Notes 2, 3] | • P20 to P23, P121 to P124, and P137<br>• $\overline{\text{RESET}}$, REGC |
| $AV_{DD}$ | ANIP0 to ANIP2 and ANIN0 to ANIN2 |

**(2) 100-pin products**

| Power Supply | Corresponding Pins |
|---|---|
| $EV_{DD1}$ | Port pins other than P20 to P25, P121 to P124, and P137[Note 1] |
| $V_{DD}$ or VBAT[Notes 2, 3] | • P20 to P25, P121 to P124, and P137<br>• $\overline{\text{RESET}}$, REGC |
| $AV_{DD}$ | ANIP0 to ANIP3 and ANIN0 to ANIN3 |

**Notes 1.** When using the battery backup function, the power supply of the internal I/O buffer of this pin is powered from the $V_{DD}$ pin even when switch to power from VBAT pin. If the power of the $V_{DD}$ pin is lost, make sure the input voltage does not exceed the absolute maximum rating.

**2.** The power supply pin for the I/O buffers can be switched between $V_{DD}$ and VBAT by using the battery backup function.

**3.** The input/output signal voltage of the pin that is defined as "$V_{DD}$ or VBAT" must match the supply voltage of the I/O buffer.

**Caution The $EV_{DD1}$ pin must be at the same potential as $V_{DD}$/$EV_{DD0}$ pin.**

Set in each port I/O, buffer, pull-up resistor is also valid for alternate functions.

### 2.1.1 80-pin products

(1/2)

<R>

| Function Name | Pin Type | I/O | After Reset Release | Alternate Function | Function |
|---|---|---|---|---|---|
| P00 | 8-1-3 | I/O | Input port | RxD2/IrRxD/VCOUT0 | Port 0.<br>8-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Input of P00, P03, P05, and P06 can be set to TTL input buffer.<br>Output of P01 to P07 can be set to N-ch open-drain output ($V_{DD}$ tolerance).<br>Output of P02 to P07 can be set to LCD output[Note 2]. |
| P01 | 7-1-4 | | | TxD2/IrTxD/VCOUT1 | |
| P02 | 7-5-10 | | Digital input invalid[Note 1] | SCL10/TI07/TO07/ INTP5/SEG32 | |
| P03 | 8-5-10 | | | RxD1/TI06/TO06/SDA10/ (VCOUT0)/SEG33 | |
| P04 | 7-5-10 | | | TxD1/TI05/TO05/INTP4/ (VCOUT1)/SEG34 | |
| P05 | 8-5-10 | | | SCK00/SCL00/TI04/ TO04/INTP3/SEG35 | |
| P06 | | | | SI00/RxD0/TI03/TO03/ SDA00/TOOLRxD/SEG36 | |
| P07 | 7-5-10 | | | SO00/TxD0/TI02/TO02/ INTP2/TOOLTxD/SEG37 | |
| P10 | 7-5-4 | I/O | Digital input invalid[Note 1] | SEG4 | Port 1.<br>8-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Input of P15 and P16 can be set to TTL input buffer.<br>Output of P15 to P17 can be set to N-ch open-drain output ($V_{DD}$ tolerance).<br>Can be set to LCD output[Note 2]. |
| P11 | | | | SEG5 | |
| P12 | | | | SEG6 | |
| P13 | | | | SEG7 | |
| P14 | | | | SEG8 | |
| P15 | 8-5-10 | | | SEG9/(SCK00)/(SCL00) | |
| P16 | | | | SEG10/(SI00)/ (RxD0)/(SDA00) | |
| P17 | 7-5-10 | | | SEG11/(SO00)/(TxD0) | |
| P20 | 4-3-3 | I/O | Analog input port | $AV_{REFP}$/ANI0 | Port 2.<br>4-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Can be set to analog input[Note 3]. |
| P21 | | | | $AV_{REFM}$/ANI1 | |
| P22 | 4-9-2 | | | ANI2/IVCMP0/IVREF1 | |
| P23 | | | | ANI3/IVCMP1/IVREF0 | |
| P30 | 7-5-4 | I/O | Digital input invalid[Note 1] | SEG24/(TI07)/(TO07) | Port 3.<br>4-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Can be set to LCD output[Note 2]. |
| P31 | | | | SEG25/(TI06)/(TO06) | |
| P32 | | | | SEG26/(PCLBUZ1) | |
| P33 | | | | SEG27/(PCLBUZ0) | |

**Notes 1.** "Digital input invalid" refers to the state in which all the digital outputs, digital inputs, and LCD outputs are disabled.

**2.** Digital or LCD for each pin can be selected with the port mode register x (PMx) and the LCD port function register x (PFSEGx) (can be set in 1-bit unit).

**3.** Setting digital or analog to each pin can be done in A/D port configuration register (ADPC).

**Remark** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR)**.

(2/2)

<R>

| Function Name | Pin Type | I/O | After Reset Release | Alternate Function | Function |
|---|---|---|---|---|---|
| P40 | 7-1-3 | I/O | Input port | TOOL0 | Port 4.<br>5-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port. |
| P41 | | | | TI01/TO01/PCLBUZ1 | |
| P42 | | | | INTP7 | |
| P43 | | | | TI00/TO00/PCLBUZ0 | |
| P44 | | | | INTP6 | |
| P60 | 12-1-3 | I/O | Input port | SCLA0/(TI00)/(TO00) | Port 6.<br>3-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Can be set to N-ch open-drain output (6 V tolerance). |
| P61 | | | | SDAA0/(TI01)/(TO01) | |
| P62 | | | | (TI02)/(TO02)/(RTC1HZ) | |
| P70 | 7-5-4 | I/O | Digital input invalid[Note 1] | SEG16/(INTP0) | Port 7.<br>8-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Can be set to LCD output[Note 2]. |
| P71 | | | | SEG17/(INTP1) | |
| P72 | | | | SEG18/(INTP2) | |
| P73 | | | | SEG19/(INTP3) | |
| P74 | | | | SEG20/(INTP4) | |
| P75 | | | | SEG21/(INTP5) | |
| P76 | | | | SEG22/(INTP6) | |
| P77 | | | | SEG23/(INTP7) | |
| P80 | 7-5-10 | I/O | Digital input invalid[Note 1] | SEG12/(SCL10) | Port 8.<br>4-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Input of P81 can be set to TTL input buffer.<br>Output of P80 to P82 can be set to N-ch open-drain output ($V_{DD}$ tolerance).<br>Can be set to LCD output[Note 2]. |
| P81 | 8-5-10 | | | SEG13/(RxD1)/(SDA10) | |
| P82 | 7-5-10 | | | SEG14/(TxD1) | |
| P83 | 7-5-4 | | | SEG15 | |
| P121 | 2-2-1 | Input | Input port | X1 | Port 12.<br>3-bit I/O port and 4-bit input only port.<br>For only P125 to P127, input/output can be specified in 1-bit units.<br>For only P125 to P127, use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>P125 to P127 can be set to LCD output[Note 2]. |
| P122 | | | | X2/EXCLK | |
| P123 | | | | XT1 | |
| P124 | | | | XT2/EXCLKS | |
| P125 | 7-5-6 | I/O | Digital input invalid[Note 1] | $V_{L3}$/INTP1/(TI05)/(TO05) | |
| P126 | 7-5-5 | | | CAPL/(TI04)/(TO04) | |
| P127 | | | | CAPH/(TI03)/(TO03) | |
| P130 | 1-1-4 | Output | Output port | RTC1HZ | Port 13.<br>1-bit output port and 1-bit input only port. |
| P137 | 2-1-2 | Input | Input port | INTP0 | |
| $\overline{\text{RESET}}$ | 2-1-1 | Input | – | – | Input only pin for external reset.<br>When external reset is not used, connect this pin to $V_{DD}$ directly or via a resistor. |

**Notes 1.** "Digital input invalid" refers to the state in which all the digital outputs, digital inputs, and LCD outputs are disabled.

**2.** Digital or LCD for each pin can be selected with the port mode register x (PMx) and the LCD port function register x (PFSEGx) (can be set in 1-bit unit).

**Remark** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR)**.

### 2.1.2 100-pin products

(1/3)

<R>

| Function Name | Pin Type | I/O | After Reset Release | Alternate Function | Function |
|---|---|---|---|---|---|
| P00 | 8-1-3 | I/O | Input port | RxD2/IrRxD/VCOUT0 | Port 0.<br>8-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Input of P00, P03, P05, and P06 can be set to TTL input buffer.<br>Output of P01 to P07 can be set to N-ch open-drain output ($V_{DD}$ tolerance). |
| P01 | 7-1-4 | | | TxD2/IrTxD/VCOUT1 | |
| P02 | | | | SCL10/TI07/TO07/INTP5 | |
| P03 | 8-1-4 | | | RxD1/TI06/TO06/ SDA10/(VCOUT0) | |
| P04 | 7-1-4 | | | TxD1/TI05/TO05/INTP4/ (VCOUT1) | |
| P05 | 8-1-4 | | | SCK00/SCL00/TI04/ TO04/INTP3 | |
| P06 | | | | SI00/RxD0/TI03/TO03/ SDA00/TOOLRxD | |
| P07 | 7-1-4 | | | SO00/TxD0/TI02/TO02/ INTP2/TOOLTxD | |
| P10 | 7-5-4 | I/O | Digital input invalid[Note 1] | SEG4 | Port 1.<br>8-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Input of P15 and P16 can be set to TTL input buffer.<br>Output of P15 to P17 can be set to N-ch open-drain output ($V_{DD}$ tolerance).<br>Can be set to LCD output[Note 2]. |
| P11 | | | | SEG5 | |
| P12 | | | | SEG6 | |
| P13 | | | | SEG7 | |
| P14 | | | | SEG8 | |
| P15 | 8-5-10 | | | SEG9/(SCK00)/(SCL00) | |
| P16 | | | | SEG10/(SI00)/(RxD0)/ (SDA00) | |
| P17 | 7-5-10 | | | SEG11/(SO00)/(TxD0) | |
| P20 | 4-3-3 | I/O | Analog input port | $AV_{REFP}$/ANI0 | Port 2.<br>6-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Can be set to analog input[Note 3]. |
| P21 | | | | $AV_{REFM}$/ANI1 | |
| P22 | 4-9-2 | | | ANI2/IVCMP0/IVREF1 | |
| P23 | | | | ANI3/IVCMP1/IVREF0 | |
| P24 | 4-3-3 | | | ANI4 | |
| P25 | | | | ANI5 | |
| P30 | 7-5-4 | I/O | Digital input invalid[Note 1] | SEG24/(TI07)/(TO07) | Port 3.<br>8-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Can be set to LCD output[Note 2]. |
| P31 | | | | SEG25/(TI06)/(TO06) | |
| P32 | | | | SEG26/(PCLBUZ1) | |
| P33 | | | | SEG27/(PCLBUZ0) | |
| P34 | | | | SEG28 | |
| P35 | | | | SEG29 | |
| P36 | | | | SEG30 | |
| P37 | | | | SEG31 | |

**Notes 1.** “Digital input invalid” refers to the state in which all the digital outputs, digital inputs, and LCD outputs are disabled.

**2.** Digital or LCD for each pin can be selected with the port mode register x (PMx) and the LCD port function register x (PFSEGx) (can be set in 1-bit unit).

**3.** Setting digital or analog to each pin can be done in A/D port configuration register (ADPC).

**Remark** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR)**.

(2/3)

<R>

| Function Name | Pin Type | I/O | After Reset Release | Alternate Function | Function |
|---|---|---|---|---|---|
| P40 | 7-1-3 | I/O | Input port | TOOL0 | Port 4.<br>5-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port. |
| P41 | | | | TI01/TO01/PCLBUZ1 | |
| P42 | | | | INTP7 | |
| P43 | | | | TI00/TO00/PCLBUZ0 | |
| P44 | | | | INTP6 | |
| P50 | 7-5-4 | I/O | Digital input invalid[Note 1] | SEG32 | Port 5.<br>8-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Can be set to LCD output[Note 2]. |
| P51 | | | | SEG33 | |
| P52 | | | | SEG34 | |
| P53 | | | | SEG35 | |
| P54 | | | | SEG36 | |
| P55 | | | | SEG37 | |
| P56 | | | | SEG38 | |
| P57 | | | | SEG39 | |
| P60 | 12-1-3 | I/O | Input port | SCLA0/(TI00)/(TO00) | Port 6.<br>3-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Can be set to N-ch open-drain output (6 V tolerance). |
| P61 | | | | SDAA0/(TI01)/(TO01) | |
| P62 | | | | (TI02)/(TO02)/(RTC1HZ) | |
| P70 | 7-5-4 | I/O | Digital input invalid[Note 1] | SEG16/(INTP0) | Port 7.<br>8-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Can be set to LCD output[Note 2]. |
| P71 | | | | SEG17/(INTP1) | |
| P72 | | | | SEG18/(INTP2) | |
| P73 | | | | SEG19/(INTP3) | |
| P74 | | | | SEG20/(INTP4) | |
| P75 | | | | SEG21/(INTP5) | |
| P76 | | | | SEG22/(INTP6) | |
| P77 | | | | SEG23/(INTP7) | |
| P80 | 7-5-10 | I/O | Digital input invalid[Note 1] | SEG12/(SCL10) | Port 8.<br>6-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Input of P81 can be set to TTL input buffer.<br>Output of P80 to P82 can be set to N-ch open-drain output ($V_{DD}$ tolerance).<br>Can be set to LCD output[Note 2]. |
| P81 | 8-5-10 | | | SEG13/(RxD1)/(SDA10) | |
| P82 | 7-5-10 | | | SEG14/(TxD1) | |
| P83 | 7-5-4 | | | SEG15 | |
| P84 | | | | SEG40 | |
| P85 | | | | SEG41 | |

**Notes 1.** "Digital input invalid" refers to the state in which all the digital outputs, digital inputs, and LCD outputs are disabled.

**2.** Digital or LCD for each pin can be selected with the port mode register x (PMx) and the LCD port function register x (PFSEGx) (can be set in 1-bit unit).

**Remark** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR)**.

(3/3)

<R>

| Function Name | Pin Type | I/O | After Reset Release | Alternate Function | Function |
|---|---|---|---|---|---|
| P121 | 2-2-1 | Input | Input port | X1 | Port 12.<br>3-bit I/O port and 4-bit input only port.<br>For only P125 to P127, input/output can be specified in 1-bit units.<br>For only P125 to P127, use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>P125 to P127 can be set to LCD output[Note 2]. |
| P122 | | | | X2/EXCLK | |
| P123 | | | | XT1 | |
| P124 | | | | XT2/EXCLKS | |
| P125 | 7-5-6 | I/O | Digital input invalid[Note 1] | $V_{L3}$/INTP1/(TI05)/(TO05) | |
| P126 | 7-5-5 | | | CAPL/(TI04)/(TO04) | |
| P127 | | | | CAPH/(TI03)/(TO03) | |
| P130 | 1-1-4 | Output | Output port | RTC1HZ | Port 13.<br>1-bit output port and 1-bit input only port. |
| P137 | 2-1-2 | Input | Input port | INTP0 | |
| $\overline{\text{RESET}}$ | 2-1-1 | Input | – | – | Input only pin for external reset.<br>When external reset is not used, connect this pin to $V_{DD}$ directly or via a resistor. |

**Notes 1.** “Digital input invalid” refers to the state in which all the digital outputs, digital inputs, and LCD outputs are disabled.

**2.** Digital or LCD for each pin can be selected with the port mode register x (PMx) and the LCD port function register x (PFSEGx) (can be set in 1-bit unit).

**Remark** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR)**.

## 2.2 Functions Other than Port Pins

### 2.2.1 With functions for each product

(1/2)

| Function Name | 100-pin | 80-pin | Function Name | 100-pin | 80-pin | Function Name | 100-pin | 80-pin |
|---|---|---|---|---|---|---|---|---|
| ANI0 | √ | √ | TxD2 | √ | √ | XT2 | √ | √ |
| ANI1 | √ | √ | SCK00 | √ | √ | EXCLKS | √ | √ |
| ANI2 | √ | √ | SI00 | √ | √ | $V_{DD}$ | √ | √ |
| ANI3 | √ | √ | SO00 | √ | √ | $EV_{DD0}$ | √ | – |
| ANI4 | √ | – | SCL00 | √ | √ | $EV_{DD1}$ | √ | – |
| ANI5 | √ | – | SCL10 | √ | √ | VBAT | √ | √ |
| ANIN0 | √ | √ | SDA00 | √ | √ | $AV_{REFP}$ | √ | √ |
| ANIN1 | √ | √ | SDA10 | √ | √ | $AV_{REFM}$ | √ | √ |
| ANIN2 | √ | √ | SDAA0 | √ | √ | $V_{SS}$ | √ | √ |
| ANIN3 | √ | – | SCLA0 | √ | √ | $EV_{SS0}$ | √ | – |
| ANIP0 | √ | √ | IrRxD | √ | √ | $EV_{SS1}$ | √ | – |
| ANIP1 | √ | √ | IrTxD | √ | √ | AVRT | √ | √ |
| ANIP2 | √ | √ | TI00 | √ | √ | AVCM | √ | √ |
| ANIP3 | √ | – | TI01 | √ | √ | AREGC | √ | √ |
| INTP0 | √ | √ | TI02 | √ | √ | $AV_{DD}$ | √ | √ |
| INTP1 | √ | √ | TI03 | √ | √ | $AV_{SS}$ | √ | √ |
| INTP2 | √ | √ | TI04 | √ | √ | TOOLRxD | √ | √ |
| INTP3 | √ | √ | TI05 | √ | √ | TOOLTxD | √ | √ |
| INTP4 | √ | √ | TI06 | √ | √ | TOOL0 | √ | √ |
| INTP5 | √ | √ | TI07 | √ | √ | COM0 | √ | √ |
| INTP6 | √ | √ | TO00 | √ | √ | COM1 | √ | √ |
| INTP7 | √ | √ | TO01 | √ | √ | COM2 | √ | √ |
| IVCMP0 | √ | √ | TO02 | √ | √ | COM3 | √ | √ |
| IVCMP1 | √ | √ | TO03 | √ | √ | COM4 | √ | √ |
| IVREF0 | √ | √ | TO04 | √ | √ | COM5 | √ | √ |
| IVREF1 | √ | √ | TO05 | √ | √ | COM6 | √ | √ |
| VCOUT0 | √ | √ | TO06 | √ | √ | COM7 | √ | √ |
| VCOUT1 | √ | √ | TO07 | √ | √ | SEG0 | √ | √ |
| PCLBUZ0 | √ | √ | $V_{L1}$ | √ | √ | SEG1 | √ | √ |
| PCLBUZ1 | √ | √ | $V_{L2}$ | √ | √ | SEG2 | √ | √ |
| RTC1HZ | √ | √ | $V_{L3}$ | √ | √ | SEG3 | √ | √ |
| REGC | √ | √ | $V_{L4}$ | √ | √ | SEG4 | √ | √ |
| $\overline{\text{RESET}}$ | √ | √ | CAPH | √ | √ | SEG5 | √ | √ |
| RxD0 | √ | √ | CAPL | √ | √ | SEG6 | √ | √ |
| RxD1 | √ | √ | X1 | √ | √ | SEG7 | √ | √ |
| RxD2 | √ | √ | X2 | √ | √ | SEG8 | √ | √ |
| TxD0 | √ | √ | EXCLK | √ | √ | SEG9 | √ | √ |
| TxD1 | √ | √ | XT1 | √ | √ | SEG10 | √ | √ |

(2/2)

| Function Name | 100-pin | 80-pin | Function Name | 100-pin | 80-pin | Function Name | 100-pin | 80-pin |
|---|---|---|---|---|---|---|---|---|
| SEG11 | √ | √ | SEG22 | √ | √ | SEG33 | √ | √ |
| SEG12 | √ | √ | SEG23 | √ | √ | SEG34 | √ | √ |
| SEG13 | √ | √ | SEG24 | √ | √ | SEG35 | √ | √ |
| SEG14 | √ | √ | SEG25 | √ | √ | SEG36 | √ | √ |
| SEG15 | √ | √ | SEG26 | √ | √ | SEG37 | √ | √ |
| SEG16 | √ | √ | SEG27 | √ | √ | SEG38 | √ | – |
| SEG17 | √ | √ | SEG28 | √ | – | SEG39 | √ | – |
| SEG18 | √ | √ | SEG29 | √ | – | SEG40 | √ | – |
| SEG19 | √ | √ | SEG30 | √ | – | SEG41 | √ | – |
| SEG20 | √ | √ | SEG31 | √ | – | | | |
| SEG21 | √ | √ | SEG32 | √ | √ | | | |

### 2.2.2 Description of Functions

(1/2)

| Function Name | I/O | Function |
|---|---|---|
| ANI0 to ANI5 | Input | A/D converter analog input (see **Figure 14-44 Analog Input Pin Connection**) |
| ANIN0 to ANIN3 | Input | 24-bit ΔΣ-type A/D converter analog input.<br>These are the negative input pins. |
| ANIP0 to ANIP3 | Input | 24-bit ΔΣ-type A/D converter analog input.<br>These are the positive input pins. |
| INTP0 to INTP7 | Input | External interrupt request input<br>Specified the valid edge: Rising edge, falling edge, or both rising and falling edges<br>INTP0 is a pin that operates at an internal $V_{DD}$. When using a battery backup function, the input threshold value is adjusted to the selected power supply ($V_{DD}$ or VBAT).<br>Maximum allowed input voltage is 5.5 V. If unused, pull up to VBAT or $V_{DD}$, whichever is higher. |
| IVCMP0, IVCMP1 | Input | Comparator analog voltage input |
| IVREF0, IVREF1 | Input | Comparator reference voltage input |
| VCOUT0, VCOUT1 | Output | Comparator output |
| PCLBUZ0, PCLBUZ1 | Output | Clock output/buzzer output |
| REGC | – | Pin for connecting regulator output stabilization capacitance for internal operation.<br>Connect this pin to $V_{SS}$ via a capacitor (0.47 to 1 μF).<br>Also, use a capacitor with good characteristics, since it is used to stabilize internal voltage. |
| RTC1HZ | Output | Real-time clock correction clock (1 Hz) output |
| $\overline{\text{RESET}}$ | Input | This is the active-low system reset input pin.<br>When the external reset pin is not used, connect this pin directly or via a resistor to $V_{DD}$.<br>$\overline{\text{RESET}}$ is a pin that operates at an internal $V_{DD}$. When using a battery backup function, the input threshold value is adjusted to the selected power supply pin ($V_{DD}$ or VBAT pin).<br>Maximum allowed input voltage is 5.5 V. If unused, pull up to VBAT or $V_{DD}$, whichever is higher. |
| RxD0 to RxD2 | Input | Serial data input pins of serial interface UART0 to UART2 |
| TxD0 to TxD2 | Output | Serial data output pins of serial interface UART0 to UART2 |
| SCK00 | I/O | Serial clock I/O pin of serial interface CSI00 |
| SI00 | Input | Serial data input pin of serial interface CSI00 |
| SO00 | Output | Serial data output pin of serial interface CSI00 |
| IrRxD | Input | Receive data for IrDA |
| IrTxD | Output | Transmit data for IrDA |
| SCL00, SCL10 | Output | Serial clock output pins of serial interface IIC00 and IIC10 |
| SDA00, SDA10 | I/O | Serial data I/O pins of serial interface IIC00 and IIC10 |
| SCLA0 | I/O | Serial clock I/O pins of serial interface IICA0 |
| SDAA0 | I/O | Serial data I/O pins of serial interface IICA0 |
| TI00 to TI07 | Input | The pins for inputting an external count clock/capture trigger to 16-bit timers 00 to 07 |
| TO00 to TO07 | Output | Timer output pins of 16-bit timers 00 to 07 |
| $V_{L1}$ to $V_{L4}$ | – | LCD drive voltage |
| CAPH, CAPL | – | Connecting a capacitor for LCD controller/driver |

(2/2)

| Function Name | I/O | Function |
|---|---|---|
| X1, X2 | – | If an external 24-bit ΔΣ type A/D converter is used for external clock input, a 12 MHz oscillator must be connected. |
| EXCLK | Input | External clock input for main system clock |
| XT1, XT2 | – | Resonator connection for subsystem clock |
| EXCLKS | Input | External clock input for subsystem clock |
| $V_{DD}$ | – | <80-pin ><br>Positive power supply for all pins<br><100-pin><br>Positive power supply for P20 to P25, P121 to P124, P137 and other than ports |
| $EV_{DD1}$ | – | Positive power supply for ports (other than P20 to P25, P121 to P124, P137) |
| VBAT | – | Power supply for battery backup |
| $AV_{REFP}$ | Input | A/D converter reference potential (+ side) input |
| $AV_{REFM}$ | Input | A/D converter reference potential (– side) input |
| $V_{SS}$ | – | <80-pin ><br>Ground potential for all pins<br><100-pin ><br>Ground potential for P20 to P25, P121 to P124, P137 and other than ports |
| $EV_{SS1}$ | – | Ground potential for ports (other than P20 to P25, P121 to P124, P137) |
| AVRT | – | Reference potential for ΔΣ ADC |
| AVCM | – | Control for ΔΣ ADC |
| AREGC | – | Regulator capacitance for ΔΣ ADC |
| $AV_{DD}$ | – | Power supply for ΔΣ ADC |
| $AV_{SS}$ | – | Ground for ΔΣ ADC |
| TOOLRxD | Input | UART reception pin for the external device connection used during flash memory programming |
| 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/debugger |
| COM0 to COM7 | Output | LCD controller/driver common signal outputs |
| SEG0 to SEG41 | Output | LCD controller/driver segment signal outputs |

**Caution After reset release, the relationships between P40/TOOL0 and the operating mode are as follows.**

**Table 2-2. Relationships Between P40/TOOL0 and Operation Mode After Reset Release**

| P40/TOOL0 | Operating Mode |
|---|---|
| $V_{DD}$ | Normal operation mode |
| 0 V | Flash memory programming mode |

**For details, see 33.4 Serial 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 $V_{SS}$, $EV_{DD0}$ to $EV_{SS0}$, $EV_{DD1}$ to $EV_{SS1}$ lines.

## 2.3 Connection of Unused Pins

Table 2-3 shows the connections of unused pins.

**Remark** The pins mounted depend on the product. See **1.3 Pin Configuration (Top View)** and **2.1 Port Function List.**

**Table 2-3. Connection of Unused Pins (1/2)**

| Pin Name | I/O | Recommended Connection of Unused Pins |
|---|---|---|
| P00 to P07 | I/O | Input: Independently connect to $EV_{DD0}$, $EV_{DD1}$ or $EV_{SS0}$, $EV_{SS1}$ via a resistor.<br>Output: Leave open. |
| P10 to P17 | | <When setting to port I/O><br>Input: Independently connect to $EV_{DD0}$, $EV_{DD1}$ or $EV_{SS0}$, $EV_{SS1}$ via a resistor.<br>Output: Leave open.<br><When setting to segment output><br>Leave open. |
| P20 to P25 | | Input: Independently connect to $V_{DD}$ or $V_{SS}$ via a resistor. In addition, individually connect to $V_{SS}$ via a resistor when using a battery backup function.<br>Output: Leave open. |
| P30 to P37 | | <When setting to port I/O><br>Input: Independently connect to $EV_{DD0}$, $EV_{DD1}$ or $EV_{SS0}$, $EV_{SS1}$ via a resistor.<br>Output: Leave open.<br><When setting to segment output><br>Leave open. |
| P40/TOOL0 | | Input: Independently connect to $EV_{DD}$ via a resistor or leave open.<br>Output: Leave open. |
| P41 to P44 | | Input: Independently connect to $EV_{DD0}$, $EV_{DD1}$ or $EV_{SS0}$, $EV_{SS1}$ via a resistor.<br>Output: Leave open. |
| P50 to P57 | | <When setting to port I/O><br>Input: Independently connect to $EV_{DD0}$, $EV_{DD1}$ or $EV_{SS0}$, $EV_{SS1}$ via a resistor.<br>Output: Leave open.<br><When setting to segment output><br>Leave open. |
| P60 to P62 | | Input: Independently connect to $EV_{DD0}$, $EV_{DD1}$ or $EV_{SS0}$, $EV_{SS1}$ via a resistor.<br>Output: Set the port's output latch to 0 and leave the pin open, or set the port's output latch to 1 and independently connect the pin to $EV_{DD0}$, $EV_{DD1}$ or $EV_{SS0}$, $EV_{SS1}$ via a resistor. |

**Remark** For the products that do not have an $EV_{DD0}$, $EV_{DD1}$, $EV_{SS0}$, or $EV_{SS1}$ pin, replace $EV_{DD0}$ and $EV_{DD1}$ with $V_{DD}$, and replace $EV_{SS0}$ and $EV_{SS1}$ with $V_{SS}$.

**Table 2-3. Connection of Unused Pins (2/2)**

| Pin Name | I/O | Recommended Connection of Unused Pins |
|---|---|---|
| P70 to P77<br>P80 to P85 | I/O | <When setting to port I/O><br>Input: Independently connect to $EV_{DD0}$, $EV_{DD1}$ or $EV_{SS0}$, $EV_{SS1}$ via a resistor.<br>Output: Leave open.<br><When setting to segment output><br>Leave open. |
| P121 to P124 | Input | Independently connect to $V_{DD}$ or $V_{SS}$ via a resistor. In addition, individually connect to $V_{SS}$ via a resistor when using a battery backup function. |
| P125 to P127 | I/O | Input: Independently connect to $EV_{DD0}$, $EV_{DD1}$ or $EV_{SS0}$, $EV_{SS1}$ via a resistor.<br>Output: Leave open. |
| P130 | Output | Leave open. |
| P137 | Input | Independently connect to $V_{DD}$ or $V_{SS}$ via a resistor. In addition, individually connect to $V_{SS}$ via a resistor when using a battery backup function. |
| $\overline{\text{RESET}}$ | | When using a battery backup function, connect directly or via resistor to the selected power supply (VBAT or $V_{DD}$ pin). |
| REGC | – | Connect to $V_{SS}$ via capacitor (0.47 to 1 μF). |
| COM0 to COM7 | Output | Leave open. |
| ANIP0 to ANIP3<br>ANIN0 to ANIN3 | Input | Leave open. |
| $V_{L1}$, $V_{L2}$, $V_{L4}$ | – | Leave open. |
| VBAT | – | Connect directly to $V_{SS}$. In addition, if the VBAT pin is not used, be sure to set the VBATEN bit to 0 with software. |
| AVRT, AVCM | – | Connect to $AV_{SS}$ via capacitor (0.47 μF). |
| $AV_{DD}$ | – | Make $AV_{DD}$ the same potential as $V_{DD}$ |
| $AV_{SS}$ | – | Make $AV_{SS}$ the same potential as $V_{SS}$ |
| AREGC | – | Connect to $AV_{SS}$ via capacitor (0.47 μF). |

**Remark** For the products that do not have an $EV_{DD0}$, $EV_{DD1}$, $EV_{SS0}$, or $EV_{SS1}$ pin, replace $EV_{DD0}$ and $EV_{DD1}$ with $V_{DD}$, and replace $EV_{SS0}$ and $EV_{SS1}$ with $V_{SS}$.

## 2.4 Block Diagrams of Pins

Figures 2-1 to 2-16 show the block diagrams of the pins described in **2.1.1 80-pin products** and **2.1.2 100-pin products**. For the 80-pin products, replace $EV_{DD1}$ and $EV_{SS1}$ with $V_{DD}$ and $V_{SS}$, respectively.

**Figure 2-1. Pin Block Diagram for Pin Type 1-1-4**

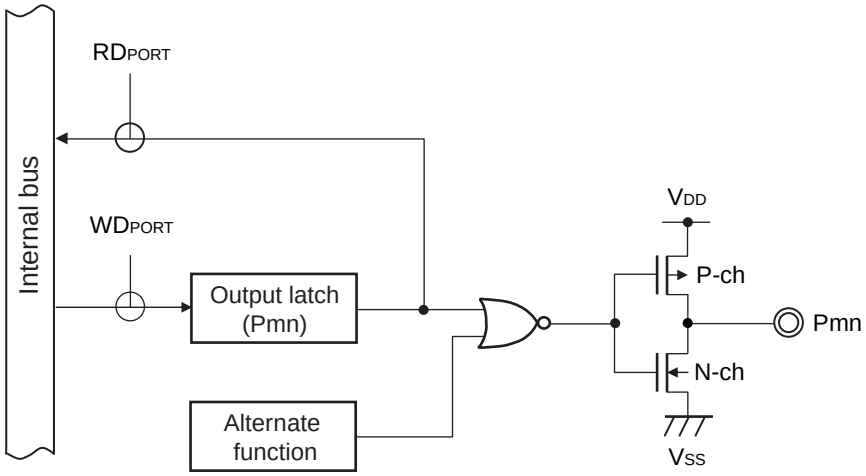
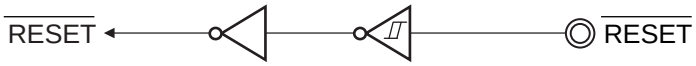
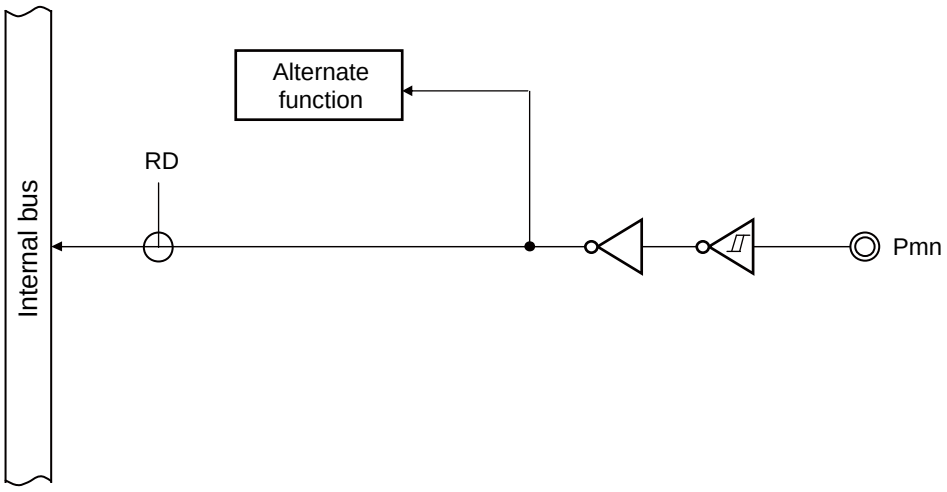



**Figure 2-2. Pin Block Diagram for Pin Type 2-1-1**



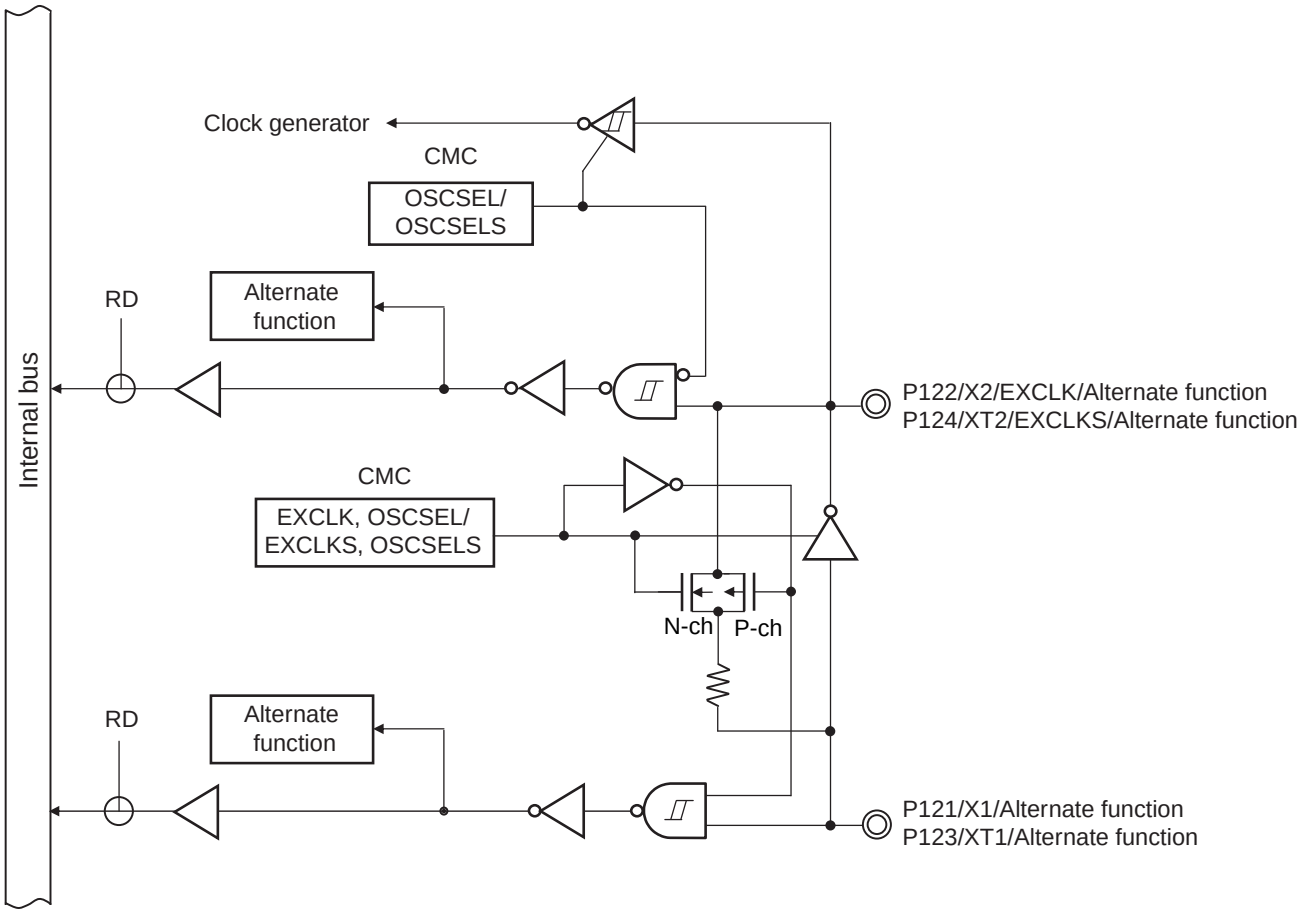

<R>

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




**Remark** For alternate functions, see **2.1 Port Function**.

**Figure 2-4. Pin Block Diagram for Pin Type 2-2-1**




**Remark** For alternate functions, see **2.1 Port Function**.

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

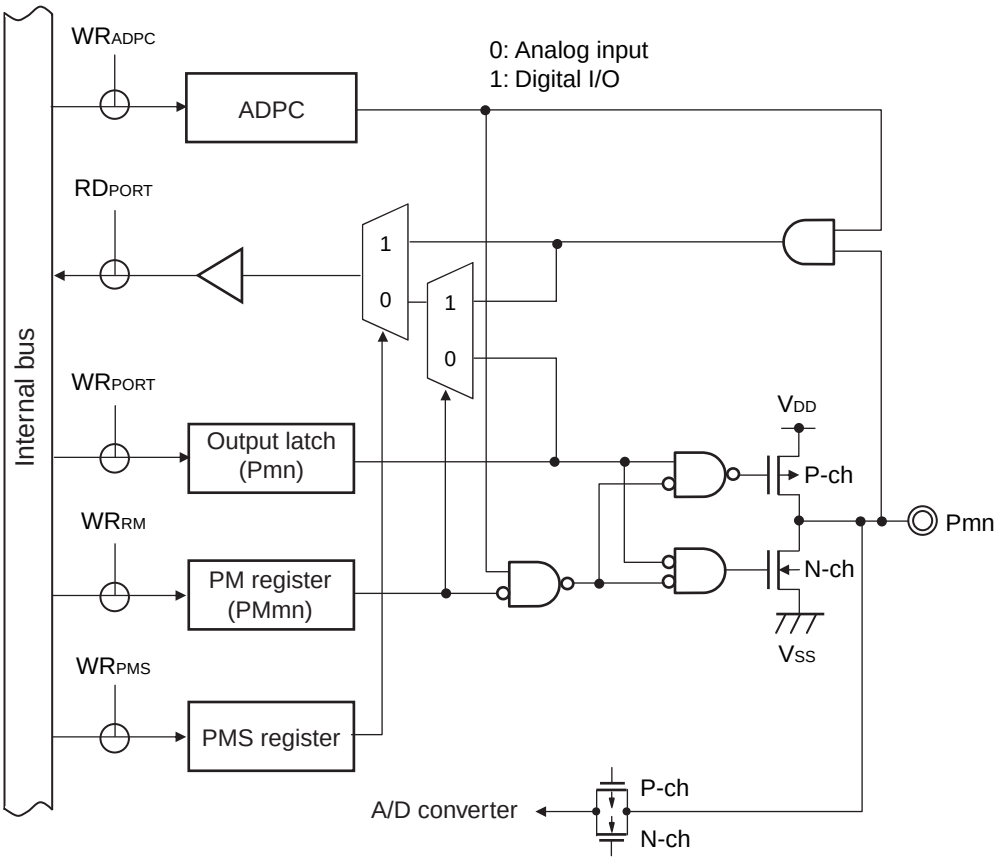


**Figure 2-6. Pin Block Diagram for Pin Type 4-9-2**

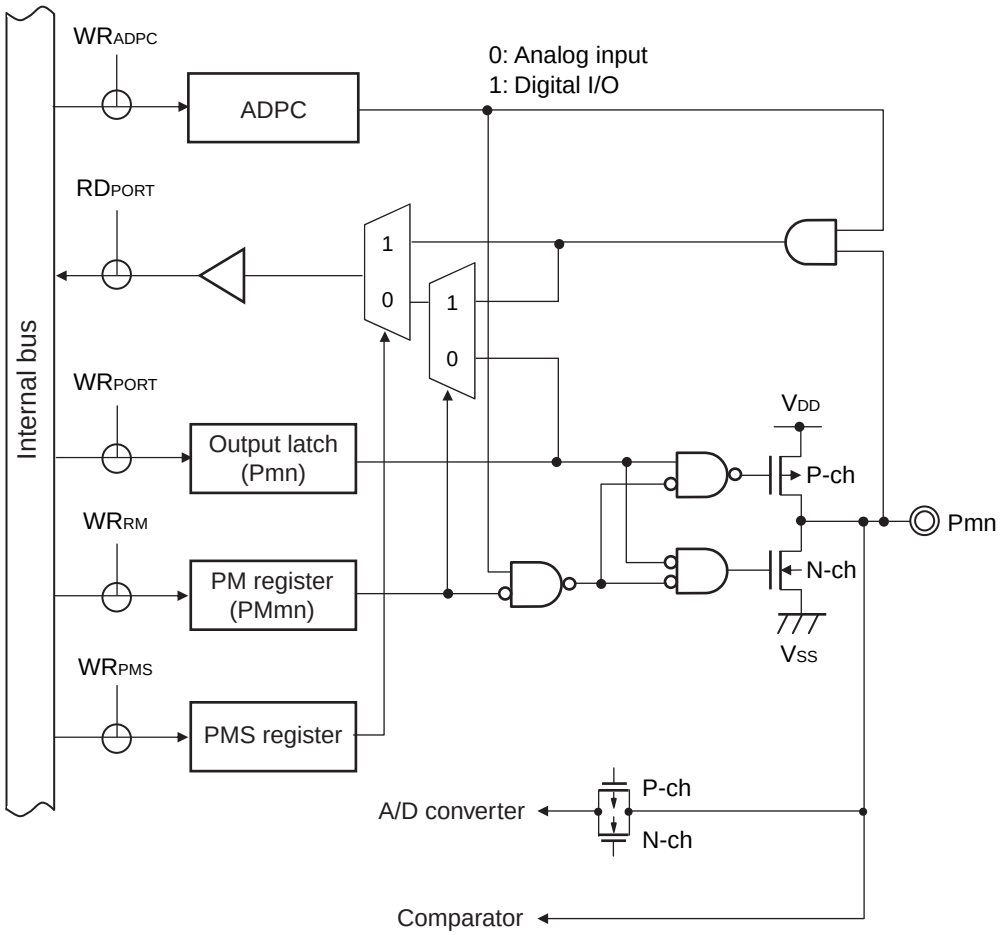
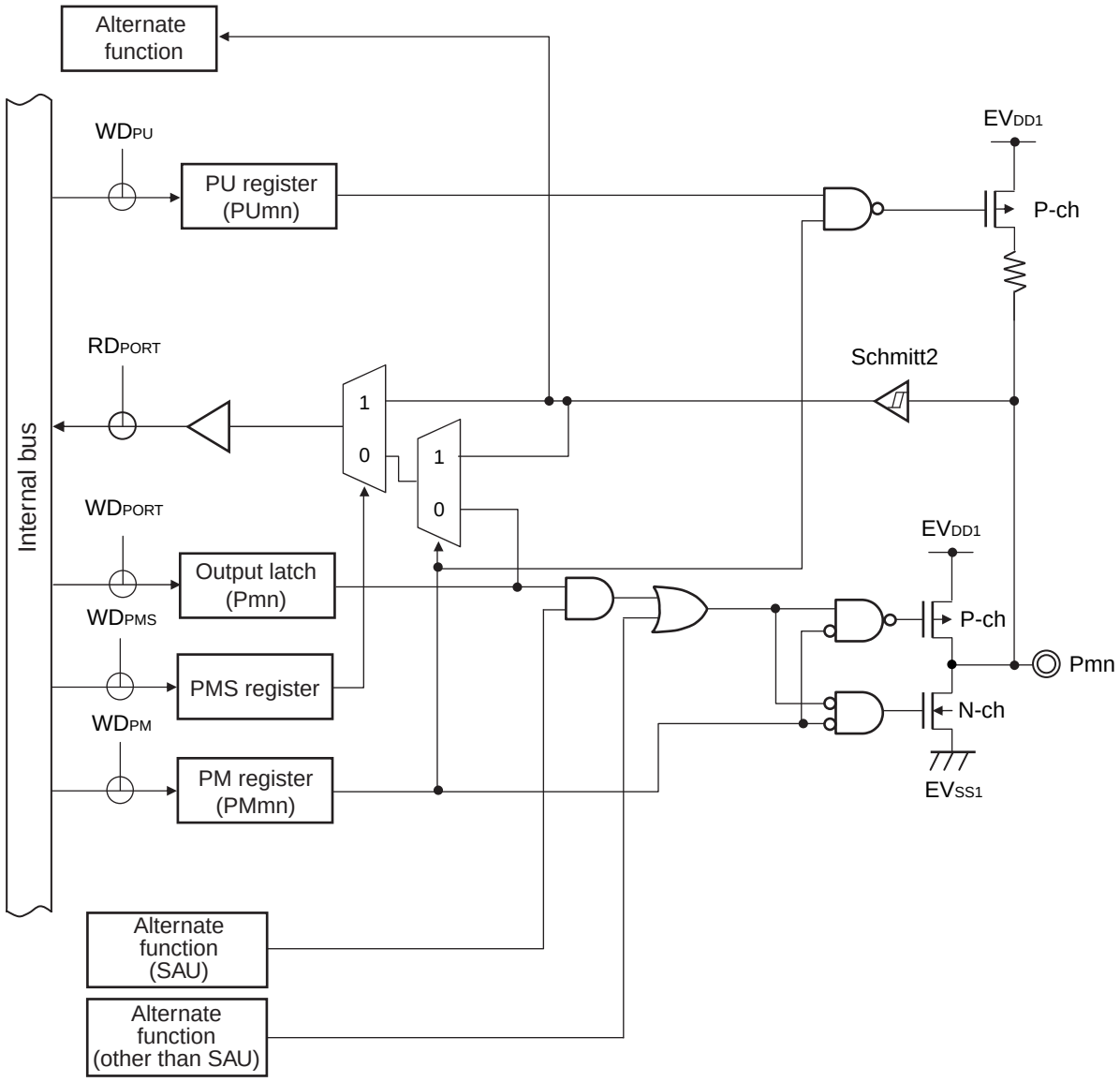


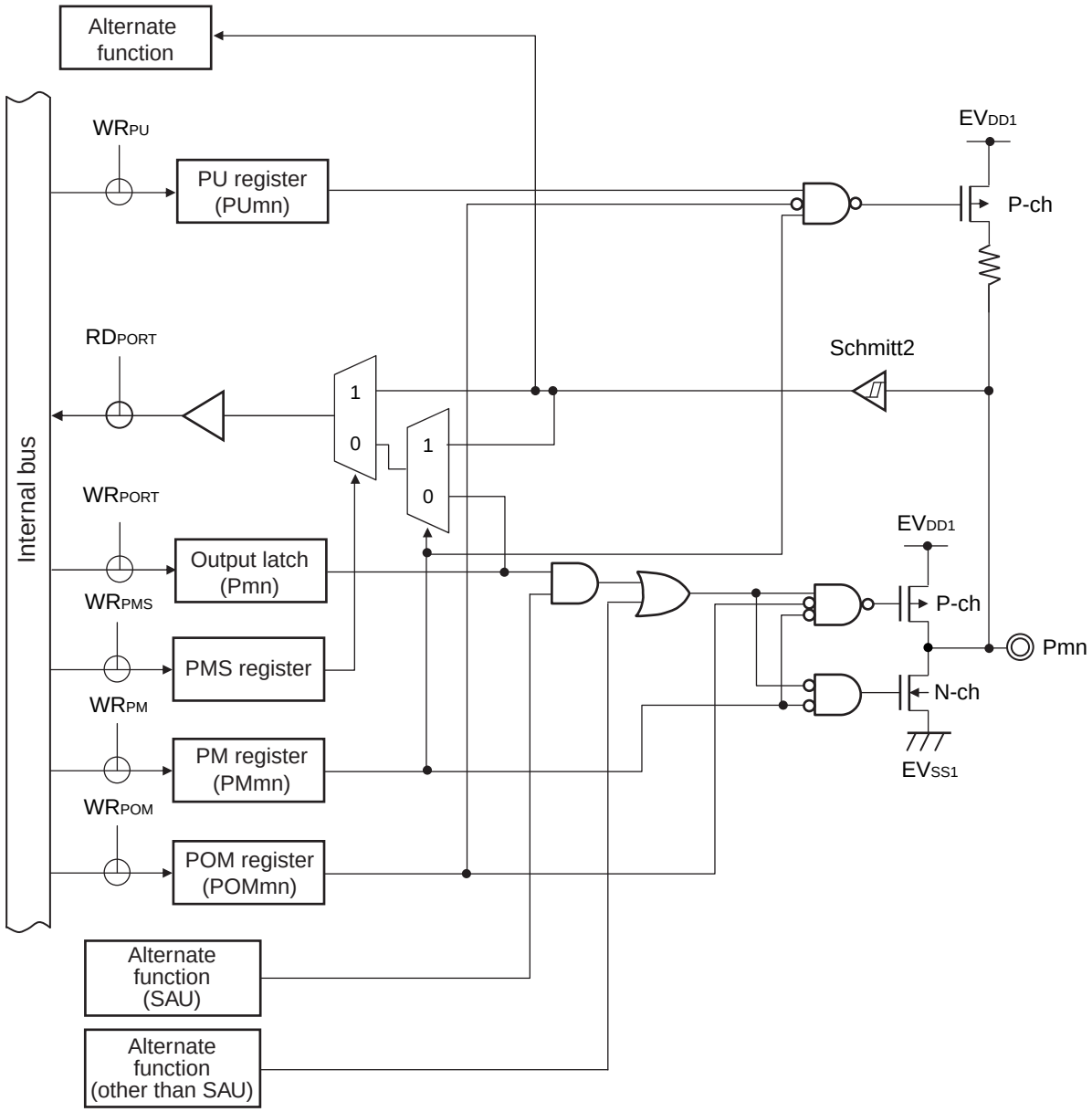
**Figure 2-7. Pin Block Diagram for Pin Type 7-1-3**




**Remarks 1.** For alternate functions, see **2.1 Port Function**.

**2.** SAU: Serial array unit

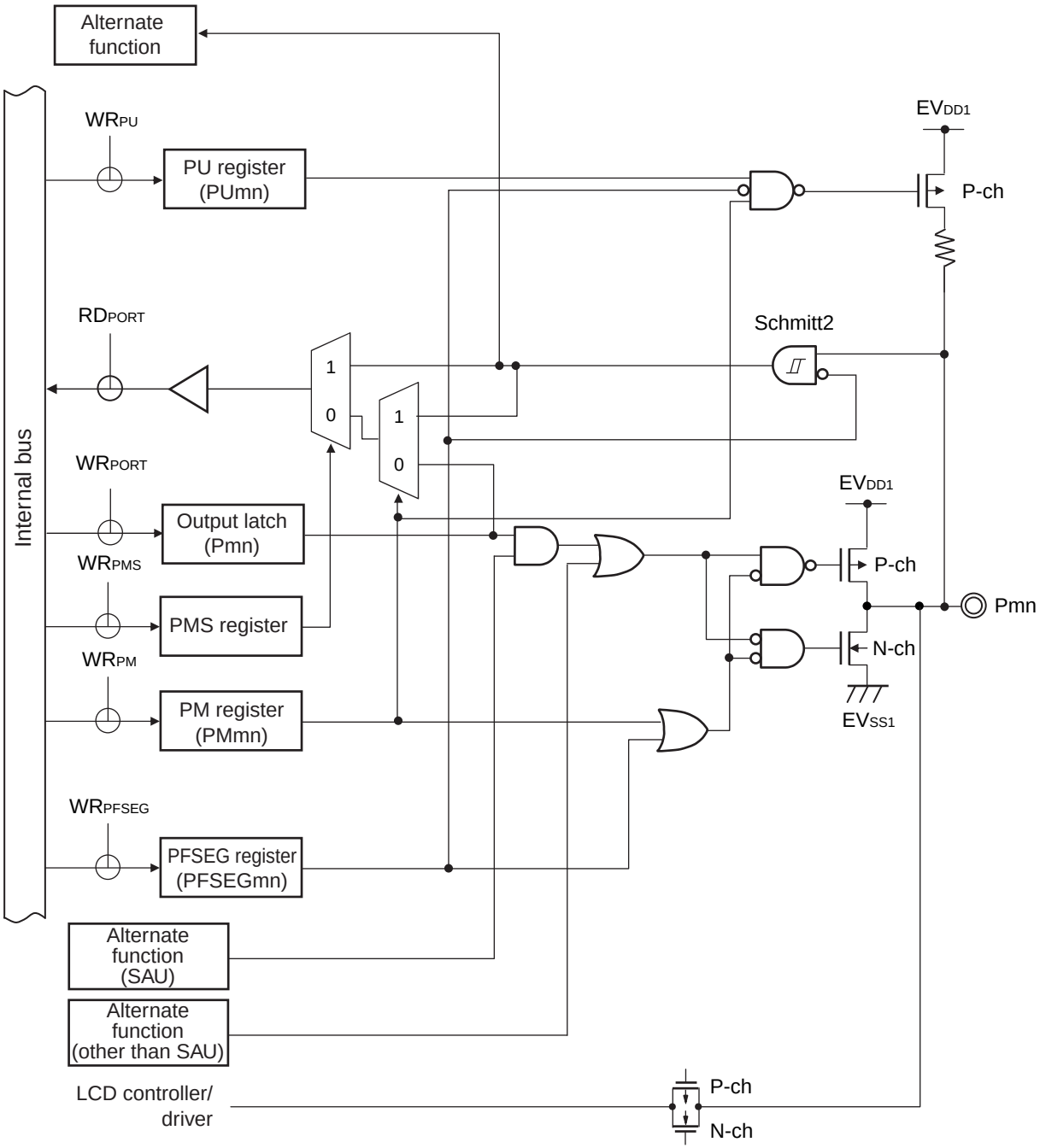
**Figure 2-8. Pin Block Diagram for Pin Type 7-1-4**




**Remarks 1.** For alternate functions, see **2.1 Port Function**.

**2.** SAU: Serial array unit

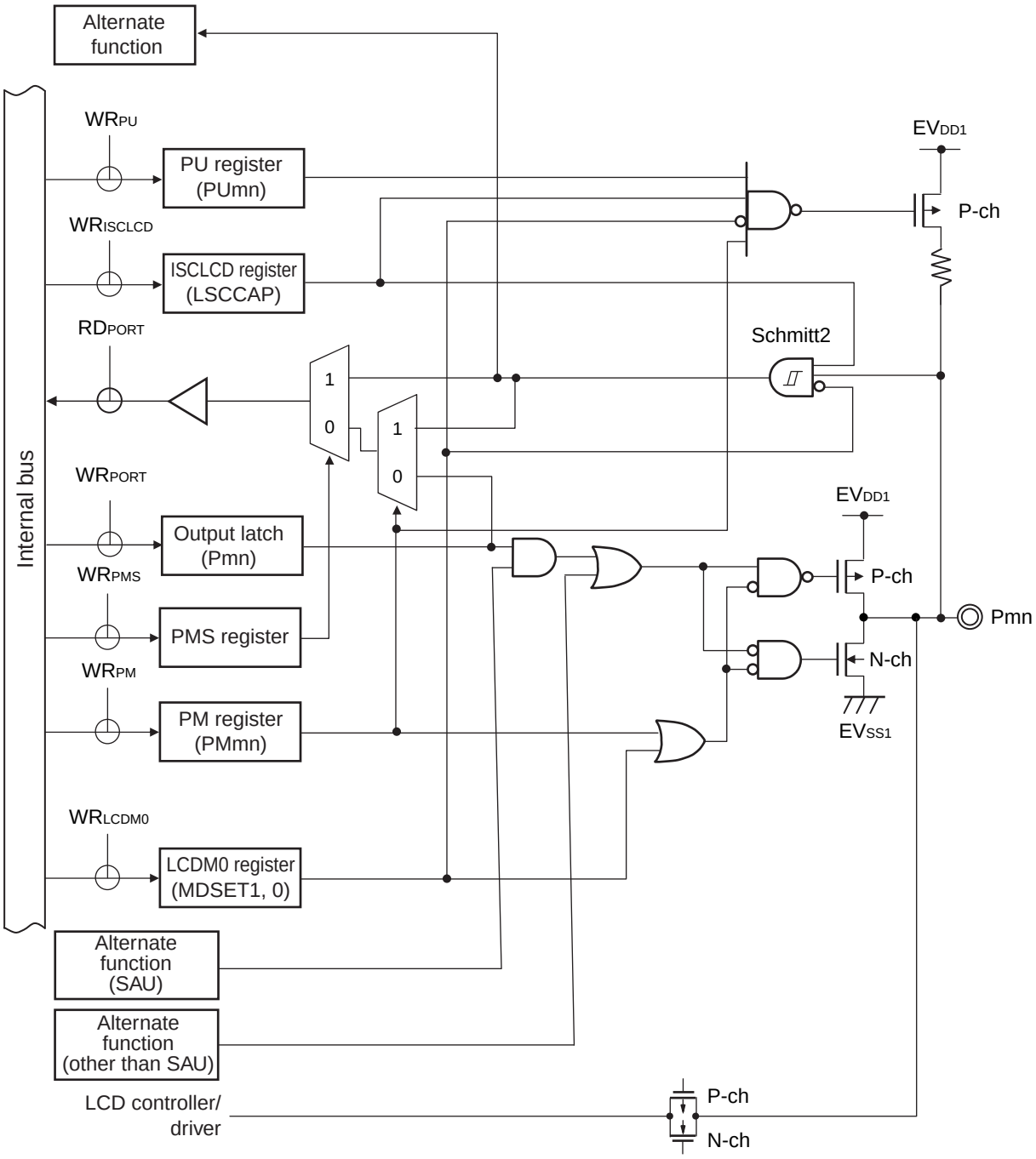
**Figure 2-9. Pin Block Diagram for Pin Type 7-5-4**




**Remarks 1.** For alternate functions, see **2.1 Port Function**.

**2.** SAU: Serial array unit

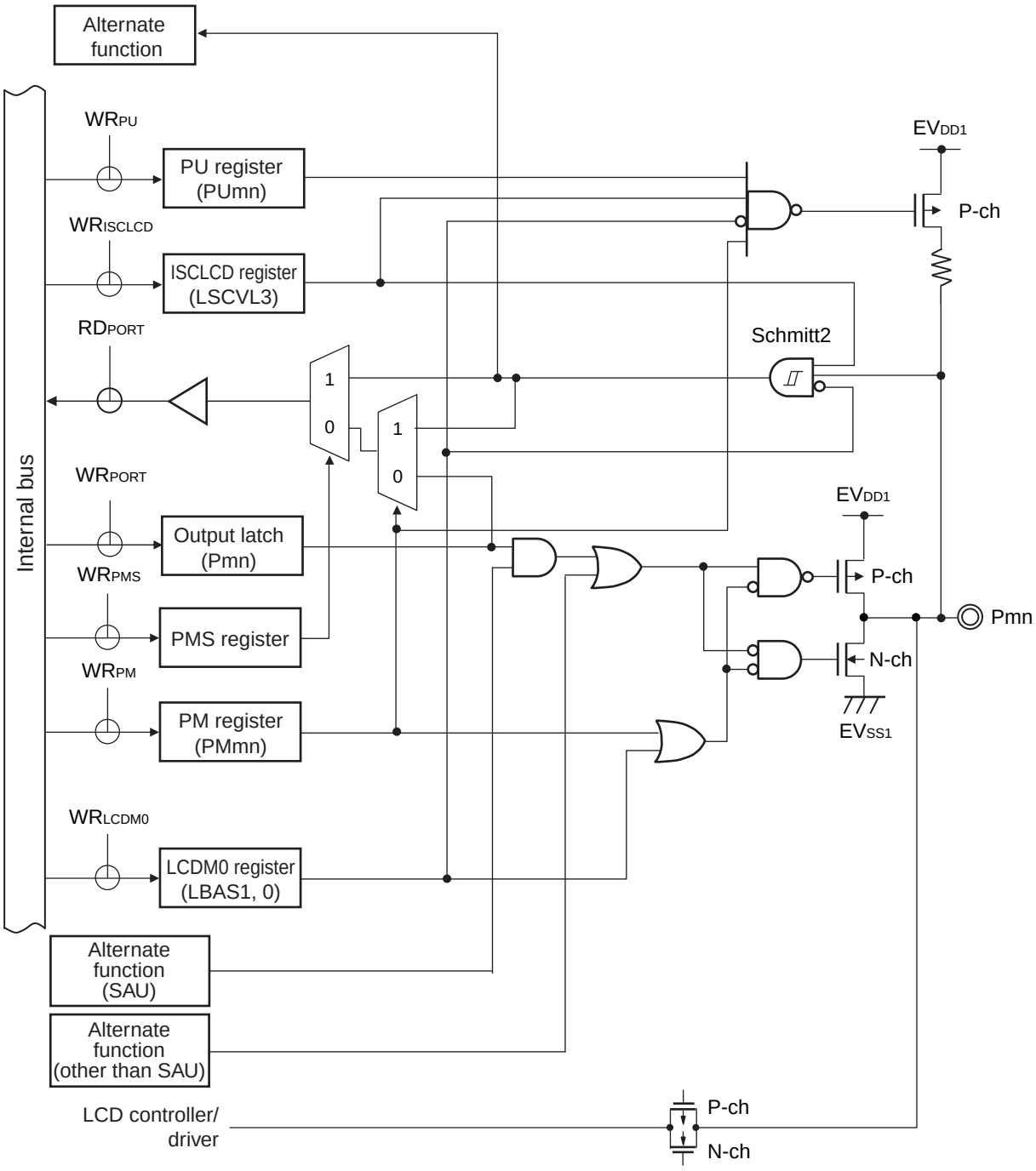
**Figure 2-10. Pin Block Diagram for Pin Type 7-5-5**




**Remarks 1.** For alternate functions, see **2.1 Port Function**.

**2.** SAU: Serial array unit

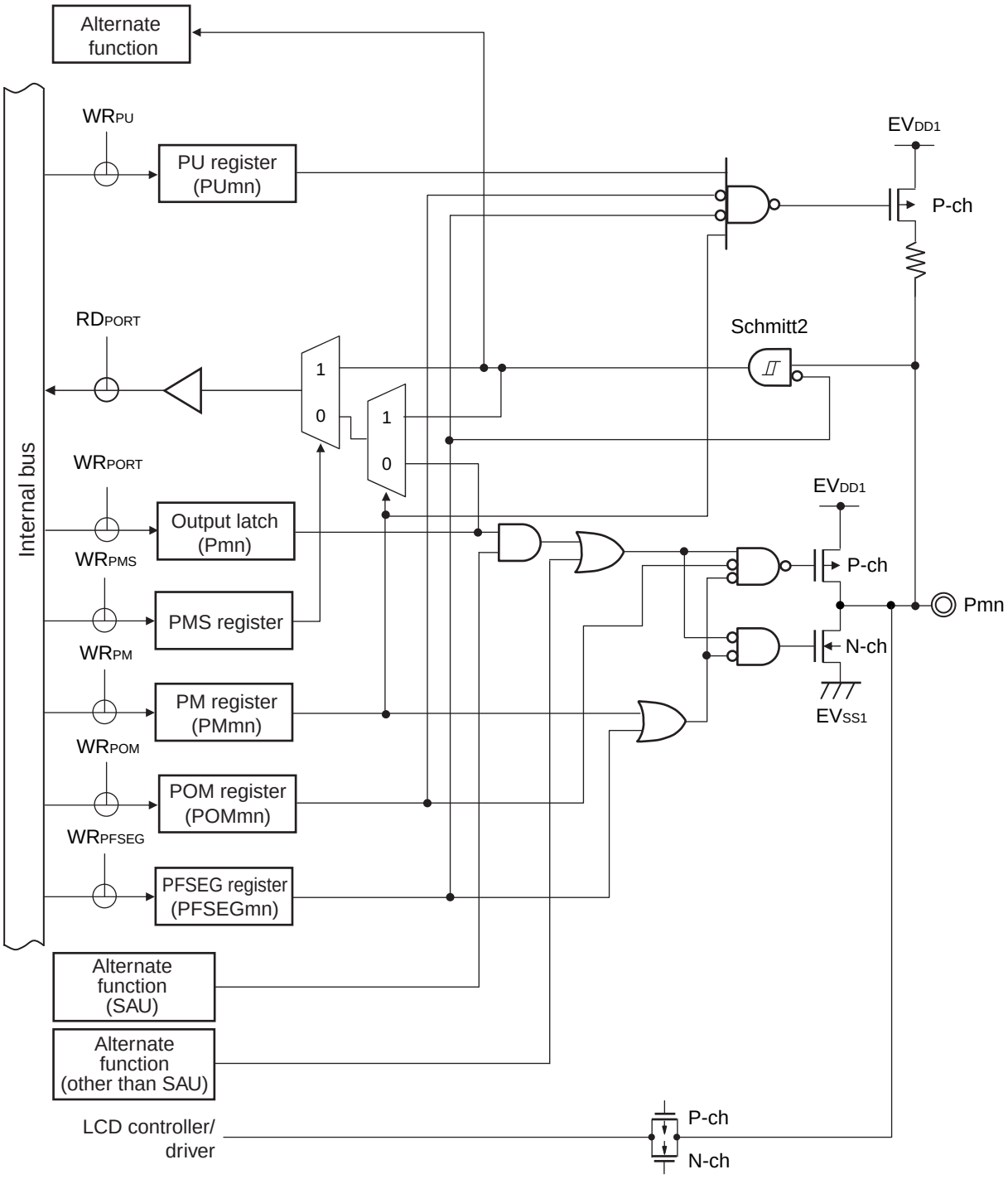
**Figure 2-11. Pin Block Diagram for Pin Type 7-5-6**




**Remarks 1.** For alternate functions, see **2.1 Port Function**.

**2.** SAU: Serial array unit

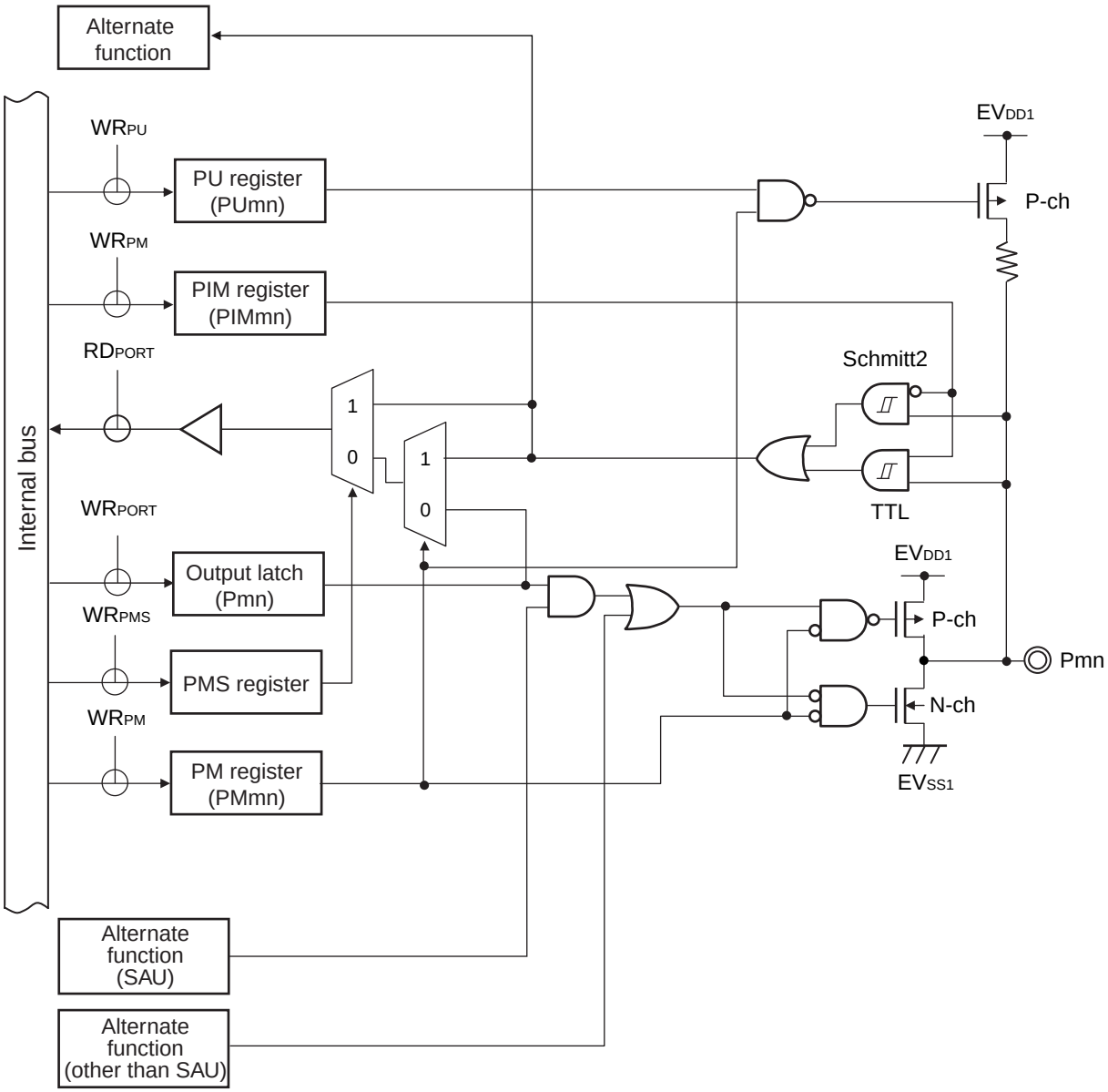
**Figure 2-12. Pin Block Diagram for Pin Type 7-5-10**




**Remarks 1.** For alternate functions, see **2.1 Port Function**.

**2.** SAU: Serial array unit

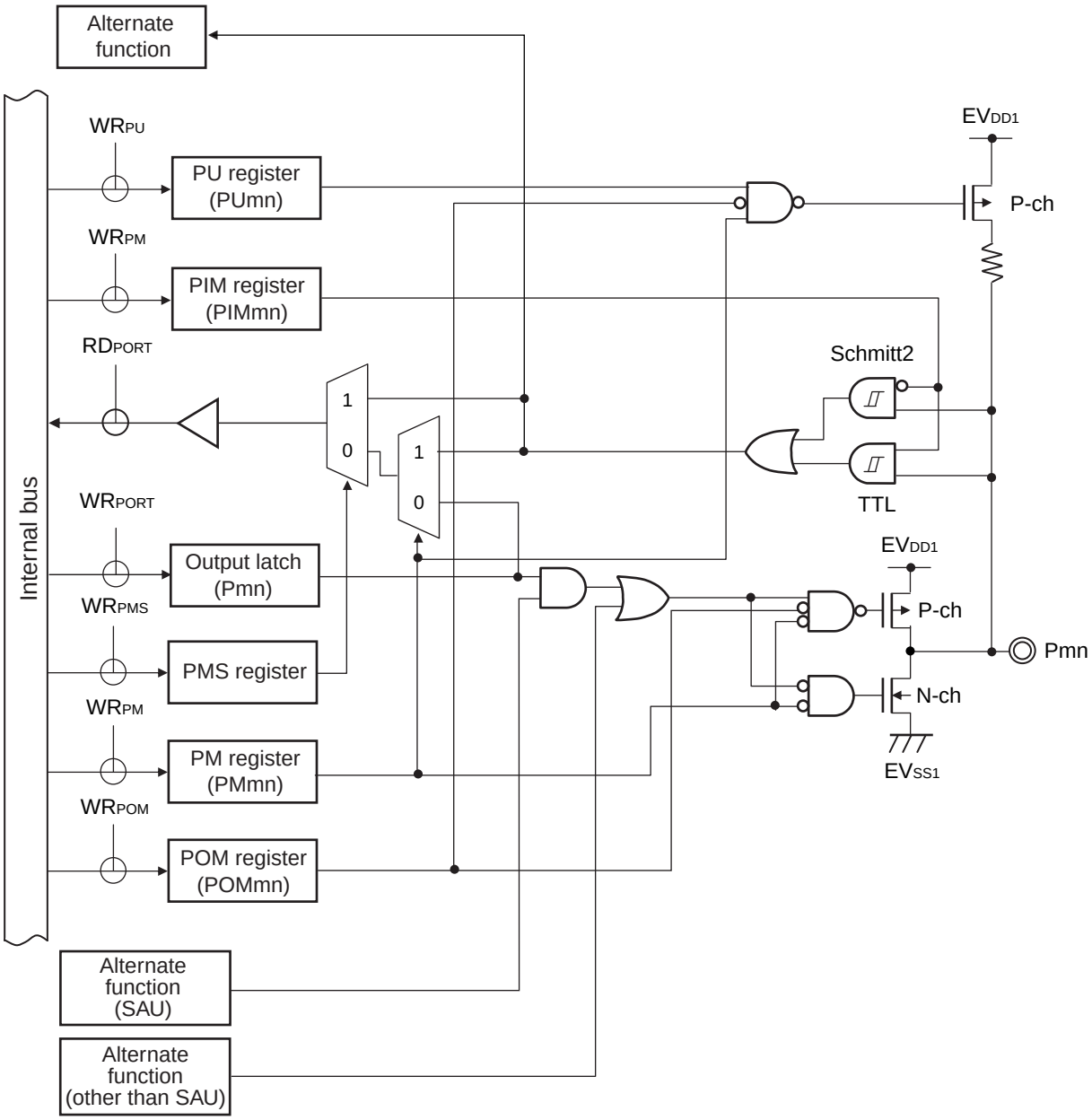
**Figure 2-13. Pin Block Diagram for Pin Type 8-1-3**




**Remarks 1.** For alternate functions, see **2.1 Port Function**.

**2.** SAU: Serial array unit

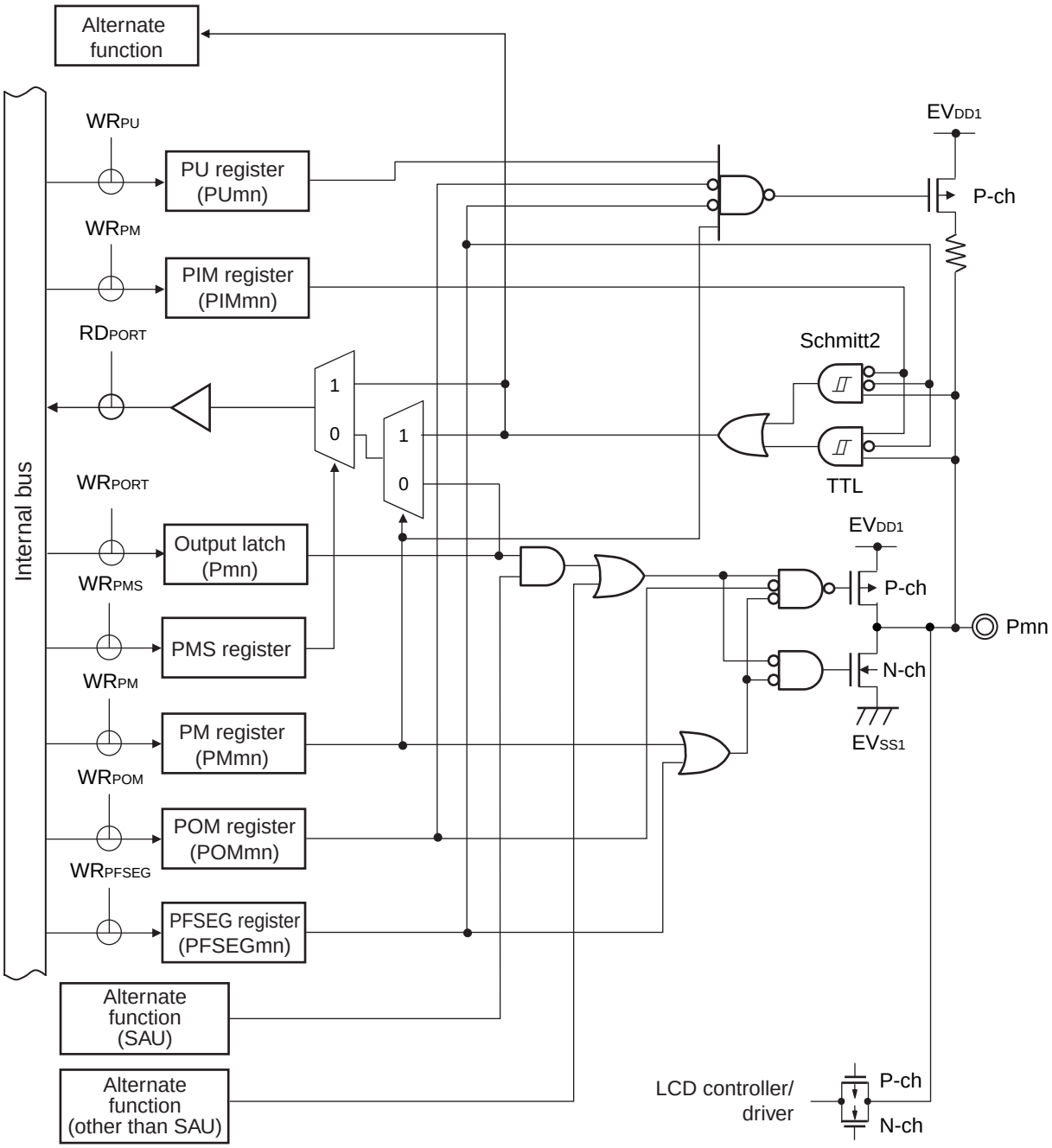
**Figure 2-14. Pin Block Diagram for Pin Type 8-1-4**




**Remarks 1.** For alternate functions, see **2.1 Port Function**.

**2.** SAU: Serial array unit

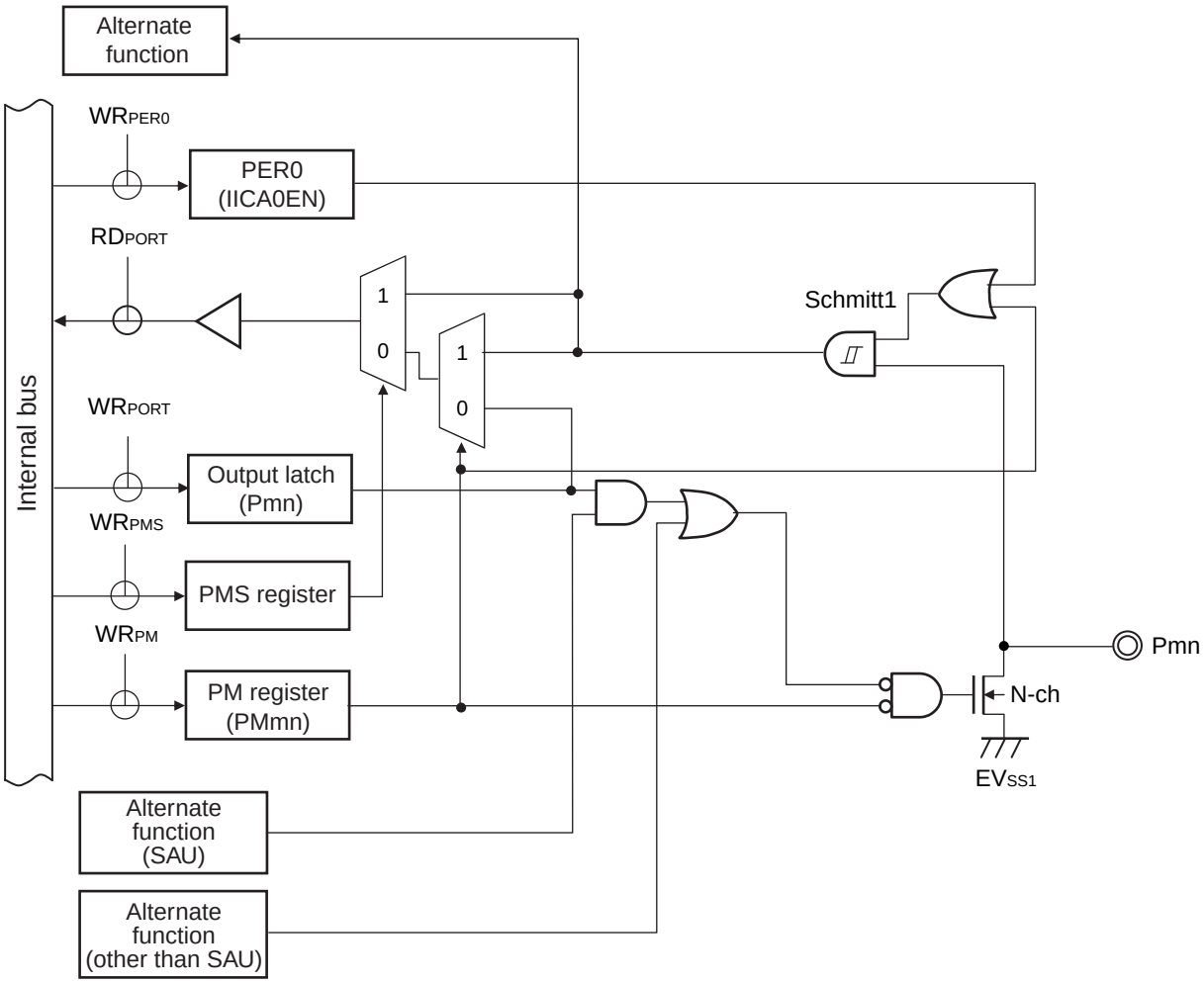
**Figure 2-15. Pin Block Diagram for Pin Type 8-5-10**




**Remarks 1.** For alternate functions, see **2.1 Port Function**.

**2.** SAU: Serial array unit

**Figure 2-16. Pin Block Diagram for Pin Type 12-1-3**

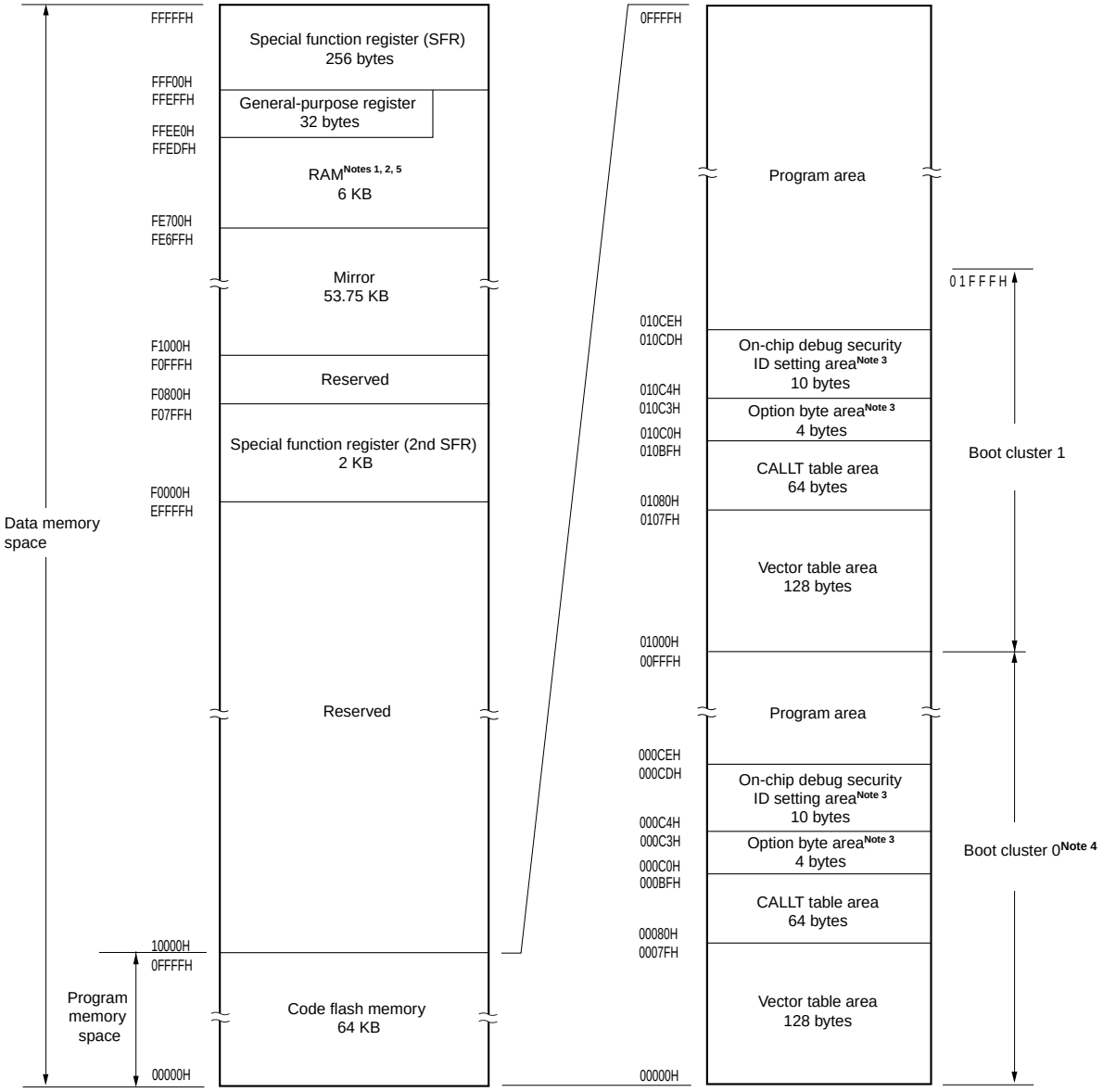


# CHAPTER 3 CPU ARCHITECTURE

## 3.1 Memory Space

Products in the RL78/I1B can access a 1 MB address space. Figures 3-1 and 3-2 show the memory maps.

**Figure 3-1. Memory Map (R5F10MME, R5F10MPE)**




**Notes 1.** Do not allocate RAM addresses which are used as a stack area, a data buffer, a branch destination of vector interrupt processing, and a DTC transfer destination/transfer source to the area FFE20H to FFEDFH when performing self-programming.

**2.** Instructions can be executed from the RAM area excluding the general-purpose register area.

**3.** When boot swap is not used: Set the option bytes to 000C0H to 000C3H, and the on-chip debug security IDs to 000C4H to 000CDH.

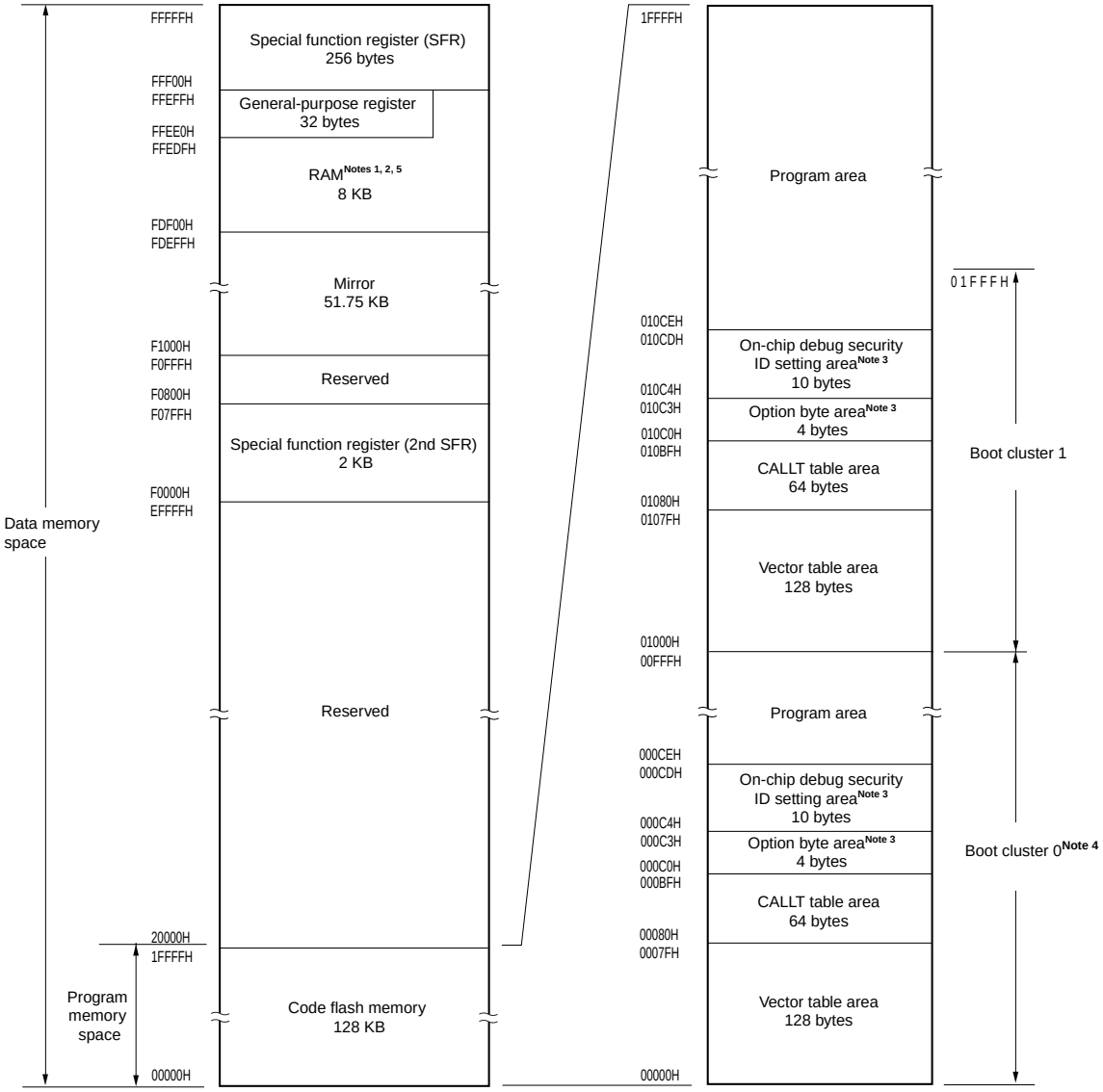
When boot swap is used: Set the option bytes to 000C0H to 000C3H and 010C0H to 010C3H, and the on-chip debug security IDs to 000C4H to 000CDH and 010C4H to 010CDH.

**4.** Writing boot cluster 0 can be prohibited depending on the setting of security (see **33.6 Security Settings**).

**5.** When using the trace function of on-chip debugging, area FE300H to FE6FFH is disabled.

**Caution When executing instructions from the RAM area 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. Reset signal generation sets RAM parity error resets to enabled (RPERDIS = 0). For details, see 30.3.3 RAM parity error detection function.**

**Figure 3-2. Memory Map (R5F10MMG, R5F10MPG)**




<R> **Notes 1.** Do not allocate the stack area, data buffers for use by the flash library, arguments of library functions, branch destinations in the processing of vectored interrupts, or destinations or sources for DTC transfer to the area from FFE20H to FFEDFH when performing self-programming. The RAM area used by the flash library starts at FDF00H. For the RAM areas used by the flash library, see Self RAM list of Flash Self-Programming Library for RL78 Family (R20UT2944).

**2.** Instructions can be executed from the RAM area excluding the general-purpose register area.

**3.** When boot swap is not used: Set the option bytes to 000C0H to 000C3H, and the on-chip debug security IDs to 000C4H to 000CDH.

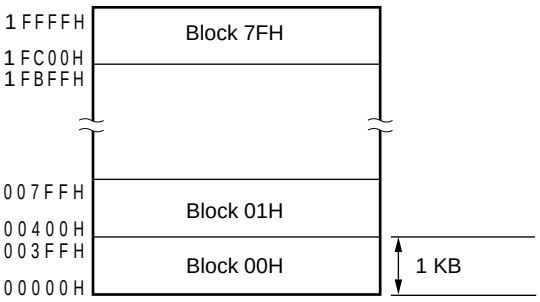
When boot swap is used: Set the option bytes to 000C0H to 000C3H and 010C0H to 010C3H, and the on-chip debug security IDs to 000C4H to 000CDH and 010C4H to 010CDH.

**4.** Writing boot cluster 0 can be prohibited depending on the setting of security (see **33.6 Security Settings**).

**5.** When using the trace function of on-chip debugging, area FE300H to FE6FFH is disabled.

**Caution When executing instructions from the RAM area 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. Reset signal generation sets RAM parity error resets to enabled (RPERDIS = 0). For details, see 30.3.3 RAM parity error detection function.**

**Remark** The flash memory is divided into blocks (one block = 1 KB). For the address values and block numbers, see **Table 3-1 Correspondence Between Address Values and Block Numbers in Flash Memory**.




(R5F10MMG, R5F10MPG)

Correspondence between the address values and block numbers in the flash memory are shown below.

**Table 3-1. Correspondence Between Address Values and Block Numbers in Flash Memory**

| Address Value | Block Number | Address Value | Block Number | Address Value | Block Number | Address Value | Block Number |
|---|---|---|---|---|---|---|---|
| 00000H to 003FFH | 00H | 08000H to 083FFH | 20H | 10000H to 103FFH | 40H | 18000H to 183FFH | 60H |
| 00400H to 007FFH | 01H | 08400H to 087FFH | 21H | 10400H to 107FFH | 41H | 18400H to 187FFH | 61H |
| 00800H to 00BFFH | 02H | 08800H to 08BFFH | 22H | 10800H to 10BFFH | 42H | 18800H to 18BFFH | 62H |
| 00C00H to 00FFFH | 03H | 08C00H to 08FFFH | 23H | 10C00H to 10FFFH | 43H | 18C00H to 18FFFH | 63H |
| 01000H to 013FFH | 04H | 09000H to 093FFH | 24H | 11000H to 113FFH | 44H | 19000H to 193FFH | 64H |
| 01400H to 017FFH | 05H | 09400H to 097FFH | 25H | 11400H to 117FFH | 45H | 19400H to 197FFH | 65H |
| 01800H to 01BFFH | 06H | 09800H to 09BFFH | 26H | 11800H to 11BFFH | 46H | 19800H to 19BFFH | 66H |
| 01C00H to 01FFFH | 07H | 09C00H to 09FFFH | 27H | 11C00H to 11FFFH | 47H | 19C00H to 19FFFH | 67H |
| 02000H to 023FFH | 08H | 0A000H to 0A3FFH | 28H | 12000H to 123FFH | 48H | 1A000H to 1A3FFH | 68H |
| 02400H to 027FFH | 09H | 0A400H to 0A7FFH | 29H | 12400H to 127FFH | 49H | 1A400H to 1A7FFH | 69H |
| 02800H to 02BFFH | 0AH | 0A800H to 0ABFFH | 2AH | 12800H to 12BFFH | 4AH | 1A800H to 1ABFFH | 6AH |
| 02C00H to 02FFFH | 0BH | 0AC00H to 0AFFFH | 2BH | 12C00H to 12FFFH | 4BH | 1AC00H to 1AFFFH | 6BH |
| 03000H to 033FFH | 0CH | 0B000H to 0B3FFH | 2CH | 13000H to 133FFH | 4CH | 1B000H to 1B3FFH | 6CH |
| 03400H to 037FFH | 0DH | 0B400H to 0B7FFH | 2DH | 13400H to 137FFH | 4DH | 1B400H to 1B7FFH | 6DH |
| 03800H to 03BFFH | 0EH | 0B800H to 0BBFFH | 2EH | 13800H to 13BFFH | 4EH | 1B800H to 1BBFFH | 6EH |
| 03C00H to 03FFFH | 0FH | 0BC00H to 0BFFFH | 2FH | 13C00H to 13FFFH | 4FH | 1BC00H to 1BFFFH | 6FH |
| 04000H to 043FFH | 10H | 0C000H to 0C3FFH | 30H | 14000H to 143FFH | 50H | 1C000H to 1C3FFH | 70H |
| 04400H to 047FFH | 11H | 0C400H to 0C7FFH | 31H | 14400H to 147FFH | 51H | 1C400H to 1C7FFH | 71H |
| 04800H to 04BFFH | 12H | 0C800H to 0CBFFH | 32H | 14800H to 14BFFH | 52H | 1C800H to 1CBFFH | 72H |
| 04C00H to 04FFFH | 13H | 0CC00H to 0CFFFH | 33H | 14C00H to 14FFFH | 53H | 1CC00H to 1CFFFH | 73H |
| 05000H to 053FFH | 14H | 0D000H to 0D3FFH | 34H | 15000H to 153FFH | 54H | 1D000H to 1D3FFH | 74H |
| 05400H to 057FFH | 15H | 0D400H to 0D7FFH | 35H | 15400H to 157FFH | 55H | 1D400H to 1D7FFH | 75H |
| 05800H to 05BFFH | 16H | 0D800H to 0DBFFH | 36H | 15800H to 15BFFH | 56H | 1D800H to 1DBFFH | 76H |
| 05C00H to 05FFFH | 17H | 0DC00H to 0DFFFH | 37H | 15C00H to 15FFFH | 57H | 1DC00H to 1DFFFH | 77H |
| 06000H to 063FFH | 18H | 0E000H to 0E3FFH | 38H | 16000H to 163FFH | 58H | 1E000H to 1E3FFH | 78H |
| 06400H to 067FFH | 19H | 0E400H to 0E7FFH | 39H | 16400H to 167FFH | 59H | 1E400H to 1E7FFH | 79H |
| 06800H to 06BFFH | 1AH | 0E800H to 0EBFFH | 3AH | 16800H to 16BFFH | 5AH | 1E800H to 1EBFFH | 7AH |
| 06C00H to 06FFFH | 1BH | 0EC00H to 0EFFFH | 3BH | 16C00H to 16FFFH | 5BH | 1EC00H to 1EFFFH | 7BH |
| 07000H to 073FFH | 1CH | 0F000H to 0F3FFH | 3CH | 17000H to 173FFH | 5CH | 1F000H to 1F3FFH | 7CH |
| 07400H to 077FFH | 1DH | 0F400H to 0F7FFH | 3DH | 17400H to 177FFH | 5DH | 1F400H to 1F7FFH | 7DH |
| 07800H to 07BFFH | 1EH | 0F800H to 0FBFFH | 3EH | 17800H to 17BFFH | 5EH | 1F800H to 1FBFFH | 7EH |
| 07C00H to 07FFFH | 1FH | 0FC00H to 0FFFFH | 3FH | 17C00H to 17FFFH | 5FH | 1FC00H to 1FFFFH | 7FH |

**Remark** R5F10MME, R5F10MPE : Block numbers 00H to 3FH
R5F10MMG, R5F10MPG : Block numbers 00H to 7FH

### 3.1.1 Internal program memory space

The internal program memory space stores the program and table data.

The RL78/I1B products incorporate internal ROM (flash memory), as shown below.

**Table 3-2. Internal ROM Capacity**

| Part Number | Internal ROM | |
|---|---|---|
| | Structure | Capacity |
| R5F10MME, R5F10MPE | Flash memory | 65536 × 8 bits (00000H to 0FFFFH) |
| R5F10MMG, R5F10MPG | | 131072 × 8 bits (00000H to 1FFFFH) |

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 01000H to 0107FH.

**Table 3-3. Vector Table (1/2)**

| Vector Table Address | Interrupt Source |
|---|---|
| 0000H | RESET, POR, LVD, WDT, TRAP, IAW, RPE |
| 0004H | INTWDTI |
| 0006H | INTLVI |
| 0008H | INTP0 |
| 000AH | INTP1 |
| 000CH | INTP2 |
| 000EH | INTP3 |
| 0010H | INTP4 |
| 0012H | INTP5 |
| 0014H | INTST2 |
| 0016H | INTSR2 |
| 0018H | INTSRE2 |
| 001EH | INTST0/INTCSI00/INTIIC00 |
| 0020H | INTTM00 |
| 0022H | INTSR0 |
| 0024H | INTSRE0 |
| | INTTM01H |
| 0026H | INTST1/INTIIC10 |
| 0028H | INTSR1 |
| 002AH | INTSRE1 |
| | INTTM03H |
| 002CH | INTIICA0 |
| 002EH | INTRTIT |
| 0030H | INTFM |
| 0032H | INTTM01 |
| 0034H | INTTM02 |
| 0036H | INTTM03 |
| 0038H | INTAD |
| 003AH | INTRTC |
| 003CH | INTIT |
| 0044H | INTDSAD |

**Table 3-3. Vector Table (2/2)**

| Vector Table Address | Interrupt Source |
|---|---|
| 0046H | INTTM04 |
| 0048H | INTTM05 |
| 004AH | INTP6 |
| 004CH | INTP7 |
| 0050H | INTCMP0 |
| 0052H | INTCMP1 |
| 0054H | INTTM06 |
| 0056H | INTTM07 |
| 0058H | INTIT00 |
| 005AH | INTIT01 |
| 005CH | INTCR |
| 0060H | INTOSDC |
| 0068H | INTIT10 |
| 006AH | INTIT11 |
| 006CH | INTVBAT |
| 007EH | BRK |

**(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 of 2 bytes).
To use the boot swap function, set a CALLT instruction table also at 01080H to 010BFH.

**(3) Option byte area**

A 4-byte area of 000C0H to 000C3H can be used as an option byte area. Set the option byte at 010C0H to 010C3H when the boot swap is used. For details, see **CHAPTER 32 OPTION BYTE**.

**(4) On-chip debug security ID setting area**

A 10-byte area of 000C4H to 000CDH and 010C4H to 010CDH can be used as an on-chip debug security ID setting area. Set the on-chip debug security ID of 10 bytes at 000C4H to 000CDH when the boot swap is not used and at 000C4H to 000CDH and 010C4H to 010CDH when the boot swap is used. For details, see **CHAPTER 34 ON-CHIP DEBUG FUNCTION**.

### 3.1.2 Mirror area

The RL78/I1B mirrors the code flash area of 00000H to 0FFFFH, to F0000H to FFFFFH. The products with 128 KB flash memory mirror the code flash area of 00000H to 0FFFFH or 10000H to 1FFFFH, to F0000H to FFFFFH (the code flash area to be mirrored is set by the processor mode control register (PMC)).

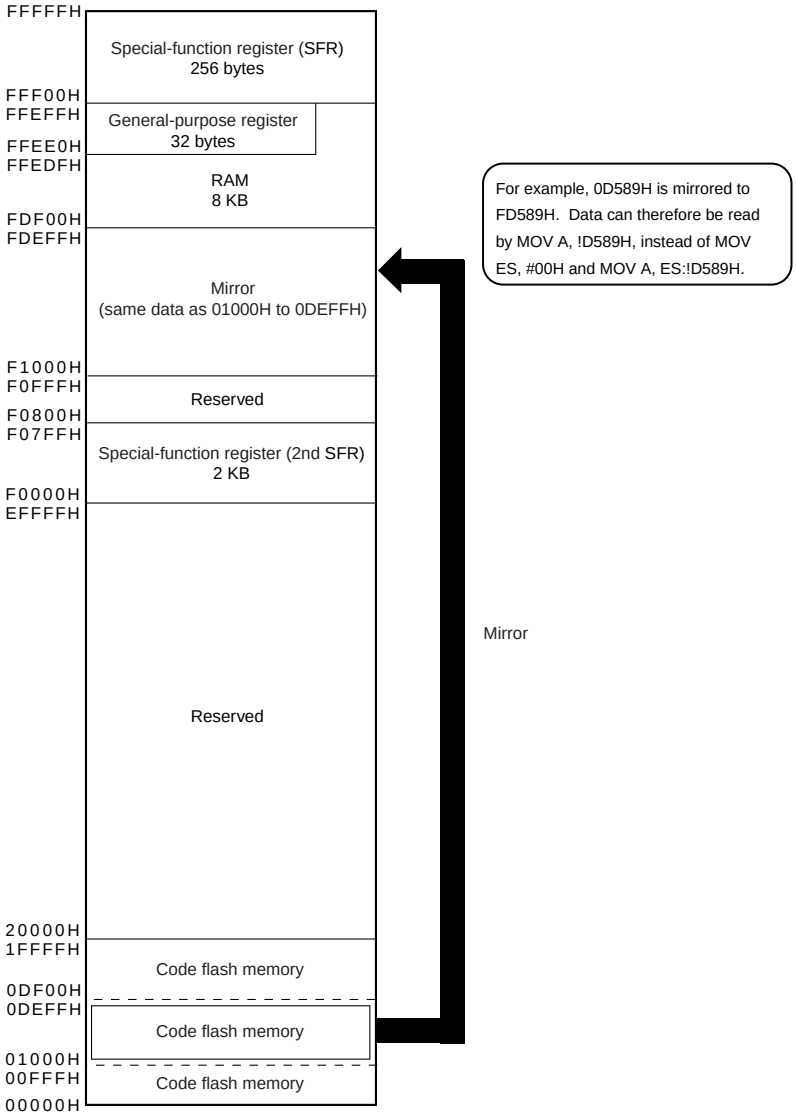
By reading data from F0000H to FFFFFH, an instruction that does not have the ES register as an operand can be used, and thus the contents of the code flash can be read with the shorter code. However, the code flash area is not mirrored to the SFR, extended SFR, RAM, and use prohibited areas.

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 R5F10MMG, R5F10MPG (Flash memory: 128 KB, RAM: 8 KB)**




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.

  Reset signal generation sets this register to 00H.

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

Address: FFFFEH After reset: 00H 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 |

**Cautions 1. In products with 64 KB flash memory, be sure to clear bit 0 (MAA) of this register to 0 (default value).**

**2. After setting the PMC register, wait for at least one instruction and access the mirror area.**

### 3.1.3 Internal data memory space

The RL78/I1B products incorporate the following RAMs.

**Table 3-4. Internal RAM Capacity**

| Part Number | Internal RAM |
|---|---|
| R5F10MME, R5F10MPE | 6144 × 8 bits (FE700H to FFEFFH) |
| R5F10MMG, R5F10MPG | 8192 × 8 bits (FDF00H to FFEFFH) |

The internal RAM can be used as a data area and a program area where instructions are executed. (Instructions cannot be executed in the area to which general-purpose registers are allocated.) 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.

**Cautions 1. The space (FFEE0H to FFEFFH) that the general-purpose registers are allocated cannot be used for fetching instructions or as a stack area.**

**2. Do not allocate RAM addresses which are used as a stack area, a data buffer, a branch destination of vector interrupt processing, and a DTC transfer destination/transfer source to the area FFE20H to FFEDFH when performing self-programming.**

**3. Use of the RAM areas of the following products is prohibited when performing self-programming, because these areas are used for each library.**

**R5F10MMG, R5F10MPG : FDF00H to FE309H**

**4. The internal RAM area of the following products cannot be used as a stack memory when using the trace function of on-chip debugging.**

**R5F10MME, R5F10MPE, R5F10MMG, R5F10MPG: 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** 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** in **3.2.5 Extended special function registers (2nd SFRs: 2nd Special Function Registers)**).

SFRs other than those in the SFR area (FFF00H to FFFFFH) are allocated to this area. An instruction that accesses the extended SFR area, however, is 1 byte longer than an instruction that accesses the SFR area.

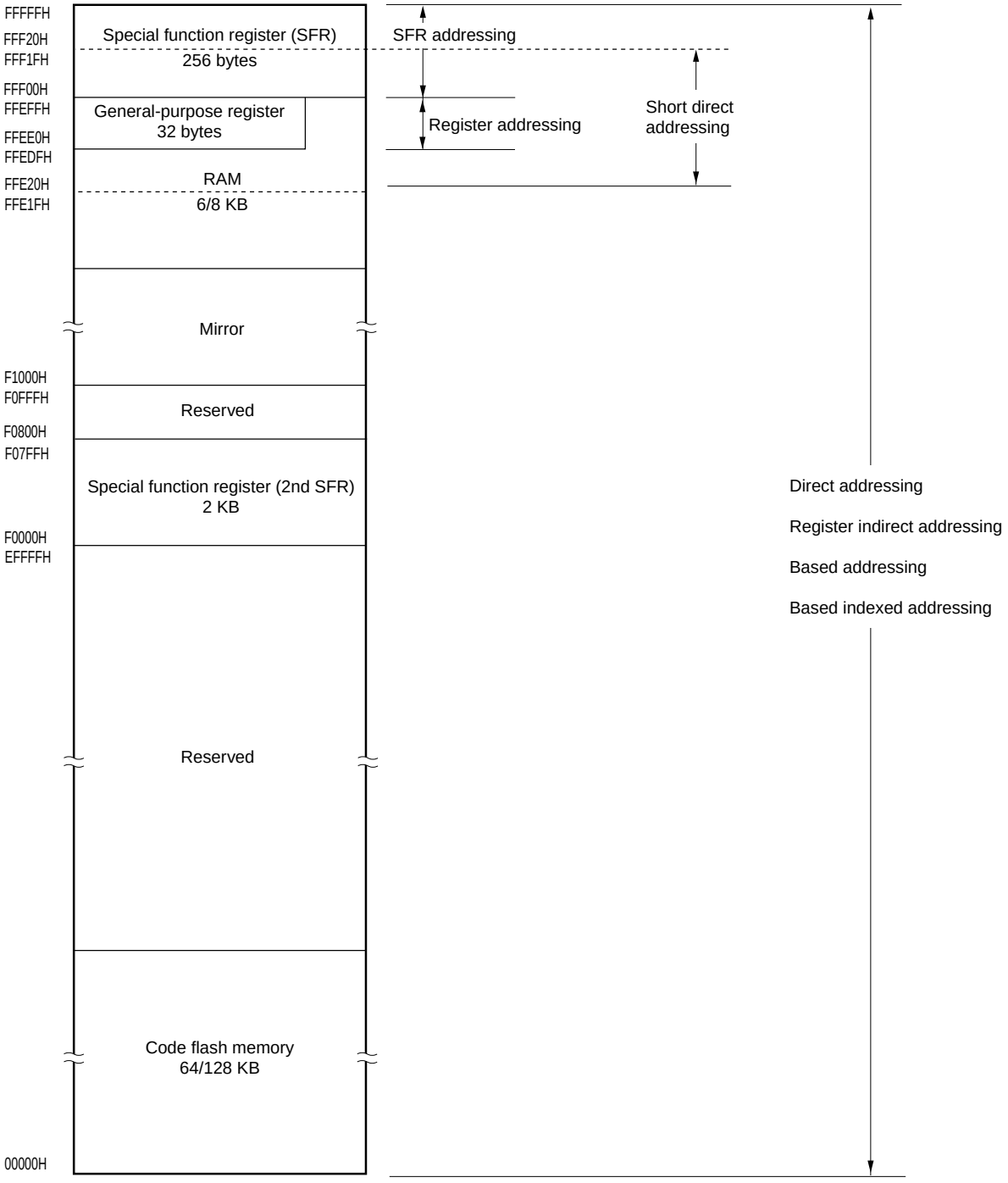
**Caution Do not access addresses to which extended SFRs are not assigned.**

### 3.1.6 Data memory addressing

Addressing refers to the method of specifying the address of the instruction to be executed next or the address of the register or memory relevant to the execution of instructions.

Several addressing modes are provided for addressing the memory relevant to the execution of instructions for the RL78/I1B, based on operability and other considerations. For areas containing data memory in particular, special addressing methods designed for the functions of the special function registers (SFR) and general-purpose registers are available for use. Figure 3-4 shows correspondence between data memory and addressing. For details of each addressing, see **3.4 Addressing for Processing Data Addresses**.

**Figure 3-4. Correspondence Between Data Memory and Addressing**



## 3.2 Processor Registers

The RL78/I1B products incorporate the following processor registers.

### 3.2.1 Control registers

The control registers control the program sequence, statuses 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)**

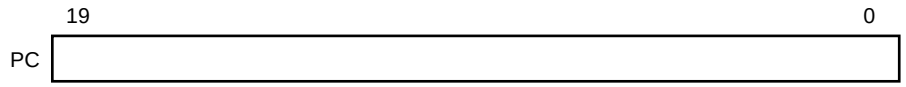
The program counter is a 20-bit register that holds the address information of the next program to be executed.

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.

Reset signal generation sets the reset vector table values at addresses 0000H and 0001H to the program counter.

**Figure 3-5. Format of Program Counter**




**(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. Reset signal generation sets the PSW register to 06H.

**Figure 3-6. Format of Program Status Word**

| | 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 maskable 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 (0) upon DI instruction execution or interrupt acknowledgment and is set (1) upon EI instruction execution.

**(b) Zero flag (Z)**

When the operation result is zero or equal, this flag is set (1). It is reset (0) in all other cases.

**(c) Register bank select flags (RBS0, RBS1)**

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 (1). It is reset (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 ISP0 and ISP1 flags by the priority specification flag registers (PRn0L, PRn0H, PRn1L, PRn1H, PRn2L, PRn2H, PRn3L) (see **23.3.3**) can not be acknowledged. Actual vectored interrupt 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**

| | 15 | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SP | SP15 | SP14 | SP13 | SP12 | SP11 | SP10 | SP9 | SP8 | 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 (restored) from the stack memory.

**Cautions 1. Since reset signal generation makes the SP contents undefined, be sure to initialize the SP before using the stack.**

**2. It is prohibited to use the general-purpose register (FFEE0H to FFEFFH) space for fetching instructions or a stack area.**

**3. Do not allocate RAM addresses which are used as a stack area, a data buffer, a branch destination of vector interrupt processing, and a DTC transfer destination/transfer source to the area FFE20H to FFEDFH when performing self-programming.**

**4. Use of the RAM areas of the following products is prohibited when performing self-programming, because these areas are used for each library.**

**R5F10MMG, R5F10MPG: FDF00H to FE309H**

**5. The internal RAM area of the following products cannot be used as a stack memory when using the trace function of on-chip debugging.**

**R5F10MME, R5F10MPE, R5F10MMG, R5F10MPG: 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 consists of 4 banks, each bank consisting of eight 8-bit registers (X, A, C, B, E, D, L, and H).

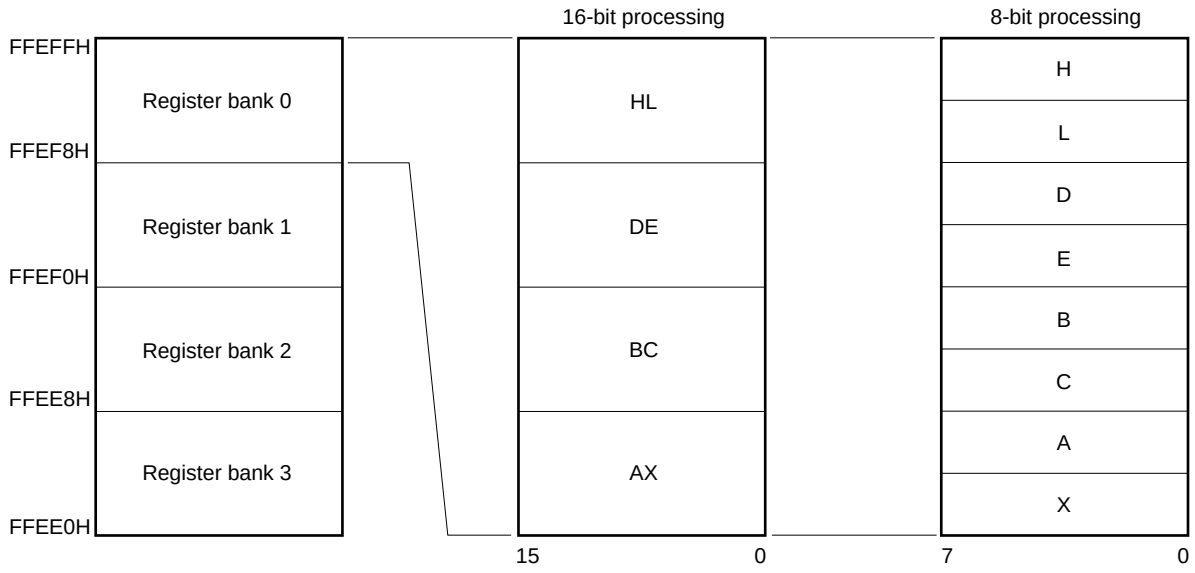
Each register can be used as an 8-bit register, and two 8-bit registers can also be used in a pair as a 16-bit register (AX, BC, DE, and HL).

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 interrupts for each bank.

**Caution It is prohibited to use the general-purpose register (FFEE0H to FFEFFH) space for fetching instructions or as a stack area.**

**Figure 3-8. Configuration of General-Purpose Registers**

**(a) 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 direct addressing), respectively.

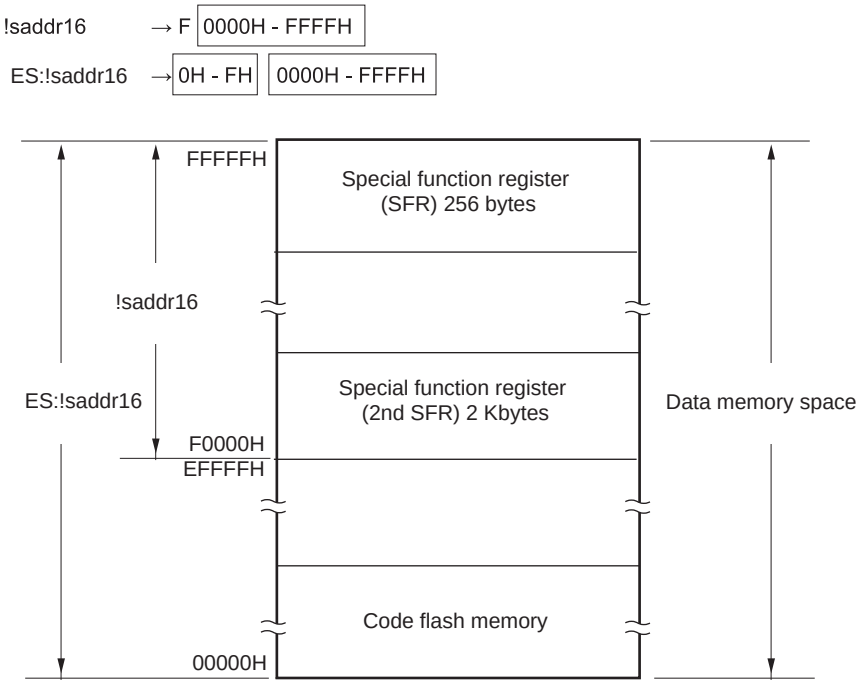
The default value of the ES register after reset is 0FH, and that of the CS register is 00H.

**Figure 3-9. Configuration of ES and CS Registers**

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

The data area that can be accessed by using 16-bit addresses is the 64 KB from F0000H to FFFFFH. By using the ES register, this area can be extended to the 1 MB 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 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 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 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 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 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 a list of the SFRs. The meanings of items in the table are as follows.

- Symbol
  Symbol indicating 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
  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
  Indicates each register status upon reset signal generation.

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

**Remark** For extended SFRs (2nd SFRs), see **3.2.5 Extended special function registers (2nd SFRs: 2nd Special Function Registers)**.

**Table 3-5. SFR List (1/5)**

| Address | Special Function Register (SFR) Name | Symbol | | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|---|
| | | | | | 1-bit | 8-bit | 16-bit | |
| 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 |
| FFF08H | Port register 8 | P8 | | R/W | √ | √ | – | 00H |
| FFF0CH | Port register 12 | P12 | | R/W | √ | √ | – | Undefined |
| FFF0DH | Port register 13 | P13 | | R/W | √ | √ | – | Undefined |
| FFF10H | Serial data register 00 | TXD0/ SIO00 | SDR00 | R/W | – | √ | √ | 0000H |
| FFF11H | | – | | | – | – | | |
| FFF12H | Serial data register 01 | RXD0 | 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 |
| FFF28H | Port mode register 8 | PM8 | | R/W | √ | √ | – | FFH |
| FFF2CH | Port mode register 12 | PM12 | | 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 |

**Table 3-5. SFR List (2/5)**

| Address | Special Function Register (SFR) Name | Symbol | | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|---|
| | | | | | 1-bit | 8-bit | 16-bit | |
| FFF38H | External interrupt rising edge enable register 0 | EGP0 | | R/W | √ | √ | – | 00H |
| FFF39H | External interrupt falling edge enable register 0 | EGN0 | | R/W | √ | √ | – | 00H |
| FFF40H | LCD mode register 0 | LCDM0 | | R/W | – | √ | – | 00H |
| FFF41H | LCD mode register 1 | LCDM1 | | R/W | √ | √ | – | 00H |
| FFF42H | LCD clock control register | LCDC0 | | R/W | – | √ | – | 00H |
| FFF43H | LCD boost level control register | VLCD | | R/W | – | √ | – | 04H |
| FFF44H | Serial data register 02 | TXD1/ SIO10 | SDR02 | R/W | – | √ | √ | 0000H |
| FFF45H | | – | | | – | – | | |
| FFF46H | Serial data register 03 | RXD1 | SDR03 | R/W | – | √ | √ | 0000H |
| FFF47H | | – | | | – | – | | |
| FFF48H | Serial data register 10 | TXD2 | SDR10 | R/W | – | √ | √ | 0000H |
| FFF49H | | – | | | – | – | | |
| FFF4AH | Serial data register 11 | RXD2 | 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 | | | | | | | | |

**Table 3-5. SFR List (3/5)**

| Address | Special Function Register (SFR) Name | Symbol | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|
| | | | | 1-bit | 8-bit | 16-bit | |
| FFF90H<br>FFF91H | 12-bit interval timer control register | ITMC | R/W | – | – | √ | 0FFFH |
| FFF92H | Second count register | SEC | R/W | – | √ | – | Undefined |
| FFF93H | Minute count register | MIN | R/W | – | √ | – | Undefined |
| FFF94H | Hour count register | HOUR | R/W | – | √ | – | Undefined |
| FFF95H | Week count register | WEEK | R/W | – | √ | – | Undefined |
| FFF96H | Day count register | DAY | R/W | – | √ | – | Undefined |
| FFF97H | Month count register | MONTH | R/W | – | √ | – | Undefined |
| FFF98H | Year count register | YEAR | R/W | – | √ | – | Undefined |
| FFF9AH | Alarm minute register | ALARMWM | R/W | – | √ | – | Undefined |
| FFF9BH | Alarm hour register | ALARMWH | R/W | – | √ | – | Undefined |
| FFF9CH | Alarm week register | ALARMWW | R/W | – | √ | – | Undefined |
| FFF9DH | Real-time clock control register 0 | RTCC0 | R/W | √ | √ | – | 00H Note 1 |
| FFF9EH | Real-time clock control register 1 | RTCC1 | R/W | √ | √ | – | 00H Note 1 |
| FFFA0H | Clock operation mode control register | CMC | R/W | – | √ | – | 00H Note 1 |
| FFFA1H | Clock operation status control register | CSC | R/W | √ | √ | – | C0H Note 1 |
| 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 |
| FFFA5H | Clock output select register 0 | CKS0 | R/W | √ | √ | – | 00H |
| FFFA6H | Clock output select register 1 | CKS1 | R/W | √ | √ | – | 00H |
| FFFA8H | Reset control flag register | RESF | R | – | √ | – | Undefined Note 2 |
| FFFA9H | Voltage detection register | LVIM | R/W | √ | √ | – | 00H Note 2 |
| FFFAAH | Voltage detection level register | LVIS | R/W | √ | √ | – | 00H/01H/81H Note 2 |
| FFFABH | Watchdog timer enable register | WDTE | R/W | – | √ | – | 1AH/9AH Note 3 |
| FFFACH | CRC input register | CRCIN | R/W | – | √ | – | 00H |

**Notes 1.** This register is reset only by a power-on reset.

**2.** The reset values of the registers vary depending on the reset source as shown below.

| Register \ Reset Source | | $\overline{\text{RESET}}$ 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 LVD |
|---|---|---|---|---|---|---|---|---|
| RESF | TRAP | Cleared (0) | | Set (1) | Held | | | Held |
| | WDTRF | | | Held | Set (1) | Held | | |
| | RPERF | | | Held | | Set (1) | Held | |
| | IAWRF | | | Held | | | Set (1) | |
| | LVIRF | | | Held | | | | Set (1) |
| LVIM | LVISEN | Cleared (0) | | | | | | Held |
| | LVIOMSK | Held | | | | | | |
| | LVIF | | | | | | | |
| LVIS | | Cleared (00H/01H/81H) | | | | | | Clear (00H/81H) Note 4 |

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

**4.** When option byte LVIMDS1, LVIMDS0 = 0, 1: LVD reset is not generated.

**Table 3-5. SFR List (4/5)**

| Address | Special Function Register (SFR) Name | Symbol | | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|---|
| | | | | | 1-bit | 8-bit | 16-bit | |
| FFFD0H | Interrupt request flag register 2L | IF2L | IF2 | R/W | √ | √ | √ | 00H |
| FFFD1H | Interrupt request flag register 2H | IF2H | | R/W | √ | √ | | 00H |
| FFFD2H | Interrupt request flag register 3L | IF3L | IF3 | R/W | √ | √ | √ | 00H |
| FFFD4H | Interrupt mask flag register 2L | MK2L | MK2 | R/W | √ | √ | √ | FFH |
| FFFD5H | Interrupt mask flag register 2H | MK2H | | R/W | √ | √ | | FFH |
| FFFD6H | Interrupt mask flag register 3L | MK3L | MK3 | R/W | √ | √ | √ | FFH |
| FFFD8H | Priority specification flag register 02L | PR02L | PR02 | R/W | √ | √ | √ | FFH |
| FFFD9H | Priority specification flag register 02H | PR02H | | R/W | √ | √ | | FFH |
| FFFDAH | Priority specification flag register 03L | PR03L | PR03 | R/W | √ | √ | √ | FFH |
| FFFDCH | Priority specification flag register 12L | PR12L | PR12 | R/W | √ | √ | √ | FFH |
| FFFDDH | Priority specification flag register 12H | PR12H | | R/W | √ | √ | | FFH |
| FFFDEH | Priority specification flag register 13L | PR13L | PR13 | R/W | √ | √ | √ | FFH |
| FFFE0H | Interrupt request flag register 0L | IF0L | IF0 | R/W | √ | √ | √ | 00H |
| FFFE1H | Interrupt request flag register 0H | IF0H | | R/W | √ | √ | | 00H |
| FFFE2H | Interrupt request flag register 1L | IF1L | IF1 | R/W | √ | √ | √ | 00H |
| FFFE3H | Interrupt request flag register 1H | IF1H | | R/W | √ | √ | | 00H |
| FFFE4H | Interrupt mask flag register 0L | MK0L | MK0 | R/W | √ | √ | √ | FFH |
| FFFE5H | Interrupt mask flag register 0H | MK0H | | R/W | √ | √ | | FFH |
| FFFE6H | Interrupt mask flag register 1L | MK1L | MK1 | R/W | √ | √ | √ | FFH |
| FFFE7H | Interrupt mask flag register 1H | MK1H | | R/W | √ | √ | | FFH |
| FFFE8H | Priority specification flag register 00L | PR00L | PR00 | R/W | √ | √ | √ | FFH |
| FFFE9H | Priority specification flag register 00H | PR00H | | R/W | √ | √ | | FFH |

**Table 3-5. SFR List (5/5)**

| Address | Special Function Register (SFR) Name | Symbol | | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|---|
| | | | | | 1-bit | 8-bit | 16-bit | |
| FFFEAH | Priority specification flag register 01L | PR01L | PR01 | R/W | √ | √ | √ | FFH |
| FFFEBH | Priority specification flag register 01H | PR01H | | R/W | √ | √ | | FFH |
| FFFECH | Priority specification flag register 10L | PR10L | PR10 | R/W | √ | √ | √ | FFH |
| FFFEDH | Priority specification flag register 10H | PR10H | | R/W | √ | √ | | FFH |
| FFFEEH | Priority specification flag register 11L | PR11L | PR11 | R/W | √ | √ | √ | FFH |
| FFFEFH | Priority specification flag register 11H | PR11H | | R/W | √ | √ | | FFH |
| FFFF0H<br>FFFF1H | Multiply and accumulation register (L) | MACRL | | R/W | – | – | √ | 0000H |
| FFFF2H<br>FFFF3H | Multiply and accumulation register (H) | MACRH | | R/W | – | – | √ | 0000H |
| FFFFEH | Processor mode control register | PMC | | R/W | √ | √ | – | 00H |

**Remark** For extended SFRs (2nd SFRs), see **Table 3-6 Extended SFR (2nd SFR) List**.

### 3.2.5 Extended special function registers (2nd SFRs: 2nd Special Function Registers)

Unlike a general-purpose register, each extended SFR (2nd SFR) has a special function.

Extended 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 extended SFR area, however, is 1 byte longer than an instruction that accesses the SFR area.

Extended SFRs can be manipulated like general-purpose registers, using operation, transfer, and bit 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 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 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 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 a list of the extended SFRs. The meanings of items in the table are as follows.

- Symbol
  Symbol indicating the address of an extended 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
  Indicates whether the corresponding extended 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
  Indicates each register status upon reset signal generation.

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

**Remark** For SFRs in the SFR area, see **3.2.4 Special function registers (SFRs)**.

**Table 3-6. Extended SFR (2nd SFR) List (1/8)**

| Address | Special Function Register (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 |
| F0038H | Pull-up resistor option register 8 | PU8 | R/W | √ | √ | – | 00H |
| F003CH | Pull-up resistor option register 12 | PU12 | R/W | √ | √ | – | 00H |
| F0040H | Port input mode register 0 | PIM0 | R/W | √ | √ | – | 00H |
| F0041H | Port input mode register 1 | PIM1 | R/W | √ | √ | – | 00H |
| F0048H | Port input mode register 8 | PIM8 | R/W | √ | √ | – | 00H |
| F0050H | Port output mode register 0 | POM0 | R/W | √ | √ | – | 00H |
| F0051H | Port output mode register 1 | POM1 | R/W | √ | √ | – | 00H |
| F0058H | Port output mode register 8 | POM8 | R/W | √ | √ | – | 00H |
| 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 |
| F0076H | A/D port configuration register | ADPC | R/W | – | √ | – | 00H |
| F0077H | Peripheral I/O redirection register | PIOR | R/W | – | √ | – | 00H |
| F0078H | Invalid memory access detection control register | IAWCTL | R/W | – | √ | – | 00H |
| F007AH | Peripheral enable register 1 | PER1 | R/W | √ | √ | – | 00H |
| F007BH | Port mode select resister | PMS | R/W | √ | √ | – | 00H |
| F007DH | Global digital input disable register | GDIDIS | R/W | √ | √ | – | 00H |
| F0098H | Peripheral clock control register | PCKC | R/W | √ | √ | – | 00H |
| F00A8H | High-speed on-chip oscillator frequency select register | HOCODIV | R/W | – | √ | – | Undefined[Note 1] |
| F00F0H | Peripheral enable register 0 | PER0 | R/W | √ | √ | – | 00H |
| F00F3H | Subsystem clock supply mode control register | OSMC | R/W | – | √ | – | 00H |
| F00F5H | RAM parity error control register | RPECTL | R/W | √ | √ | – | 00H |
| F00F9H | Power-on-reset status register | PORSR | R/W | – | √ | – | 00H[Note 2] |
| F00FEH | BCD adjust result register | BCDADJ | R | – | √ | – | Undefined |

**Notes 1.** The reset value of the HOCODIV register is determined by the setting of the option byte (000C2H).

**2.** This register is reset only by a power-on reset.

**Table 3-6. Extended SFR (2nd SFR) List (2/8)**

| Address | Special Function Register (SFR) Name | Symbol | | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|---|
| | | | | | 1-bit | 8-bit | 16-bit | |
| 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 | | – | | | – | – | | |
| 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 | | – | | | – | – | | |

**Table 3-6. Extended SFR (2nd SFR) List (3/8)**

| Address | Special Function Register (SFR) Name | Symbol | | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|---|
| | | | | | 1-bit | 8-bit | 16-bit | |
| 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 | | – | | | – | – | | |

**Table 3-6. Extended SFR (2nd SFR) List (4/8)**

| Address | Special Function Register (SFR) Name | Symbol | | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|---|
| | | | | | 1-bit | 8-bit | 16-bit | |
| F0180H | Timer counter register 00 | TCR00 | | R | – | – | √ | FFFFH |
| F0181H | | | | | | | | |
| 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 | | – | | | – | – | | |

**Table 3-6. Extended SFR (2nd SFR) List (5/8)**

| Address | Special Function Register (SFR) Name | Symbol | | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|---|
| | | | | | 1-bit | 8-bit | 16-bit | |
| 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 | | – | | | – | – | | |
| 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 | | – | | | – | – | | |
| 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 |
| F02D0H | Oscillation stop detection register | OSDC | | R/W | – | – | √ | 0FFFH |
| F02D8H | High-speed on-chip oscillator clock frequency correction control register | HOCOFC | | R/W | – | √ | – | 00H |
| F02E0H | DTC base address register | DTCBAR | | R/W | – | √ | – | 00H |
| F02E8H | DTC enable register 0 | DTCEN0 | | R/W | √ | √ | – | 00H |
| F02E9H | DTC enable register 1 | DTCEN1 | | R/W | √ | √ | – | 00H |
| F02EAH | DTC enable register 2 | DTCEN2 | | R/W | √ | √ | – | 00H |
| F02EBH | DTC enable register 3 | DTCEN3 | | R/W | √ | √ | – | 00H |
| 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 | LCD port function register 0 | PFSEG0 | | R/W | √ | √ | – | F0H |
| F0301H | LCD port function register 1 | PFSEG1 | | R/W | √ | √ | – | FFH |
| F0302H | LCD port function register 2 | PFSEG2 | | R/W | √ | √ | – | FFH |
| F0303H | LCD port function register 3 | PFSEG3 | | R/W | √ | √ | – | FFH |
| F0304H | LCD port function register 4 | PFSEG4 | | R/W | √ | √ | – | FFH |
| F0305H | LCD port function register 5 | PFSEG5 | | R/W | √ | √ | – | FFH |

**Table 3-6. Extended SFR (2nd SFR) List (6/8)**

| Address | Special Function Register (SFR) Name | Symbol | | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|---|
| | | | | | 1-bit | 8-bit | 16-bit | |
| F0308H | LCD Input switch control register | ISCLCD | | R/W | √ | √ | – | 00H |
| F0310H | Watch error correction register | SUBCUD | | R/W | – | – | √ | 0020H[Note] |
| F0312H | Frequency measurement count register L | FMCRL | | R | – | – | √ | 0000H |
| F0314H | Frequency measurement count register H | FMCRH | | R | – | – | √ | 0000H |
| F0316H | Frequency measurement control register | FMCTL | | R/W | √ | – | – | 00H |
| F0330H | Backup power switch control register 0 | BUPCTL0 | | R/W | √ | √ | – | 00H |
| F0340H | Comparator mode setting register | COMPMDR | | R/W | √ | √ | – | 00H |
| F0341H | Comparator filter control register | COMPFIR | | R/W | √ | √ | – | 00H |
| F0342H | Comparator output control register | COMPOCR | | R/W | √ | √ | – | 00H |
| F0350H | 8-bit interval timer compare register 00 | TRTCMP00 | TRTCMP0 | R/W | – | √ | √ | FFH |
| F0351H | 8-bit interval timer compare register 01 | TRTCMP01 | | R/W | – | √ | | FFH |
| F0352H | 8-bit interval timer control register 0 | TRTCR0 | | R/W | √ | √ | – | 00H |
| F0353H | 8-bit interval timer frequency division register 0 | TRTMD0 | | R/W | – | √ | – | 00H |
| F0358H | 8-bit interval timer compare register 10 | TRTCMP10 | TRTCMP1 | R/W | – | √ | √ | FFH |
| F0359H | 8-bit interval timer compare register 11 | TRTCMP11 | | R/W | – | √ | | FFH |
| F035AH | 8-bit interval timer control register 1 | TRTCR1 | | R/W | √ | √ | – | 00H |
| F035BH | 8-bit interval timer frequency division register 1 | TRTMD1 | | R/W | – | √ | – | 00H |
| F03A0H | IrDA control register | IRCR | | R/W | √ | √ | – | 00H |
| F03B0H | Temperature sensor control register | TMPCTL | | R/W | √ | √ | – | 00H |

**Note** This register is reset only by a power-on reset.

**Table 3-6. Extended SFR (2nd SFR) List (7/8)**

| Address | Special Function Register (SFR) Name | Symbol | | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|---|
| | | | | | 1-bit | 8-bit | 16-bit | |
| F03C0H | ΔΣ A/D converter mode register | DSADMR | | R/W | – | – | √ | 0000H |
| F03C2H | ΔΣ A/D converter gain control register 0 | DSADGCR0 | | R/W | – | √ | – | 00H |
| F03C3H | ΔΣ A/D converter gain control register 1 | DSADGCR1 | | R/W | – | √ | – | 00H |
| F03C5H | ΔΣ A/D converter HPF control register | DSADHPFCR | | R/W | – | √ | – | 00H |
| F03C6H | ΔΣ A/D converter phase control register 0 | DSADPHCR0 | | R/W | – | – | √ | 0000H |
| F03C8H | ΔΣ A/D converter phase control register 1 | DSADPHCR1 | | R/W | – | – | √ | 0000H |
| F03D0H | ΔΣ A/D converter conversion result register 0L | DSADCR0L | DSADCR0 | R | – | √ | √ | 00H |
| F03D1H | ΔΣ A/D converter conversion result register 0M | DSADCR0M | | R | – | √ | | 00H |
| F03D2H | ΔΣ A/D converter conversion result register 0H | DSADCR0H | | R | – | √ | – | 00H |
| F03D4H | ΔΣ A/D converter conversion result register 1L | DSADCR1L | DSADCR1 | R | – | √ | √ | 00H |
| F03D5H | ΔΣ A/D converter conversion result register 1M | DSADCR1M | | R | – | √ | | 00H |
| F03D6H | ΔΣ A/D converter conversion result register 1H | DSADCR1H | | R | – | √ | – | 00H |
| F03D8H | ΔΣ A/D converter conversion result register 2L | DSADCR2L | DSADCR2 | R | – | √ | √ | 00H |
| F03D9H | ΔΣ A/D converter conversion result register 2M | DSADCR2M | | R | – | √ | | 00H |
| F03DAH | ΔΣ A/D converter conversion result register 2H | DSADCR2H | | R | – | √ | – | 00H |
| F03DCH | ΔΣ A/D converter conversion result register 3L | DSADCR3L | DSADCR3 | R | – | √ | √ | 00H |
| F03DDH | ΔΣ A/D converter conversion result register 3M | DSADCR3M | | R | – | √ | | 00H |
| F03DEH | ΔΣ A/D converter conversion result register 3H | DSADCR3H | | R | – | √ | – | 00H |
| F0400H | LCD display data memory 0 | SEG0 | | R/W | – | √ | – | 00H |
| F0401H | LCD display data memory 1 | SEG1 | | R/W | – | √ | – | 00H |
| F0402H | LCD display data memory 2 | SEG2 | | R/W | – | √ | – | 00H |
| F0403H | LCD display data memory 3 | SEG3 | | R/W | – | √ | – | 00H |
| F0404H | LCD display data memory 4 | SEG4 | | R/W | – | √ | – | 00H |
| F0405H | LCD display data memory 5 | SEG5 | | R/W | – | √ | – | 00H |
| F0406H | LCD display data memory 6 | SEG6 | | R/W | – | √ | – | 00H |
| F0407H | LCD display data memory 7 | SEG7 | | R/W | – | √ | – | 00H |
| F0408H | LCD display data memory 8 | SEG8 | | R/W | – | √ | – | 00H |
| F0409H | LCD display data memory 9 | SEG9 | | R/W | – | √ | – | 00H |
| F040AH | LCD display data memory 10 | SEG10 | | R/W | – | √ | – | 00H |

**Table 3-6. Extended SFR (2nd SFR) List (8/8)**

| Address | Special Function Register (SFR) Name | Symbol | | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|---|
| | | | | | 1-bit | 8-bit | 16-bit | |
| F040BH | LCD display data memory 11 | SEG11 | | R/W | – | √ | – | 00H |
| F040CH | LCD display data memory 12 | SEG12 | | R/W | – | √ | – | 00H |
| F040DH | LCD display data memory 13 | SEG13 | | R/W | – | √ | – | 00H |
| F040EH | LCD display data memory 14 | SEG14 | | R/W | – | √ | – | 00H |
| F040FH | LCD display data memory 15 | SEG15 | | R/W | – | √ | – | 00H |
| F0410H | LCD display data memory 16 | SEG16 | | R/W | – | √ | – | 00H |
| F0411H | LCD display data memory 17 | SEG17 | | R/W | – | √ | – | 00H |
| F0412H | LCD display data memory 18 | SEG18 | | R/W | – | √ | – | 00H |
| F0413H | LCD display data memory 19 | SEG19 | | R/W | – | √ | – | 00H |
| F0414H | LCD display data memory 20 | SEG20 | | R/W | – | √ | – | 00H |
| F0415H | LCD display data memory 21 | SEG21 | | R/W | – | √ | – | 00H |
| F0416H | LCD display data memory 22 | SEG22 | | R/W | – | √ | – | 00H |
| F0417H | LCD display data memory 23 | SEG23 | | R/W | – | √ | – | 00H |
| F0418H | LCD display data memory 24 | SEG24 | | R/W | – | √ | – | 00H |
| F0419H | LCD display data memory 25 | SEG25 | | R/W | – | √ | – | 00H |
| F041AH | LCD display data memory 26 | SEG26 | | R/W | – | √ | – | 00H |
| F041BH | LCD display data memory 27 | SEG27 | | R/W | – | √ | – | 00H |
| F041CH | LCD display data memory 28 | SEG28 | | R/W | – | √ | – | 00H |
| F041DH | LCD display data memory 29 | SEG29 | | R/W | – | √ | – | 00H |
| F041EH | LCD display data memory 30 | SEG30 | | R/W | – | √ | – | 00H |
| F041FH | LCD display data memory 31 | SEG31 | | R/W | – | √ | – | 00H |
| F0420H | LCD display data memory 32 | SEG32 | | R/W | – | √ | – | 00H |
| F0421H | LCD display data memory 33 | SEG33 | | R/W | – | √ | – | 00H |
| F0422H | LCD display data memory 34 | SEG34 | | R/W | – | √ | – | 00H |
| F0423H | LCD display data memory 35 | SEG35 | | R/W | – | √ | – | 00H |
| F0424H | LCD display data memory 36 | SEG36 | | R/W | – | √ | – | 00H |
| F0425H | LCD display data memory 37 | SEG37 | | R/W | – | √ | – | 00H |
| F0426H | LCD display data memory 38 | SEG38 | | R/W | – | √ | – | 00H |
| F0427H | LCD display data memory 39 | SEG39 | | R/W | – | √ | – | 00H |
| F0428H | LCD display data memory 40 | SEG40 | | R/W | – | √ | – | 00H |
| F0429H | LCD display data memory 41 | SEG41 | | R/W | – | √ | – | 00H |
| F0540H | 8-bit interval timer count register 00 | TRT00 | TRT0 | R | – | √ | √ | 00H |
| F0541H | 8-bit interval timer count register 01 | TRT01 | | R | – | √ | | 00H |
| F0548H | 8-bit interval timer count register 10 | TRT10 | TRT1 | R | – | √ | √ | 00H |
| F0549H | 8-bit interval timer count register 11 | TRT11 | | R | – | √ | | 00H |

**Remark** For SFRs in the SFR area, see **Table 3-5 SFR List**.

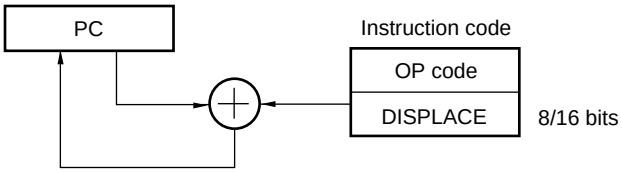
## 3.3 Instruction Address Addressing

### 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**

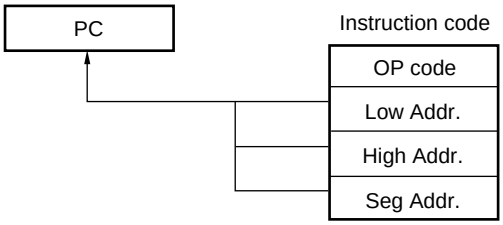
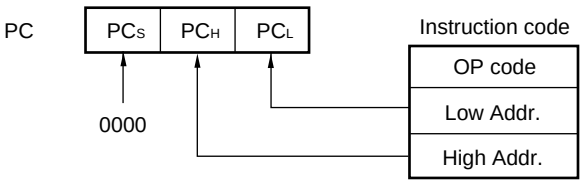



**Figure 3-13. Example of CALL !addr16/BR !addr16**



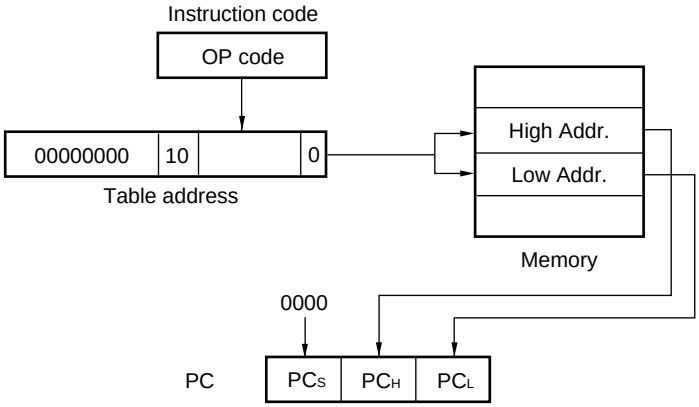
### 3.3.3 Table indirect addressing

**[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 KB space from 00000H to 0FFFFH.

**Figure 3-14. Outline of Table Indirect Addressing**

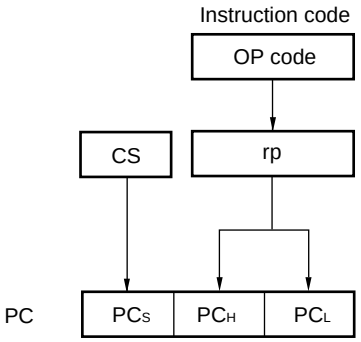



### 3.3.4 Register direct addressing

**[Function]**

Register direct 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 direct addressing can be applied only to the CALL AX, BC, DE, HL, and BR AX instructions.

**Figure 3-15. Outline of Register Direct Addressing**



## 3.4 Addressing for Processing Data Addresses

### 3.4.1 Implied addressing

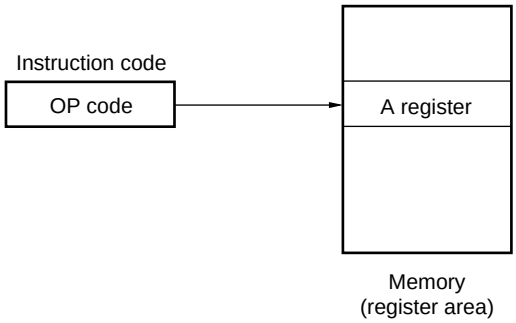
**[Function]**

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

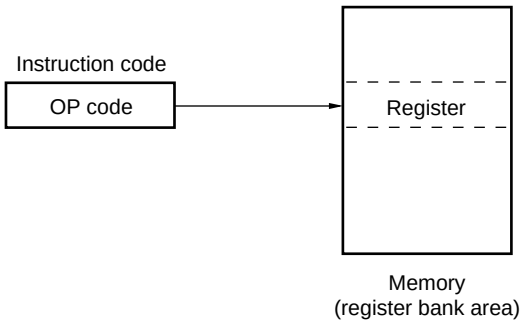
**[Function]**

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

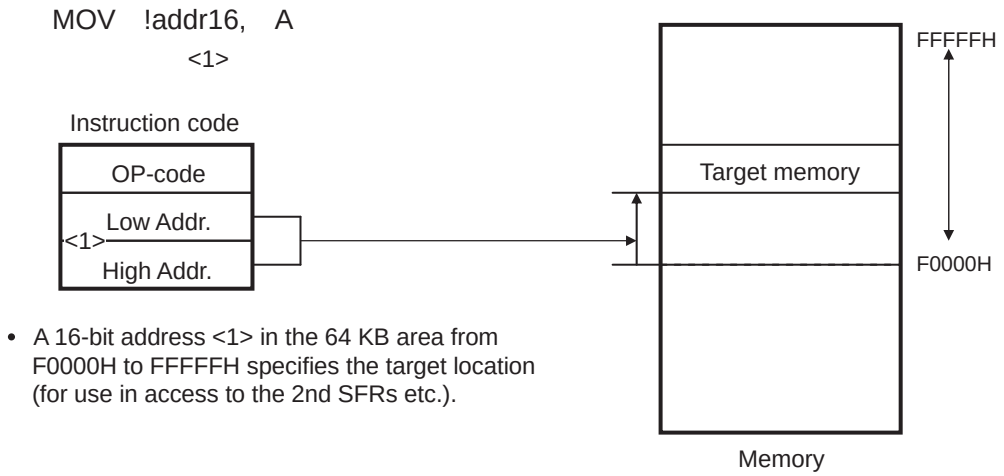
**[Function]**

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

**[Operand format]**

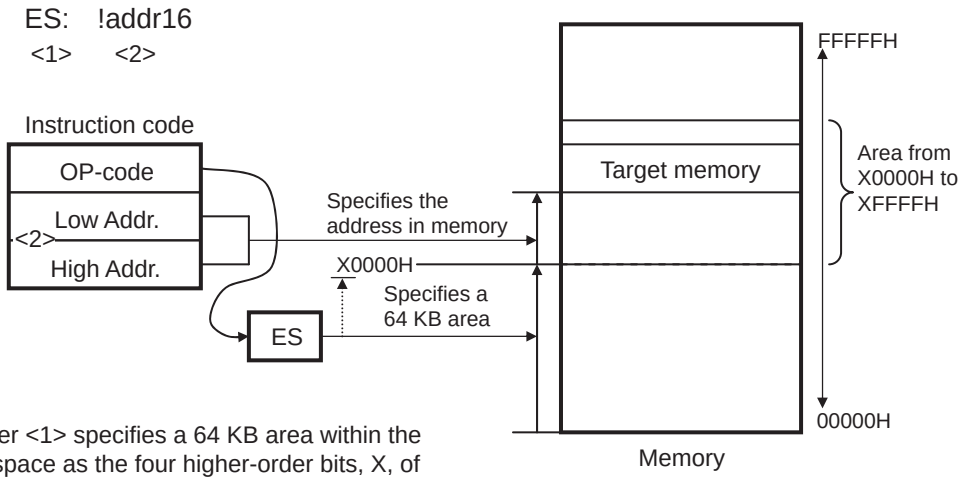
| Identifier | Description |
|---|---|
| !addr16 | Label or 16-bit immediate data (only the space from F0000H to FFFFFH is specifiable) |
| ES:!addr16 | Label or 16-bit immediate data (higher 4-bit addresses are specified by the ES register) |

**Figure 3-18. Example of !addr16**




- A 16-bit address <1> in the 64 KB area from F0000H to FFFFFH specifies the target location (for use in access to the 2nd SFRs etc.).

**Figure 3-19. Example of ES:!addr16**




- The ES register <1> specifies a 64 KB area within the overall 1 MB space as the four higher-order bits, X, of the address range.
- A 16-bit address <2> in the area from X0000H to XFFFFH and the ES register <1> specify the target location; this is used for access to fixed data other than that in mirrored areas.

### 3.4.4 Short direct addressing

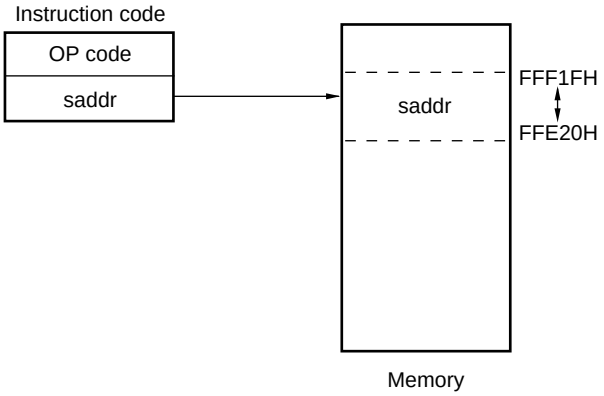
**[Function]**

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<br>(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)<br>(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

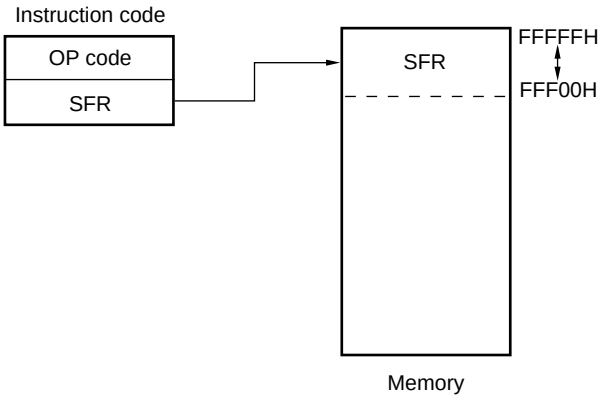
**[Function]**

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

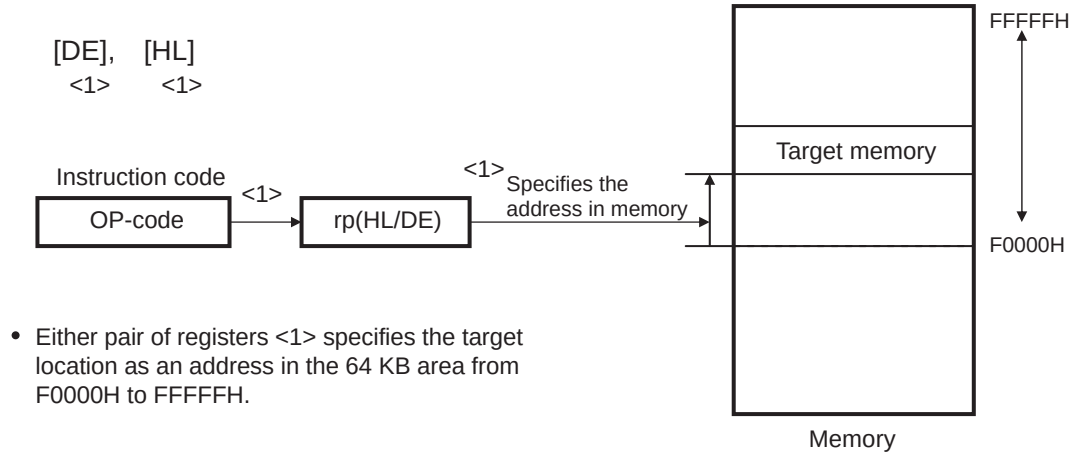
**[Function]**

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]**

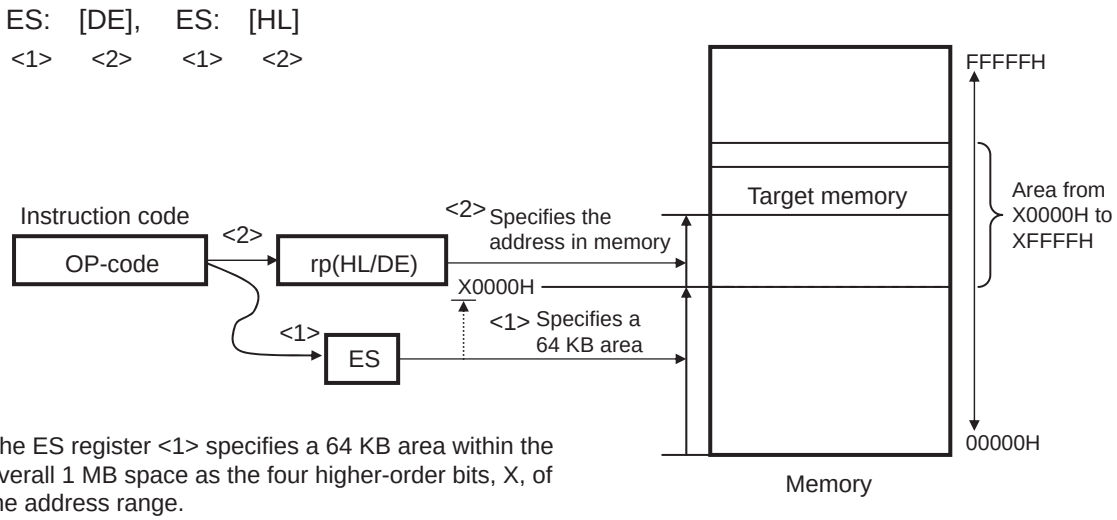
| Identifier | Description |
|---|---|
| – | [DE], [HL] (only the space from F0000H to FFFFFH is specifiable) |
| – | ES:[DE], ES:[HL] (higher 4-bit addresses are specified by the ES register) |

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




- Either pair of registers <1> specifies the target location as an address in the 64 KB area from F0000H to FFFFFH.

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




- The ES register <1> specifies a 64 KB area within the overall 1 MB space as the four higher-order bits, X, of the address range.
- Either pair of registers <2> and the ES register <1> specify the target location in the area from X0000H to XFFFFH.

### 3.4.7 Based addressing

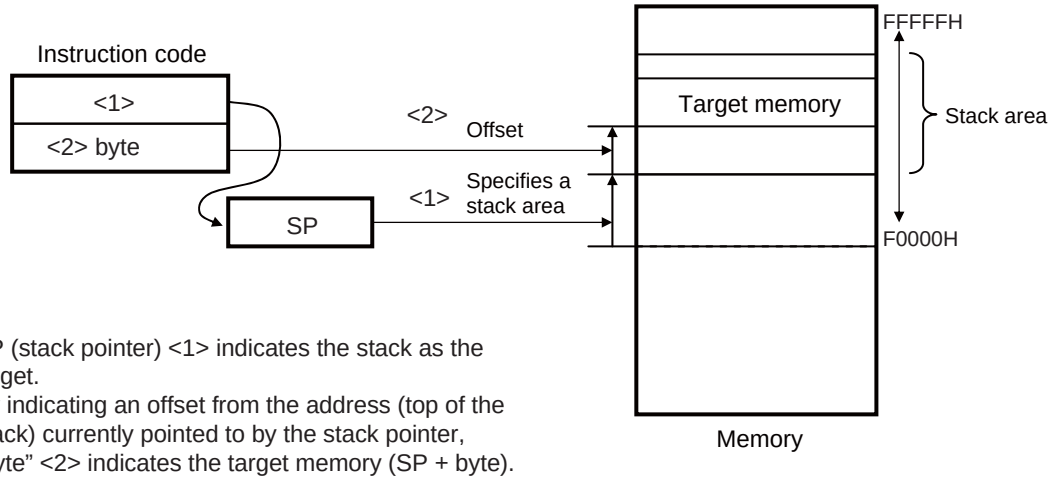
**[Function]**

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]**

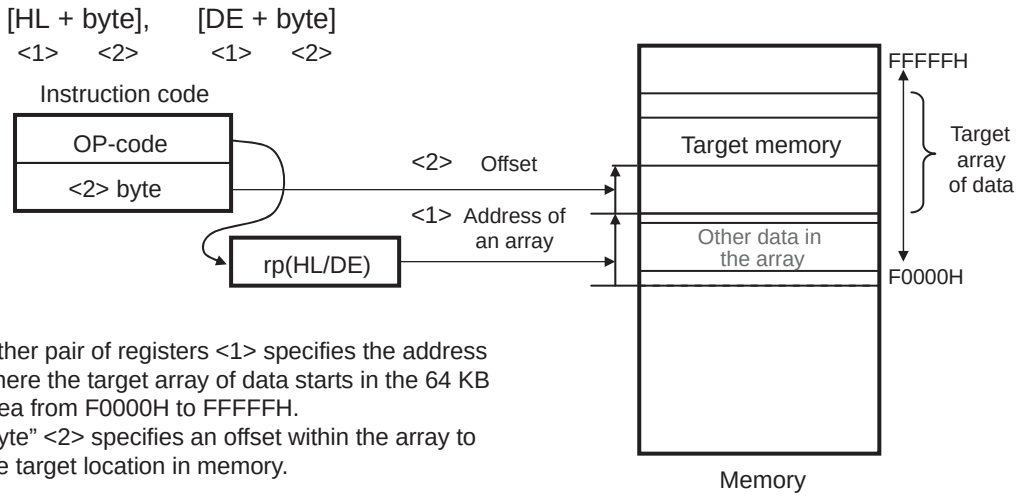
| Identifier | Description |
|---|---|
| – | [HL + byte], [DE + byte], [SP + byte] (only the space from F0000H to FFFFFH is specifiable) |
| – | word[B], word[C] (only the space from F0000H to FFFFFH is specifiable) |
| – | word[BC] (only the space from F0000H to FFFFFH is specifiable) |
| – | ES:[HL + 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]**



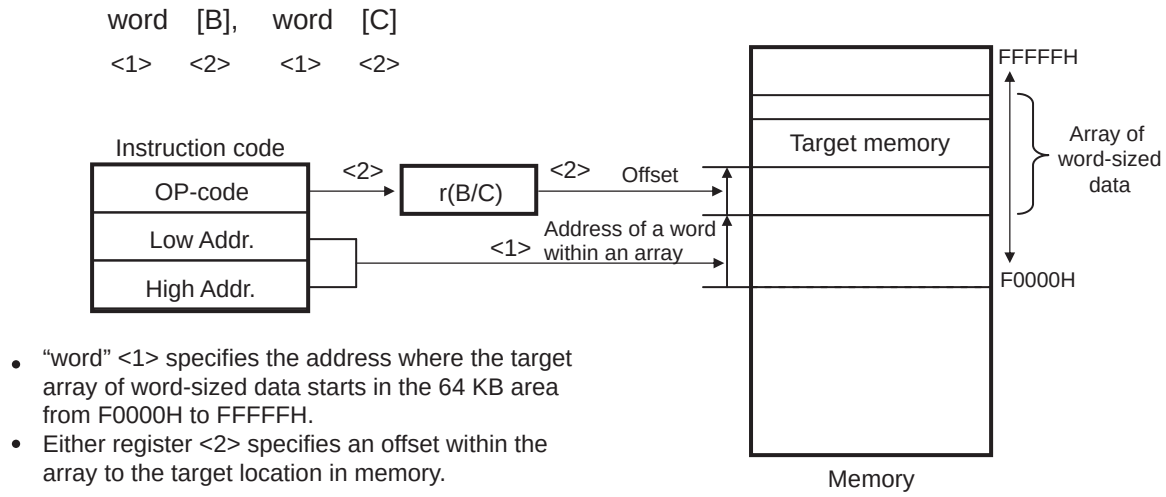

- SP (stack pointer) <1> indicates the stack as the target.
- By indicating an offset from the address (top of the stack) currently pointed to by the stack pointer, “byte” <2> indicates the target memory (SP + byte).

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



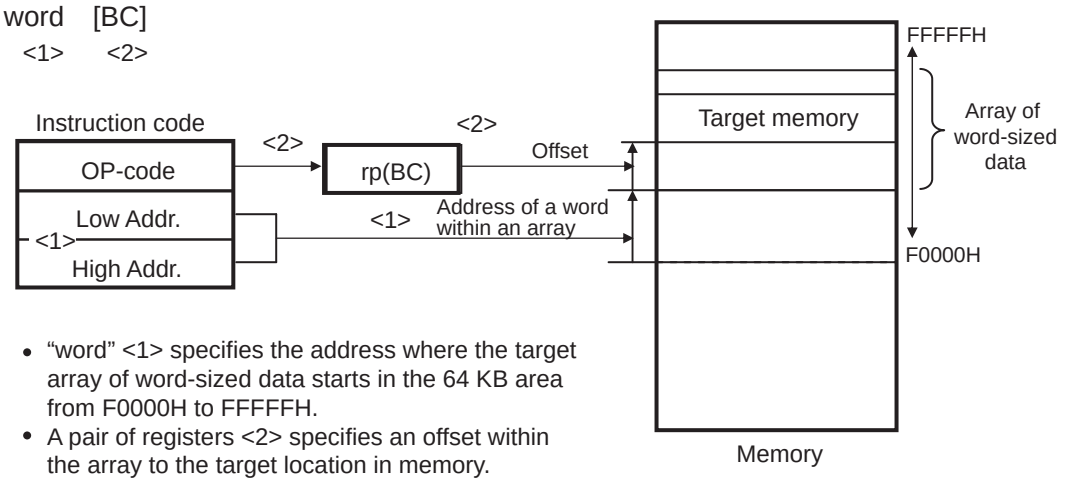

- Either pair of registers <1> specifies the address where the target array of data starts in the 64 KB area from F0000H to FFFFFH.
- "byte" <2> specifies an offset within the array to the target location in memory.

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



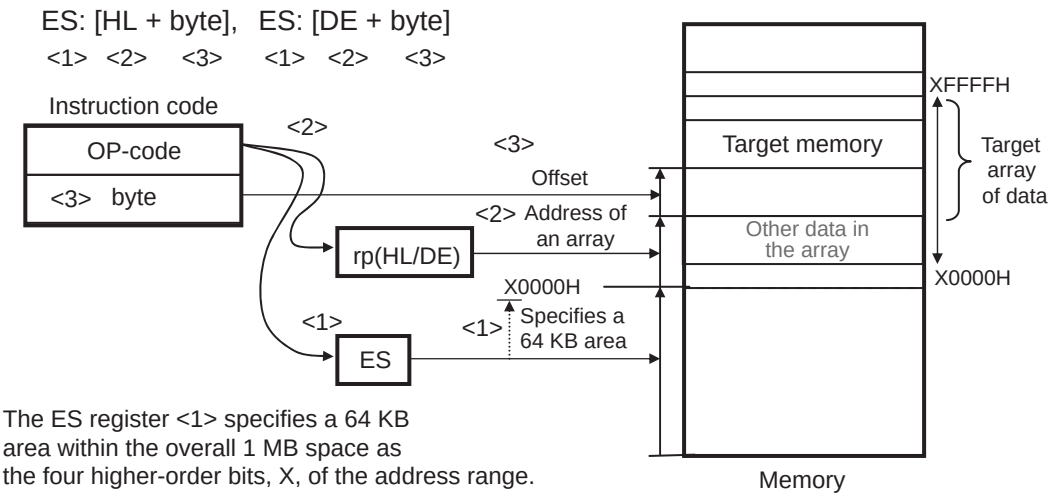

- "word" <1> specifies the address where the target array of word-sized data starts in the 64 KB area from F0000H to FFFFFH.
- Either register <2> specifies an offset within the array to the target location in memory.

**Figure 3-27. Example of word[BC]**



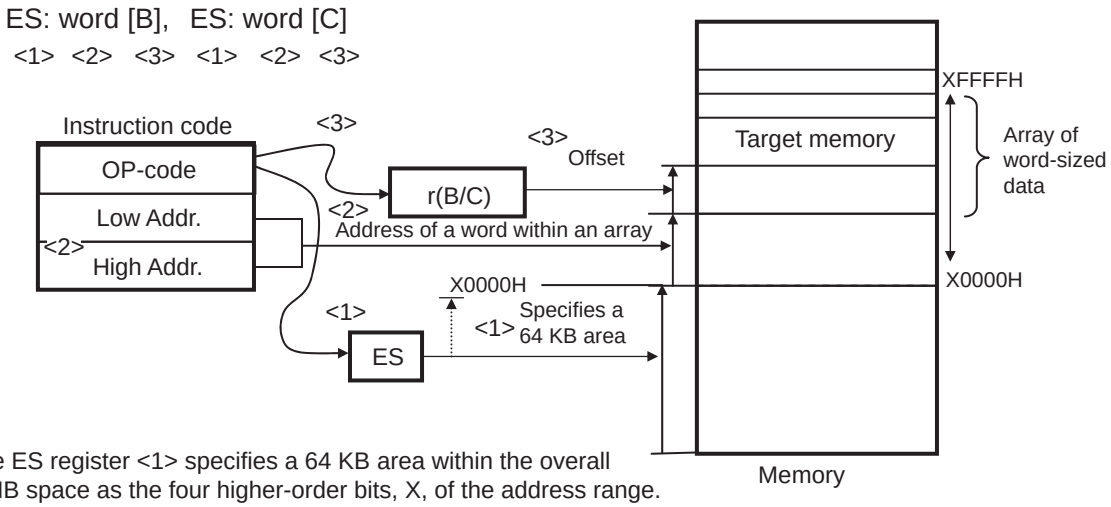

- "word" <1> specifies the address where the target array of word-sized data starts in the 64 KB area from F0000H to FFFFFH.
- A pair of registers <2> specifies an offset within the array to the target location in memory.

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



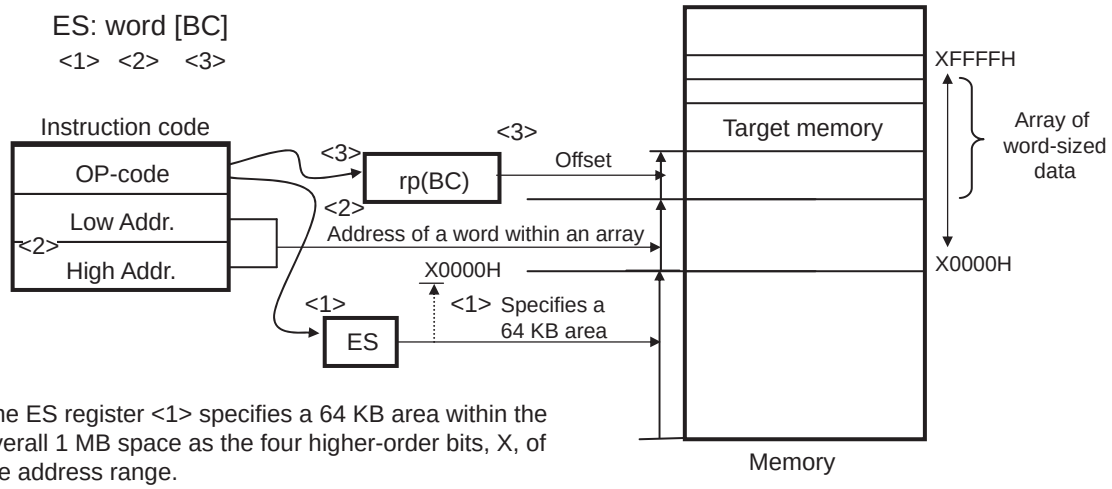

- The ES register <1> specifies a 64 KB area within the overall 1 MB space as the four higher-order bits, X, of the address range.
- Either pair of registers <2> specifies the address where the target array of data starts in the 64 KB area specified in the ES register <1>.
- “byte” <3> specifies an offset within the array to the target location in memory.

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




- The ES register <1> specifies a 64 KB area within the overall 1 MB space as the four higher-order bits, X, of the address range.
- “word” <2> specifies the address where the target array of word-sizeddata starts in the 64 KB area specified in the ES register <1>.
- Either register <3> specifies an offset within the array tothe target location in memory.

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




- The ES register <1> specifies a 64 KB area within the overall 1 MB 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 KB 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

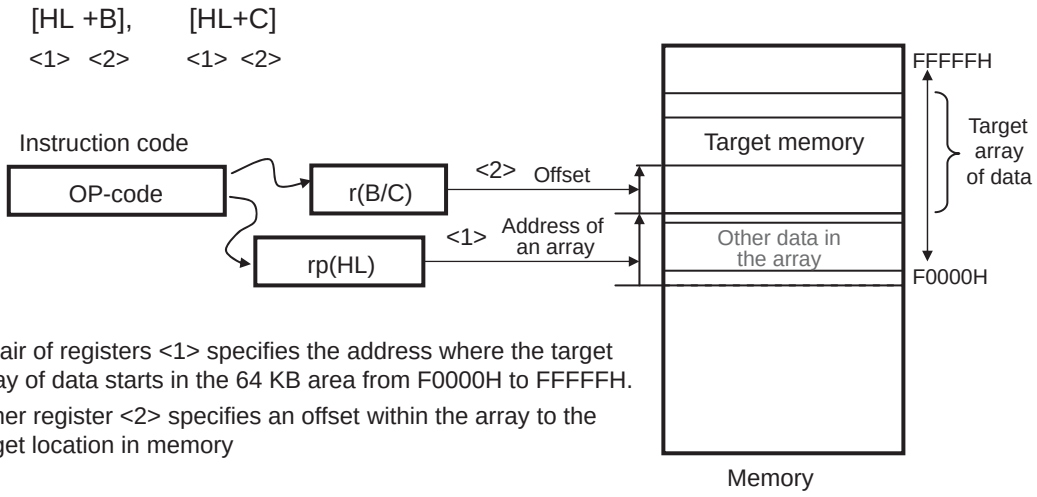
**[Function]**

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]**

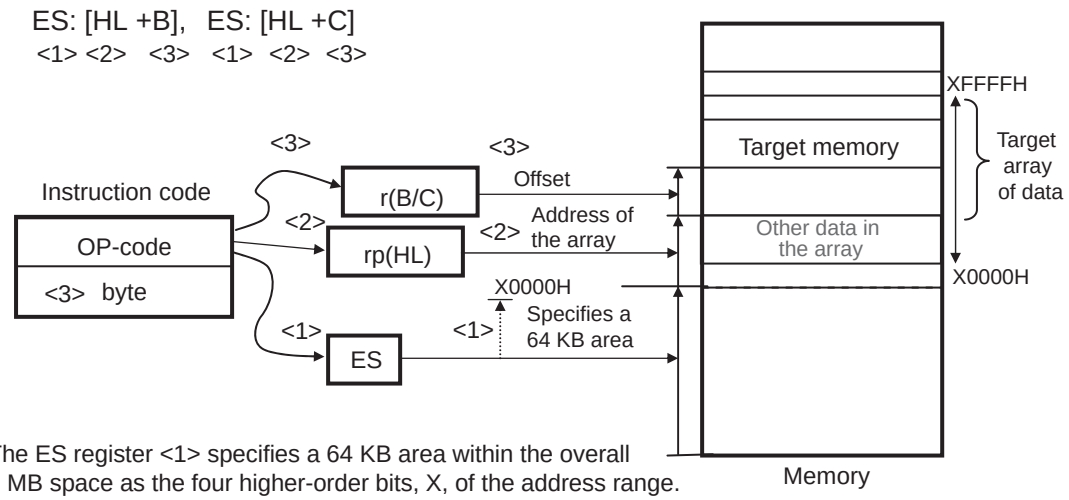
| Identifier | Description |
|---|---|
| – | [HL+B], [HL+C] (only the space from F0000H to FFFFFH is specifiable) |
| – | ES:[HL+B], ES:[HL+C] (higher 4-bit addresses are specified by the ES register) |

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




- A pair of registers <1> specifies the address where the target array of data starts in the 64 KB area from F0000H to FFFFFH.
- Either register <2> specifies an offset within the array to the target location in memory

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




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

### 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.

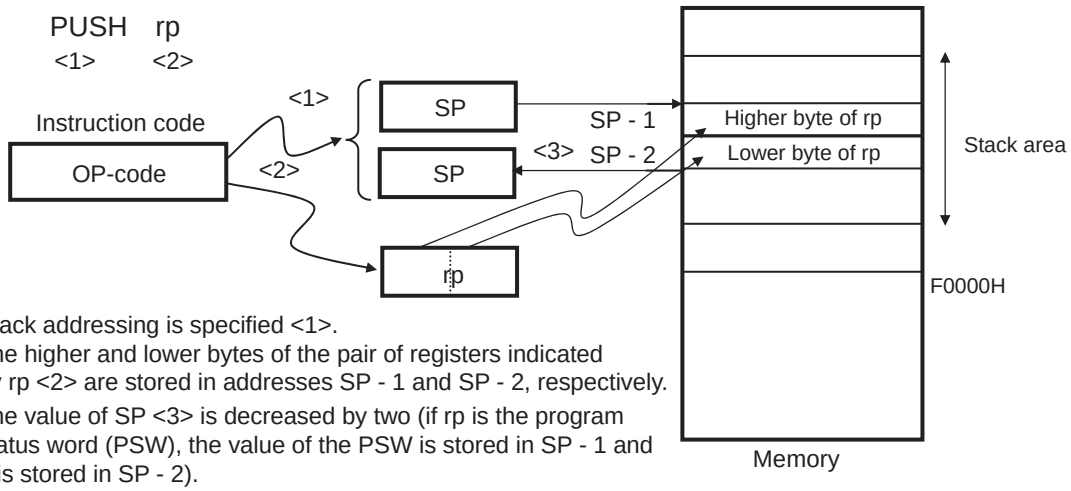
Only the internal RAM area can be set as the stack area.

**[Operand format]**

| Identifier | Description |
|---|---|
| – | PUSH PSW AX/BC/DE/HL<br>POP PSW AX/BC/DE/HL<br>CALL/CALLT<br>RET<br>BRK<br>RETB<br>(Interrupt request generated)<br>RETI |

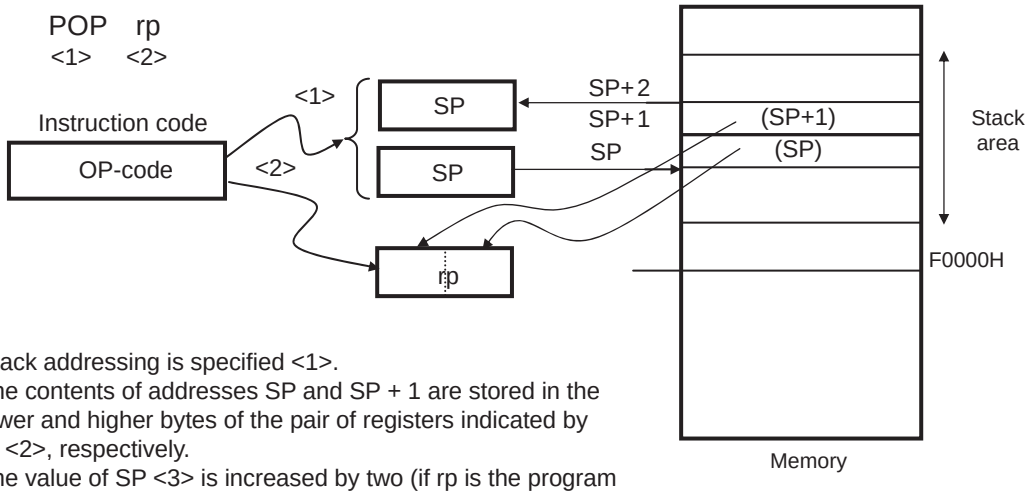
Each stack operation saves or restores data as shown in **Figures 3-33** to **3-38**.

**Figure 3-33. Example of PUSH rp**



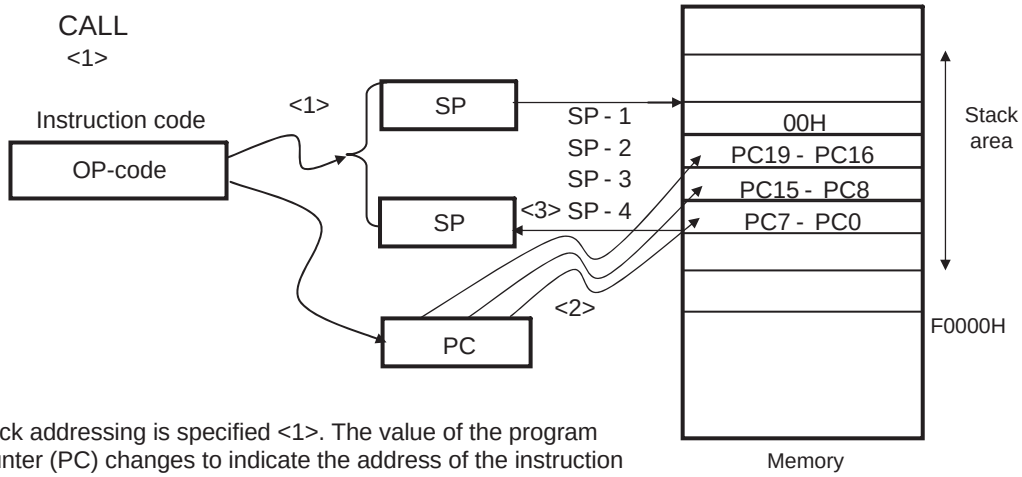

- Stack addressing is specified <1>.
- The higher and lower bytes of the pair of registers indicated by rp <2> are stored in addresses SP - 1 and SP - 2, respectively.
- The value of SP <3> is decreased by two (if rp is the program status word (PSW), the value of the PSW is stored in SP - 1 and 0 is stored in SP - 2).

**Figure 3-34. Example of POP**



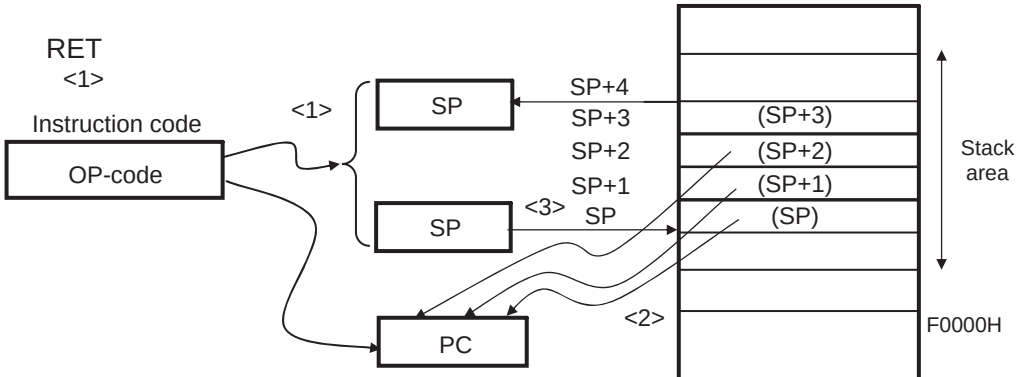

- Stack addressing is specified <1>.
- The contents of addresses SP and SP + 1 are stored in the lower and higher bytes of the pair of registers indicated by rp <2>, respectively.
- The value of SP <3> is increased by two (if rp is the program status word (PSW), the content of address SP + 1 is stored in the PSW).

**Figure 3-35. Example of CALL, CALLT**



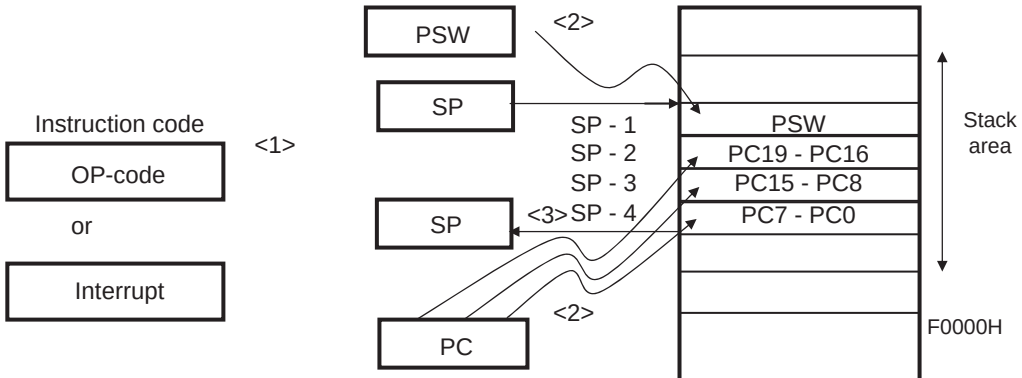

- Stack addressing is specified <1>. The value of the program counter (PC) changes to indicate the address of the instruction following the CALL instruction.
- 00H, the values of PC bits 19 to 16, 15 to 8, and 7 to 0 are stored in addresses SP - 1, SP - 2, SP - 3, and SP - 4, respectively <2>.
- The value of the SP <3> is decreased by 4.

**Figure 3-36. Example of RET**



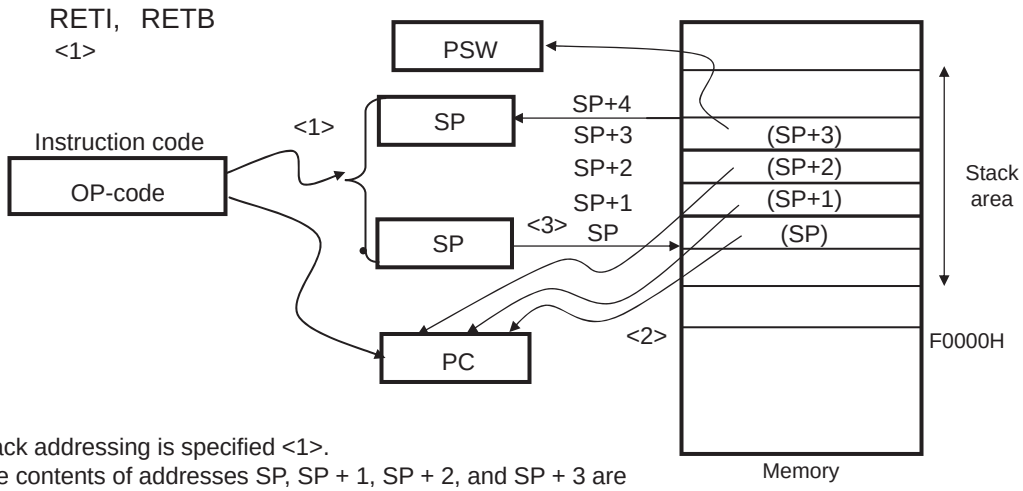

- Stack addressing is specified <1>.
- The contents of addresses SP, SP + 1, and SP + 2 are stored in PC bits 7 to 0, 15 to 8, and 19 to 16, respectively <2>.
- The value of SP <3> is increased by four.

**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.

# CHAPTER 4 PORT FUNCTIONS

## 4.1 Port Functions

The RL78/I1B microcontrollers 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 **CHAPTER 2 PIN FUNCTIONS**.

## 4.2 Port Configuration

Ports include the following hardware.

**Table 4-1. Port Configuration**

| Item | Configuration |
|---|---|
| Control registers | Port mode registers (PM0 to PM8, PM12)<br>Port registers (P0 to P8, P12, P13)<br>Pull-up resistor option registers (PU0, PU1, PU3 to PU5, PU7, PU8, PU12)<br>Port input mode registers (PIM0, PIM1, PIM8)<br>Port output mode registers (POM0, POM1, POM8)<br>A/D port configuration register (ADPC)<br>Peripheral I/O redirection register (PIOR)<br>Global digital input disable register (GDIDIS)<br>LCD port function registers (PFSEG0 to PFSEG5)<br>LCD input switch control register (ISCLCD) |
| Port | • 80-pin products<br>Total: 53 (CMOS I/O: 44 (N-ch open drain I/O [$V_{DD}$ tolerance]: 13), CMOS input: 5, CMOS output: 1, N-ch open drain I/O [6 V tolerance]: 3)<br>• 100-pin products<br>Total: 69 (CMOS I/O: 60 (N-ch open drain I/O [$V_{DD}$ tolerance]: 13), CMOS input: 5, CMOS output: 1, N-ch open drain I/O [6 V tolerance]: 3) |
| Pull-up resistor | • 80-pin products Total: 40<br>• 100-pin products Total: 54 |

### 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 using port mode register 0 (PM0). When the P00 to P07 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by pull-up resistor option register 0 (PU0).

Input to the P00, P03, P05 and P06 pins can be specified through a normal input buffer or a TTL input buffer in 1-bit units using port input mode register 0 (PIM0).

Output from the P01 to P07 pins can be specified as N-ch open-drain output ($V_{DD}$ tolerance) in 1-bit units using port output mode register 0 (POM0).

This port can also be used for programming UART transmission/reception, IrDA transmission, serial interface data I/O, and clock I/O, timer I/O, external interrupt request input, and comparator output. For the 80-pin products, this port can be used for segment output of LCD controller/driver.

Reset signal generation sets port 0 to input mode. For the 80-pin products, the P00 and P01 pins are set to input mode, and P02 to P07 pins are set to the digital input invalid mode[Note].

**Note** "Digital input invalid" refers to the state in which all the digital outputs, digital inputs, and LCD outputs are disabled.

### 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 using port mode register 1 (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 pull-up resistor option register 1 (PU1).

Input to the P15 and P16 pins can be specified through a normal input buffer or a TTL input buffer in 1-bit units using port input mode register 1 (PIM1).

Output from the P15 to P17 pins can be specified as N-ch open-drain output ($V_{DD}$ tolerance) in 1-bit units using port output mode register 1 (POM1).

This port can also be used for serial interface data I/O, clock I/O, and segment output of LCD controller/driver.

Reset signal generation sets port 1 to the digital input invalid mode[Note].

**Note** "Digital input invalid" refers to the state in which all the digital outputs, digital inputs, and LCD outputs are disabled.

### 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 using port mode register 2 (PM2).

This port can also be used for A/D converter analog input, (+side and –side) reference voltage input, comparator reference voltage input, and comparator analog voltage input.

To use P20/ANI0 to P25/ANI15 as digital I/O pins, set them in the digital I/O mode by using the A/D port configuration register (ADPC). Use these pins starting from the upper bit.

To use P20/ANI0 to P25/ANI15 as analog input pins, set them in the analog input mode by using the A/D port configuration register (ADPC) and in the input mode by using the PM2 register. Use these pins starting from the lower bit.

Reset signal generation sets port 2 to the analog input mode.

**Table 4-2. Setting Functions of ANI0/P21 and ANI1/P20 Pins**

| ADPC Register | PM2 Register | ADS Register | P20/$AV_{REFP}$/ANI0, P21/$AV_{REFM}$/ANI1, P22/ANI2/IVCMP0/IVREF1, P23/ANI3/IVCMP1/IVREF0, P24/ANI4, and P25/ANI5 Pins |
|---|---|---|---|
| Digital I/O selection | Input mode | × | Digital input |
| | Output mode | × | Digital output |
| Analog input selection | Input mode | Selects ANI. | Analog input (to be converted)<br>(when ANI0 to ANI5 pins are used) |
| | | Does not select ANI. | Analog input (not to be converted)<br>(when IVCMPn and IVREFn pins are used) |
| | Output mode | Selects ANI. | Setting prohibited |
| | | Does not select ANI. | |

**Remark** ×: don't care

### 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 using port mode register 3 (PM3). When the P30 to P37 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by pull-up resistor option register 3 (PU3).

This port can also be used for clock/buzzer output, timer I/O, and segment output of LCD controller/driver.

Reset signal generation sets port 3 to the digital input invalid mode[Note].

**Note** "Digital input invalid" refers to the state in which all the digital outputs, digital inputs, and LCD outputs are disabled.

### 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 using port mode register 4 (PM4). When the P40 to P44 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by pull-up resistor option register 4 (PU4).

This port can also be used for external interrupt request input , clock/buzzer output, timer I/O, and data I/O for a flash memory programmer/debugger.

Reset signal generation sets port 4 to input mode.

### 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 using port mode register 5 (PM5). When the P50 to P57 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by pull-up resistor option register 5 (PU5).

This port can also be used for segment output of LCD controller/driver.

Reset signal generation sets port 5 to the digital input invalid mode[Note].

**Note** "Digital input invalid" refers to the state in which all the digital outputs, digital inputs, and LCD outputs are disabled.

### 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 using port mode register 6 (PM6).

The output of the P60, P61, and P62 pins is N-ch open-drain output (6 V tolerance).

This port can also be used for serial interface data I/O, clock I/O, timer I/O, and real-time clock correction clock output.

Reset signal generation sets port 6 to input mode.

### 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 using port mode register 7 (PM7). When the P70 to P77 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by pull-up resistor option register 7 (PU7).

This port can also be used for segment output of LCD controller/driver and external interrupt request input.

Reset signal generation sets port 7 to the digital input invalid mode[Note].

**Note** "Digital input invalid" refers to the state in which all the digital outputs, digital inputs, and LCD outputs are disabled.

### 4.2.9 Port 8

Port 8 is an I/O port with an output latch. Port 8 can be set to the input mode or output mode in 1-bit units using port mode register 8 (PM8). When the P80 to P85 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by pull-up resistor option register 8 (PU8).

Input to the P81 pin can be specified through a normal input buffer or a TTL input buffer in 1-bit units using port input mode register 8 (PIM8).

Output from the P80 to P82 pins can be specified as N-ch open-drain output ($V_{DD}$ tolerance) in 1-bit units using port output mode register 8 (POM8).

This port can also be used for serial interface data I/O, clock I/O, and segment output of LCD controller/driver.

Reset signal generation sets port 8 to the digital input invalid mode[Note].

**Note** "Digital input invalid" refers to the state in which all the digital outputs, digital inputs, and LCD outputs are disabled.

### 4.2.10 Port 12

P125 to P127 are an I/O port with an output latch. Port 12 can be set to the input mode or output mode in 1-bit units using port mode register 12 (PM12). When the P125 to P127 pins are used as an input port, use of an on-chip pull-up resistor can be specified by pull-up resistor option register 12 (PU12).

P121 to P124 are 4-bit input-only ports.

This port can also be used for connecting resonator for main system clock, connecting resonator for subsystem clock, external clock input for main system clock, external clock input for subsystem clock, connecting a capacitor for LCD controller/driver, power supply voltage pin for driving the LCD, external interrupt request input, and timer I/O.

Reset signal generation sets P121 to P124 to input mode. P125 to P127 are set in the digital invalid mode[Note].

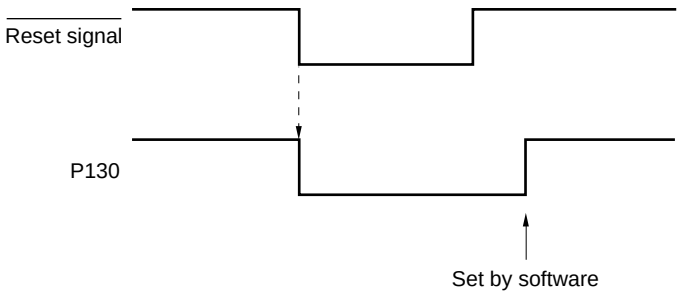
**Note** "Digital input invalid" refers to the state in which all the digital outputs, digital inputs, and LCD outputs are disabled.

### 4.2.11 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 an output mode, and P137 is fixed an input mode.

This port can also be used for real-time clock correction clock output and external interrupt request input.

**Remark** When a reset takes effect, P130 outputs a low-level signal. If P130 is set to output a high-level signal before a reset takes effect, the output signal of P130 can be dummy-output as the CPU reset signal.



## 4.3 Registers Controlling Port Function

Port functions are controlled by the following registers.

- Port mode registers (PMxx)
- Port registers (Pxx)
- Pull-up resistor option registers (PUxx)
- Port input mode registers (PIMxx)
- Port output mode registers (POMxx)
- A/D port configuration register (ADPC)
- Peripheral I/O redirection register (PIOR)
- Global digital input disable register (GDIDIS)
- LCD port function registers (PFSEG0 to PFSEG5)
- LCD input switch control register (ISCLCD)

**Caution Which registers and bits are included depends on the product. For registers and bits mounted on each product, see Table 4-3. Be sure to set bits that are not mounted to their initial values.**

**Table 4-3. PMxx, Pxx, PUxx, PIMxx, POMxx Registers and the Bits Mounted on Each Product (1/3)**

| Port | | Bit Name | | | | | 80 Pin | 100 Pin |
|---|---|---|---|---|---|---|---|---|
| | | PMxx Register | Pxx Register | PUxx Register | PIMxx Register | POMxx Register | | |
| Port 0 | 0 | PM00 | P00 | PU00 | PIM00 | – | √ | √ |
| | 1 | PM01 | P01 | PU01 | – | POM01 | √ | √ |
| | 2 | PM02 | P02 | PU02 | – | POM02 | √ | √ |
| | 3 | PM03 | P03 | PU03 | PIM03 | POM03 | √ | √ |
| | 4 | PM04 | P04 | PU04 | – | POM04 | √ | √ |
| | 5 | PM05 | P05 | PU05 | PIM05 | POM05 | √ | √ |
| | 6 | PM06 | P06 | PU06 | PIM06 | POM06 | √ | √ |
| | 7 | PM07 | P07 | PU07 | – | POM07 | √ | √ |
| Port 1 | 0 | PM10 | P10 | PU10 | – | – | √ | √ |
| | 1 | PM11 | P11 | PU11 | – | – | √ | √ |
| | 2 | PM12 | P12 | PU12 | – | – | √ | √ |
| | 3 | PM13 | P13 | PU13 | – | – | √ | √ |
| | 4 | PM14 | P14 | PU14 | – | – | √ | √ |
| | 5 | PM15 | P15 | PU15 | PIM15 | POM15 | √ | √ |
| | 6 | PM16 | P16 | PU16 | PIM16 | POM16 | √ | √ |
| | 7 | PM17 | P17 | PU17 | – | POM17 | √ | √ |
| Port 2 | 0 | PM20 | P20 | – | – | – | √ | √ |
| | 1 | PM21 | P21 | – | – | – | √ | √ |
| | 2 | PM22 | P22 | – | – | – | √ | √ |
| | 3 | PM23 | P23 | – | – | – | √ | √ |
| | 4 | PM24 | P24 | – | – | – | – | √ |
| | 5 | PM25 | P25 | – | – | – | – | √ |
| | 6 | – | – | – | – | – | – | – |
| | 7 | – | – | – | – | – | – | – |
| Port 3 | 0 | PM30 | P30 | PU30 | – | – | √ | √ |
| | 1 | PM31 | P31 | PU31 | – | – | √ | √ |
| | 2 | PM32 | P32 | PU32 | – | – | √ | √ |
| | 3 | PM33 | P33 | PU33 | – | – | √ | √ |
| | 4 | PM34 | P34 | PU34 | – | – | – | √ |
| | 5 | PM35 | P35 | PU35 | – | – | – | √ |
| | 6 | PM36 | P36 | PU36 | – | – | – | √ |
| | 7 | PM37 | P37 | PU37 | – | – | – | √ |

**Table 4-3. PMxx, Pxx, PUxx, PIMxx, POMxx Registers and the Bits Mounted on Each Product (2/3)**

| Port | | Bit Name | | | | | 80 Pin | 100 Pin |
|---|---|---|---|---|---|---|---|---|
| | | PMxx Register | Pxx Register | PUxx Register | PIMxx Register | POMxx Register | | |
| Port 4 | 0 | PM40 | P40 | PU40 | – | – | √ | √ |
| | 1 | PM41 | P41 | PU41 | – | – | √ | √ |
| | 2 | PM42 | P42 | PU42 | – | – | √ | √ |
| | 3 | PM43 | P43 | PU43 | – | – | √ | √ |
| | 4 | PM44 | P44 | PU44 | – | – | √ | √ |
| | 5 | – | – | – | – | – | – | – |
| | 6 | – | – | – | – | – | – | – |
| | 7 | – | – | – | – | – | – | – |
| Port 5 | 0 | PM50 | P50 | PU50 | – | – | – | √ |
| | 1 | PM51 | P51 | PU51 | – | – | – | √ |
| | 2 | PM52 | P52 | PU52 | – | – | – | √ |
| | 3 | PM53 | P53 | PU53 | – | – | – | √ |
| | 4 | PM54 | P54 | PU54 | – | – | – | √ |
| | 5 | PM55 | P55 | PU55 | – | – | – | √ |
| | 6 | PM56 | P56 | PU56 | – | – | – | √ |
| | 7 | PM57 | P57 | PU57 | – | – | – | √ |
| Port 6 | 0 | PM60 | P60 | – | – | – | √ | √ |
| | 1 | PM61 | P61 | – | – | – | √ | √ |
| | 2 | PM62 | P62 | – | – | – | √ | √ |
| | 3 | – | – | – | – | – | – | – |
| | 4 | – | – | – | – | – | – | – |
| | 5 | – | – | – | – | – | – | – |
| | 6 | – | – | – | – | – | – | – |
| | 7 | – | – | – | – | – | – | – |
| Port 7 | 0 | PM70 | P70 | PU70 | – | – | √ | √ |
| | 1 | PM71 | P71 | PU71 | – | – | √ | √ |
| | 2 | PM72 | P72 | PU72 | – | – | √ | √ |
| | 3 | PM73 | P73 | PU73 | – | – | √ | √ |
| | 4 | PM74 | P74 | PU74 | – | – | √ | √ |
| | 5 | PM75 | P75 | PU75 | – | – | √ | √ |
| | 6 | PM76 | P76 | PU76 | – | – | √ | √ |
| | 7 | PM77 | P77 | PU77 | – | – | √ | √ |

**Table 4-3. PMxx, Pxx, PUxx, PIMxx, POMxx Registers and the Bits Mounted on Each Product (3/3)**

| Port | | Bit Name | | | | | 80 Pin | 100 Pin |
|---|---|---|---|---|---|---|---|---|
| | | PMxx Register | Pxx Register | PUxx Register | PIMxx Register | POMxx Register | | |
| Port 8 | 0 | PM80 | P80 | PU80 | – | POM80 | √ | √ |
| | 1 | PM81 | P81 | PU81 | PIM81 | POM81 | √ | √ |
| | 2 | PM82 | P82 | PU82 | – | POM82 | √ | √ |
| | 3 | PM83 | P83 | PU83 | – | – | √ | √ |
| | 4 | PM84 | P84 | PU84 | – | – | – | √ |
| | 5 | PM85 | P85 | PU85 | – | – | – | √ |
| | 6 | – | – | – | – | – | – | – |
| | 7 | – | – | – | – | – | – | – |
| Port 12 | 0 | – | – | – | – | – | – | – |
| | 1 | – | P121 | – | – | – | √ | √ |
| | 2 | – | P122 | – | – | – | √ | √ |
| | 3 | – | P123 | – | – | – | √ | √ |
| | 4 | – | P124 | – | – | – | √ | √ |
| | 5 | PM125 | P125 | PU125 | – | – | √ | √ |
| | 6 | PM126 | P126 | PU126 | – | – | √ | √ |
| | 7 | PM127 | P127 | PU127 | – | – | √ | √ |
| Port 13 | 0 | – | P130 | – | – | – | √ | √ |
| | 1 | – | – | – | – | – | – | – |
| | 2 | – | – | – | – | – | – | – |
| | 3 | – | – | – | – | – | – | – |
| | 4 | – | – | – | – | – | – | – |
| | 5 | – | – | – | – | – | – | – |
| | 6 | – | – | – | – | – | – | – |
| | 7 | – | P137 | – | – | – | √ | √ |

### 4.3.1 Port mode registers (PMxx)

These registers specify input or output mode for the port in 1-bit units.

These registers can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets these registers to FFH.

When port pins are used as alternate-function pins, set the port mode register by referencing **4.5 Register Settings When Using Alternate Function**.

**Figure 4-1. Format of Port Mode Register**

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset | R/W |
|---|---|---|---|---|---|---|---|---|---|---|---|
| PM0 | PM07 | 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 | 1 | 1 | PM25 | PM24 | PM23 | PM22 | PM21 | PM20 | FFF22H | FFH | R/W |
| PM3 | PM37 | PM36 | PM35 | PM34 | PM33 | PM32 | PM31 | PM30 | FFF23H | FFH | R/W |
| PM4 | 1 | 1 | 1 | PM44 | PM43 | PM42 | PM41 | PM40 | FFF24H | FFH | R/W |
| PM5 | PM57 | PM56 | PM55 | PM54 | PM53 | PM52 | PM51 | PM50 | FFF25H | FFH | R/W |
| PM6 | 1 | 1 | 1 | 1 | 1 | PM62 | PM61 | PM60 | FFF26H | FFH | R/W |
| PM7 | PM77 | PM76 | PM75 | PM74 | PM73 | PM72 | PM71 | PM70 | FFF27H | FFH | R/W |
| PM8 | 1 | 1 | PM85 | PM84 | PM83 | PM82 | PM81 | PM80 | FFF28H | FFH | R/W |
| PM12 | PM127 | PM126 | PM125 | 1 | 1 | 1 | 1 | 1 | FFF2CH | FFH | R/W |

| PMmn | Pmn pin I/O mode selection<br>(m = 0 to 8, 12; 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 mounted to their initial values.**

### 4.3.2 Port registers (Pxx)

These registers set the output latch value of a port.

If the data is read in the input mode, the pin level is read. If it is read in the output mode, the output latch value is read[Note].

These registers can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H.

**Note** If P20 to P25 are set up as analog inputs of the A/D converter or comparator, when a port is read while in the input mode, 0 is always returned, not the pin level.

**Figure 4-2. Format of Port Register**

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset | R/W |
|---|---|---|---|---|---|---|---|---|---|---|---|
| P0 | P07 | 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 | 0 | 0 | P25 | P24 | P23 | P22 | P21 | P20 | FFF02H | 00H (output latch) | R/W |
| P3 | P37 | P36 | P35 | P34 | P33 | P32 | P31 | P30 | FFF03H | 00H (output latch) | R/W |
| P4 | 0 | 0 | 0 | P44 | P43 | P42 | P41 | P40 | FFF04H | 00H (output latch) | R/W |
| P5 | P57 | P56 | P55 | P54 | P53 | P52 | P51 | P50 | FFF05H | 00H (output latch) | R/W |
| P6 | 0 | 0 | 0 | 0 | 0 | P62 | P61 | P60 | FFF06H | 00H (output latch) | R/W |
| P7 | P77 | P76 | P75 | P74 | P73 | P72 | P71 | P70 | FFF07H | 00H (output latch) | R/W |
| P8 | 0 | 0 | P85 | P84 | P83 | P82 | P81 | P80 | FFF08H | 00H (output latch) | R/W |
| P12 | P127 | P126 | P125 | P124 | P123 | P122 | P121 | 0 | FFF0CH | Undefined | R/W[Note] |
| P13 | P137 | 0 | 0 | 0 | 0 | 0 | 0 | P130 | FFF0DH | Undefined | R/W[Note] |

| 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** P121 to P124 and P137 are read-only.

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

**Remark** m = 0 to 8, 12, 13 ; n = 0 to 7

### 4.3.3 Pull-up resistor option registers (PUxx)

These registers specify whether the on-chip pull-up resistors are to be used or not. On-chip pull-up resistors can be used in 1-bit units only for the bits set to 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 cannot be connected to bits set to output mode and bits used as alternate-function output pins and analog setting (ADPC = 1), regardless of the settings of these registers.

These registers can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H (Only PU4 is set to 01H).

**Caution When a port with the PIMn register is input from different potential device to TTL buffer, pull up to the power supply of the different potential device via an external pull-up resistor by setting PUmn = 0.**

**Figure 4-3. Format of Pull-up Resistor Option Register**

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset | R/W |
|---|---|---|---|---|---|---|---|---|---|---|---|
| PU0 | PU07 | 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 | PU37 | PU36 | PU35 | PU34 | PU33 | PU32 | PU31 | PU30 | F0033H | 00H | R/W |
| PU4 | 0 | 0 | 0 | PU44 | PU43 | PU42 | PU41 | PU40 | F0034H | 01H | R/W |
| PU5 | PU57 | PU56 | PU55 | PU54 | PU53 | PU52 | PU51 | PU50 | F0035H | 00H | R/W |
| PU7 | PU77 | PU76 | PU75 | PU74 | PU73 | PU72 | PU71 | PU70 | F0037H | 00H | R/W |
| PU8 | 0 | 0 | PU85 | PU84 | PU83 | PU82 | PU81 | PU80 | F0038H | 00H | R/W |
| PU12 | PU127 | PU126 | PU125 | 0 | 0 | 0 | 0 | 0 | F003CH | 00H | R/W |

| PUmn | Pmn pin on-chip pull-up resistor selection<br>(m = 0, 1, 3 to 5, 7, 8, 12 ; n = 0 to 7) |
|---|---|
| 0 | On-chip pull-up resistor not connected |
| 1 | On-chip pull-up resistor connected |

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

### 4.3.4 Port input mode registers (PIMxx)

These registers set the input buffer in 1-bit units.

TTL input buffer can be selected during serial communication with an external device of the different potential.

Port input mode registers can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H.

**Figure 4-4. Format of Port Input Mode Register**

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset | R/W |
|---|---|---|---|---|---|---|---|---|---|---|---|
| PIM0 | 0 | PIM06 | PIM05 | 0 | PIM03 | 0 | 0 | PIM00 | F0040H | 00H | R/W |
| PIM1 | 0 | PIM16 | PIM15 | 0 | 0 | 0 | 0 | 0 | F0041H | 00H | R/W |
| PIM8 | 0 | 0 | 0 | 0 | 0 | 0 | PIM81 | 0 | F0048H | 00H | R/W |

| PIMmn | Pmn pin input buffer selection<br>(m = 0, 1, 8 ; n = 0, 1, 3, 5, 6) |
|---|---|
| 0 | Normal input buffer |
| 1 | TTL input buffer |

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

### 4.3.5 Port output mode registers (POMxx)

These registers set the output mode in 1-bit units.

N-ch open drain output ($V_{DD}$ tolerance) mode can be selected during serial communication with an external device of the different potential, and for the SDA00 and SDA10 pins during simplified $I^2C$ communication with an external device of the same potential.

In addition, POMxx register is set with PUxx register, whether or not to use the on-chip pull-up resistor.

These registers can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H.

**Caution An on-chip pull-up resistor is not connected to a bit for which N-ch open drain output ($V_{DD}$ tolerance[Note 1]/$EV_{DD1}$ tolerance[Note 2]) mode (POMmn = 1) is set.**

**Figure 4-5. Format of Port Input Mode Register**

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset | R/W |
|---|---|---|---|---|---|---|---|---|---|---|---|
| POM0 | POM07 | POM06 | POM05 | POM04 | POM03 | POM02 | POM01 | 0 | F0050H | 00H | R/W |
| POM1 | POM17 | POM16 | POM15 | 0 | 0 | 0 | 0 | 0 | F0051H | 00H | R/W |
| POM8 | 0 | 0 | 0 | 0 | 0 | POM82 | POM81 | POM80 | F0058H | 00H | R/W |

| POMmn | Pmn pin output mode selection (m = 0, 1, 8 ; n = 0 to 7) |
|---|---|
| 0 | Normal output mode |
| 1 | N-ch open-drain output ($V_{DD}$ tolerance[Note 1]/$EV_{DD1}$ tolerance[Note 2]) mode |

**Notes 1.** For 80-pin products

**2.** For 100-pin products

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

### 4.3.6 A/D port configuration register (ADPC)

This register switches the P20/AV$_{REFP}$/ANI0, P21/AV$_{REFM}$/ANI1, P22/ANI2/IVCMP0/IVREF1, P23/ANI3/IVCMP1/IVREF0, P24/ANI4, and P25/ANI5 pins to digital I/O of port or analog input of A/D converter or comparator.

The ADPC register can be set by an 8-bit memory manipulation instruction.

Reset signal generation sets this register to 00H.

**Figure 4-6. Format of A/D Port Configuration Register (ADPC)**

Address: F0076H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ADPC | 0 | 0 | 0 | 0 | 0 | ADPC2 | ADPC1 | ADPC0 |

| ADPC2 | ADPC1 | ADPC0 | Analog input (A)/digital I/O (D) switching | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | ANI5/P25 | ANI4/P24 | ANI3/IVCMP1 /IVREF0/P23 | ANI2/IVCMP0 /IVREF1/P22 | ANI1/P21 | ANI0/P20 |
| 0 | 0 | 0 | A | A | A | A | A | A |
| 0 | 0 | 1 | D | D | D | D | D | D |
| 0 | 1 | 0 | D | D | D | D | D | A |
| 0 | 1 | 1 | D | D | D | D | A | A |
| 1 | 0 | 0 | D | D | D | A | A | A |
| 1 | 0 | 1 | D | D | A | A | A | A |
| 1 | 1 | 0 | D | A | A | A | A | A |
| Other than above | | | Setting prohibited | | | | | |

**Cautions 1. Set the port to analog input by ADPC register to the input mode by using port mode register 2 (PM2).**

**2. Do not set the pin set by the ADPC register as digital I/O by the analog input channel specification register (ADS).**

**3. When using AV$_{REFP}$ and AV$_{REFM}$, set ANI0 and ANI1 to analog input and set the port mode register to the input mode.**

### 4.3.7 Global digital input disable register (GDIDIS)

This register is used to prevent through-current flowing from the input buffers when the battery backup function is enabled and power supply to $V_{DD}$ and $EV_{DD}$ is stopped.

By setting the GDIDIS0 bit to 1, input to any input buffer connected to $EV_{DD}$ is prohibited, preventing through-current from flowing when the power supply connected to $EV_{DD}$ is turned off.

The GDIDIS register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to 00H.

**Figure 4-7.** Format of **Global Digital Input Disable Register (GDIDIS)**

Address: F007DH After reset: 00H 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 input buffers permitted (default) |
| 1 | Input to input buffers prohibited. No through-current flows to the input buffers. |

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

1. Prohibit 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. Permit input to input buffers (set GDIDIS0 = 0).

**Cautions 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.**

**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 the port output mode register (POMxx) (N-ch open drain output ($EV_{DD}$ tolerance) mode), the value read from the port register (Pxx) is 0.**

**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.

### 4.3.8 Peripheral I/O redirection register (PIOR)

This register is used to specify whether to enable or disable the peripheral I/O redirect function.

This function is used to switch ports to which alternate functions are assigned.

Use the PIOR register to assign a port to the function to redirect and enable the function.

In addition, can be changed the settings for redirection until its function enable operation.

The PIOR register can be set by an 8-bit memory manipulation instruction.

Reset signal generation sets this register to 00H.

**Figure 4-8. Format of Peripheral I/O Redirection Register (PIOR)**

Address: F0077H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PIOR | 0 | 0 | 0 | PIOR4 | PIOR3 | PIOR2 | PIOR1 | PIOR0 |

| Bit | Function | 80-pin | | 100-pin | |
|---|---|---|---|---|---|
| | | Setting value | | Setting value | |
| | | 0 | 1 | 0 | 1 |
| PIOR4 | INTP0[Note] | P137 | P70 | P137 | P70 |
| | INTP1 | P125 | P71 | P125 | P71 |
| | INTP2 | P07 | P72 | P07 | P72 |
| | INTP3 | P05 | P73 | P05 | P73 |
| | INTP4 | P04 | P74 | P04 | P74 |
| | INTP5 | P02 | P75 | P02 | P75 |
| | INTP6 | P44 | P76 | P44 | P76 |
| | INTP7 | P42 | P77 | P42 | P77 |
| PIOR3 | PCLBUZ0 | P43 | P33 | P43 | P33 |
| | PCLBUZ1 | P41 | P32 | P41 | P32 |
| | VCOUT0 | P00 | P03 | P00 | P03 |
| | VCOUT1 | P01 | P04 | P01 | P04 |
| | RTC1HZ | P130 | P62 | P130 | P62 |
| PIOR2 | TxD1 | P04 | P82 | P04 | P82 |
| | RxD1 | P03 | P81 | P03 | P81 |
| | SCL10 | P02 | P80 | P02 | P80 |
| | SDA10 | P03 | P81 | P03 | P81 |
| PIOR1 | TxD0 | P07 | P17 | P07 | P17 |
| | RxD0 | P06 | P16 | P06 | P16 |
| | SCL00 | P05 | P15 | P05 | P15 |
| | SDA00 | P06 | P16 | P06 | P16 |
| | SI00 | P06 | P16 | P06 | P16 |
| | SO00 | P07 | P17 | P07 | P17 |
| | SCK00 | P05 | P15 | P05 | P15 |
| PIOR0 | TI00/TO00 | P43 | P60 | P43 | P60 |
| | TI01/TO01 | P41 | P61 | P41 | P61 |
| | TI02/TO02 | P07 | P62 | P07 | P62 |
| | TI03/TO03 | P06 | P127 | P06 | P127 |
| | TI04/TO04 | P05 | P126 | P05 | P126 |
| | TI05/TO05 | P04 | P125 | P04 | P125 |
| | TI06/TO06 | P03 | P31 | P03 | P31 |
| | TI07/TO07 | P02 | P30 | P02 | P30 |

**Note** Uses a battery backup function and the P137 pin is enabled when supplying power from VBAT.
When the INTP0 function is assigned to P70, note that the interrupt function is disabled when supplying power from VBAT.

### 4.3.9 LCD port function registers 0 to 5 (PFSEG0 to PFSEG5)

These registers set whether to use pins P10 to P17, P30 to P37, P50 to P57, P70 to P77, P80 to P85 as port pins (other than segment output pins) or segment output pins.

These registers can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets these registers to FFH (PFSEG0 is set to F0H, and PFSEG5 is set to 03H).

**Remark** The correspondence between the segment output pins (SEGxx) and the PFSEG register (PFSEGxx bits) and the existence of SEGxx pins in each product are shown in Table 4-4 Segment Output Pins in Each Product and Correspondence with PFSEG Register (PFSEG Bits).

<R> **Figure 4-9.** Format of **LCD port function registers 0 to 5 (PFSEG0 to PFSEG5)**

Address: F0300H After reset: F0H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFSEG0 | PFSEG07 | PFSEG06 | PFSEG05 | PFSEG04 | 0 | 0 | 0 | 0 |

Address: F0301H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFSEG1 | PFSEG15 | PFSEG14 | PFSEG13 | PFSEG12 | PFSEG11 | PFSEG10 | PFSEG09 | PFSEG08 |

Address: F0302H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFSEG2 | PFSEG23 | PFSEG22 | PFSEG21 | PFSEG20 | PFSEG19 | PFSEG18 | PFSEG17 | PFSEG16 |

Address: F0303H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFSEG3 | PFSEG31 Note | PFSEG30 Note | PFSEG29 Note | PFSEG28 Note | PFSEG27 | PFSEG26 | PFSEG25 | PFSEG24 |

Address: F0304H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFSEG4 | PFSEG39 Note | PFSEG38 Note | PFSEG37 | PFSEG36 | PFSEG35 | PFSEG34 | PFSEG33 | PFSEG32 |

Address: F0305H After reset: 03H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFSEG5 | 0 | 0 | 0 | 0 | 0 | 0 | PFSEG41 Note | PFSEG40 Note |

| PFSEGxx (xx = 04 to 41) | Port (other than segment output)/segment outputs specification of Pmn pins (mn = 10 to 17, 30 to 37, 50 to 57, 70 to 77, 80 to 85) |
|---|---|
| 0 | Used the Pmn pin as port (other than segment output) |
| 1 | Used the Pmn pin as segment output |

<R> **Note** Be sure to set "1" for 80-pin products.

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

**Remark** To use the Pmn pins as segment output pins (PFSEGxx = 1), be sure to set the PUmn bit of the PUm register, POMmn bit of the POMm register, and PIMmn bit of the PIMm register to “0”.

**Table 4-4. Segment Output Pins in Each Product and Correspondence with PFSEG Register (PFSEG Bits)**

| Bit Name of PFSEG Register | Corresponding SEGxx Pins | Alternate Port | 80-pin | 100-pin |
|---|---|---|---|---|
| PFSEG04 | SEG4 | P10 | √ | √ |
| PFSEG05 | SEG5 | P11 | √ | √ |
| PFSEG06 | SEG6 | P12 | √ | √ |
| PFSEG07 | SEG7 | P13 | √ | √ |
| PFSEG08 | SEG8 | P14 | √ | √ |
| PFSEG09 | SEG9 | P15 | √ | √ |
| PFSEG10 | SEG10 | P16 | √ | √ |
| PFSEG11 | SEG11 | P17 | √ | √ |
| PFSEG12 | SEG12 | P80 | √ | √ |
| PFSEG13 | SEG13 | P81 | √ | √ |
| PFSEG14 | SEG14 | P82 | √ | √ |
| PFSEG15 | SEG15 | P83 | √ | √ |
| PFSEG16 | SEG16 | P70 | √ | √ |
| PFSEG17 | SEG17 | P71 | √ | √ |
| PFSEG18 | SEG18 | P72 | √ | √ |
| PFSEG19 | SEG19 | P73 | √ | √ |
| PFSEG20 | SEG20 | P74 | √ | √ |
| PFSEG21 | SEG21 | P75 | √ | √ |
| PFSEG22 | SEG22 | P76 | √ | √ |
| PFSEG23 | SEG23 | P77 | √ | √ |
| PFSEG24 | SEG24 | P30 | √ | √ |
| PFSEG25 | SEG25 | P31 | √ | √ |
| PFSEG26 | SEG26 | P32 | √ | √ |
| PFSEG27 | SEG27 | P33 | √ | √ |
| PFSEG28 | SEG28 | P34 | – | √ |
| PFSEG29 | SEG29 | P35 | – | √ |
| PFSEG30 | SEG30 | P36 | – | √ |
| PFSEG31 | SEG31 | P37 | – | √ |
| PFSEG32 | SEG32 | 80-pin products: P02<br>100-pin products: P50 | √ | √ |
| PFSEG33 | SEG33 | 80-pin products: P03<br>100-pin products: P51 | √ | √ |
| PFSEG34 | SEG34 | 80-pin products: P04<br>100-pin products: P52 | √ | √ |
| PFSEG35 | SEG35 | 80-pin products: P05<br>100-pin products: P53 | √ | √ |
| PFSEG36 | SEG36 | 80-pin products: P06<br>100-pin products: P54 | √ | √ |
| PFSEG37 | SEG37 | 80-pin products: P07<br>100-pin products: P55 | √ | √ |
| PFSEG38 | SEG38 | P56 | – | √ |
| PFSEG39 | SEG39 | P57 | – | √ |
| PFSEG40 | SEG40 | P84 | – | √ |
| PFSEG41 | SEG41 | P85 | – | √ |

### 4.3.10 LCD input switch control register (ISCLCD)

This register sets whether to use pins P125 to P127 as port pins (other than LCD function pins) or LCD function pins ($V_{L3}$, CAPL, CAPH).

The ISCLCD register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to 00H.

**Figure 4-10.** Format of **LCD input switch control register (ISCLCD)**

Address: F0308H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ISCLCD | 0 | 0 | 0 | 0 | 0 | 0 | ISCVL3 | ISCCAP |

| ISCVL3 | Control of schmitt trigger buffer of $V_{L3}$/P125 pin |
|---|---|
| 0 | Makes digital input invalid (used as LCD function pin ($V_{L3}$)) |
| 1 | Makes digital input valid |

| ISCCAP | Control of schmitt trigger buffer of CAPL/ P126 and CAPH/P127 pins |
|---|---|
| 0 | Makes digital input invalid (used as LCD function pins (CAPL,CAPH)) |
| 1 | Makes digital input valid |

**Caution If ISCVL3 bit = 0 and ISCCAP bit = 0, set the corresponding port control registers as follows:**

**PU127 bit of PU12 register = 0, P127 bit of P12 register = 0**

**PU126 bit of PU12 register = 0, P126 bit of P12 register = 0**

**PU125 bit of PU12 register = 0, P125 bit of P12 register = 0**

## 4.4 Port Function Operations

Port operations differ depending on whether the input or output mode is set, 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 status 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 status 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 its contents. The result of the operation is written to the output latch, but since the output buffer is off, the pin status 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 Connecting to external device with different potential (1.8 V, 2.5 V, 3 V)

When connecting an external device operating on a different potential (1.8 V, 2.5 V or 3 V), it is possible to connect the I/O pins of general ports by changing $EV_{DD}$ to accord with the power supply of the connected device.

### 4.4.5 Handling different potential (1.8 V, 2.5 V, 3 V) by using I/O buffers

It is possible to connect an external device operating on a different potential (1.8 V, 2.5 V or 3 V) by switching I/O buffers with port input mode registers 0, 1, and 8 (PIM0, PIM1, and PIM8) and port output mode registers 0, 1, and 8 (POM0, POM1, and POM8).

When receiving input from an external device with a different potential (1.8 V, 2.5 V or 3 V), set port input mode registers 0, 1, and 8 (PIM0, PIM1, and PIM8) on a bit-by-bit basis to enable normal input (CMOS)/TTL input buffer switching.

When outputting data to an external device with a different potential (1.8 V, 2.5 V or 3 V), set port output mode registers 0, 1, and 8 (POM0, POM1, and POM8) on a bit-by-bit basis to enable normal output (CMOS)/N-ch open drain ($V_{DD}$ tolerance[Note 1]/$EV_{DD}$ tolerance[Note 2]) switching.

The connection of a serial interface is described in the following.

**Notes 1.** For 80-pin products
**2.** For 100-pin products

#### (1) Setting procedure when using input pins of UART0 to UART2, and CSI00 functions for the TTL input buffer

In case of UART0: P06 (P16)
In case of UART1: P03 (P81)
In case of UART2: P00
In case of CSI00: P05, P06 (P15, P16)

**Remark** Functions in parentheses can be assigned via settings in the peripheral I/O redirection register (PIOR).

<1> Using an external resistor, pull up externally the pin to be used to the power supply of the target device (on-chip pull-up resistor cannot be used).
<2> Set the corresponding bit of the PIM0, PIM1, and PIM8 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/CSI mode.

**(2) Setting procedure when using output pins of UART0 to UART2, and CSI00 functions in N-ch open-drain output mode**

In case of UART0: P07 (P17)
In case of UART1: P04 (P82)
In case of UART2: P01
In case of CSI00: P05, P07 (P15, P17)

**Remark** Functions in parentheses can be assigned via settings in the peripheral I/O redirection register (PIOR).

<1> Using an external resistor, pull up externally the pin to be used to the power supply of the target device (on-chip pull-up resistor cannot be used).
<2> After reset release, the port mode changes to the input mode (Hi-Z).
<3> Set the output latch of the corresponding port to 1.
<4> Set the corresponding bit of the POM0, POM1, and POM8 registers to 1 to set the N-ch open drain output ($V_{DD}$ tolerance[Note 1]/$EV_{DD}$ tolerance[Note 2]) mode.
<5> Enable the operation of the serial array unit and set the mode to the UART/CSI mode.
<6> Set the output mode by manipulating the PM0, PM1, and PM8 registers.
At this time, the output data is high level, so the pin is in the Hi-Z state.

**Notes 1.** For 80-pin products
**2.** For 100-pin products

**(3) Setting procedure when using I/O pins of IIC00 and IIC10 functions with a different potential (1.8 V, 2.5 V, 3 V)**

In case of IIC00: P05, P06 (P15, P16)
In case of IIC10: P02, P03 (P80, P81)

**Remark** Functions in parentheses can be assigned via settings in the peripheral I/O redirection register (PIOR).

<1> Using an external resistor, pull up externally the pin to be used to the power supply of the target device (on-chip pull-up resistor cannot be used).
<2> After reset release, the port mode is the input mode (Hi-Z).
<3> Set the output latch of the corresponding port to 1.
<4> Set the corresponding bit of the POM0, POM1, and POM8 registers to 1 to set the N-ch open drain output ($V_{DD}$ tolerance[Note 1]/$EV_{DD}$ tolerance[Note 2]) mode.
<5> Set the corresponding bit of the PIM0, PIM1, and PIM8 registers to 1 to switch 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, PM1, and PM8 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.

**Notes 1.** For 80-pin products
**2.** For 100-pin products

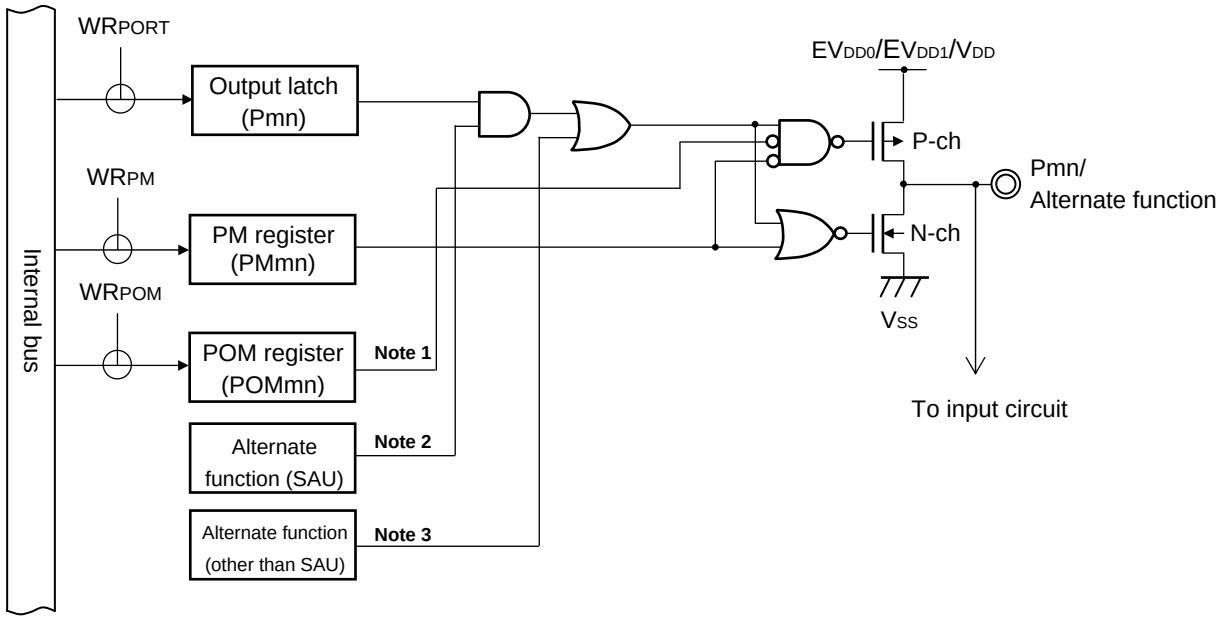
## 4.5 Register Settings When Using Alternate Function

### 4.5.1 Basic concept when using alternate function

In the beginning, for a pin also assigned to be used for analog input, use the ADPC register to specify whether to use the pin for analog input or digital input/output.

Figure 4-11 shows the basic configuration of an output circuit for pins used for digital input/output. The output of the output latch for the port and the output of the alternate SAU function are input to an AND gate. The output of the AND gate is input to an OR gate. The output of an alternate function other than SAU (TAU, RTC2, clock/buzzer output, IICA, etc.) is connected to the other input pin of the OR gate. When such kind of pins are used by the port function or an alternate function, the unused alternate function must not hinder the output of the function to be used. An idea of basic settings for this kind of case is shown in Table 4-5.

**Figure 4-11. Basic Configuration of Output Circuit for Pins**




**Notes 1.** When there is no POM register, this signal should be considered to be low level (0).

**2.** When there is no alternate function, this signal should be considered to be high level (1).

**3.** When there is no alternate function, this signal should be considered to be low level (0).

**Remark** m: Port number (m = 0 to 8, 12, 13); n: Bit number (n = 0 to 7)

**Table 4-5. Concept of Basic Settings**

| Output Function of Used Pin | Output Settings of Unused Alternate Function | | |
|---|---|---|---|
| | Port Function | Output Function for SAU | Output Function for other than SAU |
| Output function for port | – | Output is high (1) | Output is low (0) |
| Output function for SAU | High (1) | – | Output is low (0) |
| Output function for other than SAU | Low (0) | Output is high (1) | Output is low (0)Note |

**Note** Since more than one output function other than SAU may be assigned to a single pin, the output of an unused alternate function must be set to low level (0). 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 the target, the output can be switched to another pin by setting the peripheral I/O redirection register (PIOR). 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). 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). 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 bit in timer output register 0 (TO0) to 0 (low). 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) PCLBUZn = 0 (setting when clock/buzzer output is not used)
When the clock/buzzer output is not used, set the PCLOEn bit in clock output select register n (CKSn) to 0 (output disabled). This is the same setting as the initial state.

(6) VCOUTn = 0 (setting when VCOUTn is not used)
When VCOUTn of comparator is not used, set the bits 5 and 1 in the comparator output control register (COMPOCR) to 0 (VCOUTn pin of comparator n output disabled). This is the same setting as the initial state.

### 4.5.3 Register setting examples for used port and alternate functions

Register setting examples for used port and alternate functions are shown in Table 4-6. The registers used to control the port functions should be set as shown in Table 4-6. See the following remark for legends used in Table 4-6.

**Remark** –: Not supported
×: don't care
PIORx: Peripheral I/O redirection register
POMxx: Port output mode register
PMxx: Port mode register
Pxx: Port output latch
PUxx: Pull-up resistor option register
PIMcc: Port input mode register
PFSEG××: LCD port function register
ISCLCD: LCD input switch control register

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (1/9)**

| Pin Name | Used Function | | PIOR× | POM×× | PM×× | P×× | PFSEG×× (ISCVL3, ISCCAP)[Note] | Alternate Function Output | | 80-pin | 100-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | SAU Output Function | Other than SAU | | |
| P00 | P00 | Input | – | – | 1 | × | 0 | – | – | √ | √ |
| | | Output | × | – | 0 | 0/1 | 0 | – | – | | |
| | RxD2 | Input | × | – | 1 | × | 0 | – | – | | |
| | IrRxD | Input | × | – | 1 | × | 0 | – | – | | |
| | VCOUT0 | Analog output | PIOR3 = 0 | – | 0 | 0 | 0 | – | – | | |
| P01 | P01 | Input | – | × | 1 | × | 0 | – | – | √ | √ |
| | | Output | × | 0 | 0 | 0/1 | 0 | TxD2/IrTxD = 1 | – | | |
| | | N-ch open drain output | × | 1 | 0 | 0/1 | 0 | | | | |
| | TxD2 | Output | × | 0/1 | 0 | 1 | 0 | – | – | | |
| | IrTxD | Output | × | 0/1 | 0 | 1 | 0 | – | – | | |
| | VCOUT1 | Analog output | PIOR3 = 0 | 0 | 0 | 0 | 0 | TxD2/IrTxD = 1 | – | | |
| P02 | P02 | Input | – | × | 1 | × | 0 | – | – | √ | √ |
| | | Output | × | 0 | 0 | 0/1 | 0 | SCL10 = 1 | TO07 = 0 | | |
| | | N-ch open drain output | × | 1 | 0 | 0/1 | 0 | | | | |
| | SCL10 | Output | PIOR2 = 0 | 0/1 | 0 | 1 | 0 | – | TO07 = 0 | | |
| | TI07 | Input | PIOR0 = 0 | × | 1 | × | 0 | – | – | | |
| | TO07 | Output | PIOR0 = 0 | 0 | 0 | 0 | 0 | SCL10=1 | – | | |
| | INTP5 | Input | PIOR4 = 0 | × | 1 | × | 0 | – | – | | |
| | SEG32 | Output | × | 0 | 0 | 0 | 1 | – | – | √ | × |
| P03 | P03 | Input | – | × | 1 | × | 0 | – | – | √ | √ |
| | | Output | × | 0 | 0 | 0/1 | 0 | SDA10 = 1 | TO06 = 0 | | |
| | | N-ch open drain output | × | 1 | 0 | 0/1 | 0 | | | | |
| | RxD1 | Input | PIOR2 = 0 | × | 1 | × | 0 | – | – | | |
| | TI06 | Input | PIOR0 = 0 | × | 1 | × | 0 | – | – | | |
| | TO06 | Output | PIOR0 = 0 | 0 | 0 | 0 | 0 | SDA10 = 1 | – | | |
| | SDA10 | I/O | PIOR2 = 0 | 1 | 0 | 1 | 0 | – | TO06 = 0 | | |
| | (VCOUT0) | Analog output | PIOR3 = 1 | 0 | 0 | 0 | 0 | – | – | | |
| | SEG33 | Output | × | 0 | 0 | 0 | 1 | – | – | √ | × |
| P04 | P04 | Input | – | × | 1 | × | 0 | – | – | √ | √ |
| | | Output | × | 0 | 0 | 0/1 | 0 | TxD1 = 1 | TO05 = 0 | | |
| | | N-ch open drain output | × | 1 | 0 | 0/1 | 0 | | | | |
| | TxD1 | Output | PIOR2 = 0 | 0/1 | 0 | 1 | 0 | – | TO05 = 0 | | |
| | TI05 | Input | PIOR0 = 0 | × | 1 | × | 0 | – | – | | |
| | TO05 | Output | PIOR0 = 0 | 0 | 0 | 0 | 0 | TxD1 = 1 | – | | |
| | INTP4 | Input | PIOR4 = 0 | × | 1 | × | 0 | – | – | | |
| | (VCOUT1) | Analog output | PIOR3 = 1 | 0 | 0 | 0 | 0 | – | – | | |
| | SEG34 | Output | × | 0 | 0 | 0 | 1 | – | – | √ | × |
| P05 | P05 | Input | – | × | 1 | × | 0 | – | – | √ | √ |
| | | Output | × | 0 | 0 | 0/1 | 0 | SCK00/SCL00 = 1 | TO04 = 0 | | |
| | | N-ch open drain output | × | 1 | 0 | 0/1 | 0 | | | | |
| | SCK00 | Input | PIOR1 = 0 | × | 1 | × | 0 | – | – | | |
| | | Output | | 0/1 | 0 | 1 | 0 | – | TO04 = 0 | | |
| | SCL00 | Output | PIOR1 = 0 | 0/1 | 0 | 1 | 0 | – | TO04 = 0 | | |
| | TI04 | Input | PIOR0 = 0 | × | 1 | × | 0 | – | – | | |
| | TO04 | Output | PIOR0 = 0 | 0 | 0 | 0 | 0 | SCK00/SCL00 = 1 | – | | |
| | INTP3 | Input | PIOR4 = 0 | × | 1 | × | 0 | – | – | | |
| | SEG35 | Output | × | 0 | 0 | 0 | 1 | – | – | √ | × |

**Note** ISCVL3 and ISCCAP are registers that correspond to P125, and P126 and P127, respectively.

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (2/9)**

| Pin Name | Used Function | | PIOR× | POM×× | PM×× | P×× | PFSEG×× (ISCVL3, ISCCAP)[Note] | Alternate Function Output | | 80-pin | 100-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | SAU Output Function | Other than SAU | | |
| P06 | P06 | Input | – | × | 1 | × | 0 | – | – | √ | √ |
| | | Output | × | 0 | 0 | 0/1 | 0 | SDA00 = 1 | TO03 = 0 | | |
| | | N-ch open drain output | × | 1 | 0 | 0/1 | 0 | | | | |
| | SI00 | Input | PIOR1 = 0 | × | 1 | × | 0 | – | – | | |
| | RxD0 | Input | PIOR1 = 0 | × | 1 | × | 0 | – | – | | |
| | TI03 | Input | PIOR0 = 0 | × | 1 | × | 0 | – | – | | |
| | TO03 | Output | PIOR0 = 0 | 0 | 0 | 0 | 0 | SDA00 = 1 | – | | |
| | SDA00 | I/O | PIOR1 = 0 | 1 | 0 | 1 | 0 | – | TO03 = 0 | | |
| | TOOLRxD | Input | × | × | 1 | × | 0 | – | – | | |
| | SEG36 | Output | × | 0 | 0 | 0 | 1 | – | – | √ | × |
| P07 | P07 | Input | – | × | 1 | × | 0 | – | – | √ | √ |
| | | Output | × | 0 | 0 | 0/1 | 0 | SO00/TxD0 = 1 | TO02 = 0 | | |
| | | N-ch open drain output | × | 1 | 0 | 0/1 | 0 | | | | |
| | SO00 | Output | PIOR1 = 0 | 0/1 | 0 | 1 | 0 | – | TO02 = 0 | | |
| | TxD0 | Output | PIOR1 = 0 | 0/1 | 0 | 1 | 0 | – | TO02 = 0 | | |
| | TI02 | Input | PIOR0 = 0 | × | 1 | × | 0 | – | – | | |
| | TO02 | Output | PIOR0 = 0 | 0 | 0 | 0 | 0 | SO00/TxD0 = 1 | – | | |
| | INTP2 | Input | PIOR4 = 0 | × | 1 | × | 0 | – | – | | |
| | TOOLTxD | Output | × | 0/1 | 0 | 1 | 0 | – | – | | |
| | SEG37 | Output | × | 0 | 0 | 0 | 1 | – | – | √ | × |
| P10 | P10 | Input | – | – | 1 | × | 0 | – | – | √ | √ |
| | | Output | × | – | 0 | 0/1 | 0 | – | – | | |
| | SEG4 | Output | × | – | 0 | 0 | 1 | – | – | | |
| P11 | P11 | Input | – | – | 1 | × | 0 | – | – | √ | √ |
| | | Output | × | – | 0 | 0/1 | 0 | – | – | | |
| | SEG5 | Output | × | – | 0 | 0 | 1 | – | – | | |
| P12 | P12 | Input | – | – | 1 | × | 0 | – | – | √ | √ |
| | | Output | × | – | 0 | 0/1 | 0 | – | – | | |
| | SEG6 | Output | × | – | 0 | 0 | 1 | – | – | | |
| P13 | P13 | Input | – | – | 1 | × | 0 | – | – | √ | √ |
| | | Output | × | – | 0 | 0/1 | 0 | – | – | | |
| | SEG7 | Output | × | – | 0 | 0 | 1 | – | – | | |
| P14 | P14 | Input | – | – | 1 | × | 0 | – | – | √ | √ |
| | | Output | × | – | 0 | 0/1 | 0 | – | – | | |
| | SEG8 | Output | × | – | 0 | 0 | 1 | – | – | | |
| P15 | P15 | Input | – | × | 1 | × | 0 | – | – | √ | √ |
| | | Output | × | 0 | 0 | 0/1 | 0 | (SCK00/SCL00) = 1 | – | | |
| | | N-ch open drain output | × | 1 | 0 | 0/1 | 0 | | | | |
| | SEG9 | Output | × | 0 | 0 | 0 | 1 | – | – | | |
| | (SCK00) | Input | PIOR1 = 1 | × | 1 | × | 0 | – | – | | |
| | | Output | PIOR1 = 1 | 0/1 | 0 | 1 | 0 | – | – | | |
| | (SCL00) | Output | PIOR1 = 1 | 0/1 | 0 | 1 | 0 | – | – | | |
| P16 | P16 | Input | – | × | 1 | × | 0 | – | – | √ | √ |
| | | Output | × | 0 | 0 | 0/1 | 0 | (SDA00) = 1 | – | | |
| | | N-ch open drain output | × | 1 | 0 | 0/1 | 0 | | | | |
| | SEG10 | Output | × | 0 | 0 | 0 | 1 | – | – | | |
| | (SI00) | Input | PIOR1 = 1 | × | 1 | × | 0 | – | – | | |
| | (RxD0) | Input | PIOR1 = 1 | × | 1 | × | 0 | – | – | | |
| | (SDA00) | I/O | PIOR1 = 1 | 1 | 0 | 0 | 0 | – | – | | |

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (3/9)**

| Pin Name | Used Function | | PIOR× | POM×× | PM×× | P×× | PFSEG×× (ISCVL3, ISCCAP)Note | Alternate Function Output | | 80-pin | 100-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | SAU Output Function | Other than SAU | | |
| P17 | P17 | Input | – | × | 1 | × | 0 | – | – | √ | √ |
| | | Output | × | 0 | 0 | 0/1 | 0 | (SO00/TxD0) = 1 | – | | |
| | | N-ch open drain output | × | 1 | 0 | 0/1 | 0 | | | | |
| | SEG11 | Output | × | 0 | 0 | 0 | 1 | – | – | | |
| | (SO00) | Output | PIOR1 = 1 | 0/1 | 0 | 1 | 0 | – | – | | |
| | (TxD0) | Output | PIOR1 = 1 | 0/1 | 0 | 1 | 0 | – | – | | |

**Note** ISCVL3 and ISCCAP are registers that correspond to P125, and P126 and P127, respectively.

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (4/9)**

| Pin Name | Used Function | | ADPC | ADM2 | PM×× | P×× | 80-pin | 100-pin |
|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | |
| P20 | P20 | Input | 01H | × | 1 | × | √ | √ |
| | | Output | 01H | × | 0 | 0/1 | | |
| | ANI0 | Analog input | 00H/02H to 06H | 00x0xx0xB<br>10x0xx0xB | 1 | × | | |
| | $AV_{REFP}$ | Reference voltage input | 00H/02H to 06H | 01x0xx0xB | 1 | × | | |
| P21 | P21 | Input | 01H/02H | × | 1 | × | √ | √ |
| | | Output | 01H/02H | × | 0 | 0/1 | | |
| | ANI1 | Analog input | 00H/03H to 06H | xx00xx0xB | 1 | × | | |
| | $AV_{REFM}$ | Reference voltage input | 00H/03H to 06H | xx10xx0xB | 1 | × | | |
| P22 | P22 | Input | 01H to 03H | × | 1 | × | √ | √ |
| | | Output | 01H to 03H | × | 0 | 0/1 | | |
| | ANI2 | Analog input | 00H/04H to 06H | × | 1 | × | | |
| | IVCMP0 | Analog input | 00H/04H to 06H | × | 1 | × | | |
| | IVREF1 | Analog input | 00H/04H to 06H | × | 1 | × | | |
| P23 | P23 | Input | 01H to 04H | × | 1 | × | √ | √ |
| | | Output | 01H to 04H | × | 0 | 0/1 | | |
| | ANI3 | Analog input | 00H/05H/06H | × | 1 | × | | |
| | IVCMP1 | Analog input | 00H/05H/06H | × | 1 | × | | |
| | IVREF0 | Analog input | 00H/05H/06H | × | 1 | × | | |
| P24 | P24 | Input | 01H to 05H | × | 1 | × | × | √ |
| | | Output | 01H to 05H | × | 0 | 0/1 | | |
| | ANI4 | Analog input | 00H/06H | × | 1 | × | | |
| P25 | P25 | Input | 01H to 06H | × | 1 | × | × | √ |
| | | Output | 01H to 06H | × | 0 | 0/1 | | |
| | ANI5 | Analog input | 00H | × | 1 | × | | |

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (5/9)**

| Pin Name | Used Function | | PIOR× | POM×× | PM×× | P×× | PFSEG×× (ISCVL3, ISCCAP)[Note] | Alternate Function Output | | 80-pin | 100-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | SAU Output Function | Other than SAU | | |
| P30 | P30 | Input | – | – | 1 | × | 0 | – | – | √ | √ |
| | | Output | × | – | 0 | 0/1 | 0 | – | (TI07) = 0 | | |
| | SEG24 | Output | × | – | 0 | 0 | 1 | – | – | | |
| | (TI07) | Input | PIOR0 = 1 | – | 1 | × | 0 | – | – | | |
| | (TO07) | Output | PIOR0 = 1 | – | 0 | 0 | 0 | – | – | | |
| P31 | P31 | Input | – | – | 1 | × | 0 | – | – | √ | √ |
| | | Output | × | – | 0 | 0/1 | 0 | – | (TI06) = 0 | | |
| | SEG25 | Output | × | – | 0 | 0 | 1 | – | – | | |
| | (TI06) | Input | PIOR0 = 1 | – | 1 | × | 0 | – | – | | |
| | (TO06) | Output | PIOR0 = 1 | – | 0 | 0 | 0 | – | – | | |
| P32 | P32 | Input | – | – | 1 | × | 0 | – | – | √ | √ |
| | | Output | × | – | 0 | 0/1 | 0 | – | (PCLBUZ1) = 0 | | |
| | SEG26 | Output | × | – | 0 | 0 | 1 | – | – | | |
| | (PCLBUZ1) | Output | PIOR3 = 1 | – | 0 | 0 | 0 | – | – | | |
| P33 | P33 | Input | – | – | 1 | × | 0 | – | – | √ | √ |
| | | Output | × | – | 0 | 0/1 | 0 | – | (PCLBUZ0) = 0 | | |
| | SEG27 | Output | × | – | 0 | 0 | 1 | – | – | | |
| | (PCLBUZ0) | Output | PIOR3 = 1 | – | 0 | 0 | 0 | – | – | | |
| P34 | P34 | Input | – | – | 1 | × | 0 | – | – | × | √ |
| | | Output | – | – | 0 | 0/1 | 0 | – | – | | |
| | SEG28 | Output | – | – | 0 | 0 | 1 | – | – | | |
| P35 | P35 | Input | – | – | 1 | × | 0 | – | – | × | √ |
| | | Output | – | – | 0 | 0/1 | 0 | – | – | | |
| | SEG29 | Output | – | – | 0 | 0 | 1 | – | – | | |
| P36 | P36 | Input | – | – | 1 | × | 0 | – | – | × | √ |
| | | Output | – | – | 0 | 0/1 | 0 | – | – | | |
| | SEG30 | Output | – | – | 0 | 0 | 1 | – | – | | |
| P37 | P37 | Input | – | – | 1 | × | 0 | – | – | × | √ |
| | | Output | – | – | 0 | 0/1 | 0 | – | – | | |
| | SEG31 | Output | – | – | 0 | 0 | 1 | – | – | | |
| P40 | P40 | Input | – | – | 1 | × | – | – | – | √ | √ |
| | | Output | × | – | 0 | 0/1 | – | – | – | | |
| | TOOL0 | I/O | × | – | × | × | – | – | – | | |
| P41 | P41 | Input | – | – | 1 | × | – | – | – | √ | √ |
| | | Output | × | – | 0 | 0/1 | – | – | TO01 = 0<br>PCLBUZ1 = 0 | | |
| | TI01 | Input | PIOR0 = 0 | – | 1 | × | – | – | – | | |
| | TO01 | Output | PIOR0 = 0 | – | 0 | 0 | – | – | PCLBUZ1 = 0 | | |
| | PCLBUZ1 | Output | PIOR3 = 0 | – | 0 | 0 | – | – | TO01 = 0 | | |
| P42 | P42 | Input | – | – | 1 | × | – | – | – | √ | √ |
| | | Output | × | – | 0 | 0/1 | – | – | – | | |
| | INTP7 | Input | PIOR4=0 | – | 1 | × | – | – | – | | |
| P43 | P43 | Input | – | – | 1 | × | – | – | – | √ | √ |
| | | Output | × | – | 0 | 0/1 | – | – | TO00 = 0<br>PCLBUZ0 = 0 | | |
| | TI00 | Input | PIOR0 = 0 | – | 1 | × | – | – | – | | |
| | TO00 | Output | PIOR0 = 0 | – | 0 | 0 | – | – | PCLBUZ0 = 0 | | |
| | PCLBUZ0 | Output | PIOR3 = 0 | – | 0 | 0 | – | – | TO00 = 0 | | |
| P44 | P44 | Input | – | – | 1 | × | – | – | – | √ | √ |
| | | Output | × | – | 0 | 0/1 | – | – | – | | |
| | INTP6 | Input | PIOR4 = 0 | – | 1 | × | – | – | – | | |

**Note** ISCVL3 and ISCCAP are registers that correspond to P125, and P126 and P127, respectively.

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (6/9)**

| Pin Name | Used Function | | PIOR× | POM×× | PM×× | P×× | PFSEG×× (ISCVL3, ISCCAP)[Note] | Alternate Function Output | | 80-pin | 100-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | SAU Output Function | Other than SAU | | |
| P50 | P50 | Input | – | – | 1 | × | 0 | – | – | × | √ |
| | | Output | – | – | 0 | 0/1 | 0 | – | – | | |
| | SEG32 | Output | – | – | 0 | 0 | 1 | – | – | | |
| P51 | P51 | Input | – | – | 1 | × | 0 | – | – | × | √ |
| | | Output | – | – | 0 | 0/1 | 0 | – | – | | |
| | SEG33 | Output | – | – | 0 | 0 | 1 | – | – | | |
| P52 | P52 | Input | – | – | 1 | × | 0 | – | – | × | √ |
| | | Output | – | – | 0 | 0/1 | 0 | – | – | | |
| | SEG34 | Output | – | – | 0 | 0 | 1 | – | – | | |
| P53 | P53 | Input | – | – | 1 | × | 0 | – | – | × | √ |
| | | Output | – | – | 0 | 0/1 | 0 | – | – | | |
| | SEG35 | Output | – | – | 0 | 0 | 1 | – | – | | |
| P54 | P54 | Input | – | – | 1 | × | 0 | – | – | × | √ |
| | | Output | – | – | 0 | 0/1 | 0 | – | – | | |
| | SEG36 | Output | – | – | 0 | 0 | 1 | – | – | | |
| P55 | P55 | Input | – | – | 1 | × | 0 | – | – | × | √ |
| | | Output | – | – | 0 | 0/1 | 0 | – | – | | |
| | SEG37 | Output | – | – | 0 | 0 | 1 | – | – | | |
| P56 | P56 | Input | – | – | 1 | × | 0 | – | – | × | √ |
| | | Output | – | – | 0 | 0/1 | 0 | – | – | | |
| | SEG38 | Output | – | – | 0 | 0 | 1 | – | – | | |
| P57 | P57 | Input | – | – | 1 | × | 0 | – | – | × | √ |
| | | Output | – | – | 0 | 0/1 | 0 | – | – | | |
| | SEG39 | Output | – | – | 0 | 0 | 1 | – | – | | |
| P60 | P60 | Input | – | – | 1 | × | – | – | – | √ | √ |
| | | N-ch open drain output (6 V tolerance) | × | – | 0 | 0/1 | – | – | SCLA0 = 0 (TO00) = 0 | | |
| | SCLA0 | I/O | × | – | 0 | 0 | – | – | (TO00) = 0 | | |
| | (TI00) | Input | PIOR0 = 1 | – | 1 | × | – | – | – | | |
| | (TO00) | Output | PIOR0 = 1 | – | 0 | 0 | – | – | SCLA0 = 0 | | |
| P61 | P61 | Input | – | – | 1 | × | – | – | – | √ | √ |
| | | N-ch open drain output (6 V tolerance) | × | – | 0 | 0/1 | – | – | SDAA0 = 0 (TO01) = 0 | | |
| | SDAA0 | I/O | × | – | 0 | 0 | – | – | (TO01) = 0 | | |
| | (TI01) | Input | PIOR0 = 1 | – | 1 | × | – | – | – | | |
| | (TO01) | Output | PIOR0 = 1 | – | 0 | 0 | – | – | SDAA0 = 0 | | |
| P62 | P62 | Input | – | – | 1 | × | – | – | – | √ | √ |
| | | N-ch open drain output (6 V tolerance) | × | – | 0 | 0/1 | – | – | (TO02) = 0 (RTC1HZ) = 0 | | |
| | (TI02) | Input | PIOR0 = 1 | – | 1 | × | – | – | – | | |
| | (TO02) | Output | PIOR0 = 1 | – | 0 | 0 | – | – | (RTC1HZ) = 0 | | |
| | (RTC1HZ) | Output | PIOR3 = 1 | – | 0 | 0 | – | – | (TO02) = 0 | | |
| P70 | P70 | Input | – | – | 1 | × | 0 | – | – | √ | √ |
| | | Output | – | – | 0 | 0/1 | 0 | – | – | | |
| | SEG16 | Output | – | – | 0 | 0 | 1 | – | – | | |
| | (INTP0) | Input | PIOR4 = 1 | – | 1 | × | 0 | – | – | | |
| P71 | P71 | Input | – | – | 1 | × | 0 | – | – | √ | √ |
| | | Output | – | – | 0 | 0/1 | 0 | – | – | | |
| | SEG17 | Output | – | – | 0 | 0 | 1 | – | – | | |
| | (INTP1) | Input | PIOR4 = 1 | – | 1 | × | 0 | – | – | | |

**Note** ISCVL3 and ISCCAP are registers that correspond to P125, and P126 and P127, respectively.

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (7/9)**

| Pin Name | Used Function | | PIOR× | POM×× | PM×× | P×× | PFSEG×× (ISCVL3, ISCCAP)[Note] | Alternate Function Output | | 80-pin | 100-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | SAU Output Function | Other than SAU | | |
| P72 | P72 | Input | – | – | 1 | × | 0 | – | – | √ | √ |
| | | Output | – | – | 0 | 0/1 | 0 | – | – | | |
| | SEG18 | Output | – | – | 0 | 0 | 1 | – | – | | |
| | (INTP2) | Input | PIOR4 = 1 | – | 1 | × | 0 | – | – | | |
| P73 | P73 | Input | – | – | 1 | × | 0 | – | – | √ | √ |
| | | Output | – | – | 0 | 0/1 | 0 | – | – | | |
| | SEG19 | Output | – | – | 0 | 0 | 1 | – | – | | |
| | (INTP3) | Input | PIOR4 = 1 | – | 1 | × | 0 | – | – | | |
| P74 | P74 | Input | – | – | 1 | × | 0 | – | – | √ | √ |
| | | Output | – | – | 0 | 0/1 | 0 | – | – | | |
| | SEG20 | Output | – | – | 0 | 0 | 1 | – | – | | |
| | (INTP4) | Input | PIOR4 = 1 | – | 1 | × | 0 | – | – | | |
| P75 | P75 | Input | – | – | 1 | × | 0 | – | – | √ | √ |
| | | Output | – | – | 0 | 0/1 | 0 | – | – | | |
| | SEG21 | Output | – | – | 0 | 0 | 1 | – | – | | |
| | (INTP5) | Input | PIOR4 = 1 | – | 1 | × | 0 | – | – | | |
| P76 | P76 | Input | – | – | 1 | × | 0 | – | – | √ | √ |
| | | Output | – | – | 0 | 0/1 | 0 | – | – | | |
| | SEG22 | Output | – | – | 0 | 0 | 1 | – | – | | |
| | (INTP6) | Input | PIOR4 = 1 | – | 1 | × | 0 | – | – | | |
| P77 | P77 | Input | – | – | 1 | × | 0 | – | – | √ | √ |
| | | Output | – | – | 0 | 0/1 | 0 | – | – | | |
| | SEG23 | Output | – | – | 0 | 0 | 1 | – | – | | |
| | (INTP7) | Input | PIOR4 = 1 | – | 1 | × | 0 | – | – | | |
| P80 | P80 | Input | – | × | 1 | × | 0 | – | – | √ | √ |
| | | Output | × | 0 | 0 | 0/1 | 0 | (SCL10) = 1 | – | | |
| | | N-ch open drain output | × | 1 | 0 | 0/1 | 0 | | | | |
| | SEG12 | Output | × | × | 0 | 0 | 1 | – | – | | |
| | (SCL10) | Output | PIOR2 = 1 | 0/1 | 0 | 1 | 0 | – | – | | |
| P81 | P81 | Input | – | × | 1 | × | 0 | – | – | √ | √ |
| | | Output | × | 0 | 0 | 0/1 | 0 | (SDA10) = 1 | – | | |
| | | N-ch open drain output | × | 1 | 0 | 0/1 | 0 | | | | |
| | SEG13 | Output | × | × | 0 | 0 | 1 | – | – | | |
| | (RxD1) | Input | PIOR2 = 1 | × | 1 | × | 0 | – | – | | |
| | (SDA10) | I/O | PIOR2 = 1 | 1 | 0 | 1 | 0 | – | – | | |
| P82 | P82 | Input | – | × | 1 | × | 0 | – | – | √ | √ |
| | | Output | × | 0 | 0 | 0/1 | 0 | (TxD1) = 1 | – | | |
| | | N-ch open drain output | × | 1 | 0 | 0/1 | 0 | | | | |
| | SEG14 | Output | × | × | 0 | 0 | 1 | – | – | | |
| | (TxD1) | Output | PIOR2 = 1 | 0/1 | 0 | 1 | 0 | – | – | | |
| P83 | P83 | Input | – | – | 1 | × | 0 | – | – | √ | √ |
| | | Output | – | – | 0 | 0/1 | 0 | – | – | | |
| | SEG15 | Output | – | – | 0 | 0 | 1 | – | – | | |
| P84 | P84 | Input | – | – | 1 | × | 0 | – | – | × | √ |
| | | Output | – | – | 0 | 0/1 | 0 | – | – | | |
| | SEG40 | Output | – | – | 0 | 0 | 1 | – | – | | |
| P85 | P85 | Input | – | – | 1 | × | 0 | – | – | × | √ |
| | | Output | – | – | 0 | 0/1 | 0 | – | – | | |
| | SEG41 | Output | – | – | 0 | 0 | 1 | – | – | | |

**Note** ISCVL3 and ISCCAP are registers that correspond to P125, and P126 and P127, respectively.

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (8/9)**

| Pin Name | Used Function | | CMC (EXCLK, OSCSEL, EXCLKS, OSCSELS) | P×× | 80-pin | 100-pin |
|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | |
| P121 | P121 | Input | 00xx/10xx/11xx | × | √ | √ |
| | X1 | – | 01xx | – | | |
| P122 | P122 | Input | 00xx/10xx/11xx | × | √ | √ |
| | X2 | – | 01xx | – | | |
| | EXCLK | Input | 11xx | – | | |
| P123 | P123 | Input | xx00/xx10/xx11 | × | √ | √ |
| | XT1 | – | xx01 | – | | |
| P124 | P124 | Input | xx00/xx10/xx11 | × | √ | √ |
| | XT2 | – | xx01 | – | | |
| | EXCLKS | Input | xx11 | – | | |

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (9/9)**

| Pin Name | Used Function | | PIOR× | POM×× | PM×× | P×× | PFSEG×× (ISCVL3, ISCCAP)[Note] | Alternate Function Output | | 80-pin | 100-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | SAU Output Function | Other than SAU | | |
| P125 | P125 | Input | – | – | 1 | × | 1 | – | – | √ | √ |
| | | Output | × | – | 0 | 0/1 | 1 | – | (TO05) = 0 | | |
| | $V_{L3}$ | – | × | – | 1 | × | 0 | – | – | | |
| | INTP1 | Input | PIOR4 = 0 | – | 1 | × | 1 | – | – | | |
| | (TI05) | Input | PIOR0 = 1 | – | 1 | × | 1 | – | – | | |
| | (TO05) | Output | PIOR0 = 1 | – | 0 | 0 | 1 | – | – | | |
| P126 | P126 | Input | – | – | 1 | × | 1 | – | – | √ | √ |
| | | Output | × | – | 0 | 0/1 | 1 | – | (TO04) = 0 | | |
| | CAPL | – | × | – | 1 | × | 0 | – | – | | |
| | (TI04) | Input | PIOR0 = 1 | – | 1 | × | 1 | – | – | | |
| | (TO04) | Output | PIOR0 = 1 | – | 0 | 0 | 1 | – | – | | |
| P127 | P127 | Input | – | – | 1 | × | 1 | – | – | √ | √ |
| | | Output | × | – | 0 | 0/1 | 1 | – | (TO03) = 0 | | |
| | CAPH | – | × | – | 1 | × | 0 | – | – | | |
| | (TI03) | Input | PIOR0 = 1 | – | 1 | × | 1 | – | – | | |
| | (TO03) | Output | PIOR0 = 1 | – | 0 | 0 | 1 | – | – | | |
| P130 | P130 | Output | × | – | 0 | 0/1 | – | – | RTC1HZ = 0 | √ | √ |
| | RTC1HZ | Output | PIOR3 = 0 | – | 0 | 0 | – | – | – | | |
| P137 | P137 | Input | × | – | – | × | – | – | – | √ | √ |
| | INTP0 | Input | PIOR4 = 0 | – | – | × | – | – | – | | |

**Note** ISCVL3 and ISCCAP are registers that correspond to P125, and P126 and P127, respectively.

### 4.5.4 Operation of Ports That Alternately Function as SEGxx Pins

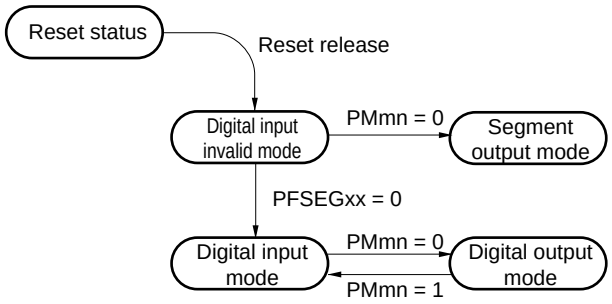
The functions of ports that also serve as segment output pins (SEGxx) can be selected by using the port mode register (PMxx), and LCD port function registers 0 to 5 (PFSEG0 to PFSEG5).

**Table 4-7. Settings of SEGxx/Port Pin Function**

| PFSEGxx Bit of PFSEG0 to PFSEG5 Registers | PMxx Bit of PMxx Register | Pin Function | Initial Status |
|---|---|---|---|
| 1 | 1 | Digital input invalid mode | √ |
| 0 | 0 | Digital output mode | – |
| 0 | 1 | Digital input mode | – |
| 1 | 0 | Segment output mode | – |

The following shows the SEGxx/port pin function status transitions.

**Figure 4-12. SEGxx/Port Pin Function Status Transition Diagram**




**Caution Be sure to set the segment output mode before segment output starts (while SCOC of LCD mode register 1 (LCDM1) is 0).**

### 4.5.5 Operation of Ports That Alternately Function as $V_{L3}$, CAPL, CAPH Pins

The functions of the $V_{L3}$/P125, CAPL/P126, CAPH/P127 pins can be selected by using the LCD input switch control register (ISCLCD), LCD mode register 0 (LCDM0), and port mode register 12 (PM12).

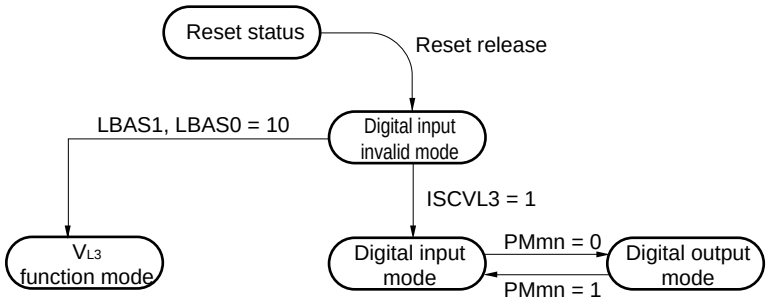
#### (1) $V_{L3}$/P125

**Table 4-8. Settings of $V_{L3}$/P125 Pin Function**

| Bias Setting (LBAS1 and LBAS0 Bits of LCDM0 Register ) | ISCVL3 Bit of ISCLCD Register | PM125 Bit of PM12 Register | Pin Function | Initial Status |
|---|---|---|---|---|
| other than 1/4 bias method (LBAS1, LBAS0 = 00 or 01) | 0 | 1 | Digital input invalid mode | √ |
| | 1 | 0 | Digital output mode | – |
| | 1 | 1 | Digital input mode | – |
| 1/4 bias method (LBAS1, LBAS0 = 10) | 0 | 1 | $V_{L3}$ function mode | – |
| Other than above | | | Setting prohibited | |

The following shows the $V_{L3}$/P125 pin function status transitions.

**Figure 4-13.** $V_{L3}$/P125 Pin Function Status Transition Diagram




**Caution Be sure to set the $V_{L3}$ function mode before segment output starts (while SCOC of LCD mode register 1 (LCDM1) is 0).**

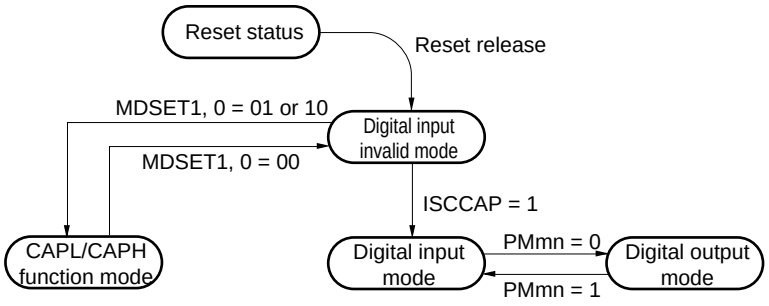
### (2) CAPL/P126, CAPH/P127

**Table 4-9. Settings of CAPL/P126, CAPH/P127 Pins Function**

| LCD Drive Voltage Generator (MDSET1 and MDSET0 Bits of LCDM0 Register ) | ISCCAP Bit of ISCLCD Register | PM126, PM127 Bits of PM12 Register | Pin Function | Initial Status |
|---|---|---|---|---|
| External resistance division (MDSET1, MDSET0 = 00) | 0 | 1 | Digital input invalid mode | √ |
| | 1 | 0 | Digital output mode | – |
| | 1 | 1 | Digital input mode | – |
| Internal voltage boosting or capacitor split (MDSET1, MDSET0 = 01 or 10) | 0 | 1 | CAPL/CAPH function mode | – |
| Other than above | | | Setting prohibited | |

The following shows the CAPL/P126 and CAPH/P127 pins function status transitions.

**Figure 4-14.** CAPL/P126 and CAPH/P127 Pins Function Status Transition Diagram




**Caution Be sure to set the CAPL/CAPH function mode before segment output starts (while SCOC of LCD mode register 1 (LCDM1) is 0).**

## 4.6 Cautions When Using Port Function

### 4.6.1 Cautions on 1-Bit Manipulation Instruction for Port Register n (Pn)

When a 1-bit 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 when switching a port from input mode to output mode.

<Example> When P10 is an output port, P11 to P17 are input ports (all pin statuses are high level), and the port 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 manipulation instruction, the output latch value of port 1 is FFH.

Explanation: The targets of writing to and reading from the Pn register of a port whose PMnm bit is 1 are the output latch and pin status, respectively.

A 1-bit manipulation instruction is executed in the following order in the RL78/I1B.

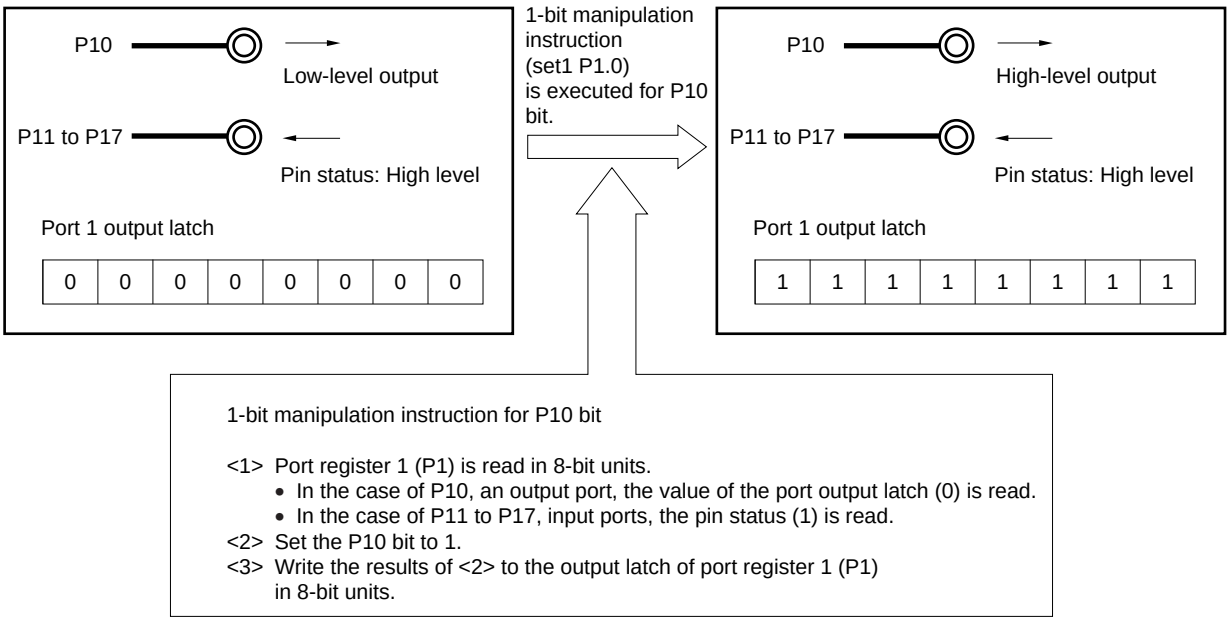
<1> The Pn register is read in 8-bit units.
<2> The targeted one bit is manipulated.
<3> The Pn register is written in 8-bit units.

In step <1>, the output latch value (0) of P10, which is an output port, is read, while the pin statuses of P11 to P17, which are input ports, are read. If the pin statuses 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 4-15. Bit Manipulation Instruction (P10)**



### 4.6.2 Notes on specifying the pin settings

If the output function of an alternate function is assigned to a pin that is also used as an output pin, the output of the unused alternate function must be set to its initial state so as to prevent conflicting outputs. This also applies to the functions assigned by using the peripheral I/O redirection register (PIOR). For details about the alternate output function, see **4.5 Register Settings When Using Alternate Function**.

No specific setting is required for input pins because the output function 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 output, is recommended to lower power consumption.

# CHAPTER 5 CLOCK GENERATOR

## 5.1 Functions of Clock Generator

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

### (1) Main system clock

#### <1> X1 oscillator

This circuit oscillates the X1 oscillator clock ($f_X$ = 1 to 20 MHz) by connecting a resonator to the X1 and X2 pins.
Oscillation can be stopped by executing the STOP instruction or setting the MSTOP bit (bit 7 of the clock operation status control register (CSC)).

#### <2> High-speed on-chip oscillator

The oscillation frequency ($f_{IH}$) can be selected from 24, 12, 6, or 3 MHz (typ.) by using the option byte (000C2H). After reset release, the CPU always starts operating on this high-speed on-chip oscillator clock. Oscillation can be stopped by executing the STOP instruction or setting the HIOSTOP bit (bit 0 of the CSC register).
The frequency specified by using the option byte can be changed by using the high-speed on-chip oscillator frequency select register (HOCODIV). For details about the frequency, see **Figure 5-10 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 option byte and the high-speed on-chip oscillator frequency select register (HOCODIV) are shown below.

| Power Supply Voltage | Oscillation Frequency (MHz) | | | |
|---|---|---|---|---|
| | 3 | 6 | 12 | 24 |
| 2.7 V ≤ $V_{DD}$ ≤ 5.5 V | √ | √ | √ | √ |
| 2.4 V ≤ $V_{DD}$ ≤ 5.5 V | √ | √ | √ | – |
| 1.9 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. The external main system clock input can be disabled by executing the STOP instruction or setting the MSTOP bit.
As the main system clock, a high-speed system clock (X1 clock or external main system clock) or high-speed on-chip oscillator clock can be selected by setting the MCM0 bit (bit 4 of the system clock control register (CKC)).
<R> However, note that the usable frequency range of the main system clock differs depending on the setting of the power supply voltage (VDD). The operating voltage of the flash memory must be set by using the CMODE0 and CMODE1 bits of the option byte (000C2H) (see CHAPTER 32 OPTION BYTE).

**(2) Subsystem clock**

- **XT1 clock oscillator**
  This circuit oscillates the XT1 oscillator clock ($f_{XT}$ = 32.768 kHz) by connecting a 32.768 kHz resonator to the XT1 and XT2 pins. Oscillation can be stopped by setting the XTSTOP bit (bit 6 of the clock operation status control register (CSC)).

An external subsystem clock ($f_{EXS}$ = 32.768 kHz) can also be supplied from the EXCLKS/XT2/P124 pin. An external subsystem clock input can be disabled by the setting of the XTSTOP bit.

**(3) Low-speed on-chip oscillator clock**

This circuit oscillates the low-speed on-chip oscillator clock ($f_{IL}$ = 15 kHz (TYP.)).
The low-speed on-chip oscillator clock cannot be used as the CPU clock.
Only the following peripheral hardware runs on the low-speed on-chip oscillator clock.

- Watchdog timer
- Real-time clock 2 (except high-accuracy 1 Hz output function)
- 12-bit interval timer
- Oscillation stop detection circuit
- LCD controller/driver

This clock operates when either bit 4 (WDTON) of the option byte (000C0H) or bit 4 (WUTMMCK0) of the subsystem clock supply mode control register (OSMC), or both, are set to 1.
However, when WDTON = 1, WUTMMCK0 = 0, and bit 0 (WDSTBYON) of the option byte (000C0H) is 0, oscillation of the low-speed on-chip oscillator stops if the HALT or STOP instruction is executed.

**Cautions 1. The low-speed on-chip oscillator clock ($f_{IL}$) can only be selected as real-time clock 2 count clock when the fixed-cycle interrupt function is used.**
**2. Because the low-speed on-chip oscillator clock must always operate to use the oscillator stop detector, be sure to set bit 4 (WUTMMCK0) of the OSMC register to 1, or bit 4 (WDTON) and bit 0 (WDSTBYON) of the option byte (000C0H) to 1.**

**Remark** $f_X$: X1 clock oscillation frequency
$f_{IH}$: High-speed on-chip oscillator clock frequency
$f_{EX}$: External main system clock frequency
$f_{XT}$: XT1 clock oscillation frequency
$f_{EXS}$: External subsystem clock frequency
$f_{IL}$: Low-speed on-chip oscillator clock frequency

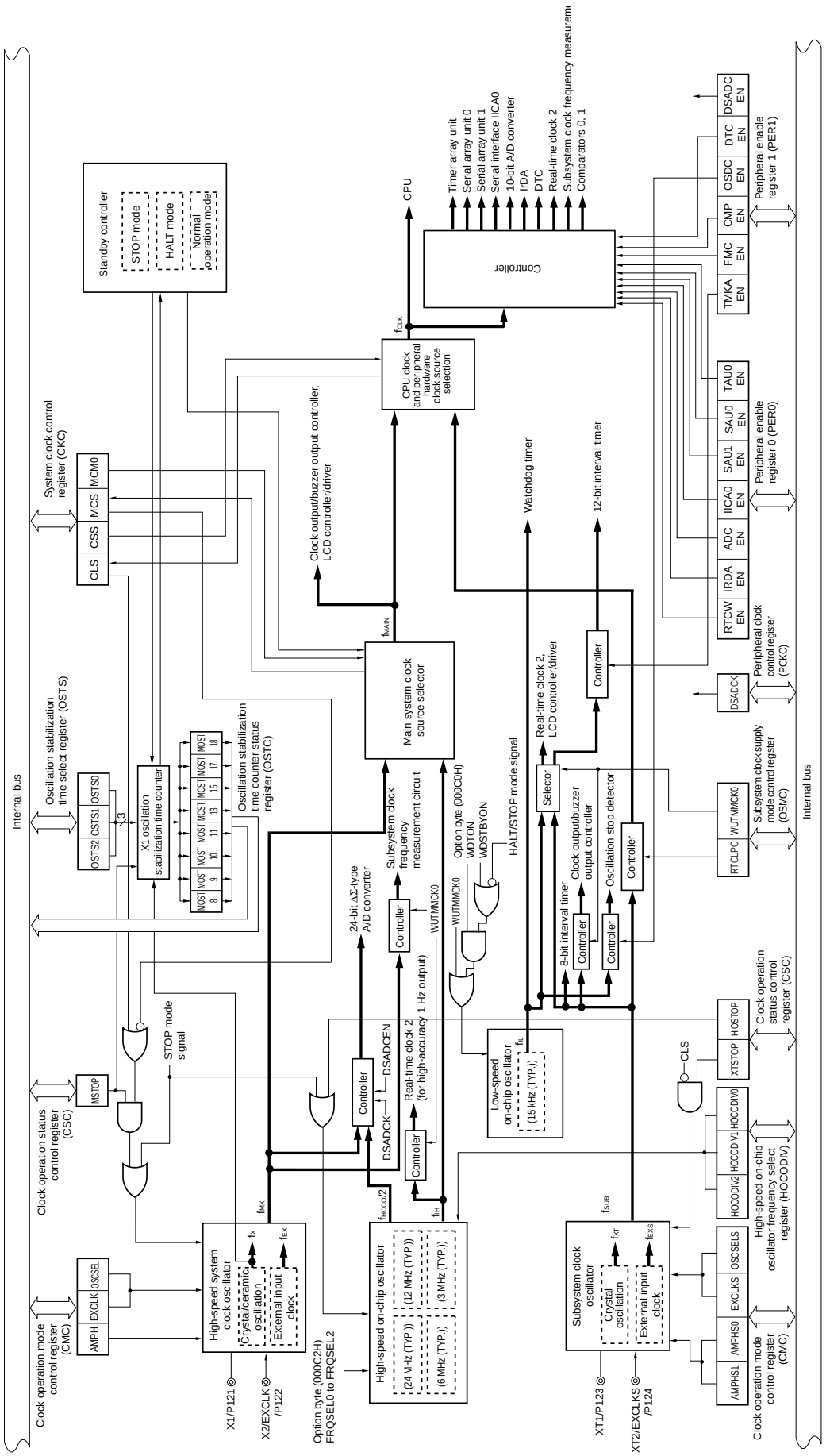
## 5.2 Configuration of Clock Generator

The clock generator includes the following hardware.

**Table 5-1. Configuration of Clock Generator**

| Item | Configuration |
|---|---|
| Control registers | Clock operation mode control register (CMC)<br>System clock control register (CKC)<br>Clock operation status control register (CSC)<br>Oscillation stabilization time counter status register (OSTC)<br>Oscillation stabilization time select register (OSTS)<br>Peripheral enable registers 0 and 1 (PER0, PER1)<br>Subsystem clock supply mode control register (OSMC)<br>High-speed on-chip oscillator frequency select register (HOCODIV)<br>Peripheral clock control register (PCKC) |
| Oscillators | X1 oscillator<br>XT1 oscillator<br>High-speed on-chip oscillator<br>Low-speed on-chip oscillator |

**Figure 5-1. Block Diagram of Clock Generator**




**(Remark** is listed on the next page.)

**Remark** $f_X$: X1 clock oscillation frequency
$f_{HOCO}$: Dedicated clock frequency (24 MHz)
$f_{IH}$: High-speed on-chip oscillator clock frequency (24/12/6/3 MHz)
$f_{EX}$: External main system clock frequency
$f_{MX}$: High-speed system clock frequency
$f_{MAIN}$: Main system clock frequency
$f_{XT}$: XT1 clock oscillation frequency
$f_{EXS}$: External subsystem clock frequency
$f_{SUB}$: Subsystem clock frequency[Note]
$f_{CLK}$: CPU/peripheral hardware clock frequency
$f_{IL}$: Low-speed on-chip oscillator clock frequency

**Note** Selecting $f_{SUB}$ as the output clock of the clock output/buzzer output controller is prohibited when the WUTMMCK0 bit is set to 1.

## 5.3 Registers Controlling 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 and 1 (PER0, PER1)
- Subsystem clock supply mode control register (OSMC)
- High-speed on-chip oscillator frequency select register (HOCODIV)
- Peripheral clock control register (PCKC)

### 5.3.1 Clock operation mode control register (CMC)

This register is used to set the operation mode of the X1/P121, X2/EXCLK/P122, XT1/P123, and XT2/EXCLKS/P124 pins, and to select a gain of the oscillator.

The CMC register can be written only once by an 8-bit memory manipulation instruction after reset release. This register can be read by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Caution The EXCLKS, OSCSELS, AMPHS1, and AMPHS0 bits are reset only by a power-on reset; they retain the previous values when a reset caused by another factor occurs.**

**Figure 5-2. Format of Clock Operation Mode Control Register (CMC)**

Address: FFFA0H After reset: 00H[Note] R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CMC | EXCLK | OSCSEL | EXCLKS[Note] | OSCSELS[Note] | 0 | AMPHS1[Note] | AMPHS0[Note] | AMPH |

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

| EXCLKS | OSCSELS | Subsystem clock pin operation mode | XT1/P123 pin | XT2/EXCLKS/P124 pin |
|---|---|---|---|---|
| 0 | 0 | Input port mode | Input port | |
| 0 | 1 | XT1 oscillation mode | Crystal resonator connection | |
| 1 | 0 | Input port mode | Input port | |
| 1 | 1 | External clock input mode | Input port | External clock input |

| AMPHS1 | AMPHS0 | XT1 oscillator oscillation mode selection |
|---|---|---|
| 0 | 0 | Low power consumption oscillation (default) |
| 0 | 1 | Normal oscillation |
| 1 | 0 | Ultra-low power consumption oscillation |
| 1 | 1 | Setting prohibited |

| AMPH | Control of X1 clock oscillation frequency |
|---|---|
| 0 | 1 MHz ≤ $f_X$ ≤ 10 MHz |
| 1 | 10 MHz < $f_X$ ≤ 20 MHz |

**Note** The EXCLKS, OSCSELS, AMPHS1, and AMPHS0 bits are reset only by a power-on reset; they retain the values when a reset caused by another factor occurs.

**Cautions 1. The CMC register can be written only once after a reset ends, by an 8-bit memory manipulation instruction. When using the CMC register with its initial value (00H), be sure to set the register to 00H after a reset ends in order to prevent malfunction due to a program loop. A malfunction caused by mistakenly writing a value other than 00H is unrecoverable.**

**2. After a reset ends, set up the CMC register before setting the clock operation status control register (CSC) to start X1 or XT1 oscillation .**

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

**4. Specify the settings for the AMPH, AMPHS1, and AMPHS0 bits while $f_{IH}$ is selected as $f_{CLK}$ after a reset ends (before $f_{CLK}$ is switched to $f_{MX}$).**

**5. Count the $f_{XT}$ oscillation stabilization time by using software.**

(The **cautions** continue and **Remark** is given on the next page.)

**Cautions 6. Although the maximum system clock frequency is 24 MHz, the maximum frequency of the X1 oscillator is 20 MHz.**

**7. If a reset other than a power-on reset occurs after the CMC register is written and then the reset ends, be sure to set the CMC register to the value specified before the reset occurred, to prevent a malfunction if a program loop occurs.**

**8. The XT1 oscillator is a circuit with low amplification in order to achieve low-power consumption. Note the following points when designing the circuit.**

- **Pins and circuit boards include parasitic capacitance. Therefore, perform oscillation evaluation using a circuit board to be actually used and confirm that there are no problems.**
- **Before using the ultra-low power consumption oscillation (AMPHS1, AMPHS0 = 1, 0) as the mode of the XT1 oscillator, evaluate the resonators described in 5.7 Resonator and Oscillator Constants.**
- **Make the wiring between the XT1 and XT2 pins and the resonators as short as possible, and minimize the parasitic capacitance and wiring resistance. Note this particularly when the ultra-low power consumption oscillation (AMPHS1, AMPHS0 = 1, 0) is selected.**
- **Configure the circuit of the circuit board, using material with little wiring resistance.**
- **Place a ground pattern that has the same potential as $V_{SS}$ as much as possible near the XT1 oscillator.**
- **Be sure that the signal lines between the XT1 and XT2 pins, and the resonators do not cross with the other signal lines. Do not route the wiring near a signal line through which a high fluctuating current flows.**
- **The impedance between the XT1 and XT2 pins may drop and oscillation may be disturbed due to moisture absorption of the circuit board in a high-humidity environment or dew condensation on the board. When using the circuit board in such an environment, take measures to damp-proof the circuit board, such as by coating.**
- **When coating the circuit board, use material that does not cause capacitance or leakage between the XT1 and XT2 pins.**

**Remark** $f_X$: X1 clock frequency

### 5.3.2 System clock control register (CKC)

This 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.

Reset signal generation sets this register to 00H.

**Figure 5-3. Format of System Clock Control Register (CKC)**

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

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

| CLS | Status of CPU/peripheral hardware clock ($f_{CLK}$) |
|---|---|
| 0 | Main system clock ($f_{MAIN}$) |
| 1 | Subsystem clock ($f_{SUB}$) |

| CSS | Selection of CPU/peripheral hardware clock ($f_{CLK}$) |
|---|---|
| 0 | Main system clock ($f_{MAIN}$) |
| 1[Note 2] | Subsystem clock ($f_{SUB}$) |

| MCS | Status of 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 ($f_{MAIN}$) |

**Notes 1.** Bits 7 and 5 are read-only.

**2.** Changing the value of the MCM0 bit is prohibited while the CSS bit is set to 1.

**Remark** $f_{IH}$: High-speed on-chip oscillator clock frequency
$f_{MX}$: High-speed system clock frequency
$f_{MAIN}$: Main system clock frequency
$f_{SUB}$: Subsystem clock frequency

(**Cautions** are listed on the next page.)

**Cautions 1. Be sure to set bits 3 to 0 to “0”.**

**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 real-time clock 2, subsystem clock frequency measurement circuit, 12-bit interval timer, clock output/buzzer output controller, LCD controller/driver, 8-bit interval timer, oscillation stop detector, and watchdog timer) is also changed at the same time. Consequently, you should stop each peripheral function when changing the CPU/peripheral hardware clock.**

**3. If the subsystem clock is used as the peripheral hardware clock, the operations of the A/D converter and IICA are not guaranteed. For the operating characteristics of the peripheral hardware, refer to the chapters describing the various peripheral hardware as well as CHAPTER 37 ELECTRICAL SPECIFICATIONS.**

### 5.3.3 Clock operation status control register (CSC)

This register is used to control the operations of the high-speed system clock, high-speed on-chip oscillator clock, and subsystem clock (except the low-speed on-chip oscillator clock).

The CSC register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to C0H.

**Caution The XTSTOP bit is reset only by a power-on reset; it retains the value when a reset caused by another factor occurs.**

**Figure 5-4. Format of Clock Operation Status Control Register (CSC)**

Address: FFFA1H After reset: C0H R/W

| Symbol | <7> | <6> | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| CSC | MSTOP | XTSTOP[Note] | 0 | 0 | 0 | 0 | 0 | HIOSTOP |

| MSTOP | High-speed system clock operation control | | |
|---|---|---|---|
| | X1 oscillation mode | External clock input mode | Input port mode |
| 0 | X1 oscillator operating | External clock from EXCLK pin is valid | Input port |
| 1 | X1 oscillator stopped | External clock from EXCLK pin is invalid | |

| XTSTOP | Subsystem clock operation control | | |
|---|---|---|---|
| | XT1 oscillation mode | External clock input mode | Input port mode |
| 0 | XT1 oscillator operating | External clock from EXCLKS pin is valid | Input port |
| 1 | XT1 oscillator stopped | External clock from EXCLKS pin is invalid | |

| HIOSTOP | High-speed on-chip oscillator clock operation control |
|---|---|
| 0 | High-speed on-chip oscillator operating |
| 1 | High-speed on-chip oscillator stopped |

**Note** The XTSTOP bit is reset only by a power-on reset; it retains the value when a reset caused by another factor occurs.

**Cautions 1. After reset release, set the clock operation mode control register (CMC) before setting the CSC register.**

**2. Set up the oscillation stabilization time select register (OSTS) before setting the MSTOP bit to 0 after releasing reset. Note that if the OSTS register is used with its default settings, setting the OSTS register is not required here.**

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

**4. When starting XT1 oscillation by setting the XSTOP bit, wait for oscillation of the subsystem clock to stabilize by setting a wait time using software.**

(The **cautions** continue on the next page.)

**Cautions 5. Do not stop the clock selected for the CPU/peripheral hardware clock ($f_{CLK}$) by using the OSC register.**

<R> **6. The setting of the flags of the register to stop clock oscillation (disabling the external clock input) and the condition before clock oscillation is stopped are shown in Table 5-2. When stopping the clock, confirm the condition before stopping clock.**

**Table 5-2. Stopping the Clock**

| Clock | Condition Before Stopping Clock (Disabling External Clock Input) | Setting of CSC Register Flags |
|---|---|---|
| X1 oscillator clock<br>External main system clock | The CPU/peripheral hardware clock is a clock other than the high-speed system clock.<br>(CLS = 0 and MCS = 0, or CLS = 1) | MSTOP = 1 |
| XT1 oscillator clock<br>External subsystem clock | The CPU/peripheral hardware clock is a clock other than the subsystem clock.<br>(CLS = 0) | XTSTOP = 1 |
| High-speed on-chip oscillator clock | The CPU/peripheral hardware clock is a clock other than the high-speed on-chip oscillator clock.<br>(CLS = 0 and MCS = 1, or CLS = 1) | HIOSTOP = 1 |

### 5.3.4 Oscillation stabilization time counter status register (OSTC)

This is the register that indicates the count status of 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 oscillating while the high-speed on-chip oscillator clock or subsystem clock is used as the CPU clock
- If the STOP mode is entered and then exited while the high-speed on-chip oscillator clock is used 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.

Occurrence of a reset signal, executing the STOP instruction, or setting MSTOP (bit 7 of clock operation status control register (CSC)) to 1 clears the OSTC register to 00H.

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

- When oscillation of the X1 clock starts (EXCLK, OSCSEL = 0, 1 → MSTOP = 0)
- When the STOP mode is exited

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

Address: FFFA2H After reset: 00H R

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| OSTC | MOST 8 | MOST 9 | MOST 10 | MOST 11 | MOST 13 | MOST 15 | MOST 17 | MOST 18 |

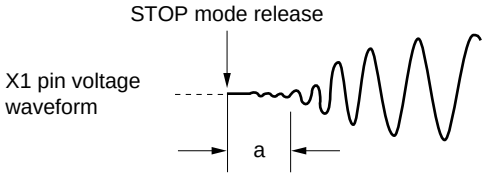
| MOST 8 | MOST 9 | MOST 10 | MOST 11 | MOST 13 | MOST 15 | MOST 17 | MOST 18 | Oscillation stabilization time status | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | $f_X$ = 10 MHz | $f_X$ = 20 MHz |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | $2^8/f_X$ max. | 25.6 μs max. | 12.8 μs max. |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | $2^8/f_X$ min. | 25.6 μs min. | 12.8 μs min. |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | $2^9/f_X$ min. | 51.2 μs min. | 25.6 μs min. |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | $2^{10}/f_X$ min. | 102 μs min. | 51.2 μs min. |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | $2^{11}/f_X$ min. | 204 μs min. | 102 μs min. |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | $2^{13}/f_X$ min. | 819 μs min. | 409 μs min. |
| 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | $2^{15}/f_X$ min. | 3.27 ms min. | 1.63 ms min. |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | $2^{17}/f_X$ min. | 13.1 ms min. | 6.55 ms min. |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | $2^{18}/f_X$ min. | 26.2 ms min. | 13.1 ms min. |

**Cautions 1. After the above time has elapsed, the bits are set to 1 starting from the MOST8 bit, and remain 1.**

**2. The oscillation stabilization time counter counts up to the oscillation stabilization time set by the oscillation stabilization time select register (OSTS).**
**In the following cases, set the oscillation stabilization time of the OSTS register to a value greater than the count value to be monitored by using the OSTC register after the oscillation starts.**

- **To start X1 clock oscillation while the high-speed on-chip oscillator clock or subsystem clock is used as the CPU clock.**
- **To enter and exit the STOP mode while the high-speed on-chip oscillator clock is used as the CPU clock and the X1 clock is oscillating.**
**(Note, therefore, that only the status up to the oscillation stabilization time set by the OSTS register is set to the OSTC register after the STOP mode is exited.)**

**3. The X1 clock oscillation stabilization wait time does not include the time until clock oscillation starts (“a” below).**




**Remark** $f_X$: X1 clock oscillation frequency

### 5.3.5 Oscillation stabilization time select register (OSTS)

This register is used to select the X1 clock oscillation stabilization wait time.

When the X1 clock is made to oscillate by clearing the MSTOP bit to start the X1 oscillation circuit operating, actual operation is automatically delayed for the time set in the OSTS register.

When switching the CPU clock from the high-speed on-chip oscillator clock or the subsystem clock to the X1 clock, and when using the high-speed on-chip oscillator clock for switching the X1 clock from the oscillating state to STOP mode, use the oscillation stabilization time counter status register (OSTC) to confirm that the desired oscillation stabilization time has elapsed after release from the STOP mode. That is, use the OSTC register to check that the oscillation stabilization time corresponding to its setting has been reached.

The OSTS register can be set by an 8-bit memory manipulation instruction.

Reset signal generation sets the OSTS register to 07H.

**Figure 5-6. Format of Oscillation Stabilization Time Select Register (OSTS)**

Address: FFFA3H After reset: 07H R/W

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

| OSTS2 | OSTS1 | OSTS0 | Oscillation stabilization time selection | | |
|---|---|---|---|---|---|
| | | | | $f_X$ = 10 MHz | $f_X$ = 20 MHz |
| 0 | 0 | 0 | $2^8/f_X$ | 25.6 μs | 12.8 μs |
| 0 | 0 | 1 | $2^9/f_X$ | 51.2 μs | 25.6 μs |
| 0 | 1 | 0 | $2^{10}/f_X$ | 102 μs | 51.2 μs |
| 0 | 1 | 1 | $2^{11}/f_X$ | 204 μs | 102 μs |
| 1 | 0 | 0 | $2^{13}/f_X$ | 819 μs | 409 μs |
| 1 | 0 | 1 | $2^{15}/f_X$ | 3.27 ms | 1.63 ms |
| 1 | 1 | 0 | $2^{17}/f_X$ | 13.1 ms | 6.55 ms |
| 1 | 1 | 1 | $2^{18}/f_X$ | 26.2 ms | 13.1 ms |

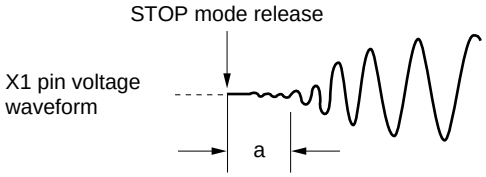
**Cautions 1. Change the setting of the OSTS register before setting the MSTOP bit of the clock operation status control register (CSC) to 0.**

**2. The oscillation stabilization time counter counts up to the oscillation stabilization time set by the OSTS register.**

**In the following cases, set the oscillation stabilization time of the OSTS register to a value greater than the count value to be monitored by using the OSTC register after the oscillation starts.**

- **To start X1 clock oscillation while the high-speed on-chip oscillator clock or subsystem clock is used as the CPU clock.**
- **To enter and exit the STOP mode while the high-speed on-chip oscillator clock is used as the CPU clock and the X1 clock is oscillating.**
  **(Note, therefore, that only the status up to the oscillation stabilization time set by the OSTS register is set to the OSTC register after the STOP mode is exited.)**

**3. The X1 clock oscillation stabilization wait time does not include the time until clock oscillation starts (“a” below).**




**Remark** $f_X$: X1 clock oscillation frequency

### 5.3.6 Peripheral enable registers 0 and 1 (PER0, PER1)

These registers are used to enable or disable supplying the clock to the peripheral hardware. Clock supply to the hardware not used is also stopped so as to reduce the power consumption and noise.

To use the peripheral functions below, which are controlled by these registers, set the bit corresponding to each function to 1 before initial setup of the peripheral functions.

- Real-time clock 2
- IrDA
- 10-bit A/D converter
- Serial interface IICA0
- Serial array unit 1
- Serial array unit 0
- Timer array unit
- 12-bit interval timer
- Subsystem clock frequency measurement circuit
- Comparators 0 and 1
- Oscillation stop detector
- DTC
- 24-bit ΔΣ-type A/D Converter

The PER0 and PER1 registers can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H.

**Figure 5-7. Format of Peripheral Enable Register 0 (PER0) (1/2)**

Address: F00F0H After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PER0 | RTCWEN | IRDAEN | ADCEN | IICA0EN | SAU1EN | SAU0EN | 0 | TAU0EN |

| RTCWEN | Control of real-time clock 2 (RTC2) input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFRs used by real-time clock 2 (RTC2) cannot be written.<br>• Real-time clock 2 (RTC2) is operable. |
| 1 | Enables input clock supply.<br>• SFRs used by real-time clock 2 (RTC2) can be read and written.<br>• Real-time clock 2 (RTC2) is operable. |

| IRDAEN | Control of IrDA input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFRs used by the IrDA cannot be written.<br>• The IrDA is in the reset status. |
| 1 | Enables input clock supply.<br>• SFRs used by the IrDA can be read and written. |

**Caution Be sure to clear bit 1 to "0".**

**Figure 5-7. Format of Peripheral Enable Register 0 (PER0) (2/2)**

Address: F00F0H After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PER0 | RTCWEN | IRDAEN | ADCEN | IICA0EN | SAU1EN | SAU0EN | 0 | TAU0EN |

| ADCEN | Control of A/D converter input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFRs used by the A/D converter cannot be written.<br>• The A/D converter is in the reset status. |
| 1 | Enables input clock supply.<br>• SFRs used by the A/D converter can be read and written. |

| IICA0EN | Control of serial interface IICA0 input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFRs used by serial interface IICA0 cannot be written.<br>• Serial interface IICA0 is in the reset status. |
| 1 | Enables input clock supply.<br>• SFRs used by serial interface IICA0 can be read and written. |

| SAU1EN | Control of serial array unit 1 input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFRs used by serial array unit 1 cannot be written.<br>• Serial array unit 1 is in the reset status. |
| 1 | Enables input clock supply.<br>• SFRs used by serial array unit 1 can be read and written. |

| SAU0EN | Control of serial array unit 0 input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFRs used by serial array unit 0 cannot be written.<br>• Serial array unit 0 is in the reset status. |
| 1 | Enables input clock supply.<br>• SFRs used by serial array unit 0 can be read and written. |

| TAU0EN | Control of timer array unit input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFRs used by timer array unit cannot be written.<br>• Timer array unit is in the reset status. |
| 1 | Enables input clock supply.<br>• SFRs used by timer array unit can be read and written. |

**Caution Be sure to clear bit 1 to “0”.**

**Figure 5-8. Format of Peripheral Enable Register 1 (PER1)**

Address: F007AH After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PER1 | TMKAEN | FMCEN | CMPEN | OSDCEN | DTCEN | 0 | 0 | DSADCEN |

| TMKAEN | Control of 12-bit interval timer input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFRs used by the 12-bit interval timer cannot be written.<br>• The 12-bit interval timer is in the reset status. |
| 1 | Enables input clock supply.<br>• SFRs used by the 12-bit interval timer can be read and written. |

| FMCEN | Subsystem clock frequency measurement circuit input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFRs used by the subsystem clock frequency measurement circuit cannot be written.<br>• SUBCUD register used by real-time clock 2 cannot be written.<br>• The subsystem clock frequency measurement circuit is in the reset status. |
| 1 | Enables input clock supply.<br>• SFRs used by the subsystem clock frequency measurement circuit can be read and written.<br>• SUBCUD register used by real-time clock 2 can be read and written. |

| CMPEN | Control of comparators 0/1 input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFRs used by comparators 0 and 1 cannot be written.<br>• Comparators 0 and 1 are in the reset status. |
| 1 | Enables input clock supply.<br>• SFRs used by comparators 0 and 1 can be read and written. |

| OSDCEN | Control of oscillation stop detector input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFRs used by the oscillation stop detector cannot be written.<br>• The oscillation stop detector is in the reset status. |
| 1 | Enables input clock supply.<br>• SFRs used by the oscillation stop detector can be read and written. |

| DTCEN | Control of DTC input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• The DTC cannot run. |
| 1 | Enables input clock supply.<br>• The DTC can run. |

| DSADCEN | Control of 24-bit ΔΣ-type A/D converter input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFRs used by the 24-bit ΔΣ-type A/D converter cannot be written.<br>• The 24-bit ΔΣ-type A/D converter is in the reset status. |
| 1 | Enables input clock supply.<br>• SFRs used by the 24-bit ΔΣ-type A/D converter be read and written. |

**Caution Be sure to clear bits 2 and 1 to “0”.**

### 5.3.7 Subsystem clock supply mode control register (OSMC)

This register is used to reduce power consumption by stopping unnecessary clock functions.

If the RTCLPC bit is set to 1, power consumption can be reduced, because clock supply to the peripheral functions other than real-time clock 2, 12-bit interval timer, clock output/buzzer output controller, LCD controller/driver, 8-bit interval timer, and oscillation stop detector is stopped in STOP mode or in HALT mode while the subsystem clock is selected as the CPU clock.

In addition, the OSMC register can be used to select the operation clock of real-time clock 2, 12-bit interval timer, clock output/buzzer output controller, LCD controller/driver, 8-bit interval timer, and subsystem clock frequency measurement circuit.

The OSMC register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 5-9. Format of Subsystem Clock Supply Mode Control Register (OSMC)**

Address: F00F3H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| OSMC | RTCLPC | 0 | 0 | WUTMMCK0 | 0 | 0 | 0 | 0 |

| RTCLPC | Setting in STOP mode or in HALT mode while subsystem clock is selected as CPU clock |
|---|---|
| 0 | Enables supplying the subsystem clock to peripheral functions<br>(See **Tables 24-1** and **24-2** for peripheral functions whose operations are enabled.) |
| 1 | Stops supplying the subsystem clock to peripheral functions other than real-time clock 2, 12-bit interval timer, clock output/buzzer output controller, LCD controller/driver, 8-bit interval timer, and oscillation stop detector. |

| WUTMMCK0 | Selection of operation clock for real-time clock 2, 12-bit interval timer, and LCD controller/driver. | Selection of clock output from PCLBUZn pin of clock output/buzzer output controller and selection of operation clock for 8-bit interval timer. | Operation of subsystem clock frequency measurement circuit. |
|---|---|---|---|
| 0 | Subsystem clock ($f_{SUB}$) | Selecting the subsystem clock ($f_{SUB}$) is enabled. | Enable |
| 1 | Low-speed on-chip oscillator clock ($f_{IL}$) | Selecting the subsystem clock ($f_{SUB}$) is disabled. | Disable |

**Cautions 1. Setting the RTCLPC bit to 1 can reduce current consumption in STOP mode and in HALT mode with the CPU operating on the subsystem clock. However, setting the RTCLPC bit to 1 means that there is no clock supply to peripheral circuits other than real-time clock 2, 12-bit interval timer, clock output/buzzer output controller, and LCD controller/driver in HALT mode with the CPU operating on the subsystem clock. Before setting the system to HALT mode with the CPU operating on the subsystem clock, therefore, be sure to set bit 7 (RTCWEN) of peripheral enable register 0 (PER0) and bit 7 (TMKAEN) of peripheral enable register 1 (PER1) to 1, and bits 0, 2, and 3 of PER0, and bit 5 of PER1 to 0.**

**2. If the subsystem clock is oscillating, only the subsystem clock can be selected (WUTMMCK0 = 0).**

**3. When WUTMMCK0 is set to 1, the low-speed on-chip oscillator clock oscillates.**

(The **cautions** and **remark** are given on the next page.)

**Cautions 4. When WUTMMCK0 is set to 1, only the constant-period interrupt function of real-time clock 2 can be used. The year, month, day of the week, day, hour, minute, and second counters and the 1 Hz output function of real-time clock 2 cannot be used.**

**The interval of the constant-period interrupt is calculated by constant period (value selected by using the RTCC0 register) × $f_{SUB}/f_{IL}$.**

**5. The subsystem clock and low-speed on-chip oscillator clock can only be switched by using the WUTMMCK0 bit if real-time clock 2, 12-bit interval timer, and LCD controller/driver are all stopped.**

**These are stopped as follows:**

**Real-time clock 2: Set the RTCE bit to 0.**

**12-bit interval timer: Set the RINTE bit to 0.**

**LCD controller/driver: Set the SCOC and VLCON bits to 0.**

**6. Do not select $f_{SUB}$ as the clock output from the output/buzzer output controller or the operation clock of the 8-bit interval timer when the WUTMMCK0 bit is 1.**

**Remark** RTCE: Bit 7 of real-time clock control register 0 (RTCC0)
RINTE: Bit 15 of the 12-bit interval timer control register (ITMC)
SCOC: Bit 6 of LCD mode register 1 (LCDM1)
VLCON: Bit 5 of LCD mode register 1 (LCDM1)

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

This register is used to change the high-speed on-chip oscillator frequency set by an option byte (000C2H).

The HOCODIV register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to the value set by the FRQSEL2 to FRQSEL0 bits of the option byte (000C2H).

**Figure 5-10. Format of High-speed On-chip Oscillator Frequency Select Register (HOCODIV)**

Address: F00A8H After reset: the value set by FRQSEL2 to FRQSEL0 of the option byte (000C2H) 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 high-speed on-chip oscillator clock frequency |
|---|---|---|---|
| 0 | 0 | 0 | $f_{IH}$ = 24 MHz |
| 0 | 0 | 1 | $f_{IH}$ = 12 MHz |
| 0 | 1 | 0 | $f_{IH}$ = 6 MHz |
| 0 | 1 | 1 | $f_{IH}$ = 3 MHz |
| Other than above | | | Setting prohibited |

**Cautions 1. Set the HOCODIV register within the operable voltage range of the flash operation mode set in the option byte (000C2H) both before and after changing the frequency.**

| Option Byte (000C2H) Value | | Flash Operation Mode | Operating Frequency Range | Operating Voltage Range |
|---|---|---|---|---|
| CMODE1 | CMODE0 | | | |
| 1 | 0 | LS (low-speed main) mode | 6/3 MHz | 1.9 to 5.5 V |
| 1 | 1 | HS (high-speed main) mode | 12/6/3 MHz | 2.4 to 5.5 V |
| | | | 24/12/6/3 MHz | 2.7 to 5.5 V |
| Other than above | | Setting prohibited | | |

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

**3. After the frequency has been changed using the HOCODIV register and the following transition time has been elapsed, the frequency is switched.**

- **Operation for up to three clocks at the pre-change frequency**
- **CPU/peripheral hardware clock wait at the post-change frequency for up to three clocks**

### 5.3.9 Peripheral clock control register (PCKC)

This register is used to select the operation clock of the 24-bit ΔΣ-type A/D converter.

The high-speed system clock ($f_{MX}$), use the12 MHz crystal resonator.

The PCKC register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 5-11. Format of Peripheral Clock Control Register (PCKC)**

Address: F0098H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PCKC | 0 | 0 | 0 | 0 | 0 | 0 | 0 | DSADCK |

| DSADCK | 24-bit ΔΣ-type A/D converter operation clock selection |
|---|---|
| 0 | High-speed on-chip oscillator clock ($f_{IH}$) is supplied. ($f_{MX}$ supply stop.) |
| 1 | High-speed system clock ($f_{MX}$) is supplied.[Note] |

**Note** Only crystal oscillator 12 MHz is allowed as high-speed system clock frequency ($f_{MX}$).

## 5.4 System Clock Oscillator

### 5.4.1 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 bits 7 and 6 (EXCLK, OSCSEL) of the clock operation mode control register (CMC) as follows.

- Crystal or ceramic oscillation: EXCLK, OSCSEL = 0, 1
- External clock input: EXCLK, OSCSEL = 1, 1

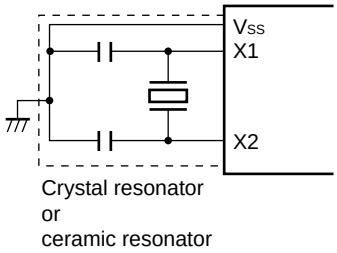
When the X1 oscillator is not used, specify the input port mode (EXCLK, OSCSEL = 0, 0).

When the X1 and X2 pins are not used as input port pins, either, see **Table 2-3 Connection of Unused Pins**.

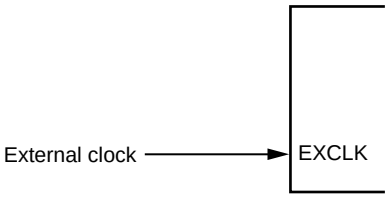
Figure 5-12 shows an example of the external circuit connected to the X1 oscillator.

**Figure 5-12. Example of External Circuit Connected to X1 Oscillator**

**(a) Crystal or ceramic oscillation**




**(b) External clock**




Cautions are listed on the next page.

### 5.4.2 XT1 oscillator

The XT1 oscillator oscillates with a crystal resonator (32.768 kHz typ.) connected to the XT1 and XT2 pins.

To use the XT1 oscillator, set bit 4 (OSCSELS) of the clock operation mode control register (CMC) to 1.

An external clock can also be input. In this case, input the clock signal to the EXCLKS pin.

To use the XT1 oscillator, set bits 5 and 4 (EXCLKS, OSCSELS) of the clock operation mode control register (CMC) as follows.

- Crystal or ceramic oscillation: EXCLKS, OSCSELS = 0, 1
- External clock input: EXCLKS, OSCSELS = 1, 1

When the XT1 oscillator is not used, specify the input port mode (EXCLKS, OSCSELS = 0, 0).

When the XT1 and XT2 pins are not used as input port pins, either, see **Table 2-3 Connection of Unused Pins**.

Figure 5-13 shows an example of the external circuit connected to the XT1 oscillator.

**Figure 5-13. Example of External Circuit Connected to XT1 Oscillator**

**(a) Crystal oscillation** **(b) External clock**




**Caution When using the X1 oscillator and XT1 oscillator, wire as follows in the area enclosed by the broken lines in the Figures 5-12 and 5-13 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 high fluctuating current flows.**
- **Always make the ground point of the oscillator capacitor the same potential as Vss. Do not ground the capacitor to a ground pattern through which a high current flows.**
- **Do not fetch signals from the oscillator.**

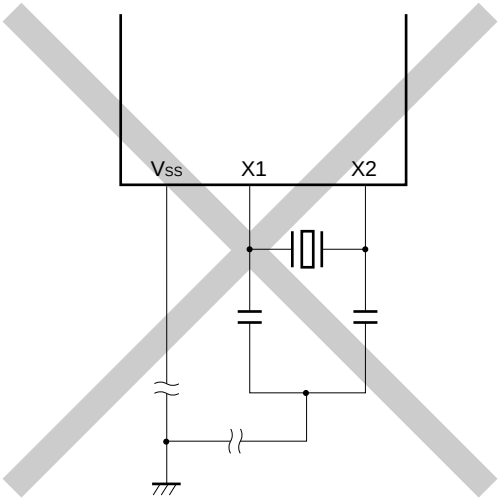
**The XT1 oscillator is a circuit with low amplification in order to achieve low-power consumption. Note the following points when designing the circuit.**

- **Pins and circuit boards include parasitic capacitance. Therefore, perform oscillation evaluation using a circuit board to be actually used and confirm that there are no problems.**
- **Before using the ultra-low power consumption oscillation (AMPHS1, AMPHS0 = 1, 0) as the mode of the XT1 oscillator, evaluate the resonators described in 5.7 Resonator and Oscillator Constants.**
- **Make the wiring between the XT1 and XT2 pins and the resonators as short as possible, and minimize the parasitic capacitance and wiring resistance. Note this particularly when the ultra-low power consumption oscillation (AMPHS1, AMPHS0 = 1, 0) is selected.**
- **Configure the circuit of the circuit board, using material with little wiring resistance.**
- **Place a ground pattern that has the same potential as Vss as much as possible near the XT1 oscillator.**
- **Be sure that the signal lines between the XT1 and XT2 pins, and the resonators do not cross with the other signal lines. Do not route the wiring near a signal line through which a high fluctuating current flows.**
- **The impedance between the XT1 and XT2 pins may drop and oscillation may be disturbed due to moisture absorption of the circuit board in a high-humidity environment or dew condensation on the board. When using the circuit board in such an environment, take measures to damp-proof the circuit board, such as by coating.**
- **When coating the circuit board, use material that does not cause capacitance or leakage between the XT1 and XT2 pins.**

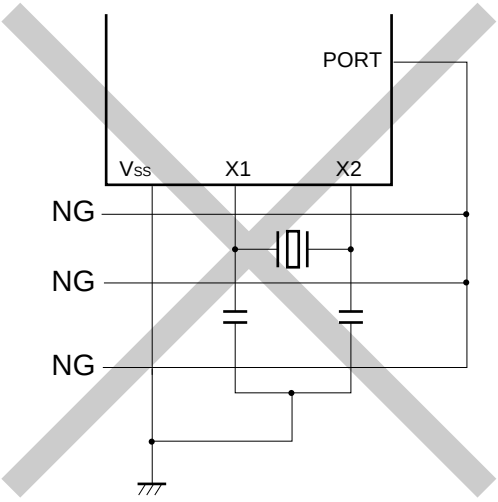
Figure 5-14 shows examples of incorrect resonator connection.

**Figure 5-14. Examples of Incorrect Resonator Connection (1/2)**

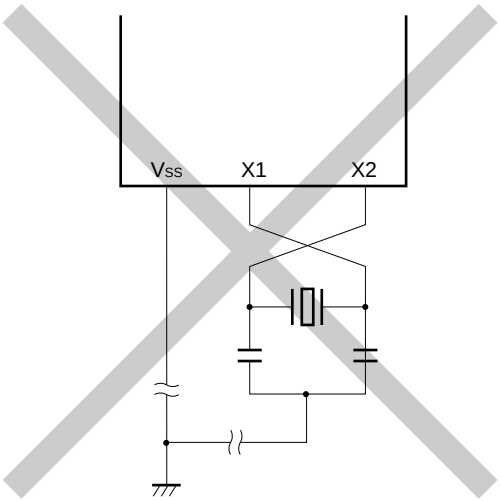
**(a) Too long wiring**



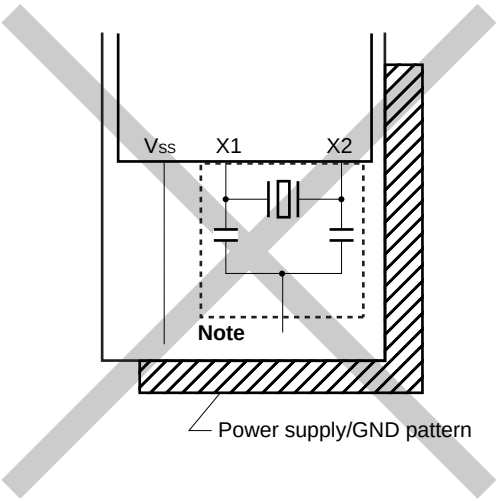

**(b) Crossed signal line**




**(c) The X1 and X2 signal line wires cross.**




**(d) A power supply/GND pattern exists under the X1 and X2 wires.**

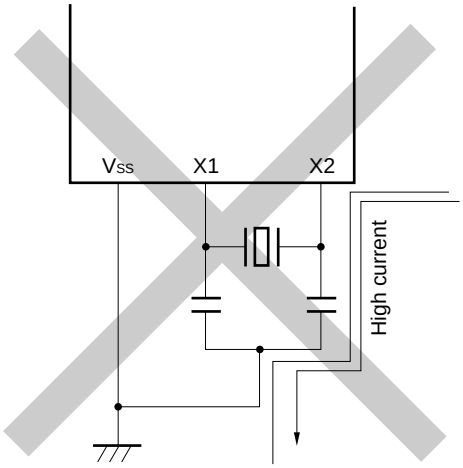



**Note** Do not place a power supply/GND pattern under the wiring section (section indicated by a broken line in the figure) of the X1 and X2 pins and the resonators in a multi-layer board or double-sided board.
Do not configure a layout that will cause capacitance elements and affect the oscillation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

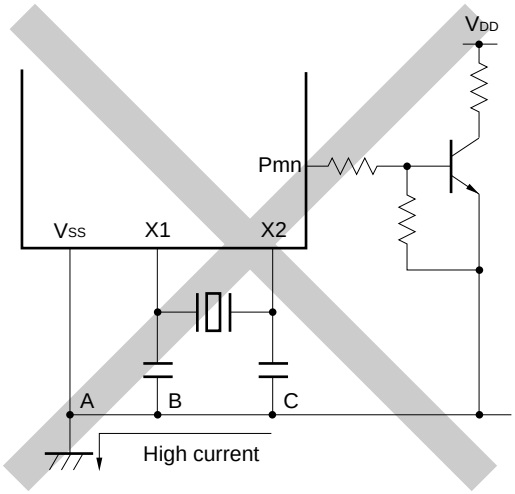
**Remark** When using the subsystem clock, replace X1 and X2 with XT1 and XT2, respectively. Also, insert resistors in series on the XT2 side.

**Figure 5-14. Examples of Incorrect Resonator Connection (2/2)**

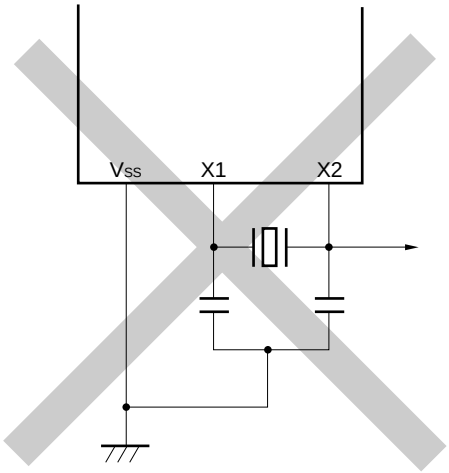
**(e) Wiring near high alternating current**




**(f) Current flowing through ground line of oscillator (potential at points A, B, and C fluctuates)**




**(g) Signals are fetched**




**Caution When X2 and XT1 are wired in parallel, the crosstalk noise of X2 may increase with XT1, resulting in malfunctioning.**

**Remark** When using the subsystem clock, replace X1 and X2 with XT1 and XT2, respectively. Also, insert resistors in series on the XT2 side.

### 5.4.3 High-speed on-chip oscillator

A high-speed on-chip oscillator is incorporated in the RL78/I1B. The frequency can be selected from among 24, 12, 6, or 3 MHz by using the option byte (000C2H). Oscillation can be controlled by using bit 0 (HIOSTOP) of the clock operation status control register (CSC). The high-speed on-chip oscillator automatically starts oscillating after reset release.

### 5.4.4 Low-speed on-chip oscillator

A low-speed on-chip oscillator is incorporated in the RL78/I1B.

The low-speed on-chip oscillator clock is used only as the clock for the watchdog timer, real-time clock 2, 12-bit interval timer, LCD controller/driver, and the oscillation stop detector. The low-speed on-chip oscillator clock cannot be used as the CPU clock.

This clock operates when either bit 4 (WDTON) of the option byte (000C0H) or bit 4 (WUTMMCK0) of the subsystem clock supply mode control register (OSMC), or both, are set to 1.

As long as the watchdog timer is not operating and WUTMMCK0 is not zero, the low-speed on-chip oscillator continues oscillating. Note that only when the watchdog timer is operating and the WUTMMCK0 bit is 0, oscillation of the low-speed on-chip oscillator will stop while the WDSTBYON bit is 0 and operation is in the HALT, STOP, or SNOOZE mode. The low-speed on-chip oscillator clock does not stop even if a program loop that stops the system occurs while the watchdog timer is operating.

**Caution Because the low-speed on-chip oscillator clock must always operate to use the oscillator stop detector, be sure to set bit 4 (WUTMMCK0) of the OSMC register to 1, or bit 4 (WDTON) and bit 0 (WDSTBYON) of the option byte (000C0H) to 1.**

## 5.5 Clock Generator Operation

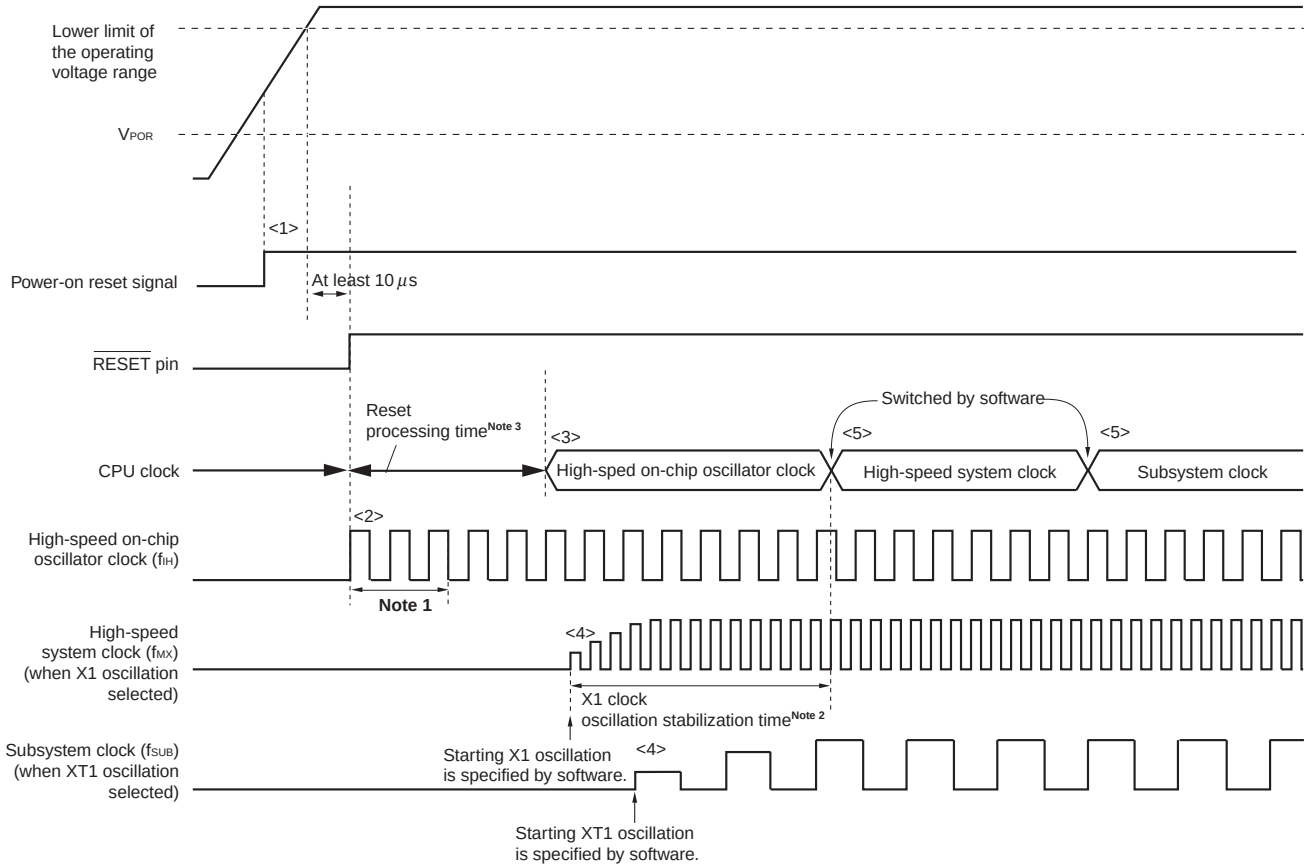
The clock generator generates the following clocks and controls the operation modes of the CPU, such as standby mode (see **Figure 5-1**).

- Main system clock $f_{MAIN}$
  - High-speed system clock $f_{MX}$
    X1 clock $f_X$
    External main system clock $f_{EX}$
  - High-speed on-chip oscillator clock $f_{IH}$
- Subsystem clock $f_{SUB}$
  - XT1 clock $f_{XT}$
  - External subsystem clock $f_{EXS}$
- Low-speed on-chip oscillator clock $f_{IL}$
- CPU/peripheral hardware clock $f_{CLK}$

In the RL78/I1B, the CPU starts operating when the high-speed on-chip oscillator starts generating the clock after reset release.

The clock generator operation after the power supply voltage is turned on is shown in Figure 5-15.

**Figure 5-15. 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 (POR) 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 **37.4 AC Characteristics** (the above figure is an example when the external reset is in use).

<2> When the reset is released, the high-speed on-chip oscillator automatically starts oscillation.

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

<4> Set the start of oscillation of the X1 or XT1 clock via software (see **5.6.2 Example of setting X1 oscillation clock** and **5.6.3 Example of setting XT1 oscillation clock**).

<5> When switching the CPU clock to the X1 or XT1 clock, wait for the clock oscillation to stabilize, and then set switching via software (see **5.6.2 Example of setting X1 oscillation clock** and **5.6.3 Example of setting XT1 oscillation clock**).

**Notes 1.** The reset processing time includes the oscillation accuracy stabilization time of the high-speed on-chip oscillator clock.

**2.** When releasing a reset, confirm the X1 clock oscillation stabilization time by using the oscillation stabilization time counter status register (OSTC).

**3.** For the reset processing time, see **CHAPTER 26 POWER-ON-RESET CIRCUIT.**

**Caution Waiting for the oscillation to stabilize is not necessary when an external clock input from the EXCLK pin is used.**

## 5.6 Controlling the Clock

### 5.6.1 Example of setting high-speed on-chip oscillator

After reset release, the high-speed on-chip oscillator clock is used as the CPU/peripheral hardware clock ($f_{CLK}$). The frequency of the high-speed on-chip oscillator can be selected from 24, 12, 6, or 3 MHz by using FRQSEL0 to FRQSEL2 of the option byte (000C2H). The frequency can also be changed by the high-speed on-chip oscillator frequency select register (HOCODIV).

[Option byte setting]

Address: 000C2H

| Option byte (000C2H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CMODE1<br>0/1 | CMODE0<br>0/1 | 1 | 0 | 0 | FRQSEL2<br>0/1 | FRQSEL1<br>0/1 | FRQSEL0<br>0/1 |

| CMODE1 | CMODE0 | Setting of flash operation mode | |
|---|---|---|---|
| 1 | 0 | LS (low speed main) mode | $V_{DD}$ = 1.9 V to 5.5 V @ 6/3 MHz |
| 1 | 1 | HS (high speed main) mode | $V_{DD}$ = 2.4 V to 5.5 V @ 12/6/3 MHz<br>$V_{DD}$ = 2.7 V to 5.5 V @ 24/12/6/3 MHz |
| Other than above | | Setting prohibited | |

| FRQSEL2 | FRQSEL1 | FRQSEL0 | Frequency of the high-speed on-chip oscillator |
|---|---|---|---|
| | | | $f_{IH}$ |
| 0 | 0 | 0 | 24 MHz |
| 0 | 0 | 1 | 12 MHz |
| 0 | 1 | 0 | 6 MHz |
| 0 | 1 | 1 | 3 MHz |
| Other than above | | | Setting prohibited |

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

Address: F00A8H

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

| HOCODIV2 | HOCODIV1 | HOCODIV0 | Selection of high-speed on-chip oscillator clock frequency |
|---|---|---|---|
| 0 | 0 | 0 | $f_{IH}$ = 24 MHz |
| 0 | 0 | 1 | $f_{IH}$ = 12 MHz |
| 0 | 1 | 0 | $f_{IH}$ = 6 MHz |
| 0 | 1 | 1 | $f_{IH}$ = 3 MHz |
| Other than above | | | Setting prohibited |

### 5.6.2 Example of setting X1 oscillation clock

After reset release, the high-speed on-chip oscillator clock is used as the CPU/peripheral hardware clock ($f_{CLK}$). To change the clock to the X1 oscillation clock, specify the oscillator settings by using the oscillation stabilization time select register (OSTS), clock operation mode control register (CMC), and clock operation status control register (CSC) to start oscillation, and then make sure that oscillation has stabilized by checking the oscillation stabilization time counter status register (OSTC). After the oscillation stabilizes, select the X1 oscillation clock as $f_{CLK}$ by using the system clock control register (CKC).

[Register settings] Set the register according to steps <1> to <5> below.

<1> Set the OSCSEL bit of the CMC register to 1. If $f_X$ is 10 MHz or less, set the AMPH bit to 1 instead, to start the X1 oscillator.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CMC | EXCLK | OSCSEL | EXCLKS | OSCSELS | | AMPHS1 | AMPHS0 | AMPH |
| | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0/1 |

<2> Using the OSTS register, select the oscillation stabilization time of the X1 oscillator after the STOP mode is exited.
Example: Specify as below to wait for oscillation to stabilize for at least 102.4 μs when using a 10 MHz resonator.

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

<3> Clear the MSTOP bit of the CSC register to 0 to start oscillation of the X1 oscillator.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CSC | MSTOP | XTSTOP | | | | | | HIOSTOP |
| | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

<4> Use the OSTC register to wait for oscillation of the X1 oscillator to stabilize.
Example: Wait until the bits are set to the following values to wait for at least 102.4 μs for oscillation to stabilize when using a 10 MHz resonator.

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

<5> Use the MCM0 bit of the CKC register to specify the X1 oscillation clock as the CPU/peripheral hardware clock.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CKC | CLS | CSS | MCS | MCM0 | | | | |
| | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

**Cautions 1 The EXCLKS, OSCSELS, AMPHS1, AMPHS0 and XTSTOP bits are reset only by a power-on reset; they retain the previous values when a reset caused by another factor occurs.**

<R> **Cautions 2. Set the HOCODIV register within the operable voltage range of the flash operation mode set in the option byte (000C2H) both before and after changing the frequency.**

| Option Byte (000C2H) Value | | Flash Operation Mode | Operating Frequency Range | Operating Voltage Range |
|---|---|---|---|---|
| CMODE1 | CMODE0 | | | |
| 1 | 0 | LS (low-speed main) mode | 6/3 MHz | 1.9 to 5.5 V |
| 1 | 1 | HS (high-speed main) mode | 12/6/3 MHz | 2.4 to 5.5 V |
| | | | 24/12/6/3 MHz | 2.7 to 5.5 V |
| Other than above | | Setting prohibited | | |

### 5.6.3 Example of setting XT1 oscillation clock

After reset release, the high-speed on-chip oscillator clock is used as the CPU/peripheral hardware clock ($f_{CLK}$). To change the clock to the XT1 oscillation clock, specify the oscillator settings by using the subsystem clock supply mode control register (OSMC), clock operation mode control register (CMC), and clock operation status control register (CSC) to start oscillation, and then select the XT1 oscillation clock as $f_{CLK}$ by using the system clock control register (CKC).

[Register settings] Set the register according to steps <1> to <5> below.

<1> Set the RTCLPC bit to 1 to run only real-time clock 2, 12-bit interval timer, LCD controller/driver, 8-bit interval timer, and oscillation stop detector on the subsystem clock (for ultra-low current consumption) in the STOP mode or HALT mode during CPU operation on the subsystem clock.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| OSMC | RTCLPC | | | WUTMMCK0 | | | | |
| | 0/1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

<2> Set the OSCSELS bit of the CMC register to 1 to operate the XT1 oscillator.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CMC | EXCLK | OSCSEL | EXCLKS | OSCSELS | | AMPHS1 | AMPHS0 | AMPH |
| | 0 | 0 | 0 | 1 | 0 | 0/1 | 0/1 | 0 |

AMPHS0 and AMPHS1 bits: Use these bits to specify the oscillation mode of the XT1 oscillator.

<3> Clear the XTSTOP bit of the CSC register to 0 to start oscillation of the XT1 oscillator.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CSC | MSTOP | XTSTOP | | | | | | HIOSTOP |
| | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

<4> Use features such as the timer to wait for oscillation of the subsystem clock to stabilize by using software.

<5> Use the CSS bit of the CKC register to specify the XT1 oscillation clock as the CPU/peripheral hardware clock.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CKC | CLS | CSS | MCS | MCM0 | | | | |
| | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

**Caution The EXCLKS, OSCSELS, AMPHS1, AMPHS0 and XTSTOP bits are reset only by a power-on reset; they retain the previous values when a reset caused by another factor occurs.**

### 5.6.4 CPU clock status transition diagram

Figure 5-16 shows the CPU clock status transition diagram of this product.

**Figure 5-16. CPU Clock Status Transition**

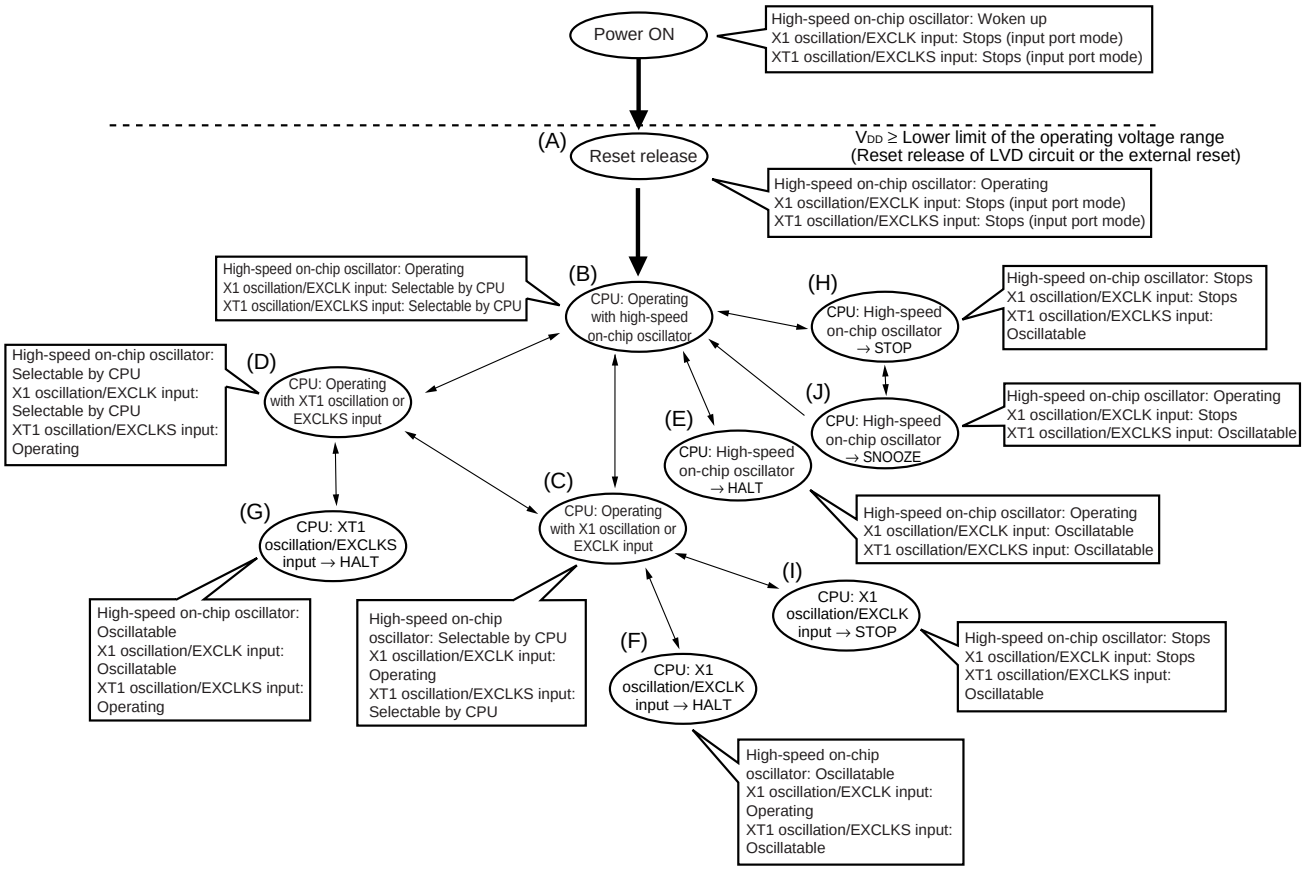


Table 5-3 shows transition of the CPU clock and examples of setting the special function registers (SFRs).

**Table 5-3. CPU Clock Transition and SFR Setting Examples (1/5)**

**(1) CPU operating on high-speed on-chip oscillator clock (B) after reset release (A)**

| Status Transition | SFR Setting |
|---|---|
| (A) → (B) | SFR setting not required (SFRs are in the default status after reset release). |

**(2) CPU operating on high-speed system clock (C) after reset release (A)**

(The CPU operates on the high-speed on-chip oscillator clock immediately after reset release (B).)

(SFR setting sequence) ⟶

| SFR Flag to Set / Status Transition | CMC Register[Note 1] | | | OSTS Register | CSC Register | OSTC Register | CKC Register |
|---|---|---|---|---|---|---|---|
| | EXCLK | OSCSEL | AMPH | | MSTOP | | MCM0 |
| (A) → (B) → (C) (X1 clock: 1 MHz ≤ fx ≤ 10 MHz) | 0 | 1 | 0 | **Note 2** | 0 | Must be checked | 1 |
| (A) → (B) → (C) (X1 clock: 10 MHz < fx ≤ 20 MHz) | 0 | 1 | 1 | **Note 2** | 0 | Must be checked | 1 |
| (A) → (B) → (C) (external main clock) | 1 | 1 | × | **Note 2** | 0 | Not need to be checked | 1 |

**Notes 1.** The clock operation mode control register (CMC) can be written only once by an 8-bit memory manipulation instruction after reset release.

**2.** Set the oscillation stabilization time as follows.

- Desired oscillation stabilization time indicated by the oscillation stabilization time counter status register (OSTC) ≤ Oscillation stabilization time set by the oscillation stabilization time select register (OSTS)

**Caution Specify the clock after the supply voltage has reached the operable voltage of the clock to be specified (see CHAPTER 37 ELECTRICAL SPECIFICATIONS).**

**(3) CPU operating on subsystem clock (D) after reset release (A)**

(The CPU operates on the high-speed on-chip oscillator clock immediately after reset release (B).)

(SFR setting sequence) ⟶

| SFR Flag to Set / Status Transition | CMC Register[Note] | | | | CSC Register | Waiting for Oscillation Stabilization | CKC Register |
|---|---|---|---|---|---|---|---|
| | EXCLKS | OSCSELS | AMPHS1 | AMPHS0 | XTSTOP | | CSS |
| (A) → (B) → (D) (XT1 clock) | 0 | 1 | 0/1 | 0/1 | 0 | Necessary | 1 |
| (A) → (B) → (D) (external subsystem clock) | 1 | 1 | × | × | 0 | Necessary | 1 |

**Note** The clock operation mode control register (CMC) can be written only once by an 8-bit memory manipulation instruction after reset release.

**Remarks 1.** ×: don't care

**2.** (A) to (J) in Table 5-3 correspond to (A) to (J) in Figure 5-16.

**Table 5-3. CPU Clock Transition and SFR Setting Examples (2/5)**

**(4) Changing CPU clock from high-speed on-chip oscillator clock (B) to high-speed system clock (C)**

(SFR setting sequence) →

| SFR Flag to Set / Status Transition | CMC Register[Note 1] | | | OSTS Register | CSC Register | OSTC Register | CKC Register |
|---|---|---|---|---|---|---|---|
| | EXCLK | OSCSEL | AMPH | | MSTOP | | MCM0 |
| (B) → (C) (X1 clock: 1 MHz ≤ $f_X$ ≤ 10 MHz) | 0 | 1 | 0 | **Note 2** | 0 | Must be checked | 1 |
| (B) → (C) (X1 clock: 10 MHz < $f_X$ ≤ 20 MHz) | 0 | 1 | 1 | **Note 2** | 0 | Must be checked | 1 |
| (B) → (C) (external main clock) | 1 | 1 | × | **Note 2** | 0 | Not need to be checked | 1 |

Setting unnecessary if these bits are already set

Setting unnecessary if the CPU is operating on the high-speed system clock

**Notes 1.** The clock operation mode control register (CMC) can be changed only once after reset release. This setting is not necessary if it has already been set.

**2.** Set the oscillation stabilization time as follows.

- Desired oscillation stabilization time indicated by the oscillation stabilization time counter status register (OSTC) ≤ Oscillation stabilization time set by the oscillation stabilization time select register (OSTS)

**Caution Specify the clock after the supply voltage has reached the operable voltage of the clock to be specified (see CHAPTER 37 ELECTRICAL SPECIFICATIONS).**

**(5) Changing CPU clock from high-speed on-chip oscillator clock (B) to subsystem clock (D)**

(Setting sequence of SFR registers) →

| Setting Flag of SFR Register / Status Transition | CMC Register[Note] | | | CSC Register | Waiting for Oscillation Stabilization | CKC Register |
|---|---|---|---|---|---|---|
| | EXCLKS | OSCSELS | AMPHS1,0 | XTSTOP | | CSS |
| (B) → (D) (XT1 clock) | 0 | 1 | 00: Low power consumption oscillation<br>01: Normal oscillation<br>10: Ultra-low power consumption oscillation | 0 | Necessary | 1 |
| (B) → (D) (external sub clock) | 1 | 1 | × | 0 | Necessary | 1 |

Unnecessary if these registers are already set

Unnecessary if the CPU is operating with the subsystem clock

**Note** The clock operation mode control register (CMC) can be written only once by an 8-bit memory manipulation instruction after reset release. This setting is not necessary if it has already been set.

**Remarks 1.** ×: don't care

**2.** (A) to (J) in Table 5-3 correspond to (A) to (J) in Figure 5-16.

**Table 5-3. CPU Clock Transition and SFR Setting Examples (3/5)**

**(6) Changing CPU clock from high-speed system clock (C) to high-speed on-chip oscillator clock (B)**

(SFR setting sequence) →

| SFR Flag to Set / Status Transition | CSC Register | Oscillation Accuracy Stabilization Time | CKC Register |
|---|---|---|---|
| | HIOSTOP | | MCM0 |
| (C) → (B) | 0 | 18 μs to 65 μs | 0 |

Setting unnecessary if the CPU is operating on the high-speed on-chip oscillator clock

**Remark** The oscillation accuracy stabilization time changes according to the temperature conditions and the STOP mode period.

**(7) Changing CPU clock from high-speed system clock (C) to subsystem clock (D)**

(SFR setting sequence) →

| SFR Flag to Set / Status Transition | CSC Register | Waiting for Oscillation Stabilization | CKC Register |
|---|---|---|---|
| | XTSTOP | | CSS |
| (C) → (D) | 0 | Necessary | 1 |

Setting unnecessary if the CPU is operating on the subsystem clock

**(8) Changing CPU clock from subsystem clock (D) to high-speed on-chip oscillator clock (B)**

(Setting sequence of SFR registers) →

| Setting Flag of SFR Register / Status Transition | CSC Register | Oscillation accuracy stabilization time | CKC Register |
|---|---|---|---|
| | HIOSTOP | | CSS |
| (D) → (B) | 0 | 18 μs to 65 μs | 0 |

Unnecessary if the CPU is operating with the high-speed on-chip oscillator clock

**Remarks 1.** (A) to (J) in Table 5-3 correspond to (A) to (J) in Figure 5-16.

**2.** The oscillation accuracy stabilization time changes according to the temperature conditions and the STOP mode period.

**Table 5-3. CPU Clock Transition and SFR Setting Examples (4/5)**

**(9) Changing CPU clock from subsystem clock (D) to high-speed system clock (C)**

(Setting sequence of SFR registers) ▶

| Setting Flag of SFR Register / Status Transition | OSTS Register | CSC Register | OSTC Register | CKC Register |
|---|---|---|---|---|
| | | MSTOP | | CSS |
| (D) → (C)<br>(X1 clock: 1 MHz ≤ $f_X$ ≤ 10 MHz) | **Note** | 0 | Must be checked | 0 |
| (D) → (C)<br>(X1 clock: 10 MHz < $f_X$ ≤ 20 MHz) | **Note** | 0 | Must be checked | 0 |
| (D) → (C) (external main clock) | **Note** | 0 | Must not be checked | 0 |

Unnecessary if the CPU is operating with the high-speed system clock

**Note** Set the oscillation stabilization time as follows.

- Desired oscillation stabilization time indicated by the oscillation stabilization time counter status register (OSTC) ≤ Oscillation stabilization time set by the oscillation stabilization time select register (OSTS)

**Caution Specify the clock after the supply voltage has reached the operable voltage of the clock to be specified (see CHAPTER 37 ELECTRICAL SPECIFICATIONS).**

**(10)**
- **HALT mode (E) entered while CPU is operating on high-speed on-chip oscillator clock (B)**
- **HALT mode (F) entered while CPU is operating on high-speed system clock (C)**
- **HALT mode (G) entered while CPU is operating on subsystem clock (D)**

| Status Transition | Setting |
|---|---|
| (B) → (E)<br>(C) → (F)<br>(D) → (G) | Execute HALT instruction |

**Remark** (A) to (J) in Table 5-3 correspond to (A) to (J) in Figure 5-16.

**Table 5-3. CPU Clock Transition and SFR Setting Examples (5/5)**

**(11)**
- **STOP mode (H) entered while CPU is operating on high-speed on-chip oscillator clock (B)**
- **STOP mode (I) entered while CPU is operating on high-speed system clock (C)**

(Setting sequence) ⟶

| Status Transition | | Setting | | |
|---|---|---|---|---|
| (B) → (H) | | Stopping peripheral functions that are disabled in STOP mode | – | Executing STOP instruction |
| (C) → (I) | In X1 oscillation | | Sets the OSTS register | |
| | External clock | | – | |

**(12) Changing CPU operating mode from STOP mode (H) to SNOOZE mode (J)**

For details about the setting for switching from the STOP mode to the SNOOZE mode, see **14.8 SNOOZE Mode Function**, **18.5.7 SNOOZE mode function**, and **18.6.3 SNOOZE mode function**.

**Remark** (A) to (J) in Table 5-3 correspond to (A) to (J) in Figure 5-16.

### 5.6.5 Conditions before changing the CPU clock and processing after changing CPU clock

The conditions before changing the CPU clock and processing after changing the CPU clock are shown below.

<R>

**Table 5-4. Changing CPU Clock (1/2)**

| CPU Clock | | Conditions Before Change | Processing After Change |
|---|---|---|---|
| Before Change | After Change | | |
| High-speed on-chip oscillator clock | X1 clock | X1 oscillation is stable<br>• OSCSEL = 1, EXCLK = 0, MSTOP = 0<br>• The oscillation stabilization time has elapsed | After confirming that the CPU clock has changed from the high-speed on-chip oscillator clock to the X1 clock, external main system clock, XT1 clock, or external subsystem clock, operating current can be reduced by stopping the high-speed on-chip oscillator (HIOSTOP = 1). |
| | External main system clock | Inputting the external clock from the EXCLK pin is enabled<br>• OSCSEL = 1, EXCLK = 1, MSTOP = 0 | |
| | XT1 clock | XT1 oscillation is stable<br>• OSCSELS = 1, EXCLKS = 0, XTSTOP = 0<br>• The oscillation stabilization time has elapsed | |
| | External subsystem clock | Inputting the external clock from the EXCLKS pin is enabled<br>• OSCSELS = 1, EXCLKS = 1, XTSTOP = 0 | |
| X1 clock | High-speed on-chip oscillator clock | Enabling oscillation of high-speed on-chip oscillator<br>• HIOSTOP = 0<br>• The oscillation accuracy stabilization time has elapsed | After confirming that the CPU clock has changed from the X1 clock to the high-speed on-chip oscillator clock, the X1 oscillation can be stopped (MSTOP = 1). |
| | External main system clock | Transition impossible | – |
| | XT1 clock | XT1 oscillation is stable<br>• OSCSELS = 1, EXCLKS = 0, XTSTOP = 0<br>• The oscillation stabilization time has elapsed | After confirming that the CPU clock has changed from the X1 clock to the XT1 clock, the X1 oscillation can be stopped (MSTOP = 1). |
| | External subsystem clock | Inputting the external clock from the EXCLKS pin is enabled<br>• OSCSELS = 1, EXCLKS = 1, XTSTOP = 0 | After confirming that the CPU clock has changed from the X1 clock to the external subsystem clock, the X1 oscillation can be stopped (MSTOP = 1). |
| External main system clock | High-speed on-chip oscillator clock | Enabling oscillation of high-speed on-chip oscillator<br>• HIOSTOP = 0<br>• The oscillation accuracy stabilization time has elapsed | After confirming that the CPU clock has changed from the external main system clock to the high-speed on-chip oscillator clock, inputting the external main system clock can be disabled (MSTOP = 1). |
| | X1 clock | Transition impossible | – |
| | XT1 clock | XT1 oscillation is stable<br>• OSCSELS = 1, EXCLKS = 0, XTSTOP = 0<br>• The oscillation stabilization time has elapsed | After confirming that the CPU clock has changed from the external main system clock to the XT1 clock, inputting the external main system clock can be disabled (MSTOP = 1). |
| | External subsystem clock | Inputting the external clock from the EXCLKS pin is enabled<br>• OSCSELS = 1, EXCLKS = 1, XTSTOP = 0 | After confirming that the CPU clock has changed from the external main system clock to the external subsystem clock, inputting the external main system clock can be disabled (MSTOP = 1). |

<R>

**Table 5-4. Changing CPU Clock (2/2)**

| CPU Clock | | Condition Before Change | Processing After Change |
|---|---|---|---|
| Before Change | After Change | | |
| XT1 clock | High-speed on-chip oscillator clock | The high-speed on-chip oscillator is oscillating and the high-speed on-chip oscillator clock is selected as the main system clock<br>• HIOSTOP = 0, MCS = 0 | After confirming that the CPU clock has changed from the XT1 clock to the high-speed on-chip oscillator clock, X1 clock, or external main system clock, the XT1 oscillation can be stopped (XTSTOP = 1). |
| | X1 clock | X1 oscillation is stable and the high-speed system clock is selected as the main system clock<br>• OSCSEL = 1, EXCLK = 0, MSTOP = 0<br>• The oscillation stabilization time has elapsed<br>• MCS = 1 | |
| | External main system clock | Inputting the external clock from the EXCLK pin is enabled and the high-speed system clock is selected as the main system clock<br>• OSCSEL = 1, EXCLK = 1, MSTOP = 0<br>• MCS = 1 | |
| | External subsystem clock | Transition impossible | – |
| External subsystem clock | High-speed on-chip oscillator clock | The high-speed on-chip oscillator is oscillating and the high-speed on-chip oscillator clock is selected as the main system clock<br>• HIOSTOP = 0, MCS = 0 | After confirming that the CPU clock has changed from the external subsystem clock to the high-speed on-chip oscillator clock, X1 clock, or external main system clock, inputting the external subsystem clock can be disabled (XTSTOP = 1). |
| | X1 clock | X1 oscillation is stable and the high-speed system clock is selected as the main system clock<br>• OSCSEL = 1, EXCLK = 0, MSTOP = 0<br>• The oscillation stabilization time has elapsed<br>• MCS = 1 | |
| | External main system clock | Inputting the external clock from the EXCLK pin is enabled and the high-speed system clock is selected as the main system clock<br>• OSCSEL = 1, EXCLK = 1, MSTOP = 0<br>• MCS = 1 | |
| | XT1 clock | Transition impossible | – |

### 5.6.6 Time required for switching CPU clock and system clock

By setting bits 4 and 6 (MCM0, CSS) of the system clock control register (CKC), the CPU clock can be switched between the main system clock and the subsystem clock, and main system clock can be switched between the high-speed on-chip oscillator clock and the high-speed system clock.

The clock is not switched immediately after rewriting the CKC register; operation continues on the clock before the change for several clock cycles (see **Table 5-5** to **Table 5-7**).

Whether the CPU is operating on the main system clock or the subsystem clock can be checked by using bit 7 (CLS) of the CKC register. Whether the main system clock is operating on the high-speed system clock or high-speed on-chip oscillator clock can be checked by using bit 5 (MCS) of the CKC register.

When the CPU clock is switched, the peripheral hardware clock is also switched.

**Table 5-5. Maximum Time Required for System Clock Switchover**

| Clock A | Switching Directions | Clock B | Remark |
|---|---|---|---|
| $f_{IH}$ | ◄► | $f_{MX}$ | See **Table 5-6**. |
| $f_{MAIN}$ | ◄► | $f_{SUB}$ | See **Table 5-7**. |

**Table 5-6. Maximum Number of Clock Cycles Required for Switching Between $f_{IH}$ and $f_{MX}$**

| Value Before Switchover | | 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 clock cycles |
| | $f_{MX} < f_{IH}$ | | 2 $f_{IH}/f_{MX}$ clock cycles |
| 1 ($f_{MAIN} = f_{MX}$) | $f_{MX} \geq f_{IH}$ | 2 $f_{MX}/f_{IH}$ clock cycles | |
| | $f_{MX} < f_{IH}$ | 2 clock cycles | |

**Table 5-7. Maximum Number of Clocks Required for Switching Between $f_{MAIN}$ and $f_{SUB}$**

| Value Before Switchover | Value After Switchover | |
|---|---|---|
| CSS | CSS | |
| | 0 ($f_{CLK} = f_{MAIN}$) | 1 ($f_{CLK} = f_{SUB}$) |
| 0 ($f_{CLK} = f_{MAIN}$) | | 1 + 2 $f_{MAIN}/f_{SUB}$ clock cycles |
| 1 ($f_{CLK} = f_{SUB}$) | 3 clock cycles | |

**Remarks 1.** The number of clock cycles in Table 5-6 and Table 5-7 is the number of CPU clock cycles before switchover.

**2.** Calculate the number of clock cycles in Table 5-6 and Table 5-7, rounding out the decimal values.

**Example** When switching the main system clock from the high-speed system clock to the high-speed on-chip oscillator clock (when $f_{IH}$ = 6 MHz, $f_{MX}$ = 10 MHz)

2 $f_{MX}/f_{IH}$ cycles = 2 (10/6) = 3.3 → 4 clock cycles

### 5.6.7 Conditions before stopping clock oscillation

The following lists the register flag settings for stopping the clock oscillation (disabling external clock input) and conditions before the clock oscillation is stopped.

<R> When stopping the clock, confirm the condition before stopping clock.

**Table 5-8. Conditions Before Stopping the Clock Oscillation and Flag Settings**

| Clock | Conditions Before Stopping Clock Oscillation (Disabling External Clock Input) | SFR Flag Settings |
|---|---|---|
| High-speed on-chip oscillator clock | MCS = 1 or CLS = 1<br>(The CPU is operating on a clock other than the high-speed on-chip oscillator clock.) | HIOSTOP = 1 |
| X1 oscillator clock | MCS = 0 or CLS = 1<br>(The CPU is operating on a clock other than the high-speed system clock.) | MSTOP = 1 |
| External main system clock | | |
| XT1 oscillator clock | CLS = 0<br>(The CPU is operating on a clock other than the subsystem clock.) | XTSTOP = 1 |
| External subsystem clock | | |

## 5.7 Resonator and Oscillator Constants

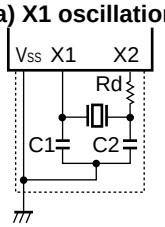
The resonators for which the operation is verified and their oscillator constants are shown below.

**Cautions 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.**

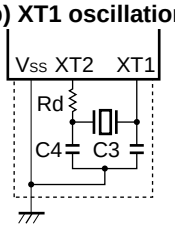
**2. The oscillation voltage and oscillation frequency only indicate the oscillator characteristic. Use the RL78 microcontroller so that the internal operation conditions are within the specifications of the DC and AC characteristics.**

**Figure 5-17. External Oscillation Circuit Example**

**(a) X1 oscillation**




**(b) XT1 oscillation**



**(1) X1 oscillation:**

As of March, 2014 (1/2)

| Manufacturer | Resonator | Part Number | SMD/ Lead | Frequency (MHz) | Flash operation mode[Note 1] | Recommended Circuit Constants[Note 2] (reference) | | | Oscillation Voltage Range (V) | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | C1 (pF) | C2 (pF) | Rd (kΩ) | MIN. | MAX. |
| Murata Manufacturing Co., Ltd.[Note 3] | Ceramic resonator | CSTCC2M00G56-R0 | SMD | 2.0 | LS | (47) | (47) | 0 | 1.9 | 5.5 |
| | | CSTCR4M00G55-R0 | SMD | 4.0 | | (39) | (39) | 0 | | |
| | | CSTLS4M00G53-B0 | Lead | | | (15) | (15) | 0 | | |
| | | CSTCR4M19G55-R0 | SMD | 4.194 | | (39) | (39) | 0 | | |
| | | CSTLS4M19G53-B0 | Lead | | | (15) | (15) | 0 | | |
| | | CSTCR4M91G53-R0 | SMD | 4.915 | | (15) | (15) | 0 | | |
| | | CSTLS4M91G53-B0 | Lead | | | (15) | (15) | 0 | | |
| | | CSTCR5M00G53-R0 | SMD | 5.0 | | (15) | (15) | 0 | | |
| | | CSTLS5M00G53-B0 | Lead | | | (15) | (15) | 0 | | |
| | | CSTCR6M00G53-R0 | SMD | 6.0 | | (15) | (15) | 0 | | |
| | | CSTLS6M00G53-B0 | Lead | | | (15) | (15) | 0 | | |
| | | CSTCE8M00G52-R0 | SMD | 8.0 | | (10) | (10) | 0 | | |
| | | CSTLS8M00G53-B0 | Lead | | | (15) | (15) | 0 | | |
| | | CSTCE8M38G52-R0 | SMD | 8.388 | HS | (10) | (10) | 0 | 2.4 | 5.5 |
| | | CSTLS8M38G53-B0 | Lead | | | (15) | (15) | 0 | | |
| | | CSTCE10M0G52-R0 | SMD | 10.0 | | (10) | (10) | 0 | | |
| | | CSTLS10M0G53-B0 | Lead | | | (15) | (15) | 0 | | |
| | | CSTCE12M0G52-R0 | SMD | 12.0 | | (10) | (10) | 0 | | |
| | | CSTCE16M0V53-R0 | SMD | 16.0 | | (15) | (15) | 0 | | |
| | | CSTLS16M0X51-B0 | Lead | | | (5) | (5) | 0 | | |
| | | CSTCE20M0V51-R0 | SMD | 20.0 | | (5) | (5) | 0 | 2.7 | 5.5 |
| | | CSTLS20M0X51-B0 | Lead | | | (5) | (5) | 0 | | |

**Notes 1.** Set the flash operation mode by using CMODE1 and CMODE0 bits of the option byte (000C2H).

**2.** Values in parentheses in the C1 and C2 columns indicate an internal capacitance.

**3.** When using this resonator, for details about the matching, contact Murata Manufacturing Co., Ltd. (http://www.murata.com).

**Remark** Relationship between operation voltage width, operation frequency of CPU and operation mode is as below.

HS (high-speed main) mode: $2.7\ V \leq V_{DD} \leq 5.5\ V$@1 MHz to 24 MHz

$2.4\ V \leq V_{DD} \leq 5.5\ V$@1 MHz to 16 MHz

LS (low-speed main) mode: $1.9\ V \leq V_{DD} \leq 5.5\ V$@1 MHz to 8 MHz

As of March, 2014 (2/2)

| Manufacturer | Resonator | Part Number[Note 2] | SMD/ Lead | Frequency (MHz) | Flash operation mode[Note 1] | Recommended Circuit Constants (reference) | | | Oscillation Voltage Range (V) | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | C1 (pF) | C2 (pF) | Rd (kΩ) | MIN. | MAX. |
| Nihon Dempa Kogyo Co., Ltd. | Crystal resonator | NX8045GB[Note 3] | SMD | 8.0 | **Note 3** | | | | | |
| | | NX5032GA[Note 3] | SMD | 16.0 | | | | | | |
| | | NX3225HA[Note 3] | SMD | 20.0 | | | | | | |
| Kyocera Crystal Device Co., Ltd. | Crystal resonator | CX8045GB04000D0PPTZ1[Note 4] | SMD | 4.0 | LS | 12 | 12 | 0 | 1.9 | 5.5 |
| | | CX8045GB04915D0PPTZ1[Note 4] | SMD | 4.915 | LS | 12 | 12 | 0 | 1.9 | 5.5 |
| | | CX8045GB08000D0PPTZ1[Note 4] | SMD | 8.0 | | 12 | 12 | 0 | | |
| | | CX8045GB10000D0PPTZ1[Note 4] | SMD | 10.0 | HS | 12 | 12 | 0 | 2.4 | 5.5 |
| | | CX3225GB12000B0PPTZ1[Note 4] | SMD | 12.0 | | 5 | 5 | 0 | | |
| | | CX3225GB16000B0PPTZ1[Note 4] | SMD | 16.0 | | 5 | 5 | 0 | | |
| | | CX3225SB20000B0PPTZ1[Note 4] | SMD | 20.0 | | 5 | 5 | 0 | 2.7 | 5.5 |
| RIVER ELETEC CORPORATION | Crystal resonator | FCX-03-8.000MHZ-J21140[Note 5] | SMD | 8.0 | HS | 3 | 3 | 0 | 2.4 | 5.5 |
| | | FCX-04C-10.000MHZ-J21139[Note 5] | SMD | 10.0 | | 4 | 4 | 0 | | |
| | | FCX-05-12.000MHZ-J21138[Note 5] | SMD | 12.0 | | 6 | 6 | 0 | | |
| | | FCX-06-16.000MHZ-J21137[Note 5] | SMD | 16.0 | | 4 | 4 | 0 | | |

**Notes 1.** Set the flash operation mode by using CMODE1 and CMODE0 bits of the option byte (000C2H).

**2.** This resonator supports operation at up to 85°C.

**3.** When using this resonator, for details about the matching, contact Nihon Dempa Kogyo Co., Ltd (http://www.ndk.com/en).

**4.** When using this resonator, for details about the matching, contact Kyocera Crystal Device Co., Ltd. (http://www.kyocera-crystal.jp/eng/index.html, http://global.kyocera.com).

**5.** When using this resonator, for details about the matching, contact RIVER ELETEC CORPORATION (http://www.river-ele.co.jp/english/index.html).

**Remark** Relationship between operation voltage width, operation frequency of CPU and operation mode is as below.

HS (high-speed main) mode: 2.7 V ≤ $V_{DD}$ ≤ 5.5 V@1 MHz to 24 MHz

2.4 V ≤ $V_{DD}$ ≤ 5.5 V@1 MHz to 16 MHz

LS (low-speed main) mode: 1.9 V ≤ $V_{DD}$ ≤ 5.5 V@1 MHz to 8 MHz

(2) XT1 oscillation: Crystal resonator

As of March, 2014

| Manufacturer | Part Number[Note 2] | SMD/ Lead | Frequency (kHz) | Load Capacitance CL (pF) | XT1 oscillation mode[Note 1] | Recommended Circuit Constants (reference) | | | Oscillation Voltage Range (V) | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | C3 (pF) | C4 (pF) | Rd (kΩ) | MIN. | MAX. |
| Seiko Instruments Inc. | SSP-T7-F[Note 3] | SMD | 32.768 | 7 | Normal oscillation | 11 | 11 | 0 | 1.9 | 5.5 |
| | SSP-T7-FL[Note 3] | | | 6 | | 9 | 9 | 0 | | |
| | | | | 6 | Low power consumption oscillation | 9 | 9 | 0 | | |
| | | | | 4.4 | | 6 | 5 | 0 | | |
| | | | | 4.4 | Ultra-low power consumption oscillation | 6 | 5 | 0 | | |
| | | | | 3.7 | | 4 | 4 | 0 | | |
| | VT-200-FL[Note 3] | Lead | | 6 | Normal oscillation | 9 | 9 | 0 | | |
| | | | | 6 | Low power consumption oscillation | 9 | 9 | 0 | | |
| | | | | 4.4 | | 6 | 5 | 0 | | |
| | | | | 4.4 | Ultra-low power consumption oscillation | 6 | 5 | 0 | | |
| | | | | 3.7 | | 4 | 4 | 0 | | |
| Nihon Dempa Kogyo Co., Ltd. | NX3215SA[Note 4] | SMD | 32.768 | 6 | Normal oscillation | 7 | 7 | 0 | 1.9 | 5.5 |
| | | | | | Low power consumption oscillation | | | | | |
| | | | | | Ultra-low power consumption oscillation | | | | | |
| | NX2012SA[Note 4] | SMD | 32.768 | 6 | Normal oscillation | 7 | 7 | 0 | | |
| | | | | | Low power consumption oscillation | | | | | |
| | | | | | Ultra-low power consumption oscillation | | | | | |
| Kyocera Crystal Device Co., Ltd. | ST3215SB[Note 5] | SMD | 32.768 | 7 | Normal oscillation | 10 | 10 | 0 | 1.9 | 5.5 |
| | | | | | Low power consumption oscillation | | | | | |
| | | | | | Ultra-low power consumption oscillation | | | | | |
| RIVER ELETEC CORPORATION | TFX-02-32.768KHZ-J20986[Note 6] | SMD | 32.768 | 9 | Normal oscillation | 12 | 10 | 0 | 1.9 | 5.5 |
| | | | | | Low power consumption oscillation | | | | | |
| | TFX-03-32.768KHZ-J13375[Note 6] | SMD | 32.768 | 7 | Normal oscillation | 12 | 10 | 0 | | |

**Notes 1.** Set the XT1 oscillation mode by using AMPHS0 and AMPHS1 bits of the clock operation mode control register (CMC).

**2.** This resonator supports operation at up to 85°C.

**3.** This oscillator is a low-power-consumption product. When using it, for details about the matching, contact Seiko Instruments Inc., Ltd (http://www.sii.co.jp/components/quartz/topEN.jsp).

**4.** When using this resonator, for details about the matching, contact Nihon Dempa Kogyo Co., Ltd (http://www.ndk.com/en).

**5.** When using this resonator, for details about the matching, contact Kyocera Crystal Device Co., Ltd. (http://www.kyocera-crystal.jp/eng/index.html, http://global.kyocera.com).

**6.** When using this resonator, for details about the matching, contact RIVER ELETEC CORPORATION (http://www.river-ele.co.jp/english/index.html).

# CHAPTER 6 HIGH-SPEED ON-CHIP OSCILLATOR CLOCK FREQUENCY CORRECTION FUNCTION

## 6.1 High-speed On-chip Oscillator Clock Frequency Correction Function

Using the subsystem clock $f_{SUB}$ (32.768 kHz) as a reference, the frequency of high-speed on-chip oscillator is measured, and the accuracy of the high-speed on-chip oscillator clock ($f_{IH}$) frequency is corrected in real time.

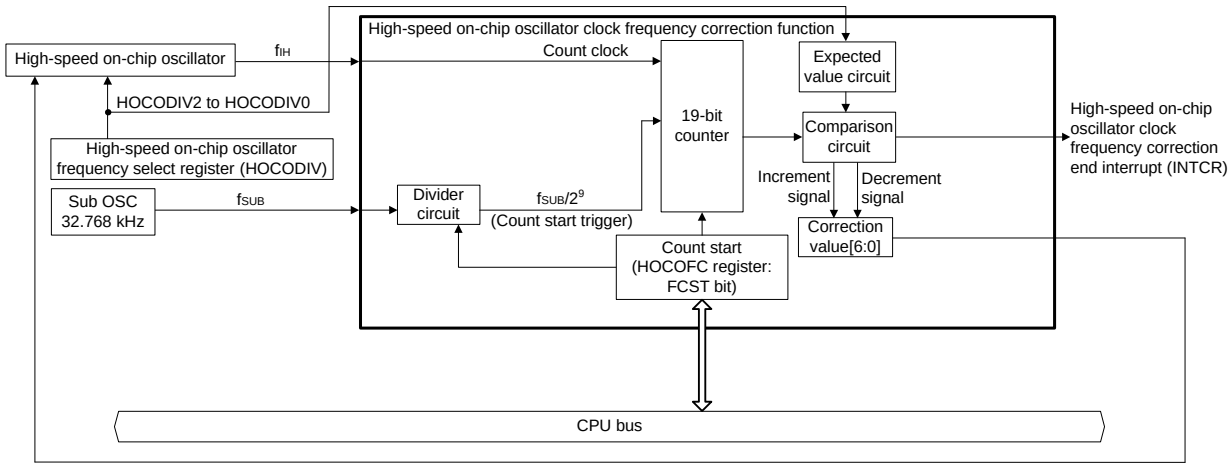
Table 6-1 lists the operation specifications of high-speed on-chip oscillator clock frequency correction function and Figure 6-1 shows the block diagram of high-speed on-chip oscillator clock frequency correction function.

**Table 6-1. Operation Specifications of High-speed On-chip Oscillator Clock Frequency Correction Function**

| Item | Description |
|---|---|
| Reference clock | • $f_{SUB}/2^9$ (subsystem clock: 32.768 kHz) |
| Clock to be corrected | • $f_{IH}$ (high-speed on-chip oscillator clock) |
| Operating modes | • Continuous operating mode<br>The high-speed on-chip oscillator clock frequency is corrected continuously.<br>• Intermittent operating mode<br>The high-speed on-chip oscillator clock frequency is corrected intermittently using a timer interrupt, etc. |
| Clock accuracy correction function | • Correction time: Correction cycle (31.2 ms) × (number of corrections − 0.5)[Note] |
| Interrupt | • Output when high-speed on-chip oscillator clock frequency correction is completed (while interrupt output is enabled). |

**Note** Correction time: Varies depending on the number of corrections.
Correction cycle: Total time of the frequency measurement phase and the frequency correction phase
Number of corrections: The number of repeated correction cycles until the frequency is adjusted to the expected value range.

**Figure 6-1. Block Diagram of High-speed On-chip Oscillator Clock Frequency Correction Function**




**Cautions 1. A subsystem clock is necessary to use the high-speed on-chip oscillator clock frequency correction function. Connect a sub clock oscillator to XT1 and XT2.**

**2. Use this function as necessary to select a high-speed on-chip oscillator as the operating clock when using a 24 bit ΔΣ type A/D converter.**

## 6.2 Register

Table 6-2 lists the register used for the high-speed on-chip oscillator clock frequency correction function.

**Table 6-2. Register for High-speed On-chip Oscillator Clock Frequency Correction Function**

| Item | Configuration |
|---|---|
| Control registers | High-speed on-chip oscillator clock frequency correction control register (HOCOFC) |

### 6.2.1 High-speed on-chip oscillator clock frequency correction control register (HOCOFC)

This register is used to control the high-speed on-chip oscillator clock frequency correction function.

The HOCOFC register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 6-2. Format of High-Speed On-Chip Oscillator Clock Frequency Correction Control Register (HOCOFC)**

Address: F02D8H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| HOCOFC | FCMD | FCIE | 0 | 0 | 0 | 0 | 0 | FCST |

| FCMD[Note 1] | High-speed on-chip oscillator clock frequency correction function operating mode |
|---|---|
| 0 | Continuous operating mode |
| 1 | Intermittent operating mode |

| FCIE | Control of high-speed on-chip oscillator clock frequency correction end interrupt |
|---|---|
| 0 | No interrupt is generated when high-speed on-chip oscillator clock frequency correction is completed |
| 1 | An interrupt is generated when high-speed on-chip oscillator clock frequency correction is completed |

| FCST[Note 2] | High-speed on-chip oscillator clock frequency correction circuit operation control/status |
|---|---|
| 0 | High-speed on-chip oscillator clock frequency correction circuit stops operating/frequency correction is completed |
| 1 | High-speed on-chip oscillator clock frequency correction circuit starts operating/frequency correction is operating |
| In continuous operating mode, operation is stopped by writing 0 to this bit by software.<br>In intermittent operating mode, the FCST bit is cleared by hardware after correction is completed. | |

**Notes 1.** Do not rewrite the FCMD bit when the FCST bit is 1.

**2.** When writing 1 to the FCST bit, confirm that the FCST bit is 0 before writing 1 to FCST. However, when writing 1 to the FCST bit immediately after intermittent operation is completed (an interrupt is generated when high-speed on-chip oscillator clock frequency correction is completed), wait for at least one $f_{IH}$ cycle to elapse after the high-speed on-chip oscillator clock frequency correction end interrupt is generated because clearing by hardware has priority.

After writing 0 (high-speed on-chip oscillator clock frequency correction circuit stops operating) to the FCST bit, do not write 1 (high-speed on-chip oscillator clock frequency correction circuit starts operating) to the FCST bit for two $f_{IH}$ cycles.

**Caution Be sure to clear bits 5 to 1 to "0".**

## 6.3 Operation

### 6.3.1 Operation overview

In high-speed on-chip oscillator clock frequency correction, a correction cycle is generated using the subsystem clock ($f_{SUB}$) as a reference, the frequency of the high-speed on-chip oscillator is measured, and the accuracy of the high-speed on-chip oscillator clock frequency is corrected in real time. In clock correction, operations of the frequency measurement and frequency correction phases are repeated. In the frequency measurement phase, correction is calculated. In the frequency correction phase, the output of the correction value that reflects the correction calculation result is retained.

Table 6-3 lists the high-speed on-chip oscillator input frequency and correction cycle and Figure 6-3 shows the operation timing (details) of high-speed on-chip oscillator clock frequency correction.

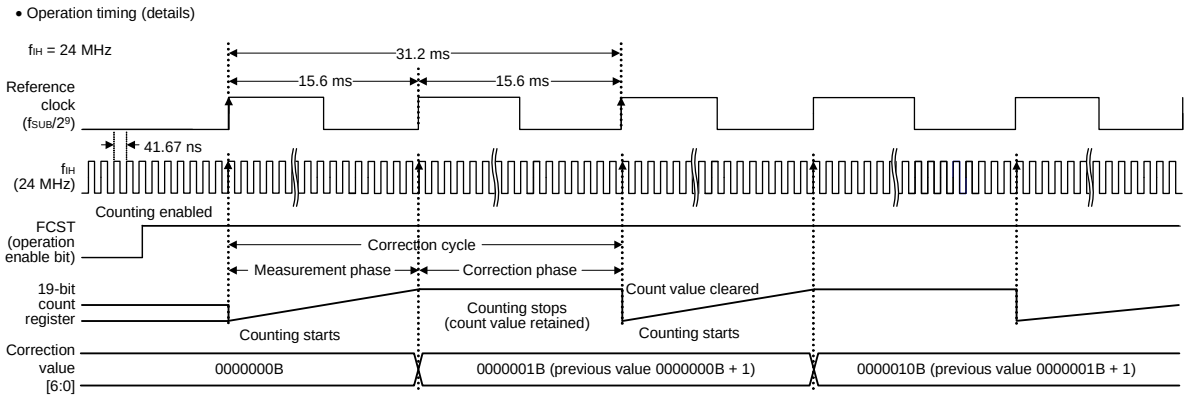
**Table 6-3. High-Speed On-Chip Oscillator Input Frequency and Correction Cycle**

| $f_{IH}$ (MHz) | HOCODIV2 to HOCODIV0[Note] (HOCODIV Register) | Correction Cycle (ms) |
|---|---|---|
| 24 | 000 | 31.2 (frequency measurement phase + frequency correction phase) |
| 12 | 001 | |
| 6 | 010 | |
| 3 | 011 | |
| Other than above | Setting prohibited | |

**Note** Be sure to change the high-speed on-chip oscillator frequency select register (HOCODIV) only when the high-speed on-chip oscillator clock frequency correction function is not used.

The frequency measurement phase period for the correction cycle is counted using the high-speed on-chip oscillator clock, and the high-speed on-chip oscillator frequency is corrected depending on the count result and whether it is greater or smaller than the expected value.

**Figure 6-3. Operation Timing (Details) of High-speed On-chip Oscillator Clock Frequency Correction**




**Remark** Basic operation is the same in both continuous and intermittent operating modes. Only the difference between these modes is clearing the FCST bit is controlled by either software or hardware. In addition, the correction value is not cleared until a reset is applied to the system.

**(1) Continuous operating mode**

In continuous operating mode, the high-speed on-chip oscillator clock frequency is corrected continuously. This mode is selected by setting the FCMD bit in the HOCOFC register to 0.

Operation of high-speed on-chip oscillator clock frequency correction is started by setting the FCST bit in the HOCOFC register to 1. Similarly, operation of high-speed on-chip oscillator clock frequency correction is stopped by setting the FCST bit in the HOCOFC register to 0.

When operation of high-speed on-chip oscillator clock frequency correction is started, frequency counting starts at the rising edge of the reference clock ($f_{SUB}/2^9$) and stops at the next rising edge of the reference clock ($f_{SUB}/2^9$) in the frequency measurement phase.

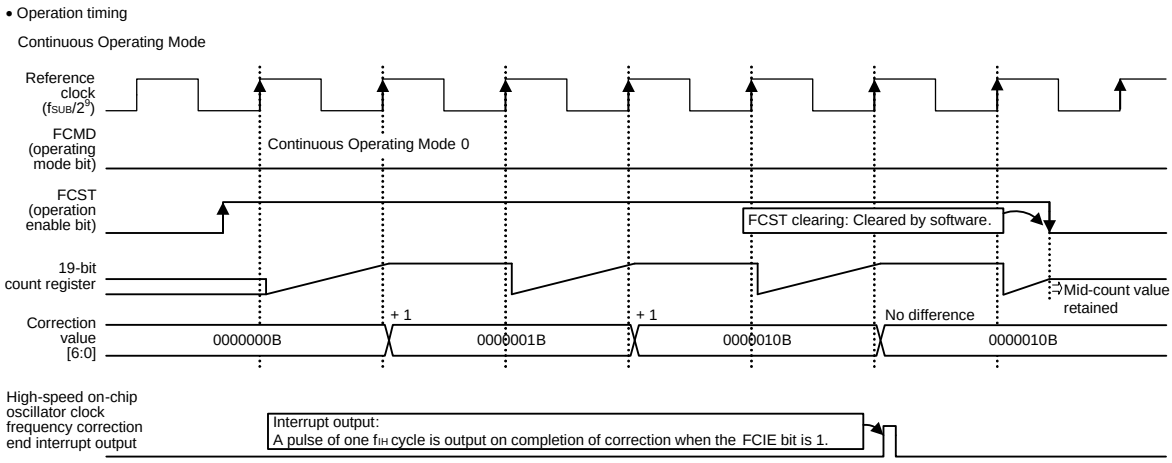
Next, the count value and the expected value are compared, and the correction value is adjusted as follows in the frequency correction phase:

- When the count value is greater than the expected value: Correction value − 1
- When the count value is smaller than the expected value: Correction value + 1
- When the count value is in the range of the expected value: The correction value is retained (high-speed on-chip oscillator clock frequency correction is completed)

When the FCIE bit in the HOCOFC register is set to 1, a high-speed on-chip oscillator clock frequency correction end interrupt is output every time high-speed on-chip oscillator clock frequency correction is completed. In continuous operating mode, the frequency measurement phase and the frequency correction phase are repeated until the high-speed on-chip oscillator clock frequency correction function is stopped.

Figure 6-4 shows the continuous operating mode timing.

**Figure 6-4. Continuous Operating Mode Timing**



**(2) Intermittent operating mode**

In intermittent operating mode, the high-speed on-chip oscillator clock frequency is corrected intermittently using a timer interrupt, etc. This mode is selected by setting the FCMD bit in the HOCOFC register to 1.

Operation of high-speed on-chip oscillator clock frequency correction is started by setting the FCST bit in the HOCOFC register to 1.

When operation of high-speed on-chip oscillator clock frequency correction is started, frequency counting starts at the rising edge of the reference clock ($f_{SUB}/2^9$) and stops at the next rising edge of the reference clock ($f_{SUB}/2^9$) in the frequency measurement phase.

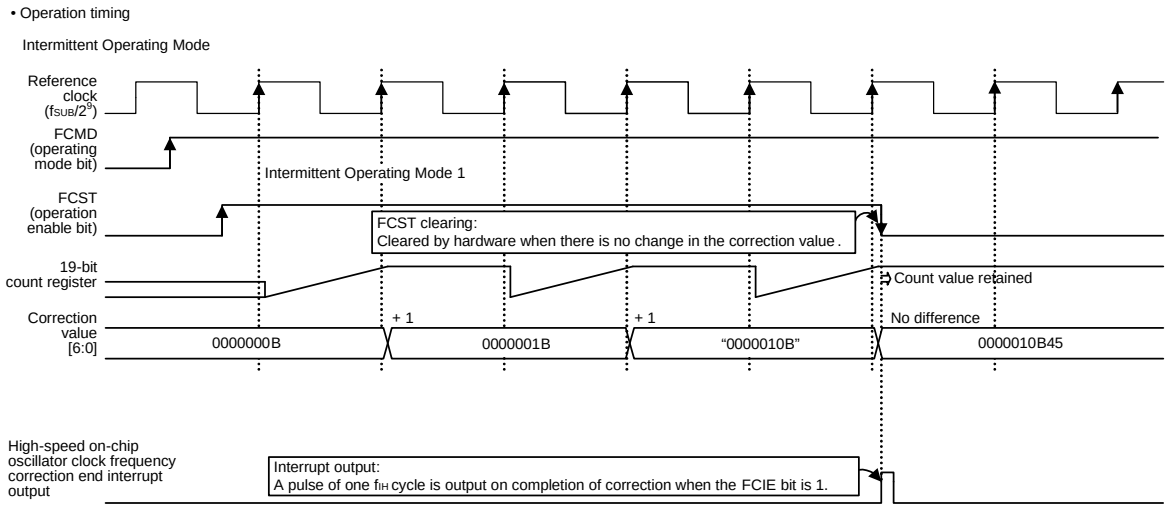
Next, the count value and the expected value are compared, and the correction value is adjusted as follows in the frequency correction phase:

- When the count value is greater than the expected value: Correction value − 1
- When the count value is smaller than the expected value: Correction value + 1
- When the count value is in the range of the expected value: The correction value is retained and the FCST bit is cleared (high-speed on-chip oscillator clock frequency correction is completed)

While the FCIE bit in the HOCOFC register is set to 1, a high-speed on-chip oscillator clock frequency correction end interrupt is output when high-speed on-chip oscillator clock frequency correction is completed. In intermittent operating mode, the frequency measurement phase and the frequency correction phase are repeated, and high-speed on-chip oscillator clock frequency correction operation is stopped after high-speed on-chip oscillator clock frequency correction is completed.

Figure 6-5 shows the intermittent operating mode timing.

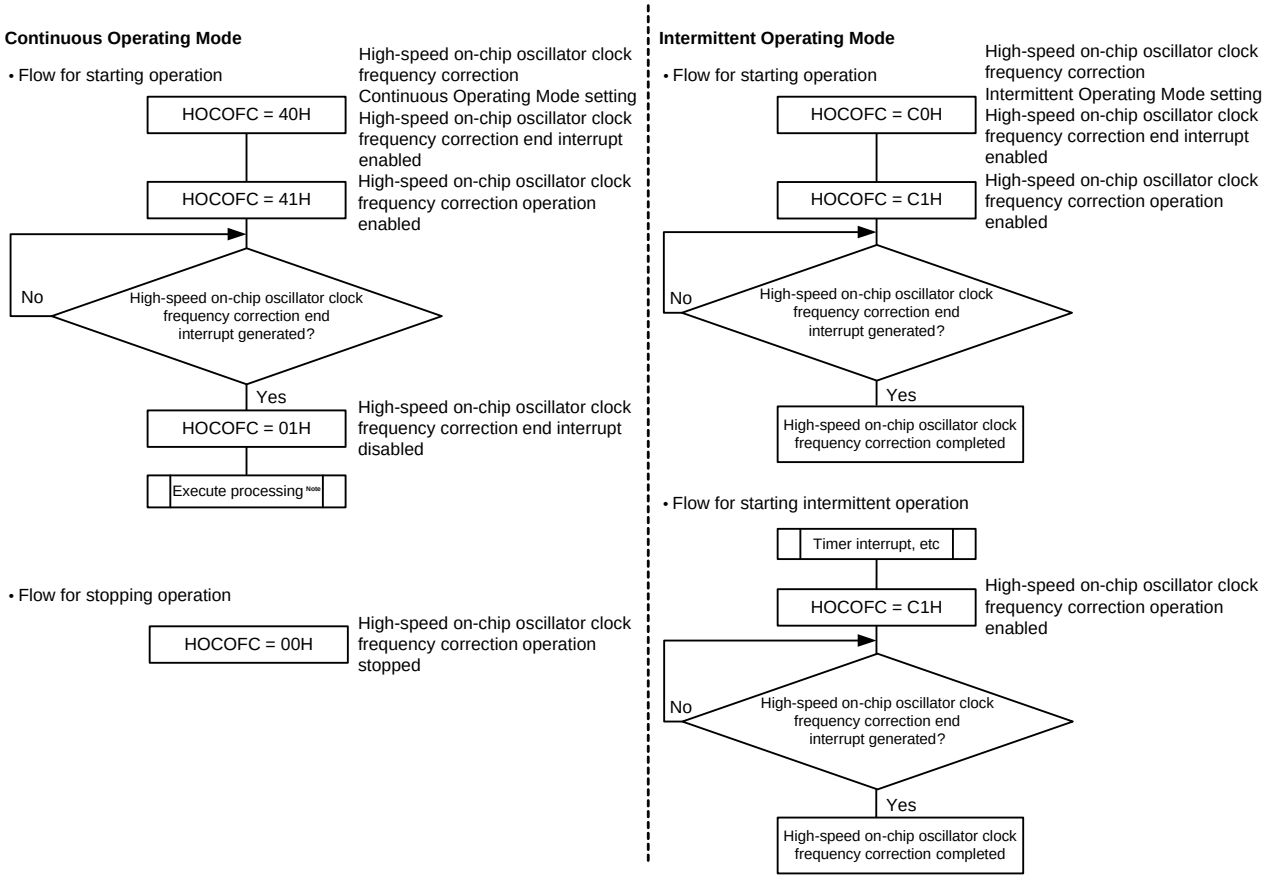
**Figure 6-5. Intermittent Operating Mode Timing**



### 6.3.2 Operation procedure

The following shows the flow for starting and stopping operation when the high-speed on-chip oscillator clock frequency correction function is used.

**Figure 6-6. Example of Procedure for Setting Operating Mode**




**Note** The high speed on-chip oscillator clock frequency correction is repeated until the high speed on-chip oscillator clock frequency correction function is stopped.

## 6.4 Usage Notes

### 6.4.1 SFR access

When writing 1 to FCST to control the FCST bit in intermittent operating mode, confirm that the FCST bit is 0 before writing 1 to the FCST bit. However, when writing 1 to the FCST bit immediately after intermittent operation is completed, wait for at least one $f_{IH}$ cycle after a high-speed on-chip oscillator clock frequency correction end interrupt is generated because clearing by hardware has priority.

### 6.4.2 Operation during standby state

Be sure to stop operation of high-speed on-chip oscillator clock frequency correction before executing the STOP instruction.

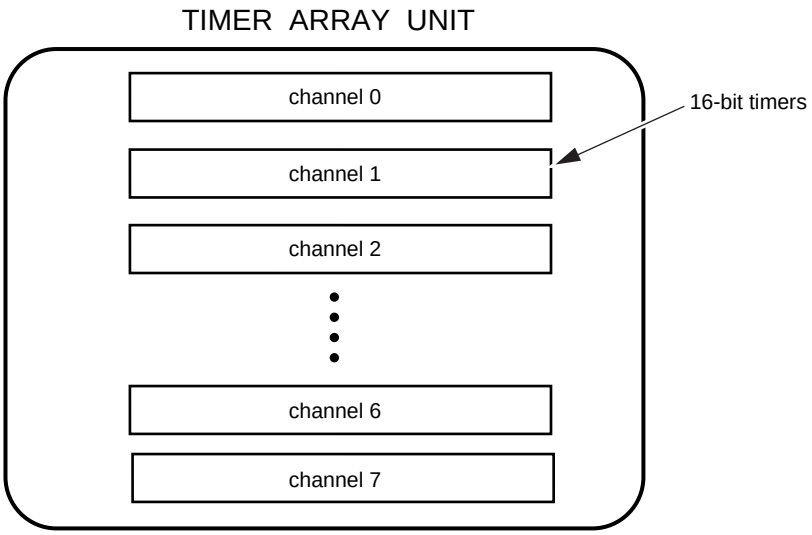
### 6.4.3 Changing high-speed on-chip oscillator frequency select register (HOCODIV)

Be sure to change the high-speed on-chip oscillator frequency select register (HOCODIV) only when the high-speed on-chip oscillator clock frequency correction function is not used.

# CHAPTER 7 TIMER ARRAY UNIT

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**)<br>• Square wave output (→ see **7.8.1**)<br>• External event counter (→ see **7.8.2**)<br>• Input pulse interval measurement (→ see **7.8.3**)<br>• Measurement of high-/low-level width of input signal (→ see **7.8.4**)<br>• Delay counter (→ see **7.8.5**) | • One-shot pulse output(→ see **7.9.1**)<br>• PWM output(→ see **7.9.2**)<br>• Multiple PWM output(→ see **7.9.3**) |

It is possible to use the 16-bit timer of channels 1 and 3 as two 8-bit timers (higher and lower). The functions that can use channels 1 and 3 as 8-bit timers are as follows:

- Interval timer (upper or lower 8-bit timer)/square wave output (lower 8-bit timer only)
- External event counter (lower 8-bit timer only)
- Delay counter (lower 8-bit timer only)

Channel 7 can be used to realize LIN-bus communication operating in combination with UART0 of the serial array unit.

## 7.1 Functions of Timer Array Unit

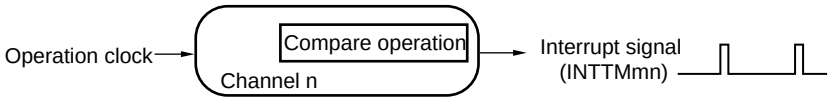
Timer array unit has the following functions.

### 7.1.1 Independent channel operation function

By operating a channel independently, it can be used for the following purposes without being affected by the operation mode of other channels.

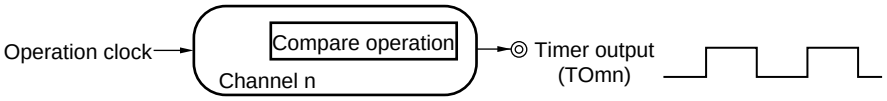
**(1) Interval timer**

Each timer of a unit can be used as a reference timer that generates an interrupt (INTTMmn) at fixed intervals.



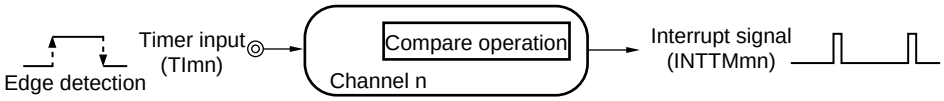

**(2) Square wave output**

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).



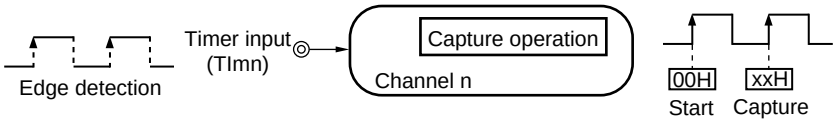

**(3) External event counter**

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) Input pulse interval measurement**

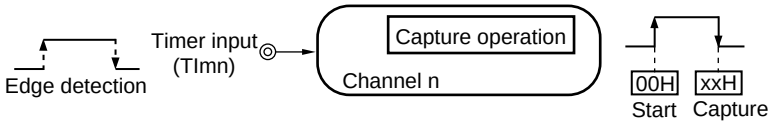
Counting is started by the valid edge of a pulse signal input to a timer input pin (TImn). The count value of the timer is captured at the valid edge of the next pulse. In this way, the interval of the input pulse can be measured.




(**Remark** is listed on the next page.)

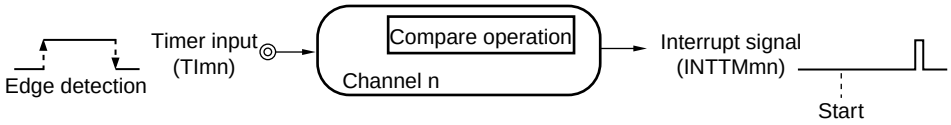
**(5) Measurement of high-/low-level width of input signal**

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 high-level or low-level width of the input signal can be measured.




**(6) 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.



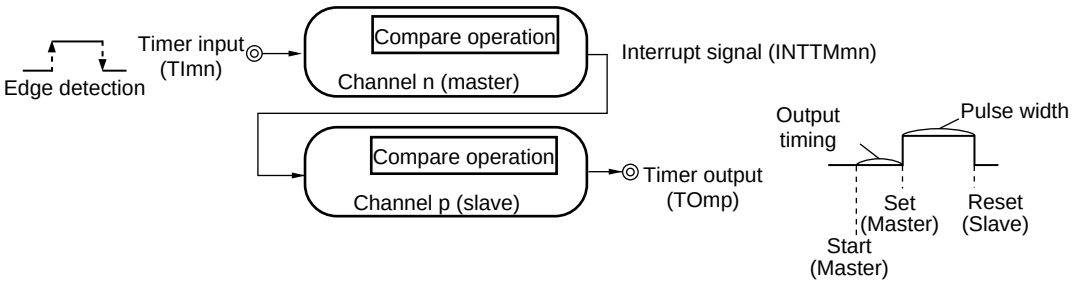

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

### 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 purposes.

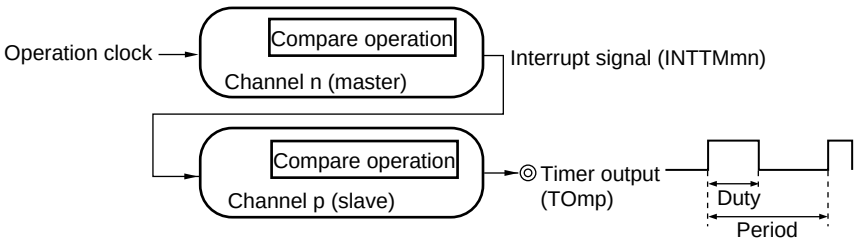
**(1) One-shot pulse output**

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**

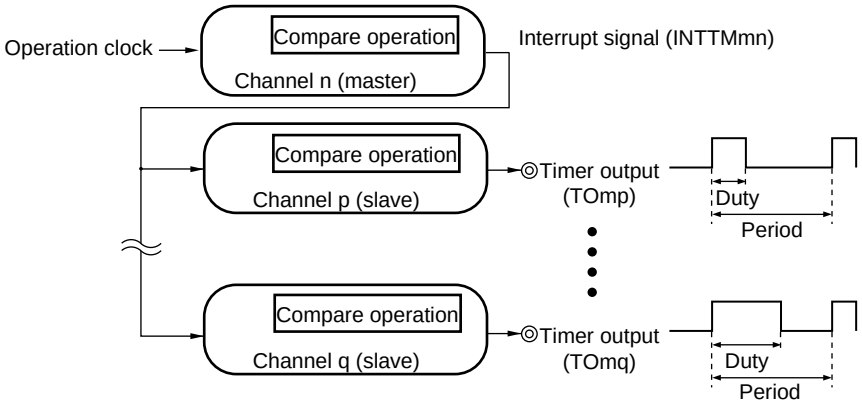
Two channels are used as a set to generate a pulse with a specified period and a specified duty factor.




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

**(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 rules of 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 operation function (channels 1 and 3 only)

The 8-bit timer operation function makes it possible to use a 16-bit timer channel in a configuration consisting of two 8-bit timer channels. This function can only be used for channels 1 and 3.

**Caution There are several rules for using 8-bit timer operation function.**
**For details, see 7.4.2 Basic rules of 8-bit timer operation function (channels 1 and 3 only).**

### 7.1.4 LIN-bus supporting function (channel 7 only)

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 (RxD0) of UART0 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 (RxD0) of UART0 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 (RxD0) of UART0 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 **7.3.13 Input switch control register (ISC)** and **7.8.4 Operation as input signal high-/low-level width measurement**.

## 7.2 Configuration of Timer Array Unit

Timer array unit includes the following hardware.

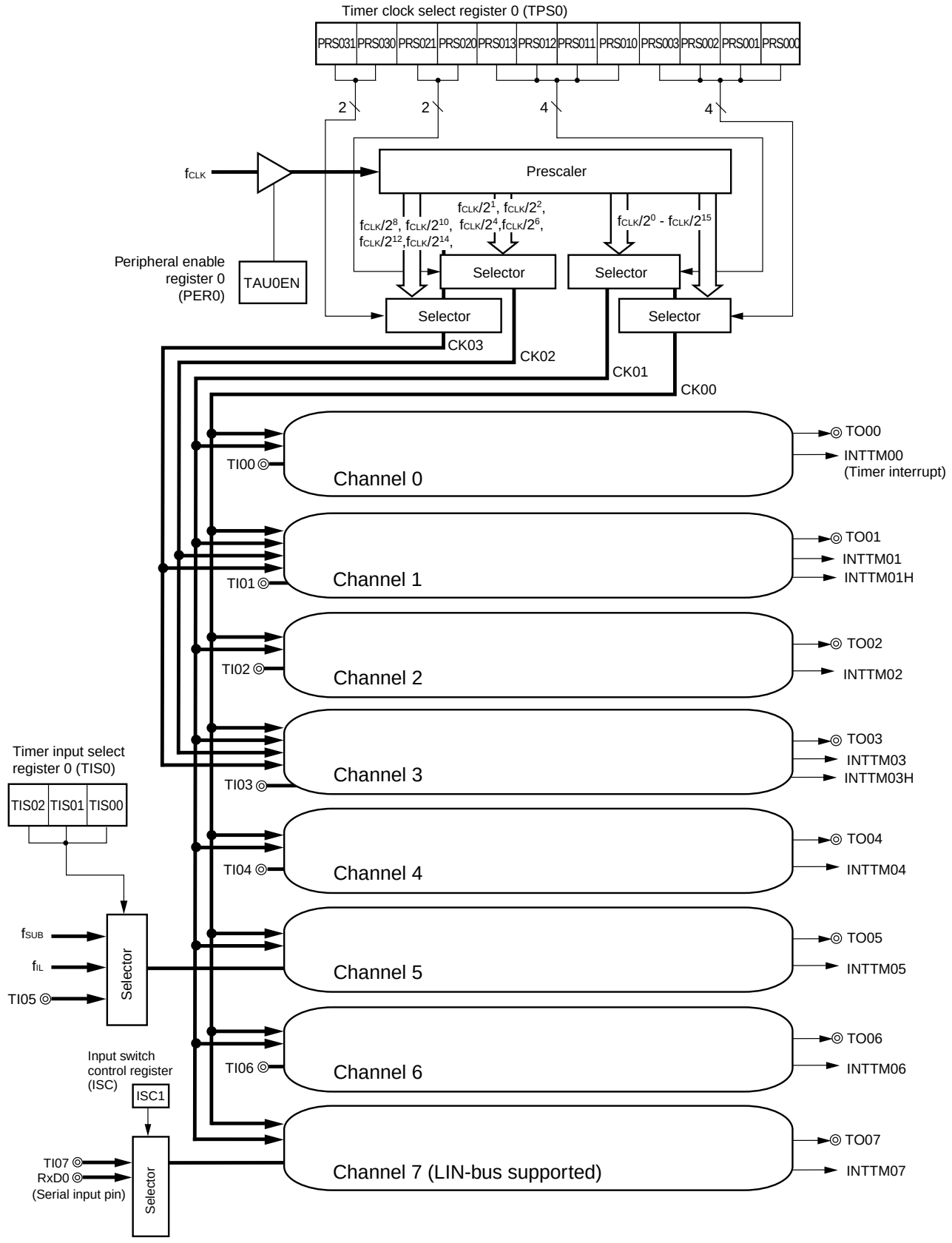
**Table 7-1. Configuration of Timer Array Unit**

| Item | Configuration |
|---|---|
| Timer/counter | Timer count register mn (TCRmn) |
| Register | Timer data register mn (TDRmn) |
| Timer input | TI00 to TI07, RxD0 pin (for LIN-bus) |
| Timer output | TO00 to TO07 pins, output controller |
| Control registers | <Registers of unit setting block><br>• Peripheral enable register 0 (PER0)<br>• Timer clock select register m (TPSm)<br>• Timer channel enable status register m (TEm)<br>• Timer channel start register m (TSm)<br>• Timer channel stop register m (TTm)<br>• Timer input select register 0 (TIS0)<br>• Timer output enable register m (TOEm)<br>• Timer output register m (TOm)<br>• Timer output level register m (TOLm)<br>• Timer output mode register m (TOMm) |
| | <Registers of each channel><br>• Timer mode register mn (TMRmn)<br>• Timer status register mn (TSRmn)<br>• Input switch control register (ISC)<br>• Noise filter enable register 1 (NFEN1)<br>• Port mode registers (PM0, PM3, PM4, PM6, PM12)<br>• Port registers (P0, P3, P4, P6, P12) |

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

Figure 7-1 shows the block diagrams of the timer array unit.

**Figure 7-1. Entire Configuration of Timer Array Unit**




**Remark** $f_{SUB}$: Subsystem clock frequency

$f_{IL}$: Low-speed on-chip oscillator clock frequency

**Figure 7-2. Internal Block Diagram of Channel 0 of Timer Array Unit**

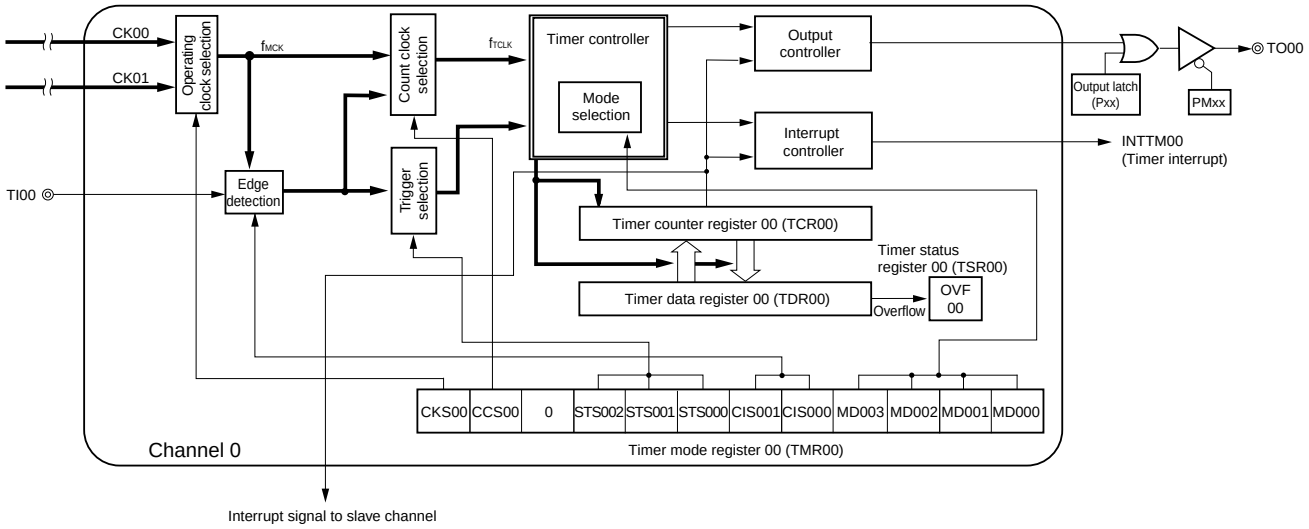
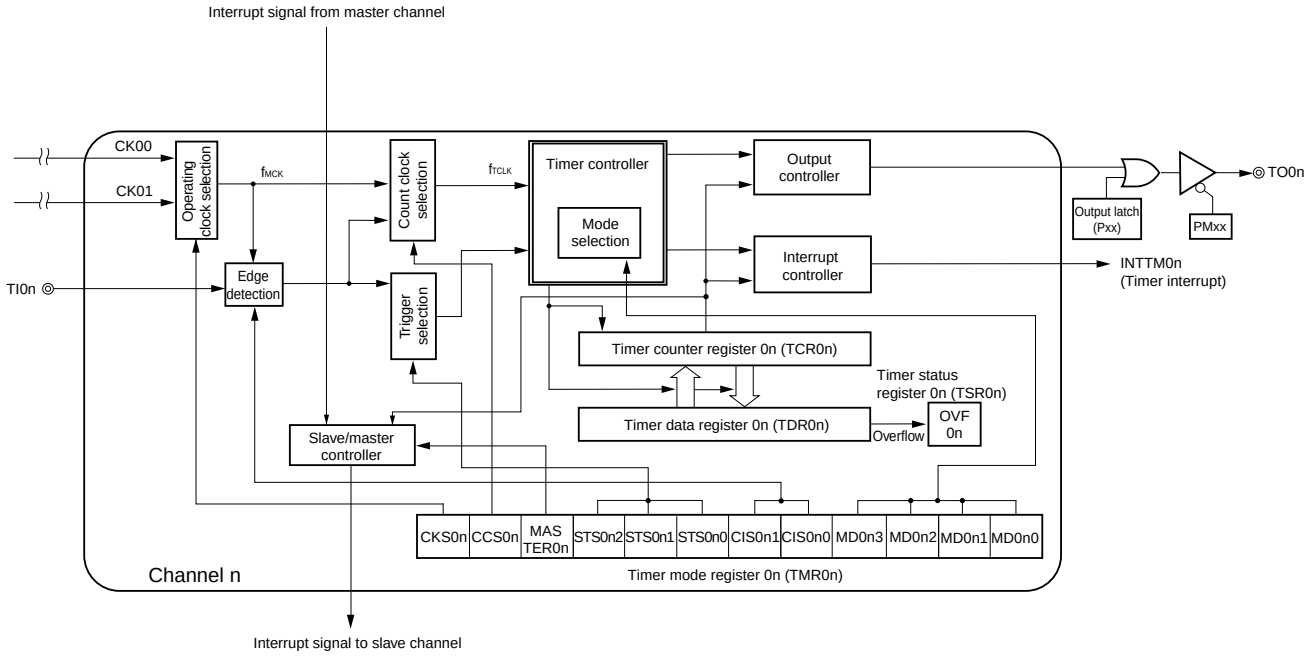
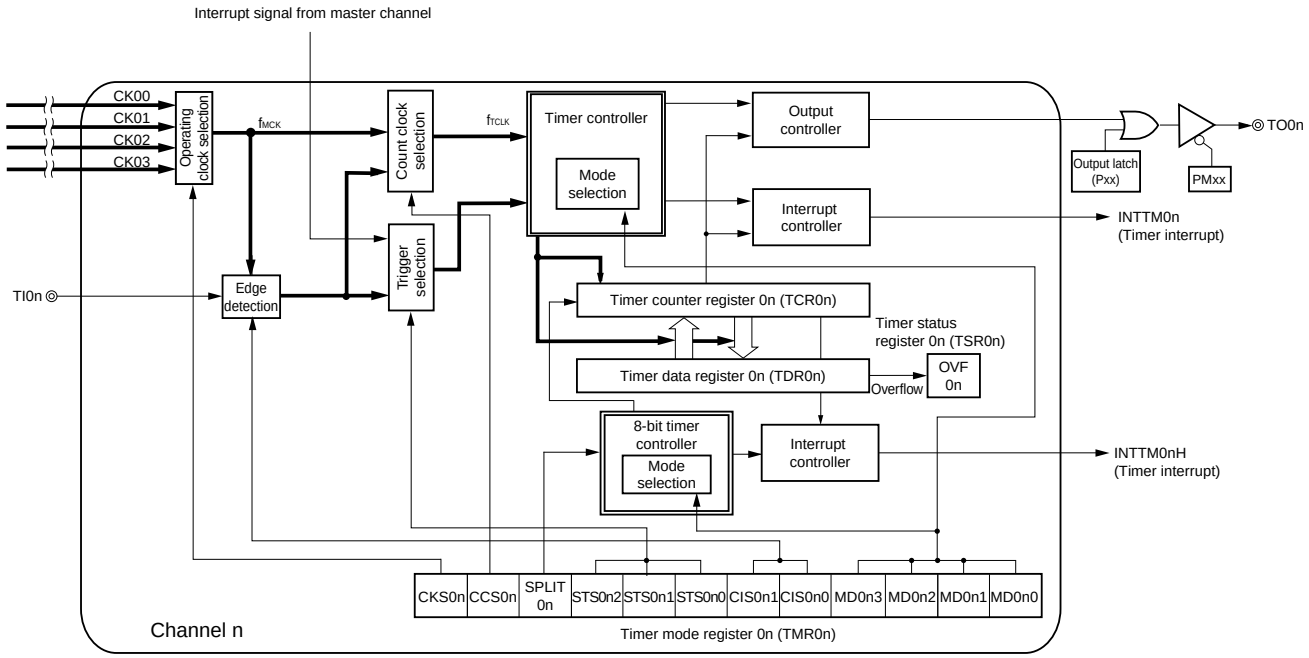



**Figure 7-3. Internal Block Diagram of Channels 2, 4, 6 of Timer Array Unit**




**Remark** n = 2, 4, 6

**Figure 7-4. Internal Block Diagram of Channels 1, 3 of Timer Array Unit**




**Remark** n = 1, 3

**Figure 7-5. Internal Block Diagram of Channel 5 of Timer Array Unit**

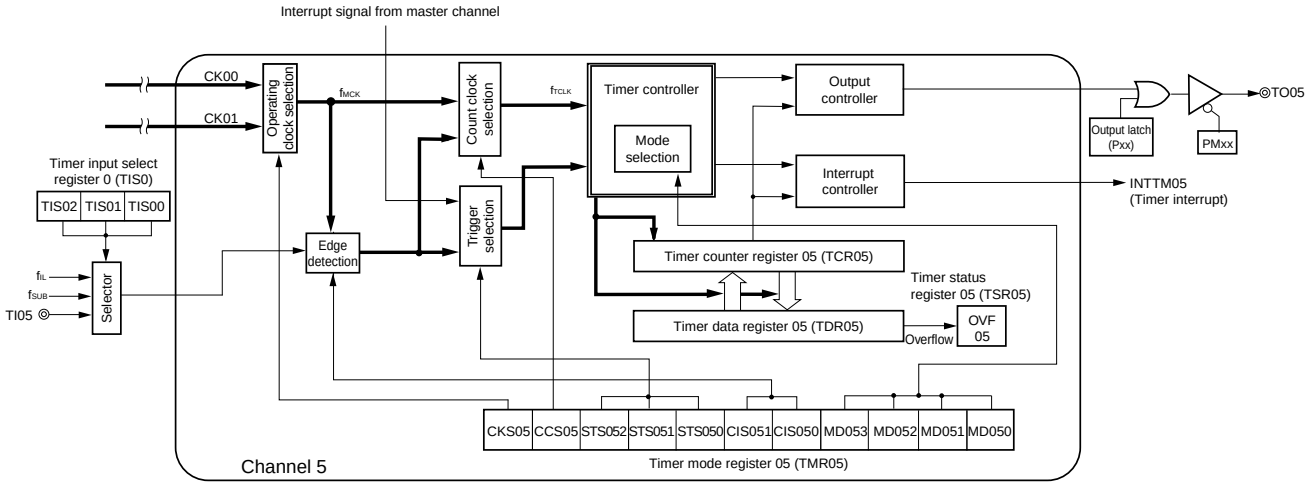
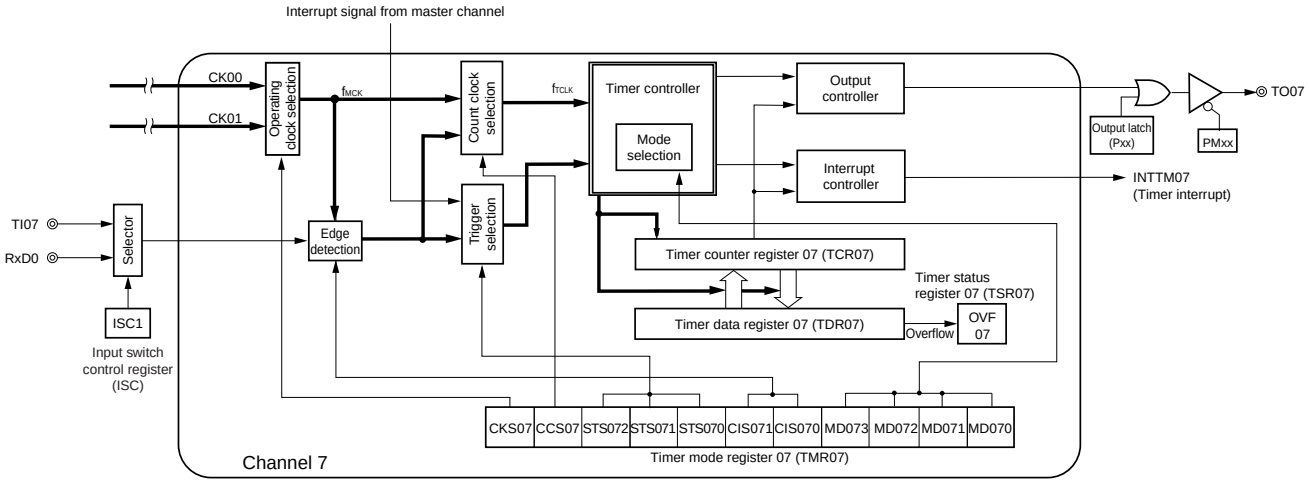


**Figure 7-6. Internal Block Diagram of Channel 7 of Timer Array Unit**




### 7.2.1 Timer count register mn (TCRmn)

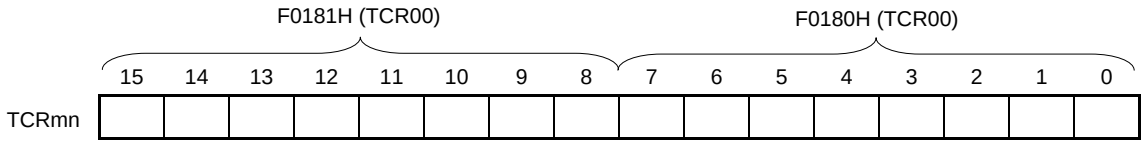
The TCRmn register is a 16-bit read-only register and is used to count clocks.

The value of this counter is incremented or decremented in synchronization with the rising edge of a count 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 **7.3.3 Timer mode register mn (TMRmn)**).

**Figure 7-7. Format of Timer Count Register mn (TCRmn)**

Address: F0180H, F0181H (TCR00) to F018EH, F018FH (TCR07) After reset: FFFFH R




**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

The count value can be read by reading timer count register mn (TCRmn).

The count value is set to FFFFH in the following cases.

- When the reset signal is generated
- When the TAUmEN bit of peripheral enable register 0 (PER0) 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 count 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 TCRmn register read value differs as follows according to operation mode changes and the operating status.

**Table 7-2. Timer Count Register mn (TCRmn) Read Value in Various Operation Modes**

| Operation Mode | Count Mode | Timer count register mn (TCRmn) Read Value[Note] | | | |
|---|---|---|---|---|---|
| | | Value if the operation mode was changed after releasing reset | Value if the Operation was restarted after count operation paused (TTmn = 1) | Value if the operation mode was changed after count operation paused (TTmn = 1) | Value when waiting for a start trigger after one count |
| Interval timer mode | Count down | FFFFH | Value if stop | Undefined | – |
| Capture mode | Count up | 0000H | Value if stop | Undefined | – |
| Event counter mode | Count down | FFFFH | Value if stop | Undefined | – |
| One-count mode | Count down | FFFFH | Value if stop | Undefined | FFFFH |
| Capture & one-count mode | Count up | 0000H | Value if stop | Undefined | Capture 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 SPLIT bits of timer mode registers m1 and m3 (TMRm1, TMRm3) are 1), it is possible to rewrite 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.

Reset signal generation clears this register to 0000H.

**Figure 7-8. 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

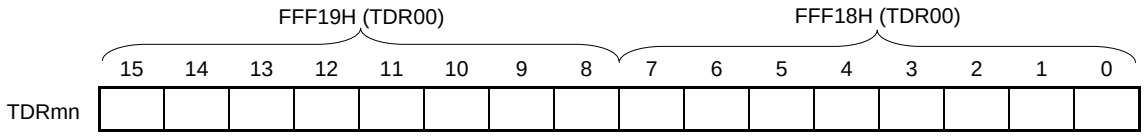
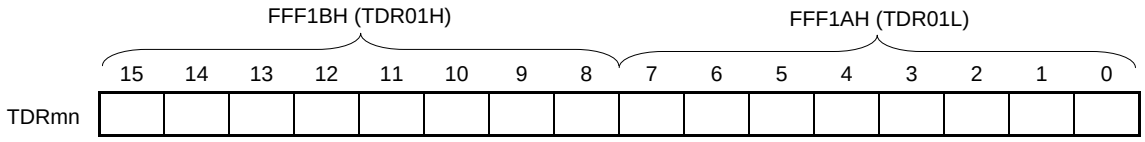



**Figure 7-9. Format of Timer Data Register mn (TDRmn) (n = 1, 3)**

Address: FFF1AH, FFF1BH (TDR01), FFF66H, FFF67H (TDR03) After reset: 0000H 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.

**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.**

**(ii) When timer data register mn (TDRmn) is used as capture register**

The count value of timer count 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).

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

## 7.3 Registers Controlling Timer Array Unit

Timer array unit is controlled by the following registers.

- Peripheral enable register 0 (PER0)
- 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 (PM0, PM3, PM4, PM6, PM12)
- Port registers (P0, P3, P4, P6, P12)

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

### 7.3.1 Peripheral enable register 0 (PER0)

This registers is used to enable or disable supplying the clock to the peripheral hardware. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

When the timer array unit is used, be sure to set bit 0 (TAU0EN) of this register to 1.

The PER0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 7-10. Format of Peripheral Enable Register 0 (PER0)**

Address: F00F0H After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PER0 | RTCWEN | IRDAEN | ADCEN | IICA0EN | SAU1EN | SAU0EN | 0 | TAU0EN |

| TAU0EN | Control of timer array unit input clock |
|---|---|
| 0 | Stops supply of input clock.<br>• SFR used by the timer array unit cannot be written.<br>• The timer array unit is in the reset status. |
| 1 | Supplies input clock.<br>• SFR used by the timer array unit can be read/written. |

**Cautions 1. When setting the timer array unit, be sure to set the following registers first while the TAUmEN bit is set to 1. If TAUmEN = 0, the values of the registers which control the timer array unit are cleared to their initial values and writing to them is ignored (except for the timer input select register 0 (TIS0), input switch control register (ISC), noise filter enable register 1 (NFEN1), port mode registers 0, 3, 4, 6, 12 (PM0, PM3, PM4, PM6, PM12), and port registers 0, 3, 4, 6, 12 (P0, P3, P4, P6, P12)).**

- **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)**

**2. Be sure to clear bit 1 to “0”.**

### 7.3.2 Timer clock select register m (TPSm)

The TPSm register is a 16-bit register that is used to select two types or four types of operation 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 operation clock (CKSmn1, CKSmn0 = 0, 0) are stopped (TEmn = 0).

If the PRSm10 to PRSm13 bits can be rewritten (n = 0 to 7):
All channels for which CKm2 is selected as the operation clock (CKSmn1, CKSmn0 = 0, 1) are stopped (TEmn = 0).

If the PRSm20 and PRSm21 bits can be rewritten (n = 1, 3):
All channels for which CKm1 is selected as the operation clock (CKSmn1, CKSmn0 = 1, 0) 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 operation clock (CKSmn1, CKSmn0 = 1, 1) are stopped (TEmn = 0).

The TPSm register can be set by a 16-bit memory manipulation instruction.

Reset signal generation clears this register to 0000H.

**Figure 7-11. Format of Timer Clock Select register m (TPSm) (1/2)**

Address: F01B6H, F01B7H After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TPSm | 0 | 0 | PRS m31 | PRS m30 | 0 | 0 | PRS m21 | PRS m20 | PRS m13 | PRS m12 | PRS m11 | PRS m10 | PRS m03 | PRS m02 | PRS m01 | PRS m00 |

| PRS mk3 | PRS mk2 | PRS mk1 | PRS mk0 | Selection of operation clock (CKmk)[Note] ( = 0, 1) | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | $f_{CLK}$ = 4 MHz | $f_{CLK}$ = 8 MHz | $f_{CLK}$ = 12 MHz | $f_{CLK}$ = 20 MHz | $f_{CLK}$ = 24 MHz |
| 0 | 0 | 0 | 0 | $f_{CLK}$ | 4 MHz | 8 MHz | 12 MHz | 20 MHz | 24 MHz |
| 0 | 0 | 0 | 1 | $f_{CLK}/2$ | 2 MHz | 4 MHz | 6 MHz | 10 MHz | 12 MHz |
| 0 | 0 | 1 | 0 | $f_{CLK}/2^2$ | 1 MHz | 2 MHz | 3 MHz | 5 MHz | 6 MHz |
| 0 | 0 | 1 | 1 | $f_{CLK}/2^3$ | 500 kHz | 1 MHz | 1.5 MHz | 2.5 MHz | 3 MHz |
| 0 | 1 | 0 | 0 | $f_{CLK}/2^4$ | 250 kHz | 500 kHz | 750 kHz | 1.25 MHz | 1.5 MHz |
| 0 | 1 | 0 | 1 | $f_{CLK}/2^5$ | 125 kHz | 250 kHz | 375 kHz | 625 kHz | 750 kHz |
| 0 | 1 | 1 | 0 | $f_{CLK}/2^6$ | 62.5 kHz | 125 kHz | 188 kHz | 313 kHz | 375 kHz |
| 0 | 1 | 1 | 1 | $f_{CLK}/2^7$ | 31.3 kHz | 62.5 kHz | 93.8 kHz | 156 kHz | 188 kHz |
| 1 | 0 | 0 | 0 | $f_{CLK}/2^8$ | 15.6 kHz | 31.3 kHz | 46.9 kHz | 78.1 kHz | 93.8 kHz |
| 1 | 0 | 0 | 1 | $f_{CLK}/2^9$ | 7.81 kHz | 15.6 kHz | 23.4 kHz | 39.1 kHz | 46.9 kHz |
| 1 | 0 | 1 | 0 | $f_{CLK}/2^{10}$ | 3.91 kHz | 7.81 kHz | 11.7 kHz | 19.5 kHz | 23.4 kHz |
| 1 | 0 | 1 | 1 | $f_{CLK}/2^{11}$ | 1.95 kHz | 3.91 kHz | 5.86 kHz | 9.76 kHz | 11.7 kHz |
| 1 | 1 | 0 | 0 | $f_{CLK}/2^{12}$ | 976 Hz | 1.95 kHz | 2.93 kHz | 4.88 kHz | 5.86 kHz |
| 1 | 1 | 0 | 1 | $f_{CLK}/2^{13}$ | 488 Hz | 976 Hz | 1.46 kHz | 2.44 kHz | 2.93 kHz |
| 1 | 1 | 1 | 0 | $f_{CLK}/2^{14}$ | 244 Hz | 488 Hz | 732 Hz | 1.22 kHz | 1.46 kHz |
| 1 | 1 | 1 | 1 | $f_{CLK}/2^{15}$ | 122 Hz | 244 Hz | 366 Hz | 610 Hz | 732 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).

**Cautions 1. Be sure to clear bits 15, 14, 11, and 10 to "0".**

**2. If $f_{CLK}$ (undivided) is selected as the operation clock (CKmk) and TDRnm is set to 0000H (n = 0, m = 0 to 7), interrupt requests output from timer array units cannot be used.**

**Remarks 1.** $f_{CLK}$: CPU/peripheral hardware clock frequency

**2.** The above $f_{CLK}/2^r$ is not a signal which is simply divided $f_{CLK}$ by $2^r$, but a signal which becomes high level for one period of $f_{CLK}$ from its rising edge (r = 1 to 15). For details, see **7.5.1 Count clock ($f_{TCLK}$)**.

**Figure 7-11. Format of Timer Clock Select register m (TPSm) (2/2)**

Address: F01B6H, F01B7H After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TPSm | 0 | 0 | PRS m31 | PRS m30 | 0 | 0 | PRS m21 | PRS m20 | PRS m13 | PRS m12 | PRS m11 | PRS m10 | PRS m03 | PRS m02 | PRS m01 | PRS m00 |

| PRS m21 | PRS m20 | Selection of operation clock (CKm2)[Note] | | | | | |
|---|---|---|---|---|---|---|---|
| | | | $f_{CLK}$ = 4 MHz | $f_{CLK}$ = 8 MHz | $f_{CLK}$ = 12 MHz | $f_{CLK}$ = 20 MHz | $f_{CLK}$ = 24 MHz |
| 0 | 0 | $f_{CLK}/2$ | 2 MHz | 4 MHz | 6 MHz | 10 MHz | 12 MHz |
| 0 | 1 | $f_{CLK}/2^2$ | 1 MHz | 2 MHz | 3 MHz | 5 MHz | 6 MHz |
| 1 | 0 | $f_{CLK}/2^4$ | 250 kHz | 500 kHz | 750 kHz | 1.25 MHz | 1.5 MHz |
| 1 | 1 | $f_{CLK}/2^6$ | 62.5 kHz | 125 kHz | 188 kHz | 313 kHz | 375 kHz |

| PRS m31 | PRS m30 | Selection of operation clock (CKm3)[Note] | | | | | |
|---|---|---|---|---|---|---|---|
| | | | $f_{CLK}$ = 4 MHz | $f_{CLK}$ = 8 MHz | $f_{CLK}$ = 12 MHz | $f_{CLK}$ = 20 MHz | $f_{CLK}$ = 24 MHz |
| 0 | 0 | $f_{CLK}/2^8$ | 15.6 kHz | 31.3 kHz | 46.9 kHz | 78.1 kHz | 93.8 kHz |
| 0 | 1 | $f_{CLK}/2^{10}$ | 3.91 kHz | 7.81 kHz | 11.7 kHz | 19.5 kHz | 23.4 kHz |
| 1 | 0 | $f_{CLK}/2^{12}$ | 976 Hz | 1.95 kHz | 2.93 kHz | 4.88 kHz | 5.86 kHz |
| 1 | 1 | $f_{CLK}/2^{14}$ | 244 Hz | 488 Hz | 732 Hz | 1.22 kHz | 1.46 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 ($f_{MCK}$) specified by using the CKSmn0, and CKSmn1 bits or the valid edge of the signal input from the TImn pin is selected as the count clock ($f_{TCLK}$).

**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 operation clock, the interval times shown in Table 7-3 can be achieved by using the interval timer function.

**Table 7-3. Interval Times Available for Operation Clock CKSm2 or CKSm3**

| Clock | | Interval time[Note] ($f_{CLK}$ = 20 MHz) | | | |
|---|---|---|---|---|---|
| | | 16 μs | 160 μs | 1.6 ms | 16 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 %.

**Remarks 1.** $f_{CLK}$: CPU/peripheral hardware clock frequency
**2.** For details of a signal of $f_{CLK}/2^j$ selected with the TPSm register, see **7.5.1 Count clock ($f_{TCLK}$).**

### 7.3.3 Timer mode register mn (TMRmn)

The TMRmn register sets an operation mode of channel n. This register is used to select the operation clock ($f_{MCK}$), select the count 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.

Reset signal generation clears this register to 0000H.

**Caution The bits mounted depend on the channels in the bit 11 of TMRmn register.**
**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) (1/4)**

Address: F0190H, F0191H (TMR00) to F019EH, F019FH (TMR07) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 2, 4, 6 ) | CKS mn1 | CKS mn0 | 0 | CCS mn | MAST ERmn | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 1, 3) | CKS mn1 | CKS mn0 | 0 | CCS mn | SPLIT mn | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 0, 5, 7) | CKS mn1 | CKS mn0 | 0 | CCS mn | 0Note | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

| CKS mn1 | CKS mn0 | Selection of operation clock ($f_{MCK}$) of channel n |
|---|---|---|
| 0 | 0 | Operation clock CKm0 set by timer clock select register m (TPSm) |
| 0 | 1 | Operation clock CKm2 set by timer clock select register m (TPSm) |
| 1 | 0 | Operation clock CKm1 set by timer clock select register m (TPSm) |
| 1 | 1 | Operation clock CKm3 set by timer clock select register m (TPSm) |

Operation clock ($f_{MCK}$ ) is used by the edge detector. A count clock ($f_{TCLK}$) and a sampling clock are generated depending on the setting of the CCSmn bit.
The operation clocks CKm2 and CKm3 can only be selected for channels 1 and 3.

| CCS mn | Selection of count clock ($f_{TCLK}$) of channel n |
|---|---|
| 0 | Operation clock ($f_{MCK}$) specified by the CKSmn0 and CKSmn1 bits |
| 1 | Valid edge of input signal input from the TImn pin<br>In channel 5, Valid edge of input signal selected by TIS0<br>In channel 7, Valid edge of input signal selected by ISC |

Count clock ($f_{TCLK}$) is used for the counter, output controller, and interrupt controller.

<R>

**Note** Bit 11 is fixed at 0 of read only, write is ignored.

**Cautions 1. Be sure to clear bits 13, 5, and 4 to "0".**

**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 CKSmn0 and CKSmn1 bits ($f_{MCK}$) or the valid edge of the signal input from the TImn pin is selected as the count clock ($f_{TCLK}$).**

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

**Figure 7-12. Format of Timer Mode Register mn (TMRmn) (2/4)**

Address: F0190H, F0191H (TMR00) to F019EH, F019FH (TMR07) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 2, 4, 6 ) | CKS mn1 | CKS mn0 | 0 | CCS mn | MASTERmn | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 1, 3) | CKS mn1 | CKS mn0 | 0 | CCS mn | SPLIT mn | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 0, 5, 7) | CKS mn1 | CKS mn0 | 0 | CCS mn | 0[Note] | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

(Bit 11 of TMRmn (n = 2, 4, 6))

| MASTER mn | 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).<br>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).<br>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.<br>(Operates in independent channel operation function or as slave channel in simultaneous channel operation function.) |
| 1 | Operates as 8-bit timer. |

| STS mn2 | STS mn1 | STS mn0 | 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 |

**Note** Bit 11 is fixed at 0 of read only, write is ignored.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

**Figure 7-12. Format of Timer Mode Register mn (TMRmn) (3/4)**

Address: F0190H, F0191H (TMR00) to F019EH, F019FH (TMR07) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 2, 4, 6 ) | CKS mn1 | CKS mn0 | 0 | CCS mn | MAST ERmn | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 1, 3) | CKS mn1 | CKS mn0 | 0 | CCS mn | SPLIT mn | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 0, 5, 7) | CKS mn1 | CKS mn0 | 0 | CCS mn | 0[Note] | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

| CIS mn1 | CIS mn0 | 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)<br>Start trigger: Falling edge, Capture trigger: Rising edge |
| 1 | 1 | Both edges (when high-level width is measured)<br>Start trigger: Rising edge, Capture trigger: Falling edge |
| If both the edges are specified when the value of the STSmn2 to STSmn0 bits is other than 010B, set the CISmn1 to CISmn0 bits to 10B. | | |

**Note** Bit 11 is fixed at 0 of read only, write is ignored.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

**Figure 7-12. Format of Timer Mode Register mn (TMRmn) (4/4)**

Address: F0190H, F0191H (TMR00) to F019EH, F019FH (TMR07) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 2, 4, 6 ) | CKS mn1 | CKS mn0 | 0 | CCS mn | MAST ERmn | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 1, 3) | CKS mn1 | CKS mn0 | 0 | CCS mn | SPLIT mn | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 0, 5, 7) | CKS mn1 | CKS mn0 | 0 | CCS mn | 0[Note 1] | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

| 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/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 high-/low-level width of input signal | Counting up |
| Other than above | | | Setting prohibited | | |
| The operation of each mode varies depending on MDmn0 bit (see the table below). | | | | | |

| Operation mode (Value set by the MDmn3 to MDmn1 bits (see table above)) | MD mn0 | Setting of starting counting and interrupt |
|---|---|---|
| • Interval timer mode (0, 0, 0) | 0 | Timer interrupt is not generated when counting is started (timer output does not change, either). |
| • Capture mode (0, 1, 0) | 1 | Timer interrupt is generated when counting is started (timer output also changes). |
| • Event counter mode (0, 1, 1) | 0 | Timer interrupt is not generated when counting is started (timer output does not change, either). |
| • One-count mode[Note 2] (1, 0, 0) | 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 (1, 1, 0) | 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. |

(**Notes** and **Remark** are listed on the next page.)

**Notes 1.** Bit 11 is fixed at 0 of read only, write is ignored.

**2.** In one-count mode, interrupt output (INTTMmn) when starting a count operation and TOmn output are not controlled.

**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).

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

### 7.3.4 Timer status register mn (TSRmn)

The TSRmn register indicates the overflow status 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 7-4** 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 lower 8 bits of the TSRmn register can be set with an 8-bit memory manipulation instruction with TSRmnL.

Reset signal generation clears this register to 0000H.

**Figure 7-13. Format of Timer Status Register mn (TSRmn)**

Address: F01A0H, F01A1H (TSR00) to F01AEH, F01AFH (TSR07) After reset: 0000H R

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TSRmn | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | OVF |

| OVF | Counter overflow status 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 7-4. OVF Bit Operation and Set/Clear Conditions in Each Operation Mode**

| Timer Operation Mode | OVF Bit | Set/clear Conditions |
|---|---|---|
| • Capture mode<br>• Capture & one-count mode | clear | When no overflow has occurred upon capturing |
| | set | When an overflow has occurred upon capturing |
| • Interval timer mode<br>• Event counter mode<br>• One-count mode | clear | – |
| | set | (Use prohibited) |

**Remark** The OVF bit does not change immediately after the counter has overflowed, but changes upon the subsequent capture.

### 7.3.5 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 lower 8 bits of the TEm register can be set with a 1-bit or 8-bit memory manipulation instruction with TEmL.

Reset signal generation clears this register to 0000H.

**Figure 7-14. Format of Timer Channel Enable Status register m (TEm)**

Address: F01B0H, F01B1H After reset: 0000H R

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TEm | 0 | 0 | 0 | 0 | TEHm3 | 0 | TEHm1 | 0 | TEm7 | TEm6 | TEm5 | TEm4 | TEm3 | TEm2 | TEm1 | TEm0 |

| TEH03 | 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. |

| TEH01 | 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 enable/stop status 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)

### 7.3.6 Timer channel start register m (TSm)

The TSm register is a trigger register that is used to initialize timer count 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 lower 8 bits of the TSm register can be set with a 1-bit or 8-bit memory manipulation instruction with TSmL.

Reset signal generation clears this register to 0000H.

**Figure 7-15. Format of Timer Channel Start register m (TSm)**

Address: F01B2H, F01B3H After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TSm | 0 | 0 | 0 | 0 | TSHm3 | 0 | TSHm1 | 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.<br>The TCRm3 register count operation start in the interval timer mode in the count operation enabled state (see **Table 7-5** in **7.5.2 Start timing of counter**). |

| 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.<br>The TCRm1 register count operation start in the interval timer mode in the count operation enabled state (see **Table 7-5** in **7.5.2 Start timing of counter**). |

| 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.<br>The TCRmn register count operation start in the count operation enabled state varies depending on each operation mode (see **Table 7-5** in **7.5.2 Start timing of counter**).<br>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. |

**Cautions 1. Be sure to clear bits 15 to 12, 10, 8 to "0"**

**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 (TNFENnm = 1): Four cycles of the operation clock ($f_{MCK}$)**

**When the TImn pin noise filter is disabled (TNFENnm = 0): Two cycles of the operation clock ($f_{MCK}$)**

**Remarks 1.** When the TSm register is read, 0 is always read.

**2.** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

### 7.3.7 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, TTHm1, TTHm3 = 0), because they are trigger bits.

The TTm register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the TTm register can be set with a 1-bit or 8-bit memory manipulation instruction with TTmL.

Reset signal generation clears this register to 0000H.

**Figure 7-16. Format of Timer Channel Stop register m (TTm)**

Address: F01B4H, F01B5H After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TTm | 0 | 0 | 0 | 0 | TTHm3 | 0 | TTHm1 | 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 clear to 0, to be count operation stop enable status.<br>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, 8 of the TTm register to “0”.**

**Remarks 1.** When the TTm register is read, 0 is always read.

**2.** m: Unit number (m = 0),n: Channel number (n = 0 to 7)

### 7.3.8 Timer input select register 0 (TIS0)

The TIS0 register is used to select the channel 5 timer input.

The TIS0 register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 7-17. Format of Timer Input Select register 0 (TIS0)**

Address: F0074H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TIS0 | 0 | 0 | 0 | 0 | 0 | TIS02 | TIS01 | TIS00 |

| TIS02 | TIS01 | TIS00 | Selection of timer input used with channel 5 |
|---|---|---|---|
| 0 | 0 | 0 | Input signal of timer input pin (TI05) |
| 0 | 0 | 1 | |
| 0 | 1 | 0 | |
| 0 | 1 | 1 | |
| 1 | 0 | 0 | Low-speed on-chip oscillator clock ($f_{IL}$) |
| 1 | 0 | 1 | Subsystem clock ($f_{SUB}$) |
| Other than above | | | Setting prohibited |

<R> **Caution High-level width, low-level width of timer input is selected, will require more than $1/f_{MCK}$ +10 ns. Therefore, when selecting $f_{SUB}$ to $f_{CLK}$ (CSS bit of CKC register = 1), can not TIS02 bit set to 1.**

### 7.3.9 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 lower 8 bits of the TOEm register can be set with a 1-bit or 8-bit memory manipulation instruction with TOEmL.

Reset signal generation clears this register to 0000H.

**Figure 7-18. Format of Timer Output Enable register m (TOEm)**

Address: F01BAH, F01BBH After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TOEm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | TOEm7 | TOEm6 | TOEm5 | TOEm4 | TOEm3 | TOEm2 | TOEm1 | TOEm0 |

| TOEmn | Timer output enable/disable of channel n |
|---|---|
| 0 | Disable output of timer.<br>Without reflecting on TOmn bit timer operation, to fixed the output.<br>Writing to the TOmn bit is enabled and the level set in the TOmn bit is output from the TOmn pin. |
| 1 | Enable output of timer.<br>Reflected in the TOmn bit timer operation, to generate the output waveform.<br>Writing to the TOmn bit is disabled (writing 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)

### 7.3.10 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 oh 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 P43/TI00/TO00, P41/TI01/TO01, P07/TI02/TO02, P06/TI03/TO03, P05/TI04/TO04, P04/TI05/TO05, P03/TI06/TO06, or P02/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 lower 8 bits of the TOm register can be set with an 8-bit memory manipulation instruction with TOmL.

Reset signal generation clears this register to 0000H.

**Figure 7-19. Format of Timer Output register m (TOm)**

Address: F01B8H, F01B9H After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TOm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 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)

### 7.3.11 Timer output level register m (TOLm)

The TOLm register is a register that 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 lower 8 bits of the TOLm register can be set with an 8-bit memory manipulation instruction with TOLmL.

Reset signal generation clears this register to 0000H.

**Figure 7-20. Format of Timer Output Level register m (TOLm)**

Address: F01BCH, F01BDH After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TOLm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 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".**

**Remarks 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.

**2.** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

### 7.3.12 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 lower 8 bits of the TOMm register can be set with an 8-bit memory manipulation instruction with TOMmL.

Reset signal generation clears this register to 0000H.

**Figure 7-21. Format of Timer Output Mode register m (TOMm)**

Address: F01BEH, F01BFH After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TOMm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 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 toggle 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
$n < p \leq 7$
(For details of the relation between the master channel and slave channel, see **7.4.1 Basic rules of simultaneous channel operation function**.)

### 7.3.13 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 (RxD0) is selected as a timer input signal.

The ISC register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 7-22. Format of Input Switch Control Register (ISC)**

Address: F0073H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ISC | 0 | 0 | 0 | 0 | 0 | 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 RxD0 pin is used as timer input (detects the wakeup signal and measures the low 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 RxD0 pin as an external interrupt (wakeup signal detection). |

**Caution Be sure to clear bits 7 to 2 to "0".**

**Remark** When the LIN-bus communication function is used, select the input signal of the RxD0 pin by setting ISC1 to 1.

### 7.3.14 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.

Reset signal generation clears this register to 00H.

**Note** For details, see **7.5.1 (2) When valid edge of input signal via the TImn pin is selected (CCSmn = 1)**, **7.5.2 Start timing of counter**, and **7.7 Timer Input (TImn) Control**.

**Figure 7-23. Format of Noise Filter Enable Register 1 (NFEN1)**

Address: F0071H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| NFEN1 | TNFEN07 | TNFEN06 | TNFEN05 | TNFEN04 | TNFEN03 | TNFEN02 | TNFEN01 | TNFEN00 |

| TNFEN07 | Enable/disable using noise filter of TI07 pin or RxD0 pin input signal[Note] |
|---|---|
| 0 | Noise filter OFF |
| 1 | Noise filter ON |

| TNFEN06 | Enable/disable using noise filter of TI06 pin input signal |
|---|---|
| 0 | Noise filter OFF |
| 1 | Noise filter ON |

| TNFEN05 | Enable/disable using noise filter of TI05 pin input signal |
|---|---|
| 0 | Noise filter OFF |
| 1 | Noise filter ON |

| TNFEN04 | Enable/disable using noise filter of TI04 pin input signal |
|---|---|
| 0 | Noise filter OFF |
| 1 | Noise filter ON |

| TNFEN03 | Enable/disable using noise filter of TI03 pin input signal |
|---|---|
| 0 | Noise filter OFF |
| 1 | Noise filter ON |

| TNFEN02 | Enable/disable using noise filter of TI02 pin input signal |
|---|---|
| 0 | Noise filter OFF |
| 1 | Noise filter ON |

| TNFEN01 | Enable/disable using noise filter of TI01 pin input signal |
|---|---|
| 0 | Noise filter OFF |
| 1 | Noise filter ON |

| TNFEN00 | Enable/disable using noise filter of TI00 pin input signal |
|---|---|
| 0 | Noise filter OFF |
| 1 | Noise filter ON |

**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 RxD0 pin can be selected.

### 7.3.15 Registers controlling port functions of pins to be used for timer I/O

Using port pins for the timer array unit functions requires setting of the registers that control the port functions multiplexed on the target pins (port mode register (PMxx) and port register (Pxx)). For details, see **4.3.1 Port mode registers (PMxx)** and **4.3.2 Port registers (Pxx)**.

The port mode register (PMxx) and port register (Pxx) to be set depend on the product. For details, see **4.5 Register Settings When Using Alternate Function**.

When using the ports (such as P43/TI00/TO00) to be shared with the timer output pin for timer output, set the port mode register (PMxx) bit and port register (Pxx) bit corresponding to each port to 0.

Example: When using P43/ TO00 for timer output
Set the PM43 bit of port mode register 4 to 0.
Set the P43 bit of port register 4 to 0.

When using the ports (such as P43/TI00) to be shared with the timer input pin for timer input, set the port mode register (PMxx) bit corresponding to each port to 1. At this time, the port register (Pxx) bit may be 0 or 1.

Example: When using P43/TO00 for timer input
Set the PM43 bit of port mode register 4 to 1.
Set the P43 bit of port register 4 to 0 or 1.

**Remarks 1.** In case of 80-pin product, in order to use a port that is shared with segment output for timer I/O function, be sure to set the corresponding bits of LCD port function register 4 (PFSEG4) bits PFSEG32 to PFSEG37 to “0”.

**2.** When using the P125/(TI05)/(TO05)/$V_{L3}$ pin for timer I/O, be sure to clear the ISCVL3 bit of the LCD Input switch control register (ISCLCD) to “0”.

**3.** When using the P126/(TI04)/(TO04)/CAPL and P127/(TI03)/(TO03)/CAPH pins for timer I/O, be sure to clear the ISCCAP bit of the LCD Input switch control register (ISCLCD) to “1”.

## 7.4 Basic Rules of Timer Array Unit

### 7.4.1 Basic rules of 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 rules apply.

(1) Only an even channel (channel 0, 2, 4, etc.) 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.
Example: If channel 2 is set as a master channel, channel 3 or those that follow (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.
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. Channels 5 to 7 cannot be set as the slave channels of master channel 0.

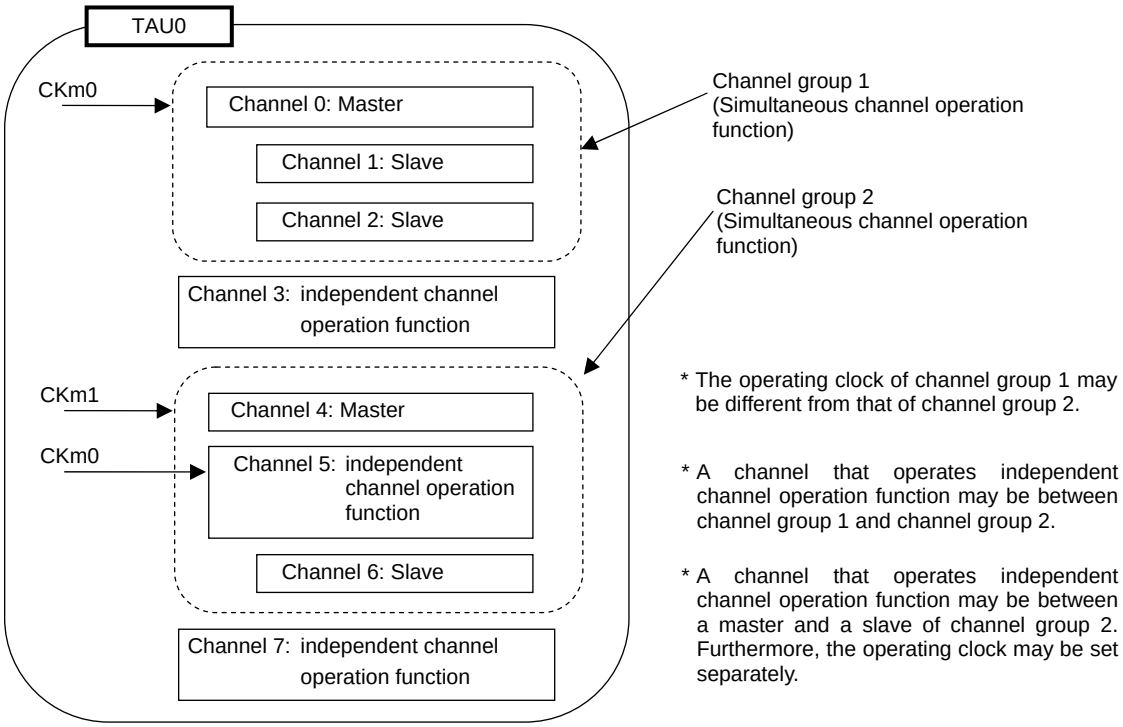
(6) The operating clock for a slave channel in combination with a master channel must be the same as that of the master channel. The CKSmn0, CKSmn1 bits (bit 15, 14 of timer mode register mn (TMRmn)) of the slave channel that operates in combination with the master channel must be the same value as that of the master channel.
(7) A master channel can transmit INTTMmn (interrupt), start software trigger, and count clock to the lower channels.
(8) A slave channel can use INTTMmn (interrupt), a start software trigger, or the count clock of the master channel as a source clock, but cannot transmit its own INTTMmn (interrupt), start software trigger, or count clock to channels with lower channel numbers.
(9) A master channel cannot use INTTMmn (interrupt), a start software trigger, or the count clock from the other higher master channel as a source clock.
(10) To simultaneously start channels that operate in combination, the channel 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 channel 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 as “0”). However, as channel 0 is the highest channel, it can be used as a master channel during simultaneous operation.

The rules 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 rules of the simultaneous channel operation function in **7.4.1 Basic rules of 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




* 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 rules of 8-bit timer operation function (channels 1 and 3 only)

The 8-bit timer operation function makes it possible to use a 16-bit timer channel in a configuration consisting of two 8-bit timer channels.

This function can only be used for channels 1 and 3, and there are several rules for using it.

The basic rules for this function are as follows:

(1) The 8-bit timer operation function applies only to channels 1 and 3.
(2) When using 8-bit timers, set the SPLIT bit of timer mode register mn (TMRmn) to 1.
(3) The higher 8 bits can be operated as the interval timer function.
(4) At the start of operation, the higher 8 bits output INTTMm1H/INTTMm3H (an interrupt) (which is the same operation performed when MDmn0 is set to 1).
(5) The operation clock of the higher 8 bits is selected according to the CKSmn1 and CKSmn0 bits of the lower-bit TMRmn register.
(6) For the higher 8 bits, the TSHm1/TSHm3 bit is manipulated to start channel operation and the TTHm1/TTHm3 bit is manipulated to stop channel operation. The channel status can be checked using the TEHm1/TEHm3 bit.
(7) The lower 8 bits operate according to the TMRmn register settings. The following three functions support operation of the lower 8 bits:
- Interval timer function
- External event counter function
- Delay count function

(8) For the lower 8 bits, the TSm1/TSm3 bit is manipulated to start channel operation and the TTm1/TTm3 bit is manipulated to stop channel operation. The channel status can be checked using the TEm1/TEm3 bit.
(9) During 16-bit operation, manipulating the TSHm1, TSHm3, TTHm1, and TTHm3 bits is invalid. The TSm1, TSm3, TTm1, and TTm3 bits are manipulated to operate channels 1 and 3. The TEHm3 and TEHm1 bits are not changed.
(10) For the 8-bit timer function, the simultaneous operation functions (one-shot pulse, PWM, and multiple PWM) cannot be used.

**Remark** m: Unit number (m = 0), n: Channel number (n = 1, 3)

## 7.5 Operation of Counter

### 7.5.1 Count clock ($f_{TCLK}$)

The count clock ($f_{TCLK}$) of the timer array unit can be selected between following by CCSmn bit of timer mode register mn (TMRmn).

- Operation clock ($f_{MCK}$) specified by the CKSmn0 and CKSmn1 bits
- Valid edge of input signal input from the TImn pin

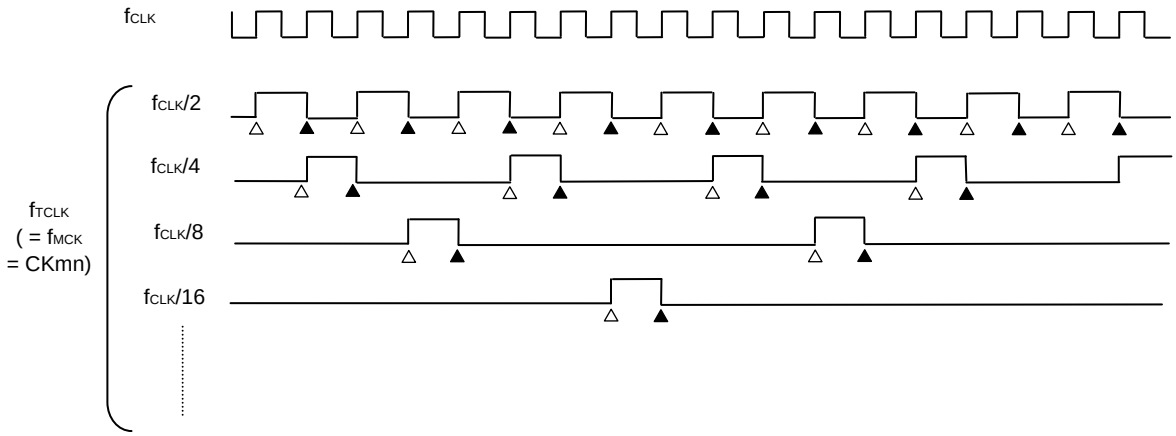
Because the timer array unit is designed to operate in synchronization with $f_{CLK}$, the timings of the count clock ($f_{TCLK}$) are shown below.

**(1) When operation clock ($f_{MCK}$) specified by the CKSmn0 and CKSmn1 bits is selected (CCSmn = 0)**

The count clock ($f_{TCLK}$) is between $f_{CLK}$ to $f_{CLK}/2^{15}$ by setting of timer clock select register m (TPSm). When a divided $f_{CLK}$ is selected, however, the clock selected in TPSmn register, but a signal which becomes high level for one period of $f_{CLK}$ from its rising edge. When a $f_{CLK}$ is selected, fixed to high level

Counting of timer count register mn (TCRmn) delayed by one period of $f_{CLK}$ from rising edge of the count clock, because of synchronization with $f_{CLK}$. But, this is described as “counting at rising edge of the count clock”, as a matter of convenience.

**Figure 7-24. Timing of $f_{CLK}$ and count clock ($f_{TCLK}$) (When CCSmn = 0)**




**Remarks 1.** △ : Rising edge of the count clock

▲ : Synchronization, increment/decrement of counter

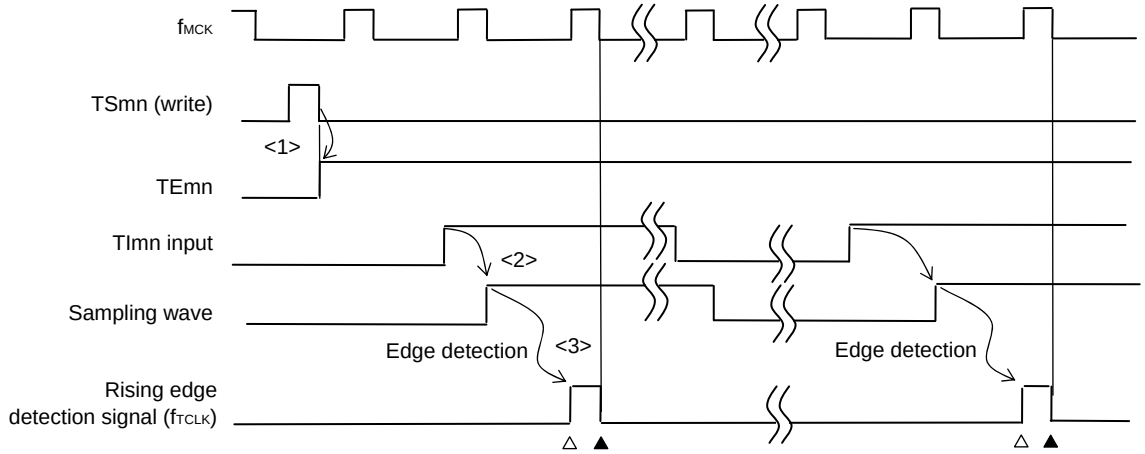
**2.** $f_{CLK}$: CPU/peripheral hardware clock

**(2) When valid edge of input signal via the TImn pin is selected (CCSmn = 1)**

The count clock ($f_{TCLK}$) becomes the signal that detects valid edge of input signal via the TImn pin and synchronizes next rising $f_{MCK}$. The count clock ($f_{TCLK}$) is delayed for 1 to 2 period of $f_{MCK}$ from the input signal via the TImn pin (when a noise filter is used, the delay becomes 3 to 4 clock).

Counting of timer count register mn (TCRmn) delayed by one period of $f_{CLK}$ from rising edge of the count clock, because of synchronization with $f_{CLK}$. 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 $f_{CLK}$ and count clock ($f_{TCLK}$) (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 $f_{MCK}$.

<3> The edge is detected by the rising of the sampled signal and the detection signal (count clock) is output.

**Remarks 1.** △ : Rising edge of the count clock
▲ : Synchronization, increment/decrement of counter

**2.** $f_{CLK}$: CPU/peripheral hardware clock
$f_{MCK}$: Operation clock of channel n

**3.** The waveform of the TImn pin input signal, which is used for input pulse interval measurement, input signal of high/low width measurement, the delay counter, and one-shot pulse output, is the same as that shown in above figure.

### 7.5.2 Start timing of counter

Timer count register mn (TCRmn) becomes enabled to operation by setting of TSmn bit of timer channel start register m (TSm).

Operations from count operation enabled state to timer count Register mn (TCRmn) count start is shown in Table 7-5.

**Table 7-5. Operations from Count Operation Enabled State to Timer Count Register mn (TCRmn) Count Start**

| Timer Operation Mode | Operation When TSmn = 1 Is Set |
|---|---|
| • Interval timer mode | No operation is carried out from start trigger detection (TSmn=1) until count clock generation.<br>The first count clock loads the value of the TDRmn register to the TCRmn register and the subsequent count clock performs count down operation (see **7.5.3 (1) Operation of interval timer mode**). |
| • Event counter mode | Writing 1 to the TSmn bit loads the value of the TDRmn register to the TCRmn register.<br>If detect edge of TImn input. The subsequent count clock performs count down operation (see **7.5.3 (2) Operation of event counter mode**). |
| • Capture mode | No operation is carried out from start trigger detection (TSmn = 1) until count clock generation.<br>The first count clock loads 0000H to the TCRmn register and the subsequent count clock performs count up operation (see **7.5.3 (3) Operation of 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).<br>No operation is carried out from start trigger detection until count clock generation.<br>The first count clock loads the value of the TDRmn register to the TCRmn register and the subsequent count clock performs count down operation (see **7.5.3 (4) Operation of 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).<br>No operation is carried out from start trigger detection until count clock generation.<br>The first count clock loads 0000H to the TCRmn register and the subsequent count clock performs count up operation (see **7.5.3 (5) Operation of capture & one-count mode (high-level width measurement)**). |

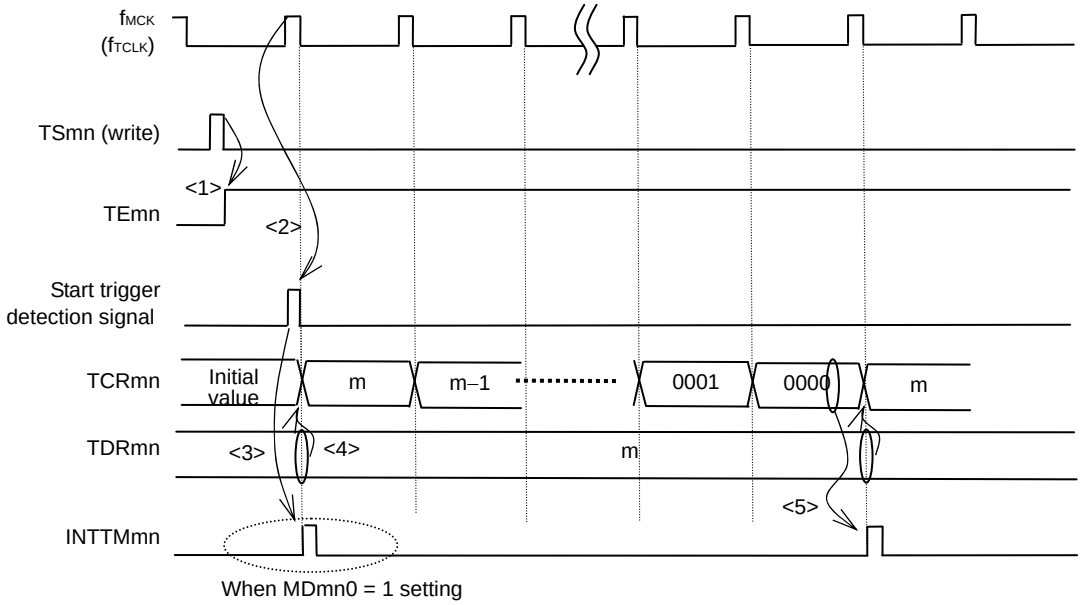
### 7.5.3 Operation of counter

Here, the counter operation in each mode is explained.

#### (1) Operation of interval timer mode

<1> Operation is enabled (TEmn = 1) by writing 1 to the TSmn bit. Timer count register mn (TCRmn) holds the initial value until count clock generation.
<2> A start trigger is generated at the first count 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 count 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. Operation Timing (In Interval Timer Mode)**



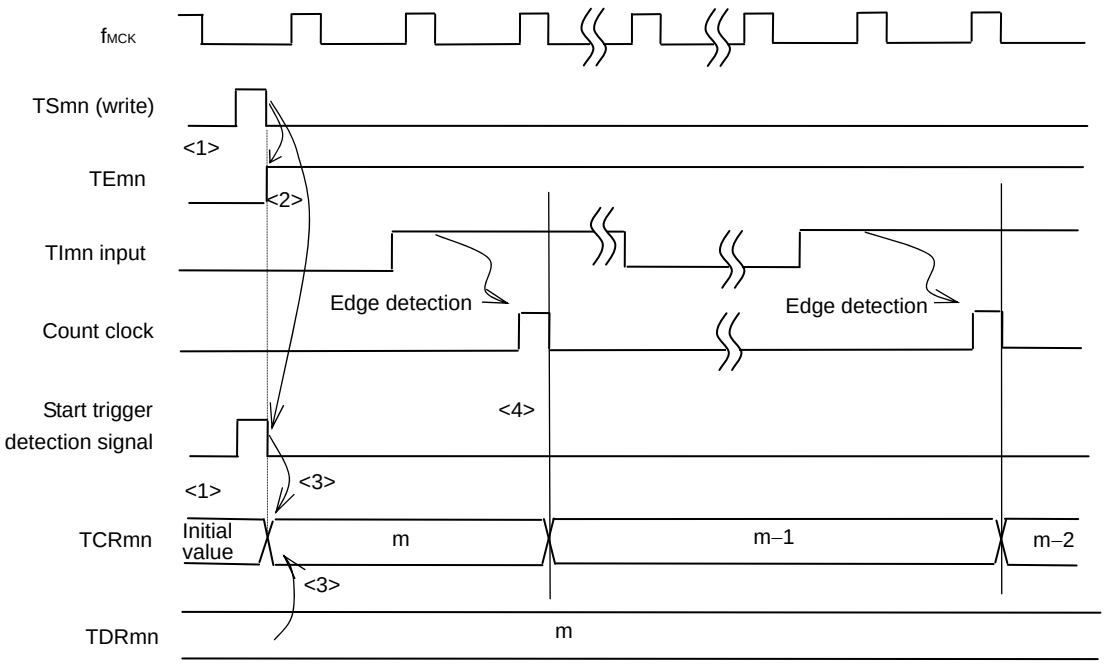

**Caution In the first cycle operation of count clock after writing the TSmn bit, an error at a maximum of one clock is generated since count start delays until count clock has been generated. When the information on count start timing is necessary, an interrupt can be generated at count start by setting MDmn0 = 1.**

**Remark** $f_{MCK}$, the start trigger detection signal, and INTTMmn become active between one clock in synchronization with $f_{CLK}$.

### (2) Operation of event counter mode

<1> Timer count 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 count clock of the valid edge of the TImn input .

**Figure 7-27. Operation Timing (In Event Counter Mode)**

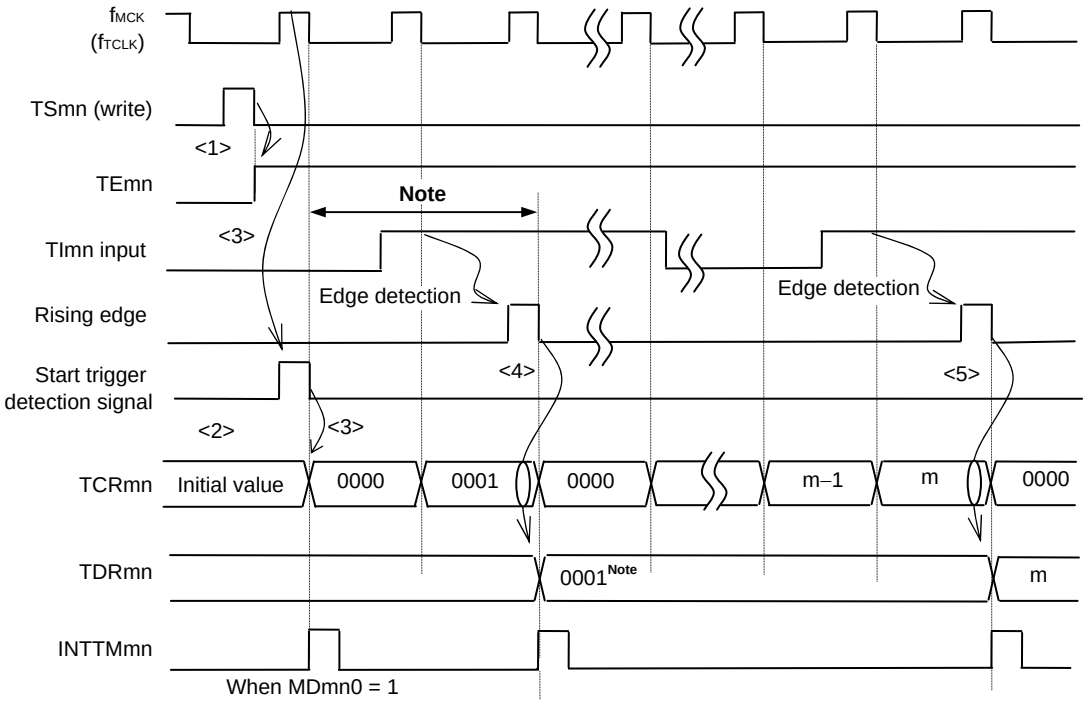



**Remark** The timing is shown in above figure indicates while the noise filter is not used. By making the noise filter on-state, the edge detection becomes 2 $f_{MCK}$ cycles (it sums up to 3 to 4 cycles) later than the normal cycle of TImn input. The error per one period occurs be the asynchronous between the period of the TImn input and that of the count clock ($f_{MCK}$).

### (3) Operation of capture mode (input pulse interval measurement)

<1> Operation is enabled (TEmn = 1) by writing 1 to the TSmn bit.
<2> Timer count register mn (TCRmn) holds the initial value until count clock generation.
<3> A start trigger is generated at the first count 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 capture value is nomeaning. 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. Operation Timing (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 count clock after writing the TSmn bit, an error at a maximum of one clock is generated since count start delays until count 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.**

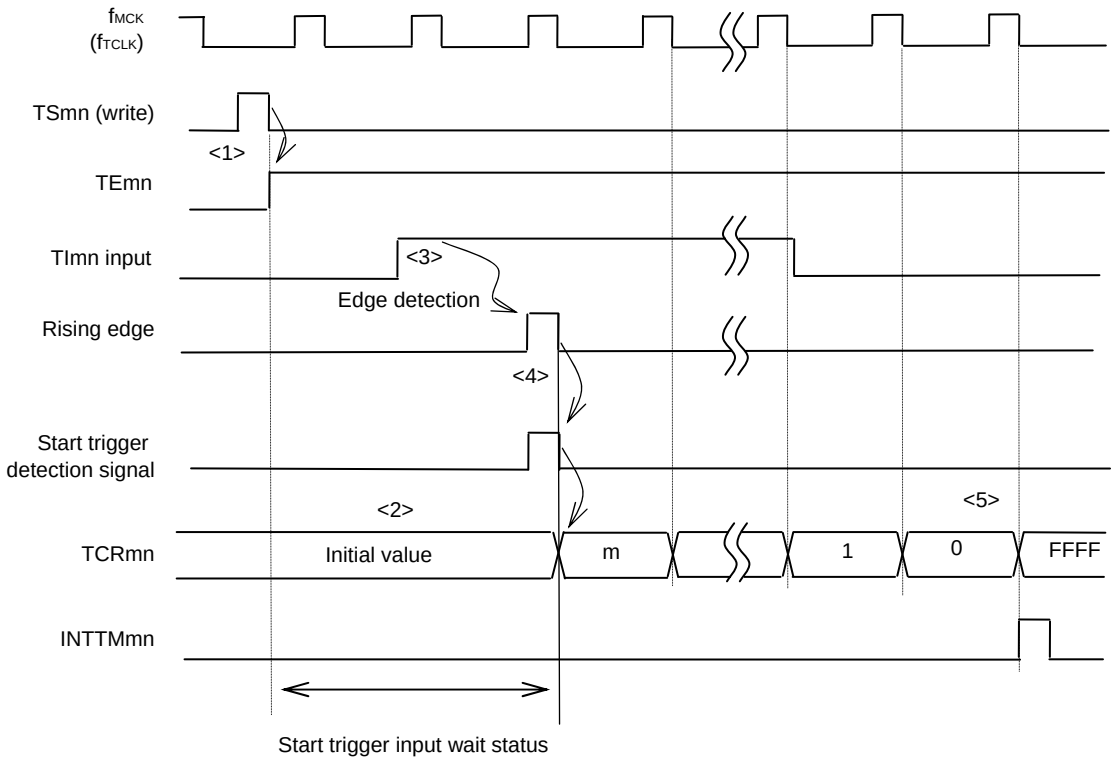
**Remark** The timing is shown in above figure indicates while the noise filter is not used. By making the noise filter on-state, the edge detection becomes 2 fMCK cycles (it sums up to 3 to 4 cycles) later than the normal cycle of TImn input. The error per one cycle occurs because the TImn input is not synchronous with the count clock (fMCK).

### (4) Operation of one-count mode

<1> Operation is enabled (TEmn = 1) by writing 1 to the TSmn bit.
<2> Timer count 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. Operation Timing (In One-count Mode)**

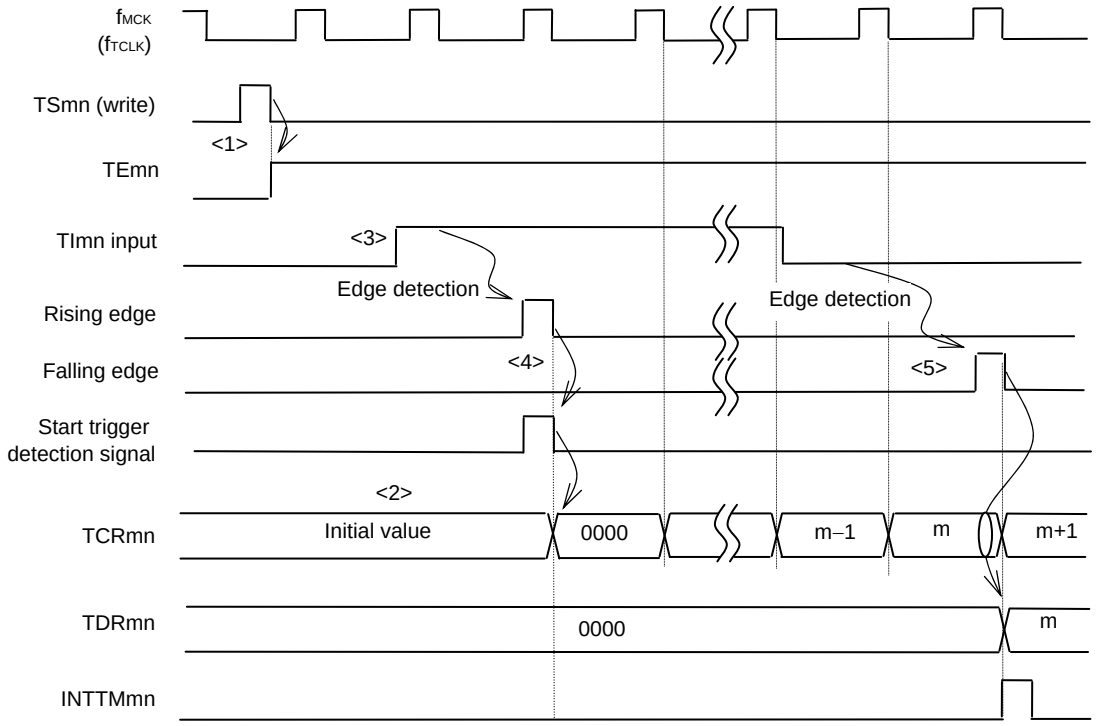



**Remark** The timing is shown in above figure indicates while the noise filter is not used. By making the noise filter on-state, the edge detection becomes 2 fMCK cycles (it sums up to 3 to 4 cycles) later than the normal cycle of TImn input. The error per one period occurs be the asynchronous between the period of the TImn input and that of the count clock (fMCK).

### (5) Operation of 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 count 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. Operation Timing (In Capture & One-count Mode: High-level Width Measurement)**

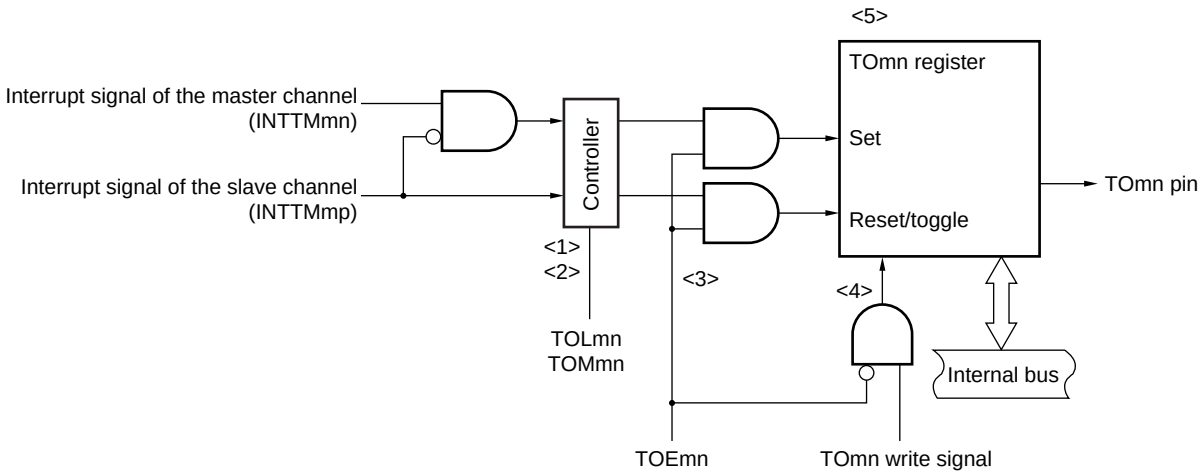



**Remark** The timing is shown in above figure indicates while the noise filter is not used. By making the noise filter on-state, the edge detection becomes 2 fMCK cycles (it sums up to 3 to 4 cycles) later than the normal cycle of TImn input. The error per one period occurs be the asynchronous between the period of the TImn input and that of the count 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 stopeed (TOEmn = 0) and to write a value to the TOm register.

<4> While timer output is disabeled (TOEmn = 0), writing to the TOmn bit to the target channel (TOmn write signal) becomes valid. When timer output is disabeled (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.

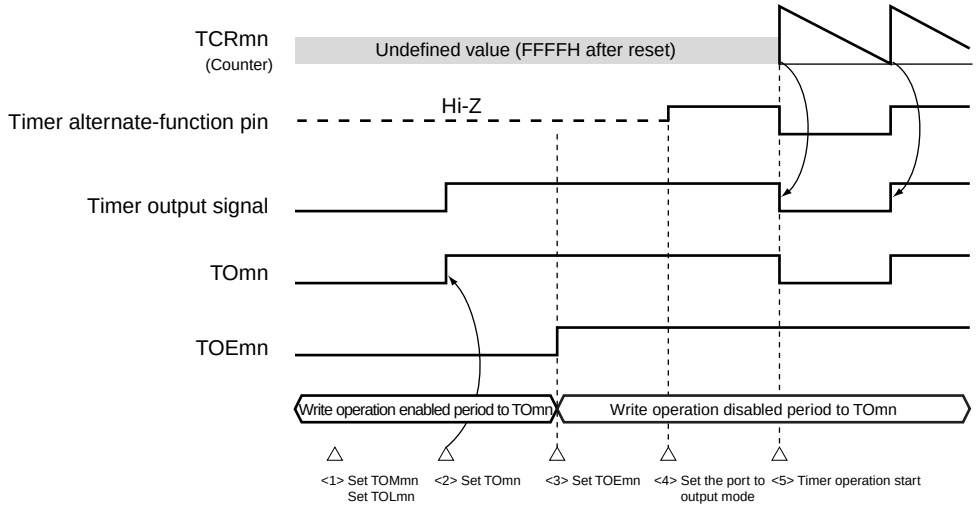
**Caution Since outputs are N-ch open-drain outputs, an external pull-up resistor is required to use P60, P61, and P62 as channel output.**

**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
$n < p \leq 7$

### 7.6.2 TOmn Pin Output Setting

The following figure shows the procedure and status transition of the TOmn output pin from initial setting to timer operation start.

**Figure 7-32. Status Transition from Timer Output Setting to Operation Start**




<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 status 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 I/O setting is set to output (see **7.3.15 Registers controlling port functions of pins to be used for timer I/O**).
<5> 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 count 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 by 7.8 and 7.9.

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)

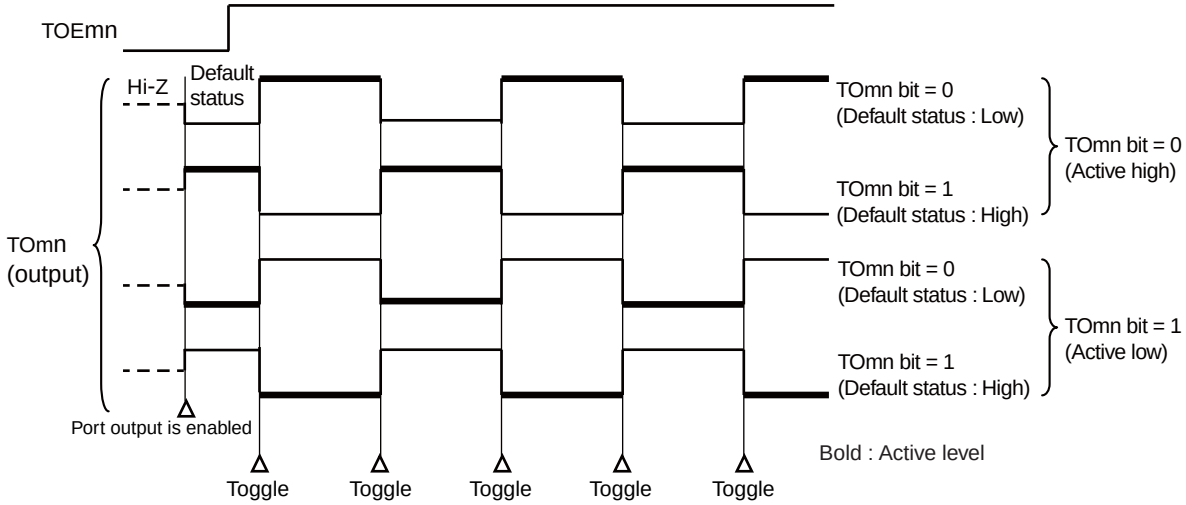
**(2) Default level of TOmn pin and output level after timer operation start**

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 reversed.

**Figure 7-33. TOmn Pin Output Status at Toggle Output (TOMmn = 0)**



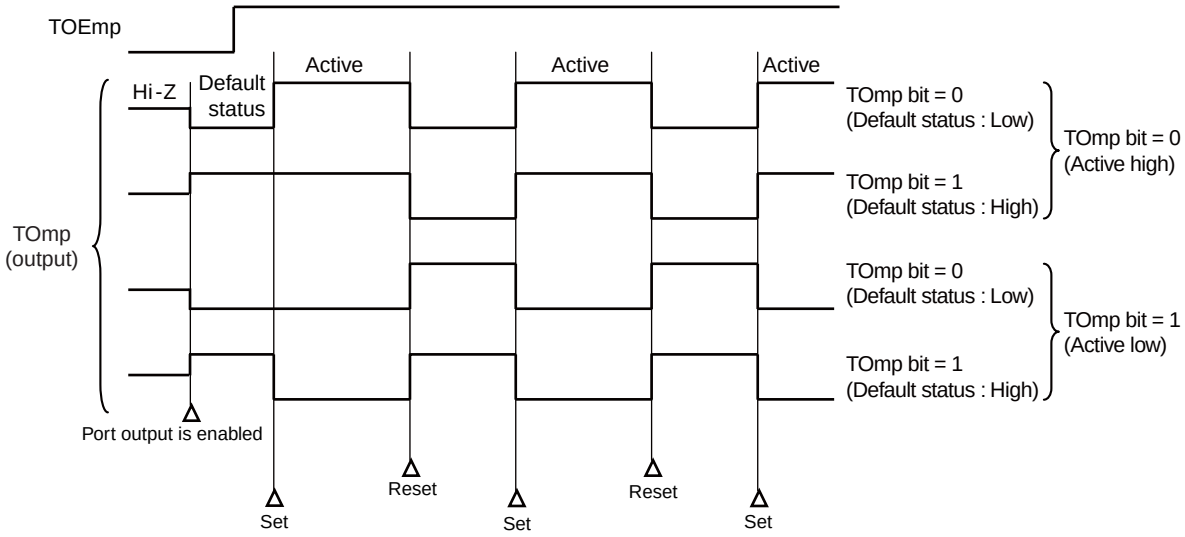

**Remarks 1.** Toggle: Reverse TOmn pin output status

**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 Status at PWM Output (TOMmp = 1)**




**Remarks 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.

**2.** m: Unit number (m = 0), p: Channel number (p = 1 to 7)

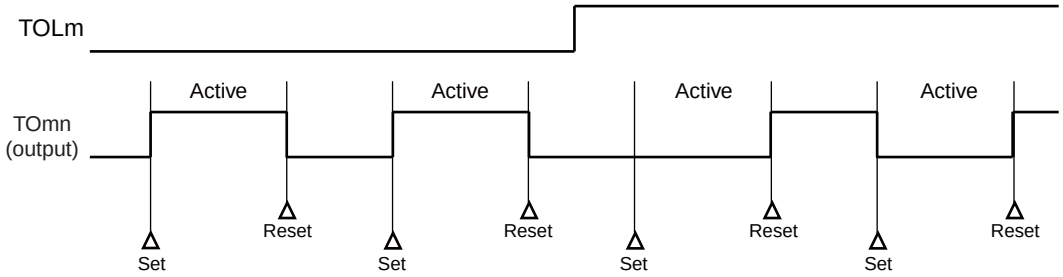
### (3) Operation of TOmn pin in slave channel output mode (TOMmn = 1)

#### (a) When timer output level register m (TOLm) setting has been 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 TOLm Register Has Been Changed Contents During Timer Operation**




**Remarks 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.

**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 count clock by the slave channel.

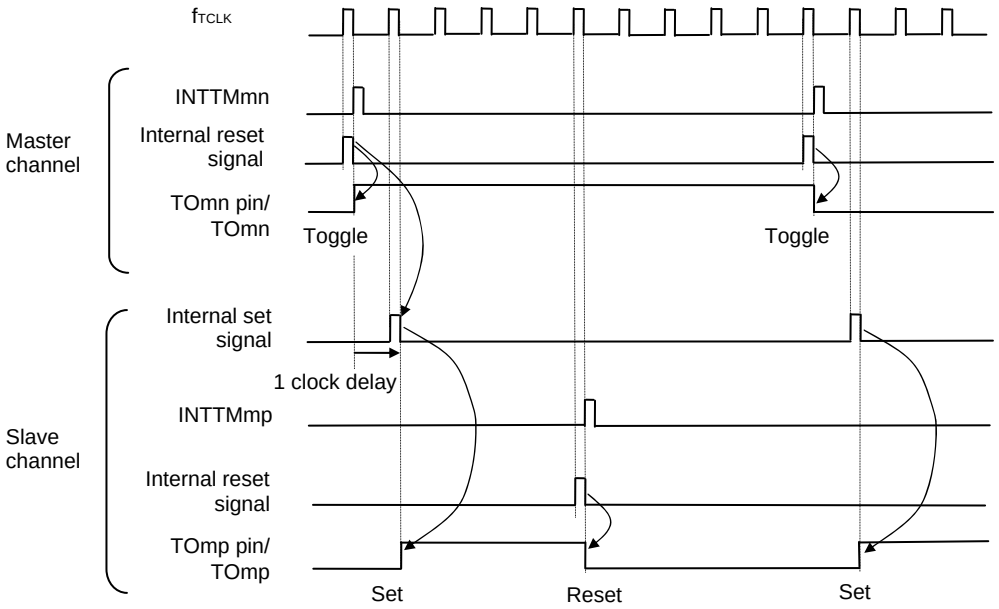
If the set condition and reset condition are generated at the same time, a higher priority is given to the latter.

Figure 7-36 shows the set/reset operating statuses where the master/slave channels are set as follows.

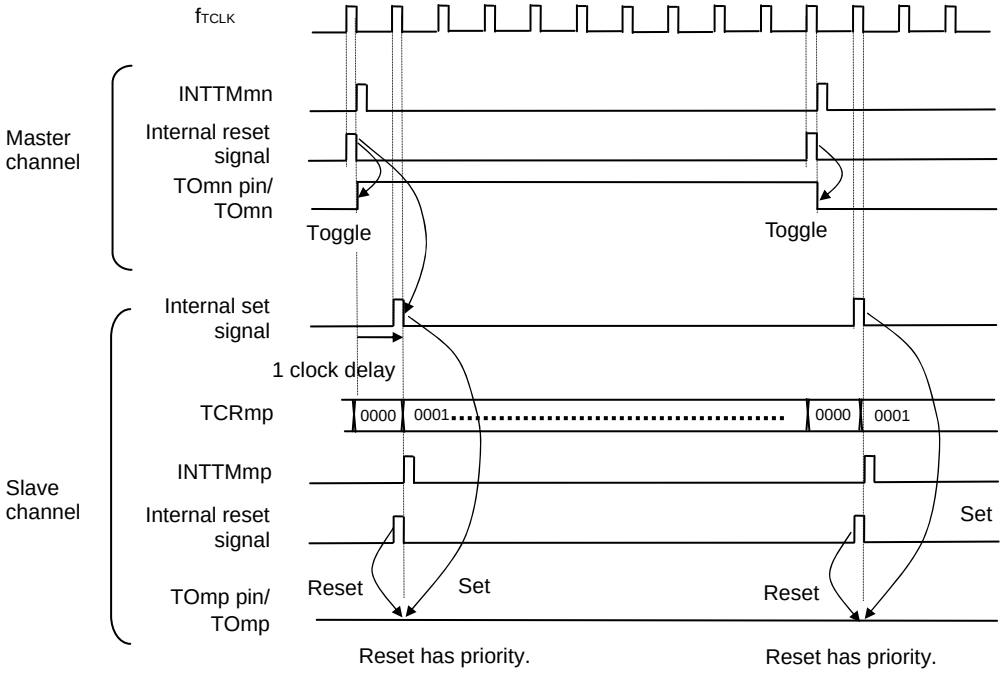
Master channel: TOEmn = 1, TOMmn = 0, TOLmn = 0
Slave channel: TOEmp = 1, TOMmp = 1, TOLmp = 0

**Figure 7-36. Set/Reset Timing Operating Statuses**

**(1) Basic operation timing**




**(2) Operation timing when 0 % duty**




**Remarks 1.** Internal reset signal: TOmn pin reset/toggle signal
Internal set signal: TOmn pin set signal

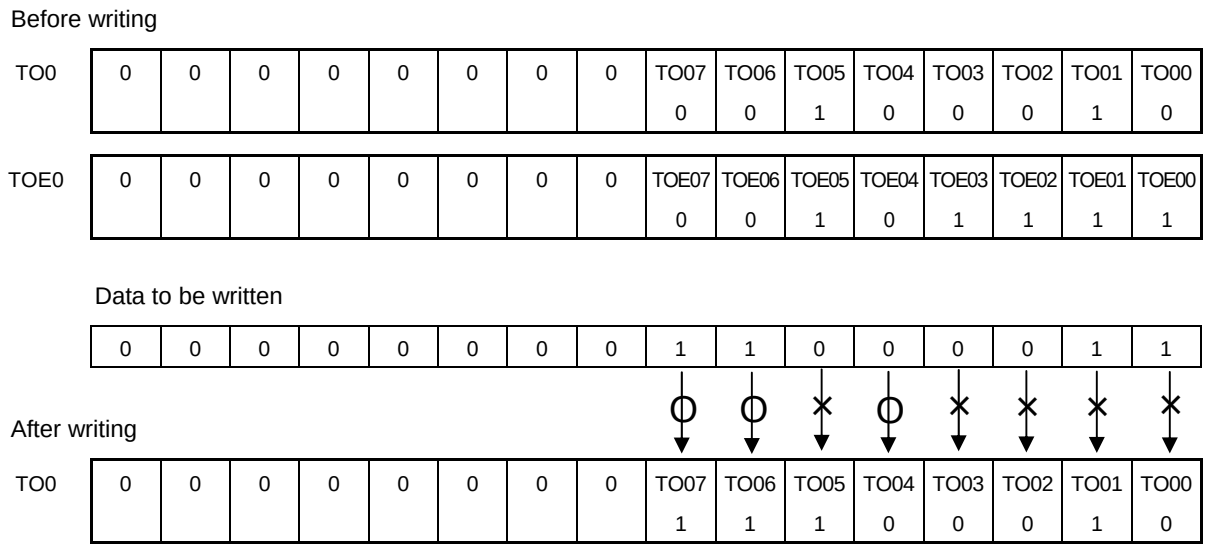
**2.** m: Unit number (m = 0)
n: Channel number
n = 0 to 7 (n = 0, 2, 4, 6 for master channel)
p: Slave channel number
n < p ≤ 7

### 7.6.4 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).

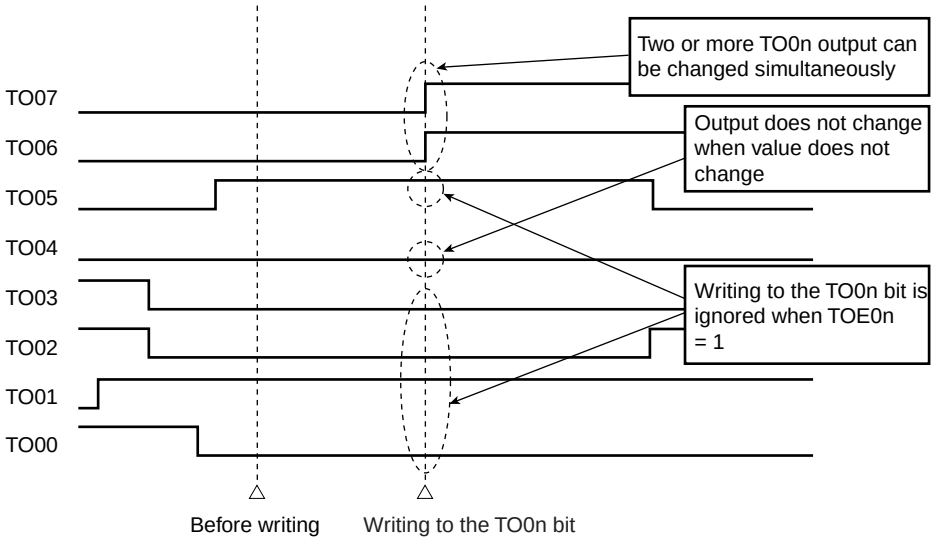
**Figure 7-37 Example of TO0n Bit Collective Manipulation**




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 Statuses by Collective Manipulation of TO0n Bit**




<R>

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

### 7.6.5 Timer Interrupt and TOmn Pin Output at Operation Start

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.

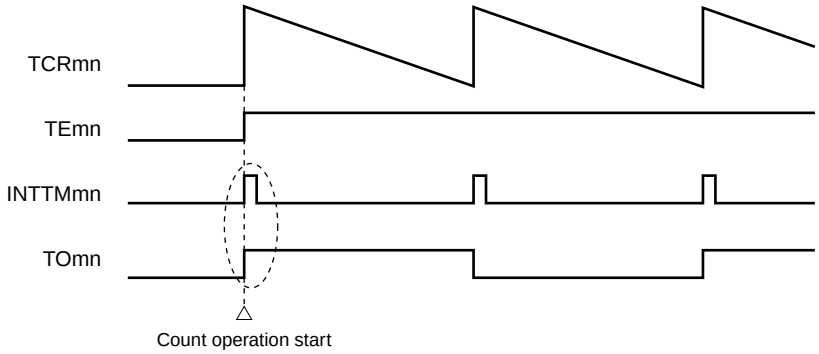
When MDmn0 is set to 1, the count operation start timing can be known by the timer interrupt (INTTMmn) generation.

In the other modes, neither timer interrupt at count operation start nor TOmn output is controlled.

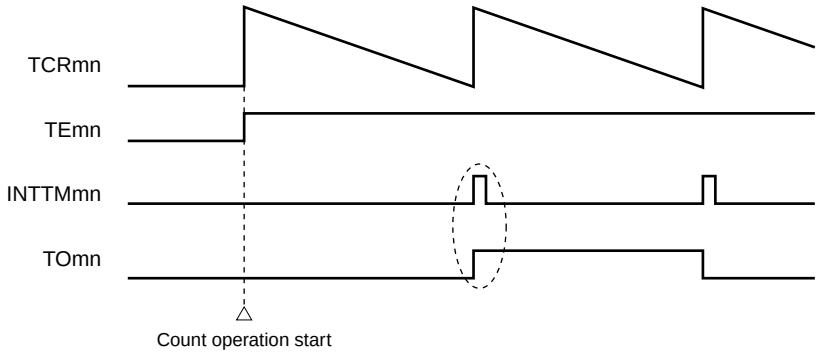
Figure 7-39 shows operation examples when the interval timer mode (TOEmn = 1, TOMmn = 0) is set.

**Figure 7-39. Operation Examples of Timer Interrupt at Count Operation Start and TOmn Output**

**(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.

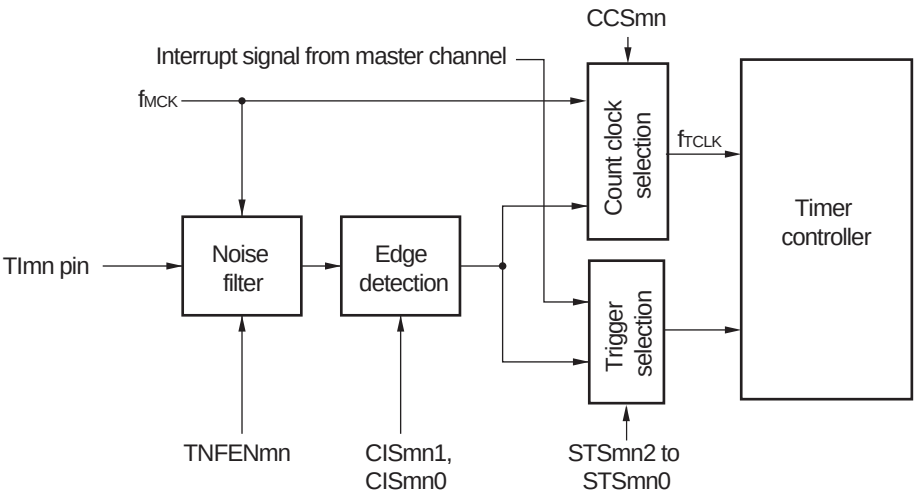
**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

## 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.

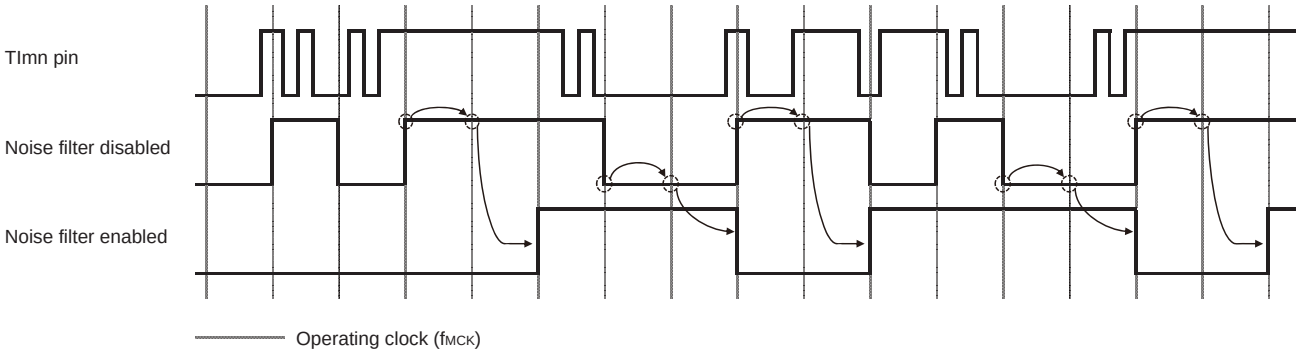
**Figure 7-40. Input Circuit Configuration**




### 7.7.2 Noise filter

When the noise filter is disabled, the input signal is only synchronized with the operating clock ($f_{MCK}$) for channel n. When the noise filter is enabled, after synchronization with the operating clock ($f_{MCK}$) 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**




**Caution The TImn pin input waveform is shown to explain the noise filter ON/OFF operation. For actual operation, refer to the high-level width/low-level width in 37.4 AC Characteristics.**

### 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 bits 12 (CCSmn), 9 (STSmn1), and 8 (STSmn0) in the timer mode register mn (TMRmn) are 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 bits 12 (CCSmn), 9 (STSmn1), and 8 (STSmn0) 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 interval timer/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 count clock × (Set value of TDRmn + 1)

**(2) Operation as 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 count clock × (Set value of TDRmn + 1) × 2
- Frequency of square wave output from TOmn = Frequency of count clock/{(Set value of TDRmn + 1) × 2}

Timer count 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 count clock after the channel start trigger bit (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 TOmn is not toggled. If the MDmn0 bit of the TMRmn register is 1, INTTMmn is output and TOmn is toggled.

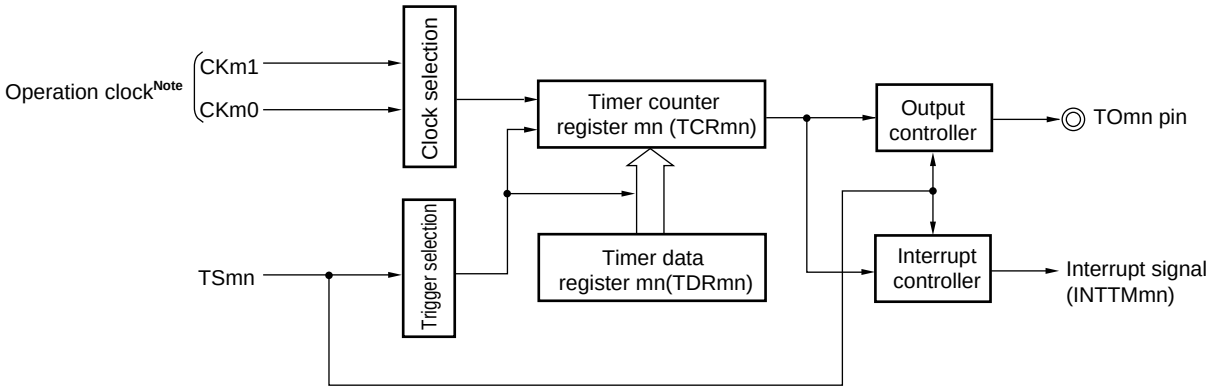
After that, the TCRmn register count down in synchronization with the count clock.

When TCRmn = 0000H, INTTMmn is output and TOmn is toggled at the next count 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.

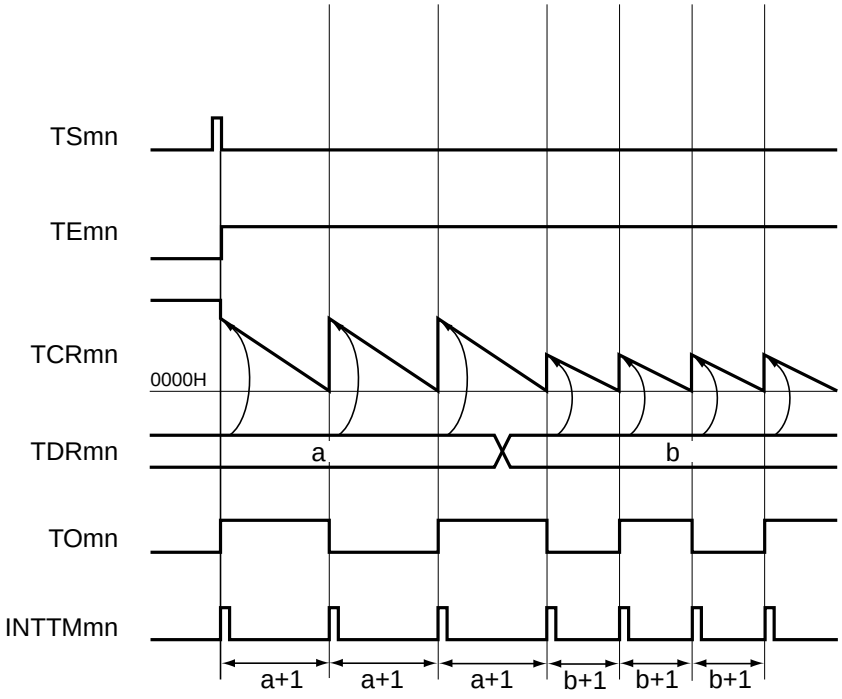
**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

**Figure 7-42. Block Diagram of Operation as Interval Timer/Square Wave Output**




**Note** When channels 1 and 3, the clock can be selected from CKm0, CKm1, CKm2 and CKm3.

**Figure 7-43. Example of Basic Timing of Operation as Interval Timer/Square Wave Output (MDmn0 = 1)**

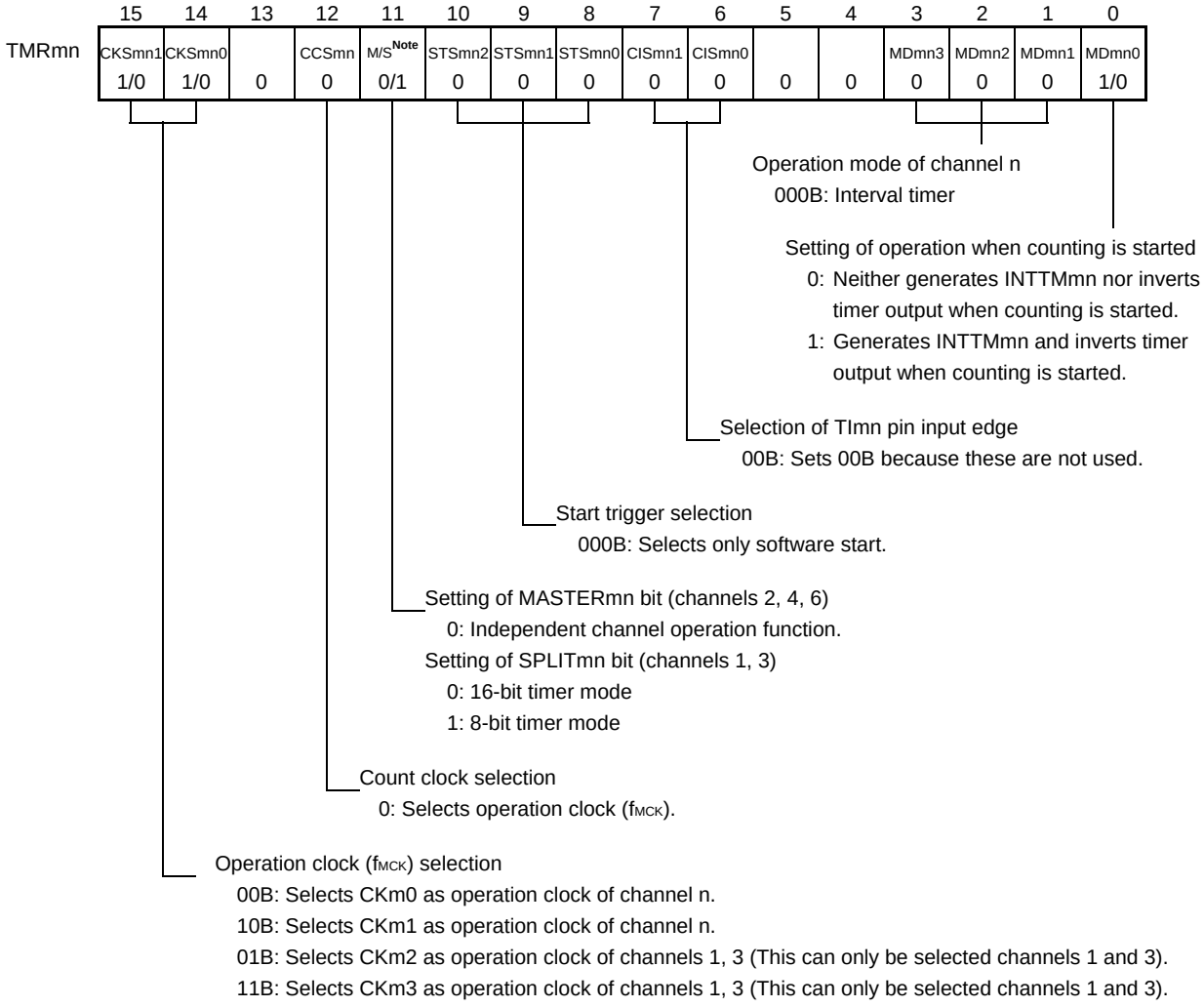



**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

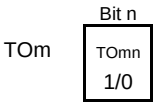
**2.** TSmn: Bit n of timer channel start register m (TSm)
TEmn: Bit n of timer channel enable status register m (TEm)
TCRmn: Timer count register mn (TCRmn)
TDRmn: Timer data register mn (TDRmn)
TOmn: TOmn pin output signal

**Figure 7-44. Example of Set Contents of Registers During Operation as Interval Timer/Square Wave Output (1/2)**

**(a) Timer mode register mn (TMRmn)**

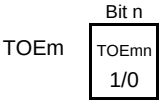



**(b) Timer output register m (TOm)**




0: Outputs 0 from TOmn.
1: Outputs 1 from TOmn.

**(c) Timer output enable register m (TOEm)**




0: Stops the TOmn output operation by counting operation.
1: Enables the TOmn output operation by counting operation.

**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 7-44. Example of Set Contents of Registers During Operation as Interval Timer/Square Wave Output (2/2)**

**(d) Timer output level register m (TOLm)**

| | Bit n | |
|---|---|---|
| TOLm | TOLmn<br>0 | 0: Cleared to 0 when TOMmn = 0 (master channel output mode) |

**(e) Timer output mode register m (TOMm)**

| | Bit n | |
|---|---|---|
| TOMm | TOMmn<br>0 | 0: Sets master channel output mode. |

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

**Figure 7-45. Operation Procedure of Interval Timer/Square Wave Output Function (1/2)**

| | Software Operation | Hardware Status |
|---|---|---|
| TAU default setting | | Power-off status<br>(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 status. Each channel stops operating.<br>(Clock supply is started and writing to each register is enabled.) |
| | Sets timer clock select register m (TPSm).<br>Determines clock frequencies of CKm0 to CKm3. | |
| Channel default setting | Sets timer mode register mn (TMRmn) (determines operation mode of channel).<br>Sets interval (period) value to timer data register mn (TDRmn). | Channel stops operating.<br>(Clock is supplied and some power is consumed.) |
| | To use the TOmn output<br>Clears the TOMmn bit of timer output mode register m (TOMm) to 0 (master channel output mode).<br>Clears the TOLmn bit to 0.<br>Sets the TOmn bit and determines default level of the TOmn output. → | The TOmn pin goes into Hi-Z output state.<br>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.).<br>Sets the TSmn (TSHm1, TSHm3) bit to 1. →<br>The TSmn (TSHm1, TSHm3) bit automatically returns to 0 because it is a trigger bit. | TEmn (TEHm1, TEHm3) = 1, and count operation starts.<br>Value of the TDRmn register is loaded to timer count register mn (TCRmn). INTTMmn is generated and TOmn performs toggle operation if the MDmn0 bit of the TMRmn register is 1. |
| During operation | Set value of the TDRmn register can be changed.<br>The TCRmn register can always be read.<br>The TSRmn register is not used.<br>Set values of the TOm and TOEm registers can be changed.<br>Set values of the TMRmn register, TOMmn, and TOLmn bits cannot 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.<br>By detecting TCRmn = 0000H, INTTMmn is generated and TOmn performs toggle operation.<br>After that, the above operation is repeated. |
| Operation stop | The TTmn (TTHm1, TTHm3) bit is set to 1. →<br>The TTmn (TTHm1, TTHm3) bit automatically returns to 0 because it is a trigger bit. | TEmn (TEHm1, TEHm3), and count operation stops.<br>The TCRmn register holds count value and stops.<br>The TOmn output is not initialized but holds current status. |
| | The TOEmn bit is cleared to 0 and value is set to the TOmn bit. → | The TOmn pin outputs the TOmn bit set level. |

Operation is resumed.

(**Remark** is listed on the next page.)

**Figure 7-45. Operation Procedure of Interval Timer/Square Wave Output Function (2/2)**

| | Software Operation | Hardware Status |
|---|---|---|
| TAU stop | To hold the TOmn pin output level<br>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<br>Setting not required. | |
| | The TAUmEN bit of the PER0 register is cleared to 0. → | Power-off status<br>All circuits are initialized and SFR of each channel is also initialized.<br>(The TOmn bit is cleared to 0 and the TOmn pin is set to port mode.) |

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

### 7.8.2 Operation as 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 count 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 channel start trigger bit (TSmn, TSHm1, TSHm3) of timer channel start register m (TSm) to 1.

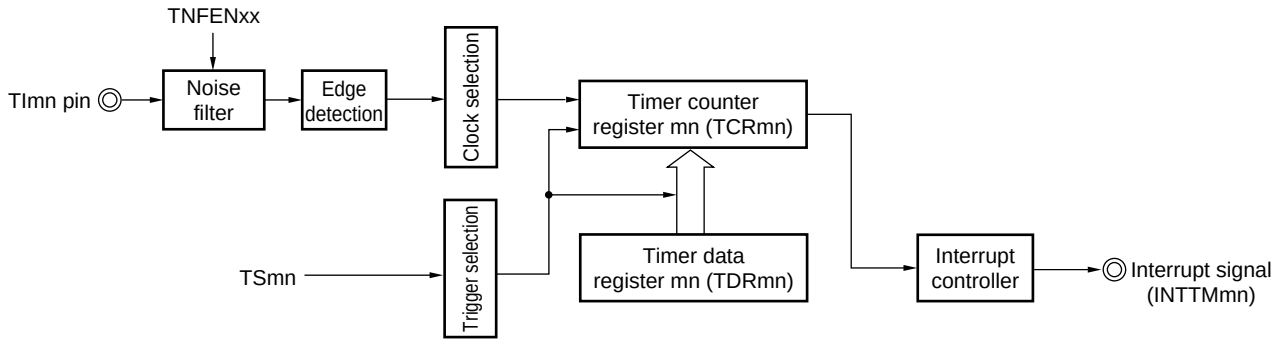
The TCRmn register counts down each time the valid input edge of the TImn pin has been detected. When TCRmn = 0000H, the TCRmn register loads the value of the TDRmn register again, and outputs INTTMmn.

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.

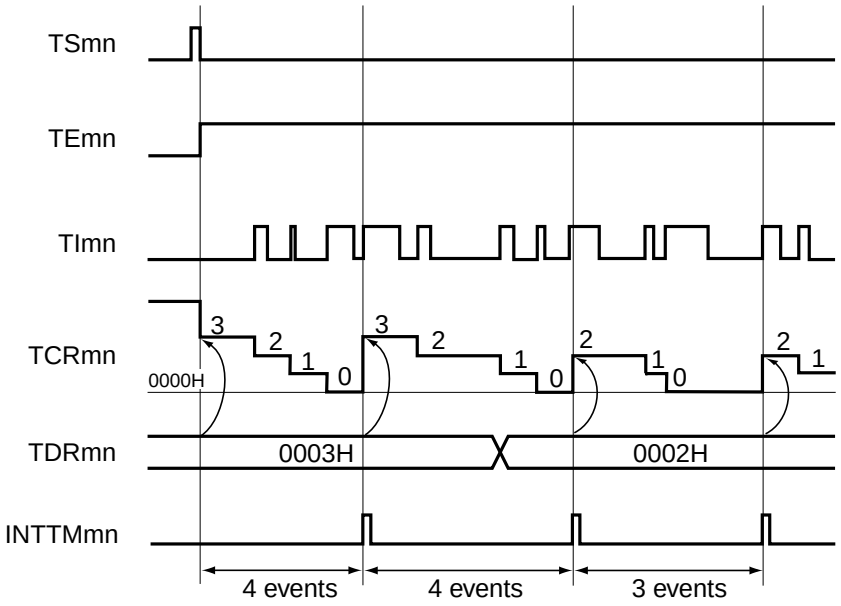
The TDRmn register can be rewritten at any time. The new value of the TDRmn register becomes valid during the next count period.

**Figure 7-46. Block Diagram of Operation as External Event Counter**




**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

**Figure 7-47. Example of Basic Timing of Operation as External Event Counter**

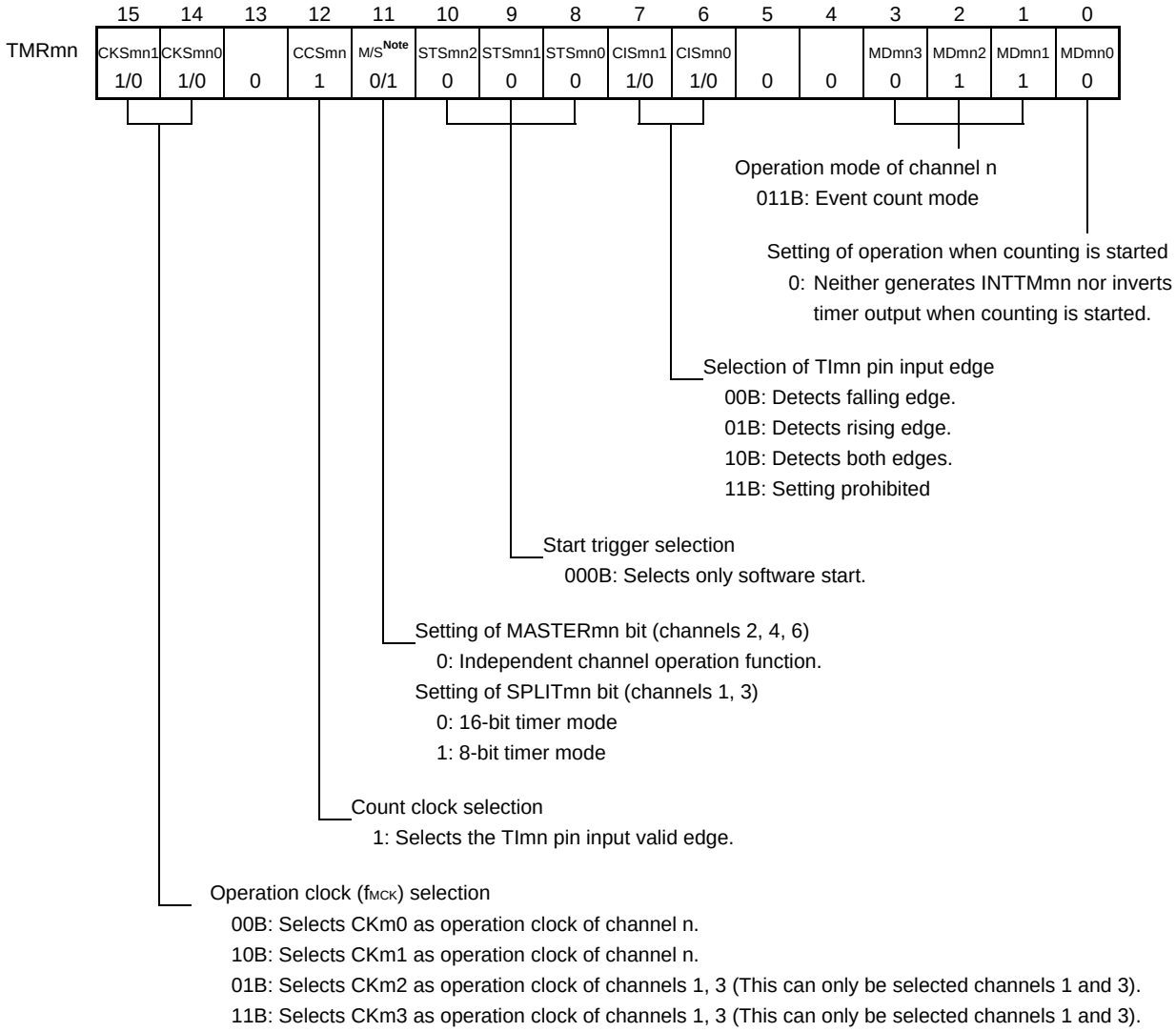



**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

**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 count register mn (TCRmn)
TDRmn: Timer data register mn (TDRmn)

**Figure 7-48. Example of Set Contents of Registers in External Event Counter Mode (1/2)**

**(a) Timer mode register mn (TMRmn)**




**(b) Timer output register m (TOm)**

| | Bit n | |
|---|---|---|
| TOm | TOmn | 0: Outputs 0 from TOmn. |
| | 0 | |

**(c) Timer output enable register m (TOEm)**

| | Bit n | |
|---|---|---|
| TOEm | TOEmn | 0: Stops the TOmn output operation by counting operation. |
| | 0 | |

**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 7-48. Example of Set Contents of Registers in External Event Counter Mode (2/2)**

**(d) Timer output level register m (TOLm)**

| | Bit n | |
|---|---|---|
| TOLm | TOLmn<br>0 | 0: Cleared to 0 when TOMmn = 0 (master channel output mode). |

**(e) Timer output mode register m (TOMm)**

| | Bit n | |
|---|---|---|
| TOMm | TOMmn<br>0 | 0: Sets master channel output mode. |

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

**Figure 7-49. Operation Procedure When External Event Counter Function Is Used**

| | Software Operation | Hardware Status |
|---|---|---|
| TAU default setting | | Power-off status<br>(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 status. Each channel stops operating.<br>(Clock supply is started and writing to each register is enabled.) |
| | Sets timer clock select register m (TPSm).<br>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).<br>Sets timer mode register mn (TMRmn) (determines operation mode of channel).<br>Sets number of counts to timer data register mn (TDRmn).<br>Clears the TOEmn bit of timer output enable register m (TOEm) to 0. | Channel stops operating.<br>(Clock is supplied and some power is consumed.) |
| Operation start | Sets the TSmn bit to 1. →<br>The TSmn bit automatically returns to 0 because it is a trigger bit. | TEmn = 1, and count operation starts.<br>Value of the TDRmn register is loaded to timer count register mn (TCRmn) and detection of the TImn pin input edge is awaited. |
| During operation | Set value of the TDRmn register can be changed.<br>The TCRmn register can always be read.<br>The TSRmn register is not used.<br>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.<br>After that, the above operation is repeated. |
| Operation stop | The TTmn bit is set to 1. →<br>The TTmn bit automatically returns to 0 because it is a trigger bit. | TEmn = 0, and count operation stops.<br>The TCRmn register holds count value and stops. |
| TAU stop | The TAUmEN bit of the PER0 register is cleared to 0. → | Power-off status<br>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 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 count clock × ((10000H × TSRmn: OVF) + (Capture 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 of up to one operating clock cycle occurs.**

Timer count register mn (TCRmn) operates as an up counter in the capture mode.

When the channel 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 count clock.

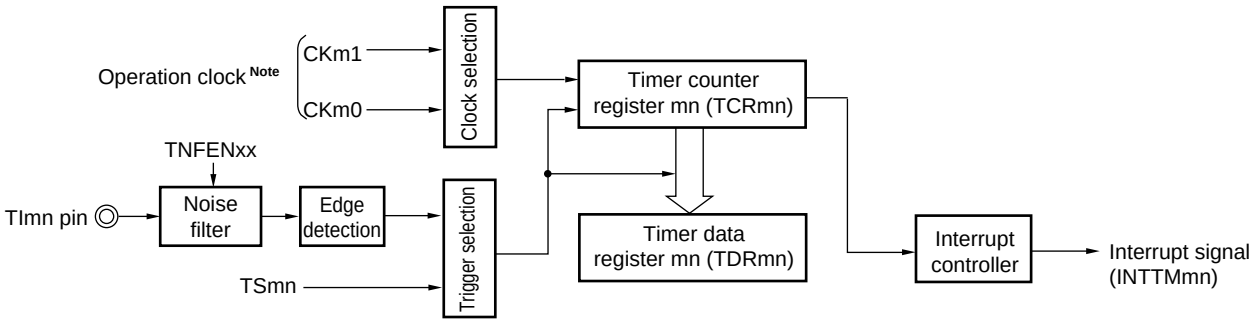
When the TImn pin input valid edge is detected, the count value of the TCRmn register is transferred (captured) to timer data register mn (TDRmn) and, at the same time, the TCRmn register is cleared to 0000H, and the INTTMmn is output. If the counter overflows at this time, the OVF bit of timer status 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 status 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.

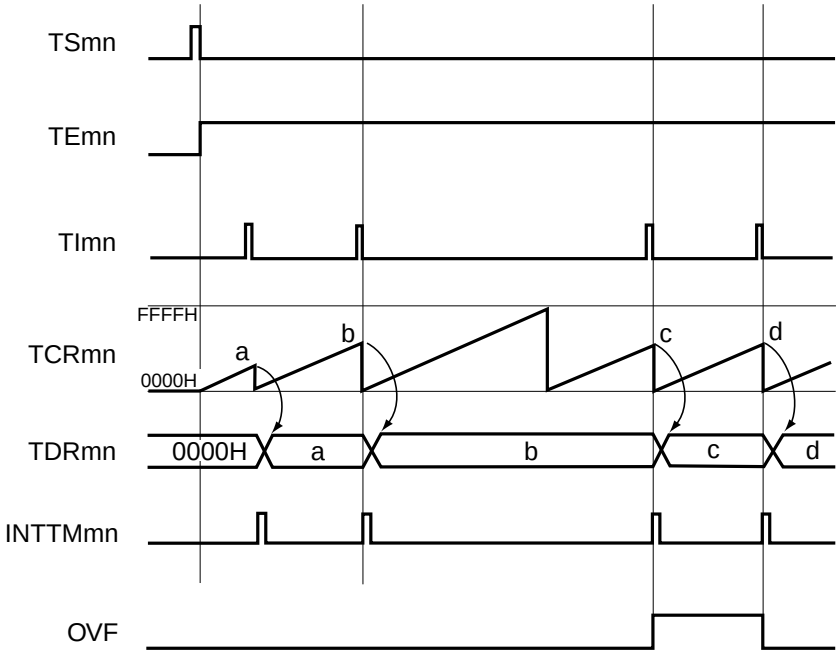
**Figure 7-50. Block Diagram of Operation as Input Pulse Interval Measurement**




**Note** When channels 1 and 3, the clock can be selected from CKm0, CKm1, CKm2 and CKm3.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

**Figure 7-51. Example of Basic Timing of Operation as Input Pulse Interval Measurement (MDmn0 = 0)**

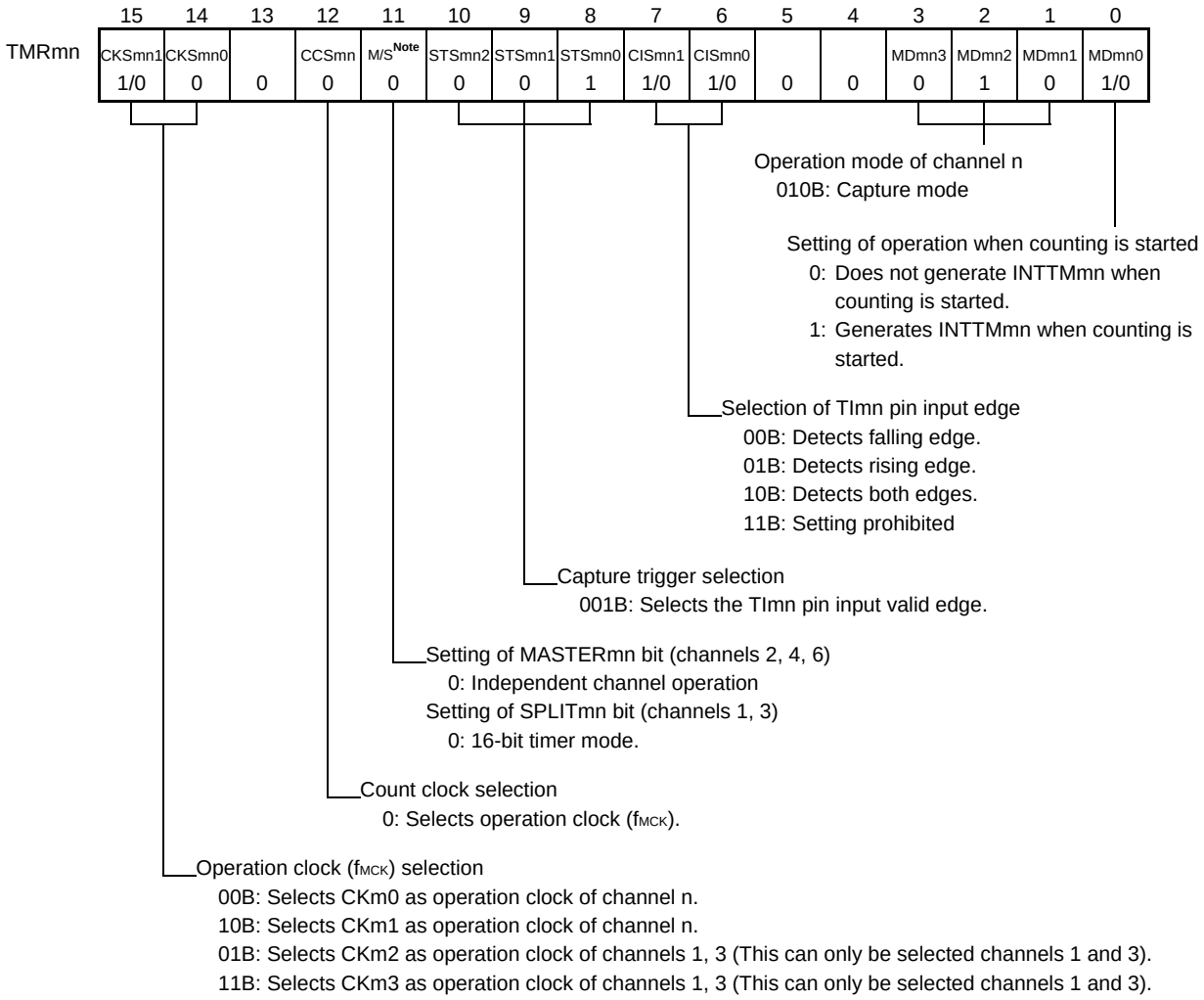



**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

**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 count register mn (TCRmn)
TDRmn: Timer data register mn (TDRmn)
OVF: Bit 0 of timer status register mn (TSRmn)

**Figure 7-52. Example of Set Contents of Registers to Measure Input Pulse Interval**

**(a) Timer mode register mn (TMRmn)**




**(b) Timer output register m (TOm)**

| | Bit n | |
|---|---|---|
| TOm | TOmn | 0: Outputs 0 from TOmn. |
| | 0 | |

**(c) Timer output enable register m (TOEm)**

| | Bit n | |
|---|---|---|
| TOEm | TOEmn | 0: Stops TOmn output operation by counting operation. |
| | 0 | |

**(d) Timer output level register m (TOLm)**

| | Bit n | |
|---|---|---|
| TOLm | TOLmn | 0: Cleared to 0 when TOMmn = 0 (master channel output mode). |
| | 0 | |

**(e) Timer output mode register m (TOMm)**

| | Bit n | |
|---|---|---|
| TOMm | TOMmn | 0: Sets master channel output mode. |
| | 0 | |

**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 7-53. Operation Procedure When Input Pulse Interval Measurement Function Is Used**

| | Software Operation | Hardware Status |
|---|---|---|
| TAU default setting | | Power-off status<br>(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 status. Each channel stops operating.<br>(Clock supply is started and writing to each register is enabled.) |
| | Sets timer clock select register m (TPSm).<br>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).<br>Sets timer mode register mn (TMRmn) (determines operation mode of channel). | Channel stops operating.<br>(Clock is supplied and some power is consumed.) |
| Operation start | Sets TSmn bit to 1. →<br>The TSmn bit automatically returns to 0 because it is a trigger bit. | TEmn = 1, and count operation starts.<br>Timer count register mn (TCRmn) is cleared to 0000H.<br>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.<br>The TDRmn register can always be read.<br>The TCRmn register can always be read.<br>The TSRmn register can always be read.<br>Set values of the TOMmn, TOLmn, TOmn, and TOEmn bits cannot be changed. | Counter (TCRmn) counts up from 0000H. When the TImn pin input valid edge 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.<br>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.<br>After that, the above operation is repeated. |
| Operation stop | The TTmn bit is set to 1. →<br>The TTmn bit automatically returns to 0 because it is a trigger bit. | TEmn = 0, and count operation stops.<br>The TCRmn register holds count value and stops.<br>The OVF bit of the TSRmn register is also held. |
| TAU stop | The TAUmEN bit of the PER0 register is cleared to 0. → | Power-off status<br>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)

### 7.8.4 Operation as input signal high-/low-level width measurement

**Caution When using a channel to implement the LIN-bus, set bit 1 (ISC1) of the input switch control register (ISC) to 1. In the following descriptions, read TImn as RxD0.**

By starting counting at one edge of the TImn pin input and capturing the number of counts at another edge, the signal width (high-level width/low-level width) of TImn can be measured. The signal width of TImn can be calculated by the following expression.

| Signal width of TImn input = Period of count clock × ((10000H × TSRmn: OVF) + (Capture 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 operation clock occurs.**

Timer count register mn (TCRmn) operates as an up counter in the capture & one-count mode.

When the channel 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 status 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 count 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 status 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 status 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.

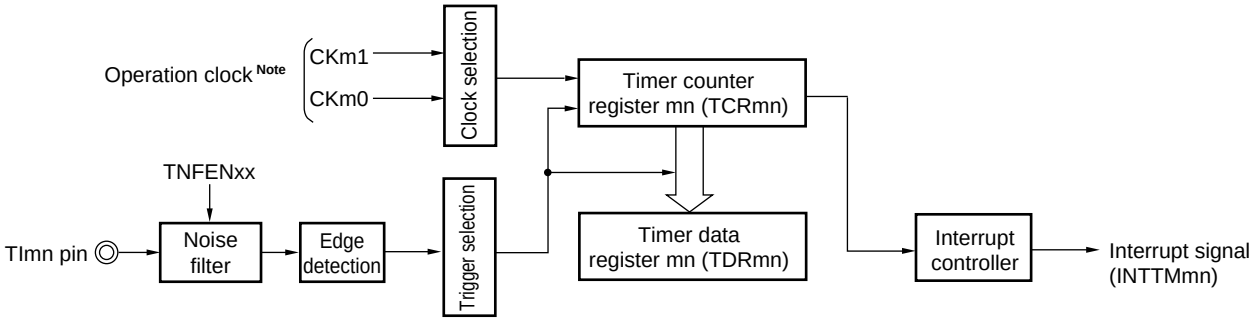
Whether the high-level width or low-level width of the TImn pin is to be measured can be selected by using the CISmn1 and CISmn0 bits of the TMRmn register.

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.

CISmn1, CISmn0 of TMRmn register = 10B: Low-level width is measured.

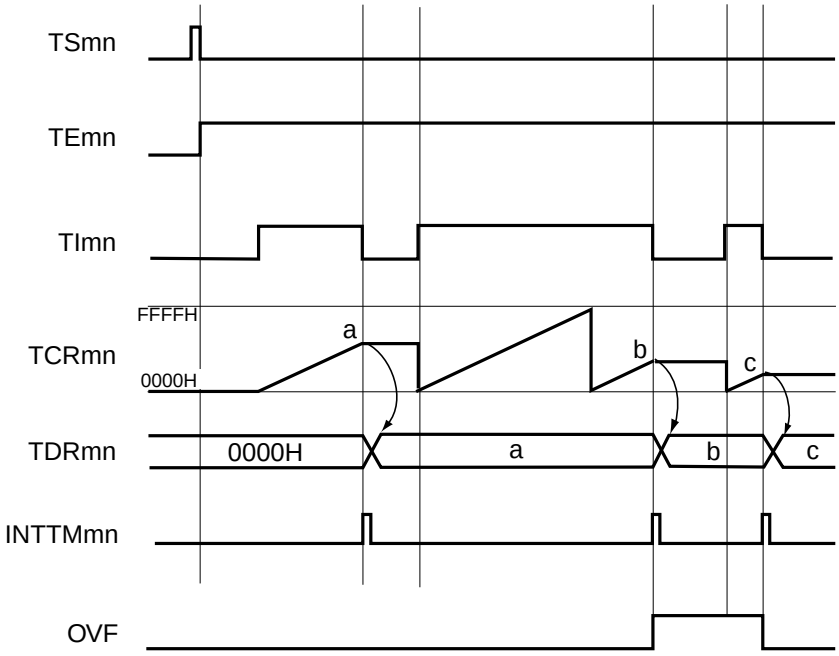
CISmn1, CISmn0 of TMRmn register = 11B: High-level width is measured.

**Figure 7-54. Block Diagram of Operation as Input Signal High-/Low-Level Width Measurement**




**Note** For channels 1 and 3, the clock can be selected from CKm0, CKm1, CKm2 and CKm3.

**Figure 7-55. Example of Basic Timing of Operation as Input Signal High-/Low-Level Width Measurement**




**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

**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 count register mn (TCRmn)
TDRmn: Timer data register mn (TDRmn)
OVF: Bit 0 of timer status register mn (TSRmn)

**Figure 7-56. Example of Set Contents of Registers to Measure Input Signal High-/Low-Level Width**

**(a) Timer mode register mn (TMRmn)**

TMRmn

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CKSmn1 | CKSmn0 | | CCSmn | M/S Note | STSmn2 | STSmn1 | STSmn0 | CISmn1 | CISmn0 | | | MDmn3 | MDmn2 | MDmn1 | MDmn0 |
| 1/0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1/0 | 0 | 0 | 1 | 1 | 0 | 0 |

Operation mode of channel n
110B: Capture & one-count

Setting of operation when counting is started
0: Does not generate INTTMmn when counting is started.

Selection of TImn pin input edge
10B: Both edges (to measure low-level width)
11B: Both edges (to measure high-level width)

Start trigger selection
010B: Selects the TImn pin input valid edge.

Setting of MASTERmn bit (channels 2, 4, 6)
0: Independent channel operation function.
Setting of SPLITmn bit (channels 1, 3)
0: 16-bit timer mode.

Count clock selection
0: Selects operation clock ($f_{MCK}$).

Operation clock ($f_{MCK}$) selection
00B: Selects CKm0 as operation clock of channel n.
10B: Selects CKm1 as operation clock of channel n.
01B: Selects CKm2 as operation clock of channels 1, 3 (This can only be selected channels 1 and 3).
11B: Selects CKm3 as operation clock of channels 1, 3 (This can only be selected channels 1 and 3).

**(b) Timer output register m (TOm)**

TOm

| Bit n |
|---|
| TOmn |
| 0 |

0: Outputs 0 from TOmn.

**(c) Timer output enable register m (TOEm)**

TOEm

| Bit n |
|---|
| TOEmn |
| 0 |

0: Stops the TOmn output operation by counting operation.

**(d) Timer output level register m (TOLm)**

TOLm

| Bit n |
|---|
| TOLmn |
| 0 |

0: Cleared to 0 when TOMmn = 0 (master channel output mode).

**(e) Timer output mode register m (TOMm)**

TOMm

| Bit n |
|---|
| 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 7-57. Operation Procedure When Input Signal High-/Low-Level Width Measurement Function Is Used**

| | Software Operation | Hardware Status |
|---|---|---|
| TAU default setting | | Power-off status<br>(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 status. Each channel stops operating.<br>(Clock supply is started and writing to each register is enabled.) |
| | Sets timer clock select register m (TPSm).<br>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).<br>Sets timer mode register mn (TMRmn) (determines operation mode of channel).<br>Clears the TOEmn bit to 0 and stops operation of TOmn. | Channel stops operating.<br>(Clock is supplied and some power is consumed.) |
| Operation start | Sets the TSmn bit to 1. →<br>The TSmn bit automatically returns to 0 because it is a trigger bit. | TEmn = 1, and the TImn pin start edge detection wait status is set. |
| | Detects the TImn pin input count start valid edge. → | Clears timer count register mn (TCRmn) to 0000H and starts counting up. |
| During operation | Set value of the TDRmn register can be changed.<br>The TCRmn register can always be read.<br>The TSRmn register is not used.<br>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.<br>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. →<br>The TTmn bit automatically returns to 0 because it is a trigger bit. | TEmn = 0, and count operation stops.<br>The TCRmn register holds count value and stops.<br>The OVF bit of the TSRmn register is also held. |
| TAU stop | The TAUmEN bit of the PER0 register is cleared to 0. → | Power-off status<br>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 as 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 can also generate INTTMmn (timer interrupt) at any interval by making a software set TSmn = 1 and the count down start during the period of TEmn = 1.

The interrupt generation period can be calculated by the following expression.

| Generation period of INTTMmn (timer interrupt) = Period of count clock × (Set value of TDRmn + 1) |
|---|

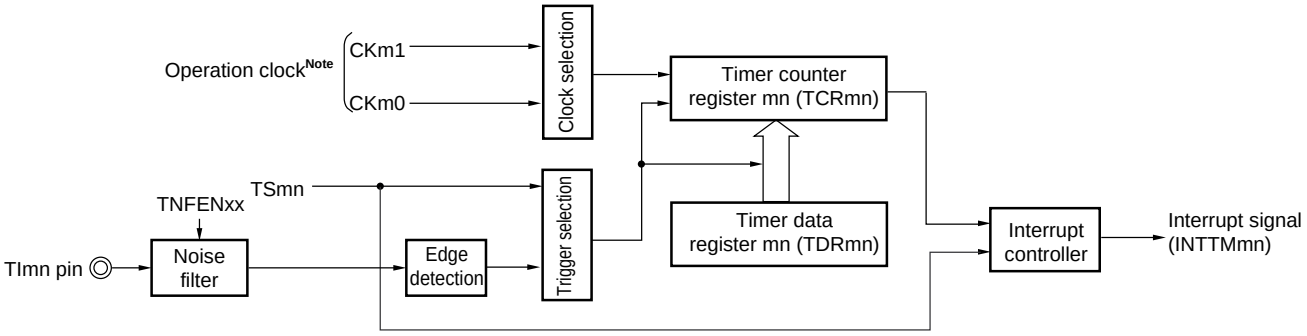
Timer count register mn (TCRmn) operates as a down counter in the one-count mode.

When the channel 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 status is set.

Timer count 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 count 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.

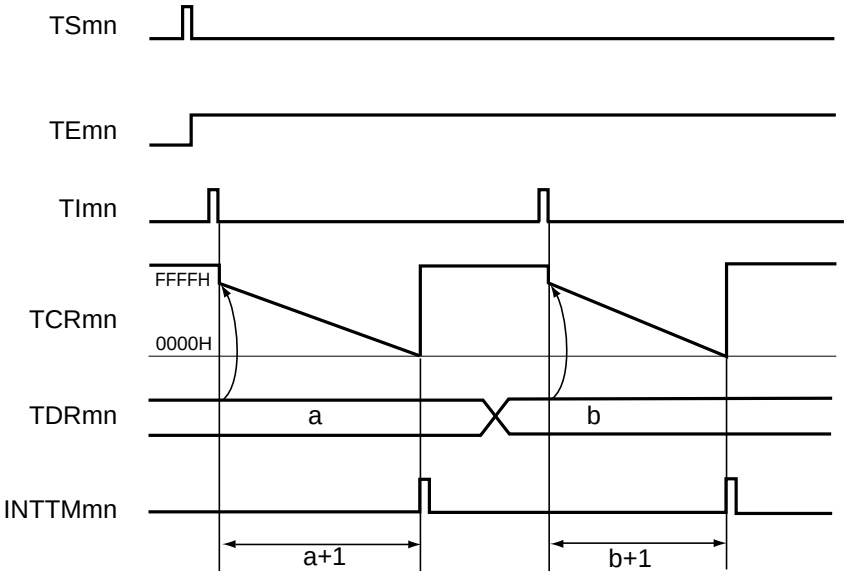
**Figure 7-58. Block Diagram of Operation as Delay Counter**




**Note** For using channels 1 and 3, the clock can be selected from CKm0, CKm1, CKm2 and CKm3.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

**Figure 7-59. Example of Basic Timing of Operation as Delay Counter**




**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

**2.** TSmn: Bit n of timer channel start register m (TSm)

TEmn: Bit n of timer channel enable status register m (TEm)

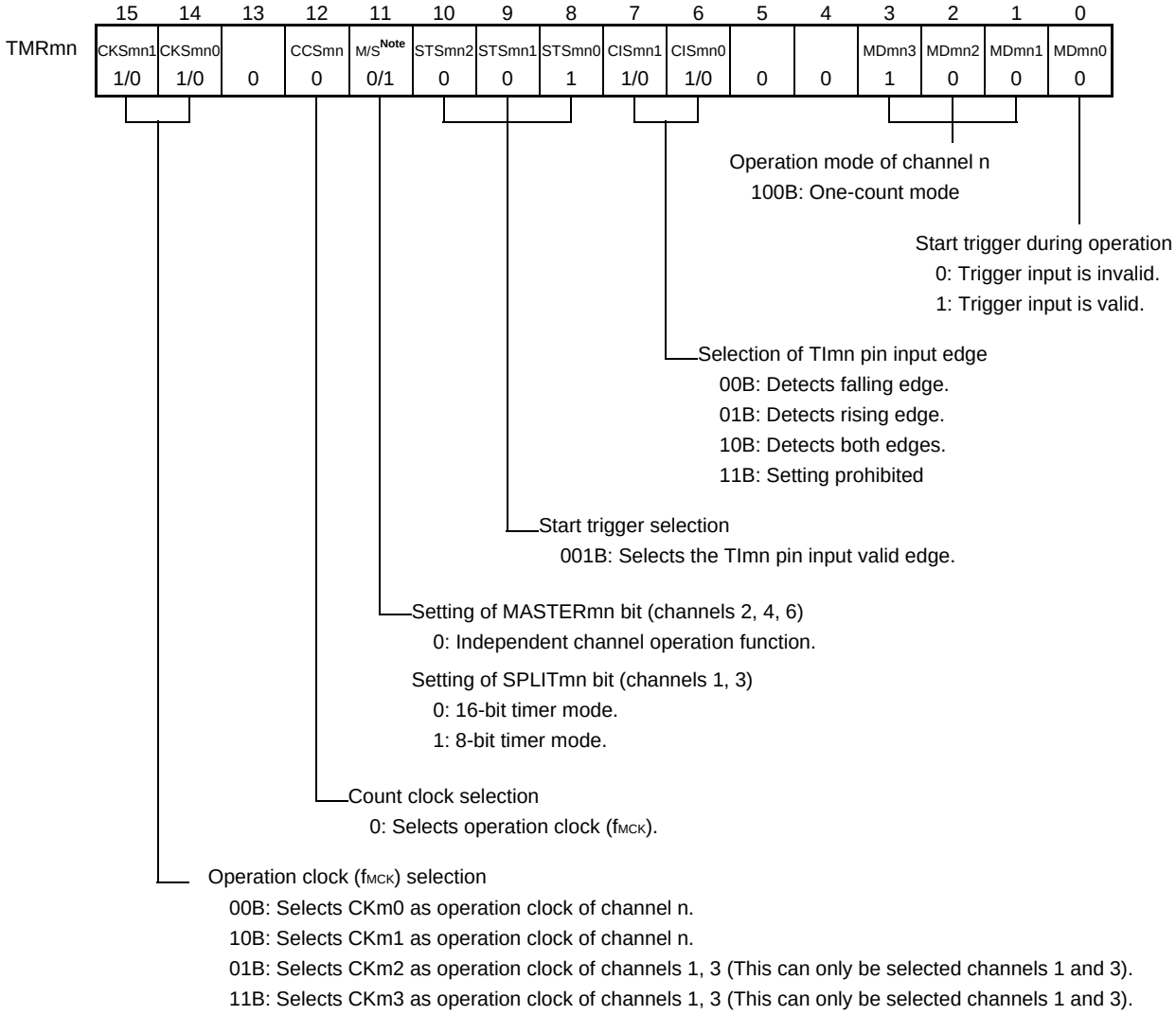
TImn: TImn pin input signal

TCRmn: Timer count register mn (TCRmn)

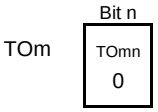
TDRmn: Timer data register mn (TDRmn)

**Figure 7-60. Example of Set Contents of Registers to Delay Counter (1/2)**

**(a) Timer mode register mn (TMRmn)**

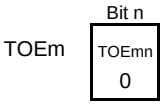



**(b) Timer output register m (TOm)**




0: Outputs 0 from TOmn.

**(c) Timer output enable register m (TOEm)**




0: Stops the TOmn output operation by counting operation.

**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 7-60. Example of Set Contents of Registers to Delay Counter (2/2)**

**(d) Timer output level register m (TOLm)**

| | Bit n | |
|---|---|---|
| TOLm | TOLmn<br>0 | 0: Cleared to 0 when TOMmn = 0 (master channel output mode). |

**(e) Timer output mode register m (TOMm)**

| | Bit n | |
|---|---|---|
| TOMm | TOMmn<br>0 | 0: Sets master channel output mode. |

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

**Figure 7-61. Operation Procedure When Delay Counter Function Is Used**

| | Software Operation | Hardware Status |
|---|---|---|
| TAU default setting | | Power-off status<br>(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 status. Each channel stops operating.<br>(Clock supply is started and writing to each register is enabled.) |
| | Sets timer clock select register m (TPSm).<br>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).<br>Sets timer mode register mn (TMRmn) (determines operation mode of channel).<br>INTTMmn output delay is set to timer data register mn (TDRmn).<br>Clears the TOEmn bit to 0 and stops operation of TOmn. | Channel stops operating.<br>(Clock is supplied and some power is consumed.) |
| Operation start | Sets the TSmn bit to 1. →<br>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 status is set. |
| | The counter starts counting down by the next start trigger detection. →<br>• Detects the TImn pin input valid edge.<br>• Sets the TSmn bit to 1 by the software. | Value of the TDRmn register is loaded to the timer count register mn (TCRmn). |
| During operation | Set value of the TDRmn register can be changed.<br>The TCRmn register can always be read.<br>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. →<br>The TTmn bit automatically returns to 0 because it is a trigger bit. | TEmn = 0, and count operation stops.<br>The TCRmn register holds count value and stops. |
| TAU stop | The TAUmEN bit of the PER0 register is cleared to 0. → | Power-off status<br>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 as 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} × Count clock period |
| Pulse width = {Set value of TDRmp (slave)} × Count clock period |

The master channel operates in the one-count mode and counts the delays. Timer count 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 count clock. When TCRmn = 0000H, it outputs INTTMmn and stops counting until the next start trigger is detected.

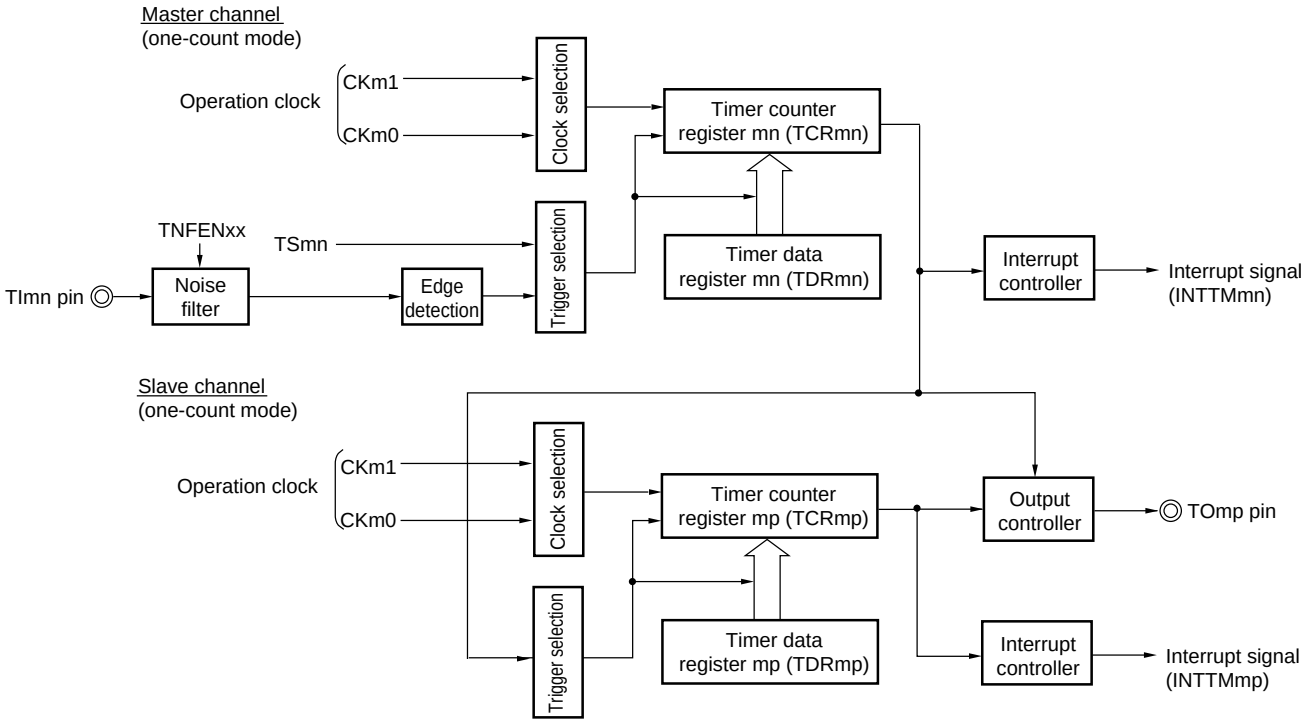
The slave channel operates in the one-count mode and counts the pulse width. The TCRmp register of the slave channel starts operation using INTTMmn of the master channel as a start trigger, and loads the value of the TDRmp register. The TCRmp register counts down from the value of The TDRmp register 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 count 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.

<R> **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 counting, therefore, an illegal waveform may be output in conflict with the timing of loading. Rewrite the TDRmn register after INTTMmn is generated and the TDRmp register after INTTMmp is generated.**

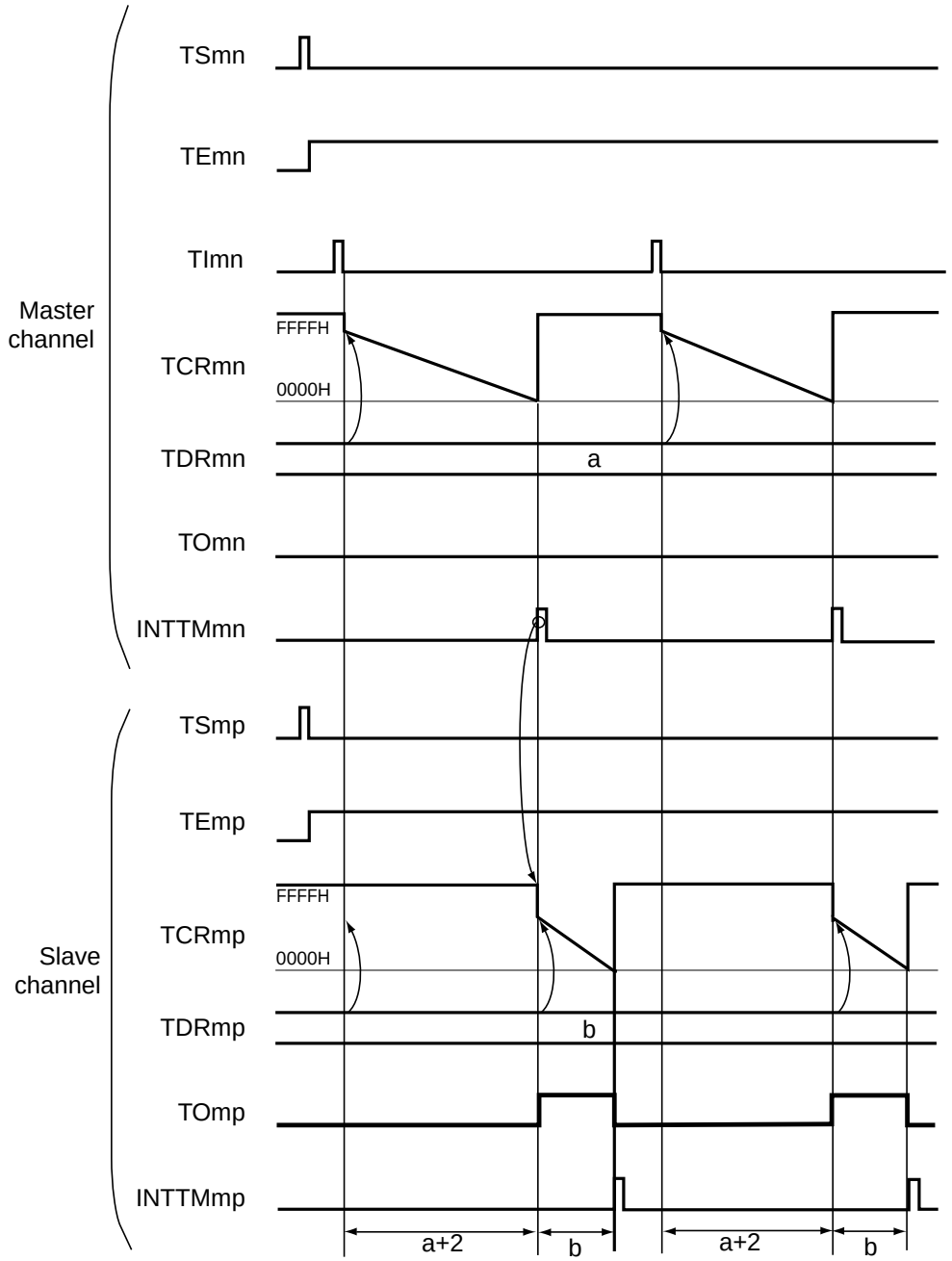
**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)

**Figure 7-62. Block Diagram of Operation as One-Shot Pulse Output Function**




**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)

**Figure 7-63. Example of Basic Timing of Operation as One-Shot Pulse Output Function**

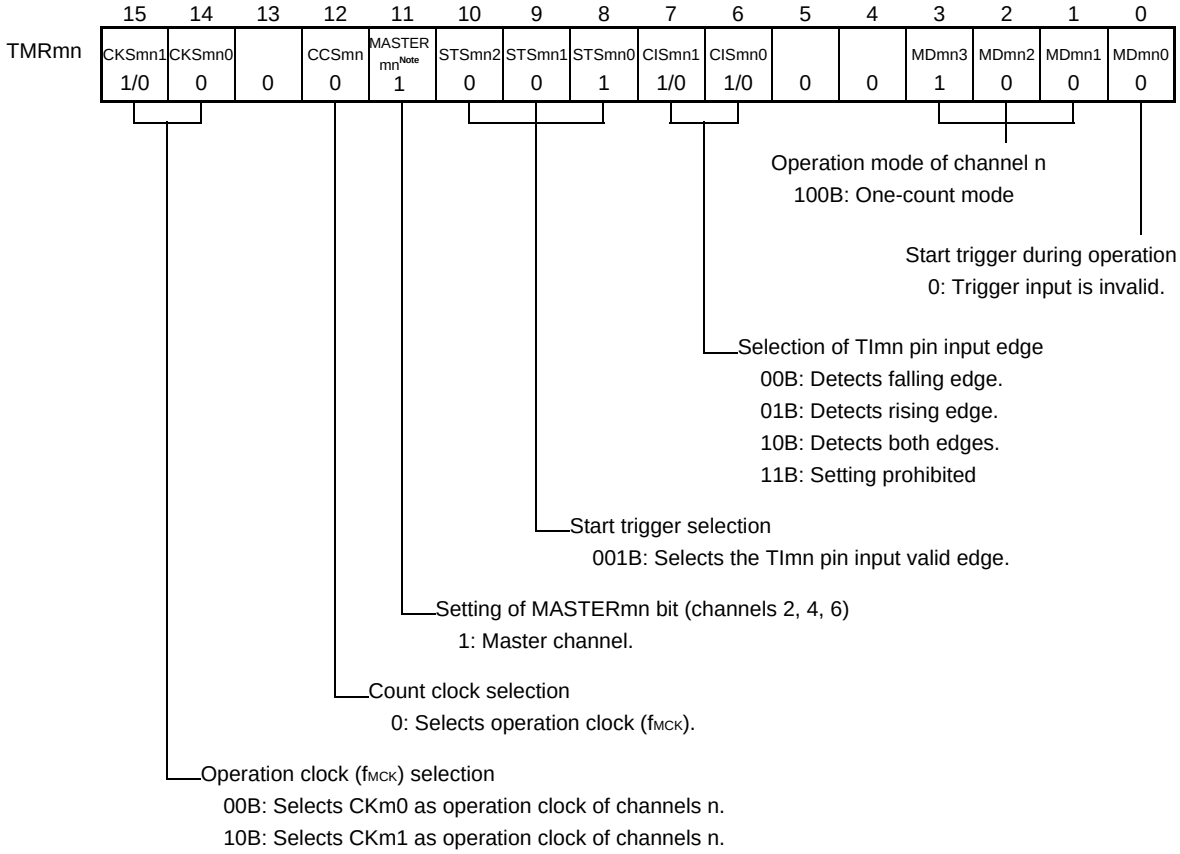



**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)

**2.** TSmn, TSmp: Bit n, p of timer channel start register m (TSm)
TEmn, TEmp: Bit n, p of timer channel enable status register m (TEm)
TImn, TImp: TImn and TImp pins input signal
TCRmn, TCRmp: Timer count registers mn, mp (TCRmn, TCRmp)
TDRmn, TDRmp: Timer data registers mn, mp (TDRmn, TDRmp)
TOmn, TOmp: TOmn and TOmp pins output signal

**Figure 7-64. Example of Set Contents of Registers When One-Shot Pulse Output Function Is Used (Master Channel)**

**(a) Timer mode register mn (TMRmn)**




**(b) Timer output register m (TOm)**

| | Bit n | |
|---|---|---|
| TOm | TOmn<br>0 | 0: Outputs 0 from TOmn. |

**(c) Timer output enable register m (TOEm)**

| | Bit n | |
|---|---|---|
| TOEm | TOEmn<br>0 | 0: Stops the TOmn output operation by counting operation. |

**(d) Timer output level register m (TOLm)**

| | Bit n | |
|---|---|---|
| TOLm | TOLmn<br>0 | 0: Cleared to 0 when TOMmn = 0 (master channel output mode). |

**(e) Timer output mode register m (TOMm)**

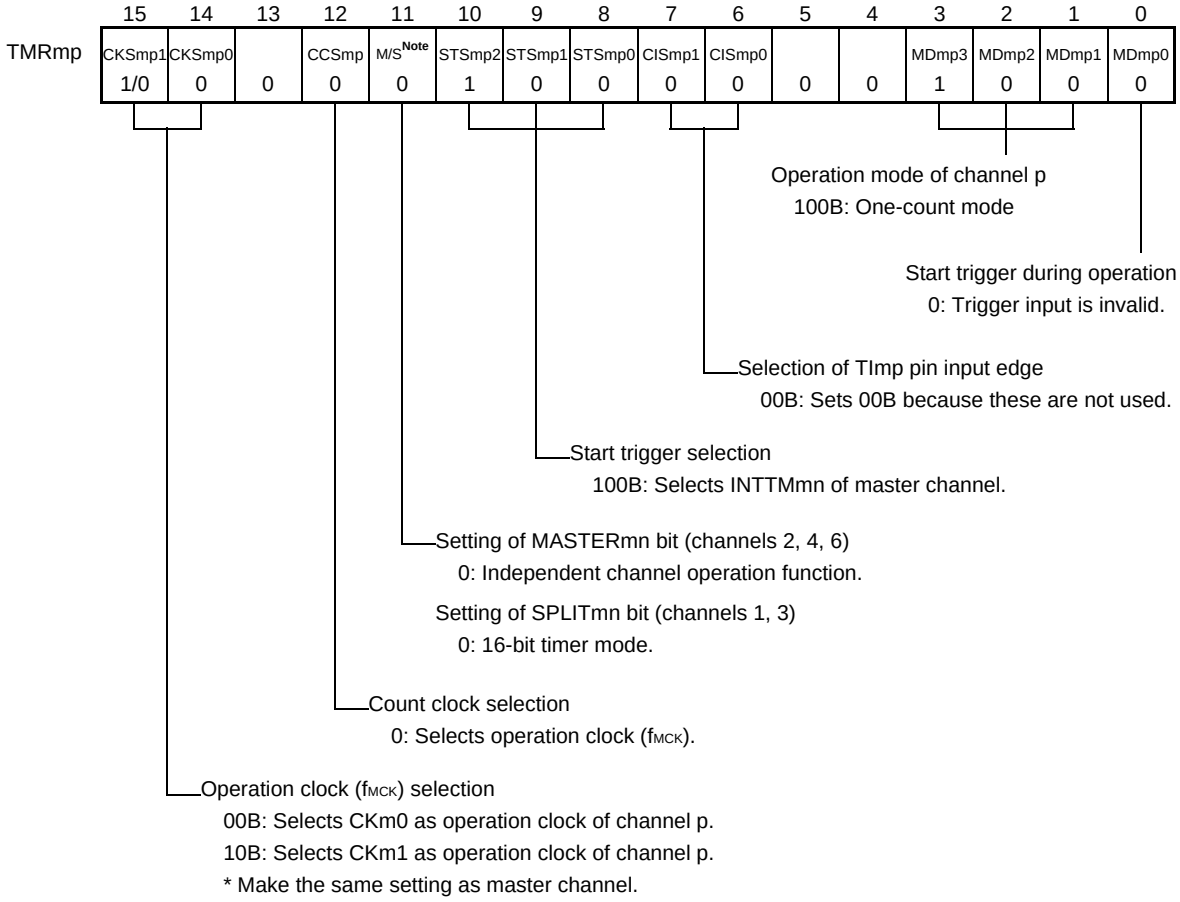
| | Bit n | |
|---|---|---|
| TOMm | TOMmn<br>0 | 0: Sets master channel output mode. |

**Note** TMRm2, TMRm4, TMRm6: MASTERmn = 1
TMRm0: Fixed to 0

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4, 6)

**Figure 7-65. Example of Set Contents of Registers When One-Shot Pulse Output Function Is Used (Slave Channel)**

**(a) Timer mode register mp (TMRmp)**




**(b) Timer output register m (TOm)**

| | Bit p | |
|---|---|---|
| TOm | TOmp<br>1/0 | 0: Outputs 0 from TOmp.<br>1: Outputs 1 from TOmp. |

**(c) Timer output enable register m (TOEm)**

| | Bit p | |
|---|---|---|
| TOEm | TOEmp<br>1/0 | 0: Stops the TOmp output operation by counting operation.<br>1: Enables the TOmp output operation by counting operation. |

**(d) Timer output level register m (TOLm)**

| | Bit p | |
|---|---|---|
| TOLm | TOLmp<br>1/0 | 0: Positive logic output (active-high)<br>1: Negative logic output (active-low) |

**(e) Timer output mode register m (TOMm)**

| | Bit p | |
|---|---|---|
| TOMm | TOMmp<br>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: Channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)

**Figure 7-66. Operation Procedure of One-Shot Pulse Output Function (1/2)**

| | Software Operation | Hardware Status |
|---|---|---|
| TAU default setting | | Power-off status<br>(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 status. Each channel stops operating.<br>(Clock supply is started and writing to each register is enabled.) |
| | Sets timer clock select register m (TPSm).<br>Determines clock frequencies of CKm0 and CKm1. | |
| Channel default setting | Sets the corresponding bit of the noise filter enable register 1 (NFEN1) to 1 (on).<br>Sets timer mode register mn, mp (TMRmn, TMRmp) of two channels to be used (determines operation mode of channels).<br>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.<br>(Clock is supplied and some power is consumed.) |
| | Sets slave channel.<br>The TOMmp bit of timer output mode register m (TOMm) is set to 1 (slave channel output mode).<br>Sets the TOLmp bit.<br>Sets the TOmp bit and determines default level of the TOmp output. → | The TOmp pin goes into Hi-Z output state.<br>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. |

(**Remark** is listed on the next page.)

**Figure 7-66. Operation Procedure of One-Shot Pulse Output Function (2/2)**

| | Software Operation | Hardware Status |
|---|---|---|
| Operation start | Sets the TOEmp bit (slave) to 1 (only when operation is resumed).<br>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 status.<br>Counter stops operating. |
| | Count operation of the master channel is started by start trigger detection of the master channel.<br>• Detects the TImn pin input valid edge.<br>• Sets the TSmn bit of the master channel to 1 by software[Note].<br><br>**Note** Do not set the TSmn bit of the slave channel to 1. | Master channel starts counting. |
| During operation | Set values of only the CISmn1 and CISmn0 bits of the TMRmn register can be changed.<br>Set values of the TMRmp, TDRmn, TDRmp registers, TOMmn, TOMmp, TOLmn, and TOLmp bits cannot be changed.<br>The TCRmn and TCRmp registers can always be read.<br>The TSRmn and TSRmp registers are not used.<br>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 count 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.<br>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.<br>The output level of TOmp becomes active one count clock after generation of INTTMmn from the master channel. It becomes inactive when TCRmp = 0000H, and the counting operation is stopped.<br>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.<br>The TCRmn and TCRmp registers hold count value and stop.<br>The TOmp output is not initialized but holds current status. |
| | 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<br>Clears the TOmp bit to 0 after the value to be held is set to the port register. → | The TOmp pin output level is held by port function. |
| | When holding the TOmp pin output level is not necessary<br>Setting not required. | |
| | The TAUmEN bit of the PER0 register is cleared to 0. → | Power-off status<br>All circuits are initialized and SFR of each channel is also initialized.<br>(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: Channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)

### 7.9.2 Operation as 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} × Count 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 channel 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 count register mn (TCRmn), and the counter counts down in synchronization with the count 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 channel 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.

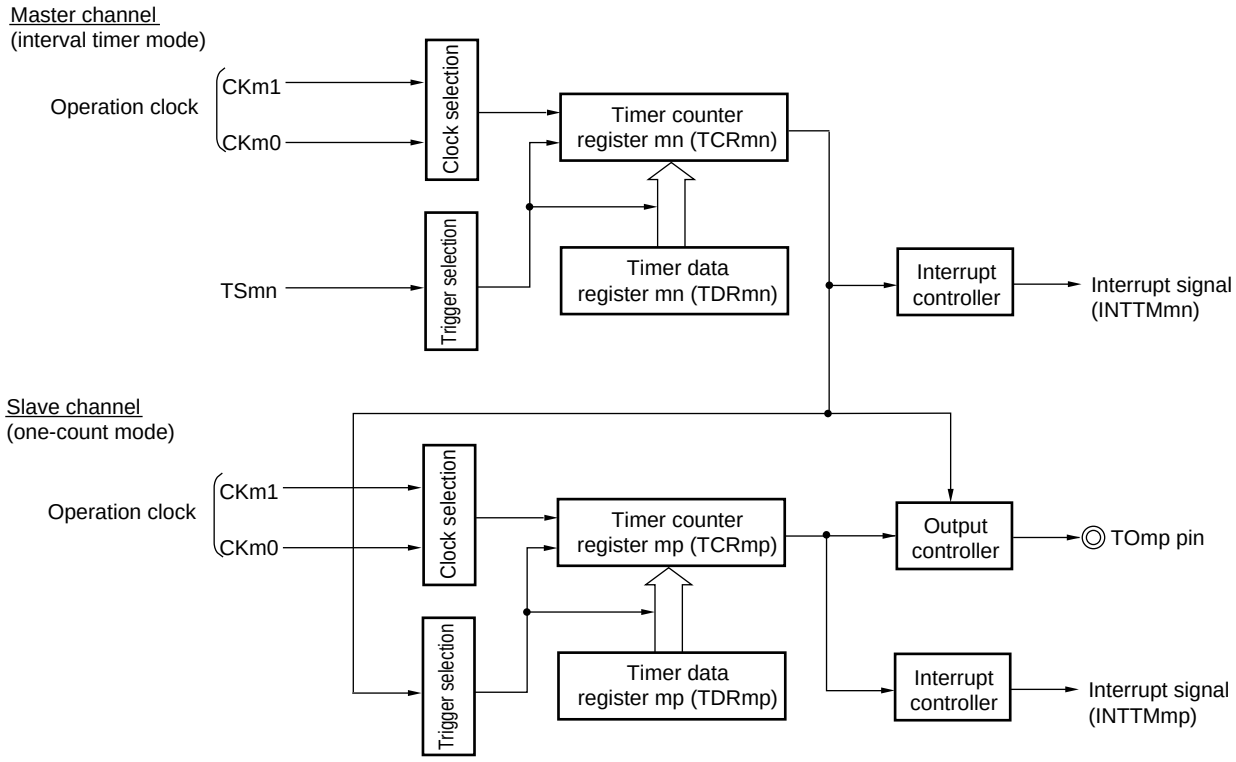
If two channels are used to output a PWM waveform, the period until the slave channel counts down to 0000H is the PWM output (TOmp) duty.

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.**

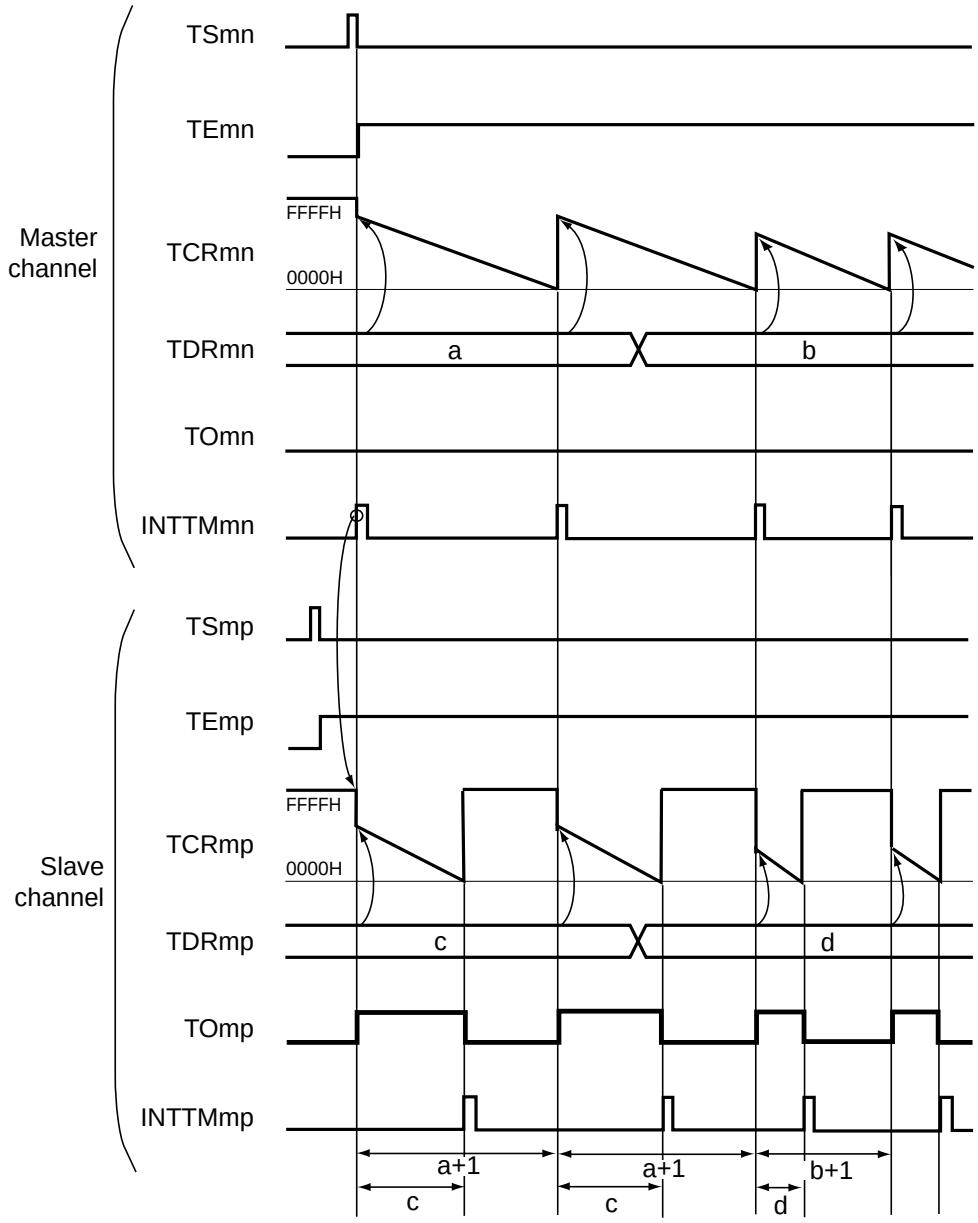
**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)

**Figure 7-67. Block Diagram of Operation as PWM Function**




**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)

**Figure 7-68. Example of Basic Timing of Operation as PWM Function**

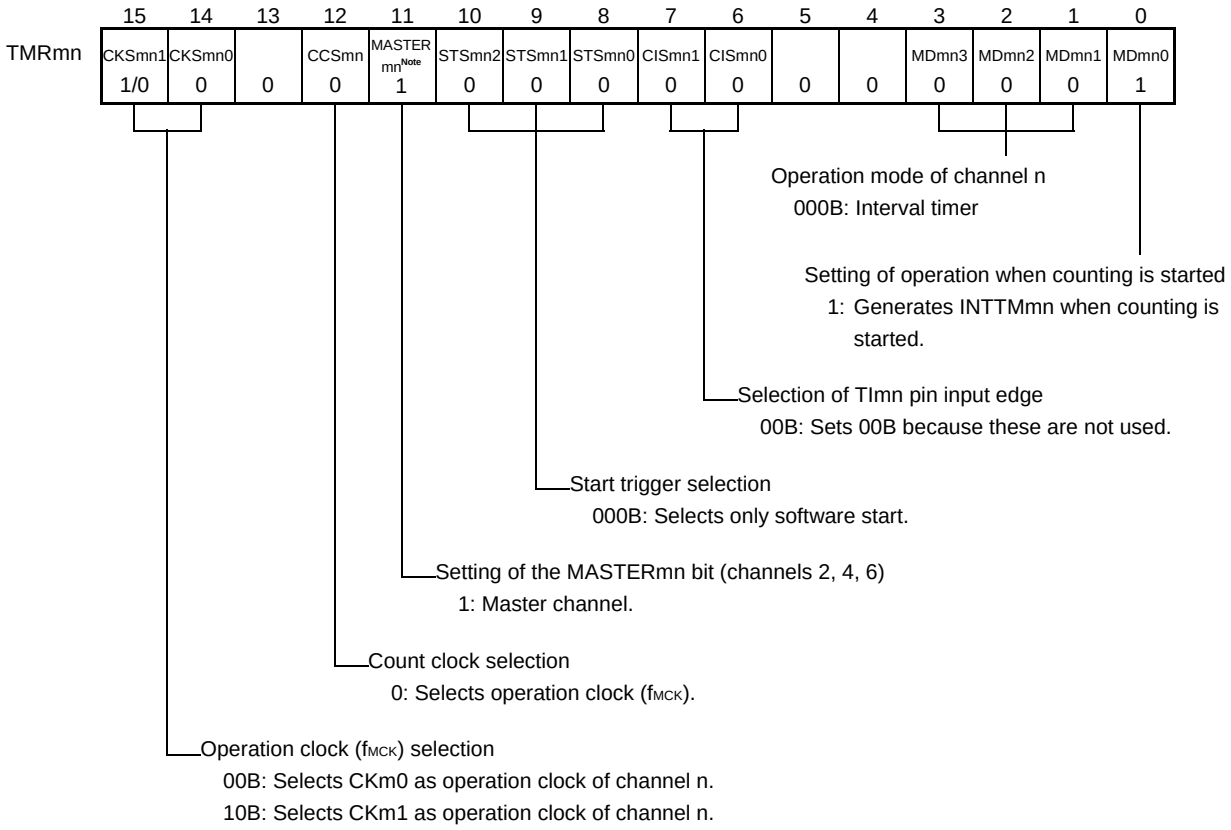



**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)

**2.** TSmn, TSmp: Bit n, p of timer channel start register m (TSm)
TEmn, TEmp: Bit n, p of timer channel enable status register m (TEm)
TCRmn, TCRmp: Timer count registers mn, mp (TCRmn, TCRmp)
TDRmn, TDRmp: Timer data registers mn, mp (TDRmn, TDRmp)
TOmn, TOmp: TOmn and TOmp pins output signal

**Figure 7-69. Example of Set Contents of Registers When PWM Function (Master Channel) Is 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: Cleared to 0 when TOMmn = 0 (master channel output mode). |

**(e) Timer output mode register m (TOMm)**

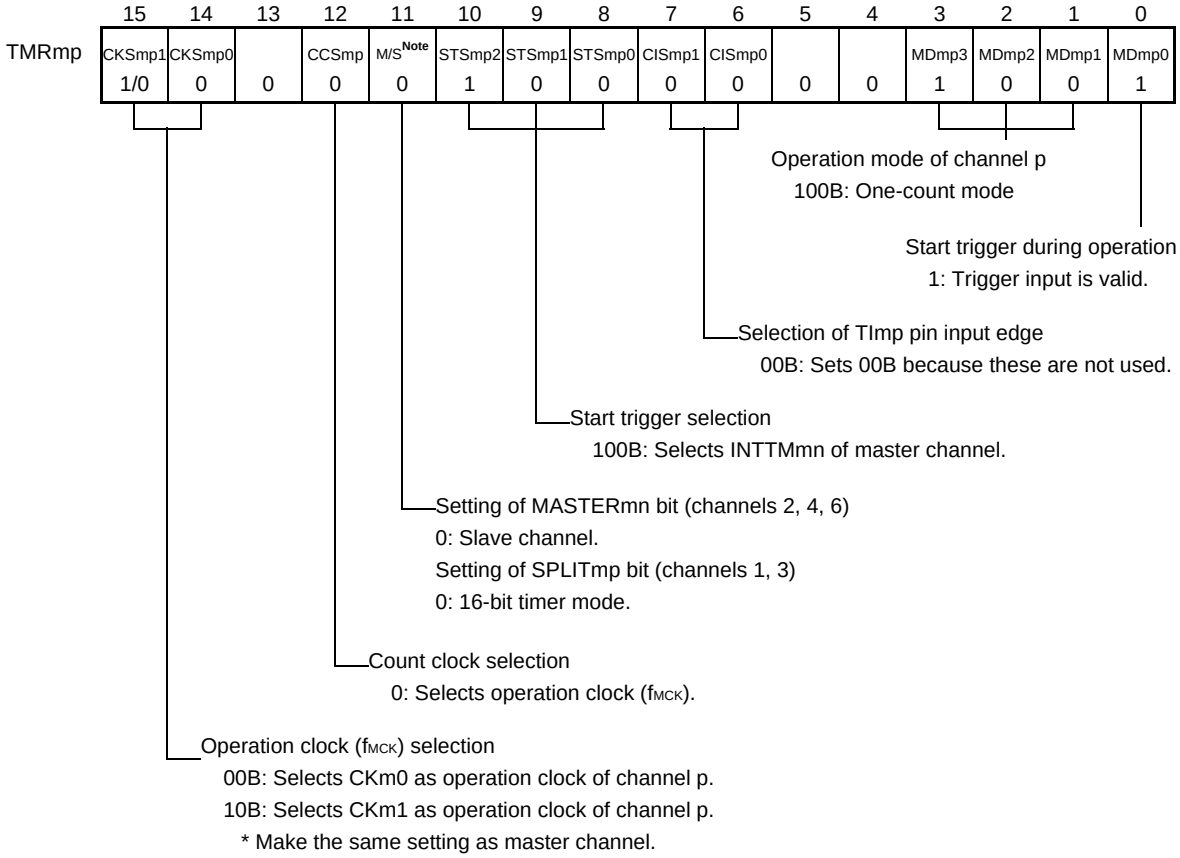
| | Bit n | | |
|---|---|---|---|
| TOMm | TOMmn | 0 | 0: Sets master channel output mode. |

**Note** TMRm2, TMRm4, TMRm6: MASTERmn = 1
TMRm0: Fixed to 0

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4, 6)

**Figure 7-70. Example of Set Contents of Registers When PWM Function (Slave Channel) Is Used**

**(a) Timer mode register mp (TMRmp)**




**(b) Timer output register m (TOm)**

| | Bit p | |
|---|---|---|
| TOm | TOmp<br>1/0 | 0: Outputs 0 from TOmp.<br>1: Outputs 1 from TOmp. |

**(c) Timer output enable register m (TOEm)**

| | Bit p | |
|---|---|---|
| TOEm | TOEmp<br>1/0 | 0: Stops the TOmp output operation by counting operation.<br>1: Enables the TOmp output operation by counting operation. |

**(d) Timer output level register m (TOLm)**

| | Bit p | |
|---|---|---|
| TOLm | TOLmp<br>1/0 | 0: Positive logic output (active-high)<br>1: Negative logic output (active-low) |

**(e) Timer output mode register m (TOMm)**

| | Bit p | |
|---|---|---|
| TOMm | TOMmp<br>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: Channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)

**Figure 7-71. Operation Procedure When PWM Function Is Used (1/2)**

| | Software Operation | Hardware Status |
|---|---|---|
| TAU default setting | | Stops supply of timer array unit m input clock.<br>(Clock supply is stopped and writing to each register is disabled.) |
| | Sets the TAUmEN bit of peripheral enable register 0 (PER0) to 1. → | Supplies timer array unit m input clock.<br>Each channel stops operating.<br>(Clock supply is started and writing to each register is enabled.) |
| | Sets timer clock select register m (TPSm).<br>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).<br>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.<br>(Clock is supplied and some power is consumed.) |
| | Sets slave channel.<br>The TOMmp bit of timer output mode register m (TOMm) is set to 1 (slave channel output mode).<br>Sets the TOLmp bit.<br>Sets the TOmp bit and determines default level of the TOmp output. → | The TOmp pin goes into Hi-Z output state.<br>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. |

(**Remark** is listed on the next page.)

**Figure 7-71. Operation Procedure When PWM Function Is Used (2/2)**

| | Software Operation | Hardware Status |
|---|---|---|
| Operation start | Sets the TOEmp bit (slave) to 1 (only when operation is resumed).<br>The TSmn (master) and TSmp (slave) bits of timer channel start register m (TSm) are set to 1 at the same time. → | TEmn = 1, TEmp = 1<br>→ When the master channel starts counting, INTTMmn is generated. Triggered by this interrupt, the slave channel also starts counting. |
| | The TSmn and TSmp bits automatically return to 0 because they are trigger bits. | |
| During operation | Set values of the TMRmn and TMRmp registers, TOMmn, TOMmp, TOLmn, and TOLmp bits cannot be changed.<br>Set values of the TDRmn and TDRmp registers can be changed after INTTMmn of the master channel is generated.<br>The TCRmn and TCRmp registers can always be read.<br>The TSRmn and TSRmp registers are not used. | The counter of the master channel loads the TDRmn register value to timer count 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.<br>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.<br>The output level of TOmp becomes active one count clock after generation of the INTTMmn output from the master channel. It becomes inactive when TCRmp = 0000H, and the counting operation is stopped.<br>After that, the above operation is repeated. |
| Operation stop | The TTmn (master) and TTmp (slave) bits are set to 1 at the same time. → | TEmn, TEmp = 0, and count operation stops. |
| | The TTmn and TTmp bits automatically return to 0 because they are trigger bits. | The TCRmn and TCRmp registers hold count value and stop.<br>The TOmp output is not initialized but holds current status. |
| | 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<br>Clears the TOmp bit to 0 after the value to be held is set to the port register. → | The TOmp pin output level is held by port function. |
| | When holding the TOmp pin output level is not necessary<br>Setting not required. | |
| | The TAUmEN bit of the PER0 register is cleared to 0. → | Power-off status<br>All circuits are initialized and SFR of each channel is also initialized.<br>(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: Channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)

### 7.9.3 Operation as 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} × Count 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 count 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 count clock after generation of INTTMmn from the master channel, and inactive when TCRmp = 0000H.

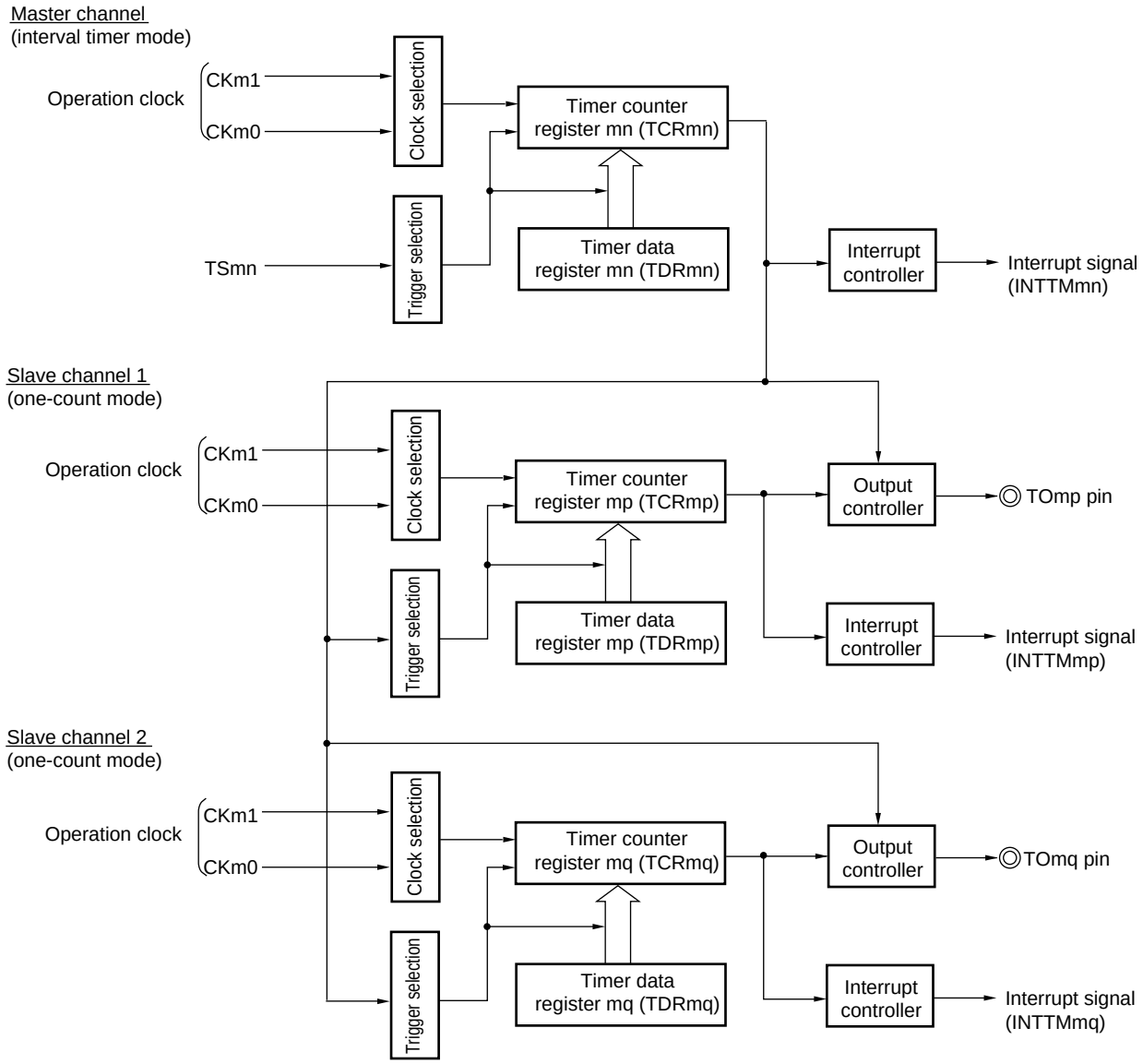
In the same way as the TCRmp register of the slave channel 1, the TCRmq register of the slave channel 2 operates in one-count mode, counts the duty factor, and outputs a PWM waveform from the TOmq pin. The TCRmq register loads the value of the TDRmq register, using INTTMmn of the master channel as a 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 count 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).**

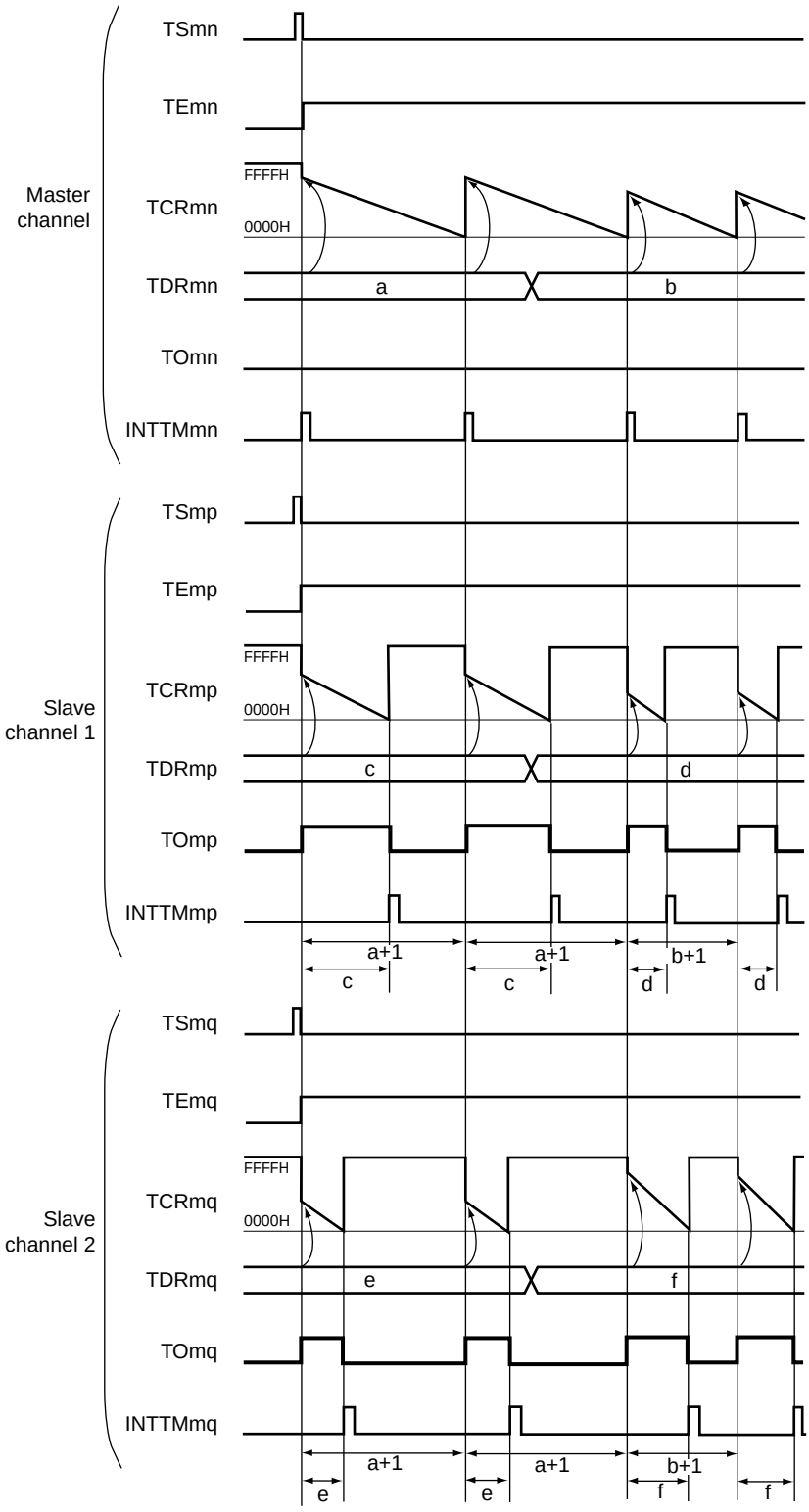
**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4)
p: Slave channel number, q: Slave channel number
n < p < q ≤ 7 (Where p and q are integers greater than n)

**Figure 7-72. Block Diagram of Operation as Multiple PWM Output Function (Output Two Types of PWMs)**




**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4)
p: Slave channel number, q: Slave channel number
$n < p < q \leq 7$ (Where p and q are integers greater than n)

**Figure 7-73. Example of Basic Timing of Operation as Multiple PWM Output Function (Output Two Types of PWMs)**




(**Remark** is listed on the next page.)

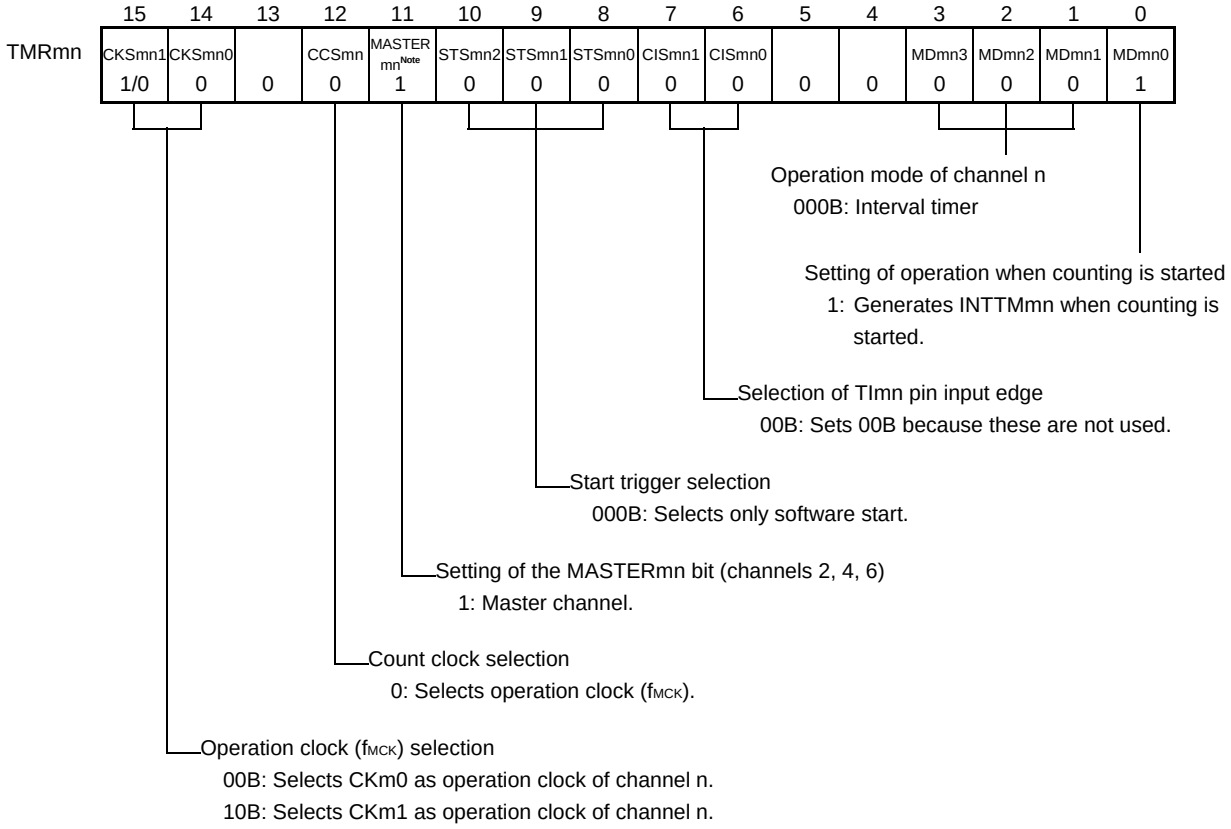
**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4)
p: Slave channel number, q: Slave channel number
n < p < q ≤ 7 (Where p and q are integers greater than n)

**2.**

| | |
|---|---|
| TSmn, TSmp, TSmq: | Bit n, p, q of timer channel start register m (TSm) |
| TEmn, TEmp, TEmq: | Bit n, p, q of timer channel enable status register m (TEm) |
| TCRmn, TCRmp, TCRmq: | Timer count registers mn, mp, mq (TCRmn, TCRmp, TCRmq) |
| TDRmn, TDRmp, TDRmq: | Timer data registers mn, mp, mq (TDRmn, TDRmp, TDRmq) |
| TOmn, TOmp, TOmq: | TOmn, TOmp, and TOmq pins output signal |

**Figure 7-74. Example of Set Contents of Registers When Multiple PWM Output Function (Master Channel) Is Used**

**(a) Timer mode register mn (TMRmn)**




**(b) Timer output register m (TOm)**

| | Bit n | |
|---|---|---|
| TOm | TOmn<br>0 | 0: Outputs 0 from TOmn. |

**(c) Timer output enable register m (TOEm)**

| | Bit n | |
|---|---|---|
| TOEm | TOEmn<br>0 | 0: Stops the TOmn output operation by counting operation. |

**(d) Timer output level register m (TOLm)**

| | Bit n | |
|---|---|---|
| TOLm | TOLmn<br>0 | 0: Cleared to 0 when TOMmn = 0 (master channel output mode). |

**(e) Timer output mode register m (TOMm)**

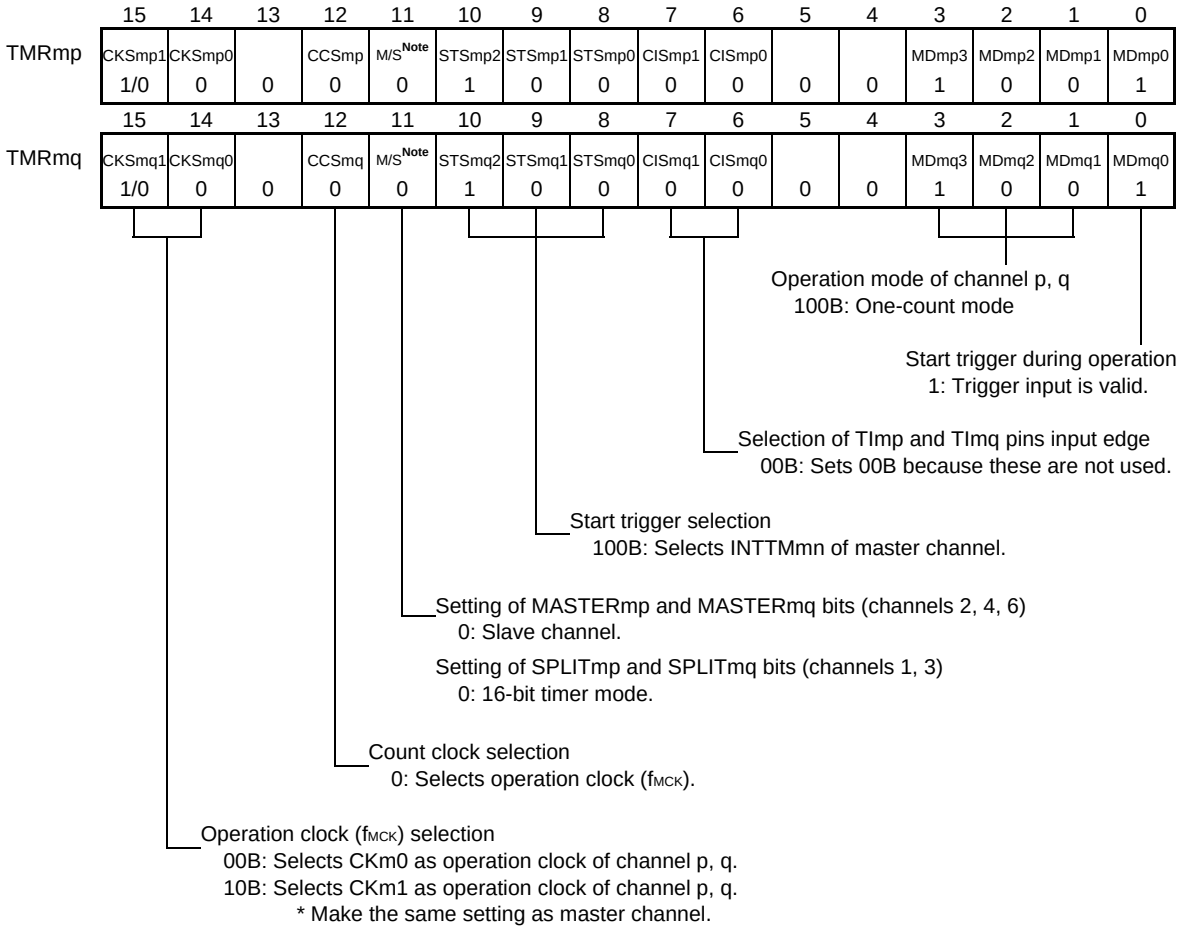
| | Bit n | |
|---|---|---|
| TOMm | TOMmn<br>0 | 0: Sets master channel output mode. |

**Note** TMRm2, TMRm4, TMRm6: MASTERmn = 1
TMRm0: Fixed to 0

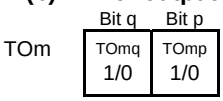
**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4)

**Figure 7-75. Example of Set Contents of Registers When Multiple PWM Output Function (Slave Channel) Is Used (Output Two Types of PWMs)**

**(a) Timer mode register mp, mq (TMRmp, TMRmq)**

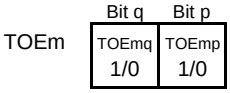



**(b) Timer output register m (TOm)**



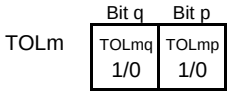

0: Outputs 0 from TOmp or TOmq.
1: Outputs 1 from TOmp or TOmq.

**(c) Timer output enable register m (TOEm)**



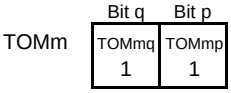

0: Stops the TOmp or TOmq output operation by counting operation.
1: Enables the TOmp or TOmq output operation by counting operation.

**(d) Timer output level register m (TOLm)**




0: Positive logic output (active-high)
1: Negative logic output (active-low)

**(e) Timer output mode register m (TOMm)**




1: Sets the slave channel output mode.

**Note** TMRm2, TMRm4, TMRm6: MASTERmp, MASTERmq bit
TMRm1, TMRm3: SPLITmp, SPLIT0q bit
TMRm5, TMRm7: Fixed to 0

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4)
p: Slave channel number, q: Slave channel number
n < p < q ≤ 7 (Where p and q are integers greater than n)

**Figure 7-76. Operation Procedure When Multiple PWM Output Function Is Used (1/2)**

| | Software Operation | Hardware Status |
|---|---|---|
| TAU default setting | | Stops supply of timer array unit m input clock.<br>(Clock supply is stopped and writing to each register is disabled.) |
| | Sets the TAUmEN bit of peripheral enable register 0 (PER0) to 1. → | Supplies timer array unit m input clock.<br>Each channel stops operating.<br>(Clock supply is started and writing to each register is enabled.) |
| | Sets timer clock select register m (TPSm).<br>Determines clock frequencies of CKm0 and CKm1. | |
| Channel default setting | Sets timer mode registers mn, mp, mq (TMRmn, TMRmp, TMRmq) of each channel to be used (determines operation mode of channels).<br>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.<br>(Clock is supplied and some power is consumed.) |
| | Sets slave channels.<br>The TOMmp and TOMmq bits of timer output mode register m (TOMm) are set to 1 (slave channel output mode).<br>Sets the TOLmp and TOLmq bits.<br>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.<br>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. |

(**Remark** is listed on the next page.)

**Figure 7-76. Operation Procedure When Multiple PWM Output Function Is Used (2/2)**

| | Software Operation | Hardware Status |
|---|---|---|
| Operation start | (Sets the TOEmp and TOEmq (slave) bits to 1 only when resuming operation.)<br>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.<br>The TSmn, TSmp, and TSmq bits automatically return to 0 because they are trigger bits. | TEmn = 1, TEmp, TEmq = 1<br>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.<br>Set values of the TDRmn, TDRmp, and TDRmq registers can be changed after INTTMmn of the master channel is generated.<br>The TCRmn, TCRmp, and TCRmq registers can always be read.<br>The TSRmn, TSRmp, and TSR0q registers are not used. | The counter of the master channel loads the TDRmn register value to timer count 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.<br>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 count clock after generation of the INTTMmn output from the master channel. It becomes inactive when TCRmp = 0000H, and the counting operation is stopped.<br>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 count clock after generation of the INTTMmn output from the master channel. It becomes inactive when TCRmq = 0000H, and the counting operation is stopped.<br>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.<br>The TTmn, TTmp, and TTmq bits automatically return to 0 because they are trigger bits. | TEmn, TEmp, TEmq = 0, and count operation stops.<br>The TCRmn, TCRmp, and TCRmq registers hold count value and stop.<br>The TOmp and TOmq output are not initialized but hold current status. |
| | 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<br>Clears the TOmp and TOmq bits to 0 after the value to be held is set to the port register.<br>When holding the TOmp and TOmq pin output levels are not necessary<br>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. | Power-off status<br>All circuits are initialized and SFR of each channel is also initialized.<br>(The TOmp and TOmq bits are cleared to 0 and the TOmp and TOmq pins are set to port mode.) |

Operation is resumed.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4)
p: Slave channel number, q: Slave channel number
$n < p < q \leq 7$ (Where p and q are a consecutive integer greater than n)

## 7.10 Cautions When Using Timer Array Unit

### 7.10.1 Cautions When Using Timer output

Depends on products, a pin is assigned a timer output and other alternate functions. In this case, outputs of the other alternate functions must be set in initial status.

For details, see **4.5 Register Settings When Using Alternate Function**.

**(a) Using TO02 to TO07 outputs (80-pin products only)**

In addition to clearing the port mode register (the PMxx bit) and the port register (the Pxx bit) to 0, be sure to clear the corresponding bit of LCD port function register 4 (PFSEG37 to PFSEG32) to “0”.

**(b) Using TO00 and TO01 outputs assigned to the P43 and P41**

So that the alternated PCLBUZ1 and PCLBUZ0 outputs become 0, not only set the port mode register (the PM43 and PM41 bits) and the port register (the P43 and P41 bits) to 0, but also use the bit 7 of the clock output select register n (CKSn) with the same setting as the initial status.

**(c) Using TO02 to TO07 outputs assigned to the P07 to P02**

So that the alternated P07/SO00/TxD0, P06/SDA00, P05/SCK00/SCL00, P04/TxD1, P03/SDA10 and P02/SCL10 outputs become 1, not only set the port mode register (the PM07 to PM02 bits) and the port register (the P07 to P02 bits) to 0, but also use the serial channel enable status register 0 (SE0), serial output register 0 (SO0), and serial output enable register 0 (SOE0) with the same setting as the initial status.

**(d) Using TO05 and TO06 outputs assigned to the P04 and P03 (When PIOR3 = 1)**

So that the alternated VCOUT0 and VCOUT1 outputs become 0, not only set the port mode register (the PM04 and PM03 bits) and the port register (the P04 and P03 bits) to 0, but also use the bit 1 of the comparator output control register (COMPOCR) with the same setting as the initial status.

# CHAPTER 8 REAL-TIME CLOCK 2

## 8.1 Functions of Real-time Clock 2

Real-time clock 2 (RTC2) has the following functions.

- Counters of year, month, day of the week, date, hour, minute, and second, that can count up to 99 years (with leap year correction function)
- Constant-period interrupt function (period: 0.5 seconds, 1 second, 1 minute, 1 hour, 1 day, 1 month)
- Alarm interrupt function (alarm: day of the week, hour, and minute)
- Pin output function of 1 Hz (normal 1 Hz output, high accuracy 1 Hz output)

The real-time clock interrupt signal (INTRTC) can be utilized for wakeup from STOP mode and triggering an A/D converter's SNOOZE mode.

**Cautions 1. The year, month, week, day, hour, minute and second can only be counted when a subsystem clock ($f_{SUB}$ = 32.768 kHz) is selected as the operation clock of real-time clock 2.**
**When the low-speed oscillation clock ($f_{IL}$ = 15 kHz) is selected, only the constant-period interrupt function is available.**
**However, the constant-period interrupt interval when $f_{IL}$ is selected will be calculated with the constant-period (the value selected with RTCC0 register) × $f_{SUB}/f_{IL}$.**

**2. When using the high accuracy 1 Hz pin output, set the high-speed on-chip oscillator clock ($f_{IH}$) to 24 MHz.**

## 8.2 Configuration of Real-time Clock 2

Real-time clock 2 includes the following hardware.

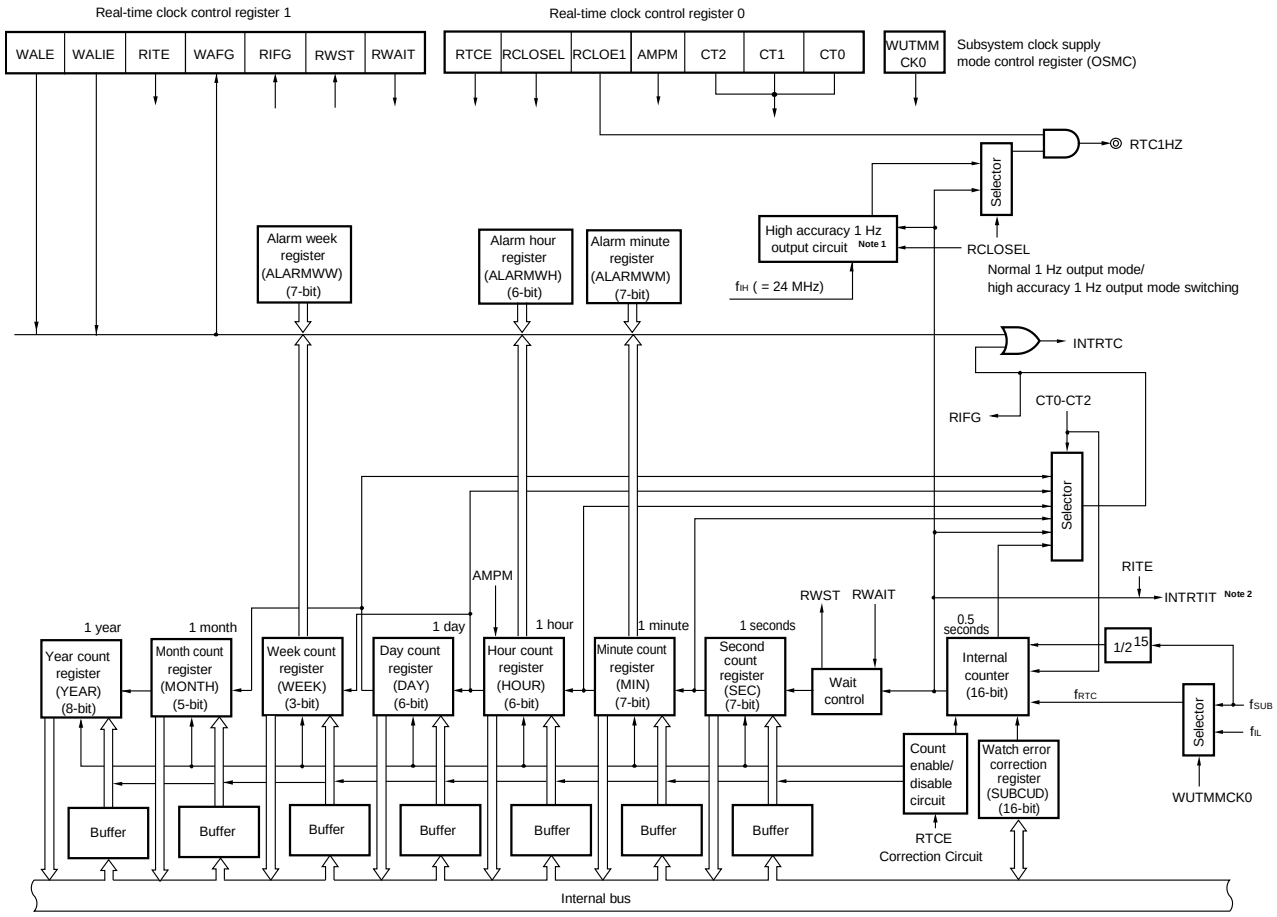
**Table 8-1. Configuration of Real-time Clock 2**

| Item | Configuration |
|---|---|
| Counter | Counter (16-bit) |
| Control registers | Peripheral enable register 0 (PER0) |
| | Peripheral enable register 1 (PER1) |
| | Subsystem clock supply mode control register (OSMC) |
| | Power-on-reset status register (PORSR) |
| | Real-time clock control register 0 (RTCC0) |
| | Real-time clock control register 1 (RTCC1) |
| | Second count register (SEC) |
| | Minute count register (MIN) |
| | Hour count register (HOUR) |
| | Day count register (DAY) |
| | Week count register (WEEK) |
| | Month count register (MONTH) |
| | Year count register (YEAR) |
| | Watch error correction register (SUBCUD) |
| | Alarm minute register (ALARMWM) |
| | Alarm hour register (ALARMWH) |
| | Alarm week register (ALARMWW) |

Figure 8-1 shows the real-time clock 2 diagram.

<R>

**Figure 8-1. Real-time Clock 2 Diagram**




**Notes 1.** A high-speed on-chip oscillator (HOCO: 24 MHz) can be used for high precision 1 Hz output. HOCO must be set to ON in order to run in high precision 1 Hz output mode. To run in normal 1 Hz mode, there is no need to set HOCO to ON.

**2.** An interrupt that indicates the timing to get the correction value from the clock error correction register (SUBCUD).

The fetch timing is 1 second ($f_{SUB}$ base) interval.

## 8.3 Registers Controlling Real-time Clock 2

Real-time clock 2 is controlled by the following registers.

- Peripheral enable register 0 (PER0)
- Peripheral enable register 1 (PER1)
- Subsystem clock supply mode control register (OSMC)
- Power-on-reset status register (PORSR)
- Real-time clock control register 0 (RTCC0)
- Real-time clock control register 1 (RTCC1)
- Second count register (SEC)
- Minute count register (MIN)
- Hour count register (HOUR)
- Day count register (DAY)
- Week count register (WEEK)
- Month count register (MONTH)
- Year count register (YEAR)
- Watch error correction register (SUBCUD)
- Alarm minute register (ALARMWM)
- Alarm hour register (ALARMWH)
- Alarm week register (ALARMWW)

The following shows the register states depending on reset sources.

| Reset Source | System-related Register[Note 1] | Calendar-related Register[Note 2] |
|---|---|---|
| POR | Reset | Not reset |
| External reset | Retained | Retained |
| WDT | Retained | Retained |
| TRAP | Retained | Retained |
| LVD | Retained | Retained |
| Other internal reset | Retained | Retained |

**Notes 1.** RTCC0, RTCC1, SUBCUD

**2.** SEC, MIN, HOUR, DAY, WEEK, MONTH, YEAR, ALARMWM, ALARMWH, ALARMWW, (Counter)

Reset generation does not reset the SEC, MIN, HOUR, DAY, WEEK, MONTH, YEAR, ALARMWM, ALARMWH, or ALARMWW register. Initialize all the registers after power on.

The PORSR register is used to check the occurrence of a power-on reset.

### 8.3.1 Peripheral enable register 0 (PER0)

This register is used to enable or disable supplying the clock to the register used for real-time clock 2. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

When the real-time clock 2 registers are manipulated, be sure to set bit 7 (RTCWEN) of this register to 1.

The PER0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 8-2. Format of Peripheral Enable Register 0 (PER0)**

Address: F00F0H After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PER0 | RTCWEN | IRDAEN | ADCEN | IICA0EN | SAU1EN | SAU0EN | 0 | TAU0EN |

| RTCWEN | Control of internal clock supply to real-time clock 2 |
|---|---|
| 0 | Stops input clock supply.<br>• SFR used by real-time clock 2 cannot be written.<br>• Real-time clock 2 can operate. |
| 1 | Enables input clock supply.<br>• SFR used by real-time clock 2 can be read/written.<br>• Real-time clock 2 can operate. |

**Cautions 1. The clock error correction register (SUBCUD) can be read or written by setting RTCWEN of peripheral enable register 0 (PER0) to 1 or setting FMCEN of peripheral enable register 1 (PER1) to 1.**

**2. When using real-time clock 2, first set the RTCWEN bit to 1 and then set the following registers, while oscillation of the count clock ($f_{RTC}$) is stable. If RTCWEN = 0, writing to the control registers of real-time clock 2 is ignored, and read values are the values set when RTCWEN = 1 (except for the subsystem clock supply mode control register (OSMC) and power-on reset status register (PORSR)).**

- **Real-time clock control register 0 (RTCC0)**
- **Real-time clock control register 1 (RTCC1)**
- **Second count register (SEC)**
- **Minute count register (MIN)**
- **Hour count register (HOUR)**
- **Day count register (DAY)**
- **Week count register (WEEK)**
- **Month count register (MONTH)**
- **Year count register (YEAR)**
- **Watch error correction register (SUBCUD)**
- **Alarm minute register (ALARMWM)**
- **Alarm hour register (ALARMWH)**
- **Alarm week register (ALARMWW)**

**3. Be sure to set bit 1 to “0”.**

### 8.3.2 Peripheral enable register 1 (PER1)

This register is used to enable or disable supplying the clock to the register used for the subsystem clock frequency measurement circuit. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

Reading and writing the clock error correction register (SUBCUD), a register used to control the real-time clock 2, is enabled by setting bit 6 (FMCEN) of this register to 1.

The PER1 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 8-3. Format of Peripheral Enable Register 1 (PER1)**

Address: F007AH After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PER1 | TMKAEN | FMCEN | CMPEN | OSDCEN | DTCEN | 0 | 0 | DSADCEN |

| FMCEN | Control of internal clock supply to subsystem clock frequency measurement circuit |
|---|---|
| 0 | Stops input clock supply.<br>• SFR used by the subsystem clock frequency measurement circuit cannot be written.<br>• SUBCUD register used by real-time clock 2 cannot be written.<br>• The subsystem clock frequency measurement circuit is in the reset status. |
| 1 | Enables input clock supply.<br>• SFR used by the subsystem clock frequency measurement circuit can be read/written.<br>• SUBCUD register used by real-time clock 2 can be read/written. |

**Cautions 1. The clock error correction register (SUBCUD) can be read or written by setting RTCWEN of peripheral enable register 0 (PER0) to 1 or setting FMCEN of peripheral enable register 1 (PER1) to 1.**

**2. Be sure to set bits 1 and 2 to “0”.**

### 8.3.3 Subsystem clock supply mode control register (OSMC)

This register is used to reduce power consumption by stopping unnecessary clock functions.

If the RTCLPC bit is set to 1, power consumption can be reduced, because clock supply to the peripheral functions other than real-time clock 2, 12-bit interval timer, clock output/buzzer output controller, LCD controller/driver, 8-bit interval timer, and oscillation stop detector is stopped in STOP mode or in HALT mode while the subsystem clock is selected as the CPU clock.

In addition, the OSMC register is used to select the operation clock of real-time clock 2, 12-bit interval timer, clock output/buzzer output controller, LCD controller/driver, 8-bit interval timer, and subsystem clock frequency measurement circuit.

The OSMC register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 8-4. Format of Subsystem Clock Supply Mode Control Register (OSMC)**

Address: F00F3H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| OSMC | RTCLPC | 0 | 0 | WUTMMCK0 | 0 | 0 | 0 | 0 |

| RTCLPC | In STOP mode and in HALT mode while the CPU operates using the subsystem clock |
|---|---|
| 0 | Enables subsystem clock supply to peripheral functions.<br>For peripheral functions for which operation is enabled, see **Tables 24-1** and **24-2**. |
| 1 | Stops subsystem clock supply to peripheral functions other than real-time clock 2, 12-bit interval timer, clock output/buzzer output controller, LCD controller/driver, 8-bit interval timer, and oscillation stop detector. |

| WUTMMCK0 | Selection of operation clock for real-time clock 2, 12-bit interval timer, and LCD controller/driver. | Selection of clock output from PCLBUZn pin of clock output/buzzer output controller and selection of operation clock for 8-bit interval timer. | Operation of subsystem clock frequency measurement circuit. |
|---|---|---|---|
| 0 | Subsystem clock ($f_{SUB}$) | Selecting the subsystem clock ($f_{SUB}$) is enabled. | Enable |
| 1 | Low-speed on-chip oscillator clock ($f_{IL}$) | Selecting the subsystem clock ($f_{SUB}$) is disabled. | Disable |

**Cautions 1. Setting the RTCLPC bit to 1 can reduce current consumption in STOP mode and in HALT mode with the CPU operating on the subsystem clock. However, setting the RTCLPC bit to 1 means that there is no clock supply to peripheral circuits other than real-time clock 2, 12-bit interval timer, clock output/buzzer output controller, and LCD controller/driver in HALT mode with the CPU operating on the subsystem clock. Before setting the system to HALT mode with the CPU operating on the subsystem clock, therefore, be sure to set bit 7 (RTCWEN) of peripheral enable register 0 (PER0) and bit 7 (TMKAEN) of peripheral enable register 1 (PER1) to 1, and bits 0, 2, and 3 of PER0, and bit 5 of PER1 to 0.**

**2. If the subsystem clock is oscillating, be sure to select the subsystem clock (WUTMMCK0 bit = 0).**

**3. When WUTMMCK0 is set to 1, the low-speed on-chip oscillator clock oscillates.**

(The **Cautions** are given on the next page.)

**Cautions 4. When WUTMMCK0 is set to 1, only the constant-period interrupt function of real-time clock 2 can be used. The year, month, day of the week, day, hour, minute, and second counters and the 1 Hz output function of real-time clock 2 cannot be used.**
**The interval of the constant-period interrupt is calculated by constant period (value selected by using the RTCC0 register) × $f_{SUB}/f_{IL}$.**

**5. The subsystem clock and low-speed on-chip oscillator clock can only be switched by using the WUTMMCK0 bit if real-time clock 2, 12-bit interval timer, and LCD controller/driver are all stopped.**

**6. The count of year, month, week, day, hour, minutes and second can only be performed when a subsystem clock ($f_{SUB}$ = 32.768 kHz) is selected as the operation clock of real-time clock 2. When the low-speed oscillation clock ($f_{IL}$ = 15 kHz) is selected, only the constant-period interrupt function is available.**
**However, the constant-period interrupt interval when $f_{IL}$ is selected will be calculated with the constant-period (the value selected with RTCC0 register) × $1/f_{IL}$.**

### 8.3.4 Power-on-reset status register (PORSR)

The PORSR register is used to check the occurrence of a power-on reset.

Writing “1” to bit 0 (PORF) of the PORSR register is valid, and writing “0” is ignored.

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 an 8-bit memory manipulation instruction.

Power-on reset signal generation clears this register to 00H.

**Cautions 1. The PORSR register is reset only by a power-on reset; it retains the value when a reset caused by another factor occurs.**

**2. If the PORF bit is set to 1, it guarantees that no power-on reset has occurred, but it does not guarantee that the RAM value is retained.**

**Figure 8-5. Format of Power-on-Reset Status Register (PORSR)**

Address: F00F9H After reset: 00H 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. |

### 8.3.5 Real-time clock control register 0 (RTCC0)

The RTCC0 register is an 8-bit register that is used to start or stop the real-time clock 2 operation, control the RTC1HZ pin, set the 12- or 24-hour system, and set the constant-period interrupt function.

RTCC0 can be set by a 1-bit or 8-bit memory manipulation instruction.

Internal reset generated by the power-on-reset circuit clears this register to 00H.

**Figure 8-6. Format of Real-time Clock Control Register 0 (RTCC0) (1/2)**

Address: FFF9DH After reset: 00H R/W

| Symbol | <7> | <6> | <5> | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| RTCC0 | RTCE | RCLOSEL | RCLOE1 | 0 | AMPM | CT2 | CT1 | CT0 |

| RTCE[Note 1] | Real-time clock 2 operation control |
|---|---|
| 0 | Stops counter operation. |
| 1 | Starts counter operation. |

| RCLOSEL | RTC1HZ pin output mode control |
|---|---|
| 0 | Normal 1 Hz output mode |
| 1 | High accuracy 1 Hz output mode |

| RCLOE1[Note 2] | RTC1HZ pin output control |
|---|---|
| 0 | Disables output of the RTC1HZ pin (1 Hz) |
| 1 | Enables output of the RTC1HZ pin (1 Hz) |
| Output of 1 Hz is not output because the clock counter does not operate when RTCE = 0. | |

**Notes 1.** When shifting to STOP mode immediately after setting RTCE to 1, use the procedure shown in **Figure 8-20 Procedure for Shifting to HALT/STOP Mode After Setting RTCE = 1**.

**2.** When the RCLOE1 bit is set while the clock counter operates (RTCE = 1), a glitch may be output to the 1 Hz output pin (RTC1HZ).

**Cautions 1. The high accuracy 1 Hz output is available only when 24 MHz is selected for the high-speed on-chip oscillator ($f_{IH}$) and the high-speed on-chip oscillator is running (HIOSTOP = 0). There is no need to select $f_{IH}$ for CPU clock. Also, Using clock error correction when high accuracy 1 Hz output is used.**

**2. Be sure to set bit 4 to "0".**

**Figure 8-6. Format of Real-time Clock Control Register 0 (RTCC0) (2/2)**

Address: FFF9DH After reset: 00H R/W

| Symbol | <7> | <6> | <5> | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| RTCC0 | RTCE | RCLOSEL | RCLOE1 | 0 | AMPM | CT2 | CT1 | CT0 |

**Table 8-2. Relation Between RTCE, RCLOSEL, and RCLOE1 Settings and Status**

| Register Settings | | | Status | |
|---|---|---|---|---|
| RTCE | RCLOSEL | RCLOE1 | Real-time clock 2 | RTC1HZ pin output |
| 0 | × | × | Counting stopped | No output |
| 1 | 0 | 0 | Count operation | No output |
| | | 1 | Count operation | Normal 1 Hz output |
| | 1 | 0 | Count operation | No output |
| | | 1 | Count operation | High accuracy 1 Hz output |

| AMPM | 12-/24-hour system select |
|---|---|
| 0 | 12-hour system (a.m. and p.m. are displayed.) |
| 1 | 24-hour system |
| When changing the value of the AMPM bit while the clock counter operates (RTCE = 1), set RWAIT (bit 0 of RTCC1) and then set the hour counter (HOUR) again.<br>When the AMPM value is 0, the 12-hour system is displayed. When the value is 1, the 24-hour system is displayed. Table 8-3 shows the displayed time digits. | |

| CT2 | CT1 | CT0 | Constant-period interrupt (INTRTC) selection |
|---|---|---|---|
| 0 | 0 | 0 | Does not use constant-period interrupt function. |
| 0 | 0 | 1 | Once per 0.5 s (synchronized with second count up) |
| 0 | 1 | 0 | Once per 1 s (same time as second count up) |
| 0 | 1 | 1 | Once per 1 m (second 00 of every minute) |
| 1 | 0 | 0 | Once per 1 hour (minute 00 and second 00 of every hour) |
| 1 | 0 | 1 | Once per 1 day (hour 00, minute 00, and second 00 of every day) |
| 1 | 1 | × | Once per 1 month (Day 1, hour 00 a.m., minute 00, and second 00 of every month) |
| When changing the values of the CT2 to CT0 bits while the counter operates (RTCE = 1), rewrite the values of the CT2 to CT0 bits after disabling interrupt servicing INTRTC by using the interrupt mask flag register. Furthermore, after rewriting the values of the CT2 to CT0 bits, enable interrupt servicing after clearing the RIFG and RTCIF flags. | | | |

**Caution Be sure to set bit 4 to "0".**

**Remark** ×: don't care

### 8.3.6 Real-time clock control register 1 (RTCC1)

The RTCC1 register is an 8-bit register that is used to control the alarm interrupt function and the wait time of the counter.

RTCC1 can be set by a 1-bit or 8-bit memory manipulation instruction.

Internal reset generated by the power-on-reset circuit clears this register to 00H.

**Figure 8-7. Format of Real-time Clock Control Register 1 (RTCC1) (1/3)**

Address: FFF9EH After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | 2 | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| RTCC1 | WALE | WALIE | RITE | WAFG | RIFG | 0 | RWST | RWAIT |

| WALE | Alarm operation control |
|---|---|
| 0 | Match operation is invalid. |
| 1 | Match operation is valid. |
| When setting a value to the WALE bit while the counter operates (RTCE = 1) and WALIE = 1, rewrite the WALE bit after disabling interrupt servicing INTRTC by using the interrupt mask flag register. Furthermore, clear the WAFG and RTCIF flags after rewriting the WALE bit. When setting each alarm register (WALIE flag of real-time clock control register 1 (RTCC1), the alarm minute register (ALARMWM), the alarm hour register (ALARMWH), and the alarm week register (ALARMWW)), set match operation to be invalid (“0”) for the WALE bit. | |

| WALIE | Control of alarm interrupt (INTRTC) function operation |
|---|---|
| 0 | Does not generate interrupt on matching of alarm. |
| 1 | Generates interrupt on matching of alarm. |

**Caution If writing is performed to RTCC1 with a 1-bit manipulation instruction, the RIFG and WAFG flags may be cleared. Therefore, to perform writing to RTCC1, be sure to use an 8-bit manipulation instruction.**

**To prevent the RIFG and WAFG flags from being cleared during writing, set 1 (writing disabled) to the corresponding bit. If the RIFG and WAFG flags are not used and the value may be changed, RTCC1 may be written by using a 1-bit manipulation instruction.**

**Figure 8-7. Format of Real-time Clock Control Register 1 (RTCC1) (2/3)**

Address: FFF9EH After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | 2 | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| RTCC1 | WALE | WALIE | RITE | WAFG | RIFG | 0 | RWST | RWAIT |

| RITE | Control of correction timing signal interrupt (INTRTIT) function operation |
|---|---|
| 0 | Does not generate interrupt of correction timing signal. |
| 1 | Generates interrupt of correction timing signal. |

| WAFG | Alarm detection status flag |
|---|---|
| 0 | Alarm mismatch |
| 1 | Detection of matching of alarm |
| This is a status flag that indicates detection of matching with the alarm. It is valid only when WALE = 1 and is set to “1” one clock (32.768 kHz) after matching of the alarm is detected.<br>This flag is cleared when “0” is written to it. Writing “1” to it is invalid. | |

| RIFG | Constant-period interrupt status flag |
|---|---|
| 0 | Constant-period interrupt is not generated. |
| 1 | Constant-period interrupt is generated. |
| This flag indicates the status of generation of the constant-period interrupt.<br>When the constant-period interrupt is generated, it is set to “1”.<br>This flag is cleared when “0” is written to it. Writing 1 to it is invalid. | |

**Caution If writing is performed to RTCC1 with a 1-bit manipulation instruction, the RIFG and WAFG flags may be cleared. Therefore, to perform writing to RTCC1, be sure to use an 8-bit manipulation instruction.**

**To prevent the RIFG and WAFG flags from being cleared during writing, set 1 (writing disabled) to the corresponding bit. If the RIFG and WAFG flags are not used and the value may be changed, RTCC1 may be written by using a 1-bit manipulation instruction.**

**Figure 8-7. Format of Real-time Clock Control Register 1 (RTCC1) (3/3)**

Address: FFF9EH After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | 2 | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| RTCC1 | WALE | WALIE | RITE | WAFG | RIFG | 0 | RWST | RWAIT |

| RWST | Wait status flag of real-time clock 2 |
|---|---|
| 0 | Counter is operating. |
| 1 | Mode to read or write counter value. |
| This status flag indicates whether the setting of the RWAIT bit is valid.<br>Before reading or writing the counter value, confirm that the value of this flag is 1.<br>Even if the RWAIT bit is set to 0, the RWST bit is not set to 0 while writing to the counter. After writing is completed, the RWST bit is set to 0. | |

| RWAIT | Wait control of real-time clock 2 |
|---|---|
| 0 | Sets counter operation. |
| 1 | Stops SEC to YEAR counters. Mode to read or write counter value. |
| This bit controls the operation of the counter.<br>Be sure to write "1" to it to read or write the counter value.<br>As the counter (16-bit) is continuing to run, complete reading or writing within one second and turn back to 0.<br>When RWAIT = 1, it takes up to 1 clock of $f_{RTC}$ until the counter value can be read or written (RWST = 1)Notes1, 2.<br>When the internal counter (16-bit) overflowed while RWAIT = 1, it keeps the event of overflow until RWAIT = 0, then counts up.<br>However, when it wrote a value to second count register, it will not keep the overflow event. | |

<R>

**Notes 1.** When the RWAIT bit is set to 1 within one cycle of $f_{RTC}$ clock after setting the RTCE bit to 1, the RWST bit being set to 1 may take up to two cycles of the operating clock ($f_{RTC}$).

<R> **2.** When the RWAIT bit is set to 1 within one cycle of $f_{RTC}$ clock after release from the standby mode (HALT mode, STOP mode, or SNOOZE mode), the RWST bit being set to 1 may take up to two cycles of the operating clock ($f_{RTC}$).

**Caution If writing is performed to RTCC1 with a 1-bit manipulation instruction, the RIFG and WAFG flags may be cleared. Therefore, to perform writing to RTCC1, be sure to use an 8-bit manipulation instruction.**
**To prevent the RIFG and WAFG flags from being cleared during writing, set 1 (writing disabled) to the corresponding bit. If the RIFG and WAFG flags are not used and the value may be changed, RTCC1 may be written by using a 1-bit manipulation instruction.**

**Remarks 1.** Constant-period interrupts and alarm match interrupts use the same interrupt source (INTRTC). When using these two types of interrupts at the same time, which interrupt occurred can be judged by checking the constant-period interrupt status flag (RIFG) and the alarm detection status flag (WAFG) upon INTRTC occurrence.

**2.** The internal counter (16 bits) is cleared when the second count register (SEC) is written.

### 8.3.7 Second count register (SEC)

The SEC register is an 8-bit register that takes a value of 0 to 59 (decimal) and indicates the count value of seconds.

It is a decimal counter that counts up when the counter (16-bit) overflows.

When data is written to this register, it is written to a buffer and then to the counter up to 2 clocks of $f_{RTC}$ later.

Set a decimal value of 00 to 59 to this register in BCD code.

The SEC register can be set by an 8-bit memory manipulation instruction.

Reset signal generation not clears this register to default value.

**Figure 8-8. Format of Second Count Register (SEC)**

Address: FFF92H After reset: Undefined R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| SEC | 0 | SEC40 | SEC20 | SEC10 | SEC8 | SEC4 | SEC2 | SEC1 |

**Caution When reading or writing to SEC while the clock counter operates (RTCE = 1), be sure to use the flows shown in 8.4.3 Reading real-time clock 2 counter and 8.4.4 Writing to real-time clock 2 counter.**

**Remark** The internal counter (16 bits) is cleared when the second count register (SEC) is written.

### 8.3.8 Minute count register (MIN)

The MIN register is an 8-bit register that takes a value of 0 to 59 (decimal) and indicates the count value of minutes.

It is a decimal counter that counts up when the second counter overflows.

When data is written to this register, it is written to a buffer and then to the counter up to 2 clocks of $f_{RTC}$ later. Even if the second count register overflows while this register is being written, this register ignores the overflow and is set to the value written. Set a decimal value of 00 to 59 to this register in BCD code.

The MIN register can be set by an 8-bit memory manipulation instruction.

Reset signal generation not clears this register to default value.

**Figure 8-9. Format of Minute Count Register (MIN)**

Address: FFF93H After reset: Undefined R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| MIN | 0 | MIN40 | MIN20 | MIN10 | MIN8 | MIN4 | MIN2 | MIN1 |

**Caution When reading or writing to MIN while the clock counter operates (RTCE = 1), be sure to use the flows shown in 8.4.3 Reading real-time clock 2 counter and 8.4.4 Writing to real-time clock 2 counter.**

### 8.3.9 Hour count register (HOUR)

The HOUR register is an 8-bit register that takes a value of 00 to 23 or 01 to 12 and 21 to 32 (decimal) and indicates the count value of hours.

It is a decimal counter that counts up when the minute counter overflows.

When data is written to this register, it is written to a buffer and then to the counter up to 2 clocks of $f_{RTC}$ later. Even if the minute count register overflows while this register is being written, this register ignores the overflow and is set to the value written. Specify a decimal value of 00 to 23, 01 to 12, or 21 to 32 by using BCD code according to the time system specified using bit 3 (AMPM) of real-time clock control register 0 (RTCC0).

If the AMPM bit value is changed, the values of the HOUR register change according to the specified time system.

The HOUR register can be set by an 8-bit memory manipulation instruction.

Reset signal generation not clears this register to default value.

**Figure 8-10. Format of Hour Count Register (HOUR)**

Address: FFF94H After reset: Undefined R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| HOUR | 0 | 0 | HOUR20 | HOUR10 | HOUR8 | HOUR4 | HOUR2 | HOUR1 |

**Cautions 1. Bit 5 (HOUR20) of the HOUR register indicates AM(0)/PM(1) if AMPM = 0 (if the 12-hour system is selected).**

**2. When reading or writing to HOUR while the clock counter operates (RTCE = 1), be sure to use the flows shown in 8.4.3 Reading real-time clock 2 counter and 8.4.4 Writing to real-time clock 2 counter.**

Table 8-3 shows the relationship between the setting value of the AMPM bit, the hour count register (HOUR) value, and time.

**Table 8-3. Displayed Time Digits**

| 24-Hour Display (AMPM = 1) | | 12-Hour Display (AMPM = 0) | |
|---|---|---|---|
| Time | HOUR Register | Time | HOUR Register |
| 0 | 00H | 12 a.m. | 12H |
| 1 | 01H | 1 a.m. | 01H |
| 2 | 02H | 2 a.m. | 02H |
| 3 | 03H | 3 a.m. | 03H |
| 4 | 04H | 4 a.m. | 04H |
| 5 | 05H | 5 a.m. | 05H |
| 6 | 06H | 6 a.m. | 06H |
| 7 | 07H | 7 a.m. | 07H |
| 8 | 08H | 8 a.m. | 08H |
| 9 | 09H | 9 a.m. | 09H |
| 10 | 10H | 10 a.m. | 10H |
| 11 | 11H | 11 a.m. | 11H |
| 12 | 12H | 12 p.m. | 32H |
| 13 | 13H | 1 p.m. | 21H |
| 14 | 14H | 2 p.m. | 22H |
| 15 | 15H | 3 p.m. | 23H |
| 16 | 16H | 4 p.m. | 24H |
| 17 | 17H | 5 p.m. | 25H |
| 18 | 18H | 6 p.m. | 26H |
| 19 | 19H | 7 p.m. | 27H |
| 20 | 20H | 8 p.m. | 28H |
| 21 | 21H | 9 p.m. | 29H |
| 22 | 22H | 10 p.m. | 30H |
| 23 | 23H | 11 p.m. | 31H |

The HOUR register value is set to 12-hour display when the AMPM bit is “0” and to 24-hour display when the AMPM bit is “1”.

In 12-hour display, the fifth bit of the HOUR register displays 0 for AM and 1 for PM.

### 8.3.10 Date count register (DAY)

The DAY register is an 8-bit register that takes a value of 1 to 31 (decimal) and indicates the count value of days.

It is a decimal counter that count ups when the hour counter overflows.

This counter counts as follows.

[DAY count values]

- 01 to 31 (January, March, May, July, August, October, December)
- 01 to 30 (April, June, September, November)
- 01 to 29 (February, leap year)
- 01 to 28 (February, normal year)

When data is written to this register, it is written to a buffer and then to the counter up to 2 clocks of $f_{RTC}$ later. Even if the hour count register overflows while this register is being written, this register ignores the overflow and is set to the value written. Set a decimal value of 01 to 31 to this register in BCD code.

The DAY register can be set by an 8-bit memory manipulation instruction.

Reset signal generation not clears this register to default value.

**Figure 8-11. Format of Day-of-week Count Register (DAY)**

Address: FFF96H After reset: Undefined R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DAY | 0 | 0 | DAY20 | DAY10 | DAY8 | DAY4 | DAY2 | DAY1 |

**Caution When reading or writing to DAY while the clock counter operates (RTCE = 1), be sure to use the flows shown in 8.4.3 Reading real-time clock 2 counter and 8.4.4 Writing to real-time clock 2 counter.**

### 8.3.11 Day-of-week count register (WEEK)

The WEEK register is an 8-bit register that takes a value of 0 to 6 (decimal) and indicates the count value of weekdays.

It is a decimal counter that counts up when a carry to the date counter occurs.

When data is written to this register, it is written to a buffer and then to the counter up to 2 clocks of $f_{RTC}$ later.

Set a decimal value of 00 to 06 to this register in BCD code.

The WEEK register can be set by an 8-bit memory manipulation instruction.

Reset signal generation not clears this register to default value.

**Figure 8-12. Format of Date Count Register (WEEK)**

Address: FFF95H After reset: Undefined R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| WEEK | 0 | 0 | 0 | 0 | 0 | WEEK4 | WEEK2 | WEEK1 |

**Cautions 1. The value corresponding to the month count register (MONTH) or the day count register (DAY) is not stored in the week count register (WEEK) automatically. After reset release, set the week count register as follow.**

| Day | WEEK |
|---|---|
| Sunday | 00H |
| Monday | 01H |
| Tuesday | 02H |
| Wednesday | 03H |
| Thursday | 04H |
| Friday | 05H |
| Saturday | 06H |

**2. When reading or writing to WEEK while the clock counter operates (RTCE = 1), be sure to use the flows shown in 8.4.3 Reading real-time clock 2 counter and 8.4.4 Writing to real-time clock 2 counter.**

### 8.3.12 Month count register (MONTH)

The MONTH register is an 8-bit register that takes a value of 1 to 12 (decimal) and indicates the count value of months.

It is a decimal counter that count ups when the date counter overflows.

When data is written to this register, it is written to a buffer and then to the counter up to 2 clocks of $f_{RTC}$ later. Even if the day count register overflows while this register is being written, this register ignores the overflow and is set to the value written. Set a decimal value of 01 to 12 to this register in BCD code.

The MONTH register can be set by an 8-bit memory manipulation instruction.

Reset signal generation not clears this register to default value.

**Figure 8-13. Format of Month Count Register (MONTH)**

Address: FFF97H After reset: Undefined R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| MONTH | 0 | 0 | 0 | MONTH10 | MONTH8 | MONTH4 | MONTH2 | MONTH1 |

**Caution When reading or writing to MONTH while the clock counter operates (RTCE = 1), be sure to use the flows shown in 8.4.3 Reading real-time clock 2 counter and 8.4.4 Writing to real-time clock 2 counter.**

### 8.3.13 Year count register (YEAR)

The YEAR register is an 8-bit register that takes a value of 0 to 99 (decimal) and indicates the count value of years.

It is a decimal counter that counts up when the month count register (MONTH) overflows.

Values 00, 04, 08, …, 92, and 96 indicate a leap year.

When data is written to this register, it is written to a buffer and then to the counter up to 2 clocks of $f_{RTC}$ later. Even if the MONTH register overflows while this register is being written, this register ignores the overflow and is set to the value written. Set a decimal value of 00 to 99 to this register in BCD code.

The YEAR register can be set by an 8-bit memory manipulation instruction.

Reset signal generation not clears this register to default value.

**Figure 8-14. Format of Year Count Register (YEAR)**

Address: FFF98H After reset: Undefined R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| YEAR | YEAR80 | YEAR40 | YEAR20 | YEAR10 | YEAR8 | YEAR4 | YEAR2 | YEAR1 |

**Caution When reading or writing to YEAR while the clock counter operates (RTCE = 1), be sure to use the flows shown in 8.4.3 Reading real-time clock 2 counter and 8.4.4 Writing to real-time clock 2 counter.**

### 8.3.14 Clock error correction register (SUBCUD)

This register is used to correct the clock with a minimum resolution and accuracy of 0.96 ppm when it is slow or fast by changing the counter value every second.

F8 to F0 of SUBCUD is a 9 bit fixed-point (2's complement) register. For details, see **Table 8-5 Clock Error Correction Values**.

The SUBCUD register can be set by an 16-bit memory manipulation instruction.

Internal reset generated by the power-on-reset circuit clears this register to 0020H.

**Figure 8-15. Format of Clock Error Correction Register (SUBCUD)**

Address: F0310H After reset: 0020H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SUBCUD | F15 | 0 | 0 | 0 | 0 | 0 | 0 | F8 | F7 | F6 | F5 | F4 | F3 | F2 | F1 | F0 |

| F15 | Clock error correction enable |
|---|---|
| 0 | Stops clock error correction. |
| 1 | Enables clock error correction. |

The range of value that can be corrected by using the clock error correction register (SUBCUD) is shown in Table 8-4.

**Table 8-4. Correctable Range of Crystal Resonator Oscillation Frequency Deviation**

| Item | Value |
|---|---|
| Correctable range | –274.6 ppm to +212.6 ppm |
| Maximum quantization error | ±0.48 ppm |
| Minimum resolution | 0.96 ppm |

**Table 8-5. Clock Error Correction Values**

| SUBCUD | | | | | | | | | | Target Correction Values |
|---|---|---|---|---|---|---|---|---|---|---|
| F15 | F8 | F7 | F6 | F5 | F4 | F3 | F2 | F1 | F0 | |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | –274.6 ppm |
| | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | –273.7 ppm |
| | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | –272.7 ppm |
| | • | • | • | • | • | • | • | • | • | • |
| | • | • | • | • | • | • | • | • | • | • |
| | • | • | • | • | • | • | • | • | • | • |
| | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | –33.3 ppm |
| | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | –32.4 ppm |
| | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | –31.4 ppm |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | –30.5 ppm |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | –29.6 ppm |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | –28.6 ppm |
| | • | • | • | • | • | • | • | • | • | • |
| | • | • | • | • | • | • | • | • | • | • |
| | • | • | • | • | • | • | • | • | • | • |
| | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | –0.95 ppm |
| | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 ppm |
| | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0.95 ppm |
| | • | • | • | • | • | • | • | • | • | • |
| | • | • | • | • | • | • | • | • | • | • |
| | • | • | • | • | • | • | • | • | • | • |
| | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 210.7 ppm |
| | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 211.7 ppm |
| | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 212.6 ppm |
| 0 | × | × | × | × | × | × | × | × | × | Clock error correction stopped |

The F8 to F0 value of the SUBCUD register is calculated from the target correction value by using the following expression.

$$\text{SUBCUD[8:0]} = \left( \frac{\text{Target correction value [ppm]} \times 2^{15}}{10^{6}} \right)_{\substack{\text{2's complement} \\ \text{(9 bit fixed-point} \\ \text{format)}}} + 0001.00000\text{B}$$

**Caution The target correction value is the oscillation frequency deviation (unit: [ppm]) of the crystal resonator. For calculating the correction value, see 8.4.8 Example of watch error correction of real-time clock 2.**

**Examples 1.** When target correction value = 18.3 [ppm]

$\text{SUBCUD[8:0]} = (18.3 \times 2^{15} / 10^{6})_{\text{2's complement (9 bit fixed-point format)}} + 0001.00000\text{B}$

$= (0.59375)_{\text{2's complement (9 bit fixed-point format)}} + 0001.00000\text{B}$

$= 0000.10011\text{B} + 0001.00000\text{B}$

$= 0001.10011\text{B}$

**Examples 2.** When target correction value = –18.3 [ppm]

SUBCUD[8:0] = $(-18.3 \times 2^{15} / 10^{6})$ 2's complement (9 bit fixed-point format) + 0001.00000B
= $(-0.59965)$ 2's complement (9 bit fixed-point format) + 0001.00000B
= 1111.01101B + 0001.00000B
= 0000.01101B

### 8.3.15 Alarm minute register (ALARMWM)

This register is used to set the minute of an alarm.

The ALARMWM register can be set by an 8-bit memory manipulation instruction.

Reset signal generation not clears this register to default value.

**Figure 8-16. Format of Alarm minute register (ALARMWM)**

Address: FFF9AH After reset: Undefined R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ALARMWM | 0 | WM40 | WM20 | WM10 | WM8 | WM4 | WM2 | WM1 |

**Caution Set a decimal value of 00 to 59 to this register in BCD code. If a value outside the range is set, the alarm is not detected.**

### 8.3.16 Alarm hour register (ALARMWH)

This register is used to set the hour of an alarm.

The ALARMWH register can be set by an 8-bit memory manipulation instruction.

Reset signal generation not clears this register to default value.

**Figure 8-17. Format of Alarm hour register (ALARMWH)**

Address: FFF9BH After reset: Undefined R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ALARMWH | 0 | 0 | WH20 | WH10 | WH8 | WH4 | WH2 | WH1 |

**Cautions 1. Set a decimal value of 00 to 23 or 01 to 12 and 21 to 32 to this register in BCD code. If a value outside the range is set, the alarm is not detected.**

**2. Bit 5 (WH20) of the ALARMWH register indicates AM(0)/PM(1) if AMPM = 0 (if the 12-hour system is selected).**

### 8.3.17 Alarm day-of-week register (ALARMWW)

This register is used to set the day of the week of an alarm.

The ALARMWW register can be set by an 8-bit memory manipulation instruction.

Reset signal generation not clears this register to default value.

**Figure 8-18. Format of Alarm day-of-week Register (ALARMWW)**

Address: FFF9CH After reset: Undefined R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ALARMWW | 0 | WW6 | WW5 | WW4 | WW3 | WW2 | WW1 | WW0 |

Table 8-6 shows an example of setting the alarm.

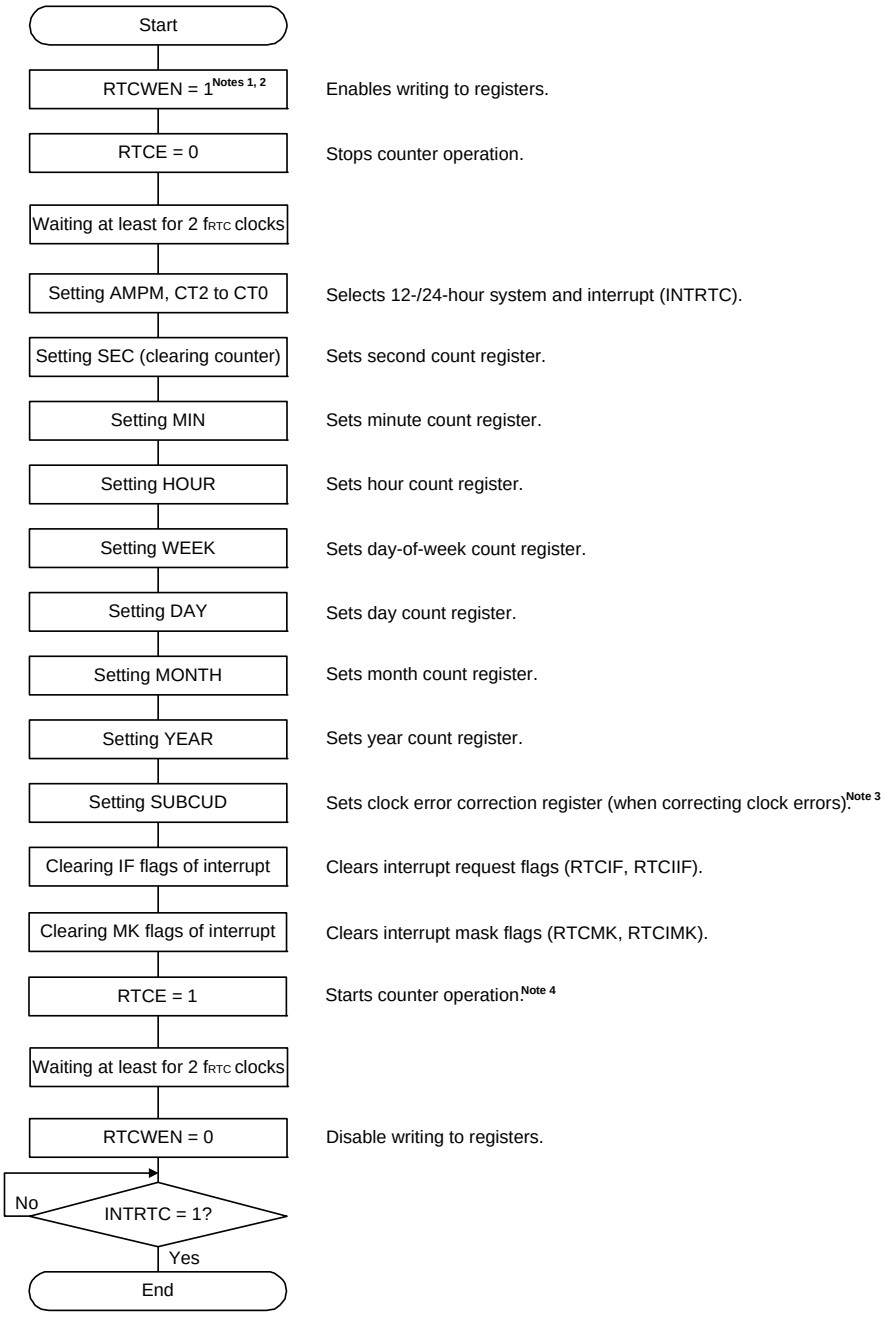
**Table 8-6. Setting Alarm**

| Time of Alarm | Day of the Week | | | | | | | 12-Hour Display | | | | 24-Hour Display | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Sun. | Mon. | Tue. | Wed. | Thu. | Fri. | Sat. | Hour 10 | Hour 1 | Min. 10 | Min. 1 | Hour 10 | Hour 1 | Min. 10 | Min. 1 |
| | WW0 | WW1 | WW2 | WW3 | WW4 | WW5 | WW6 | | | | | | | | |
| Every day, 0:00 a.m. | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 0 | 0 | 0 | 0 | 0 | 0 |
| Every day, 1:30 a.m. | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 3 | 0 | 0 | 1 | 3 | 0 |
| Every day, 11:59 a.m. | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 5 | 9 | 1 | 1 | 5 | 9 |
| Monday through Friday, 0:00 p.m. | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 3 | 2 | 0 | 0 | 1 | 2 | 0 | 0 |
| Sunday, 1:30 p.m. | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 1 | 3 | 0 | 1 | 3 | 3 | 0 |
| Monday, Wednesday, Friday, 11:59 p.m. | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 3 | 1 | 5 | 9 | 2 | 3 | 5 | 9 |

## 8.4 Real-time Clock 2 Operation

### 8.4.1 Starting operation of real-time clock 2

**Figure 8-19. Procedure for Starting Operation of Real-time Clock 2**




**Notes 1.** Set RTCWEN to 0, except when accessing the RTC register, in order to prevent error when writing to the clock counter.

**2.** First set the RTCWEN bit to 1, while oscillation of the count clock ($f_{RTC}$) is stable.

**3.** Set up the SUBCUD register only if the watch error must be corrected. For details about how to calculate the correction value, see **8.4.8 Example of watch error correction of real-time clock 2**.

**4.** Confirm the procedure described in **8.4.2 Shifting to HALT/STOP mode after starting operation** when shifting to HALT/STOP mode without waiting for INTRTC = 1 after RTCE = 1.

### 8.4.2 Shifting to HALT/STOP mode after starting operation

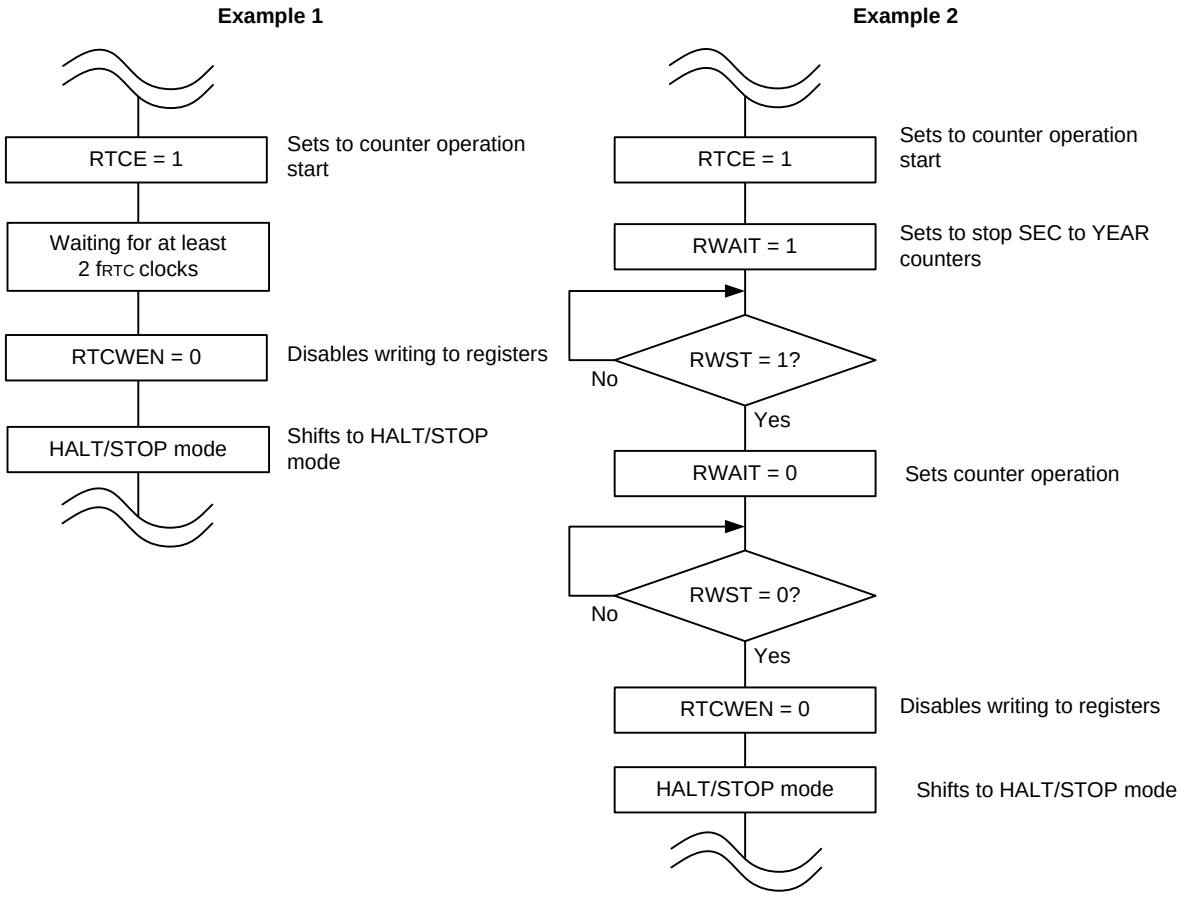
Perform one of the following processing when shifting to STOP mode immediately after setting the RTCE bit to 1.

However, after setting the RTCE bit to 1, this processing is not required when shifting to STOP mode after the first INTRTC interrupt has occurred.

(1) Shifting to HALT/STOP mode when at least two input clocks of the count clock ($f_{RTC}$) have elapsed after setting the RTCE bit to 1 (see **Example 1** of **Figure 8-20**).

(2) Checking by polling the RWST bit to become 1, after setting the RTCE bit to 1 and then setting the RWAIT bit to 1. Afterward, setting the RWAIT bit to 0 and shifting to HALT/STOP mode after checking again by polling that the RWST bit has become 0 (see **Example 2** of **Figure 8-20**).

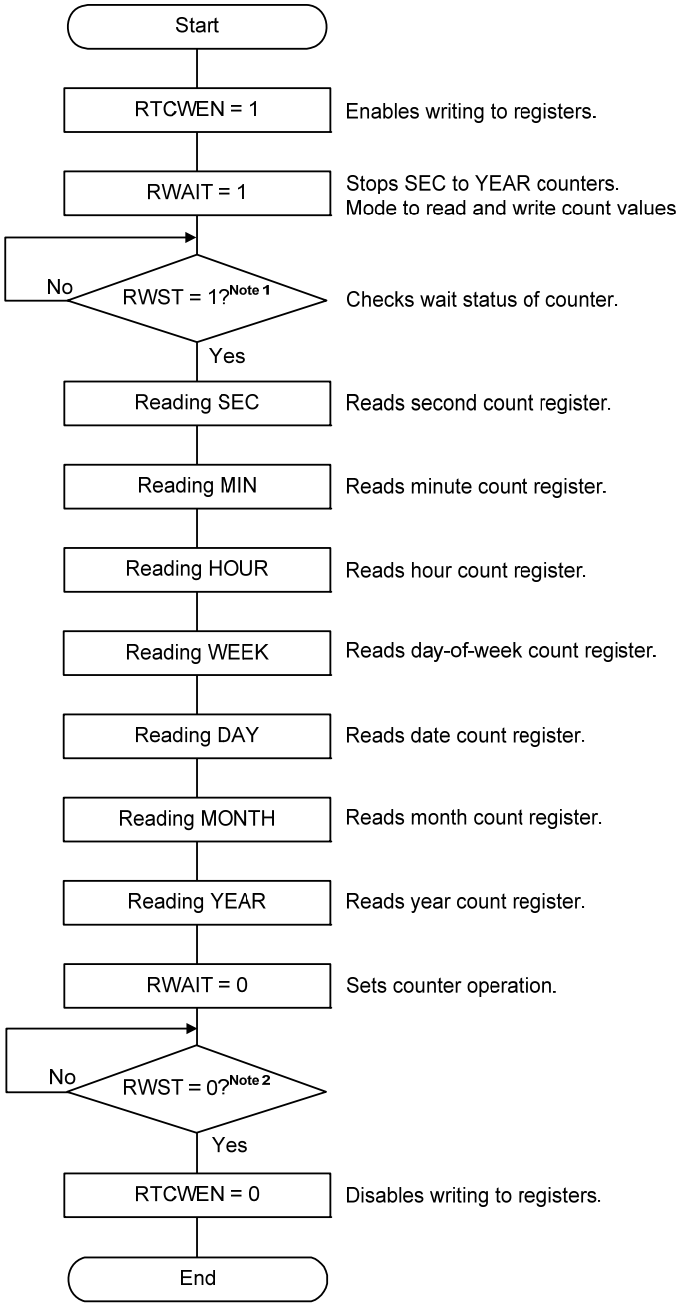
**Figure 8-20. Procedure for Shifting to HALT/STOP Mode After Setting RTCE = 1**



### 8.4.3 Reading real-time clock 2 counter

During counter operation (RTCE = 1), read to the counter after setting to 1 to RWAIT first.
Set RWAIT to 0 after completion of reading the counter.

**Figure 8-21. Procedure for Reading Real-time Clock 2**




**Notes 1.** When the counter is stopped (RTCE = 0), RWST is not set to 1.

**2.** Be sure to confirm that RWST = 0 before setting STOP mode.

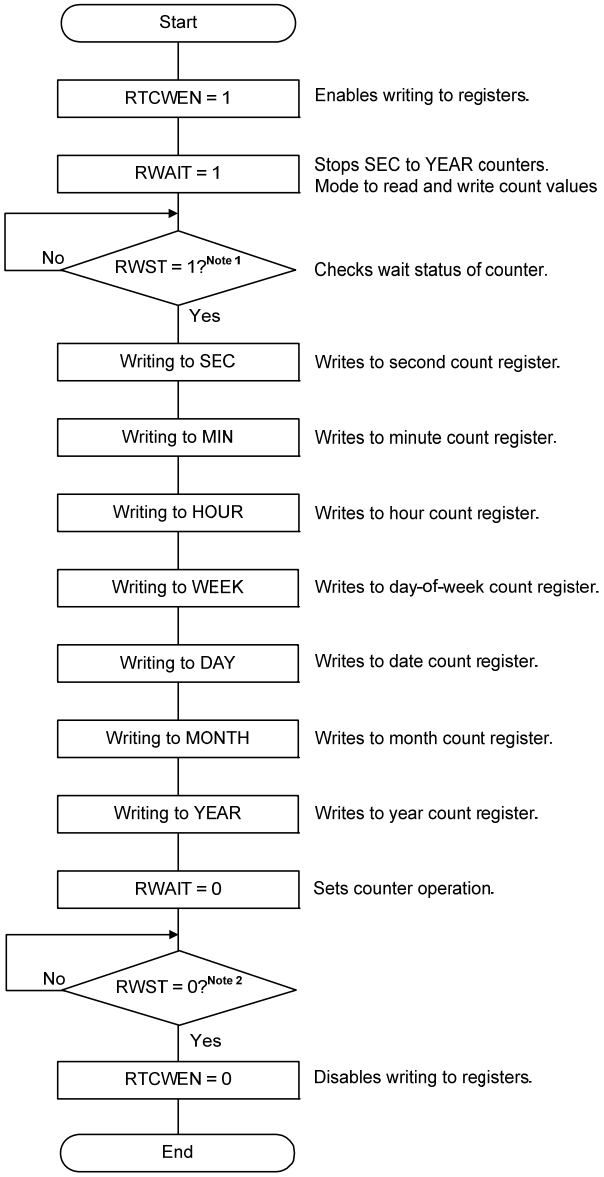
**Caution Complete the series of operations of setting the RWAIT bit to 1 to clearing the RWAIT bit to 0 within 1 second.**

**Remark** SEC, MIN, HOUR, WEEK, DAY, MONTH, and YEAR may be read in any sequence. All the registers do not have to be set and only some registers may be read.

### 8.4.4 Writing to real-time clock 2 counter

During counter operation (RTCE = 1), Write to the counter after setting to 1 to RWAIT first.
Set RWAIT to 0 after completion of writing the counter.

**Figure 8-22. Procedure for Writing Real-time Clock 2**




**Notes 1.** When the counter is stopped (RTCE = 0), RWST is not set to 1.

**2.** Be sure to confirm that RWST = 0 before setting STOP mode.

**Cautions 1. Complete the series of operations of setting the RWAIT bit to 1 to clearing the RWAIT bit to 0 within 1 second.**

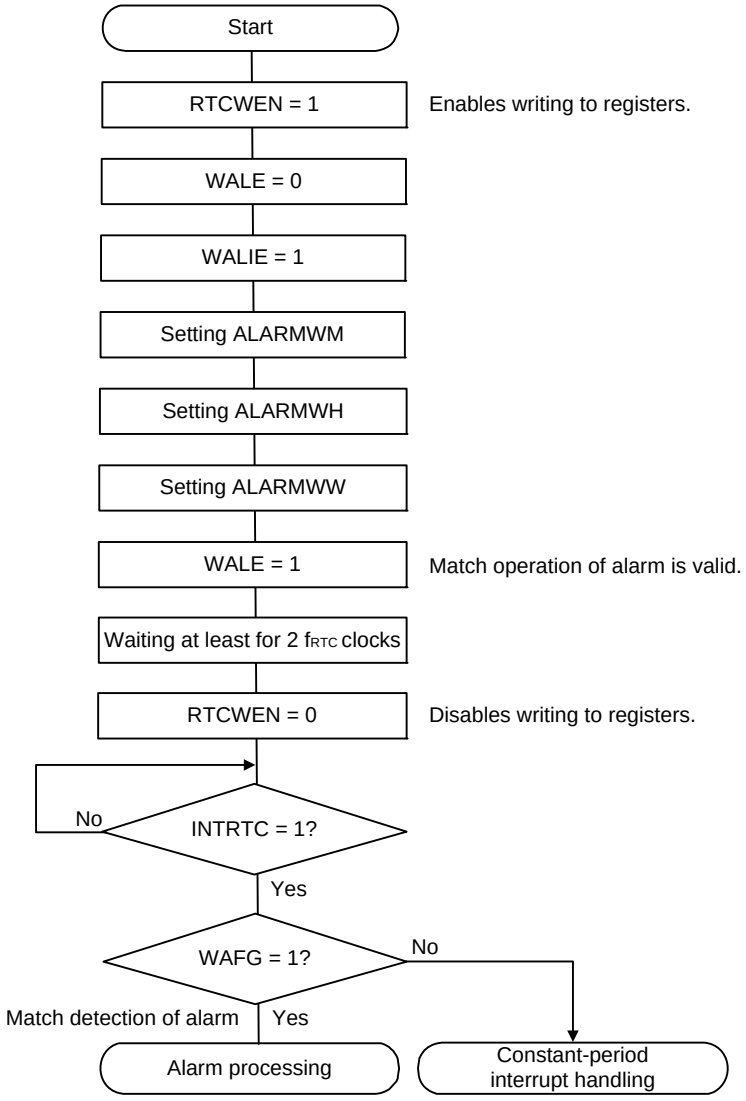
**2. When changing the values of the SEC, MIN, HOUR, WEEK, DAY, MONTH, and YEAR register while the counter operates (RTCE = 1), rewrite the values of the MIN register after disabling interrupt servicing INTRTC by using the interrupt mask flag register. Furthermore, clear the WAFG, RIFG and RTCIF flags after rewriting the MIN register.**

**Remark** SEC, MIN, HOUR, WEEK, DAY, MONTH, and YEAR may be read in any sequence. All the registers do not have to be set and only some registers may be written.

### 8.4.5 Setting alarm of real-time clock 2

Set the alarm time after setting 0 to WALE (alarm operation invalid.) first.

**Figure 8-23. Alarm Setting Procedure**

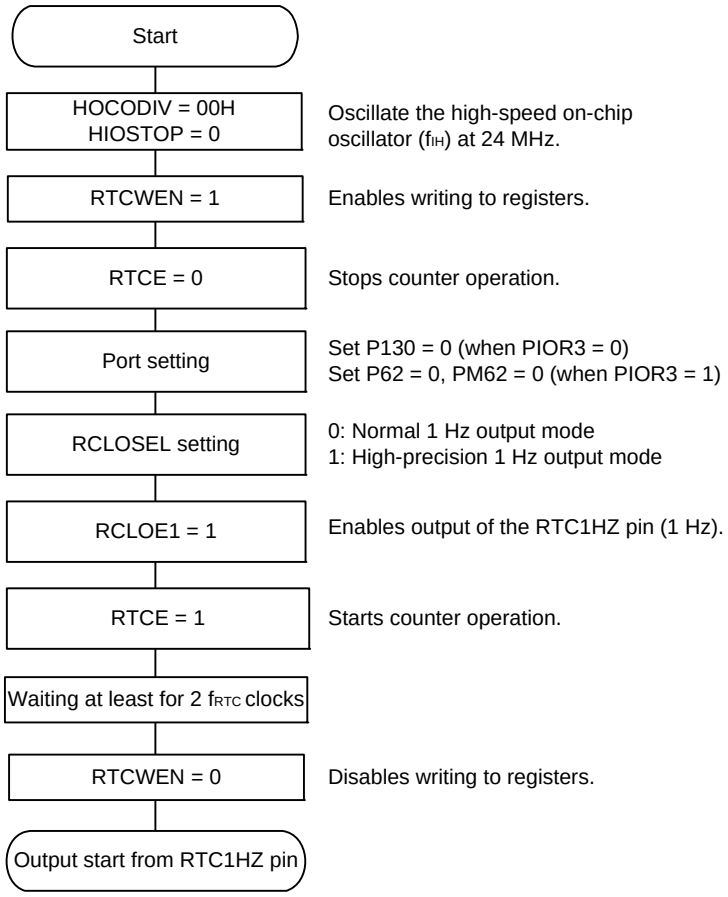



**Remarks 1.** ALARMWM, ALARMWH, and ALARMWW may be written in any sequence.

**2.** Constant-period interrupts and alarm match interrupts use the same interrupt source (INTRTC). When using these two types of interrupts at the same time, which interrupt occurred can be judged by checking the constant-period interrupt status flag (RIFG) and the alarm detection status flag (WAFG) upon INTRTC occurrence.

### 8.4.6 1 Hz output of real-time clock 2

**Figure 8-24. 1 Hz Output Setting Procedure**




**Caution When using a high-precision 1 Hz pin output, select 24 MHz for high-speed on-chip oscillator clock ($f_{IH}$) and operate the high-speed on-chip oscillator (HIOSTOP=0). There is no need to select it for CPU clock.**

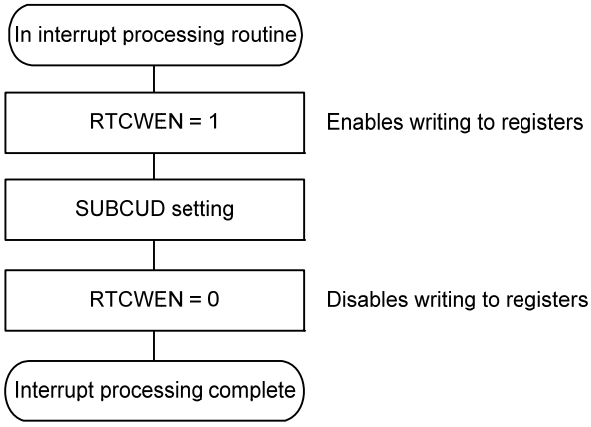
### 8.4.7 Clock error correction register setting procedure

Use either of the following procedures to set the clock error correction register (SUBCUD).

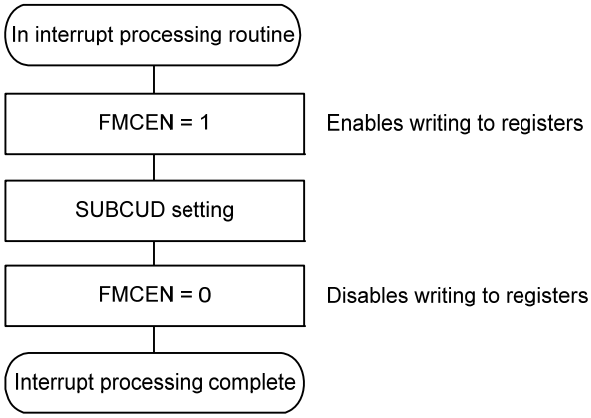
In order to prevent write error to the clock register, write privilege with (2) FMCEN is recommended for rewrite of the SUBCUD register.

RTC correction may not be successful if there is a conflict between the clock error correction register (SUBCUD) rewrite and correction timing. In order to prevent conflict between the correction timing and rewrite of the SUBCUD register, be sure to complete rewrite of the SUBCUD register before the next correction timing occurs (within approx. 0.5 seconds), which is calculated starting from the correction timing interrupt (INTRTIT) or periodic interrupt (INTRTC) that is synchronized with the correction timing.

(1) Set the clock error correction register after setting RTCWEN to 1 first. Then set RTCWEN to 0.




(2) Set the clock error correction register after setting FMCEN to 1 first. Then set FMCEN to 0.



### 8.4.8 Example of watch error correction of real-time clock 2

The clock can be corrected every second with a minimum resolution and accuracy of 0.96 ppm when it is slow or fast, by setting a value to the clock error correction register.

The following shows how to calculate the target correction value, and how to calculate the F8 to F0 values of the clock error correction register from the target correction value.

Calculating the target correction value 1

(When using output frequency of the RTC1HZ pin)

[Measuring the oscillation frequency]

Measure the oscillator frequency of each product[Note] by output of normal 1 Hz output from the RTC1HZ pin when the clock error correction register (SUBCUD) F15 is “0” (stop clock error correction).

**Note** See **8.4.6 1 Hz output of real-time clock 2** for the procedure of outputting about 1 Hz from the RTC1HZ pin.

[Calculating the target correction value]

(When the output frequency from the RTCCL pin is 0.9999817 Hz)

Oscillation frequency = 32768 × 0.9999817 ≈ 32767.40 Hz

Assume the target frequency to be 32768 Hz. Then the target correction value is calculated as follows.

$$\begin{aligned}\text{Target correction value} &= (\text{Oscillation frequency} - \text{Target frequency}) \div \text{Target frequency}\\ &= (32767.40 - 32768.00) \div 32768.00\\ &\approx -18.3 \text{ ppm}\end{aligned}$$

**Remarks 1.** The oscillation frequency is the input clock ($f_{RTC}$). It can be calculated from the output frequency of the RTC1HZ pin × 32768 when stops the watch error correction.

**2.** The target correction value is the oscillation frequency deviation (unit: [ppm]) of the crystal resonator.

**3.** The target frequency is the frequency resulting after watch error correction performed.

Calculating the target correction value 2

(When using subsystem clock frequency measurement circuit)

[Measuring the oscillation frequency]

The oscillation frequency[Note] of each product is measured by using subsystem clock frequency measurement circuit.

The oscillation frequency is calculated by using the following expression.

$$\text{Oscillation frequency} = \frac{f_{MX}\text{ frequency [Hz]} \times \text{operating trigger division ratio}}{(\text{Frequency measurement count registers H, L (FMCRH, FMCRL) value})_{\text{Decimal}}}\ [\text{Hz}]$$

**Note** See **9.4.1 Setting crystal oscillation frequency measurement circuit using reference clock** for the operating procedure of subsystem clock frequency measurement.

[Calculating the target correction value]

(When the frequency measurement count registers H, L value is 9999060D)

- High-speed system clock frequency ($f_{MX}$) = 10 MHz
- When FMDIV2 to FMDIV0 of the frequency measurement control register = 111B (operating trigger division ratio = $2^{15}$).

Then the oscillation frequency is calculated as follows.

$$\begin{aligned}\text{Oscillation frequency} &= f_{MX}\text{ frequency [Hz]} \times \text{operating trigger division ratio} \div \text{(FMCRH, FMCRL) value}\\ &= 10 \times 10^5 \times 2^{15} \div 9999060\text{D}\\ &= 32771.0804816\text{ Hz}\end{aligned}$$

Assume the target frequency to be 32768 Hz. Then the target correction value is calculated as follows.

$$\begin{aligned}\text{Target correction value} &= \text{Oscillation frequency} \div \text{Target frequency} - 1\\ &= 32771.0804846 \div 32768 - 1\\ &\approx 94.0\text{ ppm}\end{aligned}$$

**Remarks 1.** The operating trigger division ratio is the division ratio of $f_{SUB}$ set by FMDIV2 to FMDIV0 of the frequency measurement control register. The operating trigger division ratio is $2^8$ when FMDIV2 to FMDIV0 = 000B, and $2^{15}$ when FMDIV2 to FMDIV0 = 111B.

**2.** The target correction value is the oscillation frequency deviation (unit: [ppm]) of the crystal resonator.

**3.** The target frequency is the frequency resulting after watch error correction performed.

Calculating the F8 to F0 value of the watch error correction register (SUBCUD)

The F8 to F0 value of the SUBCUD register is calculated from the target correction value by using the following expression.

$$\text{SUBCUD[8:0]} = \left( \frac{\text{Target correction value [ppm]} \times 2^{15}}{10^{6}} \right)_{\text{2's complement (9 bit fixed-point format)}} + 0001.00000\text{B}$$

**Examples 1.** When target correction value = –18.3 [ppm]

SUBCUD[8:0] = $(-18.3 \times 2^{15} / 10^{6})_{\text{2's complement (9 bit fixed-point format)}}$ + 0001.00000B
= $(-0.59965)_{\text{2's complement (9 bit fixed-point format)}}$ + 0001.00000B
= 1111.01101B + 0001.00000B
= 0000.01101B

**Examples 2.** When target correction value = 94.0 [ppm]

SUBCUD[8:0] = $(94.0 \times 2^{15} / 10^{6})_{\text{2's complement (9 bit fixed-point format)}}$ + 0001.00000B
= $(+3.08019)_{\text{2's complement (9 bit fixed-point format)}}$ + 0001.00000B
= 0011.00011B + 0001.00000B
= 0100.00011B

### 8.4.9 High-accuracy 1 Hz output

Clock correction by clock error correction register is possible at minimum resolution of 0.96 ppm by correcting the counter every 0.5 seconds, but since the counter is synchronized with $f_{RTC}$, the minimum resolution of normal 1 Hz output generated from counter overflow is 1/32.768 KHz (≈ 30.5 μs = 30.5 ppm). This means, normal 1 Hz output has a minimum resolution of 0.96 ppm over a long period, but each 1 Hz output includes an error of up to 30.5 ppm.

On the other hand, high-accuracy 1 Hz output allows each 1 Hz output to be corrected with a minimum resolution of 0.96 ppm and output, by using the correction value in the clock error correction register and counting the correction time with $f_{IH}$[Note]

**Note** Actual high-accuracy 1-Hz output includes quantization error in $f_{IH}$ accuracy and counting correction time. When using a high-precision 1 Hz output, select 24 MHz for high-speed on-chip oscillator clock ($f_{IH}$) and operate the high-speed on-chip oscillator (HIOSTOP=0). There is no need to select it for CPU clock.

# CHAPTER 9 SUBSYSTEM CLOCK FREQUENCY MEASUREMENT CIRCUIT

## 9.1 Subsystem Clock Frequency Measurement Circuit

The subsystem clock frequency measurement circuit is used to measure the frequency of the subsystem clock ($f_{SUB}$), by inputting the reference clock externally.

RTC clock error correction is possible without using a temperature sensor by measuring the subsystem clock ($f_{SUB}$) frequency with the following method.

- Input external main system clock ($f_{EX}$) as reference clock from an externally mounted temperature compensated crystal oscillator (TCXO)
- Use X1 oscillation clock ($f_X$) as reference clock by connecting an AT cut oscillator with good temperature characteristics to X1 and X2

**Caution The subsystem clock frequency measurement circuit can be used only when the subsystem clock ($f_{SUB}$ = 32.768 kHz) is selected as the operating clock (WUTMMCK0 in the OSMC register = 0).**

## 9.2 Configuration of Subsystem Clock Frequency Measurement Circuit

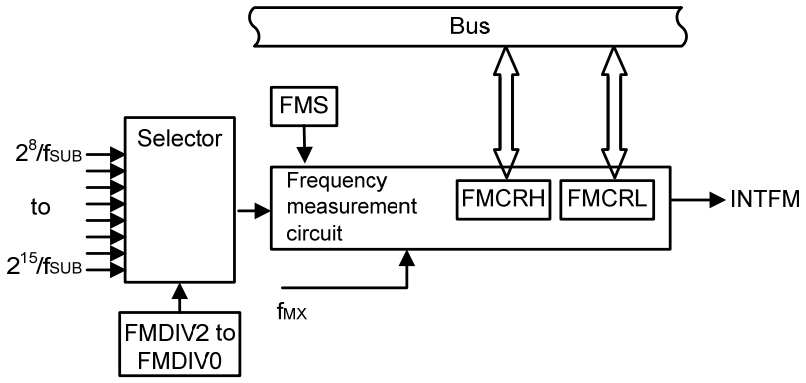
The subsystem clock frequency measurement circuit includes the following hardware.

**Table 9-1. Configuration of Subsystem Clock Frequency Measurement Circuit**

| Item | Configuration |
|---|---|
| Counter | Counter (32-bit) |
| Control registers | Peripheral enable register 1 (PER1) |
| | Subsystem clock supply mode control register (OSMC) |
| | Frequency measurement count register L (FMCRL) |
| | Frequency measurement count register H (FMCRH) |
| | Frequency measurement control register (FMCTL) |

Figure 9-1 shows the subsystem clock frequency measurement circuit diagram.

**Figure 9-1. Subsystem Clock Frequency Measurement Circuit Diagram**



## 9.3 Registers Controlling Subsystem Clock Frequency Measurement Circuit

The subsystem clock frequency measurement circuit is controlled by the following registers.

- Peripheral enable register 1 (PER1)
- Subsystem clock supply mode control register (OSMC)
- Frequency measurement count register L (FMCRL)
- Frequency measurement count register H (FMCRH)
- Frequency measurement control register (FMCTL)

### 9.3.1 Peripheral enable register 1 (PER1)

This register is used to enable or disable supplying the clock to the register used for the subsystem clock frequency measurement circuit. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

Of the registers that are used to control the subsystem clock frequency measurement circuit and real-time clock 2, the clock error correction register (SUBCUD) can be set by setting bit 6 (FMCEN) of this register to 1.

The PER1 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 9-2. Format of Peripheral Enable Register 1 (PER1)**

Address: F007AH After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PER1 | TMKAEN | FMCEN | CMPEN | OSDCEN | DTCEN | 0 | 0 | DSADCEN |

| FMCEN | Subsystem clock frequency measurement circuit |
|---|---|
| 0 | Stops input clock supply.<br>• SFR used by the subsystem clock frequency measurement f circuit cannot be written.<br>• SUBCUD register used by real-time clock 2 cannot be written.<br>• The subsystem clock frequency measurement circuit is in the reset status. |
| 1 | Enables input clock supply.<br>• SFR used by the subsystem clock frequency measurement circuit can be read/written.<br>• SUBCUD register used by real-time clock 2 can be read and written. |

**Cautions 1. The clock error correction register (SUBCUD) can be read or written by setting RTCWEN of peripheral enable register 0 (PER0) to 1 or setting FMCEN of peripheral enable register 1 (PER1) to 1.**

**2. Be sure to set bits 1 and 2 to “0”.**

### 9.3.2 Subsystem clock supply mode control register (OSMC)

This register is used to reduce power consumption by stopping unnecessary clock functions.

If the RTCLPC bit is set to 1, power consumption can be reduced, because clock supply to the peripheral functions other than real-time clock 2, 12-bit interval timer, clock output/buzzer output controller, LCD controller/driver, 8-bit interval timer, and oscillation stop detector is stopped in STOP mode or in HALT mode while the subsystem clock is selected as the CPU clock.

In addition, the OSMC register is used to select the operation clock of real-time clock 2, 12-bit interval timer, clock output/buzzer output controller, LCD controller/driver, 8-bit interval timer, and subsystem clock frequency measurement circuit.

The OSMC register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 9-3. Format of Subsystem Clock Supply Mode Control Register (OSMC)**

Address: F00F3H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| OSMC | RTCLPC | 0 | 0 | WUTMMCK0 | 0 | 0 | 0 | 0 |

| RTCLPC | In STOP mode and in HALT mode while the CPU operates using the subsystem clock |
|---|---|
| 0 | Enables subsystem clock supply to peripheral functions.<br>For peripheral functions for which operation is enabled, see **CHAPTER 24 STANDBY FUNCTION**. |
| 1 | Stops subsystem clock supply to peripheral functions other than real-time clock 2, 12-bit interval timer, clock output/buzzer output controller, LCD controller/driver, 8-bit interval timer, and oscillation stop detector. |

| WUTMMCK0 | Selection of operation clock for real-time clock 2, 12-bit interval timer, and LCD controller/driver. | Selection of clock output from PCLBUZn pin of clock output/buzzer output controller and selection of operation clock for 8-bit interval timer. | Operation of subsystem clock frequency measurement circuit. |
|---|---|---|---|
| 0 | Subsystem clock ($f_{SUB}$) | Selecting the subsystem clock ($f_{SUB}$) is enabled. | Enable |
| 1 | Low-speed on-chip oscillator clock ($f_{IL}$) | Selecting the subsystem clock ($f_{SUB}$) is disabled. | Disable |

**Cautions 1. Setting the RTCLPC bit to 1 can reduce current consumption in STOP mode and in HALT mode with the CPU operating on the subsystem clock. However, setting the RTCLPC bit to 1 means that there is no clock supply to peripheral circuits other than real-time clock 2, 12-bit interval timer, clock output/buzzer output controller, and LCD controller/driver in HALT mode with the CPU operating on the subsystem clock. Before setting the system to HALT mode with the CPU operating on the subsystem clock, therefore, be sure to set bit 7 (RTCWEN) of peripheral enable register 0 (PER0) and bit 7 (TMKAEN) of peripheral enable register 1 (PER1) to 1, and bits 0, 2, and 3 of PER0, and bit 5 of PER1 to 0.**

**2. If the subsystem clock is oscillating, only the subsystem clock can be selected (WUTMMCK0 = 0).**

**3. When WUTMMCK0 is set to 1, the low-speed on-chip oscillator clock oscillates.**

**4. When WUTMMCK0 is set to 1, only the constant-period interrupt function of real-time clock 2 can be used. The year, month, day of the week, day, hour, minute, and second counters and the 1 Hz output function of real-time clock 2 cannot be used.**

**The interval of the constant-period interrupt is calculated by constant period (value selected by using the RTCC0 register) × $f_{SUB}/f_{IL}$.**

### 9.3.3 Frequency measurement count register L (FMCRL)

This register represents the lower 16 bits of the frequency measurement count register (FMCR) in the frequency measurement circuit.

The FMCRL register can be read by a 16-bit memory manipulation instruction.

Reset signal generation clears the FMCRL register to 0000H.

**Figure 9-4. Format of Frequency Measurement Count Register L (FMCRL)**

Address: F0312H After reset: 0000H R

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FMCRL | | | | | | | | | | | | | | | | |

**Cautions 1. Do not read the value of FMCRL when FMS = 1.**

**2. Read the value of FMCRL after the frequency measurement complete interrupt is generated.**

### 9.3.4 Frequency measurement count register H (FMCRH)

This register represents the upper 16 bits of the frequency measurement count register (FMCR) in the frequency measurement circuit.

The FMCRH register can be read by a 16-bit memory manipulation instruction.

Reset signal generation clears the FMCRH register to 0000H.

**Figure 9-5. Frequency Measurement Count Register H (FMCRH)**

Address: F0314H After reset: 0000H R

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FMCRH | | | | | | | | | | | | | | | | |

**Cautions 1. Do not read the value of FMCRH when FMS = 1.**

**2. Read the value of FMCRH after the frequency measurement complete interrupt is generated.**

**Figure 9-6. Frequency Measurement Count Register (FMCRH, FMCRL)**

| Symbol | 31 | | | | | | | | | | | | | | | 16 | 15 | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FMCR | FMCRH | | | | | | | | | | | | | | | | FMCRL | | | | | | | | | | | | | | | |

### 9.3.5 Frequency measurement control register (FMCTL)

The FMCTL register is used to set the operation of the subsystem clock frequency measurement circuit. This register is used to start operation and set the period of frequency measurement.

The FMCTL register can be set by a 1-bit or 8-bit memory manipulation instruction.

<R> Reset signal generation clears the FMCTL register to 00H.

**Figure 9-7. Format of Frequency Measurement Control Register (FMCTL)**

Address: F0316H After reset: 00H R/W

| Symbol | <7> | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| FMCTL | FMS | 0 | 0 | 0 | 0 | FMDIV2 | FMDIV1 | FMDIV0 |

| FMS | Frequency measurement circuit operation enable |
|---|---|
| 0 | Stops the frequency measurement circuit. |
| 1 | Operates the frequency measurement circuit.<br>Starts counting on the rising edge of the operating clock and stops counting on the next rising edge of the operating clock. |

| FMDIV2 | FMDIV1 | FMDIV0 | Frequency measurement period setting |
|---|---|---|---|
| 0 | 0 | 0 | $2^8/f_{SUB}$ (7.8125 ms) |
| 0 | 0 | 1 | $2^9/f_{SUB}$ (15.625 ms) |
| 0 | 1 | 0 | $2^{10}/f_{SUB}$ (31.25 ms) |
| 0 | 1 | 1 | $2^{11}/f_{SUB}$ (62.5 ms) |
| 1 | 0 | 0 | $2^{12}/f_{SUB}$ (0.125 s) |
| 1 | 0 | 1 | $2^{13}/f_{SUB}$ (0.25 s) |
| 1 | 1 | 0 | $2^{14}/f_{SUB}$ (0.5 s) |
| 1 | 1 | 1 | $2^{15}/f_{SUB}$ (1 s) |

**Caution Do not read the value of the FMDIV2 to FMDIV0 bits when FMS = 1.**

**Remark** The frequency measurement resolution can be calculated by the formula below.

- Frequency measurement resolution = $10^6$/(frequency measurement period × reference clock frequency ($f_{MX}$) [Hz]) [ppm]

  Example 1) When FMDIV2 to FMDIV0 = 000B and $f_{MX}$ = 20 MHz, measurement resolution = 6.4 ppm

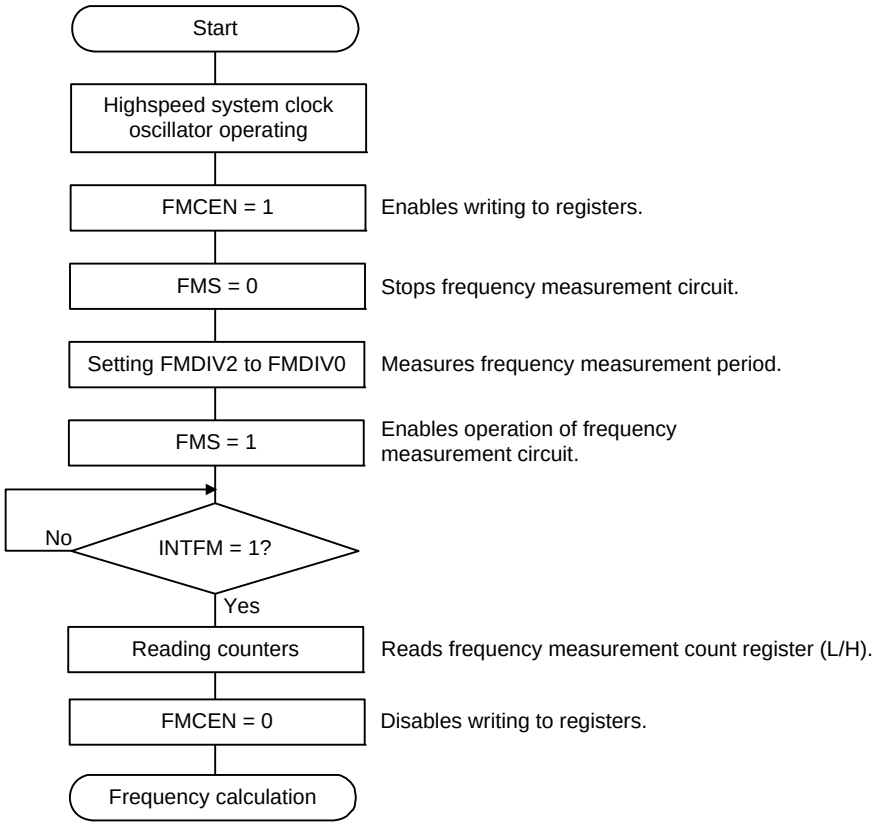
  Example 2) When FMDIV2 to FMDIV0 = 111B and $f_{MX}$ = 1 MHz, measurement resolution = 1 ppm

## 9.4 Subsystem Clock Frequency Measurement Circuit Operation

### 9.4.1 Setting subsystem clock frequency measurement circuit

Set subsystem clock frequency measurement circuit after setting 0 to FMS first.

**Figure 9-8. Procedure for Setting Subsystem Clock Frequency Measurement Circuit Using Reference Clock**




**Caution After the frequency measurement count register (L/H) is read, be sure to set FMCEN to 0.**

The $f_{SUB}$ oscillation frequency is calculated by using the following expression.

$$f_{SUB}\ \text{oscillation frequency} = \frac{\text{Reference clock frequency [Hz]} \times \text{operation trigger division ratio}}{\text{Frequency measurement count register value (FMCR)}}\ \text{[Hz]}$$

For example, when the frequency is measured under the following conditions

- Count clock frequency: $f_{MX}$ = 10 MHz
- Frequency measurement period setting register: FMDIV2 to FMDIV0 = 111B (operation trigger division ratio: $2^{15}$)

and the measurement result is as follows,

- Frequency measurement count register: FMCR = 10000160D

the $f_{SUB}$ oscillation frequency is obtained as below.

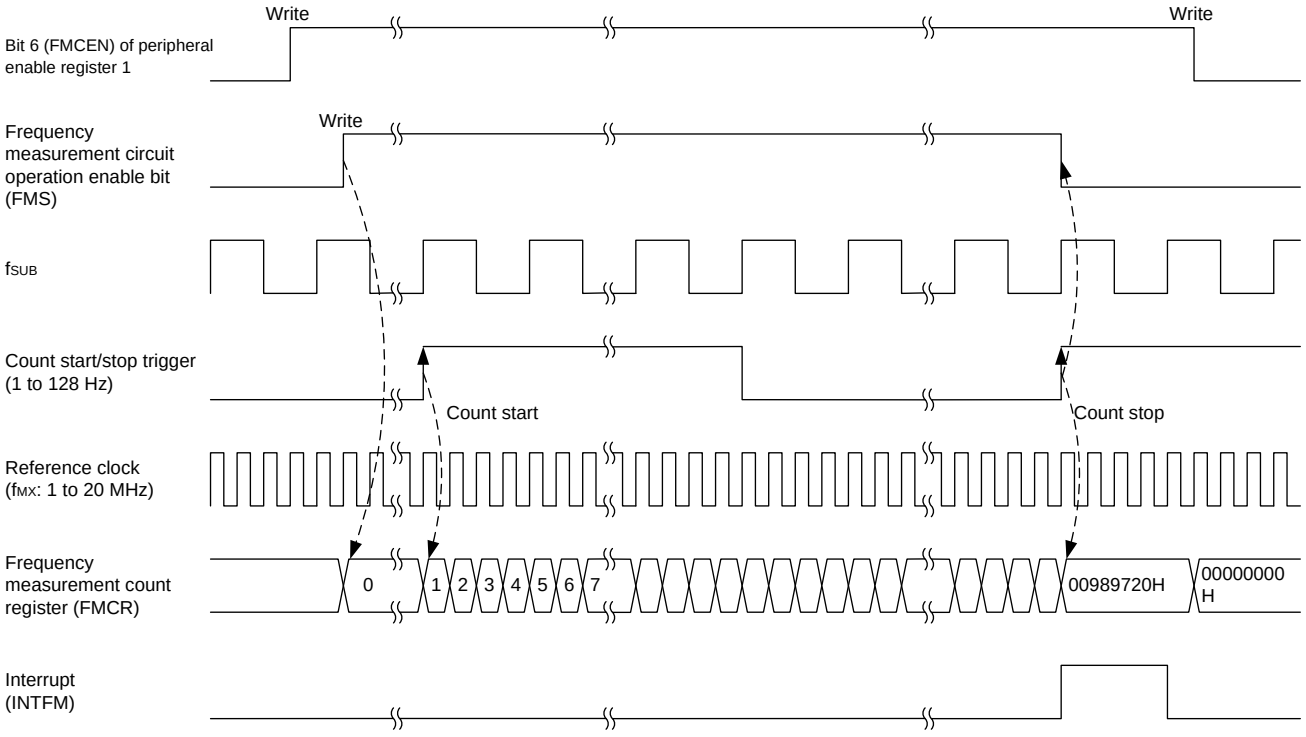
$$f_{SUB}\ \text{oscillation frequency} = \frac{(10 \times 10^6) \times 2^{15}}{10000160}\ 32767.47572\ \text{[Hz]}$$

### 9.4.2 Subsystem clock frequency measurement circuit operation timing

The operation timing of the subsystem clock frequency measurement circuit is shown in Figure 9-9.

After the frequency measurement circuit operation enable bit (FMS) is set to 1, counting is started by the count start trigger set with the frequency measurement period setting bits (FMDIV2 to FMDIV0) and stopped by the next trigger. After counting is stopped, the count value is retained, and the frequency measurement circuit operation enable bit (FMS) is reset to 0. An interrupt is also generated for one clock of $f_{SUB}$. After the operation of the frequency measurement circuit is completed (FMS = 0) and the frequency measurement count register (L/H) is read, be sure to set bit 6 (FMCEN) of peripheral enable register 1 to 0.

**Figure 9-9. Subsystem Clock Frequency Measurement Circuit Operation Timing**



# CHAPTER 10 12-BIT INTERVAL TIMER

## 10.1 Functions of 12-bit Interval Timer

An interrupt (INTIT) is generated at any previously specified time interval. It can be utilized for wakeup from STOP mode and triggering an A/D converter's SNOOZE mode.

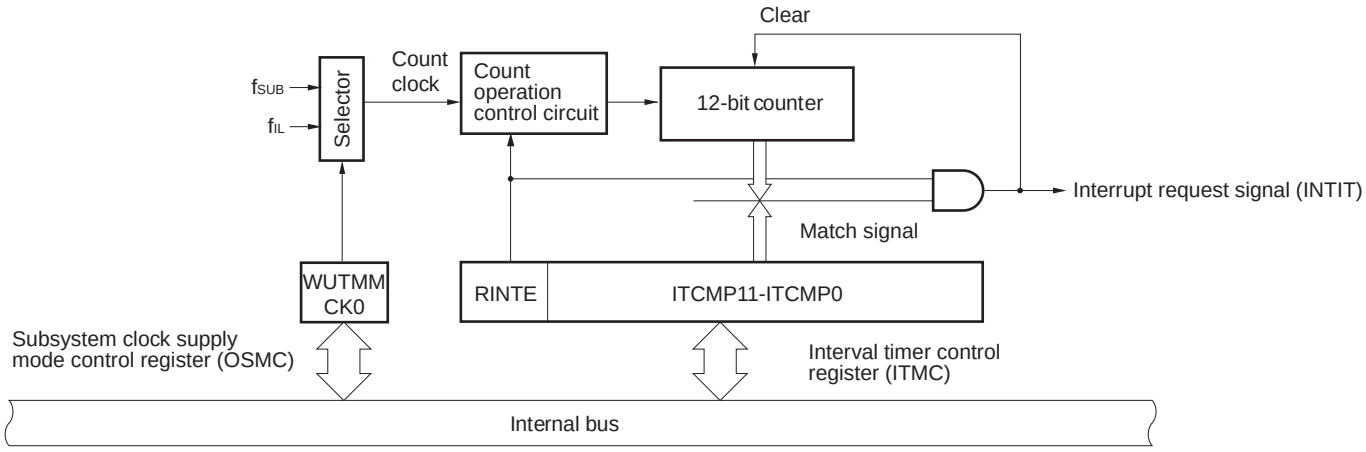
## 10.2 Configuration of 12-bit Interval Timer

The 12-bit interval timer includes the following hardware.

**Table 10-1. Configuration of 12-bit Interval Timer**

| Item | Configuration |
|---|---|
| Counter | 12-bit counter |
| Control registers | Peripheral enable register 1 (PER1) |
| | Subsystem clock supply mode control register (OSMC) |
| | 12-bit interval timer control register (ITMC) |

**Figure 10-1. Block Diagram of 12-bit Interval Timer**



## 10.3 Registers Controlling 12-bit Interval Timer

The 12-bit interval timer is controlled by the following registers.

- Peripheral enable register 1 (PER1)
- Subsystem clock supply mode control register (OSMC)
- 12-bit interval timer control register (ITMC)

### 10.3.1 Peripheral enable register 1 (PER1)

This register is used to enable or disable supplying the clock to the peripheral hardware. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

When the 12-bit interval timer is used, be sure to set bit 7 (TMKAEN) of this register to 1.

The PER1 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 10-2. Format of Peripheral Enable Register 1 (PER1)**

Address: F007AH After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PER1 | TMKAEN | FMCEN | CMPEN | OSDCEN | DTCEN | 0 | 0 | DSADCEN |

| TMKAEN | Control of 12-bit interval timer input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFRs used by the 12-bit interval timer cannot be written.<br>• The 12-bit interval timer is in the reset status. |
| 1 | Enables input clock supply.<br>• SFRs used by the 12-bit interval timer can be read and written. |

**Cautions 1. When using an 12-bit interval timer, be sure to set TMKAEN = 1 beforehand with the count clock oscillation stabilized. If TMKAEN = 0, writing to a control register of the 12-bit interval timer is ignored, and, even if the register is read, only the default value is read. (except the subsystem clock supply mode control register (OSMC))**

**2. Clock supply to peripheral functions other than real-time clock 2, 12-bit interval timer, clock output/buzzer output controller, LCD controller/driver, 8-bit interval timer, and oscillation stop detector can be stopped in STOP mode or HALT mode when the subsystem clock is used, by setting the RTCLPC bit of the subsystem clock supply mode control register (OSMC) to 1.**

**3. Be sure to set bits 2 and 1 to “0”.**

### 10.3.2 Subsystem clock supply mode control register (OSMC)

The OSMC register is used to reduce power consumption by stopping unnecessary clock functions.

If the RTCLPC bit is set to 1, power consumption can be reduced, because clock supply to the peripheral functions, except real-time clock 2, 12-bit interval timer, clock output/buzzer output controller, and LCD controller/driver, is stopped in STOP mode or HALT mode while subsystem clock is selected as CPU clock.

In addition, the OSMC register can be used to select the operation clock of real-time clock 2, 12-bit interval timer, clock output/buzzer output controller, and LCD controller/driver.

The OSMC register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 10-3. Format of Subsystem Clock Supply Mode Control Register (OSMC)**

Address: F00F3H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| OSMC | RTCLPC | 0 | 0 | WUTMMCK0 | 0 | 0 | 0 | 0 |

| WUTMMCK0 | Selection of operation clock for real-time clock 2, 12-bit interval timer, and LCD controller/driver. | Selection of clock output from PCLBUZn pin of clock output/buzzer output controller and selection of operation clock for 8-bit interval timer. | Operation of subsystem clock frequency measurement circuit. |
|---|---|---|---|
| 0 | Subsystem clock ($f_{SUB}$) | Selecting the subsystem clock ($f_{SUB}$) is enabled. | Enable |
| 1 | Low-speed on-chip oscillator clock ($f_{IL}$) | Selecting the subsystem clock ($f_{SUB}$) is disabled. | Disable |

**Cautions 1. Be sure to select the subsystem clock (WUTMMCK0 bit = 0) if the subsystem clock is oscillating.**

**2. When WUTMMCK0 is set to 1, the low-speed on-chip oscillator clock oscillates.**

**3. The subsystem clock and low-speed on-chip oscillator clock can only be switched by using the WUTMMCK0 bit if real-time clock 2, 12-bit interval timer, and LCD controller/driver are all stopped.**

### 10.3.3 12-bit interval timer control register (ITMC)

This register is used to set up the starting and stopping of the 12-bit interval timer operation and to specify the timer compare value.

The ITMC register can be set by a 16-bit memory manipulation instruction.

Reset signal generation clears this register to 0FFFH.

**Figure 10-4. Format of 12-bit Interval Timer Control Register (ITMC)**

Address: FFF90H After reset: 0FFFH R/W

| Symbol | 15 | 14 | 13 | 12 | 11 to 0 |
|---|---|---|---|---|---|
| ITMC | RINTE | 0 | 0 | 0 | ITMCMP11 to ITMCMP0 |

| RINTE | 12-bit interval timer operation control |
|---|---|
| 0 | Count operation stopped (count clear) |
| 1 | Count operation started |

| ITMCMP11 to ITMCMP0 | Specification of the 12-bit interval timer compare value |
|---|---|
| 001H | These bits generate an interrupt at the fixed cycle (count clock cycles × (ITMCMP setting + 1)). |
| • • • | |
| FFFH | |
| 000H | Setting prohibit |

Example interrupt cycles when 001H or FFFH is specified for ITMCMP11 to ITMCMP0

- ITMCMP11 to ITMCMP0 = 001H, count clock: when $f_{SUB}$ = 32.768 kHz
  1/32.768 [kHz] × (1 + 1) = 0.06103515625 [ms] ≅ 61.03 [μs]
- ITMCMP11 to ITMCMP0 = FFFH, count clock: when $f_{SUB}$ = 32.768 kHz
  1/32.768 [kHz] × (4095 + 1) = 125 [ms]

**Cautions 1. Before changing the RINTE bit from 1 to 0, use the interrupt mask flag register to disable the INTIT interrupt servicing. When the operation starts (from 0 to 1) again, clear the ITIF flag, and then enable the interrupt servicing.**

**2. The value read from the RINTE bit is applied one count clock cycle after setting the RINTE bit.**

**3. When setting the RINTE bit after returned from standby mode and entering standby mode again, confirm that the written value of the RINTE bit is reflected, or wait that more than one clock of the count clock has elapsed after returned from standby mode. Then enter standby mode.**

**4. Only change the setting of the ITMCMP11 to ITMCMP0 bits when RINTE = 0.**
**However, it is possible to change the settings of the ITMCMP11 to ITMCMP0 bits at the same time as when changing RINTE from 0 to 1 or 1 to 0.**

## 10.4 12-bit Interval Timer Operation

### 10.4.1 12-bit interval timer operation timing

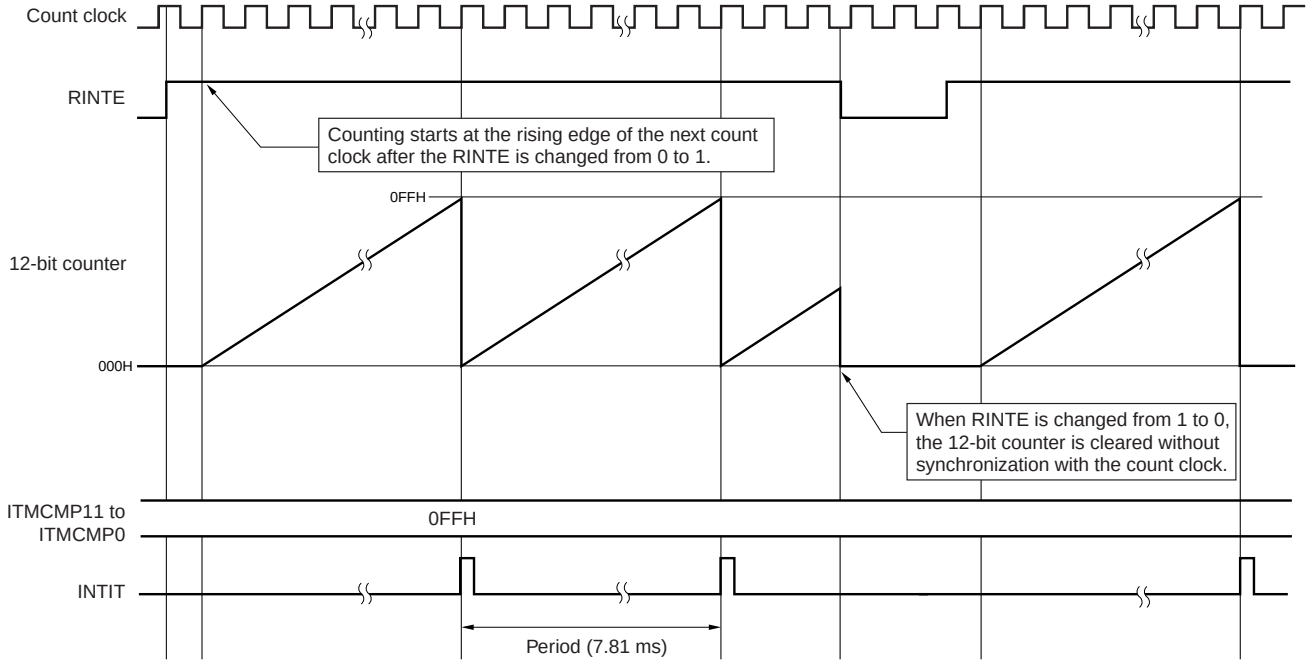
The count value specified for the ITMCMP11 to ITMCMP0 bits is used as an interval to operate an 12-bit interval timer that repeatedly generates interrupt requests (INTIT).

When the RINTE bit is set to 1, the 12-bit counter starts counting.

When the 12-bit counter value matches the value specified for the ITMCMP11 to ITMCMP0 bits, the 12-bit counter value is cleared to 0, counting continues, and an interrupt request signal (INTIT) is generated at the same time.

The basic operation of the 12-bit interval timer is as follows.

**Figure 10-5. 12-bit Interval Timer Operation Timing (ITMCMP11 to ITMCMP0 = 0FFH, Count Clock: $f_{SUB}$ = 32.768 kHz)**

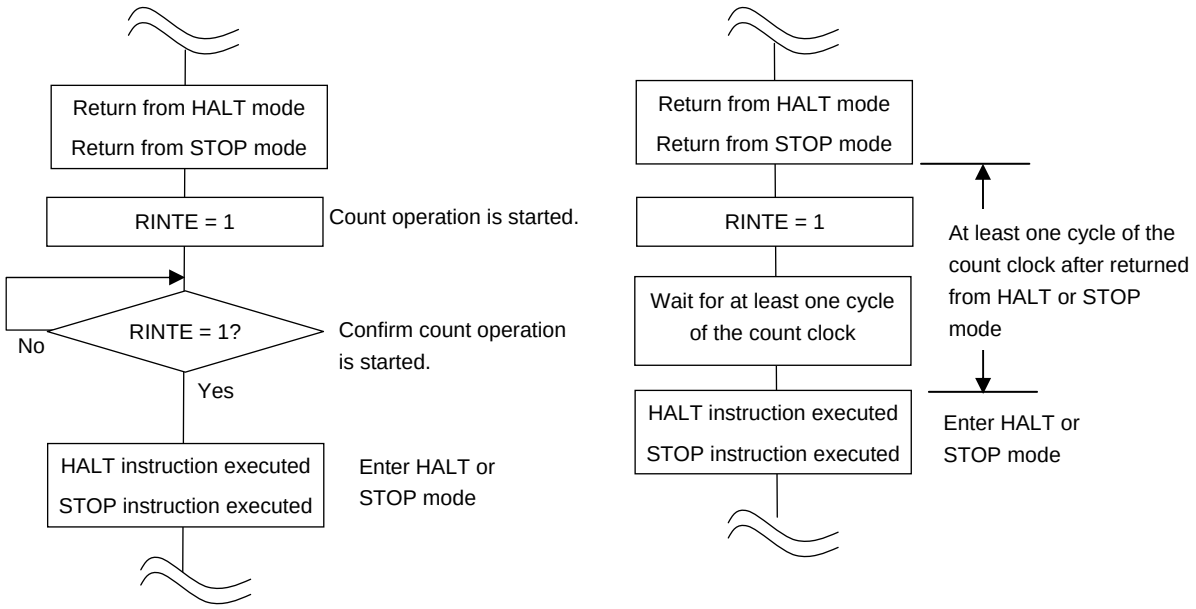


### 10.4.2 Start of count operation and re-enter to HALT/STOP mode after returned from HALT/STOP mode

When setting the RINTE bit to 1 after returned from HALT or STOP mode and entering HALT or STOP mode again, write 1 to the RINTE bit, and confirm the written value of the RINTE bit is reflected or wait for at least one cycle of the count clock. Then, enter HALT or STOP mode.

- After setting RINTE to 1, confirm by polling that the RINTE bit has become 1, and then enter HALT or STOP mode (see **Example 1** in **Figure 10-6**).
- After setting RINTE to 1, wait for at least one cycle of the count clock and then enter HALT or STOP mode (see **Example 2** in **Figure 10-6**).

**Figure 10-6. Procedure of Entering to HALT or STOP Mode After Setting RINTE to 1**




**Example 1** **Example 2**

# CHAPTER 11 8-BIT INTERVAL TIMER

The 8-bit interval timer has two 8-bit timers (channel 0 and channel 1) with each operating independently. In addition, the two 8-bit timers can be connected to operate as a 16-bit timer.

The 8-bit interval timer contains two units, 8-bit interval timer_0 and 8-bit interval timer_1, which have the same function. This chapter describes these units as the 8-bit interval timer unless there are differences between them.

## 11.1 Overview

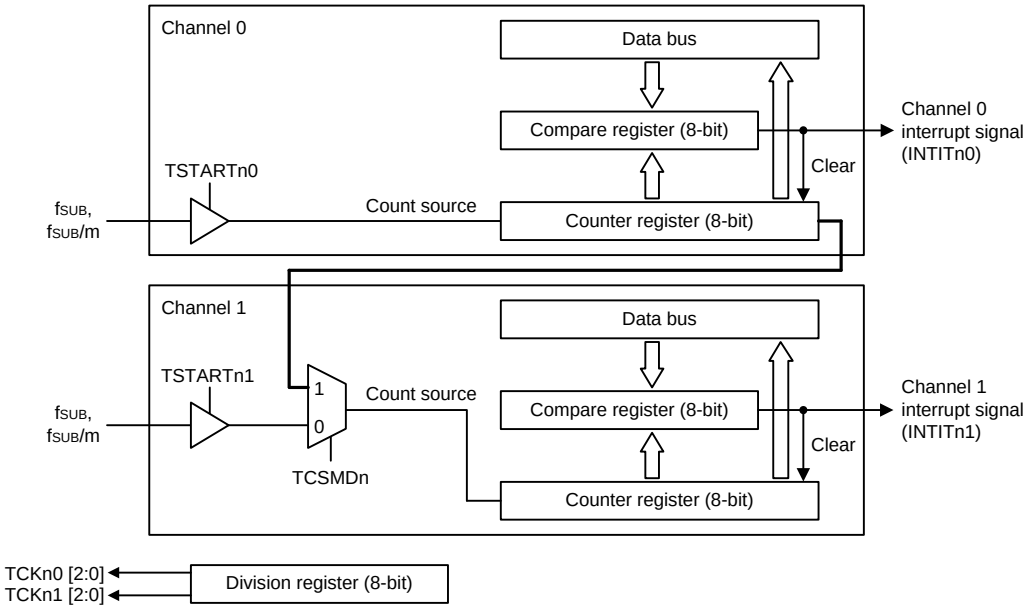
The 8-bit interval timer is an 8-bit timer that operates using the $f_{SUB}$ clock, which is asynchronous with the CPU.

Table 11-1 lists the 8-bit interval timer specifications and Figure 11-1 shows the 8-bit interval timer block diagram.

**Table 11-1. 8-bit Interval Timer Specifications**

| Item | Description |
|---|---|
| Count source (operating clock) | • $f_{SUB}$, $f_{SUB}/2$, $f_{SUB}/4$, $f_{SUB}/8$, $f_{SUB}/16$, $f_{SUB}/32$, $f_{SUB}/64$, $f_{SUB}/128$ |
| Operating mode | • 8-bit counter mode<br>Channel 0 and channel 1 operate independently as an 8-bit counter<br>• 16-bit counter mode<br>Channel 0 and channel 1 are connected to operate as a 16-bit counter |
| Interrupt | • Output when the counter value matches the compare value |

**Figure 11-1. 8-bit Interval Timer Block Diagram**




TSTARTni (i = 0, 1), TCSMDn, TCLKENn: Bits in TRTCRn register
TCKni [2:0]: Bits in TRTMDn register
m = 2, 4, 8, 16, 32, 64, 128
n = 0, 1

## 11.2 I/O Pins

The 8-bit interval timer does not have any I/O pins.

## 11.3 Registers

Table 11-2 lists the 8-bit interval timer registers.

**Table 11-2. Registers**

| Item | Configuration |
|---|---|
| Control registers | 8-bit interval timer counter register 00 (TRT00)Note 1 |
| | 8-bit interval timer counter register 01 (TRT01)Note 1 |
| | 8-bit interval timer counter register 0 (TRT0)Note 2 |
| | 8-bit interval timer compare register 00 (TRTCMP00)Note 1 |
| | 8-bit interval timer compare register 01 (TRTCMP01)Note 1 |
| | 8-bit interval timer compare register 0 (TRTCMP0)Note 2 |
| | 8-bit interval timer control register 0 (TRTCR0) |
| | 8-bit interval timer division register 0 (TRTMD0) |
| | 8-bit interval timer counter register 10 (TRT10)Note 1 |
| | 8-bit interval timer counter register 11 (TRT11)Note 1 |
| | 8-bit interval timer counter register 1 (TRT1)Note 2 |
| | 8-bit interval timer compare register 10 (TRTCMP10)Note 1 |
| | 8-bit interval timer compare register 11 (TRTCMP11)Note 1 |
| | 8-bit interval timer compare register 1 (TRTCMP1)Note 2 |
| | 8-bit interval timer control register 1 (TRTCR1) |
| | 8-bit interval timer division register 1 (TRTMD1) |

**Notes 1.** Can be accessed only when the TCSMDn bit in the TRTCRn register is 0.

**2.** Can be accessed only when the TCSMDn bit in the TRTCRn register is 1.

**Remark** n = 0, 1

### 11.3.1 8-bit interval timer counter register ni (TRTni) (n = 0 or 1, i = 0 or 1)

This is the 8-bit interval timer counter register. It is used as a counter that counts up based on the count clock.

The TRTni register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 11-2. Format of 8-bit Interval Timer Counter Register ni (TRTni)**

Address: F0540H (TRT00), F0541H (TRT01), F0548H (TRT10), F0549H (TRT11) After reset: 00H R[Notes 1, 2]

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TRTni | | | | | | | | |

**Notes 1.** The TRTni register is set to 00H two count clock cycles after the compare register TRTCMPni is write-accessed. See **11.4.4 Timing of updating compare register values**.

**2.** Can be accessed only when the mode select bit (TCSMDn) in 8-bit interval timer control register n (TRTCRn) is 0.

### 11.3.2 8-bit interval timer counter register n (TRTn) (n = 0 or 1)

This is a 16-bit counter register when the 8-bit interval timer is used in 16-bit interval timer mode.

The TRTn register can be set by a 16-bit memory manipulation instruction.

Reset signal generation sets this register to 0000H.

**Figure 11-3. Format of 8-bit Interval Timer Counter Register n (TRTn)**

Address: F0540H (TRT0), F0548H (TRT1) After reset: 0000H R[Notes 1, 2]

| | F0541H (TRT01) / F0549H (TRT11) | | | | | | | | F0540H (TRT00) / F0548H (TRT10) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| TRTn | | | | | | | | | | | | | | | | |

**Notes 1.** The TRTn register is set to 0000H two count clock cycles after the compare register TRTCMPn is write-accessed. See **11.4.4 Timing of updating compare register values**.

**2.** Can be accessed only when the mode select bit (TCSMDn) in 8-bit interval timer control register n (TRTCRn) is 1.

### 11.3.3 8-bit interval timer compare register ni (TRTCMPni) (n = 0 or 1, i = 0 or 1)

This is the 8-bit interval timer compare value register.

The TRTCMPni register can be set by an 8-bit memory manipulation instruction.

Reset signal generation sets this register to FFH.

Setting range is 01H to FFH[Note 1].

This register is used to store the compare value of registers TRTn0 and TRTn1 (counters).

Write-access clears the count value (TRTn0, TRTn1) to 00H.

See **11.4.4 Timing of updating compare register values** for the timing of rewriting the compare value.

**Figure 11-4. Format of 8-bit Interval Timer Compare Register ni (TRTCMPni)**

Address: F0350H (TRTCMP00), F0351H (TRTCMP01), F0358H (TRTCMP10), F0359H (TRTCMP11) After reset: FFH R/W[Note 2]

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TRTCMPni | | | | | | | | |

**Notes 1.** The TRTCMPni register must not be set to 00H.

**2.** Can be accessed only when the mode select bit (TCSMDn) in 8-bit interval timer control register n (TRTCRn) is 0.

### 11.3.4 8-bit interval timer compare register n (TRTCMPn) (n = 0 or 1)

This is a compare value register when the 8-bit interval timer is used in 16-bit interval timer mode.

The TRTCMPn register can be set by a 16-bit memory manipulation instruction.

Reset signal generation sets this register to FFFFH.

Setting range is 0001H to FFFFH[Note 1].

This register is used to store the compare value of the TRTn register (counter).

Write-access clears the count value (TRTn) to 0000H.

See **11.4.4 Timing of updating compare register values** for the timing of rewriting the compare value.

**Figure 11-5. Format of 8-bit Interval Timer Compare Register n (TRTCMPn)**

Address: F0350H (TRTCMP0), F0358H (TRTCMP1) After reset: FFFFH R/W[Note 2]

| | F0351H (TRTCMP01) / F0359H (TRTCMP11) | | | | | | | | F0350H (TRTCMP00) / F0358H (TRTCMP10) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| TRTCMPn | | | | | | | | | | | | | | | | |

**Notes 1.** The TRTCMPn register must not be set to 0000H.

**2.** Can be accessed only when the mode select bit (TCSMDn) in 8-bit interval timer control register n (TRTCRn) is 1.

### 11.3.5 8-bit interval timer control register n (TRTCRn) (n = 0 or 1)

This register is used to start and stop counting by the 8-bit interval timer and to switch between using the 8-bit interval timer as an 8-bit counter or a 16-bit counter.

The TRTCRn register can be set by a 1-bit or 8-bit manipulation instruction.

Reset signal generation resets this register to 00H.

**Figure 11-6. Format of 8-bit Interval Timer Control Register n (TRTCRn)**

Address: F0352H (TRTCR0), F035AH (TRTCR1) After reset: 00H R/W[Note 3]

| Symbol | 7 | 6 | 5 | 4 | 3 | <2> | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| TRTCRn | TCSMDn | 0 | 0 | TCLKENn | 0 | TSTARTn1 | 0 | TSTARTn0 |

| TCSMDn | Mode select |
|---|---|
| 0 | Operates as 8-bit counter |
| 1 | Operates as 16-bit counter (channel 0 and channel 1 are connected) |
| See **11.4 Operation** for details. | |

| TCLKENn | 8-bit interval timer clock enable[Note 1] |
|---|---|
| 0 | Clock is stopped |
| 1 | Clock is supplied |

| TSTARTn1 | 8-bit interval timer 1 count start[Notes 1, 2] |
|---|---|
| 0 | Counting stops |
| 1 | Counting starts |
| In 8-bit interval timer mode, writing 1 to the TSTARTn1 bit triggers the start of counting by TRTn1, and writing 0 stops counting by TRTn1.<br>In 16-bit interval timer mode, this bit is invalid because it is not used. See **11.4 Operation** for details. | |

| TSTARTn0 | 8-bit interval timer 0 count start[Notes 1, 2] |
|---|---|
| 0 | Counting stops |
| 1 | Counting starts |
| In 8-bit interval timer mode, writing 1 to the TSTARTn0 bit triggers the start of counting by TRTn0, and writing 0 stops counting by TRTn0.<br>In 16-bit interval timer mode, writing 1 to the TSTARTn0 bit triggers the start of counting by TRTn, and writing 0 stops counting by TRTn.<br>See **11.4 Operation** for details. | |

**Notes 1.** Be sure to set the TCLKENn bit to 1 before setting the 8-bit interval timer. To stop the clock, set TSTARTn0 and TSTARTn1 to 0 and then set the TCLKENn bit to 0 after one or more $f_{SUB}$ cycles have elapsed. See **11.5.3 8-bit interval timer setting procedure** for details.

**2.** See **11.5.1 Changing the operating mode and clock settings** for notes on using bits TSTARTn0, TSTARTn1, and TCSMDn.

**3.** Bits 6, 5, 3, and 1 are read-only. When writing, write 0. When reading, 0 is read.

### 11.3.6 8-bit interval timer division register n (TRTMDn) (n = 0 or 1)

This register is used to select the division ratio of the count source used by the 8-bit interval timer.

The TRTMDn register can be set by an 8-bit memory manipulation instruction.

Reset signal generation sets this register to 00H.

**Figure 11-7. Format of 8-bit Interval Timer Division Register n (TRTMDn)**

Address: F0353H (TRTMD0), F035BH (TRTMD1) After reset: 00H R/W[Note 4]

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TRTMDn | 0 | TCKn1 | | | 0 | TCKn0 | | |

| TCKn1 | | | Selection of division ratio for 8-bit interval timer 1[Notes 1, 2, 3] |
|---|---|---|---|
| Bit 6 | Bit 5 | Bit 4 | |
| 0 | 0 | 0 | $f_{SUB}$ |
| 0 | 0 | 1 | $f_{SUB}/2$ |
| 0 | 1 | 0 | $f_{SUB}/4$ |
| 0 | 1 | 1 | $f_{SUB}/8$ |
| 1 | 0 | 0 | $f_{SUB}/16$ |
| 1 | 0 | 1 | $f_{SUB}/32$ |
| 1 | 1 | 0 | $f_{SUB}/64$ |
| 1 | 1 | 1 | $f_{SUB}/128$ |
| In 8-bit interval timer mode, TRTn1 counts based on the count clock specified by TCKn1.<br>In 16-bit interval timer mode, set these bits to 000B because they are not used. See **11.4 Operation** for details. | | | |

| TCKn0 | | | Selection of division ratio for 8-bit interval timer 0[Notes 1, 2, 3] |
|---|---|---|---|
| Bit 2 | Bit 1 | Bit 0 | |
| 0 | 0 | 0 | $f_{SUB}$ |
| 0 | 0 | 1 | $f_{SUB}/2$ |
| 0 | 1 | 0 | $f_{SUB}/4$ |
| 0 | 1 | 1 | $f_{SUB}/8$ |
| 1 | 0 | 0 | $f_{SUB}/16$ |
| 1 | 0 | 1 | $f_{SUB}/32$ |
| 1 | 1 | 0 | $f_{SUB}/64$ |
| 1 | 1 | 1 | $f_{SUB}/128$ |
| In 8-bit interval timer mode, TRTn0 counts based on the count clock specified by TCKn0.<br>In 16-bit interval timer mode, TRTn counts based on the count clock specified by TCKn0.<br>See **11.4 Operation** for details. | | | |

**Notes 1.** Do not switch the count source during counting. When switching the count source, set these bits while the TSTARTni bit in the TRTCRn register is 0 (counting is stopped).

**2.** Set TCKni (i = 0, 1) of the unused channel to 000B.

**3.** Be sure to set the TCKni (i = 0, 1) bit before setting the TRTCMPni register.

**4.** Bits 7 and 3 are read-only. When writing, write 0. When reading, 0 is read.

## 11.4 Operation

### 11.4.1 Counter mode

The following two modes are supported: 8-bit counter mode and 16-bit counter mode. Table 11-3 lists the registers and settings used in 8-bit counter mode and Table 11-4 lists the registers and settings used in 16-bit counter mode.

**Table 11-3. Registers and Settings Used in 8-bit Counter Mode**

| Register Name (Symbol) | Bit | Function |
|---|---|---|
| 8-bit interval timer counter register n0 (TRTn0) | b7 to b0 | 8-bit counter of channel 0.<br>The count value can be read. |
| 8-bit interval timer counter register n1 (TRTn1) | b7 to b0 | 8-bit counter of channel 1.<br>The count value can be read. |
| 8-bit interval timer compare register n0 (TRTCMPn0) | b7 to b0 | 8-bit compare value of channel 0.<br>Set the compare value. |
| 8-bit interval timer compare register n1 (TRTCMPn1) | b7 to b0 | 8-bit compare value of channel 1.<br>Set the compare value. |
| 8-bit interval timer control register n (TRTCRn) | TSTARTn0 | Select whether to start/stop counting by channel 0. |
| | TSTARTn1 | Select whether to start/stop counting by channel 1. |
| | TCLKENn | Set to 1. |
| | TCSMDn | Set to 0. |
| 8-bit interval timer division register n (TRTMDn) | TCKn0 | Select the count clock of channel 0. |
| | TCKn1 | Select the count clock of channel 1. |

**Remark** n = 0, 1

**Table 11-4. Registers and Settings Used in 16-bit Counter Mode**

| Register Name (Symbol) | Bit | Function |
|---|---|---|
| 8-bit interval timer counter register n (TRTn) | b15 to b0 | 16-bit counter.<br>The count value can be read. |
| 8-bit interval timer compare register n (TRTCMPn) | b15 to b0 | 16-bit compare value.<br>Set the compare value. |
| 8-bit interval timer control register n (TRTCRn) | TSTARTn0 | Select whether to start/stop counting. |
| | TSTARTn1 | Set to 0. |
| | TCLKENn | Set to 1. |
| | TCSMDn | Set to 1. |
| 8-bit interval timer division register n (TRTMDn) | TCKn0 | Select the count clock. |
| | TCKn1 | Set to 000B. |

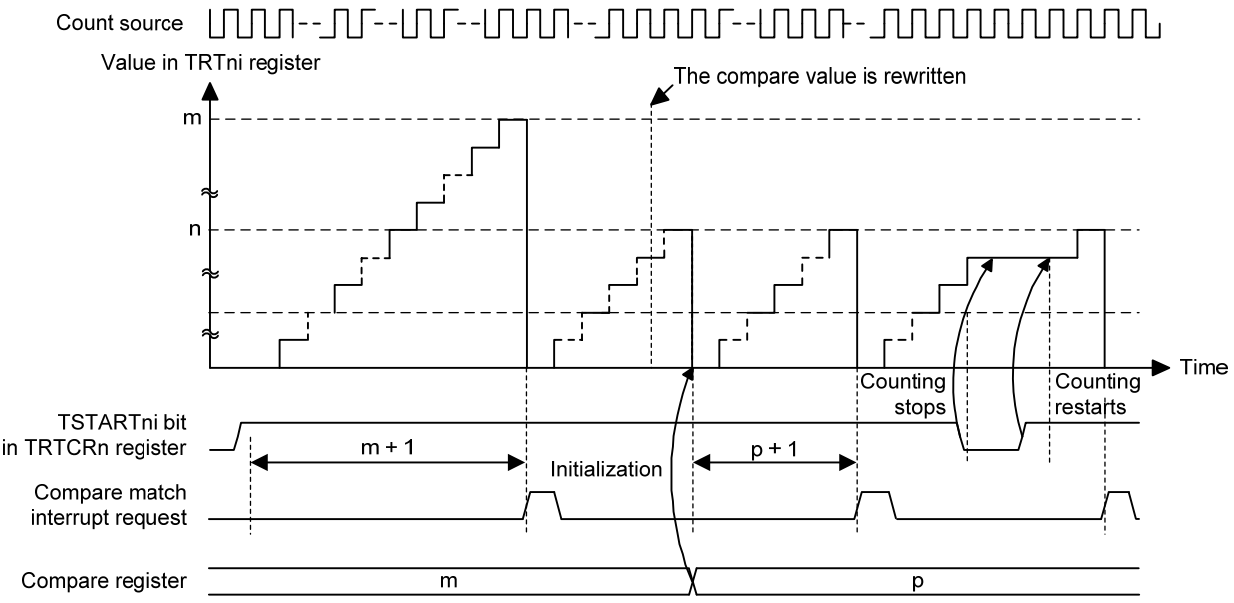
**Remark** n = 0, 1

### 11.4.2 Timer operation

Count is incremented by the count source selected by the TCKni (n = 0, 1, i = 0, 1) bit of the divider register (TRTMDn). The counter is incremented by 1 each time a count source is input and when the count reaches the compare value, compare match occurs and interrupt request is generated next time the count source is input. The interrupt request is output with a single count source synchronization pulse. However, when the count value reaches to 00H and stops by setting the TSTARTn1 bit of the TRTCRn register to 0, interrupt request is continuously generated.

When operation stops, the counter retains the count value immediately before operation was stopped. To clear the count value, set the compare value to the TRTCMPni register again. After the TRTCMPni register is written, the count value is cleared after two count source cycles have elapsed.

**Figure 11-8. Example of Timer Operation**




**Remark** n = 0 or 1, i = 0 or 1 m, p: Values set in TRTCMPni register

However, the initial 00H count interval when starting count varies as follows according to the timing 1 is written in the TSTARTni (I = 0, 1) bit of the TRTCR register.

- When $f_{SUB}$ is selected as the count source
  Maximum: Two count source cycle
  Minimum: One count source cycle
- When $f_{SUB}/2^m$ is selected as the count source
  Maximum: One count source cycle
  Minimum: One subsystem clock ($f_{SUB}$) cycle

When the count value matches the compare value, the count value is cleared at the next count source cycle. When the compare value in the TRTCMPni register is rewritten, the count value is also cleared two count source cycles after writing.

Table 11-5 lists the interrupt sources in 8-bit and 16-bit counter modes.

**Table 11-5. Interrupt Sources in 8-bit and 16-bit Counter Modes**

| Interrupt Name | Interrupt Source in 8-bit Counter Mode | Interrupt Source in 16-bit Counter Mode |
|---|---|---|
| INTITn0 | Rising edge of the count source cycle after compare match on channel 0 | Rising edge of the count source cycle after compare match |
| INTITn1 | Rising edge of the count source cycle after compare match on channel 1 | Not generated |

**Remark** n = 0, 1

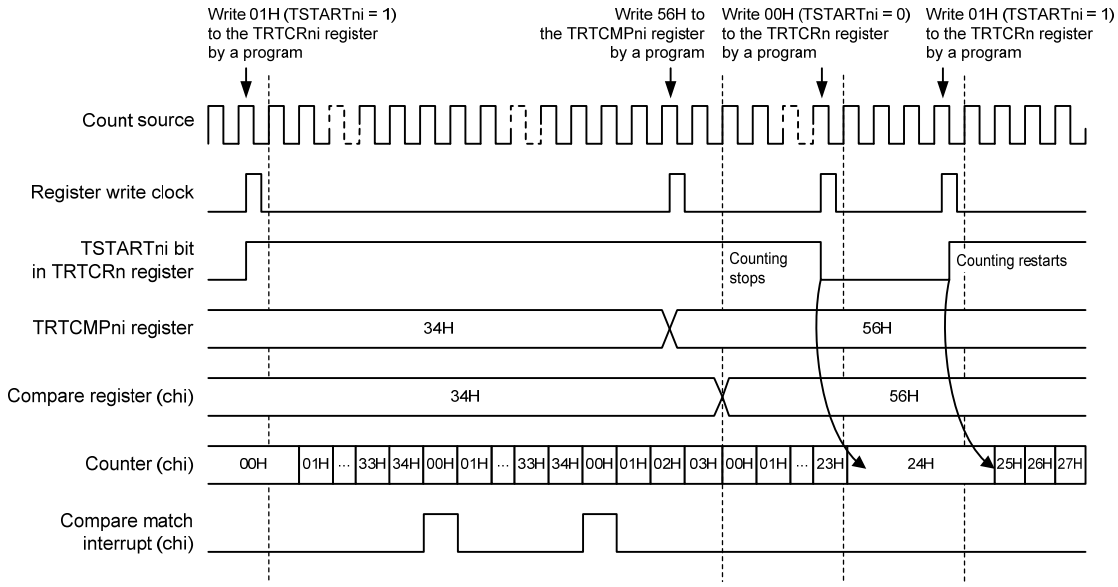
### 11.4.3 Count start/stop timing

**(1) When $f_{SUB}$ is selected as count source**

After 1 is written to the TSTARTni (n = 0 or 1, i = 0 or 1) bit in the TRTCRn register, counting starts at the next subsystem clock ($f_{SUB}$) cycle, and then the counter is incremented from 00H to 01H at the next count source ($f_{SUB}$) cycle. Similarly, after writing 0 in the TSTARTni, the count is stopped after counting with the subsystem clock ($f_{SUB}$).

Figure 11-9 shows the count start/stop timing and Figure 11-10 shows the timing for stop count → set compare register (clear count) → start count. Figure 11-9 and Figure 11-10 show the update timing in 8-bit counter mode, but the operation is performed at the same timing even in 16-bit counter mode.

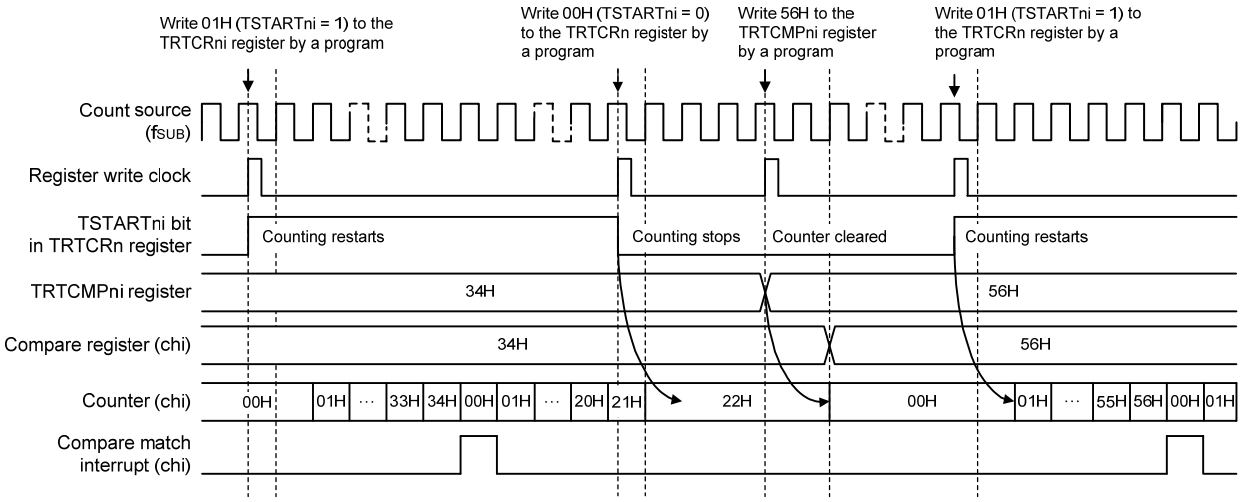
**Figure 11-9. Example of Count Start/Stop Operation (When $f_{SUB}$ Is Selected)**




The TCSMDn bit in the TRTCRn register is set to 0 (8-bit counter operation).

**Remark** n = 0 or 1, i = 0 or 1

**Figure 11-10. Example of Stopping Counting → Clearing the Counter → Restarting Counting (When $f_{SUB}$ Is Selected)**




The TCSMDn bit in the TRTCRn register is set to 0 (8-bit counter operation).

**Remark** n = 0 or 1, i = 0 or 1

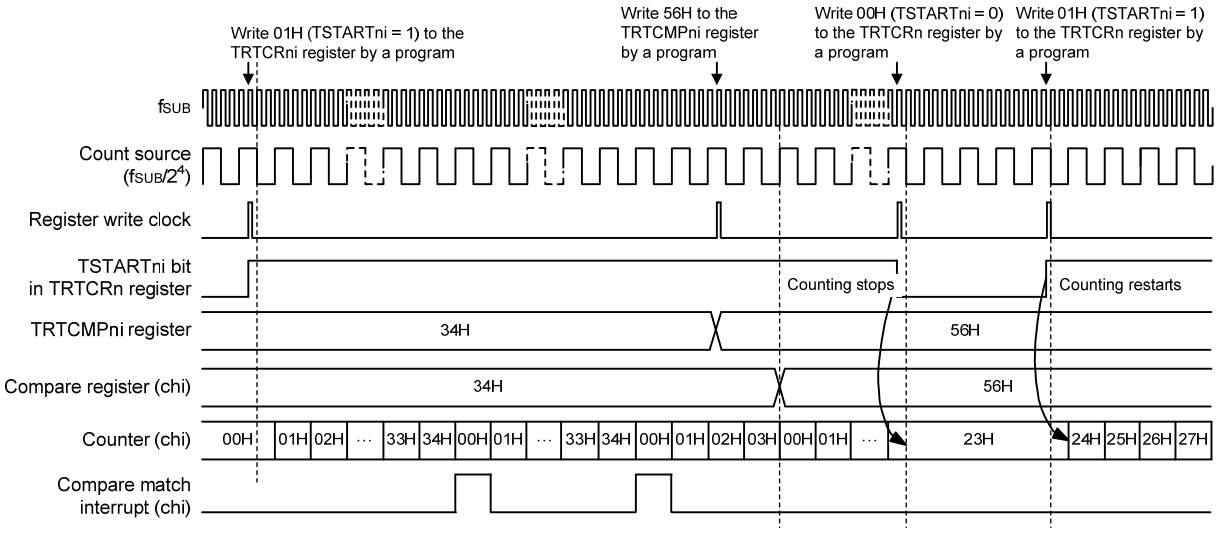
**(2) When $f_{SUB}/2^m$ is selected as count source**

After 1 is written to the TSTARTni (n = 0 or 1, i = 0 or 1) bit in the TRTCRn register, counting starts at the next subsystem clock ($f_{SUB}$) cycle, and then the counter is incremented from 00H to 01H at the next count source ($f_{SUB}/2^m$) cycle. Similarly, after writing 0 in the TSTARTni, the count is stopped after counting with the subsystem clock ($f_{SUB}$). However, the initial 00H count interval at the start of count becomes shorter than 1 cycle of the count source as follows due to the TSTARTni bit write timing and the timing of the next count source.

Minimum: One subsystem clock ($f_{SUB}$) cycle
Maximum: One count source cycle

Figure 11-11 shows the count start/stop timing and Figure 11-12 shows the timing for stop count → set compare register (clear count) → start count. Figure 11-11 and Figure 11-12 show the update timing in 8-bit counter mode, but the operation is performed at the same timing even in 16-bit counter mode.

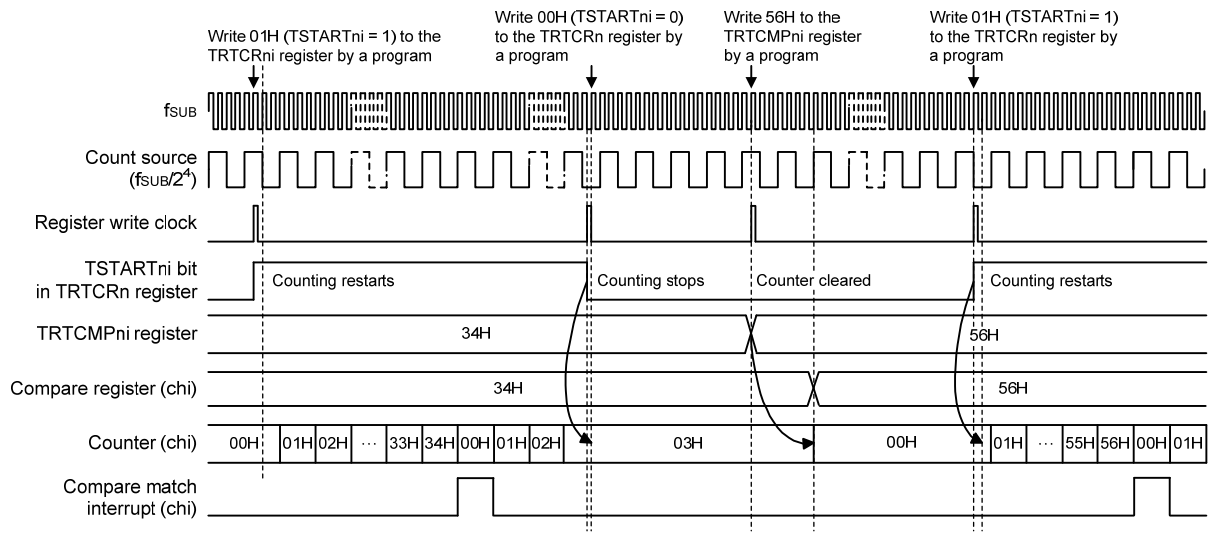
**Figure 11-11. Example of Count Start/Stop Operation (When $f_{SUB}$ Is Selected)**




The TCSMDn bit in the TRTCRn register is set to 0 (8-bit counter operation).

**Remark** n = 0 or 1, i = 0 or 1

**Figure 11-12. Example of Stopping Counting → Clearing the Counter → Starting Counting (When $f_{SUB}/2^m$ Is Selected)**




The TCSMDn bit in the TRTCRn register is set to 0 (8-bit counter operation).

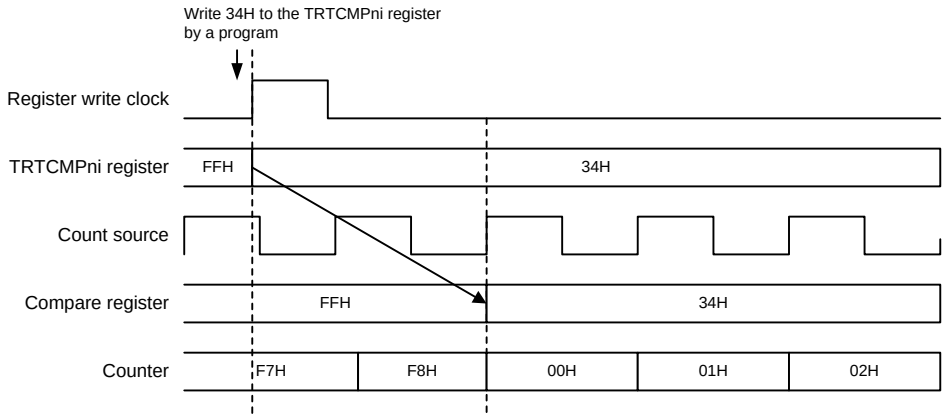
**Remark** n = 0 or 1, i = 0 or 1

### 11.4.4 Timing of updating compare register values

The timing of updating the value of the TRTCMPni (n = 0 or 1, i = 0 or 1) register does not change, regardless of the value of the TSTARTni bit in the TRTCRn register. After write access to TRTCMPni, it is stored in the compare register after 2 cycles of the count source. The counter is cleared (with 00H for 8-bit counter mode, 0000H for 16-bit counter mode) when storing in the compare register.

Figure 11-13 shows the timing of rewriting the compare value. This figure shows the update timing in 8-bit counter mode, but the operation is performed at the same timing in 16-bit counter mode.

**Figure 11-13. Timing of Rewriting the Compare Value**




**Remark** n = 0 or 1, i = 0 or 1

## 11.5 Notes on 8-bit Interval Timer

### 11.5.1 Changing the operating mode and clock settings

The settings of bits TCSMDn and TCKni (n = 0 or 1, i = 0 or 1) must be changed while the TSTARTni bit in the TRTCRn register is 0 (counting is stopped). After the value of the TSTARTni bit is changed from 1 to 0 (counting is stopped), allow at least one $f_{SUB}$ cycle to elapse before accessing the registers (TRTCRn and TRTMDn) associated with the 8-bit interval timer.

### 11.5.2 Accessing compare registers

Do not write to the same compare register (TRTCMPn0, TRTCMPn1, or TRTCMPn) successively. When writing successively, allow at least two count source cycles between writes.

Writing to the compare register (TRTCMPn0, TRTCMPn1, or TRTCMPn) must be executed after the 8-bit interval timer clock enable bit (TCLKENn) is set to1 while the count source is oscillating.

### 11.5.3 8-bit interval timer setting procedure

To supply the clock, set the 8-bit interval timer clock enable bit (TCLKENn) in the 8-bit interval timer control register (TRTCRn) to 1 and then set the TSTARTni bit. (Do not set bits TCLKENn and TSTARTni at the same time.)

To stop the clock, set TSTARTni to 0 and then allow at least one $f_{SUB}$ cycle to elapse before setting the TCLKENn bit to 0.

# CHAPTER 12 CLOCK OUTPUT/BUZZER OUTPUT CONTROLLER

## 12.1 Functions of Clock Output/Buzzer Output Controller

The clock output controller is intended for clock output for supply to peripheral ICs. Buzzer output is a function to output a square wave of buzzer frequency.

One pin can be used to output a clock or buzzer sound.

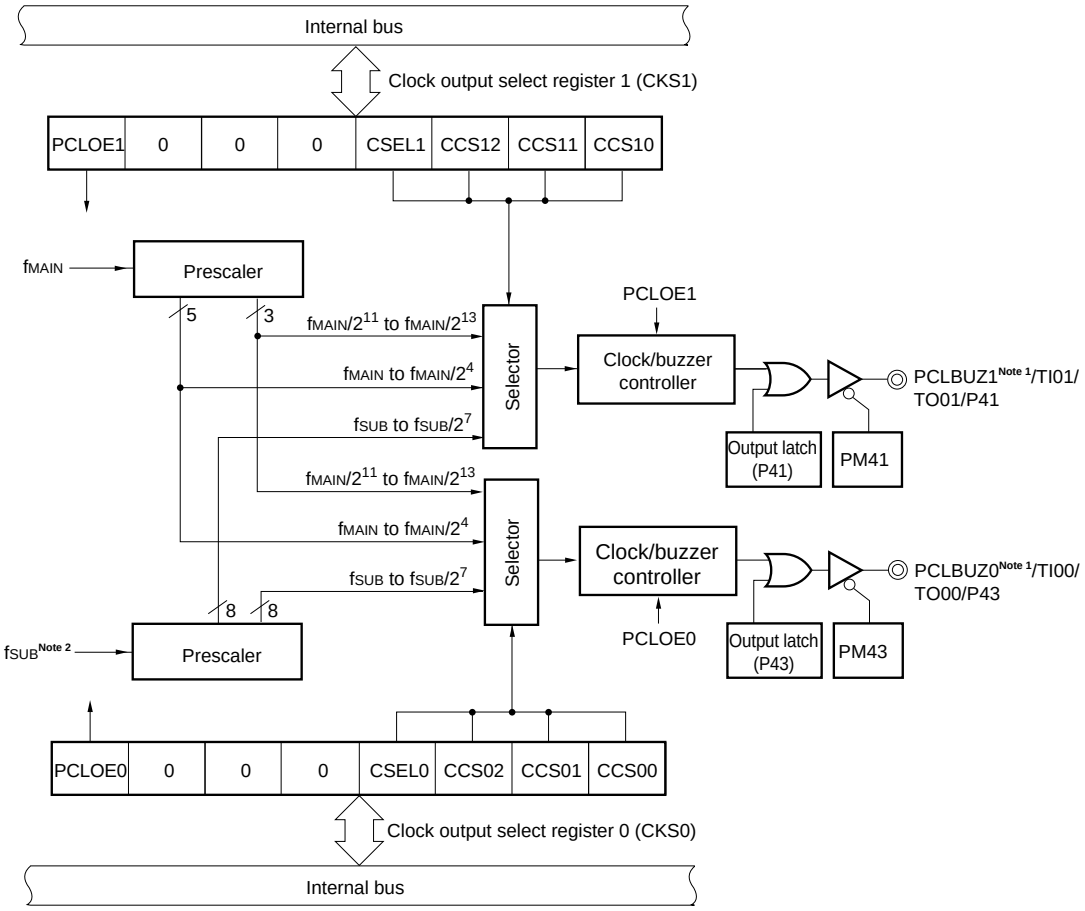
The PCLBUZn pin outputs a clock selected by clock output select register n (CKSn).

Figure 12-1 shows the block diagram of clock output/buzzer output controller.

**Caution It is not possible to output the subsystem clock ($f_{SUB}$) from the PCLBUZn pin while the RTCLPC bit of the subsystem clock supply mode control register (OSMC), is set to 1 and moreover while HALT mode is set with the subsystem clock ($f_{SUB}$) selected as CPU clock.**

**Remark** n = 0, 1

**Figure 12-1. Block Diagram of Clock Output/Buzzer Output Controller**




**Notes 1.** For output frequencies available from PCLBUZ0 and PCLBUZ1, see **37.4 AC Characteristics**.

**2.** Selecting $f_{SUB}$ as the output clock of the clock output/buzzer output controller is prohibited when the WUTMMCK0 bit of the OSMC register is set to 1.

**Remark** The clock output/buzzer output pins in above diagram shows the information with PIOR3 = 0.

## 12.2 Configuration of Clock Output/Buzzer Output Controller

The clock output/buzzer output controller includes the following hardware.

**Table 12-1. Configuration of Clock Output/Buzzer Output Controller**

| Item | Configuration |
|---|---|
| Control registers | Clock output select registers n (CKSn)<br>Port mode register 3, 4 (PM3, PM4)<br>Port register 3, 4 (P3, P4) |

## 12.3 Registers Controlling Clock Output/Buzzer Output Controller

### 12.3.1 Clock output select registers n (CKSn)

These registers set output enable/disable for clock output or for the buzzer frequency output pin (PCLBUZn), and set the output clock.

Select the clock to be output from the PCLBUZn pin by using the CKSn register.

The CKSn register are set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H.

**Figure 12-2. Format of Clock Output Select Register n (CKSn)**

Address: FFFA5H (CKS0), FFFA6H (CKS1) After reset: 00H R/W

| Symbol | <7> | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CKSn | PCLOEn | 0 | 0 | 0 | CSELn | CCSn2 | CCSn1 | CCSn0 |

| PCLOEn | PCLBUZn pin output enable/disable specification |
|---|---|
| 0 | Output disable (default) |
| 1 | Output enable |

| CSELn | CCSn2 | CCSn1 | CCSn0 | PCLBUZn pin output clock selection | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | $f_{MAIN}$ = 5 MHz | $f_{MAIN}$ = 10 MHz | $f_{MAIN}$ = 20 MHz | $f_{MAIN}$ = 24 MHz |
| 0 | 0 | 0 | 0 | $f_{MAIN}$ | 5 MHz | 10 MHz Note 1 | Setting prohibited Note 1 | Setting prohibited Note 1 |
| 0 | 0 | 0 | 1 | $f_{MAIN}/2$ | 2.5 MHz | 5 MHz | 10 MHz Note 1 | 12 MHz Note 1 |
| 0 | 0 | 1 | 0 | $f_{MAIN}/2^2$ | 1.25 MHz | 2.5 MHz | 5 MHz | 6 MHz |
| 0 | 0 | 1 | 1 | $f_{MAIN}/2^3$ | 625 kHz | 1.25 MHz | 2.5 MHz | 3 MHz |
| 0 | 1 | 0 | 0 | $f_{MAIN}/2^4$ | 312.5 kHz | 625 kHz | 1.25 MHz | 1.5 MHz |
| 0 | 1 | 0 | 1 | $f_{MAIN}/2^{11}$ | 2.44 kHz | 4.88 kHz | 9.76 kHz | 11.7 kHz |
| 0 | 1 | 1 | 0 | $f_{MAIN}/2^{12}$ | 1.22 kHz | 2.44 kHz | 4.88 kHz | 5.86 kHz |
| 0 | 1 | 1 | 1 | $f_{MAIN}/2^{13}$ | 610 Hz | 1.22 kHz | 2.44 kHz | 2.93 kHz |
| 1 | 0 | 0 | 0 | $f_{SUB}$ Note 2 | 32.768 kHz | | | |
| 1 | 0 | 0 | 1 | $f_{SUB}/2$ Note 2 | 16.384 kHz | | | |
| 1 | 0 | 1 | 0 | $f_{SUB}/2^2$ Note 2 | 8.192 kHz | | | |
| 1 | 0 | 1 | 1 | $f_{SUB}/2^3$ Note 2 | 4.096 kHz | | | |
| 1 | 1 | 0 | 0 | $f_{SUB}/2^4$ Note 2 | 2.048 kHz | | | |
| 1 | 1 | 0 | 1 | $f_{SUB}/2^5$ Note 2 | 1.024 kHz | | | |
| 1 | 1 | 1 | 0 | $f_{SUB}/2^6$ Note 2 | 512 Hz | | | |
| 1 | 1 | 1 | 1 | $f_{SUB}/2^7$ Note 2 | 256 Hz | | | |

**Notes 1.** Use the output clock within a range of 16 MHz. See **37.4 AC Characteristics** for details.

**2.** Selecting $f_{SUB}$ as the output clock of the clock output/buzzer output controller is prohibited when the WUTMMCK0 bit of the OSMC register is set to 1.

**Cautions 1. Change the output clock after disabling clock output (PCLOEn = 0).**

**2. To shift to STOP mode when the main system clock is selected (CSELn = 0), set PCLOEn = 0 before executing the STOP instruction. When the subsystem clock is selected (CSELn = 1), PCLOEn = 1 can be set because the clock can be output in STOP mode.**

**3. It is not possible to output the subsystem clock ($f_{SUB}$) from the PCLBUZn pin while the RTCLPC bit of the subsystem clock supply mode control register (OSMC), is set to 1 and moreover while HALT mode is set with the subsystem clock ($f_{SUB}$) selected as CPU clock.**

**Remarks 1.** n = 0, 1

**2.** $f_{MAIN}$: Main system clock frequency

$f_{SUB}$: Subsystem clock frequency

### 12.3.2 Registers controlling port functions of pins to be used for clock or buzzer output

Using a port pin for clock or buzzer output requires setting of the registers that control the port functions multiplexed on the target pin (port mode register (PMxx), port register (Pxx)). For details, see **4.3.1 Port mode registers (PMxx)** and **4.3.2 Port registers (Pxx)**.

Specifically, using a port pin with a multiplexed clock or buzzer output function (e.g. P43/TI00/TO00/PCLBUZ0, P41/TI01/TO01/PCLBUZ1) for clock or buzzer output, requires setting the corresponding bits in the port mode register (PMxx) and port register (Pxx) to 0.

Example: When P43/TI00/TO00/PCLBUZ0 is to be used for clock or buzzer output
Set the PM43 bit of port mode register 4 to 0.
Set the P43 bit of port register 4 to 0.
Configuration to stop use of timer array unit channel 0.

## 12.4 Operations of Clock Output/Buzzer Output Controller

One pin can be used to output a clock or buzzer sound.

The PCLBUZ0 pin outputs a clock/buzzer selected by the clock output select register 0 (CKS0).

The PCLBUZ1 pin outputs a clock/buzzer selected by the clock output select register 1 (CKS1).

### 12.4.1 Operation as output pin

The PCLBUZn pin is output as the following procedure.

<1> Set 0 in the bit of the port mode register (PMxx) and port register (Px) which correspond to the port which has a pin used as the PCLBUZ0 pin.

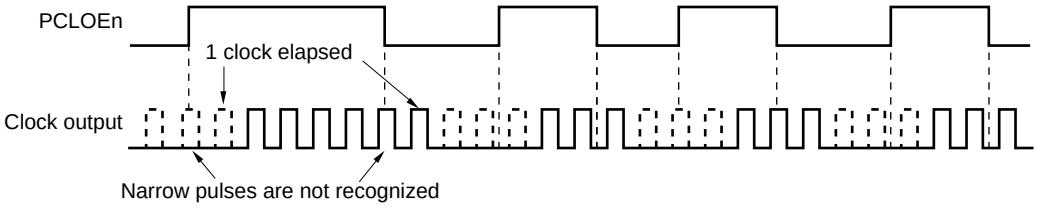
<2> Select the output frequency with bits 0 to 3 (CCSn0 to CCSn2, CSELn) of the clock output select register (CKSn) of the PCLBUZn pin (output in disabled status).

<3> Set bit 7 (PCLOEn) of the CKSn register to 1 to enable clock/buzzer output.

**Remarks 1.** The controller used for outputting the clock starts or stops outputting the clock one clock after enabling or disabling clock output (PCLOEn bit) is switched. At this time, pulses with a narrow width are not output. Figure 12-3 shows enabling or stopping output using the PCLOEn bit and the timing of outputting the clock.

**2.** n = 0, 1

**Figure 12-3. Timing of Outputting Clock from PCLBUZn Pin**




## 12.5 Cautions of Clock Output/Buzzer Output Controller

<R> When the main system clock is selected for the PCLBUZn output (CSEL = 0), if STOP mode is entered within 1.5 clock cycles output from the PCLBUZn pin after the output is disabled (PCLOEn = 0), the PCLBUZn output width becomes shorter.

# CHAPTER 13 WATCHDOG TIMER

## 13.1 Functions of Watchdog Timer

The counting operation of the watchdog timer is set by the option byte (000C0H).

The watchdog timer operates on the low-speed on-chip oscillator clock ($f_{IL}$).

The watchdog timer is used to detect an inadvertent program loop. If a program loop is detected, an internal reset signal is generated.

Program loop is detected in the following cases.

- If the watchdog timer counter overflows
- If a 1-bit manipulation instruction is executed on the watchdog timer enable register (WDTE)
- If data other than “ACH” is written to the WDTE register
- If data is written to the WDTE register during a window close period

When a reset occurs due to the watchdog timer, bit 4 (WDTRF) of the reset control flag register (RESF) is set to 1. For details of the RESF register, see **CHAPTER 25 RESET FUNCTION**.

When 75% + 1/2/$f_{IL}$ of the overflow time is reached, an interval interrupt can be generated.

## 13.2 Configuration of Watchdog Timer

The watchdog timer includes the following hardware.

**Table 13-1. Configuration of Watchdog Timer**

| Item | Configuration |
|---|---|
| Counter | Internal counter (17 bits) |
| Control register | Watchdog timer enable register (WDTE) |

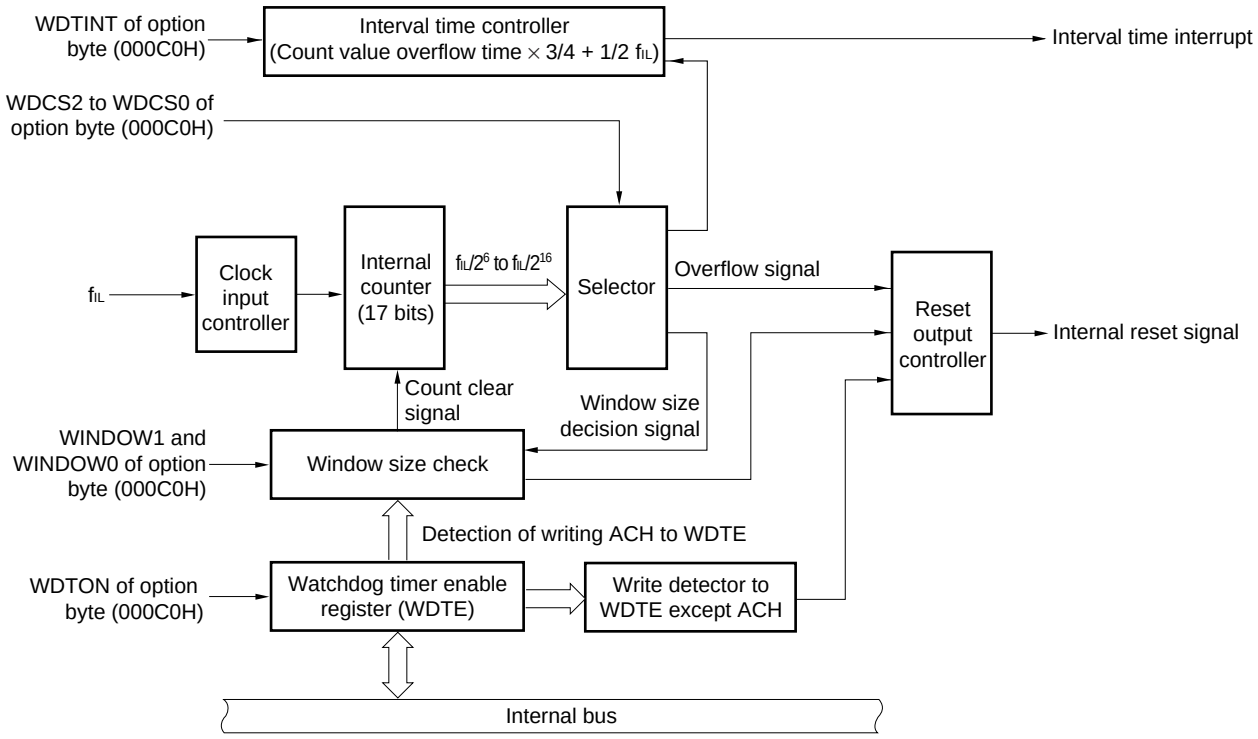
How the counter operation is controlled, overflow time, window open period, and interval interrupt are set by the option byte.

**Table 13-2. Setting of Option Bytes and Watchdog Timer**

| Setting of Watchdog Timer | 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 option byte, see **CHAPTER 32 OPTION BYTE**.

**Figure 13-1. Block Diagram of Watchdog Timer**




**Remark** $f_{IL}$: Low-speed on-chip oscillator clock

## 13.3 Register Controlling Watchdog Timer

The watchdog timer is controlled by the watchdog timer enable register (WDTE).

### 13.3.1 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.

Reset signal generation sets this register to 9AH or 1AH[Note].

**Figure 13-2. Format of Watchdog Timer Enable Register (WDTE)**

Address: FFFABH After reset: 9AH/1AH[Note] R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| WDTE | | | | | | | | |

**Note** The WDTE register reset value differs depending on the WDTON bit setting value of the option byte (000C0H). To operate watchdog timer, set the WDTON bit to 1.

| WDTON Bit Setting Value | WDTE Register Reset Value |
|---|---|
| 0 (watchdog timer count operation disabled) | 1AH |
| 1 (watchdog timer count operation enabled) | 9AH |

**Cautions 1. If a value other than “ACH” is written to the WDTE register, an internal reset signal is generated.**

**2. If a 1-bit memory manipulation instruction is executed for the WDTE register, an internal reset signal is generated.**

**3. The value read from the WDTE register is 9AH/1AH (this differs from the written value (ACH)).**

## 13.4 Operation of Watchdog Timer

### 13.4.1 Controlling operation of watchdog timer

1. When the watchdog timer is used, its operation is specified by the option byte (000C0H).

   - Enable counting operation of the watchdog timer by setting bit 4 (WDTON) of the option byte (000C0H) to 1 (the counter starts operating after a reset release) (for details, see **CHAPTER 32**).

| WDTON | Watchdog Timer Counter |
|---|---|
| 0 | Counter operation disabled (counting stopped after reset) |
| 1 | Counter operation enabled (counting started after reset) |

   - Set an overflow time by using bits 3 to 1 (WDCS2 to WDCS0) of the option byte (000C0H) (for details, see **13.4.2** and **CHAPTER 32**).
   - Set a window open period by using bits 6 and 5 (WINDOW1 and WINDOW0) of the option byte (000C0H) (for details, see **13.4.3** and **CHAPTER 32**).

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 option byte, the watchdog timer is cleared and starts counting again.
4. After that, write the WDTE register the second time or later after a reset release during the window open period. If the WDTE register is written during a window close period, an internal reset signal is generated.
5. If the overflow time expires without “ACH” written to the WDTE register, an internal reset signal is generated.
   An internal reset signal is generated in the following cases.

   - If a 1-bit manipulation instruction is executed on the WDTE register
   - If data other than “ACH” is written to the WDTE register

**Cautions 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.**

**2. After “ACH” is written to the WDTE register, an error of up to 2 clocks ($f_{IL}$) may occur before the watchdog timer is cleared.**

**3. The watchdog timer can be cleared immediately before the count value overflows.**

**Cautions 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 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.**

### 13.4.2 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 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 13-3. Setting of Overflow Time of Watchdog Timer**

| WDCS2 | WDCS1 | WDCS0 | Overflow Time of Watchdog Timer ($f_{IL}$ = 17.25 kHz (MAX.)) |
|---|---|---|---|
| 0 | 0 | 0 | $2^6/f_{IL}$ (3.71 ms) |
| 0 | 0 | 1 | $2^7/f_{IL}$ (7.42 ms) |
| 0 | 1 | 0 | $2^8/f_{IL}$ (14.84 ms) |
| 0 | 1 | 1 | $2^9/f_{IL}$ (29.68 ms) |
| 1 | 0 | 0 | $2^{11}/f_{IL}$ (118.72 ms) |
| 1 | 0 | 1 | $2^{13}/f_{IL}$ (474.89 ms) |
| 1 | 1 | 0 | $2^{14}/f_{IL}$ (949.79 ms) |
| 1 | 1 | 1 | $2^{16}/f_{IL}$ (3799.18 ms) |

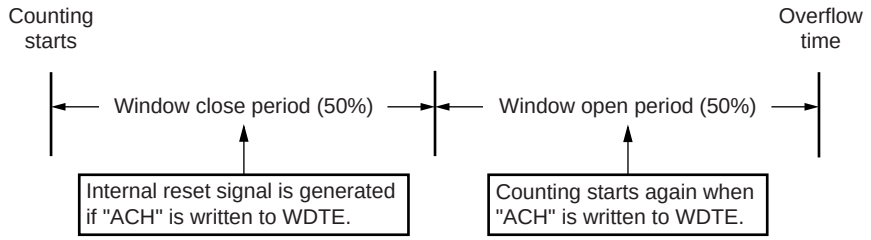
**Remark** $f_{IL}$: Low-speed on-chip oscillator clock frequency

### 13.4.3 Setting window open period of watchdog timer

Set the window open period of the watchdog timer by using bits 6 and 5 (WINDOW1, WINDOW0) of the 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**: If 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 13-4. Setting Window Open Period of Watchdog Timer**

| WINDOW1 | WINDOW0 | Window Open Period of Watchdog Timer |
|---|---|---|
| 0 | 0 | Setting prohibited |
| 0 | 1 | 50% |
| 1 | 0 | 75% |
| 1 | 1 | 100% |

**Caution When bit 0 (WDSTBYON) of the option byte (000C0H) = 0, the window open period is 100% regardless of the values of the WINDOW1 and WINDOW0 bits.**

**Remark** If the overflow time is set to $2^9/f_{IL}$, the window close time and open time are as follows.

| | Setting of Window Open Period | | |
|---|---|---|---|
| | 50% | 75% | 100% |
| Window close time | 0 to 20.08 ms | 0 to 10.04 ms | None |
| Window open time | 20.08 to 29.68 ms | 10.04 to 29.68 ms | 0 to 29.68 ms |

<When window open period is 50%>

- Overflow time:
  $2^9/f_{IL}$ (MAX.) = $2^9$/17.25 kHz = 29.68 ms
- Window close time:
  0 to $2^9/f_{IL}$ (MIN.) × (1 − 0.5) = 0 to $2^9$/12.75 kHz × 0.5 = 0 to 20.08 ms
- Window open time:
  $2^9/f_{IL}$ (MIN.) × (1 − 0.5) to $2^9/f_{IL}$ (MAX.) = $2^9$/12.75 kHz × 0.5 to $2^9$/17.25 kHz = 20.08 to 29.68 ms

### 13.4.4 Setting watchdog timer interval interrupt

Depending on the setting of bit 7 (WDTINT) of an option byte (000C0H), an interval interrupt (INTWDTI) can be generated when 75% + 1/2$f_{IL}$ of the overflow time is reached.

**Table 13-5. 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% + 1/2$f_{IL}$ of overflow time 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.

# CHAPTER 14 A/D CONVERTER

The number of analog input channels of the A/D converter differs, depending on the product.

| | 80-pin | 100-pin |
|---|---|---|
| Analog input channels | 4 ch<br>(ANI0 to ANI3) | 6 ch<br>(ANI0 to ANI5) |

## 14.1 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 6 channels of A/D converter analog inputs (ANI0 to ANI5). 10-bit or 8-bit resolution can be selected by the ADTYP bit of the A/D converter mode register 2 (ADM2).

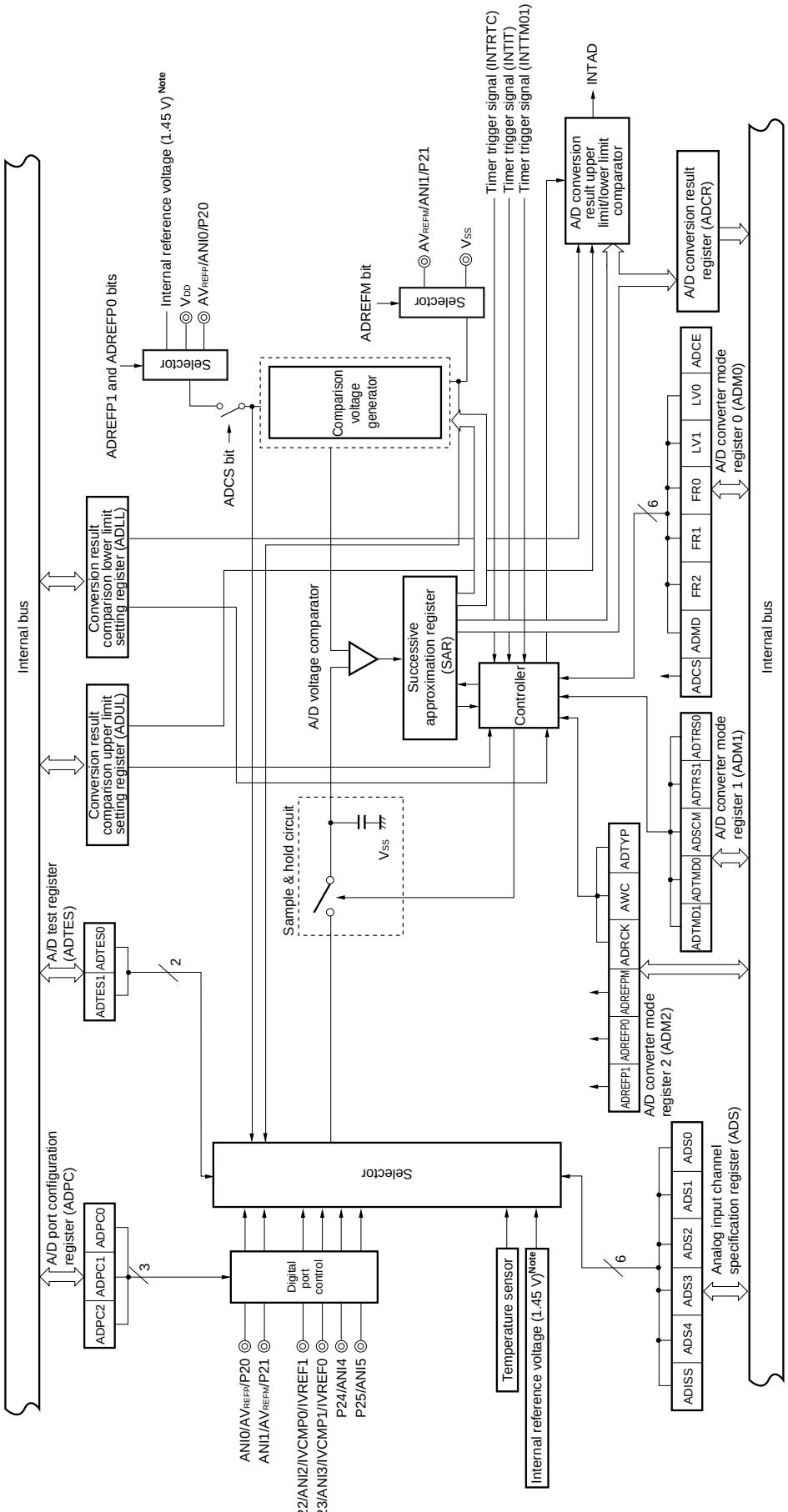
The A/D converter has the following function.

- **10-bit/8-bit resolution A/D conversion**
  10-bit or 8-bit resolution A/D conversion is carried out repeatedly for one analog input channel selected from ANI0 to ANI5. Each time an A/D conversion operation ends, an interrupt request (INTAD) is generated (when in the select mode).

Various A/D conversion modes can be specified by using the mode combinations below.

| | | |
|---|---|---|
| Trigger mode | Software trigger | Conversion is started by software. |
| | Hardware trigger no-wait mode | Conversion is started by detecting a hardware trigger. |
| | Hardware trigger wait mode | The power is turned on by detecting a hardware trigger while the system is off and in the conversion standby state, and conversion is then started automatically after the stabilization wait time passes.<br>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 ANI5 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 | Standard 1 or standard 2 mode | Conversion is done in the operation voltage range of 2.7 V ≤ $V_{DD}$ ≤ 5.5 V. |
| | Low voltage 1 or low voltage 2 mode | Conversion is done in the operation voltage range of 1.9 V ≤ $V_{DD}$ ≤ 5.5 V.<br>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:<br>7 $f_{AD}$ | The sampling time in standard 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:<br>5 $f_{AD}$ | The sampling time in standard 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). |

**Figure 14-1. Block Diagram of A/D Converter**




**Remark** Analog input pin for figure 14-1 when a 100-pin product is used.

**Note** When using an internal reference voltage, it must be used in HS mode.
The minimum operating voltage in HS mode is 2.4 V.
Use an external reference voltage if you need to operate at 2.4 V or less.

## 14.2 Configuration of A/D Converter

The A/D converter includes the following hardware.

**(1) ANI0 to ANI5 pins**

These are the analog input pins of the 6 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 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.

**Remark** $AV_{REF}$: The + side reference voltage of the A/D converter. This can be selected from $AV_{REFP}$, the internal reference voltage (1.45 V), and $V_{DD}$.

**(4) Comparison voltage generator**

The comparison voltage generator generates the comparison voltage input from an analog input pin.

**(5) Successive approximation register (SAR)**

The SAR register is a register that sets 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 stores 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) $AV_{REFP}$ pin**

This pin inputs an external reference voltage ($AV_{REFP}$).

If using $AV_{REFP}$ as the + side 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 ANI5 are converted to digital signals based on the voltage applied between $AV_{REFP}$ and the – side reference voltage ($AV_{REFM}$/$V_{SS}$).

In addition to $AV_{REFP}$, it is possible to select $V_{DD}$ or the internal reference voltage (1.45 V) as the + side reference voltage of the A/D converter.

**(10) $AV_{REFM}$ pin**

This pin inputs an external reference voltage ($AV_{REFM}$). If using $AV_{REFM}$ as the – side reference voltage of the A/D converter, set the ADREFM bit of the ADM2 register to 1.

In addition to $AV_{REFM}$, it is possible to select $V_{SS}$ as the – side reference voltage of the A/D converter.

## 14.3 Registers Controlling A/D Converter

The A/D converter is controlled by the following registers.

- Peripheral enable register 0 (PER0)
- 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)
- A/D port configuration register (ADPC)
- Port mode register 2 (PM2)

### 14.3.1 Peripheral enable register 0 (PER0)

This register is used to enable or disable supplying the clock to the peripheral hardware. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

When the A/D converter is used, be sure to set bit 5 (ADCEN) of this register to 1.

The PER0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 14-2. Format of Peripheral Enable Register 0 (PER0)**

Address: F00F0H After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PER0 | RTCWEN | IRDAEN | ADCEN | IICA0EN | SAU1EN | SAU0EN | 0 | TAU0EN |

| ADCEN | Control of A/D converter input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFR used by the A/D converter cannot be written.<br>• The A/D converter is in the reset status. |
| 1 | Enables input clock supply.<br>• SFR used by the A/D converter can be read/written. |

**Cautions 1. When setting the A/D converter, be sure to set the following registers first while the ADCEN bit is set to 1. If ADCEN = 0, the values of the A/D converter control registers are cleared to their initial values and writing to them is ignored (except for port mode register 2 (PM2) and A/D port configuration register (ADPC)).**

- **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).**

**2. Be sure to clear bit 1 to 0.**

### 14.3.2 A/D converter mode register 0 (ADM0)

This register sets the conversion time for analog input to be A/D converted, and starts/stops conversion.

The ADM0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 14-3. Format of A/D Converter Mode Register 0 (ADM0)**

Address: FFF30H After reset: 00H R/W

| Symbol | <7> | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| ADM0 | ADCS | ADMD | FR2 Note 1 | FR1 Note 1 | FR0 Note 1 | LV1 Note 1 | LV0 Note 1 | ADCE |

| ADCS | A/D conversion operation control |
|---|---|
| 0 | Stops conversion operation<br>[When read]<br>Conversion stopped/standby status |
| 1 | Enables conversion operation<br>[When read]<br>While in the software trigger mode: Conversion operation status<br>While in the hardware trigger wait mode: A/D power supply stabilization wait status + conversion operation status |

| ADMD | Specification of the A/D conversion channel selection mode |
|---|---|
| 0 | Select mode |
| 1 | Scan mode |

| ADCE | A/D voltage comparator operation control Note 2 |
|---|---|
| 0 | Stops A/D voltage comparator operation |
| 1 | Enables A/D voltage comparator operation |

**Notes 1.** For details of the FR2 to FR0, LV1, LV0 bits, and A/D conversion, see **Table 14-3 A/D Conversion Time Selection**.

**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 stabilize. Therefore, when the ADCS bit is set to 1 after 1 μs or more has elapsed from the time ADCE bit is set to 1, the conversion result at that time has priority over the first conversion result. Otherwise, ignore data of the first conversion.

**Cautions 1. Change the ADMD, FR2 to FR0, LV1, LV0, and ADCE bits while conversion is stopped (ADCS = 0, ADCE = 0).**

**2. Do not set ADCS = 1 and ADCE = 0.**

**3. Do not change the ADCE and ADCS bits from 0 to 1 at the same time by using an 8-bit manipulation instruction. Be sure to set these bits in the order described in 14.7 A/D Converter Setup Flowchart.**

**Table 14-1. Settings of ADCS and ADCE Bits**

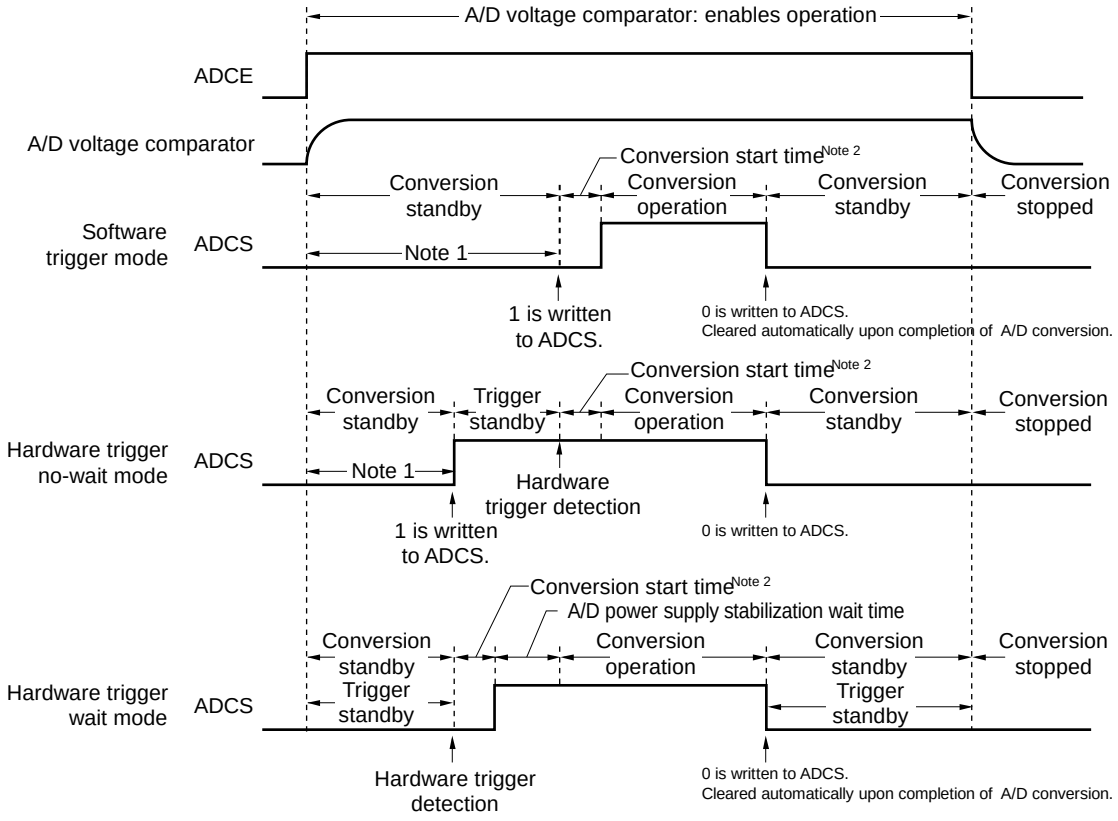
| ADCS | ADCE | A/D Conversion Operation |
|---|---|---|
| 0 | 0 | Conversion stopped state |
| 0 | 1 | Conversion standby state |
| 1 | 0 | Setting prohibited |
| 1 | 1 | Conversion-in-progress state |

**Table 14-2. Setting and Clearing Conditions for ADCS Bit**

| A/D Conversion Mode | | | Set Conditions | Clear Conditions |
|---|---|---|---|---|
| Software trigger | Select mode | Sequential conversion mode | When 1 is written to ADCS | When 0 is written to ADCS |
| | | One-shot conversion mode | | • When 0 is written to ADCS<br>• The bit is automatically cleared to 0 when A/D conversion ends. |
| | Scan mode | Sequential conversion mode | | When 0 is written to ADCS |
| | | One-shot conversion mode | | • When 0 is written to ADCS<br>• The bit is automatically cleared to 0 when conversion ends on the specified four channels. |
| Hardware trigger no-wait mode | Select mode | Sequential conversion mode | | When 0 is written to ADCS |
| | | One-shot conversion mode | | When 0 is written to ADCS |
| | Scan mode | Sequential conversion mode | | When 0 is written to ADCS |
| | | One-shot conversion mode | | When 0 is written to ADCS |
| Hardware trigger wait mode | Select mode | Sequential conversion mode | When a hardware trigger is input | When 0 is written to ADCS |
| | | One-shot conversion mode | | • When 0 is written to ADCS<br>• The bit is automatically cleared to 0 when A/D conversion ends. |
| | Scan mode | Sequential conversion mode | | When 0 is written to ADCS |
| | | One-shot conversion mode | | • When 0 is written to ADCS<br>• The bit is automatically cleared to 0 when conversion ends on the specified four channels. |

<R>

**Figure 14-4. Timing Chart When A/D Voltage Comparator Is Used**




**Notes 1.** While in the software trigger mode or hardware trigger no-wait mode, the time from the rising of the ADCE bit to the falling of the ADCS bit must be 1 μs or longer to stabilize the internal circuit.

**2.** In starting conversion, the longer will take up to following time

| ADM0 | | | Conversion Clock ($f_{AD}$) | Conversion Start Time (Number of $f_{CLK}$ Clock) | |
|---|---|---|---|---|---|
| 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 | |
| 0 | 1 | 0 | $f_{CLK}$/16 | 15 | |
| 0 | 1 | 1 | $f_{CLK}$/8 | 7 | |
| 1 | 0 | 0 | $f_{CLK}$/6 | 5 | |
| 1 | 0 | 1 | $f_{CLK}$/5 | 4 | |
| 1 | 1 | 0 | $f_{CLK}$/4 | 3 | |
| 1 | 1 | 1 | $f_{CLK}$/2 | 1 | |

However, for the second and subsequent conversion in sequential conversion mode, the conversion start time and stabilization wait time for A/D power supply do not occur after a hardware trigger is detected.

**Cautions 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 status.**

**2. While in the one-shot conversion mode of the hardware trigger no-wait mode, the ADCS flag is not automatically cleared to 0 when A/D conversion ends. Instead, 1 is retained.**

**Cautions 3. Only rewrite the value of the ADCE bit when ADCS = 0 (while in the conversion stopped/conversion standby status).**

**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 + conversion start time + A/D conversion time**

**Hardware trigger wait mode: 2 $f_{CLK}$ clock + conversion start time + A/D power supply stabilization wait time + A/D conversion time**

**Remark** $f_{CLK}$: CPU/peripheral hardware clock frequency

**Table 14-3. A/D Conversion Time Selection (1/4)**

(1) When there is no A/D power supply stabilization wait time

Normal mode 1, 2 (software trigger mode/hardware trigger no-wait mode)

| A/D Converter Mode Register 0 (ADM0) | | | | | Mode | Conversion Clock ($f_{AD}$) | Number of Conversion Clock Note | Conversion Time | Conversion Time Selection 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 | $f_{CLK}$ = 24 MHz |
| 0 | 0 | 0 | 0 | 0 | Normal 1 | $f_{CLK}$/64 | 19 $f_{AD}$ | 1216/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | |
| 0 | 0 | 1 | | | | $f_{CLK}$/32 | (number of | 608/$f_{CLK}$ | | | | 38 μs | 25.3333 μs |
| 0 | 1 | 0 | | | | $f_{CLK}$/16 | sampling | 304/$f_{CLK}$ | | | 38 μs | 19 μs | 12.6667 μs |
| 0 | 1 | 1 | | | | $f_{CLK}$/8 | clock: | 152/$f_{CLK}$ | | 38 μs | 19 μs | 9.5 μs | 6.3333 μs |
| 1 | 0 | 0 | | | | $f_{CLK}$/6 | 7 $f_{AD}$) | 114/$f_{CLK}$ | | 28.5 μs | 14.25 μs | 7.125 μs | 4.75 μs |
| 1 | 0 | 1 | | | | $f_{CLK}$/5 | | 95/$f_{CLK}$ | | 23.75 μs | 11.875 μs | 5.938 μs | 3.9583 μs |
| 1 | 1 | 0 | | | | $f_{CLK}$/4 | | 76/$f_{CLK}$ | | 19 μs | 9.5 μs | 4.75 μs | 3.1667 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 $f_{AD}$ | 1088/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | |
| 0 | 0 | 1 | | | | $f_{CLK}$/32 | (number of | 544/$f_{CLK}$ | | | | 34 μs | 22.6667 μs |
| 0 | 1 | 0 | | | | $f_{CLK}$/16 | sampling | 272/$f_{CLK}$ | | | 34 μs | 17 μs | 11.3333 μs |
| 0 | 1 | 1 | | | | $f_{CLK}$/8 | clock: | 136/$f_{CLK}$ | | 34 μs | 17 μs | 8.5 μs | 5.6667 μs |
| 1 | 0 | 0 | | | | $f_{CLK}$/6 | 5 $f_{AD}$) | 102/$f_{CLK}$ | | 25.5 μs | 12.75 μs | 6.375 μs | 4.25 μs |
| 1 | 0 | 1 | | | | $f_{CLK}$/5 | | 85/$f_{CLK}$ | | 21.25 μs | 10.625 μs | 5.3125 μs | 3.5417 μs |
| 1 | 1 | 0 | | | | $f_{CLK}$/4 | | 68/$f_{CLK}$ | | 17 μs | 8.5 μs | 4.25 μs | 2.8333 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 ($f_{AD}$).

**Cautions 1. The A/D conversion time must also be within the relevant range of conversion time ($t_{CONV}$) described in 37.6.1 A/D converter characteristics.**

**2. Rewriting the FR2 to FR0, LV1, and LV0 bits to other than the same data while conversion is stopped (ADCS = 0, ADCE = 0).**

**3. The above conversion time does not include conversion start time. Conversion start time add in the first conversion. Select conversion time, taking clock frequency errors into consideration.**

**Remark** $f_{CLK}$: CPU/peripheral hardware clock frequency

**Table 14-3. A/D Conversion Time Selection (2/4)**

(2) When there is no A/D power supply stabilization wait time

Low-voltage mode 1, 2 (software trigger mode/hardware trigger no-wait mode)

| A/D Converter Mode Register 0 (ADM0) | | | | | Mode | Conversion Clock ($f_{AD}$) | Number of Conversion Clock[Note 1] | Conversion Time | Conversion Time Selection at 10-Bit Resolution | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | 1.9 V ≤ $V_{DD}$ ≤ 5.5 V | | | **Note 2** | **Note 3** |
| FR2 | FR1 | FR0 | LV1 | LV0 | | | | | $f_{CLK}$ = 1 MHz | $f_{CLK}$ = 4 MHz | $f_{CLK}$ = 8 MHz | $f_{CLK}$ = 16 MHz | $f_{CLK}$ = 24 MHz |
| 0 | 0 | 0 | 1 | 0 | Low-voltage 1 | $f_{CLK}$/64 | 19 $f_{AD}$ (number of sampling clock: 7 $f_{AD}$) | 1216/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | |
| 0 | 0 | 1 | | | | $f_{CLK}$/32 | | 608/$f_{CLK}$ | | | | 38 µs | 25.3333 µs |
| 0 | 1 | 0 | | | | $f_{CLK}$/16 | | 304/$f_{CLK}$ | | | 38 µs | 19 µs | 12.6667 µs |
| 0 | 1 | 1 | | | | $f_{CLK}$/8 | | 152/$f_{CLK}$ | | 38 µs | 19 µs | 9.5 µs | 6.3333 µs |
| 1 | 0 | 0 | | | | $f_{CLK}$/6 | | 114/$f_{CLK}$ | | 28.5 µs | 14.25 µs | 7.125 µs | 4.75 µs |
| 1 | 0 | 1 | | | | $f_{CLK}$/5 | | 95/$f_{CLK}$ | | 23.75 µs | 11.875 µs | 5.938 µs | 3.9587 µs |
| 1 | 1 | 0 | | | | $f_{CLK}$/4 | | 76/$f_{CLK}$ | | 19 µs | 9.5 µs | 4.75 µs | 3.1667 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 $f_{AD}$ (number of sampling clock: 5 $f_{AD}$) | 1088/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | |
| 0 | 0 | 1 | | | | $f_{CLK}$/32 | | 544/$f_{CLK}$ | | | | 34 µs | 22.6667 µs |
| 0 | 1 | 0 | | | | $f_{CLK}$/16 | | 272/$f_{CLK}$ | | | 34 µs | 17 µs | 11.3333 µs |
| 0 | 1 | 1 | | | | $f_{CLK}$/8 | | 136/$f_{CLK}$ | | 34 µs | 17 µs | 8.5 µs | 5.6667 µs |
| 1 | 0 | 0 | | | | $f_{CLK}$/6 | | 102/$f_{CLK}$ | | 25.5 µs | 12.75 µs | 6.375 µs | 4.25 µs |
| 1 | 0 | 1 | | | | $f_{CLK}$/5 | | 85/$f_{CLK}$ | | 21.25 µs | 10.625 µs | 5.3125 µs | 3.5417 µs |
| 1 | 1 | 0 | | | | $f_{CLK}$/4 | | 68/$f_{CLK}$ | | 17 µs | 8.5 µs | 4.25 µs | 2.8333 µs |
| 1 | 1 | 1 | | | | $f_{CLK}$/2 | | 34/$f_{CLK}$ | 34 µs | 8.5 µs | 4.25 µs | 2.125 µs | Setting prohibited |

**Notes 1.** 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 ($f_{AD}$).

**2.** 2.4 V ≤ $V_{DD}$ ≤ 5.5 V

**3.** 2.7 V ≤ $V_{DD}$ ≤ 5.5 V

**Cautions 1. The A/D conversion time must also be within the relevant range of conversion time ($t_{CONV}$) described in 37.6.1 A/D converter characteristics.**

**2. Rewriting the FR2 to FR0, LV1, and LV0 bits to other than the same data while conversion is stopped (ADCS = 0, ADCE = 0).**

**3. The above conversion time does not include conversion start time. Conversion start time add in the first conversion. Select conversion time, taking clock frequency errors into consideration.**

**Remark** $f_{CLK}$: CPU/peripheral hardware clock frequency

**Table 14-3. A/D Conversion Time Selection (3/4)**

(3) When there is A/D power supply stabilization wait time
Normal mode 1, 2 (hardware trigger wait mode[Note 1])

| A/D Converter Mode Register 0 (ADM0) | | | | | Mode | Conversion Clock ($f_{AD}$) | Number of A/D Power Supply Stabilization Wait Cock | Number of Conversion Clock[Note 2] | A/D Power Supply Stabilization Wait Cock + Conversion Time | A/D Power Supply Stabilization Wait Cock + 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 | $f_{CLK}$ = 24 MHz |
| 0 | 0 | 0 | 0 | 0 | Normal 1 | $f_{CLK}$/64 | 8 $f_{AD}$ | 19 $f_{AD}$ (number of sampling clock: 7 $f_{AD}$) | 1728/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | |
| 0 | 0 | 1 | | | | $f_{CLK}$/32 | | | 864/$f_{CLK}$ | | | | 54 µs | 36 µs |
| 0 | 1 | 0 | | | | $f_{CLK}$/16 | | | 432/$f_{CLK}$ | | | 54 µs | 27 µs | 18 µs |
| 0 | 1 | 1 | | | | $f_{CLK}$/8 | | | 216/$f_{CLK}$ | | 54 µs | 27 µs | 13.5 µs | 9 µs |
| 1 | 0 | 0 | | | | $f_{CLK}$/6 | | | 162/$f_{CLK}$ | | 40.5 µs | 20.25 µs | 10.125 µs | 6.75 µs |
| 1 | 0 | 1 | | | | $f_{CLK}$/5 | | | 135/$f_{CLK}$ | | 33.75 µs | 16.875 µs | 8.4375 µs | 5.625 µs |
| 1 | 1 | 0 | | | | $f_{CLK}$/4 | | | 108/$f_{CLK}$ | | 27 µs | 13.5 µs | 6.75 µs | 4.5 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 $f_{AD}$ (number of sampling clock: 5 $f_{AD}$) | 1600/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | |
| 0 | 0 | 1 | | | | $f_{CLK}$/32 | | | 800/$f_{CLK}$ | | | | 50 µs | 33.3333 µs |
| 0 | 1 | 0 | | | | $f_{CLK}$/16 | | | 400/$f_{CLK}$ | | | 50 µs | 25 µs | 16.6667 µs |
| 0 | 1 | 1 | | | | $f_{CLK}$/8 | | | 200/$f_{CLK}$ | | 50 µs | 25 µs | 12.5 µs | 8.3333 µs |
| 1 | 0 | 0 | | | | $f_{CLK}$/6 | | | 150/$f_{CLK}$ | | 37.5 µs | 18.75 µs | 9.375 µs | 6.25 µs |
| 1 | 0 | 1 | | | | $f_{CLK}$/5 | | | 125/$f_{CLK}$ | | 31.25 µs | 15.625 µs | 7.8125 µs | 5.2083 µs |
| 1 | 1 | 0 | | | | $f_{CLK}$/4 | | | 100/$f_{CLK}$ | | 25 µs | 12.5 µs | 6.25 µs | 4.1667 µs |
| 1 | 1 | 1 | | | | $f_{CLK}$/2 | | | 50/$f_{CLK}$ | 50 µs | 12.5 µs | 6.25 µs | 3.125 µs | Setting prohibited |

**Notes 1.** For the second and subsequent conversion in sequential conversion 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 14-3 (1/4)**).

**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 ($f_{AD}$).

**Cautions 1. The A/D conversion time must also be within the relevant range of conversion time ($t_{CONV}$) described in 37.6.1 A/D converter characteristics. Note that the conversion time ($t_{CONV}$) does not include the A/D power supply stabilization wait time.**

**2. Rewriting the FR2 to FR0, LV1, and LV0 bits to other than the same data while conversion is stopped (ADCS = 0, ADCE = 0).**

**3. The above conversion time does not include conversion start time. Conversion start time add in the first conversion. Select conversion time, taking clock frequency errors into consideration.**

**4. When hardware trigger wait mode, specify the conversion time, including the A/D power supply stabilization wait time from the hardware trigger detection.**

**Remark** $f_{CLK}$: CPU/peripheral hardware clock frequency

**Table 14-3. A/D Conversion Time Selection (4/4)**

(4) When there is A/D power supply stabilization wait time

Low-voltage mode 1, 2 (hardware trigger wait mode[Note 1])

| A/D Converter Mode Register 0 (ADM0) | | | | | Mode | Conversion Clock ($f_{AD}$) | Number of A/D power supply Stabilization Wait Cock | Number of Conversion Clock[Note 2] | A/D power Supply Stabilization Wait Cock + Conversion Time | A/D Power Supply Stabilization Wait Cock + Conversion Time at 10-Bit Resolution | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | 1.9 V ≤ $V_{DD}$ ≤ 5.5 V | | | **Note 3** | **Note 4** |
| FR2 | FR1 | FR0 | LV1 | LV0 | | | | | | $f_{CLK}$ = 1 MHz | $f_{CLK}$ = 4 MHz | $f_{CLK}$ = 8 MHz | $f_{CLK}$ = 16 MHz | $f_{CLK}$ = 24 MHz |
| 0 | 0 | 0 | 1 | 0 | Low-voltage 1 | $f_{CLK}$/64 | 2 $f_{AD}$ | 19 $f_{AD}$ (number of sampling clock: 7 $f_{AD}$) | 1344/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | |
| 0 | 0 | 1 | | | | $f_{CLK}$/32 | | | 672/$f_{CLK}$ | | | | 42 µs | 28 µs |
| 0 | 1 | 0 | | | | $f_{CLK}$/16 | | | 336/$f_{CLK}$ | | | 42 µs | 21 µs | 14 µs |
| 0 | 1 | 1 | | | | $f_{CLK}$/8 | | | 168/$f_{CLK}$ | | 42 µs | 21 µs | 10.5 µs | 7 µs |
| 1 | 0 | 0 | | | | $f_{CLK}$/6 | | | 126/$f_{CLK}$ | | 31.25 µs | 15.75 µs | 7.875 µs | 5.25 µs |
| 1 | 0 | 1 | | | | $f_{CLK}$/5 | | | 105/$f_{CLK}$ | | 26.25 µs | 13.125 µs | 6.5625 µs | 4.375 µs |
| 1 | 1 | 0 | | | | $f_{CLK}$/4 | | | 84/$f_{CLK}$ | | 21 µs | 10.5 µs | 5.25 µs | 3.5 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 $f_{AD}$ (number of sampling clock: 5 $f_{AD}$) | 1216/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | |
| 0 | 0 | 1 | | | | $f_{CLK}$/32 | | | 608/$f_{CLK}$ | | | | 38 µs | 25.3333 µs |
| 0 | 1 | 0 | | | | $f_{CLK}$/16 | | | 304/$f_{CLK}$ | | | 38 µs | 19 µs | 12.6667 µs |
| 0 | 1 | 1 | | | | $f_{CLK}$/8 | | | 152/$f_{CLK}$ | | 38 µs | 19 µs | 9.5 µs | 6.3333 µs |
| 1 | 0 | 0 | | | | $f_{CLK}$/6 | | | 114/$f_{CLK}$ | | 28.5 µs | 14.25 µs | 7.125 µs | 4.75 µs |
| 1 | 0 | 1 | | | | $f_{CLK}$/5 | | | 95/$f_{CLK}$ | | 23.75 µs | 11.875 µs | 5.938 µs | 3.9583 µs |
| 1 | 1 | 0 | | | | $f_{CLK}$/4 | | | 76/$f_{CLK}$ | | 19 µs | 9.5 µs | 4.75 µs | 3.1667 µs |
| 1 | 1 | 1 | | | | $f_{CLK}$/2 | | | 38/$f_{CLK}$ | 38 µs | 9.5 µs | 4.75 µs | 2.375 µs | Setting prohibited |

**Notes 1.** For the second and subsequent conversion in sequential conversion 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 14-3 (2/4)**).

**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 ($f_{AD}$).

**3.** 2.4 V ≤ $V_{DD}$ ≤ 5.5 V

**4.** 2.7 V ≤ $V_{DD}$ ≤ 5.5 V

**Cautions 1. The A/D conversion time must also be within the relevant range of conversion time ($t_{CONV}$) described in 37.6.1 A/D converter characteristics. Note that the conversion time ($t_{CONV}$) does not include the A/D power supply stabilization wait time.**

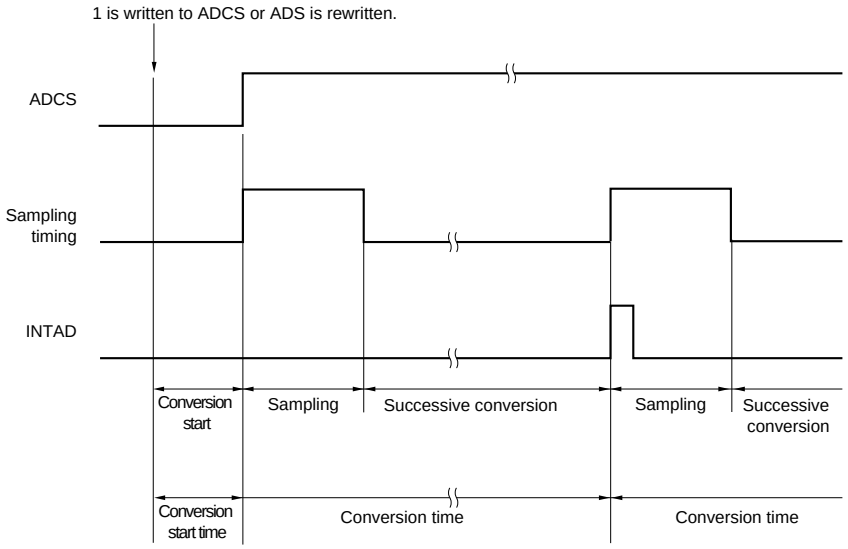
**2. Rewriting the FR2 to FR0, LV1, and LV0 bits to other than the same data while conversion is stopped (ADCS = 0, ADCE = 0).**

**3. The above conversion time does not include conversion start time. Conversion start time add in the first conversion. Select conversion time, taking clock frequency errors into consideration.**

**4. When hardware trigger wait mode, specify the conversion time, including the A/D power supply stabilization wait time from the hardware trigger detection.**

**Remark** $f_{CLK}$: CPU/peripheral hardware clock frequency

**Figure 14-5. A/D Converter Sampling and A/D Conversion Timing (Example for Software Trigger Mode)**




### 14.3.3 A/D converter mode register 1 (ADM1)

This 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.

Reset signal generation clears this register to 00H.

**Figure 14-6. Format of A/D Converter Mode Register 1 (ADM1)**

Address: FFF32H After reset: 00H 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 | End of timer channel 01 count or capture interrupt signal (INTTM01) |
| 0 | 1 | Setting prohibited |
| 1 | 0 | Real-time clock 2 interrupt signal (INTRTC) |
| 1 | 1 | 12-bit interval timer interrupt signal (INTIT) |

**Cautions 1. Rewrite the value of the ADM1 register while conversion is stopped (ADCS = 0, ADCE = 0).**

**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 + conversion start time + A/D conversion time**
**Hardware trigger wait mode: 2 $f_{CLK}$ clock + conversion start time + A/D power supply stabilization wait time + A/D conversion time**

**3. In modes other than SNOOZE mode, input of the next INTRTC or INTIT will not be recognized as a valid hardware trigger for up to four $f_{CLK}$ cycles after the first INTRTC or INTIT is input.**

**Remarks 1.** ×: don't care

**2.** $f_{CLK}$: CPU/peripheral hardware clock frequency

### 14.3.4 A/D converter mode register 2 (ADM2)

This register is used to select the + side or – side reference voltage of the A/D converter, check the upper limit and lower limit A/D conversion result values, 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.

Reset signal generation clears this register to 00H.

**Figure 14-7. Format of A/D Converter Mode Register 2 (ADM2) (1/2)**

Address: F0010H After reset: 00H 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 + side reference voltage of the A/D converter |
|---|---|---|
| 0 | 0 | Supplied from $V_{DD}$ Note 2 |
| 0 | 1 | Supplied from P20/$AV_{REFP}$/ANI0 |
| 1 | 0 | Supplied from the internal reference voltage (1.45 V) Note 1 |
| 1 | 1 | Setting prohibited |

- When ADREFP1 or ADREFP0 bit is rewritten, this must be configured in accordance with the following procedures.
  (1) Set ADCE = 0
  (2) Change the values of ADREFP1 and ADREFP0
  (3) Reference voltage stabilization wait time (A)
  (4) Set ADCE = 1
  (5) Reference voltage stabilization wait time (B)
  When ADREFP1 and ADREFP0 are set to 1 and 0, the setting is changed to A = 5 μs, B = 1 μs.
  When ADREFP1 and ADREFP0 are set to 0 and 0 or 0 and 1, A needs no wait and B = 1 μs.
  After (5) stabilization time, start the A/D conversion.
- When ADREFP1 and ADREFP0 are set to 1 and 0, respectively, A/D conversion cannot be performed on the temperature sensor output voltage and internal reference voltage (1.45 V).
  Be sure to perform A/D conversion while ADISS = 0.

| ADREFM | Selection of the – side reference voltage source of the A/D converter |
|---|---|
| 0 | Supplied from $V_{SS}$ |
| 1 | Supplied from P21/$AV_{REFM}$/ANI1 |

**Notes 1.** This setting can be used only in HS (high-speed main) mode.
When using a temperature sensor, be sure to use an internal reference voltage.

**2.** When using reference voltage (+) = $V_{DD}$, take into account the voltage drop due to the effect of the power switching circuit of the battery backup function and use the A/D conversion result. In addition, enter HALT mode during A/D conversion and set $V_{DD}$ port to input.

(**Cautions** are listed on the next page.)

**Cautions 1. Only rewrite the value of the ADM2 register while conversion is stopped (ADCS = 0, ADCE = 0).**

**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 = 1, 0), the A/D converter reference voltage current ($I_{ADREF}$) indicated in 37.3.2 Supply current characteristics will be added.**

**3. When using $AV_{REFP}$ and $AV_{REFM}$, specify ANI0 and ANI1 as the analog input channels and specify input mode by using the port mode register.**

**Figure 14-7. Format of A/D Converter Mode Register 2 (ADM2) (2/2)**

Address: F0010H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | <3> | <2> | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| ADM2 | ADREFP1 | ADREFP0 | ADREFM | 0 | ADRCK | AWC | 0 | ADTYP |

| ADRCK | Checking the upper limit and lower limit conversion result values |
|---|---|
| 0 | The interrupt signal (INTAD) is output when the ADLL register ≤ the ADCR register ≤ the ADUL register (AREA 1). |
| 1 | The interrupt signal (INTAD) is output when the ADCR register < the ADLL register (AREA 2) or the ADUL register < the ADCR register (AREA 3). |
| **Figure 14-8** shows the generation range of the interrupt signal (INTAD) for AREA 1 to AREA 3. | |

| 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).

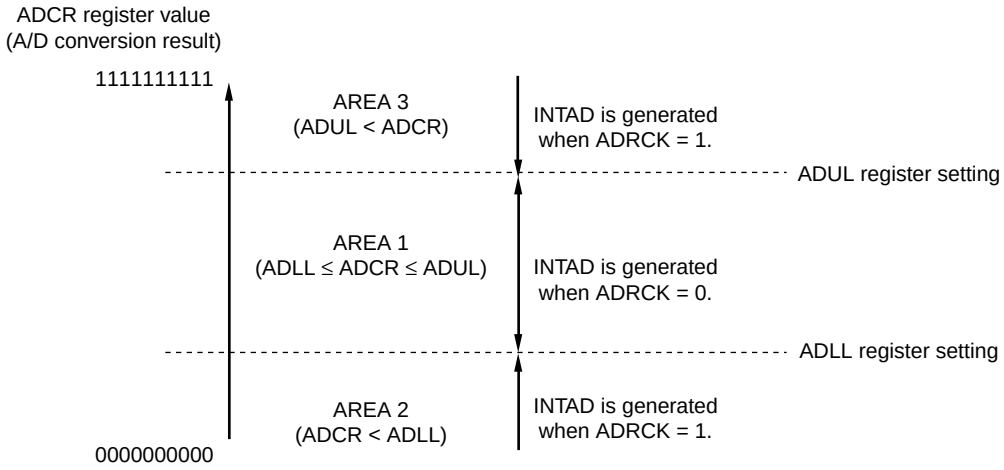
- The SNOOZE mode function can only be specified when the high-speed on-chip oscillator clock is selected for the CPU/peripheral hardware clock ($f_{CLK}$). If any other clock is selected, specifying this mode is prohibited.
- Using the SNOOZE mode function in the software trigger mode or hardware trigger 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] + conversion start time + A/D power supply stabilization wait time + A/D conversion time +2 $f_{CLK}$ clock”
- Even when using SNOOZE mode, 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 or normal operation.

| ADTYP | Selection of the A/D conversion resolution |
|---|---|
| 0 | 10-bit resolution |
| 1 | 8-bit resolution |

**Note** Refer to “Transition time from STOP mode to SNOOZE mode” in **24.3.3 SNOOZE mode**

**Caution Only rewrite the value of the ADM2 register while conversion is stopped (ADCS = 0, ADCE = 0).**

**Figure 14-8. ADRCK Bit Interrupt Signal Generation Range**




**Remark** If INTAD does not occur, the A/D conversion result is not stored in the ADCR or ADCRH register.

### 14.3.5 10-bit A/D conversion result register (ADCR)

This register is a 16-bit register that stores the A/D conversion result. The lower 6 bits are fixed to 0. Each time A/D conversion ends, 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.

Reset signal generation clears this register to 0000H.

**Note** If the A/D conversion result is outside the range specified by using the A/D conversion comparison function (the value specified by the ADRCK bit of the ADM2 register and ADUL/ADLL registers; see Figure 14-8), the result is not stored.

**Figure 14-9. Format of 10-bit A/D Conversion Result Register (ADCR)**

Address: FFF1FH, FFF1EH After reset: 0000H R

| Symbol | FFF1FH | | | | | | | | FFF1EH | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ADCR | | | | | | | | | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Cautions 1. When 8-bit resolution A/D conversion is selected (when the ADTYP bit of A/D converter mode register 2 (ADM2) is 1) and the ADCR register is read, 0 is read from the lower two bits (bits 7 and 6 of the ADCR register).**

**2. When the ADCR register is accessed in 16-bit units, the higher 10 bits of the conversion result are read in order starting at bit 15 of the ADCR register.**

### 14.3.6 8-bit A/D conversion result register (ADCRH)

This register is an 8-bit register that stores 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.

Reset signal generation clears this register to 00H.

**Note** If the A/D conversion result is outside the range specified by using the A/D conversion comparison function (the value specified by the ADRCK bit of the ADM2 register and ADUL/ADLL registers; see Figure 14-8), the result is not stored.

**Figure 14-10. Format of 8-bit A/D Conversion Result Register (ADCRH)**

Address: FFF1FH After reset: 00H R

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ADCRH | | | | | | | | |

**Caution When writing to the A/D converter mode register 0 (ADM0), analog input channel specification register (ADS), and A/D port configuration register (ADPC), the contents of the ADCRH register may become undefined. Read the conversion result following conversion completion before writing to the ADM0, ADS, and ADPC registers. Using timing other than the above may cause an incorrect conversion result to be read.**

### 14.3.7 Analog input channel specification register (ADS)

This 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.

Reset signal generation clears this register to 00H.

**Figure 14-11. Format of Analog Input Channel Specification Register (ADS)**

Address: FFF31H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ADS | ADISS | 0 | 0 | ADS4 | ADS3 | ADS2 | ADS1 | ADS0 |

O Select mode (ADMD = 0)

| ADISS | ADS4 | ADS3 | ADS2 | ADS1 | ADS0 | Analog input channel | Input source |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | ANI0 | P20/ANI0/$AV_{REFP}$ pin |
| 0 | 0 | 0 | 0 | 0 | 1 | ANI1 | P21/ANI1/$AV_{REFM}$ 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 | 1 | 1 | 1 | 0 | 1 | – | Temperature sensor output voltage[Note] |
| 1 | 0 | 0 | 0 | 0 | 1 | – | Internal reference voltage (1.45 V)[Note] |
| Other than above | | | | | | Setting prohibited | |

O Scan mode (ADMD = 1)

| ADS4 | ADS3 | ADS2 | ADS1 | ADS0 | Analog input channel | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | Scan 0 | Scan 1 | Scan 2 | Scan 3 |
| 0 | 0 | 0 | 0 | 0 | ANI0 | ANI1 | ANI2 | ANI3 |
| 0 | 0 | 0 | 0 | 1 | ANI1 | ANI2 | ANI3 | ANI4 |
| 0 | 0 | 0 | 1 | 0 | ANI2 | ANI3 | ANI4 | ANI5 |
| Other than above | | | | | Setting prohibited | | | |

**Note** This setting can be used only in HS (high-speed main) mode.
When using a temperature sensor, be sure to use an internal reference voltage.

**Cautions 1. Be sure to clear bits 5 and 6 to 0.**

**2. Set a channel to be set the analog input by ADPC register in the input mode by using port mode register 2 (PM2).**

**3. Do not set the pin that is set by the A/D port configuration register (ADPC) as digital I/O by the ADS register.**

**4. Rewrite the value of the ADISS bit while conversion is stopped (ADCS = 0, ADCE = 0).**

**5. If using $AV_{REFP}$ as the + side reference voltage of the A/D converter, do not select ANI0 as an A/D conversion channel.**

**6. If using $AV_{REFM}$ as the – side reference voltage of the A/D converter, do not select ANI1 as an A/D conversion channel.**

**Cautions 7. If ADISS is set to 1, the internal reference voltage (1.45 V) cannot be used for the + side reference voltage. After the ADISS bit is set to 1, the initial conversion result cannot be used. For the setting flow, see 14.7.4 Setup when temperature sensor output voltage/internal reference voltage is selected.**

**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. Also, if the ADREFP1 bit is set to 1, the A/D converter reference voltage current ($I_{ADREF}$) indicated in 37.3.2 Supply current characteristics will be added to the current consumption when shifting to HALT mode while the CPU is operating on the main system clock.**

**9. Ignore the conversion result if the corresponding ANI pin does not exist in the product used.**

### 14.3.8 Conversion result comparison upper limit setting register (ADUL)

This register is used to specify the setting for checking the upper limit of the A/D conversion results.

The A/D conversion results and ADUL register value are compared, and interrupt signal (INTAD) generation is controlled in the range specified for the ADRCK bit of A/D converter mode register 2 (ADM2) (shown in **Figure 14-8**).

The ADUL register can be set by an 8-bit memory manipulation instruction.

Reset signal generation sets this register to FFH.

**Figure 14-12. Format of Conversion Result Comparison Upper Limit Setting Register (ADUL)**

Address: F0011H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ADUL | ADUL7 | ADUL6 | ADUL5 | ADUL4 | ADUL3 | ADUL2 | ADUL1 | ADUL0 |

### 14.3.9 Conversion result comparison lower limit setting register (ADLL)

This register is used to specify the setting for checking the lower limit of the A/D conversion results.

The A/D conversion results and ADLL register value are compared, and interrupt signal (INTAD) generation is controlled in the range specified for the ADRCK bit of A/D converter mode register 2 (ADM2) (shown in **Figure 14-8**).

The ADLL register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 14-13. Format of Conversion Result Comparison Lower Limit Setting Register (ADLL)**

Address: F0012H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ADLL | ADLL7 | ADLL6 | ADLL5 | ADLL4 | ADLL3 | ADLL2 | ADLL1 | ADLL0 |

**Cautions 1. When 10-bit resolution A/D conversion is selected, the higher eight bits of the 10-bit A/D conversion result register (ADCR) are compared with the values in the ADUL and ADLL registers.**

**2. Only write new values to the ADUL and ADLL registers while conversion is stopped (ADCS = 0, ADCE = 0).**

**3. The setting of the ADUL registers must be greater than that of the ADLL register.**

### 14.3.10 A/D test register (ADTES)

This register is used to select the + side reference voltage or – side reference voltage of the A/D converter, or the analog input channel (ANIxx) as the target for A/D conversion. When using this register to test the converter, set as follows.

- For zero-scale measurement, select the – side reference voltage as the target for conversion.
- For full-scale measurement, select the + side reference voltage as the target for conversion.

The ADTES register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 14-14. Format of A/D Test Register (ADTES)**

Address: F0013H After reset: 00H 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[Note]/internal reference voltage (1.45 V)[Note] (This is specified using the analog input channel specification register (ADS).) |
| 1 | 0 | The – side reference voltage (selected by the ADREFM bit of the ADM2 register) |
| 1 | 1 | The + side reference voltage (selected by the ADREFP1 or ADREFP0 bit of the ADM2 register) |
| Other than above | | Setting prohibited |

**Note** The temperature sensor output voltage and internal reference voltage (1.45 V) can be selected only in the HS (high-speed main) mode.

**Caution For details of the A/D test function, see CHAPTER 30 SAFETY FUNCTIONS.**

### 14.3.11 Registers controlling port function of analog input pins

Set up the registers for controlling the functions of the ports shared with the analog input pins of the A/D converter (port mode registers (PMxx) and A/D port configuration register (ADPC)).

For details, see **4.3.1 Port mode registers (PMxx)** and **4.3.6 A/D port configuration register (ADPC)**.

When using the ANI0 to ANI5 pins for analog input of the A/D converter, set the port mode register (PMxx) bit corresponding to each port to 1 and select analog input through the A/D port configuration register (ADPC).

## 14.4 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 circuit.

<2> When sampling has been done for a certain time, the sample & hold 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 preset value of bit 9, 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> Upon completion of the comparison of 10 bits, an effective digital result value remains in the SAR register, and the result value is transferred to the A/D conversion result register (ADCR, ADCRH) and then latched[Note 1].
At the same time, the A/D conversion end interrupt request (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.

**Notes 1.** If the A/D conversion result is outside the A/D conversion result range specified by the ADRCK bit and the ADUL and ADLL registers (see **Figure 14-8**), the A/D conversion result interrupt request signal is not generated and no A/D conversion results are stored in the ADCR and ADCRH registers.

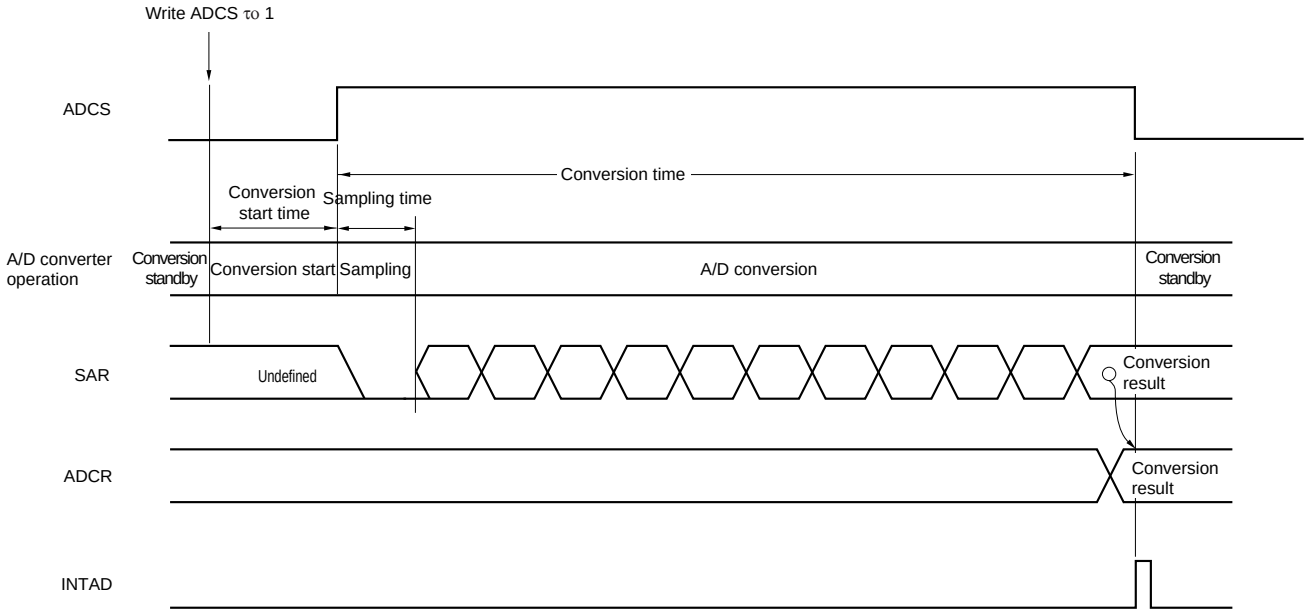
**2.** While in the sequential conversion mode, the ADCS flag is not automatically cleared to 0. This flag is not automatically cleared to 0 while in the one-shot conversion mode of the hardware trigger no-wait mode, either. Instead, 1 is retained.

**Remarks 1.** Two types of the A/D conversion result registers are available.

- ADCR register (16 bits): Store 10-bit A/D conversion value
- ADCRH register (8 bits): Store 8-bit A/D conversion value

**2.** $AV_{REF}$: The + side reference voltage of the A/D converter. This can be selected from $AV_{REFP}$, the internal reference voltage (1.45 V), and $V_{DD}$.

**Figure 14-15. Conversion Operation of A/D Converter (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 bit 7 (ADCS) of the A/D converter mode register 0 (ADM0) to 0.

When the value of the analog input channel specification register (ADS) is rewritten or overwritten during conversion, the current A/D conversion is interrupted, and A/D conversion is performed on the analog input newly specified in the ADS register. The partially converted data is discarded.

Reset signal generation clears the A/D conversion result register (ADCR, ADCRH) to 0000H or 00H.

## 14.5 Input Voltage and Conversion Results

The relationship between the analog input voltage input to the analog input pins (ANI0 to ANI5) 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

$$\left(\frac{ADCR}{64} - 0.5\right) \times \frac{AV_{REF}}{1024} \leq V_{AIN} < \left(\frac{ADCR}{64} + 0.5\right) \times \frac{AV_{REF}}{1024}$$

where, INT( ): Function which returns integer part of value in parentheses

$V_{AIN}$: Analog input voltage

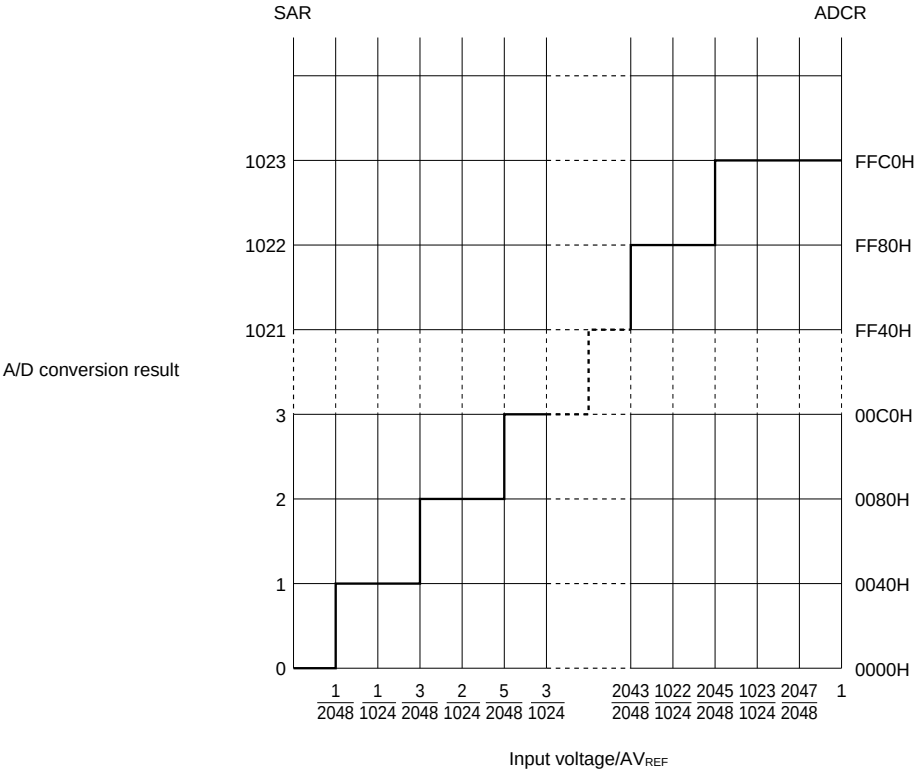
$AV_{REF}$: $AV_{REF}$ pin voltage

ADCR: A/D conversion result register (ADCR) value

SAR: Successive approximation register

Figure 14-16 shows the relationship between the analog input voltage and the A/D conversion result.

**Figure 14-16. Relationship Between Analog Input Voltage and A/D Conversion Result**




**Remark** $AV_{REF}$: The + side reference voltage of the A/D converter. This can be selected from $AV_{REFP}$, the internal reference voltage (1.45 V), and $V_{DD}$.

## 14.6 A/D Converter Operation Modes

The operation of each A/D converter mode is described below. In addition, the procedure for specifying each mode is described in **14.7 A/D Converter Setup Flowchart**.

### 14.6.1 Software trigger mode (select mode, sequential conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.

<2> After the software counts up to the stabilization wait time (1 μs), the ADCS bit of the ADM0 register is set to 1 to perform the A/D conversion of the analog input specified by the analog input channel specification register (ADS).

<3> When A/D conversion ends, the conversion result is stored in the A/D conversion result register (ADCR, ADCRH), and the A/D conversion end interrupt request signal (INTAD) is generated. After A/D conversion ends, the next A/D conversion immediately starts.

<4> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.

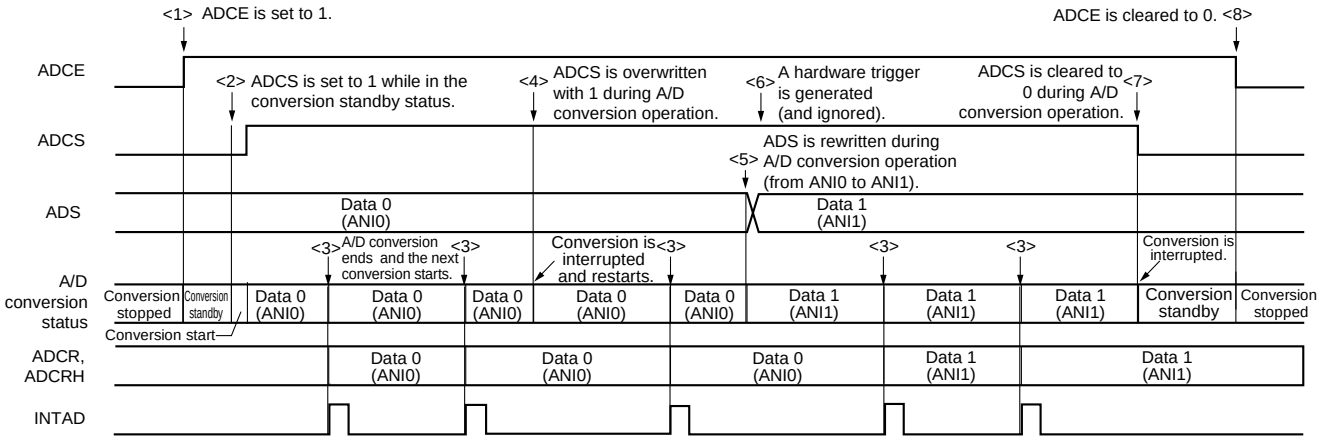
<5> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the analog input respecified by the ADS register. The partially converted data is discarded.

<6> Even if a hardware trigger is input during conversion operation, A/D conversion does not start.

<7> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status.

<8> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCE = 0, specifying 1 for ADCS is ignored and A/D conversion does not start.

**Figure 14-17. Example of Software Trigger Mode (Select Mode, Sequential Conversion Mode) Operation Timing**



### 14.6.2 Software trigger mode (select mode, one-shot conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.

<2> After the software counts up to the stabilization wait time (1 μs), the ADCS bit of the ADM0 register is set to 1 to perform the A/D conversion of the analog input specified by the analog input channel specification register (ADS).

<3> When A/D conversion ends, the conversion result is stored in the A/D conversion result register (ADCR, ADCRH), and the A/D conversion end interrupt request signal (INTAD) is generated.

<4> After A/D conversion ends, the ADCS bit is automatically cleared to 0, and the system enters the A/D conversion standby status.

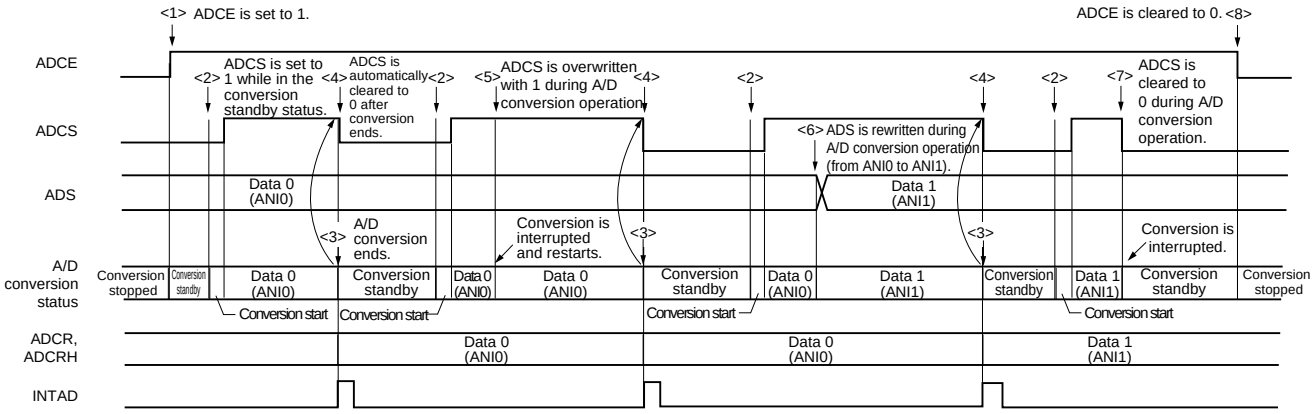
<5> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.

<6> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the analog input respecified by the ADS register. The partially converted data is discarded.

<7> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status.

<8> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCE = 0, specifying 1 for ADCS is ignored and A/D conversion does not start. In addition, A/D conversion does not start even if a hardware trigger is input while in the A/D conversion standby status.

**Figure 14-18. Example of Software Trigger Mode (Select Mode, One-shot Conversion Mode) Operation Timing**



### 14.6.3 Software trigger mode (scan mode, sequential conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.

<2> After the software counts up to the stabilization wait time (1 μs), the ADCS bit of the ADM0 register is set to 1 to perform A/D conversion on the four analog input channels specified by scan 0 to scan 3, which are specified by the analog input channel specification register (ADS). A/D conversion is performed on the analog input channels in order, starting with that specified by scan 0.

<3> A/D conversion is sequentially performed on the four analog input channels, the conversion results are stored in the A/D conversion result register (ADCR, ADCRH) each time conversion ends, and the A/D conversion end interrupt request signal (INTAD) is generated. After A/D conversion of the four channels ends, the A/D conversion of the channel following the specified channel automatically starts (until all four channels are finished).

<4> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.

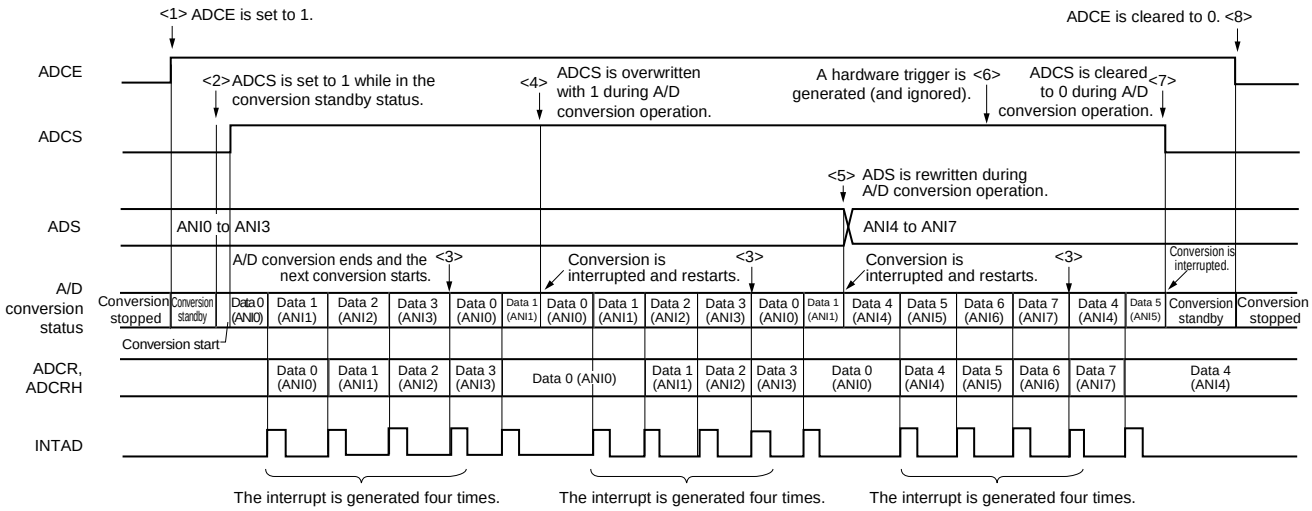
<5> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the first channel respecified by the ADS register. The partially converted data is discarded.

<6> Even if a hardware trigger is input during conversion operation, A/D conversion does not start.

<7> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status.

<8> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCE = 0, specifying 1 for ADCS is ignored and A/D conversion does not start.

**Figure 14-19. Example of Software Trigger Mode (Scan Mode, Sequential Conversion Mode) Operation Timing**



### 14.6.4 Software trigger mode (scan mode, one-shot conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.

<2> After the software counts up to the stabilization wait time (1 μs), the ADCS bit of the ADM0 register is set to 1 to perform A/D conversion on the four analog input channels specified by scan 0 to scan 3, which are specified by the analog input channel specification register (ADS). A/D conversion is performed on the analog input channels in order, starting with that specified by scan 0.

<3> A/D conversion is sequentially performed on the four analog input channels, the conversion results are stored in the A/D conversion result register (ADCR, ADCRH) each time conversion ends, and the A/D conversion end interrupt request signal (INTAD) is generated.

<4> After A/D conversion of the four channels ends, the ADCS bit is automatically cleared to 0, and the system enters the A/D conversion standby status.

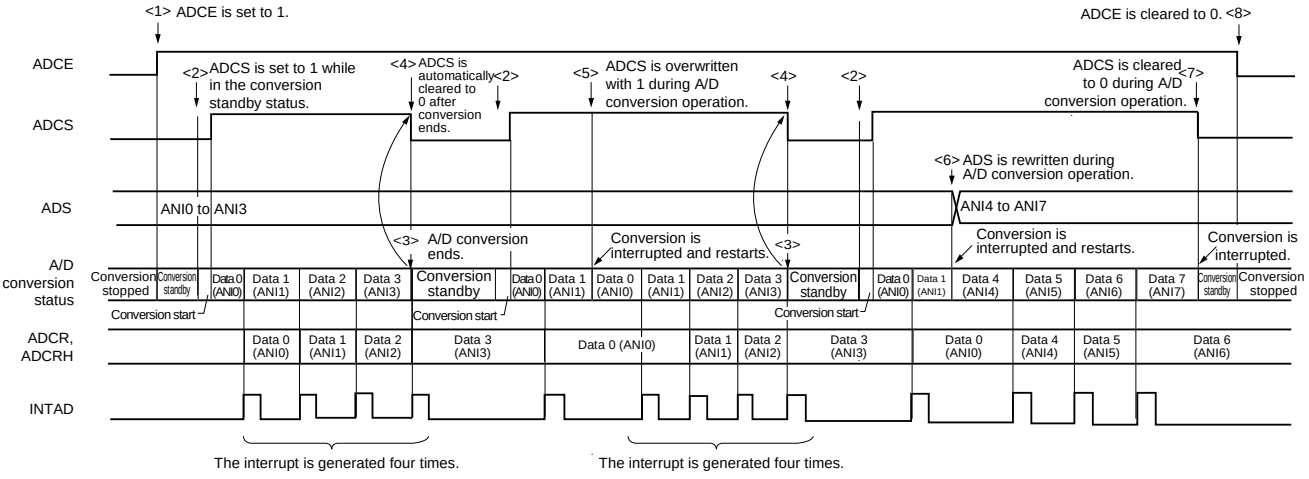
<5> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.

<6> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the first channel respecified by the ADS register. The partially converted data is discarded.

<7> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status.

<8> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCE = 0, specifying 1 for ADCS is ignored and A/D conversion does not start. In addition, A/D conversion does not start even if a hardware trigger is input while in the A/D conversion standby status.

**Figure 14-20. Example of Software Trigger Mode (Scan Mode, One-shot Conversion Mode) Operation Timing**



### 14.6.5 Hardware trigger no-wait mode (select mode, sequential conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.

<2> After the software counts up to the stabilization wait time (1 μs), the ADCS bit of the ADM0 register is set to 1 to place the system in the hardware trigger standby status (and conversion does not start at this stage). Note that, while in this status, A/D conversion does not start even if ADCS is set to 1.

<3> If a hardware trigger is input while ADCS = 1, A/D conversion is performed on the analog input specified by the analog input channel specification register (ADS).

<4> When A/D conversion ends, the conversion result is stored in the A/D conversion result register (ADCR, ADCRH), and the A/D conversion end interrupt request signal (INTAD) is generated. After A/D conversion ends, the next A/D conversion immediately starts.

<5> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.

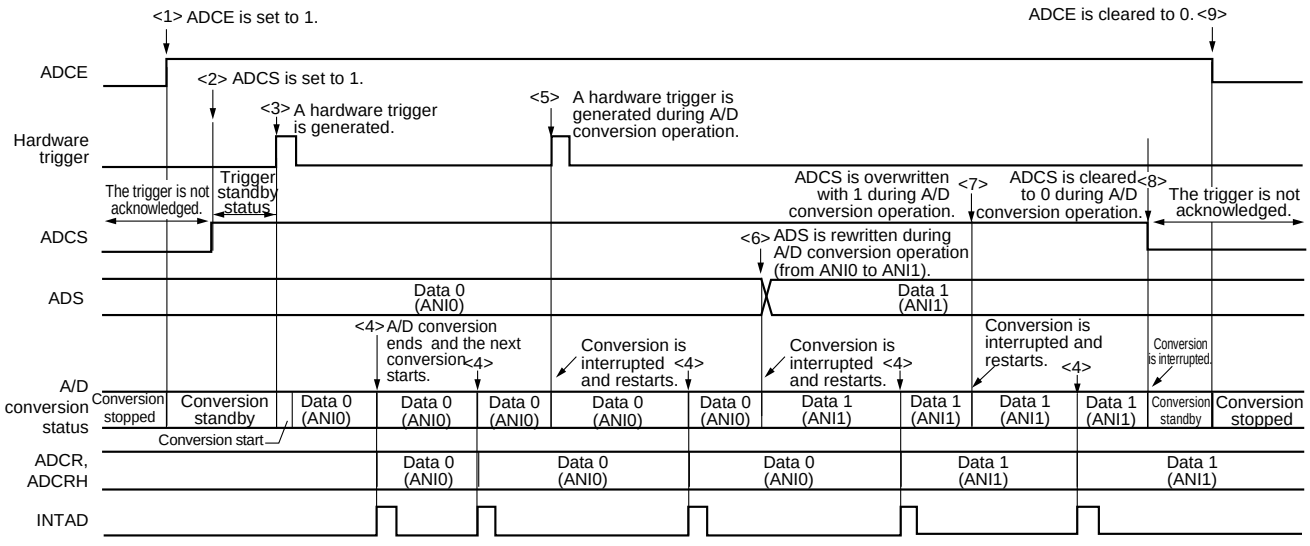
<6> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the analog input respecified by the ADS register. The partially converted data is discarded.

<7> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.

<8> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status. However, the A/D converter does not stop in this status.

<9> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCS = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

**Figure 14-21. Example of Hardware Trigger No-wait Mode (Select Mode, Sequential Conversion Mode) Operation Timing**



### 14.6.6 Hardware trigger no-wait mode (select mode, one-shot conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.

<2> After the software counts up to the stabilization wait time (1 μs), the ADCS bit of the ADM0 register is set to 1 to place the system in the hardware trigger standby status (and conversion does not start at this stage). Note that, while in this status, A/D conversion does not start even if ADCS is set to 1.

<3> If a hardware trigger is input while ADCS = 1, A/D conversion is performed on the analog input specified by the analog input channel specification register (ADS).

<4> When A/D conversion ends, the conversion result is stored in the A/D conversion result register (ADCR, ADCRH), and the A/D conversion end interrupt request signal (INTAD) is generated.

<5> After A/D conversion ends, the ADCS bit remains set to 1, and the system enters the A/D conversion standby status.

<6> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.

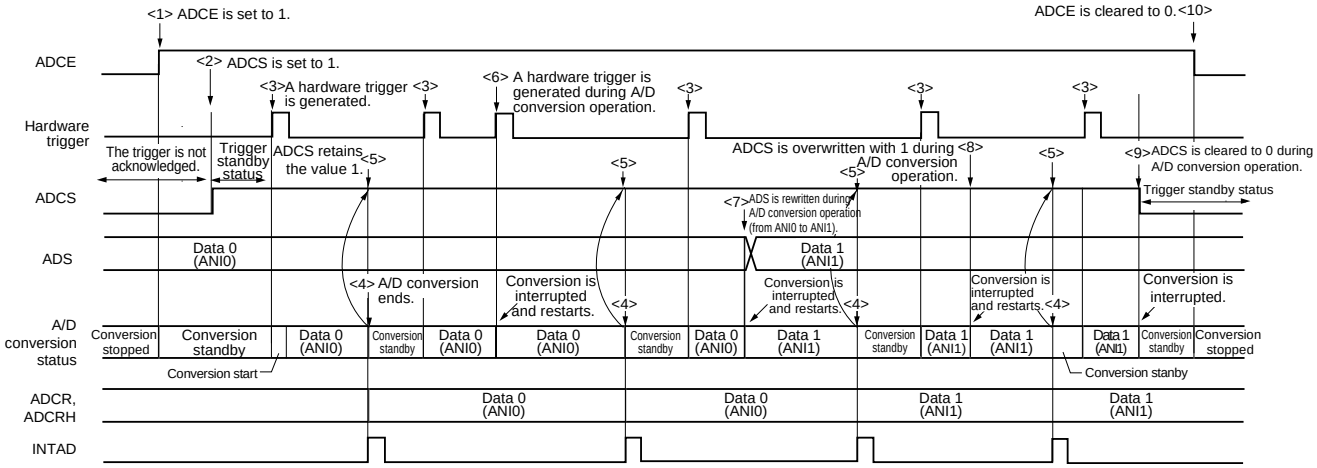
<7> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the analog input respecified by the ADS register. The partially converted data is discarded.

<8> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.

<9> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status. However, the A/D converter does not stop in this status.

<10> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCS = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

**Figure 14-22. Example of Hardware Trigger No-wait Mode (Select Mode, One-shot Conversion Mode) Operation Timing**



### 14.6.7 Hardware trigger no-wait mode (scan mode, sequential conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.

<2> After the software counts up to the stabilization wait time (1 μs), the ADCS bit of the ADM0 register is set to 1 to place the system in the hardware trigger standby status (and conversion does not start at this stage). Note that, while in this status, A/D conversion does not start even if ADCS is set to 1.

<3> If a hardware trigger is input while ADCS = 1, A/D conversion is performed on the four analog input channels specified by scan 0 to scan 3, which are specified by the analog input channel specification register (ADS). A/D conversion is performed on the analog input channels in order, starting with that specified by scan 0.

<4> A/D conversion is sequentially performed on the four analog input channels, the conversion results are stored in the A/D conversion result register (ADCR, ADCRH) each time conversion ends, and the A/D conversion end interrupt request signal (INTAD) is generated. After A/D conversion of the four channels ends, the A/D conversion of the channel following the specified channel automatically starts.

<5> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.

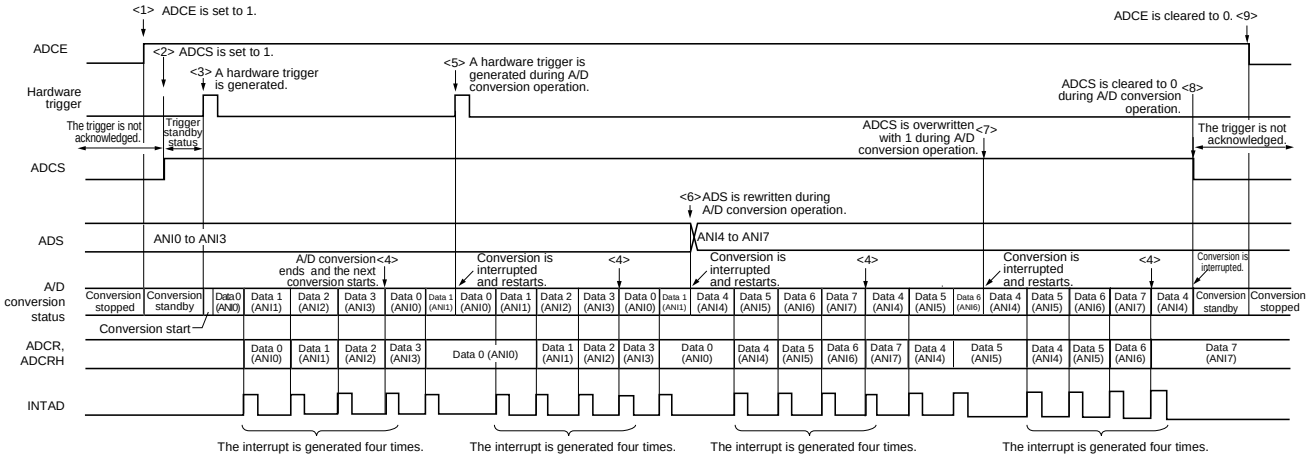
<6> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the first channel respecified by the ADS register. The partially converted data is discarded.

<7> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.

<8> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status. However, the A/D converter does not stop in this status.

<9> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCE = 0, specifying 1 for ADCS is ignored and A/D conversion does not start.

**Figure 14-23. Example of Hardware Trigger No-wait Mode (Scan Mode, Sequential Conversion Mode) Operation Timing**



### 14.6.8 Hardware trigger no-wait mode (scan mode, one-shot conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.

<2> After the software counts up to the stabilization wait time (1 μs), the ADCS bit of the ADM0 register is set to 1 to place the system in the hardware trigger standby status (and conversion does not start at this stage). Note that, while in this status, A/D conversion does not start even if ADCS is set to 1.

<3> If a hardware trigger is input while ADCS = 1, A/D conversion is performed on the four analog input channels specified by scan 0 to scan 3, which are specified by the analog input channel specification register (ADS). A/D conversion is performed on the analog input channels in order, starting with that specified by scan 0.

<4> A/D conversion is sequentially performed on the four analog input channels, the conversion results are stored in the A/D conversion result register (ADCR, ADCRH) each time conversion ends, and the A/D conversion end interrupt request signal (INTAD) is generated.

<5> After A/D conversion of the four channels ends, the ADCS bit remains set to 1, and the system enters the A/D conversion standby status.

<6> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.

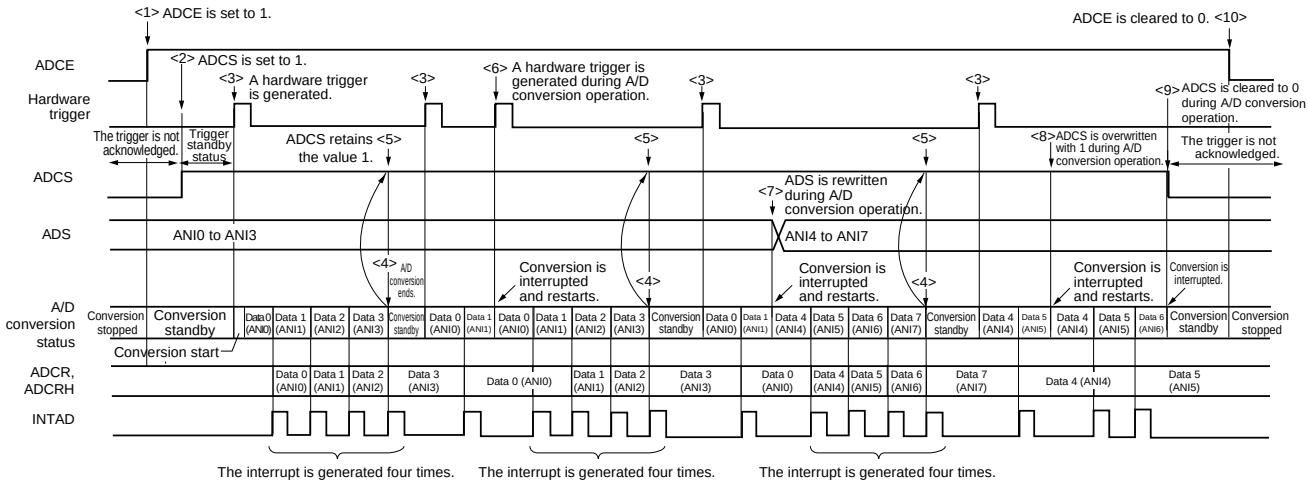
<7> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the first channel respecified by the ADS register. The partially converted data is discarded.

<8> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the first channel. The partially converted data is discarded.

<9> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status. However, the A/D converter does not stop in this status.

<10> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCS = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

**Figure 14-24. Example of Hardware Trigger No-wait Mode (Scan Mode, One-shot Conversion Mode) Operation Timing**



### 14.6.9 Hardware trigger wait mode (select mode, sequential conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the hardware trigger standby status.

<2> If a hardware trigger is input while in the hardware trigger standby status, A/D conversion is performed on the analog input specified by the analog input channel specification register (ADS). The ADCS bit of the ADM0 register is automatically set to 1 according to the hardware trigger input.

<3> When A/D conversion ends, the conversion result is stored in the A/D conversion result register (ADCR, ADCRH), and the A/D conversion end interrupt request signal (INTAD) is generated. After A/D conversion ends, the next A/D conversion immediately starts. (At this time, no hardware trigger is necessary.)

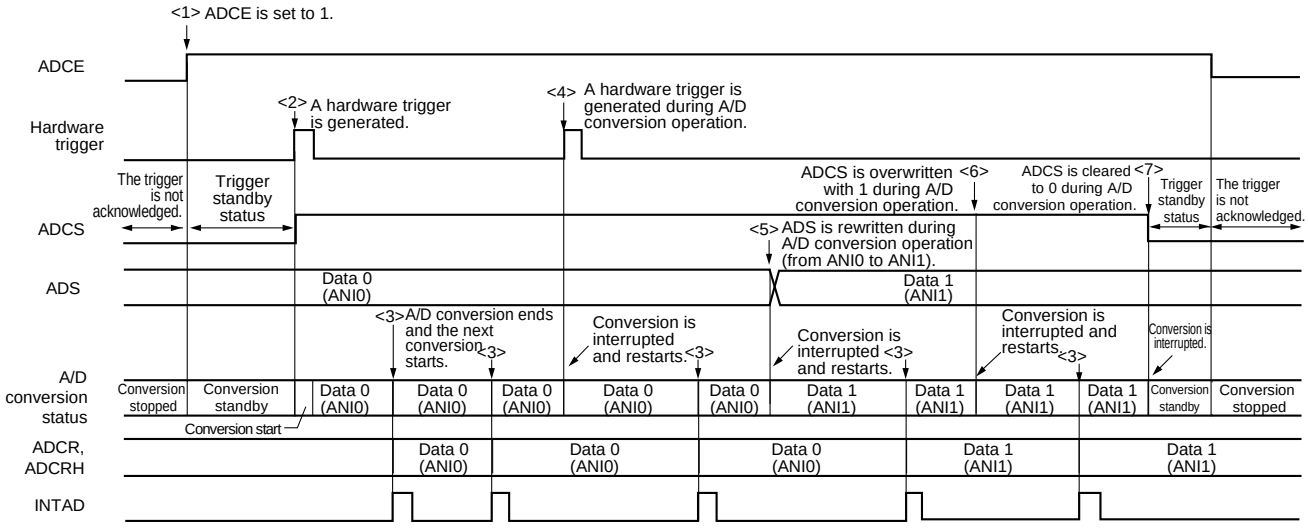
<4> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.

<5> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the analog input respecified by the ADS register. The partially converted data is discarded.

<6> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.

<7> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, the system enters the hardware trigger standby status, and the A/D converter enters the stop status. When ADCE = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

**Figure 14-25. Example of Hardware Trigger Wait Mode (Select Mode, Sequential Conversion Mode) Operation Timing**



### 14.6.10 Hardware trigger wait mode (select mode, one-shot conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the hardware trigger standby status.

<2> If a hardware trigger is input while in the hardware trigger standby status, A/D conversion is performed on the analog input specified by the analog input channel specification register (ADS). The ADCS bit of the ADM0 register is automatically set to 1 according to the hardware trigger input.

<3> When A/D conversion ends, the conversion result is stored in the A/D conversion result register (ADCR, ADCRH), and the A/D conversion end interrupt request signal (INTAD) is generated.

<4> After A/D conversion ends, the ADCS bit is automatically cleared to 0, and the A/D converter enters the stop status.

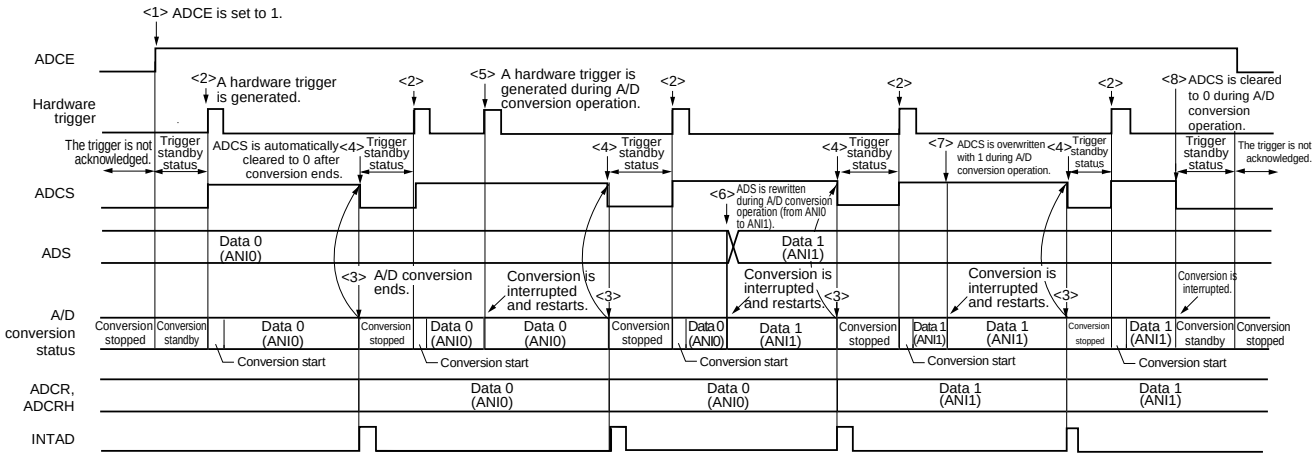
<5> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.

<6> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the analog input respecified by the ADS register. The partially converted data is discarded.

<7> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is initialized.

<8> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, the system enters the hardware trigger standby status, and the A/D converter enters the stop status. When ADCE = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

**Figure 14-26. Example of Hardware Trigger Wait Mode (Select Mode, One-shot Conversion Mode) Operation Timing**



### 14.6.11 Hardware trigger wait mode (scan mode, sequential conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.

<2> If a hardware trigger is input while in the hardware trigger standby status, A/D conversion is performed on the four analog input channels specified by scan 0 to scan 3, which are specified by the analog input channel specification register (ADS). The ADCS bit of the ADM0 register is automatically set to 1 according to the hardware trigger input. A/D conversion is performed on the analog input channels in order, starting with that specified by scan 0.

<3> A/D conversion is sequentially performed on the four analog input channels, the conversion results are stored in the A/D conversion result register (ADCR, ADCRH) each time conversion ends, and the A/D conversion end interrupt request signal (INTAD) is generated. After A/D conversion of the four channels ends, the A/D conversion of the channel following the specified channel automatically starts.

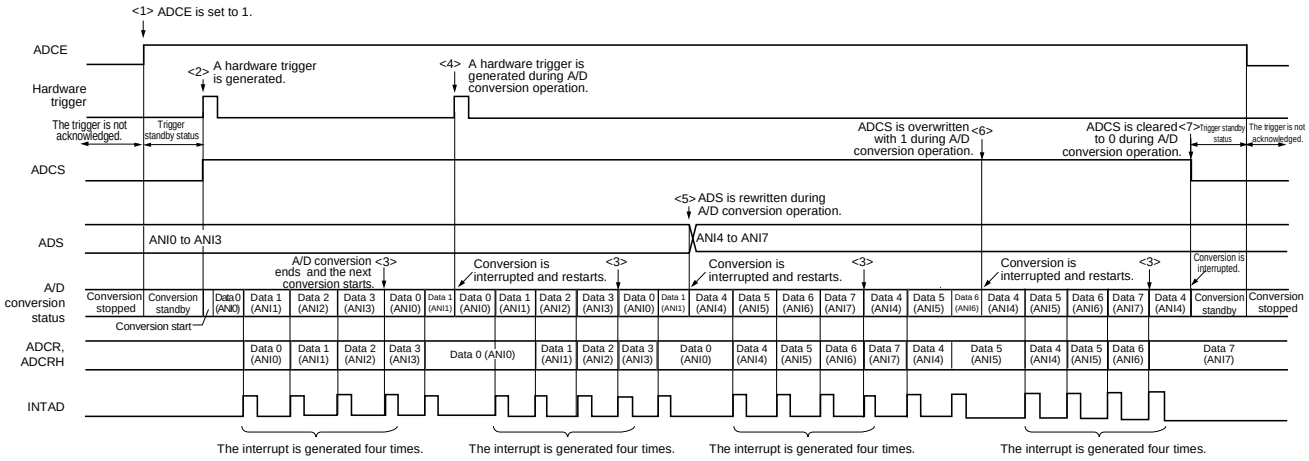
<4> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.

<5> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the first channel respecified by the ADS register. The partially converted data is discarded.

<6> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.

<7> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, the system enters the hardware trigger standby status, and the A/D converter enters the stop status. When ADCE = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

**Figure 14-27. Example of Hardware Trigger Wait Mode (Scan Mode, Sequential Conversion Mode) Operation Timing**



### 14.6.12 Hardware trigger wait mode (scan mode, one-shot conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.

<2> If a hardware trigger is input while in the hardware trigger standby status, A/D conversion is performed on the four analog input channels specified by scan 0 to scan 3, which are specified by the analog input channel specification register (ADS). The ADCS bit of the ADM0 register is automatically set to 1 according to the hardware trigger input. A/D conversion is performed on the analog input channels in order, starting with that specified by scan 0.

<3> A/D conversion is sequentially performed on the four analog input channels, the conversion results are stored in the A/D conversion result register (ADCR, ADCRH) each time conversion ends, and the A/D conversion end interrupt request signal (INTAD) is generated.

<4> After A/D conversion ends, the ADCS bit is automatically cleared to 0, and the A/D converter enters the stop status.

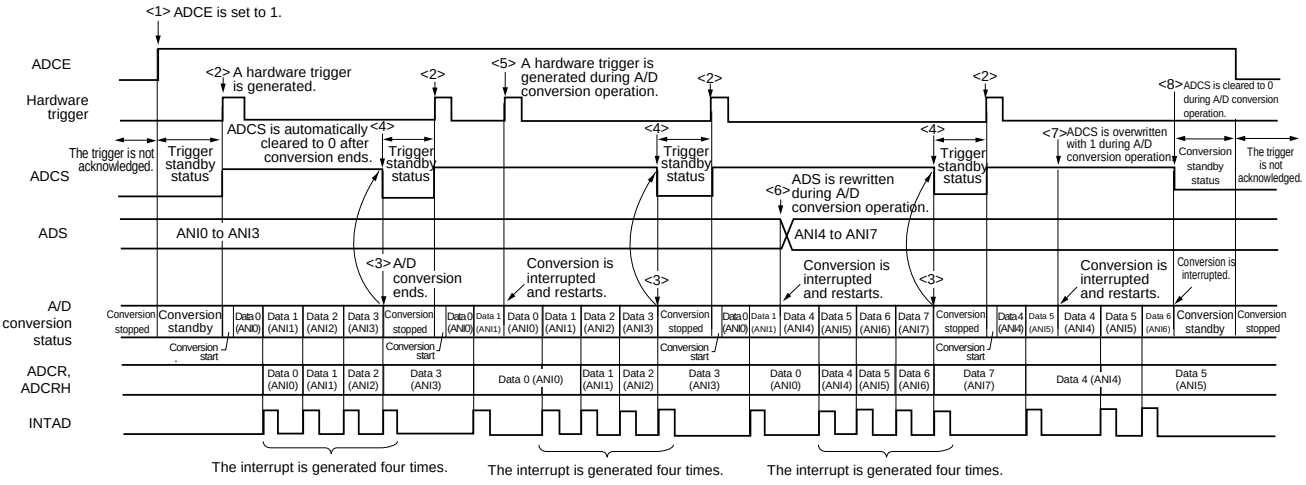
<5> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.

<6> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the first channel respecified by the ADS register. The partially converted data is discarded.

<7> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.

<8> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, the system enters the hardware trigger standby status, and the A/D converter enters the stop status. When ADCE = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

**Figure 14-28. Example of Hardware Trigger Wait Mode (Scan Mode, One-shot Conversion Mode) Operation Timing**



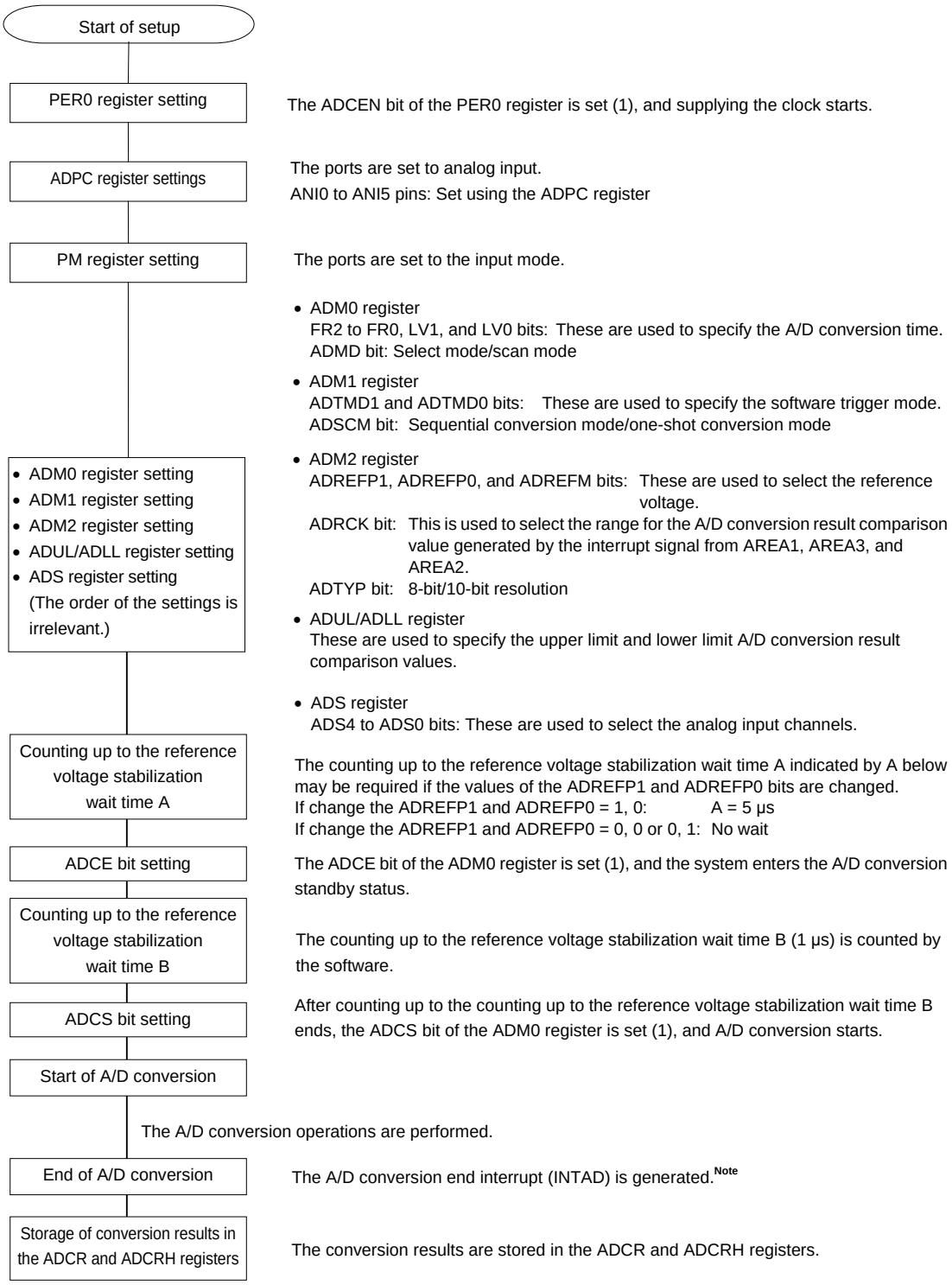
## 14.7 A/D Converter Setup Flowchart

The A/D converter setup flowchart in each operation mode is described below.

### 14.7.1 Setting up software trigger mode

<R>

**Figure 14-29. Setting up Software Trigger Mode**

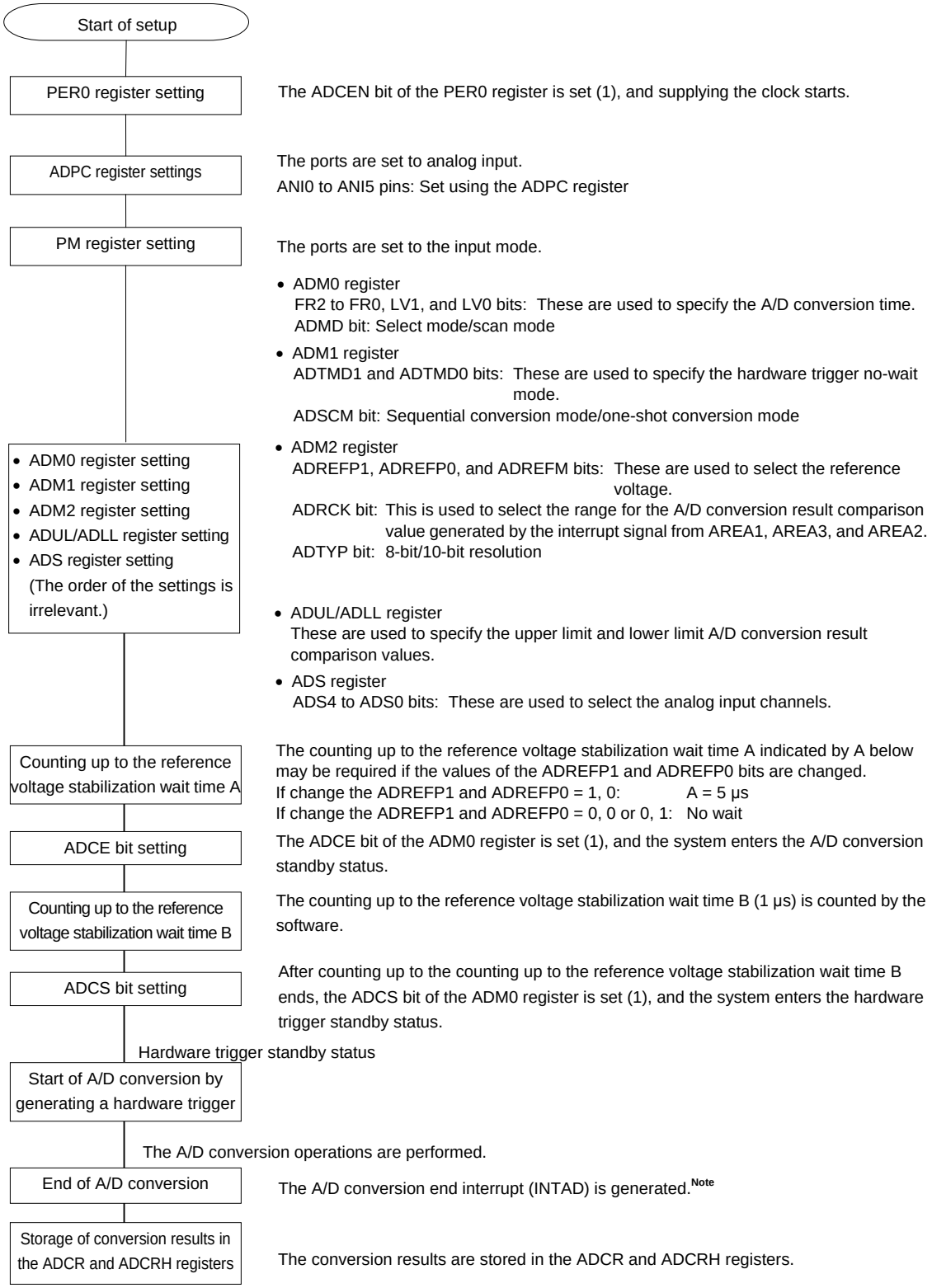



**Note** Depending on the settings of the ADRCK bit and ADUL/ADLL register, there is a possibility of no interrupt signal being generated. In this case, the results are not stored in the ADCR, ADCRH registers.

### 14.7.2 Setting up hardware trigger no-wait mode

<R>

**Figure 14-30. Setting up Hardware Trigger No-wait Mode**

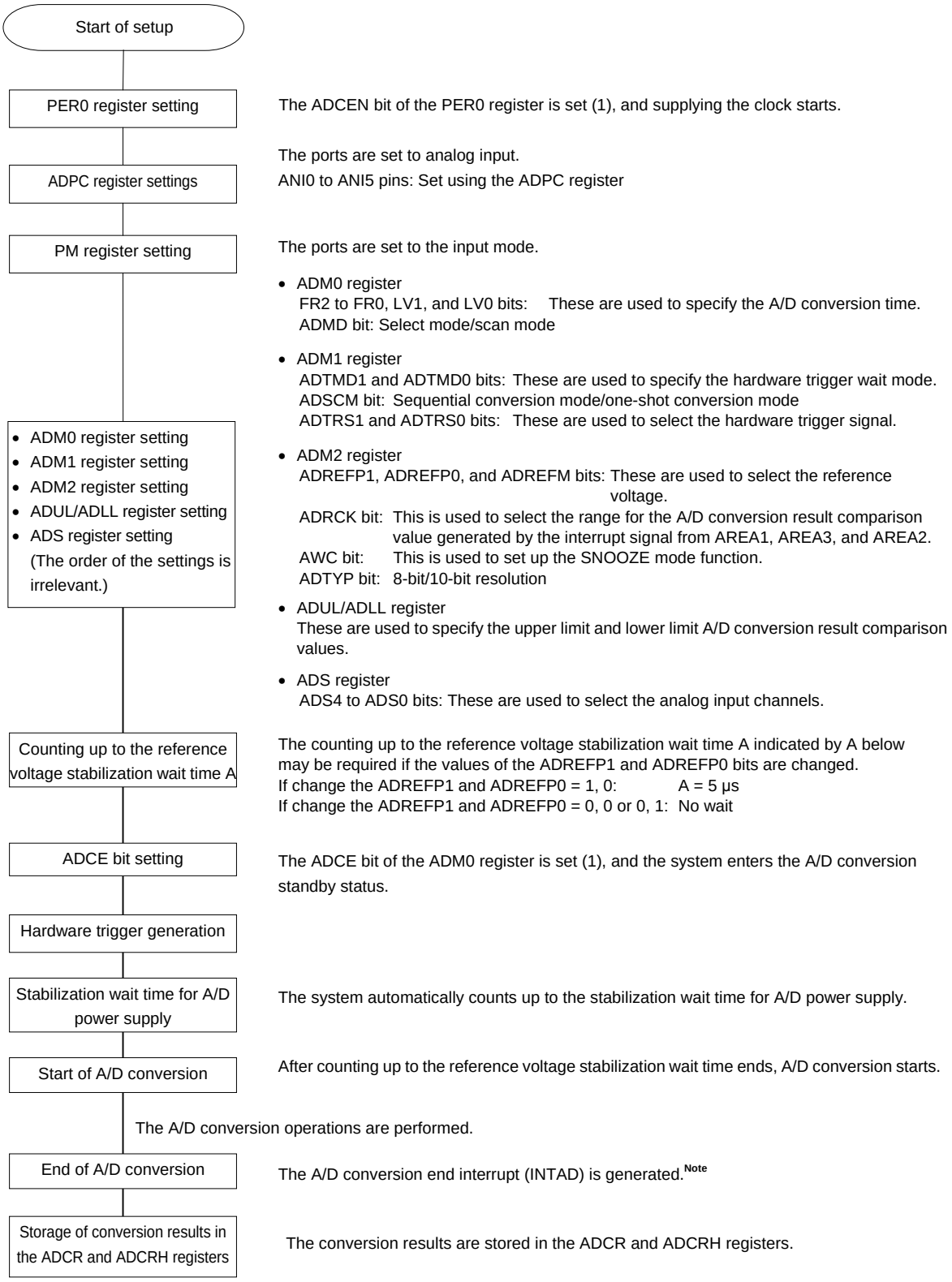



**Note** Depending on the settings of the ADRCK bit and ADUL/ADLL register, there is a possibility of no interrupt signal being generated. In this case, the results are not stored in the ADCR, ADCRH registers.

### 14.7.3 Setting up hardware trigger wait mode

<R>

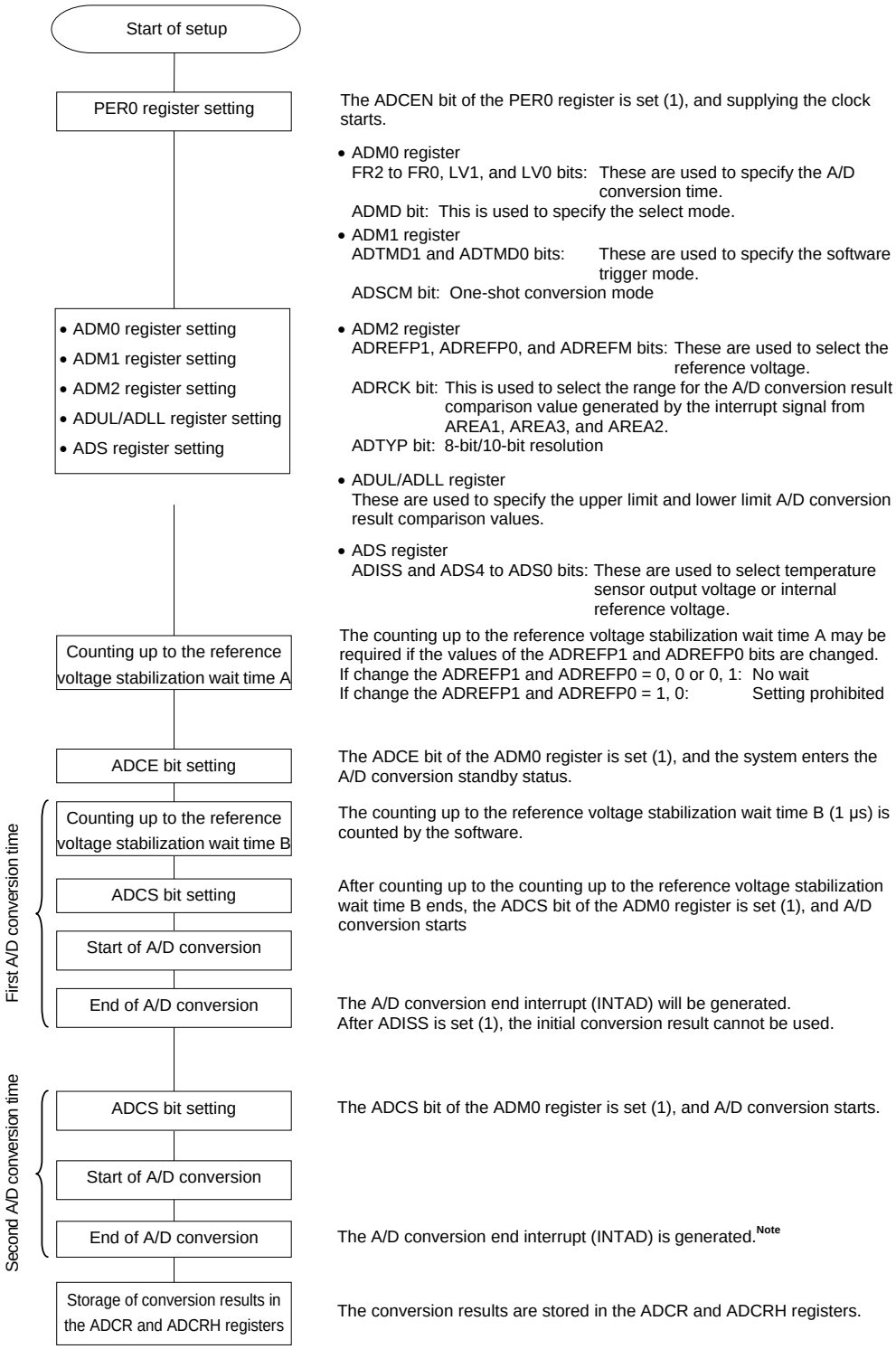
**Figure 14-31. Setting up Hardware Trigger Wait Mode**




**Note** Depending on the settings of the ADRCK bit and ADUL/ADLL register, there is a possibility of no interrupt signal being generated. In this case, the results are not stored in the ADCR, ADCRH registers.

### 14.7.4 Setup when temperature sensor output voltage/internal reference voltage is selected (example for software trigger mode and one-shot conversion mode)

<R> **Figure 14-32. Setup When Temperature Sensor Output Voltage/Internal Reference Voltage Is Selected**



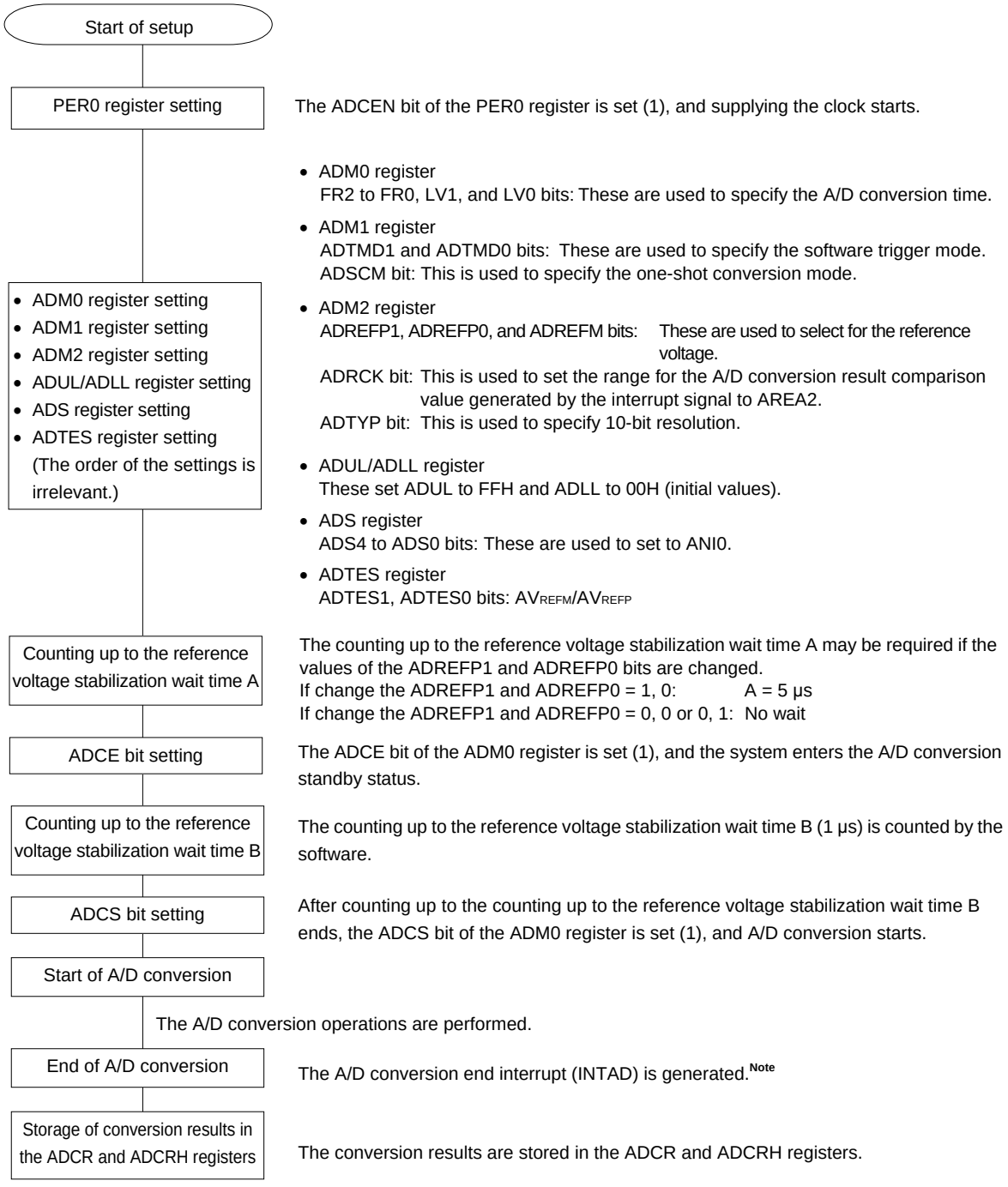

**Note** Depending on the settings of the ADRCK bit and ADUL/ADLL register, there is a possibility of no interrupt signal being generated. In this case, the results are not stored in the ADCR and ADCRH registers.

**Caution This setting can be used only in HS (high-speed main) mode.**

### 14.7.5 Setting up test mode

<R>

**Figure 14-33. Setting up Test Mode**




**Note** Depending on the settings of the ADRCK bit and ADUL/ADLL register, there is a possibility of no interrupt signal being generated. In this case, the results are not stored in the ADCR, ADCRH registers.

**Caution For the procedure for testing the A/D converter, see 30.3.8 A/D test function.**

## 14.8 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, A/D conversion can be performed without operating the CPU. This is effective for reducing the operation current.

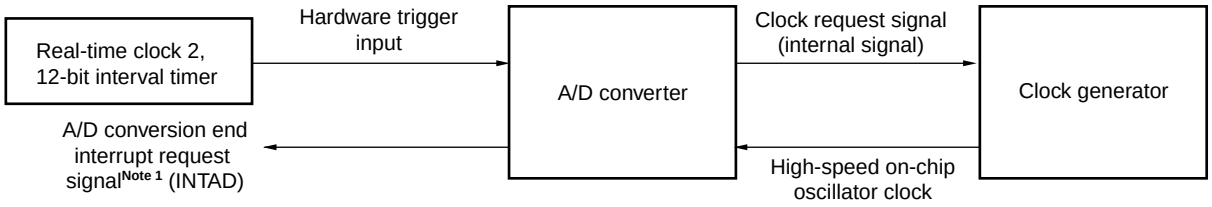
If the A/D conversion result range is specified using the ADUL and ADLL registers, A/D conversion results can be judged at a certain interval of time in SNOOZE mode. Using this function enables power supply voltage monitoring and input key judgment based on A/D inputs.

In the SNOOZE mode, only the following conversion modes can be used:

- Hardware trigger wait mode (select mode, one-shot conversion mode)
- Hardware trigger wait mode (scan mode, one-shot conversion mode)

**Caution That the SNOOZE mode can only be specified when the high-speed on-chip oscillator clock is selected for $f_{CLK}$.**

**Figure 14-34. 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 **14.7.3 Setting up hardware trigger wait mode**[Note 2]). Just before move to STOP mode, bit 2 (AWC) of A/D converter mode register 2 (ADM2) is set to 1. After the initial settings are specified, bit 0 (ADCE) 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 system 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 ends differs depending on whether an interrupt signal is generated[Note 1].

**Notes 1.** Depending on the setting of the A/D conversion result comparison function (ADRCK bit, ADUL/ADLL register), there is a possibility of no interrupt signal being generated.

**2.** Be sure to set the ADM1 register to E2H or E3H.

**Remark** The hardware trigger is INTRTC or INTIT.
Specify the hardware trigger by using the A/D Converter Mode Register 1 (ADM1).

**(1) If an interrupt is generated after A/D conversion ends**

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/ADLL register), the A/D conversion end interrupt request signal (INTAD) is generated.

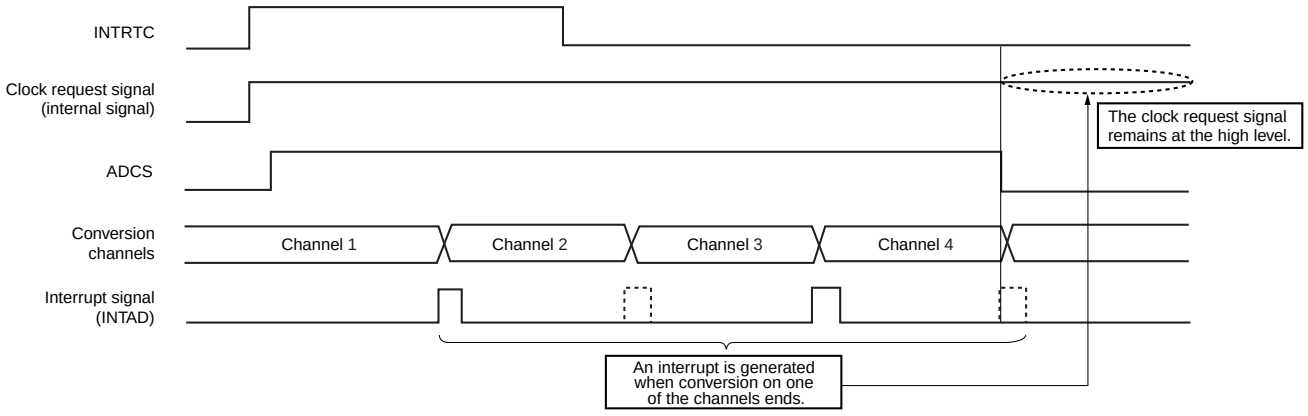
- While in the select mode

  When A/D conversion ends 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 clear bit 2 (AWC = 0: SNOOZE mode release) of the A/D converter mode register 2 (ADM2). 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 A/D conversion end interrupt request signal (INTAD) is generated during A/D conversion of the four channels, the clock request signal remains at the high level, and the A/D converter switches from the SNOOZE mode to the normal operation mode. At this time, be sure to clear bit 2 (AWC = 0: SNOOZE mode release) 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 14-35. Operation Example When Interrupt Is Generated After A/D Conversion Ends (While in Scan Mode)**



**(2) If no interrupt is generated after A/D conversion ends**

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/ADLL register), the A/D conversion end interrupt request signal (INTAD) is not generated.

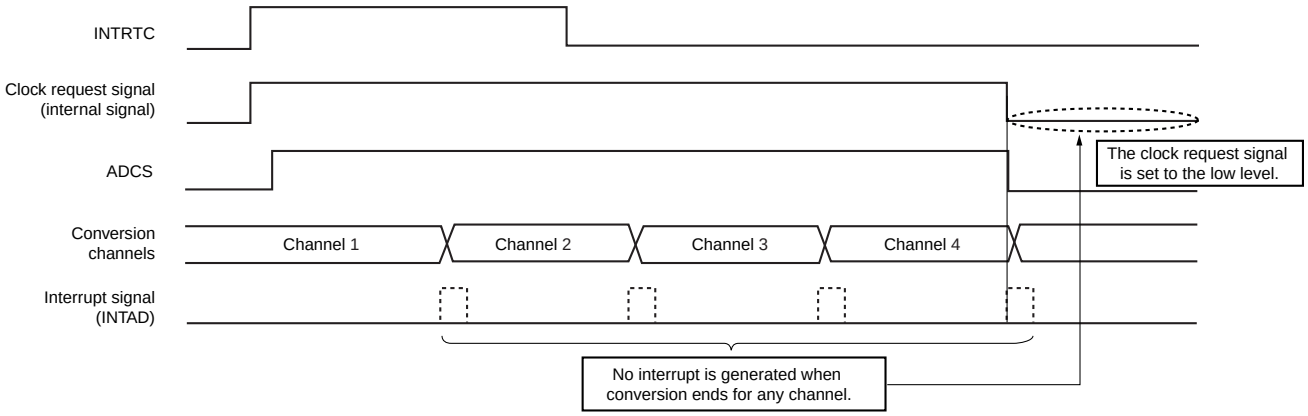
- While in the select mode

  If the A/D conversion end interrupt request signal (INTAD) is not generated after A/D conversion ends, 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

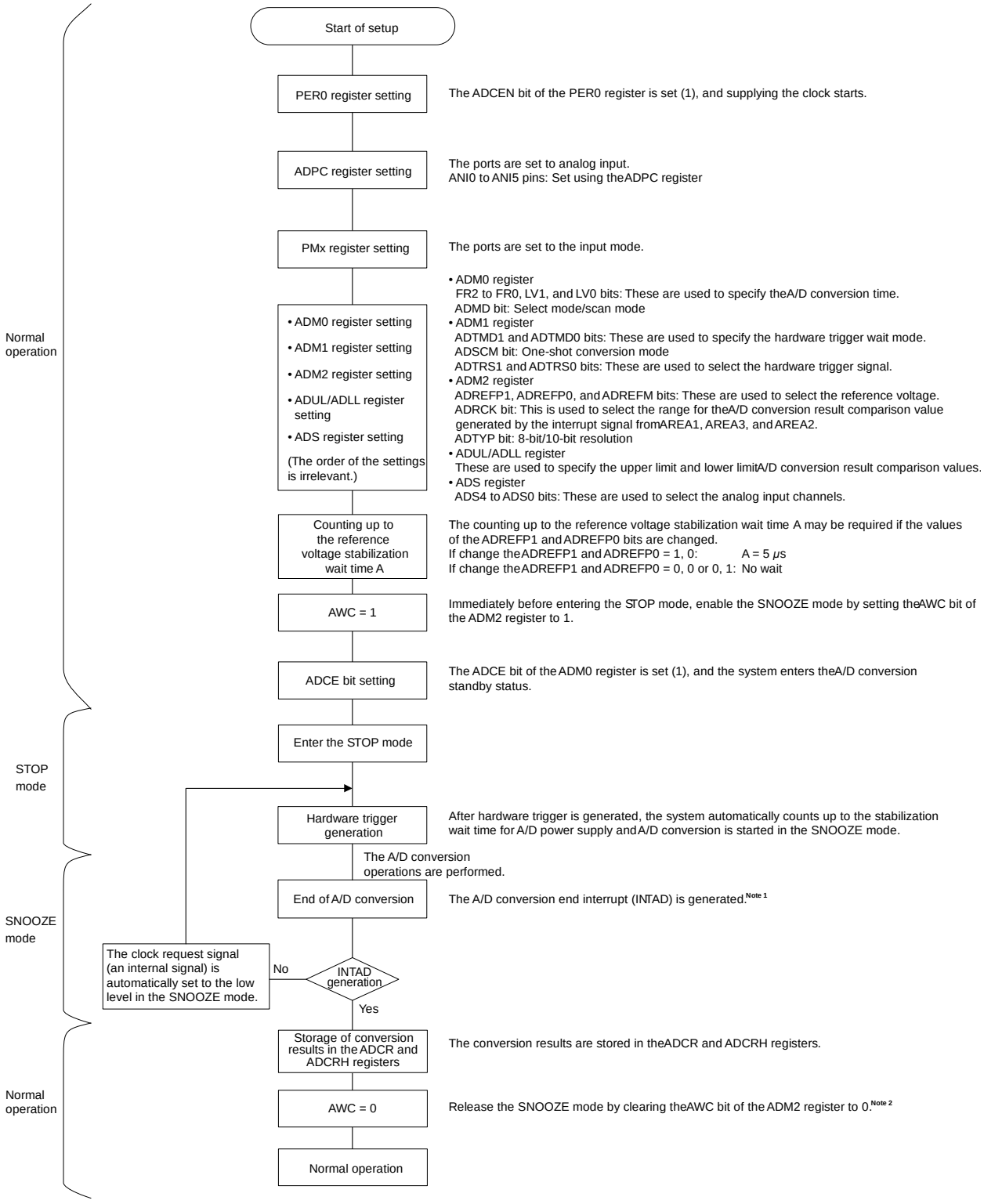
  If the A/D conversion end interrupt request signal (INTAD) is not generated even once during A/D conversion of the four channels, the clock request signal (an internal signal) is automatically set to the low level after A/D conversion of the four channels ends, and supplying the high-speed on-chip oscillator clock stops. If a hardware trigger is input later, A/D conversion work is again performed in the SNOOZE mode.

**Figure 14-36. Operation Example When No Interrupt Is Generated After A/D Conversion Ends (While in Scan Mode)**



<R>

**Figure 14-37. Flowchart for Setting up SNOOZE Mode**




**Notes 1.** If the A/D conversion end interrupt request signal (INTAD) is not generated by setting ADRCK bit and ADUL/ADLL register, the result is not stored in the ADCR and ADCRH registers.
The system enters the STOP mode again. If a hardware trigger is input later, A/D conversion operation is again performed in the SNOOZE mode.

**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 clear the AWC bit to 0.

## 14.9 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 1LSB (Least Significant Bit). The percentage of 1LSB with respect to the full scale is expressed by %FSR (Full Scale Range).

1LSB is as follows when the resolution is 10 bits.

$1LSB = 1/2^{10} = 1/1024$
$= 0.098\%FSR$

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 errors that are combinations of these express the overall error.
Note that the quantization error is not included in the overall error in the characteristics table.

### (3) Quantization error

When analog values are converted to digital values, a ±1/2LSB error naturally occurs. In an A/D converter, an analog input voltage in a range of ±1/2LSB 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 14-38. Overall Error**

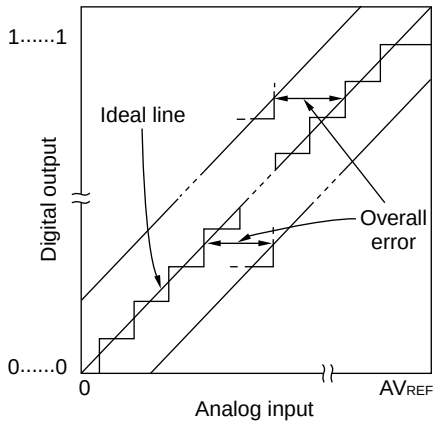
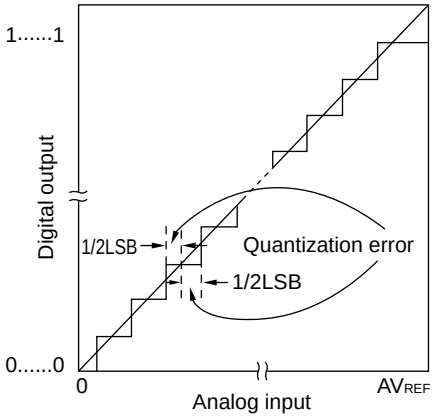



**Figure 14-39. Quantization Error**




### (4) Zero-scale error

This shows the difference between the actual measurement value of the analog input voltage and the theoretical value (1/2LSB) 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 (3/2LSB) when the digital output changes from 0……001 to 0……010.

**(5) Full-scale error**

This shows the difference between the actual measurement value of the analog input voltage and the theoretical value (Full-scale – 3/2LSB) when the digital output changes from 1.....110 to 1.....111.

**(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.

**(7) Differential linearity error**

While the ideal width of code output is 1LSB, this indicates the difference between the actual measurement value and the ideal value.

**Figure 14-40. Zero-Scale Error**

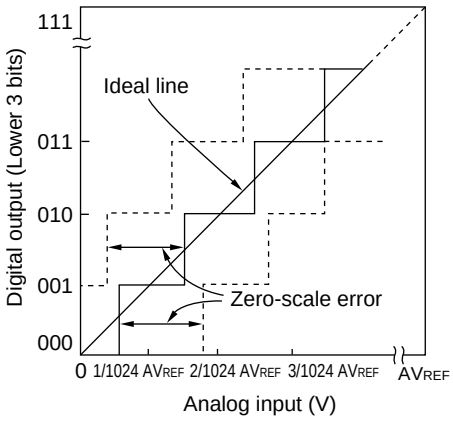



**Figure 14-41. Full-Scale Error**

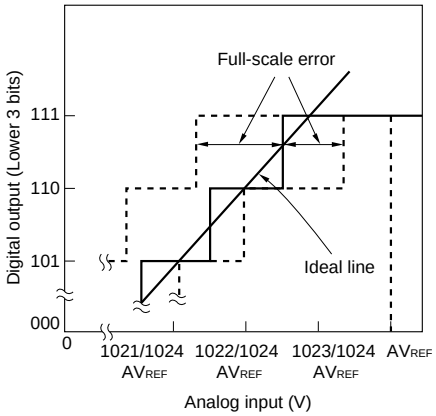



**Figure 14-42. Integral Linearity Error**

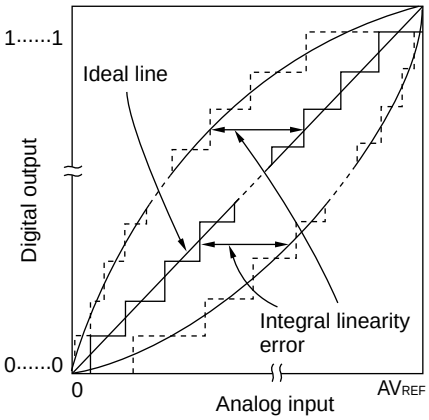
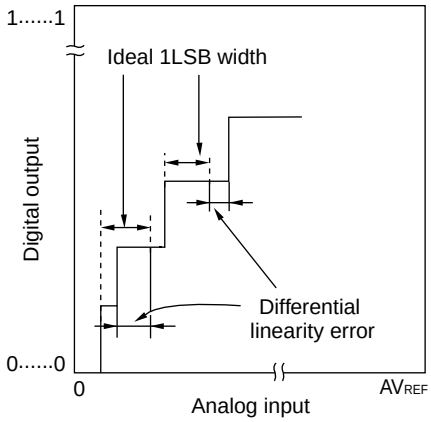



**Figure 14-43. Differential Linearity Error**



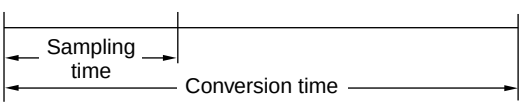

**(8) Conversion time**

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.



## 14.10 Cautions for A/D Converter

**(1) Operating current in STOP mode**

Shift to STOP mode after stopping the A/D converter (by setting bit 7 (ADCS) of A/D converter mode register 0 (ADM0) to 0). The operating current can be reduced by setting bit 0 (ADCE) of the ADM0 register to 0 at the same time.

To restart from the standby status, clear bit 0 (ADIF) of interrupt request flag register 1H (IF1H) to 0 and start operation.

**(2) Input range of ANI0 to ANI5 pins**

Observe the rated range of the ANI0 to ANI5 pins input voltage. If a voltage exceeding $V_{DD}$ and $AV_{REFP}$ or below $V_{SS}$ 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 (1.45 V) is selected reference voltage for the + side of the A/D converter, do not input voltage exceeding internal reference voltage (1.45 V) to a pin selected by the ADS register. However, it is no problem that a voltage exceeding the internal reference voltage (1.45 V) is input to a pin not selected by the ADS register.

**Caution Internal reference voltage (1.45 V) can be used only in HS (high-speed main) mode.**

**(3) Conflicting operations**

<1> Conflict between the A/D conversion result register (ADCR, ADCRH) write and the ADCR or ADCRH register read by instruction upon the end of conversion

The ADCR or ADCRH register read has priority. After the read operation, the new conversion result is written to the ADCR or ADCRH registers.

<2> Conflict between the ADCR or ADCRH register write and the A/D converter mode register 0 (ADM0) write, the analog input channel specification register (ADS), or A/D port configuration register (ADPC) write upon the end of conversion

The ADM0, ADS, or ADPC registers write has priority. The ADCR or ADCRH register write 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 ANI5 pins.

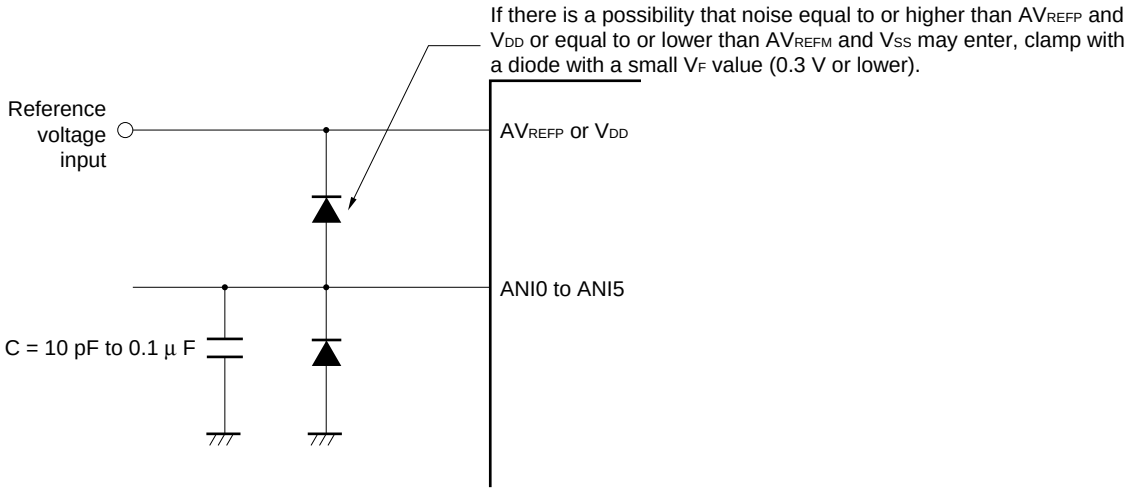
<1> Connect a capacitor with a low equivalent resistance and a good frequency response (capacitance of about 0.01 μF) via the shortest possible run of relatively thick wiring to the power supply.

<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 14-44 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 14-44. Analog Input Pin Connection**




**(5) Analog input (ANIn) pins**

<1> The analog input pins (ANI0 to ANI5) are also used as input port pins (P20 to P25).
When A/D conversion is performed with any of the ANI0 to ANI5 pins selected, do not change to output value P20 to P25 while conversion is in progress; otherwise the conversion resolution may be degraded.

<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 avoid the input or output of digital signals and signals with similarly sharp transitions during A/D conversion.

**(6) Input impedance of analog input (ANIn) 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 ANI5 which the source is connected (see **Figure 14-44**). 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 after the value of the ADCS bit has been 1 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.

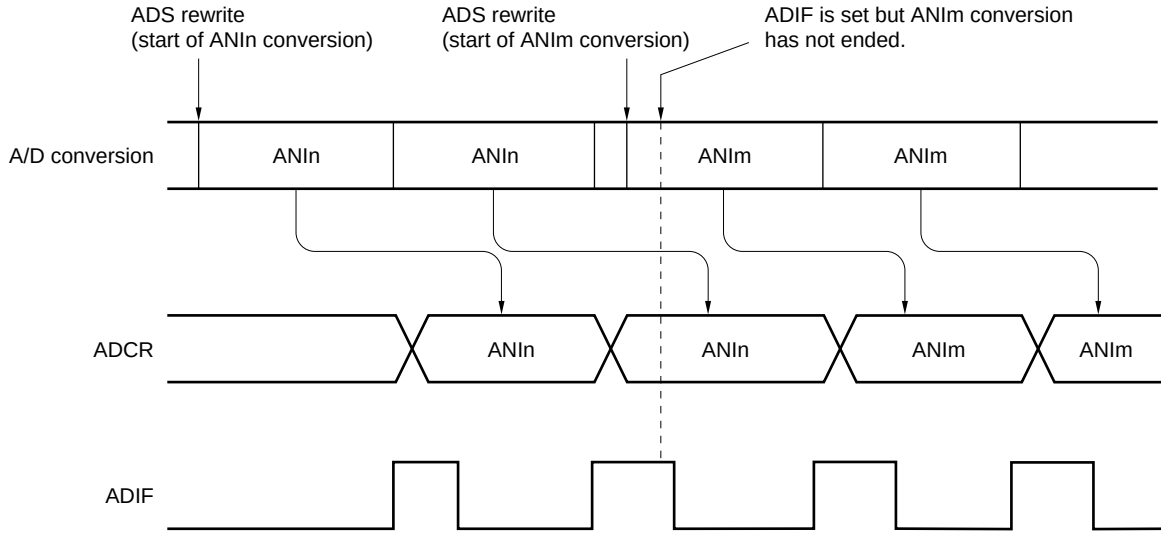
**(7) Interrupt request flag (ADIF)**

The interrupt request flag (ADIF) is not cleared 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 before the A/D conversion operation is resumed.

**Figure 14-45. 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 (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), and A/D port configuration register (ADPC), 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 ADPC 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 14-46. Internal Equivalent Circuit of ANIn Pin**

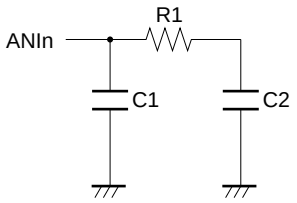



**Table 14-4. Resistance and Capacitance Values of Equivalent Circuit (Reference Values)**

| $AV_{REFP}$, $V_{DD}$ | ANIn Pins | R1 [kΩ] | C1 [pF] | C2 [pF] |
|---|---|---|---|---|
| 3.6 V ≤ $V_{DD}$ ≤ 5.5 V | ANI0 to ANI5 | 14 | 8 | 2.5 |
| 2.7 V ≤ $V_{DD}$ ≤ 3.6 V | ANI0 to ANI5 | 39 | 8 | 2.5 |
| 1.9 V ≤ $V_{DD}$ < 2.7 V | ANI0 to ANI5 | 231 | 8 | 2.5 |

**Remark** The resistance and capacitance values shown in Table 14-4 are not guaranteed values.

**(11) Starting the A/D converter**

Start the A/D converter after the $AV_{REFP}$ and $V_{DD}$ voltages stabilize.

# CHAPTER 15 TEMPERATURE SENSOR 2

## 15.1 Functions of Temperature Sensor

The RL78/I1B has an on-chip temperature sensor. Temperature can be measured by measuring the output voltage from the temperature sensor using the 10-bit A/D converter. The mode of the temperature sensor can be switched to one of the following three modes by setting the temperature control register.

- High-temperature range mode: Mode 1, 0 °C to 90 °C (Output Image Diagram Mode 1)
- Normal-temperature range mode: Mode 2, –20 °C to 70 °C (Output Image Diagram Mode 2)
- Low-temperature range mode: Mode 3, –40 °C to 50 °C (Output Image Diagram Mode 3)

Temperature sensor may be used in HS (high-speed main) mode.

Figure 15-1 shows a block diagram of temperature sensor.

**Figure 15-1. Block Diagram**

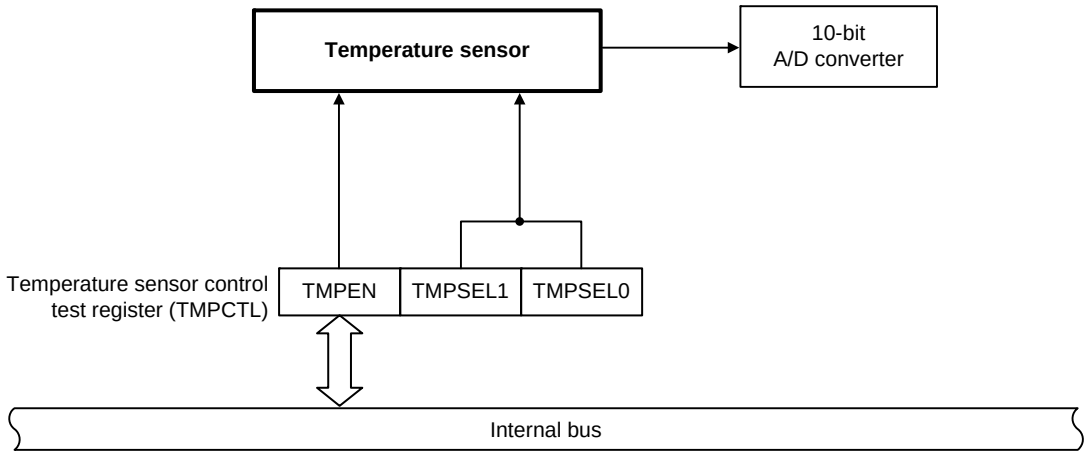
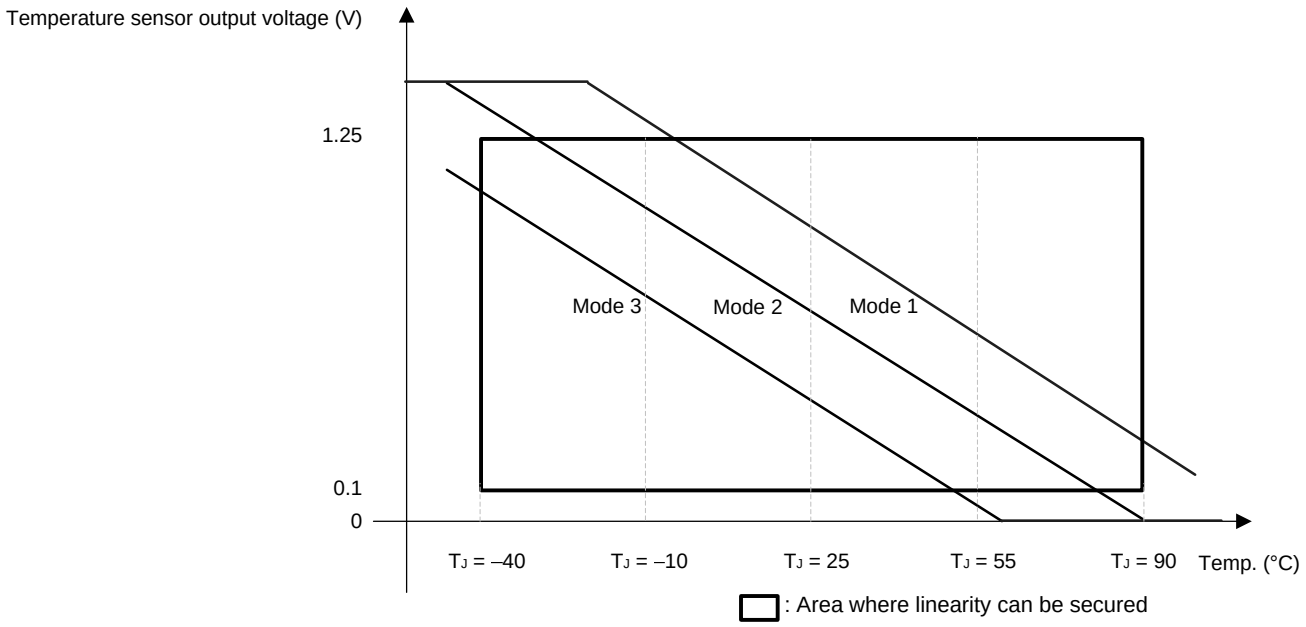



**Figure 15-2. Output Image Diagram**



## 15.2 Registers

Table 15-1 shows the register used for the temperature sensor.

**Table 15-1. Register**

| Item | Configuration |
|---|---|
| Control registers | Temperature sensor control test register (TMPCTL) |

### 15.2.1 Temperature sensor control test register (TMPCTL)

The TMPCTL register is used to stop or start operation of the temperature sensor, and select the mode of the temperature sensor.

The TMPCTL register can be set by a 1-bit or 8-bit memory manipulation instruction. Reset generation clears this register to 00H.

**Figure 15-3. Format of Temperature sensor control test register (TMPCTL)**

Address: F03B0H After reset: 00H R/W

| Symbol | <7> | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TMPCTL | TMPEN Note 1 | 0 | 0 | 0 | 0 | 0 | TMPSEL1 Note 2 | TMPSEL0 Note 2 |

| TMPEN | Temperature sensor operation control |
|---|---|
| 0 | Temperature sensor stops operation |
| 1 | Temperature sensor starts operation |

| TMPSEL1 | TMPSEL0 | Temperature sensor operation selection |
|---|---|---|
| 0 | 0 | Normal-temperature range (Mode 2) |
| 0 | 1 | Low-temperature range (Mode 3) |
| 1 | 0 | High-temperature range (Mode 1) |
| Other than above | | Setting prohibited |

**Notes 1.** After setting the TMPEN bit to 1, a 50 μs operation stabilization wait time is necessary.

**2.** After changing bits TMPSEL1-TMPSEL0, a 15 μs mode switch stabilization wait time is necessary.

**Cautions 1. Be sure to clear bits 6 to 2 to "0".**

**2. When using a temperature sensor, use a 10-bit A/D converter at internal reference voltage. If you select $V_{DD}$ as reference voltage, operation will not be normal.**

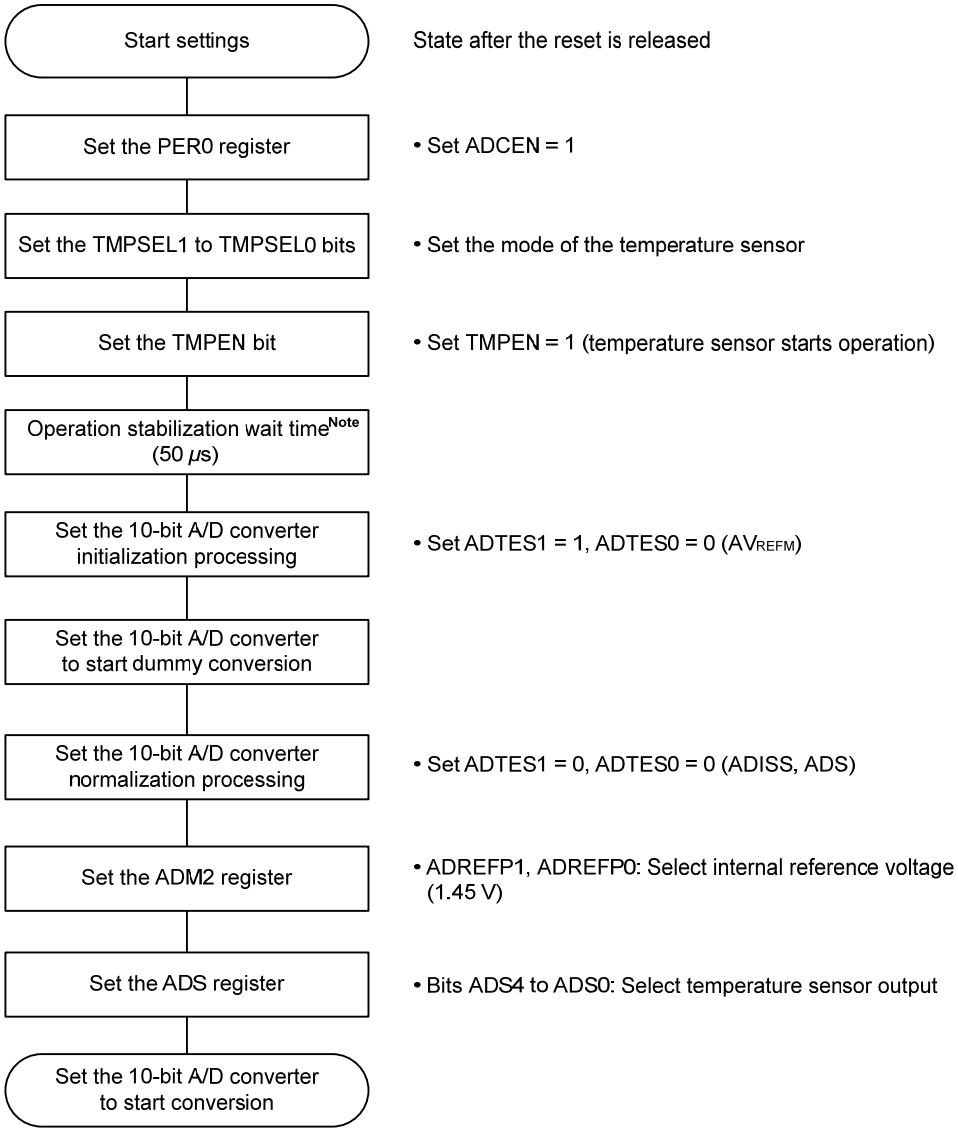
## 15.3 Setting Procedures

The procedures for setting the temperature sensor are shown below.

### 15.3.1 A/D converter mode register 0 (ADM0)

Figure 15-4 shows the setting flowchart when starting operation of temperature sensor.

**Figure 15-4. Setting Flowchart When Starting Operation of Temperature Sensor**



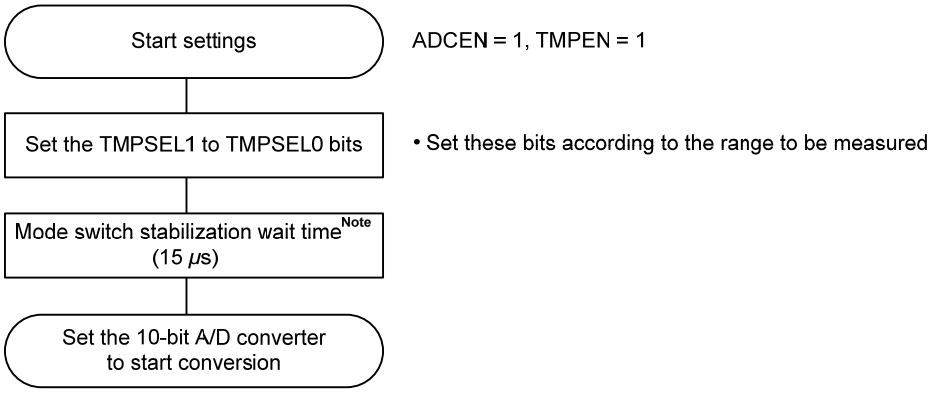

**Note** Operation stabilization wait time is required until the A/D converter starts conversion.

**Caution Select internal reference voltage for 10-bit A/D converter.**

### 15.3.2 Switching modes

Figure 15-5 shows the setting flowchart when switching mode of temperature sensor.

**Figure 15-5. Setting Flowchart When Switching Mode of Temperature Sensor**




**Note** Mode switch stabilization wait time is required until the A/D converter starts conversion.

**Caution Select internal reference voltage for 10-bit A/D converter.**

# CHAPTER 16 24-BIT ΔΣ A/D CONVERTER

The 24-bit ΔΣ A/D converter has a 24-bit resolution when converting an analog input signal to digital values.

## 16.1 Functions of 24-bit ΔΣ A/D Converter

The 24-bit ΔΣ A/D converter has the following functions:

- O S/N+D ratio: 80 dB min. (when pre-amplifier gain of ×1 is selected)
- O 24-bit resolution (conversion result register: 24 bits)
- O 3 channels (current channel: 2 channels voltage channel: 1 channel) (80-pin products)
- O 4 channels (current channel: 2 channels voltage channel: 2 channels) (100-pin products)
- O Analog input: 8 (positive, negative input/channel)
- O ΔΣ conversion mode
- O Pre-amplifier gain selectable: ×1, ×2, ×4, ×8, ×16, or ×32[Note] (channels 0 and 2: current channels)
  ×1, ×2, ×4, ×8, or ×16 (channels 1 and 3: voltage channels)
- O Operating voltage: $AV_{DD}$ = 2.4 to 5.5 V, $AV_{SS}$ = 0 V
- O Analog input voltage: ±0.500 V (when pre-amplifier gain of ×1 is selected)
  ±0.250 V (when pre-amplifier gain of ×2 is selected)
  ±0.125 V (when pre-amplifier gain of ×4 is selected)
  ±62.5 mV (when pre-amplifier gain of ×8 is selected)
  ±31.25 mV (when pre-amplifier gain of ×16 is selected)
  ±15.625 mV (when pre-amplifier gain of ×32[Note] is selected)
- O Reference voltage generation (0.8 V (TYP.) can be output)
- O Sampling frequency: 3906.25 Hz (4 kHz sampling mode)/1953.125 Hz (2 kHz sampling mode)
- O HPF cutoff frequency: 0.607 Hz, 1.214 Hz, 2.429 Hz, or 4.857 Hz can be selected
- O Operating clock: High-speed system clock ($f_{MX}$) (only 12 MHz crystal resonator can be used)
  High-speed on-chip oscillator ($f_{IH}$)

**Note** The gain is multiplied by 2 by the digital filter.

**Caution When using the high-speed system clock ($f_{MX}$) by setting DSADCK in the PCKC register to 1, supply 12 MHz.**

Table 16-1 lists the configuration of 24-bit ΔΣ A/D converter. Figures 16-1 and 16-2 show the block diagram of 24-bit ΔΣ A/D converter, respectively.

**Table 16-1. Configuration of 24-bit ΔΣ A/D Converter**

| Item | Configuration |
|---|---|
| Analog input | 3 channels and 6 inputs (80-pin products)<br>4 channels and 8 inputs (100-pin products) |
| Internal units | Pre-amplifier block<br>ΔΣ A/D converter<br>Reference voltage generation<br>Phase adjustment circuit (PHC0, PHC1)<br>Digital filter (DF)<br>High-pass filter (HPF) |

**Figure 16-1. Block Diagram of 24-bit ΔΣ A/D Converter (100-pin Products)**

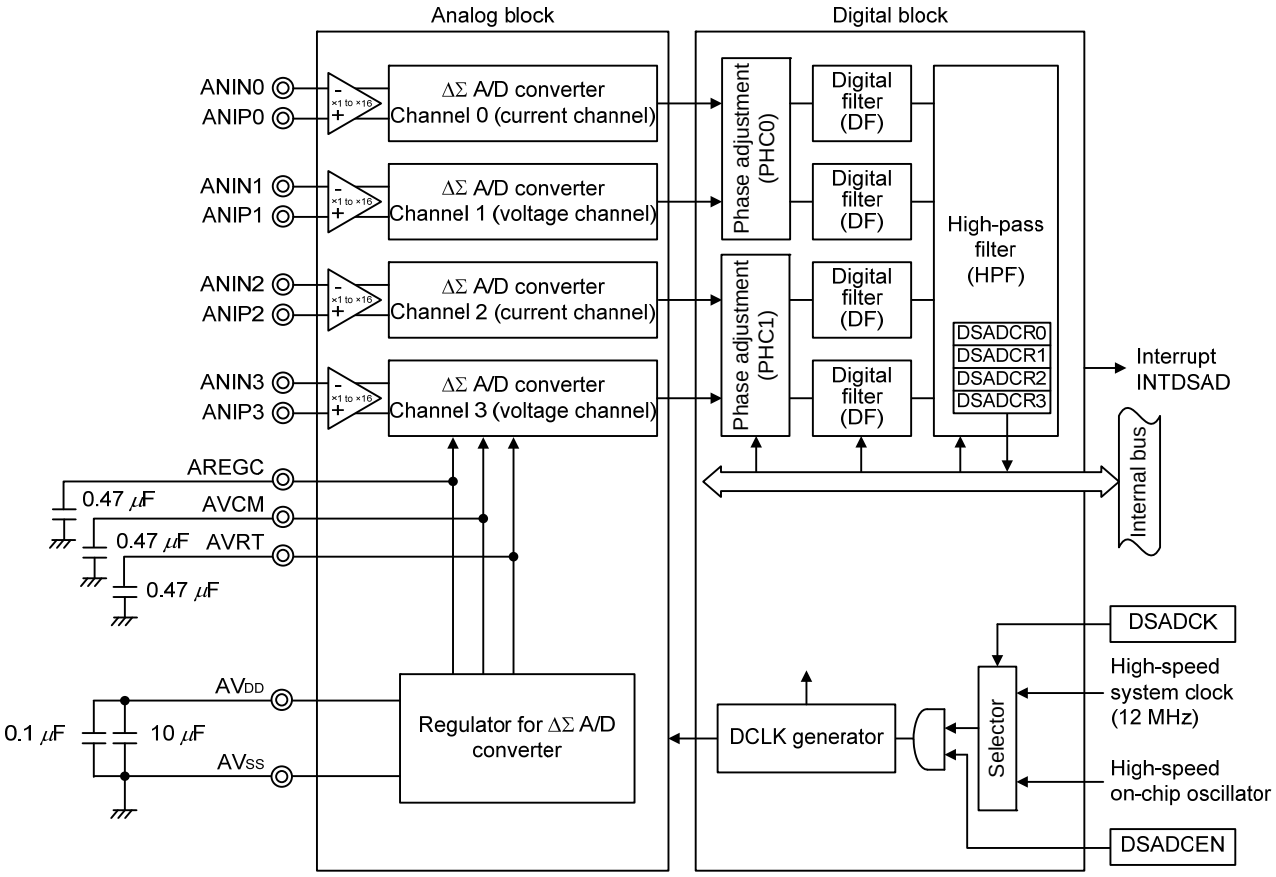
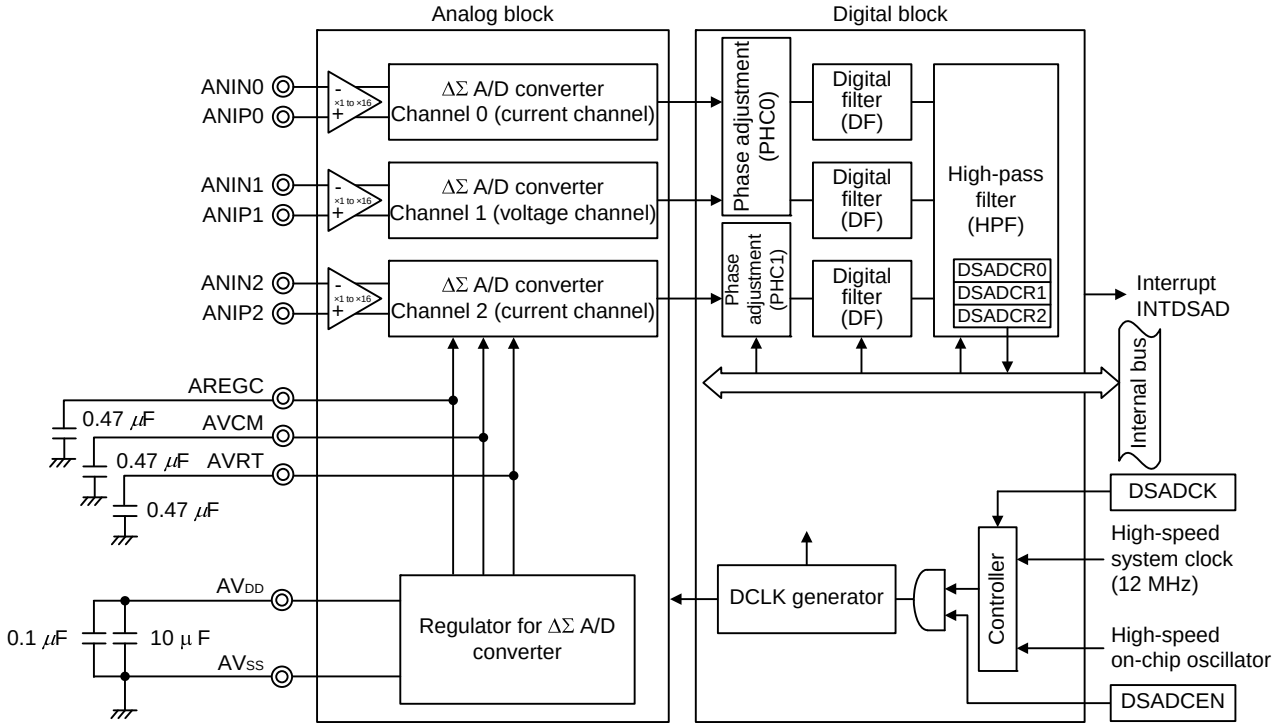



**Figure 16-2. Block Diagram of 24-bit ΔΣ A/D Converter (80-pin Products)**



### 16.1.1 I/O pins

Table 16-2 lists the I/O pins for the 24-bit ΔΣ A/D converter.

**Table 16-2. Pin Configuration**

| Name | Symbol | I/O | Function |
|---|---|---|---|
| Analog input positive pin 0 to analog input positive pin 3 | ANIPn | Input | Analog input pin for ΔΣ A/D converter (positive input)Notes 1, 2, 4 |
| Analog input negative pin 0 to analog input negative pin 3 | ANINn | Input | Analog input pin for ΔΣ A/D converter (negative input)Notes 1, 2, 4 |
| ΔΣ A/D converter power supply voltage pin | AREGC | – | ΔΣ A/D converter power supply voltage |
| Common voltage pin | AVCM | – | Common voltage |
| Reference voltage pin | AVRT | – | Reference voltage |
| Analog power supply pin | $AV_{DD}$ | – | Analog power supplyNote 3 |
| Analog GND | $AV_{SS}$ | – | Analog GND pin |

**Notes 1.** One channel inputs two signals. The ANINn pin is the negative input, while the ANIPn pin is the positive input.

**2.** Channels 0 and 2 are current channels and channels 1 and 3 are voltage channels.

**3.** Connect capacitors of 10 μF + 0.1 μF as stabilization capacitance between the $AV_{DD}$ and $AV_{SS}$ pins.

**4.** Consider the sensor delay when selecting the pin for a single phase two-wire meter.

**Remark** n = 0 to 3 for 100-pin products, n = 0 to 2 for 80-pin products

### 16.1.2 Pre-amplifier

This unit amplifies an analog input signal to be input to the ANINn and ANIPn pins.

The gain can be set to ×1, ×2, ×4, ×8, ×16, or ×32Note using the register settings.

**Note** Current channels (channel 0 and channel 2) only.

**Remark** n = 0 to 3 for 100-pin products, n = 0 to 2 for 80-pin products

### 16.1.3 ΔΣ A/D converter

Four ΔΣ A/D converter circuits are provided so that a total of four channels of analog inputs can be converted into 2-bit digital signals. These four ΔΣ A/D converter circuits operate synchronously. Each 2-bit digital value is passed through the phase adjustment circuit, the digital filter, and the high-pass filter, and then stored into the conversion result registers (DSADCR0 to DSADCR3) as the conversion result of each channel. Each time conversion of all four channels is completed, the interrupt request signal is generated to inform the CPU that the conversion result can be read. The sampling frequency (fs) can be selected as 3906.25 Hz or 1953.125 Hz. The maximum pending time and over-sampling frequency vary as follows depending on the sampling frequency. Complete reading of the ΔΣ A/D conversion result register before the maximum pending time.

| Sampling frequency (fs) | Maximum pending time | Over-sampling frequency |
|---|---|---|
| 3906.25 Hz (4 kHz sampling mode) | 192 μs | 1.5 MHz |
| 1953.125 Hz (2 kHz sampling mode) | 384 μs | 750 kHz |

### 16.1.4 Reference voltage generator

An internal reference voltage source (band-gap reference circuit) is provided and a reference voltage is output from the reference voltage output pin AVRT. Connect a capacitor of 0.47 μF as external capacitance.

### 16.1.5 Phase adjustment circuits (PHC0, PHC1)

This circuit adjusts the phase of input analog signals. The phase between analog signals is adjusted in steps (one step = 384 fs) up to 1151 steps.

Phase shifts between input analog signals occur due to external components (such as current sensors). Use the DSADPHC0 and DSADPHC1 registers to correct such phase shifts in advance, because these shifts can decrease the precision of power calculations.

A step for correcting phase shifts can be adjusted in 0.0144° units if the line frequency is 60 Hz, or in 0.0120° units if the line frequency is 50 Hz.

There are two phase adjustment circuits (PHC0, PHC1) and phase can be adjusted for up to two input signals. The combination is either ch0 or ch1 and either ch2 or ch3.

### 16.1.6 Digital filter (DF)

This unit eliminates high harmonic noise included in the ΔΣ A/D converter and thins out the data rate to 1/384.

### 16.1.7 High-pass filter (HPF)

This unit eliminates the DC component included in the input signal and the DC offset generated by the analog circuit. Whether the high-pass filter is inserted or not can be selected for each channel.

## 16.2 Registers

Table 16-3 lists the registers used for the 24-bit ΔΣ A/D converter.

**Table 16-3. Registers**

| Item | Configuration |
|---|---|
| Control registers | ΔΣ A/D converter mode register (DSADMR) |
| | ΔΣ A/D converter gain control register 0 (DSADGCR0) |
| | ΔΣ A/D converter gain control register 1 (DSADGCR1) |
| | ΔΣ A/D converter HPF control register (DSADHPFCR) |
| | ΔΣ A/D converter phase control register 0 (DSADPHCR0) |
| | ΔΣ A/D converter phase control register 1 (DSADPHCR1) |
| Registers | ΔΣ A/D converter conversion result register 0L (DSADCR0L) |
| | ΔΣ A/D converter conversion result register 0M (DSADCR0M) |
| | ΔΣ A/D converter conversion result register 0H (DSADCR0H) |
| | ΔΣ A/D converter conversion result register 1L (DSADCR1L) |
| | ΔΣ A/D converter conversion result register 1M (DSADCR1M) |
| | ΔΣ A/D converter conversion result register 1H (DSADCR1H) |
| | ΔΣ A/D converter conversion result register 2L (DSADCR2L) |
| | ΔΣ A/D converter conversion result register 2M (DSADCR2M) |
| | ΔΣ A/D converter conversion result register 2H (DSADCR2H) |
| | ΔΣ A/D converter conversion result register 3L (DSADCR3L) |
| | ΔΣ A/D converter conversion result register 3M (DSADCR3M) |
| | ΔΣ A/D converter conversion result register 3H (DSADCR3H) |
| | ΔΣ A/D converter conversion result register 0 (DSADCR0) |
| | ΔΣ A/D converter conversion result register 1 (DSADCR1) |
| | ΔΣ A/D converter conversion result register 2 (DSADCR2) |
| | ΔΣ A/D converter conversion result register 3 (DSADCR3) |
| Control registers | Peripheral enable register 1 (PER1) |
| | Peripheral clock control register (PCKC) |

### 16.2.1 ΔΣ A/D converter mode register (DSADMR)

The DSADMR register is used to set the operating mode of the ΔΣ A/D converter. This register is used to select the sampling period and the resolution of the ΔΣ A/D converter, and control powering on each channel and enabling its operation.

The DSADMR register can be set by a 16-bit memory manipulation instruction.

Reset signal generation clears this register to 0000H.

**Figure 16-3. Format of ΔΣ A/D Converter Mode Register (DSADMR)**

Address: F03C0H After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DSADMR | DSADFR | DSADTYP | 0 | 0 | DSADPON3 | DSADPON2 | DSADPON1 | DSADPON0 | 0 | 0 | 0 | 0 | DSADCE3 | DSADCE2 | DSADCE1 | DSADCE0 |

| DSADFR | Sampling frequency selection |
|---|---|
| 0 | 3906.25 Hz |
| 1 | 1953.125 Hz |
| This bit is used to select the sampling frequency. | |

| DSADTYP | Resolution selection when reading ΔΣ A/D converter conversion result register |
|---|---|
| 0 | 24-bit resolution |
| 1 | 16-bit resolution |
| When DSADTYP = 0:<br>The lower 16 bits in the ΔΣ A/D converter conversion result register can be read by reading the ΔΣ A/D converter conversion result register (DSADCRn). Read DSADCRnH as the higher 8 bits.<br>When DSADTYP = 1:<br>The higher 16 bits in the ΔΣ A/D converter conversion result register can be read by reading the ΔΣ A/D converter conversion result register (DSADCRn). | |

| DSADPONn Note | ΔΣ A/D converter power-on control (analog block) of channel n |
|---|---|
| 0 | Power down |
| 1 | Power on |

| DSADCEn Note | ΔΣ A/D converter operation enable (analog and digital blocks) of channel n |
|---|---|
| 0 | Electric charge reset |
| 1 | Normal operation |
| This bit is used to enable conversion operation of the ΔΣ A/D converter. The charge of the analog block and the conversion result of the digital block are reset. To reset the charge of the ΔΣ A/D converter normally, first set the DSADCEn bit from 1 to 0, and then wait for at least 1.4 μs before performing conversion again. | |

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

**Note** For 80-pin products, when adjusting the phase of the current channel (I1: channel 2) using a ΔΣ A/D converter phase control register 1 (DSADPHCR1), be sure to set the DSADCE3 bit of the ΔΣ A/D converter mode register (DSADMR) to 1. Otherwise, be sure to set the DSADCE3 bit to 0.

**Cautions 1. When a clock faster than 12 MHz is selected as the CPU clock ($f_{CLK}$), do not write to the DSADMR register successively. When writing to this register successively, allow at least one cycle of $f_{CLK}$ between writes. Three cycles is required until the ΔΣ A/D converter is powered down after the DSADPONn bit is set to 0. When setting the DSADPONn bit to 1 again, be sure to allow at least three cycles of $f_{CLK}$ before powering on the ΔΣ A/D converter.**

**2. Be sure to clear bits 13, 12, and 7 to 4 to "0".**

**Remark** n = 0 to 3

**Table 16-4. Channel Modes**

| DSADPON3 to DSADPON0 | Channel 3 | Channel 2 | Channel 1 | Channel 0 | Channel Mode |
|---|---|---|---|---|---|
| 0000B | – | – | – | – | Power-down |
| 0001B | – | – | – | I0 | |
| 0010B | – | – | V0 | – | |
| 0011B | – | – | V0 | I0 | Single-phase two-wire I: 1 channel, V: 1 channel |
| 0100B | – | I1 | – | – | |
| 0101B | – | I1 | – | I0 | |
| 0110B | – | I1 | V0 | – | |
| 0111B | – | I1 | V0 | I0 | Single-phase two-wire I: 2 channels, V: 1 channel |
| 1000B | V1 | – | – | – | |
| 1001B | V1 | – | – | I0 | |
| 1010B | V1 | – | V0 | – | |
| 1011B | V1 | – | V0 | I0 | |
| 1100B | V1 | I1 | – | – | Single-phase two-wire I: 1 channel, V: 1 channel |
| 1101B | V1 | I1 | – | I0 | Single-phase two-wire I: 2 channels, V: 1 channel |
| 1110B | V1 | I1 | V0 | – | |
| 1111B | V1 | I1 | V0 | I0 | Single-phase three-wire I: 2 channels, V: 2 channels |

**Caution When adjusting the phase using the ΔΣ A/D converter phase control register 0 (DSADPHCR0), be sure to set the DSADCE0 and DSADCE1 bits of the ΔΣ A/D converter mode register (DSADMR) to "1".**
**Especially when using with single-phase two-wire (I0: channel 0, I1: channel 2, V1: channel 3) and adjusting the phase of the current channel (I0: channel 0), set DSADPHCCTL0 = 1, DSADPON0 = 1, DSADCE0 = 1, DSADPON1 = 0, and DSADCE1 = 1.**
**Also, when adjusting the phase using the ΔΣ A/D converter phase control register 1 (DSADPHCR1), be sure to set the DSADCE2 and DSADCE3 bits of the ΔΣ A/D converter mode register (DSADMR) to "1".**
**Especially when using with single-phase two-wire (I0: channel 0, V0: channel 1, I1: channel 2) and adjusting the phase of the current channel (I1: channel 2), set DSADPHCCTL1 = 1, DSADPON2 = 1, DSADCE2 = 1, DSADPON3 = 0, and DSADCE3 = 1.**

### 16.2.2 ΔΣ A/D converter gain control register 0 (DSADGCR0)

The DSADGCR0 register is used to select the gain of the programmable gain amplifier of channels 0 and 1.

DSADGCR0 can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 16-4. Format of ΔΣ A/D Converter Gain Control Register 0 (DSADGCR0)**

Address: F03C2H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DSADGCR0 | 0 | DSADGAIN12 | DSADGAIN11 | DSADGAIN10 | 0 | DSADGAIN02 | DSADGAIN01 | DSADGAIN00 |

| DSADGAIN12 | DSADGAIN11 | DSADGAIN10 | Selection of programmable amplifier gain of channel 1 |
|---|---|---|---|
| Bit 6 | Bit 5 | Bit 4 | |
| 0 | 0 | 0 | PGA gain: ×1 |
| 0 | 0 | 1 | PGA gain: ×2 |
| 0 | 1 | 0 | PGA gain: ×4 |
| 0 | 1 | 1 | PGA gain: ×8 |
| 1 | 0 | 0 | PGA gain: ×16 |
| Other than above | | | Setting prohibited |
| These bits are used to control the PGA gain. The gain can be set in the range of ×1 to ×16. | | | |

| DSADGAIN02 | DSADGAIN01 | DSADGAIN00 | Selection of programmable amplifier gain of channel 0 |
|---|---|---|---|
| Bit 2 | Bit 1 | Bit 0 | |
| 0 | 0 | 0 | PGA gain: ×1 |
| 0 | 0 | 1 | PGA gain: ×2 |
| 0 | 1 | 0 | PGA gain: ×4 |
| 0 | 1 | 1 | PGA gain: ×8 |
| 1 | 0 | 0 | PGA gain: ×16 |
| 1 | 0 | 1 | PGA gain: ×32[Note] |
| Other than above | | | Setting prohibited |
| These bits are used to control the PGA gain. The gain can be set in the range of ×1 to ×32. | | | |

**Note** The gain is doubled by the digital filter (for current channels (ch0, ch2) only).

**Caution Be sure to clear bits 7 and 3 to "0".**

### 16.2.3 ΔΣ A/D converter gain control register 1 (DSADGCR1)

The DSADGCR1 register is used to select the gain of the programmable gain amplifier of channels 2 and 3.

DSADGCR1 can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 16-5. Format of ΔΣ A/D Converter Gain Control Register 1 (DSADGCR1)**

Address: F03C3H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DSADGCR1 | 0 | DSADGAIN32 | DSADGAIN31 | DSADGAIN30 | 0 | DSADGAIN22 | DSADGAIN21 | DSADGAIN20 |

| DSADGAIN32 | DSADGAIN31 | DSADGAIN30 | Selection of programmable amplifier gain of channel 3 |
|---|---|---|---|
| Bit 6 | Bit 5 | Bit 4 | |
| 0 | 0 | 0 | PGA gain: ×1 |
| 0 | 0 | 1 | PGA gain: ×2 |
| 0 | 1 | 0 | PGA gain: ×4 |
| 0 | 1 | 1 | PGA gain: ×8 |
| 1 | 0 | 0 | PGA gain: ×16 |
| Other than above | | | Setting prohibited |
| These bits are used to control the PGA gain. The gain can be set in the range of ×1 to ×16. | | | |

| DSADGAIN22 | DSADGAIN21 | DSADGAIN20 | Selection of programmable amplifier gain of channel 2 |
|---|---|---|---|
| Bit 2 | Bit 1 | Bit 0 | |
| 0 | 0 | 0 | PGA gain: ×1 |
| 0 | 0 | 1 | PGA gain: ×2 |
| 0 | 1 | 0 | PGA gain: ×4 |
| 0 | 1 | 1 | PGA gain: ×8 |
| 1 | 0 | 0 | PGA gain: ×16 |
| 1 | 0 | 1 | PGA gain: ×32[Note] |
| Other than above | | | Setting prohibited |
| These bits are used to control the PGA gain. The gain can be set in the range of ×1 to ×32. | | | |

**Note** The gain is doubled by the digital filter (for current channels (ch0, ch2) only).

**Caution Be sure to clear bits 7 and 3 to "0".**

### 16.2.4 ΔΣ A/D converter HPF control register (DSADHPFCR)

The DSADHPFCR register is used to select the cutoff frequency of the high pass filter and disable or enable the high-pass filter for each channel.

DSADHPFCR can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 16-6. Format of ΔΣ A/D Converter HPF Control Register (DSADHPFCR)**

Address: F03C5H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DSADHPFCR | DSADCOF1 | DSADCOF0 | 0 | 0 | DSADTHR3 | DSADTHR2 | DSADTHR1 | DSADTHR0 |

| DSADCOF1 | DSADCOF0 | Selection of cutoff frequency of high-pass filter |
|---|---|---|
| Bit 7 | Bit 6 | |
| 0 | 0 | 0.607 Hz |
| 0 | 1 | 1.214 Hz |
| 1 | 0 | 2.429 Hz |
| 1 | 1 | 4.857 Hz |

| DSADTHR3 | High-pass filter disable of channel 3 |
|---|---|
| 0 | High-pass filter used |
| 1 | High-pass filter not used |

| DSADTHR2 | High-pass filter disable of channel 2 |
|---|---|
| 0 | High-pass filter used |
| 1 | High-pass filter not used |

| DSADTHR1 | High-pass filter disable of channel 1 |
|---|---|
| 0 | High-pass filter used |
| 1 | High-pass filter not used |

| DSADTHR0 | High-pass filter disable of channel 0 |
|---|---|
| 0 | High-pass filter used |
| 1 | High-pass filter not used |

**Caution Be sure to clear bits 5 and 4 to "0".**

**Remark** The high-pass filter convergence time can be changed by changing the high-pass filter cut-off frequency. The convergence time decreases as the cut-off frequency increases.
The DSADEN bit of the peripheral enable register (PRE1) must be reset in order to clear the high-pass filter.

### 16.2.5 ΔΣ A/D converter phase control register 0 (DSADPHCR0)

The DSADPHCR0 register is used to select the channel for input to the phase adjustment 0 circuit and set the adjustment step.

DSADPHCR0 can be set by a 16-bit memory manipulation instruction.

Reset signal generation clears this register to 0000H.

**Figure 16-7. Format of ΔΣ A/D Converter Phase Control Register 0 (DSADPHCR0)**

Address: F03C6H After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DSADPHCR0 | DSADPHCCTL0 | 0 | 0 | 0 | 0 | DSADPHC010 | DSADPHC09 | DSADPHC08 | DSADPHC07 | DSADPHC06 | DSADPHC05 | DSADPHC04 | DSADPHC03 | DSADPHC02 | DSADPHC01 | DSADPHC00 |

| DSADPHCCTL0 | PHC0 input channel selection |
|---|---|
| 0 | Voltage channel selected (V0: channel 1) |
| 1 | Current channel selected (I0: channel 0) |

| DSADPHC010 to DSADPHC00[Note] | I0 to V0 phase adjustment |
|---|---|
| 000H | Through (no phase adjustment) |
| 001H | One step |
| ... | ... |
| 47EH | 1150 steps |
| 47FH | 1151 steps |
| These bits are used to adjust the phase of 2-bit ΔΣ A/D conversion data input from the analog block.<br>The DSADPHC010 to DSADPHC00 bits are used to specify the phase adjustment (one step = 384 fs).<br>Since the sampling frequency (3906.25 Hz) is included in the calculation of the adjustment value, the phase that can be adjusted by correcting one step is 1 [s]/(384 [fs] × 3906.25 [Hz]) = 0.6667 [μs].<br>Example: To adjust the phase of V0 by 100 μs compared to I0, the register set value will be 96H since 100/0.6667 = 150 [steps]. | |

**Note** These bits cannot be set to a value of 480H or greater.

**Cautions 1. Be sure to clear bits 14 to 11 to "0".**

**2. When adjusting the phase using the ΔΣ A/D converter phase control register 0 (DSADPHCR0), be sure to set the DSADCE0 and DSADCE1 bits of the ΔΣ A/D converter mode register (DSADMR) to "1". Especially when using with single-phase two-wire (I0: channel 0, I1: channel 2, V1: channel 3) and adjusting the phase of the current channel (I0: channel 0), set DSADPHCCTL0 = 1, DSADPON0 = 1, DSADCE0 = 1, DSADPON1 = 0, and DSADCE1 = 1.**

### 16.2.6 ΔΣ A/D converter phase control register 1 (DSADPHCR1)

The DSADPHCR1 register is used to select the channel for input to the phase adjustment 1 circuit and set the adjustment step.

DSADPHCR1 can be set by a 16-bit memory manipulation instruction.

Reset signal generation clears this register to 0000H.

**Figure 16-8. Format of ΔΣ A/D Converter Phase Control Register 1 (DSADPHCR1)**

Address: F03C8H After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DSADPHCR1 | DSADPHCCTL1 | 0 | 0 | 0 | 0 | DSADPHC110 | DSADPHC19 | DSADPHC18 | DSADPHC17 | DSADPHC16 | DSADPHC15 | DSADPHC14 | DSADPHC13 | DSADPHC12 | DSADPHC11 | DSADPHC10 |

| DSADPHCCTL1 | PHC1 input channel selection |
|---|---|
| 0 | Voltage channel selected (V1: channel 3) |
| 1 | Current channel selected (I1: channel 2) |

| DSADPHC110 to DSADPHC10[Note] | I1 to V1 phase adjustment |
|---|---|
| 000H | Through (no phase adjustment) |
| 001H | One step |
| ... | ... |
| 47EH | 1150 steps |
| 47FH | 1151 steps |
| These bits are used to adjust the phase of 2-bit ΔΣ A/D conversion data input from the analog block.<br>The DSADPHC110 to DSADPHC10 bits are used to specify the phase adjustment (one step = 384 fs).<br>Since the sampling frequency (3906.25 Hz) is included in the calculation of the adjustment value, the phase that can be adjusted by correcting one step is 1 [s]/(384 [fs] × 3906.25 [Hz]) = 0.6667 [μs].<br>Example: To adjust the phase of V1 by 100 μs compared to I1, the register set value will be 96H since 100/0.6667 = 150 [steps]. | |

**Note** These bits cannot be set to a value of 480H or greater.

**Cautions 1. Be sure to clear bits 14 to 11 to "0".**

**2. When adjusting the phase using the ΔΣ A/D converter phase control register 1 (DSADPHCR1), be sure to set the DSADCE2 and DSADCE3 bits of the ΔΣ A/D converter mode register (DSADMR) to "1". Especially when using with single-phase two-wire (I0: channel 0, V0: channel 1, I1: channel 2) and adjusting the phase of the current channel (I1: channel 2), set DSADPHCCTL1 = 1, DSADPON2 = 1, DSADCE2 = 1, DSADPON3 = 0, and DSADCE3 = 1.**

### 16.2.7 ΔΣ A/D converter conversion result register n (DSADCRnL, DSADCRnM, DSADCRnH) (n = 0, 1, 2, 3)

The DSADCRn (H/M/L) registers are 24-bit registers used to retain the conversion results of the ΔΣ A/D converter of each channel.

The DSADCRnL, DSADCRnM, and DSADCRnH registers can be read individually by an 8-bit manipulation instruction. Reading of the conversion result of the ΔΣ A/D converter differs depending on the setting of the DSADTYP bit in the ΔΣ A/D converter mode register (DSADMR).

Setting the DSADCEn bit in the ΔΣ A/D converter mode register (DSADMR) to 0 or reset signal generation clears the DSADCRnL, DSADCRnM, and DSADCRnH registers to 00H.

**Figure 16-9. Format of ΔΣ A/D Converter Conversion Result Register n (DSADCRnL, DSADCRnM, DSADCRnH) (n = 0, 1, 2, 3)**

Address: F03D0H (DSADCR0L) F03D1H (DSADCR0M) F03D2H (DSADCR0H)
F03D4H (DSADCR1L) F03D5H (DSADCR1M) F03D6H (DSADCR1H)
F03D8H (DSADCR2L) F03D9H (DSADCR2M) F03DAH (DSADCR2H)
F03DCH (DSADCR3L) F03DDH (DSADCR3M) F03DEH (DSADCR3H)

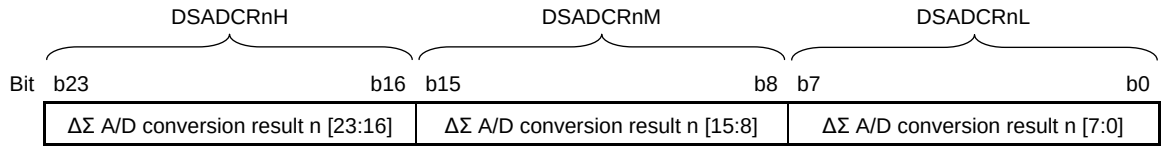
After reset: 00H R

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DSADCRnH | DSADCRnH [7:0] | | | | | | | |

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DSADCRnM | DSADCRnM [7:0] | | | | | | | |

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DSADCRnL | DSADCRnL [7:0] | | | | | | | |

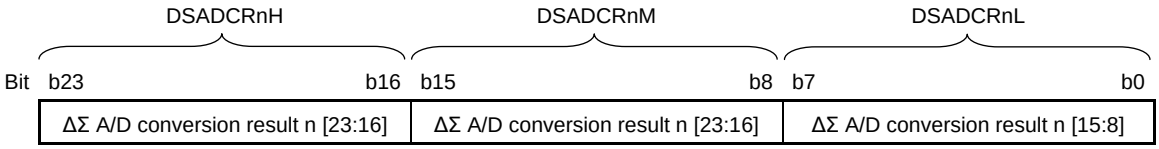
- When 24-bit resolution is set (DSADTYP in the DSADMR register = 0)




| Bit | Symbol | Conversion result of channel n |
|---|---|---|
| b7 to b0 | DSADCRnL [7:0] | Conversion result bits 7 to 0 of channel n |
| b15 to b8 | DSADCRnM [7:0] | Conversion result bits 15 to 8 of channel n |
| b23 to b16 | DSADCRnH [7:0] | Conversion result bits 23 to 16 of channel n |

(**Caution** is listed on the next page.)

- When 16-bit resolution is set (DSADTYP in the DSADMR register = 1)




| Bit | Symbol | Conversion result of channel n |
|---|---|---|
| b7 to b0 | DSADCRnL [7:0] | Conversion result bits 15 to 8 of channel n |
| b15 to b8 | DSADCRnM [7:0] | Conversion result bits 23 to 16 of channel n |
| b23 to b16 | DSADCRnH [7:0] | Conversion result bits 23 to 16 of channel n |

**Caution Be sure to read the ΔΣ A/D converter conversion result register within its maximum pending time after the ΔΣ A/D conversion end interrupt is generated.**

### 16.2.8 ΔΣ A/D converter conversion result register n (DSADCRn) (n = 0, 1, 2, 3)

The DSADCRn register is used to access the conversion result of each channel using a 16-bit memory manipulation instruction.

The DSADCRn register can be read by a 16-bit memory manipulation instruction. Reading of the conversion result of the ΔΣ A/D converter differs depending on the setting of the DSADTYP bit in the ΔΣ A/D converter mode register (DSADMR).

Setting the DSADCEn bit in the ΔΣ A/D converter mode register (DSADMR) to 0 or reset signal generation clears the DSADCRn register to 0000H.

**Figure 16-10. Format of ΔΣ A/D Converter Conversion Result Register n (DSADCRn) (n = 0, 1, 2, 3)**

Address: F03D0H (DSADCR0) F03D4H (DSADCR1) F03D8H (DSADCR2)
F03DCH (DSADCR3)
After reset: 0000H R

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DSADCRn | DSADCRn [15:0] | | | | | | | | | | | | | | | |

- When 24-bit resolution is set (DSADTYP in the DSADMR register = 0)[Note]

| Bit | Symbol | Conversion result of channel n |
|---|---|---|
| b15 to b0 | DSADCRn [15:0] | Conversion result bits 15 to 0 of channel n |

- When 16-bit resolution is set (DSADTYP in the DSADMR register = 1)[Note]

| Bit | Symbol | Conversion result of channel n |
|---|---|---|
| b15 to b0 | DSADCRn [15:0] | Conversion result bits 23 to 8 of channel n |

**Note** Access to the DSADCRn register changes depending on the setting of the DSADTYP bit in the DSADMR register.
- DSADTYP = 0: The lower 16 bits can be read. Read DSADCRnH as the higher 8 bits.
- DSADTYP = 1: The higher 16 bits can be read.

**Caution Be sure to read the ΔΣ A/D converter conversion result register within its maximum pending time after the ΔΣ A/D conversion end interrupt is generated.**

### 16.2.9 Peripheral enable register 1 (PER1)

The PER1 register is used to enable or disable supplying the clock to the peripheral hardware. Clock supply to the hardware that is not used is also stopped so as to decrease the power consumption and noise.

To use the 24-bit ΔΣ A/D converter, be sure to set bit 0 (DSADCEN) to 1.

The PER1 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 16-11. Format of Peripheral Enable Register 1 (PER1)**

Address: F007AH After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PER1 | TMKAEN | FMCEN | CMPEN | OSDCEN | DTCEN | 0 | 0 | DSADCEN |

| DSADCEN | Control of 24-bit ΔΣ A/D converter input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFR used by the 24-bit ΔΣ A/D converter cannot be written.<br>• The 24-bit ΔΣ A/D converter is in the reset status. |
| 1 | Enables input clock supply.<br>• SFR used by the 24-bit ΔΣ A/D converter can be read and written. |

**Cautions 1. When setting the 24-bit ΔΣ A/D converter, be sure to set the DSADCEN bit to 1 first. If DSADCEN = 0, writing to a control register of the ΔΣ A/D converter is ignored, and all read values are default values.**

**2. Be sure to clear bits 2 and 1 to "0".**

**3. When a high-speed on-chip oscillator is selected as the input clock, be sure to run the high-speed on-chip oscillator clock frequency correction function to input clock with high frequency precision.**

### 16.2.10 Peripheral clock control register (PCKC)

The PCKC register is used to control peripheral clocks. Set bit 0 to select a clock for the 24-bit ΔΣ A/D converter.

The PCKC register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 16-12. Format of Peripheral Clock Control Register (PCKC)**

Address: F0098H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PCKC | 0 | 0 | 0 | 0 | 0 | 0 | 0 | DSADCK |

| DSADCK | Selection of operation clock for 24-bit ΔΣ A/D converter |
|---|---|
| 0 | Supply high-speed on-chip oscillator clock ($f_{IH}$). (Stop $f_{MX}$ supply)Note 1 |
| 1 | Supply high-speed system clock ($f_{MX}$)Note 2 |

**Notes 1.** When selecting the high-speed on-chip oscillator clock, be sure to run the high-speed on-chip oscillator clock frequency correction function.

**2.** Only a 12 MHz crystal oscillator can be used as the high-speed system clock frequency ($f_{MX}$).

**Caution Be sure to clear bits 7 to 1 to "0".**

## 16.3 Operation

The 24-bit ΔΣ A/D converter has the digital signal input pins for four ΔΣ A/D converter conversion results. By passing 2-bit values obtained from these ΔΣ A/D converter conversion results through the digital filter, the value is converted into 24-bit digital values.

The mode setting of the ΔΣ A/D converter of the analog block depends on the values of the DSADMR, DSADGCR0, and DSADGCR1 register. Table 16-5 lists the mode settings.

**Table 16-5. Mode Settings**

| Signal/Mode | <1> Normal | <2> ΔΣ A/D Conversion Stop | <3> Power-down |
|---|---|---|---|
| DSADGAINn2 to DSADGAINn0 | Any value | Any value | Any value |
| DSADPONn | 1 | 1 | 0 |
| DSADCEn | 1 | 0 | 0 |

**Remark** n = 0 to 3

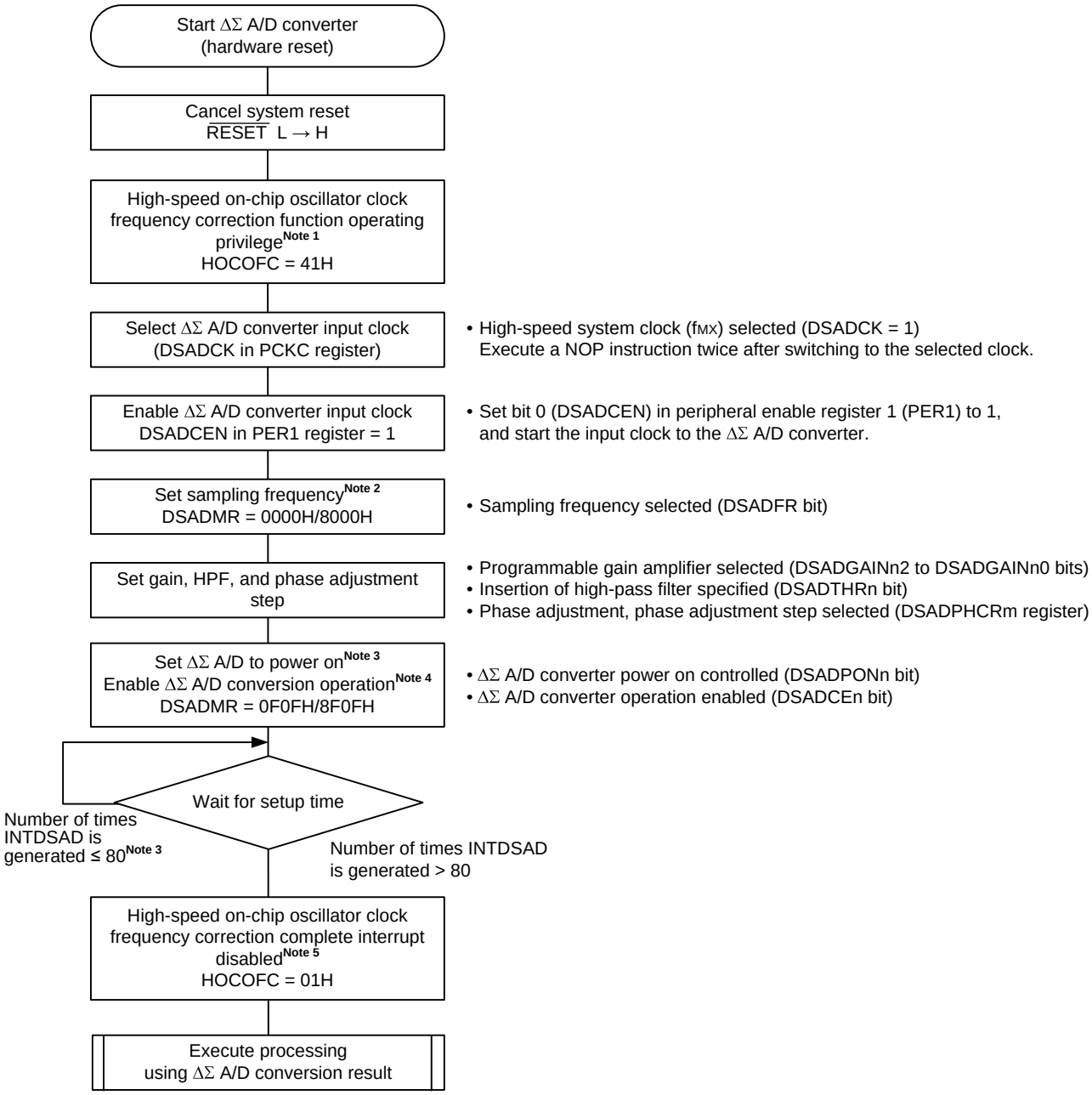
### 16.3.1 Operation of 24-bit ΔΣ A/D converter

When selecting the high-speed on-chip oscillator clock ($f_{IH}$), be sure to run the high-speed on-chip oscillator clock frequency correction function according to **6.3.2 Operation procedure** before running the ΔΣ A/D converter.

When selecting the high-speed system clock ($f_{MX}$), execute a NOP instruction twice after switching to the selected clock.

The 24-bit converter starts operating when the DSADPONn bit (n = 0 to 3) and the DSADCEn bit in the DSADMR register are set to 1. The setup time of the analog block and digital filter block is required after power on and start of conversion. Perform initialization in accordance with the flowchart below.

**Figure 16-13. Initialization Flowchart**




(**Note** and **Remark** are listed on the next page.)

**Notes 1.** When selecting the high-speed on-chip oscillator clock, be sure to run the high-speed on-chip oscillator clock frequency correction function before running the ΔΣ A/D converter.

**2.** Set the sampling frequency while the ΔΣ A/D converter is powered down.

**3.** The setup time (the number of times INTDSAD is to be generated) when DSADPONn is set to 0 and then 1 will be officially determined after evaluation.

**4.** If the ΔΣ A/D converter is temporarily stopped for initialization (DSADCEn = 0 with DSADPONn = 1) and then restarted, it is necessary to wait for a certain setup time. In this case, since stabilization time is necessary for the converter, wait for one INTDSAD to be generated as the setup time.
To initialize the ΔΣ A/D converter, make sure that DSADCEn remains 0 for at least 1.4 μs.

**5.** Perform only when selecting the high-speed on-chip oscillator clock.

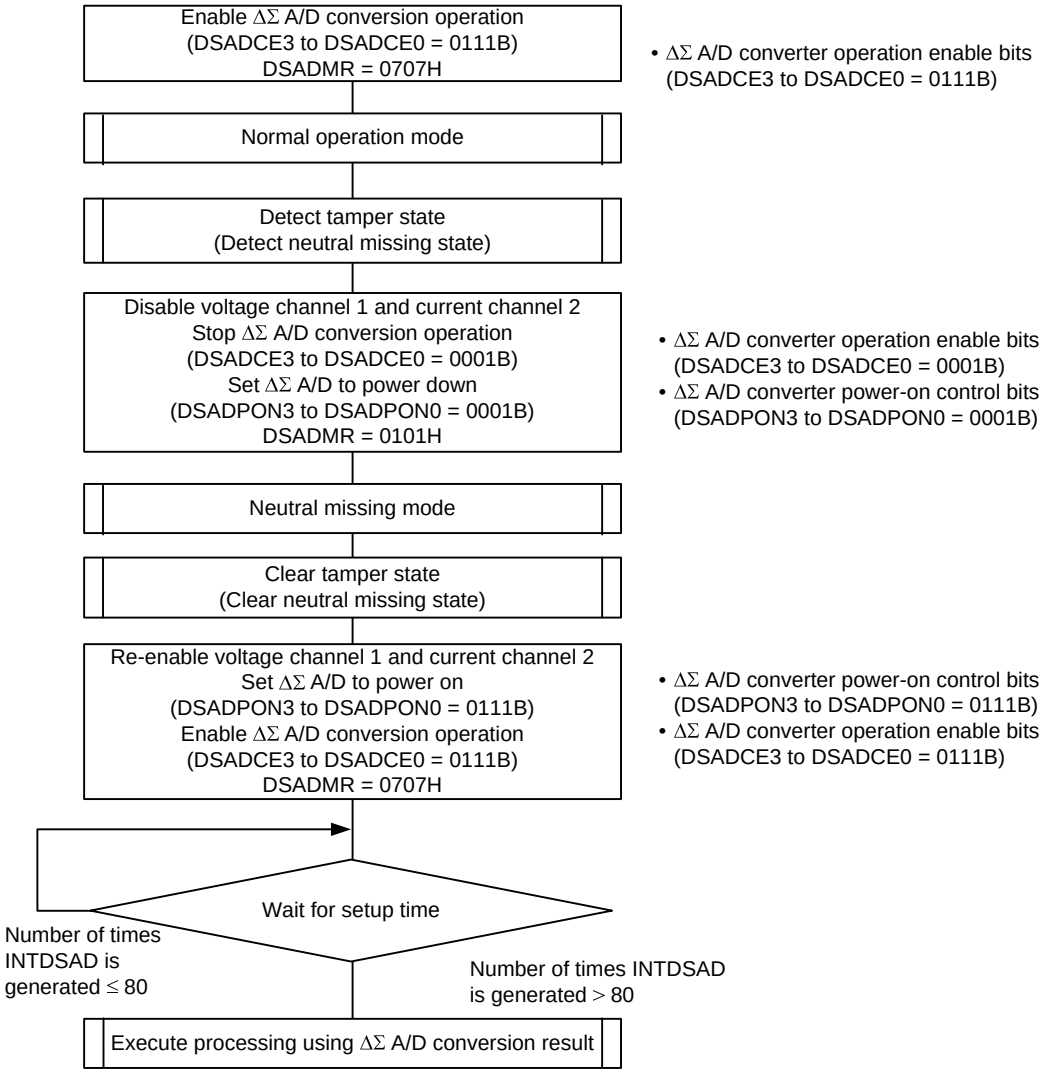
**Remark** n = 0 to 3; m = 0, 1

### 16.3.2 Procedure for switching from normal operation mode to neutral missing mode

Figure 16-14 shows the procedure for switching from normal operation (with anti-tamper) (a total of three: current channel 0, voltage channel 1, and current channel 2 operate) to neutral missing mode (only current channel 0 operates), in single-phase two-wire mode.

In neutral missing mode, there are cases when only current channel 0 operates and only current channel 2 operates. Use the same procedure when switching the mode.

**Figure 16-14. Procedure for Switching from Normal Operation Mode to Neutral Missing Mode**

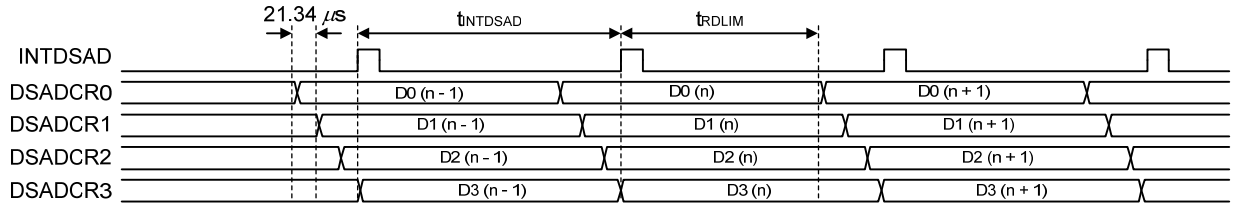


### 16.3.3 Interrupt operation

When ΔΣ A/D conversion is enabled, conversion of the signals on the four channels of analog input pins (ANINn and ANIPn) is started. Four sets of ΔΣ A/D converter circuits are provided, and each of which executes conversion. Each time conversion of all four channels is completed, the interrupt request signal (INTDSAD) is generated to inform the CPU that the conversion result can be read.

The generation cycle of INTDSAD (tINTDSAD) differs depending on the sampling frequency specified by the DSADFR bit in the DSADMR register. The maximum pending time for reading the ΔΣ A/D converter conversion result register n (DSADCRn) by interrupt servicing is as shown in Figure 16-15. Complete reading of the DSADCRn register within this time.

**Figure 16-15. Timing of Generation of INTDSAD Signal and Storing in DSADCRn Register**




$t_{INTDSAD}$: Interrupt generation cycle: 256 μs (DSADFR = 0)
512 μs (DSADFR = 1)
$t_{RDLIM}$: DSADCR read pending time (max): 192 μs (DSADFR = 0)
384 μs (DSADFR = 1)

**Remark** n = 0 to 3

### 16.3.4 Operation in standby state

In STOP operation mode, the ΔΣ A/D converter and the digital filter do not operate. To reduce current consumption, stop operation of the ΔΣ A/D converter (DSADCEn in the DSADMR register = 0000B) and power down the ΔΣ A/D converter (DSADPONn in the DSADMR register = 0000B) before executing the STOP instruction.

**Remark** n = 0 to 3

## 16.4 Notes on Using 24-Bit ΔΣ A/D Converter

### 16.4.1 External pins

The $AV_{DD}$ pin is the analog power supply pin of the ΔΣ A/D converter. Always keep the voltage on this pin the same as that on the $V_{DD}$ pin even when the ΔΣ A/D converter is not used.

The $AV_{SS}$ pin is the ground power supply pin of the ΔΣ A/D converter. Always keep the voltage on this pin the same as that on the $V_{SS}$ pin even when the ΔΣ A/D converter is not used.

### 16.4.2 SFR access

(1) Read the DSADCRn register by ΔΣ A/D conversion end interrupt (INTDSAD) servicing. If the DSADCRn register is read before a ΔΣ A/D conversion end interrupt is generated, an illegal value may be read because of a conflict between storing the conversion value in the DSADCRn register and reading the register. The period of the INTDSAD processing during which the DSADCRn register is read is 192 μs (when DSADFR is set to 0) or 384 μs (when DSADFR is set to 1), so complete reading of the register within this time.

(2) After powering on the ΔΣ A/D converter (DSADPONn in the DSADMR register = 1), internal setup time is necessary. Consequently, the data of the first 80 conversions is invalid.

(3) Setup time is also necessary when the ΔΣ A/D converter has been temporarily stopped for initialization (by clearing the DSADCEn bit in the DSADMR register to 0 with DSADPONn = 1) and then restarted. In this case, since stabilization time is necessary for the converter, wait for one INTDSAD to be generated as the setup time. To initialize the ΔΣ A/D converter, make sure that DSADCEn remains 0 for at least 1.4 μs.

(4) The time required for the correct data to be output after the conversion operation has been enabled (by setting the DSADCEn bit to 1) differs depending on the analog input status at that time. This is because the stabilization time of the high-pass filter changes depending on the analog input status.

(5) Set the conversion rate while the DSADPONn bit in the DSADMR register is 0. Be sure to set the gain and the DSADPHCR0 and DSADPHCR1 registers while the ΔΣ A/D converter is stopped (DSADCEn = 0).

(6) Since the DSADCRn register is initialized when the DSADCEn bit is 0, read the DSADCRn register when the DSADCEn bit is 1.

(7) Clear the DSADPONn bit in the DSADMR register to 0 before shifting to software STOP mode. If software STOP mode is entered with the DSADPONn bit set to 1, a current will flow.

**Remark** n = 0 to 3

### 16.4.3 Setting operating clock

When using the high-speed system clock ($f_{MX}$) by setting DSADCK in the PCKC register to 1, supply 12 MHz.

Also, when selecting the high-speed on-chip oscillator clock ($f_{IH}$), be sure to run the high-speed on-chip oscillator frequency correction function.

**Cautions 1. Count the INTDSAD signal 80 times after the ΔΣ A/D converter is started and then load the converted data when the next INTDSAD signal is generated. The setup time is subject to change. Consult Renesas Electronics before using the setup time.**

**2. Thoroughly evaluate the stabilization time in the environment in which the ΔΣ A/D converter is used.**

To stop the 24-bit ΔΣ A/D converter while it is operating, set the DSADPON3 to DSADPON0 bits in the DSADMR register to 0000B, and then set the DSADCEN bit in the PER1 register to 0.

### 16.4.4 Phase adjustment for single-phase two-wire

When adjusting the phase using the ΔΣ A/D converter phase control register 0 (DSADPHCR0), be sure to set the DSADCE0 and DSADCE1 bits of the ΔΣ A/D converter mode register (DSADMR) to “1”.

Especially when using with single-phase two-wire (I0: channel 0, I1: channel 2, V1: channel 3) and adjusting the phase of the current channel (I0: channel 0), set DSADPHCCTL0 = 1, DSADPON0 = 1, DSADCE0 = 1, DSADPON1 = 0, and DSADCE1 = 1.

Also, when adjusting the phase using the ΔΣ A/D converter phase control register 1 (DSADPHCR1), be sure to set the DSADCE2 and DSADCE3 bits of the ΔΣ A/D converter mode register (DSADMR) to “1”.

Especially when using with single-phase two-wire (I0: channel 0, V0: channel 1, I1: channel 2) and adjusting the phase of the current channel (I1: channel 2), set DSADPHCCTL1 = 1, DSADPON2 = 1, DSADCE2 = 1, DSADPON3 = 0, and DSADCE3 = 1.

# CHAPTER 17 COMPARATOR

The comparator compares a reference input voltage to an analog input voltage. It consists of two independent comparators: comparator 0 and comparator 1.

## 17.1 Functions of Comparator

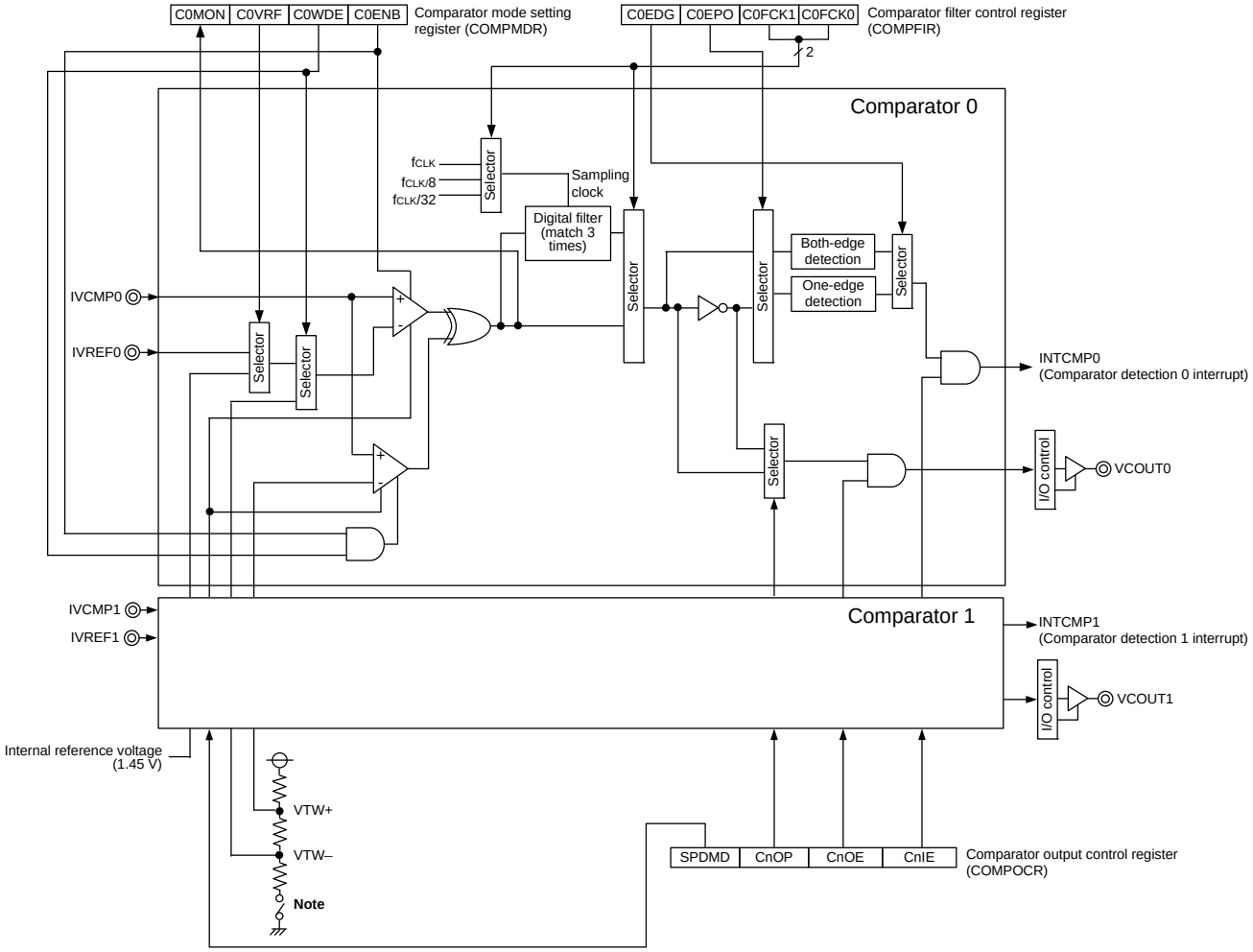
The comparator has the following functions.

- Comparator high-speed mode, comparator low-speed mode, or comparator window mode can be selected.
- The external reference voltage input or internal reference voltage can be selected as the reference voltage.
- The canceling width of the noise canceling digital filter can be selected.
- An interrupt signal can be generated by detecting an active edge of the comparator output.

## 17.2 Configuration of Comparator

Figure 17-1 shows the compare block diagram.

**Figure 17-1. Comparator Block Diagram**




**Note** When either or both of the C0WDE and C1WDE bits are set to 1, this switch is turned on and the divider resistors for generating the comparison voltage are enabled.

**Remark** n = 0, 1

## 17.3 Registers Controlling Comparator

Table 17-3 lists the registers controlling comparator.

**Table 17-1. Registers Controlling Comparator**

| Register Name | Symbol |
|---|---|
| Peripheral enable register 1 | PER1 |
| Comparator mode setting register | COMPMDR |
| Comparator filter control register | COMPFIR |
| Comparator output control register | COMPOCR |
| A/D port configuration register | ADPC |
| Port mode registers 0, 2 | PM0, PM2 |
| Port registers 0, 2 | P0, P2 |

### 17.3.1 Peripheral enable register 1 (PER1)

The PER1 register is used to enable or disable use of each peripheral hardware macro. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

When the Comparator is used, be sure to set bit 5 (CMPEN) of this register to 1.

The PER1 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 17-2. Format of Peripheral Enable Register 1 (PER1)**

Address: F007AH After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PER1 | TMKAEN | FMCEN | CMPEN | OSDCEN | DTCEN | 0 | 0 | DSADCEN |

| CMPEN | Control of comparator input clock |
|---|---|
| 0 | Stops input clock supply.<br>• SFR used by the Comparator cannot be written.<br>• The Comparator is in the reset status. |
| 1 | Supplies input clock.<br>• SFR used by the Comparator can be read/written. |

**Cautions 1. When setting the comparator, be sure to set the CMPEN bit to 1 first. If CMPEN = 0, writing to a control register of the comparator is ignored, and all read values are default values (except for A/D port configuration register (ADPC), port mode registers 0, 2 (PM0, PM2), port registers 0, 2 (P0, P2)).**

- **Comparator mode setting register (COMPMDR)**
- **Comparator filter control register (COMPFIR)**
- **Comparator output control register (COMPOCR)**

**2. Be sure to clear the bits 2 and 1 to "0".**

### 17.3.2 Comparator mode setting register (COMPMDR)

**Figure 17-3. Format of Comparator Mode Setting Register (COMPMDR)**

Address: F0340H After reset: 00H R/W

| Symbol | <7> | 6 | 5 | <4> | <3> | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| COMPMDR | C1MON | C1VRF | C1WDE | C1ENB | C0MON | C0VRF | C0WDE | C0ENB |

| C1MON | Comparator 1 monitor flag[Notes 3, 7] |
|---|---|
| 0 | In standard mode:<br>IVCMP1 < comparator 1 reference voltage or comparator 1 stopped<br>In window mode:<br>IVCMP1 < low-voltage reference or IVCMP1 > high-voltage reference |
| 1 | In standard mode:<br>IVCMP1 > comparator 1 reference voltage<br>In window mode:<br>Low-voltage reference < IVCMP1 < high-voltage reference |

| C1VRF | Comparator 1 reference voltage selection[Notes 1, 4, 5, 6] |
|---|---|
| 0 | Comparator 1 reference voltage is IVREF1 input |
| 1 | Comparator 1 reference voltage is internal reference voltage (1.45 V) |

| C1WDE | Comparator 1 window mode selection[Note 2] |
|---|---|
| 0 | Comparator 1 standard mode |
| 1 | Comparator 1 window mode |

| C1ENB | Comparator 1 operation enable |
|---|---|
| 0 | Comparator 1 operation disabled |
| 1 | Comparator 1 operation enabled |

| C0MON | Comparator 0 monitor flag[Notes 3, 7] |
|---|---|
| 0 | In standard mode:<br>IVCMP0 < comparator 0 reference voltage or comparator 0 stopped<br>In window mode:<br>IVCMP0 < low-voltage reference or IVCMP0 > high-voltage reference |
| 1 | In standard mode:<br>IVCMP0 > comparator 0 reference voltage<br>In window mode:<br>Low-voltage reference < IVCMP0 < high-voltage reference |

(**Notes** are listed on the next page.)

| C0VRF | Comparator 0 reference voltage selection[Notes 1, 4, 5, 6] |
|---|---|
| 0 | Comparator 0 reference voltage is IVREF0 input |
| 1 | Comparator 0 reference voltage is internal reference voltage (1.45 V) |

| C0WDE | Comparator 0 window mode selection[Note 2] |
|---|---|
| 0 | Comparator 0 standard mode |
| 1 | Comparator 0 window mode |

| C0ENB | Comparator 0 operation enable |
|---|---|
| 0 | Comparator 0 operation disabled |
| 1 | Comparator 0 operation enabled |

**Notes 1.** Valid only when standard mode is selected. In window mode, the reference voltage in the comparator is selected regardless of the setting of this bit.

**2.** Window mode cannot be set when low-speed mode is selected (the SPDMD bit in the COMPOCR register is 0).

**3.** The initial value is 0 immediately after a reset is released. However, the value is undefined when C0ENB is set to 0 and C1ENB is set to 0 after operation of the comparator is enabled once.

**4.** The internal reference voltage (1.45 V) can be selected in HS (high-speed main) mode.

**5.** Do not select the internal reference voltage in STOP mode.

**6.** Do not select the internal reference voltage when the subsystem clock ($f_{XT}$) is selected as the CPU clock and both the high-speed system clock ($f_{MX}$) and high-speed on-chip oscillator clock ($f_{IH}$) are stopped.

**7.** Writing to this bit is ignored.

### 17.3.3 Comparator filter control register (COMPFIR)

**Figure 17-4. Format of Comparator Filter Control Register (COMPFIR)**

Address: F0341H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| COMPFIR | C1EDG | C1EPO | C1FCK1 | C1FCK0 | C0EDG | C0EPO | C0FCK1 | C0FCK0 |

| C1EDG | Comparator 1 edge detection selection[Note 1] |
|---|---|
| 0 | Interrupt request by comparator 1 one-edge detection |
| 1 | Interrupt request by comparator 1 both-edge detection |

| C1EPO | Comparator 1 edge polarity switching[Note 1] |
|---|---|
| 0 | Interrupt request at comparator 1 rising edge |
| 1 | Interrupt request at comparator 1 falling edge |

| C1FCK1 | C1FCK0 | Comparator 1 filter selection[Note 1] |
|---|---|---|
| 0 | 0 | No comparator 1 filter |
| 0 | 1 | Comparator 1 filter enabled, sampling at $f_{CLK}$ |
| 1 | 0 | Comparator 1 filter enabled, sampling at $f_{CLK}/8$ |
| 1 | 1 | Comparator 1 filter enabled, sampling at $f_{CLK}/32$ |

| C0EDG | Comparator 0 edge detection selection[Note 2] |
|---|---|
| 0 | Interrupt request by comparator 0 one-edge detection |
| 1 | Interrupt request by comparator 0 both-edge detection |

| C0EPO | Comparator 0 edge polarity switching[Note 2] |
|---|---|
| 0 | Interrupt request at comparator 0 rising edge |
| 1 | Interrupt request at comparator 0 falling edge |

| C0FCK1 | C0FCK0 | Comparator 0 filter selection[Note 2] |
|---|---|---|
| 0 | 0 | No comparator 0 filter |
| 0 | 1 | Comparator 0 filter enabled, sampling at $f_{CLK}$ |
| 1 | 0 | Comparator 0 filter enabled, sampling at $f_{CLK}/8$ |
| 1 | 1 | Comparator 0 filter enabled, sampling at $f_{CLK}/32$ |

**Notes 1.** If bits C1FCK1, C1FCK0, C1EPO, and C1EDG are changed, a comparator 1 interrupt request may be generated. Also, be sure to clear (0) bit 7 (CMPIF1) in interrupt request flag register 2L (IF2L). If bits C1FCK1 and C1FCK0 are changed from 00B (no comparator 1 filter) to a value other than 00B (comparator 1 filter enabled), allow four sampling times to elapse until the filter output is updated, and then use the comparator 1 interrupt request.

**2.** If bits C0FCK1, C0FCK0, C0EPO, and C0EDG are changed, a comparator 0 interrupt request may be generated. Also, be sure to clear (0) bit 6 (CMPIF0) in request flag register 2L (IF2L). If bits C0FCK1 and C0FCK0 are changed from 00B (no comparator 0 filter) to a value other than 00B (comparator 0 filter enabled), allow four sampling times to elapse until the filter output is updated, and then use the comparator 0 interrupt request.

### 17.3.4 Comparator output control register (COMPOCR)

**Figure 17-5. Format of Comparator Output Control Register (COMPOCR)**

Address: F0342H After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | 3 | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| COMPOCR | SPDMD | C1OP | C1OE | C1IE | 0 | C0OP | C0OE | C0IE |

| SPDMD | Comparator speed selection[Note 1] |
|---|---|
| 0 | Comparator low-speed mode |
| 1 | Comparator high-speed mode |

| C1OP | VCOUT1 output polarity selection |
|---|---|
| 0 | Comparator 1 output is output to VCOUT1 |
| 1 | Inverted comparator 1 output is output to VCOUT1 |

| C1OE | VCOUT1 pin output enable |
|---|---|
| 0 | Comparator 1 VCOUT1 pin output disabled |
| 1 | Comparator 1 VCOUT1 pin output enabled |

| C1IE | Comparator 1 interrupt request enable[Note 2] |
|---|---|
| 0 | Comparator 1 interrupt request disabled |
| 1 | Comparator 1 interrupt request enabled |

| C0OP | VCOUT0 output polarity selection |
|---|---|
| 0 | Comparator 0 output is output to VCOUT0 |
| 1 | Inverted comparator 0 output is output to VCOUT0 |

| C0OE | VCOUT0 pin output enable |
|---|---|
| 0 | Comparator 0 VCOUT0 pin output disabled |
| 1 | Comparator 0 VCOUT0 pin output enabled |

| C0IE | Comparator 0 interrupt request enable[Note 3] |
|---|---|
| 0 | Comparator 0 interrupt request disabled |
| 1 | Comparator 0 interrupt request enabled |

**Notes 1.** When rewriting the SPDMD bit, be sure to set the CiENB bit (i = 0 or 1) in the COMPMDR register to 0 in advance.

**2.** If C1IE is changed from 0 (interrupt request disabled) to 1 (interrupt request enabled), bit 7 (CMPIF1) in interrupt request flag register 2L (IF2L) may be set to 1 (interrupt requested), so be sure to clear (0) bit 7 (CMPIF1) in interrupt request flag register 2L (IF2L) before using an interrupt.

**3.** If C0IE is changed from 0 (interrupt request disabled) to 1 (interrupt request enabled), bit 6 (CMPIF0) in interrupt request flag register 2L (IF2L) may be set to 1 (interrupt requested), so be sure to clear (0) bit 6 (CMPIF0) in interrupt request flag register 2L (IF2L) before using an interrupt.

### 17.3.5 Registers controlling port functions of analog input pins

When using the IVCMP0, IVCMP1, IVREF0, and IVREF1 pins for analog input of the comparator, specify the A/D port configuration register (ADPC) corresponding to each port as analog input channel and set the port mode register (PMxx) to analog input.

When using the VCOUT0 and VCOUT1 functions, set the registers (port mode register (PMxx) and port register (Pxx) that control the port functions) shared with the target channels. For details, see **4.3.1 Port mode registers (PMxx)** and **4.3.2 Port registers (Pxx)**.

## 17.4 Operation

Comparator 0 and comparator 1 operate independently. Their setting methods and operations are the same. Table 17-2 lists the Procedure for Setting Comparator Associated Registers.

**Table 17-2. Procedure for Setting Comparator Associated Registers**

| Step | Register | Bit | Setting Value | | |
|---|---|---|---|---|---|
| 1 | PER1 | CMPEN | 1 (input clock supply) | | |
| 2 | ADPC | ADPC2 to ADPC0 | Select the function of pins IVCMPi and IVREFi.<br>Set the ADPC2 to ADPC0 bits to 101B, 110B, or 000B (analog input).<br>Set the PM2n bit to 1 (input mode).<br>See **17.3.5 Registers controlling port functions of analog input pins**. | | |
| | PM2 | PM2n | | | |
| 3 | COMPOCR | SPDMD | Select the comparator response speed (0: Low-speed mode/1: High-speed mode).Note 1 | | |
| 4 | COMPMDR | CiWDE | 0 (standard mode) | | 1 (window mode)Note 2. |
| | | CiVRF | 0<br>(Reference = IVREFi input) | 1<br>(Reference = internal reference voltage (1.45 V))Note 4 | Window comparator operation (reference = internal VREF) |
| | | CiENB | 1 (operation enabled) | | |
| 5 | Wait for comparator stabilization time $t_{CMP}$ | | | | |
| 6 | COMPFIR | CiFCK1, CiFCK0 | Select whether the digital filter is used or not and the sampling clock. | | |
| | | CiEPO, CiEDG | Select the edge detection condition for an interrupt request (rising edge/falling edge/both edges). | | |
| 7 | COMPOCR | CiOP, CiOE | Set the VCOUTi output (select the polarity and set output enabled or disabled).<br>See **17.4.3 Comparator i output (i = 0 or 1)**. | | |
| | | CiIE | Set the interrupt request output enabled or disabled.<br>See **17.4.3 Comparator i output (i = 0 or 1)**. | | |
| 8 | PR2L | CMPPR0i, CMPPR1i | When using an interrupt: Select the interrupt priority level. | | |
| 9 | MK2L | CMPMKi | When using an interrupt: Select the interrupt masking. | | |
| 10 | IF2L | CMPIFi | When using an interrupt: 0 (no interrupt requested: initialization)Note 3 | | |

**Notes 1.** Comparator 0 and comparator 1 cannot be set independently.

**2.** Can be set in high-speed mode (SPDMD = 1).

**3.** After setting the comparator, an unnecessary interrupt may occur until operation becomes stable, so initialize the interrupt flag.

**4.** Can be set in HS (high-speed main) mode.

**Remark** i = 0, 1, n = 2, 3

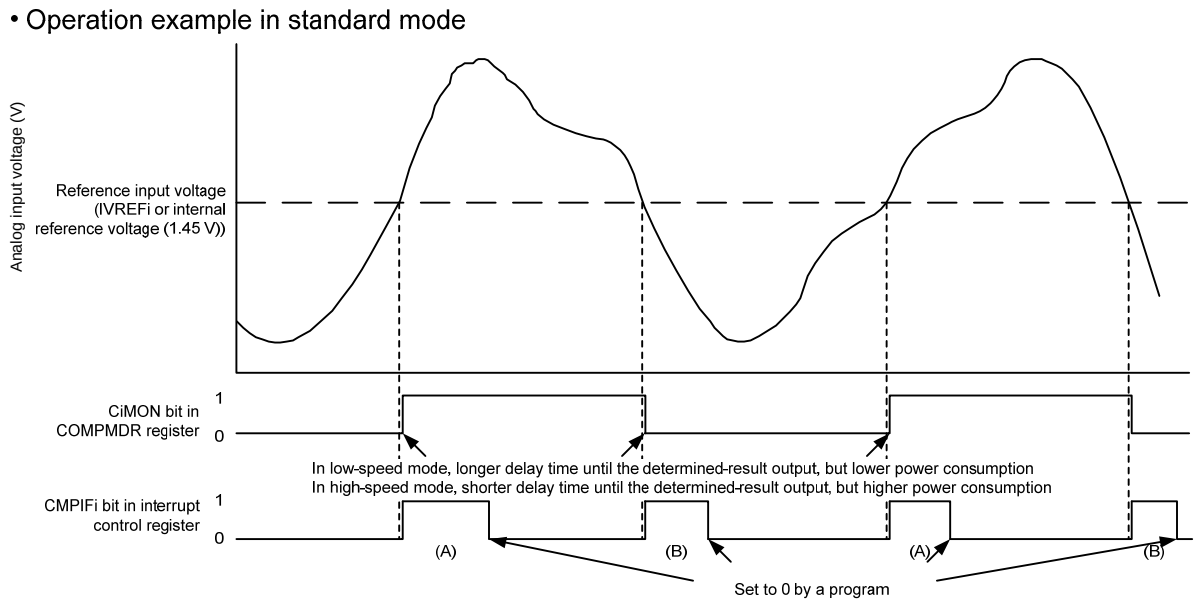
Figures 17-6 and 17-7 show comparator i (i = 0 or 1) operation examples. In standard mode, the CiMON bit in the COMPMDR register is set to 1 when the analog input voltage is higher than the reference input voltage, and the CiMON bit is set to 0 when the analog input voltage is lower than the reference input voltage.

In window mode, the CiMON bit in the COMPMDR register is set to 1 when the analog input voltage meets the following condition, and the CiMON bit is set to 0 when the analog input voltage does not meet the following condition:

“Low-voltage reference voltage < analog input voltage < high-voltage reference voltage”

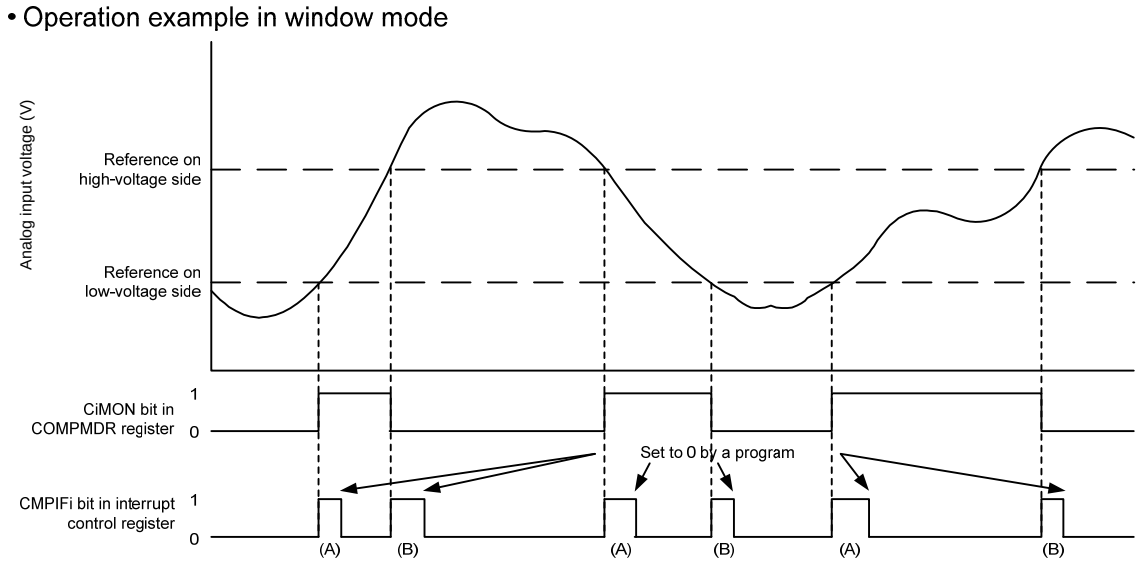
When using the comparator i interrupt, set CiIE in the COMPOCR register to 1 (interrupt request output enabled). If the comparison result changes at this time, a comparator i interrupt request is generated. For details on interrupt requests, see **17.4.2 Comparator i (i = 0 or 1) Interrupts**.

**Figure 17-6. Comparator i (i = 0 or 1) Operation Example in Standard Mode**




**Caution The above diagram applies when CiFCK1 to CiFCK0 in the COMPFIR register = 00B (no filter) and CiEDG = 1 (both edges). When CiEDG = 0 and CiEPO = 0 (rising edge), CMPIFi changes as shown by (A) only. When CiEDG = 0 and CiEPO = 1 (falling edge), CMPIFi changes as shown by (B) only.**

**Figure 17-7. Comparator i (i = 0 or 1) Operation Example in Window Mode**



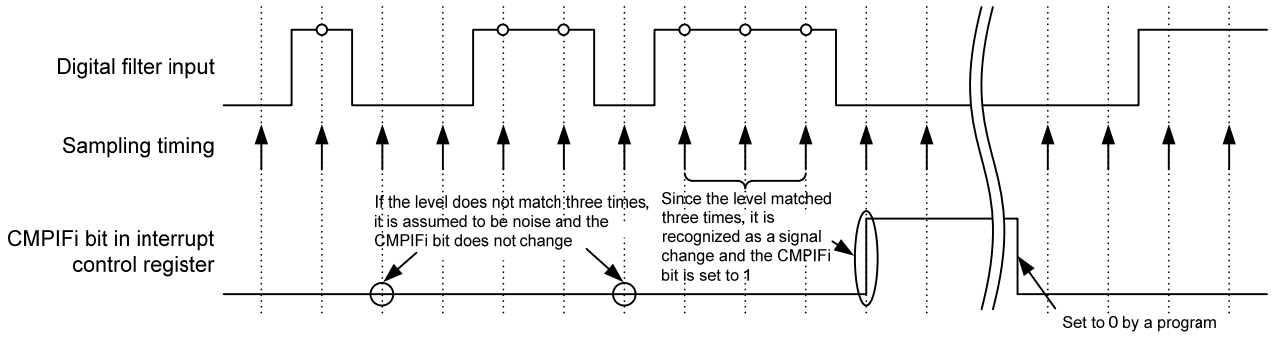

**Caution The above diagram applies when CiFCK1 to CiFCK0 in the COMPFIR register = 00B (no filter) and CiEDG = 1 (both edges). When CiEDG = 0 and CiEPO = 0 (rising edge), CMPIFi changes as shown by (A) only. When CiEDG = 0 and CiEPO = 1 (falling edge), CMPIFi changes as shown by (B) only.**

### 17.4.1 Comparator i digital filter (i = 0 or 1)

Comparator i contains a digital filter. The sampling clock can be selected by bits CiFCK1 and CiFCK0 in the COMPFIR register. The comparator i output signal is sampled every sampling clock, and when the level matches three times, that value is determined as the digital filter output at the next sampling clock.

Figure 17-8 shows the comparator i (i = 0 or 1) digital filter and interrupt operation example.

**Figure 17-8. Comparator i (i = 0 or 1) Digital Filter and Interrupt Operation Example**




**Caution The above operation example applies when bits CiFCK1 and CiFCK0 in the COMPFIR register is 01B, 10B, or 11B (digital filter enabled).**

### 17.4.2 Comparator i (i = 0 or 1) interrupts

The comparator generates interrupt requests from two sources, comparator 0 and comparator 1. The comparator i interrupt each uses a priority level specification flag, an interrupt mask flag, an interrupt request flag, and a single vector.

When using the comparator i interrupt, set the CiIE bit in the COMPOCR register to 1 (interrupt request output enabled). The condition for interrupt request generation can be set by the COMPFIR register. The comparator outputs can also be passed through the digital filter. Three different sampling clocks can be selected for the digital filter.

For details on the register setting and interrupt request generation, see **17.3.3 Comparator filter control register (COMPFIR)** and **17.3.4 Comparator output control register (COMPOCR)**.

### 17.4.3 Comparator i Output (i = 0 or 1)

The comparison result from the comparator can be output to external pins. Bits CiOP and CiOE in the COMPOCR register can be used to set the output polarity (non-inverted output or inverted output) and output enabled or disabled. For the correspondence between the register setting and the comparator output, see **17.3.4 Comparator output control register (COMPOCR)**.

To output the comparator comparison result to the VCOUTi output pin, use the following procedure to set the ports. Note that the ports are set to input after reset.

<1> Set the mode for the comparator (Steps 1 to 4 as listed in Table 17-2 Procedure for Setting Comparator Associated Registers).
<2> Set the VCOUTi output for the comparator (set the COMPOCR register to select the polarity and enable the output).
<3> Set the corresponding port register bit for the VCOUTi output pin to 0.
<4> Set the corresponding port direction register for the VCOUTi output pin to output (start outputting from the pin).

### 17.4.4 Stopping or supplying comparator clock

To stop the comparator clock by setting peripheral enable register 1 (PER1), use the following procedure:

<1> Set the CiENB bit in the COMPMDR register to 0 (stop the comparator).
<2> Set the CMPIFi bit in registers IF2L to 0 (clear any unnecessary interrupt before stopping the comparator).
<3> Set the CMPEN bit in the PER1 register to 0.

When the clock is stopped by setting PER1, all the internal registers in the comparator are initialized. To use the comparator again, follow the procedure in Table 17-2 to set the registers.

**Caution When DTC activation is enabled under either of the following conditions, a DTC transfer is started and an interrupt is generated after completion of the transfer. Therefore, enable DTC activation after confirming the comparator monitor flag (CnMON) as necessary. (n = 0, 1)**

- **The comparator is set to an interrupt request on one-edge detection (CnEDG = 0), an interrupt request at the rising edge for the comparator, and IVCMP > IVREF (or internal reference voltage: 1.45 V)**
- **The comparator is set to an interrupt request on one-edge detection (CnEDG = 0), an interrupt request at the falling edge for the comparator, and IVCMP < IVREF (or internal reference voltage: 1.45 V)**

# CHAPTER 18 SERIAL ARRAY UNIT

Serial array unit has up to four serial channels. Each channel can achieve 3-wire serial (CSI), UART, and simplified $I^2C$ communication.

Function assignment of each channel supported by the RL78/I1B is as shown below.

| Unit | Channel | Used as CSI | Used as UART | Used as Simplified $I^2C$ |
|---|---|---|---|---|
| 0 | 0 | CSI00 | UART0 (supporting LIN-bus) | IIC00 |
| | 1 | – | | – |
| | 2 | – | UART1 | IIC10 |
| | 3 | – | | – |
| 1 | 0 | – | UART2 (supporting IrDA) | – |
| | 1 | – | | – |

When “UART0” is used for channels 0 and 1 of the unit 0, CSI00 and IIC00 cannot be used, but UART1 or IIC10 can be used.

## 18.1 Functions of Serial Array Unit

Each serial interface supported by the RL78/I1B has the following features.

### 18.1.1 3-wire serial I/O (CSI00)

Data is transmitted or received in synchronization with the serial clock (SCK) output from the master channel.
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 **18.5 Operation of 3-Wire Serial I/O (CSI00) 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. $f_{MCK}/2$
  - During slave communication: Max. $f_{MCK}/6$

[Interrupt function]

- Transfer end interrupt/buffer empty interrupt

[Error detection flag]

- Overrun error

In addition, CSI00 supports the SNOOZE mode. When SCK input is detected while in the STOP mode, the SNOOZE mode makes data reception that does not require the CPU possible.

**Note** Use the clocks within a range satisfying the SCK cycle time ($t_{KCY}$) characteristics. For details, see **CHAPTER 37 ELECTRICAL SPECIFICATIONS**.

### 18.1.2 UART (UART0 to UART2)

This is a start-stop synchronization 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 **18.6 Operation of UART (UART0 to UART2) Communication.**

[Data transmission/reception]

- Data length of 7, 8, or 9 bits[Note]
- Select the MSB/LSB first
- 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, UART0 reception supports the SNOOZE mode. When RxD input is detected while in the STOP mode, the SNOOZE mode makes data reception that does not require the CPU possible.

The LIN-bus is accepted in UART0 (0 and 1 channels of unit 0).

[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 can be specified for the 9-bit data length.

### 18.1.3 Simplified I²C (IIC00, IIC10)

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 **18.8 Operation of Simplified I²C (IIC00, IIC10) 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
- Wait detection functions

**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 the processing flow in **18.8.3 (2)** for details.

**Remarks 1.** To use an I²C bus of full function, see **CHAPTER 19 SERIAL INTERFACE IICA**.

**2.** m: Unit number (m = 0), n: Channel number (n = 0, 2)

### 18.1.4 IrDA

By combining UART2 of the serial array unit and the IrDA module, IrDA communication waveforms can be transmitted or received based on IrDA (Infrared Data Association) standard 1.0. For details, see **CHAPTER 20 IrDA**.

[Data transmission/reception]

- Transfer rate: 115.2 kbps/57.6 kbps/38.4 kbps/19.2 kbps/9600 bps/2400 bps

## 18.2 Configuration of Serial Array Unit

The serial array unit includes the following hardware.

**Table 18-1. Configuration of Serial Array Unit**

| Item | Configuration |
|---|---|
| Shift register | 8 bits or 9 bits[Note 1] |
| Buffer register | Lower 8 bits or 9 bits of serial data register mn (SDRmn)[Notes 1, 2] |
| Serial clock I/O | SCK00 pin (for 3-wire serial I/O), SCL00, SCL10 pins (for simplified $I^2C$) |
| Serial data input | SI00 pin (for 3-wire serial I/O), RxD1 to RxD2 pins (for UART), RxD0 pin (for UART supporting LIN-bus) |
| Serial data output | SO00 pin (for 3-wire serial I/O), TxD1 to TxD2 pins (for UART), TxD0 pin (for UART supporting LIN-bus) |
| Serial data I/O | SDA00, SDA10 pins (for simplified $I^2C$) |
| Control registers | <Registers of unit setting block><br>• Peripheral enable register 0 (PER0)<br>• Serial clock select register m (SPSm)<br>• Serial channel enable status register m (SEm)<br>• Serial channel start register m (SSm)<br>• Serial channel stop register m (STm)<br>• Serial output enable register m (SOEm)<br>• Serial output register m (SOm)<br>• Serial output level register m (SOLm)<br>• Serial standby control register 0 (SSC0)<br>• Input switch control register (ISC)<br>• Noise filter enable register 0 (NFEN0) |
| | <Registers of each channel><br>• Serial data register mn (SDRmn)<br>• Serial mode register mn (SMRmn)<br>• Serial communication operation setting register mn (SCRmn)<br>• Serial status register mn (SSRmn)<br>• Serial flag clear trigger register mn (SIRmn) |
| | • Port input mode registers 0, 1, 8 (PIM0, PIM1, PIM8)<br>• Port output mode registers 0, 1, 8 (POM0, POM1, POM8)<br>• Port mode registers 0, 1, 8 (PM0, PM1, PM8)<br>• Port registers 0, 1, 8 (P0, P1, P8) |

(**Notes** and **Remark** are listed on the next page.)

**Notes 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

**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), q: UART number (q = 0 to 2), r: IIC number (r = 00, 10), mn = 00 to 03, 10, 11

Figure 18-1 shows the block diagram of the serial array unit 0.

<R>

**Figure 18-1. Block Diagram of Serial Array Unit 0**

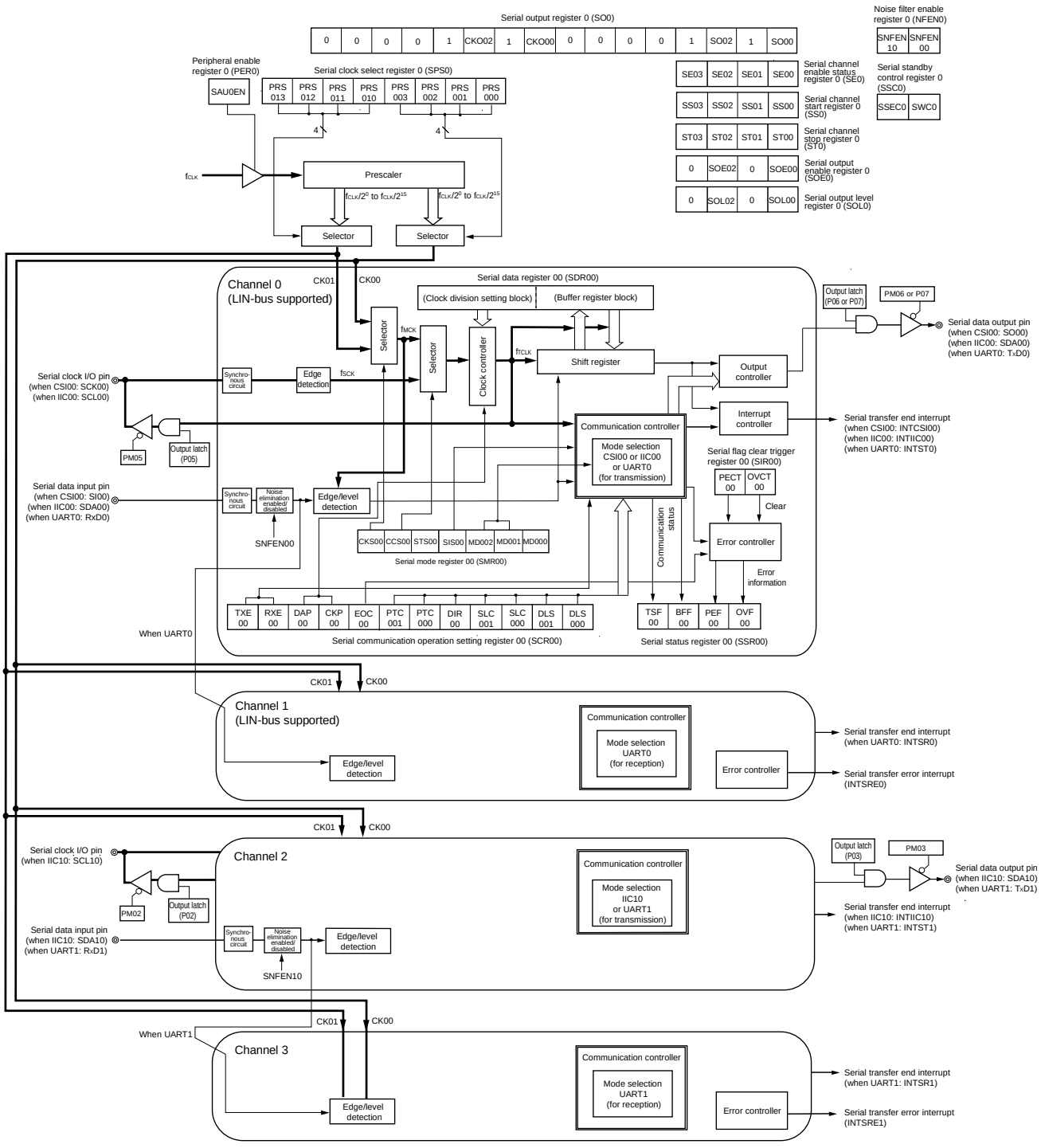
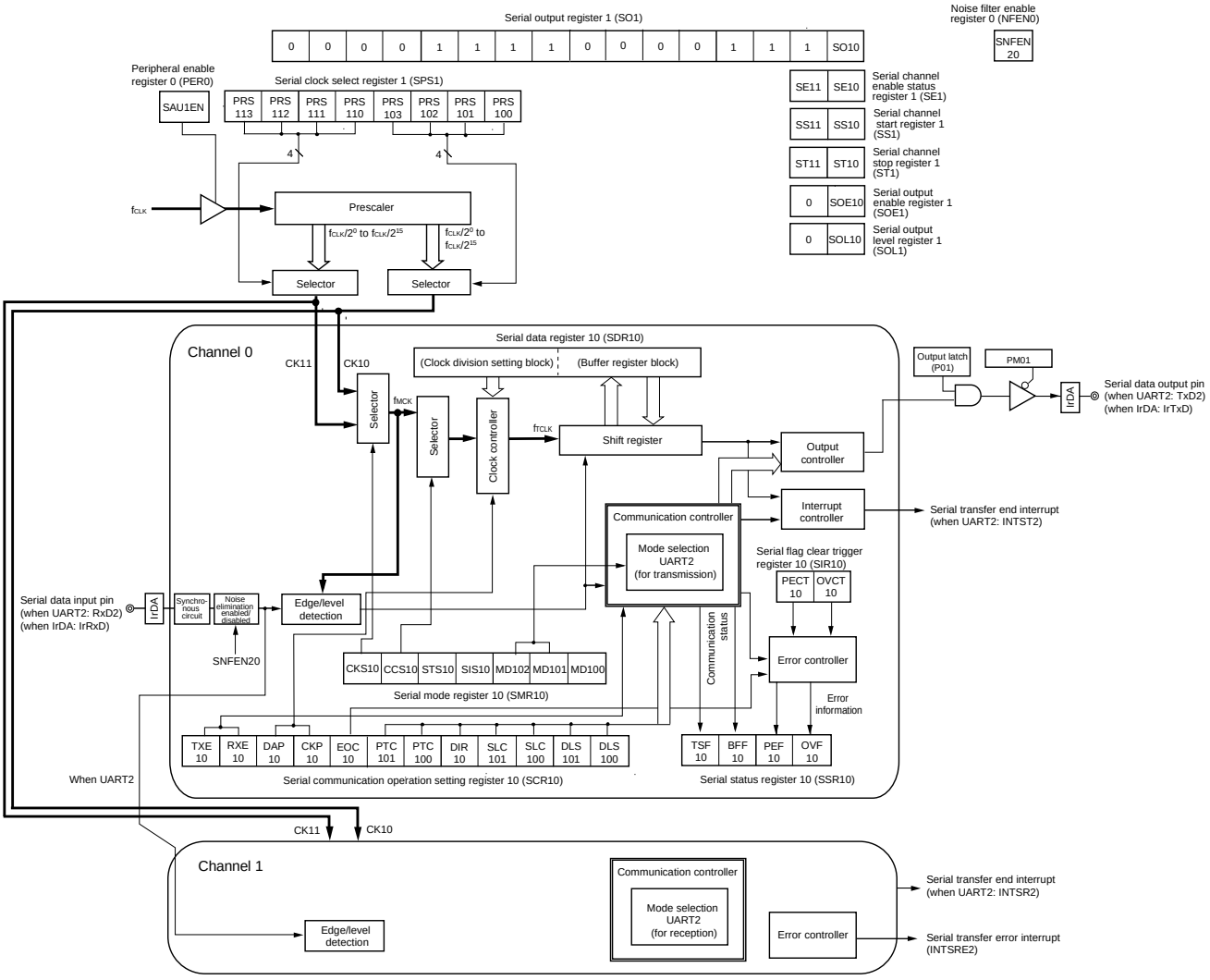


Figure 18-2 shows the block diagram of the serial array unit 1.

<R>

**Figure 18-2. Block Diagram of Serial Array Unit 1**



### 18.2.1 Shift register

This is a 9-bit register that converts parallel data into serial data or vice versa.

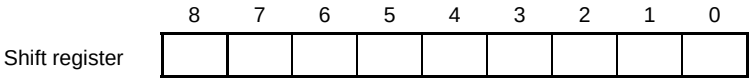
In case of the UART communication of nine bits of data, nine bits (bits 0 to 8) are used[Note 1].

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 8/9 bits of serial data register mn (SDRmn).




### 18.2.2 Lower 8/9 bits of the serial data register mn (SDRmn)

The SDRmn register 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 8/9 bits. When data is to be transmitted, set transmit to be transferred to the shift register to the lower 8/9 bits.

The data stored in the lower 8/9 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 SDRmn register)
- 8-bit data length (stored in bits 0 to 7 of SDRmn register)
- 9-bit data length (stored in bits 0 to 8 of SDRmn register)[Note 1]

The SDRmn register can be read or written in 16-bit units.

The lower 8/9 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)

Reset signal generation clears the SDRmn register to 0000H.

**Notes 1.** Only UART0 can be specified for the 9-bit data length.

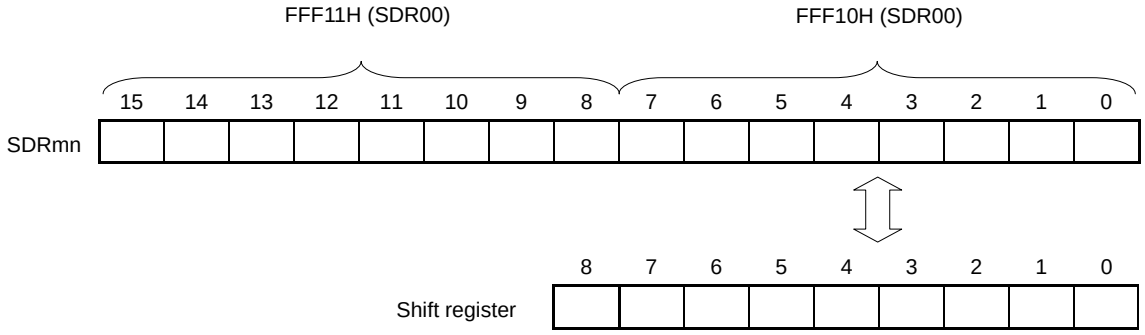
**2.** Rewriting SDRmn[7:0] by 8-bit memory manipulation instruction is prohibited when the operation is stopped (SEmn = 0) (all of SDRmn[15:9] are cleared (0)).

**Remarks 1.** After data is received, “0” is stored in bits 0 to 8 in bit portions that exceed the data length.

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00), q: UART number (q = 0 to 2), r: IIC number (r = 00, 10), mn = 00 to 03, 10, 11

**Figure 18-3. Format of Serial Data Register mn (SDRmn) (mn = 00, 01, 10, 11)**

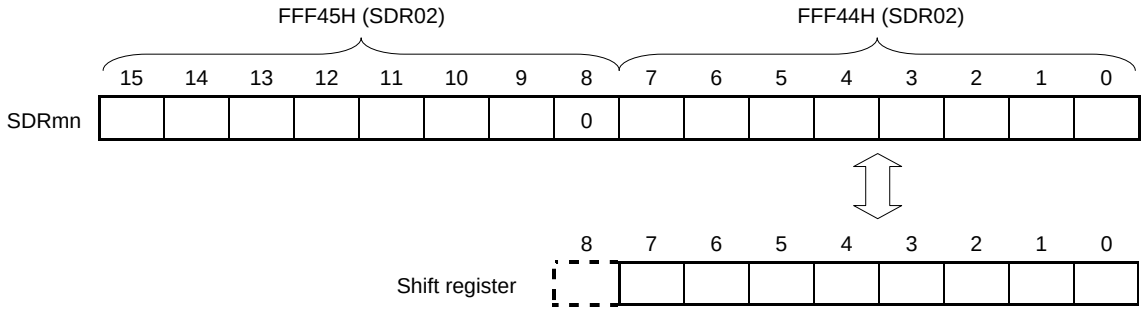
Address: FFF10H, FFF11H (SDR00), FFF12H, FFF13H (SDR01), FFF48H, FFF49H (SDR10), FFF4AH, FFF4BH (SDR11) After reset: 0000H R/W




**Remark** For the function of the higher 7 bits of the SDRmn register, see **18.3 Registers Controlling Serial Array Unit**.

**Figure 18-4. Format of Serial Data Register mn (SDRmn) (mn = 02, 03)**

Address: FFF44H, FFF45H (SDR02), FFF46H, FFF47H (SDR03) After reset: 0000H R/W




**Caution Be sure to clear bit 8 to "0".**

**Remark** For the function of the higher 7 bits of the SDRmn register, see **18.3 Registers Controlling Serial Array Unit**.

## 18.3 Registers Controlling Serial Array Unit

Serial array unit is controlled by the following registers.

- Peripheral enable register 0 (PER0)
- 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 0 (SSC0)
- Input switch control register (ISC)
- Noise filter enable register 0 (NFEN0)
- Port input mode registers 0, 1, 8 (PIM0, PIM1, PIM8)
- Port output mode registers 0, 1, 8 (POM0, POM1, POM8)
- Port mode registers 0, 1, 8 (PM0, PM1, PM8)
- Port registers 0, 1, 8 (P0, P1, P8)

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

### 18.3.1 Peripheral enable register 0 (PER0)

PER0 is used to enable or disable supplying the clock to the peripheral hardware. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

When serial array unit 0 is used, be sure to set bit 2 (SAU0EN) of this register to 1.

When serial array unit 1 is used, be sure to set bit 3 (SAU1EN) of this register to 1.

The PER0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears the PER0 register to 00H.

**Figure 18-5. Format of Peripheral Enable Register 0 (PER0)**

Address: F00F0H After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PER0 | RTCWEN | IRDAEN | ADCEN | IICA0EN | SAU1EN | SAU0EN | 0 | TAU0EN |

| SAUmEN | Control of serial array unit m input clock supply |
|---|---|
| 0 | Stops supply of input clock.<br>• SFR used by serial array unit m cannot be written.<br>• Serial array unit m is in the reset status. |
| 1 | Enables input clock supply.<br>• SFR used by serial array unit m can be read/written. |

**Cautions 1. When setting serial array unit m, be sure to first set the following registers with the SAUmEN bit set to 1. If SAUmEN = 0, writing to a control register of serial array unit m is ignored, and, even if the register is read, only the default value is read (except for the input switch control register (ISC), noise filter enable register 0 (NFEN0), port input mode registers 0, 1, 8 (PIM0, PIM1, PIM8), port output mode registers 0, 1, 8 (POM0, POM1, POM8), port mode registers 0, 1, 8 (PM0, PM1, PM8), and port registers 0, 1, 8 (P0, P1, P8)).**

- **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)**

**2. Be sure to clear bit 1 to “0”.**

### 18.3.2 Serial clock select register m (SPSm)

The SPSm register 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 lower 8 bits of the SPSm register can be set with an 8-bit memory manipulation instruction with SPSmL.

Reset signal generation clears the SPSm register to 0000H.

**Figure 18-6. Format of Serial Clock Select Register m (SPSm)**

Address: F0126H, F0127H (SPS0), F0166H, F0167H (SPS1) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SPSm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PRS m13 | PRS m12 | PRS m11 | PRS m10 | PRS m03 | PRS m02 | PRS m01 | PRS m00 |

| PRS mk3 | PRS mk2 | PRS mk1 | PRS mk0 | Section of operation clock (CKmk)[Note] | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | $f_{CLK}$ = 4 MHz | $f_{CLK}$ = 8 MHz | $f_{CLK}$ = 12 MHz | $f_{CLK}$ = 20 MHz | $f_{CLK}$ = 24 MHz |
| 0 | 0 | 0 | 0 | $f_{CLK}$ | 4 MHz | 8 MHz | 12 MHz | 20 MHz | 24 MHz |
| 0 | 0 | 0 | 1 | $f_{CLK}/2$ | 2 MHz | 4 MHz | 6 MHz | 10 MHz | 12 MHz |
| 0 | 0 | 1 | 0 | $f_{CLK}/2^2$ | 1 MHz | 2 MHz | 3 MHz | 5 MHz | 6 MHz |
| 0 | 0 | 1 | 1 | $f_{CLK}/2^3$ | 500 kHz | 1 MHz | 1.5 MHz | 2.5 MHz | 3 MHz |
| 0 | 1 | 0 | 0 | $f_{CLK}/2^4$ | 250 kHz | 500 kHz | 750 kHz | 1.25 MHz | 1.5 MHz |
| 0 | 1 | 0 | 1 | $f_{CLK}/2^5$ | 125 kHz | 250 kHz | 375 kHz | 625 kHz | 750 KHz |
| 0 | 1 | 1 | 0 | $f_{CLK}/2^6$ | 62.5 kHz | 125 kHz | 187.5 kHz | 313 kHz | 375 kHz |
| 0 | 1 | 1 | 1 | $f_{CLK}/2^7$ | 31.25 kHz | 62.5 kHz | 93.8 kHz | 156 kHz | 187.5 kHz |
| 1 | 0 | 0 | 0 | $f_{CLK}/2^8$ | 15.62 kHz | 31.25 kHz | 46.9 kHz | 78.1 kHz | 93.8 kHz |
| 1 | 0 | 0 | 1 | $f_{CLK}/2^9$ | 7.81 kHz | 15.62 kHz | 23.4 kHz | 39.1 kHz | 46.9 kHz |
| 1 | 0 | 1 | 0 | $f_{CLK}/2^{10}$ | 3.91 kHz | 7.81 kHz | 11.7 kHz | 19.5 kHz | 23.4 kHz |
| 1 | 0 | 1 | 1 | $f_{CLK}/2^{11}$ | 1.95 kHz | 3.91 kHz | 5.86 kHz | 9.77 kHz | 11.7 kHz |
| 1 | 1 | 0 | 0 | $f_{CLK}/2^{12}$ | 976 Hz | 1.95 kHz | 2.93 kHz | 4.88 kHz | 5.86 kHz |
| 1 | 1 | 0 | 1 | $f_{CLK}/2^{13}$ | 488 Hz | 976 Hz | 1.46 kHz | 2.44 kHz | 2.93 kHz |
| 1 | 1 | 1 | 0 | $f_{CLK}/2^{14}$ | 244 Hz | 488 Hz | 732 Hz | 1.22 kHz | 1.46 kHz |
| 1 | 1 | 1 | 1 | $f_{CLK}/2^{15}$ | 122 Hz | 244 Hz | 366 Hz | 610 Hz | 732 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 clear bits 15 to 8 to "0".**

**Remarks 1.** $f_{CLK}$: CPU/peripheral hardware clock frequency

**2.** m: Unit number (m = 0, 1)

**3.** k = 0, 1

### 18.3.3 Serial mode register mn (SMRmn)

The SMRmn register is a register that sets 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, an operation mode (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.

Reset signal generation sets the SMRmn register to 0020H.

**Figure 18-7. 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

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SMRmn | CKSmn | CCSmn | 0 | 0 | 0 | 0 | 0 | STSmn[Note] | 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 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 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. | |

**Note** The SMR01, SMR03, and SMR11 registers only.

**Caution Be sure to clear 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), q: UART number (q = 0 to 2), r: IIC number (r = 00, 10), mn = 00 to 03, 10, 11

**Figure 18-7. Format of Serial Mode Register mn (SMRmn) (2/2)**

Address: F0110H, F0111H (SMR00) to F0116H, F0117H (SMR03), After reset: 0020H R/W
F0150H, F0151H (SMR10), F0152H, F0153H (SMR11)

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SMRmn | CKS mn | CCS mn | 0 | 0 | 0 | 0 | 0 | STS mn[Note] | 0 | SIS mn0[Note] | 1 | 0 | 0 | MD mn2 | MD mn1 | MD mn0 |

| SIS mn0 | Controls inversion of level of receive data of channel n in UART mode |
|---|---|
| 0 | Falling edge is detected as the start bit.<br>The input communication data is captured as is. |
| 1 | Rising edge is detected as the start bit.<br>The input communication data is inverted and captured. |

| MD mn2 | MD mn1 | Setting of operation mode of channel n |
|---|---|---|
| 0 | 0 | CSI mode |
| 0 | 1 | UART mode |
| 1 | 0 | Simplified $I^2C$ mode |
| 1 | 1 | Setting prohibited |

| MD mn0 | Selection of interrupt source of channel n |
|---|---|
| 0 | Transfer end interrupt |
| 1 | Buffer empty interrupt<br>(Occurs when data is transferred from the SDRmn register to the shift register.) |
| For successive transmission, the next transmit data is written by setting the MDmn0 bit to 1 when SDRmn data has run out. | |

**Note** The SMR01, SMR03, and SMR11 registers only.

**Caution Be sure to clear 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), q: UART number (q = 0 to 2), r: IIC number (r = 00, 10), mn = 00 to 03, 10, 11

### 18.3.4 Serial communication operation setting register mn (SCRmn)

The SCRmn register 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.

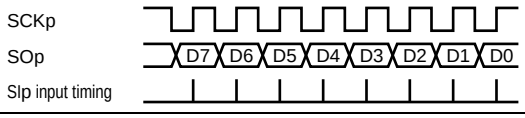
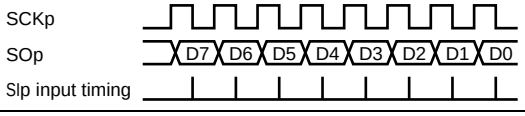
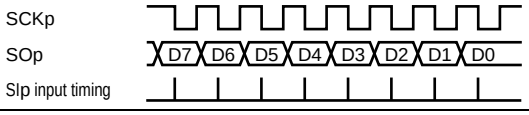
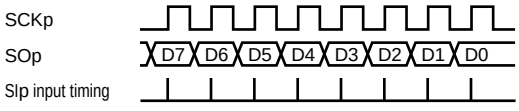
Reset signal generation sets the SCRmn register to 0087H.

**Figure 18-8. Format of Serial Communication Operation Setting Register mn (SCRmn) (1/2)**

Address: F0118H, F0119H (SCR00) to F011EH, F011FH (SCR03), After reset: 0087H R/W
F0158H, F0159H (SCR10), F015AH, F015BH (SCR11)

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SCRmn | TXE mn | RXE mn | DAP mn | CKP mn | 0 | EOC mn | PTC mn1 | PTC mn0 | DIR mn | 0 | SLCm n1 Note 1 | SLC mn0 | 0 | 1 | DLSm n1 Note 2 | DLS mn0 |

| TXE mn | RXE mn | Setting of operation mode of channel n |
|---|---|---|
| 0 | 0 | Disable communication. |
| 0 | 1 | Reception only |
| 1 | 0 | Transmission only |
| 1 | 1 | Transmission/reception |

| DAP mn | CKP mn | Selection of data and clock phase in CSI mode | Type |
|---|---|---|---|
| 0 | 0 |  | 1 |
| 0 | 1 |  | 2 |
| 1 | 0 |  | 3 |
| 1 | 1 |  | 4 |
| Be sure to set DAPmn, CKPmn = 0, 0 in the UART mode and simplified $I^2C$ mode. | | | |

| EOC mn | Mask control of error interrupt signal (INTSREx (x = 0 to 2)) |
|---|---|
| 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 = 0 in the CSI mode, simplified $I^2C$ mode, and during UART transmission Note 3. | |

**Notes 1.** The SCR00, SCR02, and SCR10 registers only.

**2.** The SCR00 and SCR01 registers only. Others are fixed to 1.

**3.** When using CSImn not with EOCmn = 0, error interrupt INTSREn may be generated.

**Caution Be sure to clear bits 3, 6, and 11 to "0" (Also clear 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), mn = 00 to 03, 10, 11

**Figure 18-8. Format of Serial Communication Operation Setting Register mn (SCRmn) (2/2)**

Address: F0118H, F0119H (SCR00) to F011EH, F011FH (SCR03), After reset: 0087H R/W
F0158H, F0159H (SCR10), F015AH, F015BH (SCR11)

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SCRmn | TXE mn | RXE mn | DAP mn | CKP mn | 0 | EOC mn | PTC mn1 | PTC mn0 | DIR mn | 0 | SLCm n1 Note 1 | SLC mn0 | 0 | 1 | DLSm n1 Note 2 | DLS mn0 |

| PTC mn1 | PTC mn0 | 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 3. | No parity judgment |
| 1 | 0 | Outputs even parity. | Judged as even parity. |
| 1 | 1 | Outputs odd parity. | Judges as odd parity. |
| Be sure to set PTCmn1, PTCmn0 = 0, 0 in the CSI mode and simplified $I^2C$ mode. | | | |

| DIR mn | Selection of data transfer sequence in CSI and UART modes |
|---|---|
| 0 | Inputs/outputs data with MSB first. |
| 1 | Inputs/outputs data with LSB first. |
| Be sure to clear DIRmn = 0 in the simplified $I^2C$ mode. | |

| SLCm n1 Note 1 | SLC mn0 | 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.<br>Set 1 bit (SLCmn1, SLCmn0 = 0, 1) during UART reception and in the simplified $I^2C$ mode.<br>Set no stop bit (SLCmn1, SLCmn0 = 0, 0) in the CSI mode.<br>Set 1 bit (SLCmn1, SLCmn0 = 0, 1) or 2 bits (SLCmn1, SLCmn0 = 1, 0) during UART transmission. | | |

| DLSm n1 Note 2 | DLS mn0 | Setting of data length in 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 |
| Be sure to set DLSmn1, DLSmn0 = 1, 1 in the simplified $I^2C$ mode. | | |

**Notes 1.** The SCR00, SCR02, and SCR10 registers only.

**2.** The SCR00 and SCR01 registers only. Others are fixed to 1.

**3.** 0 is always added regardless of the data contents.

**Caution Be sure to clear bits 3, 6, and 11 to "0" (Also clear 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), mn = 00 to 03, 10, 11

### 18.3.5 Serial data register mn (SDRmn)

<R> The SDRmn register is the transmit/receive data register (16 bits) of channel n. Bits 8 to 0 (lower 9 bits) of SDR00, SDR01, SDR10 and SDR11 or bits 7 to 0 (lower 8 bits) of SDR02 and SDR03 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 operating 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 SDR00 to 0000000B. The input clock $f_{SCK}$ (slave transfer in CSI mode) from the SCKp pin is used as the transfer clock.

The lower 8/9 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 8/9 bits, and during transmission, the data to be transmitted to the shift register is set to the lower 8/9 bits.

The SDRmn register can be read or written in 16-bit units.

However, the higher 7 bits can be written or read only when the operation is stopped (SEmn = 0). During operation (SEmn = 1), a value is written only to the lower 8/9 bits of the SDRmn register. When the SDRmn register is read during operation, the higher 7 bits are always read as 0.

Reset signal generation clears the SDRmn register to 0000H.

**Figure 18-9. Format of Serial Data Register mn (SDRmn)**

Address: FFF10H, FFF11H (SDR00), FFF12H, FFF13H (SDR01) After reset: 0000H R/W

| | FFF11H (SDR00) | | | | | | | | FFF10H (SDR00) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SDRmn | | | | | | | | 0 | | | | | | | | |

Address: FFF44H, FFF45H (SDR02), FFF46H, FFF47H (SDR03) After reset: 0000H R/W
FFF48H, FFF49H (SDR10), FFF4AH, FFF4BH (SDR11)

| | FFF45H (SDR02) | | | | | | | | FFF44H (SDR02) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SDRmn | | | | | | | | 0 | | | | | | | | |

| SDRmn[15:9] | | | | | | | Transfer clock setting by dividing the operating clock ($f_{MCK}$) |
|---|---|---|---|---|---|---|---|
| 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 |

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

**Cautions 1. Be sure to clear bit 8 of the SDR02, SDR03, SDR10 and SDR11 registers to "0".**

**2. Setting SDRmn[15:9] = (0000000B, 0000001B) is prohibited when UART is used.**

**3. Setting SDRmn[15:9] = 0000000B is prohibited when simplified $I^2C$ is used. Set SDRmn[15:9] to 0000001B or greater.**

**4. Rewriting SDRmn[7:0] by 8-bit memory manipulation instruction is prohibited when the operation is stopped (SEmn = 0) (all of SDRmn[15:9] are cleared (0)).**

**Remarks 1.** For the function of the lower 8/9 bits of the SDRmn register, see **18.2 Configuration of Serial Array Unit**.

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

### 18.3.6 Serial flag clear trigger register mn (SIRmn)

The SIRmn register 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 register 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 lower 8 bits of the SIRmn register can be set with an 8-bit memory manipulation instruction with SIRmnL.

Reset signal generation clears the SIRmn register to 0000H.

**Figure 18-10. 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

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SIRmn | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | FECTmn[Note] | PECTmn | OVCTmn |

| FECTmn | Clear trigger of framing error 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** The SIR01, SIR03, and SIR11 registers only.

**Caution Be sure to clear bits 15 to 3 (or bits 15 to 2 for the SIR00, SIR02, or SIR10 register) to “0”.**

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

**2.** When the SIRmn register is read, 0000H is always read.

### 18.3.7 Serial status register mn (SSRmn)

The SSRmn register is a register that indicates the communication status and error occurrence status of 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 lower 8 bits of the SSRmn register can be set with an 8-bit memory manipulation instruction with SSRmnL.

Reset signal generation clears the SSRmn register to 0000H.

**Figure 18-11. 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

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SSRmn | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | TSFmn | BFFmn | 0 | 0 | FEFmn[Note] | PEFmn | OVFmn |

| TSFmn | Communication status indication flag of channel n |
|---|---|
| 0 | Communication is stopped or suspended. |
| 1 | Communication is in progress. |
| <Clear conditions><br>• 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).<br>• Communication ends.<br><Set condition><br>• Communication starts. | |

| BFFmn | Buffer register status indication flag of channel n |
|---|---|
| 0 | Valid data is not stored in the SDRmn register. |
| 1 | Valid data is stored in the SDRmn register. |
| <Clear conditions><br>• Transferring transmit data from the SDRmn register to the shift register ends during transmission.<br>• Reading receive data from the SDRmn register ends during reception.<br>• 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).<br><Set conditions><br>• Transmit data is written to the SDRmn register while the TXEmn bit of the SCRmn register is set to 1 (transmission or transmission and reception mode in each communication mode).<br>• Receive data is stored in the SDRmn register while the RXEmn bit of the SCRmn register is set to 1 (reception or transmission and reception mode in each communication mode).<br>• A reception error occurs. | |

**Note** The SSR01, SSR03, and SSR11 registers only.

**Caution When the CSI is performing reception operations in the SNOOZE mode (SWCm = 1), the BFFmn flag will not change.**

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

**Figure 18-11. Format of Serial Status Register mn (SSRmn) (2/2)**

Address: F0100H, F0101H (SSR00) to F0106H, F0107H (SSR03), F0140H, F0141H (SSR10), F0142H, F0143H (SSR11) After reset: 0000H R

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SSRmn | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | TSFmn | BFFmn | 0 | 0 | FEFmn Note | PEFmn | OVFmn |

| FEFmn | Framing error detection flag of channel n |
|---|---|
| 0 | No error occurs. |
| 1 | An error occurs (during UART reception). |
| <Clear condition><br>• 1 is written to the FECTmn bit of the SIRmn register.<br><Set condition><br>• 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 condition><br>• 1 is written to the PECTmn bit of the SIRmn register.<br><Set condition><br>• The parity of the transmit data and the parity bit do not match when UART reception ends (parity error).<br>• No ACK signal is returned from the slave channel 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 condition><br>• 1 is written to the OVCTmn bit of the SIRmn register.<br><Set condition><br>• 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 and reception mode in each communication mode).<br>• Transmit data is not ready for slave transmission or transmission and reception in CSI mode. | |

**Note** The SSR01, SSR03, and SSR11 registers only.

**Cautions 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 (OVEmn = 1) is detected.**

**2. When the CSI is performing reception operations in the SNOOZE mode (SWCm = 1), the OVFmn flag will not change.**

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

### 18.3.8 Serial channel start register m (SSm)

The SSm register is a trigger register that is used to enable starting communication/count by each channel.

When 1 is written a bit of this register (SSmn), 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 lower 8 bits of the SSm register can be set with an 1-bit or 8-bit memory manipulation instruction with SSmL.

Reset signal generation clears the SSm register to 0000H.

**Figure 18-12. Format of Serial Channel Start Register m (SSm)**

Address: F0122H, F0123H (SS0) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SS0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SS03 | SS02 | SS01 | SS00 |

Address: F0162H, F0163H (SS1) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SS1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SS11 | SS10 |

| SSmn | Operation start trigger of channel n |
|---|---|
| 0 | No trigger operation |
| 1 | Sets the SEmn bit to 1 and enters the communication wait status Note. |

**Note** If set the SSmn = 1 to during a communication operation, will wait status to stop the communication. At this time, holding status value of control register and shift register, SCKmn and SOmn pins, and FEFmn, PEFmn, OVFmn flags.

**Cautions 1. Be sure to clear bits 15 to 4 of the SS0 register, bits 15 to 2 of the SS1 register to "0".**

**2. For the UART reception, set the RXEmn bit of SCRmn register to 1, and then be sure to set SSmn to 1 after 4 or more $f_{MCK}$ clocks have elapsed.**

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

**2.** When the SSm register is read, 0000H is always read.

### 18.3.9 Serial channel stop register m (STm)

The STm register is a trigger register that is used to enable stopping communication/count by each channel.

When 1 is written a bit of this register (STmn), 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 set written by a 16-bit memory manipulation instruction.

The lower 8 bits of the STm register can be set with a 1-bit or 8-bit memory manipulation instruction with STmL.

Reset signal generation clears the STm register to 0000H.

**Figure 18-13. Format of Serial Channel Stop Register m (STm)**

Address: F0124H, F0125H (ST0) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ST0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ST03 | ST02 | ST01 | ST00 |

Address: F0164H, F0165H (ST1) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ST1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 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** Holding status value of the control register and shift register, the SCKmn and SOmn pins, and FEFmn, PEFmn, OVFmn flags.

**Caution Be sure to clear bits 15 to 4 of the ST0 register, bits 15 to 2 of the ST1 register to "0".**

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

**2.** When the STm register is read, 0000H is always read.

### 18.3.10 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 a bit of serial channel start register m (SSm), the corresponding bit of this register is set to 1. When 1 is written a bit of serial channel stop register m (STm), the corresponding bit is cleared to 0.

Channel n that is enabled to operate cannot rewrite by software the value of the CKOmn bit (serial clock output of channel n) of serial output register m (SOm) to be described below, and a value reflected by a communication operation is output from the serial clock pin.

Channel n that stops operation can set the value of the CKOmn bit of the SOm register by software and output its value 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 lower 8 bits of the SEm register can be set with a 1-bit or 8-bit memory manipulation instruction with SEmL.

Reset signal generation clears the SEm register to 0000H.

**Figure 18-14. Format of Serial Channel Enable Status Register m (SEm)**

Address: F0120H, F0121H (SE0) After reset: 0000H R

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SE0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SE03 | SE02 | SE01 | SE00 |

Address: F0160H, F0161H (SE1) After reset: 0000H R

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SE1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SE11 | SE10 |

| SEmn | Indication of operation enable/stop status of channel n |
|---|---|
| 0 | Operation stops |
| 1 | Operation is enabled. |

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

### 18.3.11 Serial output enable register m (SOEm)

The SOEm register is a register that is used to enable or stop output of the serial communication operation of each channel.

Channel n that enables serial output cannot rewrite by software the value of the SOmn bit of serial output register m (SOm) to be described below, 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 of the start condition and stop condition can be created by software.

The SOEm register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the SOEm register can be set with a 1-bit or 8-bit memory manipulation instruction with SOEmL.

Reset signal generation clears the SOEm register to 0000H.

**Figure 18-15. Format of Serial Output Enable Register m (SOEm)**

Address: F012AH, F012BH (SOE0) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SOE0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SOE02 | 0 | SOE00 |

Address: F016AH, F016BH (SOE1) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SOE1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 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 clear bits 15 to 3 and 1 of the SOE0 register, bits 15 to 3 and 1 of the SOE1 register to "0".**

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), mn = 00 to 03, 10, 11

### 18.3.12 Serial output register m (SOm)

The SOm register 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.

<R> Reset signal generation clears the SO0 register to 0F0FH, the SO1 register to 0303H.

**Figure 18-16. Format of Serial Output Register m (SOm)**

Address: F0128H, F0129H (SO0) After reset: 0F0FH R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SO0 | 0 | 0 | 0 | 0 | 1 | CKO02 | 1 | CKO00 | 0 | 0 | 0 | 0 | 1 | SO02 | 1 | SO00 |

Address: F0168H, F0169H (SO1) After reset: 0F0FH R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SO1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 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 clear bits 15 to 12 and 7 to 4 of the SO0 register to “0”. And be sure to set bits 11, 9, 3, and 1 to “1”.**
**Be sure to clear bits 15 to 12 and 7 to 4 of the SO1 register to “0”. And be sure to set bits 11 to 8, 3, and 1 to “1”.**

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), mn = 00 to 03, 10, 11

### 18.3.13 Serial output level register m (SOLm)

The SOLm register is a register that 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 corresponding bit in the CSI mode and simplifies $I^2C$ 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 lower 8 bits of the SOLm register can be set with an 8-bit memory manipulation instruction with SOLmL.

Reset signal generation clears the SOLm register to 0000H.

**Figure 18-17. Format of Serial Output Level Register m (SOLm)**

Address: F0134H, F0135H (SOL0) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SOL0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SOL02 | 0 | SOL00 |

Address: F0174H, F0175H (SOL1) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SOL1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 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 clear bits 15 to 3, and 1 of the SOL0 register, bits 15 to 1 of the SOL1 register to "0".**

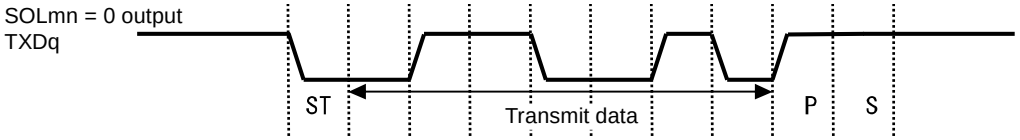
**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), mn = 00 to 03, 10, 11

Figure 18-18 shows examples in which the level of transmit data is reversed during UART transmission.

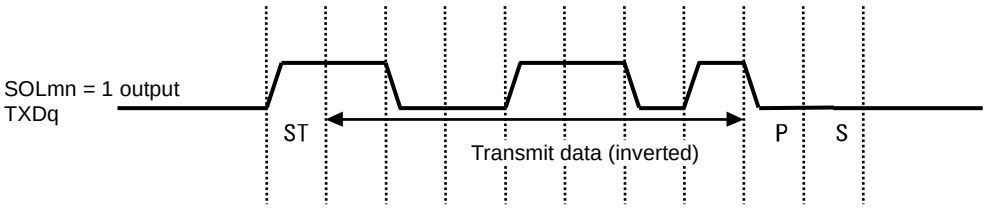
<R>

**Figure 18-18. Examples of Reverse Transmit Data**

**(1) Non-reverse Output (SOLmn = 0)**




**(2) Reverse Output (SOLmn = 1)**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), mn = 00 to 03, 10, 11

### 18.3.14 Serial standby control register 0 (SSC0)

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 SSC0 register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the SSC0 register can be set with an 8-bit memory manipulation instruction with SSC0L.

Reset signal generation clears the SSC0 register to 0000H.

**Caution The maximum transfer rate in the SNOOZE mode is as follows.**

- **When using CSI00 : Up to 1 Mbps**
- **When using UART0 : 4800 bps only**

**Figure 18-19. Format of Serial Standby Control Register 0 (SSC0)**

Address: F0138H (SSC0) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SSC0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SSEC0 | SWC0 |

| SSEC0 | 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) |
| 1 | Stop the generation of error interrupts (INTSRE0) |
| • The SSECm bit can be set to 1 or 0 only when both the SWC0 and EOCmn bits are set to 1 during UART reception in the SNOOZE mode. In other cases, clear the SSEC0 bit to 0.<br>• Setting SSEC0, SWC0 = 1, 0 is prohibited. | |

| SWC0 | 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 A/D conversion is performed without operating the CPU (the SNOOZE mode).<br>• 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.<br>• Even when using SNOOZE mode, be sure to set the SWC0 bit to 0 in normal operation mode and change it to 1 just before shifting to STOP mode.<br>Also, be sure to change the SWC0 bit to 0 after returning from STOP mode to normal operation mode. | |

**Figure 18-20. 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. |

### 18.3.15 Input switch control register (ISC)

The ISC1 and ISC0 bits of the ISC register are used to realize a LIN-bus communication operation by UART0 in coordination with an external interrupt and the timer array unit.

When bit 0 is set to 1, the input signal of the serial data input (RxD0) 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 (RxD0) pin is selected as a timer input, so that wake up signal can be detected, the low width of the break field, and the pulse width of the sync field can be measured by the timer.

The ISC register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears the ISC register to 00H.

**Figure 18-21. Format of Input Switch Control Register (ISC)**

Address: F0073H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ISC | 0 | 0 | 0 | 0 | 0 | 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 RxD0 pin is used as timer input (detects the wakeup signal and measures the low 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 RxD0 pin as an external interrupt (wakeup signal detection). |

**Caution Be sure to clear bits 7 to 2 to “0”.**

### 18.3.16 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 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.

Reset signal generation clears the NFEN0 register to 00H.

**Figure 18-22. Format of Noise Filter Enable Register 0 (NFEN0)**

Address: F0070H After reset: 00H 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.<br>Clear 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.<br>Clear the SNFEN10 bit 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.<br>Clear the SNFEN00 bit to 0 to use the other than RxD0 pin. | |

**Caution Be sure to clear bits 7 to 5, 3, and 1 to "0".**

### 18.3.17 Registers controlling port functions of serial input/output pins

When using the serial array unit set the registers that control the port functions multiplexed on the target channel (port mode register (PMxx), port register (Pxx), port input mode register (PIMxx), port output mode register (POMxx)).

For details, see **4.3.1 Port mode registers (PMxx)**, **4.3.2 Port registers (Pxx)**, **4.3.4 Port input mode registers (PIMxx)**, and **4.3.5 Port output mode registers (POMxx)**.

When using a port pin with a multiplexed serial data or serial clock output function (e.g. P07/SO00/TxD0/TI02/TO02/INTP2/TOOLTxD, P15/SEG9/(SCK00)/(SCL00)) for serial data or serial clock output, requires setting the corresponding bits in the port mode register (PMxx) to 0, and the corresponding bit in the port register (Pxx) to 1.

When using the port pin in N-ch open-drain output ($V_{DD}$ tolerance) mode, set the corresponding bit in the port output mode register (POMxx) to 1. When connecting an external device operating on a different potential (1.8 V, 2.5 V or 3 V), see **4.4.4 Connecting to external device with different potential (1.8 V, 2.5 V, 3 V)**.

Example: When using P07/SO00/TxD0/TI02/TO02/INTP2/TOOLTxD for serial data output
Set the PM07 bit of the port mode register 0 to 0.
Set the P07 bit of the port register 0 to 1.

When using a port pin with a multiplexed serial data or serial clock input function (e.g. P05/SCK00/SCL00/TI04/TO04/INTP3, P06/SI00/RxD0/TI03/TO03/SDA00/TOOLRxD) for serial data or serial clock input, requires setting the corresponding bit in the port mode register (PMxx) to 1. In this case, the corresponding bit in the port register (Pxx) can be set to 0 or 1.

When the TTL input buffer is selected, set the corresponding bit in the port input mode register (PIMxx) to 1. When connecting an external device operating on a different potential (1.8 V, 2.5 V or 3 V), see **4.4.4 Connecting to external device with different potential (1.8 V, 2.5 V, 3 V)**.

Example: When using P06/SI00/RxD0/TI03/TO03/SDA00/TOOLRxD for serial data input
Set the PM06 bit of port mode register 0 to 1.
Set the P06 bit of port register 0 to 0 or 1.

The PM0, PM1 and PM8 registers can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets the PM0, PM1 and PM8 registers to FFH.

See Tables 4-3 to see which PMxx registers are provided for each product.

## 18.4 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.

### 18.4.1 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 supplying the clock to the peripheral hardware. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

To stop the operation of serial array unit 0, set bit 2 (SAU0EN) to 0.

To stop the operation of serial array unit 1, set bit 3 (SAU1EN) to 0.

**Figure 18-23. Peripheral Enable Register 0 (PER0) Setting When Stopping the Operation by Units**

**(a) Peripheral enable register 0 (PER0) … Set only the bit of SAUm to be stopped to 0.**

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PER0 | RTCWEN | IRDAEN | ADCEN | IICA0EN | SAU1EN | SAU0EN | 0 | TAU0EN |
| | × | × | × | × | 0/1 | 0/1 | | × |

Control of SAUm input clock
0: Stops supply of input clock
1: Supplies input clock

**Cautions 1. If SAUmEN = 0, writing to a control register of serial array unit m is ignored, and, even if the register is read, only the default 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 0, 1, 8 (PIM0, PIM1, PIM8)**
- **Port output mode registers 0, 1, 8 (POM0, POM1, POM8)**
- **Port mode registers 0, 1, 8 (PM0, PM1, PM8)**
- **Port registers 0, 1, 8 (P0, P1, P8)**

**2. Be sure to clear bit 1 to 0.**

**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 usage of the user

### 18.4.2 Stopping the operation by channels

The stopping of the operation by channels is set using each of the following registers.

**Figure 18-24. Each Register Setting When Stopping the Operation by Channels**

**(a) Serial channel stop register m (STm) … This register is a trigger register that is used to enable stopping communication/count by each channel.**

STm

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | STm3 Note 0/1 | STm2 Note 0/1 | STm1 0/1 | STm0 0/1 |

1: Clears the SEmn bit to 0 and stops the communication operation

* 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.**

SEm

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SEm3 Note 0/1 | SEm2 Note 0/1 | SEm1 0/1 | SEm0 0/1 |

0: Operation stops

* The SEm register 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 a register that is used to enable or stop output of the serial communication operation of each channel.**

SOEm

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SOEm2 Note 0/1 | 0 | SOEm0 0/1 |

0: Stops output by serial communication operation

* 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) …This register is a buffer register for serial output of each channel.**

SOm

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | CKOm0 Note 0/1 | 0 | 0 | 0 | 0 | 1 | SOm2 Note 0/1 | 1 | SOm0 0/1 |

1: Serial clock output value is "1" 1: Serial data output value is "1"

* When using pins corresponding to each channel as port function pins, set the corresponding CKOmn, SOmn bits to "1".

**Note** When serial array unit 0 only.

**Remarks 1. m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)**

**2. ☐ : Setting disabled (fixed by hardware), 0/1: Set to 0 or 1 depending on the usage of the user**

## 18.5 Operation of 3-Wire Serial I/O (CSI00) Communication

This is a clocked communication function that uses three lines: serial clock (SCK) and serial data (SI and SO) lines.

[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. $f_{MCK}/2$
  - During slave communication: Max. $f_{MCK}/6$

[Interrupt function]

- Transfer end interrupt/buffer empty interrupt

[Error detection flag]

- Overrun error

In addition, CSI00 supports the SNOOZE mode. When SCK input is detected while in the STOP mode, the SNOOZE mode makes data reception that does not require the CPU possible. CSI00 supports the asynchronous reception.

**Note** Use the clocks within a range satisfying the SCK cycle time ($t_{KCY}$) characteristics. For details, see **CHAPTER 37 ELECTRICAL SPECIFICATIONS**.

The channels supporting 3-wire serial I/O (CSI00) are channels 0 of SAU0.

| Unit | Channel | Used as CSI | Used as UART | Used as Simplified $I^2C$ |
|---|---|---|---|---|
| 0 | 0 | CSI00 | UART0 (supporting LIN-bus) | IIC00 |
| | 1 | – | | – |
| | 2 | – | UART1 | IIC10 |
| | 3 | – | | – |
| 1 | 0 | – | UART2 | – |
| | 1 | – | | – |

3-wire serial I/O (CSI00) performs the following seven types of communication operations.

- Master transmission (See **18.5.1**.)
- Master reception (See **18.5.2**.)
- Master transmission/reception (See **18.5.3**.)
- Slave transmission (See **18.5.4**.)
- Slave reception (See **18.5.5**.)
- Slave transmission/reception (See **18.5.6**.)
- SNOOZE mode function (See **18.5.7**.)

### 18.5.1 Master transmission

Master transmission is that the RL78 microcontroller outputs a transfer clock and transmits data to another device.

| 3-Wire Serial I/O | CSI00 |
|---|---|
| Target channel | Channel 0 of SAU0 |
| Pins used | SCK00, SO00 |
| Interrupt | INTCSI00 |
| | 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]<br>Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [Hz] $f_{CLK}$: System clock frequency |
| Data phase | Selectable by the DAPmn bit of the SCRmn register<br>• DAPmn = 0: Data output starts from the start of the operation of the serial clock.<br>• DAPmn = 1: Data output starts half a clock before the start of the serial clock operation. |
| Clock phase | Selectable by the CKPmn bit of the SCRmn register<br>• CKPmn = 0: Non-reverse (data output at the falling edge and data input at the rising edge of SCK)<br>• CKPmn = 1: Reverse (data output at the rising edge and data input at the falling edge of SCK) |
| Data direction | MSB or LSB first |

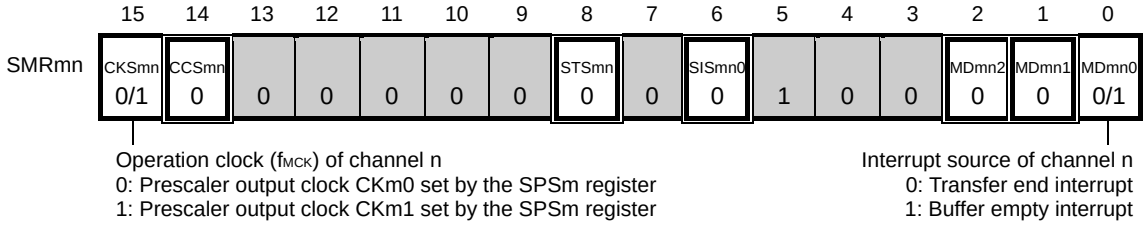
**Note** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 37 ELECTRICAL SPECIFICATIONS**).

**Remark** m: Unit number (m = 0), n: Channel number (n = 0), mn = 00

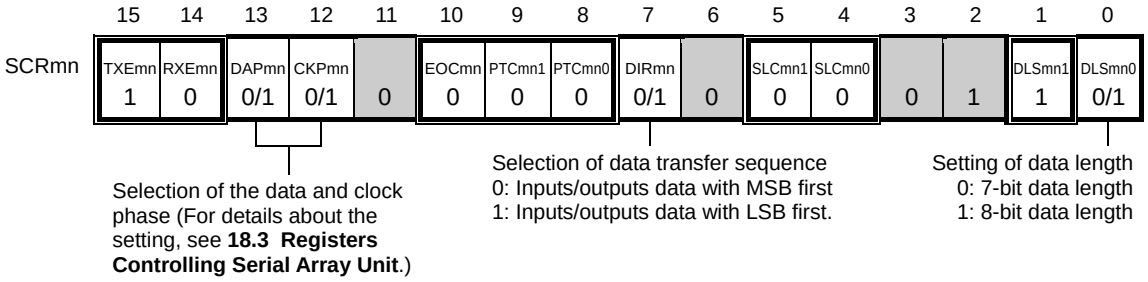
### (1) Register setting

**Figure 18-25. Example of Contents of Registers for Master Transmission of 3-Wire Serial I/O (CSI00) (1/2)**

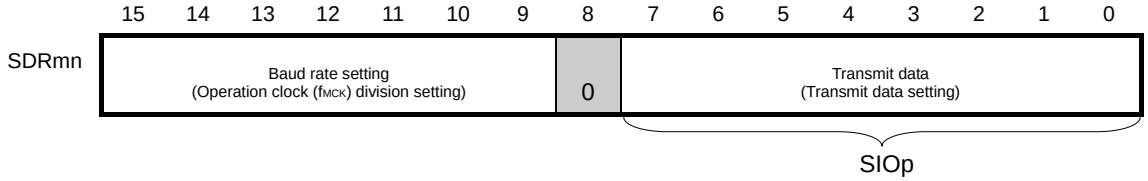
**(a) Serial mode register mn (SMRmn)**



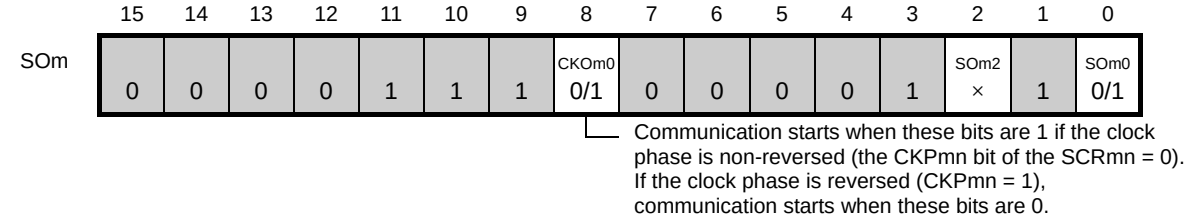

**(b) Serial communication operation setting register mn (SCRmn)**




**(c) Serial data register mn (SDRmn) (lower 8 bits: SIOp)**




**(d) Serial output register m (SOm) … Sets only the bits of the target channel.**




**Note** Only provided for the SCR00 register. This bit is fixed to 1 for the other registers.

**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0, 2), p: CSI number (p = 00, 10), mn = 00

**2.** ☐: Setting is fixed in the CSI master transmission mode, ▒: Setting disabled (set to the initial value)
0/1: Set to 0 or 1 depending on the usage of the user

**Figure 18-25. Example of Contents of Registers for Master Transmission of 3-Wire Serial I/O (CSI00) (2/2)**

**(e) Serial output enable register m (SOEm) … Sets only the bits of the target channel to 1.**

| | 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 | 0 | SOEm2<br>× | 0 | SOEm0<br>0/1 |

**(f) Serial channel start register m (SSm) … Sets only the bits 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<br>× | SSm2<br>× | SSm1<br>× | SSm0<br>0/1 |

**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0, 2), p: CSI number (p = 00), mn = 00

**2.** ☐: 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 usage of the user

### (2) Operation procedure

**Figure 18-26. Initial Setting Procedure for Master Transmission**

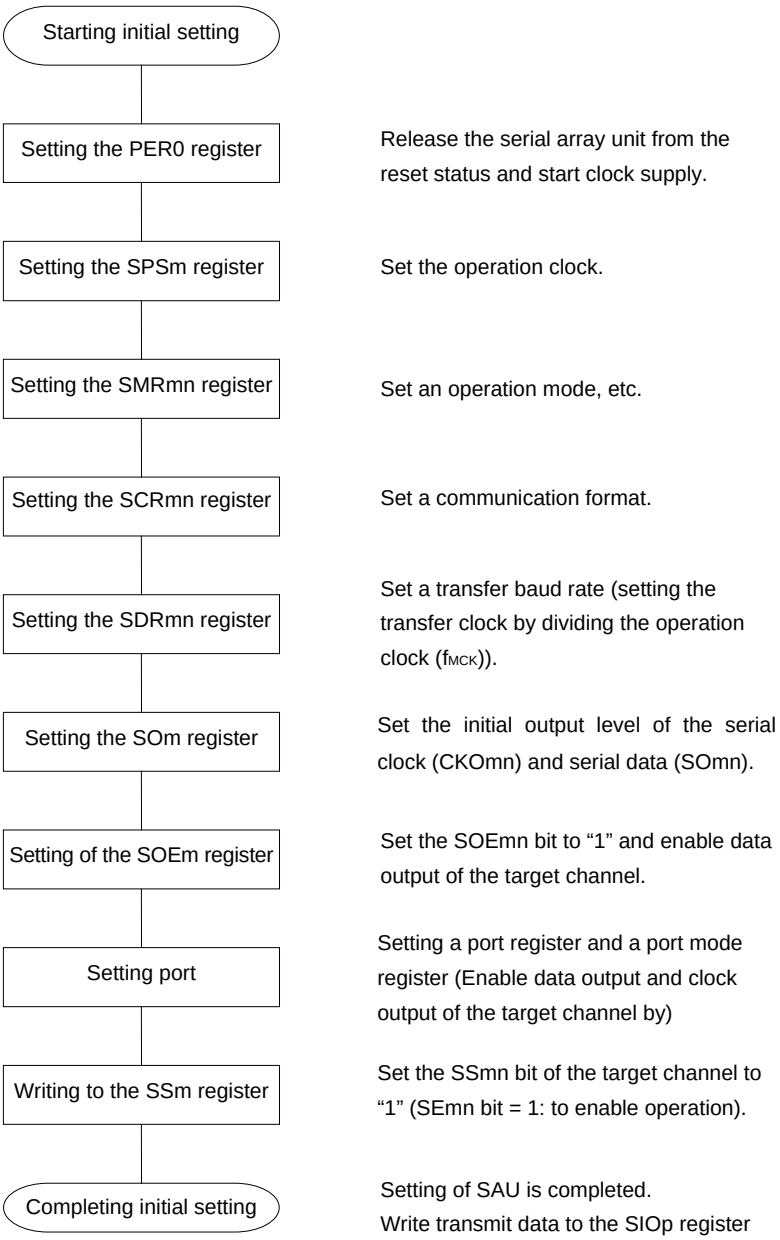


**Figure 18-27. Procedure for Stopping Master Transmission**

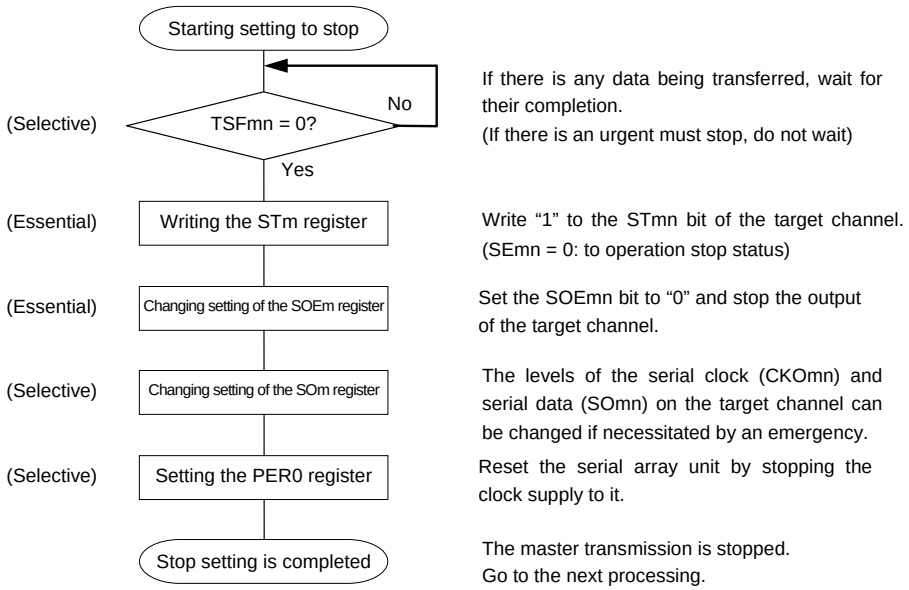
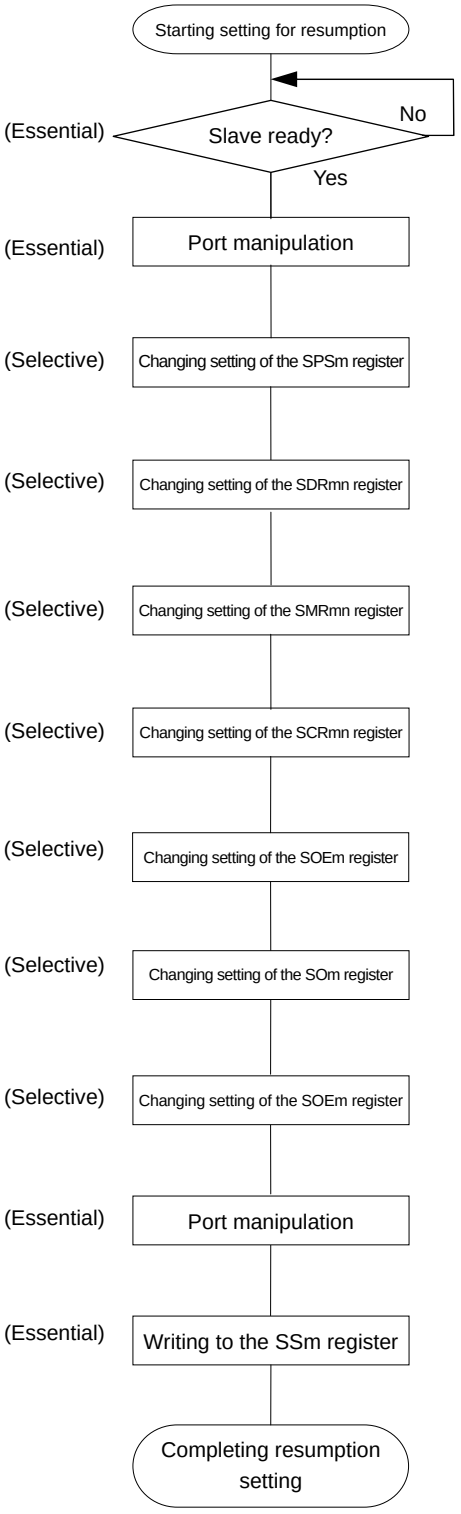


**Figure 18-28. Procedure for Resuming Master Transmission**




Wait until stop the communication target (slave) or communication operation completed

Disable data output and clock output of the target channel by setting a port register and a port mode register.

Re-set the register to change the operation clock setting.

Re-set the register to change the transfer baud rate setting (setting the transfer clock by dividing the operation clock ($f_{MCK}$)).

Re-set the register to change serial mode register mn (SMRmn) setting.

Re-set the register to change serial communication operation setting register mn (SCRmn) setting.

Set the SOEmn bit to “0” to stop output from the target channel.

Set the initial output level of the serial clock (CKOmn) and serial data (SOmn).

Set the SOEmn bit to “1” and enable output from the target channel.

Enable data output and clock output of the target channel by setting a port register and a port mode register.

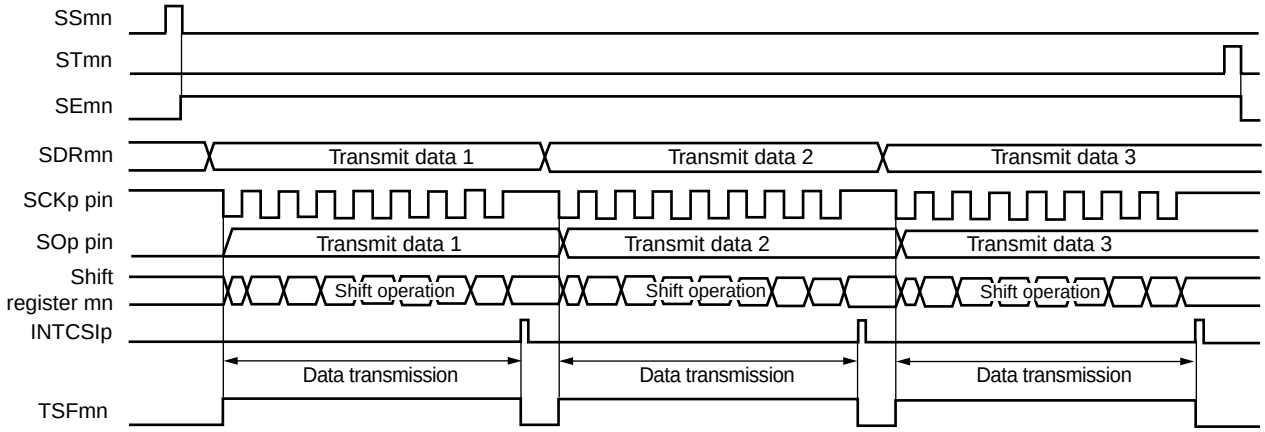
Set the SSmn bit of the target channel to “1” (SEmn = 1: to enable operation).

Setting is completed
Sets transmit data to the SIOp register (bits 7 to 0 of the SDRmn register) and start communication.

**Remark** If PER0 is rewritten while stopping the master transmission and the clock supply is stopped, wait until the transmission target (slave) stops or transmission finishes, and then perform initialization instead of restarting the transmission.

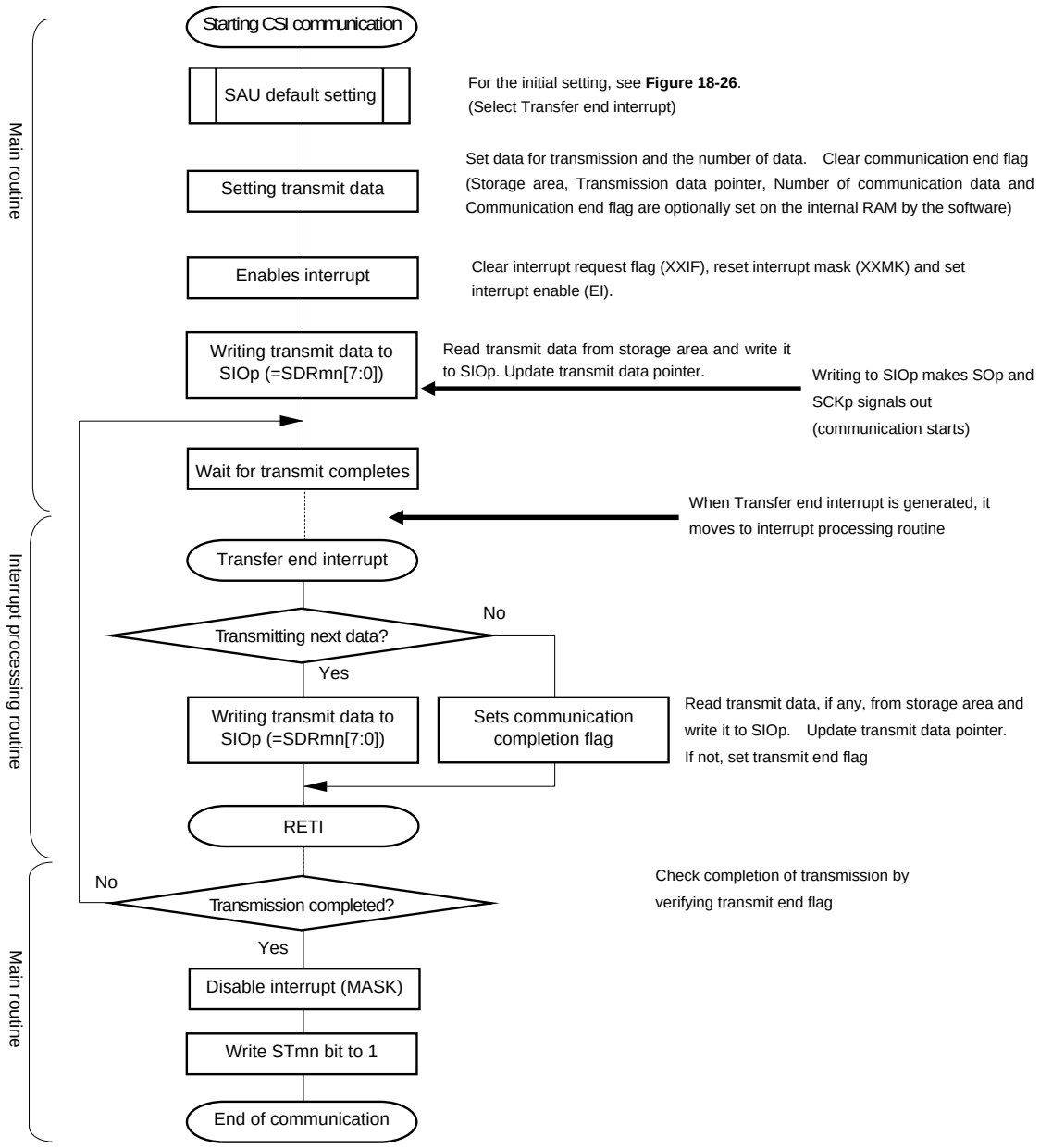
### (3) Processing flow (in single-transmission mode)

**Figure 18-29. Timing Chart of Master Transmission (in Single-Transmission Mode)**
**(Type 1: DAPmn = 0, CKPmn = 0)**



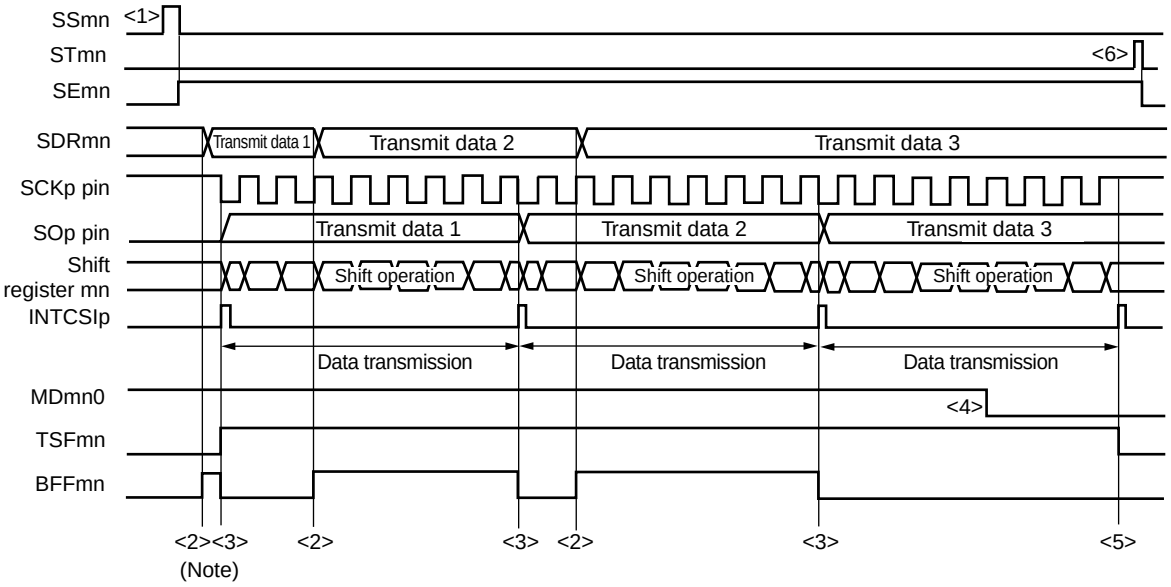

**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

**Figure 18-30. Flowchart of Master Transmission (in Single-Transmission Mode)**



### (4) Processing flow (in continuous transmission mode)

**Figure 18-31. Timing Chart of Master Transmission (in Continuous Transmission Mode)**
**(Type 1: DAPmn = 0, CKPmn = 0)**

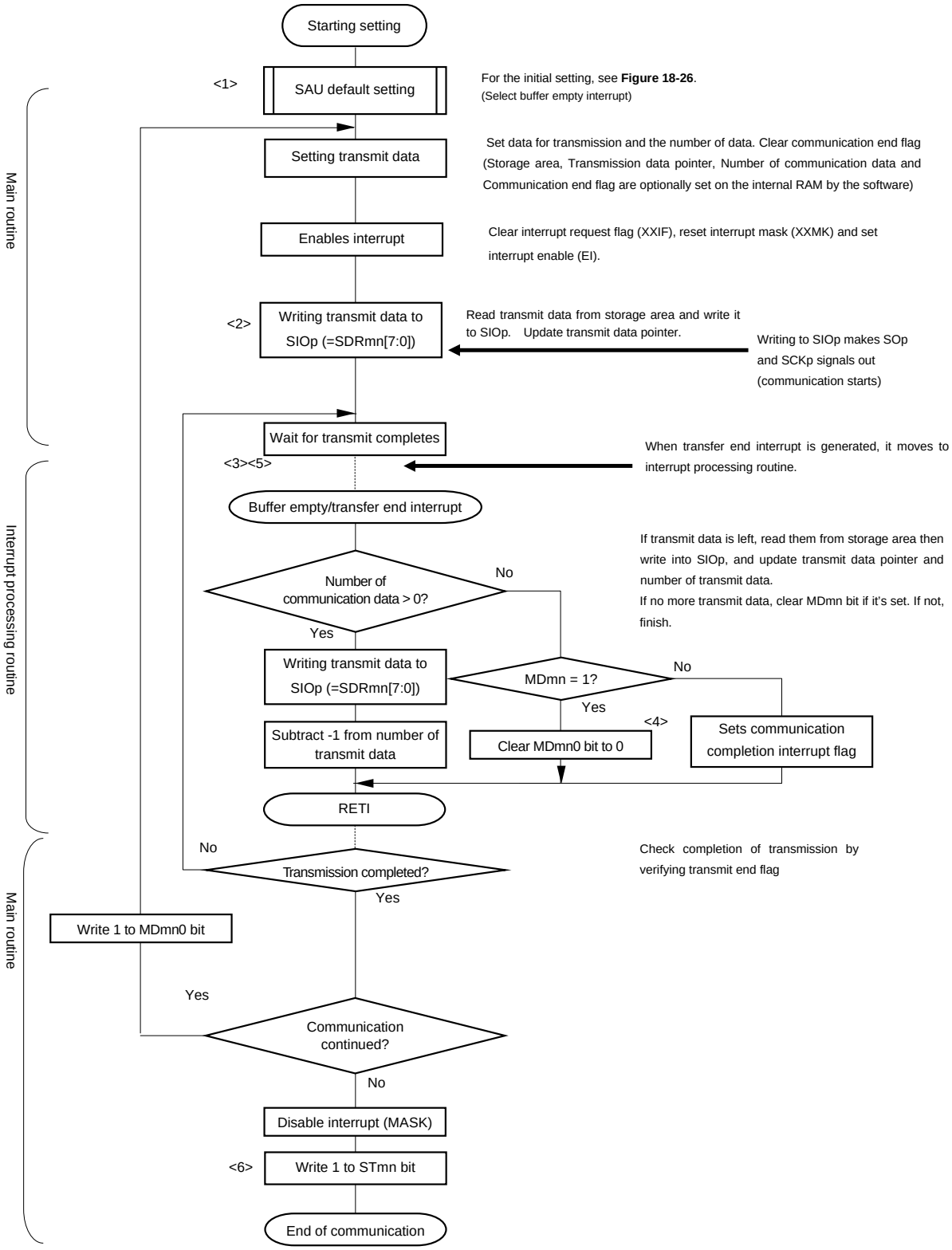



**Note** If transmit data is written to the SDRmn register while the BFFmn bit of serial status register mn (SSRmn) is 1 (valid data is stored in serial data register mn (SDRmn)), the transmit data is overwritten.

**Caution The MDmn0 bit of serial mode register mn (SMRmn) can be rewritten even during operation. However, rewrite it before transfer of the last bit is started, so that it will be rewritten before the transfer end interrupt of the last transmit data.**

**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

**Figure 18-32. Flowchart of Master Transmission (in Continuous Transmission Mode)**




**Remark** <1> to <6> in the figure correspond to <1> to <6> in **Figure 18-31 Timing Chart of Master Transmission (in Continuous Transmission Mode)**.

### 18.5.2 Master reception

Master reception is that the RL78 microcontroller outputs a transfer clock and receives data from other device.

| 3-Wire Serial I/O | CSI00 |
|---|---|
| Target channel | Channel 0 of SAU0 |
| Pins used | SCK00, SI00 |
| Interrupt | INTCSI00 |
| | 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]<br>Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [Hz] $f_{CLK}$: System clock frequency |
| Data phase | Selectable by the DAPmn bit of the SCRmn register<br>• DAPmn = 0: Data input starts from the start of the operation of the serial clock.<br>• DAPmn = 1: Data input starts half a clock before the start of the serial clock operation. |
| Clock phase | Selectable by the CKPmn bit of the SCRmn register<br>• CKPmn = 0: Non-reverse<br>• CKPmn = 1: Reverse |
| Data direction | MSB or LSB first |

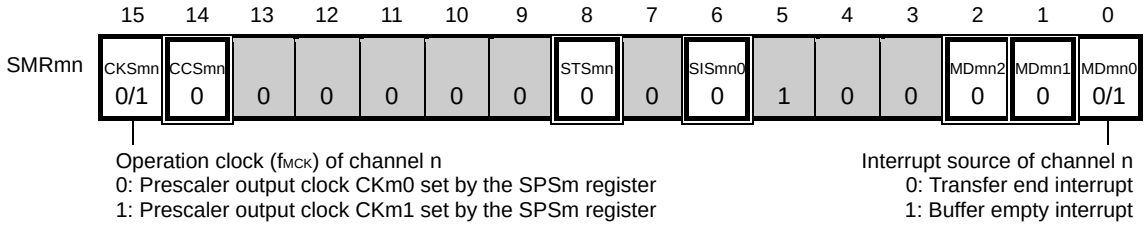
**Note** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 37 ELECTRICAL SPECIFICATIONS**).

**Remark** m: Unit number (m = 0), n: Channel number (n = 0), mn = 00

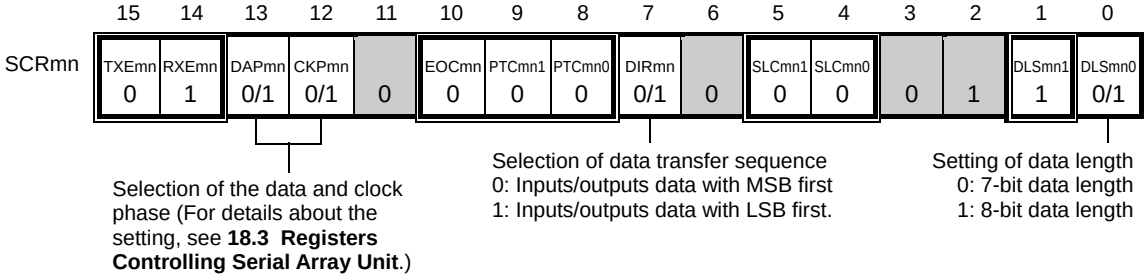
### (1) Register setting

**Figure 18-33. Example of Contents of Registers for Master Reception of 3-Wire Serial I/O (CSI00) (1/2)**

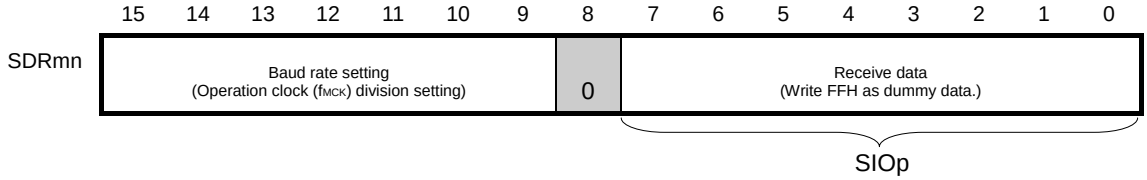
**(a) Serial mode register mn (SMRmn)**



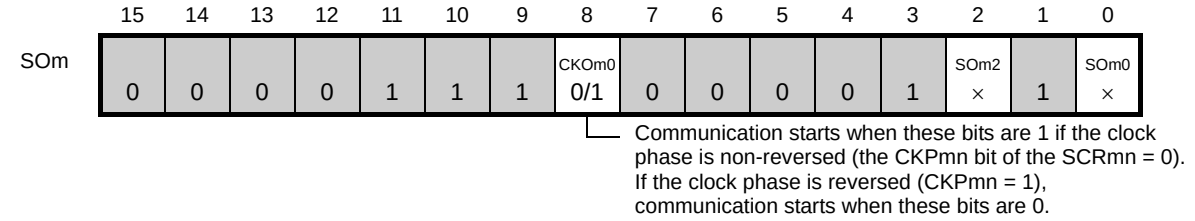

**(b) Serial communication operation setting register mn (SCRmn)**




**(c) Serial data register mn (SDRmn) (lower 8 bits: SIOp)**




**(d) Serial output register m (SOm) … Sets only the bits of the target channel.**




**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0, 2), p: CSI number (p = 00), mn = 00

**2.** ☐: Setting is fixed in the 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 usage of the user

**Figure 18-33. Example of Contents of Registers for Master Reception of 3-Wire Serial I/O (CSI00) (2/2)**

**(e) Serial output enable register m (SOEm) …The register that not used in this mode.**

| | 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 | 0 | SOEm2 × | 0 | SOEm0 × |

**(f) Serial channel start register m (SSm) … Sets only the bits 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 × | SSm1 × | SSm0 0/1 |

**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

**2.** ☐ : 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 usage of the user

### (2) Operation procedure

**Figure 18-34. Initial Setting Procedure for Master Reception**

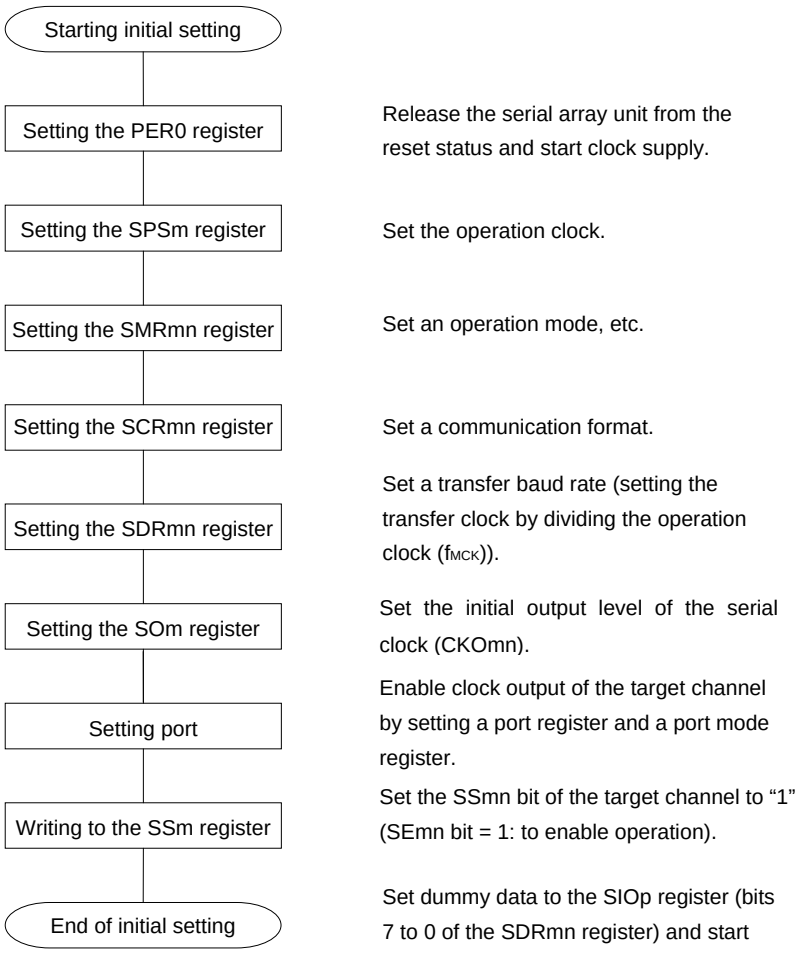



**Figure 18-35. Procedure for Stopping Master Reception**

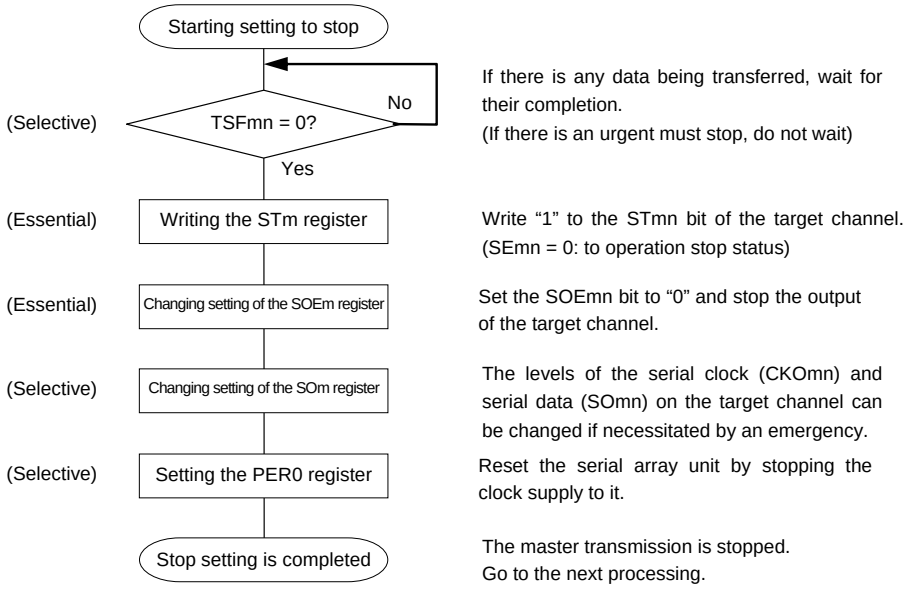
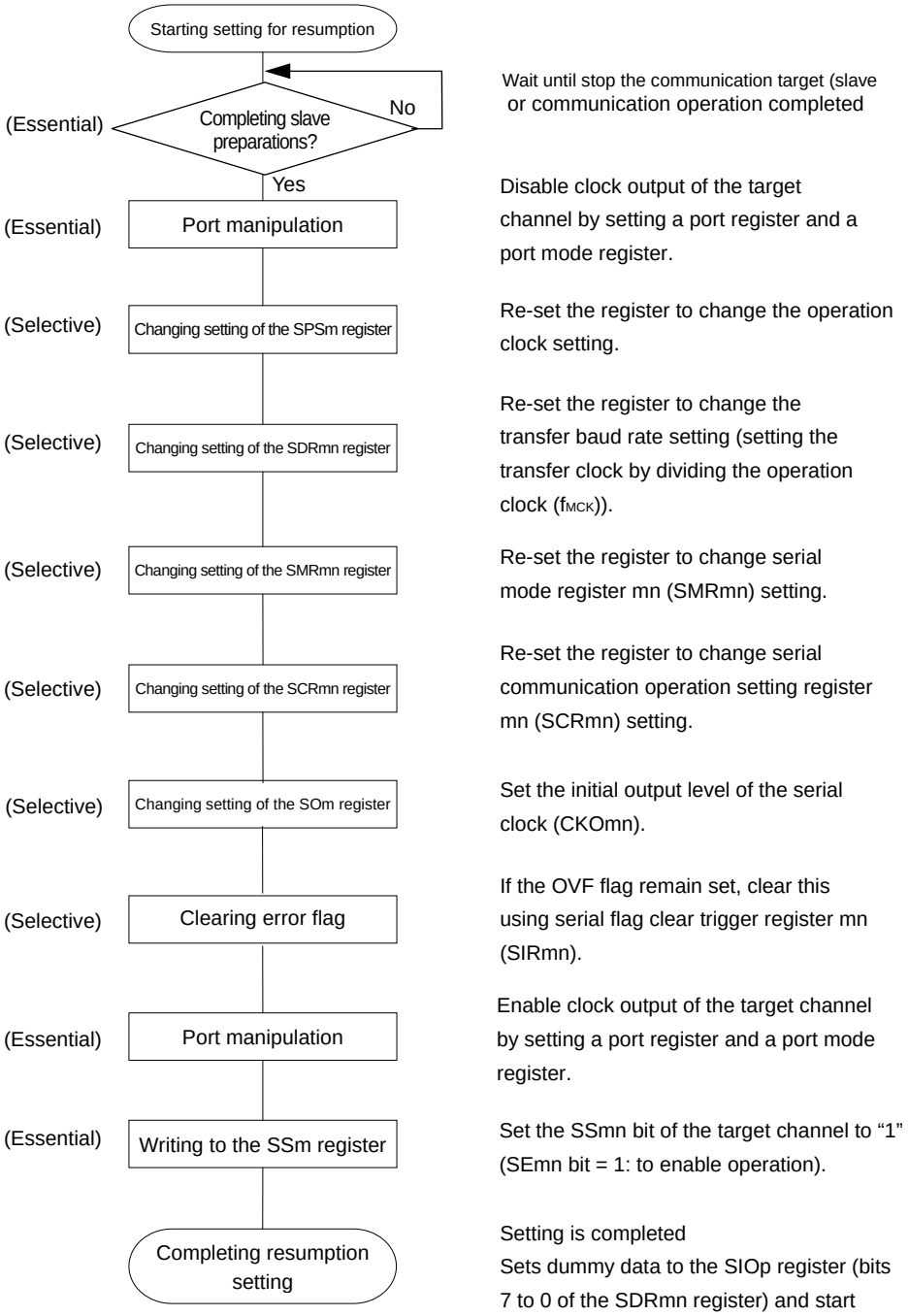


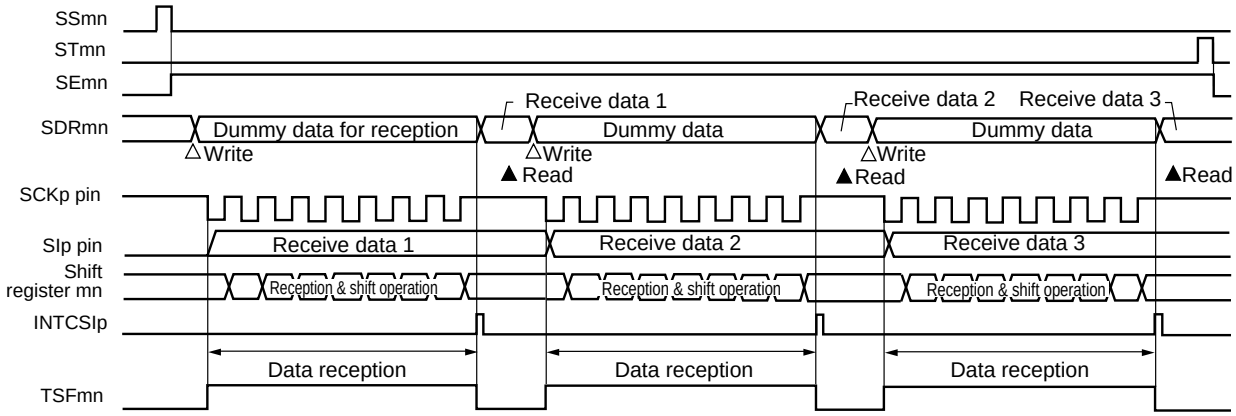
**Figure 18-36. Procedure for Resuming Master Reception**




**Remark** If PER0 is rewritten while stopping the master transmission and the clock supply is stopped, wait until the transmission target (slave) stops or transmission finishes, and then perform initialization instead of restarting the transmission.

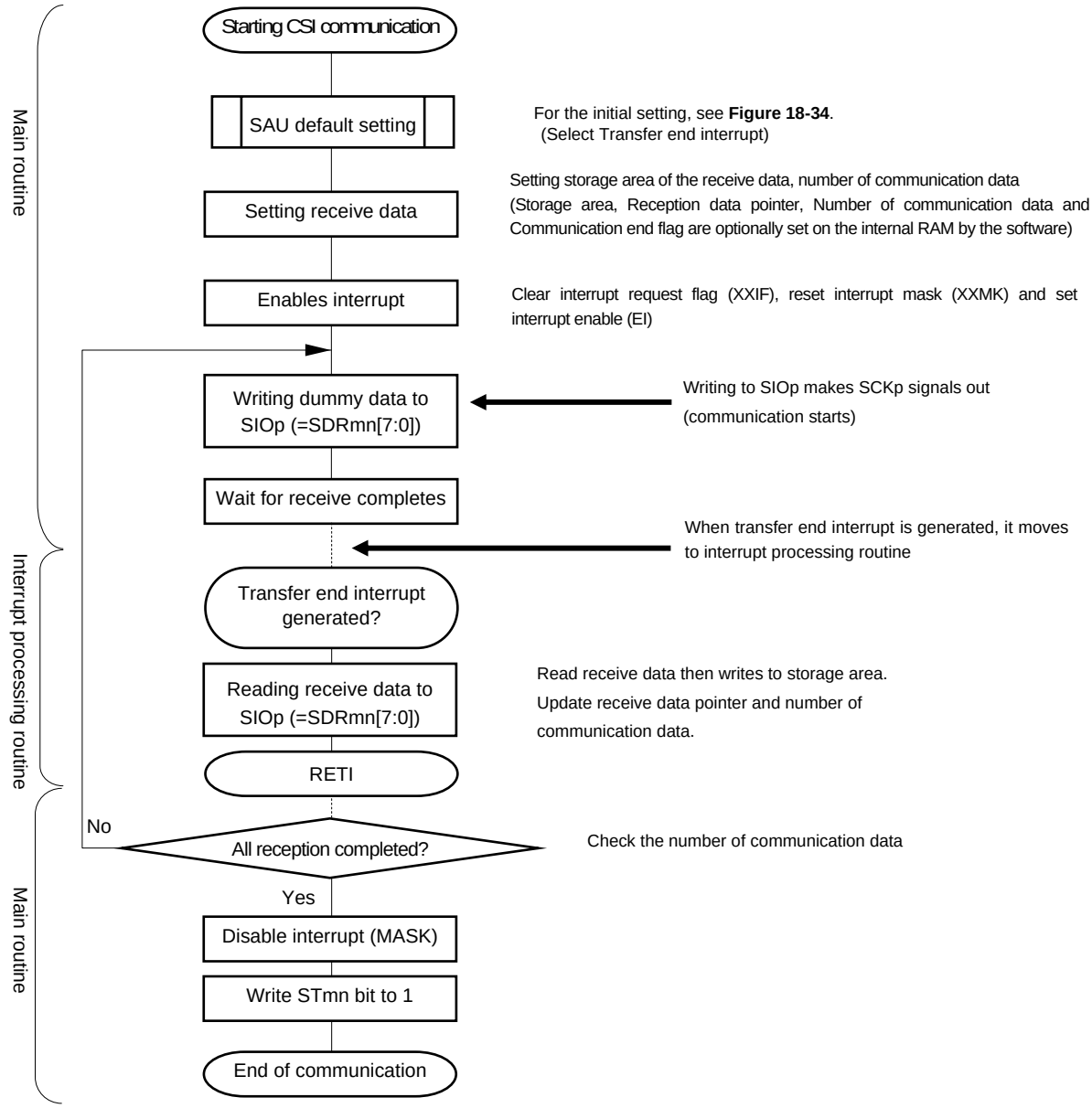
### (3) Processing flow (in single-reception mode)

**Figure 18-37. Timing Chart of Master Reception (in Single-reception Mode)**
**(Type 1: DAPmn = 0, CKPmn = 0)**



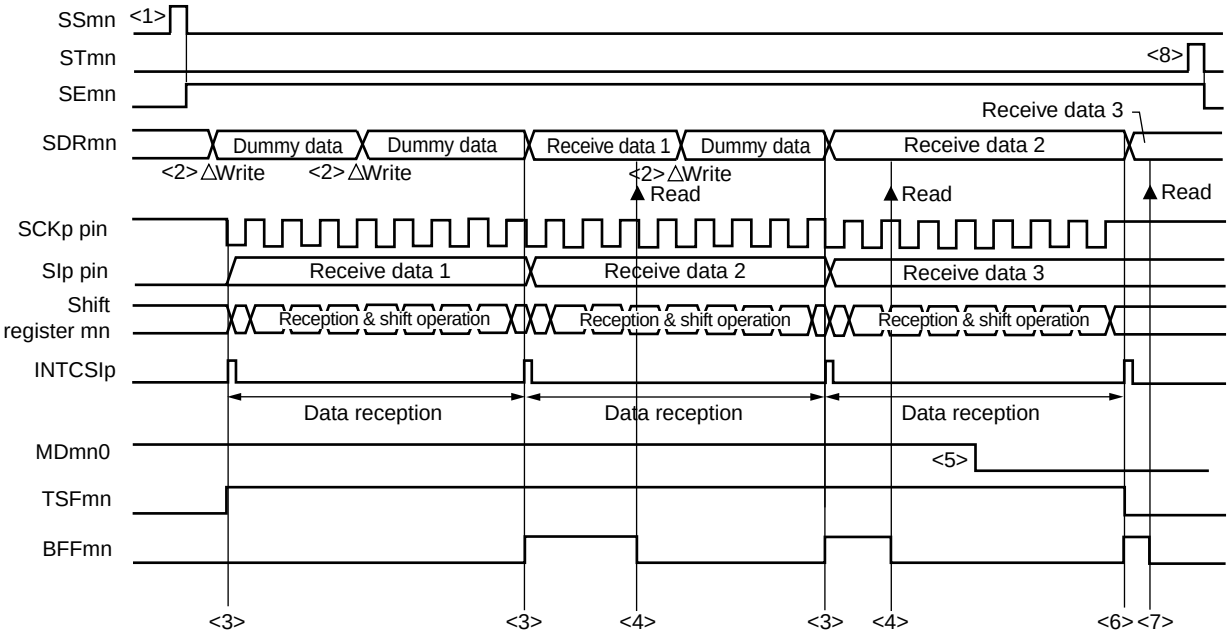

**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

**Figure 18-38. Flowchart of Master Reception (in Single-reception Mode)**



### (4) Processing flow (in continuous reception mode)

**Figure 18-39. Timing Chart of Master Reception (in Continuous Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)**

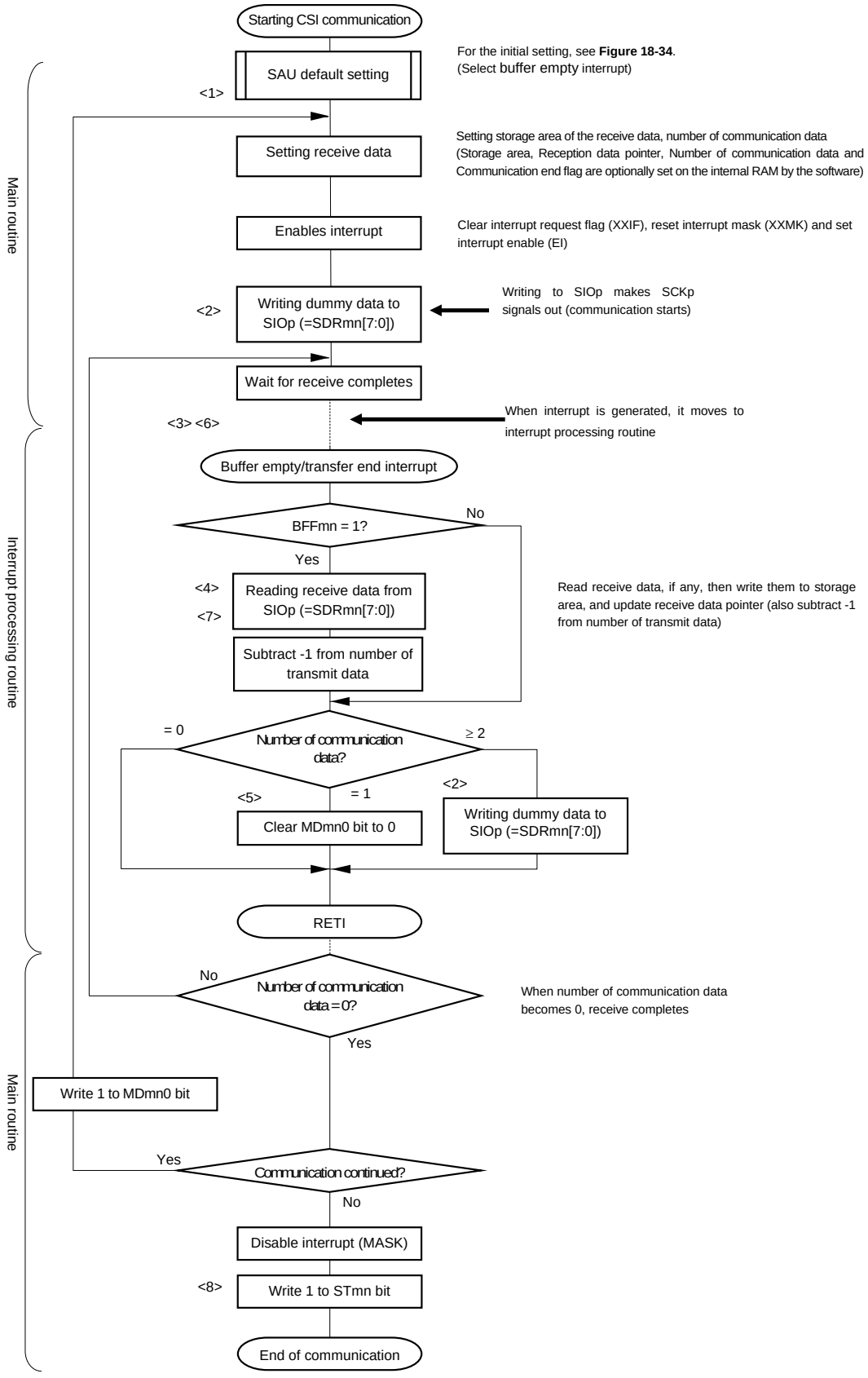



**Caution The MDmn0 bit can be rewritten even during operation.**
**However, rewrite it before receive of the last bit is started, so that it has been rewritten before the transfer end interrupt of the last receive data.**

**Remarks 1.** <1> to <8> in the figure correspond to <1> to <8> in **Figure 18-40 Flowchart of Master Reception (in Continuous Reception Mode)**.

**2.** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

**Figure 18-40. Flowchart of Master Reception (in Continuous Reception Mode)**




**Remark** <1> to <8> in the figure correspond to <1> to <8> in **Figure 18-39 Timing Chart of Master Reception (in Continuous Reception Mode)**.

### 18.5.3 Master transmission/reception

Master transmission/reception is that the RL78 microcontroller outputs a transfer clock and transmits/receives data to/from other device.

| 3-Wire Serial I/O | CSI00 |
|---|---|
| Target channel | Channel 0 of SAU0 |
| Pins used | SCK00, SI00, SO00 |
| Interrupt | INTCSI00 |
| | 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]<br>Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [Hz] $f_{CLK}$: System clock frequency |
| Data phase | Selectable by the DAPmn bit of the SCRmn register<br>• DAPmn = 0: Data I/O starts at the start of the operation of the serial clock.<br>• DAPmn = 1: Data I/O starts half a clock before the start of the serial clock operation. |
| Clock phase | Selectable by the CKPmn bit of the SCRmn register<br>• CKPmn = 0: Non-reverse<br>• CKPmn = 1: Reverse |
| Data direction | MSB or LSB first |

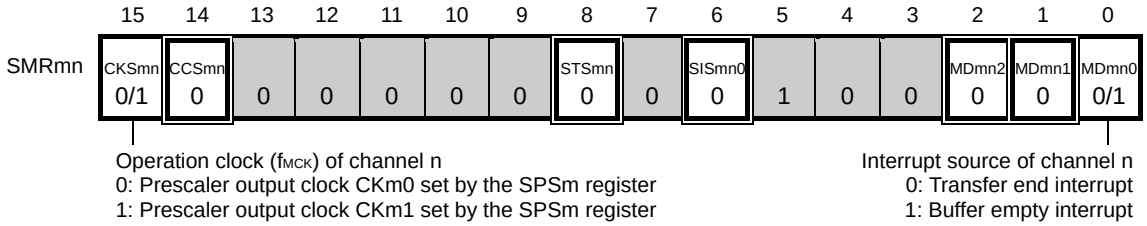
**Note** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 37 ELECTRICAL SPECIFICATIONS**).

**Remark** m: Unit number (m = 0), n: Channel number (n = 0), mn = 00

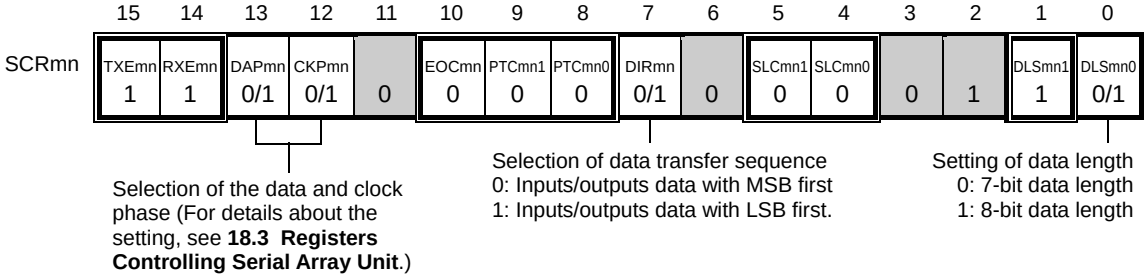
### (1) Register setting

**Figure 18-41. Example of Contents of Registers for Master Transmission/Reception of 3-Wire Serial I/O (CSI00) (1/2)**

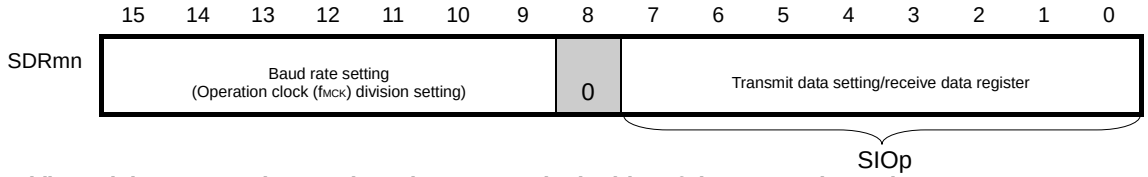
**(a) Serial mode register mn (SMRmn)**



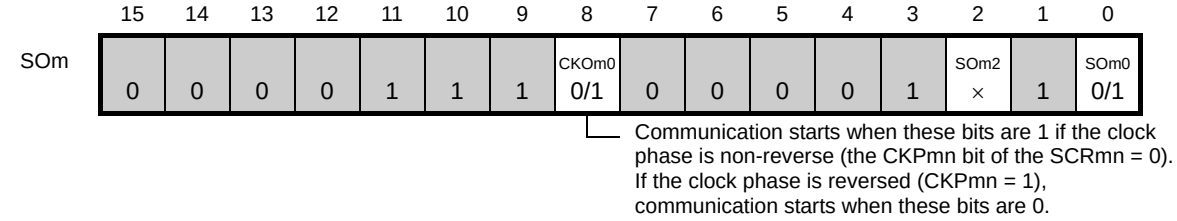

**(b) Serial communication operation setting register mn (SCRmn)**




**(c) Serial data register mn (SDRmn) (lower 8 bits: SIOp)**




**(d) Serial output register m (SOm) … Sets only the bits of the target channel.**




**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

**2.** ☐: Setting is fixed in the CSI master 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 usage of the user

**Figure 18-41. Example of Contents of Registers for Master Transmission/Reception of 3-Wire Serial I/O (CSI00) (2/2)**

**(e) Serial output enable register m (SOEm) … Sets only the bits of the target channel to 1.**

| | 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 | 0 | SOEm2<br>× | 0 | SOEm0<br>0/1 |

**(f) Serial channel start register m (SSm) … Sets only the bits 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<br>× | SSm2<br>× | SSm1<br>× | SSm0<br>0/1 |

**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

**2.** ☐ : 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 usage of the user

### (2) Operation procedure

**Figure 18-42. Initial Setting Procedure for Master Transmission/Reception**

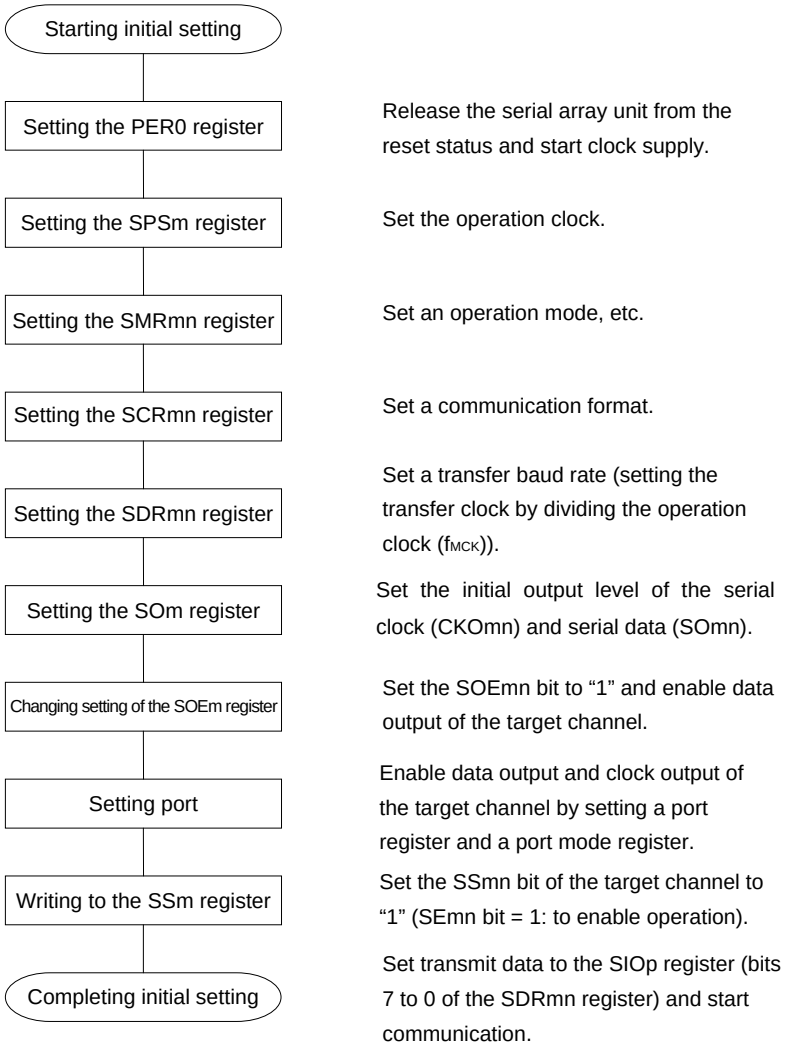


**Figure 18-43. Procedure for Stopping Master Transmission/Reception**

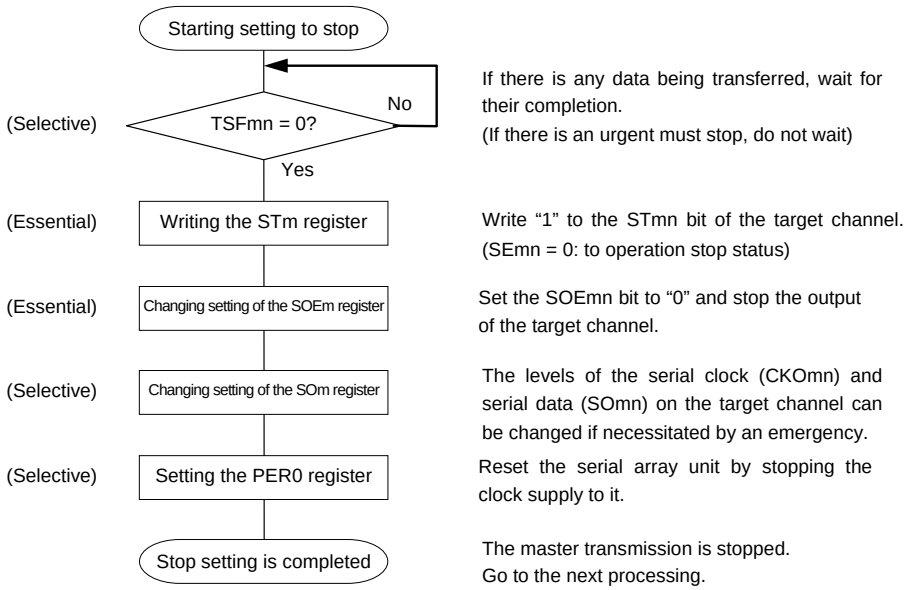
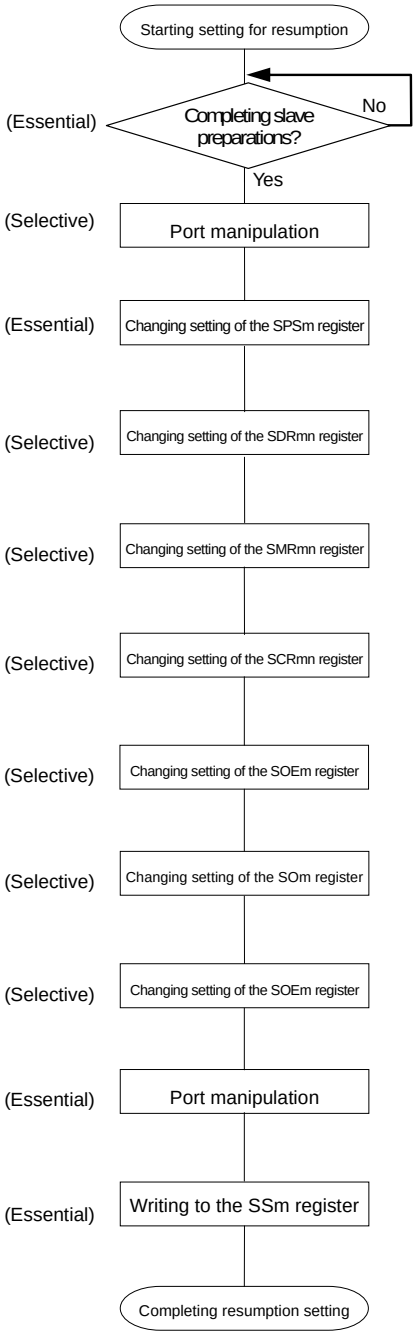


**Figure 18-44. Procedure for Resuming Master Transmission/Reception**




Wait until stop the communication target (slave) or communication operation completed

Disable data output and clock output of the target channel by setting a port register and a port mode register.

Re-set the register to change the operation clock setting.

Re-set the register to change the transfer baud rate setting (setting the transfer clock by dividing the operation clock ($f_{MCK}$)).

Re-set the register to change serial mode register mn (SMRmn) setting.

Re-set the register to change serial communication operation setting register mn (SCRmn) setting.

Set the SOEmn bit to “0” to stop output from the target channel.

Set the initial output level of the serial clock (CKOmn) and serial data (SOmn).

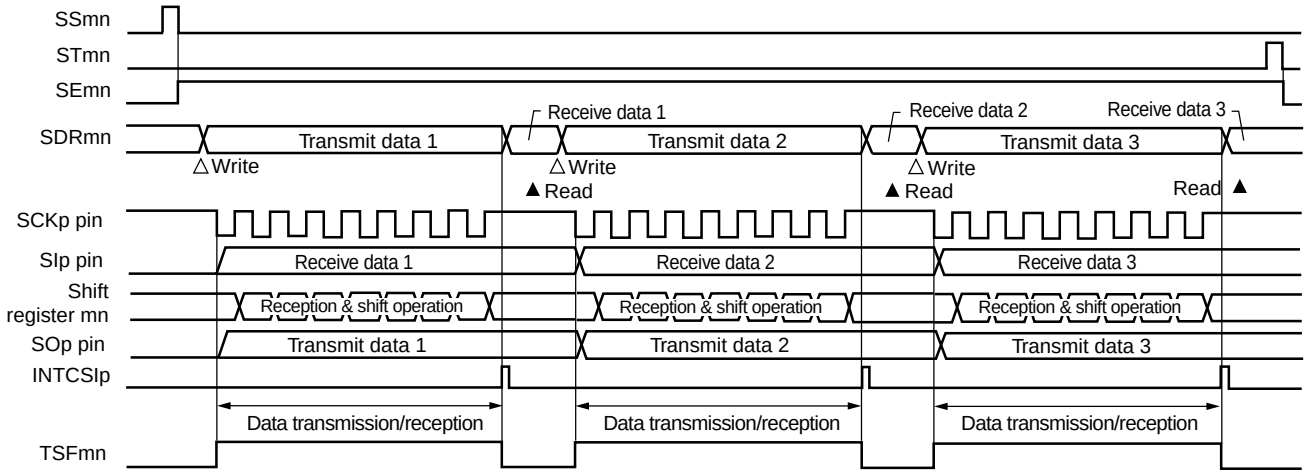
Set the SOEmn bit to “1” and enable output from the target channel.

Enable data output and clock output of the target channel by setting a port register and a port mode register.

Set the SSmn bit of the target channel to “1” (SEmn = 1: to enable operation).

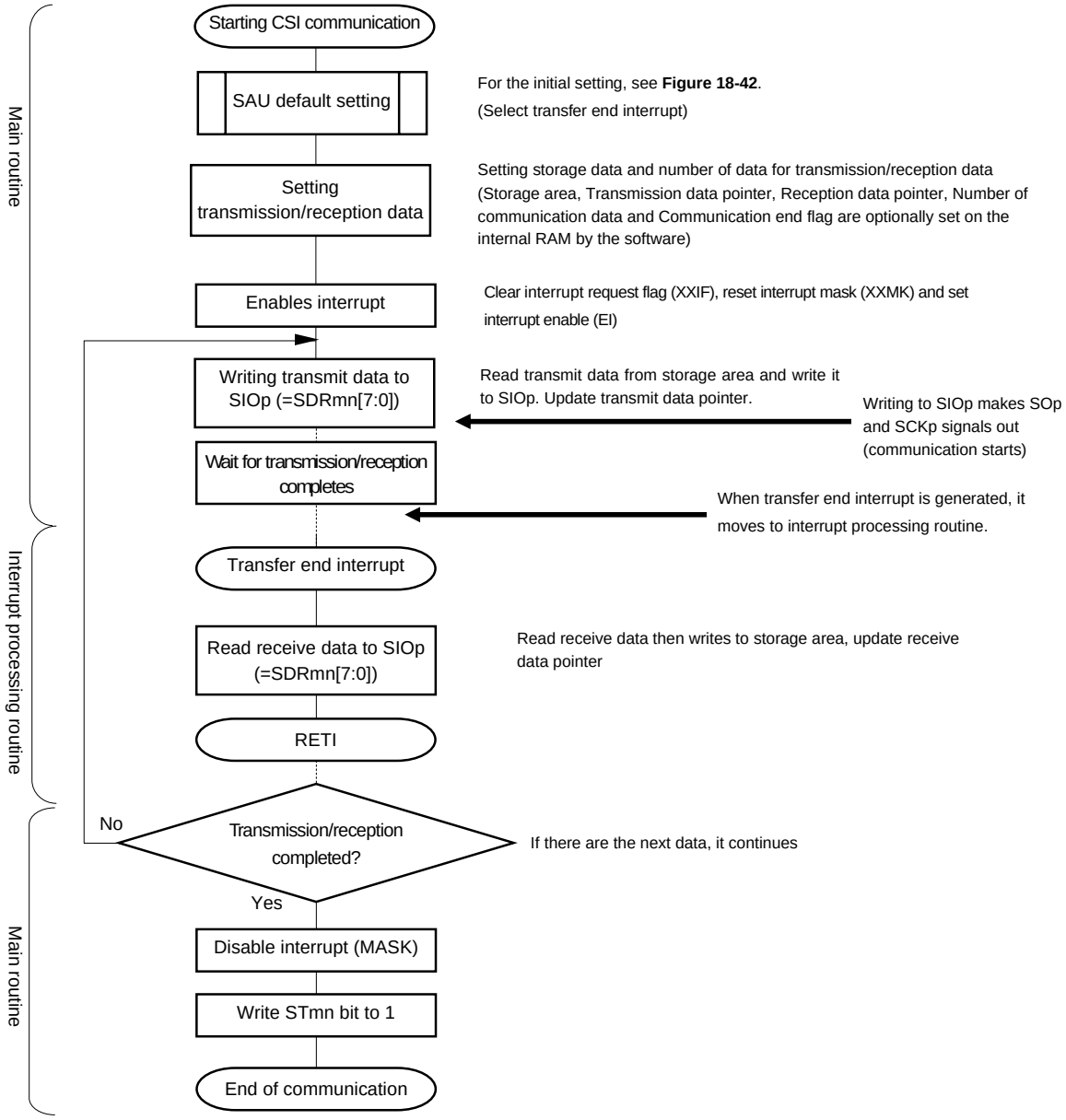
### (3) Processing flow (in single-transmission/reception mode)

**Figure 18-45. Timing Chart of Master Transmission/Reception (in Single-Transmission/reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)**



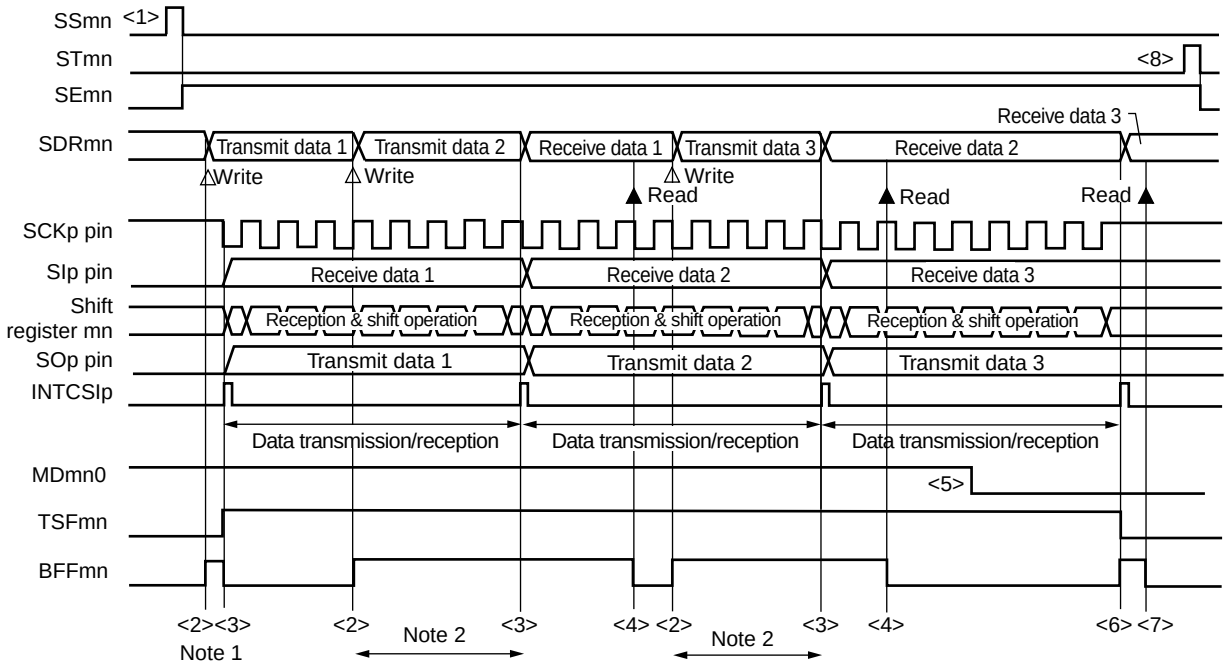

**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

**Figure 18-46. Flowchart of Master Transmission/Reception (in Single-Transmission/reception Mode)**



### (4) Processing flow (in continuous transmission/reception mode)

**Figure 18-47. Timing Chart of Master Transmission/Reception (in Continuous Transmission/Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)**




**Notes 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.

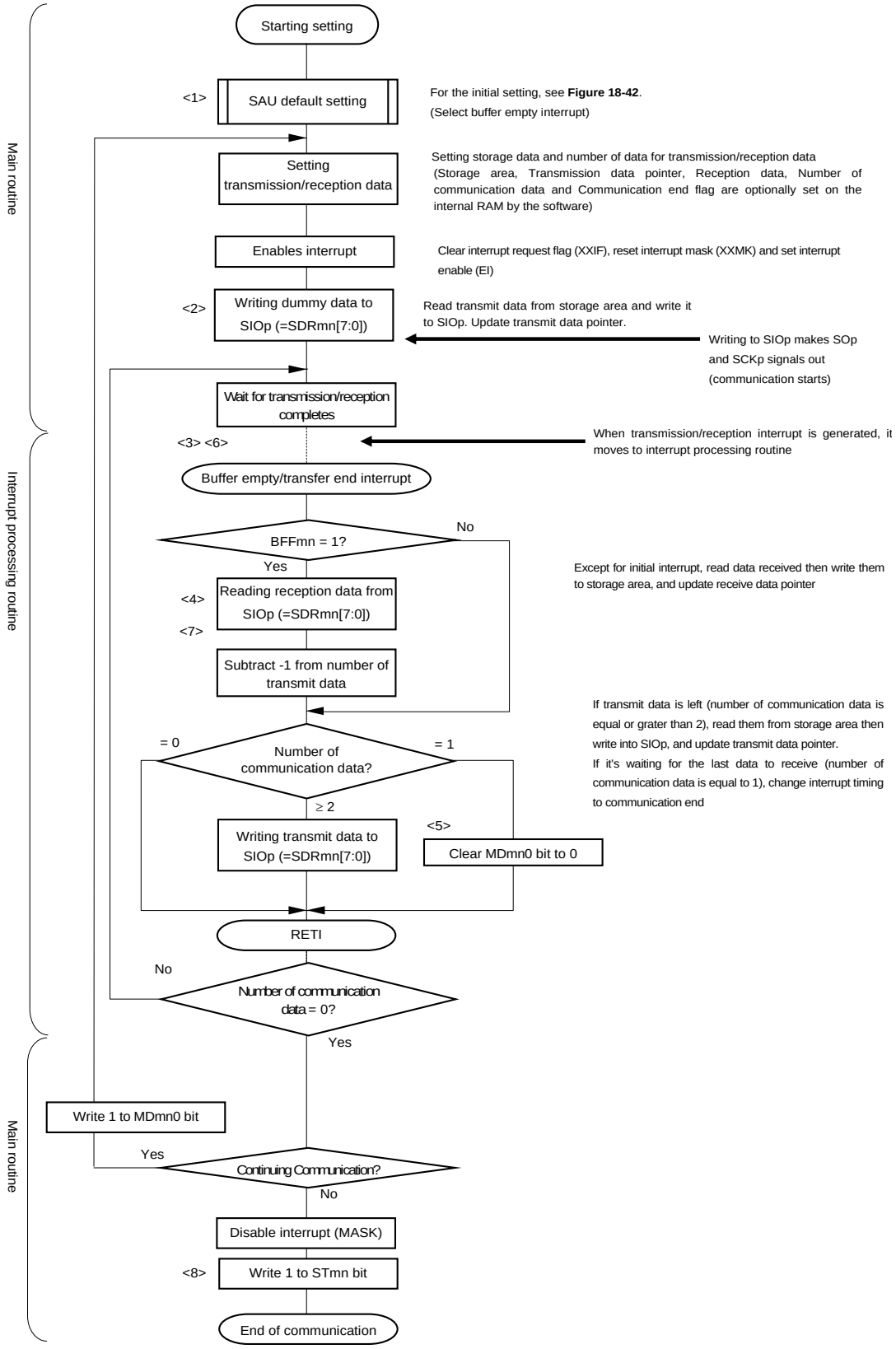
**2.** The transmit data can be read by reading the SDRmn register during this period. At this time, the transfer operation is not affected.

**Caution The MDmn0 bit of serial mode register mn (SMRmn) 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.**

**Remarks 1.** <1> to <8> in the figure correspond to <1> to <8> in **Figure 18-48 Flowchart of Master Transmission/Reception (in Continuous Transmission/Reception Mode**).

**2.** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

**Figure 18-48. Flowchart of Master Transmission/Reception (in Continuous Transmission/Reception Mode)**




**Remark** <1> to <8> in the figure correspond to <1> to <8> in **Figure 18-47 Timing Chart of Master Transmission/Reception (in Continuous Transmission/Reception Mode)**.

### 18.5.4 Slave transmission

Slave transmission is that the RL78 microcontroller transmits data to another device in the state of a transfer clock being input from another device.

| 3-Wire Serial I/O | CSI00 |
|---|---|
| Target channel | Channel 0 of SAU0 |
| Pins used | SCK00, SO00 |
| Interrupt | INTCSI00 |
| | 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<br>• DAPmn = 0: Data output starts from the start of the operation of the serial clock.<br>• DAPmn = 1: Data output starts half a clock before the start of the serial clock operation. |
| Clock phase | Selectable by the CKPmn bit of the SCRmn register<br>• CKPmn = 0: Non-reverse<br>• CKPmn = 1: Reverse |
| Data direction | MSB or LSB first |

**Notes 1.** Because the external serial clock input to the SCK00 pin is sampled internally and used, the fastest transfer rate is $f_{MCK}$/6 [Hz].

**2.** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 37 ELECTRICAL SPECIFICATIONS**).

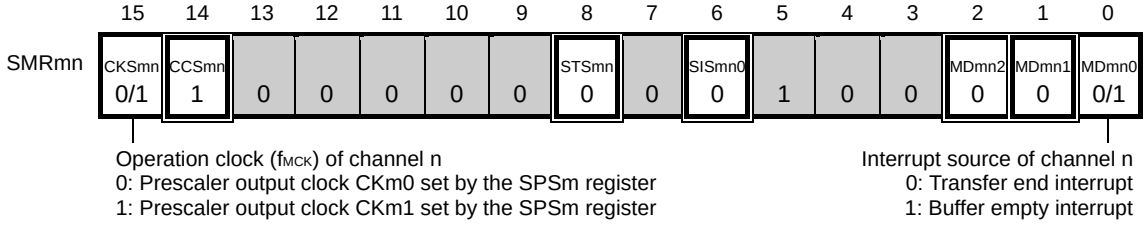
**Remarks 1.** $f_{MCK}$: Operation clock frequency of target channel

**2.** m: Unit number (m = 0), n: Channel number (n = 0), mn = 00

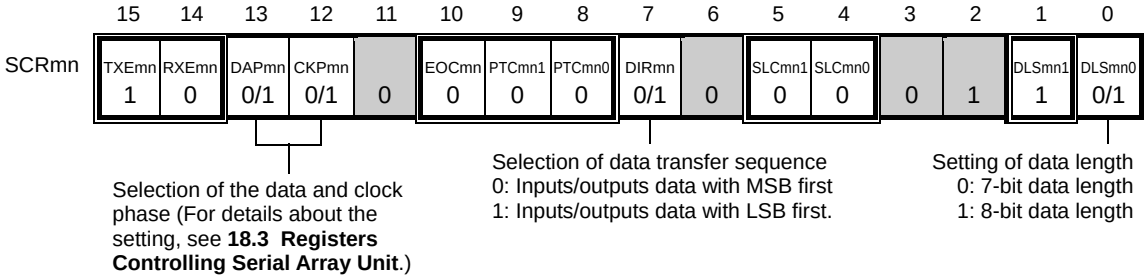
**(1) Register setting**

**Figure 18-49. Example of Contents of Registers for Slave Transmission of 3-Wire Serial I/O (CSI00) (1/2)**

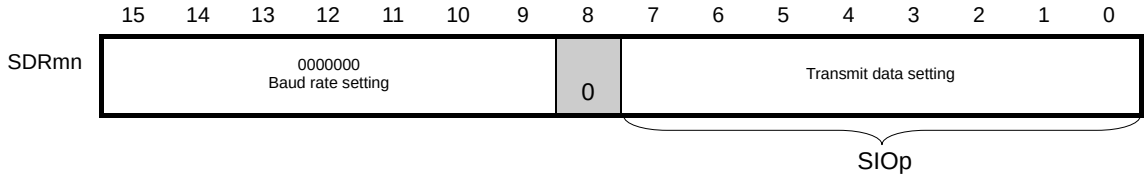
**(a) Serial mode register mn (SMRmn)**



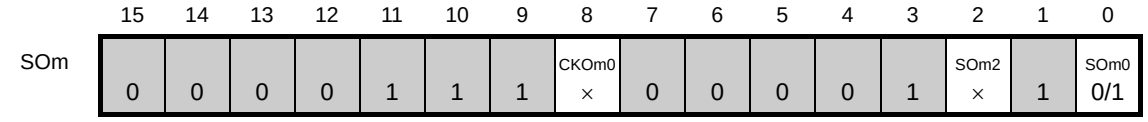

**(b) Serial communication operation setting register mn (SCRmn)**




**(c) Serial data register mn (SDRmn) (lower 8 bits: SIOp)**




**(d) Serial output register m (SOm) … Sets only the bits of the target channel.**




**Note** Only provided for the SCR00 register. This bit is fixed to 1 for the other registers.

**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

**2.** ☐: Setting is fixed in the 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 usage of the user

**Figure 18-49. Example of Contents of Registers for Slave Transmission of 3-Wire Serial I/O (CSI00) (2/2)**

**(e) Serial output enable register m (SOEm) … Sets only the bits of the target channel to 1.**

| | 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 | 0 | SOEm2 × | 0 | SOEm0 0/1 |

**(f) Serial channel start register m (SSm) … Sets only the bits 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 × | SSm1 × | SSm0 0/1 |

**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

**2.** ☐: 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 usage of the user

### (2) Operation procedure

**Figure 18-50. Initial Setting Procedure for Slave Transmission**

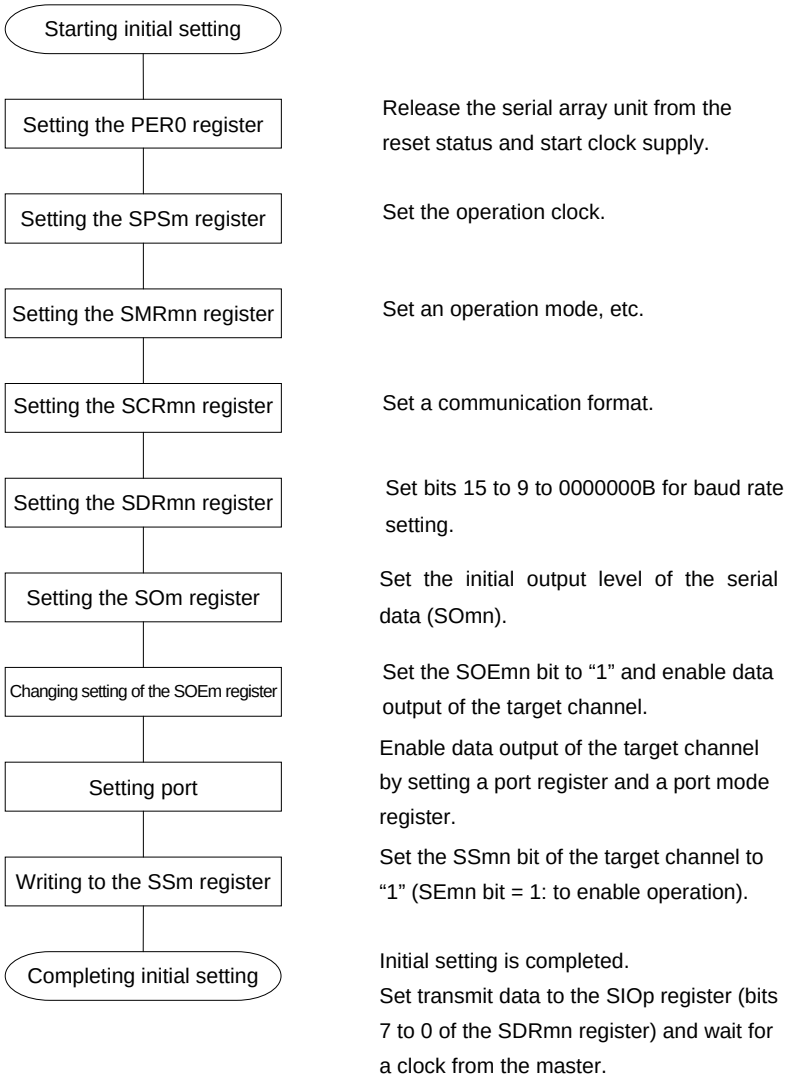


**Figure 18-51. Procedure for Stopping Slave Transmission**

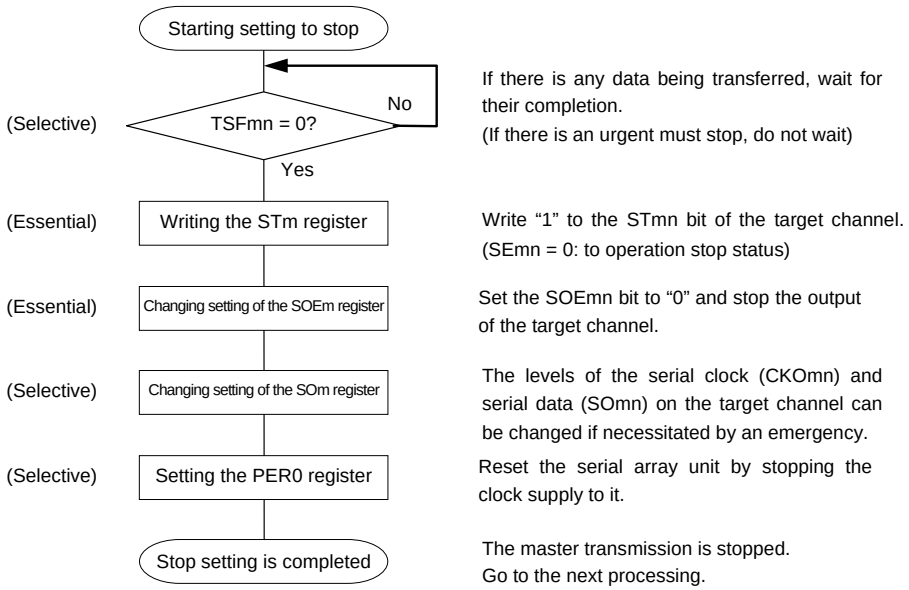
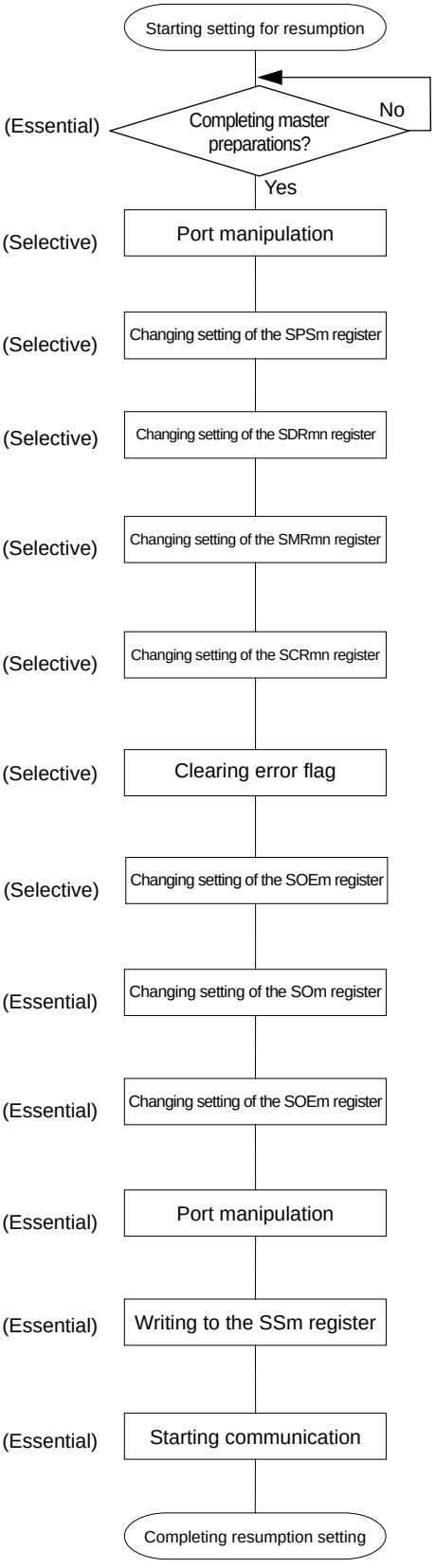


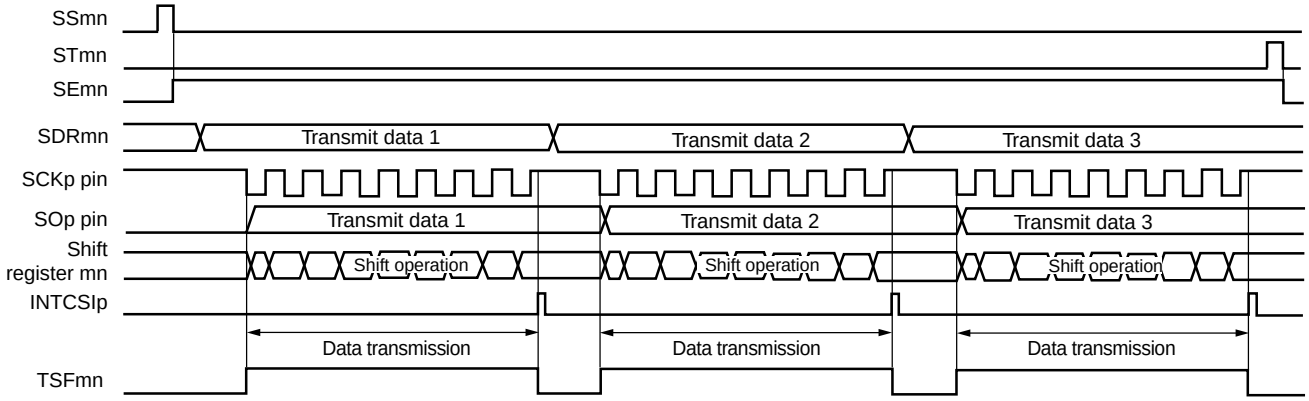
**Figure 18-52. Procedure for Resuming Slave Transmission**




**Remark** If PER0 is rewritten while stopping the master transmission and the clock supply is stopped, wait until the transmission target (master) stops or transmission finishes, and then perform initialization instead of restarting the transmission.

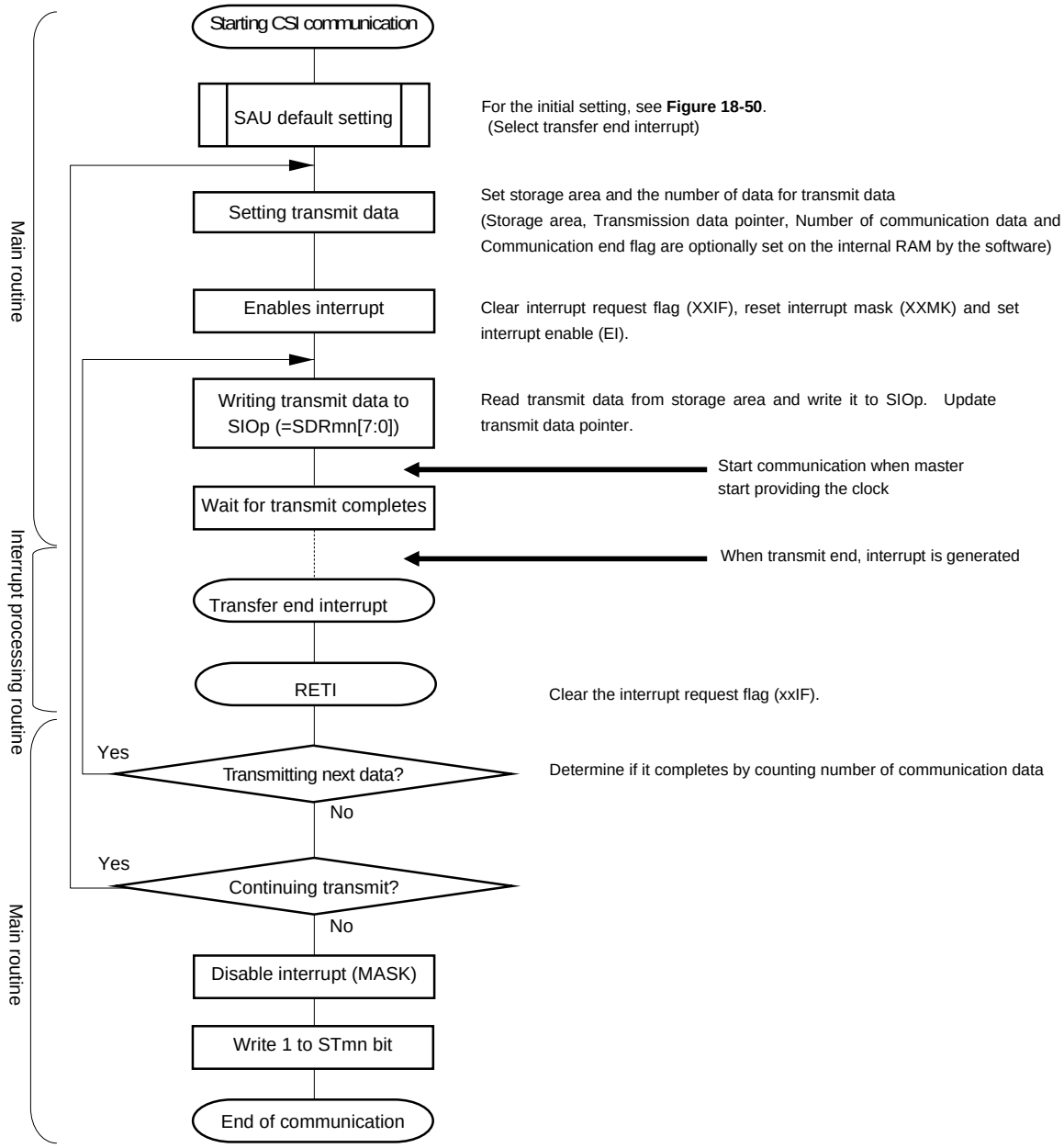
### (3) Processing flow (in single-transmission mode)

**Figure 18-53. Timing Chart of Slave Transmission (in Single-Transmission Mode)**
**(Type 1: DAPmn = 0, CKPmn = 0)**



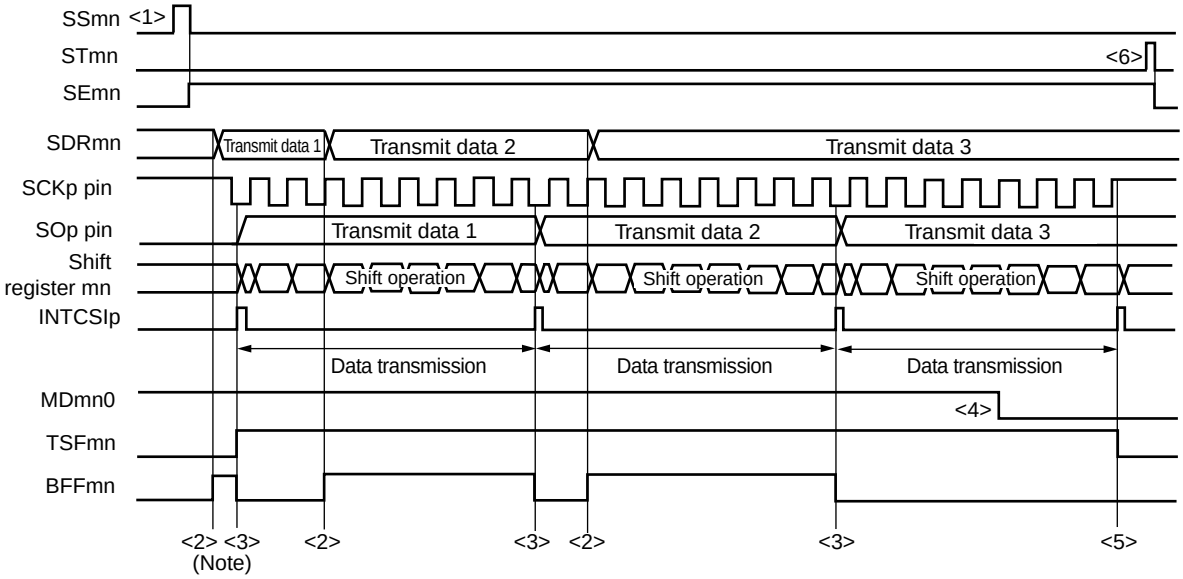

**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

**Figure 18-54. Flowchart of Slave Transmission (in Single-Transmission Mode)**



### (4) Processing flow (in continuous transmission mode)

**Figure 18-55. Timing Chart of Slave Transmission (in Continuous Transmission Mode)**
**(Type 1: DAPmn = 0, CKPmn = 0)**

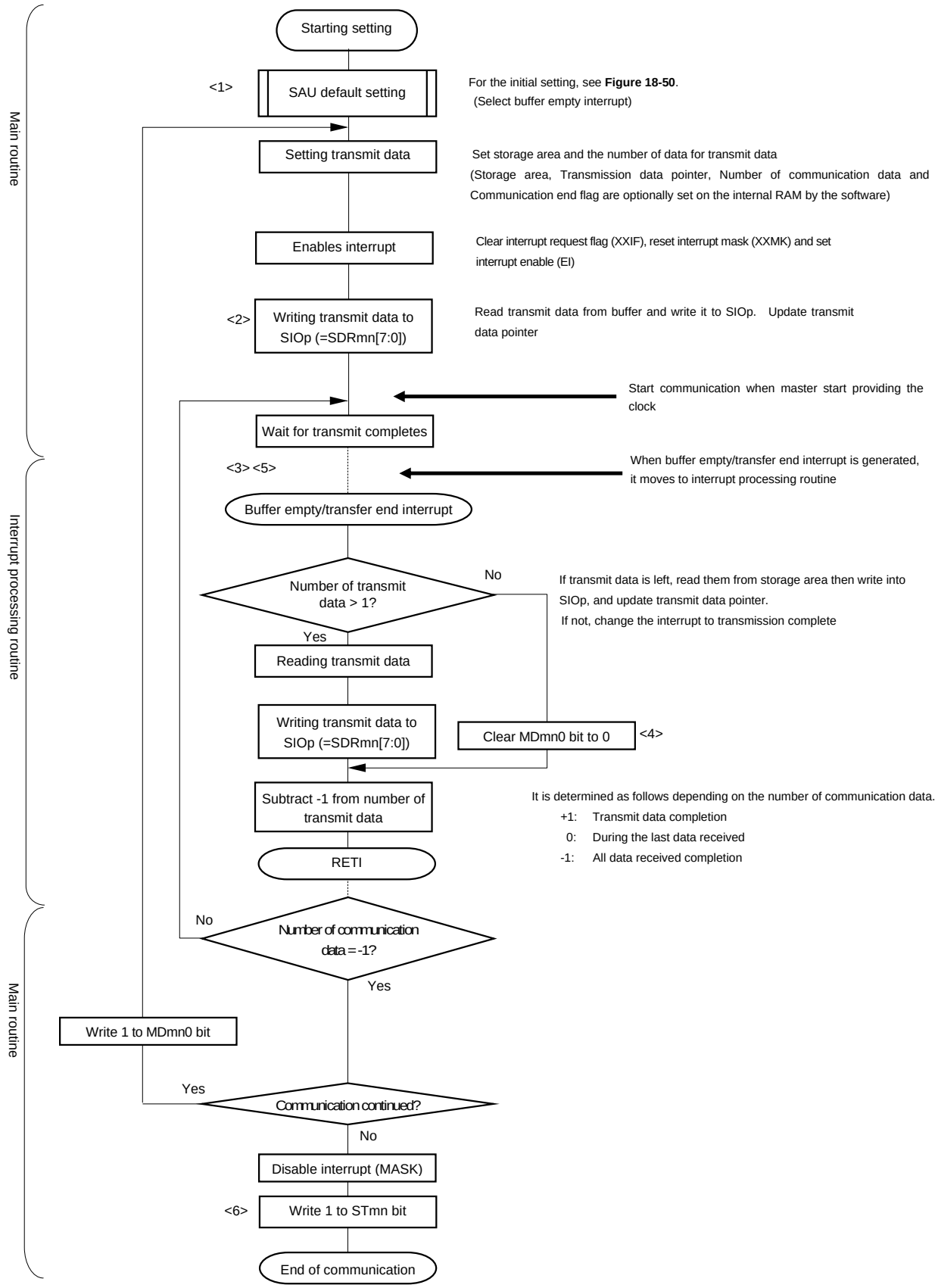



**Note** If transmit data is written to the SDRmn register while the BFFmn bit of serial status register mn (SSRmn) is 1 (valid data is stored in serial data register mn (SDRmn)), the transmit data is overwritten.

**Caution The MDmn0 bit of serial mode register mn (SMRmn) can be rewritten even during operation. However, rewrite it before transfer of the last bit is started.**

**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

**Figure 18-56. Flowchart of Slave Transmission (in Continuous Transmission Mode)**




**Remark** <1> to <6> in the figure correspond to <1> to <6> in **Figure 18-55 Timing Chart of Slave Transmission (in Continuous Transmission Mode)**.

### 18.5.5 Slave reception

Slave reception is that the RL78 microcontroller receive data from another device in the state of a transfer clock being input from another device.

| 3-Wire Serial I/O | CSI00 |
|---|---|
| Target channel | Channel 0 of SAU0 |
| Pins used | SCK00, SI00 |
| Interrupt | INTCSI00 |
| | 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<br>• DAPmn = 0: Data input starts from the start of the operation of the serial clock.<br>• DAPmn = 1: Data input starts half a clock before the start of the serial clock operation. |
| Clock phase | Selectable by the CKPmn bit of the SCRmn register<br>• CKPmn = 0: Non-reverse<br>• CKPmn = 1: Reverse |
| Data direction | MSB or LSB first |

**Notes 1.** Because the external serial clock input to the SCK00 pin is sampled internally and used, the fastest transfer rate is $f_{MCK}$/6 [Hz].

**2.** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 37 ELECTRICAL SPECIFICATIONS**).

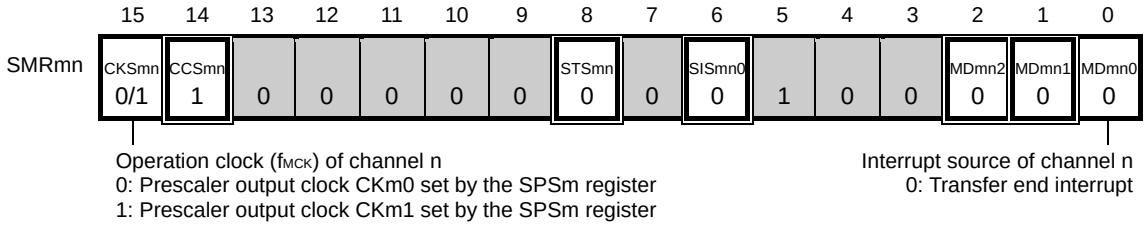
**Remarks 1.** $f_{MCK}$: Operation clock frequency of target channel

**2.** m: Unit number (m = 0), n: Channel number (n = 0), mn = 00

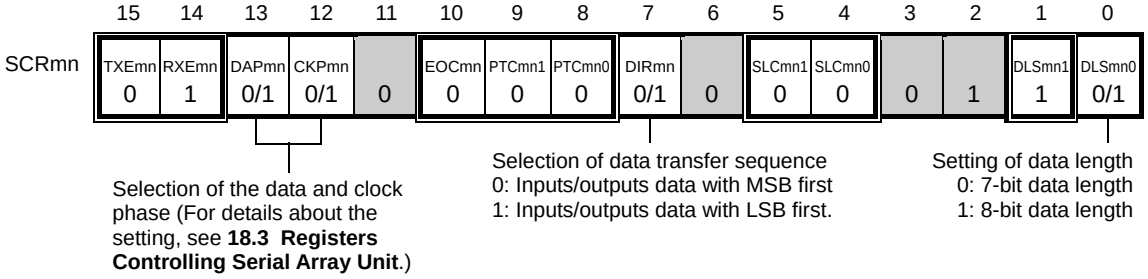
### (1) Register setting

**Figure 18-57. Example of Contents of Registers for Slave Reception of 3-Wire Serial I/O (CSI00) (1/2)**

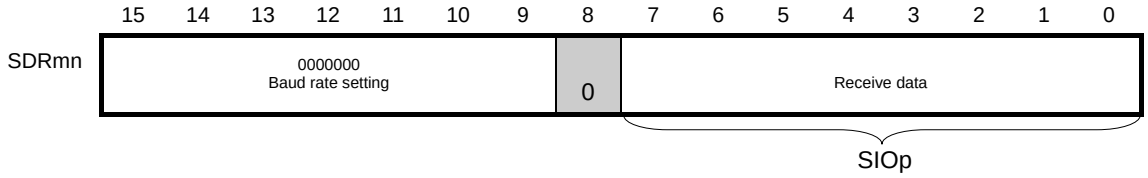
**(a) Serial mode register mn (SMRmn)**



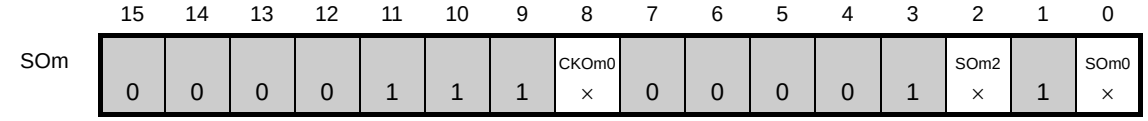

**(b) Serial communication operation setting register mn (SCRmn)**




**(c) Serial data register mn (SDRmn) (lower 8 bits: SIOp)**




**(d) Serial output register m (SOm) …The Register that not used in this mode.**




**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

**2.** ☐: Setting is fixed in the 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 usage of the user

**Figure 18-57. Example of Contents of Registers for Slave Reception of 3-Wire Serial I/O (CSI00) (2/2)**

**(e) Serial output enable register m (SOEm) …The Register that not used in this mode.**

| | 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 | 0 | SOEm2<br>× | 0 | SOEm0<br>× |

**(f) Serial channel start register m (SSm) … Sets only the bits 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<br>× | SSm2<br>× | SSm1<br>× | SSm0<br>0/1 |

**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

**2.** ☐ : 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 usage of the user

### (2) Operation procedure

**Figure 18-58. Initial Setting Procedure for Slave Reception**

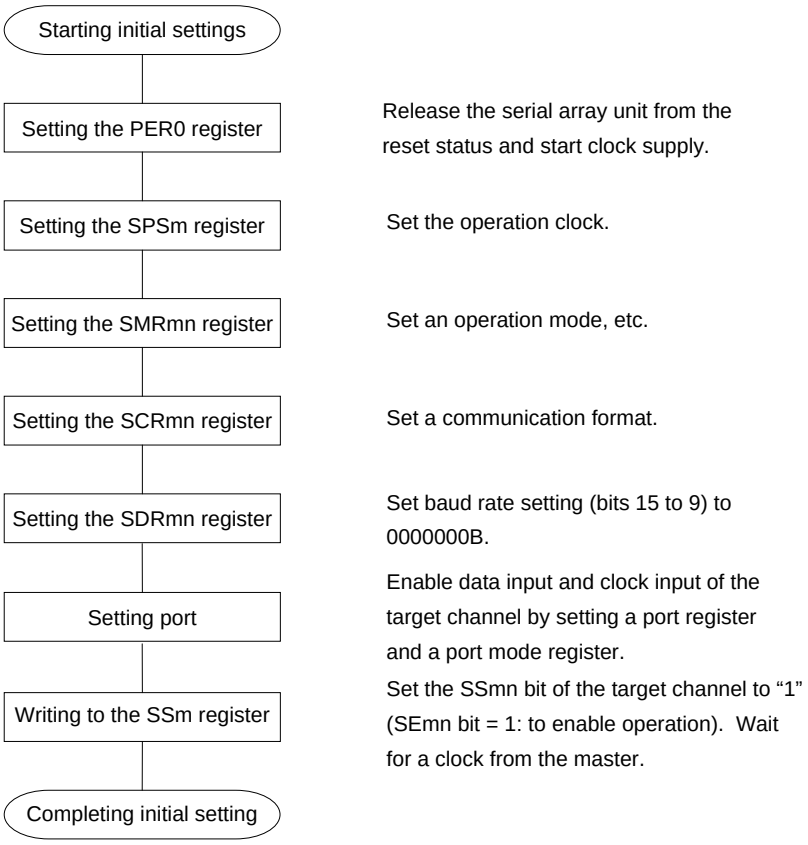



**Figure 18-59. Procedure for Stopping Slave Reception**

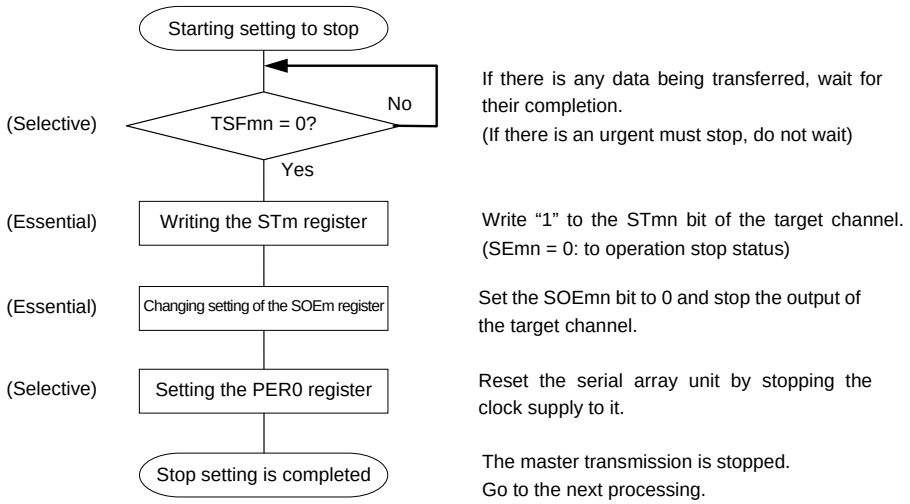
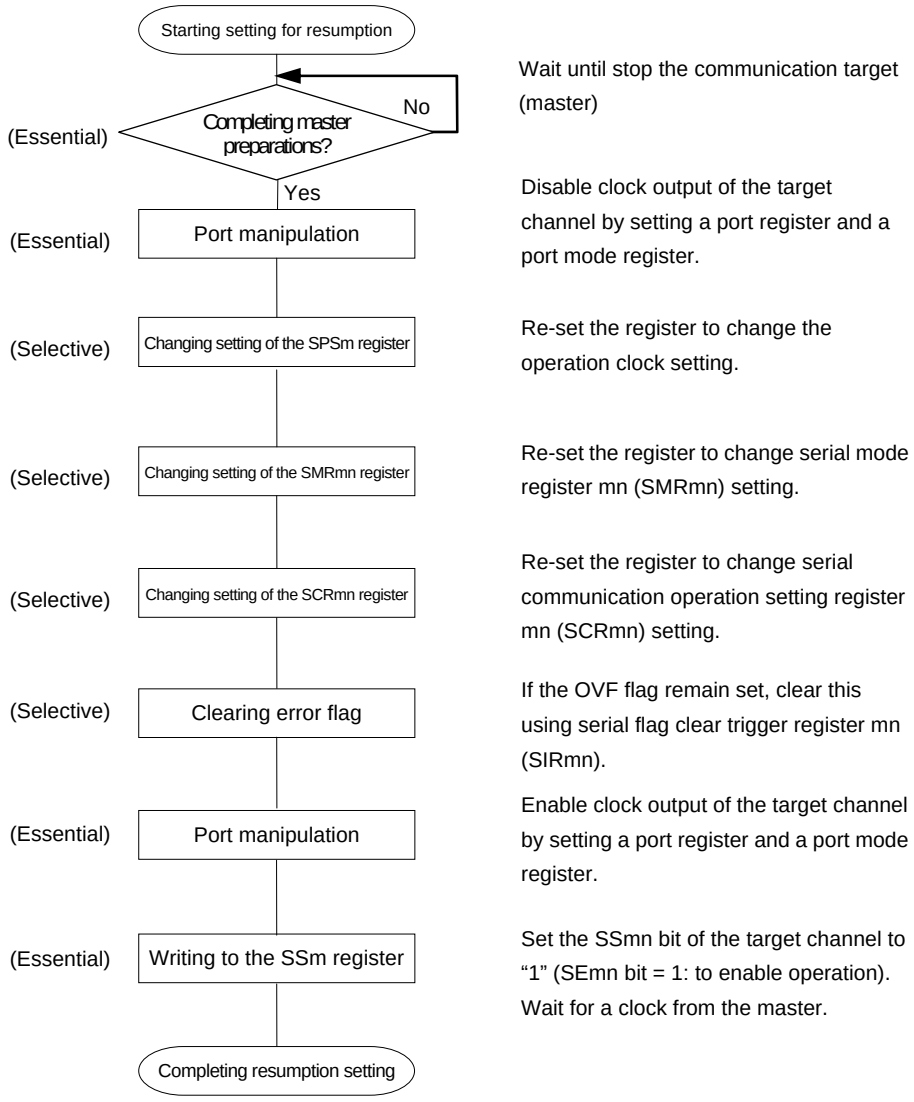


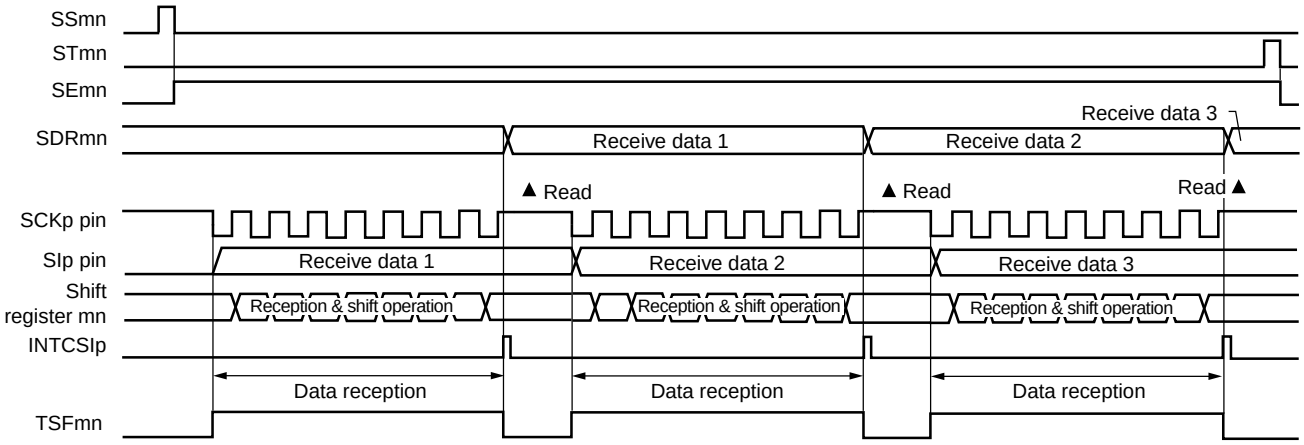
**Figure 18-60. Procedure for Resuming Slave Reception**




**Remark** If PER0 is rewritten while stopping the master transmission and the clock supply is stopped, wait until the transmission target (master) stops or transmission finishes, and then perform initialization instead of restarting the transmission.

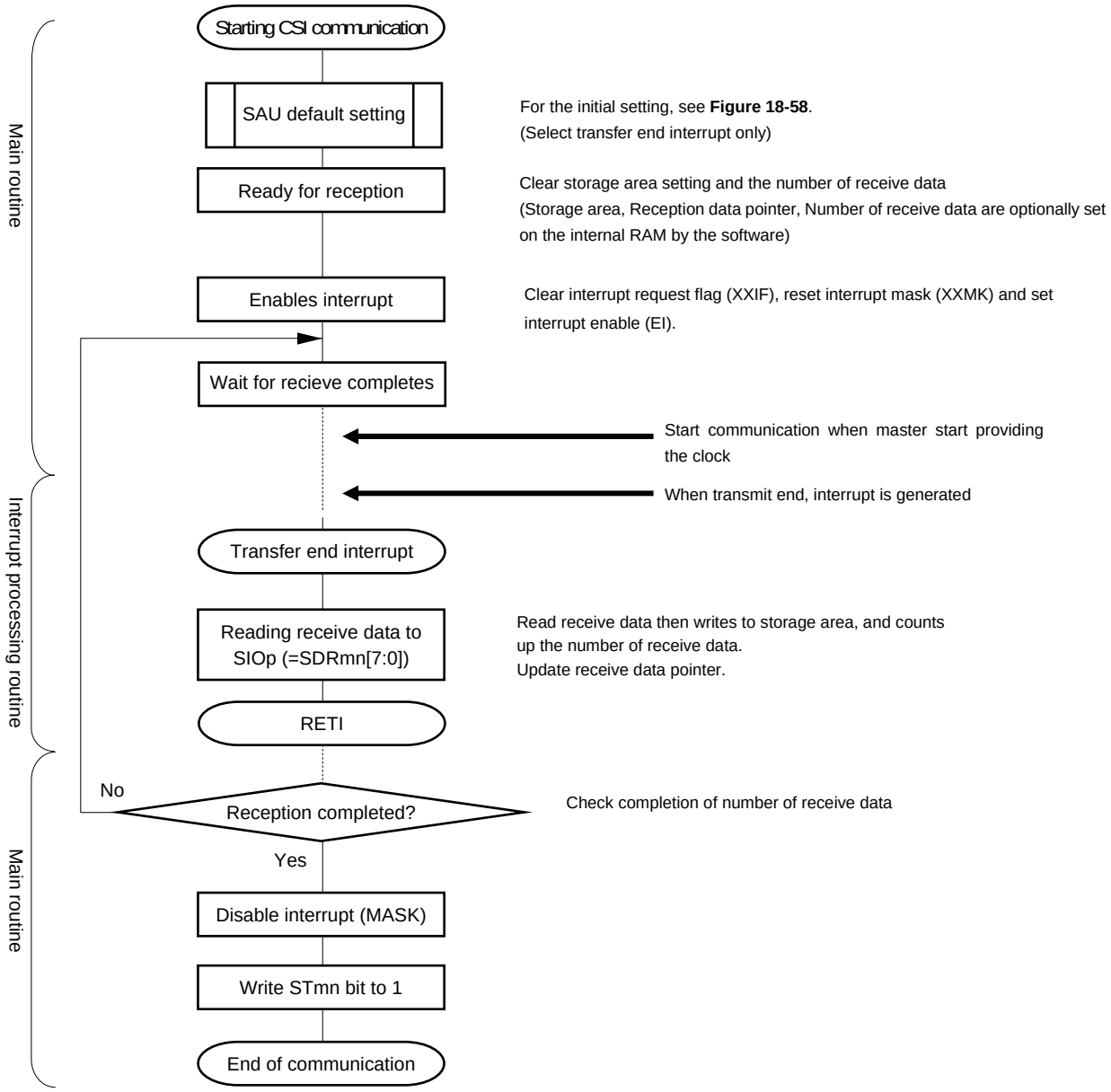
### (3) Processing flow (in single-reception mode)

**Figure 18-61. Timing Chart of Slave Reception (in Single-reception Mode)**
**(Type 1: DAPmn = 0, CKPmn = 0)**




**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

**Figure 18-62. Flowchart of Slave Reception (in Single-reception Mode)**



### 18.5.6 Slave transmission/reception

Slave transmission/reception is that the RL78 microcontroller transmit/receive data to/from another device in the state of a transfer clock being input from another device.

| 3-Wire Serial I/O | CSI00 |
|---|---|
| Target channel | Channel 0 of SAU0 |
| Pins used | SCK00, SI00, SO00 |
| Interrupt | INTCSI00 |
| | 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<br>• DAPmn = 0: Data I/O starts from the start of the operation of the serial clock.<br>• DAPmn = 1: Data I/O starts half a clock before the start of the serial clock operation. |
| Clock phase | Selectable by the CKPmn bit of the SCRmn register<br>• CKPmn = 0: Non-reverse<br>• CKPmn = 1: Reverse |
| Data direction | MSB or LSB first |

**Notes 1.** Because the external serial clock input to the SCK00 pin is sampled internally and used, the fastest transfer rate is $f_{MCK}$/6 [Hz].

**2.** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 37 ELECTRICAL SPECIFICATIONS**).

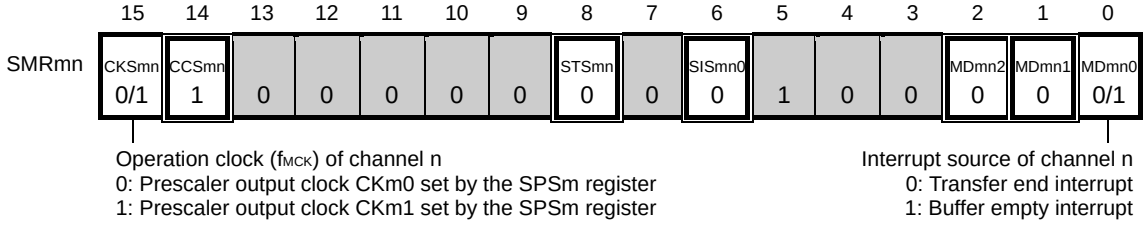
**Remarks 1.** $f_{MCK}$: Operation clock frequency of target channel

**2.** m: Unit number (m = 0), n: Channel number (n = 0), mn = 00

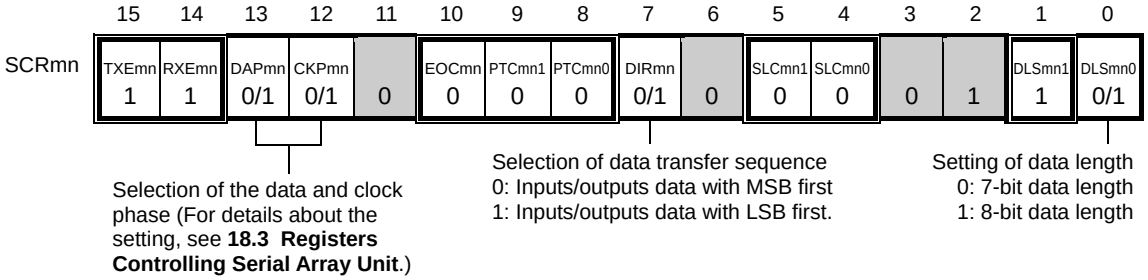
**(1) Register setting**

**Figure 18-63. Example of Contents of Registers for Slave Transmission/Reception of 3-Wire Serial I/O (CSI00) (1/2)**

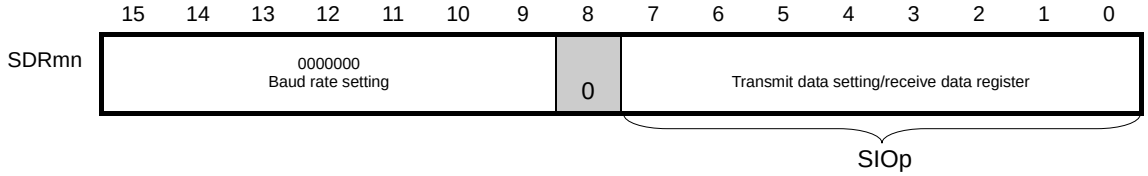
**(a) Serial mode register mn (SMRmn)**



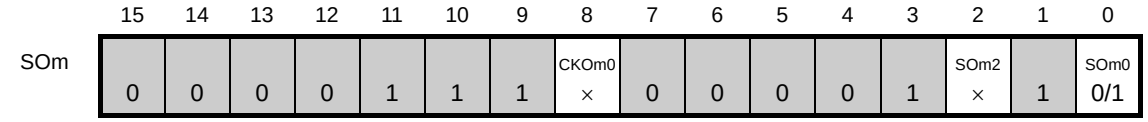

**(b) Serial communication operation setting register mn (SCRmn)**




**(c) Serial data register mn (SDRmn) (lower 8 bits: SIOp)**




**(d) Serial output register m (SOm) … Sets only the bits of the target channel.**




**Caution Be sure to set transmit data to the SIOp register before the clock from the master is started.**

**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

**2.** ☐ : Setting is fixed in the 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 usage of the user

**Figure 18-63. Example of Contents of Registers for Slave Transmission/Reception of 3-Wire Serial I/O (CSI00) (2/2)**

**(e) Serial output enable register m (SOEm) … Sets only the bits of the target channel to 1.**

| | 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 | 0 | SOEm2 × | 0 | SOEm0 0/1 |

**(f) Serial channel start register m (SSm) … Sets only the bits 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 × | SSm1 × | SSm0 0/1 |

**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

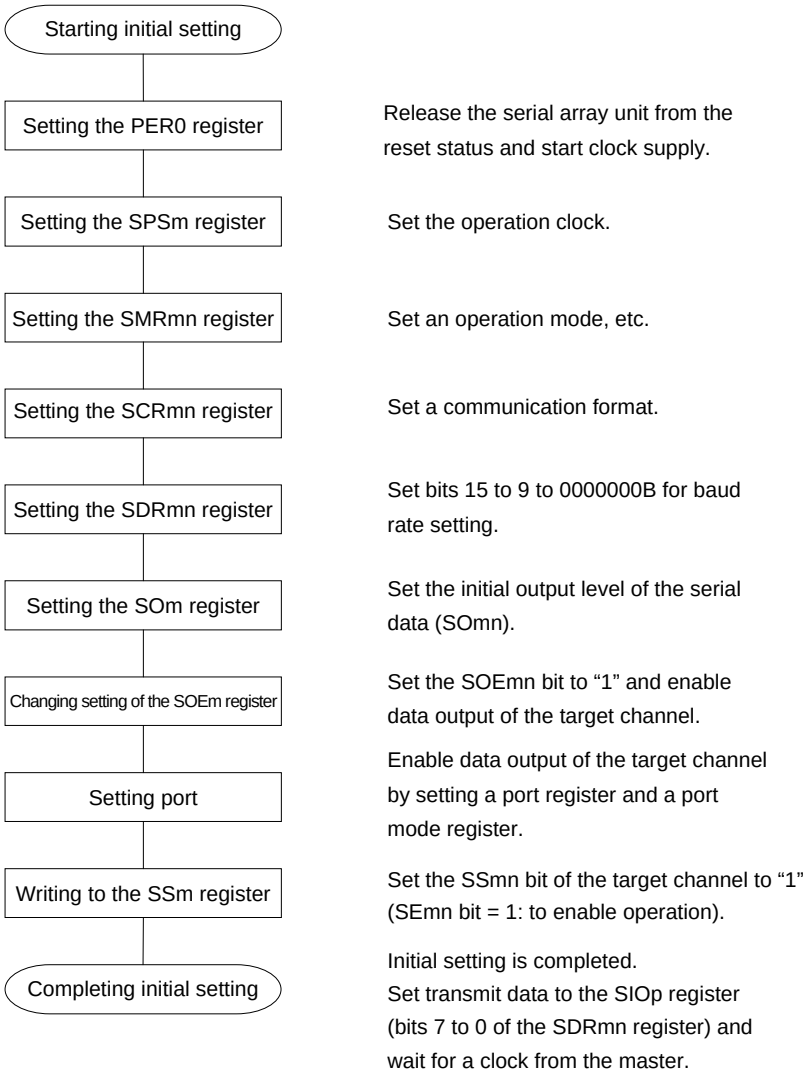
**2.** ☐ : Setting disabled (set to the initial value)

×: 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 usage of the user

### (2) Operation procedure

**Figure 18-64. 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 18-65. Procedure for Stopping Slave Transmission/Reception**

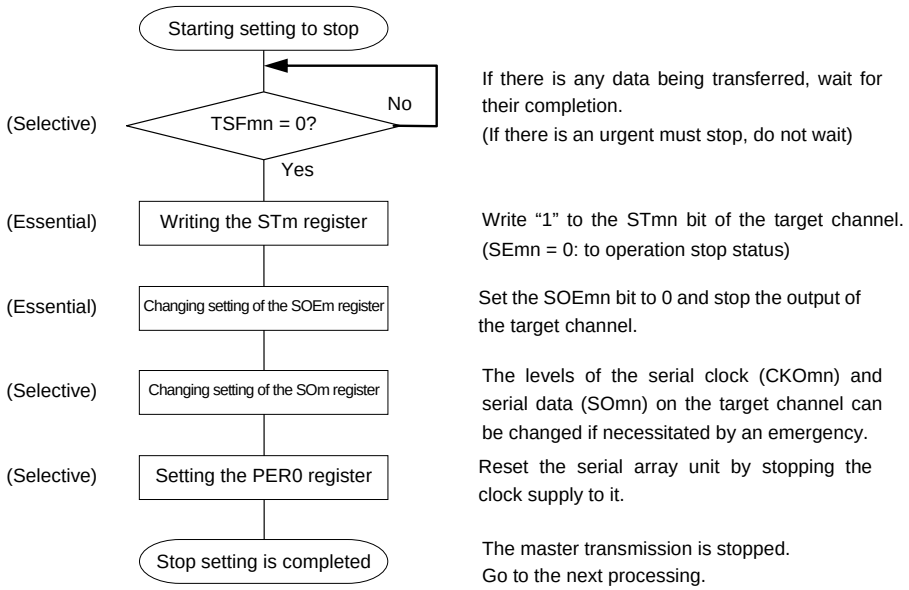
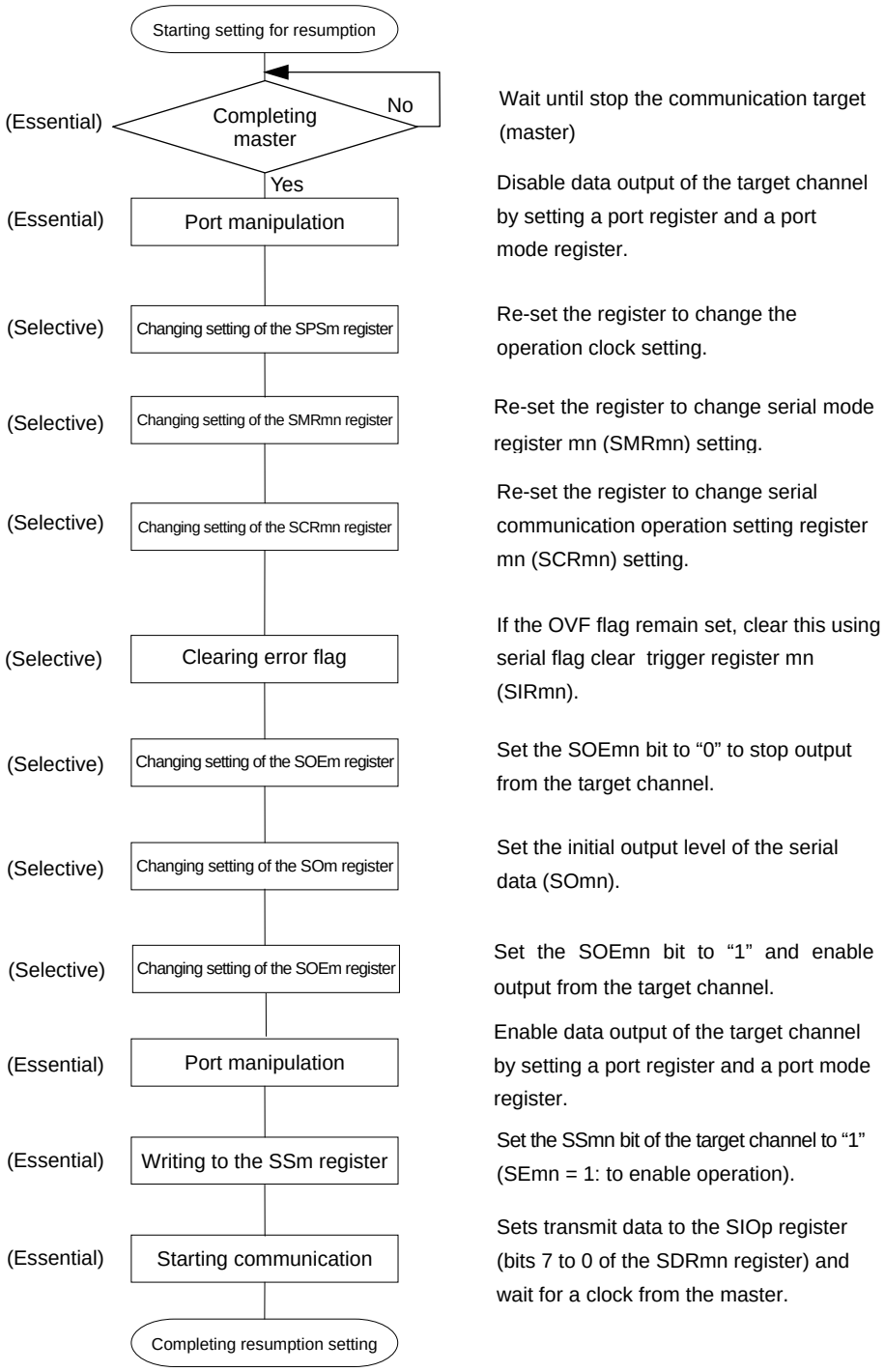


**Figure 18-66. Procedure for Resuming Slave Transmission/Reception**

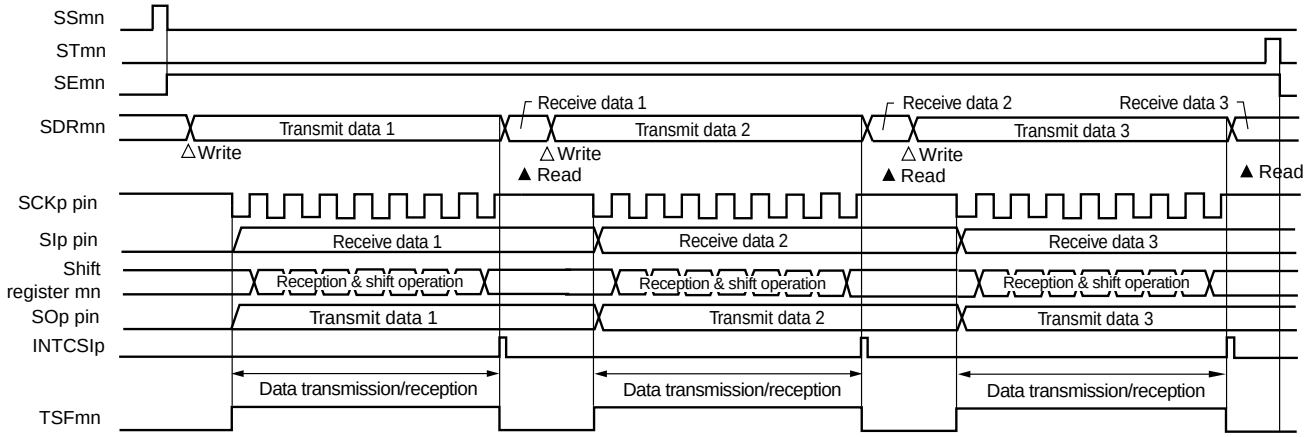



**Cautions 1. Be sure to set transmit data to the SIOp register before the clock from the master is started.**

**2. If PER0 is rewritten while stopping the master transmission and the clock supply is stopped, wait until the transmission target (master) stops or transmission finishes, and then perform initialization instead of restarting the transmission.**

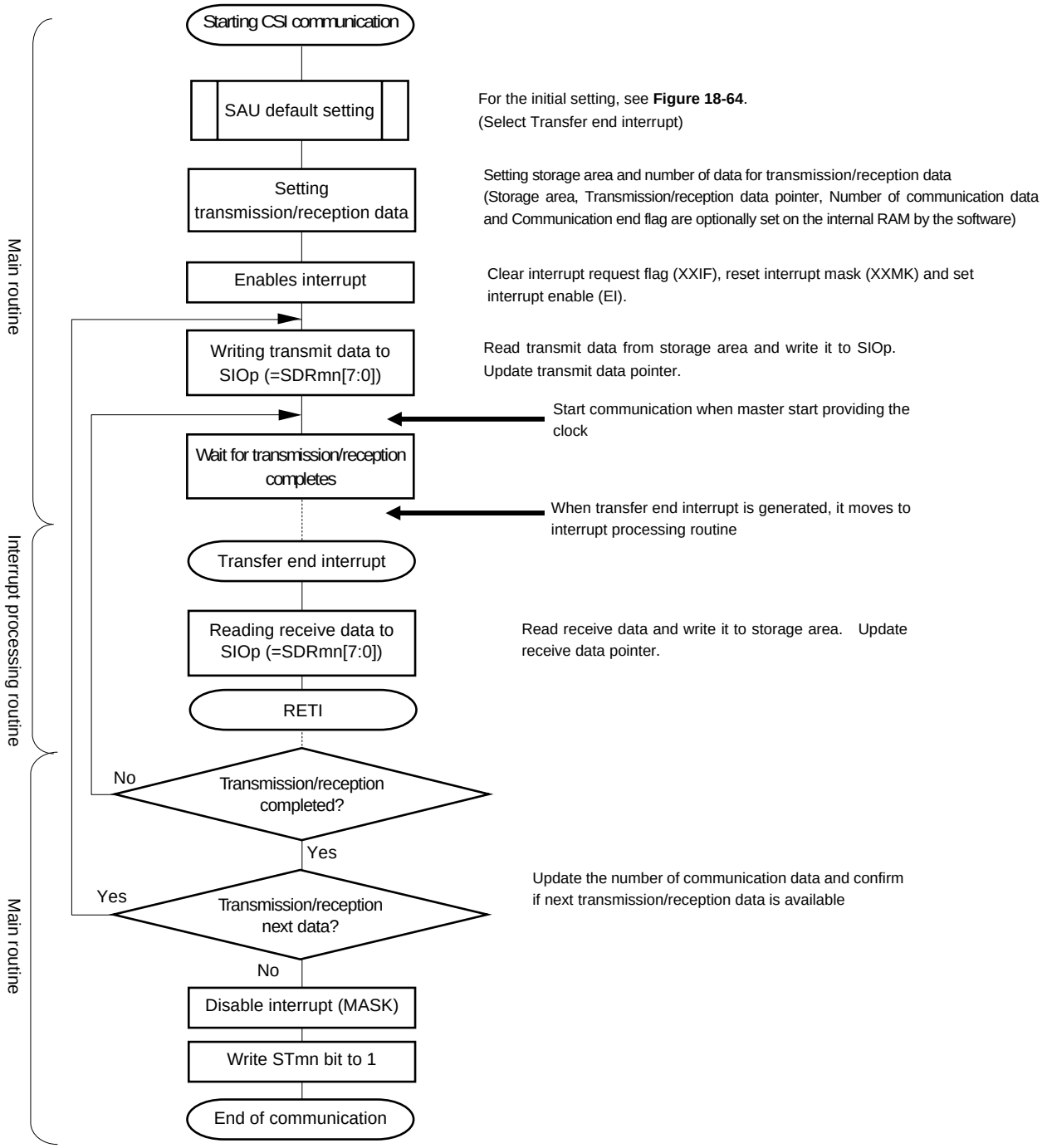
### (3) Processing flow (in single-transmission/reception mode)

**Figure 18-67. Timing Chart of Slave Transmission/Reception (in Single-Transmission/Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)**




**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

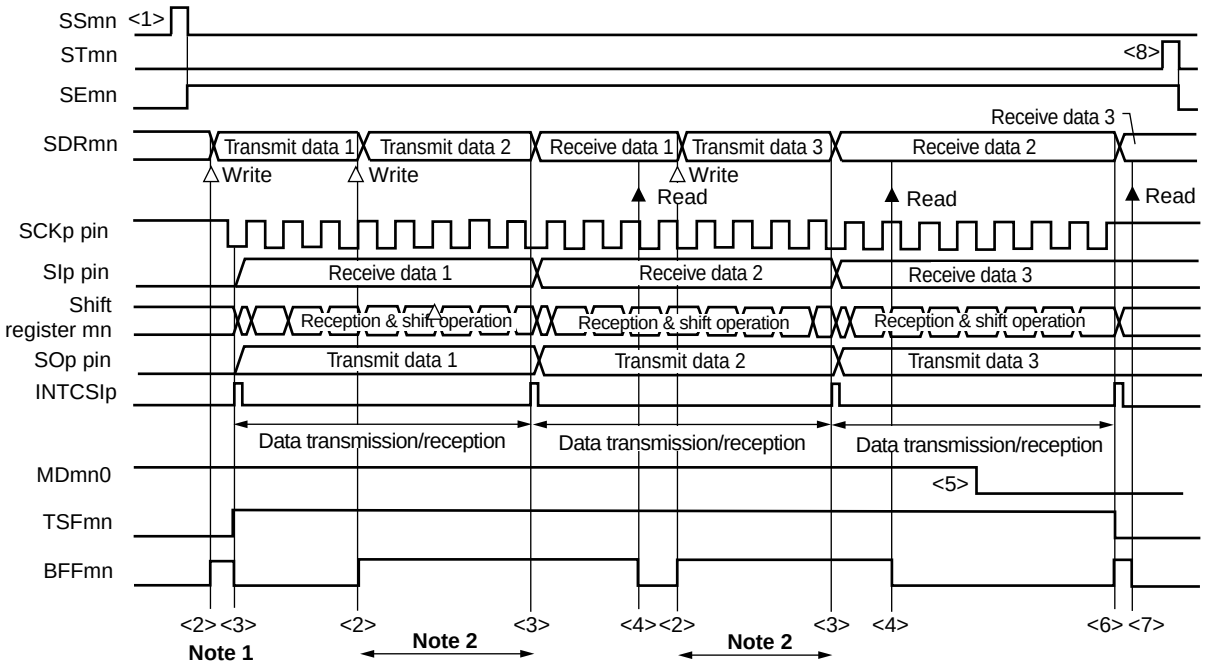
**Figure 18-68. Flowchart of Slave Transmission/Reception (in Single-Transmission/Reception Mode)**




**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 18-69. Timing Chart of Slave Transmission/Reception (in Continuous Transmission/Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)**




**Notes 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.

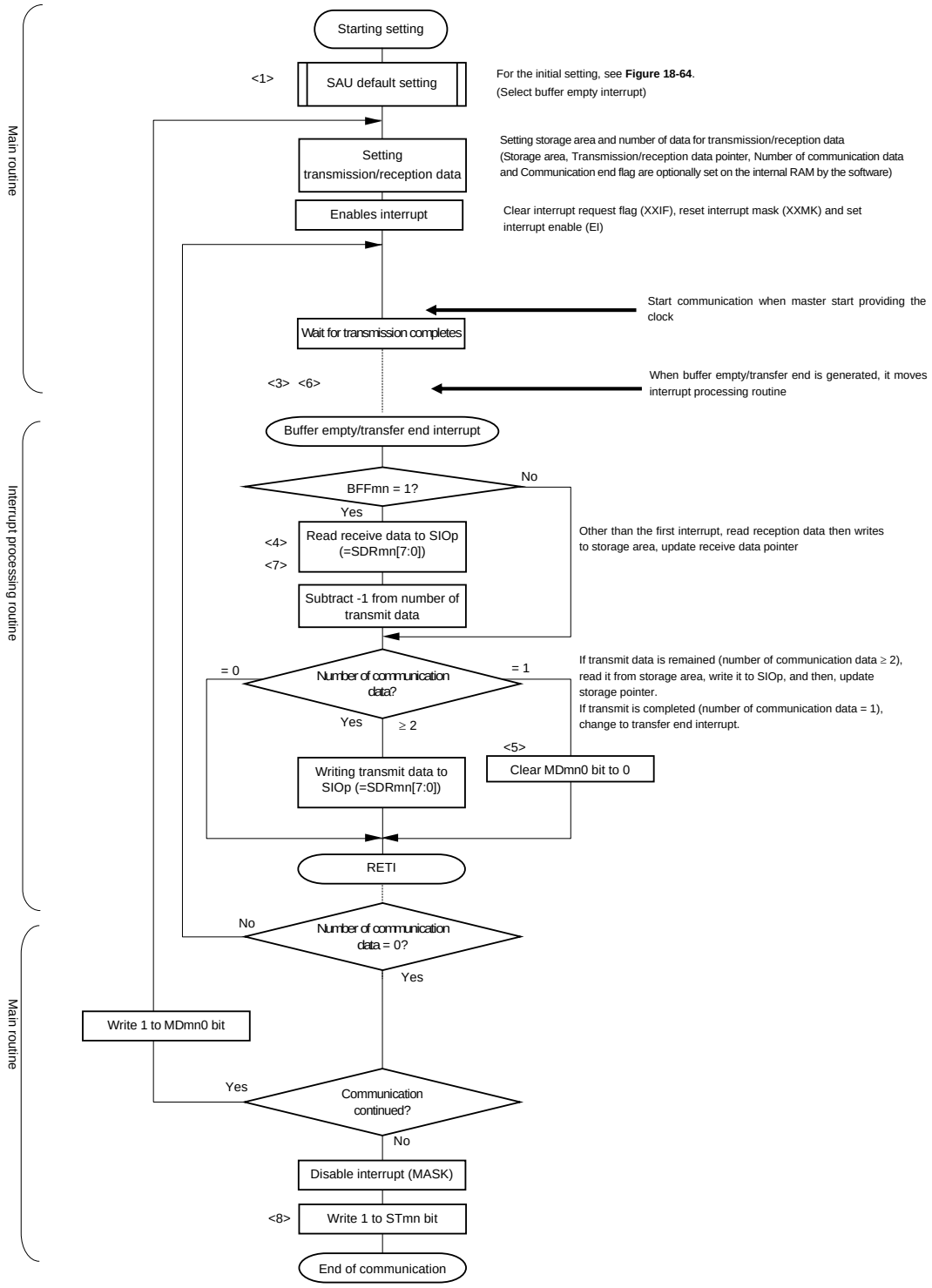
**2.** The transmit data can be read by reading the SDRmn register during this period. At this time, the transfer operation is not affected.

**Caution The MDmn0 bit of serial mode register mn (SMRmn) 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.**

**Remarks 1.** <1> to <8> in the figure correspond to <1> to <8> in **Figure 18-70 Flowchart of Slave Transmission/Reception (in Continuous Transmission/Reception Mode)**.

**2.** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

**Figure 18-70. 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 18-69 Timing Chart of Slave Transmission/Reception (in Continuous Transmission/Reception Mode)**.

### 18.5.7 SNOOZE mode function

SNOOZE mode makes CSI operate reception by SCKp pin input detection while the STOP mode. Normally CSI stops communication in the STOP mode. But, using the SNOOZE mode makes reception CSI operate unless the CPU operation by detecting SCKp pin input.

When using the CSI in SNOOZE mode, make the following setting before switching to the STOP mode (see **Figure 18-72 Flowchart of SNOOZE Mode Operation (Once Startup)** and **Figure 18-74 Flowchart of SNOOZE Mode Operation (Continuous Startup)**).

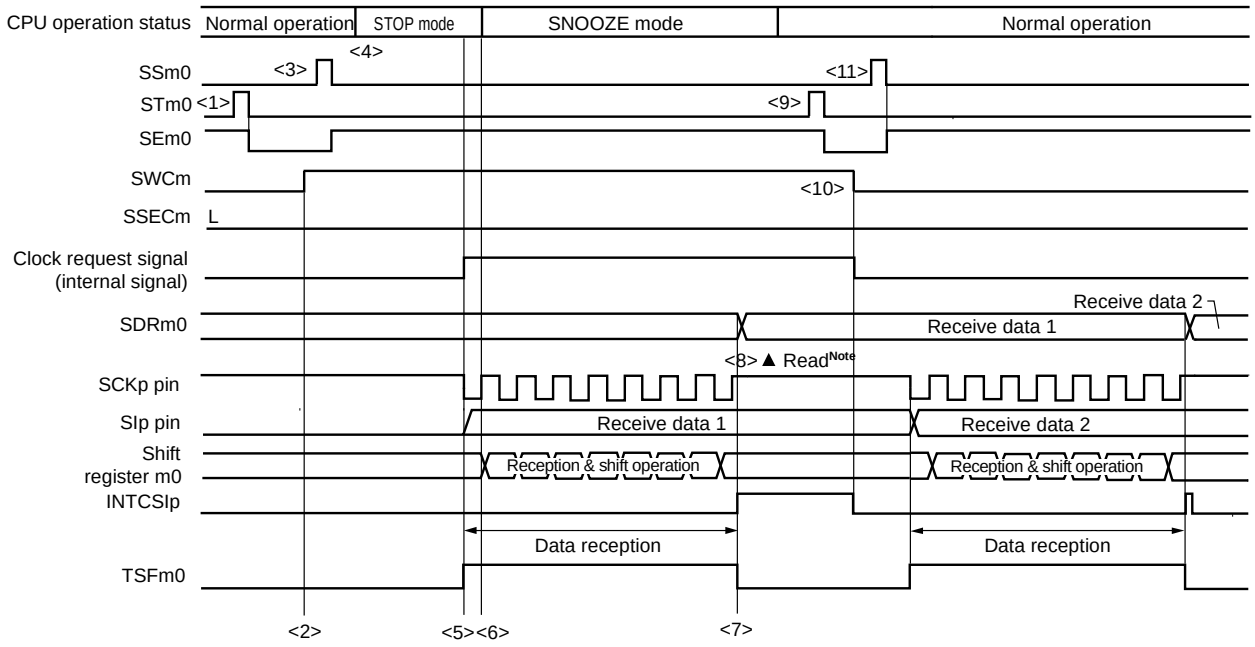
<R>
- When using the SNOOZE mode function, set the SWCm bit of serial standby control register m (SSCm) to 1 just before switching to the STOP mode. After the initial setting has been completed, set the SSm0 bit of serial channel start register m (SSm) to 1.
- The CPU shifts to the SNOOZE mode on detecting 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.

**Cautions 1. The SNOOZE mode can only be specified when the high-speed on-chip oscillator clock is selected for $f_{CLK}$.**

**2. The maximum transfer rate when using CSIp in the SNOOZE mode is 1 Mbps.**

#### (1) SNOOZE mode operation (once startup)

<R> **Figure 18-71. Timing Chart of SNOOZE Mode Operation (Once Startup) (Type 1: DAPm0 = 0, CKPm0 = 0)**




<R> **Note** Only read received data while SWCm = 1 and before the next valid edge of the SCKp pin input is detected.

**Cautions 1. Before switching to the SNOOZE mode or after reception operation in the SNOOZE mode finishes, set the STm0 bit to 1 (clear the SEm0 bit, and stop the operation). And after completion the receive operation, also clearing SWCm bit to 0 (SNOOZE mode release).**

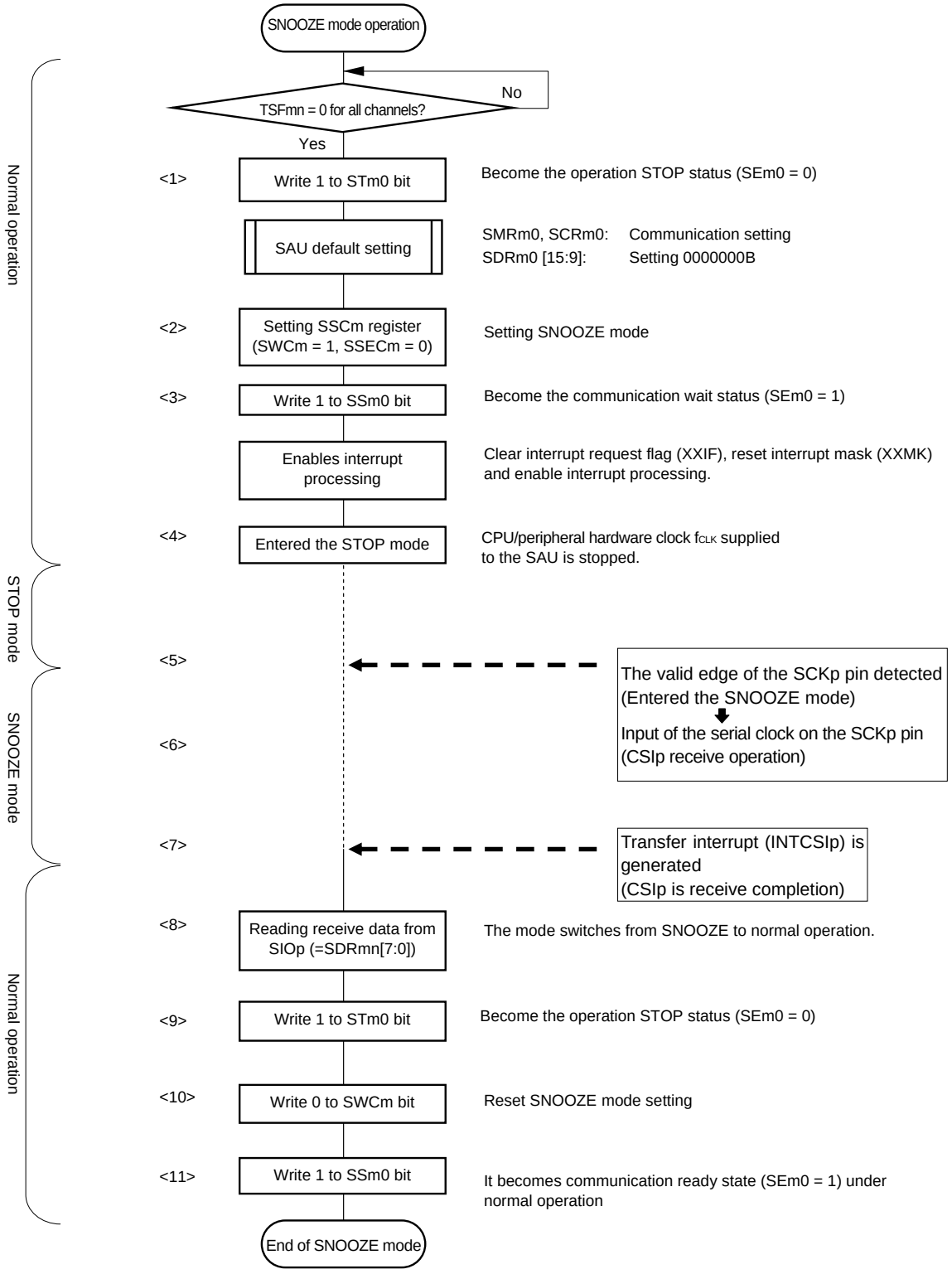
**2. When SWCm = 1, the BFFm1 and OVFm1 flags will not change.**

**Remarks 1.** <1> to <11> in the figure correspond to <1> to <11> in **Figure 18-72 Flowchart of SNOOZE Mode Operation (Once Startup)**.

**2.** m = 0; p = 00

<R>

**Figure 18-72. Flowchart of SNOOZE Mode Operation (Once Startup)**

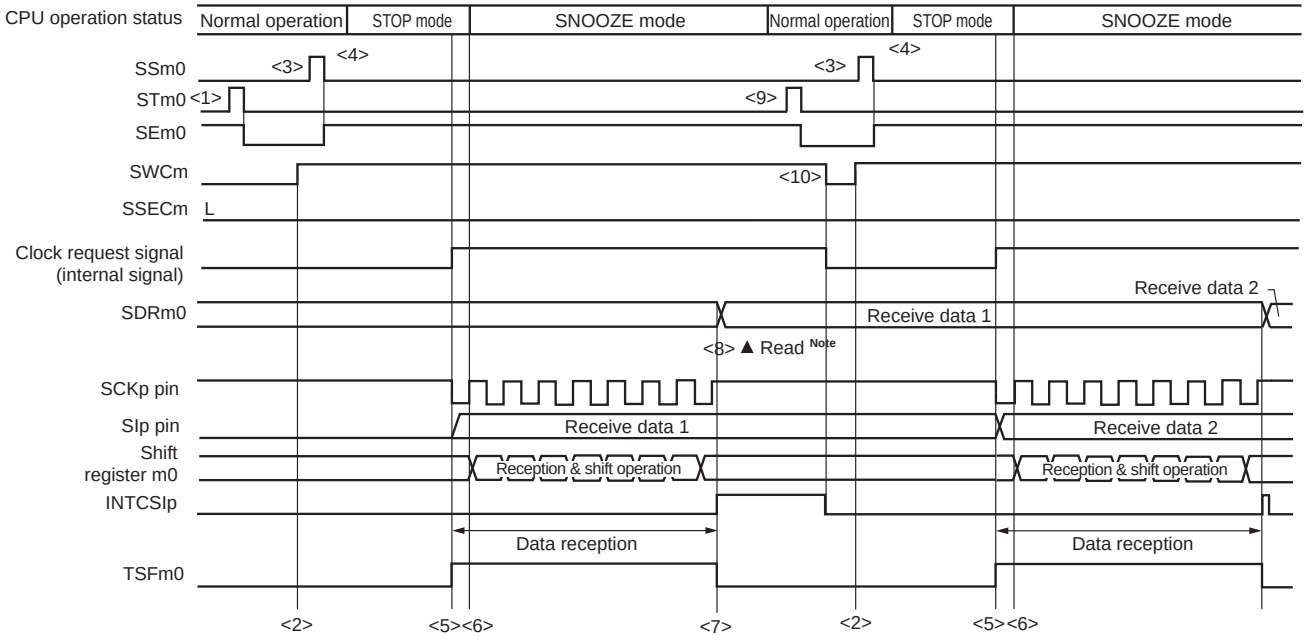



**Remarks 1.** <1> to <11> in the figure correspond to <1> to <11> in **Figure 18-71 Timing Chart of SNOOZE Mode Operation (Once Startup)**.

**2.** m = 0; p = 00

### (2) SNOOZE mode operation (continuous startup)

<R> **Figure 18-73. Timing Chart of SNOOZE Mode Operation (Continuous Startup) (Type 1: DAPm0 = 0, CKPm0 = 0)**

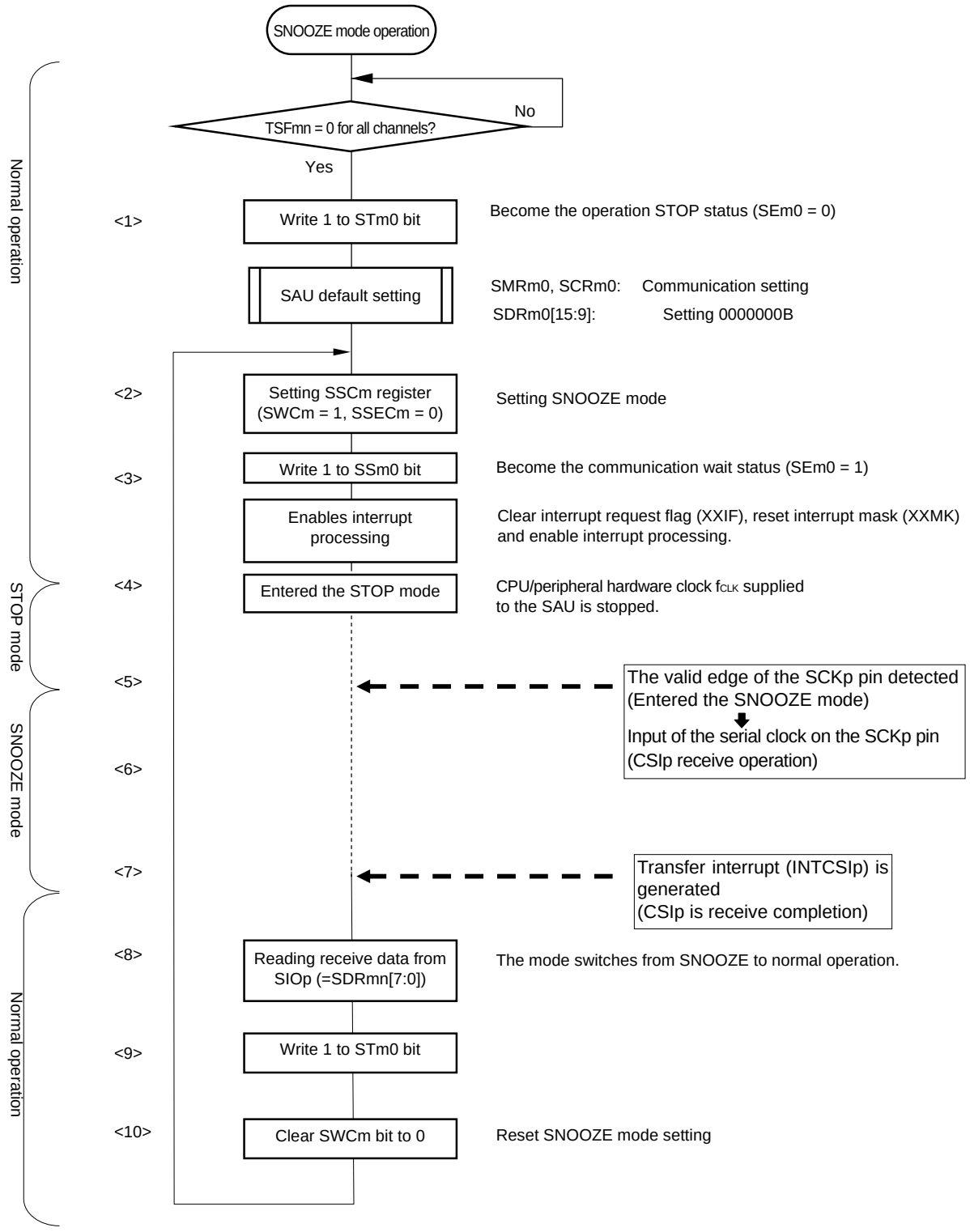



<R> **Note** Only read received data while SWCm = 1 and before the next valid edge of the SCKp pin input is detected.

**Cautions 1. Before switching to the SNOOZE mode or after reception operation in the SNOOZE mode finishes, set the STm0 bit to 1 (clear the SEm0 bit, and stop the operation).**
**And after completion the receive operation, also clearing SWCm bit to 0 (SNOOZE release).**
**2. When SWCm = 1, the BFFm1 and OVFm1 flags will not change.**

**Remarks 1.** <1> to <10> in the figure correspond to <1> to <10> in **Figure 18-74 Flowchart of SNOOZE Mode Operation (Continuous Startup)**.
**2.** m = 0; p = 00

<R> **Figure 18-74. Flowchart of SNOOZE Mode Operation (Continuous Startup)**




**Remarks 1.** <1> to <10> in the figure correspond to <1> to <10> in **Figure 18-73 Timing Chart of SNOOZE Mode Operation (Continuous Startup)**.

**2.** m = 0; p = 00

### 18.5.8 Calculating transfer clock frequency

The transfer clock frequency for 3-wire serial I/O (CSI00) 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 bit 15 (CKSmn) of serial mode register mn (SMRmn).

**Table 18-2. Selection of Operation Clock For 3-Wire Serial I/O**

| 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 | | $f_{CLK}$ = 24 MHz |
| 0 | X | X | X | X | 0 | 0 | 0 | 0 | $f_{CLK}$ | 24 MHz |
| | X | X | X | X | 0 | 0 | 0 | 1 | $f_{CLK}/2$ | 12 MHz |
| | X | X | X | X | 0 | 0 | 1 | 0 | $f_{CLK}/2^2$ | 6 MHz |
| | X | X | X | X | 0 | 0 | 1 | 1 | $f_{CLK}/2^3$ | 3 MHz |
| | X | X | X | X | 0 | 1 | 0 | 0 | $f_{CLK}/2^4$ | 1.5 MHz |
| | X | X | X | X | 0 | 1 | 0 | 1 | $f_{CLK}/2^5$ | 750 kHz |
| | X | X | X | X | 0 | 1 | 1 | 0 | $f_{CLK}/2^6$ | 375 kHz |
| | X | X | X | X | 0 | 1 | 1 | 1 | $f_{CLK}/2^7$ | 187.5 kHz |
| | X | X | X | X | 1 | 0 | 0 | 0 | $f_{CLK}/2^8$ | 93.8 kHz |
| | X | X | X | X | 1 | 0 | 0 | 1 | $f_{CLK}/2^9$ | 46.9 kHz |
| | X | X | X | X | 1 | 0 | 1 | 0 | $f_{CLK}/2^{10}$ | 23.4 kHz |
| | X | X | X | X | 1 | 0 | 1 | 1 | $f_{CLK}/2^{11}$ | 11.7 kHz |
| | X | X | X | X | 1 | 1 | 0 | 0 | $f_{CLK}/2^{12}$ | 5.86 kHz |
| | X | X | X | X | 1 | 1 | 0 | 1 | $f_{CLK}/2^{13}$ | 2.93 kHz |
| | X | X | X | X | 1 | 1 | 1 | 0 | $f_{CLK}/2^{14}$ | 1.46 kHz |
| | X | X | X | X | 1 | 1 | 1 | 1 | $f_{CLK}/2^{15}$ | 732 Hz |
| 1 | 0 | 0 | 0 | 0 | X | X | X | X | $f_{CLK}$ | 24 MHz |
| | 0 | 0 | 0 | 1 | X | X | X | X | $f_{CLK}/2$ | 12 MHz |
| | 0 | 0 | 1 | 0 | X | X | X | X | $f_{CLK}/2^2$ | 6 MHz |
| | 0 | 0 | 1 | 1 | X | X | X | X | $f_{CLK}/2^3$ | 3 MHz |
| | 0 | 1 | 0 | 0 | X | X | X | X | $f_{CLK}/2^4$ | 1.5 MHz |
| | 0 | 1 | 0 | 1 | X | X | X | X | $f_{CLK}/2^5$ | 750 kHz |
| | 0 | 1 | 1 | 0 | X | X | X | X | $f_{CLK}/2^6$ | 375 kHz |
| | 0 | 1 | 1 | 1 | X | X | X | X | $f_{CLK}/2^7$ | 187.5 kHz |
| | 1 | 0 | 0 | 0 | X | X | X | X | $f_{CLK}/2^8$ | 93.8 kHz |
| | 1 | 0 | 0 | 1 | X | X | X | X | $f_{CLK}/2^9$ | 46.9 kHz |
| | 1 | 0 | 1 | 0 | X | X | X | X | $f_{CLK}/2^{10}$ | 23.4 kHz |
| | 1 | 0 | 1 | 1 | X | X | X | X | $f_{CLK}/2^{11}$ | 11.7 kHz |
| | 1 | 1 | 0 | 0 | X | X | X | X | $f_{CLK}/2^{12}$ | 5.86 kHz |
| | 1 | 1 | 0 | 1 | X | X | X | X | $f_{CLK}/2^{13}$ | 2.93 kHz |
| | 1 | 1 | 1 | 0 | X | X | X | X | $f_{CLK}/2^{14}$ | 1.46 kHz |
| | 1 | 1 | 1 | 1 | X | X | X | X | $f_{CLK}/2^{15}$ | 732 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).

**Remarks 1.** X: Don't care

**2.** m: Unit number (m = 0), n: Channel number (n = 0), mn = 00

### 18.5.9 Procedure for processing errors that occurred during 3-wire serial I/O (CSI00) communication

The procedure for processing errors that occurred during 3-wire serial I/O (CSI00) communication is described in Figure 18-75.

**Figure 18-75. Processing Procedure in Case of Overrun Error**

| Software Manipulation | Hardware Status | 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). | | Error type is identified and the read value is used to clear error flag. |
| Writes 1 to serial flag clear trigger register mn (SIRmn). → | Error flag is cleared. | Error can be cleared only during reading, by writing the value read from the SSRmn register to the SIRmn register without modification. |

**Remark** m: Unit number (m = 0), n: Channel number (n = 0), mn = 00

## 18.6 Operation of UART (UART0 to UART2) Communication

This is a start-stop synchronization 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 UART0, timer array unit 0 (channel 7), and an external interrupt (INTP0).

[Data transmission/reception]

- Data length of 7, 8, or 9 bits**Note**
- Select the MSB/LSB first
- 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, UART0 reception supports the SNOOZE mode. When RxD pin input is detected while in the STOP mode, the SNOOZE mode makes data reception that does not require the CPU possible. Only UART0 can be specified for the reception baud rate adjustment function.

The LIN-bus is accepted in UART0 (channels 0 and 1 of unit 0).

[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 0 (channel 7)

**Note** Only UART0 can be specified for the 9-bit data length.

UART0 uses channels 0 and 1 of SAU0.
UART1 uses channels 2 and 3 of SAU0.
UART2 uses channels 0 and 1 of SAU1.

| Unit | Channel | Used as CSI | Used as UART | Used as Simplified $I^2C$ |
|---|---|---|---|---|
| 0 | 0 | CSI00 | UART0 (supporting LIN-bus) | IIC00 |
| | 1 | – | | – |
| | 2 | – | UART1 | IIC10 |
| | 3 | – | | – |
| 1 | 0 | – | UART2 (supporting IrDA) | – |
| | 1 | – | | – |

Select any function for each channel. Only the selected function is possible. If UART0 is selected for channels 0 and 1 of unit 0, for example, these channels cannot be used for CSI00. At this time, however, channel 2, 3, or other channels of the same unit can be used for a function other than UART0, such as 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 **18.6.1**.)
- UART reception (See **18.6.2**.)
- LIN transmission (UART0 only) (See **18.7.1**.)
- LIN reception (UART0 only) (See **18.7.2**.)

### 18.6.1 UART transmission

UART transmission is an operation to transmit data from the RL78 microcontroller 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 | Max. $f_{MCK}$/6 [bps] (SDRmn[15:9] = 2 or more), Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [bps][Note 2] | | |
| Data phase | Non-reverse output (default: high level)<br>Reverse output (default: low level) | | |
| Parity bit | The following selectable<br>• No parity bit<br>• Appending 0 parity<br>• Appending even parity<br>• Appending odd parity | | |
| Stop bit | The following selectable<br>• Appending 1 bit<br>• Appending 2 bits | | |
| Data direction | MSB or LSB first | | |

**Notes 1.** Only UART0 can be specified for the 9-bit data length.

**2.** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 37 ELECTRICAL SPECIFICATIONS**).

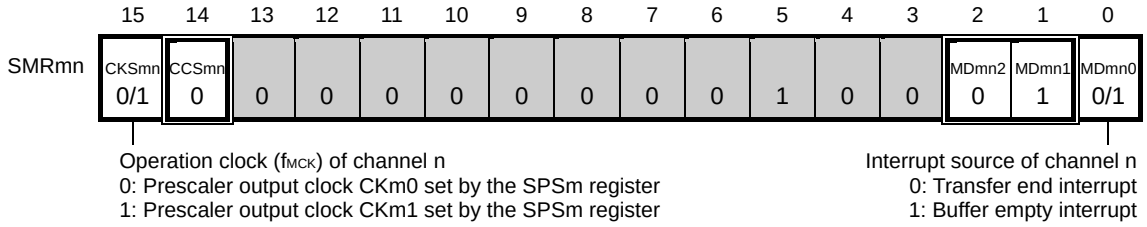
**Remarks 1.** $f_{MCK}$: Operation clock frequency of target channel
$f_{CLK}$: System clock frequency

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), mn = 00, 02, 10

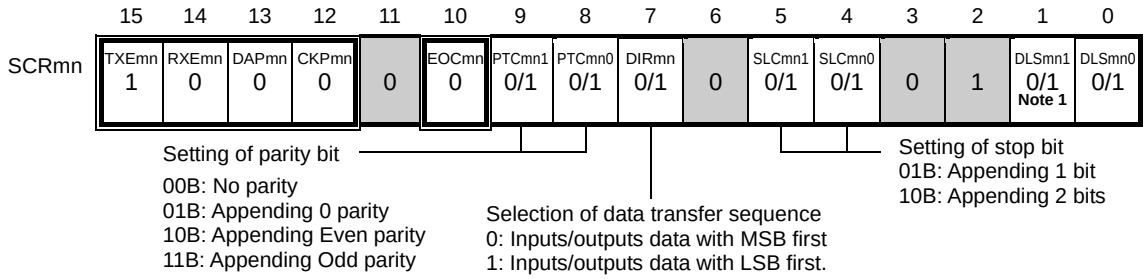
### (1) Register setting

**Figure 18-76. Example of Contents of Registers for UART Transmission of UART (UART0 to UART2) (1/2)**

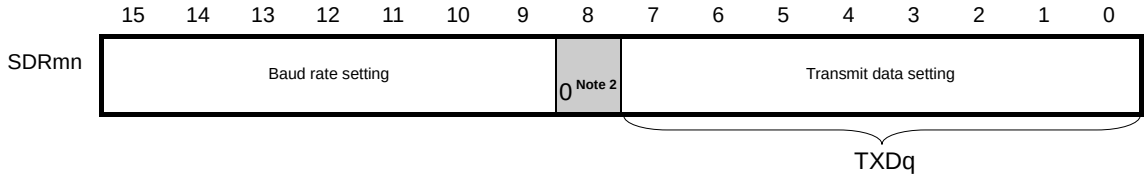
**(a) Serial mode register mn (SMRmn)**



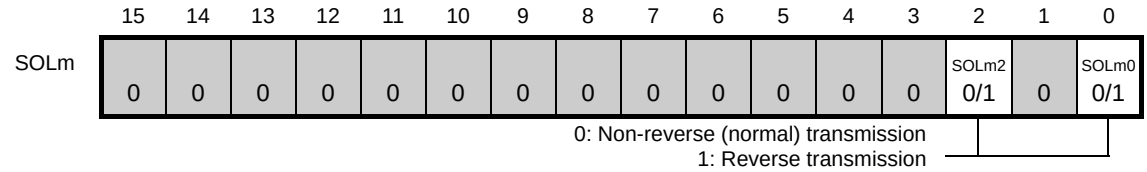

**(b) Serial communication operation setting register mn (SCRmn)**




**(c) Serial data register mn (SDRmn) (lower 8 bits: TXDq)**




**(d) Serial output level register m (SOLm) … Sets only the bits of the target channel.**




**Notes 1.** Only provided for the SCR00, SCR01, SCR10 and SCR11 registers. This bit is fixed to 1 for the other registers.

**2.** When UART0 performs 9-bit communication, bits 0 to 8 of the SDRm0 register are used as the transmission data specification area. Only UART0 can be specified for the 9-bit data length.

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), q: UART number (q = 0 to 2), mn = 00, 02, 10

**2.** ☐: Setting is fixed in the UART 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 usage of the user

**Figure 18-76. Example of Contents of Registers for UART Transmission of UART (UART0 to UART2) (2/2)**

**(e) Serial output register m (SOm) … Sets only the bits of the target channel.**

SOm

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | CKOm0 **Note 2** × | 0 | 0 | 0 | 0 | 1 | SOm2 0/1 **Notes 1, 2** | 1 | SOm0 0/1 **Note 1** |

0: Serial data output value is “0”
1: Serial data output value is “1”

**(f) Serial output enable register m (SOEm) … Sets only the bits of the target channel to 1.**

SOEm

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SOEm2 **Note 2** 0/1 | 0 | SOEm0 0/1 |

**(g) Serial channel start register m (SSm) … Sets only the bits of the target channel to 1.**

SSm

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SSm3 **Note 2** × | SSm2 **Note 2** 0/1 | SSm1 × | SSm0 0/1 |

**Notes 1.** Before transmission is started, be sure to set to 1 when the SOLmn bit of the target channel is set to 0, and set 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.

**2.** Serial array unit 0 only.

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), q: UART number (q = 0 to 2)
mn = 00, 02, 10

**2.** ☐ : 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 usage of the user

### (2) Operation procedure

**Figure 18-77. Initial Setting Procedure for UART Transmission**

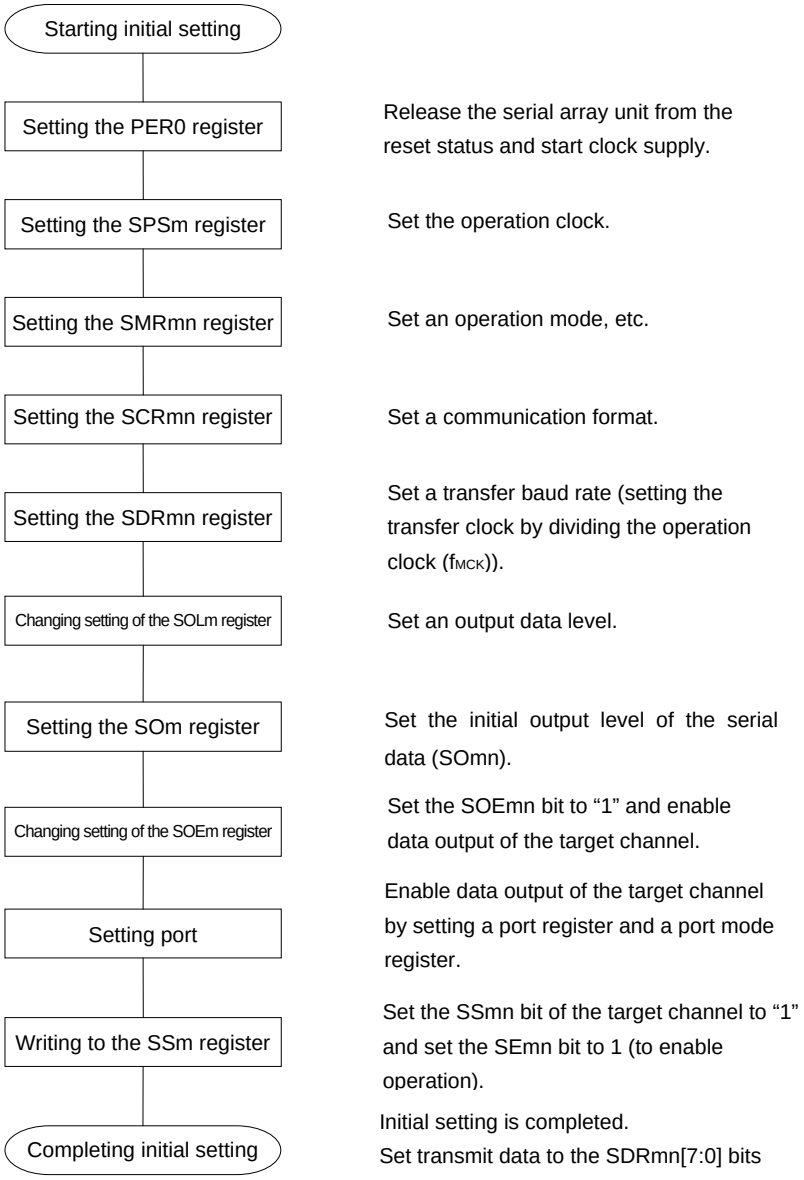


**Figure 18-78. Procedure for Stopping UART Transmission**

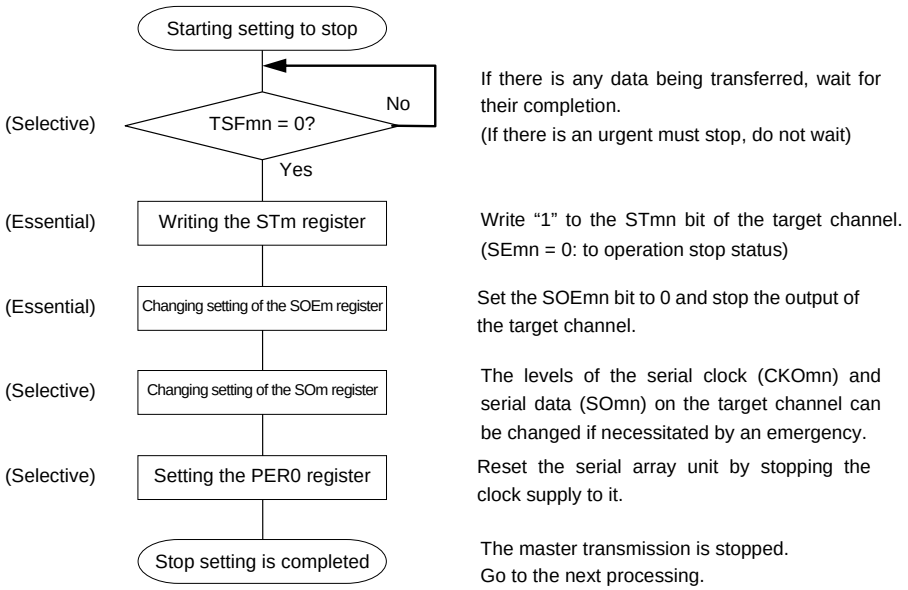
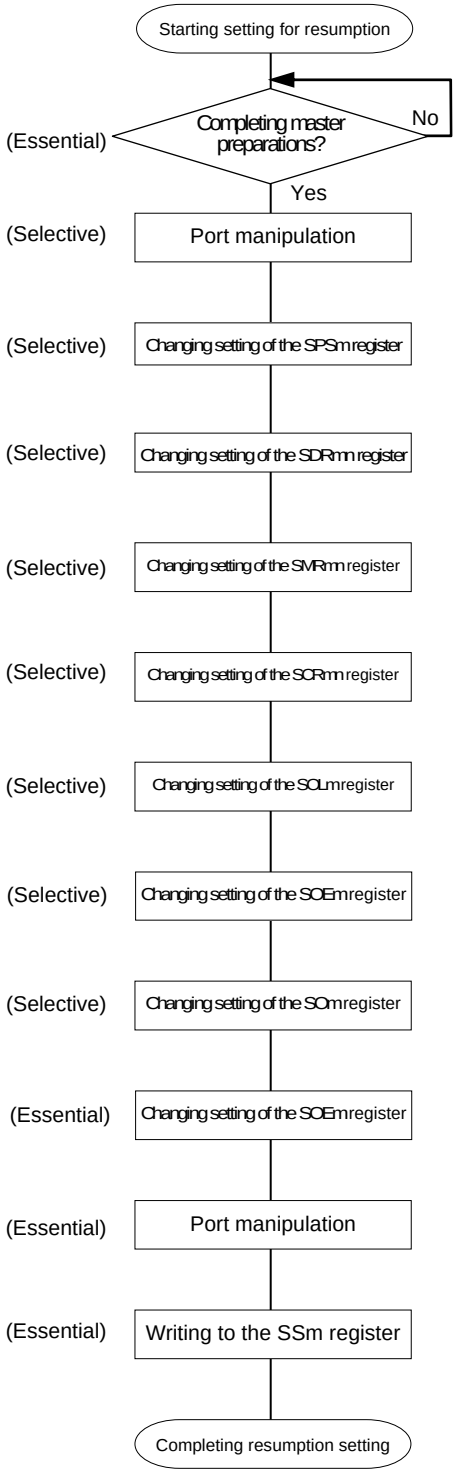


**Figure 18-79. Procedure for Resuming UART Transmission**




Wait until stop the communication target or communication operation completed

Disable data output of the target channel by setting a port register and a port mode register.

Re-set the register to change the operation clock setting.

Re-set the register to change the transfer baud rate setting (setting the transfer clock by dividing the operation clock ($f_{MCK}$)).

Re-set the register to change serial mode register mn (SMRmn) setting.

Re-set the register to change the serial communication operation setting register mn (SCRmn) setting.

Re-set the register to change serial output level register m (SOLm) setting.

Clear the SOEmn bit to “0” and stop output.

Set the initial output level of the serial data (SOmn).

Set the SOEmn bit to “1” and enable output.

Enable data output of the target channel by setting a port register and a port mode register.

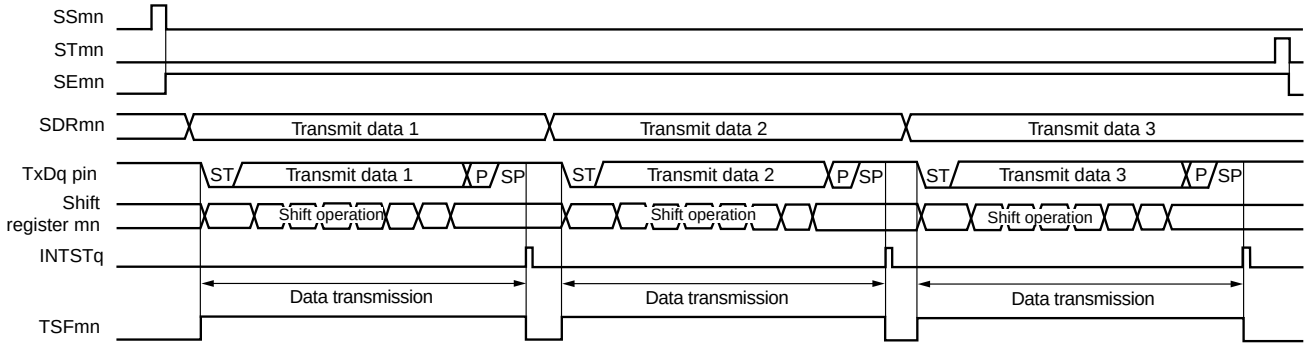
Set the SSmn bit of the target channel to “1” and set the SEmn bit to “1” (to enable operation).

Setting is completed.
Set transmit data to the SDRmn[7:0] bits (TXDq register) (8 bits) or the SDRmn[8:0] bits (9 bits) and start communication.

**Remark** If PER0 is rewritten while stopping the master transmission and the clock supply is stopped, wait until the transmission target stops or transmission finishes, and then perform initialization instead of restarting the transmission.

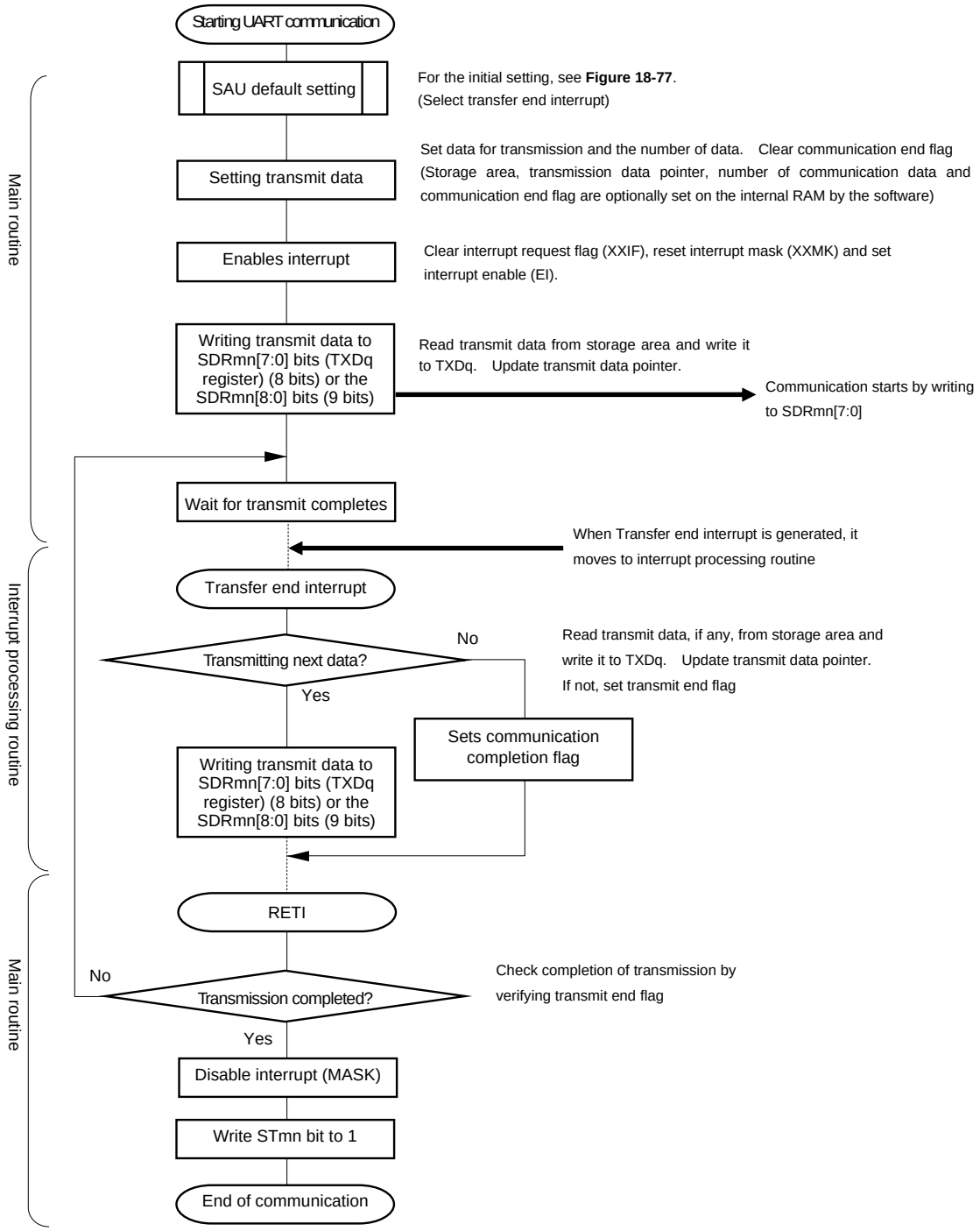
### (3) Processing flow (in single-transmission mode)

**Figure 18-80. Timing Chart of UART Transmission (in Single-Transmission Mode)**



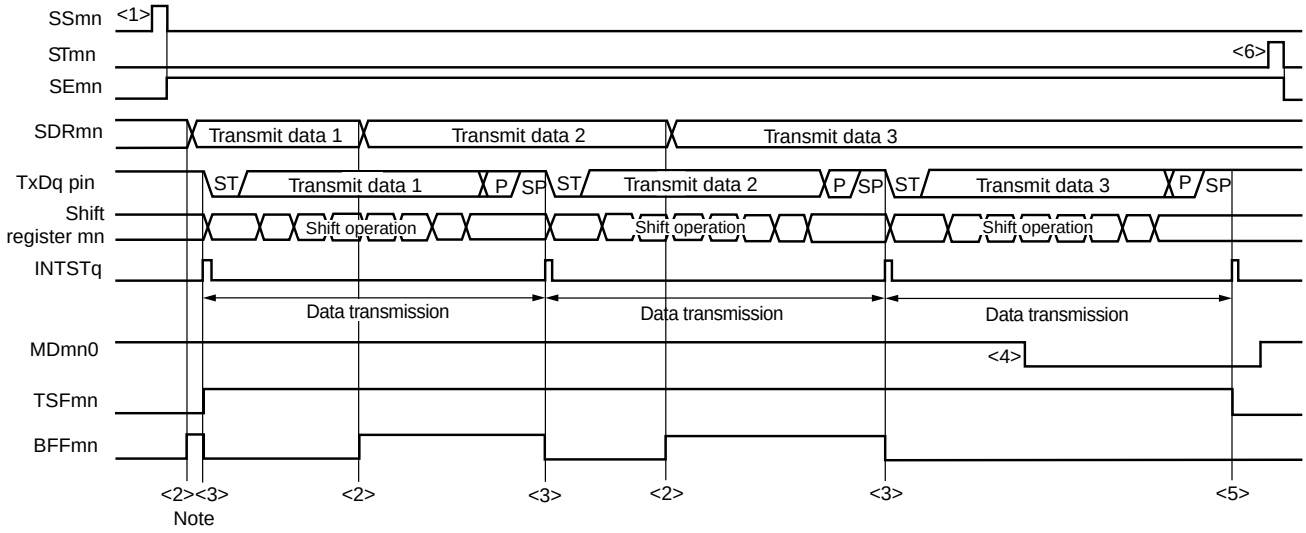

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), q: UART number (q = 0 to 2)
mn = 00, 02, 10

**Figure 18-81. Flowchart of UART Transmission (in Single-Transmission Mode)**



### (4) Processing flow (in continuous transmission mode)

**Figure 18-82. Timing Chart of UART Transmission (in Continuous Transmission Mode)**

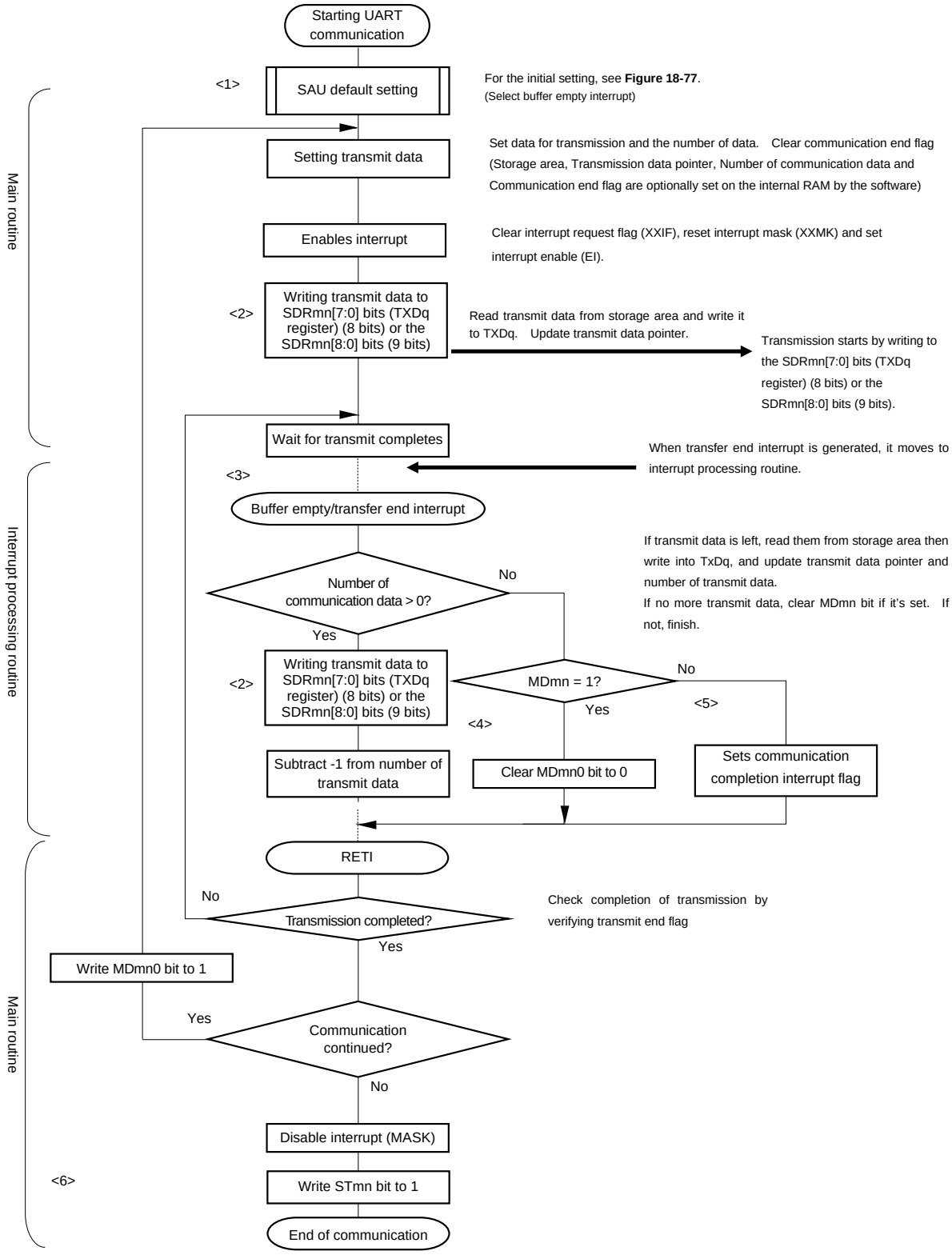



**Note** If transmit data is written to the SDRmn register while the BFFmn bit of serial status register mn (SSRmn) is 1 (valid data is stored in serial data register mn (SDRmn)), the transmit data is overwritten.

**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 will be 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 18-83. Flowchart of UART Transmission (in Continuous Transmission Mode)**




**Remark** <1> to <6> in the figure correspond to <1> to <6> in **Figure 18-82 Timing Chart of UART Transmission (in Continuous Transmission Mode)**.

### 18.6.2 UART reception

UART reception is an operation wherein the RL78 microcontroller 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)<br>• Parity error detection flag (PEFmn)<br>• Overrun error detection flag (OVFmn) | | |
| Transfer data length | 7, 8 or 9 bits Note 1 | | |
| Transfer rate | Max. $f_{MCK}/6$ [bps] (SDRmn[15:9] = 2 or more), Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [bps] Note 2 | | |
| Data phase | Non-reverse output (default: high level)<br>Reverse output (default: low level) | | |
| Parity bit | The following selectable<br>• No parity bit (no parity check)<br>• No parity judgment (0 parity)<br>• Even parity check<br>• Odd parity check | | |
| Stop bit | Appending 1 bit | | |
| Data direction | MSB or LSB first | | |

**Notes 1.** Only UART0 can be specified for the 9-bit data length.

**2.** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 37 ELECTRICAL SPECIFICATIONS**).

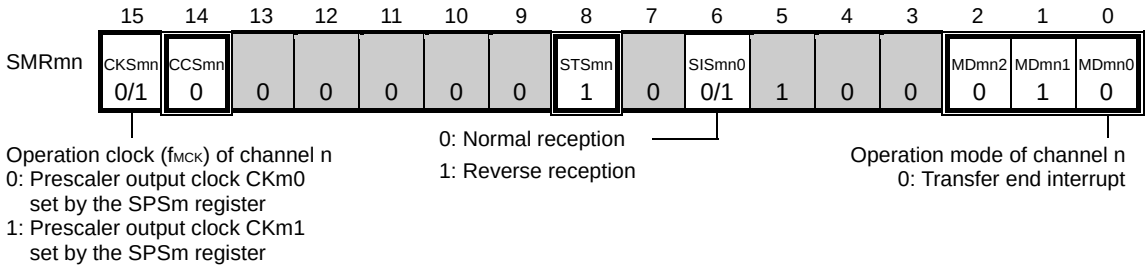
**Remarks 1.** $f_{MCK}$: Operation clock frequency of target channel
$f_{CLK}$: System clock frequency

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 1, 3), mn = 01, 03, 11

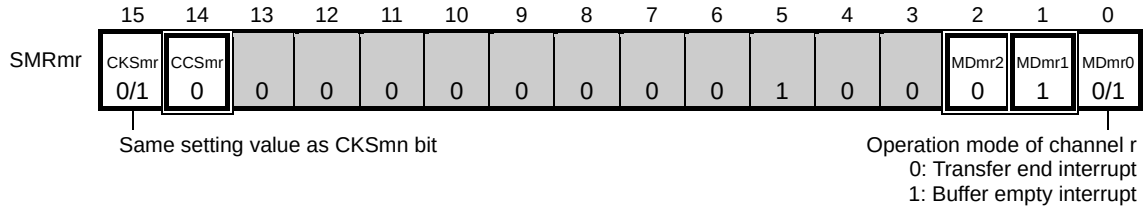
### (1) Register setting

**Figure 18-84. Example of Contents of Registers for UART Reception of UART (UART0 to UART2) (1/2)**

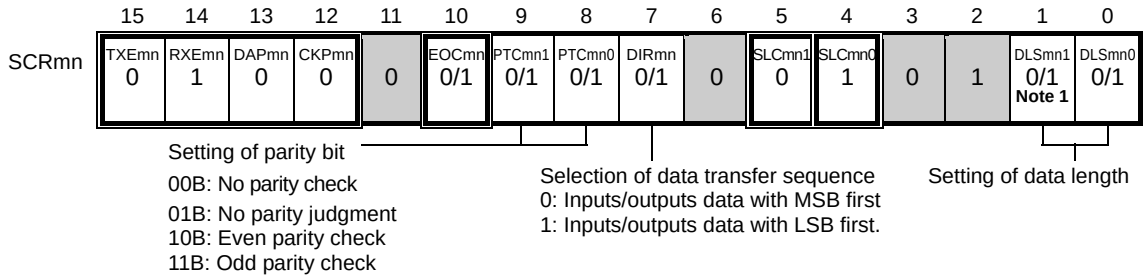
**(a) Serial mode register mn (SMRmn)**



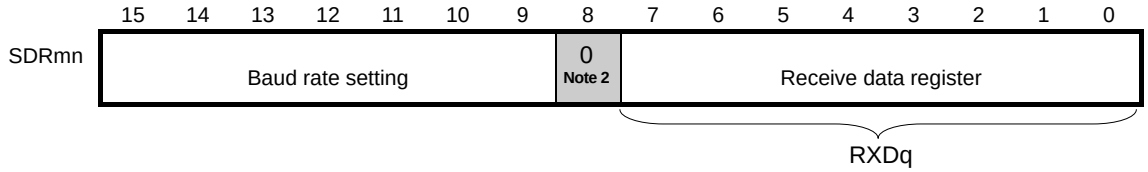

**(b) Serial mode register mr (SMRmr)**




**(c) Serial communication operation setting register mn (SCRmn)**




**(d) Serial data register mn (SDRmn) (lower 8 bits: RXDq)**




**Notes 1.** Only provided for the SCR00, SCR01, SCR10 and SCR11 registers. This bit is fixed to 1 for the other registers.

**2.** When UART performs 9-bit communication, bits 0 to 8 of the SDRm1 register are used as the transmission data specification area. Only UART0 can be specified for 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.**

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 1, 3), mn = 01, 03, 11
r: Channel number (r = n – 1), q: UART number (q = 0 to 2)

**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 usage of the user

**Figure 18-84. Example of Contents of Registers for UART Reception of UART (UART0 to UART2) (2/2)**

**(e) Serial output register m (SOm) … The register that not used in this mode.**

SOm

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | CKOm0 **Note** × | 0 | 0 | 0 | 0 | 1 | SOm2 **Note** × | 1 | SOm0 × |

**(f) Serial output enable register m (SOEm) …The register that not used in this mode.**

SOEm

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SOEm2 **Note** × | 0 | SOEm0 × |

**(g) Serial channel start register m (SSm) … Sets only the bits of the target channel is 1.**

SSm

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SSm3 **Note** 0/1 | SSm2 **Note** × | SSm1 0/1 | SSm0 × |

**Note** Serial array unit 0 only.

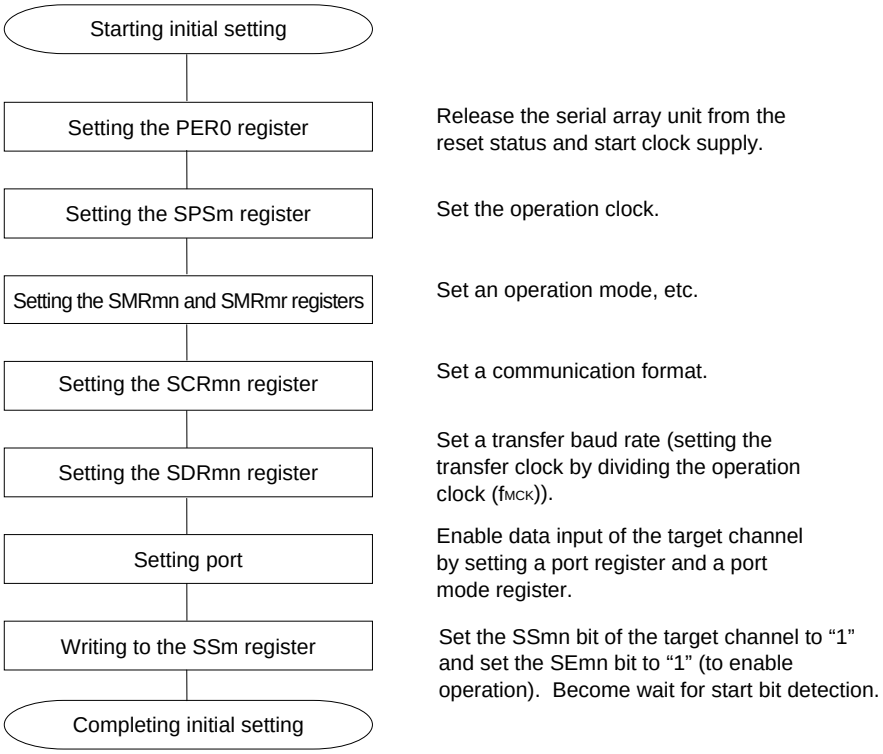
**Caution For the UART reception, be sure to set the SMRmr register of channel r to UART Transmission mode that is to be paired with channel n.**

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 1, 3), mn = 01, 03, 11
r: Channel number (r = n – 1), q: UART number (q = 0 to 2)

**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 usage of the user

### (2) Operation procedure

**Figure 18-85. 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 4 or more $f_{MCK}$ clocks have elapsed.**

**Figure 18-86. Procedure for Stopping UART Reception**

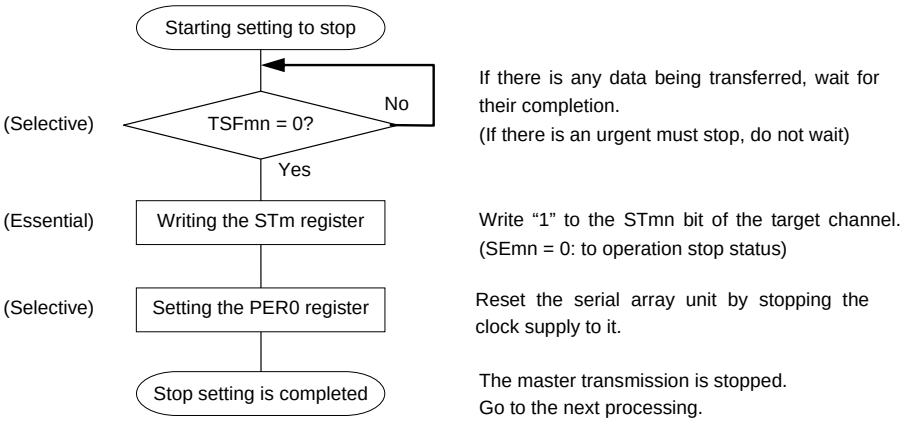
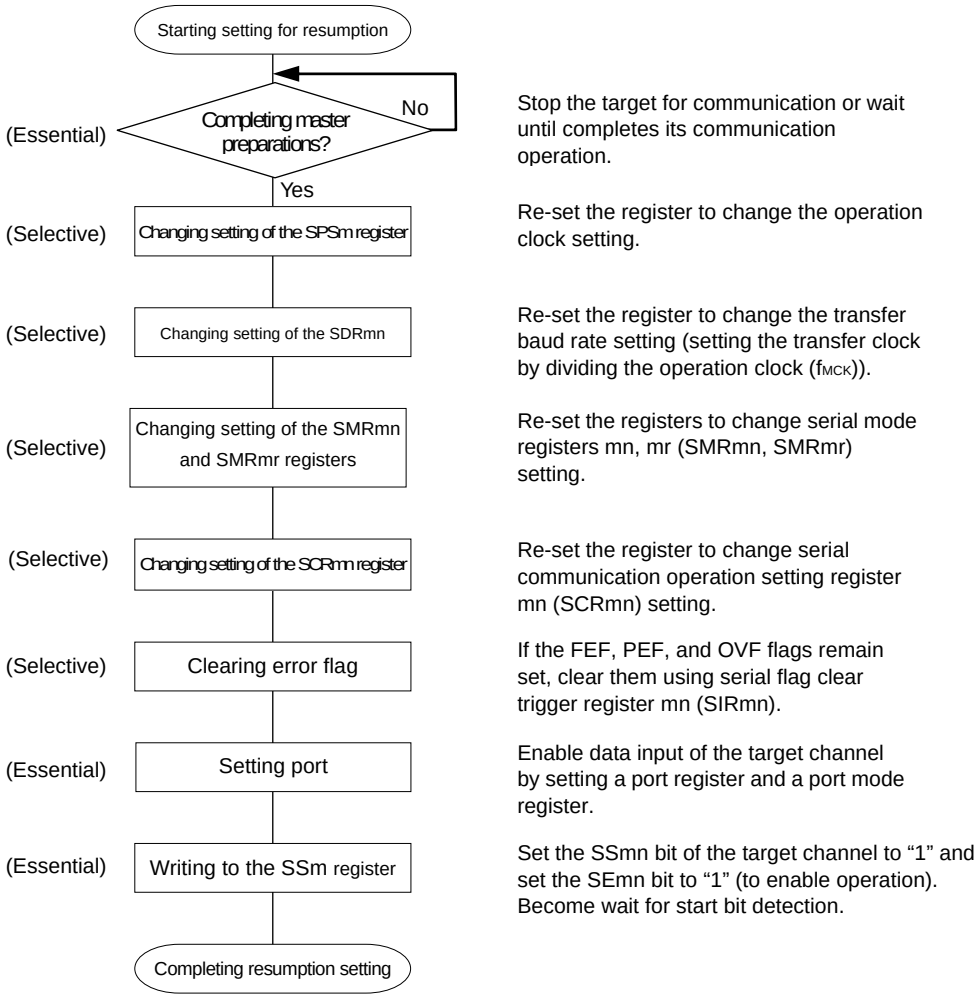


**Figure 18-87. Procedure for Resuming UART Reception**

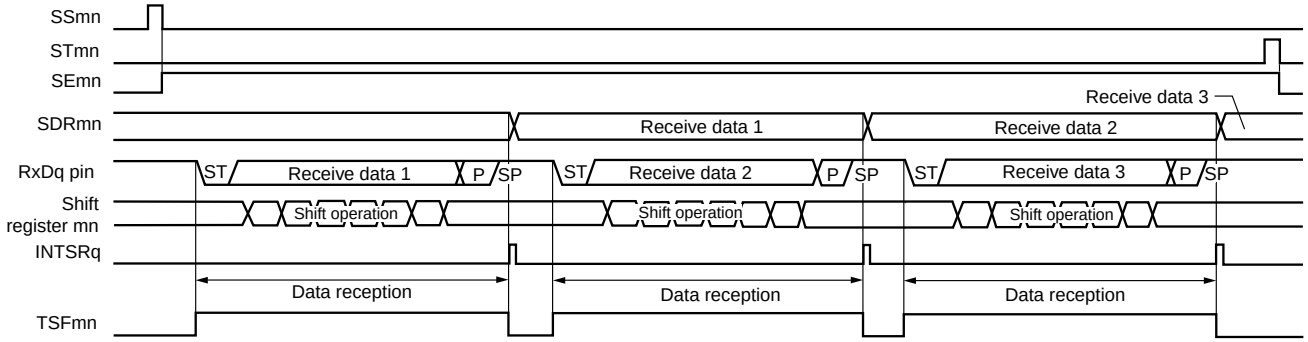



**Caution After is set RXEmn bit to 1 of SCRmn register, set the SSmn = 1 from an interval of at least four clocks of $f_{MCK}$.**

**Remark** If PER0 is rewritten while stopping the master transmission and the clock supply is stopped, wait until the transmission target (slave) stops or transmission finishes, and then perform initialization instead of restarting the transmission.

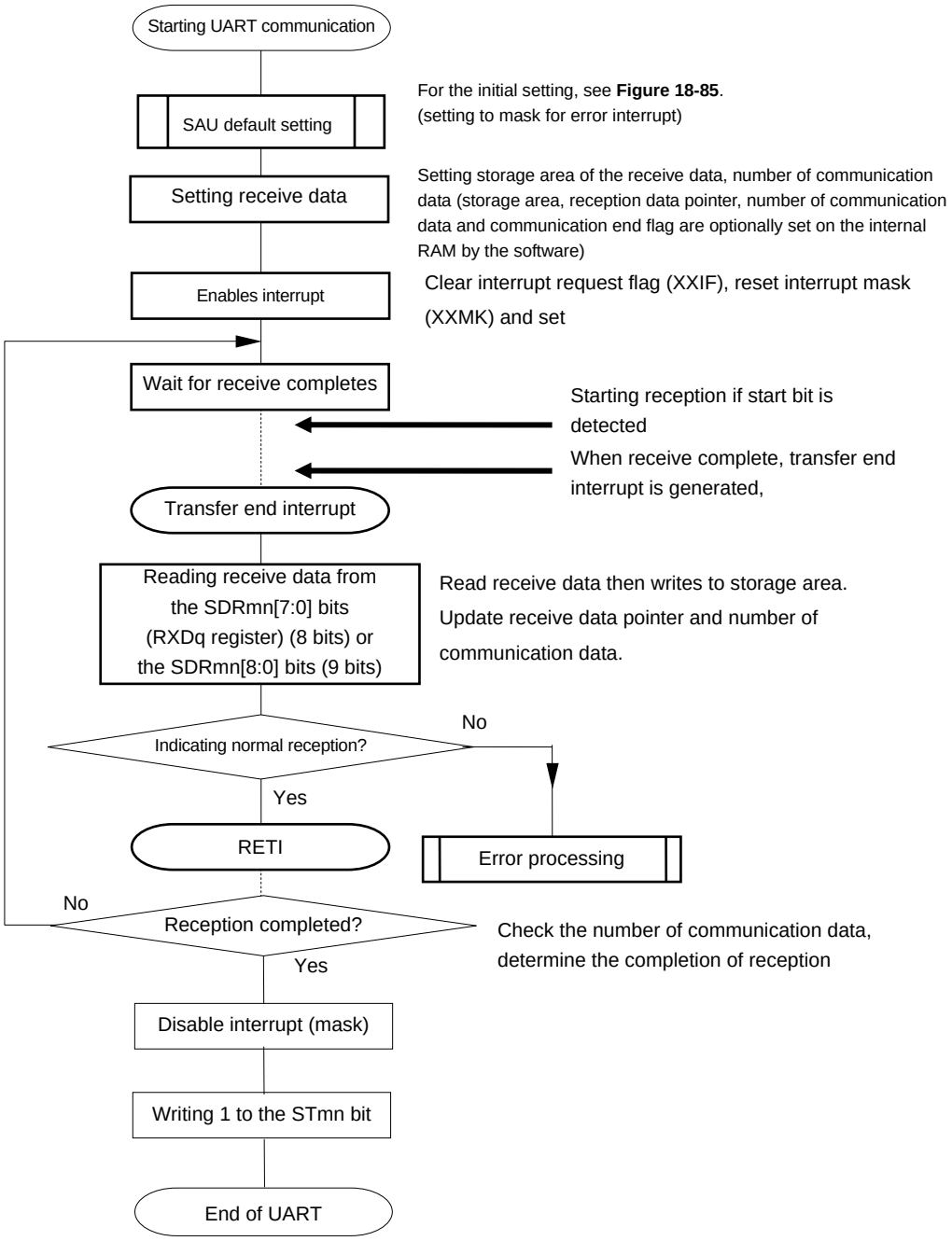
### (3) Processing flow

**Figure 18-88. Timing Chart of UART Reception**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 1, 3), mn = 01, 03, 11
r: Channel number (r = n – 1), q: UART number (q = 0 to 2)

**Figure 18-89. Flowchart of UART Reception**



### 18.6.3 SNOOZE mode function

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 UART to perform reception operations without CPU operation. Only UART0 can be set to the SNOOZE mode.

When using UARTq in the SNOOZE mode, make the following settings before entering the STOP mode. (See **Figure 18-92** and **Figure 18-94 Flowchart of SNOOZE Mode Operation**.)

- 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 18-3.
- Set the EOCmn and SSECmn bits. This is for enabling or stopping generation of an error interrupt (INTSRE0) 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 completed, set the SSm1 bit of serial channel start register m (SSm) to 1.
- <R> 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.

**Cautions 1. The SNOOZE mode can only be specified when the high-speed on-chip oscillator clock is selected for $f_{CLK}$.**

**2. The maximum transfer rate when using UARTq in the SNOOZE mode is 4800 bps.**

**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**

**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, or OVFmn flag before setting the SWC0 bit to 1 and read the value in bits 7 to 0 (RxDq register) of the SDRm1 register.**

<R> **5. The CPU shifts from the STOP mode to the SNOOZE mode on detecting the valid edge of the RxDq signal. Note, however, that transfer 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 transfer.**

**Table 18-3. Baud Rate Setting for UART Reception in SNOOZE Mode**

| High-speed On-chip Oscillator ($f_{IH}$) | Baud Rate for UART Reception in SNOOZE Mode | | | |
|---|---|---|---|---|
| | Baud Rate of 4800 bps | | | |
| | Operation Clock ($f_{MCK}$) | SDRmn[15:9] | Maximum Permissible Value | Minimum Permissible Value |
| 24 MHz ± 1.0%Note | $f_{CLK}/2^5$ | 79 | 1.60% | –2.18% |
| 12 MHz ± 1.0%Note | $f_{CLK}/2^4$ | 79 | 1.60% | –2.19% |
| 6 MHz ± 1.0%Note | $f_{CLK}/2^3$ | 79 | 1.60% | –2.19% |
| 3 MHz ± 1.0%Note | $f_{CLK}/2^2$ | 79 | 1.60% | –2.19% |

**Note** When the accuracy of the clock frequency of the high-speed on-chip oscillator is ±1.5%, the permissible range becomes smaller as shown below.

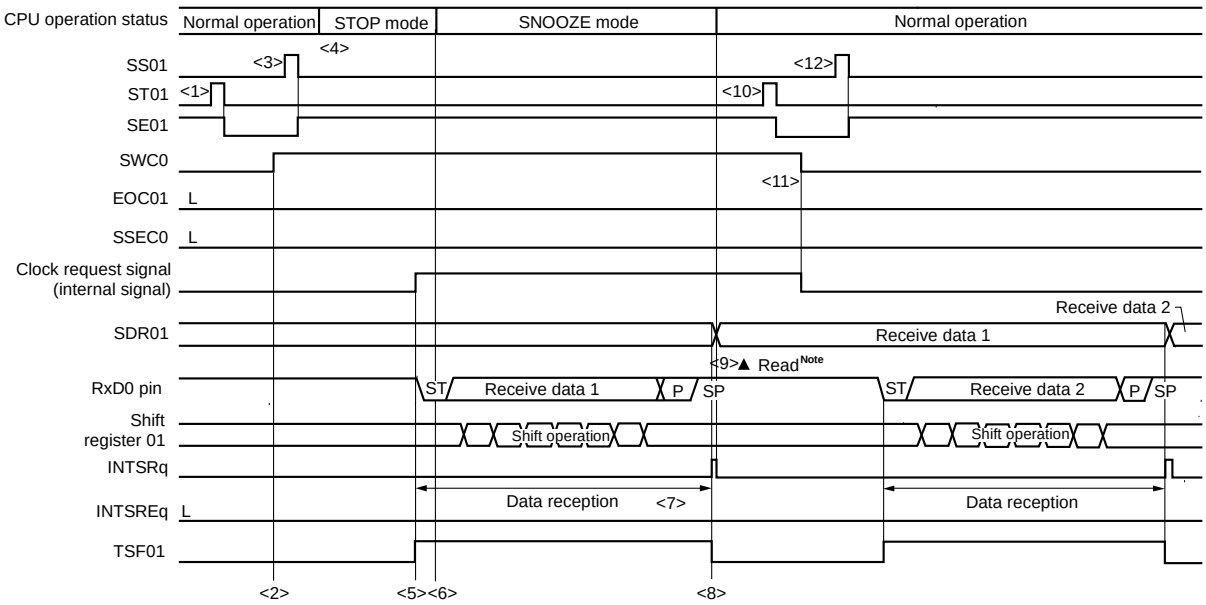
- In the case of $f_{IH}$ ± 1.5%, perform (Maximum permissible value – 0.5%) and (Minimum permissible value + 0.5%) to the values in the above table.

**Remark** The maximum permissible value and minimum permissible value are permissible values for the baud rate in UART reception. The baud rate on the transmitting side should be set to fall inside this range.

**(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.

<R> **Figure 18-90. Timing Chart of SNOOZE Mode Operation (EOCm1 = 0, SSECm = 0/1)**




**Note** Read the received data when SWCm is 1

**Caution Before switching to the SNOOZE mode or after reception operation in the SNOOZE mode finishes, set the STm1 bit to 1 (clear the SEm1 bit, and stop the operation).**
**And after completion the receive operation, also clearing SWCm bit to 0 (SNOOZE mode release).**

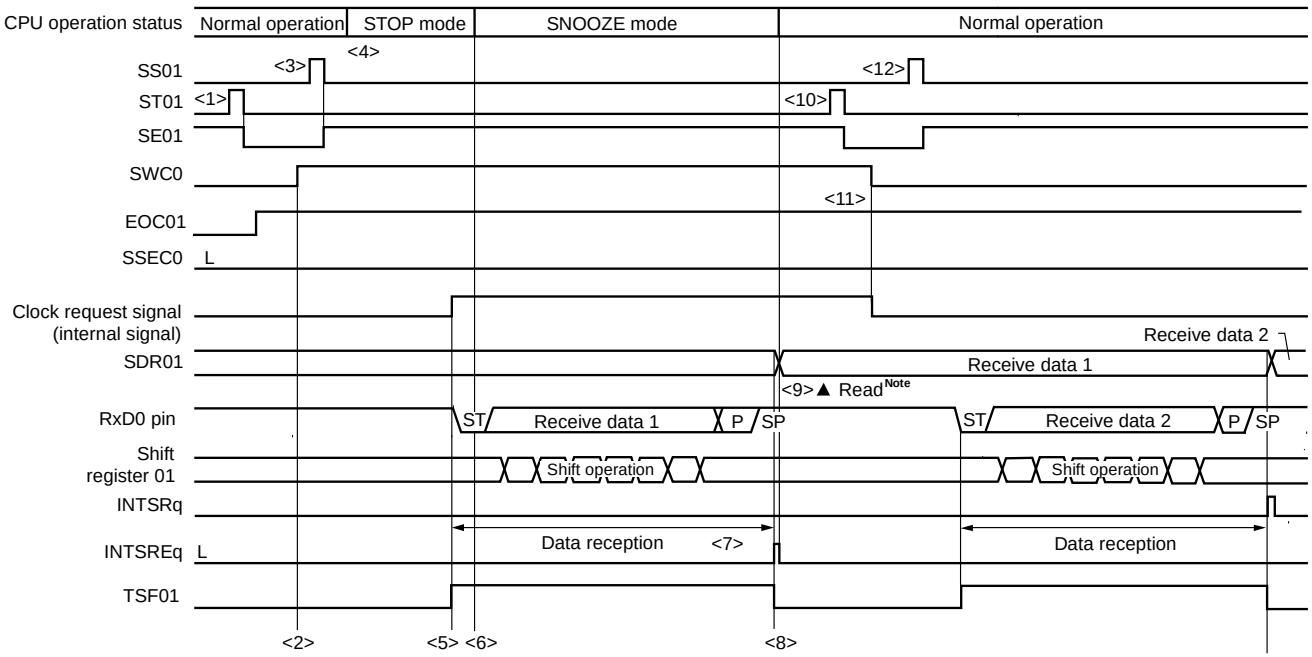
**Remarks 1.** <1> to <12> in the figure correspond to <1> to <12> in **Figure 18-92 Flowchart of SNOOZE Mode Operation (EOCm1 = 0, SSECm = 0/1 or EOCm1 = 1, SSECm = 0)**.

**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.

<R> **Figure 18-91. Timing Chart of SNOOZE Mode Operation (EOCm1 = 1, SSECm = 0)**



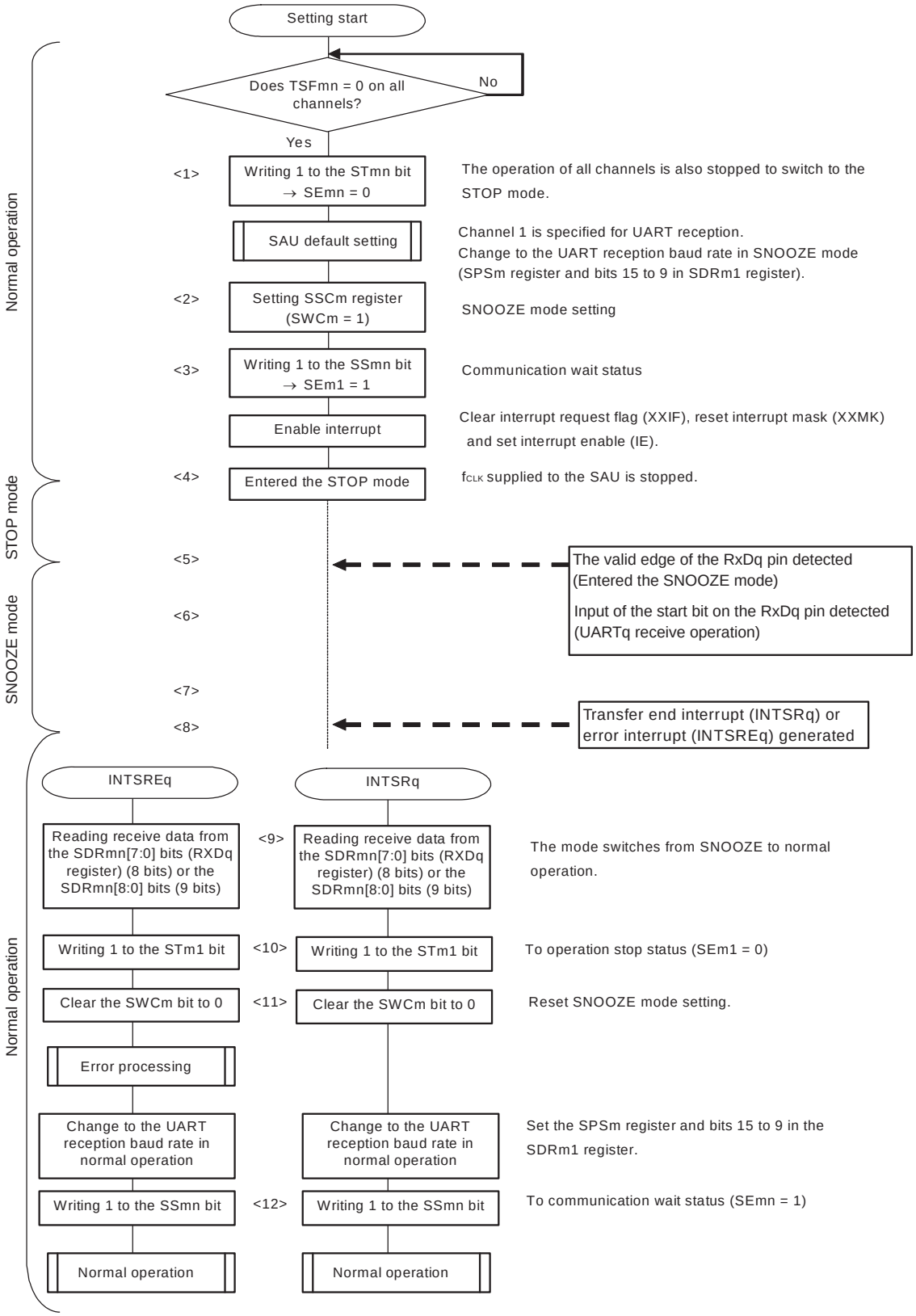

**Note** Read the received data when SWCm is 1

**Caution Before switching to the SNOOZE mode or after reception operation in the SNOOZE mode finishes, set the STm1 bit to 1 (clear the SEm1 bit, and stop the operation).**
**And after completion the receive operation, also clearing SWCm bit to 0 (SNOOZE mode release).**

**Remarks 1.** <1> to <12> in the figure correspond to <1> to <12> in **Figure 18-92 Flowchart of SNOOZE Mode Operation (EOCm1 = 0, SSECm = 0/1 or EOCm1 = 1, SSECm = 0)**.

**2.** m = 0; q = 0

<R> **Figure 18-92. Flowchart of SNOOZE Mode Operation (EOCm1 = 0, SSECm = 0/1 or EOCm1 = 1, SSECm = 0)**



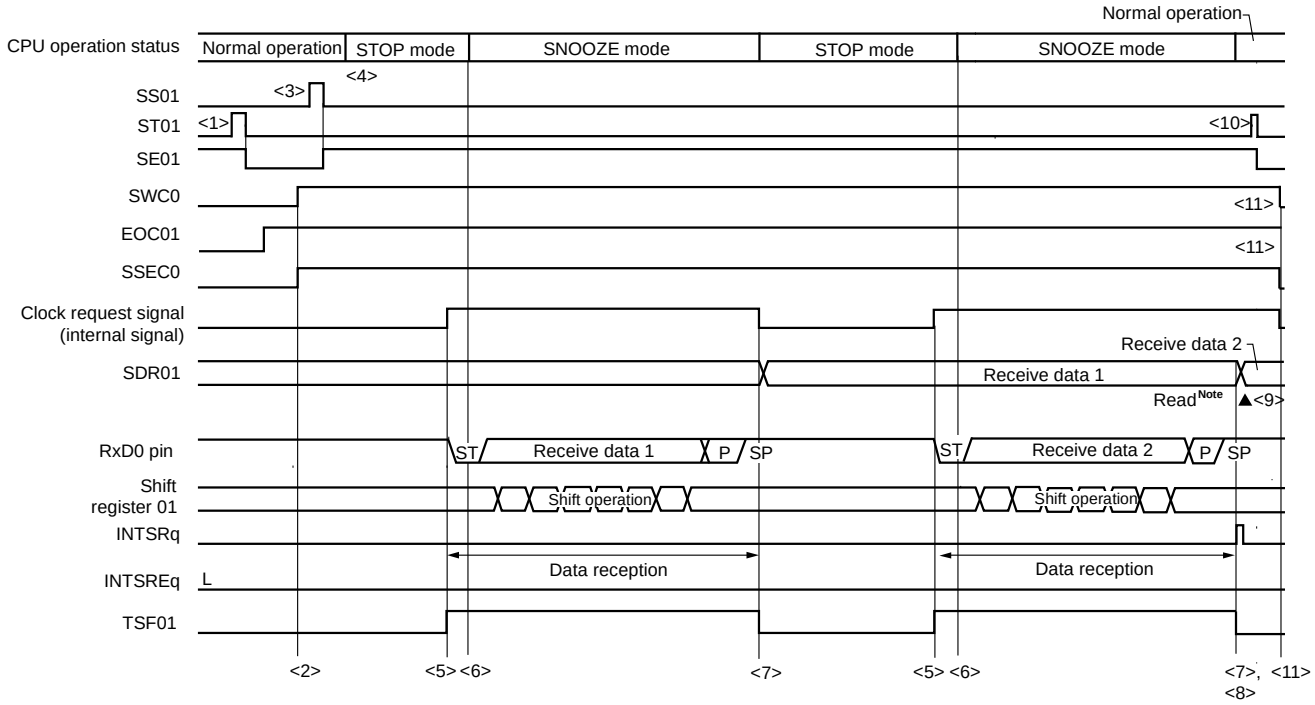

**Remarks 1.** <1> to <12> in the figure correspond to <1> to <12> in **Figure 18-90 Timing Chart of SNOOZE Mode Operation (EOCm1 = 0, SSECm = 0/1)** and **Figure 18-91 Timing Chart of SNOOZE Mode Operation (EOCm1 = 1, SSECm = 0)**.

**2.** m = 0; q = 0

**(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.

<R> **Figure 18-93. Timing Chart of SNOOZE Mode Operation (EOCm1 = 1, SSECm = 1)**




**Note** Read the received data when SWCm = 1.

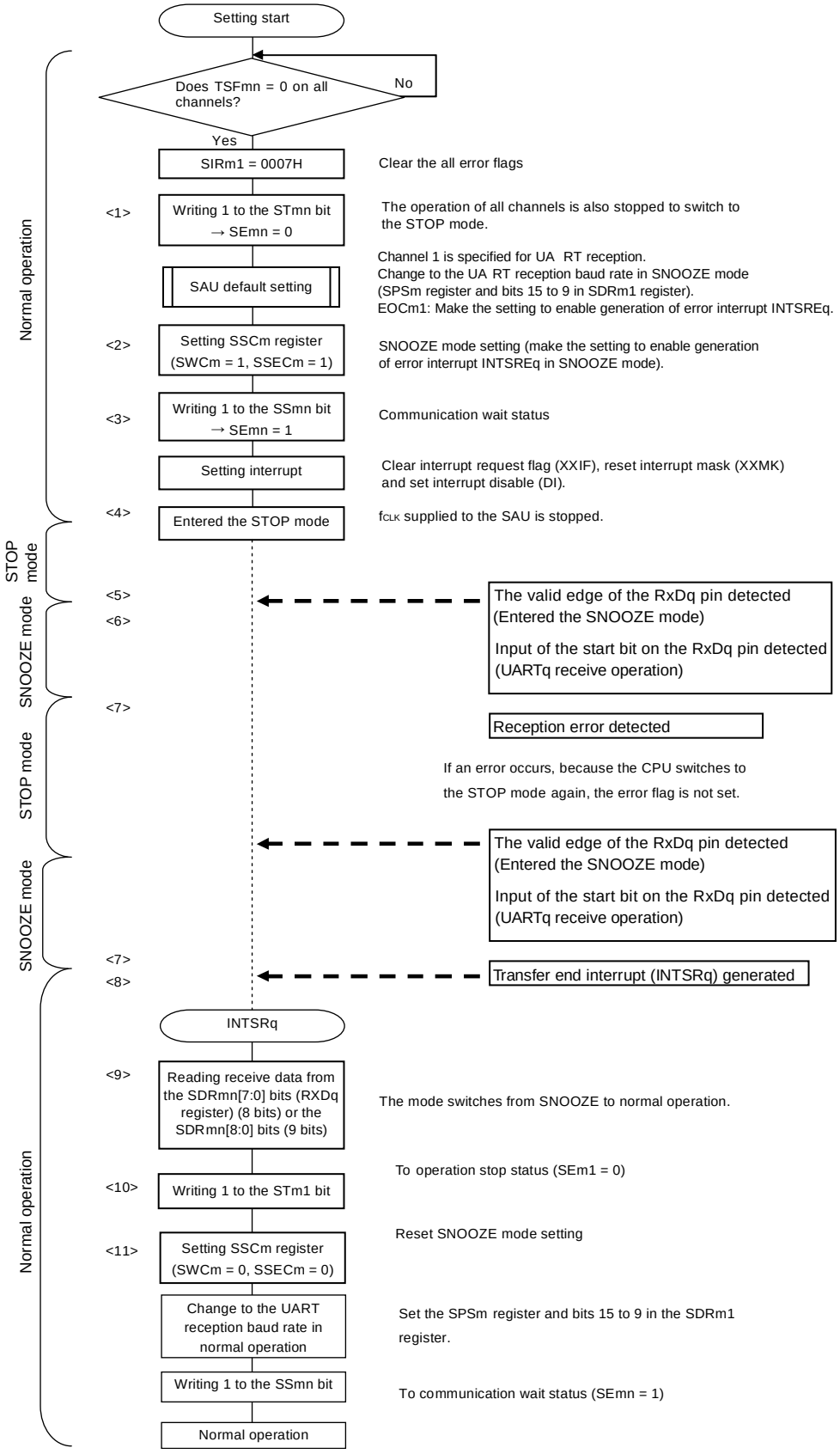
**Cautions 1. Before switching to the SNOOZE mode or after reception operation in the SNOOZE mode finishes, set the STm1 bit to 1 (clear the SEm1 bit and stop the operation).**
**After the receive operation completes, also clear the SWCm bit to 0 (SNOOZE mode release).**

**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, or OVFm1 flag 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).**

**Remarks 1.** <1> to <11> in the figure correspond to <1> to <11> in **Figure 18-94 Flowchart of SNOOZE Mode Operation (EOCm1 = 1, SSECm = 1)**.

**2.** m = 0; q = 0

<R> **Figure 18-94. Flowchart of SNOOZE Mode Operation (EOCm1 = 1, SSECm = 1)**




(**Caution** and **Remarks** are listed on the next page.)

**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, or OVFm1 flag 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).**

**Remarks 1.** <1> to <11> in the figure correspond to <1> to <11> in **Figure 18-93 Timing Chart of SNOOZE Mode Operation (EOCm1 = 1, SSECm = 1)**.

**2.** m = 0; q = 0

### 18.6.4 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.**

**Remarks 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.

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 2), mn = 00 to 03, 10, 11

The operation clock ($f_{MCK}$) is determined by serial clock select register m (SPSm) and bit 15 (CKSmn) of serial mode register mn (SMRmn).

**Table 18-4. 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 | | $f_{CLK}$ = 24 MHz |
| 0 | X | X | X | X | 0 | 0 | 0 | 0 | $f_{CLK}$ | 24 MHz |
| | X | X | X | X | 0 | 0 | 0 | 1 | $f_{CLK}/2$ | 12 MHz |
| | X | X | X | X | 0 | 0 | 1 | 0 | $f_{CLK}/2^2$ | 6 MHz |
| | X | X | X | X | 0 | 0 | 1 | 1 | $f_{CLK}/2^3$ | 3 MHz |
| | X | X | X | X | 0 | 1 | 0 | 0 | $f_{CLK}/2^4$ | 1.5 MHz |
| | X | X | X | X | 0 | 1 | 0 | 1 | $f_{CLK}/2^5$ | 750 kHz |
| | X | X | X | X | 0 | 1 | 1 | 0 | $f_{CLK}/2^6$ | 375 kHz |
| | X | X | X | X | 0 | 1 | 1 | 1 | $f_{CLK}/2^7$ | 187.5 kHz |
| | X | X | X | X | 1 | 0 | 0 | 0 | $f_{CLK}/2^8$ | 93.8 kHz |
| | X | X | X | X | 1 | 0 | 0 | 1 | $f_{CLK}/2^9$ | 46.9 kHz |
| | X | X | X | X | 1 | 0 | 1 | 0 | $f_{CLK}/2^{10}$ | 23.4 kHz |
| | X | X | X | X | 1 | 0 | 1 | 1 | $f_{CLK}/2^{11}$ | 11.7 kHz |
| | X | X | X | X | 1 | 1 | 0 | 0 | $f_{CLK}/2^{12}$ | 5.86 kHz |
| | X | X | X | X | 1 | 1 | 0 | 1 | $f_{CLK}/2^{13}$ | 2.93 kHz |
| | X | X | X | X | 1 | 1 | 1 | 0 | $f_{CLK}/2^{14}$ | 1.46 kHz |
| | X | X | X | X | 1 | 1 | 1 | 1 | $f_{CLK}/2^{15}$ | 732 Hz |
| 1 | 0 | 0 | 0 | 0 | X | X | X | X | $f_{CLK}$ | 24 MHz |
| | 0 | 0 | 0 | 1 | X | X | X | X | $f_{CLK}/2$ | 12 MHz |
| | 0 | 0 | 1 | 0 | X | X | X | X | $f_{CLK}/2^2$ | 6 MHz |
| | 0 | 0 | 1 | 1 | X | X | X | X | $f_{CLK}/2^3$ | 3 MHz |
| | 0 | 1 | 0 | 0 | X | X | X | X | $f_{CLK}/2^4$ | 1.5 MHz |
| | 0 | 1 | 0 | 1 | X | X | X | X | $f_{CLK}/2^5$ | 750 kHz |
| | 0 | 1 | 1 | 0 | X | X | X | X | $f_{CLK}/2^6$ | 375 kHz |
| | 0 | 1 | 1 | 1 | X | X | X | X | $f_{CLK}/2^7$ | 187.5 kHz |
| | 1 | 0 | 0 | 0 | X | X | X | X | $f_{CLK}/2^8$ | 93.8 kHz |
| | 1 | 0 | 0 | 1 | X | X | X | X | $f_{CLK}/2^9$ | 46.9 kHz |
| | 1 | 0 | 1 | 0 | X | X | X | X | $f_{CLK}/2^{10}$ | 23.4 kHz |
| | 1 | 0 | 1 | 1 | X | X | X | X | $f_{CLK}/2^{11}$ | 11.7 kHz |
| | 1 | 1 | 0 | 0 | X | X | X | X | $f_{CLK}/2^{12}$ | 5.86 kHz |
| | 1 | 1 | 0 | 1 | X | X | X | X | $f_{CLK}/2^{13}$ | 2.93 kHz |
| | 1 | 1 | 1 | 0 | X | X | X | X | $f_{CLK}/2^{14}$ | 1.46 kHz |
| | 1 | 1 | 1 | 1 | X | X | X | X | $f_{CLK}/2^{15}$ | 732 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).

**Remarks 1.** X: Don't care

**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 $f_{CLK}$ = 24 MHz.

| UART Baud Rate (Target Baud Rate) | $f_{CLK}$ = 24 MHz | | | |
|---|---|---|---|---|
| | Operation Clock ($f_{MCK}$) | SDRmn[15:9] | Calculated Baud Rate | Error from Target Baud Rate |
| 300 bps | $f_{CLK}/2^9$ | 77 | 300.48 bps | +0.16 % |
| 600 bps | $f_{CLK}/2^8$ | 77 | 600.96 bps | +0.16 % |
| 1200 bps | $f_{CLK}/2^7$ | 77 | 1201.92 bps | +0.16 % |
| 2400 bps | $f_{CLK}/2^6$ | 77 | 2403.85 bps | +0.16 % |
| 4800 bps | $f_{CLK}/2^5$ | 77 | 4807.69 bps | +0.16 % |
| 9600 bps | $f_{CLK}/2^4$ | 77 | 9615.38 bps | +0.16 % |
| 19200 bps | $f_{CLK}/2^3$ | 77 | 19230.8 bps | +0.16 % |
| 31250 bps | $f_{CLK}/2^3$ | 47 | 31250.0 bps | ±0.0 % |
| 38400 bps | $f_{CLK}/2^2$ | 77 | 38461.5 bps | +0.16 % |
| 76800 bps | $f_{CLK}/2$ | 77 | 76923.1 bps | +0.16 % |
| 153600 bps | $f_{CLK}$ | 77 | 153846 bps | +0.16 % |
| 312500 bps | $f_{CLK}$ | 37 | 315789 bps | +1.05 % |

**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.

$$\text{(Maximum receivable baud rate)} = \frac{2 \times k \times Nfr}{2 \times k \times Nfr - k + 2} \times Brate$$

$$\text{(Minimum receivable baud rate)} = \frac{2 \times k \times (Nfr - 1)}{2 \times k \times Nfr - k - 2} \times Brate$$

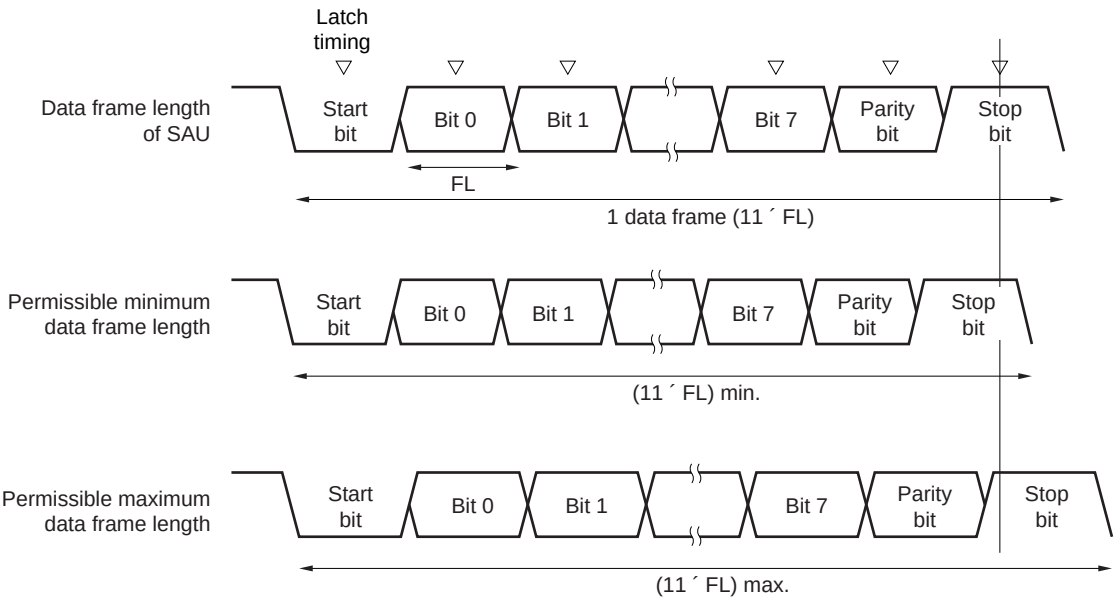
Brate: Calculated baud rate value at the reception side (See **18.6.4 (1) Baud rate calculation expression**.)

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 18-95. Permissible Baud Rate Range for Reception (1 Data Frame Length = 11 Bits)**




As shown in Figure 18-95, 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.

### 18.6.5 Procedure for processing errors that occurred during UART (UART0 to UART2) communication

The procedure for processing errors that occurred during UART (UART0 to UART2) communication is described in Figures 18-96 and 18-97.

**Figure 18-96. Processing Procedure in Case of Parity Error or Overrun Error**

| Software Manipulation | Hardware Status | 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). | | Error type is identified and the read value is used to clear error flag. |
| Writes 1 to serial flag clear trigger register mn (SIRmn). → | Error flag is cleared. | Error can be cleared only during reading, by writing the value read from the SSRmn register to the SIRmn register without modification. |

**Figure 18-97. Processing Procedure in Case of Framing Error**

| Software Manipulation | Hardware Status | 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). | | Error type is identified and the read value is used to clear error flag. |
| Writes serial flag clear trigger register mn (SIRmn). → | Error flag is cleared. | Error can be cleared only during reading, 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 operating. | |
| 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

## 18.7 LIN Communication Operation

### 18.7.1 LIN transmission

Of UART transmission, UART0 support LIN communication.

For LIN transmission, channel 0 of unit 0 is used.

| UART | UART0 | UART1 | UART2 |
|---|---|---|---|
| Support of LIN communication | Supported | Not supported | Not supported |
| Target channel | Channel 0 of SAU0 | – | – |
| Pins used | TxD0 | – | – |
| Interrupt | INTST0 | – | – |
| | 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 | Max. $f_{MCK}$/6 [bps] (SDR00[15:9] = 2 or more), Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [bps]Note | | |
| Data phase | Non-reverse output (default: high level)<br>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 in the electrical specifications (see **CHAPTER 37 ELECTRICAL SPECIFICATIONS**). In addition, LIN communication is usually 2.4/9.6/19.2 kbps is often used.

**Remark** $f_{MCK}$: Operation clock frequency of target channel

$f_{CLK}$: System 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.

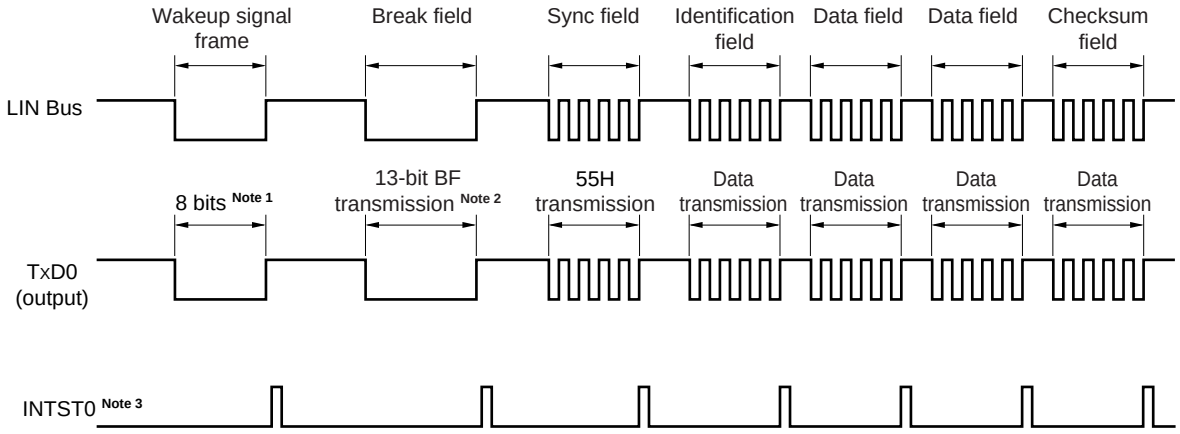
Usually, the master is connected to a network such as CAN (Controller Area Network).

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 18-98 outlines a master transmission operation of LIN.

**Figure 18-98. Transmission Operation of LIN**




**Notes 1.** Set the baud rate in accordance with the wakeup signal regulations and transmit data of 80H.

**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

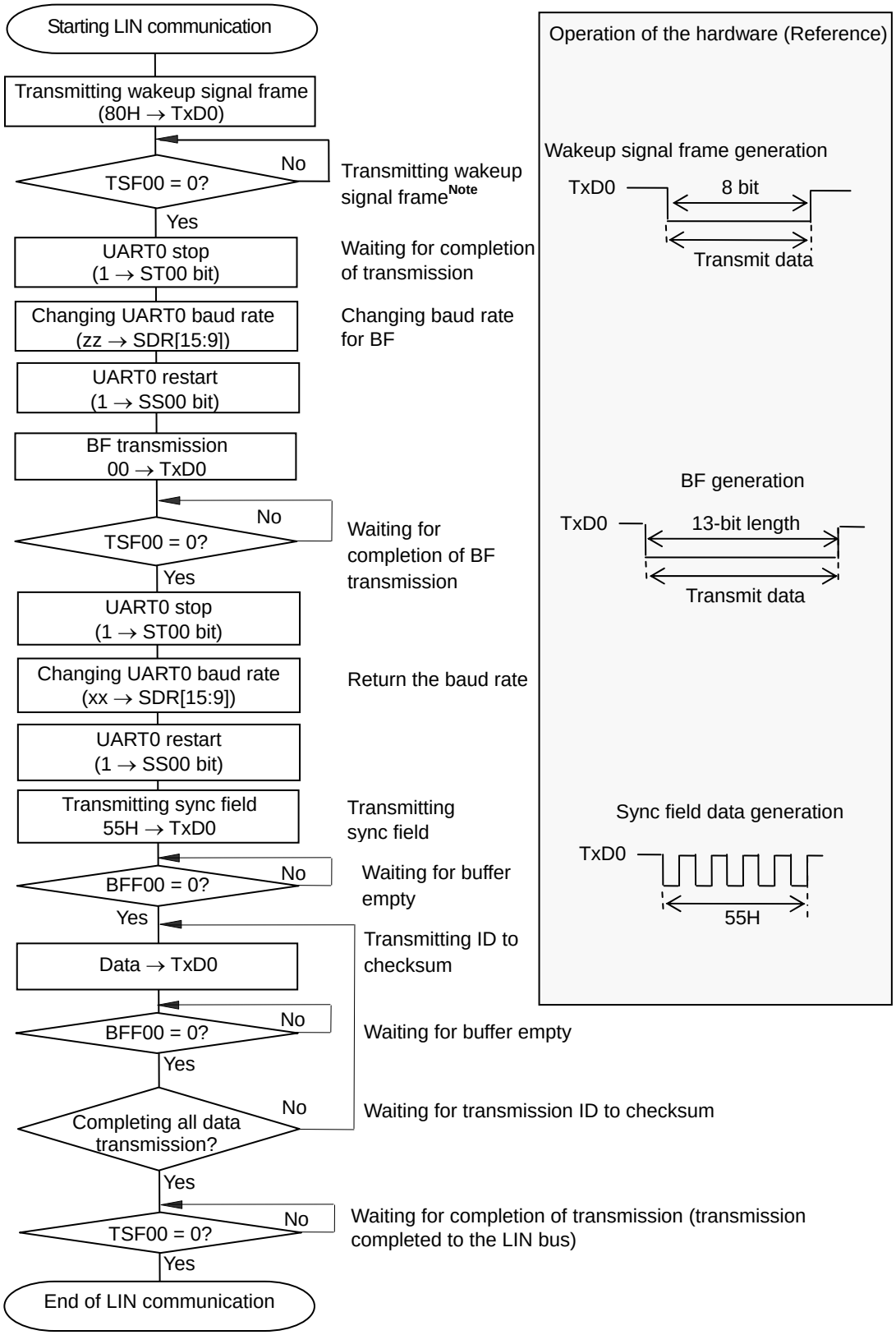
By transmitting data of 00H at this baud rate, a break field is generated.

**3.** INTST0 is output upon completion of transmission. INTST0 is also output at BF transmission.

**Remark** The interval between fields is controlled by software.

<R>

**Figure 18-99. Flowchart for LIN Transmission**




**Note** When LIN-bus start from sleep status only

**Remark** Default setting of the UART is complete, and the flow from the transmission enable status.

### 18.7.2 LIN reception

Of UART reception, UART0 support LIN communication.

For LIN reception, channel 1 of unit 1 is used.

| UART | UART0 | UART1 | UART2 |
|---|---|---|---|
| Support of LIN communication | Supported | Not supported | Not supported |
| Target channel | Channel 1 of SAU0 | – | – |
| Pins used | RxD0 | – | – |
| Interrupt | INTSR0 | – | – |
| | Transfer end interrupt only (Setting the buffer empty interrupt is prohibited.) | | |
| Error interrupt | INTSRE0 | – | – |
| Error detection flag | • Framing error detection flag (FEF01)<br>• Overrun error detection flag (OVF01) | | |
| Transfer data length | 8 bits | | |
| Transfer rate | Max. $f_{MCK}$/6 [bps] (SDR01[15:9] = 2 or more), Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [bps]Note | | |
| Data phase | Non-reverse output (default: high level)<br>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 | | |

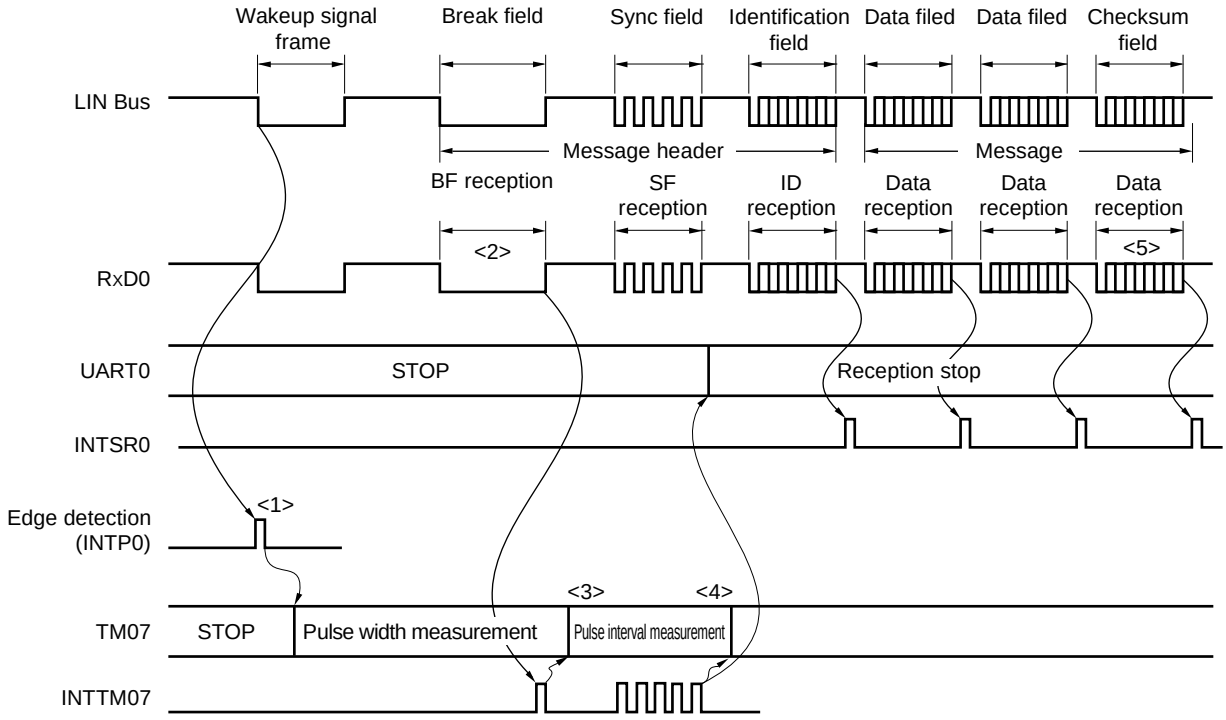
**Note** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 37 ELECTRICAL SPECIFICATIONS**).

**Remark** $f_{MCK}$: Operation clock frequency of target channel
$f_{CLK}$: System clock frequency

Figure 18-100 outlines a reception operation of LIN.

<R>

**Figure 18-100. Reception Operation of LIN**




Here is the flow of signal processing.

<1> The wakeup signal is detected by detecting an interrupt edge (INTP0) on a pin. When the wakeup signal is detected, change TM07 to pulse width measurement upon detection of the wakeup signal 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 RxD0 signal in the Sync field four times.

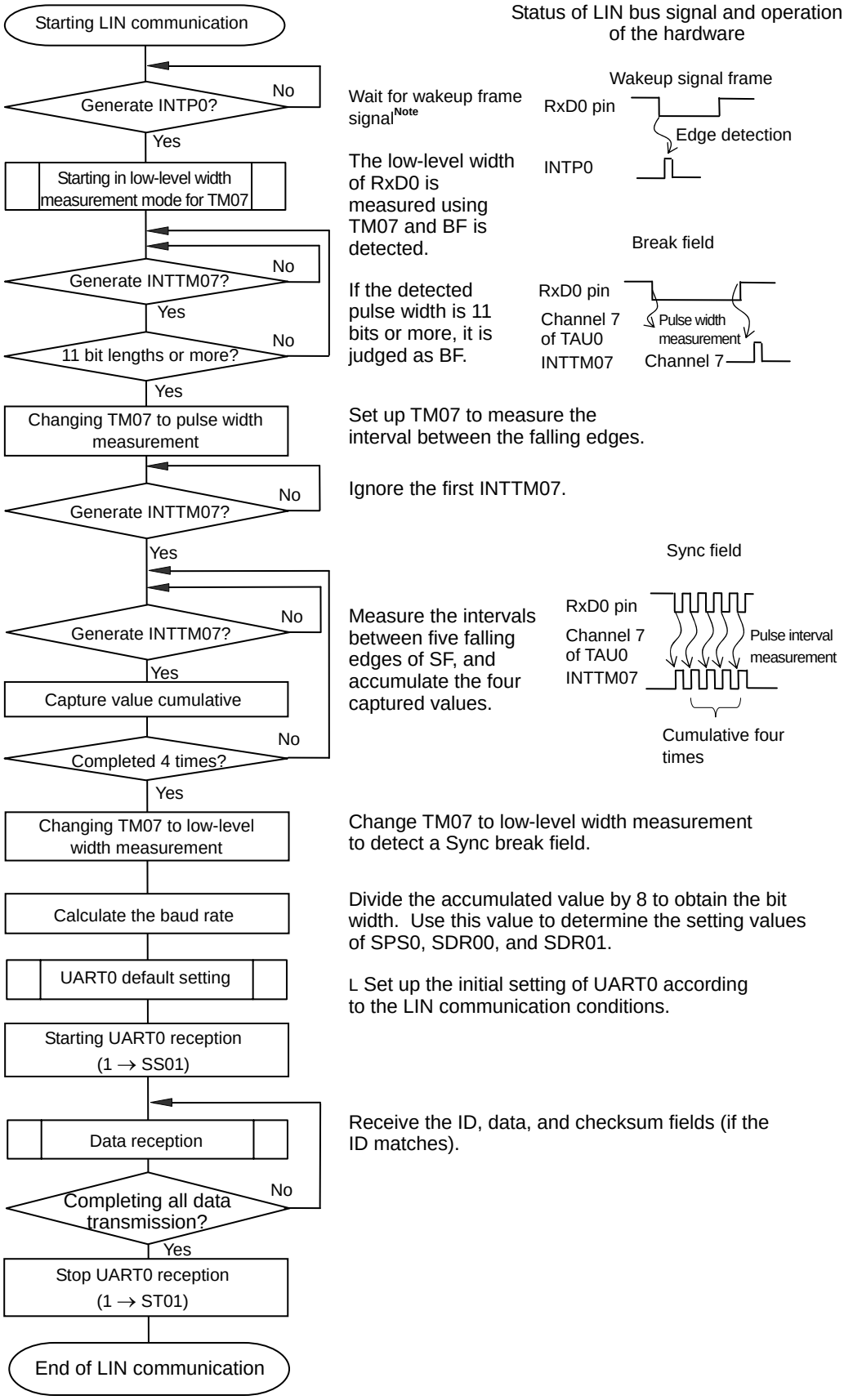
<4> When BF reception has been correctly completed, start channel 7 of the timer array unit and measure the bit interval (pulse width) of the sync field (see **7.8.3 Operation as input pulse interval measurement**).

<5> Calculate a baud rate error from the bit interval of sync field (SF). Stop UART0 once and adjust (re-set) the baud rate.

<6> The checksum field should be distinguished by software. In addition, processing to initialize UART0 after the checksum field is received and to wait for reception of BF should also be performed by software.

<R>

**Figure 18-101. Flowchart for LIN Reception**



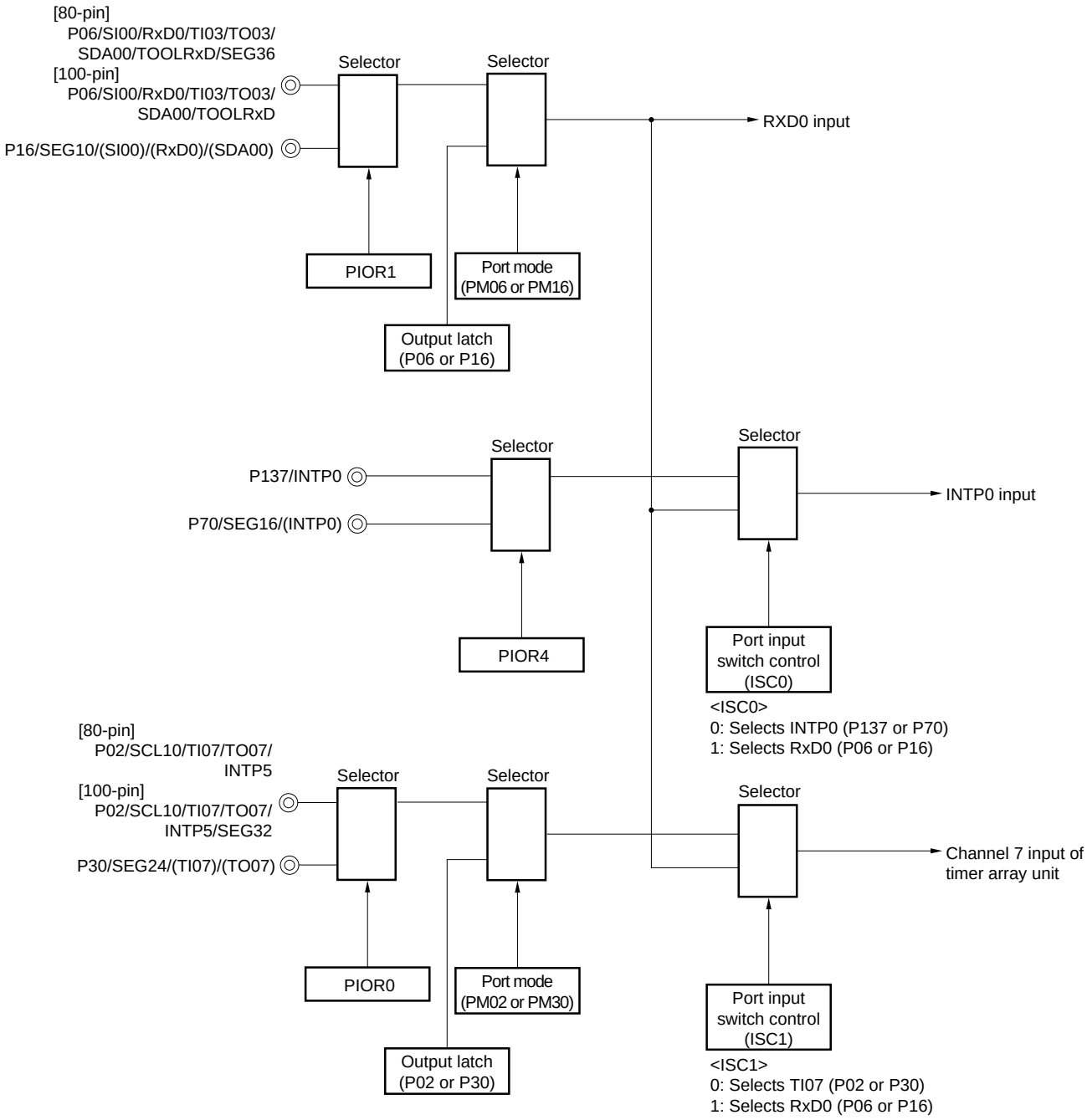

**Note** Required in the sleep status only.

Figure 18-102 shows the configuration of a port that manipulates reception of LIN.

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 controlling switch of port input (ISC0/ISC1), the input source of port input (RxD0) for reception can be input to the external interrupt pin (INTP0) and timer array unit

**Figure 18-102. Port Configuration for Manipulating Reception of LIN**




**Remarks 1.** ISC0, ISC1: Bits 0 and 1 of the input switch control register (ISC) (See **Figure 18-21**.)

PIOR0, PIOR1, PIOR4: Bits 0 to 4 of the peripheral I/O redirection register (PIOR) (See **Figure 4-8**.).

**2.** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR)**.

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 an error (The interval of the edge input to RxD0 is measured in the capture mode.)
  Measured the low-level width, determine whether break field (BF).
- Channels 0 and 1 (UART0) of serial array unit 0 (SAU0)

## 18.8 Operation of Simplified $I^2C$ (IIC00, IIC10) 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 communication function is designed to execute single communication with devices such as EEPROM, flash memory, and A/D converter, and therefore, can be used only by the master.

Operate the control registers by software for setting the start and stop conditions while observing the specifications of the $I^2C$ 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.)
- Gneration of start condition and stop condition for software

[Interrupt function]

- Transfer end interrupt

[Error detection flag]

- Parity error (ACK error)

* [Functions not supported by simplified $I^2C$]

- Slave transmission, slave reception
- Multi-master function (arbitration loss detection function)
- Wait detection function

**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 the processing flow in **18.8.3 (2)** for details.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2), mn = 00, 02

The channel supporting simplified $I^2C$ (IIC00, IIC10) is channels 0 and 2 of SAU0.

| Unit | Channel | Used as CSI | Used as UART | Used as Simplified $I^2C$ |
|---|---|---|---|---|
| 0 | 0 | CSI00 | UART0 (supporting LIN-bus) | IIC00 |
| | 1 | – | | – |
| | 2 | – | UART1 | IIC10 |
| | 3 | – | | – |
| 1 | 0 | – | UART2 (supporting IrDA) | – |
| | 1 | – | | – |

Simplified $I^2C$ (IIC00, IIC10) performs the following four types of communication operations.

- Address field transmission (See **18.8.1**.)
- Data transmission (See **18.8.2**.)
- Data reception (See **18.8.3**.)
- Stop condition generation (See **18.8.4**.)

### 18.8.1 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 | IIC10 |
|---|---|---|
| Target channel | Channel 0 of SAU0 | Channel 2 of SAU0 |
| Pins used | SCL00, SDA00Note 1 | SCL10, SDA10Note 1 |
| Interrupt | INTIIC00 | INTIIC10 |
| | 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 rateNote 2 | Max. $f_{MCK}$/4 [Hz] (SDRmn[15:9] = 1 or more) $f_{MCK}$: Operation clock frequency of target channel<br>However, the following condition must be satisfied in each mode of $I^2C$.<br>• Max. 1 MHz (fast mode plus)<br>• Max. 400 kHz (fast mode)<br>• Max. 100 kHz (standard mode) | |
| Data level | Non-reversed output (default: high level) | |
| Parity bit | No parity bit | |
| Stop bit | Appending 1 bit (for ACK reception timing) | |
| Data direction | MSB first | |

**Notes 1.** To perform communication via simplified $I^2C$, set the N-ch open-drain output ($V_{DD}$ tolerance) mode for the port output mode register (POM0) (see **4.3.5 Port output mode registers (POMxx)** for details). When IIC00, IIC10 communicating with an external device with a different potential, set the N-ch open-drain output ($V_{DD}$ tolerance) mode also for the clock input/output pins (SCL00, SCL10) (see **4.4.4 Connecting to external device with different potential (1.8 V, 2.5 V, 3 V)** for details).

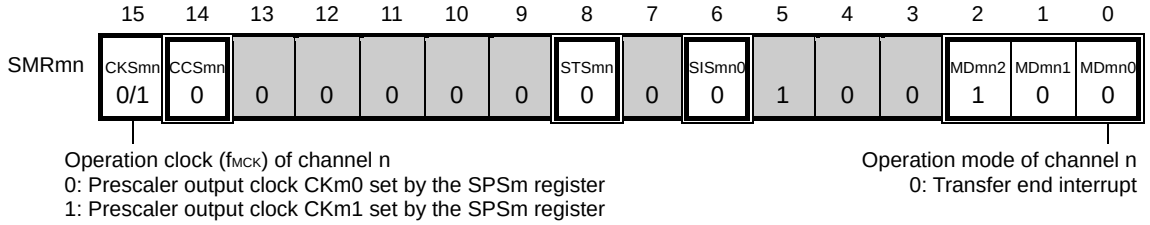
**2.** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 37 ELECTRICAL SPECIFICATIONS**)).

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2), mn = 00, 02

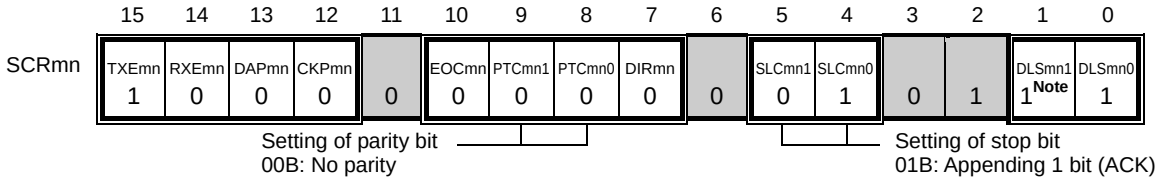
**(1) Register setting**

**Figure 18-103. Example of Contents of Registers for Address Field Transmission of Simplified $I^2C$ (IIC00, IIC10) (1/2)**

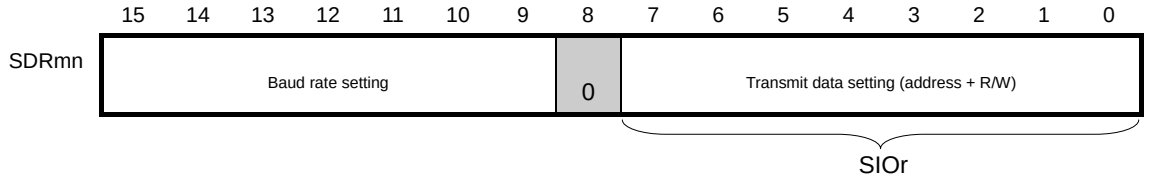
**(a) Serial mode register mn (SMRmn)**



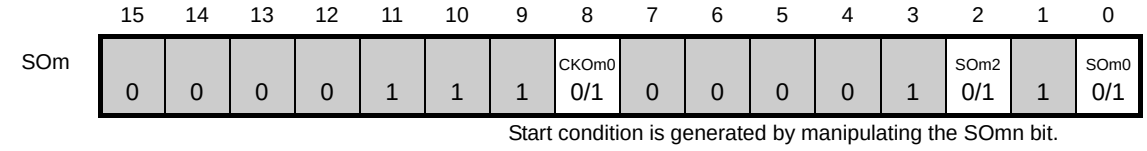

**(b) Serial communication operation setting register mn (SCRmn)**



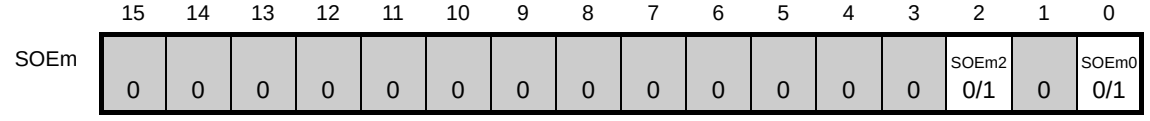

**(c) Serial data register mn (SDRmn) (lower 8 bits: SIOr)**




**(d) Serial output register m (SOm)**




**(e) Serial output enable register m (SOEm)**




**Note** Only provided for the SCR00 register. This bit is fixed to 1 for the other registers.

**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0, 2), r: IIC number (r = 00, 10), mn = 00, 02

**2.** ☐: Setting is fixed in the IIC 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 usage of the user

**Figure 18-103. Example of Contents of Registers for Address Field Transmission of Simplified $I^2C$ (IIC00, IIC10) (2/2)**

**(f) Serial channel start register m (SSm) … Sets only the bits of the target channel is 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<br>× | SSm2<br>0/1 | SSm1<br>× | SSm0<br>0/1 |

SSmn = 0 until the start condition is generated, and SSmn = 1 after generation.

**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0, 2), r: IIC number (r = 00, 10), mn = 00, 02

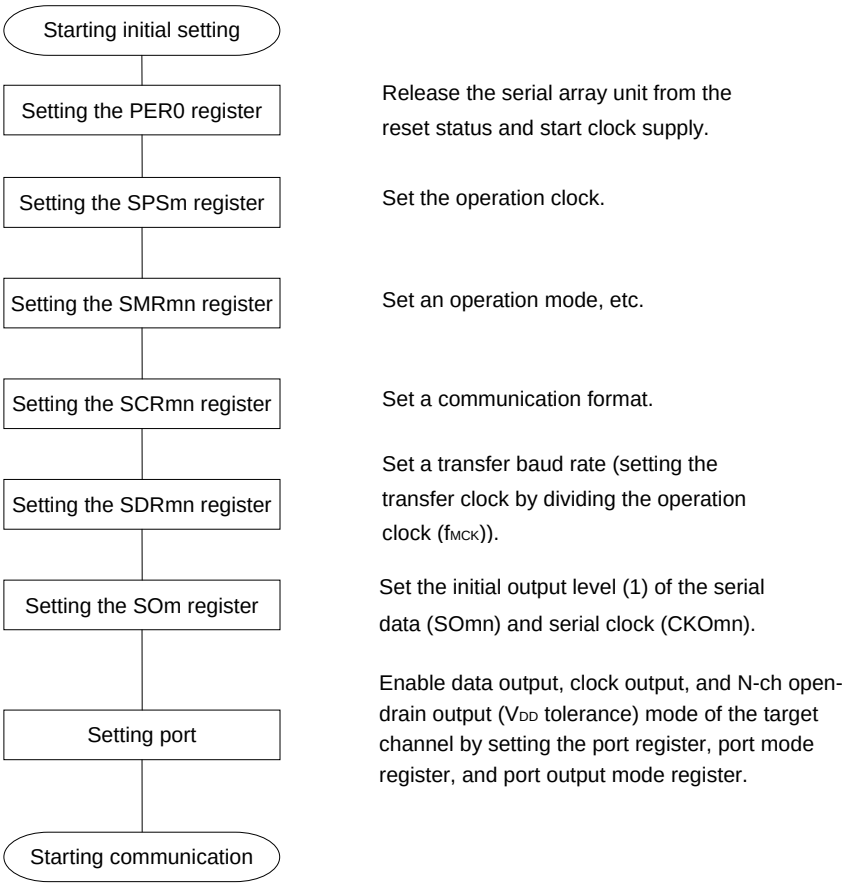
**2.** ☐ : Setting disabled (set to the initial value)

×: 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 usage of the user

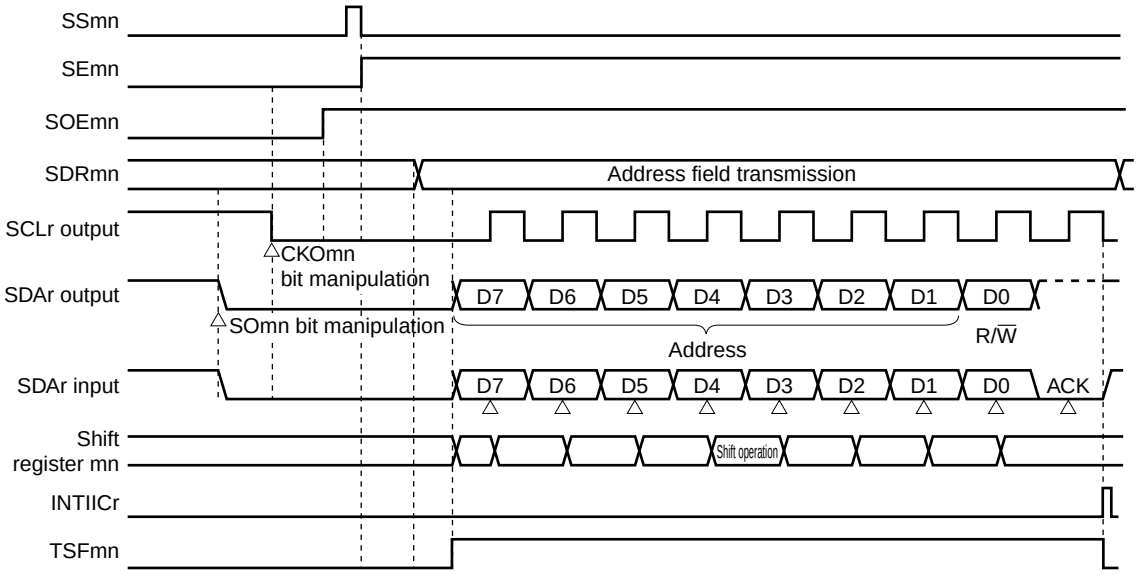
### (2) Operation procedure

**Figure 18-104. Initial Setting Procedure for Simplified $I^2C$**



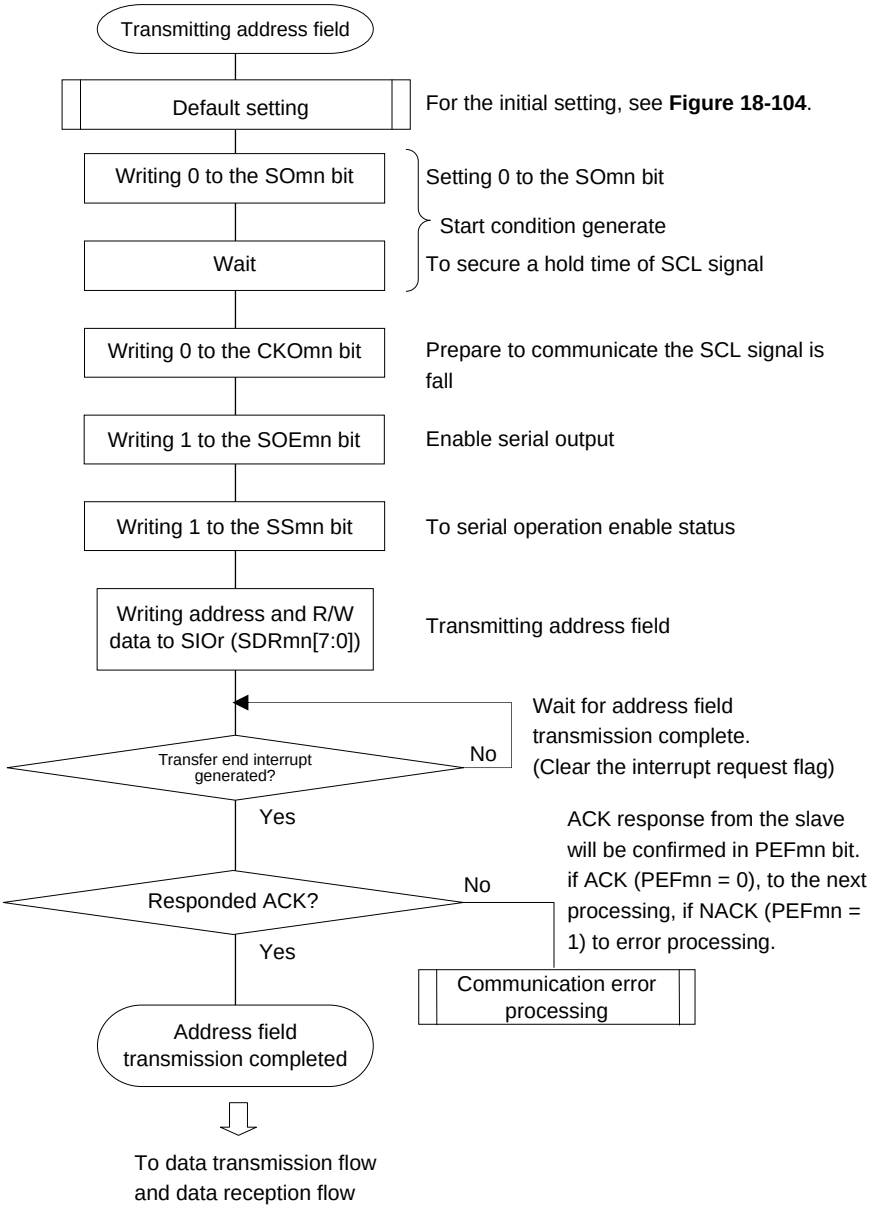
### (3) Processing flow

**Figure 18-105. Timing Chart of Address Field Transmission**




**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2), r: IIC number (r = 00, 10), mn = 00, 02

**Figure 18-106. Flowchart of Simplified I²C Address Field Transmission**



### 18.8.2 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 $I^2C$ | IIC00 | IIC10 |
|---|---|---|
| Target channel | Channel 0 of SAU0 | Channel 2 of SAU0 |
| Pins used | SCL00, SDA00Note 1 | SCL10, SDA10Note 1 |
| Interrupt | INTIIC00 | INTIIC10 |
| | 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. $f_{MCK}$/4 [Hz] (SDRmn[15:9] = 1 or more) $f_{MCK}$: Operation clock frequency of target channel<br>However, the following condition must be satisfied in each mode of $I^2C$.<br>• Max. 1 MHz (fast mode plus)<br>• Max. 400 kHz (fast mode)<br>• Max. 100 kHz (standard mode) | |
| Data level | Non-reversed output (default: high level) | |
| Parity bit | No parity bit | |
| Stop bit | Appending 1 bit (for ACK reception timing) | |
| Data direction | MSB first | |

**Notes 1.** To perform communication via simplified $I^2C$, set the N-ch open-drain output ($V_{DD}$ tolerance) mode for the port output mode registers (POM0) (see **4.3.5 Port output mode registers (POMxx)** for details). When IIC00, IIC10 communicating with an external device with a different potential, set the N-ch open-drain output ($V_{DD}$ tolerance) mode also for the clock input/output pins (SCL00, SCL10) (see **4.4.4 Connecting to external device with different potential (1.8 V, 2.5 V, 3 V)** for details).

**2.** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 37 ELECTRICAL SPECIFICATIONS**).

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2), mn = 00, 02

### (1) Register setting

**Figure 18-107. Example of Contents of Registers for Data Transmission of Simplified $I^2C$ (IIC00, IIC10) (1/2)**

**(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 | CCSmn | | | | | | STSmn | | SISmn0 | | | | MDmn2 | MDmn1 | MDmn0 |
| | 0/1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 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 | RXEmn | DAPmn | CKPmn | | EOCmn | PTCmn1 | PTCmn0 | DIRmn | | SLCmn1 | SLCmn0 | | | DLSmn1 | DLSmn0 |
| | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 Note 1 | 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 setting Note 2 | | | | | | | 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 | | | | | | CKOm2 | | CKOm0 | | | | | | SOm2 | | SOm0 |
| | 0 | 0 | 0 | 0 | 1 | 0/1 Note3 | 1 | 0/1 Note3 | 0 | 0 | 0 | 0 | 1 | 0/1 Note3 | 1 | 0/1 Note3 |

**(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 | | | | | | | | | | | | | | SOEm2 | | SOEm0 |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |

**Notes 1.** Only provided for the SCR00 register. This bit is fixed to 1 for the other registers.

**2.** Because the setting is completed by address field transmission, setting is not required.

**3.** The value varies depending on the communication data during communication operation.

**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0, 2), r: IIC number (r = 00, 10), mn = 00, 02

**2.** ☐: Setting is fixed in the IIC 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 usage of the user

**Figure 18-107. Example of Contents of Registers for Data Transmission of Simplified $I^2C$ (IIC00, IIC10) (2/2)**

**(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<br>× | SSm2<br>0/1 | SSm1<br>× | SSm0<br>0/1 |

**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0, 2), r: IIC number (r = 00, 10), mn = 00, 02

**2.** ☐ : 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 usage of the user

### (2) Processing flow

**Figure 18-108. Timing Chart of Data Transmission**

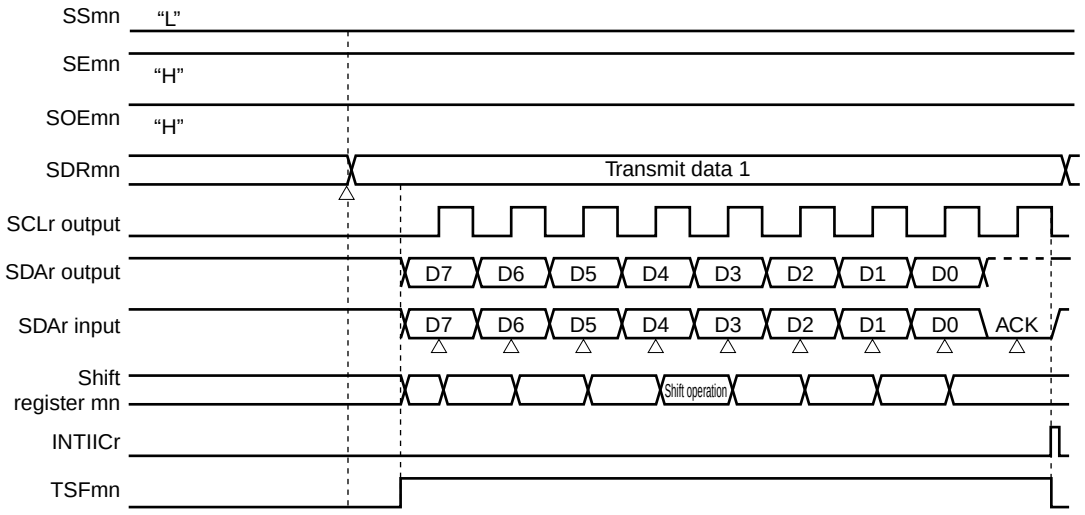
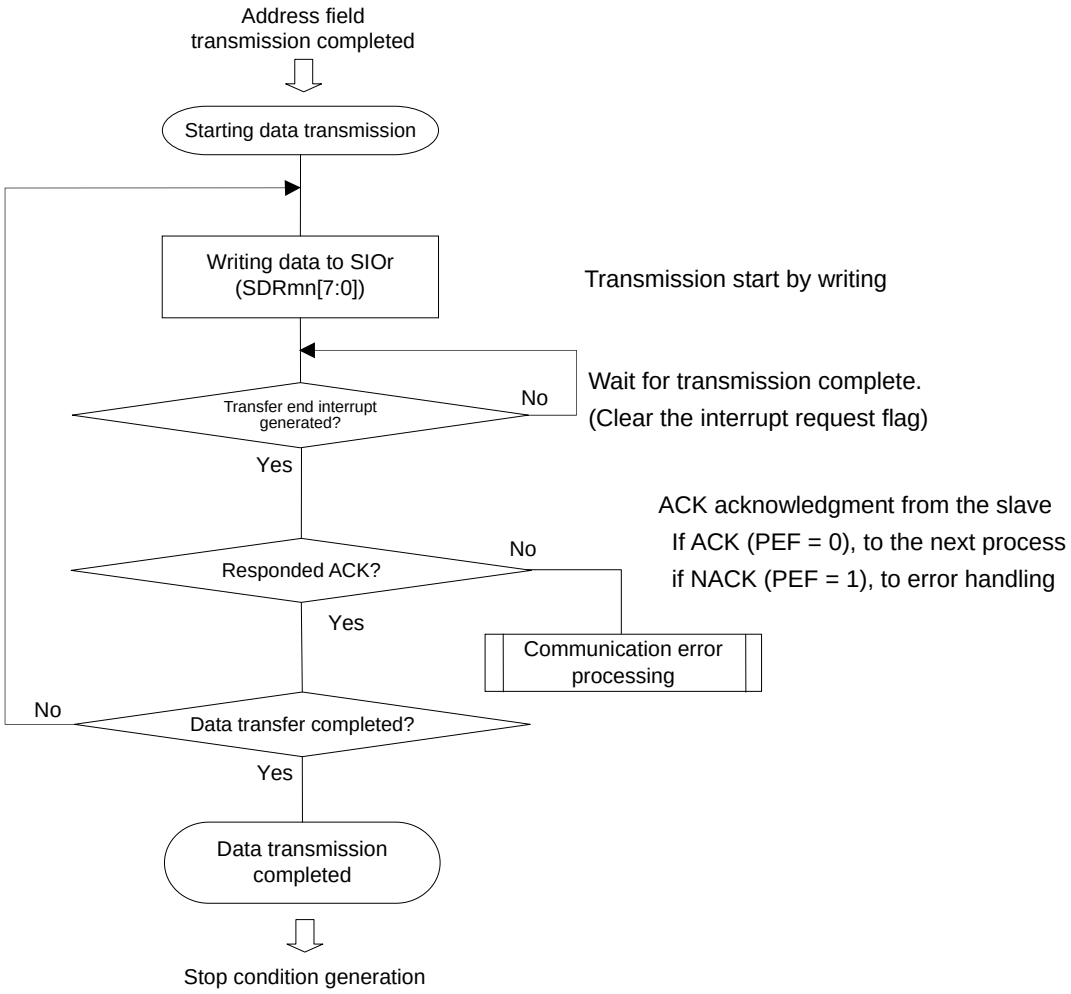



**Figure 18-109. Flowchart of Simplified $I^2C$ Data Transmission**



### 18.8.3 Data reception

Data reception is an operation to receive data to the target for transfer (slave) after transmission of an address field. After all data are received to the slave, a stop condition is generated and the bus is released.

| Simplified $I^2C$ | IIC00 | IIC10 |
|---|---|---|
| Target channel | Channel 0 of SAU0 | Channel 2 of SAU0 |
| Pins used | SCL00, SDA00Note 1 | SCL10, SDA10Note 1 |
| Interrupt | INTIIC00 | INTIIC10 |
| | 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. $f_{MCK}$/4 [Hz] (SDRmn[15:9] = 1 or more) $f_{MCK}$: Operation clock frequency of target channel<br>However, the following condition must be satisfied in each mode of $I^2C$.<br>• Max. 1 MHz (fast mode plus)<br>• Max. 400 kHz (fast mode)<br>• Max. 100 kHz (standard mode) | |
| Data level | Non-reversed output (default: high level) | |
| Parity bit | No parity bit | |
| Stop bit | Appending 1 bit (ACK transmission) | |
| Data direction | MSB first | |

**Notes 1.** To perform communication via simplified $I^2C$, set the N-ch open-drain output ($V_{DD}$ tolerance) mode for the port output mode registers (POM0) (see **4.3.5 Port output mode registers (POMxx)** for details). When IIC00, IIC10 communicating with an external device with a different potential, set the N-ch open-drain output ($V_{DD}$ tolerance) mode also for the clock input/output pins (SCL00, SCL10) (see **4.4.4 Connecting to external device with different potential (1.8 V, 2.5 V, 3 V)** for details).

**2.** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 37 ELECTRICAL SPECIFICATIONS**).

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2), mn = 00, 02

### (1) Register setting

**Figure 18-110. Example of Contents of Registers for Data Reception of Simplified I²C (IIC00, IIC10) (1/2)**

**(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 | CCSmn | 0 | 0 | 0 | 0 | 0 | STSmn | 0 | SISmn0 | 1 | 0 | 0 | MDmn2 | MDmn1 | MDmn0 |
| | 0/1 | 0 | | | | | | 0 | | 0 | | | | 1 | 0 | 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 | RXEmn | DAPmn | CKPmn | 0 | EOCmn | PTCmn1 | PTCmn0 | DIRmn | 0 | SLCmn1 | SLCmn0 | 0 | 1 | DLSmn1 | DLSmn0 |
| | 0 | 1 | 0 | 0 | | 0 | 0 | 0 | 0 | | 0 | 1 | | | 1 Note 1 | 1 |

**(c) Serial data register mn (SDRmn) (lower 8 bits: SIOr)**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SDRmn | Baud rate setting Note 2 | | | | | | | 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 | 1 | CKOm2 | 1 | CKOm0 | 0 | 0 | 0 | 0 | 1 | SOm2 | 1 | SOm0 |
| | | | | | | 0/1 Note3 | | 0/1 Note3 | | | | | | 0/1 Note3 | | 0/1 Note3 |

**(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 | 0 | SOEm2 | 0 | SOEm0 |
| | | | | | | | | | | | | | | 0/1 | | 0/1 |

**Notes 1.** Only provided for the SCR00 register. This bit is fixed to 1 for the other registers.

**2.** The baud rate setting is not required because the baud rate has already been set when the address field was transmitted.

**3.** The value varies depending on the communication data during communication operation.

**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0, 2), r: IIC number (r = 00, 10), mn = 00, 02

**2.** ☐: Setting is fixed in the IIC 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 usage of the user

**Figure 18-110. Example of Contents of Registers for Data Reception of Simplified $I^2C$ (IIC00, IIC10) (2/2)**

**(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<br>× | SSm2<br>0/1 | SSm1<br>× | SSm0<br>0/1 |

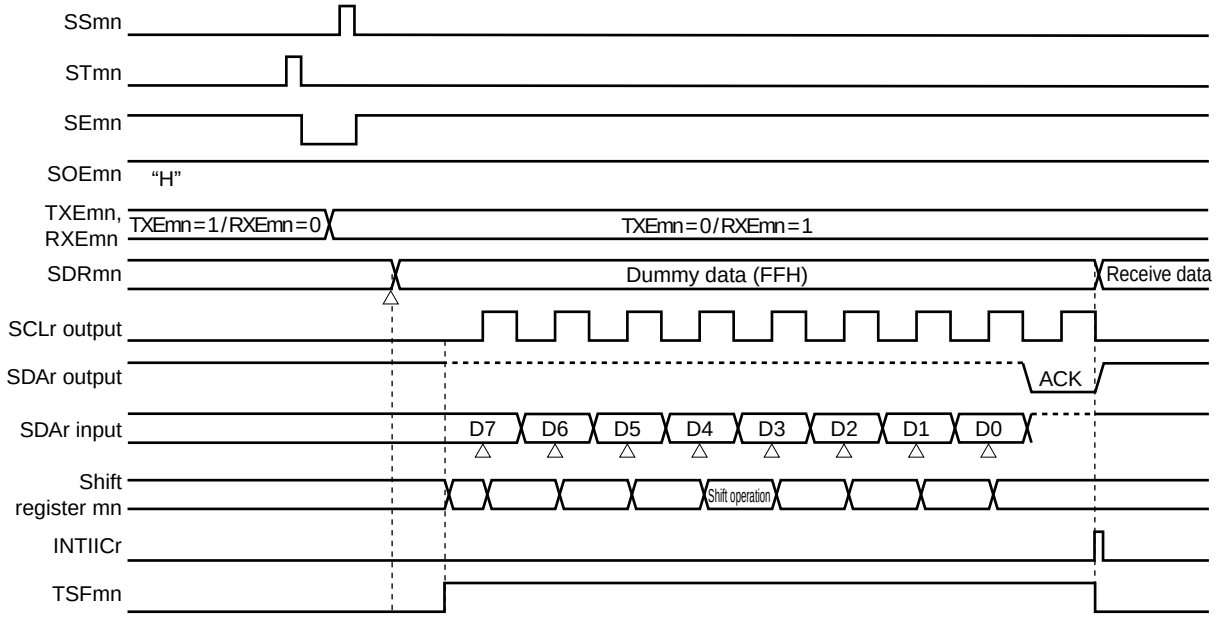
**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0, 2), r: IIC number (r = 00, 10), mn = 00, 02

**2.** ☐: Setting is fixed in the CSI master 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 usage of the user

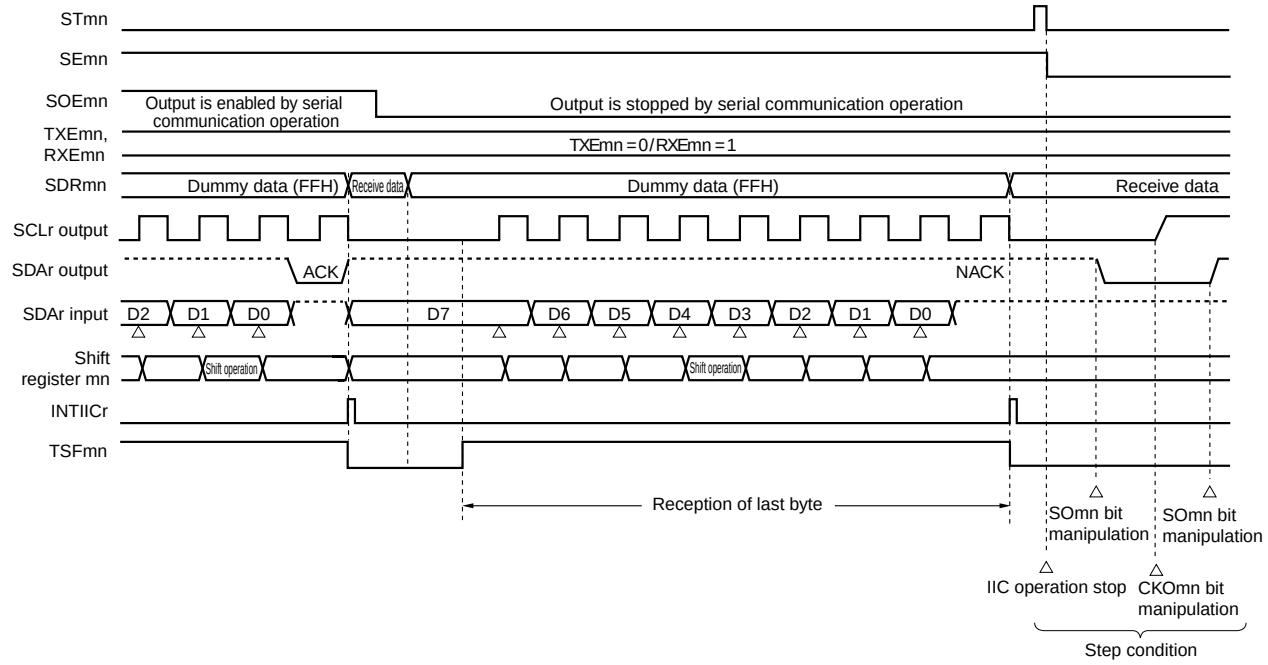
### (2) Processing flow

**Figure 18-111. Timing Chart of Data Reception**

#### (a) When starting data reception

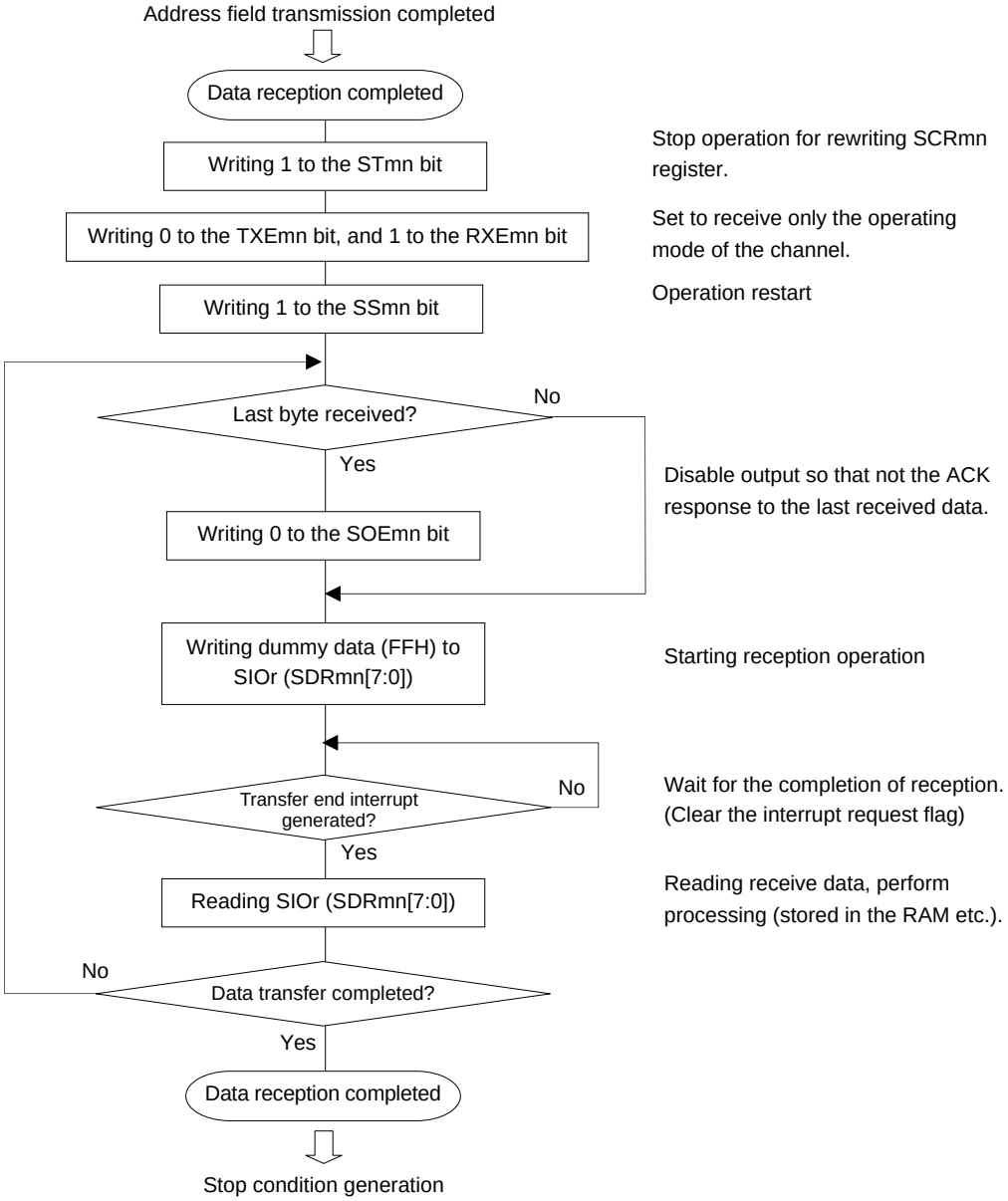



#### (b) When receiving last data




**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2), r: IIC number (r = 00, 10), mn = 00, 02

**Figure 18-112. Flowchart of Data Reception**



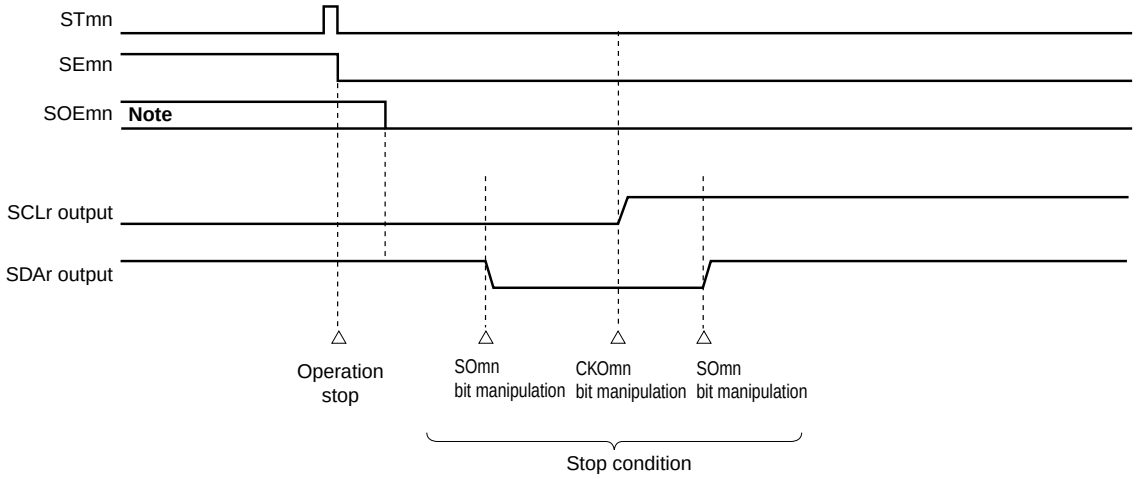

**Caution ACK is not output when the last data is received (NACK). Communication is then completed by setting "1" to the STmn bit of serial channel stop register m (STm) to stop operation and generating a stop condition.**

### 18.8.4 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.

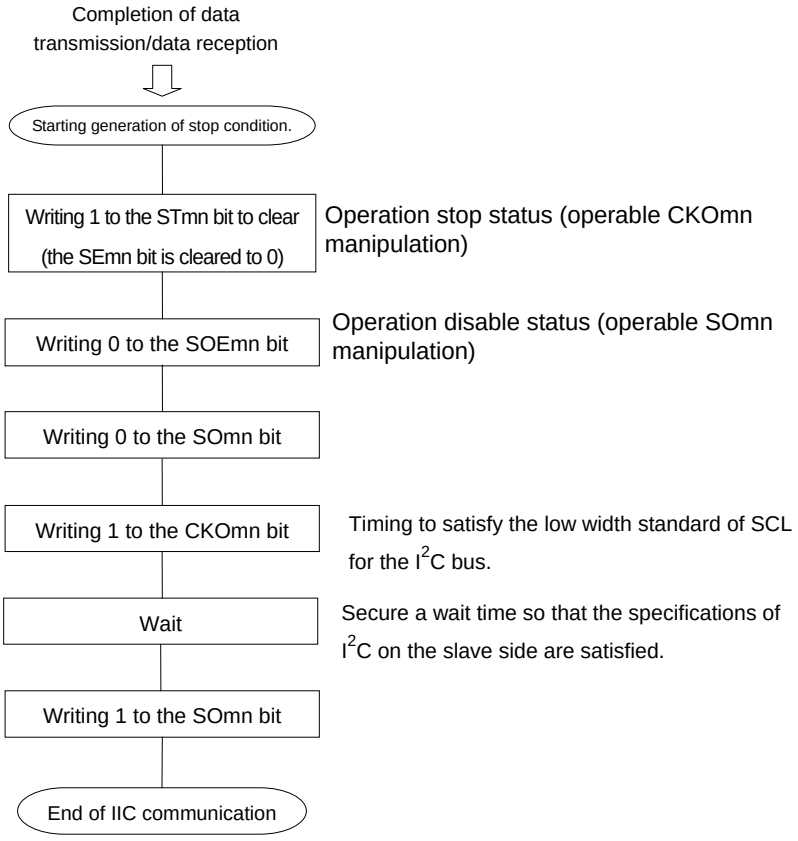
#### (1) Processing flow

**Figure 18-113. 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 18-114. Flowchart of Stop Condition Generation**



### 18.8.5 Calculating transfer rate

The transfer rate for simplified $I^2C$ (IIC00, IIC10) communication can be calculated by the following expressions.

(Transfer rate) = {Operation clock ($f_{MCK}$) frequency of target channel} ÷ (SDRmn[15:9] + 1) ÷ 2

**Caution SDRmn[15:9] must not be set to 00000000B. Be sure to set a value of 00000001B or greater for SDRmn[15:9]. The duty ratio of the SCL signal output by the simplified $I^2C$ is 50%. The $I^2C$ 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 $I^2C$ bus specifications. Make sure that the SDRmn[15:9] value satisfies the $I^2C$ bus specifications.**

**Remarks 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.

**2.** m: Unit number (m = 0), n: Channel number (n = 0, 2), mn = 00, 02

The operation clock ($f_{MCK}$) is determined by serial clock select register m (SPSm) and bit 15 (CKSmn) of serial mode register mn (SMRmn).

**Table 18-5. Selection of Operation Clock For Simplified $I^2C$**

| 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 | | $f_{CLK}$ = 24 MHz |
| 0 | X | X | X | X | 0 | 0 | 0 | 0 | $f_{CLK}$ | 24 MHz |
| | X | X | X | X | 0 | 0 | 0 | 1 | $f_{CLK}/2$ | 12 MHz |
| | X | X | X | X | 0 | 0 | 1 | 0 | $f_{CLK}/2^2$ | 6 MHz |
| | X | X | X | X | 0 | 0 | 1 | 1 | $f_{CLK}/2^3$ | 3 MHz |
| | X | X | X | X | 0 | 1 | 0 | 0 | $f_{CLK}/2^4$ | 1.5 MHz |
| | X | X | X | X | 0 | 1 | 0 | 1 | $f_{CLK}/2^5$ | 750 kHz |
| | X | X | X | X | 0 | 1 | 1 | 0 | $f_{CLK}/2^6$ | 375 kHz |
| | X | X | X | X | 0 | 1 | 1 | 1 | $f_{CLK}/2^7$ | 187.5 kHz |
| | X | X | X | X | 1 | 0 | 0 | 0 | $f_{CLK}/2^8$ | 93.8 kHz |
| | X | X | X | X | 1 | 0 | 0 | 1 | $f_{CLK}/2^9$ | 46.9 kHz |
| | X | X | X | X | 1 | 0 | 1 | 0 | $f_{CLK}/2^{10}$ | 23.4 kHz |
| | X | X | X | X | 1 | 0 | 1 | 1 | $f_{CLK}/2^{11}$ | 11.7 kHz |
| 1 | 0 | 0 | 0 | 0 | X | X | X | X | $f_{CLK}$ | 24 MHz |
| | 0 | 0 | 0 | 1 | X | X | X | X | $f_{CLK}/2$ | 12 MHz |
| | 0 | 0 | 1 | 0 | X | X | X | X | $f_{CLK}/2^2$ | 6 MHz |
| | 0 | 0 | 1 | 1 | X | X | X | X | $f_{CLK}/2^3$ | 3 MHz |
| | 0 | 1 | 0 | 0 | X | X | X | X | $f_{CLK}/2^4$ | 1.5 MHz |
| | 0 | 1 | 0 | 1 | X | X | X | X | $f_{CLK}/2^5$ | 750 kHz |
| | 0 | 1 | 1 | 0 | X | X | X | X | $f_{CLK}/2^6$ | 375 kHz |
| | 0 | 1 | 1 | 1 | X | X | X | X | $f_{CLK}/2^7$ | 187.5 kHz |
| | 1 | 0 | 0 | 0 | X | X | X | X | $f_{CLK}/2^8$ | 93.8 kHz |
| | 1 | 0 | 0 | 1 | X | X | X | X | $f_{CLK}/2^9$ | 46.9 kHz |
| | 1 | 0 | 1 | 0 | X | X | X | X | $f_{CLK}/2^{10}$ | 23.4 kHz |
| | 1 | 0 | 1 | 1 | X | X | X | X | $f_{CLK}/2^{11}$ | 11.7 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).

**Remarks 1.** X: Don't care

**2.** m: Unit number (m = 0), n: Channel number (n = 0, 2), mn = 00, 02

Here is an example of setting an $I^2C$ transfer rate where $f_{MCK} = f_{CLK}$ = 24 MHz.

| $I^2C$ Transfer Mode (Desired Transfer Rate) | $f_{CLK}$ = 24 MHz | | | |
|---|---|---|---|---|
| | Operation Clock ($f_{MCK}$) | SDRmn[15:9] | Calculated Transfer Rate | Error from Desired Transfer Rate |
| 100 kHz | $f_{CLK}/2$ | 59 | 100 kHz | 0.0% |
| 400 kHz | $f_{CLK}$ | 29 | 380 kHz | 5.0%[Note] |
| 1 MHz | $f_{CLK}$ | 5 | 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%.

### 18.8.6 Procedure for processing errors that occurred during simplified I²C (IIC00, IIC10) communication

The procedure for processing errors that occurred during simplified I²C (IIC00, IIC10) communication is described in Figures 18-115 and 18-116.

**Figure 18-115. Processing Procedure in Case of Overrun Error**

| Software Manipulation | Hardware Status | 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. |

**Figure 18-116. Processing Procedure in Case of ACK Error in Simplified I²C Mode**

| Software Manipulation | Hardware Status | Remark |
|---|---|---|
| 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. |
| 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), n: Channel number (n = 0, 2), r: IIC number (r = 00, 10), mn = 00, 02

# CHAPTER 19 SERIAL INTERFACE IICA

## 19.1 Functions of Serial Interface IICA

Serial interface IICA has the following three modes.

**(1) Operation stop mode**

This mode is used when serial transfers are not performed. It can therefore be used to reduce power consumption.

**(2) $I^2C$ bus mode (multimaster 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 generated “start condition”, “address”, “transfer direction specification”, “data”, and “stop condition” data to the slave device, via the serial data bus. The slave device automatically detects these received status 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 a local address has been received while in STOP mode. This can be set by using the WUPn bit of IICA control register n1 (IICCTLn1).

Figure 19-1 shows a block diagram of serial interface IICA.

**Remark** n = 0

**Figure 19-1. Block Diagram of Serial Interface IICA0**

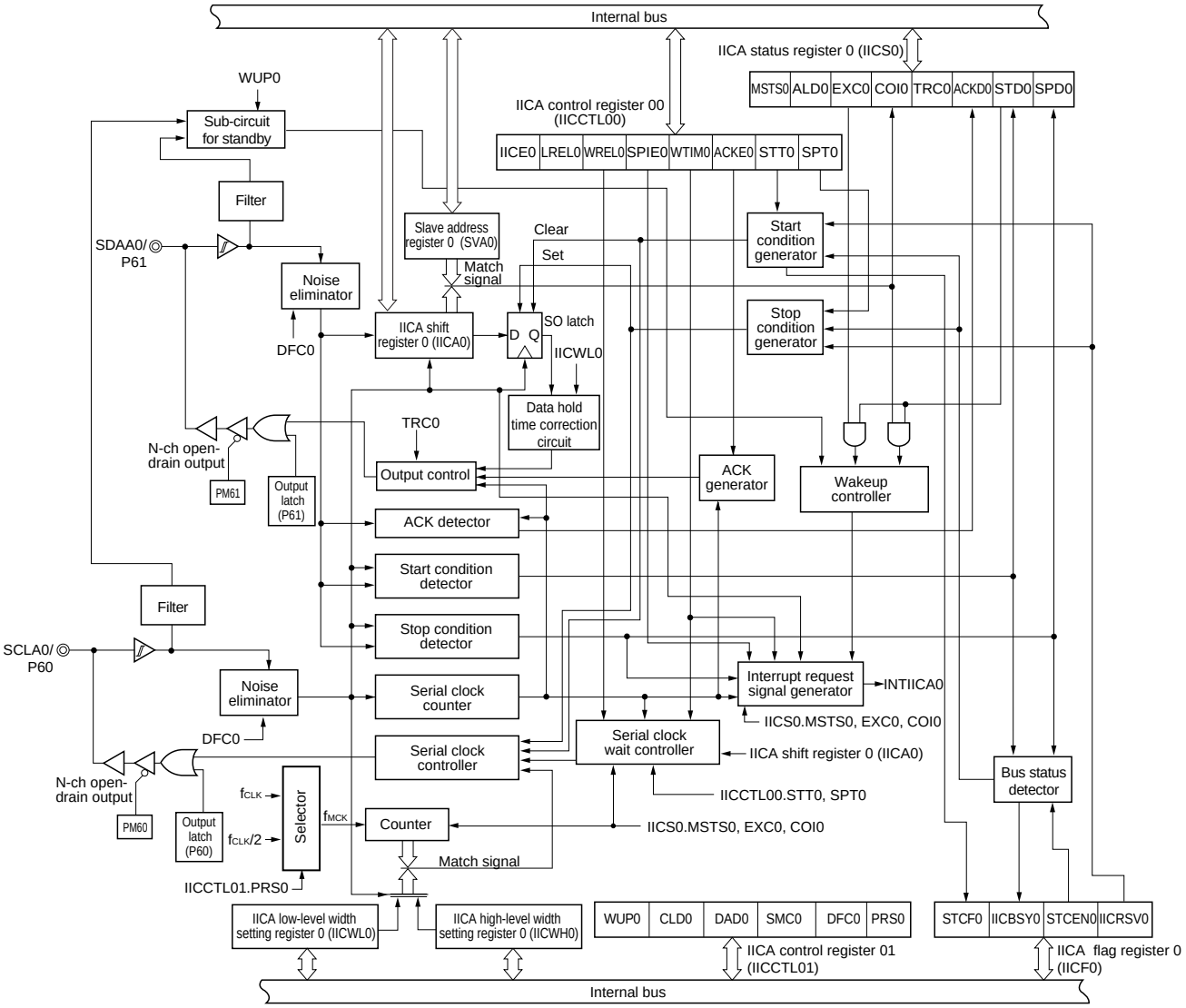
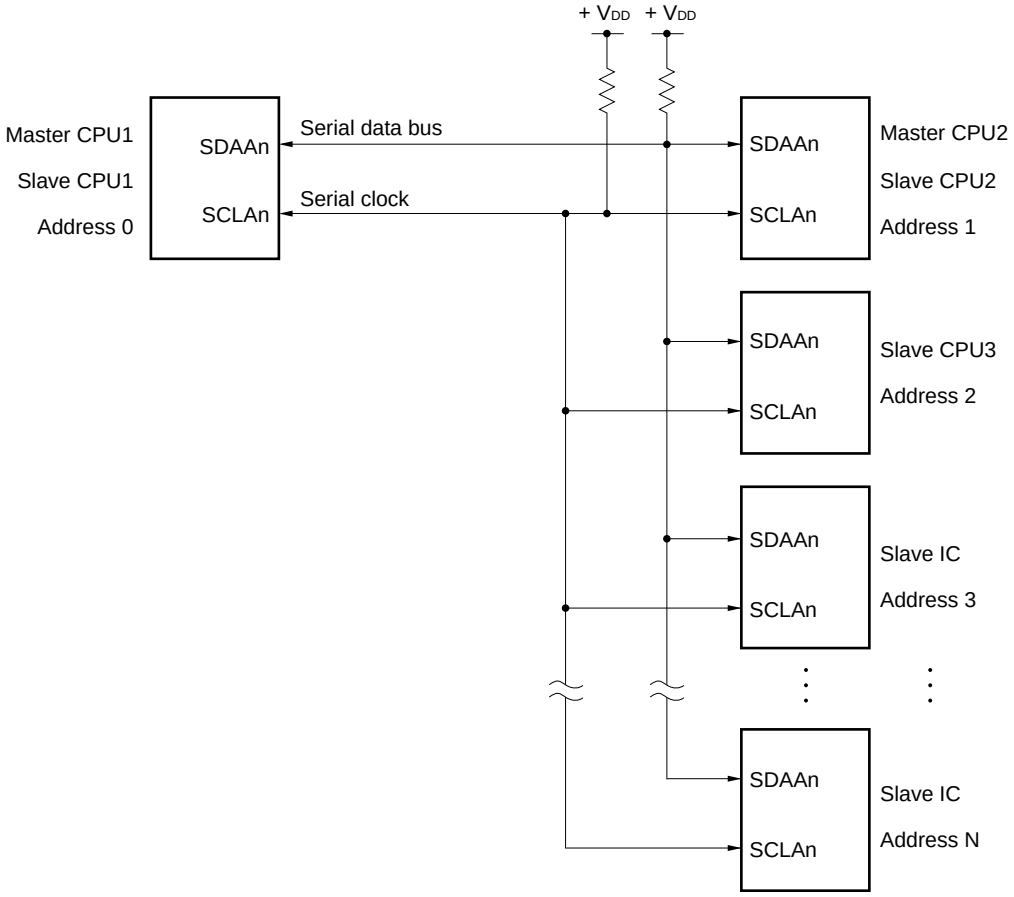


Figure 19-2 shows a serial bus configuration example.

**Figure 19-2. Serial Bus Configuration Example Using I²C Bus**




**Remark** n = 0

## 19.2 Configuration of Serial Interface IICA

Serial interface IICA includes the following hardware.

**Table 19-1. Configuration of Serial Interface IICA**

| Item | Configuration |
|---|---|
| Registers | IICA shift register n (IICAn)<br>Slave address register n (SVAn) |
| Control registers | Peripheral enable register 0 (PER0)<br>IICA control register n0 (IICCTLn0)<br>IICA status register n (IICSn)<br>IICA flag register n (IICFn)<br>IICA control register n1 (IICCTLn1)<br>IICA low-level width setting register n (IICWLn)<br>IICA high-level width setting register n (IICWHn)<br>Port mode register 6 (PM6)<br>Port register 6 (P6) |

**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 and reading operations to the IICAn register.

Cancel the wait state and start data transfer by writing data to the IICAn register during the wait period.

The IICAn register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears IICAn to 00H.

**Figure 19-3. Format of IICA Shift Register n (IICAn)**

Address: FFF50H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| IICAn | | | | | | | | |

**Cautions 1. Do not write data to the IICAn register during data transfer.**

**2. Write or read the IICAn register only during the wait period. Accessing the IICAn register in a communication state other than during the wait period is prohibited. When the device serves as the master, however, the IICAn register can be written only once after the communication trigger bit (STTn) is set to 1.**

**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 stores seven bits of local addresses {A6, A5, A4, A3, A2, A1, A0} 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).

Reset signal generation clears the SVAn register to 00H.

**Figure 19-4. Format of Slave Address Register n (SVAn)**

Address: F0234H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| SVAn | A6 | A5 | A4 | A3 | A2 | A1 | A0 | 0Note |

**Note** Bit 0 is fixed to 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 (INTIICAn) when the address received by this register matches the address value set to the slave address register n (SVAn) or when an extension code is received.

**(5) Serial clock counter**

This counter counts the serial clocks 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 WTIMn bit)
- Interrupt request generated when a stop condition is detected (set by the SPIEn bit)

**Remark** WTIMn bit: Bit 3 of IICA control register n0 (IICCTLn0)
SPIEn bit: Bit 4 of IICA control register n0 (IICCTLn0)

**(7) Serial clock controller**

In master mode, this circuit generates the clock output via the SCLAn pin from a sampling clock.

**(8) Serial clock wait controller**

This circuit controls the wait timing.

**(9) ACK generator, stop condition detector, start condition detector, and ACK detector**

These circuits generate and detect each status.

**(10) Data hold time correction circuit**

This circuit generates the hold time for data corresponding to the falling edge of the serial clock.

**Remark** n = 0

**(11) Start condition generator**

This circuit generates a start condition when the STTn bit is set to 1.

However, in the communication reservation disabled status (IICRSVn bit = 1), when the bus is not released (IICBSYn bit = 1), start condition requests are ignored and the STCFn bit is set to 1.

**(12) Stop condition generator**

This circuit generates a stop condition when the SPTn bit is set to 1.

**(13) Bus status detector**

This circuit detects whether or not the bus is released by detecting start conditions and stop conditions.

However, as the bus status cannot be detected immediately following operation, the initial status is set by the STCENn bit.

**Remarks 1.** STTn bit: Bit 1 of IICA control register n0 (IICCTLn0)
SPTn bit: Bit 0 of IICA control register n0 (IICCTLn0)
IICRSVn bit: Bit 0 of IICA flag register n (IICFn)
IICBSYn bit: Bit 6 of IICA flag register n (IICFn)
STCFn bit: Bit 7 of IICA flag register n (IICFn)
STCENn bit: Bit 1 of IICA flag register n (IICFn)

**2.** n = 0

## 19.3 Registers Controlling Serial Interface IICA

Serial interface IICA is controlled by the following eight registers.

- Peripheral enable register 0 (PER0)
- 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)
- Port mode register 6 (PM6)
- Port register 6 (P6)

**Remark** n = 0

### 19.3.1 Peripheral enable register 0 (PER0)

This register is used to enable or disable supplying the clock to the peripheral hardware. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

When serial interface IICAn is used, be sure to set bit 4 (IICA0EN) of this register to 1.

The PER0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 19-5. Format of Peripheral Enable Register 0 (PER0)**

Address: F00F0H After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PER0 | RTCWEN | IRDAEN | ADCEN | IICA0EN | SAU1EN | SAU0EN | 0 | TAU0EN |

| IICAnEN | Control of serial interface IICAn input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFR used by serial interface IICAn cannot be written.<br>• Serial interface IICAn is in the reset status. |
| 1 | Enables input clock supply.<br>• SFR used by serial interface IICAn can be read/written. |

**Cautions 1. When setting serial interface IICAn, be sure to set the following registers first while the IICAnEN bit is set to 1. If IICAnEN = 0, the control registers of serial interface IICA are set to their initial values, and writing to them is ignored (except for port mode register 6 (PM6) and port register 6 (P6)).**

- **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)**

**2. Be sure to clear bit 1 to “0”.**

**Remark** n = 0

### 19.3.2 IICA control register n0 (IICCTLn0)

This register is used to enable/stop $I^2C$ operations, set wait timing, and set other $I^2C$ operations.

The IICCTLn0 register can be set by a 1-bit or 8-bit memory manipulation instruction. However, set the SPIEn, WTIMn, and ACKEn bits while IICEn = 0 or during the wait period. These bits can be set at the same time when the IICEn bit is set from “0” to “1”.

Reset signal generation clears this register to 00H.

**Remark** n = 0

**Figure 19-6. Format of IICA Control Register n0 (IICCTLn0) (1/4)**

Address: F0230H After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| IICCTLn0 | IICEn | LRELn | WRELn | SPIEn | WTIMn | ACKEn | STTn | SPTn |

| IICEn | $I^2C$ 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 (1) while the SCLAn and SDAAn lines are at high level. | |
| Condition for clearing (IICEn = 0) | Condition for setting (IICEn = 1) |
| • Cleared by instruction<br>• Reset | • Set by instruction |

| LRELn[Notes 2, 3] | Exit from communications |
|---|---|
| 0 | Normal operation |
| 1 | This exits from the current communications and sets standby mode. This setting is automatically cleared to 0 after being executed.<br>Its uses include cases in which a locally irrelevant extension code has been received.<br>The SCLAn and SDAAn lines are set to high impedance.<br>The following flags of IICA control register n0 (IICCTLn0) and the IICA status register n (IICSn) are cleared to 0.<br>• STTn • SPTn • MSTSn • EXCn • COIn • TRCn • ACKDn • STDn |
| The standby mode following exit from communications remains in effect until the following communications entry conditions are met.<br>• After a stop condition is detected, restart is in master mode.<br>• An address match or extension code reception occurs after the start condition. | |
| Condition for clearing (LRELn = 0) | Condition for setting (LRELn = 1) |
| • Automatically cleared after execution<br>• Reset | • Set by instruction |

| WRELn[Notes 2, 3] | Wait cancellation |
|---|---|
| 0 | Do not cancel wait |
| 1 | Cancel wait. This setting is automatically cleared after wait is canceled. |
| When the WRELn bit is set (wait canceled) during the wait period at the ninth clock pulse in the transmission status (TRCn = 1), the SDAAn line goes into the high impedance state (TRCn = 0). | |
| Condition for clearing (WRELn = 0) | Condition for setting (WRELn = 1) |
| • Automatically cleared after execution<br>• Reset | • Set by instruction |

**Notes 1.** The IICA status register n (IICSn), the STCFn and IICBSYn bits of the IICA flag register n (IICFn), and the CLDn and DADn bits of IICA control register n1 (IICCTLn1) are reset.

**2.** The signal of this bit is invalid while IICEn is 0.

**3.** When the LRELn and WRELn bits are read, 0 is always read.

**Caution 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 (DFCn bit of IICCTLn1 register = 1), a start condition will be inadvertently detected immediately. In this case, set (1) the LRELn bit by using a 1-bit memory manipulation instruction immediately after enabling operation of $I^2C$ (IICEn = 1).**

**Remark** n = 0

**Figure 19-6. Format of IICA Control Register n0 (IICCTLn0) (2/4)**

| SPIEn[Note 1] | Enable/disable generation of interrupt request when stop condition is detected |
|---|---|
| 0 | Disable |
| 1 | Enable |
| If the WUPn bit of IICA control register n1 (IICCTLn1) is 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<br>• Reset | • Set by instruction |

| WTIMn[Note 1] | Control of wait and interrupt request generation |
|---|---|
| 0 | Interrupt request is generated at the eighth clock's falling edge.<br>Master mode: After output of eight clocks, clock output is set to low level and wait is set.<br>Slave mode: After input of eight clocks, the clock is set to low level and wait is set for master device. |
| 1 | Interrupt request is generated at the ninth clock's falling edge.<br>Master mode: After output of nine clocks, clock output is set to low level and wait is set.<br>Slave mode: After input of nine clocks, the clock is set to low level and wait is set for master device. |
| An interrupt is generated at the falling edge of the ninth clock during address transfer independently of the setting of this bit. The setting of this bit is valid when the address transfer is completed. When in master mode, a wait is inserted at the falling edge of the ninth clock during address transfers. For a slave device that has received a local address, a wait is inserted at the falling edge of the ninth clock after an acknowledge (ACK) is issued. However, when the slave device has received an extension code, a wait is inserted at the falling edge of the eighth clock. | |
| Condition for clearing (WTIMn = 0) | Condition for setting (WTIMn = 1) |
| • Cleared by instruction<br>• Reset | • Set by instruction |

| ACKEn[Notes 1, 2] | 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<br>• Reset | • Set by instruction |

**Notes 1.** The signal of this bit is invalid while IICEn is 0. Set this bit during that period.

**2.** The set value is invalid during address transfer and if the code is not an extension code.
When the device serves as a slave and the addresses match, an acknowledgment is generated regardless of the set value.

**Remark** n = 0

**Figure 19-6. Format of IICA Control Register n0 (IICCTLn0) (3/4)**

| STTn[Notes 1, 2] | Start condition trigger |
|---|---|
| 0 | Do not generate a start condition. |
| 1 | When bus is released (in standby state, when IICBSYn = 0):<br>If this bit is set (1), a start condition is generated (startup as the master).<br>When a third party is communicating:<br>• When communication reservation function is enabled (IICRSVn = 0)<br>Functions as the start condition reservation flag. When set to 1, automatically generates a start condition after the bus is released.<br>• When communication reservation function is disabled (IICRSVn = 1)<br>Even if this bit is set (1), the STTn bit is cleared and the STTn clear flag (STCFn) is set (1). No start condition is generated.<br>In the wait state (when master device):<br>Generates a restart condition after releasing the wait. |

Cautions concerning set timing

- For master reception: Cannot be set to 1 during transfer. Can be set to 1 only in the waiting 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 wait 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 (1), setting it again (1) 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.<br>• Cleared by loss in arbitration<br>• Cleared after start condition is generated by master device<br>• Cleared by LRELn = 1 (exit from communications)<br>• When IICEn = 0 (operation stop)<br>• Reset | • Set by instruction |

**Notes 1.** The signal of this bit is invalid while IICEn is 0.

**2.** The STTn bit is always read as 0.

**Remarks 1.** IICRSVn: Bit 0 of IIC flag register n (IICFn)
STCFn: Bit 7 of IIC flag register n (IICFn)

**2.** n = 0

**Figure 19-6. Format of IICA Control Register n0 (IICCTLn0) (4/4)**

| SPTn[Note] | 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 in the waiting 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 wait 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 wait period that follows output of eight clocks, note that a stop condition will be generated during the high-level period of the ninth clock. The WTIMn bit should be changed from 0 to 1 during the wait period following the output of eight clocks, and the SPTn bit should be set to 1 during the wait period that follows the output of the ninth clock.
- Once SPTn is set (1), setting it again (1) before the clear condition is met is not allowed.

| Condition for clearing (SPTn = 0) | Condition for setting (SPTn = 1) |
|---|---|
| • Cleared by loss in arbitration<br>• Automatically cleared after stop condition is detected<br>• Cleared by LRELn = 1 (exit from communications)<br>• When IICEn = 0 (operation stop)<br>• Reset | • Set by instruction |

**Note** The SPTn bit is always read as 0.

**Caution When bit 3 (TRCn) of the IICA status register n (IICSn) is set to 1 (transmission status), bit 5 (WRELn) of IICA control register n0 (IICCTLn0) is set to 1 during the ninth clock and wait is canceled, after which the TRCn bit is cleared (reception status) and the SDAAn line is set to high impedance. Release the wait performed while the TRCn bit is 1 (transmission status) by writing to the IICA shift register n.**

**Remark** n = 0

### 19.3.3 IICA status register n (IICSn)

This register indicates the status of $I^2C$.

The IICSn register is read by a 1-bit or 8-bit memory manipulation instruction only when STTn = 1 and during the wait period.

Reset signal generation clears this register to 00H.

**Caution Reading the IICSn register while the address match wakeup function is enabled (WUPn = 1) in STOP mode is prohibited. When the WUPn bit is changed from 1 to 0 (wakeup operation is stopped), regardless of the INTIICAn interrupt request, the change in status is not reflected until the next start condition or stop condition is detected. To use the wakeup function, therefore, enable (SPIEn = 1) the interrupt generated by detecting a stop condition and read the IICSn register after the interrupt has been detected.**

**Remark** STTn: bit 1 of IICA control register n0 (IICCTLn0)
WUPn: bit 7 of IICA control register n1 (IICCTLn1)

**Figure 19-7. Format of IICA Status Register n (IICSn) (1/3)**

Address: FFF51H After reset: 00H 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 device status or communication standby status |
| 1 | Master device communication status |

| Condition for clearing (MSTSn = 0) | Condition for setting (MSTSn = 1) |
|---|---|
| • When a stop condition is detected<br>• When ALDn = 1 (arbitration loss)<br>• Cleared by LRELn = 1 (exit from communications)<br>• When the IICEn bit changes from 1 to 0 (operation stop)<br>• Reset | • When a start condition is generated |

| ALDn | Detection of arbitration loss |
|---|---|
| 0 | This status means either that there was no arbitration or that the arbitration result was a “win”. |
| 1 | This status indicates the arbitration result was a “loss”. The MSTSn bit is cleared. |

| Condition for clearing (ALDn = 0) | Condition for setting (ALDn = 1) |
|---|---|
| • Automatically cleared after the IICSn register is read[Note]<br>• When the IICEn bit changes from 1 to 0 (operation stop)<br>• Reset | • When the arbitration result is a “loss”. |

**Note** This register is also cleared when a 1-bit memory manipulation instruction is executed for bits other than the IICSn register. Therefore, when using the ALDn bit, read the data of this bit before the data of the other bits.

**Remarks 1.** LRELn: Bit 6 of IICA control register n0 (IICCTLn0)
IICEn: Bit 7 of IICA control register n0 (IICCTLn0)
**2.** n = 0

**Figure 19-7. Format of IICA Status Register n (IICSn) (2/3)**

| EXCn | Detection of extension code reception |
|---|---|
| 0 | Extension code was not received. |
| 1 | Extension code was received. |

| Condition for clearing (EXCn = 0) | Condition for setting (EXCn = 1) |
|---|---|
| • When a start condition is detected<br>• When a stop condition is detected<br>• Cleared by LRELn = 1 (exit from communications)<br>• When the IICEn bit changes from 1 to 0 (operation stop)<br>• 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). |

| COIn | Detection of matching addresses |
|---|---|
| 0 | Addresses do not match. |
| 1 | Addresses match. |

| Condition for clearing (COIn = 0) | Condition for setting (COIn = 1) |
|---|---|
| • When a start condition is detected<br>• When a stop condition is detected<br>• Cleared by LRELn = 1 (exit from communications)<br>• When the IICEn bit changes from 1 to 0 (operation stop)<br>• Reset | • When the received address matches the local address (slave address register n (SVAn)) (set at the rising edge of the eighth clock). |

| TRCn | Detection of transmit/receive status |
|---|---|
| 0 | Receive status (other than transmit status). The SDAAn line is set for high impedance. |
| 1 | Transmit status. 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><br>• When a stop condition is detected<br>• Cleared by LRELn = 1 (exit from communications)<br>• When the IICEn bit changes from 1 to 0 (operation stop)<br>• Cleared by WRELn = 1[Note] (wait cancel)<br>• When the ALDn bit changes from 0 to 1 (arbitration loss)<br>• Reset<br>• When not used for communication (MSTSn, EXCn, COIn = 0)<br><Master><br>• When “1” is output to the first byte’s LSB (transfer direction specification bit)<br><Slave><br>• When a start condition is detected<br>• When “0” is input to the first byte’s LSB (transfer direction specification bit) | <Master><br>• When a start condition is generated<br>• When 0 (master transmission) is output to the LSB (transfer direction specification bit) of the first byte (during address transfer)<br><Slave><br>• When 1 (slave transmission) is input to the LSB (transfer direction specification bit) of the first byte from the master (during address transfer) |

**Note** When bit 3 (TRCn) of the IICA status register n (IICSn) is set to 1 (transmission status), bit 5 (WRELn) of IICA control register n0 (IICCTLn0) is set to 1 during the ninth clock and wait is canceled, after which the TRCn bit is cleared (reception status) and the SDAAn line is set to high impedance. Release the wait performed while the TRCn bit is 1 (transmission status) by writing to the IICA shift register n.

**Remarks 1.** LRELn: Bit 6 of IICA control register n0 (IICCTLn0)
IICEn: Bit 7 of IICA control register n0 (IICCTLn0)
**2.** n = 0

**Figure 19-7. Format of IICA Status Register n (IICSn) (3/3)**

| 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<br>• At the rising edge of the next byte's first clock<br>• Cleared by LRELn = 1 (exit from communications)<br>• When the IICEn bit changes from 1 to 0 (operation stop)<br>• Reset | • After the SDAAn line is set to low level at the rising edge of SCLAn line's ninth clock |

| 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<br>• At the rising edge of the next byte's first clock following address transfer<br>• Cleared by LRELn = 1 (exit from communications)<br>• When the IICEn bit changes from 1 to 0 (operation stop)<br>• 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<br>• When the WUPn bit changes from 1 to 0<br>• When the IICEn bit changes from 1 to 0 (operation stop)<br>• Reset | • When a stop condition is detected |

**Remarks 1.** LRELn: Bit 6 of IICA control register n0 (IICCTLn0)
IICEn: Bit 7 of IICA control register n0 (IICCTLn0)

**2.** n = 0

### 19.3.4 IICA flag register n (IICFn)

This register sets the operation mode of $I^2C$ and indicates the status of the $I^2C$ bus.

The IICFn register can be set by a 1-bit or 8-bit memory manipulation instruction. However, the STTn clear flag (STCFn) and $I^2C$ bus status flag (IICBSYn) bits are read-only.

The IICRSVn bit can be used to enable/disable the communication reservation function.

The STCENn bit can be used to set the initial value of the IICBSYn bit.

The IICRSVn and STCENn bits can be written only when the operation of $I^2C$ is disabled (bit 7 (IICEn) of IICA control register n0 (IICCTLn0) = 0). When operation is enabled, the IICFn register can be read.

Reset signal generation clears this register to 00H.

**Figure 19-8. Format of IICA Flag Register n (IICFn)**

Address: FFF52H After reset: 00H 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 STTn = 1<br>• When IICEn = 0 (operation stop)<br>• Reset | • Generating start condition unsuccessful and the STTn bit cleared to 0 when communication reservation is disabled (IICRSVn = 1). |

| IICBSYn | $I^2C$ bus status flag |
|---|---|
| 0 | Bus release status (communication initial status when STCENn = 1) |
| 1 | Bus communication status (communication initial status when STCENn = 0) |

| Condition for clearing (IICBSYn = 0) | Condition for setting (IICBSYn = 1) |
|---|---|
| • Detection of stop condition<br>• When IICEn = 0 (operation stop)<br>• Reset | • Detection of start condition<br>• Setting of the IICEn bit when STCENn = 0 |

| STCENn | Initial start enable trigger |
|---|---|
| 0 | After operation is enabled (IICEn = 1), enable generation of a start condition upon detection of a stop condition. |
| 1 | After operation is enabled (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<br>• Detection of start condition<br>• Reset | • Set by instruction |

| 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<br>• Reset | • Set by instruction |

**Note** Bits 6 and 7 are read-only.

**Cautions 1. Write to the STCENn bit only when the operation is stopped (IICEn = 0).**

**2. As the bus release status (IICBSYn = 0) is recognized regardless of the actual bus status when STCENn = 1, when generating the first start condition (STTn = 1), it is necessary to verify that no third party communications are in progress in order to prevent such communications from being destroyed.**

**3. Write to IICRSVn only when the operation is stopped (IICEn = 0).**

**Remarks 1.** STTn: Bit 1 of IICA control register n0 (IICCTLn0)
IICEn: Bit 7 of IICA control register n0 (IICCTLn0)

**2.** n = 0

### 19.3.5 IICA control register n1 (IICCTLn1)

This register is used to set the operation mode of $I^2C$ and detect the statuses 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 $I^2C$ is disabled (bit 7 (IICEn) of IICA control register n0 (IICCTLn0) is 0).

Reset signal generation clears this register to 00H.

**Figure 19-9. Format of IICA Control Register n1 (IICCTLn1) (1/2)**

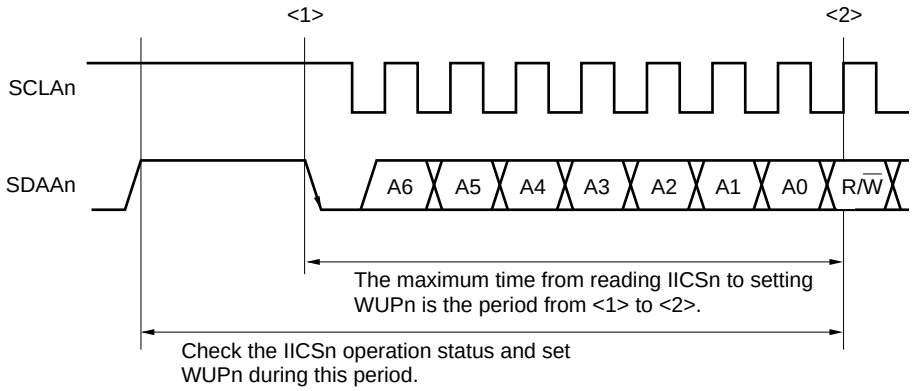
Address: F0231H After reset: 00H R/W[Note 1]

| Symbol | <7> | 6 | <5> | <4> | <3> | <2> | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| IICCTLn1 | WUPn | 0 | 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 clocks of $f_{MCK}$ after setting (1) the WUPn bit (see **Figure 19-22 Flow When Setting WUPn = 1**).<br>Clear (0) the WUPn bit after the address has matched or an extension code has been received. The subsequent communication can be entered by the clearing (0) WUPn bit. (The wait must be released and transmit data must be written after the WUPn bit has been cleared (0).)<br>The interrupt timing when the address has matched 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 the SPIEn bit is set to 1. | |
| Condition for clearing (WUPn = 0) | Condition for setting (WUPn = 1) |
| • Cleared by instruction (after address match 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] |

**Notes 1.** Bits 4 and 5 are read-only.

**2.** The status of the IICA status register n (IICSn) must be checked and the WUPn bit must be set during the period shown below.




**Remark** n = 0

**Figure 19-9. Format of IICA Control Register n1 (IICCTLn1) (2/2)**

| CLDn | Detection of SCLAn pin level (valid only when 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<br>• When IICEn = 0 (operation stop)<br>• Reset | • When the SCLAn pin is at high level |

| DADn | Detection of SDAAn pin level (valid only when 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<br>• When IICEn = 0 (operation stop)<br>• 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. |
| Digital filter can be used only in fast mode and fast mode plus.<br>The digital filter is used for noise elimination. The transfer clock does not vary, regardless of the DFCn bit being set (1) or cleared (0). | |

| PRSn | IICA operation clock ($f_{MCK}$) control |
|---|---|
| 0 | Selects $f_{CLK}$ (1 MHz ≤ $f_{CLK}$ ≤ 20 MHz) |
| 1 | Selects $f_{CLK}/2$ (20 MHz < $f_{CLK}$) |

**Cautions 1. The maximum operating frequency of the IICA operating clock ($f_{MCK}$) is 20 MHz (Max.). Set the IICA control register n1 (IICCTLn1) bit 0 (PRSn) to "1" only when $f_{CLK}$ exceeds 20 MHz.**

**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.)**

**3. The fast mode plus is only available in the products for "A: Consumer applications ($T_A$ = –40°C to +85°C)" and "D: Industrial applications ($T_A$ = –40°C to +85°C)".**

**Remarks 1.** IICEn: Bit 7 of IICA control register n0 (IICCTLn0)

**2.** n = 0

### 19.3.6 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 (bit 7 (IICEn) of IICA control register n0 (IICCTLn0) is 0).

Reset signal generation sets this register to FFH.

For details about setting the IICWLn register, see **19.4.2 Setting transfer clock by using IICWLn and IICWHn registers**.

<R> The data hold time is one-quarter of the time set by the IICWLn register.

**Figure 19-10. Format of IICA Low-Level Width Setting Register n (IICWLn)**

Address: F0232H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| IICWLn | | | | | | | | |

### 19.3.7 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 (bit 7 (IICEn) of IICA control register n0 (IICCTLn0) is 0).

Reset signal generation sets this register to FFH.

**Figure 19-11. Format of IICA High-Level Width Setting Register n (IICWHn)**

Address: F0233H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| IICWHn | | | | | | | | |

**Remarks 1.** For setting procedures of the transfer clock on master side and of the IICWLn and IICWHn registers on slave side, see **19.4.2 (1)** and **19.4.2 (2)**, respectively.

**2.** n = 0

### 19.3.8 Port mode register 6 (PM6)

This register sets the input/output of port 6 in 1-bit units.

When using the P60/SCLA0 pin as clock I/O and the P61/SDAA0 pin as serial data I/O, clear PM60 and PM61, and the output latches of P60 and P61 to 0.

Set the IICEn bit (bit 7 of IICA control register n0 (IICCTLn0)) to 1 before setting the output mode because the P60/SCLA0 and P61/SDAA0 pins output a low level (fixed) when the IICEn bit is 0.

The PM6 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to FFH.

**Figure 19-12. Format of Port Mode Register 6 (PM6)**

Address: FFF26H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM6 | 1 | 1 | 1 | 1 | 1 | PM62 | PM61 | PM60 |

| PM6n | P6n pin I/O mode selection (n = 0 to 2) |
|---|---|
| 0 | Output mode (output buffer on) |
| 1 | Input mode (output buffer off) |

## 19.4 $I^2C$ Bus Mode Functions

### 19.4.1 Pin configuration

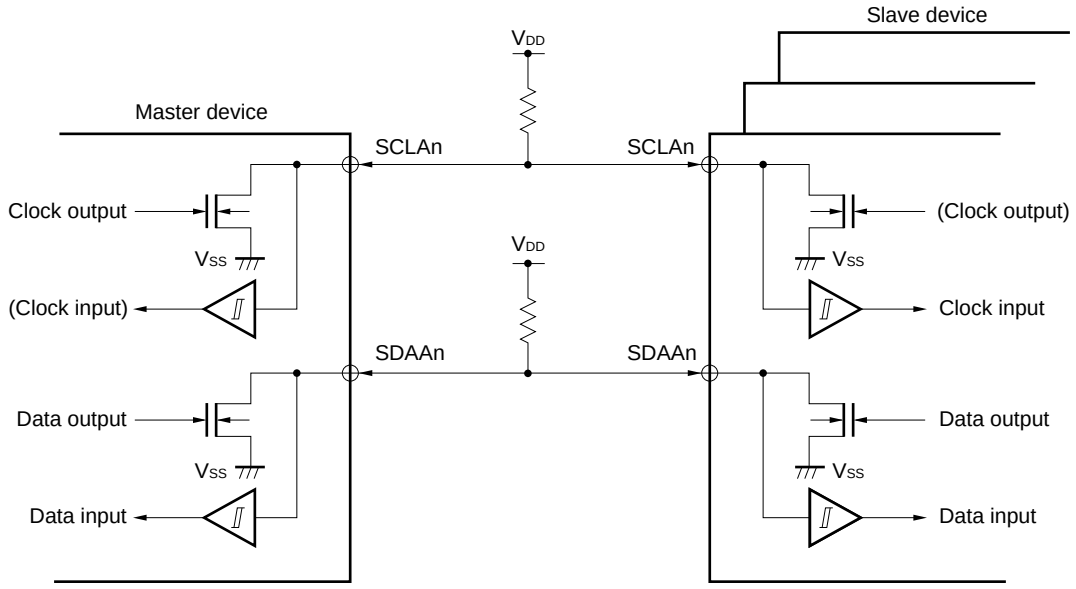
The serial clock pin (SCLAn) and the serial data bus pin (SDAAn) are configured as follows.

(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 19-13. Pin Configuration Diagram**




**Remark** n = 0

### 19.4.2 Setting transfer clock by using IICWLn and IICWHn registers

#### (1) Setting transfer clock on master side

$$\text{Transfer clock} = \frac{f_{MCK}}{IICWL0 + IICWH0 + f_{MCK}(t_R + t_F)}$$

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 truncated.)

- 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** and **Remarks** are listed on the next page.)

**Cautions 1. The fastest operation frequency of the IICA operation clock ($f_{MCK}$) is 20 MHz (Max.).**
**Set bit 0 (PRSn) of the IICA control register n1 (IICCTLn1) to "1" only when the $f_{CLK}$ exceeds 20 MHz.**

**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.)**

**Remarks 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 resistance and wire load.

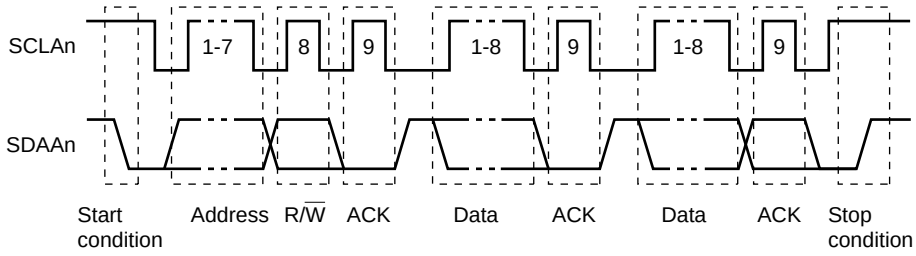
**2.** IICWLn: IICA low-level width setting register n
IICWHn: IICA high-level width setting register n
$t_F$: SDAAn and SCLAn signal falling times
$t_R$: SDAAn and SCLAn signal rising times
$f_{MCK}$: IICA operation clock frequency

**3.** n = 0

## 19.5 $I^2C$ Bus Definitions and Control Methods

The following section describes the $I^2C$ bus's serial data communication format and the signals used by the $I^2C$ bus. Figure 19-14 shows the transfer timing for the "start condition", "address", "data", and "stop condition" output via the $I^2C$ bus's serial data bus.

**Figure 19-14. $I^2C$ Bus Serial Data Transfer Timing**



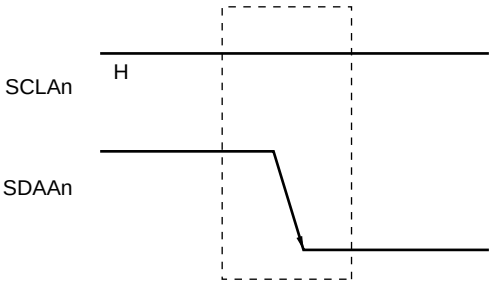

The master device generates the start condition, slave address, and stop condition.
The acknowledge (ACK) can be generated by either the master or slave device (normally, it is output by the device that receives 8-bit data).

The serial clock (SCLAn) is continuously output by the master device. However, in the slave device, the SCLAn pin low level period can be extended and a wait can be inserted.

### 19.5.1 Start conditions

A start condition is met when the SCLAn pin is at high level and the SDAAn pin changes from high level to low level. The start conditions for the SCLAn pin and SDAAn pin are signals 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 19-15. Start Conditions**




A start condition is output when bit 1 (STTn) of IICA control register n0 (IICCTLn0) is set (1) after a stop condition has been detected (SPDn: Bit 0 of the IICA status register n (IICSn) = 1). When a start condition is detected, bit 1 (STDn) of the IICSn register is set (1).

**Remark** n = 0

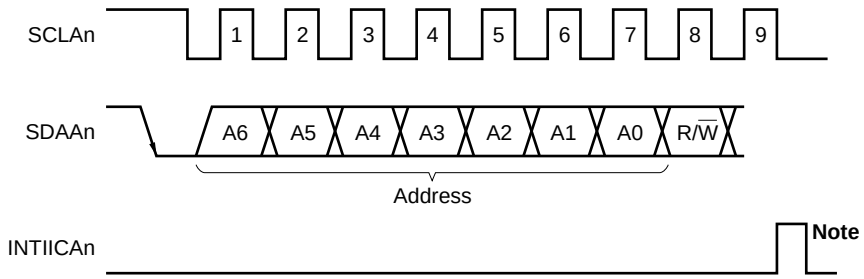
### 19.5.2 Addresses

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 19-16. Address**




**Note** INTIICAn is not issued if data other than a local address or extension code is received during slave device operation.

Addresses are output when a total of 8 bits consisting of the slave address and the transfer direction described in **19.5.3 Transfer direction specification** are written to the IICA shift register n (IICAn). The received addresses are written to the IICAn register.

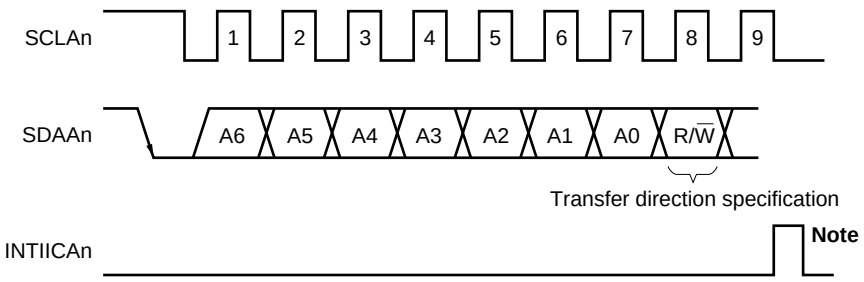
The slave address is assigned to the higher 7 bits of the IICAn register.

### 19.5.3 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 19-17. Transfer Direction Specification**




**Note** INTIICAn is not issued if data other than a local address or extension code is received during slave device operation.

**Remark** n = 0

### 19.5.4 Acknowledge (ACK)

ACK is used to check the status 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 bit 2 (ACKDn) 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.

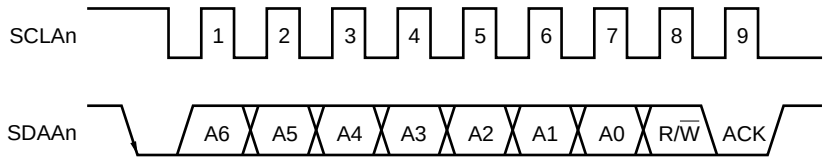
To generate ACK, the reception side makes the SDAAn line low at the ninth clock (indicating normal reception).

Automatic generation of ACK is enabled by setting bit 2 (ACKEn) of IICA control register n0 (IICCTLn0) to 1. Bit 3 (TRCn) of the IICSn register is set by the data 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 19-18. 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, ACK is generated if the ACKEn bit is set to 1 in advance.

How ACK is generated when data is received differs as follows depending on the setting of the wait timing.

- When 8-clock wait state is selected (bit 3 (WTIMn) of IICCTLn0 register = 0):
  By setting the ACKEn bit to 1 before releasing the wait state, ACK is generated at the falling edge of the eighth clock of the SCLAn pin.
- When 9-clock wait state is selected (bit 3 (WTIMn) of IICCTLn0 register = 1):
  ACK is generated by setting the ACKEn bit to 1 in advance.

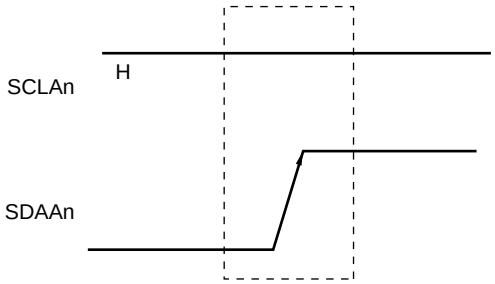
**Remark** n = 0

### 19.5.5 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 19-19. Stop Condition**




A stop condition is generated when bit 0 (SPTn) of IICA control register n0 (IICCTLn0) is set to 1. When the stop condition is detected, bit 0 (SPDn) of the IICA status register n (IICSn) is set to 1 and INTIICAn is generated when bit 4 (SPIEn) of the IICCTLn0 register is set to 1.

**Remark** n = 0

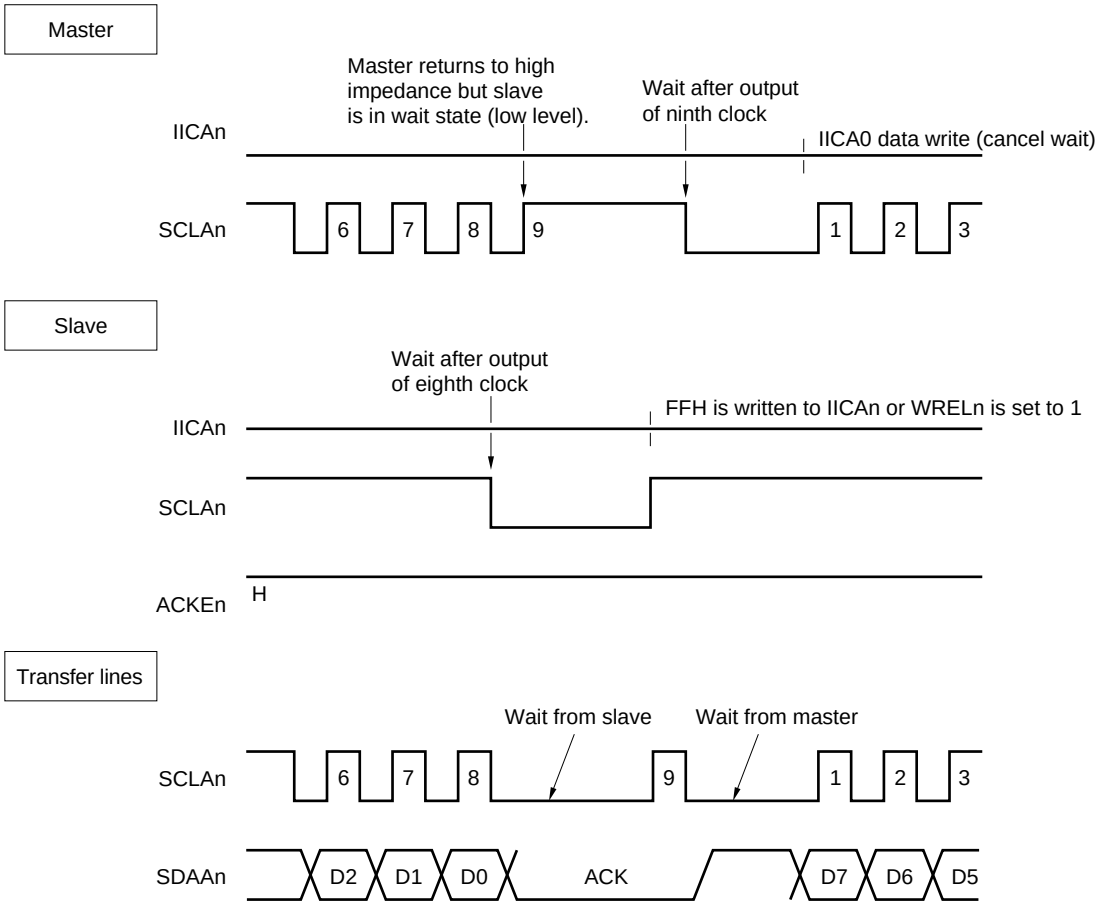
### 19.5.6 Wait

The wait is used to notify the communication partner that a device (master or slave) is preparing to transmit or receive data (i.e., is in a wait state).

Setting the SCLAn pin to low level notifies the communication partner of the wait state. When wait state has been canceled for both the master and slave devices, the next data transfer can begin.

**Figure 19-20. Wait (1/2)**

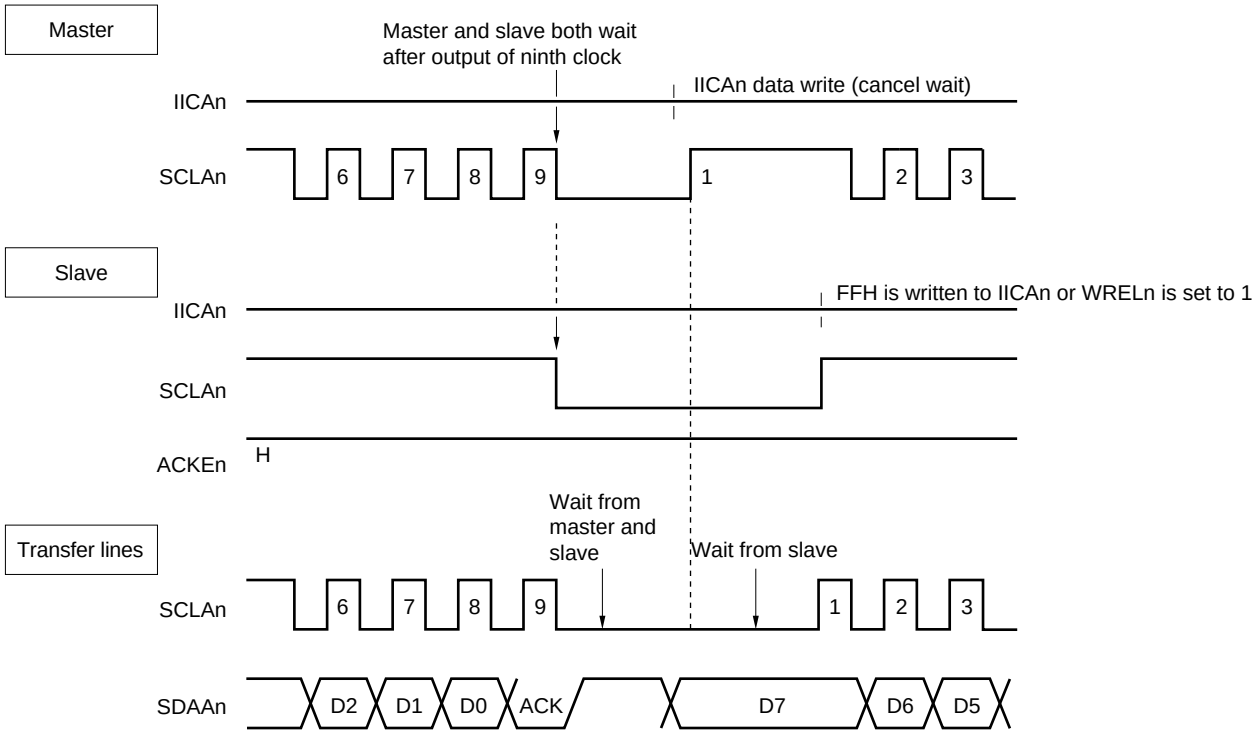
**(1) When master device has a nine-clock wait and slave device has an eight-clock wait (master transmits, slave receives, and ACKEn = 1)**




**Remark** n = 0

**Figure 19-20. Wait (2/2)**

**(2) When master and slave devices both have a nine-clock wait (master transmits, slave receives, and ACKEn = 1)**




**Remark** ACKEn: Bit 2 of IICA control register n0 (IICCTLn0)
WRELn: Bit 5 of IICA control register n0 (IICCTLn0)

A wait may be automatically generated depending on the setting of bit 3 (WTIMn) of IICA control register n0 (IICCTLn0).

Normally, the receiving side cancels the wait state when bit 5 (WRELn) of the IICCTLn0 register is set to 1 or when FFH is written to the IICA shift register n (IICAn), and the transmitting side cancels the wait state when data is written to the IICAn register.

The master device can also cancel the wait state via either of the following methods.

- By setting bit 1 (STTn) of the IICCTLn0 register to 1
- By setting bit 0 (SPTn) of the IICCTLn0 register to 1

**Remark** n = 0

### 19.5.7 Canceling wait

The $I^2C$ usually cancels a wait state by the following processing.

- Writing data to the IICA shift register n (IICAn)
- Setting bit 5 (WRELn) of IICA control register n0 (IICCTLn0) (canceling wait)
- Setting bit 1 (STTn) of the IICCTLn0 register (generating start condition)**Note**
- Setting bit 0 (SPTn) of the IICCTLn0 register (generating stop condition)**Note**

**Note** Master only

When the above wait canceling processing is executed, the $I^2C$ cancels the wait state and communication is resumed.

To cancel a wait state and transmit data (including addresses), write the data to the IICAn register.

To receive data after canceling a wait state, or to complete data transmission, set bit 5 (WRELn) of the IICCTLn0 register to 1.

To generate a restart condition after canceling a wait state, set bit 1 (STTn) of the IICCTLn0 register to 1.

To generate a stop condition after canceling a wait state, set bit n (SPTn) of the IICCTLn0 register to 1.

Execute the canceling processing only once for one wait state.

If, for example, data is written to the IICAn register after canceling a wait 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, communication is stopped if the IICEn bit is cleared to 0 when communication has been aborted, so that the wait state can be canceled.

If the $I^2C$ bus has deadlocked due to noise, processing is saved from communication by setting bit 6 (LRELn) of the IICCTLn0 register, so that the wait state can be canceled.

**Caution If a processing to cancel a wait state is executed when WUPn = 1, the wait state will not be canceled.**

**Remark** n = 0

### 19.5.8 Interrupt request (INTIICAn) generation timing and wait control

The setting of bit 3 (WTIMn) of IICA control register n0 (IICCTLn0) determines the timing by which INTIICAn is generated and the corresponding wait control, as shown in Table 19-2.

**Table 19-2. INTIICAn Generation Timing and Wait Control**

| 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 | 9 Note 2 | 9 Note 2 | 9 | 9 | 9 |

**Notes 1.** The slave device's INTIICAn signal and wait period occurs at the falling edge of the ninth clock 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 IICCTLn0 register's bit 2 (ACKEn). For a slave device that has received an extension code, 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 9th clock, but wait does not occur.

**2.** If the received address does not match the contents of the slave address register n (SVAn) and extension code is not received, neither INTIICAn nor a wait occurs.

**Remark** The numbers in the table indicate the number of the serial clock's clock signals. Interrupt requests and wait control are both synchronized with the falling edge of these clock signals.

**(1) During address transmission/reception**

- Slave device operation: Interrupt and wait timing are determined depending on the conditions described in Notes 1 and 2 above, regardless of the WTIMn bit.
- Master device operation: Interrupt and wait timing occur at the falling edge of the ninth clock regardless of the WTIMn bit.

**(2) During data reception**

- Master/slave device operation: Interrupt and wait timing are determined according to the WTIMn bit.

**(3) During data transmission**

- Master/slave device operation: Interrupt and wait timing are determined according to the WTIMn bit.

**(4) Wait cancellation method**

The four wait cancellation methods are as follows.

- Writing data to the IICA shift register n (IICAn)
- Setting bit 5 (WRELn) of IICA control register n0 (IICCTLn0) (canceling wait)
- Setting bit 1 (STTn) of IICCTLn0 register (generating start condition)Note
- Setting bit 0 (SPTn) of IICCTLn0 register (generating stop condition)Note

**Note** Master only.

When an 8-clock wait has been selected (WTIMn = 0), the presence/absence of ACK generation must be determined prior to wait cancellation.

**(5) Stop condition detection**

INTIICAn is generated when a stop condition is detected (only when SPIEn = 1).

**Remark** n = 0

### 19.5.9 Address match detection method

In $I^2C$ 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 (INTIICAn) occurs when the address set to the slave address register n (SVAn) matches the slave address sent by the master device, or when an extension code has been received.

### 19.5.10 Error detection

In $I^2C$ bus mode, the status 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

### 19.5.11 Extension code

(1) When the higher 4 bits of the receive address are either “0000” or “1111”, the extension code reception flag (EXCn) is set to 1 for extension code reception and an interrupt request (INTIICAn) is issued at the falling edge of the eighth clock. 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 when the SVAn register is set to 11110xx0. Note that INTIICAn occurs at the falling edge of the eighth clock.

- Higher four bits of data match: EXCn = 1
- Seven bits of data match: COIn = 1

**Remark** EXCn: Bit 5 of IICA status register n (IICSn)
COIn: Bit 4 of IICA status register n (IICSn)

(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 while a slave device is operating, 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 bit 6 (LRELn) of IICA control register n0 (IICCTLn0) to 1 to set the standby mode for the next communication operation.

**Table 19-3. 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) |

**Remarks 1.** See the $I^2C$ bus specifications issued by NXP Semiconductors for details of extension codes other than those described above.

**2.** n = 0

### 19.5.12 Arbitration

When several master devices simultaneously generate a start condition (when the STTn bit is set to 1 before the STDn bit 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.

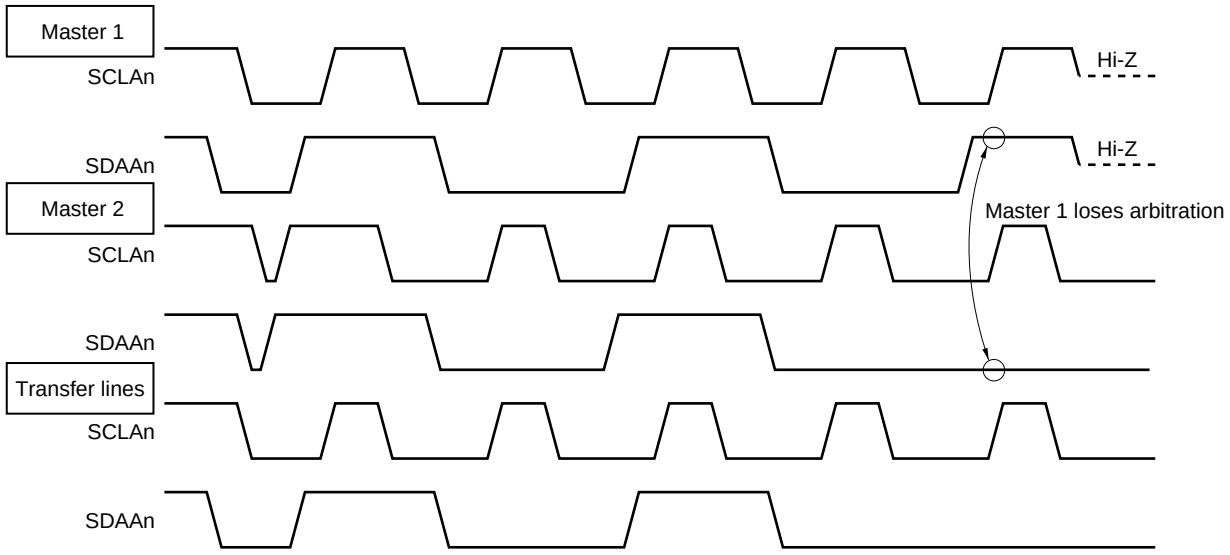
When one of the master devices loses in arbitration, an arbitration loss flag (ALDn) in the IICA status register n (IICSn) is set (1) via the timing by which the arbitration loss occurred, and the SCLAn and SDAAn lines are both set to high impedance, which releases the bus.

The arbitration loss is detected based on the timing of the next interrupt request (the eighth or ninth clock, when a stop condition is detected, etc.) and the ALDn = 1 setting that has been made by software.

For details of interrupt request timing, see **19.5.8 Interrupt request (INTIICAn) generation timing and wait control**.

**Remark** STDn: Bit 1 of IICA status register n (IICSn)
STTn: Bit 1 of IICA control register n0 (IICCTLn0)

**Figure 19-21. Arbitration Timing Example**




**Remark** n = 0

**Table 19-4. Status During Arbitration and Interrupt Request Generation Timing**

| Status 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 | |

**Notes 1.** When the WTIMn bit (bit 3 of IICA control register n0 (IICCTLn0)) = 1, an interrupt request occurs at the falling edge of the ninth clock. When WTIMn = 0 and the extension code's slave address is received, an interrupt request occurs at the falling edge of the eighth clock.

**2.** When there is a chance that arbitration will occur, set SPIEn = 1 for master device operation.

**Remarks 1.** SPIEn: Bit 4 of IICA control register n0 (IICCTLn0)

**2.** n = 0

### 19.5.13 Wakeup function

The $I^2C$ bus slave function is a function that generates an interrupt request signal (INTIICAn) when a local address and extension code have been received.

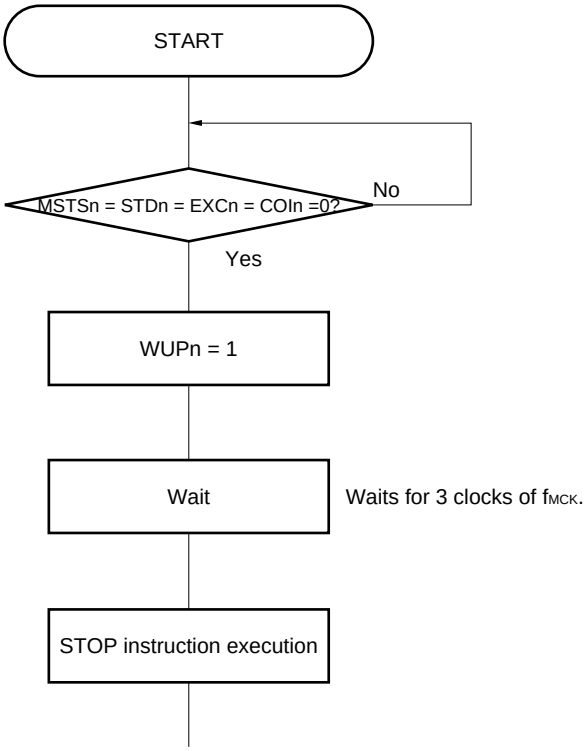
This function makes processing more efficient by preventing unnecessary INTIICAn signal from occurring when addresses do not match.

When a start condition is detected, wakeup standby mode is set. This wakeup standby mode is in effect while addresses are transmitted due to the possibility that an arbitration loss may change the master device (which has generated a start condition) to a slave device.

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 a local address and extension code have been received. Operation returns to normal operation by using an instruction to clear (0) the WUPn bit after this interrupt has been generated.

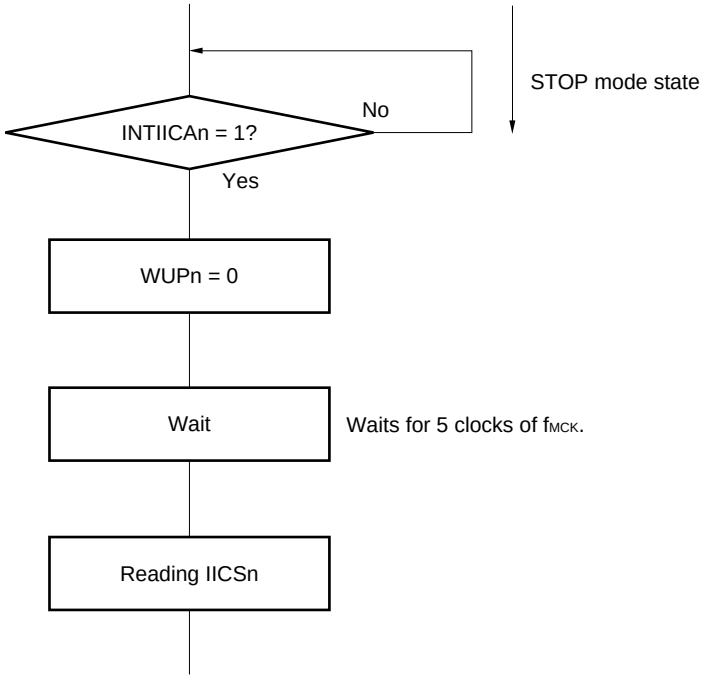
Figure 19-22 shows the flow for setting WUPn = 1 and Figure 19-23 shows the flow for setting WUPn = 0 upon an address match.

**Figure 19-22. Flow When Setting WUPn = 1**




**Remark** n = 0

**Figure 19-23. Flow When Setting WUPn = 0 upon Address Match (Including Extension Code Reception)**

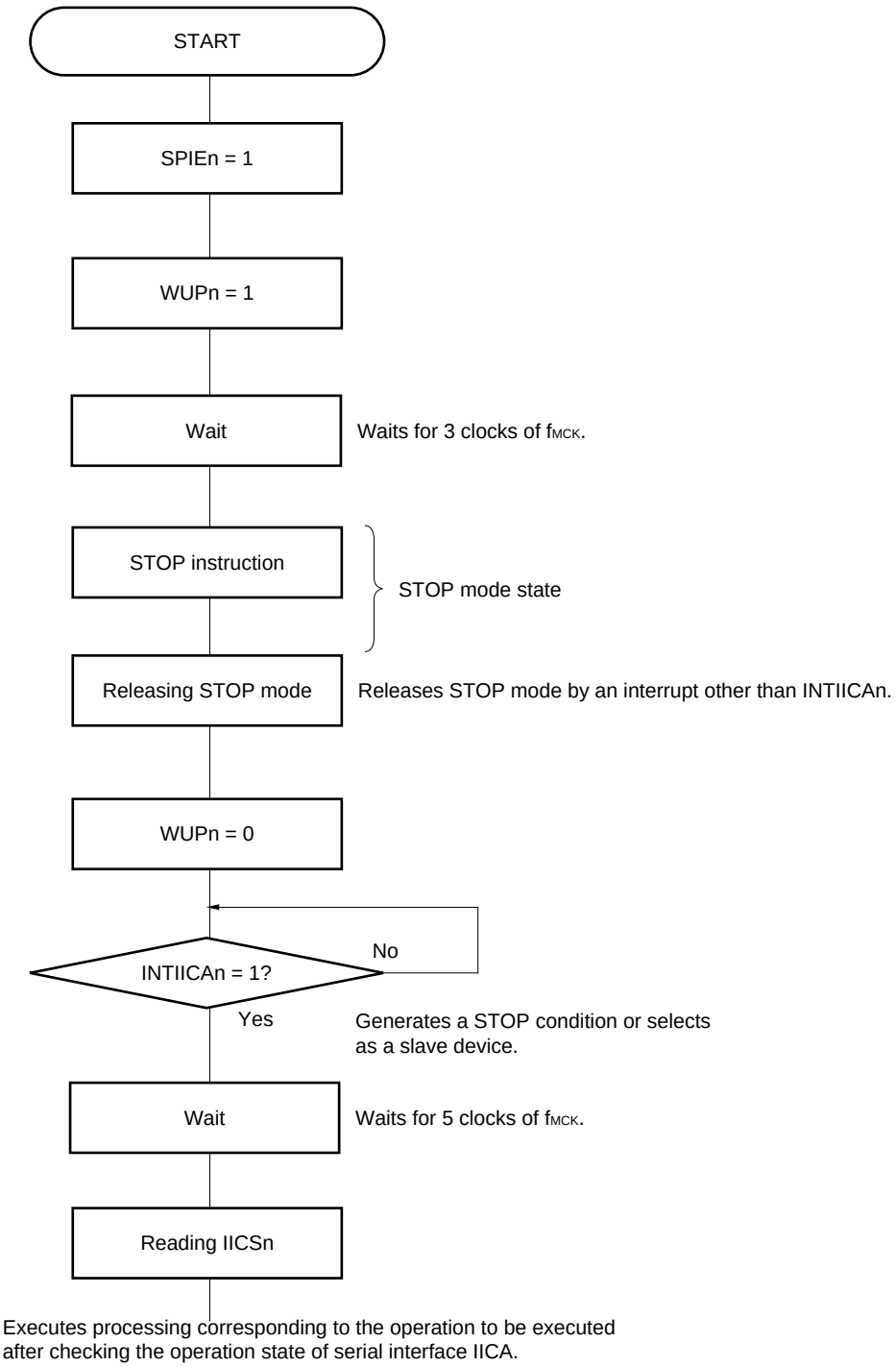



Use the following flows to perform the processing to release the STOP mode other than by an interrupt request (INTIICAn) generated from serial interface IICA.

- When operating next IIC communication as master: Flow shown in Figure 19-24
- When operating next IIC communication as slave:
  When restored by INTIICAn interrupt: Same as the flow in Figure 19-23
  When restored by other than INTIICAn interrupt: Until the INTIICAn interrupt occurs, continue operating with WUPn left set to 1

**Remark** n = 0

**Figure 19-24. When Operating as Master Device After Releasing STOP Mode Other than by INTIICAn**




**Remark** n = 0

### 19.5.14 Communication reservation

**(1) When communication reservation function is enabled (bit n (IICRSVn) of IICA flag register n (IICFn) = 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
- When an extension code is received and slave operation is disabled (ACK is not returned and the bus was released by setting bit 6 (LRELn) of IICA control register n0 (IICCTLn0) to 1 and saving communication).

If bit 1 (STTn) of the IICCTLn0 register is set to 1 while the bus is not used (after a stop condition is detected), a start condition is automatically generated and wait state is set.
If an address is written to the IICA shift register n (IICAn) after bit 4 (SPIEn) 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 bus status.

- 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 bit (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.

<R>

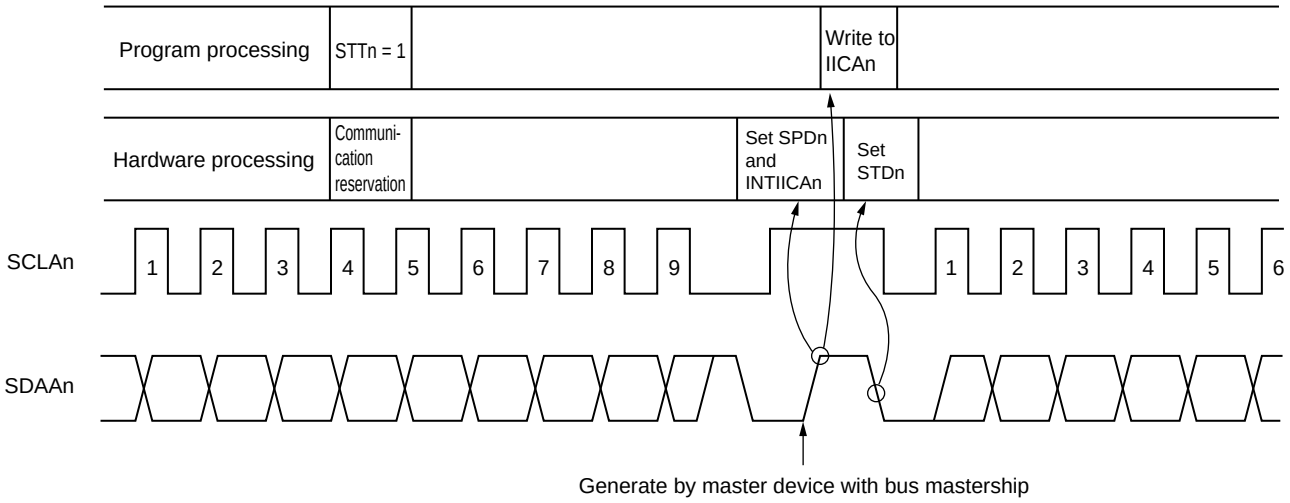
| Wait time from setting STTn = 1 to checking the MSTSn flag: |
|---|
| (IICWLn setting value + IICWHn setting value + 4)/fMCK + tF × 2 |

**Remarks 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

**2.** n = 0

Figure 19-25 shows the communication reservation timing.

**Figure 19-25. Communication Reservation Timing**




**Remark** 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)

Communication reservations are accepted via the timing shown in Figure 19-26. After bit 1 (STDn) of the IICA status register n (IICSn) is set to 1, a communication reservation can be made by setting bit 1 (STTn) of IICA control register n0 (IICCTLn0) to 1 before a stop condition is detected.

**Figure 19-26. Timing for Accepting Communication Reservations**

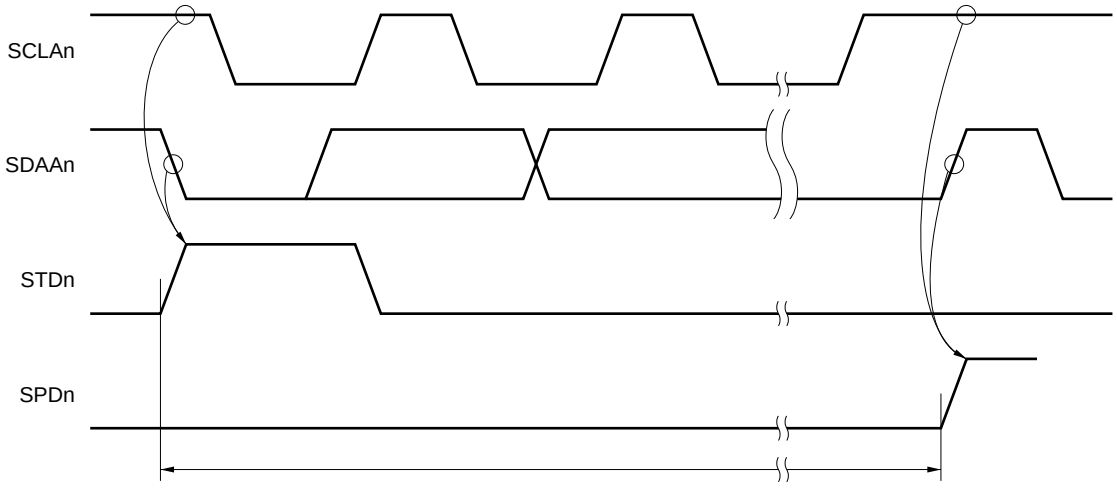
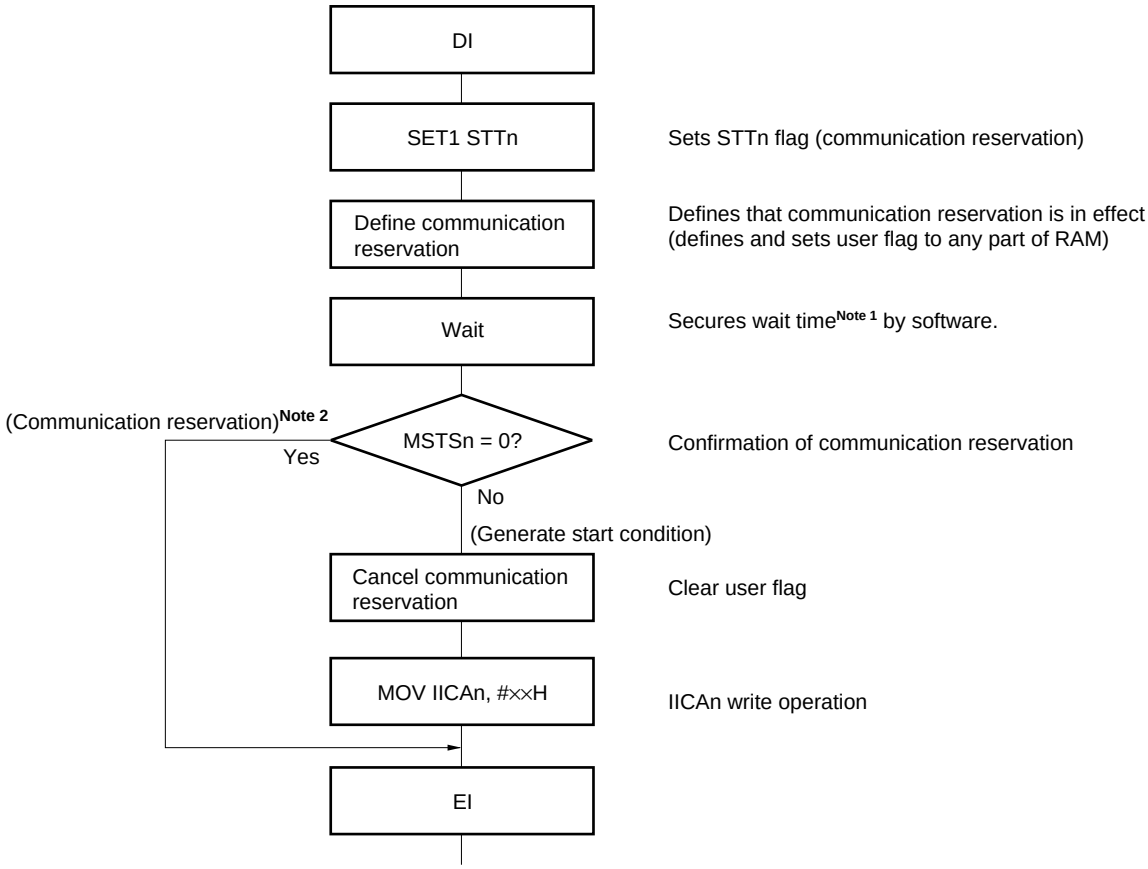



Figure 19-27 shows the communication reservation protocol.

**Remark** n = 0

**Figure 19-27. Communication Reservation Protocol**




<R> **Notes 1.** The wait time is calculated as follows.
(IICWLn setting value + IICWHn setting value + 4)/fMCK + tF × 2

**2.** The communication reservation operation executes a write to the IICA shift register n (IICAn) when a stop condition interrupt request occurs.

**Remarks 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

**2.** n = 0

**(2) When communication reservation function is disabled (bit 0 (IICRSVn) of IICA flag register n (IICFn) = 1)**

When bit 1 (STTn) of 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 following two statuses are included in the status where bus is not used.

- When arbitration results in neither master nor slave operation
- When an extension code is received and slave operation is disabled (ACK is not returned and the bus was released by setting bit 6 (LRELn) of the IICCTLn0 register to 1 and saving communication)

To confirm whether the start condition was generated or request was rejected, check STCFn (bit 7 of the IICFn register). It takes up to 5 clocks of $f_{MCK}$ until the STCFn bit is set to 1 after setting STTn = 1. Therefore, secure the time by software.

**Remark** n = 0

### 19.5.15 Cautions

(1) When STCENn = 0

Immediately after $I^2C$ operation is enabled (IICEn = 1), the bus communication status (IICBSYn = 1) is recognized regardless of the actual bus status. 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 bit 7 (IICEn) of IICA control register n0 (IICCTLn0) to 1.
<3> Set bit 0 (SPTn) of the IICCTLn0 register to 1.

(2) When STCENn = 1

Immediately after $I^2C$ operation is enabled (IICEn = 1), the bus released status (IICBSYn = 0) is recognized regardless of the actual bus status. 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 macro of $I^2C$ 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, ACK is returned, but this interferes with other $I^2C$ communications. To avoid this, start $I^2C$ in the following sequence.

<1> Clear bit 4 (SPIEn) of the IICCTLn0 register to 0 to disable generation of an interrupt request signal (INTIICAn) when the stop condition is detected.
<2> Set bit 7 (IICEn) of the IICCTLn0 register to 1 to enable the operation of $I^2C$.
<3> Wait for detection of the start condition.
<4> Set bit 6 (LRELn) of the IICCTLn0 register to 1 before ACK is returned (4 to 72 clocks 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 bit (bit 7 of the IICA status register n (IICSn)) is detected by software.

**Remark** n = 0

### 19.5.16 Communication operations

The following shows three operation procedures with the flowchart.

**(1) Master operation in single master system**

The flowchart when using the RL78/I1B 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 multimaster system**

In the $I^2C$ bus multimaster system, whether the bus is released or used cannot be judged by the $I^2C$ bus specifications when the bus takes part in a communication. Here, when data and clock are at a high level for a certain period (1 frame), the RL78/I1B 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 RL78/I1B looses 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 the RL78/I1B 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 communication status is judged and its result is passed as a flag over to the main processing.

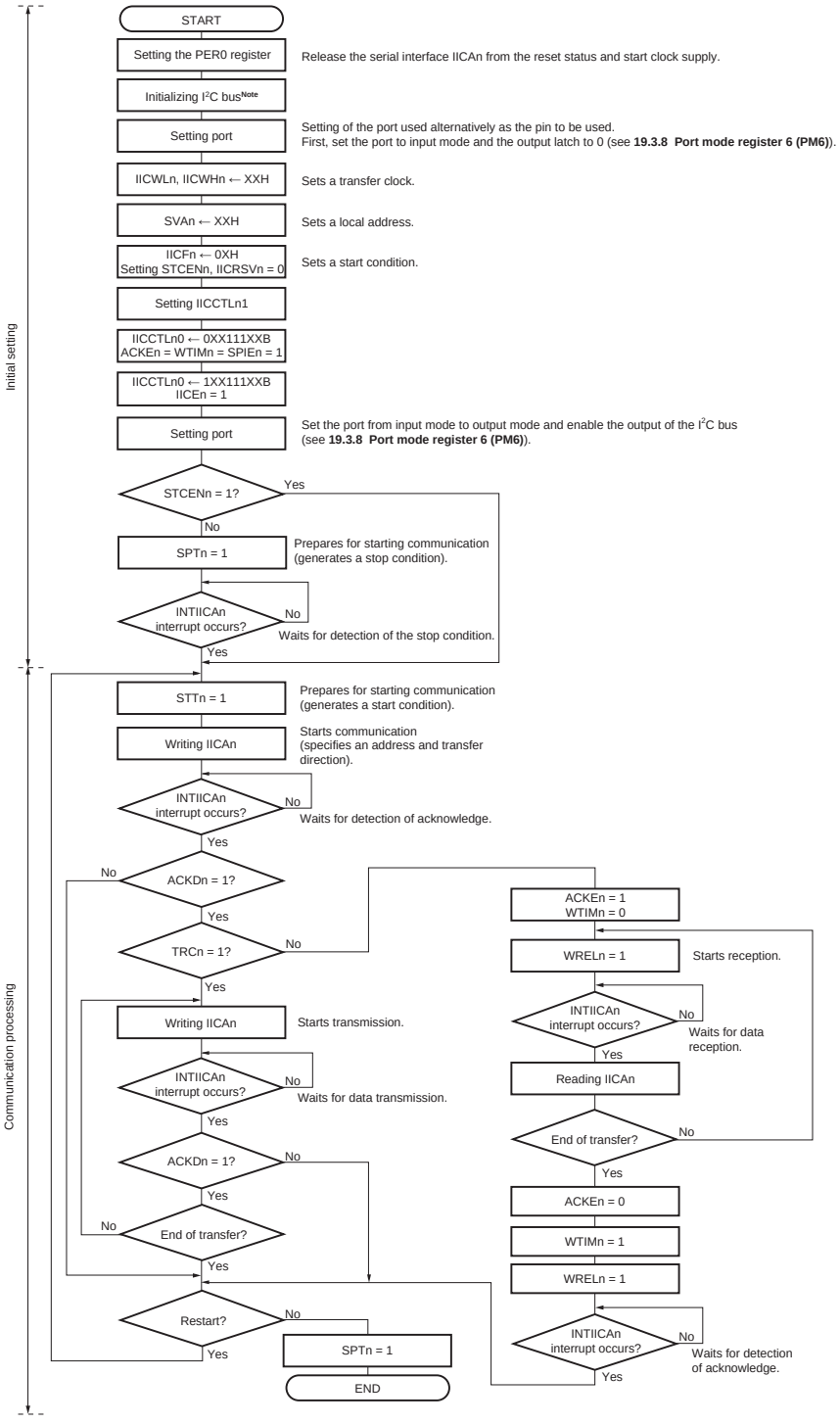
By checking the flags, necessary communication processing is performed.

**Remark** n = 0

### (1) Master operation in single-master system

<R>

**Figure 19-28. 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.

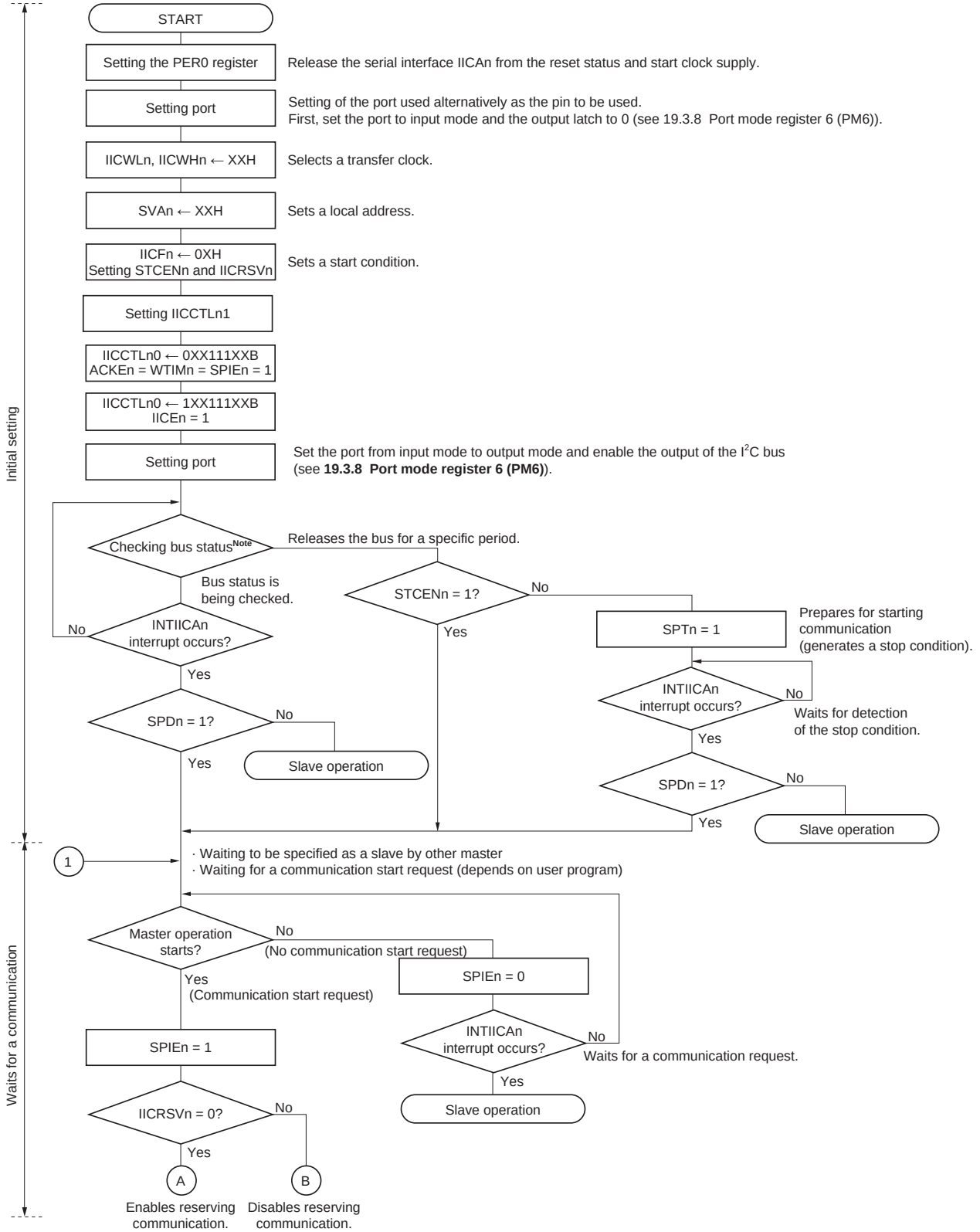
**Remarks 1.** Conform to the specifications of the product that is communicating, with respect to the transmission and reception formats.

**2.** n = 0

### (2) Master operation in multi-master system

<R>

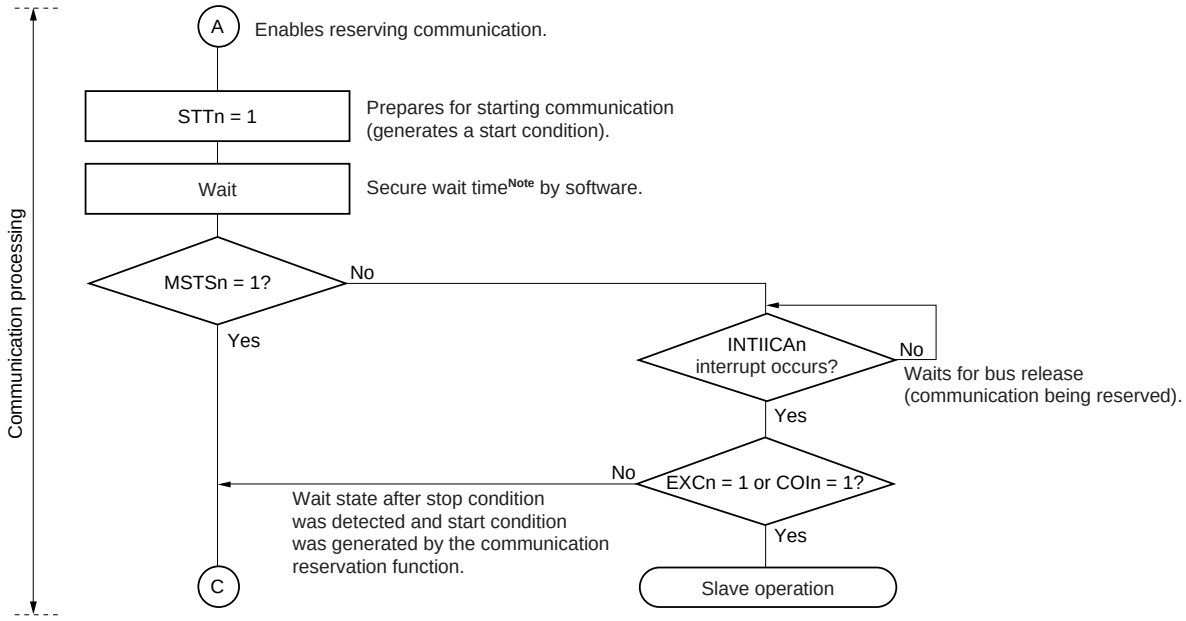
**Figure 19-29. Master Operation in Multi-Master System (1/3)**




**Note** Confirm that the bus is released (CLDn bit = 1, DADn bit = 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.

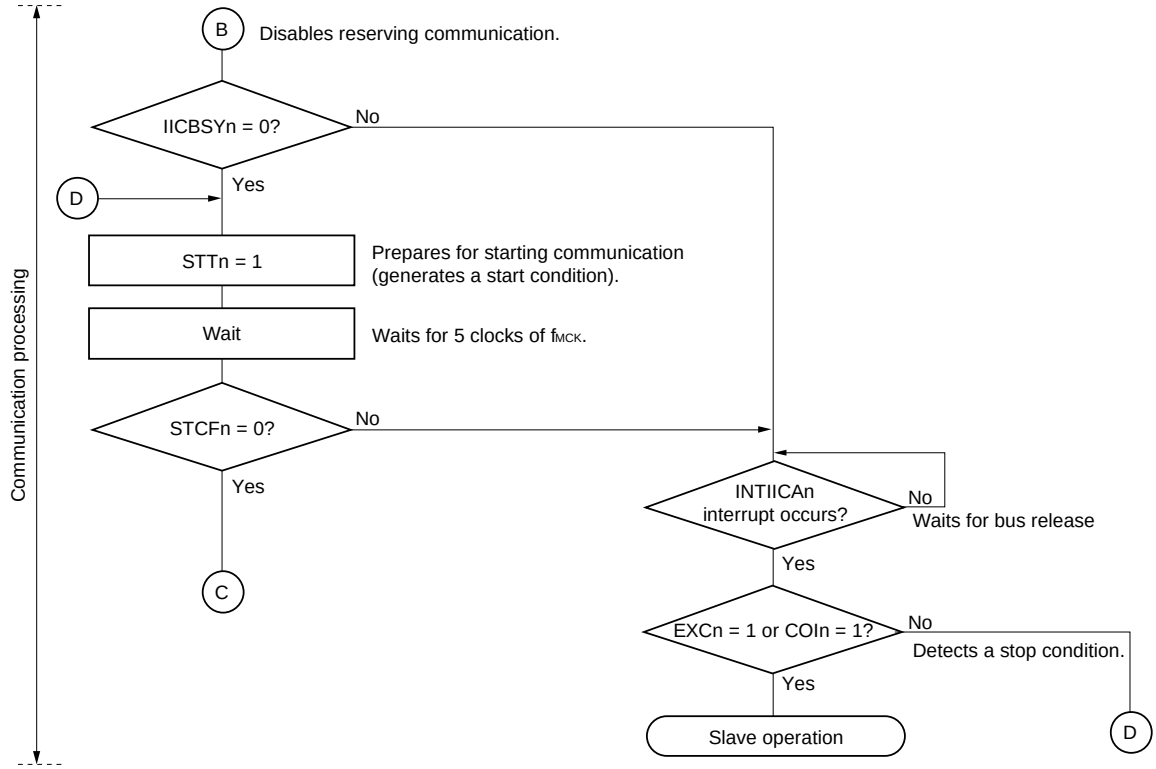
**Remark** n = 0

**Figure 19-29. Master Operation in Multi-Master System (2/3)**




<R> **Note** The wait time is calculated as follows.

(IICWLn setting value + IICWHn setting value + 4)/fMCK + tF × 2




**Remarks 1.** IICWLn: IICA low-level width setting register n

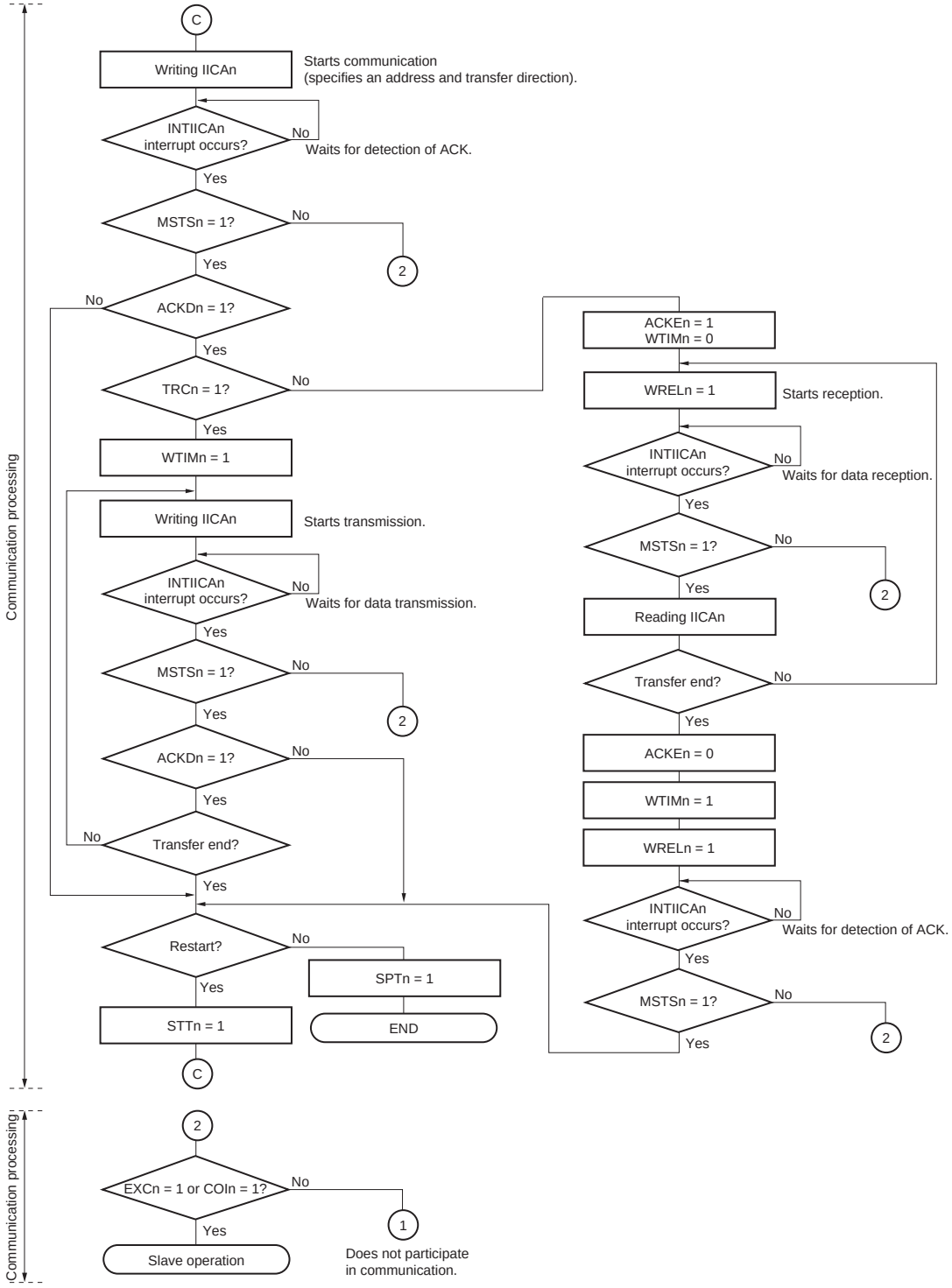
IICWHn: IICA high-level width setting register n

tF: SDAAn and SCLAn signal falling times

fMCK: IICA operation clock frequency

**2.** n = 0

**Figure 19-29. Master Operation in Multi-Master System (3/3)**




**Remarks 1.** Conform to the specifications of the product that is communicating, with respect to the transmission and reception formats.

**2.** To use the device as a master in a multi-master system, read the MSTSn bit each time interrupt INTIICAn has occurred to check the arbitration result.

**3.** To use the device as a slave in a multi-master system, check the status 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.

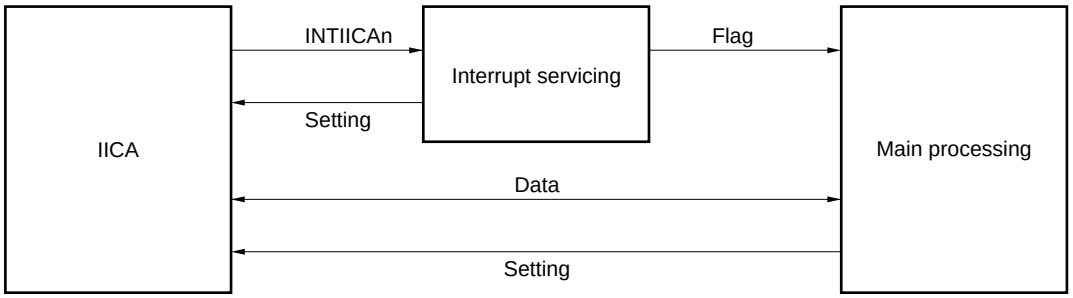
**4.** n = 0

### (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 operation status such as detection of a stop condition during communication) is necessary.

In the following explanation, it is assumed that the extension code is not supported for data communication. It is also assumed that the INTIICAn interrupt servicing only performs status transition processing, and that actual data communication is performed by the main processing.




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 communication statuses.

- Clear mode: Status in which data communication is not performed
- Communication mode: Status in which data communication is performed (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 bit.

**Remark** n = 0

The main processing of the slave operation is explained next.

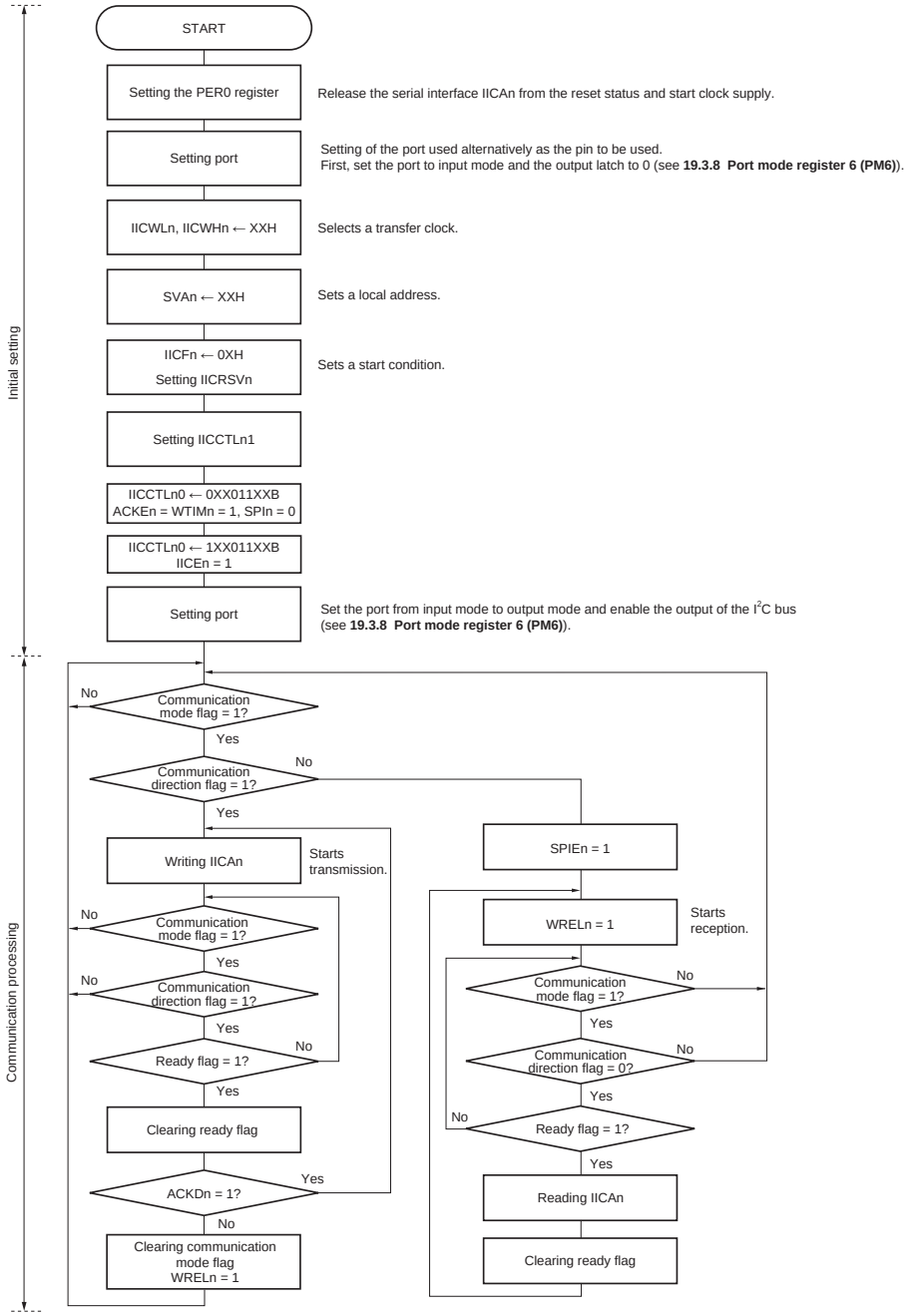
Start serial interface IICA and wait until communication is enabled. When communication is enabled, execute communication by using the communication mode flag and ready flag (processing of the stop condition and start condition is performed by an interrupt. Here, check the status by using the flags).

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 communication status occurs in this way.

<R>

**Figure 19-30. Slave Operation Flowchart (1)**




**Remarks 1.** Conform to the specifications of the product that is in communication, regarding the transmission and reception formats.

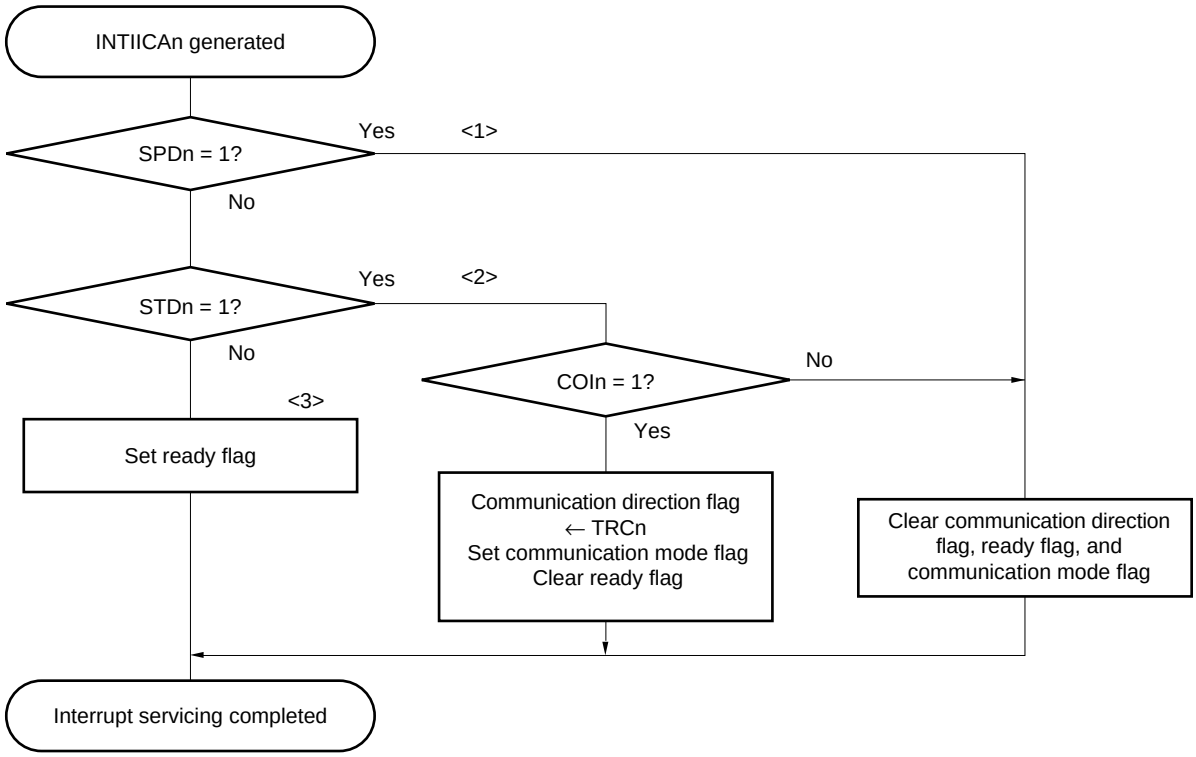
**2.** n = 0

An example of the processing procedure of the slave with the INTIICAn interrupt is explained below (processing is performed assuming that no extension code is used). The INTIICAn interrupt checks the status, 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 cancelled, 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 19-31 Slave Operation Flowchart (2).

**Figure 19-31. Slave Operation Flowchart (2)**




**Remark** n = 0

### 19.5.17 Timing of $I^2C$ interrupt request (INTIICAn) occurrence

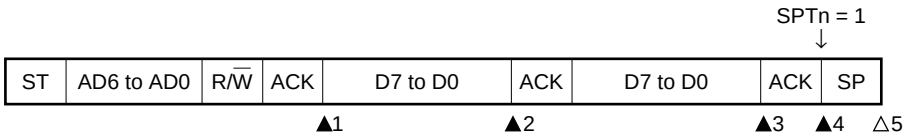
The timing of transmitting or receiving data and generation of interrupt request signal INTIICAn, and the value of the IICA status register n (IICSn) when the INTIICAn signal is generated are shown below.

**Remarks 1.** ST: Start condition
AD6 to AD0: Address
R/$\overline{W}$: Transfer direction specification
ACK: Acknowledge
D7 to D0: Data
SP: Stop condition
**2.** n = 0

**(1) Master device operation**

**(a) Start ~ Address ~ Data ~ Data ~ Stop (transmission/reception)**

**(i) When WTIMn = 0**

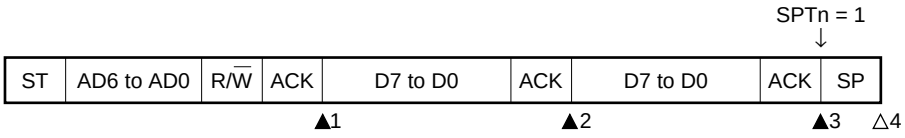



▲1: IICSn = 1000×110B
▲2: IICSn = 1000×000B
▲3: IICSn = 1000×000B (Sets the WTIMn bit to 1)[Note]
▲4: IICSn = 1000××00B (Sets the SPTn bit to 1)[Note]
△5: IICSn = 00000001B

**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**



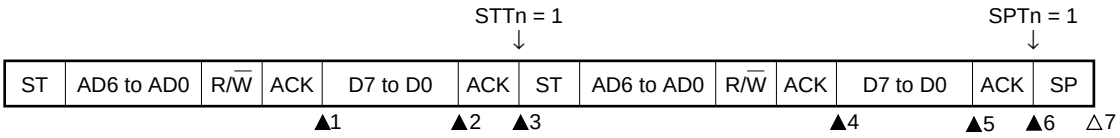

▲1: IICSn = 1000×110B
▲2: IICSn = 1000×100B
▲3: IICSn = 1000××00B (Sets the SPTn bit to 1)
△4: IICSn = 00000001B

**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**

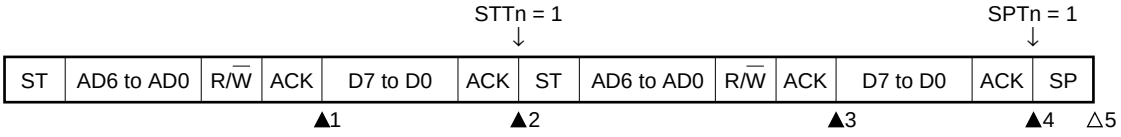



▲1: IICSn = 1000×110B
▲2: IICSn = 1000×000B (Sets the WTIMn bit to 1)[Note 1]
▲3: IICSn = 1000××00B (Clears the WTIMn bit to 0[Note 2], sets the STTn bit to 1)
▲4: IICSn = 1000×110B
▲5: IICSn = 1000×000B (Sets the WTIMn bit to 1)[Note 3]
▲6: IICSn = 1000××00B (Sets the SPTn bit to 1)
△7: IICSn = 00000001B

**Notes 1.** To generate a start condition, set the WTIMn bit to 1 and change the timing for generating the INTIICAn interrupt request signal.
**2.** Clear the WTIMn bit to 0 to restore the original setting.
**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**



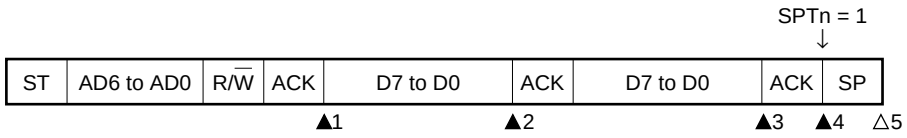

▲1: IICSn = 1000×110B
▲2: IICSn = 1000××00B (Sets the STTn bit to 1)
▲3: IICSn = 1000×110B
▲4: IICSn = 1000××00B (Sets the SPTn bit to 1)
△5: IICSn = 00000001B

**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**

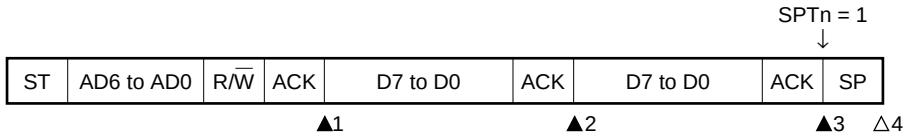



▲1: IICSn = 1010×110B
▲2: IICSn = 1010×000B
▲3: IICSn = 1010×000B (Sets the WTIMn bit to 1)[Note]
▲4: IICSn = 1010××00B (Sets the SPTn bit to 1)
△5: IICSn = 00000001B

**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 = 00001001B

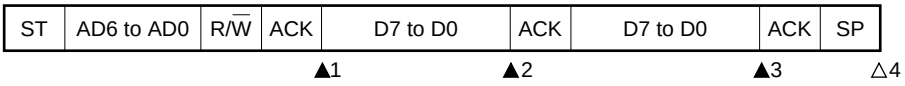
**Remark** ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

**Remark** n = 0

**(2) Slave device operation (slave address data reception)**

**(a) Start ~ Address ~ Data ~ Data ~ Stop**

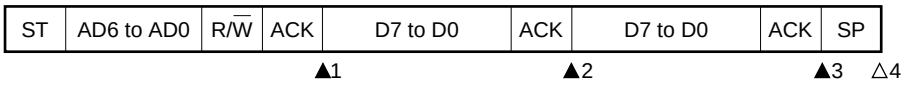
**(i) When WTIMn = 0**




▲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**



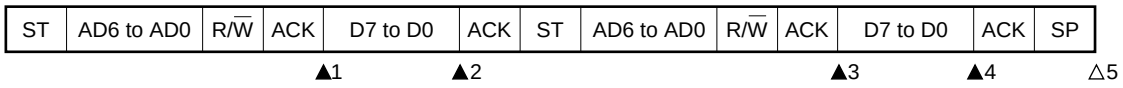

▲1: IICSn = 0001×110B
▲2: IICSn = 0001×100B
▲3: IICSn = 0001××00B
△4: IICSn = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

**Remark** n = 0

**(b) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop**

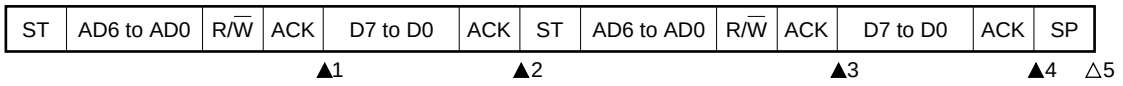
**(i) When WTIMn = 0 (after restart, matches with SVAn)**




▲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)**




▲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**

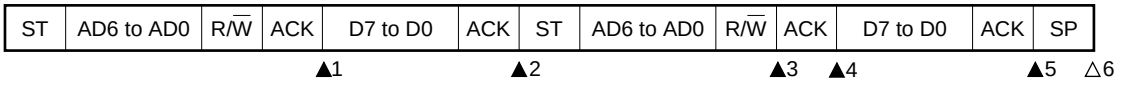
**(i) When WTIMn = 0 (after restart, does not match address (= extension code))**




▲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))**




▲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**

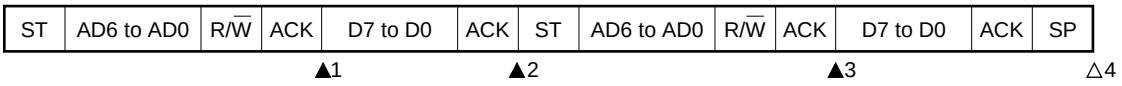
**(i) When WTIMn = 0 (after restart, does not match address (= not extension code))**




▲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))**




▲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

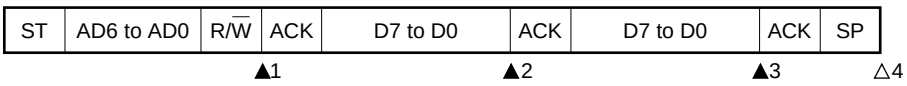
**Remark** n = 0

**(3) Slave device operation (when receiving extension code)**

The device is always participating in communication when it receives an extension code.

**(a) Start ~ Code ~ Data ~ Data ~ Stop**

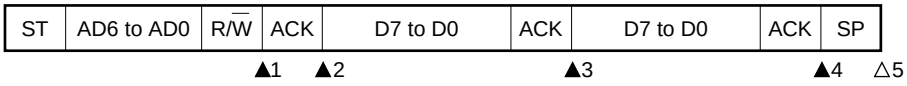
**(i) When WTIMn = 0**




▲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**



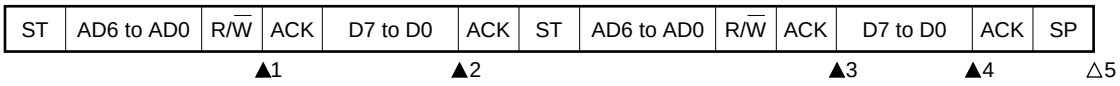

▲1: IICSn = 0010×010B
▲2: IICSn = 0010×110B
▲3: IICSn = 0010×100B
▲4: IICSn = 0010××00B
△5: IICSn = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

**Remark** n = 0

**(b) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop**

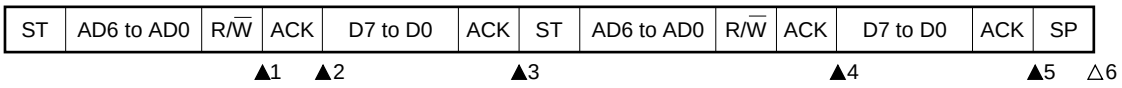
**(i) When WTIMn = 0 (after restart, matches SVAn)**




▲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 SVAn)**



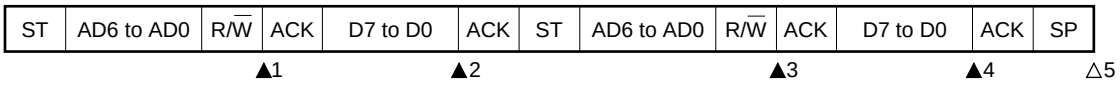

▲1: IICSn = 0010×010B
▲2: IICSn = 0010×110B
▲3: IICSn = 0010××00B
▲4: IICSn = 0001×110B
▲5: IICSn = 0001××00B
△6: IICSn = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

**Remark** n = 0

**(c) Start ~ Code ~ Data ~ Start ~ Code ~ Data ~ Stop**

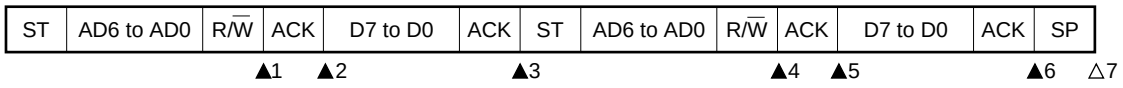
**(i) When WTIMn = 0 (after restart, extension code reception)**




▲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)**




▲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**

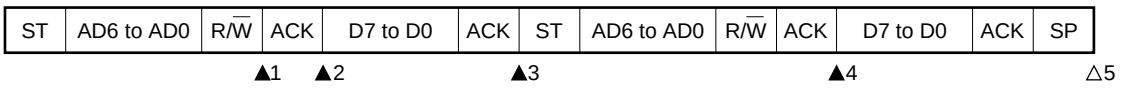
**(i) When WTIMn = 0 (after restart, does not match address (= not extension code))**




▲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))**




▲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**

| ST | AD6 to AD0 | R/$\overline{W}$ | ACK | D7 to D0 | ACK | D7 to D0 | ACK | SP |
|---|---|---|---|---|---|---|---|---|

△1

△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 bit 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**

| ST | AD6 to AD0 | R/$\overline{W}$ | ACK | D7 to D0 | ACK | D7 to D0 | ACK | SP |
|---|---|---|---|---|---|---|---|---|

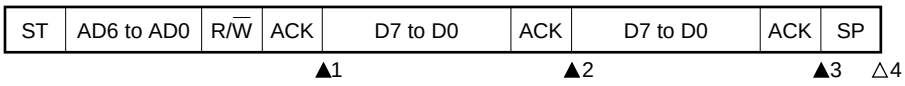
▲1 ▲2 ▲3 △4

▲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

**Remark** n = 0

**(ii) When WTIMn = 1**



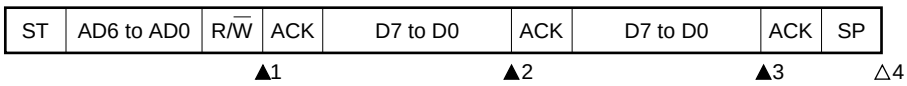

▲1 ▲2 ▲3 △4

▲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

**(b) When arbitration loss occurs during transmission of extension code**

**(i) When WTIMn = 0**



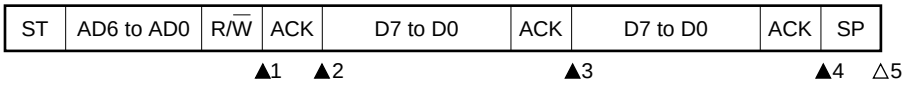

▲1 ▲2 ▲3 △4

▲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

**Remark** n = 0

**(ii) When WTIMn = 1**



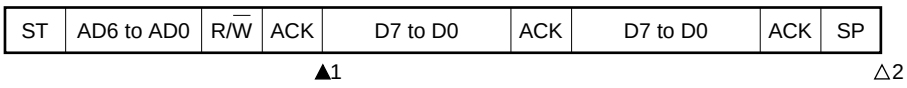

▲1: IICSn = 0110×010B
▲2: IICSn = 0010×110B
▲3: IICSn = 0010×100B
▲4: IICSn = 0010××00B
△5: IICSn = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

**(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 bit 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)**

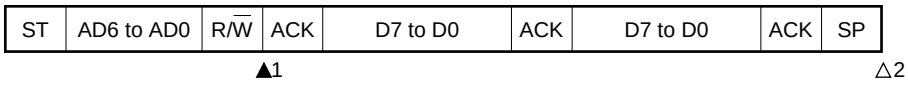



▲1: IICSn = 01000110B
△2: IICSn = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIEn = 1

**Remark** n = 0

**(b) When arbitration loss occurs during transmission of extension code**



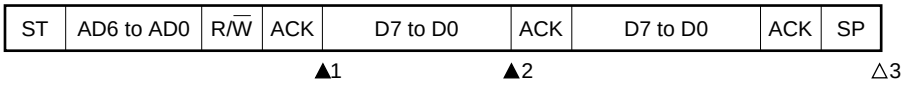

▲1 △2

▲1: IICSn = 0110×010B
Sets LRELn = 1 by software
△2: IICSn = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care

**(c) When arbitration loss occurs during transmission of data**

**(i) When WTIMn = 0**



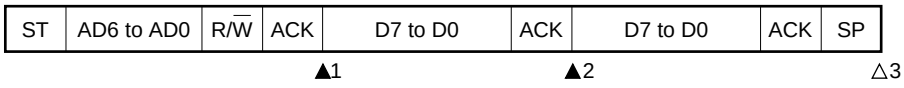

▲1 ▲2 △3

▲1: IICSn = 10001110B
▲2: IICSn = 01000000B
△3: IICSn = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIEn = 1

**Remark** n = 0

**(ii) When WTIMn = 1**

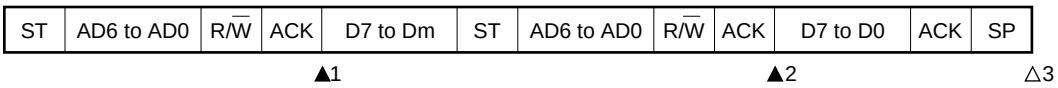



▲1: IICSn = 10001110B
▲2: IICSn = 01000100B
△3: IICSn = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIEn = 1

**(d) When loss occurs due to restart condition during data transfer**

**(i) Not extension code (Example: unmatches with SVAn)**

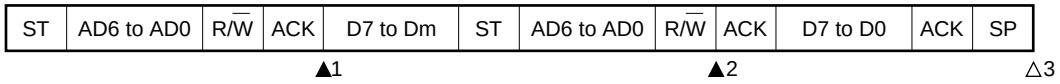



▲1: IICSn = 1000×110B
▲2: IICSn = 01000110B
△3: IICSn = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIEn = 1
×: Don't care
m = 6 to 0

**Remark** n = 0

**(ii) Extension code**




▲1: IICSn = 1000×110B

▲2: IICSn = 01100010B

Sets LRELn = 1 by software

△3: IICSn = 00000001B

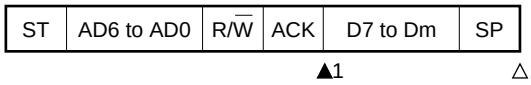
**Remark** ▲: Always generated

△: 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

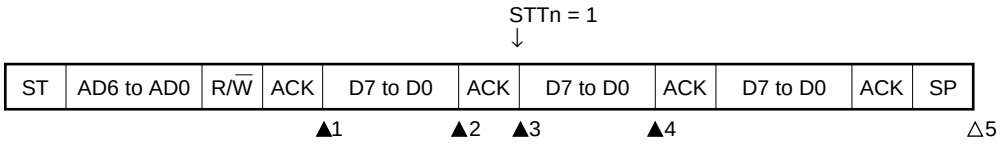
×: Don't care

m = 6 to 0

**Remark** n = 0

**(f) When arbitration loss occurs due to low-level data when attempting to generate a restart condition**

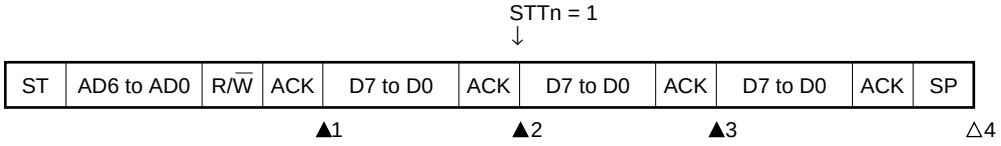
**(i) When WTIMn = 0**




▲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**



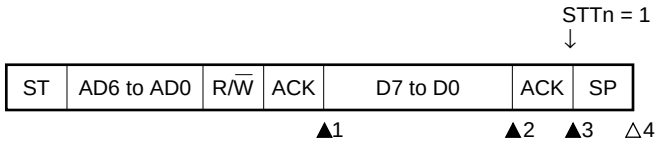

▲1: IICSn = 1000×110B
▲2: IICSn = 1000×100B (Sets the STTn bit to 1)
▲3: IICSn = 01000100B
△4: IICSn = 00000001B

**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**

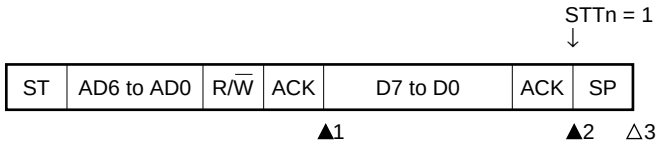
**(i) When WTIMn = 0**




▲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**



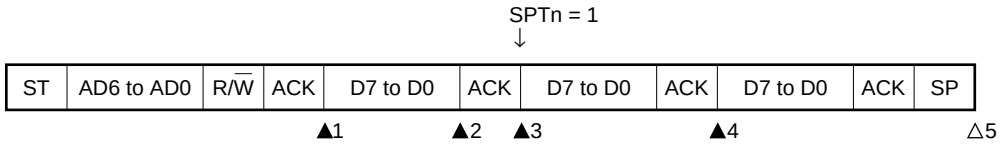

▲1: IICSn = 1000×110B
▲2: IICSn = 1000××00B (Sets the STTn bit to 1)
△3: IICSn = 01000001B

**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**

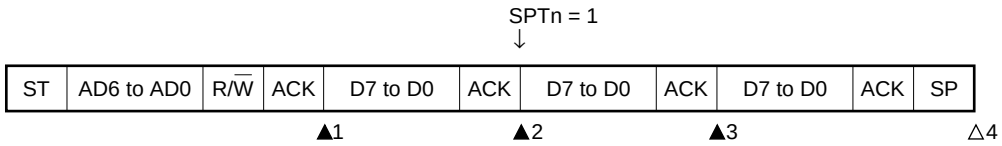
**(i) When WTIMn = 0**




▲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

## 19.6 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 bit (bit 3 of the IICA status register n (IICSn)), which specifies the data transfer direction, and then starts serial communication with the slave device.

Figures 19-32 and 19-33 show timing charts of the data communication.

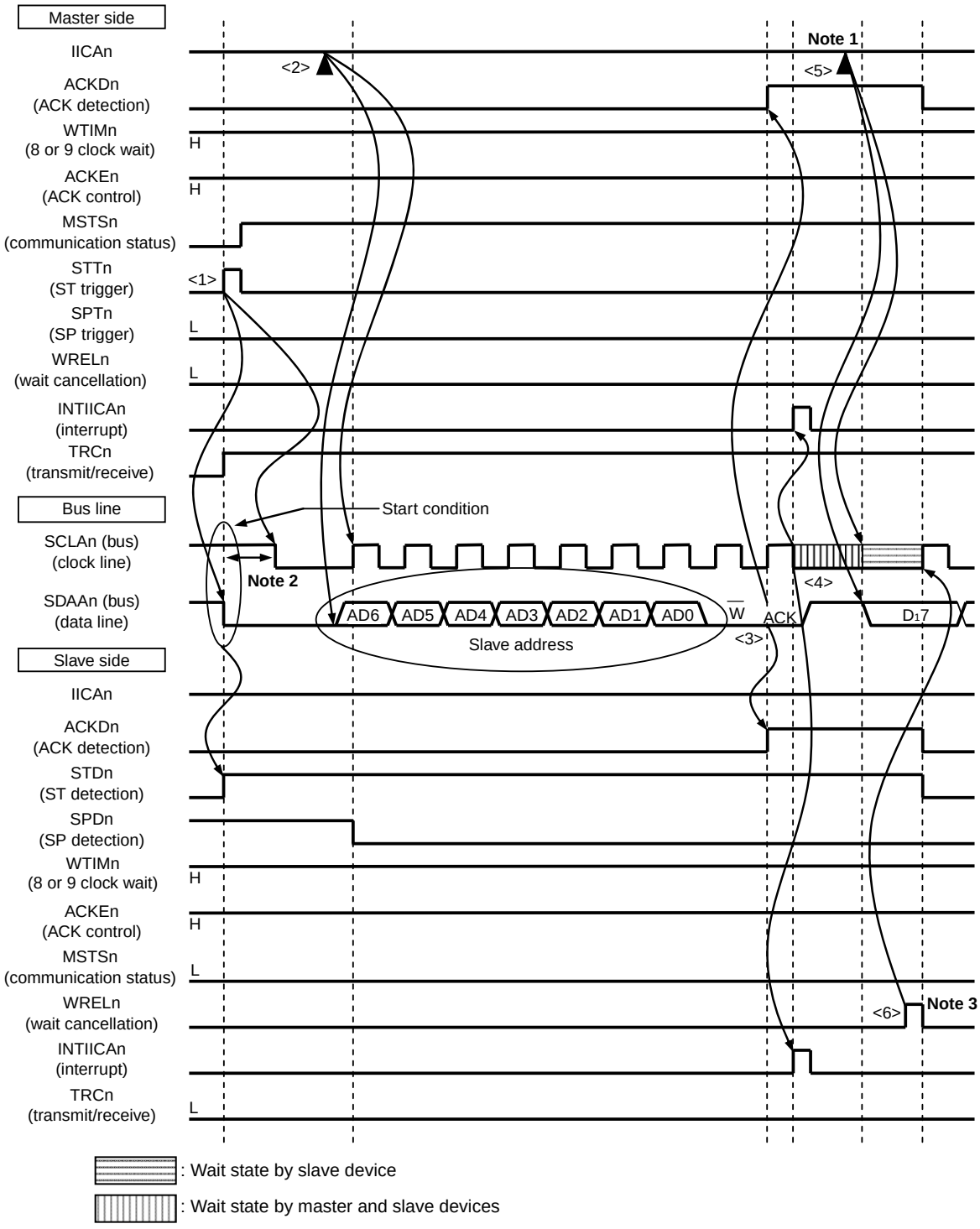
The IICA shift register n (IICAn)'s shift operation is synchronized with the falling edge of the serial clock (SCLAn). The transmit data is transferred to the SO latch and is output (MSB first) via the SDAAn pin.

Data input via the SDAAn pin is captured into IICAn at the rising edge of SCLAn.

**Remark** n = 0

**Figure 19-32. Example of Master to Slave Communication (9-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (1/4)**

**(1) Start condition ~ address ~ data**




**Notes 1.** Write data to IICAn, not setting the WRELn bit, in order to cancel a wait state during transmission by a master device.

**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.

**3.** For releasing wait state during reception of 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 ~ address ~ data in Figure 19-32 are explained below.

<1> The start condition trigger is set by the master device (STTn = 1) and a start condition (i.e. SCLAn = 1 changes SDAAn 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 communication status (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 whose address matched the transmitted slave address sets a wait status (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 wait status that it set by the master device.

<6> If the slave device releases the wait status (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 a wait status. The master device, however, issues the INTIICAn interrupt (end of address transmission) regardless of whether it receives an ACK or NACK.

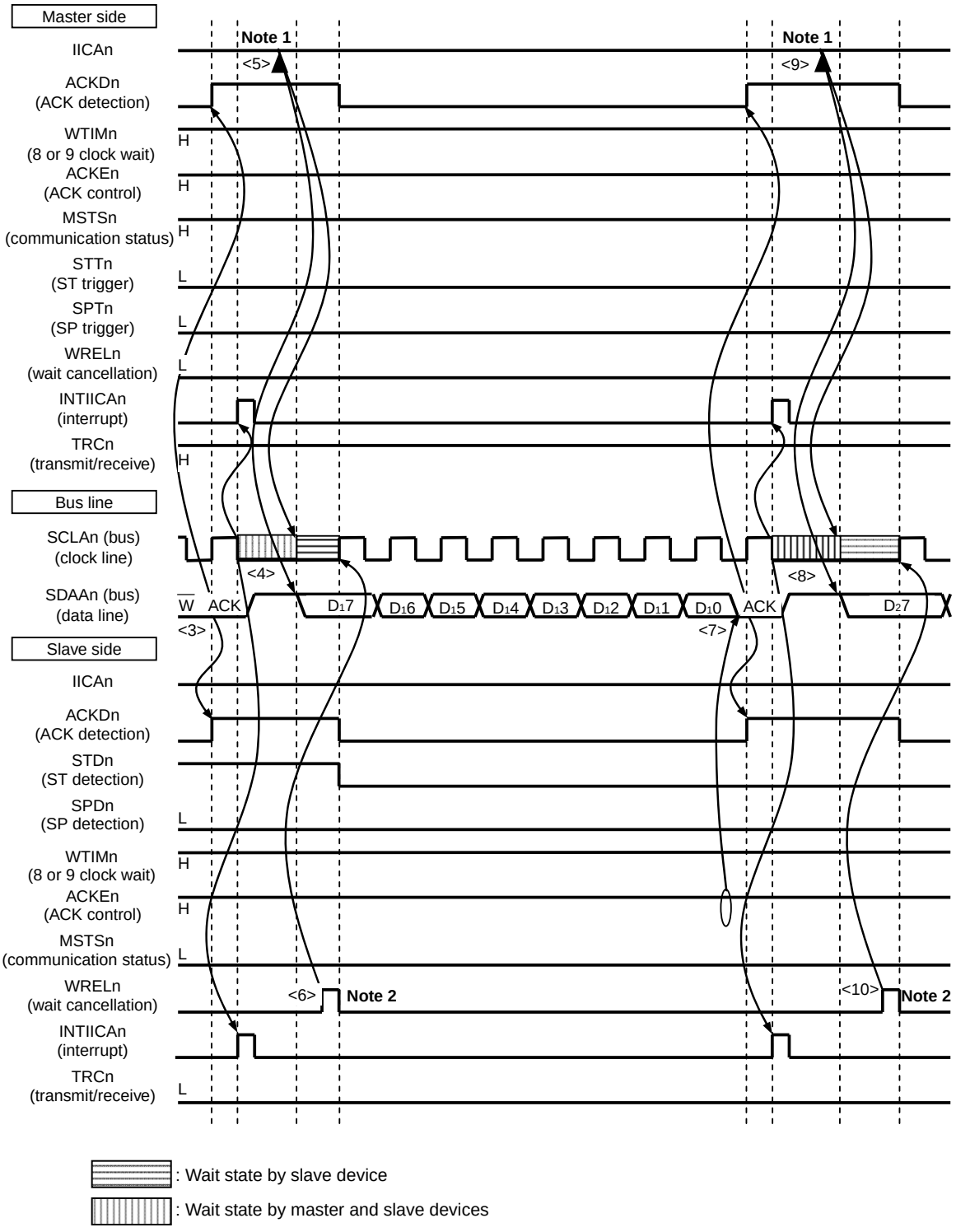
**Remarks 1.** <1> to <15> in Figure 19-32 represent the entire procedure for communicating data using the $I^2C$ bus.

Figure 19-32 (1) Start condition ~ address ~ data shows the processing from <1> to <6>, Figure 19-32 (2) Address ~ data ~ data shows the processing from <3> to <10>, and Figure 19-32 (3) Data ~ data ~ stop condition shows the processing from <7> to <15>.

**2.** n = 0

**Figure 19-32. Example of Master to Slave Communication**
**(9-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (2/4)**

**(2) Address ~ data ~ data**




**Notes 1.** Write data to IICAn, not setting the WRELn bit, in order to cancel a wait state during transmission by a master device.

**2.** For releasing wait state during reception of a slave device, write “FFH” to IICAn or set the WRELn bit.

**Remark** n = 0

The meanings of <3> to <10> in (2) Address ~ data ~ data in Figure 19-32 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 whose address matched the transmitted slave address sets a wait status (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 wait status that it set by the master device.

<6> If the slave device releases the wait status (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 a wait status (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 wait status that it set by the master device.

<10> The slave device reads the received data and releases the wait status (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 a wait status. The master device, however, issues the INTIICAn interrupt (end of address transmission) regardless of whether it receives an ACK or NACK.

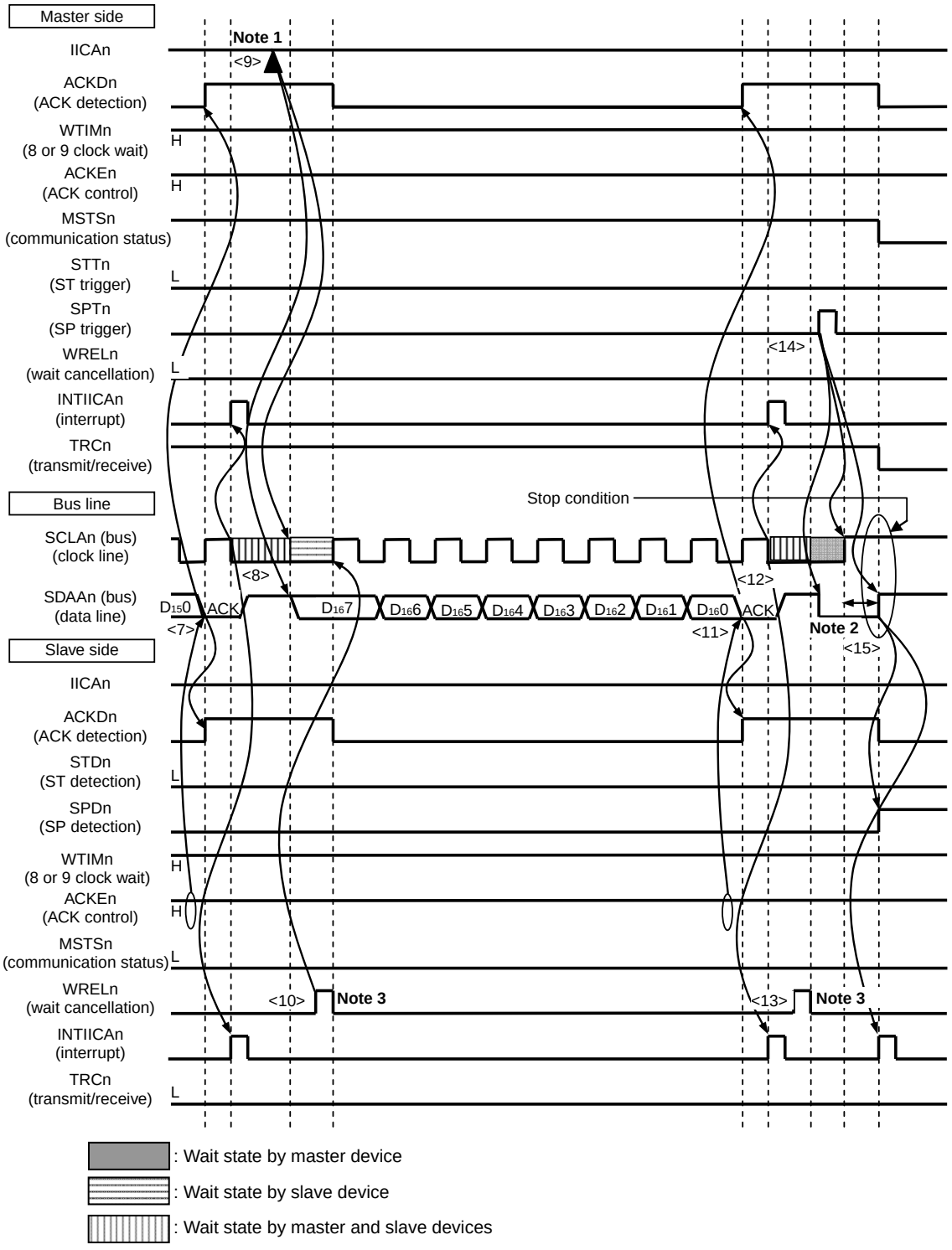
**Remarks 1.** <1> to <15> in Figure 19-32 represent the entire procedure for communicating data using the $I^2C$ bus.

Figure 19-32 (1) Start condition ~ address ~ data shows the processing from <1> to <6>, Figure 19-32 (2) Address ~ data ~ data shows the processing from <3> to <10>, and Figure 19-32 (3) Data ~ data ~ stop condition shows the processing from <7> to <15>.

**2.** n = 0

**Figure 19-32. Example of Master to Slave Communication (9-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (3/4)**

**(3) Data ~ data ~ Stop condition**




**Notes 1.** Write data to IICAn, not setting the WRELn bit, in order to cancel a wait state during transmission by a master device.

**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.

**3.** For releasing wait state during reception of a slave device, write “FFH” to IICAn or set the WRELn bit.

**Remark** n = 0

The meanings of <7> to <15> in (3) Data ~ data ~ stop condition in Figure 19-32 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 a wait status (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 wait status that it set by the master device.

<10> The slave device reads the received data and releases the wait status (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 a wait status (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 wait status (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 changes SDAAn 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).

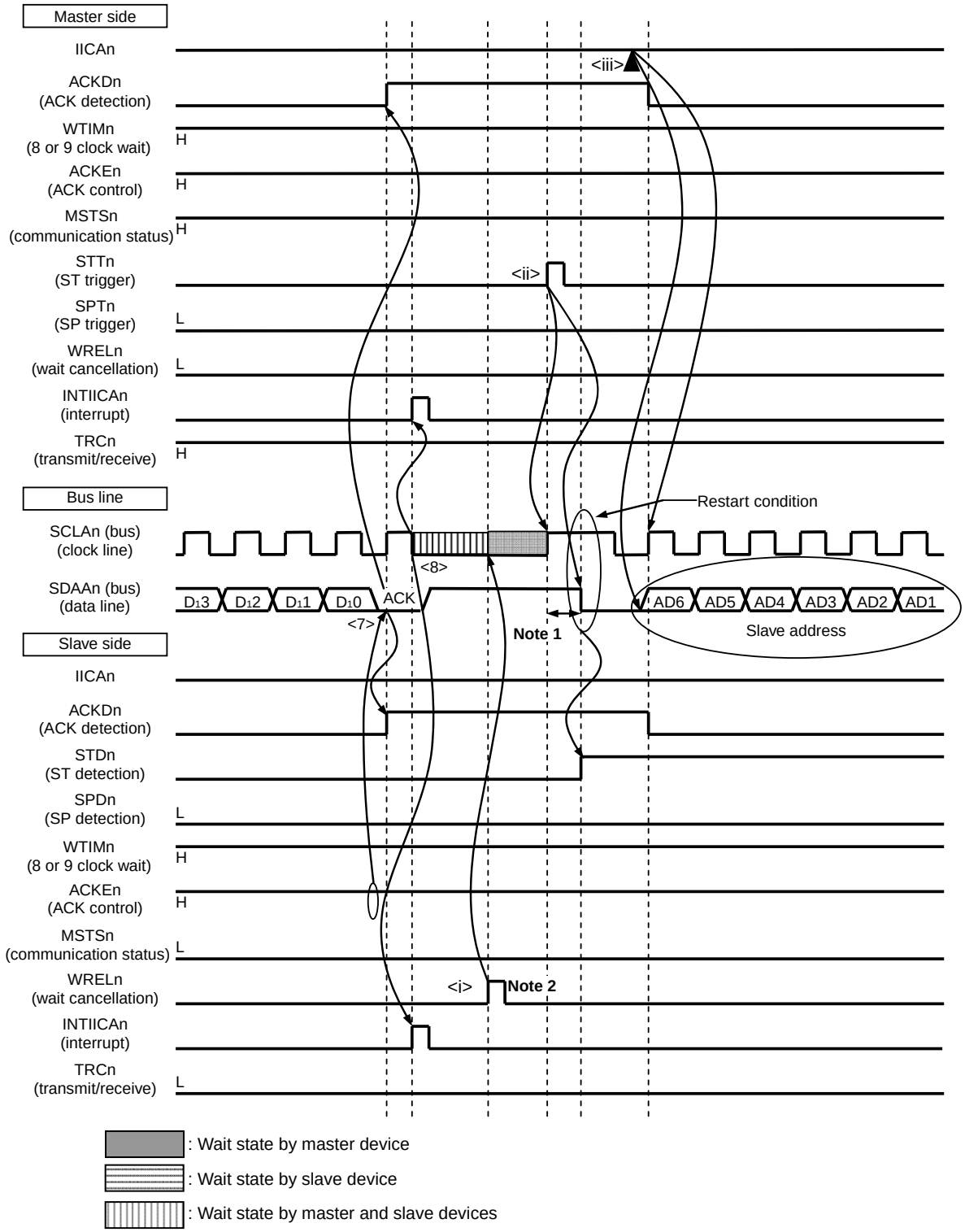
**Remarks 1.** <1> to <15> in Figure 19-32 represent the entire procedure for communicating data using the $I^2C$ bus.

Figure 19-32 (1) Start condition ~ address ~ data shows the processing from <1> to <6>, Figure 19-32 (2) Address ~ data ~ data shows the processing from <3> to <10>, and Figure 19-32 (3) Data ~ data ~ stop condition shows the processing from <7> to <15>.

**2.** n = 0

**Figure 19-32. Example of Master to Slave Communication (9-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (4/4)**

**(4) Data ~ restart condition ~ address**




**Notes 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.

**2.** For releasing wait state during reception of a slave device, write “FFH” to IICAn or set the WRELn bit.

**Remark** n = 0

The following describes the operations in Figure 19-32 (4) Data ~ restart condition ~ address. After the operations in steps <7> and <8>, the operations in steps <i> to <iii> are performed. These steps return the processing to step <iii>, 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 a wait status (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 wait status (WRELn = 1).

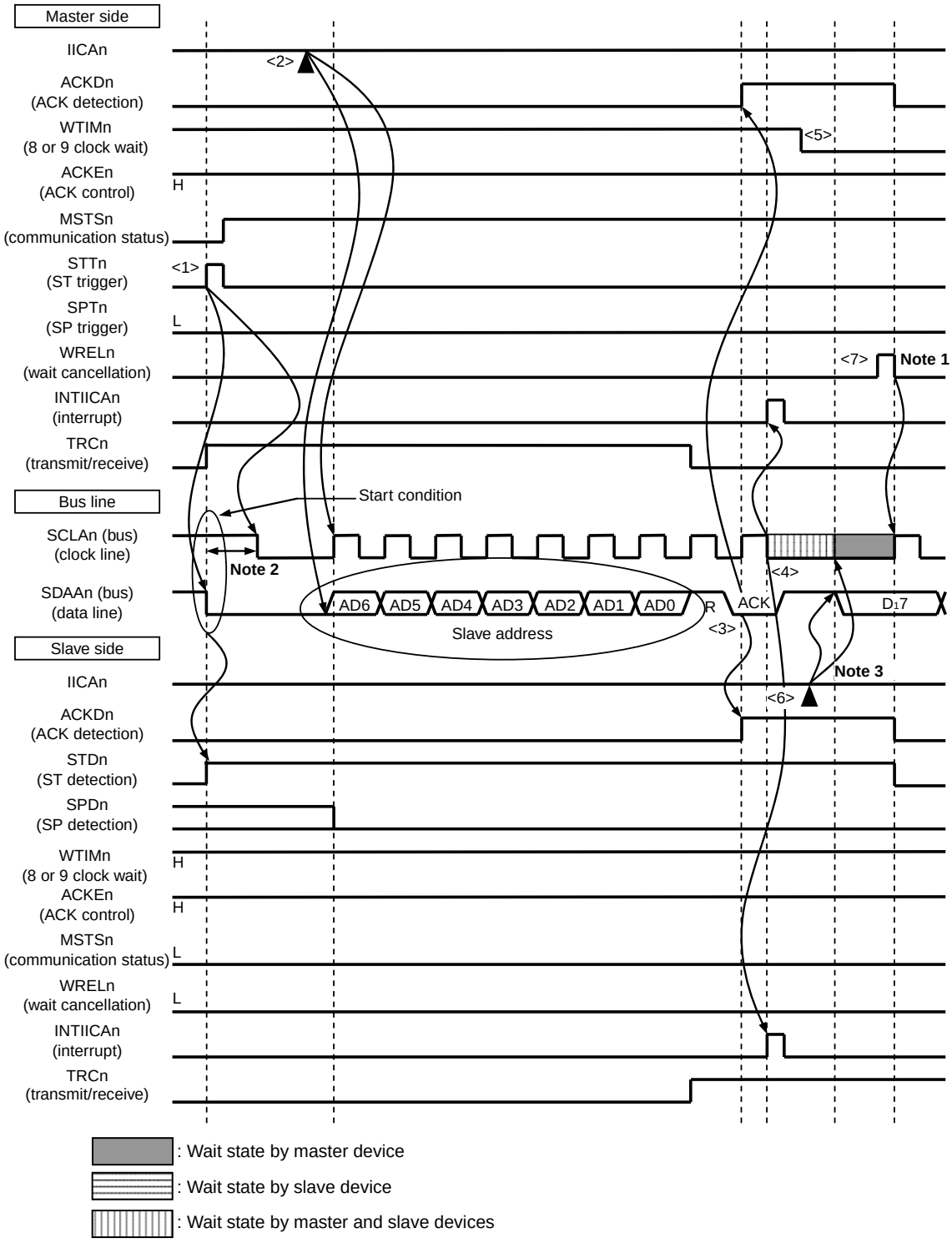
<ii> The start condition trigger is set again by the master device (STTn = 1) and a start condition (i.e. SCLAn =1 changes SDAAn from 1 to 0) is generated once the bus clock line goes high (SCLAn = 1) and the bus data line goes low (SDAAn = 0) after the restart condition setup time has elapsed. When 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 19-33. Example of Slave to Master Communication (8-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (1/3)**

**(1) Start condition ~ address ~ data**




**Notes 1.** For releasing wait state during reception of a master device, write “FFH” to IICAn or set the WRELn bit.

**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.

**3.** Write data to IICAn, not setting the WRELn bit, in order to cancel a wait state during transmission by a slave device.

**Remark** n = 0

The meanings of <1> to <7> in (1) Start condition ~ address ~ data in Figure 19-33 are explained below.

<1> The start condition trigger is set by the master device (STTn = 1) and a start condition (i.e. SCLAn =1 changes SDAAn 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 communication status (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 whose address matched the transmitted slave address sets a wait status (SCLAn = 0) and issues an interrupt (INTIICAn: address match)[Note].

<5> The timing at which the master device sets the wait status changes to the 8th clock (WTIMn = 0).

<6> The slave device writes the data to transmit to the IICAn register and releases the wait status that it set by the slave device.

<7> The master device releases the wait status (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 a wait status. The master device, however, issues the INTIICAn interrupt (end of address transmission) regardless of whether it receives an ACK or NACK.

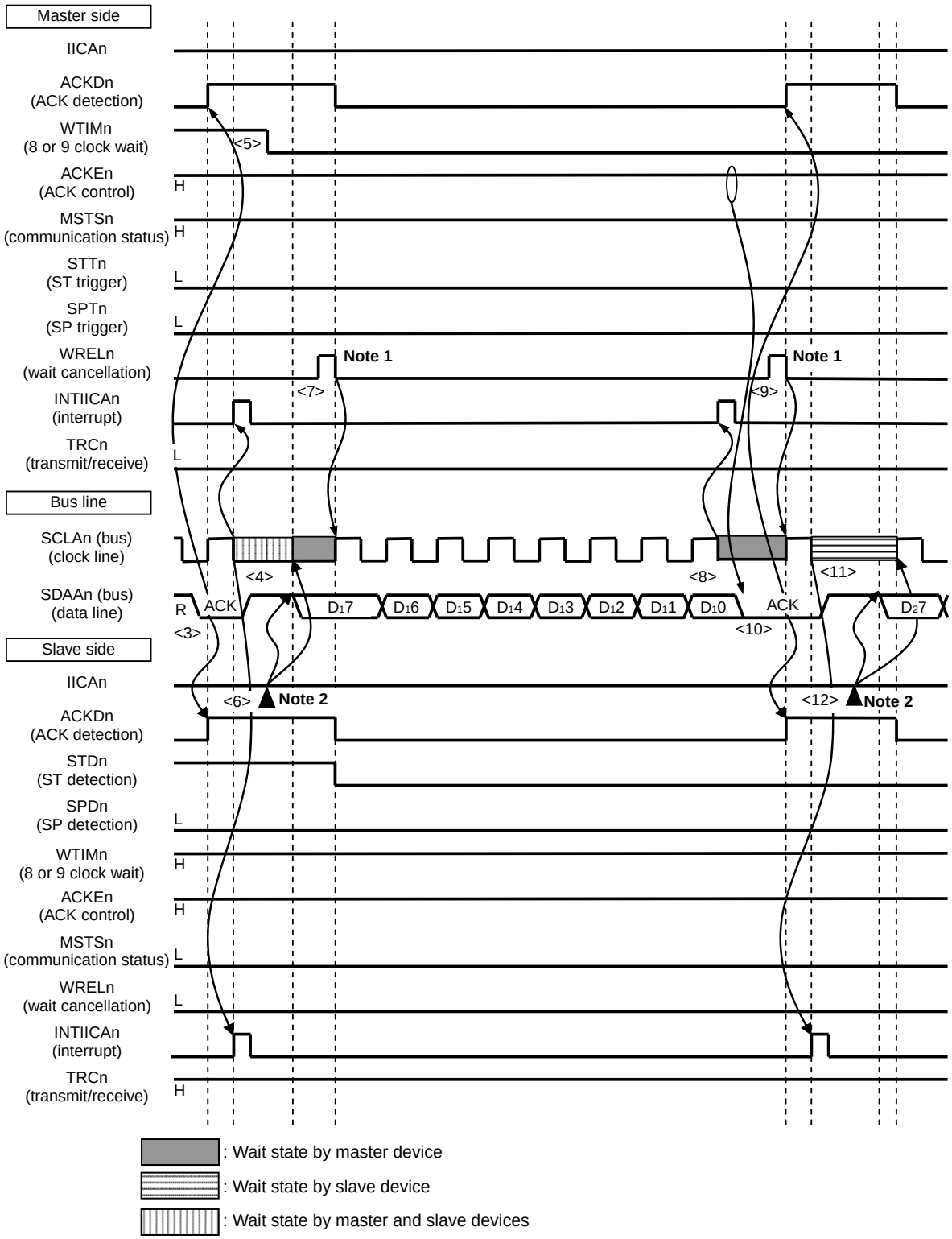
**Remarks 1.** <1> to <19> in Figure 19-33 represent the entire procedure for communicating data using the $I^2C$ bus.

Figure 19-33 (1) Start condition ~ address ~ data shows the processing from <1> to <7>, Figure 19-33 (2) Address ~ data ~ data shows the processing from <3> to <12>, and Figure 19-33 (3) Data ~ data ~ stop condition shows the processing from <8> to <19>.

**2.** n = 0

**Figure 19-33. Example of Slave to Master Communication (8-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (2/3)**

**(2) Address ~ data ~ data**




**Notes 1.** For releasing wait state during reception of a master device, write “FFH” to IICAn or set the WRELn bit.

**2.** Write data to IICAn, not setting the WRELn bit, in order to cancel a wait state during transmission by a slave device.

**Remark** n = 0

The meanings of <3> to <12> in (2) Address ~ data ~ data in Figure 19-33 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 whose address matched the transmitted slave address sets a wait status (SCLAn = 0) and issues an interrupt (INTIICAn: address match)[Note].

<5> The master device changes the timing of the wait status 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 wait status that it set by the slave device.

<7> The master device releases the wait status (WRELn = 1) and starts transferring data from the slave device to the master device.

<8> The master device sets a wait status (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 wait status (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 set a wait status (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 wait status 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 a wait status. The master device, however, issues the INTIICAn interrupt (end of address transmission) regardless of whether it receives an ACK or NACK.

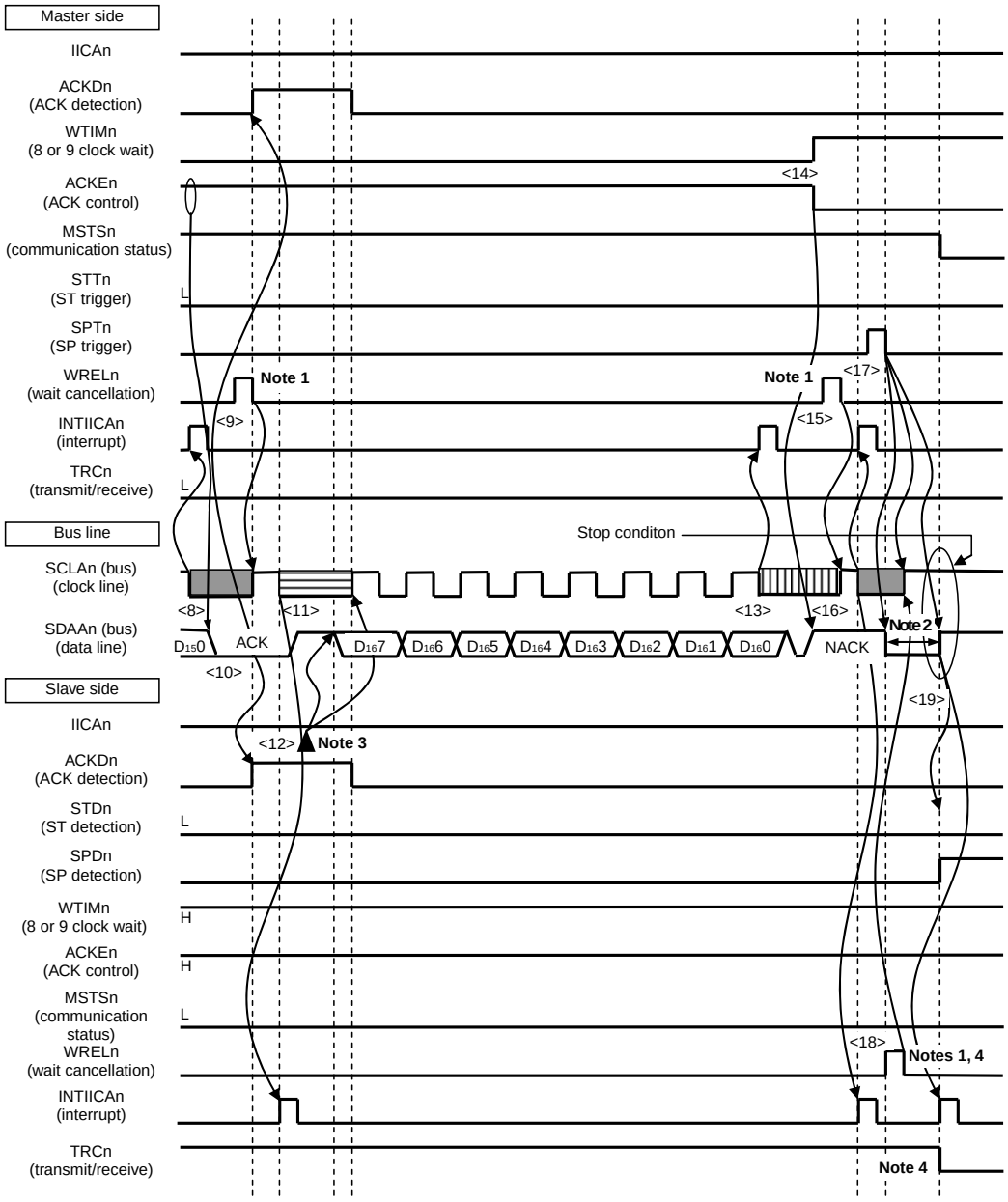
**Remarks 1.** <1> to <19> in Figure 19-33 represent the entire procedure for communicating data using the $I^2C$ bus.

Figure 19-33 (1) Start condition ~ address ~ data shows the processing from <1> to <7>, Figure 19-33 (2) Address ~ data ~ data shows the processing from <3> to <12>, and Figure 19-33 (3) Data ~ data ~ stop condition shows the processing from <8> to <19>.

**2.** n = 0

**Figure 19-33. Example of Slave to Master Communication (8-Clock and 9-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (3/3)**

**(3) Data ~ data ~ stop condition**




**Notes 1.** To cancel a wait state, write “FFH” to IICAn or set the WRELn bit.

**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.

**3.** Write data to IICAn, not setting the WRELn bit, in order to cancel a wait state during transmission by a slave device.

**4.** If a wait state during transmission by a slave device is canceled by setting the WRELn bit, the TRCn bit will be cleared.

**Remark** n = 0

The meanings of <8> to <19> in (3) Data ~ data ~ stop condition in Figure 19-33 are explained below.

<8> The master device sets a wait status (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 wait status (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 set a wait status (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 wait status 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 a wait status (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 wait status to the 9th clock (WTIMn = 1).

<15> If the master device releases the wait status (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 a wait status (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 wait status. 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 wait status (WRELn = 1) to end communication. Once the slave device releases the wait status, 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 changes SDAAn from 0 to 1). The slave device detects the generated stop condition and slave device issue an interrupt (INTIICAn: stop condition).

**Remarks 1.** <1> to <19> in Figure 19-33 represent the entire procedure for communicating data using the $I^2C$ bus.

Figure 19-33 (1) Start condition ~ address ~ data shows the processing from <1> to <7>, Figure 19-33 (2) Address ~ data ~ data shows the processing from <3> to <12>, and Figure 19-33 (3) Data ~ data ~ stop condition shows the processing from <8> to <19>.

**2.** n = 0

# CHAPTER 20 IrDA

The IrDA sends and receives IrDA data communication waveforms in cooperation with the Serial Array Unit (SAU) based on the IrDA (Infrared Data Association) standard 1.0.

## 20.1 Functions of IrDA

Enabling the IrDA function by using the IRE bit in the IRCR register allows encoding and decoding the TxD2 and RxD2 signals of the SAU to the waveforms conforming to the IrDA standard 1.0 (IrTxD and IrRxD pins). Connecting these waveforms to an infrared transmitter/receiver implements infrared data communication conforming to the IrDA standard 1.0 system.

With the IrDA standard 1.0 system, data transfer can be started at 9600 bps and the transfer rate can be changed whenever necessary. Since the IrDA cannot change the transfer rate automatically, the transfer rate should be changed through software.

When the high-speed on-chip oscillator ($f_{IH}$ =24/12/6/3 MHz) is selected, the following baud rates can be selected:

- 115.2 kbps/57.6 kbps/38.4 kbps/19.2 kbps/9600 bps/2400 bps

Figures 20-1 is a block diagram showing cooperation between IrDA and SAU.

**Figure 20-1. Block Diagram Showing Cooperation Between IrDA and SAU**

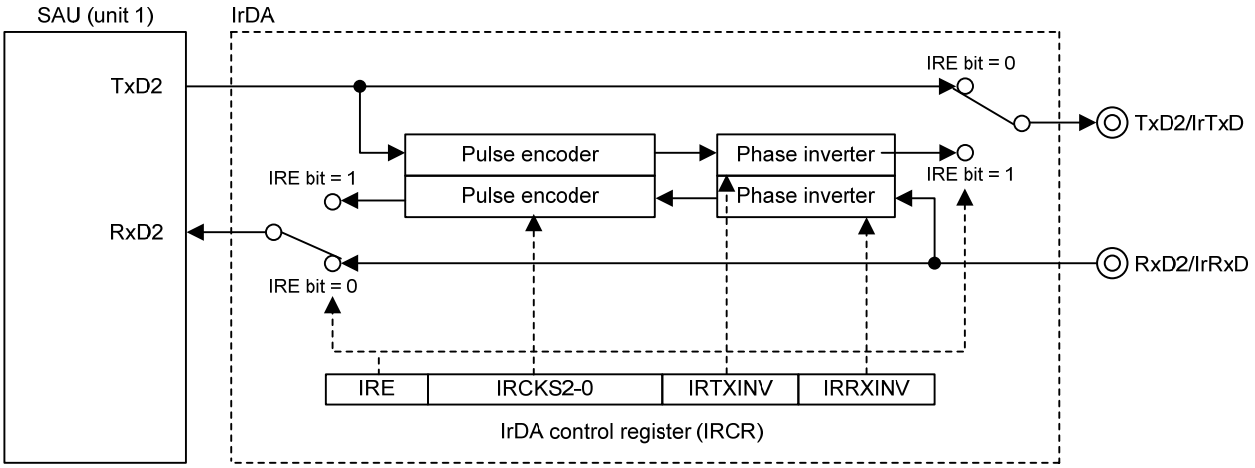



**Table 20-1. IrDA Pin Configuration**

| Pin Name | I/O | Function |
|---|---|---|
| IrTxD | Output | Outputs data to be transmitted. |
| IrRxD | Input | Inputs received data. |

## 20.2 Registers

Table 20-2 lists the IrDA register configuration.

**Table 20-1. IrDA Register Configuration**

| Item | Configuration |
|---|---|
| Control registers | Peripheral enable register 0 (PER0) |
| | IrDA control register (IRCR) |

### 20.2.1 Peripheral enable register 0 (PER0)

This register is used to enable or disable supplying the clock to the peripheral hardware. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

When the IrDA is used, be sure to set bit 6 (IRDAEN) of this register to 1.

The PER0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 20-2. Format of Peripheral Enable Register 0 (PER0)**

Address: F00F0H After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PER0 | RTCWEN | IRDAEN | ADCEN | IICA0EN | SAU1EN | SAU0EN | 0 | TAU0EN |

| IRDAEN | Control of IrDA input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFR used by the IrDA cannot be written.<br>• The IrDA in the reset status. |
| 1 | Enables input clock supply.<br>• SFR used by the IrDA can be read/written. |

**Cautions 1. When setting the IrDA, be sure to set the IRDAEN bit to 1 first. If IRDAEN = 0, writing to a control register of the IrDA is ignored, and all read values are default values.**

**2. Be sure to set bit 1 to "0".**

### 20.2.2 IrDA control register (IRCR)

The IRCR register is used to control the IrDA function. This register is used to switch the polarity of receive data and transmit data, select the IrDA clock, and select the serial I/O pin function (normal serial function or IrDA function).

The IRCR register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 20-3. Format of IrDA Control Register (IRCR)**

Address: F03A0H After reset: 00H R/W

| Symbol | <7> | 6 | 5 | 4 | <3> | <2> | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| IRCR | IRE | IRCKS2 | IRCKS2 | IRCKS0 | IRTXINV | IRRXINV | 0 | 0 |

| IRE | IrDA enable |
|---|---|
| 0 | Serial I/O pins are used for normal serial communication. |
| 1 | Serial I/O pins are used for IrDA data communication. |

| IRCKS2 | IRCKS1 | IRCKS0 | IrDA clock selection |
|---|---|---|---|
| 0 | 0 | 0 | B × 3/16 (B = bit rate) |
| 0 | 0 | 1 | $f_{CLK}$/2 |
| 0 | 1 | 0 | $f_{CLK}$/4 |
| 0 | 1 | 1 | $f_{CLK}$/8 |
| 1 | 0 | 0 | $f_{CLK}$/16 |
| 1 | 0 | 1 | $f_{CLK}$/32 |
| 1 | 1 | 0 | $f_{CLK}$/64 |
| 1 | 1 | 1 | Setting prohibited |

| IRTXINV | IrTxD data polarity switching |
|---|---|
| 0 | Data to be transmitted is output to IrTxD as is. |
| 1 | Data to be transmitted is output to IrTxD after the polarity is inverted. |

| IRRXINV | IrRxD data polarity switching |
|---|---|
| 0 | IrRxD input is used as received data as is. |
| 1 | IrRxD input is used as received data after the polarity is inverted. |

**Cautions 1. Be sure to clear bits 1 and 0 to "0".**

**2. IRCKS[2:0], IRTXINV, and IRRXINV can be set only when IRE bit is 0.**

## 20.3 Operation

### 20.3.1 IrDA communication operation procedure

(1) IrDA Communication Initial configuration flow
Perform IrDA initial configuration as follows:

<1> Set PER0 register bit IRDAEN to 1.
<2> Set the IRCR register.
<3> Set the SAU related registers (refer to the UART mode configuration procedure).

(2) IrDA communication termination flow

<1> Configure the port register and port mode register to set the status of the IrTxD pin after stopping IrDA communication.

**Remark** The output status may change because the IrTxD pin changes to normal serial interface UART data output when IrDA is reset in step 3.

- To output low level from IrTxD pin
  Set port register to 0. Immediately after this, the IrTxD pin is fixed at low level.
- To output high level from IrTxD pin
  Set port register to 1. This will fix IrTxD pin at high level immediately after IrDA reset in step 3.
- To set IrTxD pin to Hi-Z status
  Set port mode register to 1. Immediately after this, IrTxD pin is set to Hi-Z.

<2> Set STm register (SAU related register) bits STm0 and STm1 to 1 (stop SAU channels 0 and 1).
<3> Set PER0 register bit IRDAEN to 0 and reset IrDA.

Do not set STm register bits STm0 and STm1 to 1 or IrDA bit IRE to 0 with any procedure other than the above.

(3) Procedure when IrDA framing error occurs
If a framing error occurs during IrDA communication, the following procedure is necessary to enable receiving of subsequent data.

<1> Set SAU STm register bit STm1 to 1 (stop SAU CH1 operation)
<2> Set SAU SSm register bit SSm1 to 1 (start SAU CH1 operation)

**Remark** m: Unit number (m = 0, 1)

Also refer to the chapter on SAU for information on SAU framing error processing.

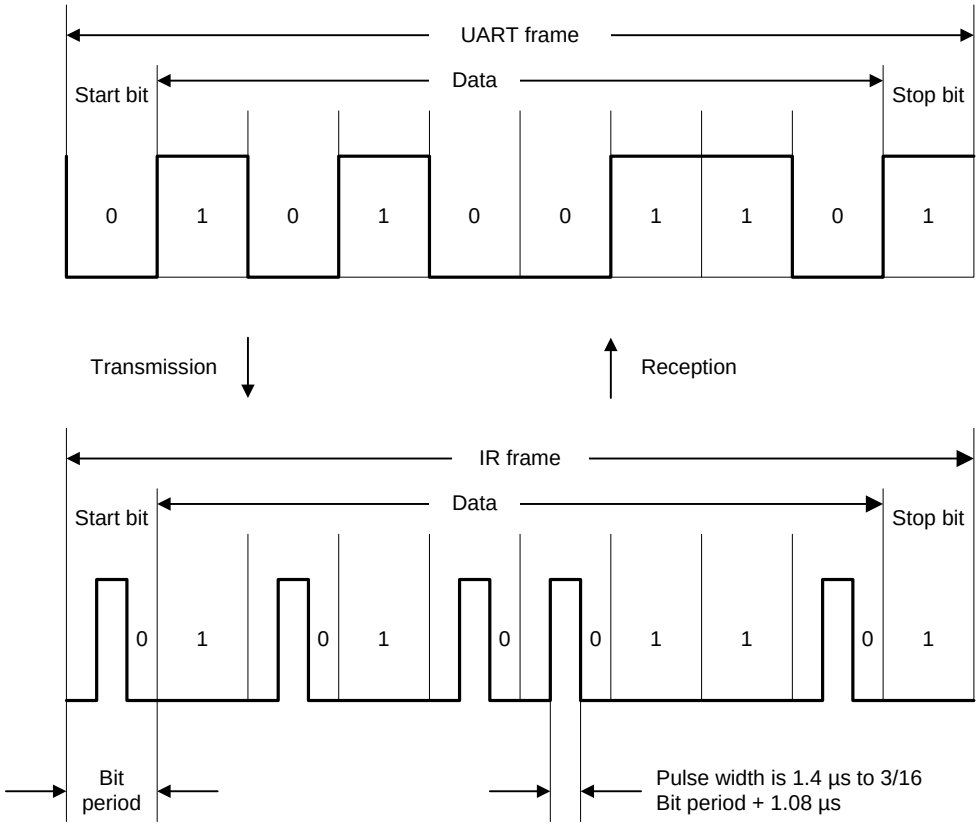
### 20.3.2 Transmission

In transmission, the signals output from the SAU (UART frames) are converted to the IR frame data through the IrDA (see **Figure 20-4**). When IRTXINV bit is 0 and serial data is 0, high-level pulses with the width of 3/16 the bit rate (1-bit width period) are output (initial setting). The high-level pulse width can be changed by using the IRCKS2 to IRCKS0 bits. The standard prescribes that the minimum high-level pulse width should be 1.41 μs and the maximum high-level pulse width be (3/16 + 2.5%) × bit rate or (3/16 × bit rate) + 1.08 μs.

When the CPU/peripheral hardware clock ($f_{CLK}$) is 20 MHz, the high-level pulse width can be 1.41 μs to 1.6 μs.

When serial data is 1, no pulses are output.

**Figure 20-4. IrDA Transmission/Reception**



### 20.3.3 Reception

In reception, the IR frame data is converted to the UART frame data through the IrDA and is input to the SAU.

Low-level data is output when the IRRXINV bit is 0 and a high-level pulse is detected, and high-level data is output when no pulse is detected for 1-bit period. Note that a pulse shorter than 1.41 μs, which is the minimum pulse width, is identified as a low signal.

### 20.3.4 Selecting High-Level Pulse Width

When the pulse width should be shorter than the bit rate × 3/16 for transmission, applicable IRCKS2 to IRCKS0 bit settings (minimum pulse width) and the corresponding high-level pulse widths shown in Table 20-3 can be used.

**Table 20-3. IRCKS2 to IRCKS0 Bit Settings**

| $f_{CLK}$ [MHz] | Item | <Upper Row> Bit Rate [kbps]<br><Lower Row> Bit Rate × 3/16 [μs] | | | | | |
|---|---|---|---|---|---|---|---|
| | | 2.4 | 9.6 | 19.2 | 38.4 | 57.6 | 115.2 |
| | | 78.13 | 19.53 | 9.77 | 4.87 | 3.26 | 1.63 |
| 1 | IRCKS2 to IRCKS0 | 001 | 001 | 001 | –Note 1 | –Note 1 | –Note 1 |
| | High-level pulse width [μs] | 2.00 | 2.00 | 2.00 | –Note 1 | –Note 1 | –Note 1 |
| 2 | IRCKS2 to IRCKS0 | 010 | 010 | 010 | 010 | 010 | –Note 1 |
| | High-level pulse width [μs] | 2.00 | 2.00 | 2.00 | 2.00 | 2.00 | –Note 1 |
| 3 | IRCKS2 to IRCKS0 | 011 | 011 | 011 | 011 | 011 | –Note 1 |
| | High-level pulse width [μs] | 2.67 | 2.67 | 2.67 | 2.67 | 2.67 | –Note 1 |
| 4 | IRCKS2 to IRCKS0 | 011 | 011 | 011 | 011 | 011 | 000Note 2 |
| | High-level pulse width [μs] | 2.00 | 2.00 | 2.00 | 2.00 | 2.00 | 1.50 |
| 6 | IRCKS2 to IRCKS0 | 100 | 100 | 100 | 100 | 100 | 000Note 2 |
| | High-level pulse width [μs] | 2.67 | 2.67 | 2.67 | 2.67 | 2.67 | 1.50 |
| 8 | IRCKS2 to IRCKS0 | 100 | 100 | 100 | 100 | 100 | 000Note 2 |
| | High-level pulse width [μs] | 2.00 | 2.00 | 2.00 | 2.00 | 2.00 | 1.50 |
| 12 | IRCKS2 to IRCKS0 | 101 | 101 | 101 | 101 | 101 | 000Note 2 |
| | High-level pulse width [μs] | 2.67 | 2.67 | 2.67 | 2.67 | 2.67 | 1.50 |
| 16 | IRCKS2 to IRCKS0 | 101 | 101 | 101 | 101 | 101 | 000Note 2 |
| | High-level pulse width [μs] | 2.00 | 2.00 | 2.00 | 2.00 | 2.00 | 1.50 |
| 24 | IRCKS2 to IRCKS0 | 110 | 110 | 110 | 110 | 110 | 000Note 2 |
| | High-level pulse width [μs] | 2.67 | 2.67 | 2.67 | 2.67 | 2.67 | 1.50 |

**Notes 1.** "–" indicates that the communication specification cannot be satisfied.

**2.** The pulse width cannot be shorter than the bit rate × 3/16.

## 20.4 Usage Notes on IrDA

(1) The IrDA function cannot be used to transition to SNOOZE via IrRxD reception.

(2) The input of IrDA operating clock can be disabled/enabled with the peripheral enable register. Initially, register access is disabled because clock input is disabled. Enable IrDA operating clock input with the peripheral enable register before setting the register.

(3) During HALT mode, the IrDA function continues to run.

(4) The use of SAU initialization function (SS bit= 1) is prohibited during IrDA communication.

(5) The IRCR register bits IRRXINV, IRTXINV, and IRCKS[2:0] can be set only when IRE bit is 0.

# CHAPTER 21 LCD CONTROLLER/DRIVER

The number of LCD display function pins of the RL78/I1B differs depending on the product. The following table shows the number of pins of each product.

**Table 21-1. Number of LCD Display Function Pins of Each Product**

| Item | | RL78/I1B | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 80 pins (R5F10MMx (x = G, E)) | | | | | | | | 100 pins (R5F10MPx (x = G, E)) | | | | | | | |
| LCD controller/ driver | | Segment signal outputs: 34 (30)[Note]<br>Common signal outputs: 8 | | | | | | | | Segment signal outputs: 42 (38)[Note]<br>Common signal outputs: 8 | | | | | | | |
| Multiplexed I/O port | | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 |
| Segment | P0 | SEG 37 | SEG 36 | SEG 35 | SEG 34 | SEG 33 | SEG 32 | – | – | – | – | – | – | – | – | – | – |
| | P1 | SEG 11 | SEG 10 | SEG 9 | SEG 8 | SEG 7 | SEG 6 | SEG 5 | SEG 4 | SEG 11 | SEG 10 | SEG 9 | SEG 8 | SEG 7 | SEG 6 | SEG 5 | SEG 4 |
| | P3 | – | – | – | – | SEG 27 | SEG 26 | SEG 25 | SEG 24 | SEG 31 | SEG 30 | SEG 29 | SEG 28 | SEG 27 | SEG 26 | SEG 25 | SEG 24 |
| | P5 | – | – | – | – | – | – | – | – | SEG 39 | SEG 38 | SEG 37 | SEG 36 | SEG 35 | SEG 34 | SEG 33 | SEG 32 |
| | P7 | SEG 23 | SEG 22 | SEG 21 | SEG 20 | SEG 19 | SEG 18 | SEG 17 | SEG 16 | SEG 23 | SEG 22 | SEG 21 | SEG 20 | SEG 19 | SEG 18 | SEG 17 | SEG 16 |
| | P8 | – | – | – | – | SEG 15 | SEG 14 | SEG 13 | SEG 12 | – | – | SEG 41 | SEG 40 | SEG 15 | SEG 14 | SEG 13 | SEG 12 |
| Alternate relationship between COM signal output pins and I/O pots | | – | | | | | | | | | | | | | | | |
| Alternate relationship between COM signal output pins and LCD display function pins | COM4 | SEG0 | | | | | | | | SEG0 | | | | | | | |
| | COM5 | SEG1 | | | | | | | | SEG1 | | | | | | | |
| | COM6 | SEG2 | | | | | | | | SEG2 | | | | | | | |
| | COM7 | SEG3 | | | | | | | | SEG3 | | | | | | | |

**Note** ( ) indicates the number of signal output pins when 8 com is used.

## 21.1 Functions of LCD Controller/Driver

The functions of the LCD controller/driver in the RL78/I1B microcontrollers are as follows.

(1) Waveform A or B selectable
(2) The LCD driver voltage generator can switch internal voltage boosting method, capacitor split method, and external resistance division method.
(3) Automatic output of segment and common signals based on automatic display data register read
(4) The reference voltage to be generated when operating the voltage boost circuit can be selected from 16 steps (contrast adjustment).
(5) LCD blinking is available

Table 21-2 lists the maximum number of pixels that can be displayed in each display mode.

**Table 21-2. Maximum Number of Pixels (1/2)**

**(a) 80-pin products**

| Drive Waveform for LCD Driver | LCD Driver Voltage Generator | Bias Mode | Number of Time Slices | Maximum Number of Pixels |
|---|---|---|---|---|
| Waveform A | External resistance division | – | Static | 34(34 segment signals, 1 common signal) |
| | | 1/2 | 2 | 68 (34 segment signals, 2 common signals) |
| | | | 3 | 102 (34 segment signals, 3 common signals) |
| | | 1/3 | 3 | |
| | | | 4 | 136 (34 segment signals, 4 common signals) |
| | | 1/4 | 8 | 240 (30 segment signals, 8 common signals) |
| | Internal voltage boosting | 1/3 | 3 | 102 (34 segment signals, 3 common signals) |
| | | | 4 | 136 (34 segment signals, 4 common signals) |
| | | 1/4 | 6 | 192 (32 segment signals, 6 common signals) |
| | | | 8 | 240 (30 segment signals, 8 common signals) |
| | Capacitor split | 1/3 | 3 | 102 (34 segment signals, 3 common signals) |
| | | | 4 | 136 (34 segment signals, 4 common signals) |
| Waveform B | External resistance division, internal voltage boosting | 1/3 | 4 | |
| | | 1/4 | 8 | 240 (30 segment signals, 8 common signals) |
| | Capacitor split | 1/3 | 4 | 136 (34 segment signals, 4 common signals) |

**Table 21-2. Maximum Number of Pixels (2/2)**

**(b) 100-pin products**

| Drive Waveform for LCD Driver | LCD Driver Voltage Generator | Bias Mode | Number of Time Slices | Maximum Number of Pixels |
|---|---|---|---|---|
| Waveform A | External resistance division | – | Static | 42 (42 segment signals, 1 common signal) |
| | | 1/2 | 2 | 84 (42 segment signals, 2 common signals) |
| | | | 3 | 126 (42 segment signals, 3 common signals) |
| | | 1/3 | 3 | |
| | | | 4 | 168 (42 segment signals, 4 common signals) |
| | | 1/4 | 8 | 304 (38 segment signals, 8 common signals) |
| | Internal voltage boosting | 1/3 | 3 | 126 (42 segment signals, 3 common signals) |
| | | | 4 | 168 (42 segment signals, 4 common signals) |
| | | 1/4 | 6 | 240 (40 segment signals, 6 common signals) |
| | | | 8 | 304 (38 segment signals, 8 common signals) |
| | Capacitor split | 1/3 | 3 | 126 (42 segment signals, 3 common signals) |
| | | | 4 | 168 (42 segment signals, 4 common signals) |
| Waveform B | External resistance division, internal voltage boosting | 1/3 | 4 | |
| | | 1/4 | 8 | 304 (38 segment signals, 8 common signals) |
| | Capacitor split | 1/3 | 4 | 168 (42 segment signals, 4 common signals) |

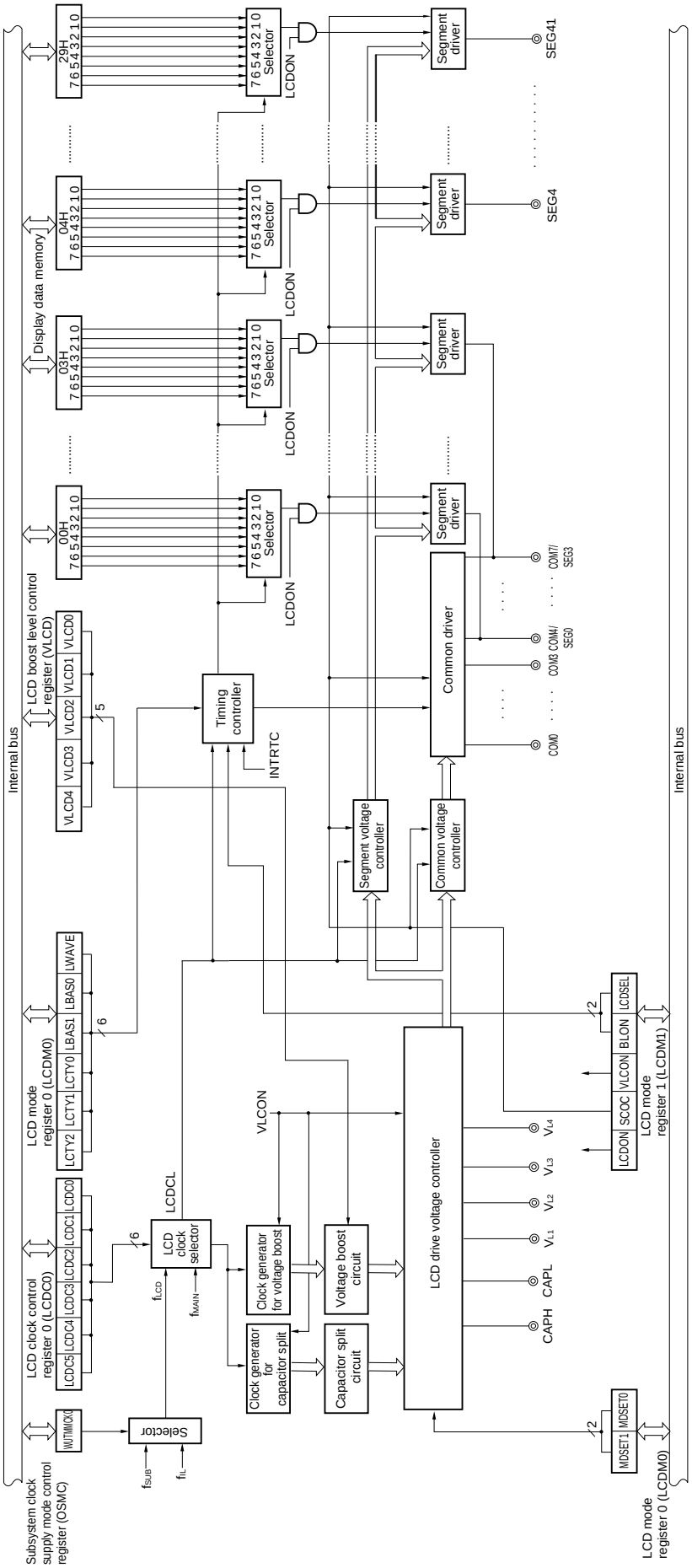
## 21.2 Configuration of LCD Controller/Driver

The LCD controller/driver consists of the following hardware.

**Table 21-3. Configuration of LCD Controller/Driver**

| Item | Configuration |
|---|---|
| Control registers | LCD mode register 0 (LCDM0)<br>LCD mode register 1 (LCDM1)<br>Subsystem clock supply mode control register (OSMC)<br>LCD clock control register 0 (LCDC0)<br>LCD boost level control register (VLCD)<br>LCD input switch control register (ISCLCD)<br>LCD port function registers 0 to 5 (PFSEG0 to PFSEG5)<br>Port mode registers 0, 1, 3, 5, 7, 8 (PM0, PM1, PM3, PM5, PM7, PM8) |

**Figure 21-1. Block Diagram of LCD Controller/Driver**



## 21.3 Registers Controlling LCD Controller/Driver

The following ten registers are used to control the LCD controller/driver.

- LCD mode register 0 (LCDM0)
- LCD mode register 1 (LCDM1)
- Subsystem clock supply mode control register (OSMC)
- LCD clock control register 0 (LCDC0)
- LCD boost level control register (VLCD)
- LCD input switch control register (ISCLCD)
- LCD port function registers 0 to 5 (PFSEG0 to PFSEG5)
- Port mode registers 0, 1, 3, 5, 7, 8 (PM0, PM1, PM3, PM5, PM7, PM8)

### 21.3.1 LCD mode register 0 (LCDM0)

LCDM0 specifies the LCD operation.

This register is set by using an 8-bit memory manipulation instruction.

Reset signal generation sets LCDM0 to 00H.

**Figure 21-2. Format of LCD Mode Register 0 (LCDM0) (1/2)**

Address: FFF40H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| LCDM0 | MDSET1 | MDSET0 | LWAVE | LDTY2 | LDTY1 | LDTY0 | LBAS1 | LBAS0 |

| MDSET1 | MDSET0 | LCD drive voltage generator selection |
|---|---|---|
| 0 | 0 | External resistance division method |
| 0 | 1 | Internal voltage boosting method |
| 1 | 0 | Capacitor split method |
| 1 | 1 | Setting prohibited |

| LWAVE | LCD display waveform selection |
|---|---|
| 0 | Waveform A |
| 1 | Waveform B |

| LDTY2 | LDTY1 | LDTY0 | Selection of time slice of LCD display |
|---|---|---|---|
| 0 | 0 | 0 | Static |
| 0 | 0 | 1 | 2-time slice |
| 0 | 1 | 0 | 3-time slice |
| 0 | 1 | 1 | 4-time slice |
| 1 | 0 | 0 | 6-time slice |
| 1 | 0 | 1 | 8-time slice |
| Other than above | | | Setting prohibited |

**Figure 21-2. Format of LCD Mode Register 0 (LCDM0) (2/2)**

Address: FFF40H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| LCDM0 | MDSET1 | MDSET0 | LWAVE | LDTY2 | LDTY1 | LDTY0 | LBAS1 | LBAS0 |

| LBAS1 | LBAS0 | LCD display bias mode selection |
|---|---|---|
| 0 | 0 | 1/2 bias method |
| 0 | 1 | 1/3 bias method |
| 1 | 0 | 1/4 bias method |
| 1 | 1 | Setting prohibited |

**Cautions 1. Do not rewrite the LCDM0 value while the SCOC bit of the LCDM1 register = 1.**

**2. When "Static" is selected (LDTY2 to LDTY0 bits = 000B), be sure to set the LBAS1 and LBAS0 bits to the default value (00B). Otherwise, the operation will not be guaranteed.**

**3. Only the combinations of display waveform, number of time slices, and bias method shown in Table 21-4 are supported.**
**Combinations of settings not shown in Table 21-4 are prohibited.**

**Table 21-4. Combinations of Display Waveform, Time Slices, Bias Method, and Frame Frequency**

| Display Mode | | | Set Value | | | | | | Driving Voltage Generation Method | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Display Waveform | Number of Time Slices | Bias Mode | LWAVE | LDTY2 | LDTY1 | LDTY0 | LBAS1 | LBAS0 | External Resistance Division | Internal Voltage Boosting | Capacitor Split |
| Waveform A | 8 | 1/4 | 0 | 1 | 0 | 1 | 1 | 0 | O (24 to 128 Hz) | O (24 to 64 Hz) | × |
| Waveform A | 6 | 1/4 | 0 | 1 | 0 | 0 | 1 | 0 | × | O (32 to 86 Hz) | × |
| Waveform A | 4 | 1/3 | 0 | 0 | 1 | 1 | 0 | 1 | O (24 to 128 Hz) | O (24 to 128 Hz) | O (24 to 128 Hz) |
| Waveform A | 3 | 1/3 | 0 | 0 | 1 | 0 | 0 | 1 | O (32 to 128 Hz) | O (32 to 128 Hz) | O (32 to 128 Hz) |
| Waveform A | 3 | 1/2 | 0 | 0 | 1 | 0 | 0 | 0 | O (32 to 128 Hz) | × | × |
| Waveform A | 2 | 1/2 | 0 | 0 | 0 | 1 | 0 | 0 | O (24 to 128 Hz) | × | × |
| Waveform A | Static | | 0 | 0 | 0 | 0 | 0 | 0 | O (24 to 128 Hz) | × | × |
| Waveform B | 8 | 1/4 | 1 | 1 | 0 | 1 | 1 | 0 | O (24 to 128 Hz) | O (24 to 64 Hz) | × |
| Waveform B | 4 | 1/3 | 1 | 0 | 1 | 1 | 0 | 1 | O (24 to 128 Hz) | O (24 to 128 Hz) | O (24 to 128 Hz) |

**Remark** O: Supported
×: Not supported

### 21.3.2 LCD mode register 1 (LCDM1)

LCDM1 enables or disables display operation, voltage boost circuit operation, and capacitor split circuit operation, and specifies the display data area and the low voltage mode.

LCDM1 is set using a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets LCDM1 to 00H.

**Figure 21-3. Format of LCD Mode Register 1 (LCDM1) (1/2)**

Address: FFF41H After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| LCDM1 | LCDON | SCOC | VLCON | BLON | LCDSEL | 0 | 0 | LCDVLM |

| SCOC | LCDON | LCD display enable/disable |
|---|---|---|
| | | When normal liquid crystal waveform (waveform A or B) is output |
| 0 | 0 | Output ground level to segment/common pin |
| 0 | 1 | |
| 1 | 0 | Display off (all segment outputs are deselected.) |
| 1 | 1 | Display on |

| VLCON[Note 1] | Voltage boost circuit or capacitor split circuit operation enable/disable |
|---|---|
| 0 | Stops voltage boost circuit or capacitor split circuit operation |
| 1 | Enables voltage boost circuit or capacitor split circuit operation |

| BLON[Note 2] | LCDSEL | Display data area control |
|---|---|---|
| 0 | 0 | Displaying an A-pattern area data (lower four bits of LCD display data register) |
| 0 | 1 | Displaying a B-pattern area data (higher four bits of LCD display data register) |
| 1 | 0 | Alternately displaying A-pattern and B-pattern area data (blinking display corresponding to the constant-period interrupt (INTRTC) timing of real-time clock 2 (RTC2)) |
| 1 | 1 | |

**Notes 1.** Cannot be set during external resistance division mode.

**2.** When $f_{IL}$ is selected as the LCD source clock ($f_{LCD}$), be sure to set the BLON bit to “0”.

(**Cautions** are listed on the next page.)

**Figure 21-3. Format of LCD Mode Register 1 (LCDM1) (2/2)**

Address: FFF41H After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| LCDM1 | LCDON | SCOC | VLCON | BLON | LCDSEL | 0 | 0 | LCDVLM |

| LCDVLM[Note] | Control of default value of voltage boosting pin |
|---|---|
| 0 | Set when $V_{DD} \geq 2.7$ V |
| 1 | Set when $V_{DD} \leq 4.2$ V |

**Note** A function to set the initial state of the $V_{Lx}$ pin and efficiently boost voltage when using a voltage boosting circuit. Set LCDVLM bit = 0 when $V_{DD}$ at the start of voltage boosting is 2.7 V or more. Set LCDVLM bit = 1 when $V_{DD}$ is 4.2 V or less.
However, when $2.7 \text{ V} \leq V_{DD} \leq 4.2$ V, operation is possible with LCDVLM = 0 or LCDVLM = 1.

**Cautions 1. When the voltage boost circuit is used, set SCOC = 0 and VLCON = 0, and MDSET1, MDSET0 = 00 in order to reduce power consumption when the LCD is not used. When MDSET1, MDSET0 = 01, power is consumed by the internal reference voltage generator.**
**2. When the external resistance division method has been set (MDSET1 and MDSET0 of LCDM0 = 00B) or capacitor split method has been set (MDSET1 and MDSET0 = 10B), set the LCDVLM bit to 0.**
**3. Do not rewrite the VLCON and LCDVLM bits while SCOC = 1.**
**4. Set the BLON and LCDSEL bits to 0 when 8 has been selected as the number of time slices for the display mode.**
**5. To use the internal voltage boosting method, specify the reference voltage by using the VLCD register (select the internal boosting method (by setting the MDSET1 and MDSET0 bits of the LCDM0 register to 01B) if the default reference voltage is used), wait for the reference voltage setup time (5 ms (min.)), and then set the VLCON bit to 1.**

<R> **Remark** RTCE: Bit 7 of real-time clock control register 0 (RTCC0)
RINTE: Bit 15 of interval timer control register (ITMC)
SCOC: Bit 6 of LCD mode register 1 (LCDM1)
VLCON: Bit 5 of LCD mode register 1 (LCDM1)

### 21.3.3 Subsystem clock supply mode control register (OSMC)

OSMC is used to reduce power consumption by stopping as many unnecessary clock functions as possible.

If the RTCLPC bit is set to 1, power consumption can be reduced, because clock supply to the peripheral functions, except real-time clock 2, 12-bit interval timer, clock output/buzzer output, and LCD controller/driver, is stopped in STOP mode or HALT mode while the subsystem clock is selected as the CPU clock.

In addition, the OSMC register can be used to select the operation clock of real-time clock 2, 12-bit interval timer, clock output/buzzer output, LCD controller/driver, and subsystem clock frequency measurement circuit.

This register is set by using an 8-bit memory manipulation instruction.

Reset signal generation sets this register to 00H.

**Figure 21-4. Format of Subsystem clock supply mode Control Register (OSMC)**

Address: F00F3H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| OSMC | RTCLPC | 0 | 0 | WUTMMCK0 | 0 | 0 | 0 | 0 |

| RTCLPC | Setting in STOP mode or HALT mode while subsystem clock is selected as CPU clock |
|---|---|
| 0 | Enables subsystem clock supply to peripheral functions.<br>(See **Tables 24-1** and **24-2** for the peripheral functions whose operations are enabled.) |
| 1 | Stops subsystem clock supply to peripheral functions except real-time clock 2, 12-bit interval timer, clock output/buzzer output, and LCD controller/driver. |

| WUTMMCK0 | Selection of operation clock for real-time clock 2, 12-bit interval timer, and LCD controller/driver. | Selection of clock output from PCLBUZn pin of clock output/buzzer output controller and selection of operation clock for 8-bit interval timer. | Operation of subsystem clock frequency measurement circuit. |
|---|---|---|---|
| 0 | Subsystem clock ($f_{SUB}$) | Selecting the subsystem clock ($f_{SUB}$) is enabled. | Enable |
| 1 | Low-speed on-chip oscillator clock ($f_{IL}$) | Selecting the subsystem clock ($f_{SUB}$) is disabled. | Disable |

**Cautions 1. Be sure to select the subsystem clock (WUTMMCK0 bit = 0) if the subsystem clock is oscillating.**

**2. When WUTMMCK0 is set to “1”, the low-speed on-chip oscillator clock oscillates.**

**3. The subsystem clock and low-speed on-chip oscillator clock can only be switched by using the WUTMMCK0 bit if real-time clock 2, 12-bit interval timer, and LCD controller/driver are all stopped.**

<R>

### 21.3.4 LCD clock control register 0 (LCDC0)

LCDC0 specifies the LCD source clock and LCD clock.

The frame frequency is determined according to the LCD clock and the number of time slices.

This register is set by using an 8-bit memory manipulation instruction.

Reset signal generation sets LCDC0 to 00H.

**Figure 21-5. Format of LCD Clock Control Register 0 (LCDC0)**

Address: FFF42H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| LCDC0 | 0 | 0 | LCDC05 | LCDC04 | LCDC03 | LCDC02 | LCDC01 | LCDC00 |

| LCDC05 | LCDC04 | LCDC03 | LCDC02 | LCDC01 | LCDC00 | LCD clock (LCDCL) | |
|---|---|---|---|---|---|---|---|
| | | | | | | WUTMMCK0 = 0 | WUTMMCK0 = 1 |
| 0 | 0 | 0 | 0 | 0 | 1 | $f_{SUB}/2^2$ | $f_{IL}/2^2$ |
| 0 | 0 | 0 | 0 | 1 | 0 | $f_{SUB}/2^3$ | $f_{IL}/2^3$ |
| 0 | 0 | 0 | 0 | 1 | 1 | $f_{SUB}/2^4$ | $f_{IL}/2^4$ |
| 0 | 0 | 0 | 1 | 0 | 0 | $f_{SUB}/2^5$ | $f_{IL}/2^5$ |
| 0 | 0 | 0 | 1 | 0 | 1 | $f_{SUB}/2^6$ | $f_{IL}/2^6$ |
| 0 | 0 | 0 | 1 | 1 | 0 | $f_{SUB}/2^7$ | $f_{IL}/2^7$ |
| 0 | 0 | 0 | 1 | 1 | 1 | $f_{SUB}/2^8$ | $f_{IL}/2^8$ |
| 0 | 0 | 1 | 0 | 0 | 0 | $f_{SUB}/2^9$ | $f_{IL}/2^9$ |
| 0 | 0 | 1 | 0 | 0 | 1 | $f_{SUB}/2^{10}$ | |
| 0 | 1 | 0 | 0 | 0 | 1 | $f_{MAIN}/2^8$ | |
| 0 | 1 | 0 | 0 | 1 | 0 | $f_{MAIN}/2^9$ | |
| 0 | 1 | 0 | 0 | 1 | 1 | $f_{MAIN}/2^{10}$ | |
| 0 | 1 | 0 | 1 | 0 | 0 | $f_{MAIN}/2^{11}$ | |
| 0 | 1 | 0 | 1 | 0 | 1 | $f_{MAIN}/2^{12}$ | |
| 0 | 1 | 0 | 1 | 1 | 0 | $f_{MAIN}/2^{13}$ | |
| 0 | 1 | 0 | 1 | 1 | 1 | $f_{MAIN}/2^{14}$ | |
| 0 | 1 | 1 | 0 | 0 | 0 | $f_{MAIN}/2^{15}$ | |
| 0 | 1 | 1 | 0 | 0 | 1 | $f_{MAIN}/2^{16}$ | |
| 0 | 1 | 1 | 0 | 1 | 0 | $f_{MAIN}/2^{17}$ | |
| 0 | 1 | 1 | 0 | 1 | 1 | $f_{MAIN}/2^{18}$ | |
| 1 | 0 | 1 | 0 | 1 | 1 | $f_{MAIN}/2^{19}$ | |
| Other than above | | | | | | Setting prohibited | |

**Cautions 1. Be sure to set bits 6 and 7 to “0”.**

**2. Set the frame frequency between 32 and 128 Hz (24 to 128 Hz when $f_{IL}$ is selected). Also, when set to internal voltage boosting method, capacitor spit method, set the LCD clock (LCDCL) to 512 Hz or less (235 Hz or less when $f_{IL}$ is selected).**

**3. Do not set LCDC0 when the SCOC bit of the LCDM1 register is 1.**

**Remark** $f_{MAIN}$: Main system clock frequency

$f_{SUB}$: Subsystem clock frequency

$f_{IL}$: Low-speed on-chip oscillator clock frequency

### 21.3.5 LCD boost level control register (VLCD)

VLCD selects the reference voltage that is to be generated when operating the voltage boost circuit (contrast adjustment). The reference voltage can be selected from 16 steps.

This register is set by using an 8-bit memory manipulation instruction.

Reset signal generation sets VLCD to 04H.

**Figure 21-6. Format of LCD Boost Level Control Register (VLCD)**

Address: FFF43H After reset: 04H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| VLCD | 0 | 0 | 0 | VLCD4 | VLCD3 | VLCD2 | VLCD1 | VLCD0 |

| VLCD4 | VLCD3 | VLCD2 | VLCD1 | VLCD0 | Reference voltage selection (contrast adjustment) | $V_{L4}$ voltage | |
|---|---|---|---|---|---|---|---|
| | | | | | | 1/3 bias method | 1/4 bias method |
| 0 | 0 | 1 | 0 | 0 | 1.00 V (default) | 3.00 V | 4.00 V |
| 0 | 0 | 1 | 0 | 1 | 1.05 V | 3.15 V | 4.20 V |
| 0 | 0 | 1 | 1 | 0 | 1.10 V | 3.30 V | 4.40 V |
| 0 | 0 | 1 | 1 | 1 | 1.15 V | 3.45 V | 4.60 V |
| 0 | 1 | 0 | 0 | 0 | 1.20 V | 3.60 V | 4.80 V |
| 0 | 1 | 0 | 0 | 1 | 1.25 V | 3.75 V | 5.00 V |
| 0 | 1 | 0 | 1 | 0 | 1.30 V | 3.90 V | 5.20 V |
| 0 | 1 | 0 | 1 | 1 | 1.35 V | 4.05 V | Setting prohibited |
| 0 | 1 | 1 | 0 | 0 | 1.40 V | 4.20 V | Setting prohibited |
| 0 | 1 | 1 | 0 | 1 | 1.45 V | 4.35 V | Setting prohibited |
| 0 | 1 | 1 | 1 | 0 | 1.50 V | 4.50 V | Setting prohibited |
| 0 | 1 | 1 | 1 | 1 | 1.55 V | 4.65 V | Setting prohibited |
| 1 | 0 | 0 | 0 | 0 | 1.60 V | 4.80 V | Setting prohibited |
| 1 | 0 | 0 | 0 | 1 | 1.65 V | 4.95 V | Setting prohibited |
| 1 | 0 | 0 | 1 | 0 | 1.70 V | 5.10 V | Setting prohibited |
| 1 | 0 | 0 | 1 | 1 | 1.75 V | 5.25 V | Setting prohibited |
| Other than above | | | | | Setting prohibited | | |

**Cautions 1. The VLCD setting is valid only when the voltage boost circuit is operating.**

**2. Be sure to set bits 5 to 7 to “0”.**

**3. Be sure to change the VLCD value after having stopped the operation of the voltage boost circuit (VLCON = 0).**

**4. To use the internal voltage boosting method, specify the reference voltage by using the VLCD register (select the internal boosting method (by setting the MDSET1 and MDSET0 bits of the LCDM0 register to 01B) if the default reference voltage is used), wait for the reference voltage setup time (5 ms (min.)), and then set VLCON to 1.**

**5. To use the external resistance division method or capacitor split method, use the VLCD register with its initial value (04H).**

### 21.3.6 LCD input switch control register (ISCLCD)

Input to the Schmitt trigger buffer must be disabled until the CAPL/P126, CAPH/P127, and $V_{L3}$/P125 pins are set to operate as LCD function pins in order to prevent through-current from entering.

This register is set by using a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets ISCLCD to 00H.

**Figure 21-7. Format of LCD Input Switch Control Register (ISCLCD)**

Address: F0308H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ISCLCD | 0 | 0 | 0 | 0 | 0 | 0 | ISCVL3 | ISCCAP |

| ISCVL3 | $V_{L3}$/P125 pin Schmitt trigger buffer control |
|---|---|
| 0 | Input invalid |
| 1 | Input valid |

| ISCCAP | CAPL/P126, CAPH/P127 pins Schmitt trigger buffer control |
|---|---|
| 0 | Input invalid |
| 1 | Input valid |

**Cautions 1. If ISCVL3 = 0, set the corresponding port registers as follows:**
**PU125 bit of PU12 register = 0, P125 bit of P12 register = 0**
**2. If ISCCAP = 0, set the corresponding port registers as follows:**
**PU126 bit of PU12 register = 0, P126 bit of P12 register = 0**
**PU127 bit of PU12 register = 0, P127 bit of P12 register = 0**

#### (1) Operation of ports that alternately function as $V_{L3}$, CAPL, and CAPH pins

The functions of the $V_{L3}$/P125, CAPL/P126, and CAPH/P127 pins can be selected by using the LCD input switch control register (ISCLCD), LCD mode register 0 (LCDM0), and port mode register 12 (PM12).

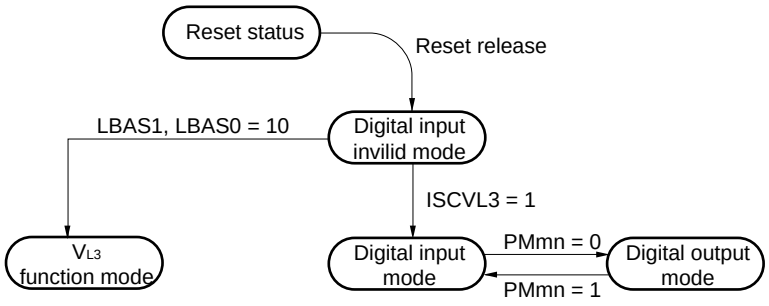
- **$V_{L3}$/P125**

**Table 21-5. Settings of $V_{L3}$/P125 Pin Function**

| Bias Setting (LBAS1 and LBAS0 Bits of LCDM0 Register ) | ISCVL3 Bit of ISCLCD Register | PM125 Bit of PM12 Register | Pin Function | Initial Status |
|---|---|---|---|---|
| Other than 1/4 bias method (LBAS1, LBAS0 = 00 or 01) | 0 | 1 | Digital input invalid mode | √ |
| | 1 | 0 | Digital output mode | – |
| | 1 | 1 | Digital input mode | – |
| 1/4 bias method (LBAS1, LBAS0 = 10) | 0 | 1 | $V_{L3}$ function mode | – |
| Other than above | | | Setting prohibited | |

The following shows the $V_{L3}$/P125 pin function status transitions.

**Figure 21-8.** $V_{L3}$/P125 Pin Function Status Transitions




**Caution Be sure to set the $V_{L3}$ function mode before segment output starts (while SCOC bit of LCD mode register 1 (LCDM1) is 0).**

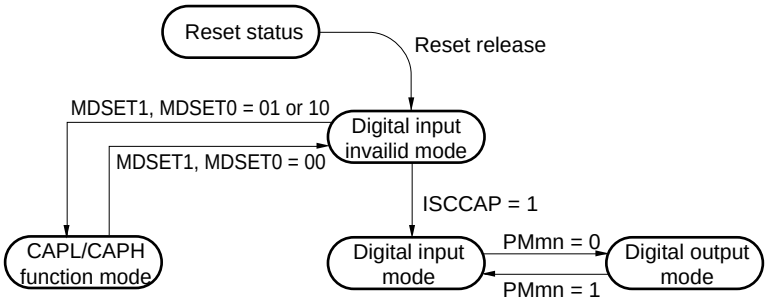
- **CAPL/P126 and CAPH/P127**

**Table 21-6. Settings of CAPL/P126 and CAPH/P127 Pin Functions**

| LCD Drive Voltage Generator (MDSET1 and MDSET0 Bits of LCDM0 Register) | ISCCAP Bit of ISCLCD Register | PM126 and PM127 Bits of PM12 Register | Pin Function | Initial Status |
|---|---|---|---|---|
| External resistance division (MDSET1, MDSET0 = 00) | 0 | 1 | Digital input invalid mode | √ |
| | 1 | 0 | Digital output mode | – |
| | 1 | 1 | Digital input mode | – |
| Internal voltage boosting or capacitor split (MDSET1, MDSET0 = 01 or 10) | 0 | 1 | CAPL/CAPH function mode | – |
| Other than above | | | Setting prohibited | |

The following shows the CAPL/P126 and CAPH/P127 pin function status transitions.

**Figure 21-9.** CAPL/P126 and CAPH/P127 Pin Function Status Transitions




**Caution Be sure to set the CAPL/CAPH function mode before segment output starts (while SCOC bit of LCD mode register 1 (LCDM1) is 0).**

### 21.3.7 LCD port function registers 0 to 5 (PFSEG0 to PFSEG5)

These registers specify whether to use pins P02 to P07, P10 to P17, P30 to P37, P50 to P57, P70 to P77, P80 to P85 as port pins (other than segment output pins) or segment output pins.

These registers are set by using a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets these registers to FFH (PFSEG0 is F0H, PFSEG5 is 02H).

**Remark** The correspondence between the segment output pins (SEGxx) and the PFSEG register (PFSEGxx bits) and the existence of SEGxx pins in each product are shown in **Table 21-7 Segment Output Pins in Each Product and Correspondence with PFSEG Register (PFSEG Bits)**.

**Figure 21-10. Format of LCD Port Function Registers 0 to 5**

Address: F0300H After reset: F0H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFSEG0 | PFSEG07 | PFSEG06 | PFSEG05 | PFSEG04 | 0 | 0 | 0 | 0 |

Address: F0301H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFSEG1 | PFSEG15 | PFSEG14 | PFSEG13 | PFSEG12 | PFSEG11 | PFSEG10 | PFSEG09 | PFSEG08 |

Address: F0302H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFSEG2 | PFSEG23 | PFSEG22 | PFSEG21 | PFSEG20 | PFSEG19 | PFSEG18 | PFSEG17 | PFSEG16 |

Address: F0303H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFSEG3 | PFSEG31Note | PFSEG30Note | PFSEG29Note | PFSEG28Note | PFSEG27 | PFSEG26 | PFSEG25 | PFSEG24 |

Address: F0304H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFSEG4 | PFSEG39Note | PFSEG38Note | PFSEG37 | PFSEG36 | PFSEG35 | PFSEG34 | PFSEG33 | PFSEG32 |

Address: F0305H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PFSEG5 | 0 | 0 | 0 | 0 | 0 | 0 | PFSEG41Note | PFSEG40Note |

| PFSEGxx (xx = 04 to 41) | Port (other than segment output)/segment outputs specification of Pmn pins (mn = 02 to 07, 10 to 17, 30 to 37, 50 to 57, 70 to 77, 80 to 85) |
|---|---|
| 0 | Used as port (other than segment output) |
| 1 | Used as segment output |

<R> **Note** Be sure to set "1" for 80-pin products.

**Caution To use the Pmn pins as segment output pins (PFSEGxx = 1), be sure to set the PUmn bit of the PUm register, POMmn bit of the POMm register, and PIMmn bit of the PIMm register to "0".**

**Table 21-7. Segment Output Pins in Each Product and Correspondence with PFSEG Register (PFSEG Bits)**

| Bit name of PFSEG register | Corresponding SEGxx pins | Alternate port | 100-pin | 80-pin |
|---|---|---|---|---|
| PFSEG04 | SEG4 | P10 | √ | √ |
| PFSEG05 | SEG5 | P11 | √ | √ |
| PFSEG06 | SEG6 | P12 | √ | √ |
| PFSEG07 | SEG7 | P13 | √ | √ |
| PFSEG08 | SEG8 | P14 | √ | √ |
| PFSEG09 | SEG9 | P15 | √ | √ |
| PFSEG10 | SEG10 | P16 | √ | √ |
| PFSEG11 | SEG11 | P17 | √ | √ |
| PFSEG12 | SEG12 | P80 | √ | √ |
| PFSEG13 | SEG13 | P81 | √ | √ |
| PFSEG14 | SEG14 | P82 | √ | √ |
| PFSEG15 | SEG15 | P83 | √ | √ |
| PFSEG16 | SEG16 | P70 | √ | √ |
| PFSEG17 | SEG17 | P71 | √ | √ |
| PFSEG18 | SEG18 | P72 | √ | √ |
| PFSEG19 | SEG19 | P73 | √ | √ |
| PFSEG20 | SEG20 | P74 | √ | √ |
| PFSEG21 | SEG21 | P75 | √ | √ |
| PFSEG22 | SEG22 | P76 | √ | √ |
| PFSEG23 | SEG23 | P77 | √ | √ |
| PFSEG24 | SEG24 | P30 | √ | √ |
| PFSEG25 | SEG25 | P31 | √ | √ |
| PFSEG26 | SEG26 | P32 | √ | √ |
| PFSEG27 | SEG27 | P33 | √ | √ |
| PFSEG28 | SEG28 | P34 | √ | – |
| PFSEG29 | SEG29 | P35 | √ | – |
| PFSEG30 | SEG30 | P36 | √ | – |
| PFSEG31 | SEG31 | P37 | √ | – |
| PFSEG32 | SEG32 | P50 | √ | – |
| | | P02 | – | √ |
| PFSEG33 | SEG33 | P51 | √ | – |
| | | P03 | – | √ |
| PFSEG34 | SEG34 | P52 | √ | – |
| | | P04 | – | √ |
| PFSEG35 | SEG35 | P53 | √ | – |
| | | P05 | – | √ |
| PFSEG36 | SEG36 | P54 | √ | – |
| | | P06 | – | √ |
| PFSEG37 | SEG37 | P55 | √ | – |
| | | P07 | – | √ |
| PFSEG38 | SEG38 | P56 | √ | – |
| PFSEG39 | SEG39 | P57 | √ | – |
| PFSEG40 | SEG40 | P84 | √ | – |
| PFSEG41 | SEG41 | P85 | √ | – |

**(1) Operation of ports that alternately function as SEGxx pins**

The functions of ports that also serve as segment output pins (SEGxx) can be selected by using the port mode register (PMxx) and LCD port function registers 0 to 5 (PFSEG0 to PFSEG5).

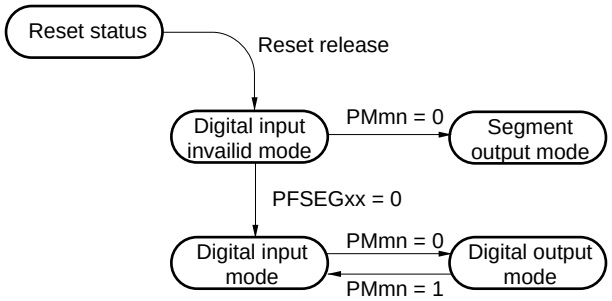
- **P02 to P07, P10 to P17, P30 to P37, P50 to P57, P70 to P77, P80 to P85 (ports that do not serve as analog input pins (ANIxx))**

**Table 21-8. Settings of SEGxx/Port Pin Function**

| PFSEGxx Bit of PFSEG0 to PFSEG5 Registers | PMxx Bit of PMxx Register | Pin Function | Initial Status |
|---|---|---|---|
| 1 | 1 | Digital input invalid mode | √ |
| 0 | 0 | Digital output mode | – |
| 0 | 1 | Digital input mode | – |
| 1 | 0 | Segment output mode | – |

The following shows the SEGxx/Pxx pin function status transitions.

**Figure 21-11.** SEGxx/Pxx Pin Function Status Transitions




**Caution Be sure to set the segment output mode before segment output starts (while SCOC bit of LCD mode register 1 (LCDM1) is 0).**

### 21.3.8 Port mode registers 0, 1, 3, 5, 7, 8 (PM0, PM1, PM3, PM5, PM7, PM8)

These registers specify input/output of ports 0, 1, 5, 7, and 8 in 1-bit units.

When using the ports (such as P10/SEG4) to be shared with the segment output pin for segment output, set the port mode register (PMxx) bit and port register (Pxx) bit corresponding to each port to 0.

**Example:** When using P10/SEG4 for segment output
Set the PM10 bit of port mode register 1 to “0”.
Set the P10 bit of port register 1 to “0”.

These registers are set by using a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets these registers to FFH.

**Figure 21-12. Format of Port Mode Registers 0, 1, 3, 5, 7, 8 (PM0, PM1, PM3, PM5, PM7, PM8)**

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset | R/W |
|---|---|---|---|---|---|---|---|---|---|---|---|
| PM0 | PM07 | PM06 | PM05 | PM04 | PM03 | PM02 | PM01 | PM00 | FFF20H | FFH | R/W |
| PM1 | PM17 | PM16 | PM15 | PM14 | PM13 | PM12 | PM11 | PM10 | FFF21H | FFH | R/W |
| PM3 | 1 | 1 | PM35 | PM34 | PM33 | PM32 | PM31 | PM30 | FFF23H | FFH | R/W |
| PM5 | PM57 | PM56 | PM55 | PM54 | PM53 | PM52 | PM51 | PM50 | FFF25H | FFH | R/W |
| PM7 | PM77 | PM76 | PM75 | PM74 | PM73 | PM72 | PM71 | PM70 | FFF27H | FFH | R/W |
| PM8 | 1 | 1 | 1 | 1 | PM83 | PM82 | PM81 | PM80 | FFF28H | FFH | R/W |

| PMmn | Pmn pin I/O mode selection<br>(m = 0, 1, 3, 5, 7, 8; n = 0 to 7) |
|---|---|
| 0 | Output mode (output buffer on) |
| 1 | Input mode (output buffer off) |

**Remark** The figure shown above presents the format of port mode registers 0, 1, 3, 5, 7, and 8. The format of the port mode register of other products, see **Table 4-3 PMxx, Pxx, PUxx, PIMxx, POMxx registers and the bits mounted on each product**.

## 21.4 LCD Display Data Registers

The LCD display data registers are mapped as shown in Table 21-9. The contents displayed on the LCD can be changed by changing the contents of the LCD display data registers.

**Table 21-9. Relationship Between LCD Display Data Register Contents and Segment/Common Outputs (1/4)**

**(a) Other than 6-time slice and 8-time slice (static, 2-time slice, 3-time slice, and 4-time slice) (1/2)**

| Register Name | Address | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 | 100-pin | 80-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | COM7 | COM6 | COM5 | COM4 | COM3 | COM2 | COM1 | COM0 | | |
| SEG0 | F0400H | SEG0 (B-pattern area) | | | | SEG0 (A-pattern area) | | | | √ | √ |
| SEG1 | F0401H | SEG1 (B-pattern area) | | | | SEG1 (A-pattern area) | | | | √ | √ |
| SEG2 | F0402H | SEG2 (B-pattern area) | | | | SEG2 (A-pattern area) | | | | √ | √ |
| SEG3 | F0403H | SEG3 (B-pattern area) | | | | SEG3 (A-pattern area) | | | | √ | √ |
| SEG4 | F0404H | SEG4 (B-pattern area) | | | | SEG4 (A-pattern area) | | | | √ | √ |
| SEG5 | F0405H | SEG5 (B-pattern area) | | | | SEG5 (A-pattern area) | | | | √ | √ |
| SEG6 | F0406H | SEG6 (B-pattern area) | | | | SEG6 (A-pattern area) | | | | √ | √ |
| SEG7 | F0407H | SEG7 (B-pattern area) | | | | SEG7 (A-pattern area) | | | | √ | √ |
| SEG8 | F0408H | SEG8 (B-pattern area) | | | | SEG8 (A-pattern area) | | | | √ | √ |
| SEG9 | F0409H | SEG9 (B-pattern area) | | | | SEG9 (A-pattern area) | | | | √ | √ |
| SEG10 | F040AH | SEG10 (B-pattern area) | | | | SEG10 (A-pattern area) | | | | √ | √ |
| SEG11 | F040BH | SEG11 (B-pattern area) | | | | SEG11 (A-pattern area) | | | | √ | √ |
| SEG12 | F040CH | SEG12 (B-pattern area) | | | | SEG12 (A-pattern area) | | | | √ | √ |
| SEG13 | F040DH | SEG13 (B-pattern area) | | | | SEG13 (A-pattern area) | | | | √ | √ |
| SEG14 | F040EH | SEG14 (B-pattern area) | | | | SEG14 (A-pattern area) | | | | √ | √ |
| SEG15 | F040FH | SEG15 (B-pattern area) | | | | SEG15 (A-pattern area) | | | | √ | √ |
| SEG16 | F0410H | SEG16 (B-pattern area) | | | | SEG16 (A-pattern area) | | | | √ | √ |
| SEG17 | F0411H | SEG17 (B-pattern area) | | | | SEG17 (A-pattern area) | | | | √ | √ |
| SEG18 | F0412H | SEG18 (B-pattern area) | | | | SEG18 (A-pattern area) | | | | √ | √ |
| SEG19 | F0413H | SEG19 (B-pattern area) | | | | SEG19 (A-pattern area) | | | | √ | √ |
| SEG20 | F0414H | SEG20 (B-pattern area) | | | | SEG20 (A-pattern area) | | | | √ | √ |
| SEG21 | F0415H | SEG21 (B-pattern area) | | | | SEG21 (A-pattern area) | | | | √ | √ |
| SEG22 | F0416H | SEG22 (B-pattern area) | | | | SEG22 (A-pattern area) | | | | √ | √ |
| SEG23 | F0417H | SEG23 (B-pattern area) | | | | SEG23 (A-pattern area) | | | | √ | √ |
| SEG24 | F0418H | SEG24 (B-pattern area) | | | | SEG24 (A-pattern area) | | | | √ | √ |
| SEG25 | F0419H | SEG25 (B-pattern area) | | | | SEG25 (A-pattern area) | | | | √ | √ |
| SEG26 | F041AH | SEG26 (B-pattern area) | | | | SEG26 (A-pattern area) | | | | √ | √ |
| SEG27 | F041BH | SEG27 (B-pattern area) | | | | SEG27 (A-pattern area) | | | | √ | √ |
| SEG28 | F041CH | SEG28 (B-pattern area) | | | | SEG28 (A-pattern area) | | | | √ | – |
| SEG29 | F041DH | SEG29 (B-pattern area) | | | | SEG29 (A-pattern area) | | | | √ | – |
| SEG30 | F041EH | SEG30 (B-pattern area) | | | | SEG30 (A-pattern area) | | | | √ | – |
| SEG31 | F041FH | SEG31 (B-pattern area) | | | | SEG31 (A-pattern area) | | | | √ | – |
| SEG32 | F0420H | SEG32 (B-pattern area) | | | | SEG32 (A-pattern area) | | | | √ | √ |
| SEG33 | F0421H | SEG33 (B-pattern area) | | | | SEG33 (A-pattern area) | | | | √ | √ |

**Table 21-9. Relationship Between LCD Display Data Register Contents and Segment/Common Outputs (2/4)**

**(a) Other than 6-time slice and 8-time slice (static, 2-time slice, 3-time slice, and 4-time slice) (2/2)**

| Register Name | Address | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 | 100-pin | 80-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | COM7 | COM6 | COM5 | COM4 | COM3 | COM2 | COM1 | COM0 | | |
| SEG34 | F0422H | SEG34 (B-pattern area) | | | | SEG34 (A-pattern area) | | | | √ | √ |
| SEG35 | F0423H | SEG35 (B-pattern area) | | | | SEG35 (A-pattern area) | | | | √ | √ |
| SEG36 | F0424H | SEG36 (B-pattern area) | | | | SEG36 (A-pattern area) | | | | √ | √ |
| SEG37 | F0425H | SEG37 (B-pattern area) | | | | SEG37 (A-pattern area) | | | | √ | √ |
| SEG38 | F0426H | SEG38 (B-pattern area) | | | | SEG38 (A-pattern area) | | | | √ | – |
| SEG39 | F0427H | SEG39 (B-pattern area) | | | | SEG39 (A-pattern area) | | | | √ | – |
| SEG40 | F0428H | SEG40 (B-pattern area) | | | | SEG40 (A-pattern area) | | | | √ | – |
| SEG41 | F0429H | SEG41 (B-pattern area) | | | | SEG41 (A-pattern area) | | | | √ | – |

**Remark** √: Supported, –: Not supported

**Table 21-9. Relationship Between LCD Display Data Register Contents and Segment/Common Outputs (3/4)**

**(b) 6-time slice and 8-time slice (1/2)**

| Register Name | Address | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 | 100-pin | 80-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | COM7 | COM6 | COM5 | COM4 | COM3 | COM2 | COM1 | COM0 | | |
| SEG0 | F0400H | SEG0[Note] | | | | | | | | √ | √ |
| SEG1 | F0401H | SEG1[Note] | | | | | | | | √ | √ |
| SEG2 | F0402H | SEG2[Note] | | | | | | | | √ | √ |
| SEG3 | F0403H | SEG3[Note] | | | | | | | | √ | √ |
| SEG4 | F0404H | SEG4 | | | | | | | | √ | √ |
| SEG5 | F0405H | SEG5 | | | | | | | | √ | √ |
| SEG6 | F0406H | SEG6 | | | | | | | | √ | √ |
| SEG7 | F0407H | SEG7 | | | | | | | | √ | √ |
| SEG8 | F0408H | SEG8 | | | | | | | | √ | √ |
| SEG9 | F0409H | SEG9 | | | | | | | | √ | √ |
| SEG10 | F040AH | SEG10 | | | | | | | | √ | √ |
| SEG11 | F040BH | SEG11 | | | | | | | | √ | √ |
| SEG12 | F040CH | SEG12 | | | | | | | | √ | √ |
| SEG13 | F040DH | SEG13 | | | | | | | | √ | √ |
| SEG14 | F040EH | SEG14 | | | | | | | | √ | √ |
| SEG15 | F040FH | SEG15 | | | | | | | | √ | √ |
| SEG16 | F0410H | SEG16 | | | | | | | | √ | √ |
| SEG17 | F0411H | SEG17 | | | | | | | | √ | √ |
| SEG18 | F0412H | SEG18 | | | | | | | | √ | √ |
| SEG19 | F0413H | SEG19 | | | | | | | | √ | √ |
| SEG20 | F0414H | SEG20 | | | | | | | | √ | √ |
| SEG21 | F0415H | SEG21 | | | | | | | | √ | √ |
| SEG22 | F0416H | SEG22 | | | | | | | | √ | √ |
| SEG23 | F0417H | SEG23 | | | | | | | | √ | √ |
| SEG24 | F0418H | SEG24 | | | | | | | | √ | √ |
| SEG25 | F0419H | SEG25 | | | | | | | | √ | √ |
| SEG26 | F041AH | SEG26 | | | | | | | | √ | √ |
| SEG27 | F041BH | SEG27 | | | | | | | | √ | √ |
| SEG28 | F041CH | SEG28 | | | | | | | | √ | – |
| SEG29 | F041DH | SEG29 | | | | | | | | √ | – |
| SEG30 | F041EH | SEG30 | | | | | | | | √ | – |
| SEG31 | F041FH | SEG31 | | | | | | | | √ | – |
| SEG32 | F0420H | SEG32 | | | | | | | | √ | √ |
| SEG33 | F0421H | SEG33 | | | | | | | | √ | √ |
| SEG34 | F0422H | SEG34 | | | | | | | | √ | √ |
| SEG35 | F0423H | SEG35 | | | | | | | | √ | √ |
| SEG36 | F0424H | SEG36 | | | | | | | | √ | √ |
| SEG37 | F0425H | SEG37 | | | | | | | | √ | √ |
| SEG38 | F0426H | SEG38 | | | | | | | | √ | – |
| SEG39 | F0427H | SEG39 | | | | | | | | √ | – |
| SEG40 | F0428H | SEG40 | | | | | | | | √ | – |

**Table 21-9. Relationship Between LCD Display Data Register Contents and Segment/Common Outputs (4/4)**

**(b) 6-time slice and 8-time slice (2/2)**

| Register Name | Address | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 | 100-pin | 80-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | COM7 | COM6 | COM5 | COM4 | COM3 | COM2 | COM1 | COM0 | | |
| SEG41 | F0429H | SEG41 | | | | | | | | √ | – |

**Note** The COM4 to COM7 pins and SEG0 to SEG3 pins are used alternatively.

**Remark** √: Supported, –: Not supported

To use the LCD display data register when the number of time slices is static, two, three, or four, the lower four bits and higher four bits of each address of the LCD display data register become an A-pattern area and a B-pattern area, respectively.

The correspondences between A-pattern area data and COM signals are as follows: bit 0 ⇔ COM0, bit 1 ⇔ COM1, bit 2 ⇔ COM2, and bit 3 ⇔ COM3.

The correspondences between B-pattern area data and COM signals are as follows: bit 4 ⇔ COM0, bit 5 ⇔ COM1, bit 6 ⇔ COM2, and bit 7 ⇔ COM3.

A-pattern area data will be displayed on the LCD panel when BLON = LCDSEL = 0 has been selected, and B-pattern area data will be displayed on the LCD panel when BLON = 0 and LCDSEL = 1 have been selected.

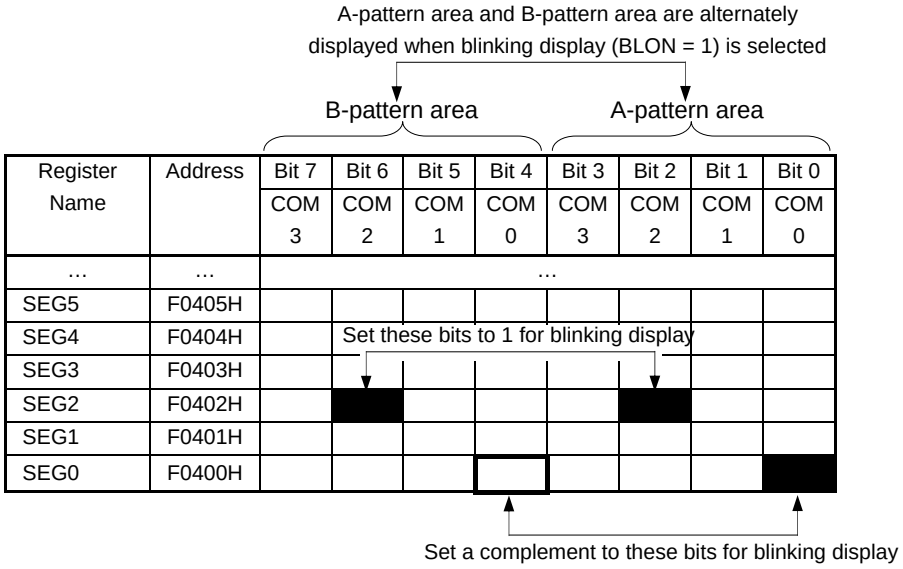
## 21.5 Selection of LCD Display Register

With RL78/I1B, to use the LCD display data registers when the number of time slices is static, two, three, or four, the LCD display data register can be selected from the following three types, according to the BLON and LCDSEL bit settings.

- Displaying an A-pattern area data (lower four bits of LCD display data register)
- Displaying a B-pattern area data (higher four bits of LCD display data register)
- Alternately displaying A-pattern and B-pattern area data (blinking display corresponding to the constant-period interrupt timing of real-time clock 2 (RTC2))

**Caution When the number of time slices is six or eight, LCD display data registers (A-pattern, B-pattern, or blinking display) cannot be selected.**

**Figure 21-13. Example of Setting LCD Display Registers When Pattern Is Changed**



### 21.5.1 A-pattern area and B-pattern area data display

When BLON = LCDSEL = 0, A-pattern area (lower four bits of the LCD display data register) data will be output as the LCD display register.

When BLON = 0, and LCDSEL = 1, B-pattern area (higher four bits of the LCD display data register) data will be output as the LCD display register.

See **21.4 LCD Display Data Registers** about the display area.

### 21.5.2 Blinking display (Alternately displaying A-pattern and B-pattern area data)

When BLON = 1 has been set, A-pattern and B-pattern area data will be alternately displayed, according to the constant-period interrupt (INTRTC) timing of real-time clock 2 (RTC2). See **CHAPTER 8 REAL-TIME CLOCK 2** about the setting of the RTC constant-period interrupt (INTRTC, 0.5 s setting only) timing.

For blinking display of the LCD, set inverted values to the B-pattern area bits corresponding to the A-pattern area bits. (Example: Set 1 to bit 0 of 00H, and set 0 to bit 4 of F0400H for blinking display.) When not setting blinking display of the LCD, set the same values. (Example: Set 1 to bit 2 of F0402H, and set 1 to bit 6 of F0402H for lighting display.)

See **21.4 LCD Display Data Registers** about the display area.

Next, the timing operation of display switching is shown.

**Figure 21-14. Switching Operation from A-Pattern Display to Blinking Display**

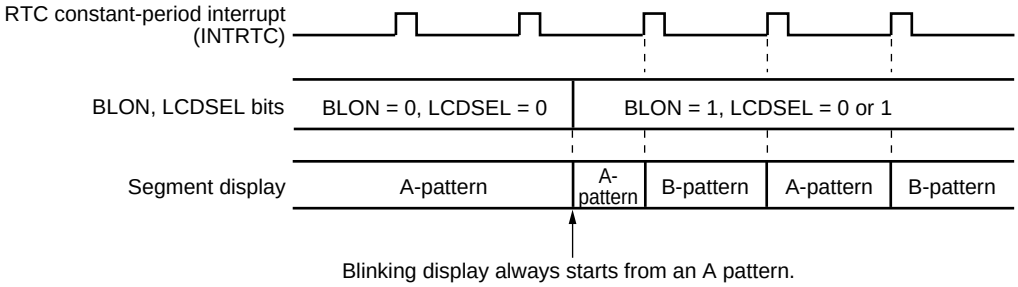
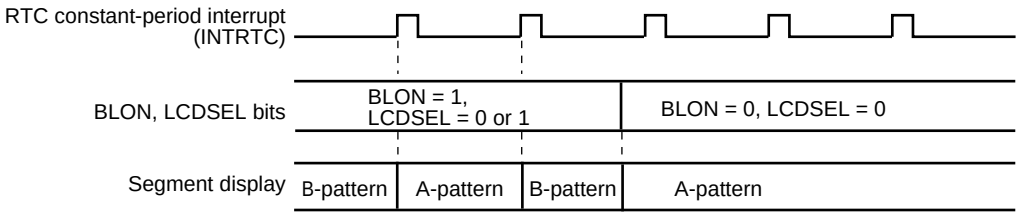



**Figure 21-15. Switching Operation from Blinking Display to A-Pattern Display**



## 21.6 Setting the LCD Controller/Driver

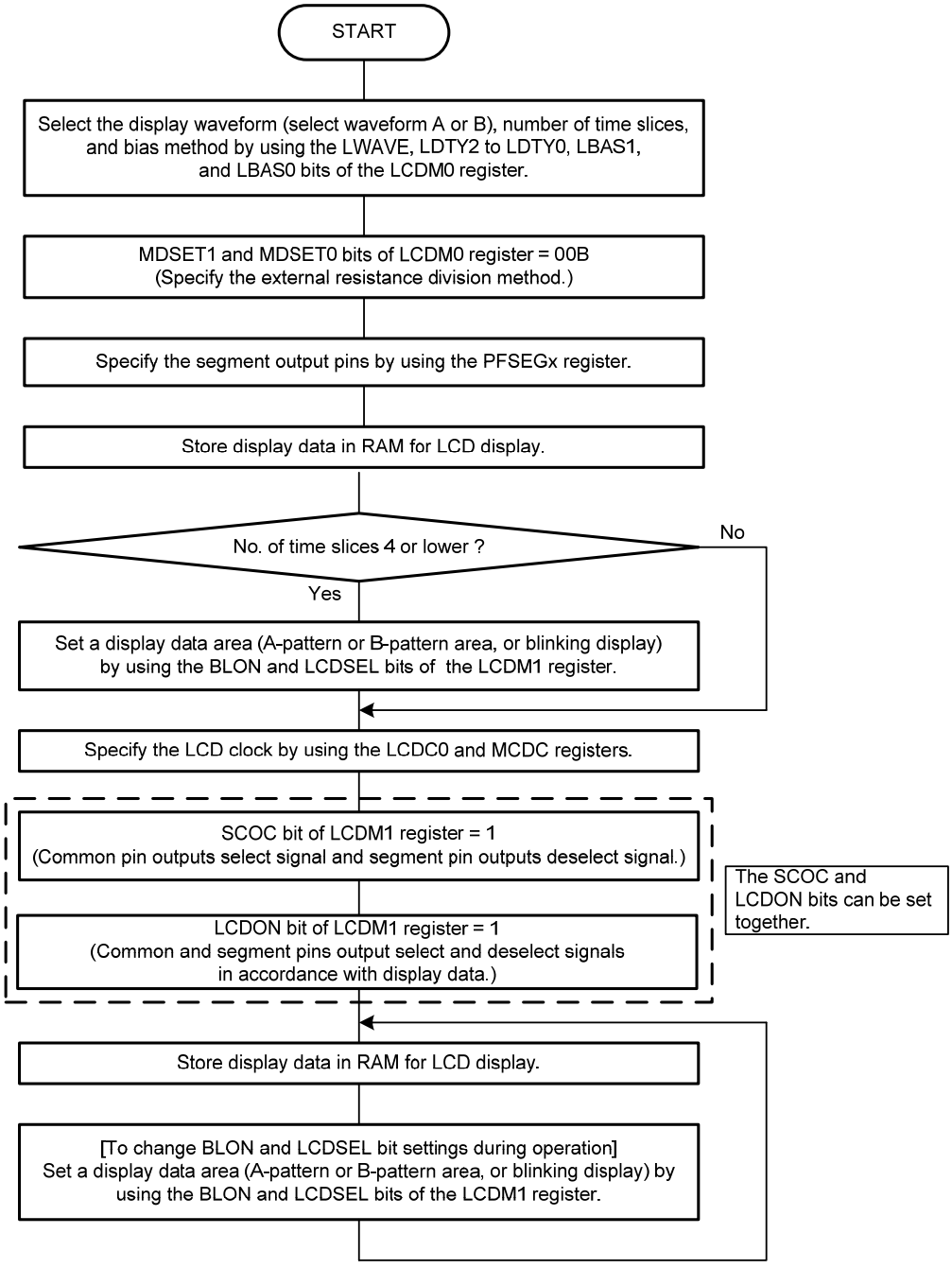
Set the LCD controller/driver using the following procedure.

**Cautions 1. To operate the LCD controller/driver, be sure to follow procedures (1) to (3). Unless these procedures are observed, the operation will not be guaranteed.**

**2. The steps shown in the flowcharts in (1) to (3) are performed by the CPU.**

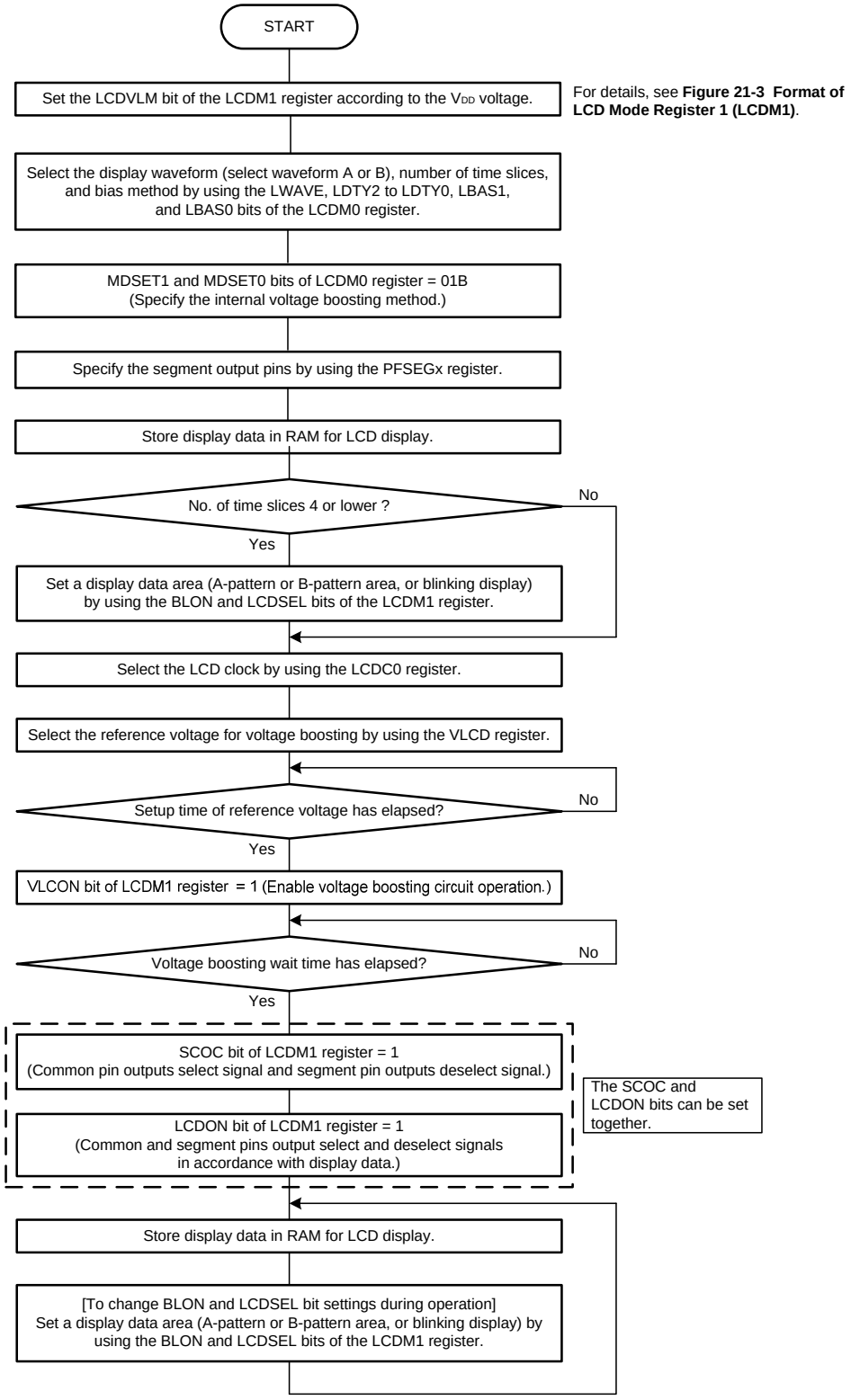
### (1) External resistance division method

**Figure 21-16. External Resistance Division Method Setting Procedure**



### (2) Internal voltage boosting method

**Figure 21-17. Internal Voltage Boosting Method Setting Procedure**

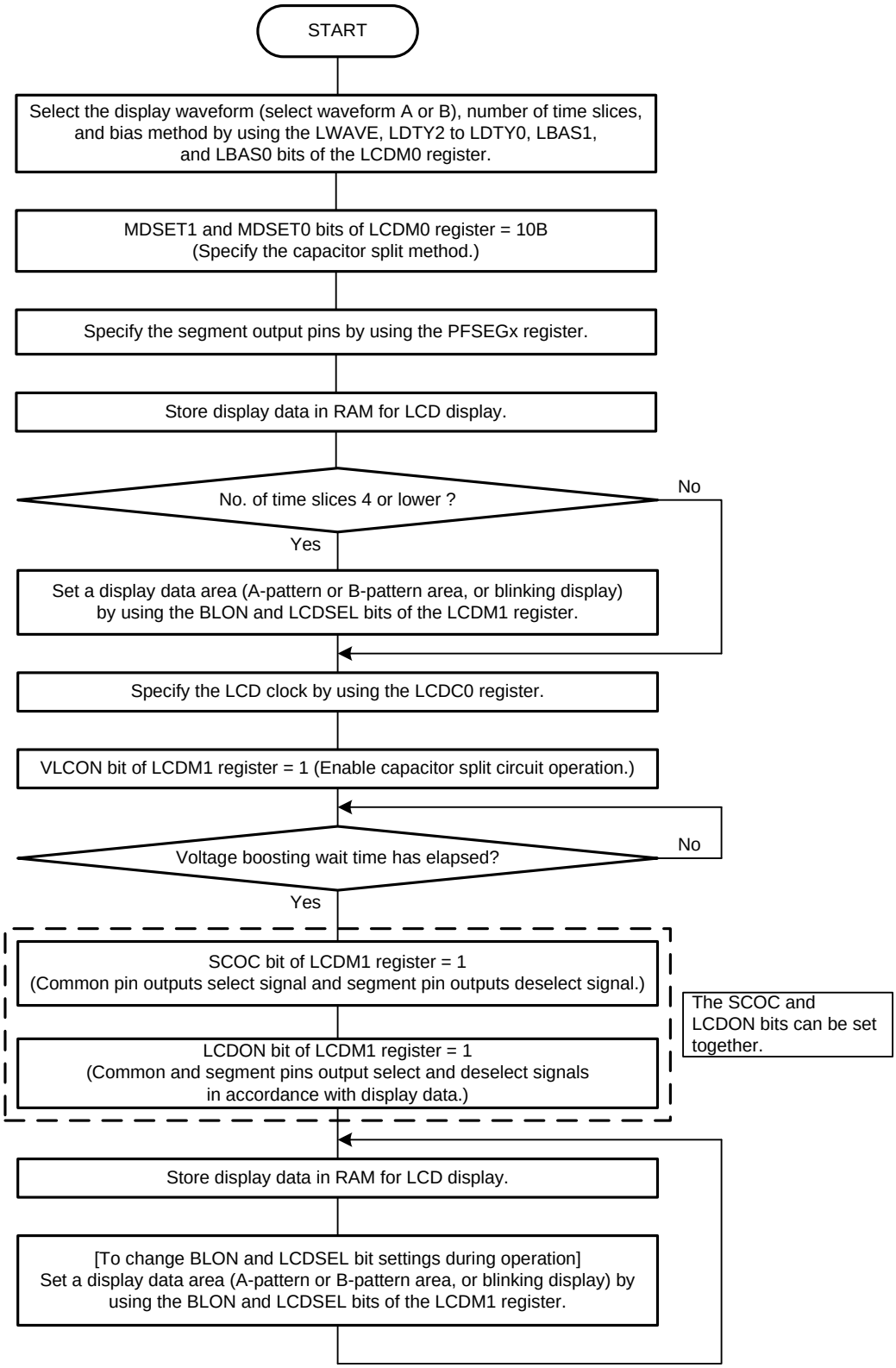



**Cautions 1. Wait until the setup time has elapsed even if not changing the setting of the VLCD register.**

**2. For the specifications of the reference voltage setup time and voltage boosting wait time, see CHAPTER 37 ELECTRICAL SPECIFICATIONS.**

### (3) Capacitor split method

**Figure 21-18. Capacitor Split Method Setting Procedure**

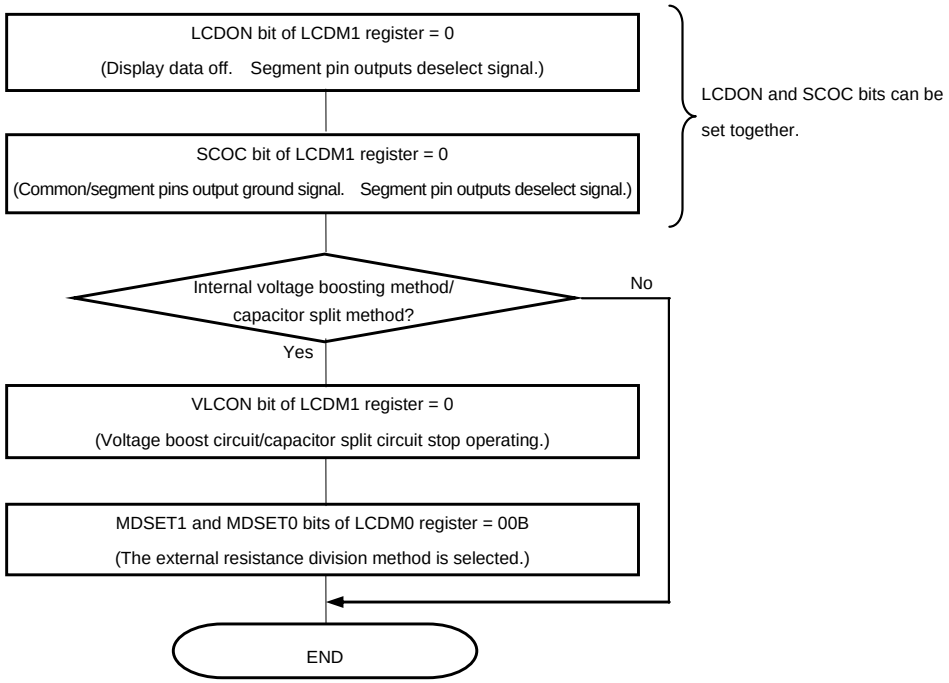



**Caution For the specifications of the voltage boosting wait time, see CHAPTER 37 ELECTRICAL SPECIFICATIONS.**

## 21.7 Operation Stop Procedure

To stop the operation of the LCD while it is displaying waveforms, follow the steps shown in the flowchart below.
The LCD stops operating when the LCDON bit of LCDM1 register and SCOC bit of the LCDM1 register are set to “0”.

**Figure 21-19. Operation Stop Procedure**




**Caution Stopping the voltage boost/capacitor split circuits is prohibited while the display is on (SCOC and LCDON bits of LCDM1 register = 11B). Otherwise, the operation will not be guaranteed. Be sure to turn off display (SCOC and LCDON bits of LCDM1 register = 00B) before stopping the voltage boost/capacitor split circuits (VLCON bit of LCDM1 register = 0).**

## 21.8 Supplying LCD Drive Voltages $V_{L1}$, $V_{L2}$, $V_{L3}$, and $V_{L4}$

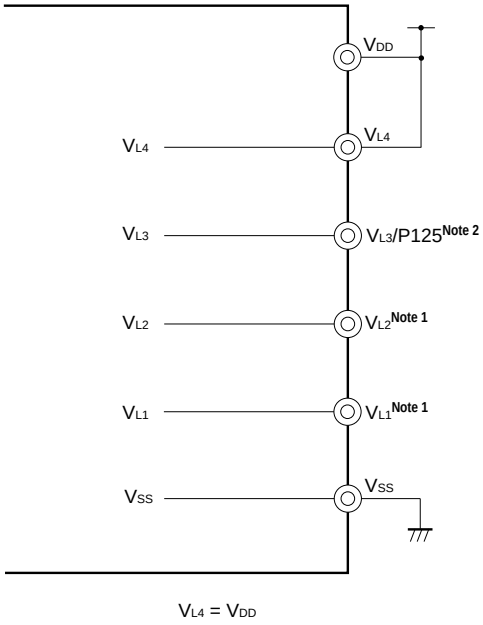
The external resistance division method, internal voltage boosting method, and capacitor split method can be selected as LCD drive power generating method.

### 21.8.1 External resistance division method

Figure 21-20 shows examples of LCD drive voltage connection, corresponding to each bias method.

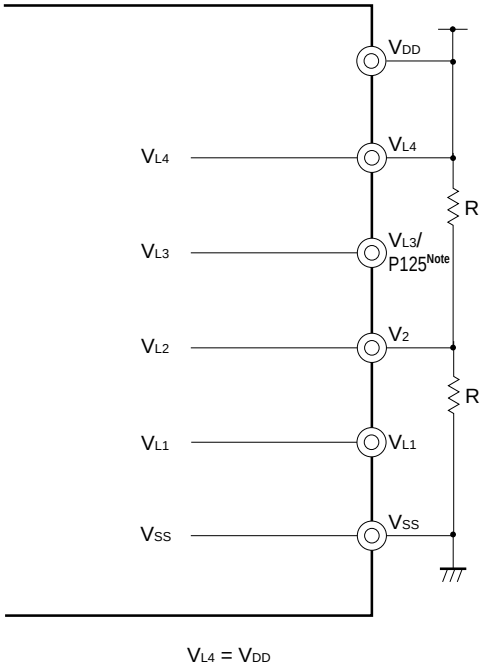
**Figure 21-20. Examples of LCD Drive Power Connections (External Resistance Division Method) (1/2)**

**(a) Static display mode**




$V_{L4} = V_{DD}$

**(b) 1/2 bias method**



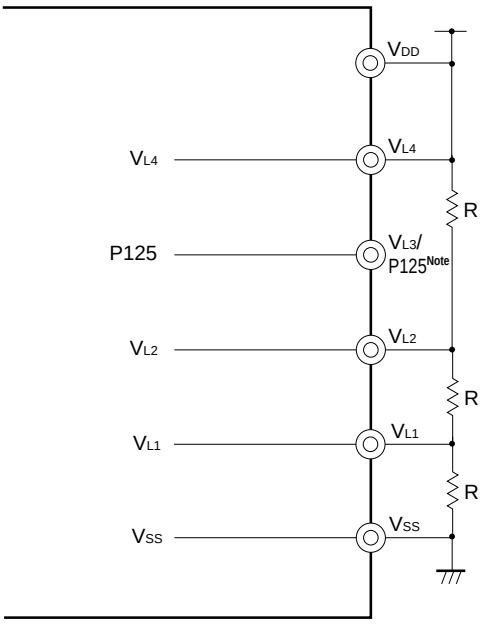

$V_{L4} = V_{DD}$

**Notes 1.** Connect $V_{L1}$ and $V_{L2}$ to GND or leave open.

**2.** $V_{L3}$ can be used as port (P125).

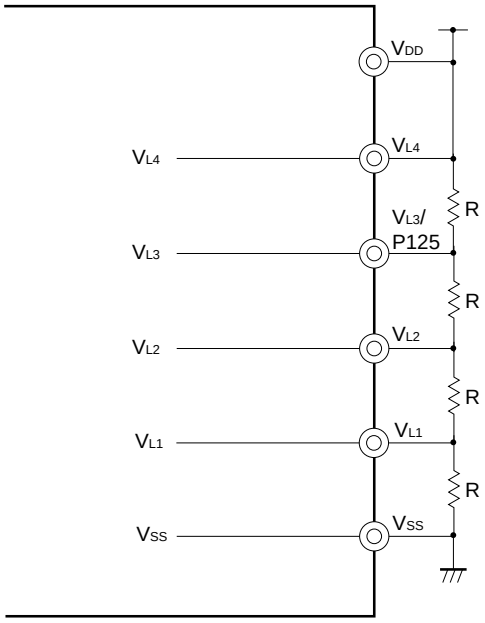
**Figure 21-20. Examples of LCD Drive Power Connections (External Resistance Division Method) (2/2)**

**(c) 1/3 bias method**




$V_{L4} = V_{DD}$

**(d) 1/4 bias method**




$V_{L4} = V_{DD}$

**Note** $V_{L3}$ can be used as port (P125).

**Caution The reference resistance "R" value for external resistance division is 10 kΩ to 1 MΩ. Also, to stabilize the potential of the $V_{L1}$ to $V_{L4}$ pins, connect a capacitor between each of pins $V_{L1}$ to $V_{L4}$ and the GND pin as needed. The reference capacitance is about 0.47 μF but it depends on the LCD panel used, the number of segment pins, the number of common pins, the frame frequency, and the operating environment. Thoroughly evaluate these values in accordance with your system and adjust and determine the capacitance.**

### 21.8.2 Internal voltage boosting method

RL78/I1B contains an internal voltage boost circuit for generating LCD drive power supplies. The internal voltage boost circuit and external capacitors (0.47 μF±30%) are used to generate an LCD drive voltage. Only 1/3 bias mode or 1/4 bias mode can be set for the internal voltage boosting method.

The LCD drive voltage of the internal voltage boosting method can supply a constant voltage, regardless of changes in $V_{DD}$, because it is a power supply separate from the main unit.

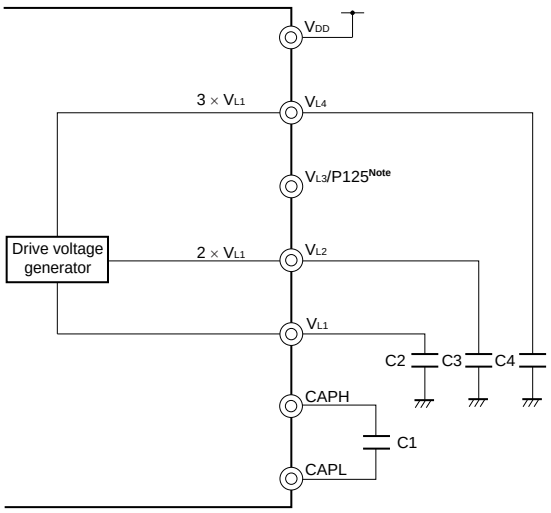
In addition, a contrast can be adjusted by using the LCD boost level control register (VLCD).

**Table 21-10. LCD Drive Voltages (Internal Voltage Boosting Method)**

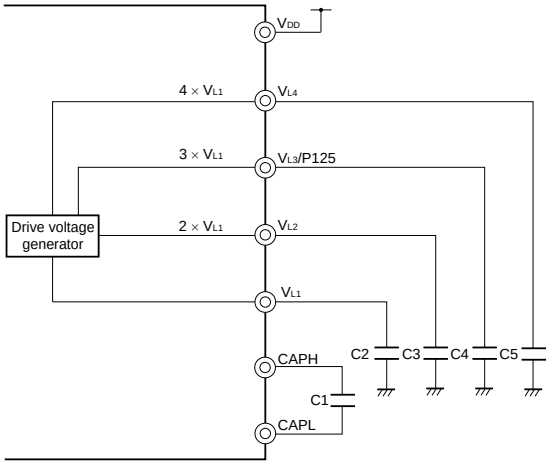
| Bias Method / LCD Drive Voltage Pin | 1/3 Bias Method | 1/4 Bias Method |
|---|---|---|
| $V_{L4}$ | $3 \times V_{L1}$ | $4 \times V_{L1}$ |
| $V_{L3}$ | – | $3 \times V_{L1}$ |
| $V_{L2}$ | $2 \times V_{L1}$ | $2 \times V_{L1}$ |
| $V_{L1}$ | LCD reference voltage | LCD reference voltage |

**Figure 21-21. Examples of LCD Drive Power Connections (Internal Voltage Boosting Method)**

**(a) 1/3 bias method**




**(b) 1/4 bias method**




**Note** $V_{L3}$ can be used as port (P125).

**Remark** Use a capacitor with as little leakage as possible.
In addition, make C1 a nonpolar capacitor.

### 21.8.3 Capacitor split method

RL78/I1B contains an internal voltage reduction circuit for generating LCD drive power supplies. The internal voltage reduction circuit and external capacitors (0.47 μF±30%) are used to generate an LCD drive voltage. Only 1/3 bias mode can be set for the capacitor split method.

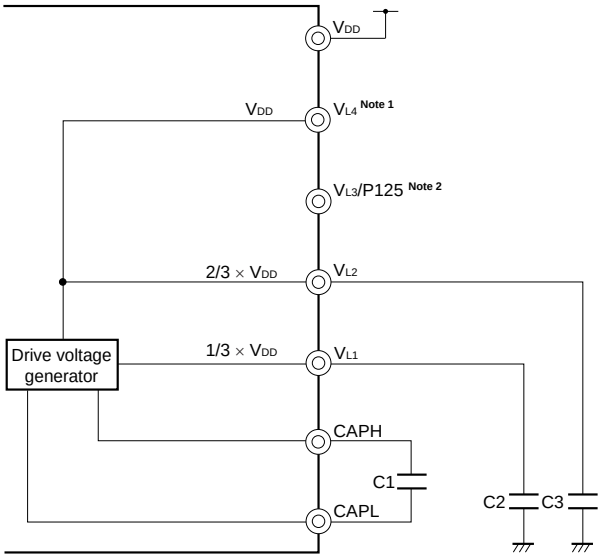
Different from the external resistance division method, there is always no current flowing with the capacitor split method, so current consumption can be reduced.

**Table 21-11. LCD Drive Voltages (Capacitor Split Method)**

| Bias Method / LCD Drive Voltage Pin | 1/3 Bias Method |
|---|---|
| $V_{L4}$ | $V_{DD}$ |
| $V_{L3}$ | – |
| $V_{L2}$ | $2/3 \times V_{L4}$ |
| $V_{L1}$ | $1/3 \times V_{L4}$ |

**Figure 21-22. Examples of LCD Drive Power Connections (Capacitor Split Method)**

- **1/3 bias method**




**Notes 1.** When switching to internal voltage boosting method, connect capacitor C4 as shown in Figure 21-21. Examples of LCD Drive Power Connections (Internal Voltage Boosting Method)

**2.** $V_{L3}$ can be used as port (P125).

**Remark** Use a capacitor with as little leakage as possible.
In addition, make C1 a nonpolar capacitor.

## 21.9 Common and Segment Signals

### 21.9.1 Normal liquid crystal waveform

Each pixel of the LCD panel turns on when the potential difference between the corresponding common and segment signals becomes higher than a specific voltage (LCD drive voltage, $V_{LCD}$). The pixels turn off when the potential difference becomes lower than $V_{LCD}$.

Applying DC voltage to the common and segment signals of an LCD panel causes deterioration. To avoid this problem, this LCD panel is driven by AC voltage.

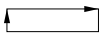
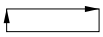
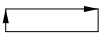
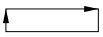
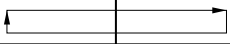
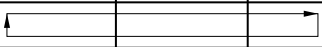
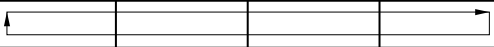
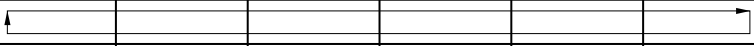
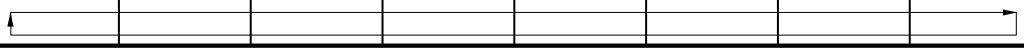
**(1) Common signals**

Each common signal is selected sequentially according to a specified number of time slices at the timing listed in Table 21-12. In the static display mode, the same signal is output to COM0 to COM3.

In the two-time-slice mode, leave the COM2 and COM3 pins open. In the three-time-slice mode, leave the COM3 pin open.

Use the COM4 to COM7 pins other than in the six-time-slice mode and eight-time-slice mode, and COM6, COM7 pins in the six-time-slice mode as open or segment pins.

**Table 21-12. COM Signals**

| COM Signal / Number of Time Slices | COM0 | COM1 | COM2 | COM3 | COM4 | COM5 | COM6 | COM7 |
|---|---|---|---|---|---|---|---|---|
| Static display mode |  |  |  |  | **Note** | **Note** | **Note** | **Note** |
| Two-time-slice mode |  | | Open | Open | **Note** | **Note** | **Note** | **Note** |
| Three-time-slice mode |  | | | Open | **Note** | **Note** | **Note** | **Note** |
| Four-time-slice mode |  | | | | **Note** | **Note** | **Note** | **Note** |
| Six-time-slice mode |  | | | | | | **Note** | **Note** |
| Eight-time-slice mode |  | | | | | | | |

**Note** Use the pins as open or segment pins.

**(2) Segment signals**

The segment signals correspond to the LCD display data register (see **21.4 LCD Display Data Registers**).

When the number of time slices is eight, bits 0 to 7 of each display data register are read in synchronization with COM0 to COM7, respectively. If a bit is 1, it is converted to the select voltage, and if it is 0, it is converted to the deselect voltage. The conversion results are output to the segment pins (SEG4 to SEG41).

When the number of time slices is number other than eight, bits 0 to 3 of each byte in A-pattern area are read in synchronization with COM0 to COM3, and bits 4 to 7 of each byte in B-pattern area are read in synchronization with COM0 to COM3, respectively. If a bit is 1, it is converted to the select voltage, and if it is 0, it is converted to the deselect voltage. The conversion results are output to the segment pins (SEG0 to SEG41).

Check, with the information given above, what combination of front-surface electrodes (corresponding to the segment signals) and rear-surface electrodes (corresponding to the common signals) forms display patterns in the LCD display data register, and write the bit data that corresponds to the desired display pattern on a one-to-one basis.

**Remark** The mounted segment output pins vary depending on the product.

- 80-pin products: SEG0 to SEG27, SEG32 to SEG37
- 100-pin products: SEG0 to SEG41

### (3) Output waveforms of common and segment signals

The voltages listed in Table 21-13 are output as common and segment signals.

When both common and segment signals are at the select voltage, a display on-voltage of $\pm V_{LCD}$ is obtained. The other combinations of the signals correspond to the display off-voltage.

**Table 21-13. LCD Drive Voltage**

**(a) Static display mode**

| Common Signal \ Segment Signal | Select Signal Level | Deselect Signal Level |
|---|---|---|
| | $V_{SS}/V_{L4}$ | $V_{L4}/V_{SS}$ |
| $V_{L4}/V_{SS}$ | $-V_{LCD}/+V_{LCD}$ | 0 V/0 V |

**(b) 1/2 bias method**

| Common Signal \ Segment Signal | | Select Signal Level | Deselect Signal Level |
|---|---|---|---|
| | | $V_{SS}/V_{L4}$ | $V_{L4}/V_{SS}$ |
| Select signal level | $V_{L4}/V_{SS}$ | $-V_{LCD}/+V_{LCD}$ | 0 V/0 V |
| Deselect signal level | $V_{L2}$ | $-\frac{1}{2}V_{LCD}/+\frac{1}{2}V_{LCD}$ | $+\frac{1}{2}V_{LCD}/-\frac{1}{2}V_{LCD}$ |

**(c) 1/3 bias method (waveform A or B)**

| Common Signal \ Segment Signal | | Select Signal Level | Deselect Signal Level |
|---|---|---|---|
| | | $V_{SS}/V_{L4}$ | $V_{L2}/V_{L1}$ |
| Select signal level | $V_{L4}/V_{SS}$ | $-V_{LCD}/+V_{LCD}$ | $-\frac{1}{3}V_{LCD}/+\frac{1}{3}V_{LCD}$ |
| Deselect signal level | $V_{L1}/V_{L2}$ | $-\frac{1}{3}V_{LCD}/+\frac{1}{3}V_{LCD}$ | $+\frac{1}{3}V_{LCD}/-\frac{1}{3}V_{LCD}$ |

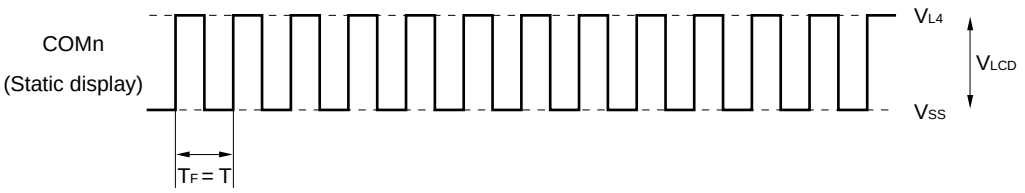
**(d) 1/4 bias method (waveform A or B)**

| Common Signal \ Segment Signal | | Select Signal Level | Deselect Signal Level |
|---|---|---|---|
| | | $V_{SS}/V_{L4}$ | $V_{L2}$ |
| Select signal level | $V_{L4}/V_{SS}$ | $-V_{LCD}/+V_{LCD}$ | $-\frac{1}{2}V_{LCD}/+\frac{1}{2}V_{LCD}$ |
| Deselect signal level | $V_{L1}/V_{L3}$ | $-\frac{1}{4}V_{LCD}/+\frac{1}{4}V_{LCD}$ | $+\frac{1}{4}V_{LCD}/-\frac{1}{4}V_{LCD}$ |

Figure 21-23 shows the common signal waveforms, and Figure 21-24 shows the voltages and phases of the common and segment signals.

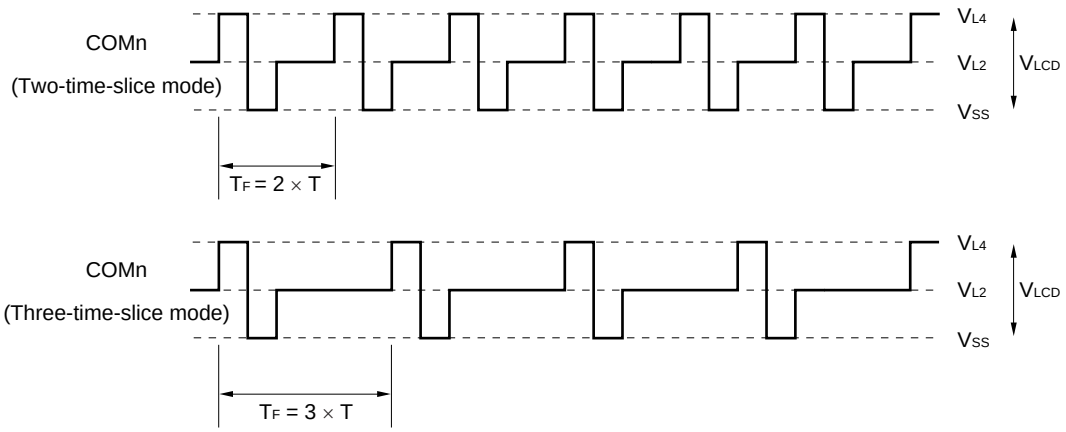
**Figure 21-23. Common Signal Waveforms (1/2)**

**(a) Static display mode**




T: One LCD clock period $T_F$: Frame frequency

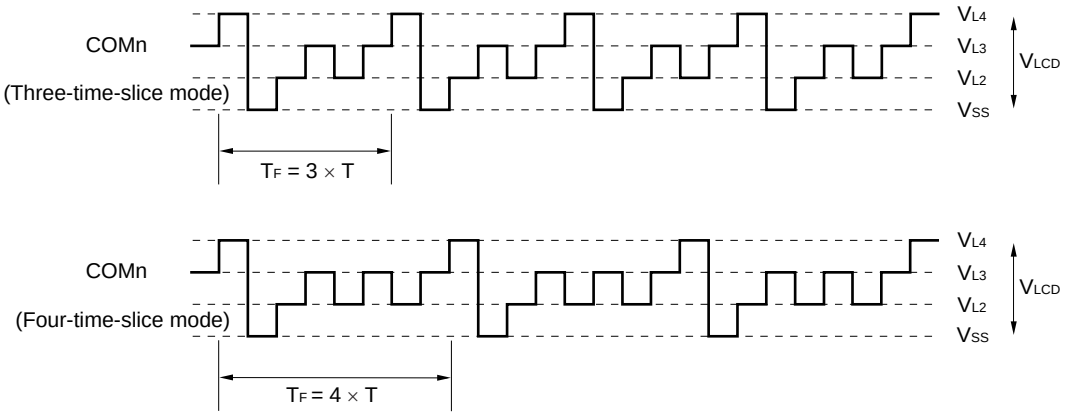
**(b) 1/2 bias method**




T: One LCD clock period $T_F$: Frame frequency

**Figure 21-23. Common Signal Waveforms (2/2)**

**(c) 1/3 bias method**



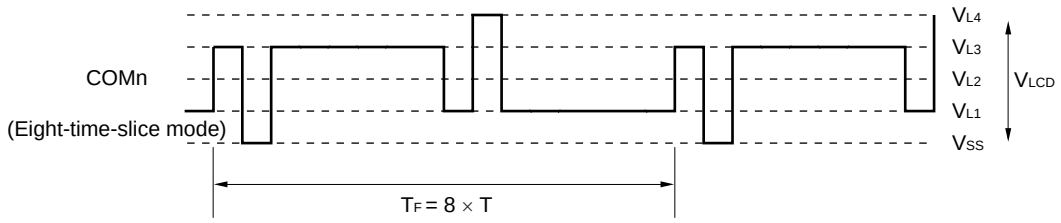

T: One LCD clock period $T_F$: Frame frequency

< Example of calculation of LCD frame frequency (When four-time-slice mode is used) >

LCD clock: $32768/2^7$ = 256 Hz (When setting to LCDC0 = 06H)

LCD frame frequency: 64 Hz

**(d) 1/4 bias method**




T: One LCD clock period $T_F$: Frame frequency

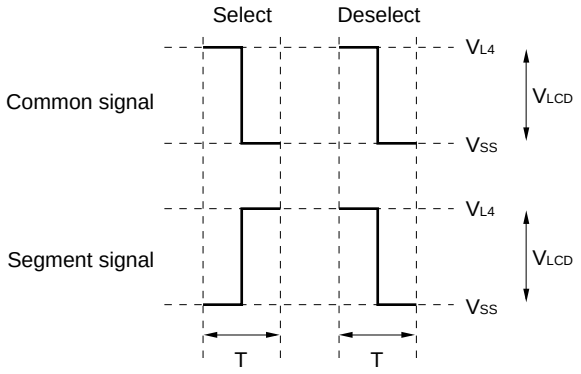
< Example of calculation of LCD frame frequency (When eight-time-slice mode is used) >

LCD clock: $32768/2^7$ = 256 Hz (When setting to LCDC0 = 06H)

LCD frame frequency: 32 Hz

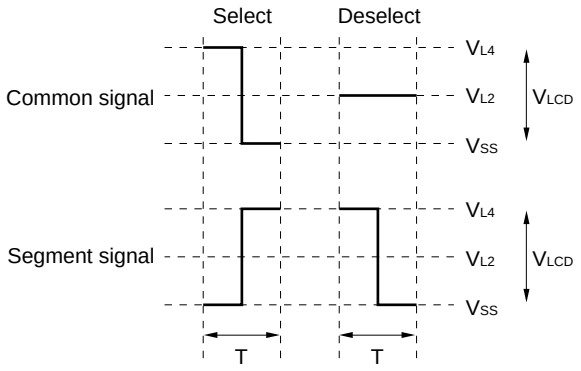
**Figure 21-24. Voltages and Phases of Common and Segment Signals (1/3)**

**(a) Static display mode (waveform A)**




T: One LCD clock period

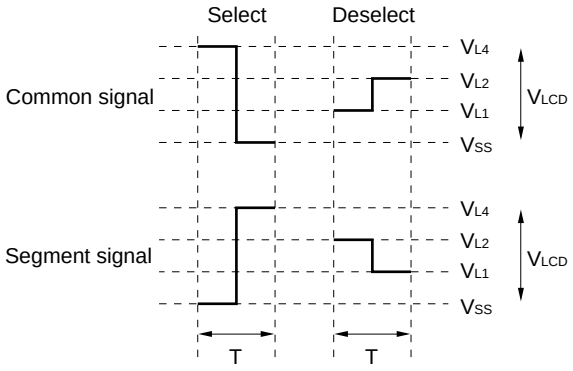
**(b) 1/2 bias method (waveform A)**




T: One LCD clock period

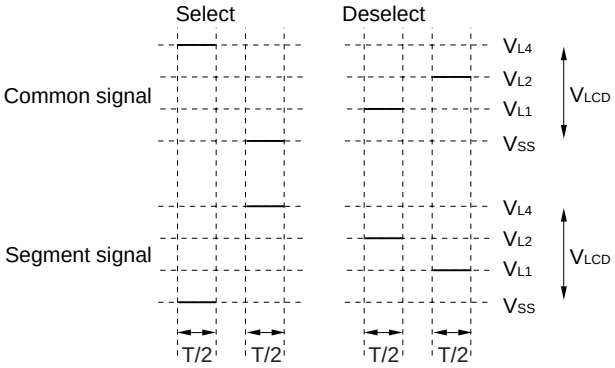
**Figure 21-24. Voltages and Phases of Common and Segment Signals (2/3)**

**(c) 1/3 bias method (waveform A)**




T: One LCD clock period

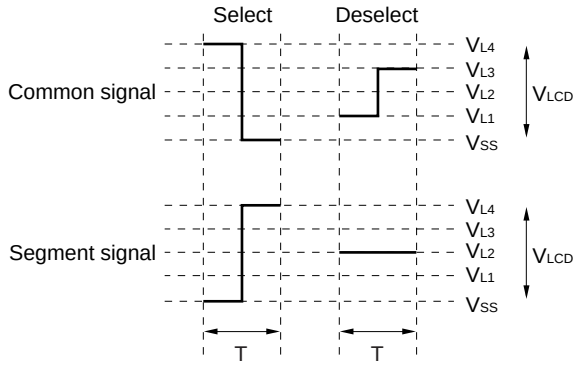
**(d) 1/3 bias method (waveform B)**




T: One LCD clock period

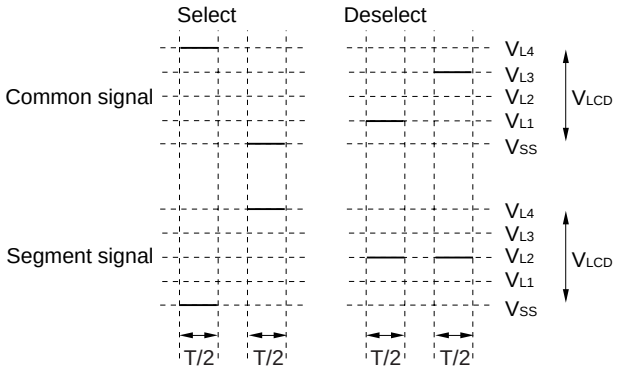
**Figure 21-24. Voltages and Phases of Common and Segment Signals (3/3)**

**(e) 1/4 bias method (waveform A)**




T: One LCD clock period

**(f) 1/4 bias method (waveform B)**




T: One LCD clock period

## 21.10 Display Modes

### 21.10.1 Static display example

Figure 21-26 shows how the three-digit LCD panel having the display pattern shown in Figure 21-25 is connected to the segment signals (SEG0 to SEG23) and the common signal (COM0). This example displays data “12.3” in the LCD panel. The contents of the display data register (F0400H to F0417H) correspond to this display.

The following description focuses on numeral “2.” (2.) displayed in the second digit. To display “2.” in the LCD panel, it is necessary to apply the select or deselect voltage to the SEG8 to SEG15 pins according to Table 21-14 at the timing of the common signal COM0; see Figure 21-25 for the relationship between the segment signals and LCD segments.

**Table 21-14. Select and Deselect Voltages (COM0)**

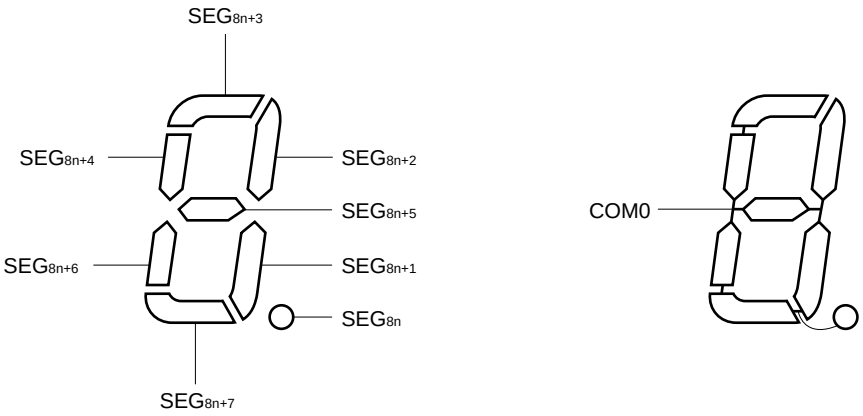
| Segment / Common | SEG8 | SEG9 | SEG10 | SEG11 | SEG12 | SEG13 | SEG14 | SEG15 |
|---|---|---|---|---|---|---|---|---|
| COM0 | Select | Deselect | Select | Select | Deselect | Select | Select | Select |

According to Table 21-14, it is determined that the bit-0 pattern of the display data register locations (F0408H to F040FH) must be 10110111.

Figure 21-27 shows the LCD drive waveforms of SEG11 and SEG12, and COM0. When the select voltage is applied to SEG11 at the timing of COM0, an alternate rectangle waveform, $+V_{LCD}/-V_{LCD}$, is generated to turn on the corresponding LCD segment.

COM1 to COM3 are supplied with the same waveform as for COM0. So, COM0 to COM3 may be connected together to increase the driving capacity.

**Figure 21-25. Static LCD Display Pattern and Electrode Connections**




**Remark** 100-pin products: n = 0 to 4

**Figure 21-26. Example of Connecting Static LCD Panel**

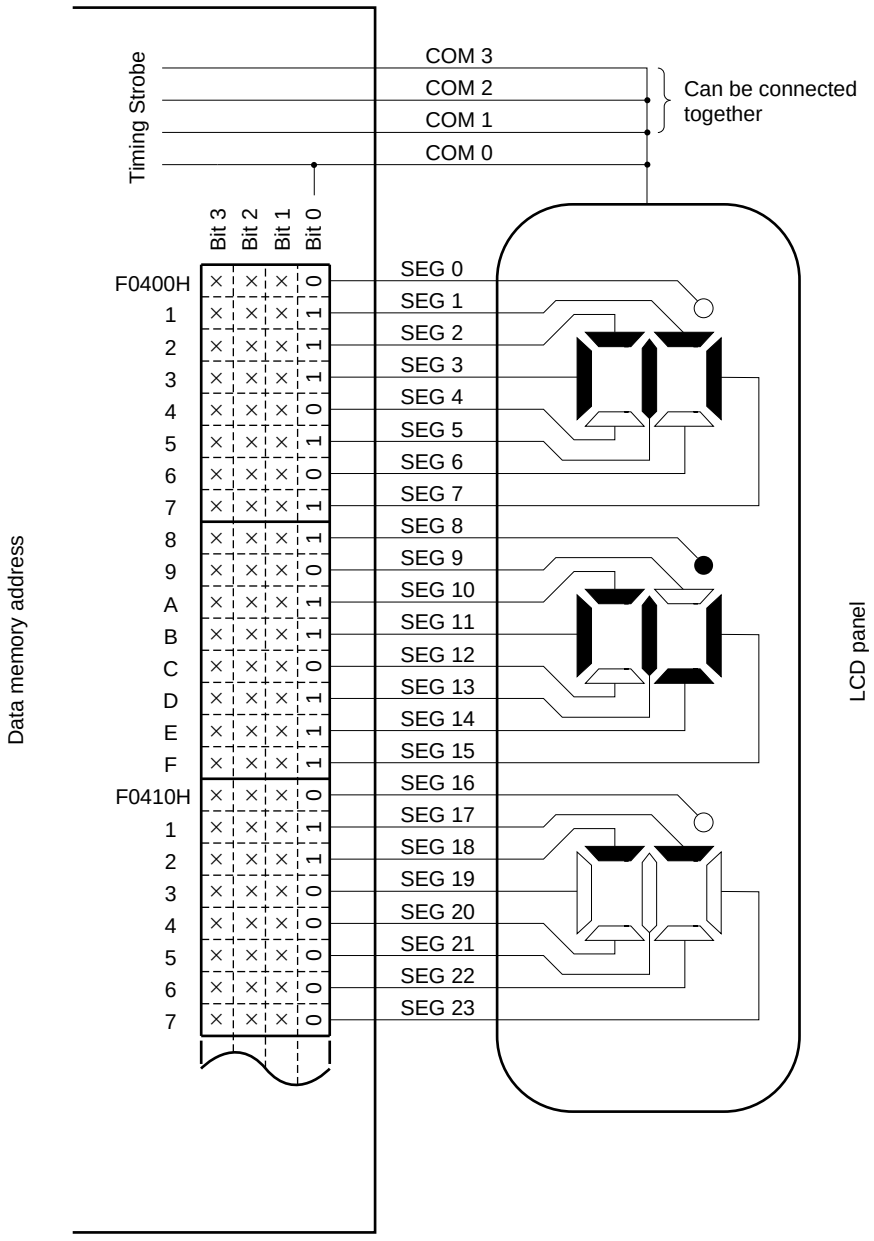
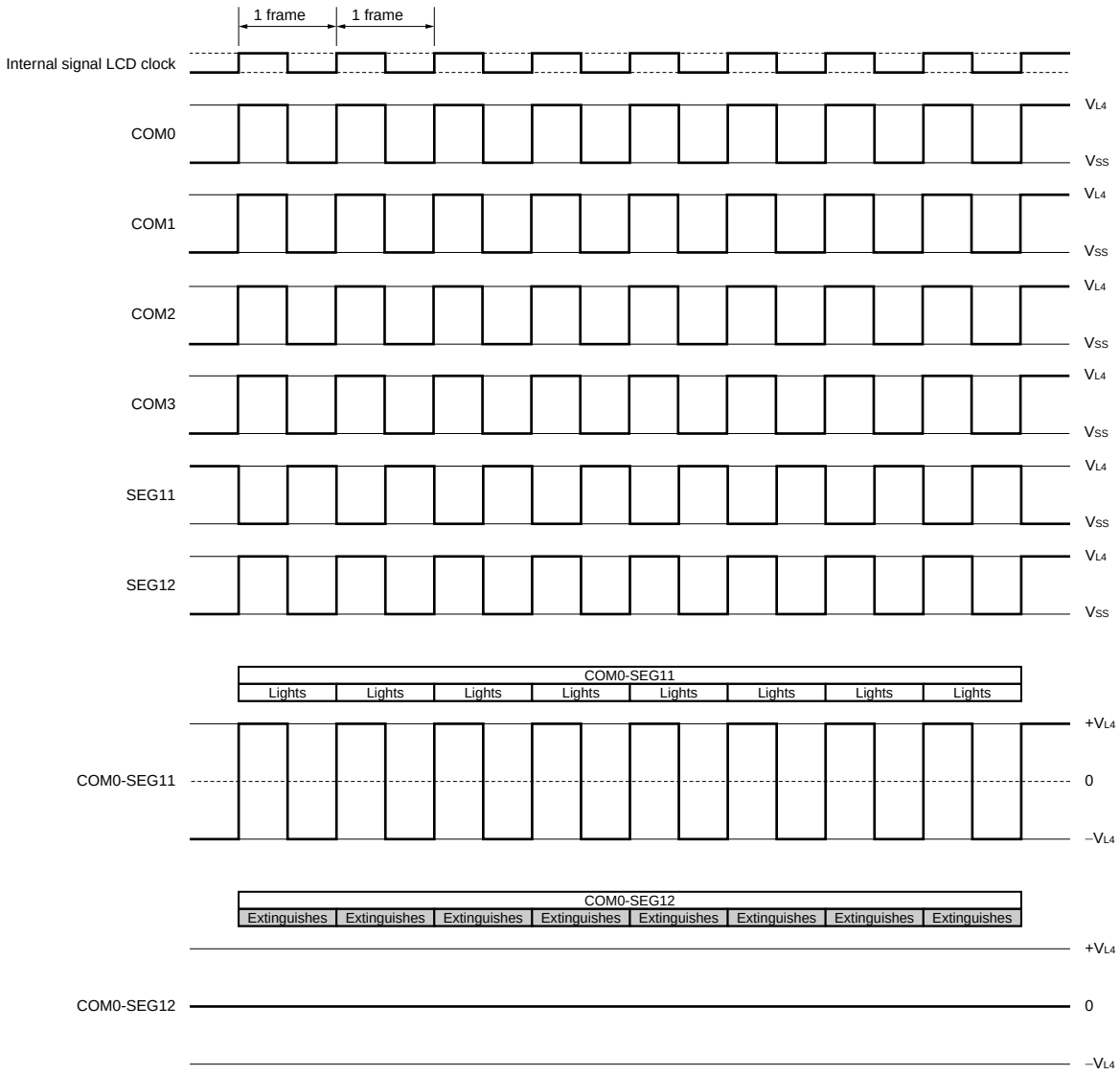


**Figure 21-27. Static LCD Drive Waveform Examples for SEG11, SEG12, and COM0**



### 21.10.2 Two-time-slice display example

Figure 21-29 shows how the 6-digit LCD panel having the display pattern shown in Figure 21-28 is connected to the segment signals (SEG0 to SEG23) and the common signals (COM0 and COM1). This example displays data “12345.6” in the LCD panel. The contents of the display data register (F0400H to F0417H) correspond to this display.

The following description focuses on numeral “3” ( 3 ) displayed in the fourth digit. To display “3” in the LCD panel, it is necessary to apply the select or deselect voltage to the SEG12 to SEG15 pins according to Table 21-15 at the timing of the common signals COM0 and COM1; see Figure 21-28 for the relationship between the segment signals and LCD segments.

**Table 21-15. Select and Deselect Voltages (COM0 and COM1)**

| Segment / Common | SEG12 | SEG13 | SEG14 | SEG15 |
|---|---|---|---|---|
| COM0 | Select | Select | Deselect | Deselect |
| COM1 | Deselect | Select | Select | Select |

According to Table 21-15, it is determined that the display data register location (F040FH) that corresponds to SEG15 must contain xx10.

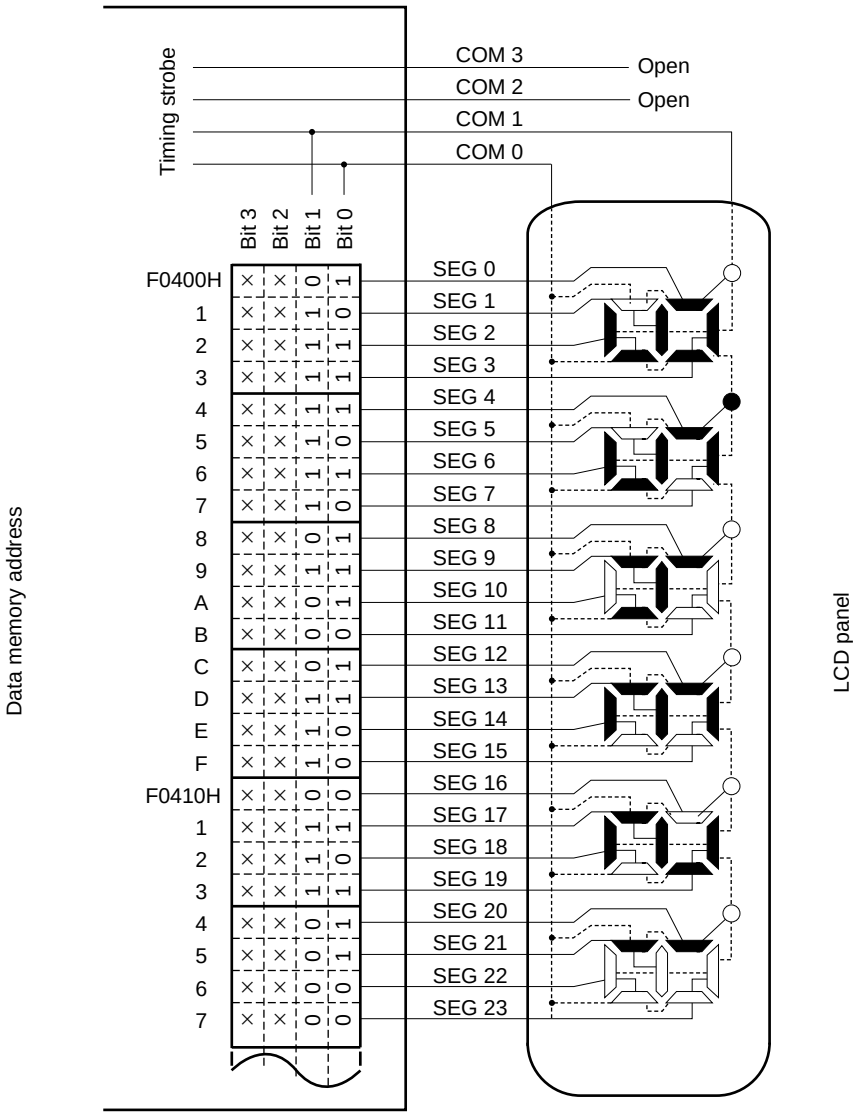
Figure 21-30 shows examples of LCD drive waveforms between the SEG15 signal and each common signal. When the select voltage is applied to SEG15 at the timing of COM1, an alternate rectangle waveform, $+V_{LCD}/-V_{LCD}$, is generated to turn on the corresponding LCD segment.

**Figure 21-28. Two-Time-Slice LCD Display Pattern and Electrode Connections**



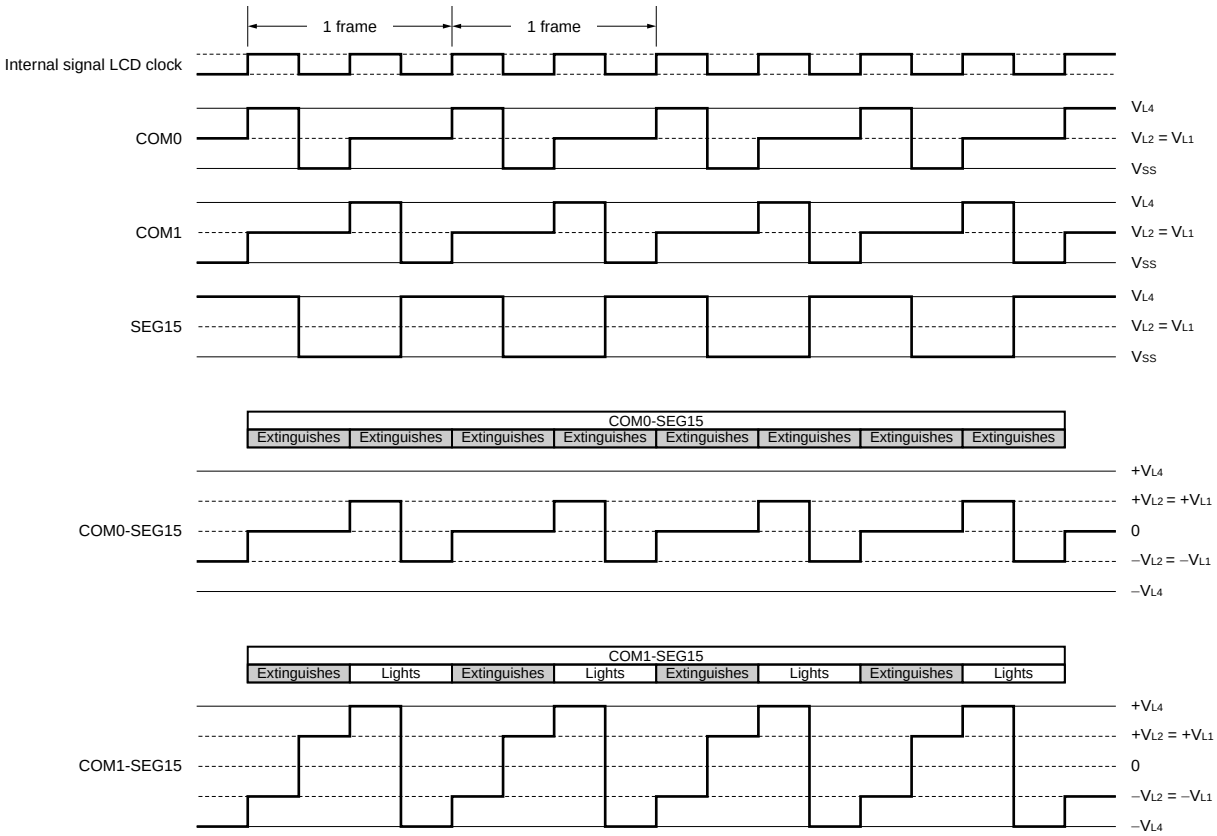

**Remark** 100-pin products: n = 0 to 9

**Figure 21-29. Example of Connecting Two-Time-Slice LCD Panel**




×: Can always be used to store any data because the two-time-slice mode is being used.

**Figure 21-30. Two-Time-Slice LCD Drive Waveform Examples Between SEG15 and Each Common Signals (1/2 Bias Method)**



### 21.10.3 Three-time-slice display example

Figure 21-32 shows how the 8-digit LCD panel having the display pattern shown in Figure 21-31 is connected to the segment signals (SEG0 to SEG23) and the common signals (COM0 to COM2). This example displays data “123456.78” in the LCD panel. The contents of the display data register (addresses F0400H to F0417H) correspond to this display.

The following description focuses on numeral “6.” ( 6. ) displayed in the third digit. To display “6.” in the LCD panel, it is necessary to apply the select or deselect voltage to the SEG6 to SEG8 pins according to Table 21-16 at the timing of the common signals COM0 to COM2; see Figure 21-31 for the relationship between the segment signals and LCD segments.

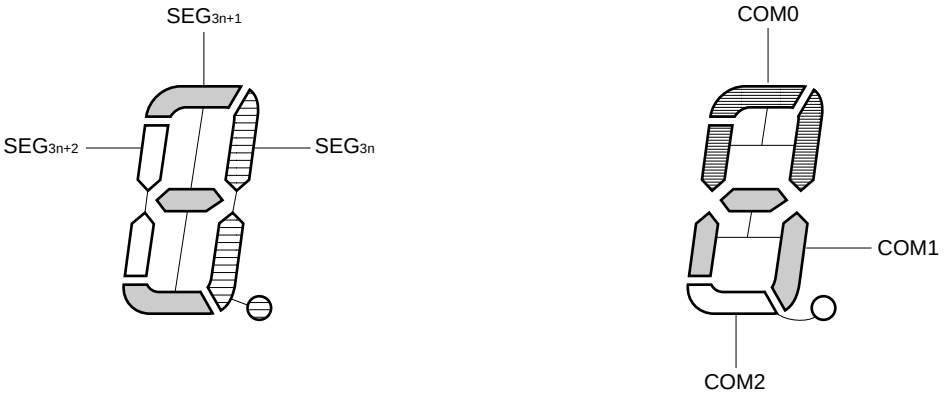
**Table 21-16. Select and Deselect Voltages (COM0 to COM2)**

| Segment / Common | SEG6 | SEG7 | SEG8 |
|---|---|---|---|
| COM0 | Deselect | Select | Select |
| COM1 | Select | Select | Select |
| COM2 | Select | Select | – |

According to Table 21-16, it is determined that the display data register location (F0406H) that corresponds to SEG6 must contain x110.

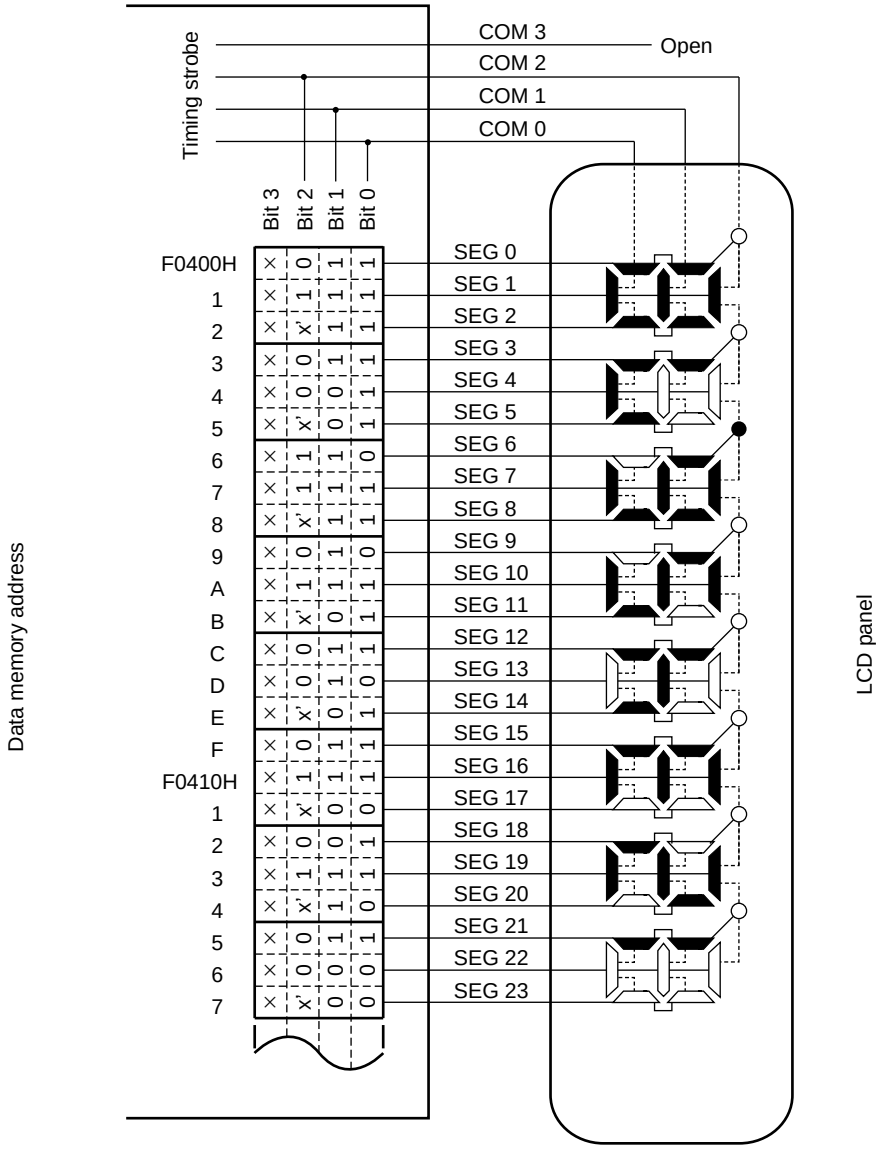
Figures 21-33 and 21-34 show examples of LCD drive waveforms between the SEG6 signal and each common signal in the 1/2 and 1/3 bias methods, respectively. When the select voltage is applied to SEG6 at the timing of COM1 or COM2, an alternate rectangle waveform, $+V_{LCD}/-V_{LCD}$, is generated to turn on the corresponding LCD segment.

**Figure 21-31. Three-Time-Slice LCD Display Pattern and Electrode Connections**




**Remark** 100-pin products: n = 0 to 13

**Figure 21-32. Example of Connecting Three-Time-Slice LCD Panel**




×': Can be used to store any data because there is no corresponding segment in the LCD panel.

×: Can always be used to store any data because the three-time-slice mode is being used.

**Figure 21-33. Three-Time-Slice LCD Drive Waveform Examples Between SEG6 and Each Common Signals (1/2 Bias Method)**

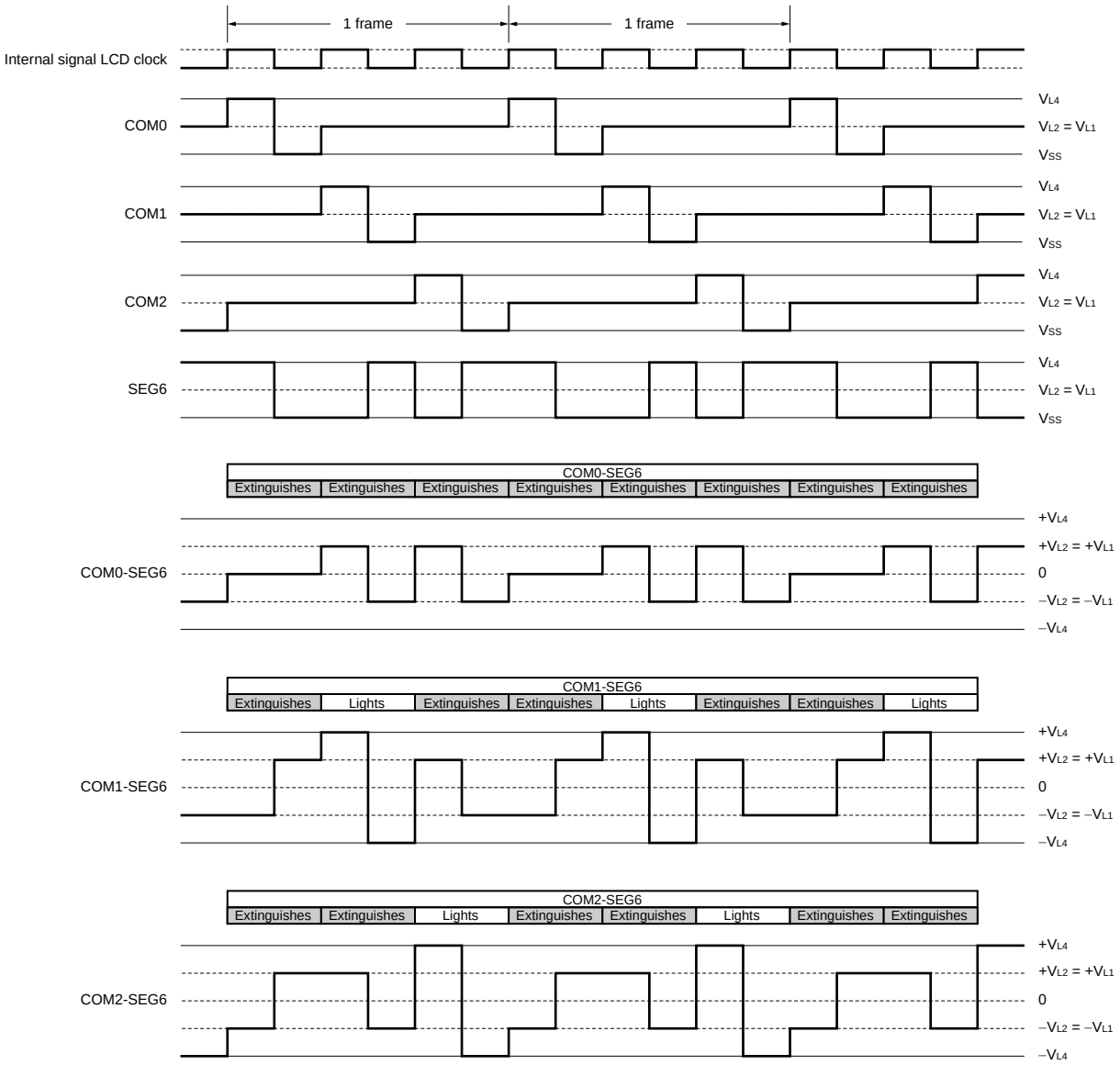
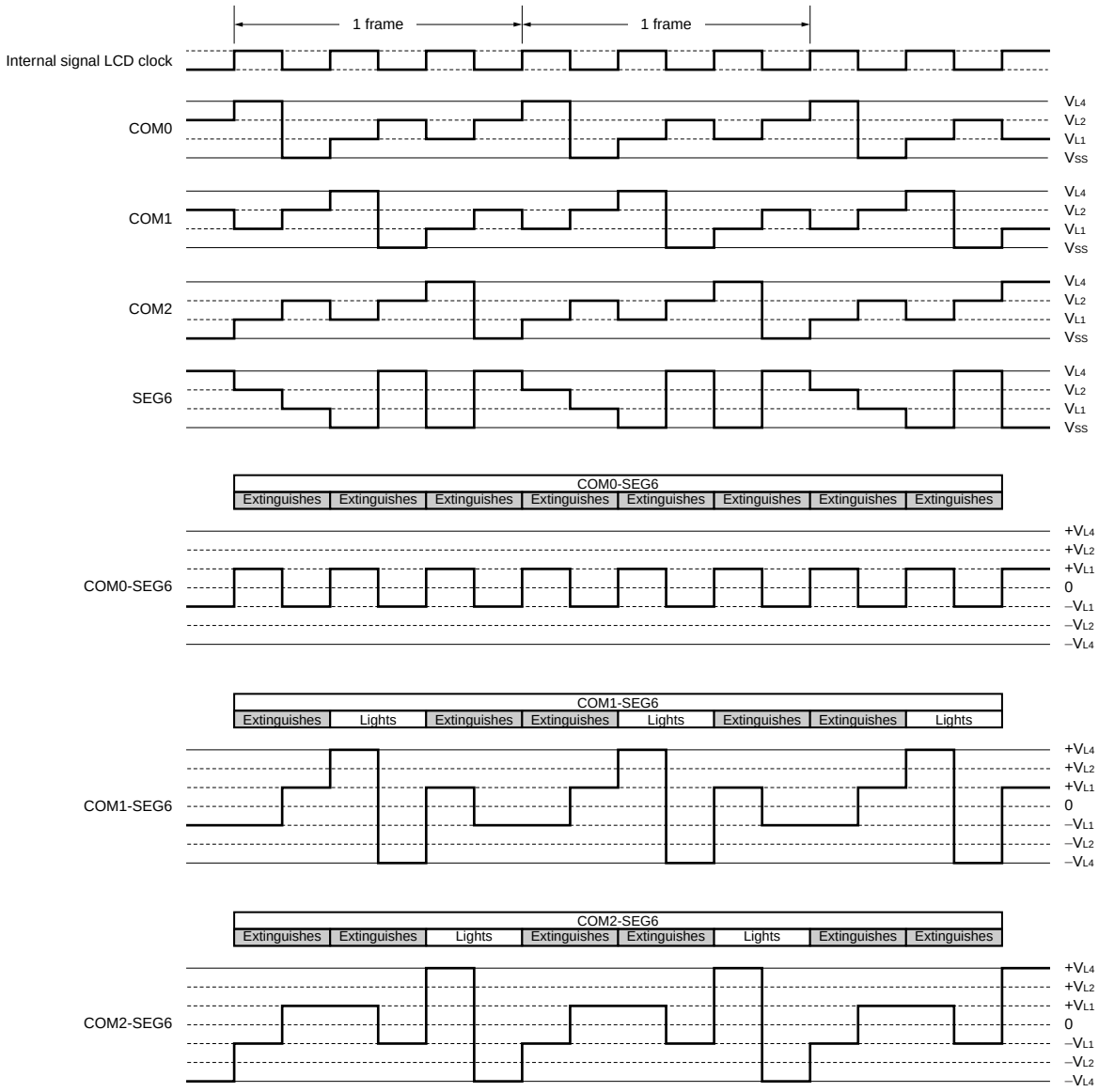


**Figure 21-34. Three-Time-Slice LCD Drive Waveform Examples Between SEG6 and Each Common Signals (1/3 Bias Method)**



### 21.10.4 Four-time-slice display example

Figure 21-36 shows how the 12-digit LCD panel having the display pattern shown in Figure 21-35 is connected to the segment signals (SEG0 to SEG23) and the common signals (COM0 to COM3). This example displays data “123456.789012” in the LCD panel. The contents of the display data register (addresses F0400H to F0417H) correspond to this display.

The following description focuses on numeral “6.” (6.) displayed in the seventh digit. To display “6.” in the LCD panel, it is necessary to apply the select or deselect voltage to the SEG12 and SEG13 pins according to Table 21-17 at the timing of the common signals COM0 to COM3; see Figure 21-35 for the relationship between the segment signals and LCD segments.

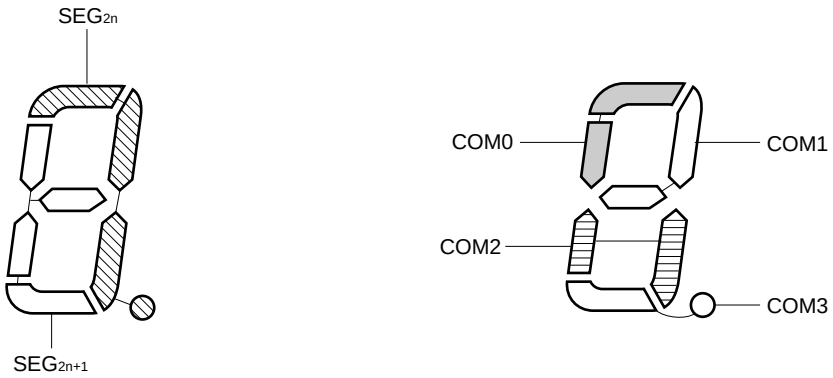
**Table 21-17. Select and Deselect Voltages (COM0 to COM3)**

| Segment / Common | SEG12 | SEG13 |
|---|---|---|
| COM0 | Select | Select |
| COM1 | Deselect | Select |
| COM2 | Select | Select |
| COM3 | Select | Select |

According to Table 21-17, it is determined that the display data register location (F040CH) that corresponds to SEG12 must contain 1101.

Figure 21-37 shows examples of LCD drive waveforms between the SEG12 signal and each common signal. When the select voltage is applied to SEG12 at the timing of COM0, an alternate rectangle waveform, $+V_{LCD}/-V_{LCD}$, is generated to turn on the corresponding LCD segment.

**Figure 21-35. Four-Time-Slice LCD Display Pattern and Electrode Connections**




**Remark** 100-pin products: n = 0 to 20

**Figure 21-36. Example of Connecting Four-Time-Slice LCD Panel**

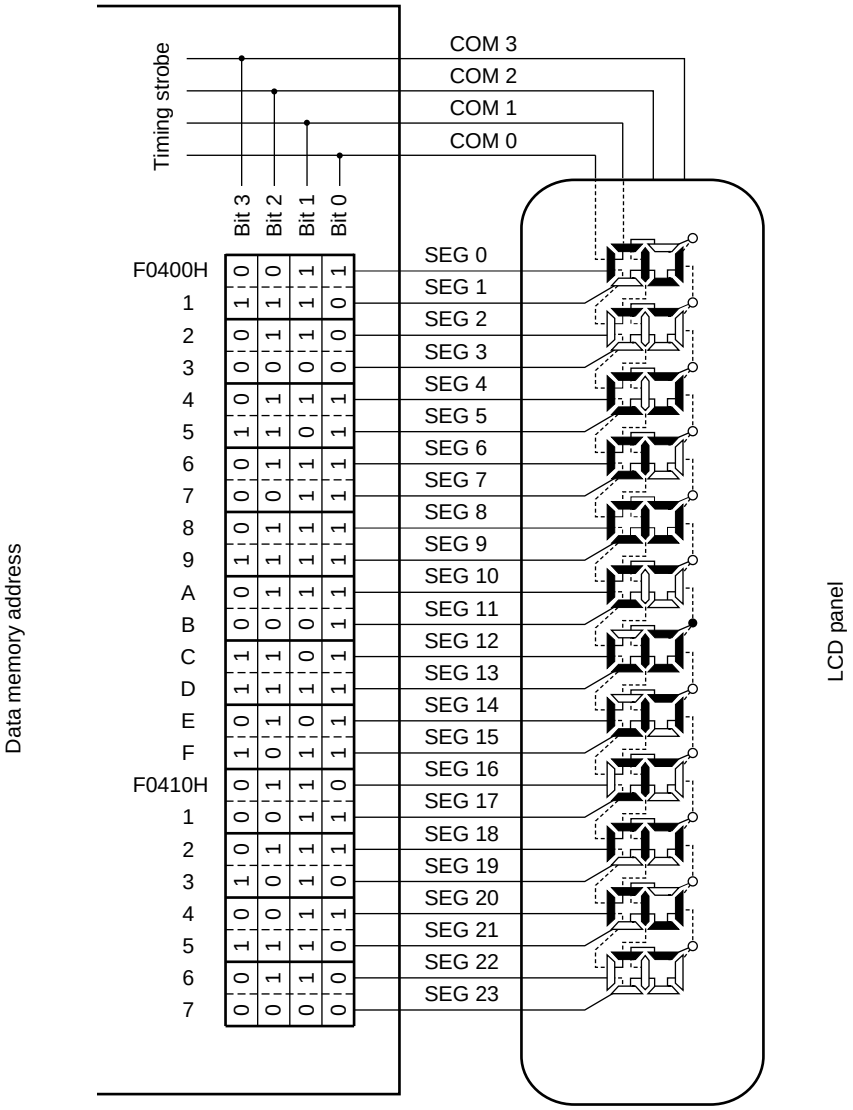


**Figure 21-37. Four-Time-Slice LCD Drive Waveform Examples Between SEG12 and Each Common Signals (1/3 Bias Method) (1/2)**

**(a) Waveform A**

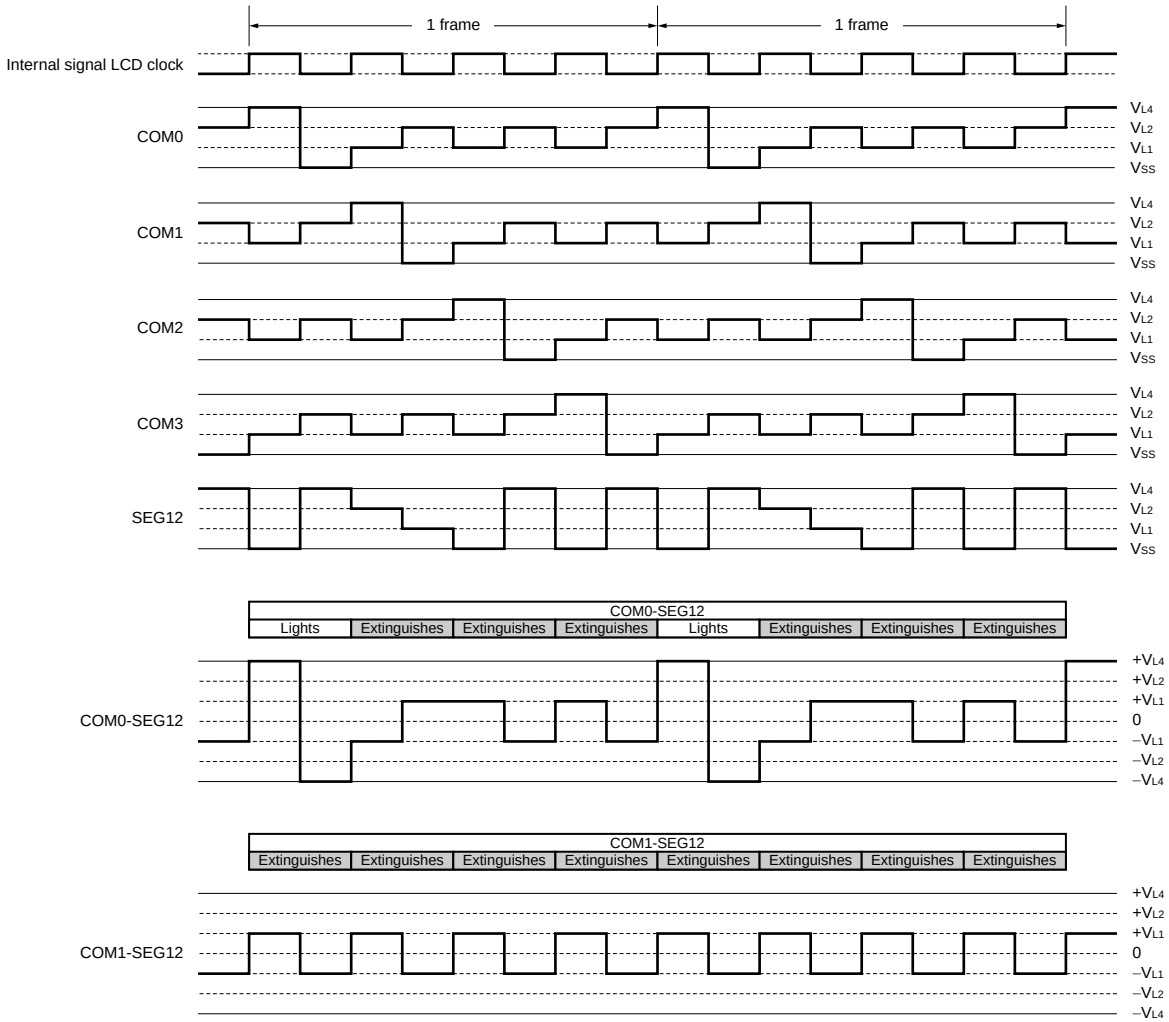
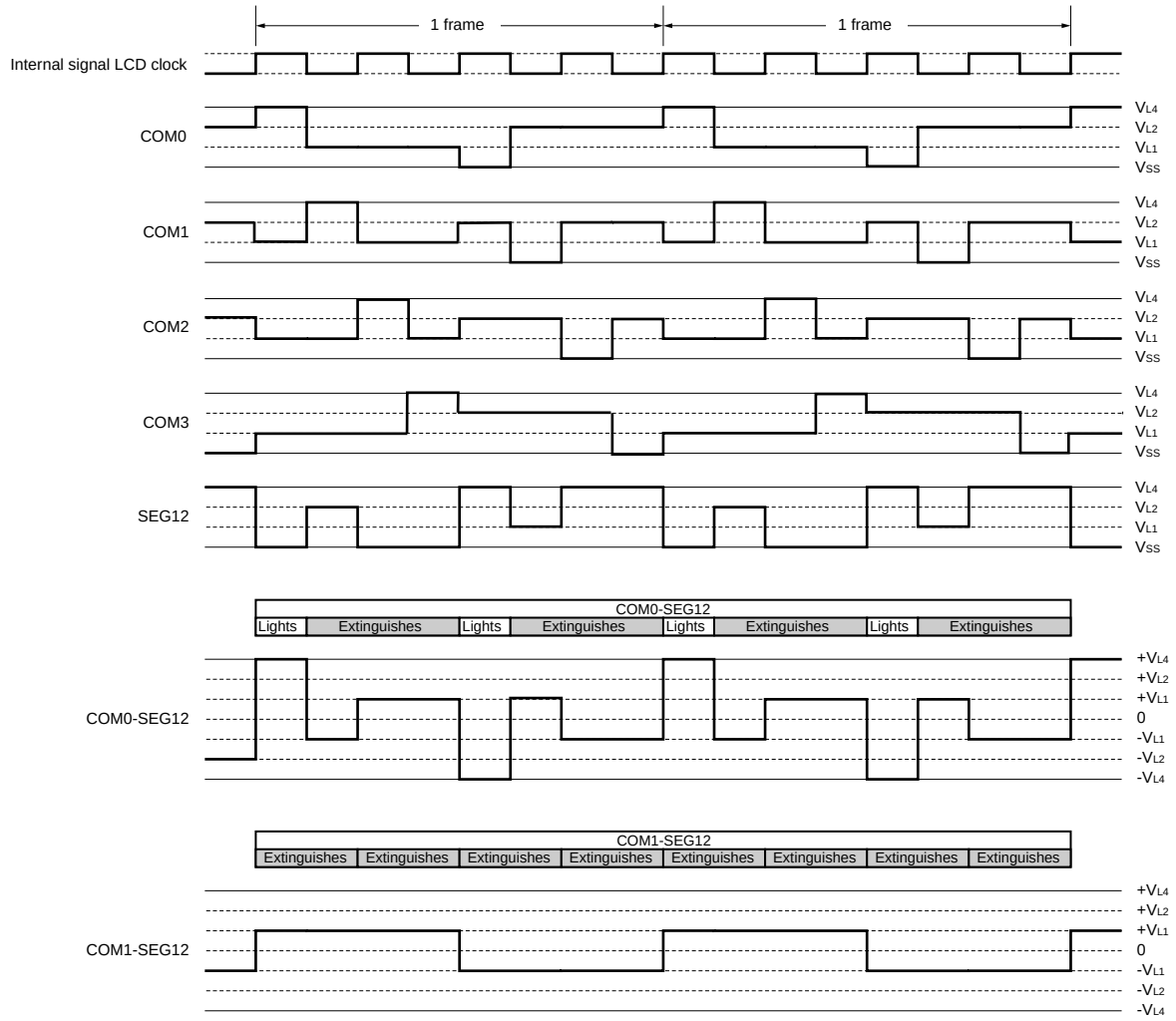


**Figure 21-37. Four-Time-Slice LCD Drive Waveform Examples Between SEG12 and Each Common Signals (1/3 Bias Method) (2/2)**

**(b) Waveform B**



### 21.10.5 Six-time-slice display example

Figure 21-39 shows how the 15x6 dot LCD panel having the display pattern shown in Figure 21-38 is connected to the segment signals (SEG2 to SEG16) and the common signals (COM0 to COM5). This example displays data “123” in the LCD panel. The contents of the display data register (addresses F0402H to F0410H) correspond to this display.

The following description focuses on numeral “3.” ( ∃ ) displayed in the first digit. To display “3.” in the LCD panel, it is necessary to apply the select or deselect voltage to the SEG2 to SEG6 pins according to Table 21-18 at the timing of the common signals COM0 to COM5; see Figure 21-38 for the relationship between the segment signals and LCD segments.

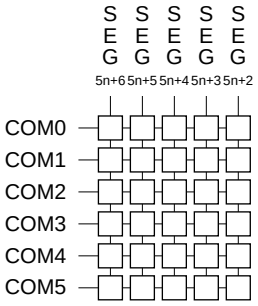
**Table 21-18. Select and Deselect Voltages (COM0 to COM5)**

| Segment / Common | SEG2 | SEG3 | SEG4 | SEG5 | SEG6 |
|---|---|---|---|---|---|
| COM0 | Select | Select | Select | Select | Select |
| COM1 | Deselect | Select | Deselect | Deselect | Deselect |
| COM2 | Deselect | Deselect | Select | Deselect | Deselect |
| COM3 | Deselect | Select | Deselect | Deselect | Deselect |
| COM4 | Select | Deselect | Deselect | Deselect | Select |
| COM5 | Deselect | Select | Select | Select | Deselect |

According to Table 21-18, it is determined that the display data register location (F0402H) that corresponds to SEG2 must contain 010001.

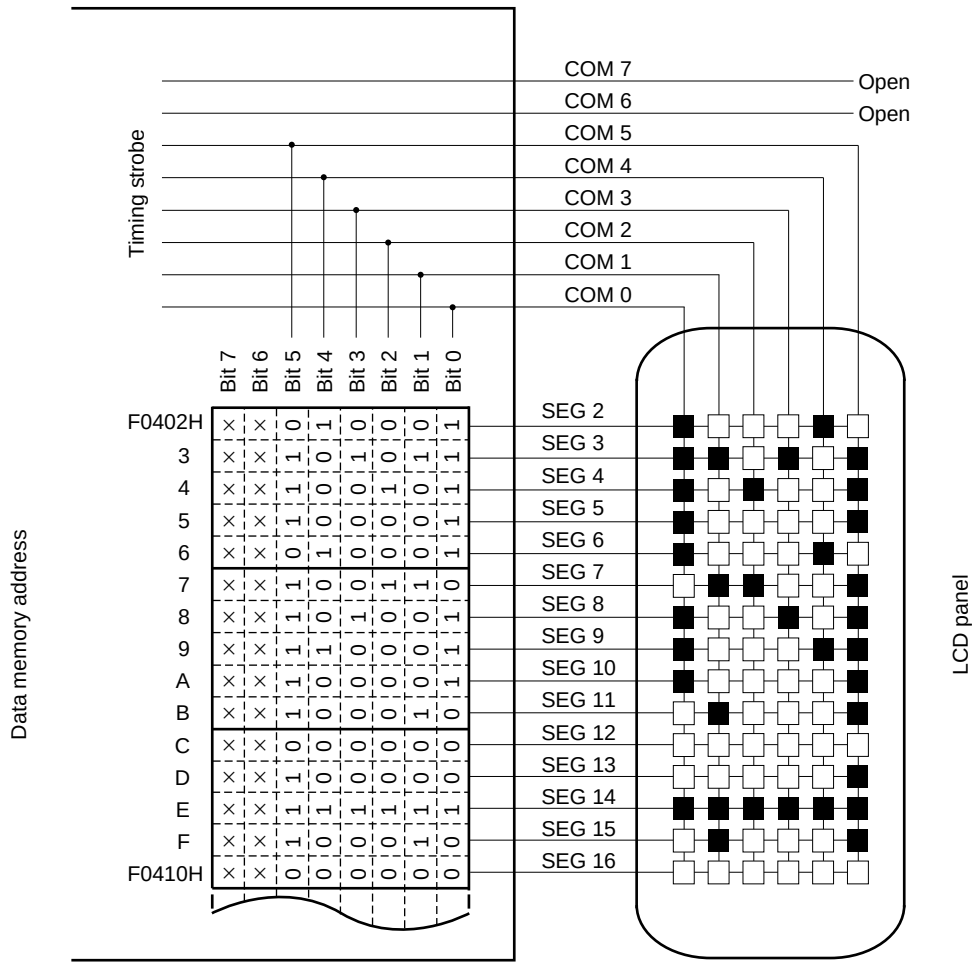
Figure 21-40 shows examples of LCD drive waveforms between the SEG2 signal and each common signal. When the select voltage is applied to SEG2 at the timing of COM0, a waveform is generated to turn on the corresponding LCD segment.

**Figure 21-38. Six-Time-Slice LCD Display Pattern and Electrode Connections**




**Remark** 100-pin products: n = 0 to 7

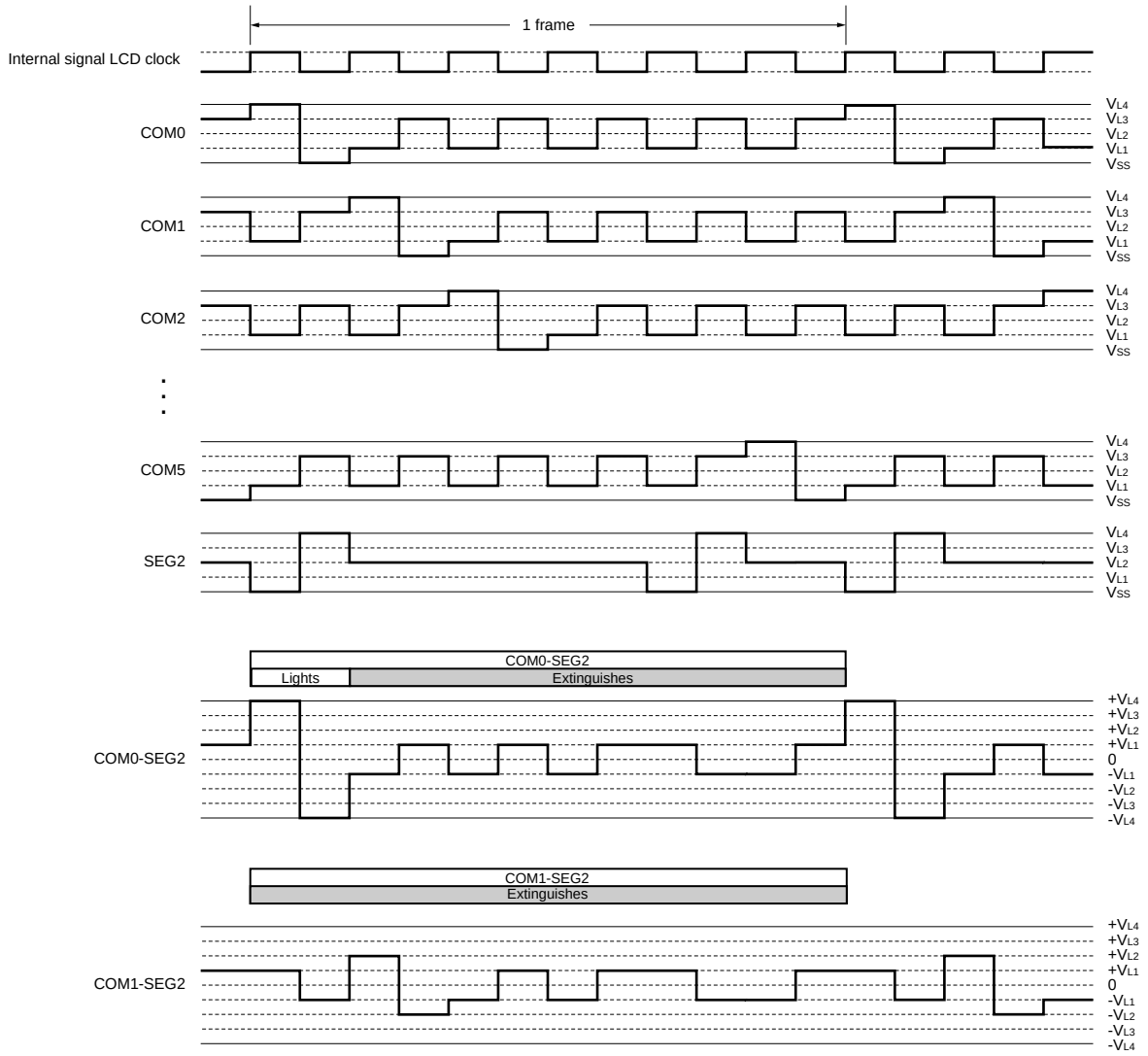
**Figure 21-39. Example of Connecting Six-Time-Slice LCD Panel**




×: Can always be used to store any data because the six-time-slice mode is being used.

**Figure 21-40. Six-Time-Slice LCD Drive Waveform Examples Between SEG4 and Each Common Signals (1/4 Bias Method)**

**(a) Waveform A**



### 21.10.6 Eight-time-slice display example

Figure 21-42 shows how the 15x8 dot LCD panel having the display pattern shown in Figure 21-41 is connected to the segment signals (SEG4 to SEG18) and the common signals (COM0 to COM7). This example displays data “123” in the LCD panel. The contents of the display data register (addresses F0404H to F0412H) correspond to this display.

The following description focuses on numeral “3.” ( ∃ ) displayed in the first digit. To display “3.” in the LCD panel, it is necessary to apply the select or deselect voltage to the SEG4 to SEG8 pins according to Table 21-19 at the timing of the common signals COM0 to COM7; see Figure 21-41 for the relationship between the segment signals and LCD segments.

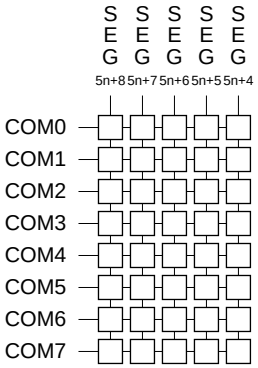
**Table 21-19. Select and Deselect Voltages (COM0 to COM7)**

| Segment / Common | SEG4 | SEG5 | SEG6 | SEG7 | SEG8 |
|---|---|---|---|---|---|
| COM0 | Select | Select | Select | Select | Select |
| COM1 | Deselect | Select | Deselect | Deselect | Deselect |
| COM2 | Deselect | Deselect | Select | Deselect | Deselect |
| COM3 | Deselect | Select | Deselect | Deselect | Deselect |
| COM4 | Select | Deselect | Deselect | Deselect | Deselect |
| COM5 | Select | Deselect | Deselect | Deselect | Select |
| COM6 | Deselect | Select | Select | Select | Deselect |
| COM7 | Deselect | Deselect | Deselect | Deselect | Deselect |

According to Table 21-19, it is determined that the display data register location (F0404H) that corresponds to SEG4 must contain 00110001.

Figure 21-43 shows examples of LCD drive waveforms between the SEG4 signal and each common signal. When the select voltage is applied to SEG4 at the timing of COM0, a waveform is generated to turn on the corresponding LCD segment.

**Figure 21-41. Eight-Time-Slice LCD Display Pattern and Electrode Connections**




**Remark** 100-pin products: n = 0 to 6

**Figure 21-42. Example of Connecting Eight-Time-Slice LCD Panel**

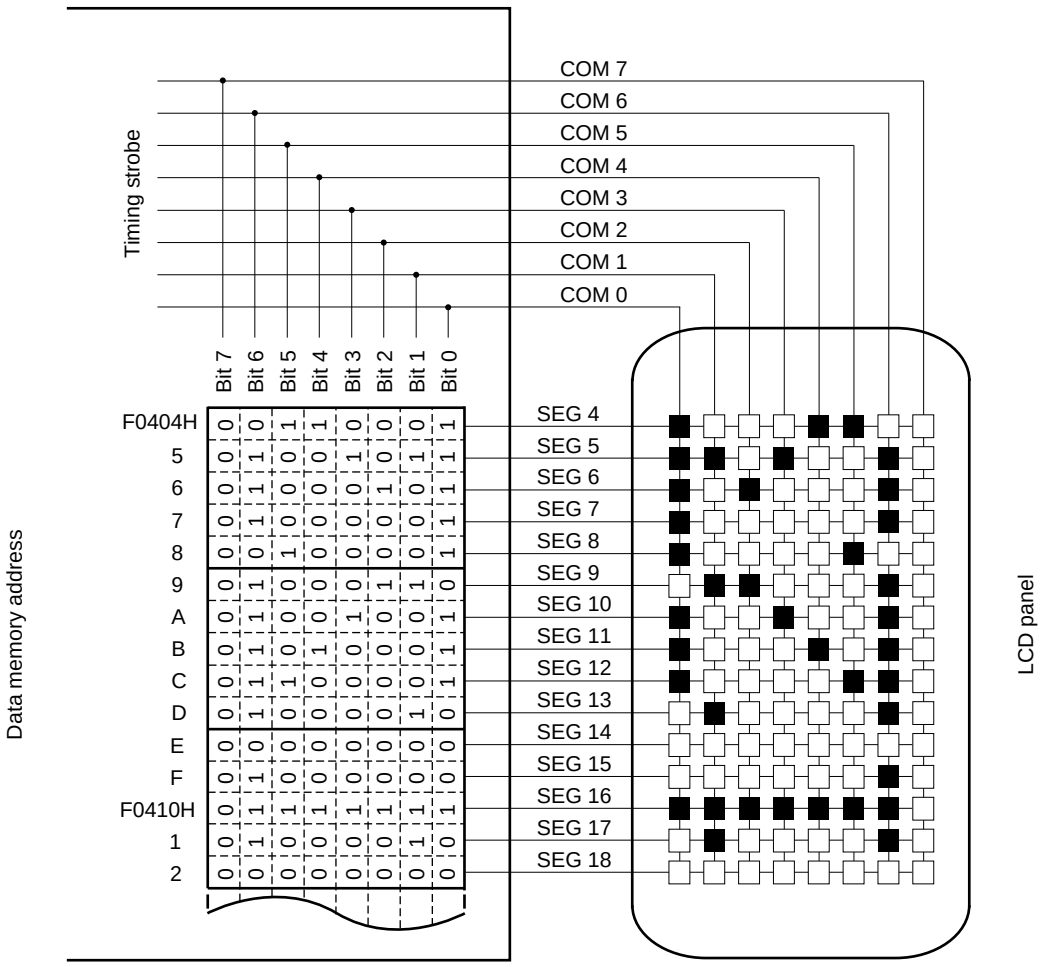


**Figure 21-43. Eight-Time-Slice LCD Drive Waveform Examples Between SEG4 and Each Common Signals (1/4 Bias Method) (1/2)**

**(a) Waveform A**

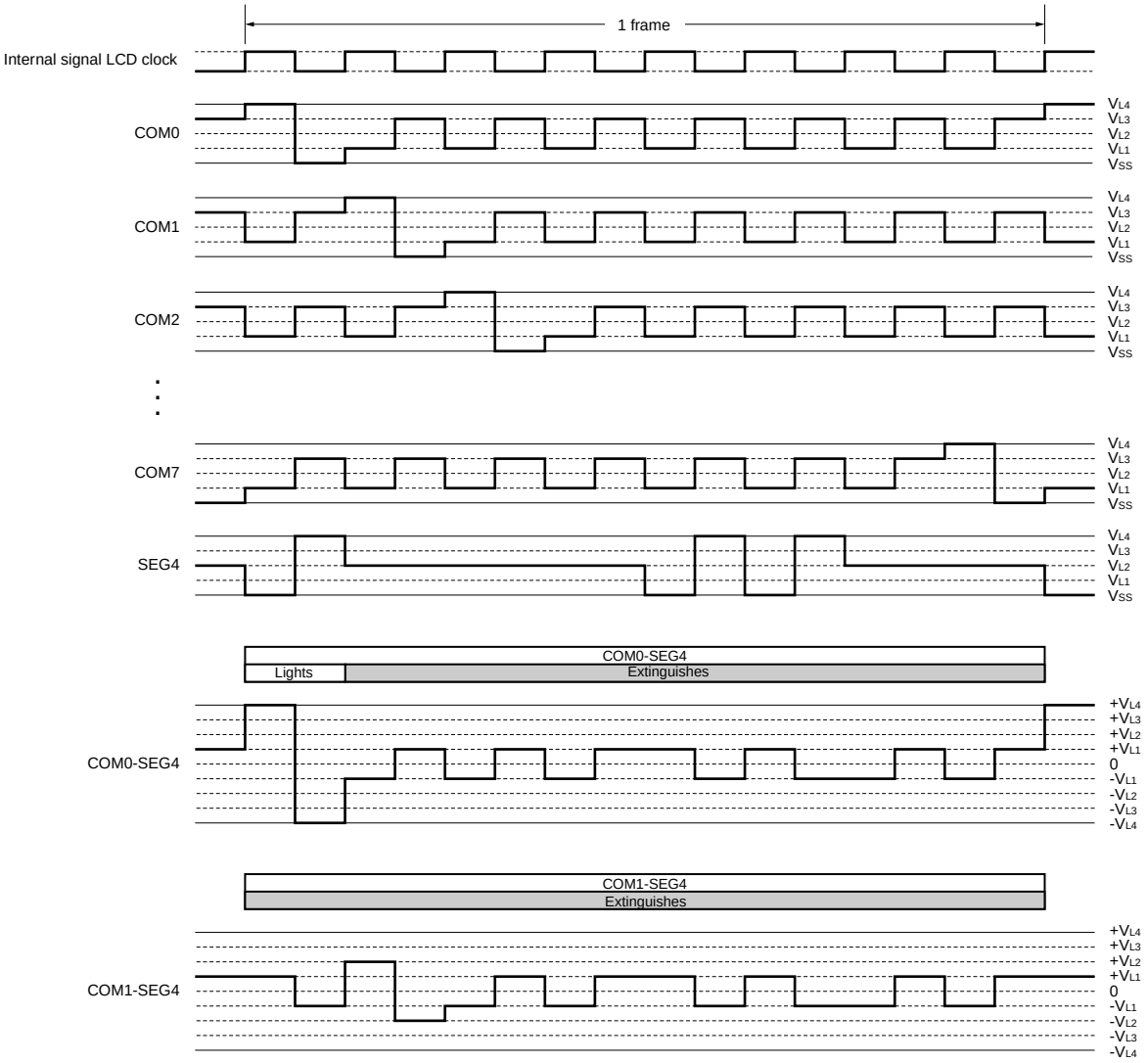
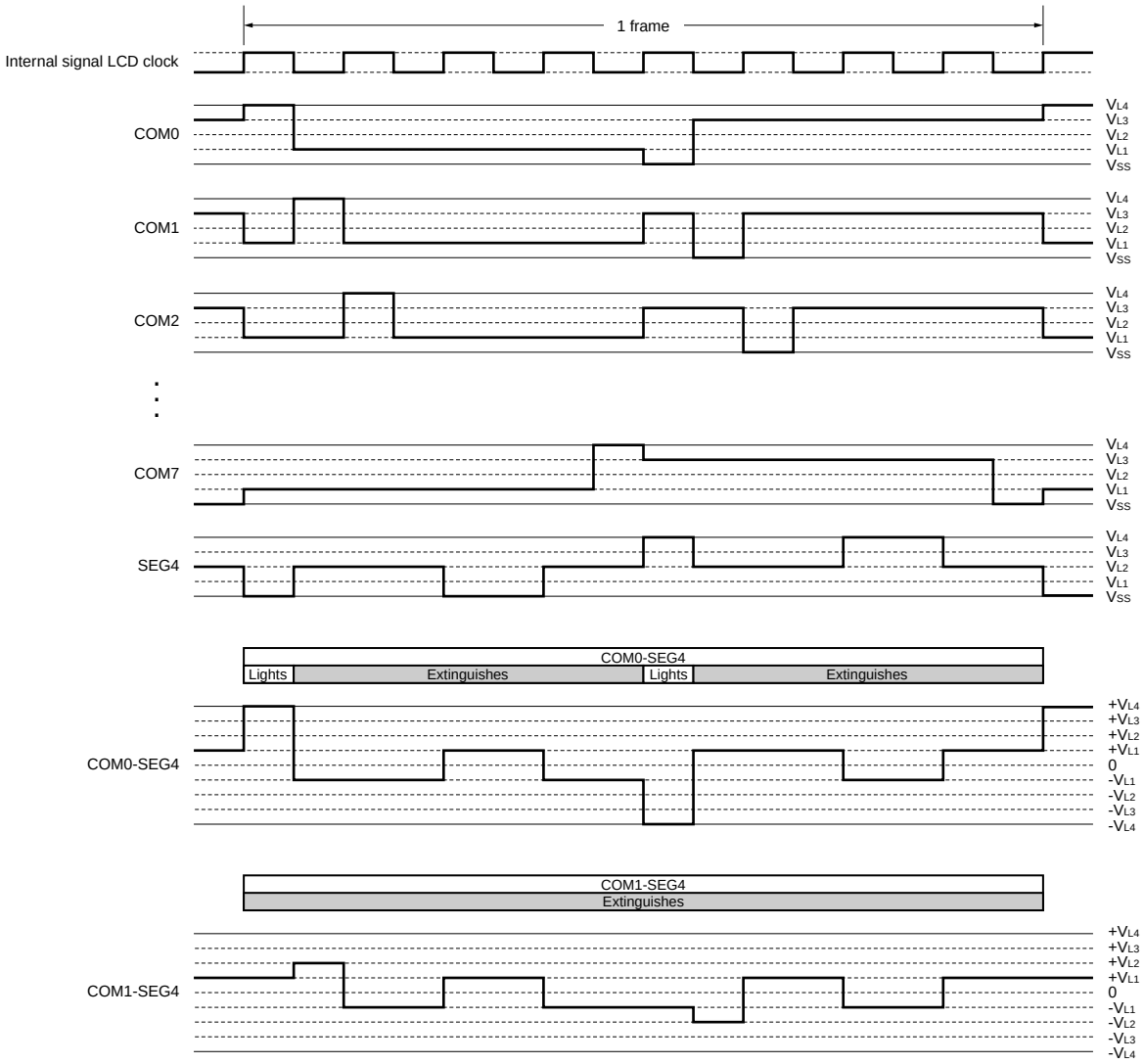


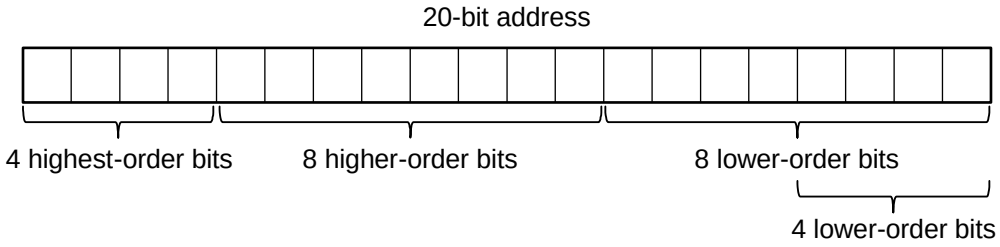
**Figure 21-43. Eight-Time-Slice LCD Drive Waveform Examples Between SEG4 and Each Common Signals (1/4 Bias Method) (2/2)**

**(b) Waveform B**



# CHAPTER 22 DATA TRANSFER CONTROLLER (DTC)

<R> The term “8 higher-order bits of the address” in this chapter indicates bits 15 to 8 of 20-bit address as shown below.




Unless otherwise specified, the 4 highest-order address bits all become 1 (values are of the form FxxxxH).

## 22.1 Functions of DTC

The data transfer controller (DTC) is a function that transfers data between memories without using the CPU. The DTC is activated by a peripheral function interrupt to perform data transfers. The DTC and CPU use the same bus, and the DTC takes priority over the CPU in using the bus.

Table 22-1 lists the DTC specifications.

**Table 22-1. DTC Specifications**

| Item | | Specification |
|---|---|---|
| <R> Activation sources | | 30 sources |
| Allocatable control data | | 24 sets |
| Address space which can be transferred | Address space | 64 Kbytes (F0000H to FFFFFH), excluding general-purpose registers |
| | Sources | Special function register (SFR), RAM area (excluding general-purpose registers), mirror area[Note], extended special function register (2nd SFR) |
| | Destinations | Special function register (SFR), RAM area (excluding general-purpose registers), extended special function register (2nd SFR) |
| Maximum number of transfers | Normal mode | 256 times |
| | Repeat mode | 255 times |
| Maximum size of block to be transferred | Normal mode (8-bit transfer) | 256 bytes |
| | Normal mode (16-bit transfer) | 512 bytes |
| | Repeat mode | 255 bytes |
| Unit of transfers | | 8 bits/16 bits |
| Transfer mode | Normal mode | Transfers end on completion of the transfer causing the DTCCTj register value to change from 1 to 0. |
| | Repeat mode | On completion of the transfer causing the DTCCTj register value to change from 1 to 0, the repeat area address is initialized and the DTRLDj register value is reloaded to the DTCCTj register to continue transfers. |
| Address control | Normal mode | Fixed or incremented |
| | Repeat mode | Addresses of the area not selected as the repeat area are fixed or incremented. |
| Priority of activation sources | | See **Table 22-5 DTC Activation Sources and Vector Addresses**. |
| Interrupt request | Normal mode | When the data transfer causing the DTCCTj register value to change from 1 to 0 is performed, the activation source interrupt request is generated for the CPU, and interrupt handling is performed on completion of the data transfer. |
| | Repeat mode | When the data transfer causing the DTCCTj register value to change from 1 to 0 is performed while the RPTINT bit in the DTCCRj register is 1 (interrupt generation enabled), the activation source interrupt request is generated for the CPU, and interrupt handling is performed on completion of the transfer. |
| Transfer start | | When bits DTCENi0 to DTCENi7 in the DTCENi registers are 1 (activation enabled), data transfer is started each time the corresponding DTC activation sources are generated. |
| Transfer stop | Normal mode | • When bits DTCENi0 to DTCENi7 are set to 0 (activation disabled).<br>• When the data transfer causing the DTCCTj register value to change from 1 to 0 is completed. |
| | Repeat mode | • When bits DTCENi0 to DTCENi7 are set to 0 (activation disabled).<br>• When the data transfer causing the DTCCTj register value to change from 1 to 0 is completed while the RPTINT bit is 1 (interrupt generation enabled). |

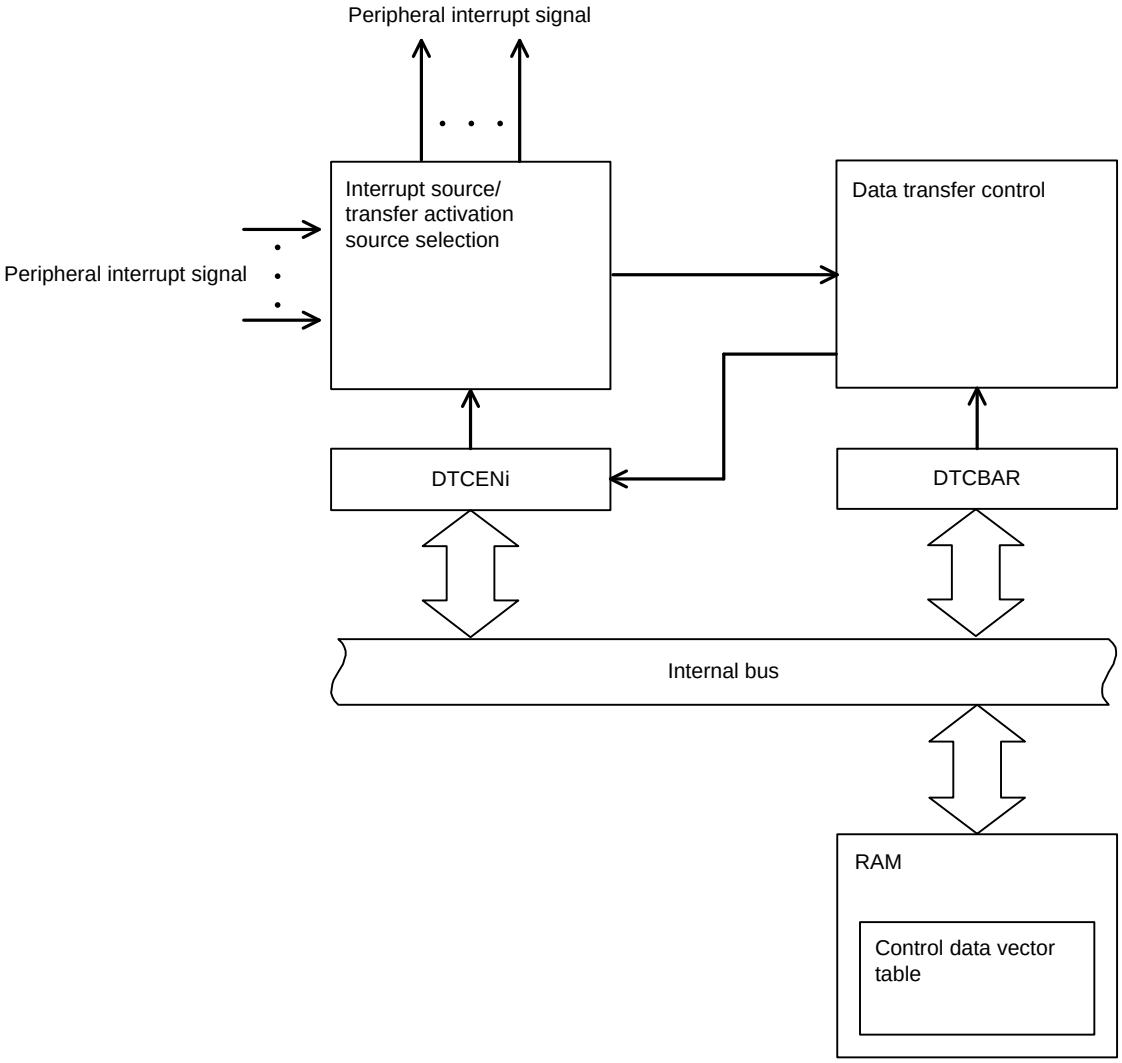
**Note** In the HALT and SNOOZE modes, these areas cannot be set as the sources for DTC transfer since the flash memory is stopped.

**Remark** i = 0 to 3, j = 0 to 23

## 22.2 Configuration of DTC

Figure 22-1 shows the DTC block diagram.

**Figure 22-1. DTC Block Diagram**



## 22.3 Registers Controlling DTC

Table 22-2 lists the registers controlling DTC.

**Table 22-2. Registers Controlling DTC**

| Register Name | Symbol |
|---|---|
| Peripheral Enable Register 1 | PER1 |
| DTC Activation Enable Register 0 | DTCEN0 |
| DTC Activation Enable Register 1 | DTCEN1 |
| DTC Activation Enable Register 2 | DTCEN2 |
| DTC Activation Enable Register 3 | DTCEN3 |
| DTC Base Address Register | DTCBAR |

Table 22-3 lists DTC control data.

DTC control data is allocated in the DTC control data area in RAM.

The DTCBAR register is used to set the 256-byte area, including the DTC control data area and the DTC vector table area where the start address for control data is stored.

**Table 22-3. DTC Control Data**

| Register Name | Symbol |
|---|---|
| DTC Control Register j | DTCCRj |
| DTC Block Size Register j | DTBLSj |
| DTC Transfer Count Register j | DTCCTj |
| DTC Transfer Count Reload Register j | DTRLDj |
| DTC Source Address Register j | DTSARj |
| DTC Destination Address Register j | DTDARj |

**Remark** j = 0 to 23

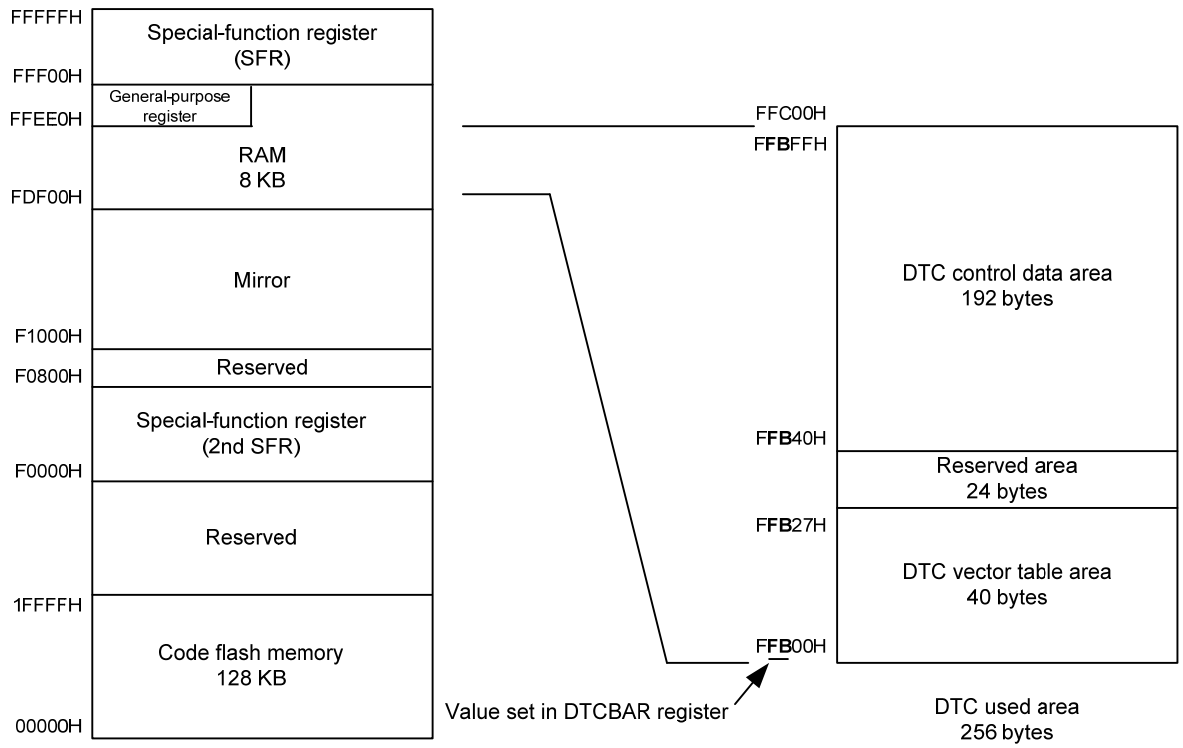
### 22.3.1 Allocation of DTC control data area and DTC vector table area

The DTCBAR register is used to set the 256-byte area where DTC control data and the vector table within the RAM area.

Figure 22-2 shows a memory map example when DTCBAR register is set to FBH.

In the 192-byte DTC control data area, the space not used by the DTC can be used as RAM.

**Figure 22-2. Memory Map Example When DTCBAR Register Is Set to FBH (R5F10MMGDFB, R5F10MPGDFB)**




The areas where the DTC control data and vector table can be allocated differ depending on the product.

**Cautions 1. It is prohibited to use the general-purpose register (FFEE0H to FFEFFH) space as the DTC control data area or DTC vector table area.**

**2. Make sure the stack area, the DTC control data area, and the DTC vector table area do not overlap.**

**3. The internal RAM area in the following products cannot be used as the DTC control data area or DTC vector table area when using the self-programming.**

**R5F10MMGDFB, R5F10MPGDFB: FDF00H to FE309H**

**R5F10MMEDFB, R5F10MPEDFB: FE700H to FEB09H**

**4. The internal RAM area of the following products cannot be used as the DTC control data area or DTC vector table area when using the trace function of on-chip debugging.**

**R5F10MME, R5F10MPE, R5F10MMG, R5F10MPG: FE300H to FE6FFH**

### 22.3.2 Control data allocation

Control data is allocated beginning with each start address in the order: Registers DTCCRj, DTBLSj, DTCCTj, DTRLDj, DTSARj, and DTDARj (j = 0 to 23).

The higher 8 bits for start addresses 0 to 23 are set by the DTCBAR register, and the lower 8 bits are separately set according to the vector table assigned to each activation source.

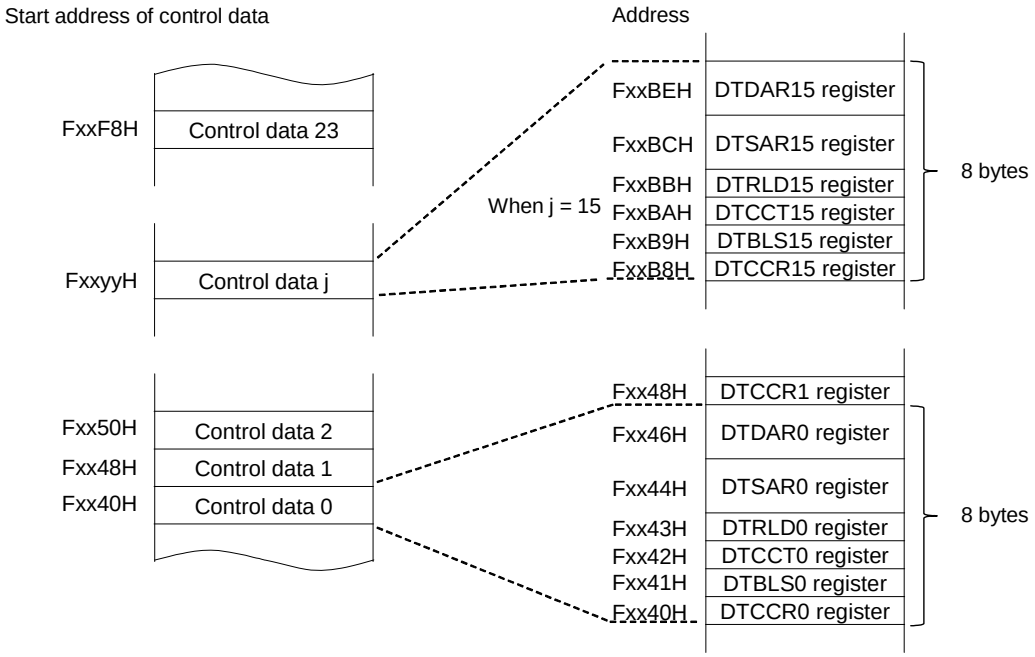
Figure 22-3 shows control data allocation.

**Notes 1.** Change the data in registers DTCCRj, DTBLSj, DTCCTj, DTRLDj, DTSARj, and DTDARj when the corresponding bit among bits DTCENi0 to DTCENi7 (i = 0 to 3) in the DTCENi register is set to 0 (DTC activation disabled).

**2.** Do not access DTCCRj, DTBLSj, DTCCTj, DTRLDj, DTSARj, or DTDARj using a DTC transfer.

<R>

**Figure 22-3. Control Data Allocation**




**Remark** xx: Value set in DTCBAR register

<R>

**Table 22 - 4 Start Address of Control Data**

| j | Address | j | Address |
|---|---|---|---|
| 11 | Fxx98H | 23 | FxxF8H |
| 10 | Fxx90H | 22 | FxxF0H |
| 9 | Fxx88H | 21 | FxxE8H |
| 8 | Fxx80H | 20 | FxxE0H |
| 7 | Fxx78H | 19 | FxxD8H |
| 6 | Fxx70H | 18 | FxxD0H |
| 5 | Fxx68H | 17 | FxxC8H |
| 4 | Fxx60H | 16 | FxxC0H |
| 3 | Fxx58H | 15 | FxxB8H |
| 2 | Fxx50H | 14 | FxxB0H |
| 1 | Fxx48H | 13 | FxxA8H |
| 0 | Fxx40H | 12 | FxxA0H |

**Remark** xx: Value set in DTCBAR register

### 22.3.3 Vector table

When the DTC is activated, one control data is selected according to the data read from the vector table which has been assigned to each activation source, and the selected control data is read from the DTC control data area.

Table 22-5 lists the activation sources and vector addresses. A one byte of the vector table is assigned to each activation source, and data from 40H to F8H is stored in each area to select one of the 24 control data sets. The higher 8 bits for the vector address are set by the DTCBAR register, and 00H to 27H are allocated to the lower 8 bits corresponding to the activation source.

**Note** Change the start address of the DTC control data area to be set in the vector table when the corresponding bit among bits DTCENi0 to DTCENi7 (i = 0 to 3) in the DTCENi register is set to 0 (activation disabled).

<R>

**Figure 22 - 4 Start Address of Control Data and Vector Table**

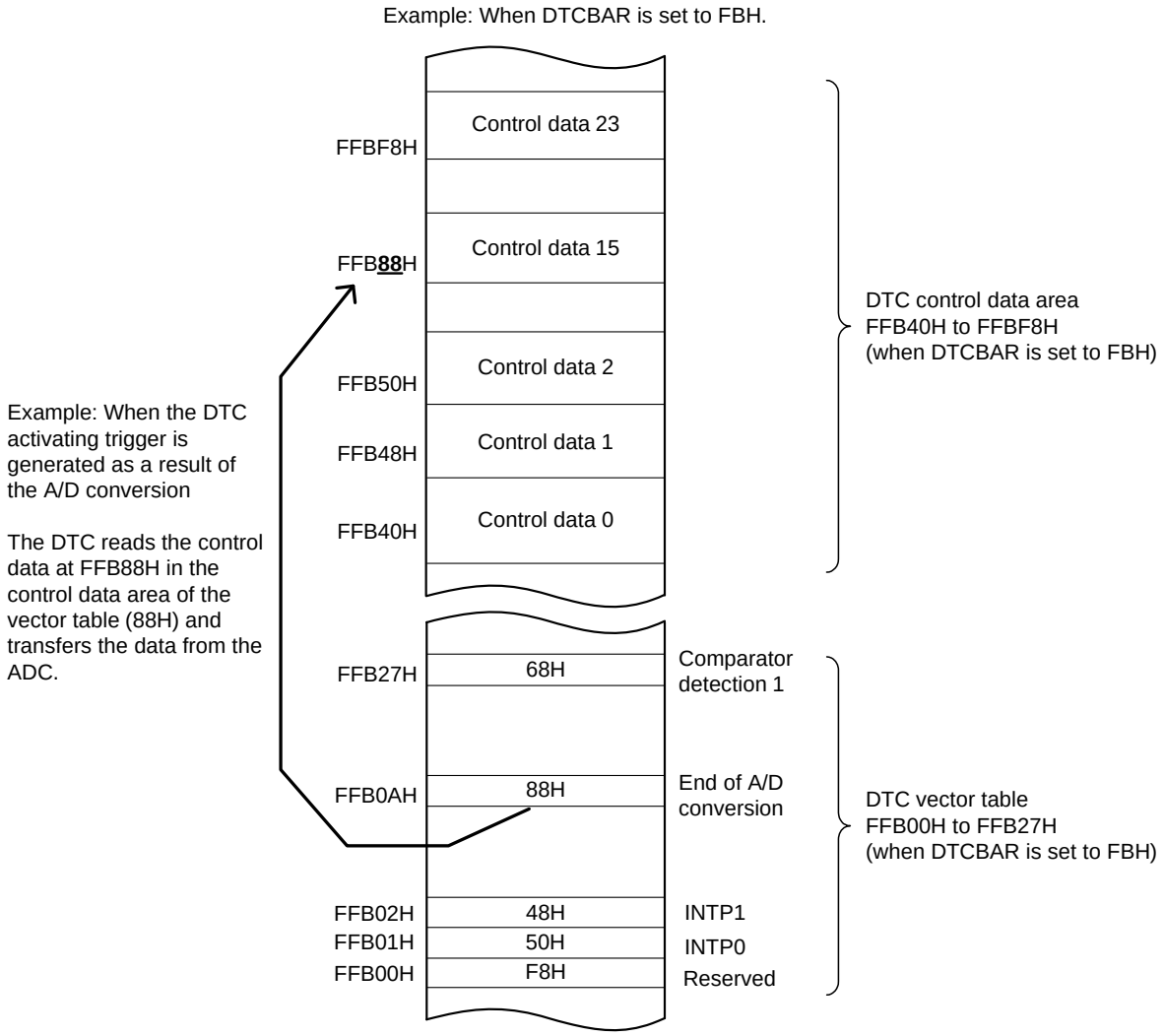


**Table 22-5. DTC Activation Sources and Vector Addresses**

| Interrupt Request Source | Source No. | Vector Address | Priority |
|---|---|---|---|
| Reserved | 0 | Address set in DTCBAR register +00H | Highest |
| INTP0 | 1 | Address set in DTCBAR register +01H | |
| INTP1 | 2 | Address set in DTCBAR register +02H | |
| INTP2 | 3 | Address set in DTCBAR register +03H | |
| INTP3 | 4 | Address set in DTCBAR register +04H | |
| INTP4 | 5 | Address set in DTCBAR register +05H | |
| INTP5 | 6 | Address set in DTCBAR register +06H | |
| INTP6 | 7 | Address set in DTCBAR register +07H | |
| INTP7 | 8 | Address set in DTCBAR register +08H | |
| 24-bit ΔΣ-type A/D converter | 9 | Address set in DTCBAR register +09H | |
| 10-bit SAR-type A/D conversion end | 10 | Address set in DTCBAR register +0AH | |
| UART0 reception transfer end | 11 | Address set in DTCBAR register +0BH | |
| UART0 transmission transfer end/CSI00 transfer end or buffer empty/IIC00 transfer end | 12 | Address set in DTCBAR register +0CH | |
| UART1 reception transfer end | 13 | Address set in DTCBAR register +0DH | |
| UART1 transmission transfer end/IIC10 transfer end | 14 | Address set in DTCBAR register +0EH | |
| UART2 reception transfer end | 15 | Address set in DTCBAR register +0FH | |
| UART2 transmission transfer end | 16 | Address set in DTCBAR register +10H | |
| End of channel 0 of timer array unit 0 count or capture | 17 | Address set in DTCBAR register +11H | |
| End of channel 1 of timer array unit 0 count or capture | 18 | Address set in DTCBAR register +12H | |
| End of channel 2 of timer array unit 0 count or capture | 19 | Address set in DTCBAR register +13H | |
| End of channel 3 of timer array unit 0 count or capture | 20 | Address set in DTCBAR register +14H | |
| End of channel 4 of timer array unit 0 count or capture | 21 | Address set in DTCBAR register +15H | |
| End of channel 5 of timer array unit 0 count or capture | 22 | Address set in DTCBAR register +16H | |
| End of channel 6 of timer array unit 0 count or capture | 23 | Address set in DTCBAR register +17H | |
| End of channel 7 of timer array unit 0 count or capture | 24 | Address set in DTCBAR register +18H | |
| 8-bit interval timer 00 | 25 | Address set in DTCBAR register +19H | |
| 8-bit interval timer 01 | 26 | Address set in DTCBAR register +1AH | |
| 8-bit interval timer 10 | 27 | Address set in DTCBAR register +1BH | |
| 8-bit interval timer 11 | 28 | Address set in DTCBAR register +1CH | |
| Comparator detection 0 | 29 | Address set in DTCBAR register +1DH | |
| Comparator detection 1 | 30 | Address set in DTCBAR register +1EH | |
| Reserved | 31 | Address set in DTCBAR register +1FH | |
| Reserved | 32 | Address set in DTCBAR register +20H | |
| Reserved | 33 | Address set in DTCBAR register +21H | |
| Reserved | 34 | Address set in DTCBAR register +22H | |
| Reserved | 35 | Address set in DTCBAR register +23H | |
| Reserved | 36 | Address set in DTCBAR register +24H | |
| Reserved | 37 | Address set in DTCBAR register +25H | |
| Reserved | 38 | Address set in DTCBAR register +26H | |
| Reserved | 39 | Address set in DTCBAR register +27H | Lowest |

### 22.3.4 Peripheral enable register 1 (PER1)

The PER1 register is used to enable or disable supplying the clock to the peripheral hardware. Clock supply to the hardware that is not used is also stopped so as to decrease the power consumption and noise.

When using the DTC, be sure to set bit 3 (DTCEN) to 1.

The PER1 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 22-5. Format of Peripheral Enable Register 1 (PER1)**

Address: F007AH After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PER1 | TMKAEN | FMCEN | CMPEN | OSDCEN | DTCEN | 0 | 0 | DSADCEN |

| DTCEN | Control of DTC input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• DTC cannot run. |
| 1 | Enables input clock supply.<br>• DTC can run. |

**Caution Be sure to clear bits 2 and 1 to “0”.**

### 22.3.5 DTC control register j (DTCCRj) (j = 0 to 23)

The DTCCRj register is used to control the DTC operating mode.

**Figure 22-6. Format of DTC Control Register j (DTCCRj)**

Address: See **22.3.2 Control data allocation**. After reset: Undefined R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DTCCRj | 0 | SZ | RPTINT | CHNE | DAMOD | SAMOD | RPTSEL | MODE |

| SZ | Transfer data size selection |
|---|---|
| 0 | 8 bits |
| 1 | 16 bits |

| RPTINT | Enabling/disabling repeat mode interrupts |
|---|---|
| 0 | Interrupt generation disabled |
| 1 | Interrupt generation enabled |
| The setting of the RPTINT bit is invalid when the MODE bit is 0 (normal mode). | |

| CHNE | Enabling/disabling chain transfers |
|---|---|
| 0 | Chain transfers disabled |
| 1 | Chain transfers enabled |
| Set the CHNE bit in the DTCCR23 register to 0 (chain transfers disabled). | |

| DAMOD | Transfer destination address control |
|---|---|
| 0 | Fixed |
| 1 | Incremented |
| The setting of the DAMOD bit is invalid when the MODE bit is 1 (repeat mode) and the RPTSEL bit is 0 (transfer destination is the repeat area). | |

| SAMOD | Transfer source address control |
|---|---|
| 0 | Fixed |
| 1 | Incremented |
| The setting of the SAMOD bit is invalid when the MODE bit is 1 (repeat mode) and the RPTSEL bit is 1 (transfer source is the repeat area). | |

| RPTSEL | Repeat area selection |
|---|---|
| 0 | Transfer destination is the repeat area |
| 1 | Transfer source is the repeat area |
| The setting of the RPTSEL bit is invalid when the MODE bit is 0 (normal mode). | |

| MODE | Transfer mode selection |
|---|---|
| 0 | Normal mode |
| 1 | Repeat mode |

**Caution Do not access the DTCCRj register using a DTC transfer.**

### 22.3.6 DTC block size register j (DTBLSj) (j = 0 to 23)

This register is used to set the block size of the data to be transferred by one activation.

**Figure 22-7. Format of DTC Block Size Register j (DTBLSj)**

Address: See **22.3.2 Control data allocation**. After reset: Undefined R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DTBLSj | DTBLSj7 | DTBLSj6 | DTBLSj5 | DTBLSj4 | DTBLSj3 | DTBLSj2 | DTBLSj1 | DTBLSj0 |

| DTBLSj | Transfer block size | |
|---|---|---|
| | 8-bit transfer | 16-bit transfer |
| 00H | 256 bytes | 512 bytes |
| 01H | 1 byte | 2 bytes |
| 02H | 2 bytes | 4 bytes |
| 03H | 3 bytes | 6 bytes |
| ... | ... | ... |
| FDH | 253 bytes | 506 bytes |
| FEH | 254 bytes | 508 bytes |
| FFH | 255 bytes | 510 bytes |

**Caution Do not access the DTBLSj register using a DTC transfer.**

### 22.3.7 DTC transfer count register j (DTCCTj) (j = 0 to 23)

This register is used to set the number of DTC data transfers. The value is decremented by 1 each time DTC transfer is activated once.

**Figure 22-8. Format of DTC Transfer Count Register j (DTCCTj)**

Address: See **22.3.2 Control data allocation**. After reset: Undefined R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DTCCTj | DTCCTj7 | DTCCTj6 | DTCCTj5 | DTCCTj4 | DTCCTj3 | DTCCTj2 | DTCCTj1 | DTCCTj0 |

| DTCCTj | Number of transfers |
|---|---|
| 00H | 256 times |
| 01H | Once |
| 02H | 2 times |
| 03H | 3 times |
| ... | ... |
| FDH | 253 times |
| FEH | 254 times |
| FFH | 255 times |

**Caution Do not access the DTCCTj register using a DTC transfer.**

### 22.3.8 DTC transfer count reload register j (DTRLDj) (j = 0 to 23)

This register is used to set the initial value of the transfer count register in repeat mode. Since the value of this register is reloaded to the DTCCT register in repeat mode, set the same value as the initial value of the DTCCT register.

**Figure 22-9. Format of DTC Transfer Count Reload Register j (DTRLDj)**

Address: See **22.3.2 Control data allocation**. After reset: Undefined R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DTRLDj | DTRLDj7 | DTRLDj6 | DTRLDj5 | DTRLDj4 | DTRLDj3 | DTRLDj2 | DTRLDj1 | DTRLDj0 |

**Caution Do not access the DTRLDj register using a DTC transfer.**

### 22.3.9 DTC source address register j (DTSARj) (j = 0 to 23)

This register is used to specify the transfer source address for data transfer.

When the SZ bit in the DTCCRj register is set to 1 (16-bit transfer), the lowest bit is ignored and the address is handled as an even address.

**Figure 22-10. Format of DTC Source Address Register j (DTSARj)**

Address: See **22.3.2 Control data allocation**. After reset: Undefined R/W

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DTSARj | DTSA Rj15 | DTSA Rj14 | DTSA Rj13 | DTS ARj12 | DTS ARj11 | DTSA Rj10 | DTS ARj9 | DTS ARj8 | DTS ARj7 | DTS ARj6 | DTS ARj5 | DTS ARj4 | DTS ARj3 | DTS ARj2 | DTS ARj1 | DTS ARj0 |

**Cautions 1. Do not set the general-purpose register (FFEE0H to FFEFFH) space to the transfer source address.**

**2. Do not access the DTSARj register using a DTC transfer.**

### 22.3.10 DTC destination address register j (DTDARj) (j = 0 to 23)

This register is used to specify the transfer destination address for data transfer.

When the SZ bit in the DTCCRj register is set to 1 (16-bit transfer), the lowest bit is ignored and the address is handled as an even address.

**Figure 22-11. Format of DTC Destination Address Register j (DTDARj)**

Address: See **22.3.2 Control data allocation**. After reset: Undefined R/W

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DTDARj | DTDA Rj15 | DTD ARj14 | DTD ARj13 | DTDA Rj12 | DTDA Rj11 | DTD ARj10 | DTD ARj9 | DTD ARj8 | DTD ARj7 | DTD ARj6 | DTD ARj5 | DTD ARj4 | DTD ARj3 | DTD ARj2 | DTD ARj1 | DTD ARj0 |

**Cautions 1. Do not set the general-purpose register (FFEE0H to FFEFFH) space to the transfer source address.**

**2. Do not access the DTDARj register using a DTC transfer.**

### 22.3.11 DTC activation enable register i (DTCENi) (i = 0 to 3)

This is an 8-bit register which enables or disables DTC activation by interrupt sources. Table 22-6 lists the correspondence between interrupt sources and bits DTCENi0 to DTCENi7.

The DTCENi register can be set by an 8-bit memory manipulation instruction and a 1-bit memory manipulation instruction.

**Notes 1.** Modify bits DTCENi0 to DTCENi7 if an activation source corresponding to the bit has not been generated.

**2.** Do not access the DTCENi register using a DTC transfer.

**Figure 22-12. DTC Activation Enable Register i (DTCENi) (i = 0 to 3)**

Address: F02E8H (DTCEN0), F02E9H (DTCEN1), F02EAH (DTCEN2), F02EBH (DTCEN3) — After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| DTCENi | DTCENi7 | DTCENi6 | DTCENi5 | DTCENi4 | DTCENi3 | DTCENi2 | DTCENi1 | DTCENi0 |

| DTCENi7 | DTC activation enable i7 |
|---|---|
| 0 | Activation disabled |
| 1 | Activation enabled |
| The DTCENi7 bit is set to 0 (activation disabled) by a condition for generating a transfer end interrupt. | |

| DTCENi6 | DTC activation enable i6 |
|---|---|
| 0 | Activation disabled |
| 1 | Activation enabled |
| The DTCENi6 bit is set to 0 (activation disabled) by a condition for generating a transfer end interrupt. | |

| DTCENi5 | DTC activation enable i5 |
|---|---|
| 0 | Activation disabled |
| 1 | Activation enabled |
| The DTCENi5 bit is set to 0 (activation disabled) by a condition for generating a transfer end interrupt. | |

| DTCENi4 | DTC activation enable i4 |
|---|---|
| 0 | Activation disabled |
| 1 | Activation enabled |
| The DTCENi4 bit is set to 0 (activation disabled) by a condition for generating a transfer end interrupt. | |

| DTCENi3 | DTC activation enable i3 |
|---|---|
| 0 | Activation disabled |
| 1 | Activation enabled |
| The DTCENi3 bit is set to 0 (activation disabled) by a condition for generating a transfer end interrupt. | |

| DTCENi2 | DTC activation enable i2 |
|---|---|
| 0 | Activation disabled |
| 1 | Activation enabled |
| The DTCENi2 bit is set to 0 (activation disabled) by a condition for generating a transfer end interrupt. | |

| DTCENi1 | DTC activation enable i1 |
|---|---|
| 0 | Activation disabled |
| 1 | Activation enabled |
| The DTCENi1 bit is set to 0 (activation disabled) by a condition for generating a transfer end interrupt. | |

| DTCENi0 | DTC activation enable i0 |
|---|---|
| 0 | Activation disabled |
| 1 | Activation enabled |
| The DTCENi0 bit is set to 0 (activation disabled) by a condition for generating a transfer end interrupt. | |

**Table 22-6. Correspondences Between Interrupt Sources and Bits DTCENi0 to DTCENi7**

| Register | DTCENi7 Bit | DTCENi6 Bit | DTCENi5 Bit | DTCENi4 Bit | DTCENi3 Bit | DTCENi2 Bit | DTCENi1 Bit | DTCENi0 Bit |
|---|---|---|---|---|---|---|---|---|
| DTCEN0 | Reserved | INTP0 | INTP1 | INTP2 | INTP3 | INTP4 | INTP5 | INTP6 |
| DTCEN1 | INTP7 | 24-bit ΔΣ-type A/D converter | A/D conversion end | UART0 reception transfer end | UART0 transmission transfer end/CSI00 transfer end or buffer empty/IIC00 transfer end | UART1 reception transfer end | UART1 transmission transfer end/IIC10 transfer end | UART2 reception transfer end |
| DTCEN2 | UART2 transmission transfer end | End of channel 0 of timer array unit 0 count or capture | End of channel 1 of timer array unit 0 count or capture | End of channel 2 of timer array unit 0 count or capture | End of channel 3 of timer array unit 0 count or capture | End of channel 4 of timer array unit 0 count or capture | End of channel 5 of timer array unit 0 count or capture | End of channel 6 of timer array unit 0 count or capture |
| DTCEN3 | End of channel 7 of timer array unit 0 count or capture | 8-bit interval timer 00 | 8-bit interval timer 01 | 8-bit interval timer 10 | 8-bit interval timer 11 | Comparator detection 0 | Comparator detection 1 | Reserved |

**Remark** i = 0 to 3

### 22.3.12 DTC base address register (DTCBAR)

This is an 8-bit register used to set the following addresses: the vector address where the start address of the DTC control data area is stored and the address of the DTC control data area. The value of the DTCBAR register is handled as the higher 8 bits to generate a 16-bit address.

**Cautions 1. Change the DTCBAR register value with all DTC activation sources set to activation disabled.**

**2. Do not rewrite the DTCBAR register more than once.**

**3. Do not access the DTCBAR register using a DTC transfer.**

**4. For the allocation of the DTC control data area and the DTC vector table area, see the Notes on 22.3.1 Allocation of DTC control data area and DTC vector table area.**

**Figure 22-13. Format of DTC Base Address Register (DTCBAR)**

Address: F02E0H After reset: FDH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DTCBAR | DTCBAR7 | DTCBAR6 | DTCBAR5 | DTCBAR4 | DTCBAR3 | DTCBAR2 | DTCBAR1 | DTCBAR0 |

## 22.4 DTC Operation

When the DTC is activated, control data is read from the DTC control data area to perform data transfers and control data after data transfer is written back to the DTC control data area. Twenty-four sets of control data can be stored in the DTC control data area, which allows 24 types of data transfers to be performed.

There are two transfer modes (normal mode and repeat mode) and two transfer sizes (8-bit transfer and 16-bit transfer). When the CHNE bit in the DTCCRj (j = 0 to 23) register is set to 1 (chain transfers enabled), multiple control data is read and data transfers are continuously performed by one activation source (chain transfers).

A transfer source address is specified by the 16-bit register DTSARj, and a transfer destination address is specified by the 16-bit register DTDARj.

The values in registers DTSARj and DTDARj are separately incremented or fixed according to the control data after the data transfer.

### 22.4.1 Activation sources

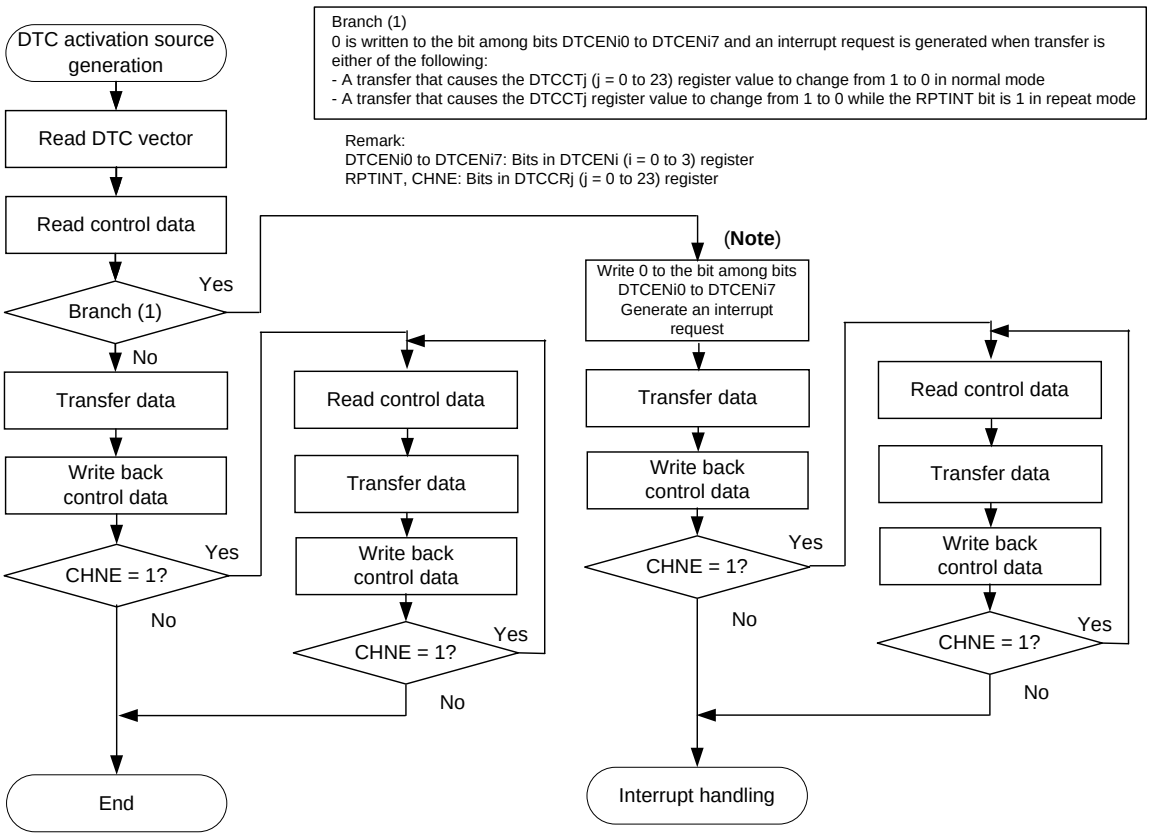
The DTC is activated by an interrupt signal from the peripheral functions. The interrupt signals to activate the DTC are selected with the DTCENi (i = 0 to 3) register.

The DTC sets the corresponding bit among bits DTCENi0 to DTCENi7 in the DTCENi register to 0 (activation disabled) during operation when the setting of data transfer (the first transfer in chain transfers) is either of the following:

- A transfer that causes the DTCCTj (j = 0 to 23) register value to change to 0 in normal mode
- A transfer that causes the DTCCTj register value to change to 0 while the RPTINT bit in the DTCCRj register is 1 (interrupt generation enabled) in repeat mode

Figure 22-14 shows the DTC internal operation flowchart.

**Figure 22-14. DTC Internal Operation Flowchart**




**Note** 0 is not written to the bit among bits DTCENi0 to DTCENi7 for data transfers activated by the setting to enable chain transfers (the CHNE bit is 1). Also, no interrupt request is generated.

### 22.4.2 Normal mode

One to 256 bytes of data are transferred by one activation during 8-bit transfer and 2 to 512 bytes during 16-bit transfer. The number of transfers can be 1 to 256 times. When the data transfer causing the DTCCTj (j = 0 to 23) register value to change to 0 is performed, the DTC generates an interrupt request corresponding to the activation source to the interrupt controller during DTC operation, and sets the corresponding bit among bits DTCENi0 to DTCENi7 (i = 0 to 3) in the DTCENi register to 0 (activation disabled).

Table 22-7 shows register functions in normal mode. Figure 22-15 shows data transfers in normal mode.

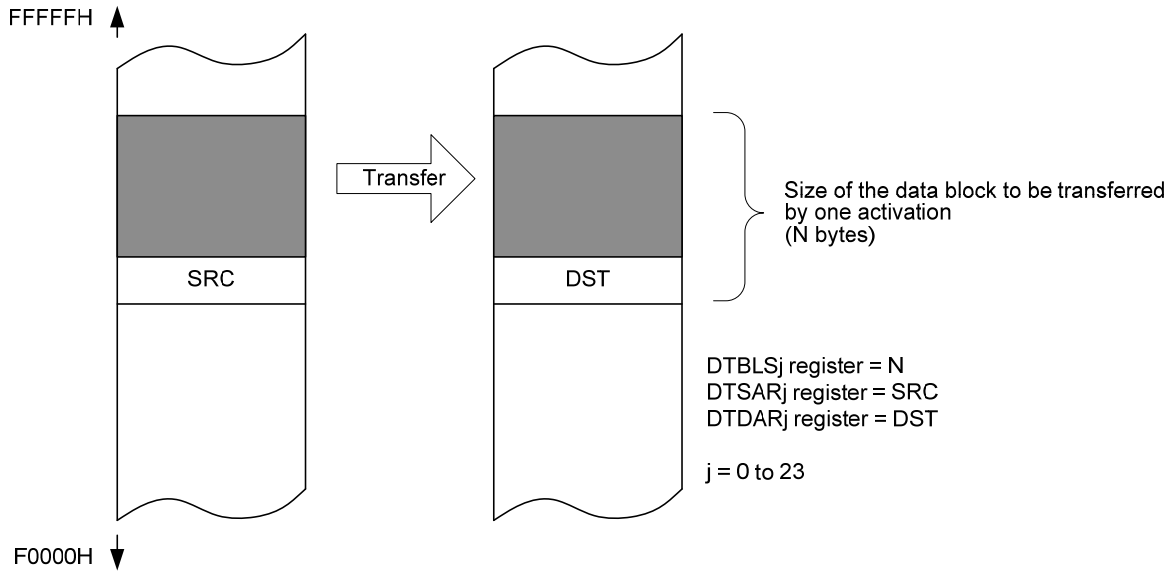
**Table 22-7. Register Functions in Normal Mode**

| Register Name | Symbol | Function |
|---|---|---|
| DTC block size register j | DTBLSj | Size of the data block to be transferred by one activation |
| DTC transfer count register j | DTCCTj | Number of data transfers |
| DTC transfer count reload register j | DTRLDj | Not used[Note] |
| DTC source address register j | DTSARj | Data transfer source address |
| DTC destination address register j | DTDARj | Data transfer destination address |

**Note** Initialize this register to 00H when parity error resets are enabled (RPERDIS = 0) using the RAM parity error detection function.

**Remark** j = 0 to 23

**Figure 22-15. Data Transfers in Normal Mode**




| DTCCR Register Setting | | | | Source Address Control | Destination Address Control | Source Address After Transfer | Destination Address After Transfer |
|---|---|---|---|---|---|---|---|
| DAMOD | SAMOD | RPTSEL | MODE | | | | |
| 0 | 0 | X | 0 | Fixed | Fixed | SRC | DST |
| 0 | 1 | X | 0 | Incremented | Fixed | SRC + N | DST |
| 1 | 0 | X | 0 | Fixed | Incremented | SRC | DST + N |
| 1 | 1 | X | 0 | Incremented | Incremented | SRC + N | DST + N |

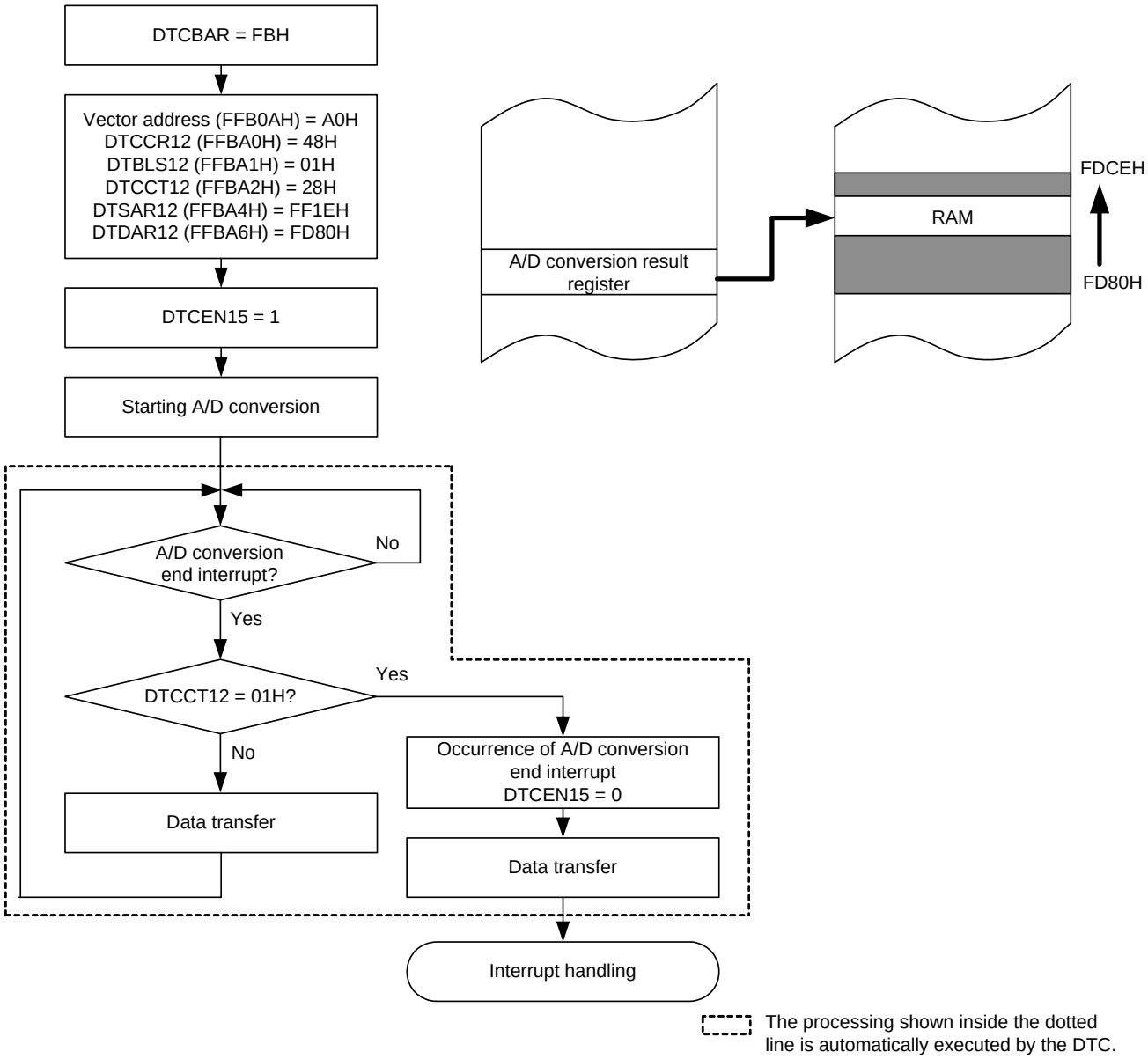
X: 0 or 1

**(1) Example 1 of using normal mode: Consecutively capturing A/D conversion results**

The DTC is activated by an A/D conversion end interrupt and the value of the A/D conversion result register is transferred to RAM.

- The vector address is FFB0AH and control data is allocated at FFBA0H to FFBA7H
- Transfers 2-byte data of the A/D conversion result register (FFF1EH, FFF1FH) to 80 bytes of FFD80H to FFDCFH of RAM 40 times

<R> **Figure 22-16. Example 1 of Using Normal Mode: Consecutively Capturing A/D Conversion Results**



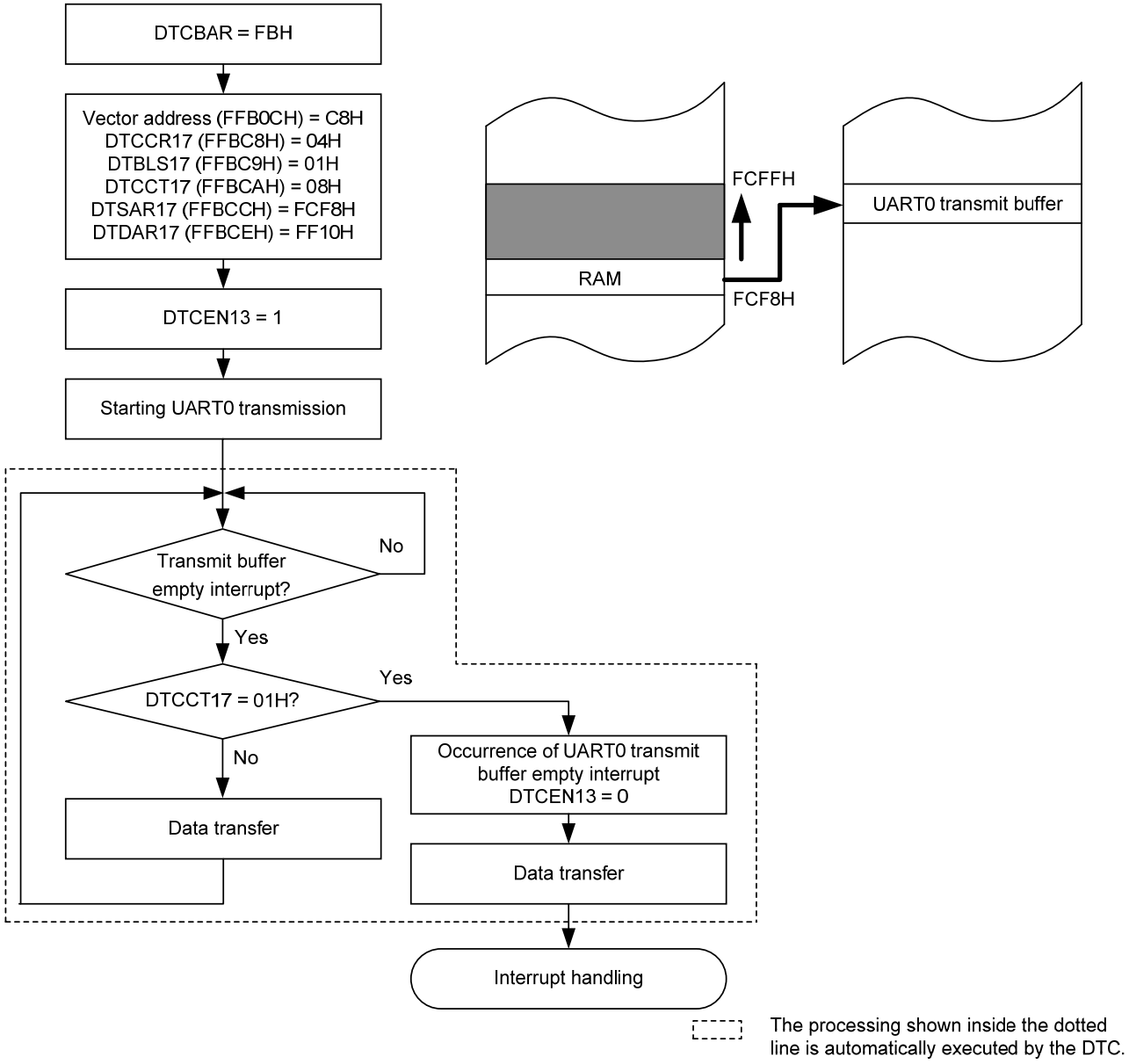

The value of the DTRLD12 register is not used because of normal mode, but initialize the register to 00H when parity error resets are enabled (RPERDIS = 0) using the RAM parity error detection function.

**(2) Example 2 of using normal mode: UART0 consecutive transmission**

The DTC is activated by a UART0 transmit buffer empty interrupt and the value of RAM is transferred to the UART0 transmit buffer.

- The vector address is FFB0CH and control data is allocated at FFBC8H to FFBCFH
- Transfers 8 bytes of FFCF8H to FFCFFH of RAM to the UART0 transmit buffer (FFF10H)

**Figure 22-17. Example 2 of Using Normal Mode: UART0 Consecutive Transmission**




The value of the DTRLD17 register is not used because of normal mode, but initialize the register to 00H when parity error resets are enabled (RPERDIS = 0) using the RAM parity error detection function.
Start the first UART0 transmission by software. The second and subsequent transmissions are automatically sent when the DTC is activated by a transmit buffer empty interrupt.

### 22.4.3 Repeat mode

One to 255 bytes of data are transferred by one activation. Either of the transfer source or destination should be specified as the repeat area. The number of transfers can be 1 to 255 times. On completion of the specified number of transfers, the DTCCTj (i = 0 to 23) register and the address specified for the repeat area are initialized to continue transfers. When the data transfer causing the DTCCTj register value to change to 0 is performed while the RPTINT bit in the DTCCRj register is 1 (interrupt generation enabled), the DTC generates an interrupt request corresponding to the activation source to the interrupt controller during DTC operation, and sets the corresponding bit among bits DTCENi0 to DTCENi7 to 0 (activation disabled). When the RPTINT bit in the DTCCRj register is 0 (interrupt generation disabled), no interrupt request is generated even if the data transfer causing the DTCCTj register value to change to 0 is performed. Also, bits DTCENi0 to DTCENi7 are not set to 0.

Table 22-8 lists register functions in repeat mode. Figure 22-18 shows data transfers in repeat mode.

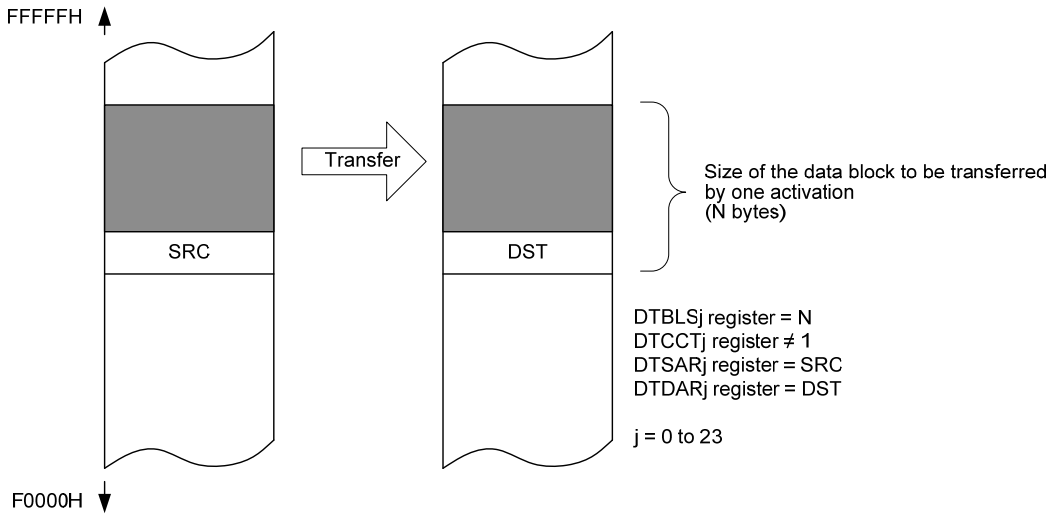
**Table 22-8. Register Functions in Repeat Mode**

| Register Name | Symbol | Function |
|---|---|---|
| DTC block size register j | DTBLSj | Size of the data block to be transferred by one activation |
| DTC transfer count register j | DTCCTj | Number of data transfers |
| DTC transfer count reload register j | DTRLDj | This register value is reloaded to the DTCCT register (the number of transfers is initialized). |
| DTC source address register j | DTSARj | Data transfer source address |
| DTC destination address register j | DTDARj | Data transfer destination address |

**Remark** j = 0 to 23

**Figure 22-18. Data Transfers in Repeat Mode**

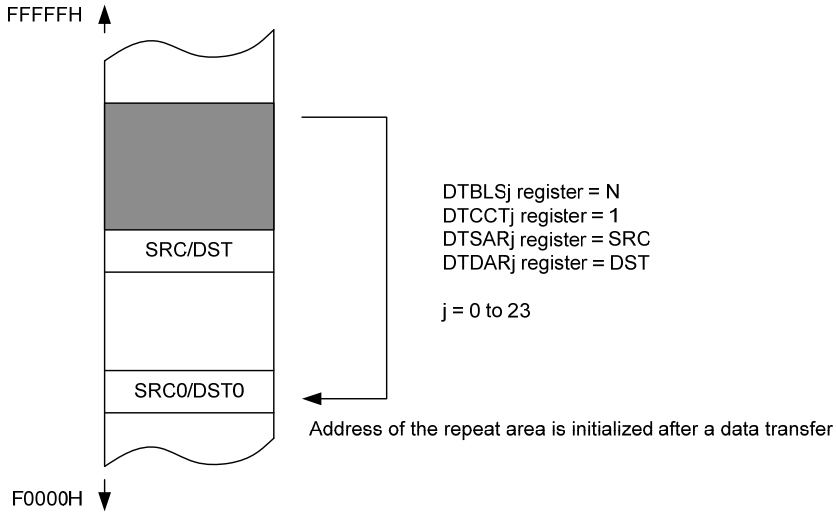
DTCCTj register ≠ 1




| DTCCR Register Setting | | | | Source Address Control | Destination Address Control | Source Address After Transfer | Destination Address After Transfer |
|---|---|---|---|---|---|---|---|
| DAMOD | SAMOD | RPTSEL | MODE | | | | |
| 0 | X | 1 | 1 | Repeat area | Fixed | SRC + N | DST |
| 1 | X | 1 | 1 | Repeat area | Incremented | SRC + N | DST + N |
| X | 0 | 0 | 1 | Fixed | Repeat area | SRC | DST + N |
| X | 1 | 0 | 1 | Incremented | Repeat area | SRC + N | DST + N |

X: 0 or 1

DTCCTj register = 1




| DTCCR Register Setting | | | | Source Address Control | Destination Address Control | Source Address After Transfer | Destination Address After Transfer |
|---|---|---|---|---|---|---|---|
| DAMOD | SAMOD | RPTSEL | MODE | | | | |
| 0 | X | 1 | 1 | Repeat area | Fixed | SRC0 | DST |
| 1 | X | 1 | 1 | Repeat area | Incremented | SRC0 | DST + N |
| X | 0 | 0 | 1 | Fixed | Repeat area | SRC | DST0 |
| X | 1 | 0 | 1 | Incremented | Repeat area | SRC + N | DST0 |

SRC0: Initial source address value
DST0: Initial destination address value
X: 0 or 1

**Cautions 1. When repeat mode is used, the lower 8 bits of the initial value for the repeat area address must be 00H.**

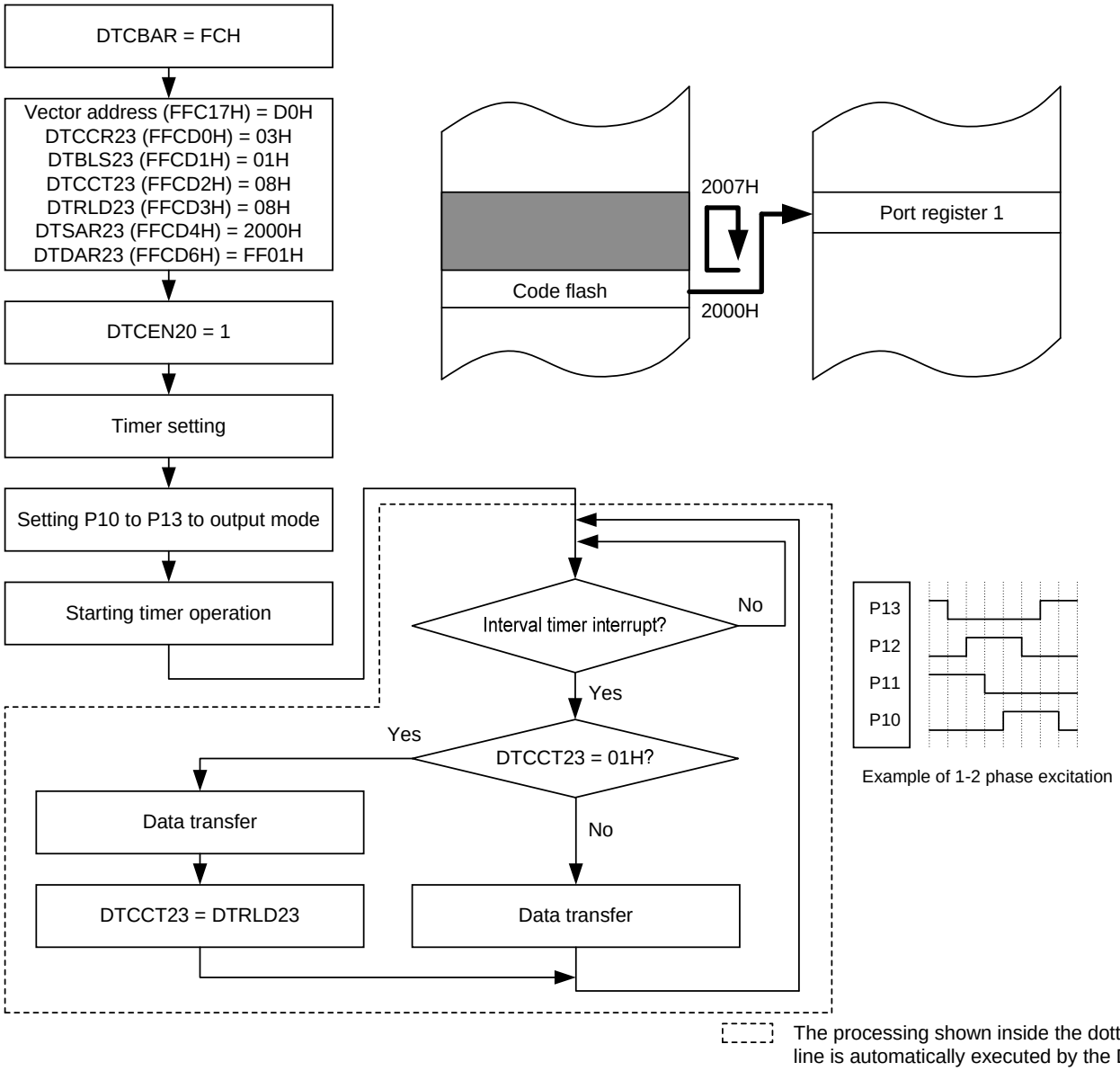
**2. When repeat mode is used, the data size of the repeat area must be set to 255 bytes or less.**

**(1) Example of using repeat mode: Outputting a stepping motor control pulse using ports**

The DTC is activated by an interval timer interrupt and the pattern of the motor control pulse stored in the code flash memory is transferred to general-purpose ports.

- The vector address is FFC0CH and control data is allocated at FFCD0H to FFCD7H
- Transfers 8-byte data of 02000H to 02007H of the code flash memory from the mirror space (F2000H to F2007H) to port register 1 (FFF01H)
- A repeat mode interrupt is disabled

**Figure 22-19. Example 1 of Using Repeat Mode: Outputting a Stepping Motor Control Pulse Using Ports**




To stop the output, stop the timer first and then clear DTCEN20.

### 22.4.4 Chain transfers

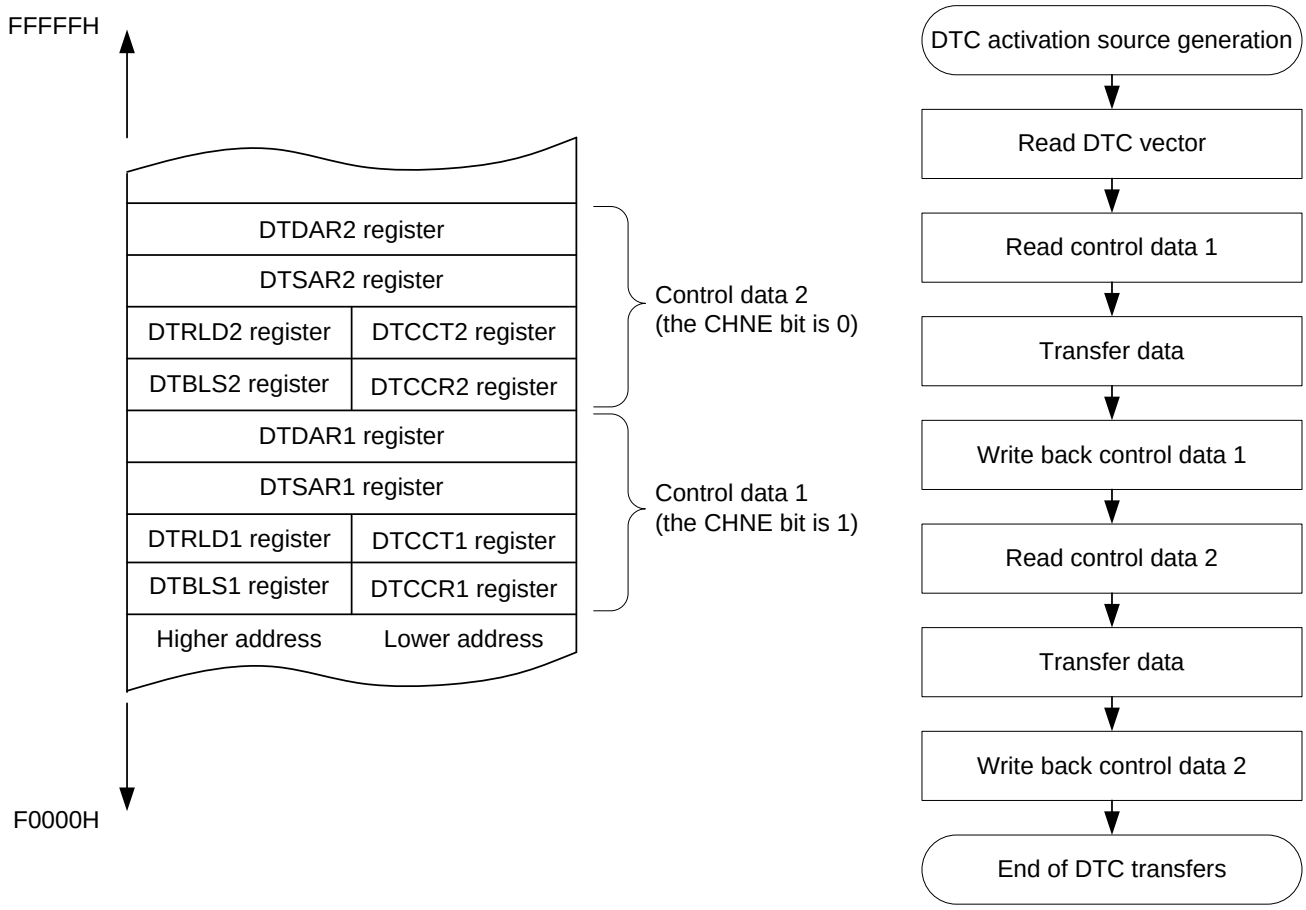
When the CHNE bit in the DTCCRj (j = 0 to 22) register is 1 (chain transfers enabled), multiple data transfers can be continuously performed by one activation source.

When the DTC is activated, one control data is selected according to the data read from the DTC vector address corresponding to the activation source, and the selected control data is read from the DTC control data area. When the CHNE bit for the control data is 1 (chain transfers enabled), the next control data immediately following the current control data is read and transferred after the current transfer is completed. This operation is repeated until the data transfer with the control data for which the CHNE bit is 0 (chain transfers disabled) is completed.

When chain transfers are performed using multiple control data, the number of transfers set for the first control data is enabled, and the number of transfers set for the second and subsequent control data to be processed will be invalid

Figure 22-20 shows data transfers during chain transfers.

**Figure 22-20. Data Transfers During Chain Transfers**




**Notes 1.** Set the CHNE bit in the DTCCR23 register to 0 (chain transfers disabled).

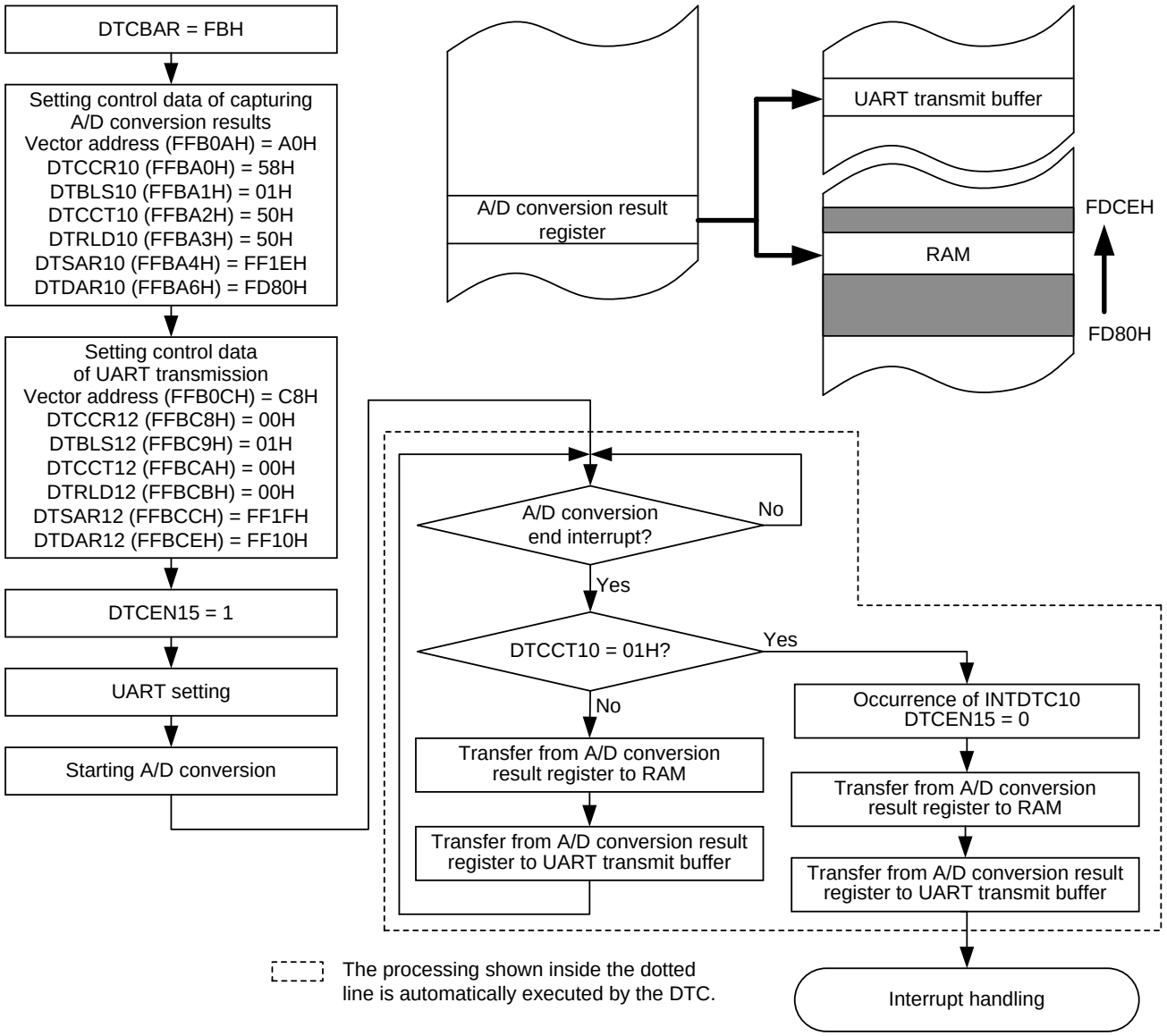
**2.** During chain transfers, bits DTCENi0 to DTCENi7 (i = 0 to 3) in the DTCENi register are not set to 0 (DTC activation disabled) for the second and subsequent transfers. Also, no interrupt request is generated.

**(1) Example of using chain transfers: Consecutively capturing A/D conversion results and UART transmission**

The DTC is activated by an A/D conversion end interrupt and A/D conversion results are transferred to RAM, and then transmitted using the UART.

- The vector address is FFB0AH
- Control data of capturing A/D conversion results is allocated at FFBA0H to FFBA7H
- Control data of UART transmission is allocated at FFBA8H at FFBAFH
- An A/D conversion end interrupt is assigned to TRIGER23
- Transfers 2-byte data of the A/D conversion result register (FFF1FH, FFF1EH) to FFD80H to FFDCFH of RAM, and transfers the upper 1 byte (FFF1FH) of the A/D conversion result register to the UART transmit buffer (FFF10H)

**Figure 22-21. Example of Using Chain Transfers: Consecutively Capturing A/D Conversion Results and UART Transmission**



## 22.5 Notes on DTC

### 22.5.1 Setting DTC control data and vector table

- Do not access the DTC SFRs, the DTC control data area, the DTC vector table area, or the general-register (FFEE0H to FFEFFH) space using a DTC transfer.
- Modify the DTC base address register (DTCBAR) while all DTC activation sources are set to activation disabled.
- Do not rewrite the DTC base address register (DTCBAR) twice or more.
- Modify the data of the DTCCRj, DTBLSj, DTCCTj, DTRLDj, DTSARj, or DTDARj register when the corresponding bit among bits DTCENi0 to DTCENi7 in the DTCENi (i = 0 to 3) register is 0 (DTC activation disabled).
- Modify the start address of the DTC control data area to be set in the vector table when the corresponding bit among bits DTCENi0 to DTCENi7 in the DTCENi (i = 0 to 3) register is 0 (DTC activation disabled).
- Do not allocate RAM addresses which are used as a DTC transfer destination/transfer source to the area FFE20H to FFEDFH when performing self-programming.

### 22.5.2 Allocation of DTC control data area and DTC vector table area

The areas where the DTC control data and vector table can be allocated differ.

- It is prohibited to use the general-purpose register (FFEE0H to FFEFFH) space as the DTC control data area or DTC vector table area.
- Make sure the stack area, the DTC control data area, and the DTC vector table area do not overlap.
- The internal RAM area in the following products cannot be used as the DTC control data area or DTC vector table area when using the self-programming.
  R5F10MMGDFB, R5F10MPGDFB : FDF00H-FE309H
  R5F10MMEDFB, R5F10MPEDFB : FE700H-FEB09H
- The internal RAM area in the following products cannot be used as the DTC control data area or DTC vector table area when using the on-chip trace function.
  R5F10MME, R5F10MPE, R5F10MMG, R5F10MPG: FE300H to FE6FFH
- Initialize the DTRLD register to 00H even in normal mode when parity error resets are enabled (RPERDIS = 0) using the RAM parity error detection function.

### 22.5.3 DTC pending instruction

Even if a DTC transfer request is generated, data transfer is held pending immediately after the following instructions. Also, the DTC is not activated between PREFIX instruction code and the instruction immediately after that code.

- Call/return instruction
- Unconditional branch instruction
- Conditional branch instruction
- Read access instruction for code flash memory
- Bit manipulation instructions for IFxx, MKxx, PRxx, and PSW, and an 8-bit manipulation instruction that has the ES register as operand

<R>
- Instruction of Multiply, Divide, Multiply & Accumulate (excluding MULU)

**Cautions 1. When a DTC transfer request is acknowledged, all interrupt requests are held pending until DTC transfer is completed.**

**2. While the DTC is held pending by the DTC pending instruction, all interrupt requests are held pending.**

### 22.5.4 Number of DTC execution clock cycles

Table 22-9 lists the operations following DTC activation and required number of clock cycles for each operation.

**Table 22-9. Operations Following DTC Activation and Required Number of Cycles**

| Vector Read | Control Data | | Data Read | Data Write |
|---|---|---|---|---|
| | Read | Write-back | | |
| 1 | 4 | **Note 1** | **Note 2** | **Note 2** |

**Notes 1.** For the number of clock cycles required for control data write-back, see **Table 22-10 Number of Clock Cycles Required for Control Data Write-Back Operation**.

**2.** For the number of clock cycles required for data read/write, see **Table 22-11 Number of Clock Cycles Required for Data Read/Write Operation**.

**Table 22-10. Number of Clock Cycles Required for Control Data Write-Back Operation**

| DTCCR Register Setting | | | | Address Setting | | Control Register to be Written Back | | | | Number of Clock Cycles |
|---|---|---|---|---|---|---|---|---|---|---|
| DAMOD | SAMOD | RPTSEL | MODE | Source | Destination | DTCCTj Register | DTRLDj Register | DTSARj Register | DTDARj Register | |
| 0 | 0 | X | 0 | Fixed | Fixed | Written back | Written back | Not written back | Not written back | 1 |
| 0 | 1 | X | 0 | Incremented | Fixed | Written back | Written back | Written back | Not written back | 2 |
| 1 | 0 | X | 0 | Fixed | Incremented | Written back | Written back | Not written back | Written back | 2 |
| 1 | 1 | X | 0 | Incremented | Incremented | Written back | Written back | Written back | Written back | 3 |
| 0 | X | 1 | 1 | Repeat area | Fixed | Written back | Written back | Written back | Not written back | 2 |
| 1 | X | 1 | 1 | | Incremented | Written back | Written back | Written back | Written back | 3 |
| X | 0 | 0 | 1 | Fixed | Repeat area | Written back | Written back | Not written back | Written back | 2 |
| X | 1 | 0 | 1 | Incremented | | Written back | Written back | Written back | Written back | 3 |

**Remark** j = 0 to 23; X: 0 or 1

**Table 22-11. Number of Clock Cycles Required for Data Read/Write Operation**

| Operation | RAM | Code Flash Memory | SFR | 2nd SFR | |
|---|---|---|---|---|---|
| | | | | No Wait State | Wait States |
| Data read | 1 | 2 | 1 | 1 | 1 + number of wait states[Note] |
| Data write | 1 | - | 1 | 1 | 1 + number of wait states[Note] |

**Note** The number of wait states differs depending on the specifications of the register allocated to the second SFR to be accessed.

### 22.5.5 DTC response time

Table 22-12 lists the DTC response time. The DTC response time is the time from when the DTC activation source is detected until DTC transfer starts, excluding the number of DTC execution clocks.

**Table 22-12. DTC Response Time**

| | Minimum Time | Maximum Time |
|---|---|---|
| Response Time | 3 clocks | 19 clocks |

Note that the response from the DTC may be further delayed under the following cases. The number of delayed clock cycles differs depending on the conditions.

- When executing an instruction from the internal RAM
  Maximum response time: 20 clocks
- When executing a DTC pending instruction (see **22.5.3 DTC pending instruction**)
  Maximum response time: Maximum response time for each condition + execution clock cycles for the instruction to be held pending under the condition.
- When accessing a register that a wait occurs
  Maximum response time: Maximum response time for each condition + 1 clock

**Remark** 1 clock: $1/f_{CLK}$ ($f_{CLK}$: CPU/peripheral hardware clock)

### 22.5.6 DTC activation sources

- After inputting a DTC activation source, do not input the same activation source again until DTC transfer is completed.
- While a DTC activation source is generated, do not manipulate the DTC activation enable bit corresponding to the source.
- If DTC activation sources conflict, their priority levels are determined in order to select the source for activation when the CPU acknowledges the DTC transfer. For details on the priority levels of activation sources, see **22.3.3 Vector table**.
- When DTC activation is enabled under either of the following conditions, a DTC transfer is started and an interrupt is generated after completion of the transfer. Therefore, enable DTC activation after confirming the comparator monitor flag (CnMON) as necessary. (n = 0, 1)
  - The comparator is set to an interrupt request on one-edge detection (CnEDG = 0), an interrupt request at the rising edge for the comparator, and IVCMP > IVREF (or internal reference voltage: 1.45 V)
  - The comparator is set to an interrupt request on one-edge detection (CnEDG = 0), an interrupt request at the falling edge for the comparator, and IVCMP < IVREF (or internal reference voltage: 1.45 V)

### 22.5.7 Operation in standby mode status

| Status | DTC Operation |
|---|---|
| HALT mode | Operable (Operation is disabled while in the low power consumption RTC mode) |
| STOP mode | DTC activation sources can be accepted[Note 2] |
| SNOOZE mode | Operable[Notes 1, 3, 4, 5] |

**Notes 1**. The SNOOZE mode can only be specified when the high-speed on-chip oscillator clock is selected as $f_{CLK}$.

**2.** In the STOP mode, detecting a DTC activation source enables transition to SNOOZE mode and DTC transfer. After completion of transfer, the system returns to the STOP mode. However, since the code flash memory is stopped during the HALT or SNOOZE mode, the flash memory cannot be set as the transfer source.

**3.** When a transfer end interrupt is set as a DTC activation source from the CSIp SNOOZE mode function, release the SNOOZE mode using the transfer end interrupt to start CPU processing after completion of DTC transfer, or use a chained transfer to set CSIp reception again (writing 1 to the STm0 bit, writing 0 to the SWCm bit, setting of the SSCm register, and writing 1 to the SSm0 bit).

**4.** When a transfer end interrupt is set as a DTC activation source from the UARTq SNOOZE mode function, release the SNOOZE mode using the transfer end interrupt to start CPU processing after completion of DTC transfer, or use a chained transfer to set UARTq reception again (writing 1 to the STm1 bit, writing 0 to the SWCm bit, setting of the SSCm register, and writing 1 to the SSm1 bit).

**5.** When an A/D conversion end interrupt is set as a DTC activation source from the A/D converter SNOOZE mode function, release the SNOOZE mode using the A/D conversion end interrupt to start CPU processing after completion of DTC transfer, or use a chained transfer to set the A/D converter SNOOZE mode function again after clear the AWC bit.

**Caution The SNOOZE function for the DTC and the SNOOZE function for UART cannot be used at the same time.**

**Remark** p = 00; q = 0; m = 0

# CHAPTER 23 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.

| | | Number of interrupt sources |
|---|---|---|
| Maskable interrupts | External | 10 |
| | Internal | 33 |

## 23.1 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. Default priority, see **Table 23-1**.

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 interrupts**

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.

## 23.2 Interrupt Sources and Configuration

Interrupt sources include maskable interrupts and software interrupts. In addition, they also have up to seven reset sources (see **Table 23-1**). 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 23-1. Interrupt Source List (1/3)**

| Interrupt Type | Default Priority[Note 1] | Interrupt Source | | Internal/ External | Vector Table Address | Basic Configuration Type[Note 2] |
|---|---|---|---|---|---|---|
| | | Name | Trigger | | | |
| Maskable | 0 | INTWDTI | Watchdog timer interval[Note 3] (75% of overflow time+1/2$f_{IL}$) | Internal | 0004H | (A) |
| | 1 | INTLVI | Voltage detection[Note 4] | | 0006H | |
| | 2 | INTP0[Note 5] | Pin input edge detection | External | 0008H | (B) |
| | 3 | INTP1 | | | 000AH | |
| | 4 | INTP2 | | | 000CH | |
| | 5 | INTP3 | | | 000EH | |
| | 6 | INTP4 | | | 0010H | |
| | 7 | INTP5 | | | 0012H | |
| | 8 | INTST2 | UART2 transmission transfer end or buffer empty interrupt | Internal | 0014H | (A) |
| | 9 | INTSR2 | UART2 reception transfer end | | 0016H | |
| | 10 | INTSRE2 | UART2 reception communication error occurrence | | 0018H | |
| | 11 | INTST0/ INTCSI00/ INTIIC00 | UART0 transmission transfer end or buffer empty interrupt/CSI00 transfer end or buffer empty interrupt/IIC00 transfer end | | 001EH | |
| | 12 | INTTM00 | End of timer channel 00 count or capture | | 0020H | |
| | 13 | INTSR0 | UART0 reception transfer end | | 0022H | |
| | 14 | INTSRE0 | UART0 reception communication error occurrence | | 0024H | |
| | | INTTM01H | End of timer channel 01 count or capture (at higher 8-bit timer operation) | | | |
| | 15 | INTST1/ INTIIC10 | UART1 transmission transfer end or buffer empty interrupt/IIC10 transfer end | | 0026H | |
| | 16 | INTSR1 | UART1 reception transfer end | | 0028H | |
| | 17 | INTSRE1 | UART1 reception communication error occurrence | | 002AH | |
| | | INTTM03H | End of timer channel 03 count or capture (at higher 8-bit timer operation) | | | |
| | 18 | INTIICA0 | End of IICA0 communication | | 002CH | |
| | 19 | INTRTIT | RTC correction timing | | 002EH | |
| | 20 | INTFM | End of frequency measurement | | 0030H | |
| | 21 | INTTM01 | End of timer channel 01 count or capture (at 16-bit/lower 8-bit timer operation) | | 0032H | |

**Notes 1.** The default priority determines the sequence of interrupts if two or more maskable interrupts occur simultaneously. Zero indicates the highest priority and 42 indicates the lowest priority.

**2.** Basic configuration types (A) to (C) correspond to (A) to (C) in Figure 23-1.

**3.** When bit 7 (WDTINT) of the option byte (000C0H) is set to 1.

**4.** When bit 7 (LVIMD) of the voltage detection level register (LVIS) is cleared to 0.

**5.** The input buffer power supply of the INTP0 pin is connected to internal $V_{DD}$. Interrupts can be accepted even when a battery backup function is used and power is supplied from the VBAT pin.

**Table 23-1. Interrupt Source List (2/3)**

| Interrupt Type | Default Priority Note 1 | Interrupt Source | | Internal/ External | Vector Table Address | Basic Configuration Type Note 2 |
|---|---|---|---|---|---|---|
| | | Name | Trigger | | | |
| Maskable | 22 | INTTM02 | End of timer channel 02 count or capture | Internal | 0034H | (A) |
| | 23 | INTTM03 | End of timer channel 03 count or capture (at 16-bit/lower 8-bit timer operation) | | 0036H | |
| | 24 | INTAD | End of A/D conversion | | 0038H | |
| | 25 | INTRTC | Fixed-cycle signal of real-time clock 2/alarm match detection | | 003AH | |
| | 26 | INTIT | Interval signal of 12-bit interval timer detection | | 003CH | |
| | 27 | INTDSAD | End of ΔΣ A/D conversion | | 0044H | |
| | 28 | INTTM04 | End of timer channel 04 count or capture | | 0046H | |
| | 29 | INTTM05 | End of timer channel 05 count or capture | | 0048H | |
| | 30 | INTP6 | Pin input edge detection | External | 004AH | (B) |
| | 31 | INTP7 | | | 004CH | |
| | 32 | INTCMP0 | Comparator detection 0 | | 0050H | |
| | 33 | INTCMP1 | Comparator detection 1 | | 0052H | |
| | 34 | INTTM06 | End of timer channel 06 count or capture | Internal | 0054H | (A) |
| | 35 | INTTM07 | End of timer channel 07 count or capture | | 0056H | |
| | 36 | INTIT00 | 8-bit interval timer channel 00/channel 0 (when cascade) compare match detection | | 0058H | |
| | 37 | INTIT01 | 8-bit interval timer channel 01 compare match detection | | 005AH | |
| | 38 | INTCR | End of high-speed on-chip oscillator clock frequency correction | | 005CH | |
| | 39 | INTOSDC | Oscillation stop detection | | 0060H | |
| | 40 | INTIT10 | 8-bit interval timer channel 10/channel 1 (when cascade) compare match detection | | 0068H | |
| | 41 | INTIT11 | 8-bit interval timer channel 11 compare match detection | | 006AH | |
| | 42 | INTVBAT | Power switching detection interrupt | | 006CH | |

**Notes 1.** The default priority determines the sequence of interrupts if two or more maskable interrupts occur simultaneously. Zero indicates the highest priority and 42 indicates the lowest priority.

**2.** Basic configuration types (A) to (C) correspond to (A) to (C) in Figure 23-1.

**Table 23-1. Interrupt Source List (3/3)**

| Interrupt Type | Default Priority[Note 1] | Interrupt Source | | Internal/ External | Vector Table Address | Basic Configuration Type[Note 2] |
|---|---|---|---|---|---|---|
| Software | – | BRK | Execution of BRK instruction | – | 007EH | (C) |
| Reset | – | RESET | $\overline{\text{RESET}}$ pin input | – | 0000H | – |
| | | POR | Power-on-reset | | | |
| | | LVD | Voltage detection[Note 3] | | | |
| | | WDT | Overflow of watchdog timer | | | |
| | | TRAP | Execution of illegal instruction[Note 4] | | | |
| | | IAW | Illegal-memory access | | | |
| | | RPE | RAM parity error | | | |

**Notes 1.** The default priority determines the sequence of interrupts if two or more maskable interrupts occur simultaneously. Zero indicates the highest priority and 42 indicates the lowest priority.

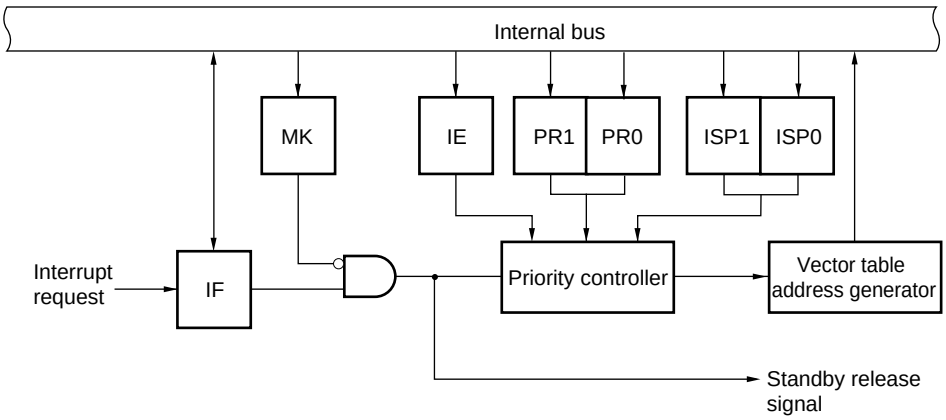
**2.** Basic configuration types (A) to (C) correspond to (A) to (C) in Figure 23-1.

**3.** When bit 7 (LVIMD) of the voltage detection level register (LVIS) is set to 1.

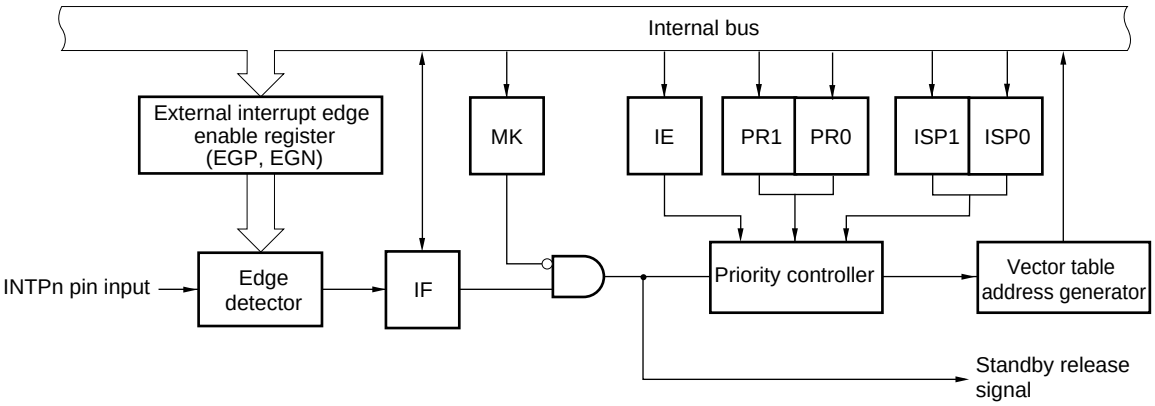
**4.** When the instruction code in FFH is executed.
Reset by the illegal instruction execution not issued by emulation with the in-circuit emulator or on-chip debug emulator.

**Figure 23-1. Basic Configuration of Interrupt Function**

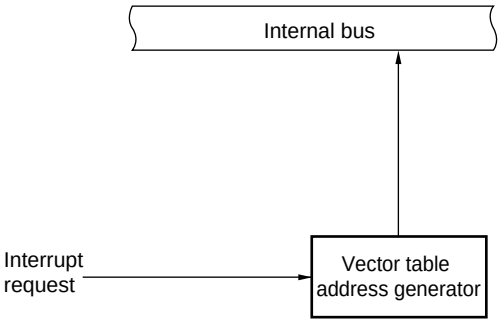
**(A) Internal maskable interrupt**




**(B) External maskable interrupt (INTPn)**




**(C) Software interrupt**




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** n = 0 to 7

## 23.3 Registers Controlling Interrupt Functions

The following 6 types of 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 register (EGP0)
- External interrupt falling edge enable register (EGN0)
- Program status word (PSW)

Table 23-2 shows a list of interrupt request flags, interrupt mask flags, and priority specification flags corresponding to interrupt request sources.

**Table 23-2. Flags Corresponding to Interrupt Request Sources (1/3)**

| Interrupt Source | Interrupt Request Flag | | Interrupt Mask Flag | | Priority Specification Flag | |
|---|---|---|---|---|---|---|
| | | Register | | Register | | Register |
| INTWDTI | WDTIIF | IF0L | WDTIMK | MK0L | WDTIPR0, WDTIPR1 | PR00L, PR10L |
| INTLVI | LVIIF | | LVIMK | | LVIPR0, LVIPR1 | |
| 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 | |
| INTST2 | STIF2 | IF0H | STMK2 | MK0H | STPR02, STPR12 | PR00H, PR10H |
| INTSR2 | SRIF2 | | SRMK2 | | SRPR02, SRPR12 | |
| INTSRE2 | SREIF2 | | SREMK2 | | SREPR02, SREPR12 | |
| INTCSI00[Note] | CSIIF00[Note] | | CSIMK00[Note] | | CSIPR000, CSIPR100[Note] | |
| INTIIC00[Note] | IICIF00[Note] | | IICMK00[Note] | | IICPR000, IICPR100[Note] | |
| INTST0[Note] | STIF0[Note] | | STMK0[Note] | | STPR00, STPR10[Note] | |
| INTTM00 | TMIF00 | | TMMK00 | | TMPR000, TMPR100 | |
| INTSR0 | SRIF0 | | SRMK0 | | SRPR00, SRPR10 | |

**Note** If one 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.

**Table 23-2. Flags Corresponding to Interrupt Request Sources (2/3)**

| Interrupt Source | Interrupt Request Flag | Register | Interrupt Mask Flag | Register | Priority Specification Flag | Register |
|---|---|---|---|---|---|---|
| INTSRE0[Note 1] | SREIF0[Note 1] | IF1L | SREMK0[Note 1] | MK1L | SREPR00, SREPR10[Note 1] | PR01L, PR11L |
| INTTM01H[Note 1] | TMIF01H[Note 1] | | TMMK01H[Note 1] | | TMPR001H, TMPR101H[Note 1] | |
| INTST1[Note 2] | STIF1[Note 2] | | STMK1[Note 2] | | STPR01, STPR11[Note 2] | |
| INTIIC10[Note 2] | IICIF10[Note 2] | | IICMK10[Note 2] | | IICPR010, IICPR110[Note 2] | |
| INTSR1 | SRIF1 | | SRMK1 | | SRPR01, SRPR11 | |
| INTSRE1[Note 3] | SREIF1[Note 3] | | SREMK1[Note 3] | | SREPR01, SREPR11[Note 3] | |
| INTTM03H[Note 3] | TMIF03H[Note 3] | | TMMK03H[Note 3] | | TMPR003H, TMPR103H[Note 3] | |
| INTIICA0 | IICAIF0 | | IICAMK0 | | IICAPR00, IICAPR10 | |
| INTRTIT | RTITIF | | RTITMK | | RTITPR0, RTITPR1 | |
| INTFM | FMIF | | FMMK | | FMPR0, FMPR1 | |
| INTTM01 | TMIF01 | | TMMK01 | | TMPR001, TMPR101 | |
| INTTM02 | TMIF02 | IF1H | TMMK02 | MK1H | TMPR002, TMPR102 | PR01H, PR11H |
| INTTM03 | TMIF03 | | TMMK03 | | TMPR003, TMPR103 | |
| INTAD | ADIF | | ADMK | | ADPR0, ADPR1 | |
| INTRTC | RTCIF | | RTCMK | | RTCPR0, RTCPR1 | |
| INTIT | TMKAIF | | TMKAMK | | TMKAPR0, TMKAPR1 | |
| INTDSAD | DSAIF | IF2L | DSAMK | MK2L | DSAPR0, DSAPR1 | PR02L, PR12L |
| INTTM04 | TMIF04 | | TMMK04 | | TMPR004, TMPR104 | |
| INTTM05 | TMIF05 | | TMMK05 | | TMPR005, TMPR105 | |
| INTP6 | PIF6 | | PMK6 | | PPR06, PPR16 | |
| INTP7 | PIF7 | | PMK7 | | PPR07, PPR17 | |
| INTCMP0 | CMPIF0 | | CMPMK0 | | CMPPR00, CMPPR10 | |
| INTCMP1 | CMPIF1 | | CMPMK1 | | CMPPR01, CMPPR11 | |

**Notes 1.** 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 one of the interrupt sources INTSRE0 and INTTM01H is generated, bit 0 of the IF1L register is set to 1. Bit 0 of the MK1L, PR01L, and PR11L registers supports these two interrupt sources.

**2.** If one of the interrupt sources INTST1 and INTIIC10 is generated, bit 1 of the IF1L register is set to 1. Bit 1 of the MK1L, PR01L, and PR11L registers supports these two interrupt sources.

**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 one of the interrupt sources INTSRE1 and INTTM03H is generated, bit 3 of the IF1L register is set to 1. Bit 3 of the MK1L, PR01L, and PR11L registers supports these two interrupt sources.

**Table 23-2. Flags Corresponding to Interrupt Request Sources (3/3)**

| Interrupt Source | Interrupt Request Flag | | Interrupt Mask Flag | | Priority Specification Flag | |
|---|---|---|---|---|---|---|
| | | Register | | Register | | Register |
| INTTM06 | TMIF06 | IF2H | TMMK06 | MK2H | TMPR006, TMPR106 | PR02H, PR12H |
| INTTM07 | TMIF07 | | TMMK07 | | TMPR007, TMPR107 | |
| INTIT00 | ITIF00 | | ITMK00 | | ITPR000, ITPR100 | |
| INTIT01 | ITIF01 | | ITMK01 | | ITPR001, ITPR101 | |
| INTCR | CRIF | | CRMK | | CRPR0, CRPR1 | |
| INTOSDC | OSDIF | | OSDMK | | OSDPR0, OSDPR1 | |
| INTIT10 | ITIF10 | IF3L | ITMK10 | MK3L | ITPR010, ITPR110 | PR03L, PR13L |
| INTIT11 | ITIF11 | | ITMK11 | | ITPR011, ITPR111 | |
| INTVBAT | VBAIF | | VBAMK | | VBAPR0, VBAPR1 | |

### 23.3.1 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 an instruction is executed upon acknowledgment of an interrupt request or upon reset signal generation.

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.

Reset signal generation clears these registers to 00H.

**Remark** If an instruction that writes data to this register is executed, the number of instruction execution clocks increases by 2 clocks.

**Figure 23-2. Format of Interrupt Request Flag Registers (IF0L, IF0H, IF1L, IF1H, IF2L, IF2H, IF3L) (1/2)**

Address: FFFE0H After reset: 00H 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

| Symbol | <7> | <6> | <5> | 4 | 3 | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| IF0H | SRIF0 | TMIF00 | STIF0<br>CSIIF00<br>IICIF00 | 0 | 0 | SREIF2 | SRIF2 | STIF2 |

Address: FFFE2H After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| IF1L | TMIF01 | FMIF | RTITIF | IICAIF0 | SREIF1<br>TMIF03H | SRIF1 | STIF1<br>IICIF10 | SREIF0<br>TMIF01H |

Address: FFFE3H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| IF1H | 0 | 0 | 0 | TMKAIF | RTCIF | ADIF | TMIF03 | TMIF02 |

Address: FFFD0H After reset: 00H R/W

| Symbol | <7> | <6> | 5 | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| IF2L | CMPIF1 | CMPIF0 | 0 | PIF7 | PIF6 | TMIF05 | TMIF04 | DSAIF |

Address: FFFD1H After reset: 00H R/W

| Symbol | 7 | <6> | 5 | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| IF2H | 0 | OSDIF | 0 | CRIF | ITIF01 | ITIF00 | TMIF07 | TMIF06 |

**Figure 23-2. Format of Interrupt Request Flag Registers (IF0L, IF0H, IF1L, IF1H, IF2L, IF2H, IF3L) (2/2)**

Address: FFFD2H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | <4> | <3> | <2> | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| IF3L | 0 | 0 | 0 | VBAIF | ITIF11 | ITIF10 | 0 | 0 |

| XXIFX | Interrupt request flag |
|---|---|
| 0 | No interrupt request signal is generated |
| 1 | Interrupt request is generated, interrupt request status |

**Cautions 1. For details about the bits, see Table 23-2. Be sure to clear bits that are not available to 0.**

**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 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.**

### 23.3.2 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.

Reset signal generation sets these registers to FFH.

**Remark** If an instruction that writes data to this register is executed, the number of instruction execution clocks increases by 2 clocks.

**Figure 23-3. Format of Interrupt Mask Flag Registers (MK0L, MK0H, MK1L, MK1H, MK2L, MK2H, MK3L)**

Address: FFFE4H After reset: FFH 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

| Symbol | <7> | <6> | <5> | 4 | 3 | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| MK0H | SRMK0 | TMMK00 | STMK0<br>CSIMK00<br>IICMK00 | 1 | 1 | SREMK2 | SRMK2 | STMK2 |

Address: FFFE6H After reset: FFH R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| MK1L | TMMK01 | FMMK | RTITMK | IICAMK0 | SREMK1<br>TMMK03H | SRMK1 | STMK1<br>IICMK10 | SREMK0<br>TMMK01H |

Address: FFFE7H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| MK1H | 1 | 1 | 1 | TMKAMK | RTCMK | ADMK | TMMK03 | TMMK02 |

Address: FFFD4H After reset: FFH R/W

| Symbol | <7> | <6> | 5 | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| MK2L | CMPMK1 | CMPMK0 | 1 | PMK7 | PMK6 | TMMK05 | TMMK04 | DSAMK |

Address: FFFD5H After reset: FFH R/W

| Symbol | 7 | <6> | 5 | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| MK2H | 1 | OSDMK | 1 | CRMK | ITMK01 | ITMK00 | TMMK07 | TMMK06 |

Address: FFFD6H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | <4> | <3> | <2> | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| MK3L | 1 | 1 | 1 | VBAMK | ITMK11 | ITMK10 | 1 | 1 |

| XXMKX | Interrupt servicing control |
|---|---|
| 0 | Interrupt servicing enabled |
| 1 | Interrupt servicing disabled |

**Caution For details about the bits, see Table 23-2. Be sure to set bits that are not available to the initial value.**

### 23.3.3 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, or 2H).

The PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR03L, PR10L, PR10H, PR11L, PR11H, PR12L, PR12H, and the 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.

Reset signal generation sets these registers to FFH.

**Remark** If an instruction that writes data to this register is executed, the number of instruction execution clocks increases by 2 clocks.

**Figure 23-4. Format of Priority Specification Flag Registers (PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR03L, PR10L, PR10H, PR11L, PR11H, PR12L, PR12H, PR13L) (1/2)**

Address: FFFE8H After reset: FFH 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

| 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

| Symbol | <7> | <6> | <5> | 4 | 3 | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PR00H | SRPR00 | TMPR000 | STPR00<br>CSIPR000<br>IICPR000 | 1 | 1 | SREPR02 | SRPR02 | STPR02 |

Address: FFFEDH After reset: FFH R/W

| Symbol | <7> | <6> | <5> | 4 | 3 | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PR10H | SRPR10 | TMPR100 | STPR10<br>CSIPR100<br>IICPR100 | 1 | 1 | SREPR12 | SRPR12 | STPR12 |

Address: FFFEAH After reset: FFH R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PR01L | TMPR001 | FMPR0 | RTITPR0 | IICAPR00 | SREPR01<br>TMPR003H | SRPR01 | STPR01<br>IICPR010 | SREPR00<br>TMPR001H |

Address: FFFEEH After reset: FFH R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PR11L | TMPR101 | FMPR1 | RTITPR1 | IICAPR10 | SREPR11<br>TMPR103H | SRPR11 | STPR11<br>IICPR110 | SREPR10<br>TMPR101H |

**Figure 23-4. Format of Priority Specification Flag Registers (PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR03L, PR10L, PR10H, PR11L, PR11H, PR12L, PR12H, PR13L) (2/2)**

Address: FFFEBH After reset: FFH R/W

| Symbol | 7 | 6 | 5 | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PR01H | 1 | 1 | 1 | TMKAPR0 | RTCPR0 | ADPR0 | TMPR003 | TMPR002 |

Address: FFFEFH After reset: FFH R/W

| Symbol | 7 | 6 | 5 | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PR11H | 1 | 1 | 1 | TMKAPR1 | RTCPR1 | ADPR1 | TMPR103 | TMPR102 |

Address: FFFD8H After reset: FFH R/W

| Symbol | <7> | <6> | 5 | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PR02L | CMPPR01 | CMPPR00 | 1 | PPR07 | PPR06 | TMPR005 | TMPR004 | DSAPR0 |

Address: FFFDCH After reset: FFH R/W

| Symbol | <7> | <6> | 5 | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PR12L | CMPPR11 | CMPPR10 | 1 | PPR17 | PPR16 | TMPR105 | TMPR104 | DSAPR1 |

Address: FFFD9H After reset: FFH R/W

| Symbol | 7 | <6> | 5 | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PR02H | 1 | OSDPR0 | 1 | CRPR0 | ITPR001 | ITPR000 | TMPR007 | TMPR006 |

Address: FFFDDH After reset: FFH R/W

| Symbol | 7 | <6> | 5 | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PR12H | 1 | OSDPR1 | 1 | CRPR1 | ITPR101 | ITPR100 | TMPR107 | TMPR106 |

Address: FFFDAH After reset: FFH R/W

| Symbol | 7 | 6 | 5 | <4> | <3> | <2> | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PR03L | 1 | 1 | 1 | VBAPR0 | ITPR011 | ITPR010 | 1 | 1 |

Address: FFFDEH After reset: FFH R/W

| Symbol | 7 | 6 | 5 | <4> | <3> | <2> | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PR13L | 1 | 1 | 1 | VBAPR1 | ITPR111 | ITPR110 | 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 For details about the bits, see Table 23-2. Be sure to set bits that are not available to the initial value.**

### 23.3.4 External interrupt rising edge enable register (EGP0), external interrupt falling edge enable register (EGN0)

These registers specify the valid edge for INTP0 to INTP7.

The EGP0 and EGN0 registers can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H.

**Figure 23-5. Format of External Interrupt Rising Edge Enable Register (EGP0) and External Interrupt Falling Edge Enable Register (EGN0)**

Address: FFF38H After reset: 00H 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

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| EGN0 | EGN7 | EGN6 | EGN5 | EGN4 | EGN3 | EGN2 | EGN1 | EGN0 |

| EGPn | EGNn | INTPn pin valid edge selection (n = 0 to 7) |
|---|---|---|
| 0 | 0 | Edge detection disabled |
| 0 | 1 | Falling edge |
| 1 | 0 | Rising edge |
| 1 | 1 | Both rising and falling edges |

Table 23-3 shows the ports corresponding to the EGPn and EGNn bits.

**Table 23-3. Ports Corresponding to EGPn and EGNn bits**

| Detection Enable Bit | | Interrupt Request Signal |
|---|---|---|
| EGP0 | EGN0 | INTP0 |
| EGP1 | EGN1 | INTP1 |
| EGP2 | EGN2 | INTP2 |
| EGP3 | EGN3 | INTP3 |
| EGP4 | EGN4 | INTP4 |
| EGP5 | EGN5 | INTP5 |
| EGP6 | EGN6 | INTP6 |
| EGP7 | EGN7 | INTP7 |

**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 port mode register (PMxx) to 0 after disabling the edge detection (by setting EGPn and EGNn to 0).**

**Remarks 1.** For edge detection port, see **2.1 Port Function List.**

**2.** n = 0 to 7

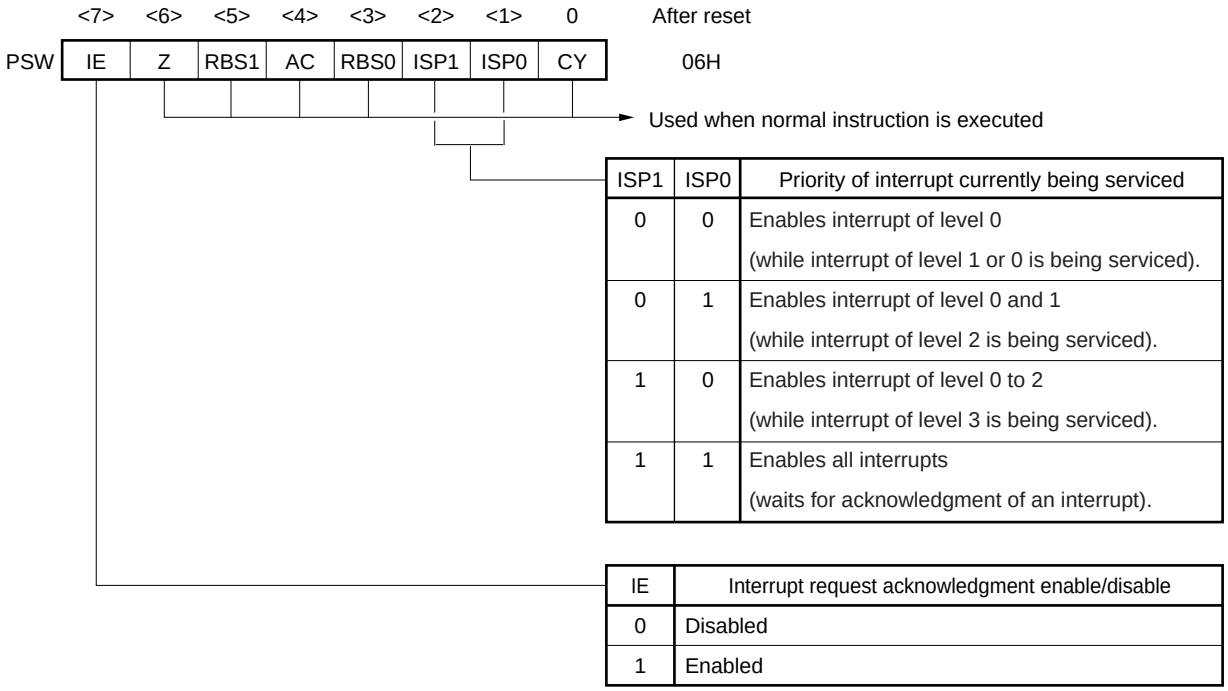
### 23.3.5 Program status word (PSW)

The program status word is a register used to hold the instruction execution result and the current status for an interrupt request. The IE flag that sets maskable interrupt enable/disable and the ISP0 and ISP1 flags that controls multiple interrupt servicing are mapped to the PSW.

Besides 8-bit read/write, this register can carry out operations using bit 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 ISP0 and ISP1 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.

Reset signal generation sets PSW to 06H.

**Figure 23-6. Configuration of Program Status Word**




| ISP1 | ISP0 | Priority of interrupt currently being serviced |
|---|---|---|
| 0 | 0 | Enables interrupt of level 0<br>(while interrupt of level 1 or 0 is being serviced). |
| 0 | 1 | Enables interrupt of level 0 and 1<br>(while interrupt of level 2 is being serviced). |
| 1 | 0 | Enables interrupt of level 0 to 2<br>(while interrupt of level 3 is being serviced). |
| 1 | 1 | Enables all interrupts<br>(waits for acknowledgment of an interrupt). |

| IE | Interrupt request acknowledgment enable/disable |
|---|---|
| 0 | Disabled |
| 1 | Enabled |

## 23.4 Interrupt Servicing Operations

### 23.4.1 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.

The times from generation of a maskable interrupt request until vectored interrupt servicing is performed are listed in Table 23-4 below.

For the interrupt request acknowledgment timing, see **Figures 23-8** and **23-9**.

**Table 23-4. Time from Generation of Maskable Interrupt Until Servicing**

| | Minimum Time | Maximum Time[Note] |
|---|---|---|
| Servicing time | 9 clocks | 16 clocks |

**Note** Maximum time does not apply when an instruction from the internal RAM area is executed.

**Remark** 1 clock: $1/f_{CLK}$ ($f_{CLK}$: 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 interrupts requests have the same priority level, the request with the highest default priority is acknowledged first.

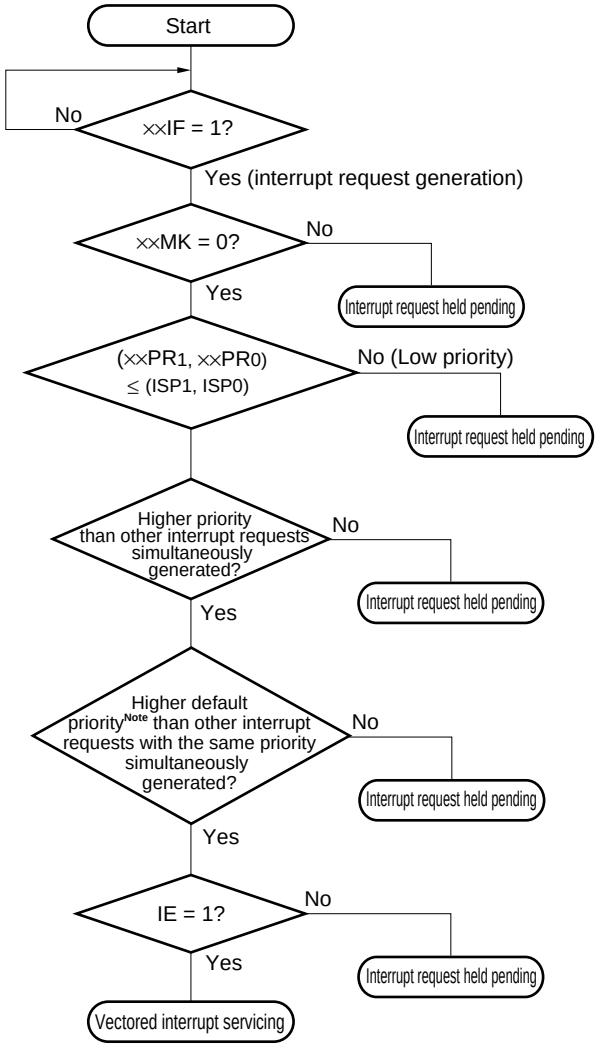
An interrupt request that is held pending is acknowledged when it becomes acknowledgeable.

Figure 23-7 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 (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 the loaded into the PC and branched.

Restoring from an interrupt is possible by using the RETI instruction.

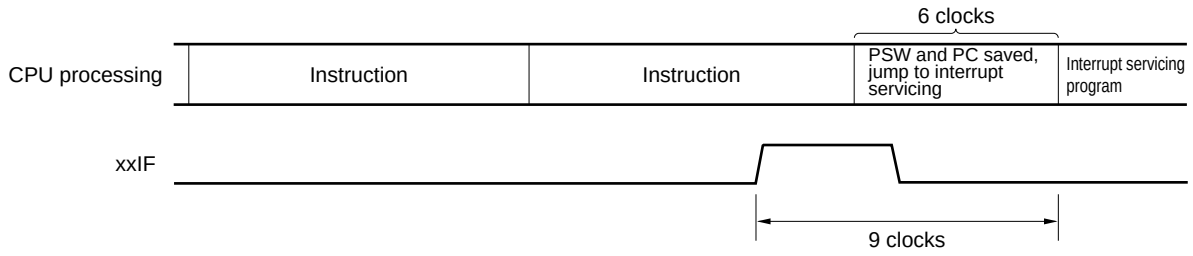
**Figure 23-7. Interrupt Request Acknowledgment Processing Algorithm**




××IF: Interrupt request flag
××MK: Interrupt mask flag
××PR0: Priority specification flag 0
××PR1: Priority specification flag 1
IE: Flag that controls acknowledgment of maskable interrupt request (1 = Enable, 0 = Disable)
ISP0, ISP1: Flag that indicates the priority level of the interrupt currently being serviced (see **Figure 23-6**)

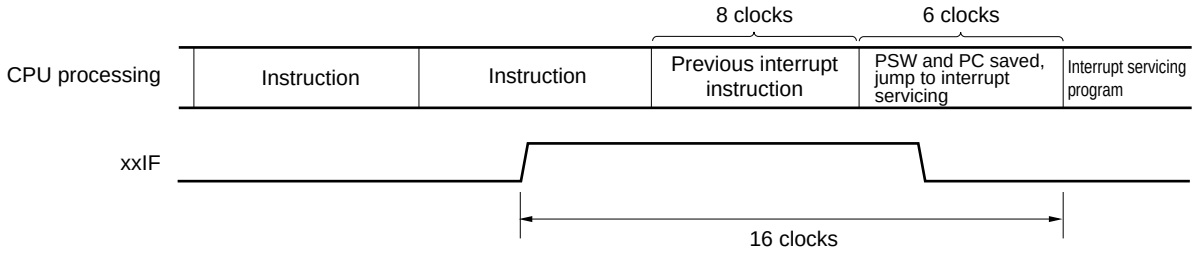
**Note** For the default priority, see **Table 23-1 Interrupt Source List**.

**Figure 23-8. Interrupt Request Acknowledgment Timing (Minimum Time)**




**Remark** 1 clock: 1/$f_{CLK}$ ($f_{CLK}$: CPU clock)

**Figure 23-9. Interrupt Request Acknowledgment Timing (Maximum Time)**




**Remark** 1 clock: 1/$f_{CLK}$ ($f_{CLK}$: CPU clock)

### 23.4.2 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 (0), and the contents of the vector table (0007EH, 0007FH) are loaded into the PC and branched.

Restoring from a software interrupt is possible by using the RETB instruction.

**Caution Can not use the RETI instruction for restoring from the software interrupt.**

### 23.4.3 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 (1) the IE flag 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 equal to or higher than that of the interrupt currently being serviced is generated, it is acknowledged for multiple interrupt servicing. If an interrupt with a priority lower than that of the interrupt currently being serviced is generated during interrupt servicing, it is not acknowledged for multiple interrupt servicing. 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 execution.

Table 23-5 shows relationship between interrupt requests enabled for multiple interrupt servicing and Figure 23-10 shows multiple interrupt servicing examples.

**Table 23-5. Relationship Between Interrupt Requests Enabled for Multiple Interrupt Servicing During Interrupt Servicing**

| Multiple Interrupt Request / Interrupt Being Serviced | | 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 | |
| Maskable interrupt | ISP1 = 0 ISP0 = 0 | O | × | × | × | × | × | × | × | O |
| | ISP1 = 0 ISP0 = 1 | O | × | O | × | × | × | × | × | O |
| | ISP1 = 1 ISP0 = 0 | O | × | O | × | O | × | × | × | O |
| | ISP1 = 1 ISP0 = 1 | O | × | O | × | O | × | O | × | O |
| Software interrupt | | O | × | O | × | O | × | O | × | O |

**Remarks 1.** O: Multiple interrupt servicing enabled

**2.** ×: Multiple interrupt servicing disabled

**3.** ISP0, ISP1, and IE are flags contained in the PSW.

ISP1 = 0, ISP0 = 0: An interrupt of level 1 or level 0 is being serviced.

ISP1 = 0, ISP0 = 1: An interrupt of level 2 is being serviced.

ISP1 = 1, ISP0 = 0: An interrupt of level 3 is being serviced.

ISP1 = 1, ISP0 = 1: Wait for an interrupt acknowledgment (all interrupts are enabled).

IE = 0: Interrupt request acknowledgment is disabled.

IE = 1: Interrupt request acknowledgment is enabled.

**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 ××PR1× = 0, ××PR0× = 0 (higher priority level)

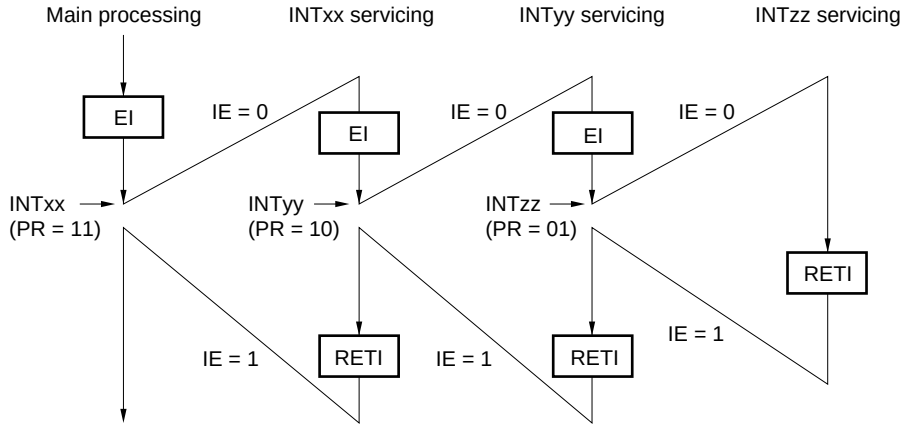
PR = 01: Specify level 1 with ××PR1× = 0, ××PR0× = 1

PR = 10: Specify level 2 with ××PR1× = 1, ××PR0× = 0

PR = 11: Specify level 3 with ××PR1× = 1, ××PR0× = 1 (lower priority level)

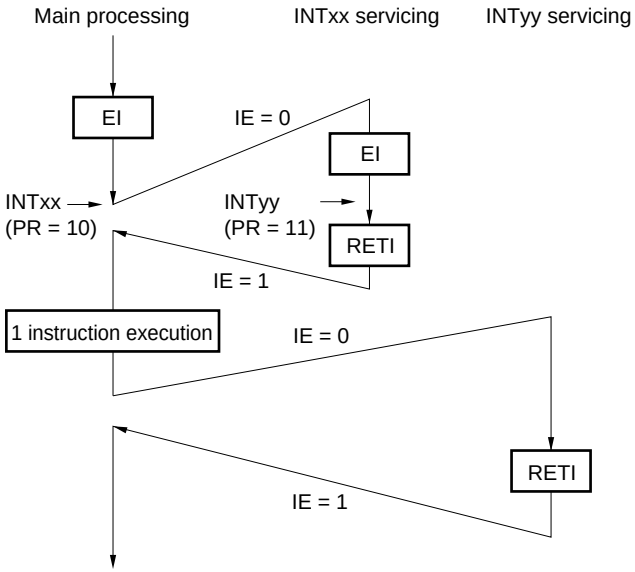
**Figure 23-10. Examples of Multiple Interrupt Servicing (1/2)**

**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**

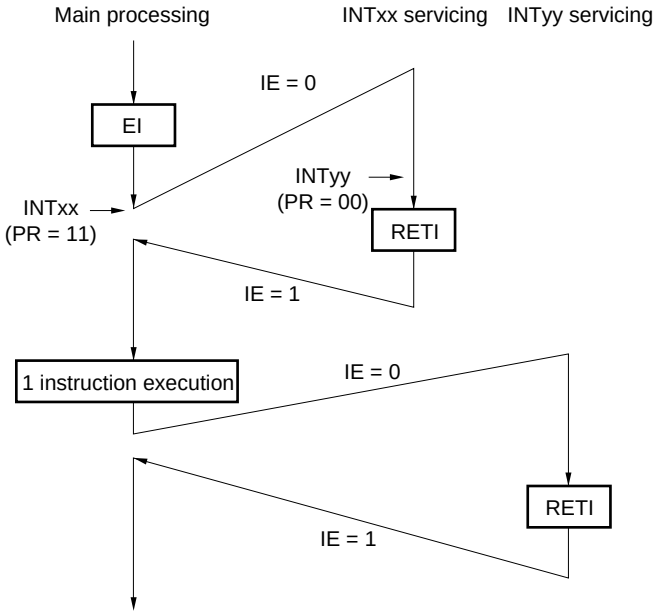



Interrupt request INTyy issued during servicing of interrupt INTxx is not acknowledged because its priority is lower than that of INTxx, and multiple interrupt servicing does not take place. The INTyy interrupt request is held pending, and is acknowledged following execution of one main processing instruction.

PR = 00: Specify level 0 with ××PR1× = 0, ××PR0× = 0 (higher priority level)
PR = 01: Specify level 1 with ××PR1× = 0, ××PR0× = 1
PR = 10: Specify level 2 with ××PR1× = 1, ××PR0× = 0
PR = 11: Specify level 3 with ××PR1× = 1, ××PR0× = 1 (lower priority level)
IE = 0: Interrupt request acknowledgment is disabled
IE = 1: Interrupt request acknowledgment is enabled.

**Figure 23-10. 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.

PR = 00: Specify level 0 with ××PR1× = 0, ××PR0× = 0 (higher priority level)
PR = 01: Specify level 1 with ××PR1× = 0, ××PR0× = 1
PR = 10: Specify level 2 with ××PR1× = 1, ××PR0× = 0
PR = 11: Specify level 3 with ××PR1× = 1, ××PR0× = 1 (lower priority level)
IE = 0: Interrupt request acknowledgment is disabled
IE = 1: Interrupt request acknowledgment is enabled.

<R> **23.4.4 Interrupt servicing during division instruction**

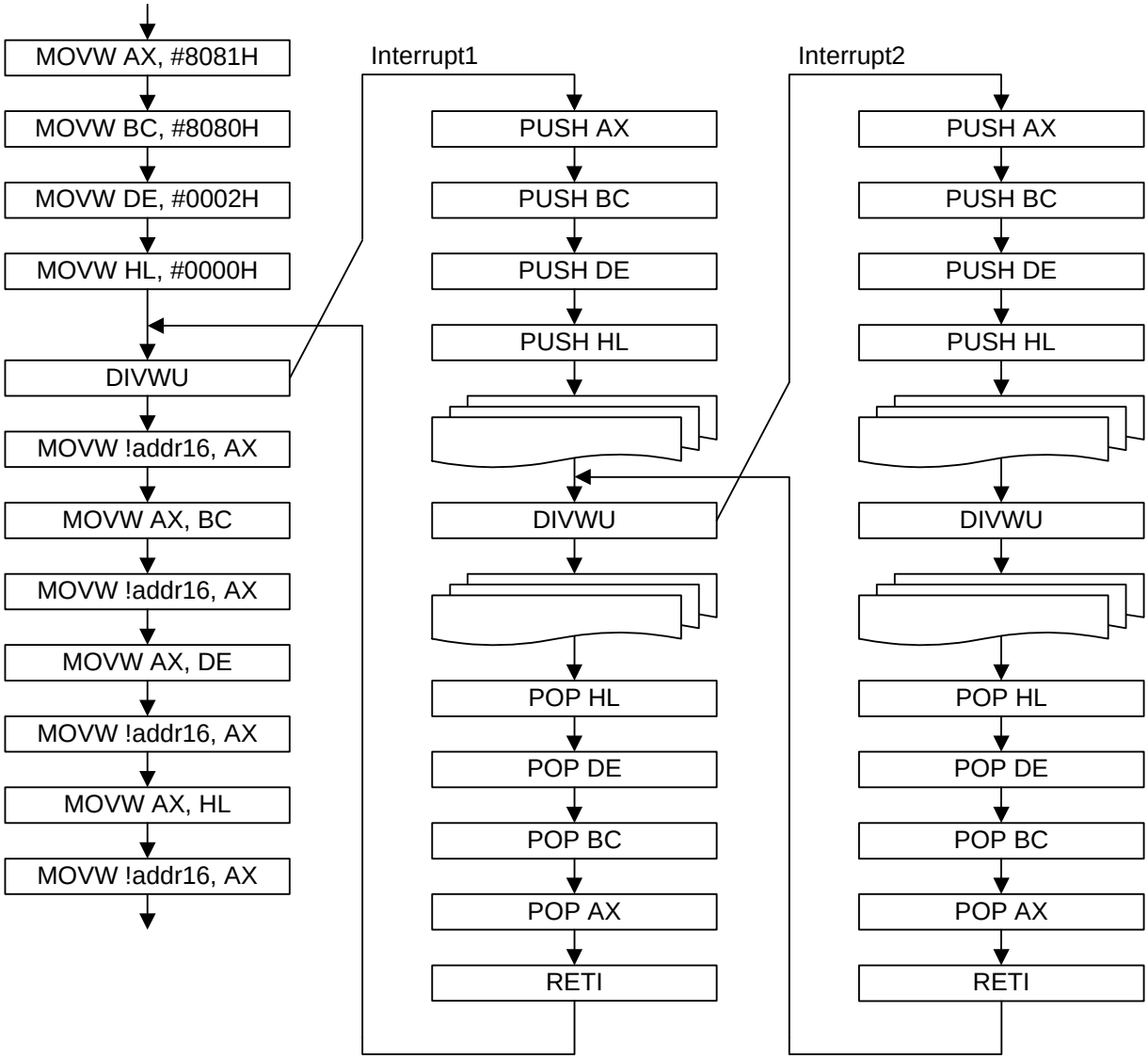
The RL78/I1B handles interrupts during the DIVHU/DIVWU instruction in order to enhance the interrupt response when a division instruction is executed.

- When an interrupt is generated while the DIVHU/DIVWU instruction is executed, the instruction is suspended
- After the instruction is suspended, the PC indicates the next instruction after DIVHU/DIVWU
- An interrupt is generated by the next instruction
- PC-3 is stacked to execute the DIVHU/DIVWU instruction again

| Normal interrupt | Interrupts while Executing DIVHU/DIVWU Instruction |
|---|---|
| (SP-1) ← PSW | (SP-1) ← PSW |
| (SP-2) ← (PC)S | (SP-2) ← (PC-3)S |
| (SP-3) ← (PC)H | (SP-3) ← (PC-3)H |
| (SP-4) ← (PC)L | (SP-4) ← (PC-3)L |
| PCS ← 0000 | PCS ← 0000 |
| PCH ← (Vector) | PCH ← (Vector) |
| PCL ← (Vector) | PCL ← (Vector) |
| SP ← SP-4 | SP ← SP-4 |
| IE ← 0 | IE ← 0 |

The AX, BC, DE, and HL registers are used for DIVHU/DIVWU. Use these registers by stacking them for interrupt servicing.

<R>




**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. 1.71 and later versions of the CA78K0R (Renesas Electronics compiler), for both C and assembly language source code**
- **Service pack 1.40.6 and later versions of the EWRL78 (IAR compiler), for C language source code**
- **GNURL78 (KPIT compiler), for C language source code**

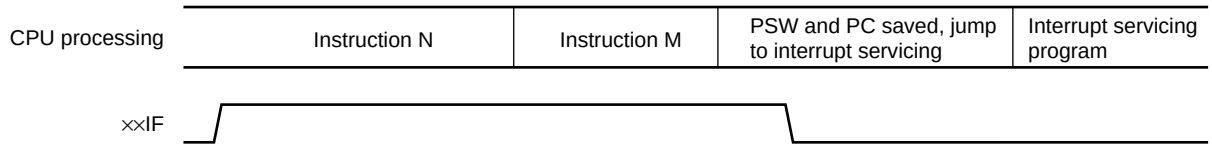
### 23.4.5 Interrupt request hold

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 (interrupt request hold instructions) 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
- <R> MULHU
- <R> MULH
- <R> MACHU
- <R> MACH
- 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 23-11 shows the timing at which interrupt requests are held pending.

**Figure 23-11. Interrupt Request Hold**




**Remarks 1.** Instruction N: Interrupt request hold instruction

**2.** Instruction M: Instruction other than interrupt request hold instruction

# CHAPTER 24 STANDBY FUNCTION

## 24.1 Standby Function

The standby function reduces the operating current of the system, and the following three modes are available.

**(1) HALT mode**

HALT instruction execution sets the HALT mode. In the HALT mode, the CPU operation 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**

STOP instruction execution sets 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 cleared by an interrupt request, it enables intermittent operations to be carried out. 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 the case of CSI00 or UART0 data reception, an A/D conversion request by the timer trigger signal (the interrupt request signal (INTRTC/INTIT)), and DTC start source, the STOP mode is exited, the CSI00 or UART0 data is received without operating the CPU, A/D conversion, and DTC conversion is performed. This can only be specified when the high-speed on-chip oscillator is selected for the CPU/peripheral hardware clock ($f_{CLK}$).

In all these modes, all the contents of registers, flags and data memory just before the standby mode is set are held. The I/O port output latches and output buffer statuses are also held.

**Cautions**

1. **The STOP mode can be used only when the CPU is operating on the main system clock. Do not set to the STOP mode while the CPU operates with the subsystem clock. The HALT mode can be used when the CPU is operating on either the main system clock or the subsystem clock.**
2. **When shifting to the STOP mode, be sure to stop the peripheral hardware operation operating with main system clock before executing STOP instruction (except SNOOZE mode setting unit).**
3. **When using CSI00, UART0, or the A/D converter in the SNOOZE mode, set up serial standby control register 0 (SSC0) and A/D converter mode register 2 (ADM2) before switching to the STOP mode. For details, see 18.3 Registers Controlling Serial Array Unit and 14.3 Registers Controlling A/D Converter.**
4. **The following sequence is recommended for power consumption reduction of the A/D converter when the standby function is used: First clear bit 7 (ADCS) and bit 0 (ADCE) of A/D converter mode register 0 (ADM0) to 0 to stop the A/D conversion operation, and then execute the STOP instruction.**
5. **It can be selected by the option byte whether the low-speed on-chip oscillator continues oscillating or stops in the HALT or STOP mode. For details, see CHAPTER 32 OPTION BYTE.**

## 24.2 Registers Controlling Standby Function

The registers which control the standby function are described below.

- Subsystem clock supply mode control register (OSMC)
- Oscillation stabilization time counter status register (OSTC)
- Oscillation stabilization time select register (OSTS)

**Remark** For details of registers described above, see **CHAPTER 5 CLOCK GENERATOR**. For registers which control the SNOOZE mode, **CHAPTER 14 A/D CONVERTER** and **CHAPTER 18 SERIAL ARRAY UNIT**.

## 24.3 Standby Function Operation

### 24.3.1 HALT mode

**(1) 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.

The operating statuses in the HALT mode are shown below.

**Caution Because the interrupt request signal is used to clear the HALT mode, if the interrupt mask flag is 0 (the interrupt processing is enabled) and the interrupt request flag is 1 (the interrupt request signal is generated), the HALT mode is not entered even if the HALT instruction is executed in such a situation.**

**Table 24-1. Operating Statuses in HALT Mode (1/2)**

| HALT Mode Setting / Item | | | When HALT Instruction Is Executed While CPU Is Operating on Main System Clock | | |
|---|---|---|---|---|---|
| | | | When CPU Is Operating on High-speed On-chip Oscillator Clock ($f_{IH}$) | When CPU Is Operating on X1 Clock ($f_X$) | When CPU Is Operating on External Main System Clock ($f_{EX}$) |
| System clock | | | Clock supply to the CPU is stopped | | |
| | Main system clock | $f_{IH}$ | Operation continues (cannot be stopped) | Operation disabled | |
| | | $f_X$ | Operation disabled | Operation continues (cannot be stopped) | Cannot operate |
| | | $f_{EX}$ | | Cannot operate | Operation continues (cannot be stopped) |
| | Subsystem clock | $f_{XT}$ | Status before HALT mode was set is retained | | |
| | | $f_{EXS}$ | | | |
| | $f_{IL}$ | | Set by bits 0 (WDSTBYON) and 4 (WDTON) of option byte (000C0H), and WUTMMCK0 bit of subsystem clock supply mode control register (OSMC)<br>• WUTMMCK0 = 1: Oscillates<br>• WUTMMCK0 = 0 and WDTON = 0: Stops<br>• WUTMMCK0 = 0, WDTON = 1, and WDSTBYON = 1: Oscillates<br>• WUTMMCK0 = 0, WDTON = 1, and WDSTBYON = 0: Stops | | |
| CPU | | | Operation stopped | | |
| Code flash memory | | | | | |
| <R> RAM | | | Operation stopped (Operable while in the DTC is executed) | | |
| Port (latch) | | | Status before HALT mode was set is retained | | |
| Timer array unit | | | Operable | | |
| Real-time clock 2 | | | Operable | Operable (High accuracy 1 Hz output mode is operation disabled.) | |
| Subsystem clock frequency measurement circuit | | | Operation disabled | Operable | |
| High-speed on-chip oscillator clock frequency correction function | | | Operable (when $f_{XT}$ or $f_{EXS}$ is supplied) | Operation disabled | |
| Oscillation stop detection | | | Operable (only when $f_{IL}$ is oscillating) | | |
| Battery backup function | | | Operable (when VBATEN = 1 and VBATSEL = 0) | | |
| 12-bit interval timer | | | Operable | | |
| 8-bit interval timer | | | | | |
| Watchdog timer | | | Operable (See **CHAPTER 13 WATCHDOG TIMER**) | | |
| Clock output/buzzer output | | | Operable | | |
| A/D converter | | | | | |
| ΔΣ A/D Converter | | | | | |
| Temperature sensor 2 | | | | | |
| Comparator | | | | | |
| Serial array unit (SAU) | | | | | |
| IrDA | | | | | |
| Serial interface (IICA) | | | | | |
| LCD controller/driver | | | Operable (However, this depends on the status of the clock selected as the LCD source clock: operation is possible if the selected clock is operating, but operation will stop if the selected clock is stopped.) | | |
| Data transfer controller (DTC) | | | Operable | | |
| Power-on-reset function | | | | | |
| Voltage detection function | | | | | |
| External interrupt | | | | | |
| CRC operation function | High-speed CRC | | | | |
| | General-purpose CRC | | | | |
| RAM parity error detection function | | | Operation stopped (Operable when DTC is executed only) | | |
| RAM guard function | | | | | |
| SFR guard function | | | | | |
| Illegal-memory access detection function | | | | | |

(**Remark** is listed on the next page.)

**Remark** Operation stopped: Operation is automatically stopped before switching to the HALT mode.
Operation disabled: Operation is stopped before switching to the HALT mode.
$f_{IH}$: High-speed on-chip oscillator clock
$f_{IL}$: Low-speed on-chip oscillator clock
$f_X$: X1 clock
$f_{EX}$: External main system clock
$f_{XT}$: XT1 clock
$f_{EXS}$: External subsystem clock

**Table 24-1. Operating Statuses in HALT Mode (2/2)**

| HALT Mode Setting / Item | | | When HALT Instruction Is Executed While CPU Is Operating on Subsystem Clock | |
|---|---|---|---|---|
| | | | When CPU Is Operating on XT1 Clock ($f_{XT}$) | When CPU Is Operating on External Subsystem Clock ($f_{EXS}$) |
| System clock | | | Clock supply to the CPU is stopped | |
| | Main system clock | $f_{IH}$ | Operation disabled | |
| | | $f_X$ | | |
| | | $f_{EX}$ | | |
| | Subsystem clock | $f_{XT}$ | Operation continues (cannot be stopped) | Cannot operate |
| | | $f_{EXS}$ | Cannot operate | Operation continues (cannot be stopped) |
| | $f_{IL}$ | | Set by bits 0 (WDSTBYON) and 4 (WDTON) of option byte (000C0H), and WUTMMCK0 bit of subsystem clock supply mode control register (OSMC) (However, WUTMMCK0 cannot be set to 1 while the CPU is operating with subsystem clock)<br>• WUTMMCK0 = 0 and WDTON = 0: Stops<br>• WUTMMCK0 = 0, WDTON = 1, and WDSTBYON = 1: Oscillates<br>• WUTMMCK0 = 0, WDTON = 1, and WDSTBYON = 0: Stops | |
| CPU | | | Operation stopped | |
| Code flash memory | | | | |
| <R> RAM | | | Operation stopped (Operable while in the DTC is executed) | |
| Port (latch) | | | Status before HALT mode was set is retained | |
| Timer array unit | | | Operable when the RTCLPC bit is 0 (operation is disabled when the RTCLPC bit is not 0). | |
| Real-time clock 2 | | | Operable (Operation in high-accuracy 1 Hz output mode is disabled.) | |
| Subsystem clock frequency measurement circuit | | | Operation disabled | |
| High-speed on-chip oscillator clock frequency correction function | | | | |
| Oscillation stop detection | | | | |
| Battery backup function | | | Operable (when VBATEN = 1 and VBATSEL = 0) | |
| 12-bit interval timer | | | Operable | |
| 8-bit interval timer | | | | |
| Watchdog timer | | | Operable (See **CHAPTER 13 WATCHDOG TIMER**) | |
| Clock output/buzzer output | | | Operable | |
| A/D converter | | | Operation disabled | |
| ΔΣ A/D Converter | | | | |
| Temperature sensor 2 | | | | |
| Comparator | | | Operable when external input (IVREFn) is selected for comparator reference voltage. | |
| Serial array unit (SAU) | | | Operable when the RTCLPC bit is 0 (operation is disabled when the RTCLPC bit is not 0). | |
| IrDA | | | Operation disabled | |
| Serial interface (IICA) | | | Operation disabled | |
| LCD controller/driver | | | Operable (However, this depends on the status of the clock selected as the LCD source clock: operation is possible if the selected clock is operating, but operation will stop if the selected clock is stopped.) | |
| Data transfer controller (DTC) | | | Operable when the RTCLPC bit is 0 (operation is disabled when the RTCLPC bit is not 0). | |
| Power-on-reset function | | | Operable | |
| Voltage detection function | | | | |
| External interrupt | | | | |
| CRC operation function | High-speed CRC | | Operation disabled | |
| | General-purpose CRC | | In the calculation of the RAM area, operable when DTC is executed | |
| RAM parity error detection function | | | Operation stopped (Operable when DTC is executed only) | |
| RAM guard function | | | | |
| SFR guard function | | | | |
| Illegal-memory access detection function | | | | |

(**Remark** is listed on the next page.)

**Remark** Operation stopped: Operation is automatically stopped before switching to the HALT mode.
Operation disabled: Operation is stopped before switching to the HALT mode.
$f_{IH}$: High-speed on-chip oscillator clock $f_{EX}$: External main system clock
$f_{IL}$: Low-speed on-chip oscillator clock $f_{XT}$: XT1 clock
$f_X$: X1 clock $f_{EXS}$: External subsystem clock

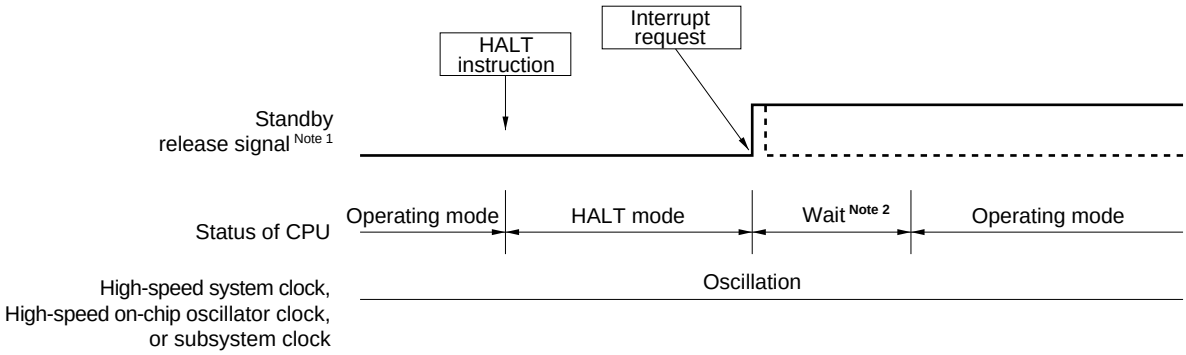
### (2) HALT mode release

The HALT mode can be released by the following two sources.

#### (a) Release by unmasked interrupt request

When an unmasked interrupt request is generated, the HALT mode is released. If interrupt acknowledgment is enabled, vectored interrupt servicing is carried out. If interrupt acknowledgment is disabled, the next address instruction is executed.

**Figure 24-1. HALT Mode Release by Interrupt Request Generation**




**Notes 1.** For details of the standby release signal, see **Figure 23-1**

**2.** Wait time for HALT mode release

- When vectored interrupt servicing is carried out
  Main system clock: 15 to 16 clock
  Subsystem clock (RTCLPC = 0): 10 to 11 clock
  Subsystem clock (RTCLPC = 1): 11 to 12 clock
- When vectored interrupt servicing is not carried out
  Main system clock: 9 to 10 clock
  Subsystem clock (RTCLPC = 0): 4 to 5 clock
  Subsystem clock (RTCLPC = 1): 5 to 6 clock

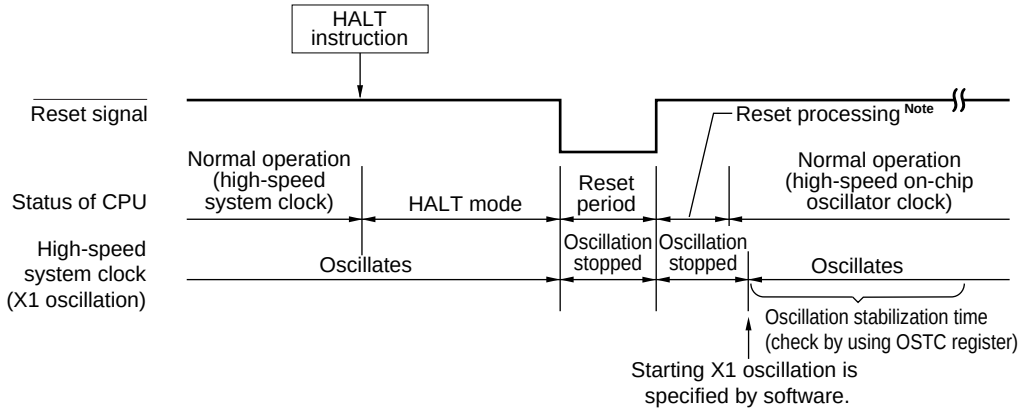
**Remark** The broken lines indicate the case when the interrupt request which has released the standby mode is acknowledged.

**(b) Release by reset signal generation**

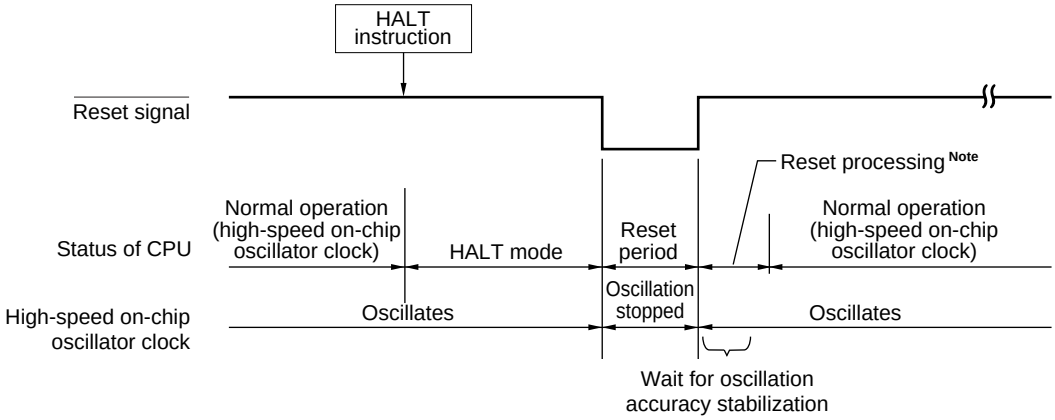
When the reset signal is generated, HALT mode is released, and then, as in the case with a normal reset operation, the program is executed after branching to the reset vector address.

**Figure 24-2. HALT Mode Release by Reset**

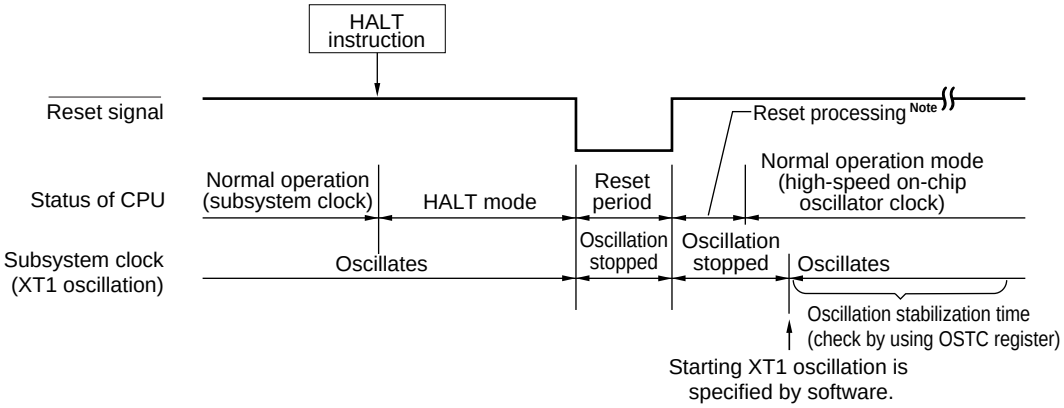
**(1) When high-speed system clock is used as CPU clock**




**(2) When high-speed on-chip oscillator clock is used as CPU clock**




**(3) When subsystem clock is used as CPU clock**




**Note** For the reset processing time, see **CHAPTER 25 RESET FUNCTION**.

For the reset processing time of the power-on-reset circuit (POR) and voltage detector (LVD), see **CHAPTER 26 POWER-ON-RESET CIRCUIT**.

### 24.3.2 STOP mode

**(1) STOP mode setting and operating statuses**

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 high-speed on-chip oscillator clock, X1 clock, or external main system clock.

**Caution Because the interrupt request signal is used to clear the STOP mode, if the interrupt mask flag is 0 (the interrupt processing is enabled) and the interrupt request flag is 1 (the interrupt request signal is generated), the STOP mode is immediately cleared if set when the STOP instruction is executed in such a situation. 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.**

The operating statuses in the STOP mode are shown below.

**Table 24-2. Operating Statuses in STOP Mode**

| STOP Mode Setting / Item | | | When STOP Instruction Is Executed While CPU Is Operating on Main System Clock | | |
|---|---|---|---|---|---|
| | | | When CPU Is Operating on High-speed on-chip oscillator clock ($f_{IH}$) | When CPU Is Operating on X1 Clock ($f_X$) | When CPU Is Operating on External Main System Clock ($f_{EX}$) |
| System clock | | | Clock supply to the CPU is stopped | | |
| | Main system clock | $f_{IH}$ | Stopped | | |
| | | $f_X$ | | | |
| | | $f_{EX}$ | | | |
| | Subsystem clock | $f_{XT}$ | Status before STOP mode was set is retained | | |
| | | $f_{EXS}$ | | | |
| | $f_{IL}$ | | Set by bits 0 (WDSTBYON) and 4 (WDTON) of option byte (000C0H), and WUTMMCK0 bit of subsystem clock supply mode control register (OSMC)<br>• WUTMMCK0 = 1: Oscillates<br>• WUTMMCK0 = 0 and WDTON = 0: Stops<br>• WUTMMCK0 = 0, WDTON = 1, and WDSTBYON = 1: Oscillates<br>• WUTMMCK0 = 0, WDTON = 1, and WDSTBYON = 0: Stops | | |
| CPU | | | Operation stopped | | |
| Code flash memory | | | | | |
| RAM | | | | | |
| Port (latch) | | | Status before STOP mode was set is retained | | |
| Timer array unit | | | Operation disabled | | |
| Real-time clock 2 | | | Operable (Operation in high-accuracy 1 Hz output mode is disabled.) | | |
| Subsystem clock frequency measurement circuit | | | Operation disabled | | |
| High-speed on-chip oscillator clock frequency correction function | | | | | |
| Oscillation stop detection | | | Operable (only when $f_{IL}$ is oscillating) | | |
| Battery backup function | | | Operable (when VBATEN = 1 and VBATSEL = 0) | | |
| 12-bit interval timer | | | Operable | | |
| 8-bit interval timer | | | | | |
| Watchdog timer | | | Operable (See **CHAPTER 13 WATCHDOG TIMER**) | | |
| Clock output/buzzer output | | | Operable only when subsystem clock is selected as the count clock (when low-consumption RTC mode (set RTCLPC bit of OSMC register to 1), operation disabled) | | |
| A/D converter | | | Wakeup operation is enabled (switching to the SNOOZE mode) | | |
| ΔΣA/D Converter | | | Operation disabled | | |
| Temperature sensor 2 | | | | | |
| Comparator | | | Operable (when digital filter is not used and external input (IVREFn) is selected for comparator reference voltage) | | |
| Serial array unit (SAU) | | | Wakeup operation is enabled only for CSI00 and UART0 (switching to the SNOOZE mode)<br>Operation is disabled for anything other than CSI00 and UART0 | | |
| IrDA | | | Operation disabled | | |
| Serial interface (IICA) | | | Wakeup by address match operable | | |
| LCD controller/driver | | | Operable (However, this depends on the status of the clock selected as the LCD source clock: operation is possible if the selected clock is operating, but operation will stop if the selected clock is stopped.) | | |
| Data transfer controller (DTC) | | | DTC start source acknowledge operation is enabled (switching to the SNOOZE mode) | | |
| Power-on-reset function | | | Operable | | |
| Voltage detection function | | | | | |
| External interrupt | | | | | |
| CRC operation function | High-speed CRC | | Operation stopped | | |
| | General-purpose CRC | | | | |
| RAM parity error detection function | | | | | |
| RAM guard function | | | | | |
| SFR guard function | | | | | |
| Illegal-memory access detection function | | | | | |

(**Remark** is listed on the next page.)

**Remark** Operation stopped: Operation is automatically stopped before switching to the STOP mode.
Operation disabled: Operation is stopped before switching to the STOP mode.
$f_{IH}$: High-speed on-chip oscillator clock $f_{IL}$: Low-speed on-chip oscillator clock
$f_X$: X1 clock $f_{EX}$: External main system clock
$f_{XT}$: XT1 clock $f_{EXS}$: External subsystem clock

### (2) STOP mode release

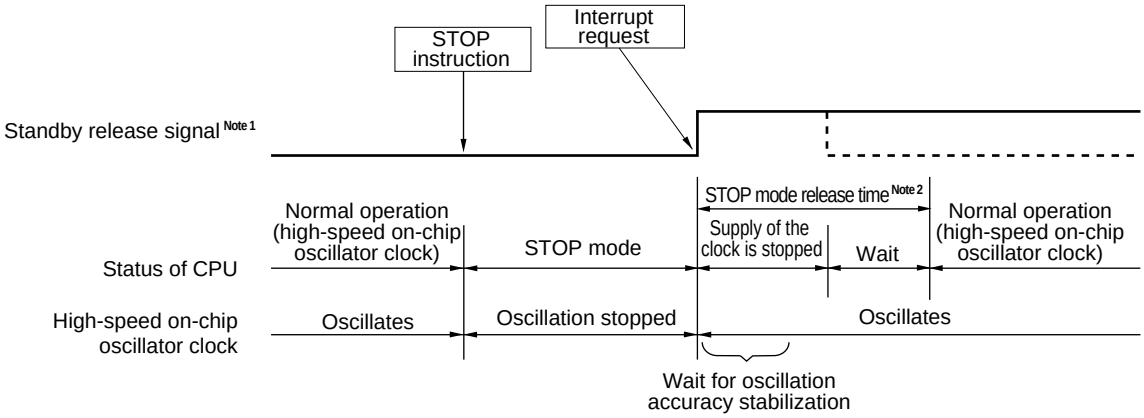
The STOP mode can be released by the following two sources.

#### (a) Release by unmasked interrupt request

When an unmasked interrupt request is generated, the STOP mode is released. After the oscillation stabilization time has elapsed, if interrupt acknowledgment is enabled, vectored interrupt servicing is carried out. If interrupt acknowledgment is disabled, the next address instruction is executed.

**Figure 24-3. STOP Mode Release by Interrupt Request Generation (1/2)**

**(1) When high-speed on-chip oscillator clock is used as CPU clock**




**Notes 1.** For details of the standby release signal, see **Figure 23-1**.

**2.** STOP mode release time

Supply of the clock is stopped: 18 μs to 65 μs

Wait

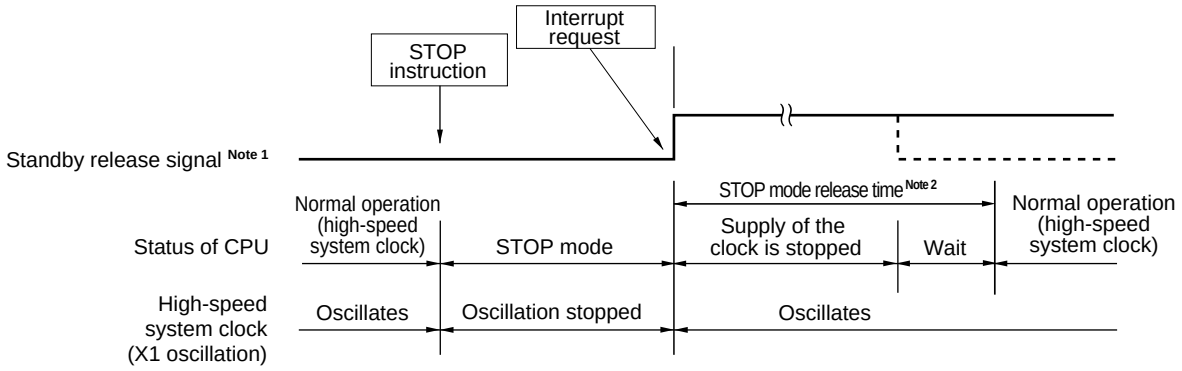
- When vectored interrupt servicing is carried out: 7 clocks
- When vectored interrupt servicing is not carried out: 1 clock

**Remarks 1.** The clock supply stop time varies depending on the temperature conditions and STOP mode period.

**2.** The broken lines indicate the case when the interrupt request that has released the standby mode is acknowledged.

**Figure 24-3. STOP Mode Release by Interrupt Request Generation (2/2)**

**(2) When high-speed system clock (X1 oscillation) is used as CPU clock**




**Notes 1.** For details of the standby release signal, see **Figure 23-1**.

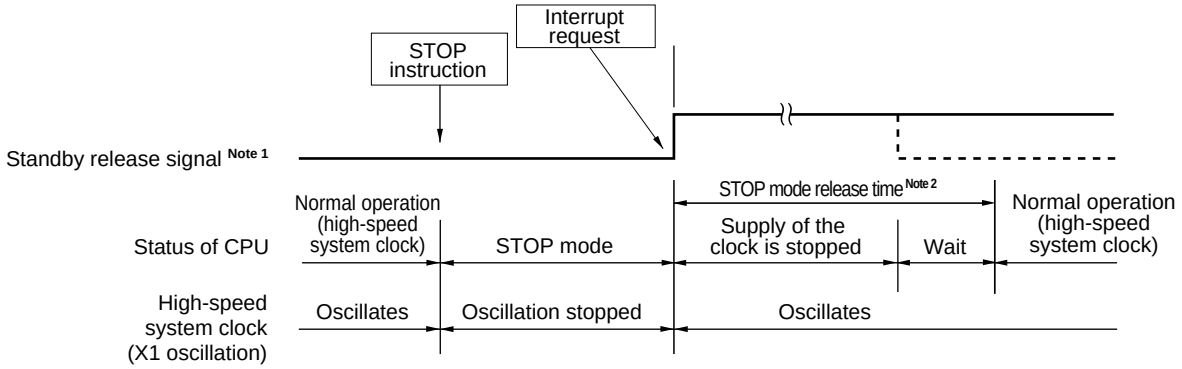
**2.** STOP mode release time

Supply of the clock is stopped: 18 μs to "whichever is longer 65 μs and the oscillation stabilization time (set by OSTS)"

Wait

- When vectored interrupt servicing is carried out: 10 to 11 clocks
- When vectored interrupt servicing is not carried out: 4 to 5 clocks

**(3) When high-speed system clock (external clock input) is used as CPU clock**




**Notes 1.** For details of the standby release signal, see **Figure 23-1**.

**2.** STOP mode release time

Supply of the clock is stopped: 18 μs to 65 μs

Wait

- When vectored interrupt servicing is carried out: 7 clocks
- When vectored interrupt servicing is not carried out: 1 clock

**Caution To reduce the oscillation stabilization time after release from the STOP mode while CPU operates based on the high-speed system clock (X1 oscillation), switch the clock to the high-speed on-chip oscillator clock temporarily before executing the STOP instruction.**

**Remarks 1.** The clock supply stop time varies depending on the temperature conditions and STOP mode period.

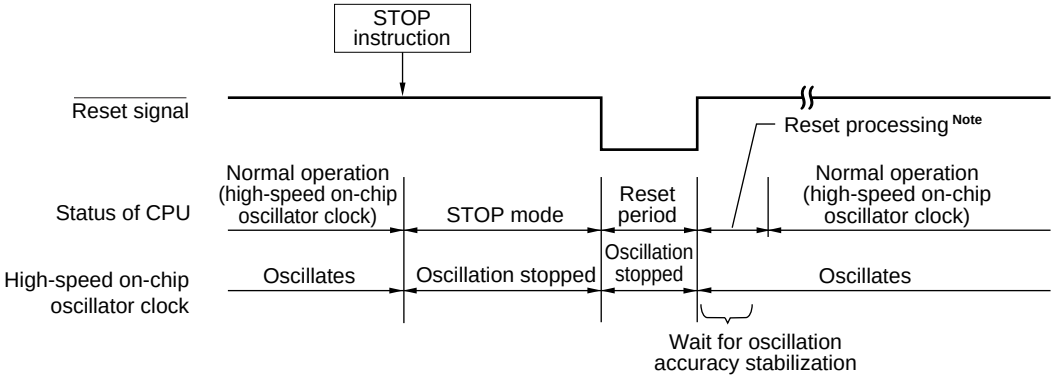
**2.** The broken lines indicate the case when the interrupt request that has released the standby mode is acknowledged.

**(b) Release by reset signal generation**

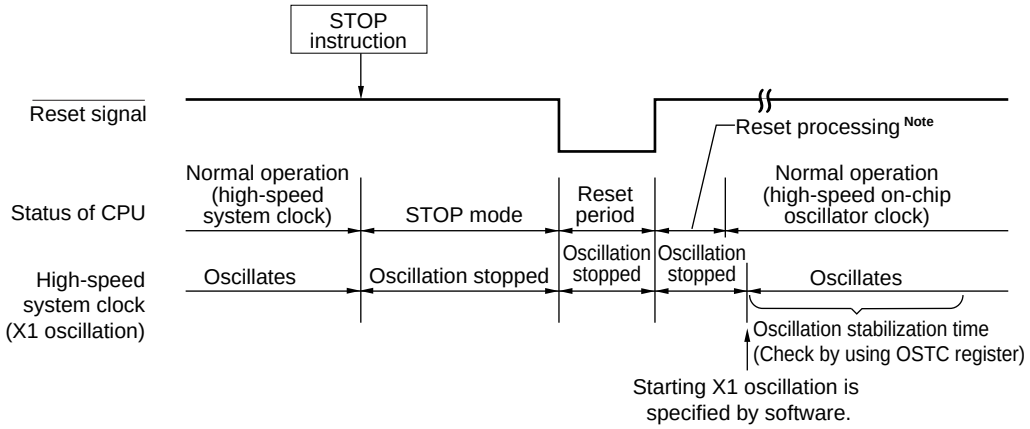
When the reset signal is generated, STOP mode is released, and then, as in the case with a normal reset operation, the program is executed after branching to the reset vector address.

**Figure 24-4. 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 **CHAPTER 25 RESET FUNCTION**.

For the reset processing time of the power-on-reset circuit (POR) and voltage detector (LVD), see **CHAPTER 26 POWER-ON-RESET CIRCUIT**.

### 24.3.3 SNOOZE mode

**(1) SNOOZE mode setting and operating statuses**

The SNOOZE mode can only be specified for CSI00, UART0, DTC, or the A/D converter. Note that this mode can only be specified if the CPU clock is the high-speed on-chip oscillator clock.

When using CSI00 or UART0 in the SNOOZE mode, set the SWCm bit of serial standby control register m (SSCm) to 1 immediately before switching to the STOP mode. For details, see **18.3 Registers Controlling Serial Array Unit**.

When using the A/D converter in the SNOOZE mode, set the AWC bit of A/D converter mode register 2 (ADM2) to 1 immediately before switching to the STOP mode. For details, see **14.3 Registers Controlling A/D Converter**.

When DTC transfer is used in SNOOZE mode, before switching to the STOP mode, allow DTC activation by interrupt to be used. During STOP mode, detecting DTC activation by interrupt enables DTC transit to SNOOZE mode, automatically. For details, see **22.3 Registers Controlling DTC**.

In SNOOZE mode transition, wait status to be only following time.

Transition time from STOP mode to SNOOZE mode: 18 μs to 65 μs

**Remark** Transition time from STOP mode to SNOOZE mode varies depending on the temperature conditions and the STOP mode period.

Transition time from SNOOZE mode to normal operation:

- When vectored interrupt servicing is carried out:
  HS (High-speed main) mode : “4.99 μs to 9.44 μs” + 7 clocks
  LS (Low-speed main) mode : “1.10 μs to 5.08 μs” + 7 clocks
- When vectored interrupt servicing is not carried out:
  HS (High-speed main) mode : “4.99 μs to 9.44 μs” + 1 clock
  LS (Low-speed main) mode : “1.10 μs to 5.08 μs” + 1 clock

The operating statuses in the SNOOZE mode are shown below.

**Table 24-3. Operating Statuses in SNOOZE Mode**

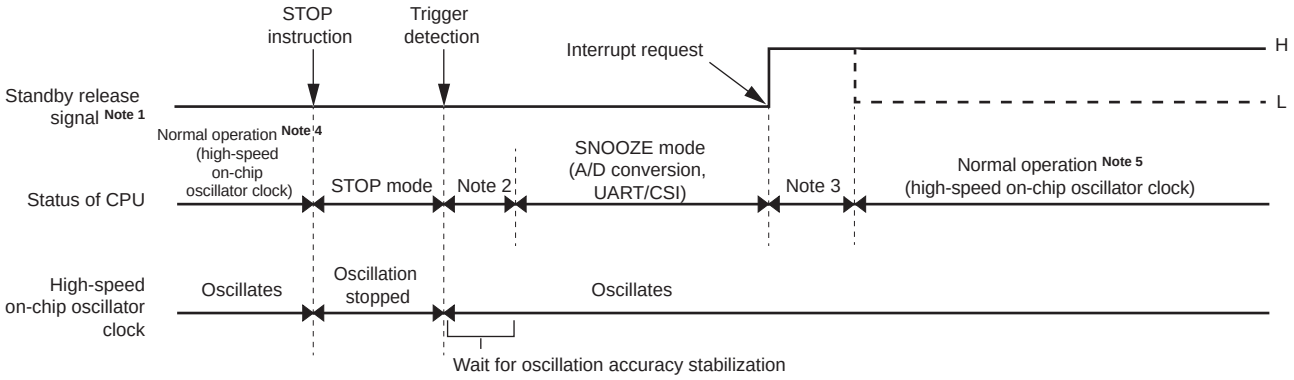
| Item \ STOP Mode Setting | | When Inputting CSI00/UART0 Data Reception Signal or A/D Converter Timer Trigger Signal While in STOP Mode |
|---|---|---|
| | | When CPU Is Operating on High-speed on-chip oscillator clock ($f_{IH}$) |
| System clock | | Clock supply to the CPU is stopped |
| Main system clock | $f_{IH}$ | Operation started |
| | $f_X$ | Stopped |
| | $f_{EX}$ | |
| Subsystem clock | $f_{XT}$ | Use of the status while in the STOP mode continues |
| | $f_{EXS}$ | |
| $f_{IL}$ | | Set by bits 0 (WDSTBYON) and 4 (WDTON) of option byte (000C0H), and WUTMMCK0 bit of subsystem clock supply mode control register (OSMC)<br>• WUTMMCK0 = 1: Oscillates<br>• WUTMMCK0 = 0 and WDTON = 0: Stops<br>• WUTMMCK0 = 0, WDTON = 1, and WDSTBYON = 1: Oscillates<br>• WUTMMCK0 = 0, WDTON = 1, and WDSTBYON = 0: Stops |
| CPU | | Operation stopped |
| Code flash memory | | |
| <R> RAM | | Operation stopped (Operable while in the DTC is executed) |
| Port (latch) | | Use of the status while in the STOP mode continues |
| Timer array unit | | Operation disabled |
| Real-time clock 2 | | Operable |
| Subsystem clock frequency measurement circuit | | Operation disabled |
| High-speed on-chip oscillator clock frequency correction function | | |
| Oscillation stop detection | | Operable (only when $f_{IL}$ is oscillating) |
| Battery backup function | | Operable (when VBATEN = 1 and VBATSEL = 0) |
| 12-bit interval timer | | Operable |
| 8-bit interval timer | | |
| Watchdog timer | | Operable (See **CHAPTER 13 WATCHDOG TIMER**) |
| Clock output/buzzer output | | Operable only when subsystem clock is selected as the count clock (when low-consumption RTC mode (set RTCLPC bit of OSMC register to 1), operation disabled) |
| A/D converter | | Operable |
| ΔΣA/D Converter | | Operation disabled |
| Temperature sensor 2 | | |
| Comparator | | Operable (when digital filter is not used) |
| Serial array unit (SAU) | | Operable only CSI00 and UART0 only. Operation disabled other than CSI00 and UART0. |
| IrDA | | Operation disabled |
| Serial interface (IICA) | | |
| LCD controller/driver | | Operable (However, this depends on the status of the clock selected as the LCD source clock: operation is possible if the selected clock is operating, but operation will stop if the selected clock is stopped.) |
| Data transfer controller (DTC) | | Operable |
| Power-on-reset function | | |
| Voltage detection function | | |
| External interrupt | | |
| CRC operation function | High-speed CRC | Operation stopped |
| | General-purpose CRC | |
| RAM parity error detection function | | |
| RAM guard function | | |
| SFR guard function | | |
| Illegal-memory access detection function | | |

(**Remark** is listed on the next page.)

**Remark** Operation stopped: Operation is automatically stopped before switching to the SNOOZE mode.
Operation disabled: Operation is stopped before switching to the SNOOZE mode.
$f_{IH}$: High-speed on-chip oscillator clock $f_{IL}$: Low-speed on-chip oscillator clock
$f_X$: X1 clock $f_{EX}$: External main system clock
$f_{XT}$: XT1 clock $f_{EXS}$: External subsystem clock

### (2) Timing diagram when the interrupt request signal is generated in the SNOOZE mode

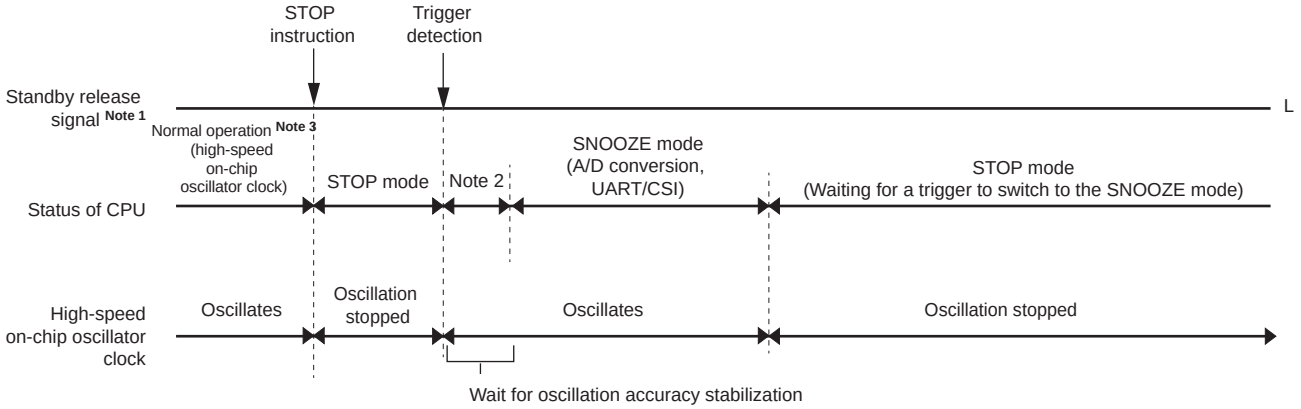
**Figure 24-5. When the Interrupt Request Signal is Generated in the SNOOZE Mode**




**Notes 1.** For details of the standby release signal, see **Figure 23-1**.
**2.** Transition time from STOP mode to SNOOZE mode
**3.** Transition time from SNOOZE mode to normal operation
**4.** Enable the SNOOZE mode (AWC = 1 or SWC = 1) immediately before switching to the STOP mode.
**5.** Be sure to release the SNOOZE mode (AWC = 0 or SWC = 0) immediately after return to the normal operation.

### (3) Timing diagram when the interrupt request signal is not generated in the SNOOZE mode

**Figure 24-6. When the Interrupt Request Signal is not Generated in the SNOOZE Mode**




**Notes 1.** For details of the standby release signal, see **Figure 23-1**.

**2.** Transition time from STOP mode to SNOOZE mode

**3.** Enable the SNOOZE mode (AWC = 1 or SWC = 1) immediately before switching to the STOP mode.

**Remark** For details of the SNOOZE mode function, see **CHAPTER 14 A/D CONVERTER** and **CHAPTER 18 SERIAL ARRAY UNIT**.

# CHAPTER 25 RESET FUNCTION

The following seven operations are available to generate a reset signal.

(1) External reset input via $\overline{\text{RESET}}$ pin
(2) Internal reset by watchdog timer program loop detection
(3) Internal reset by comparison of supply voltage and detection voltage of power-on-reset (POR) circuit
(4) Internal reset by comparison of supply voltage of the voltage detector (LVD) and detection voltage
(5) Internal reset by execution of illegal instruction[Note]
(6) Internal reset by RAM parity error
(7) Internal reset by illegal-memory access

External and internal resets start program execution from the address at 0000H and 0001H when the reset signal is generated.

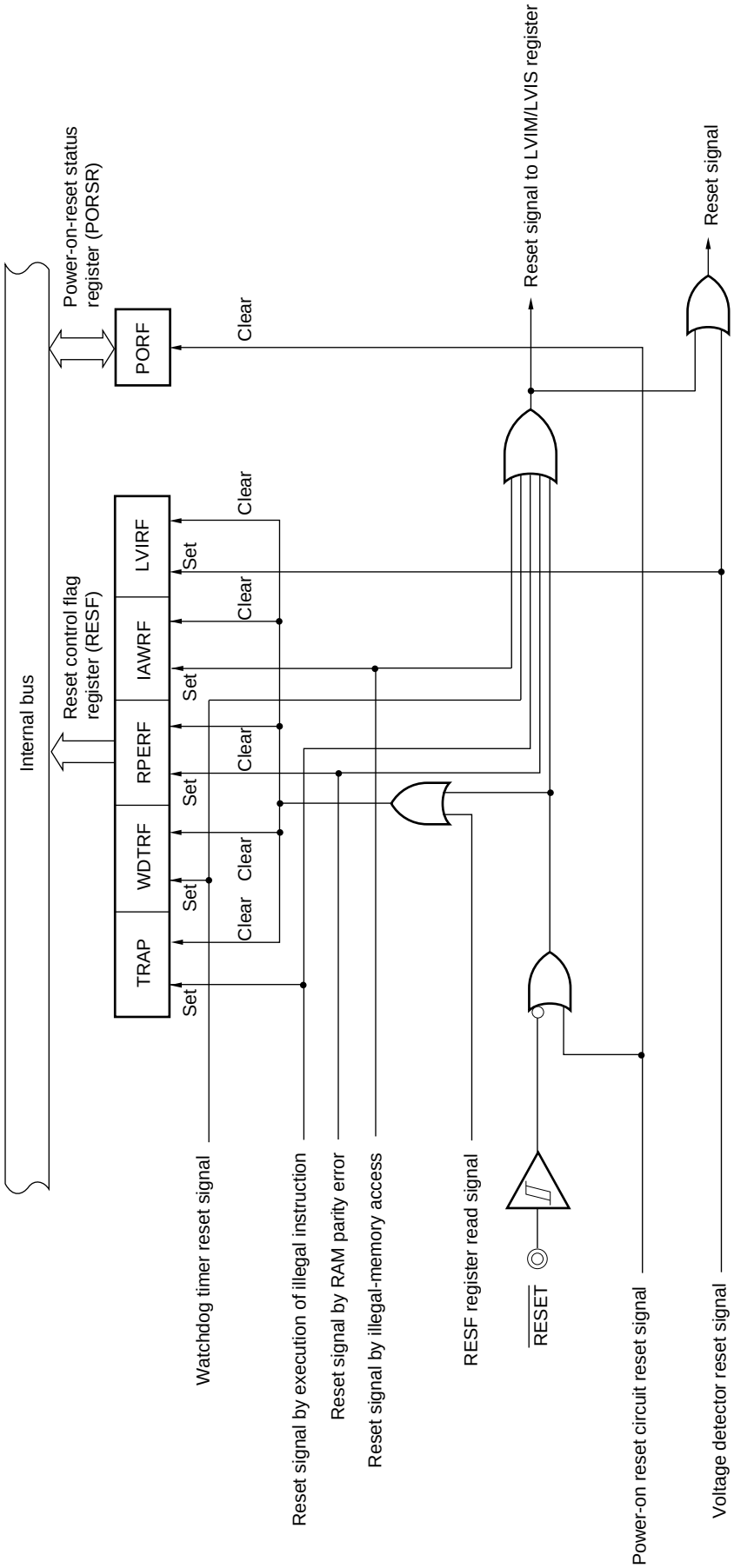
A reset is effected when a low level is input to the $\overline{\text{RESET}}$ pin, the watchdog timer overflows, or by POR and LVD circuit voltage detection, execution of illegal instruction[Note], RAM parity error or illegal-memory access, and each item of hardware is set to the status shown in Table 25-1.

**Note** This reset occurs when instruction code FFH is executed.
This reset does not occur during emulation using an in-circuit emulator or an on-chip debugging emulator.

**Cautions 1. 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 37.4 AC Characteristics, and then input a high level to the pin.**
**2. During reset input, the X1 clock, high-speed on-chip oscillator clock, and low-speed on-chip oscillator clock stop oscillating. External main system clock input and external subsystem clock input become invalid.**
**3. The port pins become the following state because each SFR and 2nd SFR are initialized after reset.**
- **P40: High-impedance during the external reset period or reset period by the POR. High level during other types of reset or after receiving a reset signal (connected to the internal pull-up resistor).**
- **P130: High-impedance during the reset period. Low level after receiving a reset signal.**
- **Ports other than P40 and P130: High-impedance during the reset period or after receiving a reset signal.**

**Remark** $V_{POR}$: POR power supply rise detection voltage
$V_{LVD}$: LVD detection voltage

**Figure 25-1. Block Diagram of Reset Function**




**Caution An LVD circuit internal reset does not reset the LVD circuit.**

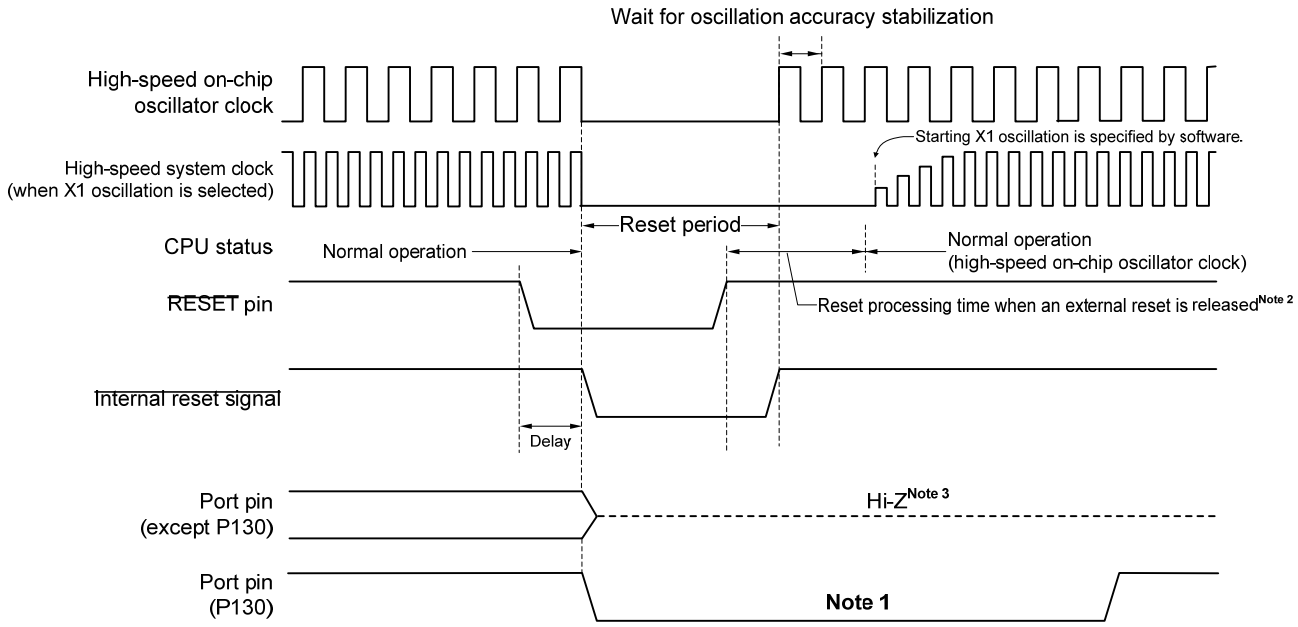
**Remarks 1.** LVIM: Voltage detection register

**2.** LVIS: Voltage detection level register

## 25.1 Timing of Reset Operation

This LSI 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 25-2. Timing of Reset by $\overline{\text{RESET}}$ Input**

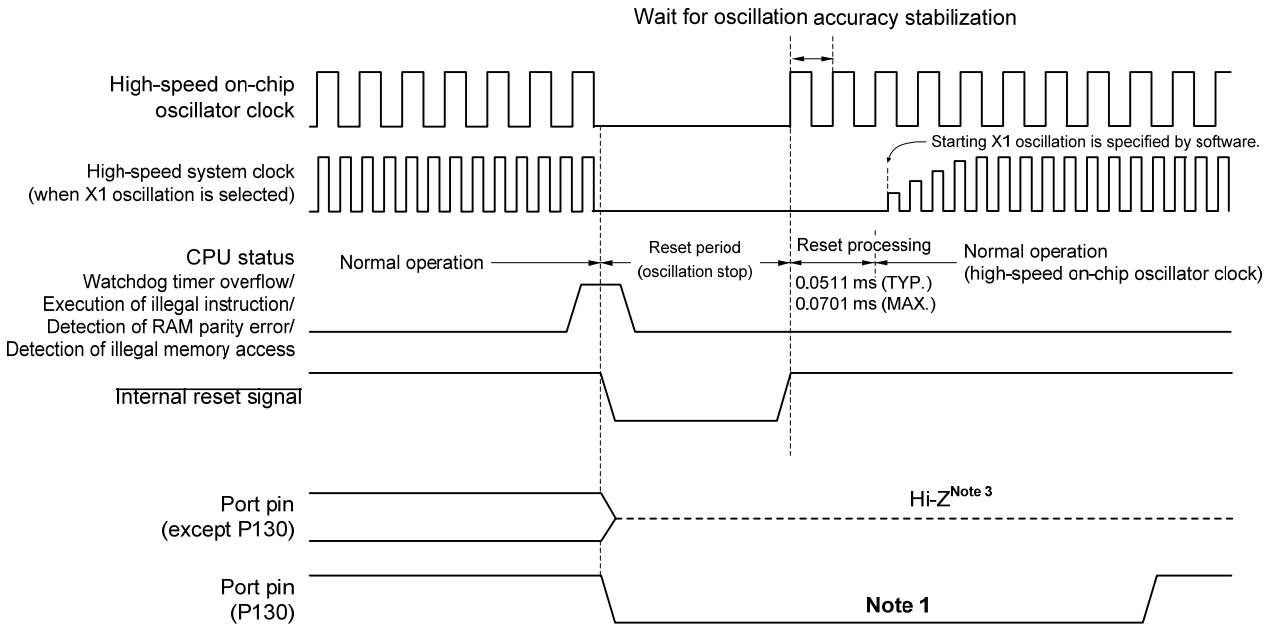



The input buffer of the $\overline{\text{RESET}}$ pin is connected to internal $V_{DD}$. When using the battery backup function, input signal based on the voltage of the selected power supply source ($V_{DD}$ pin or VBAT pin).

Release from the reset state is automatic in the case of a reset due to a watchdog timer overflow, 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.

(**Notes** and **Caution** are listed on the next page.)

**Figure 25-3. Timing of Reset Due to Watchdog Timer Overflow, Execution of Illegal Instruction, Detection of RAM Parity Error, or Detection of Illegal Memory**




**Notes 1.** When P130 is set to high-level output before reset is effected, the output signal of P130 can be dummy-output as a reset signal to an external device, because P130 outputs a low level when reset is effected. To release a reset signal to an external device, set P130 to high-level output by software.

**2.** Reset times (times for release from the external reset state)

After the first release of the POR: 0.672 ms (typ.), 0.832 ms (max.) when the LVD is in use.
0.399 ms (typ.), 0.519 ms (max.) when the LVD is off.

After the second release of the POR: 0.531 ms (typ.), 0.675 ms (max.) when the LVD is in use.
0.259 ms (typ.), 0.362 ms (max.) when the LVD is off.

After power is supplied, a voltage stabilization waiting time of about 0.99 ms (typ.) and up to 2.30 ms (max.) is required before reset processing starts after release of the external reset.

**3.** The state of P40 is as follows.

- High-impedance during the external reset period or reset period by the POR.
- High level during other types of reset or after receiving a reset signal (connected to the internal pull-up resistor).

<R>

Reset by POR and LVD circuit supply voltage detection is automatically released when internal $V_{DD} \geq V_{POR}$ or internal $V_{DD} \geq V_{LVD}$ after the reset. After reset processing, execution of the program with the high-speed on-chip oscillator clock as the operating clock starts.

For details, see **CHAPTER 26 POWER-ON-RESET CIRCUIT** or **CHAPTER 27 VOLTAGE DETECTOR**.

**Remark** $V_{POR}$: POR power supply rise detection voltage
$V_{LVD}$: LVD detection voltage

## 25.2 States of Operation During Reset Periods

Table 25-1 shows the states of operation during reset periods. Table 25-2 shows the states of the hardware after receiving a reset signal.

**Table 25-1. Operation Statuses During Reset Period**

| Item | | | During Reset Period |
|---|---|---|---|
| System clock | | | Clock supply to the CPU is stopped. |
| | Main system clock | $f_{IH}$ | Operation stopped |
| | | $f_X$ | Operation stopped (the X1 and X2 pins are input port mode) |
| | | $f_{EX}$ | Clock input invalid (the pin is input port mode) |
| | Subsystem clock | $f_{XT}$ | Operation possible (the XT1 and XT2 pins are input port mode) |
| | | $f_{EXS}$ | Clock input invalid (the pin is input port mode) |
| | $f_{IL}$ | | Operation stopped |
| CPU | | | |
| Code flash memory | | | |
| RAM | | | |
| Port (latch) | P40 | | Except pin reset and POR reset: Pull-up function enable<br>Pin reset and POR reset: High impedance |
| | P130 | | Undefined |
| | Other than P40, p130 | | High impedance[Note] |
| Timer array unit | | | Operation stopped |
| Real-time clock 2 | | | During a reset other than the POR reset: Operation possible<br>During a POR reset: Calendar operation possible; operation of the RTCC0, RTCC1, and SUBCUD registers stops. |
| Subsystem clock frequency measurement circuit | | | Operation stopped |
| High-speed on-chip oscillator clock frequency correction function | | | |
| Oscillation stop detection | | | |
| Battery backup function | | | |
| 12-bit interval timer | | | |
| 8-bit interval timer | | | |
| Watchdog timer | | | |
| Clock output/buzzer output | | | |
| A/D converter | | | |
| ΔΣ A/D Converter | | | |
| Temperature sensor 2 | | | |
| Comparator | | | |
| Serial array unit (SAU) | | | |
| IrDA | | | |
| Serial interface (IICA) | | | |
| LCD controller/driver | | | |
| Data transfer controller (DTC) | | | |
| Power-on-reset function | | | Detection operation possible |
| Voltage detection function | | | Operation is possible in the case of an LVD reset and stopped in the case of other types of reset. |
| External interrupt | | | Operation stopped |
| CRC operation function | High-speed CRC | | |
| | General-purpose CRC | | |
| RAM parity error detection function | | | |
| RAM guard function | | | |
| SFR guard function | | | |
| Illegal-memory access detection function | | | |

(**Note** and **Remark** are listed on the next page.)

**Note** P40 and P130 become the following state.

- P40: High-impedance during the external reset period or reset period by the POR. High level during other types of reset (connected to the internal pull-up resistor).
- P130: Low level during the reset period

**Remark** $f_{IH}$: High-speed on-chip oscillator clock
$f_X$: X1 oscillation clock
$f_{EX}$: External main system clock
$f_{XT}$: XT1 oscillation clock
$f_{EXS}$: External subsystem clock
$f_{IL}$: Low-speed on-chip oscillator clock

**Table 25-2. Hardware Statuses After Reset Acknowledgment**

| Hardware | | After Reset Acknowledgment[Note] |
|---|---|---|
| Program counter (PC) | | The contents of the reset vector table (0000H, 0001H) are set. |
| Stack pointer (SP) | | Undefined |
| Program status word (PSW) | | 06H |
| RAM | Data memory | Undefined |
| | General-purpose registers | Undefined |

**Note** During reset signal generation or oscillation stabilization time wait, only the PC contents among the hardware statuses become undefined. All other hardware statuses remain unchanged after reset.

**Remark** For the state of the special function register (SFR) after receiving a reset signal, see **3.2.4 Special function register (SFR) area** and **3.2.5 Extended special function register (2nd SFR: 2nd Special Function Register) area**.

## 25.3 Register for Confirming Reset Source

### 25.3.1 Reset control flag register (RESF)

Many internal reset generation sources exist in the RL78 microcontroller. The reset control flag register (RESF) is used to store which source has generated the reset request.

The RESF register can be read by an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input, reset by power-on-reset (POR) circuit, and reading the RESF register clear TRAP, WDTRF, RPERF, IAWRF, and LVIRF flags.

**Figure 25-4. Format of Reset Control Flag Register (RESF)**

Address: FFFA8H After reset: Undefined[Note 1] 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 | No internal reset request has been generated, or the RESF register has been cleared. |
| 1 | An internal reset request has been generated. |

| WDTRF | Internal reset request by watchdog timer (WDT) |
|---|---|
| 0 | No internal reset request has been generated, or the RESF register has been cleared. |
| 1 | An internal reset request has been generated. |

| RPERF | Internal reset request by RAM parity |
|---|---|
| 0 | No internal reset request has been generated, or the RESF register has been cleared. |
| 1 | An internal reset request has been generated. |

| IAWRF | Internal reset request by illegal-memory access |
|---|---|
| 0 | No internal reset request has been generated, or the RESF register has been cleared. |
| 1 | An internal reset request has been generated. |

| LVIRF | Internal reset request by voltage detector (LVD) |
|---|---|
| 0 | No internal reset request has been generated, or the RESF register has been cleared. |
| 1 | An internal reset request has been generated. |

**Notes 1.** The value after reset varies depending on the reset source. See **Table 25-3**.

**2.** This reset occurs when instruction code FFH is executed.
This reset does not occur during emulation using an in-circuit emulator or an on-chip debugging emulator.

**Cautions 1. Do not read data by a 1-bit memory manipulation instruction.**

**2. When enabling RAM parity error resets (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 30.3.3 RAM parity error detection function**.

The status of the RESF register when a reset request is generated is shown in Table 25-3.

**Table 25-3. RESF Register Status When Reset Request Is Generated**

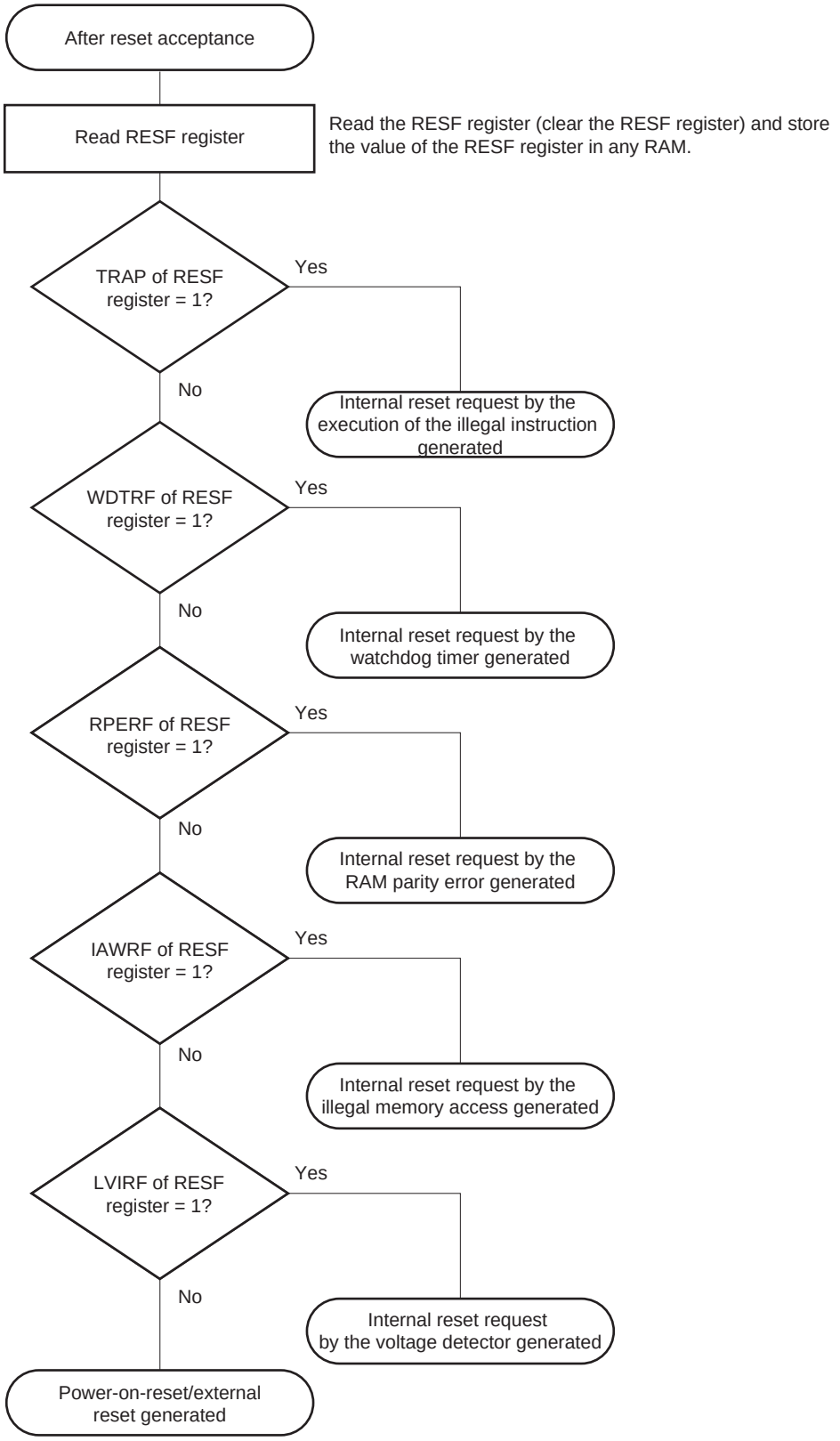
| Flag \ Reset Source | $\overline{\text{RESET}}$ Input | Reset by POR | Reset by Execution of Illegal Instruction | Reset by WDT | Reset by RAM parity error | Reset by illegal-memory access | Reset by LVD |
|---|---|---|---|---|---|---|---|
| TRAP bit | Cleared (0) | Cleared (0) | Set (1) | Held | Held | Held | Held |
| WDTRF bit | | | Held | Set (1) | | | |
| RPERF bit | | | | Held | Set (1) | | |
| IAWRF bit | | | | | Held | Set (1) | |
| LVIRF bit | | | | | | Held | Set (1) |

The RESF register is automatically cleared when it is read by an 8-bit memory manipulation instruction.

Figure 25-5 shows the procedure for checking a reset source.

<R>

**Figure 25-5. Example of Procedure for Checking Reset Source**



### 25.3.2 Power-on-reset status register (PORSR)

The PORSR register is used to check the occurrence of a power-on reset.

Writing “1” to bit 0 (PORF) of the PORSR register is valid, and writing “0” is ignored.

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 an 8-bit memory manipulation instruction.

Power-on reset signal generation clears this register to 00H.

**Cautions 1. The PORSR register is reset only by a power-on reset; it retains the value when a reset caused by another factor occurs.**

**2. If the PORF bit is set to 1, it guarantees that no power-on reset has occurred, but it does not guarantee that the RAM value is retained.**

**Figure 25-6. Format of Power-on-Reset Status Register (PORSR)**

Address: F00F9H After reset: 00H 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. |

# CHAPTER 26 POWER-ON-RESET CIRCUIT

## 26.1 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 ($V_{DD}$)[Note] exceeds the detection voltage ($V_{POR}$). However, be sure to maintain the reset state until the power supply voltage reaches the operating voltage range specified in **37.4 AC Characteristics**, by using the voltage detector or external reset pin.

- Compares supply voltage ($V_{DD}$)[Note] and detection voltage ($V_{PDR}$), generates internal reset signal when $V_{DD}$[Note] < $V_{PDR}$.
  Note that, after power is supplied, 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 operation voltage falls below the range defined in **37.4 AC Characteristics**. When restarting the operation, make sure that the operation voltage has returned within the range of operation.

**Note** Internal power supply voltage (internal $V_{DD}$) when using the battery backup function.

**Caution If an internal reset signal is generated in the power-on-reset circuit, the reset control flag register (RESF) and power-on-reset status register (PORSR) are cleared to 00H.**

**Remarks 1.** The RL78 microcontroller 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 **CHAPTER 25 RESET FUNCTION**.

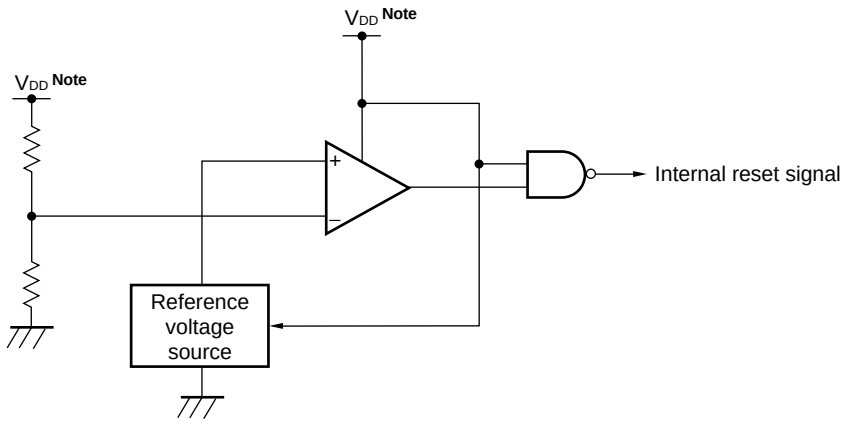
**2.** Whether an internal reset has been generated by the power-on reset circuit can be checked by using the power-on-reset status register (PORSR). For details of the PORSR register, see **CHAPTER 25 RESET FUNCTION**.

**3.** $V_{POR}$: POR power supply rise detection voltage
$V_{PDR}$: POR power supply fall detection voltage
For details, see **37.6.5 POR circuit characteristics**.

## 26.2 Configuration of Power-on-reset Circuit

The block diagram of the power-on-reset circuit is shown in Figure 26-1.

**Figure 26-1. Block Diagram of Power-on-reset Circuit**



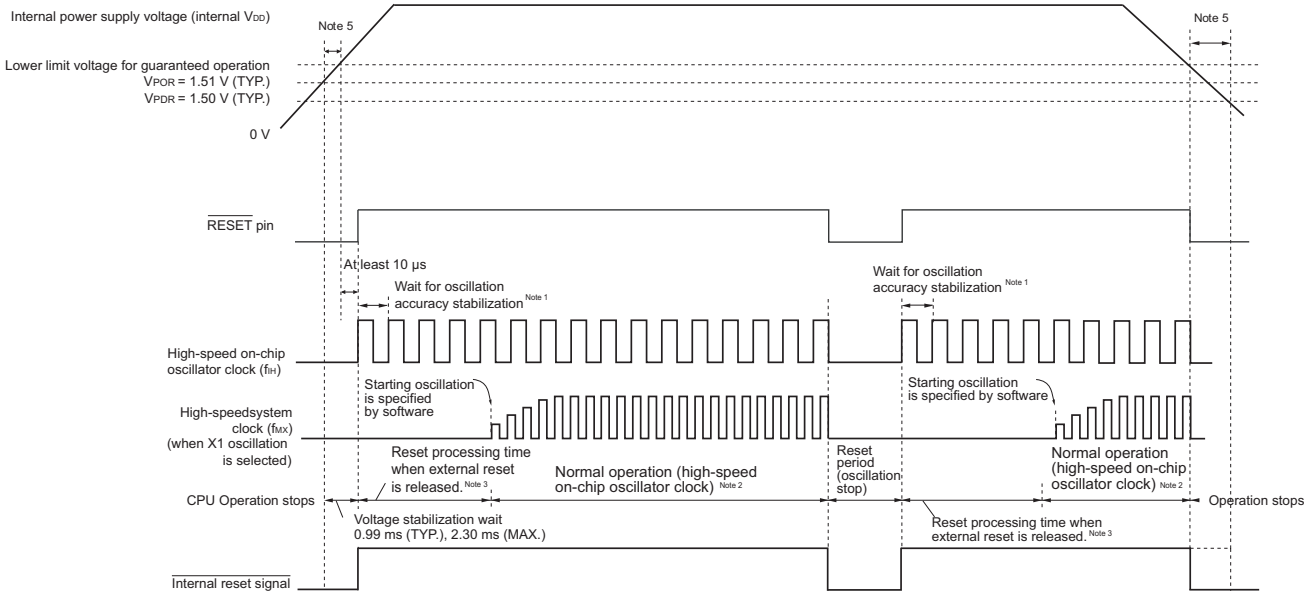

**Note** Internal power supply voltage (internal $V_{DD}$) when using the battery backup function.

## 26.3 Operation of Power-on-reset Circuit

The timing of generation of the internal reset signal by the power-on-reset circuit and voltage detector is shown below.

**Figure 26-2. Timing of Generation of Internal Reset Signal by Power-on-reset Circuit and Voltage Detector (1/3)**

**(1) When the externally input reset signal on the $\overline{\text{RESET}}$ pin is used**




**Notes 1.** The internal reset processing time includes the oscillation accuracy stabilization time of the high-speed on-chip oscillator clock.

**2.** The high-speed on-chip oscillator clock and a high-speed system clock or subsystem clock can be selected as the CPU clock. To use the X1 clock, use the oscillation stabilization time counter status register (OSTC) to confirm the lapse of the oscillation stabilization time. To use the XT1 clock, use the timer function for confirmation of the lapse of the stabilization time.

<R> **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 (1) as well as the voltage stabilization wait time after $V_{POR}$ (1.51 V, typ.) is reached.
Reset processing time when the external reset is released is shown below.
Release from the first external reset following release from the POR state:
0.672 ms (typ.), 0.832 ms (max.) (when the LVD is in use)
0.399 ms (typ.), 0.519 ms (max.) (when the LVD is off)

<R> **4.** Reset times in cases of release from an external reset other than the above are listed below.
Release from the reset state for external resets other than the above case:
0.531 ms (typ.), 0.675 ms (max.) (when the LVD is in use)
0.259 ms (typ.), 0.362 ms (max.) (when the LVD is off)

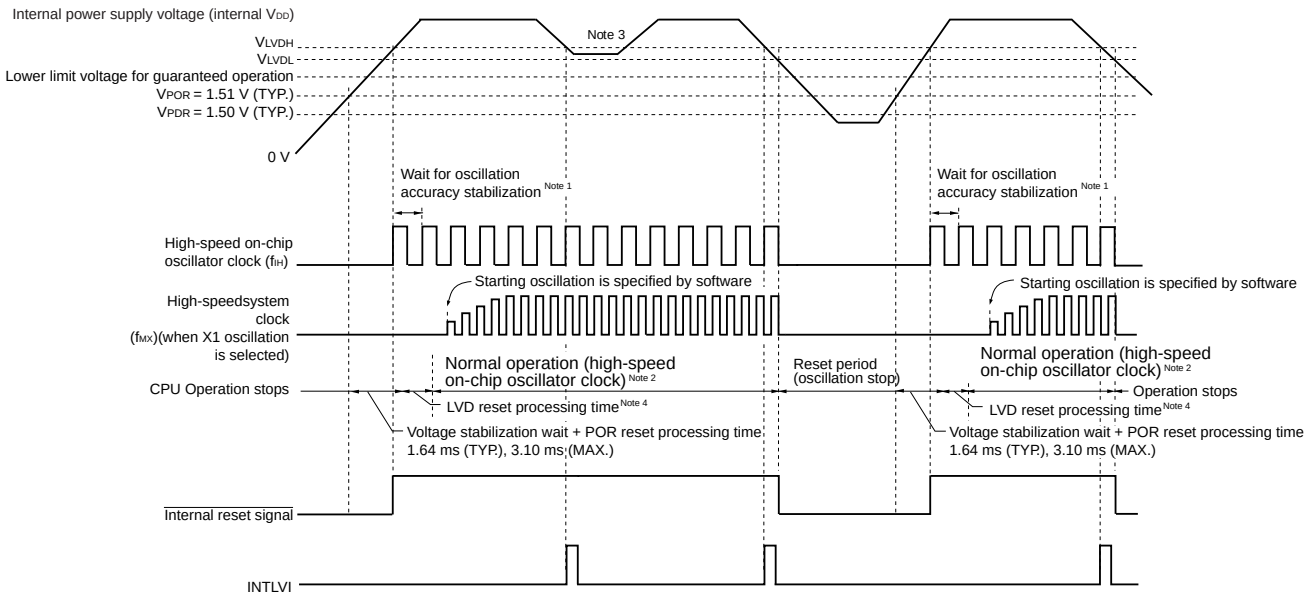
**5.** After power is supplied, the reset state must be retained until the operating voltage becomes in the range defined in **37.4 AC Characteristics**. This is done by controlling the externally input reset signal. After power supply is turned off, this LSI should be placed in the STOP mode, or in the reset state by utilizing the voltage detection circuit or externally input reset signal, before the voltage falls below the operating range. When restarting the operation, make sure that the operation voltage has returned within the range of operation.

**Remark** $V_{POR}$: POR power supply rise detection voltage
$V_{PDR}$: POR power supply fall detection voltage

**Caution For power-on reset, be sure to use the externally input reset signal on the $\overline{\text{RESET}}$ pin when the LVD is off. For details, see CHAPTER 27 VOLTAGE DETECTOR.**

**Figure 26-2. Timing of Generation of Internal Reset Signal by Power-on-reset Circuit and Voltage Detector (2/3)**

**(2) LVD interrupt & reset mode (option byte 000C1H: LVIMDS1, LVIMDS0 = 1, 0)**




**Notes 1.** The internal reset processing time includes the oscillation accuracy stabilization time of the high-speed on-chip oscillator clock.

**2.** The high-speed on-chip oscillator clock and a high-speed system clock or subsystem clock can be selected as the CPU clock. To use the X1 clock, use the oscillation stabilization time counter status register (OSTC) to confirm the lapse of the oscillation stabilization time. To use the XT1 clock, use the timer function for confirmation of the lapse of the stabilization time.

<R> **3.** After the interrupt request signal (INTLVI) is generated, the LVILV and LVIMD bits of the voltage detection level register (LVIS) are automatically set to 1. After INTLVI is generated, appropriate settings should be made according to **Figure 27-8 Setting Procedure for Operating Voltage Check/Reset** and **Figure 27-9 Initial Setting of Interrupt and Reset Mode**, taking into consideration that the supply voltage might return to the high voltage detection level ($V_{LVDH}$) or higher without falling below the low voltage detection level ($V_{LVDL}$).

**4.** The time until normal operation starts includes the following LVD reset processing time after the LVD detection level ($V_{LVDH}$) is reached as well as the voltage stabilization wait + POR reset processing time after the $V_{POR}$ (1.51 V, typ.) is reached.

LVD reset processing time: 0 ms to 0.0701 ms (max.)

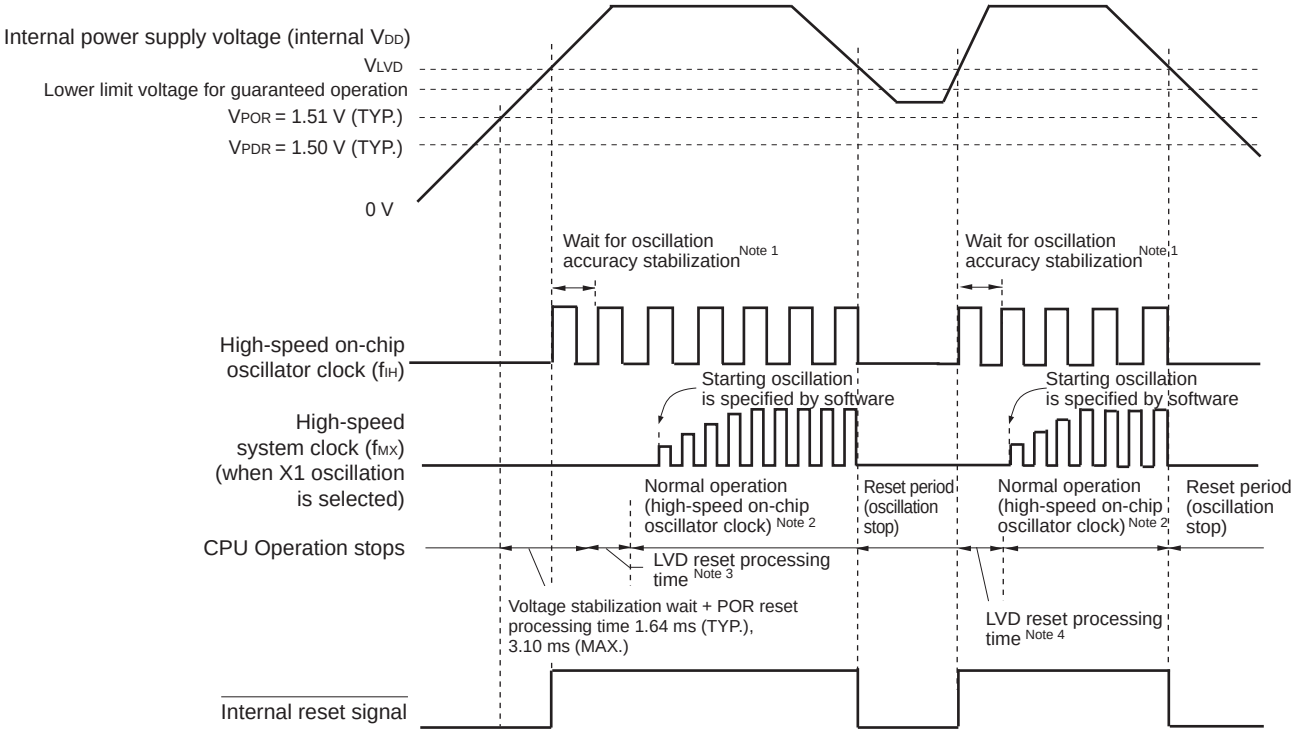
**Remark** $V_{LVDH}$, $V_{LVDL}$: LVD detection voltage

$V_{POR}$: POR power supply rise detection voltage

$V_{PDR}$: POR power supply fall detection voltage

**Figure 26-2. Timing of Generation of Internal Reset Signal by Power-on-reset Circuit and Voltage Detector (3/3)**

**(3) LVD reset mode (option byte 000C1H: LVIMDS1 = 1, LVIMDS0 = 1)**




**Notes 1.** The internal reset processing time includes the oscillation accuracy stabilization time of the high-speed on-chip oscillator clock.

**2.** The high-speed on-chip oscillator clock and a high-speed system clock or subsystem clock can be selected as the CPU clock. To use the X1 clock, use the oscillation stabilization time counter status register (OSTC) to confirm the lapse of the oscillation stabilization time. To use the XT1 clock, use the timer function for confirmation of the lapse of the stabilization time.

**3.** The time until normal operation starts includes the following LVD reset processing time after the LVD detection level ($V_{LVD}$) is reached as well as the voltage stabilization wait + POR reset processing time after the $V_{POR}$ (1.51 V, typ.) is reached.

LVD reset processing time: 0 ms to 0.0701 ms (max.)

**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 (LVD), the following LVD reset processing time is required after the LVD detection level ($V_{LVD}$) is reached.

LVD reset processing time: 0.0511 ms (typ.), 0.0701 ms (max.)

**Remarks 1.** $V_{LVDH}$, $V_{LVDL}$: LVD detection voltage

$V_{POR}$: POR power supply rise detection voltage

$V_{PDR}$: POR power supply fall detection voltage

**2.** When the LVD interrupt mode is selected (option byte 000C1H: LVIMD1 = 0, LVIMD0 = 1), the time until normal operation starts after power is turned on is the same as the time specified in **Note 3** of Figure 26-2 (3).

# CHAPTER 27 VOLTAGE DETECTOR

## 27.1 Functions of Voltage Detector

The operation mode and detection voltages ($V_{LVDH}$, $V_{LVDL}$, $V_{LVD}$) for the voltage detector is set by using the option byte (000C1H).

The voltage detector (LVD) has the following functions.

- The LVD circuit compares the internal power supply voltage (internal $V_{DD}$) that supplied from the $V_{DD}$ or VBAT pin with the detection voltage ($V_{LVDH}$, $V_{LVDL}$, $V_{LVD}$), and generates an internal reset or internal interrupt signal.
- The detection level for the internal power supply detection voltage ($V_{LVDH}$, $V_{LVDL}$, $V_{LVD}$) can be selected by using the option byte as one of 11 levels (for details, see **CHAPTER 32 OPTION BYTE**).
- Operable in STOP mode.
- <R> After power is supplied, the reset state must be retained until the operating voltage becomes in the range defined in **37.4 AC Characteristics**. This is done by utilizing the voltage detector 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 010C2H).

(a) Interrupt & reset mode (option byte LVIMDS1, LVIMDS0 = 1, 0)

The two detection voltages ($V_{LVDH}$, $V_{LVDL}$) are selected by the option byte 000C1H. The high-voltage detection level ($V_{LVDH}$) is used for releasing resets and generating interrupts. The low-voltage detection level ($V_{LVDL}$) is used for generating resets.

(b) Reset mode (option byte LVIMDS1, LVIMDS0 = 1, 1)

The detection voltage ($V_{LVD}$) selected by the option byte 000C1H is used for triggering and ending resets.

(c) Interrupt mode (option byte LVIMDS1, LVIMDS0 = 0, 1)

The detection voltage ($V_{LVD}$) selected by the option byte 000C1H is used for releasing resets and generating interrupts.

The reset and internal interrupt signals are generated in each mode as follows.

| Interrupt & Reset Mode (LVIMDS1, LVIMDS0 = 1, 0) | Reset Mode (LVIMDS1, LVIMDS0 = 1, 1) | Interrupt Mode (LVIMDS1, LVIMDS0 = 0, 1) |
|---|---|---|
| Generates an interrupt request signal by detecting internal power supply voltage (internal $V_{DD}$) < $V_{LVDH}$ when the operating voltage falls, and an internal reset by detecting internal power supply voltage (internal $V_{DD}$) < $V_{LVDL}$.<br>Releases an internal reset by detecting internal power supply voltage (internal $V_{DD}$) ≥ $V_{LVDH}$. | Releases an internal reset by detecting internal power supply voltage (internal $V_{DD}$) ≥ $V_{LVD}$.<br>Generates an internal reset by detecting internal power supply voltage (internal $V_{DD}$) < $V_{LVD}$. | The state of an internal reset by LVD is retained until internal $V_{DD}$ ≥ $V_{LVD}$ immediately after reset generation. The internal reset is released when internal $V_{DD}$ ≥ $V_{LVD}$ is detected.<br>After that, an interrupt request signal (INTLVI) is generated when internal $V_{DD}$ < $V_{LVD}$ or internal $V_{DD}$ ≥ $V_{LVD}$ is detected. |

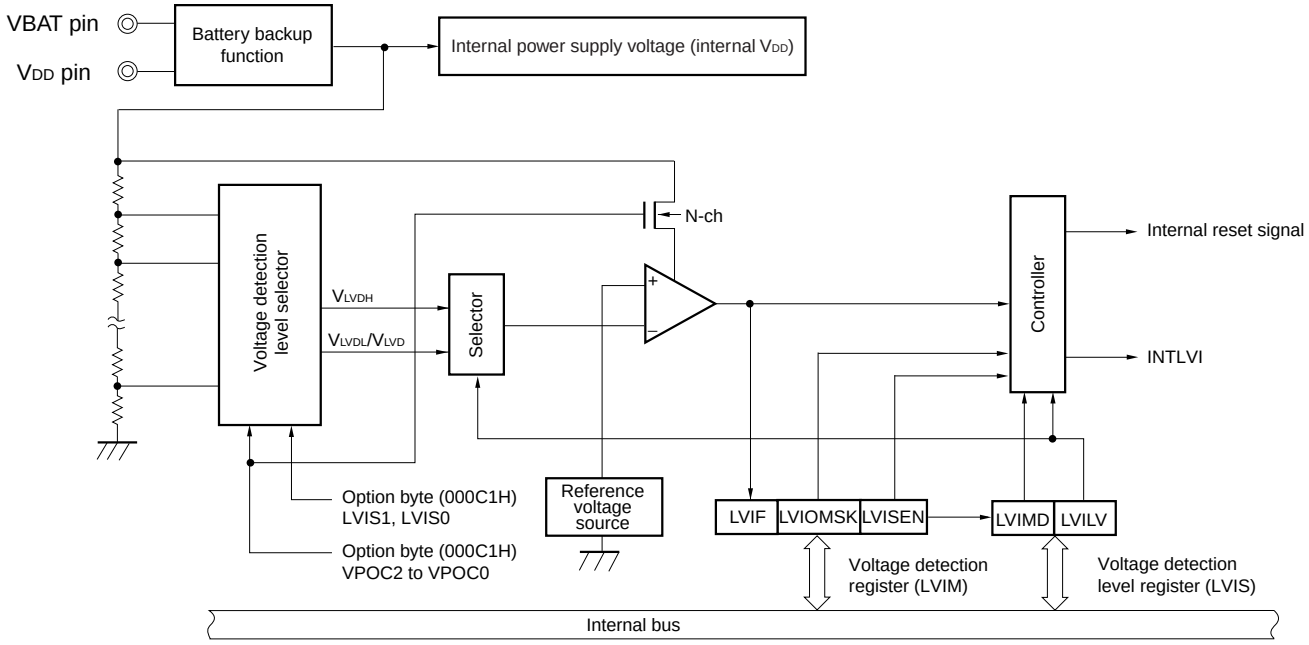
While the voltage detector is operating, whether the internal supply voltage or the input voltage from an external input pin is more than or less than the detection level can be checked by reading the voltage detection flag (LVIF: bit 0 of the voltage detection register (LVIM)).

Bit 0 (LVIRF) of the reset control flag register (RESF) is set to 1 if reset occurs. For details of the RESF register, see **CHAPTER 25 RESET FUNCTION**.

## 27.2 Configuration of Voltage Detector

The block diagram of the voltage detector is shown in Figure 27-1.

**Figure 27-1. Block Diagram of Voltage Detector**




## 27.3 Registers Controlling Voltage Detector

The voltage detector is controlled by the following registers.

- Voltage detection register (LVIM)
- Voltage detection level register (LVIS)

### 27.3.1 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 LVD output mask status.

This register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 27-2. Format of Voltage Detection Register (LVIM)**

Address: FFFA9H After reset: 00H[Note 1] R/W[Note 2]

| Symbol | <7> | 6 | 5 | 4 | 3 | 2 | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| LVIM | LVISEN[Note 3] | 0 | 0 | 0 | 0 | 0 | LVIOMSK | LVIF |

| LVISEN[Note 3] | Specification of whether to enable or disable rewriting the voltage detection level register (LVIS) |
|---|---|
| 0 | Disabling of rewriting the LVIS register (LVIOMSK = 0 (Mask of LVD output is invalid) |
| 1 | Enabling of rewriting the LVIS register (LVIOMSK = 1 (Mask of LVD output is valid) |

| LVIOMSK | Mask status flag of LVD output |
|---|---|
| 0 | Mask of LVD output is invalid |
| 1 | Mask of LVD output is valid[Notes 3, 4] |

| LVIF | Voltage detection flag |
|---|---|
| 0 | Internal power supply voltage (internal $V_{DD}$) ≥ detection voltage ($V_{LVD}$), or when LVD is off |
| 1 | Internal power supply voltage (internal $V_{DD}$) < detection voltage ($V_{LVD}$) |

**Notes 1.** The reset value changes depending on the reset source.
If the LVIS register is reset by LVD, it is not reset but holds the current value. In other reset, LVISEN is cleared to 0.

**2.** Bits 0 and 1 are read-only.

**3.** This can only be set in the interrupt & reset mode (option byte LVIMDS1, LVIMDS0 = 1, 0). Do not change the initial value in other modes.

**4.** LVIOMSK bit is automatically set to “1” only in the interrupt & reset mode (option byte LVIMDS1, LVIMDS0 = 1, 0) and reset or interrupt by LVD is masked.
- Period during LVISEN = 1
- Waiting period from the time when LVD interrupt is generated until LVD detection voltage becomes stable
- Waiting period from the time when the value of LVILV bit changes until LVD detection voltage becomes stable

### 27.3.2 Voltage detection level register (LVIS)

This register selects the voltage detection level.

This register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation input sets this register to 00H/01H/81H[Note 1].

**Figure 27-3. Format of Voltage Detection Level Select Register (LVIS)**

Address: FFFAAH After reset: 00H/01H/81H[Note 1] R/W

| Symbol | <7> | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| LVIS | LVIMD[Note 2] | 0 | 0 | 0 | 0 | 0 | 0 | LVILV[Note 2] |

| LVIMD[Note 2] | Operation mode of voltage detection |
|---|---|
| 0 | Interrupt mode |
| 1 | Reset mode |

| LVILV[Note 2] | LVD detection level |
|---|---|
| 0 | High-voltage detection level ($V_{LVDH}$) |
| 1 | Low-voltage detection level ($V_{LVDL}$ or $V_{LVDL}$) |

**Notes 1.** The reset value changes depending on the reset source and the setting of the option byte.
This register is not cleared (00H) by LVD reset.
The generation of reset signal other than an LVD reset sets as follows.
- When option byte LVIMDS1, LVIMDS0 = 1, 0: 00H
- When option byte LVIMDS1, LVIMDS0 = 1, 1: 81H
- When option byte LVIMDS1, LVIMDS0 = 0, 1: 01H

**2.** Writing “0” can only be allowed in the interrupt & reset mode (option byte LVIMDS1, LVIMDS0 = 1, 0). Do not set LVIMD and LVILV in other cases. The value is switched automatically when reset or interrupt is generated in the interrupt & reset mode.

**Cautions 1. Rewrite the value of the LVIS register according to Figures 27-8 and 27-9.**

**2. Specify the LVD operation mode and detection voltage ($V_{LVDH}$, $V_{LVDL}$, $V_{LVD}$) of each mode by using the option byte 000C1H. Figure 27-4 shows the format of the user option byte (000C1H/010C1H). For details about the option byte, see CHAPTER 32 OPTION BYTE.**

**Figure 27-4. LVD Operation Mode and Detection Voltage Settings for User Option Byte (000C1H) (1/2)**

Address: 000C1H/010C1H[Note]

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| VPOC2 | VPOC1 | VPOC0 | 1 | LVIS1 | LVIS0 | LVIMDS1 | LVIMDS0 |

- LVD setting (interrupt & reset mode)

| Detection voltage | | | Option byte Setting Value | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $V_{LVDH}$ | | $V_{LVDL}$ | VPOC2 | VPOC1 | VPOC0 | LVIS1 | LVIS0 | Mode setting | |
| Rising edge | Falling edge | Falling edge | | | | | | LVIMDS1 | LVIMDS0 |
| 2.61 V | 2.55 V | 2.45 V | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| 2.71 V | 2.65 V | | | | | 0 | 1 | | |
| 3.75 V | 3.67 V | | | | | 0 | 0 | | |
| 2.92 V | 2.86 V | 2.75 V | | 1 | 1 | 1 | 0 | | |
| 3.02 V | 2.96 V | | | | | 0 | 1 | | |
| 4.06 V | 3.98 V | | | | | 0 | 0 | | |
| – | | | Setting of values other than above is prohibited. | | | | | | |

- LVD setting (reset mode)

| Detection voltage | | Option byte Setting Value | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $V_{LVD}$ | | VPOC2 | VPOC1 | VPOC0 | LVIS1 | LVIS0 | Mode setting | |
| Rising edge | Falling edge | | | | | | LVIMDS1 | LVIMDS0 |
| 1.98 V | 1.94 V | 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| 2.09 V | 2.04 V | | 0 | 1 | 0 | 1 | | |
| 2.50 V | 2.45 V | | 1 | 0 | 1 | 1 | | |
| 2.61 V | 2.55 V | | 1 | 0 | 1 | 0 | | |
| 2.71 V | 2.65 V | | 1 | 0 | 0 | 1 | | |
| 2.81 V | 2.75 V | | 1 | 1 | 1 | 1 | | |
| 2.92 V | 2.86 V | | 1 | 1 | 1 | 0 | | |
| 3.02 V | 2.96 V | | 1 | 1 | 0 | 1 | | |
| 3.13 V | 3.06 V | | 0 | 1 | 0 | 0 | | |
| 3.75 V | 3.67 V | | 1 | 0 | 0 | 0 | | |
| 4.06 V | 3.98 V | | 1 | 1 | 0 | 0 | | |
| – | | Setting of values other than above is prohibited. | | | | | | |

**Note** Set the same value as 000C1H to 010C1H when the boot swap operation is used because 000C1H is replaced by 010C1H.

**Remarks 1.** For details on the LVD circuit, see **CHAPTER 27 VOLTAGE DETECTOR**.

**2.** The detection voltage is a TYP. value. For details, see **37.6.6 LVD circuit characteristics**.

(**Cautions** are listed on the next page)

**Figure 27-4. LVD Operation Mode and Detection Voltage Settings for User Option Byte (000C1H) (2/2)**

Address: 000C1H/010C1H[Note]

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| VPOC2 | VPOC1 | VPOC0 | 1 | LVIS1 | LVIS0 | LVIMDS1 | LVIMDS0 |

- LVD setting (interrupt mode)

| Detection voltage | | Option byte Setting Value | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $V_{LVD}$ | | VPOC2 | VPOC1 | VPOC0 | LVIS1 | LVIS0 | Mode setting | |
| Rising edge | Falling edge | | | | | | LVIMDS1 | LVIMDS0 |
| 1.98 V | 1.94 V | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 2.09 V | 2.04 V | | 0 | 1 | 0 | 1 | | |
| 2.50 V | 2.45 V | | 1 | 0 | 1 | 1 | | |
| 2.61 V | 2.55 V | | 1 | 0 | 1 | 0 | | |
| 2.71 V | 2.65 V | | 1 | 0 | 0 | 1 | | |
| 2.81 V | 2.75 V | | 1 | 1 | 1 | 1 | | |
| 2.92 V | 2.86 V | | 1 | 1 | 1 | 0 | | |
| 3.02 V | 2.96 V | | 1 | 1 | 0 | 1 | | |
| 3.13 V | 3.06 V | | 0 | 1 | 0 | 0 | | |
| 3.75 V | 3.67 V | | 1 | 0 | 0 | 0 | | |
| 4.06 V | 3.98 V | | 1 | 1 | 0 | 0 | | |
| – | | Setting of values other than above is prohibited. | | | | | | |

- LVD off setting (use of external reset input via $\overline{\text{RESET}}$ pin)

| Detection voltage | | Option byte Setting Value | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $V_{LVD}$ | | VPOC2 | VPOC1 | VPOC0 | LVIS1 | LVIS0 | Mode setting | |
| Rising edge | Falling edge | | | | | | LVIMDS1 | LVIMDS0 |
| – | – | 1 | × | × | × | × | × | 1 |
| – | | Setting of values other than above is prohibited. | | | | | | |

**Note** Set the same value as 000C1H to 010C1H when the boot swap operation is used because 000C1H is replaced by 010C1H.

**Cautions 1. Be sure to set bit 4 to "1".**

**2. After power is supplied, the reset state must be retained until the operating voltage becomes in the range defined in 37.4 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 010C2H).**

**Remarks 1.** ×: don't care

**2.** For details on the LVD circuit, see **CHAPTER 27 VOLTAGE DETECTOR**.

**3.** The detection voltage is a TYP. value. For details, see **37.6.6 LVD circuit characteristics**.

## 27.4 Operation of Voltage Detector

### 27.4.1 When used as reset mode

Specify the operation mode (the reset mode (LVIMDS1, LVIMDS0 = 1, 1)) and the detection voltage ($V_{LVD}$) by using the option byte 000C1H.

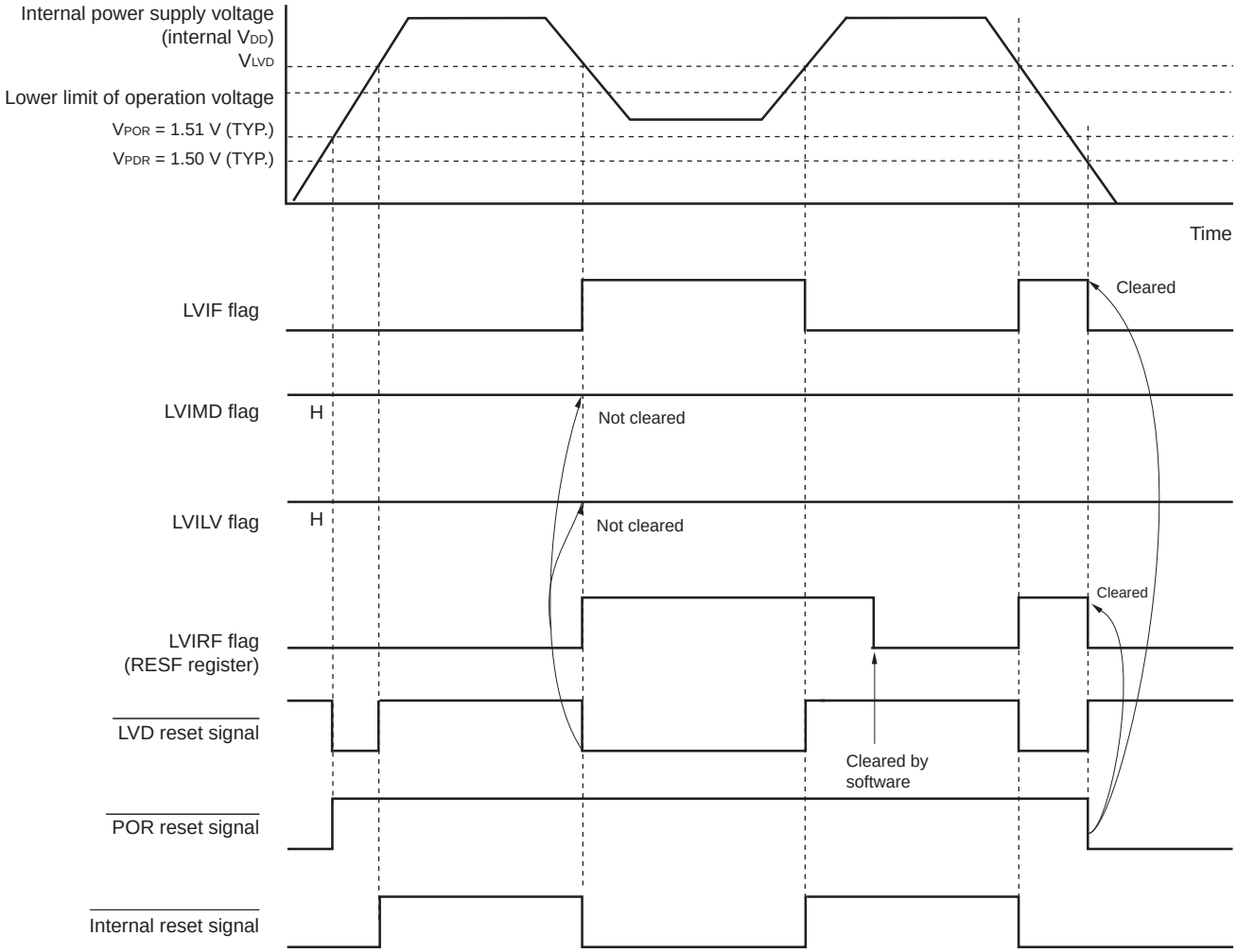
The operation is started in the following initial setting state when the reset mode is set.

- Bit 7 (LVISEN) of the voltage detection register (LVIM) is set to 0 (disable rewriting of voltage detection level register (LVIS))
- The initial value of the voltage detection level select register (LVIS) is set to 81H.
  Bit 7 (LVIMD) is 1 (reset mode).
  Bit 0 (LVILV) is 1 (low-voltage detection level: $V_{LVD}$).

- Operation in LVD reset mode
  In the reset mode (option byte LVIMDS1, LVIMDS0 = 1, 1), the state of an internal reset by LVD is retained until the internal power supply voltage (internal $V_{DD}$) exceeds the voltage detection level ($V_{LVD}$) after power is supplied. The internal reset is released when the internal power supply voltage (internal $V_{DD}$) exceeds the voltage detection level ($V_{LVD}$).
  At the fall of the operating voltage, an internal reset by LVD is generated when the internal power supply voltage (internal $V_{DD}$) falls below the voltage detection level ($V_{LVD}$).

Figure 27-5 shows the timing of the internal reset signal generated in the LVD reset mode.

**Figure 27-5. Timing of Voltage Detector Internal Reset Signal Generation (Option Byte LVIMDS1, LVIMDS0 = 1, 1)**




**Remark** $V_{POR}$: POR power supply rise detection voltage
$V_{PDR}$: POR power supply fall detection voltage

### 27.4.2 When used as interrupt mode

Specify the operation mode (the interrupt mode (LVIMDS1, LVIMDS0 = 0, 1)) and the detection voltage ($V_{LVD}$) by using the option byte 000C1H.

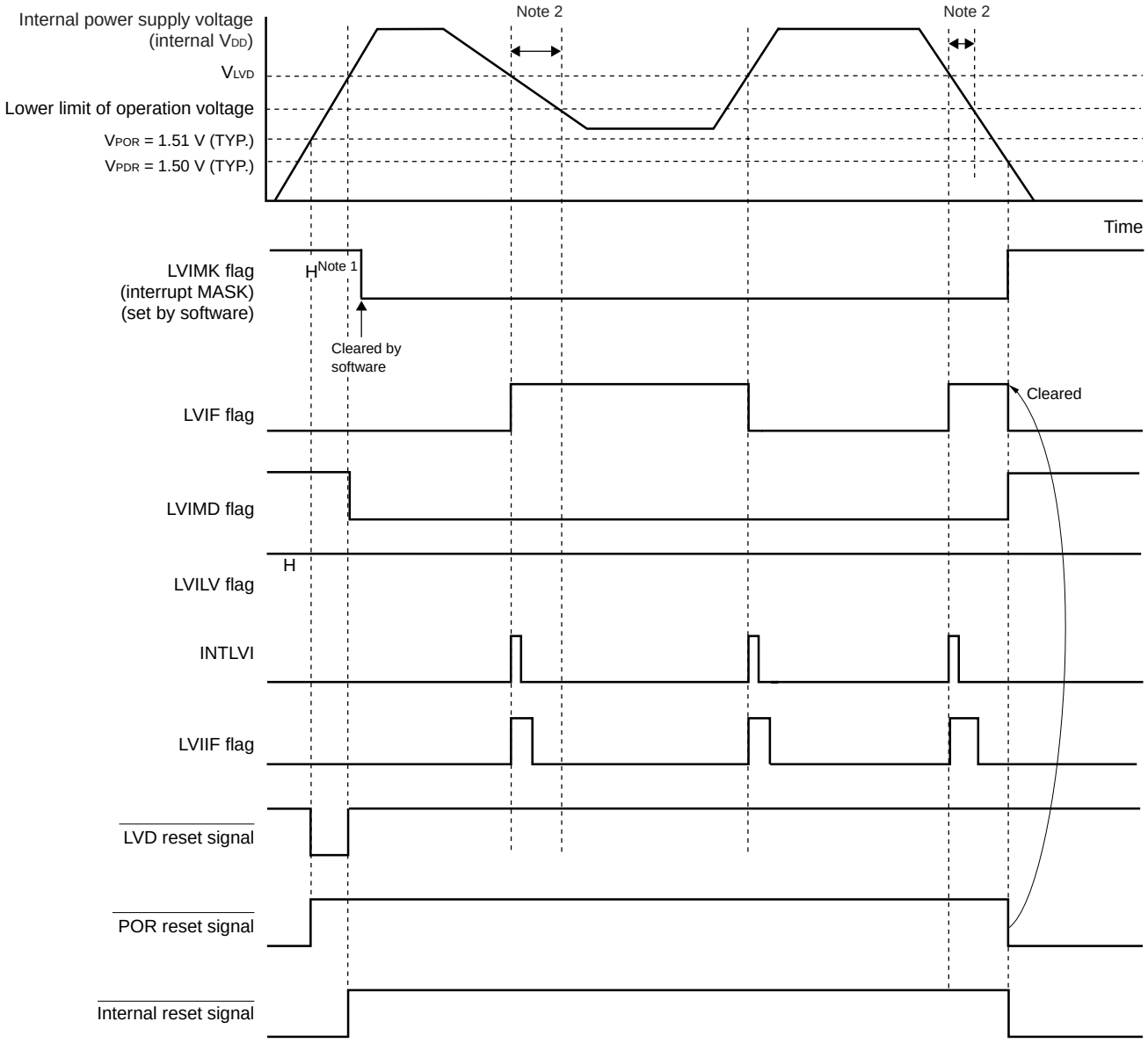
The operation is started in the following initial setting state when the interrupt mode is set.

- Bit 7 (LVISEN) of the voltage detection register (LVIM) is set to 0 (disable rewriting of voltage detection level register (LVIS))
- The initial value of the voltage detection level select register (LVIS) is set to 01H.
  Bit 7 (LVIMD) is 0 (interrupt mode).
  Bit 0 (LVILV) is 1 (low-voltage detection level: $V_{LVD}$).

- Operation in LVD interrupt mode
  In the interrupt mode (option byte LVIMDS1, LVIMDS0 = 0, 1), the state of an internal reset by LVD is retained until the internal power supply voltage (internal $V_{DD}$) exceeds the voltage detection level ($V_{LVD}$) immediately after reset generation. The internal reset by LVD is released when the internal power supply voltage (internal $V_{DD}$) exceeds the voltage detection level ($V_{LVD}$).
  After that, an interrupt request signal (INTLVI) is generated when the internal power supply voltage (internal $V_{DD}$) exceeds the voltage detection level ($V_{LVD}$). 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 **37.4 AC characteristics**. When restarting the operation, make sure that the internal power supply voltage has returned within the range of operation.

Figure 27-6 shows the timing of the interrupt request signal generated in the LVD interrupt mode.

**Figure 27-6. Timing of Voltage Detector Internal Interrupt Signal Generation (Option Byte LVIMDS1, LVIMDS0 = 0, 1)**




**Notes 1.** The LVIMK flag is set to “1” by reset signal generation.

**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 **37.4 AC characteristics**. When restarting the operation, make sure that the internal power supply voltage has returned within the range of operation.

**Remark** $V_{POR}$: POR power supply rise detection voltage
$V_{PDR}$: POR power supply fall detection voltage

### 27.4.3 When used as interrupt & reset mode

Specify the operation mode (the interrupt & reset (LVIMDS1, LVIMDS0 = 1, 0)) and the detection voltage ($V_{LVDH}$, $V_{LVDL}$) by using the option byte 000C1H.

The operation is started in the following initial setting state when the interrupt & reset mode is set.

- Bit 7 (LVISEN) of the voltage detection register (LVIM) is set to 0 (disable rewriting of voltage detection level register (LVIS))
- The initial value of the voltage detection level select register (LVIS) is set to 00H.
  Bit 7 (LVIMD) is 0 (interrupt mode).
  Bit 0 (LVILV) is 0 (high-voltage detection level: $V_{LVDH}$).

- Operation in LVD interrupt & reset mode

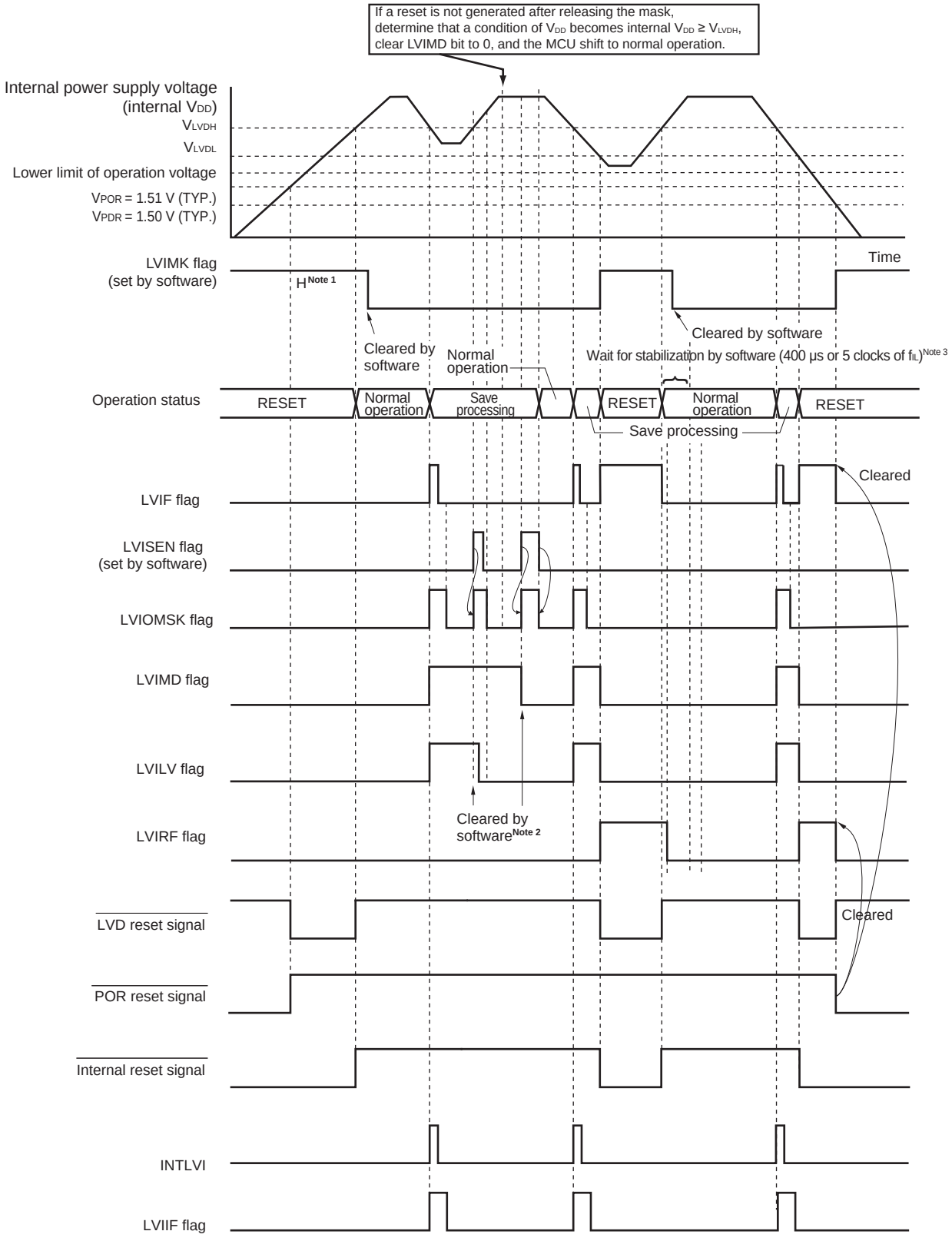
  In the interrupt & reset mode (option byte LVIMDS1, LVIMDS0 = 1, 0), the state of an internal reset by LVD is retained until the internal power supply voltage (internal $V_{DD}$) exceeds the high-voltage detection level ($V_{LVDH}$) after power is supplied. The internal reset is released when the internal power supply voltage (internal $V_{DD}$) exceeds the high-voltage detection level ($V_{LVDH}$).

  An interrupt request signal by LVD (INTLVI) is generated and arbitrary save processing is performed when the internal power supply voltage (internal $V_{DD}$) falls below the high-voltage detection level ($V_{LVDH}$). After that, an internal reset by LVD is generated when the internal power supply voltage (internal $V_{DD}$) falls below the low-voltage detection level ($V_{LVDL}$). After INTLVI is generated, an interrupt request signal is not generated even if the supply voltage becomes equal to or higher than the high-voltage detection voltage ($V_{LVDH}$) without falling below the low-voltage detection voltage ($V_{LVDL}$).

  To use the LVD reset & interrupt mode, perform the processing according to **Figure 27-8 Setting Procedure for Operating Voltage Check/Reset** and **Figure 27-9 Initial Setting of Interrupt and Reset Mode**.

Figure 27-7 shows the timing of the internal reset signal and interrupt signal generated in the LVD interrupt & reset mode.

**Figure 27-7. Timing of Voltage Detector Reset Signal and Interrupt Signal Generation (Option Byte LVIMDS1, LVIMDS0 = 1, 0) (1/2)**




(**Notes** and **Remark** are listed on the next page.)

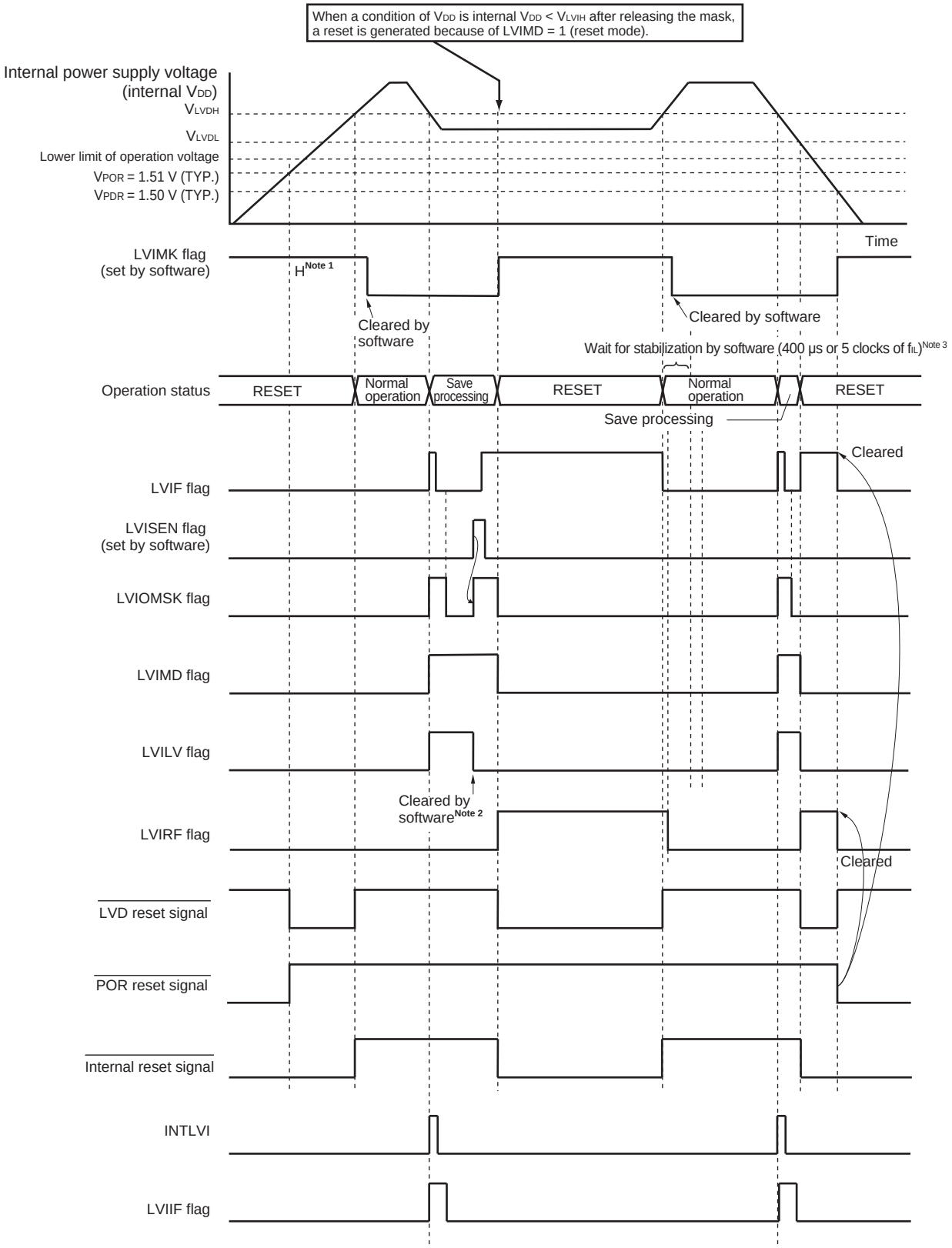
**Notes 1.** The LVIMK flag is set to “1” by reset signal generation.

**2.** After an interrupt is generated, perform the processing according to **Figure 27-8 Setting Procedure for Operating Voltage Check/Reset**.

**Remark** $V_{POR}$: POR power supply rise detection voltage

$V_{PDR}$: POR power supply fall detection voltage

**Figure 27-7. Timing of Voltage Detector Reset Signal and Interrupt Signal Generation (Option Byte LVIMDS1, LVIMDS0 = 1, 0) (2/2)**




(**Notes** and **Remark** are listed on the next page.)

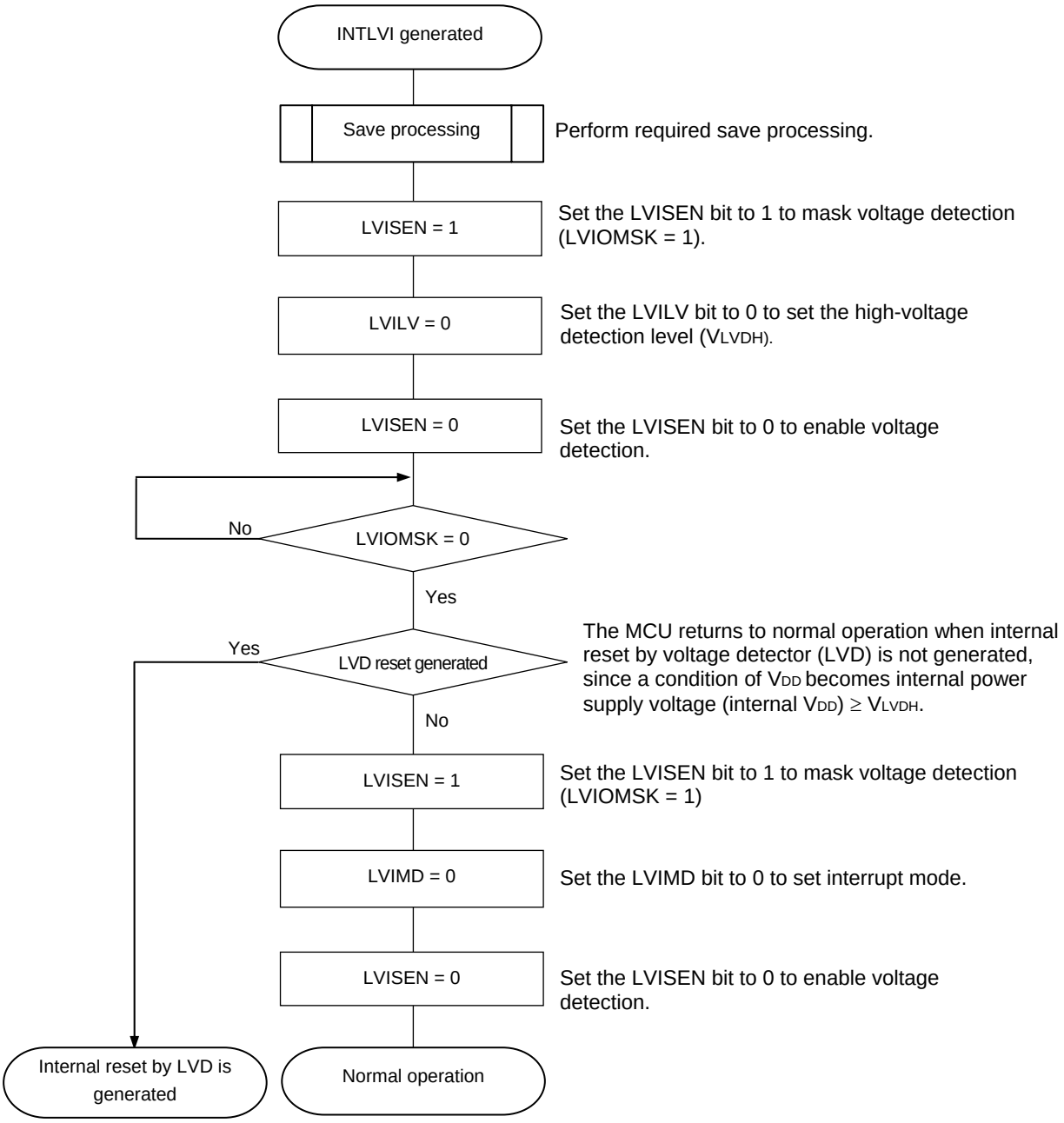
**Notes 1.** The LVIMK flag is set to "1" by reset signal generation.

**2.** After an interrupt is generated, perform the processing according to **Figure 27-8 Setting Procedure for Operating Voltage Check/Reset**.

**Remark** $V_{POR}$: POR power supply rise detection voltage

$V_{PDR}$: POR power supply fall detection voltage

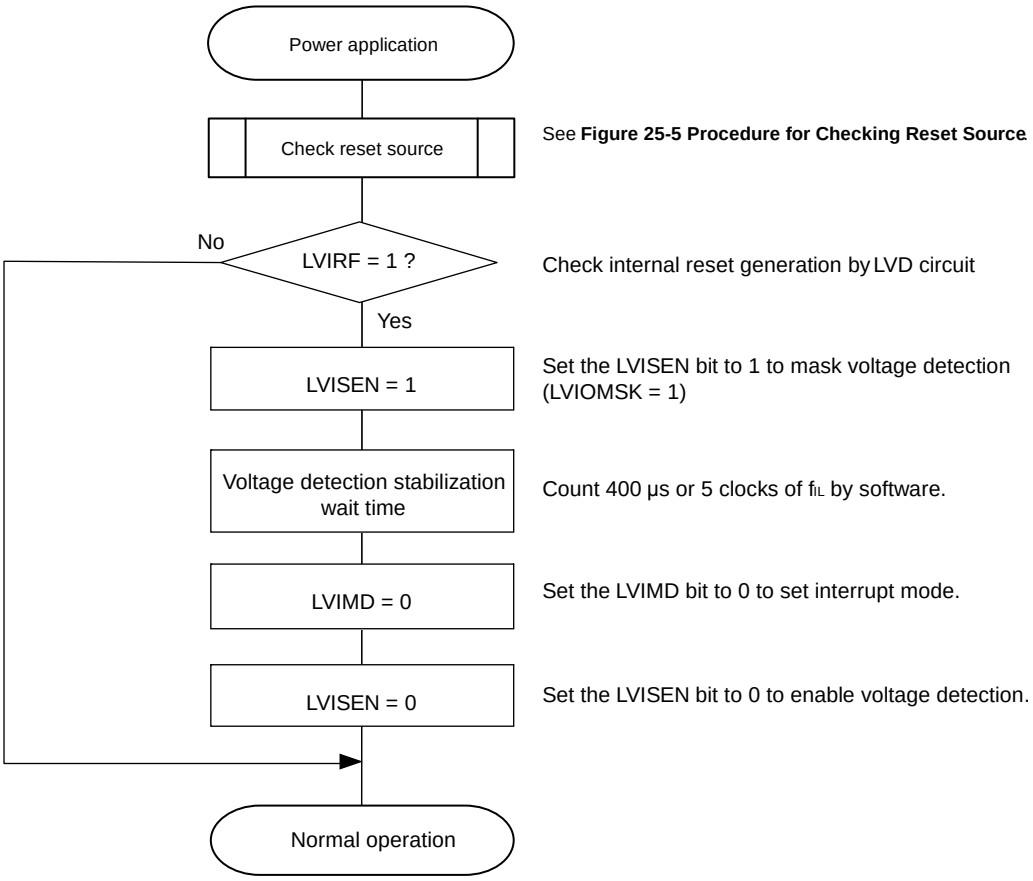
**Figure 27-8. Setting Procedure for Operating Voltage Check/Reset**



When setting an interrupt and reset mode (LVIMDS1, LVIMDS0 = 1, 0), voltage detection stabilization wait time for 400 μs or 5 clocks of $f_{IL}$ is necessary after LVD reset is released (LVIRF = 1). After waiting until voltage detection stabilizes, (0) clear the LVIMD bit for initialization. While voltage detection stabilization wait time is being counted and when the LVIMD bit is rewritten, set LVISEN to 1 to mask a reset or interrupt generation by LVD.

Figure 27-9 shows the procedure for initial setting of interrupt and reset mode.

**Figure 27-9. Initial Setting of Interrupt and Reset Mode**




**Remark** $f_{IL}$: Low-speed on-chip oscillator clock frequency

## 27.5 Cautions for Voltage Detector

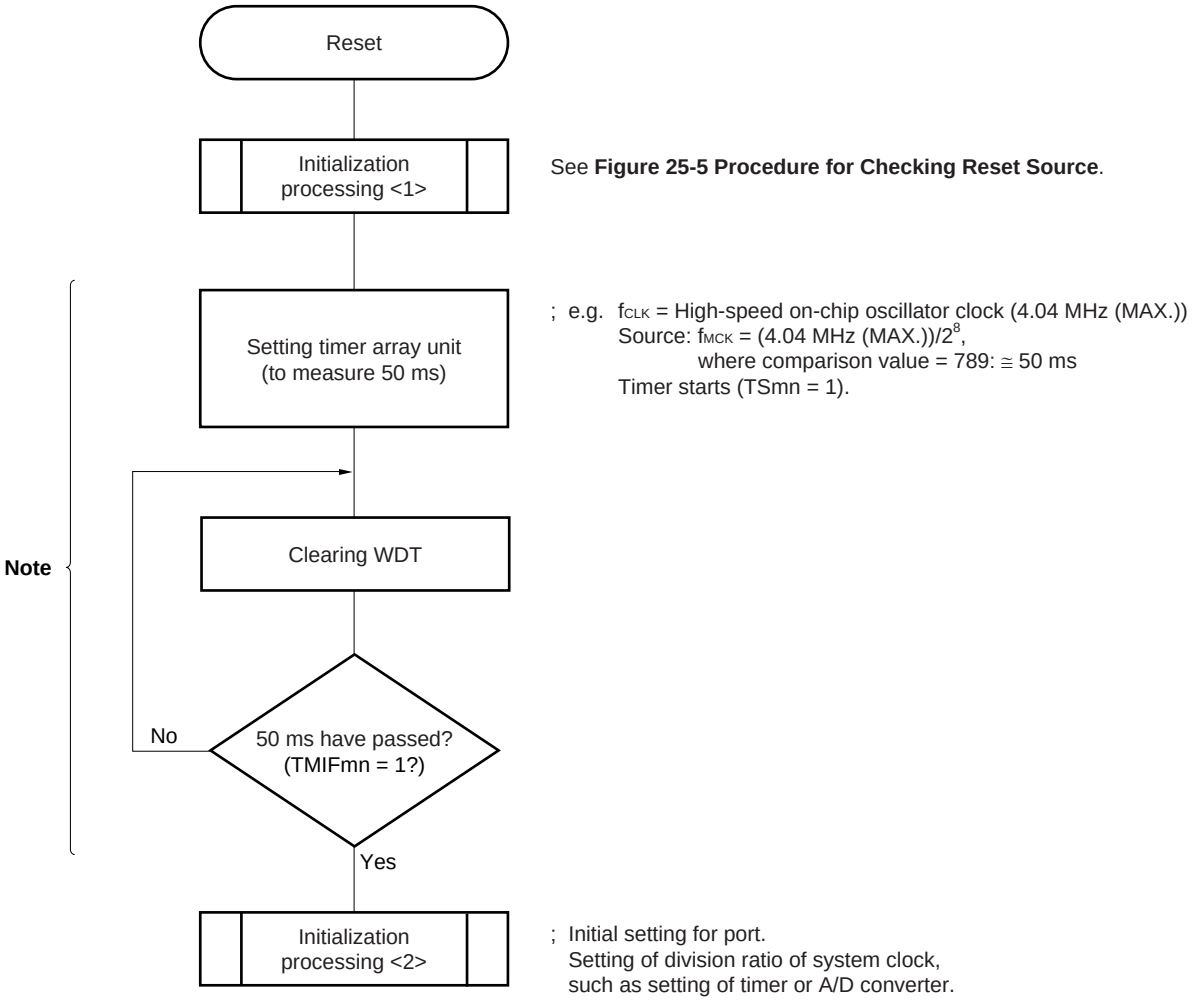
**(1) Voltage fluctuation when power is supplied**

In a system where the internal power supply voltage (internal $V_{DD}$) fluctuates for a certain period in the vicinity of the LVD detection voltage, the system may be repeatedly reset and released from the reset status. 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 internal power supply voltage fluctuation period of each system by means of a software counter that uses a timer, and then initialize the ports.

**Figure 27-10. Example of Software Processing If Internal Power Supply Voltage Fluctuation is 50 ms or Less in Vicinity of LVD Detection Voltage**




**Note** If reset is generated again during this period, initialization processing <2> is not started.

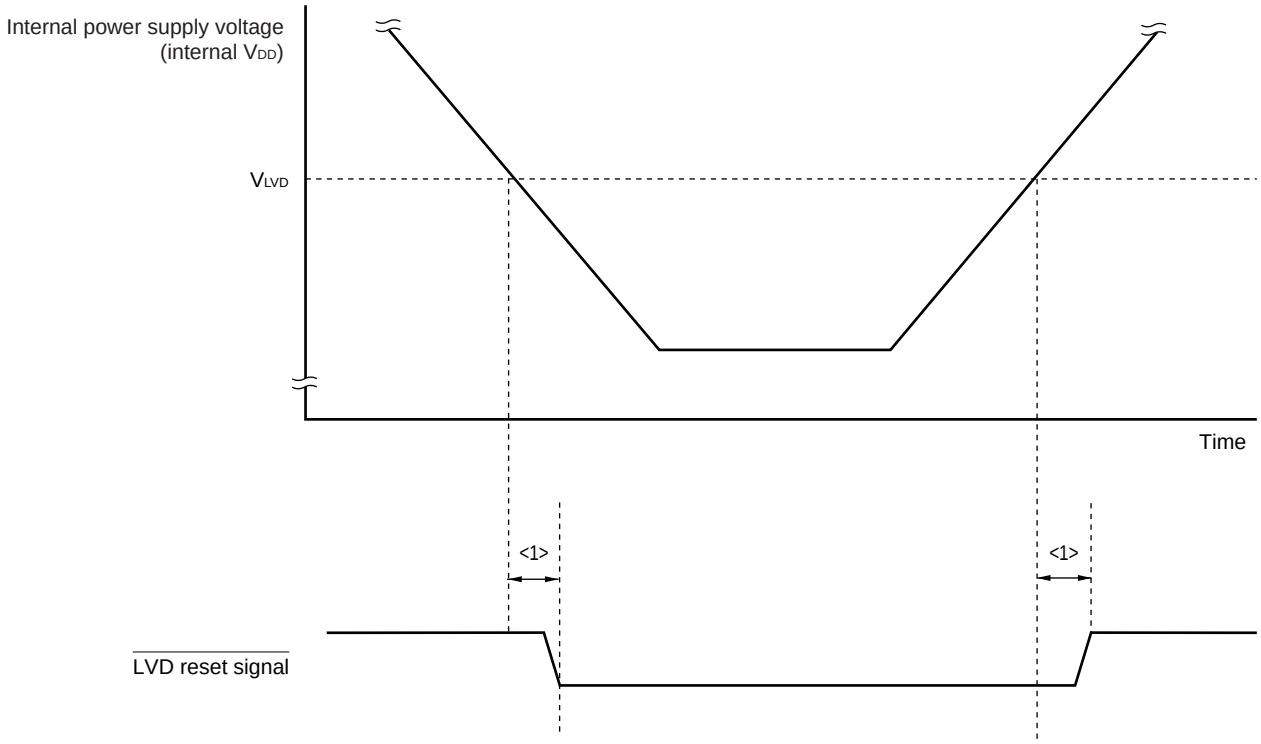
**Remark** m = 0, 1
n = 0 to 7

**(2) Delay from the time LVD reset source is generated until the time LVD reset has been generated or released**

There is some delay from the time internal power supply voltage (internal $V_{DD}$) < LVD detection voltage ($V_{LVD}$) until the time LVD reset has been generated.

In the same way, there is also some delay from the time LVD detection voltage ($V_{LVD}$) ≤ internal power supply voltage (internal $V_{DD}$) until the time LVD reset has been released (see **Figure 27-11**).

**Figure 27-11. Delay from the Time LVD Reset Source Is Generated Until the Time LVD Reset has Been Generated or Released**




<1>: Detection delay (300 μs (MAX.))

**(3) Power on when LVD is off**

Use the external rest input via the $\overline{\text{RESET}}$ pin when the LVD 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 **37.4 AC Characteristics**, and then input a high level to the pin.

**(4) Operating voltage fall when LVD is off or LVD interrupt mode is selected**

When the operating voltage falls with the LVD is off or with the LVD interrupt mode is 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 **37.4 AC characteristics**. When restarting the operation, make sure that the internal power supply voltage has returned within the range of operation.

# CHAPTER 28 BATTERY BACKUP FUNCTION

## 28.1 Functions of Battery Backup

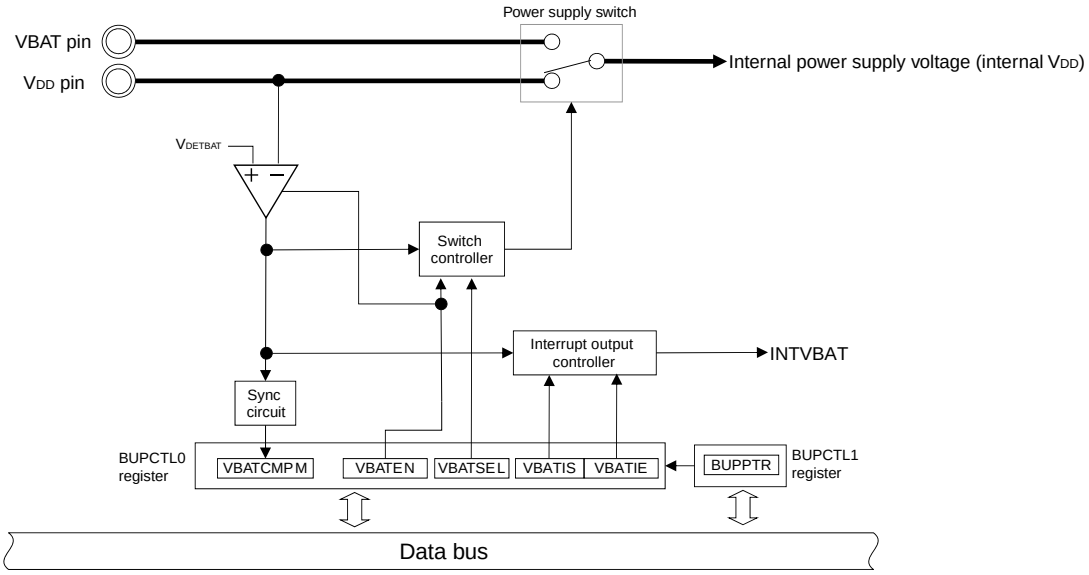
This function monitors the supply voltage at the $V_{DD}$ pin, and switches the internal power supply from the dedicated battery backup power pin (VBAT pin) when the voltage at the $V_{DD}$ pin falls below the detection voltage. The mode used to supply the internal power from the VBAT pin is referred to as battery backup mode. Even if power supply from the $V_{DD}$ pin is cut off due to a power outage, operation of real-time clock 2 (RTC2) can be continued by switching to battery backup mode by hardware. In addition to real-time clock 2 (RTC2), the CPU, the 10-bit A/D converter, the on-chip temperature sensor, the comparator, external interrupts, and $V_{DD}$ power supply system I/O[Note] can be operated in battery backup mode.

- When the voltage at the $V_{DD}$ pin falls to or below the detection voltage, the internal power supply can be switched from $V_{DD}$ supply to VBAT supply. When the voltage at the $V_{DD}$ pin rises to or above the detection voltage again, the internal power supply can be switched from VBAT supply to $V_{DD}$ supply.
- When VBAT ≥ $V_{DD}$, internal power supply can be switched to VBAT by software.
- A power switching detection interrupt (INTVBAT) can be generated when the power is switched. However, no interrupt is generated when the power is switched by software, and an interrupt is generated when the supply voltage at the $V_{DD}$ pin reaches the detection voltage.

**Note** P20 to P25, P121 to P124, P137

Figure 28-1 shows the block diagram of the battery backup function.

**Figure 28-1. Block Diagram of Battery Backup Function**




### 28.1.1 Pin configuration

Table 28-1 lists the pin configuration of battery backup function.

**Table 28-1. Pin Configuration of Battery Backup Function**

| Name | Function |
|---|---|
| $V_{DD}$ | Positive power from the pin |
| VBAT | Power for battery backup |

## 28.2 Registers

Table 28-2 lists the registers used for battery backup.

**Table 28-2. Registers**

| Register Name | Symbol |
|---|---|
| Battery backup power switching control register 0 | BUPCTL0 |
| Battery backup power switching control register 1 | BUPCTL1 |
| Global digital input disable register | GDIDIS |

### 28.2.1 Battery backup power switching control register 0 (BUPCTL0)

The BUPCTL0 register is used to control power switching operation, enable or disable power switching interrupts, and select the power supply pin.

The BUPCTL0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

VBATEN (bit 7) and VBATSEL (bit 0) are cleared to 0 only when a power-on reset is generated. Other bits are cleared to 0 when a reset signal is generated.

**Figure 28-2. Format of Battery Backup Power Switching Control Register 0 (BUPCTL0) (1/2)**

Address: F0330H After reset: 00H[Note 1] R/W

| Symbol | <7> | 6 | 5 | 4 | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| BUPCTL0 | VBATEN | 0 | 0 | 0 | VBATCMPM | VBATIE | VBATIS | VBATSEL |

| VBATEN[Note 2] | Power switching operation control |
|---|---|
| 0 | Power switching function stops[Note 3] |
| 1 | Power switching function operates |

**Notes 1.** VBATEN (bit 7) and VBATSEL (bit 0) are cleared to 0 only when a power-on reset is generated.

**2.** To set the VBATEN bit to 1, write 0 and then write 1 to this bit. If a value is written to an SFR other than BUPCTL0 after 0 has been written, the VBATEN bit cannot be set to 1.
To set the VBATEN bit to 0, write 1 and then write 0 to this bit. If a value is written to an SFR other than BUPCTL0 after 1 has been written, the VBATEN bit cannot be set to 0.

**3.** Prohibits the disable switch power supply function (VBATEN =0 ) while supplying internal power with VBAT. Be sure to check that the VBATCMPM bit is 0 and the internal power supply is $V_{DD}$ before disabling the switch power supply function (VBATEN = 0).

**Figure 28-2. Format of Battery Backup Power Switching Control Register 0 (BUPCTL0) (2/2)**

| VBATCMPM | Power switching comparator output monitor |
|---|---|
| 0 | $V_{DD}$ pin voltage ≥ power switching detection voltage ($V_{DETBAT2}$)<br>or power switching function stopped (VBATEN = 0) |
| 1 | $V_{DD}$ pin voltage < power switching detection voltage ($V_{DETBAT1}$) |

| VBATIE | Power switching interrupt control |
|---|---|
| 0 | Interrupt generation disabled |
| 1 | Interrupt generation enabled |

| VBATIS | Power switching interrupt selection |
|---|---|
| 0 | Interrupt signal generated when $V_{DD}$ pin voltage < power switching detection voltage ($V_{DETBAT1}$)<br>Interrupt generated when $V_{DD}$ is switched to VBAT[Note] |
| 1 | Interrupt signal generated when $V_{DD}$ pin voltage ≥ power switching detection voltage ($V_{DETBAT2}$)<br>Interrupt generated when VBAT is switched to $V_{DD}$[Note] |

**Note** No interrupt is generated when the power is switched by VBATSEL.

| VBATSEL[Note] | Power supply pin selection |
|---|---|
| 0 | The supply source is switched by hardware depending on the potential of $V_{DD}$ pin. |
| 1 | Power is supplied from VBAT pin. |

**Note** To set the VBATSEL bit to 1, write 0 and then write 1 to this bit. If a value is written to an SFR other than BUPCTL0 after 0 has been written, the VBATSEL bit cannot be set to 1.
To set the VBATSEL bit to 0, write 1 and then write 0 to this bit. If a value is written to an SFR other than BUPCTL0 after 1 has been written, the VBATSEL bit cannot be set to 0.

**Cautions 1. Setting VBATSEL = 1 is prohibited when $V_{DD}$ > VBAT.**
**2. Be sure to clear bits 6 to 4 to 0.**

### 28.2.2 Battery backup power switching control register 1 (BUPCTL1)

The BUPCTL1 register is used to disable or enable rewriting of the BUPCTL0 register. Since rewriting of the BUPCTL0 register is disabled when the BUPPRT bit is 0, the BUPCTL0 register can be prevented from being written inadvertently.

The BUPCTL1 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 28-3. Format of Battery Backup Power Switching Control Register 1 (BUPCTL1)**

Address: F0331H After reset: 00H R/W

| Symbol | <7> | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| BUPCTL1 | BUPPRT | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| BUPPRT | BUPCTL0 register write protection control |
|---|---|
| 0 | The BUPCTL0 register cannot be written, but it can be read. |
| 1 | The BUPCTL0 register can be written and read. |

**Caution Be sure to clear bits 6 to 0 to 0.**

### 28.2.3 Global digital input disable register (GDIDIS)

When $EV_{DD}$ and $V_{DD}$ are used at the same potential, if power supply from the $V_{DD}$ pin is stopped due to power outage, $EV_{DD}$ supply will also stop and drop to 0 V. The GDIDIS register prevents through-current to the input buffer when $EV_{DD}$ = 0 V. Setting the GDIDIS0 bit to 1 disables input to all input buffers[Note] connected to $EV_{DD}$, and prevents shoot-through current when the power connected to $EV_{DD}$ is turned off. When using the GDIDIS register, set GDIDIS0 to 1 before turning off the power for $EV_{DD}$, and then set GDIDIS0 to 0 after turning on the power for $EV_{DD}$.

The GDIDIS register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Note** Port pin other than P20 to P25, P121 to P124, and P137.
Because the power supply of the I/O buffer switches to $V_{DD}$ or VBAT pin with the battery backup function, I/O of P20 to P25, P121 to P124, and P137 can be used even when GDIDIS is set to 1.
See **Table 2-1 Pin I/O Buffer Power Supplies** for the I/O buffer power of the pins.

**Figure 28-4. Format of Global Digital Input Disable Register (GDIDIS)**

Address: F007DH After reset: 00H 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 input buffers permitted (default) |
| 1 | Input to input buffers prohibited. No through-current flows to the input buffers. |

## 28.3 Operation

### 28.3.1 Battery backup function

When the voltage from the $V_{DD}$ pin falls below the detection voltage, the power supply from the dedicated battery backup power pin (VBAT pin) can be switched to the internal power supply. When the voltage supplied from the $V_{DD}$ pin falls below the detection voltage ($V_{DETBAT1}$), the internal power is switched from $V_{DD}$ supply to VBAT supply.

At power on, the internal power is fixed to be always supplied from the $V_{DD}$ pin. When a power-on reset is generated, the VBATEN bit in the BUPCTL0 register is reset to 0. When the VBATEN bit in the BUPCTL0 register is 0, the power switching function is stopped, and the internal power is supplied from the $V_{DD}$ pin. When the VBATEN bit in the BUPCTL0 register is set to 1, the power switching function operates. When the power switching function is operating, the internal power supply is switched from VBAT supply to $V_{DD}$ supply when the $V_{DD}$ voltage rises to or above the detection voltage ($V_{DETBAT2}$) again while the power is supplied from the VBAT pin.

In addition, the power supply from the $V_{DD}$ pin can be switched to the power supply from the VBAT pin by software. When the VBATEN bit in the BUPCTL0 register is 1 (power switching function operates), the power supply is switched to the power supply from the VBAT pin by setting the VBATSEL bit in the BUPCTL0 register to 1 (power is supplied from VBAT).

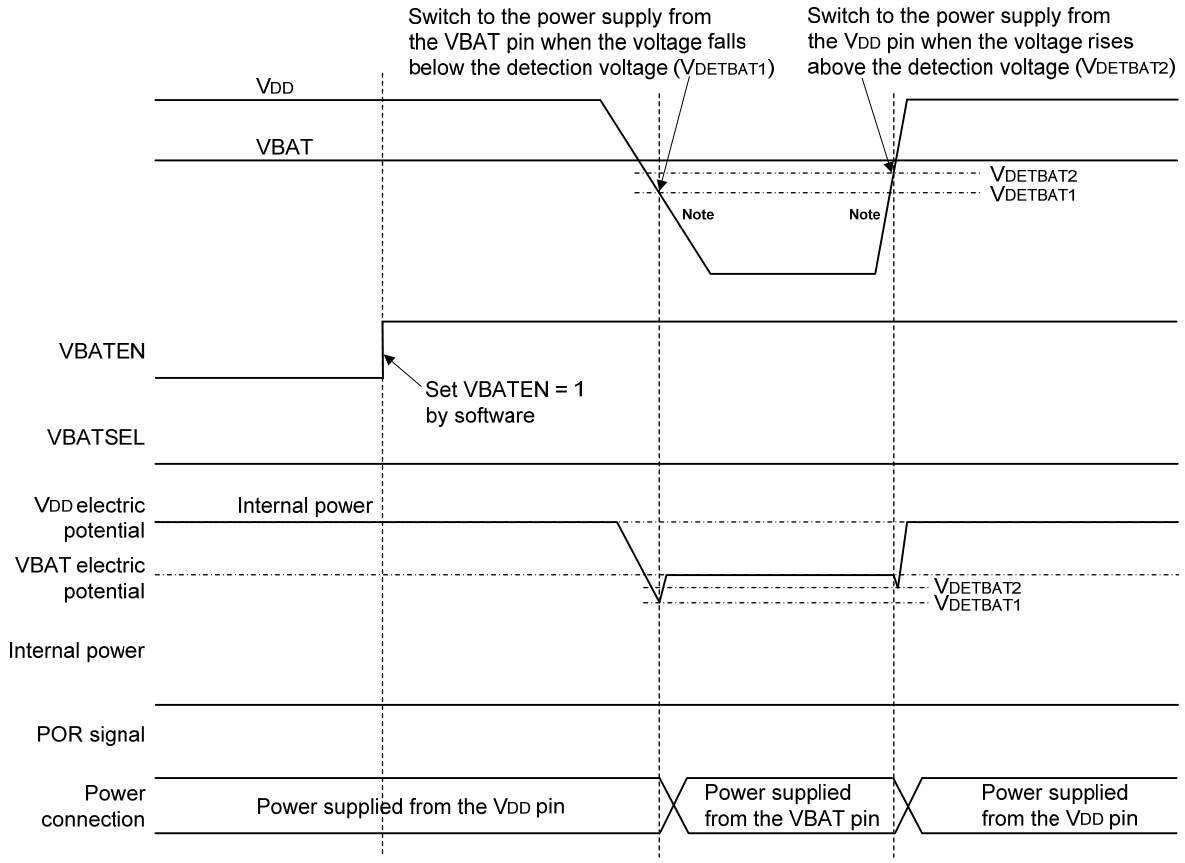
Table 28-3 lists the specifications of battery backup operation and Figure 28-5, Figure 28-6 show battery backup operation.

**Table 28-3. Specifications of Battery Backup Operation**

| Power | VBATEN | VBATSEL | Condition | Internal Power Connection |
|---|---|---|---|---|
| At power on | × | × | – | Power supplied from the $V_{DD}$ pin |
| After power on | 0 | × | – | Power supplied from the $V_{DD}$ pin |
| | 1 | 0 | $V_{DD} \geq V_{DETBAT2}$ | Power supplied from the $V_{DD}$ pin |
| | | | $V_{DETBAT1} < V_{DD} < V_{DETBAT2}$ | Power supplied from the $V_{DD}$ pin<br>or power supplied from the VBAT pin<br>(Has hysteretic characteristics) |
| | | | $V_{DD} \leq V_{DETBAT1}$ | Power supplied from the VBAT pin |
| | | 1 | – | Power supplied from the VBAT pin |

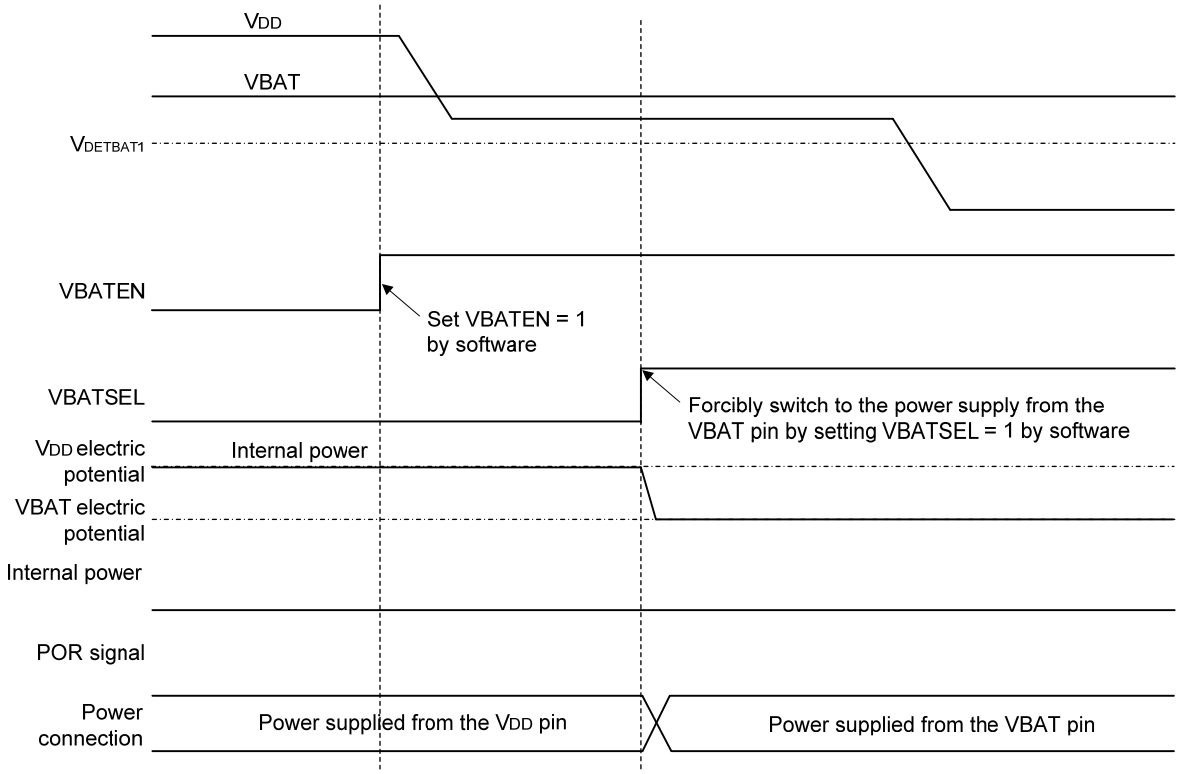
**Remark** ×: Don't care

**Figure 28-5. Battery Backup Operation (1) with VBATEN = 1 and VBATSEL = 0**




**Note** For details about the power rising and falling slopes, see **CHAPTER 37 ELECTRICAL SPECIFICATIONS**.

**Figure 28-6. Battery Backup Operation (2) with VBATEN = 1 and VBATSEL = 1**




## 28.4 Usage Notes

(1) When not using the battery backup function, connect the VBAT and Vss pins to the same potential.

(2) Setting VBATSEL = 1 is prohibited when $V_{DD}$ > VBAT.

(3) Be sure VBAT does not drop below 1.9 V when VBATSEL = 1.

(4) Do not set VBATEN and VBATSEL at the same time.

(5) Do not set VBATEN to 0 while VBATSEL is 1.

(6) For details about the power rising and falling slopes, see **CHAPTER 37 ELECTRICAL SPECIFICATIONS**.

(7) The self-programming function cannot be used when the internal power is supplied from the VBAT pin.

(8) The on-chip debug function cannot be used when the internal power is supplied from the VBAT pin.

(9) When switching the power supply by hardware (VBATEN = 1, VBATSEL = 0), disable the input buffer with the GDIDIS register (GDIDIS = 01H) to prevent leak current at the $EV_{DD}$ port pin when the power is switched to VBAT.

(10) When switching the power supply by hardware (VBATEN = 1, VBATSEL = 0), input signal must be designed so that it does not exceed the $EV_{DD}$ voltage because the input buffer of the $EV_{DD}$ port pin is controlled by the $EV_{DD}$ voltage when the power is switched to VBAT.

(11) Prohibits the disable switch power supply function (VBATEN = 0) while supplying internal power with VBAT. Be sure to check that the VBATCMPM bit is 0 and the internal power supply is $V_{DD}$ before disabling the switch power supply function (VBATEN = 0).

# CHAPTER 29 OSCILLATION STOP DETECTOR

## 29.1 Functions of Oscillation Stop Detector

The oscillation stop detection circuit monitors the subsystem clock ($f_{SUB}$) operating status with a low-speed on-chip oscillator clock ($f_{IL}$). If it detects that operation is stopped longer than a predefined interval, it assumes that an XT1 oscillator circuit error has occurred and outputs an oscillation stop interrupt signal.

When the system is reset, operation of the oscillation stop detector must be enabled by software after the reset period ends.

Operation of the oscillation stop detector is stopped by software. Or, oscillation stop detection operation is stopped by reset from the $\overline{\text{RESET}}$ pin or internal reset due to execution of an invalid instruction[Note]. Furthermore, since the oscillation of XT1 oscillator clock is also stopped with an internal reset, after a reset, enable oscillation stop detection operation after resuming oscillation of the XT1 oscillation clock with software.

**Note** Occurs when instruction code for FFH is executed.
Reset due to invalid instruction does not occur during emulation with in-circuit emulator or on-chip debug emulator.

The period used by the oscillation stop detector to judge that oscillation is stopped (oscillation stop judgment time) can be set by using the OSDCCMP11 to OSDCCMP0 bits of the oscillation stop detection control register (OSDC).

Oscillation stop judgment time = Low-speed on-chip oscillator clock ($f_{IL}$) cycle × ((value of OSDCCMP11 to OSDCCMP0) + 1)

- OSDCCMP11 to OSDCCMP0 = 003H: 232 μs (MIN.), 267 μs (TYP.), 314 μs (MAX.)
- OSDCCMP11 to OSDCCMP0 = FFFH: 237 ms (MIN.), 273 ms (TYP.), 322 ms (MAX.)

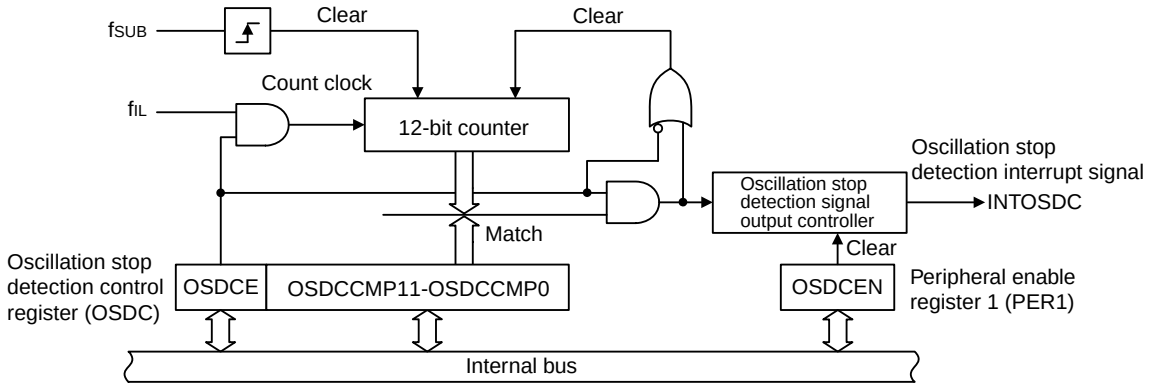
## 29.2 Configuration of Oscillation Stop Detector

The oscillation stop detector includes the following hardware.

**Table 29-1. Configuration of Oscillation Stop Detector**

| Item | Configuration |
|---|---|
| Control registers | Peripheral enable register 1 (PER1)<br>Subsystem clock supply mode control register (OSMC)<br>Oscillation stop detection control register (OSDC) |

**Figure 29-1. Block Diagram of Oscillation Stop Detector**



## 29.3 Registers Used by Oscillation Stop Detector

### 29.3.1 Peripheral enable register 1 (PER1)

This register is used to enable or disable clock supply to the peripheral hardware. Use this register to stop clock supply to unused hardware to reduce power consumption and noise.

When using the oscillation stop detector, be sure to set bit 0 (OSDCEN) to 1.

The PER1 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 29-2. Format of Peripheral Enable Register 1 (PER1)**

Address: F007AH After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PER1 | TMKAEN | FMCEN | CMPEN | OSDCEN | DTCEN | 0 | 0 | DSADCEN |

| OSDCEN | Control of oscillation stop detector input clock supply |
|---|---|
| 0 | Stop supplying the input clock.<br>• SFRs used by the oscillation stop detector cannot be written.<br>• The oscillation stop detector is in the reset status. |
| 1 | Enable the input clock supply.<br>• SFRs used by the oscillation stop detector can be read and written. |

**Cautions 1. When using the oscillation stop detector, be sure to set the OSDCEN bit to 1. If OSDCEN = 0, writing to a control register of the oscillation stop detector is ignored, and, even if the register is read, only the default value is read.**

**2. Be sure to set bits 2 and 1 to “0”.**

### 29.3.2 Subsystem clock supply mode control register (OSMC)

This register is used to reduce power consumption by stopping unnecessary clock functions.

If the RTCLPC bit is set to 1, power consumption can be reduced, because clock supply to the peripheral functions other than real-time clock 2, 12-bit interval timer, clock output/buzzer output controller, LCD controller/driver, 8-bit interval timer, and oscillation stop detector is stopped in STOP mode or in HALT mode while the subsystem clock is selected as the CPU clock.

In addition, the OSMC register is used to select the operation clock of real-time clock 2, 12-bit interval timer, clock output/buzzer output controller, LCD controller/driver, 8-bit interval timer, and subsystem clock frequency measurement function.

The OSMC register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 29-3. Format of Subsystem Clock Supply Mode Control Register (OSMC)**

Address: F00F3H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| OSMC | RTCLPC | 0 | 0 | WUTMMCK0 | 0 | 0 | 0 | 0 |

| RTCLPC | In STOP mode and in HALT mode while the CPU operates using the subsystem clock |
|---|---|
| 0 | Enables subsystem clock supply to peripheral functions.<br>For peripheral functions for which operation is enabled, see **Tables 24-1** and **24-2**. |
| 1 | Stops subsystem clock supply to peripheral functions other than real-time clock 2, 12-bit interval timer, clock output/buzzer output controller, LCD controller/driver, 8-bit interval timer, and oscillation stop detector. |

| WUTMMCK0 **Notes 1, 2, 3** | Selection of operation clock for real-time clock 2, 12-bit interval timer, and LCD controller/driver. | Selection of clock output from PCLBUZn pin of clock output/buzzer output controller and selection of operation clock for 8-bit interval timer. | Operation of subsystem clock frequency measurement circuit. |
|---|---|---|---|
| 0 | Subsystem clock ($f_{SUB}$) | Selecting the subsystem clock ($f_{SUB}$) is enabled. | Enable |
| 1 | Low-speed on-chip oscillator clock ($f_{IL}$) | Selecting the subsystem clock ($f_{SUB}$) is disabled. | Disable |

**Notes 1.** The $f_{IL}$ clock can be selected (WUTMMCK0 = 1) only when oscillation of the subsystem clock is stopped (the XTSTOP bit in the CSC register = 1).

**2.** When WUTMMCK0 is set to 1, the low-speed on-chip oscillator clock oscillates.

**3.** When WUTMMCK0 is set to 1, the 1 Hz output function of real-time clock 2 cannot be used.

**Caution The count of year, month, week, day, hour, minutes and second can only be performed when a subsystem clock ($f_{SUB}$ = 32.768 kHz) is selected as the operation clock of the real-time clock.**

**When the low-speed oscillation clock ($f_{IL}$ = 15 kHz) is selected, only the constant-period interrupt function is available.**

**However, the constant-period interrupt interval when $f_{IL}$ is selected will be calculated with the constant-period (the value selected with RTCC0 register) × 1/$f_{IL}$.**

### 29.3.3 Oscillation stop detection control register (OSDC)

This register is used to control the oscillation stop detector. Use this register to start and stop operation of the oscillation stop detector. This register can also be used to specify the oscillation stop judgment time.

Operation of the oscillation stop detector cannot be started while the OSDCE bit is 0.

The OSDC register can be set by a 16-bit memory manipulation instruction.

Reset signal generation clears this register to 0FFFH.

**Figure 29-4. Format of Oscillation Stop Detection Control Register (OSDC)**

Address: F0390H After reset: 0FFFH R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| OSDC | OSDCE | 0 | 0 | 0 | OSDCCMP11 | OSDCCMP10 | OSDCCMP9 | OSDCCMP8 |

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| OSDC | OSDCCMP7 | OSDCCMP6 | OSDCCMP5 | OSDCCMP4 | OSDCCMP3 | OSDCCMP2 | OSDCCMP1 | OSDCCMP0 |

| OSDCE | Control of oscillation stop detector operation |
|---|---|
| 0 | Stop operation of the oscillation stop detector. |
| 1 | Start operation of the oscillation stop detector. |

| OSDCCMP11 to OSDCCMP0 | Oscillation stop judgment time |
|---|---|
| 000H<br>...<br>002H | Setting prohibited |
| 003H<br>...<br>FFFH | These bits specify the oscillation stop judgment time.<br>It is judged that oscillation has stopped when oscillation has been stopped for (A-2) to (A+1) clock cycles, where A refers to the time specified by these bits.<br>Oscillation stop judgment time = Low-speed on-chip oscillator clock ($f_{IL}$) cycle × ((value of OSDCCMP11 to OSDCCMP0) + 1) |

**Cautions 1. Be sure to set the OSDCE bit to "0" (to stop operation of the oscillation stop detector) before changing the setting of the OSDCCMP11 to OSDCCMP0 bits.**

**2. The oscillation stop detector stops oscillation stop detection by setting the OSDCE bit to 0 by software or by reset from the $\overline{\text{RESET}}$ pin or internal reset due to execution of an invalid instruction[Note].**

**Furthermore, since the oscillation of XT1 oscillator clock is also stopped with an internal reset, after a reset, enable oscillation stop detection operation after resuming oscillation of the XT1 oscillation clock with software.**

**3. Be sure to set bits 14 to 12 to "0".**

**Note** Occurs when instruction code for FFH is executed.

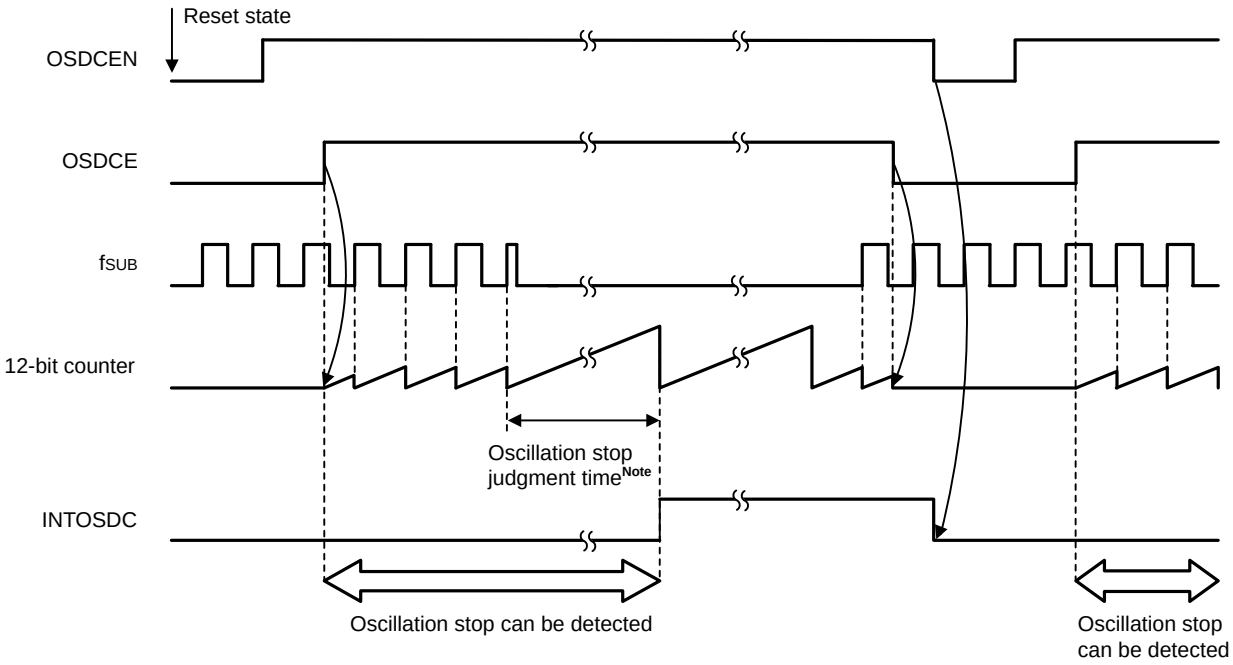
Reset due to invalid instruction does not occur during emulation with in-circuit emulator or on-chip debug emulator.

## 29.4 Operation of Oscillation Stop Detector

### 29.4.1 How the oscillation stop detector operates

1. The subsystem clock starts operating after the external reset ends.
2. A value is written to the oscillation stop detection control register (OSDC) and the oscillation stop detector starts operating.
3. While the oscillation stop detector is operating, if the subsystem clock ($f_{SUB}$) stops oscillating continuously for a period equal to the oscillation stop judgment time or longer, the oscillation stop detector outputs the oscillation stop detection interrupt signal (INTOSDC).

**Figure 29-5. Timing of Oscillation Stop Detection by Oscillation Stop Detector**




**Note** It is judged that oscillation has stopped when oscillation has been stopped for (A-2) to (A+1) clock cycles, where A refers to the time specified by these bits.

## 29.5 Cautions on Using the Oscillation Stop Detector

The oscillation stop detector should be used in conjunction with the watchdog timer.
Oscillation stop detection can be used under either of the following conditions:

- When bit 0 (WDSTBYON) and bit 4 (WDTON) of the option byte (000C0H) are set to 1 and bit 4 (WUTMMCK0) of the OSMC register is set to 0
- When bit 4 (WUTMMCK0) of the OSMC register is set to 1

# CHAPTER 30 SAFETY FUNCTIONS

## 30.1 Overview of Safety Functions

The following safety functions are provided in the RL78/I1B to comply with the IEC60730 and IEC61508 safety standards.

These functions enable the microcontroller to self-diagnose abnormalities and stop operating if an abnormality is detected.

**(1) Flash memory CRC operation function (high-speed CRC, general-purpose CRC)**

This detects data errors in the flash memory by performing CRC operations.

Two CRC functions are provided in the RL78/I1B that can be used according to 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 CRC: This can be used for checking various data in addition to the code flash memory area while the CPU is running.

**(2) RAM parity error detection function**

This detects parity errors when the RAM is read as data.

**(3) RAM guard function**

This prevents RAM data from being rewritten when the CPU freezes.

**(4) SFR guard function**

This prevents SFRs from being rewritten when the CPU freezes.

**(5) Invalid memory access detection function**

This detects illegal accesses to invalid memory areas (such as areas where no memory is allocated and areas to which access is restricted).

**(6) Frequency detection function**

This function allows a self-check of the CPU/peripheral hardware clock frequencies using the timer array unit.

**(7) A/D test function**

This is used to perform a self-check of the A/D converter by performing A/D conversion of the A/D converter's positive and negative reference voltages, analog input channel (ANI), temperature sensor output voltage, and internal reference voltage.

**(8) Digital output signal level detection function for I/O pins**

When the I/O pins are output mode, the output level of the pin can be read.

**Remark** For usage examples of the safety functions complying with the IEC60730 safety standards, refer to the **RL78 MCU series IEC60730/60335 self test library application note (R01AN1062, R01AN1296)**.

## 30.2 Registers Used by Safety Functions

The safety functions use the following registers for each function.

| Register | Each Function of Safety Function |
|---|---|
| • Flash memory CRC control register (CRC0CTL)<br>• Flash memory CRC operation result register (PGCRCL) | Flash memory CRC operation function (high-speed CRC) |
| • CRC input register (CRCIN)<br>• CRC data register (CRCD) | CRC operation function (general-purpose CRC) |
| • RAM parity error control register (RPECTL) | RAM parity error detection function |
| • Invalid memory access detection control register (IAWCTL) | RAM guard function |
| | SFR guard function |
| | Invalid memory access detection function |
| • Timer input select register 0 (TIS0) | Frequency detection function |
| • A/D test register (ADTES) | A/D test function |
| • Port mode select register (PMS) | Digital output signal level detection function for I/O ports |

The content of each register is described in **30.3 Operation of Safety Functions**.

## 30.3 Operation of Safety Functions

### 30.3.1 Flash memory CRC operation function (high-speed CRC)

The IEC60730 standard mandates the checking of data in the flash memory, and recommends using CRC to do it. The high-speed CRC provided in the RL78/I1B 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, 341 μs@24 MHz with 32 KB 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 CRC, because the general CRC operates in LSB first order.

### 30.3.1.1 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.

Reset signal generation clears this register to 00H.

**Figure 30-1. Format of Flash Memory CRC Control Register (CRC0CTL)**

Address: F02F0H After reset: 00H R/W

| Symbol | <7> | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CRC0CTL | CRC0EN | 0 | 0 | 0 | 0 | FEA2[Note] | FEA1 | FEA0 |

| CRC0EN | Control of CRC ALU operation |
|---|---|
| 0 | Stop the operation. |
| 1 | Start the operation according to HALT instruction execution. |

| FEA2[Note] | FEA1 | FEA0 | High-speed CRC operation range |
|---|---|---|---|
| 0 | 0 | 0 | 0000H to 3FFBH (16 K-4 bytes) |
| 0 | 0 | 1 | 0000H to 7FFBH (32 K-4 bytes) |
| 0 | 1 | 0 | 0000H to BFFBH (48 K-4 bytes) |
| 0 | 1 | 1 | 0000H to FFFBH (64 K-4 bytes) |
| 1 | 0 | 0 | 00000H to 13FFBH (80 K-4 bytes) |
| 1 | 0 | 1 | 00000H to 17FFBH (96 K-4 bytes) |
| 1 | 1 | 0 | 00000H to 1BFFBH (112 K-4 bytes) |
| 1 | 1 | 1 | 00000H to 1FFFBH (128 K-4 bytes) |

**Note** Be sure to set FEA2 bit to “0” on R5F10MME and R5F10MPE.

**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.

#### 30.3.1.2 Flash memory CRC operation result register (PGCRCL)

This register is used to store the high-speed CRC operation results.

The PGCRCL register can be set by a 16-bit memory manipulation instruction.

Reset signal generation clears this register to 0000H.

**Figure 30-2. Format of Flash Memory CRC Operation Result Register (PGCRCL)**

Address: F02F2H After reset: 0000H 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 |

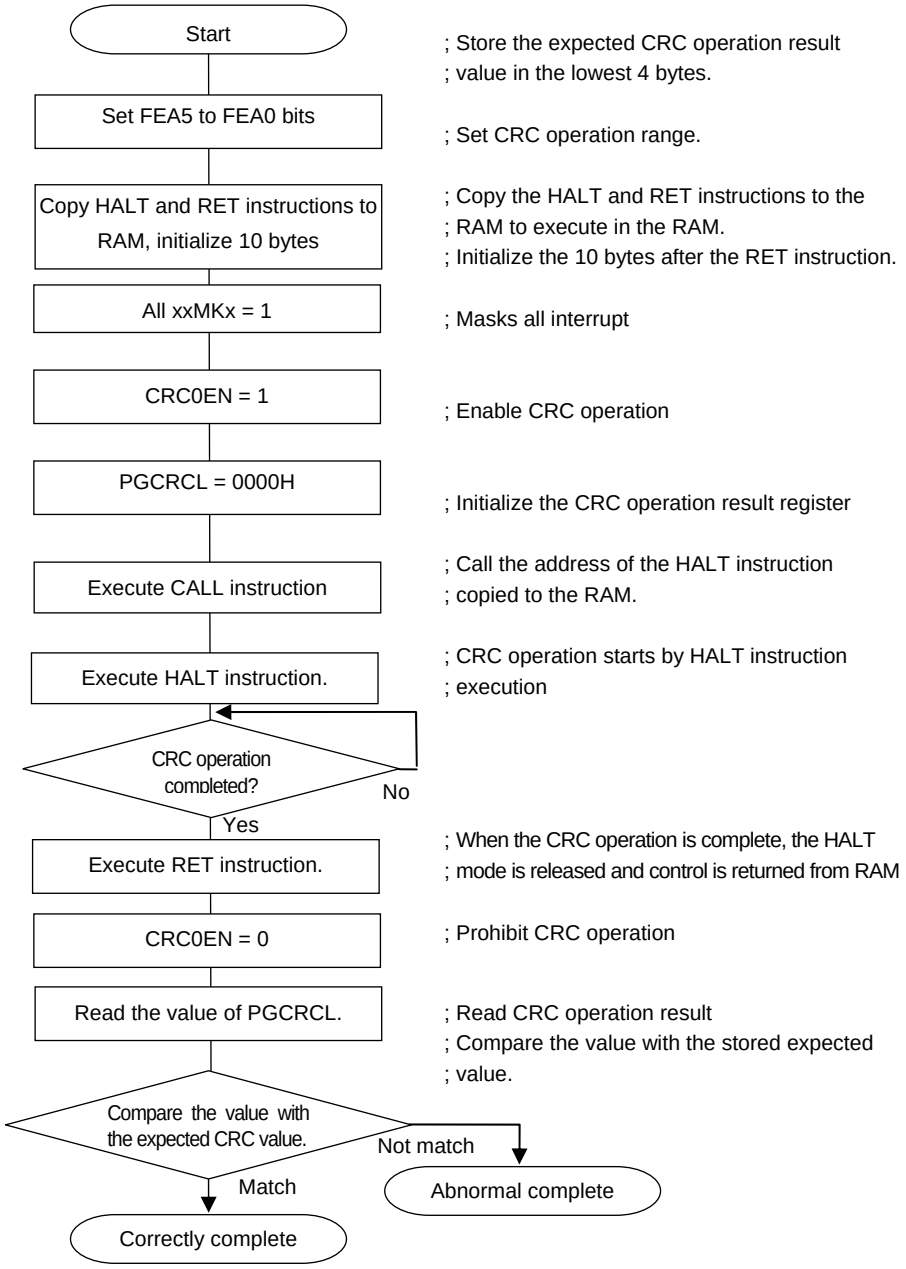
| | |
|---|---|
| PGCRC15 to 0 | High-speed CRC operation results |
| 0000H to FFFFH | Store the high-speed CRC operation results. |

**Caution The PGCRCL register can only be written if CRC0EN (bit 7 of the CRC0CTL register) = 1.**

Figure 30-3 shows the flowchart of flash memory CRC operation function (high-speed CRC).

**<Operation flow>**

**Figure 30-3. Flowchart of Flash Memory CRC Operation Function (High-speed CRC)**




**Cautions 1. The CRC operation is executed only on the code flash.**

**2. Store the expected CRC operation value in the area below the operation range in the code flash.**

**3. The CRC operation is enabled by executing the HALT instruction in the RAM area.**
**Be sure to execute the HALT instruction in RAM area.**

The expected CRC value can be calculated by using the Integrated Development Environment CubeSuite+. See the Integrated Development Environment CubeSuite+ user's manual for details.

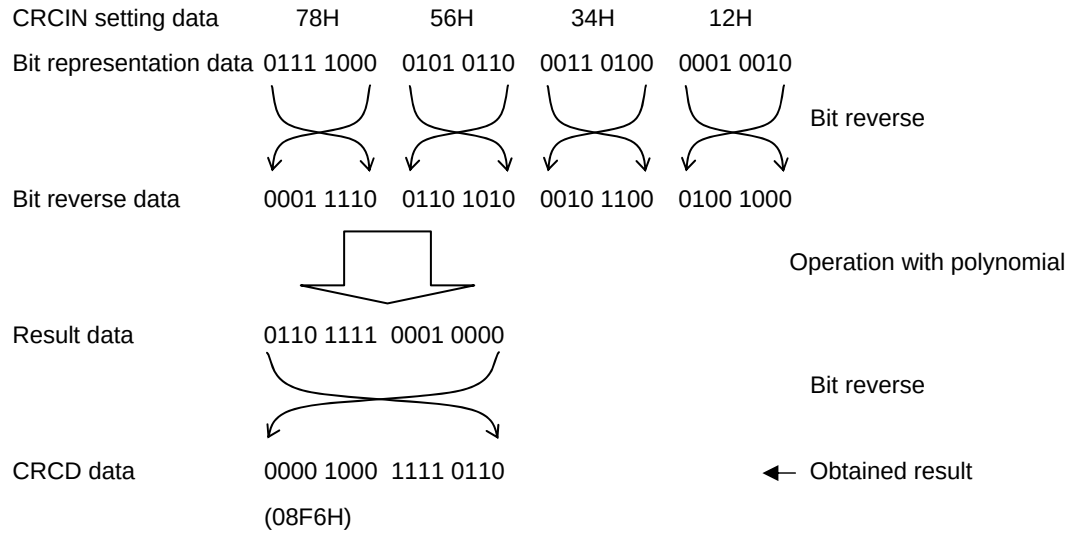
### 30.3.2 CRC operation function (general-purpose CRC)

In order to guarantee safety during operation, the IEC61508 standard mandates the checking of data even while the CPU is operating.

In the RL78/I1B, a general CRC operation can be executed as a peripheral function while the CPU is operating. The general 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). CRC calculation function in the HALT mode can be used only during the DTC transmission.

The general CRC operation can be executed in the main system clock operation mode as well as 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.**

#### 30.3.2.1 CRC input register (CRCIN)

CRCIN register is an 8-bit register that is used to set the CRC operation data of general-purpose CRC.

The possible setting range is 00H to FFH.

The CRCIN register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 30-4. Format of CRC Input Register (CRCIN)**

Address: FFFACH After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CRCIN | | | | | | | | |

| Bits 7 to 0 | Function |
|---|---|
| 00H to FFH | Data input. |

#### 30.3.2.2 CRC data register (CRCD)

This register is used to store the CRC operation result of the general-purpose CRC.

The setting range is 0000H to FFFFH.

After 1 clock of CPU/peripheral hardware clock ($f_{CLK}$) has elapsed from the time CRCIN register is written, the CRC operation result is stored to the CRCD register.

The CRCD register can be set by a 16-bit memory manipulation instruction.

Reset signal generation clears this register to 0000H.

**Figure 30-5. Format of CRC Data Register (CRCD)**

Address: F02FAH After reset: 0000H R/W

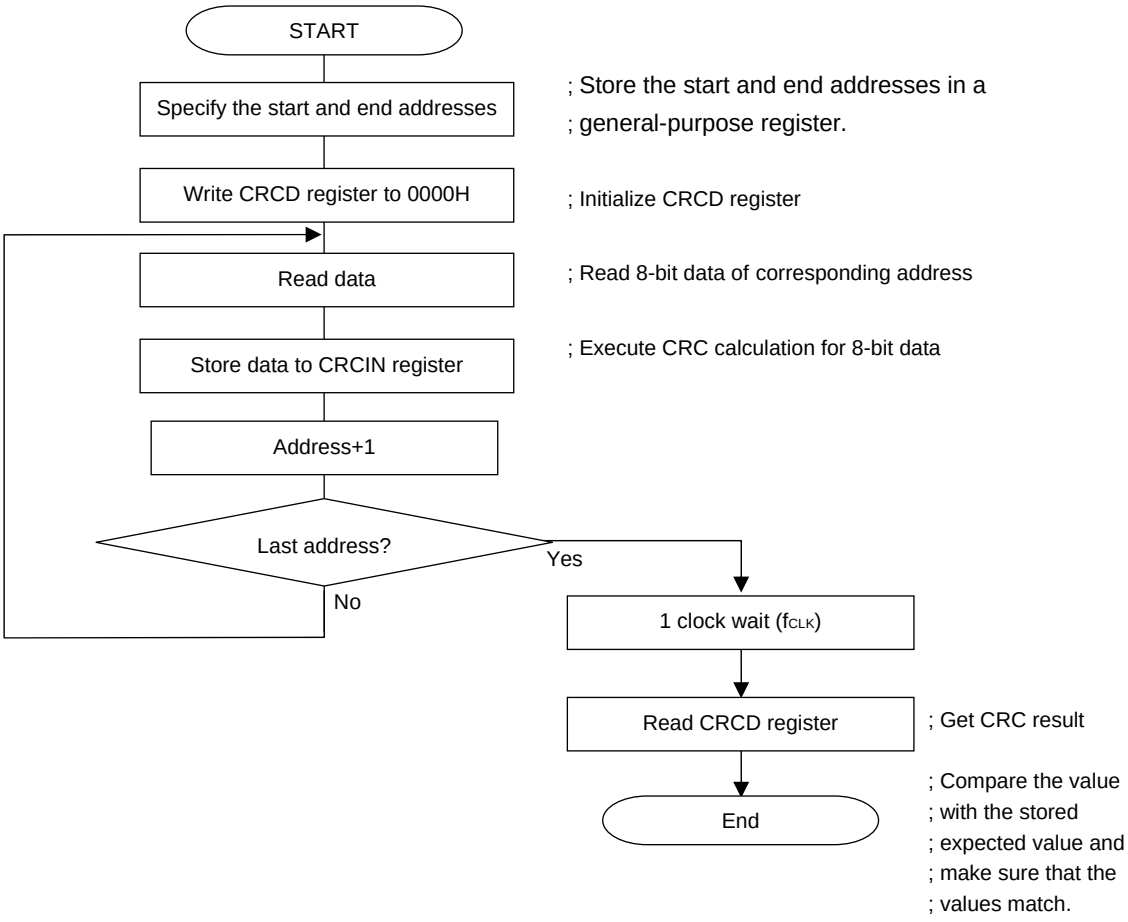
| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CRCD | | | | | | | | | | | | | | | | |

**Cautions 1. Read the value written to CRCD register before writing to CRCIN register.**

**2. If conflict between writing and storing operation result to CRCD register occurs, the writing is ignored.**

**<Operation flow>**

**Figure 30-6. CRC Operation Function (General-purpose CRC)**



### 30.3.3 RAM parity error detection function

The IEC60730 standard mandates the checking of RAM data. A single-bit parity bit is therefore added to all 8-bit data in the RL78/I1B's RAM. By using this RAM parity error detection function, 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.

#### 30.3.3.1 RAM parity error control register (RPECTL)

This register is used to control parity error generation check bit and reset generation due to parity errors.

The RPECTL register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 30-7. Format of RAM Parity Error Control Register (RPECTL)**

Address: F00F5H After reset: 00H R/W

| Symbol | <7> | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| RPECTL | RPERDIS | 0 | 0 | 0 | 0 | 0 | 0 | RPEF |

| RPERDIS | Parity error reset mask flag |
|---|---|
| 0 | Enable parity error resets. |
| 1 | Disable 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.**

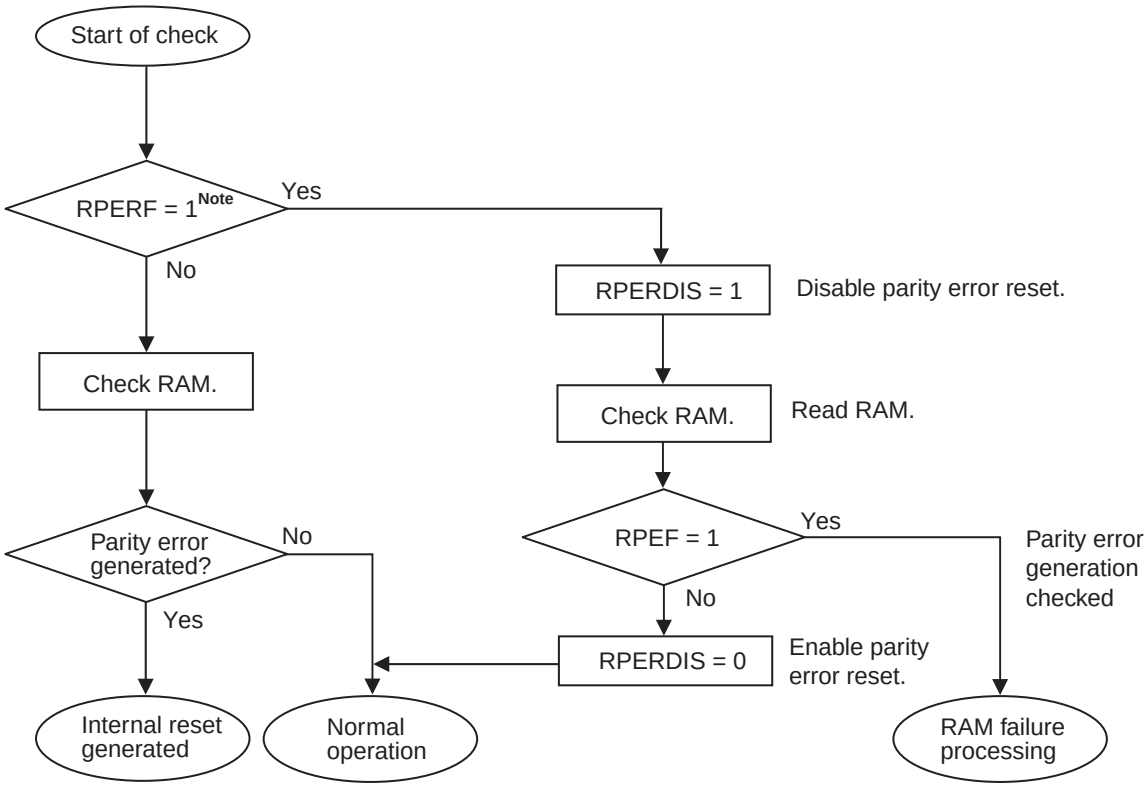
**Remarks 1.** The parity error reset is enabled by default (RPERDIS = 0).

**2.** Even if the parity error reset is disabled (RPERDIS = 1), the RPEF flag will be set (1) if a parity error occurs. If parity error resets are enabled (RPERDIS = 0) with RPEF set to 1, a parity error reset is generated when the RPERDIS bit is cleared to 0.

**3.** The RPEF flag in the RPECTL register is set (1) when the RAM parity error occurs and cleared (0) by writing 0 to it or by any reset source. When RPEF = 1, the value is retained even if RAM for which no parity error has occurred is read.

**4.** The general registers are not included for RAM parity error detection.

**Figure 30-8. Flowchart of RAM Parity Check**




**Note** To check internal reset status using a RAM parity error, see **CHAPTER 25 RESET FUNCTION**.

### 30.3.4 RAM guard function

In order to guarantee safety during operation, the IEC61508 standard mandates that important data stored in the RAM be protected, even if the CPU freezes.

This RAM guard function is used to protect data in the specified memory space.

If the RAM guard function is specified, writing to the specified RAM space is disabled, but reading from the space can be carried out as usual.

#### 30.3.4.1 Invalid memory access detection control register (IAWCTL)

This register is used to control the detection of invalid memory access and RAM/SFR guard function.

GRAM1 and GRAM0 bits are used in RAM guard function.

The IAWCTL register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 30-9. Format of Invalid Memory Access Detection Control Register (IAWCTL)**

Address: F0078H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| IAWCTL | IAWEN | 0 | GRAM1 | GRAM0 | 0 | GPORT | GINT | GCSC |

| GRAM1 | GRAM0 | RAM guard space[Note] |
|---|---|---|
| 0 | 0 | Disabled. RAM can be written to. |
| 0 | 1 | The 128 bytes of space starting at the start address in the RAM |
| 1 | 0 | The 256 bytes of space starting at the start address in the RAM |
| 1 | 1 | The 512 bytes of space starting at the start address in the RAM |

**Note** The RAM start address differs depending on the size of the RAM provided with the product.

### 30.3.5 SFR guard function

In order to guarantee safety during operation, the IEC61508 standard mandates that important SFRs be protected from being overwritten, even if the CPU freezes.

This SFR guard function is used to protect data in the control registers used by the port function, interrupt function, clock control function, voltage detection function, and RAM parity error detection function.

If the SFR guard function is specified, writing to the specified SFRs is disabled, but reading from the SFRs can be carried out as usual.

#### 30.3.5.1 Invalid memory access detection control register (IAWCTL)

This register is used to control the detection of invalid memory access and RAM/SFR guard function.

GPORT, GINT and GCSC bits are used in SFR guard function.

The IAWCTL register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 30-10. Format of Invalid Memory Access Detection Control Register (IAWCTL)**

Address: F0078H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| IAWCTL | IAWEN | 0 | GRAM1 | GRAM0 | 0 | GPORT | GINT | GCSC |

| GPORT | Control registers of port function guard |
|---|---|
| 0 | Disabled. Control registers of port function can be read or written to. |
| 1 | Enabled. Writing to control registers of port function is disabled. Reading is enabled.<br>[Guarded SFR] PMxx, PUxx, PIMxx, POMxx, ADPC, PIOR, PFSEGxx, ISCLCD[Note] |

| GINT | Registers of interrupt function guard |
|---|---|
| 0 | Disabled. Registers of interrupt function can be read or written to. |
| 1 | Enabled. Writing to registers of interrupt function is disabled. Reading is enabled.<br>[Guarded SFR] IFxx, MKxx, PRxx, EGPx, EGNx |

| GCSC | Control registers of clock control function, voltage detector and RAM parity error detection function guard |
|---|---|
| 0 | Disabled. Control registers of clock control function, voltage detector and RAM parity error detection function can be read or written to. |
| 1 | Enabled. Writing to control registers of clock control function, voltage detector and RAM parity error detection function is disabled. Reading is enabled.<br>[Guarded SFR] CMC, CSC, OSTS, CKC, PERx, OSMC, LVIM, LVIS, RPECTL |

**Note** Pxx (Port register) is not guarded.

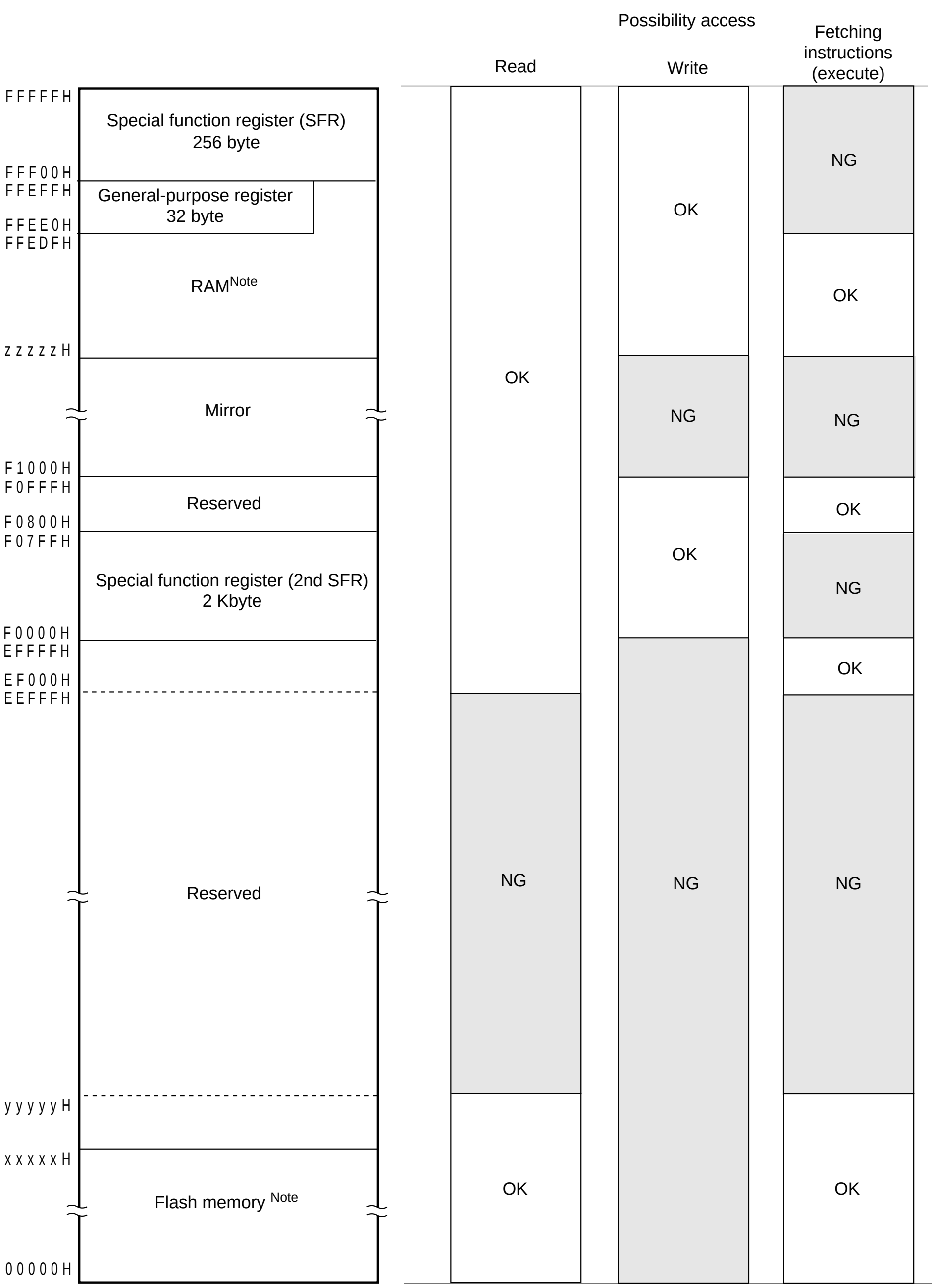
### 30.3.6 Invalid memory access detection function

The IEC60730 standard mandates checking that the CPU and interrupts are operating correctly.

The illegal memory access detection function triggers a reset if a memory space specified as access-prohibited is accessed.

The illegal memory access detection function applies to the areas indicated by NG in Figure 30-11.

**Figure 30-11. Invalid access detection area**




**Note** Code flash memory and RAM address of each product are as follows.

<R>

| Products | Code flash memory (00000H to xxxxxH) | RAM (zzzzzH to FFEFFH) | Detected lowest address for read/instruction fetch (execution) (yyyyyH) |
|---|---|---|---|
| R5F10MME, R5F10MPE | 65536 × 8 bits (00000H to 0FFFFH) | 6144 × 8 bits (FE700H to FFEFFH) | 10000H |
| R5F10MMG, R5F10MPG | 131072 × 8 bits (00000H to 1FFFFH) | 8192 × 8 bits (FDF00H to FFEFFH) | 20000H |

#### 30.3.6.1 Invalid memory access detection control register (IAWCTL)

This register is used to control the detection of invalid memory access and RAM/SFR guard function.

IAWEN bit is used in invalid memory access detection function.

The IAWCTL register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 30-12. Format of Invalid Memory Access Detection Control Register (IAWCTL)**

Address: F0078H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| IAWCTL | IAWEN | 0 | GRAM1 | GRAM0 | 0 | GPORT | GINT | GCSC |

| IAWEN[Note] | Control of invalid memory access detection |
|---|---|
| 0 | Disable the detection of invalid memory access. |
| 1 | Enable the detection of invalid memory access. |

**Note** Only writing 1 to the IAWEN bit is enabled, not writing 0 to it after setting it to 1.

**Remark** By specifying WDTON = 1 (watchdog timer operation enable) for the option byte (000C0H), the invalid memory access function is enabled even IAWEN = 0.

### 30.3.7 Frequency detection function

The IEC60730 standard mandates checking that the oscillation frequency is correct.

By using the CPU/peripheral hardware clock frequency ($f_{CLK}$) and measuring 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 stopped, the proportional relationship between the clocks cannot be determined.

**<Clocks to be compared>**

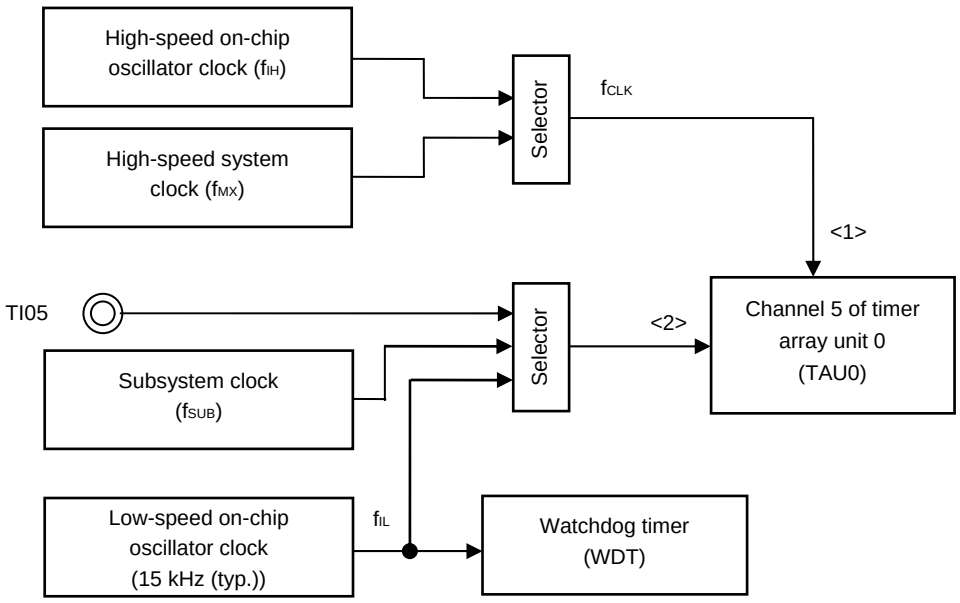
<1> CPU/peripheral hardware clock frequency ($f_{CLK}$):

- High-speed on-chip oscillator clock ($f_{IH}$)
- High-speed system clock ($f_{MX}$)

<2> Input to channel 5 of the timer array unit

- Timer input to channel 5 (TI05)
- Low-speed on-chip oscillator clock ($f_{IL}$: 15 kHz (typ.))
- Subsystem clock ($f_{SUB}$)

**Figure 30-13. Configuration of Frequency Detection Function**




If input pulse interval measurement results in an abnormal value, it can be concluded that the clock frequency is abnormal.

For how to execute input pulse interval measurement, see **7.8.3 Operation as input pulse interval measurement**.

#### 30.3.7.1 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.

Reset signal generation clears this register to 00H.

**Figure 30-14. Format of Timer Input Select Register 0 (TIS0)**

Address: F0074H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TIS0 | 0 | 0 | 0 | 0 | 0 | TIS02 | TIS01 | TIS00 |

| TIS02 | TIS01 | TIS00 | Selection of timer input used with channel 1 |
|---|---|---|---|
| 0 | 0 | 0 | Input signal of timer input pin (TI05) |
| 0 | 0 | 1 | |
| 0 | 1 | 0 | |
| 0 | 1 | 1 | |
| 1 | 0 | 0 | Low-speed on-chip oscillator clock ($f_{IL}$) |
| 1 | 0 | 1 | Subsystem clock ($f_{SUB}$) |
| Other than above | | | Setting prohibited |

### 30.3.8 A/D test function

The IEC60730 standard mandates testing the A/D converter. The A/D test function checks whether or not the A/D converter is operating normally by executing A/D conversions of the A/D converter's positive and negative reference voltages, analog input channel (ANI), temperature sensor output voltage, and the internal reference voltage. For details of the check method, see the safety function (A/D test) application note (R01AN0955).

The analog multiplexer can be checked using the following procedure.

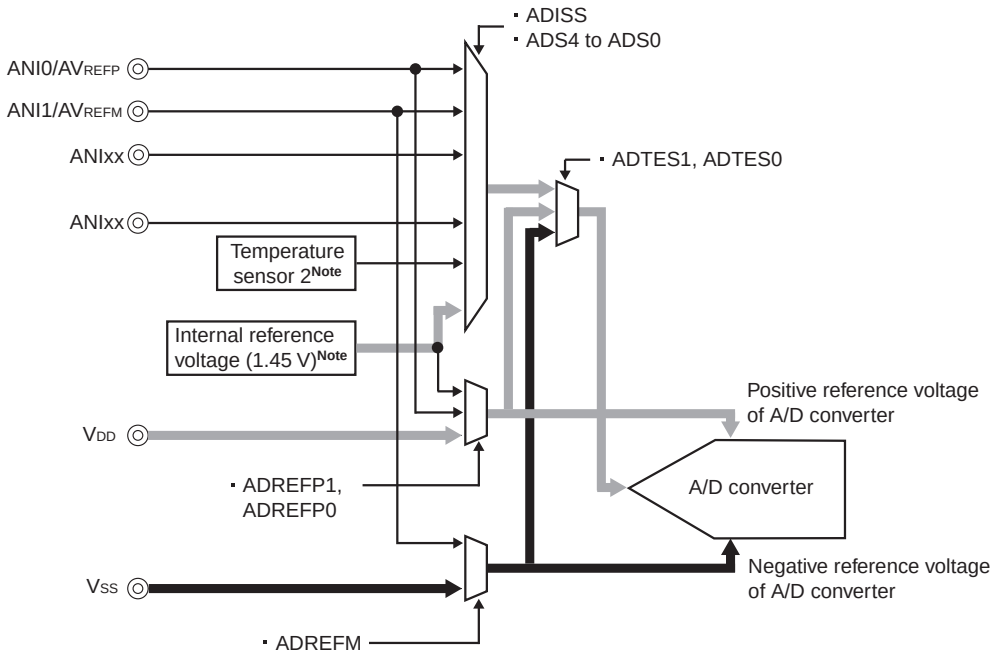
<1> Select the ANIx pin for A/D conversion using the ADTES register (ADTES1 = 0, ADTES0 = 0).
<2> Perform A/D conversion for the ANIx pin (conversion result 1-1).
<3> Select the A/D converter's negative reference voltage for A/D conversion using the ADTES register (ADTES1 = 1, ADTES0 = 0)
<4> Perform A/D conversion of the negative reference voltage of the A/D converter (conversion result 2-1).
<5> Select the ANIx pin for A/D conversion using the ADTES register (ADTES1 = 0, ADTES0 = 0).
<6> Perform A/D conversion for the ANIx pin (conversion result 1-2).
<7> Select the A/D converter's positive reference voltage for A/D conversion using the ADTES register (ADTES1 = 1, ADTES0 = 1)
<8> Perform A/D conversion of the positive reference voltage of the A/D converter (conversion result 2-2).
<9> Select the ANIx pin for A/D conversion using the ADTES register (ADTES1 = 0, ADTES0 = 0).
<10> Perform A/D conversion for the ANIx 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 conversion result 2-2 is all one.

Using the procedure above can confirm that the analog multiplexer is selected and all wiring is connected.

**Remarks 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.

**2.** The conversion results might contain an error. Consider an appropriate level of error when comparing the conversion results.

**Figure 30-15. Configuration of A/D Test Function**




**Note** This setting can be used only in HS (high-speed main) mode.

#### 30.3.8.1 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 channel (ANIxx), temperature sensor output voltage, or internal reference voltage (1.45 V) as the target of A/D conversion.

When using the A/D test function, specify the following settings:

- Select negative reference voltage as the target of A/D conversion for zero-scale measurement.
- Select 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.

Reset signal generation clears this register to 00H.

**Figure 30-16. Format of A/D Test Register (ADTES)**

Address: F0013H After reset: 00H 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[Note]/internal reference voltage (1.45 V)[Note] (This is specified using the analog input channel specification register (ADS).) |
| 1 | 0 | Negative reference voltage (selected with the ADREFM bit in ADM2) |
| 1 | 1 | Positive reference voltage (selected with the ADREFP1 or ADREFP0 bit in ADM2) |
| Other than above | | Setting prohibited |

**Note** Temperature sensor output voltage/internal reference voltage (1.45 V) can be used only in HS (high-speed main) mode.

### 30.3.8.2 Analog input channel specification register (ADS)

This register specifies the input channel of the analog voltage to be A/D converted.

Set A/D test register (ADTES) to 00H when measuring the ANIxx/temperature sensor output /internal reference voltage (1.45 V).

The ADS register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 30-17. Format of Analog Input Channel Specification Register (ADS)**

Address: FFF31H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ADS | ADISS | 0 | 0 | ADS4 | ADS3 | ADS2 | ADS1 | ADS0 |

| ADISS | ADS4 | ADS3 | ADS2 | ADS1 | ADS0 | Analog input channel | Input source |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | ANI0 | P20/ANI0/$AV_{REFP}$ pin |
| 0 | 0 | 0 | 0 | 0 | 1 | ANI1 | P21/ANI1/$AV_{REFM}$ 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 | 1 | 1 | 1 | 0 | 1 | – | Temperature sensor 2 output voltage[Note] |
| 1 | 0 | 0 | 0 | 0 | 1 | – | Internal reference voltage (1.45 V)[Note] |
| Other than above | | | | | | Setting prohibited | |

**Note** This setting can be used only in HS (high-speed main) mode.

**Cautions**

1. **Be sure to clear bits 5 and 6 to 0.**
2. **Select input mode for the ports which are set to analog input with the ADPC register, using the port mode register 2 (PM2).**
3. **Do not use the ADS register to set the pins which should be set as digital I/O with the A/D port configuration register (ADPC).**
4. **Only rewrite the value of the ADISS bit while conversion operation is stopped (ADCE = 0, ADCS = 0).**
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.**
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.**
7. **If ADISS is set to 1, the internal reference voltage (1.45 V) cannot be used for the positive reference voltage. In addition, the first conversion result obtained after setting ADISS to 1 is not available.**
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 ADISS to 1. When ADISS is 1, the A/D converter reference voltage current ($I_{ADREF}$) shown in 37.3.2 Supply current characteristics is added.**

### 30.3.9 Digital output signal level detection function for I/O ports

In the IEC60730, it is required to check that the I/O function correctly operates.

By using the digital output signal level detection function for I/O pins, the digital output level of the pin can be read when the pin is set to output mode.

#### 30.3.9.1 Port mode select register (PMS)

This register is used to select the output level from output latch level or pin output level when the port is output mode in which PMm bit of port mode register (PMm) is 0.

This register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H.

**Figure 30-18. Format of Port Mode Select Register (PMS)**

Address: F007BH After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMS | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PMS0 |

| PMS0 | Method for selecting output level to be read when port is output mode (PMmn = 0) |
|---|---|
| 0 | Pmn register value is read. |
| 1 | Digital output level of the pin is read. |

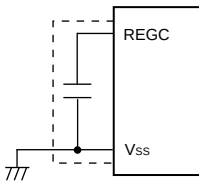
**Remark** m = 0 to 8, 12
n = 0 to 7

**Cautions**
1. **While the PMS0 bit of the PMS register is “1”, do not change the value of the Px register by using a read-modify instruction. To change the value of the Px register, use an 8-bit manipulation instruction.**
2. **PMS control cannot be used for the dedicated LCD pins and the input-only pins (P121 to P124 and P137).**
3. **PMS control cannot be used for alternate-function pins being used as segment output pins. (“L” is always read when this register is read.)**
4. **PMS control cannot be used for P61 and P60 when IICA0EN (bit 4 of the PER0 register) is 0.**

# CHAPTER 31 REGULATOR

## 31.1 Regulator Overview

The RL78/I1B contains a circuit for operating the device with a constant voltage. At this time, in order to stabilize the regulator output voltage, connect the REGC pin to $V_{SS}$ via a capacitor (0.47 to 1 μF). Also, use a capacitor with good characteristics, since it is used to stabilize internal voltage.




**Caution Keep the wiring length as short as possible for the broken-line part in the above figure.**

The regulator output voltage, see **Table 31-1**.

**Table 31-1. Regulator Output Voltage Conditions**

| Mode | Output Voltage | Condition |
|---|---|---|
| LS (low-speed main) mode | 1.8 V | |
| HS (high-speed main) mode | 1.8 V | In STOP mode |
| | | When both the high-speed system clock ($f_{MX}$) and the high-speed on-chip oscillator clock ($f_{IH}$) are stopped during CPU operation with the subsystem clock ($f_{SUB}$) |
| | | When both the high-speed system clock ($f_{MX}$) and the high-speed on-chip oscillator clock ($f_{IH}$) are stopped during the HALT mode when the CPU operation with the subsystem clock ($f_{SUB}$) has been set |
| | 2.1 V | Other than above (include during OCD mode)**Note** |

**Note** When it shifts to the subsystem clock operation or STOP mode during the on-chip debugging, the regulator output voltage is kept at 2.1 V (not decline to 1.8 V).

# CHAPTER 32 OPTION BYTE

## 32.1 Functions of Option Bytes

Addresses 000C0H to 000C3H of the flash memory of the RL78/I1B form an option byte area.

Option bytes consist of user option byte (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 bits for which no function is assigned, do not change their initial values.

To use the boot swap operation during self programming, 000C0H to 000C3H are replaced by 010C0H to 010C3H. Therefore, set the same values as 000C0H to 000C3H to 010C0H to 010C3H.

**Caution The option bytes should always be set regardless of whether each function is used.**

### 32.1.1 User option byte (000C0H to 000C2H/010C0H to 010C2H)

**(1) 000C0H/010C0H**

O Operation of watchdog timer
- Enabling or disabling of counter operation
- Enabling or disabling of counter operation in the HALT or STOP mode

O Setting of overflow time of watchdog timer

O Setting of window open period of watchdog timer

O Setting of interval interrupt of watchdog timer
- Whether or not to use the interval interrupt is selectable.

**Caution Set the same value as 000C0H to 010C0H when the boot swap operation is used because 000C0H is replaced by 010C0H.**

**(2) 000C1H/010C1H**

O Setting of LVD operation mode
- Interrupt & reset mode.
- Reset mode.
- Interrupt mode.
- LVD off (by controlling the externally input reset signal on the $\overline{\text{RESET}}$ pin)

O Setting of LVD detection level ($V_{LVDH}$, $V_{LVDL}$, $V_{LVD}$)

**Cautions 1. After power is supplied, the reset state must be retained until the operating voltage becomes in the range defined in 37.4 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 010C2H).**

**2. Set the same value as 000C1H to 010C1H when the boot swap operation is used because 000C1H is replaced by 010C1H.**

**(3) 000C2H/010C2H**

O Setting of flash operation mode

<R> 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

O Setting of the frequency of the high-speed on-chip oscillator

- Select from 3 MHz, 6 MHz, 12 MHz, and 24 MHz.

**Caution Set the same value as 000C2H to 010C2H when the boot swap operation is used because 000C2H is replaced by 010C2H.**

### 32.1.2 On-chip debug option byte (000C3H/ 010C3H)

O Control of on-chip debug operation

- On-chip debug operation is disabled or enabled.

O 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 Set the same value as 000C3H to 010C3H when the boot swap operation is used because 000C3H is replaced by 010C3H.**

## 32.2 Format of User Option Byte

The format of user option byte is shown below.

**Figure 32-1. Format of User Option Byte (000C0H/010C0H)**

Address: 000C0H/010C0H[Note 1]

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| WDTINIT | WINDOW1 | WINDOW0 | WDTON | WDCS2 | WDCS1 | WDCS0 | WDSTBYON |

| WDTINIT | Use of interval interrupt of watchdog timer |
|---|---|
| 0 | Interval interrupt is not used. |
| 1 | Interval interrupt is generated when 75% + 1/2$f_{IL}$ of the overflow time is reached. |

| WINDOW1 | WINDOW0 | Watchdog timer window open period[Note 2] |
|---|---|---|
| 0 | 0 | Setting prohibited |
| 0 | 1 | 50% |
| 1 | 0 | 75% |
| 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 ($f_{IL}$ = 17.25 kHz (MAX.)) |
|---|---|---|---|
| 0 | 0 | 0 | $2^6/f_{IL}$ (3.71 ms) |
| 0 | 0 | 1 | $2^7/f_{IL}$ (7.42 ms) |
| 0 | 1 | 0 | $2^8/f_{IL}$ (14.84 ms) |
| 0 | 1 | 1 | $2^9/f_{IL}$ (29.68 ms) |
| 1 | 0 | 0 | $2^{11}/f_{IL}$ (118.72 ms) |
| 1 | 0 | 1 | $2^{13}/f_{IL}$ (474.89 ms) |
| 1 | 1 | 0 | $2^{14}/f_{IL}$ (949.79 ms) |
| 1 | 1 | 1 | $2^{16}/f_{IL}$ (3799.18 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 |

**Notes 1.** Set the same value as 000C0H to 010C0H when the boot swap operation is used because 000C0H is replaced by 010C0H.

**2.** The window open period is 100% when WDSTBYON = 0, regardless the value of the WINDOW1 and WINDOW0 bits.

**Remark** $f_{IL}$: Low-speed on-chip oscillator clock frequency

**Figure 32-2. Format of User Option Byte (000C1H/010C1H) (1/2)**

Address: 000C1H/010C1H[Note]

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| VPOC2 | VPOC1 | VPOC0 | 1 | LVIS1 | LVIS0 | LVIMDS1 | LVIMDS0 |

- LVD setting (interrupt & reset mode)

| Detection voltage | | | Option byte setting value | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $V_{LVDH}$ | | $V_{LVDL}$ | VPOC2 | VPOC1 | VPOC0 | LVIS1 | LVIS0 | Mode setting | |
| Rising edge | Falling edge | Falling edge | | | | | | LVIMDS1 | LVIMDS0 |
| 2.61 V | 2.55 V | 2.45 V | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| 2.71 V | 2.65 V | | | | | 0 | 1 | | |
| 3.75 V | 3.67 V | | | | | 0 | 0 | | |
| 2.92 V | 2.86 V | 2.75 V | | 1 | 1 | 1 | 0 | | |
| 3.02 V | 2.96 V | | | | | 0 | 1 | | |
| 4.06 V | 3.98 V | | | | | 0 | 0 | | |
| – | | | Setting of values other than above is prohibited. | | | | | | |

- LVD setting (reset mode)

| Detection voltage | | Option byte setting value | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $V_{LVD}$ | | VPOC2 | VPOC1 | VPOC0 | LVIS1 | LVIS0 | Mode setting | |
| Rising edge | Falling edge | | | | | | LVIMDS1 | LVIMDS0 |
| 1.98 V | 1.94 V | 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| 2.09 V | 2.04 V | | 0 | 1 | 0 | 1 | | |
| 2.50 V | 2.45 V | | 1 | 0 | 1 | 1 | | |
| 2.61 V | 2.55 V | | 1 | 0 | 1 | 0 | | |
| 2.71 V | 2.65 V | | 1 | 0 | 0 | 1 | | |
| 2.81 V | 2.75 V | | 1 | 1 | 1 | 1 | | |
| 2.92 V | 2.86 V | | 1 | 1 | 1 | 0 | | |
| 3.02 V | 2.96 V | | 1 | 1 | 0 | 1 | | |
| 3.13 V | 3.06 V | | 0 | 1 | 0 | 0 | | |
| 3.75 V | 3.67 V | | 1 | 0 | 0 | 0 | | |
| 4.06 V | 3.98 V | | 1 | 1 | 0 | 0 | | |
| – | | Setting of values other than above is prohibited. | | | | | | |

**Note** Set the same value as 000C1H to 010C1H when the boot swap operation is used because 000C1H is replaced by 010C1H.

**Caution Be sure to set bit 4 to “1”.**

**Remarks 1.** Refer to LVD setting, see **27.1 Functions of Voltage Detector**.

**2.** The detection voltage is a typical value. For details, see **37.6.6 LVD circuit characteristics.**

(**Cautions** are listed on the next page.)

**Figure 32-2. Format of User Option Byte (000C1H/010C1H) (2/2)**

Address: 000C1H/010C1H[Note]

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| VPOC2 | VPOC1 | VPOC0 | 1 | LVIS1 | LVIS0 | LVIMDS1 | LVIMDS0 |

- LVD setting (interrupt mode)

| Detection voltage | | Option byte setting value | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $V_{LVD}$ | | VPOC2 | VPOC1 | VPOC0 | LVIS1 | LVIS0 | Mode setting | |
| Rising edge | Falling edge | | | | | | LVIMDS1 | LVIMDS0 |
| 1.98 V | 1.94 V | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 2.09 V | 2.04 V | | 0 | 1 | 0 | 1 | | |
| 2.50 V | 2.45 V | | 1 | 0 | 1 | 1 | | |
| 2.61 V | 2.55 V | | 1 | 0 | 1 | 0 | | |
| 2.71 V | 2.65 V | | 1 | 0 | 0 | 1 | | |
| 2.81 V | 2.75 V | | 1 | 1 | 1 | 1 | | |
| 2.92 V | 2.86 V | | 1 | 1 | 1 | 0 | | |
| 3.02 V | 2.96 V | | 1 | 1 | 0 | 1 | | |
| 3.13 V | 3.06 V | | 0 | 1 | 0 | 0 | | |
| 3.75 V | 3.67 V | | 1 | 0 | 0 | 0 | | |
| 4.06 V | 3.98 V | | 1 | 1 | 0 | 0 | | |
| – | | Setting of values other than above is prohibited. | | | | | | |

- LVD off setting (use of external reset input via $\overline{\text{RESET}}$ pin)

| Detection voltage | | Option byte setting value | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $V_{LVD}$ | | VPOC2 | VPOC1 | VPOC0 | LVIS1 | LVIS0 | Mode setting | |
| Rising edge | Falling edge | | | | | | LVIMDS1 | LVIMDS0 |
| – | – | 1 | × | × | × | × | × | 1 |
| – | | Setting of values other than above is prohibited. | | | | | | |

**Note** Set the same value as 000C1H to 010C1H when the boot swap operation is used because 000C1H is replaced by 010C1H.

**Cautions 1. Be sure to set bit 4 to "1".**

**2. After power is supplied, the reset state must be retained until the operating voltage becomes in the range defined in 37.4 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 010C2H).**

**Remarks 1.** ×: don't care

**2.** Refer to LVD setting, see **27.1 Functions of Voltage Detector**.

**3.** The detection voltage is a typical value. For details, see **37.6.6 LVD circuit characteristics.**

**Figure 32-3. Format of Option Byte (000C2H/010C2H)**

Address: 000C2H/010C2H[Note 1]

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CMODE1 | CMODE0 | 1 | 0 | 0 | FRQSEL2 | FRQSEL1 | FRQSEL0 |

<R>

| CMODE1 | CMODE0 | Setting of flash operation mode | | |
|---|---|---|---|---|
| | | | Operating frequency range ($f_{MAIN}$) | Operating voltage Range ($V_{DD}$) |
| 1 | 0 | LS (low speed main) mode | 6/3 MHz | 1.9 to 5.5 V |
| 1 | 1 | HS (high speed main) mode | 6/3 MHz | 2.1 to 5.5 V[Note 2] |
| | | | 12/6/3 MHz | 2.4 to 5.5 V |
| | | | 24/12/6/3 MHz | 2.7 to 5.5 V |
| Other than above | | Setting prohibited | | |

| FRQSEL2 | FRQSEL1 | FRQSEL0 | Frequency of the high-speed on-chip oscillator clock |
|---|---|---|---|
| | | | $f_{IH}$ |
| 0 | 0 | 0 | 24 MHz |
| 0 | 0 | 1 | 12 MHz |
| 0 | 1 | 0 | 6 MHz |
| 0 | 1 | 1 | 3 MHz |
| Other than above | | | Setting prohibited |

**Notes 1.** Set the same value as 000C2H to 010C2H when the boot swap operation is used because 000C2H is replaced by 010C2H.

**2.** Use at $-20°C \leq T_A \leq +85°C$.

**Cautions 1. Be sure to set bits 5 and 4 to 10B.**

**2. The ranges of operation frequency and operation voltage vary depending on the flash operation mode. For details, see 37.4 AC Characteristics.**

## 32.3 Format of On-chip Debug Option Byte

The format of on-chip debug option byte is shown below.

**Figure 32-4. Format of On-chip Debug Option Byte (000C3H/010C3H)**

Address: 000C3H/010C3H[Note]

| 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 debug operation. |
| 0 | 1 | Setting prohibited |
| 1 | 0 | Enables on-chip debugging.<br>Erases data of flash memory in case of failures in authenticating on-chip debug security ID. |
| 1 | 1 | Enables on-chip debugging.<br>Does not erases data of flash memory in case of failures in authenticating on-chip debug security ID. |

**Note** Set the same value as 000C3H to 010C3H when the boot swap operation is used because 000C3H is replaced by 010C3H.

**Caution Bits 7 and 0 (OCDENSET and OCDERSD) can only be specified a value.**
**Be sure to set bits 6 to 1 to 000010B.**

**Remark** The value on bits 3 to 1 will be written over when the on-chip debug function is in use and thus it will become unstable after the setting.
However, be sure to set the default values (0, 1, and 0) to bits 3 to 1 at setting.

## 32.4 Setting of Option Byte

The user option byte and on-chip debug option byte can be set using the link option, in addition to describing to the source. When doing so, the contents set by using the link option take precedence, even if descriptions exist in the source, as mentioned below.

A software description example of the option byte setting is shown below.

```
OPT   CSEG        OPT_BYTE
      DB          36H     ; Does not use interval interrupt of watchdog timer,
                          ; Enables watchdog timer operation,
                          ; Window open period of watchdog timer is 50%,
                          ; Overflow time of watchdog timer is 2^9/fIL,
                          ; Stops watchdog timer operation during HALT/STOP mode
      DB          5AH     ; Select 2.45 V for VLVDL
                          ; Select rising edge 2.61 V, falling edge 2.55 V for VLVDH
                          ; Select the interrupt & reset mode as the LVD operation mode
      DB          A3H     ; Select the LS (low speed main) mode as the flash operation mode
                            and 3 MHz as the frequency of the high-speed on-chip oscillator
      DB          85H     ; Enables on-chip debug operation, does not erase flash memory
                            data when security ID authorization fails
```

When the boot swap function is used during self programming, 000C0H to 000C3H is switched to 010C0H to 010C3H. Describe to 010C0H to 010C3H, therefore, the same values as 000C0H to 000C3H as follows.

```
OPT2  CSEG   AT   010C0H
      DB          36H     ; Does not use interval interrupt of watchdog timer,
                          ; Enables watchdog timer operation,
                          ; Window open period of watchdog timer is 50%,
                          ; Overflow time of watchdog timer is 2^10/fIL,
                          ; Stops watchdog timer operation during HALT/STOP mode
      DB          5AH     ; Select 2.45 V for VLVDL
                          ; Select rising edge 2.61 V, falling edge 2.55 V for VLVDH
                          ; Select the interrupt & reset mode as the LVD operation mode
      DB          A3H     ; Select the LS (low speed main) mode as the flash operation mode
                            and 3 MHz as the frequency of the high-speed on-chip oscillator
      DB          85H     ; Enables on-chip debug operation, does not erase flash memory
                            data when security ID authorization 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 the option byte to 010C0H to 010C3H in order to use the boot swap function, use the relocation attribute AT to specify an absolute address.**

# CHAPTER 33 FLASH MEMORY

The RL78 microcontroller incorporates the flash memory to which a program can be written, erased, and overwritten while mounted on the board. The flash memory includes the “code flash memory”, in which programs can be executed.



The following three methods for programming the flash memory are available:

The code flash memory can be rewritten to through serial programming using a flash memory programmer or an external device (UART communication), or through self-programming.

- Serial programming using flash memory programmer (see **33.4**)
  Data can be written to the flash memory on-board or off-board by using a dedicated flash memory programmer.
- Serial programming using external device (UART communication) (see **33.2**)
  Data can be written to the flash memory on-board through UART communication with an external device (microcontroller or ASIC).
- Self-programming (see **33.5**)
  The user application can execute self-programming of the code flash memory by using the flash self-programming library.

## 33.1 Serial Programming Using Flash Memory Programmer

The following dedicated flash memory programmer can be used to write data to the internal flash memory of the RL78 microcontroller.

- PG-FP5, FL-PR5
- E1 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 the RL78 microcontroller has been mounted on the target system. The connectors that connect the dedicated flash memory programmer must be mounted on the target system.

**(2) Off-board programming**

Data can be written to the flash memory with a dedicated program adapter (FA series) before the RL78 microcontroller is mounted on the target system.

**Remark** FL-PR5 and FA series are products of Naito Densei Machida Mfg. Co., Ltd.

**Table 33-1. Wiring Between RL78 microcontroller and Dedicated Flash Memory Programmer**

| Pin Configuration of Dedicated Flash Memory Programmer | | | | Pin Name | Pin No. | |
|---|---|---|---|---|---|---|
| | | | | | 80-pin | 100-pin |
| Signal Name | | I/O | Pin Function | | LFQFP (12×12) | LFQFP (14×14) |
| PG-FP5, FL-PR5 | E1 on-chip debugging emulator | | | | | |
| – | TOOL0 | I/O | Transmit/receive signal | TOOL0/P40 | 8 | 14 |
| SI/RxD | – | I/O | Transmit/receive signal | | | |
| – | $\overline{\text{RESET}}$ | Output | Reset signal | $\overline{\text{RESET}}$ | 9 | 15 |
| /RESET | – | Output | | | | |
| $V_{DD}$ | | I/O | $V_{DD}$ voltage generation/ power monitoring | $V_{DD}$ | 17 | 23 |
| GND | | – | Ground | $V_{SS}$ | 16 | 22 |
| | | | | $EV_{SS1}$ | – | 54 |
| | | | | REGC[Note] | 15 | 21 |
| FLMD1 | $EMV_{DD}$ | – | Driving power for TOOL pin | $V_{DD}$ | 17 | 23 |
| | | | | $EV_{DD1}$ | – | 63 |

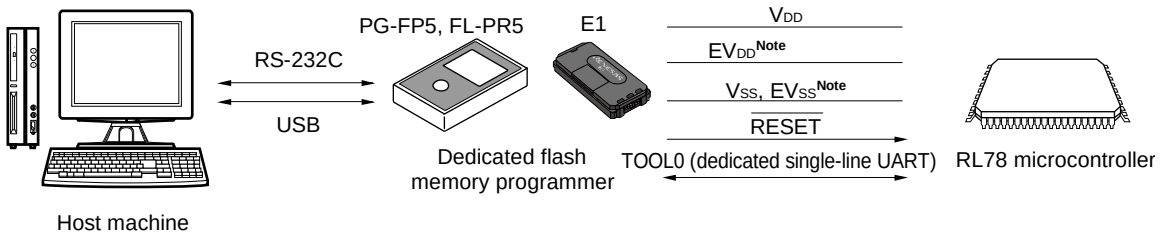
**Note** Connect REGC pin to ground via a capacitor (0.47 to 1 μF).

**Remark** Pins that are not indicated in the above table can be left open when using the flash memory programmer for flash programming.

### 33.1.1 Programming environment

The environment required for writing a program to the flash memory of the RL78 microcontroller is illustrated below.

<R> **Figure 33-1. Environment for Writing Program to Flash Memory**




**Note** 100-pin products only.

A host machine that controls the dedicated flash memory programmer is necessary.

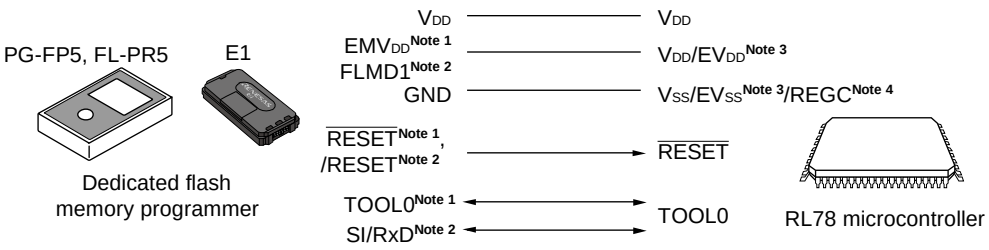
To interface between the dedicated flash memory programmer and the RL78 microcontroller, the TOOL0 pin is used for manipulation such as writing and erasing via a dedicated single-line UART.

### 33.1.2 Communication mode

Communication between the dedicated flash memory programmer and the RL78 microcontroller is established by serial communication using the TOOL0 pin via a dedicated single-line UART of the RL78 microcontroller.

Transfer rate: 1 M, 500 k, 250 k, 115.2 kbps

<R> **Figure 33-2. Communication with Dedicated Flash Memory Programmer**




**Notes 1.** When using E1 on-chip debugging emulator.

**2.** When using PG-FP5 or FL-PR5.

**3.** 100-pin products only.

**4.** Connect REGC pin to ground via a capacitor (0.47 to 1 μF).

The dedicated flash memory programmer generates the following signals for the RL78 microcontroller. See the manual of PG-FP5, FL-PR5, or E1 on-chip debugging emulator for details.

<R> **Table 33-2. Pin Connection**

| Dedicated Flash Memory Programmer | | | | RL78 Microcontroller |
|---|---|---|---|---|
| Signal Name | | I/O | Pin Function | Pin Name Note2 |
| PG-FP5, FL-PR5 | E1 on-chip debugging emulator | | | |
| $V_{DD}$ | | I/O | $V_{DD}$ voltage generation/power monitoring | $V_{DD}$ |
| GND | | – | Ground | $V_{SS}$, $EV_{SS1}$, REGC Note1 |
| FLMD1 | $EMV_{DD}$ | – | Driving power for TOOL pin | $V_{DD}$, $EV_{DD1}$ |
| /RESET | – | Output | Reset signal | $\overline{RESET}$ |
| – | $\overline{RESET}$ | Output | | |
| – | TOOL0 | I/O | Transmit/receive signal | TOOL0 |
| SI/RxD | – | I/O | Transmit/receive signal | |

**Notes1.** Connect REGC pin to ground via a capacitor (0.47 to 1 μF).

<R> **2.** Pins to be connected differ with the product. For details, see **Table 33 - 1**.

## 33.2 Serial Programming Using External Device (That Incorporates UART)

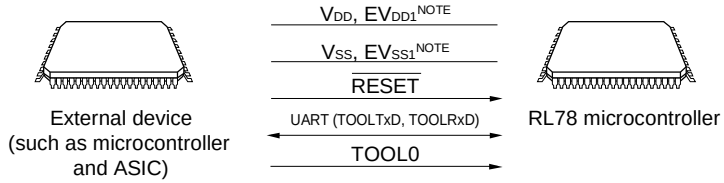
On-board data writing to the internal flash memory is possible by using the RL78 microcontroller and an external device (a microcontroller or ASIC) connected to a UART.

On the development of flash memory programmer by user, refer to the **RL78 Microcontrollers (RL78 Protocol A) Programmer Edition Application Note (R01AN0815)**.

### 33.2.1 Programming environment

The environment required for writing a program to the flash memory of the RL78 microcontroller is illustrated below.

<R> **Figure 33-3. Environment for Writing Program to Flash Memory**




<R> **Note** 100-pin products only.

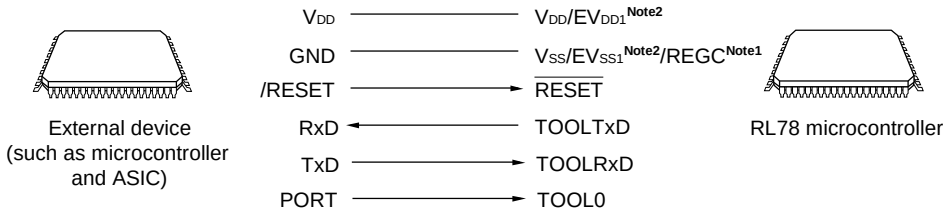
Processing to write data to or delete data from the RL78 microcontroller by using an external device is performed on-board. Off-board writing is not possible.

### 33.2.2 Communication mode

Communication between the external device and the RL78 microcontroller is established by serial communication using the TOOLTxD and TOOLRxD pins via the dedicated UART of the RL78 microcontroller.

Transfer rate: 1 M, 500 k, 250 k, 115.2kbps

<R> **Figure 33-4. Communication with External Device**




**Notes1.** Connect REGC pin to ground via a capacitor (0.47 to 1 μF).

<R> **2.** 100-pin products only.

**Caution Make $EV_{DD}$ the same potential as $V_{DD}$.**

The external device generates the following signals for the RL78 microcontroller.

<R> **Table 33-3. Pin Connection**

| External Device | | | RL78 Microcontroller |
|---|---|---|---|
| Signal Name | I/O | Pin Function | Pin Name |
| $V_{DD}$ | I/O | $V_{DD}$ voltage generation/power monitoring | $V_{DD}$, $EV_{DD1}$ Note2 |
| GND | – | Ground | $V_{SS}$, $EV_{SS1}$ Note2, REGC Note1 |
| RESETOUT | Output | Reset signal output | $\overline{RESET}$ |
| RxD | Input | Receive signal | TOOLTxD |
| TxD | Output | Transmit signal | TOOLRxD |
| PORT | Output | Mode signal | TOOL0 |

**Notes1.** Connect REGC pin to ground via a capacitor (0.47 to 1 μF).

<R> **2.** 100-pin products only.

**Caution Make $EV_{DD1}$ the same potential as $V_{DD}$.**

## 33.3 Connection of Pins on Board

To write the flash memory on-board by using the flash memory programmer, connectors that connect the dedicated flash memory programmer must be provided on the target system. First provide a function that selects the normal operation mode or flash memory programming mode on the board.

When the flash memory programming mode is set, all the pins not used for programming the flash memory are in the same status as immediately after reset. Therefore, if the external device does not recognize the state immediately after reset, the pins must be handled as described below.

**Remark** Refer to flash programming mode, see **33.4.2 Flash memory programming mode**.

### 33.3.1 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 low-level is prohibited for $t_{HD}$ period after pin reset release. However, when this pin is used via pull-down resistors, use the 500 kΩ or more resistors.

When used as an output pin: When this pin is used via pull-down resistors, use the 500 kΩ or more resistors.

**Remarks 1.** $t_{HD}$: How long to keep the TOOL0 pin at the low level from when the external and internal resets end for setting of the flash memory programming mode (see **37.12 Timing Specs for Switching Flash Memory Programming Modes**)

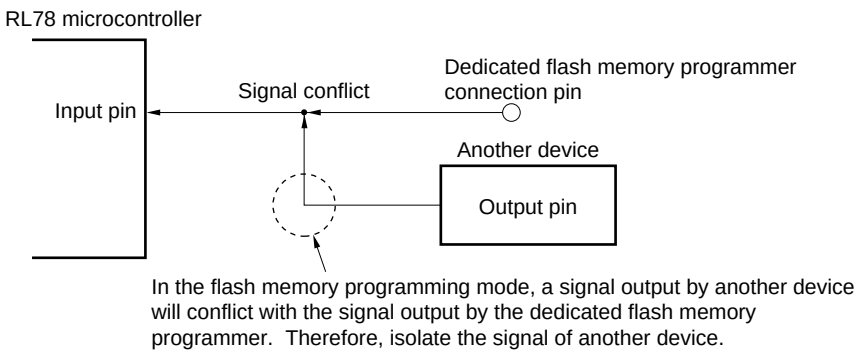
**2.** The SAU and IICA pins are not used for communication between the RL78 microcontroller and dedicated flash memory programmer, because single-line UART (TOOL0 pin) is used.

### 33.3.2 $\overline{\text{RESET}}$ pin

Signal conflict will occur if the reset signal of the dedicated flash memory programmer and external device are 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 33-5. Signal Conflict ($\overline{\text{RESET}}$ Pin)**



### 33.3.3 Port pins

When the flash memory programming mode is set, all the pins not used for flash memory programming enter the same status as that immediately after reset. If external devices connected to the ports do not recognize the port status immediately after reset, the port pin must be connected to either to $V_{DD}$ or $EV_{DD1}$, or $V_{SS}$ or $EV_{SS1}$, via a resistor.

### 33.3.4 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.

### 33.3.5 X1 and X2 pins

Connect X1 and X2 in the same status as in the normal operation mode.

**Remark** In the flash memory programming mode, the high-speed on-chip oscillator clock ($f_{IH}$) is used.

### 33.3.6 Power supply

To use the supply voltage output of the flash memory programmer, connect the $V_{DD}$ pin to $V_{DD}$ of the flash memory programmer, and the $V_{SS}$ pin to GND of the flash memory programmer.

To use the on-board supply voltage, connect in compliance with the normal operation mode.

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 $V_{DD}$ and $V_{SS}$ pins to $V_{DD}$ and GND of the flash memory programmer to use the power monitor function with the flash memory programmer.

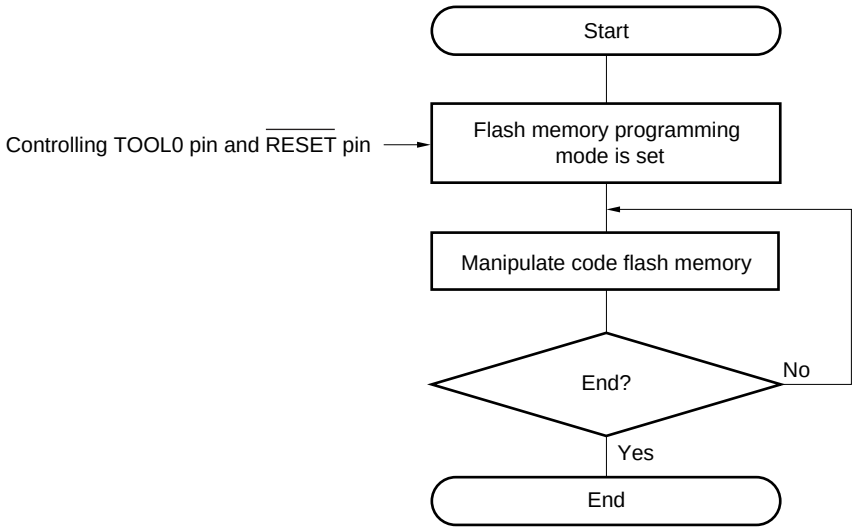
Supply the same other power supplies ($EV_{DD1}$, $EV_{SS1}$) as those $V_{DD}$ and $V_{SS}$.

## 33.4 Serial Programming Method

### 33.4.1 Serial programming procedure

The following figure illustrates a flow for rewriting the code flash memory through serial programming.

**Figure 33-6. Code Flash Memory Manipulation Procedure**



### 33.4.2 Flash memory programming mode

To rewrite the contents of the code flash memory through serial programming, specify the flash memory programming mode. To enter the mode, set as follows.

<Serial programming using the dedicated flash memory programmer >

Connect the RL78 microcontroller to a dedicated flash memory programmer. Communication from the dedicated flash memory programmer is performed to automatically switch to the flash memory programming mode.

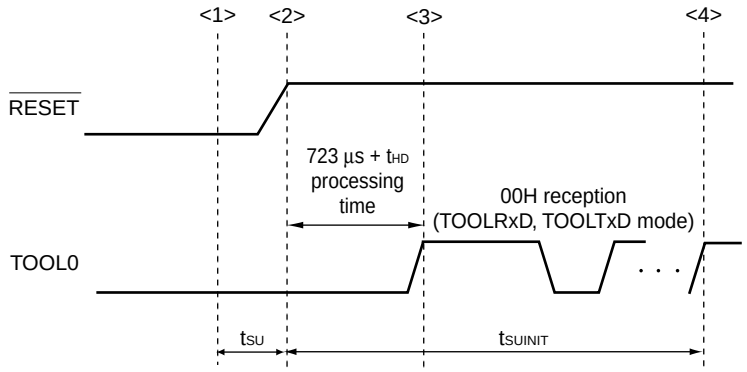
<Serial programming using an external device (UART communication)>

Set the TOOL0 pin to the low level, and then cancel the reset (see **Table 33-4**). After that, enter flash memory programming mode according to the procedures <1> to <4> shown in **Figure 33-7**. For details, refer to the **RL78 Microcontrollers (RL78 Protocol A) Programmer Edition Application Note (R01AN0815)**.

**Table 33-4. Relationship Between TOOL0 Pin and Operation Mode After Reset Release**

| TOOL0 | Operation Mode |
|---|---|
| $EV_{DD}$ | Normal operation mode |
| 0 | Flash memory programming mode |

**Figure 33-7. Setting of Flash Memory Programming Mode**




<1> The low level is input to the TOOL0 pin.

<2> The pins reset ends (POR and LVD reset must end before the pin reset ends.).

<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 segment shows that it is necessary to finish specifying the initial communication settings within 100 ms from when the resets end.

$t_{SU}$: How long from when the TOOL0 pin is placed at the low level until a pin reset ends

$t_{HD}$: How long to keep the TOOL0 pin at the low level from when the external and internal resets end (the flash firmware processing time is excluded)

For details, see **37.12 Timing Specs for Switching Flash Memory Programming Modes**.

There are two flash memory programming modes: wide voltage mode and full speed mode. The supply voltage value applied to the microcontroller during write operations and the setting information of the user option byte for setting of the flash memory programming mode determine which mode is selected.

When a dedicated flash memory programmer is used for serial programming, setting the voltage on GUI selects the mode automatically.

**Table 33-5. Programming Modes and Voltages at Which Data Can Be Written, Erased, or Verified**

| Power Supply Voltage ($V_{DD}$) | User Option Byte Setting for Switching to Flash Memory Programming Mode | | Flash Programming Mode |
|---|---|---|---|
| | Flash Operation Mode | Operating Frequency | |
| 2.7 V ≤ $V_{DD}$ ≤ 5.5 V | Blank state | | Full speed mode |
| | HS (high speed main) mode | 1 MHz to 24 MHz | Full speed mode |
| | LS (low speed main) mode | 1 MHz to 8 MHz | Wide voltage mode |
| 2.4 V ≤ $V_{DD}$ < 2.7 V | Blank state | | Full speed mode |
| | HS (high speed main) mode | 1 MHz to 16 MHz | Full speed mode |
| | LS (low speed main) mode | 1 MHz to 8 MHz | Wide voltage mode |
| 1.9 V ≤ $V_{DD}$ < 2.4 V | Blank state | | Wide voltage mode |
| | LS (low speed main) mode | 1 MHz to 8 MHz | Wide voltage mode |

**Remarks 1.** Using both the wide voltage mode and full speed mode imposes no restrictions on writing, deletion, or verification.

**2.** For details about communication commands, see **33.4.4 Communication commands**.

### 33.4.3 Selecting communication mode

Communication mode of the RL78 microcontroller as follows.

**Table 33-6. Communication Modes**

| Communication Mode | Standard Setting[Note 1] | | | | Pins Used |
|---|---|---|---|---|---|
| | Port | Speed[Note 2] | Frequency | Multiply Rate | |
| 1-line mode (when flash memory programmer is used or when external device is used) | UART | 115200 bps, 250000 bps, 500000 bps, 1 Mbps | – | – | TOOL0 |
| Dedicated UART (when external device is used) | UART | 115200 bps, 250000 bps, 500000 bps, 1 Mbps | – | – | TOOLTxD, TOOLRxD |

**Notes 1.** Selection items for Standard settings on GUI of the flash memory programmer.

**2.** Because factors other than the baud rate error, such as the signal waveform slew, also affect UART communication, thoroughly evaluate the slew as well as the baud rate error.

### 33.4.4 Communication commands

The RL78 microcontroller executes serial programming through the commands listed in **Table 33-7.**

The signals sent from the dedicated flash memory programmer or external device to the RL78 microcontroller are called commands, and programming functions corresponding to the commands are executed. For details, refer to the **RL78 Microcontrollers (RL78 Protocol A) Programmer Edition Application Note (R01AN0815)**.

**Table 33-7. Flash Memory Control Commands**

| Classification | Command Name | Function |
|---|---|---|
| Verify | Verify | Compares the contents of a specified area of the flash memory with data transmitted from the programmer. |
| Erase | Block Erase | Erases a specified area in the flash memory. |
| Blank check | Block Blank Check | Checks if a specified block in the flash memory has been correctly erased. |
| Write | Programming | Writes data to a specified area in the flash memory[Note]. |
| Getting information | Silicon Signature | Gets the RL78 microcontroller information (such as the part number, 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 | Release setting of prohibition of writing. |
| Others | Reset | Used to detect synchronization status of communication. |
| | Baud Rate Set | Sets baud rate when UART communication mode is selected. |

**Note** Confirm that no data has been written to the write area. Because data cannot be erased after block erase is prohibited, do not write data if the data has not been erased.

Product information (such as product name and firmware version) can be obtained by executing the “Silicon Signature” command.

**Table 33-8** is a list of signature data and **Table 33-9** shows an example of signature data.

**Table 33-8. Signature Data List**

| Field Name | Description | Number 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 last address | Last address of code flash memory area<br>(Sent from lower address.<br>Example. 00000H to 0FFFFH (64 KB) → FFH, 1FH, 00H) | 3 bytes |
| Firmware version | Version information of firmware for programming<br>(Sent from upper address.<br>Example. From Ver. 1.23 → 01H, 02H, 03H) | 3 bytes |

**Table 33-9. Example of Signature Data**

| Field Name | Description | Number of Transmit Data | Data (Hexadecimal) |
|---|---|---|---|
| Device code | RL78 protocol A | 3 bytes | 10<br>00<br>06 |
| Device name | R5F10MPG | 10 bytes | 52 = “R”<br>35 = “5”<br>46 = “F”<br>31 = “1”<br>30 = “0”<br>4D = “M”<br>50 = “P”<br>47 = “G”<br>20 = “ ”<br>20 = “ ” |
| Flash memory area last address | Flash memory area<br>00000H to 0FFFFH (64 KB) | 3 bytes | FF<br>FF<br>00 |
| Firmware version | Ver.1.23 | 3 bytes | 01<br>02<br>03 |

## 33.5 Self-Programming

The RL78 microcontroller supports a self-programming function that can be used to rewrite the code flash memory via a user program. Because this function allows a user application to rewrite the code flash memory by using the flash self-programming library, it can be used to upgrade the program in the field.

**Cautions 1. The self-programming function cannot be used when the CPU operates with the subsystem clock.**

**2. To prohibit an interrupt during self-programming, in the same way as in the normal operation mode, execute the self-programming library in the state where the IE flag is cleared (0) by the DI instruction. To enable an interrupt, clear (0) the interrupt mask flag to accept in the state where the IE flag is set (1) by the EI instruction, and then execute the self-programming library.**

**3. The high-speed on-chip oscillator needs to oscillate during self-programming. When stopping the high-speed on-chip oscillator, oscillate the high-speed on-chip oscillator clock (HIOSTOP = 0) and execute the self-programming library after 30 μs elapses.**

**4. The self-programming function cannot be used when the internal power is supplied from the VBAT pin.**

<R> **Remarks 1.** For details of the self-programming function, refer to **RL78 Microcontroller Flash Self Programming Library Type01 User's Manual (R01US0050).**

**2.** For details of the time required to execute self programming, see the notes on use that accompany the flash self programming library tool.

The self-programming function has two flash memory programming modes; wide voltage mode and full speed mode.

Specify the mode that corresponds to the flash operation mode specified in bits CMODE1 and CMODE0 in option byte 000C2H.

Set to full speed mode when the HS (high speed main) mode is specified. Set to wide voltage mode when the LS (low speed main) mode is specified.

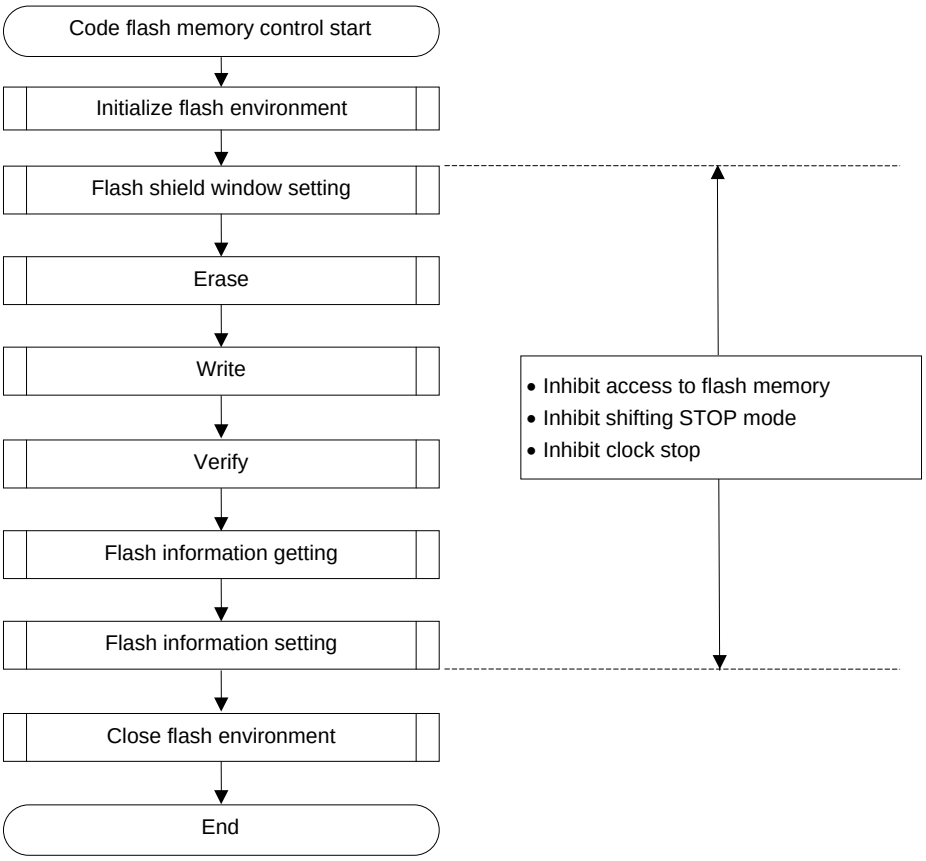
If the argument fsl_flash_voltage_u08 is 00H when the FSL_Init function of the flash self-programming library provided by Renesas Electronics is executed, full speed mode is specified. If the argument is other than 00H, the wide voltage mode is specified.

**Remark** Using both the wide voltage mode and full speed mode imposes no restrictions on writing, deletion, or verification.

### 33.5.1 Self-programming procedure

The following figure illustrates a flow for rewriting the code flash memory by using a flash self-programming library.

**Figure 33-8. Flow of Self Programming (Rewriting Flash Memory)**



### 33.5.2 Boot swap function

If rewriting the boot area 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.

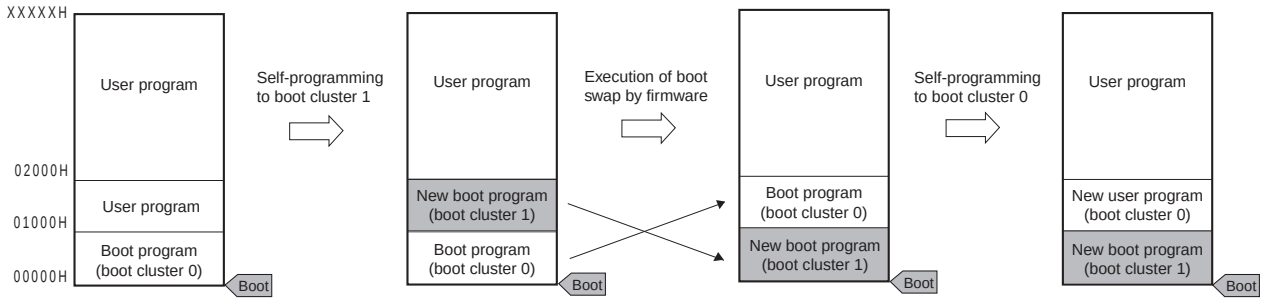
The boot swap function is used to avoid this problem.

Before erasing boot cluster 0[Note], which is a boot program area, by self-programming, write a new boot program to boot cluster 1 in advance. When the program has been correctly written to boot cluster 1, swap this boot cluster 1 and boot cluster 0 by using the set information function of the firmware of the RL78 microcontroller, so that boot cluster 1 is used as a boot area. After that, erase or write the original area, 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 to be swapped when the program is reset and started next.

**Note** A boot cluster is a 4 KB area and boot clusters 0 and 1 are swapped by the boot swap function.

**Figure 33-9. Boot Swap Function**

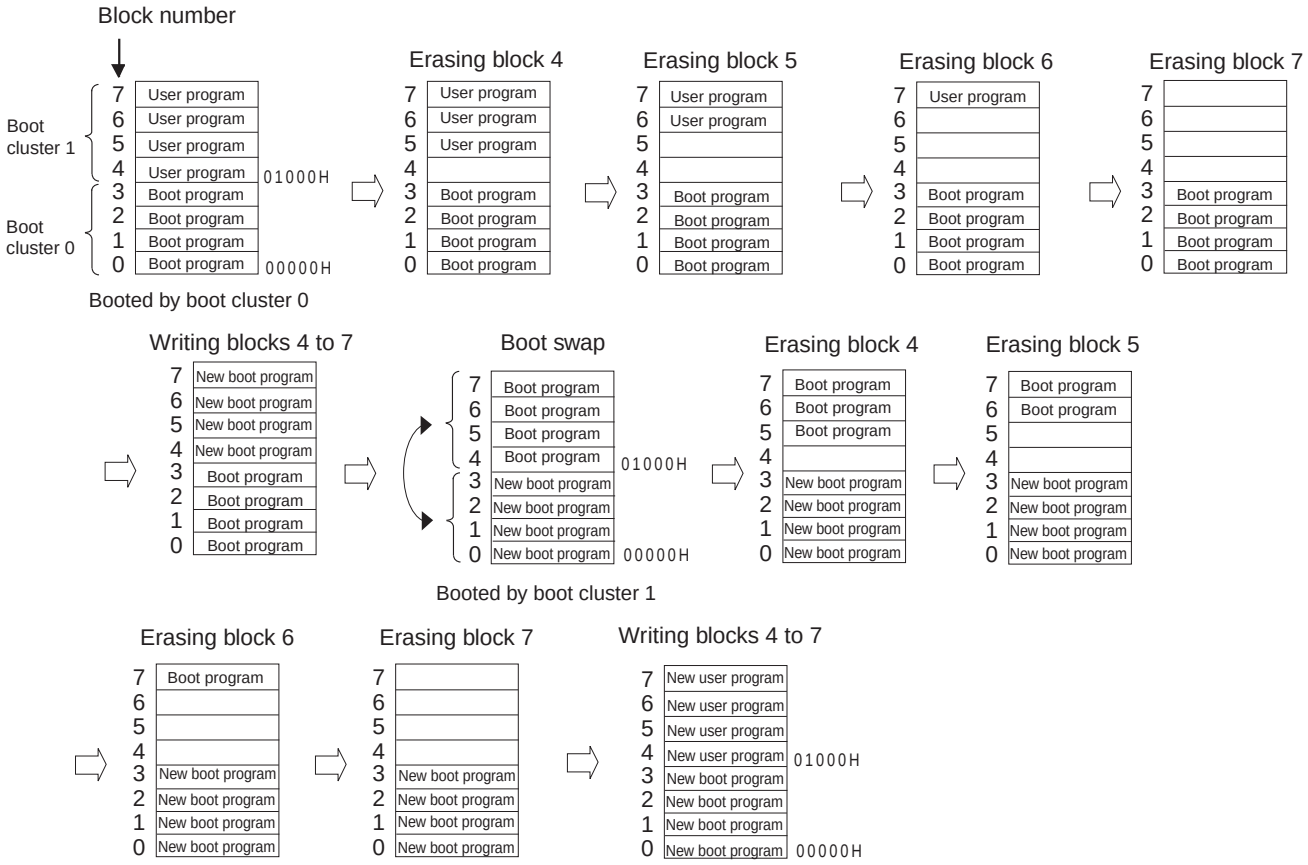



In an example of above figure, it is as follows.

Boot cluster 0: Boot area before boot swap

Boot cluster 1: Boot area after boot swap

**Figure 33-10. Example of Executing Boot Swapping**



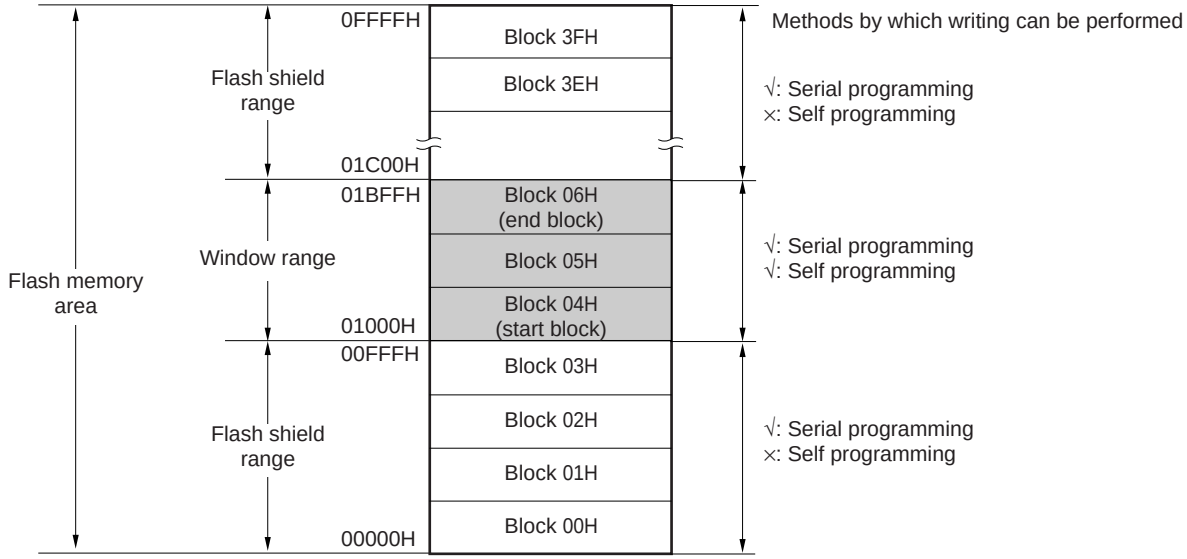
### 33.5.3 Flash shield window function

The flash shield window function is provided as one of the security functions for self programming. It disables writing to and erasing areas outside the range specified as a window only during self programming.

The window range can be set by specifying the start and end blocks. The window range can be set or changed during serial programming and self programming.

Writing to and erasing areas outside the window range are disabled during self programming. During serial programming, however, areas outside the range specified as a window can be written and erased.

**Figure 33-11. Flash Shield Window Setting Example**
**(Target Devices: R5F10MME, R5F10MPE, Start Block: 04H, End Block: 06H)**




**Caution If the rewrite-prohibited area of the boot cluster 0 overlaps with the flash shield window range, prohibition to rewrite the boot cluster 0 takes priority.**

**Table 33-10. Relationship Between Flash Shield Window Function Setting/Change Methods and Commands**

| Programming Conditions | Window Range Setting/Change Methods | Execution Commands | |
|---|---|---|---|
| | | Block Erase | Write |
| Self-programming | Specify the starting and ending blocks by the flash self programming library. | Block erasing is enabled only within the window range. | Writing is enabled only within the range of window range. |
| Serial programming | Specify the starting and ending blocks on GUI of dedicated flash memory programmer, etc. | Block erasing is enabled also outside the window range. | Writing is enabled also outside the window range. |

**Remark** See **33.6 Security Settings** to prohibit writing/erasing during serial programming.

## 33.6 Security Settings

The RL78 microcontroller 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 using the Security Set command.

- Disabling block erase
  Execution of the block erase command for a specific block in the flash memory is prohibited during serial programming. However, blocks can be erased by means of self programming.

- Disabling write
  Execution of the write command for entire blocks in the code flash memory is prohibited during serial programming. However, blocks can be written by means of self programming.
  After the setting of prohibition of writing is specified, releasing the setting by the Security Release command is enabled by a reset.

- Disabling rewriting boot cluster 0
  Execution of the block erase command and write command on boot cluster 0 (00000H to 00FFFH) in the code flash memory is prohibited by this setting.

The block erase, write commands and rewriting boot cluster 0 are enabled by the default setting when the flash memory is shipped. Security can be set by serial programming and self programming. Each security setting can be used in combination.

Table 33-11 shows the relationship between the erase and write commands when the RL78 microcontroller security function is enabled.

**Caution The security function of the flash programmer does not support self-programming.**

**Remark** To prohibit writing and erasing during self-programming, use the flash shield window function (see **33.5.3** for detail).

**Table 33-11. Relationship Between Enabling Security Function and Command**

**(1) During serial programming**

| Valid Security | Executed Command | |
|---|---|---|
| | Block Erase | Write |
| Prohibition of block erase | Blocks cannot be erased. | Can be performed.[Note] |
| Prohibition of writing | Blocks can be erased. | Cannot be performed. |
| Prohibition of rewriting boot cluster 0 | Boot cluster 0 cannot be erased. | Boot cluster 0 cannot be written. |

**Note** Confirm that no data has been written to the write area. Because data cannot be erased after block erase is prohibited, do not write data if the data has not been erased.

**(2) During self programming**

| Valid Security | Executed Command | |
|---|---|---|
| | Block Erase | Write |
| Prohibition of block erase | Blocks can be erased. | Can be performed. |
| Prohibition of writing | | |
| Prohibition of rewriting boot cluster 0 | Boot cluster 0 cannot be erased. | Boot cluster 0 cannot be written. |

**Remark** To prohibit writing and erasing during self-programming, use the flash shield window function (see **33.5.3** for detail).

**Table 33-12. Setting Security in Each Programming Mode**

**(1) During serial programming**

| Security | Security Setting | How to Disable Security Setting |
|---|---|---|
| Prohibition of block erase | Set via GUI of dedicated flash memory programmer, etc. | Cannot be disabled after set. |
| Prohibition of writing | | Set via GUI of dedicated flash memory programmer, etc. |
| Prohibition of rewriting boot cluster 0 | | Cannot be disabled after set. |

**Caution Releasing the setting of prohibition of writing is enabled only when the security is not set as the block erase prohibition and the boot cluster 0 rewrite prohibition with code flash memory area being blanks.**

**(2) During self programming**

| Security | Security Setting | How to Disable Security Setting |
|---|---|---|
| Prohibition of block erase | Set by using flash self programming library. | Cannot be disabled after set. |
| Prohibition of writing | | Cannot be disabled during self-programming (set via GUI of dedicated flash memory programmer, etc. during serial programming). |
| Prohibition of rewriting boot cluster 0 | | Cannot be disabled after set. |

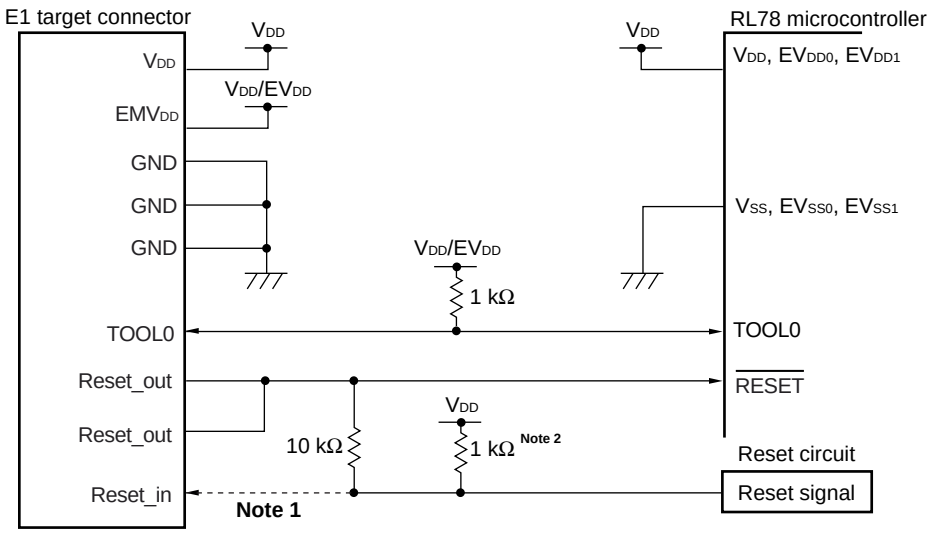
# CHAPTER 34 ON-CHIP DEBUG FUNCTION

## 34.1 Connecting E1 On-chip Debugging Emulator

The RL78 microcontroller uses the $V_{DD}$, $\overline{RESET}$, TOOL0, and $V_{SS}$ pins to communicate with the host machine via an E1 on-chip debugging emulator. Serial communication is performed by using a single-line UART that uses the TOOL0 pin.

**Caution The RL78 microcontroller has an on-chip debug function, which is provided for development and evaluation. Do not use the on-chip debug function in products designated for mass production, because the guaranteed number of rewritable times of the flash memory may be exceeded when this function is used, and product reliability therefore cannot be guaranteed. Renesas Electronics is not liable for problems occurring when the on-chip debug function is used.**
**Also, note that the debug function is disabled when power is supplied from the VBAT pin with the battery backup function.**

**Figure 34-1. Connection Example of E1 On-chip Debugging Emulator**




**Notes 1.** Connecting the dotted line is not necessary during serial programming.

**2.** If the reset circuit on the target system does not have a buffer and generates a reset signal only with resistors and capacitors, this pull-up resistor is not necessary.

**Caution This circuit diagram is assumed that the reset signal outputs from an N-ch O.D. buffer (output resistor: 100 Ω or less)**

**Remark** With products not provided with an $EV_{DD0}$, $EV_{DD1}$, $EV_{SS0}$, or $EV_{SS1}$ pin, replace $EV_{DD0}$ and $EV_{DD1}$ with $V_{DD}$, or replace $EV_{SS0}$ and $EV_{SS1}$ with $V_{SS}$.

## 34.2 On-Chip Debug Security ID

The RL78 microcontroller has an on-chip debug operation control bit in the flash memory at 000C3H (see **CHAPTER 32 OPTION BYTE**) and an on-chip debug security ID setting area at 000C4H to 000CDH, to prevent third parties from reading memory content.

When the boot swap function is used, also set a value that is the same as that of 010C3H and 010C4H to 010CDH in advance, because 000C3H, 000C4H to 000CDH and 010C3H, and 010C4H to 010CDH are switched.

**Table 34-1. On-Chip Debug Security ID**

| Address | On-Chip Debug Security ID |
|---|---|
| 000C4H to 000CDH | Any ID code of 10 bytes |
| 010C4H to 010CDH | |

## 34.3 Securing of User Resources

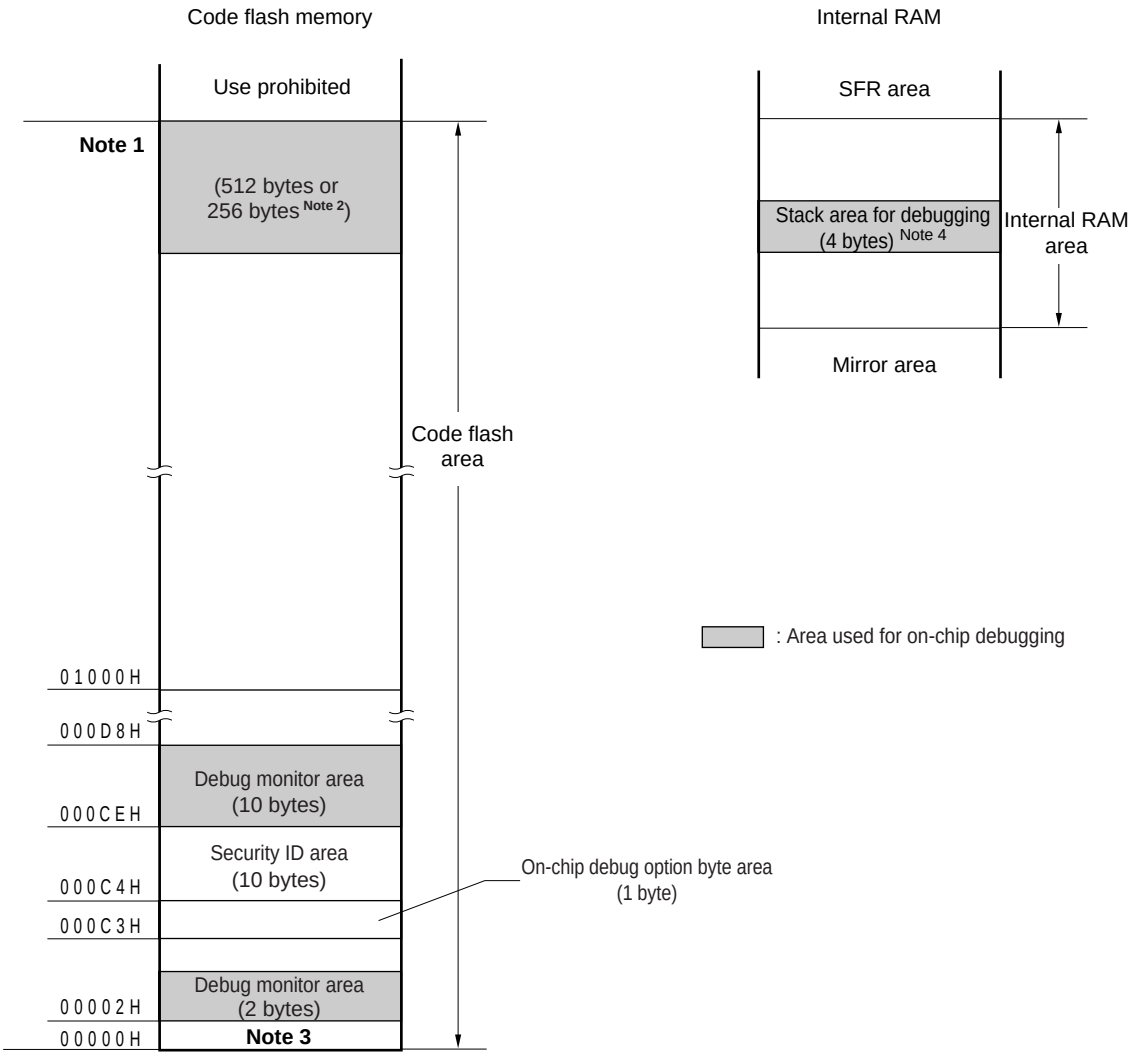
To perform communication between the RL78 microcontroller and E1 on-chip debugging emulator, as well as each debug function, the securing of memory space must be done beforehand.

If Renesas Electronics assembler or compiler is used, the items can be set by using link options.

**(1) Securement of memory space**

The shaded portions in Figure 34-2 are the areas reserved for placing the debug monitor program, so user programs or data cannot be allocated in these spaces. When using the on-chip debug function, these spaces must be secured so as not to be used by the user program. Moreover, this area must not be rewritten by the user program.

**Figure 34-2. Memory Spaces Where Debug Monitor Programs Are Allocated**




**Notes 1.** Address differs depending on products as follows.

| Products (code flash memory capacity) | Address of **Note 1** |
|---|---|
| R5F10MME, R5F10MPE | 0FFFFH |
| R5F10MMG, R5F10MPG | 1FFFFH |

**2.** When real-time RAM monitor (RRM) function and dynamic memory modification (DMM) function are not used, it is 256 bytes.

**3.** In debugging, reset vector is rewritten to address allocated to a monitor program.

**4.** Since this area is allocated immediately before the stack area, the address of this area varies depending on the stack increase and decrease. That is, 4 extra bytes are consumed for the stack area used. When using self-programming, 12 extra bytes are consumed for the stack area used.

# CHAPTER 35 BCD CORRECTION CIRCUIT

## 35.1 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).

## 35.2 Registers Used by BCD Correction Circuit

The BCD correction circuit uses the following registers.

- BCD correction result register (BCDADJ)

### 35.2.1 BCD correction result register (BCDADJ)

The BCDADJ register stores 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.

Reset input sets this register to undefined.

**Figure 35-1. Format of BCD Correction Result Register (BCDADJ)**

Address: F00FEH After reset: undefined R

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| BCDADJ | | | | | | | | |

## 35.3 BCD Correction Circuit Operation

The basic operation of the BCD correction circuit is as follows.

**(1) Addition: Calculating the result of adding a BCD code value and another BCD code value by using a BCD code 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 value 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 flag and AC flag) is restored by the RETI instruction.**

An example is shown below.

Examples 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 | – |

Examples 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 | – |

Examples 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: Calculating the result of subtracting a BCD code value from another BCD code value by using a BCD code 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 flag and AC flag) 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 | – |

# CHAPTER 36 INSTRUCTION SET

This chapter 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).**

## 36.1 Conventions Used in Operation List

### 36.1.1 Operand identifiers and specification 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 36-1. Operand Identifiers and Specification Methods**

| Identifier | Description Method |
|---|---|
| r<br>rp<br>sfr<br>sfrp | X (R0), A (R1), C (R2), B (R3), E (R4), D (R5), L (R6), H (R7)<br>AX (RP0), BC (RP1), DE (RP2), HL (RP3)<br>Special-function register symbol (SFR symbol) FFF00H to FFFFFH<br>Special-function register symbols (16-bit manipulatable SFR symbol. Even addresses only[Note]) FFF00H to FFFFFH |
| saddr<br>saddrp | FFE20H to FFF1FH Immediate data or labels<br>FFE20H to FF1FH Immediate data or labels (even addresses only[Note]) |
| addr20<br>addr16<br>addr5 | 00000H to FFFFFH Immediate data or labels<br>0000H to FFFFH Immediate data or labels (only even addresses for 16-bit data transfer instructions[Note])<br>0080H to 00BFH Immediate data or labels (even addresses only) |
| word<br>byte<br>bit | 16-bit immediate data or label<br>8-bit immediate data or label<br>1-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 SFR List** 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 Extended SFR (2nd SFR) List** for the symbols of the extended special function registers.

### 36.1.2 Description of operation column

The operation when the instruction is executed is shown in the “Operation” column using the following symbols.

**Table 36-2. 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 |
| $X_H$, $X_L$<br>$X_S$, $X_H$, $X_L$ | 16-bit registers: $X_H$ = higher 8 bits, $X_L$ = lower 8 bits<br>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) |

### 36.1.3 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 36-3. Symbols in “Flag” Column**

| Symbol | Change of Flag Value |
|---|---|
| (Blank) | Unchanged |
| 0 | Cleared to 0 |
| 1 | Set to 1 |
| × | Set/cleared according to the result |
| R | Previously saved value is restored |

### 36.1.4 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.

A interrupt and DTC transfer are not acknowledged between a PREFIX instruction code and the instruction immediately after.

**Table 36-4. 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.**

## 36.2 Operation List

**Table 36-5. Operation List (1/18)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | 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, A Note 3 | 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 | | | |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ($f_{CLK}$) when the code flash area is accessed.

**3.** Except r = A

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 36-5. Operation List (2/18)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | 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 | |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ($f_{CLK}$) when the code flash area is accessed.

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 36-5. Operation List (3/18)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | 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)<br>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) | |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ($f_{CLK}$) when the code flash area is accessed.

**3.** Except r = A

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 36-5. Operation List (4/18)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | 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, AX Note 3 | 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 | |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ($f_{CLK}$) when the code flash area is accessed.

**3.** Except rp = AX

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 36-5. Operation List (5/18)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | 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 | |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ($f_{CLK}$) when the code flash area is accessed.

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 36-5. Operation List (6/18)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | 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) | × × × |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ($f_{CLK}$) when the code flash area is accessed.

**3.** Except rp = AX

**4.** Except r = A

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 36-5. Operation List (7/18)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | 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, rv 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) | × | × | × |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ($f_{CLK}$) when the code flash area is accessed.

**3.** Except r = A

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 36-5. Operation List (8/18)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | 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) | × | | |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ($f_{CLK}$) when the code flash area is accessed.

**3.** Except r = A

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 36-5. Operation List (9/18)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | 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∨(H) | × |
| | | 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) | × |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ($f_{CLK}$) when the code flash area is accessed.

**3.** Except r = A

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 36-5. Operation List (10/18)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | 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 Note3 | 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) | × | × | × |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ($f_{CLK}$) when the code flash area is accessed.

**3.** Except r = A

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 36-5. Operation List (11/18)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | 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) | × | × | × |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ($f_{CLK}$) when the code flash area is accessed.

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 36-5. Operation List (12/18)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Operation | Flag | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | **Note 1** | **Note 2** | | Z | AC | CY |
| Multiply, Divide, Multiply & accumu-late | 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) | | × | × |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ($f_{CLK}$) when the code flash area is accessed.

<R> **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. 1.71 and later versions of the CA78K0R (Renesas Electronics compiler), for both C and assembly language source code**

**- Service pack 1.40.6 and later versions of the EWRL78 (IAR compiler), for C language source code**

**- GNURL78 (KPIT compiler), for C language source code**

**Remarks 1.** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**2.** MACR indicates the multiplication and accumulation register (MACRH, MACRL).

**Table 36-5. Operation List (13/18)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | 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 | | | × |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ($f_{CLK}$) when the code flash area is accessed.

**Remarks 1.** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**2.** cnt indicates the bit shift count.

**Table 36-5. Operation List (14/18)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | 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 | – | CYX ← 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 | × |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ($f_{CLK}$) when the code flash area is accessed.

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 36-5. Operation List (15/18)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | 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 ← $\overline{CY}$ | | | × |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed

**2.** Number of CPU clocks ($f_{CLK}$) when the code flash area is accessed.

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 36-5. Operation List (16/18)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | 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 |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ($f_{CLK}$) when the code flash area is accessed.

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 36-5. Operation List (17/18)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | 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 | – | $rp_L$ ←(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 | |
| Un-conditional 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 | 2/4 Note3 | – | PC ← PC + 2 + jdisp8 if CY = 1 | |
| | BNC | $addr20 | 2 | 2/4 Note3 | – | PC ← PC + 2 + jdisp8 if CY = 0 | |
| | BZ | $addr20 | 2 | 2/4 Note3 | – | PC ← PC + 2 + jdisp8 if Z = 1 | |
| | BNZ | $addr20 | 2 | 2/4 Note3 | – | PC ← PC + 2 + jdisp8 if Z = 0 | |
| | BH | $addr20 | 3 | 2/4 Note3 | – | PC ← PC + 3 + jdisp8 if (Z∨CY)=0 | |
| | BNH | $addr20 | 3 | 2/4 Note3 | – | PC ← PC + 3 + jdisp8 if (Z∨CY)=1 | |
| | BT | saddr.bit, $addr20 | 4 | 3/5 Note3 | – | PC ← PC + 4 + jdisp8 if (saddr).bit = 1 | |
| | | sfr.bit, $addr20 | 4 | 3/5 Note3 | – | PC ← PC + 4 + jdisp8 if sfr.bit = 1 | |
| | | A.bit, $addr20 | 3 | 3/5 Note3 | – | PC ← PC + 3 + jdisp8 if A.bit = 1 | |
| | | PSW.bit, $addr20 | 4 | 3/5 Note3 | – | PC ← PC + 4 + jdisp8 if PSW.bit = 1 | |
| | | [HL].bit, $addr20 | 3 | 3/5 Note3 | 6/7 | PC ← PC + 3 + jdisp8 if (HL).bit = 1 | |
| | | ES:[HL].bit, $addr20 | 4 | 4/6 Note3 | 7/8 | PC ← PC + 4 + jdisp8 if (ES, HL).bit = 1 | |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ($f_{CLK}$) when the code flash area is accessed.

**3.** This indicates the number of clocks “when condition is not met/when condition is met”.

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 36-5. Operation List (18/18)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | Flag |
|---|---|---|---|---|---|---|---|
| | | | | Note 1 | Note 2 | | Z AC CY |
| Conditional branch | BF | saddr.bit, $addr20 | 4 | 3/5 Note3 | – | PC ← PC + 4 + jdisp8 if (saddr).bit = 0 | |
| | | sfr.bit, $addr20 | 4 | 3/5 Note3 | – | PC ← PC + 4 + jdisp8 if sfr.bit = 0 | |
| | | A.bit, $addr20 | 3 | 3/5 Note3 | – | PC ← PC + 3 + jdisp8 if A.bit = 0 | |
| | | PSW.bit, $addr20 | 4 | 3/5 Note3 | – | PC ← PC + 4 + jdisp8 if PSW.bit = 0 | |
| | | [HL].bit, $addr20 | 3 | 3/5 Note3 | 6/7 | PC ← PC + 3 + jdisp8 if (HL).bit = 0 | |
| | | ES:[HL].bit, $addr20 | 4 | 4/6 Note3 | 7/8 | PC ← PC + 4 + jdisp8 if (ES, HL).bit = 0 | |
| | BTCLR | saddr.bit, $addr20 | 4 | 3/5 Note3 | – | PC ← PC + 4 + jdisp8 if (saddr).bit = 1 then reset (saddr).bit | |
| | | sfr.bit, $addr20 | 4 | 3/5 Note3 | – | PC ← PC + 4 + jdisp8 if sfr.bit = 1 then reset sfr.bit | |
| | | A.bit, $addr20 | 3 | 3/5 Note3 | – | PC ← PC + 3 + jdisp8 if A.bit = 1 then reset A.bit | |
| | | PSW.bit, $addr20 | 4 | 3/5 Note3 | – | PC ← PC + 4 + jdisp8 if PSW.bit = 1 then reset PSW.bit | × × × |
| | | [HL].bit, $addr20 | 3 | 3/5 Note3 | – | PC ← PC + 3 + jdisp8 if (HL).bit = 1 then reset (HL).bit | |
| | | ES:[HL].bit, $addr20 | 4 | 4/6 Note3 | – | 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 Note4 | RBn | 2 | 1 | – | RBS[1:0] ← 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 | |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ($f_{CLK}$) when the code flash area is accessed.

**3.** This indicates the number of clocks “when condition is not met/when condition is met”.

**4.** n indicates the number of register banks (n = 0 to 3).

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

# CHAPTER 37 ELECTRICAL SPECIFICATIONS

**Cautions 1. The RL78 microcontrollers have an on-chip debug function, which is provided for development and evaluation. Do not use the on-chip debug function in products designated for mass production, because the guaranteed number of rewritable times of the flash memory may be exceeded when this function is used, and product reliability therefore cannot be guaranteed. Renesas Electronics is not liable for problems occurring when the on-chip debug function is used.**

**2. The pins mounted depend on the product. See 2.1 Port Function List to 2.2.1 With functions for each product.**

**Remarks 1.** In the descriptions in this chapter, read $EV_{DD}$ as $EV_{DD0}$ and $EV_{DD1}$, and $EV_{SS}$ as $EV_{SS0}$ and $EV_{SS1}$.

**2.** For 80-pin products, read $EV_{DD}$ as $V_{DD}$ and $EV_{SS}$ as $V_{SS}$.

## 37.1 Absolute Maximum Ratings

### Absolute Maximum Ratings (1/3)

| Parameter | Symbols | Conditions | Ratings | Unit |
|---|---|---|---|---|
| Supply voltage | $V_{DD}$ | | −0.5 to +6.5 | V |
| | $EV_{DD}$ | $EV_{DD1} = V_{DD}$ | −0.5 to +6.5 | V |
| | VBAT | | −0.5 to +6.5 | V |
| | $AV_{DD}$ | | −0.5 to +6.5 and −0.5 to $V_{DD}$ Note 4 +0.6 | V |
| REGC pin input voltage | $V_{IREGC}$ | REGC | −0.3 to +2.8 and −0.3 to $V_{DD}$ Note 4 +0.3 Note 1 | V |
| Input voltage | $V_{I1}$ | P00 to P07, P10 to P17, P30 to P37, P40 to P44, P50 to P57, P70 to P77, P80 to P85, P125 to P127 | −0.3 to $EV_{DD}$ +0.3 and −0.3 to $V_{DD}$ Note 4 +0.3 Note 2 | V |
| | $V_{I2}$ | P60 to P62 (N-ch open-drain) | −0.3 to +6.5 | V |
| | $V_{I3}$ | P20 to P25, P121 to P124, P137, EXCLK, EXCLKS | −0.3 to $V_{DD}$ Note 4 +0.3 Note 2 | V |
| | $V_{I4}$ | $\overline{\text{RESET}}$ | −0.3 to +6.5 | V |
| Output voltage | $V_{O1}$ | P00 to P07, P10 to P17, P30 to P37, P40 to P44, P50 to P57, P60 to P62, P70 to P77, P80 to P85, P125 to P127, P130 | −0.3 to $EV_{DD}$ +0.3 and −0.3 to $V_{DD}$ Note 4 +0.3 Note 2 | V |
| | $V_{O2}$ | P20 to P25 | −0.3 to $V_{DD}$ Note 4 +0.3 Note 2 | V |
| Analog input voltage | $V_{AI1}$ | ANI0 to ANI5 | −0.3 to $V_{DD}$ Note 4 +0.3 and −0.3 to $AV_{REF(+)}$ +0.3 Notes 2, 3 | V |
| | $V_{AI2}$ | ANIP0 to ANIP3, ANIN0 to ANIN3 | −0.6 to +2.8 and −0.6 to AREGC +0.3 Note 5 | V |
| Reference supply voltage | $V_{IDSAD}$ | AREGC, AVCM, AVRT | −0.3 to +2.8 and −0.3 to $AV_{DD}$ +0.3 Note 6 | V |

**Notes 1.** Connect the REGC pin to Vss via a capacitor (0.47 to 1 μF). This value regulates the absolute maximum rating of the REGC pin. Do not use this pin with voltage applied to it.

**2.** Must be 6.5 V or lower.

**3.** Do not exceed $AV_{REF(+)}$ + 0.3 V in case of A/D conversion target pin.

**4.** The power supply voltage (VBAT pin or $V_{DD}$ pin) selected by the battery backup feature.

**5.** The ΔΣ A/D conversion target pin must not exceed AREGC +0.3 V.

**6.** Connect AREGC, AVCM, and AVRT terminals to $V_{SS}$ via capacitor (0.47 μF).
This value defines the absolute maximum rating of AREGC, AVCM, and AVRT terminal. Do not use with voltage applied.

**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.**

**Remarks 1.** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

**2.** $AV_{REF(+)}$: + side reference voltage of the A/D converter.

**3.** $V_{SS}$: Reference voltage

**Absolute Maximum Ratings (2/3)**

| Parameter | Symbols | Conditions | | Ratings | Unit |
|---|---|---|---|---|---|
| LCD voltage | $V_{LI1}$ | $V_{L1}$ voltage[Note 1] | | −0.3 to 2.8 and −0.3 to $V_{L4}$ +0.3 | V |
| | $V_{LI2}$ | $V_{L2}$ voltage[Note 1] | | −0.3 to $V_{L4}$ +0.3[Note 2] | V |
| | $V_{LI3}$ | $V_{L3}$ voltage[Note 1] | | −0.3 to $V_{L4}$ +0.3[Note 2] | V |
| | $V_{LI4}$ | $V_{L4}$ voltage[Note 1] | | −0.3 to +6.5 | V |
| | $V_{LCAP}$ | CAPL, CAPH voltage[Note 1] | | −0.3 to $V_{L4}$ +0.3[Note 2] | V |
| | $V_{OUT}$ | COM0 to COM7, SEG0 to SEG41, output voltage | External resistance division method | −0.3 to $V_{DD}$[Note 3] +0.3[Note 2] | V |
| | | | Capacitor split method | −0.3 to $V_{DD}$[Note 3] +0.3[Note 2] | V |
| | | | Internal voltage boosting method | −0.3 to $V_{L4}$ +0.3[Note 2] | V |

**Notes 1.** This value only indicates the absolute maximum ratings when applying voltage to the $V_{L1}$, $V_{L2}$, $V_{L3}$, and $V_{L4}$ pins; it does not mean that applying voltage to these pins is recommended. When using the internal voltage boosting method or capacitance split method, connect these pins to $V_{SS}$ via a capacitor (0.47 μF ± 30%) and connect a capacitor (0.47 μF ± 30%) between the CAPL and CAPH pins.

**2.** Must be 6.5 V or lower.

**3.** The power supply voltage (VBAT pin or $V_{DD}$ pin) selected by the battery backup feature.

**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** $V_{SS}$: Reference voltage

**Absolute Maximum Ratings (3/3)**

| Parameter | Symbols | Conditions | | Ratings | Unit |
|---|---|---|---|---|---|
| Output current, high | $I_{OH1}$ | Per pin | P00 to P07, P10 to P17, P30 to P37, P40 to P44, P50 to P57, P70 to P77, P80 to P85, P125 to P127, P130 | −40 | mA |
| | | Total of all pins −170 mA | P00 to P07, P40 to P44, P130 | −70 | mA |
| | | | P10 to P17, P30 to P37, P50 to P57, P70 to P77, P80 to P85, P125 to P127 | −100 | mA |
| | $I_{OH2}$ | Per pin | P20 to P25 | −0.5 | mA |
| | | Total of all pins | | −2 | mA |
| Output current, low | $I_{OL1}$ | Per pin | P00 to P07, P10 to P17, P30 to P37, P40 to P44, P50 to P57, P60 to P62, P70 to P77, P80 to P85, P125 to P127, P130 | 40 | mA |
| | | Total of all pins 170 mA | P00 to P07, P40 to P44, P130 | 70 | mA |
| | | | P10 to P17, P30 to P37, P50 to P57, P60 to P62, P70 to P77, P80 to P85, P125 to P127 | 100 | mA |
| | $I_{OL2}$ | Per pin | P20 to P25 | 1 | mA |
| | | Total of all pins | | 5 | mA |
| Operating ambient temperature | $T_A$ | In normal operation mode | | −40 to +85 | °C |
| | | In flash memory programming mode | | | |
| 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** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

## 37.2 Oscillator Characteristics

### 37.2.1 X1, XT1 oscillator characteristics

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = 0 V)**

| Parameter | Resonator | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| X1 clock oscillation frequency ($f_X$) Notes 1, 2 | Ceramic resonator/ crystal resonator | 2.7 V ≤ $V_{DD}$ ≤ 5.5 V | 1.0 | | 20.0 | MHz |
| | | 2.4 V ≤ $V_{DD}$ < 2.7 V | 1.0 | | 16.0 | MHz |
| | | 1.9 V ≤ $V_{DD}$ < 2.4 V | 1.0 | | 8.0 | MHz |
| XT1 clock oscillation frequency ($f_{XT}$) Notes 1, 2 | Crystal resonator | | 32 | 32.768 | 35 | kHz |

**Notes 1.** Indicates only permissible oscillator frequency ranges. See **37.4 AC Characteristics** for instruction execution time. Request evaluation by the manufacturer of the oscillator circuit mounted on a board to check the oscillator characteristics.

**2.** Voltage range is the power supply voltage (VBAT pin or $V_{DD}$ pin) selected by the battery backup function.

**Caution Since the CPU is started by the high-speed on-chip oscillator clock after a reset release, check the X1 clock oscillation stabilization time using the oscillation stabilization time counter status register (OSTC) by the user. Determine the oscillation stabilization time of the OSTC register and the oscillation stabilization time select register (OSTS) after sufficiently evaluating the oscillation stabilization time with the resonator to be used.**

**Remark** When using the X1 oscillator and XT1 oscillator, see **5.4 System Clock Oscillator**.

### 37.2.2 On-chip oscillator characteristics

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Oscillators | Parameters | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| High-speed on-chip oscillator clock frequency Notes 1, 2 | $f_{IH}$ | | | 3 | | 24 | MHz |
| High-speed on-chip oscillator clock frequency accuracy | | –20 to +85°C | 1.9 V ≤ $V_{DD}$ Note 3 ≤ 5.5 V | –1.0 | | +1.0 | % |
| | | –40 to –20°C | 1.9 V ≤ $V_{DD}$ Note 3 ≤ 5.5 V | –1.5 | | +1.5 | % |
| Low-speed on-chip oscillator clock frequency | $f_{IL}$ | | | | 15 | | kHz |
| Low-speed on-chip oscillator clock frequency accuracy | | | | –15 | | +15 | % |

**Notes 1.** The high-speed on-chip oscillator frequency is selected by using bits 0 to 3 of option byte (000C2H/010C2H) and bits 0 to 2 of the HOCODIV register.

**2.** This indicates the oscillator characteristics only. See **37.4 AC Characteristics** for the instruction execution time.

**3.** The power supply voltage (VBAT pin or $V_{DD}$ pin) selected by the battery backup feature.

## 37.3 DC Characteristics

### 37.3.1 Pin characteristics

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Items | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Output current, high[Note 1] | $I_{OH1}$ | Per pin for P00 to P07, P10 to P17, P30 to P37, P40 to P44, P50 to P57, P70 to P77, P80 to P85, P125 to P127, P130 | 1.9 V ≤ $EV_{DD}$ ≤ 5.5 V | | | –10.0[Note 2] | mA |
| | | Total of P00 to P07, P40 to P44, P130 (When duty = 70%[Note 3]) | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V | | | –55.0 | mA |
| | | | 2.7 V ≤ $EV_{DD}$ < 4.0 V | | | –10.0 | mA |
| | | | 1.9 V ≤ $EV_{DD}$ < 2.7 V | | | –5.0 | mA |
| | | Total of P10 to P17, P30 to P37, P50 to P57, P70 to P77, P80 to P85, P125 to P127 (When duty = 70%[Note 3]) | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V | | | –80.0 | mA |
| | | | 2.7 V ≤ $EV_{DD}$ < 4.0 V | | | –19.0 | mA |
| | | | 1.9 V ≤ $EV_{DD}$ < 2.7 V | | | –10.0 | mA |
| | | Total of all pins (When duty = 70%[Note 3]) | | | | –100.0 | mA |
| | $I_{OH2}$ | Per pin for P20 to P25 | 1.9 V ≤ $V_{DD}$[Note 4] ≤ 5.5 V | | | –0.1[Note 2] | mA |
| | | Total of all pins (When duty = 70%[Note 3]) | 1.9 V ≤ $V_{DD}$[Note 4] ≤ 5.5 V | | | –0.6 | mA |

**Notes 1.** Value of current at which the device operation is guaranteed even if the current flows from the $EV_{DD}$ and $V_{DD}$ pins to an output pin.

**2.** Do not exceed the total current value.

**3.** Specification under conditions where the duty factor ≤ 70%.
The output current value that has changed to the duty factor > 70% the duty ratio can be calculated with the following expression (when changing the duty factor from 70% to n%).

- Total output current of pins = ($I_{OH}$ × 0.7)/(n × 0.01)

<Example> Where n = 80% and $I_{OH}$ = –10.0 mA

Total output current of pins = (–10.0 × 0.7)/(80 × 0.01) ≅ –8.7 mA

However, the current that is allowed to flow into one pin does not vary depending on the duty factor.
A current higher than the absolute maximum rating must not flow into one pin.

**4.** The power supply voltage (VBAT pin or $V_{DD}$ pin) selected by the battery backup feature.

**Caution P01 to P07, P15 to P17, and P80 to P82 do not output high level in N-ch open-drain mode.**

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Items | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Output current, low[Note 1] | $I_{OL1}$ | Per pin for P00 to P07, P10 to P17, P30 to P37, P40 to P44, P50 to P57, P70 to P77, P80 to P85, P125 to P127, P130 | | | | 20.0[Note 2] | mA |
| | | Per pin for P60 to P62 | | | | 15.0[Note 2] | mA |
| | | Total of P00 to P07, P40 to P44, P130 (When duty = 70%[Note 3]) | 4.0 V ≤ $V_{DD}$ ≤ 5.5 V | | | 70.0 | mA |
| | | | 2.7 V ≤ $V_{DD}$ < 4.0 V | | | 15.0 | mA |
| | | | 1.9 V ≤ $V_{DD}$ < 2.7 V | | | 9.0 | mA |
| | | Total of P10 to P17, P30 to P37, P50 to P57, P60 to P62, P70 to P77, P80 to P85, P125 to P127 (When duty = 70%[Note 3]) | 4.0 V ≤ $V_{DD}$ ≤ 5.5 V | | | 80.0 | mA |
| | | | 2.7 V ≤ $V_{DD}$ < 4.0 V | | | 35.0 | mA |
| | | | 1.9 V ≤ $V_{DD}$ < 2.7 V | | | 20.0 | mA |
| | | Total of all pins (When duty = 70%[Note 3]) | | | | 150.0 | mA |
| | $I_{OL2}$ | Per pin for P20 to P25 | 1.9 V ≤ $V_{DD}$[Note 4] ≤ 5.5 V | | | 0.4[Note 2] | mA |
| | | Total of all pins (When duty = 70%[Note 3]) | 1.9 V ≤ $V_{DD}$[Note 4] ≤ 5.5 V | | | 2.4 | mA |

**Notes 1.** Value of current at which the device operation is guaranteed even if the current flows from an output pin to the $EV_{SS}$ and $V_{SS}$ pins.

**2.** However, do not exceed the total current value.

**3.** Specification under conditions where the duty factor ≤ 70%.
The output current value that has changed to the duty factor > 70% the duty ratio can be calculated with the following expression (when changing the duty factor from 70% to n%).

- Total output current of pins = ($I_{OL}$ × 0.7)/(n × 0.01)

<Example> Where n = 80% and $I_{OL}$ = 10.0 mA

Total output current of pins = (10.0 × 0.7)/(80 × 0.01) ≅ 8.7 mA

However, the current that is allowed to flow into one pin does not vary depending on the duty factor. A current higher than the absolute maximum rating must not flow into one pin.

**4.** The power supply voltage (VBAT pin or $V_{DD}$ pin) selected by the battery backup feature.

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Items | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Input voltage, high | $V_{IH1}$ | P00 to P07, P10 to P17, P30 to P37, P40 to P44, P50 to P57, P70 to P77, P80 to P85, P125 to P127 | Normal input buffer | 0.8$EV_{DD}$ | | $EV_{DD}$ | V |
| | $V_{IH2}$ | P00, P03, P05, P06, P15, P16, P81 | TTL input buffer 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V | 2.2 | | $EV_{DD}$ | V |
| | | | TTL input buffer 3.3 V ≤ $EV_{DD}$ < 4.0 V | 2.0 | | $EV_{DD}$ | V |
| | | | TTL input buffer 1.9 V ≤ $EV_{DD}$ < 3.3 V | 1.5 | | $EV_{DD}$ | V |
| | $V_{IH3}$ | P20 to P25 | | 0.7$V_{DD}$ Note | | $V_{DD}$ Note | V |
| | $V_{IH4}$ | P60 to P62 | | 0.7$EV_{DD}$ | | 6.0 | V |
| | $V_{IH5}$ | P121 to P124, P137, EXCLK, EXCLKS | | 0.8$V_{DD}$ Note | | $V_{DD}$ Note | V |
| | $V_{IH6}$ | $\overline{\text{RESET}}$ | | 0.8$V_{DD}$ Note | | 6.0 | V |
| Input voltage, low | $V_{IL1}$ | P00 to P07, P10 to P17, P30 to P37, P40 to P44, P50 to P57, P70 to P77, P80 to P85, P125 to P127 | Normal input buffer | 0 | | 0.2$EV_{DD}$ | V |
| | $V_{IL2}$ | P00, P03, P05, P06, P15, P16, P81 | TTL input buffer 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V | 0 | | 0.8 | V |
| | | | TTL input buffer 3.3 V ≤ $EV_{DD}$ < 4.0 V | 0 | | 0.5 | V |
| | | | TTL input buffer 1.9 V ≤ $EV_{DD}$ < 3.3 V | 0 | | 0.32 | V |
| | $V_{IL3}$ | P20 to P25 | | 0 | | 0.3$V_{DD}$ Note | V |
| | $V_{IL4}$ | P60 to P62 | | 0 | | 0.3$EV_{DD}$ | V |
| | $V_{IL5}$ | P121 to P124, P137, EXCLK, EXCLKS, $\overline{\text{RESET}}$ | | 0 | | 0.2$V_{DD}$ Note | V |

**Note** The power supply voltage (VBAT pin or $V_{DD}$ pin) selected by the battery backup feature.

**Caution The maximum value of $V_{IH}$ of pins P01 to P07, P15 to P17, and P80 to P82 is $V_{DD}$, even in the N-ch open-drain mode.**

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Items | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Output voltage, high | $V_{OH1}$ | P00 to P07, P10 to P17, P30 to P37, P40 to P44, P50 to P57, P70 to P77, P80 to P85, P125 to P127, P130 | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, $I_{OH1}$ = –10.0 mA | $EV_{DD}$ – 1.5 | | | V |
| | | | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, $I_{OH1}$ = –3.0 mA | $EV_{DD}$ – 0.7 | | | V |
| | | | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V, $I_{OH}$ = –2.0 mA | $EV_{DD}$ – 0.6 | | | V |
| | | | 1.9 V ≤ $EV_{DD}$ ≤ 5.5 V, $I_{OH}$ = –1.5 mA | $EV_{DD}$ – 0.5 | | | V |
| | $V_{OH2}$ | P20 to P25 | 1.9 V ≤ $V_{DD}$[Note] ≤ 5.5 V, $I_{OH2}$ = –100 μA | $V_{DD}$ – 0.5 | | | V |
| Output voltage, low | $V_{OL1}$ | P00 to P07, P10 to P17, P30 to P37, P40 to P44, P50 to P57, P70 to P77, P80 to P85, P125 to P127, P130 | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, $I_{OL1}$ = 20 mA | | | 1.3 | V |
| | | | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, $I_{OL1}$ = 8.5 mA | | | 0.7 | V |
| | | | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V, $I_{OL}$ = 3.0 mA | | | 0.6 | V |
| | | | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V, $I_{OL1}$ = 1.5 mA | | | 0.4 | V |
| | | | 1.9 V ≤ $EV_{DD}$ ≤ 5.5 V, $I_{OL1}$ = 0.6 mA | | | 0.4 | V |
| | $V_{OL2}$ | P20 to P25 | 1.9 V ≤ $V_{DD}$[Note] ≤ 5.5 V, $I_{OL2}$ = 400 μA | | | 0.4 | V |
| | $V_{OL3}$ | P60 to P62 | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, $I_{OL3}$ = 15.0 mA | | | 2.0 | V |
| | | | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, $I_{OL3}$ = 5.0 mA | | | 0.4 | V |
| | | | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V, $I_{OL3}$ = 3.0 mA | | | 0.4 | V |
| | | | 1.9 V ≤ $EV_{DD}$ ≤ 5.5 V, $I_{OL3}$ = 2.0 mA | | | 0.4 | V |

**Note** The power supply voltage (VBAT pin or $V_{DD}$ pin) selected by the battery backup feature.

**Caution P01 to P07, P15 to P17, and P80 to P82 do not output high level in N-ch open-drain mode.**

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Items | Symbol | Conditions | | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|---|
| Input leakage current, high | $I_{LIH1}$ | P00 to P07, P10 to P17, P30 to P37, P40 to P44, P60 to P62, P70 to P77, P80 to P85, P125 to P127 | $V_I = EV_{DD}$ | | | | 1 | μA |
| | $I_{LIH2}$ | P20 to P25, P137, $\overline{\text{RESET}}$ | $V_I = V_{DD}$ Note | | | | 1 | μA |
| | $I_{LIH3}$ | P121 to P124 (X1, X2, XT1, XT2, EXCLK, EXCLKS) | $V_I = V_{DD}$ Note | In input port or external clock input | | | 1 | μA |
| | | | | In resonator connection | | | 10 | μA |
| Input leakage current, low | $I_{LIL1}$ | P00 to P07, P10 to P17, P30 to P37, P40 to P44, P60 to P62, P70 to P77, P80 to P85, P125 to P127 | $V_I = EV_{SS}$ | | | | –1 | μA |
| | $I_{LIL2}$ | P20 to P25, P137, $\overline{\text{RESET}}$ | $V_I = V_{SS}$ | | | | –1 | μA |
| | $I_{LIL3}$ | P121 to P124 (X1, X2, XT1, XT2, EXCLK, EXCLKS) | $V_I = V_{SS}$ | In input port or external clock input | | | –1 | μA |
| | | | | In resonator connection | | | –10 | μA |
| <R> On-chip pull-up resistance | $R_{U1}$ | P10 to P17, P30 to P37, P50 to P57, P70 to P77, P80 to P85, P125 to P127 | $V_I = V_{SS}$ | 2.4 V ≤ $EV_{DD}$ ≤ 5.5 V | 10 | 20 | 100 | kΩ |
| <R> | | | | 1.9 V ≤ $EV_{DD}$ ≤ 5.5 V | 10 | 30 | 100 | kΩ |
| | $R_{U2}$ | P00 to P07, P40 to P44 | $V_I = V_{SS}$ | | 10 | 20 | 100 | kΩ |

**Note** The power supply voltage (VBAT pin or $V_{DD}$ pin) selected by the battery backup feature.

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

### 37.3.2 Supply current characteristics

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V) (1/4)**

| Parameter | Symbol | Conditions | | | | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| <R> Supply current[Note 1] <R> | $I_{DD1}$ | Operating mode | HS (high-speed main) mode[Note 5] | $f_{IH}$ = 24 MHz[Note 3] | Basic operation | $V_{DD}$ = 5.0 V | | 1.5 | | mA |
| | | | | | | $V_{DD}$ = 3.0 V | | 1.5 | | mA |
| | | | | | Normal operation | $V_{DD}$ = 5.0 V | | 4.1 | 6.6 | mA |
| | | | | | | $V_{DD}$ = 3.0 V | | 4.1 | 6.6 | mA |
| | | | | $f_{IH}$ = 12 MHz[Note 3] | Normal operation | $V_{DD}$ = 5.0 V | | 2.5 | 3.8 | mA |
| | | | | | | $V_{DD}$ = 3.0 V | | 2.5 | 3.8 | mA |
| | | | | $f_{IH}$ = 6 MHz[Note 3] | Normal operation | $V_{DD}$ = 5.0 V | | 1.6 | 2.5 | mA |
| | | | | | | $V_{DD}$ = 3.0 V | | 1.6 | 2.5 | mA |
| | | | | $f_{IH}$ = 3 MHz[Note 3] | Normal operation | $V_{DD}$ = 5.0 V | | 1.2 | 1.9 | mA |
| | | | | | | $V_{DD}$ = 3.0 V | | 1.2 | 1.9 | mA |
| | | | LS (low-speed main) mode[Note 5] | $f_{IH}$ = 6 MHz[Note 3] | Normal operation | $V_{DD}$ = 3.0 V | | 1.3 | 2.1 | mA |
| | | | | | | $V_{DD}$ = 2.0 V | | 1.3 | 2.1 | mA |
| | | | | $f_{IH}$ = 3 MHz[Note 3] | Normal operation | $V_{DD}$ = 3.0 V | | 0.9 | 1.5 | mA |
| | | | | | | $V_{DD}$ = 2.0 V | | 0.9 | 1.5 | mA |
| | | | HS (high-speed main) mode[Note 5] | $f_{MX}$ = 20 MHz[Note 2], $V_{DD}$ = 5.0 V | Normal operation | Square wave input | | 3.4 | 5.5 | mA |
| | | | | | | Resonator connection | | 3.6 | 5.7 | mA |
| | | | | $f_{MX}$ = 20 MHz[Note 2], $V_{DD}$ = 3.0 V | Normal operation | Square wave input | | 3.4 | 5.5 | mA |
| | | | | | | Resonator connection | | 3.6 | 5.7 | mA |
| | | | | $f_{MX}$ = 16 MHz[Note 2], $V_{DD}$ = 5.0 V | Normal operation | Square wave input | | 2.8 | 4.4 | mA |
| | | | | | | Resonator connection | | 2.9 | 4.6 | mA |
| | | | | $f_{MX}$ = 16 MHz[Note 2], $V_{DD}$ = 3.0 V | Normal operation | Square wave input | | 2.8 | 4.4 | mA |
| | | | | | | Resonator connection | | 2.9 | 4.6 | mA |
| | | | | $f_{MX}$ = 12 MHz[Note 2], $V_{DD}$ = 5.0 V | Normal operation | Square wave input | | 2.3 | 3.6 | mA |
| | | | | | | Resonator connection | | 2.4 | 3.7 | mA |
| | | | | $f_{MX}$ = 12 MHz[Note 2], $V_{DD}$ = 3.0 V | Normal operation | Square wave input | | 2.3 | 3.6 | mA |
| | | | | | | Resonator connection | | 2.4 | 3.7 | mA |
| | | | | $f_{MX}$ = 10 MHz[Note 2], $V_{DD}$ = 5.0 V | Normal operation | Square wave input | | 2.1 | 3.2 | mA |
| | | | | | | Resonator connection | | 2.1 | 3.3 | mA |
| | | | | $f_{MX}$ = 10 MHz[Note 2], $V_{DD}$ = 3.0 V | Normal operation | Square wave input | | 2.1 | 3.2 | mA |
| | | | | | | Resonator connection | | 2.1 | 3.3 | mA |
| | | | LS (low-speed main) mode[Note 5] | $f_{MX}$ = 8 MHz[Note 2], $V_{DD}$ = 3.0 V | Normal operation | Square wave input | | 1.2 | 2.0 | mA |
| | | | | | | Resonator connection | | 1.2 | 2.1 | mA |
| | | | | $f_{MX}$ = 8 MHz[Note 2], $V_{DD}$ = 2.0 V | Normal operation | Square wave input | | 1.2 | 2.0 | mA |
| | | | | | | Resonator connection | | 1.2 | 2.1 | mA |
| | | | Subclock operation | $f_{SUB}$ = 32.768 kHz[Note 4], $T_A$ = –40°C | Normal operation | Square wave input | | 4.8 | 5.9 | µA |
| | | | | | | Resonator connection | | 4.9 | 6.0 | µA |
| | | | | $f_{SUB}$ = 32.768 kHz[Note 4], $T_A$ = +25°C | Normal operation | Square wave input | | 4.9 | 5.9 | µA |
| | | | | | | Resonator connection | | 5.0 | 6.0 | µA |
| | | | | $f_{SUB}$ = 32.768 kHz[Note 4], $T_A$ = +50°C | Normal operation | Square wave input | | 4.9 | 7.6 | µA |
| | | | | | | Resonator connection | | 5.0 | 7.7 | µA |
| | | | | $f_{SUB}$ = 32.768 kHz[Note 4], $T_A$ = +70°C | Normal operation | Square wave input | | 5.2 | 9.3 | µA |
| | | | | | | Resonator connection | | 5.3 | 9.4 | µA |
| | | | | $f_{SUB}$ = 32.768 kHz[Note 4], $T_A$ = +85°C | Normal operation | Square wave input | | 6.1 | 13.3 | µA |
| | | | | | | Resonator connection | | 6.2 | 13.4 | µA |

(**Notes** and **Remarks** are listed on the next page.)

**Notes 1.** Total current flowing into $V_{DD}$ and $EV_{DD}$, including the input leakage current flowing when the level of the input pin is fixed to $V_{DD}$, $EV_{DD}$ or $V_{SS}$, $EV_{SS}$. The values below the MAX. column include the peripheral operation current. However, not including the current flowing into the LCD controller/driver, A/D converter, ΔΣ A/D converter, LVD circuit, comparator, battery backup circuit, I/O port, and on-chip pull-up/pull-down resistors. When the VBAT pin (pin for battery backup) is selected, current flowing into VBAT.

**2.** When high-speed on-chip oscillator and subsystem clock are stopped.

**3.** When high-speed system clock and subsystem clock are stopped.

**4.** When high-speed on-chip oscillator and high-speed system clock are stopped. When setting ultra-low current consumption (AMPHS1 = 1). However, not including the current flowing into real-time clock 2, 12-bit interval timer, and watchdog timer.

**5.** Relationship between operation voltage width, operation frequency of CPU and operation mode is as below.

HS (high-speed main) mode: $2.7\ V \leq V_{DD} \leq 5.5\ V$@1 MHz to 24 MHz

$2.4\ V \leq V_{DD} \leq 5.5\ V$@1 MHz to 16 MHz

LS (low-speed main) mode: $1.9\ V \leq V_{DD} \leq 5.5\ V$@1 MHz to 8 MHz

**Remarks 1.** $f_{MX}$: High-speed system clock frequency (X1 clock oscillation frequency or external main system clock frequency)

**2.** $f_{IH}$: High-speed on-chip oscillator clock frequency

**3.** $f_{SUB}$: Subsystem clock frequency (XT1 clock oscillation frequency)

**4.** Except subsystem clock operation, temperature condition of the TYP. value is $T_A = 25°C$

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V) (2/4)**

| Parameter | Symbol | Conditions | | | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|---|---|
| Supply current[Note 1] | $I_{DD2}$[Note 2] | HALT mode | HS (high-speed main) mode[Note 7] | $f_{IH}$ = 24 MHz[Note 4] | $V_{DD}$ = 5.0 V | | 0.50 | 1.45 | mA |
| | | | | | $V_{DD}$ = 3.0 V | | 0.50 | 1.45 | mA |
| | | | | $f_{IH}$ = 12 MHz[Note 4] | $V_{DD}$ = 5.0 V | | 0.40 | 0.91 | mA |
| | | | | | $V_{DD}$ = 3.0 V | | 0.40 | 0.91 | mA |
| | | | | $f_{IH}$ = 6 MHz[Note 4] | $V_{DD}$ = 5.0 V | | 0.33 | 0.63 | mA |
| | | | | | $V_{DD}$ = 3.0 V | | 0.33 | 0.63 | mA |
| | | | | $f_{IH}$ = 3 MHz[Note 4] | $V_{DD}$ = 5.0 V | | 0.29 | 0.49 | mA |
| | | | | | $V_{DD}$ = 3.0 V | | 0.29 | 0.49 | mA |
| | | | LS (low-speed main) mode[Note 7] | $f_{IH}$ = 6 MHz[Note 4] | $V_{DD}$ = 3.0 V | | 290 | 620 | μA |
| | | | | | $V_{DD}$ = 2.0 V | | 290 | 620 | μA |
| | | | | $f_{IH}$ = 3 MHz[Note 4] | $V_{DD}$ = 3.0 V | | 250 | 534 | μA |
| | | | | | $V_{DD}$ = 2.0 V | | 250 | 534 | μA |
| | | | HS (high-speed main) mode[Note 7] | $f_{MX}$ = 20 MHz[Note 3], $V_{DD}$ = 5.0 V | Square wave input | | 0.31 | 1.08 | mA |
| | | | | | Resonator connection | | 0.48 | 1.28 | mA |
| | | | | $f_{MX}$ = 20 MHz[Note 3], $V_{DD}$ = 3.0 V | Square wave input | | 0.31 | 1.08 | mA |
| | | | | | Resonator connection | | 0.48 | 1.28 | mA |
| | | | | $f_{MX}$ = 16 MHz[Note 3], $V_{DD}$ = 5.0 V | Square wave input | | 0.26 | 0.86 | mA |
| | | | | | Resonator connection | | 0.38 | 1.00 | mA |
| | | | | $f_{MX}$ = 16 MHz[Note 3], $V_{DD}$ = 3.0 V | Square wave input | | 0.26 | 0.86 | mA |
| | | | | | Resonator connection | | 0.38 | 1.00 | mA |
| | | | | $f_{MX}$ = 12 MHz[Note 3], $V_{DD}$ = 5.0 V | Square wave input | | 0.22 | 0.70 | mA |
| | | | | | Resonator connection | | 0.31 | 0.79 | mA |
| | | | | $f_{MX}$ = 12 MHz[Note 3], $V_{DD}$ = 3.0 V | Square wave input | | 0.22 | 0.70 | mA |
| | | | | | Resonator connection | | 0.31 | 0.79 | mA |
| | | | | $f_{MX}$ = 10 MHz[Note 3], $V_{DD}$ = 5.0 V | Square wave input | | 0.21 | 0.63 | mA |
| | | | | | Resonator connection | | 0.28 | 0.71 | mA |
| | | | | $f_{MX}$ = 10 MHz[Note 3], $V_{DD}$ = 3.0 V | Square wave input | | 0.21 | 0.63 | mA |
| | | | | | Resonator connection | | 0.28 | 0.71 | mA |
| | | | LS (low-speed main) mode[Note 7] | $f_{MX}$ = 8 MHz[Note 3], $V_{DD}$ = 3.0 V | Square wave input | | 110 | 360 | μA |
| | | | | | Resonator connection | | 160 | 420 | μA |
| | | | | $f_{MX}$ = 8 MHz[Note 3], $V_{DD}$ = 2.0 V | Square wave input | | 110 | 360 | μA |
| | | | | | Resonator connection | | 160 | 420 | μA |
| | | | Subsystem clock operation | $f_{SUB}$ = 32.768 kHz[Note 5], $T_A$ = –40°C | Square wave input | | 0.36 | 0.77 | μA |
| | | | | | Resonator connection | | 0.55 | 0.98 | μA |
| | | | | $f_{SUB}$ = 32.768 kHz[Note 5], $T_A$ = +25°C | Square wave input | | 0.42 | 0.91 | μA |
| | | | | | Resonator connection | | 0.61 | 1.30 | μA |
| | | | | $f_{SUB}$ = 32.768 kHz[Note 5], $T_A$ = +50°C | Square wave input | | 0.50 | 2.45 | μA |
| | | | | | Resonator connection | | 0.69 | 2.64 | μA |
| | | | | $f_{SUB}$ = 32.768 kHz[Note 5], $T_A$ = +70°C | Square wave input | | 0.86 | 4.28 | μA |
| | | | | | Resonator connection | | 1.05 | 4.47 | μA |
| | | | | $f_{SUB}$ = 32.768 kHz[Note 5], $T_A$ = +85°C | Square wave input | | 2.29 | 8.44 | μA |
| | | | | | Resonator connection | | 2.48 | 8.63 | μA |
| | $I_{DD3}$[Note 6] | STOP mode[Note 8] | $T_A$ = –40°C | | | | 0.27 | 0.70 | μA |
| | | | $T_A$ = +25°C | | | | 0.33 | 0.82 | μA |
| | | | $T_A$ = +50°C | | | | 0.41 | 2.36 | μA |
| | | | $T_A$ = +70°C | | | | 0.77 | 4.19 | μA |
| | | | $T_A$ = +85°C | | | | 2.20 | 8.35 | μA |

(**Notes** and **Remarks** are listed on the next page.)

**Notes 1.** Total current flowing into $V_{DD}$ and $EV_{DD}$, including the input leakage current flowing when the level of the input pin is fixed to $V_{DD}$, $EV_{DD}$ or $V_{SS}$, $EV_{SS}$. The values below the MAX. column include the peripheral operation current. However, not including the current flowing into the LCD controller/driver, A/D converter, ΔΣ A/D converter, LVD circuit, comparator, battery backup circuit, I/O port, and on-chip pull-up/pull-down resistors. When the VBAT pin (pin for battery backup) is selected, current flowing into VBAT.

**2.** During HALT instruction execution by flash memory.

**3.** When high-speed on-chip oscillator and subsystem clock are stopped.

**4.** When high-speed system clock and subsystem clock are stopped.

**5.** When operating real-time clock 2 (RTC2) and setting ultra-low current consumption (AMPHS1 = 1). When high-speed on-chip oscillator and high-speed system clock are stopped. However, not including the current flowing into the 12-bit interval timer and watchdog timer.

**6.** When high-speed on-chip oscillator, high-speed system clock, and subsystem clock are stopped. However, not including the current flowing into real-time clock 2 (RTC2), 12-bit interval timer, and watchdog timer.

**7.** Relationship between operation voltage width, operation frequency of CPU and operation mode is as below.

HS (high-speed main) mode: $2.7\ V \leq V_{DD} \leq 5.5\ V$@1 MHz to 24 MHz

$2.4\ V \leq V_{DD} \leq 5.5\ V$@1 MHz to 16 MHz

LS (low-speed main) mode: $1.9\ V \leq V_{DD} \leq 5.5\ V$@1 MHz to 8 MHz

**8.** If operation of the subsystem clock when STOP mode, same as when HALT mode of subsystem clock operation.

**Remarks 1.** $f_{MX}$: High-speed system clock frequency (X1 clock oscillation frequency or external main system clock frequency)

**2.** $f_{IH}$: High-speed on-chip oscillator clock frequency

**3.** $f_{SUB}$: Subsystem clock frequency (XT1 clock oscillation frequency)

**4.** Except subsystem clock operation and STOP mode, temperature condition of the TYP. value is $T_A = 25°C$

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V) (3/4)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Low-speed on-chip oscillator operating current | $I_{FIL}$ Note 1 | | | | 0.24 | | μA |
| RTC2 operating current | $I_{RTC}$ Notes 1, 2, 3 | $f_{SUB}$ = 32.768 kHz | | | 0.02 | | μA |
| 12-bit interval timer operating current | $I_{TMKA}$ Notes 1, 2, 4 | $f_{SUB}$ = 32.768 kHz, $f_{MAIN}$ is stopped | | | 0.04 | | μA |
| 8-bit interval timer operating current | $I_{TMT}$ Notes 1, 2, 5 | $f_{SUB}$ = 32.768 kHz, $f_{MAIN}$ is stopped, per unit | 8-bit counter mode × 2 ch operation | | 0.12 | | μA |
| | | | 16-bit counter mode operation | | 0.10 | | μA |
| Watchdog timer operating current | $I_{WDT}$ Notes 1, 2, 6 | $f_{IL}$ = 15 kHz, $f_{MAIN}$ is stopped | | | 0.22 | | μA |
| LVD operating current | $I_{LVD}$ Notes 1, 7 | | | | 0.08 | | μA |
| Oscillation stop detection circuit operating current | $I_{OSDC}$ Note 1 | | | | 0.02 | | μA |
| Battery backup circuit operating current | $I_{BUP}$ Note 1 | | | | 0.05 | | μA |
| A/D converter operating current | $I_{ADC}$ Notes 1, 8 | When conversion at maximum speed | Normal mode, $AV_{REFP}$ = $V_{DD}$ = 5.0 V | | 1.3 | 2.4 | mA |
| | | | Low voltage mode, $AV_{REFP}$ = $V_{DD}$ = 3.0 V | | 0.5 | 1.0 | mA |
| A/D converter reference voltage current | $I_{ADREF}$ Note 1 | | | | 75.0 | | μA |
| Temperature sensor operating current | $I_{TMPS}$ Note 1 | | | | 105 | | μA |
| Comparator operating current | $I_{CMP}$ Notes 1, 9 | $V_{DD}$ = 5.0 V, Regulator output voltage = 2.1 V | Window mode | | 12.5 | | μA |
| | | | Comparator high-speed mode | | 6.5 | | μA |
| | | | Comparator low-speed mode | | 1.7 | | μA |
| | | $V_{DD}$ = 5.0 V, Regulator output voltage = 1.8 V | Window mode | | 8.0 | | μA |
| | | | Comparator high-speed mode | | 4.0 | | μA |
| | | | Comparator low-speed mode | | 1.3 | | μA |
| | | $V_{DD}$ = 5.0 V, STOP mode | Window mode | | 8.0 | | μA |
| | | | Comparator high-speed mode | | 4.0 | | μA |
| | | | Comparator low-speed mode | | 1.3 | | μA |
| BGO operating current | $I_{BGO}$ Notes 1, 10 | | | | 2.00 | 12.20 | mA |
| Self-programming operating current | $I_{FSP}$ Notes 1, 11 | | | | 2.00 | 12.20 | mA |

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V) (4/4)**

| Parameter | Symbol | Conditions | | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|---|
| 24-Bit ΔΣ A/D Converter operating current | $I_{DSAD}$ Notes 1, 12 | In 4 ch ΔΣ A/D converter operation | | | | 1.50 | 2.25 | mA |
| | | In 3 ch ΔΣ A/D converter operation | | | | 1.18 | 1.77 | mA |
| | | In 1 ch ΔΣ A/D converter operation | | | | 0.53 | 0.80 | mA |
| SNOOZE operating current | $I_{SNOZ}$ Notes 1, 13 | ADC operation | The mode is performed | | | 0.50 | 0.80 | mA |
| | | | The A/D conversion operations are performed, low voltage mode, $AV_{REFP}$ = $V_{DD}$ = 3.0 V | | | 1.20 | 1.80 | mA |
| | | CSI/UART operation | | | | 0.70 | 1.05 | mA |
| | | DTC operation | | | | 2.20 | | mA |
| LCD operating current | $I_{LCD1}$ Notes 1, 14, 15 | External resistance division method | $f_{LCD}$ = $f_{SUB}$<br>LCD clock = 128 Hz<br>1/3 bias, four-time-slices | $V_{DD}$ = 5.0 V,<br>$V_{L4}$ = 5.0 V | | 0.06 | | μA |
| | $I_{LCD2}$ Notes 1, 14 | Internal voltage boosting method | $f_{LCD}$ = $f_{SUB}$<br>LCD clock = 128 Hz<br>1/3 bias, four-time-slices | $V_{DD}$ = 3.0 V,<br>$V_{L4}$ = 3.0 V<br>(VLCD = 04H) | | 0.85 | | μA |
| | | | | $V_{DD}$ = 5.0 V,<br>$V_{L4}$ = 5.1 V<br>(VLCD = 12H) | | 1.55 | | μA |
| | $I_{LCD3}$ Notes 1, 14 | Capacitor split method | $f_{LCD}$ = $f_{SUB}$<br>LCD clock = 128 Hz<br>1/3 bias, four-time-slices | $V_{DD}$ = 3.0 V,<br>$V_{L4}$ = 3.0 V | | 0.20 | | μA |

**Notes 1.** Current flowing to $V_{DD}$. When the VBAT pin (battery backup power supply pin) is selected, current flowing to the VBAT.

**2.** When high speed on-chip oscillator and high-speed system clock are stopped.

**3.** Current flowing only to real-time clock 2 (excluding the low-speed on-chip oscillator and operating current of the XT1 oscillator). The value of the current value of the RL78 microcontrollers is the sum of the values of either $I_{DD1}$ or $I_{DD2}$, and $I_{RTC}$, when real-time clock 2 operates in operation mode or HALT mode. When the low-speed on-chip oscillator is selected, $I_{FIL}$ should be added. $I_{DD2}$ subsystem clock operation includes the operational current of real-time clock 2.

**4.** Current flowing only to the 12-bit interval timer (excluding the operating current of the low-speed on-chip oscillator and XT1 oscillator). The value of the current value of the RL78 microcontrollers is the sum of the values of either $I_{DD1}$ or $I_{DD2}$, and $I_{TMKA}$, when the 12-bit interval timer operates in operation mode or HALT mode. When the low-speed on-chip oscillator is selected, $I_{FIL}$ should be added.

**5.** Current flowing only to the 8-bit interval timer (excluding the operating current of the low-speed on-chip oscillator and XT1 oscillator). The value of the current value of the RL78 microcontrollers is the sum of the values of either $I_{DD1}$ or $I_{DD2}$, and $I_{TMT}$, when the 8-bit interval timer operates in operation mode or HALT mode. When the low-speed on-chip oscillator is selected, $I_{FIL}$ should be added.

**6.** Current flowing only to the watchdog timer (including the operating current of the low-speed on-chip oscillator). The current value of the RL78 microcontrollers is the sum of $I_{DD1}$, $I_{DD2}$ or $I_{DD3}$ and $I_{WDT}$ when the watchdog timer operates.

**7** Current flowing only to the LVD circuit. The current value of the RL78 microcontrollers is the sum of $I_{DD1}$, $I_{DD2}$ or $I_{DD3}$ and $I_{LVD}$ when the LVD circuit operates.

**8.** Current flowing only to the A/D converter. The current value of the RL78 microcontrollers is the sum of $I_{DD1}$ or $I_{DD2}$ and $I_{ADC}$ when the A/D converter operates in an operation mode or the HALT mode.

**9.** Current flowing only to the comparator circuit. The current value of the RL78 microcontrollers is the sum of $I_{DD1}$, $I_{DD2}$ or $I_{DD3}$ and $I_{CMP}$ when the comparator circuit operates.

**Notes 10.** Current flowing only during rewrite of 1 KB code flash memory.

**11.** Current flowing only during self programming.

**12.** Current flowing only to the 24-bit ΔΣ A/D converter. The current value of the RL78 microcontrollers is the sum of $I_{DD1}$ or $I_{DD2}$, and $I_{DSAD}$ when the 24-bit ΔΣ A/D converter operates.

**13.** For shift time to the SNOOZE mode, see **24.3.3 SNOOZE mode**.

**14.** Current flowing only to the LCD controller/driver. The current value of the RL78 microcontrollers is the sum of the LCD operating current ($I_{LCD1}$, $I_{LCD2}$ or $I_{LCD3}$) to the supply current ($I_{DD1}$, or $I_{DD2}$) when the LCD controller/driver operates in an operation mode or HALT mode. Not including the current that flows through the LCD panel. Conditions of the TYP. value and MAX. value are as follows.

- Setting 20 pins as the segment function and blinking all
- Selecting $f_{SUB}$ for system clock when LCD clock = 128 Hz (LCDC0 = 07H)
- Setting four time slices and 1/3 bias

**15.** Not including the current flowing into the external division resistor when using the external resistance division method.

**Remarks 1.** $f_{IL}$: Low-speed on-chip oscillator clock frequency

**2.** $f_{SUB}$: Subsystem clock frequency (XT1 clock oscillation frequency)

**3.** $f_{CLK}$: CPU/peripheral hardware clock frequency

**4.** Temperature condition of the TYP. value is $T_A$ = 25°C

## 37.4 AC Characteristics

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Items | 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 | 2.7 V ≤ $V_{DD}$ Note 1 ≤ 5.5 V | 0.0417 | | 1 | μs |
| | | | | 2.4 V ≤ $V_{DD}$ Note 1 < 2.7 V | 0.0625 | | 1 | μs |
| | | | LS (low-speed main) mode | 1.9 V ≤ $V_{DD}$ Note 1 ≤ 5.5 V | 0.125 | | 1 | μs |
| | | Subsystem clock ($f_{SUB}$) operation | | 1.9 V ≤ $V_{DD}$ Note 1 ≤ 5.5 V | 28.5 | 30.5 | 31.3 | μs |
| | | In the self programming mode | HS (high-speed main) mode | 2.7 V ≤ $V_{DD}$ Note 1 ≤ 5.5 V | 0.0417 | | 1 | μs |
| | | | | 2.4 V ≤ $V_{DD}$ Note 1 < 2.7 V | 0.0625 | | 1 | μs |
| | | | LS (low-speed main) mode | 1.9 V ≤ $V_{DD}$ Note 1 ≤ 5.5 V | 0.125 | | 1 | μs |
| External system clock frequency | $f_{EX}$ | 2.7 V ≤ $V_{DD}$ Note 1 ≤ 5.5 V | | | 1.0 | | 20.0 | MHz |
| | | 2.4 V ≤ $V_{DD}$ Note 1 < 2.7 V | | | 1.0 | | 16.0 | MHz |
| | | 1.9 V ≤ $V_{DD}$ Note 1 < 2.4 V | | | 1.0 | | 8.0 | MHz |
| | $f_{EXS}$ | | | | 32 | | 35 | kHz |
| External system clock input high-level width, low-level width | $t_{EXH}$, $t_{EXL}$ | 2.7 V ≤ $V_{DD}$ Note 1 ≤ 5.5 V | | | 24 | | | ns |
| | | 2.4 V ≤ $V_{DD}$ Note 1 < 2.7 V | | | 30 | | | ns |
| | | 1.9 V ≤ $V_{DD}$ Note 1 < 2.4 V | | | 60 | | | ns |
| | $t_{EXHS}$, $t_{EXLS}$ | | | | 13.7 | | | μs |
| TI00 to TI07 input high-level width, low-level width | $t_{TIH}$, $t_{TIL}$ | | | | 1/$f_{MCK}$+10 | | | ns Note 2 |
| TO00 to TO07 output frequency | $f_{TO}$ | HS (high-speed main) mode | | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V | | | 12 | MHz |
| | | | | 2.7 V ≤ $EV_{DD}$ < 4.0 V | | | 8 | MHz |
| | | | | 2.4 V ≤ $EV_{DD}$ < 2.7 V | | | 4 | MHz |
| | | LS (low-speed main) mode | | 1.9 V ≤ $EV_{DD}$ ≤ 5.5 V | | | 4 | MHz |
| PCLBUZ0, PCLBUZ1 output frequency | $f_{PCL}$ | HS (high-speed main) mode | | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V | | | 16 | MHz |
| | | | | 2.7 V ≤ $EV_{DD}$ < 4.0 V | | | 8 | MHz |
| | | | | 2.4 V ≤ $EV_{DD}$ < 2.7 V | | | 4 | MHz |
| | | LS (low-speed main) mode | | 1.9 V ≤ $EV_{DD}$ ≤ 5.5 V | | | 4 | MHz |
| Interrupt input high-level width, low-level width | $t_{INTH}$, $t_{INTL}$ | INTP0 | | 1.9 V ≤ $V_{DD}$ Note 1 ≤ 5.5 V | 1 | | | μs |
| | | INTP1 to INTP7 | | 1.9 V ≤ $EV_{DD}$ ≤ 5.5 V | 1 | | | μs |
| $\overline{\text{RESET}}$ low-level width | $t_{RSL}$ | | | | 10 | | | μs |

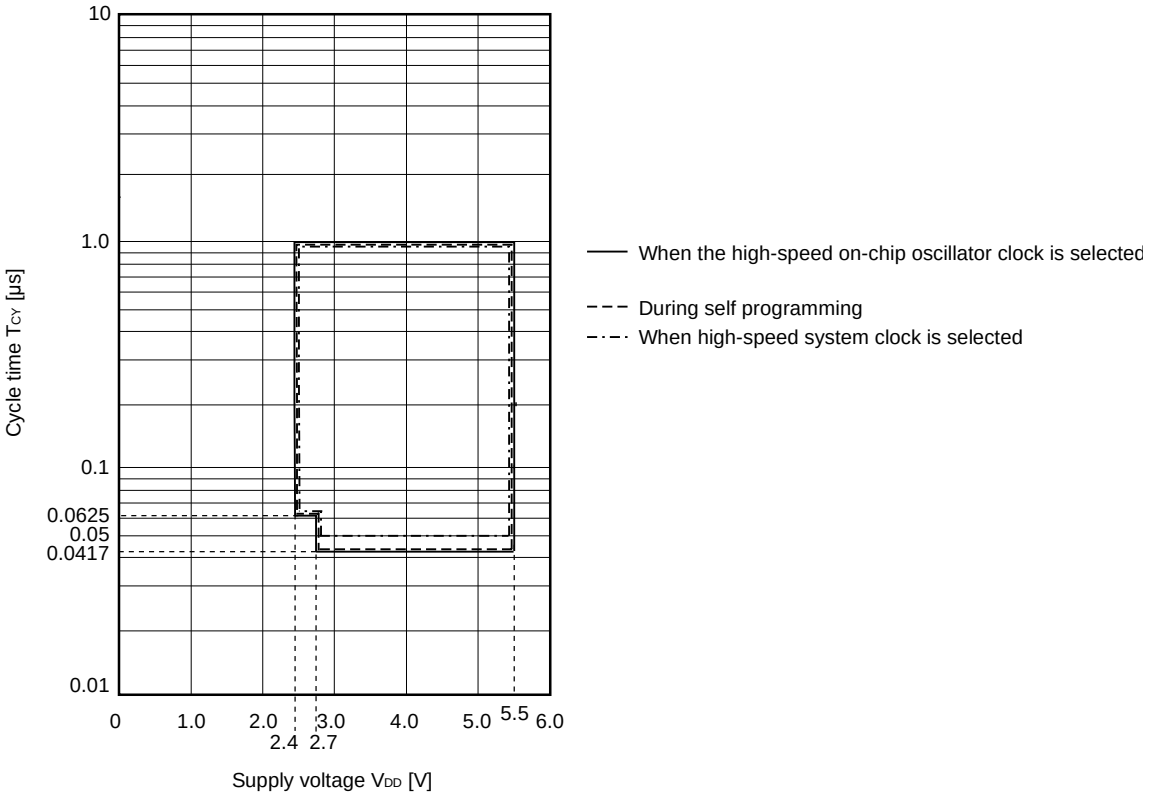
**Notes 1.** The power supply voltage (VBAT pin or $V_{DD}$ pin) selected by the battery backup feature.

**2.** The following conditions are required for low voltage interface:
1.9 V ≤ $V_{DD}$ < 2.7 V: MIN. 125 ns

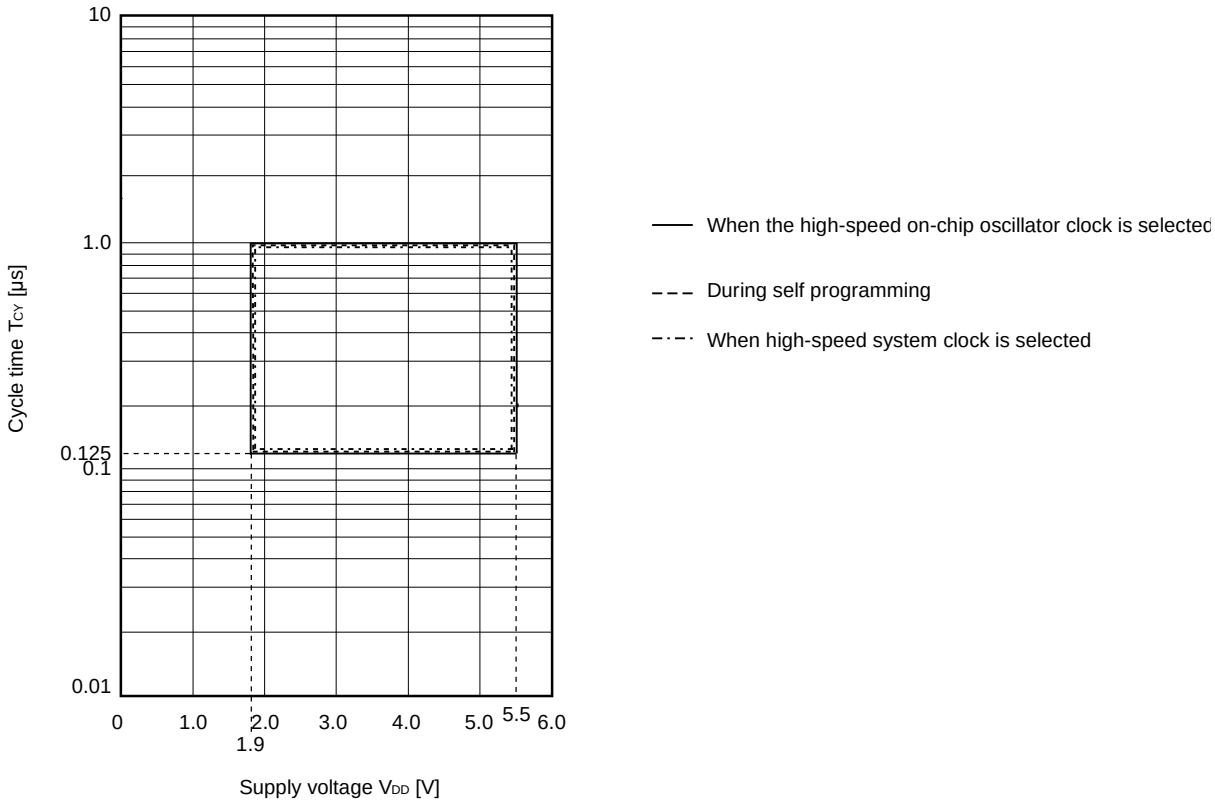
**Remark** $f_{MCK}$: Timer array unit operation clock frequency
(Operation clock to be set by the CKSmn0, CKSmn1 bits of timer mode register mn (TMRmn)
m: Unit number (m = 0), n: Channel number (n = 0 to 7))

**Minimum Instruction Execution Time during Main System Clock Operation**

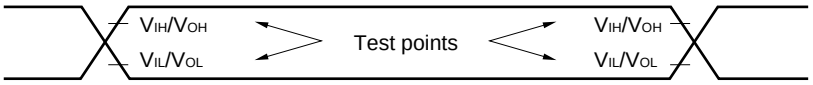
$T_{CY}$ vs $V_{DD}$ (HS (high-speed main) mode)



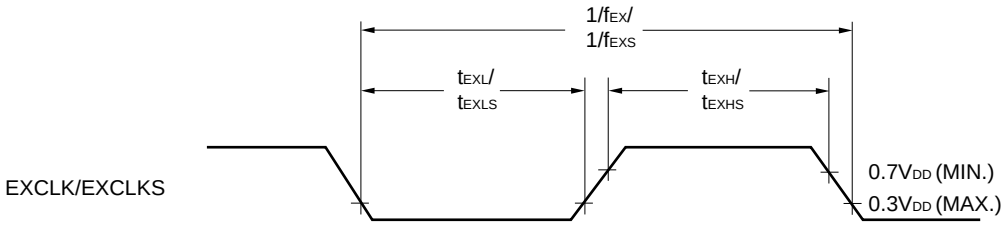

$T_{CY}$ vs $V_{DD}$ (LS (low-speed main) mode)



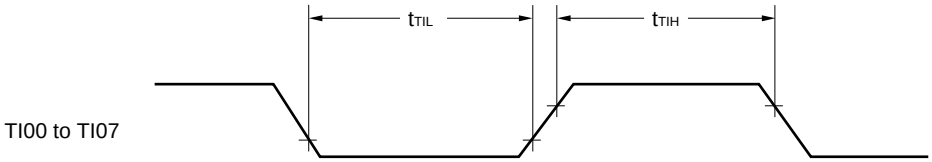
### AC Timing Test Points

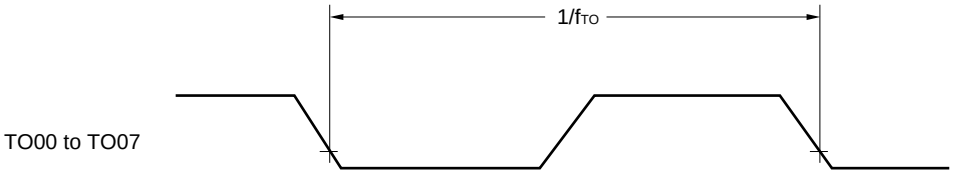



### External System Clock Timing

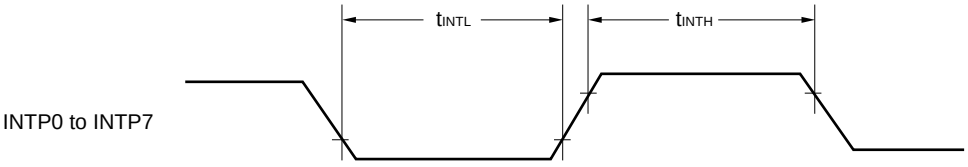



### TI/TO Timing

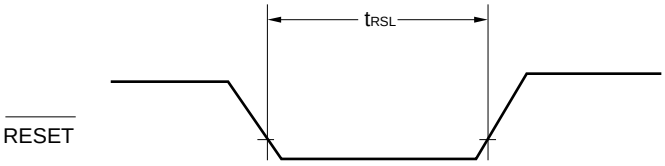



### Interrupt Request Input Timing

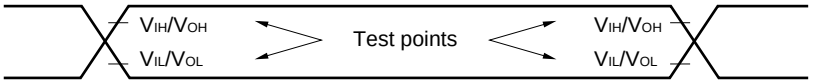



### $\overline{\text{RESET}}$ Input Timing



## 37.5 Peripheral Functions Characteristics

### AC Timing Test Points




### 37.5.1 Serial array unit

**(1) During communication at same potential (UART mode) (dedicated baud rate generator output)**

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | HS (high-speed main) Mode | | LS (low-speed main) Mode | | Unit |
|---|---|---|---|---|---|---|---|
| | | | MIN. | MAX. | MIN. | MAX. | |
| Transfer rate[Note 1] | | 2.4 V≤ $V_{DD}$ ≤ 5.5 V | | $f_{MCK}/6$[Note 2] | | $f_{MCK}/6$[Note 2] | bps |
| | | Theoretical value of the maximum transfer rate $f_{MCK} = f_{CLK}$[Note 3] | | 4.0 | | 1.3 | Mbps |
| | | 1.9 V ≤ $V_{DD}$ ≤ 5.5 V | | | | $f_{MCK}/6$[Note 2] | bps |
| | | Theoretical value of the maximum transfer rate $f_{MCK} = f_{CLK}$[Note 3] | | | | 1.3 | Mbps |

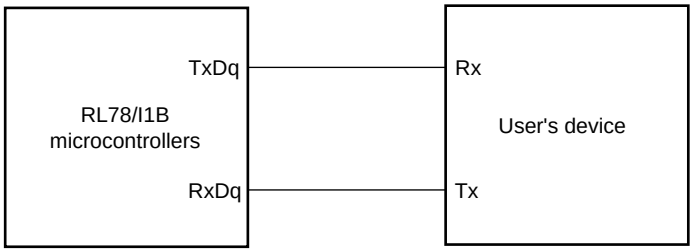
**Notes 1.** Transfer rate in the SNOOZE mode is 4800 bps only.

**2.** The following conditions are required for low voltage interface.
2.4 V ≤ $EV_{DD}$ < 2.7 V: MAX. 2.6 Mbps
1.9 V ≤ $EV_{DD}$ < 2.4 V: MAX. 1.3 Mbps

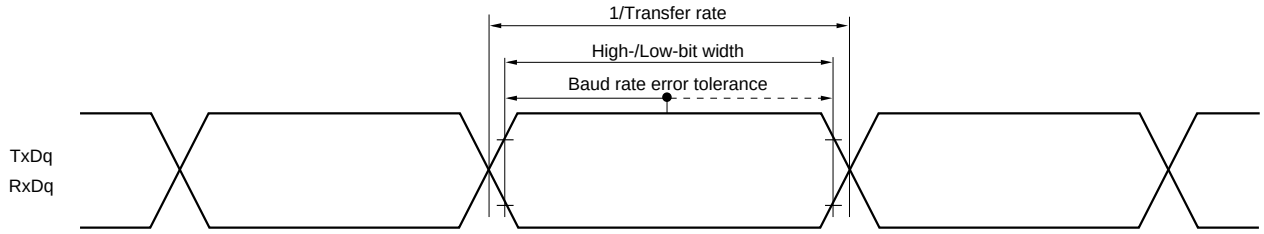
**3.** The maximum operating frequencies of the CPU/peripheral hardware clock ($f_{CLK}$) are:
HS (high-speed main) mode: 24 MHz
LS (low-speed main) mode: 8 MHz

**Caution Select the normal input buffer for the RxDq pin and the normal output mode for the TxDq pin by using port input mode register g (PIMg) and port output mode register g (POMg).**

**UART mode connection diagram (during communication at same potential)**



**UART mode bit width (during communication at same potential) (reference)**




**Remarks 1.** q: UART number (q = 0 to 2), g: PIM and POM number (g = 0, 1, 8)

**2.** $f_{MCK}$: Serial array unit operation clock frequency
(Operating clock that is set with the serial clock select register m (SPSm) and the CKSmn bit of serial mode register mn (SMRmn).
m: Unit number, n: Channel number (mn = 00 to 03, 10, 11))

**(2) During communication at same potential (CSI mode) (master mode, SCKp... internal clock output)**

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | | HS (high-speed main) Mode | | LS (low-speed main) Mode | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | | | MIN. | MAX. | MIN. | MAX. | |
| SCKp cycle time | $t_{KCY1}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 167 | | 500 | | ns |
| | | 2.4 V ≤ $EV_{DD}$ ≤ 5.5 V | | 250 | | 500 | | ns |
| | | 1.9 V ≤ $EV_{DD}$ ≤ 5.5 V | | | | 500 | | ns |
| SCKp high-/low-level width | $t_{KH1}$, $t_{KL1}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V | | $t_{KCY1}$/2 – 12 | | $t_{KCY1}$/2 – 50 | | ns |
| | | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | $t_{KCY1}$/2 – 18 | | $t_{KCY1}$/2 – 50 | | ns |
| | | 2.4 V ≤ $EV_{DD}$ ≤ 5.5 V | | $t_{KCY1}$/2 – 38 | | $t_{KCY1}$/2 – 50 | | ns |
| | | 1.9 V ≤ $EV_{DD}$ ≤ 5.5 V | | | | $t_{KCY1}$/2 – 50 | | ns |
| SIp setup time (to SCKp↑)[Note 1] | $t_{SIK1}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V | | 44 | | 110 | | ns |
| | | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 44 | | 110 | | ns |
| | | 2.4 V ≤ $EV_{DD}$ ≤ 5.5 V | | 75 | | 110 | | ns |
| | | 1.9 V ≤ $EV_{DD}$ ≤ 5.5 V | | | | 110 | | ns |
| SIp hold time (from SCKp↑)[Note 2] | $t_{KSI1}$ | 2.4 V ≤ $EV_{DD}$ ≤ 5.5 V | | 19 | | 19 | | ns |
| | | 1.9 V ≤ $EV_{DD}$ ≤ 5.5 V | | | | 19 | | ns |
| Delay time from SCKp↓ to SOp output[Note 3] | $t_{KSO1}$ | C = 30 pF[Note 4] | 2.4 V ≤ $EV_{DD}$ ≤ 5.5 V | | 25 | | 25 | ns |
| | | | 1.9 V ≤ $EV_{DD}$ ≤ 5.5 V | | | | 25 | ns |

**Notes 1.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The SIp setup time becomes "to SCKp↓" when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

**2.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The SIp hold time becomes "from SCKp↓" when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

**3.** 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.

**4.** 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 port input mode register g (PIMg) and port output mode register g (POMg).**

**Remarks 1.** p: CSI number (p = 00), m: Unit number (m = 0), n: Channel number (n = 0),
g: PIM and POM numbers (g = 0, 1)

**2.** $f_{MCK}$: Serial array unit operation clock frequency
(Operating clock that is set with the serial clock select register m (SPSm) and the CKSmn bit of serial mode register mn (SMRmn).
m: Unit number, n: Channel number (mn = 00))

**(3) During communication at same potential (CSI mode) (slave mode, SCKp... external clock input)**

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | | HS (high-speed main) Mode | | LS (low-speed main) Mode | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | | | MIN. | MAX. | MIN. | MAX. | |
| SCKp cycle time[Note 5] | $t_{KCY2}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V | 20 MHz < $f_{MCK}$ | 8/$f_{MCK}$ | | — | | ns |
| | | | $f_{MCK}$ ≤ 20 MHz | 6/$f_{MCK}$ | | 6/$f_{MCK}$ | | ns |
| | | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | 16 MHz < $f_{MCK}$ | 8/$f_{MCK}$ | | — | | ns |
| | | | $f_{MCK}$ ≤ 16 MHz | 6/$f_{MCK}$ | | 6/$f_{MCK}$ | | ns |
| | | 2.4 V ≤ $EV_{DD}$ ≤ 5.5 V | | 6/$f_{MCK}$ and 500 | | 6/$f_{MCK}$ | | ns |
| | | 1.9 V ≤ $EV_{DD}$ ≤ 5.5 V | | | | 6/$f_{MCK}$ | | ns |
| SCKp high-/low-level width | $t_{KH2}$, $t_{KL2}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V | | $t_{KCY2}$/2 – 7 | | $t_{KCY2}$/2 – 7 | | ns |
| | | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | $t_{KCY2}$/2 – 8 | | $t_{KCY2}$/2 – 8 | | ns |
| | | 2.4 V ≤ $EV_{DD}$ ≤ 5.5 V | | $t_{KCY2}$/2 – 18 | | $t_{KCY2}$/2 – 18 | | ns |
| | | 1.9 V ≤ $EV_{DD}$ ≤ 5.5 V | | | | $t_{KCY2}$/2 – 18 | | ns |
| SIp setup time (to SCKp↑)[Note 1] | $t_{SIK2}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 1/$f_{MCK}$+20 | | 1/$f_{MCK}$+30 | | ns |
| | | 2.4 V ≤ $EV_{DD}$ ≤ 5.5 V | | 1/$f_{MCK}$+30 | | 1/$f_{MCK}$+30 | | ns |
| | | 1.9 V ≤ $EV_{DD}$ ≤ 5.5 V | | | | 1/$f_{MCK}$+30 | | ns |
| SIp hold time (from SCKp↑)[Note 2] | $t_{KSI2}$ | 2.4 V ≤ $EV_{DD}$ ≤ 5.5 V | | 1/$f_{MCK}$+31 | | 1/$f_{MCK}$+31 | | ns |
| | | 1.9 V ≤ $EV_{DD}$ ≤ 5.5 V | | | | 1/$f_{MCK}$+31 | | ns |
| Delay time from SCKp↓ to SOp output[Note 3] | $t_{KSO2}$ | C = 30 pF[Note 4] | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 2/$f_{MCK}$+44 | | 2/$f_{MCK}$+110 | ns |
| | | | 2.4 V ≤ $EV_{DD}$ ≤ 5.5 V | | 2/$f_{MCK}$+75 | | 2/$f_{MCK}$+110 | ns |
| | | | 1.9 V ≤ $EV_{DD}$ ≤ 5.5 V | | | | 2/$f_{MCK}$+110 | ns |

**Notes 1.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The SIp setup time becomes "to SCKp↓" when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

**2.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The SIp hold time becomes "from SCKp↓" when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

**3.** 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.

**4.** C is the load capacitance of the SOp output lines.

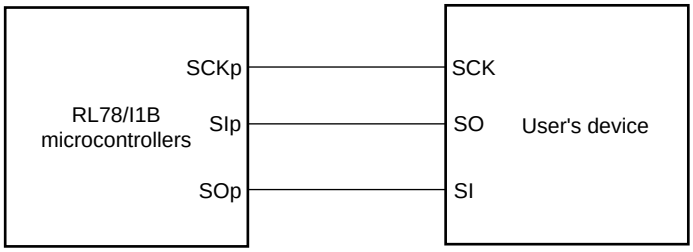
**5.** Transfer rate in the SNOOZE mode: MAX. 1 Mbps

**Caution Select the normal input buffer for the SIp pin and SCKp pin and the normal output mode for the SOp pin by using port input mode register g (PIMg) and port output mode register g (POMg).**

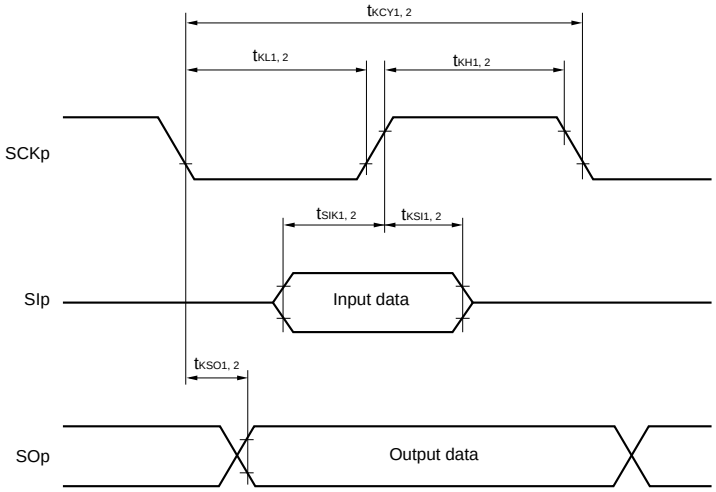
**Remarks 1.** p: CSI number (p = 00), m: Unit number (m = 0), n: Channel number (n = 0), g: PIM number (g = 0, 1)

**2.** $f_{MCK}$: Serial array unit operation clock frequency
(Operating clock that is set with the serial clock select register m (SPSm) and the CKSmn bit of serial mode register mn (SMRmn).
m: Unit number, n: Channel number (mn = 00))

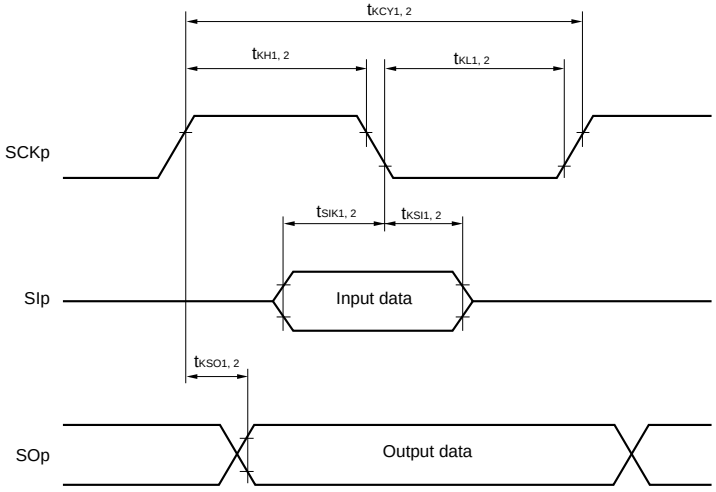
**CSI mode connection diagram (during communication at same potential)**




**CSI mode serial transfer timing (during communication at same potential)**
**(when DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1)**




**CSI mode serial transfer timing (during communication at same potential)**
**(when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0)**




**Remarks 1.** p: CSI number (p = 00)

**2.** m: Unit number, n: Channel number (mn = 00)

**(4) During communication at same potential (simplified $I^2C$ mode)**

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | HS (high-speed main) Mode | | LS (low-speed main) Mode | | Unit |
|---|---|---|---|---|---|---|---|
| | | | MIN. | MAX. | MIN. | MAX. | |
| SCLr clock frequency | $f_{SCL}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | | 1000 Note 1 | | 400 Note 1 | kHz |
| | | 1.9 V ≤ $EV_{DD}$ ≤ 5.5 V, $C_b$ = 100 pF, $R_b$ = 3 kΩ | | 400 Note 1 | | 400 Note 1 | kHz |
| | | 1.9 V Note 3 ≤ $EV_{DD}$ < 2.7 V, $C_b$ = 100 pF, $R_b$ = 5 kΩ | | 300 Note 1 | | 300 Note 1 | kHz |
| Hold time when SCLr = “L” | $t_{LOW}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 475 | | 1150 | | ns |
| | | 1.9 V ≤ $EV_{DD}$ ≤ 5.5 V, $C_b$ = 100 pF, $R_b$ = 3 kΩ | 1150 | | 1150 | | ns |
| | | 1.9 V Note 3 ≤ $EV_{DD}$ < 2.7 V, $C_b$ = 100 pF, $R_b$ = 5 kΩ | 1550 | | 1550 | | ns |
| Hold time when SCLr = “H” | $t_{HIGH}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 475 | | 1150 | | ns |
| | | 1.9 V ≤ $EV_{DD}$ ≤ 5.5 V, $C_b$ = 100 pF, $R_b$ = 3 kΩ | 1150 | | 1150 | | ns |
| | | 1.9 V Note 3 ≤ $EV_{DD}$ < 2.7 V, $C_b$ = 100 pF, $R_b$ = 5 kΩ | 1550 | | 1550 | | ns |
| Data setup time (reception) | $t_{SU:DAT}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | $1/f_{MCK}$ + 85 Notes 1, 2 | | $1/f_{MCK}$ + 145 Notes 1, 2 | | ns |
| | | 1.9 V ≤ $EV_{DD}$ ≤ 5.5 V, $C_b$ = 100 pF, $R_b$ = 3 kΩ | $1/f_{MCK}$ + 145 Notes 1, 2 | | $1/f_{MCK}$ + 145 Notes 1, 2 | | ns |
| | | 1.9 V Note 3 ≤ $EV_{DD}$ < 2.7 V, $C_b$ = 100 pF, $R_b$ = 5 kΩ | $1/f_{MCK}$ + 230 Notes 1, 2 | | $1/f_{MCK}$ + 230 Notes 1, 2 | | ns |
| Data hold time (transmission) | $t_{HD:DAT}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 0 | 305 | 0 | 305 | ns |
| | | 1.9 V ≤ $EV_{DD}$ ≤ 5.5 V, $C_b$ = 100 pF, $R_b$ = 3 kΩ | 0 | 355 | 0 | 355 | ns |
| | | 1.9 V Note 3 ≤ $EV_{DD}$ < 2.7 V, $C_b$ = 100 pF, $R_b$ = 5 kΩ | 0 | 405 | 0 | 405 | ns |

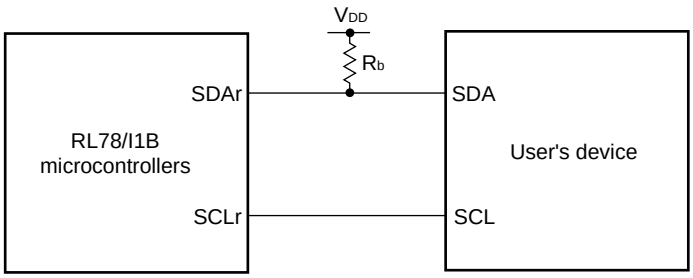
**Notes 1.** The value must also be equal to or less than $f_{MCK}$/4.

**2.** Set the $f_{MCK}$ value to keep the hold time of SCLr = “L” and SCLr = “H”.

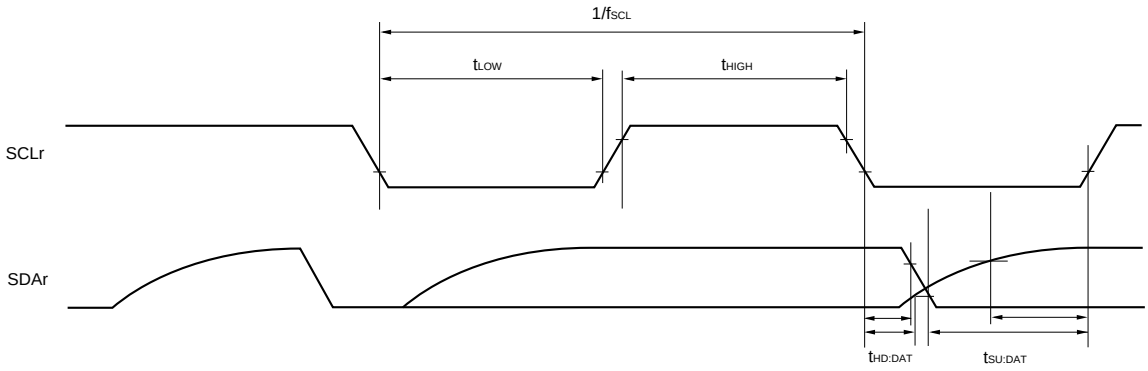
**3.** When HS (high-speed main) mode, this value becomes 2.4 V.

(**Caution** and **Remarks** are listed on the next page.)

**Simplified I²C mode connection diagram (during communication at same potential)**




**Simplified I²C mode serial transfer timing (during communication at same potential)**




**Caution Select the normal input buffer and the N-ch open drain output ($V_{DD}$ tolerance) mode for the SDAr pin and the normal output mode for the SCLr pin by using port input mode register g (PIMg) and port output mode register g (POMg).**

**Remarks 1.** $R_b$[Ω]:Communication line (SDAr) pull-up resistance, $C_b$[F]: Communication line (SDAr, SCLr) load capacitance

**2.** r: IIC number (r = 00, 10), g: PIM and POM number (g = 0, 1)

**3.** $f_{MCK}$: Serial array unit operation clock frequency
(Operating clock that is set with the serial clock select register m (SPSm) and the CKSmn bit of serial mode register mn (SMRmn).
m: Unit number (m = 0), n: Channel number (n = 0, 2), mn = 00, 02))

**(5) Communication at different potential (1.9 V, 2.5 V, 3 V) (UART mode) (1/2)**

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | | HS (high-speed main) Mode | | LS (low-speed main) Mode | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | | | MIN. | MAX. | MIN. | MAX. | |
| Transfer rate | | Reception | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V | | $f_{MCK}/6$ Note 1 | | $f_{MCK}/6$ Note 1 | bps |
| | | | Theoretical value of the maximum transfer rate $f_{MCK} = f_{CLK}$ Note 4 | | 4.0 | | 1.3 | Mbps |
| | | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V | | $f_{MCK}/6$ Note 1 | | $f_{MCK}/6$ Note 1 | bps |
| | | | Theoretical value of the maximum transfer rate $f_{MCK} = f_{CLK}$ Note 4 | | 4.0 | | 1.3 | Mbps |
| | | | 1.9 V Note 5 ≤ $EV_{DD}$ < 3.3 V, 1.8 V ≤ $V_b$ ≤ 2.0 V | | $f_{MCK}/6$ Notes 1 to 3 | | $f_{MCK}/6$ Notes 1, 2 | bps |
| | | | Theoretical value of the maximum transfer rate $f_{MCK} = f_{CLK}$ Note 4 | | 4.0 | | 1.3 | Mbps |

**Notes 1.** Transfer rate in the SNOOZE mode is 4800 bps only.

**2.** Use it with $EV_{DD} \geq V_b$.

**3.** The following conditions are required for low voltage interface.
2.4 V ≤ $EV_{DD}$ < 2.7 V: MAX. 2.6 Mbps
1.9 V ≤ $EV_{DD}$ < 2.4 V: MAX. 1.3 Mbps

**4.** The maximum operating frequencies of the CPU/peripheral hardware clock ($f_{CLK}$) are:
HS (high-speed main) mode: 24 MHz
LS (low-speed main) mode: 8 MHz

**5.** When HS (high-speed main) mode, this value becomes 2.4 V.

**Caution Select the TTL input buffer for the RxDq pin and the N-ch open drain output ($V_{DD}$ tolerance) mode for the TxDq pin by using port input mode register g (PIMg) and port output mode register g (POMg). For $V_{IH}$ and $V_{IL}$, see the DC characteristics with TTL input buffer selected.**

**Remarks 1.** $V_b$[V]: Communication line voltage

**2.** q: UART number (q = 0 to 2), g: PIM and POM number (g = 0, 1, 8)

**3.** $f_{MCK}$: Serial array unit operation clock frequency
(Operating clock that is set with the serial clock select register m (SPSm) and the CKSmn bit of serial mode register mn (SMRmn).
m: Unit number, n: Channel number (mn = 00 to 03, 10, 11))

**(5) Communication at different potential (1.8 V, 2.5 V, 3 V) (UART mode) (2/2)**

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | | HS (high-speed main) Mode | | LS (low-speed main) Mode | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | | | MIN. | MAX. | MIN. | MAX. | |
| Transfer rate | | Transmission | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V | | **Notes 1, 2** | | **Notes 1, 2** | 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 3 | | 2.8 Note 3 | Mbps |
| | | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V | | **Notes 2, 4** | | **Notes 2, 4** | 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 5 | | 1.2 Note 5 | Mbps |
| | | | 1.9 V Note 9 ≤ $EV_{DD}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V | | **Notes 2, 6, 7** | | **Notes 2, 6, 7** | 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 8 | | 0.43 Note 8 | Mbps |

**Notes 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 ≤ $EV_{DD}$ ≤ 5.5 V and 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.

**2.** Transfer rate in the SNOOZE mode is 4800 bps only.

**3.** This value as an example is calculated when the conditions described in the “Conditions” column are met. Refer to **Note 1** above to calculate the maximum transfer rate under conditions of the customer.

**4.** 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 ≤ $EV_{DD}$ < 4.0 V and 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.

**5.** This value as an example is calculated when the conditions described in the “Conditions” column are met. Refer to **Note 4** above to calculate the maximum transfer rate under conditions of the customer.

**6.** Use it with $EV_{DD} \geq V_b$.

**Notes 7.** 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 $1.9\ V \leq EV_{DD} < 2.7\ V$ and $1.6\ V \leq V_b \leq 2.0\ V$

$$\text{Maximum transfer rate} = \frac{1}{\{-C_b \times R_b \times \ln(1 - \frac{1.5}{V_b})\} \times 3} \text{ [bps]}$$

$$\text{Baud rate error (theoretical value)} = \frac{\frac{1}{\text{Transfer rate} \times 2} - \{-C_b \times R_b \times \ln(1 - \frac{1.5}{V_b})\}}{(\frac{1}{\text{Transfer rate}}) \times \text{Number of transferred bits}} \times 100 \text{ [\%]}$$

* This value is the theoretical value of the relative difference between the transmission and reception sides.

**8.** This value as an example is calculated when the conditions described in the “Conditions” column are met. Refer to **Note 7** above to calculate the maximum transfer rate under conditions of the customer.

**9.** When HS (high-speed main) mode, this value becomes 2.4 V.

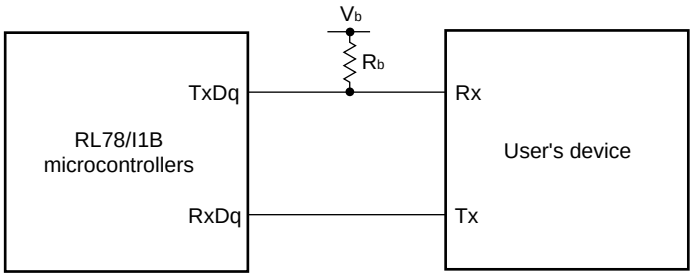
**Caution Select the TTL input buffer for the RxDq pin and the N-ch open drain output ($V_{DD}$ tolerance) mode for the TxDq pin by using port input mode register g (PIMg) and port output mode register g (POMg). For $V_{IH}$ and $V_{IL}$, see the DC characteristics with TTL input buffer selected.**

**Remarks 1.** $R_b$[Ω]:Communication line (TxDq) pull-up resistance,
$C_b$[F]: Communication line (TxDq) load capacitance, $V_b$[V]: Communication line voltage

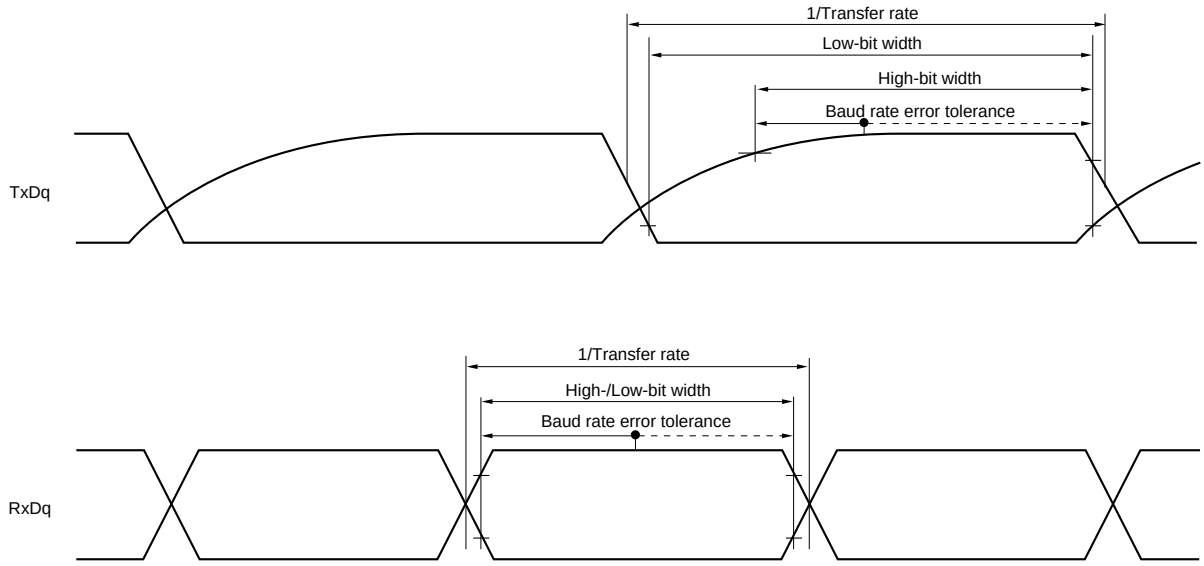
**2.** q: UART number (q = 0 to 2), g: PIM and POM number (g = 0, 1, 8)

**3.** $f_{MCK}$: Serial array unit operation clock frequency
(Operating clock that is set with the serial clock select register m (SPSm) and the CKSmn bit of serial mode register mn (SMRmn).
m: Unit number, n: Channel number (mn = 00 to 03, 10, 11))

**UART mode connection diagram (during communication at different potential)**



**UART mode bit width (during communication at different potential) (reference)**




**Caution Select the TTL input buffer for the RxDq pin and the N-ch open drain output ($V_{DD}$ tolerance) mode for the TxDq pin by using port input mode register g (PIMg) and port output mode register g (POMg). For $V_{IH}$ and $V_{IL}$, see the DC characteristics with TTL input buffer selected.**

**Remarks 1.** $R_b$[Ω]:Communication line (TxDq) pull-up resistance, $V_b$[V]: Communication line voltage

**2.** q: UART number (q = 0 to 2), g: PIM and POM number (g = 0, 1, 8)

**(6) Communication at different potential (2.5 V, 3 V) ($f_{MCK}$/2) (CSI mode) (master mode, SCKp... internal clock output, corresponding CSI00 only)**

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | | HS (high-speed main) Mode | | LS (low-speed main) Mode | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | | | MIN. | MAX. | MIN. | MAX. | |
| SCKp cycle time | $t_{KCY1}$ | $t_{KCY1} \geq 2/f_{CLK}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V, $C_b$ = 20 pF, $R_b$ = 1.4 kΩ | 200 | | 1150 | | ns |
| | | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 20 pF, $R_b$ = 2.7 kΩ | 300 | | 1150 | | ns |
| SCKp high-level width | $t_{KH1}$ | 4.0 V ≤ $EV_{DD}$ ≤ 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 | | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 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 | | ns |
| SCKp low-level width | $t_{KL1}$ | 4.0 V ≤ $EV_{DD}$ ≤ 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 – 50 | | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 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 – 50 | | ns |
| SIp setup time (to SCKp↑)[Note 1] | $t_{SIK1}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V, $C_b$ = 20 pF, $R_b$ = 1.4 kΩ | | 58 | | 479 | | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 20 pF, $R_b$ = 2.7 kΩ | | 121 | | 479 | | ns |
| SIp hold time (from SCKp↑)[Note 1] | $t_{KSI1}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V, $C_b$ = 20 pF, $R_b$ = 1.4 kΩ | | 10 | | 10 | | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 20 pF, $R_b$ = 2.7 kΩ | | 10 | | 10 | | ns |
| Delay time from SCKp↓ to SOp output[Note 1] | $t_{KSO1}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V, $C_b$ = 20 pF, $R_b$ = 1.4 kΩ | | | 60 | | 60 | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 20 pF, $R_b$ = 2.7 kΩ | | | 130 | | 130 | ns |
| SIp setup time (to SCKp↓)[Note 2] | $t_{SIK1}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V, $C_b$ = 20 pF, $R_b$ = 1.4 kΩ | | 23 | | 110 | | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 20 pF, $R_b$ = 2.7 kΩ | | 33 | | 110 | | ns |
| SIp hold time (from SCKp↓)[Note 2] | $t_{KSI1}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V, $C_b$ = 20 pF, $R_b$ = 1.4 kΩ | | 10 | | 10 | | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 20 pF, $R_b$ = 2.7 kΩ | | 10 | | 10 | | ns |
| Delay time from SCKp↑ to SOp output[Note 2] | $t_{KSO1}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V, $C_b$ = 20 pF, $R_b$ = 1.4 kΩ | | | 10 | | 10 | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 20 pF, $R_b$ = 2.7 kΩ | | | 10 | | 10 | ns |

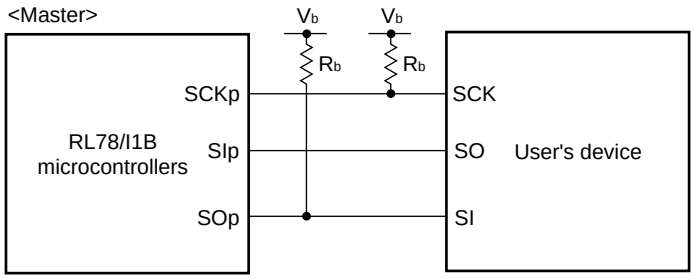
(**Caution** and **Remarks** are listed on the next page.)

**Notes 1.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1.

**2.** 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 ($V_{DD}$ tolerance) mode for the SOp pin and SCKp pin by using port input mode register g (PIMg) and port output mode register g (POMg). For $V_{IH}$ and $V_{IL}$, see the DC characteristics with TTL input buffer selected.**

**CSI mode connection diagram (during communication at different potential)**




**Remarks 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

**2.** p: CSI number (p = 00), m: Unit number (m = 0), n: Channel number (n = 0), g: PIM and POM number (g = 1)

**3.** $f_{MCK}$: Serial array unit operation clock frequency
(Operating clock that is set with the serial clock select register m (SPSm) and the CKSmn bit of serial mode register mn (SMRmn).
m: Unit number, n: Channel number (mn = 00))

**4.** This specification is valid only when CSI00's peripheral I/O redirect function is not used.

**(7) Communication at different potential (1.8 V, 2.5 V, 3 V) ($f_{MCK}$/4) (CSI mode) (master mode, SCKp... internal clock output) (1/2)**

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | | HS (high-speed main) Mode | | LS (low-speed main) Mode | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | | | MIN. | MAX. | MIN. | MAX. | |
| SCKp cycle time | $t_{KCY1}$ | $t_{KCY1} \geq 4/f_{CLK}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V, $C_b$ = 30 pF, $R_b$ = 1.4 kΩ | 300 | | 1150 | | ns |
| | | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | 500 | | 1150 | | ns |
| | | | 1.9 V[Note 4] ≤ $EV_{DD}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V, $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | 1150 | | 1150 | | ns |
| SCKp high-level width | $t_{KH1}$ | 4.0 V ≤ $EV_{DD}$ ≤ 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 | | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 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 | | ns |
| | | 1.9 V[Note 4] ≤ $EV_{DD}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 3], $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | | $t_{KCY1}$/2 – 458 | | $t_{KCY1}$/2 – 458 | | ns |
| SCKp low-level width | $t_{KL1}$ | 4.0 V ≤ $EV_{DD}$ ≤ 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 – 50 | | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 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 – 50 | | ns |
| | | 1.9 V[Note 4] ≤ $EV_{DD}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 3], $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | | $t_{KCY1}$/2 – 50 | | $t_{KCY1}$/2 – 50 | | ns |

(**Notes**, **Caution** and **Remarks** are listed on the page after the next page.)

**(7) Communication at different potential (1.8 V, 2.5 V, 3 V) ($f_{MCK}$/4) (CSI mode) (master mode, SCKp... internal clock output) (2/2)**

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | HS (high-speed main) Mode | | LS (low-speed main) Mode | | Unit |
|---|---|---|---|---|---|---|---|
| | | | MIN. | MAX. | MIN. | MAX. | |
| SIp setup time (to SCKp↑)[Note 1] | $t_{SIK1}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V, $C_b$ = 30 pF, $R_b$ = 1.4 kΩ | 81 | | 479 | | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | 177 | | 479 | | ns |
| | | 1.9 V[Note 4] ≤ $EV_{DD}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 3], $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | 479 | | 479 | | ns |
| SIp hold time (from SCKp↑)[Note 1] | $t_{KSI1}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V, $C_b$ = 30 pF, $R_b$ = 1.4 kΩ | 19 | | 19 | | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | 19 | | 19 | | ns |
| | | 1.9 V[Note 4] ≤ $EV_{DD}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 3], $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | 19 | | 19 | | ns |
| Delay time from SCKp↓ to SOp output[Note 1] | $t_{KSO1}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V, $C_b$ = 30 pF, $R_b$ = 1.4 kΩ | | 100 | | 100 | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | | 195 | | 195 | ns |
| | | 1.9 V[Note 4] ≤ $EV_{DD}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 3], $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | | 483 | | 483 | ns |
| SIp setup time (to SCKp↓)[Note 2] | $t_{SIK1}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V, $C_b$ = 30 pF, $R_b$ = 1.4 kΩ | 44 | | 110 | | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | 44 | | 110 | | ns |
| | | 1.9 V[Note 4] ≤ $EV_{DD}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 3], $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | 110 | | 110 | | ns |
| SIp hold time (from SCKp↓)[Note 2] | $t_{KSI1}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V, $C_b$ = 30 pF, $R_b$ = 1.4 kΩ | 19 | | 19 | | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | 19 | | 19 | | ns |
| | | 1.9 V[Note 4] ≤ $EV_{DD}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 3], $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | 19 | | 19 | | ns |
| Delay time from SCKp↑ to SOp output[Note 2] | $t_{KSO1}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V, $C_b$ = 30 pF, $R_b$ = 1.4 kΩ | | 25 | | 25 | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | | 25 | | 25 | ns |
| | | 1.9 V[Note 4] ≤ $EV_{DD}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 3], $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | | 25 | | 25 | ns |

(**Notes**, **Caution** and **Remarks** are listed on the next page.)

**Notes 1.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1.

**2.** When DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

**3.** Use it with $EV_{DD} \geq V_b$.

**4.** When HS (high-speed main) mode, this value becomes 2.4 V.

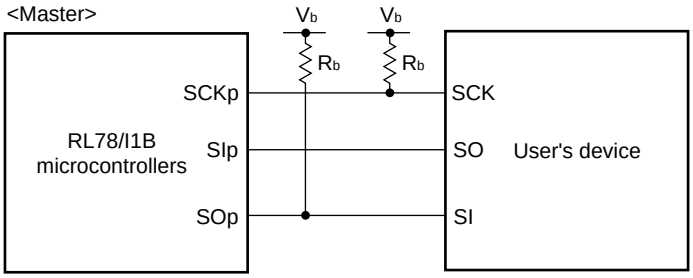
**Caution Select the TTL input buffer for the SIp pin and the N-ch open drain output ($V_{DD}$ tolerance) mode for the SOp pin and SCKp pin by using port input mode register g (PIMg) and port output mode register g (POMg). For $V_{IH}$ and $V_{IL}$, see the DC characteristics with TTL input buffer selected.**

**Remarks 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

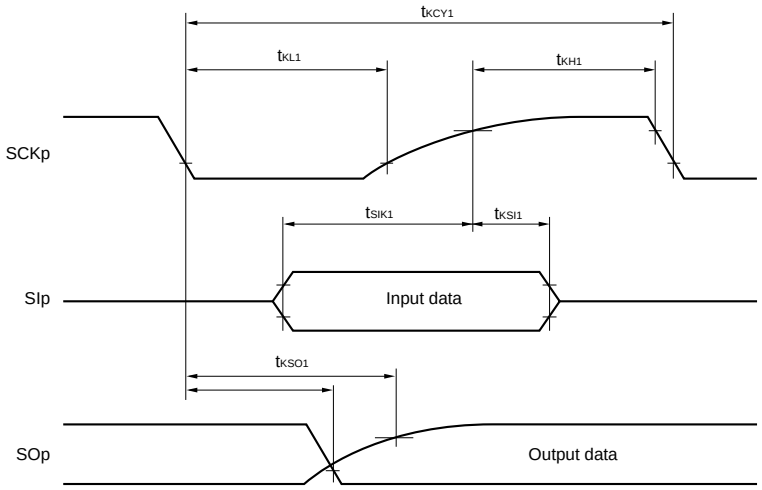
**2.** p: CSI number (p = 00), m: Unit number , n: Channel number (mn = 00),
g: PIM and POM number (g = 0, 1)

**3.** $f_{MCK}$: Serial array unit operation clock frequency
(Operation clock to be set by the CKSmn bit of serial mode register mn (SMRmn).
m: Unit number, n: Channel number (mn = 00))

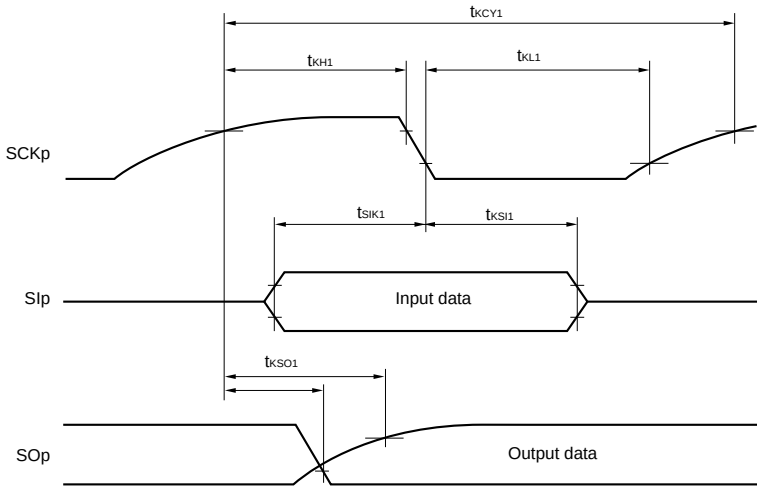
**CSI mode connection diagram (during communication at different potential)**



**CSI mode serial transfer timing (master mode) (during communication at different potential)**
**(when DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1)**




**CSI mode serial transfer timing (master mode) (during communication at different potential)**
**(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 ($V_{DD}$ tolerance) mode for the SOp pin and SCKp pin by using port input mode register g (PIMg) and port output mode register g (POMg). For $V_{IH}$ and $V_{IL}$, see the DC characteristics with TTL input buffer selected.**

**Remark** p: CSI number (p = 00), m: Unit number, n: Channel number (mn = 00),
g: PIM and POM number (g = 0, 1)

**(8) Communication at different potential (1.8 V, 2.5 V, 3 V) (CSI mode) (slave mode, SCKp ... external clock input)**
**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | | HS (high-speed main) Mode | | LS (low-speed main) Mode | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | | | MIN. | MAX. | MIN. | MAX. | |
| SCKp cycle time[Note 1] | $t_{KCY2}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V | 20 MHz < $f_{MCK}$ ≤ 24 MHz | 12/$f_{MCK}$ | | — | | ns |
| | | | 8 MHz < $f_{MCK}$ ≤ 20 MHz | 10/$f_{MCK}$ | | — | | ns |
| | | | 4 MHz < $f_{MCK}$ ≤ 8 MHz | 8/$f_{MCK}$ | | 16/$f_{MCK}$ | | ns |
| | | | $f_{MCK}$ ≤ 4 MHz | 6/$f_{MCK}$ | | 10/$f_{MCK}$ | | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V | 20 MHz < $f_{MCK}$ ≤ 24 MHz | 16/$f_{MCK}$ | | — | | ns |
| | | | 16 MHz < $f_{MCK}$ ≤ 20 MHz | 14/$f_{MCK}$ | | — | | ns |
| | | | 8 MHz < $f_{MCK}$ ≤ 16 MHz | 12/$f_{MCK}$ | | — | | ns |
| | | | 4 MHz < $f_{MCK}$ ≤ 8 MHz | 8/$f_{MCK}$ | | 16/$f_{MCK}$ | | ns |
| | | | $f_{MCK}$ ≤ 4 MHz | 6/$f_{MCK}$ | | 10/$f_{MCK}$ | | ns |
| | | 1.9 V[Note 6] ≤ $EV_{DD}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 2] | 20 MHz < $f_{MCK}$ ≤ 24 MHz | 36/$f_{MCK}$ | | — | | ns |
| | | | 16 MHz < $f_{MCK}$ ≤ 20 MHz | 32/$f_{MCK}$ | | — | | ns |
| | | | 8 MHz < $f_{MCK}$ ≤ 16 MHz | 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/$f_{MCK}$ | | ns |
| SCKp high-/low-level width | $t_{KH2}$, $t_{KL2}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V | | $t_{KCY2}$/2 – 12 | | $t_{KCY2}$/2 – 50 | | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V | | $t_{KCY2}$/2 – 18 | | $t_{KCY2}$/2 – 50 | | ns |
| | | 1.9 V[Note 6] ≤ $EV_{DD}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 2] | | $t_{KCY2}$/2 – 50 | | $t_{KCY2}$/2 – 50 | | ns |
| SIp setup time (to SCKp↑)[Note 3] | $t_{SIK2}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.3 V ≤ $V_b$ ≤ 4.0 V[Note 2] | | 1/$f_{MCK}$ + 20 | | 1/$f_{MCK}$ + 30 | | ns |
| | | 1.9 V[Note 6] ≤ $EV_{DD}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 2] | | 1/$f_{MCK}$ + 30 | | 1/$f_{MCK}$ + 30 | | ns |
| SIp hold time (from SCKp↑)[Note 4] | $t_{KSI2}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.3 V ≤ $V_b$ ≤ 4.0 V[Note 2] | | 1/$f_{MCK}$ + 31 | | 1/$f_{MCK}$ + 31 | | ns |
| | | 1.9 V[Note 6] ≤ $EV_{DD}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 2] | | 1/$f_{MCK}$ + 31 | | 1/$f_{MCK}$ + 31 | | ns |
| Delay time from SCKp↓ to SOp output[Note 5] | $t_{KSO2}$ | 4.0 V ≤ $EV_{DD}$ ≤ 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/$f_{MCK}$ + 573 | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 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/$f_{MCK}$ + 573 | ns |
| | | 1.9 V[Note 6] ≤ $EV_{DD}$ < 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 | ns |

(**Notes**, **Caution** and **Remarks** are listed on the next page.)

**Notes 1.** Transfer rate in the SNOOZE mode: MAX. 1 Mbps

**2.** Use it with $EV_{DD} \geq V_b$.

**3.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The SIp setup time becomes "to SCKp↓" when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

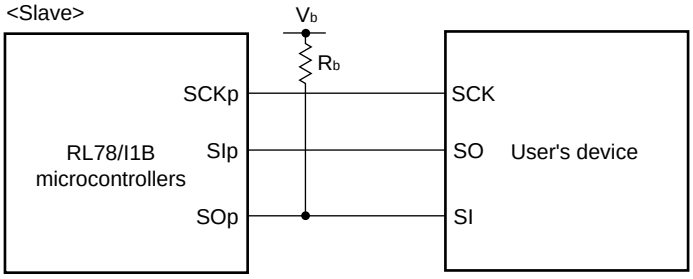
**4.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The SIp hold time becomes "from SCKp↓" when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

**5.** 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.

**6.** When HS (high-speed main) mode, this value becomes 2.4 V.

**Caution Select the TTL input buffer for the SIp pin and SCKp pin and the N-ch open drain output ($V_{DD}$ tolerance) mode for the SOp pin by using port input mode register g (PIMg) and port output mode register g (POMg). For $V_{IH}$ and $V_{IL}$, see the DC characteristics with TTL input buffer selected.**

**CSI mode connection diagram (during communication at different potential)**

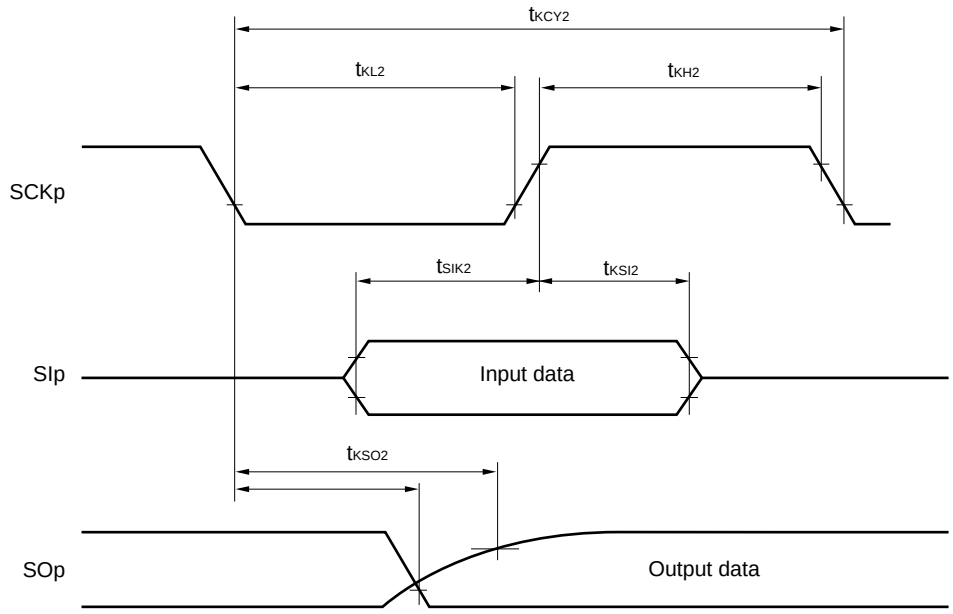



**Remarks 1.** $R_b$[Ω]:Communication line (SOp) pull-up resistance, $C_b$[F]: Communication line (SOp) load capacitance, $V_b$[V]: Communication line voltage

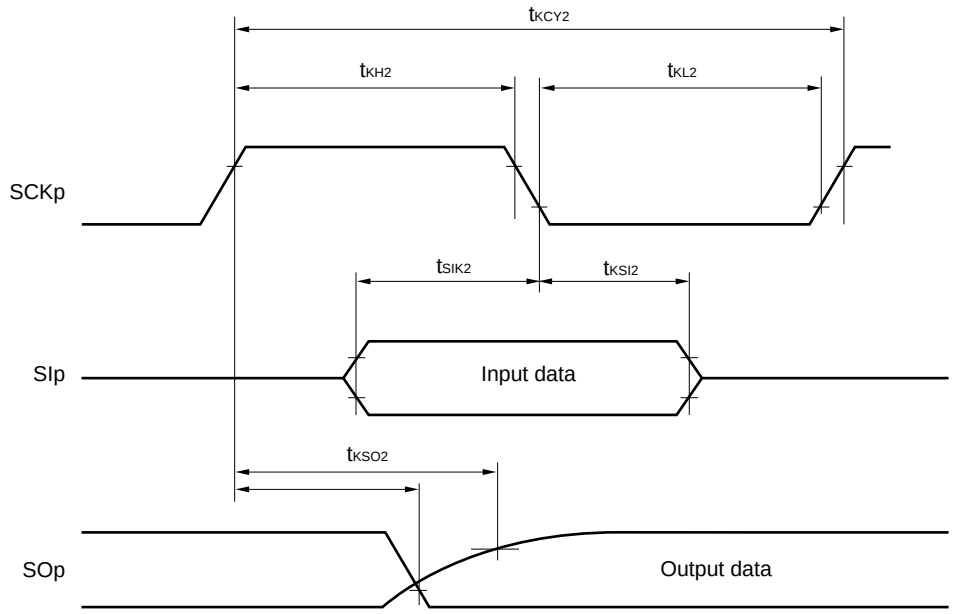
**2.** p: CSI number (p = 00), m: Unit number, n: Channel number (mn = 00), g: PIM and POM number (g = 0, 1)

**3.** $f_{MCK}$: Serial array unit operation clock frequency
(Operating clock that is set with the serial clock select register m (SPSm) and the CKSmn bit of serial mode register mn (SMRmn).
m: Unit number, n: Channel number (mn = 00))

**CSI mode serial transfer timing (slave mode) (during communication at different potential)**
**(when DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1)**




**CSI mode serial transfer timing (slave mode) (during communication at different potential)**
**(When DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.)**




**Caution Select the TTL input buffer for the SIp pin and SCKp pin and the N-ch open drain output ($V_{DD}$ tolerance) mode for the SOp pin by using port input mode register g (PIMg) and port output mode register g (POMg). For $V_{IH}$ and $V_{IL}$, see the DC characteristics with TTL input buffer selected.**

**Remark** p: CSI number (p = 00), m: Unit number, n: Channel number (mn = 00),
g: PIM and POM number (g = 0, 1)

**(9) Communication at different potential (1.8 V, 2.5 V, 3 V) (simplified $I^2C$ mode) (1/2)**

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | HS (high-speed main) Mode | | LS (low-speed main) Mode | | Unit |
|---|---|---|---|---|---|---|---|
| | | | MIN. | MAX. | MIN. | MAX. | |
| SCLr clock frequency | $f_{SCL}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | | 1000 Note 1 | | 300 Note 1 | kHz |
| | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | | 1000 Note 1 | | 300 Note 1 | kHz |
| | | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V, $C_b$ = 100 pF, $R_b$ = 2.8 kΩ | | 400 Note 1 | | 300 Note 1 | kHz |
| | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 100 pF, $R_b$ = 2.7 kΩ | | 400 Note 1 | | 300 Note 1 | kHz |
| | | 1.9 V Note 4 ≤ $EV_{DD}$ < 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 | kHz |
| Hold time when SCLr = “L” | $t_{LOW}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 475 | | 1550 | | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 475 | | 1550 | | ns |
| | | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V, $C_b$ = 100 pF, $R_b$ = 2.8 kΩ | 1150 | | 1550 | | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 100 pF, $R_b$ = 2.7 kΩ | 1150 | | 1550 | | ns |
| | | 1.9 V Note 4 ≤ $EV_{DD}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V Note 2, $C_b$ = 100 pF, $R_b$ = 5.5 kΩ | 1550 | | 1550 | | ns |
| Hold time when SCLr = “H” | $t_{HIGH}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 245 | | 610 | | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 200 | | 610 | | ns |
| | | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V, $C_b$ = 100 pF, $R_b$ = 2.8 kΩ | 675 | | 610 | | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 100 pF, $R_b$ = 2.7 kΩ | 600 | | 610 | | ns |
| | | 1.9 V Note 4 ≤ $EV_{DD}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V Note 2, $C_b$ = 100 pF, $R_b$ = 5.5 kΩ | 610 | | 610 | | ns |

(**Notes**, **Caution** and **Remarks** are listed on the next page.)

**(9) Communication at different potential (1.8 V, 2.5 V, 3 V) (simplified $I^2C$ mode) (2/2)**

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | HS (high-speed main) Mode | | LS (low-speed main) Mode | | Unit |
|---|---|---|---|---|---|---|---|
| | | | MIN. | MAX. | MIN. | MAX. | |
| Data setup time (reception) | $t_{SU:DAT}$ | 4.0 V ≤ $EV_{DD}$ ≤ 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 | | $1/f_{MCK}$ + 190 Note 3 | | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 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 | | $1/f_{MCK}$ + 190 Note 3 | | ns |
| | | 4.0 V ≤ $EV_{DD}$ ≤ 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 | | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 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 | | ns |
| | | 1.9 V Note 4 ≤ $EV_{DD}$ < 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 | | ns |
| Data hold time (transmission) | $t_{HD:DAT}$ | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 0 | 305 | | 305 | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 0 | 305 | | 305 | ns |
| | | 4.0 V ≤ $EV_{DD}$ ≤ 5.5 V, 2.7 V ≤ $V_b$ ≤ 4.0 V, $C_b$ = 100 pF, $R_b$ = 2.8 kΩ | 0 | 355 | | 355 | ns |
| | | 2.7 V ≤ $EV_{DD}$ < 4.0 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 100 pF, $R_b$ = 2.7 kΩ | 0 | 355 | | 355 | ns |
| | | 1.9 V Note 4 ≤ $EV_{DD}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V Note 2, $C_b$ = 100 pF, $R_b$ = 5.5 kΩ | 0 | 405 | | 405 | ns |

**Notes 1.** The value must also be equal to or less than $f_{MCK}$/4.

**2.** Use it with $EV_{DD} \geq V_b$.

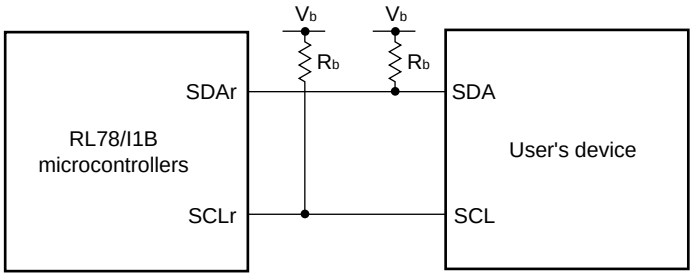
**3.** Set the $f_{MCK}$ value to keep the hold time of SCLr = “L” and SCLr = “H”.

**4.** When HS (high-speed main) mode, this value becomes 2.4 V.

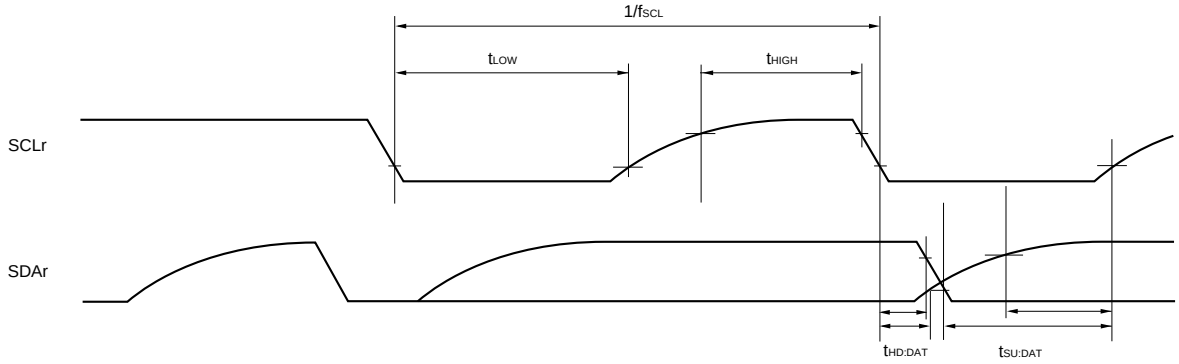
**Caution Select the TTL input buffer and the N-ch open drain output ($V_{DD}$ tolerance) mode for the SDAr pin and the N-ch open drain output ($V_{DD}$ tolerance) mode for the SCLr pin by using port input mode register g (PIMg) and port output mode register g (POMg). For $V_{IH}$ and $V_{IL}$, see the DC characteristics with TTL input buffer selected.**

(**Remarks** is listed on the next page.)

**Simplified I²C mode connection diagram (during communication at different potential)**




**Simplified I²C mode serial transfer timing (during communication at different potential)**




**Caution Select the TTL input buffer and the N-ch open drain output ($V_{DD}$ tolerance) mode for the SDAr pin and the N-ch open drain output ($V_{DD}$ tolerance) mode for the SCLr pin by using port input mode register g (PIMg) and port output mode register g (POMg). For $V_{IH}$ and $V_{IL}$, see the DC characteristics with TTL input buffer selected.**

**Remarks 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

**2.** r: IIC number (r = 00, 10), g: PIM, POM number (g = 0, 1)

**3.** $f_{MCK}$: Serial array unit operation clock frequency
(Operating clock that is set with the serial clock select register m (SPSm) and the CKSmn bit of serial mode register mn (SMRmn).
m: Unit number, n: Channel number (mn = 00, 02))

### 37.5.2 Serial interface IICA

#### (1) $I^2C$ standard mode

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | | HS (high-speed main) Mode | | LS (low-speed main) Mode | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | | | MIN. | MAX. | MIN. | MAX. | |
| SCLA0 clock frequency | $f_{SCL}$ | Standard mode: $f_{CLK}$ ≥ 1 MHz | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | 0 | 100 | 0 | 100 | kHz |
| | | | 1.9 V[Note 3] ≤ $EV_{DD}$ ≤ 5.5 V | 0 | 100 | 0 | 100 | kHz |
| Setup time of restart condition | $t_{SU:STA}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 4.7 | | 4.7 | | μs |
| | | 1.9 V[Note 3] ≤ $EV_{DD}$ ≤ 5.5 V | | 4.7 | | 4.7 | | μs |
| Hold time[Note 1] | $t_{HD:STA}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 4.0 | | 4.0 | | μs |
| | | 1.9 V[Note 3] ≤ $EV_{DD}$ ≤ 5.5 V | | 4.0 | | 4.0 | | μs |
| Hold time when SCLA0 = “L” | $t_{LOW}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 4.7 | | 4.7 | | μs |
| | | 1.9 V[Note 3] ≤ $EV_{DD}$ ≤ 5.5 V | | 4.7 | | 4.7 | | μs |
| Hold time when SCLA0 = “H” | $t_{HIGH}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 4.0 | | 4.0 | | μs |
| | | 1.9 V[Note 3] ≤ $EV_{DD}$ ≤ 5.5 V | | 4.0 | | 4.0 | | μs |
| Data setup time (reception) | $t_{SU:DAT}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 250 | | 250 | | ns |
| | | 1.9 V[Note 3] ≤ $EV_{DD}$ ≤ 5.5 V | | 250 | | 250 | | ns |
| Data hold time (transmission)[Note 2] | $t_{HD:DAT}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 0 | 3.45 | 0 | 3.45 | μs |
| | | 1.9 V[Note 3] ≤ $EV_{DD}$ ≤ 5.5 V | | 0 | 3.45 | 0 | 3.45 | μs |
| Setup time of stop condition | $t_{SU:STO}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 4.0 | | 4.0 | | μs |
| | | 1.9 V[Note 3] ≤ $EV_{DD}$ ≤ 5.5 V | | 4.0 | | 4.0 | | μs |
| Bus-free time | $t_{BUF}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 4.7 | | 4.7 | | μs |
| | | 1.9 V[Note 3] ≤ $EV_{DD}$ ≤ 5.5 V | | 4.7 | | 4.7 | | μs |

**Notes 1.** The first clock pulse is generated after this period when the start/restart condition is detected.

**2.** The maximum value (MAX.) of $t_{HD:DAT}$ is during normal transfer and a wait state is inserted in the ACK (acknowledge) timing.

**3.** When HS (high-speed main) mode, this value becomes 2.4 V.

**Remark** The maximum value of $C_b$ (communication line capacitance) and the value of $R_b$ (communication line pull-up resistor) at that time in each mode are as follows.

Standard mode: $C_b$ = 400 pF, $R_b$ = 2.7 kΩ

### (2) $I^2C$ fast mode

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | | HS (high-speed main) Mode | | LS (low-speed main) Mode | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | | | MIN. | MAX. | MIN. | MAX. | |
| SCLA0 clock frequency | $f_{SCL}$ | Fast mode: $f_{CLK}$ ≥ 3.5 MHz | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | 0 | 400 | 0 | 400 | kHz |
| | | | 1.9 V[Note 3] ≤ $EV_{DD}$ ≤ 5.5 V | 0 | 400 | 0 | 400 | kHz |
| Setup time of restart condition | $t_{SU:STA}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 0.6 | | 0.6 | | μs |
| | | 1.9 V[Note 3] ≤ $EV_{DD}$ ≤ 5.5 V | | 0.6 | | 0.6 | | μs |
| Hold time[Note 1] | $t_{HD:STA}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 0.6 | | 0.6 | | μs |
| | | 1.9 V[Note 3] ≤ $EV_{DD}$ ≤ 5.5 V | | 0.6 | | 0.6 | | μs |
| Hold time when SCLA0 = “L” | $t_{LOW}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 1.3 | | 1.3 | | μs |
| | | 1.9 V[Note 3] ≤ $EV_{DD}$ ≤ 5.5 V | | 1.3 | | 1.3 | | μs |
| Hold time when SCLA0 = “H” | $t_{HIGH}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 0.6 | | 0.6 | | μs |
| | | 1.9 V[Note 3] ≤ $EV_{DD}$ ≤ 5.5 V | | 0.6 | | 0.6 | | μs |
| Data setup time (reception) | $t_{SU:DAT}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 100 | | 100 | | ns |
| | | 1.9 V[Note 3] ≤ $EV_{DD}$ ≤ 5.5 V | | 100 | | 100 | | ns |
| Data hold time (transmission)[Note 2] | $t_{HD:DAT}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 0 | 0.9 | 0 | 0.9 | μs |
| | | 1.9 V[Note 3] ≤ $EV_{DD}$ ≤ 5.5 V | | 0 | 0.9 | 0 | 0.9 | μs |
| Setup time of stop condition | $t_{SU:STO}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 0.6 | | 0.6 | | μs |
| | | 1.9 V[Note 3] ≤ $EV_{DD}$ ≤ 5.5 V | | 0.6 | | 0.6 | | μs |
| Bus-free time | $t_{BUF}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 1.3 | | 1.3 | | μs |
| | | 1.9 V[Note 3] ≤ $EV_{DD}$ ≤ 5.5 V | | 1.3 | | 1.3 | | μs |

**Notes 1.** The first clock pulse is generated after this period when the start/restart condition is detected.

**2.** The maximum value (MAX.) of $t_{HD:DAT}$ is during normal transfer and a wait state is inserted in the ACK (acknowledge) timing.

**3.** When HS (high-speed main) mode, this value becomes 2.4 V.

**Remark** The maximum value of $C_b$ (communication line capacitance) and the value of $R_b$ (communication line pull-up resistor) at that time in each mode are as follows.

Fast mode: $C_b$ = 320 pF, $R_b$ = 1.1 kΩ

### (3) $I^2C$ fast mode plus

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | | HS (high-speed main) Mode | | LS (low-speed main) Mode | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | | | MIN. | MAX. | MIN. | MAX. | |
| SCLA0 clock frequency | $f_{SCL}$ | Fast mode plus: $f_{CLK}$ ≥ 10 MHz | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | 0 | 1000 | – | – | kHz |
| Setup time of restart condition | $t_{SU:STA}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 0.26 | | – | – | μs |
| Hold time[Note 1] | $t_{HD:STA}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 0.26 | | – | – | μs |
| Hold time when SCLA0 = “L” | $t_{LOW}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 0.5 | | – | – | μs |
| Hold time when SCLA0 = “H” | $t_{HIGH}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 0.26 | | – | – | μs |
| Data setup time (reception) | $t_{SU:DAT}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 50 | | – | – | ns |
| Data hold time (transmission)[Note 2] | $t_{HD:DAT}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 0 | 0.45 | – | – | μs |
| Setup time of stop condition | $t_{SU:STO}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 0.26 | | – | – | μs |
| Bus-free time | $t_{BUF}$ | 2.7 V ≤ $EV_{DD}$ ≤ 5.5 V | | 0.5 | | – | – | μs |

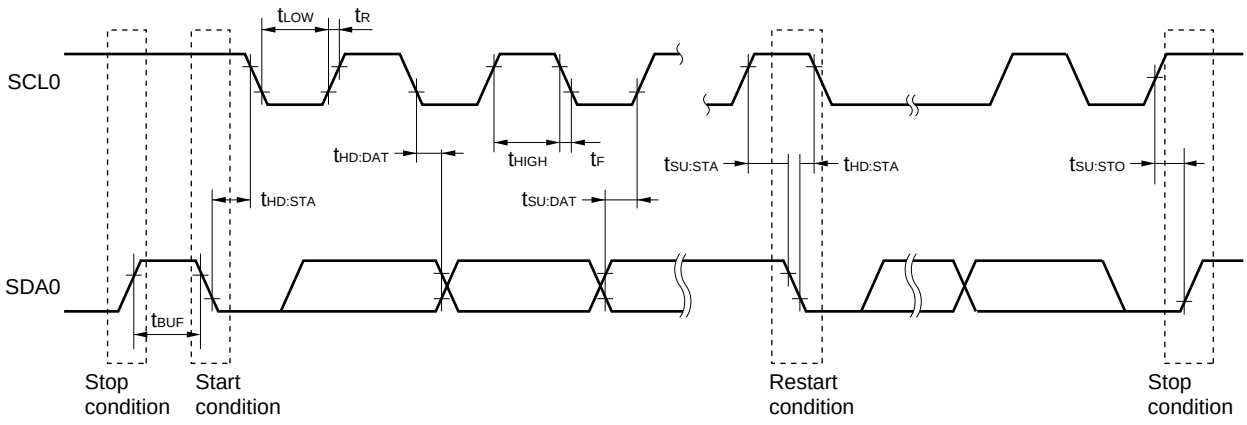
**Notes 1.** The first clock pulse is generated after this period when the start/restart condition is detected.

**2.** The maximum value (MAX.) of $t_{HD:DAT}$ is during normal transfer and a wait state is inserted in the ACK (acknowledge) timing.

**Remark** The maximum value of $C_b$ (communication line capacitance) and the value of $R_b$ (communication line pull-up resistor) at that time in each mode are as follows.

Fast mode plus: $C_b$ = 120 pF, $R_b$ = 1.1 kΩ

**IICA serial transfer timing**



## 37.6 Analog Characteristics

### 37.6.1 A/D converter characteristics

**(1) When reference voltage (+) = $AV_{REFP}$/ANI0 (ADREFP1 = 0, ADREFP0 = 1), reference voltage (–) = $AV_{REFM}$/ANI1 (ADREFM = 1), target pins: ANI2 to ANI5 and internal reference voltage**

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V, reference voltage (+) = $AV_{REFP}$, reference voltage (–) = $AV_{REFM}$ = 0 V)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Resolution | RES | | | 8 | | 10 | bit |
| Overall error[Notes 1, 2] | AINL | 10-bit resolution<br>$AV_{REFP}$ = $V_{DD}$ | 1.9 V ≤ $AV_{REFP}$ ≤ 5.5 V | | 1.2 | ±5.0 | LSB |
| Conversion time | $t_{CONV}$ | 10-bit resolution | 3.6 V ≤ $V_{DD}$ ≤ 5.5 V | 2.125 | | 39 | μs |
| | | | 2.7 V ≤ $V_{DD}$ ≤ 5.5 V | 3.1875 | | 39 | μs |
| | | | 1.9 V ≤ $V_{DD}$ ≤ 5.5 V | 17 | | 39 | μs |
| Zero-scale error[Notes 1, 2] | $E_{ZS}$ | 10-bit resolution<br>$AV_{REFP}$ = $V_{DD}$ | 1.9 V ≤ $AV_{REFP}$ ≤ 5.5 V | | | ±0.35 | %FSR |
| Full-scale error[Notes 1, 2] | $E_{FS}$ | 10-bit resolution<br>$AV_{REFP}$ = $V_{DD}$ | 1.9 V ≤ $AV_{REFP}$ ≤ 5.5 V | | | ±0.35 | %FSR |
| Integral linearity error[Note 1] | ILE | 10-bit resolution<br>$AV_{REFP}$ = $V_{DD}$ | 1.9 V ≤ $AV_{REFP}$ ≤ 5.5 V | | | ±3.5 | LSB |
| Differential linearity error[Note 1] | DLE | 10-bit resolution<br>$AV_{REFP}$ = $V_{DD}$ | 1.9 V ≤ $AV_{REFP}$ ≤ 5.5 V | | | ±2.0 | LSB |
| Reference voltage (+) | $AV_{REFP}$ | | | 1.9 | | $V_{DD}$ | V |
| Analog input voltage | $V_{AIN}$ | | | 0 | | $AV_{REFP}$ | V |
| | $V_{BGR}$ | Select interanal reference voltage output<br>2.4 V ≤ $V_{DD}$ ≤ 5.5 V, HS (high-speed main) mode | | 1.38 | 1.45 | 1.5 | V |

**Notes 1.** Excludes quantization error (±1/2 LSB).

**2.** This value is indicated as a ratio (%FSR) to the full-scale value.

**(2) When reference voltage (+) = $V_{DD}$ (ADREFP1 = 0, ADREFP0 = 0), reference voltage (–) = $V_{SS}$ (ADREFM = 0), target pins: ANI0 to ANI5 and internal reference voltage**

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V, reference voltage (+) = $V_{DD}$, reference voltage (–) = $V_{SS}$)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Resolution | RES | | | 8 | | 10 | bit |
| Overall error[Notes 1, 2] | AINL | 10-bit resolution | 1.9 V ≤ $V_{DD}$ ≤ 5.5 V | | 1.2 | ±10.5 | LSB |
| Conversion time | $t_{CONV}$ | 10-bit resolution | 3.6 V ≤ $V_{DD}$ ≤ 5.5 V | 2.125 | | 39 | μs |
| | | | 2.7 V ≤ $V_{DD}$ ≤ 5.5 V | 3.1875 | | 39 | μs |
| | | | 1.9 V ≤ $V_{DD}$ ≤ 5.5 V | 17 | | 39 | μs |
| Zero-scale error[Notes 1, 2] | $E_{ZS}$ | 10-bit resolution | 1.9 V ≤ $V_{DD}$ ≤ 5.5 V | | | ±0.85 | %FSR |
| Full-scale error[Notes 1, 2] | $E_{FS}$ | 10-bit resolution | 1.9 V ≤ $V_{DD}$ ≤ 5.5 V | | | ±0.85 | %FSR |
| Integral linearity error[Note 1] | ILE | 10-bit resolution | 1.9 V ≤ $V_{DD}$ ≤ 5.5 V | | | ±4.0 | LSB |
| Differential linearity error[Note 1] | DLE | 10-bit resolution | 1.9 V ≤ $V_{DD}$ ≤ 5.5 V | | | ±2.0 | LSB |
| Analog input voltage | $V_{AIN}$ | | | 0 | | $V_{DD}$ | V |
| | $V_{BGR}$ | Select interanal reference voltage output, 2.4 V ≤ $V_{DD}$ ≤ 5.5 V, HS (high-speed main) mode | | 1.38 | 1.45 | 1.5 | V |

**Notes 1.** Excludes quantization error (±1/2 LSB).

**2.** This value is indicated as a ratio (%FSR) to the full-scale value.

**Caution When using reference voltage (+) = $V_{DD}$, taking into account the voltage drop due to the effect of the power switching circuit of the battery backup function and use the A/D conversion result. In addition, enter HALT mode during A/D conversion and set $V_{DD}$ port to input.**

**(3) When reference voltage (+) = Internal reference voltage (ADREFP1 = 1, ADREFP0 = 0), reference voltage (–) = $AV_{REFM}$/ANI1 (ADREFM = 1), target pins: ANI0, ANI2 to ANI5**

**($T_A$ = –40 to +85°C, 2.4 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V, reference voltage (+) = $V_{BGR}$, reference voltage (–) = $AV_{REFM}$ = 0 V, HS (high-speed main) mode)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Resolution | RES | | | 8 | | | bit |
| Conversion time | $t_{CONV}$ | 8-bit resolution | 2.4 V ≤ $V_{DD}$ ≤ 5.5 V | 17 | | 39 | μs |
| Zero-scale error[Notes 1, 2] | $E_{ZS}$ | 8-bit resolution | 2.4 V ≤ $V_{DD}$ ≤ 5.5 V | | | ±0.60 | %FSR |
| Integral linearity error[Note 1] | ILE | 8-bit resolution | 2.4 V ≤ $V_{DD}$ ≤ 5.5 V | | | ±2.0 | LSB |
| Differential linearity error[Note 1] | DLE | 8-bit resolution | 2.4 V ≤ $V_{DD}$ ≤ 5.5 V | | | ±1.0 | LSB |
| Reference voltage (+) | $V_{BGR}$ | | | 1.38 | 1.45 | 1.5 | V |
| Analog input voltage | $V_{AIN}$ | | | 0 | | $V_{BGR}$ | V |

**Notes 1.** Excludes quantization error (±1/2 LSB).

**2.** This value is indicated as a ratio (%FSR) to the full-scale value.

### 37.6.2 24-bit ΔΣ A/D converter characteristics

#### (1) Reference voltage

**($T_A$ = –40 to +85°C, $AV_{DD} \leq V_{DD}$ + 0.3 V, 2.4 V ≤ $AV_{DD}$ ≤ 5.5 V, 2.4 V ≤ $V_{DD}$ ≤ 5.5 V, $V_{SS}$ = $AV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Internal reference voltage | $V_{AVRTO}$ | | | 0.8 | | V |
| Temperature coefficient for internal reference voltage | dREF/dt | 0.47 µF capacitor connected to AREGC, AVRT, and AVCM pins | | 30 | 90 | ppm/°C |

#### (2) Analog input

**($T_A$ = –40 to +85°C, $AV_{DD} \leq V_{DD}$ + 0.3 V, 2.4 V ≤ $AV_{DD}$ ≤ 5.5 V, 2.4 V ≤ $V_{DD}$ ≤ 5.5 V, $V_{SS}$ = $AV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Input voltage range (differential voltage) | $V_{AIN}$ | x1 gain | –500 | | 500 | mV |
| | | x2 gain | –250 | | 250 | |
| | | x4 gain | –125 | | 125 | |
| | | x8 gain | –62.5 | | 62.5 | |
| | | x16 gain | –31.25 | | 31.25 | |
| | | x32 gain (for current channels) | –15.625 | | 15.625 | |
| Input gain | ainGAIN | x1 gain | | 1 | | dB |
| | | x2 gain | | 2 | | |
| | | x4 gain | | 4 | | |
| | | x8 gain | | 8 | | |
| | | x16 gain | | 16 | | |
| | | x32 gain (for current channels) | | 32 | | |
| Input impedance | ainRIN | Differential voltage | 150 | 360 | | kΩ |
| | | Single-ended voltage | 100 | 240 | | |

### (3) 4 kHz sampling mode

**($T_A$ = –40 to +85°C, $AV_{DD} \leq V_{DD}$ + 0.3 V, 2.4 V ≤ $AV_{DD}$ ≤ 5.5 V, 2.4 V ≤ $V_{DD}$ ≤ 5.5 V, $V_{SS}$ = $AV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Operation clock | $f_{DSAD}$ | $f_X$ oscillation clock, input external clock or high-speed on-chip oscillator clock is used | | 12 | | MHz |
| Sampling frequency | $f_S$ | | | 3906.25 | | Hz |
| Oversampling frequency | $f_{OS}$ | | | 1.5 | | MHz |
| Output data rate | $T_{DATA}$ | | | 256 | | µs |
| Data width | RES | | | 24 | | bit |
| SNDR | SNDR | x1 gain<br>High-speed system clock is selected as operating clock of 24-bit ΔΣ A/D converter (bit 0 of PCKC register (DSADCK) = 1) | | 80 | | dB |
| | | x16 gain<br>High-speed system clock is selected as operating clock of 24-bit ΔΣ A/D converter (bit 0 of PCKC register (DSADCK) = 1) | 69 | 74 | | |
| | | x32 gain<br>High-speed system clock is selected as operating clock of 24-bit ΔΣ A/D converter (bit 0 of PCKC register (DSADCK) = 1) | 65 | 69 | | |
| Passband (low pass band) | $f_{Chpf}$ | At –3 dB (phase in high pass filter not adjusted)<br>Bits 7 and 6 of DSADHPFCR register (DSADCOF1, DSADCOF0) = 00 | | 0.607 | | Hz |
| | | At –3 dB (phase in high pass filter not adjusted)<br>Bits 7 and 6 of DSADHPFCR register (DSADCOF1, DSADCOF0) = 01 | | 1.214 | | Hz |
| | | At –3 dB (phase in high pass filter not adjusted)<br>Bits 7 and 6 of DSADHPFCR register (DSADCOF1, DSADCOF0) = 10 | | 2.429 | | Hz |
| | | At –3 dB (phase in high pass filter not adjusted)<br>Bits 7 and 6 of DSADHPFCR register (DSADCOF1, DSADCOF0) = 11 | | 4.857 | | Hz |
| In-band ripple 1 | rp1 | 45 Hz to 55 Hz @50 Hz<br>54 Hz to 66 Hz @60 Hz | –0.01 | | 0.01 | dB |
| In-band ripple 2 | rp2 | 45 Hz to 275 Hz @50 Hz<br>54 Hz to 330 Hz @60 Hz | –0.1 | | 0.1 | |
| In-band ripple 3 | rp3 | 45 Hz to 1100 Hz @50 Hz<br>54 Hz to 1320 Hz @60 Hz | –0.1 | | 0.1 | |
| Passband (high pass band) | $f_{Clpf}$ | –3 dB | | 1672 | | Hz |
| Stopband (high pass band) | fatt | –80 dB | | 2545 | | Hz |
| Out-band attenuation | ATT1 | $f_S$ | –80 | | | dB |
| | ATT2 | 2 $f_S$ | –80 | | | dB |

**(4) 2 kHz sampling mode**

**($T_A$ = –40 to +85°C, $AV_{DD} \leq V_{DD}$ + 0.3 V, 2.4 V ≤ $AV_{DD}$ ≤ 5.5 V, 2.4 V ≤ $V_{DD}$ ≤ 5.5 V, $V_{SS}$ = $AV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Operation clock | $f_{DSAD}$ | $f_X$ oscillation clock, input external clock or high-speed on-chip oscillator clock is used | | 12 | | MHz |
| <R> Sampling frequency | $f_S$ | | | 1953.125 | | Hz |
| Oversampling frequency | $f_{OS}$ | | | 0.75 | | MHz |
| Output data rate | $T_{DATA}$ | | | 512 | | µs |
| Data width | RES | | | 24 | | bit |
| SNDR | SNDR | x1 gain<br>High-speed system clock is selected as operating clock of 24-bit ΔΣ A/D converter (bit 0 of PCKC register (DSADCK) = 1) | | 80 | | dB |
| | | x16 gain<br>High-speed system clock is selected as operating clock of 24-bit ΔΣ A/D converter (bit 0 of PCKC register (DSADCK) = 1) | 69 | 74 | | |
| | | x32 gain<br>High-speed system clock is selected as operating clock of 24-bit ΔΣ A/D converter (bit 0 of PCKC register (DSADCK) = 1) | 65 | 69 | | |
| Passband (low pass band) | $f_{Chpf}$ | At –3 dB (phase in high pass filter not adjusted) | | 0.303 | | Hz |
| In-band ripple 1 | rp1 | 45 Hz to 55 Hz @50 Hz<br>54 Hz to 66 Hz @60 Hz | –0.01 | | 0.01 | dB |
| In-band ripple 2 | rp2 | 45 Hz to 275 Hz @50 Hz<br>54 Hz to 330 Hz @60 Hz | –0.1 | | 0.1 | |
| In-band ripple 3 | rp3 | 45 Hz to 660 Hz @50 Hz<br>54 Hz to 550 Hz @60 Hz | –0.1 | | 0.1 | |
| Passband (high pass band) | $f_{Clpf}$ | –3 dB | | 836 | | Hz |
| Stopband (high pass band) | fatt | –80 dB | | 1273 | | Hz |
| Out-band attenuation | ATT1 | $f_S$ | –80 | | | dB |
| | ATT2 | 2 $f_S$ | –80 | | | dB |

### 37.6.3 Temperature sensor 2 characteristics

**($T_A$ = –40 to +85°C, 2.4 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V, HS (high-speed main) mode)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Temperature sensor 2 output voltage | $V_{OUT}$ | | | 0.67 | | V |
| Temperature coefficient | $F_{VTMPS2}$ | Temperature sensor that depends on the temperature | –11.7 | –10.7 | –9.7 | mV/°C |
| Operation stabilization wait time[Note] | $t_{TMPON}$ | Operable | | 15 | 50 | µs |
| | $t_{TMPCHG}$ | Switching mode | | 5 | 15 | µs |

**Note** Time to drop to output stable value ±5LSB (±7 mV) or less.

### 37.6.4 Comparator

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V )**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Input voltage range | Ivref | | | 0 | | $V_{DD}$ – 1.4 | V |
| | Ivcmp | | | –0.3 | | $V_{DD}$ + 0.3 | V |
| Output delay | td | $V_{DD}$ = 3.0 V<br>Input slew rate > 50 mV/μs | Comparator high-speed mode, standard mode | | | 1.2 | μs |
| | | | Comparator high-speed mode, window mode | | | 2.0 | μs |
| | | | Comparator low-speed mode, standard mode | | 3 | 5.0 | μs |
| High-electric-potential reference voltage | VTW+ | Comparator high-speed mode, window mode | | | 0.76$V_{DD}$ | | V |
| Low-electric-potential reference voltage | VTW– | Comparator high-speed mode, window mode | | | 0.24$V_{DD}$ | | V |
| Operation stabilization wait time | $t_{CMP}$ | | | 100 | | | μs |
| Reference output voltage | $V_{CMPREF}$ | | | 1.00 | 1.45 | 1.50 | V |

### 37.6.5 POR circuit characteristics

**($T_A$ = –40 to +85°C, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Detection voltage | $V_{POR}$ | When power supply rises[Note 1] | 1.47 | 1.51 | 1.55 | V |
| | $V_{PDR}$ | When power supply falls[Note 2] | 1.46 | 1.50 | 1.54 | V |

**Notes 1.** Be sure to maintain the reset state until the power supply voltage rises over the minimum $V_{DD}$ value in the operating voltage range specified in **37.4 AC Characteristics**, by using the voltage detector or external reset pin.

**2.** If the power supply voltage falls while the voltage detector is off, be sure to either shift to STOP mode or execute a reset by using the voltage detector or external reset pin before the power supply voltage falls below the minimum operating voltage specified in **37.4 AC Characteristics**.

### 37.6.6 LVD circuit characteristics

#### LVD Detection Voltage of Reset Mode and Interrupt Mode

**($T_A$ = –40 to +85°C, $V_{PDR} \leq V_{DD} = EV_{DD} \leq 5.5$ V, $V_{SS} = EV_{SS} = 0$ V)**

| Parameter | | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Detection voltage | Supply voltage level | $V_{LVD0}$ | When power supply rises | 3.98 | 4.06 | 4.24 | V |
| | | | When power supply falls | 3.90 | 3.98 | 4.16 | V |
| | | $V_{LVD1}$ | When power supply rises | 3.68 | 3.75 | 3.92 | V |
| | | | When power supply falls | 3.60 | 3.67 | 3.84 | V |
| | | $V_{LVD2}$ | When power supply rises | 3.07 | 3.13 | 3.29 | V |
| | | | When power supply falls | 3.00 | 3.06 | 3.22 | V |
| | | $V_{LVD3}$ | When power supply rises | 2.96 | 3.02 | 3.18 | V |
| | | | When power supply falls | 2.90 | 2.96 | 3.12 | V |
| | | $V_{LVD4}$ | When power supply rises | 2.86 | 2.92 | 3.07 | V |
| | | | When power supply falls | 2.80 | 2.86 | 3.01 | V |
| | | $V_{LVD5}$ | When power supply rises | 2.76 | 2.81 | 2.97 | V |
| | | | When power supply falls | 2.70 | 2.75 | 2.91 | V |
| | | $V_{LVD6}$ | When power supply rises | 2.66 | 2.71 | 2.86 | V |
| | | | When power supply falls | 2.60 | 2.65 | 2.80 | V |
| | | $V_{LVD7}$ | When power supply rises | 2.56 | 2.61 | 2.76 | V |
| | | | When power supply falls | 2.50 | 2.55 | 2.70 | V |
| | | $V_{LVD8}$ | When power supply rises | 2.45 | 2.50 | 2.65 | V |
| | | | When power supply falls | 2.40 | 2.45 | 2.60 | V |
| | | $V_{LVD9}$ | When power supply rises | 2.05 | 2.09 | 2.23 | V |
| | | | When power supply falls | 2.00 | 2.04 | 2.18 | V |
| | | $V_{LVD10}$ | When power supply rises | 1.94 | 1.98 | 2.12 | V |
| | | | When power supply falls | 1.90 | 1.94 | 2.08 | V |
| Minimum pulse width | | $t_{LW}$ | | 300 | | | μs |
| Detection delay time | | | | | | 300 | μs |

**LVD Detection Voltage of Interrupt & Reset Mode**

**($T_A$ = –40 to +85°C, $V_{PDR} \leq V_{DD} = EV_{DD} \leq 5.5$ V, $V_{SS} = EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|---|
| | $V_{LVD8}$ | VPOC2, VPOC1, VPOC0 = 0, 1, 0, falling reset voltage | | | 2.40 | 2.45 | 2.60 | V |
| | $V_{LVD7}$ | | LVIS1, LVIS0 = 1, 0 | Rising release reset voltage | 2.56 | 2.61 | 2.76 | V |
| | | | | Falling interrupt voltage | 2.50 | 2.55 | 2.70 | V |
| | $V_{LVD6}$ | | LVIS1, LVIS0 = 0, 1 | Rising release reset voltage | 2.66 | 2.71 | 2.86 | V |
| | | | | Falling interrupt voltage | 2.60 | 2.65 | 2.80 | V |
| | $V_{LVD1}$ | | LVIS1, LVIS0 = 0, 0 | Rising release reset voltage | 3.68 | 3.75 | 3.92 | V |
| | | | | Falling interrupt voltage | 3.60 | 3.67 | 3.84 | V |
| | $V_{LVD5}$ | VPOC2, VPOC1, VPOC0 = 0, 1, 1, falling reset voltage | | | 2.70 | 2.75 | 2.91 | V |
| | $V_{LVD4}$ | | LVIS1, LVIS0 = 1, 0 | Rising release reset voltage | 2.86 | 2.92 | 3.07 | V |
| | | | | Falling interrupt voltage | 2.80 | 2.86 | 3.01 | V |
| | $V_{LVD3}$ | | LVIS1, LVIS0 = 0, 1 | Rising release reset voltage | 2.96 | 3.02 | 3.18 | V |
| | | | | Falling interrupt voltage | 2.90 | 2.96 | 3.12 | V |
| | $V_{LVD0}$ | | LVIS1, LVIS0 = 0, 0 | Rising release reset voltage | 3.98 | 4.06 | 4.24 | V |
| | | | | Falling interrupt voltage | 3.90 | 3.98 | 4.16 | V |

### 37.6.7 Power supply voltage rising slope characteristics

**($T_A$ = –40 to +85°C, $V_{SS}$ = 0 V)**

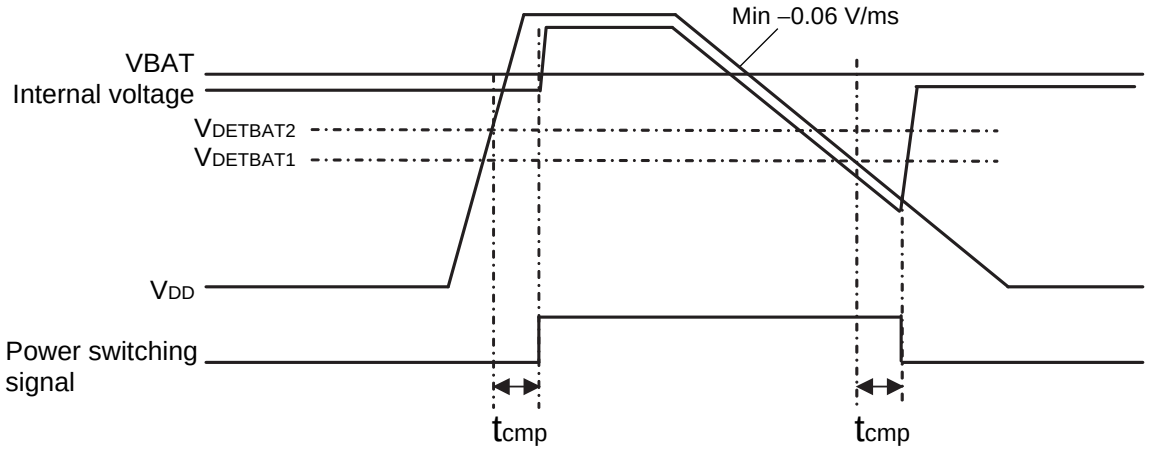
| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Power supply voltage rising slope | $SV_{DDR}$ | | | | 54 | V/ms |

**Caution Make sure to keep the internal reset state by the LVD circuit or an external reset until $V_{DD}$ reaches the operating voltage range shown in 37.4 AC Characteristics.**

## 37.7 Battery Backup Function

**($T_A$ = –40 to +85°C, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Power swiching detection voltage | $V_{DETBAT1}$ | $V_{DD} \rightarrow$ VBAT | 1.92 | 2.00 | 2.08 | V |
| | $V_{DETBAT2}$ | VBAT $\rightarrow V_{DD}$ | 2.02 | 2.10 | 2.18 | V |
| $V_{DD}$ fall slope | $SV_{DDF}$ | | –0.06 | | | V/ms |
| Response time of power switch detector | $t_{cmp}$ | | | | 300 | µs |



## 37.8 LCD Characteristics

### 37.8.1 Resistance division method

**(1) Static display mode**

**($T_A$ = –40 to +85°C, $V_{L4}$ (MIN.) ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| LCD drive voltage | $V_{L4}$ | | 2.0 | | $V_{DD}$ | V |

**(2) 1/2 bias method, 1/4 bias method**

**($T_A$ = –40 to +85°C, $V_{L4}$ (MIN.) ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| LCD drive voltage | $V_{L4}$ | | 2.7 | | $V_{DD}$ | V |

**(3) 1/3 bias method**

**($T_A$ = –40 to +85°C, $V_{L4}$ (MIN.) ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| LCD drive voltage | $V_{L4}$ | | 2.5 | | $V_{DD}$ | V |

### 37.8.2 Internal voltage boosting method

#### (1) 1/3 bias method

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| LCD output voltage variation range | $V_{L1}$ | C1 to C4[Note 1] = 0.47 μF[Note 2] | VLCD = 04H | 0.90 | 1.00 | 1.08 | V |
| | | | VLCD = 05H | 0.95 | 1.05 | 1.13 | V |
| | | | VLCD = 06H | 1.00 | 1.10 | 1.18 | V |
| | | | VLCD = 07H | 1.05 | 1.15 | 1.23 | V |
| | | | VLCD = 08H | 1.10 | 1.20 | 1.28 | V |
| | | | VLCD = 09H | 1.15 | 1.25 | 1.33 | V |
| | | | VLCD = 0AH | 1.20 | 1.30 | 1.38 | V |
| | | | VLCD = 0BH | 1.25 | 1.35 | 1.43 | V |
| | | | VLCD = 0CH | 1.30 | 1.40 | 1.48 | V |
| | | | VLCD = 0DH | 1.35 | 1.45 | 1.53 | V |
| | | | VLCD = 0EH | 1.40 | 1.50 | 1.58 | V |
| | | | VLCD = 0FH | 1.45 | 1.55 | 1.63 | V |
| | | | VLCD = 10H | 1.50 | 1.60 | 1.68 | V |
| | | | VLCD = 11H | 1.55 | 1.65 | 1.73 | V |
| | | | VLCD = 12H | 1.60 | 1.70 | 1.78 | V |
| | | | VLCD = 13H | 1.65 | 1.75 | 1.83 | V |
| Doubler output voltage | $V_{L2}$ | C1 to C4[Note 1] = 0.47 μF | | 2 $V_{L1}$–0.10 | 2 $V_{L1}$ | 2 $V_{L1}$ | V |
| Tripler output voltage | $V_{L4}$ | C1 to C4[Note 1] = 0.47 μF | | 3 $V_{L1}$–0.15 | 3 $V_{L1}$ | 3 $V_{L1}$ | V |
| Reference voltage setup time[Note 2] | $t_{VWAIT1}$ | | | 5 | | | ms |
| Voltage boost wait time[Note 3] | $t_{VWAIT2}$ | C1 to C4[Note 1] = 0.47 μF | | 500 | | | ms |

**Notes 1.** This is a capacitor that is connected between voltage pins used to drive the LCD.
C1: A capacitor connected between CAPH and CAPL
C2: A capacitor connected between $V_{L1}$ and GND
C3: A capacitor connected between $V_{L2}$ and GND
C4: A capacitor connected between $V_{L4}$ and GND
C1 = C2 = C3 = C4 = 0.47 μF±30 %

**2.** This is the time required to wait from when the reference voltage is specified by using the VLCD register (or when the internal voltage boosting method is selected (by setting the MDSET1 and MDSET0 bits of the LCDM0 register to 01B) if the default value reference voltage is used) until voltage boosting starts (VLCON = 1).

**3.** This is the wait time from when voltage boosting is started (VLCON = 1) until display is enabled (LCDON = 1).

**(2) 1/4 bias method**

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| LCD output voltage variation range | $V_{L1}$ | C1 to C5[Note 1] = 0.47 μF[Note 2] | VLCD = 04H | 0.90 | 1.00 | 1.08 | V |
| | | | VLCD = 05H | 0.95 | 1.05 | 1.13 | V |
| | | | VLCD = 06H | 1.00 | 1.10 | 1.18 | V |
| | | | VLCD = 07H | 1.05 | 1.15 | 1.23 | V |
| | | | VLCD = 08H | 1.10 | 1.20 | 1.28 | V |
| | | | VLCD = 09H | 1.15 | 1.25 | 1.33 | V |
| | | | VLCD = 0AH | 1.20 | 1.30 | 1.38 | V |
| Doubler output voltage | $V_{L2}$ | C1 to C5[Note 1] = 0.47 μF | | 2 $V_{L1}$–0.08 | 2 $V_{L1}$ | 2 $V_{L1}$ | V |
| Tripler output voltage | $V_{L3}$ | C1 to C5[Note 1] = 0.47 μF | | 3 $V_{L1}$–0.12 | 3 $V_{L1}$ | 3 $V_{L1}$ | V |
| Quadruply output voltage | $V_{L4}$ | C1 to C5[Note 1] = 0.47 μF | | 4 $V_{L1}$–0.16 | 4 $V_{L1}$ | 4 $V_{L1}$ | V |
| Reference voltage setup time[Note 2] | $t_{VWAIT1}$ | | | 5 | | | ms |
| Voltage boost wait time[Note 3] | $t_{VWAIT2}$ | C1 to C5[Note 1] = 0.47 μF | | 500 | | | ms |

**Notes 1.** This is a capacitor that is connected between voltage pins used to drive the LCD.
C1: A capacitor connected between CAPH and CAPL
C2: A capacitor connected between $V_{L1}$ and GND
C3: A capacitor connected between $V_{L2}$ and GND
C4: A capacitor connected between $V_{L3}$ and GND
C5: A capacitor connected between $V_{L4}$ and GND
C1 = C2 = C3 = C4 = C5 = 0.47 μF±30 %

**2.** This is the time required to wait from when the reference voltage is specified by using the VLCD register (or when the internal voltage boosting method is selected (by setting the MDSET1 and MDSET0 bits of the LCDM0 register to 01B) if the default value reference voltage is used) until voltage boosting starts (VLCON = 1).

**3.** This is the wait time from when voltage boosting is started (VLCON = 1) until display is enabled (LCDON = 1).

### 37.8.3 Capacitor split method

#### (1) 1/3 bias method

**($T_A$ = –40 to +85°C, 2.2 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| $V_{L4}$ voltage | $V_{L4}$ | C1 to C4 = 0.47 μF[Note 2] | | $V_{DD}$ | | V |
| $V_{L2}$ voltage | $V_{L2}$ | C1 to C4 = 0.47 μF[Note 2] | 2/3 $V_{L4}$– 0.1 | 2/3 $V_{L4}$ | 2/3 $V_{L4}$ + 0.1 | V |
| $V_{L1}$ voltage | $V_{L1}$ | C1 to C4 = 0.47 μF[Note 2] | 1/3 $V_{L4}$– 0.1 | 1/3 $V_{L4}$ | 1/3 $V_{L4}$ + 0.1 | V |
| Capacitor split wait time[Note 1] | $t_{VWAIT}$ | | 100 | | | ms |

**Notes 1.** This is the wait time from when voltage bucking is started (VLCON = 1) until display is enabled (LCDON = 1).

**2.** This is a capacitor that is connected between voltage pins used to drive the LCD.

C1: A capacitor connected between CAPH and CAPL

C2: A capacitor connected between $V_{L1}$ and GND

C3: A capacitor connected between $V_{L2}$ and GND

C4: A capacitor connected between $V_{L4}$ and GND

C1 = C2 = C3 = C4 = 0.47 μF±30 %

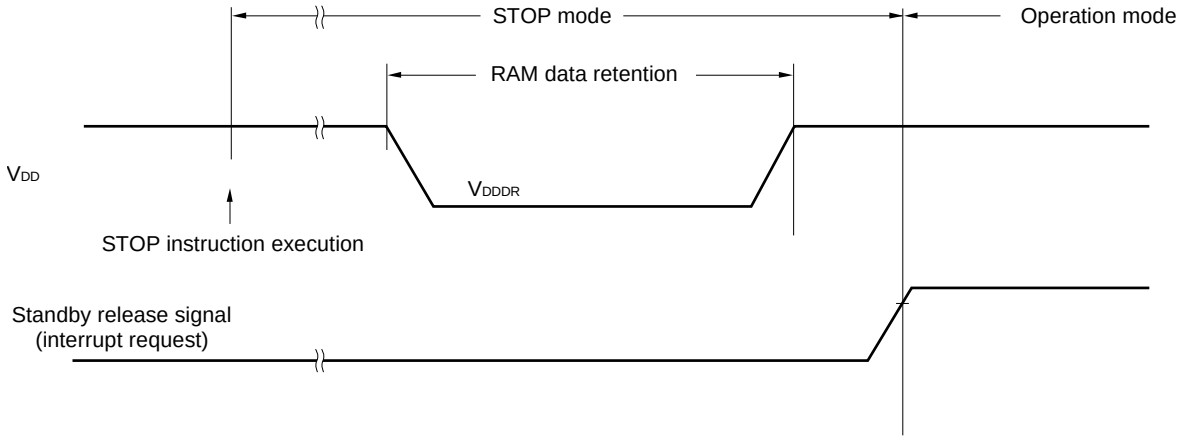
## <R> 37.9 RAM Data Retention Characteristics

**($T_A$ = –40 to +85°C, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Data retention supply voltage | $V_{DDDR}$ | | 1.46[Note] | | 5.5 | V |

<R> **Note** The value depends on the POR detection voltage. When the voltage drops, the RAM data is retained before a POR reset is effected, but RAM data is not retained when a POR reset is effected.

<R>




## 37.10 Flash Memory Programming Characteristics

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| System clock frequency | $f_{CLK}$ | 1.9 V ≤ $V_{DD}$ ≤ 5.5 V | 1 | | 24 | MHz |
| <R> Number of code flash rewrites[Notes 1, 2, 3] | $C_{erwr}$ | Retained for 20 years<br>$T_A$ = 85°C | 1,000 | | | Times |

**Notes 1.** 1 erase + 1 write after the erase is regarded as 1 rewrite. The retaining years are until next rewrite after the rewrite.

**2.** When using flash memory programmer and Renesas Electronics self programming library

**3.** This characteristic indicates the flash memory characteristic and based on Renesas Electronics reliability test.

## 37.11 Dedicated Flash Memory Programmer Communication (UART)

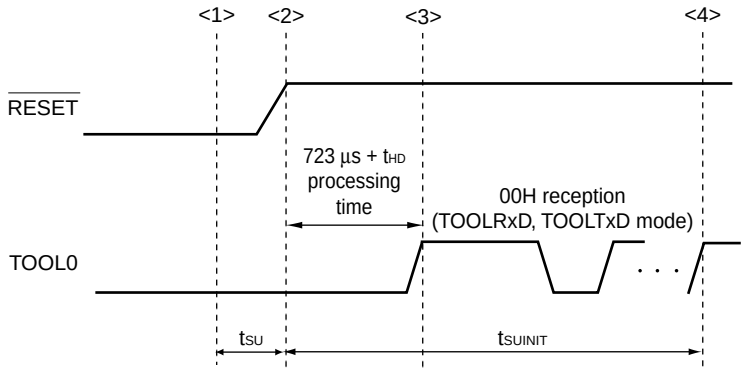
**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Transfer rate | | During serial programming | 115,200 | | 1,000,000 | bps |

## 37.12 Timing Specs for Switching Flash Memory Programming Modes

**($T_A$ = –40 to +85°C, 1.9 V ≤ $V_{DD}$ = $EV_{DD}$ ≤ 5.5 V, $V_{SS}$ = $EV_{SS}$ = 0 V)**

| Parameter | 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 (excluding the processing time of the firmware to control the flash memory) | $t_{HD}$ | 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 (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 segment shows that it is necessary to finish specifying the initial communication settings within 100 ms from when the resets end.

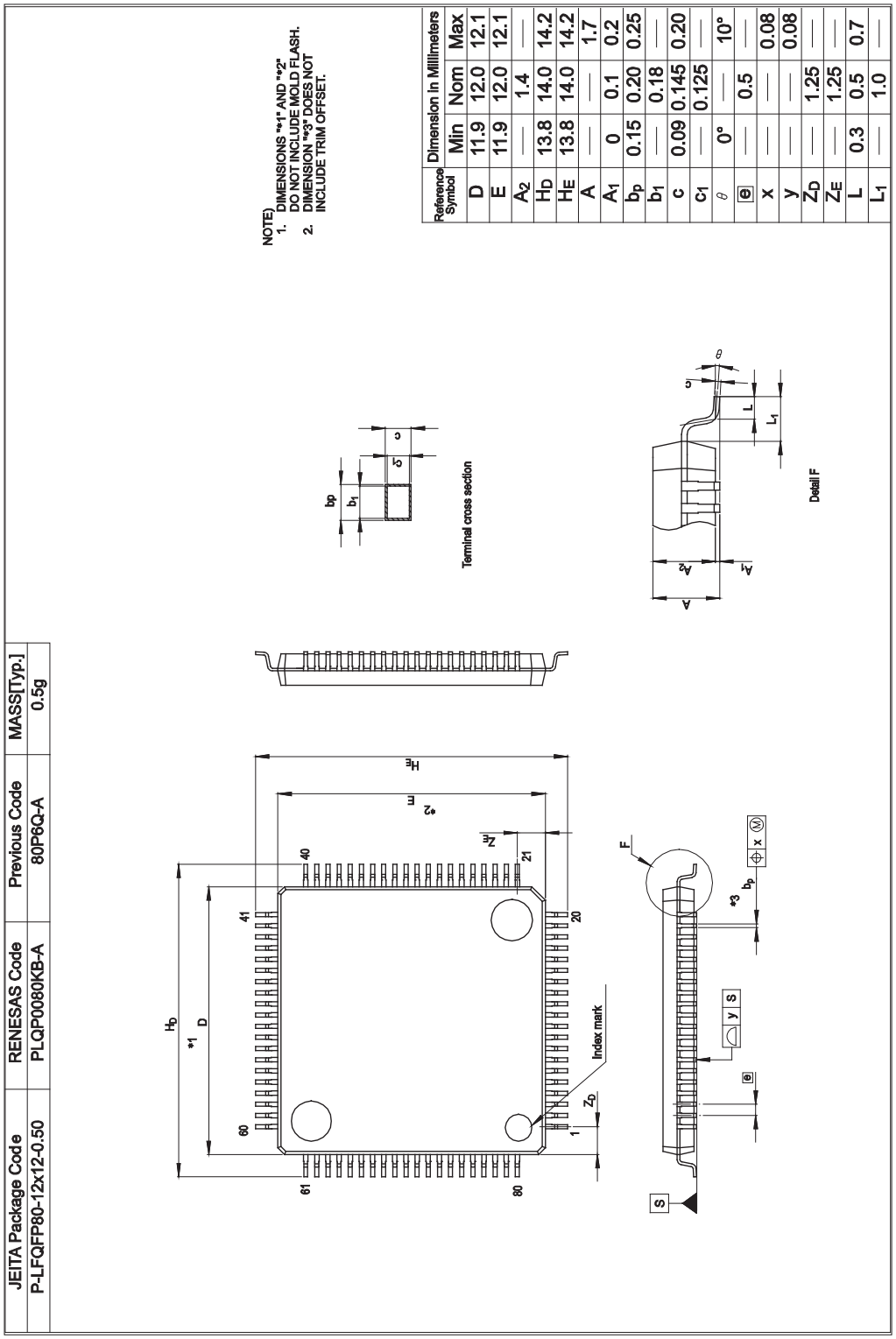
$t_{SU}$: Time to release the external reset after the TOOL0 pin is set to the low level.

$t_{HD}$: Time to hold the TOOL0 pin at the low level after the external reset is released (excluding the processing time of the firmware to control the flash memory)

# CHAPTER 38 PACKAGE DRAWINGS

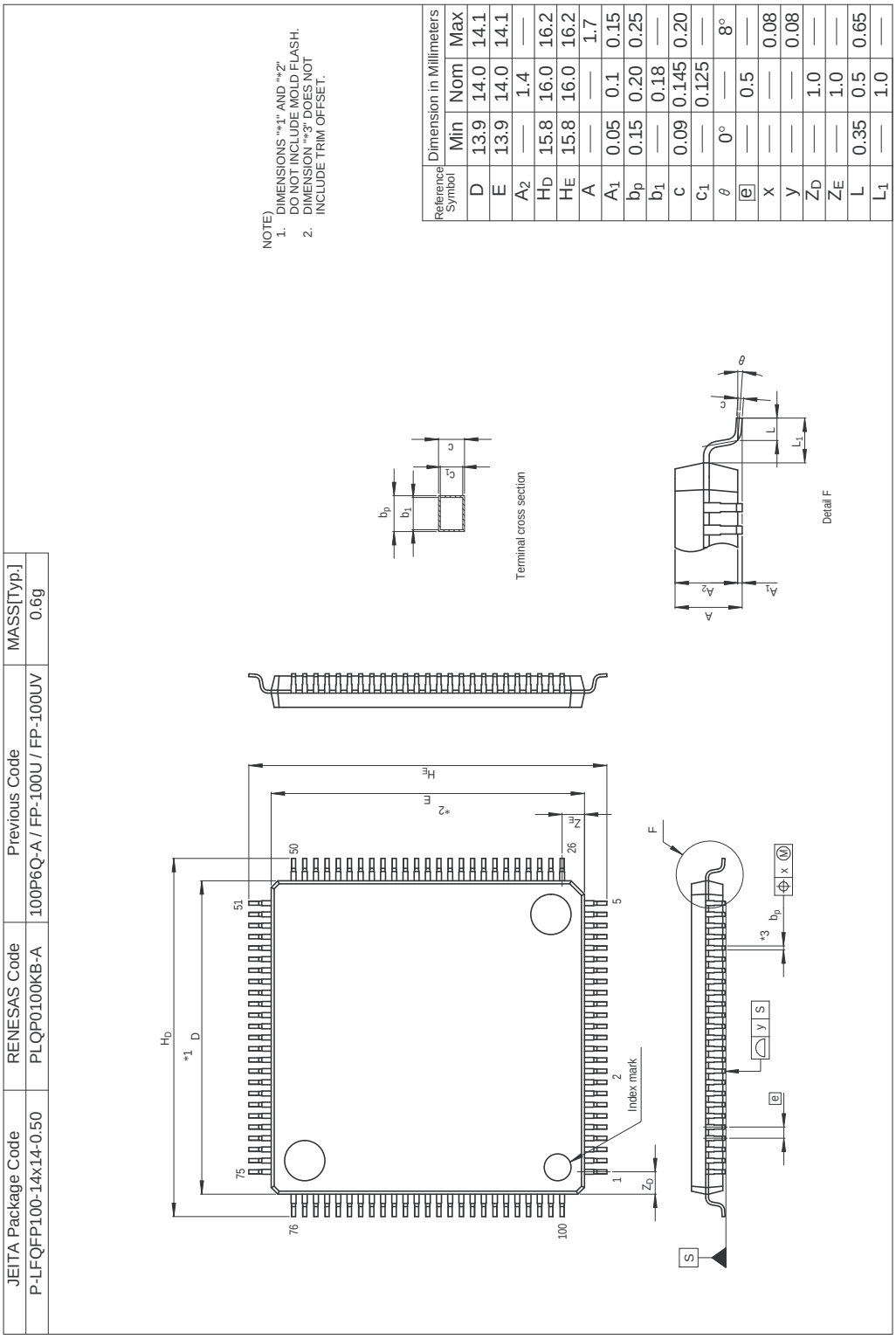
## 38.1 80-pin Products

R5F10MMEDFB, R5F10MMGDFB



## 38.2 100-pin Products

R5F10MPEDFB, R5F10MPGDFB



# APPENDIX A REVISION HISTORY

## A.1 Major Revisions in This Edition

(1/4)

| Page | Description | Classification |
|---|---|---|
| **CHAPTER 1 OUTLINE** | | |
| p.5 | Modification of Top View in **1.3.1 80-pin products** | (c) |
| p.6 | Modification of Top View in **1.3.2 100-pin products** | (c) |
| p.10 | Modification of Main system clock in**1.6 Outline of Functions** | (c) |
| **CHAPTER 2 PIN FUNCTIONS** | | |
| p.13, 14 | Modification of table item in **2.1.1 80-pin products** | (c) |
| p.15 to 17 | Modification of table item in **2.1.2 100-pin products** | (c) |
| p.24 | Modification of **Figure 2-3** in **2.4 Block Diagrams of Pins** | (a) |
| **CHAPTER 3 CPU ARCHITECTURE** | | |
| p.40 | Modification of Notes 1 in **Figure 3-2** in **3.1 Memory Space** | (c) |
| **CHAPTER 4 PORT FUNCTIONS** | | |
| p.102 | Modification of **Figure 4-9** in **4.3.9 LCD port function registers 0 to 5** | (c) |
| p.102 | Addition of Note in **Figure 4-9** in **4.3.9 LCD port function registers 0 to 5** | (c) |
| **CHAPTER 5 CLOCK GENERATOR** | | |
| p.123 | Addition of description in **5.1 Functions of Clock Generator** | (c) |
| p.133 | Modification of caution 6 in **5.3.3 Clock operation status control register** | (c) |
| p.153 | Addition of caution 2 in **5.6.2 Example of setting X1 oscillation clock** | (c) |
| p.161, 162 | Modification of **Table 5-4** in **5.6.5 Conditions before changing the CPU clock and processing after changing CPU clock** | (c) |
| p.164 | Addition of description in **5.6.7 Conditions before stopping clock oscillation** | (c) |
| **CHAPTER 7 TIMER ARRAY UNIT** | | |
| p.194 | Modification of **Figure 7-12** in **7.3.3 Timer mode register mn** | (c) |
| p.202 | Modification of caution in **7.3.8 Timer input select register 0** | (a) |
| p.229 | Deletion of caution in **7.6.4 Collective manipulation of TOmn bit** | (c) |
| p.257 | Modification of caution in **7.9.1 Operation as one-shot pulse output function** | (c) |
| **CHAPTER 8 REAL-TIME CLOCK 2** | | |
| p.282 | Modification of **Figure 8-1** in **8.2 Configuration of Real-time Clock 2** | (a) |
| p.292 | Addition of notes 1 and 2 in **8.3.6 Real-time clock control register 1** | (c) |
| **CHAPTER 9 SUBSYSTEM CLOCK FREQUENCY MEASUREMENT CIRCUIT** | | |
| p.320 | Modification of description in **9.3.5 Frequency measurement control register** | (a) |
| **CHAPTER 12 CLOCK OUTPUT/BUZZER OUTPUT CONTROLLER** | | |
| p.348 | Modification of description in **12.5 Cautions of Clock Output/Buzzer Output Controller** | (c) |

**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

(2/4)

| Page | Description | Classification |
|---|---|---|
| **CHAPTER 14 A/D CONVERTER** | | |
| p.365 | Modification of **Figure 14-4** in **14.3.2 A/D converter mode register 0** | (c) |
| p.394 | Modification of **Figure 14-29** in **14.7.1 Setting up software trigger mode** | (c) |
| p.395 | Modification of **Figure 14-30** in **14.7.2 Setting up hardware trigger no-wait mode** | (c) |
| p.396 | Modification of **Figure 14-31** in **14.7.3 Setting up hardware trigger wait mode** | (c) |
| p.397 | Modification of **Figure 14-32** in **14.7.4 Setup when temperature sensor output voltage/internal reference voltage is selected** | (c) |
| p.398 | Modification of **Figure 14-33** in **14.7.5 Setting up test mode** | (c) |
| p.402 | Modification of **Figure 14-37** in **14.8 SNOOZE Mode Function** | (c) |
| **CHAPTER 18 SERIAL ARRAY UNIT** | | |
| p.456 | Modification of **Figure 18-1** in **18.2 Configuration of Serial Array Unit** | (c) |
| p.457 | Modification of **Figure 18-2** in **18.2 Configuration of Serial Array Unit** | (c) |
| p.467 | Modification of description in **18.3.5 Serial data register mn (SDRmn)** | (c) |
| p.476 | Modification of description in **18.3.12 Serial output register m** | (c) |
| p.478 | Modification of **Figure 18-18** in **18.3.13 Serial output level register m** | (c) |
| p.544 | Modification of description in **18.5.7 SNOOZE mode function** | (c) |
| p.544 | Modification of **Figure 18-71** in **18.5.7 SNOOZE mode function** | (a) |
| p.544 | Modification of note in **18.5.7 SNOOZE mode function** | (c) |
| p.545 | Modification of **Figure 18-72** in **18.5.7 SNOOZE mode function** | (c) |
| p.546 | Modification of **Figure 18-73** in **18.5.7 SNOOZE mode function** | (a) |
| p.546 | Modification of note in **18.5.7 SNOOZE mode function** | (c) |
| p.547 | Modification of **Figure 18-74.** in **18.5.7 SNOOZE mode function** | (c) |
| p.570 | Modification of description in **18.6.3 SNOOZE mode function** | (c) |
| p.570 | Addition of caution 5 in **18.6.3 SNOOZE mode function** | (c) |
| p.572 | Modification of **Figure 18-90** in **18.6.3 SNOOZE mode function** | (a) |
| p.573 | Modification of **Figure 18-91** in **18.6.3 SNOOZE mode function** | (a) |
| p.574 | Modification of **Figure 18-92** in **18.6.3 SNOOZE mode function** | (c) |
| p.575 | Modification of **Figure 18-93** in **18.6.3 SNOOZE mode function** | (a) |
| p.576 | Modification of **Figure 18-94** in **18.6.3 SNOOZE mode function** | (c) |
| p.585 | Modification of **Figure 18-99** in **18.7.1 LIN transmission** | (a) |
| p.587 | Modification of **Figure 18-100** in **18.7.2 LIN reception** | (a) |
| p.588 | Modification of **Figure 18-101** in **18.7.2 LIN reception** | (c) |

**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

(3/4)

| Page | Description | Classification |
|---|---|---|
| **CHAPTER 19 SERIAL INTERFACE IICA** | | |
| p.630 | Addition of description in **19.3.6 IICA low-level width setting register n** | (c) |
| p.649 | Modification of calculation formula in **19.5.14 Communication reservation** | (c) |
| p.651 | Modification of note 1 in **Figure 19-27** in **19.5.14 Communication reservation** | (c) |
| p.655 | Modification of **Figure 19-28** in **19.5.16 Communication operations** | (c) |
| p.656 | Modification of **Figure 19-29 (1/3)** in **19.5.16 Communication operations** | (c) |
| p.657 | Modification of note in **Figure 19-29** in **19.5.16 Communication operations** | (c) |
| p.660 | Modification of **Figure 19-30** in **19.5.16 Communication operations** | (c) |
| **CHAPTER 21 LCD CONTROLLER/DRIVER** | | |
| p.713 | Addition of remark in **21.3.2 LCD mode register 1** | (c) |
| p.714 | Deletion of remark in **21.3.3 Subsystem clock supply mode control register (moved to 21.3.2)** | (c) |
| p.719 | Modification of note in **21.3.7 LCD port function registers 0 to 5** | (c) |
| **CHAPTER 22 DATA TRANSFER CONTROLLER (DTC)** | | |
| p.766 | Addition of description in **CHAPTER 22 DATA TRANSFER CONTROLLER** | (c) |
| p.767 | Modification of **Table 22-1** in **22.1 Functions of DTC** | (c) |
| p.771 | Modification of **Figure 22-3** in **22.3.2 Control data allocation** | (c) |
| p.772 | Modification of **Table 22-4** in **22.3.2 Control data allocation** | (c) |
| p.773 | Addition of **Figure 22-4** in **22.3.3 Vector table** | (c) |
| p.784 | Modification of description in **22.4.2 Normal mode** | (c) |
| p.784 | Modification of **Figure 22-16** in **22.4.2 Normal mode** | (c) |
| p.791 | Addition of description in **22.5.3 DTC pending instruction** | (c) |
| **CHAPTER 23 INTERRUPT FUNCTIONS** | | |
| p.817, 818 | Addition of **23.4.4 Interrupt servicing during division instruction** | (c) |
| p.819 | Addition of description in **23.4.5 Interrupt request hold** | (c) |
| **CHAPTER 24 STANDBY FUNCTION** | | |
| p.823 | Modification of **Table 24-1 (1/2)** in **24.3.1 HALT mode** | (c) |
| p.825 | Modification of **Table 24-1 (2/2)** in **24.3.1 HALT mode** | (c) |
| p.835 | Modification of **Table 24-3** in **24.3.3 SNOOZE mode** | |
| **CHAPTER 25 RESET FUNCTION** | | |
| p.841 | Deletion of caution in **25.1 Timing of Reset Operation** | (c) |
| p.846 | Modification of title in **Figure 25-5** in **25.3.1 Reset control flag register (RESF)** | (c) |
| **CHAPTER 26 POWER-ON-RESET CIRCUIT** | | |
| p.850 | Modification of note 3,4 in **26.3 Operation of Power-on-reset Circuit** | (c) |
| p.852 | Modification of note 3 in **26.3 Operation of Power-on-reset Circuit** | (a) |
| **CHAPTER 27 VOLTAGE DETECTOR** | | |
| p.854 | Modification of description in **27.1 Functions of Voltage Detector** | (a) |
| p.854 | Modification of table in **27.1 Functions of Voltage Detector** | (c) |
| **CHAPTER 30 SAFETY FUNCTIONS** | | |
| p.897 | Modification of note in **30.3.6 Invalid memory access detection function** | (a) |
| **CHAPTER 32 OPTION BYTE** | | |
| p.907 | Modification of description in **32.1.1 User option byte** | (c) |
| p.911 | Modification of **Figure 32-3** in **32.2 Format of User Option Byte** | (c) |

**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

(4/4)

| Page | Description | Classification |
|---|---|---|
| **CHAPTER 33 FLASH MEMORY** | | |
| p.916 | Modification of **Table 33-1** in **33.1 Serial Programming Using Flash Memory Programmer** | (c) |
| p.917 | Modification of **Figure 33-1** in **33.1.1 Programming environment** | (c) |
| p.917 | Modification of **Figure 33-2** in **33.1.2 Communication mode** | (c) |
| p.918 | Modification of **Table 33-2** in **33.1.2 Communication mode** | (c) |
| p.918 | Modification of note 2 in **33.1.2 Communication mode** | (c) |
| p.918 | Addition of **Figure 33-3** in **33.2.1 Programming environment** | (c) |
| p.918 | Addition of note 2 in **33.2.1 Programming environment** | (c) |
| p.919 | Modification of **Figure 33-4** in **33.2.2 Communication mode** | (c) |
| p.919 | Addition of note 2 in **33.2.2 Communication mode** | (c) |
| p.919 | Modification of **Table 33-3** in **33.2.2 Communication mode** | (c) |
| p.919 | Addition of note 2 in **33.2.2 Communication mode** | (c) |
| p.927 | Modification of remark 1 in **33.5 Self-Programming** | (b) |
| **CHAPTER 36 INSTRUCTION SET** | | |
| p.954 | Addition of caution in **36.2 Operation List** | (c) |
| **CHAPTER 37 ELECTRICAL SPECIFICATIONS** | | |
| p.971 | Modification of on-chip pull-up resistance in **37.3.1 Pin characteristics** | (a) |
| p.972 | Modification of supply current in **37.3.2 Supply current characteristics** | (a) |
| p.1012 | Modification of sampling frequency in **37.6.2 24-bit ΔΣ A/D converter characteristics** | (a) |
| p.1021 | Modification of title in **37.9 RAM Data Retention Characteristics** | (c) |
| p.1021 | Modification of note in **37.9 RAM Data Retention Characteristics** | (c) |
| p.1021 | Modification of figure in **37.9 RAM Data Retention Characteristics** | (c) |
| p.1021 | Modification of table in **37.10 Flash Memory Programming Characteristics** | (c) |

**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 Revision History of Preceding Editions

Here is the revision history of the preceding editions. Chapter indicates the chapter of each edition.

(1/6)

| Edition | Description | Chapter |
|---|---|---|
| Rev.1.00 | Change of **1.2 Ordering Information** | **CHAPTER 1 OUTLINE** |
| | Change of **2.1 Port Function List** | **CHAPTER 2 PIN FUNCTIONS** |
| | Change of **2.2 Functions other than port pins** | |
| | Change of **2.3 Pin I/O Circuits and Recommended Connection of Unused Pins** | |
| | Addition of **3.1 Overview** | **CHAPTER 3 CPU ARCHITECTURE** |
| | Change of **3.2 Memory Space** | |
| | Change of **3.3 Processor Registers** | |
| | Change of **4.5 Settings of Port Related Register When Using Alternate Function** | **CHAPTER 4 PORT FUNCTIONS** |
| | Change of **5.1 Functions of Clock Generator** | **CHAPTER 5 CLOCK GENERATOR** |
| | Change of **5.2 Configuration of Clock Generator** | |
| | Change of **5.3 Registers Controlling Clock Generator** | |
| | Change of **5.4.4 Low-speed on-chip oscillator** | |
| | Change of **5.5 Clock Generator Operation** | |
| | Change of **5.6 Controlling the Clock** | |
| | Modification of **6.1 High-speed On-chip Oscillator Clock Frequency Correction Function** | **CHAPTER 6 HIGH-SPEED ON-CHIP OSCILLATOR CLOCK FREQUENCY CORRECTION FUNCTION** |
| | Change of **7.2 Configuration of Timer Array Unit** | **CHAPTER 7 TIMER ARRAY UNIT** |
| | Change of **7.3 Registers Controlling Timer Array Unit** | |
| | Change of **7.5 Operation of Counter** | |
| | Change of **7.7 Independent Channel Operation Function of Timer Array Unit** | |
| | Change of **7.8 Simultaneous Channel Operation Function of Timer Array Unit** | |
| | Change of **8.1 Functions of High Accuracy Real-time Clock** | **CHAPTER 8 HIGH ACCURACY REAL-TIME CLOCK** |
| | Change of **8.2 Configuration of High Accuracy Real-time Clock** | |
| | Change of **8.3 Registers Controlling High Accuracy Real-time Clock** | |
| | Change of **8.4 High Accuracy Real-time Clock Operation** | |
| | Change of **9.1 Subsystem Clock Frequency Measurement Circuit** | **CHAPTER 9 SUBSYSTEM CLOCK FREQUENCY MEASUREMENT CIRCUIT** |
| | Change of **9.2 Configuration of Subsystem Clock Frequency Measurement Circuit** | |
| | Change of **9.3 Registers Controlling Subsystem Clock Frequency Measurement Circuit** | |
| | Change of **9.4 Subsystem Clock Frequency Measurement Circuit Operation** | |
| | Change of **10.2 Configuration of 12-bit Interval Timer** | **CHAPTER 10 12-BIT INTERVAL TIMER** |
| | Change of **10.3 Registers Controlling 12-bit Interval Timer** | |
| | Change of **10.4 12-bit Interval Timer Operation** | |
| | Change of **11.1 Overview** | **CHAPTER 11 8-BIT INTERVAL TIMER** |
| | Change of **11.3 Registers** | |
| | Change of **11.4 Operation** | |

(2/6)

| Edition | Description | Chapter |
|---|---|---|
| Rev.1.00 | Change of **12.1 Functions of Clock Output/Buzzer Output Controller** | **CHAPTER 12 CLOCK OUTPUT/BUZZER OUTPUT CONTROLLER** |
| | Change of **12.2 Configuration of Clock Output/Buzzer Output Controller** | |
| | Change of **12.3 Registers Controlling Clock Output/Buzzer Output Controller** | |
| | Change of **12.4 Operations of Clock Output/Buzzer Output Controller** | |
| | Change of **13.1 Functions of Watchdog Timer** | **CHAPTER 13 WATCHDOG TIMER** |
| | Change of **13.2 Configuration of Watchdog Timer** | |
| | Change of **13.4 Operation of Watchdog Timer** | |
| | Change of **14.1 Function of A/D Converter** | **CHAPTER 14 A/D CONVERTER** |
| | Change of **14.2 Configuration of A/D Converter** | |
| | Change of **14.3 Registers Controlling A/D Converter** | |
| | Change of **14.4 A/D Converter Conversion Operations** | |
| | Change of **14.6 A/D Converter Operation Modes** | |
| | Change of **14.7 A/D Converter Setup Flowchart** | |
| | Change of **14.9 How to Read A/D Converter Characteristics Table** | |
| | Change of **14.10 Cautions for A/D Converter** | |
| | Change of **15.1 Functions of Temperature Sensor** | **CHAPTER 15 HIGH-ACCURACY TEMPERATURE SENSOR** |
| | Change of **15.2 Registers** | |
| | Change of **15.3 Setting Procedures** | |
| | Change of **16.1 Functions of 24-bit ΔΣ A/D Converter** | **CHAPTER 16 24-BIT ΔΣ A/D CONVERTER** |
| | Change of **16.2 Registers** | |
| | Change of **16.3 Operation** | |
| | Change of **16.4 Notes on Using 24-Bit ΔΣ A/D Converter** | |
| | Change of **17.1 Overview** | **CHAPTER 17 COMPARATOR** |
| | Change of **17.4 Operation** | |
| | Change of **18.1 Functions of Serial Array Unit** | **CHAPTER 18 SERIAL ARRAY UNIT** |
| | Change of **18.2 Configuration of Serial Array Unit** | |
| | Change of **18.3 Registers Controlling Serial Array Unit** | |
| | Change of **18.5 Operation of 3-Wire Serial I/O (CSI00) Communication** | |
| | Change of **18.6 Operation of UART (UART0 to UART2) Communication** | |
| | Change of **18.7 LIN Communication Operation** | |
| | Change of **18.8 Operation of Simplified $I^2C$ (IIC00, IIC10) Communication** | |
| | Change of **19.3 Registers Controlling Serial Interface IICA** | **CHAPTER 19 SERIAL INTERFACE IICA** |
| | Change of **19.5 $I^2C$ Bus Definitions and Control Methods** | |
| | Change of **20.2 Registers** | **CHAPTER 20 IrDA** |
| | Change of **20.3 Operation** | |
| | Change of **20.4 Usage Notes on IrDA** | |
| | Change of **21.2 Configuration of LCD Controller/Driver** | **CHAPTER 21 LCD CONTROLLER/DRIVER** |
| | Change of **21.3 Registers Controlling LCD Controller/Driver** | |
| | Change of **21.6 Setting the LCD Controller/Driver** | |
| | Change of **21.7 Operation stop procedure** | |
| | Change of **21.8 Supplying LCD Drive Voltages $V_{L1}$, $V_{L2}$, $V_{L3}$, and $V_{L4}$** | |
| | Change of **21.10 Display Modes** | |

(3/6)

| Edition | Description | Chapter |
|---|---|---|
| Rev.1.00 | Change of **22.2 Registers** | **CHAPTER 22 DATA TRANSFER CONTROLLER** |
| | Change of **22.4 Notes on DTC** | |
| | Change of **23.2 Interrupt Sources and Configuration** | **CHAPTER 23 INTERRUPT FUNCTIONS** |
| | Change of **23.3 Registers Controlling Interrupt Functions** | |
| | Change of **23.4 Interrupt Servicing Operations** | |
| | Change of **24.3 Standby Function Operation** | **CHAPTER 24 STANDBY FUNCTION** |
| | Change of **CHAPTER 25 RESET FUNCTION** | **CHAPTER 25 RESET FUNCTION** |
| | Change of **25.1 Register for Confirming Reset Source** | |
| | Change of **26.1 Functions of Power-on-reset Circuit** | **CHAPTER 26 POWER-ON-RESET CIRCUIT** |
| | Change of **26.2 Configuration of Power-on-reset Circuit** | |
| | Change of **26.3 Operation of Power-on-reset Circuit** | |
| | Change of **27.1 Functions of Voltage Detector** | **CHAPTER 27 VOLTAGE DETECTOR** |
| | Change of **27.2 Configuration of Voltage Detector** | |
| | Change of **27.3 Registers Controlling Voltage Detector** | |
| | Change of **27.4 Operation of Voltage Detector** | |
| | Change of **27.5 Cautions for Voltage Detector** | |
| | Change of **28.1 Functions of Battery Backup** | **CHAPTER 28 BATTERY BACKUP FUNCTION** |
| | Change of **28.2 Registers** | |
| | Change of **28.3 Operation** | |
| | Change of **28.4 Usage Notes** | |
| | Change of **29.1 Functions of Oscillation Stop Detector** | **CHAPTER 29 OSCILLATION STOP DETECTOR** |
| | Change of **29.2 Configuration of Oscillation Stop Detector** | |
| | Change of **29.3 Registers Used by Oscillation Stop Detector** | |
| | Change of **30.1 Overview of Safety Functions** | **CHAPTER 30 SAFETY FUNCTIONS** |
| | Change of **30.3 Operation of Safety Functions** | |
| | Change of **32.1 Functions of Option Bytes** | **CHAPTER 32 OPTION BYTE** |
| | Change of **32.2 Format of User Option Byte** | |
| | Change of **CHAPTER 33 FLASH MEMORY** | **CHAPTER 33 FLASH MEMORY** |
| | Change of **33.1 Writing to Flash Memory by Using Flash Memory Programmer** | |
| | Change of **33.2 Writing to Flash Memory by Using External Device (that Incorporates UART)** | |
| | Change of **33.3 Connection of Pins on Board** | |
| | Change of **33.4 Programming Method** | |
| | Change of **33.5 Security Settings** | |
| | Change of **33.6 Self-Programming** | |
| | Change of **34.1 Connecting E1 On-chip Debugging Emulator to RL78/I1B** | **CHAPTER 34 ON-CHIP DEBUG FUNCTION** |

(4/6)

| Edition | Description | Chapter |
|---|---|---|
| Rev.1.00 | Change of **37.1 Absolute Maximum Ratings** | **CHAPTER 37 ELECTRICAL SPECIFICATIONS** |
| | Change of **37.2 Oscillator Characteristics** | |
| | Change of **37.3 DC Characteristics** | |
| | Change of **37.4 AC Characteristics** | |
| | Change of **37.5 Peripheral Functions Characteristics** | |
| | Change of **37.6 Analog Characteristics** | |
| | Change of **37.7 Battery Backup Function** | |
| | Change of **37.8 LCD Characteristics** | |
| | Addition of **37.11 Dedicated Flash Memory Programmer Communication (UART)** | |
| | Change of **37.12 Timing Specs for Switching Flash Memory Programming Modes** | |
| Rev.2.00 | Change of high accuracy RTC to RTC2, and high accuracy real-time clock to real-time clock 2 | Throughout |
| | Change of high accuracy temperature sensor to temperature sensor 2 | |
| | Modification of **1.1 Features** | **CHAPTER 1 OUTLINE** |
| | Modification of **1.2 List of Part Numbers** | |
| | Modification of **1.3 Pin Configuration (Top View)** | |
| | Modification of **2.1 Port Function List** | **CHAPTER 2 PIN FUNCTIONS** |
| | Modification of **2.2 Functions Other than Port Pins** | |
| | Modification of **2.3 Connection of Unused Pins** | |
| | Addition of **2.4 Block Diagrams of Pins** | |
| | Modification of **3.1 Memory Space** | **CHAPTER 3 CPU ARCHITECTURE** |
| | Modification of **3.2 Processor Registers** | |
| | Modification of **3.3 Instruction Address Addressing** | |
| | Modification of **3.4 Addressing for Processing Data Addresses** | |
| | Modification of **4.2 Port Configuration** | **CHAPTER 4 PORT FUNCTIONS** |
| | Modification of **4.3 Registers Controlling Port Function** | |
| | Modification of **4.4 Port Function Operations** | |
| | Modification of **4.5 Register Settings When Using Alternate Function** | |
| | Modification of **4.6 Cautions When Using Port Function** | |
| | Modification of **5.3 Registers Controlling Clock Generator** | **CHAPTER 5 CLOCK GENERATOR** |
| | Modification of **5.4 System Clock Oscillator** | |
| | Modification of **5.6 Controlling the Clock** | |
| | Addition of **5.7 Resonator and Oscillator Constants** | |
| | Modification of **7.2 Configuration of Timer Array Unit** | **CHAPTER 7 TIMER ARRAY UNIT** |
| | Modification of **7.3 Registers Controlling Timer Array Unit** | |
| | Modification of **7.5 Operation of Counter** | |
| | Modification of **7.6 Channel Output (TOmn Pin) Control** | |
| | Addition of **7.7 Timer Input (TImn) Control** | |
| | Modification of **7.8 Independent Channel Operation Function of Timer Array Unit** | |
| | Modification of **7.9 Simultaneous Channel Operation Function of Timer Array Unit** | |
| | Modification of **8.1 Functions of Real-time Clock 2** | **CHAPTER 8 REAL-TIME CLOCK 2** |
| | Modification of **8.2 Configuration of Real-time Clock 2** | |
| | Modification of **8.3 Registers Controlling Real-time Clock 2** | |
| | Modification of **8.4 Real-time Clock 2 Operation** | |

(5/6)

| Edition | Description | Chapter |
|---|---|---|
| Rev.2.00 | Modification of **10.3 Registers Controlling 12-bit Interval Timer** | **CHAPTER 10 12-BIT INTERVAL TIMER** |
| | Modification of **11.4 Operation** | **CHAPTER 11 8-BIT INTERVAL TIMER** |
| | Modification of **11.5 Notes on 8-bit Interval Timer** | |
| | Modification of **12.5 Cautions of Clock Output/Buzzer Output Controller** | **CHAPTER 12 CLOCK OUTPUT/BUZZER OUTPUT CONTROLLER** |
| | Modification of **13.2 Configuration of Watchdog Timer** | **CHAPTER 13 WATCHDOG TIMER** |
| | Modification of **14.3 Registers Controlling A/D Converter** | **CHAPTER 14 A/D CONVERTER** |
| | Modification of **14.4 A/D Converter Conversion Operations** | |
| | Modification of **14.6 A/D Converter Operation Modes** | |
| | Modification of **14.7 A/D Converter Setup Flowchart** | |
| | Modification of **14.8 SNOOZE Mode Function** | |
| | Modification of **14.10 Cautions for A/D Converter** | |
| | Modification of **15.1 Functions of Temperature Sensor** | **CHAPTER 15 TEMPERATURE SENSOR 2** |
| | Modification of **16.1 Functions of 24-bit ΔΣ A/D Converter** | **CHAPTER 16 24-BIT ΔΣ A/D CONVERTER** |
| | Modification of **16.2 Registers** | |
| | Modification of **17.1 Functions of Comparator** | **CHAPTER 17 COMPARATOR** |
| | Modification of **17.2 Configuration of Comparator** | |
| | Modification of **17.3 Registers Controlling Comparator** | |
| | Modification of **17.4 Operation** | |
| | Modification of **18.1 Functions of Serial Array Unit** | **CHAPTER 18 SERIAL ARRAY UNIT** |
| | Modification of **18.2 Configuration of Serial Array Unit** | |
| | Modification of **18.3 Registers Controlling Serial Array Unit** | |
| | Modification of **18.5 Operation of 3-Wire Serial I/O (CSI00) Communication** | |
| | Modification of **18.6 Operation of UART (UART0 to UART2) Communication** | |
| | Modification of **18.7 LIN Communication Operation** | |
| | Modification of **18.8 Operation of Simplified $I^2C$ (IIC00, IIC10) Communication** | |
| | Modification of **19.1 Functions of Serial Interface IICA** | **CHAPTER 19 SERIAL INTERFACE IICA** |
| | Modification of **19.3 Registers Controlling Serial Interface IICA** | |
| | Modification of **19.4 $I^2C$ Bus Mode Functions** | |
| | Modification of **19.5 $I^2C$ Bus Definitions and Control Methods** | |
| | Modification of **20.4 Usage Notes on IrDA** | **CHAPTER 20 IrDA** |
| | Modification of **CHAPTER 21 LCD CONTROLLER/DRIVER** | **CHAPTER 21 LCD CONTROLLER/DRIVER** |
| | Modification of **21.1 Functions of LCD Controller/Driver** | |
| | Modification of **21.3 Registers Controlling LCD Controller/Driver** | |
| | Modification of **21.5 Selection of LCD Display Register** | |
| | Modification of **21.6 Setting the LCD Controller/Driver** | |
| | Modification of **22.1 Functions of DTC** | **CHAPTER 22 DATA TRANSFER CONTROLLER (DTC)** |
| | Modification of **22.2 Configuration of DTC** | |
| | Modification of **22.3 Registers Controlling DTC** | |
| | Modification of **22.4 DTC Operation** | |
| | Modification of **22.5 Notes on DTC** | |
| | Modification of **23.3 Registers Controlling Interrupt Functions** | **CHAPTER 23 INTERRUPT FUNCTIONS** |
| | Modification of **23.4 Interrupt Servicing Operations** | |

(6/6)

| Edition | Description | Chapter |
|---|---|---|
| Rev.2.00 | Modification of **24.2 Registers Controlling Standby Function** | **CHAPTER 24 STANDBY FUNCTION** |
| | Modification of **24.3 Standby Function Operation** | |
| | Modification of **CHAPTER 25 RESET FUNCTION** | **CHAPTER 25 RESET FUNCTION** |
| | Modification of **25.1 Timing of Reset Operation** | |
| | Modification of **25.2 States of Operation During Reset Periods** | |
| | Modification of **25.3 Register for Confirming Reset Source** | |
| | Modification of **26.3 Operation of Power-on-reset Circuit** | **CHAPTER 26 POWER-ON-RESET CIRCUIT** |
| | Modification of **27.1 Functions of Voltage Detector** | **CHAPTER 27 VOLTAGE DETECTOR** |
| | Modification of **27.2 Configuration of Voltage Detector** | |
| | Modification of **27.3 Registers Controlling Voltage Detector** | |
| | Modification of **27.4 Operation of Voltage Detector** | |
| | Modification of **29.3 Registers Used by Oscillation Stop Detector** | **CHAPTER 29 OSCILLATION STOP DETECTOR** |
| | Modification of **30.1 Overview of Safety Functions** | **CHAPTER 30 SAFETY FUNCTIONS** |
| | Modification of **30.3 Operation of Safety Functions** | |
| | Modification of **31.1 Regulator Overview** | **CHAPTER 31 REGULATOR** |
| | Modification of **32.1 Functions of Option Bytes** | **CHAPTER 32 OPTION BYTE** |
| | Modification of **32.2 Format of User Option Byte** | |
| | Modification of **33.1 Serial Programming Using Flash Memory Programmer** | **CHAPTER 33 FLASH MEMORY** |
| | Modification of **33.2 Serial Programming Using External Device (That Incorporates UART)** | |
| | Modification of **33.4 Serial Programming Method** | |
| | Modification of **33.5 Self-Programming** | |
| | Modification of **33.6 Security Settings** | |
| | Modification of **34.1 Connecting E1 On-chip Debugging Emulator** | **CHAPTER 34 ON-CHIP DEBUG FUNCTION** |
| | Modification of **36.1 Conventions Used in Operation List** | **CHAPTER 36 INSTRUCTION SET** |
| | Modification of **37.1 Absolute Maximum Ratings** | **CHAPTER 37 ELECTRICAL SPECIFICATIONS** |
| | Modification of **37.2 Oscillator Characteristics** | |
| | Modification of **37.3 DC Characteristics** | |
| | Modification of **37.4 AC Characteristics** | |
| | Modification of **37.5 Peripheral Functions Characteristics** | |
| | Modification of **37.6 Analog Characteristics** | |
| | Modification of **37.8 LCD Characteristics** | |
| | Modification of **37.10 Flash Memory Programming Characteristics** | |
| | Modification of **37.11 Dedicated Flash Memory Programmer Communication (UART)** | |
| | Modification of **38.1 80-pin Products** | **CHAPTER 38 PACKAGE DRAWINGS** |

RL78/I1B User’s Manual: Hardware

Publication Date: Rev.1.00 Aug 30, 2013
Rev.2.10 Apr 25, 2016

Published by: Renesas Electronics Corporation



**SALES OFFICES** Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.


**California Eastern Laboratories, Inc.**
4590 Patrick Henry Drive, Santa Clara, California 95054-1817, U.S.A.
Tel: +1-408-919-2500, Fax: +1-408-988-0279

**Renesas Electronics Europe Limited**
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

**Renesas Electronics Europe GmbH**
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

**Renesas Electronics (China) Co., Ltd.**
Room 1709, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100191, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

**Renesas Electronics (Shanghai) Co., Ltd.**
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, P. R. China 200333
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

**Renesas Electronics Hong Kong Limited**
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

**Renesas Electronics Taiwan Co., Ltd.**
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

**Renesas Electronics Singapore Pte. Ltd.**
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

**Renesas Electronics Malaysia Sdn.Bhd.**
Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

**Renesas Electronics India Pvt. Ltd.**
No.777C, 100 Feet Road, HALII Stage, Indiranagar, Bangalore, India
Tel: +91-80-67208700, Fax: +91-80-67208777

**Renesas Electronics Korea Co., Ltd.**
12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141



© 2016 Renesas Electronics Corporation. All rights reserved.

Colophon 4.0

RL78/I1B

Renesas Electronics Corporation

R01UH0407EJ0210