CMS32L032 Reference Manual

Low power 32-bit microcontroller based on ARM® Cortex®-M0+

V1.0.2

Please note the following CMS IP policy

* China Micro Semicon (Shenzhen) Co., Ltd. (hereinafter referred to as the Company) has applied for a patent and enjoys absolute legal rights and interests. The patent rights related to the Company's MCUs or other products have not been authorized to be licensed, and any company, organization or individual who infringes the Company's patent rights through improper means will take all possible legal actions to curb the infringer's improper infringement and recover the losses suffered by the Company as a result of the infringement or the illegal benefits obtained by the infringer.

* The name and logo of China Micro Semicon (Shenzhen) Co., Ltd. are registered trademarks of the Company.

* The Company reserves the right to further explain the reliability, functionality and design improvements of the products in the data sheet. However, the Company is not responsible for the use of the Specification Contents. The applications mentioned herein are for illustrative purposes only and the Company does not warrant and does not represent that these applications can be applied without further modification, nor does it recommend that its products be used in places that may cause harm to persons due to malfunction or other reasons. The Company's products are not licensed for lifesaving, life-sustaining devices or systems as critical devices. The Company reserves the right to modify the product without prior notice, please refer to the official website www.mcu.com.cn for the latest information.

Documentation Instructions

This manual is a technical reference manual for the CMS32L032 microcontroller product. The technical reference manual is the application instruction material on how to use this series of products, including the structure, function description, working mode and register configuration of each functional module.

The technical reference manual is a description of all functional modules of this series of products. If you want to know the feature description of the product (that is, the functional configuration), you can refer to the respective data sheet.

The data sheet information is as follows:

CMS32L032xx: CMS32L032_datasheet_Vx.x.x. pdf

Usually in the early stage of chip selection, you shall first check the data sheet to evaluate whether the product can meet the functional requirements of the design; after basically selecting the required product, you need to check the technical reference manual to determine whether the working mode of each functional module does meet the requirement; When determining the selection and entering the programming design stage, you need to read the technical reference manual in detail to understand the specific implementation and register configuration of each function. Refer to the data sheet for information on voltages, currents, drive capabilities, and pin assignments when designing hardware.

For a detailed description of the Cortex-M0+ core, SysTick timer and NVIC, please refer to the respective ARM documents.

Contents

Documentation Instructions 2
Chapter 1 CPU 13
1.1 Overview 13
1.2 Cortex-M0+ core features 13
1.3 Debugging features 13
1.4 SWD interface pins 15
1.5 ARM reference document 16
Chapter 2 Port Function 17
2.1 Port function 17
2.2 Port multiplexing function 17
2.3 Registers for controlling port functions 18
2.3.1 Port mode register (PMxx) 21
2.3.2 Port register (Pxx) 22
2.3.3 Port set control register (PSETxx) 23
2.3.4 Port clear control register (PCLRxx) 24
2.3.5 Pull-up resistor selection register (PUxx) 25
2.3.6 Pull-down resistor selection register (PDxx) 26
2.3.7 Port output mode register (POMxx) 27
2.3.8 Port mode control register (PMCxx) 28
2.3.9 Port readback register (PREADxx) 29
2.3.10 Port multiplexing function configuration register (PxxCFG) 30
2.3.11 Description of the special function port RESINB 31
Chapter 3 System Architecture 32
3.1 Overview 32
3.2 System address partitioning 33
Chapter 4 Clock Generation Circuit 35
4.1 Function of clock generation circuit 35
4.2 Configuration of clock generation circuit 37
4.3 Registers for controlling clock generation circuit 40
4.3.1 Clock operation mode control register (CMC) 40
4.3.2 System clock control register (CKC) 42
4.3.3 Clock operation status control register (CSC) 43
4.3.4 Oscillation stabilization time counter status register (OSTC) 45
4.3.5 Oscillation stabilization time select register (OSTS) 47
4.3.6 Peripheral enable registers 0, 1 (PER0, PER1) 49
4.3.7 Subsystem clock supply mode control register (OSMC) 53
4.3.8 High-speed on-chip oscillator frequency select register (HOCODIV) 54
4.3.9 High-speed on-chip oscillator trimming register (HIOTRM) 55
4.3.10 Subsystem clock select register (SUBCKSEL) 56
4.3.11 Power mode control protection register (PMUKEY) 57
4.3.12 Power mode control register (PMUCTL) 57
4.4 System clock oscillation circuit 58
4.4.1 X1 oscillation circuit 58
4.4.2 XT1 oscillation circuit 59

4.4.3 High-speed on-chip oscillator 62
4.4.4 Low-speed on-chip oscillator 62
4.5 Operation of clock generation circuit 63
4.6 Clock control 65
4.6.1 Example of setting up a high-speed on-chip oscillator 65
4.6.2 Example of setting X1 oscillation circuit 67
4.6.3 Example of controlling XT1 oscillation clock 68
4.6.4 CPU clock status transition diagram 69
4.6.5 Conditions before CPU clock transfer and post-transfer processing 75
4.6.6 Time required to switch CPU clock and main system clock 77
4.6.7 Conditions before clock oscillation is stopped 78
4.7 High-speed on-chip oscillation correction 79
4.7.1 High-speed on-chip oscillation self-adjustment function 79
4.7.2 Description of register 80
4.7.3 Description of operation 81
4.7.4 Usage notes 85
Chapter 5 General-Purpose Timer Unit Timer4 86
5.1 Function of general-purpose timer unit 87
5.1.1 Independent channel operation function 87
5.1.2 Multi-channel linkage operation functions 89
5.1.3 8-bit timer operation function (channels 1 and 3 of unit 0 only) 91
5.1.4 LIN-bus supporting function (channel 3 of unit 0 only) 91
5.2 Structure of general-purpose timer unit 92
5.2.1 Register list of general-purpose timer unit 0 95
5.2.2 Register list of general-purpose timer unit 1 96
5.2.3 Timer count register mn (TCRmn) 97
5.2.4 Timer data register mn (TDRmn) 98
5.3 Registers for controlling general-purpose timer unit 99
5.3.1 Peripheral enable register 0 (PER0) 100
5.3.2 Timer clock select register m (TPSm) 101
5.3.3 Timer mode register mn (TMRmn) 104
5.3.4 Timer status register mn (TSRmn) 109
5.3.5 Timer channel enable status register m (TEm) 110
5.3.6 Timer channel start register m (TSm) 111
5.3.7 Timer channel stop register m (TTm) 112
5.3.8 Timer input output select register (TIOS0) 113
5.3.9 Timer output enable register m (TOEm) 114
5.3.10 Timer output register m (TOm) 115
5.3.11 Timer output level register m (TOLm) 116
5.3.12 Timer output mode register m (TOMm) 117
5.3.13 Noise filter enable register 1 (NFEN1) 118
5.3.14 Noise filter enable register 2 (NFEN2) 119
5.3.15 Registers controlling port functions of timer input/output pins 120
5.4 Basic rules of general-purpose timer unit 121
5.4.1 Basic rules of multi-channel linkage operation function 121
5.4.2 Basic rules of 8-bit timer operation function (channels 1 and 3 of unit 0 only) 123
5.5 Operation of counter 124

5.5.1 Count clock (F_{TCLK}) 124
5.5.2 Start timing of counter 126
5.5.3 Operation of counter 127
5.6 Channel output (TOmn pin) control 132
5.6.1 TOmn pin output circuit configuration 132
5.6.2 TOmn pin output setting 133
5.6.3 Cautions for channel output operation 134
5.6.4 One-time operation of TOmn bit 139
5.6.5 Timer interrupt and TOmn pin output when counting starts 140
5.7 Control of timer input (TImn) 141
5.7.1 Structure of TImn pin input circuit 141
5.7.2 Noise filter 142
5.7.3 Cautions on channel input operation 143
5.8 Independent channel operation function of general-purpose timer unit 144
5.8.1 Operation as interval timer/square wave output 144
5.8.2 Operation as external event counter 148
5.8.3 Operation as frequency divider 151
5.8.4 Operation as input pulse interval measurement 154
5.8.5 Operation as input signal high-/low-level width measurement 157
5.8.6 Operation as delay counter 161
5.9 Multi-channel linkage operation function for general purpose timer unit 164
5.9.1 Operation as single trigger pulse output function 164
5.9.2 Operation as PWM function 171
5.9.3 Operation as multiple PWM output function 178
Chapter 6 EPWM Output Control Circuit 186
6.1 Structure of output control circuit 186
6.2 Registers for controlling EPWM output control circuit 187
6.2.1 Peripheral enable register 1 (PER1) 187
6.2.2 EPWM input source select register (EPWMSRC) 187
6.2.3 EPWM force truncated input select register (EPWMSTC) 188
6.2.4 EPWM output control register (EPWMCTL) 189
6.2.5 EPWM force truncated output select register (EPWMSTL) 190
6.2.6 EPWM status register (EPWMSTR) 190
6.2.7 EPWM deadband control register (EPWMDTC) 191
6.2.8 Control registers for port functions of EPWM output pins 191
6.3 Operation of EPWM output control circuit 192
6.3.1 Initial setup 192
6.3.2 Normal operation 193
6.3.3 Force truncation processing 194
6.4 Control example of brushless DC motor 195
6.4.1 Example of hardware connections 195
6.4.2 Control timing of three-phase brushless DC motors 196
6.4.3 Example of register setting 197
6.5 Example of stepper motor control 198
6.5.1 Example of a hardware connection 198
6.5.2 Control method 199
6.5.3 Example of register setting 200

Chapter 7 Real-Time Clock 201
7.1 Function of real-time clock 201
7.2 Structure of real-time clock 201
7.3 Registers for controlling real-time clock 203
7.3.1 Peripheral enable register 0 (PER0) 204
7.3.2 Real-time clock selection register (RTCCL) 205
7.3.3 Real-time clock control register 0 (RTCC0) 206
7.3.4 Real-time clock control register 1 (RTCC1) 207
7.3.5 Clock error correction register (SUBCUD) 209
7.3.6 Second count register (SEC) 210
7.3.7 Minute count register (MIN) 210
7.3.8 Hour count register (HOUR) 211
7.3.9 Day count register (DAY) 213
7.3.10 Week count register (WEEK) 214
7.3.11 Month count register (MONTH) 215
7.3.12 Year count register (YEAR) 215
7.3.13 Alarm minute register (ALARMWM) 216
7.3.14 Alarm hour register (ALARMWH) 216
7.3.15 Alarm week register (ALARMWW) 217
7.3.16 Port mode register and port register 217
7.4 Operation of real-time clock 218
7.4.1 Start of real-time clock operation 218
7.4.2 Shifting to sleep mode after starting operation 219
7.4.3 Reading/writing real-time clock 220
7.4.4 Setting alarm of real-time clock 222
7.4.5 1Hz output of real-time clock 223
7.4.6 Example of clock error correction of real-time clock 224
Chapter 8 15-Bit Interval Timer 226
8.1 Function of 15-bit interval timer 226
8.2 Structure of 15-bit interval timer 226
8.3 Registers for controlling 15-bit interval timer 227
8.3.1 Peripheral enable register 0 (PER0) 227
8.3.2 Real-time clock selection register (RTCCL) 228
8.3.3 15-bit interval timer control register (ITMC) 229
8.4 Operation of 15-bit interval timer 230
8.4.1 Operation timing of 15-bit interval timer 230
8.4.2 Start of count operation and re-enter to sleep mode after returned from sleep mode 231
Chapter 9 Clock Output/Buzzer Output Controller 232
9.1 Function of clock output/buzzer output controller 232
9.2 Structure of clock output/buzzer output controller 233
9.3 Registers for controlling clock output/buzzer output controller 234
9.3.1 Clock output select register n (CKSn) 234
9.3.2 Registers for controlling clock output/buzzer output port functions 236
9.4 Operation of clock output/buzzer output controller 237
9.4.1 Operation as output pin 237
9.5 Cautions for clock output/buzzer output controller 237

Chapter 10 Watch Dog Timer 238
10.1 Function of watchdog timer 238
10.2 Structure of watch dog timer 238
10.3 Registers for controlling watchdog timer 240
10.3.1 Watchdog timer enable register (WDTE) 240
10.3.2 LOCKUP control register (LOCKCTL) and its protection register (PRCR) 241
10.4 Operation of watchdog timer 242
10.4.1 Operational control of watchdog timer 242
10.4.2 Setting overflow time of watchdog timer 244
10.4.3 Setting window open period of watchdog timer 245
10.4.4 Setting watchdog timer interval interrupt 246
10.4.5 Operation of watchdog timer during LOCKUP 246
Chapter 11 A/D Converter 247
11.1 Function of A/D converter 247
11.2 Registers for controlling A/D converter 249
11.2.1 Peripheral enable register 0 (PER0) 250
11.2.2 A/D converter mode register 0 (ADM0) 251
11.2.3 A/D converter mode register 1 (ADM1) 256
11.2.4 A/D converter mode register 2 (ADM2) 257
11.2.5 A/D converter trigger mode register (ADTRG) 258
11.2.6 Analog input channel specification register (ADS) 259
11.2.7 12-bit A/D conversion result register (ADCR) 261
11.2.8 8-bit A/D conversion result register (ADCRH) 262
11.2.9 Conversion result comparison upper limit setting register (ADUL) 262
11.2.10 Conversion result comparison lower limit setting register (ADLL) 263
11.2.11 A/D converter sampling time control register (ADNSMP) 263
11.2.12 Registers for controlling analog input port function 263
11.3 Input voltage and conversion results 264
11.4 Operation mode of A/D converter 265
11.4.1 Software trigger mode (select mode, sequential conversion mode) 265
11.4.2 Software trigger mode (select mode, single conversion mode) 266
11.4.3 Software trigger mode (scan mode, sequential conversion mode) 267
11.4.4 Software trigger mode (scan mode, single conversion mode) 268
11.4.5 Hardware trigger no-wait mode (select mode, sequential conversion mode) 269
11.4.6 Hardware trigger no-wait mode (select mode, single conversion mode) 270
11.4.7 Hardware trigger no-wait mode (scan mode, sequential conversion mode) 271
11.4.8 Hardware trigger no-wait mode (scan mode, single conversion mode) 272
11.4.9 Hardware trigger wait mode (select mode, sequential conversion mode) 273
11.4.10 Hardware trigger wait mode (select mode, single conversion mode) 274
11.4.11 Hardware trigger wait mode (scan mode, sequential conversion mode) 275
11.4.12 Hardware trigger wait mode (scan mode, single conversion mode) 276
11.5 A/D converter setup flowchart 277
11.5.1 Setting up software trigger mode 277
11.5.2 Setting up hardware trigger no-wait mode 278
11.5.3 Setting up hardware trigger wait mode 279
Chapter 12 General-Purpose Serial Communication Unit 280

12.1 Function of general-purpose serial communication unit 281
12.1.1 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11) 281
12.1.2 UART (UART0, UART1) 282
12.2 Structure of general-purpose serial communication unit 283
12.2.1 Shift register 285
12.2.2 Serial data register mn (SDRmn) 285
12.3 Registers for controlling general-purpose serial communication unit 287
12.3.1 Peripheral enable register 0 (PER0) 289
12.3.2 Serial clock select register m (SPSm) 290
12.3.3 Serial mode register mn (SMRmn) 291
12.3.4 Serial communication run setting register mn (SCRmn) 293
12.3.5 Serial data register mn (SDRmn) 296
12.3.6 Serial flag clear trigger register mn(SIRmn) 297
12.3.7 Serial status register mn (SSRmn) 298
12.3.8 Serial channel start register m(SSm) 300
12.3.9 Serial channel stop register m(STm) 301
12.3.10 Serial channel enable status register m (SEm) 302
12.3.11 Serial output enable register m (SOEm) 303
12.3.12 Serial output register m (SOm) 304
12.3.13 Serial output level register m (SOLm) 305
12.3.14 Input switching control register (ISC) 306
12.3.15 Noise filter enable register 0 (NFEN0) 307
12.3.16 Registers controlling port functions of serial input/output pins 308
12.4 Operation stop mode 309
12.4.1 Stopping the operation by units 309
12.4.2 Stopping the operation by channels 310
12.5 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11) communication 311
12.5.1 Master transmission 312
12.5.2 Master reception 320
12.5.3 Master transmission and reception 329
12.5.4 Slave transmission 337
12.5.5 Slave reception 345
12.5.6 Slave transmission and reception 351
12.5.7 Calculation of transmission clock frequency 360
12.5.8 Procedure for handling errors during 3-wire serial I/O communication (SSPI00, SSPI01, SSPI10, SSPI11) 362
12.6 Operation of clock-synchronous serial communication with slave selection input function 363
12.6.1 Slave transmission 366
12.6.2 Slave reception 376
12.6.3 Slave transmission and reception 383
12.6.4 Calculation of transmission clock frequency 393
12.6.5 Procedure for handling errors during clock-synchronous serial communication with the slave selection input function 394
12.7 Operation of UART (UART0~UART1) communication 395
12.7.1 UART transmission 396
12.7.2 UART reception 404
12.7.3 Calculation of baud rate 411
12.7.4 Handling steps when an error occurs during UART (UART0~UART1) communication 415

12.8 Operation of LIN communication 416
12.8.1 LIN transmission 416
12.8.2 LIN reception 419
Chapter 13 Serial Interface SPI 424
13.1 Function of serial interface SPI 424
13.2 Structure of serial interface SPI 424
13.3 Registers for controlling serial interface SPI 425
13.3.1 Peripheral enable register 0 (PER0) 425
13.3.2 SPI operation mode register (SPIM) 426
13.3.3 SPI clock selection register (SPIC) 427
13.3.4 Transmit buffer register (SDRO) 428
13.3.5 Receive buffer register (SDRI) 428
13.3.6 Registers controlling port functions of SPI pins 428
13.4 Operation of serial interface SPI 429
13.4.1 Master tramission and reception 429
13.4.2 Master reception 432
13.4.3 Slave transmission and reception 435
13.4.4 Slave reception 438
Chapter 14 Serial Interface IICA 441
14.1 Function of serial interface IICA 441
14.2 Structure of serial interface IICA 444
14.3 Registers for controlling serial interface IICA 447
14.3.1 Peripheral enable register 0 (PER0) 448
14.3.2 IICA control register n0 (IICCTLn0) 448
14.3.3 IICA status register n(IICSn) 453
14.3.4 IICA flag register n(IICFn) 456
14.3.5 IICA control register n1(IICCTLn1) 458
14.3.6 IICA low-level width setting register n(IICWLn) 460
14.3.7 IICA high level width setting register n(IICWHn) 460
14.3.8 Registers controlling port functions of IICA pins 460
14.4 Function of I^2C-bus mode 461
14.4.1 Pin structure 461
14.4.2 Setting transfer clock via IICWLn and IICWHn registers 462
14.5 Definition and control method of I^2C-bus 464
14.5.1 Start condition 465
14.5.2 Address 466
14.5.3 Transfer direction specification 466
14.5.4 Acknowledge (ACK) 467
14.5.5 Stop condition 468
14.5.6 Wait 469
14.5.7 Method of releasing wait state 471
14.5.8 Generation timing and waiting control of interrupt requests (INTIICAn) 472
14.5.9 Detection method for address matching 474
14.5.10 Error detection 474
14.5.11 Extension code 475
14.5.12 Arbitration 476

14.5.13 Wake-up function 478
14.5.14 Communication reservation 481
14.5.15 Cautions 485
14.5.16 Communication operation 486
14.5.17 Timing of I²C interrupt request (INTIICAn) generation 495
14.6 Timing diagram 515
Chapter 15 Linkage Controller EVENTC 530
15.1 Function of EVENTC 530
15.2 Structure of EVENTC 530
15.3 Control registers 531
15.3.1 Output target selection register n(ELSELRn) (n=00~12) 532
15.4 Operation of EVENTC 534
Chapter 16 Interrupt Function 535
16.1 Types of interrupt functions 535
16.2 Interrupt sources and structures 535
16.3 Registers for controlling interrupt function 540
16.3.1 Interrupt request flag register (IF00~IF31) 540
16.3.2 Interrupt mask flag register (MK00~MK31) 541
16.3.3 External interrupt rising edge enable register (EGP0), External interrupt falling edge enable register (EGN0) 543
16.4 Operation of interrupt handling 544
16.4.1 Maskable interrupt request acknowledgment 544
16.4.2 Non-maskable interrupt request acknowledgment 544
Chapter 17 Key Interrupt Function 545
17.1 Function of key interrupt 545
17.2 Structure of key interrupt 546
17.3 Registers for controlling key interrupts 547
17.3.1 Key return mode register (KRM) 547
17.3.2 Port mode register (PMx) 547
Chapter 18 Standby Function 548
18.1 Standby function 548
18.2 Sleep mode 549
18.2.1 Setting of sleep mode 549
18.2.2 Sleep mode release 552
18.3 Deep sleep mode 553
18.3.1 Setting of deep sleep mode 553
18.3.2 Deep sleep mode release 555
18.4 Deep sleep mode with partial power down 556
18.4.1 Setting of deep sleep mode with partial power down 556
18.4.2 Release deep sleep mode with partial power down 559
Chapter 19 Reset Function 560
19.1 Registers for confirming the reset source 565
19.1.1 Reset control flag register (RESF) 565
Chapter 20 Power-On Reset Circuit 567
20.1 Function of power-on reset circuit 567

20.2 Structure of power-on reset circuit 568
20.3 Operation of power-on reset circuit 569
Chapter 21 Voltage Detection Circuit 572
21.1 Function of voltage detection circuit 572
21.2 Structure of voltage detection circuit 573
21.3 Registers for controlling voltage detection circuit 573
21.3.1 Voltage detection register (LVIM) 574
21.3.2 Voltage detection level register (LVIS) 575
21.4 Operation of voltage detection circuit 579
21.4.1 When used as reset mode 579
21.4.2 When used as interrupt mode 580
21.4.3 When used as interrupt & reset mode 582
21.5 Cautions for voltage detection circuits 589
Chapter 22 Safety Function 591
22.1 Summary of safety functions 591
22.2 Registers used for safety functions 591
22.3 Operation of safety functions 592
22.3.1 Flash CRC operation function (high-speed CRC) 592
22.3.2 CRC operation function (general-purpose CRC) 595
22.3.3 SFR guard function 599
22.3.4 Frequency detection function 600
22.3.5 A/D test function 601
22.3.6 Digital output signal level detection function for input/output pins 602
22.3.7 Product unique identification register 603
Chapter 23 Temperature sensor 604
23.1 Function of temperature sensor 604
23.2 Temperature sensor register 604
23.2.1 Temperature sensor calibration data register TSN25 604
23.3 Instructions for using temperature sensor 605
Chapter 24 Option Bytes 606
24.1 Function of option bytes 606
24.1.1 User option bytes (000C0H~000C2H) 606
24.1.2 Flash data protection option bytes (000C3H, 500004H) 607
24.2 Format of user option bytes 608
24.3 Format of flash data protection option bytes 614
Chapter 25 FLASH Control 615
25.1 FLASH control function description 615
25.2 Structure of FLASH memory 615
25.3 Registers for controlling FLASH 616
25.3.1 Flash write protection register (FLPROT) 616
25.3.2 FLASH operation control register (FLOPMD1, FLOPMD2) 617
25.3.3 Flash erase control register (FLERMD) 617
25.3.4 Flash status register (FLSTS) 618
25.3.5 Flash chip erase time control register (FLCERCNT) 618
25.3.6 Flash sector erase time control register (FLSERCNT) 619

25.3.7 Flash write time control register (FLPROCNT).....620
25.4 How to operate FLASH.....621
25.4.1 Sector erase.....621
25.4.2 Chip erase.....622
25.4.3 Word program.....622
25.5 Flash read.....622
25.6 Cautions for FLASH operation.....622
Appendix Revision History..... 623

Chapter 1 CPU

1.1 Overview

This chapter provides a brief introduction to the features and debugging features of the ARM Cortex-M0+ core. For details, please refer to the ARM related documentation.

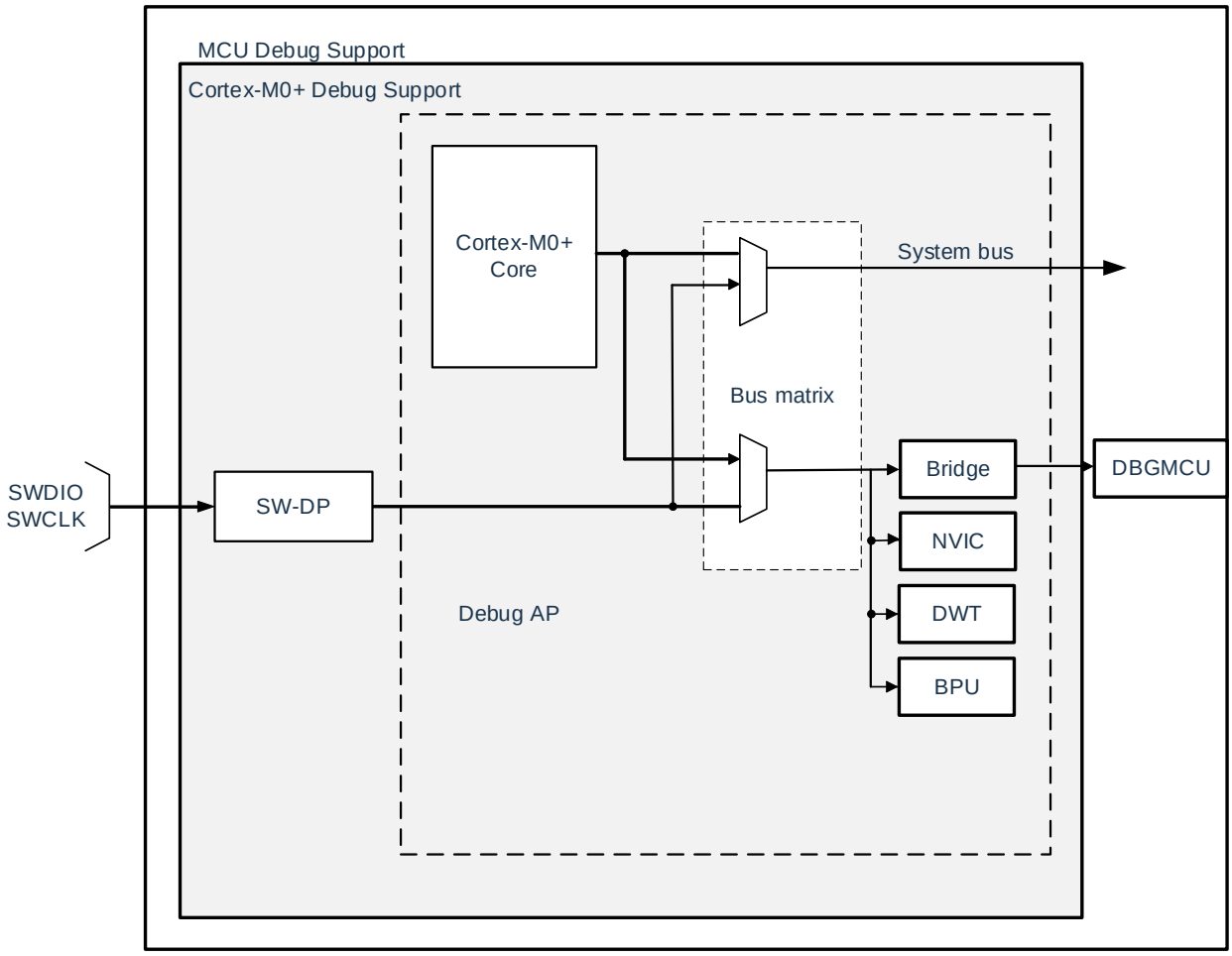
1.2 Cortex-M0+ core features

- ARM Cortex-M0+ processor is a 32-bit RISC core with a 2-stage pipeline that supports privileged and user modes
- 32-cycle hardware multiplier
- Nested vector interrupt controller (NVIC)
  - 1 non-maskable interrupt (NMI)
  - Support 32 maskable interrupt requests (IRQ)
  - 4 interrupt priority levels
- System Timer (SysTick) is a 24-bit countdown timer with a choice of F_{CLK} or F_{IL} count clock
- Vector table offset register (VTOR)
  - The software can write VTOR to relocate the vector table start address to a different location.
  - The default value of this register is 0x0000_0000, the lower 8 bits are ignored for writing and zero for reading, which means the offset is 256 bytes aligned.

1.3 Debugging features

- 2-wire SWD debug interface
- Support for suspending, resuming and single-step execution of programs
- Access to the processor's core registers and special function registers
- 4 hardware breakpoints (BPU)
- Unlimited software breakpoints (BKPT instruction)
- 2 data observation points (DWT)
- Accessing memory while the core is executing

Figure 1-1: Debug block diagram of Cortex-M0+



Notice: SWD does not work in deep sleep mode, please do debug operation in active and sleep mode.

1.4 SWD interface pins

The 2 GPIOs of this product can be used as SWD interface pins, which exist in all packages.

Table 1-1: SWD debug port pins

SWD port name	Debugging functions	Pin assignment
SWCLK	Serial clock	P21/P37
SWDIO	Serial data input/output	P20/P36

When the SWD function is not used, SWD can be disabled by setting the debug stop control register (DBGSTOPCR).

Address: 0x4001B004 After reset: 00H R/W

Bit No.	31	30	29	28	27	26	25	24
DBGSTOPCR	-	-	-	-	-	-	-	SWDIS
Default Value	0	0	0	0	0	0	0	0

Bit No.	23	22	21	20	19	18	17	16
DBGSTOPCR	-	-	-	-	-	-	-	-
Default Value	0	0	0	0	0	0	0	0

Bit No.	15	14	13	12	11	10	9	8
DBGSTOPCR	-	-	-	-	-	-	-	-
Default Value	0	0	0	0	0	0	0	0

Bit No.	7	6	5	4	3	2	1	0
DBGSTOPCR	-	-	-	-	-	-	FRZEN1	FRZEN0
Default Value	0	0	0	0	0	0	0	0

SWDIS	SWD debug interface status
0	Enable the SWD debug interface. P20/P36 cannot be used as GPIO (because ENO and DOUT of this IOBUF are controlled by the debugger at this time).
1	Disable the SWD debug interface. P20/P36 can be used as GPIO.

FRZEN0	When the debugger is connected and the CPU is in debug state (HALTED=1), the timer system peripheral module acts/stops Note1.
0	Peripheral acts
1	Peripheral stops

FRZEN1	When the debugger is connected and the CPU is in debug state (HALTED=1), the peripheral module of the communication system acts/stops Note2.
0	Peripheral acts
1	Peripheral stops

Note 1: The timer system peripheral module of this product includes: general-purpose timer unit Timer4

Note 2: The communication system peripheral module of this product includes: serial communication unit, serial IICA.

1.5 ARM reference document

The built-in debugging feature in the Cortex®-M0+ core is part of the ARM® CoreSight design suite. For documentation, refer to:

- Cortex®-M0+ Technical Reference Manual (TRM)
- ARM® Debug Interface V5
- ARM® CoreSight Design Suite Version r1p1 Technical Reference Manual
- ARM® CoreSight™ MTB-M0+ Technical Reference Manual

Chapter 2 Port Function

2.1 Port function

See the datasheet for each product series.

2.2 Port multiplexing function

See the datasheet for each product series.

2.3 Registers for controlling port functions

The port functions are controlled through the following registers.

(1) Port mode register (PMxx)
(2) Port register (Pxx)
(3) Pull-up resistor selection register (PUxx)
(4) Pull-down resistor selection register (PDxx)
(5) Port output mode register (POMx)
(6) Port mode control register (PMCxx)
(7) Port set control register (PSETxx)
(8) Port clear control register (PCLRxx)
(9) Port status readback register (PREADxx)
(10) Port output multiplexing configuration register (PxxCFG)

Table 2-1: PMxx, Pxx, PSETxx, PCLRxx, PUxx, PDxx, POMxx, PMCxx registers and their bits assigned to each product

Ports		Bit name									24 pins	20 pins
		PMxx register	Pxx register	PSETxx register	PCLRxx register	PUxx register	PDxx register	POMxx register	PMCxx register	PxxCFG register		
Port 0	0	PM00	P00	PSET00	PCLR00	PU00	—	POM00	PMC00	P00CFG	○	○
	1	PM01	P01	PSET01	PCLR01	PU01	PD01	POM01	PMC01	P01CFG	○	○
	2	PM02	P02	PSET02	PCLR02	PU02	PD02	POM02	PMC02	P02CFG	○	○
Port 1	0	PM10	P10	PSET10	PCLR10	PU10	PD10	POM10	PMC10	P10CFG	○	—
	1	PM11	P11	PSET11	PCLR11	PU11	PD11	POM11	PMC11	P11CFG	○	○
	2	PM12	P12	PSET12	PCLR12	—	—	POM12	PMC12	P12CFG	○	○
	3	PM13	P13	PSET13	PCLR13	—	—	POM13	PMC13	P13CFG	○	○
Port 2	0	PM20	P20	PSET20	PCLR20	PU20	PD20	POM20	PMC20	P20CFG	○	○
	1	PM21	P21	PSET21	PCLR21	PU21	PD21	POM21	PMC21	P21CFG	○	○
	2	PM22	P22	PSET22	PCLR22	PU22	PD22	POM22	PMC22	P22CFG	○	—
	3	PM23	P23	PSET23	PCLR23	PU23	PD23	POM23	PMC23	P23CFG	○	—
	4	PM24	P24	PSET24	PCLR24	PU24	PD24	POM24	PMC24	P24CFG	○	○
	5	PM25	P25	PSET25	PCLR25	PU25	PD25	POM25	PMC25	P25CFG	○	○
	6	PM26	P26	PSET26	PCLR26	PU26	PD26	POM26	PMC26	P26CFG	○	○
Port 3	0	PM30	P30	PSET30	PCLR30	PU30	PD30	POM30	PMC30	P30CFG	○	○
	1	PM31	P31	PSET31	PCLR31	PU31	PD31	POM31	PMC31	P31CFG	○	○
	2	PM32	P32	PSET32	PCLR32	PU32	PD32	POM32	PMC32	P32CFG	○	○
	3	PM33	P33	PSET33	PCLR33	PU33	PD33	POM33	PMC33	P33CFG	○	—
	4	PM34	P34	PSET34	PCLR34	PU34	PD34	POM34	PMC34	P34CFG	○	—
	5	PM35	P35	PSET35	PCLR35	PU35	PD35	POM35	PMC35	P35CFG	○	○
	6	PM36	P36	PSET36	PCLR36	PU36	PD36	POM36	PMC36	P36CFG	○	○
	7	PM37	P37	PSET37	PCLR37	PU37	PD37	POM37	PMC37	P37CFG	○	○

Table 2-2: Digital mapping of pin functions

Function name	Input/Output	PxxCFG	PMCxx	PMxx	POMxx	Pxx	Remark
Analog function	Input/Output	6'h00	1	x	x	x	
GPIO	Input	6'h00	0	1	0	x	
	Output		0	0	0	x	
	Open drain		0	0	1	x	
INTP0	Input	6'h02	0	1	0	x	
INTP1	Input	6'h03	0	1	0	x	
INTP2	Input	6'h04	0	1	0	x	
INTP3	Input	6'h05	0	1	0	x	
TI00	Input	6'h06	0	1	0	x	
TI01	Input	6'h07	0	1	0	x	
TI02	Input	6'h08	0	1	0	x	
TI03	Input	6'h09	0	1	0	x	
TI10	Input	6'h0a	0	1	0	x	
TI11	Input	6'h0b	0	1	0	x	
TI12	Input	6'h0c	0	1	0	x	
TI13	Input	6'h0d	0	1	0	x	
TO00	Output	6'h0e	0	0	0	x	
TO01	Output	6'h0f	0	0	0	x	
TO02	Output	6'h10	0	0	0	x	
TO03	Output	6'h11	0	0	0	x	
TO10	Output	6'h12	0	0	0	x	
TO11	Output	6'h13	0	0	0	x	
TO12	Output	6'h14	0	0	0	x	
TO13	Output	6'h15	0	0	0	x	
SCLA0	Input/Output	6'h16	0	0	1	x	
SDAA0	Input/Output	6'h17	0	0	1	x	
CLKBUZ0	Output	6'h18	0	0	0	x	
CLKBUZ1	Output	6'h19	0	0	0	x	
RTC1HZ	Output	6'h1a	0	0	0	x	
Reserved		6'h1b	0	x	x	x	Disable access
SPI_SSI	Input	6'h1c	0	1	0	x	
SPI_MOSI	Output/Input	6'h1d	0	0	0	x	
SPI_MISO	Input/Output	6'h1e	0	0	0	x	
SPI_CLKOI	Output/Input	6'h1f	0	0	0	x	
Reserved		6'h20	0	x	x	x	Disable access
Reserved		6'h21	0	x	x	x	Disable access
Reserved		6'h22	0	x	x	x	Disable access
Reserved		6'h23	0	x	x	x	Disable access

Reserved		6'h24	0	x	x	x	Disable access
Reserved		6'h25	0	x	x	x	Disable access
SAU0_SS	Input	6'h26	0	1	0	x	
SAU1_SS	Input	6'h27	0	1	0	x	
SCLKOI00	Output/Input	6'h28	0	0	0	x	
SCLKOI01	Output/Input	6'h29	0	0	0	x	
SCLKOI10	Output/Input	6'h2a	0	0	0	x	
SCLKOI11	Output/Input	6'h2b	0	0	0	x	
SDI00/RxD0	Input	6'h2c	0	1	0	x	
SDI01	Input	6'h2d	0	1	0	x	
SDI10/RxD1	Input	6'h2e	0	1	0	x	
SDI11	Input	6'h2f	0	1	0	x	
SDO00/TxD0	Output	6'h30	0	0	0	x	
SDO01	Output	6'h31	0	0	0	x	
SDO10/TxD1	Output	6'h32	0	0	0	x	
SDO11	Output	6'h33	0	0	0	x	
Reserved	Output	6'h34	0	0	0	x	Disable access
Reserved	Output	6'h35	0	0	0	x	Disable access
Reserved	Output	6'h36	0	0	0	x	Disable access
Reserved	Output	6'h37	0	0	0	x	Disable access
EPWMO00	Output	6'h38	0	0	0	x	
EPWMO01	Output	6'h39	0	0	0	x	
EPWMO02	Output	6'h3a	0	0	0	x	
EPWMO03	Output	6'h3b	0	0	0	x	
EPWMO04	Output	6'h3c	0	0	0	x	
EPWMO05	Output	6'h3d	0	0	0	x	
EPWMO06	Output	6'h3e	0	0	0	x	
EPWMO07	Output	6'h3f	0	0	0	x	

Note: Since this product supports arbitrary GPIO mapping for all digital functions, the corresponding functions in the above table and the value of PxxCFG need to correspond one to one.

Configuration description:

(1) When using the port's multiplexing function, the port must be configured in digital mode (PMCxx=0).

(2) When using the port's multiplexing function, the port must be configured in output mode (push-pull or open-drain) (PMxx=0).

(3) When using the GPIO function or multiplexing function of P01, P02 ports, make sure that their X1 oscillation mode and external clock input mode are not turned on. Refer to section 4.3.

(4) When using the GPIO function or multiplexing function of P12, P13 ports, make sure that their XT1 oscillation mode and external clock input mode are not turned on. Refer to section 4.3.

2.3.1 Port mode register (PMxx)

When a port is used as a digital channel, this is the register that sets its input/output in bits. After a reset signal is generated, all ports default to the input state. When using a port pin as a multiplexing function, it must be set according to the register settings for the multiplexing function.

Register address = base address + offset address; the base address of PM register is 0x40040000, and the offset address is shown in the table below.

Table 2-3: Format of port mode register

Symbol	7	6	5	4	3	2	1	0	Offset address	After reset	R/W
PM0	1	1	1	1	1	PM02	PM01	PM00	0x020	FFH	R/W
PM1	1	1	1	1	PM13	PM12	PM11	PM10	0x021	FFH	R/W
PM2	1	PM26	PM25	PM24	PM23	PM22	PM21	PM20	0x022	FFH	R/W
PM3	PM37	PM36	PM35	PM34	PM33	PM32	PM31	PM30	0x023	FFH	R/W

PMmn	Selection of input/output mode for Pmn pin (m=0~3, n=0~7)
0	Output mode (used as output port (output buffer ON))
1	Input mode (used as input port (output buffer OFF))

2.3.2 Port register (Pxx)

This is the register that sets the value of the port's output latch in bits. Reading this register in input mode gives the pin level, while reading it in output mode gives the value of the port's output latch. After a reset signal is generated, the value of these registers changes to "00H".

Register address = base address + offset address; the base address of Port register is 0x40040000, and the offset address is shown in the table below.

Table 2-4: Format of the port register

Symbol	7	6	5	4	3	2	1	0	Offset address	After reset	R/W
P0	0	0	0	0	0	P02	P01	P00	0x000	00H	R/W
P1	0	0	0	0	P13	P12	P11	P10	0x001	00H	R/W
P2	0	P26	P25	P24	P23	P22	P21	P20	0x002	00H	R/W
P3	P37	P36	P35	P34	P33	P32	P31	P30	0x003	00H	R/W

Pmn	m=0~3, n=0~7	
	Control of output data (output mode)	Reading of input data (input mode)
0	The port outputs "0".	Port input low level.
1	The port outputs "1".	Port input high level.

2.3.3 Port set control register (PSETxx)

These are registers that reset the port output latch in bit units. The value of these registers changes to "00H" after a reset signal is generated.

Register address = base address + offset address; the base address of PSET register is 0x40040000, and the offset address is shown in the table below.

Table 2-5: Format of port set control register

Symbol	7	6	5	4	3	2	1	0	Offset address	After reset	R/W
PSET0	0	0	0	0	0	PSET02	PSET01	PSET00	0x070	00H	W
PSET1	0	0	0	0	PSET13	PSET12	PSET11	PSET10	0x071	00H	W
PSET2	0	PSET26	PSET25	PSET24	PSET23	PSET22	PSET21	PSET20	0x072	00H	W
PSET3	PSET37	PSET36	PSET35	PSET34	PSET33	PSET32	PSET31	PSET30	0x073	00H	W

PSETmn	Setting control of Pmn pin (m=0~3, n=0~7)
0	No operation
1	Set the corresponding Pmn to 1

2.3.4 Port clear control register (PCLRxx)

This is the register to set the port output latch in bit units. After a reset signal is generated, the value of these registers becomes “00H”.

Register address = base address + offset address; the base address of PCLR register is 0x40040000, and the offset address is shown in the table below.

Table 2-6: Format of port clear control register

Symbol	7	6	5	4	3	2	1	0	Offset address	After reset	R/W
PCLR0	0	0	0	0	0	PCLR02	PCLR01	PCLR00	0x080	00H	W
PCLR1	0	0	0	0	PCLR13	PCLR12	PCLR11	PCLR10	0x081	00H	W
PCLR2	0	PCLR26	PCLR25	PCLR24	PCLR23	PCLR22	PCLR21	PCLR20	0x082	00H	W
PCLR3	PCLR37	PCLR36	PCLR35	PCLR34	PCLR33	PCLR32	PCLR31	PCLR30	0x083	00H	W

PCLRmn	Clear control of Pmn pin (m=0~3, n=0~7)
0	No operation
1	Set the corresponding Pmn to 0

2.3.5 Pull-up resistor selection register (PUxx)

This is the internal pull-up resistor selection register. For pins configured for digital functions, internal pull-up resistors can be used in bits through the pull-up resistor selection register; pins set for analog functions do not connect internal pull-up resistors.

After generating the reset signal, the pull-up function of the four ports P00, P20, P21, P36, and P37 is turned on by default (the reset value of PU00, PU20, PU21, PU36, and P37 is "1"), and the pull-up function of the other ports is not turned on by default.

Register address = base address + offset address; the base address of PU register is 0x40040000, and the offset address is shown in the table below.

Table 2-7: Format of pull-up resistor selection register

Symbol	7	6	5	4	3	2	1	0	Offset address	After reset	R/W
PU0	0	0	0	0	0	PU02	PU01	PU00	0x030	01H	R/W
PU1	0	0	0	0	0	0	PU11	PU10	0x031	00H	R/W
PU2	0	PU26	PU25	PU24	PU23	PU22	PU21	PU20	0x032	03H	R/W
PU3	PU37	PU36	PU35	PU34	PU33	PU32	PU31	PU30	0x033	C0H	R/W

PUmn	Selection of internal pull-up resistor for Pmn pin (m=0~3, n=0~7)
0	No internal pull-up resistor is connected.
1	Connect the internal pull-up resistor.

Notice: Ports P12 and P13 of this product do not support pull-up function, and must be connected to external pull-up resistors when used.

2.3.6 Pull-down resistor selection register (PDxx)

This is the internal pull-down resistor selection register. For pins configured for digital functions, internal pull-down resistors can be used in bits through the pull-down resistor selection register; pins set for analog functions do not connect internal pull-down resistors.

After a reset signal is generated, the value of the register changes to "00H".

Register address = base address + offset address; the base address of PD register is 0x40040000, and the offset address is shown in the table below.

Table 2-8: Format of pull-down resistor selection register

Symbol	7	6	5	4	3	2	1	0	Offset address	After reset	R/W
PD0	0	0	0	0	0	PD02	PD01	0	0x040	00H	R/W
PD1	0	0	0	0	0	0	PD11	PD10	0x041	00H	R/W
PD2	0	PD26	PD25	PD24	PD23	PD22	PD21	PD20	0x042	00H	R/W
PD3	PD37	PD36	PD35	PD34	PD33	PD32	PD31	PD30	0x043	00H	R/W

PDmn	Selection of internal pull-down resistor for Pmn pin (m=0~3, n=0~7)
0	No internal pull-down resistor is connected.
1	Connect the internal pull-down resistor.

Notice: Ports P00, P12, and P13 of this product do not support the pull-down function and must be used with external pull-down resistors.

2.3.7 Port output mode register (POMxx)

This is a register that sets the output mode in bits. The pin selects the N-channel open drain output mode when communicating between devices.

After a reset signal is generated, the value of the register changes to "00H".

Register address = base address + offset address; the base address of POM register is 0x40040000, and the offset address is shown in the table below.

Table 2-9: Format of port output mode register

Symbol	7	6	5	4	3	2	1	0	Offset address	After reset	R/W
POM0	0	0	0	0	0	POM02	POM01	POM00	0x050	00H	R/W
POM1	0	0	0	0	POM13	POM12	POM11	POM10	0x051	00H	R/W
POM2	0	POM26	POM25	POM24	POM23	POM22	POM21	POM20	0x052	00H	R/W
POM3	POM37	POM36	POM35	POM34	POM33	POM32	POM31	POM30	0x053	00H	R/W

POMmn	Selection of output mode for Pmn pin (m=0~3, n=0~7)
0	Typical output mode
1	N-channel open-drain output mode

2.3.8 Port mode control register (PMCxx)

The PMC register sets the port in bits to be used as a digital input/output or as an analog channel.

After the reset signal is generated, P00, P01, P02, P20, P21, P36, P37 are used as digital channels by default (PMC00, PMC01, PMC02, PMC20, PMC21, PMC36, PMC37 reset values are "0"), and the other ports are used as analog channel by default.

Register address = base address + offset address; the base address of PMC register is 0x40040000, and the offset address is shown in the table below.

Table 2-10: Format of port mode control register

Symbol	7	6	5	4	3	2	1	0	Offset address	After reset	R/W
PMC0	1	1	1	1	1	PMC02	PMC01	PMC00	0x060	F8H	R/W
PMC1	1	1	1	1	PMC13	PMC12	PMC11	PMC10	0x061	FFH	R/W
PMC2	1	PMC26	PMC25	PMC24	PMC23	PMC22	PMC21	PMC20	0x062	FCH	R/W
PMC3	PMC37	PMC36	PMC35	PMC34	PMC33	PMC32	PMC31	PMC30	0x063	3FH	R/W

PMCmn	Selection of digital inputs/outputs or analog inputs for Pmn pins (m=0~3, n=0~7)
0	Digital inputs/outputs (multiplexing functions other than analog inputs)
1	Analog inputs

2.3.9 Port readback register (PREADxx)

This is a read-only register that can be read to get the corresponding port level when the port is used as a digital GPIO.

Register address = base address + offset address; the base address of PREAD register is 0x40040000, and the offset address is shown in the table below.

Symbol	7	6	5	4	3	2	1	0	Offset address	Reset value	R/W
PREAD0	--	--	--	--	--	PREAD02	PREAD01	PREAD00	0x90	xxxxH	R
PREAD1	--	--	--	--	PREAD13	PREAD12	PREAD11	PREAD10	0x91	xxxxH	R
PREAD2	--	PREAD26	PREAD25	PREAD24	PREAD23	PREAD22	PREAD21	PREAD20	0x92	xxxxH	R
PREAD3	PREAD37	PREAD36	PREAD35	PREAD34	PREAD33	PREAD32	PREAD31	PREAD30	0x93	xxxxH	R

PREADmn	m=0~3, n=0~7
	Digital output mode/input mode
0	Port is low
1	Port is high

Remark: PREAD is only used for read operation and does not support write operation. Its read value after reset depends on the corresponding port level.

2.3.10 Port multiplexing function configuration register (PxxCFG)

The port multiplexing function configuration register can map the output function of some peripheral modules to any port. If the reset value of the port multiplexing function configuration register is "00H", then the port is GPIO function by default.

Register address = base address + offset address; the base address of PxxCFG register is 0x40040800, and the offset address is shown in the table below.

Base address: 0x40040800

Table 2-11: Format of port multiplexing function configuration register

Symbol	7	6	5	4	3	2	1	0
P0xCFG	--	--	--	--	--	P02CFG	P01CFG	P00CFG
Offset address	--	--	--	--	--	0x04	0x02	0x00
P1xCFG	--	--	--	--	P13CFG	P12CFG	P11CFG	P10CFG
Offset address	--	--	--	--	0x16	0x14	0x12	0x10
P2xCFG	--	P26CFG	P25CFG	P24CFG	P23CFG	P22CFG	P21CFG	P20CFG
Offset address	--	0x2c	0x2a	0x28	0x26	0x24	0x22	0x20
P3xCFG	P37CFG	P36CFG	P35CFG	P34CFG	P33CFG	P32CFG	P31CFG	P30CFG
Offset address	0x3e	0x3c	0x3a	0x38	0x36	0x34	0x32	0x30

Remark: For specific function mapping, refer to Table 2-2: Digital Mapping of Pin Functions.

2.3.11 Description of the special function port RESINB

RESINB is valid by default when the product is powered on, if you need to use this port as GPIO, then you need to turn off the reset function through the registers, and the registers are described as follows.

Base address: 0x40020400 Offset address: 0x0B

Symbol	7	6	5	4	3	2	1	0	Reset value	R/W
RSTM	0	0	0	0	0	0	0	RSTM	00H	R/W

RSTM	External reset function mask on RESINB pin
0	RESINB external reset pin
1	RESINB as GPIO pin

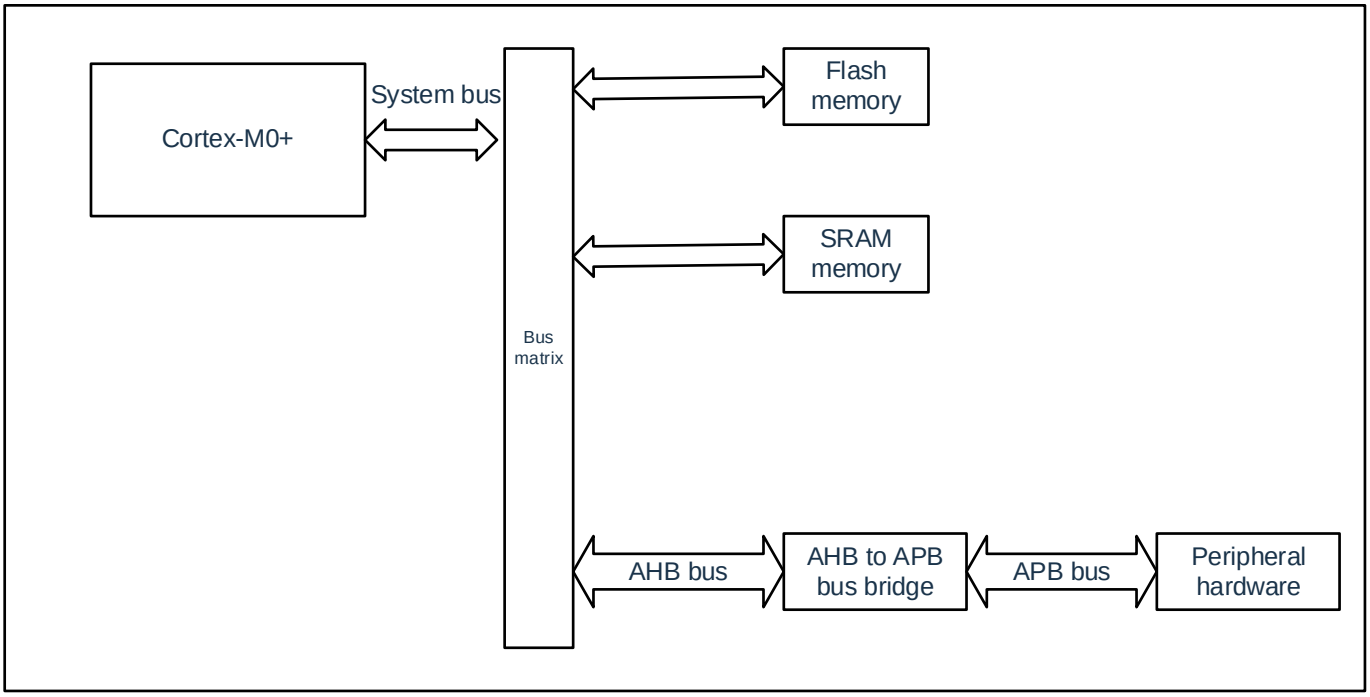
Chapter 3 System Architecture

3.1 Overview

This product system consists of the following components:

- 1 AHB buses Master:
  - Cortex-M0+
- 3 AHB buses Slaves:
  - FLASH memory
  - SRAM memory
  - AHB to APB Bridge, contains all APB interface peripherals

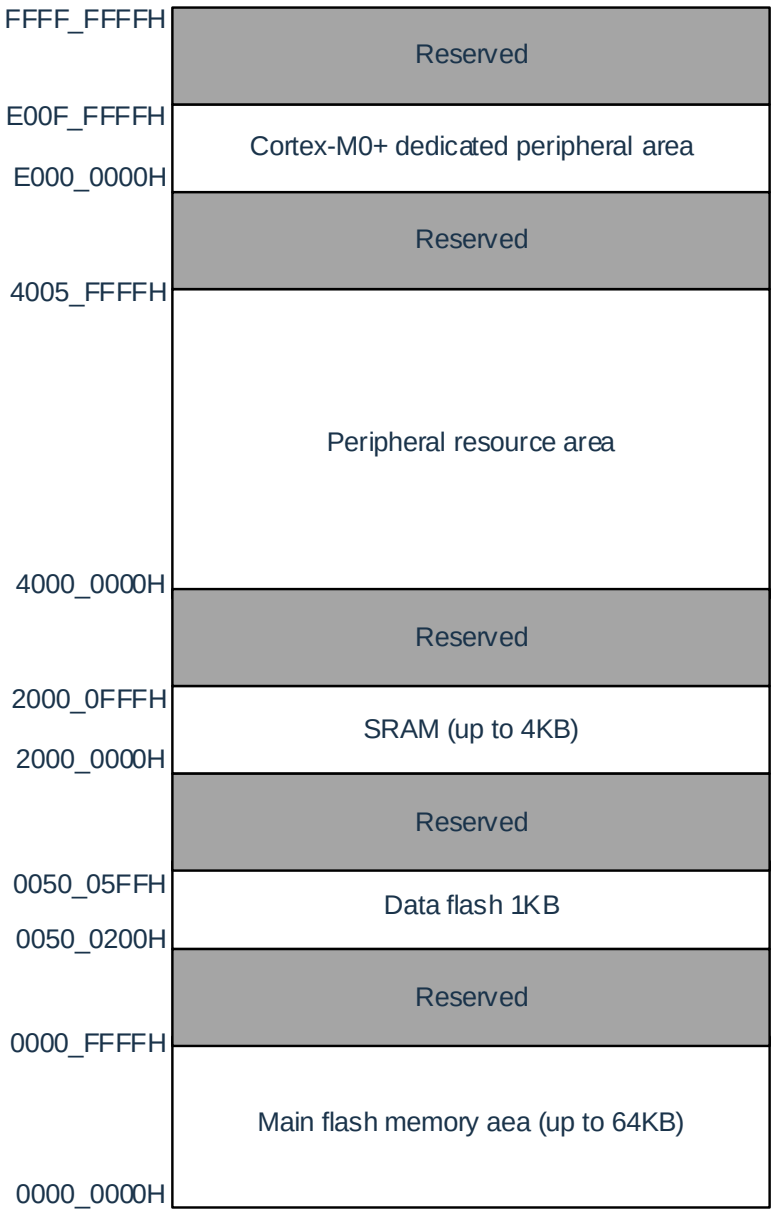
Figure 3-1: Block diagram of system architecture



1) System Bus: This bus connects the system bus (peripheral bus) of the Cortex-M0+ core to the bus matrix.
2) Bus Matrix: The bus matrix coordinates access to other buses on the core system bus.
3) AHB to APB Bridge: The AHB to APB Bridge provides a synchronous connection between the AHB and APB buses. Refer to Figure 3-1 for address mapping of the different peripherals connected to each bridge.

3.2 System address partitioning

Figure 3-2: Map of address area



Peripheral Address Assignment

Table 3-1: Starting address of the peripheral's register group

Start address	Peripheral	Remark
0x4000_0000 - 0x4000_4FFF	Reserved	
0x4000_5000 - 0x4000_5FFF	Reserved	
0x4000_6000 - 0x4000_6FFF	Interrupt control	
0x4000_7000 - 0x4001_8FFF	Reserved	
0x4001_9000 - 0x4001_9FFF	Reserved	
0x4001_A000 - 0x4001_FFFF	Reserved	
0x4002_0000 - 0x4002_03FF	FLASH control	
0x4002_0400 - 0x4002_0FFF	FLASH control	
0x4002_1000 - 0x4002_1001	Watchdog timer	
0x4002_1002 - 0x4002_1800	Reserved	
0x4002_1800 - 0x4002_1BFF	High-speed CRC	See Chapter 22 Safety Function
0x4002_1C00 - 0x4002_1FFF	Clock control	
0x4002_2000 - 0x4003_FFFF	Reserved	
0x4004_0000 - 0x4004_0FFF	GPIO	
0x4004_1100 - 0x4004_19FF	Serial communication unit	
0x4004_1A00 - 0x4004_1CFF	Serial interface IICA	
0x4004_1D00 - 0x4004_1FFF	Timer array 0	
0x4004_2000 - 0x4004_21FF	Timer array 1	
0x4004_2200 - 0x4004_23FF	Reserved	
0x4004_2400 - 0x4004_27FF	SPI	
0x4004_2800 - 0x4004_31FF	Reserved	
0x4004_3200 - 0x4004_32FF	General-purpose CRC	
0x4004_3300 - 0x4004_33FF	General-purpose CRC	
0x4004_3400 - 0x4004_37FF	Linkage controller	
0x4004_3C00 - 0x4004_3FFF	Reserved	
0x4004_4000 - 0x4004_43FF	Reserved	
0x4004_4400 - 0x4004_47FF	EPWM	
0x4004_4800 - 0x4004_4EFF	Reserved	
0x4004_4F00 - 0x4004_4FFF	Real time clock	
0x4004_5000 - 0x4004_53FF	AD converter	
0x4004_5400 - 0x4004_5AFF	Reserved	
0x4004_5B00 - 0x4004_5BFF	External interrupt control	
0x4008_0000 - 0x4008_01FF	Reserved	
0x4008_0200 - 0xDFFF_FFFF	Reserved	

Chapter 4 Clock Generation Circuit

The presence or absence of the resonator connection pin/external clock input pin for the main system clock and the resonator connection pin/external clock input pin for the sub-system clock varies by product.

4.1 Function of clock generation circuit

The clock generation circuit is a circuit that generates a clock supplied to the CPU and peripheral hardware. There are the following 3 types of system clock and clock oscillation circuits.

(1) Main system clock

① X1 oscillation circuit

The clock of F_X=4~16MHz can be oscillated by connecting resonators to pins X1 and X2, and the oscillation can be stopped by entering deep sleep mode or by setting the MSTOP bit (bit 7 of the clock operation status control register (CSC)).

② High-speed on-chip oscillator (high-speed OCO)

The frequency at which to oscillate can be selected from among F_{IH}=64MHz, 48MHz, 32MHz, 24MHz, 16MHz, 12MHz, 8MHz, 6MHz, 4MHz, 3MHz and 2MHz by using the option byte (000C2H). After the reset is released, the CPU must start operation with this high-speed on-chip oscillator clock F_{IH}. Oscillation can be stopped by entering deep sleep mode or by setting the HIOSTOP bit (bit0 of the CSC register). The frequency set by the option byte can be changed by the frequency selection register (HOCODIV) of the high-speed on-chip oscillator. Refer to “Table 4-11: Format of high-speed on-chip oscillator frequency select register (HOCODIV)” for frequency settings.

In addition, an external master system clock (F_{EX}=4~16MHz) can be provided by the EXCLK/X2 pin, and the input of the external master system clock can be disabled by entering deep sleep mode or setting the MSTOP bit.

Switching between the high-speed system clock (X1 clock or external master system clock) and the high-speed on-chip oscillator clock can be performed by setting the MCM0 bit (bit 4 of the system clock control register (CKC)).

(2) Subsystem clock

XT1 oscillation circuit

The clock of F_{XT}=32.768KHz can be oscillated by connecting 32.768KHz resonators to the XT1 and XT2 pins, and the oscillation can be stopped by setting the XTSTOP bit (bit 6 of the clock operation status control register (CSC)).

In addition, an external subsystem clock (F_{EXS}=32.768KHz) can be provided by the EXCLKS/XT2 pin, and the input of the external subsystem clock can be disabled by setting the XTSTOP bit.

(3) Low-speed on-chip oscillator clock (low-speed OCO)

This circuit oscillates a clock of F_{IL} = 15 KHz.

The low-speed on-chip oscillator clock can be used as the system clock.

The low-speed on-chip oscillator oscillates when bit4 (WDTON) of the option byte (000C0H) or bit4 (WUTMMCK0) of the subsystem clock supply mode control register (OSMC) is “1” or when bit0 (SELLOSC) of the subsystem clock selection register (SUBCKSEL) is “1”.

However, the low-speed on-chip oscillator stops oscillating if the deep sleep mode or sleep mode is entered when the WDTON bit is "1" and the WUTMMCK0 bit, the SELLOSC bit is "0" and bit0 (WDSTBYON) of the option byte (000C0H) is "0".

Notice: The low-speed on-chip oscillator clock (F_{IL}) can be selected as the count clock for the real-time clock only when the fixed-cycle interrupt function is used.

Remark: F_X: X1 clock oscillation frequency

F_{HOCO}: High-speed on-chip oscillator clock 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

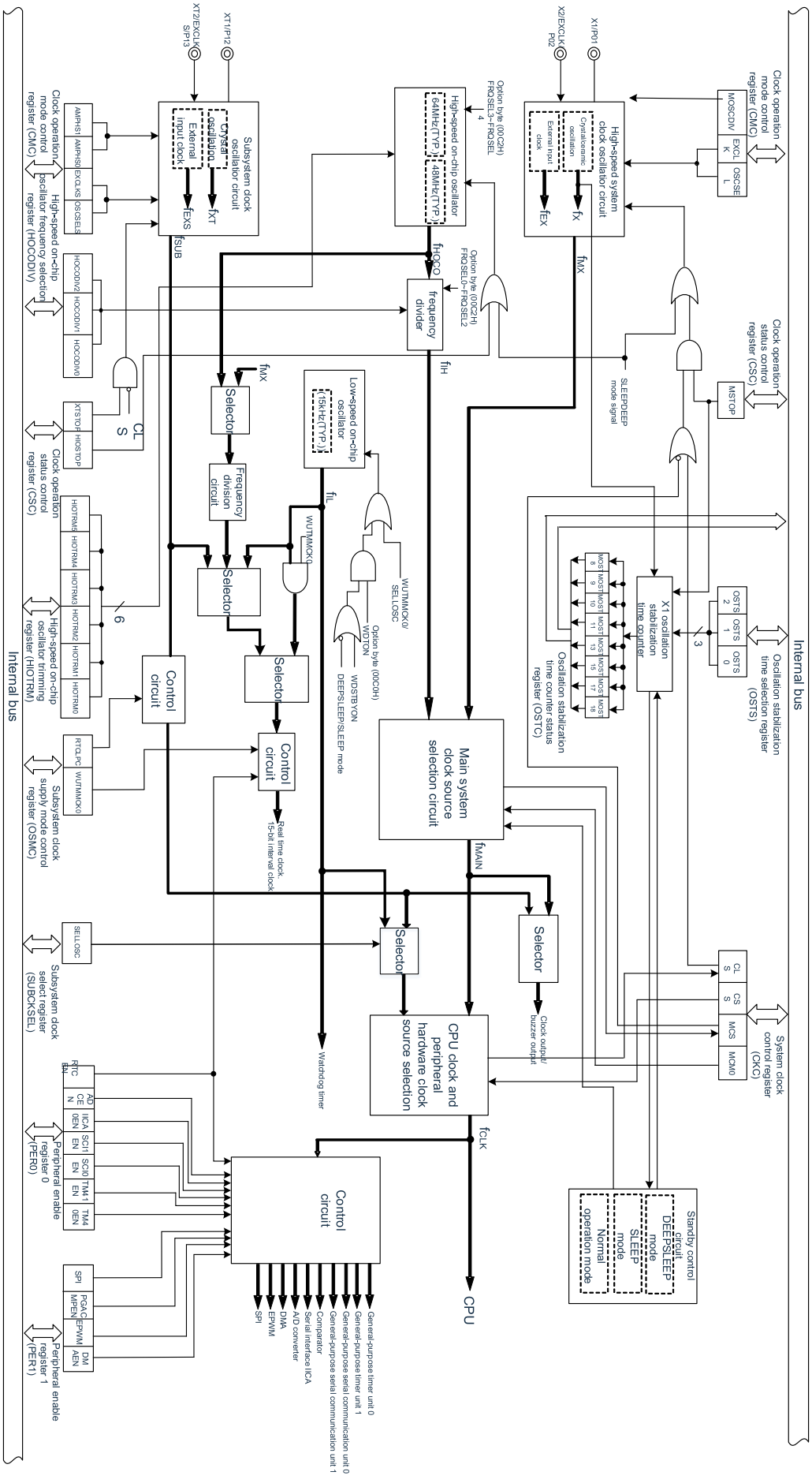
4.2 Configuration of clock generation circuit

The clock generation circuit includes the following hardware.

Table 4-1: Configuration of clock generation circuit

Item	Configuration
Control registers	Clock operation mode control register (CMC) System clock control register (CKC) Clock operation status control register (CSC) Oscillation stabilization time counter status register (OSTC) Oscillation stabilization time selection register (OSTS) Peripheral enable registers 0, 1 (PER0, PER1) Subsystem clock supply mode control register (OSMC) High-speed on-chip oscillator frequency selection register (HOCODIV) High-speed on-chip oscillator trim register (HIOTRM) Subsystem clock selection register (SUBCKSEL)
Oscillatior circuits	X1 oscillation circuit XT1 oscillation circuit High-speed on-chip oscillator Low-speed on-chip oscillator

Figure 4-1: Block diagram of clock generation circuit



Remark: F_X: X1 clock oscillation frequency

F_{HOCO}: High-speed on-chip oscillator clock frequency

F_{IH}: High-speed on-chip oscillator clock frequency

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

F_{CLK}: CPU/peripheral hardware clock frequency

F_{IL}: Low-speed on-chip oscillator clock frequency

4.3 Registers for controlling clock generation circuit

The clock generation circuit is controlled through the following registers.

Clock operation mode control register (CMC)

System clock control register (CKC)

Clock operation status control register (CSC)

Oscillation stabilization time counter status register (OSTC)

Oscillation stabilization time select register (OSTS)

Peripheral enable registers 0, 1 (PER0, PER1)

Subsystem clock supply mode control register (OSMC)

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

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

Subsystem clock select register (SUBCKSEL)

Notice: The assigned registers and bits vary from product to product. The initial values must be set for unassigned bits.

4.3.1 Clock operation mode control register (CMC)

This is the register that sets the operation mode of the X1, X2/EXCLK, XT1, and XT2/EXCLKS pins and selects the gain of the oscillation circuit.

The CMC register can be written only once by an 8-bit memory manipulation instruction after the reset is released. Reading this register can be done with an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register changes to "00H".

Table 4-2: Format of clock operation mode control register (CMC)

Address: 40020400H After reset: 00H R/W Note 1

Symbol	7	6	5	4	3	2	1	0
CMC	EXCLK	OSCSEL	EXCLKS Note	OSCSELS Note	0	AMPHS Note	AMPHS0 Note	0

EXCLK	OSCSEL	Operation mode of high-speed system clock pin	X1 pin	X2/EXCLK pin
0	0	Port mode	Input/output port	
0	1	X1 oscillation mode	Connect a crystal or ceramic resonator.	
1	0	Port mode	Input/output port	
1	1	External clock input mode	Input/output port	External clock input

EXCLKS	OSCSELS	Operation mode of subsystem clock pin	XT1 pin	XT2/EXCLKS pin
0	0	Port mode	Input/output port	
0	1	XT1 oscillation mode	Connect a crystal resonator.	
1	0	Port mode	Input/output port	
1	1	External clock input mode	Input/output port	External clock input

AMPHS1	AMPHS0	Selection of oscillation modes for XT1 oscillation circuit
0	0	Low power oscillation (default)
0	1	Normal oscillation
1	0	Low power oscillation
1	1	Disable setting.

Note 1: Bit3 is a write-only bit.

Note 2: The EXCLKS bit, the OSCSELS bit, the AMPHS1 bit and the AMPHS0 bit are initialized only at power-on reset and remain unchanged at other resets.

Note 3: The CMC register can be written only once by an 8-bit memory manipulation instruction after the reset is released. When the CMC register is used at the initial value (“00H”), the CMC register must be set to “00H” after reset is released in order to prevent misoperation in the event of program runaway (the value other than “00H” cannot be recovered if it is written by mistake).

Note 4: The CMC register must be set after the reset is released and before starting X1 or XT1 oscillation by setting the clock operation status control register (CSC).

Note 5: The AMPHS1 bit and AMPHS0 bit must be set after the reset is released and in the state where FIH is selected as the F_{CLK} (the state before switching the F_{CLK} to F_{MX} or F_{SUB}).

Note 6: The oscillation stabilization time of the F_{XT} must be counted by software.

Note 7: The maximum frequency of the system clock is 64MHz, but the maximum frequency of the X1 oscillator circuit is 16MHz.

Note 8: F_X: X1 clock oscillation frequency.

4.3.2 System clock control register (CKC)

This is a register that selects the CPU/peripheral hardware clock and the main system clock.

The CKC register is set by an 8-bit memory manipulation instruction.

After the reset signal is generated, the value of this register changes to "00H".

Table 4-3: Format of system clock control register (CKC)

Address:	40020404H	After reset: 00H		R/W Note1			
Symbol	7	6	5	4	3	2	1	0
CKC	CLS	CSS	MCS	MCM0	0	0	0	0

CLS	CPU/peripheral hardware clock (F_{CLK}) status
0	Main system clock (F_{MAIN})
1	Subsystem clock (F_{SUB})

CSS Note1	CPU/peripheral hardware clock (F_{CLK}) selection
0	Main system clock (F_{MAIN})
1	Subsystem clock (F_{SUB})

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

MCM0 Note1	Operation control of the main system clock (F_{MAIN})
0	The high-speed on-chip oscillator clock (F_{IH}) is selected as the main system clock (F_{MAIN}).
1	The high-speed system clock (F_{MX}) is selected as the main system clock (F_{MAIN}).

Note 1: Bit7 and bit5 are read-only bits, bit0~3 must be set to "0".

Note: It is prohibited to change the value of the MCM0 bit with the CSS bit set to "1".

Remark:

1. Provides CSS bit setting clocks for the CPU and peripheral hardware. If you change the CPU clock, change the peripheral hardware clock at the same time (except for real-time clocks, 15-bit interval timers, clock output/buzzer output, and watchdog timer). Therefore, if you want to change the clock on the CPU/peripheral hardware, you must stop the peripheral functions.
2. If the subsystem clock is used as the peripheral hardware clock, operation of the A/D converter and IICA cannot be guaranteed. For the operating characteristics of the peripheral hardware, refer to the sections and datasheets for each peripheral hardware.
3. F_{HOCO}: High-speed on-chip oscillator clock frequency
   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

4.3.3 Clock operation status control register (CSC)

This is the register that controls the operation of the high-speed system clock, the high-speed on-chip oscillator clock, and the subsystem clock (except the low-speed internal oscillator clock). The CSC register is set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register changes to "C0H".

Table 4-4: Format of the clock operation status control register (CSC)

Address:	40020401H	After reset: C0H	R/W					
Symbol	7	6	5	4	3	2	1	0
CSC	MSTOP	XTSTOP	0	0	0	0	0	HIOSTOP

MSTOP	Operation control of high-speed system clock		
	X1 oscillation mode	External clock input mode	Port mode
0	X1 oscillation circuit runs	The external clock on the EXCLK pin is valid	Input/output ports
1	X1 oscillation circuit stops	The external clock on the EXCLK pin is invalid	

XTSTOP	Operation control of subsystem clock		
	XT1 oscillation mode	External clock input mode	Port mode
0	XT1 oscillation circuit runs	The external clock on the EXCLKS pin is valid	Input/output ports
1	XT1 oscillation circuit stops	The external clock on the EXCLKS pin is invalid	

HIOSTOP	Operation control of high-speed on-chip oscillator clock
0	High-speed on-chip oscillator runs
1	High-speed on-chip oscillator stops

Notice:

1. After reset release, set the clock operation mode control register (CMC) before setting the CSC register.
2. Set the oscillation stabilization time select register (OSTS) before setting the MSTOP bit to 0 after releasing reset. Note that if the OSTS register is being used with its default settings, the OSTS register is not required to be set here.
3. To start X1 oscillation as set by 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 to 0, wait for oscillation of the subsystem clock to stabilize by setting a wait time using software.
5. Do not stop the clock selected for the CPU peripheral hardware clock (F_{CLK}) with the CSC register.

Remark: The setting of the flags of the register to stop clock oscillation (invalidate the external clock input) and the condition before clock oscillation is to be stopped are as Table 4-5.

Table 4-5: Condition before stopping clock oscillation

Clock	Condition before stopping clock (Invalidating external clock input)	Setting of CSC register flags
X1 clock	CPU/peripheral hardware clock runs on a clock other than the high-speed system clock. (CLS=0 and MCS=0, or CLS=1)	MSTOP=1
External main system clock		

XT1 clock External subsystem clock	CPU/peripheral hardware clock runs on a clock other than the subsystem clock. (CLS=0)	XTSTOP=1
High-speed on-chip oscillator clock	CPU/peripheral hardware clock runs on a clock other than the high-speed on-chip oscillator clock. (CLS=0 and MCS=1, or CLS=1)	HIOSTOP=1

4.3.4 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 oscillation while the high-speed on-chip oscillator clock or subsystem clock is being used as the CPU clock.

If the deep sleep mode is entered and then released while the high-speed on-chip oscillator clock is being used as the CPU clock with the X1 clock oscillating.

The OSTC register can be read by an 8-bit memory manipulation instruction.

The value of this register is changed to "00H" by generating the reset signal, entering the deep sleep mode, or when the MSTOP bit (bit 7 of the clock operation status control register (CSC)) is "1".

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

1) When oscillation of the X1 clock starts (EXCLK, OSCSEL = 0, 1 MSTOP = 0)
2) When the deep sleep mode is released

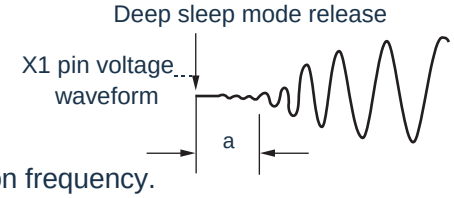
Table 4-6: Format of oscillation stabilization time counter status register (OSTC)

Address:	40020402H	After reset: 00H	R				
Symbol	7	6	5	4	3	2	1	0
OSTC	MOST8	MOST9	MOST10	MOST11	MOST13	MOST15	MOST17	MOST18

MOST8	MOST9	MOST10	MOST11	MOST13	MOST15	MOST17	MOST18		Oscillation stabilization time status	
									F_X=4MHz	F_X=16MHz
0	0	0	0	0	0	0	0	Max. $2^8/F_X$	Max. 64us	Max. 16us
1	0	0	0	0	0	0	0	Min. $2^8/F_X$	Min. 64us	Min. 16us
1	1	0	0	0	0	0	0	Min. $2^9/F_X$	Min. 128us	Min. 32us
1	1	1	0	0	0	0	0	Min. $2^{10}/F_X$	Min. 256us	Min. 64us
1	1	1	1	0	0	0	0	Min. $2^{11}/F_X$	Min. 512us	Min. 128us
1	1	1	1	1	0	0	0	Min. $2^{13}/F_X$	Min. 2.048ms	Min. 512us
1	1	1	1	1	1	0	0	Min. $2^{15}/F_X$	Min. 8.192ms	Min. 2.048ms
1	1	1	1	1	1	1	0	Min. $2^{17}/F_X$	Min. 32.768ms	Min. 8.192ms
1	1	1	1	1	1	1	1	Min. $2^{18}/F_X$	Min. 65.536ms	Min. 16.384ms

Remark:

1. After the above time has elapsed, the bits are set to 1 in order 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 the value greater than the count value which is to be checked by the OSTC register.
   1) If the X1 clock starts oscillation while the high-speed on-chip oscillator clock or subsystem clock is being used as the CPU clock.
   2)
   3) If the deep sleep mode is entered and then released while the high-speed on-chip oscillator clock is being used as the CPU clock with the X1 clock 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 deep sleep mode is released.)
3. The X1 clock oscillation stabilization wait time does not include the time until clock oscillation starts (see “a” below).



4. F_X: X1 clock oscillation frequency.

4.3.5 Oscillation stabilization time select register (OSTS)

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

If the X1 clock is made to oscillate, it automatically waits for the time set in the OSTS register after the X1 oscillation circuit runs (MSTOP=0).

If the CPU clock is switched from the high-speed on-chip oscillator clock or the sub-system clock to the X1 clock, or if the CPU clock is the high-speed on-chip oscillator clock and is released from deep sleep mode after being transferred to deep sleep mode while the X1 clock is oscillating, it is necessary to confirm whether or not an oscillation stabilization time has elapsed by means of the oscillation stabilization time counter status register (OSTC).

It can confirm the time set in advance by the OSTS register through the OSTC register.

The OSTS register is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register changes to "07H".

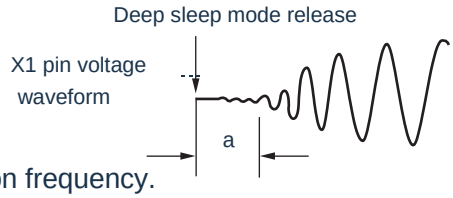
Table 4-7: Format of oscillation stabilization time select register (OSTS)

Address:	40020403H	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=4MHz	F_X=16MHz
0	0	0	$2^8/F_X$	64us	16us
0	0	1	$2^9/F_X$	128us	32us
0	1	0	$2^{10}/F_X$	256us	64us
0	1	1	$2^{11}/F_X$	512us	128us
1	0	0	$2^{13}/F_X$	2.048ms	512us
1	0	1	$2^{15}/F_X$	8.192ms	2.048ms
1	1	0	$2^{17}/F_X$	32.768ms	8.192ms
1	1	1	$2^{18}/F_X$	65.536ms	16.384ms

Remark:

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 the value greater than the count value which is to be checked by the OSTC register after the oscillation starts.
   1) If the X1 clock starts oscillation while the high-speed on-chip oscillator clock or subsystem clock is being used as the CPU clock.
   2) If the deep sleep mode is entered and then released while the high-speed on-chip oscillator clock is being used as the CPU clock with the X1 clock 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 deep sleep mode is released.)
3. The X1 clock oscillation stabilization wait time does not include the time until clock oscillation starts (see “a” below).



4. F_X: X1 clock oscillation frequency.

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

These registers are 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 following peripheral functions controlled by these registers, the corresponding bit must be set to "1" before initial setting of the peripheral functions.

Real-time clock, 15-bit interval timer

A/D converter

Serial interface IICA0

General-purpose serial communication unit 1

General-purpose serial communication unit 0

General-purpose timer unit 1

General-purpose timer unit 0

EPWM

SPI

The PER0 register and PER1 register are set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of these registers changes to "00H".

Table 4-8: Format of peripheral enable register 0 (PER0) (1/3)

Address:	40020420H	After reset: 00H		R/W			
Symbol	7	6	5	4	3	2	1	0
PER0	RTCEN[Note]	0	ADCEN	IICAEN	SCI1EN	SCI0EN	TM41EN	TM40EN

RTCEN	Control of real-time clock (RTC) and 15-bit interval timer input clock supply
0	Stops input clock supply. • SFR used by the real-time clock (RTC) and 15-bit interval timer cannot be written. • The real-time clock (RTC) and interval timer are in the reset status.
1	Enables input clock supply. • SFR used by the real-time clock (RTC) and 15-bit interval timer can be read and written.

Note: The RTCEN bit is initialized only at power-on reset and remains unchanged at other resets.

Table 4-8: Format of peripheral enable register 0 (PER0) (2/3)

Address:	40020420H	After reset: 00H		R/W			
Symbol	7	6	5	4	3	2	1	0
PER0	RTCEN	0	ADCEN	IICAEN	SCI1EN	SCI0EN	TM41EN	TM40EN

ADCEN	Control of A/D converter input clock supply
0	Stop to supply the input clock. • The SFR used by the A/D converter cannot be written. • The A/D converter is in the reset state.
1	Supply the input clock. • The SFR used by the A/D converter can read and write.

IICAEN	Control of serial interface IICA0 input clock supply
0	Stop to supply the input clock. • The SFR used by the serial interface IICA0 cannot be written. • Serial interface IICA0 is in the reset state.
1	Supply the input clock. • The SFR used by the serial interface IICA0 can read and write.

SCI1EN	Control of general-purpose serial communication unit 1 input clock supply
0	Stop to supply the input clock. • The SFR used by general-purpose serial communication unit 1 cannot be written. • General purpose serial communication unit 1 is in the reset state.
1	Supply the input clock. • The SFR used by the general-purpose serial communication unit 1 can read and write.

SCI0EN	Control of general-purpose serial communication unit 0 input clock supply
0	Stop to supply the input clock. • The SFR used by general-purpose serial communication unit 0 cannot be written. • General purpose serial communication unit 0 is in the reset state.
1	Supply the input clock. • The SFR used by the general-purpose serial communication unit 0 can read and write.

Table 4-8: Format of peripheral enable register 0 (PER0) (3/3)

Address:	40020420H	After reset: 00H	R/W				
Symbol	7	6	5	4	3	2	1	0
PER0	RTCEN	0	ADCEN	IICAEN	SCI1EN	SCI0EN	TM41EN	TM40EN

TM41EN	Control of general-purpose timer unit 1 input clock supply
0	Stop to supply the input clock. • The SFR used by general-purpose timer unit 1 cannot be written. • General-purpose timer unit 1 is in the reset state.
1	Supply the input clock. • The SFR used by general-purpose timer unit 1 can read and write.

TM40EN	Control of general-purpose timer unit 0 input clock supply
0	Stop to supply the input clock. • The SFR used by general-purpose timer unit 0 cannot be written. • General-purpose timer unit 0 is in the reset state.
1	Supply the input clock. • The SFR used by general-purpose timer unit 0 can read and write.

Table 4-9: Format of peripheral enable register 1 (PER1)

Address:	4002081AH	After reset: 00H	R/W				
Symbol	7	6	5	4	3	2	1	0
PER1	SPIEN	0	-	0	0	EPWMEN	0	0

SPIEN	Control of SPI input clock supply
0	Stop to supply the input clock. • SPI cannot run.
1	Supply the input clock. • SPI can run.

EPWMEN	Control of EPWM input clock supply
0	Stop to supply the input clock. • EPWM cannot run.
1	Supply the input clock. • EPWM can run.

4.3.7 Subsystem clock supply mode control register (OSMC)

This register is used to reduce power consumption by stopping unnecessary clock functions.

Setting the RTCLPC bit to "1" stops clocking peripheral functions other than the real-time clock and the 15-bit interval timer in the deep sleep mode or in the sleep mode in which the CPU operates with the subsystem clock, and thus reduces power consumption.

In addition, the real-time clock and the operation clock of the 15-bit interval timer can be selected via the OSMC register.

The OSMC registers are set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register changes to "00H".

Table 4-10: Format of subsystem clock supply mode control register (OSMC)

Address:	40020423H	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 deep sleep mode or sleep mode while subsystem clock is selected as CPU clock
0	Enables sub-system clocks for peripheral functions (Refer to Tables 18-1 to 18-3 for the peripheral functions that are enabled to operate).
1	Stop to supply a subsystem clock to peripheral functions other than the real-time clock and 15-bit interval timer.

WUTMMCK0	Selection of operation clock for real-time clock and 15-bit interval timer.
0	The subsystem clock is the operating clock of the real-time clock and the 15-bit interval timer.
1	The low-speed internal oscillator clock is the operating clock of the real-time clock and the 15-bit interval timer.

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

This is the register that changes the frequency of the high-speed on-chip oscillator set by the option byte (000C2H). However, the frequency that can be selected varies depending on the values of the FRQSEL4 bit and FRQSEL3 bit of the option byte (000C2H).

The HOCODIV register is set by an 8-bit memory manipulation instruction.

After the reset signal is generated, the value of this register changes to the set value of the FRQSEL2 to FRQSEL0 bits of the option byte (000C2H).

Table 4-11: Format of high-speed on-chip oscillator frequency select register (HOCODIV)

Address:	40021C20H	After reset: Setting value of FRQSEL2~FRQSEL0 bits of 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 clock frequency for high-speed on-chip oscillator	
			FRQSEL4,3=00	FRQSEL4,3=01
0	0	0	F_{IH}=48MHz F_{HOCO}=48MHz	F_{IH}=64MHz F_{HOCO}=64MHz
0	0	1	F_{IH}=24MHz F_{HOCO}=48MHz	F_{IH}=32MHz F_{HOCO}=64MHz
0	1	0	F_{IH}=12MHz F_{HOCO}=48MHz	F_{IH}=16MHz F_{HOCO}=64MHz
0	1	1	F_{IH}=6MHz F_{HOCO}=48MHz	F_{IH}=8MHz F_{HOCO}=64MHz
1	0	0	F_{IH}=3MHz F_{HOCO}=48MHz	F_{IH}=4MHz F_{HOCO}=64MHz
1	0	1	Prohibit settings.	F_{IH}=2MHz F_{HOCO}=64MHz
Other than the above			Prohibit settings.	

Notice:

1. The HOCODIV register must be set in the state where the high-speed on-chip oscillator clock (F_{IH}) is selected as the CPU/peripheral hardware clock (F_{CLK}).
2. After changing the frequency via the HOCODIV register, frequency switching is performed after the following transfer times.
   1) Runs for up to 3 clocks at the frequency before the change.
   2) Wait for up to 3 CPU/peripheral hardware clocks at the changed frequency

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

This register is used to adjust the accuracy of the high-speed on-chip oscillator. It can be used for self-measurement and accuracy correction of the high-speed on-chip oscillator frequency using a timer with high-precision external clock input, etc. The HIOTRM register is set by an 8-bit memory manipulation instruction.

Table 4-12: Format of high-speed on-chip oscillator trim register (HIOTRM)

Address: 40021C00H After reset: Note R/W

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

HIOTRM5	HIOTRM4	HIOTRM3	HIOTRM2	HIOTRM1	HIOTRM0	High-speed on-chip oscillator
0	0	0	0	0	0	minimum speed
0	0	0	0	0	1	↑
0	0	0	0	1	0	↑
0	0	0	0	1	1	↑
0	0	0	1	0	0	↑
• • •						↑
1	1	1	1	1	0	↓
1	1	1	1	1	1	maximum speed

Note: If the temperature and V_{DD} pin voltage change after correcting for accuracy, the frequency changes.

Notice:

1. In case of temperature and V_{DD} pin voltage changes, it is necessary to perform calibration before or at regular intervals to request frequency accuracy.
2. The reset value is the adjusted value at the time of shipment.
3. Each bit of the HIOTRM register corrects the clock accuracy of the high-speed on-chip oscillator by about 0.05%.

4.3.10 Subsystem clock select register (SUBCKSEL)

The SUBCKSEL register is the register that selects the subsystem clock F_{SUB} and the low-speed on-chip oscillator clock F_{IL}.

The SUBCKSEL register is set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register changes to "00H".

Table 4-13: Format of subsystem clock select register (SUBCKSEL)

Address:	40020407H	After reset: 00H	R/W				
Symbol	7	6	5	4	3	2	1	0
SUBCKSEL	0	0	0	0	0	0	0	SELLOSC

SELLOSC	Subsystem and low-speed on-chip oscillator clock frequency selection
0	• Select the subsystem clock.
1	• Selects the low-speed on-chip oscillator clock.

4.3.11 Power mode control protection register (PMUKEY)

The PMUKEY register is a register for controlling the protection of PMUCTL by the power supply mode.

The PMUKEY register is set by a 16-bit memory manipulation instruction.

After a reset signal is generated, the value of this register changes to "0000H".

Table 4-14: Format of power mode control protection register (PMUKEY)

Address:	40020408H	After reset: 0000H	R/W Note1				
Symbol	15						0
PMUKEY							

PMUKEY	Selection of power mode control protection register
Write 192AH first Then 3E4FH	• Release the PMUCTL write protection. Write control of the PWDNEN bit of PMUCTL is enabled by writing 192AH and 3E4FH to PMUKEY successively.
Other	• The PMUCTL write setting is invalid.

4.3.12 Power mode control register (PMUCTL)

The PMUCTL register is the register that controls the enable power supply control mode.

The PMUCTL register is set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register changes to "00H" and the write protection is turned on, and the write control is released by PMUKEY.

Table 4-15: Format of power mode control register (PMUCTL)

Address:	4002040AH	After reset: 00H	R/W				
Symbol	7	6	5	4	3	2	1	0
PMUCTL								PWDNEN

PWDNEN	Selection of power mode control register
0	• Partial power-down mode disabled.
1	• Partial power-down mode enabled.

Notice: Release PMUCTL write protection via PMUKEY .

4.4 System clock oscillation circuit

4.4.1 X1 oscillation circuit

The X1 oscillator circuit is oscillated by a crystal resonator or ceramic resonator (4~16MHz) connected to pins X1 and X2. An external clock can also be input, where a clock signal must be input to the EXCLK pin.

When using the X1 oscillation circuit, bit 7 and bit 6 (EXCLK, OSCSEL) of the clock operation mode control register (CMC) must be set as follows:

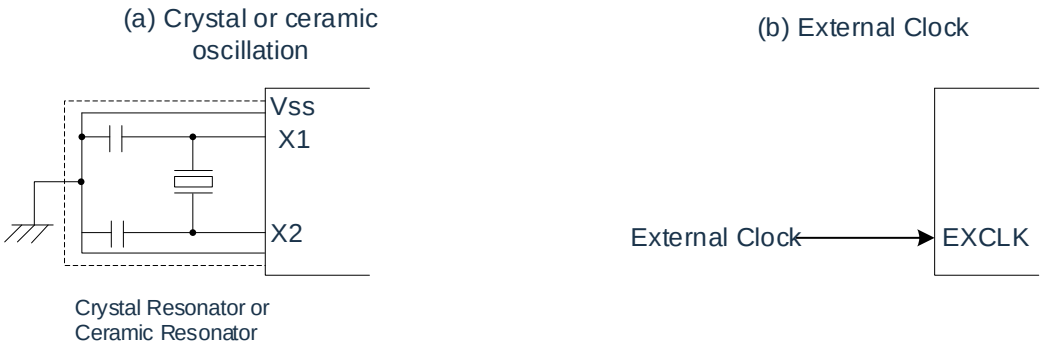
Crystal or ceramic oscillation. EXCLK, OSCSEL=0, 1

External clock input: EXCLK, OSCSEL=1, 1

When the X1 oscillation circuit is not used, it must be set to the port mode (EXCLK, OSCSEL=0, 0). Also, refer to "Table 2-3 Handling of Unused Pins" when not used as an input/output port.

An example of an external circuit for the X1 oscillation circuit is shown in Figure 4-2.

Figure 4-2: Example of an external circuit for X1 oscillation circuit



Notices are shown on the next page.

4.4.2 XT1 oscillation circuit

The XT1 oscillation circuit is oscillated by a crystal resonator (32.768KHz (typical)) connected to the XT1 pin and XT2 pin. When using the XT1 oscillation circuit, bit 4 (OSCSELS) of the clock operation mode control register (CMC) must be set to "1" to enable the external clock to be input as well, in which case the clock signal must be input to the EXCLKS pin.

When using the XT1 oscillation circuit, bit 5 and bit 4 (EXCLKS, OSCSELS) of the clock operation mode control register (CMC) must be set as follows:

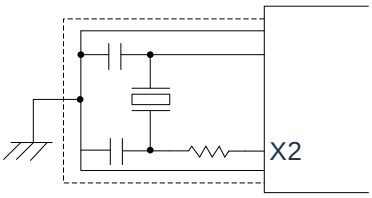
Crystal oscillation: EXCLKS, OSCSELS=0, 1

External clock input: EXCLKS, OSCSELS=1, 1

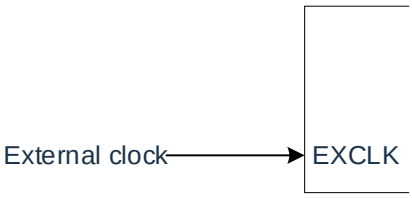
When the XT1 oscillation circuit is not used, it must be set to the port mode (EXCLKS, OSCSELS=0, 0). Also, when not used as an input/output port, refer to “Table 2-3 Handling of Unused Pins”. An example of an external circuit for the XT1 oscillation circuit is shown in Figure 4-3.

Figure 4-3: Example of an external circuit for XT1 oscillation circuit

(a) Crystal oscillation



(b) External clock



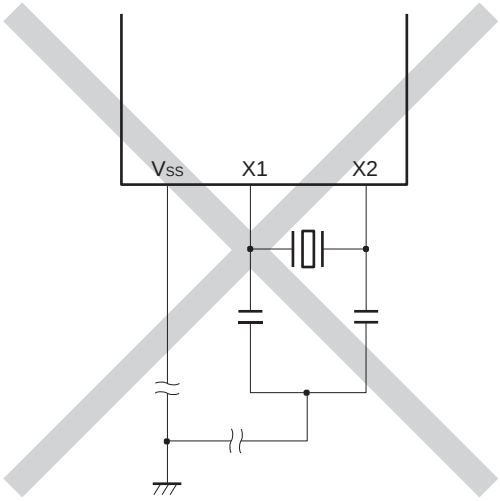
Notice: When using the X1 oscillator circuit and the XT1 oscillator circuit, the dashed portions of Figure 4-2(a) and Figure 4-2(b) must be routed in order to avoid the effects of wiring capacitance, etc., by the following method:

1) Keep the wiring length as short as possible.
2) 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 flow.
3) Always make the ground point of the oscillator capacitor the same potential as V_{SS}. Do not ground the capacitor to a ground pattern through which a high current flow.
4) Do not fetch signals from the oscillator.

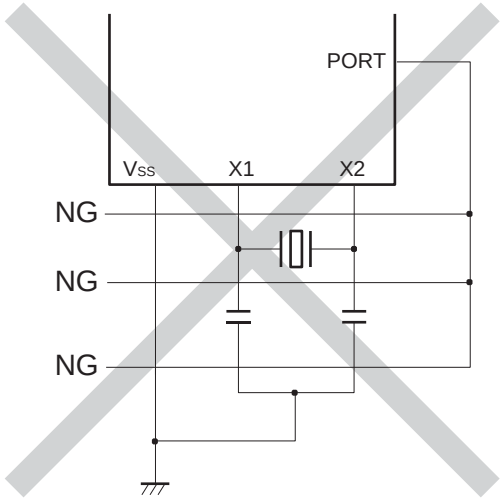
Figure 4-4 shows examples of incorrect resonator connection.

Figure 4-4: Examples of incorrect resonator connection (1/2)

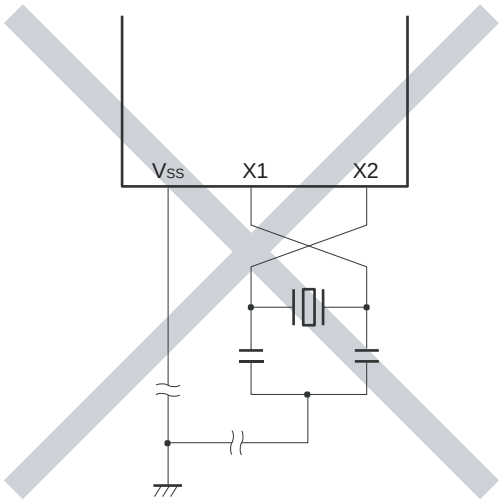
(a) Too long wiring



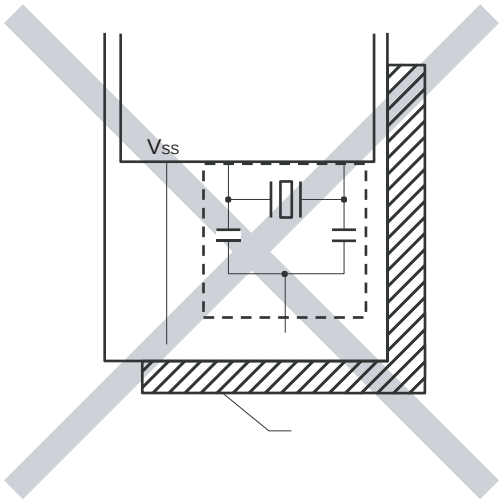
(b) Crossed signal line



(c) The X1 and X2 signal line wires cross.



(d) A power supply/GND pattern exists under the X1 and X2 wires.

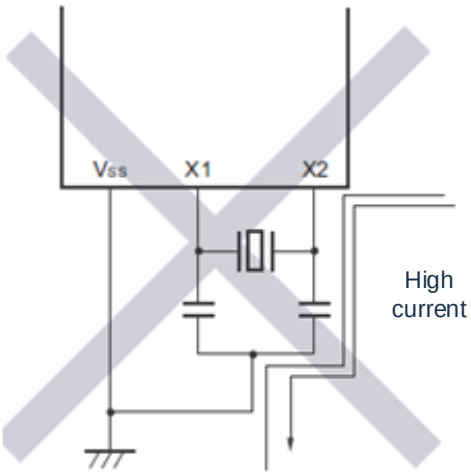


Notice:

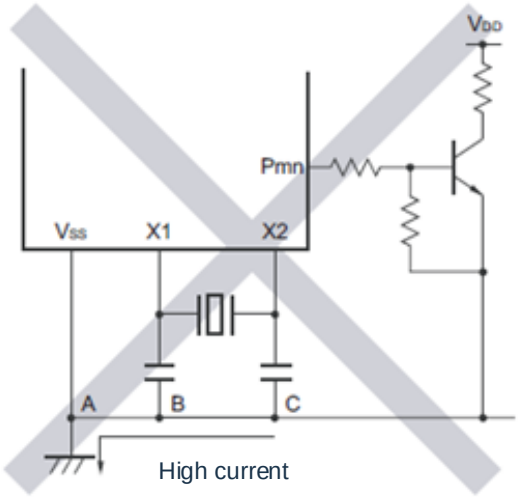
1. In multilayer boards or double-sided boards, power or ground graphics should not be configured below the wiring area (dashed portion of the diagram) for pins X1, X2 and the resonator. The wiring must not create a capacitive component that would affect the oscillation characteristics.
2. In the case of using the subsystem clock, please read with XT1 and XT2 instead of X1 and X2 respectively, and insert series resistors on the XT2 side.

Figure 4-4: Examples of incorrect resonator connection (2/2)

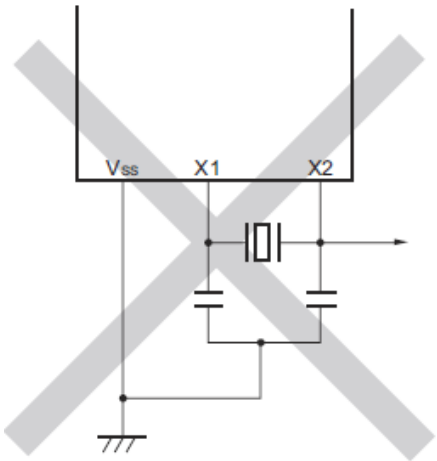
(e) Wiring near high alternating current



(f) Current flowing through ground line of oscillator (potential at points A, B, and C fluctuates)



(g) Signals are fetched



Notice:

1. When X2 and XT1 are in parallel, the crosstalk noise of X2 will be superimposed to XT1 and cause misoperation.
2. In the case of using the subsystem clock, please read with XT1 and XT2 instead of X1 and X2 respectively, and insert series resistors on the XT2 side.

4.4.3 High-speed on-chip oscillator

The CMS32L032 has a built-in high-speed on-chip oscillator. The frequency can be selected from 64MHz, 48MHz, 32MHz, 24MHz, 16MHz, 12MHz, 8MHz, 6MHz, 4MHz, 3MHz, and 2MHz using the option byte (000C2H). Oscillation can be controlled by bit0 (HIOSTOP) of the clock operation status control register (CSC).

The high-speed on-chip oscillator automatically starts oscillating after power-on reset is released.

4.4.4 Low-speed on-chip oscillator

The CMS32L032 has a built-in low-speed on-chip oscillator.

The low-speed on-chip oscillator clock is used as the clock for the watchdog timer, the real-time clock, the 15-bit interval timer, and the external reference clock for the SysTick timer, as well as the CPU clock and peripheral module clock.

When bit4 (WDTON) of the option byte (000C0H) or bit4 (WUTMMCK0) of the subsystem clock supply mode control register (OSMC) is "1", the low-speed on-chip oscillator oscillates.

When the watchdog timer stops running and the WUTMMCK0 bit is not "0", the low-speed internal oscillator continues to oscillate. However, if the watchdog timer is running and the WUTMMCK0 bit or the SELLOSC bit is "0", the low-speed internal oscillator stops oscillating when the WDSTBYON bit is "0" and it is in the sleep mode or deep sleep mode. When the watchdog timer is running, the low-speed internal oscillator clock does not stop running even if the program is out of control.

4.5 Operation of clock generation circuit

The clock generation circuit generates various clocks as shown below and controls the CPU operation modes such as standby mode (refer to Figure 4-1).

F_{MAIN}: Main system clock frequency

F_{MX}: High-speed system clock frequency

F_X: X1 clock oscillation frequency

F_{EX}: External main system clock frequency

F_{IH}: High-speed on-chip oscillator clock frequency

F_{SUB}: Subsystem clock frequency

F_{XT}: XT1 clock oscillation frequency

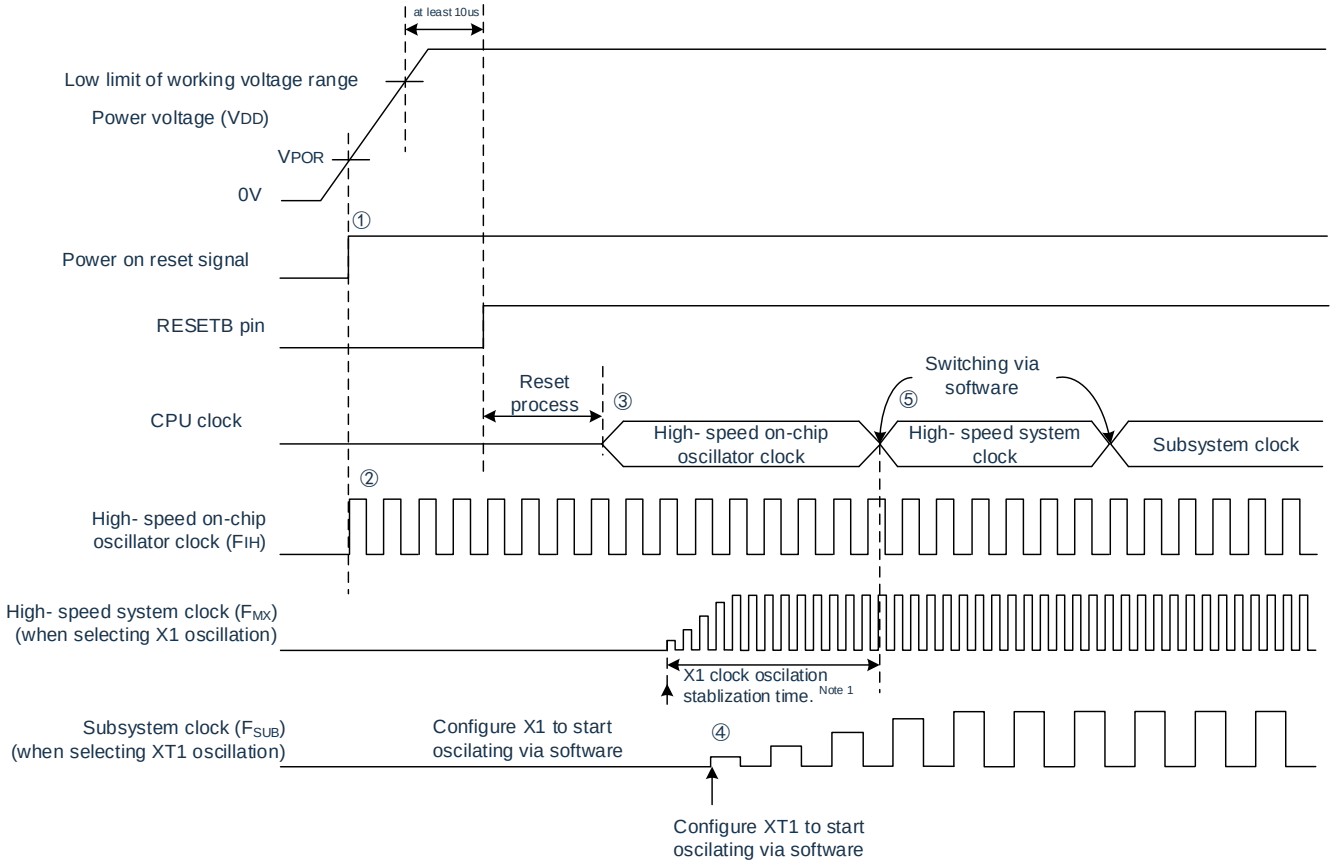
F_{EXS}: External subsystem clock frequency

F_{CLK}: CPU/peripheral hardware clock frequency

F_{IL}: Low-speed on-chip oscillator clock frequency

After the CMS32L032 is released from reset, the CPU begins operation through the output of the high-speed on-chip oscillator. The operation of the clock generation circuit when the power is turned on is shown in Figure 4-5.

Figure 4-5: Operation of the clock generation circuit when the power is turned on



1) After power is turned on, an internal reset signal is generated through the power-on reset (POR) circuit. However, the reset state is maintained by a voltage detection circuit or an external reset until the operating voltage range shown in the AC characteristics of the datasheet is reached (the above figure shows an example when an external reset is used).
2) The high-speed on-chip oscillator starts oscillating automatically after the reset is released.
3) After the reset is released, voltage stabilization waiting and reset processing are performed, and then the CPU starts running with a high-speed on-chip oscillator clock.
4) The start of oscillation of the X1 clock or XT1 clock must be set by software (see “4.6.2 Example of setting X1 oscillation circuit” and “4.6.3 Example of controlling XT1 oscillation clock”).
5) If you want to switch the CPU clock to X1 clock or XT1 clock, you must set the switch by software after waiting for the clock oscillation to stabilize (see“4.6.2 Example of setting X1 oscillation circuit”and “4.6.3 Example of controlling XT1 oscillation clock”).

Note 1: When the reset is released, the oscillation stabilization time of the X1 clock must be confirmed by the oscillation stabilization time counter status register (OSTC).

Remark: If you use an external clock input from the EXCLK pin, there is no need for an oscillation stabilization wait time.

4.6 Clock control

4.6.1 Example of setting up a high-speed on-chip oscillator

The CPU/peripheral hardware clock (F_{CLK}) must run at the high-speed on-chip oscillator clock after the reset is released. The frequency of the high-speed on-chip oscillator can be selected from 64MHz, 48MHz, 32MHz, 24MHz, 16MHz, 12MHz, 8MHz, 6MHz, 4MHz, 3MHz, and 2MHz by using bits FRQSEL0 to FRQSEL4 of the option byte (000C2H). In addition, the frequency can be changed by the high-speed on-chip oscillator register (HOCODIV).

[Option byte setting]

Address: 000C2H	7	6	5	4	3	2	1	0
option byte (000C2H)	1	1	1	FRQSEL4 0	FRQSEL3 0/1	FRQSEL2 0/1	FRQSEL1 0/1	FRQSEL0 0/1

FRQSEL4	FRQSEL3	FRQSEL2	FRQSEL1	FRQSEL0	High-speed on-chip oscillator frequency	
					F_{HOCO}	F_{IH}
0	1	0	0	0	64MHz	64MHz
0	0	0	0	0	48MHz	48MHz
0	1	0	0	1	64MHz	32MHz
0	0	0	0	1	48MHz	24MHz
0	1	0	1	0	64MHz	16MHz
0	0	0	1	0	48MHz	12MHz
0	1	0	1	1	64MHz	8MHz
0	0	0	1	1	48MHz	6MHz
0	1	1	0	0	64MHz	4MHz
0	0	1	0	0	48MHz	3MHz
0	1	1	0	1	64MHz	2MHz
Other than the above					Prohibited settings.	

[Setting of high-speed on-chip oscillator frequency selection register (HOCODIV)]

Address: 0x40021C20

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

HOCODIV2	HOCODIV1	HOCODIV0	Selection of clock frequency for high-speed on-chip oscillator	
			FRQSEL4,3=00	FRQSEL4,3=01
0	0	0	F_{IH}=48MHz F_{HOCO}=48MHz	F_{IH}=64MHz F_{HOCO}=64MHz
0	0	1	F_{IH}=24MHz F_{HOCO}=48MHz	F_{IH}=32MHz F_{HOCO}=64MHz
0	1	0	F_{IH}=12MHz F_{HOCO}=48MHz	F_{IH}=16MHz F_{HOCO}=64MHz
0	1	1	F_{IH}=6MHz F_{HOCO}=48MHz	F_{IH}=8MHz F_{HOCO}=64MHz
1	0	0	F_{IH}=3MHz F_{HOCO}=48MHz	F_{IH}=4MHz F_{HOCO}=64MHz
1	0	1	Prohibited settings.	F_{IH}=2MHz F_{HOCO}=64MHz
Other than the above			Prohibited settings.	

4.6.2 Example of setting X1 oscillation circuit

After a reset release, the CPU/peripheral hardware clock (F_{CLK}) always starts operating with the high-speed on-chip oscillator clock. To subsequently change the clock to the X1 oscillation clock, set the oscillator and start oscillation by using the oscillation stabilization time select register (OSTS) and clock operation mode control register (CMC) and clock operation status control register (CSC) and wait for oscillation to stabilize by using the oscillation stabilization time counter status register (OSTC). After the oscillation stabilizes, set the X1 oscillation clock to F_{CLK} by using the system clock control register (CKC).

[Setting of registers] The registers must be set in the order of ① to ⑤.

① Set the OSCSEL bit of the CMC register to "1" to set the frequency division selection of the X1 clock through MOSCDIV to operate the X1 oscillation circuit.

	7	6	5	4	3	2	1	0
CMC	EXCLK0	OSCSEL1	EXCLKS0	OSCSELS0	MOSCDIV 0	AMPHS10	AMPHS00	0

② Select the oscillation stabilization time of the X1 oscillation circuit when the deep sleep mode is released through the OSTS register.

Example) Setting values when a wait of at least 102us is set based on a 10 MHz resonator.

	7	6	5	4	3	2	1	0
OSTS	0	0	0	0	0	OSTS20	OSTS11	OSTS00

③ Clear the MSTOP bit of the CSC register to "0" so that the X1 oscillator circuit starts oscillating.

	7	6	5	4	3	2	1	0
CSC	MSTOP0	XTSTOP1	0	0	0	0	0	HIOSTOP0

④ ④ Wait for the oscillation of the X1 oscillation circuit to stabilize through the OSTC register.

Example) Wait until the bits reach the following values when a wait of at least 102us is set based on a 10 MHz resonator.

	7	6	5	4	3	2	1	0
OSTC	MOST81	MOST91	MOST101	MOST110	MOST130	MOST150	MOST170	MOST180

⑤ Set the X1 oscillating clock to the CPU/peripheral hardware clock via the MCM0 bit of the CKC register.

	7	6	5	4	3	2	1	0
CKC	CLS0	CSS0	MCS0	MCM01	0	0	0	0

4.6.3 Example of controlling XT1 oscillation clock

After a reset release, the CPU/peripheral hardware clock (F_{CLK}) always starts operating with the high-speed on-chip oscillator clock. To subsequently change the clock to the XT1 oscillation clock, set the oscillator and start oscillation by using the operation speed mode control register (OSMC), clock operation mode control register (CMC), and clock operation status control register (CSC), set the XT1 oscillation clock to F_{CLK} by using the system clock control register (CKC).

[Setting of registers] The registers must be set in the order of ① to ⑤.

① In the deep sleep mode or the sleep mode where the CPU is running on the sub-system clock, the RTCLPC bit must be set to "1" whenever the real-time clock and the 15-bit interval timer are made to run on the sub-system clock (Low consumption current).

	7	6	5	4	3	2	1	0
OSMC	RTCLPC 0/1	0	0	WUTMMCK00	0	0	0	0

② Set the OSCSELS bit of the CMC register to "1" to operate the XT1 oscillation circuit.

	7	6	5	4	3	2	1	0
CMC	EXCLK0	OSCSEL0	EXCLKS0	OSCSELS1	MOCDIV0	AMPHS1 0/1	AMPHS0 0/1	0

AMPHS0 bit and AMPHS1 bit: Set the oscillation mode of XT1 oscillation circuit.

③ Clear the XTSTOP bit of the CSC register to "0" so that the XT1 oscillator circuit starts oscillating.

	7	6	5	4	3	2	1	0
CSC	MSTOP1	XTSTOP0	0	0	0	0	0	HIOSTOP0

④ The oscillation stabilization time required by the subsystem clock must be waited for by software, timer function, etc.

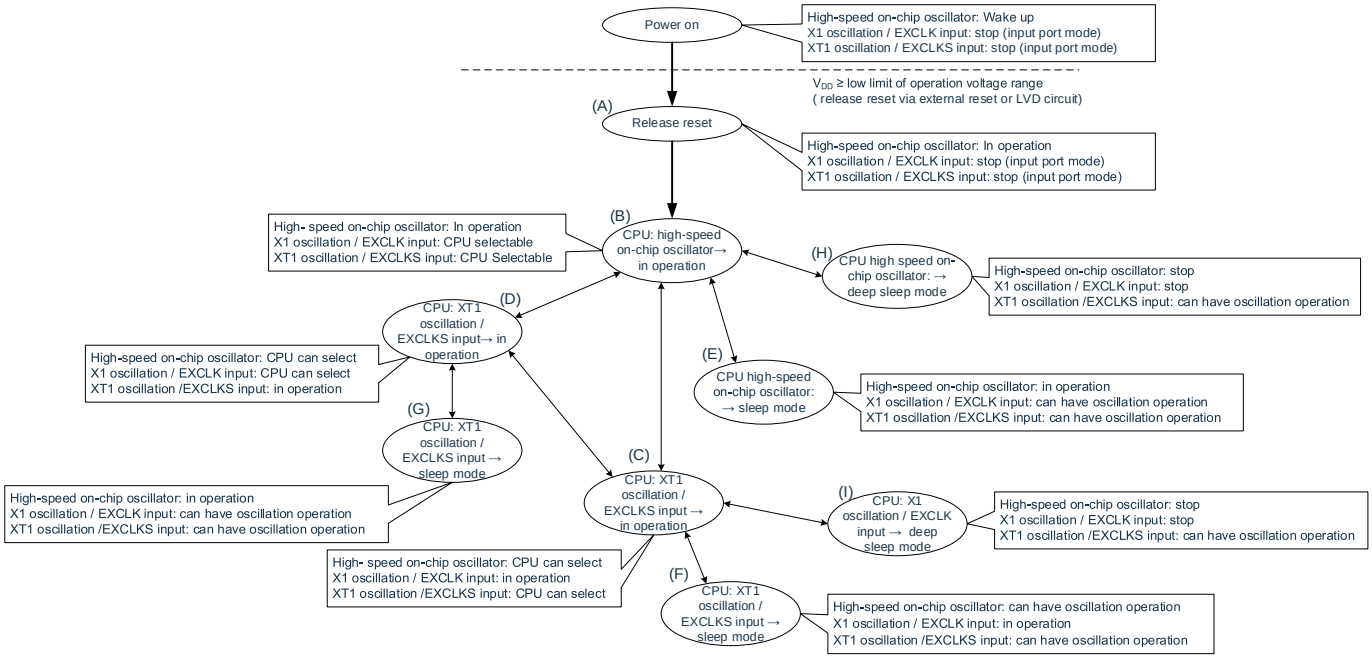
⑤ Set the XT1 oscillating clock to the CPU/peripheral hardware clock via the CSS bit of the CKC register.

	7	6	5	4	3	2	1	0
CKC	CLS0	CSS1	MCS0	MCM0	0	0	0	0

4.6.4 CPU clock status transition diagram

Figure 4-6 shows the CPU clock status transition diagram of this product.

Figure 4-6: State transfer diagram for CPU clock



Examples of CPU clock transfer and SFR register setting are in Table 4-16.

Table 4-16: CPU clock transition and SFR register setting examples (1/4)

(1) CPU operating with high-speed on-chip oscillator clock (B) after reset release (A)

Status transition	SFR register setting
(A)→(B)	SFR registers do not have to be set (default status after reset release).

(2) CPU operating with high-speed system clock (C) after reset release (A) (The CPU operates (B) with a high-speed on-chip oscillator clock immediately after the reset is released)

(SFR register setting order) ⟶

SFR register setting flag / State transition	CMC register Note 1		OSTS register	CSC register	OSTC register	CKC register
	EXCLK	OSCSEL		MSTOP		MCM0
(A)→(B)→(C) (X1 clock)	0	1	Note2	0	Need to confirm	1
(A)→(B)→(C) (External main system clock)	1	1	Note2	0	No need to confirm	1

(3) After the reset (A) is released, the CPU is transferred to the subsystem clock to operate (D). (The CPU operates (B) with a high-speed on-chip oscillator clock immediately after the reset is released)

(SFR register setting order) ⟶

SFR register setting flag / State transition	CMC register Note 3				CSC register	Oscillation stabilization waiting	CKC register
	EXCLKS	OSCSELS	AMPHS1	AMPHS0	XTSTOP		CSS
(A)→(B)→(D) (XT1 clock)	0	1	0/1	0/1	0	Need	1
(A)→(B)→(D) (External subsystem clock)	1	1	×	×	0	Need	1

Note 1: The clock operation mode control register (CMC) can only be written 1 time by an 8-bit memory manipulation instruction after the reset is released.

Note 2: The oscillation stabilization time of the oscillation stabilization time selection register (OSTS) must be set as follows.

Expected oscillation stabilization time of oscillation stabilization time counter status register (OSTC) ≤ oscillation stabilization time set in the OSTS register

Note 3: The clock operation mode control register (CMC) can only be written 1 time by an 8-bit memory manipulation instruction after the reset is released.

Notice: The clock must be set after the supply voltage reaches the set clock runnable voltage (refer to the datasheet for electrical characteristics).

Remark:

1. ×: Ignore.
2. (A) to (I) of Table 4-16 correspond to (A) to (I) of Figure 4-6.

Table 4-16: CPU clock transition and SFR register setting examples (2/4)

(4) The CPU moves from high-speed on-chip oscillator clock operation (B) to high-speed system clock operation (C).

(SFR register setting order) →

SFR register setting flag / State transition	CMC register Note 1		OSTS register	CSC register	OSTC register	CKC register
	EXCLK	OSCSEL		MSTOP		MCM0
(B)→(C) (X1 clock)	0	1	Note2	0	Need to confirm	1
(B)→(C) (External main system clock)	1	1	Note2	0	No need to confirm	1

Not required if already set. Not required for high-speed system clock operation.

(5) The CPU moves from high-speed on-chip oscillator clock operation (B) to subsystem clock operation (D).

(SFR register setting order) →

SFR register setting flag / State transition	CMC register Note 1			CSC register	Oscillation stabilization waiting	CKC register
	EXCLKS	OSCSELS	AMPHS1, 0	XTSTOP		CSS
(B)→(D) (XT1 clock)	0	1	00: Low power oscillation 01: Normal oscillation 10: Low power oscillation	0	Need	1
(B)→(D) (External subsystem clock)	1	1	×	0	Need	1

Not required if already set. Not required for subsystem clock operation.

(6) The CPU moves from high-speed system clock operation (C) to high-speed on-chip oscillator clock operation (B).

(SFR register setting order) →

SFR register setting flag / State transition	CSC register	Oscillation stabilization waiting	CKC register
	HIOSTOP		MCM0
(C)→(B)	0	1us	0

Not required for high-speed on-chip oscillator clock

Note 1: The clock operation mode control register (CMC) can only be written 1 time by an 8-bit memory manipulation instruction after the reset is released.

Note 2: The oscillation stabilization time of the oscillation stabilization time selection register (OSTS) must be set as follows.

Expected oscillation stabilization time of oscillation stabilization time status register (OSTC) ⩽ oscillation stabilization time set in the OSTS register

Notice: The clock must be set after the supply voltage reaches the set clock runnable voltage (refer to the datasheet for electrical characteristics).

Remark:

1. ×: Ignore.
2. (A) to (I) of Table 4-16 correspond to (A) to (I) of Figure 4-6.
3. The oscillation accuracy of the high-speed on-chip oscillator clock stabilization wait time varies depending on temperature conditions and during deep sleep mode.

Table 4-16: CPU clock transition and SFR register setting examples (3/4)

(7) CPU clock changing from high-speed system clock (C) to subsystem clock (D)

(SFR register setting order) →

SFR register setting flag / State transition	CSC register	Oscillation stabilization waiting	CKC register
	XTSTOP		CSS
(C)→(D)	0	Need	1

Unnecessary if the CPU is operating with the subsystem clock

(8) CPU clock changing from subsystem clock (D) to high-speed on-chip oscillator clock (B)

(SFR register setting order) →

SFR register setting flag / State transition	CSC register	Oscillation stabilization waiting	CKC register
	HIOSTOP		CSS
(D)→(B)	0	1us	0

Unnecessary if the CPU is operating with the high-speed on-chip oscillator clock

(9) CPU clock changing from subsystem clock (D) to high-speed system clock (C)

(SFR register setting order) →

SFR register setting flag / State transition	OSTS register	CSC register	OSTC register	CKC register
		MSTOP		CSS
(D)→(C) (X1 clock)	Note	0	Need to confirm	0
(D)→(C) (External main system clock)	Note	0	No need to confirm	0

Unnecessary if the CPU is operating with the high-speed system clock

Note: The oscillation stabilization time of the Oscillation Stabilization Time Selection Register (OSTS) must be set as follows.

Expected oscillation stabilization time of oscillation stabilization time counter's status register (OSTC) ≤ oscillation stabilization time set in the OSTS register

Notice:

1. The Clock Operation Mode Control Register (CMC) can be written only once by an 8-bit memory manipulation instruction after the reset is released.
2. 2. The clock must be set after the supply voltage reaches the set clock runnable voltage (refer to the data sheet for electrical characteristics).

Remark:

1. ×: Ignore.
2. (A) to (I) of Table 4-16 correspond to (A) to (I) of Figure 4-6.

Table 4-16: CPU clock transition and SFR register setting examples (4/4)

(10) The CPU moves from high-speed on-chip oscillator clock operation (B) to sleep mode (E)

The CPU moves from high-speed system clock operation (C) to sleep mode (F).

The CPU moves from subsystem clock operation (D) to sleep mode (G).

State transition	Setting contents
(B)→(E) (C)→(F) (D)→(G)	Execute the WFI instruction.

(11) CPU moves from high-speed on-chip oscillator clock operation (B) to deep sleep mode (H).

CPU moves from high-speed system clock operation (C) to deep sleep mode (I).

(Setting order) ⟶

<table>
<tr><th colspan="2">State transition</th><th colspan="3">Setting contents</th></tr>
<tr><td colspan="2">(B)→(H)</td><td rowspan="3">Stop
Peripheral functions that cannot be run in deep sleep mode.</td><td>—</td><td rowspan="3">Bit2 of the SCR register (SLEEPDEEP) is set to 1 and the WFI instruction is executed.</td></tr>
<tr><td rowspan="2">(C)→(I)</td><td>X1 oscillation</td><td>OSTS register setting</td></tr>
<tr><td>External clock</td><td>—</td></tr>
</table>

Notice:

1. The oscillation stabilization time of the Oscillation Stabilization Time Selection Register (OSTS) must be set as follows.
   Expected oscillation stabilization time of oscillation stabilization time counter status register (OSTC) ⩽ oscillation stabilization time set in the OSTS register
2. The Clock Operation Mode Control Register (CMC) can only be written 1 time via an 8-bit memory manipulation instruction after an unreset.
3. The clock must be set after the supply voltage reaches the set clock runnable voltage (refer to the data sheet for electrical characteristics).

Remark:

4. ×: Ignore.
5. (A) to (I) of Table 4-16 correspond to (A) to (I) of Figure 4-6.

4.6.5 Conditions before CPU clock transfer and post-transfer processing

The conditions before the CPU clock transfer and the processing after the transfer are shown below.

Table 4-17: Transfer of CPU clocks (1/2)

CPU clock		Conditions before transfer	Post-transfer processing
Before transfer	After transfer		
High-speed on-chip oscillator clock	X1 clock	The X1 oscillation is stable. • OSCSEL=1, EXCLK=0, MSTOP=0 After oscillation stabilization time	If the oscillation of the high-speed on-chip oscillator is stopped (HIOSTOP=1), the operation current can be reduced.
	External main system clock	Set the external clock entered by the EXCLK pin to be valid. • OSCSEL=1, EXCLK=1, MSTOP=0	
	XT1 clock	The XT1 oscillation is stable. • OSCSELS=1, EXCLKS=0, XTSTOP=0 After oscillation stabilization time	
	External subsystem clock	Set the external clock entered by the EXCLKS pin to be valid. • OSCSELS=1, EXCLKS=1, XTSTOP=0	
X1 clock	High-speed on-chip oscillator clock	Enables high-speed on-chip oscillator to oscillate. • HIOSTOP=0 After oscillation stabilization time	It can stop the X1 oscillation (MSTOP=1).
	External main system clock	Can't transfer	-
	XT1 clock	The XT1 oscillation is stable. • OSCSELS=1, EXCLKS=0, XTSTOP=0 After oscillation stabilization time	It can stop the X1 oscillation (MSTOP=1).
	External subsystem clock	Set the external clock entered by the EXCLKS pin to be valid. • OSCSELS=1, EXCLKS=1, XTSTOP=0	It can stop the X1 oscillation (MSTOP=1).
External main system clock	High-speed on-chipl oscillator clock	Enables high-speed on-chip oscillator to oscillate. • HIOSTOP=0 After oscillation stabilization time	Ability to set the input of the external main system clock invalid (MSTOP=1).
	X1 Clock	Can't transfer	-
	XT1 Clock	The XT1 oscillation is stable. • OSCSELS=1, EXCLKS=0, XTSTOP=0 After oscillation stabilization time	Ability to set the input of the external main system clock invalid (MSTOP=1).
	External subsystem clock	Set the external clock entered by the EXCLKS pin to be valid. • OSCSELS=1, EXCLKS=1, XTSTOP=0	Ability to set the input of the external main system clock invalid (MSTOP=1).

Table 4-17: Transfer of CPU clocks (2/2)

CPU Clock		Conditions before transfer	Post-transfer processing
Before transfer	After transfer		
XT1 clock	High-speed on-chip oscillator	High-speed on-chip oscillator is oscillating and selecting high-speed on-chip	It can stop the XT1 oscillation (XTSTOP=1).
	Clock	oscillator clock is used as the main system clock. • HIOSTOP=0, MCS=0	
	X1 clock	X1 oscillation stabilization and select high-speed system clock as the main system Clock. • OSCSEL=1, EXCLK=0, MSTOP=0 • After oscillation stabilization time • MCS=1	
	External main system clock	The external clock input by the EXCLK pin is set to be valid and the high-speed system clock is selected as the main system clock. • OSCSEL=1, EXCLK=1, MSTOP=0 • MCS=1	
	External subsystem clock	Can't transfer	-
External subsystem clock	High-speed on-chip oscillator	High-speed on-chip oscillator is oscillating and selecting high-speed on-chip	Ability to set the input of the external subsystem clock invalid
	Clock	The oscillator clock is used as the main system clock. • HIOSTOP=0, MCS=0	(XTSTOP=1).
	X1 clock	X1 oscillation stabilization and select high-speed system clock as the main system Clock. • OSCSEL=1, EXCLK=0, MSTOP=0 • After oscillation stabilization time • MCS=1	
	External main system clock	The external clock input by the EXCLK pin is set to be valid and the high-speed system clock is selected as the main system clock. • OSCSEL=1, EXCLK=1, MSTOP=0 • MCS=1	
	XT1 clock	Can't transfer	-

4.6.6 Time required to switch CPU clock and main system clock

It can switch CPU clock (main system clock↔sub system clock) and main system clock (high speed internal oscillator clock↔high speed system clock) by setting bit6 and bit4 (CSS, MCM0) of system clock control register.

The actual switchover does not occur immediately after the CKC register is overridden, but several clocks continue to run with the clock before the switchover after the CKC register is changed (see Table 4-18~Table 4-20).

The CPU can be judged by the bit7 (CLS) of the CKC register whether the CPU is run with the main system clock or the sub system clock. The bit5 (MCS) of the CKC register can be used to determine whether the main system clock operates with a high-speed system clock or a high-speed on-chipl oscillator clock.

If you switch the CPU clock, switch the peripheral hardware clock at the same time.

Table 4-18: Maximum time required to switch main system clock

Clock A	Switch direction	Clock B	Remark
F_{IH}	◀⟷▶	F_{MX}	Refer to Table 4-18.
F_{MAIN}	◀⟷▶	F_{SUB}	Refer to Table 4-19.

Table 4-19: Maximum number of clocks required for F_{IH}↔F_{MX}

Set value before switching		Set value after switching	
MCM0		MCM0	
		0 ($F_{MAIN}=F_{IH}$)	1 ($F_{MAIN}=F_{MX}$)
0 ($F_{MAIN}=F_{IH}$)	$F_{MX}\geq F_{IH}$		2 clocks
	$F_{MX}<F_{IH}$		2 F_{IH}/F_{MX} clocks
1 ($F_{MAIN}=F_{MX}$)	$F_{MX}\geq F_{IH}$	2 F_{MX}/F_{IH} clocks	
	$F_{MX}<F_{IH}$	2 clocks	

Table 4-20: Maximum number of clocks required for F_{MAIN}↔F_{SUB}

Set value before switching	Set value after switching	
CSS	CSS	
	0 ($F_{CLK}=F_{MAIN}$)	1 ($F_{CLK}=F_{SUB}$)
0 ($F_{CLK}=F_{MAIN}$)		1+2 F_{MAIN}/F_{SUB} clocks
1 ($F_{CLK}=F_{SUB}$)	3 clocks	

Remark:

1. The number of clocks in Table 4-19 and Table 4-20 is the number of CPU clocks before the switch.
2. The number of clocks in Table 4-19 and Table 4-20 is the number of clocks rounded to the decimal portion.
   Example when switching the main system clock from the high-speed system clock to the high-speed on-chip oscillator clock (oscillation with F_{IH}=8MHz, F_{MX}=10MHz)
   $2F_{MX}/F_{IH}=2(10/8)=2.53$ clocks

4.6.7 Conditions before clock oscillation is stopped

The following lists the register flag settings for stopping the clock oscillation (disabling external clock input) and conditions before the clock oscillation is stopped.

Table 4-21: Conditions and flag settings before clock oscillation stops

Clock	Conditions before clock oscillation is stopped (external clock input disabled)	Flag settings of SFR register
High-speed on-chip oscillator clock	MCS=1 or CLS=1 (CPU runs at a clock other than the high-speed on-chip oscillator clock)	HIOSTOP=1
X1 clock External main system clock	(MCS=0 or CLS=1) (CPU runs at a clock other than the high-speed system clock)	MSTOP=1
XT1 clock External subsystem clock	CLS=0 (CPU runs at a clock other than the subsystem clock)	XTSTOP=1

4.7 High-speed on-chip oscillation correction

4.7.1 High-speed on-chip oscillation self-adjustment function

This function measures the frequency of a high-speed on-chip oscillator using the subsystem clock F_{SUB} (32.768KHz) as a reference, and corrects the frequency accuracy of the high-speed on-chip oscillator F_{HOCO} in real time.

Table 4-22 is the operating specifications for the high-speed on-chip oscillation frequency correction function, Figure 4-7 is the block diagram of the high-speed on-chip oscillation frequency correction function.

Table 4-22: Operating specifications for the high-speed on-chip oscillation frequency correction function

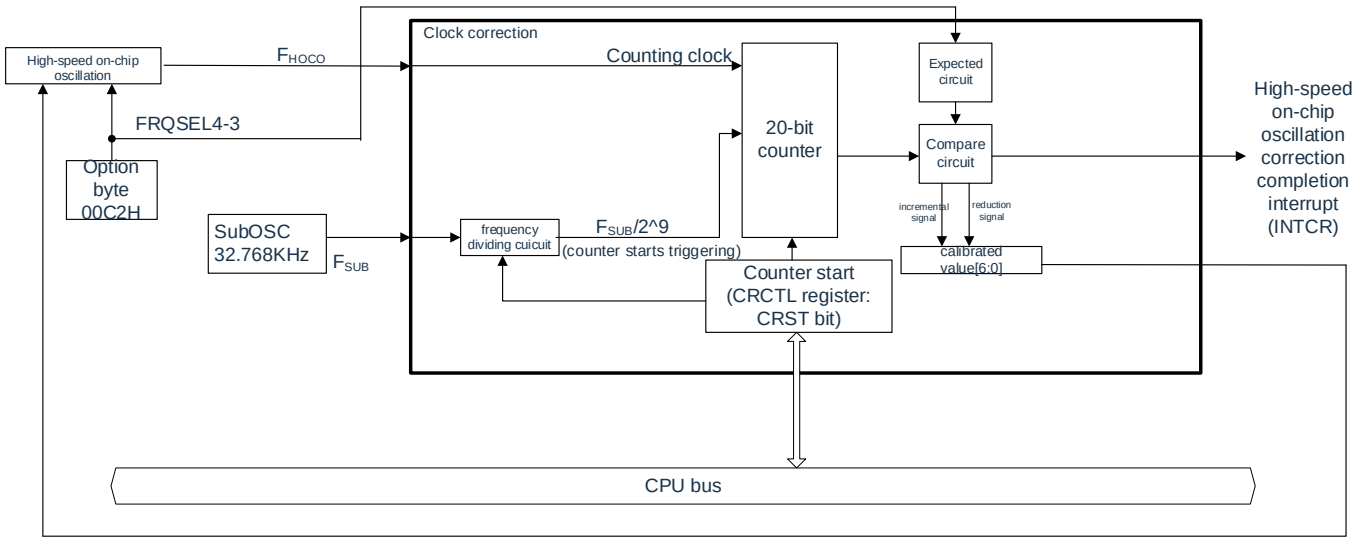
Item	Content
Reference clock	• $F_{SUB}/2^9$ (subsystem clock 32.768KHz)
Calibration object clock	• F_{HOCO}(High-speed on-chip oscillation)
Action mode	• Continuous action mode Continuous high-speed on-chip oscillation frequency correction mode • Interval action mode A mode of high-speed on-chip oscillation frequency correction at intervals using timer clock terminals, etc.
Clock accuracy adjustment function	• Calibration time: Calibration period (31.2ms) X (number of calibrations - 0.5) Note
Interrupt	• Interrupt generated when high-speed on-chip oscillation frequency correction is completed (when interrupt is enabled)

Note: Calibration time: It varies depending on the number of calibrations.

Calibration period: The total time of frequency measuring stage and frequency calibration stage.

Calibration times: The number of times the frequency is corrected to the expected value range

Figure 4-7: Operating specifications for the high-speed on-chip oscillation frequency correction function



4.7.2 Description of register

Table 4-23 shows a list of registers used for the high-speed on-chip oscillation frequency correction function.

Table 4-23: Register list of high-speed on-chip oscillation frequency correction function

Item	Structure
Control register	High-speed on-chip oscillation frequency correction control register (HOCOFC)

4.2.7.1 High-speed on-chip oscillation frequency correction control register (HOCOFC)

Control register for the high-speed on-chip oscillation frequency correction function.

The HOCOFC register is set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register becomes "00H".

Table 4-24: Format of high-speed on-chip oscillation frequency correction control register (HOCOFC)

Address: 0x40022400 After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
HOCOFC	FCMD	FCIE	0	0	0	0	0	FCST

FCMD[Note1]	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[Note2]	High-speed on-chip oscillator clock frequency correction circuit operation control/status
0	High-speed on-chip oscillation frequency correction circuit operation stop/in stop progress
1	High-speed on-chip oscillation frequency correction circuit operation start/in operation
In continuous action mode, the software writes 0 to stop the action. In interval action mode, the hardware clears the FCST bit when the correction is complete.	

Note 1: When the FCST bit is 1, rewriting the FCMD bit is disabled.

Note 2: When writing 1 to the FCST bit, first confirm that the current value of the FCST bit is 0 before writing 1. Due to the hardware clear priority, when writing 1 to the FCST bit immediately after the interval action is completed (when the high-speed on-chip oscillator frequency correction completion interrupt is generated), the operation should be performed after at least 1 cycle of F_{HOCO} after the high-speed on-chip oscillator frequency correction completion interrupt is generated.

Notice:

1. After writing 0 to the FCST bit (high-speed on-chip oscillator frequency correction circuit stops), F_{HOCO} prohibits writing 1 to the FCST bit (high-speed on-chip oscillator frequency correction circuit starts) for 2 cycles.
2. Bit5~1 must write 0.

4.7.3 Description of operation

4.7.3.1 Overview

The high-speed on-chip oscillator frequency correction function generates a correction cycle based on the subsystem clock (F_{SUB}), measures the frequency of the high-speed on-chip oscillator, and corrects the frequency accuracy of the high-speed on-chip oscillator in real time. The clock adjustment repeats the operation of the frequency measurement stage and the frequency correction stage. The correction algorithm is performed in the frequency measurement stage, and the correction value reflecting the result of the correction algorithm is saved in the frequency correction stage.

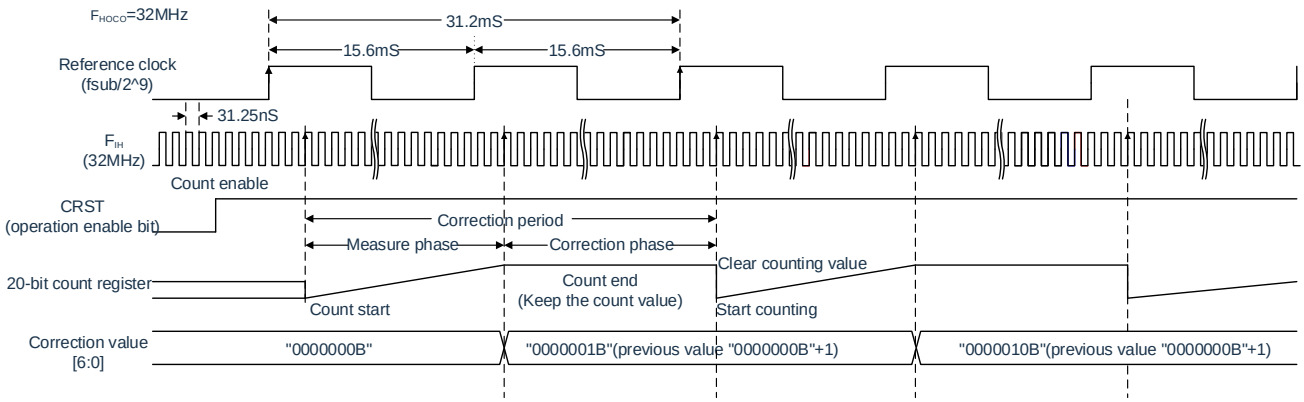
Table 4-25 is the high-speed on-chip oscillator input frequency and calibration cycle. Figure 4-8 is the action block diagram of the high-speed on-chip oscillation frequency correction function.

Table 4-25: High-speed on-chip oscillator input frequency and calibration cycle

F_{HOCO} (MHz)	FRQSEL4-FRQSEL3[Note]	Correction cycle (ms)
64	11	31.2 (Frequency measurement stage + frequency calibration stage)
48	10	

In the frequency measurement phase of the calibration cycle, the frequency of the high-speed on-chip oscillator is calibrated using high-speed on-chip oscillator counting based on the result of the magnitude of the counted value and the expected value.

Figure 4-8: Timing diagram of high-speed on-chip oscillation frequency correction (detailed)



Note: FRQSEL4-FRQSEL3 are bit4-bit3 of option byte 00C2H.

Notice: The basic actions of continuous action mode and interval action mode are the same. The difference is whether the clearing of FCST bit is controlled by software or hardware. In addition, only the system reset can clear the correction value.

(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

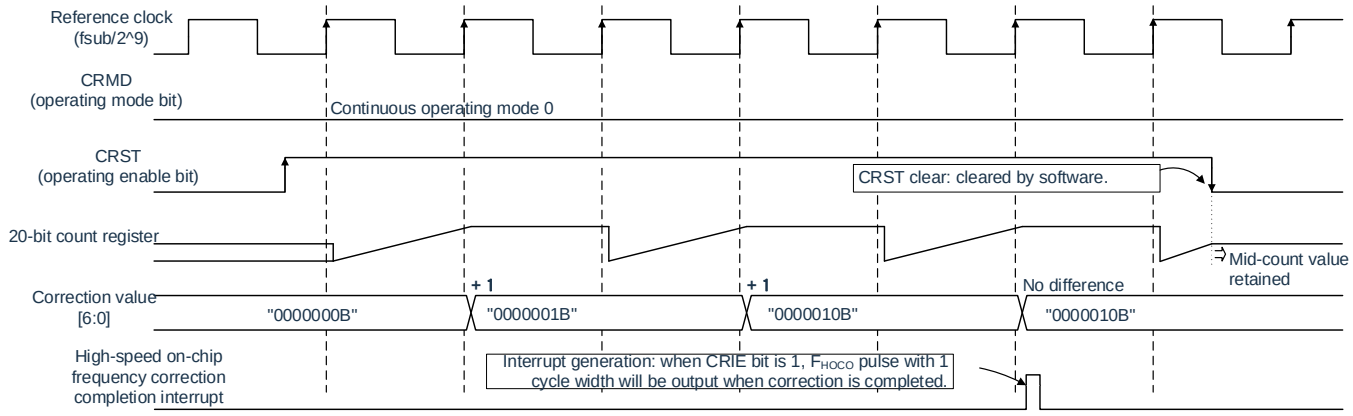
When the count value is smaller than the expected value: Correction value + 1

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 highspeed on-chip oscillator clock frequency correction function is stopped.

Figure 4-9 shows the continuous operating mode timing.

Figure 4-9: 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

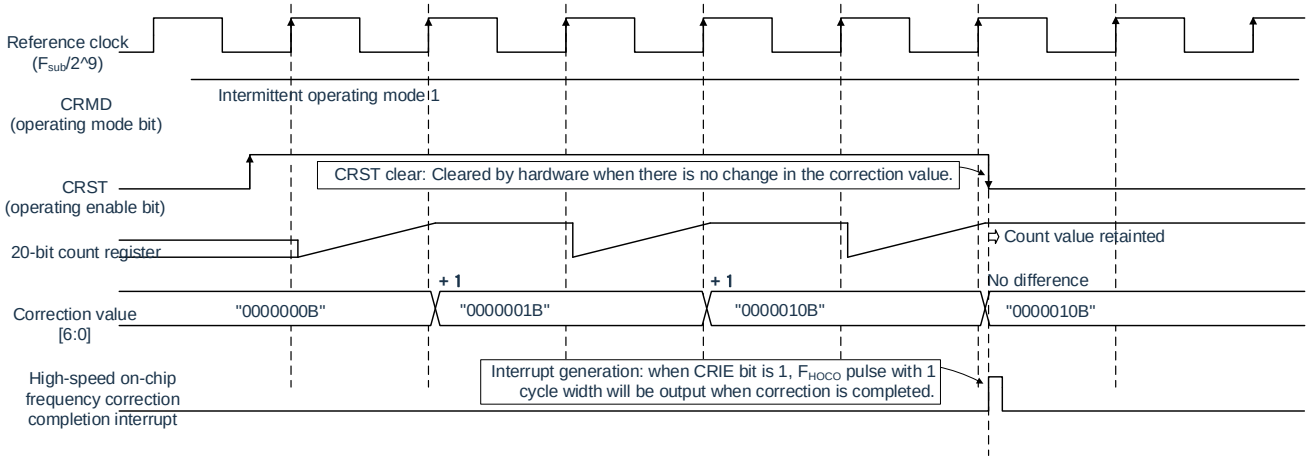
When the count value is smaller than the expected value: Correction value + 1

When the count value is in the range of the expected value: The correction value is retained (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 highspeed on-chip oscillator clock frequency correction operation is stopped after high-speed on-chip oscillator clock frequency correction is completed.

Figure 4-10 shows the intermittent operating mode timing.

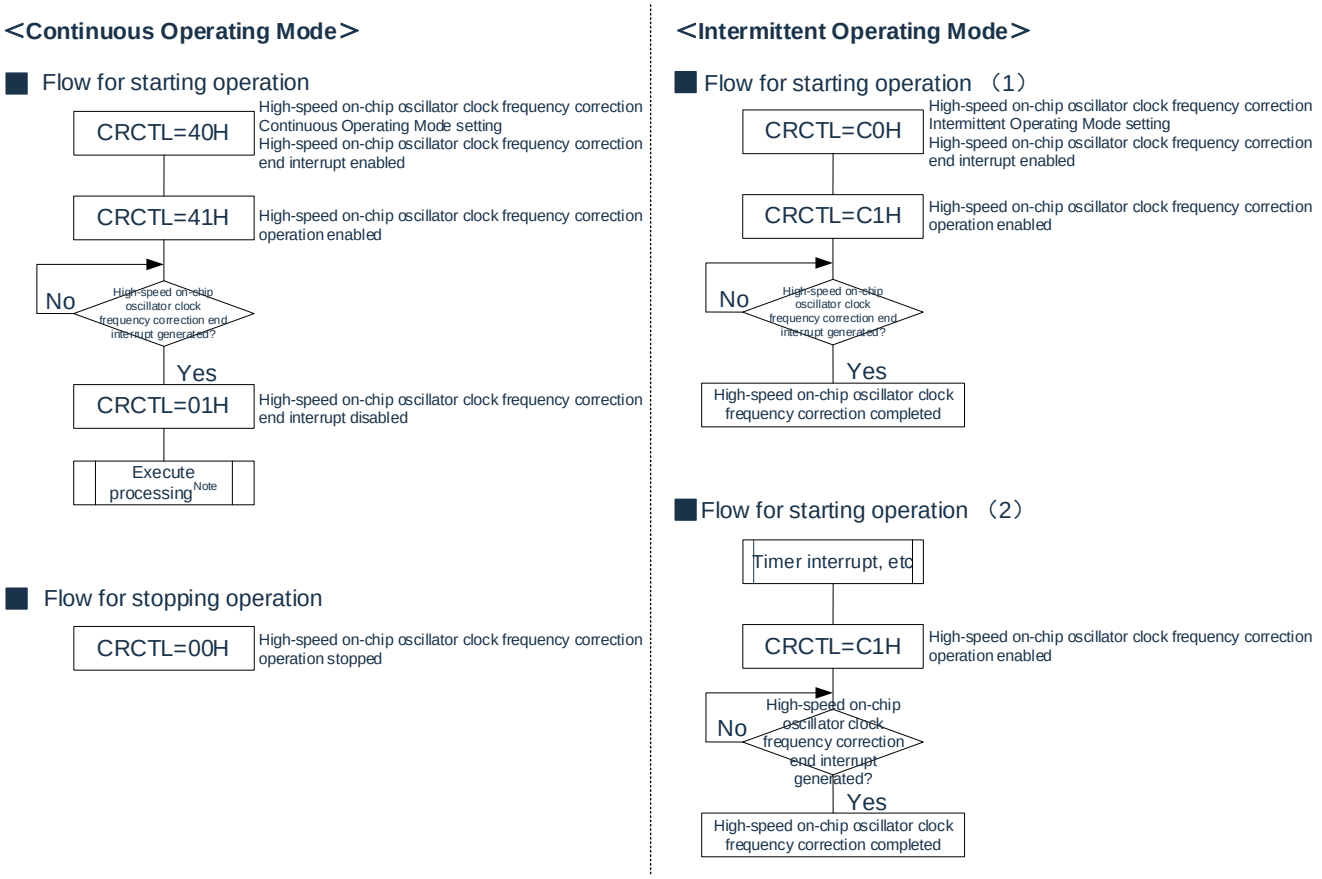
Figure 4-10: Intermittent operating mode timing



4.7.3.2 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 4-11: 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.

4.7.4 Usage notes

4.7.4.1 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_{HOCO} cycle after a high-speed on-chip oscillator clock frequency correction end interrupt is generated.

4.7.4.2 Operation during reset

The high-speed on-chip oscillator clock frequency correction function must be stopped before entering deep sleep.

Chapter 5 General-Purpose Timer Unit Timer4

This product is equipped with two general-purpose timer units, each containing four channels.

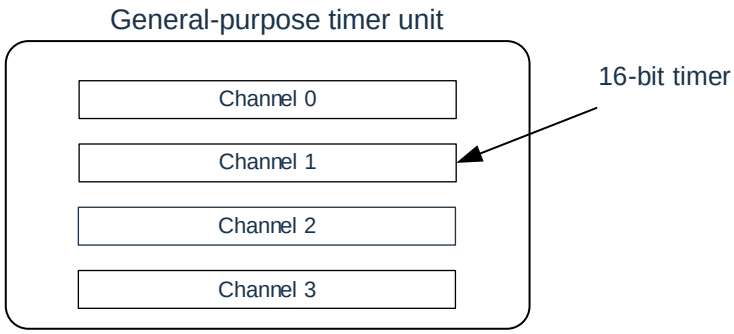
Description:

1. The symbol "m" in the following part of this chapter stands for the unit number. This product is equipped with two general-purpose timers, Timer4, so m=0, 1.

2. The symbol "n" in the following part of this chapter stands for the channel number (n=0~3 in this chapter).

Each general-purpose timer unit has four 16-bit timers.

Each 16-bit timer is referred to as a "channel" and can be used individually as a stand-alone timer or in combination with multiple channels for advanced timer functions.



For details of each function, please refer to the following table.

Independent channel operation functions	Multi-channel linkage operation functions
• Interval timer (refer to 5.8.1) • Square wave output (refer to 5.8.1) • External event counter (refer to 5.8.2) • Frequency divider (refer to 5.8.3) • Measurement of input pulse interval (refer to 5.8.4) • Measurement of the high-/low-level width of the input signal (refer to 5.8.5) • Delay counter (refer to 5.8.6)	• Single trigger pulse output (refer to 5.9.1) • PWM output (refer to 5.9.2) • Multiple PWM outputs (refer to 5.9.3)

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)

5.1 Function of general-purpose timer unit

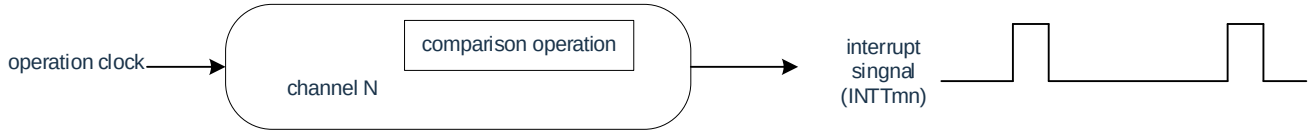
The general-purpose timer unit has the following functions.

5.1.1 Independent channel operation function

The independent channel operation function is a function that enables independent use of any channel without being affected by other channel operation modes.

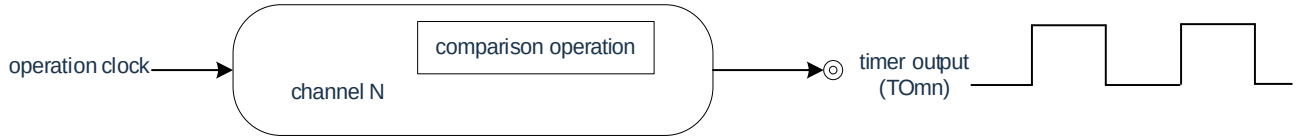
(1) Interval timer

Each timer of a unit can be used as a reference timer that generates an interrupt (INTTMmn) at fixed intervals.



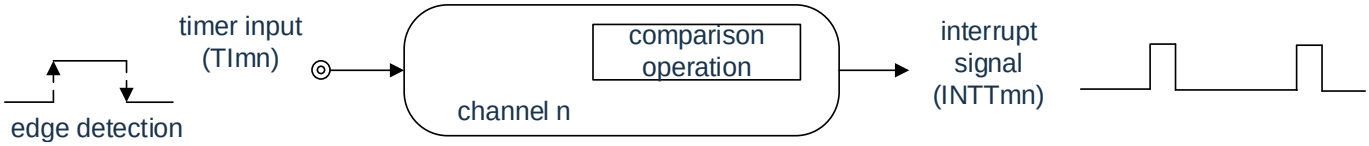
(2) Square wave output

A toggle operation is performed each time INTTMmn interrupt is generated and a square wave with a duty cycle of 50% is output from a timer output pin (TOmn).



(3) External event counter

The valid edge of the input signal of the timer input pin (TImn) is counted, and if the specified number of times is reached, it can be used as an event counter for generating interrupts.



(4) Frequency divider function (channel 0 of unit 0 only)

The input clock from the timer input pin (TI00) is divided and then output from the output pin (TO00).

(5) Measurement of input pulse interval

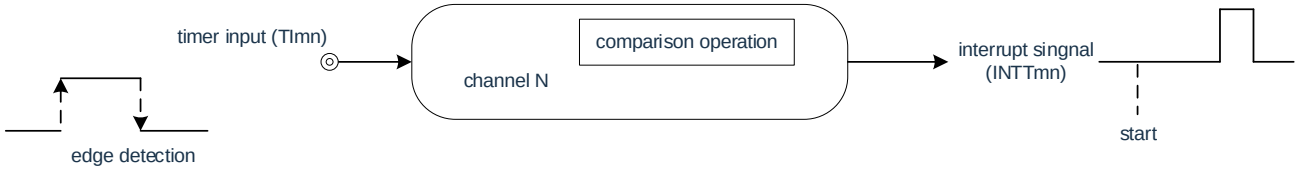
The interval between input pulses is measured by starting counting at the active edge of the input pulse signal at the timer input pin (TImn) and capturing the count value at the active edge of the next pulse.

(6) Measurement of the high-/low-level width of the input signal

The high- and low-level width of the input signal is measured by starting the count on one edge of the input signal at the timer input pin (TImn) and capturing the count value on the other edge.

(7) Delay counter

Counting begins on the active edge of the input signal to the timer input pin (TImn) and an interrupt is generated after an arbitrary delay period.



Remark:

1. m: unit number (m=0, 1) n: channel number (n=0~3)
2. Please refer to "Chapter 2 Port Function" for the configurable timer input/output pins of channel 0~3.

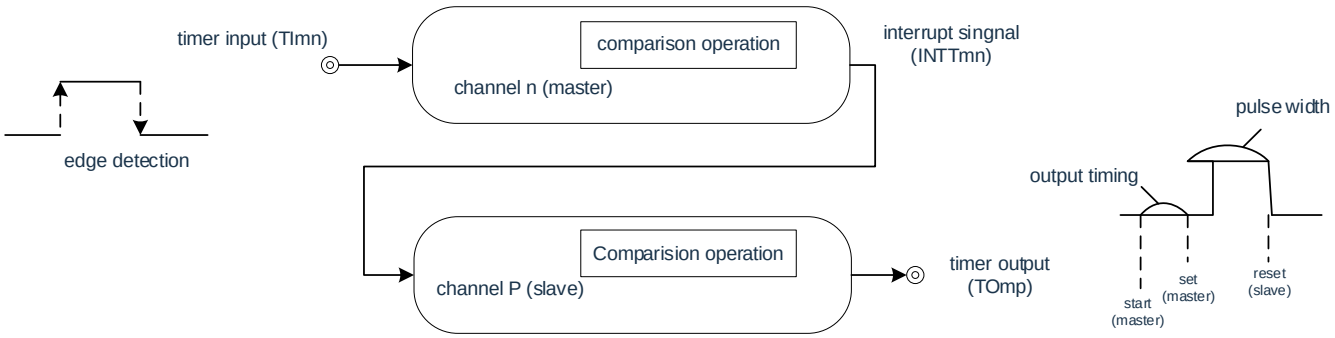
5.1.2 Multi-channel linkage operation functions

The multi-channel linked operation function is a combination of a master channel (the reference timer for the master control cycle) and a slave channel (a timer that operates in compliance with the master channel).

The multi-channel linkage operation function can be used as the following modes.

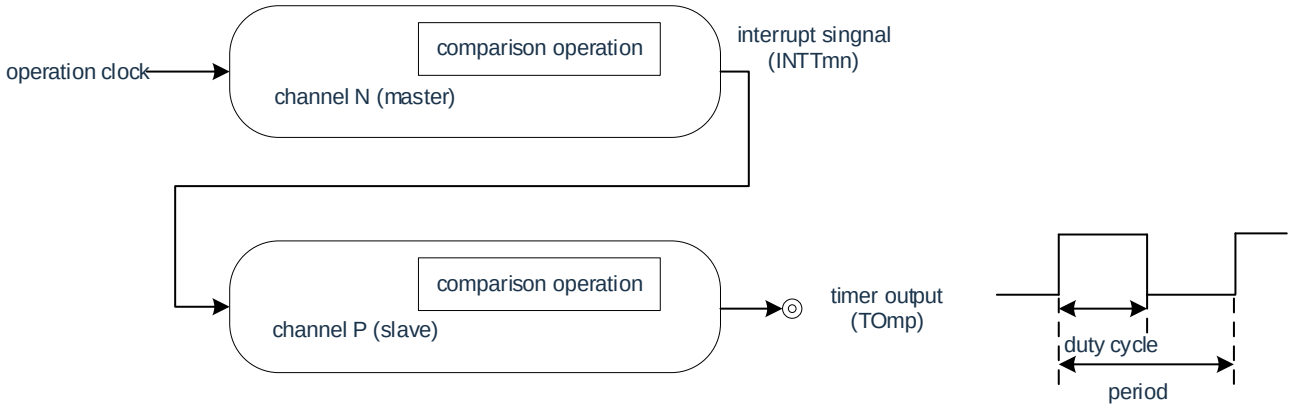
(1) Single trigger pulse output

Using the 2 channels in pairs, a single trigger pulse with arbitrary output timing and pulse width can be generated.



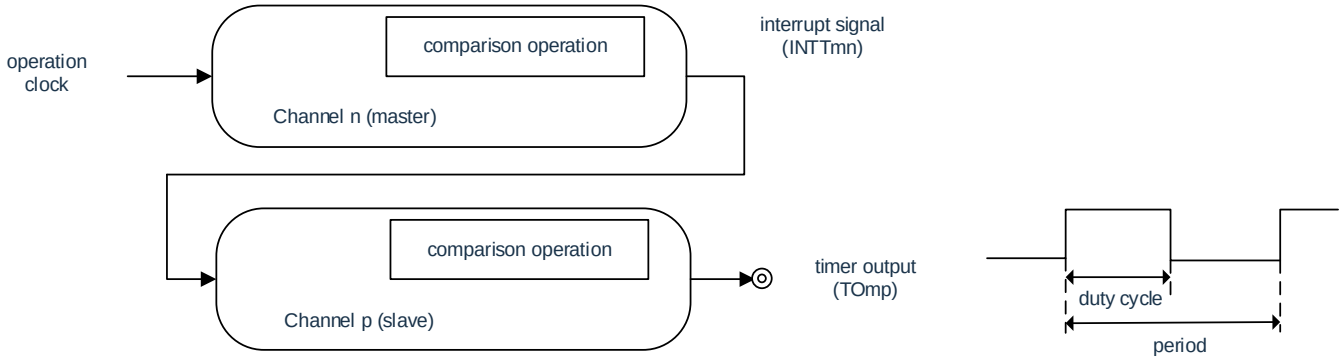
(2) PWM (Pulse Width Modulation) output

Using the 2 channels in pairs, pulses with arbitrary period and duty cycle can be generated.



(3) Multiple PWM (Pulse Width Modulation) outputs

The PWM function can be extended to generate up to 3 PWM signals of any duty cycle with a fixed period using one master channel and multiple slave channels.



Remark:

1. Note: Please refer to "5.4.1 Basic rules of multi-channel linkage operation function" for the rule details of multi-channel linkage operation function.
2. 2. m: unit number (m=0,1) n: channel number (n=0 ~ 3) p, q: slave channel number (n < p < q ⩽ 3)

5.1.3 8-bit timer operation function (channels 1 and 3 of unit 0 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.

Remark:

1. There are several rules for using 8-bit timer operation function.
2. For details, see 5.4.2 Basic rules of 8-bit timer operation function (channels 1 and 3 only).

5.1.4 LIN-bus supporting function (channel 3 of unit 0 only)

The received signal in the LIN-bus communication is checked by the general-purpose timer unit to see if it fits the LIN-bus communication table.

(1) Detection of wake-up signals

The low-level width is measured by starting a count on the falling edge of the input signal at the UART0 serial data input pin (RxD0) and capturing the count value on the rising edge. If the low-level width is greater than or equal to a fixed value, it is considered a wake-up signal.

(2) Detection of break field

After a wake-up signal is detected, the low-level width is measured by counting on the falling edge of the input signal at the UART0 serial data input pin (RxD0) and capturing the count value on the rising edge. If the low-level width is greater than or equal to a fixed value, it is considered a break field.

(3) Measurement of sync field pulse width

After the sync field is detected, the low-level width and high-level width of the input signal at the UART0 serial data input pin (RxD0) are measured. Based on the bit space of the sync field measured in this way, the baud rate is calculated.

Remark: Refer to “5.3.13: Input switching control register (ISC)” and "5.8.5: Operation as input signal high and low levelwidth measurement” for the operation setting of LIN-bus support functions.

5.2 Structure of general-purpose timer unit

The general-purpose timer unit consists of the following hardware.

Table 5-1: Structure of general-purpose timer unit

Item	Structure
Counter	Timer count register mn (TCRmn)
Register	Timer data register mn (TDRmn)
Timer input	TI00~TI03 Note 1, TI10~TI13 Note 1
Timer output	TO00~TO03 Note 1, TO10~TO13 Note 1, output control circuit
Control registers	＜Registers of unit setting section＞ • Peripheral enable register 0 (PER0) • Timer clock select register m (TPSm) • Timer channel enable status register m (TEm) • Timer channel start register m (TSm) • Timer channel stop register m (TTm) • Timer input select register 0 (TIOS0) Note 2 • Timer output enable register m (TOEm) • Timer output register m (TOm) • Timer output level register m (TOLm) • Timer output mode register m (TOMm)
	＜Registers of each channel＞ • Timer mode register mn (TMRmn) • Timer status register mn (TSRmn) • Noise filter enable register 1, 2 (NFEN1, NFEN2) • Port mode control register (PMCxx) Note 3 • Port mode register (PMxx) Note 3 • Port output multiplexing function configuration register (PxxCFG) Note 3 • Port input multiplexing function configuration register (TI1XPCFG) Note 3

Note 1: The input/output pins of general-purpose timer unit 0 are multiplexed to fixed ports, and the timer input/output pins of channels 0 to 3 of general-purpose timer unit 1 can be configured to each port except RESETB. For details, refer to “Chapter 2 Port Function”.

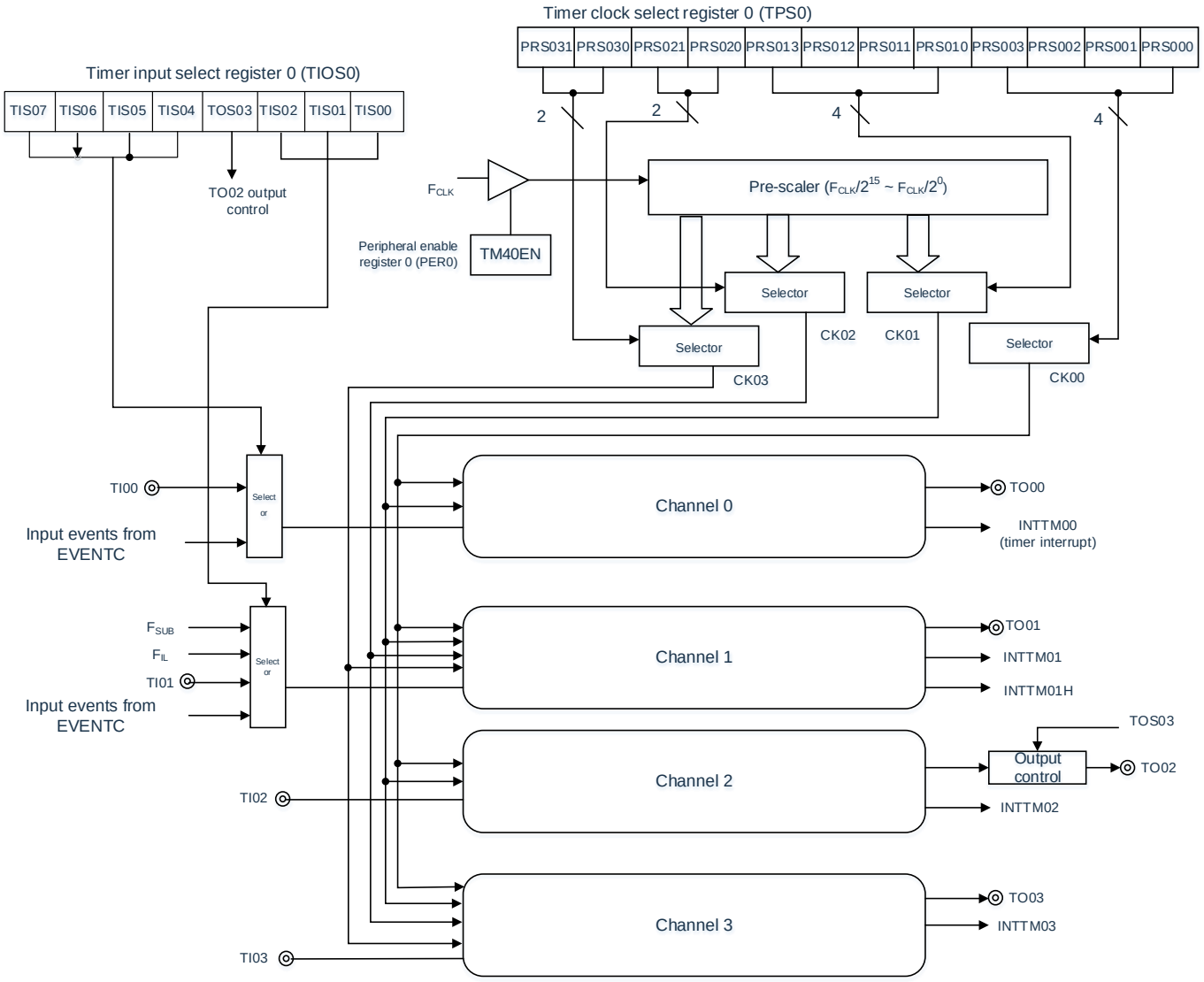
Note 2: Only for channel selection of unit 0.

Note 3: Timer input/output pin configuration for channel 0~3. For details, please refer to “Chapter 2 Port Function”.

Remark: m: unit number (m=0, 1) n: channel number (n=0~3)

The block diagram of the general-purpose timer unit is shown in Figure 5-1.

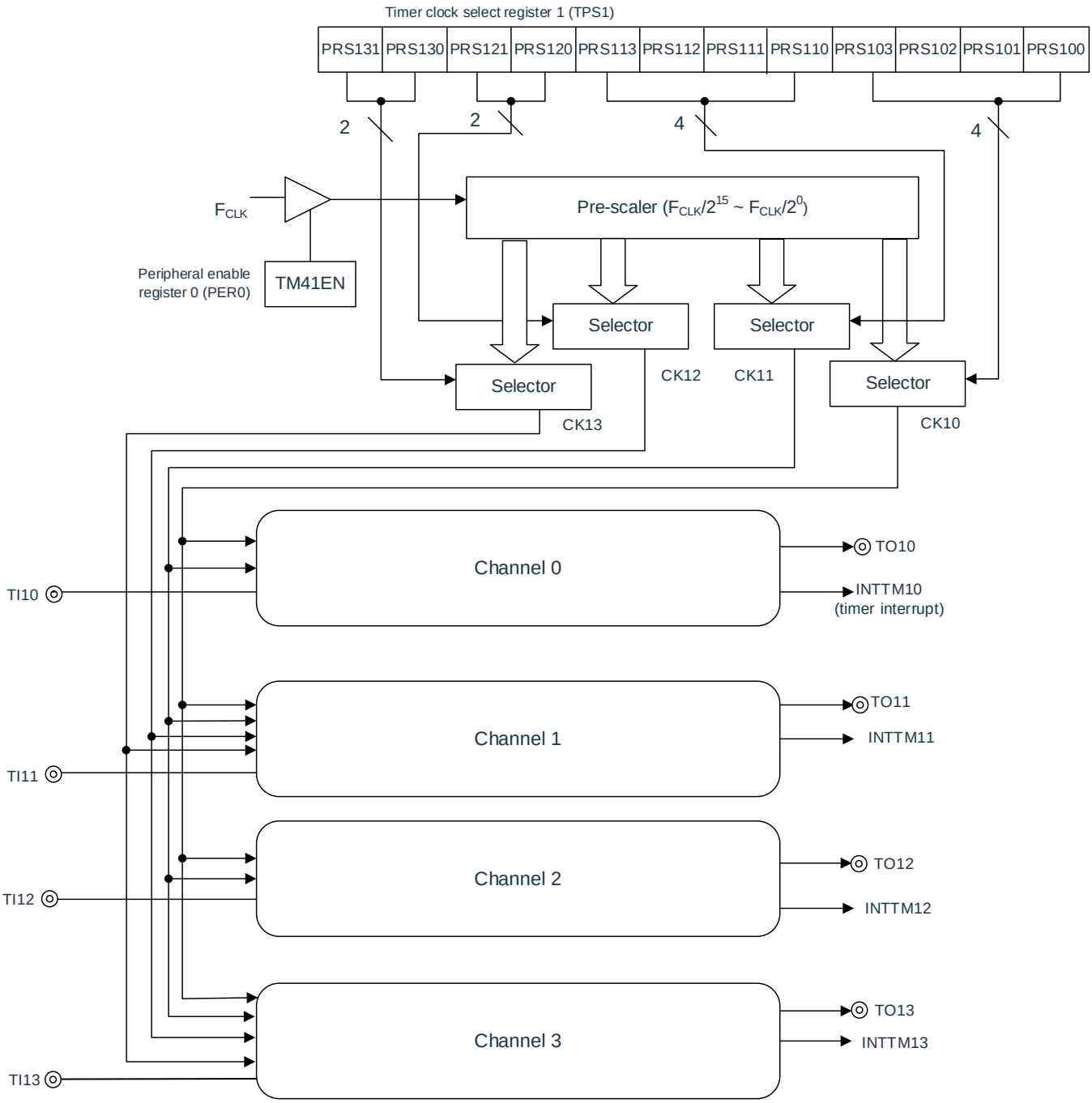
Figure 5-1: Overall block diagram of general-purpose timer unit 0



Remark: F_{SUB}: Subsystem clock frequency

F_{IL}: Low-speed on-chip oscillator clock frequency

Figure 5-2: Overall block diagram of general-purpose timer unit 1



5.2.1 Register list of general-purpose timer unit 0

Register base address of unit 0: 0x40041C00

Offset address	Register name	R/W	Bit width	Reset value
0x180	TCR00	R	16	FFFFH
0x182	TCR01	R	16	FFFFH
0x184	TCR02	R	16	FFFFH
0x186	TCR03	R	16	FFFFH
0x190	TMR00	R/W	16	0000H
0x192	TMR01	R/W	16	0000H
0x194	TMR02	R/W	16	0000H
0x196	TMR03	R/W	16	0000H
0x1A0	TSR00	R	16	0000H
0x1A0	TSR00L	R	8	00H
0x1A2	TSR01	R	16	0000H
0x1A2	TSR01L	R	8	00H
0x1A4	TSR02	R	16	0000H
0x1A4	TSR02L	R	8	00H
0x1A6	TSR03	R	16	0000H
0x1A6	TSR03L	R	8	00H
0x1B0	TE0	R	16	0000H
0x1B0	TE0L	R	8	00H
0x1B2	TS0	R/W	16	0000H
0x1B2	TS0L	R/W	8	00H
0x1B4	TT0	R/W	16	0000H
0x1B4	TT0L	R/W	8	00H
0x1B6	TPS0	R/W	16	0000H
0x1B8	TO0	R/W	16	0000H
0x1B8	TO0L	R/W	8	00H
0x1BA	TOE0	R/W	16	0000H
0x1BA	TOE0L	R/W	8	00H
0x1BC	TOL0	R/W	16	0000H
0x1BC	TOL0L	R/W	8	00H
0x1BE	TOM0	R/W	16	0000H
0x1BE	TOM0L	R/W	8	00H
0x318	TDR00	R/W	16	0000H
0x31A	TDR01	R/W	16	0000H
0x31A	TDR01L	R/W	8	00H
0x31B	TDR01H	R/W	8	00H
0x364	TDR02	R/W	16	0000H
0x366	TDR03	R/W	16	0000H
0x366	TDR03L	R/W	8	00H
0x367	TDR03H	R/W	8	00H

5.2.2 Register list of general-purpose timer unit 1

Register base address of unit 1: 0x40042000

Offset address	Register name	R/W	Bit width	Reset value
0x180	TCR10	R	16	FFFFH
0x182	TCR11	R	16	FFFFH
0x184	TCR12	R	16	FFFFH
0x186	TCR13	R	16	FFFFH
0x190	TMR10	R/W	16	0000H
0x192	TMR11	R/W	16	0000H
0x194	TMR12	R/W	16	0000H
0x196	TMR13	R/W	16	0000H
0x1A0	TSR10	R	16	0000H
0x1A0	TSR10L	R	8	00H
0x1A2	TSR11	R	16	0000H
0x1A2	TSR11L	R	8	00H
0x1A4	TSR12	R	16	0000H
0x1A4	TSR12L	R	8	00H
0x1A6	TSR13	R	16	0000H
0x1A6	TSR13L	R	8	00H
0x1B0	TE1	R	16	0000H
0x1B0	TE1L	R	8	00H
0x1B2	TS1	R/W	16	0000H
0x1B2	TS1L	R/W	8	00H
0x1B4	TT1	R/W	16	0000H
0x1B4	TT1L	R/W	8	00H
0x1B6	TPS1	R/W	16	0000H
0x1B8	TO1	R/W	16	0000H
0x1B8	TO1L	R/W	8	00H
0x1BA	TOE1	R/W	16	0000H
0x1BA	TOE1L	R/W	8	00H
0x1BC	TOL1	R/W	16	0000H
0x1BC	TOL1L	R/W	8	00H
0x1BE	TOM1	R/W	16	0000H
0x1BE	TOM1L	R/W	8	00H
0x318	TDR10	R/W	16	0000H
0x31A	TDR11	R/W	16	0000H
0x31A	TDR11L	R/W	8	00H
0x31B	TDR11H	R/W	8	00H
0x364	TDR12	R/W	16	0000H
0x366	TDR13	R/W	16	0000H
0x366	TDR13L	R/W	8	00H
0x367	TDR13H	R/W	8	00H

5.2.3 Timer count register mn (TCRmn)

The TCRmn register is a 16-bit read-only register that counts the count clock. The count is incremented or decremented synchronously with the rising edge of the count clock.

The operation mode is selected by the MDmn3 to MDmn0 bits of the Timer Mode Register mn (TMRmn) to switch between incremental and decremental counting (refer to “5.3.3: Timer Mode Register mn (TMRmn)").

Table 5-2: Table of timer count register mn (TCRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TCRmn																

The count value can be read by reading the timer count register mn (TCRmn).

In the following cases, the count value becomes "FFFFH".

(1) When a reset signal is generated

(2) When clearing the TM4mEN bit of the peripheral enable register 0 (PER0)

(3) At the end of the count of the slave channel in PWM output mode

(4) At the end of the count of the slave channel in delayed count mode

(5) At the end of counting of master/slave channels in single trigger pulse output mode

(6) At the end of the count of the slave channel in the multiple PWM output mode

In the following cases, the count value becomes "0000H".

(1) When input starts triggering in capture mode

(2) At the end of the capture in capture mode

Remark:

1. Even if the TCRmn register is read, the count value is not captured to the timer data register mn (TDRmn).
2. 2. m: unit number (m=0, 1) n: channel number (n=0~3)

As shown below, the read values of the TCRmn register vary depending on the operating mode and operating state.

Table 5-3: The read value of the Timer Count Register mn (TCRmn) in various operating modes

Operation mode	Counting method	Timer Count Register mn (TCRmn) read value[Note1]			
		Value if the operation mode was changed after releasing reset	Counting pause Value at (TTmn = 1)	Counting pause (TTmn=1) after changing the value of the operating mode	Wait after a single count The value at the start of the trigger
Interval timer mode	Count down	FFFFH	value when stopped	undefined	-
Capture Mode	Count up	0000H	value when stopped	undefined	-
Event counter mode	Count down	FFFFH	value when stopped	undefined	-
Single count mode	Count down	FFFFH	value when stopped	undefined	FFFFH
Capture & Single Count Mode	Count up	0000H	value when stopped	undefined	TDRmn register capture value +1

Note 1: It indicates the read value of the TCRmn register when channel n is in the timer stop state (TEmn=0) and the count enable state (Tsmn=1). Hold this value in the TCRmn register until counting starts.

Remark: m: unit number (m=0, 1) n: channel number (n=0~3)

5.2.4 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.

Table 5-4: Table of timer data registers mn (TDRmn) (n=0, 2)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TDRmn																

Table 5-5: Table of timer data registers mn (TDRmn) (n=1, 3)

(TDR01H supports 8-bit operation) (TDR01L supports 8-bit operation)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TDRmn																

(1) 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.

(2) 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:

1. Even if a capture trigger signal is input, the TDRmn register set to the compare function does not perform capture operation.
2. m: unit number (m=0, 1) n: channel number (n=0~3)

5.3 Registers for controlling general-purpose timer unit

The registers that control the general-purpose timer unit are as follows:

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 output selection register (TIOS0)

Timer output enable register m (TOEm)

Timer output register m (TOm)

Timer output level register m (TOLm)

Timer output mode register m (TOMm)

Noise filter enable register 1 (NFEN1)

Noise filter enable register 2 (NFEN2)

Port mode control register (PMCxx)

Port mode register (PMxx)

Port multiplexing function configuration register (PxxCFG)

Notice: The assigned registers and bits vary from product to product. The initial values must be set for unassigned bits.

Remark: Unit number (m=0, 1) n: channel number (n=0~3)

5.3.1 Peripheral enable register 0 (PER0)

The PER0 register is the register that sets whether to enable or disable the supply of clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocks to hardware that is not in use.

To use general-purpose timer unit 0, bit0 (TM40EN) must be set to "1". The PER0 register is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of the PER0 register changes to "00H".

Table 5-6: Table of peripheral enable register 0 (PER0)

Address: 0x40020420 After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
PER0	RTCEN	0	ADCEN	IICA0EN	SAU1EN	SAU0EN	TM41EN	TM40EN

TM40EN	Control of the input clock of general-purpose timer unit 0
0	Stop to supply the input clock. • The SFR used by general-purpose timer unit 0 cannot be written. • General-purpose timer unit 0 is in the reset state.
1	Supply the input clock. • The SFR used by general-purpose timer unit 0 can read and write.

TM41EN	Control of the input clock of general-purpose timer unit 1
0	Stop to supply the input clock. • The SFR used by general-purpose timer unit 1 cannot be written. • General-purpose timer unit 1 is in the reset state.
1	Supply the input clock. • The SFR used by general-purpose timer unit 1 can read and write.

Notice: To set the general-purpose timer unit, the following registers must be set with the TM4mEN bit at "1". When the TM4mEN bit is "0", the values of the Timer Array Unit's control registers are initialized, and write operations are ignored (timer input/output select register 0 (TIOS0), noise filter enable register 1 (NFEN1), noise filter enable register 2 (NFEN2), port mode control register (PMCx), port mode register (PMx), and port multiplexing function configuration register (PxxCFG) are excluded).

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)

5.3.2 Timer clock select register m (TPSm)

The TPSm register is a 16-bit register that selects the two or four common operating clocks (CKm0, CKm1, CKm2, CKm3) provided to each channel. CKm0 is selected via bits 3~0 of the TPSm register, and CKm1 is selected via bits 7~4 of the TPSm register. In addition, only channel 1 and channel 3 can select CKm2 and CKm3, and CKm2 is selected via bits 9~8 of the TPSm register, and CKm3 is selected via bits 13 and 12 of the TPSm register.

The TPSm register in timer operation can only be rewritten in the following cases.

1) If the PRSm00 to PRSm03 bits can be rewritten (n = 0 to 3):
   All channels for which CKm0 is selected as the operation clock (CKSmn1, CKSmn0 = 0, 0) are stopped (TEmn = 0).
2) If the PRSm10 to PRSm13 bits can be rewritten (n = 0 to 3):
   All channels for which CKm2 is selected as the operation clock (CKSmn1, CKSmn0 = 0, 1) are stopped (TEmn = 0).
3) 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).
4) 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.

Table 5-7: Table of timer clock select register m (TPSm) (1/2)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TPSm			PRSm31	PRSm31			PRSm21	PRSm20	PRSm13	PRSm12	PRSm11	PRSm10	PRSm03	PRSm02	PRSm01	PRSm00

PRSmk3	PRSmk2	PRSmk1	PRSmk0	Selection of operating clock (CKmk)[Note] (k=0, 1)
0	0	0	0	F_{CLK}
0	0	0	1	$F_{CLK}/2$
0	0	1	0	$F_{CLK}/2^2$
0	0	1	1	$F_{CLK}/2^3$
0	1	0	0	$F_{CLK}/2^4$
0	1	0	1	$F_{CLK}/2^5$
0	1	1	0	$F_{CLK}/2^6$
0	1	1	1	$F_{CLK}/2^7$
1	0	0	0	$F_{CLK}/2^8$
1	0	0	1	$F_{CLK}/2^9$
1	0	1	0	$F_{CLK}/2^{10}$
1	0	1	1	$F_{CLK}/2^{11}$
1	1	0	0	$F_{CLK}/2^{12}$
1	1	0	1	$F_{CLK}/2^{13}$
1	1	1	0	$F_{CLK}/2^{14}$
1	1	1	1	$F_{CLK}/2^{15}$

Note: Bit 15, 14, 11 and 10 must be set to "0".

Notice:

1. In case of changing the clock selected as F_{CLK} (changing the value of the system clock control register (CKC)), the general-purpose timer unit must be stopped (TTm=0,100FH). The general-purpose timer unit needs to be stopped even when the operation clock (F_{MCK}) is selected or when the active edge of the TImn pin input signal is used.
2. If F_{CLK} (undivided) is selected as the operation clock (CKmk) and TDRnm is set to "0000H" (n=0, 1, m=0~3), the interrupt request of general-purpose timer unit cannot be used.
3. The clock waveform selected by the TPSm register is high for only 1 F_{CLK} cycle from the rising edge. For details, refer to “5.5.1 Counting Clock (F_{TCLK})”.
4. F_{CLK}: CPU/peripheral hardware clock frequency.

Table 5-7: Table of timer clock select register m (TPSm) (2/2)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TPSm			PRS m31	PRS m31			PRS m21	PRS m20	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00

PRSm21	PRSm20	Selection of operation clock (CKm2) Note
0	0	$F_{CLK}/2$
0	1	$F_{CLK}/2^2$
1	0	$F_{CLK}/2^4$
1	1	$F_{CLK}/2^6$

PRSm31	PRSm30	Selection of operation clock (CKm3) Note
0	0	$F_{CLK}/2^8$
0	1	$F_{CLK}/2^{10}$
1	0	$F_{CLK}/2^{12}$
1	1	$F_{CLK}/2^{14}$

If Channel 1 and Channel 3 are used in 8-bit timer mode and CKm2 and CKm3 are used as the operating clocks, the interval time shown in the following table can be achieved by using the interval timer function.

Table 5-8: Interval time that can be set by operation clocks CKSm2 and CKSm3

Clock		Interval time Note (F_{CLK}=32MHz)			
		10us	100us	1ms	10ms
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: Bits 15, 14, 11, 10 must be set to "0".

Notice:

1. The general-purpose timer unit (TTm=000FH) must be stopped if the clock selected as F_{CLK} is changed (the value of the system clock control register (CKC) is changed). It is necessary to stop the general-purpose timer unit even when the operation clock (F_{MCK}) is selected or when the active edge of the input signal to the TImn pin is selected.
2. ○The margin is within 5%.
3. F_{CLK}: CPU/peripheral hardware clock frequency
4. Refer to “5.5.1 Counting Clock (F_{TCLK})” for details of the $F_{CLK}/2^r$ waveform selected for the TPSm register.

5.3.3 Timer mode register mn (TMRmn)

The TMRmn register is the register that sets the operation mode of channel n. It performs the selection of the operation clock (F_{MCK}), the selection of the count clock, the selection of master/slave, the selection of the 16-bit/8-bit timer (channel 1 and channel 3 of unit 0 only), the setting of the start trigger and the capture trigger, the selection of the effective edge of the timer input, and the operation modes (interval, capture, event counter, single count, capture & single count) settings.

It is prohibited to rewrite the TMRmn register during operation (TEmn=1). However, bit7 and bit6 (CISmn1, CISmn0) can be rewritten during part of the function operation (TEmn=1) (for details, refer to "5.8 Independent Channel Operation Function of General-Purpose Timer Unit" and "5.9 Multi-Channel Operation Function of General-Purpose Timer Unit ").

The TMRmn register is set by a 16-bit memory manipulation instruction. After a reset signal is generated, the value of the TMRmn register changes to "0000H".

Remark: The bit11 of the TMRmn register varies from channel to channel.

TMRm2: MASTERmn bit (n=2).

TMRm1, TMRm3: SPLITmn bits (n=1, 3)

TMRm0: Fixed to "0".

Table 5-9: Table of timer mode register mn (TMRmn)(1/4)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn (n=2)	CKS mn1	CKS mn0	0	CCS mn	MAS TERmn	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0		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	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)	CKS mn1	CKS mn0	0	CCS mn	0Note1	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0		0	MD mn3	MD mn2	MD mn1	MD mn0

CKSmn1	CKSmn0	Selection of channel n operation clock (F_{MCK})
0	0	The operation clock CKm0 set by the timer clock selection register m (TPSm).
0	1	The operation clock CKm2 set by the timer clock selection register m (TPSm).
1	0	The operation clock CKm1 set by the timer clock selection register m (TPSm).
1	1	The operation clock CKm3 set by the timer clock selection register m (TPSm).
The operation clock (F_{MCK}) is used for edge detection circuits. The sample clock and count clock (F_{TCLK}) are generated by setting the CCSmn bit. Only Channel 1 and Channel 3 can select operation clocks CKm2 and CKm3.		

CCSmn	Selection of channel n count clock (F_{TCLK})
0	CKSmn0 bit and CKSmn1 bit specified operation clock (F_{MCK})
1	The active edge of the TImn pin input signal Unit 0 status: Channel 0: The active edge of the input signal selected by TIS0 Channel 1: The active edge of the input signal selected by TIS0
Counting clocks (F_{TCLK}) are used in counters, output control circuits, and interrupt control circuits.	

Note 1: Bit 11 is a read-only bit, fixed to "0", and write is ignored.

Notice: Bits 13, 5 and 4 must be set to "0".

Remark:

1. To change the clock selected as F_{CLK} (change the value of the system clock control register (CKC)), the Timer Array Unit (TTm=00FFH) must be stopped even if the operation clock (F_{MCK}) specified by the CKSmn0 bit and the CKSmn1 bit, or the active edge of the input signal to the TImn pin, is selected as the count clock (F_{TCLK}).
2. m: unit number (m=0, 1) n: channel number (n=0~3)

Table 5-9: Table of timer mode register mn (TMRmn)(2/4)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn (n=2)	CKS mn1	CKS mn0	0	CCS mn	MAS TERmn	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0		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	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)	CKS mn1	CKS mn0	0	CCS mn	0Note1	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0		0	MD mn3	MD mn2	MD mn1	MD mn0

(bit11 of TMRmn (n=2))

MASTERmn	Selection of independent channel operation/multi-channel linked operation (slave or master) for channel n
0	Used as a slave channel for independent or multi-channel linked operation functions.
1	Used as a master control channel for the multi-channel linked operation function.
Only channel 2 can be set as the master channel (MASTERmn=1). Channel 0 is fixed to "0" (since channel 0 is the channel with the highest bit, it is used as the master channel regardless of the setting of this bit). For channels used as independent channel operation functions, set the MASTERmn bit to "0".	

(bit11 of TMRmn (n=1, 3))

SPLITmn	Operation selection of 8-bit timer/16-bit timer for channel 1 and channel 3
0	Used as a 16-bit timer. (Used as a slave channel for independent channel operation or multi-channel linkage operation)
1	Used as an 8-bit timer.

STSmn2	STSmn1	STSmn0	Start trigger and capture trigger settings for channel n
0	0	0	Only software triggering is active at the start (no other trigger source is selected).
0	0	1	Use the active edge of the TImn pin input for start triggering and capture triggering.
0	1	0	Use the double edges of the TImn pin input for start triggering and capture triggering respectively.
1	0	0	Use interrupt signals from the master channel (in the case of slave channels with multi-channel linkage operation function).
Other than the above			Settings are disabled.

Note 1: Bit11 is a read-only bit, fixed to “0”, and write is ignored.

Remark: m: unit number (m=0, 1) n: channel number (n=0~3)

Table 5-9: Table of timer mode register mn (TMRmn)(3/4)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn (n=2)	CKS mn1	CKS mn0	0	CCS mn	MAS TERmn	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0		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	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)	CKS mn1	CKS mn0	0	CCS mn	0Note1	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0		0	MD mn3	MD mn2	MD mn1	MD mn0

CISmn1	CISmn0	Active edge selection for TImn pins
0	0	Falling edge
0	1	Rising edge
1	0	Double edge (when measuring low level width) Start trigger: falling edge, capture trigger: rising edge
1	1	Double edge (when measuring high level width) Start trigger: rising edge, capture trigger: falling edge
When the STSmn2~STSmn0 bits are not "010B" and are specified with a double edge, the CISmn1~CISmn0 bits must be "10B".		

Note 1: Bit11 is a read-only bit, fixed to “0”, and write is ignored.

Remark: m: unit number (m=0, 1) n: channel number (n=0~3)

Table 5-9: Table of timer mode register mn (TMRmn)(4/4)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn (n=2)	CKS mn1	CKS mn0	0	CCS mn	MAS TERmn	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0		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	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)	CKS mn1	CKS mn0	0	CCS mn	0Note1	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0		0	MD mn3	MD mn2	MD mn1	MD mn0

MD mn3	MD mn2	MD mn1	Setting of channel n operation mode	Corresponding functions	Count operation of TCR
0	0	0	Interval timer mode	Interval timer/square wave output/ Frequency divider function/PWM output (master)	Count down
0	1	0	Capture mode	Measurement of input pulse interval	Count up
0	1	1	Event counter mode	External event counter	Count down
1	0	0	Single count mode	Delay counter/single trigger pulse output/PWM output (slave)	Count down
1	1	0	Capture & Single count mode	Measurement of the high- and low-level width of the input signal	Count up
Other than the above			Settings are disabled.		
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))	MDmn0	Setting of starting counting and interrupt
• Interval timer mode (0, 0, 0) • Capture mode (0, 1, 0)	0	No timer interrupt is generated when counting starts (the output of the timer does not change).
	1	A timer interrupt is generated when counting starts (the output of the timer also changes).
• Event counter mode (0, 1, 1)	0	No timer interrupt is generated when counting starts (the output of the timer does not change).
• Single count mode Note 2 (1, 0, 0)	0	The start trigger in the count operation is invalid. No interruption at this time.
	1	The start trigger in the count operation is validNote 3. No interruption at this time.
• Capture & single count mode (1, 1, 0)	0	No timer interrupt is generated when counting starts (the output of the timer does not change). The start trigger in the count operation is invalid. No interruption at this time.

Note 1: Bit11 is a read-only bit, fixed to “0”, and write is ignored.

Note 2: In single count mode, the interrupt output (INTTMmn) and TOmn output at the start of counting are not controlled.

Note 3: If a start trigger is generated during operation (TSmn=1), the counter is initialized and counting is restarted (no interrupt request is generated).

Remark: m: unit number (m=0, 1) n: channel number (n=0~3)

5.3.4 Timer status register mn (TSRmn)

The TSRmn register is a register that indicates the overflow status of the channel n counter.

The TSRmn register is valid only in capture mode (MDmn3~MDmn1=010B) and capture & single count mode (MDmn3~MDmn1=110B). Refer to Table 5-11 for the OVF bit changes and set/clear conditions in each operation mode.

The TSRmn register is read by a 16-bit memory manipulation instruction.

The lower 8 bits of the TSRmn register can be read with TSRmnL and 8-bit memory manipulation instructions. After a reset signal is generated, the value of the TSRmn register changes to "0000H".

Table 5-10: Table of timer status register mn (TSRmn)

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	No overflow occurred.
1	Overflow occurred.
If the OVF bit is "1", this flag is cleared when the next count does not overflow and the count value is captured (OVF=0).	

Table 5-11: OVF bit change and set/clear conditions in each operation mode

Timer operation mode	OVF bit	Set/clear conditions
• Capture mode • Capture & single count mode	Clear	No overflow occurred at the capture.
	Set	Overflow occurred at the capture.
• Interval timer mode • Event counter mode • Single count mode	Clear	— (N/A)
	Set	

Remark:

1. m: unit number (m=0, 1) n: channel number (n=0~3)
2. Even if the counter overflows, the OVF bit does not change immediately, but changes on subsequent captures.

5.3.5 Timer channel enable status register m (TEm)

The TEm register is a register that indicates the enable or stop status of each channel timer operation.

Each of the TEm register corresponds to each of the timer channel start register m (TSm) and timer channel stop register m (TTm). If each bit of the TSm register is "1", the corresponding bit of the TEm register is "1". If each bit of the TTm register is "1", the corresponding bit of the TTm register is cleared to "0".

The TEm register is read by a 16-bit memory manipulation instruction.

The lower 8 bits of the TEm register can be read with TEmL and 8-bit memory manipulation instructions. After a reset signal is generated, the value of the TEm register changes to "0000H".

Table 5-12: Table of timer channel enable status register m (TEm)

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	0	0	0	0	TEm3	TEm2	TEm1	TEm0

TEHm3	Indication of whether operation of the higher 8-bit timer is enabled or stopped when channel 3 is in the 8-bit timer mode
0	Operation is stopped.
1	Operation is enabled.

TEHm1	Indication of whether operation of the higher 8-bit timer is enabled or stopped when channel 1 is in the 8-bit timer mode
0	Operation is stopped.
1	Operation is enabled.

TEmn	Indication of operation 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, 1) n: channel number (n=0~3)

5.3.6 Timer channel start register m (TSm)

The TSm register is a trigger register that initializes the timer counter register mn (TCRmn) and sets the start of counting operation for each channel. If each bit is set to "1", the corresponding bit of the timer channel enable status register m (TEm) is set to "1". Since the TSmn bit, the TSHm1 bit and the TSHm3 bit are trigger bits, the TSmn bit, the TSHm1 bit and the TSHm3 bit are cleared immediately if the operation enbale state is changed (TEmn, TEHm1, TEHm3 = 1).

The TSm register is set by a 16-bit memory manipulation instruction.

The lower 8 bits of the TSm register can be set by TSmL and by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of the TSm register changes to "0000H".

Table 5-13: Table of timer channel start register m (TSm)

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	0	0	0	0	TSm3	TSm2	TSm1	TSm0

TSHm3	Trigger to enable (start) operation of the higher 8-bit timer when channel 3 is in the 8-bit timer mode
0	No trigger operation.
1	Set the TEHm3 bit to "1" to enter the counting enable state. If the counting of the TCRm3 register is started in the count enable state, the interval timer mode is entered (refer to Table 5-22 of "5.5.2 Start Timing of Counter").

TSHm1	Trigger to enable (start) operation of the higher 8-bit timer when channel 1 is in the 8-bit timer mode
0	No trigger operation.
1	Set the TEHm1 bit to "1" to enter the counting enable state. If counting in the TCRm1 register is started in the count enable state, the interval timer mode is entered (refer to Table 5-22 of "5.5.2 Start Timing of Counter").

TSmn	Operation enable (start) trigger of channel n
0	No trigger operation.
1	Set the TEmn bit to "1" to enter the counting enable state. The start of counting in the TCRmn register in the count enable state varies with each operation mode (refer to Table 5-22 of "5.5.2 Start Timing of Counter"). When channel 1 and channel 3 are in 8-bit timer mode, TSm1 and TSm3 are operation enable (start) triggers for the lower 8-bit timer.

Notice:

1. Bits 15~12, 10, 8~4 must be set to “0”.
2. When switching from a function that does not use TImn pin input to a function that uses TImn pin input, the following period of waiting is required from setting the timer mode register mn (TMRmn) until the TSmn bit is set to "1":
   ① When the TImn pin noise filter is valid (TNFENmn=1): 4 operating clocks (F_{MCK})
   ② When the TImn pin noise filter is invalid (TNFENmn=0): 2 operating clocks (F_{MCK})

Remark:

1. The TSm register always reads "0".
2. m: unit number (m=0, 1) n: channel number (n=0~3)

5.3.7 Timer channel stop register m (TTm)

The TTm register is a trigger register to set the count stop of each channel.

If each bit is set to "1", the corresponding bit in the timer channel enable status register m (TEm) is cleared to "0". Since the TTmn bit, TTHm1 bit, and TTHm3 bit are trigger bits, the TTmn bit, TTHm1 bit, and TTHm3 bit are cleared immediately if the operation stop state is changed (TEmn, TEHm1, and TEHm3 = 0).

The TTm register is set by a 16-bit memory manipulation instruction.

The lower 8 bits of the TTm register can be set by TTmL and by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of the TTm register changes to "0000H".

Table 5-14: Table of timer channel stop register m (TTm)

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	0	0	0	0	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. 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.

Notice: Bits 15~12, 10, 8~4 must be set to “0”.

Remark:

1. The TTm register always reads "0".
2. m: unit number (m=0, 1) n: channel number (n=0~3)

5.3.8 Timer input output select register (TIOS0)

The TIOS0 register is used to make selections for the inputs and outputs of unit 0. The timer inputs for channel 0 and channel 1 and the timer output for channel 2 of unit 0 are selected. The TIOS0 register is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of the TIOS0 register changes to "00H".

Table 5-15: Table of timer input select register 0 (TIOS0)

Address:	0x40020474	After reset: 00H		R/W			
Symbol	7	6	5	4	3	2	1	0
TIOS0	TIS07	TIS06	TIS05	TIS04	TOS03	TIS02	TIS01	TIS00

TIS07	TIS06	TIS05	Selection of timer input used for channel 0
0	0	0	Input signal for timer input pin (TI00)
Other			Settings are disabled.

TIS04	Selection of timer input used for channel 0
0	Input signal selected by TIS07~TIS05
1	Event input signal of ELC

TOS03	Enable channel 2 timer output
0	Output enable
1	Output disable (output fixed to 0)

TIS02	TIS01	TIS00	Selection of timer input used for channel 1
0	0	0	Input signal for timer input pin (TI01)
0	0	1	Event input signal of ELC
0	1	0	Input signal to timer input pin (TI01)
0	1	1	
1	0	0	Low-speed on-chip oscillator clock (F_{IL})
1	0	1	Subsystem clock (F_{SUB})
Other than above			Settings are disabled.

Notice:

1. The high-/low-level width of the selected timer inputs needs to be greater than or equal to $1/F_{MCK}$+10ns. Therefore, when F_{SUB} is selected as the F_{CLK} (CSS bit of the CKC register =1), the TIS02 bit cannot be set to "1".
2. When selecting the event input signal for ELC via timer input select register 0 (TIOS0), F_{CLK} must be selected via timer clock select register 0 (TPS0).

5.3.9 Timer output enable register m (TOEm)

The TOEm register is a register that sets to enable or disable the 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 is set by a 16-bit memory manipulation instruction.

The lower 8 bits of the TOEm register can be set by TOEmL and by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of the TOEm register changes to "0000H".

Table 5-16: Table of timer output enable register m (TOEm)

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	0	0	0	0	TOEm3	TOEm2	TOEm1	TOEm0

TOEmn	Enable/disbale the timer output of channel n
0	Disable timer output. The operation of the timer is not reflected to the TOmn bit, fixed output. The TOmn bit can be written and the level set by the TOmn bit is output from the TOmn pin.
1	Enable timer output. The operation of the timer is reflected to the TOmn bit, producing an output waveform. The write of the TOmn bit is ignored.

Notice: Bit15~4 must be set to “0”.

Remark: m: unit number (m=0, 1) n: channel number (n=0~3)

5.3.10 Timer output register m (TOm)

The TOm register is a buffer register for each channel timer output.

The bit value of this register is output from the output pin (TOmn) of each channel timer.

The TOmn bit of this register can be rewritten by software only when timer output is disabled (TOEmn=0). When enabling the timer output (TOEmn=1), rewrite operations via software are ignored and its value is changed only by the operation of the timer.

To use the TI00/TO00, TI01/TO01, TI02/TO02, and TI03/TO03 pins as port functions, the corresponding TOmn bit must be set to "0".

The TOm register is set by a 16-bit memory manipulation instruction.

The lower 8 bits of the TOm register can be set by TOmL and by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of the TOm register changes to "0000H".

Table 5-17: Table of timer output register m (TOm)

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	0	0	0	0	TOm3	TOm2	TOm1	TOm0

TOmn	Timer output of channel n
0	The output value of the timer is "0".
1	The output value of the timer is "1".

Notice: Bit15~4 must be set to “0”.

Remark: m: unit number (m=0, 1) n: channel number (n=0~3)

5.3.11 Timer output level register m (TOLm)

The TOLm register is a register that controls the output level of each channel timer.

When timer output (TOEmn=1) is enabled and the multi-channel linkage operation function (TOMmn=1) is used, the set and reset timing of the timer output signal reflects the inverse setting of each channel n performed by this register. In the master channel output mode (TOMmn=0), this register setting is invalid.

The TOLm register is set by a 16-bit memory manipulation instruction.

The lower 8 bits of the TOLm register can be set by TOLmL and by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of the TOLm register changes to "0000H".

Table 5-18: Table of timer output level register m (TOLm)

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	0	0	0	0	TOLm3	TOLm2	TOLm1	0

TOLmn	Control of timer output level of channel n
0	Positive logic output (active-high)
1	Inverted output (active-low)

Notice:

1. Bit15~4 and bit0 must be set to “0”.
2. If the value of this register is rewritten while the timer is operating, the timer output logic is inverted at the next time the timer output signal changes, rather than immediately after the rewrite.

Remark: m: unit number (m=0, 1) n: channel number (n=0~3)

5.3.12 Timer output mode register m (TOMm)

The TOMm register is a register that controls the output mode of each channel timer. When used as an independent channel operation function, the corresponding bit of the using channel should be set to "0".

When used as a multi-channel linkage operation function (PWM output, single trigger pulse output and multiple PWM output), the corresponding bit of the master channel is "0" and the corresponding bit of the slave channel is "1".

When the timer output (TOEmn=1) is enabled, the setting of each channel n is reflected in this register during the setting and reseting timing of the timer output signal.

The TOMm register is set by a 16-bit memory manipulation instruction.

The lower 8 bits of the TOMm register can be set by TOMmL and by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of the TOMm register changes to "0000H".

Table 5-19: Table of timer output mode register m (TOMm)

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	0	0	0	0	TOMm3	TOMm2	TOMm1	0

TOMmn	Control of channel n timer output mode
0	Master channel output mode (alternate output via timer interrupt request signal (INTTMmn))
1	Slave channel output mode (output is set via timer interrupt request signal (INTTMmn) of master channel and output is reset via timer interrupt request signal (INTTMmp) of slave channel)

Notice: Bit15~4 and bit0 must be set to “0”.

Remark: m: unit number (m=0, 1) n: channel number n=0~3 (master channel: n=0, 2) p: slave channel number (n=0: p=1, 2, 3 n=2: p=3) (for details on the relationship between the master channel and the slave channel, refer to "5.4.1 Basic Rules for Multi-Channel Linkage Operation Function".)

5.3.13 Noise filter enable register 1 (NFEN1)

The NFEN1 register sets whether the noise filter is used for the input signals of the timer input pins of each channel of Unit 0. For pins that require noise removal, the corresponding bit must be set to "1" to make the noise filter effective. 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 is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of the NFEN1 register changes to "00H".

Table 5-20: Table of noise filter enable register 1 (NFEN1)

Address: 0x40040471

Symbol	7	6	5	4	3	2	1	0
NFEN1	0	0	0	0	TNFEN03	TNFEN02	TNFEN01	TNFEN00

TNFEN03	Usage of input signal noise filter on TI03 pin
0	Noise filter OFF
1	Noise filter ON

TNFEN02	Usage of input signal noise filter on TI02 pin
0	Noise filter OFF
1	Noise filter ON

TNFEN01	Usage of input signal noise filter on TI01 pin
0	Noise filter OFF
1	Noise filter ON

TNFEN00	Usage of input signal noise filter on TI00 pin
0	Noise filter OFF
1	Noise filter ON

Remark:

1. For details, refer to "5.5.1(2) When the valid edge of TImn pin input signal (CCSmn=1) is selected", "5.5.2 Start Timing of Counter ", and "5.7 Control of Timer Input (TImn)".
2. Refer to "Chapter 2 Port Function" for the configuration of timer input/output pins of channels 0~3.

5.3.14 Noise filter enable register 2 (NFEN2)

The NFEN2 register sets whether the noise filter is used for the input signals of the timer input pins of each channel of Unit 1. For pins that require noise removal, the corresponding bit must be set to "1" to make the noise filter effective. 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 NFEN2 register is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of the NFEN2 register changes to "00H".

Table 5-21: Table of noise filter enable register 2 (NFEN2)

Address: 0x40040472

Symbol	7	6	5	4	3	2	1	0
NFEN2	0	0	0	0	TNFEN13	TNFEN12	TNFEN11	TNFEN10

TNFEN13	Usage of input signal noise filter on TI13 pin
0	Noise filter OFF
1	Noise filter ON

TNFEN12	Usage of input signal noise filter on TI12 pin
0	Noise filter OFF
1	Noise filter ON

TNFEN11	Usage of input signal noise filter on TI11 pin
0	Noise filter OFF
1	Noise filter ON

TNFEN10	Usage of input signal noise filter on TI10 pin
0	Noise filter OFF
1	Noise filter ON

Remark:

1. For details, refer to "5.5.1(2) When the valid edge of TImn pin input signal (CCSmn=1) is selected", "5.5.2 Start Timing of Counter ", and "5.7 Control of Timer Input (TImn)".
2. Refer to "Chapter 2 Port Function" for the configuration of timer input/output pins of channels 0~3.

5.3.15 Registers controlling port functions of timer input/output pins

When using the general-purpose timer unit, the input/output pins of Timer0/Timer1 can be arbitrarily configured to each port except RESETB. For details, refer to "Chapter 2 Port Function".

When multiplexing the output pins of Timer0/Timer1 to a port, you must set the bits of the port mode control register (PMCxx) and the port mode register (PMxx) corresponding to that port to "0", and set the port multiplexing function configuration register (PxxCFG). At this time, the bit of the port register (Pxx) can be “0” or “1”.

(Example) When P21 is configured as TO10 and used as a timer output

Set the PMC21 bit of port mode control register 2 to "0".

Set bit PM21 of port mode register 2 to "0".

Set port output multiplexing function configuration register P21CFG to "0x12".

When using the multiplexed ports of the Timer0/Timer1 input pins as timer inputs, you must set the bit of the port mode register (PMxx) corresponding to each port to "1" and the bit of the port mode control register (PMCxx) to "0", and set the port multiplexing function configuration register (PxxCFG). At this time, the bit of the port register (Pxx) can be "0" or "1".

(Example) When P20 is configured as TI10 and used as a timer output

Set the PMC20 bit of port mode control register 2 to "0".

Set bit PM20 of port mode register 2 to "0".

Set port output multiplexing function configuration register P20CFG to "0x0a".

5.4 Basic rules of general-purpose timer unit

5.4.1 Basic rules of multi-channel linkage operation function

The multi-channel linkage function is a function that combines a master channel (a reference timer that counts cycles) and a slave channel (a timer that operates in compliance with the master channel), and several rules need to be observed when using it.

The basic rules of the multi-channel linkage operation function are shown below.

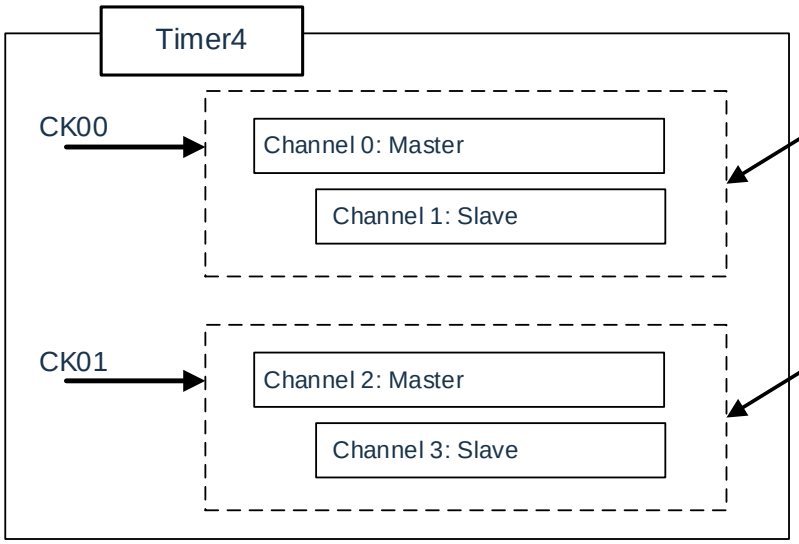
1) Only the even-number channel (channel 0, channel 2) can be set as a master channel.
2) Any channel other than channel 0 can be set as a slave channel.
3) Only the lower channel of the master channel can be set as a slave channel.
   For example, when setting channel 0 as the master channel, it is possible to set the channels starting from channel 1 (channels 1 to 3) as slave channels.
4) Multiple slave channels can be set for 1 master channel.
5) When multiple master channels are used, slave channels that span the master channel cannot be set.
   For example, when setting channel 0 and channel 2 as the master channel, channel 1 can be set as the slave channel of master channel 0, but channel 3 cannot be set as the slave channel of master channel 0.
6) The slave channels linked to the master channel need to be set to the same operating clock. The CKSmn0 bit and CKSmn1 bit (bit15 and bit14 of Timer Mode Register mn (TMRmn)) of the slave channel linked to the master channel need to be the same setting value.
7) The master channel can pass the INTTMmn (interrupt), start software trigger and count clock to the lower channel.
8) The slave channel can use the master channel's INTTMmn (interrupt), start software trigger, and count clocks as source clocks, but cannot pass its own INTTMmn (interrupt), start software trigger, and count clocks to the lower channel.
9) The master channel cannot use the INTTMmn (interrupt), start software trigger and count clocks of other high master channels as source clocks.
10) In order to start the channels to be linked at the same time, the channel start trigger bit (TSmn) of the linked channel needs to be set at the same time.
11) Only all linked channels or the master channel can use the setting of the TSmn bit in the counting operation. It is not possible to use the setting of the TSmn bit of the slave channel only.
12) In order to stop the linked channels at the same time, the channel stop trigger bit (TTmn) of the linked channel needs to be set at the same time.
13) In linked operation, CKm2/CKm3 cannot be selected because the master and slave channels need the same operating clock.
14) The timer mode register m0 (TMRm0) has no master bit and is fixed to "0". However, since channel 0 is the highest bit channel, it can be used as the master channel during linkage operation.

The basic rules of the multi-channel linkage operation function are the rules applicable to the group of channels (a collection of master and slave channels that form a multi-channel linkage operation function).

If you set 2 or more channel groups that are not linked to each other, the above basic rules do not apply to the channel groups.

Remark: m: unit number (m=0, 1) n: channel number (n=0~3)

Example 1:

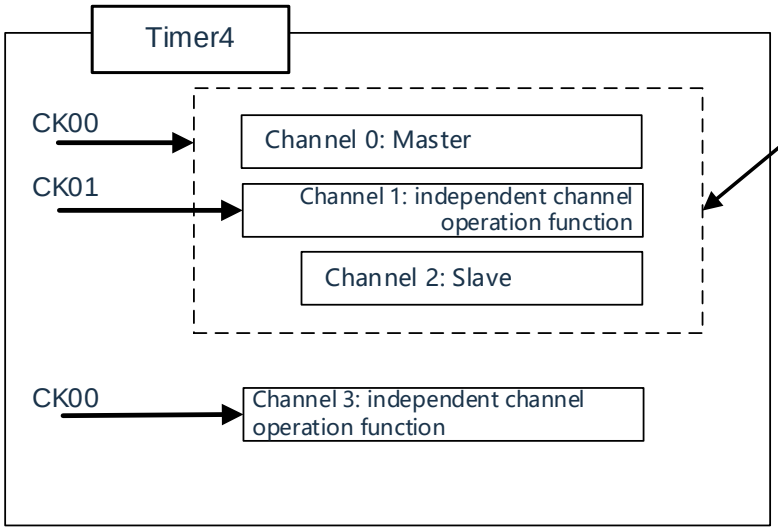


Channel group 1
(Multi-channel linkage function)

Channel group 2
(Multi-channel linkage function)

※ The operating clock of channel group 1 may be different from that of channel group 2.

Example 2:



Channel group 1
(Multi-channel linkage operation function)

※ 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.

5.4.2 Basic rules of 8-bit timer operation function (channels 1 and 3 of unit 0 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 (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:
   ① Interval timer function
   ② External event counter function
   ③ 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 linkage operation functions (single pulse, PWM, and multiple PWM) cannot be used.

Remark: unit number (m=0) n: channel number (n=1, 3)

5.5 Operation of counter

5.5.1 Count clock (F_{TCLK})

The count clock of the general-purpose timer unit (F_{TCLK}) can be selected by the CCSmn bit of the timer mode register mn (TMRmn) for any of the following clocks:

① The CKSmn0 bit and CKSmn1 bit specified operation clock (F_{MCK})

② The active edge of the TImn pin input signal

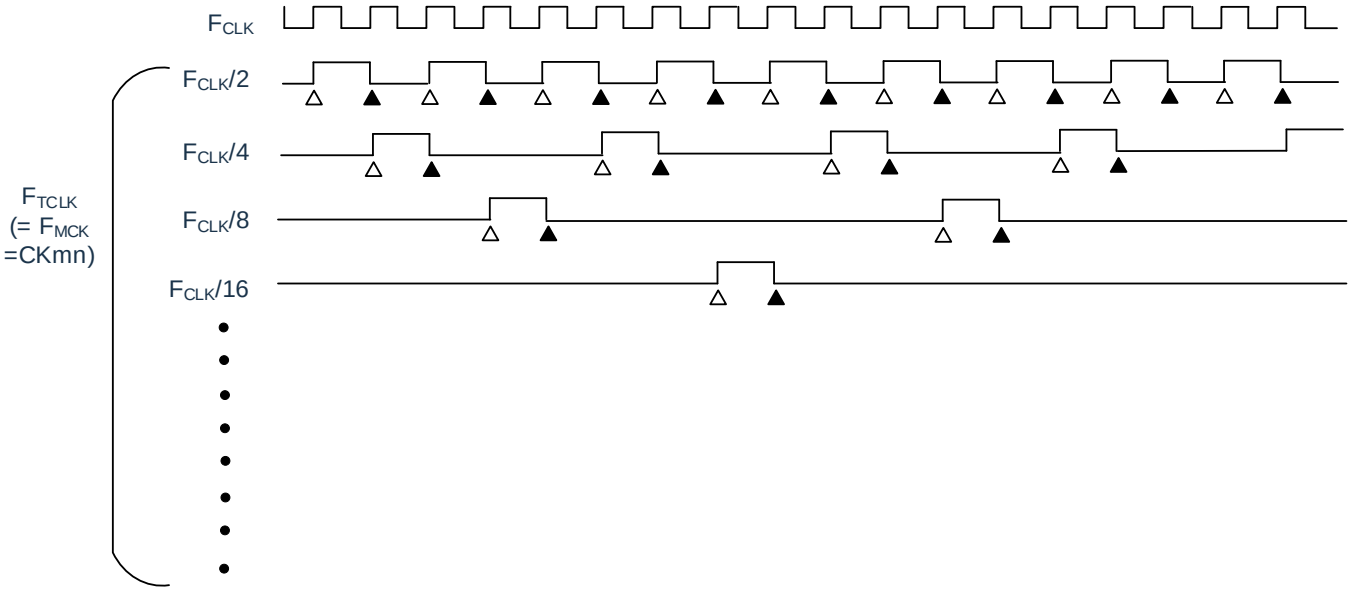
The general-purpose timer unit is designed to operate synchronously with F_{CLK}, so the timing of the count clock (F_{TCLK}) is as follows.

(1) When operation clock (F_{MCK}) specified by the CKSmn0 and CKSmn1 bits is selected (CCSmn = 0)

According to the setting of timer clock selection register m (TPSm), the counting clock (F_{TCLK}) is F_{CLK} ~F_{CLK} /2^{15}. However, when the frequency division of F_{CLK} is selected, the clock selected by TPSm register is a signal that has only 1 F_{CLK} cycle of high level from the rising edge. When F_{CLK} is selected, it is fixed to high level.

In order to obtain synchronization with F_{CLK}, timer count register mn (TCRmn) delays the counting by one F_{CLK} clock from the rising edge of the counting clock, which is called "counting at the rising edge of the counting clock" for convenience.

Figure 5-3: Timing of F_{CLK} and count clock (F_{TCLK}) (When CCSmn = 0)



Remark: △: Rising edge of the count clock

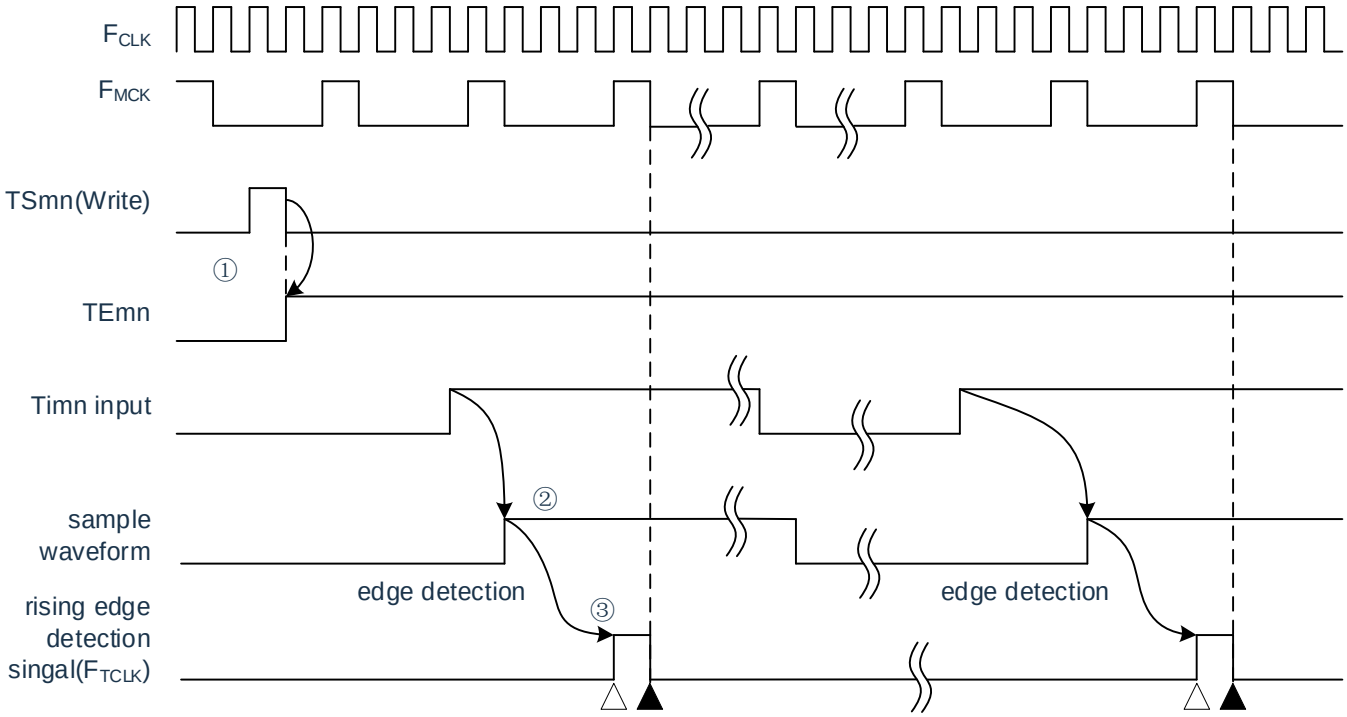
▲: Synchronization, increment/decrement of counter

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}) is a signal that detects an active edge of the TImn pin input signal and is synchronized with the next F_{MCK} rising edge. In fact, this is a signal delayed by 1~2 F_{MCK} clocks compared to the input signal of the TImn pin (delay 3~4 F_{MCK} clocks when using noise filters). In order to obtain synchronization with F_{CLK}, the timer count register mn (TCRmn) delays the count by one F_{CLK} time from the rising edge of the count clock, which is referred to as "counting at the effective edge of the TImn pin input signal" for convenience.

Figure 5-4: Timing of the counting clock (F_{TCLK}) (CCSmn=1, without noise filter)



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.

Remark:

1. △: Rising edge of the count clock
   ▲: Synchronization, increment/decrement of counter
   F_{CLK}: CPU/peripheral hardware clock
   F_{MCK}: Operation clock of channel n
2. The same waveforms are used for the measurement of the input pulse interval, the high and low measurement of the input signal, the delay counter and the TImn input for the single trigger pulse output function.

5.5.2 Start timing of counter

The timer count register mn (TCRmn) enters the operation enable state by setting TSmn bit of the timer channel start register m (TSm).

Execution from the counting enable state to the start of the timer count register mn (TCRmn) is shown in Table 5-22.

Table 5-22: Operation from the counting enable state to the start of the timer count register mn (TCRmn)

Timer operation mode	Operation after setting TSmn bit to "1"
• Interval timer mode	No operation is performed from the detection of the start trigger (TSmn=1) until the count clock is generated. The value of the TDRmn register is loaded into the TCRmn register by the first count clock and decremented by subsequent count clocks (refer to “5.5.3(1) Operation of the interval timer mode”).
• Event counter mode	The value of the TDRmn register is loaded into the TCRmn register by writing a "1" to the TSmn bit. If the input edge of TImn is detected, the count is decremented by the subsequent count clocks. (Refer to “5.5.3(2) Operation of the event counter mode”).
• Capture mode	No operation is performed from the time the start trigger is detected until the count clock is generated. The “0000H” is loaded into the TCRmn register by the first count clock, and incremental counting is performed by the subsequent count clocks (refer to “5.5.3(3) Operation of the capture mode (input pulse interval measurement)”).
• Single count mode	By writing "1" to the TSmn bit while the timer is stopped (TEmn=0), it enters the wait state for the start of the trigger. No operation is performed from the time the start trigger is detected until the count clock is generated. The value of the TDRmn register is loaded into the TCRmn register by the first count clock, and decremental counting by subsequent count clocks (refer to “5.5.3(4) Operation of the single count mode”).
• Capture & single count mode	By writing "1" to the TSmn bit while the timer is stopped (TEmn=0), it enters the wait state for the start of the trigger. No operation is performed from the time the start trigger is detected until the count clock is generated. The “0000H” is loaded into the TCRmn register by the first count clock, and incremental counting is performed by the subsequent count clocks (refer to “5.5.3(5) Operation of capture & single count mode (measurement of high-level width”).

5.5.3 Operation of counter

The following describes the counter operation for each mode.

(1) Operation of interval timer mode

① The operation enable state is entered by writing "1" to the TSmn bit (TEmn=1). The timer count register mn (TCRmn) remains at its initial value until a count clock is generated.

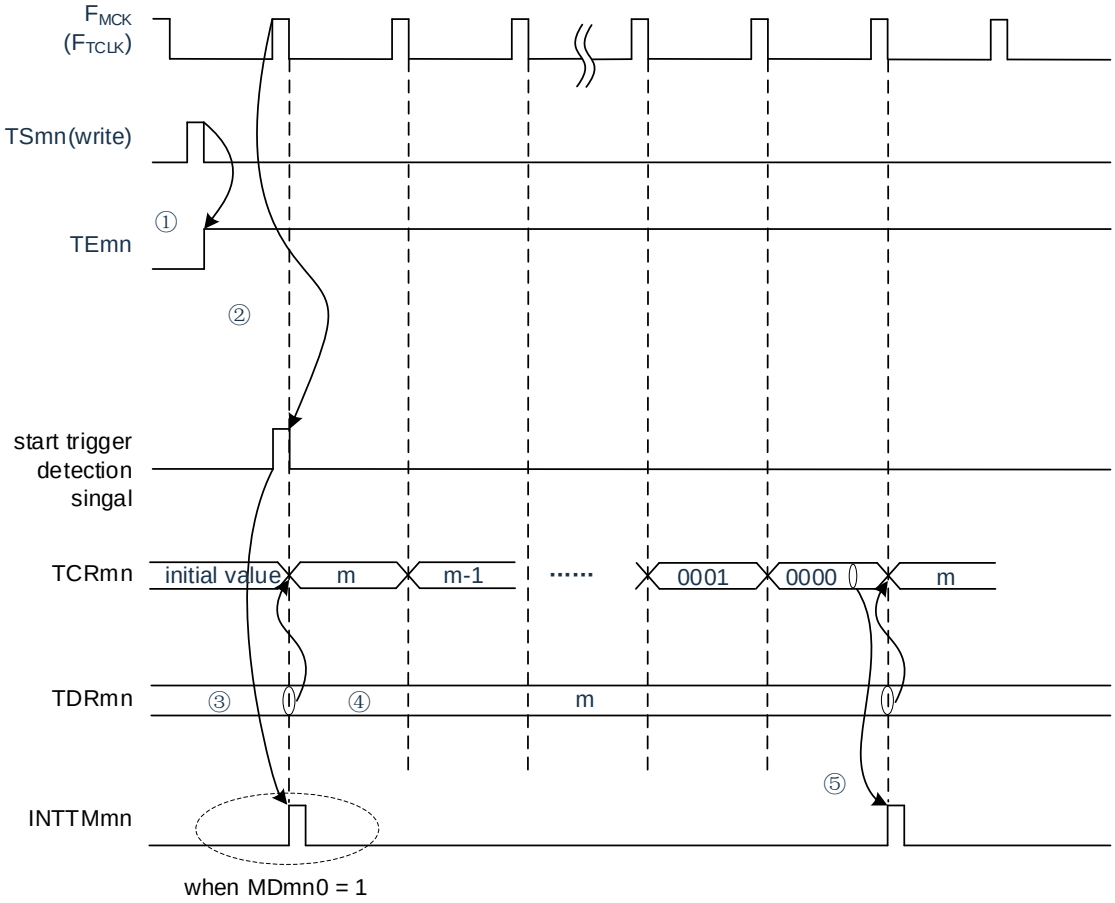
② A start trigger signal is generated by enabling the 1st count clock (F_{MCK}) after the operation.

③ When MDmn0 bit is "1", INTTMmn is generated by the start trigger signal.

④ The value of timer data register mn (TDRmn) is loaded into the TCRmn register by enabling the 1st count clock after the operation, and counting starts in interval timer mode.

⑤ If the TCRmn register decrements to "0000H", INTTMmn is generated by the next count clock (F_{MCK}) and continues counting after loading the value of timer data register mn (TDRmn) into the TCRmn register.

Figure 5-5: Operation timing (interval timer mode)



Remark:

1. 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.
2. 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

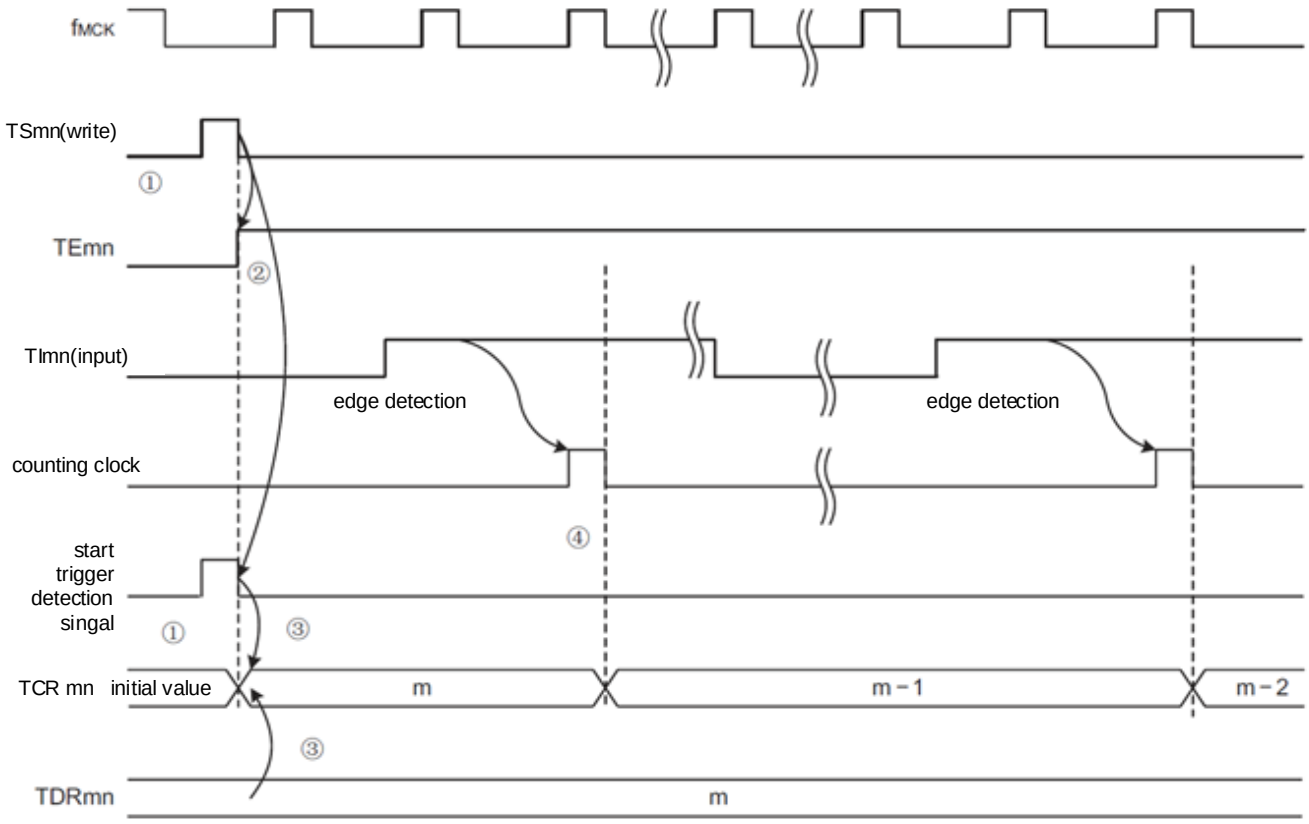
① The timer count register mn (TCRmn) holds its initial value while operation is stopped (TEmn=0).

② The operation enable state is enabled by writing "1" to the TSmn bit (TEmn=1).

③ The value of timer data register mn (TDRmn) is loaded into the TCRmn register while both the TSmn and TEmn bits are changed to "1" and counting begins.

Thereafter, the value of the TCRmn register is counted decreasingly by the count clock at the active edge of the TImn input.

Figure 5-6: Operation timing (event counter mode)



Remark: This is a timing without the noise filter. If the noise filter is used, the edge detection is delayed by 2 more F_{MCK} cycles (3~4 cycles in total) from the TImn input. The 1 cycle error is because the TImn input is not synchronized with the count clock (F_{MCK}).

(3) Operation of capture mode (interval measurement of input pulses)

① The operation enable state is entered by writing "1" to the TSmn bit (TEmn=1).

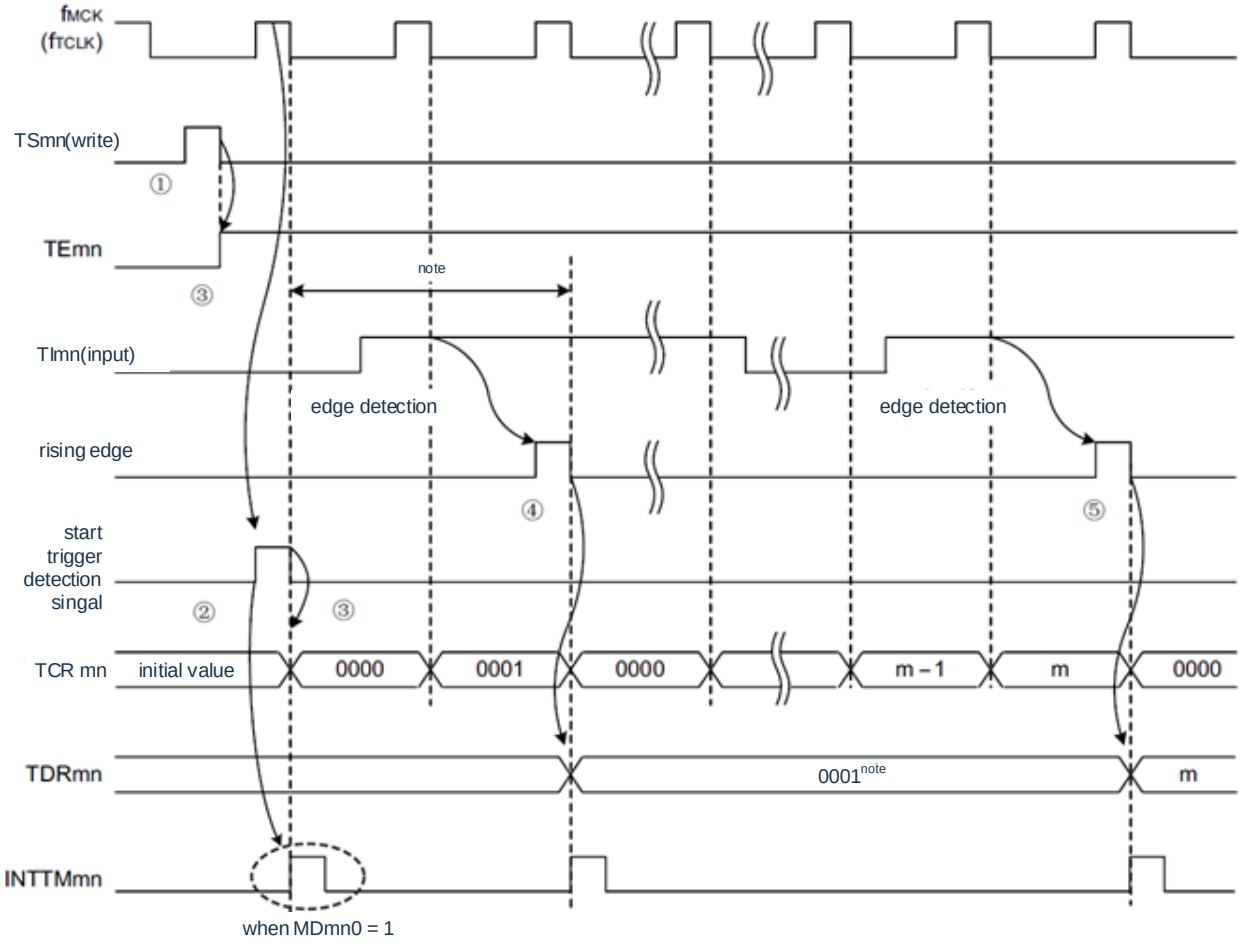
② The timer count register mn (TCRmn) remains at its initial value until a count clock is generated.

③ A start trigger signal is generated by enabling the 1st count clock (F_{MCK}) after the operation. Then, the "0000H" is loaded into the TCRmn register and counting starts in capture mode (INTTMmn is generated by the start trigger signal when MDmn0 bit is "1").

④ If an active edge of TImn input is detected, the value of TCRmn register is captured to TDRmn register and INTTMmn interrupt is generated. The capture value is meaningless at this point. The TCRmn register continues counting from the "0000H".

⑤ If an active edge of the next TImn input is detected, the value of the TCRmn register is captured to the TDRmn register and the INTTMmn interrupt is generated.

Figure 5-7: Operation timing (capture mode: interval measurement of input pulses)



Note 1: When the clock is input to TImn (with trigger) before the start, the count is started by detecting the trigger even if no edge is detected, so the capture value at the 1st capture (④) is not a pulse interval (in this example, 0001: 2 clock intervals) and must be ignored.

Note 2: Because the 1st count clock cycle runs after the TSmn bit is written and delays the start of counting before generating the count clock, an error of up to 1 clock cycle is generated. Also, if you need information about the start of the count timing, set MDmn0 to "1" so that an interrupt can be generated at the start of the count.

Remark: This is a timing without the noise filter. If the noise filter is used, the edge detection is delayed by 2 more F_{MCK} cycles (3~4 cycles in total) from the TImn input. The 1 cycle error is because the TImn input is not synchronized with the count clock (F_{MCK}).

(4) Operation of single count mode

① The operation enable state is entered by writing "1" to the TSmn bit (TEmn=1).

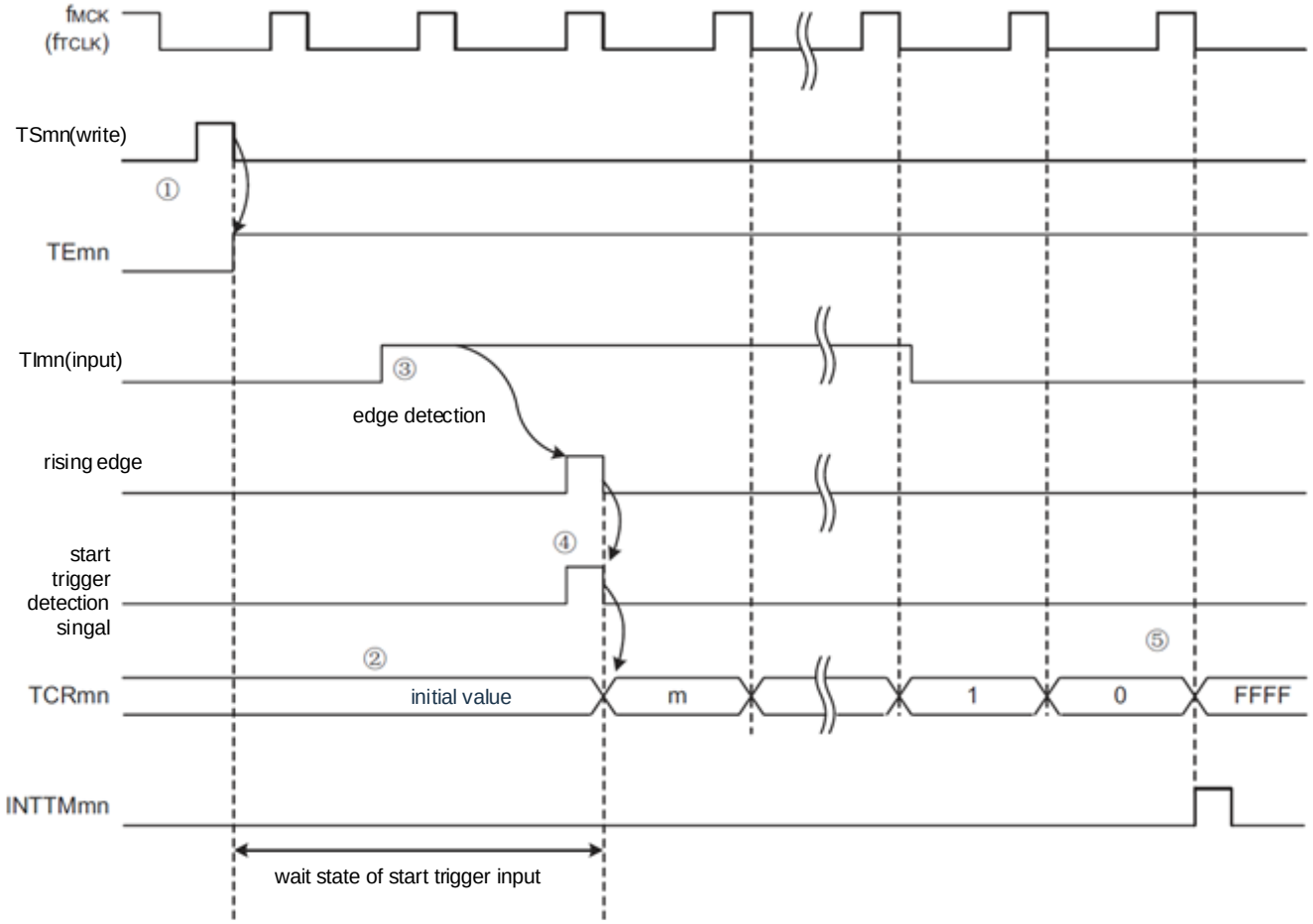
② The timer count register mn (TCRmn) remains the initial value until a start trigger signal is generated.

③ Detects the rising edge of the TImn input.

④ The value (m) of the TDRmn register is loaded into the TCRmn register after a start trigger signal is generated, and counting begins.

⑤ When the TCRmn register decrements to "0000H", the INTTMmn interrupt is generated and the value of TCRmn register changes to "FFFFH" and stop counting.

Figure 5-8: Operation timing (single count mode)



Remark: This is a timing without the noise filter. If the noise filter is used, the edge detection is delayed by 2 more F_{MCK} cycles (3~4 cycles in total) from the TImn input. The 1 cycle error is because the TImn input is not synchronized with the count clock (F_{MCK}).

(5) Operation of capture & single count mode (measurement of high-level width)

① The operation enable state is entered by writing "1" to the TSmn bit of the timer channel start register m (TSm)(TEmn=1).

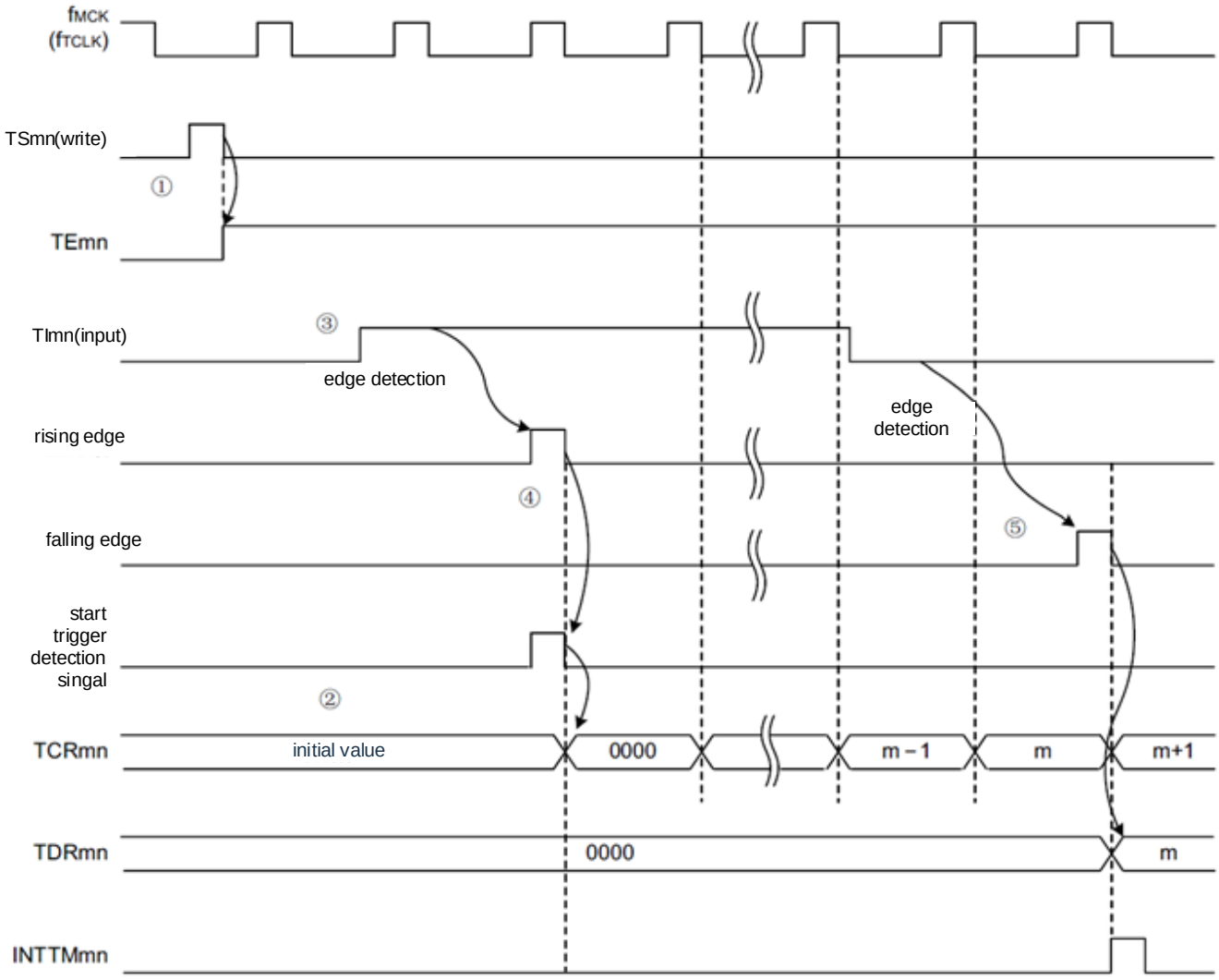
② The timer count register mn (TCRmn) remains the initial value until a start trigger signal is generated.

③ Detects the rising edge of the TImn input.

④ After the start trigger signal is generated, "0000H" is loaded into the TCRmn register and counting starts.

⑤ If the falling edge of TImn input is detected, the value of the TCRmn register is captured to the TDRmn register and an INTTMmn interrupt is generated.

Figure 5-9: Operation timing (capture & single count mode: measurement of high-level width)

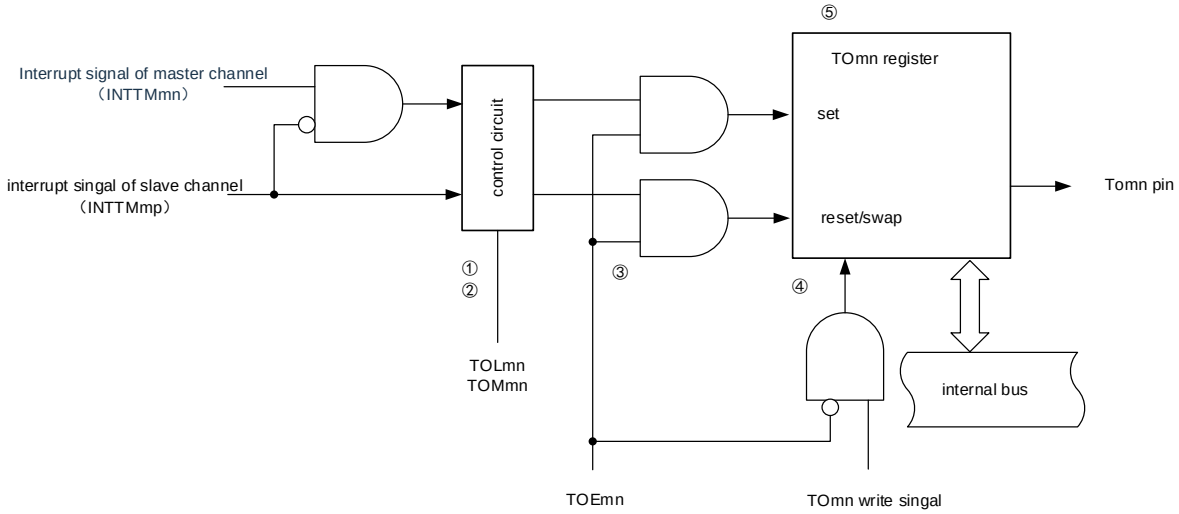


Remark: This is a timing without the noise filter. If the noise filter is used, the edge detection is delayed by 2 more F_{MCK} cycles (3~4 cycles in total) from the TImn input. The 1 cycle error is because the TImn input is not synchronized with the count clock (F_{MCK}).

5.6 Channel output (TOmn pin) control

5.6.1 TOmn pin output circuit configuration

Figure 5-10: Output circuit configuration



The following explains the output circuit of the TOmn pin.

① When the TOMmn bit is "0" (master channel output mode), the setting value of timer output level register m (TOLm) is ignored and only INTTMmp (slave channel timer interrupt) is passed to timer output register m (TOm).

② When the TOMmn bit is "1" (slave channel output mode), INTTMmn (master channel timer interrupt) and INTTMmp (slave channel timer interrupt) are passed 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, INTTMmp → reset)
When TOLmn = 1: Negative logic output (INTTMmn → reset, INTTMmp → set)
When INTTMmn and INTTMmp are simultaneously generated, (0% output of PWM), INTTMmp (reset signal) takes priority, and INTTMmn (set signal) is masked.

③ In the state of enabling timer output (TOEmn=1), INTTMmn (master channel timer interrupt) and INTTMmp (slave channel timer interrupt) are passed to TOm register. Writing to the TOm register (TOmn write signal) is invalid.
When the TOEmn bit is "1", the output of the TOmn pin is not changed except for the interrupt signal. To initialize the output level of the TOmn pin, you need to write a value to the TOm register after setting it to disable the timer output (TOEmn=0).

④ Writing to the TOmn bit for the object channel (TOmn write signal) is valid when the timer output is disabled (TOEmn=0). When the timer output is disabled (TOEmn=0), INTTMmn (master channel timer interrupt) and INTTMmp (slave channel timer interrupt) are not passed to the TOm register.

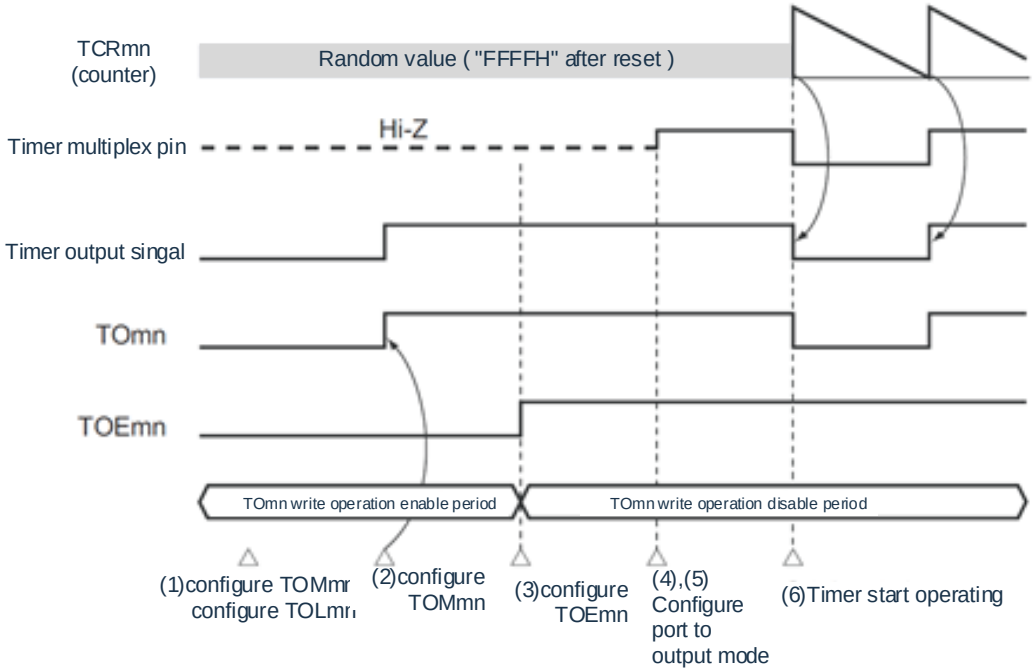
⑤ The TOm register can be read at any time and the output level of the TOmn pin can be confirmed.

Remark: m: unit number (m= 0,1) n: channel number n=0~3 (master channel: n=0, 2) p: slave channel number (n=0: p=1, 2, 3 n=2: p=3)

5.6.2 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 5-11: State change from setting timer output to start of operation



① Set the operation mode of the timer output.
TOMmn bit (0: master channel output mode, 1: slave channel output mode)
TOLmn bit (0: positive logic output, 1: negative logic output)

② The timer output signal is set to the initial state by setting the timer output register m (TOm).
Writing "1" to TOEmn bit enables timer output (writing to TOm register is disabled).

③ The port is set to digital input/output via the port mode control register (PMCxx)

④ Set the input/output of the port to output

⑤ Enable timer operation (TSmn=1).

Remark: m: unit number (m=0, 1) n: channel number (n=0~3)

5.6.3 Cautions for channel output operation

(1) Change of setting values for TOm, TOEm, TOLm, TOMm registers in timer operation

The operation of the timer (timer count register mn (TCRmn) and timer data register mn (TDRmn)) and the Tomn output circuit are independent. Therefore, changes in the setting values of timer output register m (TOm), timer output enable register m (TOEm), and timer output level register m (TOLm) do not affect the operation of the timer, and the setting values can be changed during timer operation. However, in order to output the expected waveform from the TOmn pin during the operation of each timer, the value must be set to the example of the register setting contents for each operation shown in 5.8 and 5.9.

If the setting values of TOEm register and TOLm register other than TOm register are changed before and after generating the timer interrupt (INTTMmn) signal for each channel, the waveform output from TOmn pin may be different depending on whether it is changed before or after generating the timer interrupt (INTTMmn) signal.

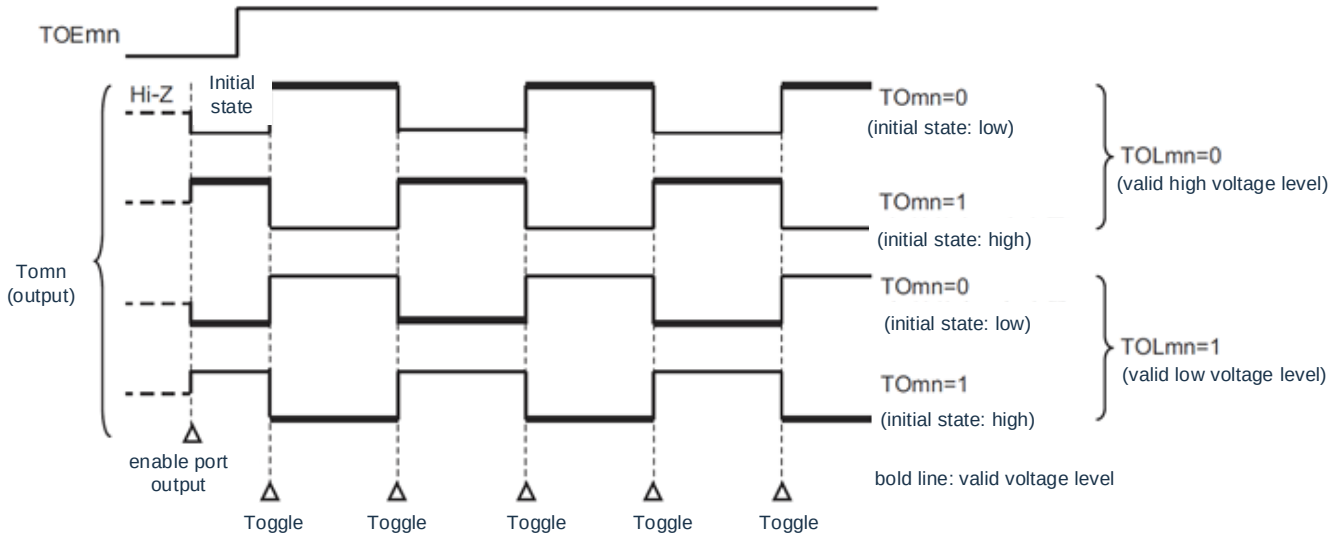
(2) Default level of TOmn pin and output level after timer operation start

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.

① Operation starts in master channel output mode (TOMmn=0)

In the master channel output mode (TOMmn=0), the setting of the timer output level register m (TOLm) is invalid. If the timer operation is started after the initial level is set, the output level of the TOmn pin is inverted by generating an alternate signal.

Figure 5-12: Output state of TOmn pin at alternate output (TOMmn=0)



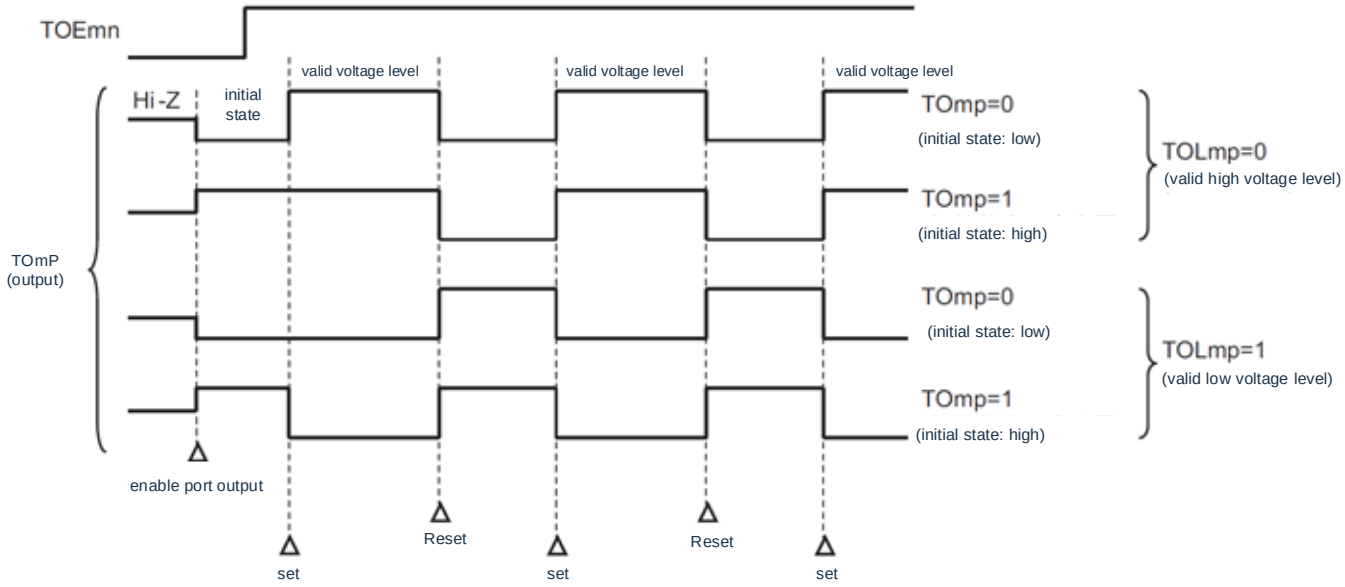
Remark:

1. Toggle: Reverse TOmn pin output status.
2. m: unit number (m=0, 1) n: channel number (n=0~3)

② When operation starts with slave channel output mode (TOMmn = 1) setting (PWM output))

In slave channel output mode (TOMmn=1), the active level depends on the setting of timer output level register m (TOLmn).

Figure 5-13: Output state of TOmn pin at PWM output (TOMmn=1)



Remark:

1. Set: The output signal from the TOmp pin changes from an invalid level to a valid level.
2. Reset: The output signal from the TOmp pin changes from a valid level to an invalid level.
3. m: unit number (m=0, 1) n: channel number n=0~3 (master channel: n=0, 2) p: slave channel number (n=0: p=1, 2, 3 n=2: p=3)

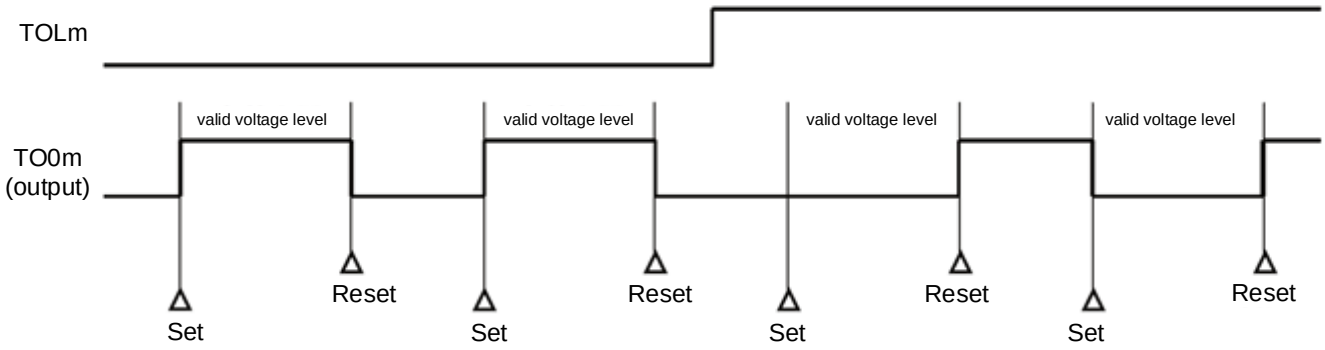
(3) Operation of TOmn pin in slave channel output mode (TOMmn = 1)

① 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 5-14: Operation when the contents of the TOLm register are changed during timer operation



② Set/reset timing

In order to achieve 0% and 100% output at PWM output, the set timing of the TOmn pin/TOmn bit when generating the master channel timer interrupt (INTTMmn) is delayed by 1 count clock via the slave channel.

When the set condition and reset condition are generated at the same time, the reset condition is given priority.

The set/reset operation status when setting the master/slave channel according to the following method is shown in Figure 5-15.

Master channel: TOEmn=1, TOMmn=0, TOLmn=0

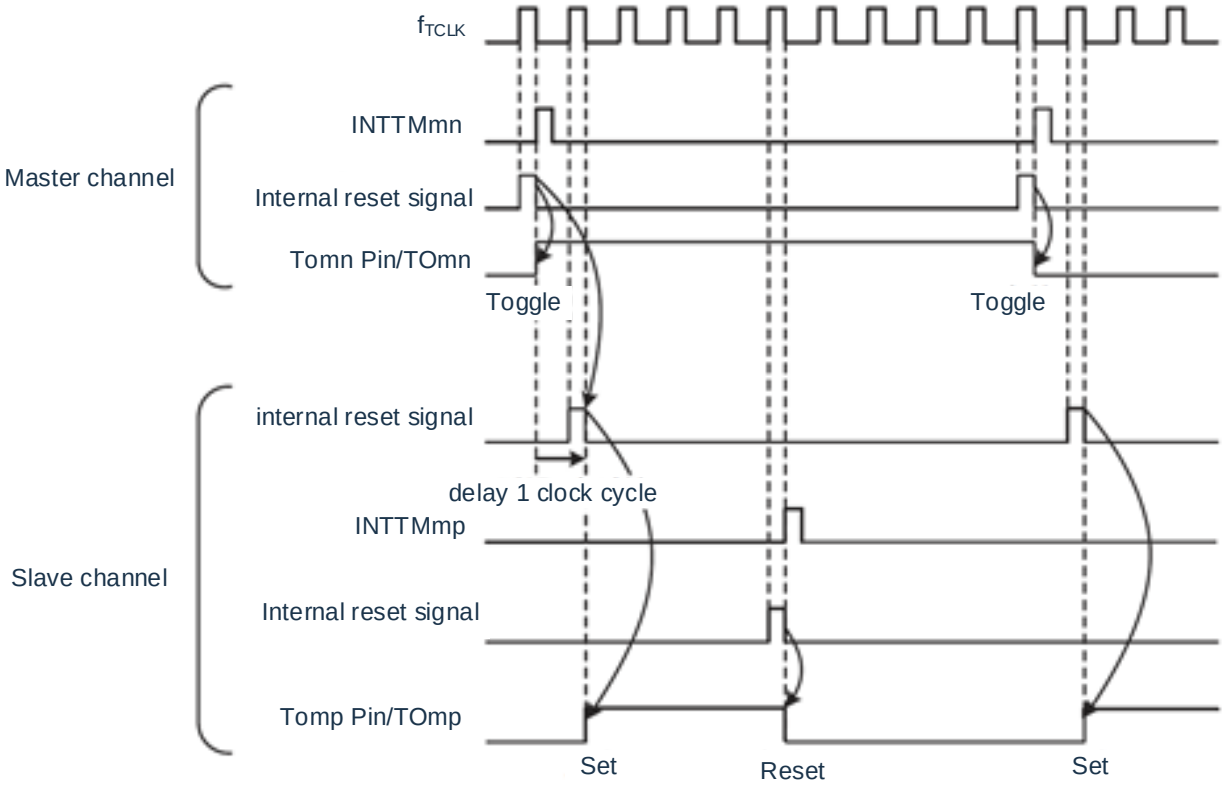
Slave channel: TOEmp=1, TOMmp=1, TOLmp=0

Remark:

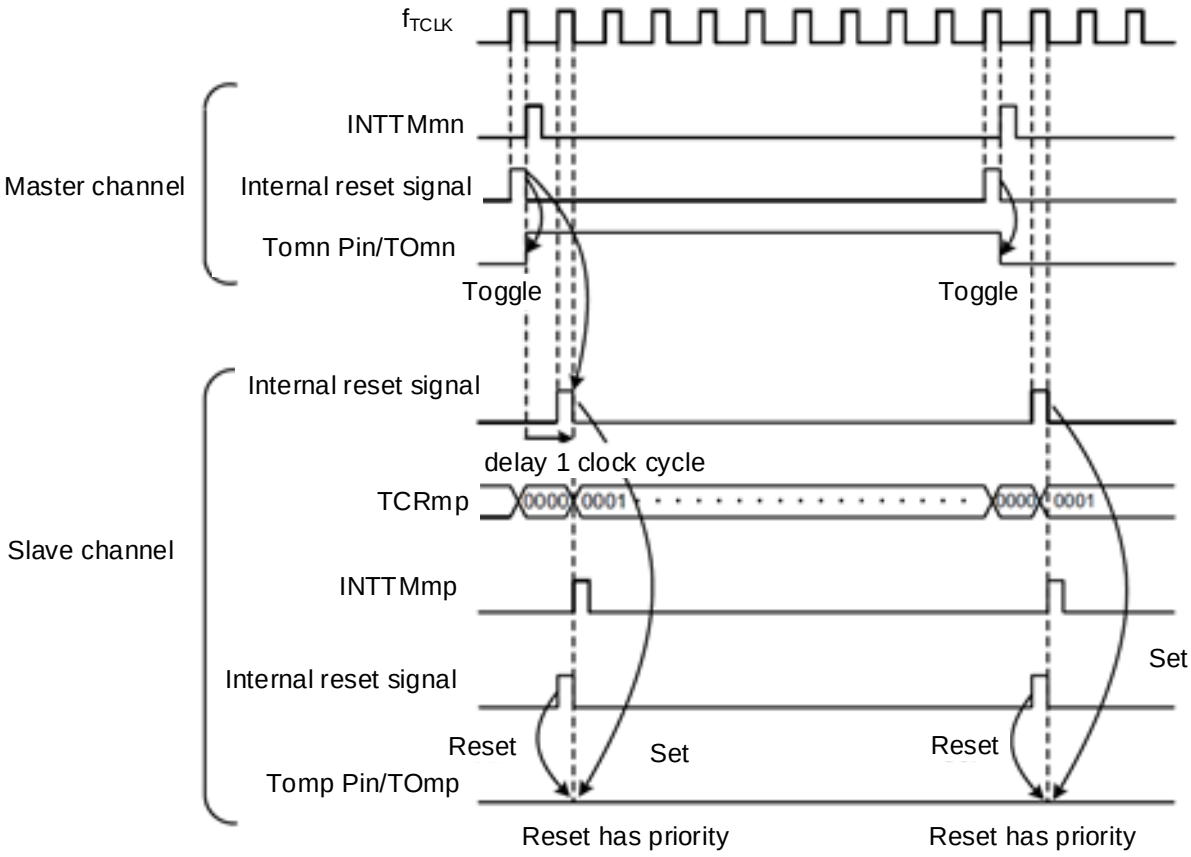
1. Set: The output signal from the TOmn pin changes from an invalid level to a valid level.
2. Reset: The output signal from the TOmn pin changes from a valid level to an invalid level.
3. m: unit number (m=0, 1) n: channel number (n=0~3)

Figure 5-15: Set/reset timing operation status

(1) Basic operation timing



(2) Operation timing when 0% duty cycle



Remark:

1. Internal reset signal: TOmn pin reset/toggle signal
2. Internal set signal: TOmn pin set signal
3. m: unit number (m=0, 1) n: channel number n=0~3 (master channel: n=0, 2) p: slave channel number (n=0: p=1, 2, 3 n=2: p=3)

5.6.4 One-time operation of TOmn bit

Like the timer channel start register m (TSm), the timer output register m (TOm) has the set bits (TOmn) for all channels and can therefore operate the TOmn bits for all channels at once.

Table 5-23: One-time operation example of TO0n bit

Before writing

TO0	0	0	0	0	0	0	0	0	0	0	0	0	TO03 1	TO02 0	TO01 1	TO00 0

TOE0	0	0	0	0	0	0	0	0	0	0	0	0	TOE03 0	TOE02 0	TOE01 0	TOE00 1

Data to be written

0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

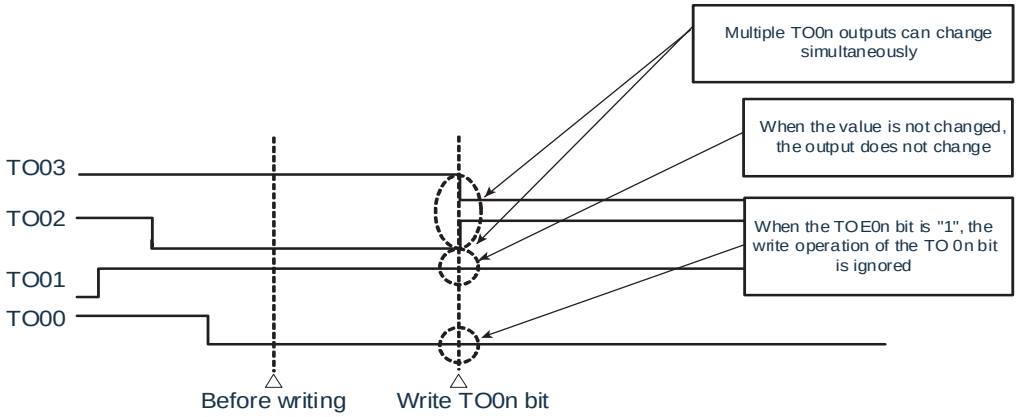
After writing

TO0	0	0	0	0	0	0	0	0	0	0	0	0	TO03 0	TO02 1	TO01 1	TO00 0

Only TOmn bits with TOEmn bit "0" can be written, and write is ignored when the TOmn bit is “1”.

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 5-16: TO0n pin state when the TO0n bit is operated at one time



Remark: m: unit number (m=0, 1) n: channel number (n=0~3)

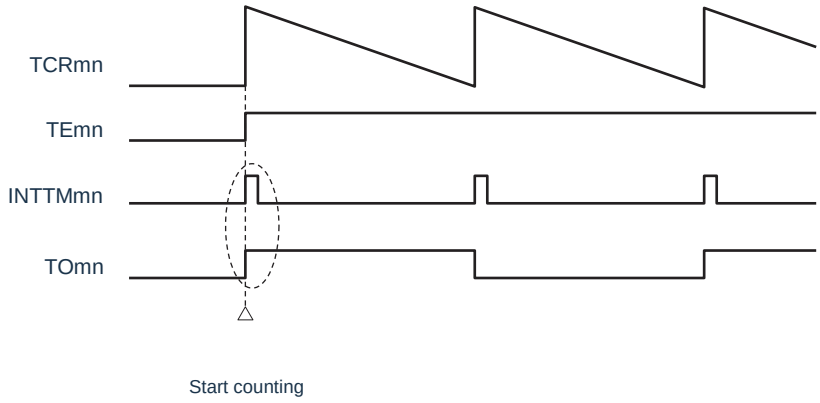
5.6.5 Timer interrupt and TOmn pin output when counting starts

In interval timer mode or capture mode, the MDmn0 bit of timer mode register mn (TMRmn) is the bit that sets whether to generate a timer interrupt when counting starts.

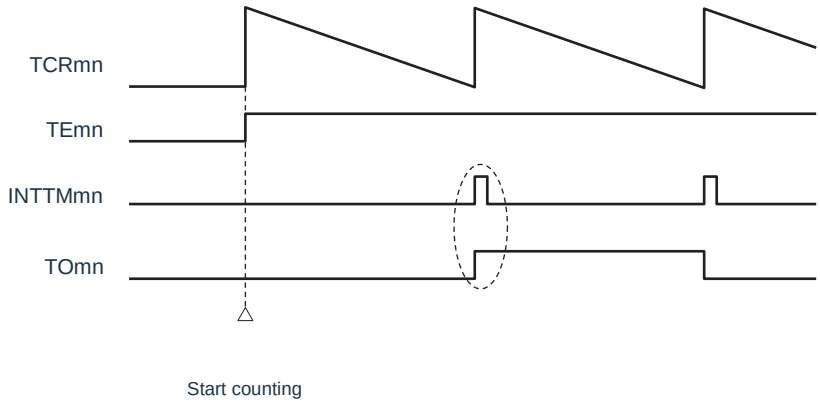
When the MDmn0 bit is "1", the start timing of the count can be known by generating a timer interrupt (INTTMmn). In other modes, the timer interrupt and TOmn output at the start of counting are not controlled. An example of operation when set to interval timer mode (TOEmn=1, TOMmn=0) is shown below.

Figure 5-17: An operation example of timer interrupt and TOmn output at start count

(a) When MDmn0 = 1



(b) When MDmn0 = 0



When MDmn0 bit is "1", the timer interrupt (INTTMmn) is output at the start of counting and TOmn is output alternately.

When MDmn0 bit is "0", no timer interrupt (INTTMmn) is output at the start of counting and TOmn is not changed, while INTTMmn is output and TOmn is alternately output after 1 cycle of counting.

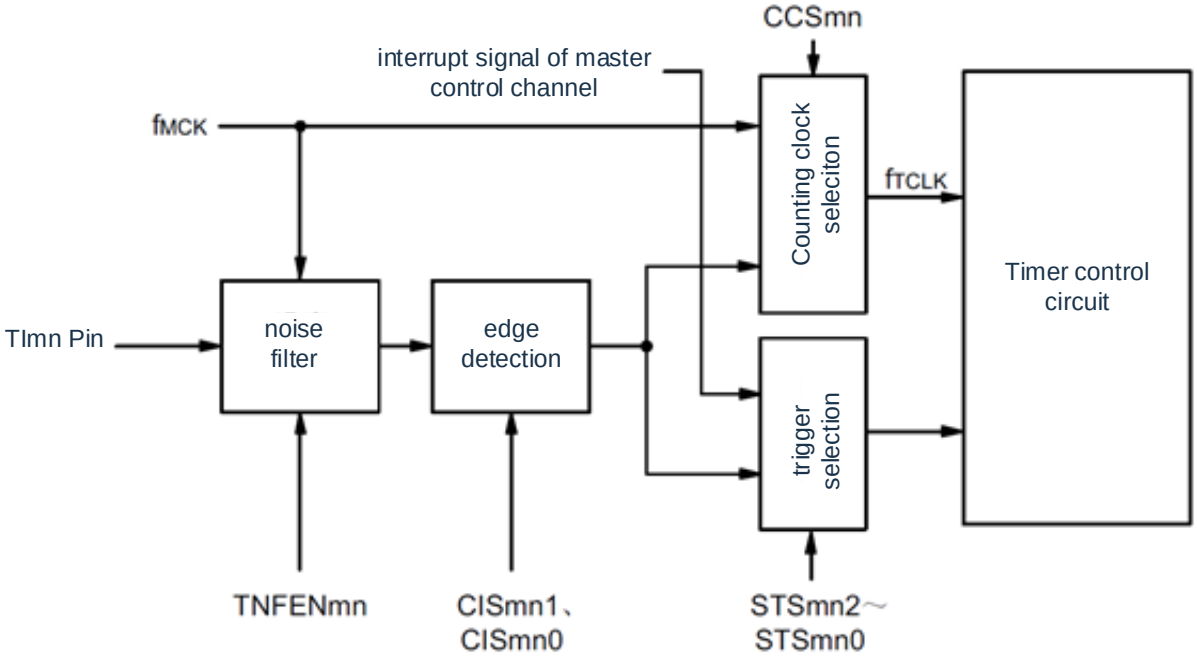
Remark: m: unit number (m=0, 1) n: channel number (n=0~3)

5.7 Control of timer input (TImn)

5.7.1 Structure of TImn pin input circuit

The signal from the timer input pins is input to the timer control circuit via a noise filter and the edge detection circuit. For pins that need to be removed from noise, the corresponding pin noise filter must be set to enable. The block diagram of the input circuit is as follows.

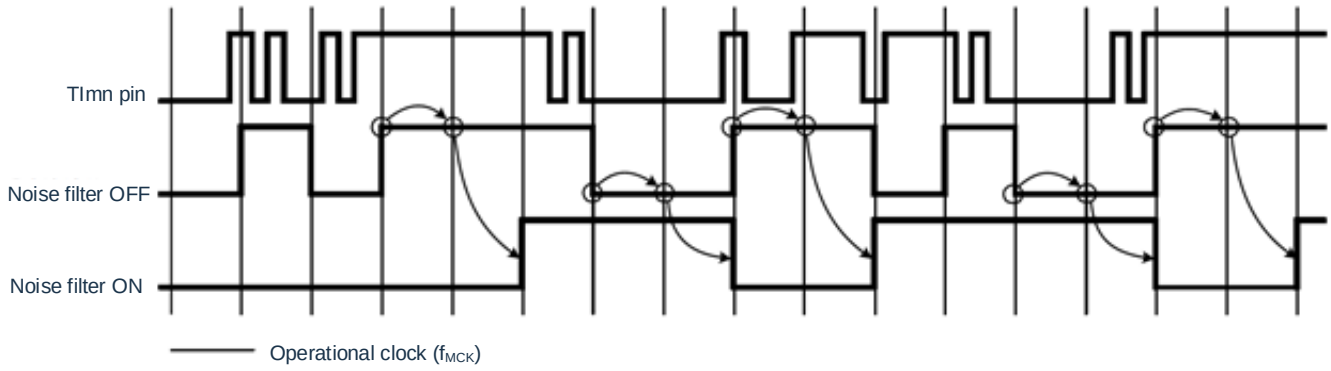
Figure 5-18: Structure of input circuit



5.7.2 Noise filter

When the noise filter is inactive, synchronization is performed only by the operation clock (F_{MCK}) of channel n. When the noise filter is active, 2 clocks are detected after synchronization by the operation clock (F_{MCK}) of channel n. The waveform of the TM4mn input pin after the noise filter circuit with the noise filter ON or OFF is shown below.

Figure 5-19: Sample waveform of TImn input pin with noise filter ON or OFF



Notice: The input waveform on the TImn pin is used to illustrate the operation of the noise filter is ON or OFF. For actual operation, the input must be made in accordance with the TImn input high- and low-level width shown in AC characteristics.

5.7.3 Cautions on channel input operation

When set to not use the timer input pin, no operating clock is provided to the noise filter circuit. Therefore, the following wait time is required from the time set to use the timer input pin to the time the channel corresponding to the timer input pin is set to operate the enable trigger.

(1) Noise filter OFF

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, 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 ON

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, 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).

5.8 Independent channel operation function of general-purpose timer unit

5.8.1 Operation as interval timer/square wave output

(1) Interval timer

It can be used as a reference timer to generate INTTMmn (timer interrupt) at fixed intervals. The interrupt generation period can be calculated using the following equation:

INTTMmn (timer interrupt) generation period = count clock period × (TDRmn set value +1)

(2) Operation as square wave output

The TOmn alternates outputs while generating the INTTMmn, outputting a square wave with a 50% duty cycle.

The period and frequency of the square wave can be calculated using the following equation:

Period of square wave output from TOmn = Period of count clock × (TDRmn set value +1) × 2

Frequency of square wave output from TOmn = Frequency of count clock / {(TDRmn set value +1) × 2}

In the interval timer mode, the timer count register mn (TCRmn) is used as a decrement counter.

After setting the channel start trigger bit (TSmn, TSHm1, TSHm3) of the timer channel start register m (TSm) to "1", the value of timer data register mn (TDRmn) is loaded into the TCRmn register by the first count clock. At this time, if the MDmn0 bit of the timer mode register n (TMRmn) is "0", INTTMmn is not output and TOmn is not alternately output. If the MDmn0 bit of TMRmn register is "1", INTTMmn is output and TOmn is alternately output. Then, the TCRmn register is decremented by the count clock.

If the TCRmn becomes "0000H", the INTTMmn and TOmn are output alternately by the next count clock. At the same time, the value of TDRmn register is loaded into TCRmn register again. After that, continue the same operation.

The TDRmn register can be rewritten at any time, and the rewritten TDRmn register value is valid from the next cycle.

Figure 5-20: Example of basic timing operating as an interval timer/square wave output (MDmn0=1)

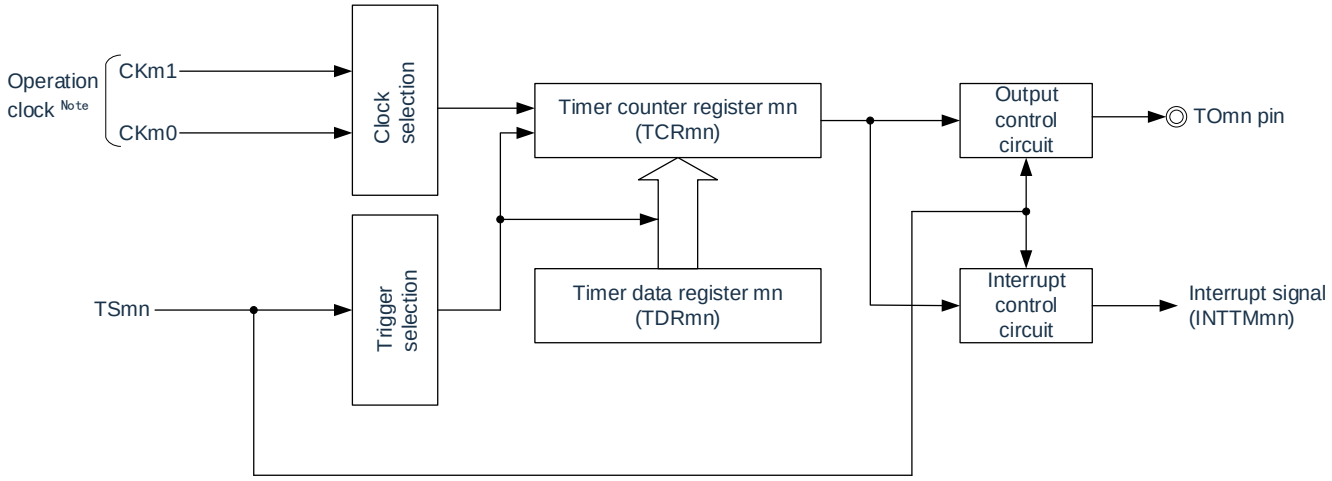
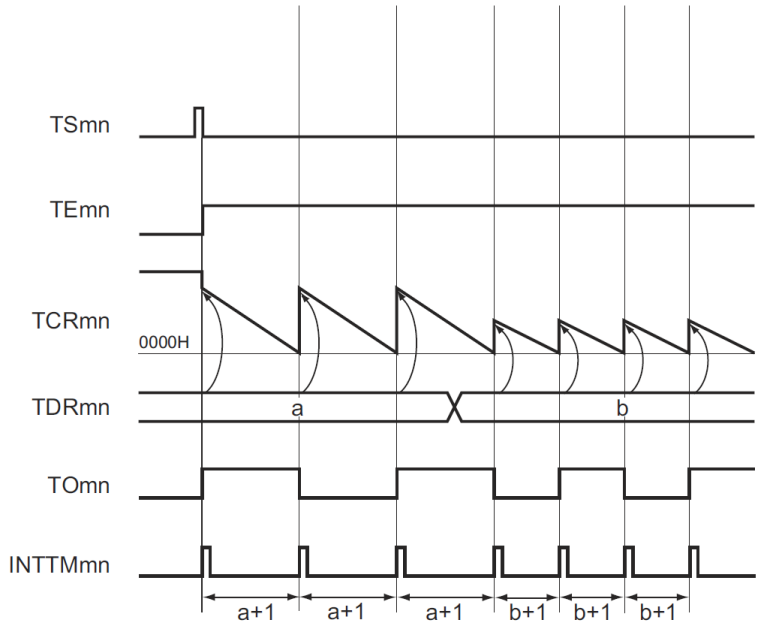


Figure 5-21: Example of basic timing operating as an interval timer/square wave output (MDmn0=1)



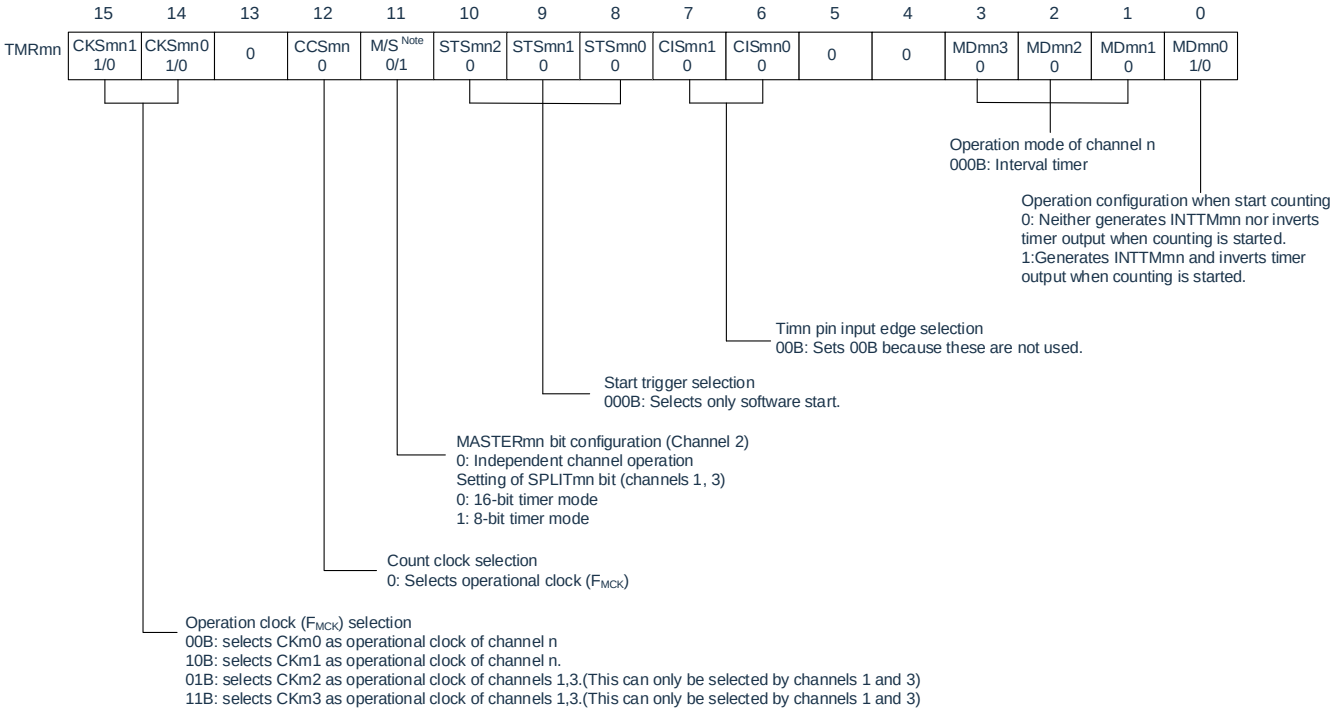
Note: At channel 1 and channel 3, it is possible to select the clock from CKm0, CKm1, CKm2 and CKm3.

Remark:

1. m: unit number (m=0, 1) n: channel number (n=0 ~ 3)
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 5-22: Example of register setting contents for interval timer/square wave output

(a) Timer mode register mn (TMRmn)



(b) Timer output register m (TOm)

	bit n	
TOm	TOmn 1/0	0: Outputs 0 from TOmn. 1: Outputs 1 from TOmn.

(c) Timer output enable register m (TOEm)

	bit n	
TOEm	TOEmn 1/0	0: Stops the TOmn output operation by counting operation. 1: Enables the TOmn output operation by counting operation.

(d) Timer output level register m (TOLm)

	bit n	
TOLm	TOLmn 0	0: 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: MASTERmn bit

Remark:

1. TMRm1, TMRm3: SPLITmn bits
2. TMRm0: Fixed to "0".
3. m: unit number (m= 0, 1) n: channel number (n=0 ~ 3)

Table 5-24: Procedure for interval timer/square wave output function

	Software operation	Hardware status
TAU initial settings		The input clock of timer unit m is in the stop-providing state. (Stop providing clock, cannot write to each register)
	Set the TM4mEN bit of peripheral enable register 0(PER0) to "1". →	The input clock of timer unit m is in the providing state. (Start providing clock, can write to each register)
	Set the timer clock selection register m (TPSm). Determine the clock frequency of CKm0 ~ CKm3.	
Initial setting of channels	Set the timer mode register mn (TMRmn) (to determine the channel's operation mode). Set the interval (period) value for the timer data register mn (TDRmn).	The channel is in the stop state. (Provides clock, and consumes some power)
	Using TOmn output: Set the TOMmn bit of Timer Output Mode Register m (TOMm) to "0" (master channel output mode). Set the TOLmn bit to "0". Set the TOmn bit to determine the initial level of the TOmn output. Set the TOEmn bit to "1" and enable TOmn output. Set the Port Register and Port Mode Register to "0". →	The TOmn pin is in Hi-Z output state. When the port mode register is in output mode and the port register is "0", the TOmn initial set level is output. The TOmn remains unchanged because the channel is in the stop state. The TOmn pin outputs the level set by the TOmn.
Start operate	(The TOEmn bit set to "1" only when the TOmn output is used and restarted) Set the TSmn bit to "1". Since the TSmn bit is a trigger bit, it automatically returns to "0". →	The TEmn bit becomes "1" and starts counting. Load the value of the TDRmn register into the Timer Count Register mn (TCRmn). When the MDmn0 bit of TMRmn register is "1", INTTMmn is generated and TOmn is output alternately.
In operation	The setting of the TDRmn register can be changed at will. The TCRmn register can be read at any time. The TSRmn register is not used. The TOm register and TOEm register settings can be changed. The setting of the TMRmn register, the TOMmn bit and the TOLmn bit cannot be changed.	The counter (TCRmn) performs decremental counting. If the count reaches "0000H", the value of the TDRmn register is loaded into the TCRmn register again and the count continues. When TCRmn is detected as "0000H", INTTMmn is generated and TOmn is alternately output. Thereafter, repeat this operation.
Stop operation	Set the TTmn bit to "1". The operation automatically returns to "0" because the TTmn bit is a trigger bit. →	The TEmn bit becomes "0" and stops counting. The TCRmn register holds the count value and stops counting. The TOmn output is not initialized but remains its state.
	Set the TOEmn bit to "0" and set the value for the TOmn bit. →	The TOmn pin outputs the level set by the TOmn bit.
TAU stop	To maintain the output level of the TOmn pin: Set TOmn bit to "0" after setting the value to be held for the port register. → No need to maintain the output level of the TOmn pin: No need to set.	The output level of the TOmn pin is maintained by the port function.
	Set the TM4mEN bit of the PER0 register to "0"。 →	The input clock of timer unit m is in the stop-providing state. Initialize all circuits and the SFR for each channel. (TOmn bit becomes "0" and TOmn pin becomes port function)

Restart the operation →

Remark: m: unit number (m= 0, 1) n: channel number (n=0 ~ 3)

5.8.2 Operation as external event counter

It can be used as an event counter to count the active edges (external events) detected on the TImn pin input and generate an interrupt if the specified count value is reached. The specified count value can be calculated using the following equation:

```
Specified count value = TDRmn set value + 1
```

In the event counter mode, the timer count register mn (TCRmn) is used as a decrement counter.

The value of timer data register mn (TDRmn) is loaded into the TCRmn register by setting any channel start trigger bit (TSmn, TSHm1, TSHm3) of timer channel start register m (TSm) to "1".

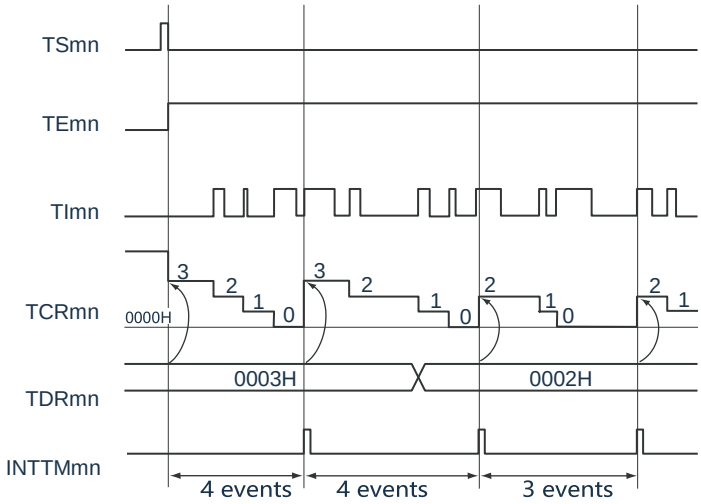
The TCRmn register decrements the count while detecting the active edge of the TImn pin input. If TCRmn becomes "0000H", the value of TDRmn register is loaded again and INTTMmn is output.

After that, continue the same operation.

The output must be stopped by setting the TOEmn bit of the timer output enable register m (TOEm) to "0" because the TOmn pin outputs irregular waveforms based on external events.

The TDRmn register can be rewritten at any time, and the rewritten TDRmn register value is valid for the next cycle.

Figure 5-23: Example of basic timing operating as external event counter

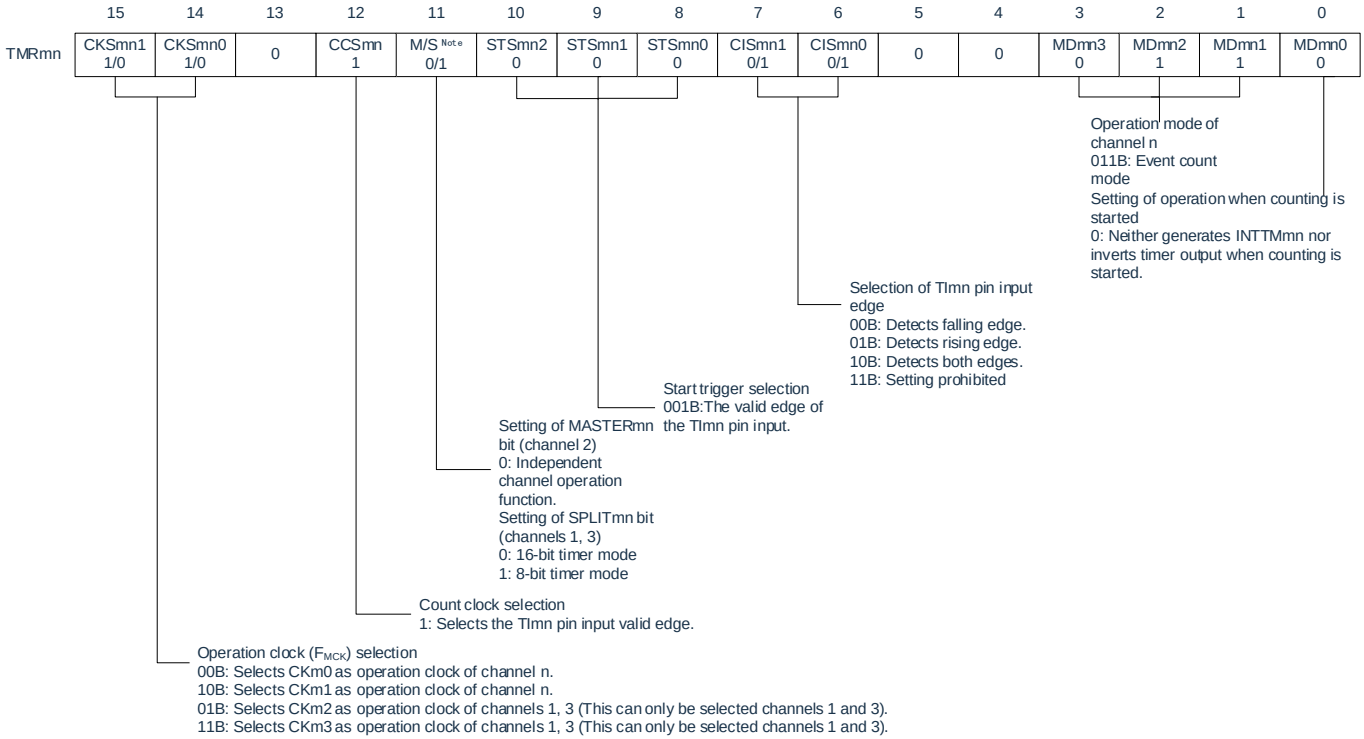


Remark:

1. m: unit number (m=0, 1) n: channel number (n=0 ~ 3)
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 5-24: Example of register contents setting in external event counter mode

(a) Timer mode register mn (TMRmn)



(b) Timer output register m (TOm)

	bit n	
TOm	TOmn 1/0	0: Outputs 0 from TOmn.

(c) Timer output enable register m (TOEm)

	bit n	
TOEm	TOEmn 1/0	0: Stops TOmn output performed by the counting operation. 1: Enables TOmn output performed by the counting operation.

(d) Timer output level register m (TOLm)

	bit n	
TOLm	TOLmn 0	Sets "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: MASTERmn bit

Remark:

1. TMRm1, TMRm3: SPLITmn bits
2. TMRm0: Fixed to "0".
3. m: unit number (m= 0, 1) n: channel number (n=0 ~ 3)

Table 5-25: Procedure for external event counter function

	Software operation	Hardware status
Timer4 initial settings		The input clock of timer unit m is in the stop-providing state. (Stop providing clock, cannot write to each register)
	Set the TM4mEN bit of peripheral enable register 0(PER0) to "1".	The input clock of timer unit m is in the providing state and the channels are in the stop state. (Start providing clock, can write to each register)
	Set the timer clock selection register m (TPSm). Determine the clock frequency of CKm0 ~ CKm3.	
Initial setting of channels	Set the corresponding bit of the Noise Filter Enable Register (NFEN1) to "0" (OFF) or "1" (ON). Set the timer mode register mn (TMRmn)(to determines the operating mode of the channel). Set the count value for the timer data register mn (TDRmn). Set the TOEmn bit of the timer output enable register m (TOEm) to "0".	The channel is in the stop state. (Provides clock, and consumes some power)
Start operating	Set the TSmn bit to "1". Since the TSmn bit is a trigger bit, it automatically returns to "0".	The TEmn bit becomes "1" and starts counting. The value of the TDRmn register is loaded into the timer count register mn (TCRmn) and enter the detection wait state of the input edge of the TImn pin.
In operation	The setting of the TDRmn register can be changed at will. The TCRmn register can be read at any time. The TSRmn register is not used. The setting of the TMRmn register, the TOMmn bit, the TOLmn bit, the Tomn bit and the TOEmn bit cannot be changed.	Whenever the input edge of the TImn pin is detected, the counter (TCRmn) is decremented. If the count reaches "0000H", the value of the TDRmn register is loaded into the TCRmn register again and the count continues. When TCRmn is detected as "0000H", INTTMmn is generated. Thereafter, repeat this operation.
Stop operating	Set the TTmn bit to "1". The operation automatically returns to "0" because the TTmn bit is a trigger bit.	The TEmn bit becomes "0" and stops counting. The TCRmn register holds the count value and stops counting.
Timer4 stop	Set the TA4mEN bit of the PER0 register to "0".	The input clock of timer unit m is in the stop-providing state. Initialize all circuits and the SFR for each channel.

Restart operation

5.8.3 Operation as frequency divider

The clock input from the TI00 pin can be divided and used as a divider for the output of the TO00 pin.

The divided clock frequency of the TO00 output can be calculated using the following equation:

- Select rising or falling edge:
  Divider clock frequency = input clock frequency/ {(TDR00 set value+1) × 2}
- Select both edges:
  Divider clock frequency≈input clock frequency/ (TDR00 set value+1)

In the interval timer mode, the timer count register 00 (TCR00) is used as a decrement counter.

After setting the channel start trigger bit (TS00) of timer channel start register 0 (TS0) to "1", the value of timer data register 00 (TDR00) is loaded into the TCR00 register by detecting an active edge of TI00. At this time, if the MD000 bit of Timer Mode Register 00 (TMR00) is "0", INTTM00 is not output and TO00 is not output alternately; if the MD000 bit of TMR00 register is "1", INTTM00 is output and TO00 is not output alternately. If the MD000 bit of TMR00 register is "1", INTTM00 is output and TO00 is output alternately.

The TCR00 register then counts down through the active edge of the TI00 pin input. If TCR00 changes to "0000H", TO00 performs an alternate output. At the same time, the value of the TDR00 register is loaded into the TCR00 register and counting continues.

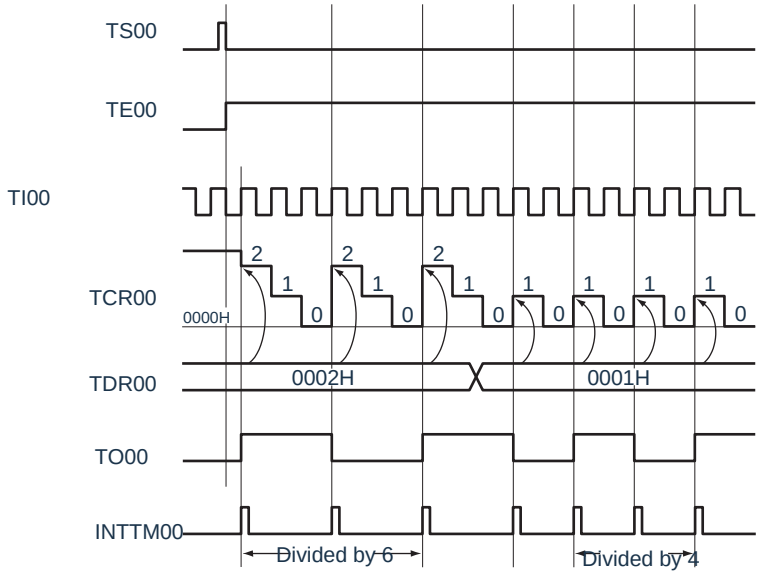
If double edge detection is selected for the TI00 pin input, the duty cycle error of the input clock affects the clock period of the TO00 output's division.

The clock period of the TO00 output contains the sampling error of 1 run clock cycle.

clock period of the TOmn output = supposed TOmn output clock period ± operating clock period (error)

The TDRmn register can be rewritten at any time, and the rewritten TDRmn register value is valid for the next cycle.

Figure 5-25: Example of basic timing operating as a frequency divider (MD000=1)



Remark: TS00: Bit 0 of timer channel start register 0 (TS0)

TE00: Bit 0 of timer channel enable status register (TE0)

TI00: TI00 pin input signal

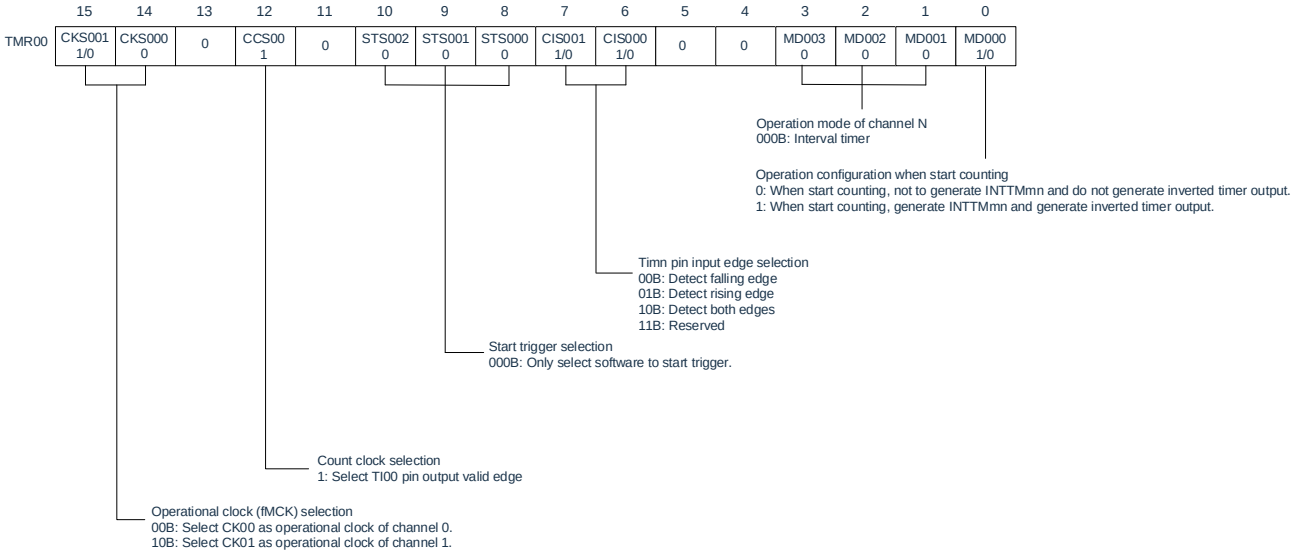
TCR00: Timer count register 00 (TCR00)

TDR00: Timer data register 00 (TDR00)

TO00: TO00pin output signal

Figure 5-26: Example of register contents setting when operating as a frequency divider

(a) Timer mode register 00 (TMR00)



(b) Timer output register 0 (TO0)

	bit 0	
TO0	TO00 1/0	0: Outputs "0" by TO00. 1: Outputs "1" by TO00.

(c) Timer output enable register 0 (TOE0)

	bit 0	
TOE0	TOE00 1/0	0: Stops the TO00 output performed by the counting operation. 1: Enables TO00 output performed by the counting operation.

(d) Timer output level register 0 (TOL0)

	bit 0	
TOL0	TOL00 0	0: Sets "0" in the master channel output mode (TOM00=0).

(e) Timer output mode register 0 (TOM0)

	bit n	
TOM0	TOM00 0	0: Sets master channel output mode.

Table 5-26: Procedure for frequency divider function

	Software operation	Hardware status
Timer4 initial settings		The input clock of timer unit 0 is in the stop-providing state. (Stop providing clock, cannot write to each
	Set the TM4mEN bit of peripheral enable register 0(PER0) to "1".	The input clock of timer unit 0 is in the providing state and the channels are in the stop state. (Start providing clock, can write to each register)
	Set the timer clock selection register 0 (TPS0). Determine the clock frequency of CK00 ~ CK03.	
Initial setting of channels	Set the corresponding bit of the Noise Filter Enable Register (NFEN1) to "0" (OFF) or "1" (ON). Set the timer mode register 00 (TMR00)(to determines the operating mode of the channel, select edge detection). Set the interval (cycles) value for the timer data register 00 (TDR00)	The channel is in the stop state. (Provides clock, and consumes some power)
	Set TOM00 bit of timer output mode register 0 (TOM0) to “0”(master control channel output mode). The TOL00 bit must be set to "0". Set the TO00 bit and determine the initial level of TO00 output Set TOE00 bit to "1" and enable TO00 output. Set the Port Register and Port Mode Register to "0”	The TO00 pin is in Hi-Z output state. When the port mode register is in output mode and the port register is "0", the TO00 initial set level is output. Since the channel is in stop state, TO00 does not change. The TO00 pin outputs the level set by the TO00.
Start operation	Set TOE00 bit to “1” (only limited to restart operation). The TS00 bit must be set to "1". The operation automatically returns to "0" because the TS00 bit is a trigger bit.	The TE00 bit becomes "1" and starts counting. Load the value of the TDR00 register into the Timer Count Register 00 (TCR00). When the MD000 bit TMR00 register is "1", INTTM00 is generated and TO00 is output alternately.
	The setting of the TDR00 register can be changed at will. The TCR00 register can be read at any time. The TSR00 register is not used. The TO0 register and TOE0 register settings can be changed. The setting of the TMR00 register, the TOM00 bit and the TOL00 bit cannot be changed.	The counter (TCR00) performs decremental counting. If the count reaches "0000H", the value of the TDR00 register is loaded into the TCR00 register again and the count continues. When the TCR00 bit is "0000H", INTTM00 is generated and TO00 is output alternately. Thereafter, repeat this operation.
Stop operating	The TT00 bit must be set to "1". The operation automatically returns to "0" because the TT00 bit is a trigger bit.	The TE00 bit becomes "0" and starts counting. The TCR00 register holds the count value and stops counting. The TO00 output is not initialized but remains its
	Set the TOE00 bit to "0" and set the value for the TO00	The TO00 pin outputs the level set by the TO00.
Timer4 stop	To maintain the output level of the TO00 pin: Set TO00 bit to "0" after setting the value to be held for the port register. No need to maintain the output level of the TO00 pin: No need to set.	The output level of the TO00 pin is maintained by the port function.
	Set the TM4mEN bit of the PER0 register to "0".	The input clock of timer unit 0 is in the stop-providing state. Initialize all circuits and the SFR for each channel. (TO00 bit becomes "0" and TO00 pin becomes port function)

Restart operation

5.8.4 Operation as input pulse interval measurement

The count value can be captured at the active edge of TImn and the interval between TImn input pulses can be measured. The software operation (TSmn=1) can also be set to capture the count value during the period when the TEmn bit is "1".

The pulse interval can be calculated using the following equation:

TImn input pulse interval = period of counting clock × ((10000H × TSRmn: OVF) + (TDRmn captured value +1))

In capture mode, the timer count register mn (TCRmn) is used as an increment counter.

If the channel start trigger bit (TSmn) of the timer channel start register m (TSm) is set to "1", the TCRmn register is incrementally counted from "0000H" by the count clock.

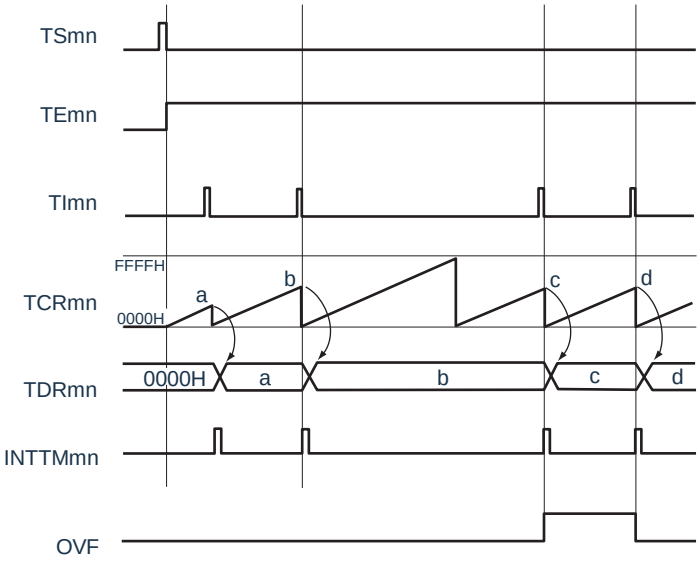
If the active edge of the TImn pin input is detected, the count value of TCRmn register is transferred (captured) to Timer Data Register mn (TDRmn), and the TCRmn register is cleared to "0000H", and then INTTMmn is output. If the counter overflows, the OVF bit of Timer Status Register mn (TSRmn) is set to "1". If the counter does not overflow, the OVF bit is cleared. After that, continue the same operation.

While capturing the count value to the TDRmn register, the OVF bit of the TSRmn register is updated according to whether or not overflow occurs during the measurement, and the overflow status of the captured value can be confirmed.

Even if the counter counts 2 or more complete cycles, the overflow is considered to have occurred and the OVF bit of the TSRmn register is set to "1". However, when two or more overflows occur, the interval value cannot be measured normally by the OVF bit.

Set the STSmn2~STSmn0 bit of the TMRmn register to "001B", and use the valid edge of TImn for start trigger and capture trigger.

Figure 5-27: Example of basic timing operating as an input pulse interval measurement (MDmn0=0)



Remark:

1. The 1 operation clock error is generated because the TImn pin input is sampled by the operation clock selected by the CKSmn bit of the Timer Mode Register mn (TMRmn).
2. m: unit number (m= 0) n: channel number (n=0 ~ 3)
3. 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

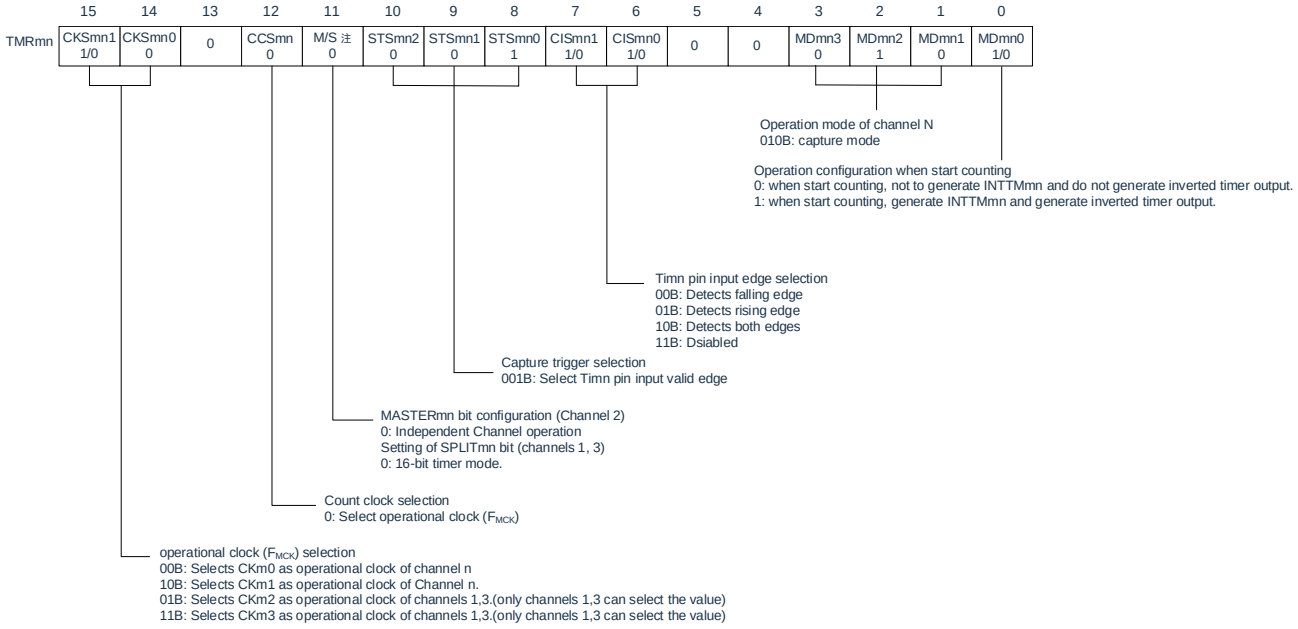
TCRmn: Timer count register mn (TCRmn)

TDRmn: Timer data register mn (TDRmn)

OVF: Bit0 of timer status register mn (TSRmn)

Figure 5-28: Example of register contents setting in measuring input pulse interval

(a) Timer mode register mn (TMRmn)



(b) Timer output register m (TOm)

	bit n	
TOm	TOmn 0	0: Outputs "0" by TOmn.

(c) Timer output enable register m (TOEm)

	bit n	
TOEm	TOEmn 0	0: Stops TOmn output performed by the counting operation.

(d) Timer output level register m (TOLm)

	bit n	
TOLm	TOLmn 0	0: Set “0” in master channel output mode (TOMmn=0).

(e) Timer output mode register m (TOMm)

	bit n	
TOMm	TOMmn 0	0: Sets master channel output mode.

Note: TMRm2: MASTERmn bit

Remark:

1. TMRm1, TMRm3: SPLITmn bits
2. TMRm0: Fixed to “0”.
3. m: unit number (m=0) n: channel number (n=0 ~ 3)

Table 5-27: Procedure for input pulse interval measurement function

	Software operation	Hardware status
Timer4 initial settings		The input clock of timer unit m is in the stop-providing state. (Stop providing clock, cannot write to each register)
	Set the TM4mEN bit of the peripheral enable register 0(PER0) to "1". →	The input clock of timer unit m is in the providing state and the channels are in the stop state. (Start providing clock, can write to each register)
	Set the timer clock selection register m (TPSm). Determine the clock frequency of CKm0 ~ CKm3.	
Initial setting of channels	Set the corresponding bit of the Noise Filter Enable Register (NFEN1) to "0" (OFF) or "1" (ON). Set the timer mode register mn (TMRmn)(to determines the operating mode of the channel).	The channel is in the stop state. (Provides clock, and consumes some power)
Start operation	Set the TSmn bit to "1". → Since the TSmn bit is a trigger bit, it automatically returns to "0".	The TEmn bit becomes "1" and starts counting. Clear the timer count register mn (TCRmn) to "0000H". When the MDmn0 bit of TMRmn register is "1", INTTMmn is generated.
In operation	The setting values of the CISmn1 bit and the CISmn0 bit of the TMRmn register can be changed. The TDRmn register can be read at any time. The TCRmn register can be read at any time. The TSRmn register can be read at any time. The setting of the the TOMmn bit, the TOLmn bit, the TOmn bit and the TOEmn bit cannot be changed.	The counter (TCRmn) starts incremental counting from "0000H" and transfers (captures) the count value to the timer data register mn (TDRmn) if it detects an active edge on the TImn pin input or sets TSmn bit to "1". If the counter overflows, set the OVF bit of Timer Status Register mn (TSRmn). If the counter does not overflow, the OVF bit is cleared. Thereafter, repeat this operation.
Stop operation	Set the TTmn bit to "1". → The operation automatically returns to "0" because the TTmn bit is a trigger bit.	The TEmn bit becomes "0" and stops counting. The TCRmn register holds the count value and stops counting. The OVF bit of the TSRmn register remains unchanged.
Timer4 stop	Set the TM4mEN bit of the PER0 register to "0". →	The input clock of timer unit m is in the stop-providing state. Initialize all circuits and the SFR for each channel.

Restart operation

Remark: m: unit number (m= 0, 1) n: channel number (n=0 ~ 3)

5.8.5 Operation as input signal high-/low-level width measurement

The signal width (high-/low-level width) of TImn can be measured by starting counting at one edge of the input to the TImn pin and capturing the count value at the other edge. The TImn signal width of the TImn output can be calculated using the following equation:

Signal width of TImn input = period of count clock × ((10000H × TSRmn: OVF) + (TDRmn captured value +1))

In the Capture & Single Count mode, the timer count register mn (TCRmn) is used as an increment counter. If the channel start trigger bit (TSmn) of the timer channel start register m(TSm) is set to "1", the TEmn bit becomes "1", and the start edge detection wait state of the TImn pin is entered.

If the start edge of the TImn pin input (rising edge of the TImn pin input at the time of high-level width measurement) is detected, it is synchronized with the count clock and counts incrementally from "0000H". Then, if an active capture edge is detected (falling edge of TImn pin input at the time of high-level width measurement), the count value is transferred to the Timer Data Register mn (TDRmn) and INTTMmn is output at the same time. If the counter overflows, the OVF bit of the Timer Status Register mn (TSRmn) is set to "1". If the counter does not overflow, the OVF bit is cleared. The value of the TCRmn register changes to "Value passed to TDRmn register +1", and the start edge detection wait state of the TImn pin is entered. After that, continue the same operation.

While capturing the count value to the TDRmn register, the OVF bit of the TSRmn register is updated according to whether or not overflow occurs during the measurement, and the overflow status of the captured value can be confirmed.

Even if the counter counts 2 or more complete cycles, the overflow is considered to have occurred and the OVF bit of the TSRmn register is set to "1". However, when two or more overflows occur, the interval value cannot be measured normally by the OVF bit.

The CISmn1 and CISmn0 bits of the TMRmn register can be used to set whether the high-level width or low-level width of the TImn pin is to be measured. This function is designed to measure the input signal width of the TImn pin, so the TSmn bit cannot be set to "1" during the period when the TEmn bit is "1".

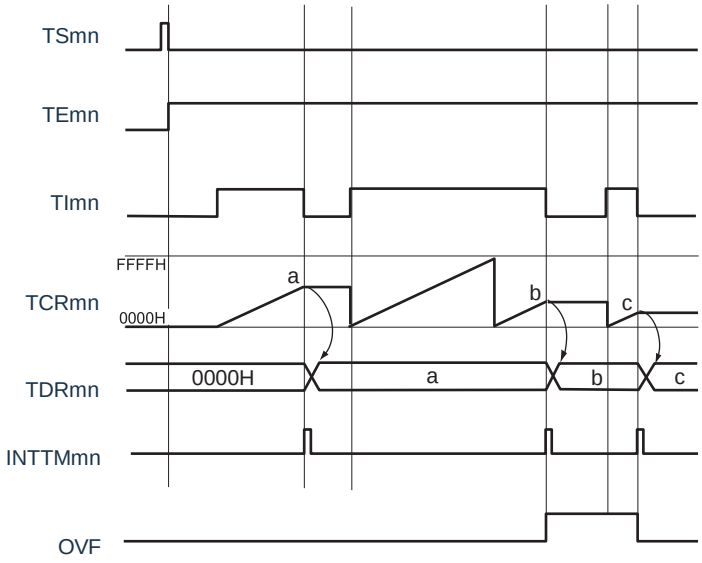
CISmn1, CISmn0=10B of the TMRmn register: Measures the low-level width.

CISmn1, CISmn0=11B of the TMRmn register: Measures the high-level width.

Notice:

1. When used as a LIN-bus support function, bit1 (ISC1) of the Input Switching Control Register (ISC) must be set to "1" and RxD0 should be used instead of TImn in the following description.
2. Because the TImn pin inputs are sampled by the operation clock selected by the CKSmn bit of the Timer Mode Registermn (TMRmn), an error of 1 operation clock is generated.

Figure 5-29: Example of basic timing operating as high-/low-level width measurement of input signal

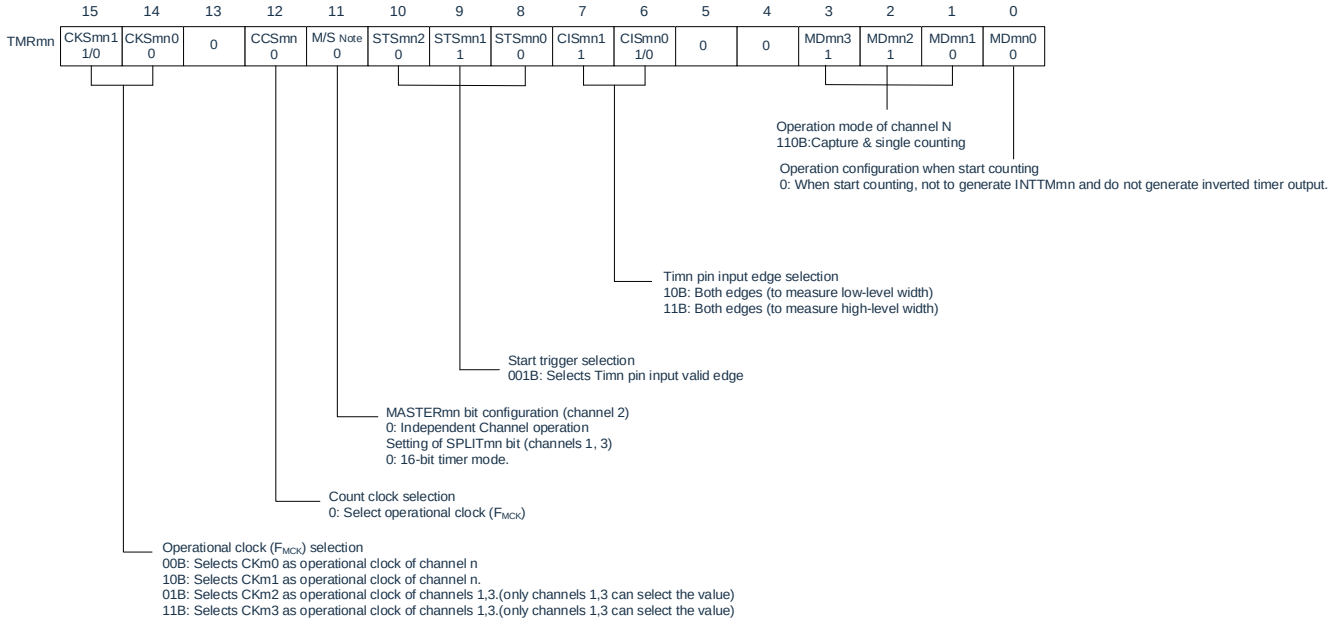


Remark:

1. m: unit number (m= 0) n: channel number (n=0 ~ 3)
2. TSmn: Bit n of timer channel start register m (TSm)
   TEmn: Bit n of timer channel enable status register 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 5-30: Example of register contents setting in measuring high-/low-level width of input signal

(a) Timer mode register mn (TMRmn)



(b) Timer output register m (TOm)

	bit n	
TOm	TOmn 0	0: Outputs 0 from TOmn.

(c) Timer output enable register m (TOEm)

	bit n	
TOEm	TOEmn 0	0: Stops TOmn output performed by the counting operation.

(d) Timer output level register m (TOLm)

	bit n	
TOLm	TOLmn 0	0: Set “0” in master channel output mode (TOMmn=0).

(e) Timer output mode register m (TOMm)

	bit n	
TOMm	TOMmn 0	0: Sets master channel output mode.

Note: TMRm2: MASTERmn bit

Remark:

1. TMRm1, TMRm3: SPLITmn bits
2. TMRm0: Fixed to “0”.
3. m: unit number (m=0) n: channel number (n=0 ~ 3)

Table 5-28: Procedure for high-/low-level width measurement function of input signal

	Software operation	Hardware status
Timer4 initial settings		The input clock of timer unit m is in the stop-providing state. (Stop providing clock, cannot write to each register)
	Set the TM4mEN bit of the peripheral enable register 0(PER0) to "1". →	The input clock of timer unit m is in the providing state and the channels are in the stop state. (Start providing clock, can write to each register)
	Set the timer clock selection register m (TPSm). Determine the clock frequency of CKm0 ~ CKm3.	
Initial setting of channels	Set the corresponding bit of the Noise Filter Enable Register (NFEN1) to "0" (OFF) or "1" (ON). Set the timer mode register mn (TMRmn)(to determines the operating mode of the channel).	The channel is in the stop state. (Provides clock, and consumes some power)
Start Operation	Set the TSmn bit to "1". → Since the TSmn bit is a trigger bit, it automatically returns to "0".	The TEmn bit changes to "1" and enters the detection wait state for start triggering (detecting the active edge of the TImn pin input or setting the TSmn bit to "1").
	Detect TImn pin input counting start edge. →	Clear timer count register mn (TCRmn) to "0000H" and start incremental counting.
In operation	The setting of the TDRmn register can be changed at will. The TCRmn register can be read at any time. The TSRmn register is not used. The setting of the TMRmn register, the TOMmn bit, the TOLmn bit, the Tomn bit and the TOEmn bit cannot be changed.	After the start edge of the TImn pin is detected, the counter (TCRmn) starts counting incrementally from "0000H". If the capture edge of the TImn pin is detected, the count value is transferred to the timer data register mn (TDRmn), and INTTMmn is generated. If the counter overflows, set the OVF bit of Timer Status Register mn (TSRmn). If the counter does not overflow, the OVF bit is cleared. The TCRmn register stops counting before the start edge of the next TImn pin is detected. Thereafter, repeat this operation.
Stop operation	Set the TTmn bit to "1". → The operation automatically returns to "0" because the TTmn bit is a trigger bit.	The TEmn bit becomes "0" and stops counting. The TCRmn register holds the count value and stops counting. The OVF bit of the TSRmn register remains unchanged.
Timer4 stop	Set the TM4mEN bit of the PER0 register to "0". →	The input clock of timer unit m is in the stop providing state. Initialize all circuits and the SFR for each channel.

Restart operation →

Remark: m: unit number (m= 0, 1) n: channel number (n=0 ~ 3)

5.8.6 Operation as delay counter

The count can be decremented by the active edge detection (external event) of the TImn pin input and INTTTMmn (timer interrupt) is generated at any set interval.

During the period when the TEmn bit is "1", the TSmn bit can be set to "1" by software to start decreasing counting and generate INTTMmn (timer interrupt) at any set interval.

The interrupt generation period can be calculated using the following equation:

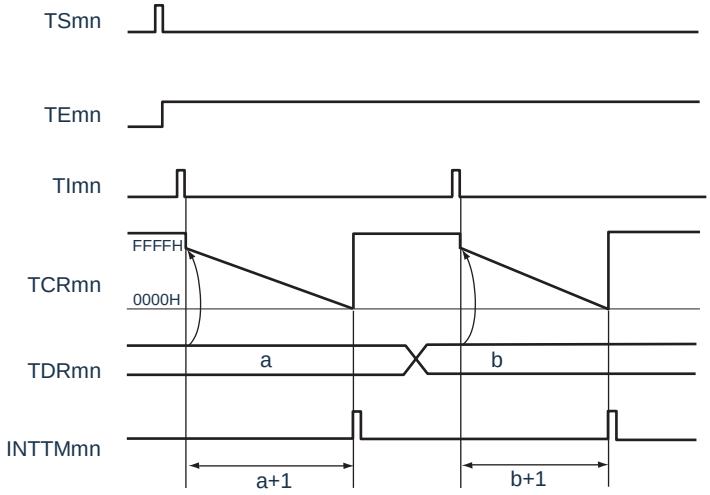
INTTMmn (timer interrupt) generation period = counting clock period × (TDRmn `set value` +1)

In the single count mode, the timer count register mn (TCRmn) is used as a decrement counter.

If the channel start trigger bit (TSmn, TSHm1, TSHm3) of the timer channel start register m(TSm) is set to "1", the TEmn bit, TEHm1 bit, TEHm3 bit become "1", and the active edge detection wait state of the TImn pin is entered. An active edge detection via the TImn pin input starts the TCRmn register and loads the value of the Timer Data Register mn (TDRmn). The TCRmn register counts decreasingly from the value of the loaded TDRmn register by counting the clock. If TCRmn becomes "0000H", INTTMmn is output and counting is stopped until the next active edge of the TImn pin input is detected.

The TDRmn register can be rewritten at any time, and the rewritten TDRmn register value is valid from the next cycle.

Figure 5-31: Example of basic timing operating as a delay counter

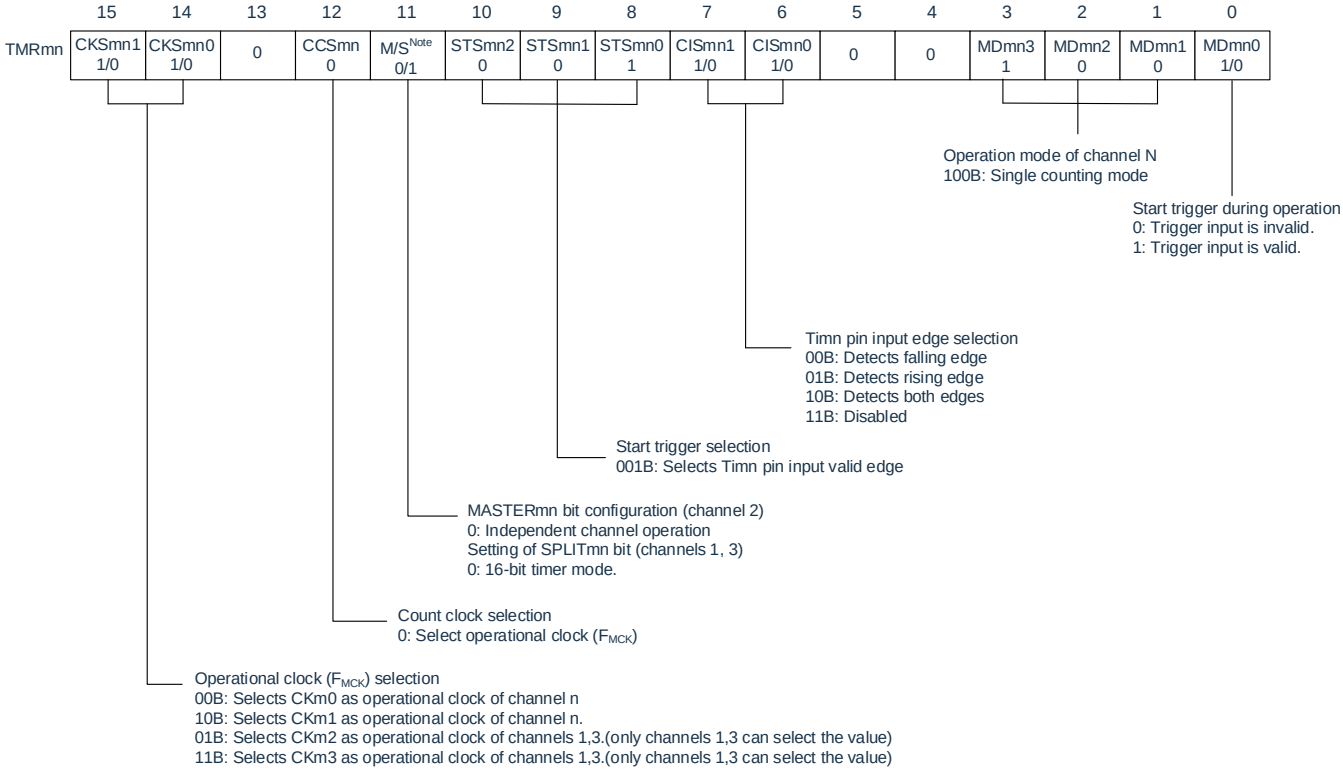


Remark:

1. m: unit number (m=0, 1) n: channel number (n=0~3)
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 5-32: Example of register contents setting for delay counter function

(a) Timer mode register mn (TMRmn)



(b) Timer output register m (TOm)

	bit n	
TOm	TOmn 0	0: Outputs 0 from TOmn.

(c) Timer output enable register m (TOEm)

	bit n	
TOEm	TOEmn 0	0: Stops TOmn output performed by the counting operation.

(d) Timer output level register m (TOLm)

	bit n	
TOLm	TOLmn 0	0: Set “0” in master channel output mode (TOMmn=0).

(e) Timer output mode register m (TOMm)

	bit n	
TOMm	TOMmn 0	0: Sets master channel output mode.

Note: TMRm2: MASTERmn bit

Remark:

1. TMRm1, TMRm3: SPLITmn bits
2. TMRm0: Fixed to “0”.
3. m: unit number (m=0) n: channel number (n=0 ~ 3)

Table 5-29: Procedure for delay counter function

	Software operation	Hardware status
Timer4 initial settings		The input clock of timer unit m is in the stop-providing state. (Stop providing clock, cannot write to each register)
	Set the TM4mEN bit of the peripheral enable register 0(PER0) to "1". →	The input clock of timer unit m is in the providing state and the channels are in the stop state. (Start providing clock, can write to each register)
	Set the timer clock selection register m (TPSm). Determine the clock frequency of CKm0 ~ CKm3.	
Initial setting of channels	Set the corresponding bit of the Noise Filter Enable Register (NFEN1) to "0" (OFF) or "1" (ON). Set the timer mode register mn (TMRmn)(to determines the operating mode of the channel). Set the output delay time for the timer data register mn (TDRmn). Set the TOEmn bit to "0" and stop TOmn operation.	The channel is in a running stop state. (Provides clock, consumes some power)
Start Operation	Set the TSmn bit to "1". → Since the TSmn bit is a trigger bit, it automatically returns to "0".	The TEmn bit turns into '1' and enter into start trigger (detect Timn pin input active edge or set TSmn bit to '1') detection waiting state.
Restart operation	Start decreasing the count by detecting the next start trigger. • The active edge of the TImn pin input. • Set the TSmn bit to "1" by software. →	Load the value of the TDRmn register into the Timer Count Register mn (TCRmn).
In operation	The setting of the TDRmn register can be changed at will. The TCRmn register can be read at any time. The TSRmn register is not used.	The counter (TCRmn) performs decremental counting. If TCRmn counts to "0000H", INTTMmn is generated and TCRmn is "1" until the next start trigger is detected (detecting an active edge on the TImn pin input or setting TSmn to "1"). The count is stopped when "0000H" is detected.
Stop operation	Set the TTmn bit to "1". → The operation automatically returns to "0" because the TTmn bit is a trigger bit.	The TEmn bit becomes "0" and stops counting. The TCRmn register holds the count value and stops counting.
Timer4 stop	Set the TM4mEN bit of the PER0 register to "0". →	The input clock of timer unit m is in the stop-providing state. Initialize all circuits and the SFR for each channel.

Remark: m: unit number (m= 0, 1) n: channel number (n=0 ~ 3)

5.9 Multi-channel linkage operation function for general purpose timer unit

5.9.1 Operation as single trigger pulse output function

Using the 2 channels in pairs, a single trigger pulse with any delay pulse width can be generated from the input of the TImn pin. The delay and pulse width can be calculated using the following equation:

Delay = {TDRmn (master) set value +2} × counting clock period Pulse width = {TDRmp (slave) set value} × counting clock period

In single count mode, the master channel operates and counts the delay. By detecting a start trigger, the timer count register mn (TCRmn) of the master channel starts to operate and loads the value of timer data register mn (TDRmn). The TCRmn register counts decreasingly from the value of the loaded TDRmn register by counting the clock. If TCRmn becomes "0000H", INTTMmn is output and counting stops before the next start trigger is detected.

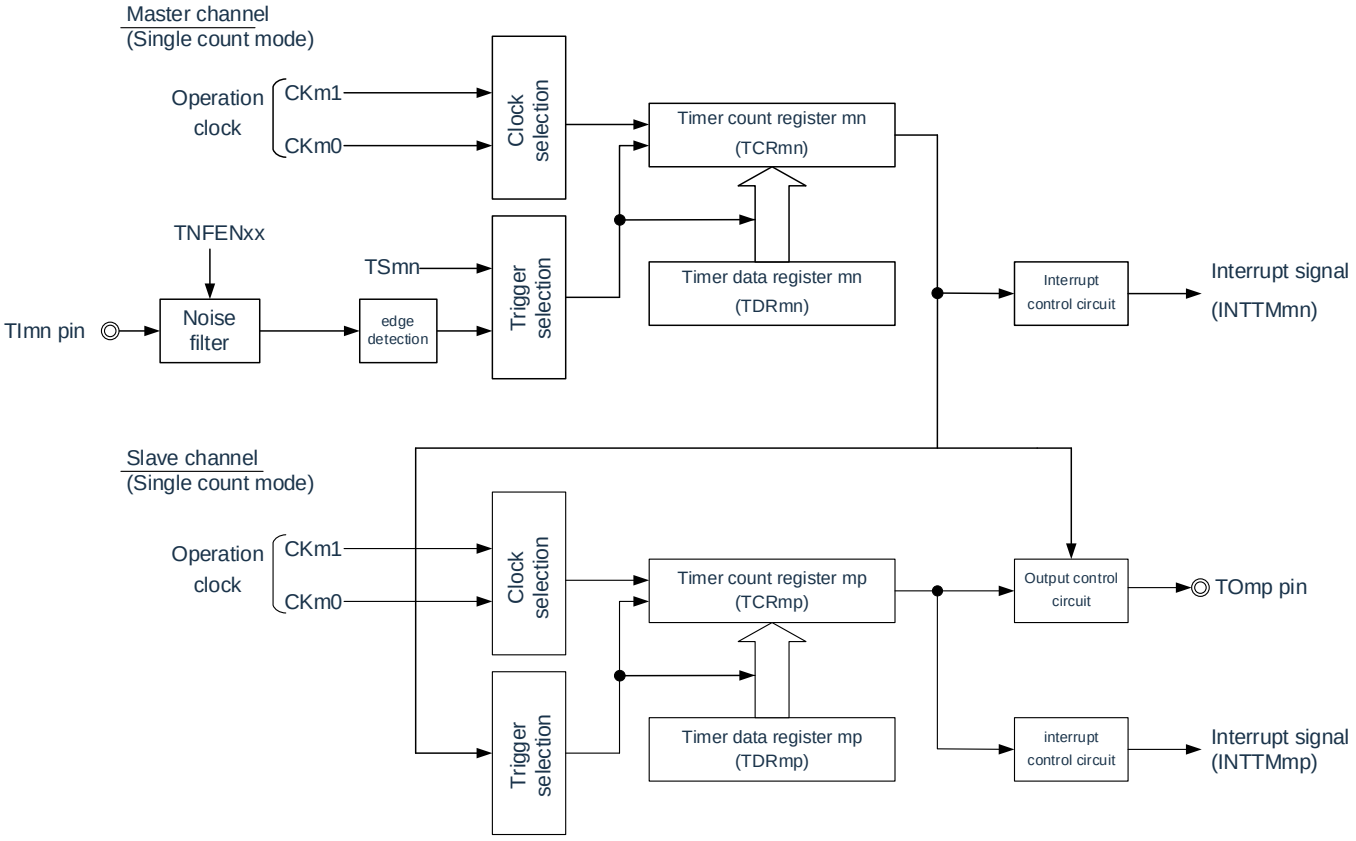
In single count mode, the slave channel operates and counts the pulse width. The INTTMmn of the master channel is used as the start trigger and the TCRmp register of the slave channel is started and loaded with the value of the TDRmp register. The TCRmp register counts decreasingly from the value of the loaded TDRmp register by counting the clock. If the count value becomes "0000H", INTTMmp is output and counting is stopped until the next start trigger (INTTMmn of the master channel) is detected. The output level of TOmp becomes valid after INTTMmn has been generated from the master channel and after 1 count clock, if TCRmp becomes "0000H", it becomes invalid.

The software operation (TSmn=1) can also be used as a start trigger to output a single trigger pulse without using the TImn pin input.

Notice: Because the TDRmn register of the master channel and the TDRmp register of the slave channel have different loading timings, if the TDRmn register and the TDRmp register are rewritten during counting, they may compete with the loading timings and output an abnormal waveform. The TDRmn register must be rewritten after generating INTTMmn and the TDRmp register must be rewritten after generating INTTMmp.

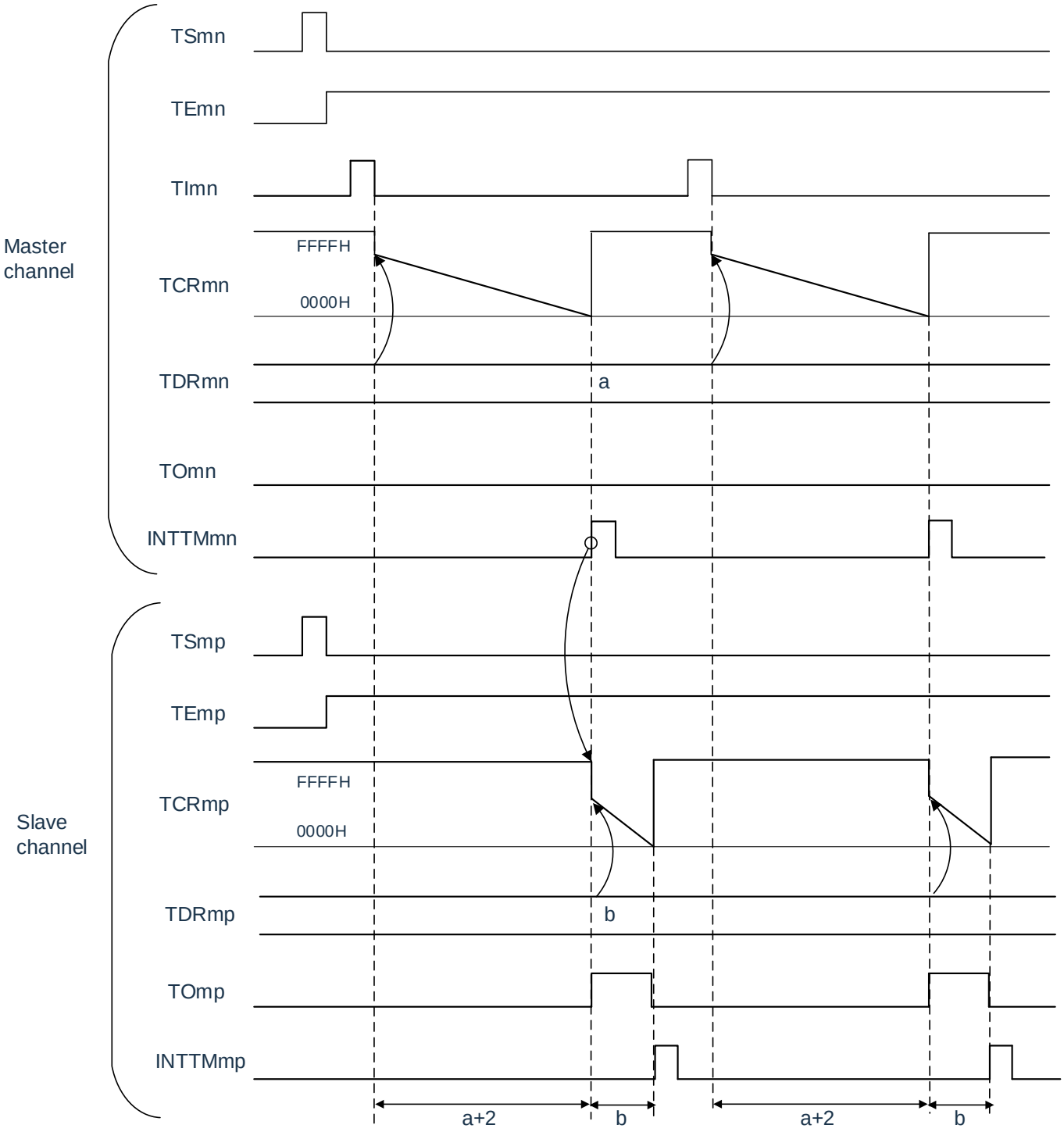
Remark: m: unit number (m=0, 1) n: master channel number (n=0, 2) p: slave channel number (n=0: p=1, 2, 3, n=2: p=3)

Figure 5-33: Block diagram of operation as single trigger pulse output function



Remark: m: unit number (m= 0, 1) n: master channel number (n=0, 2) p: slave channel number (n=0: p=1, 2, 3, n=2: p=3)

Figure 5-34: Example of basic timing operating as a single trigger pulse output function

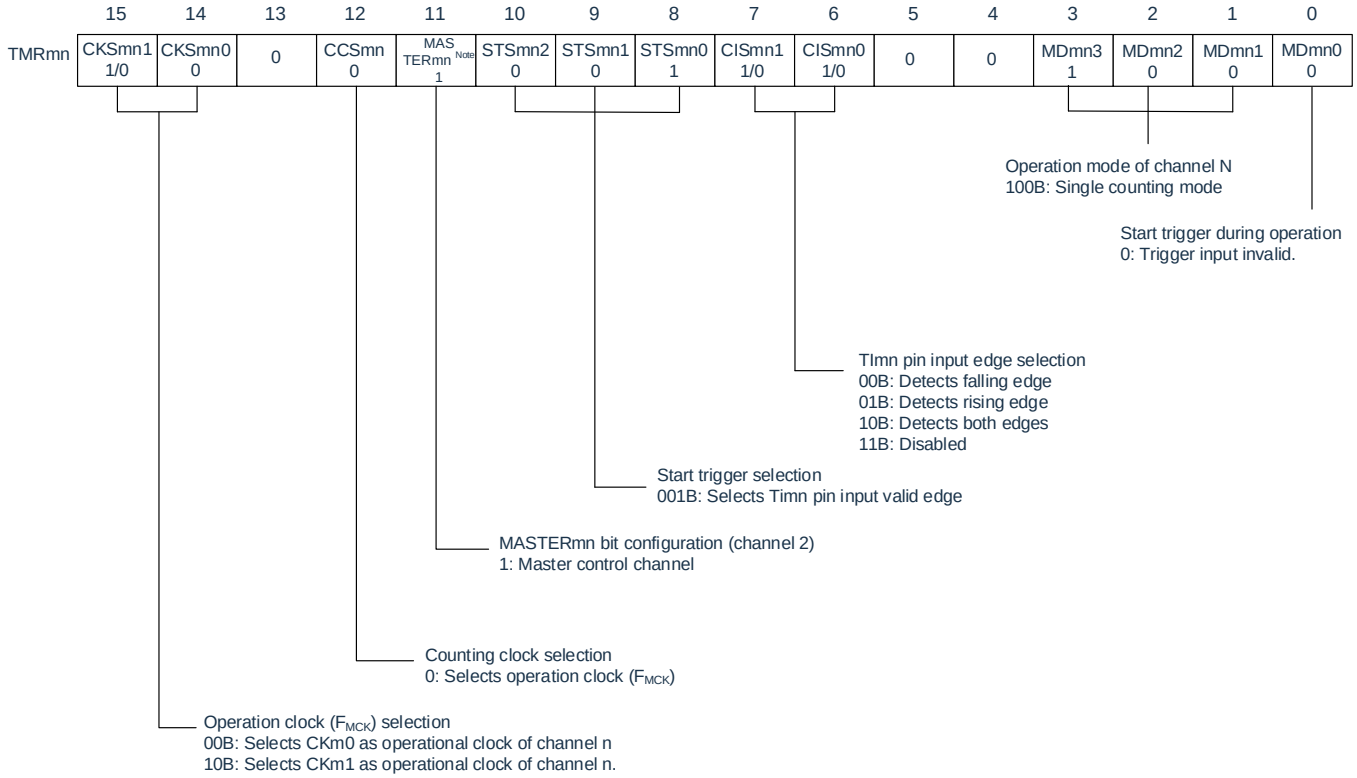


Remark:

1. m: unit number (m= 0, 1) n: master channel number (n=0, 2) p: slave channel number (n=0: p=1, 2, 3, n=2: p=3)
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: Input signals of TImn pin and TImp pin
   TCRmn, TCRmp: Timer count registers mn, mp (TCRmn, TCRmp)
   TDRmn, TDRmp: Timer data registers mn, mp (TDRmn, TDRmp)
   TOmn, TOmp: Output signals of TOmn pin and TOmp pin

Figure 5-35: Example of register contents setting for single trigger pulse output function (master channel)

(a) Timer mode register mn (TMRmn)



(b) Timer output register m (TOm)

	bit n	
TOm	TOmn 0	0: Outputs 0 from TOmn.

(c) Timer output enable register m (TOEm)

	bit n	
TOEm	TOEmn 0	0: Stops TOmn output performed by the counting operation.

(d) Timer output level register m (TOLm)

	bit n	
TOLm	TOLmn 0	0: Set “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.

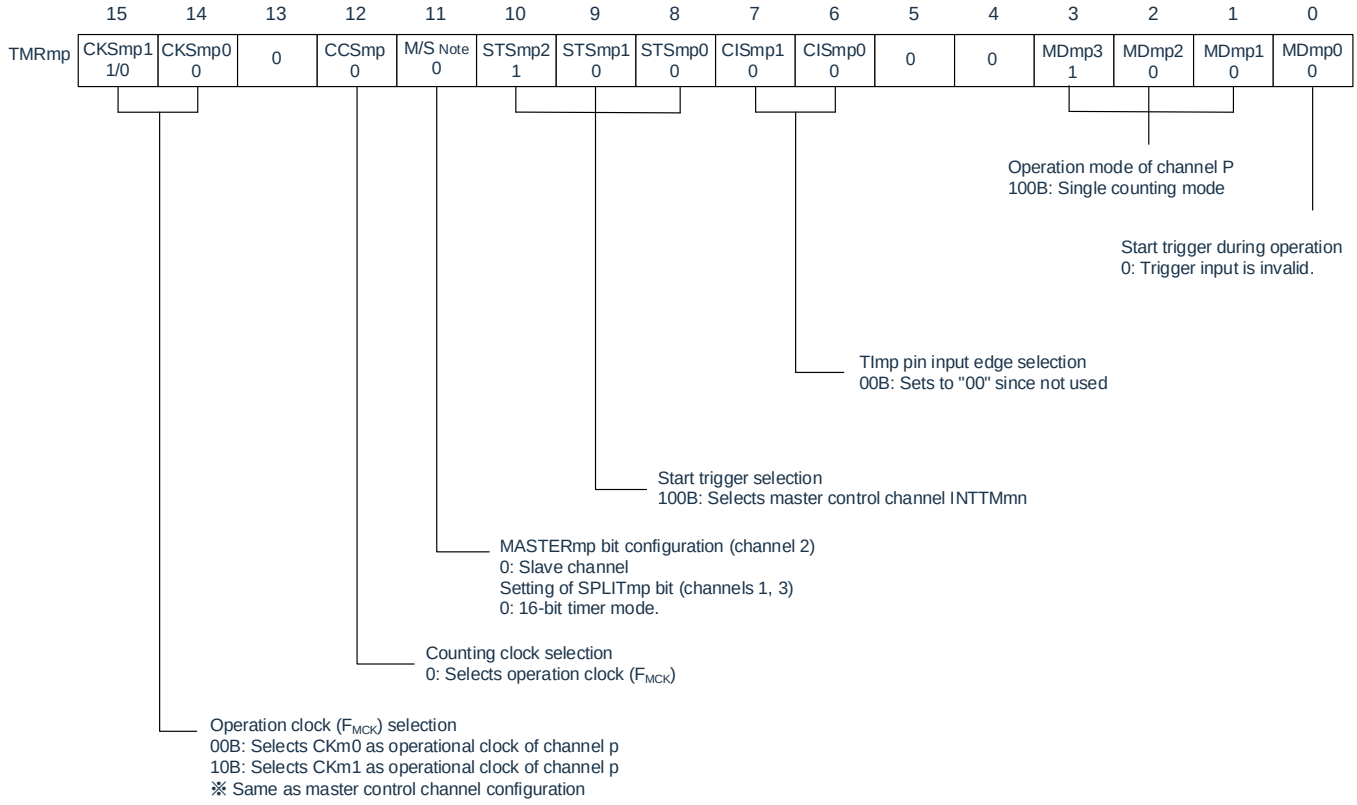
Note: TMRm2: MASTERmn=1

TMRm0: Fixed to “0”.

Remark: m: unit number (m= 0, 1) n: master channel number (n=0, 2)

Figure 5-36: Example of register contents setting for single trigger pulse output function (slave channel)

(a) Timer mode register mp (TMRmp)



(b) Timer output register m (TOm)

	bit p	
TOm	TOmp 1/0	0: Outputs "0" by TOmp. 1: Outputs "1" by TOmp.

(c) Timer output enable register m (TOEm)

	bit p	
TOEm	TOEmp 1/0	0: Stops TOmp output performed by the counting operation. 1: Enables TOmp output performed by the counting operation.

(d) Timer output level register m (TOLm)

	bit p	
TOLm	TOLmp 1/0	0: Positive logic output (active high level) 1: Negative logic output (active low level)

(e) Timer output mode register m (TOMm)

	bit p	
TOMm	TOMmp 1	1: Sets slave channel output mode.

Note: TMRm2: MASTERmp bit

TMRm1, TMRm3: SPLITmp bits

Remark: m: unit number (m= 0, 1) n: master channel number (n=0, 2) p: slave channel number (n=0: p=1, 2, 3, n=2: p=3)

Table 5-30: Procedure for single trigger pulse output function (1/2)

	Software operation	Hardware status
Timer4 initial settings		The input clock of timer unit m is in the stop-providing state. (Stop providing clock, cannot write to each register)
	Set the TM4mEN bit of the peripheral enable register 0(PER0) to "1". →	The input clock of timer unit m is in the providing state and the channels are in the stop state. (Start providing clock, can write to each register)
	Set the timer clock selection register m (TPSm). Determine the clock frequency of CKm0 ~ CKm3.	
Initial setting of channels	Set the corresponding bit of the Noise Filter Enable Register (NFEN1) to "1". Set the timer mode registers mn and mp (TMRmn, TMRmp) for the 2 channels used (to determine the operation mode of the channel). Set the output delay time for the timer data register mn (TDRmn) of the master channel, and set the pulse width for the TDRmp register of the slave channel.	The channel is in the stop state. (Provides clock, and consumes some power)
	Slave channel setting Set TOMmp bit of the timer output mode register m (TOMm) to"1" (slave channel output mode). Set the TOLmp bit. Set the TOmp bit to determine the initial level of the TOmp output. Set the TOEmp bit to "1" and enable TOmp output. Set the Port Register and Port Mode Register to "0".	The TOmp pin is in Hi-Z output state. When the port mode register is in output mode and the port register is "0", the TOmp initial set level is output. The TOmp remains unchanged because the channel is in the stop state. The TOmp pin outputs the level set by the TOmp.

Table 5-30: Procedure for single trigger pulse output function (2/2)

	Software operation	Hardware status
Start operation	Set the TOEmp bit (slave) to "1" (restart operation only). Set the TSmn (master) and TSmp (slave) bits of the Timer Channel Start Register m (TSm) to "1" at the same time. Since the TSmn bit and the TSmp bit are trigger bits, they automatically return to "0".	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. Counter stops operating.
	Count operation of the master channel is started by start trigger detection of the master channel • Detects the TImn pin input valid edge •Sets the TSmn bit of the master channel to 1 by software Note.	Master channel starts counting.
In operation	Set values of only the CISmn1 and CISmn0 bits of the TMRmn register can be changed. Set values of the TMRmp, TDRmn, TDRmp registers, TOMmn, TOMmp, TOLmn, and TOLmp bits cannot be changed. The TCRmn and TCRmp registers can always be read. The TSRmn and TSRmp registers are not used. Set values of the TOm and TOEm registers by slave channel can be changed.	Master channel loads the value of the TDRmn register to timer 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 stops counting until the next TImn pin input. The slave channel, triggered by INTTMmn of the master channel, loads the value of the TDRmp register to the TCRmp register, and the counter starts counting down. The output level of TOmp becomes active one count clock after generation of INTTMmn from the master channel. It becomes inactive when TCRmp = 0000H, and the counting operation is stopped. After that, the above operation is repeated.
Stop operation	The TTmn (master) and TTmp (slave) bits are set to 1 at the same time. The TTmn and TTmp bits automatically return to 0 because they are trigger bits.	TEmn, TEmp = 0, and count operation stops. The TCRmn and TCRmp registers hold count value and stop. The TOmp output is not initialized 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.
Timer4 stop	To hold the TOmp pin output level Clears the TOmp bit to 0 after the value to be held is set to the port register. When holding the TOmp pin output level is not necessary: Setting not required.	The TOmp pin output level is held by port function.
	The TM4mEN bit of the PER0 register is cleared to 0.	The input clock of timer unit m is in the stop-providing state. Initialize all circuits and the SFR of each channel.

Restart operation

Note: The TSmn bit of the slave channel cannot be set to "1".

Remark: m: unit number (m= 0, 1) n: master channel number (n=0)

p: slave channel number q: slave channel number n < p < q ⩽ 3 (p and q are integers greater than n)

5.9.2 Operation as PWM function

By using the 2 channels in pairs, pulses of any period and duty cycle can be generated. The period and duty cycle of the output pulses can be calculated using the following equations:

Pulse period = {TDRmn (master) set value +1} × counting clock period
Duty cycle [%] = {TDRmp (slave) set value} / {TDRmn (master) set value +1} ×100
0% output: TDRmp (slave) set value = 0000H
100% output: TDRmp (slave) set value≥ {TDRmn (master) set value +1}

The master channel is used as the interval timer mode. If the channel start trigger bit (TSmn) of the timer channel start register m (TSm) is set to "1", an interrupt (INTTMmn) is output, and then the set value of the timer data register mn (TDRmn) is loaded into the timer count register mn (TCRmn), and the count is decremented by the count clock. When the count reaches "0000H", the value of the TDRmn register is loaded into the TCRmn register again after the INTTMmn is output, and the count is decremented. Thereafter, this operation is repeated before setting the channel stop trigger bit (TTmn) of the timer channel stop register m (TTm) to "1".

When used as PWM function, the master channel decrements the count and the period until "0000H" is counted as the PWM output (TOmp) period. The slave channel is used in single count mode. The value of TDRmp register is loaded into TCRmp register with INTTMmn of the master channel as the start trigger, and the count is decremented until "0000H". When the count reaches "0000H", INTTMmp is output and the next start trigger (INTTMmn of the master channel) is waited.

When used as PWM function, the slave channel decrements the count and the duty cycle of the PWM output (TOmp) for the period until "0000H" is counted.

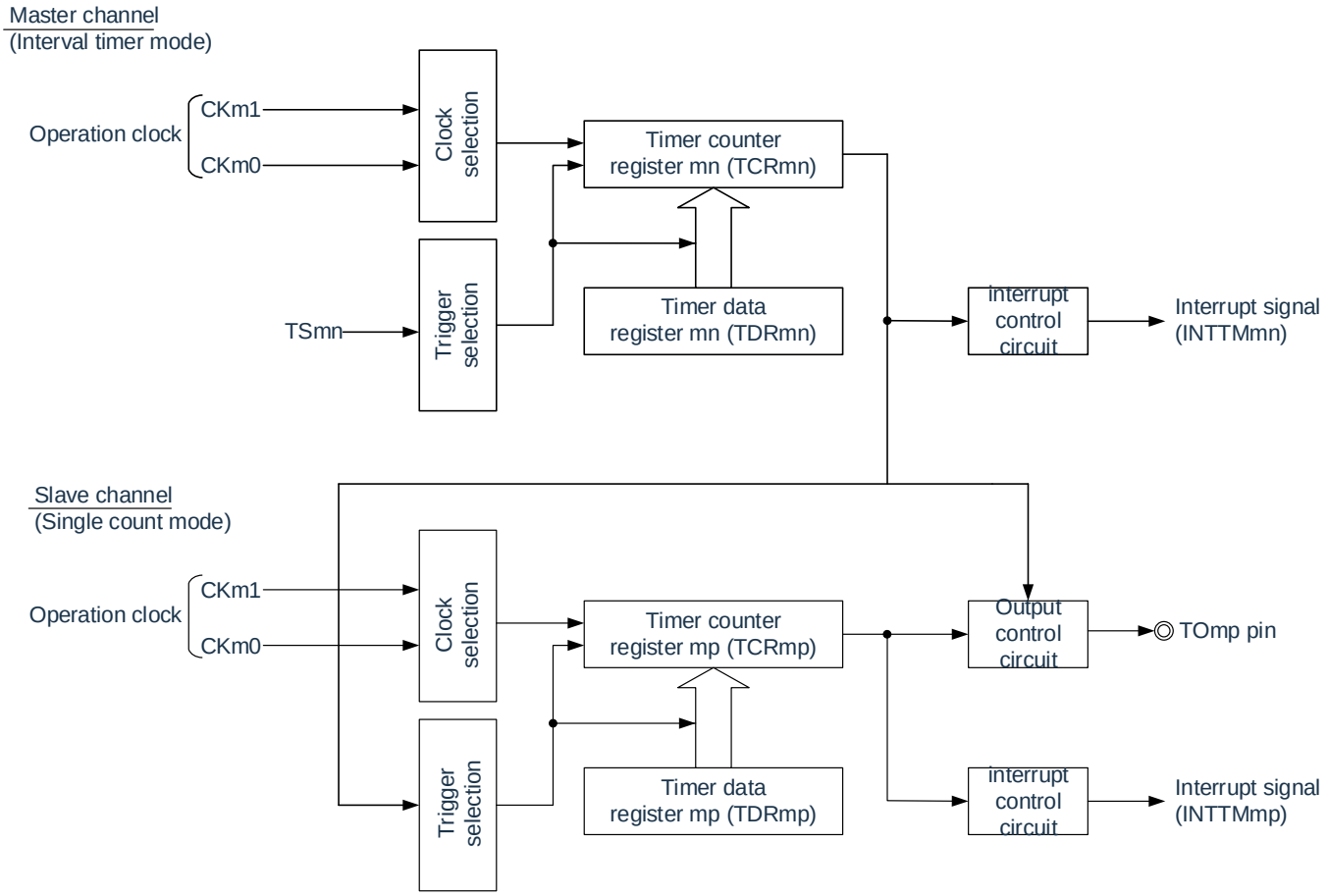
After INTTMmn is generated from the master channel and 1 clock has elapsed, the PWM output (TOmp) becomes active and it becomes invalid when the value of TCRmp register of the slave channel is "0000H".

Notice:

1. When the set value of TDRmp (slave) > {Set value of TDRmn (master) +1}, the duty cycle exceeds 100% but is 100% output.
2. 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.

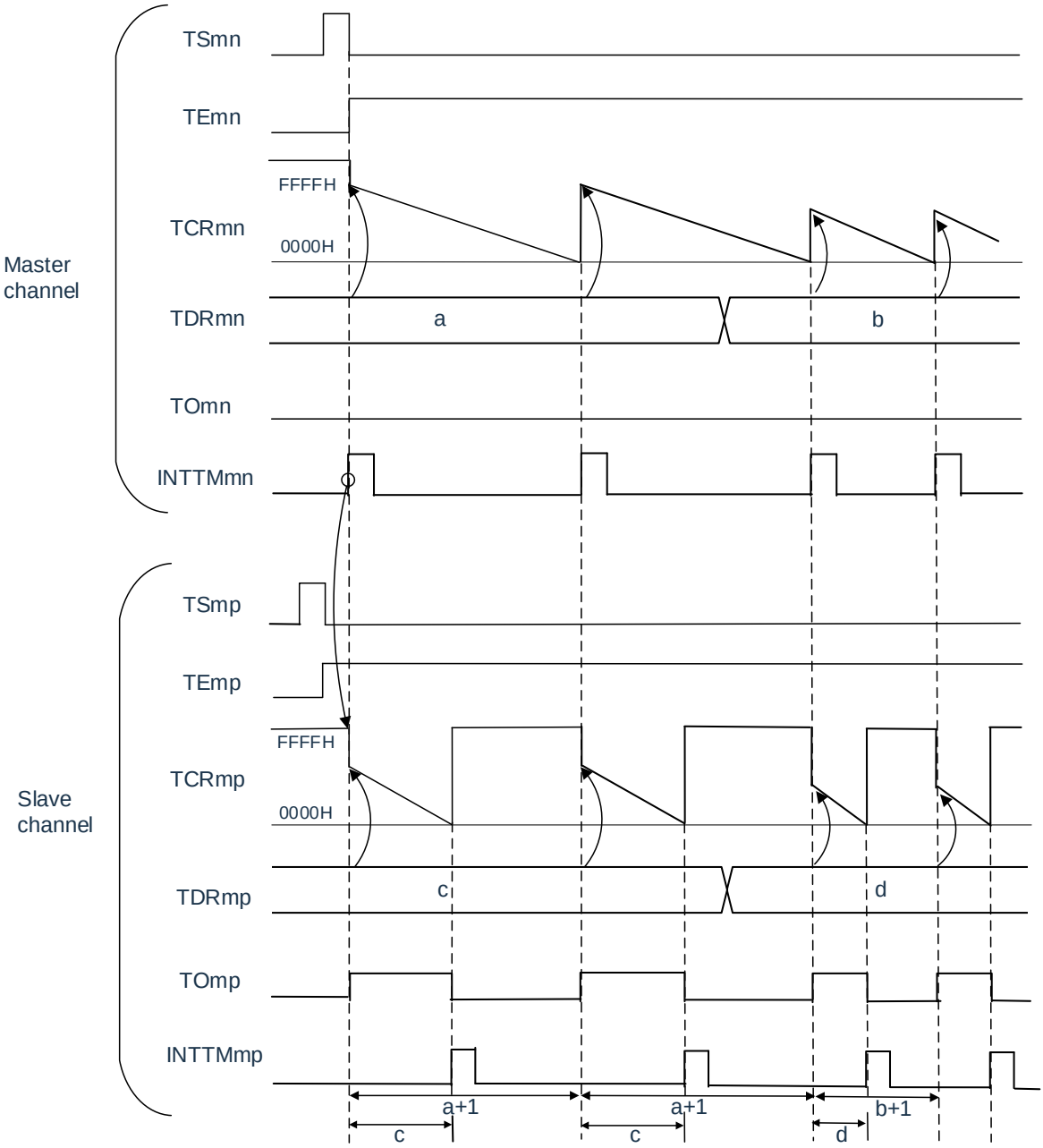
Remark: m: unit number (m= 0, 1) n: master channel number (n=0, 2) p: slave channel number (n=0: p=1, 2, 3, n=2: p=3)

Figure 5-37: Block diagram of operation as PWM function



Remark: m: unit number (m= 0, 1) n: master channel number (n=0, 2) p: slave channel number (n=0: p=1, 2, 3, n=2: p=3)

Figure 5-38: Example of basic timing operating as PWM function

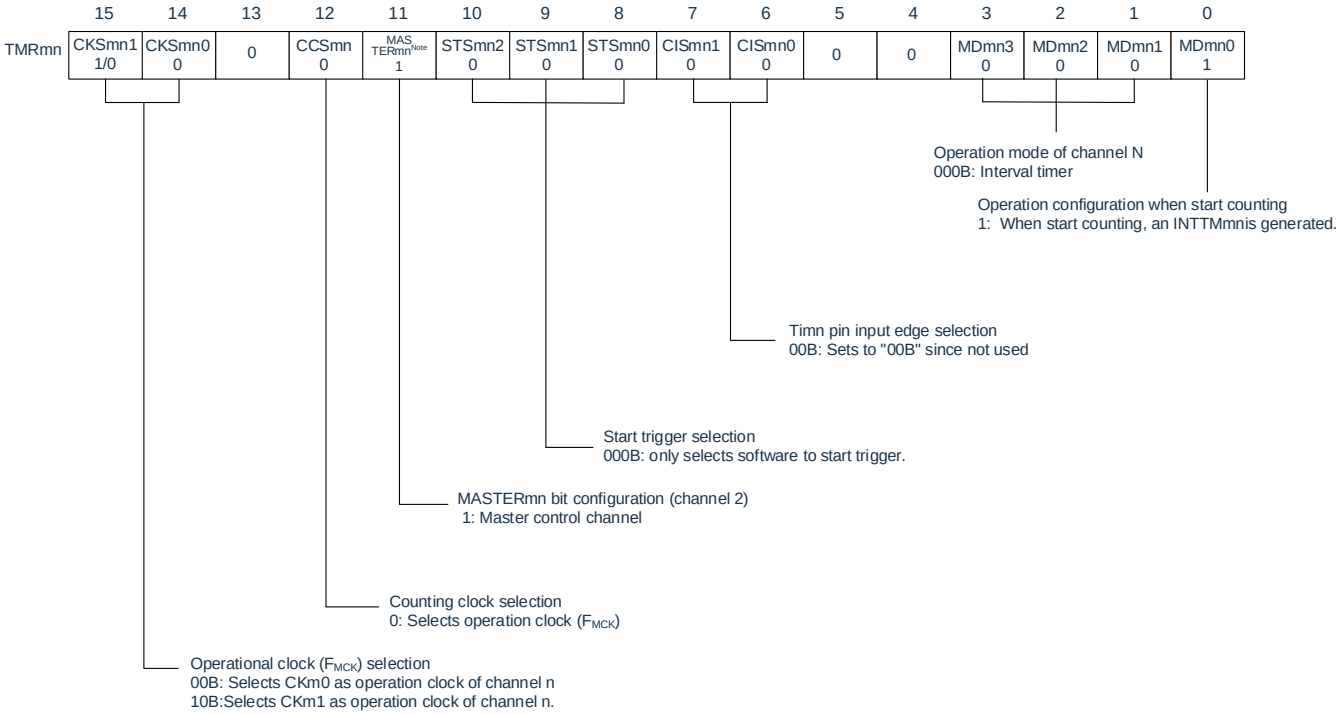


Remark:

1. m: unit number (m= 0, 1) n: master channel number (n=0, 2) p: slave channel number (n=0: p=1, 2, 3, n=2: p=3)
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: Output signals of TOmn pin and TOmp pin

Figure 5-39: Example of register contents setting for PWM function (master channel)

(a) Timer mode register mn (TMRmn)



(b) Timer output register m (TOm)

	bit n	
TOm	TOmn 0	0: Outputs 0 from TOmn.

(c) Timer output enable register m (TOEm)

	bit n	
TOEm	TOEmn 0	0: Stops TOmn output performed by the counting operation.

(d) Timer output level register m (TOLm)

	bit n	
TOLm	TOLmn 0	0: Set “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.

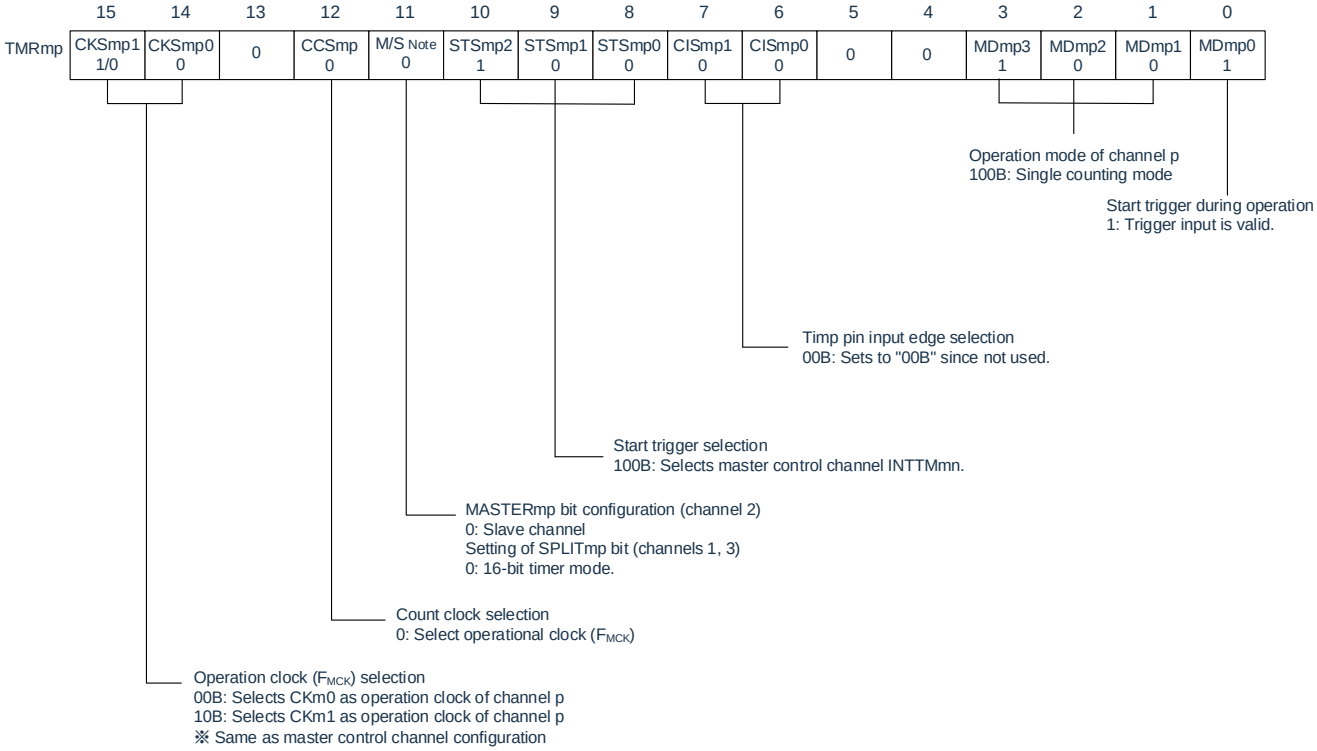
Note: TMRm2: MASTERmn=1

TMRm0: Fixed to “0”.

Remark: m: unit number (m=0, 1) n: master channel number (n=0, 2)

Figure 5-40: Example of register contents setting for PWM function (slave channel)

(a) Timer mode register mp (TMRmp)



(b) Timer output register m (TOm)

	bit p	
TOm	TOmp 1/0	0: Outputs 0 from TOmp. 1: Outputs 1 from TOmp.

(c) Timer output enable register m (TOEm)

	bit p	
TOEm	TOEmp 1/0	0: Stops TOmp output performed by the counting operation. 1: Enables TOmp output performed by the counting operation.

(d) Timer output level register m (TOLm)

	bit p	
TOLm	TOLmp 1/0	0: Positive logic output (active-high) 1: Negative logic output (active-low)

(e) Timer output mode register m (TOMm)

	bit p	
TOMm	TOMmp 1	1: Sets the slave channel output mode.

Note 1: TMRm2: MASTERmp bit

Remark:

1. TMRm1, TMRm3: SPLITmp bit
2. m: unit number (m= 0, 1) n: master channel number (n=0, 2) p: slave channel number (n=0: p=1, 2, 3, n=2: p=3)

Table 5-31: Procedure for the PWM function(1/2)

	Software operation	Hardware status
Timer4 initial settings		The input clock of timer unit m is in the stop-providing state.
	Set the TM4mEN bit of the peripheral enable register 0(PER0) to "1". →	The input clock of timer unit m is in the providing state and the channels are in the stop state. (Start providing clock, can write to each register)
	Set the timer clock selection register m (TPSm). Determine the clock frequency of CKm0 ~ CKm3.	
Initial setting of channels	Set the timer mode registers mn and mp (TMRmn, TMRmp) for the 2 channels used (to determine the operation mode of the channel). Set the interval (period) value for the timer data register mn (TDRmn) for the master channel and the duty cycle value for the TDRmp register for the slave channel.	The channel is in the stop state. (Provides clock, and consumes some power)
	Slave channel setting Set TOMmp bit of the timer output mode register m (TOMm)to"1" (slave channel output mode). Set the TOLmp bit. Set the TOmp bit to determine the initial level of the TOmp output. Set the TOEmp bit to "1" and enable TOmp output. Set the Port Register and Port Mode Register to "0".	The TOmp pin is in Hi-Z output state. When the port mode register is in output mode and the port register is "0", the initially set level of TOmp is output. The TOmp remains unchanged because the channel is in the stop state. The TOmp pin outputs the level set by the TOmp.

Table 5-31: Procedure for the PWM function (2/2)

		Software operation	Hardware status
Restart operation	Start operation	Set the TOEmp bit to “1” (only limited to restart operation). Set both the TSmn bit (master) and TSmp bit (slave) of the timer channel start register m (TSm) to "1". → The operation automatically returns to "0" because the TSmn and TSmp bits are trigger bits.	The TEmn and TEmp bits become "1". The master channel starts counting and generates INTTMmn. With this as a trigger, the slave channel also starts counting.
	n operation	The setting values of the TMRmn and TMRmp registers and the TOMmn bit, TOMmp bit, TOLmn bit, and TOLmp bit cannot be changed. Able to change the setting value of the TDRmn register and the TDRmp register after the master channel has generated INTTMmn. The TCRmn and TCRmp registers can be read at any time. The TSRmn and TSRmp registers are not used.	The master channel loads the value of the TDRmn register into the timer count register mn (TCRmn) and perform decremental counting. If TCRmn counts till "0000H", then generating INTTMmn. At the same time, load the TDRmn register value into the TCRmn register and restart decremental counting. The slave channel use INTTMmn of master channel as a trigger, load the TDRmp register value into the TCRmp register and counter start decremental counting. After INTTMmn is output from the master channel and one count clock has elapsed, the output level of TOmp is set to an active level. Then, if TCRmp counts to "0000H", it stops counting after setting the output level of TOmp to an invalid level. Thereafter, repeat this operation.
	Stop operation	Set the TTmn bit (master) and TTmp bit (slave) to "1" at the same time. → The operation automatically returns to "0" because the TTmn and TTmp bits are trigger bits.	TEmn, TEmp = 0, and count operation stops. The TCRmn and TCRmp registers hold count value and stop. The TOmp output is not initialized but holds current
		Set the TOEmp bit of slave channel to "0" and set the value for the TOmp bit. →	The TOmp pin outputs the TOmp set level.
	Timer4 stop	To maintain the output level of the TOmp pin: Set TOmp bit to "0" after setting the value to be held for the port register. → When holding the TOmp pin output level is not necessary: No need to set.	The TOmp pin output level is held by port function.
		Set the TM4mEN bit of the PER0 register to "0". →	The input clock of timer unit m is in the stop-providing state. Initialize all circuits and the SFR for each channel. (TOmn bit becomes "0" and TOmp pin becomes port function)

Remark: m: unit number (m= 0, 1) n: master channel number (n=0)

p: slave channel number q: slave channel number n < p < q ≤ 3 (p and q are integers greater than n)

5.9.3 Operation as multiple PWM output function

This is a function that extends the PWM function and uses multiple slave channels for multiple PWM outputs with different duty cycles. For example, when using 2 slave channels in pairs, the period and duty cycle of the output pulse can be calculated by using the following equation:

Pulse period= {TDRmn(master) set value +1} × count clock period Duty cycle1[%] = {TDRmp (slave 1) set value} / {TDRmn (master) set value+1} ×100 Duty cycle2[%] = {TDRmq (slave 2) set value} / {TDRmn(master) set value +1} ×100

In interval timer mode, the timer count register mn (TCRmn) of the master channel operates and counts the period. In single count mode, the TCRmp register of slave channel 1 operates and counts the duty cycle and outputs the 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”, the 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 valid after INTTMmn has been generated from the master channel and after 1 count clock, if TCRmp becomes "0000H", it becomes invalid.

In the same way as the TCRmp register of the slave channel 1, the TCRmq register of the slave channel 2 operates in single count mode, counts the duty cycle, 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 the 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 3 types of PWM signals can be output at the same time.

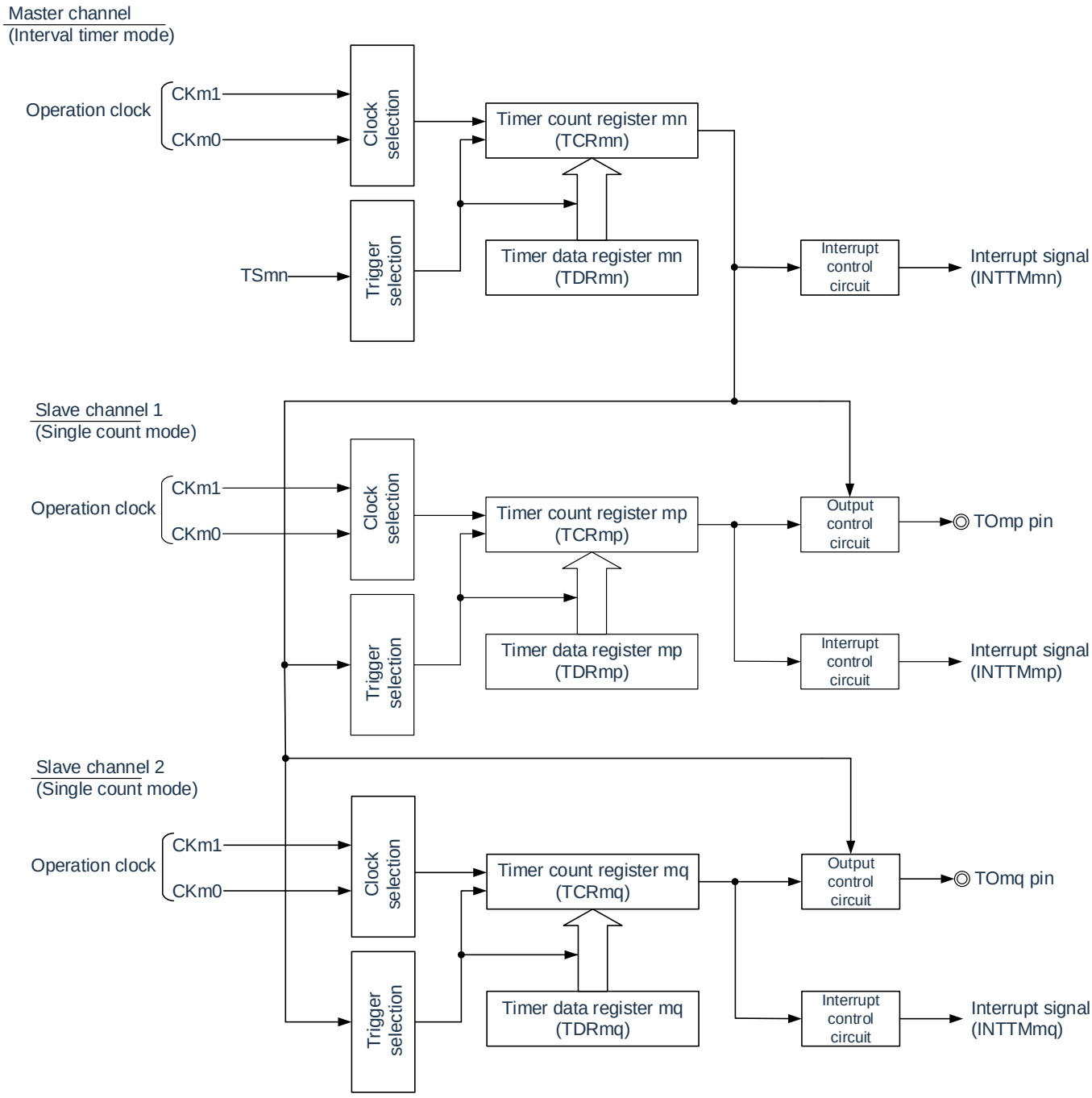
Notice:

1. When the set value of TDRmp (slave 1) > {the set value of TDRmn (master) + 1} or {the set value of TDRmq (slave 2)} > {the set value of TDRmn (master) + 1}, the duty cycle exceeds 100%, but is 100% output.
2. To rewrite the timer data register mn (TDRmn) of the master channel and the TDRmp register of the slave channel 1 at the same time, at least 2 write accesses are required. Because the values of TDRmn register and TDRmp register are loaded into the TCRmn register and TCRmp register when the master channel generates INTTMmn, the TOmp pin cannot output the expected waveform if rewriting is performed before and after the master channel generates INTTMmn respectively. Therefore, to rewrite both the master TDRmn register and the slave TDRmp register, these two registers must be rewritten immediately after the master channel generates INTTMmn (the same applies to the TDRmq register of slave channel 2).

Remark: m: unit number (m= 0, 1) n: master channel number (n=0)

p: slave channel number q: slave channel number n＜p＜q≤3 (p and q are integers greater than n)

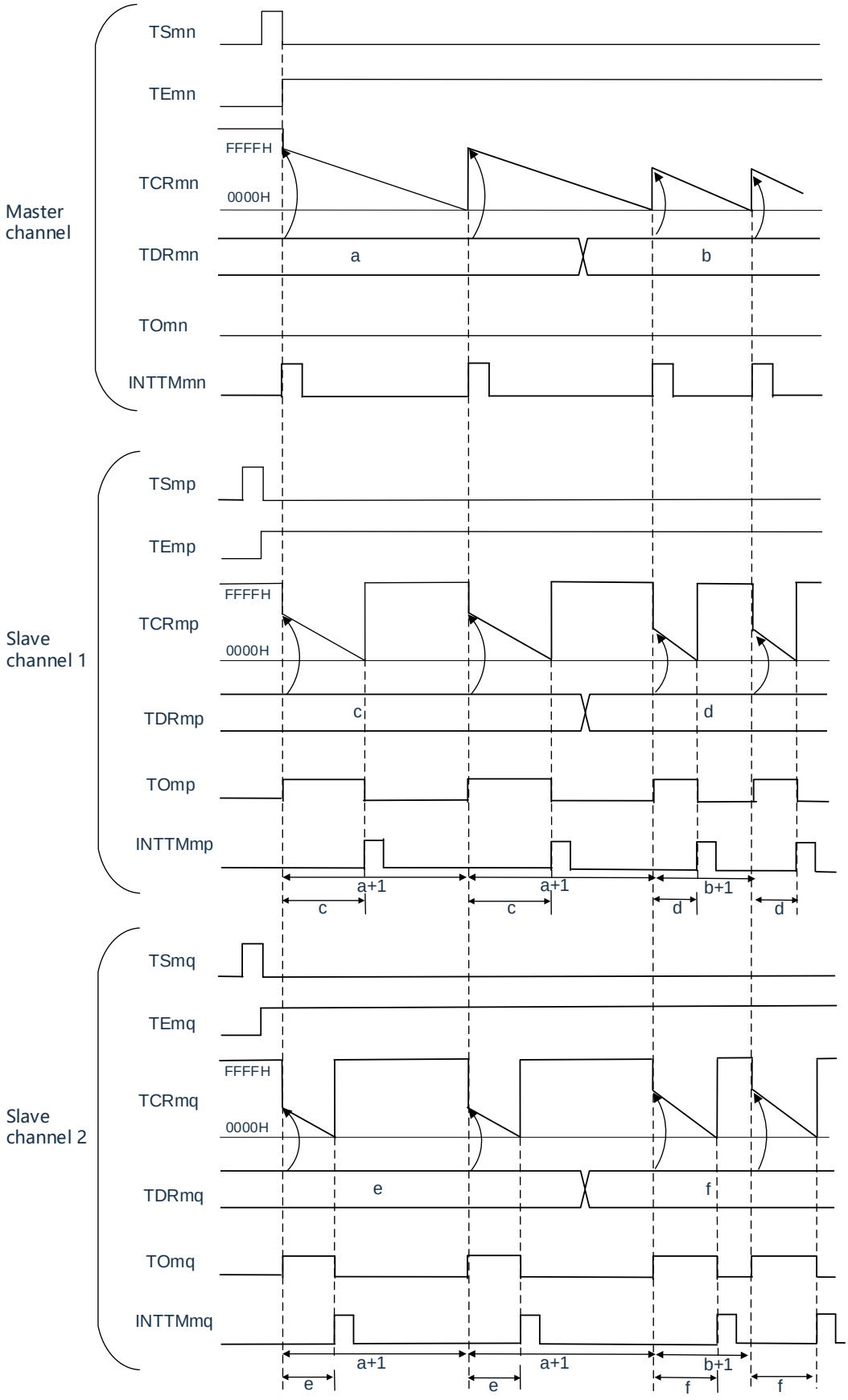
Figure 5-41: Block diagram of operation as multiple PWM output function (output two types of PWMs)



Remark: m: unit number (m= 0, 1) n: master channel number (n=0)

p: Slave channel number q: slave channel number $n<p<q\leqslant 3$ (p and q are integers greater than n)

Figure 5-42: Example of basic timing operating as multiple PWM output function (output two types of PWMs)

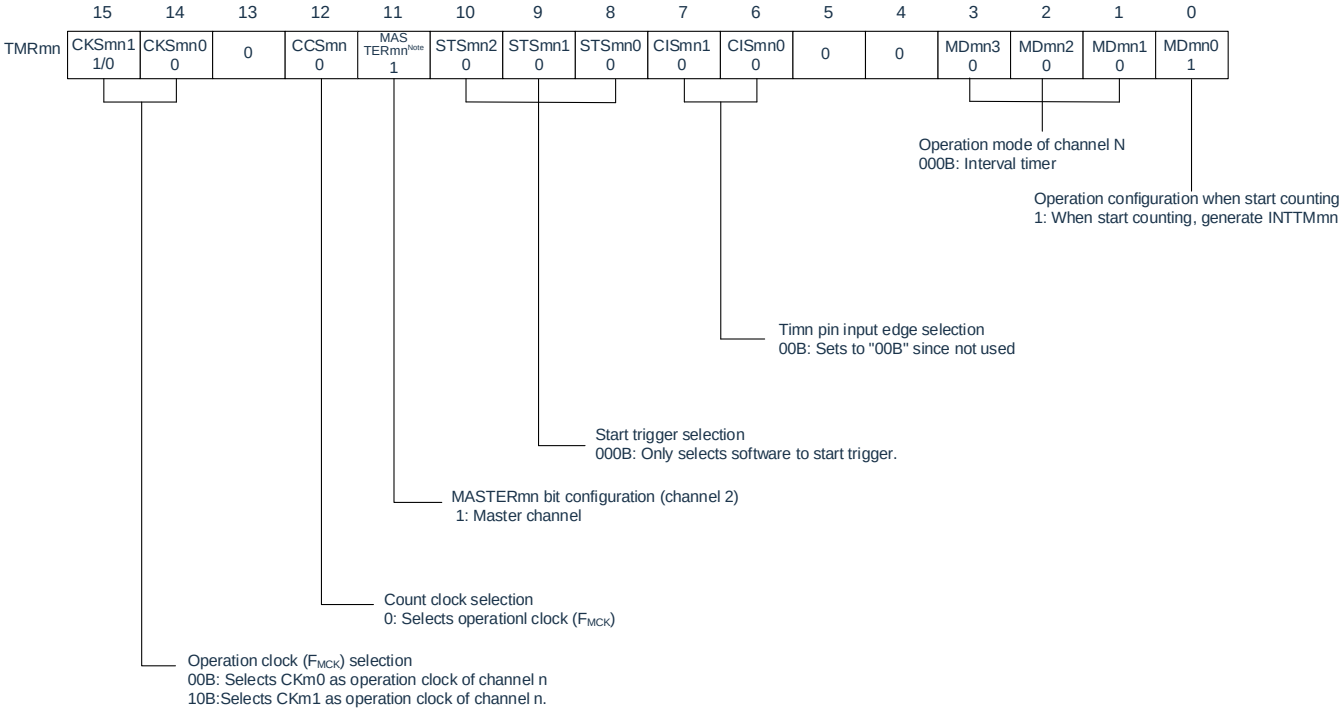


Remark:

1. m: unit number (m= 0, 1) n: master channel number (n=0)
   p: slave channel number q: slave channel number n < p < q ⩽ 3 (p and q are integers greater than n)
2. TSmn, TSmp, TSmq: Bit n, p of timer channel start register m (TSm), q
   TEmn, TEmp, TEmq: Bit n, p of timer channel enable status register m (TEm), q
   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, TOmq pin output signals

Figure 5-43: Example of register contents setting for multiple PWM output function (master channel)

(a) Timer mode register mn (TMRmn)



(b) Timer output register m (TOm)

	bit n	
TOm	TOmn 0	0: Outputs 0 from TOmn.

(c) Timer output enable register m (TOEm)

	bit n	
TOEm	TOEmn 0	0: Stops TOmn output performed by the counting operation.

(d) Timer output level register m (TOLm)

	bit n	
TOLm	TOLmn 0	0: Sets "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.

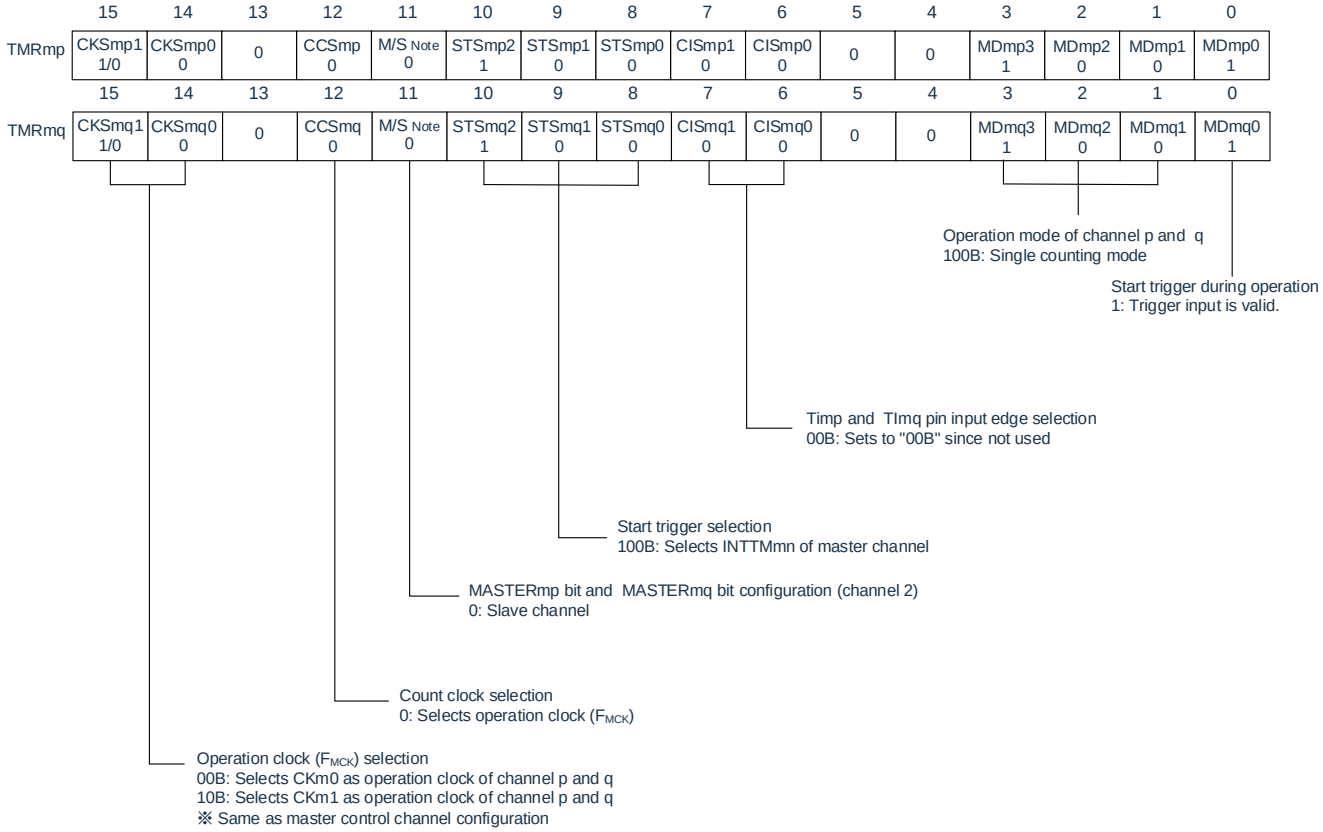
Note: TMRm2: MASTERmn=1

TMRm0: Fixed to “0”.

Remark: m: unit number (m= 0, 1) n: master channel number (n=0)

Figure 5-44: Example of register contents setting for multiple PWM output function (slave channel) (output two types of PWMs)

(a) Timer mode registers mp, mq (TMRmp, TMRmq)



(b) Timer output register m (TOm)

	bit q	bit p	
TOm	TOmq 1/0	TOmp 1/0	0: Outputs 0 from TOmp or TOmq. 1: Outputs 0 from TOmp or TOmq.

(c) Timer output enable register m (TOEm)

	bit q	bit p	
TOEm	TOEmq 1/0	TOEmp 1/0	0: Stops the TOmp or TOmq output operation by counting operation. 1: Enables the TOmp or TOmq output operation by counting operation.

(d) Timer output level register m (TOLm)

	bit q	bit p	
TOLm	TOLmq 1/0	TOLmp 1/0	0: Positive logic output (active-high) 1: Negative logic output (active-low)

(e) Timer output mode register m (TOMm)

	bit q	bit p	
TOMm	TOMmq 1	TOMmp 1	1: Sets the slave channel output mode.

Note: TMRm2: MASTERmp bit

Remark:

1. TMRm1, TMRm3: SPLITmp bits
2. m: unit number (m= 0, 1) n: master channel number (n=0, 2) p: slave channel number (n=0: p=1, 2, 3, n=2: p=3)

Table 5-32: Procedure for the multiple PWM output function (output two types of PWMs) (1/2)

	Software operation	Hardware status
Timer4 initial settings		The input clock of timer unit m is in the stop-providing state. (Stop providing clock, cannot write to each register)
	Set the TM4mEN bit of the peripheral enable register 0(PER0) to "1". →	The input clock of timer unit m is in the providing state and the channels are in the stop state. (Start providing clock, can write to each register)
	Set the timer clock selection register m (TPSm). Determine the clock frequency of CKm0 and CKm1.	
Initial setting of channels	Set the timer mode registers mn, mp, (TMRmn, TMRmp,) for each channel used (to determine the channel operation mode). Set the interval (period) value for the master channel's timer data register mn (TDRmn), and set the duty cycle	The channel is in the stop state. (Provides clock, and consumes some power)
	Slave channel setting Set TOMmp and TOMmq bits of the timer output mode register m (TOMm) to"1" (slave channel output mode). Set the TOLmp and TOLmq bits to "0". Set the TOmp and TOmq bits and determine the initial output level of the TOmp and TOmq bits. → Set the TOEmp amd TOEmq bits to "1" and enable TOmp and TOmq output. → Set the Port Register and Port Mode Register to "0". →	The TOmp pin is in Hi-Z output state. When the port mode register is in output mode and the port register is "0", the TOmp and TOmq initial set levels are output. The TOmp and TOmq remains unchanged because the channel is in the stop state. The TOmp pin and TOmq pin output the levels set by the TOmp and TOmq.

Table 5-32: Procedure for the multiple PWM output function (output two types of PWMs) (2/2)

		Software operation	Hardware status
Restart operation	Start operation	(Sets the TOEmp and TOEmq (slave) bits to 1 only when resuming operation.) The TSmn bit (master), and TSmp and TSmq (slave) bits of timer channel start register m (TSm) are set to 1 at the same time. The TSmn, TSmp, and TSmq bits automatically return to 0 because they are trigger bits.	TEmn = 1, TEmp, TEmq = 1 When the master channel starts counting, INTTMmn is generated. Triggered by this interrupt, the slave channel also starts counting.
	In operation	Set values of the TMRmn, TMRmp, TMRmq registers, TOMmn, TOMmp, TOMmq, TOLmn, TOLmp, and TOLmq bits cannot be changed. Set values of the TDRmn, TDRmp, and TDRmq registers can be changed after INTTMmn of the master channel is generated. The TCRmn, TCRmp, and TCRmq registers can always be read. The TSRmn, TSRmp, and TSR0q registers are not used.	The counter of the master channel loads the TDRmn register value to timer 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. 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. 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. After that, the above operation is repeated.
	Stop operation	The TTmn bit (master), TTmp, and TTmq (slave) bits are set to 1 at the same time. The TTmn, TTmp, and TTmq bits automatically return to 0 because they are trigger bits.	TEmn, TEmp, TEmq = 0, and count operation stops. The TCRmn, TCRmp, and TCRmq registers hold count value and stop. The TOmp and TOmq 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.
	Timer4 stop	To hold the TOmp and TOmq pin output levels Clears the TOmp and TOmq bits to 0 after the value to be held is set to the port register. When holding the TOmp and TOmq pin output levels are not necessary: Setting not required	The TOmp and TOmq pin output levels are held by port function.
		The TM4mEN bit of the PER0 register is cleared to 0.	The input clock of timer unit m is in the stop-providing state. Initialize all circuits and the SFR for each channel. (TOmp bit and TOmq bit become "0" and TOmp pin and TOmq pin become port function)

Remark: m: unit number (m= 0, 1) n: master channel number (n=0)

p: slave channel number q: slave channel number $n < p < q \leqslant 3$ (p and q are integers greater than n)

Chapter 6 EPWM Output Control Circuit

Using the PWM output function of Timer, one DC motor or two stepper motors can be controlled. The output can be truncated by truncating the source CMP0 output, the INTP0 input, and the EVENTC event. The software allows you to select from four outputs: Hi-Z output, low output, high output, and anti-truncation output during forced truncation.

6.1 Structure of output control circuit

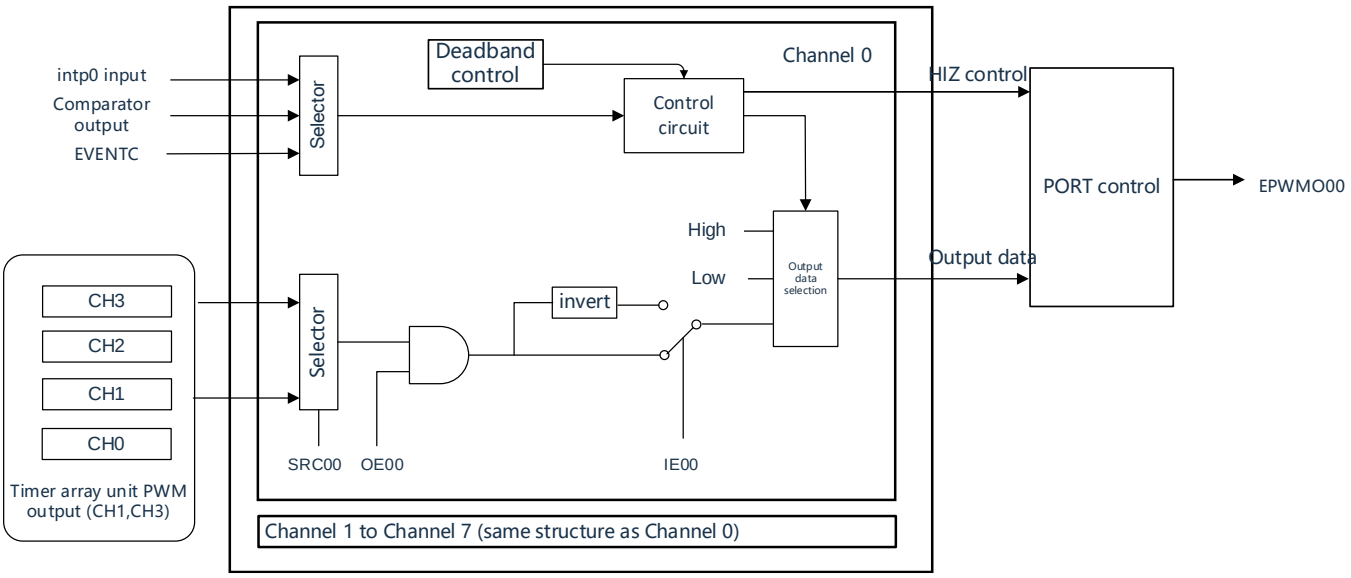
The EPWM output control circuit consists of the following hardware.

Table 6-1: Structure of EPWM output control circuit

Item	Structure
Control register	EPWM input source selection register (EPWMSRC).
	EPWM output control register (EPWMCTL).
	EPWM force truncated input selection register (EPWMSTC).
	EPWM force truncated output selection register (EPWMSTL).
	EPWM status register (EPWMSTR).
	EPWM deadband control register (EPWMDTC)
Output	EPWM output (EPWMO00~EPWMOP07)

The block diagram of the EPWM output control circuit is shown in Figure 6-1.

Figure 6-1: Block diagram of the EPWM output control circuit



6.2 Registers for controlling EPWM output control circuit

The real-time output control circuit is controlled by the following registers.

- Peripheral enable register 0 (PER1)
- EPWM input source select register (EPWMSRC)
- EPWM output control register (EPWMCTL)
- EPWM force truncated input select register (EPWMSTC)
- EPWM force truncated output select register (EPWMSTL)
- EPWM status register (EPWMSTR)
- EPWM deadband control register (EPWMDTC)
- Port mode register (PMxx)
- Port mode control register (PMCxx)
- Port register (Pxx)

6.2.1 Peripheral enable register 1 (PER1)

The PER1 register is a register that sets the clock that enables or disables clocking each peripheral hardware.

Reduce power consumption and noise by stopping clocking unused hardware.

To use the EPWM function, EPWMEN must be set to “1”.

See “4.3.6 peripheral enable registers 0, 1 (PER0, PER1)” for details.

6.2.2 EPWM input source select register (EPWMSRC)

The EPWMSRC register selects the source clock of the input clock of the real-time output circuit. Select Timer's timer output TO01 or TO03 as the source clock and input to the EPWM.

The EPWMSRC register is set via an 8-bit memory operation command.

By generating a reset signal, the value of this register becomes “00H”.

Table 6-2: Format of EPWM input source select register

Address: 0x40044400 After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
EPWMSRC	SRC07	SRC06	SRC05	SRC04	SRC03	SRC02	SRC01	SRC00

SRC0n	Selection of source clock for EPWM0n outputs
0	Selects TO01
1	Selects TO03

Remark: n: channel number (n=0~7)

6.2.3 EPWM force truncated input select register (EPWMSTC)

EPWMSTC register is to select forces truncation of the input source.

The EPWMSTC register is set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register changes to "00H".

Table 6-3: Format of EPWM force truncated input select register (EPWMSTC)

Address: 0x40044404 After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
EPWMSTC	0	0	0	REL_SEL	HS_SEL	IN_EG	SC_SEL1	SC_SEL0

SC_SEL1	SC_SEL0	Selection of truncation sources Note1,3,4
0	0	Do not select
0	1	Do not select
1	0	INTP0 terminal input
1	1	Event input from EVENTC

IN_EG	Source of force truncation/edge selection of force truncation output source Note 1,2
0	Rising edge: Output force truncation Falling edge: Output force truncation released
1	Rising edge: Output force truncation released Falling edge: Output force truncation

HS_SEL	Output mode selection for forced truncation
0	Software release
1	Hardware release

REL_SEL	Release timing selection for forced output truncation
0	After the release signal generated by hardware or software occurs, the truncation is immediately released and the pulse output is restored.
1	After the release signal generated by hardware or software occurs, wait for the following timing: Select TO01 as the channel of the source clock: Truncation is released on the rising edge of the next TO01, and the pulse output is restored. Select TO03 as the channel of the source clock: the cut-off is released on the rising edge of the next TO03 and the pulse output is restored.

Notice:

1. Set SC_SEL1 and SC_SEL0 at least three clocks apart after IN_EG is set.
2. Valid only when INTP0 input is selected.
3. When using EVENTC to release the truncation, software must be selected to release (HS_SEL set to 1). There is no restriction when using INTP0 input.
4. The effective width of the input selected INTP0 must be greater than one clock cycle.

6.2.4 EPWM output control register (EPWMCTL)

The EPWMCTL register enables control and reverse control of the waveform outputs of EPWMO00~EPWMO07.

The EPWMCTL register is set via a 16-bit memory manipulation instruction.

After the reset signal is generated, the value of this register becomes “00H”.

Table 6-4: Format of EPWM output control register (EPWMCTL)

Address: 0x40044408 After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EPWMCTL	IE07	IE06	IE05	IE04	IE03	IE02	IE01	IE00	OE07	OE06	OE05	OE04	OE03	OE02	OE01	OE00

OE0n	Control of EPWMO0n output
0	Disables outputting
1	Enables outputting

Remark: n: channel number (n=0~7)

IE0n	Reverse control of EPWMO0n output
0	Not reversed
1	Reversed

Remark: n: channel number (n=0~7)

6.2.5 EPWM force truncated output select register (EPWMSTL)

The output state of the EPWMO terminal when the EPWMSTL register is forcibly truncated.

The EPWMSTL register is set via a 16-bit memory manipulation instruction.

After the reset signal is generated, the value of this register becomes “00H”.

Table 6-5: Format of EPWM force truncated output select register (EPWMSTL)

Address: 0x4004440C After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EPWMSTL	IO71	IO70	IO61	IO60	IO51	IO50	IO41	IO40	IO31	IO30	IO21	IO20	IO11	IO10	IO01	IO00

IOn1	IOn0	Selection of terminal output when truncated
0	0	Truncation is prohibited
0	1	HI-Z output
1	0	Low level output
1	1	High level output

Remark: n: channel number (n=0~7)

6.2.6 EPWM status register (EPWMSTR)

The EPWMSTR register clears the forced truncation signal and displays the truncation status. If the clear trigger bit HZCLR is set to “1”, the truancy state is released. When the truncation status indicates that the signal of the SHTFLG is high, it enters the forced truncation state. Bit0 is write-only bit, and the read value is always “0”. Bit7~1 is read-only.

The EPWMSTR register is set via an 8-bit memory manipulation instruction.

After the reset signal is generated, the value of this register becomes “00H”.

Table 6-6: Format of EPWM status register (EPWMSTR)

Address: 0x4004410 After reset: 0000H R/W

Symbol	7	6	5	4	3	2	1	0
EPWMSTR	0	0	0	0	0	0	SHTFLG	HZCLR

SHTFLG	Force truncation status flag
0	Normal output state
1	Force truncation state

HZCLR	Software release for forced signal truncation
0	-
1	Software release truncated status

Remark: If truncation is prohibited by setting the forced truncatd output selection register (EPWMSTL), no truncation is performed even though SHTFLG is set to "1" due to the occurrence of input from an external cutoff source.

6.2.7 EPWM deadband control register (EPWMDTC)

The EPWMDTC register enables the deadband of the waveform output from EPWMO00 to EPWMO07 and controls the deadband value.

The EPWMDTC register is set by a 16-bit memory manipulation instruction.

After a reset signal is generated, the value of this register changes to "00H".

Table 6-7: Format of EPWM deadband control register (EPWMDTC)

Address: 0x40044414 After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EPWMDTC	DTCVAL [7:0]								EN 7	EN 6	EN 5	EN 4	EN 3	EN2	EN1	EN0

DTCVAL[7:0]	Dead time (shared by 8 channels)

ENn	Deadband enable for EPWMO0n outputs
1	Enables
0	Dsiables

Remark: n: channel number (n=0~7)

6.2.8 Control registers for port functions of EPWM output pins

When using the EPWM output, the control register (Port Mode Register (PMxx, PMCxx)) for the port function multiplexed with the EPWM output pin (EPWMOn pin) must be set. For details, refer to “2.3.1 Port Mode Register (PMxx)”.

When using the multiplexed ports of the EPWM pins as outputs of EPWMO, the bits of the port mode registers (PMxx, PMCxx) corresponding to each port must be set to “0”. In this case, the bit of the port register (Pxx) can be “0” or “1“.

For details, refer to “Table 6-2 Pin Function Figures Map”

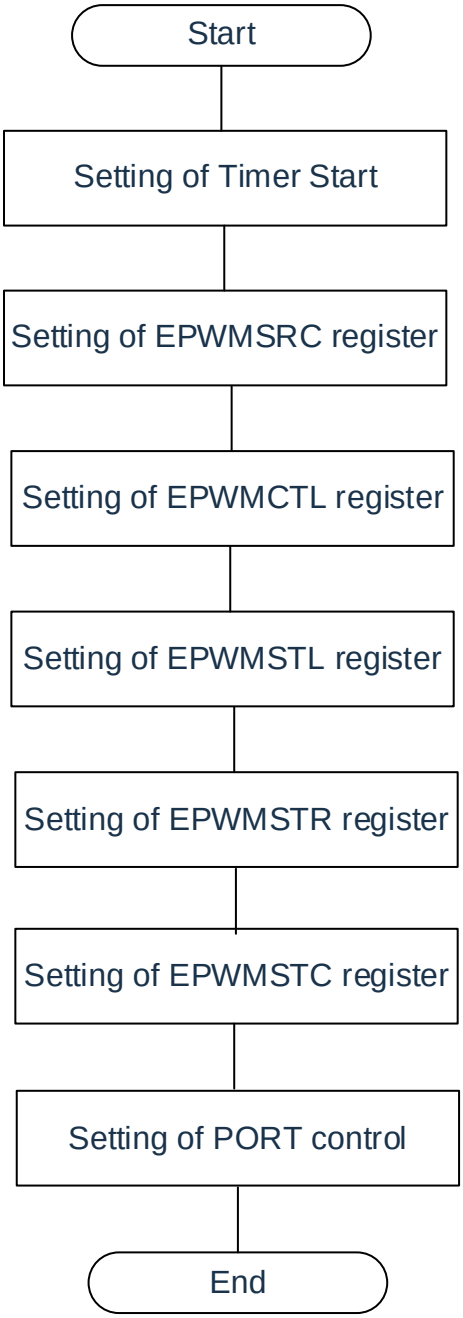
6.3 Operation of EPWM output control circuit

6.3.1 Initial setup

The timer waveform selects the TAU output (TO01, TO03) as the source clock through the EPWSRC register. The positive or inverting phase of the timer waveform can be fixed by setting the EPWMCTL register.

In the event of forced truncation, the Hi-Z output, low output, high output, or disable cut-off output can be selected through the setting of the EPWMSTL register.

Figure 6-2: Initial configuration flow of registers

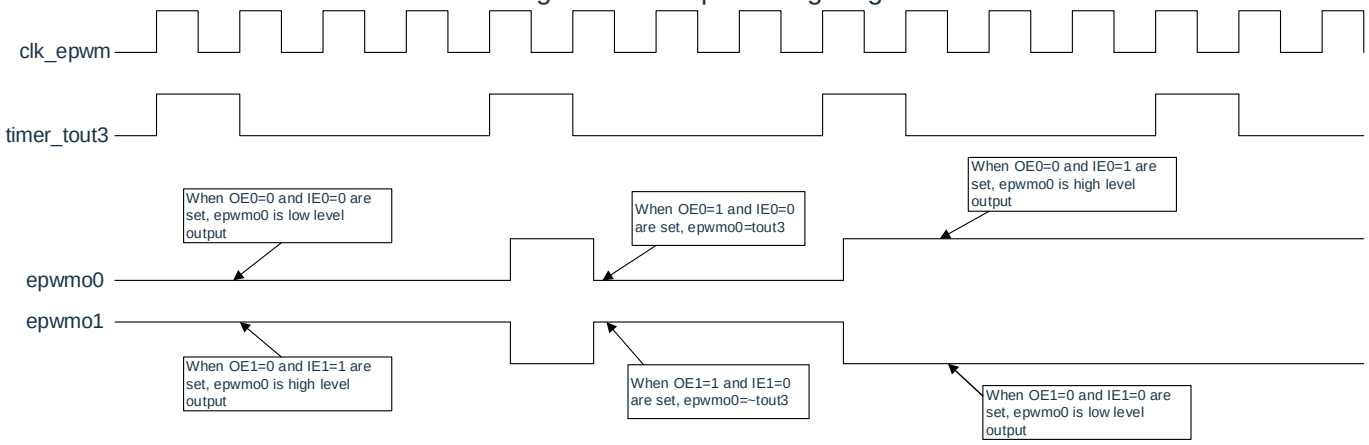


6.3.2 Normal operation

Depending on the register settings, four output data can be selected, namely forward waveform output, inverted waveform output, low level output, and high-level output. The EPWMCTL registers can be changed at runtime. Both OE0n bits and IE0n bits must be written at the same time.

For details, please refer to “Table 6-8 Operation Instructions for truncation signals”.

Figure 6-3: Output timing diagram



6.3.3 Force truncation processing

The EPWM can select CMP0 output, INTP0, and EVENTC event by setting the bit1, 0 of the EPWMSTC register, causing the EPWMO output to enter a forced truncation state.

(1) Occurrence of forced truncation

The truncated state is entered via the CMP0 output, the INTP0 input, and the EVENTC event. By bit2(IN_EG) of EPWMSTC register, it can select the rising or falling edge and enter the truncated state after 1 to 2 clocks. For details, please refer to Figure 6-4.

(2) Release of forced truncation

a) Software release: When bit3 (HS_SEL) of EPWMSTC register is 0, the software release mode is used. Bit0 (HZCLR) of EPWMSTR register is the clear bit of truncated status. When the truncated status flag SHTFLG is high, if the HZCLR bit is set to “1”, the truncated status flag SHTFLG goes low and the forced truncated status is released.

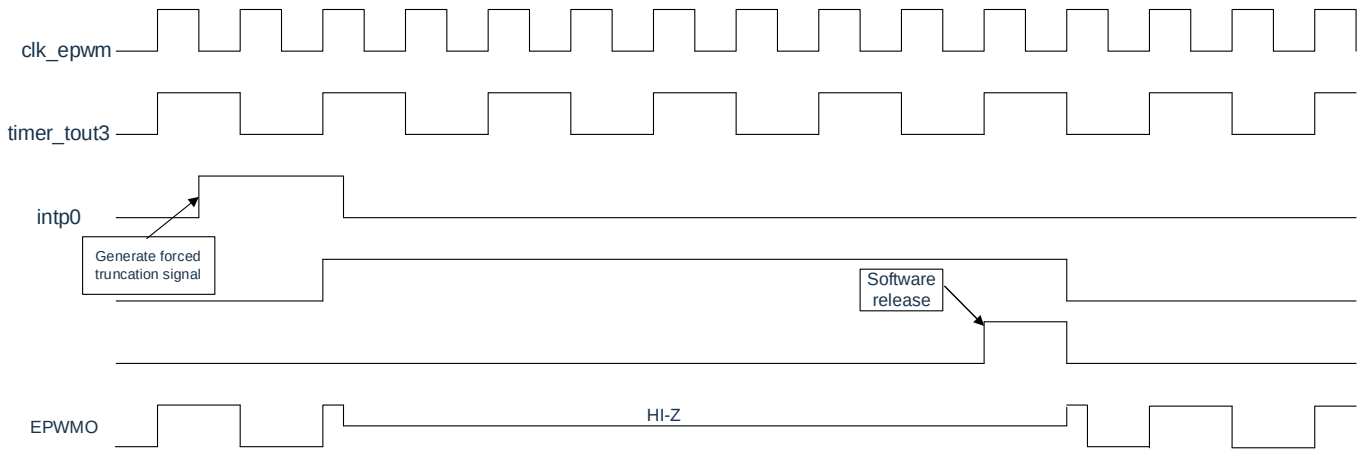
b) Hardware release: When bit3 (HS_SEL) of EPWMSTC register is 1, the hardware release mode is used. The forced truncation state is released by the edge of CMP0 output or INTP0 input.

Table 6-8: Table of operation Instructions for truncation signals

Bit	IOn1-0	OE0n	IE0n	SHTFLG	EPWM output pin
Set value	00	1	0	*	Positive rotation waveform
	00	1	1	*	Invert the waveform
	01	*	*	*	Low level output
	10	*	*	*	High level output
	11	*	*	1	HI-Z output

Remark: n=0~7

Figure 6-4: Timing diagram for generation and release of INTP0 truncation (HS_SEL=0, REL_SEL=0)



Remark: Short pulses may be generated when switching from “normal operation” to “Hi-Z”, “fixed low” or “fixed high” during forced cutoff caused by the cutoff signal INTP0, or when returning to the forced cutoff state by immediate release.

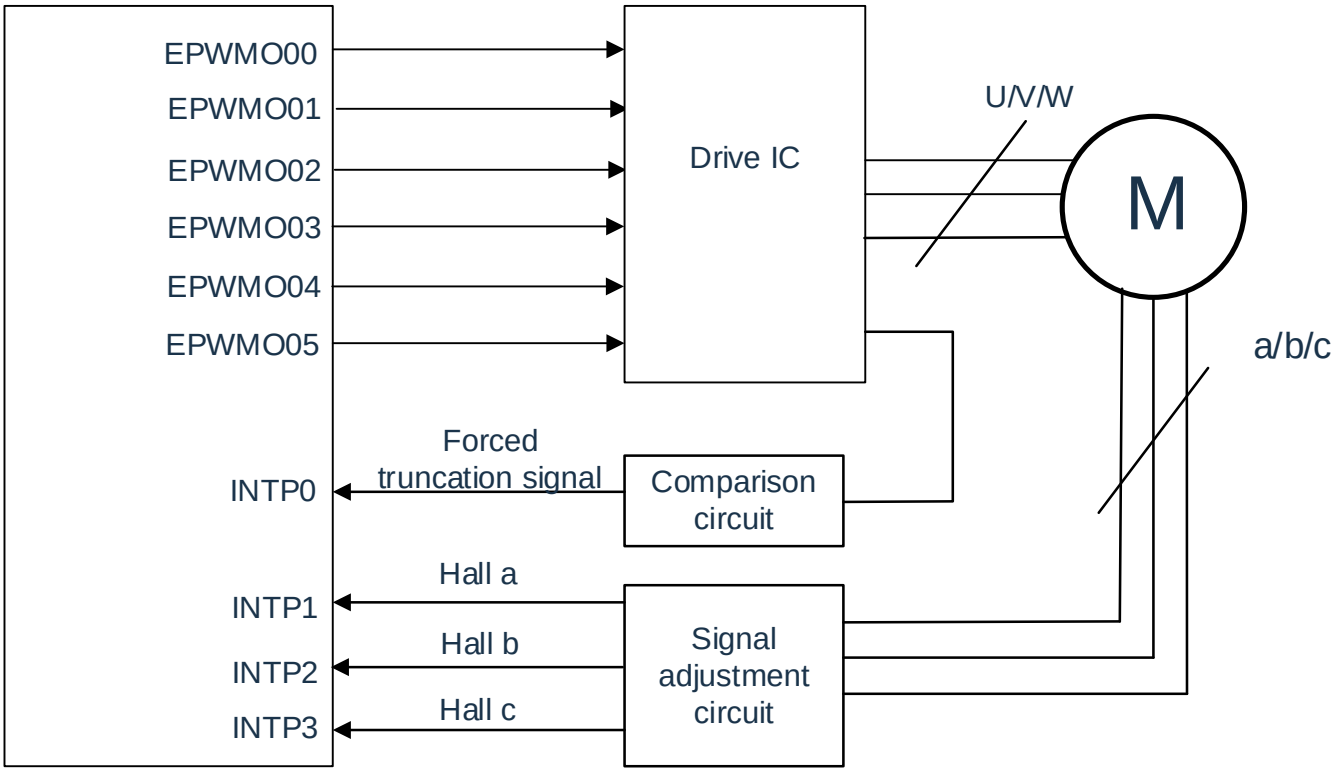
6.4 Control example of brushless DC motor

The following is an example of using the EPWM control function to control a brushless DC motor (hereinafter referred to as a BLDC motor).

6.4.1 Example of hardware connections

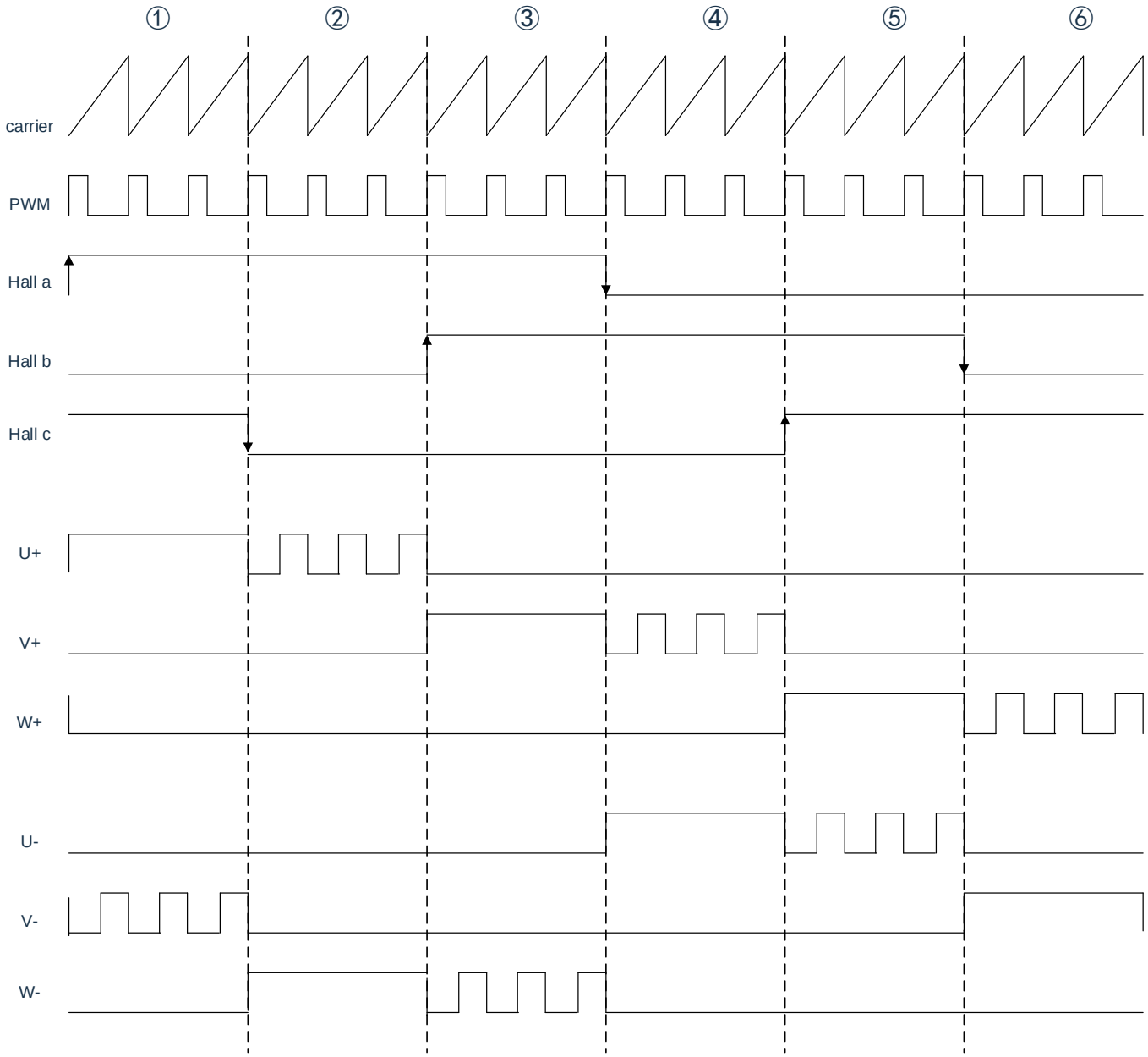
An example of a hardware connection for a brushless DC motor is shown in Figure 6-5. In this example, EPWMO00~EPWMO05 (output) is used for output control of BLDC motors, INTP1~INTP3 (input) are output signals for the Hall sensor, and INTP0 (input) is used to force a truncated signal.

Figure 6-5: Example of hardware connection



6.4.2 Control timing of three-phase brushless DC motors

Figure 6-6: Control timing of three-phase brushless DC motor



6.4.3 Example of register setting

In this example, the EPWM source select register (EPWMSRC) and EPWM control register (EPWMCTL) are initialized to simultaneously output a waveform of positive rotation from EPWM00 ~ EPWM05 to the BLDC motor.

1. Set EPWMSRC5 to EPWMSRC0 in the EPWMSRC register to “0” and channel 1 of Timer as the input source of EPWMO00 ~ EPWMO05.
2. Set EPWMOE3 to EPWMOE0 in the EPWMCTL register to “1” to allow EPWMO03 ~ EPWM00 to be output. Set EPWMIE3 to EPWMIE0 of EPWMCTL register to “0”, EPWMO00 ~ EPWMO03 will be output in positive direction.
3. Set EPWMOE5 to EPWMOE4 in the EPWMCTL register to “1” to allow EPWMO05 to EPWM04 to be output. Set EPWMIE5 ~ EPWMIE4 in the EPWMCTL register to “1” to reverse the output of EPWMO04~ EPWMO05.

Table 6-9: Example of setting EPWMCTL0 register

Description	Set value of the EPWMCTL
State ①: rising edge of Hall a Disable U+, U+ reverse outputs, enable V–, V–forward outputs.	0x0110
State ②: falling edge of Hall c Enable U+, U+ forward outputs, disable W–, W– reverse outputs.	0x2001
State ③: rising edge of Hall b Disable V+, V+ reverse outputs, enable W–, W– forward outputs.	0x0220
State ④: falling edge of Hall a Enable V+, V+ forward outputs, disable U–, U–reverse outputs.	0x0802
State ⑤: rising edge of Hall c Disable W+, W+ reverse outputs, enable U–, U–forward outputs.	0x0408
State ⑥: falling edge of Hall b Enable W+, W+ forward outputs, disable V–, V–reverse outputs.	0x1004

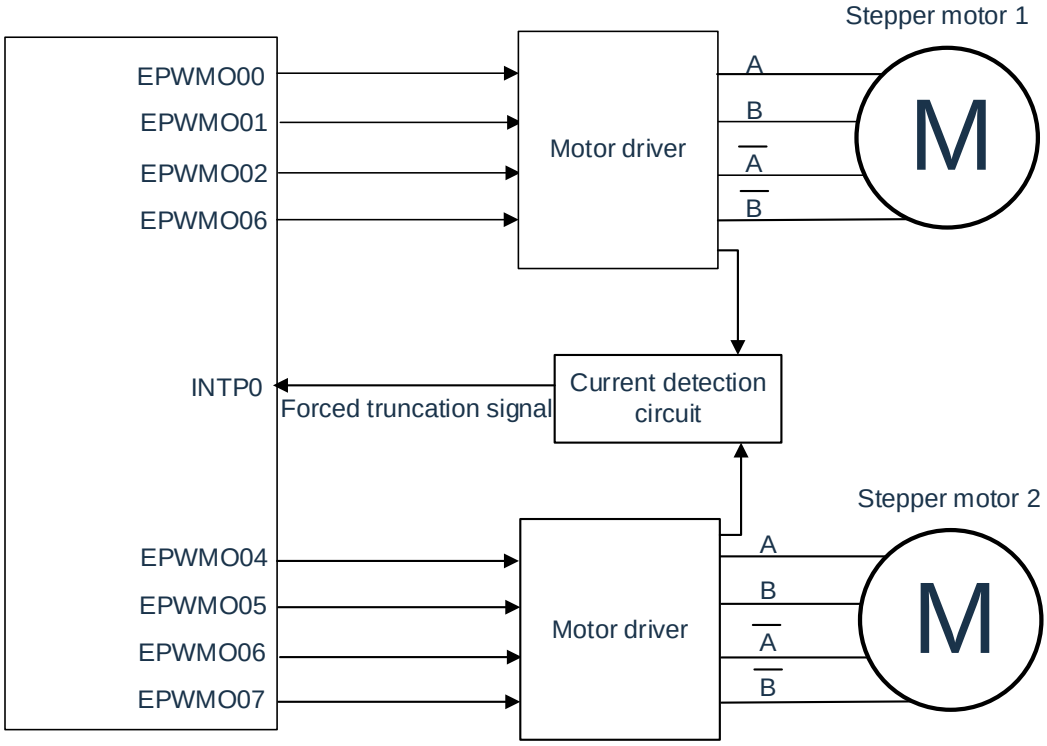
6.5 Example of stepper motor control

The following is an example of using eight real-time outputs to control two 2-phase stepper motors.

6.5.1 Example of a hardware connection

An example of a hardware connection to control two stepper motors is shown in Figure 6-7.

Figure 6-7: Example of a hardware connection



6.5.2 Control method

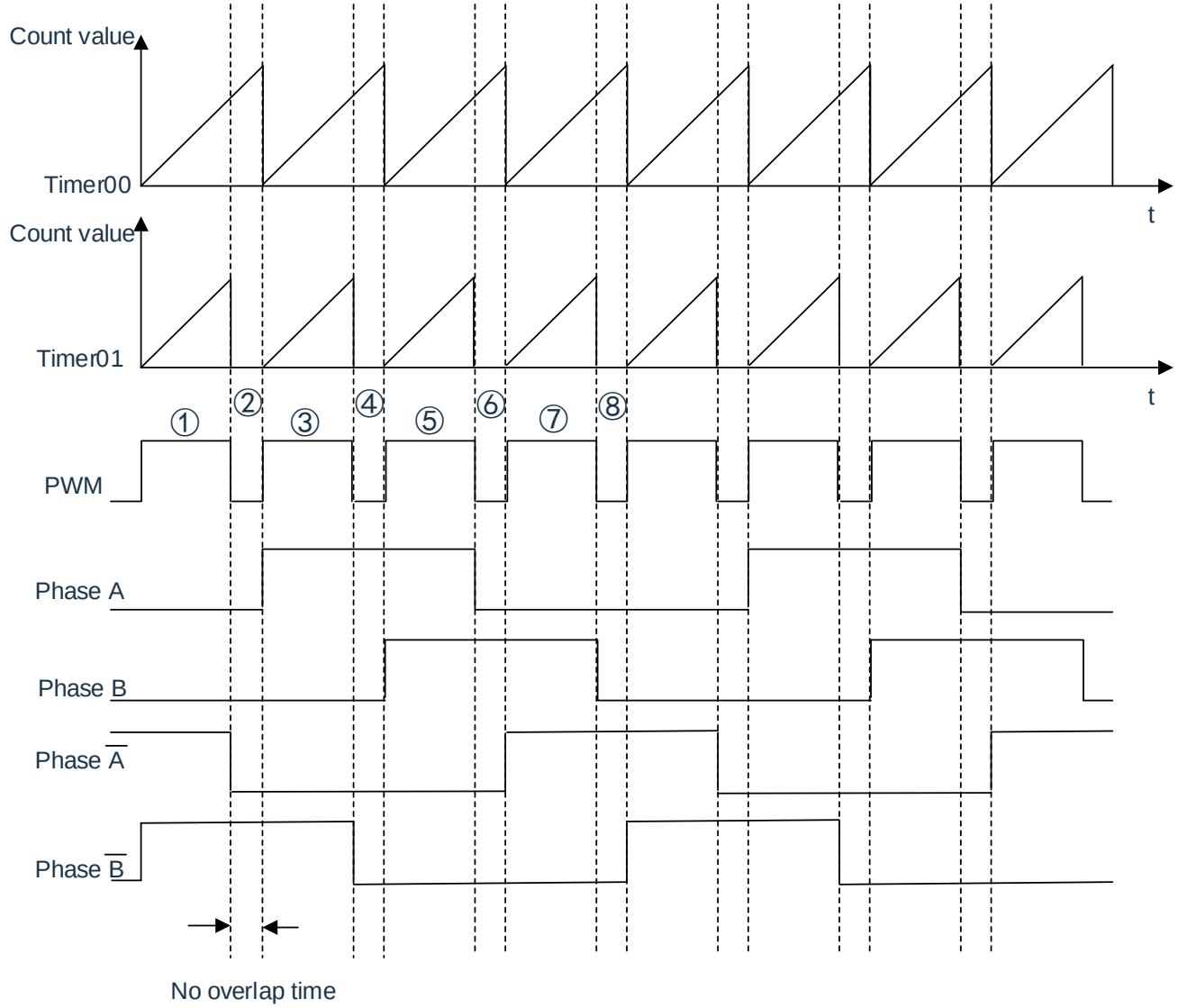
The stepper motor is rotated, reversed or stopped in two-phase excitation mode by using eight EPWMOs. Control the rotation speed via Timer's PWM mode.

In this example, Timer's CH0 and CH1 are used for the control of stepper motor 1, CH2 and CH3 are used for the control of stepper motor 2. If you combine 2 Timer channels, you can generate pulses of any period and duty cycle. CH0 and CH2 are the main control channels and operate as interval timer mode. CH1 and CH3 are slave channels and operate as single-count mode.

In addition, the cross-current prevention time (no overlapping time) is inserted when switching the output type.

An example of a waveform for stepper motor control is shown in Figure 6-8.

Figure 6-8: Waveform example of stepper motor control



6.5.3 Example of register setting

Table 6-10: Example of setting the register that controls the stepper motor

Status		Set value of EPWMSRC	Set value of EPWMCTL
	①	0x00	0x4400
	②	0x00	0x4000
	③	0x00	0x4100
	④	0x00	0x0100
	⑤	0x00	0x0300
	⑥	0x00	0x0200
	⑦	0x00	0x0600
	⑧	0x00	0x0400

Chapter 7 Real-Time Clock

7.1 Function of real-time clock

The real-time clock has the following functions.

Holds counters for years, months, weeks, days, hours, minutes, and seconds up to a maximum of 99 years.

Fixed cycle break (cycles: 0.5 seconds, 1 second, 1 minute, 1 hour, 1 day, 1 month)

Alarm clock interrupt (alarm clock: week, hour, minute)

1Hz pin out capability

7.2 Structure of real-time clock

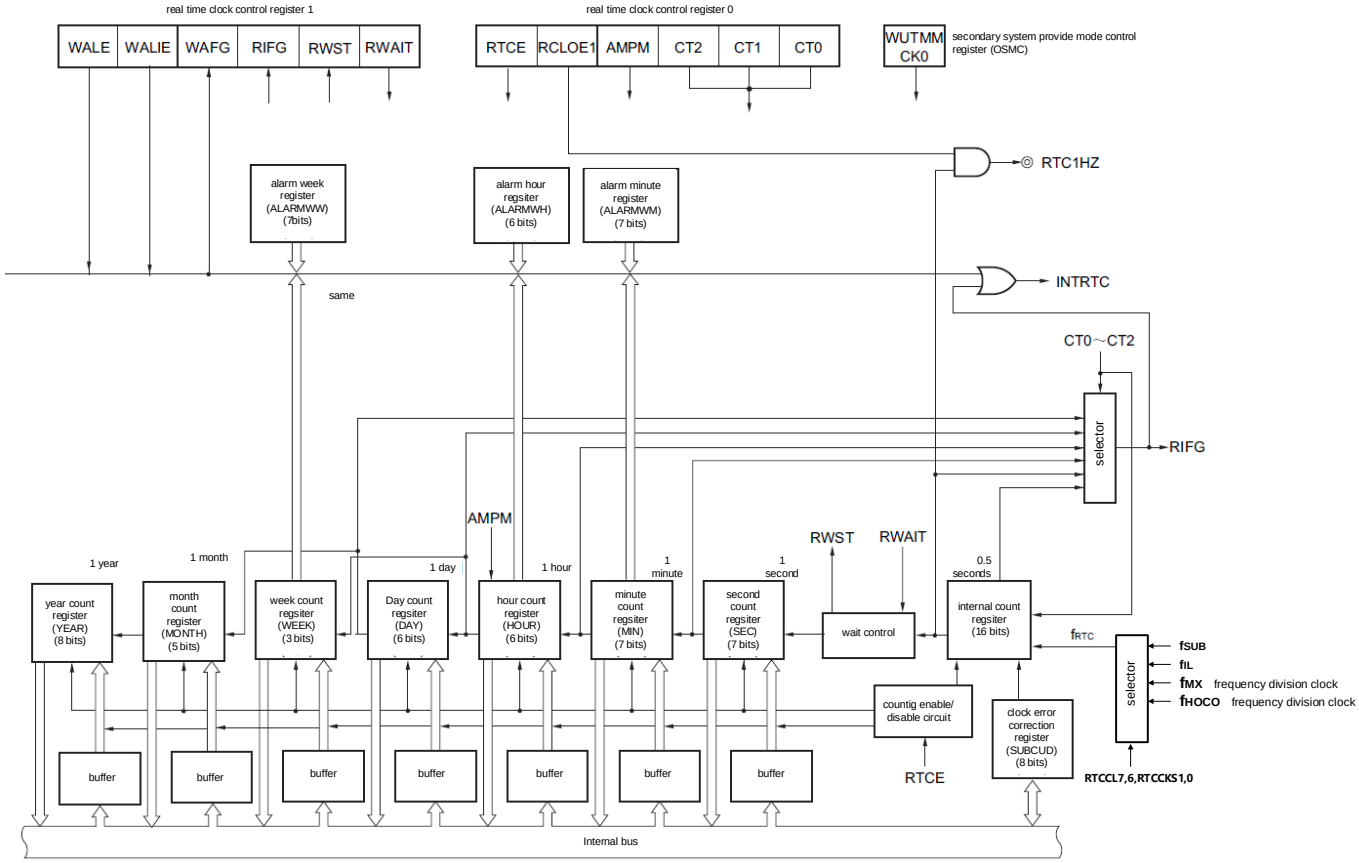
The real-time clock consists of the following hardware.

Table 7-1: Structure of real-time clock

Item	Structure
Counter	Internal counter (16-bit)
Control register	Peripheral enable register 0 (PER0)
	Real-time clock selection register (RTCCL)
	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)
	Clock error correction register (SUBCUD)
	Alarm clock minute register (ALARMWM)
	Alarm clock hour register (ALARMWH)
	Alarm clock week register (ALARMWW)

Notice: The reset of the above RTC control registers are only controlled by the POR reset.

Figure 7-1: Block diagram of real-time clock



Notice: Count years, months, weeks, days, hours, minutes and seconds only if you select F_{mx}/F_{hoco} clock (≈32, 768KHZ after every week) or the secondary system clock (F_{SUB}=32.768kHz) as the running clock for the real-time clock. When a low-speed internal oscillator clock (F_{IL}=15kHz) is selected, only a fixed cycle interrupt function is used. The fixed cycle interrupt interval when selecting F_{IL} is calculated using the following equation: Fixed period (value selected by the RTCC0 register) × F_{SUB}/F_{IL}.

7.3 Registers for controlling real-time clock

The real-time clock is controlled through the following registers.

Peripheral enable register 0 (PER0).

Real-time clock selection register (RTCCL)

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)

Clock error correction register (SUBCUD)

Alarm clock minute register (ALARMWM)

Alarm clock hour register (ALARMWH)

Alarm clock week register (ALARMWW)

Port mode register (PMxx)

Port mode control register (PMCxx)

Port multiplexing function configuration register (PxxCFG)

7.3.1 Peripheral enable register 0 (PER0)

The PER0 register is a register that sets a clock that is allowed or prohibited to supply to each peripheral hardware. Reduce power consumption and noise by stopping clock supply to unused hardware.

You must set bit7 (RTCEN) to “1” when you want to use real-time clocks. The PER0 register is set by an 8-bit memory manipulation instruction. After the reset signal is generated, the value of this register changes to "00H".

Table 7-2: Format of peripheral enable register 0 (PER0)

Address: 0x40020420 After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
PER0	RTCEN	0	ADCEN	IICA0EN	SAU1EN	SAU0EN	TM41EN	TM40EN

RTCEN	Control of an input clock of a real-time clock (RTC) and a 15-bit interval timer
0	Stop to supply the input clock. You cannot write the SFR used by the real-time clock (RTC) and 15-bit interval timers. The real-time clock (RTC) and the 15-bit interval timer are reset.
1	Supply the input clock. SFRs that can read and write real-time clocks (RTCs) and 15-bit interval timers.

Notice:

1. If you want to use the real-time clock, you must first set the RTCEN bit to "1" while the counting clock (F_{RTC}) oscillation is stable, and then set the following registers. When the RTCEN bit is "0", the write operation of the real-time clock control register is ignored, and the read values are initial (except RTCCL, port mode register, and port register).
   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)
   Clock error correction register (SUBCUD)
   Alarm clock minute register (ALARMWM)
   Alarm clock hour register (ALARMWH)
   Alarm clock week register (ALARMWW)
2. By setting the RTCLPC bit in the Subsystem Clock Supply Mode Control Register (OSMC) to "1", the subsystem clock can be stopped for peripheral functions other than the real-time clock and 15-bit interval timer in deep sleep mode or sleep mode running with the subsystem clock.

7.3.2 Real-time clock selection register (RTCCL)

A real-time clock and a count clock of a 15-bit interval timer (F_{RTC}) can be selected through RTCCL.

Table 7-3: Format of real-time clock selection register (RTCCL)

Address: 0x4004047C	After reset: 00H		R/W				
Symbol	7	6	5	4	3	2	1	0
RTCCL	RTCCL7	RTCCL6	RTCCL5	0	0	0	RTCCKS1	RTCCKS0

RTCCL7	Selection of real-time clock, 15-bit interval timer count clock source
0	Select high-speed system clock (F_{MX})
1	Select high-speed on-chip oscillator (F_{HOCO})

RTCCKS1	RTCCKS0	RTCCL6	RTCCL5	Selection of operation clock for real time clock, count clock of 15-bit interval timer
0	0	x	x	Subsystem Clock (F_{SUB})
0	1			Low-speed internal oscillator clock (F_{IL}) (WUTMMCK0 must set to 1)
1	0	0	1	Main clock F_{MAX}/F_{HOCO} (via RTCCL7 selection)/1952
1	0	0	0	Main clock F_{MAX}/F_{HOCO} (via RTCCL7 selection)/1464
1	0	1	0	Main clock F_{MAX}/F_{HOCO} (via RTCCL7 selection)/976
1	1	0	0	Main clock F_{MAX}/F_{HOCO} (via RTCCL7 selection)/488
1	1	1	0	Main clock F_{MAX}/F_{HOCO} (via RTCCL7 selection)/244

7.3.3 Real-time clock control register 0 (RTCC0)

This is an 8-bit register that sets the start or stop of real-time clock operation, the control of RTC1HZ pins, the 12/24-hour system and fixed cycle interrupts.

The RTCC0 register is set by an 8-bit memory manipulation instruction. After the reset signal is generated, the value of this register changes to "00H".

Table 7-4: Format of real-time clock control register 0 (RTCC0)

Address: 0x40044F5D | After reset: 00H | R/W

Symbol	7	6	5	4	3	2	1	0
RTCC0	RTCE	0	RCLOE1 Note	0	AMPM	CT2	CT1	CT0

RTCE	Real-time clock operation control
0	Stop the counter running.
1	Start the counter running.

RCLOE1	Output control of RTC1HZ pin
0	Disables RTC1HZ pin output (1Hz).
1	Enables RTC1HZ pin output (1Hz).

AMPM	Selection of 12-hour system/24-hour system
0	12-hour system (indicates morning or afternoon).
1	24-hour system
To change the value of the AMPM bit, the RWAIT bit (bit 0 of the Real Time Clock Control Register 1 (RTCC1)) must be set to "1" and then rewritten. If the value of the AMPM bit is changed, the value of the hour count register (HOUR) changes to the corresponding value of the set time system. The time bits are represented as shown in Table 7-11.	

CT2	CT1	CT0	Selection of fixed cycle interrupt (INTRTC)
0	0	0	The fixed-cycle interrupt function is not used.
0	0	1	Once every 0.5 seconds (synchronized with seconds accumulation)
0	1	0	Once every 1 second (synchronized with seconds accumulation)
0	1	1	Once every minute (00 seconds per minute).
1	0	0	Once every hour (00 minutes and 00 seconds per hour).
1	0	1	Once a day (00:00:00 per day).
1	1	×	Once a month (1st of each month at 00:00:00 a.m.).
To change the value of bits CT2~CT0 while the counter is running (RTCE=1), you must do the rewrite after setting INTRTC to disable interrupt processing via the interrupt mask flag register, and you must clear the RIFG flag and RTCIF flag after the rewrite, and then set it to enable interrupt processing.			

Note 1: When the RTCE bit is "1", the RCLOE1 bit cannot be changed.

Note 2: When the RTCE bit is "0", 1Hz is not output even if the RCLOE1 is set to "1".

Remark: ×: Ignore

7.3.4 Real-time clock control register 1 (RTCC1)

This is an 8-bit register that controls the alarm clock interrupt function and counter wait. The RTCC1 register is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "00H".

Table 7-5: Format of real-time clock control register 1 (RTCC1) (1/2)

Address: 0x40044F5E | After reset: 00H | R/W

Symbol	7	6	5	4	3	2	1	0
RTCC1	WALE	WALIE	0	WAFG	RIFG	0	RWST	RWAIT

WALE	Operation control of the alarm clock
0	Consistent operation is invalid.
1	Consistent operation is valid.
To set the WALE bit when the counter is running (RTCE=1) and the WALIE bit is "1", you must set INTRTC to disable interrupt processing through the interrupt mask flag register before the rewrite and clear the WAFG flag and RTCIF flag after the rewrite. To set each alarm register (WALIE flag of RTCC1 register, alarm minute register (ALARMWM), alarm hour register (ALARMWH) and the alarm week register (ALARMWW)), the WALE bit must be set to "0" (invalid for consistent operation).	

WALIE	Operation control of the alarm clock interrupt (INTRTC) function
0	No consistent alarm interruptions.
1	Generate consistent alarm interruptions.

WAFG	Alarm clock detection status flag
0	The alarm clock is inconsistent.
1	Consistent alarms detected.
This is a status flag that indicates that an alarm clock has been detected consistently. It is only valid when the WALE bit is "1" and changes to "1" after one FRTC clock has elapsed and the alarm is detected. Clear this flag by writing "0" to it. Writing a "1" is not valid.	

Table 7-6: Format of real-time clock control register 1 (RTCC1) (2/2)

RIFG	Fixed-cycle interrupt status flag
0	No fixed-cycle interruptions are generated.
1	Generate fixed-cycle interrupts.
This is a status flag that indicates a fixed-cycle interrupt. When a fixed-cycle interrupt is generated, this flag is "1". Clear this flag by writing "0" to it. Writing a "1" is not valid.	

RWST	Wait status flag for the real-time clock
0	The counter is running.
1	It is in read-write mode for the counter.
This is the state that indicates whether the setting of the RWAIT bit is valid. The count value must be read and written after confirming that this flag is "1".	

RWAIT	Wait control of the real-time clock
0	Set to counter run.
1	Set the SEC to YEAR counter to stop running and enter the read/write mode of the counter
This bit controls the operation of the counter. To read and write a count value, you must write "1" to this bit. Because the internal counter (16-bit) continues to run, the read and write must end within 1 second and then return to "0". The time required from the RWAIT bit set to "1" to the time the count value can be read and written (RWST=1) is up to 1 F_{RTC} clock. If an internal counter (16 bits) overflows when the RWAIT bit is "1", the overflow state is maintained and the count is incremented after the RWAIT bit becomes "0". However, when the second count register is written, the overflow state that occurs is not maintained.	

Notice:

1. Fixed-cycle interrupts and alarm-consistent interrupts use the same interrupt source (INTRTC). In the case of using these two interrupts at the same time, when INTRTC occurs, you can determine which interrupt occurred by acknowledging the fixed-cycle interrupt status flag (RIFG) and the alarm detection status flag (WAFG).
2. If writing to the second count register (SEC), then clear the internal counter (16-bit).

7.3.5 Clock error correction register (SUBCUD)

This is a register that can correct the clock speed with high accuracy by changing the overflow value (reference value: 7FFFH) from the internal counter (16 bits) to the second count register (SEC).

The SUBCUD register is set by a 16-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "0000H".

Table 7-7: Format of clock error correction register (SUBCUD)

Address: 0x40044F34H | After reset: 0000H | R/W

Symbol	15	14	13	12	11	10	9	8
SUBCUD	DEV	0	0	F12	F11	F10	F9	F8
	7	6	5	4	3	2	1	0
	F7	F6	F5	F4	F3	F2	F1	F0

DEV	Setting of clock error correction timings
0	Corrects clock error when the second digits are at 00, 20, or 40 (every 20 seconds).
1	Corrects clock error only when the second digits are at 00 (every 60 seconds).
Writing to the SUBCUD register at the following timing is prohibited. When DEV = 0 is set: For a period of SEC = 00H, 20H, 40H When DEV = 1 is set: For a period of SEC = 00H	

F12	Setting of clcok error correction value
0	Increases by {(F11,F10,F9,F8,F7,F6,F5,F4,F3,F2,F1,F0)-1}×2
1	Decreases by {(/ F11,/ F10,/ F9,/ F8,/ F7,/ F6,/ F5,/ F4,/ F3,/ F2,/ F1,/ F0)+1}×2
When (F12,F11,F10,F9,F8,F7,F6,F5,F4,F3,F2,F1,F0)=(0,0,0,0,0,0,0,0,0,0,0,0,0),(0,0,0,0,0,0,0,0,0,0,0,0,1),(1,0,0,0,0,0,0,0,0,0,0,0,0) or (1,0,0,0,0,0,0,0,0,0,0,0,1), the clock error is not corrected. Range of correction value: (F12=0)2, 4, 6, 8, ……, 8186, 8188 (F12=1)-2, -4, -6, -8, ……, -8186, -8188	

Remark: "/" indicates the inverted value of each bit.

The range of value that can be corrected by using the clock error correction register (SUBCUD) is shown below.

	DEV = 0 (correction every 20 seconds)	DEV = 1 (correction every 60 seconds)
Correctable range	-12496.9ppm~12496.9ppm	-4165.6ppmto4165.6ppm
Maximum excludes quantization error	±1.53ppm	±0.51ppm
Minimum resolution	±3.05ppm	±1.02ppm

Notice: If a correctable range is -4165.6 ppm or lower and 4165.6 ppm or higher, set 0 to DEV.

7.3.6 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 counts up when the internal counter (16-bit) overflows.

When data is written to this register, it is written to a buffer and then to the counter up to two cycles of F_{RTC} later. Set a decimal value of 00 to 59 to this register in BCD code.

The SEC register is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "00H".

Table 7-8: Format of second count register (SEC)

Address: 0x40044F52	After reset: 00H		R/W				
Symbol	7	6	5	4	3	2	1	0
SEC	0	SEC40	SEC20	SEC10	SEC8	SEC4	SEC2	SEC1

Notice:

1. When it reads or writes from/to the register while the counter is in operation (RTCE = 1), follow the procedures described in “7.4.3 Reading/writing real-time clock”.
2. The internal counter (16-bit) is cleared when the second count register (SEC) is written.

7.3.7 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 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 two cycles 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 is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "00H".

Table 7-9: Format of minute count register (MIN)

Address: 0x40044F53	After reset: 00H		R/W				
Symbol	7	6	5	4	3	2	1	0
MIN	0	MIN40	MIN20	MIN10	MIN8	MIN4	MIN2	MIN1

Notice: When it reads or writes from/to the register while the counter is in operation (RTCE = 1), follow the procedures described in “7.4.3 Reading/writing real-time clock”.

7.3.8 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 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 two cycles 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 is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "12H".

However, the value of this register is 00H if the AMPM bit is set to 1 after reset.

Table 7-10: Format of hour count register (HOUR)

Address: 0x40044F54	After reset: 12H		R/W				
Symbol	7	6	5	4	3	2	1	0
HOUR	0	0	HOUR20	HOUR10	HOUR8	HOUR4	HOUR2	HOUR1

Notice:

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 it reads or writes from/to the register while the counter is in operation (RTCE = 1), follow the procedures described in “7.4.3 Reading/writing real-time clock”.

Table 7-11 shows the relationship between the setting value of the AMPM bit, the hour count register (HOUR) value, and time.

Table 7-11: 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 indicates AM/PM. 0 for AM and 1 for PM.

7.3.9 Day 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 counts up when the hour counter overflows. This counter counts as follows:

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 two cycles 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 is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "01H".

Table 7-12: Format of day count register (DAY)

Address: 0x40044F56H	After reset: 01H		R/W				
Symbol	7	6	5	4	3	2	1	0
DAY	0	0	DAY20	DAY10	DAY8	DAY4	DAY2	DAY1

Notice: When it reads or writes from/to the register while the counter is in operation (RTCE = 1), follow the procedures described in “7.4.3 Reading/writing real-time clock”.

7.3.10 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 counts up in synchronization with the day counter.

When data is written to this register, it is written to a buffer and then to the counter up to two cycles of F_{RTC} later. Set a decimal value of 00 to 06 to this register in BCD code.

The WEEK register is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "00H".

Table 7-13: Format of week count register (WEEK)

Address: 0x40044F55H	After reset: 00H		R/W				
Symbol	7	6	5	4	3	2	1	0
WEEK	0	0	0	0	0	WEEK4	WEEK2	WEEK1

Notice: 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

Notice: When it reads or writes from/to the register while the counter is in operation (RTCE = 1), follow the procedures described in “7.4.3 Reading/writing real-time clock”.

7.3.11 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 counts up when the day counter overflows.

When data is written to this register, it is written to a buffer and then to the counter up to two cycles 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 is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "01H".

Table 7-14: Format of month count register (MONTH)

Address: 0x40044F57H	After reset: 01H		R/W				
Symbol	7	6	5	4	3	2	1	0
MONTH	0	0	0	MONTH10	MONTH8	MONTH4	MONTH2	MONTH1

Notice: When it reads or writes from/to the register while the counter is in operation (RTCE = 1), follow the procedures described in “7.4.3 Reading/writing real-time clock”.

7.3.12 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 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 two cycles 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 is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "00H".

Table 7-15: Format of year count register (YEAR)

Address: 0x40044F58H	After reset: 00H		R/W				
Symbol	7	6	5	4	3	2	1	0
YEAR	YEAR80	YEAR40	YEAR20	YEAR10	YEAR8	YEAR4	YEAR2	YEAR1

Notice: When it reads or writes from/to the register while the counter is in operation (RTCE = 1), follow the procedures described in “7.4.3 Reading/writing real-time clock”.

7.3.13 Alarm minute register (ALARMWM)

This register is used to set minutes of alarm.

The ALARMWM register can be set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "00H".

Note: 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.

Table 7-16: Format of alarm minute register (ALARMWM)

Address: 0x40044F5AH	After reset: 00H		R/W				
Symbol	7	6	5	4	3	2	1	0
ALARMWM	0	WM40	WM20	WM10	WM8	WM4	WM2	WM1

7.3.14 Alarm hour register (ALARMWH)

This register is used to set hours of alarm.

The ALARMWH register can be set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "12H".

However, the value of this register is 00H if the AMPM bit is set to 1 after reset.

Notice: Set a decimal value of 00 to 23, 01 to 12, or 21 to 32 to this register in BCD code. If a value outside the range is set, the alarm is not detected.

Table 7-17: Format of alarm hour register (ALARMWH)

Address: 0x40044F5BH	After reset: 12H		R/W				
Symbol	7	6	5	4	3	2	1	0
ALARMWH	0	0	WH20	WH10	WH8	WH4	WH2	WH1

Notice: Bit 5 (WH20) of the ALARMWH register indicates AM(0)/PM(1) if AMPM = 0 (if the 12-hour system is selected).

7.3.15 Alarm week register (ALARMWW)

This register is used to set date of alarm.

The ALARMWW register can be set by an 8-bit memory opearation instruction. After a reset signal is generated, the value of this register becomes "00H".

Table 7-18: Format of alarm week register (ALARMWW)

Address: 0x40044F5CH | After reset: 12H | R/W

Symbol	7	6	5	4	3	2	1	0
ALARMWW	0	WW6	WW5	WW4	WW3	WW2	WW1	WW0

Here is an example of setting the alarm.

Time of alarm	Day							12-hour display				24-hour display			
	Sunday WW 0	Monday WW 1	Tuesday WW 2	Wednesday WW 3	Thursday WW 4	Friday WW 5	Saturday WW 6	Hour 10	Hour 1	Minute 10	Minute 1	Hour 10	Hour 1	Minute 10	Minute 1
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
Mon~Fri 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
Mon, Wed, Fri 11:59 p.m.	0	1	0	1	0	1	0	3	1	5	9	2	3	5	9

7.3.16 Port mode register and port register

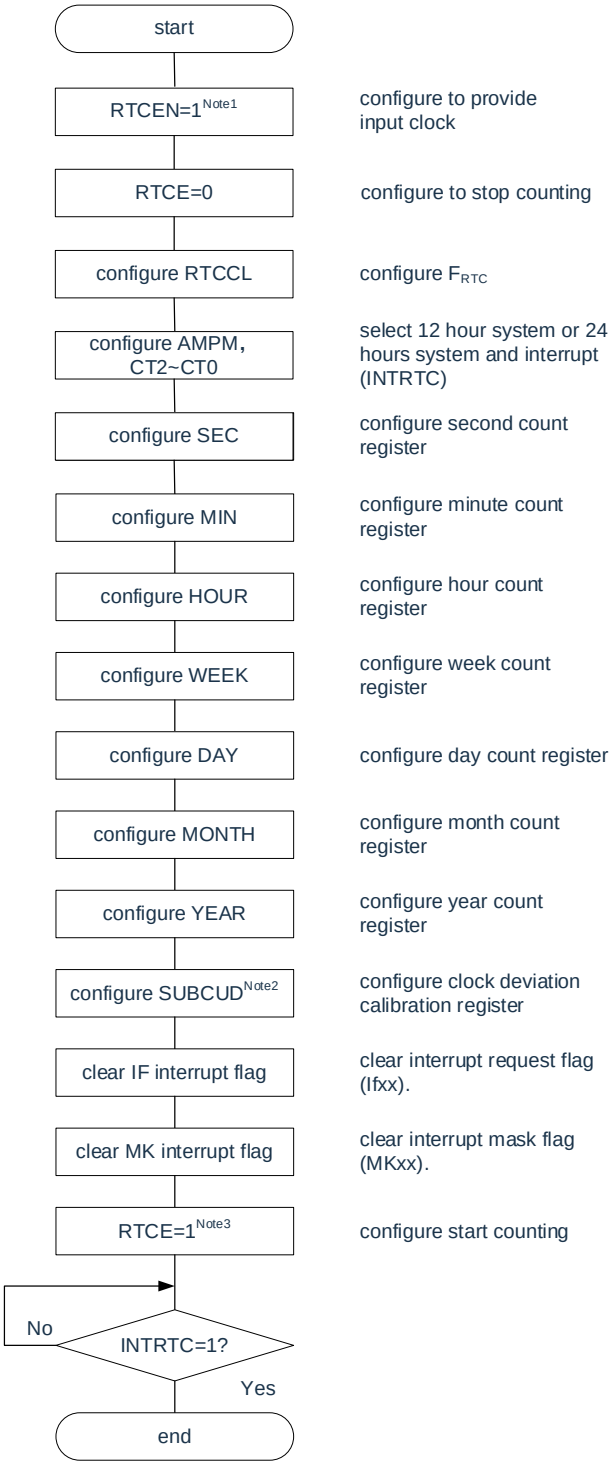
To output the multiplexed port of the RTC1HZ output pin with 1Hz, you must set "0" to the bit of the Port Mode Control Register (PMCxx), the bit of the Port Mode Register (PMxx), and the bit of the Port Register (Pxx) corresponding to each port.

The set port mode registers (PMxx), port registers (Pxx), and port mode control registers (PMCxx) differ by product. For more information, refer to “2.3 Registers for controlling port functions”.

7.4 Operation of real-time clock

7.4.1 Start of real-time clock operation

Figure 7-2: Procedure for starting operation of real-time clock



Note 1: First set the RTCEN bit to 1, while oscillation of the count clock (F_{RTC}) is stable.

Note 2: Set up the register only if the clock error must be corrected. For details about how to calculate the correction value, see “7.4.6 Example of clock error correction of real-time clock”.

Note 3: Confirm the procedure described in “7.4.2 Shifting to sleep mode after starting operation” when shifting to sleep mode without waiting for INTRTC = 1 after RTCE = 1.

7.4.2 Shifting to sleep mode after starting operation

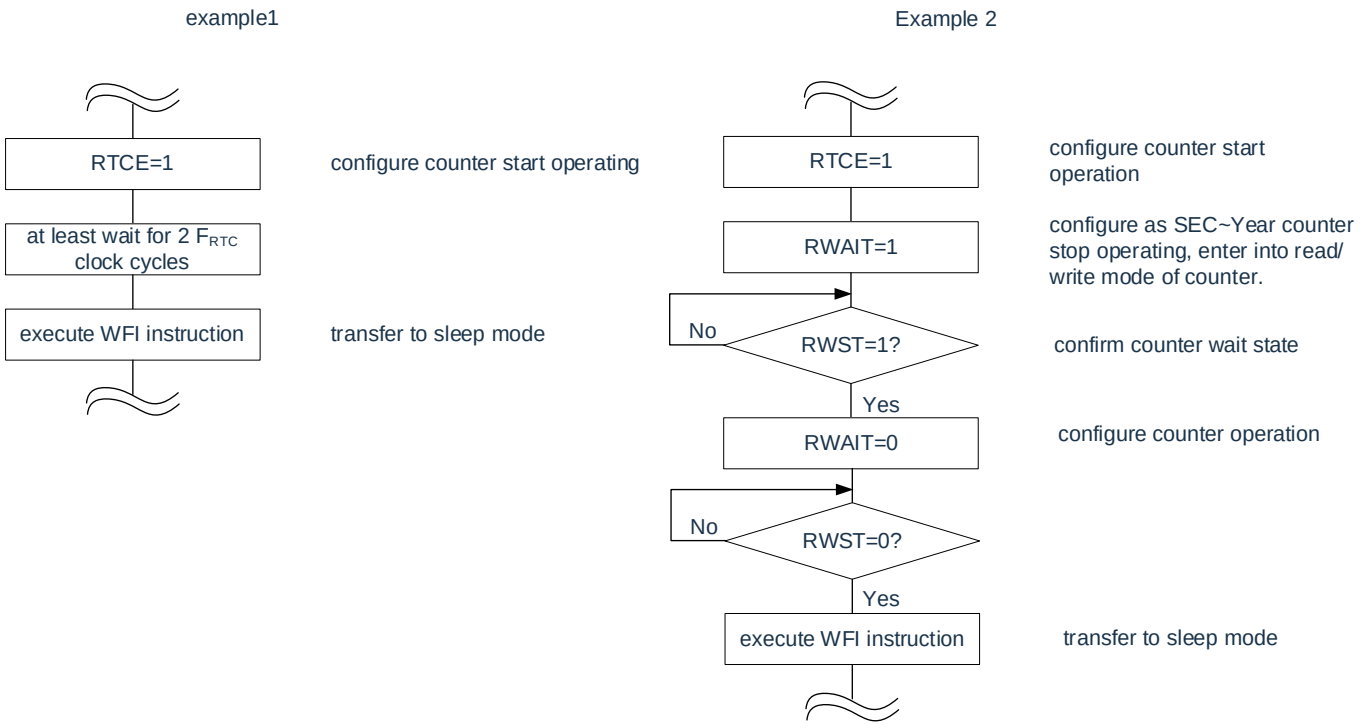
Perform some of the following processing when shifting to sleep mode immediately after setting the RTCE bit to 1.

However, after setting the RTCE bit to 1, this processing is not required when shifting to sleep mode after the INTRTC interrupt has occurred.

Shifting to sleep mode when at least two cycles of the count clock (F_{RTC}) have elapsed after setting the RTCE bit to 1 (see Figure 7-3, Example 1).

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 sleep mode after checking again by polling that the RWST bit has become 0 (see Figure 7-3, Example 2).

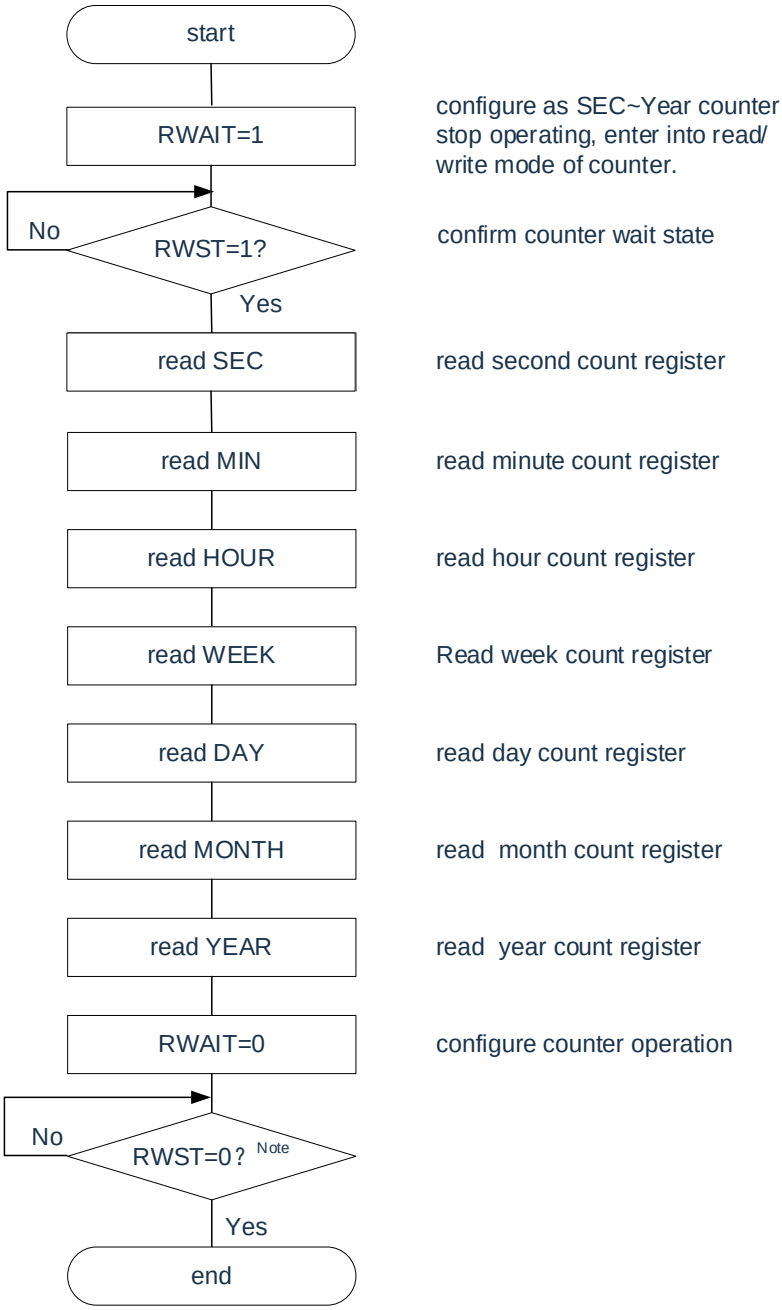
Figure 7-3: Procedure for shifting to sleep/deep sleep mode after setting RTCE bit to 1



7.4.3 Reading/writing real-time clock

Read or write the counter after setting “1” to RWAIT first. Set RWAIT to “0” after completion of reading or writing the counter.

Figure 7-4: Procedure for reading real-time clock

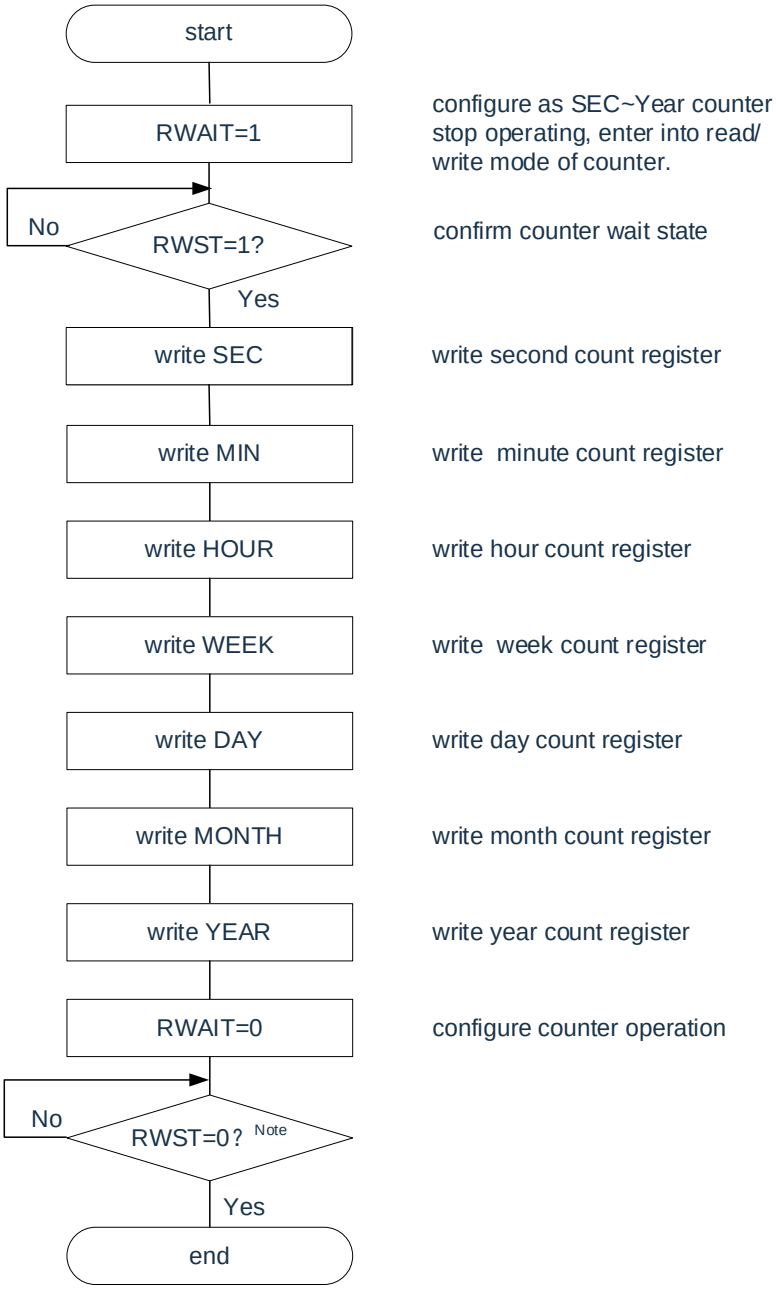


Note: Be sure to confirm that RWST = 0 before setting sleep mode.

Notice:

1. Complete the series of process of setting the RWAIT bit to 1 to clearing the RWAIT bit to 0 within 1 second.
2. The second count register (SEC), minute count register (MIN), hour count register (HOUR), week count register (WEEK), day count register (DAY), month count register (MONTH), and year count register (YEAR) may be read in any sequence. All the registers do not have to read and only some registers may be read.

Figure 7-5: Procedure for reading real-time clock



Note 1: Be sure to confirm that RWST = 0 before setting sleep mode.

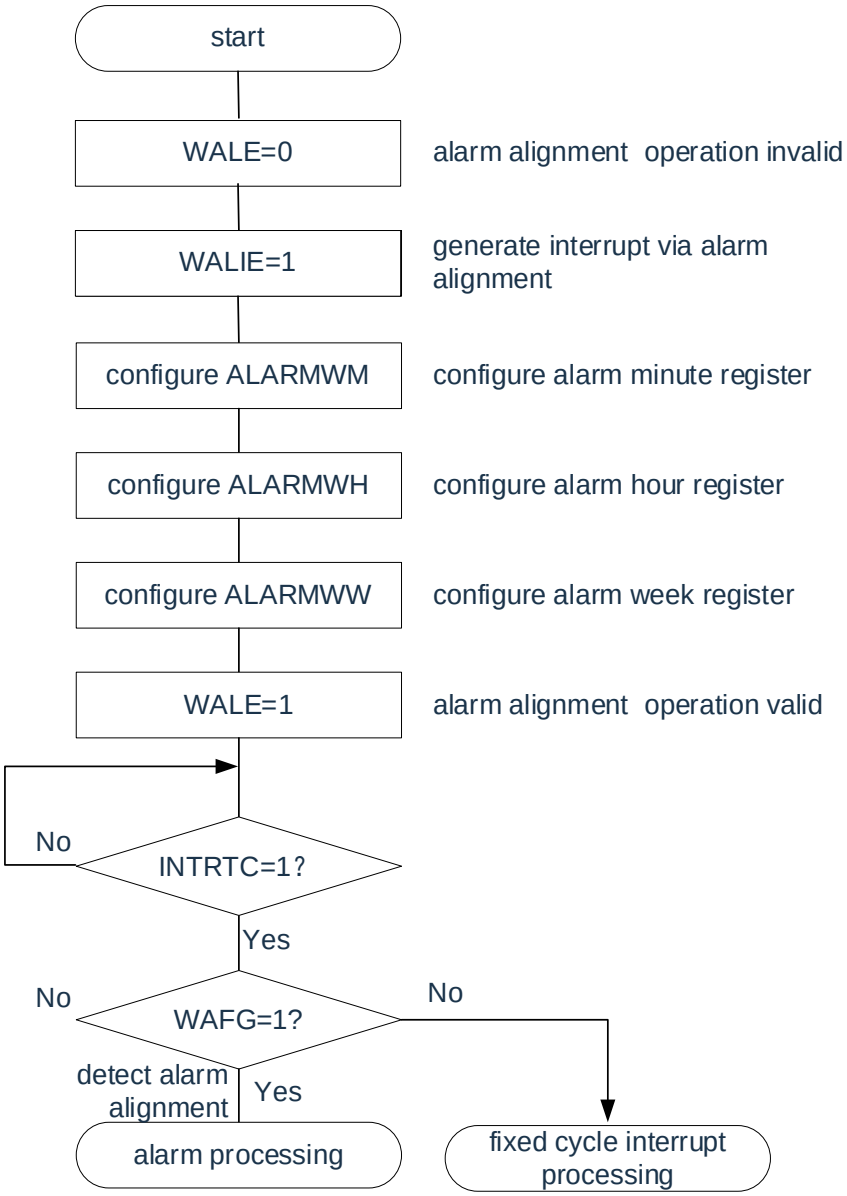
Notice:

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 after disabling interrupt servicing INTRTC by using the interrupt mask flag register, and the WAFG, RIFG, and RTCIF flags should be cleared after the rewrite.
3. The second count register (SEC), minute count register (MIN), hour count register (HOUR), week count register (WEEK), day count register (DAY), month count register (MONTH), and year count register (YEAR) may be read in any sequence. All the registers do not have to read and only some registers may be read.

7.4.4 Setting alarm of real-time clock

Set alarm time after setting 0 to WALE (alarm operation invalid) first.

Figure 7-6: Alarm setting steps

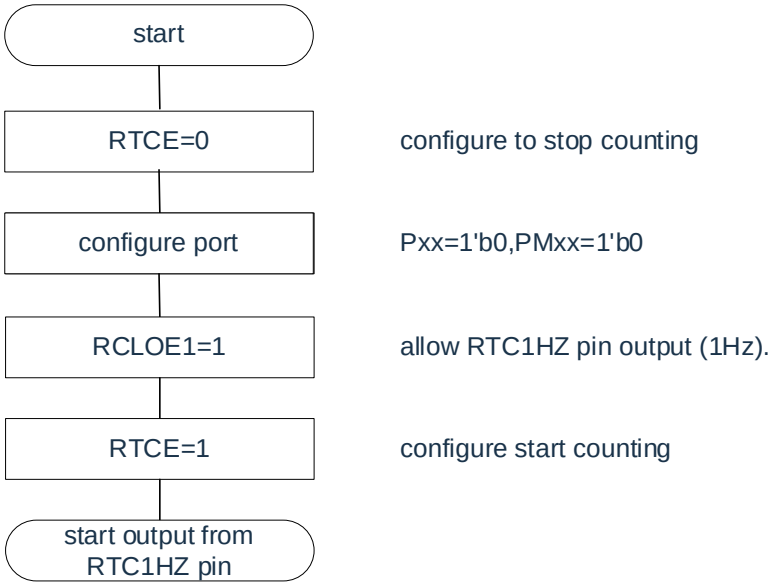


Notice:

1. There is no limit to the order of write operations for alarm minute registers (ALARMWM), alarm hour registers (ALARMWH), and alarm week registers (ALARMWW).
2. Fixed-cycle interrupts and alarm-consistent 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 fixed-cycle interrupt status flag (RIFG) and the alarm detection status flag (WAFG) upon INTRTC occurrence.

7.4.5 1Hz output of real-time clock

Figure 7-7: 1Hz output setting procedure



Notice: First set the RTCEN bit to 1, while oscillation of the count clock (F_{SUB}) is stable.

7.4.6 Example of clock error correction of real-time clock

The clock can be corrected with high accuracy when it is slow or fast, by setting a value to the clock error correction register.

Example of calculating the correction value

The correction value used when correcting the count value of the internal counter (16-bit) is calculated by using the following equation: If a correctable range is -4165.6 ppm or lower and 4165.6 ppm or higher, set 0 to DEV.

(When DEV=0)

Correction value[Note]=Number of correction counts in 1 minute÷3=(Oscillation frequency÷Target frequency-1)×32768×60÷3

(When DEV=1)

Correction value[Note]=Number of correction counts in 1 minute=(Oscillation frequency÷Target frequency-1) ×32768×60

Note: The correction value is the clock error correction value calculated by using bits 12 to 0 of the clock error correction register (SUBCUD).

(When F12=0) correction value ={(F11,F10,F9,F8,F7,F6,F5,F4,F3,F2,F1,F0)-1}×2

(When F12=0) correction value = -{(/F11,/F10,/F9,/F8,/F7,/F6,/F5,/F4,/F3,/F2,/F1,/F0)+1}×2

When (F12~F0)=(*,0,0,0,0,0,0,0,0,0,0,0,*), clock error correction is not performed. "*" is 0 or 1.

/F12~/F0 are bit-inverted values ("000000000011" when "111111111100").

Remark:

1. The correction value is 2,4,6,8,.....,8186, 8188 or -2,-4,-6,-8,.....,-8186,-8188.
2. The oscillation frequency is the value of the counting clock (FRTC), and can be calculated by the following equation: The output frequency of the RTC1HZ pin×32768 when the clock error correction register is set to its initial value (00H).
3. The target frequency is the frequency resulting after correction performed by using the clock error correction register.

Correction example

Example of correcting from 32767.4Hz to 32768Hz (32767.4Hz+18.3ppm)

[Measuring the oscillation frequency]

When the watch error correction register (SUBCUD) is the initial value ("0000H"), the oscillation frequency of each product is measured by outputting a signal of approximately 1Hz from the RTC1HZ pin[Note].

Note: For the setting of RTC1Hz output, please refer to “7.4.5 1Hz output of real-time clock”.

[Calculating the correction value]

(When the output frequency of the RTC1HZ pin is 0.9999817Hz)

Oscillation frequency = 32768×0.9999817≈32767.4Hz

Suppose the target frequency is 32768Hz (32767.4Hz+18.3ppm) and DEV=1. An equation for calculating the correction value when the DEV bit is "1" is applied.

Correction value=Number of correction counts in 1 minute=(Oscillation frequency÷Target frequency-1)×32768×60=(32767.4÷32768-1) ×32768×60= -36

[Calculating the values to be set to (F12~F0)]

(When the correction value= -36)

If the correction value is less than 0 (when faster), assume F12=1. Calculating is based on correction values (F11~F0).

-{(/F11~/F0)-1}×2= -36

(/F11~/F0) =17

(/F11~/F0) = (0,0,0,0,0,0,0,1,0,0,0,1)

(F11~F0) = (1,1,1,1,1,1,1,0,1,1,1,0)

Therefore, the correction from 32767.4Hz to 32768Hz (32767.4Hz +18.3ppm) is as follows:

If by DEV=1 and correction value=–36 (bit12~0 of the SUBCUD register:1,1,1,1,1,1,1,1,0,1,1,1,0) to set the correction register, you can correct it to 32768Hz (0ppm).

Chapter 8 15-Bit Interval Timer

8.1 Function of 15-bit interval timer

An interrupt (INTIT) is generated at any previously specified time interval. It can be utilized for wakeup from deep sleep mode.

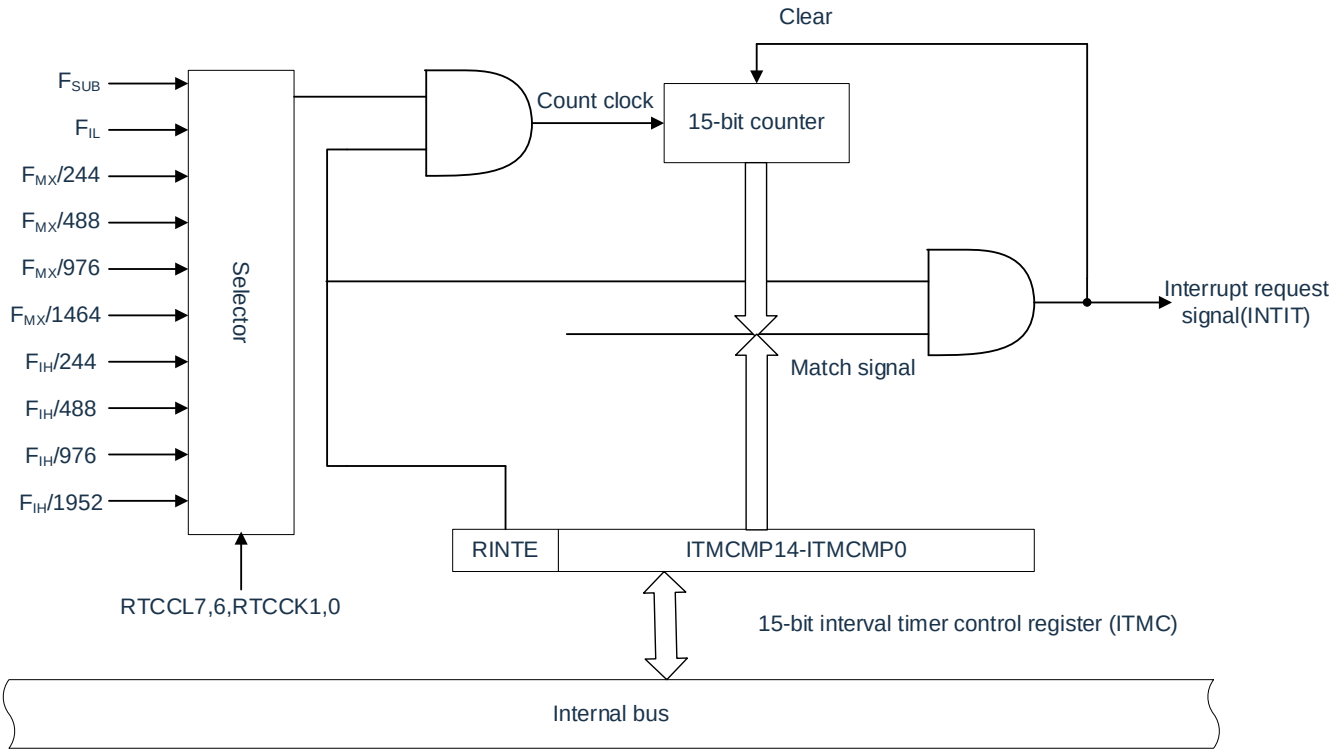
8.2 Structure of 15-bit interval timer

The 15-bit interval timer consists of the following hardware.

Table 8-1: Structure of 15-bit interval timer

Item	Structure
Counter	15-bit counter
Control register	Peripheral enable register 0 (PER0).
	Real-time clock selection register (RTCCL)
	15-bit interval timer control register (ITMC)

Figure 8-1: Block diagram of 15-bit interval timer



8.3 Registers for controlling 15-bit interval timer

The 15-bit interval timer is controlled by the following registers.

Peripheral enable register 0 (PER0).

Real-time clock selection register (RTCCL)

15-bit interval timer control register (ITMC)

8.3.1 Peripheral enable register 0 (PER0)

The PER0 register is a register that sets a clock that is allowed or prohibited to supply to each peripheral hardware. Reduce power consumption and noise by stopping clock supply to unused hardware.

When using a 15-bit interval timer, bit7 (RTCEN) must be set to "1". The PER0 register is set by an 8-bit memory manipulation instruction. After the reset signal is generated, the value of this register changes to "00H".

Table 8-2: Format of peripheral enable register 0 (PER0)

Address: 0x40020420	After reset: 00H		R/W				
Symbol	7	6	5	4	3	2	1	0
PER0	RTCEN	0	ADCEN	IICA0EN	SCI1EN	SCI0EN	TM41EN	TM40EN

RTCEN	Control of an input clock of a real-time clock (RTC) and a 15-bit interval timer
0	Stop to supply the input clock. The SFR used by the Real Time Clock (RTC) and the 15-bit interval timer cannot be written. The real-time clock (RTC) and 15-bit interval timer are in the reset state.
1	Supply the input clock. The SFR used by the Real Time Clock (RTC) and the 15-bit interval timer can be read and written.

8.3.2 Real-time clock selection register (RTCCL)

The real-time clock and the count clock (F_{RTC}) of the 15-bit interval timer can be selected via RTCCL.

Table 8-3: Format of real-time clock selection register (RTCCL)

Address: 0x4004047C | After reset: 00H | R/W

Symbol	7	6	5	4	3	2	1	0
RTCCL	RTCCL7	RTCCL6	RTCCL5	0	0	0	RTCCKS1	RTCCKS0

RTCCL7	Selection of clock source for real time clock, counting clock for 15-bit interval timer
0	Selects high-speed system clock (F_{MX})
1	Selects high-speed on-chip oscillator (F_{HOCO})

RTCCKS1	RTCCKS0	RTCCL6	RTCCL5	Selection of operating clocks for real-time clocks, counting clocks for 15-bit interval timer
0	0	x	x	Sub-system Clock (F_{SUB})
0	1			Low-speed internal oscillator clock (F_{IL}) (WUTMMCK0 must be set to 1).
1	0	0	1	Main clock F_{MAX}/F_{HOCO}(selected via RTCCL7)/1952
1	0	0	0	Main clock F_{MAX}/F_{HOCO}(selected via RTCCL7)/1464
1	0	1	0	Main clock F_{MAX}/F_{HOCO}(selected via RTCCL7)/976
1	1	0	0	Main clock F_{MAX}/F_{HOCO}(selected via RTCCL7)/488
1	1	1	0	Main clock F_{MAX}/F_{HOCO}(selected via RTCCL7)/244

8.3.3 15-bit interval timer control register (ITMC)

This register is used to set up the starting and stopping of the 15-bit interval timer operation and to specify the timer compare value.

The ITMC register is set by a 16-bit memory manipulation instruction.

After a reset signal is generated, the value of this register becomes "7FFFH".

Table 8-4: Format of 15-bit interval timer control register (ITMC)

Address: 0x40044F50 After reset: 7FFFH R/W

Symbol	15	14~0
ITMC	RINTE	ITCMP14～ITCMP0

RINTE	15-bit interval timer operation control
0	Count operation stopped (count clear)
1	Start operation of counters.

ITCMP14~ITCMP0	Specification of the 15-bit interval timer compare value
0001H	These bits generate “an interrupt at the fixed cycle, count clock cycles×(ITCMP set value+1)”.
• • •	
7FFFH	
0000H	Settings are disabled.
Example interrupt cycles when 001H or 7FFFH is specified for ITCMP14~ITCMP0 • ITCMP14~ITCMP0=0001H, count clock: F_{SUB}=32.768KHz 1/32.768[KHz]×(1+1)=0.06103515625[ms]≈61.03[us] • ITCMP14~ITCMP0=7FFFH, count clock: F_{SUB}=32.768KHz 1/32.768[KHz]×(32767+1)=1000[ms]	

Notice:

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. After shifting from sleep mode to normal operation mode, if you want to set the ITMC register and enter sleep mode again, you must confirm that the write value of the ITMC register is reflected or set the ITMC registers or wait that more than one clock of the count clock has elapsed before shifting to sleep mode.
4. Only change the setting of the ITCMP14 to ITCMP0 bits when RINTE = 0. However, it is possible to change the settings of the ITCMP14 to ITCMP0 bits at the same time as when changing RINTE from “0” to “1” or “1” to “0”.

8.4 Operation of 15-bit interval timer

8.4.1 Operation timing of 15-bit interval timer

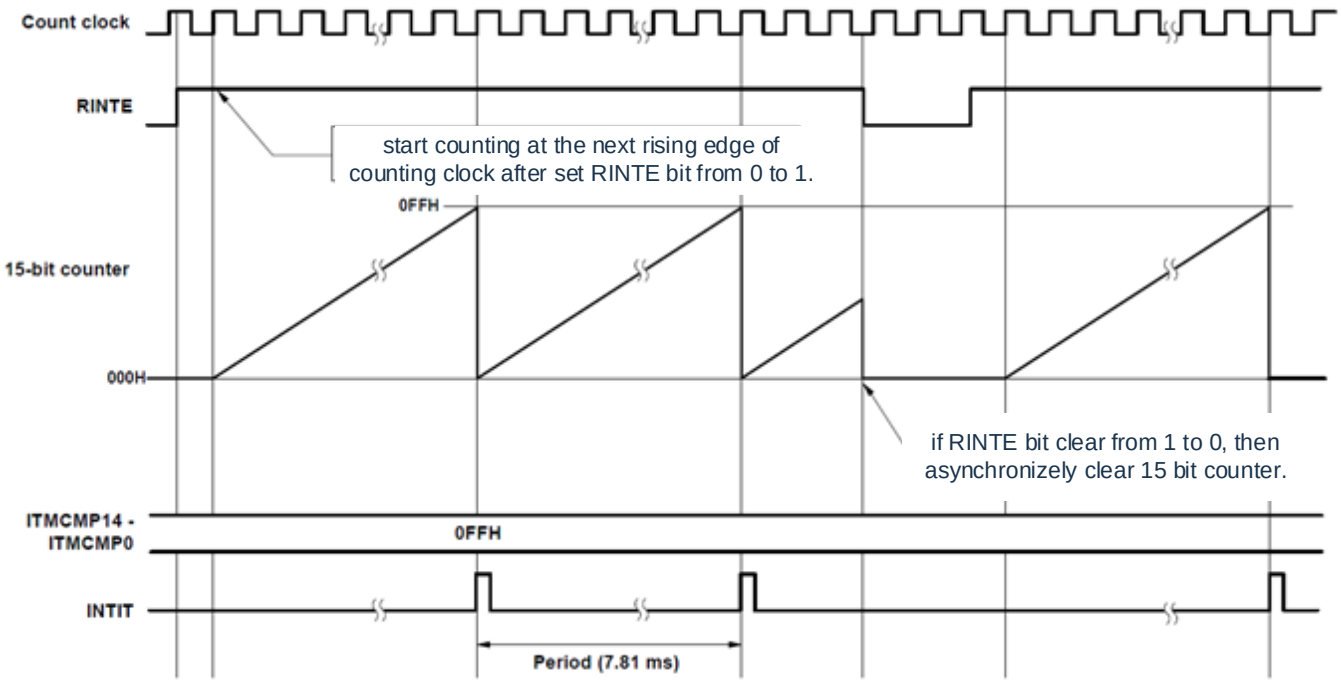
The count value specified for the ITCMP14 to ITCMP0 bits is used as an interval to operate a 15-bit interval timer that repeatedly generates interrupt requests (INTIT). When the RINTE bit is set to “1”, the 15-bit counter starts counting.

When the 15-bit counter value matches the value specified for the ITCMP14 to ITCMP0 bits, the 15-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 15-bit interval timer is as follows Figure 8-2.

Figure 8-2: Operation timing of 15-bit interval timer

(ITCMP14~ITCMP0=0FFH, count clock: F_{SUB}=32.768KHz)



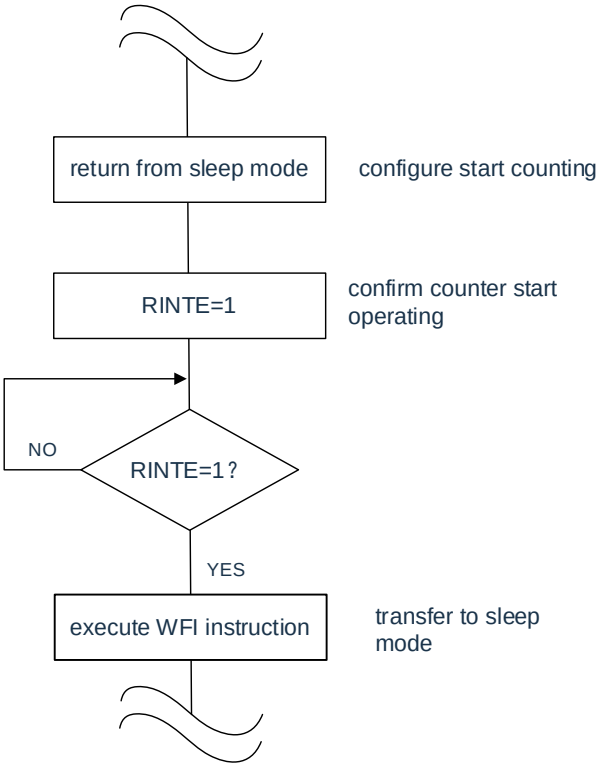
8.4.2 Start of count operation and re-enter to sleep mode after returned from sleep mode

When setting the RINTE bit after returned from sleep mode and entering sleep 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 sleep mode.

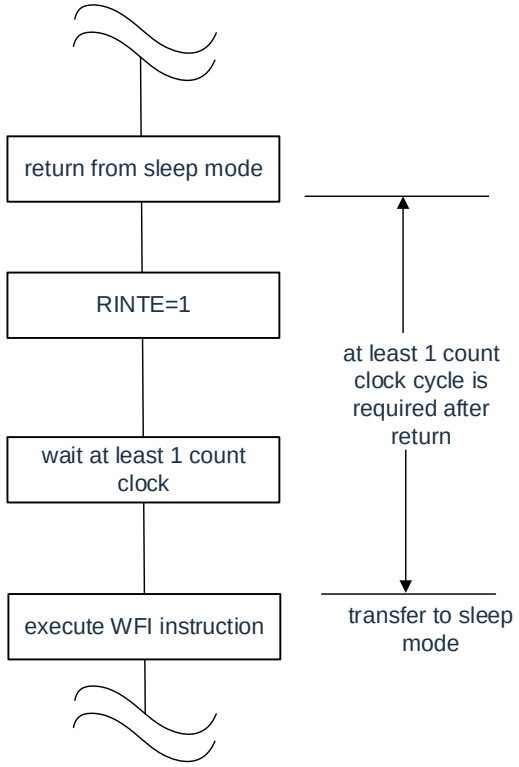
After setting RINTE to 1, confirm by polling that the RINTE bit has become 1, and then enter sleep mode (see Example 1 in the figure below).

After setting RINTE to 1, wait for at least one cycle of the count clock and then enter sleep mode (see Example 2 in the figure below).

Example 1



Example 2



Chapter 9 Clock Output/Buzzer Output Controller

9.1 Function of clock output/buzzer output controller

Clock output is the function of outputting the clock provided to the peripheral IC, and buzzer output is the function of outputting the buzzer frequency square wave.

This product has two clock output/buzzer output pins that can be selected to be used as clock output or buzzer output from any pin other than RESETB.

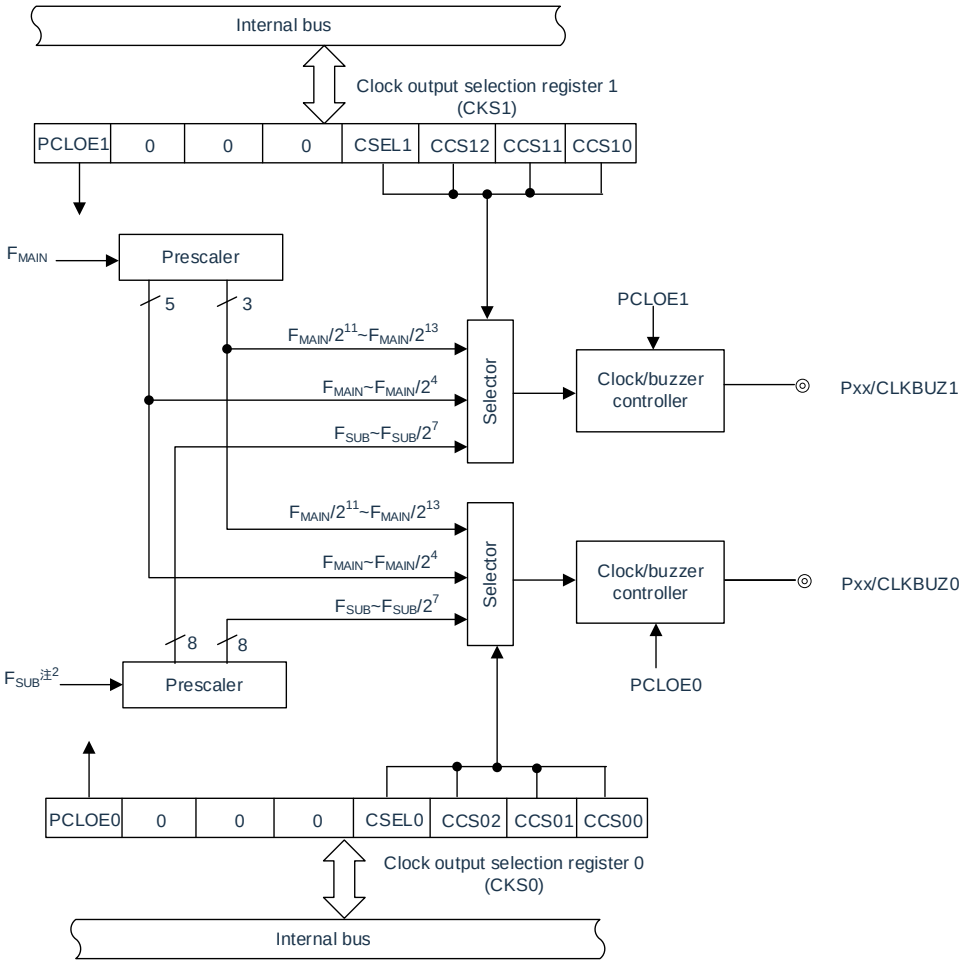
The CLKBUZn pin outputs the clock selected by the clock output selection register n (CKSn).

The block diagram of the clock output/buzzer output controller is shown in Figure 9-1.

Notice: The subsystem clock (F_{SUB}) cannot be output from the CLKBUZn pin when the RTCLPC bit of the subsystem clock supply mode control register (OSMC) is "1" and in the SLEEP mode in which the CPU is running with the subsystem clock (F_{SUB}).

Remark: n=0, 1.

Figure 9-1: Block diagram of clock output/buzzer output controller



Remark: For the frequencies that can be output from CLKBUZ0 and CLKBUZ1 pins, please refer to "AC Characteristics" in the data sheet.

9.2 Structure of clock output/buzzer output controller

The clock output/buzzer output controller consists of the following hardware.

Table 9-1: Registers for controlling clock output/buzzer output controller

Item	Structure
Control registers	Clock output select registers n (CKSn) Port mode control register (PMCxx), Port mode register (PMxx), Port multiplexing control register (PxxCFG)

9.3 Registers for controlling clock output/buzzer output controller

9.3.1 Clock output select register n (CKSn)

These registers set output enable/disable for clock output or for the buzzer frequency output pin (CLKBUZn), and set the output clock.

Select the clock to be output from the CLKBUZn pin by using the CKSn register. The CKSn register is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "00H".

Table 9-2: Format of clock output select register n (CKSn)

Address: 0x40040FA5 (CKS0), 0x40040FA6 (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	CLKBUZn pin output enable/disable specification
0	Output disable (default)
1	Output enable

CSELn	CCSn2	CCSn1	CCSn0	CLKBUZn pin output clock selection
0	0	0	0	F_{MAIN}
0	0	0	1	$F_{MAIN}/2$
0	0	1	0	$F_{MAIN}/2^2$
0	0	1	1	$F_{MAIN}/2^3$
0	1	0	0	$F_{MAIN}/2^4$
0	1	0	1	$F_{MAIN}/2^{11}$
0	1	1	0	$F_{MAIN}/2^{12}$
0	1	1	1	$F_{MAIN}/2^{13}$
1	0	0	0	F_{SUB}
1	0	0	1	$F_{SUB}/2$
1	0	1	0	$F_{SUB}/2^2$
1	0	1	1	$F_{SUB}/2^3$
1	1	0	0	$F_{SUB}/2^4$
1	1	0	1	$F_{SUB}/2^5$
1	1	1	0	$F_{SUB}/2^6$
1	1	1	1	$F_{SUB}/2^7$

Note: Use the output clock within a range of 16 MHz. For details, please refer to "AC Characteristics" in the data sheet.

Notice:

1. Change the output clock after disabling clock output (PCLOEn = 0).
2. To shift to deep sleep mode when the main system clock is selected (CSELn = 0), set PCLOEn = 0 before executing the WFI instruction. When the subsystem clock is selected (CSELn = 1), PCLOEn = 1 can be set because the clock can be output while the RTCLPC bit of the subsystem clock supply mode control (OSMC) register is set to 0 and moreover while deep sleep mode is set.
3. It is not possible to output the subsystem clock (F_{SUB}) from the CLKBUZn pin while the RTCLPC bit of the subsystem clock supply mode control register (OSMC) is set to 1 and moreover while

sleep mode is set with the subsystem clock (F_{SUB}) selected as CPU clock.

Remark:

1. n=0, 1
2. F_{MAIN}: Main system clock frequency.
   F_{SUB} : Subsystem clock frequency.

9.3.2 Registers for controlling clock output/buzzer output port functions

This product can multiplex the clock output/buzzer output function CLKBUZ0/CLKBUZ1 to any port except RESETB. To use the clock output/buzzer output function, the port multiplexing function configuration register (PxxCFG), port register (Pxx), port mode register (PMxx), and port mode control register (PMCxx) must be set. For details, refer to "Chapter 2 Port Function".

A multiplexed port configured as a clock output/buzzer output pin must have its corresponding Port Register (Pxx), Port Mode Register (PMxx) bits and Port Mode Control Register (PMCxx) bits set to "0".

(Example) Using P20 as clock output/buzzer output (CLKBUZ0).

(1) Set the bit P20 of port register 2 to "0".
(2) Set the PM20 bit of port mode register 2 to "0".
(3) Set the PMC20 bit of port mode control register 2 to "0".
(4) Set "0x18" to port multiplexing function configuration register P20CFG.

(Example) Using P15 as clock output/buzzer output (CLKBUZ1).

(1) Set the bit P15 of the port register 1 to "0".
(2) Set the PM15 bit of port mode register 1 to "0".
(3) Set "0" to PMC15 bit of port mode control register 1.
(4) Set "0x19" to port multiplexing function configuration register P15CFG.

9.4 Operation of clock output/buzzer output controller

One pin can be used as clock output or buzzer output.

The CLKBUZ0 pin outputs a clock/buzzer selected by the clock output select register 0 (CKS0).

The CLKBUZ1 pin outputs a clock/buzzer selected by the clock output select register 1 (CKS1).

9.4.1 Operation as output pin

The CLKBUZn pin is output as the following procedure.

① Set the bit of the port register (Pxx), port mode register (PMxx) and port mode control register (PMCxx) corresponding to the port used as CLKBUZn pin to "0". Set the port multiplexing function configuration register (PxxCFG).

② Select the output frequency with bits 0 to 3 (CCSn0 to CCSn2, CSELn) of the clock output select register (CKSn) of the CLKBUZn pin (output in disabled status).

③ Set bit 7 (PCLOEn) of the CKSn register to 1 to enable clock/buzzer output.

Remark:

1. The controller used as clock output starts or stops the clock output after 1 clock after the clock output (PCLOEn bit) is enabled or disabled. At this time, pulses with a narrow width are not output. Figure 9-2 shows enabling or stopping output using the PCLOEn bit and the timing of outputting the clock.
2. n=0, 1.

Figure 9-2: CLKBUZn pin output clock timing

9.5 Cautions for clock output/buzzer output controller

When the main system clock is selected for the CLKBUZn output (CSE=0), if deep sleep mode is entered within 1.5 clock cycles output from the CLKBUZn pin after the output is disabled (PCLOEn=0), the CLKBUZn output width becomes narrower.

Chapter 10 Watch Dog Timer

10.1 Function 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 internal oscillator clock (F_{IL}).

The watchdog timer is used to detect an inadvertent program loop. An internal reset signal is generated when a program loop is detected.

Program loop is detected in the following cases.

(1) If the watchdog timer counter overflows

(2) If data other than “ACH” is written to the WDTE register

(3) 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, please refer to "Chapter 19 Reset Function". When 75%+1/2F_{IL} of the overflow time is reached, an interval interrupt can be generated.

10.2 Structure of watch dog timer

The watchdog timer includes the following hardware.

Table 10-1: Structure of watch dog timer

Item	Structure
Counter	Internal counter (17-bit)
Control register	Watchdog timer enable register (WDTE)

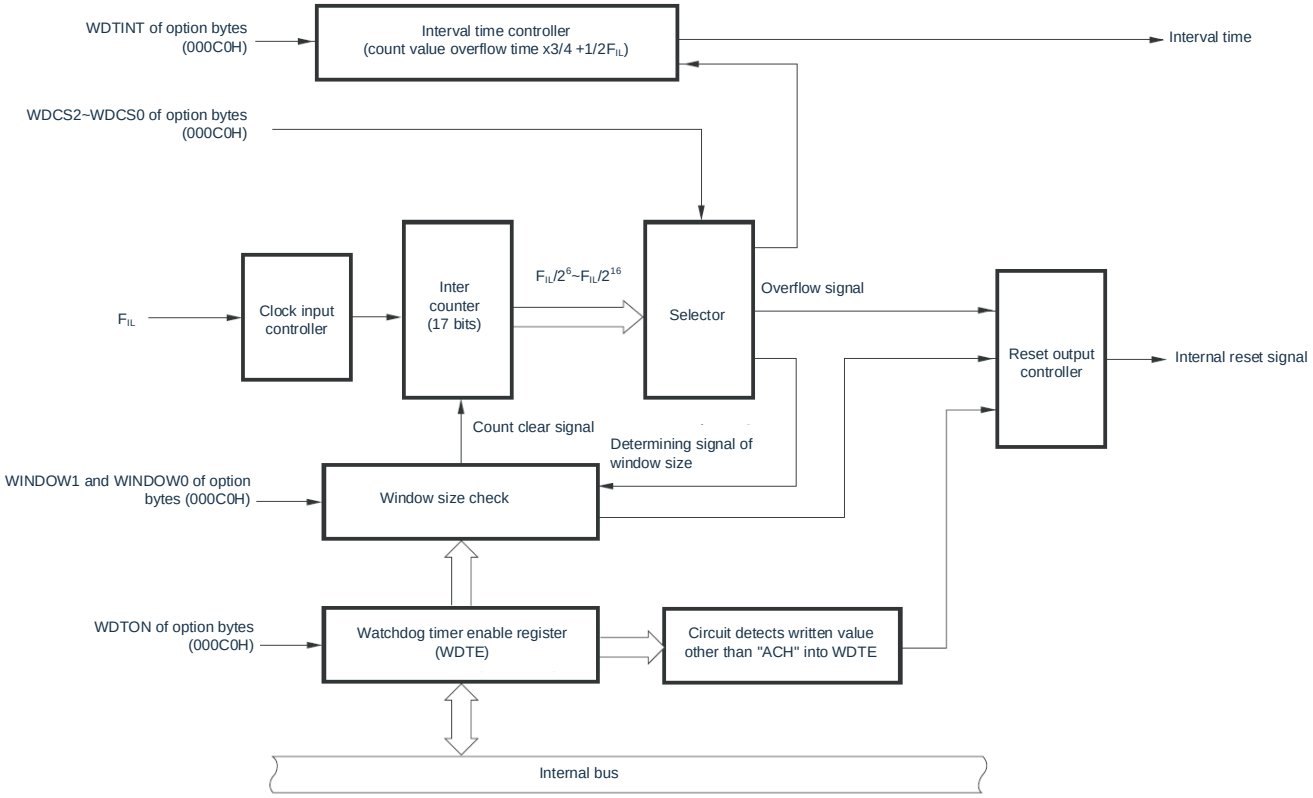
The operation of the counter is controlled by means of option bytes as well as setting the overflow time, the window opening period and the interval interruption.

Table 10-2: Setting of option bytes and watchdog timer

Setting of watchdog timer	Option byte (000C0H)
Watchdog timer interval interrupt	bit7 (WDTINT)
Setting of window open period	bit6 and bit5 (WINDOW1, WINDOW0)
Watchdog timer counter operation control	bit4 (WDTON)
Overflow time of watchdog timer	bit3~1 (WDCS2~WDCS0)
Watchdog timer counter operation control (in sleep mode)	bit0 (WDSTBYON)

Remark: For the option byte, see “Chapter 24 Option Byte”.

Figure 10-1: Block diagram of watchdog timer



Remark: F_{IL}: Low-speed on-chip oscillator clock frequency

10.3 Registers for controlling watchdog timer

The watchdog timer is controlled by the watchdog timer enable register (WDTE).

10.3.1 Watchdog timer enable register (WDTE)

Writing “ACH” to the WDTE register clears the watchdog timer counter and starts counting again. The WDTE register is set by an 8-bit memory manipulation instruction. Reset signal generation sets this register to “9AH” or “1AH” Note.

Table 10-3: Format of watchdog timer enable register (WDTE)

Address: 0x40021001 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

Remark:

1. If a value other than “ACH” is written to the WDTE register, an internal reset signal is generated.
2. The value read from the WDTE register is 9AH/1AH (this differs from the written value (“ACH”)).

10.3.2 LOCKUP control register (LOCKCTL) and its protection register (PRCR)

The LOCKCTL register is a configuration register for controlling the Cortex-M0+ LockUp function to operate the watchdog timer, and PRCR is its write-protect register.

The LOCKCTL, PRCR registers are set by an 8-bit memory manipulation instruction.

After generating a reset signal, the values of the LOCKCTL, PRCR register change to "00H".

Table 10-4: Format of LOCKUP control register (LOCKCTL) and its protection register (PRCR) (1/2)

Address: 40020405H | After reset: 01H | R/W

Symbol	7	6	5	4	3	2	1	0
LOCKCTL	0	0	0	0	0	0	0	lockup_rst

lockup_rst	Configuration of LOCKUP function
0	• LOCKUP does not cause a WDT reset
1	• LOCKUP causes the WDT to reset

Table 10-4: Format of LOCKUP control register (LOCKCTL) and its protection register (PRCR) (2/2)

Address: 40020406H | After reset: 00H | R/W

Symbol	7	6	5	4	3	2	1	0
PRCR	PRTKEY[7:1]							PRCR

PRCR	Write protection of LOCKUP control register
0	• LOCKCTL register is not writable
1	• LOCKCTL register is writable

PRTKEY[7:1]	Write protection of PRCR
78H	• PRCR is writable
Other	• PRCR is not writable

10.4 Operation of watchdog timer

10.4.1 Operational control of watchdog timer

1. When using the watchdog timer, set the following items by option byte (000C0H):
   (1) The bit 4 (WDTON) of the option byte (000C0H) must be set to "1" to enable the watchdog timer count to operate (the counter starts operating after the reset is released) (refer to Chapter 24 Option Byte for details).

WDTON	Counter of watchdog timer
0	Disables counting operation (stop counting after reset released)
1	Enables counting operation (start counting after release reset)

   (2) The overflow time must be set by bit3~1 (WDCS2~WDCS0) of the option byte (000C0H) (refer to 10.4.2 and Chapter 24 Option Byte for details).
   (3) The window opening period must be set by bit6 and bit5 (WINDOW1, WINDOW0) of the option byte (000C0H) (refer to 10.4.2 and Chapter 24 Option Byte for details).
2. After the reset is released, the watchdog timer starts counting.
3. After starting counting and before the overflow time set by the option byte, writing "ACH" to the watchdog timer enable register (WDTE) clears the watchdog timer and starts counting again.
4. Thereafter, writes to WDTE registers after the second time after the reset must be performed while the window is open. If you write the WDTE register while the window is closed, an internal reset signal is generated.
5. If you do not write "ACH" to the WDTE register and exceed the overflow time, an internal reset signal is generated. An internal reset signal is generated if:
   (1) If data other than “ACH” is written to the WDTE register

Remark:

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 F_{IL} clocks may occur before the watchdog timer is cleared.
3. The watchdog timer can be cleared immediately before the count value overflows.
4. As shown below, the watchdog timer operates in sleep or deep sleep mode depending on the set value of bit0 (WDSTBYON) of the option byte (000C0H).

	WDSTBYON=0	WDSTBYON=1
Sleep mode	Stop operation of watchdog timer.	Continue operation of watchdog timer.
Deep sleep mode		

5. When the WDSTBYON bit is "0", restart the watchdog timer count after the sleep or deep sleep mode released. At this point, the counter is cleared to "0" and the count begins.
6. When operating with the X1 oscillation clock after releasing the deep sleep mode, the CPU starts operating after the oscillation stabilization time has elapsed.
7. Therefore, if the period between the deep sleep 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 deep sleep mode release by an interval interrupt.

10.4.2 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 10-5: Setting of overflow time of watchdog timer

WDCS2	WDCS1	WDCS0	Overflow time of watchdog timer (When F_{IL}=20KHz(MAX.))
0	0	0	$2^6/F_{IL}$ (3.2ms)
0	0	1	$2^7/F_{IL}$ (6.4ms)
0	1	0	$2^8/F_{IL}$ (12.8ms)
0	1	1	$2^9/F_{IL}$ (25.6ms)
1	0	0	$2^{11}/F_{IL}$ (102.4ms)
1	0	1	$2^{13}/F_{IL}$ (409.6ms)
1	1	0	$2^{14}/F_{IL}$ (819.2ms)
1	1	1	$2^{16}/F_{IL}$ (3276.8ms)

Remark: F_{IL}: Low-speed on-chip oscillator clock frequency

10.4.3 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:

(1) 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.

(2) 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.

Remark: 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 10-6: Setting window open period of watchdog timer

WINDOW1	WINDOW0	Window open period of watchdog timer
0	-	Settings are disabled.
1	0	75%
1	1	100%

Remark:

1. 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.
2. 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	
	75%	100%
Window close time	0~12.8ms	None
Window open time	12.8~25.6ms	0~25.6ms

＜When window open period is 50%＞

(1) Overflow time:

$2^9/F_{IL}$ (Max.) =2^9/20KHz (Max.) =25.6ms

(2) Window close time:

0~$2^9/F_{IL}$ (Min.) × (1-0.75) =0~2^9/10KHz×0.25=0~12.8ms

(3) Window open time:

$2^9/F_{IL}$ (Min.) × (1-0.75) ~$2^9/F_{IL}$ (Max.) =12.8~25.6ms

10.4.4 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%+1/2F_{IL} of the overflow time is reached.

Table 10-7: 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%+1/2F_{IL} of the overflow time is reached.

Remark:

1. When operating with the X1 oscillation clock after releasing the deep sleep mode, the CPU starts operating after the oscillation stabilization time has elapsed. Therefore, if the period between the deep sleep 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 deep sleep mode release by an interval interrupt.
2. 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.

10.4.5 Operation of watchdog timer during LOCKUP

When lockup_rst bit of the lockup control register lockcTL is set to 1, once the kernel enters the LOCKUP state, the low-speed internal oscillator begins to oscillate, the watchdog timer automatically starts operating, and the overflow time control bit (WDCS2~WDCS0) is set to 3'b010, that is, the overflow time is set to 12.8ms.

Chapter 11 A/D Converter

The number of analog input channels of the A/D converter varies from product to product; refer to the corresponding product datasheet for detailed pinout.

11.1 Function of A/D converter

The A/D converter is used to convert analog input signals into digital values, including the following functions.

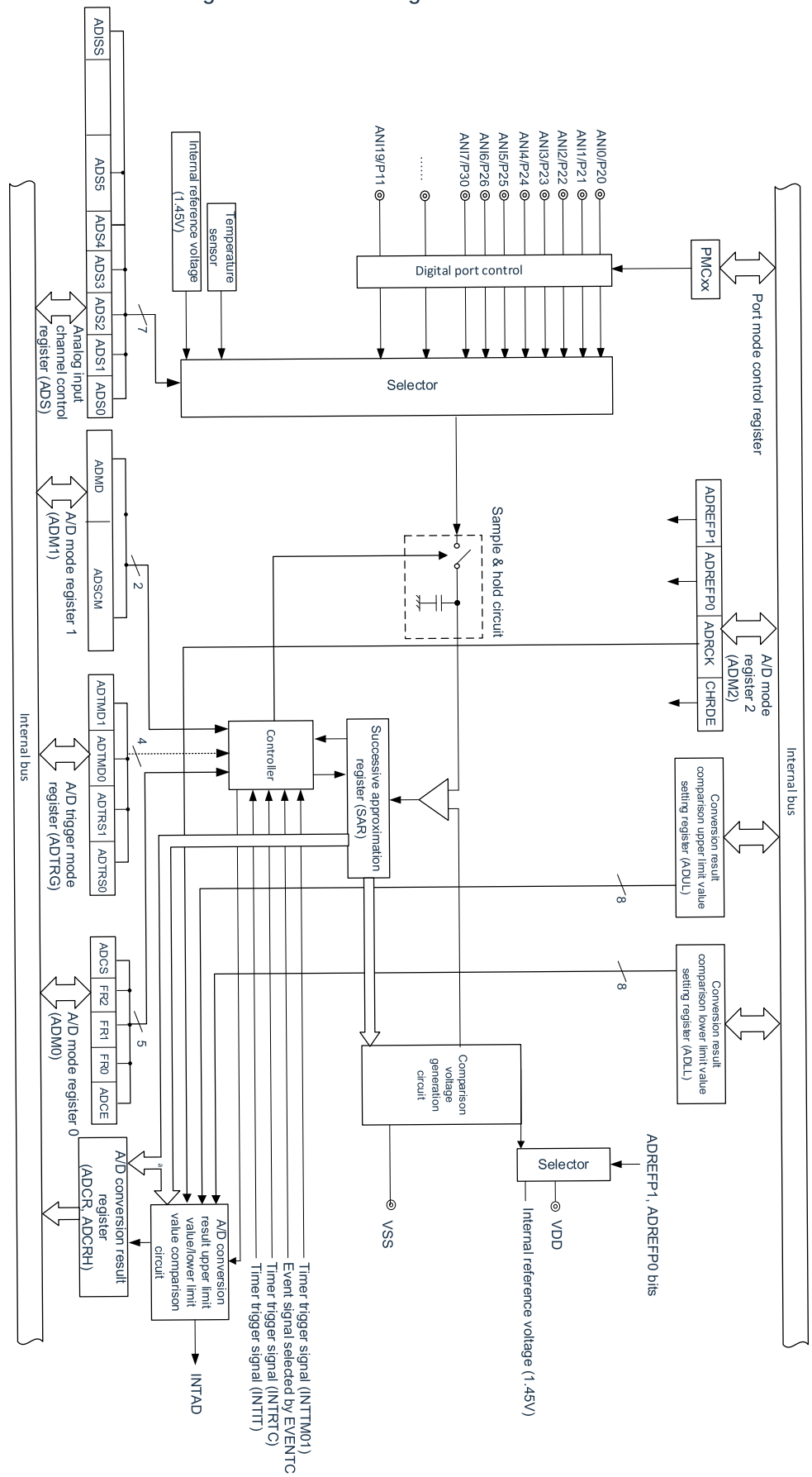
(1) 12-bit resolution A/D conversion

(2) Select 1 channel of analog input from ANI0~ANI19, V internal reference voltage and temperature sensor, and repeat the A/D conversion with 12-bit resolution. An interrupt request (INTAD) is generated for each end of 1 A/D conversion (in the case of the selection mode).

Various A/D conversion modes are set by the combination of modes as described below.

Trigger mode	Software trigger	Start the conversion with software operations.
	Hardware trigger no-wait mode	Conversion is started by detecting a hardware trigger.
	Hardware trigger wait mode	In the conversion standby state where the A/D power supply is cut off, the power supply is turned on by detecting a hardware trigger, and the conversion starts automatically after the A/D power supply stabilization waiting time.
Channel selection mode	Select mode	Select 1 channel of analog input for A/D conversion.
	Scan mode	Performs A/D conversion of 4 channels of analog input in sequence. Can select 4 consecutive channels from ANI0 to ANI15 as analog inputs.
Conversion mode	Single conversion mode	Performs 1 A/D conversion for the selected channel.
	Consecutive conversion mode	Performs continuous A/D conversion for the selected channel until stopped by the software.
Sampling time	Samle clock 4/8 ADCLKs	The sampling time can be selected via the ADSMPWAIT register, which uses 4 conversion clocks (F_{AD}) by default.

Figure 11-1: Block diagram of A/D converter



Remark: For the selection of analog input channel ANIx, please refer to11.2.6Analog input channel specification register (ADS).

11.2 Registers for controlling A/D converter

The A/D converter is controlled by the following registers:

Register base address: CSC_BASE=4002_0420H;ADC_BASE=4004_5000H;PORT_BASE=4004_0000H

Register name	Register description	R/W	Reset value	Register address
PER0	Peripheral enable register 0	R/W	00H	CSC_BASE+20H
ADM0	A/D converter mode register 0	R/W	00H	ADC_BASE+00H
ADM1	A/D converter mode register 1	R/W	00H	ADC_BASE+02H
ADM2	A/D converter mode register 2	R/W	00H	ADC_BASE+04H
ADTRG	A/D converter trigger mode register	R/W	00H	ADC_BASE+06H
ADS	Analog input channel specification register	R/W	00H	ADC_BASE+08H
ADLL	Conversion result comparison lower limit setting register	R/W	00H	ADC_BASE+0AH
ADUL	Conversion result comparison upper limit setting register	R/W	00H	ADC_BASE+0BH
ADCR	12-bit A/D conversion result register	R	0000H	ADC_BASE+0EH
ADCRH	8-bit A/D conversion result register	R	00H	ADC_BASE+0FH
ADSMPWAIT	A/D converter sampling time extension control register	R/W	00H	ADC_BASE+15H
PMCn	Port mode control register	R/W	Note 1	PORT_BASE+Note1

R: read only, W: write only, R/W: both read and write

Remark: When selecting a channel by the ADS register, you need to configure the PMC register of the channel pin as an analog channel.

11.2.1 Peripheral enable register 0 (PER0)

The PER0 register is the register that sets whether to enable or disable the supply of clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocks to hardware that is not in use.

To use the A/D converter, bit5 (ADCEN) must be set to "1".

The PER0 register is set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register becomes "00H".

Table 11-1: Table of peripheral enable register 0 (PER0)

Reset value: 00H R/W

	7	6	5	4	3	2	1	0
PER0	RTCEN	IRDAEN	ADCEN	IICA0EN	SCI1EN	SCI0EN	TM41EN	TM40EN

ADCEN	Control of the input clock for the A/D converter
0	Stop to supply the input clock. • The SFR used by the A/D converter cannot be written. • The A/D converter is in the reset state.
1	Supply the input clock. • The SFR used by the A/D converter can be read and written.

Remark: 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 control register (PMCxx)).

11.2.2 A/D converter mode register 0 (ADM0)

This register sets the clock for A/D converter, and starts/stops conversion. The ADM0 register is set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register becomes "00H".

Table 11-2: Format of A/D converter mode register 0 (ADM0)

Reset value: 00H R/W

	7	6	5	4	3	2	1	0
ADM0	ADCS	0	FR2	FR1	FR0	0	0	ADCE

ADCS	A/D conversion operation control
0	Stops conversion [When read] Stop conversion/standby status
1	Enables conversion [When read] While in the software trigger mode: Conversion operation status While in the hardware trigger wait mode: A/D power supply stabilization wait status+conversion operation status

ADCE	A/D voltage comparator operation control
0	Stops A/D voltage comparator operation.
1	Enables A/D voltage comparator operation.

Remark:

1. For details of the FR2~FR0 bits and A/D conversion, please refer to “Table 11-5 A/D Conversion Time Selection”.
2. It takes 2 μs from the start of operation for the operation to stabilize. While in the software trigger mode or hardware trigger no-wait mode, when the ADCS bit is set to 1 after 2 μs or more has elapsed from the time ADCE bit is set to 1, the conversion result is valid. Otherwise, ignore data of the conversion. In hardware-triggered wait mode, a wait time of 2us is guaranteed by design.

Notice:

1. Change the FR2~FR0 bits while conversion is stopped (ADCS = 0).
2. Do not set the ADCS bit to 1 and the ADCE bit to 0.
3. Do not change the ADCS and ADCE bits from 0 to 1 by using an 8-bit manipulation instruction. You must follow the procedure in “11.5 A/D converter setup flowchart”.

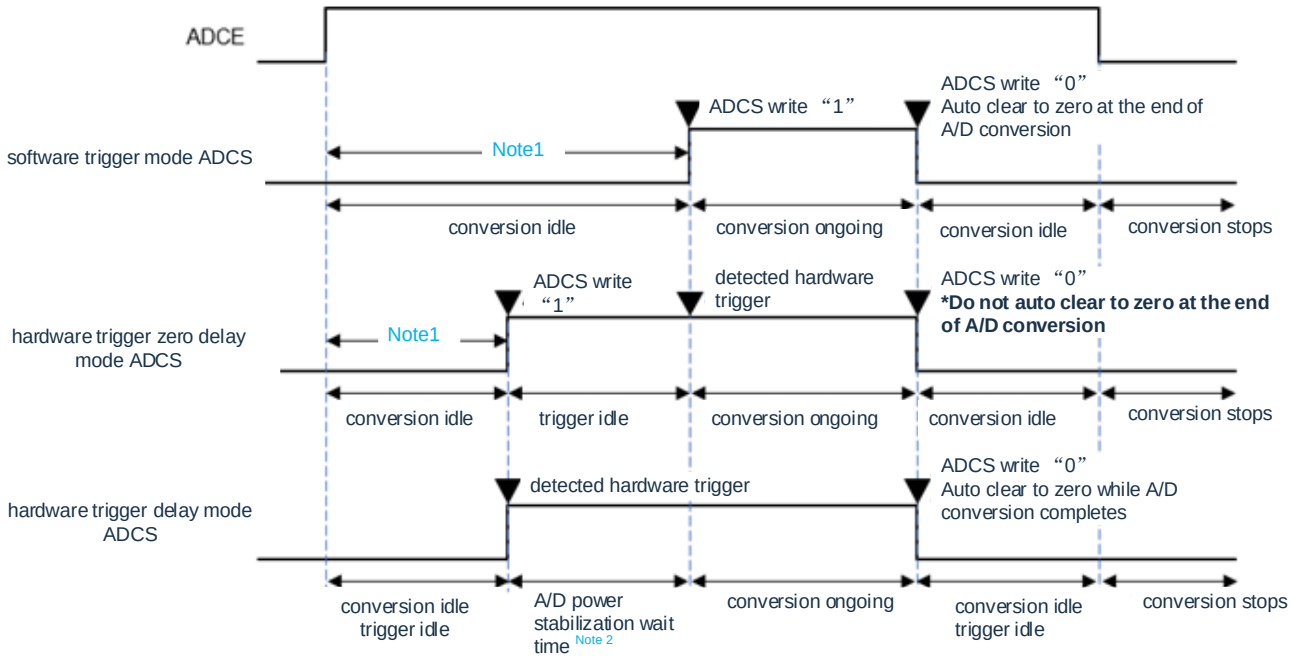
Table 11-3: ADCS bit and ADCE bit settings

ADCS	ADCE	A/D conversion operation
0	0	Conversion stop state
0	1	Conversion standby state
1	0	Prohibit settings.
1	1	Conversion operating state

Table 11-4: ADCS bit set and clear conditions

A/D conversion mode			Set conditions	Clear conditions
Software trigger	Select mode	Continuous conversion mode	When write “1” to the ADCS bit	When writing “0” to the ADCS bit
		Single conversion mode		• When writing “0” to the ADCS bit • Automatically cleares to “0” at the end of A/D conversion.
	Scan mode	Continuous conversion mode		When writing “0” to the ADCS bit
		Single conversion mode		• When writing “0” to the ADCS bit • Automatically clears to "0" when the set 4-channel conversion is completed.
Hardware trigger no-wait mode	Select mode	Continuous conversion mode	When write “1” to the ADCS bit	When writing “0” to the ADCS bit
		Single conversion mode		• When writing “0” to the ADCS bit
	Scan mode	Continuous conversion mode		When writing “0” to the ADCS bit
		Single conversion mode		• When writing “0” to the ADCS bit
Hardware trigger wait mode	Select mode	Continuous conversion mode	When triggered by input hardware	When writing “0” to the ADCS bit
		Single conversion mode		• When writing “0” to the ADCS bit • Automatically clears to “0” at the end of A/D conversion.
	Scan mode	Continuous conversion mode		When writing “0” to the ADCS bit
		Single conversion mode		• When writing “0” to the ADCS bit • Automatically clears to "0" when the set 4-channel conversion is completed.

Figure 11-2: Diagram of using various modes of A/D



Note 1: In software trigger mode or hardware trigger no-wait mode, it takes at least 2us (TBD) to rise from the ADCE bit to the ADCS bit to stabilize the internal circuitry.

Note 2: In hardware trigger wait mode, the A/D power supply stabilization time of 1us is guaranteed by design.

Notice:

1. To use the hardware trigger wait mode, setting the ADCS bit to "1" is prohibited (it is automatically switched to "1" when a hardware trigger signal is detected). However, the ADCS bit can be set to "0" in order to set the standby state for A/D conversion.
2. The ADCE bit must be overridden when the ADCS bit is “0” (Stop Transition/Transition Standby).
3. In order to end the A/D conversion, the hardware trigger interval must be set at least to the following time:

   Hardware trigger no-wait mode: 2 F_{CLK} clocks +A/D conversion time

   Hardware trigger wait mode: 2 F_{CLK} clocks +A/D power supply stable wait time+A/D conversion time

   (F_{CLK}: CPU/peripheral hardware clock frequency)

Table 11-5: A/D conversion time (1/2)

(1) No A/D power stabilization wait time (software trigger mode/hardware trigger no wait mode)

A/D converter mode register 0 (ADM0)			A/D sampling time extension register (ADSMPWAIT)	Frequency of conversion clock ADCLK (F_{AD})	12-bit resolution conversion time	
					ADC conversion time = (number of sample clocks + number of successive comparison clocks)/ F_{AD}	
FR2	FR1	FR0	ADSMPWAIT		Number of ADC conversion clocks	ADC conversion time
0	0	0	0	F_{CLK}/32	16 ADCLK (4 sample clocks +12 successive comparison clocks).	16/F_{AD}
0	0	1		F_{CLK}/16		-
0	1	0		F_{CLK}/8		
0	1	1		F_{CLK}/4		
1	0	0		F_{CLK}/2		
1	0	1		F_{CLK}/1		
0	0	0	1	F_{CLK}/32	16 ADCLK (8 sample clocks +12 successive comparison clocks).	20/F_{AD}
0	0	1		F_{CLK}/16		-
0	1	0		F_{CLK}/8		
0	1	1		F_{CLK}/4		
1	0	0		F_{CLK}/2		
1	0	1		F_{CLK}/1		

Notice: To override the FR2~FR0 bits and ADSMPWAIT bits into different data, it must be done in the conversion stop state (ADCS=0).

Remark: Time required to perform an ADC conversion = (number of sample clocks + number of successive comparison clocks)/F_{AD}, where the number of sample clocks can be adjusted via the ADSMPWAIT register, and the default is 4 ADCLKs.The fastest clock supported by ADCLK is 8MHz.

F_{CLK}: CPU/peripheral hardware clock frequency

F_{AD}: The fastest ADC conversion clock frequency is 8MHz.

Table 11-6: A/D conversion time (2/2)

(2) With A/D power stabilization wait time (hardware trigger wait mode)

A/D converter mode register 0 (ADM0)			A/D sampling time extension register (ADSMPWAIT)	Frequency of conversion clock ADCLK (F_{AD})	A/D power supply stabilization time	Number of ADC conversion clocks	A/D power supply stabilization time +ADC conversion time
FR2	FR1	FR0	ADSMPWAIT				
0	0	0	0	F_{CLK}/32	2us	16 ADCLK (4 sample clocks + 12 successive comparison clocks)	2us +16/F_{AD}
0	0	1		F_{CLK}/16			
0	1	0		F_{CLK}/8			
0	1	1		F_{CLK}/4			
1	0	0		F_{CLK}/2			
1	0	1		F_{CLK}/1			
0	0	0	1	F_{CLK}/32	2us	20 ADCLK (8 sample clocks + 16 successive comparison clocks)	2us +20/F_{AD}
0	0	1		F_{CLK}/16			
0	1	0		F_{CLK}/8			
0	1	1		F_{CLK}/4			
1	0	0		F_{CLK}/2			
1	0	1		F_{CLK}/1			

Remark:

1. In hardware trigger wait mode, the power supply settling time is guaranteed by the hardware design and does not need to be set. And in continuous conversion mode, the A/D power stabilization wait time occurs only after the hardware trigger is detected for the first time.
2. Time required for ADC conversion after hardware triggering = 2us + (number of sample clocks + number of successive comparison clocks)/F_{AD}. The number of sample clocks can be adjusted via the ADSMPWAIT register, which defaults to four ADCLK. The fastest clock supported by ADCLK is 8MHz.
3. To override the FR2~FR0 bits and ADSMPWAIT bits into different data, it must be done in the conversion stop state (ADCS=0).
4. The conversion time in the hardware trigger wait mode includes the A/D power stabilization wait time after the hardware trigger is detected.
   (F_{CLK}: CPU/peripheral hardware clock frequency)

11.2.3 A/D converter mode register 1 (ADM1)

This is the register that sets the A/D conversion mode.

The ADM1 register is set via an 8-bit memory manipulation instruction.

After the reset signal is generated, the value of this register becomes “00H”.

Table 11-7: Format of A/D converter mode register 1 (ADM1)

Reset value: 00H R/W

	7	6	5	4	3	2	1	0
ADM1	ADMD	0	0	0	ADSCM	0	0	0

ADMD	Setting of the A/D conversion channel selection mode
0	Select mode
1	Scan mode

ADSCM	Setting of the A/D conversion mode
0	Continuous conversion mode
1	Single conversion mode

Notice:

1. Bit 6~4, 2 must be set to “0”.
2. To override the ADM1 register, it must be done in the conversion stop state (ADCS=0).
3. In order to end the A/D conversion normally, the hardware trigger interval must be set at least to the following time:

   Hardware trigger no-wait mode: 2 F_{CLK} clocks+A/D conversion time

   Hardware trigger wait mode: 2 F_{CLK} clocks+A/D power supply stable wait time+A/D conversion time

   (F_{CLK}: CPU/peripheral hardware clock frequency)

11.2.4 A/D converter mode register 2 (ADM2)

The ADM2 register is set by an 8-bit memory manipulation instruction.

After the reset signal is generated, the value of this register becomes “00H”.

Table 11-8: Format of A/D converter mode register 2 (ADM2)

Reset value: 00H R/W

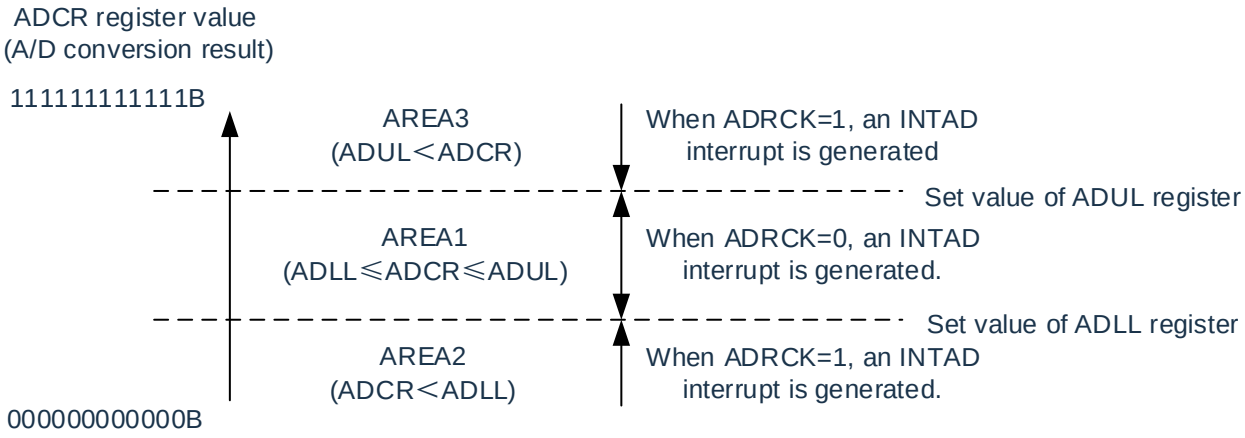
	7	6	5	4	3	2	1	0
ADM2	ADREFP1	ADREFP0	0	0	ADRCK	0	CHRDE	0

ADREFP1	ADREFP0	Selection of positive (+) voltage references for A/D converters
0	0	Provided by V_{DD}.
1	0	Provided by internal reference voltage (1.45V).
Other		Setting disabled.

ADRCK	Checking of upper and lower limit values of conversion results
0	When ADLL register ≤ ADCR register ≤ ADUL register (AREA1), an interrupt signal (INTAD) is generated.
1	When ADCR register < ADLL register (AREA2) or ADUL register < ADCR register (AREA3), an interrupt signal (INTAD) is generated.
The range of interrupt signal (INTAD) generated from AREA1 to AREA3 is shown in Figure 15-8.	

CHRDE	Output enable for channel identification during A/D converter scan mode
0	In scanning mode, the channel number is not identified in the conversion results
1	In scanning mode, the high four bits of the converted result ([15:12] of the ADCR register) are the channel numbers for this result

Figure 11-3: Range of interrupt signal generation for the ADRCK bit



Notice:

1. To override the ADM2 register, it must be done in the conversion stop state (ADCS=0).
2. When INTAD does not occur, the A/D conversion results are not saved to the ADCR register and the ADCRH register.

11.2.5 A/D converter trigger mode register (ADTRG)

This is the register that sets the A/D conversion trigger mode and the hardware trigger signal.

The ADTRG register is set via an 8-bit memory manipulation instruction.

After the reset signal is generated, the value of this register becomes “00H”.

Table 11-9: Format of A/D converter trigger mode register (ADTRG)

Reset value: 00H R/W

	7	6	5	4	3	2	1	0
ADTRG	ADTMD1	ADTMD0	0	0	0	0	ADTRS1	ADTRS0

ADTMD1	ADTMD0	Selection of A/D conversion trigger modes
0	0	Software trigger mode
0	1	
1	0	Hardware trigger no-wait mode
1	1	Hardware trigger wait mode

ADTRS1	ADTRS0	Selection of hardware trigger signals
0	0	The counting end of timer channel 1 or the capture end of interrupt signal (INTTM01).
0	1	The event signal selected by ELC
1	0	Real-time clock interrupt signal (INTRTC).
1	1	Interval timer interrupt signal (INTIT).

Notice:

1. To override the ADTRG register, it must be done in the conversion stop state (ADCS=0, ADCE=0).
2. In order to end the A/D conversion normally, the hardware trigger interval must be set at least to the following time:

   Hardware trigger no-wait mode: 2 F_{CLK} clocks +A/D conversion time

   hardware trigger wait mode: 2 F_{CLK} clocks +A/D power supply stable wait time +A/D conversion time

   (F_{CLK}: CPU/peripheral hardware clock frequency)

11.2.6 Analog input channel specification register (ADS)

This is the register that specifies the analog voltage input channel for A/D converter.

The ADS register is set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register becomes "00H".

Table 11-10: Format of analog input channel specification register (ADS)

Reset value: 00H R/W

	7	6	5	4	3	2	1	0
ADS	ADISS	0	ADS5	ADS4	ADS3	ADS2	ADS1	ADS0

- Select mode (ADM1.ADMD=0)

ADS REGISTER SET VALUE		CH SELECTION
ADISS	ADC[5:0]	
0	6'h00	ANI0(P20)
0	6'h01	ANI1(P21)
0	6'h02	ANI2(P22)
0	6'h03	ANI3(P23)
0	6'h04	ANI4(P24)
0	6'h05	ANI5(P25)
0	6'h06	ANI6(P26)
0	6'h07	ANI7(P30)
0	6'h08	ANI8(P31)
0	6'h09	ANI9(P32)
0	6'h0a	ANI10(P33)
0	6'h0b	ANI11(P34)
0	6'h0c	ANI12(P35)
0	6'h0d	ANI13(P36)
0	6'h0e	ANI14(P37)
0	6'h0f	ANI15(P00)
0	6'h10	ANI16(P01)
0	6'h11	ANI17(P02)
0	6'h12	ANI18(P10)
0	6'h13	ANI19(P11)
1	6'h00	BGR(temperature sensor0)
1	6'h01	BGR
Settings are disabled.		

Remark: The analog input channels of the A/D converters vary by product. Please refer to the data sheet for detailed channel information.

- Scan mode (ADM1.ADMD=1)

ADISS	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
0	0	0	1	1	ANI3	ANI4	ANI5	ANI6
0	0	1	0	0	ANI4	ANI5	ANI6	ANI7
0	0	1	0	1	ANI5	ANI6	ANI7	ANI8
0	0	1	1	0	ANI6	ANI7	ANI8	ANI9
0	0	1	1	1	ANI7	ANI8	ANI9	ANI10
0	1	0	0	0	ANI8	ANI9	ANI10	ANI11
0	1	0	0	1	ANI9	ANI10	ANI11	ANI12
0	1	0	1	0	ANI10	ANI11	ANI12	ANI13
0	1	0	1	1	ANI11	ANI12	ANI13	ANI14
0	1	1	0	0	ANI12	ANI13	ANI14	ANI15
Others:					Settings are disabled.			

Notice:

1. Bit4, bit5 and bit6 must be set to "0" in scan mode.
2. For the port set as analog input by PMCx register, it can be designated as analog input for A/D conversion by ADS.
3. For pins set as digital input/output by the Port Mode Control Register (PMCxx), they cannot be set by the ADS register.
4. To rewrite the ADISS bit, it must be done in the conversion stop state (ADCS=0, ADCE=0).
5. After setting the ADISS bit to "1", the result of the first conversion cannot be used.
6. The ADISS bit cannot be set to "1" when transferring to deep sleep mode or when transferring to sleep mode while the CPU is running at the sub-system clock.

11.2.7 12-bit A/D conversion result register (ADCR)

This is a 16-bit register that stores the A/D conversion result, and this register is read-only. Each time A/D conversion ends, the conversion result is loaded from the successive approximation register (SAR)[Note].

The high 4 bits of this register are fixed to "0" when the mode is selected, and the channel number of the conversion result can be configured by ADM2.CHRDE=1 in scan mode.

The ADCR register can be read by a 16-bit memory manipulation instruction.

After a reset signal is generated, the value of this register becomes "0000H".

Note: If the value of the A/D conversion result is not within the set value range of the A/D conversion result comparison function (set by the ADRCK bit and the ADUL/ADLL register), the A/D conversion results are not saved.

Table 11-11: Format of 12-bit A/D conversion result register (ADCR)

Reset value: 0000H R

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCR	ADCH3	ADCH2	ADCH1	ADCH0	ADCR[11:0]											

Remark:

1. If only 8-bit resolution A/D conversion results are required, the higher 8 bits of the conversion result can be read by the ADCRH register.
2. When 16 bits of access are made to the ADCR register, the higher 12 bits of the conversion result can be read sequentially from bit11.

- Select mode (ADM1.ADMD=0)
  The readout value of ADCH0~3 is fixed at 4‘b0000
- Scan mode (ADM1.ADMD=1) and ADM2.CHRDE=1, the relationship between the readout values of ADCH0~3 and the converted channels is as follows

ADCH3	ADCH2	ADCH1	ADCH0	Conversion channel ID
0	0	0	0	ANI0
0	0	0	1	ANI1
0	0	1	0	ANI2
0	0	1	1	ANI3
0	1	0	0	ANI4
0	1	0	1	ANI5
0	1	1	0	ANI6
0	1	1	1	ANI7
1	0	0	0	ANI8
1	0	0	1	ANI9
1	0	1	0	ANI10
1	0	1	1	ANI11
1	1	0	0	ANI12
1	1	0	1	ANI13
1	1	1	0	ANI14
1	1	1	1	ANI15

11.2.8 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 12-bit resolution are stored[Note].

The ADCRH register can be read by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register becomes "00H".

Note: If the value of the A/D conversion result is not within the set value range of the A/D conversion result comparison function (set by the ADRCK bit and the ADUL/ADLL register), the A/D conversion results are not saved.

Table 11-12: Format of 8-bit A/D conversion result register (ADCRH)

Reset value: 00H R

	7	6	5	4	3	2	1	0
ADCRH								

Notice: Read the conversion result following conversion completion before writing to the ADM0, ADS registers. Otherwise, you may not read the correct conversion results.

11.2.9 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 by the ADRCK bit of A/D converter mode register 2 (ADM2) (shown in Figure 11-3: Range of interrupt signal generation for the ADRCK bit). The ADUL register is set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register becomes “FFH”.

Note 1: Only the higher 8 bits of the 12-bit A/D conversion result register (ADCR) are compared with the ADUL register and the ADLL register.

Note 2: Rewrite the value of the ADUL register and ADLL register while conversion is stopped (ADCS = 0).

Note 3: Rewrite the value of the ADUL register and ADLL register while ADUL＞ADLL.

Table 11-13: Format of conversion result comparison upper limit setting register (ADUL)

Reset value: FFH R/W

	7	6	5	4	3	2	1	0
ADUL	ADUL7	ADUL6	ADUL5	ADUL4	ADUL3	ADUL2	ADUL1	ADUL0

11.2.10 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 by the ADRCK bit of A/D converter mode register 2 (ADM2) (shown in Figure 11-3: Range of interrupt signal generation for the ADRCK bit).

The ADLL register is set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register becomes "00H".

Table 11-14: Format of conversion result comparison lower limit setting register (ADLL)

Reset value: 00H R/W

	7	6	5	4	3	2	1	0
ADLL	ADLL7	ADLL6	ADLL5	ADLL4	ADLL3	ADLL2	ADLL1	ADLL0

Notice:

1. Only the higher 8 bits of the 12-bit A/D conversion result register (ADCR) are compared with the ADUL register and the ADLL register.
2. Rewrite the value of the ADUL register and ADLL register while conversion is stopped (ADCS = 0).
3. Rewrite the value of the ADUL register and ADLL register while ADUL＞ADLL.

11.2.11 A/D converter sampling time control register (ADNSMP)

This register controls the A/D sampling time.

The ADNSMP register is set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register becomes "00H".

Table 11-15: Format of A/D converter sampling time control register (ADNSMP)

Reset value: 00H R/W

	7	6	5	4	3	2	1	0
ADSMPWAIT	0	0	0	0	0	0	0	ADSMPWAIT

ADSMPWAIT	A/D conversion target
0	When it is "0", the A/D sampling time is 4 ADCLKs.
1	When it is "1", the A/D sampling time is 8 ADCLKs.

Remark: Set ADSMPWAIT in the conversion stop state (ADCS=0).

11.2.12 Registers for controlling analog input port function

When using the ANIx pin as the analog input to an A/D converter, the port must be configured as an analog channel by setting the corresponding Port Mode Control Register (PMCxx) bit to “1”. For details, please refer to “Chapter 2 Port Function”.

11.3 Input voltage and conversion results

The analog input voltage at the analog input pin (ANIx) and the theoretical A/D conversion result (12-bit A/D Conversion Result Register (ADCR)) are related by the following expressions.

$$ADCR=INT(\frac{V_{AIN}}{AV_{REF}}\times 4096+0.5) \text{ or } (ADCR-0.5)\times\frac{AV_{REF}}{4096}\leqslant V_{AIN}<(ADCR+0.5)\times\frac{AV_{REF}}{4096}$$

INT(): A function that returns the integer portion of a numeric value in parentheses

V_{AIN}: Analog input voltage

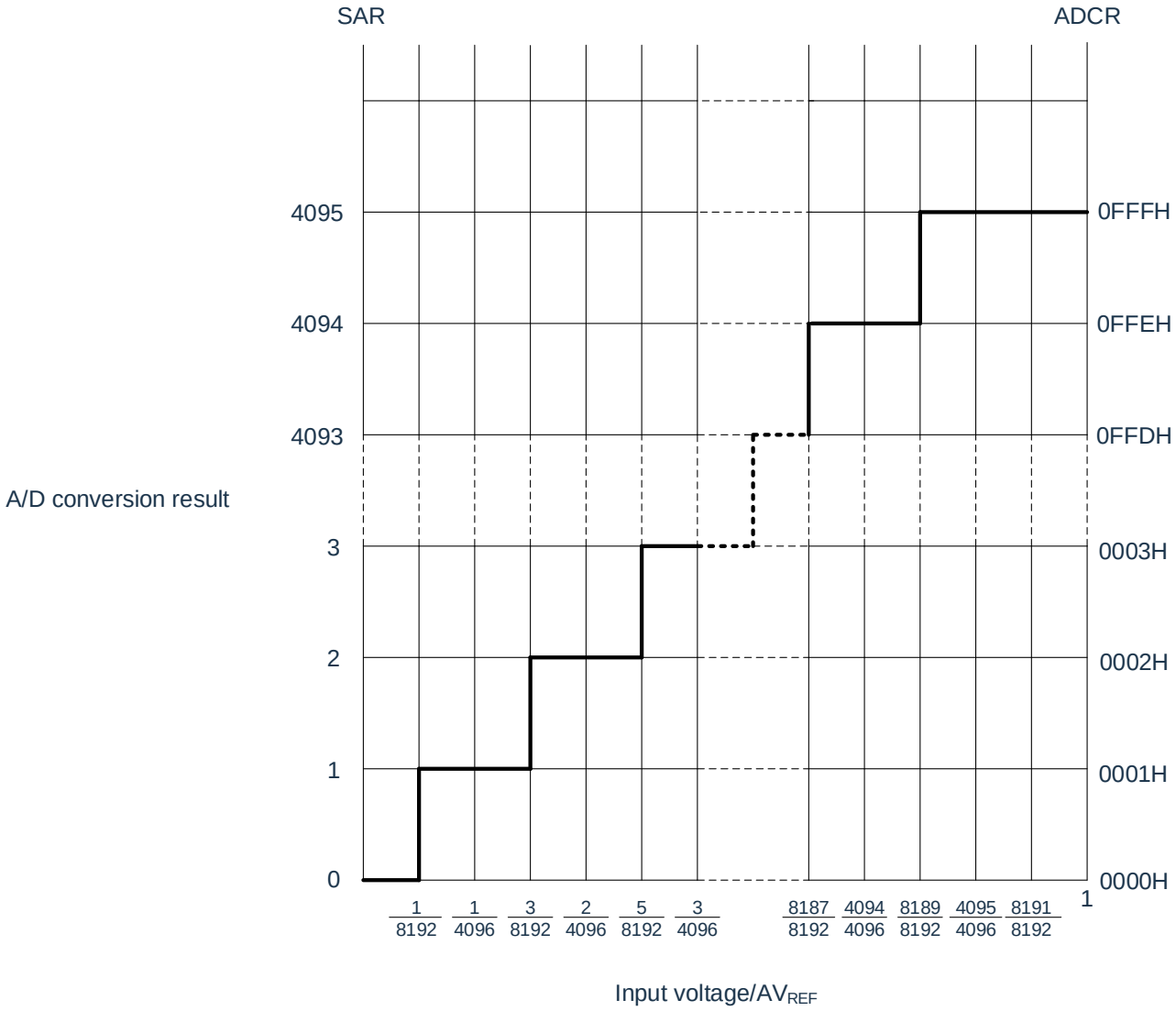
AV_{REF}: AV_{REF} pin voltage

ADCR: The value of the A/D conversion result register (ADCR)

SAR: Successive approximation register

The relationship between the analog input voltage and the A/D conversion results is shown in the following figure.

Figure 11-4: Analog input voltage vs. A/D conversion results



Remark: AV_{REF} is the positive (+) reference voltage of the A/D converter.

11.4 Operation mode of A/D converter

The A/D converter conversion operations are described below. For the setting of each mode, please refer to "11.5 A/D converter setup flowchart".

11.4.1 Software trigger mode (select mode, sequential conversion mode)

① In the stop state, enter A/D conversion standby state by setting the ADCE bit of the A/D converter mode register 0 (ADM0) to "1".

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.

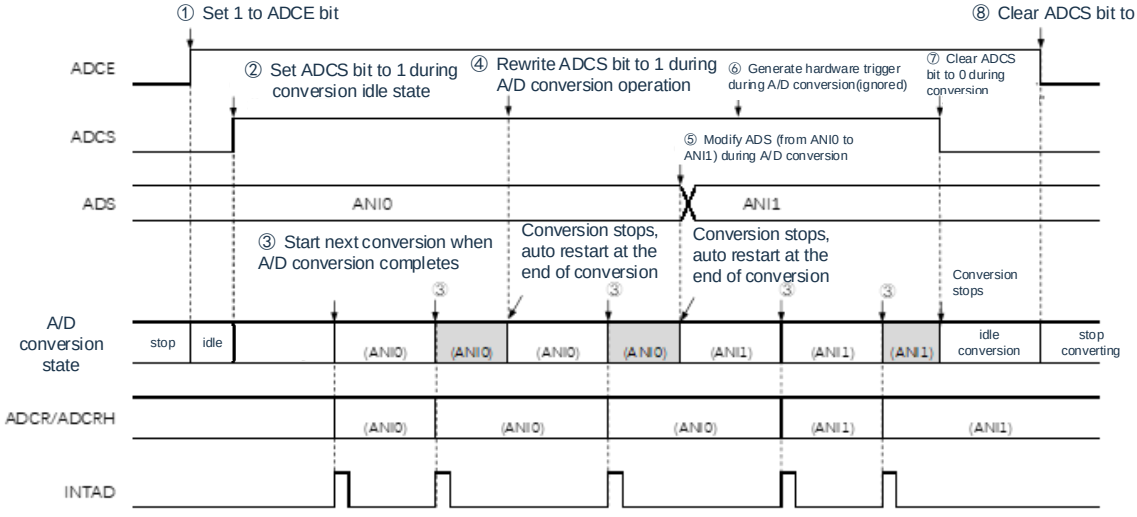
⑤ When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the analog input respecified by the ADS register.

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 11-5: Example of software trigger mode (select mode, sequential conversion mode) operation timing



11.4.2 Software trigger mode (select mode, single conversion mode)

① In the stop state, enter A/D conversion standby state by setting the ADCE bit of the A/D converter mode register 0 (ADM0) to "1".

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.

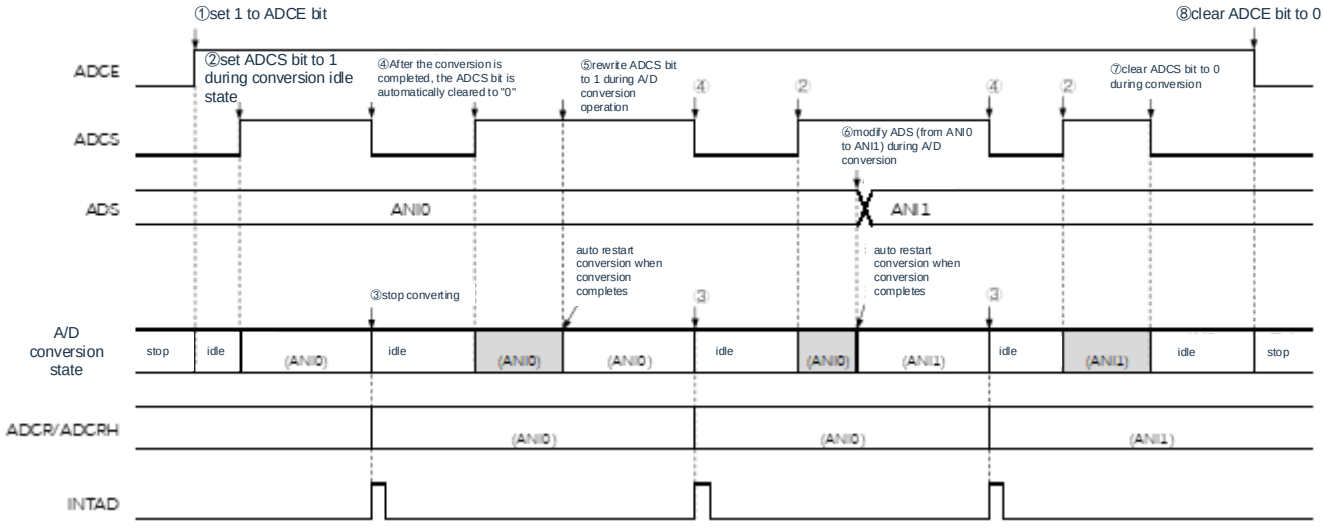
⑤ When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts.

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.

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 11-6: Example of software trigger mode (select mode, single conversion mode) operation timing



11.4.3 Software trigger mode (scan mode, sequential conversion mode)

① In the stop state, enter A/D conversion standby state by setting the ADCE bit of the A/D converter mode register 0 (ADM0) to "1".

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. 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 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.

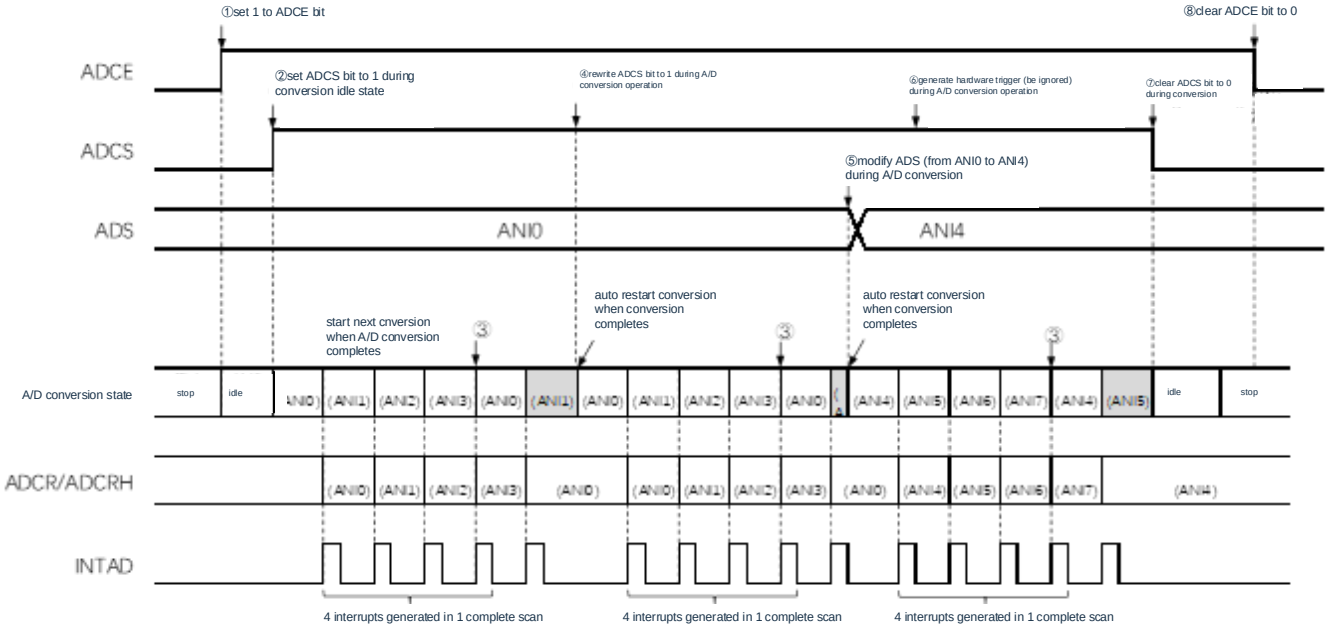
⑤ When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the first channel respecified by the ADS register.

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 11-7: Example of software trigger mode (scan mode, sequential conversion mode) operation timin



11.4.4 Software trigger mode (scan mode, single conversion mode)

① In the stop state, enter A/D conversion standby state by setting the ADCE bit of the A/D converter mode register 0 (ADM0) to "1".

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. 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 of the four channels ends, the ADCS bit is automatically cleared to 0, and the system enters the A/D conversion standby status.

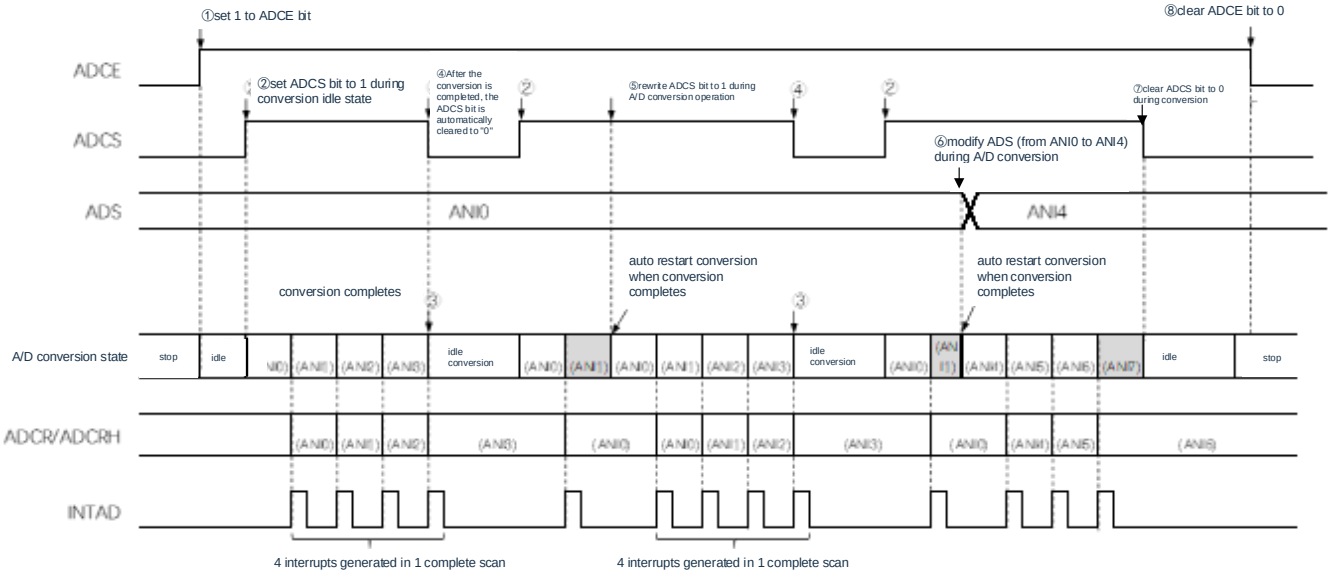
⑤ When ADCS is overwritten with “1” during conversion operation, the current A/D conversion is interrupted, and conversion restarts.

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.

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 11-8: Example of software trigger mode (scan mode, single conversion mode) operation timing



11.4.5 Hardware trigger no-wait mode (select mode, sequential conversion mode)

① In the stop state, enter A/D conversion standby state by setting the ADCE bit of the A/D converter mode register 0 (ADM0) to "1".

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.

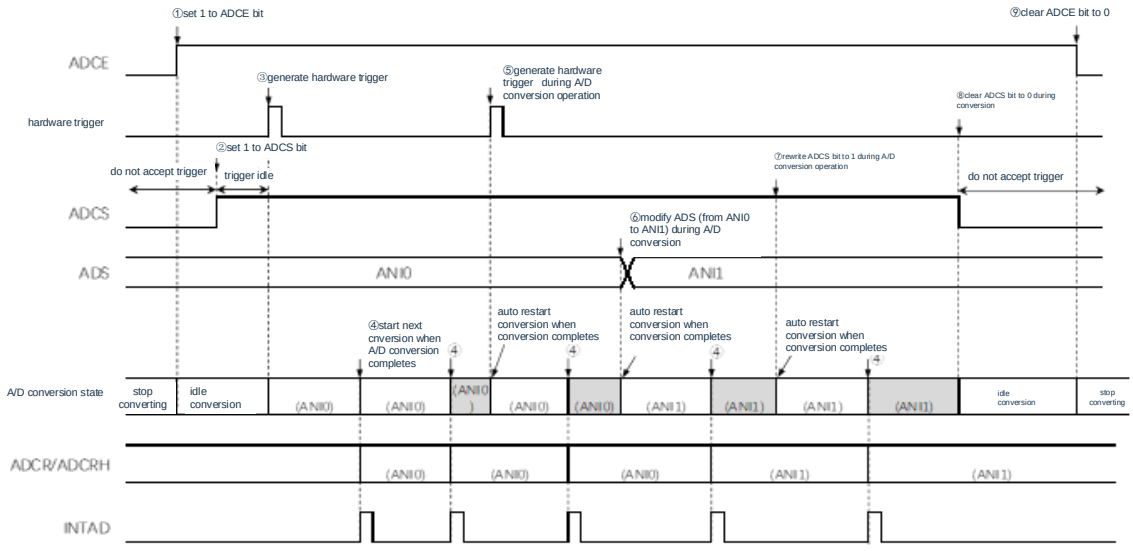
⑥ When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the analog input respecified by the ADS register.

⑦ When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts.

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 11-9: Example of hardware trigger no-wait mode (select mode, sequential conversion mode) operation timing



11.4.6 Hardware trigger no-wait mode (select mode, single conversion mode)

① In the stop state, enter A/D conversion standby state by setting the ADCE bit of the A/D converter mode register 0 (ADM0) to "1".

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.

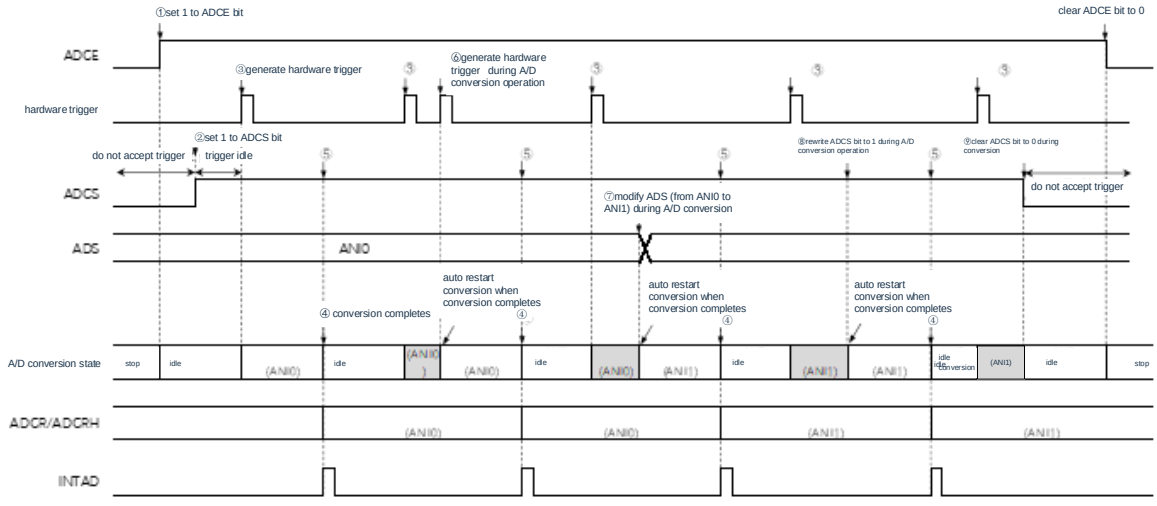
⑦ When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the analog input respecified by the ADS register.

⑧ When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts.

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 11-10: Example of hardware trigger no-wait mode (select mode, single conversion mode) operation timing



11.4.7 Hardware trigger no-wait mode (scan mode, sequential conversion mode)

① In the stop state, enter A/D conversion standby state by setting the ADCE bit of the A/D converter mode register 0 (ADM0) to "1".

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. 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 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.

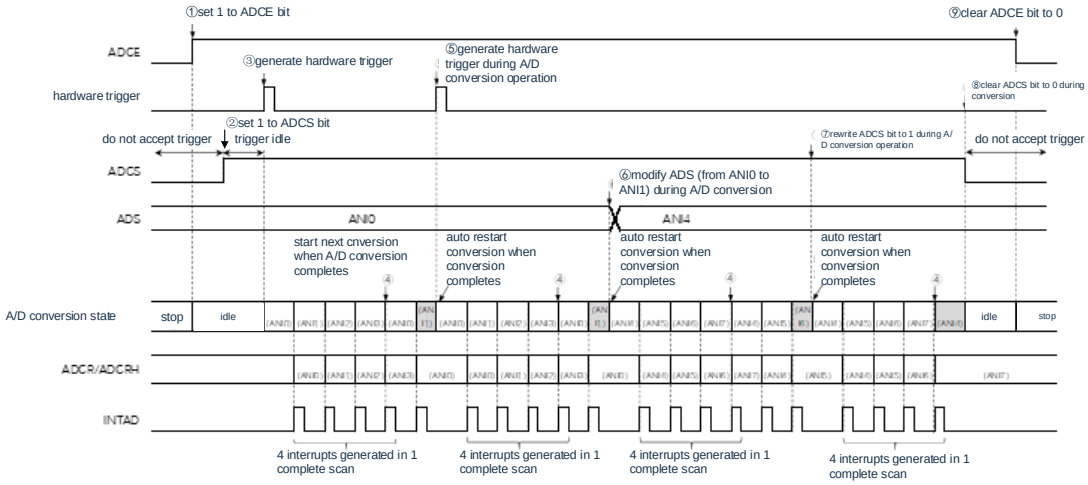
⑥ When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the channel respecified by the ADS register.

⑦ When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel.

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 11-11: Example of hardware trigger no-wait mode (scan mode, sequential conversion mode) operation timing



11.4.8 Hardware trigger no-wait mode (scan mode, single conversion mode)

① In the stop state, enter A/D conversion standby state by setting the ADCE bit of the A/D converter mode register 0 (ADM0) to "1".

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. 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 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.

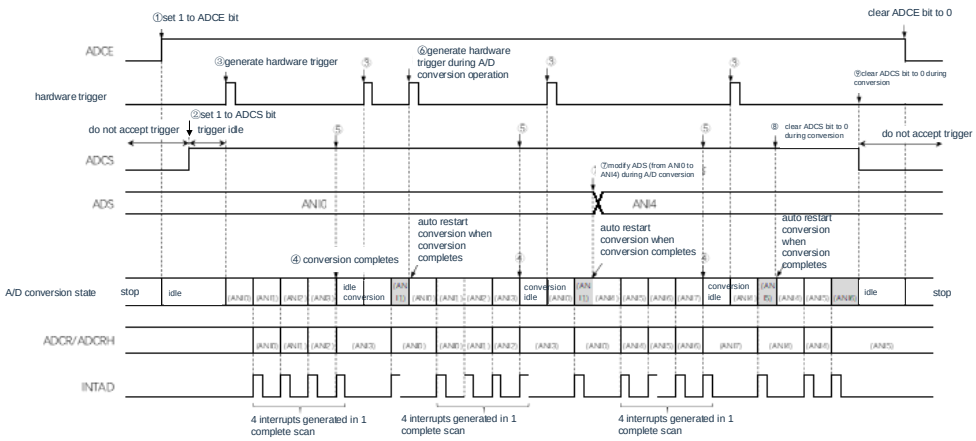
⑦ When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the first channel respecified by the ADS register.

⑧ When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel.

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 11-12: Hardware trigger no-wait mode (scan mode, single conversion mode)



11.4.9 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.)

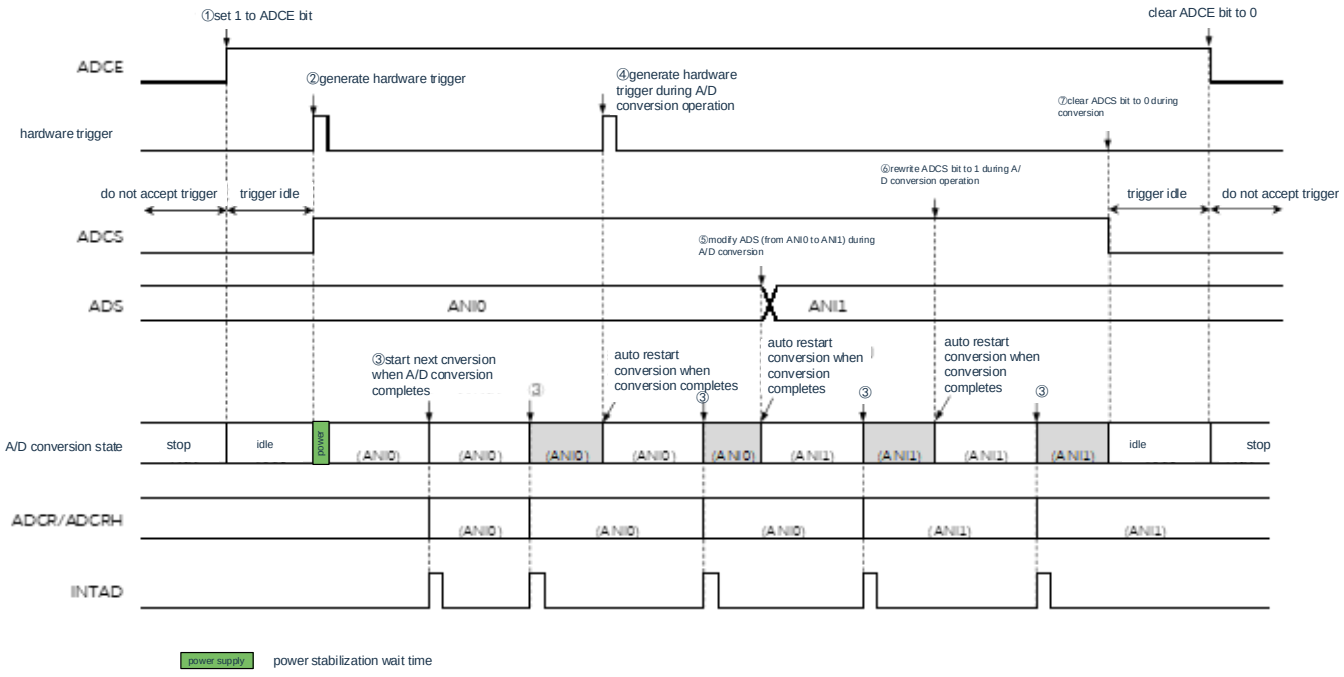
④ If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts.

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.

⑥ When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts.

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 11-13: Example of hardware trigger wait mode (select mode, sequential conversion mode) operation timing



11.4.10 Hardware trigger wait mode (select mode, single 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.

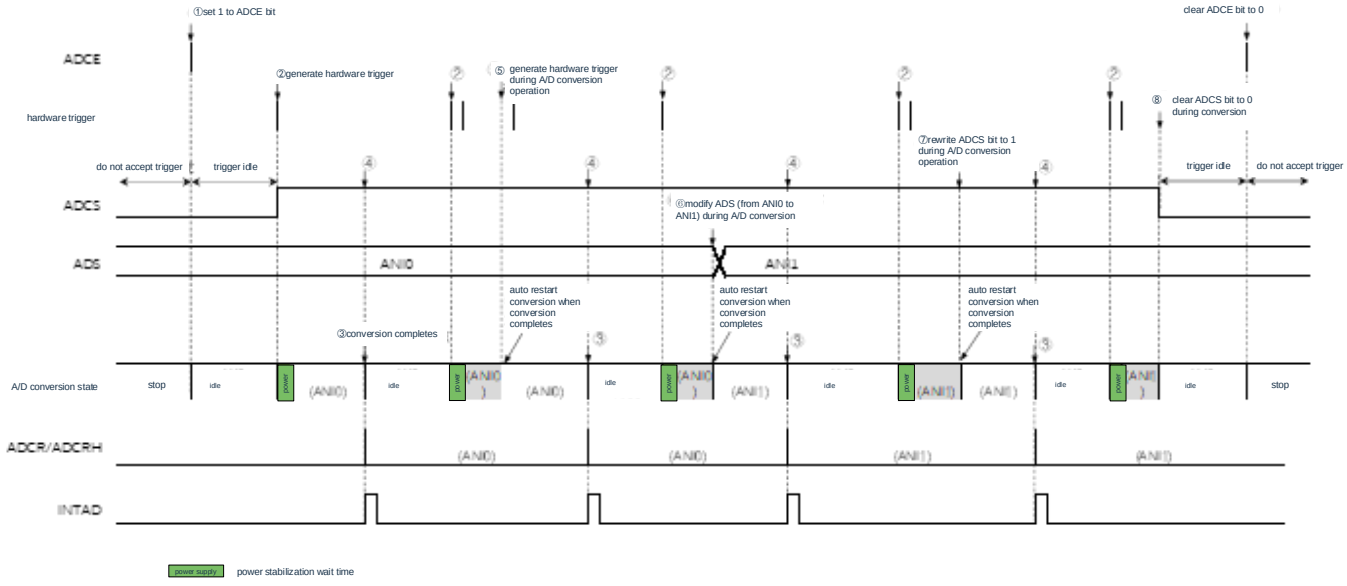
⑤ If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts.

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.

⑦ When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts.

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 11-14: Example of hardware trigger wait mode (select mode, single conversion mode) operation timing



11.4.11 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 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 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. 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 of the four channels ends, the A/D conversion of the channel following the specified channel automatically starts.

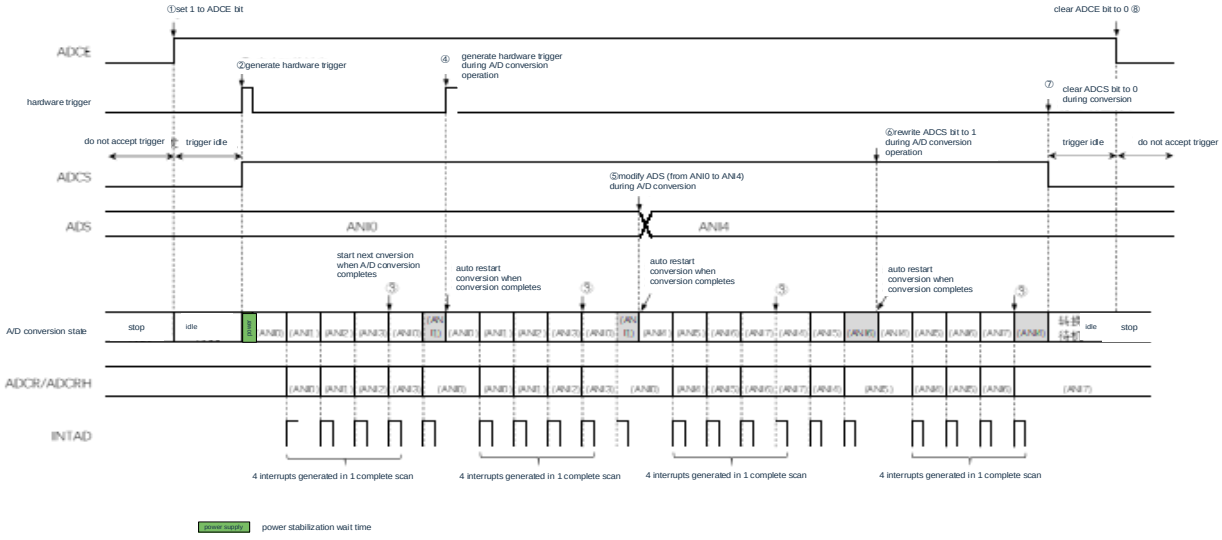
④ If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel.

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 channel respecified by the ADS register.

⑥ When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel.

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 11-15: Example of hardware trigger wait mode (scan mode, sequential conversion mode) operation timing



11.4.12 Hardware trigger wait mode (scan mode, single 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 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. 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.

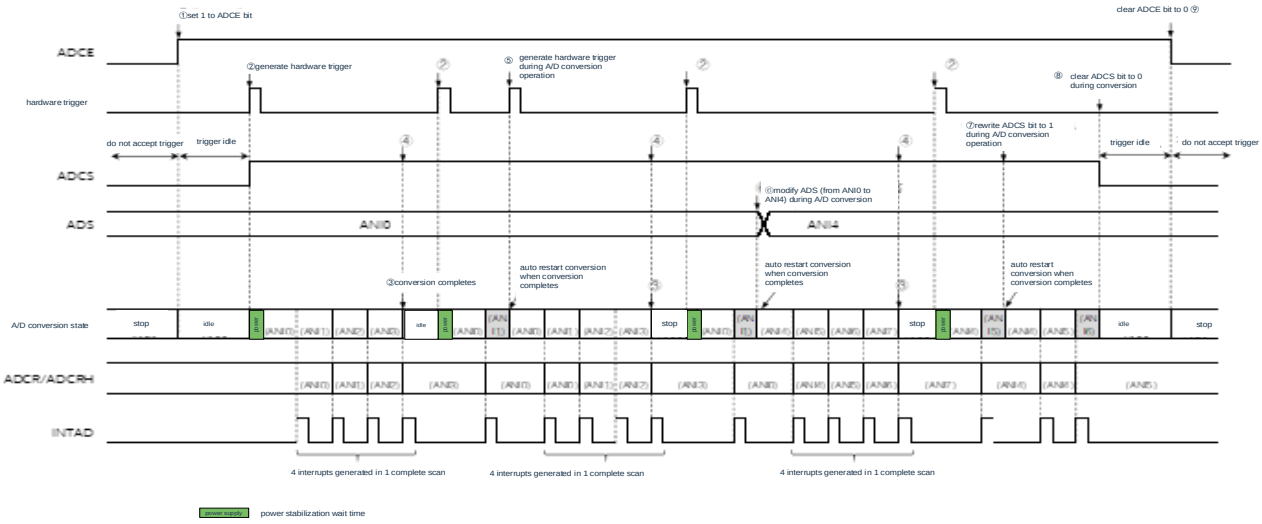
⑤ If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and scan conversion restarts at the first channel.

⑥ When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and scan conversion is performed on the channel respecified by the ADS register.

⑦ When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and scan conversion restarts at the first channel.

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 11-16: Example of hardware trigger wait mode (scan mode, single conversion mode) operation timing

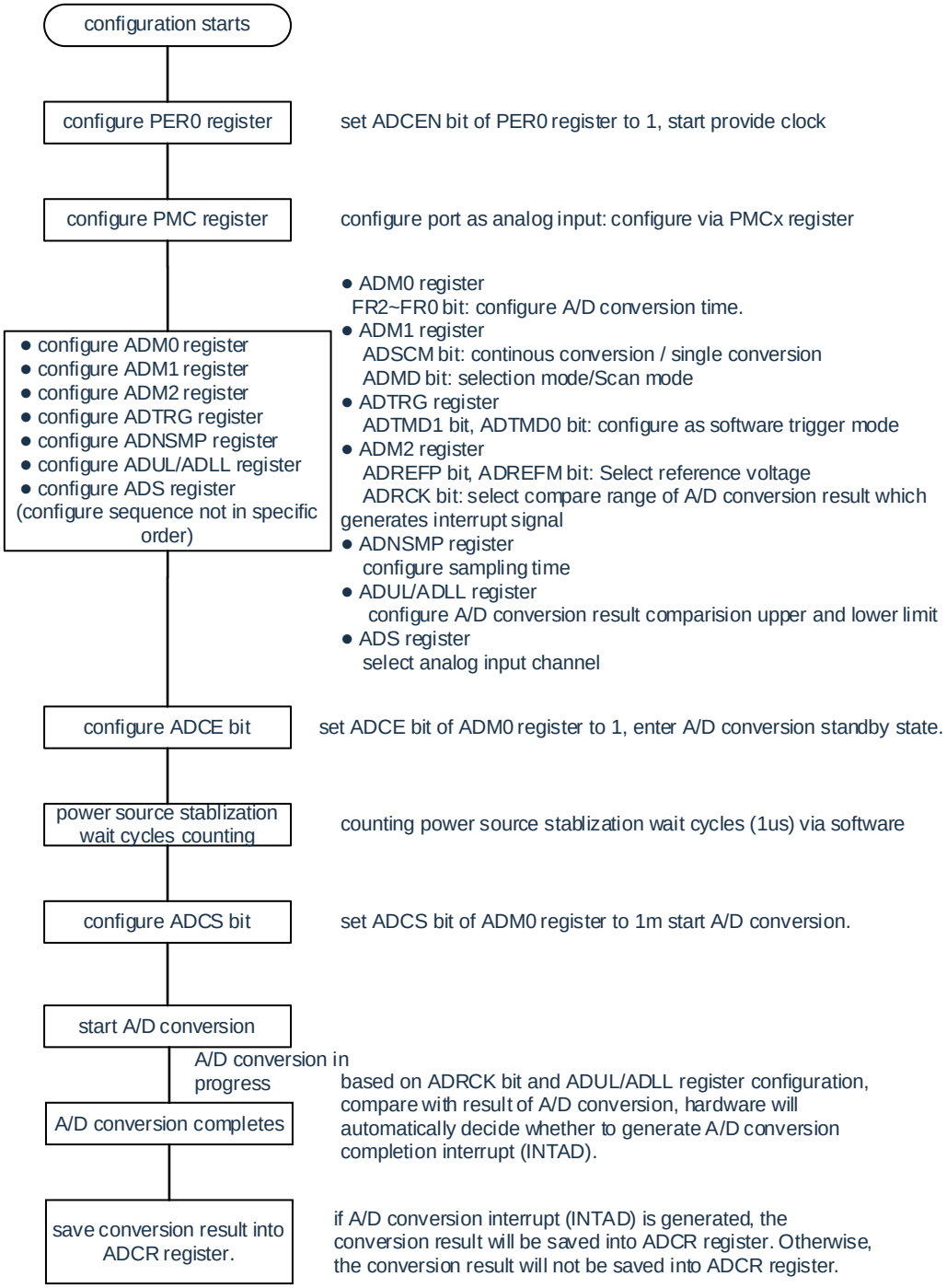


11.5 A/D converter setup flowchart

The A/D converter setup flowchart in each operation mode is described below.

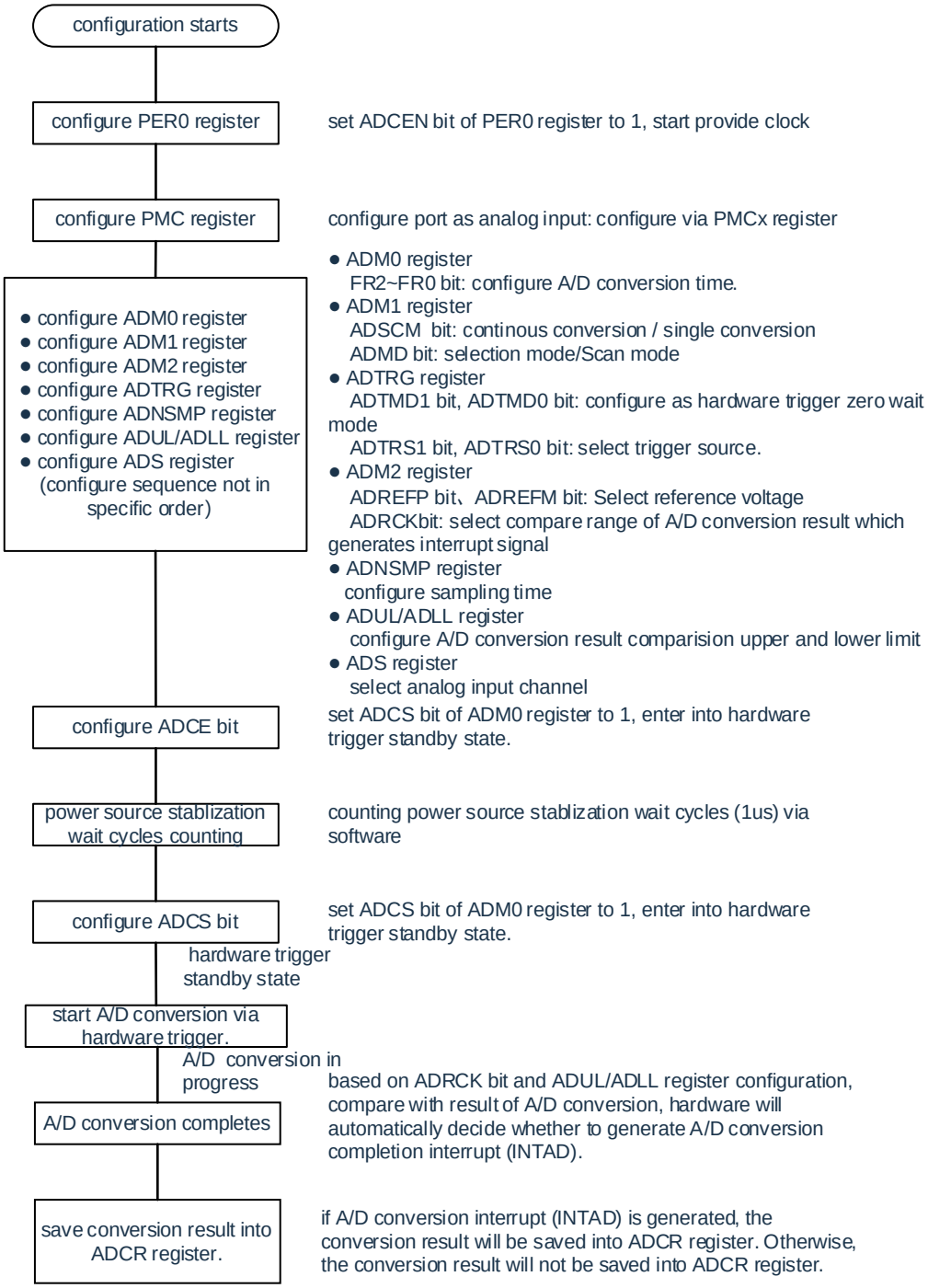
11.5.1 Setting up software trigger mode

Figure 11-17: Setting up software trigger mode



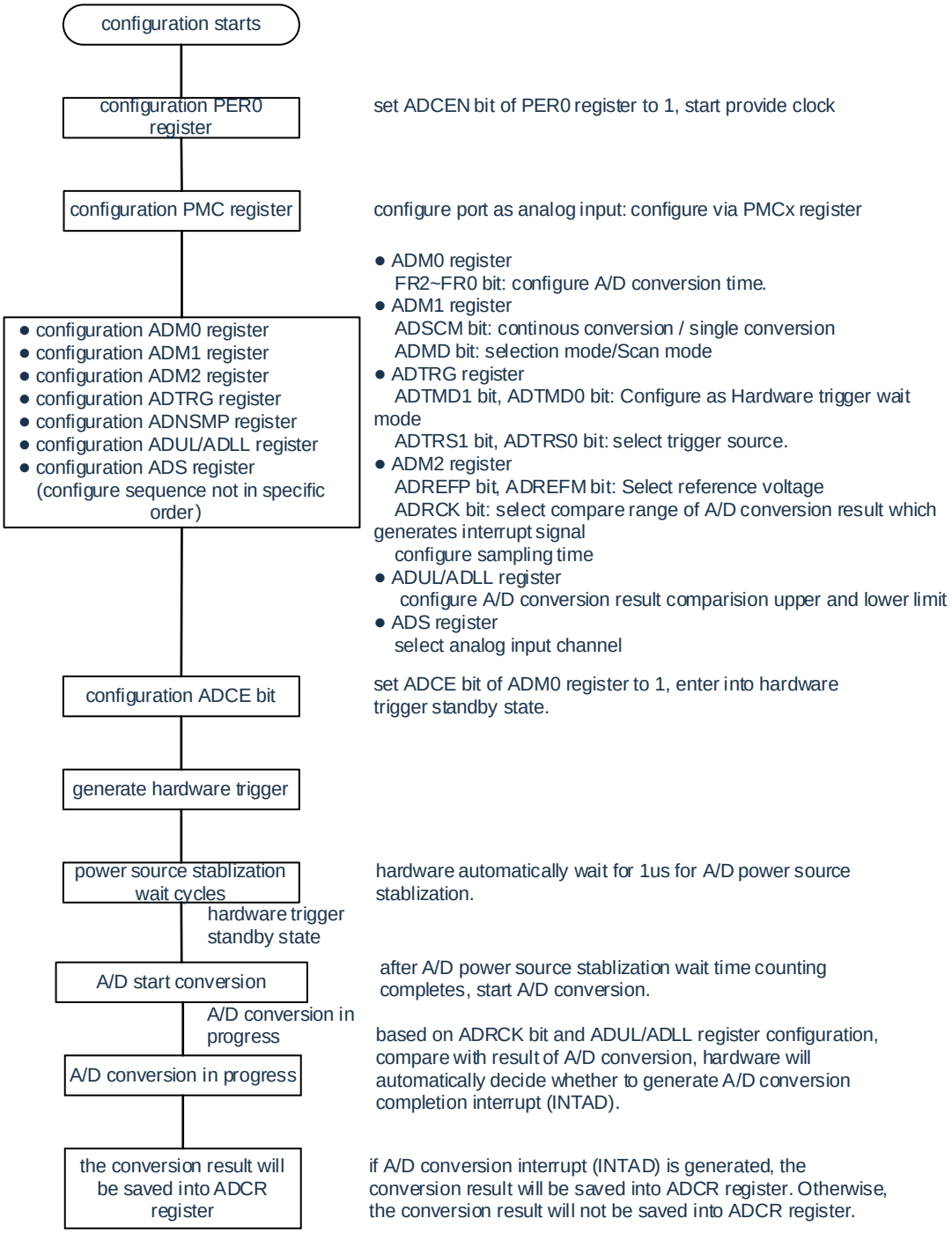
11.5.2 Setting up hardware trigger no-wait mode

Figure 11-18: Setting up hardware trigger no-wait mode



11.5.3 Setting up hardware trigger wait mode

Figure 11-19: Setting up hardware trigger wait mode



Chapter 12 General-Purpose Serial Communication Unit

This product is equipped with 2 general-purpose serial communication units, each unit has 2 serial channels, each channel can realize 3-wire serial (SSPI), UART communication. Function assignment of each channel supported by this product is as shown below.

Table 12-1: Function assignment of general-purpose serial communication unit

Unit	Channel	Used as SSPI	Used as UART
0	0	SSPI00	UART0 (supports LIN-bus)
	1	SSPI01	
1	0	SSPI10	UART1
	1	SSPI11	

Notice:

1. "-" indicates that it is not supported in this model;
2. When “UART0” is used for channels 0 and 1 of the unit 0, SSPI00, SSPI01, IIC00 and IIC01 cannot be used.
3. When “UART1” is used for channels 0 and 1 of the unit 1, SSPI10, SSPI11, IIC10 and IIC11 cannot be used.

12.1 Function of general-purpose serial communication unit

Each serial interface supported by this product has the following features.

12.1.1 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11)

Synchronizes with the serial clock (SCLK) output from the master device for data transmission and reception.

This is a clock-synchronous communication function that uses a serial clock (SCLK), a transmit serial data (SDO), and a receive serial data (SDI) for communication on a total of three communication lines.

For specific setting examples, please refer to “12.1.1 Operation of 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11) communication”.

[Data transmission and reception]

Data length of 7~16 bits

Phase control of transmit/receive data

MSB/LSB preferred option

Level setting for transmit/receive data

[Clock Control]

(1) Master/slave selection

(2) Phase control of I/O clock

(3) Setting of transfer period by prescaler and internal counter

(4) Maximum transfer rate

During master communication: Max.F_{CLK}/2

During slave communication: Max.F_{MCK}/6

[Interrupt function]

Transfer end interrupt, buffer empty interrupt

[Error detection flag]

Overflow error

Notice: Use the clocks within a range satisfying the SCLK cycle time (T_{KCY}) characteristics. For details, please refer to the data sheet.

12.1.2 UART (UART0, UART1)

This is an asynchronous function using two lines: serial data transmission (TxD) and serial data reception (RxD) lines. By using these two communication lines, data is sent and received asynchronously (using the internal baud rate) with other communicating parties by data frame (consisting of start bits, data, parity bits, and stop bits). Full-duplex UART communication can be performed by using a channel dedicated to transmission (even-numbered channel 00) and a channel dedicated to reception (odd-numbered channel 01).

For details about the settings, please refer to “12.7 Operation of UART (UART0~UART1)”.

[Data transmission and reception]

(1) Data length of 7, 8, 9 or 16 bits
(2) MSB/LSB preferred option
(3) Level setting of transmit/receive data and select of reverse
(4) Parity bit appending and parity check functions
(5) Stop bit appending

[Interrupt function]

(1) Transfer end interrupt, buffer empty interrupt
(2) Error interrupt in case of framing error, parity error, or overflow error

[Error detection flag]

Framing error, parity error, or overflow error

12.2 Structure of general-purpose serial communication unit

The general-purpose serial communication unit consists of the following hardware.

Table 12-2: Structure of general-purpose serial communication unit

Item	Structure
Shift register	16-bit
Buffer register	Serial data register mn (SDRmn) Note
Serial clock I/O	SCLKOI00, SCLKOI01, SCLKOI10, SCLKOI11 pins (for 3-wire serial I/O),
Serial data input	SDI00, SDI01, SDI10, SDI11 pins (for 3-wire serial I/O), RxD0, RxD1 pins (for UART)
Serial data output	SDO00, SDO01, SDO10, SDO11 pins (for 3-wire serial I/O), TxD0, TxD1 pins (for UART)
Slave select input	SAU0_SS pin (for serial communication unit 0 as SPI slave select input function) SAU1_SS pin (for serial communication unit 1 as SPI slave select input function)
Control registers	＜Register of unit setting section＞ Peripheral enable register 0(PER0) Serial clock select register m (SPSm) Serial channel enable status register m (SEm) Serial channel start register m (SSm) Serial channel stop register m (STm) Serial output enable register m (SOEm) Serial output register m (SOm) Serial output level register m (SOLm) Input switching control register (ISC) Noise filter enable register 0 (NFEN0)
	＜Registers of each channel＞ Serial data register mn (SDRmn) Serial mode register mn (SMRmn) Serial communication run setting register mn (SCRmn) Serial status register mn (SSRmn) Serial flag clear trigger register mn (SIRmn)
	Port multiplexing function configuration register (PxxCFG) Port output mode register (POMxx) Port mode register (PMxx) Port register (Pxx)

Note: When SEmn=1,

Remark: m: unit number (m=0, 1);

n: channel number (n=0, 1);

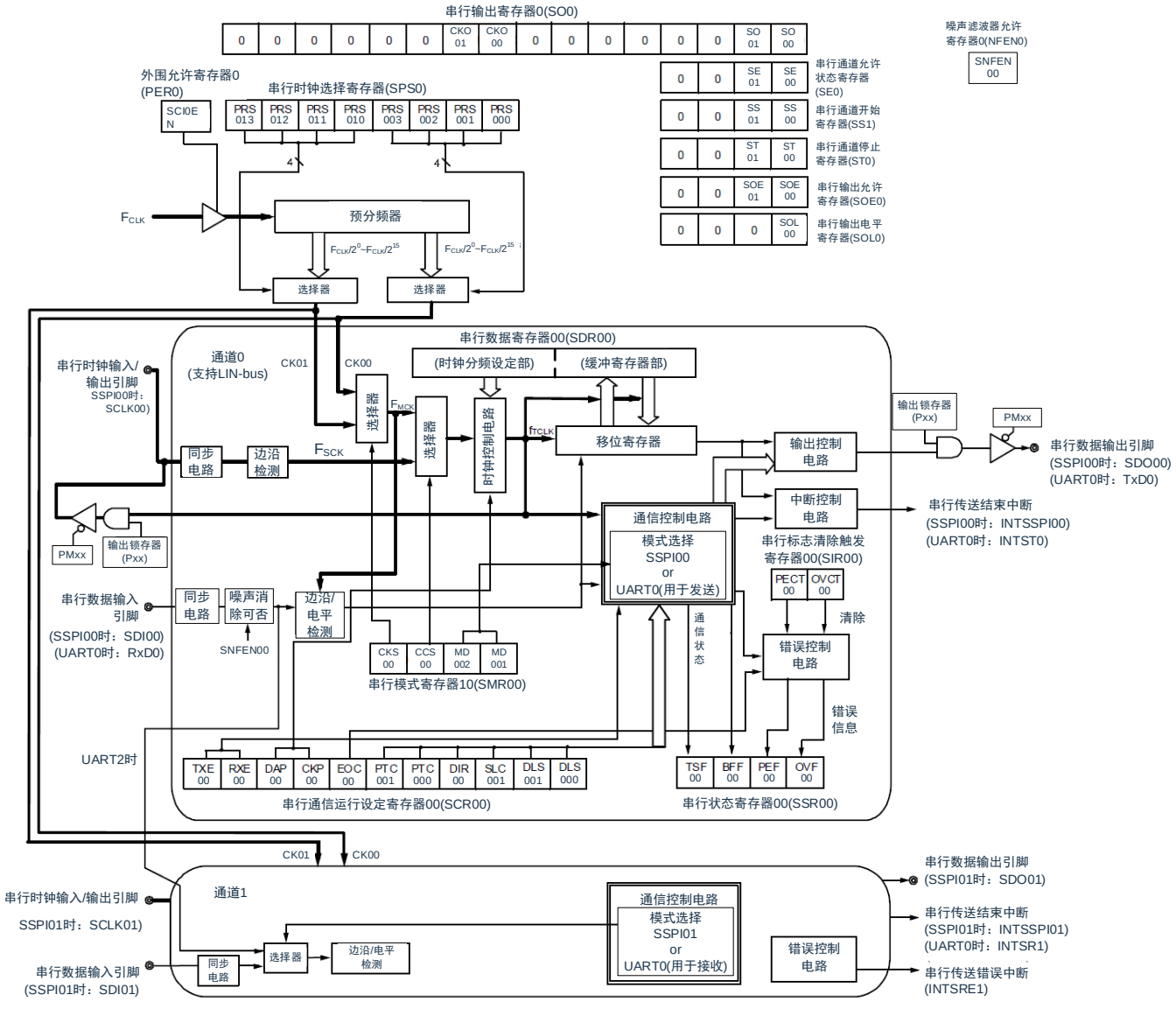
p: SSPI number (p=00, 01, 10, 11);

q: UART number (q=0, 1);

r: IIC number (r=00, 01, 10, 11).

The block diagram of general-purpose serial communication unit is shown in Figure 12-1.

Figure 12-1: Block diagram of general-purpose serial communication unit



Remark: Units 0 and 1 have the same structure

12.2.1 Shift register

This is a 16-bit register that converts parallel data into serial data or vice versa.

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 serial data register mn (SDRmn) when operation is in progress (SEmn = 1).

Table 12-3: Format of shift register

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Shift register																

12.2.2 Serial data register mn (SDRmn)

The SDRmn register is the transmit/receive data register (16 bits) of channel n.

If operation is stopped (SEmn = 0), bits 15 to 9 are used as a register that sets the division ratio of the operation clock (F_{MCK}). If operation is in progress (SEmn = 1), bit15~9 selected as a transmit/receive buffer register.

When receiving data, the parallel data converted by the shift register is saved to the serial data register SDRmn; when sending data, the send data that is transferred to the shift register is set to the serial data register SDRmn.

Regardless of the output order of the data, the data saved to the SDRmn register according to the setting of bit3 to bit0 (DLSmn3 to DLSmn0) of the serial communication run setting register mn (SCRmn) is shown below:

(1) 7-bit data length (stored in bit0~6 of SDRmn register)

(2) 8-bit data length (stored in bit0~7 of SDRmn register)

(3) 16-bit data length (stored in bit0~15 of SDRmn register)

The SDRmn register can be read or written in 16-bit units.

When SEmn=1, the lower 8 bits of the SDRmn register can be read and written in 8-bit increments as SDRmnL[Note].

According to the communication mode, it can read and write SDRmnL registers with the following SFR names.

(1) SSPIp communication……SDIOpL

(2) UARTq reception……RXDq (UARTq receive data register)

(3) UARTq transmission……TXDq (UARTq transmit data register)

After the reset signal is generated, the value of SDRmn register changes to "0000H".

Note: At run stop (SEmn=0), it is forbidden to rewrite SDRmn[7:0] by 8-bit memory manipulation instruction (otherwise, all of SDRmn[15:9] will be cleared to "0").

Remark: unit number (m=0, 1);

n: channel number (n=0, 1);

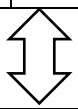
p: SSPI number (p=00, 01, 10, 11);

q: UART number (q=0, 1);

Table 12-4: Format of serial data register mn (SDRmn)

After reset: 0000H R/W

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn																



	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Shift register																

Remark:

1. For the function of the higher 7 bits of the SDRmn register, please refer to “12.3 Registers for controlling general-purpose serial communication unit”;
2. m: unit number (m=0, 1);
   n: channel number (n=0, 1).

12.3 Registers for controlling general-purpose serial communication unit

The registers controlling general-purpose serial communication unit are shown below:

(1) Peripheral enable register 0 (PER0)
(2) Serial clock select register m (SPSm)
(3) Serial mode register mn (SMRmn)
(4) Serial communication run setting register mn (SCRmn)
(5) Serial data registermn (SDRmn)
(6) Serial flag clear trigger register mn (SDIRmn)
(7) Serial status registermn (SSRmn)
(8) Serial channel start register m (SSm)
(9) Serial channel stop register m (STm)
(10) Serial channel enable status register m (SEm)
(11) Serial output enable register m (SOEm)
(12) Serial output level register m (SOLm)
(13) Serial output register m (SOm)
(14) Input switching control register (ISC)
(15) Noise filter enable register 0 (NFEN0)
(16) Port multiplexing function configuration register (PxxCFG)
(17) Port output mode register (POMx)
(18) Port mode register (PMx)
(19) Port register (Px)

Remark: m: unit number (m=0, 1); n: channel number (n=0, 1).

Serial communication unit register list

Unit 0 register base address: 0x40041100

Unit 1 register base address: 0x40041500

Table 12-5: Register list

Offset address	Register name	R/W	Reset value
0x000	SSRm0	R	0000H
0x002	SSRm1	R	0000H
0x004	SIRm0	R/W	0000H
0x006	SIRm1	R/W	0000H
0x008	SMRm0	R/W	0020H
0x00A	SMRm1	R/W	0020H
0x00C	SCRm0	R/W	0087H
0x00E	SCRm1	R/W	0087H
0x010	SEm	R/W	0000H
0x012	SSm	R/W	0000H
0x014	STm	R/W	0000H
0x016	SPSm	R/W	0000H
0x018	SOm	R/W	0303H
0x01A	SOEm	R/W	0000H
0x020	SOLm	R/W	0000H
0x040	SDRm0	R/W	0000H
0x042	SDRm1	R/W	0000H

Remark: unit number m=0, 1.

12.3.1 Peripheral enable register 0 (PER0)

The PER0 register is the register that sets whether to enable or disable the supply of clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocks to hardware that is not in use.

To use general-purpose serial communication unit 0, bit 2 (SCI0EN) must be set to "1".

To use general-purpose serial communication unit 1, bit 3 (SCI1EN) must be set to "1".

The PER0 register is set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of the PER0 register changes to "00H".

Table 12-6: Format of peripheral enabled register 0 (PER0)

Address: 0x40020420	After reset: 00H					R/W		
Symbol	7	6	5	4	3	2	1	0
PER0	RTCEN	0	ADCEN	IICAEN	SCI1EN	SCI0EN	TM41EN	TM40EN

SCImEN	Provides control of the input clock of general-purpose serial communication unit m
0	Stop to supply the input clock. The SFR used by the general-purpose serial communication unit m cannot be written. The general-purpose serial communication unit m is in the reset state.
1	Enable the input clock to be provided. Can read and write the SFR used by the general-purpose serial communication unit m.

Notice: To set the Universal Serial Communication Unit m, the following registers must be set in the state of SCImEN bit "1" first. When the SCImEN bit is "0", the write operation of the control registers of the Universal Serial Communication Unit m is ignored, and the read values are all initial values (input switching control register (ISC), noise filter allow register 0 (NFEN0), port multiplex function configuration register (PxxCFG), port output mode register (POMx), port mode register (PMx), and port mode register (PMx). Mode Register (POMx), Port Mode Register (PMx), Port Mode Control Register (PMCx), and Port Register (Px) are excluded).

(1) Serial clock select register m (SPSm)
(2) Serial mode register mn (SMRmn)
(3) Serial communication run setting registermn (SCRmn)
(4) Serial data register mn (SDRmn)
(5) Serial flag clear trigger register mn (SIRmn)
(6) Serial status register mn (SSRmn)
(7) Serial channel start register m (SSm)
(8) Serial channel stop register m (STm)
(9) Serial channel enable status register m (SEm)
(10) Serial output enable register m (SOEm)
(11) Serial output level register m (SOLm)
(12) Serial output register m (SOm)

12.3.2 Serial clock select register m (SPSm)

The SPSm register is a 16-bit register that selects two common operating clocks (CKm0, CKm1) available to each channel. Select CKm1 by bit7 to 4 of the SPSm register and select from bit3 to 0 CKm0.

It is forbidden to overwrite the SPSm register during operation (SEmn=1).

The SPSm register is set by a 16-bit memory manipulation instruction.

It can set the low 8 bits of the SPSm register with SPSmL and through the 8-bit memory manipulation instruction.

After generating a reset signal, the value of the SPSm register changes to "0000H".

Table 12-7: Format of serial clock selection register m (SPSm)

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	PRSm	PRSm	PRSm	PRSm	PRSm	PRSm	PRSm	PRSm

PRSmk3	PRSmk2	PRSmk1	PRSmk0	Operation clock (CKmk) selection[Note]
0	0	0	0	F_{CLK}
0	0	0	1	$F_{CLK}/2$
0	0	1	0	$F_{CLK}/2^2$
0	0	1	1	$F_{CLK}/2^3$
0	1	0	0	$F_{CLK}/2^4$
0	1	0	1	$F_{CLK}/2^5$
0	1	1	0	$F_{CLK}/2^6$
0	1	1	1	$F_{CLK}/2^7$
1	0	0	0	$F_{CLK}/2^8$
1	0	0	1	$F_{CLK}/2^9$
1	0	1	0	$F_{CLK}/2^{10}$
1	0	1	1	$F_{CLK}/2^{11}$
1	1	0	0	$F_{CLK}/2^{12}$
1	1	0	1	$F_{CLK}/2^{13}$
1	1	1	0	$F_{CLK}/2^{14}$
1	1	1	1	$F_{CLK}/2^{15}$

Note: When you change the clock selected as F_{CLK} (change the value of the system clock control register (CKC))during the operation of the general-purpose serial communication unit (SCI), you must stop the operation of the SCI (serial channel stop register m). (STm)=000FH) after making changes.

Notice: Bits 15~8 must be set to "0".

Remark:

1. F_{CLK}: CPU/peripheral hardware clock frequency;
2. m: Unit number (m=0, 1);
3. k=0, 1.

12.3.3 Serial mode register mn (SMRmn)

The SMRmn register is a register that sets the operating mode of channel n, selects the operating clock (F_{MCK}), specifies whether the serial clock (F_{SCLK}) input can be used, sets the start of triggering, and operates the mode settings (SSPI, UART) and selection of interrupt sources. In addition, the inverting level of the received data is set only in UART mode.

It is forbidden to overwrite the SMRmn register during operation (SEmn=1), but it is possible to overwrite the MDmn0 bit during operation.

The SMRmn register is set by a 16-bit memory manipulation instruction.

After generating a reset signal, the value of the SMRmn register changes to "0020H".

Table 12-8: Format of serial mode register mn (SMRmn) (1/2)

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	STSmn	0	SISmn0	0	1	0	0	MDmn2	MDmn1	MDmn0

CKSmn	Selection of channel n operating clock (F_{MCK})
0	The SPSm register sets the operating clock CKm0
1	The SPSm register sets the operating clock CKm1
The operating clock (F_{MCK}) is used for edge detection circuitry. By setting the CCSmn bit and the SDRmn register higher 7 bits, a transmit clock (F_{TCLK}) is generated.	

CCSmn	Selection of channel n transmission clock (F_{TCLK})
0	The CKSmn bit specifies the running clock F_{MCK} divider clock
1	Input clock F_{SCLK} from the SCLKp pin (slave transfer in SSPI mode).
The transmit clock F_{TCLK} is used for shift registers, communication control circuits, output controllers, interrupt control circuits, and error control circuits. When the CCSmn bit is at "0", the operating clock ((F_{MCK}) is set the dividing ratio of the operating clock (F_{MCK}) by the higher 7 bits of the SDRmn register.	

STSmn[Note1]	Selection of start trigger sources
0	Only software triggers are valid (selected in SSPI, UART transmit).
1	The effective edge of the RxDq pin (selected when received by the UART).
When the above conditions are met after setting the SSm register to "1", the transfer starts.	

Note 1: Limited to SMR01, and SMR11 registers only.

Notice: Bits 13~9, 7, 4, 3 (SMR00, SMR10 registers are bit13~6, 4, 3) are set "0" and set bit 5 to “1".

Remark: m: unit number (m=0, 1);

n: channel number (n=0, 1);

p: SSPI number (p=00, 01, 10, 11);

q: UART number (q=0, 1);

Table 12-8: Format of serial mode register mn (SMRmn) (2/2)

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	STSmn	0	SISmn0	0	0	0	0	MDmn2	MDmn1	MDmn0

SISmn0[Note1]	Level inversion control of channel n receives data in UART mode
0	Detect the falling edge as the starting bit. The input communication data is not inverted.
1	Detects the rising edge as the starting bit. Invert the input communication data.

MDmn2	MDmn1	Setting of channel n operation mode
0	0	SSPI mode
0	1	UART mode
1	0	Reserved
1	1	Settings are disabled.

MDmn0	Channel n interrupt source selection
0	Transfer end interrupt
1	Buffer null interrupt (Occurs when data is transferred from the SDRmn register to the shift register).
On consecutive sends, if the MDmn0 bit is "1" and the data for SDRmn is empty, write down the next send data.	

Note 1: Limited to SMR01, and SMR11 registers.

Notice: Bits 13~9, 7, 4, 3 (SMR00, SMR10 registers are bit13~6, 4, 3) are set to “0” and set bit 5 to “1".

Remark: m: unit number (m=0, 1);

n: channel number (n=0, 1);

p: SSPI number (p=00, 01, 10, 11);

q: UART number (q=0, 1);

12.3.4 Serial communication run setting register mn (SCRmn)

The SCRmn register is the communication operation setting register of channel n, which sets the data transmission and reception modes, data and clock phases, whether to mask the error signal, parity test bits, start bits, stop bits, and data length.

It is forbidden to overwrite the SCRmn register during operation (SEmn=1).

The SCRmn register is set by means of a 16-bit memory manipulation instruction.

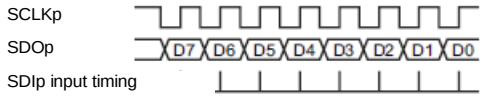
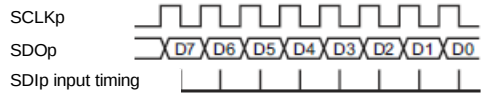
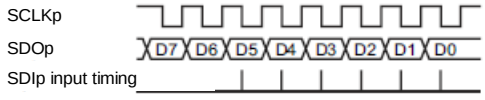
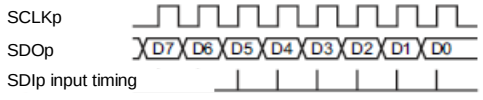
After generating a reset signal, the value of the SCRmn register changes to "0087H".

Table 12-9: Format of serial communication run setting register mn (SCRmn) (1/3)

After reset: 0087H R/W

Symbol	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	DLSmn3	DLSmn2	DLSmn1	DLSmn0

TXEmn	RXEmn	Setting of channel n operation mode
0	0	Prohibited communication.
0	1	Receive only.
1	0	Send only.
1	1	Enable sending and receiving.

DAPmn	CKPmn	Data and clock phase selection in SSPI mode	Type
0	0		1
0	1		2
1	0		3
1	1		4
In UART mode , DAPmn bit and CKPmn bit must be set to 0.			

EOCmn	Mask control of error interrupt signal (INTSREx (x=0~1))
0	Generating the error interrupt INTSREx (generating INTSRx) is prohibited.
1	Enables generation of the error interrupt INTSREx (INTSRx is not generated when an error occurs).
The EOCmn bit must be set to "0" in SSPI mode, or when transmitting from the UART Note 2.	

Note 1: Limited to SCR00, SCR10 registers;

Note 2: When the EOCmn bit is "0" and SSPImn is not used, it is possible to generate the error interrupt INTSREn;

Notice: Bits 6, 10, 11 must be set to "0" (and bit 5 of the SCR01, SCR11 registers must also be set to "0");

Remark: m: unit number (m=0, 1);

n: channel number (n=0, 1);

p: SSPI number (p=00, 01, 10, 11).

Table 12-9: Format of serial communication run setting register mn (SCRmn) (2/3)

After reset: 0087H R/W

Symbol	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	DLSmn3	DLSmn2	DLSmn1	DLSmn0

PTCmn1	PTCmn0	Setting of parity bits in UART mode	
		Transmitting	Receiving
0	0	No parity bits are output	No parity check on reception
0	1	Output parity[Note 3]	Does not determine parity
1	0	Output even-check	Determine the even parity
1	1	Output odd checksum	Determine the odd checksum
In SSPI mode, both PTCmn1 and PTCmn0 bits must be set to "0".			

DIRmn	Selection of data transfer sequence in SSPI and UART modes
0	Perform MSB-first input/output.
1	Perform LSB-first input/output.

SLCmn1[Note1]	SLCmn0	Setting of the 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, 10, 20 only)
1	1	Prohibit setting.
If the transfer end interrupt is selected, the interrupt is generated after all stop bits have been transferred. It must be set to 1 stop bit (SLCmn1, SLCmn0=0, 1) during UART receive. In SSPI mode, it must be set to no stop bit (SLCmn1, SLCmn0 = 0, 0). It must be set to 1 bit (SLCmn1, SLCmn0=0, 1) or 2 bits (SLCmn1, SLCmn0=1, 0) when UART is sent.		

Note 1: Limited to SCR00, SCR10 registers.

Notice:

1. It is always appended with "0", regardless of the content of the data.
2. Bits 6, 10 and 11 must be set to "0".

Remark: m: unit number (m=0, 1);

n: channel number (n=0, 1);

p: SSPI number (p=00, 01, 10, 11).

Table 12-9: Format of serial communication run setting register mn (SCRmn) (3/3)

After reset: 0087H R/W

Symbol	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	DLSmn3	DLSmn2	DLSmn1	DLSmn0

DLSmn3	DLSmn2	DLSmn1	DLSmn0	Settings of the data length	Serial function correspondence	
					SSPI	UART
0	1	1	0	Data length of 7 bits (bit0 to 6 stored in the SDRmn register).	○	○
0	1	1	1	Data length of 8 bits (bit0 to 7 saved in the SDRmn register).	○	○
1	0	0	0	Data length of 9 bits (bit0 to 8 saved in the SDRmn register).	○	○
1	0	0	1	Data length of 10 bits (bit0 to 9 saved in the SDRmn register).	○	×
1	0	1	0	Data length of 11 bits (bit0 to 10 stored in the SDRmn register).	○	×
1	0	1	1	Data length of 12 bits (bit0 to 11 stored in the SDRmn register).	○	×
1	1	0	0	Data length of 13 bits (bit0 to 12 saved in the SDRmn register).	○	×
1	1	0	1	Data length of 14 bits (bit0 to 13 stored in the SDRmn register).	○	×
1	1	1	0	Data length of 15 bits (bit0 to 14 saved in the SDRmn register).	○	×
1	1	1	1	Data length of 16 bits (bit0 to 15 stored in the SDRmn register).	○	○
Others:				Settings are disabled		

Note 1: Limited to SCR00, SCR10 registers.

Notice: Bits 6, 10, 11 must be set to "0".

Remark: m: unit number (m=0, 1);

n: channel number (n=0, 1);

p: SSPI number (p=00, 01, 10, 11).

12.3.5 Serial data register mn (SDRmn)

The SDRmn register is the data register (16-bit) that channel n sends and receives.

When the operation stops (SEmn=0), bit15~9 is used as a crossover setting register for the operating clock (FMCK). During operation (SEmn=1) bit15~9 is used as a transmit and receive buffer register.

If the CCSmn bit of the serial mode register mn (SMRmn) is "0", the bit15 to 9 of the SDRmn register is used. The divider clock of the operating clock (higher 7 bits) is used as the transmission clock.

The SIRmn register is set by means of a 16-bit memory manipulation instruction.

After generating a reset signal, the value of the SDRmn register changes to "0000H".

Table 12-10: Format of serial data register mn (SDRmn)

After reset: 0000H R/W

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn																

SDRmn[15:9]							Transmission clock setting for running clock division
0	0	0	0	0	0	0	F_{MCK}
0	0	0	0	0	0	1	$F_{MCK}/2$
0	0	0	0	0	1	0	$F_{MCK}/3$
0	0	0	0	0	1	1	$F_{MCK}/4$
•	•	•	•	•	•	•	•
1	1	1	1	1	1	0	$F_{MCK}/127$
1	1	1	1	1	1	1	$F_{MCK}/128$

Notice:

1. When operation is stopped (SEmn=0), bit8~0 must be cleared to zero.
2. When using UART, it is prohibited to set SDRmn[15:9] to "000000000B" and "000000001B".
3. When operation is stopped (SEmn=0), it is prohibited to rewrite SDRmn[7:0] by an 8-bit memory manipulation instruction (otherwise, all of SDRmn[15:9] is cleared to "0").

Remark:

1. For the function of the SDRmn register during operation, please refer to “12.2 Structure of general-purpose serial communication unit”.
2. m: unit number (m=0, 1);
   n: channel number (n=0, 1).

12.3.6 Serial flag clear trigger register mn(SIRmn)

This is a trigger register used to clear each error flag for channel n.

If each bit (FECTmn, PECTmn, OVCTmn) is set to "1", the corresponding bits (FEFmn, PEFmn, OVFmn) of the serial status register mn (SSRmn) are cleared to "0". Since SDIRmn register is a trigger register, if the corresponding bit of SSRmn register is cleared, SDIRmn register will also be cleared immediately.

The SIRmn register is set by a 16-bit memory manipulation instruction.

It is possible to set the lower 8 bits of the SIRmn register with SIRmnL and by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of SIRmn register changes to "0000H".

Table 12-11: Format of serial flag clear trigger register mn (SIRmn)

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	PECTmn	OVCTmn

FECTmn[Note1]	Channel n frame error flag clear trigger
0	No clearance.
1	Clear the FEFmn bit of the SSRmn register to "0".

PECTmn	Channel n parity error flag clear trigger
0	No clearance.
1	Clear the PEFmn bit of the SSRmn register to "0".

OVCTmn	Channel n overflow error flag clear trigger
0	No clearance.
1	Clear the OVFmn bit of the SSRmn register to "0".

Note 1: Limited to SIR01, SIR11 registers.

Notice: Bits 15~3 (SIR00, SIR10 registers are bit15~2) must be set to "0".

Remark:

1. m: unit number (m=0, 1);
   n: channel number (n=0, 1);
2. The read value of SIRmn register is always "0000H".

12.3.7 Serial status register mn (SSRmn)

The SSRmn register indicates the communication status of channel n and the condition in which an error occurred. Errors represented are frame errors, parity errors, and overflow errors. Read the SSRmn registers via a 16-bit memory manipulation instruction.

It can read the lower 8 bits of the SSRmn register with SSRmnL and read the SSRmn register through the 8-bit memory manipulation instruction.

After generating a reset signal, the value of the SSRmn register changes to "0000H".

Table 12-12: Format of serial status register mn (SSRmn) (1/2)

	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	PEFmn	OVFmn

TSFmn	Indication flag for channel n communication status
0	Communication stop state or communication standby state
1	Communication operation status
[Clear Condition] • When STmn of STm register is set to "1" (communication stopped state) or SSmn bit of SSm register is set to "1" (communication standby state) • When the communication ends [Set Condition]. • When communication begins	

BFFmn	Indication flag for channel n buffer register
0	The SDRmn register does not hold valid data.
1	The SDRmn register holds valid data.
[Clear Condition] • When the transmit data from the SDRmn register to the shift register is completed during the send process • When the received data is read from the SDRmn register during the receive process • When the STmn bit of STmn register is set to "1" (communication stopped state) or the SSm bit of the SSm register is set to "1" (Communication enable state) [Set Condition] • When writing and transmitting data to the SDRmn register in the state where the TXEmn bit of the SCRmn register is "1" (the transmit mode, transmit and receive mode in each communication mode). • When saving the receive data to the SDRmn register in the state where the RXEmn bit of the SCRmn register is "1" (receive mode, transmit and receive mode in each communication mode). • When a receive error occurs	

Note 1: Limited to SSR01, SSR11 registers.

Remark: If the SDRmn register is written while the BFFmn bit is "1", the saved transmit or receive data is discarded and an overflow error is detected (OVEmn=1).

m: m: unit number (m=0, 1) n: channel number (n=0, 1)

Table 12-12: Format of serial status register mn (SSRmn) (2/2)

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	PEFmn	OVFmn

FEFmn[Note1]	Detection flag for channel n frame errors
0	No errors occurred.
1	An error occurred (when the UART was received).
[Clear Condition]. • When writing "1" to the FECTmn bit of the SIRmn register [Set Condition]. • When no stop bit is detected at the end of UART reception	

PEFmn	Detection flag for channel n parity errors
0	No errors occurred.
1	An error occurred (when the UART was received)
[Clear Condition]. • When the PECTmn bit of the SIRmn register is written to "1" [Set Condition]. • When the parity and parity bits of the sent data are different at the end of the UART reception (parity error).	

OVFmn	Detection flag for channel n overflow error
0	No errors occurred.
1	An error occurred.
[Clear Condition]. • When writing "1" to the OVCTmn bit of the SIRmn register [Set Condition]. • In the state where the RXEmn bit of the SCRmn register is "1" (receive mode, transmit and receive mode in each communication mode), although the received data is saved in the SDRmn register, But when the received data is not read and the sending data is written or written down the next received data. • When data is not ready to be sent during a slave send in SSPI mode or during a slave send and receive	

Note 1: Limited to SSR01, SSR11 registers.

m: unit number (m=0, 1);

n: channel number (n=0, 1).

12.3.8 Serial channel start register m(SSm)

The SSm register is a trigger register to set the enable communication/start count for each channel.

If a "1" is written to each bit (SSmn), the corresponding bit (SEmn) in the serial channel enable status register m (SEm) is set to a "1" (operation enable status). Since the SSmn bit is a trigger bit, the SSmn bit is cleared immediately if the SEmn bit is a "1".

The SSm register is set by a 16-bit memory manipulation instruction.

It is possible to use SSmL and set the lower 8 bits of SSm register by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of SSm register changes to "0000H".

Table 12-13: Format of serial channel start register m (SSm)

	After reset							0000H						R/W		
Symbol	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	0	0	SSm1	SSm0

SSmn	Trigger at the beginning of channel n operation
0	No triggering.
1	Set the SEmn bit "1" and shift to communication standby state Note.

Note: If the SSmn bit is set to "1" during communication, communication will be stopped and enter standby state. At this time, the values of control register and shift register, SCLKmn pin and SDOmn pin, FEFmn flag, PEFmn flag and OVFmn flag remain in state.

Notice: Bits 15~2 of SSm register must be set to "0".

Remark:

1. m: unit number (m=0, 1);
   n: channel number (n=0, 1);
2. The read value of SSm register is always "0000H".

12.3.9 Serial channel stop register m(STm)

The STm register isa trigger register for setting to enable communication/stop count for each channel.

If a "1" is written to each bit (STmn), the corresponding bit (SEmn) in the serial channel enable status register m (SEm) is cleared to "0" (stop status). Since the STmn bit is a trigger bit, if the SEmn bit is "0", the STmn bit is cleared immediately.

The STm register is set by a 16-bit memory manipulation instruction.

The lower 8 bits of the STm register can be set using STmL and by an 8-bit memory manipulation instruction.

After the reset signal is generated, the value of STm register changes to "0000H".

Table 12-14: Format of serial channel stop register m (STm)

	After reset							0000H						R/W		
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STmn	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ST01	ST00

STmn	Stop trigger for channel n operation
0	No triggering.
1	Clear the SEmn bit to "0" to stop the communication operation[Note].

Note: The control register and shift register values, the SCLKmn pin and SDOmn pin, and the FEFmn flag, PEFmn flag, and OVFmn flag hold status.

Notice: Bits 15~2 of STm register must be set to "0".

Remark:

1. m: unit number (m=0, 1);
   n: channel number (n=0, 1);
2. The read value of STm register is always "0000H".

12.3.10 Serial channel enable status register m (SEm)

The SEm register is used to confirm the allow or stop status of serial transmit and receive for each channel.

If "1" is written to each of the serial start allow register m (SSm), the corresponding bit is set to "1". If you write "1" to each bit of the serial channel stop register m (STm), the corresponding bit is cleared to "0".

For channel n, which is allowed to run, the value of the CKOmn bit (serial clock output of channel n) of the serial output register m (SOm), described later, cannot be rewritten by software, and the value reflected by the communication run is output from the serial clock pin.

The value of the CKOmn bit in the SOm register can be set by software for channel n that is stopped, and the value is output from the serial clock pin. Thus, arbitrary waveforms such as start conditions or stop conditions can be generated by software.

The SEm register is read by a 16-bit memory manipulation instruction.

The lower 8 bits of the SEm register can be read with SEmL and by 8-bit memory manipulation instructions.

After the reset signal is generated, the value of SEm register becomes "0000H".

Table 12-15: Format of serial channel enable status register m (SEm)

	After reset							0000H						R		
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEmn	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SE01	SE00

SEmn	Indication of the enable or stop state of channel n operation
0	Run stop state
1	Run enable state

Remark: m: unit number (m=0, 1);

n: channel number (n=0, 1).

12.3.11 Serial output enable register m (SOEm)

The SOEm register setting enable or stops the output of serial communication for each channel.

For channel n that enable serial output, the value of the SOmn bit of the serial output register m (SOm) described below cannot be rewritten by software, but the value reflected by the communication operation is output from the serial data output pin.

For channel n that stops the serial output, the value of the SOMn bit of the SOm register can be set by software and output from the serial data output pin. Thus, arbitrary waveforms such as start conditions or stop conditions can be generated by software.

The SOEm register is set by the 16-bit memory manipulation instruction.

I can set the low 8 bits of the SOEm register with SOEmL and through the 8-bit memory manipulation instruction.

After generating a reset signal, the value of the SOEm register changes to "0000H".

Table 12-16: Format of serial output enable register m (SOEm)

	After reset							0000H						R/W	
Symbol	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	0	SOEm1	SOEm0

SOEmn	Channel n serial output enable or stop
0	Stop the output of serial communication.
1	Enable the output of serial communication.

Remark: Bits 15~2 of SOEm register are set to "0".

m: unit number (m=0, 1);

n: channel number (n=0, 1).

12.3.12 Serial output register m (SOm)

The SOm register is a buffer register for the 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 allowed (SOEmn=1), the value of the SOmn bit of this register can only be changed by serial communication, ignoring software rewriting.

The CKOmn bit of this register can be rewritten by software only when the channel is stopped (SEmn=0). When the channel is allowed to run (SEmn=1), software rewriting is ignored and the value of the CKOmn bit in this register can only be changed via serial communication.

To use the pins of the serial interface for non-serial interface functions such as port functions, the corresponding CKOmn bit and SOmn bit must be set to "1".

The SOm register is set by a 16-bit memory manipulation instruction.

After the reset signal is generated, the value of SOm register changes to "0303H".

Table 12-17: Format of serial output register m (SOm)

	After reset						0303H							R/W		
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOm	0	0	0	0	0	0	CKOm1	CKOm0	0	0	0	0	0	0	SOm1	SOm0

CKOmn	Serial clock output for channel n
0	The output value of the serial clock is "0".
1	The output value of the serial clock is "1".

SOmn	Serial data output for channel n
0	The output value of the serial data is "0".
1	The output value of the serial data is "1".

Notice: Bits 15~10 and bits 7~2 of SOm register must be set to "0".

Remark: m: unit number (m=0, 1);

n: channel number (n=0, 1).

12.3.13 Serial output level register m (SOLm)

The SOLm register is a register for setting the data output level inversion for each channel.

This register can be set only in UART mode. In SSPI mode, the corresponding bits must be set to "0". Only when serial output is allowed (SOEmn=1), set the inverse of each channel n of this register to reflect to the pin output. When serial output is disabled (SOEmn=0), the value of the SOmn bit is output directly. It is prohibited to rewrite the SOLm register during operation (SEmn=1).

The SOLm register is set by a 16-bit memory manipulation instruction.

It is possible to set the lower 8 bits of SDOLm register with SOLmL and by 8-bit memory manipulation instruction.

After the reset signal is generated, the value of SOLm register changes to "0000H".

Table 12-18: Format of the serial output level register m (SOLm)

	After reset							0000H						R/W		
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOLm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOLm0

SOLmn	Selection of channel n transmit data level inversion in UART mode
0	Output the communication data directly.
1	Invert the communication data to output.

Notice: Bits 15~1 of SOL0, SOL1 and SOL2 registers must be set to "0".

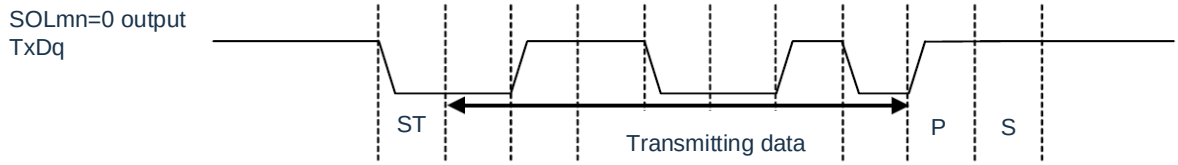
Remark: m: unit number (m=0, 1);

n: channel number (n=0).

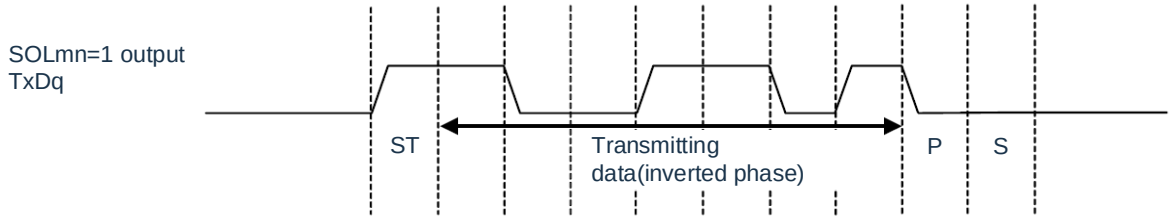
When UART transmission is performed, an example of level inversion of the transmitted data is shown in Figure 12-2.

Figure 12-2: Example of level inversion for transmitting data

(a) Normal phase output(SOLmn=0)



(b) Inverted output (SOLmn=1)



Remark: m: unit number (m=0, 1);

n: channel number (n=0).

12.3.14 Input switching control register (ISC)

When LIN-bus communication is implemented via UART0, the ISC1 bits and ISC0 bits of the ISC registers are used for coordination of external interrupts and timer array units. If bit0 is placed at "1", the input signal of the serial data input (RxD0) pin is selected as the input to the external interrupt (INTP0), so it can pass THE INTP0 interrupt detects the wake-up signal.

If bit1 is set to "1", the input signal of the serial data input (RxD0) pin is selected as the input to the timer, so the wake signal can be detected by the timer and the low width of the break field and the pulse width of the synchronization field can be measured.

The SSIE00 bit controls the SS00 pin input of channel 0 in the slave mode of SSPI00 communication. During the period when a high level is input to the SS00 pin, transmission and reception are not performed even if a serial clock is input; during the period when a low level is input to the SS00 pin, transmission and reception are performed according to the setting of each mode if a serial clock is input.

The ISC register is set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of the ISC register becomes "00H".

Table 12-19: Table of input switching control register (ISC)

Address: 40040473H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
ISC	SSIE00	0	0	0	0	0	ISC1	ISC0

SSIE00	Setting of SS00 input for channel 0 in slave mode of SSPI00 communication
0	SS00 pin input is invalid.
1	SS00 pin input is valid.

ISC1	Input switching of channel 3 of Timer4
0	Use the input signal from the TI03 pin as an input to the timer (normally operating).
1	Use the input signal on the RxD0 pin as a timer input (detects the wake-up signal and measures the low-level width of the break field and the pulse width of the sync field).

ISC0	Input switching of external interrupt (INTP0)
0	Use the input signal on the INTP0 pin as an external interrupt input (normally operating).
1	Use the input signal on the RxD0 pin as an external interrupt input (detecting a wake-up signal).

Notice: Bits 6~2 must be set to "0".

12.3.15 Noise filter enable register 0 (NFEN0)

The NFEN0 register sets whether the noise filter is used for the input signal of each channel's serial data input pin.

For pins used for SSPI communication, the corresponding bit must be "0" to invalidate the noise filter. For pins used for UART communication, the corresponding bit "1" must be set to make the noise filter effective.

When the noise filter is active, the 2 clocks are detected to be consistent after synchronization through the operating clock (F_{MCK}) of the object channel; When the noise filter is invalid, synchronization is performed only through the operating clock (F_{MCK}) of the object channel.

The NFEN0 register is set by an 8-bit memory manipulation instruction.

After generating a reset signal, the value of the NFEN0 register changes to "00H".

Table 12-20: Format of noise filter enable register 0 (NFEN0)

Address:	40040470H	After reset:	00H			R/W	
Symbol	7	6	5	4	3	2	1	0
NFEN0	0	0	0	0	0	SNFEN10	0	SNFEN00

SNFEN10	RxD1 pin noise filter is used or not
0	Noise filter OFF
1	Noise filter ON
When used as the RxD1 pin, SNFEN10 must be set to "1". When used as a function other than the RxD1 pin, the SNFEN10 must be set to "0".	

SNFEN00	RxD0 pin noise filter is used or not
0	Noise filter OFF
1	Noise filter ON
When used as the RxD0 pin, SNFEN00 must be set to "1". When used as a function other than the RxD0 pin, the SNFEN00 must be set to "0".	

Notice: Bits 7, 5, 3, 1 must be set to "0".

12.3.16 Registers controlling port functions of serial input/output pins

When using a general-purpose serial communication unit, the control registers for the multiplexed port function (Port Mode Register (PMxx), Port Multiplexing Function Configuration Register (PxxCFG), Port Output Mode Register (POMxx), and Port Mode Control Register (PMCxx) must be set).

For details, please refer to “Chapter 2 Port Function”.

When using the multiplexed port of the serial data output pin or the serial clock output pin as the serial data output or serial clock output, the bits of the corresponding port mode control register (PMCxx) and the bit of the port mode register (PMxx) corresponding to each port are “0”. In this case, the bit of the port register (Pxx) can be “0” or “1”.

In addition, when used for N-channel open-drain output mode, the bit of the port output mode register (POMxx) corresponding to each port must be “1”.

When using the multiplexed port of the serial data input pin or serial clock input pin as serial data input or serial clock input, you must set the bit of the Port Mode Register (PMxx) corresponding to each port to "1" and set the bit of the Port Mode Control Register (PMCxx) to "0". In this case, the Port Register (Pxx) bits can be "0" or "1".

12.4 Operation stop mode

Each serial interface of a general-purpose serial communication unit has an operation stop mode. Serial communication is not possible in operation stop mode, so power consumption is reduced. In addition, pins for the serial interface can be used as port functions in operation stop mode.

12.4.1 Stopping the operation by units

The unit stop is set by peripheral enable register 0/2 (PER0/2).

Per0/2 registers are registers that are set to enable or disable clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocks to unused hardware.

To stop general-purpose serial communication unit 0, the bit2 (SCI0EN) of PER0 must be set to "0"; To stop general-purpose serial communication unit 1, the bit3 (SCI1EN) of PER0 must be set to "0"

Peripheral enable register 0 (PER0) …… Set "0" to the bit corresponding to the bit where SCIm is to be stopped.

Table 12-21: Setting of peripheral enable register 0 (PER0) when stopping operation by unit

Symbol	7	6	5	4	3	2	1	0
PER0	RTCEN	0	ADCEN	IICAEN	SCI1EN	SCI0EN	TM41EN	TM40EN

SCImEN	Provides control of the input clock of the general-purpose serial communication unit m.
0	Stop to supply the input clock. • The SFR used by the general-purpose serial communication unit m cannot be written. • The general-purpose serial communication unit m is in the reset state.
1	Enables an input clock to be supplied. • Can read and write SFRs used by the general-purpose serial communication unit m.

Remark:

1. When the SCImEN bit is "0", the write operation of the control register of the general-purpose serial communication unit m is ignored, and the read values are initial values. However, the following registers are excluded:
   ① Input switching control register (ISC)
   ② Noise filter enable register (NFEN0)
   ③ Port multiplexing function configuration register (PxxCFG)
   ④ Port output mode register (POMx)
   ⑤ Port output mode register (PMx)
   ⑥ Port mode register (Px)
2. ×: Bits are not used by the general-purpose serial communication units (depending on the setting of other peripheral functions).
   0/1: Set "0" or "1" depending on the user's purpose.

12.4.2 Stopping the operation by channels

The stopping of the operation by channels is set using each of the following registers.

(1) Serial channel stop register m (STm) … This register is a trigger register that is used to enable communication/stop count by each channel.

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	STm1	STm0

① Clears the SEmn bit to 0 and stops the communication operation

② Because the STmn bit is a trigger bit, it is cleared immediately when SEmn = 0.

(2) Serial channel enable status register m (SEm) … This register indicates whether data transmission/reception operation of each channel is enabled or stopped.

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SEm1	SEm0

① 0: Operation stop state

② 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.

(3) 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.

Symbol	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	0	SOE01	SOE00

① 0: Stops output by serial communication operation

② The value of the SOmn bit of the SOm register can be set by software for channels where serial output has been stopped.

(4) Serial output register m (SOm) …This register is a buffer register for serial output of each channel.

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOm	0	0	0	0	0	0	CKOm1	CKOm0	0	0	0	0	0	0	SOm1	SOm0

① CKOm1/0: 1: output value of the serial clock is "1"; 0: output value of the serial clock is "0"

② SOm1/0: 1: output value of the serial clock is "1"; 0: output value of the serial clock is "0"

③ When using the pin corresponding to each channel as a port function, the corresponding CKOmn bit and SOmn bit must be "1".

Remark: Limited to general-purpose serial communication unit 0.

m: unit number (m=0, 1);

n: channel number (n=0, 1);

0/1: Set “0” or “1” according to the user’s purpose.

12.5 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11) communication

This is a clock synchronization communication function implemented by three lines of serial clock (SCLK) and serial data (SDI and SDO).

[Data transmission and reception]

(1) Data length of 7~16 bits

(2) Phase control of transmit/receive data

(3) MSB/LSB preferred option

[Clock control]

(1) Master/slave selection

(2) Phase control of I/O clock

(3) Setting of transfer period by prescaler and internal counter

(4) Maximum transfer rate

During master communication: Max.F_{CLK}/2

During slave communication: Max.F_{MCK}/6

[Interrupt function]

Transfer end interrupt, buffer empty interrupt

[Error detection flag]

Overflow error

Channels 0~1 of SCI0 and channels 0~1 of SCI1 are channels that support 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11).

The 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11) has the following six communication operations.

(1) Master transmission (refer to 12.5.1)

(2) Master reception (refer to 12.5.2)

(3) Master transmission and reception (refer to 12.5.3)

(4) Slave transmission (refer to 12.5.4)

(5) Slave reception (refer to 12.5.5)

(6) Slave transmission and reception (refer to 12.5.6)

Notice: It must be used within the range that satisfies the SCLK cycle time (T_{KCY}) characteristics. Refer to the datasheet for details.

12.5.1 Master transmission

Master transmission refers to the operation of this product output transmission clock and sending data to other devices.

Table 12-22: Master transmission

3-wire serial I/O	SSPI00	SSPI01	SSPI10	SSPI11
Target channel	Channel 0 of SCI0	Channel 1 of SCI0	Channel 0 of SCI1	Channel 1 of SCI1
Pins used	SCLKOI00, SDO00	SCLKOI01, SDO01	SCLKOI10, SDO10	SCLKOI11, SDO11
Interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11
	Selectable transmission end interrupt (single transmission mode) or buffer empty interrupt (continuous transmission mode).			
Error detection flag	None			
Transfer data length	7~16 bits			
Transfer rate Note	Max.F_{CLK}/2[Hz] Min.F_{CLK}/(22^{11}128)[Hz] F_{CLK}: System clock frequency			
Data phase	It can be selected by the DAPmn bit of the SCRmn register. DAPmn=0: Start data output when the serial clock starts running. DAPmn=1: Start data output half a clock before the serial clock starts running.			
Clock phase	It can be selected by the CKPmn bit of the SCRmn register. CKPmn=0: Non-reverse CKPmn=1: Reverse			
Data direction	MSB or LSB first			

Note: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheret).

m: unit number (m=0, 1);

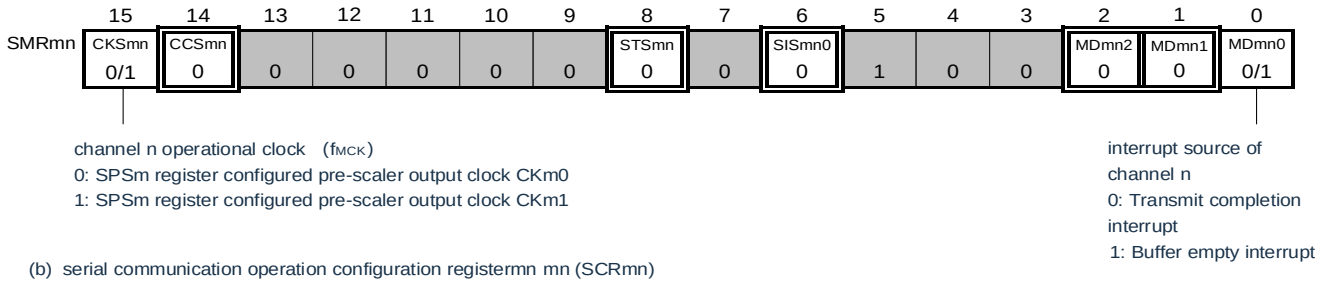
n: channel number (n=0, 1);

mn=00~01, 10~11.

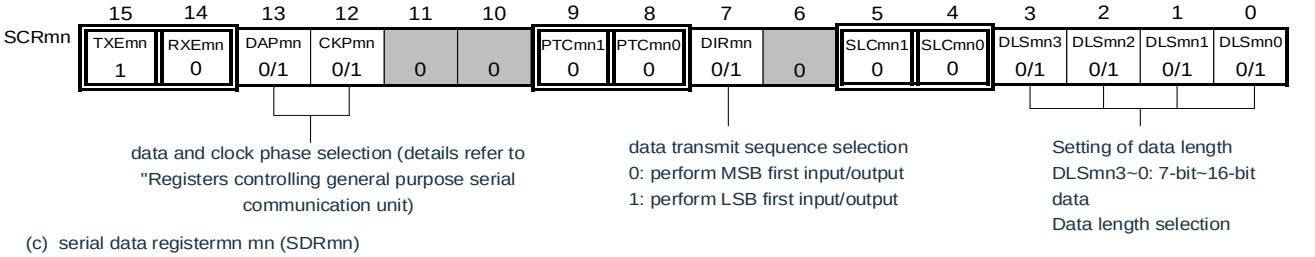
(1) Register setting

Figure 12-3: 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11)
Example of register setting content when the master transmits

(a) serial mode register mn (SMRmn)



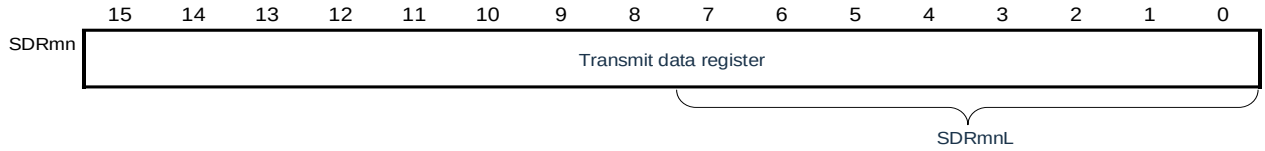
(b) serial communication operation configuration registermn mn (SCRmn)



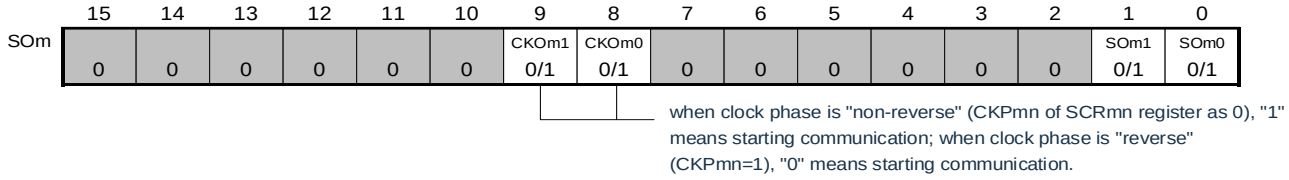
(c) serial data registermn mn (SDRmn)

(1) When operation stops (SEmn=0)

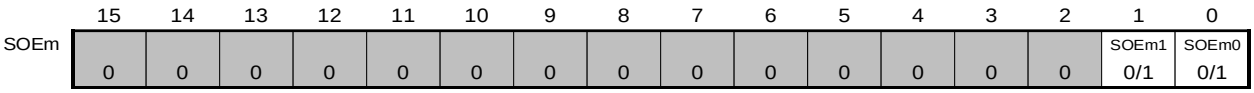
(2) During operation（SEmn=1）（lower 8 bits: SDRmnL）



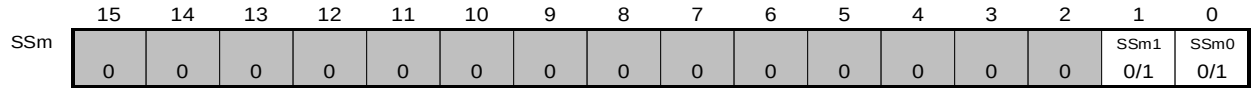
(d) serial output register m(SOm) .....Only configure bit of target channel



(e) serial output enable register m (SOEm)….only set bit of target channel to 1.



(f) serial channel start registerm (SSm)….only set bit of target channel to 1.



Remark: m: unit number (m=0, 1) n: channel number (n=0, 1) mn=00~01, 10~11

□ : Cannot be set (initial value is set). 0/1: Set “0” or “1” according to the user’s purpose.

(2) Procedure

Figure 12-4: Initial setup steps of master transmission

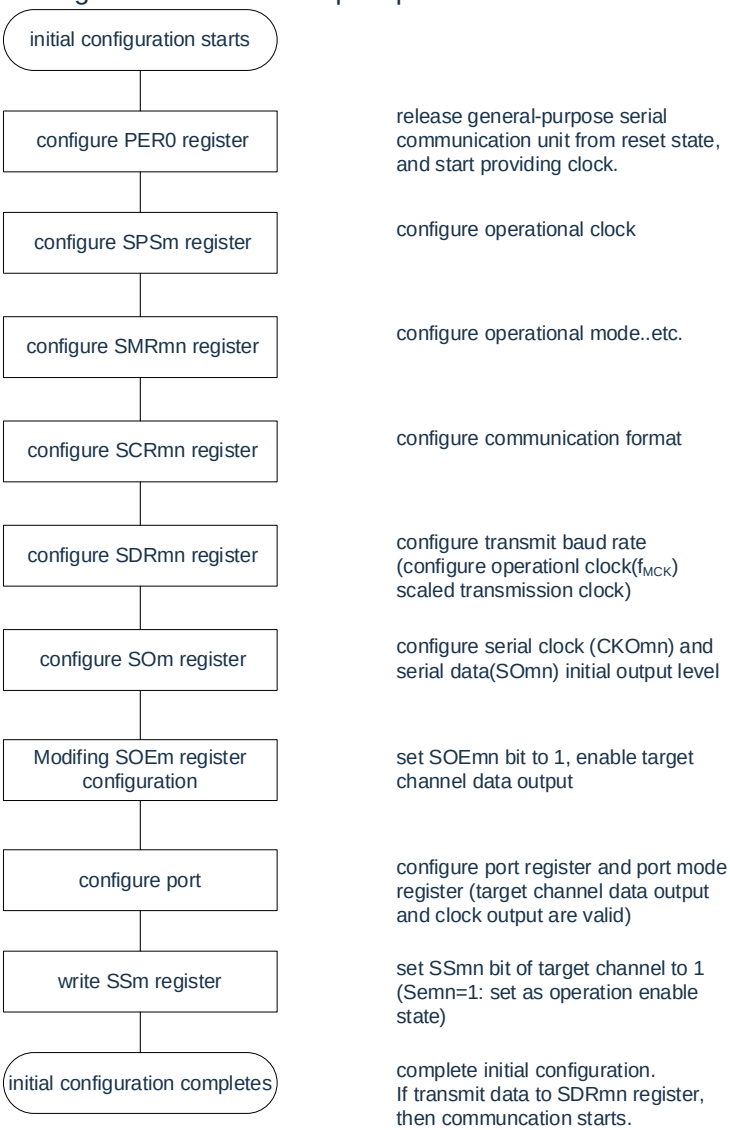


Figure 12-5: Stop steps of master transmission

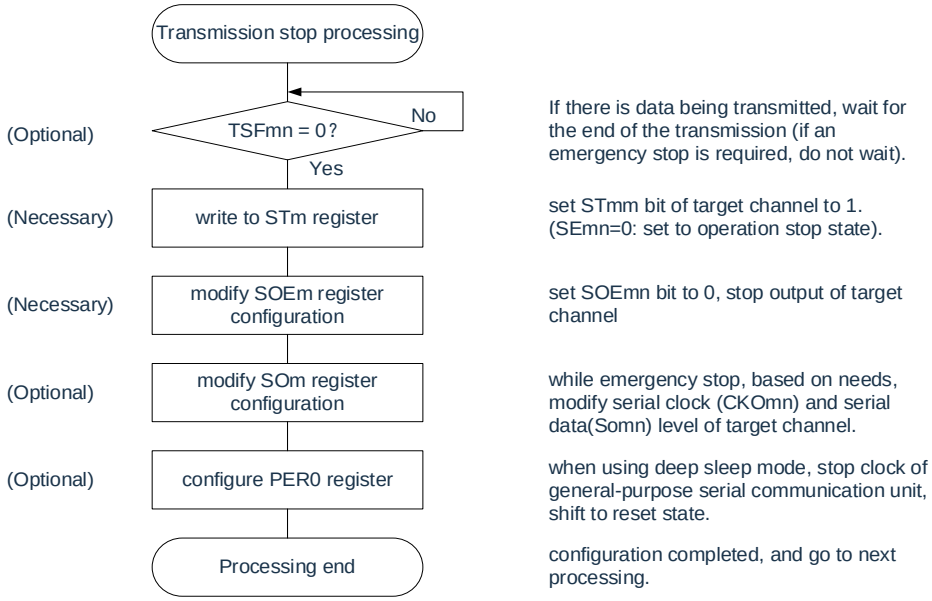
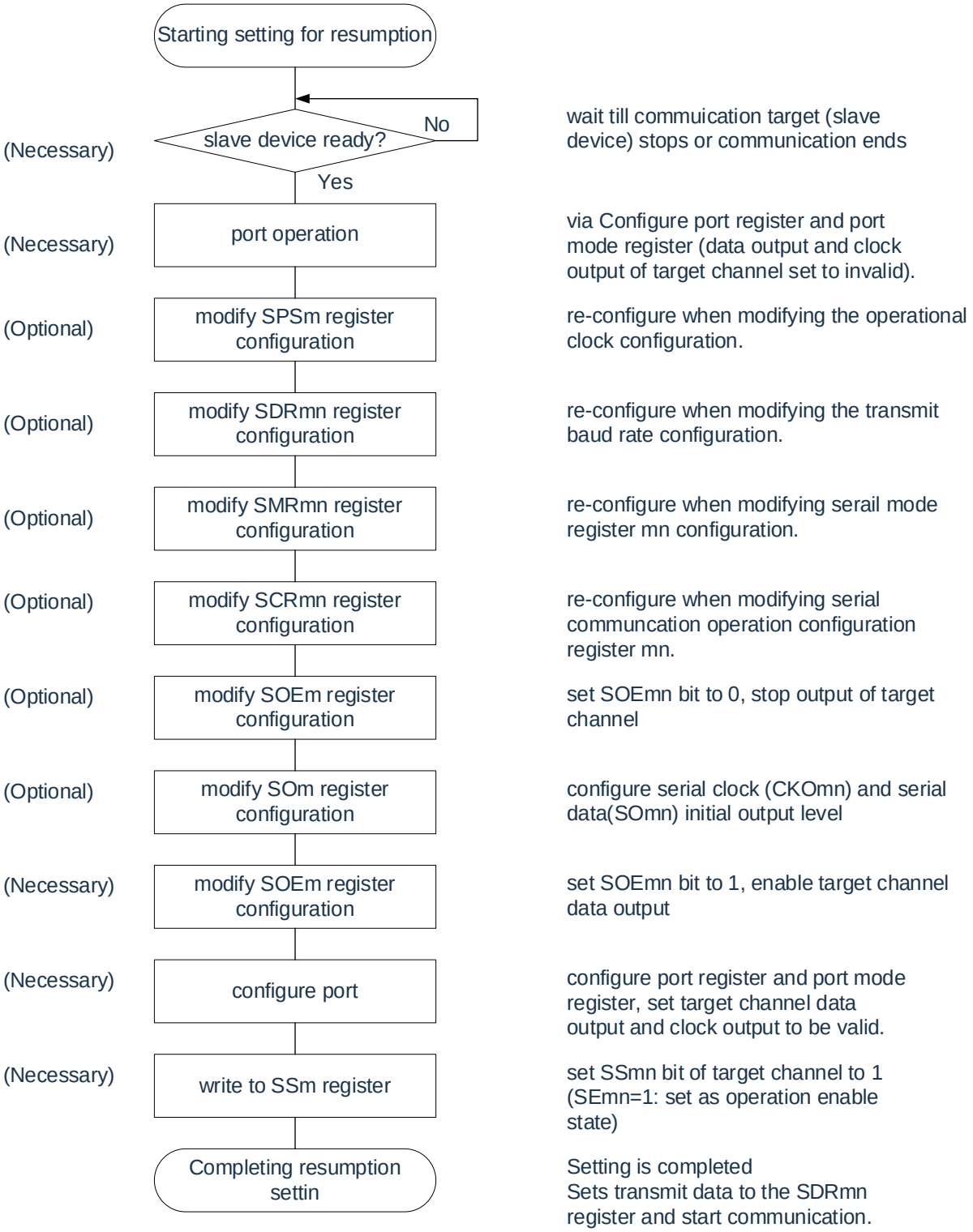


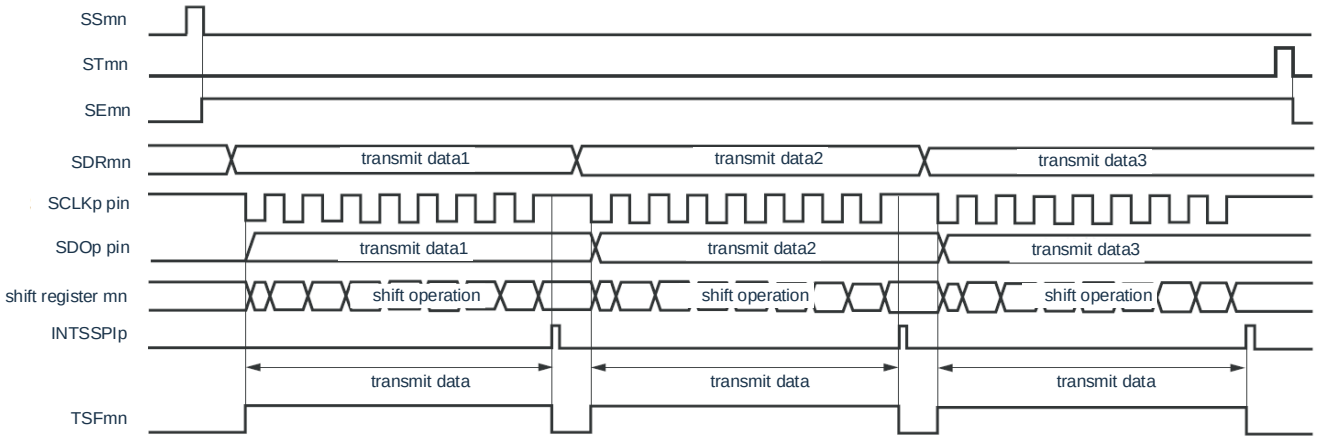
Figure 12-6: Restart steps of master transmission



Notice: If you override PER0 in the abort settings to stop providing the clock, you must make the initial settings instead of restarting them when the communication object (slave) stops or when the communication ends.

(3) Process flow (single transmit mode)

Figure 12-7: Timing diagram of master transmission (single send mode) (type 1: DAPmn= 0, CKPmn = 0)

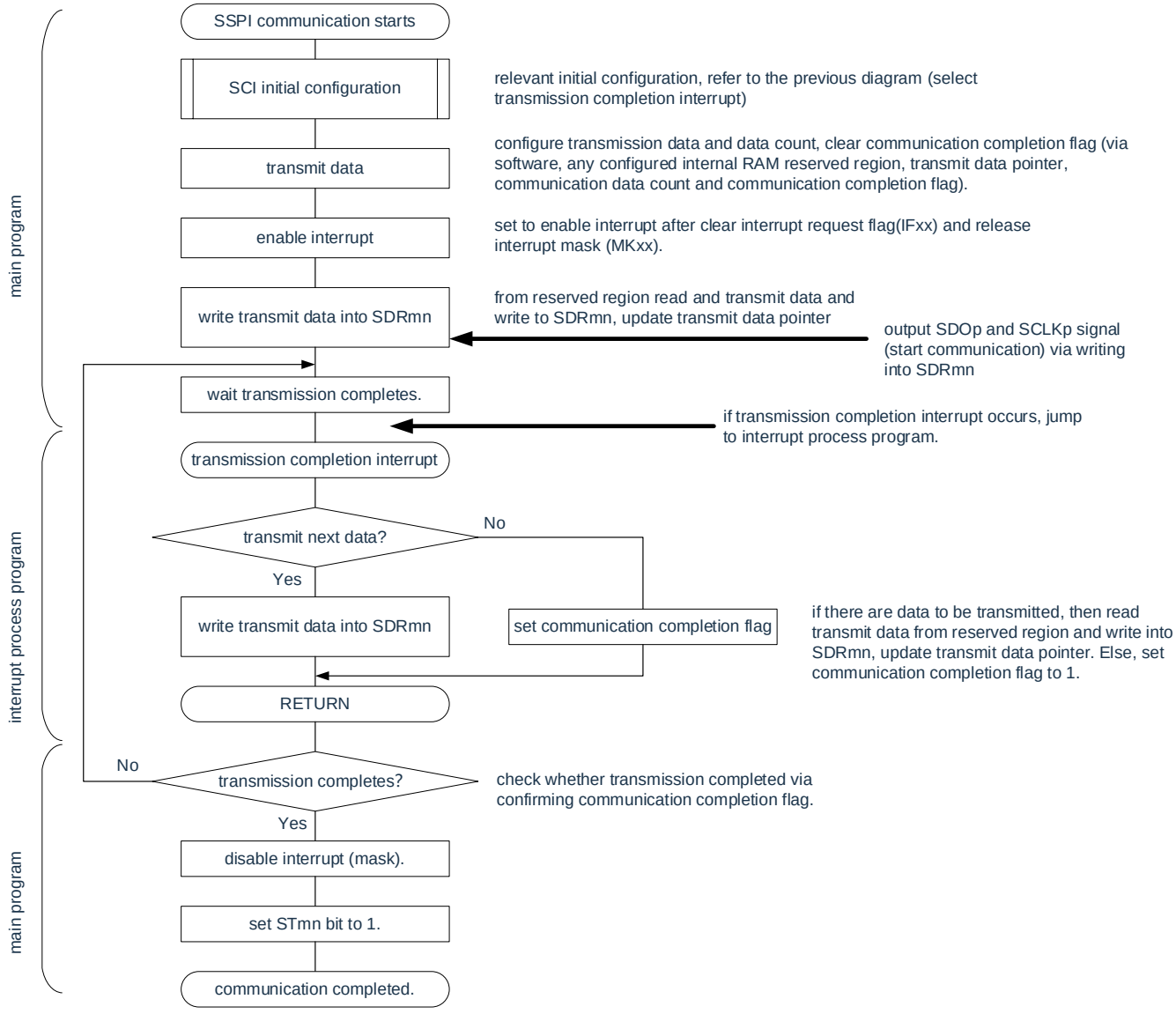


Remark: m: unit number (m=0, 1);

n: channel number (n=0, 1);

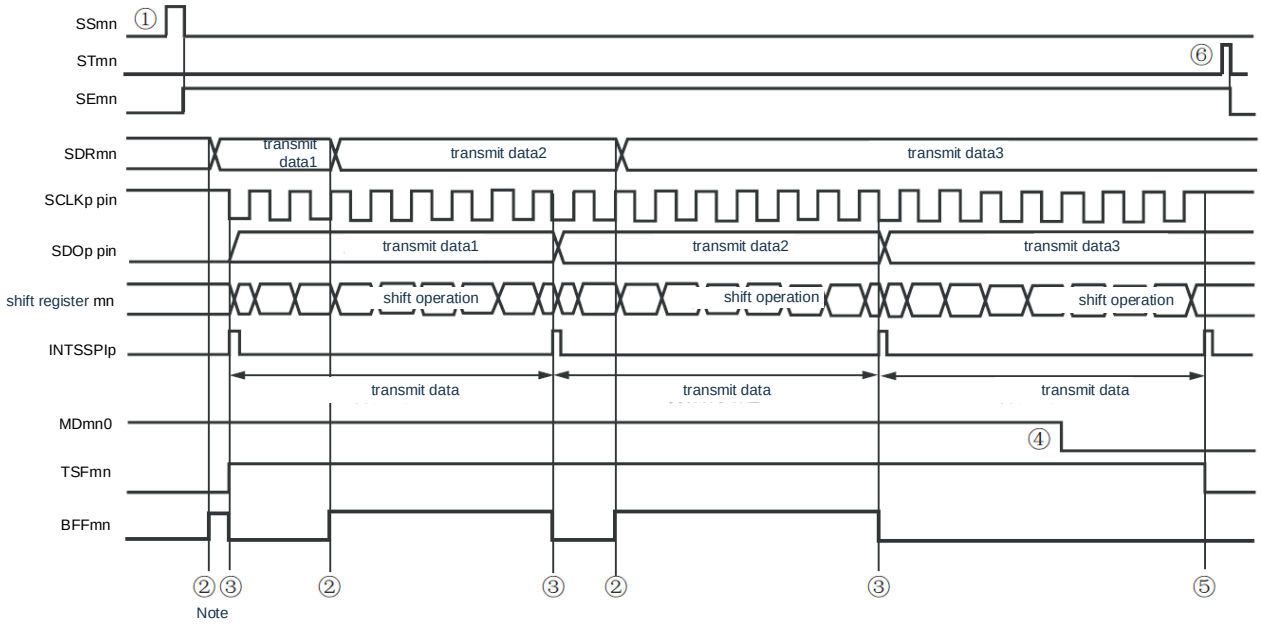
p: SSPI number (p=00, 01, 10, 11).

Figure 12-8: Flowchart of master transmission (single transmit mode)



(4) Process flow (continuous transmit mode)

Figure 12-9: Timing diagram of the master transmission (continuous transmit mode)
(type 1: DAPmn= 0, CKPmn = 0)

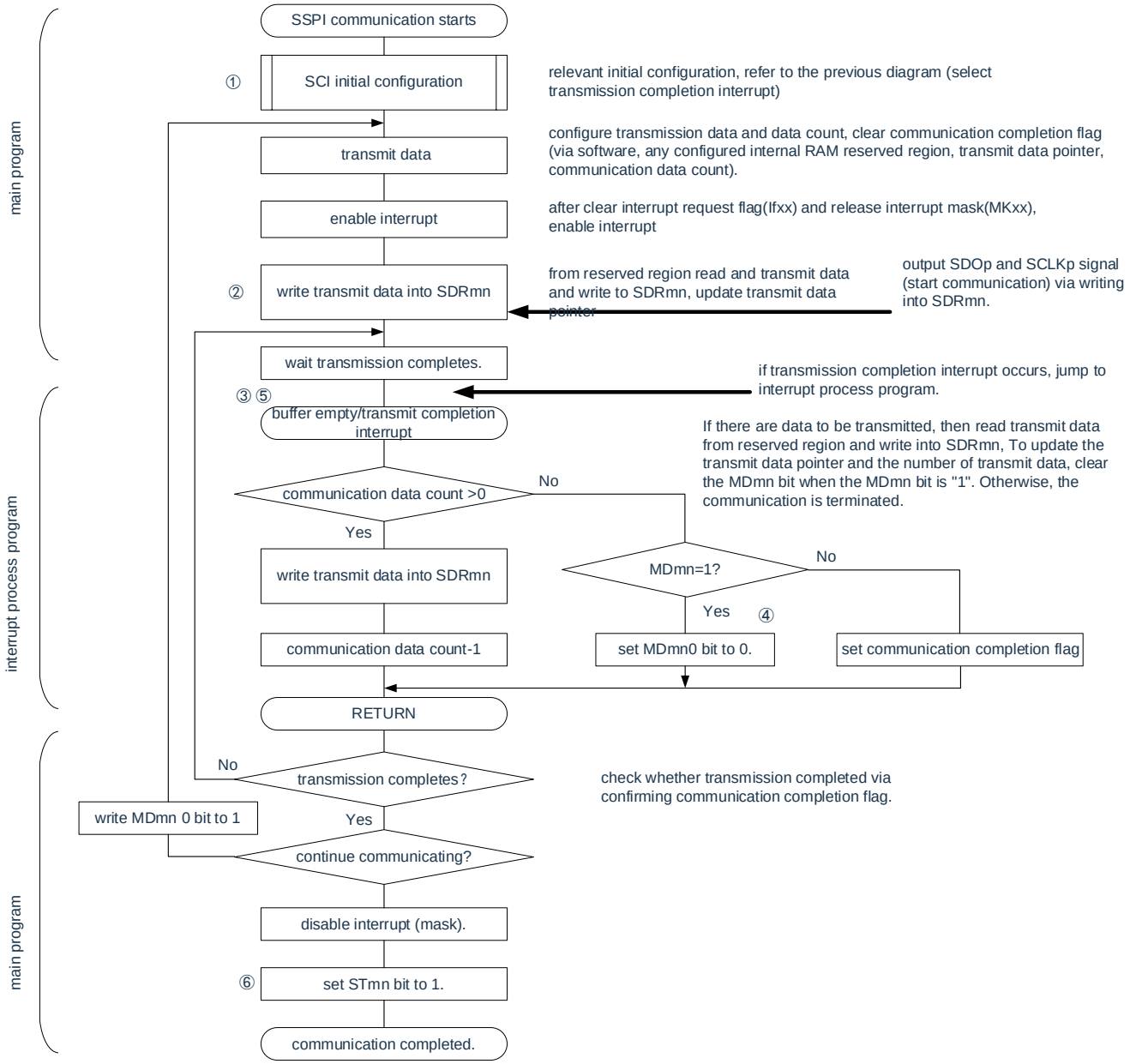


Note: If the transmit data is written to the SDRmn register during the time when the BFFmn bit of the serial status register mn (SSRmn) is “1” (when valid data is stored in the serial data register mn (SDRmn)), the transmit data is rewritten;

Notice: The MDmn0 bit of the serial mode register mn (SMRmn) can be rewritten even in operation. However, in order to catch the end-of-transmission interruption of the last sent data, it must be rewritten before the last bit of transmission begins.

Remark: m: unit number (m=0, 1);
n: channel number (n=0, 1);
p: SSPI number (p=00, 01, 10, 11).

Figure 12-10: Flowchart of master transmission (continuous transmit mode)



Remark:① to ⑥ correspond to ① to ⑥ in “Figure 12-9 Timing Diagram of Master Transmission (Continuous Transmission Mode)”.

12.5.2 Master reception

Master reception refers to the operation of this product output transmission clock and receiving data from other devices.

Table 12-23: Master reception

3-wire serial I/O	SSPI00	SSPI01	SSPI10	SSPI11
Target channel	Channel 0 of SCI0	Channel 1 of SCI0	Channel 0 of SCI1	Channel 1 of SCI1
Pins used	SCLKOI00, SDI00	SCLKOI01, SDI01	SCLKOI10, SDI10	SCLKOI11, SDI11
Interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11
	Selectable transmission end interrupt (single transmission mode) or buffer empty interrupt (continuous transmission mode).			
Error detection flag	Only the overflow error detection flag (OVFmn).			
Transfer data length	7~16 bits			
Transfer rate Note	Max.F_{CLK}/2[Hz] Min.F_{CLK}/(22^{11}128)[Hz] F_{CLK}: System clock frequency			
Data phase	It can be selected by the DAPmn bit of the SCRmn register. DAPmn=0: Start data output when the serial clock starts running. DAPmn=1: Start data output half a clock before the serial clock starts running.			
Clock phase	It can be selected by the CKPmn bit of the SCRmn register. CKPmn=0: Non-reverse CKPmn=1: Reverse			
Data direction	MSB or LSB first			

Notice: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

Remark: m: unit number (m=0, 1);

n: channel number (n=0, 1);

p: SSPI number (p=00, 01, 10, 11)

(1) Register setting

Figure 12-11: 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11)
Example of register setting content when the master receives

(a) serial mode register mn (SMRmn)

	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	0	0	0/1

channel n operational clock (f_{MCK})
0: SPSm register configured pre-scaler output clock CKm0
1: SPSm register configured pre-scaler output clock CKm1

interrupt source of channel n
0: Transmit completion interrupt
1: Buffer empty interrupt

(b) serial communication operation configuration registermn mn (SCRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn	RXEmn	DAPmn	CKPmn			PTCmn1	PTCmn0	DIRmn		SLCmn1	SLCmn0	DLSmn3	DLSmn2	DLSmn1	DLSmn0
	0	1	0/1	0/1	0	0	0	0	0/1	0	0	0	0/1	0/1	0/1	0/1

data and clock phase selection (details refer to "Registers controlling general purpose serial communication unit)

data transmit sequence selection
0: perform MSB first input/output
1: perform LSB first input/output

Setting of data length
DLSmn3~0: 7-bit~16-bit data
Data length selection

(c) serial data registermn mn (SDRmn)

(1) When operation stops (SEmn=0)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	Baud rate configuration (operation clock (F_{MCK}) scaling configuration)							0	0	0	0	0	0	0	0	0

(2) During operation（SEmn=1）（lower 8 bits: SDRmnL）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	Receive data register															

SDRmnL

(d) serial output register m(SOm) .....Only configure bit of target channel

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOm							CKOm1	CKOm0							SOm1	SOm0
	0	0	0	0	0	0	0/1	0/1	0	0	0	0	0	0	0/1	0/1

when clock phase is "non-reverse" (CKPmn of SCRmn register as 0), "1" means starting communication; when clock phase is "reverse" (CKPmn=1), "0" means starting communication.

(e) serial output enable register m (SOEm)….not used in this mode.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm															SOEm1	SOEm0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0/1	0/1

(f) serial channel start registerm (SSm)….only set bit of target channel to 1.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm															SSm1	SSm0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0/1	0/1

Remark: m: unit number (m=0, 1);

n: channel number (n=0, 1);

p: SSPI number (p=00, 01, 10, 11);

□: Fixed set in SSPI master receive mode■: Cannot be set (initial value is set);

×: This is a bit that cannot be used in this mode (sets the initial value if it is not used in other modes either);

0/1: Set “0” or “1” according to the user’s purpose.

(2) Procedure

Figure 12-12: Initial setup steps for master reception

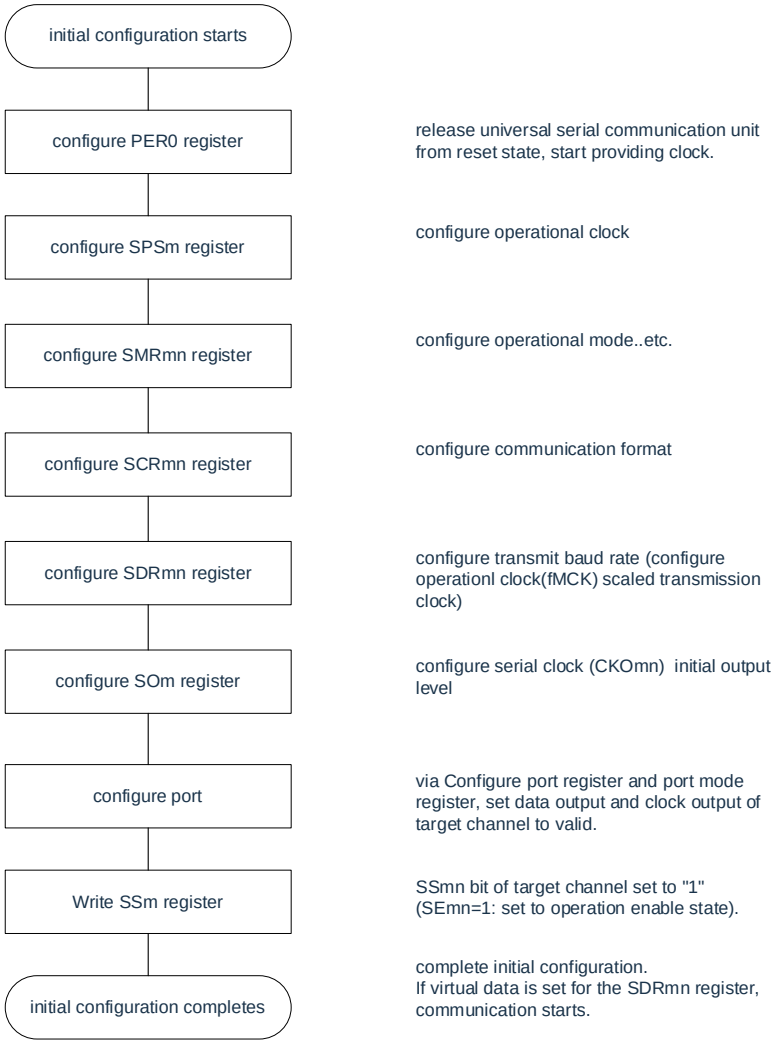


Figure 12-13: Stop steps for master reception

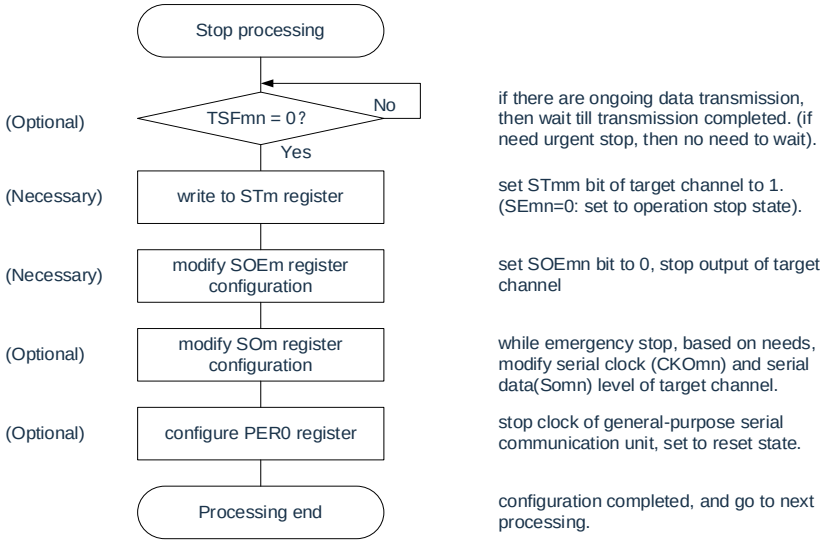
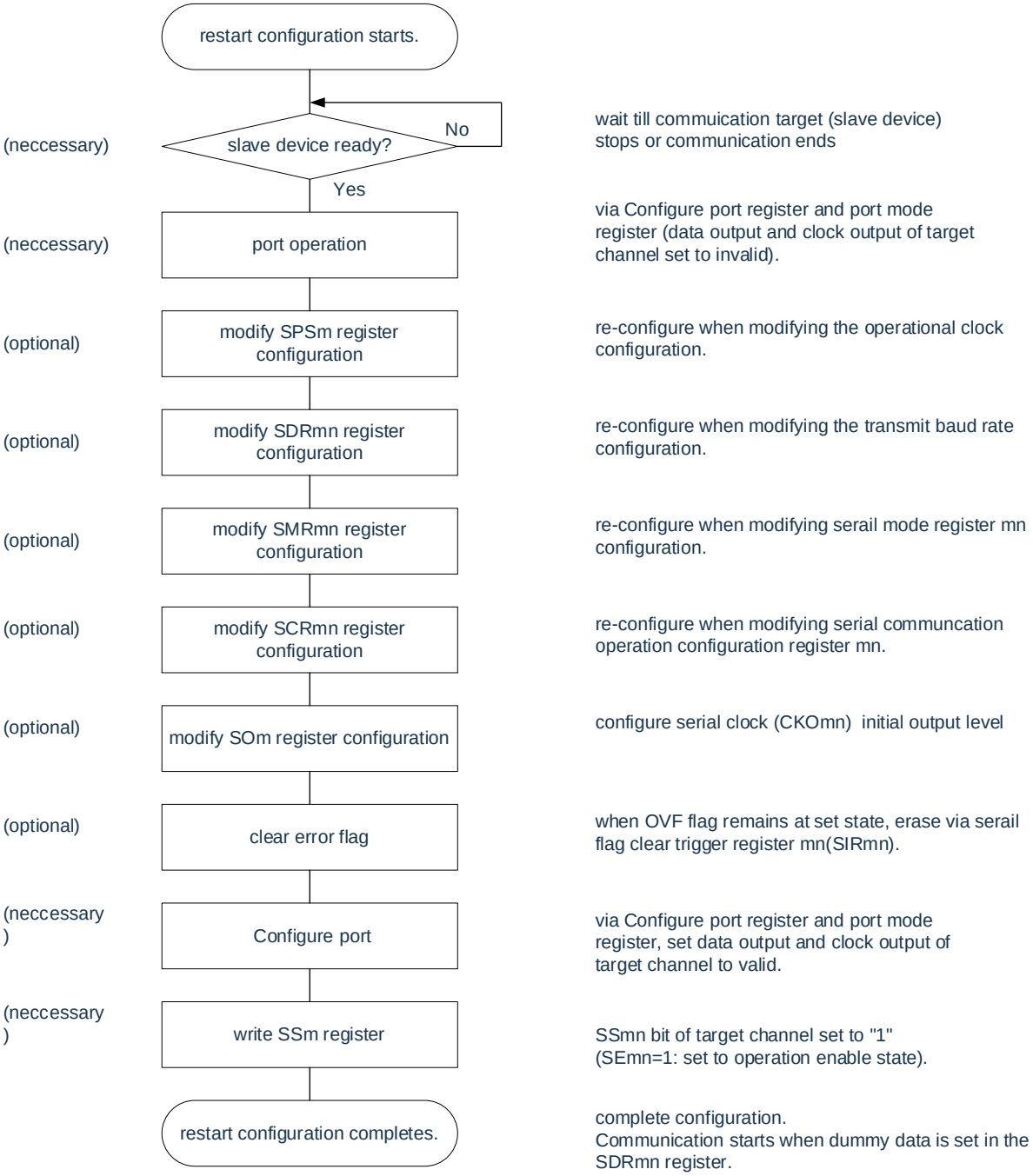


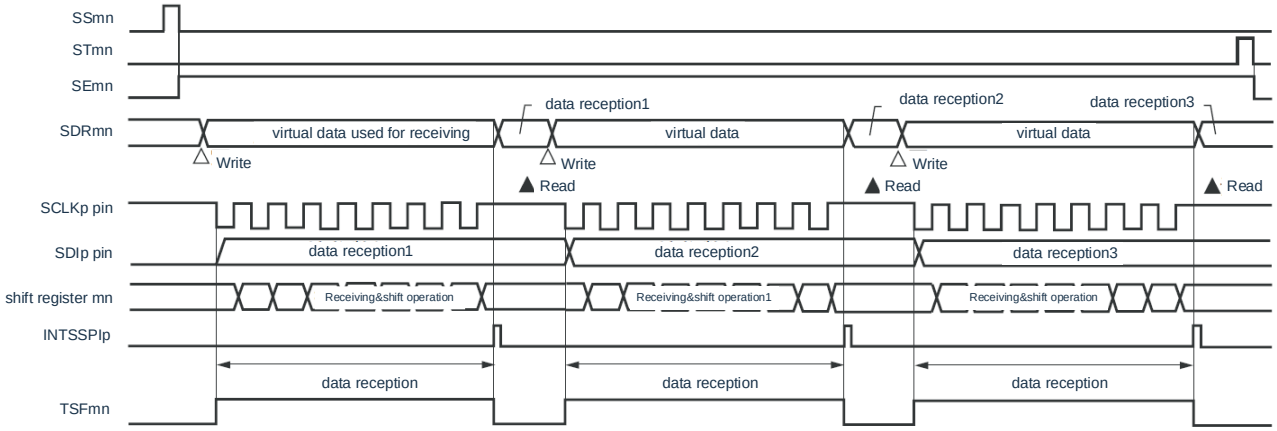
Figure 12-14: Restart steps for master transmission



Notice: If PER0 is rewritten in the abort setting to stop the clock from being supplied, it is necessary to wait until the communication object (slave device) stops or the communication is finished to perform the initial setting instead of restarting the setting.

(3) Process flow (single receive mode)

Figure 12-15: Timing diagram of the master receive (single receive mode) (type 1: DAPmn = 0, CKPmn = 0)

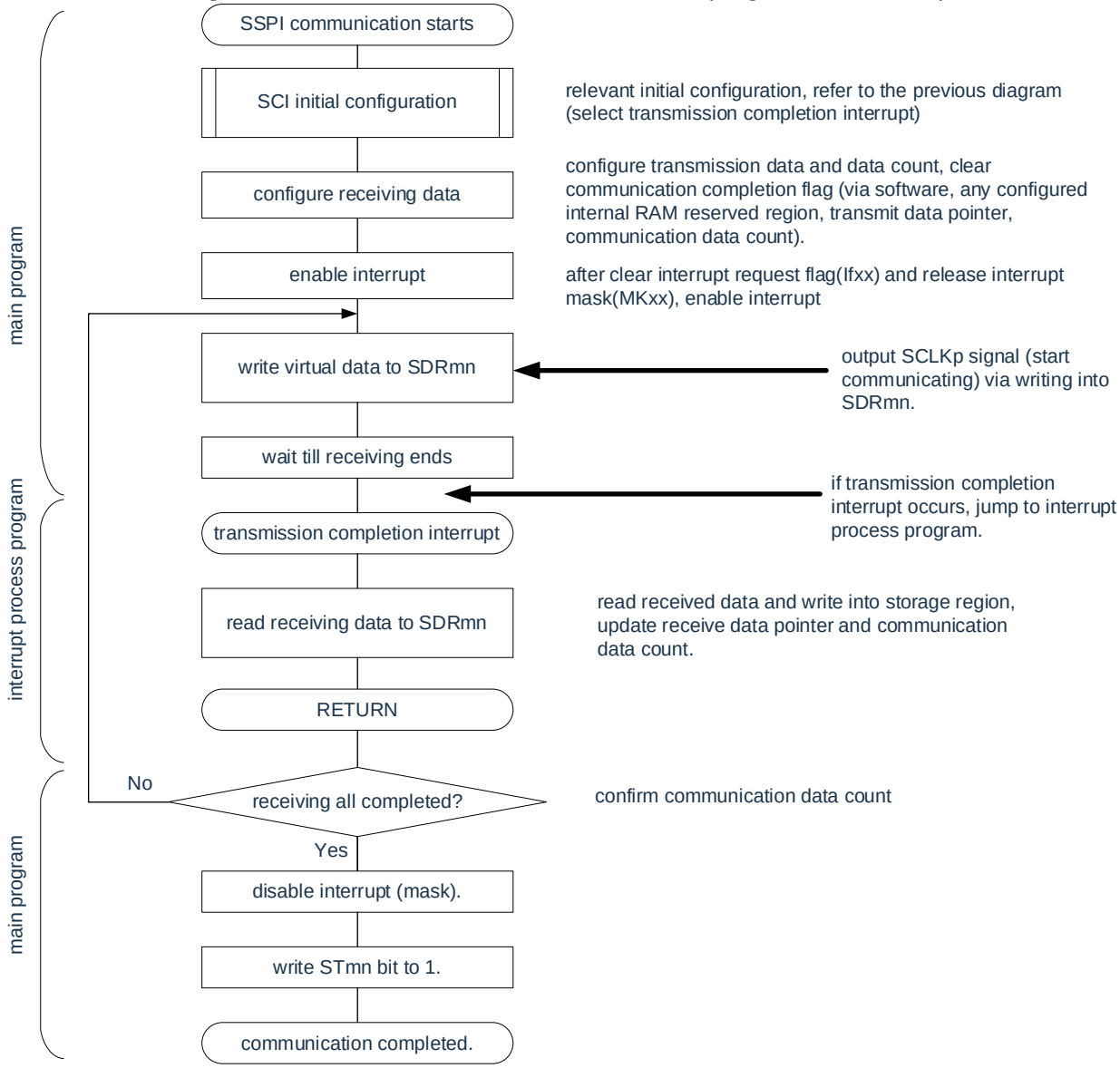


Remark: m: unit number (m=0, 1);

n: channel number (n=0, 1);

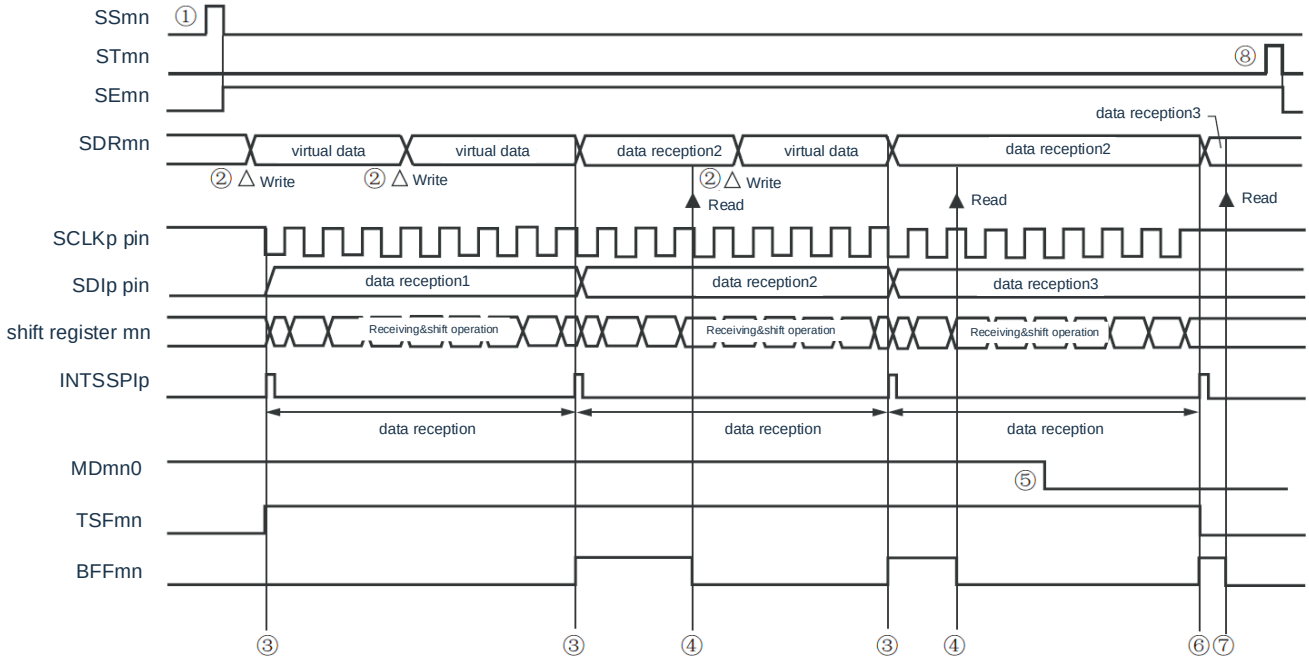
p: SSPI number (p=00, 01, 10, 11).

Figure 12-16: Flowchart of the master receive (single receive mode)



(4) Process flow (continuous receive mode)

Figure 12-17: Timing diagram of master receive (continuous receive mode) (type 1: DAPmn= 0, CKPmn = 0)

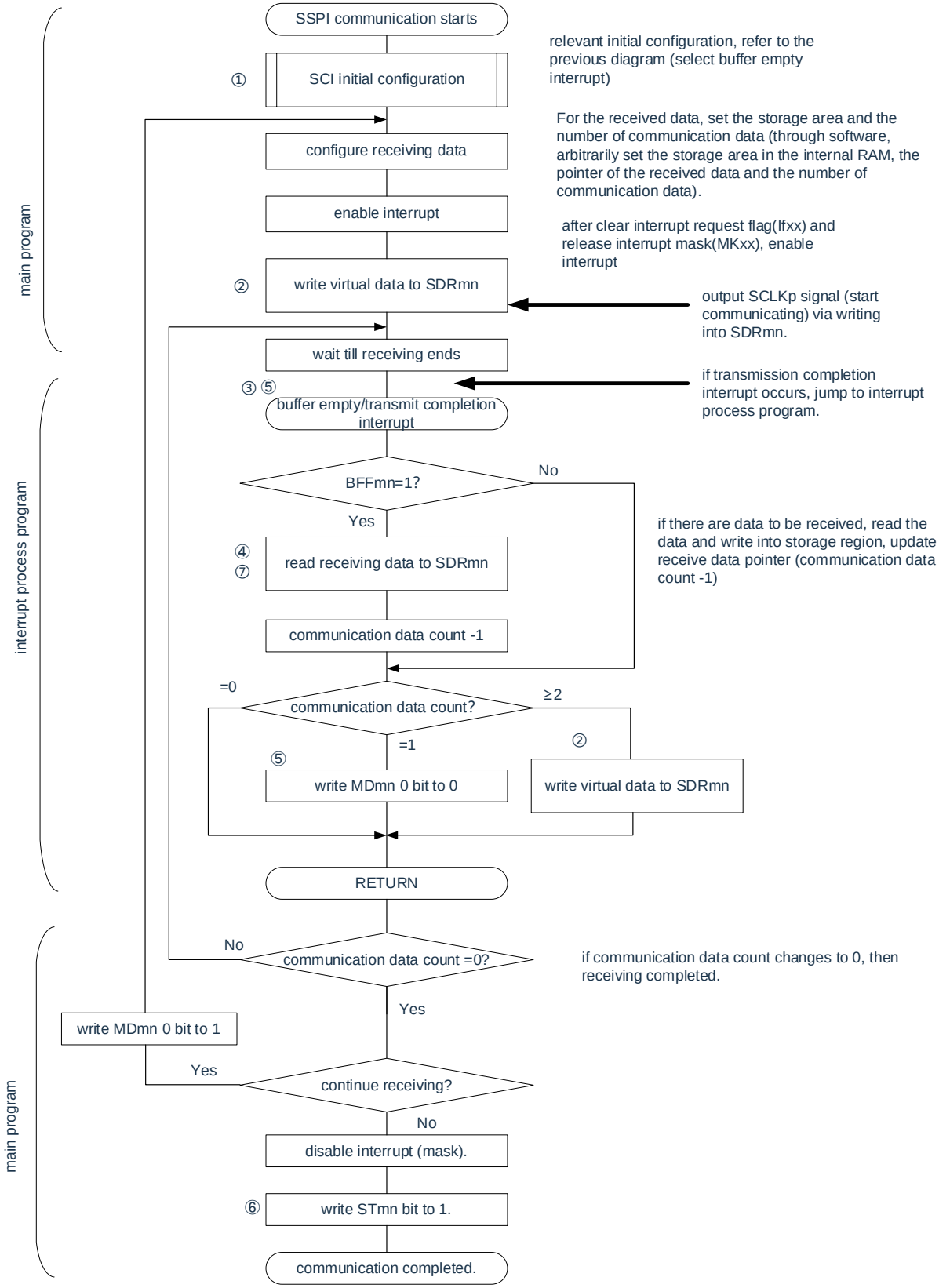


Notice: The MDmn0 bit can be rewritten even during operation. However, in order to catch up with the end-of-transmission interruption of the last received data, it must be rewritten before the last bit can be received.

Remark:

1. ①~⑧ in the figure correspond to ①~⑧ in “Figure 12-18: Flowchart of the master receiver (continuous receive mode)”.
2. m: unit number (m=0, 1);
   n: channel number (n=0, 1);
   p: SSPI number (p=00, 01, 10, 11).

Figure 12-18: Flowchart of master receiver (continuous receive mode)



Remark: ①~⑧ in the figure correspond to ①~⑧ in the “Figure 12-17 master receive (continuous receive mode)”.

12.5.3 Master transmission and reception

The transmission and reception of the master refers to the operation of this product output transmission clock and data transmission and reception with other devices.

Table 12-24: Master transmission and reception

3-wire serial I/O	SSPI00	SSPI01	SSPI10	SSPI11
Target channel	Channel 0 of SCI0	Channel 1 of SCI0	Channel 0 of SCI1	Channel 1 of SCI1
Pins used	SCLKOI00, SDI00, SDO00	SCLKOI01, SDI01, SDO01	SCLKOI10, SDI10, SDO10	SCLKOI11, SDI11, SDO11
Interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11
	Selectable transmission end interrupt (single transmission mode) or buffer empty interrupt (continuous transmission mode).			
Error detection flag	Only the overflow error detection flag (OVFmn).			
Transfer data length	7~16 bits			
Transfer rate Note	Max.F_{CLK}/2[Hz] Min.F_{CLK}/(22^{11}128)[Hz] F_{CLK}: System clock frequency			
Data phase	It can be selected by the DAPmn bit of the SCRmn register. DAPmn=0: Start data output when the serial clock starts running. DAPmn=1: Start data output half a clock before the serial clock starts running.			
Clock phase	It can be selected by the CKPmn bit of the SCRmn register. CKPmn=0: Non-reverse CKPmn=1: Reverse			
Data direction	MSB or LSB first			

Notice: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

Remark: m: unit number (m=0, 1);

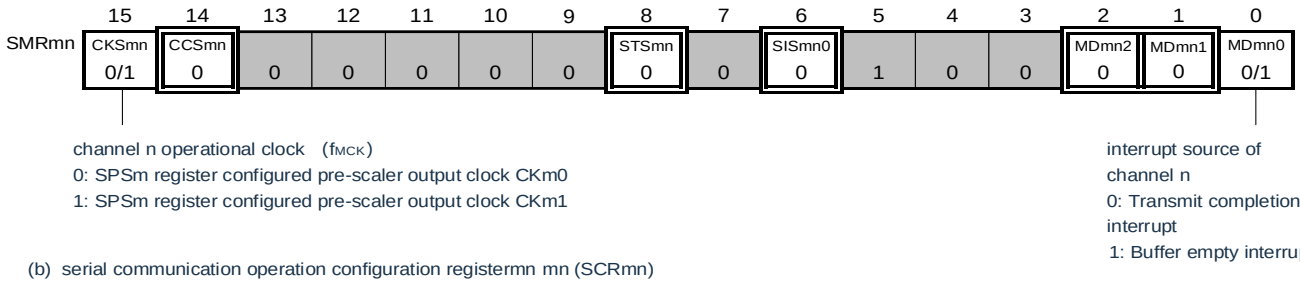
n: channel number (n=0, 1);

p: SSPI number (p=00, 01, 10, 11).

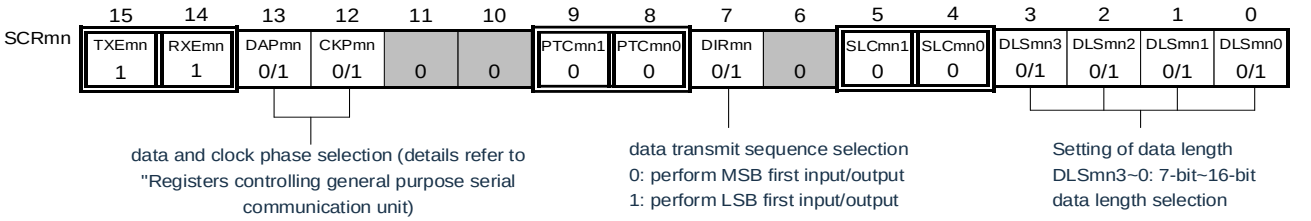
(1) Register setting

Figure 12-19: 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11)
Example of register settings for master transmission and reception

(a) serial mode register mn (SMRmn)

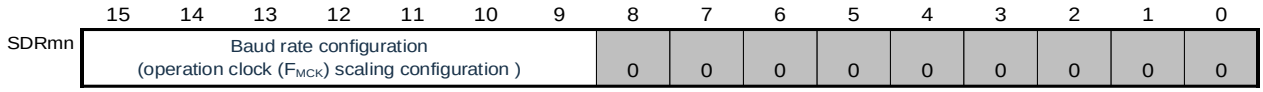


(b) serial communication operation configuration registermn mn (SCRmn)

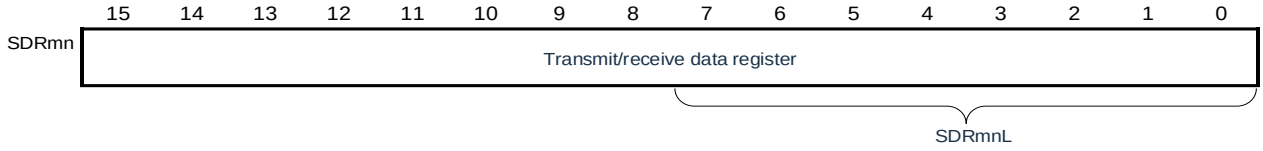


(c) serial data registermn mn (SDRmn)

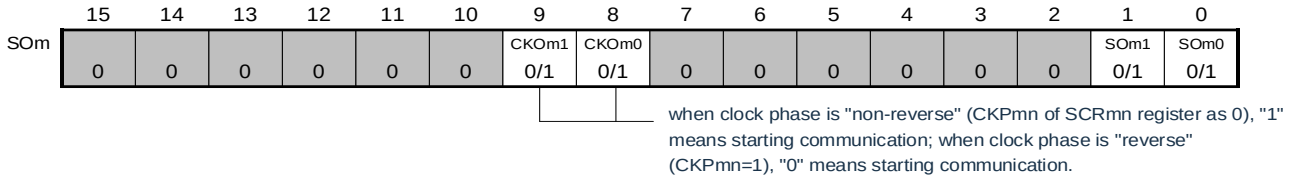
(1) When operation stops (SEmn=0)



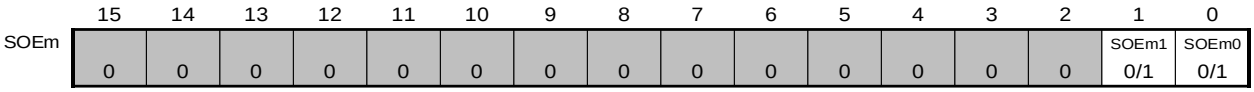
(2) During operation（SEmn=1）（lower 8 bits: SDRmnL）



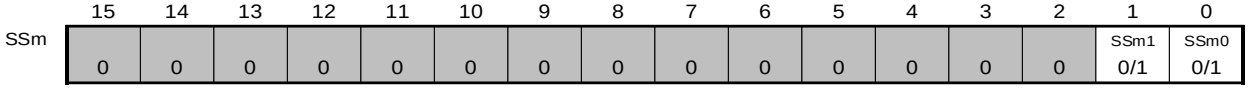
(d) serial output register m(SOm) .....Only configure bit of target channel



(e) serial output enable register m (SOEm)….Only set bit of target channel to 1.



(f) serial channel start registerm (SSm)….only set bit of target channel to 1.



Remark: m: unit number (m=0, 1);

n: channel number (n=0, 1);

p: SSPI number (p=00, 01, 10, 11);

☐ : Fixed setin SSPI master receive mode ☐ : Cannot be set (initial value is set);

0/1: Set “0” or “1” according to the user’s purpose.

(2) Procedure

Figure 12-20: Initial setup steps for master transmission and reception

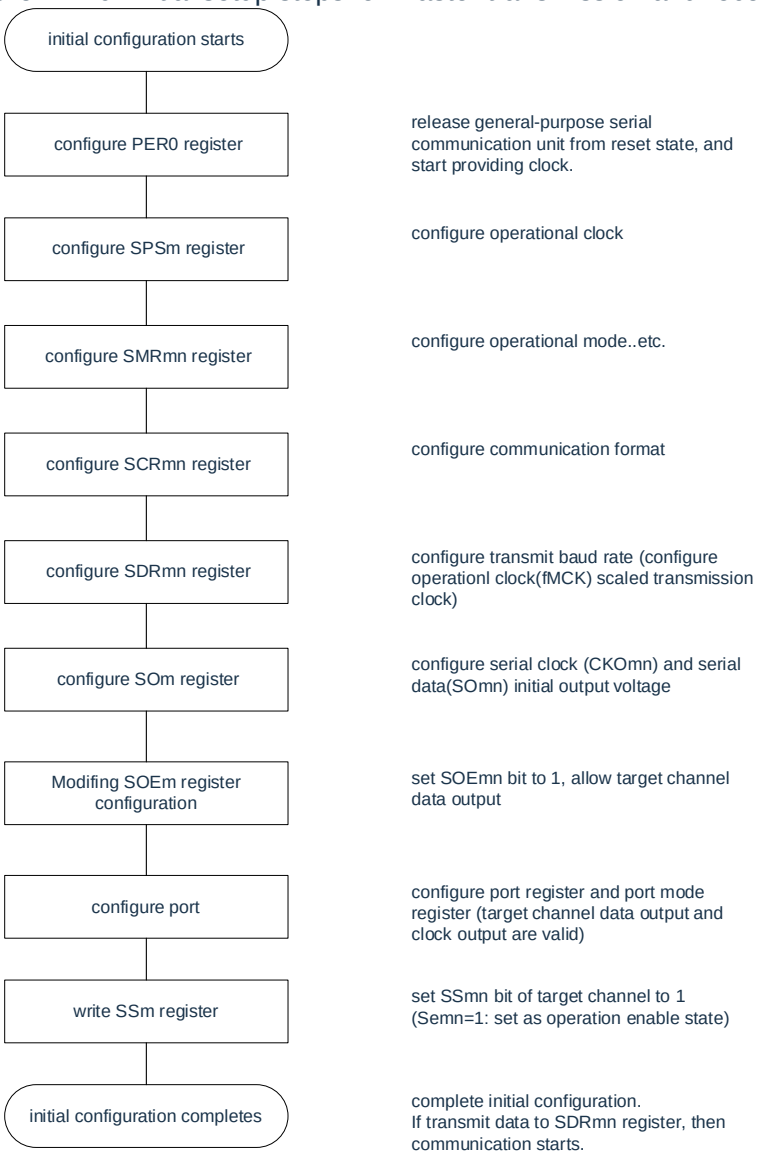


Figure 12-21: Stop steps for master transmission and reception

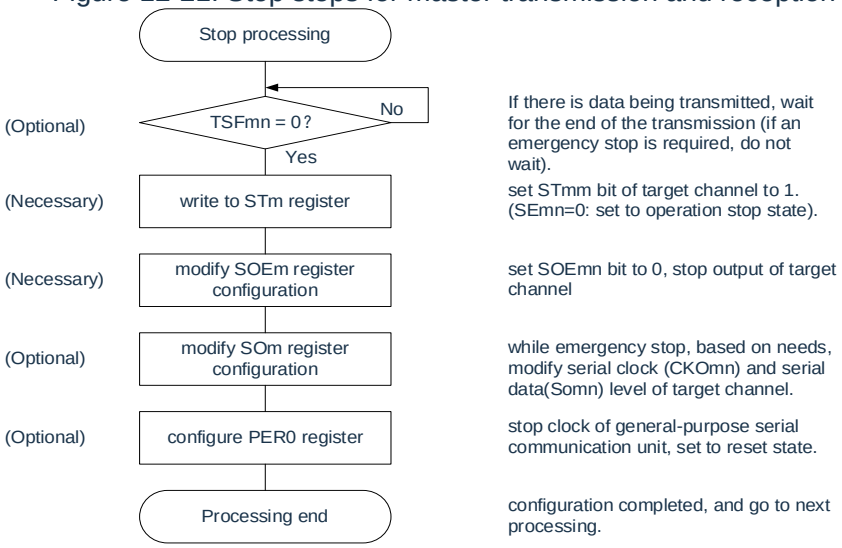
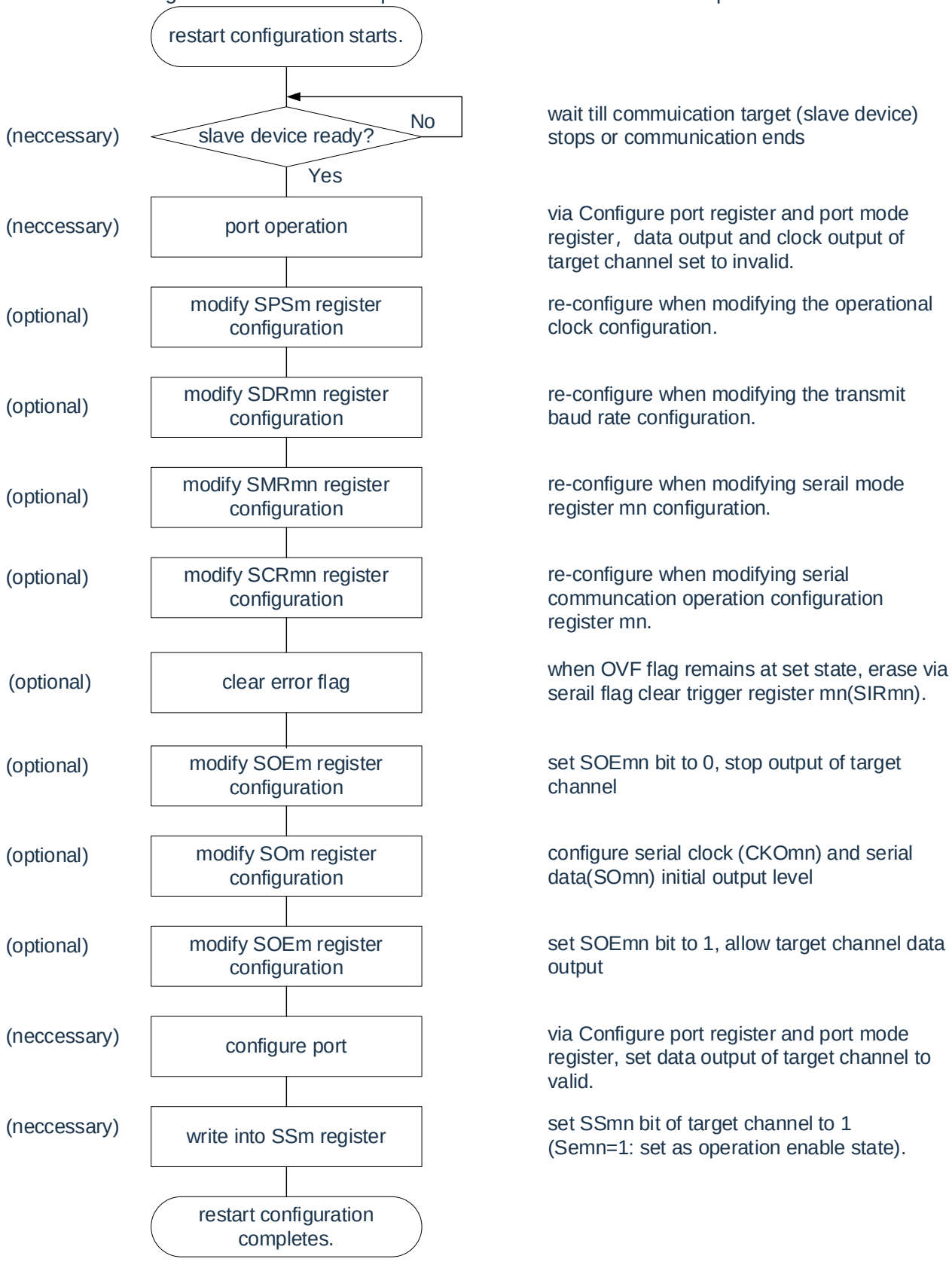
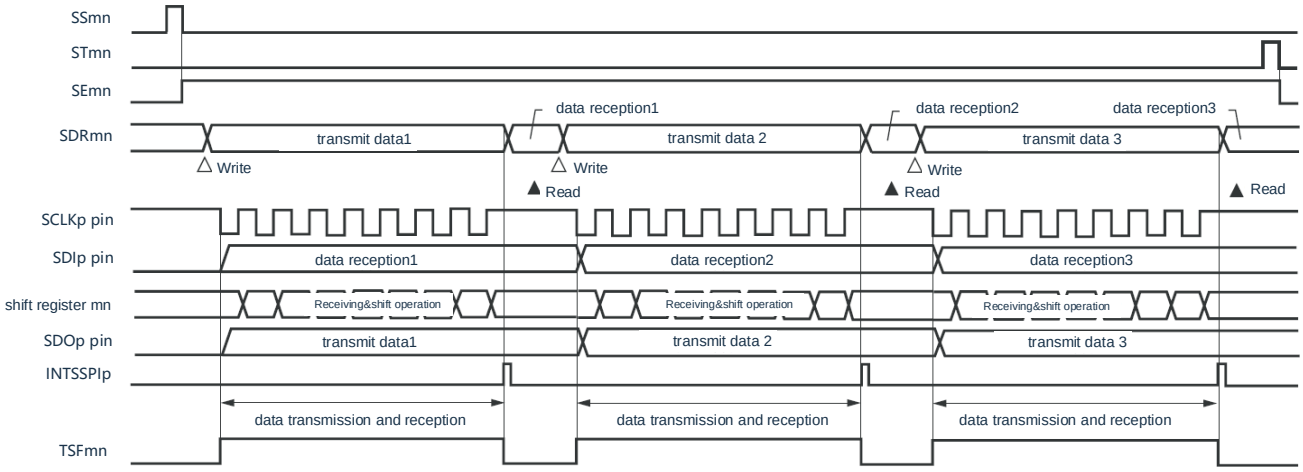


Figure 12-22: Restart steps for master transmission and reception



(3) Process flow (single transmit and receive mode)

Figure 12-23: Timing diagram of master transmit and receive (single transmit and receive mode) (type 1: DAPmn=0, CKPmn=0)

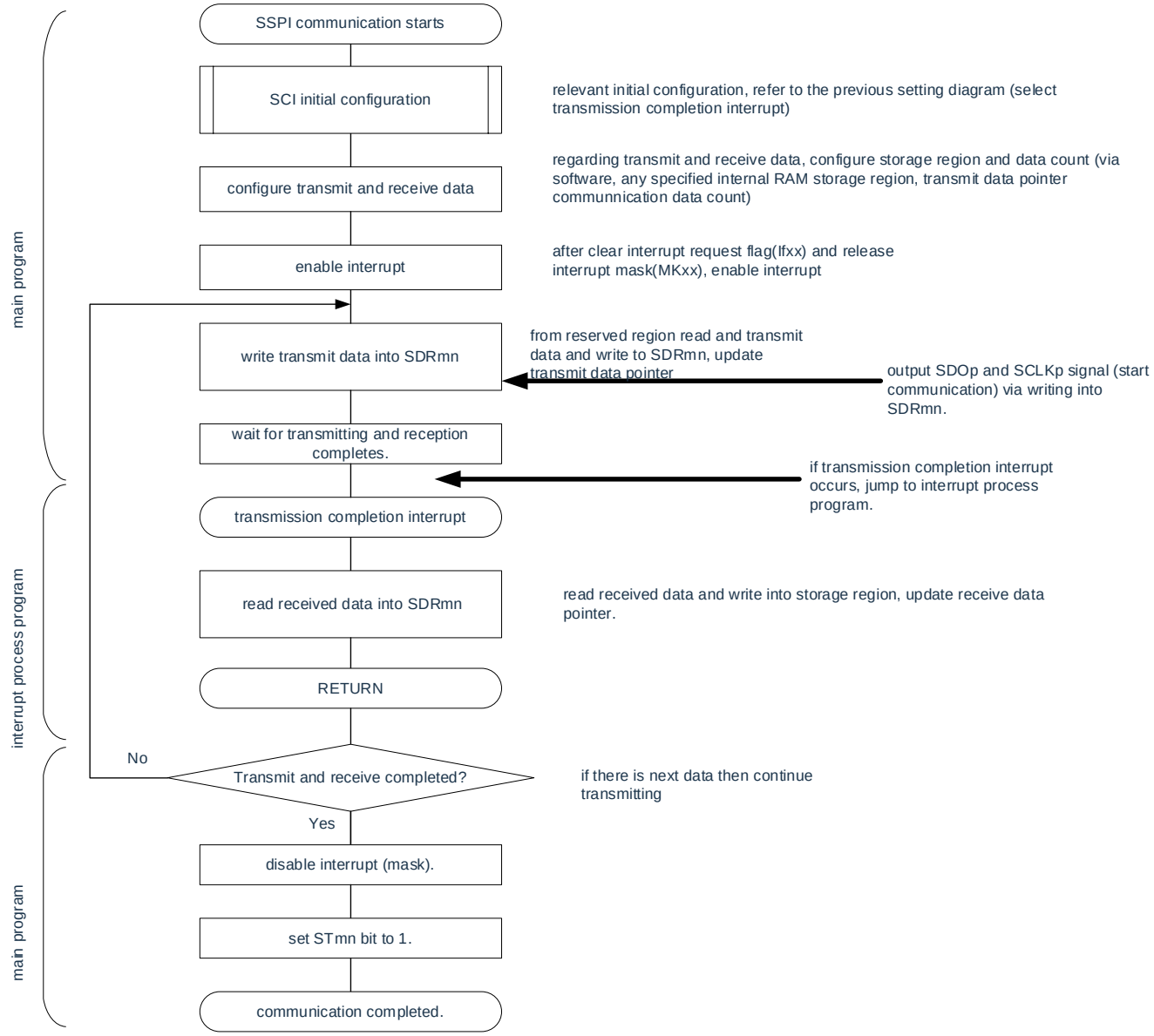


Remark: m: unit number (m=0, 1);

n: channel number (n=0, 1);

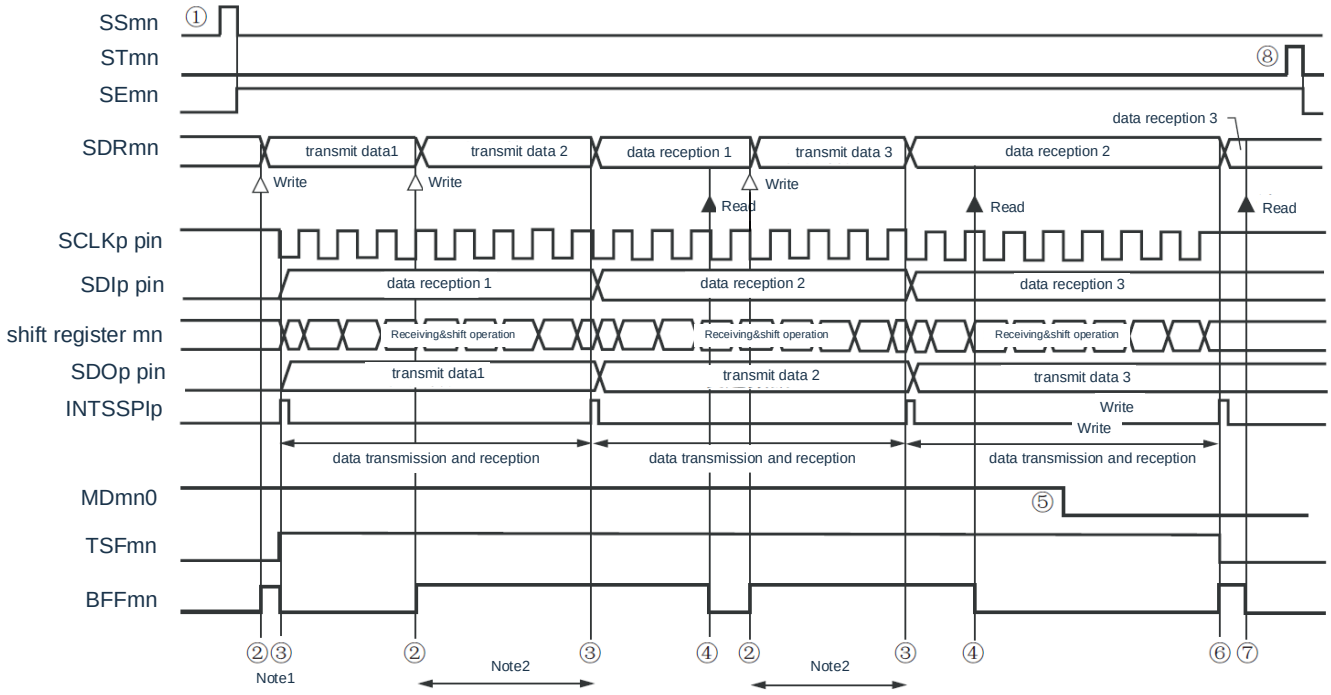
p: SSPI number (p=00, 01, 10, 11).

Figure 12-24: Flowchart of master transmit and receive (single transmit and receive mode)



(4) Process flow (continuous transmit and receive mode)

Figure 12-25: Timing diagram of master transmit and receive (continuous transmit and receive mode) (type 1: DAPmn=0, CKPmn=0)



Note 1: If the transmit data is written to the SDRmn register during the time when the BFFmn bit of the serial status register mn (SSRmn) is "1" (when valid data is stored in the serial data register mn (SDRmn)), the transmit data is rewritten.

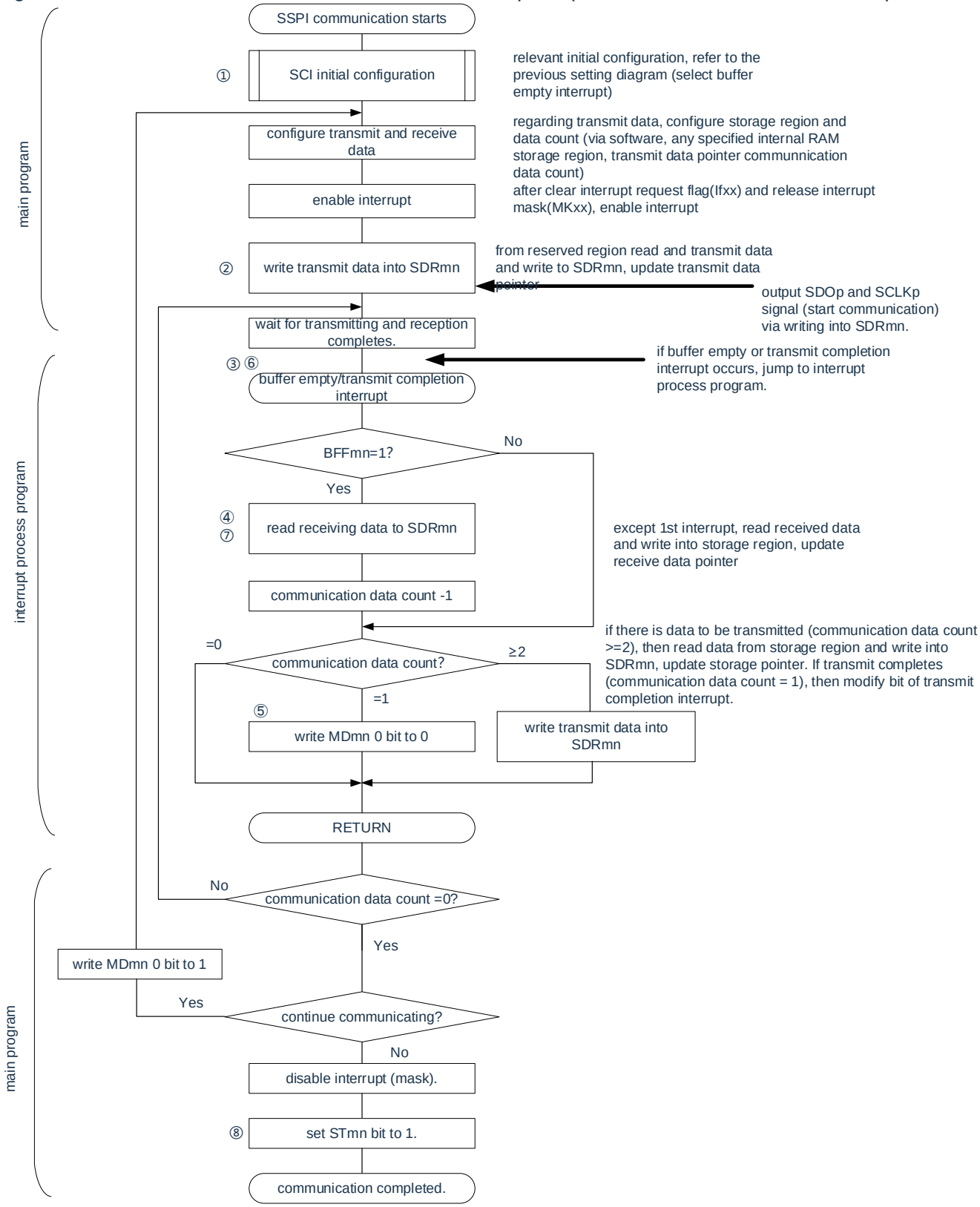
Note 2: If the SDRmn register is read during this time, the transmit data can be read. At this time, the transmission operation is not affected.

Notice: It is possible to rewrite the MDmn0 bit of the Serial Mode Register mn (SMRmn) even during operation. However, in order to catch the end-of-transmission interrupt for the last data sent, the rewrite must be done before the transmission of the last bit begins.

Remark:

1. ①~⑧ in the figure correspond to ①~⑧ in “Figure 12-26 Flowchart of Master Transmission and Reception (Continuous Transmission and Reception Mode)”.
2. m: unit number (m=0, 1);
   n: channel number (n=0, 1);
   p: SSPI number (p=00, 01, 10, 11).

Figure 12-26: Flowchart of master transmission and reception (continuous transmission and reception mode)



Remark:①~⑧ in the figure correspond to ①~⑧ in “Figure 12-25 Timing Diagram of Master Transmit and Receive (Continuous Transmit and Receive Mode)".

12.5.4 Slave transmission

Slave send is the operation of a microcontroller that sends data to other devices in a state where the transfer clock is input from the other device.

Table 12-25: Slave transmission

3-wire serial I/O	SSPI00	SSPI01	SSPI10	SSPI11
Target channel	Channel 0 of SCI0	Channel 1 of SCI0	Channel 0 of SCI1	Channel 1 of SCI1
Pins used	SCLKOI00, SDO00	SCLKOI01, SDO01	SCLKOI10, SDO10	SCLKOI11, SDO11
Interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11
	Selectable transmission end interrupt (single transmission mode) or buffer empty interrupt (continuous transmission mode).			
Error detection flag	Only the overflow error detection flag (OVFmn).			
Transfer data length	7~16 bits			
Transfer rate	Max.F_{MCK}/6[Hz]Note 1,2			
Data phase	It can be selected by the DAPmn bit of the SCRmn register. DAPmn=0: Start data output when the serial clock starts running. DAPmn=1: Start data output half a clock before the serial clock starts running.			
Clock phase	It can be selected by the CKPmn bit of the SCRmn register. CKPmn=0: Non-reverse CKPmn=1: Reverse			
Data direction	MSB or LSB first			

Note 1: The maximum transfer rate is F_{MCK}/6[Hz] because the external serial clocks input to the SCLKOI00, SCLKOI01, SCLKOI10, and SCLKOI11 pins are internally sampled and used;

Note 2: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

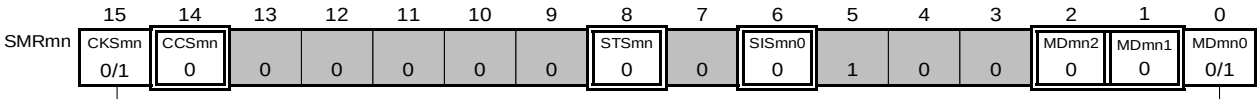
Remark:

1. F_{MCK}: Operation clock frequency of target channel
2. m: unit number (m=0, 1);
   n: channel number (n=0, 1).

(1) Register setting

Figure 12-27: 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11)
Example of register settings for slave transmission

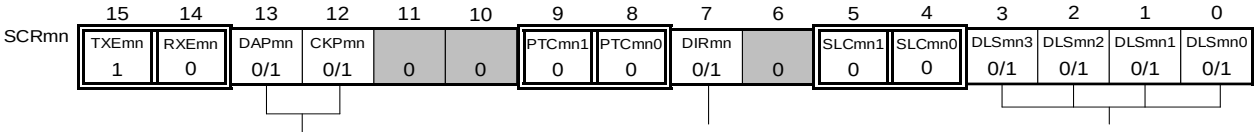
(a) serial mode register mn (SMRmn)



channel n operational clock (fMCK)
0: SPSm register configured pre-scaler output clock CKm0
1: SPSm register configured pre-scaler output clock CKm1

interrupt source of channel n
0: Transmit completion interrupt
1: Buffer empty interrupt

(b) serial communication operation configuration register mn (SCRmn)



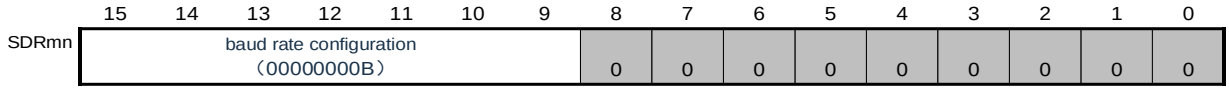
data and clock phase selection (details refer to registers controlling universal serial communication unit)

data transmit sequence selection
0: perform MSB first input/output
1: perform LSB first input/output

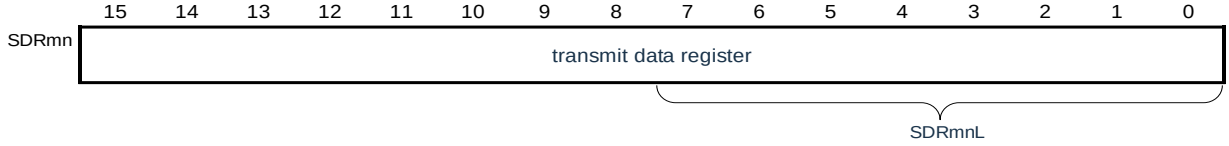
Setting of data length
DLSmn3~0: 7-bit~16-bit data length selection

(c) serial data regsiter mn (SDRmn)

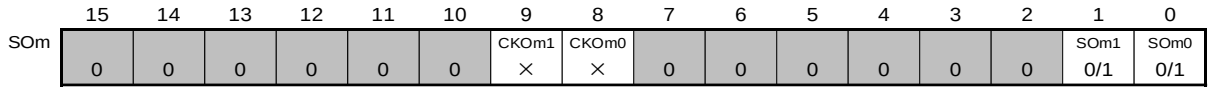
(1) When operation stops (SEmn=0)



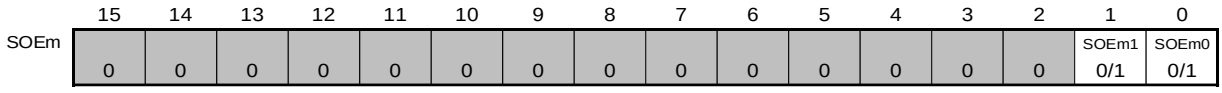
(2) During operation（SEmn=1）（lower 8 bits: SDRmnL）



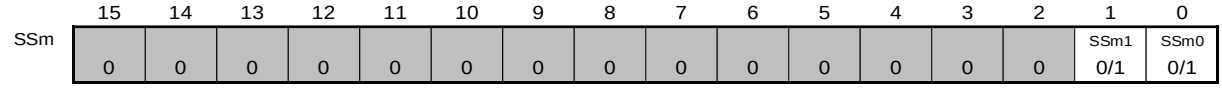
(d) serial output register m(SOm) .....Only configure bit of target channel



(e) serial output enable registerm (SOEm)….only set bit of target channel to 1.



(f) serial channel start register m (SSm) …. Only set bit of target channel to 1.



Remark: m: unit number (m=0, 1);

n: channel number (n=0, 1);

p: SSPI number (p=00, 01, 10, 11);

□: Fixed set in SSPI master receive mode ■: Cannot be set (initial value is set);

×: This is a bit that cannot be used in this mode (sets the initial value if it is not used in other modes either);

0/1: Set “0” or “1” according to the user’s purpose.

(2) Procedure

Figure 12-28: Initial setup steps for slave transmission

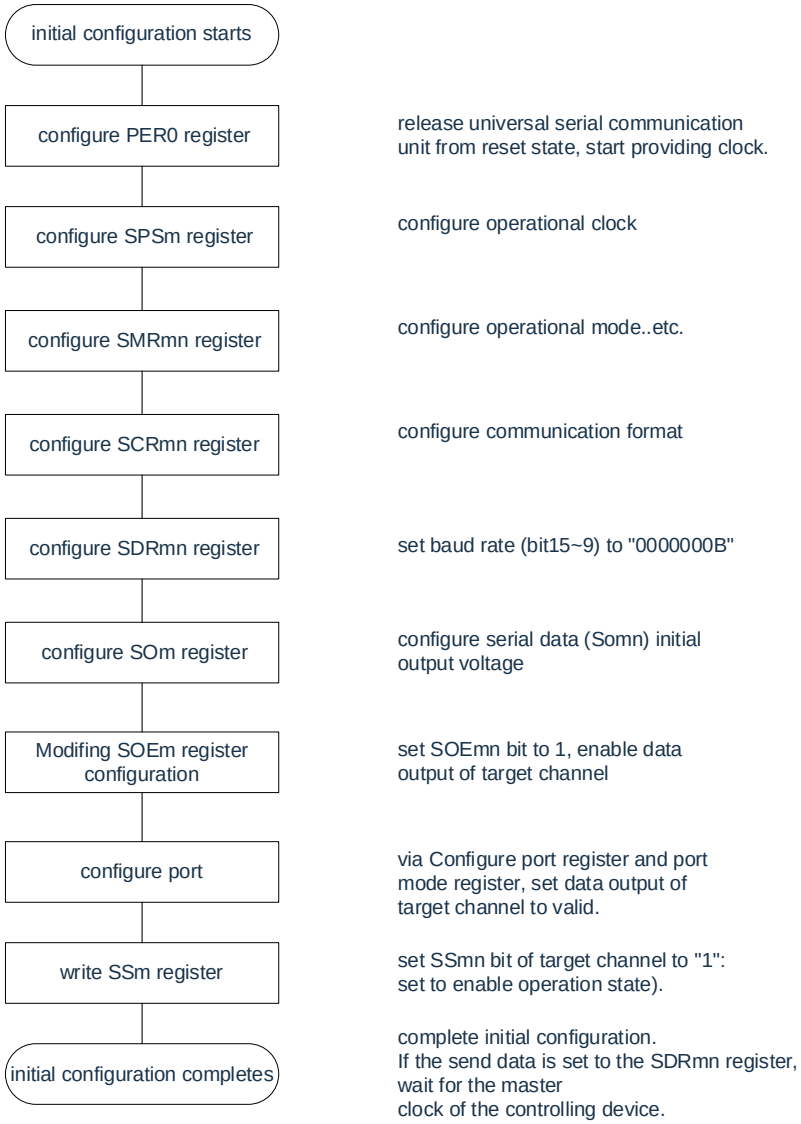


Figure 12-29: Stop steps for slave transmission

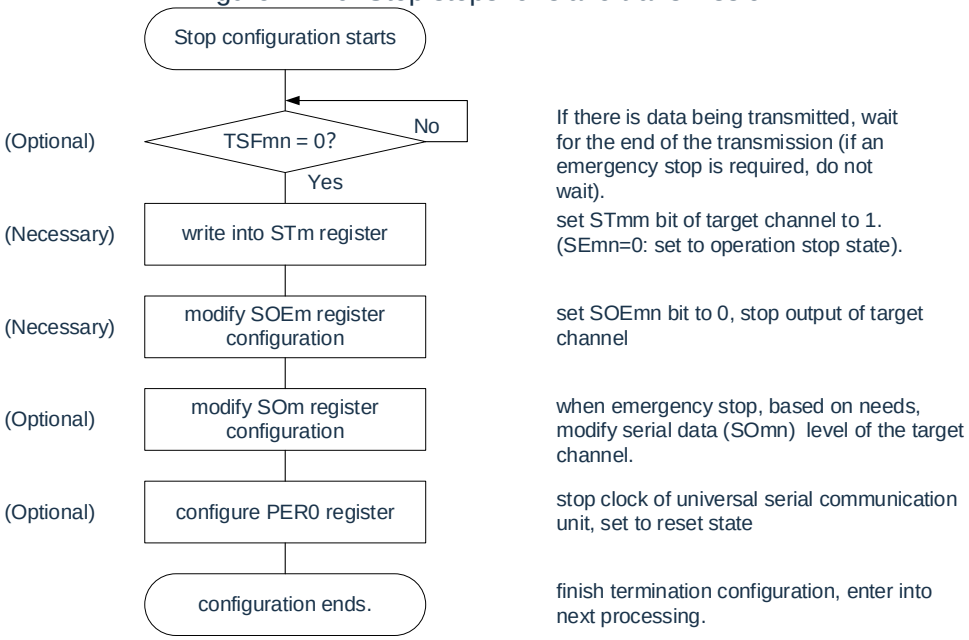
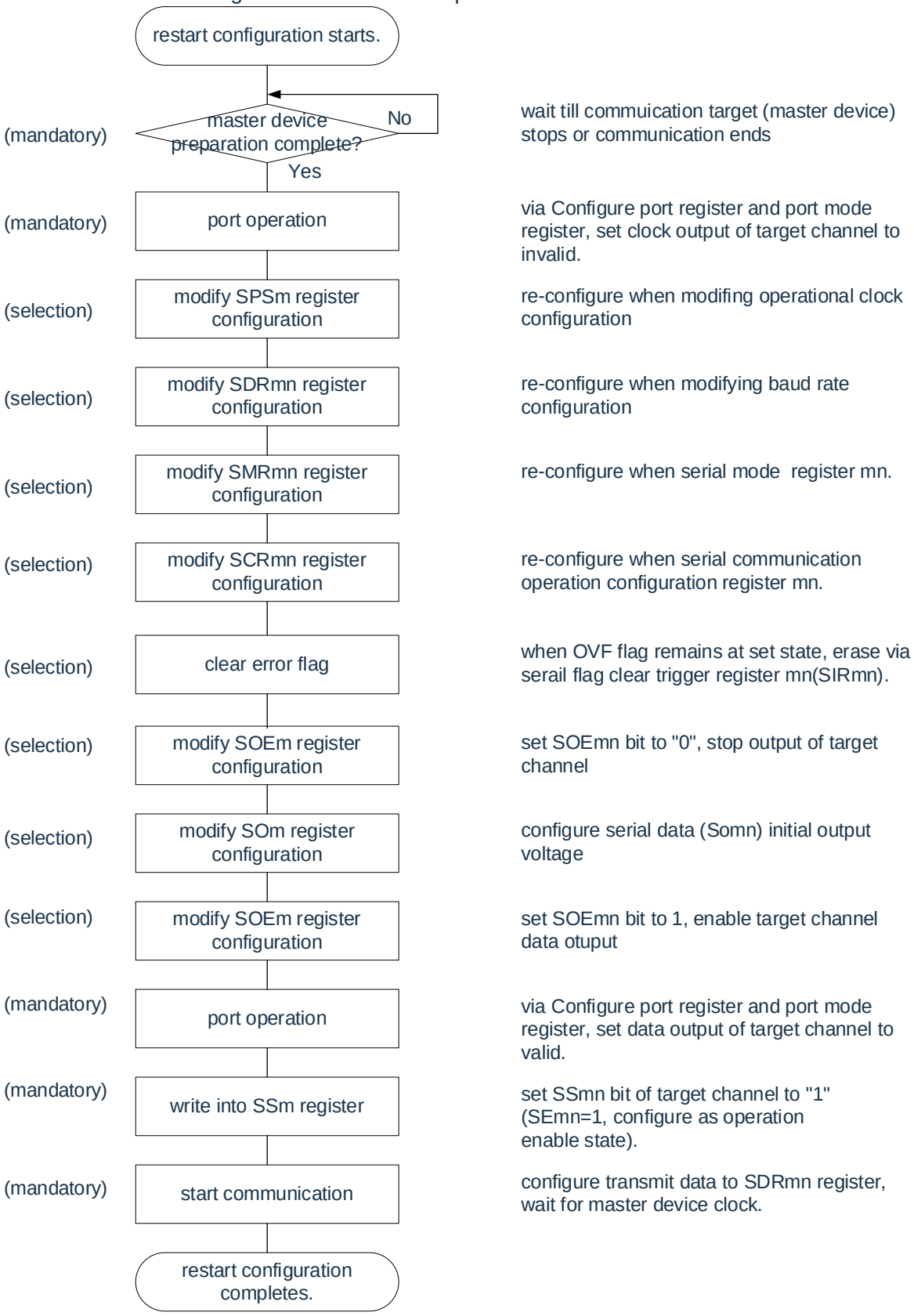


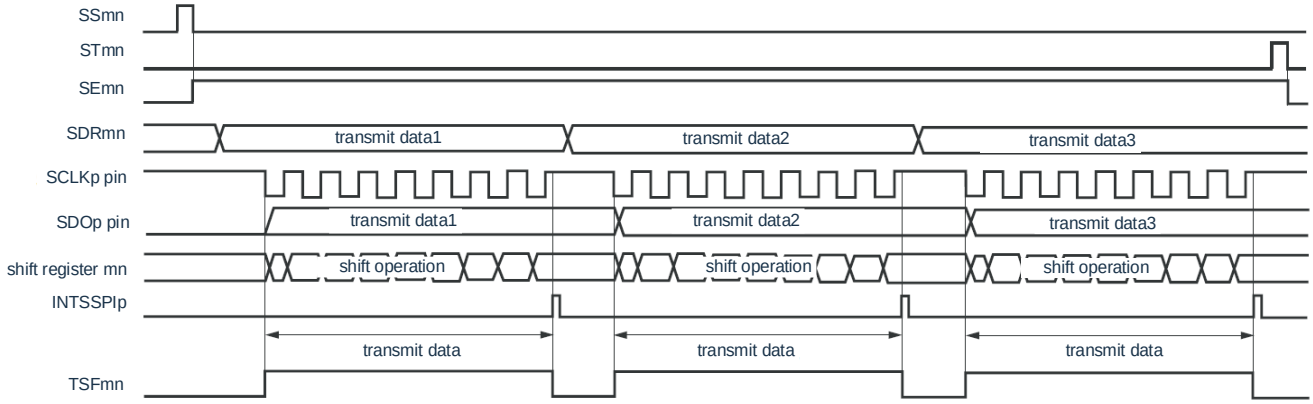
Figure 12-30: Restart steps for slave transmission



Notice: If PER0 is rewritten in the abort setting to stop supplying the clock, it is necessary to wait until the communication object (master device) stops or the communication is finished to make the initial setting instead of making the restart setting.

(3) Process flow (single transmit mode)

Figure 12-31: Timing diagram of slave tranmsmit (single tranmsmit mode) (type 1: DAPmn=0, CKPmn=0)

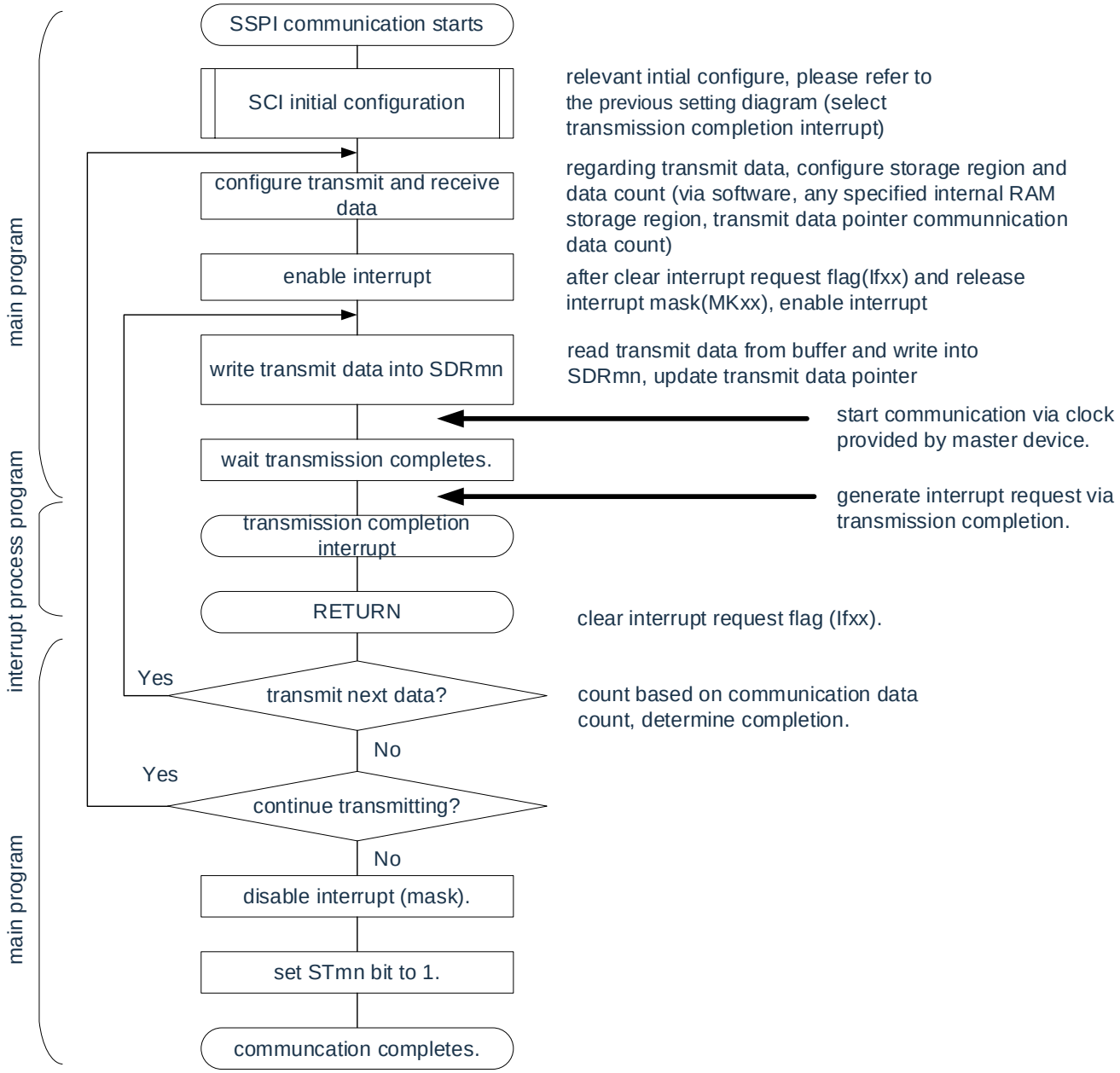


Remark: m: unit number (m=0, 1);

n: channel number (n=0, 1);

p: SSPI number (p=00, 01, 10, 11).

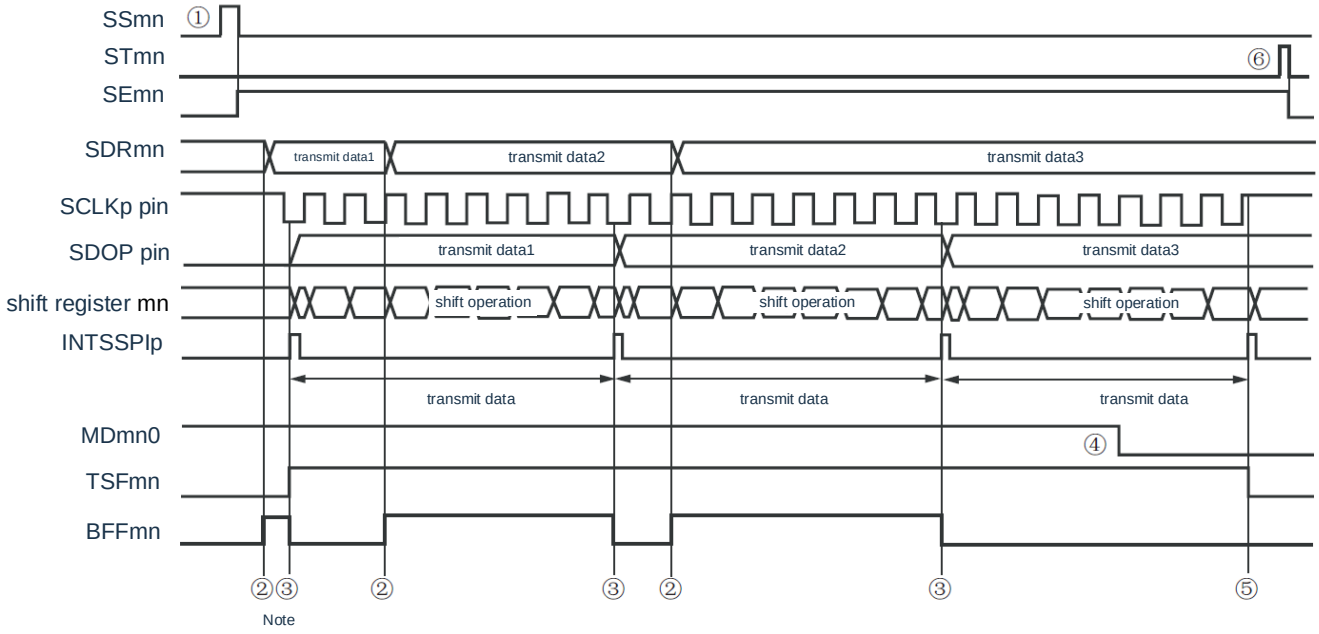
Figure 12-32: Flowchart of slave transmit (single transmit mode)



(4) Process flow (continuous transmit mode)

Figure 12-3: Timing diagram of slave transmission (continuous transmission mode)

(type 1: DAPmn=0, CKPmn=0)



Note: If the transmit data is written to the SDRmn register during the time when the BFFmn bit of the serial status register mn (SSRmn) is "1" (when valid data is stored in the serial data register mn (SDRmn)), the transmit data is rewritten.

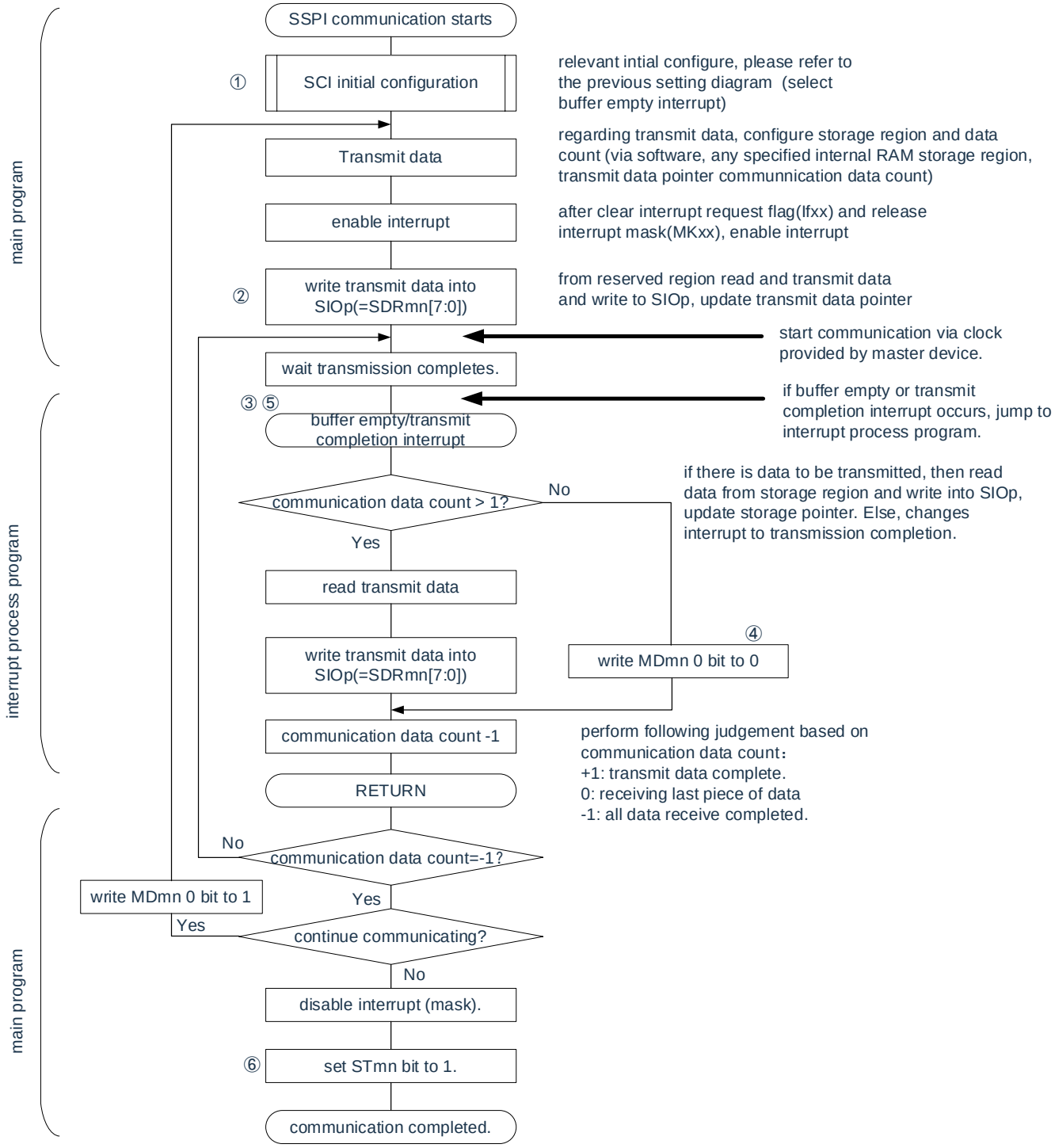
Notice: The MDmn0 bit of the Serial Mode Register mn (SMRmn) can be rewritten even during operation. However, the rewrite must be done before starting to transmit the last bit.

Remark: m: unit number (m=0, 1);

n: channel number (n=0, 1);

p: SSPI number (p=00, 01, 10, 11).

Figure 12-34: Flowchart of slave transmission (continuous transmit mode)



Remark: ① to ⑥ correspond to ① to ⑥ in “Figure 12-33 Timing Diagram of Slave Transmission (Continuous Transmission Mode)”.

12.5.5 Slave reception

Slave reception refers to the operation of this product receiving data from other devices in the state of transmitting clocks from other devices.

Table 12-26: Slave reception

3-wire serial I/O	SSPI00	SSPI01	SSPI10	SSPI11
Target channel	Channel 0 of SCI0	Channel 1 of SCI0	Channel 0 of SCI1	Channel 1 of SCI1
Pins used	SCLKOI00, SDI00	SCLKOI01, SDI01	SCLKOI10, SDI10	SCLKOI11, SDI11
Interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11
	Limited to end-of-transmit interrupts (buffer empty interrupts are prohibited).			
Error detection flag	Only the overflow error detection flag (OVFmn).			
Transfer data length	7~16 bits			
Transfer rate	Max.F_{MCK}/6[Hz]Note 1,2			
Data phase	It can be selected by the DAPmn bit of the SCRmn register. DAPmn=0: Start data output when the serial clock starts running. DAPmn=1: Start data output half a clock before the serial clock starts running.			
Clock phase	It can be selected by the CKPmn bit of the SCRmn register. CKPmn=0: Non-reverse CKPmn=1: Reverse			
Data direction	MSB or LSB first			

Note 1: The maximum transfer rate is F_{MCK}/6[Hz] because the external serial clocks input to the SCLKOI00, SCLKOI01, SCLKOI10, and SCLKOI11 pins are internally sampled and used;

Note 2: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

F_{MCK}: Operation clock frequency of target channel

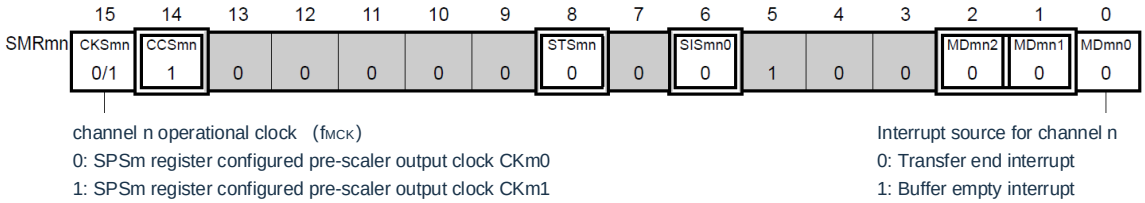
m: unit number (m=0, 1);

n: channel number (n=0, 1).

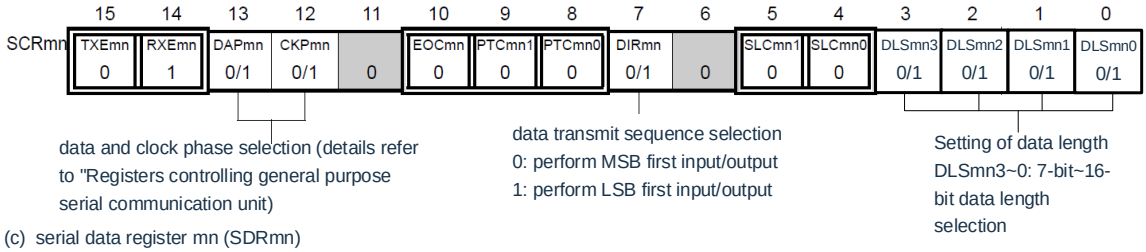
(1) Register setting

Figure 12-35: 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11)
Example of register setting content for slave reception

(a) serial mode register mn (SMRmn)

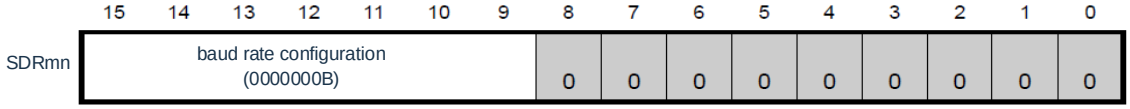


(b) serial communication operation configuration register mn (SCRmn)

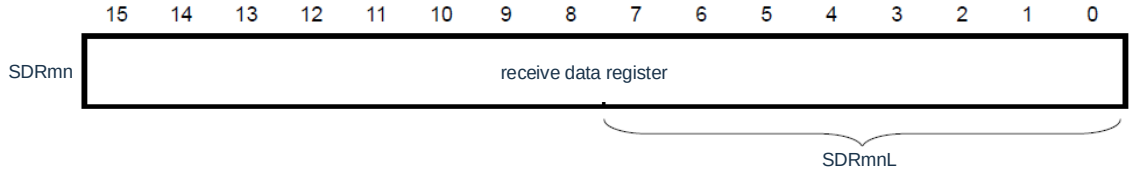


(c) serial data register mn (SDRmn)

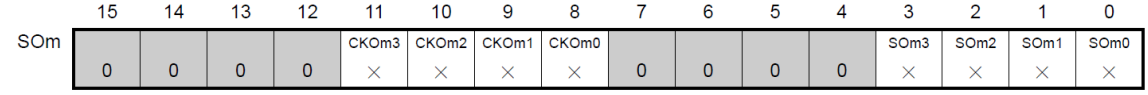
(1) When operation stops (SEmn=0)



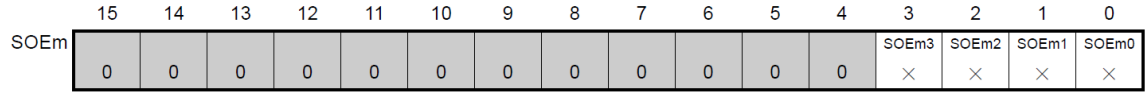
(2) During operation（SEmn=1）（lower 8 bits: SDRmnL）



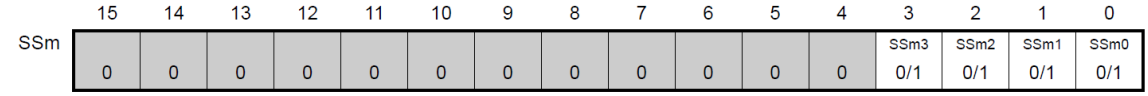
(d) serial output register m(SOm) .....not used in this mode.



(e) serial output enable register m (SOEm)….not used in this mode.



(f) serial channel start registerm (SSm)….only set bit of target channel to 1.



Remark: m: unit number (m=0, 1);

n: channel number (n=0, 1);

p: SSPI number (p=00, 01, 10, 11);

□ : Fixed set in SSPI master receive mode ■ : Cannot be set (initial value is set);

×: This is a bit that cannot be used in this mode (sets the initial value if it is not used in other modes either);

0/1: Set “0” or “1” according to the user’s purpose.

(2) Procedure

Figure 12-36: Initial setup steps of slave reception

Figure 12-37: Stop steps of slave reception

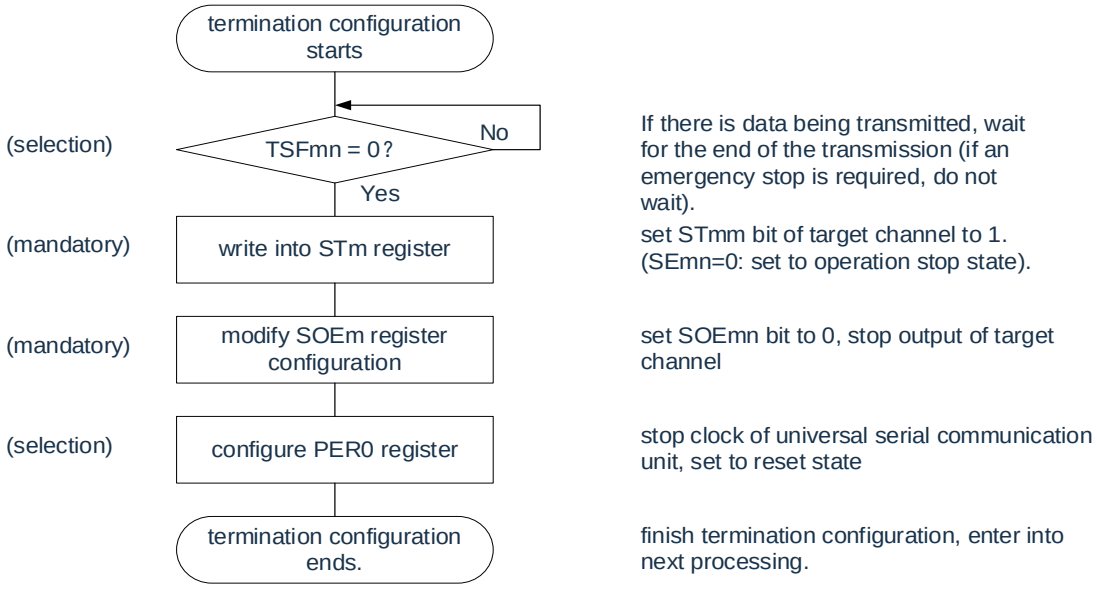
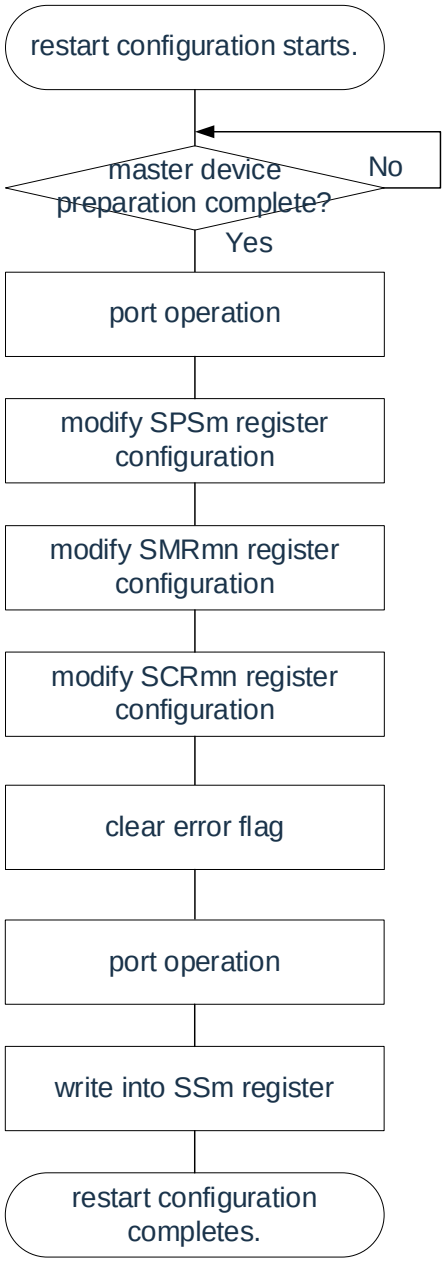


Figure 12-38: Restart steps of slave reception

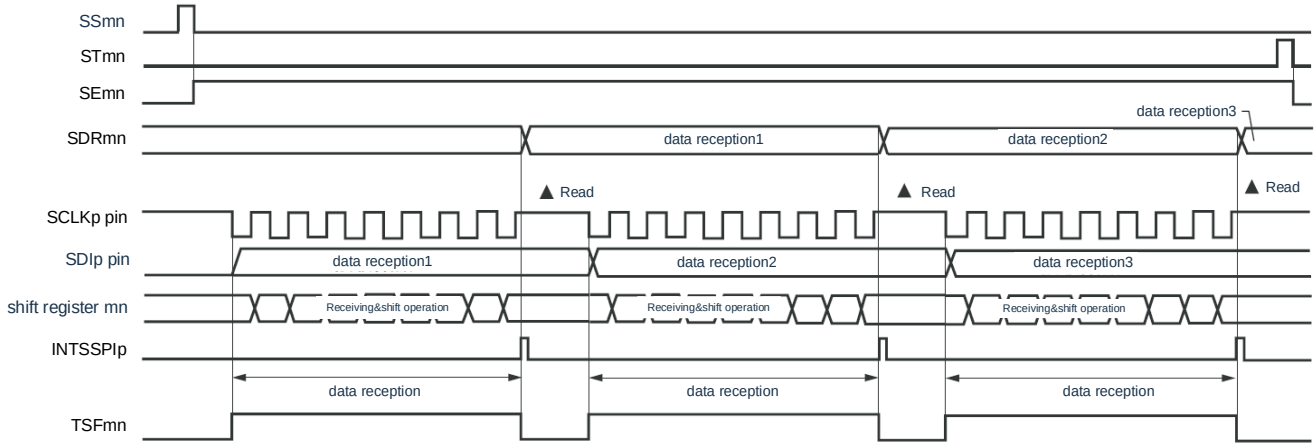


(mandatory)	master device preparation complete?	wait till commuication target (master device) stops or communication ends
(mandatory)	port operation	via Configure port register and port mode register, set clock output of target channel to invalid.
(selection)	modify SPSm register configuration	re-configure when modifing operational clock configuration
(selection)	modify SMRmn register configuration	re-configure when serial mode register mn.
(selection)	modify SCRmn register configuration	re-configure when serial communication operation configuration register mn.
(selection)	clear error flag	when OVF flag remains at set state, erase via serail flag clear trigger register mn(SIRmn).
(mandatory)	port operation	via Configure port register and port mode register, set data output of target channel to valid.
(mandatory)	write into SSm register	set SSmn bit of target channel to "1" (SEmn=1, configure as operation enable state). wait for master device clock.

Notice: If PER0 is rewritten in the abort setting to stop supplying the clock, it is necessary to wait until the communication object (master device) stops or the communication is finished to make the initial setting instead of making the restart setting.

(3) Process flow (single receive mode)

Figure 12-39: Timing diagram of slave receive (single receive mode) (type 1: DAPmn= 0, CKPmn = 0)

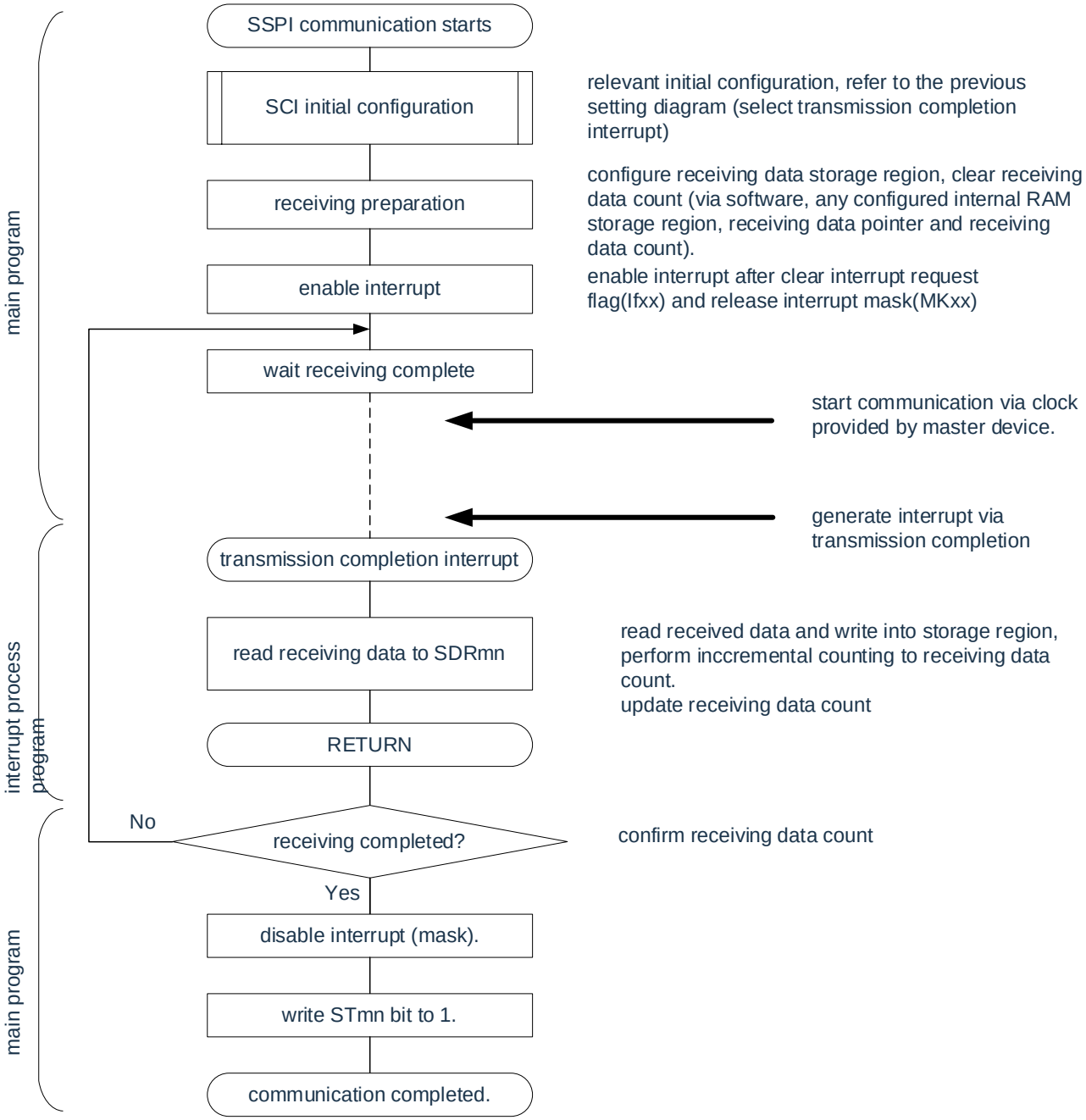


Remark: m: unit number (m=0, 1);

n: channel number (n=0, 1);

p: SSPI number (p=00, 01, 10, 11).

Figure 12-40: Flowchart of slave receive (single receive mode)



12.5.6 Slave transmission and reception

Slave transmission and reception refers to the operation of data transmission and reception by microcontrollers and other devices of this product in the state of transmitting clocks from other device inputs.

Table 12-27: Slave transmission and reception

3-wire serial I/O	SSPI00	SSPI01	SSPI10	SSPI11
Target channel	Channel 0 of SCI0	Channel 1 of SCI0	Channel 0 of SCI1	Channel 1 of SCI1
Pins used	SCLKOI00, SDI00, SDO00	SCLKOI01, SDI01, SDO01	SCLKOI10, SDI10, SDO10	SCLKOI11, SDI11, SDO11
Interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11
	Selectable transmission end interrupt (single transmission mode) or buffer empty interrupt (continuous transmission mode).			
Error detection flag	Only the overflow error detection flag (OVFmn).			
Transfer data length	7~16 bits			
Transfer rate	Max.F_{MCK}/6[Hz]Note 1,2			
Data phase	It can be selected by the DAPmn bit of the SCRmn register. DAPmn=0: Starts data input/output when the serial clock starts running. DAPmn=1: Starts data input/output half a clock before the serial clock starts running.			
Clock phase	It can be selected by the CKPmn bit of the SCRmn register. CKPmn=0: Non-reverse CKPmn=1: Reverse			
Data direction	MSB or LSB first			

Note 1: The maximum transfer rate F_{MCK}/6[Hz] because the external serial clocks input to the SCLKOI00, SCLKOI01, SCLKOI10, and SCLKOI11 pins are internally sampled and then used;

Note 2: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

Remark:

1. F_{MCK}: Operation clock frequency of target channel
2. m: unit number (m=0, 1);
   n: channel number (n=0, 1).

(1) Register setting

Figure 12-41: 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11)
Example of register settings for slave transmission and reception

(a) serial mode register mn (SMRmn)

SMRmn	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CKSmn	CCSmn						STSmn		SISmn0				MDmn2	MDmn1	MDmn0
	0/1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0/1

channel n operational clock (F_{MCK})
0: SPSm register configured pre-scaler output clock CKm0
1: SPSm register configured pre-scaler output clock CKm1

interrupt source of channel n
0: Transmit completion interrupt
1: Buffer empty interrupt

(b) serial communication operation configuration register mn (SCRmn)

SCRmn	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TXEmn	RXEmn	DAPmn	CKPmn		EOCmn	PTCmn1	PTCmn0	DIRmn		SLCmn1	SLCmn0	DLSmn3	DLSmn2	DLSmn1	DLSmn0
	1	1	0/1	0/1	0	0	0	0	0/1	0	0	0	0/1	0/1	0/1	0/1

data and clock phase selection (details refer to registers controlling universal serial communication unit)

data transmit sequence selection
0: perform MSB first input/output
1: perform LSB first input/output

Setting of data length
DLSmn3~0: 7-bit~16-bit data length selection

(c) serial data register mn (SDRmn)

(1) When operation stops (SEmn=0)

SDRmn	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	baud rate configuration (0000000B)							0	0	0	0	0	0	0	0	0

(2) During operation (SEmn=1) (lower 8 bits: SDRmnL)

SDRmn	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	transmit/receive data register															

SDRmnL

(d) serial output register m(SOm) .....Only configure bit of target channel

SOm	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					CKOm3	CKOm2	CKOm1	CKOm0					SOm3	SOm2	SOm1	SOm0
	0	0	0	0	×	×	×	×	0	0	0	0	0/1	0/1	0/1	0/1

(e) serial output enable register m (SOEm)….only set bit of target channel to 1.

SOEm	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
													SOEm3	SOEm2	SOEm1	SOEm0
	0	0	0	0	0	0	0	0	0	0	0	0	0/1	0/1	0/1	0/1

(f) serial channel start register m (SSm) …. Only set bit of target channel to 1.

SSm	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
													SSm3	SSm2	SSm1	SSm0
	0	0	0	0	0	0	0	0	0	0	0	0	0/1	0/1	0/1	0/1

Notice: The transmit data must be set to the SDRmn register before the master device starts outputting the clock.

Remark: m: unit number (m=0, 1);

n: channel number (n=0, 1);

p: SSPI number (p=00, 01, 10, 11);

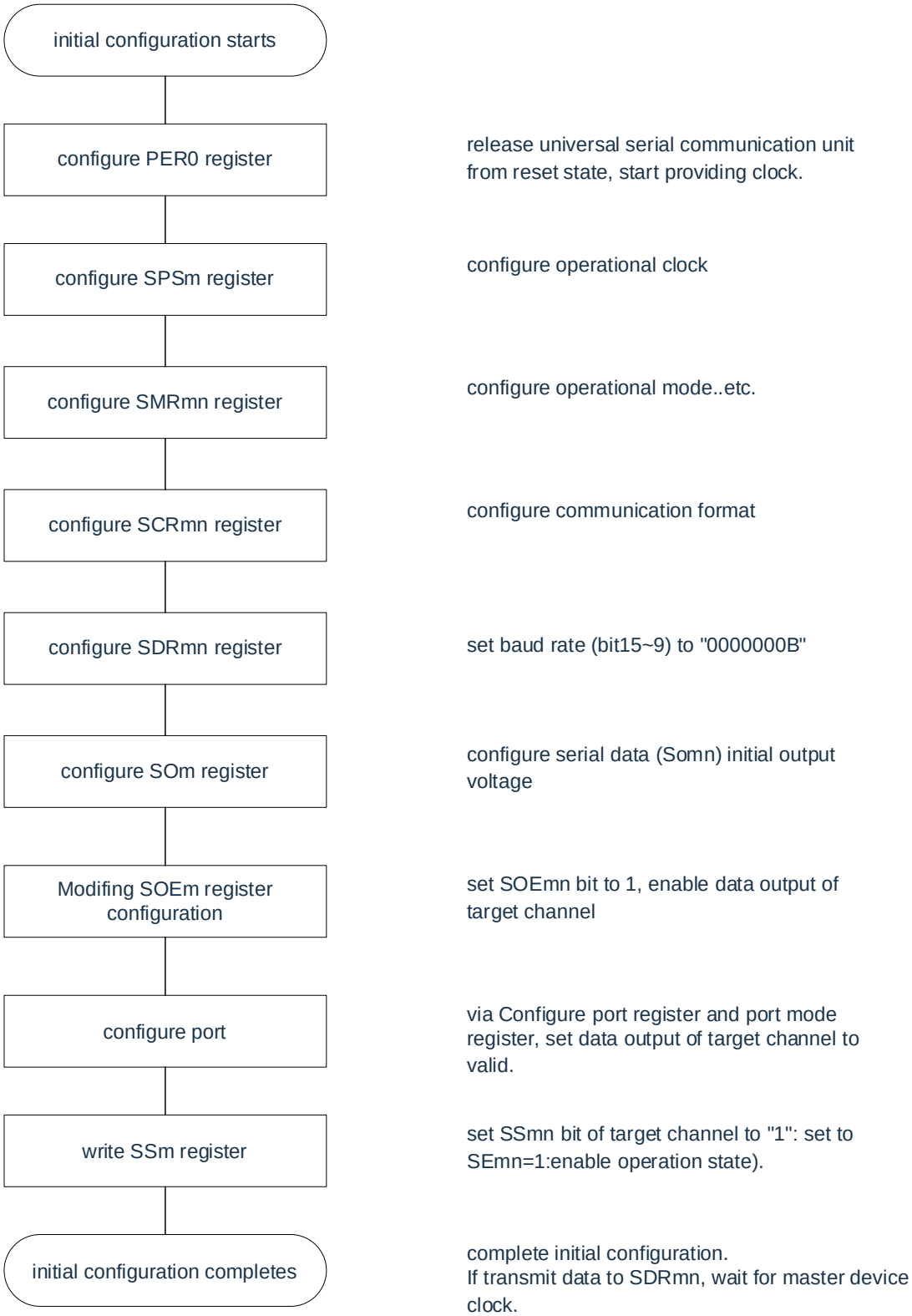
□: Fixed set in SSPI master receive mode; ■ : Cannot be set (initial value is set);

×: This is a bit that cannot be used in this mode (sets the initial value if it is not used in other modes either);

0/1: Set “0” or “1” according to the user’s purpose.

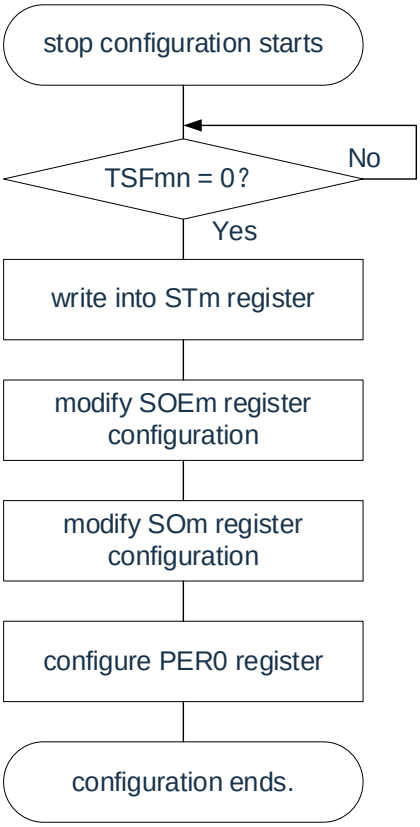
(2) Procedure

Figure 12-42: Initial setup steps for slave transmission and reception



Notice: The transmit data must be set to the SDRmn register before the master device starts outputting the clock.

Figure 12-43: Stop steps for slave transmission and reception



if there are ongoing data transmission, then wait till transmission completed. (if need urgent stop, then no need to wait).

set STmm bit of target channel to 1. (SEmn=0: set to operation stop state).

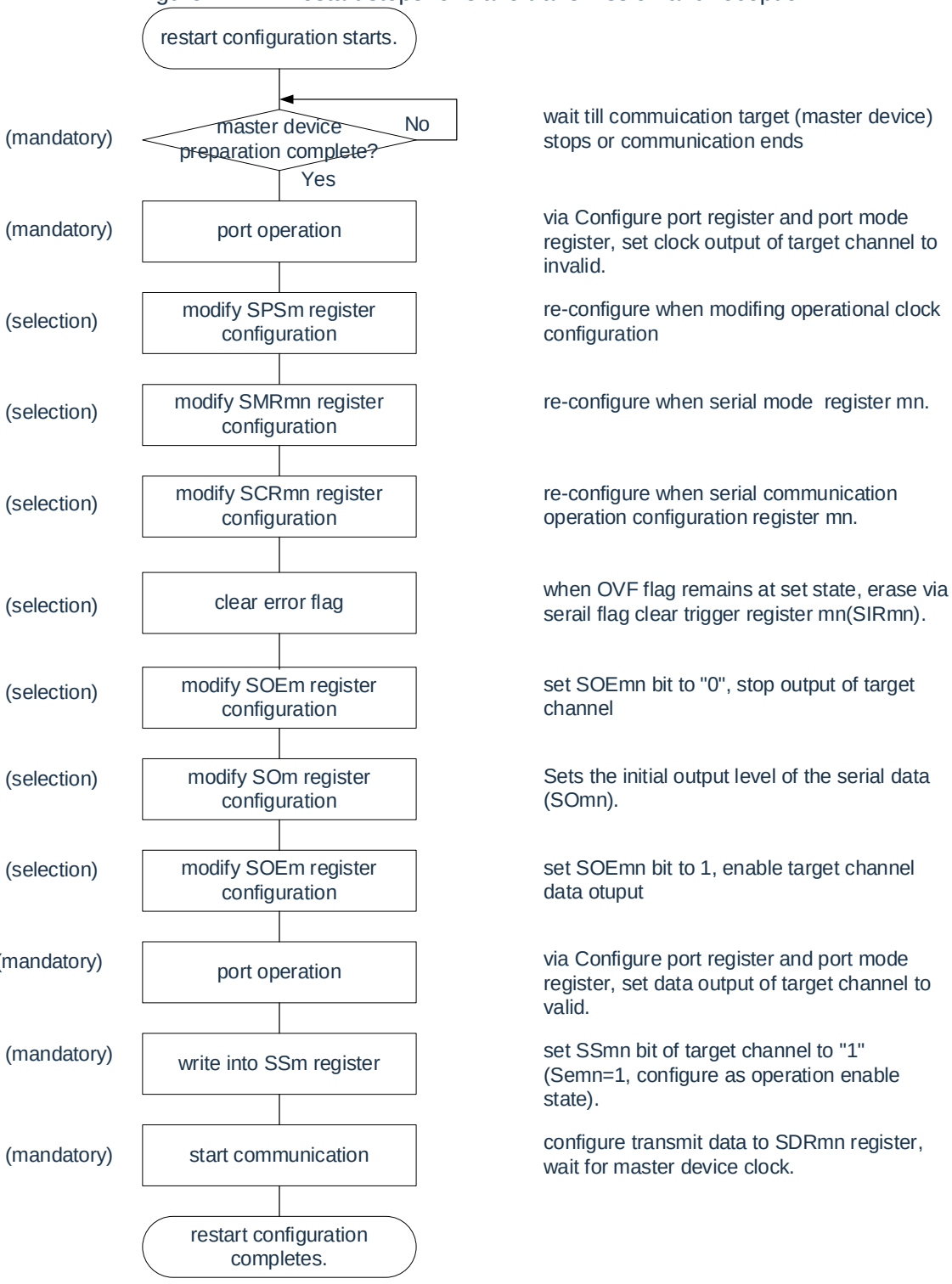
set SOEmn bit to 0, stop output of target channel

when emergency stop, based on needs, modify serial data (SOmn) level of the target channel.

stop clock of universal serial communication unit, set to reset state

finish termination configuration, enter into next processing.

Figure 12-44: Restart steps for slave transmission and reception

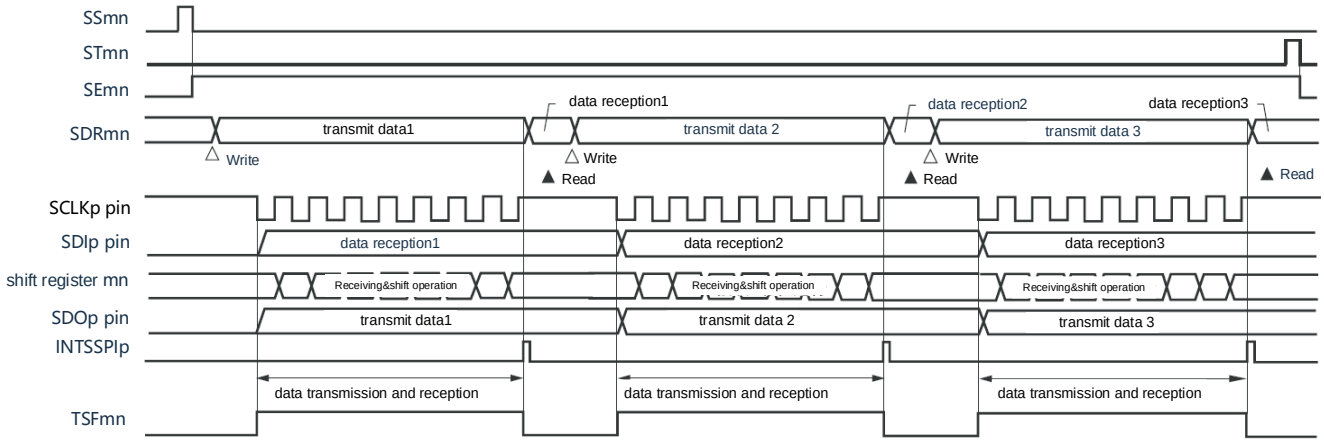


Notice:

1. The transmit data must be set to the SDRmn register before the master device starts outputting the clock.
2. If PER0 is rewritten in the abort setting to stop providing the clock, the initial setting must be made after the communication object (master device) stops or communication ends instead of the restart setting.

(3) Process flow (single transmit and receive mode)

Figure 12-45: Timing diagram of slave transmit and receive (single transmit and receive mode) (type 1: DAPmn=0, CKPmn=0)

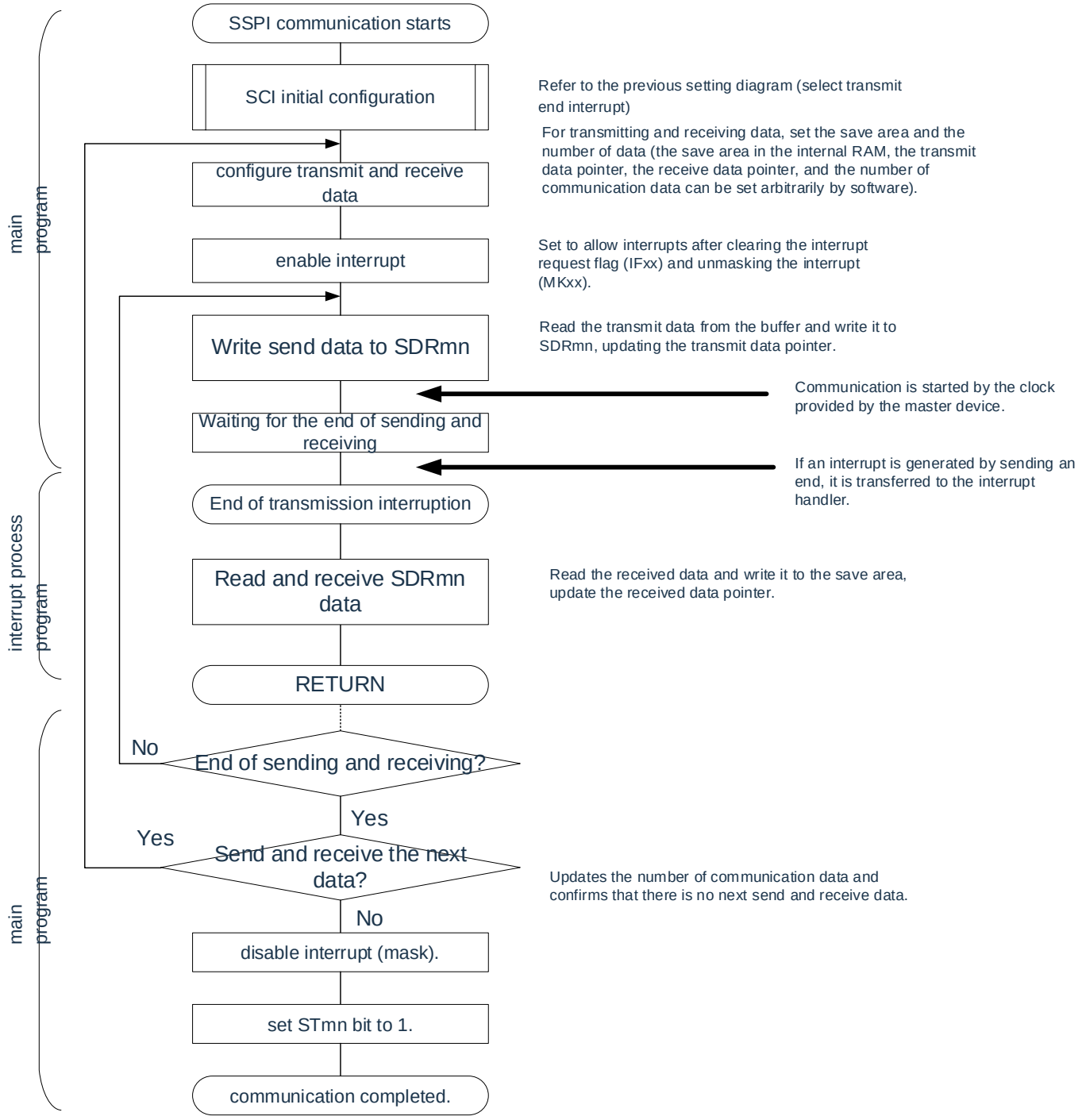


Remark: m: unit number (m=0, 1);

n: channel number (n=0, 1);

p: SSPI number (p=00, 01, 10, 11).

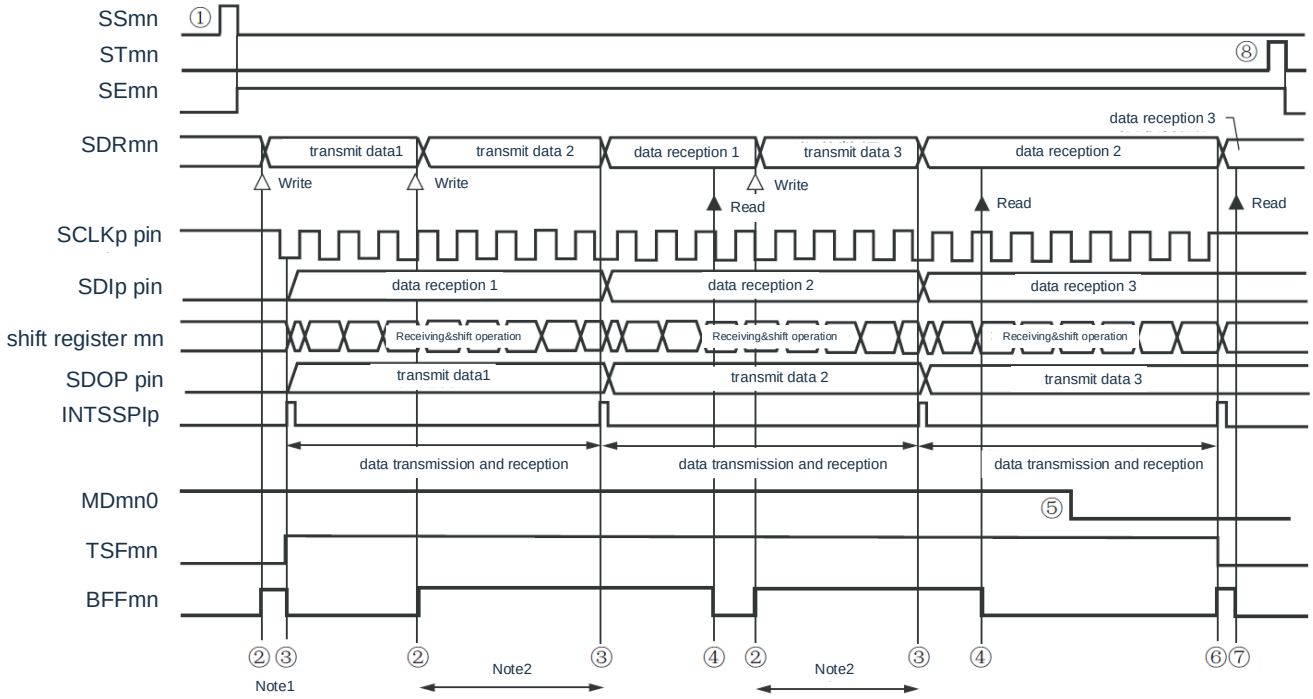
Figure 12-46: Flowchart of slave slave transmit and receive (single transmit and receive mode)



Notice: The SIOp register must be set for transmit data before the master device begins to output the clock.

(4) Process flow (continuous send and receive mode)

Figure 12-47: Timing diagram of slave transmit and receive (continuous transmit and receive mode) (type 1: DAPmn=0, CKPmn=0)



Note 1: If the transmit data is written to the SDRmn register during the time when the BFFmn bit of the serial status register mn (SSRmn) is “1” (when valid data is stored in the serial data register mn (SDRmn)), the transmit data is rewritten;

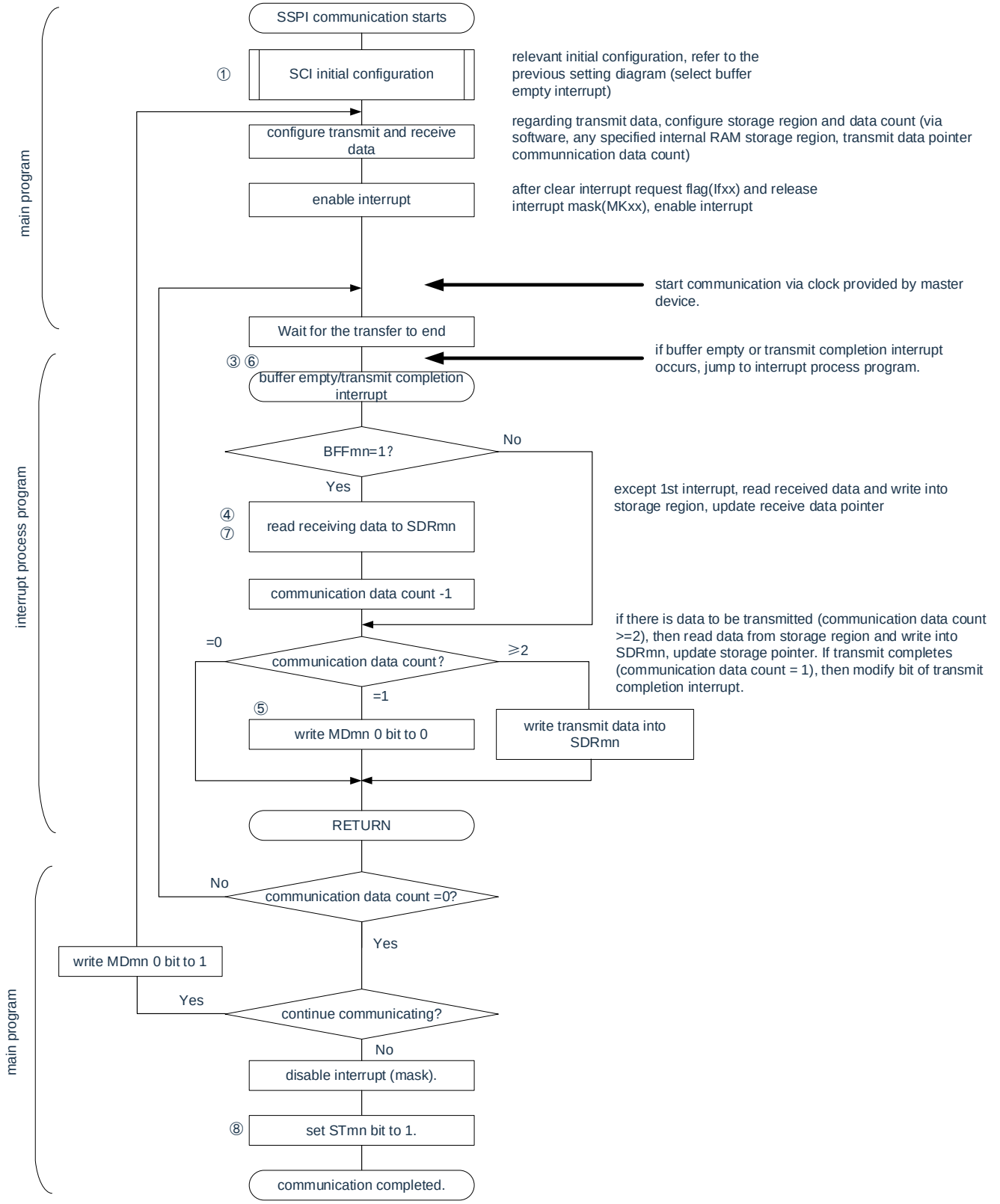
Note 2: If the SDRmn register is read during this period, the data can be read and sent. At this point, the transfer run is not affected.

Notice: It is possible to rewrite the MDmn0 bit of the Serial Mode Register mn (SMRmn) even during operation. However, in order to catch the end-of-transmission interrupt for the last data sent, the rewrite must be done before the transmission of the last bit begins;

Remark:

1. 1. ① to ⑧ in the figure correspond to ① to ⑧ in "Figure 12-48 Flowchart of Slave Transmit and Receive (Continuous Transmit and Receive Mode)";
2. m: unit number (m=0, 1);
   n: channel number (n=0, 1);
   p: SSPI number (p=00, 01, 10, 11).

Figure 12-48: Flowchart of slave transmit and receive (continuous transmit and receive mode)



Notice: The transmit data must be set to the SDRmn register before the master device starts outputting the clock.

Remark: ① to ⑧ in the figure correspond to① to ⑧ in “Figure 12-47 Timing Diagram of Slave Transmit and Receive (Continuous Transmit and Receive Mode)”.

12.5.7 Calculation of transmission clock frequency

3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11) communication transmission clock frequency can be calculated using the following calculation equations.

(1) Master

(Transmit clock frequency) = {Operation clock frequency of the target channel (F_{MCK})}÷(SDRmn [15:9] +1) ÷2[Hz]

(2) Slave

(Transmit clock frequency) = {Serial clock (SCLK) frequency provided by the master device} Note [Hz]

Remark:

1. The maximum allowable transmit clock frequency is F_{MCK}/6;
2. It is 0~127 because the value of SDRmn [15:9] is the value of bit15~9 (0000000B~1111111B) of Serial Data Registermn (SDRmn);
3. The operating clock (F_{MCK}) depends on bit15 (CKSmn) of the serial clock selection register m (SPSm) and the serial mode register mn (SMRmn).

Table 12-2: Selection of 3 serial I/O operation clock

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}=32MHz in operation
0	X	X	X	X	0	0	0	0	F_{CLK}	32MHz
	X	X	X	X	0	0	0	1	$F_{CLK}/2$	16MHz
	X	X	X	X	0	0	1	0	$F_{CLK}/2^2$	8MHz
	X	X	X	X	0	0	1	1	$F_{CLK}/2^3$	4MHz
	X	X	X	X	0	1	0	0	$F_{CLK}/2^4$	2MHz
	X	X	X	X	0	1	0	1	$F_{CLK}/2^5$	1MHz
	X	X	X	X	0	1	1	0	$F_{CLK}/2^6$	500KHz
	X	X	X	X	0	1	1	1	$F_{CLK}/2^7$	250KHz
	X	X	X	X	1	0	0	0	$F_{CLK}/2^8$	125KHz
	X	X	X	X	1	0	0	1	$F_{CLK}/2^9$	62.5KHz
	X	X	X	X	1	0	1	0	$F_{CLK}/2^{10}$	31.25KHz
	X	X	X	X	1	0	1	1	$F_{CLK}/2^{11}$	15.63KHz
	X	X	X	X	1	1	0	0	$F_{CLK}/2^{12}$	7.81KHz
	X	X	X	X	1	1	0	1	$F_{CLK}/2^{13}$	3.91KHz
	X	X	X	X	1	1	1	0	$F_{CLK}/2^{14}$	1.95KHz
	X	X	X	X	1	1	1	1	$F_{CLK}/2^{15}$	977Hz
1	0	0	0	0	X	X	X	X	F_{CLK}	32MHz
	0	0	0	1	X	X	X	X	$F_{CLK}/2$	16MHz
	0	0	1	0	X	X	X	X	$F_{CLK}/2^2$	8MHz
	0	0	1	1	X	X	X	X	$F_{CLK}/2^3$	4MHz
	0	1	0	0	X	X	X	X	$F_{CLK}/2^4$	2MHz
	0	1	0	1	X	X	X	X	$F_{CLK}/2^5$	1MHz
	0	1	1	0	X	X	X	X	$F_{CLK}/2^6$	500KHz
	0	1	1	1	X	X	X	X	$F_{CLK}/2^7$	250KHz
	1	0	0	0	X	X	X	X	$F_{CLK}/2^8$	125KHz
	1	0	0	1	X	X	X	X	$F_{CLK}/2^9$	62.5KHz
	1	0	1	0	X	X	X	X	$F_{CLK}/2^{10}$	31.25KHz
	1	0	1	1	X	X	X	X	$F_{CLK}/2^{11}$	15.63KHz
	1	1	0	0	X	X	X	X	$F_{CLK}/2^{12}$	7.81KHz
	1	1	0	1	X	X	X	X	$F_{CLK}/2^{13}$	3.91KHz
	1	1	1	0	X	X	X	X	$F_{CLK}/2^{14}$	1.95KHz
	1	1	1	1	X	X	X	X	$F_{CLK}/2^{15}$	977Hz

Note: When you change the clock selected as fCLK (change the value of the system clock control register (CKC)), you must stop the operation of the general-purpose serial communication unit (SCI) (serial channel stop register m (STm)=000FH) after making changes.

X: Ignore

m: unit number (m=0, 1);

n: channel number (n=0, 1).

12.5.8 Procedure for handling errors during 3-wire serial I/O communication (SSPI00, SSPI01, SSPI10, SSPI11)

The procedure for handling an error during 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11) communication is shown in Table 12-29.

Table 12-29: Processing steps when an overflow error occurs

Software operation	Hardware status	Remark
Read Serial Data Registermn (SDRmn). →	The BFFmn bit of the SSRmn register is "0" and channel n is receivable.	This is to prevent an overflow error from occurring if the next reception ends during error handling.
Read Serial Status Registermn (SSRmn).		Determines the type of error and reads the value for clearing the error flag.
Write "1" to the serial flag clear trigger register mn (SDIRmn). →	Clear the error flag.	Errors during read operations can only be cleared by writing the read value of the SSRmn register directly to the SDIRmn register.

Remark: m: unit number (m=0, 1);

n: channel number (n=0, 1).

12.6 Operation of clock-synchronous serial communication with slave selection input function

Channel 0 of SCI0 is a channel that supports clock-synchronous serial communication with a slave select input function.

[Data transmission and reception]

(1) Data length of 7~16 bits

(2) Phase control of transmit/receive data

(3) MSB/LSB preferred option

(4) Level setting for transmit/receive data

[Clock control]

(1) Phase control of I/O clock

(2) Setting of transfer period by prescaler and internal counter

(3) Maximum transmission rate[Note] Slave communication: Max.F_{MCK}/6

[Interrupt function]

Transfer end interrupt, buffer empty interrupt

[Error detection flag]

Overflow error

The slave selection input function has the following 3 types of communication operation:

(1) Slave transmission (refer to 12.6.1)

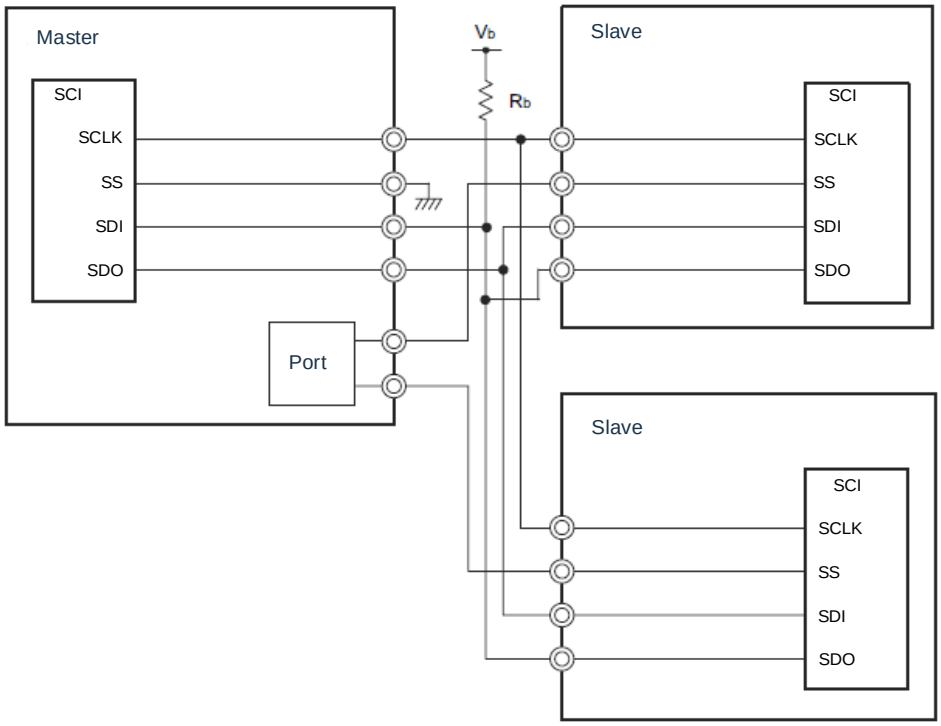
(2) Slave reception (refer to 12.6.2)

(3) Slave transmission and reception (refer to 12.6.3)

Notice: It must be used within the range that satisfies the SCLK cycle time (T_{KCY}) characteristics. Refer to the datasheet for details.

The slave selection input function enables a master to communicate with multiple slave devices. The master outputs a slave selection signal to a slave device (1) of the communication object, and each slave device judges whether it is selected as the communication object and controls the output of the SDO pin. When selected as a slave device of the communication object, the SDO pin is able to communicate sending data to the master device; When a slave device is not selected as a communication object, the SDO pin becomes a high output, so it is necessary to set the SDO pin to Nch-O.D and pull up the node when connecting multiple slaves. In addition, transmission and reception are not performed even if the serial clock of the master device is input.

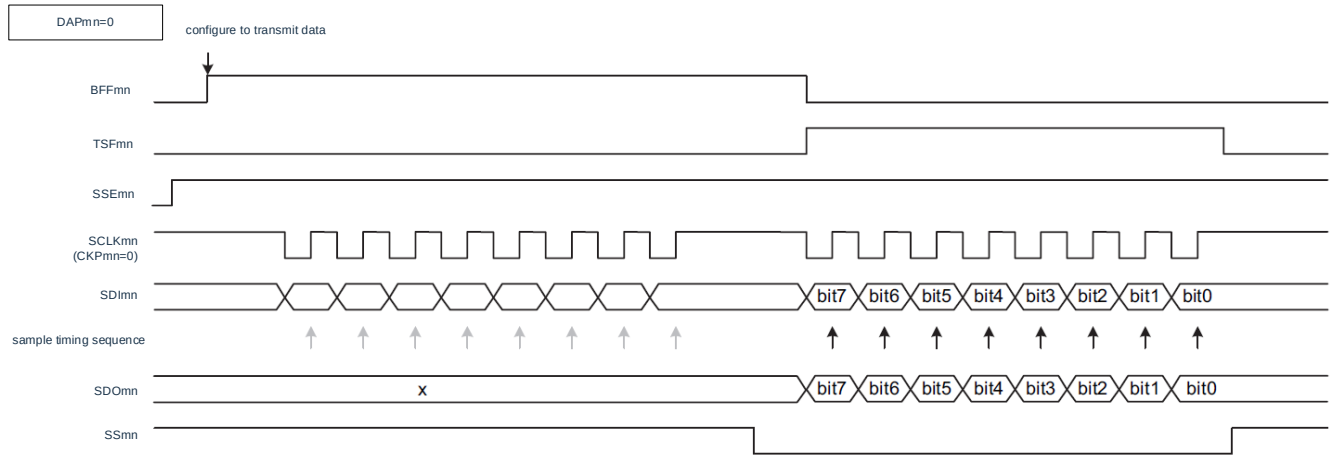
Figure 12-49: Example of slave selection input function



Notice:

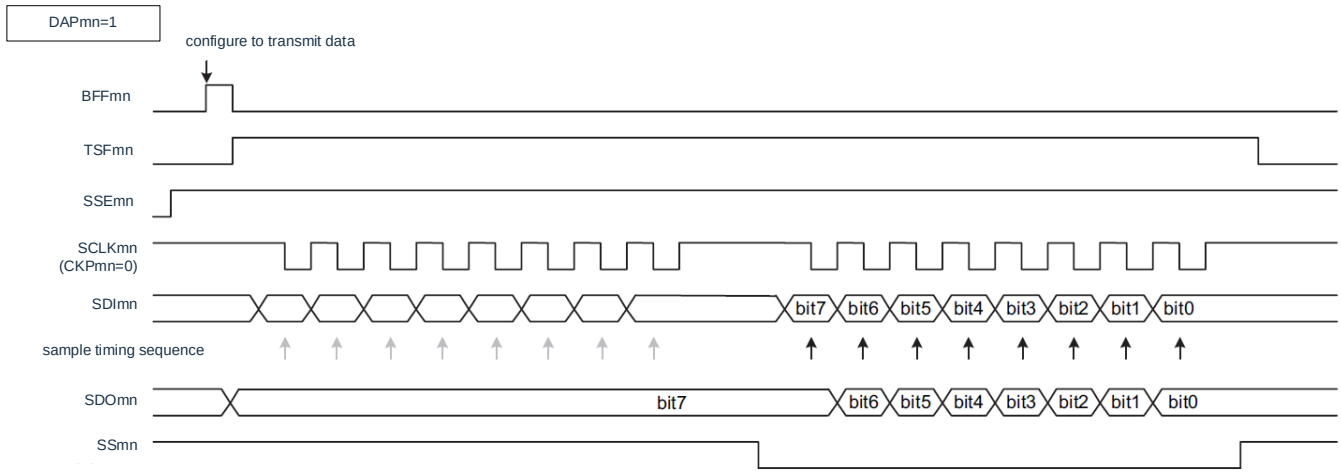
1. Select the SDO00 pin to N-channel open drain output mode;
2. A slave selection signal must be output by the operation of the port.

Figure 12-50: Timing diagram of slave select input function



When SSmn is high, no transmission is performed even at the falling edge of SCKmn (serial clock), and no sampling of received data is performed synchronously with the rising edge.

When SSmn is low, data is output (shifted) synchronously with the falling edge of the serial clock and data is received synchronously with the rising edge.



When the DAPmn bit is "1", if the data is sent when SSmn is set to be high, the initial data (bit7) is provided to the data output. However, even the rising edge of the SCLKmn (serial clock) is not shifted, and the received data is not sampled in sync with the falling edge. If SSmn goes low, the output data is synchronized with the next rising edge (shift) and the data is received synchronously with the falling edge.

Remark: m: unit number (m=0);

n: channel number (n=0).

12.6.1 Slave transmission

Slave transmission refers to the operation of this product to send data to other devices in the state of transmitting clocks from other device inputs.

Table 12-30: Slave transmission

Slave select input function	SSPI00
Target channel	Channel 0 of SCI0
Pins used	SCLKOI00, SDO00, SS00
Interrupt	INTSSPI00
	Selectable transmission end interrupt (single transmission mode) or buffer empty interrupt (continuous transmission mode).
Error detection flag	Only the overflow error detection flag (OVFmn).
Transfer data length	7~16 bits
Transfer rate	Max.F_{MCK}/6[Hz]Note1,2
Data phase	It can be selected by the DAPmn bit of the SCRmn register. DAPmn=0: Start data output when the serial clock starts running. DAPmn=1: Start data output half a clock before the serial clock starts running.
Clock phase	It can be selected by the CKPmn bit of the SCRmn register. CKPmn=0: Non-reverse CKPmn=1: Reverse
Data direction	MSB or LSB first
Slave select input function	The operation of the slave selection function can be selected.

Note 1: The maximum transfer rate is F_{MCK}/6 [Hz] because the external serial clock input to the SCLK00 pin is internally sampled and then used.

Note 2: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheret).

F_{MCK}: Operation clock frequency of target channel

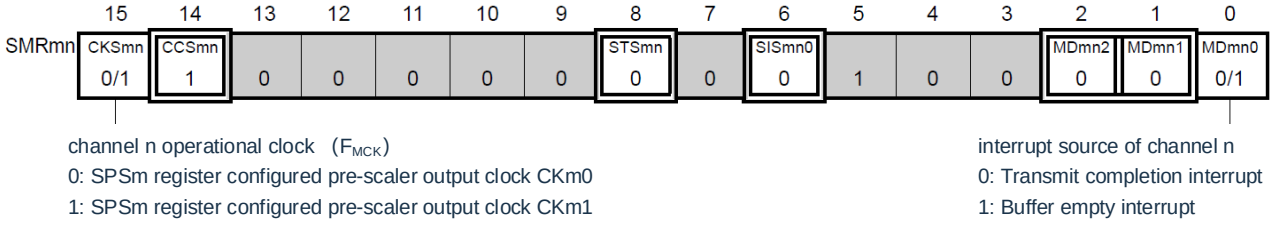
m: unit number (m=0);

n: channel number (n=0).

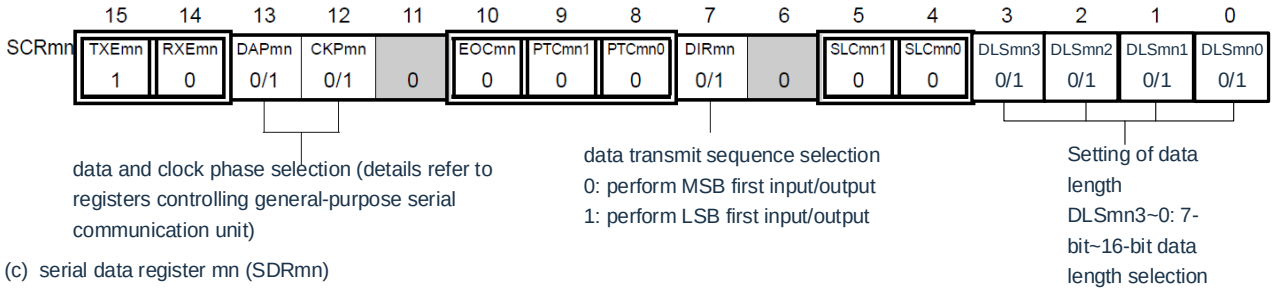
(1) Register setting

Figure 12-51: Example of register settings when slave select input function (SSPI00) slave transmits (1/2)

(a) serial mode register mn (SMRmn)

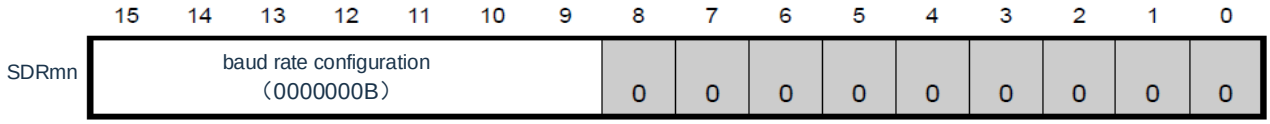


(b) serial communication operation configuration register mn(SCRmn)

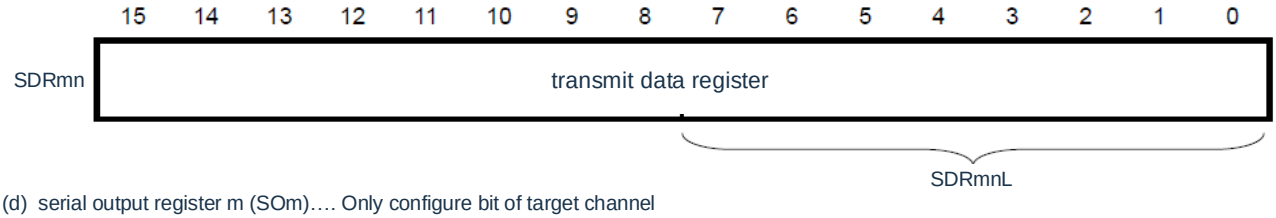


(c) serial data register mn (SDRmn)

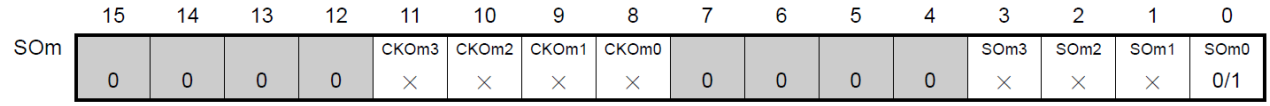
(1) When operation stops (SEmn=0)



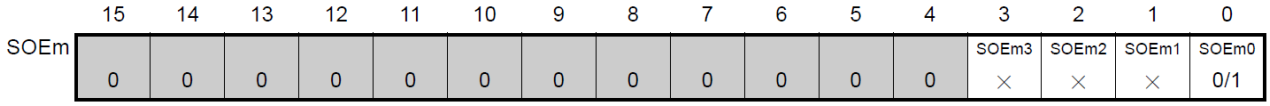
(2) During operation (SEmn=1) (Lower 8 bits: SDRmnL)



(d) serial output register m (SOm)…. Only configure bit of target channel



(e) serial output enable register m (SOEm)…. Only set bit of target channel to "1".



Remark: m: unit number (m=0);

n: channel number (n=0);

p: SSPI number (p=00);

□ : Fixed set in SSPI master receive mode; ■ : Cannot be set (initial value is set);

×: This is a bit that cannot be used in this mode (sets the initial value if it is not used in other modes either);

0/1: Set “0” or “1” according to the user’s purpose.

Figure 12-51: Example of register settings when slave select input function (SSPI00) slave transmits (2/2)

(f) serial channel start register m (SSm) ···. Only set bit of 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	0	0	SSm1 ×	SSm0 0/1

(g) Input switching control register (ISC) ..... This is the control of the SS00 pin of the SSPI00 slave channel (channel 0 of unit 0).

ISC	7	6	5	4	3	2	1	0
	SSIE00 0/1	0	0	0	0	0	ISC1 0/1	ISC0 0/1

0: SS00 pin input is invalid
1: SS00 pin input is valid

Remark: m: unit number (m=0);

n: channel number (n=0);

p: SSPI number (p=00);

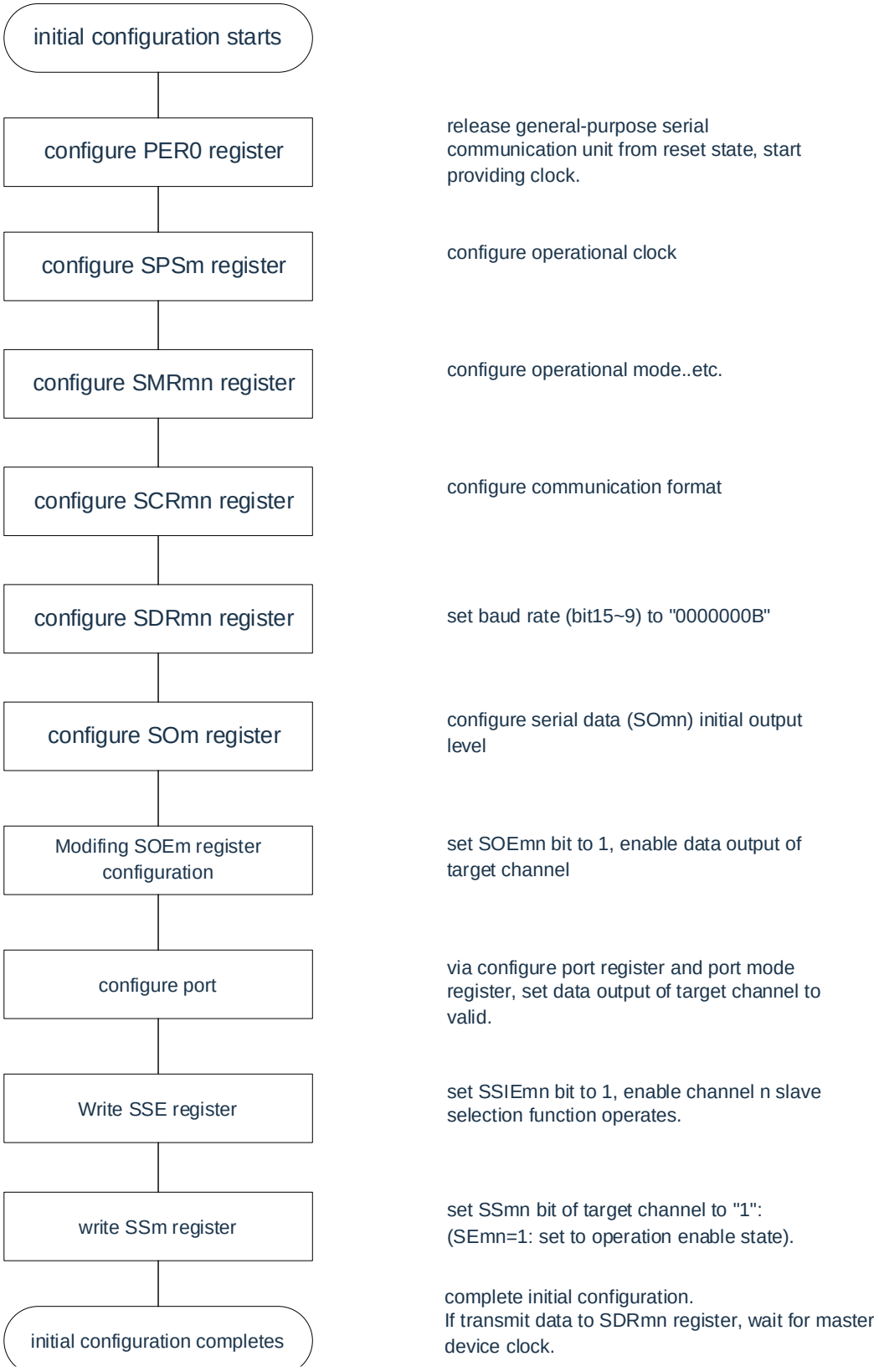
□: Fixed set in SSPI master receive mode; ■ : Cannot be set (initial value is set);

×: This is a bit that cannot be used in this mode (sets the initial value if it is not used in other modes either);

0/1: Set “0” or “1” according to the user’s purpose.

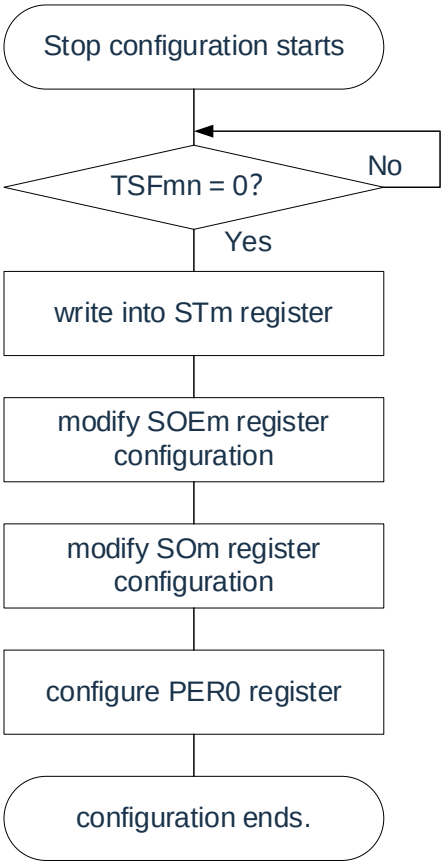
(2) Procedure

Figure 12-52: Initial setup steps for slave transmission



Remark: m: unit number (m=0); n: channel number (n=0); p: SSPI number (p=00).

Figure 12-53: Stop steps for slave transmission

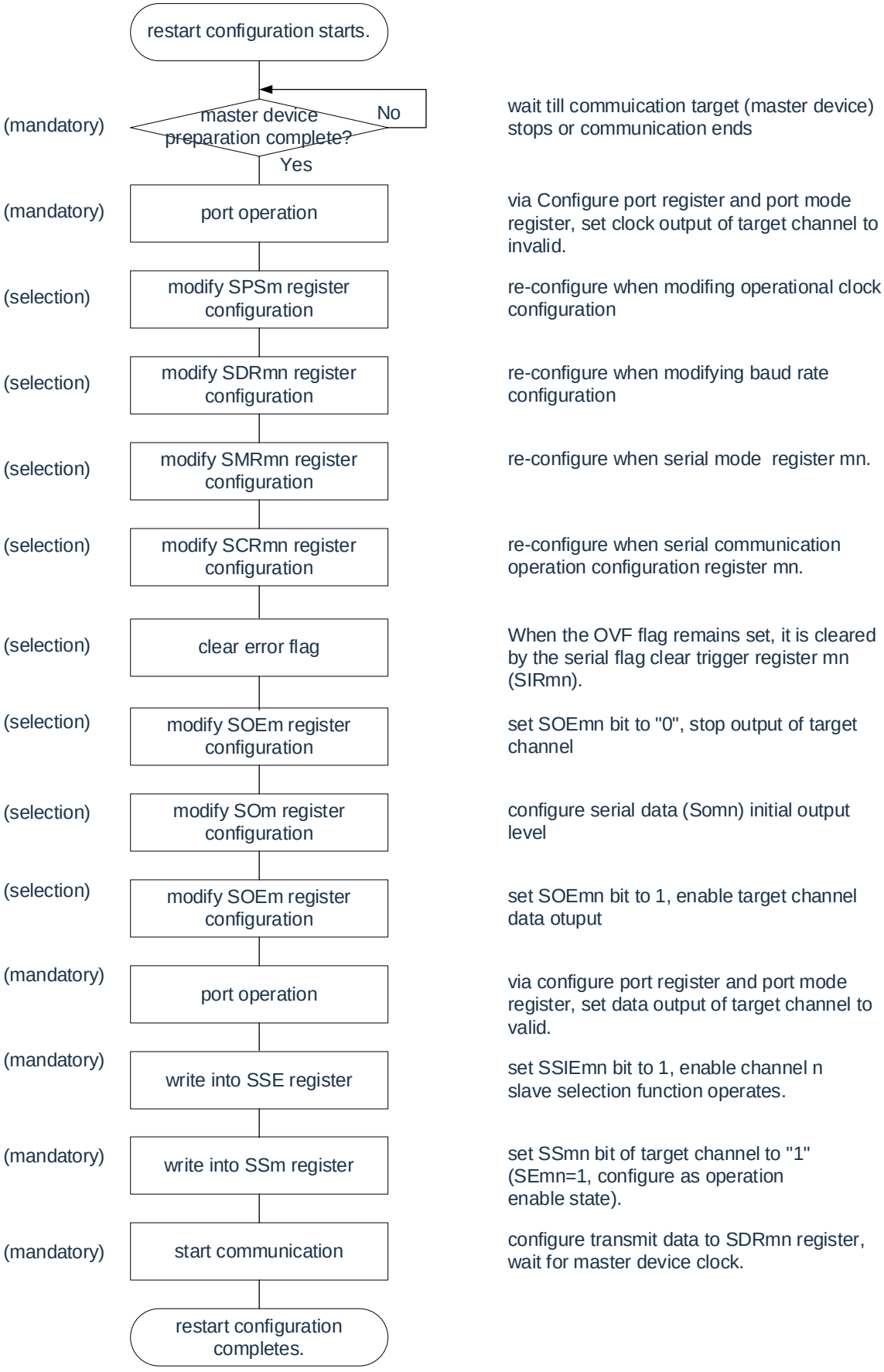


(selection)	if there are ongoing data transmission, then wait till transmission completed. (if need urgent stop, then no need to wait).
(mandatory)	set STmm bit of target channel to 1. (SEmn=0: set to operation stop state).
(mandatory)	set SOEmn bit to 0, stop output of target channel
(selection)	when emergency stop, based on needs, modify serial data (SOmn) level of the target channel.
(selection)	stop clock of universal serial communication unit, set to reset state
	finish termination configuration, enter into next processing.

Notice: If PER0 is rewritten in the abort setting to stop supplying the clock, the initial setting must be made after waiting until the communication object (master device) stops or the communication is finished, instead of restarting the setting;

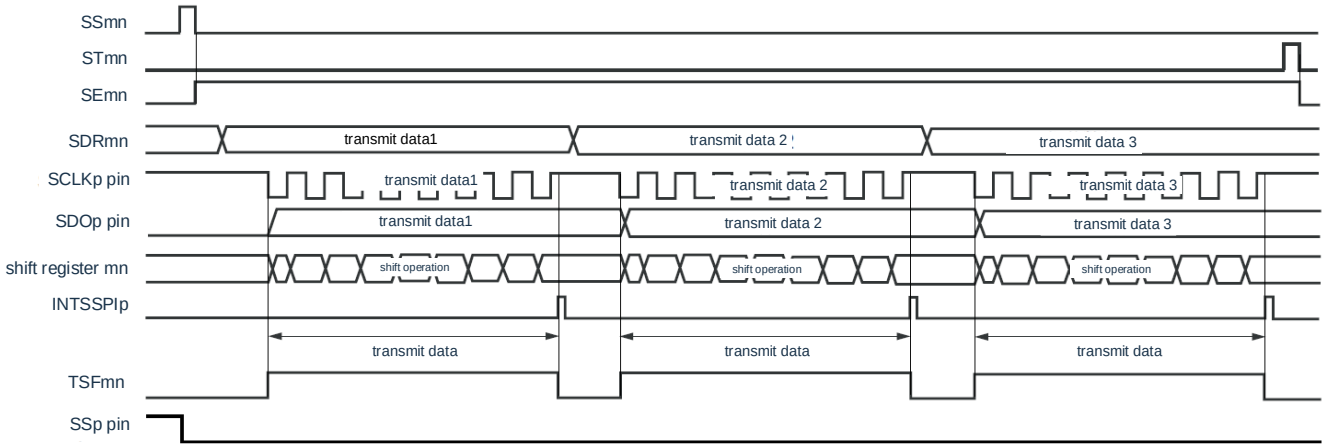
Remark: m: unit number (m=0); n: channel number (n=0); p: SSPI number (p=00);

Figure 12-54: Restart steps for slave transmission



(3) Process flow (single transmit mode)

Figure 12-55: Timing diagram of slave transmission (single transmit mode) (type 1: DAPmn=0, CKPmn=0)

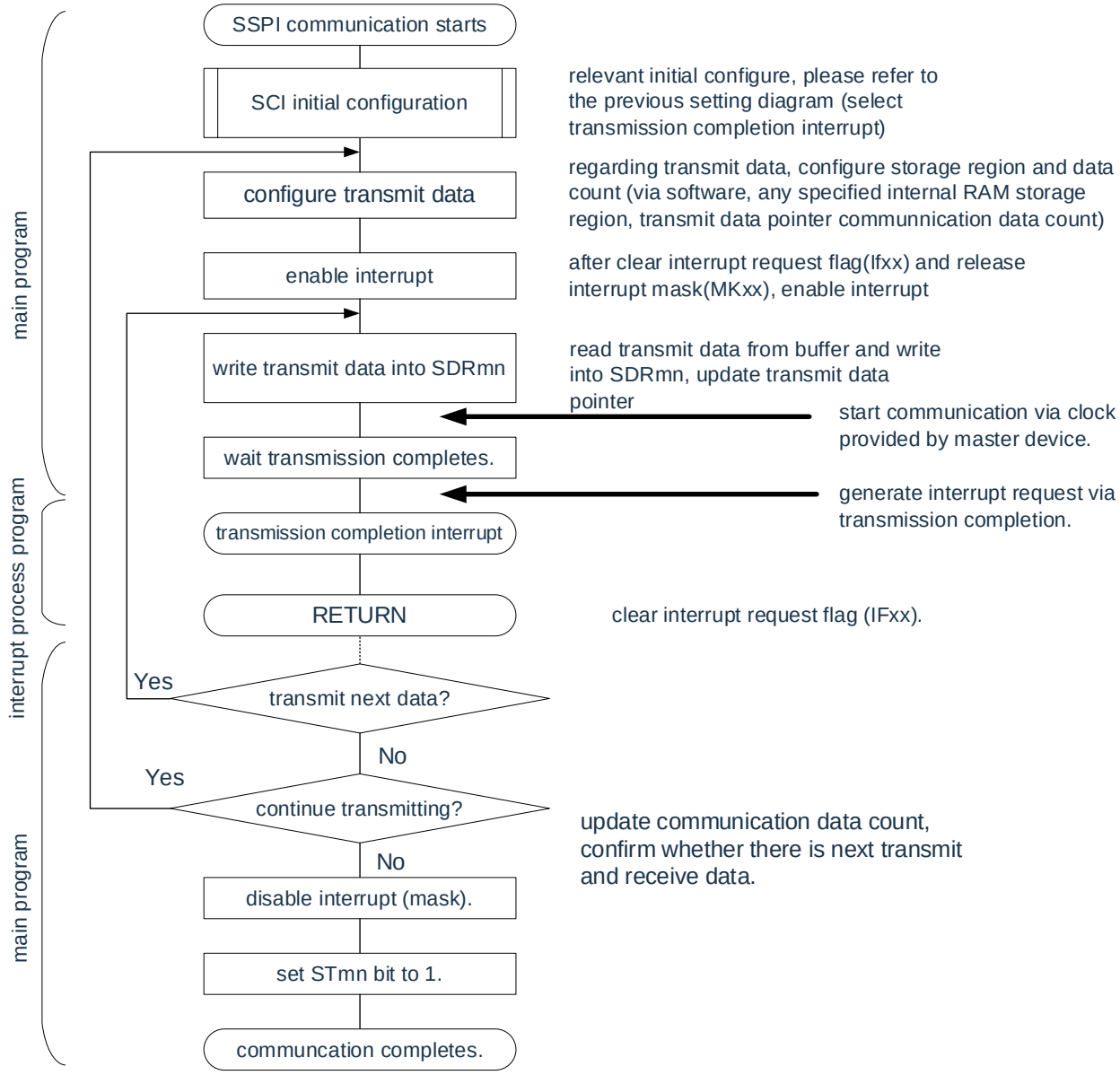


Remark: m: unit number (m=0);

n: channel number (n=0);

p: SSPI number (p=00).

Figure 12-56: Flowchart of slave transmission (single transmit mode)



Remark: m: unit number (m=0);

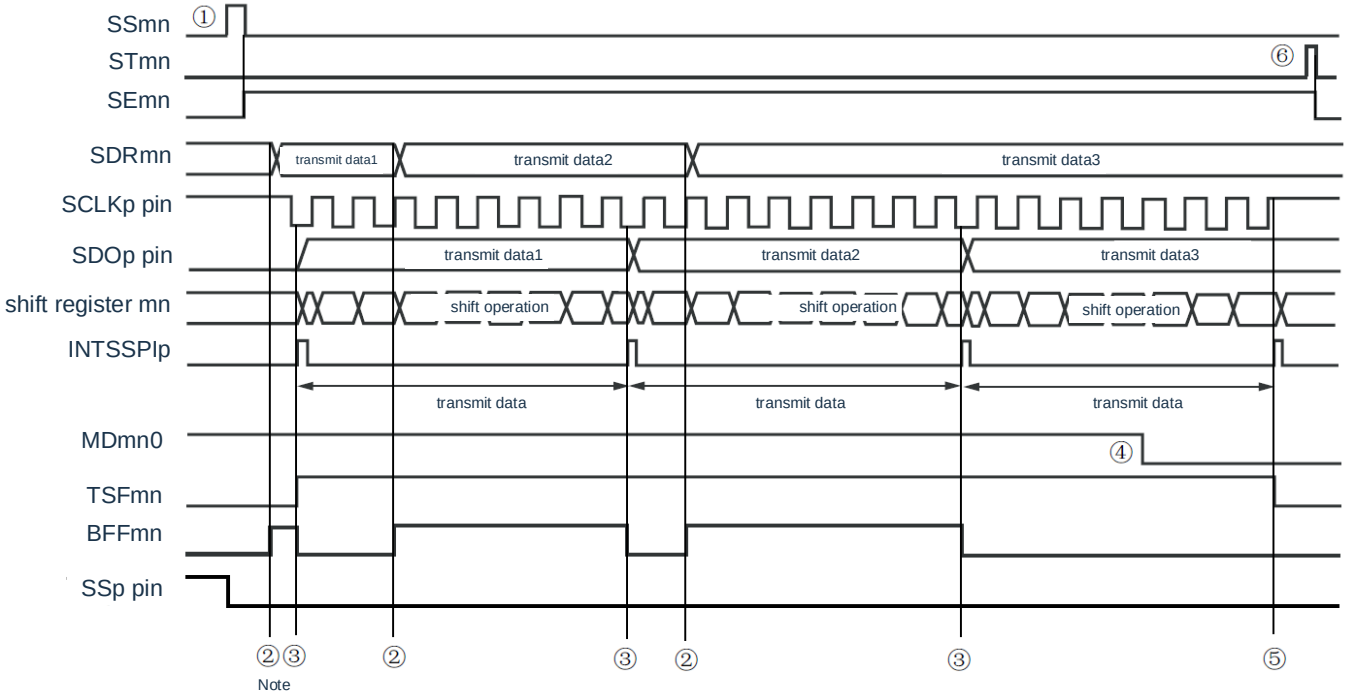
n: channel number (n=0);

p: SSPI number (p=00).

(4) Process flow (continuous transmit mode)

Figure 12-57: Timing diagram of slave transmission (continuous transmit mode)

(type 1: DAPmn= 0, CKPmn = 0)



Note: If the transmit data is written to the SDRmn register during the time when the BFFmn bit of the serial status register mn (SSRmn) is “1” (when valid data is stored in the serial data register mn (SDRmn)), the transmit data is rewritten.

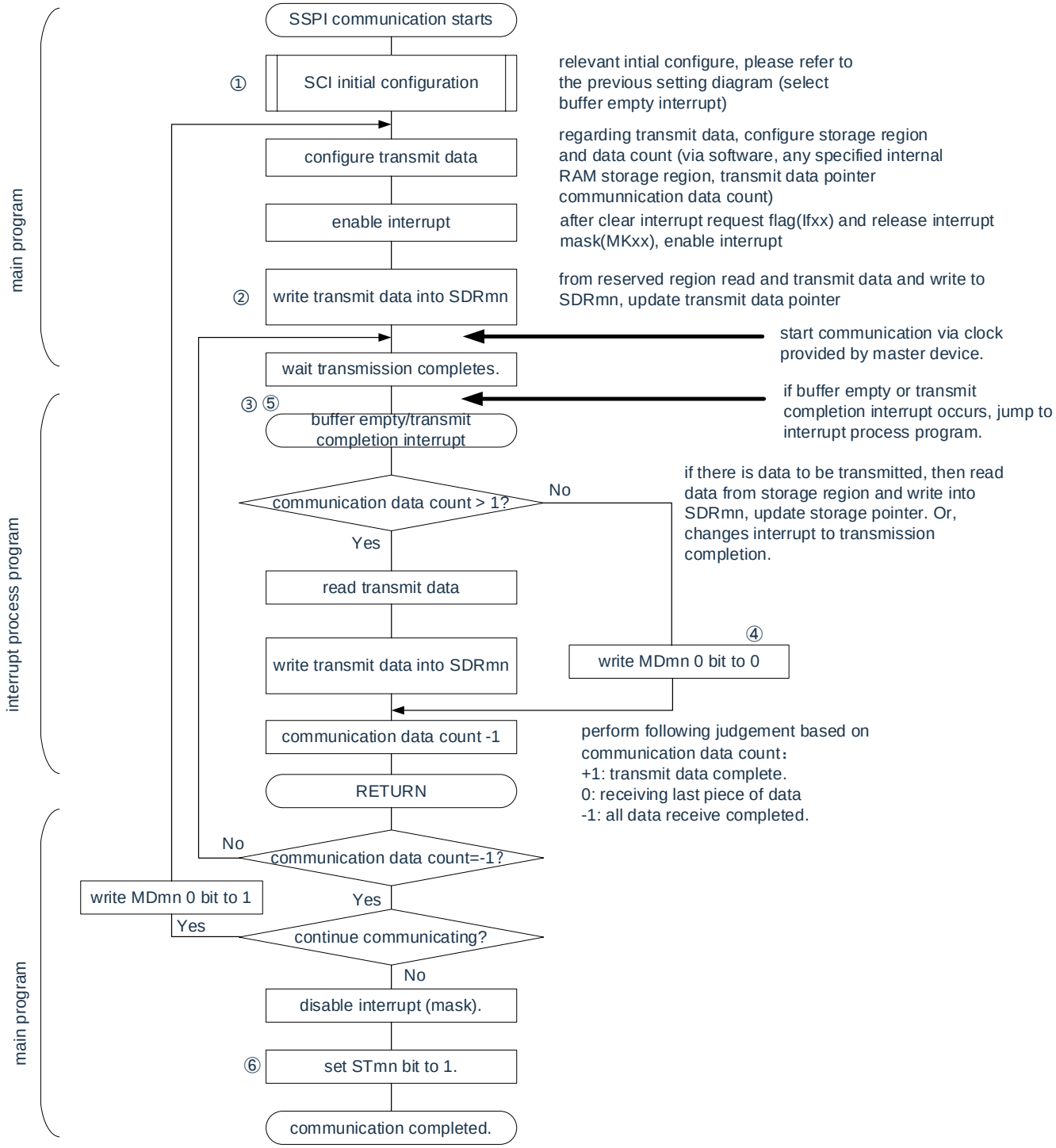
Notice: The MDmn0 bit of the Serial Mode Register mn (SMRmn) can be rewritten even during operation. However, the rewrite must be done before starting to transmit the last bit.

Remark: m: unit number (m=0);

n: channel number (n=0);

p: SSPI number (p=00).

Figure 12-58: Flowchart of slave transmission (continuous transmit mode)



Remark:① to ⑥ correspond to ① to ⑥ in “Figure 12-57 Timing Diagram of Slave Transmission (Continuous Transmit Mode)”.

m: unit number (m=0);

n: channel number (n=0);

p: SSPI number (p=00).

12.6.2 Slave reception

Slave reception refers to the operation of this product receiving data from other devices in the state of transmitting clocks from other devices.

Table 12-31: Slave reception

Slave select input function	SSPI00
Target channel	Channel 0 of SCI0
Pins used	SCLKOI00, SDI00, SS00
Interrupt	INTSSPI00
	Limited to end-of-transmit interrupts (buffer empty interrupts are prohibited).
Error detection flag	Only the overflow error detection flag (OVFmn).
Transfer data length	7~16 bits
Transfer rate	Max.F_{MCK}/6[Hz]Note1,2
Data phase	It can be selected by the DAPmn bit of the SCRmn register. DAPmn=0: Start data output when the serial clock starts running. DAPmn=1: Start data output half a clock before the serial clock starts running.
Clock phase	It can be selected by the CKPmn bit of the SCRmn register. CKPmn=0: Non-reverse CKPmn=1: Reverse
Data direction	MSB or LSB first
Slave select input function	The operation of the slave selection input function can be selected.

Note 1: The maximum transfer rate is F_{MCK}/6 [Hz] because the external serial clock input to the SCLK00 pin is internally sampled and then used.

Note 2: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheret).

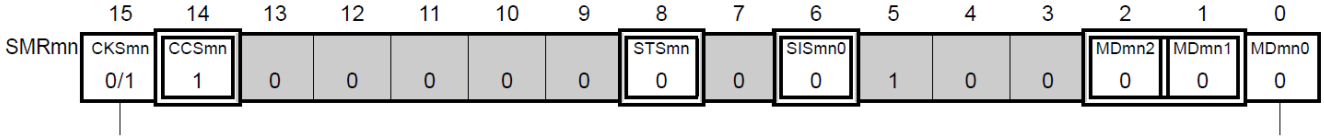
F_{MCK}: Operation clock frequency of target channel

m: unit number (m=0) n: channel number (n=0).

(1) Register setting

Figure 12-59: Slave select input function (SSPI00)
Example of register setting content when slave receives (1/2)

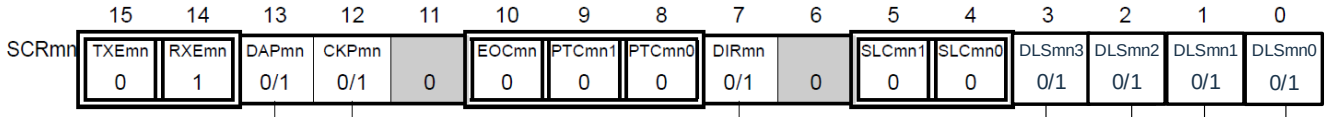
(a) serial mode register mn (SMRmn)



channel n operational clock (F_{MCK})
0: SPSm register configured pre-scaler output clock CKm0
1: SPSm register configured pre-scaler output clock CKm1

interrupt source of channel n
0: Transmit completion interrupt
1: Buffer empty interrupt

(b) serial communication operation configuration register mn(SCRmn)



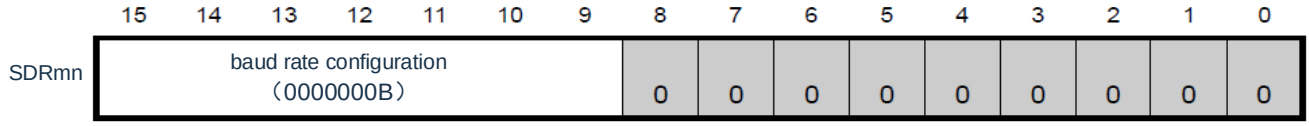
data and clock phase selection (details refer to registers controlling general-purpose serial communication unit)

data transmit sequence selection
0: perform MSB first input/output
1: perform LSB first input/output

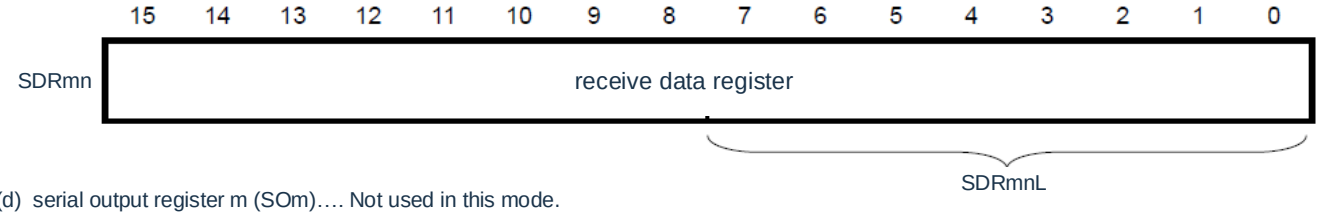
Setting of data length
DLSmn3~0: 7-bit~16-bit data length selection

(c) serial data register mn (SDRmn)

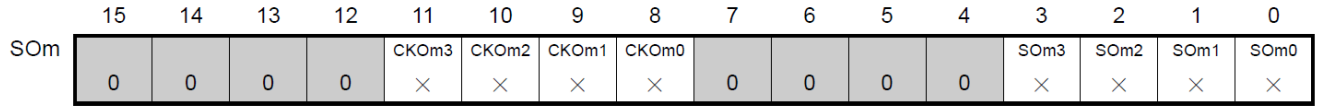
(1) When operation stops (SEmn=0)



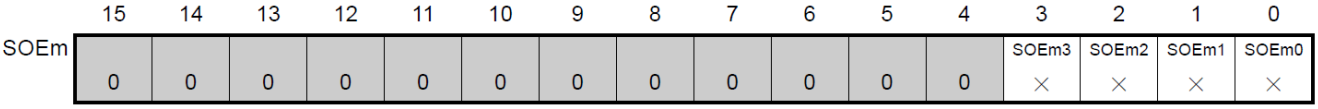
(2) During operation (SEmn=1) (Lower 8 bits: SDRmnL)



(d) serial output register m (SOm)…. Not used in this mode.



(e) serial output enable register m (SOEm)…. Not used in this mode.



Remark: m: unit number (m=0);
n: channel number (n=0);
p: SSPI number (p=00);
□ : Fixed set in SSPI master receive mode; ■ : Cannot be set (initial value is set);
×: This is a bit that cannot be used in this mode (sets the initial value if it is not used in other modes either);
0/1: Set “0” or “1” according to the user’s purpose.

Figure 12-59: Slave select input function (SSPI00)
Example of register setting content when slave receives (2/2)

(f) serial channel start register m (SSm) ···. Only set bit of 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	0	0	SSm1 ×	SSm0 0/1

(g) Input switching control register (ISC) ..... This is the control of the SS00 pin of the SSPI00 slave channel (channel 0 of unit 0).

	7	6	5	4	3	2	1	0
ISC	SSIE00 0/1	0	0	0	0	0	ISC1 0/1	ISC0 0/1

0: SS00 pin input is invalid
1: SS00 pin input is valid

Remark: m: unit number (m=0);

n: channel number (n=0);

p: SSPI number (p=00);

□ : Fixed set in SSPI master receive mode; ■ : Cannot be set (initial value is set);

×: This is a bit that cannot be used in this mode (sets the initial value if it is not used in other modes either);

0/1: Set “0” or “1” according to the user’s purpose.

(2) Procedure

Figure 12-60: Initial setup steps for slave reception

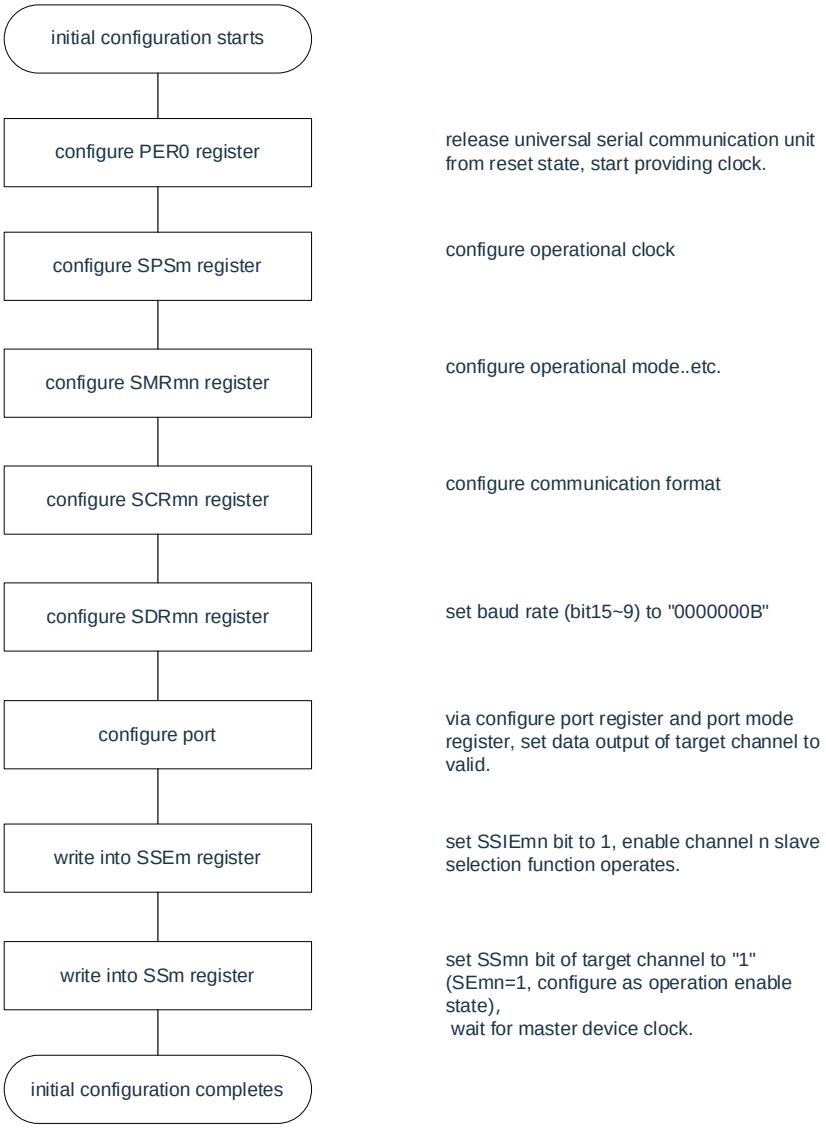
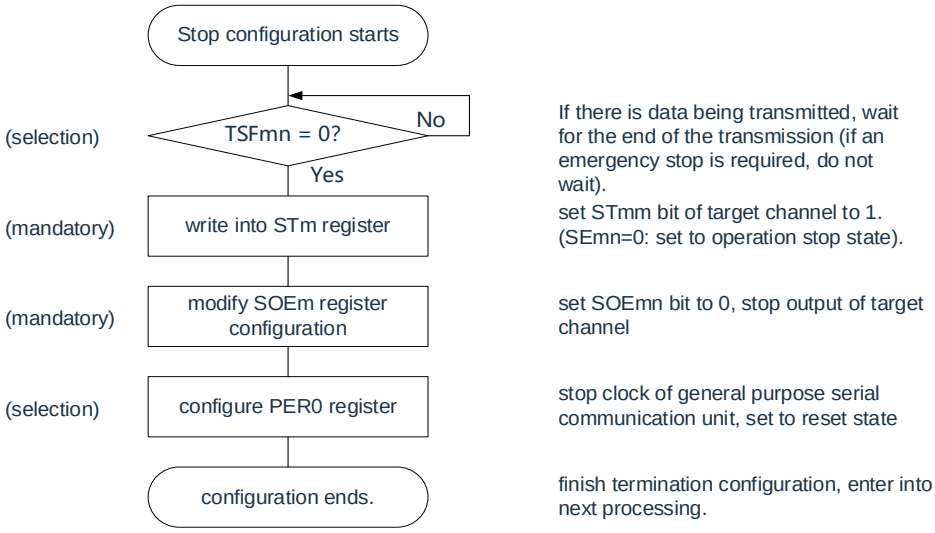
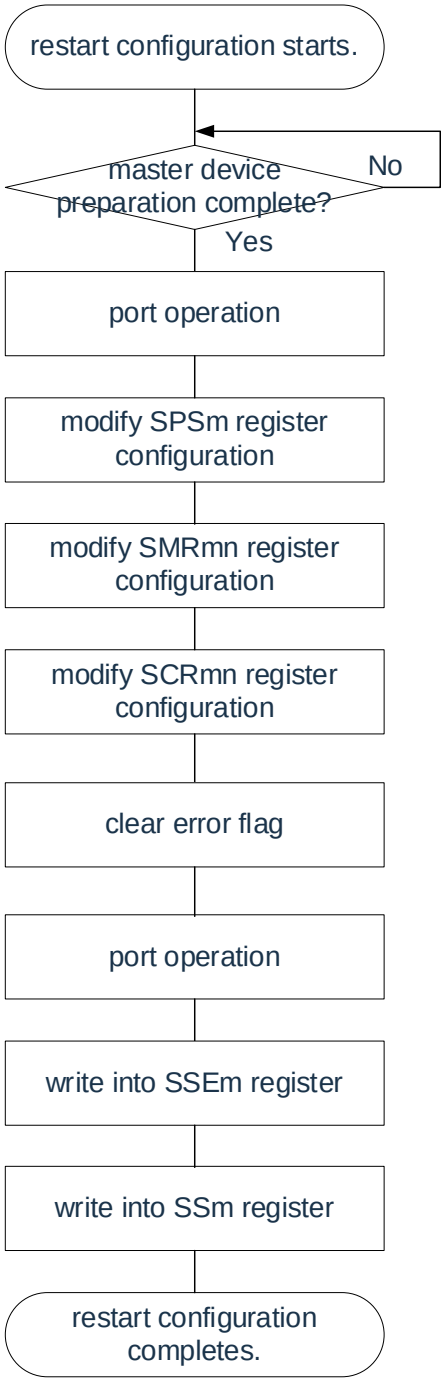


Figure 12-61: Stop steps for slave reception



Remark: m: unit number (m=0); n: channel number (n=0); p: SSPI number (p=00).

Figure 12-62: Restart steps for slave reception

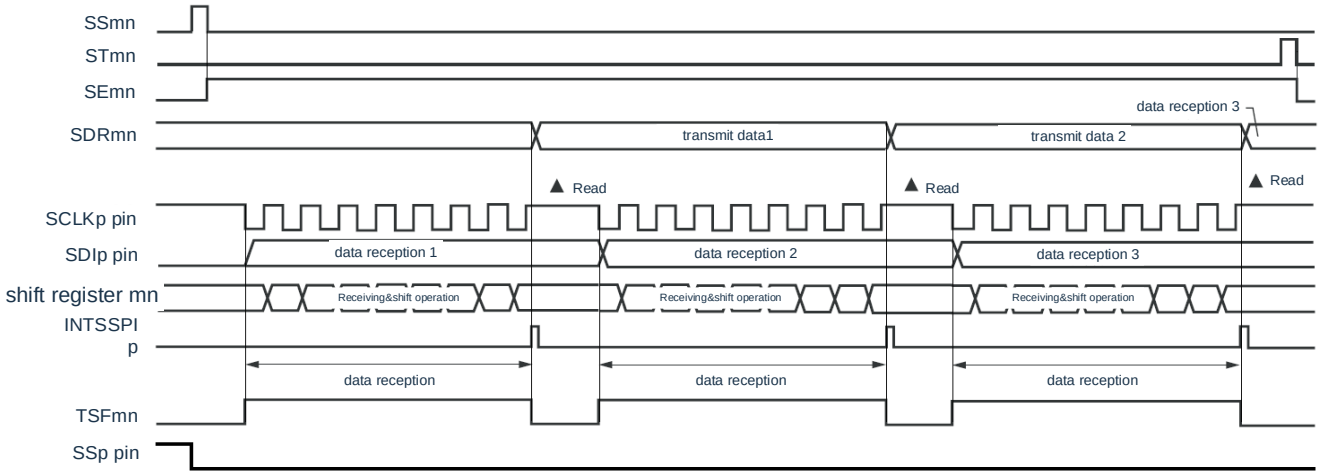


(mandatory)	master device preparation complete?	wait till commuication target (master device) stops or communication ends
(mandatory)	port operation	via Configure port register and port mode register, set clock output of target channel to invalid.
(selection)	modify SPSm register configuration	re-configure when modifing operational clock configuration
(selection)	modify SMRmn register configuration	re-configure when serial mode register mn.
(selection)	modify SCRmn register configuration	re-configure when serial communication operation configuration register mn.
(selection)	clear error flag	When the OVF flag remains set, it is cleared by the serial flag clear trigger register mn (SIRmn).
(mandatory)	port operation	via Configure port register and port mode register, set data output of target channel to valid.
(mandatory)	write into SSEm register	set SSIEmn bit to 1, enable channel 0 slave selection function operates.
(mandatory)	write into SSm register	set SSmn bit of target channel to "1" (Semn=1, configure as operation enable state), wait for master device clock.
	restart configuration completes.	

Remark: m: unit number (m=0); n: channel number (n=0); p: SSPI number (p=00).

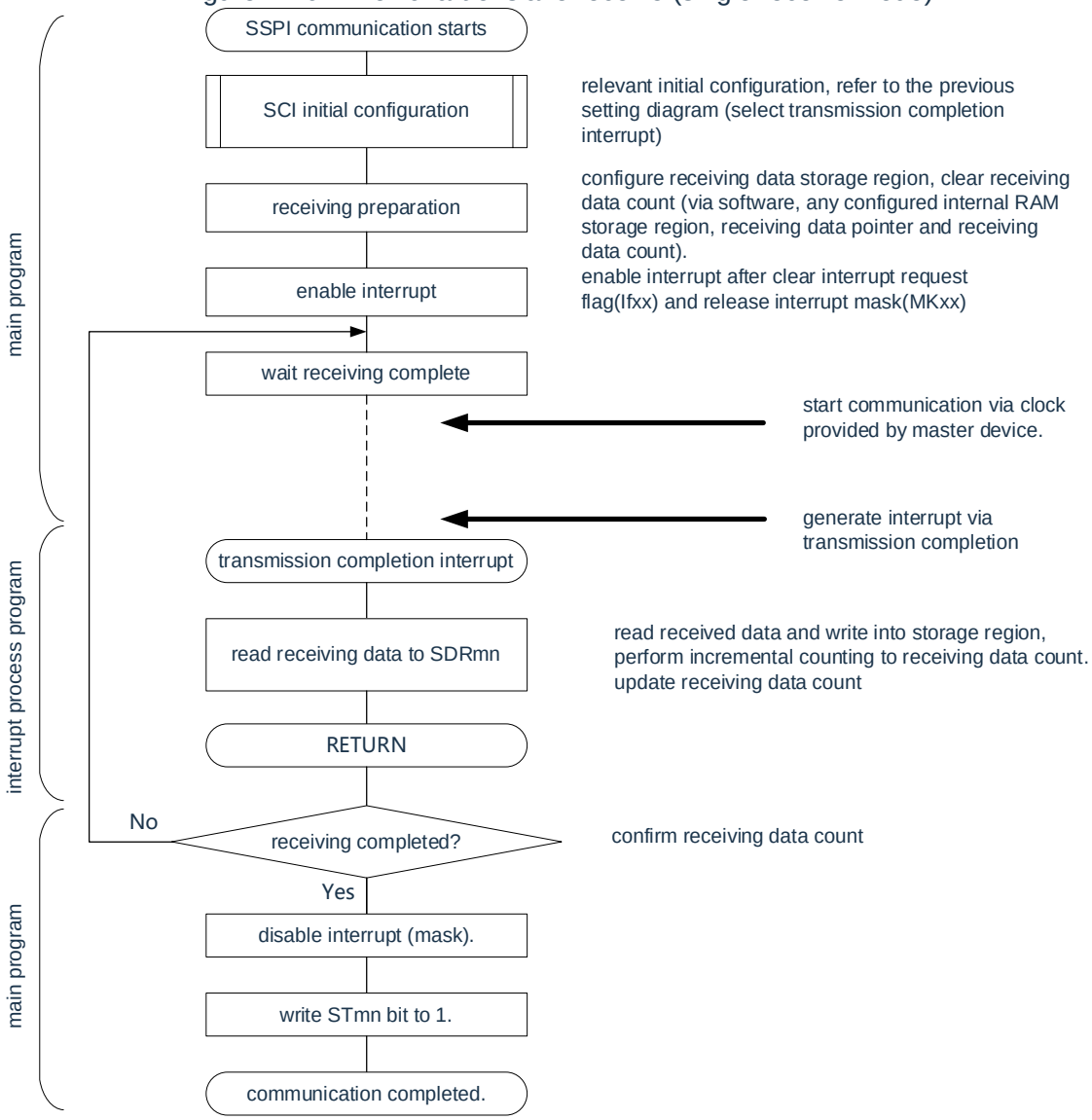
(3) Process flow (single receive mode)

Figure 12-63: Timing diagram of slave receive (single receive mode) (type 1: DAPmn= 0, CKPmn = 0)



Remark: m: unit number (m=0); n: channel number (n=0); p: SSPI number (p=00).

Figure 12-64: Flowchart of slave receive (single receive mode)



12.6.3 Slave transmission and reception

Slave transmission and reception refers to the operation of data transmission and reception of this product and other devices in the state of input transmission clock from other devices.

Table 12-32: Slave transmission and reception

Slave select input function	SSPI00
Target channel	Channel 0 of SCI0
Pins used	SCLKOI00, SDI00, SDO00, SS00
Interrupt	INTSSPI00
	Selectable transmission end interrupt (single transmission mode) or buffer empty interrupt (continuous transmission mode).
Error detection flag	Only the overflow error detection flag (OVFmn).
Transfer data length	7~16 bits
Transfer rate	Max.F_{MCK}/6[Hz]Note 1,2
Data phase	It can be selected by the DAPmn bit of the SCRmn register. DAPmn=0: Start data output when the serial clock starts running. DAPmn=1: Start data output half a clock before the serial clock starts running.
Clock phase	It can be selected by the CKPmn bit of the SCRmn register. CKPmn=0: Non-reverse CKPmn=1: Reverse
Data direction	MSB or LSB first
Slave select input function	The operation of the slave selection input function can be selected.

Note 1: The maximum transfer rate is F_{MCK}/6 [Hz] because the external serial clock input to the SCLK00 pin is internally sampled and then used.

Note 2: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheret).

F_{MCK}: Operation clock frequency of target channel

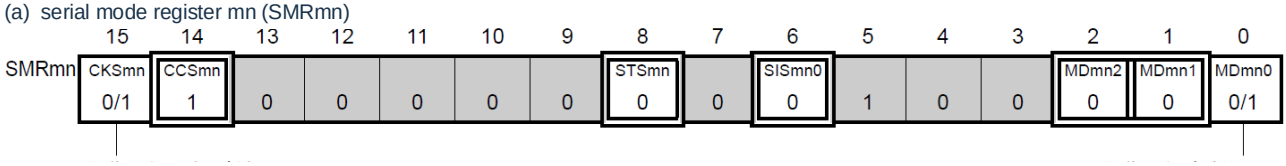
m: unit number (m=0);

n: channel number (n=0).

(1) Register setting

Figure 12-65: Slave selection input function (SSPI00)
Example of register setting content when slave transmits and receives (1/2)

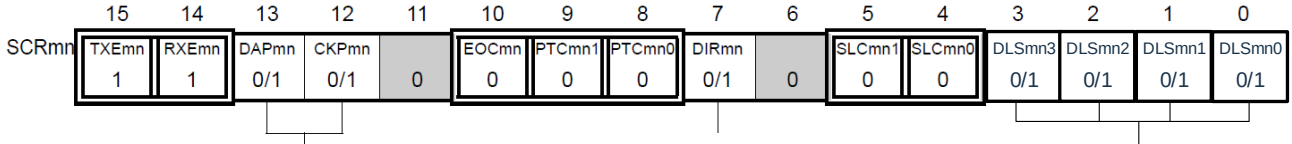
(a) serial mode register mn (SMRmn)



channel n operational clock (F_{MCK})
0: SPSm register configured pre-scaler output clock CKm0
1: SPSm register configured pre-scaler output clock CKm1

interrupt source of channel n
0: Transmit completion interrupt
1: Buffer empty interrupt

(b) serial communication operation configuration register mn(SCRmn)



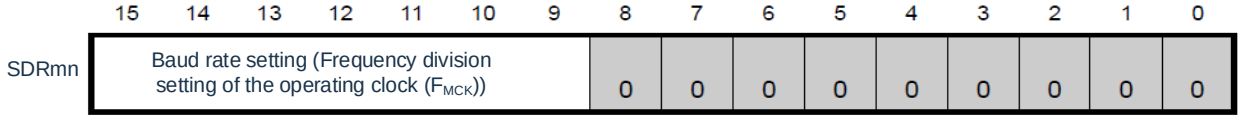
data and clock phase selection (details refer to registers controlling general-purpose serial communication unit)

data transmit sequence selection
0: perform MSB first input/output
1: perform LSB first input/output

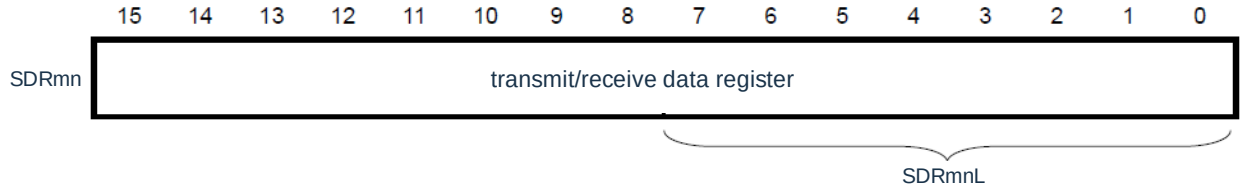
Setting of data length
DLSmn3~0: 7-bit~16-bit data length selection

(c) serial data register mn (SDRmn)

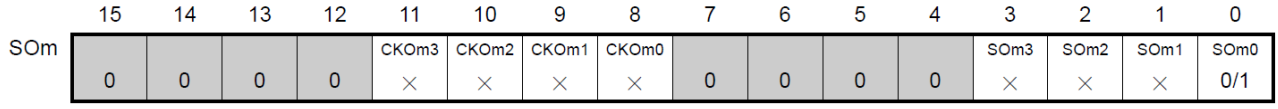
(1) When operation stops (SEmn=0)



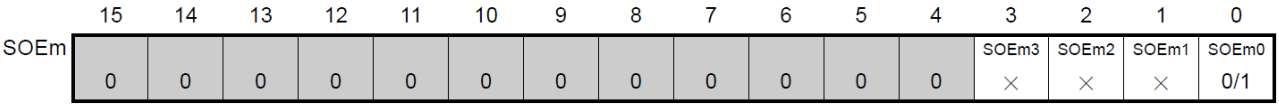
(2) During operation (SEmn=1) (Lower 8 bits: SDRmnL)



(d) serial output register m (SOm)…. Set only the bits of the target channel



(e) serial output enable register m (SOEm)…. set the bit of the target channel to 1.



Notice: The SDRmn register must be set for transmit data before the master device begins to output the clock;

Remark: m: unit number (m=0, 1);
n: channel number (n=0, 1);
p: SSPI number (p=00, 01, 10, 11);
□: Fixed set in SSPI master receive mode; ■ : Cannot be set (initial value is set);
×: This is a bit that cannot be used in this mode (sets the initial value if it is not used in other modes either);
0/1: Set “0” or “1” according to the user’s purpose.

Figure 12-65: Slave selection input function (SSPI00)
Example of register setting content when slave transmits and receives (2/2)

(f) serial channel start register m (SSm) ···. Only set bit of 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	0	0	SSm1 ×	SSm0 0/1

(g) Input switching control register (ISC) ..... This is the control of the SS00 pin of the SSPI00 slave channel (channel 0 of unit 0).

	7	6	5	4	3	2	1	0
ISC	SSIE00 0/1	0	0	0	0	0	ISC1 0/1	ISC0 0/1

0: SS00 pin input is invalid
1: SS00 pin input is valid

Notice: The SDRmn register must be set for transmit data before the master device begins to output the clock;

Remark: m: unit number (m=0, 1);

n: channel number (n=0, 1);

p: SSPI number (p=00, 01, 10, 11);

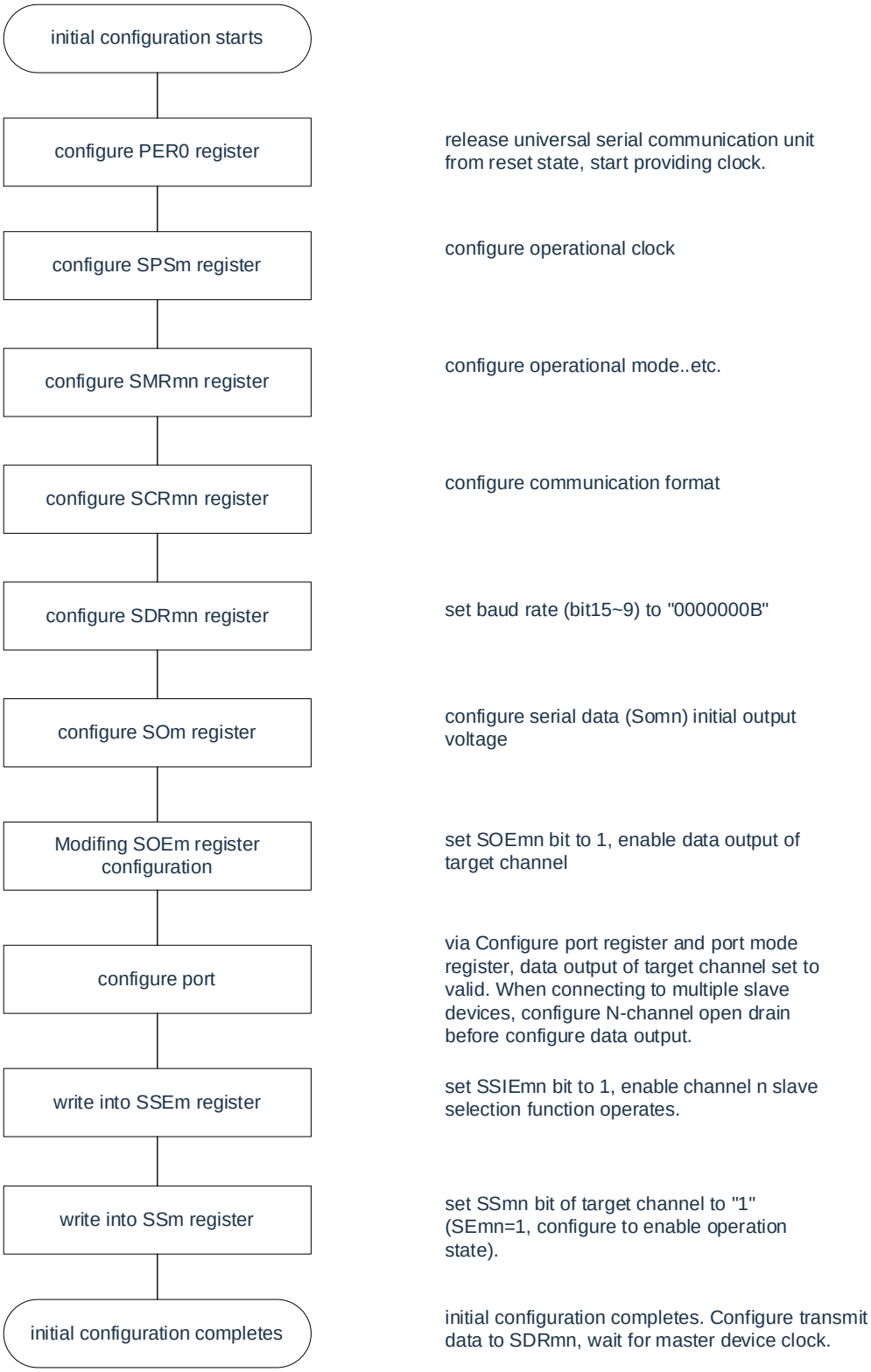
□: Fixed set in SSPI master receive mode; ■ : Cannot be set (initial value is set);

×: This is a bit that cannot be used in this mode (sets the initial value if it is not used in other modes either);

0/1: Set “0” or “1” according to the user’s purpose.

(2) Procedure

Figure 12-66: Initial setup steps for slave transmission and reception



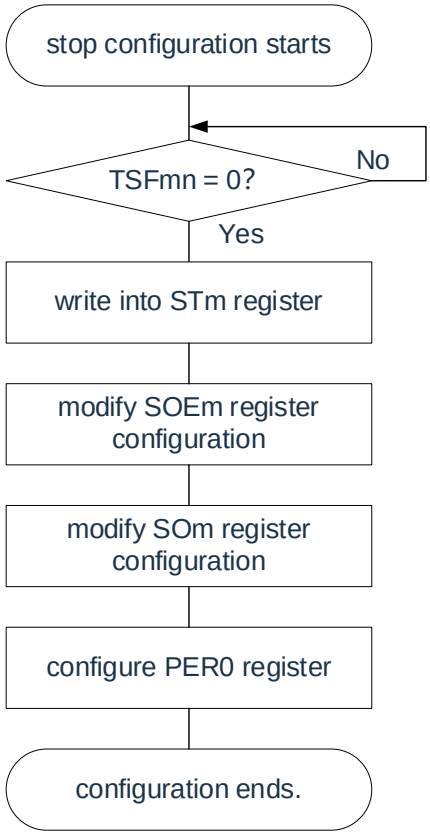
Notice: The SDRmn register must be set for transmit data before the master device begins to output the clock.

Remark: m: unit number (m=0);

n: channel number (n=0);

p: SSPI number (p=00).

Figure 12-67: Stop steps for slave transmission and reception



if there are ongoing data transmission, then wait till transmission completed. (if need urgent stop, then no need to wait).

set STmm bit of target channel to 1. (SEmn=0: set to operation stop state).

set SOEmn bit to 0, stop output of target channel

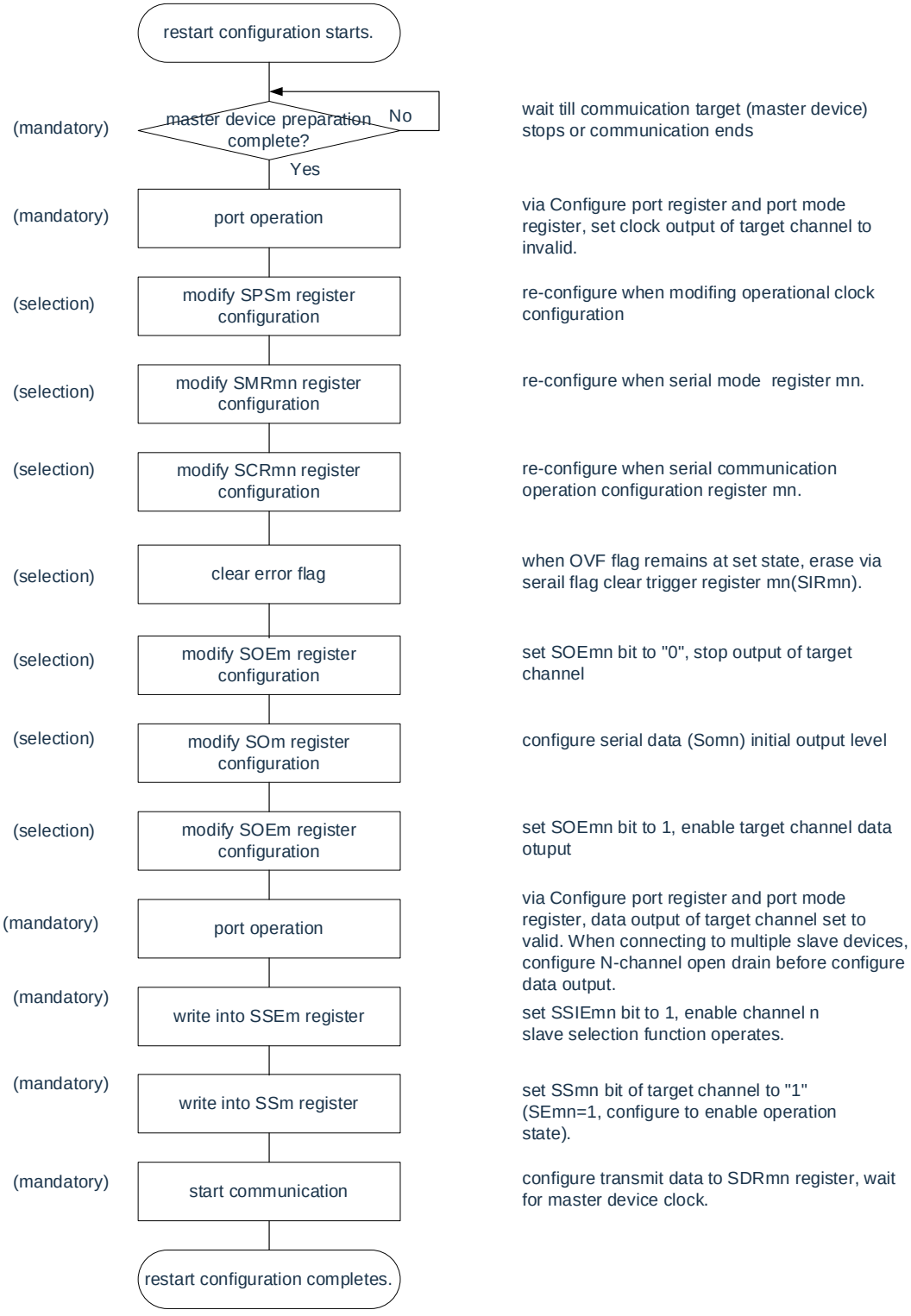
when emergency stop, based on needs, modify serial data (SOmn) level of the target channel.

stop clock of universal serial communication unit, set to reset state

finish stop configuration, enter into next processing.

Remark: m: unit number (m=0);
n: channel number (n=0);
p: SSPI number (p=00).

Figure 12-68: Restart steps for slave transmission and reception

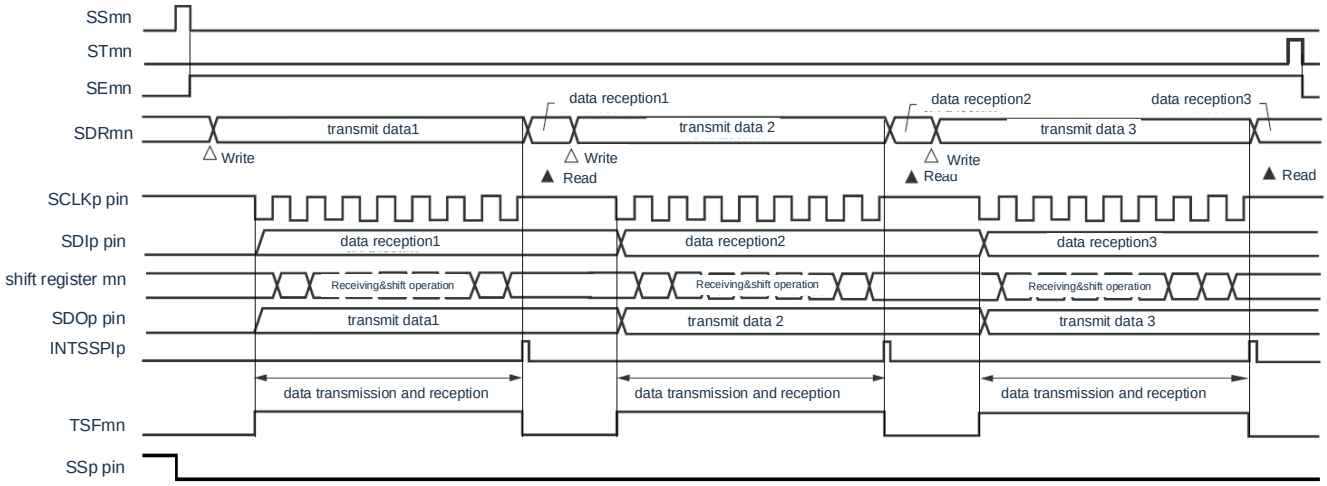


Notice:

1. The SDRmn register must be set for transmit data before the master device begins to output the clock.
2. If PER0 is rewritten in the abort setting to stop the clock supply, the initial setting must be made after the communication object (master) stops or the communication is finished, instead of restarting the setting.

(3) Process flow (single transmit and receive mode)

Figure 12-69: Timing diagram of slave transmit and receive (single transmit and receive mode) (type 1: DAPmn=0, CKPmn=0)

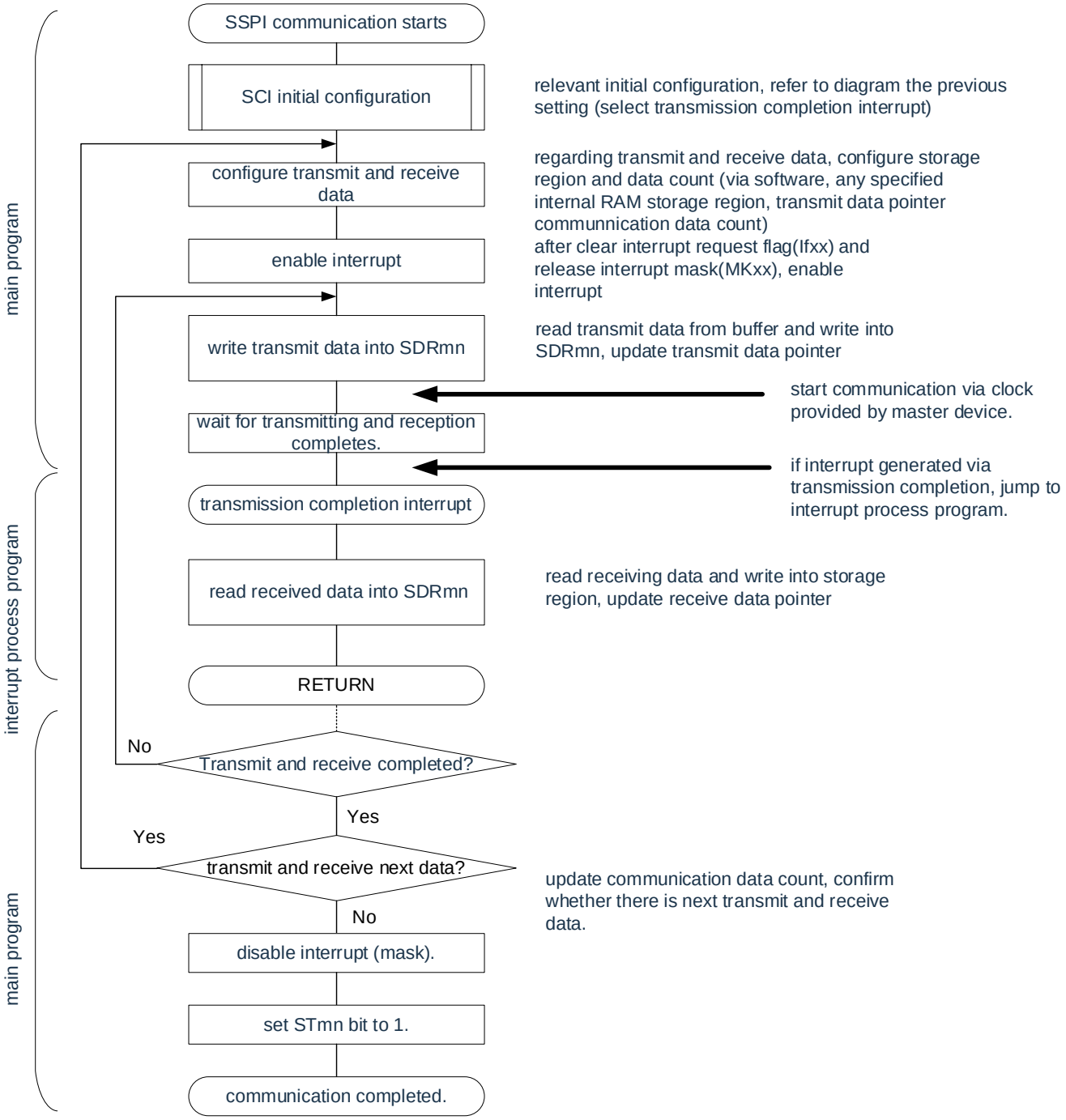


Remark: m: unit number (m=0);

n: channel number (n=0);

p: SSPI number (p=00).

Figure 12-70: Flowchart of slave transmit and receive (single transmit and receive mode)



Notice: The SDRmn register must be set for transmit data before the master device begins to output the clock.

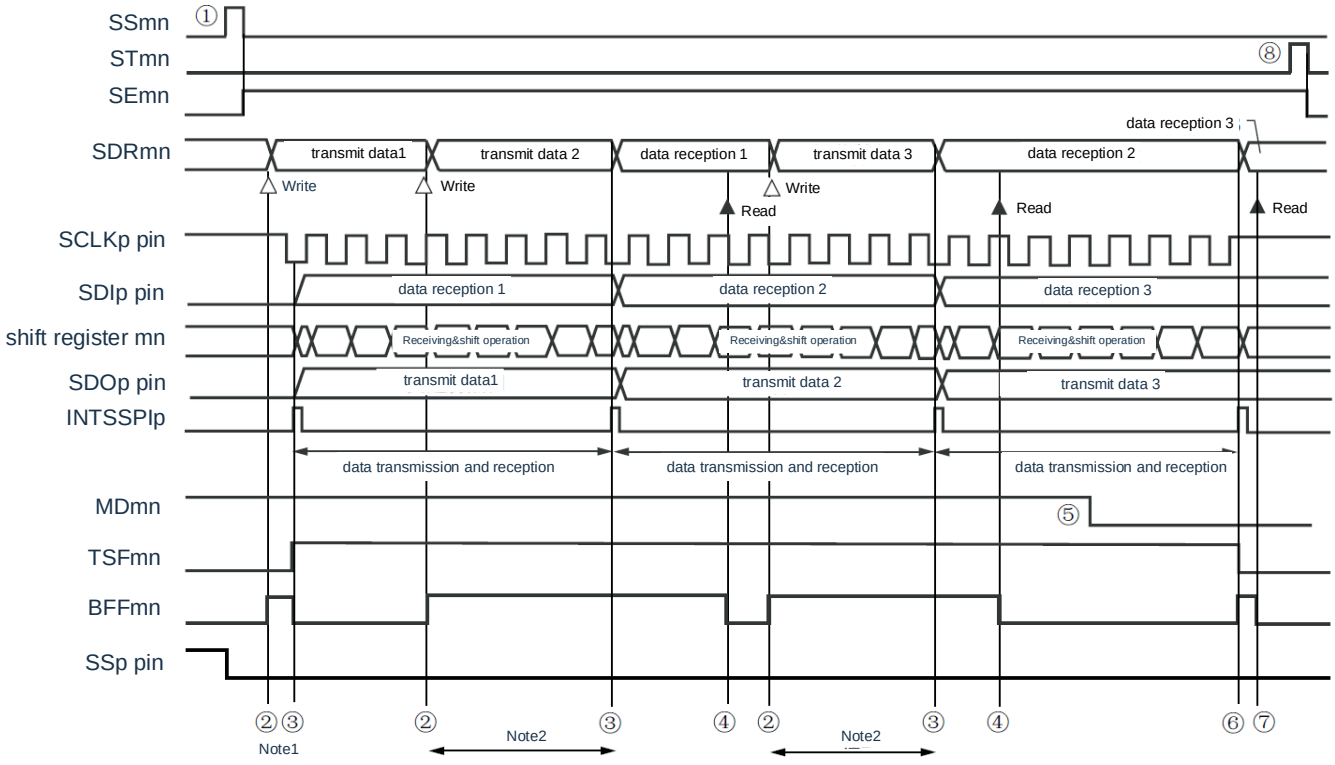
m: unit number (m=0);

n: channel number (n=0);

p: SSPI number (p=00).

(4) Process flow (continuous transmit and receive mode)

Figure 12-71: Timing diagram of slave transmit and receive (continuous transmit and receive mode) (type 1: DAPmn=0, CKPmn=0)



Note 1: If the transmit data is written to the SDRmn register during the time when the BFFmn bit of the serial status register mn (SSRmn) is “1” (when valid data is stored in the serial data register mn (SDRmn)), the transmit data is rewritten.

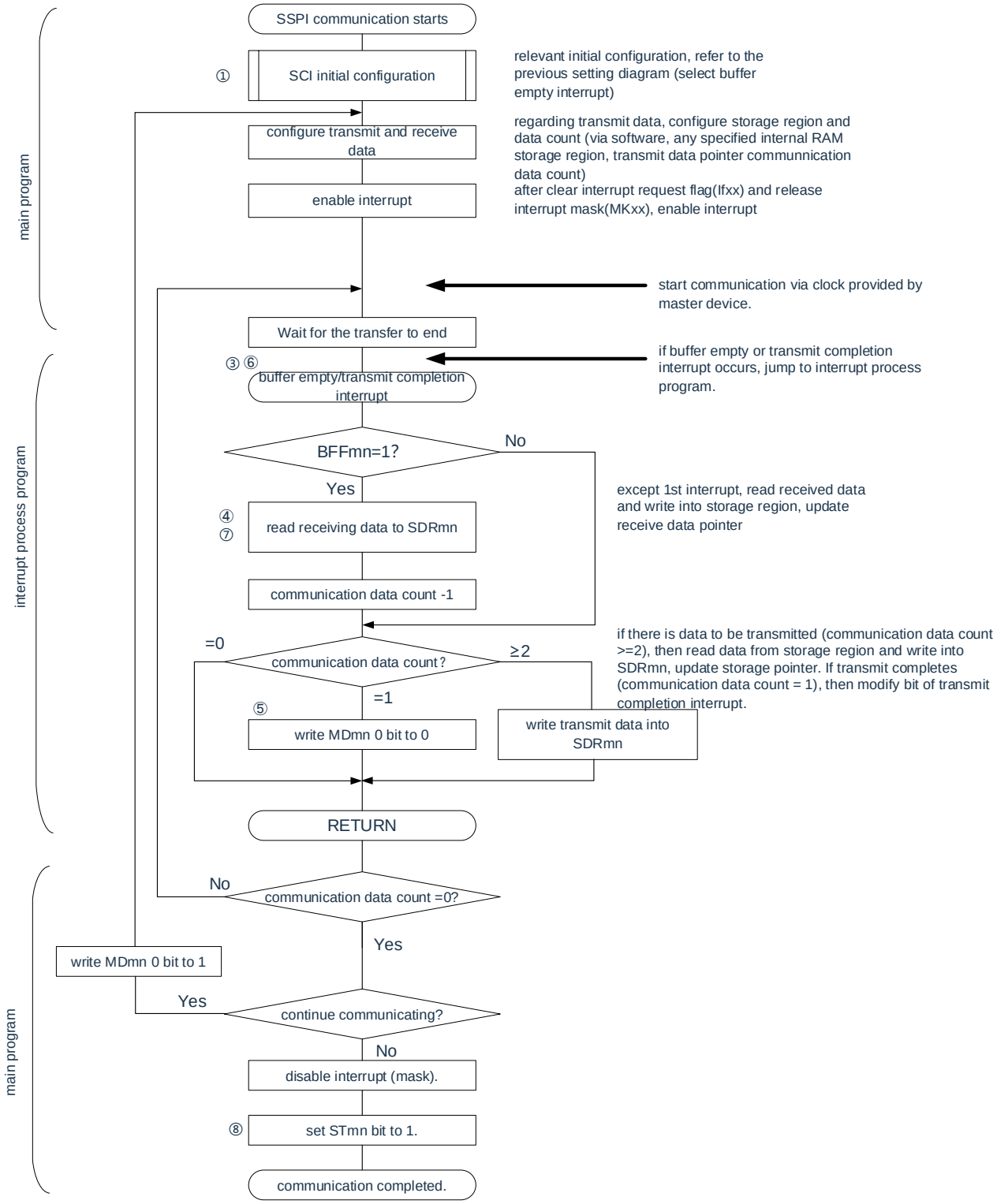
Note 2: If the SDRmn register is read during this time, the transmit data can be read. At this time, the transmission operation is not affected.

Notice: The MDmn0 bit of the serial mode register mn (SMRmn) can be rewritten even in operation. However, in order to catch the end-of-transmission interruption of the last sent data, it must be rewritten before the last bit of transmission begins.

Remark:

1. ① to ⑧ in the figure correspond to ① to ⑧ in "Figure 12-72 Flowchart of Slave Transmit and Receive (Continuous Transmit and Receive Mode)".
2. m: unit number (m=0) n: channel number (n=0) p: SSPI number (p=00).

Figure 12-72: Flowchart of slave transmit and receive (continuous transmit and receive mode)



Notice: The SDRmn register must be set for transmit data before the master device begins to output the clock.

Remark:

1. ① to ⑧ in the figure correspond to ① to ⑧ in "Figure 12-71 Timing Diagram of Slave Transmit and Receive (Continuous Transmit and Receive Mode)".
2. m: unit number (m=0) n: channel number (n=0) p: SSPI number (p=00).

12.6.4 Calculation of transmission clock frequency

The transmit clock frequency of the slave select input function (SSPI00) communication can be calculated using the following calculation equation.

(1) Slave

(Transmit clock frequency) = {Serial clock (SCLK) frequency provided by the master device} Note 1[Hz]

Note 1: The maximum allowable transmit clock frequency is F_{MCK}/6.

m: unit number (m=0) n: channel number (n=0) p: SSPI number (p=00).

Table 12-13: Selection of slave select input function operation clock

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}=32MHz in operation
0	X	X	X	X	0	0	0	0	F_{CLK}	32MHz
	X	X	X	X	0	0	0	1	F_{CLK}/2	16MHz
	X	X	X	X	0	0	1	0	$F_{CLK}/2^2$	8MHz
	X	X	X	X	0	0	1	1	$F_{CLK}/2^3$	4MHz
	X	X	X	X	0	1	0	0	$F_{CLK}/2^4$	2MHz
	X	X	X	X	0	1	0	1	$F_{CLK}/2^5$	1KHz
	X	X	X	X	0	1	1	0	$F_{CLK}/2^6$	500KHz
	X	X	X	X	0	1	1	1	$F_{CLK}/2^7$	250KHz
	X	X	X	X	1	0	0	0	$F_{CLK}/2^8$	125KHz
	X	X	X	X	1	0	0	1	$F_{CLK}/2^9$	62.5KHz
	X	X	X	X	1	0	1	0	$F_{CLK}/2^{10}$	31.25KHz
	X	X	X	X	1	0	1	1	$F_{CLK}/2^{11}$	15.63KHz
	X	X	X	X	1	1	0	0	$F_{CLK}/2^{12}$	7.81KHz
	X	X	X	X	1	1	0	1	$F_{CLK}/2^{13}$	3.91KHz
	X	X	X	X	1	1	1	0	$F_{CLK}/2^{14}$	1.95KHz
	X	X	X	X	1	1	1	1	$F_{CLK}/2^{15}$	977Hz

Note: To change the clock selected as F_{CLK} (change the value of the system clock control register (CKC)), the change must be made after stopping the operation of the general-purpose serial communication unit (SCI) (serial channel stop register m (STm) = 000FH).

X: Ignore

m: unit number (m=0) n: channel number (n=0).

12.6.5 Procedure for handling errors during clock-synchronous serial communication with the slave selection input function

The processing steps for errors that occur during clock synchronization serial communication with the slave select input function are shown in Table 12-34.

Table 12-34: Processing steps when an overflow error occurs

Software operation	Hardware status	Remark
Read Serial Data Registermn (SDRmn). →	The BFFmn bit of the SSRmn register is "0" and channel n is receivable.	This is to prevent an overflow error from occurring if the next reception ends during error handling.
Read serial status register mn (SSRmn).	-	Determines the type of error and reads the value for clearing the error flag.
Write "1" to the serial flag clear trigger register mn (SDIRmn). →	Clear the error flag.	Errors during read operations can only be cleared by writing the read value of the SSRmn register directly to the SDIRmn register.

Remark: m: unit number (m=0) n: channel number (n=0).

12.7 Operation of UART (UART0~UART1) communication

This is an asynchronous function using two lines: serial data transmission (TxD) and serial data reception (RxD) lines. By using these two communication lines, data is sent and received asynchronously (using the internal baud rate) with other communicating parties by data frame (consisting of start bits, data, parity bits, and stop bits). Full-duplex asynchronous UART communication can be achieved by using two channels dedicated for transmit (even channel 00/10) and receive dedicated (odd channel 01/11).

[Data transmission and reception]

(1) Data length of 7, 8, 9 or 16 bits Note

(2) MSB/LSB preferred option

(3) Level setting for transmit/receive data (Selects whether the level is reversed or not)

(4) Parity bit appending and parity check functions

(5) Stop bit appending, stop bit detection function

[Interrupt function]

(1) Transfer end interrupt, buffer empty interrupt

(2) Error interrupts caused by frame errors, parity errors, and overflow errors

[Error detection flag]

Framing error, parity error, or overflow error

UART0 uses channel 0 and channel 1 of SCI0.

UART1 uses channel 0 and channel 1 of SCI1.

Each channel arbitrarily selects one function to use, except for the selected function, other functions can not be run.

For example, when using UART0 for channel 0 and channel 1 of unit m, SSPI00 cannot be used.

UART has the following 2 kinds of communication operation:

(1) UART transmission (refer to 12.7.1)

(2) UART reception (refer to 12.7.2)

Notice: When used as aUART, the transmitter (even numbered channel) and receiver (odd numbered channel) can only be used for UART.

12.7.1 UART transmission

UART transmission is the operation of this product microcontroller to asynchronously send data to other devices.

An even number of the 2 channels used by UART is used for UART transmission.

Table 12-35: UART transmission

UART	UART0	UART1
Target channel	Channel 0 of SCI0	Channel 0 of SCI1
Pins used	TxD0	TxD1
Interrupt	INTST0	INTST1
	Selectable transmission end interrupt (single transmission mode) or buffer empty interrupt (continuous transmission mode).	
Error detection flag	None	
Transfer data length	Data length of 7 bits, 8 bits, 9 bits or 16 bits	
Transfer rate	Max.F_{MCK}/6[bps] (SDRmn[15:9]≥3), Min.F_{CLK}/(22^{11}128)[bps]Note	
Data phase	Non-reverse output (default: High). Reverse output (default value: low).	
Parity bit	Select from the following:	
	No parity bits.	
	Appending zero parity check.	
	Appending even parity check.	
	Appending odd parity check.	
Stop bit	Select from the following:	
	Appending 1 bit.	
	Appending 2 bits.	
Data direction	MSB or LSB first	

Notice: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheret).

Remark:

1. F_{MCK}: Operation clock frequency of target channel
2. F_{CLK}: System clock frequency
3. m: unit number (m=0, 1) n: channel number (n=0).

(1) Register setting

Figure 12-73: Register setting content when UART is transmitted by UART (UART0~UART1) (1/2)

(a) serial mode register mn (SMRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn	CCSmn	0	0	0	0	0	0	0	0	1	0	0	MDmn2	MDmn1	MDmn0
	0/1	0												0	1	0/1

channel n operational clock (F_{MCK})
0: SPSm register configured pre-scaler output clock CKm0
1: SPSm register configured pre-scaler output clock CKm1

channel n interrupt source
0: Transmit completion interrupt
1: Buffer empty interrupt

(b) serial communication operation configuration register mn (SCRmn)

	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	DLSmn3	DLSmn2	DLSmn1	DLSmn0
	1	0	0	0	0	0	0/1	0/1	0/1	0	0/1	0/1	0/1	0/1	0/1	0/1

parity check bit configuration
00B: no parity check
01B: appending zero parity
10B: appending even parity
11B: appending odd parity

data transmit sequence selection
0: perform MSB first input/output
1: perform LSB first input/output

stop bit configuration
01B: appending 1 bit
10B: appending 2 bits

(c) Serial data register mn (SDRmn)

(1) When operation stops（SEmn=0）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	baud rate configuration							0	0	0	0	0	0	0	0	0

(2) During operation（SEmn=1）（lower 8 bits：SDRmnL）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	transmit data configuration															

SDRmnL

(d) serial output voltage register m (SOLm) …. Only configure bit of target channel.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOLm	0	0	0	0	0	0	0	0	0	0	0	0	0	SOLm2	0	SOLm0
														0/1		0/1

0: non-reverse (normal) transmit
1: reverse transmit

Remark: m: unit number (m=0, 1) n: channel number (n=0) q: UART number (q=0, 1)

□: Fixed set in SSPI master receive mode; ■ : Cannot be set (initial value is set);

×: This is a bit that cannot be used in this mode (sets the initial value if it is not used in other modes either).

0/1: Set “0” or “1” according to the user’s purpose.

Figure 12-73: Register setting content when UART is transmitted by UART (UART0~UART1) (2/2)

(e) serial output register m (SOm)…. Only configure bit of target channel

SOm	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	CKOm1 ×	CKOm0 ×	0	0	0	0	0	0	SOm1 ×	SOm0 0/1 注

0: serial data output value as "0"
1: serial data output value as "1"

(f) serial output enable register m (SOEm)…. Only set bit of 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	0	SOEm1 ×	SOEm0 0/1

(g) serial channel start register m (SSm) …. Only set bit of 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	0	0	SSm1 ×	SSm0 0/1

Notice: Before starting transmission, when the SOLmn bit of the corresponding channel is "0", it must be set to "1"; when the SOLmn bit of the corresponding channel is "1", it must be set to "0". When the SOLmn bit of the corresponding channel is "1", it must be set to "0". During communication, the value changes depending on the communication data.

Remark: m: unit number (m=0, 1) n: channel number (n=0) q: UART number (q=0, 1)

☐: Fixed set in SSPI master receive mode; ■ : Cannot be set (initial value is set);

×: This is a bit that cannot be used in this mode (sets the initial value if it is not used in other modes either).

0/1: Set “0” or “1” according to the user’s purpose.

(2) Procedure

Figure 12-74: Initial setup steps for UART transmission

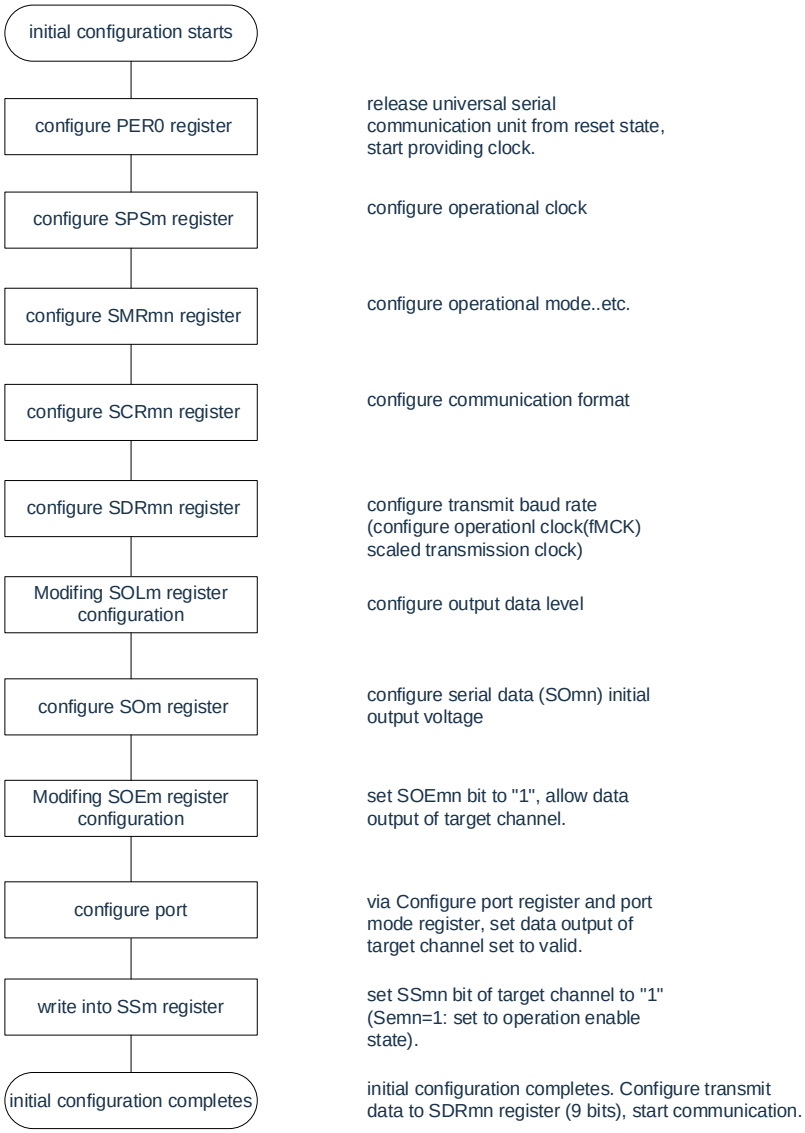


Figure 12-75: Stop steps for UART transmission

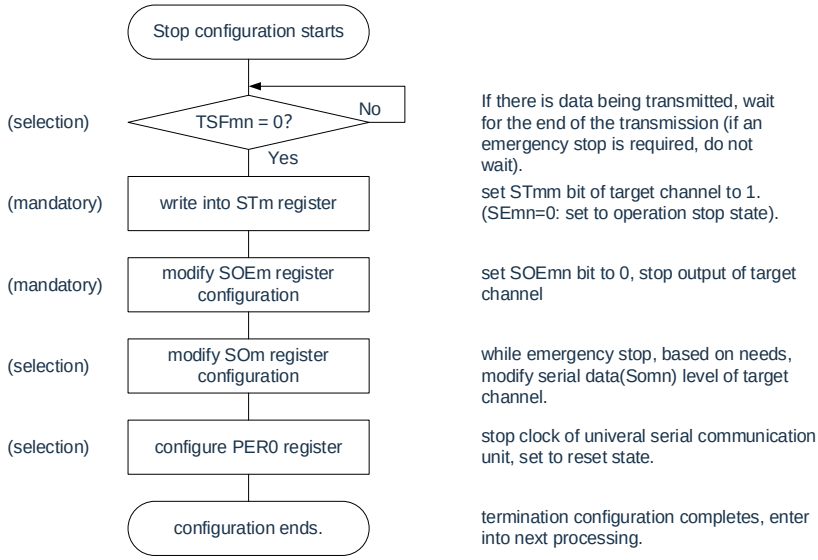
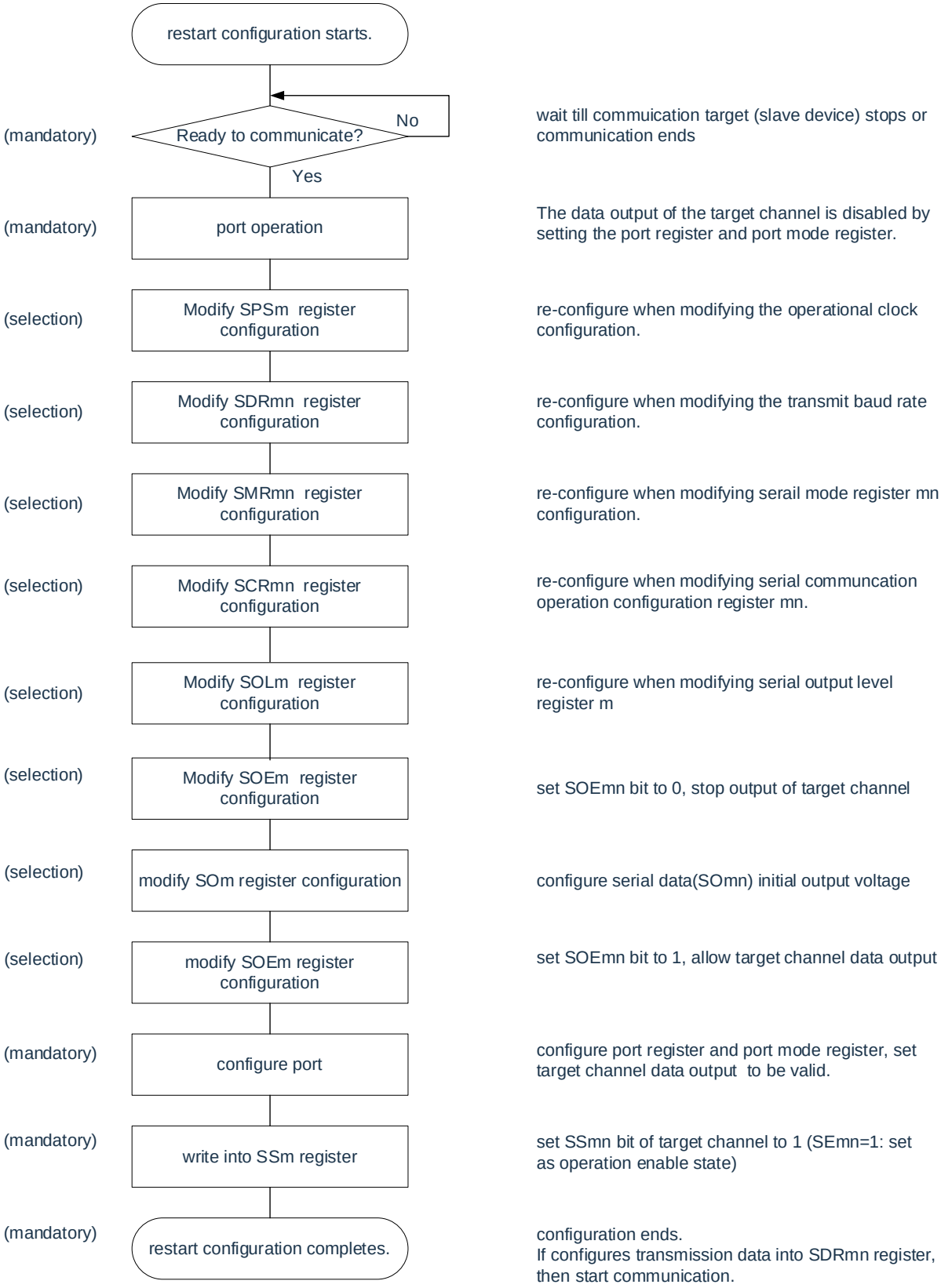


Figure 12-76: Restart steps for UART transmission



Notice: If PER0 is rewritten in the abort setting to stop the clock from being supplied, it is necessary to wait until the communication object stops or the communication is finished to make the initial setting instead of making a restart setting.

(3) Process flow (single transmit mode)

Figure 12-77: Timing diagram for UART transmission (single transmission mode)

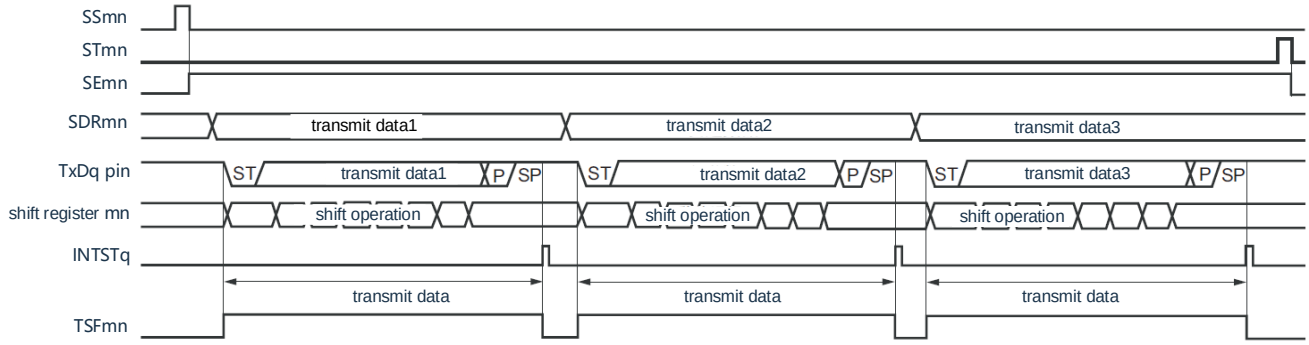
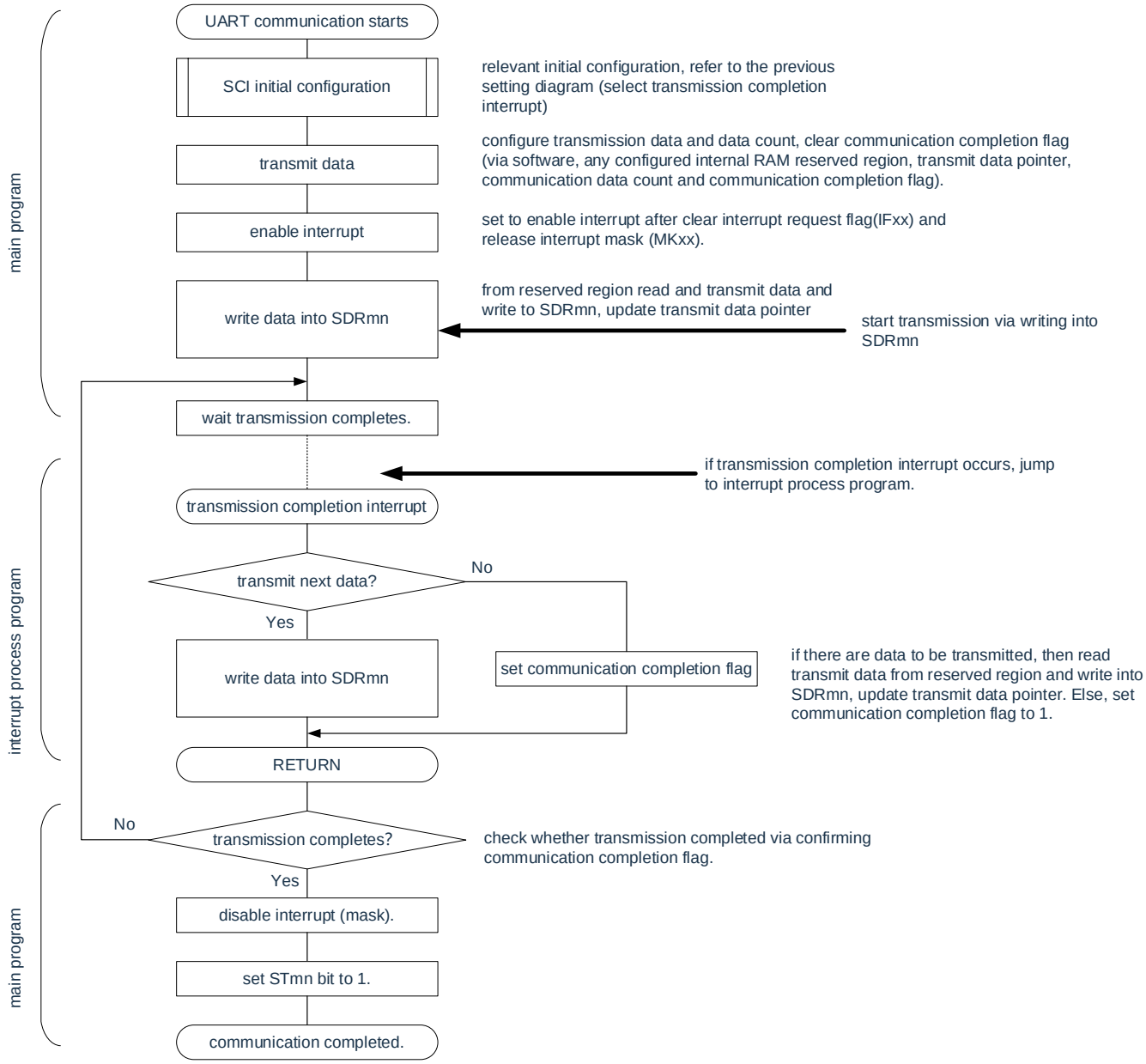


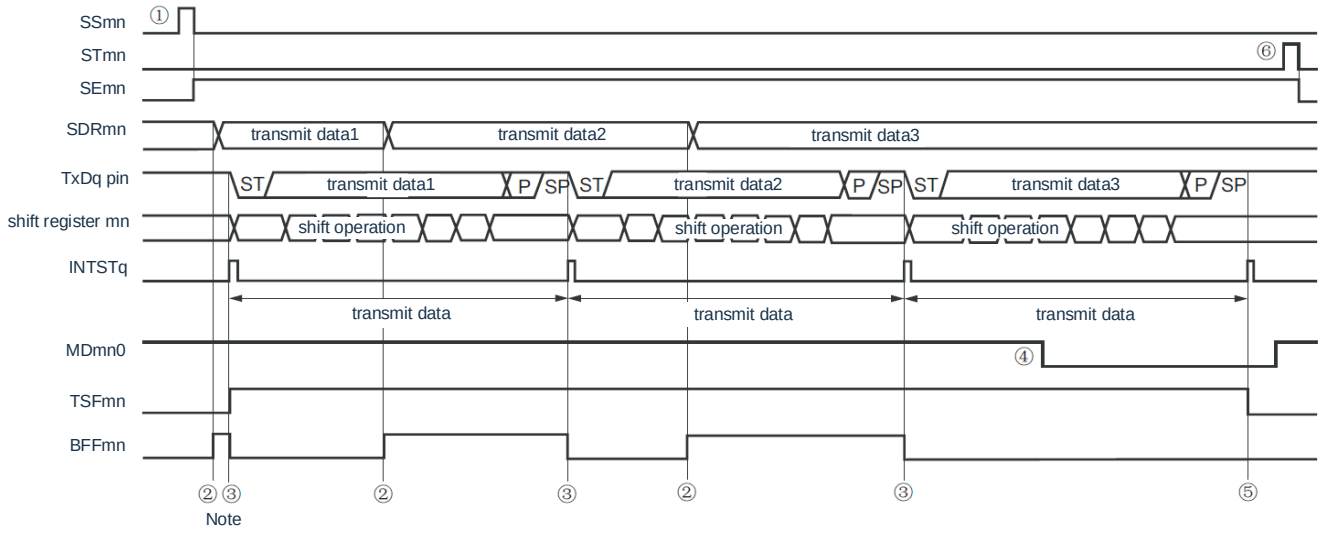
Figure 12-78: Flowchart for UART transmission (single transmission mode)



Remark: m: unit number (m=0, 1) n: channel number (n=0) q: UART number (q=0, 1).

(4) Process flow (continuous send mode)

Figure 12-79: Timing diagram for UART transmission (continuous transmission mode)

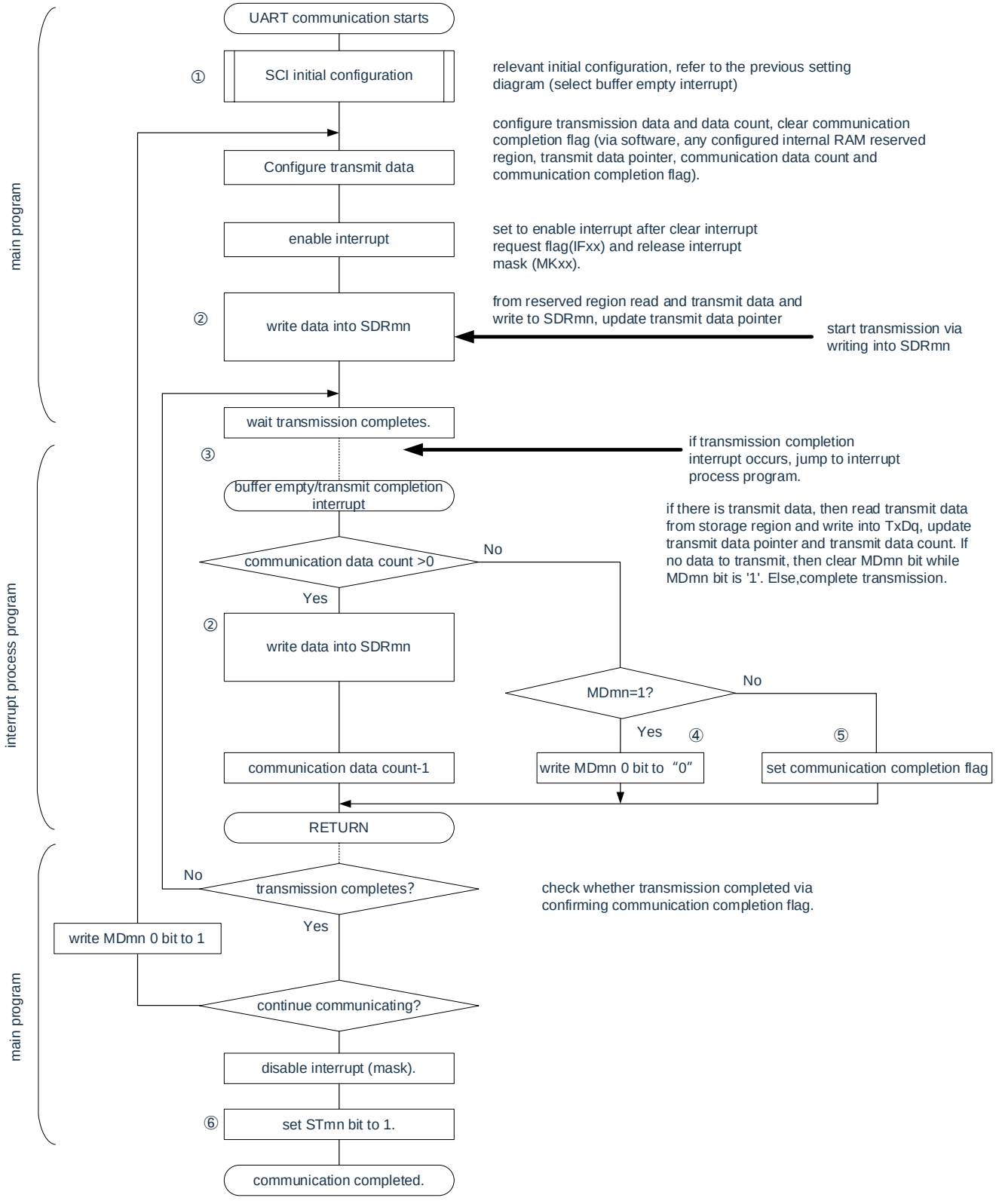


Note: If the transmit data is written to the SDRmn register during the time when the BFFmn bit of the serial status register mn (SSRmn) is “1” (when valid data is stored in the serial data register mn (SDRmn)), the transmit data is rewritten.

Notice: It is possible to rewrite the MDmn0 bit of the Serial Mode Register mn (SMRmn) even during operation. However, in order to catch the end-of-transmission interrupt for the last data sent, the rewrite must be done before the transmission of the last bit begins.

Remark: m: unit number (m=0, 1) n: channel number (n=0) q: UART number (q=0, 1).

Figure 12-80: Flowchart for UART transmission (continuous transmission ode)



Remark:① to ⑥ in the figure correspond to ① to ⑥ in "Figure 12-79 Timing Diagram of UART Transmission (Continuous Transmission Mode)".

12.7.2 UART reception

UART receive is the operation of this product microcontroller asynchronously receiving data from other devices.

The odd number of the 2 channels used by the UART is used for UART reception. However, the SMR registers for odd and even channels need to be set.

Table 12-36: UART reception

UART	UART0	UART1
Target channel	Channel 1 of SCI0	Channel 1 of SCI1
Pins used	RxD0	RxD1
Interrupt	INTSR0	INTSR1
	Limited to end-of-transmit interrupts (buffer empty interrupts are prohibited).	
Error interrupt	INTSRE0	INTSRE1
Error detection flag	Frame error detection flag (FEFmn) Parity error detection flag (PEFmn) Overflow error detection flag (OVFmn)	
Transfer data length	Data length of 7 bits, 8 bits, 9 bits or 16 bits	
Transfer rate	Max.F_{MCK}/6[bps] (SDRmn[15:9]≥2), Min.F_{CLK}/(22^{15}128)[bps]	
Data phase	Non-reverse output (default: high). Reverse output (default: low).	
Parity bit	Select from the following. No parity bit (no parity). Appending zero check (no parity). Even parity check Odd parity check	
Stop bit	Appending 1 bit.	
Data direction	MSB or LSB first	

Notice: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheret).

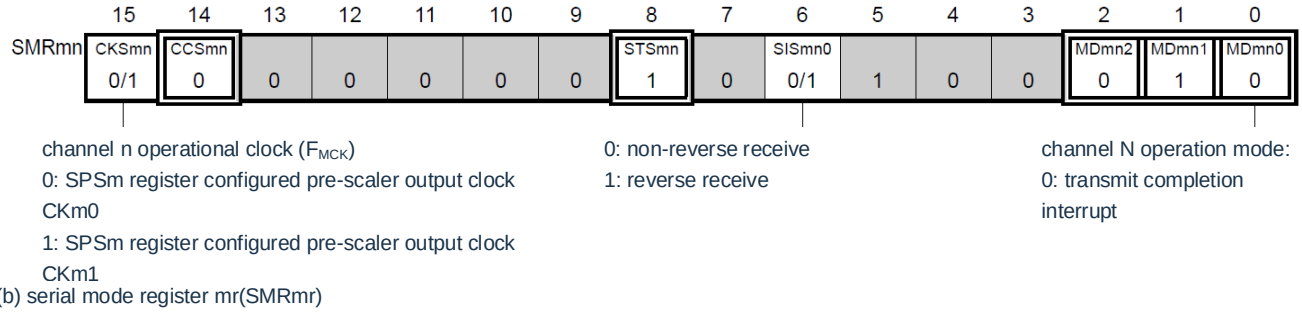
Remark:

1. F_{MCK}: Operation clock frequency of target channel
2. F_{CLK}: System clock frequency
3. m: unit number (m=0, 1) n: channel number (n=1).

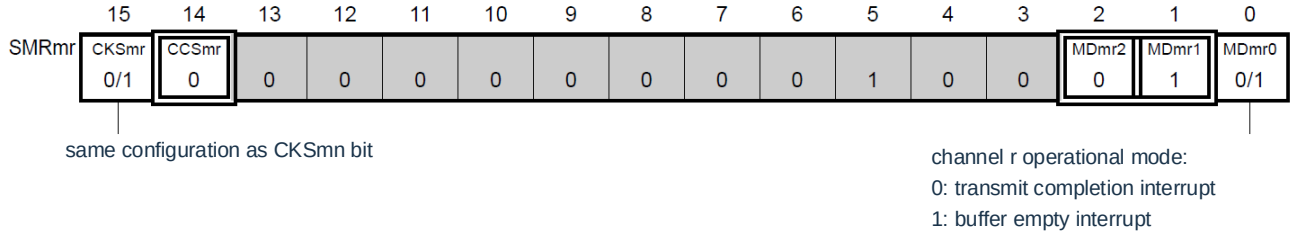
(1) Register setting

Figure 12-81: Example of register setting contents for UART reception of UART (UART0~UART1) (1/2)

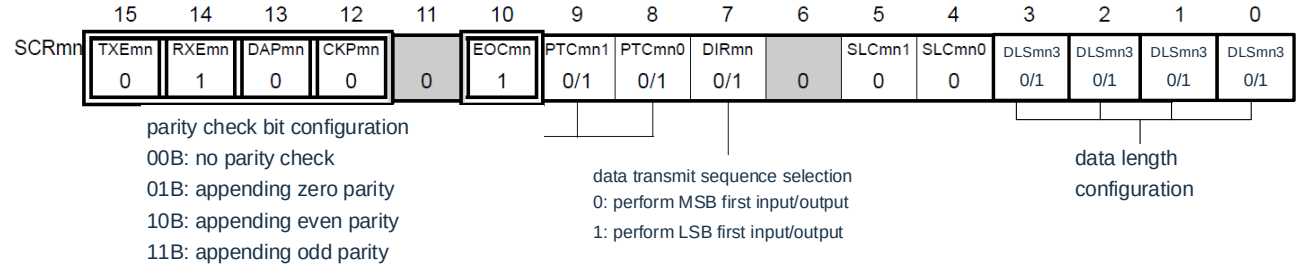
(a) serial mode register mn (SMRmn)



(b) serial mode register mr(SMRmr)

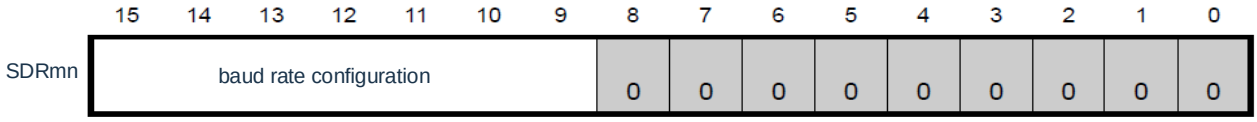


(c) serial communication operation configuration register mn(SCRmn)

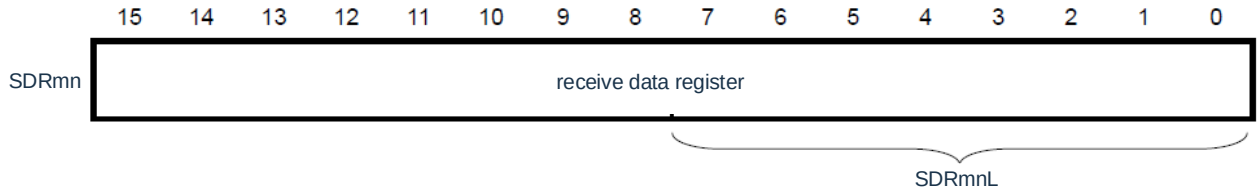


(d) serial data register mn (SDRmn)

(1) When operation stops（SEmn=0）



(2) During operation（SEmn=1）（lower 8 bits：SDRmnL）



Notice: For UART reception, the SMRmr register of channel r, which is paired with channel n, must also be set.

Remark: m: unit number (m=0, 1); n: channel number (n=1);

r: channel number (r=n-1); q: UART number (q=0~1).

□: Fixed set in SSPI master receive mode; ■ : Cannot be set (initial value is set);

×: This is a bit that cannot be used in this mode (sets the initial value if it is not used in other modes either).

0/1: Set “0” or “1” according to the user’s purpose.

Figure 12-81: Example of register setting contents for UART reception of UART (UART0~UART1) (2/2)

(e) serial output register m (SOm)…. 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	0	0	CKOm1	CKOm0	0	0	0	0	0	0	SOm1	SOm0
							×	×							×	×

(f) serial output enable register m (SOEm)…. 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	0	SOEm1	SOEm0
															×	×

(g) serial channel start register m (SSm) …. Only set bit of 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	0	0	SSm1	SSm0
															0/1	×

Notice: For UART reception, the SMRmr register of channel r, which is paired with channel n, must also be set.

Remark: m: unit number (m=0, 1); n: channel number (n=1);

r: channel number (r=n-1);q: UART number (q=0~1).

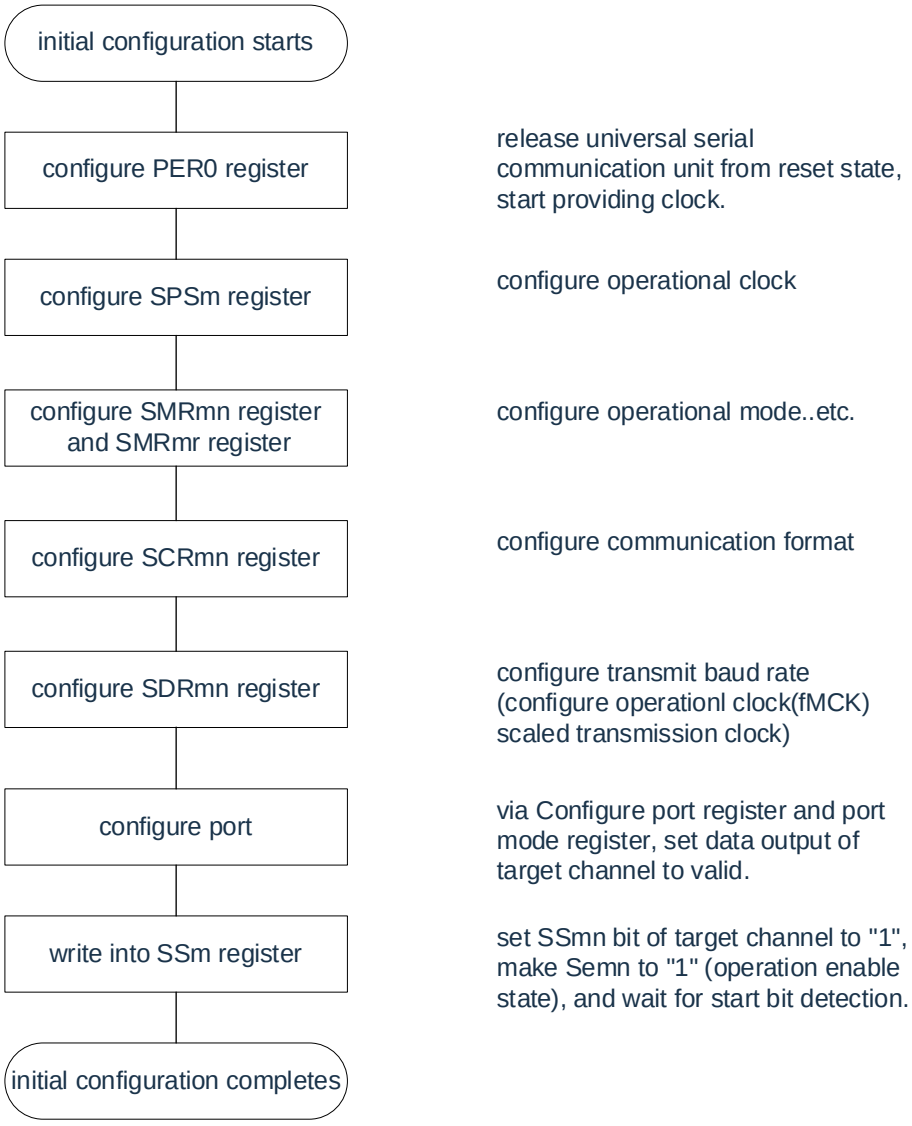
☐ : Fixed set in SSPI master receive mode;■ : Cannot be set (initial value is set);

×: This is a bit that cannot be used in this mode (sets the initial value if it is not used in other modes either).

0/1: Set “0” or “1” according to the user’s purpose.

(2) Procedure

Figure 12-82: Initial serup steps for UART reception



Notice: At least 4 F_{MCK} clocks must be spaced after setting the RXEmn bit of the SCRmn register to "1" and then set the SSmn bit to “1”.

Figure 12-83: Stop steps for UART reception

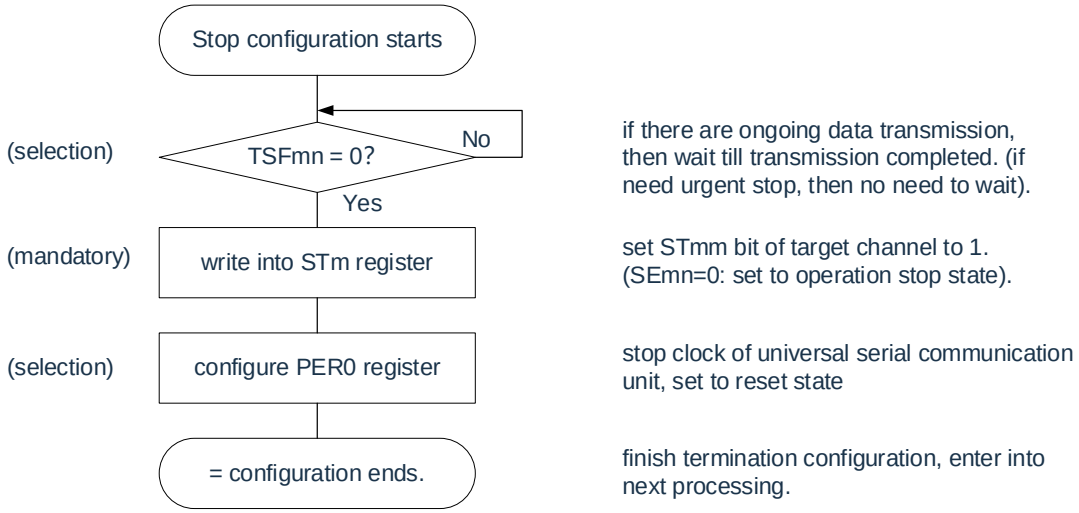
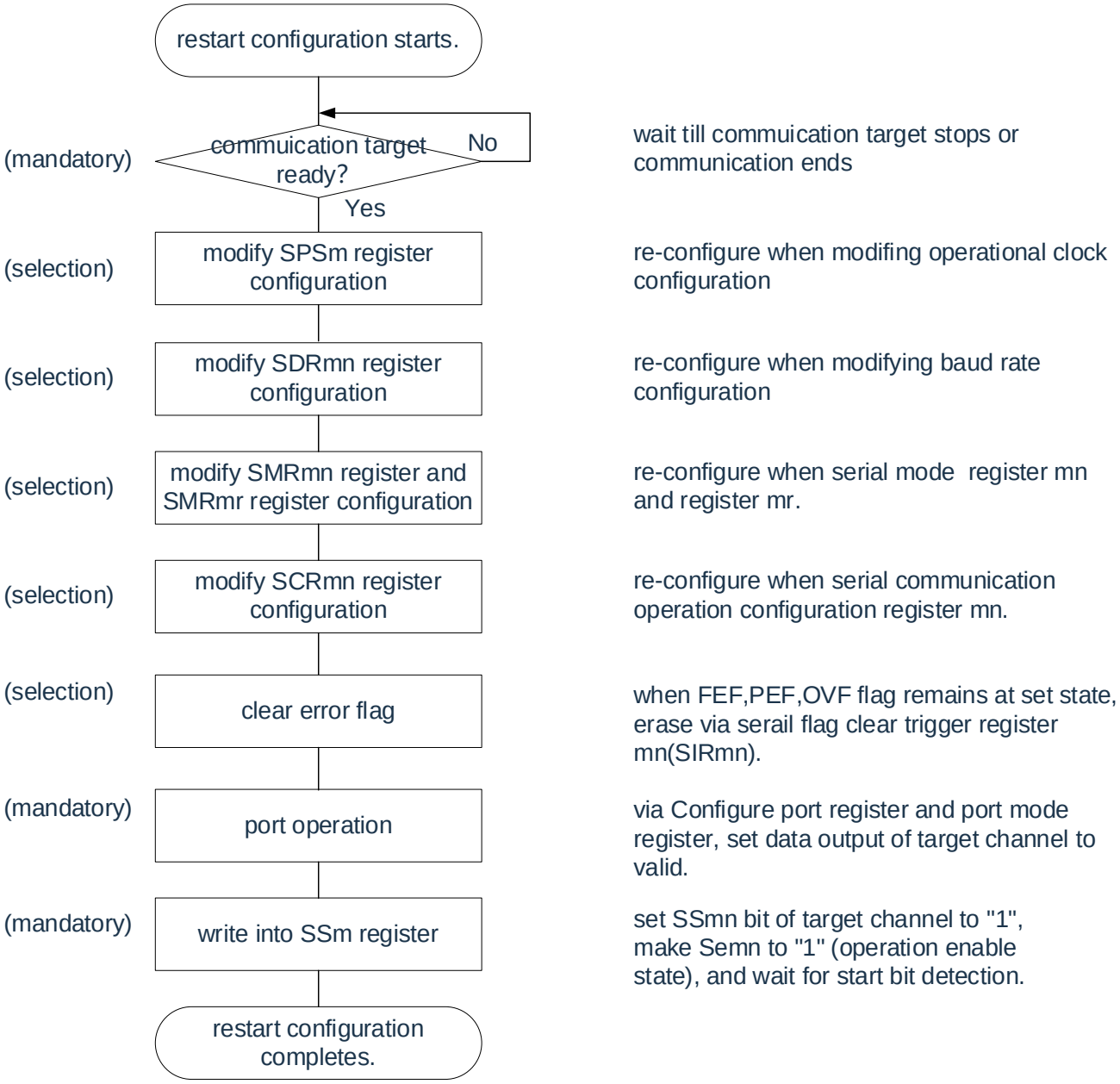


Figure 12-84: Restart steps for UART reception

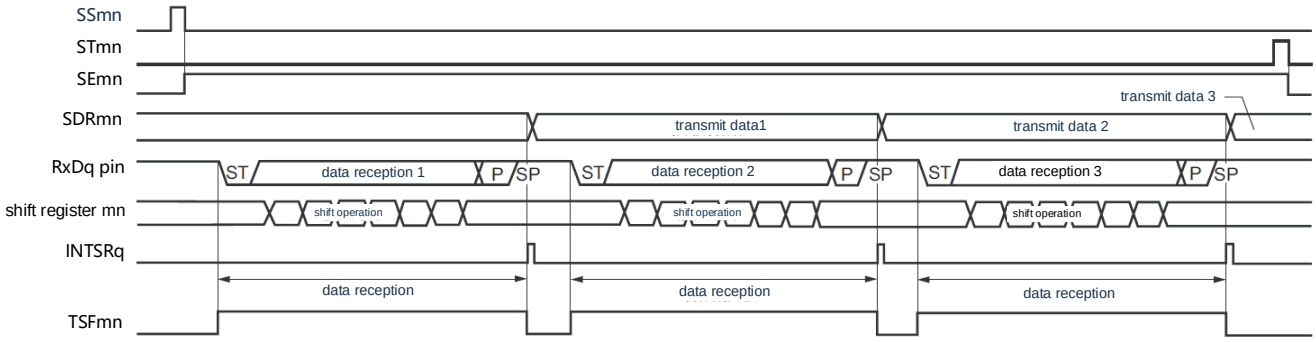


Notice:

1. At least 4 F_{MCK} clocks must elapse after setting the RXEmn bit of the SCRmn register "1" and then setting the SSmn bit "1".
2. If PER0 is rewritten in the abort setting to stop the clock from being supplied, it is necessary to wait until the communication object stops or the communication is finished to make the initial setting instead of making a restart setting.

(3) Process flow

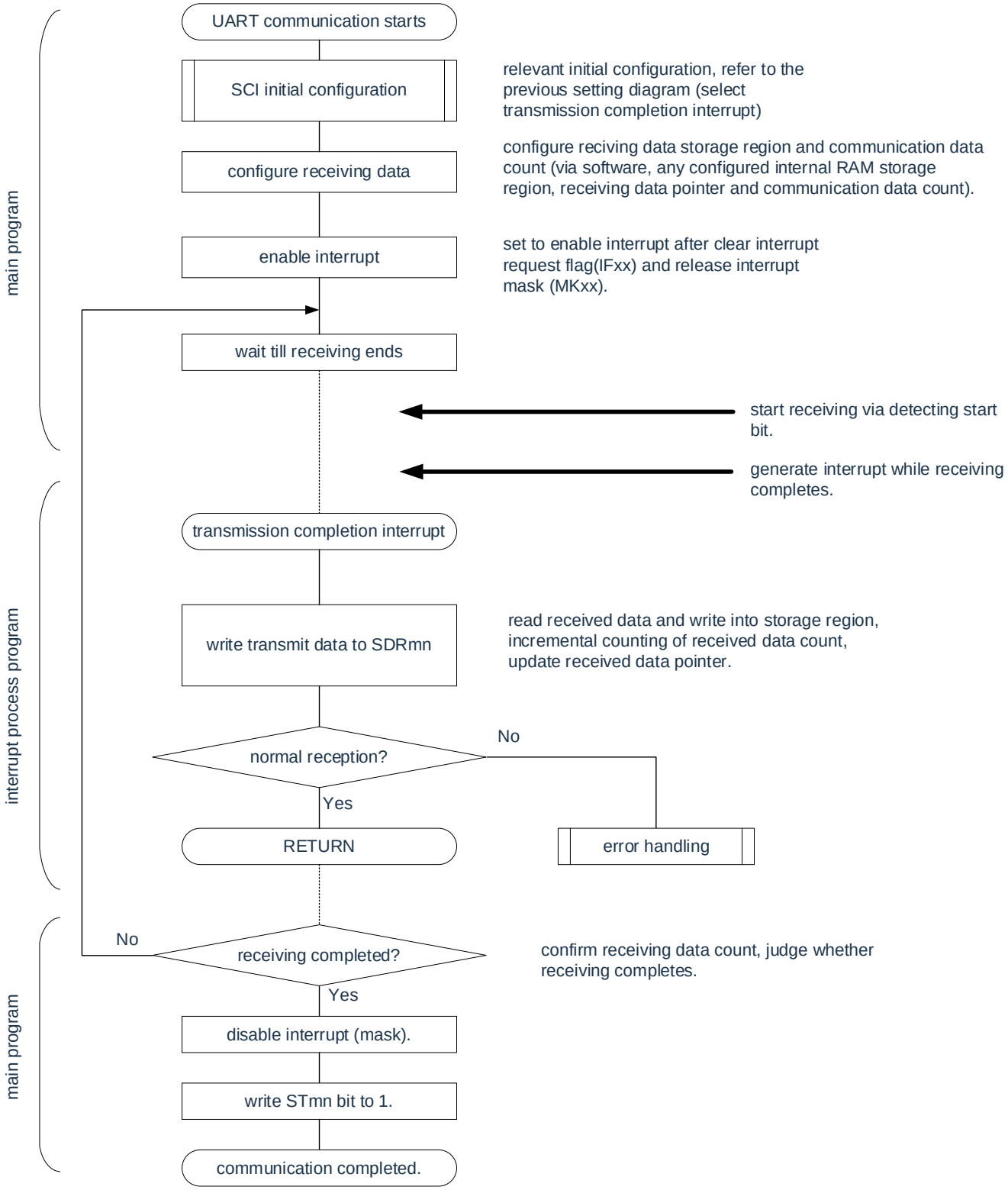
Figure 12-85: Timing diagram for UART reception



Remark: m: unit number (m=0, 1); n: channel number (n=1);

r: channel number (r=n-1); q: UART number (q=0, 1).

Figure 12-86: Flowchart of UART reception



12.7.3 Calculation of baud rate

(1) Equation for calculating baud rate

The baud rate for UART (UART0~UART1) communication can be calculated using the following equation.

(Baud rate) = {operating clock frequency (F_{MCK}) of the target channel}÷(SDRmn[15:9]+1)÷2[bps]

The operating clock (F_{MCK}) depends on bit15 (CKSmn bit) of the Serial Clock Select Register m (SPSm) and the Serial Mode Register mn (SMRmn).

Notice: It is prohibited to set SDRmn[15:9] of the serial data register mn (SDRmn) to "0000000B" and "0000001B".

Remark: It is 2~127 because the value of SDRmn[15:9] is the value of bit15~9 of SDRmn register (0000010B~1111111B) when using UART.

m: unit number (m=0, 1) n: channel number (n=0, 1).

Table 12-37: Selection of UART operation clock

SMRmn register	SPSm register								Operation clock (F_{CLK}) Note	
CKSmn	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00		F_{CLK}=32MHz in operation
0	X	X	X	X	0	0	0	0	F_{CLK}	32MHz
	X	X	X	X	0	0	0	1	$F_{CLK}/2$	16MHz
	X	X	X	X	0	0	1	0	$F_{CLK}/2^2$	8MHz
	X	X	X	X	0	0	1	1	$F_{CLK}/2^3$	4MHz
	X	X	X	X	0	1	0	0	$F_{CLK}/2^4$	2MHz
	X	X	X	X	0	1	0	1	$F_{CLK}/2^5$	1MHz
	X	X	X	X	0	1	1	0	$F_{CLK}/2^6$	500KHz
	X	X	X	X	0	1	1	1	$F_{CLK}/2^7$	250KHz
	X	X	X	X	1	0	0	0	$F_{CLK}/2^8$	125KHz
	X	X	X	X	1	0	0	1	$F_{CLK}/2^9$	62.5KHz
	X	X	X	X	1	0	1	0	$F_{CLK}/2^{10}$	31.25KHz
	X	X	X	X	1	0	1	1	$F_{CLK}/2^{11}$	15.63KHz
	X	X	X	X	1	1	0	0	$F_{CLK}/2^{12}$	7.81KHz
	X	X	X	X	1	1	0	1	$F_{CLK}/2^{13}$	3.91KHz
	X	X	X	X	1	1	1	0	$F_{CLK}/2^{14}$	1.95KHz
	X	X	X	X	1	1	1	1	$F_{CLK}/2^{15}$	977Hz
1	0	0	0	0	X	X	X	X	F_{CLK}	32MHz
	0	0	0	1	X	X	X	X	$F_{CLK}/2$	16MHz
	0	0	1	0	X	X	X	X	$F_{CLK}/2^2$	8MHz
	0	0	1	1	X	X	X	X	$F_{CLK}/2^3$	4MHz
	0	1	0	0	X	X	X	X	$F_{CLK}/2^4$	2MHz
	0	1	0	1	X	X	X	X	$F_{CLK}/2^5$	1MHz
	0	1	1	0	X	X	X	X	$F_{CLK}/2^6$	500KHz
	0	1	1	1	X	X	X	X	$F_{CLK}/2^7$	250KHz
	1	0	0	0	X	X	X	X	$F_{CLK}/2^8$	125KHz
	1	0	0	1	X	X	X	X	$F_{CLK}/2^9$	62.5KHz
	1	0	1	0	X	X	X	X	$F_{CLK}/2^{10}$	31.25KHz
	1	0	1	1	X	X	X	X	$F_{CLK}/2^{11}$	15.63KHz
	1	1	0	0	X	X	X	X	$F_{CLK}/2^{12}$	7.81KHz
	1	1	0	1	X	X	X	X	$F_{CLK}/2^{13}$	3.91KHz
	1	1	1	0	X	X	X	X	$F_{CLK}/2^{14}$	1.95KHz
	1	1	1	1	X	X	X	X	$F_{CLK}/2^{15}$	977Hz

Note: When you change the clock selected as F_{CLK} (change the value of the system clock control register (CKC)), you must stop the operation of the general-purpos serial communication unit (SCI) (serial channel stop register m (STm)=000FH) after making changes.

X: Ignore

m: unit number (m=0, 1) n: channel number (n=0, 1).

(2) Baud rate error when transmitting

The baud rate error during UART (UART0~UART1) communication transmission can be calculated using the following calculation equation, and the baud rate of the sender must be set within the allowable baud rate of the receiver.

(Baud rate error) = (calculated value of baud rate)÷(value of the target baud rate)×100-100[%]

An example of the UART baud rate at F_{CLK}=32MHz is shown below.

UART baud rate (Target baud rate)	F_{CLK}=32MHz			
	Operation clock (F_{MCK})	SDRmn[15:9]	Calculated value of the baud rate	Error from the target baud rate
300bps	$F_{CLK}/2^9$	103	300.48bps	+0.16%
600bps	$F_{CLK}/2^8$	103	600.96bps	+0.16%
1200bps	$F_{CLK}/2^7$	103	1201.92bps	+0.16%
2400bps	$F_{CLK}/2^6$	103	2403.85bps	+0.16%
4800bps	$F_{CLK}/2^5$	103	4807.69bps	+0.16%
9600bps	$F_{CLK}/2^4$	103	9615.38bps	+0.16%
19200bps	$F_{CLK}/2^3$	103	19230.8bps	+0.16%
31250bps	$F_{CLK}/2^3$	63	31250.0bps	±0.0%
38400bps	$F_{CLK}/2^2$	103	38461.5bps	+0.16%
76800bps	$F_{CLK}/2$	103	76923.1bps	+0.16%
153600bps	F_{CLK}	103	153846bps	+0.16%
312500bps	F_{CLK}	50	313725bps	±0.39%

Remark: m: unit number (m=0, 1) n: channel number (n=0).

(3) Allowable range of baud rate when receiving

The baud rate tolerance range for UART (UART0~UART1) communication reception can be calculated using the following equation, and the transmitter baud rate must be set within the the receiver tolerance range.

$$\text{(Maximum baud rate that can be received)} = \frac{2\times k\times Nfr}{2\times k\times Nfr-k+2} \times \text{Brate}$$

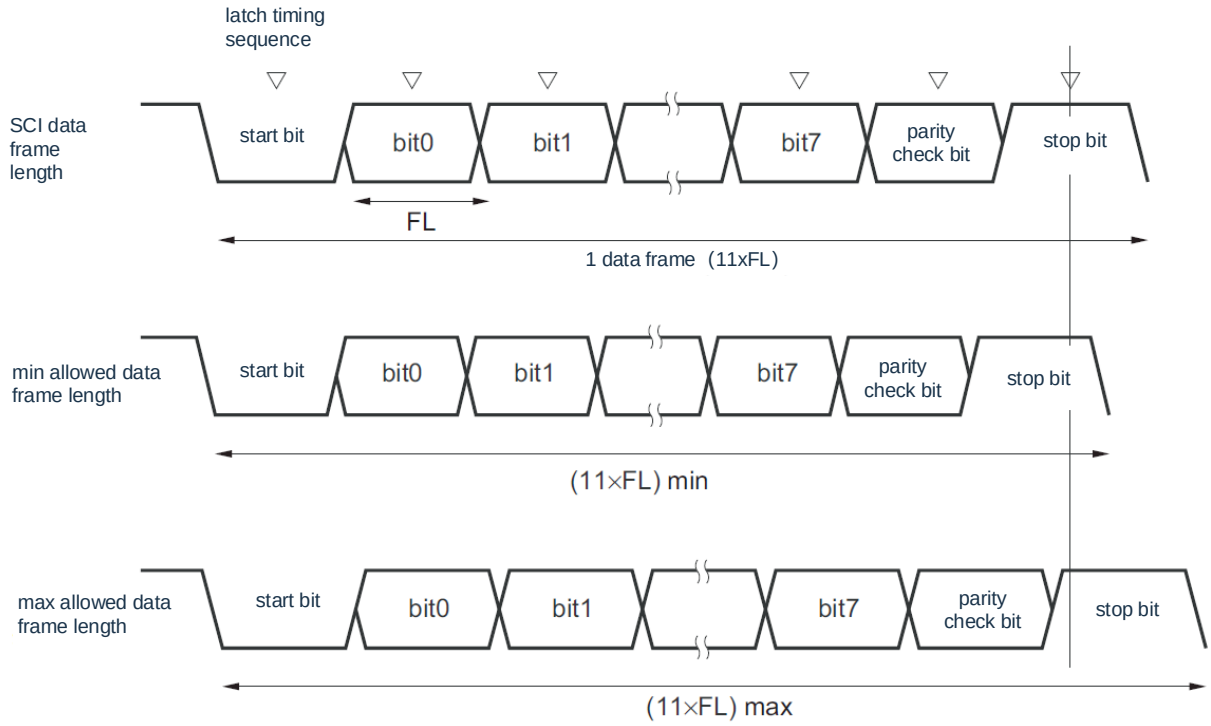
$$\text{(Maximum baud rate that can be received)} = \frac{2\times k\times (Nfr-1)}{2\times k\times Nfr-k-2} \times \text{Brate}$$

Brate: Calculated value of the receiver's baud rate (refer to “12.7.4 (1) Calculation of baud rate”)

K: SDRmn[15:9]+1

Nfr: Frame length of 1 data [bit] = (start bit) + (data length) + (parity bit) + (stop bit)

Figure 12-8: Baud rate tolerance range at reception (1 data frame length = 11 bits)



As shown in Figure12-87, after the start bit is detected, the latch timing of the received data depends on the division ratio set by bit15 to 9 of the serial data register mn (SDRmn). If the last data (stop bit) catches up with this latch timing, it can be received normally.

Remark: m: unit number (m=0, 1); n: channel number (n=1).

12.7.4 Handling steps when an error occurs during UART (UART0~UART1) communication

The processing steps for errors that occur during UART (UART0~UART1) communication are shown in Figure12-38 and Figure 12-39.

Table 12-38: Processing steps when a parity error or overflow error occurs

Software operation	Hardware status	Remark
Read Serial Data Registermn (SDRmn). →	The BFFmn bit of the SSRmn register is "0" and channel n is receivable.	This is to prevent an overflow error from occurring if the next reception ends during error handling.
Read Serial Status Registermn (SSRmn).		Determines the type of error and reads the value for clearing the error flag.
Write "1" to the serial flag clear trigger register mn (SDIRmn). →	Clear the error flag.	Errors during read operations can only be cleared by writing the read value of the SSRmn register directly to the SDIRmn register.

Table 12-39: Processing steps when a frame error occurs

Software operation	Hardware status	Remark
Read Serial Data Registermn (SDRmn). →	The BFFmn bit of the SSRmn register is "0" and channel n is receivable.	This is to prevent an overflow error from occurring if the next reception ends during error handling.
Read Serial Status Registermn (SSRmn).		Determines the type of error and reads the value for clearing the error flag.
Write serial flag clear trigger register mn (SIRmn). →	Clear the error flag.	Errors during read operations can only be cleared by writing the read value of the SSRmn register directly to the SDIRmn register.
Set the STmn bit of the serial channel stop register m (STm) to "1". →	The SEmn bit of the Serial Channel Enable Status Register m (SEm) is "0" and channel n is in an operation stop state.	
Synchronization with the communicating party.		Since the start bit is offset, it can be assumed that a framing error has occurred. Therefore, it is necessary to re-synchronize with the communicating party and restart communication.
Set the SSmn bit of serial channel start register m (SSm) to "1". →	The SEmn bit of the serial channel enable status register m (SEm) is "1" and channel n is operational.	

Remark: m: unit number (m=0, 1) n: channel number (n=0, 1).

12.8 Operation of LIN communication

12.8.1 LIN transmission

UART0 supports LIN communication in UART transmission.

LIN sends channel 0 of unit 0.

Table 12-40: LIN transmission

UART	UART0	UART1
LIN communication support	Yes	No
Target channel	Channel 0 of SCI0	—
Pins used	TxD0	—
Interrupt	INTST0	—
	Selectable transmission end interrupt (single transmission mode) or buffer empty interrupt (continuous transmission mode).	
Error detection flag	None	
Transfer data length	8 bits	
Transfer rate Note	Max.F_{MCK}/6[bps] (SDR00[15:9]≥2), Min.F_{CLK}/($2^2 \cdot 2^{15} \cdot 128$)[bps]	
Data phase	Non-reverse output (default: high). Reverse output (default: low).	
Parity check bit	No parity bits.	
Stop bit	Appending 1 bit.	
Data direction	LSB first	

Notice: It must be used within the range of peripheral functional characteristics (refer to the datasheet) that satisfy this condition as well as the electrical characteristics, and 2.4/9.6/19.2 kbps is often used in LIN communications.

Remark:

1. F_{MCK}: Operation clock frequency of target channel
2. F_{CLK}: System clock frequency

LIN is short for Local Interconnect Network, and the communication rate is 1~20Kbps. LIN communication is single-master communication, and a master device can connect up to 15 slave devices.

The LIN slave is used for the control of switches, transmission devices, sensors, etc., which are connected to the main control device through the LIN.

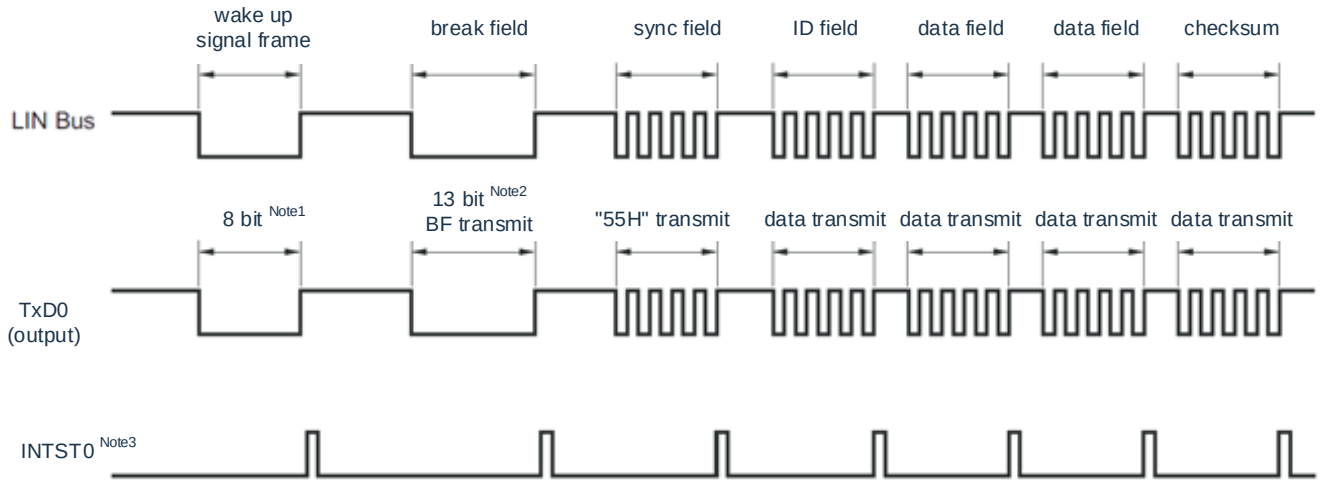
The LIN controls the network which connects CAN (Controller Area Network) and so on.

The LIN bus is a single-line bus, which connects nodes through ISDO9141-compliant transceivers.

According to the LIN protocol, the master device sends a frame with additional baud rate information, and the slave device receives the frame and corrects the baud rate error with the master device. Therefore, if the baud rate error of the slave device is not greater than ±15%, communication can be performed.

A summary of the LIN's send operations is shown in Figure 12-88.

Figure 12-88: Operation of LIN transmission



Note 1: In order to meet the requirements of wake-up signal, set baud rate and transmit "80H" data to correspond.

Note 2: The break field is specified as a 13-bit wide low-level output, so the baud rate used for the main transmission is N[bps]:

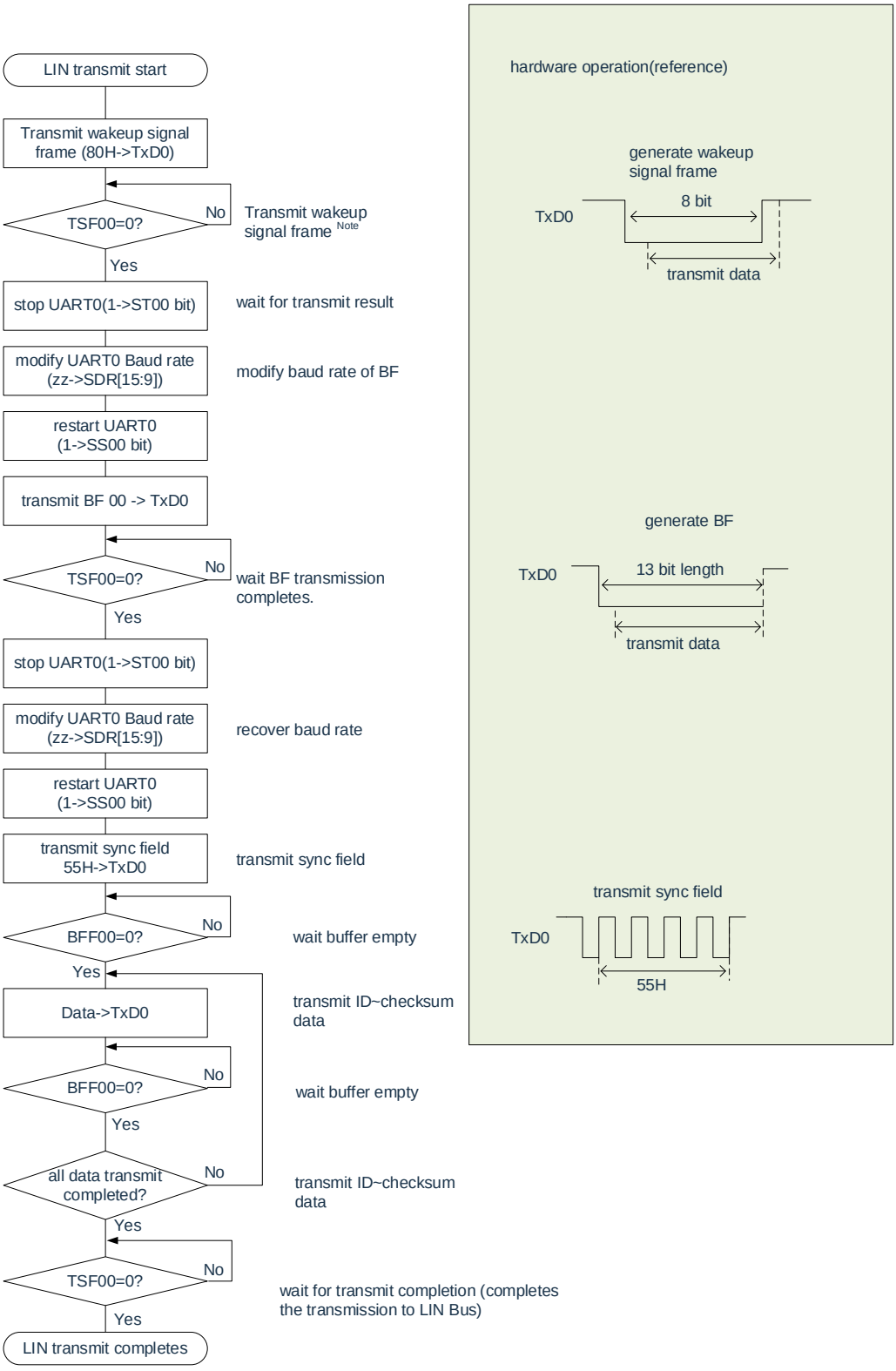
(Baud rate for break field) = 9/13×N

Transmit the data of "00H" through this baud rate to generate a break field.

Note 3: Output INTST0 at the end of each data transmission, and also output INTST0 at BF transmission.

Remark: The software controls the break between fields.

Figure 12-89: Flowchart of LIN transmission



Remark:

1. It is limit to situations starting from LIN-bus sleep.
2. This is the process that starts by ending the initial set-up of the UART and allowing slave sending.

12.8.2 LIN reception

In UART reception, UART0 supports LIN communication.

The LIN receives channel 1 of unit 0.

Table 12-41: LIN reception

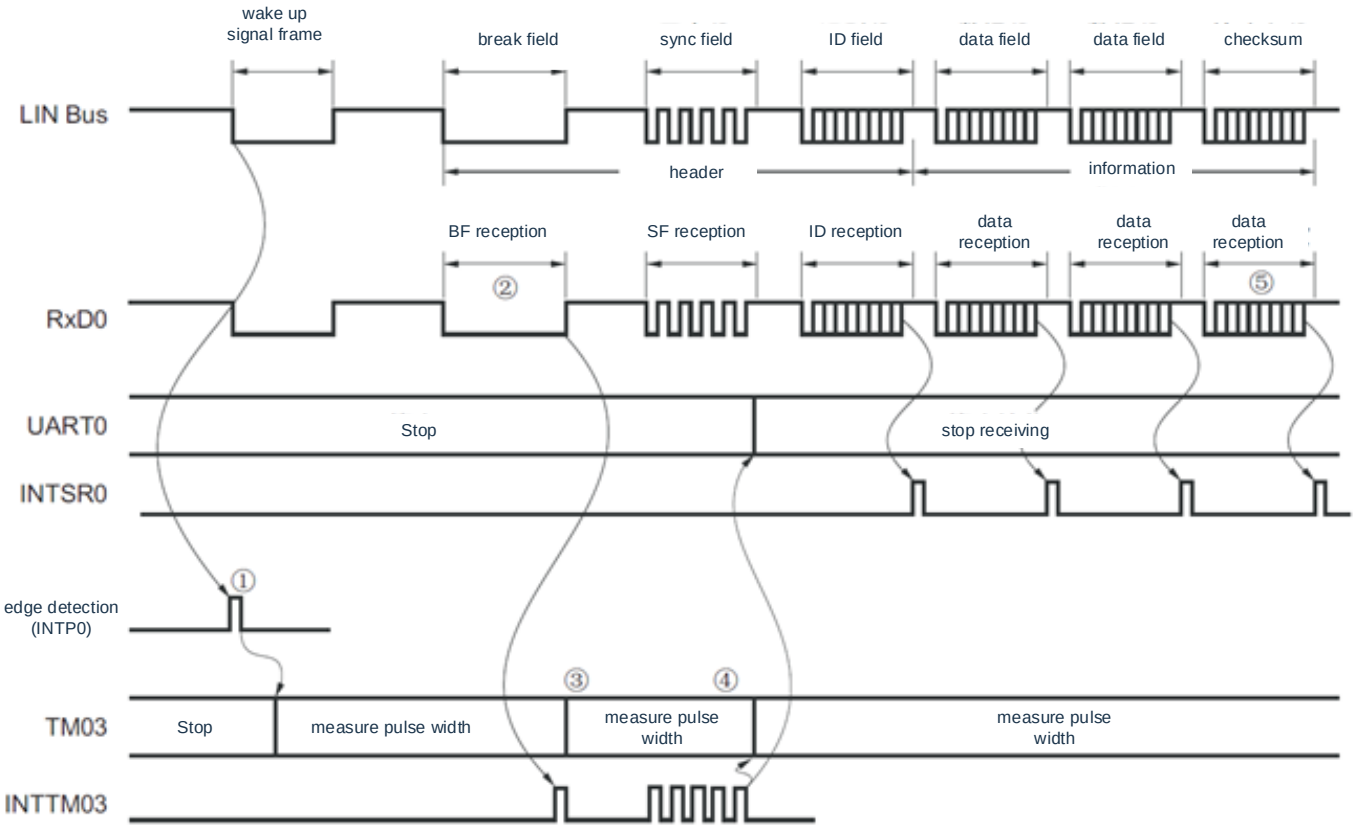
UART	UART0	UART1
LIN communication support	Yes	No
Target channel	Channel 1 of SCI0	—
Pins used	RxD0	—
Interrupt	INTSR0	—
	Limited to end-of-transmit interrupts (buffer empty interrupts are prohibited).	
Error interrupt	INTSRE0	—
Error detection flag	Frame error detection flag (FEF01) Overflow error detection flag (OVF01)	
Transfer data length	8 bits	
Transfer rate Note	Max.F_{MCK}/6[bps] (SDR01[15:9]≥2), Min.F_{CLK}/(22^{15}128)[bps]	
Data phase	Non-reverse output (default: high). Reverse output (default: low).	
Parity check bit	No parity bits (no parity).	
Stop bit	Appending 1 bit.	
Data direction	LSB first	

Notice: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheret).

F_{MCK}: Operation clock frequency of target channel F_{CLK}: System clock frequency.

A summary of the LIN receive operation is shown in Figure 12-90:

Figure 12-90: LIN reception



The signal processing flow is as follows:

(1) The wake-up signal is detected by detecting the INTP0 of the pin. When the wake signal is detected, the TM03 is set to measure the pulse width in order to measure the low-level width of BF.

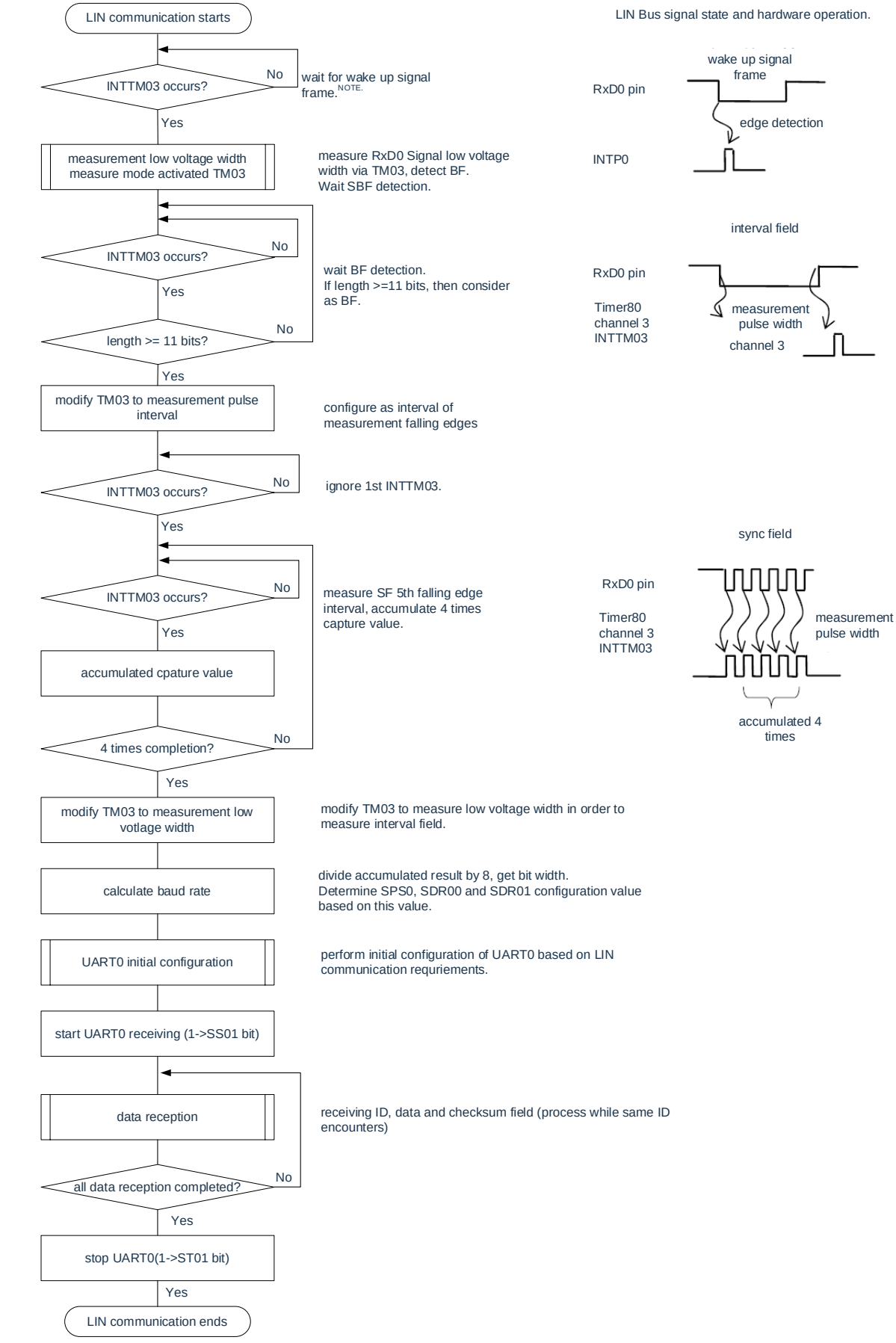
(2) If the falling edge of BF is detected, TM03 starts to measure the low-level width and captures the rising edge of BF. The BF signal is judged according to the captured value.

(3) When BF reception ends normally, TM03 must be set as the measurement pulse interval, and the interval of RxD0 signal falling edge of 4 synchronizations (See “5.8.4 Operation as input pulse interval measurement”).

(4) Calculating the baud rate error according to the bit interval of the synchronization section (SF). The baud rate must then be adjusted (reset) after the UART0 run has been paused.

(5) The checksum segment must be distinguished by software. You must also initialize the UART0 after receiving the checksum segment through the software and set it to the BF receive wait state again.

Figure 12-91: Flowchart of LIN reception



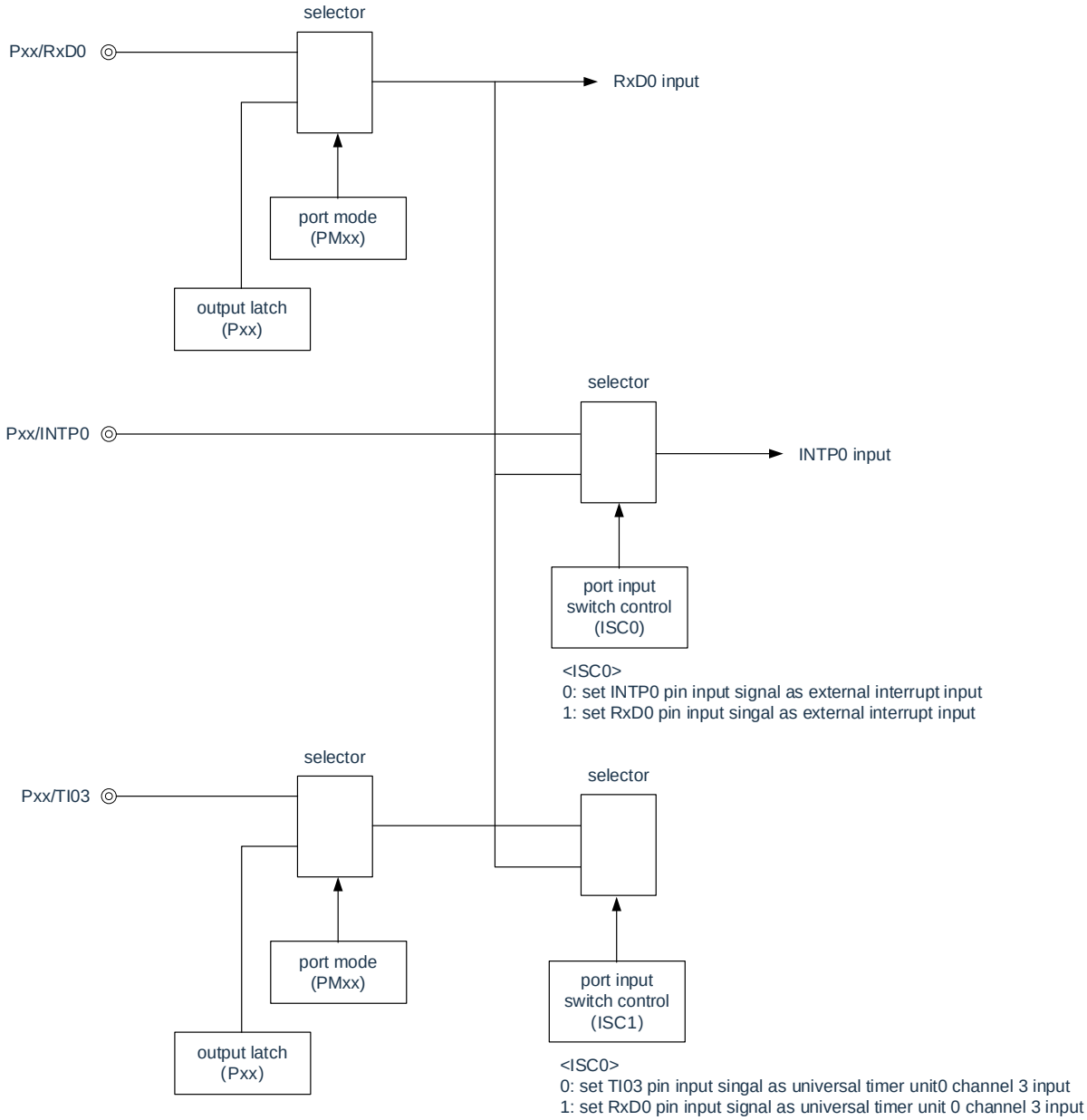
Notice: It is only needed in the sleeping state.

The port structure diagram for LIN receive operations is shown in Figure 12-92.

The wake-up signal sent by the LIN master is received through edge detection of the INTP0. The invention can measure the length of the sync field sent by the LIN master and calculate the baud rate error through external event capture operation.

The input source for the received port input (RxD0) can be input to the external interrupt (INTP0) and timer array unit without external connection by port input switching control (ISC0/ISC1).

Figure 12-92: Port structure diagram for LIN receive operation



Remark: ISC0, ISC1: bit0 and bit1 of the Input Switching Control Register (ISC)

Peripheral features for LIN communication operations are summarized as follows:

＜Peripheral Features Used＞

External interrupt (INTP0): Detection of wake-up signal

Purposes: Detects edges of wake-up signals and the start of communication.

(1) Channel 3 of General-Purpose Timer Unit: Detection of baud rate error, detection of break field (BF)

Purpose: Detects the length of a sync field (SF) and detects baud rate errors by dividing its length by the number of bits (the interval of the RxD0 input edge is measured in capture mode). Measures the width of a low level to determine if it is a break field (BF).

(2) Channel 0 and Channel 1 (UART0) of General-Purpose Serial Communication Unit 0 (SCI0)

Chapter 13 Serial Interface SPI

13.1 Function of serial interface SPI

This product is equipped with a SPI, and has the following two modes.

(1) Run stop mode

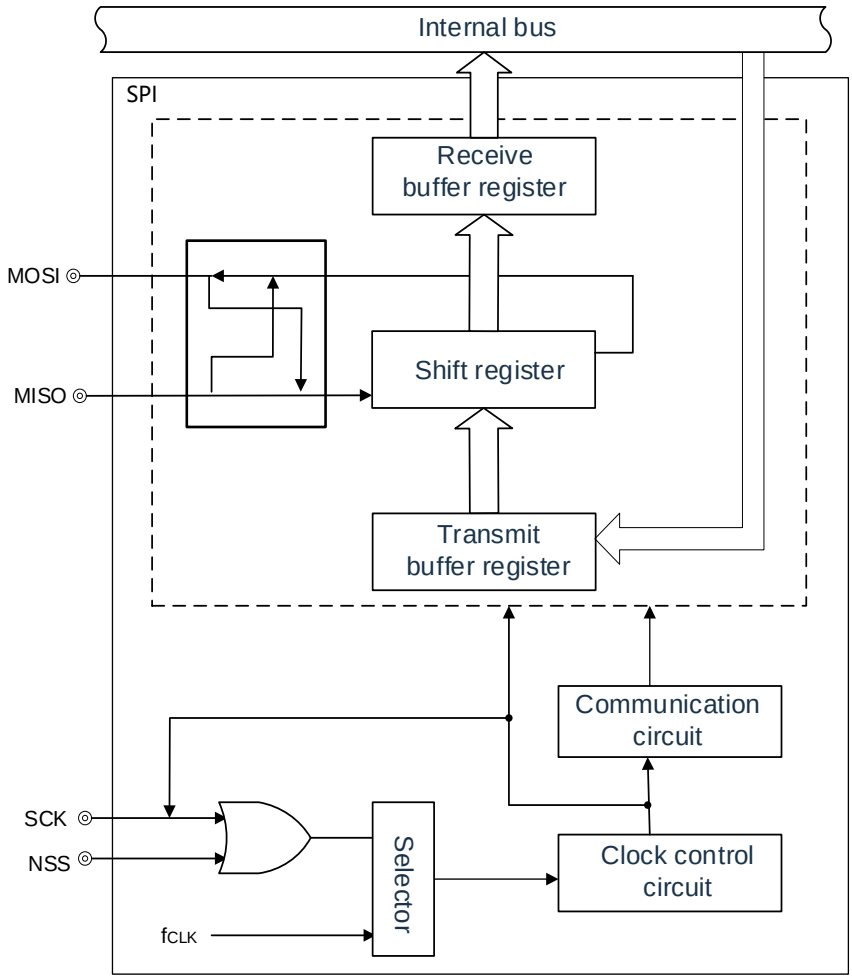
This is a mode used when serial transfer is not in progress and reduces power consumption.

(2) 3-wire serial I/O mode

This mode transmits 8-bit or 16-bit data to multiple devices via three lines of serial clock (SCK) and serial data bus (MISO and MOSI).

13.2 Structure of serial interface SPI

Figure 13-1: Diagram of serial interface SPI



13.3 Registers for controlling serial interface SPI

The serial interface SPI is controlled by the following registers.

(1) Peripheral enable register 0 (PER0)
(2) Serial operation mode register (SPIM)
(3) Serial clock selection register (SPIC)
(4) Transmit buffer register (SDRO)
(5) Receive buffer register (SDRI)
(6) Port mode register (PMxx)
(7) Port mode control register (PMCxx)
(8) Port register (Pxx)

13.3.1 Peripheral enable register 0 (PER0)

The PER0 register is a register that enables or disables the clock to be supplied to the peripheral hardware.
Reduce power consumption and noise by stopping clocks to hardware that is not in use.
To use the SPI function, SPIEN must be set to "1".
See “4.3.6 Peripheral Enable Registers 0, 1 (PER0, PER1)” for details.

13.3.2 SPI operation mode register (SPIM)

SPIM is used to select the mode of operation and control whether the operation is enabled or disabled.

SPIM can be set by an 8-bit memory manipulation instruction.

A reset signal is generated to clear the register to 00H.

Table 13-1: Format of SPI operation mode register (SPIM)

Address: 0x40042400 After reset: 00H R/W[Note1]

Symbol	7	6	5	4	3	2	1	0
SPIM	SPIE	TRMD	NSSE	DIR	INTMD	DLS	SDRIF	SPTF

SPIE	SPI operation enable
0	Stops operation.
1	Enables operation.

TRMD[Note3]	Transmit/receive mode control
0	Receives mode
1	Transmits/receives mode

NSSE[Note4]	Selection of NSS pin
0	NSS pin is not used
1	Uses NSS pin

DIR	Data transfer order selection
0	Perform MSB-first input/output.
1	Perform LSB-first input/output.

INTMD	Selection of interrupt sources
0	Transfer complete interrupt
1	Transfer buffer empty interrupt

DLS	Data length setting
0	Data length of 8 bits
1	Data length of 16 bits

SDRIF	Receive buffer non-empty flag bit
0	No newly received valid data in the receive buffer
1	The receive buffer contains the received valid data. This bit is cleared to 0 when the register SDRI is read

SPTF[Note2]	Communication status flag bit
0	Communication stops
1	Communication is ongoing

Note 1: Bit 0 and bit 1 are read-only bits.

Note 2: When SPTF=1 (during serial communication), rewriting of TRMD, DIR, NSSE is prohibited.

Note 3: When TRMD is 0, the MO or SO output is fixed low.

Note 4: Fix the NSS pin input level to 0 or 1 before setting this bit to 1.

13.3.3 SPI clock selection register (SPIC)

This register specifies the timing of data transmit/receive and sets the serial clock.

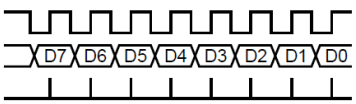
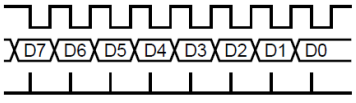
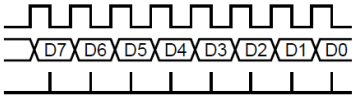
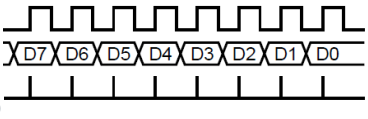
It can be set by an 8-bit memory manipulation instruction.

A reset signal is generated to clear this register to 01H.

Table 13-2: Format of SPI clock selection register (SPIC)

Address: 0x40042404 After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
SPIC	0	0	0	CKP	DAP	CKS2	CKS1	CKS0

CKP	DAP	Specification of data transmission/reception timing	Type
0	0		1
0	1		2
1	0		3
1	1		4

CKS2	CKS1	CKS0	SPI serial clock selection	Mode
0	0	0	F_{CLK}	Master mode
0	0	1	$F_{CLK}/2$	
0	1	0	$F_{CLK}/2^2$	
0	1	1	$F_{CLK}/2^3$	
1	0	0	$F_{CLK}/2^4$	
1	0	1	$F_{CLK}/2^5$	
1	1	0	$F_{CLK}/2^6$	
1	1	1	External clock input from SCK	Slave mode

Notice:

1. Write TOPICn is prohibited when SPIE=1 (operation enabled).
2. The phase type of the data clock after reset is type 1.

13.3.4 Transmit buffer register (SDRO)

This register sets the transmit data.

When bit 7 (SPIE) and bit 6 (TRMD) of the Serial Operation Mode Register (SPIM) are set to 1, transmit/receive is started by writing data to the SDRO.

The serial I/O shift register converts the data in the SDRO from parallel data to serial data and outputs it to the serial output pins.

The SDRO can be written or read by 8-bit or 16-bit memory manipulation instructions.

A reset signal is generated to clear this register to 0000H.

Table 13-3: Format of transmit buffer register (SDRO)

Address: 0x40042408							After reset: 0000H							R/W		
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRO	SDRO															

13.3.5 Receive buffer register (SDRI)

This register stores the received data.

If bit 6 (TRMD) of the Serial Operation Mode Register (SPIM) is set to 0, reception is started by reading data from the SDRI.

During reception, data is read from the serial input pins into the SDRI.

The SDRI can be read by 8-bit or 16-bit memory manipulation instructions.

A reset signal is generated to clear this register to 0000H.

Table 13-4: Format of receive buffer register (SDRI)

Address: 0x4004240C							After reset: 0000H							R		
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRI	SDRI															

13.3.6 Registers controlling port functions of SPI pins

When using SPI, you must set the control registers (Port Mode Register (PMxx, PMCxx)) for the port functions that are multiplexed with the SPI input and output pins. For details, refer to “2.3.1 Port Mode Register (PMxx)”.

When using the multiplexed port of the SPI pin as an output of SCK/SO/MO, set the bits of each port corresponding to the port mode registers (PMxx, PMCxx) to "0". When the multiplexed port of the SPI pin is used as the input of SCK/SI/MI, the bit of the port mode register (PMxx) corresponding to each port must be set to "1" and the bit of PMCxx must be set to "0". In this case, the bit of the port register (Pxx) can be "0" or "1".

For details, refer to “2.3 Registers for controlling port functions”

13.4 Operation of serial interface SPI

In 3-wire serial I/O mode, data is transmitted or received by 8-bit or 16-bit. The data is transmitted or received synchronously with the serial clock.

After communication begins, bit 0 (SPTF) of SPIM is set to 1. When the communication of data is completed, set the communication completion interrupt request flag (SPIIF) and clear SPTF to 0. The next communication is then enabled.

Notice:

1. When SPTF=1 (during serial communication), access to control registers and data registers are prohibited.
2. It must be used within the range that satisfies the SCLK cycle time (T_{KCY}) characteristics. Refer to the datasheet for details.

13.4.1 Master tramission and reception

If the bit 6 (TRMD) of the serial operation mode register (SPIM) is 1, data can be transmitted or received. When a value is written to the transmit buffer register (SDRO), transmission/reception starts.

(1) Procedure

Figure 13-2: Initial setup steps for master transmission/reception

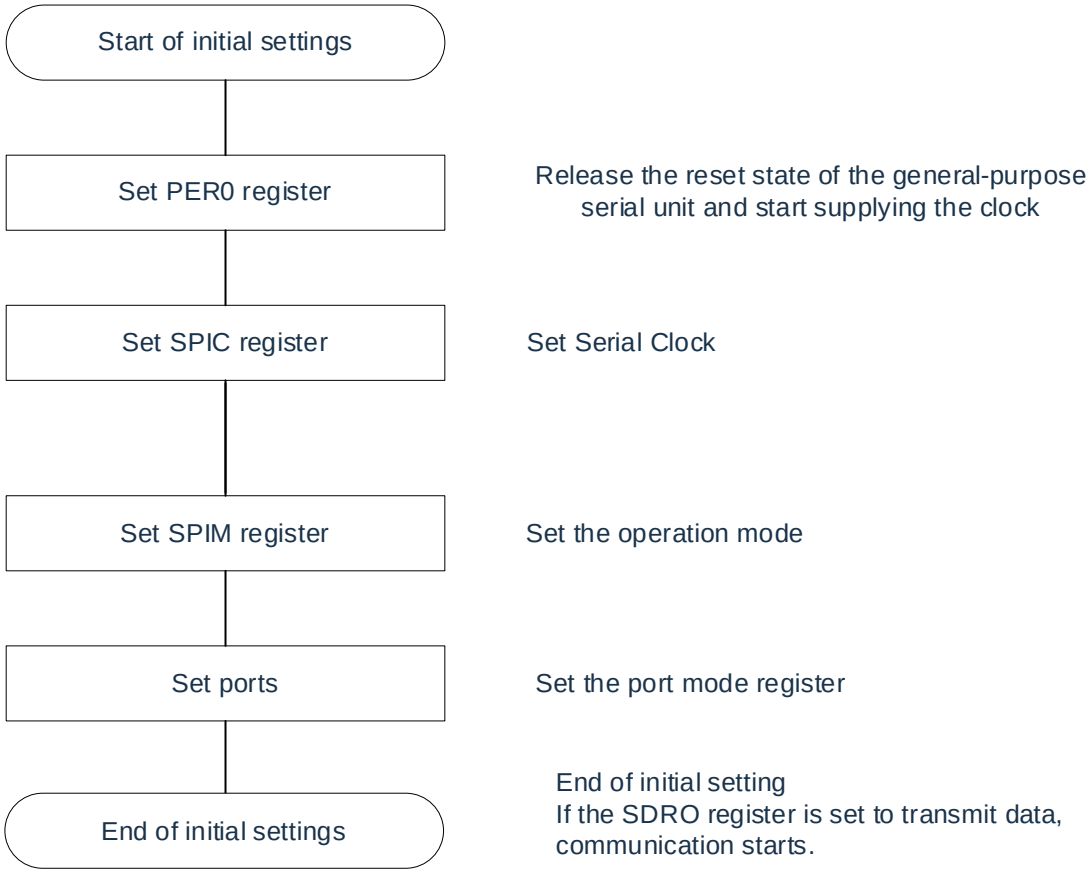
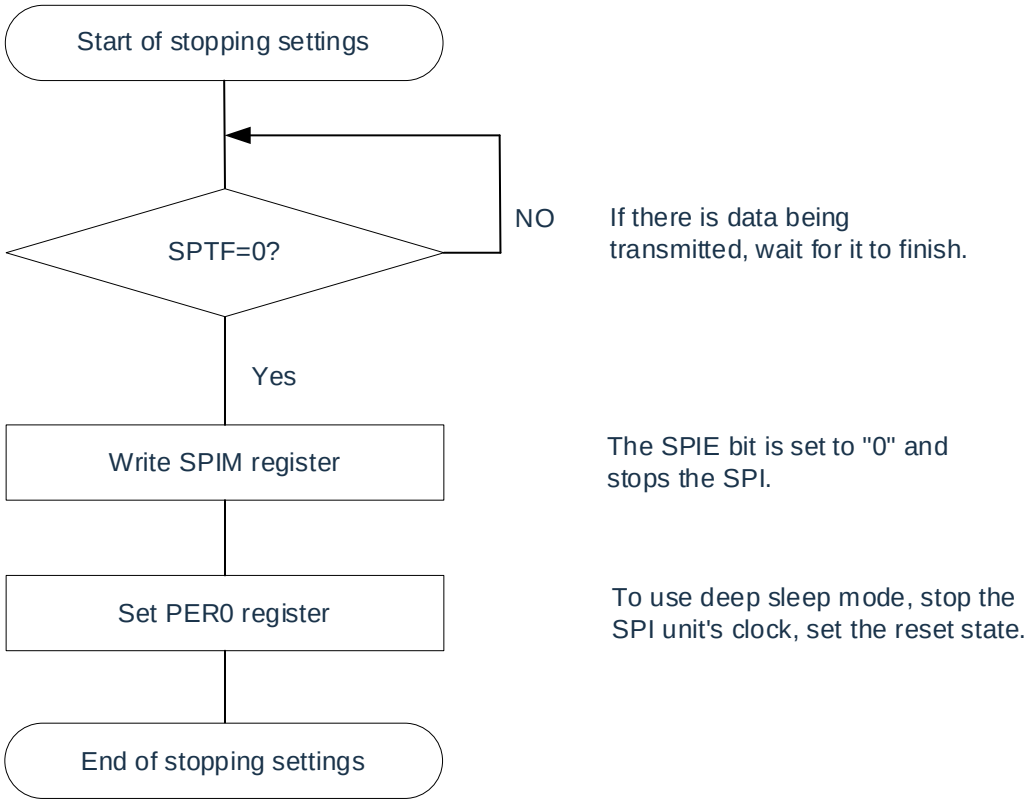


Figure 13-3: Stop steps of master transmission/reception



(2) Process flow

Figure 13-4: Timing diagram for transmit/receive (single transmit mode) (INTMD=0, DAP=0, CKPmn=0)

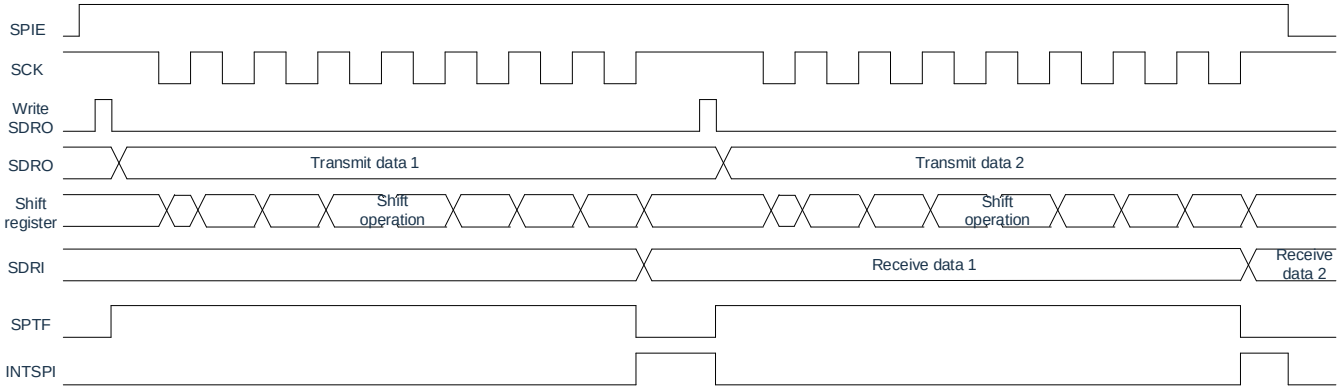
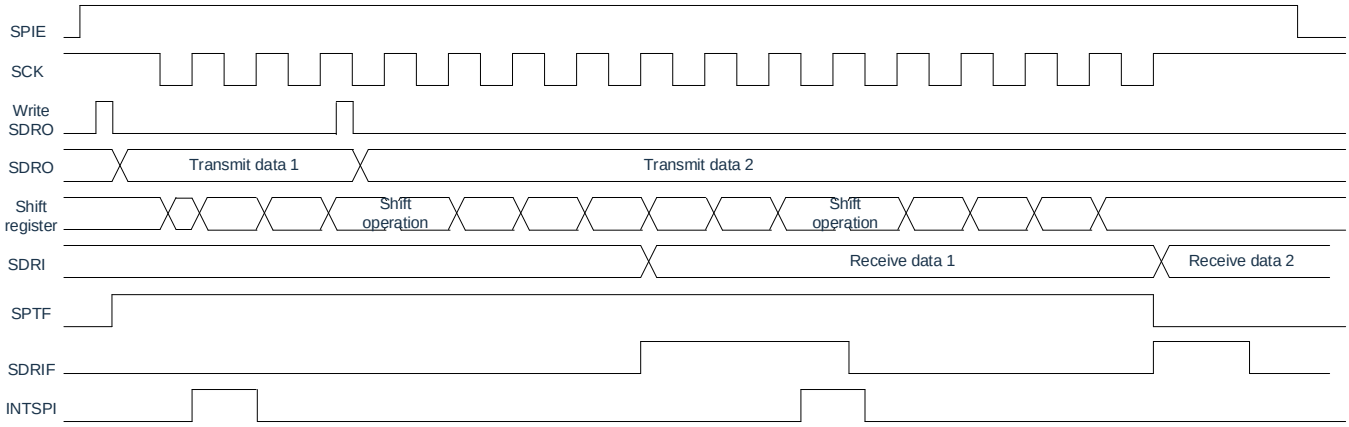


Figure 13-5: Timing diagram for transmit/receive (continuous transmit mode) (INTMD=1, DAP=0, CKPmn=0)



13.4.2 Master reception

If bit 6 (TRMD) of the Serial Operation Mode Register (SPIM) is 0, only data can be received. Reception begins when data is read from the receive buffer register (SDRI).

(1) Procedure

Figure 13-6: Initial setup steps for master reception

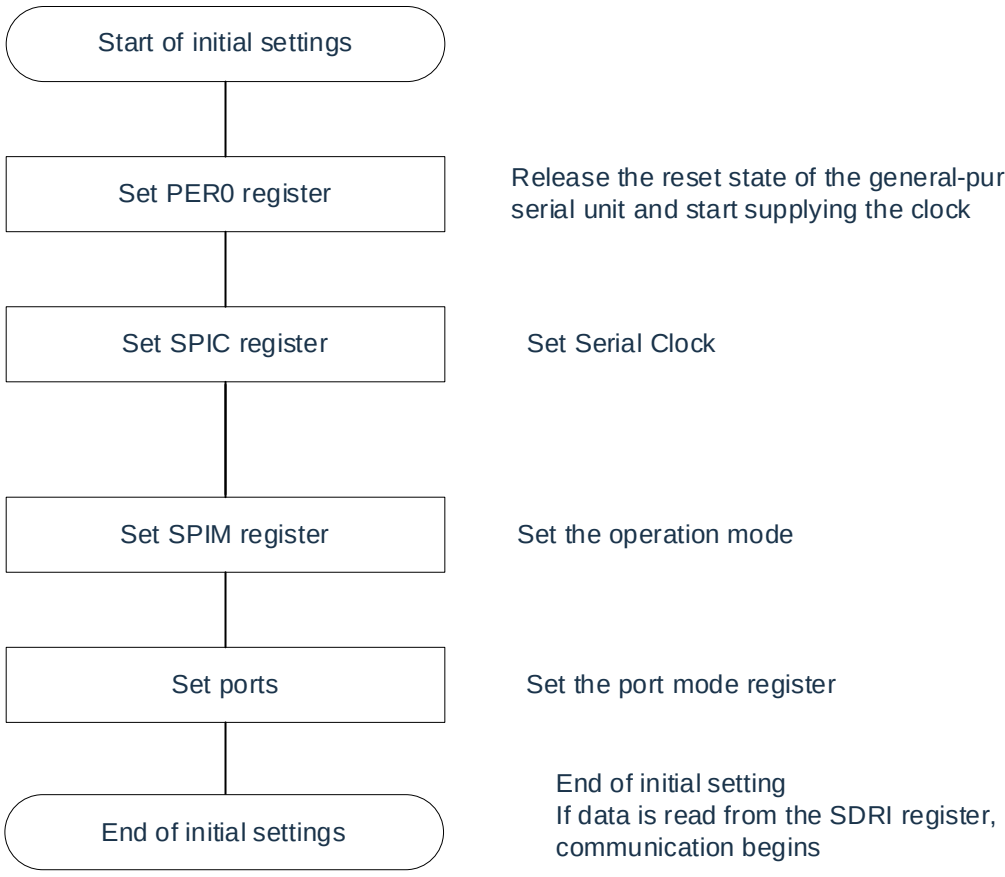
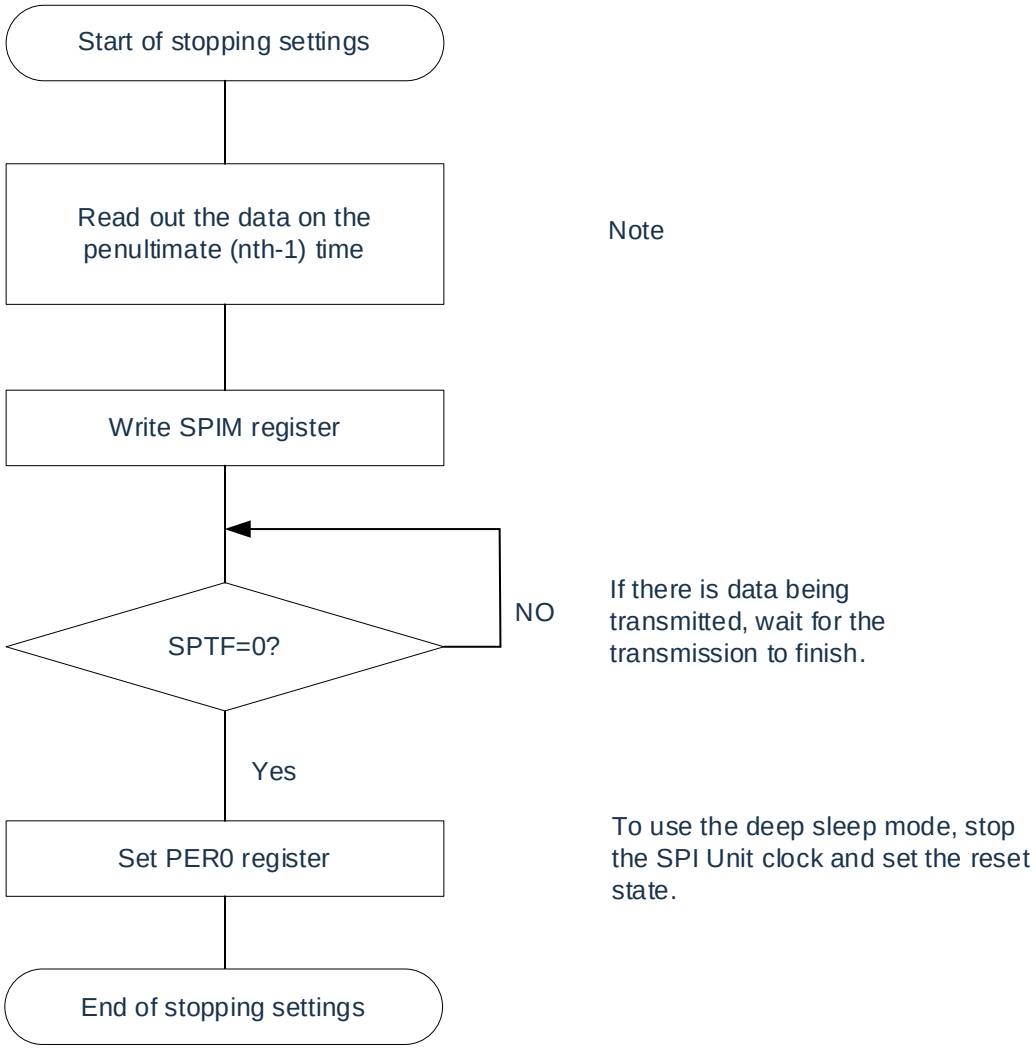


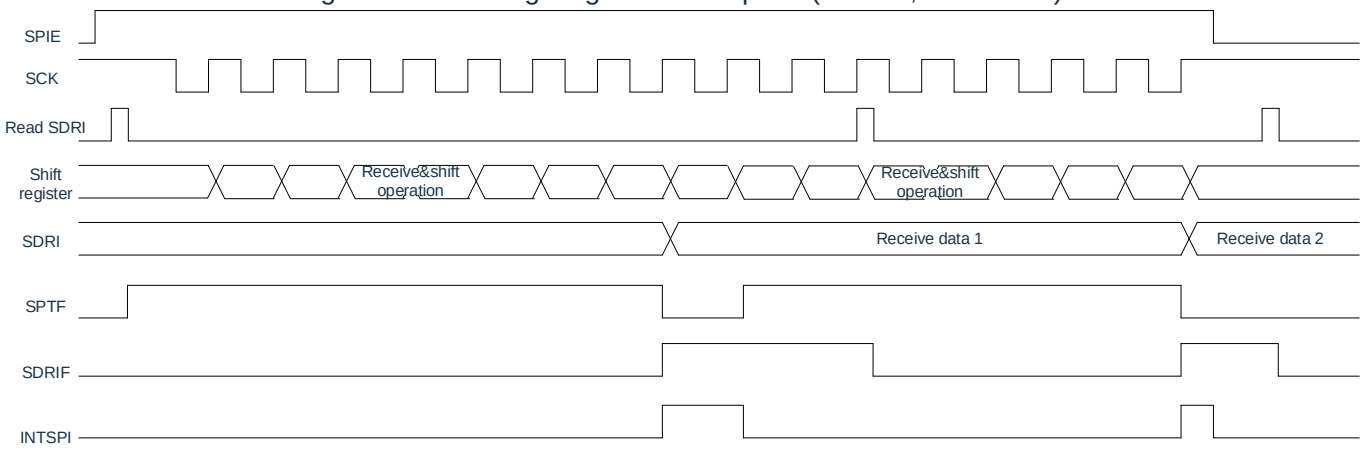
Figure 13-7: Stop steps for master reception



Note: In receive-only mode, the SPI transmission is triggered by reading the value of the SDRI register. If the SPI action is not aborted in time, there may be a redundant transmission after the last SDRI read. If you want to avoid the last redundant transmission, you can wait for one SCK cycle after the penultimate read and then turn off the SPIE. The SPI transmission will be aborted after the last data transmission is completed.

(2) Process flow

Figure 13-8: Timing diagram of reception (DAP=0, CKPmn=0)



13.4.3 Slave transmission and reception

If slave mode is selected by bits CKS2-0 of the Serial Clock Selection Register (SPIC) and bit 6 (TRMD) of the Serial Operation Mode Register (SPIM) is 1, the slave transmit/receive mode is entered. When a value is written to the transmit buffer register (SDRO), wait for the clock from the master device to start transmitting/receiving.

(1) Procedure

Figure 13-9: Initial setup steps for slave transmission and reception

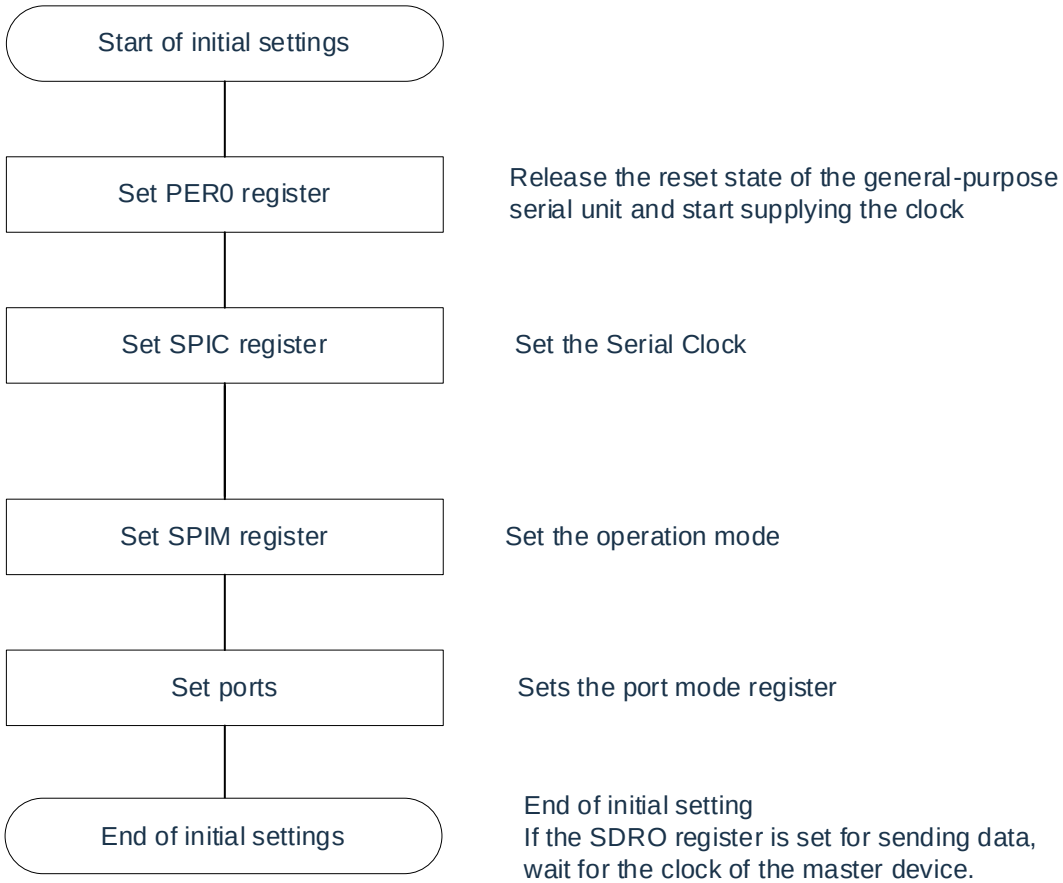
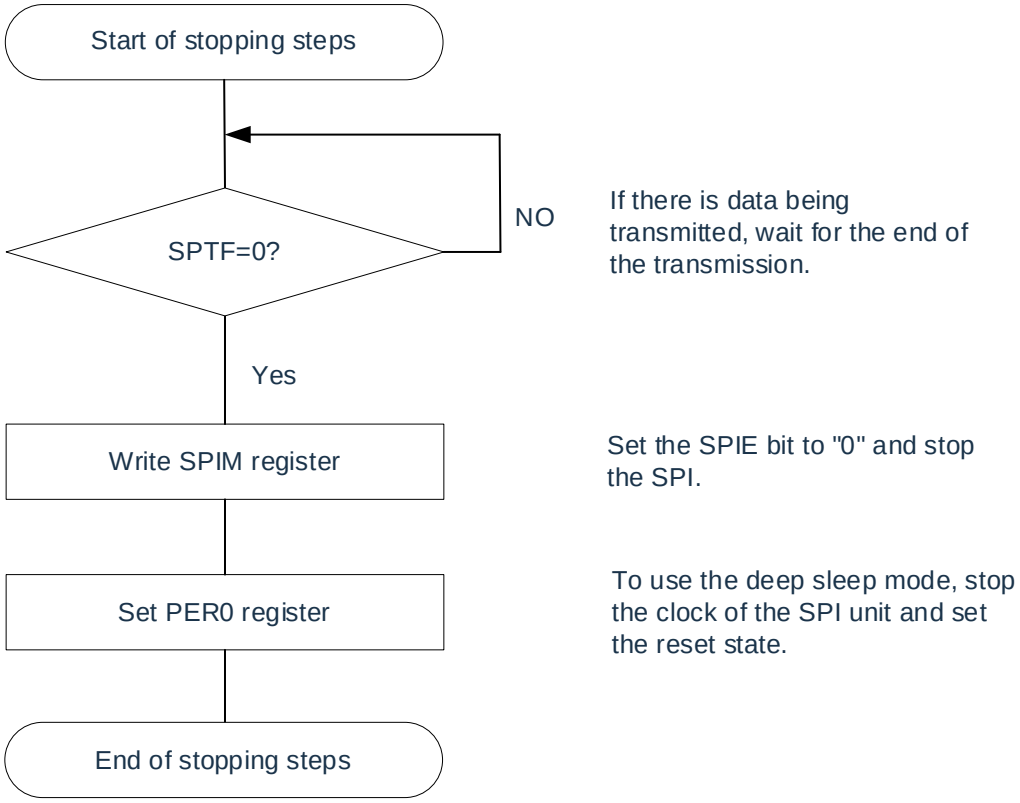


Figure 13-10: Stop steps for slave transmission and reception



(2) Process flow

Figure 13-11: Timing diagram for transmit/receive (single transmission mode) (INTMD=0, DAP=0, CKPmn=0)

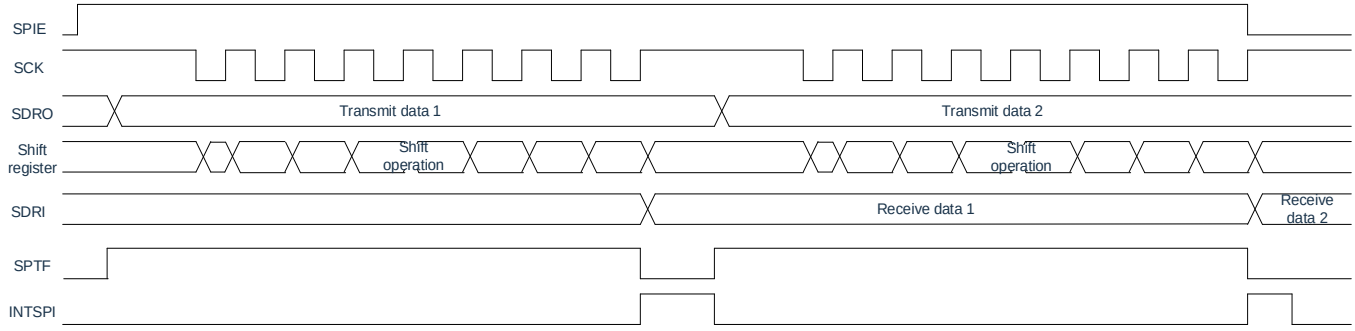
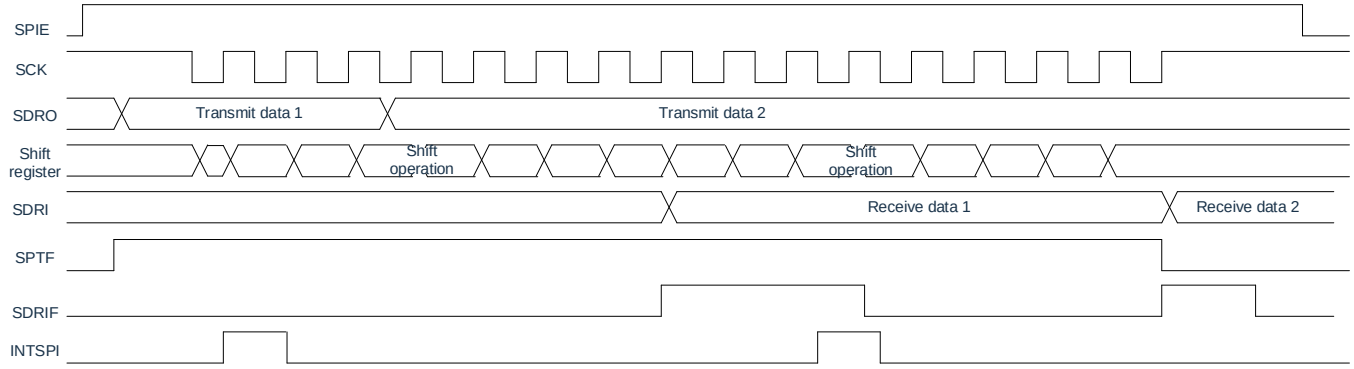


Figure 13-12: Timing diagram for transmit/receive (continuous transmit mode) (INTMD=1, DAP=0, CKPmn=0)



13.4.4 Slave reception

If slave mode is selected by bit CKS2-0 of the Serial Clock Selection Register (SPIC) and bit 6 (TRMD) of the Serial Operation Mode Register (SPIM) is 0, the slave receive mode is entered. When data is read from the receive buffer register (SDRI), wait for the clock from the master device to begin reception.

(1) Procedure

Figure 13-13: Initial setup steps for slave reception

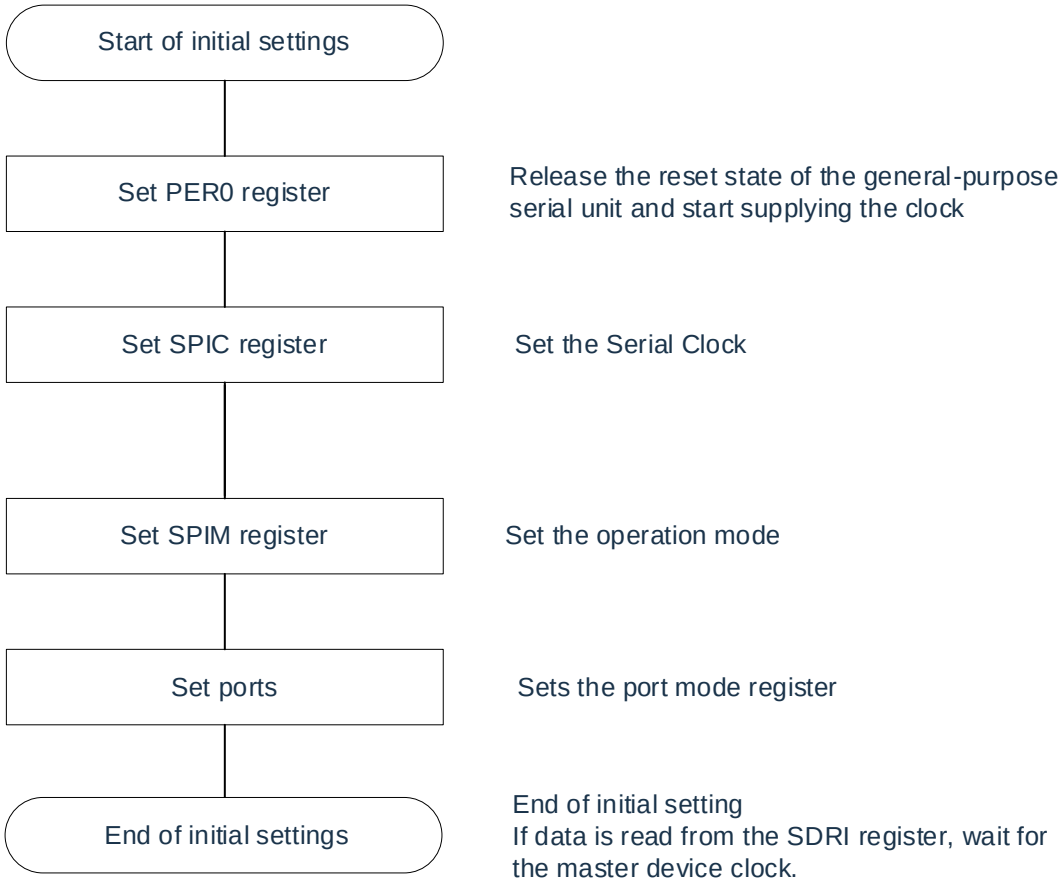
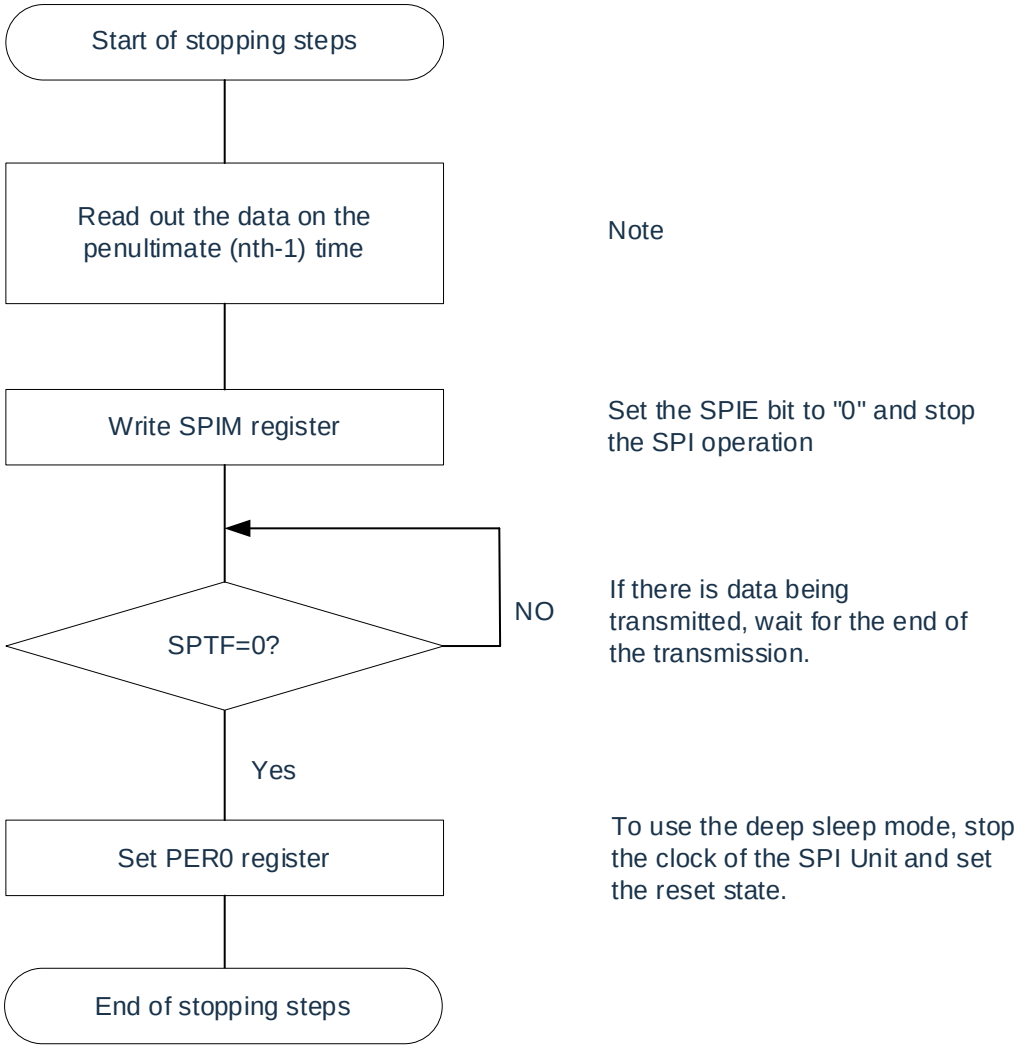


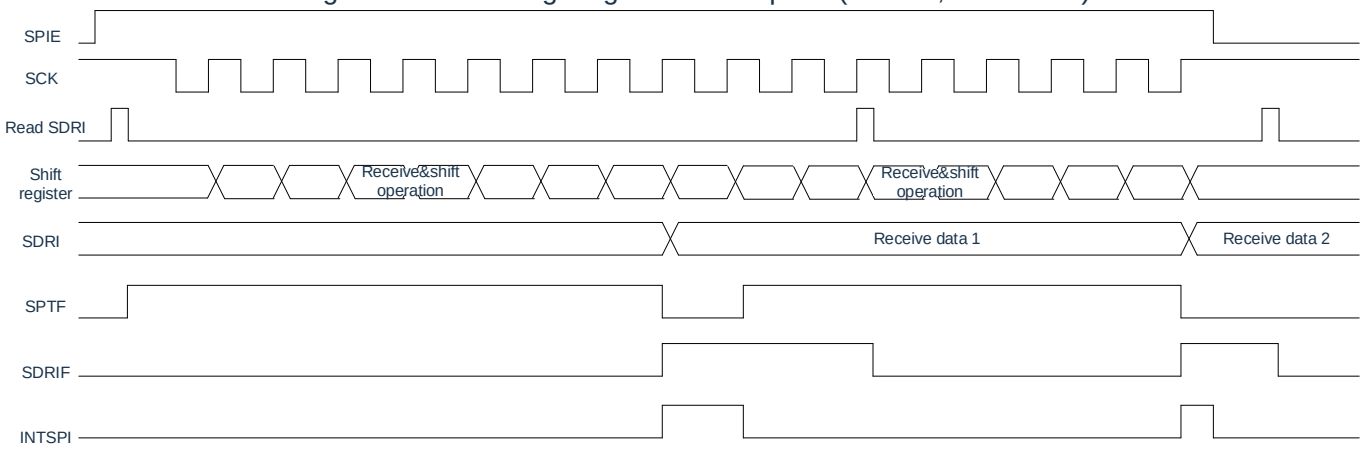
Figure 13-14: Stop steps for slave reception



Note: In receive-only mode, the SPI transmission is triggered by reading the value of the SDRI register. If the SPI action is not aborted in time, there may be a redundant transmission after the last SDRI read. If you want to avoid the last redundant transmission, you can wait for one SCK cycle after the penultimate read and then turn off the SPIE. The SPI transmission will be aborted after the last data transmission is completed.

(2) Process flow

Figure 13-15: Timing diagram for reception (DAP=0, CKPmn=0)



Chapter 14 Serial Interface IICA

14.1 Function of serial interface IICA

The serial interface IICA has the following 3 modes.

(1) Run stop mode

This is a mode used when serial transfer is not in progress and reduces power consumption.

(2) I^2C bus mode (supports multi-master)

This mode transmits 8-bit data to multiple devices via two lines of serial clock (SCLAn) and serial data bus (SDAAn). Conforming to the I^2C-bus format, the master device can generate “start conditions” and "addresses" for the slave device on the serial data bus, Indication of Transfer Direction, Data, and Stop Condition. The slave automatically detects the received status and data through the hardware. This feature simplifies the I^2C-bus control part of the application.

Because the SCLAn pin and SDAAn pin of the serial interface IICA are used as open-drain outputs, the serial clock line and serial data bus require pull-up resistors.

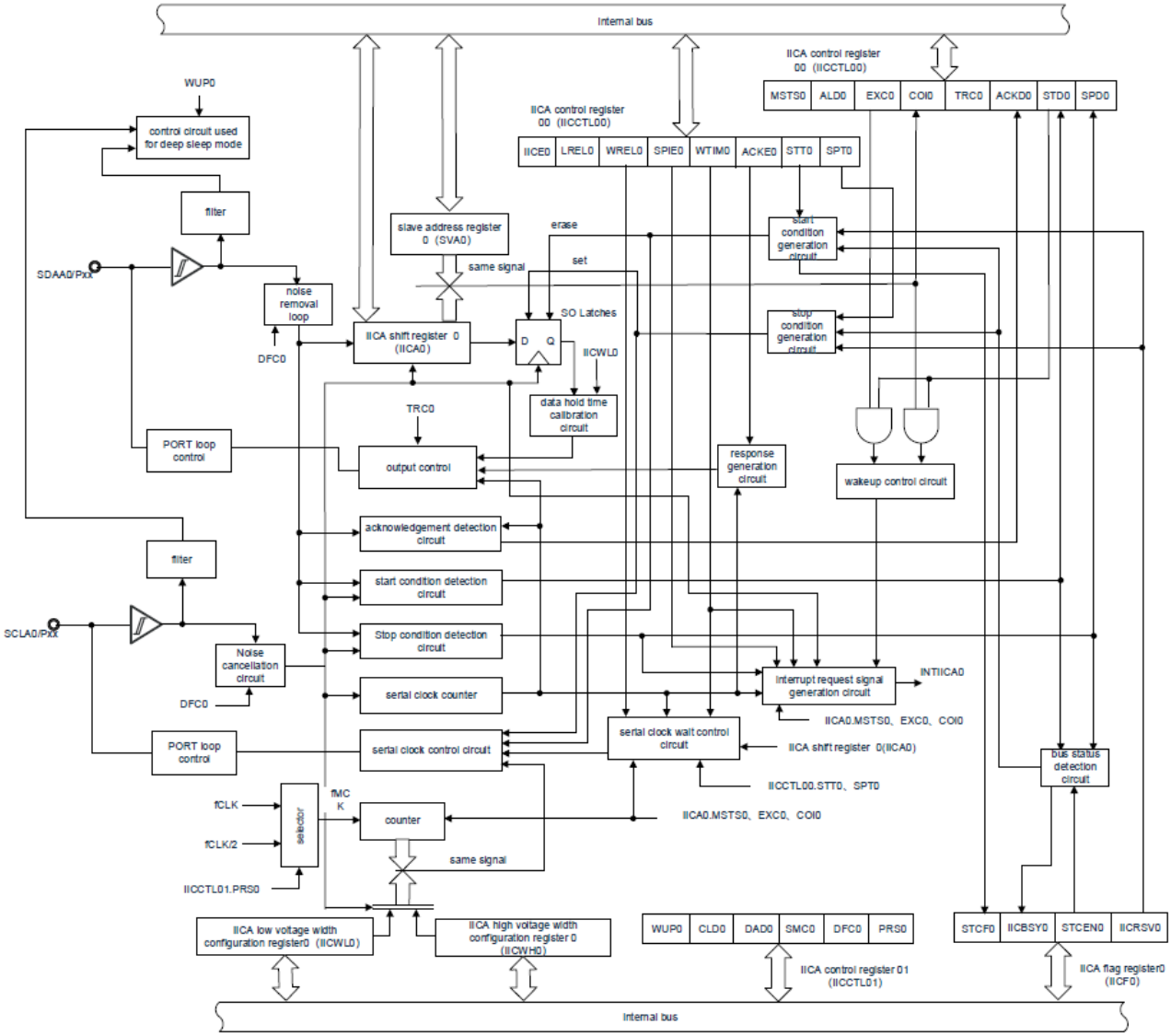
(3) Wake-up mode

In deep sleep mode, when receiving an extension code or local station address of the autonomous control device, the deep sleep mode can be released by generating an interrupt request signal (INTIICAn). It is set via the WUPn bit of IICA control register n1 (IICCTLn1).

A block diagram of the serial interface IICA is shown in Figure 14-1.

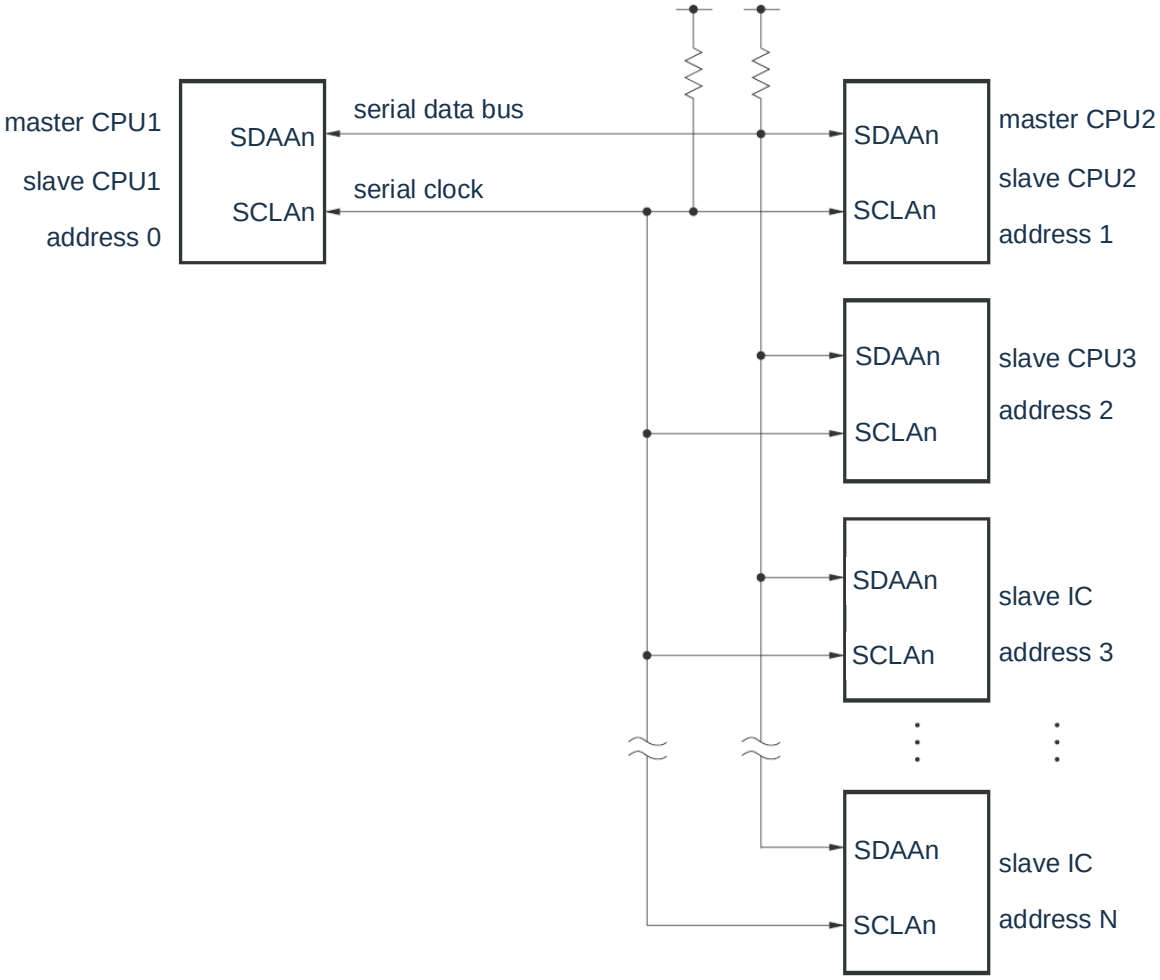
Remark: n=0

Figure 14-1: Block diagram of serial interface IICA



An example of the structure of a serial bus is shown in Figure 14-2.

Figure 14-2: Example of a serial bus structure for I^2C bus



Remark: n=0

14.2 Structure of serial interface IICA

The serial interface IICA consists of the following hardware.

Table 14-1: Structure of serial interface IICA

Item	Structure
Registers	IICA shift register n (IICAn) Slave address register n (SVAn)
Control registers	Peripheral enable register 0 (PER0) IICA control register n0 (IICCTLn0). IICA status register n (IICSn). IICA flag register n (IICFn). IICA control register n1 (IICCTLn1). IICA low-level width setting register n (IICWLn) IICA high-Level width setting register n (IICWHn) Port mode register (PMxx) Port mode control register (PMCxx). Port multiplexing function configuration register (PxxCFG)

Remark:

1. n=0
2. This product can multiplex the IICA input/output pin function to multiple ports. When a port is configured for multiplexing on the IICA pin, the N-channel open-drain output (V_{DD}/EV_{DD} withstand voltage) mode of the port is designed to automatically open, i.e. the POMxx register does not require user settings.

(1) IICA shift register n (IICAn)

The IICn register is a register that synchronizes with the serial clock for interconversion of 8-bit serial data and 8-bit parallel data for transmission and reception. Actual sending and receiving can be controlled by reading and writing IICAn registers.

During the wait, the wait is released by writing the IICAn register and the data is transferred. The IICAn registers are set via an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "00H".

Table 14-2: Format of IICA shift register n (IICAn)

Address:	0x40041B50	After reset:	00H				R/W	
Symbol	7	6	5	4	3	2	1	0
IICAn								

Notice:

1. During data transfer, data cannot be written to the IICAn register.
2. The IICAn register can only be read and written during the waiting period. Access to the IICAn registers in the communication state is prohibited except during the waiting period. However, in the case of the master device, the IICAn register can be written once after the communication trigger bit (STTn) is set to "1".
3. When making an appointment for communication, data must be written to the IICAn register after detecting an interrupt caused by a stop condition.

Remark: n=0

(2) Slave address register n(SVAn)

This is the register that holds the 7-bit local station address {A6,A5,A4,A3,A2,A1,A0} when used as a slave.

The SVAn register is set by the 8-bit memory manipulation instruction. However, when the STDn bit is "1" (start condition detected), it is forbidden to overwrite this register.

After a reset signal is generated, the value of this register becomes "00H".

Table 14-3: Format of slave address register n (SVAn)

Address:	0x40041A34	After reset:	00H			R/W		
Symbol	7	6	5	4	3	2	1	0
SVAn	A6	A5	A4	A3	A2	A1	A0	0 Note

Note: Bit 0 is fixed to "0".

(3) SO latch

The SO latch holds the output level of the SDAAn pin

(4) Wake-up control circuit

This circuit generates an interrupt request (INTIICAn) when the address value set to the slave address register n(SVAn) is the same as the received address or when an extension code is received.

(5) Serial clock counter

During transmission or reception, this counter counts the output or input serial clocks and checks whether 8-bit data transmission and reception are performed.

(6) Interrupt request signal generation circuit

This circuit controls the generation of the interrupt request signal (INTIICAn). The I^2C interrupt request is generated by the following two triggers.

Drop of the 8th or 9th serial clock (set by the WTIMn bit)

Interrupt request (set via SPIEn bit) due to detection of a stop condition.

Remark:

1. WTIMn bit: bit3 of IICA control register n0 (IICCTLn0)
2. SPIEn bit: bit4 of IICA control register n0 (IICCTLn0)

(7) Serial clock control circuit

In master mode, this circuit generates the clock output to the SCLAn pin from the sample clock.

(8) Serial clock wait control circuit

This circuit controls the wait timing.

(9) Ack generation circuit, stop condition detection circuit, start condition detection circuit, Ack detection circuit

These circuits generate and detect various states.

(10) Data hold time correction circuit

This circuit generates a data hold time for the serial clock to drop.

(11) Start condition generation circuit

If STTn is "1", the circuit generates a start condition.

However, in a state where appointment communication is prohibited (IICRSVn bit =1) and the bus is not released (IICBSYn bit=1), the start condition request is ignored and the STCFn is set to "1".

(12) Stop condition generation circuit

If the SPTn bit is "1", the circuit generates a stop condition.

(13) Bus status detection circuit

This circuit detects whether the bus is released by detecting the start and stop conditions. However, the bus state cannot be detected immediately at the very beginning of operation, so the initial state of the bus state detection circuit must be set by the STCENn bit.

Remark:

1. STTn bit: Bit1 of IICA control register n0 (IICCTLn0)
2. SPTn bit: Bit0 of IICA control register n0 (IICCTLn0)
3. IICRSVn bit: Bit0 of IICA flag register n (IICFn)
4. IICBSYn bit: Bit6 of IICA flag register n (IICFn)
5. STCFn bit: Bit7 of IICA flag register n (IICFn)
6. STCENn bit: Bit1 of IICA flag register n (IICFn)
7. n=0

14.3 Registers for controlling serial interface IICA

The serial interface IICA is controlled by the following 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 (PMxx)

Port mode control register (PMCxx)

Port multiplexing function configuration register (PxxCFG)

Port mode register (PMxx)

Port mode control Register (PMCxx)

Port multiplexing function configuration register (PxxCFG)

Note: n=0

14.3.1 Peripheral enable register 0 (PER0)

The PER0 register is the register that sets whether to enable or disable the supply of clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocks to hardware that is not in use.

To use the serial interface IICAn, bit4 (IICAEN) must be set to "1".

The PER0 register is set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register becomes "00H".

Table 14-4: Format of peripheral enable register 0(PER0)

Address: 0x40020420 After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
PER0	RTCEN	0	ADCEN	IICAEN	SCI1EN	SCI0EN	TM41EN	TM40EN

IICAEN	Control of input clocks supplied to serial interface IICA
0	Stop to supply the input clock. The SFR used by the serial interface IICA cannot be written The serial interface IICA is in reset state.
1	Enable providing input clock The SFRs used by the serial interface IICA can be read and written.

Notice: To set the serial interface IICA, the following registers must first be set when the IICAnEN bit is "1". When the IICAnEN bit is "0", the value of the control register of the serial interface IICAn is the initial value, ignoring the write operation (except port multiplexing function configuration register (PxxCFG), port mode registers (PMxx), and port mode control registers (PMCxx).

(1) IICA control register n0 (IICCTLn0).
(2) IICA flag register n (IICFn)
(3) IICA status register n (IICSn)
(4) IICA control register n1 (IICCTLn1)
(5) IICA low-level width setting register n (IICWLn)
(6) IICA high level width setting register n (IICWHn)

Remark: n=0

14.3.2 IICA control register n0 (IICCTLn0)

This is a register that enables or stops I^2C operation, sets the wait sequence, and sets other registers for I^2C operation.

The IICCTLn0 register is set by an 8-bit memory manipulation instruction. However, the SPIEn bit, WTIMn bit, and ACKEn bit must be set when the IICEn bit is "0" or during a wait period, and these bits can be set at the same time when the IICEn bit is set from "0" to "1".

After a reset signal is generated, the value of this register becomes "00H".

Remark: n=0

Table 14-5: Format of IICA control register n0(IICCTLn0) (1/4)

Address: 0x40041A30 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 and stop internal operation.
1	Enable operation.
This bit set to "1" in the state where the SCLAn and SDAAn lines are high.	
Clear condition (IICEn=0)	Set condition (IICEn=1)
Cleared by intructions. When resetting	Set by intructions

LRELn Note2,3	Exit of communication
0	Normal operation
1	Exits the current communication and enters idle status. Automatically clear "0" after execution. Use in cases such as receiving extension codes that are not related to the local station. The SCLAn line and the SDAAn line become high impedance. The following flags in IICA control register n0 (IICCTLn0) and IICA status register n (IICSn) are cleared "0" •STTn•SPTn•MSTSn•EXCn•COIn•TRCn•ACKDn•STDn
Enters idle state to exit the communication until the following communication participation conditions are met. Boot as master device after detecting a stop condition. Addresses match or extended codes are received after the start condition is detected	
Clear condition (LRELn=0)	Set condition (LRELn=1)
Automatically clears after execution. When resetting	Set by intructions

WRELn Note2,3	Release waiting
0	The wait is not released.
1	Release wait. Automatically clears after the wait is released.
If the WRELn bit (release from wait) is set during the 9th clock wait in the transmit state (TRCn=1), the SDAAn line changes to a high impedance state (TRCn=0).	
Clear condition (WRELn=0)	Set condition (WRELn=1)
Automatically cleared after execution. When resetting	Set by intructions

Note 1: A reset is performed on the STCFn bit and the IICBSYn bit of the IICA shift register n (IICAn), the IICA flag register n (IICFn), and the CLDn bit and the DADn bit of the IICA control register n1 (IICTLn1).

Note 2: In the state where the IICEn bit is "0", the signal for this bit is invalid.

Note 3: The read values for LRELn bits and WRELn bits are always "0".

Notice: If the SCLAn line is high, the SDAAn line is low, and the digital filter is ON (DFCn=1 for the IICCTLn1 register), then enable I^2C operation (IICEn=1) immediately detects the start condition. At this point, the LRELn bit is set to "1" through the bit memory manipulation instruction after enabling I^2C to run (IICEn=1).

Remark: n=0

Table 14-5: Format of IICA control register (IICCTLn0) (2/4)

SPIEn[Note1]	Enable or disable interrupt requests generated by stop condition detection
0	Disable
1	Enable
When the WUPn bit of IICA control register n1 (IICCTLn1) is "1", even if the SPIEN is "1", there is also no stop condition interrupt.	
Clear condition (SPIEn=0)	Set condition (SPIEn=1)
Cleared by intructions. When resetting	Set by intructions

WTIMn[Note1]	Control of wait and interrupt requests
0	An interrupt request signal is generated on the falling edge of the 8th clock. Master: After outputting 8 clocks, set the clock output low to wait. Slave: After inputting 8 clocks, set the clock low and wait for the master.
1	An interrupt request signal is generated on the falling edge of the 9th clock. Master: After outputting 9 clocks, set the clock output low to wait. Slave: After inputting 9 clocks, set the clock low and wait for the master.
An interrupt is generated on the falling edge of the 9th clock during the address transfer, regardless of the setting of this bit; the setting of this bit is valid at the end of the address transfer. The master device enters the wait state on the falling edge of the 9th clock during address transmission. A slave device that receives a local station address enters the wait state on the falling edge of the 9th clock after an acknowledge (ACK) is generated, but a slave device that receives an extension code enters the wait state on the falling edge of the 8th clock.	
Clear condition (WTIMn=0)	Set condition (WTIMn=1)
Cleared by intructions. When resetting	Set by intructions

ACKEn[Note1,2]	ACK control
0	No ack.
1	Enables ACK. Sets the SDAAn line low during the 9th clock.
Clear condition (ACKEn=0)	Set condition (ACKEn=1)
Cleared by intructions. When resetting	Set by intructions

Note 1: The signal of this bit is invalid when the ICEn bit is "0". This bit must be set in the meantime.

Note 2: The set value is invalid when it is not an extension code during address transmission. When it is a slave device and the address matches, an ACK is generated regardless of the set value.

Remark: n=0

Table 14-5: Format of IICA control register (IICCTLn0) (3/4)

<table>
<tr><td>STTn[Note1,2]</td><td colspan="2">Triggering of the start condition</td></tr>
<tr><td>0</td><td colspan="2">No start conditions are generated.</td></tr>
<tr><td>1</td><td colspan="2">When the bus is released (idle, IICBSYn bit is "0"): If this bit is "1", a start condition (boot as the master device) is generated. When a third party is communicating:
Enables communication reservation function (IICRSVn=0).
Used as a start condition reservation sign. If this bit set to "1", a start condition is automatically generated just after the bus is released.
When the communication reservation function is prohibited (IICRSVn=1).
Even if this bit is "1", the STTn bit is cleared and the STTn clear flag (STCFn) is set to "1" without generating a start condition. Wait status (master device):
Generates a restart condition after the wait is released.</td></tr>
<tr><td colspan="3">Notices on setting timing:
Master reception: Disables setting this bit to "1" during transmission. This bit can only be set to "1" during the waiting period when the ACKEn is "0" and notifying the slavet reception it has completed.
Master transmit: During the Ack, the start conditions may not be generated properly. This bit must be set to "1" during the wait period after the 9th clock is output.
It is prohibited to set "1" at the same time as the stop condition trigger (SPTn).
After setting the STTn bit to "1", it is prohibited to set this bit to "1" again before the clear condition is satisfied.</td></tr>
<tr><td colspan="2">Clear condition (STTn=0)</td><td>Set condition (STTn=1)</td></tr>
<tr><td colspan="2">Set the STTn bit to "1" in the state where communication reservation is disabled.
When arbitration fails
Master device generates start conditions.
Cleared due to the LRELn bit is "1" (Exit Communication).
When the IICEn bit is "0" (stop running).
When resetting</td><td>Set by intructions</td></tr>
</table>

Note 1: In the state where the IICEn bit is "0", the signal for this bit is invalid.

Note 2: The read value of the STTn bit is always "0".

Remark:

1. If bit1 (STTn) is read after setting the data, this bit becomes "0".
2. IICRSVn: Bit0 of IICA flag register n (IICFn)
3. STCFn: Bit7 of IICA flag register n (IICFn)
4. n=0

Table 14-5: Format of IICA control register (IICCTLn0) (4/4)

SPTn[Note1]	Trigger of stop condition
0	No stop conditions are generated.
1	Generates a stop condition (end of transfer as master).
Notices on setting timing: Master receive: Disables setting this bit to "1" during transmission. This bit can only be set to "1" during the waiting period when ACKEn is at "0" and notifying the slave reception has completed. Master transmit: During the Ack, the stop conditions may not be generated properly. This bit must be set to "1" during the wait period after the 9th clock is output. Prohibit setting "1" at the same time as the start condition trigger (STTn). When the WTIMn bit is "0", it must be noted that if the SPTn bit is set to "1" during the wait period after 8 clocks of output, a stop condition is generated during the high level of the 9th clock after the wait is released. The WTIMn bit must be set from "0" to "1" during the wait period after 8 clocks of output and the SPTn bit must be set to "1" during the wait period after the 9th clock of output. After setting the SPTn bit to "1", it is prohibited to set this bit to "1" again until the clear condition is satisfied.	
Clear condition (SPTn=0)	Set condition (SPTn=1)
When arbitration fails Cleared automatically when a stop condition is detected. Cleared due to the LRELn bit is "1" (exit communication). When the IICEn bit is "0" (stop running). When resetting	Set by intructions

Note 1: The read value of the SPTn bit is always "0".

Notice: If bit3 (TRCn) of the IICA status register n (IICSn) is "1" (transmit state), the wait is released by setting bit5 (WRELn) of the IICCTLn0 register to "1" on the 9th clock, the SDAAn line is set to high impedance after clearing the TRCn bit (receive state). The wait release must be performed by writing to the IICA shift register n when the TRCn bit is "1" (transmit state).

Remark: n=0

14.3.3 IICA status register n(IICSn)

This is a register that indicates the I^2C status.

The IICSn register can only be read by an 8-bit memory manipulation instruction during the STTn bit "1" and waiting. After a reset signal is generated, the value of this register becomes "00H".

Notice: In the allowed address matching wake function (WUPn=1) state in deep sleep mode, reading the IICSn register is prohibited. In the state where the WUPn bit is "1", it has nothing to do with the INTIICAn interrupt request if you change the WUPn bit from "1" to "0" (stop wake-up operation) reflects a change in state until the next start condition or stop condition is detected. Therefore, to use the wake-up function, it is necessary to allow (SPIEn=1) an interrupt due to the detection of a stop condition, and to read the IICSn register after the interrupt is detected.

Remark:

1. STTn: Bit1 of IICA control register n0 (IICCTLn0)
2. WUPn: Bit7 of IICA control register n1 (IICCTLn1)

Table 14-6: Format of IICA status register n (IICSn) (1/3)

Address: 0x40041B51 After reset: 00H R

Symbol	7	6	5	4	3	2	1	0
IICSn	MSTSn	ALDn	EXCn	COIn	TRCn	ACKDn	STDn	SPDn

MSTSn	Acknowledgement flag of master control status
0	Slave or communication idle status
1	Master communication status

Clear condition (MSTSn=0)	Set condition (MSTSn=1)
When a stop condition is detected When the ALDn bit is "1" (arbitration fails). Cleared due to the LRELn bit is "1" (Exit Communication). When the IICEn bit changes from "1" to "0" (stops running). When resetting	When generating a start condition

ALDn	Detection of arbitration failures
0	It indicates that arbitration did not occur or was won.
1	It indicates an arbitration failure. Clear the MSTSn bit.

Clear condition (ALDn=0)	Set condition (ALDn=1)
Automatically cleared after reading the IICSn register Note1. When the IICEn bit changes from "1" to "0" (stops running). When resetting	When arbitration fails

Note 1: This bit is cleared even if the bit memory manipulation instruction is executed on a bit other than the IICSn register. Therefore, when using ALDn bits, the data for the ALDn bits must be read before reading other bits.

Remark:

1. LRELn: Bit6 of IICA control register n0 (IICCTLn0)
2. IICEn: Bit7 of IICA control register n0 (IICCTLn0)
3. n=0

Table 14-6: Format of IICA status register n (IICSn) (2/3)

EXCn	Reception detection of extended codes
0	No extension code was received.
1	Extension code is received.

Clear condition (EXCn=0)	Set condition (EXCn=1)
When a start condition is detected When a stop condition is detected Cleared due to the LRELn bit is "1" (exits communication). When the IICEn bit changes from "1" to "0" (stops running). When resetting	When the high 4 bits of the received address data are "0000" or “1111”. (Set on the rising edge of the 8th clock).

COIn	Detection of address matching
0	The addresses are different.
1	The address is the same.

Clear condition (COIn=0)	Set condition (COIn=1)
When a start condition is detected When a stop condition is detected Cleared due to the LRELn bit is "1" (exits communication). When the IICEn bit changes from "1" to "0" (stops running). When resetting	When the receive address and the local station address (Slave Address Register n (SVAn)) are the same (set on the rising edge of the 8th clock)

TRCn	Transmit/receive status detection
0	In the receive state (except in the transmit state). Set the SDAAn line to high impedance.
1	In the transmitting state. Set to output the value of the SOn latch to the SDAAn line (valid after the falling edge of the 9th clock byte of the 1st byte).

Clear condition (TRCn=0)	Set condition (TRCn=1)
＜Master and slave＞	＜Master＞
When a stop condition is detected	When generating a start condition
Cleared due to the LRELn bit is "1" (exits communication).	When the LSB (transmission direction indicator bit) of byte 1 (address transmission)
When the IICEn bit changes from "1" to "0" (stops operation).	outputs "0" (master transmit).
Cleared because the WRELn bit is "1" (released from wait) Note 1	＜Slave＞
When the ALDn bit changes from "0" to "1" (arbitration fails)	When the LSB (transmission direction indication bit) in byte 1 (address transmission) of the master device
When resetting	inputs "1" (slave transmit)
When not participating in communications (MSTSn, EXCn, COIn=0)	
＜Master＞	
When "1" is output in the LSB (transmission direction indication bit) of the 1st byte	
＜Slave＞	
When a start condition is detected	
When "0" is input to the LSB (transmission direction indication bit) of the 1st byte	

Note 1: If bit3 (TRCn) of the IICA status register n (IICSn) is "1" (transmit state) and bit5 (WRELn) of the IICA control register n0 (IICCTLn0) is set to "1" on the 9th clock to release the wait, the SDAAn line is set to high impedance after clearing the TRCn bit (receive state). The wait release must be performed by writing the IICA shift register n when the TRCn bit is "1" (transmit state).

Remark:

1. LRELn: Bit6 of IICA control register n0 (IICCTLn0)
2. IICEn: Bit7 of IICA control register n0 (IICCTLn0)
3. n=0

Table 14-6: Format of IICA status register n (IICSn) (3/3)

ACKDn	Detection of acknowledge
0	No Ack was detected.
1	An Ack was detected.

Clear condition (ACKDn=0)	Set condition (ACKDn=1)
When resetting When a stop condition is detected When the next byte of the 1st clock rises Cleared because the LRELn bit is "1" (exits communication). When the IICEn bit changes from "1" to "0" (stops operation). When resetting	When the SDAAn line is set low on the 9th clock rising edge of the SCLAn line

STDn	Detection of starting conditions
0	No start condition detected.
1	A start condition is detected, indicating that it is during address transfer.

Clear condition (STDn=0)	Set condition (STDn=1)
When a stop condition is detected When the 1st clock rises after the next byte of the address is transmitted Cleared because the LRELn bit is "1" (exits communication). When the IICEn bit changes from "1" to "0" (stops operation). When resetting	When a start condition is detected

SPDn	Detection of stopping conditions
0	No stop condition detected.
1	A stop condition is detected, the master device ends communication and the bus is released.

Clear condition (SPDn=0)	Set condition (SPDn=1)
After setting this bit, the address transmits the byte after the start condition is detected when the clock rises When the WUPn bit changes from "1" to "0" When the IICEn bit changes from "1" to "0" (stops operation). When resetting	When a stop condition is detected

Remark:

1. LRELn: Bit6 of IICA control register n0 (IICCTLn0)
2. IICEn: Bit7 of IICA control register n0 (IICCTLn0)
3. n=0

14.3.4 IICA flag register n(IICFn)

This is a register that sets the I^2C operating mode and indicates the status of the I^2C-bus.

The IICFn register is set by an 8-bit memory manipulation instruction. However, only the STTn clear flag (STCFn) and the I^2C-bus status flag (IICBSYn) can be read.

The communication appointment function is allowed or disabled by the IICRSVn bit setting, and the initial value of the IICBSYn bit is set by the STCENn bit. Only bit7 (IICEn) = 0) can only write IICRSVn bits and STCENn bits. After allowing operation, only the IICFn registers can be read. After generating a reset signal, the value of this register changes to “00H”.

Table 14-7: Format of IICA flag register n(IICFn)

Address:	0x40041B52	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	Release start conditions.
1	The STTn flag cannot be cleared while the start condition cannot be issued.

Clear condition (STCFn=0)	Set condition (STCFn=1)
Cleared due to the STTn bit is "1" When the IICEn bit is "0" (stop operation). When resetting	If the STTn bit is cleared to "0" because the start condition cannot be issued in the state where communication reservation is disabled (IICRSVn=1).

IICBSYn	I^2C bus status flag
0	Bus release state (initial state of communication when STCENn=1)
1	Bus communication state (initial state of communication when STCENn=0)

Clear condition (IICBSYn=0)	Set condition (IICBSYn=1)
When a stop condition is detected When the ICEn bit is "0" (stop operation) When resetting	When a start condition is detected Sets the IICEn bit when the STCENn bit is "0"

STCENn	Initial start enable triggering
0	After enabling operation (IICEn=1), a start condition is allowed to be generated by detecting a stop condition.
1	After enabling operation (IICEn=1), the start condition is allowed to be generated without detecting a stop condition.

Clear condition (STCENn=0)	Set condition (STCENn=1)
Cleared by intructions. When a start condition is detected When resetting	Set by intructions

IICRSVn	Communication reservation function disable bit
0	Communication appointments are enabled.
1	Communication reservations are disabled.

Clear condition (IICRSVn=0)	Set condition (IICRSVn=1)
Cleared by intructions. When resetting	Set by intructions

Note: Bit6 and bit7 are read-only bits.

Notice:

1. The STCENn bit can only be written when the operation is stopped (IICEn=0).
2. If the STCENn bit is "1", the bus is considered to be released (IICBSYn=0) regardless of the actual bus state, so it is necessary to confirm that there is no third party in communication in order to avoid disrupting other communications when the first start condition (STTn=1) is issued.
3. IICRSVn can be written only when it is stopped (IICIn=0).

Remark:

1. STTn: Bit1 of IICA control register n0 (IICCTLn0)
2. IICEn: Bit7 of IICA control register n0 (IICCTLn0)

14.3.5 IICA control register n1(IICCTLn1)

This is a register used to set the I^2C operating mode and to detect the status of the SCLAn pin and SDAAn pin.

The IICCTLn1 register is set by an 8-bit memory manipulation instruction. However, only CLDn bits and DADn bits can be read.

With the exception of the WUPn bit, the IICCTLn1 register must be set when I^2C operation is disabled (bit7 (IICEn) = 0 of the IIC control register n0 (IICCTLn0)).

After a reset signal is generated, the value of this register becomes “00H”.

Table 14-8: Format of IICA control register n1 (IICCTLn1) (1/2)

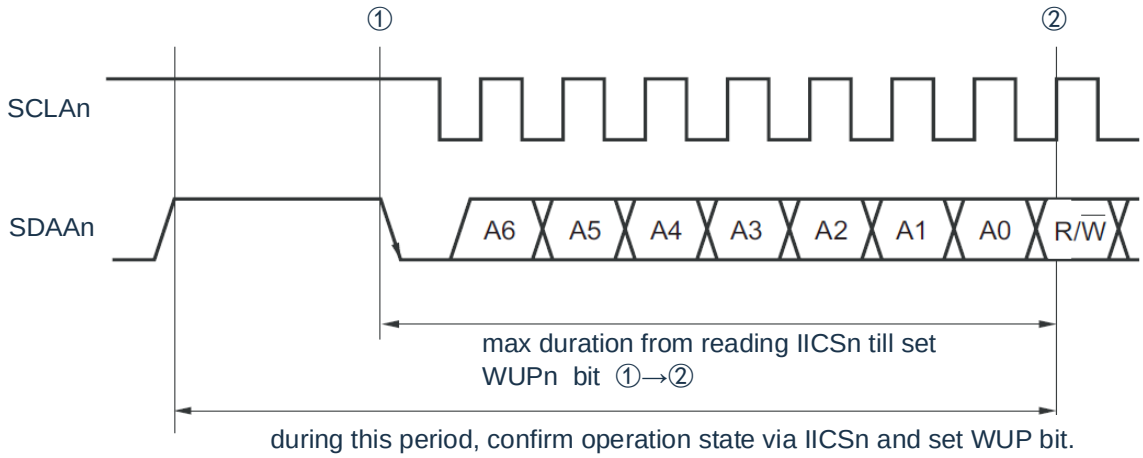
Address: 0x40041A31 After reset: 00H R/W[Note1]

Symbol	7	6	5	4	3	2	1	0
IICCTLn1	WUPn	0	CLDn	DADn	SMCn	DFCn	0	PRSn

WUPn	Control of address matching wakeup
0	Stops the operation of the Address Match Wakeup function in deep sleep mode.
1	Enables the operation of the Address Match Wakeup function in deep sleep mode.
To transfer to deep sleep mode by setting the WUPn bit to "1", at least three FMCK clocks must pass after setting the WUPn bit to "1", and then the deep sleep instruction must be executed (refer to "Figure 14-28 Flow when setting the WUPn bit to "1"). The WUPn bit must be cleared to "0" after the address is matched or the extension code is received. It is possible to participate in subsequent communication by clearing the WUPn bit to "0" (it is necessary to release the wait and write send data after clearing the WUPn bit to "0"). In the state of WUPn bit "1", the interrupt timing when the address is matched or the extension code is received is the same as the interrupt timing when WUPn bit is "0". (The delay difference of sampling error is generated according to the clock). In addition, when the WUPn bit is "1", even if the SPIEn bit is set to "1", no stop condition interrupt is generated.	
Clear condition (WUPn=0)	Set condition (WUPn=1)
Cleared by instructions (after an address match or reception of an extension code).	Set by instructions (MSTSn=0, EXCn=0, COIn=0 and STDn=0 (does not participate in communication)) [Note 2].

Note 1: Bit4 and bit5 are read-only bits.

Note 2: During the period shown below, it is necessary to check the status of the IICA status register n (IICSn) and set it.



Remark: n=0

Table 14-8: Format of IICA control register n1 (IICCTLn1) (2/2)

CLDn	Level detection of the SCLAn pin (valid only when the IICEn bit is "1").
0	SCLAn pin is detected low.
1	SCLAn pin is detected high.

Clear condition (CLDn=0)	Set condition (CLDn=1)
When the SCLAn pin is low When the IICEn bit is "0" (stops operation) When resetting	When the SCLAn pin is high

DADn	Level detection on the SDAAn pin (valid only when the IICEn bit is "1")
0	SDAAn pin is detected as low.
1	SDAAn pin is detected as high.

Clear condition (DADn=0)	Set condition (DADn=1)
When the SDAAn pin is low When the IICEn bit is "0" (stops operation). When resetting	When the SDAAn pin is high

SMCn	Switching of operating modes
0	Operates in standard mode (maximum transmission rate: 100kbps).
1	Operates in Fast Mode (maximum transfer rate: 400kbps) or Enhanced Fast Mode (maximum transfer rate: 1Mbps).

DFCn	Operation control of digital filters
0	Digital filter OFF
1	Digital filter ON
Digital filters must be used in fast mode or enhanced fast mode. Digital filters are used to eliminate noise. Whether the DFCn is "1" or "0", the transmission clock is unchanged.	

PRSn	Control of the operation clock (F_{MCK})
0	Selects F_{CLK} (4MHz≤F_{CLK}≤16MHz).
1	Selects F_{CLK}/2 (16MHz<F_{CLK}).

Notice:

1. The maximum operating frequency of the IICA running clock (F_{MCK}) is 16MHz (Max.). Bit0 (PRSn) of the IICA control register n1 (IICCTLn1) must be set to "1" only if the F_{CLK} exceeds 16MHz.
2. When setting the transmission clock, attention must be paid to the minimum operating frequency of F_{CLK}. The minimum operating frequency of the F_{CLK} of the serial interface IICA depends on the mode of operation.
   Fast mode: F_{CLK}=3.5MHz(Min.)
   Enhanced fast mode: F_{CLK}=10MHz(Min.)
   Standard mode: F_{CLK}=1MHz(Min.)

Remark:

1. IICEn: Bit7 of IICA control register n0 (IICCTLn0)
2. n=0

14.3.6 IICA low-level width setting register n(IICWLn)

This register controls the SCLAn pin signal low-level width (T_{LOW}) and SDAAn pin signal from the serial interface IICA output.

The IICWLn register is set by an 8-bit memory manipulation instruction.

The IICWLn register must be set when I^2C operation is disabled (bit 7 (IICEn) = 0 in IICA control register n0 (IICCTLn0)). The value of this register changes to "FFH" after the reset signal is generated.

For how to set the IICWLn register, refer to “14.4.2 Setting transfer clock via IICWLn and IICWHn registers”.

The data retention time is 1/4 of the time set by IICWLn.

Table 14-9 Format of IICA low-level width setting register n (IICWLn)

Address:	0x40041A32	After reset:	FFH				R/W	
Symbol	7	6	5	4	3	2	1	0
IICWLn	1	1	1	1	1	1	1	1

14.3.7 IICA high level width setting register n(IICWHn)

This register controls the high-level width of the SCLAn pin signal and the SDAAn pin signal of the serial interface IICA output. The IICWHn register is set by an 8-bit memory manipulation instruction.

The IICWHn register must be set when I^2C operation is disabled (bit7(IICEn)=0 of IICA control register n0(IICCTLn0)). After a reset signal is generated, the value of this register becomes “FFH”.

Table 14-10 Format of high level width setting register n(IICWHn)

Address:	0x40041A33	After reset:	FFH				R/W	
Symbol	7	6	5	4	3	2	1	0
IICWHn	1	1	1	1	1	1	1	1

Note 1: For the method of setting the clock transmitted by the main controller, please refer to 14.4.2(1); for how to set the slave IICWLn register and the IICWHn register, refer to 14.4.2(2).

Note 2: n=0

14.3.8 Registers controlling port functions of IICA pins

This product can multiplex the pin function of IICAn to multiple ports.

The SCALn pin and SDAAn pin can be configured to the port separately by setting the port multiplexing function configuration registers (SCLAnPCFG and SDAAnPCFG). (n=0)

Set the bits of the Port Mode Control Register (PMCxx) and the Port Mode Register (PMxx) corresponding to these two ports to "0".

When these two ports are configured for multiplexing of the IICA pins, the N-channel open drain output (V_{DD}/EV_{DD} withstand) mode of the ports is guaranteed by design to turn on automatically, i.e. the POMxx register does not need to be set by the user.

For detailed setting method, see “Chapter 2 Port Function”.

14.4 Function of I²C-bus mode

14.4.1 Pin structure

The serial clock pin (SCLAn) and serial data bus pin (SDAAn) are configured as follows.

(1) SCLAn: input/output pins of the serial clock

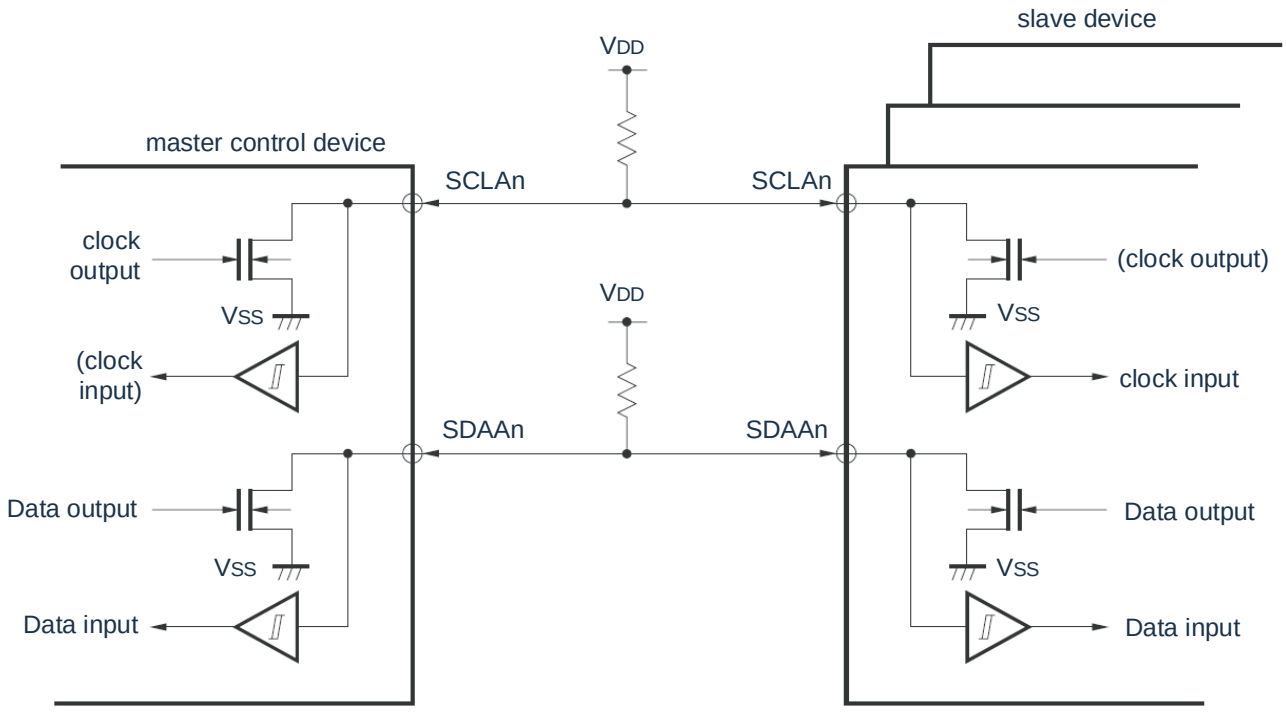
The outputs of both the master and slave devices are N-channel open-drain outputs, and the inputs are Schmidt inputs.

(2) SDAAn: input/output multiplexing pins for serial data

The outputs of both the master and slave devices are N-channel open-drain outputs, and the inputs are Schmidt inputs.

Because the outputs of the serial clock line and serial data bus are N-channel open-drain outputs, an external pull-up resistor is required.

Figure 14-3: Block diagram of pin structure



Remark: n=0

14.4.2 Setting transfer clock via IICWLn and IICWHn registers

(1) Setting transfer clock on master side

$$\text{Transfer clock} = \frac{F_{MCK}}{IICWL + IICWH + F_{MCK}(T_R+T_F)}$$

At this point, the optimal setpoints for the IICWLn register and the IICWHn register are as follows:

(The fractional parts of all setting values are rounded up.)

① Fast mode

$IICWLn=\frac{0.52}{\text{transfer clock}} \times F_{MCK}$

$IICWHn=(\frac{0.48}{\text{transfer clock}} - T_R - T_F) \times F_{MCK}$

② Standard mode

$IICWLn=\frac{0.47}{\text{transfer clock}} \times F_{MCK}$

$IICWHn=(\frac{0.53}{\text{transfer clock}} - T_R - T_F) \times F_{MCK}$

③ Enhanced fast mode

$IICWLn=\frac{0.50}{\text{transfer clock}} \times F_{MCK}$

$IICWHn=(\frac{0.50}{\text{transfer clock}} - T_R - T_F) \times F_{MCK}$

(2) Setting IICWLn and IICWHn registers on slave side

(The fractional parts of all setting values are truncated.)

① Fast mode

$IICWLn=1.3us \times F_{MCK}$

$IICWHn=(1.2us - T_R - T_F) \times F_{MCK}$

② Standard mode

$IICWLn=4.7us \times F_{MCK}$

$IICWHn=(5.3us - T_R - T_F) \times F_{MCK}$

③ Enhanced fast mode

$IICWLn=0.50us \times F_{MCK}$

$IICWHn=(0.50us - T_R - T_F) \times F_{MCK}$

Notice:

1. The maximum operating frequency of the IICA operating clock (FMCK) is 16MHz (Max.). The bit0 (PRSn) of the IICA control register n1 (IICCTLn1) must be set to "1" only when the F_{CLK} exceeds 16MHz.
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}=3.5MHz(Min.)
   Enhanced fast mode: F_{CLK}=10MHz(Min.)
   Standard mode: F_{CLK}=1MHz(Min.)
3. 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.

Remark:

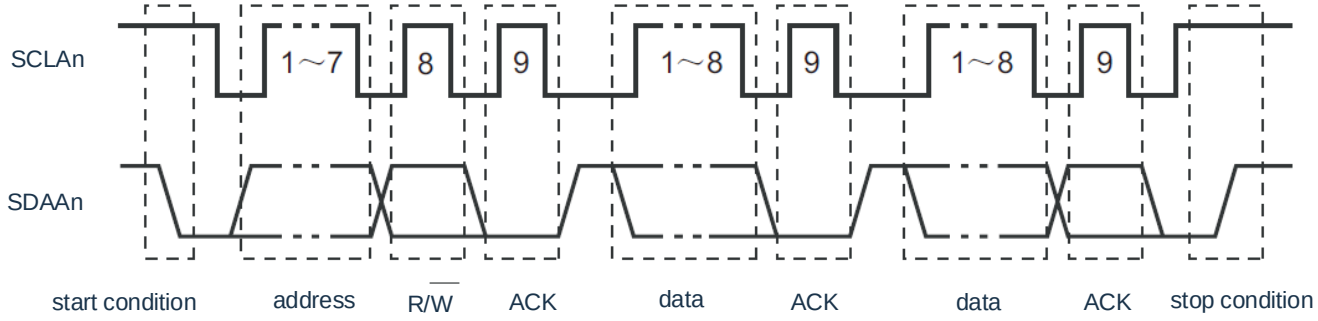
1. IICWLn: IICA low-level width setting register n; IICWHn: IICA high-level width setting register n
   T_F: SDAAn and SCLAn signal falling times; T_R: SDAAn and SCLAn signal rising times
   F_{MCK}: IICA operation clock frequency;
2. n=0

14.5 Definition and control method of I²C-bus

The following section describes the I²C bus's serial data communication format and the signals used by the I²C bus.

The figure below shows the transfer timing for the "start condition", "address", "data", and "stop condition" output via the I²C bus's serial data bus.

Figure 14-4: I²C bus serial data transfer timing



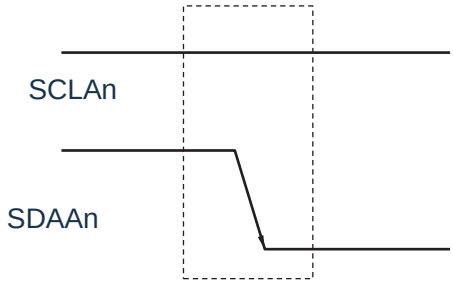
The master generates start conditions, slave addresses, and stop conditions.

Both the master and slave devices can generate a reply (ACK) (in general, the receiver outputs 8 bits of data). The master device continuously outputs a serial clock (SCLAn). However, the slave can extend the low level of the SCLAn pin during and insert a wait.

14.5.1 Start condition

When the SCLAn pin is high, a start condition is generated if the SDAAn pin changes from high to low. The starting conditions for the SCLAn pin and the SDAAn pin are the signals generated when the master device starts serially transmitting to the slave. When used as a slave, the start condition is detected.

Figure 14-5: Start condition



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

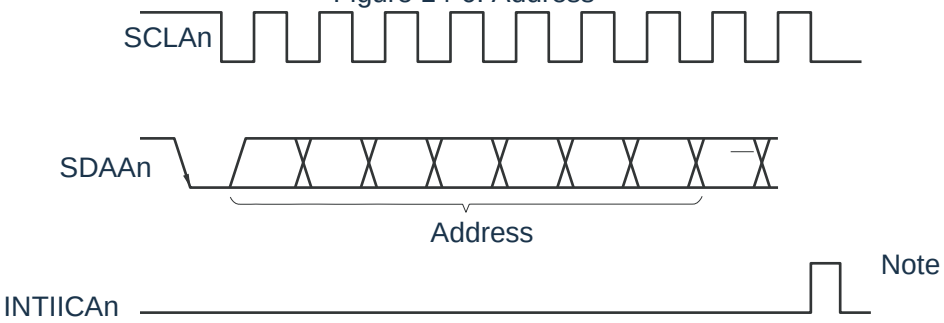
14.5.2 Address

The next 7 bits of data for the start condition are defined as addresses.

The address is 7 bits of data output by the master device in order to select a particular slave device from a plurality of slave devices connected to the bus. Therefore, the slave devices on the bus need to be set to completely different addresses.

The slave detects the start condition through the hardware and checks whether the 7-bit data is the same as the contents of the slave address register n(SVAn). At this time, if the 7-bit data and the value of the SVAn register are the same, the slave is selected to communicate with the master device before the master generates a start or stop condition.

Figure 14-6: Address



Note: If data other than the local station address or extension code is received while the slave is running, INTIICAn is not generated.

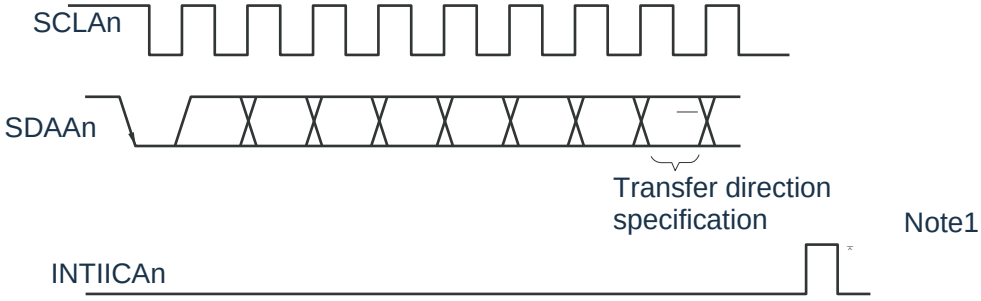
If the 8-bit data consisting of the slave address and the transfer direction described in “14.5.3 Transfer direction specification” is written to the IICA shift register n (IICAn), the address is output. The received address is written to the IICAn register. The slave address is assigned to the higher 7 bits of the IICAn register.

14.5.3 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 14-7: Transfer direction specification



Note 1: INTIICAn is not issued if data other than a local address or extension code is received during slave device operation.

Remark: n=0

14.5.4 Acknowledge (ACK)

The serial data status of the sender and receiver can be acknowledged by answer (ACK). The receiver returns a reply each time it receives 8 bits of data.

Typically, the sender receives a reply after sending 8 bits of data. When the receiver returns the reply, it is deemed to have been received normally and continues processing. Bit2 (ACKDn) can pass through the IICA status register n (IICSn). Confirm the detection of the Ack. When the master receives the last data for the received state, a stop condition is generated without returning a reply. When the slave does not return a reply after receiving the data, the master device outputs a stop condition or a restart condition to stop the transmission. The reasons why a reply is not returned 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.

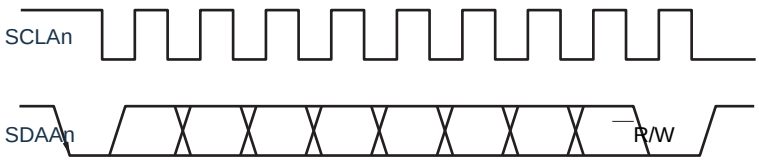
To generate ACK, the reception side makes the SDAAn line low at the ninth clock (indicating normal reception).

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 14-8: 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 clock stretch timing.

(1) When 8-clock clock stretch state is selected (bit 3 (WTIMn) of IICCTLn0 register = 0): By setting the ACKEn bit to 1 before releasing the clock stretch state, ACK is generated at the falling edge of the eighth clock of the SCLAn pin.

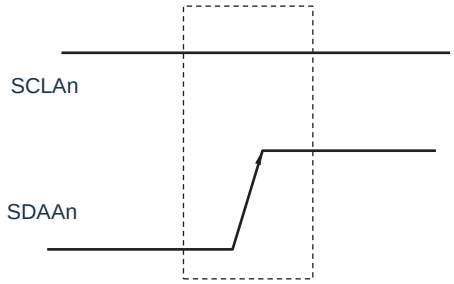
(2) When 9-clock clock stretch state is selected (bit 3 (WTIMn) of IICCTLn0 register = 1): ACK is generated by setting the ACKEn bit to 1 in advance.

Remark: n=0

14.5.5 Stop condition

When the SCLAn pin is high, a stop condition is generated if the SDAAn pin changes from low to high. The stop condition is the signal generated when the master ends serial transmission to the slave. When used as a slave, a stop condition is detected.

Figure 14-9: Stop condition



If the bit0 (SPTn) of the IICA control register n0 (IICCTLn0) is set to "1", a stop condition is generated. If a stop condition is detected, set bit0 (SPDn) of the IICA status register n (IICSn) to "1" and generates INTIICAn when bit4 (SPIEn) of the IICCTLn0 register is "1".

Remark: n=0

14.5.6 Wait

Notify the other master or slave that the other master or slave is preparing to send/receive data by waiting (waiting status).

Notify the other party that it is in a waiting state by setting the SCLAn pin low. If both the master and slave wait states are released, the next transfer can begin.

Figure 14-10: 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)

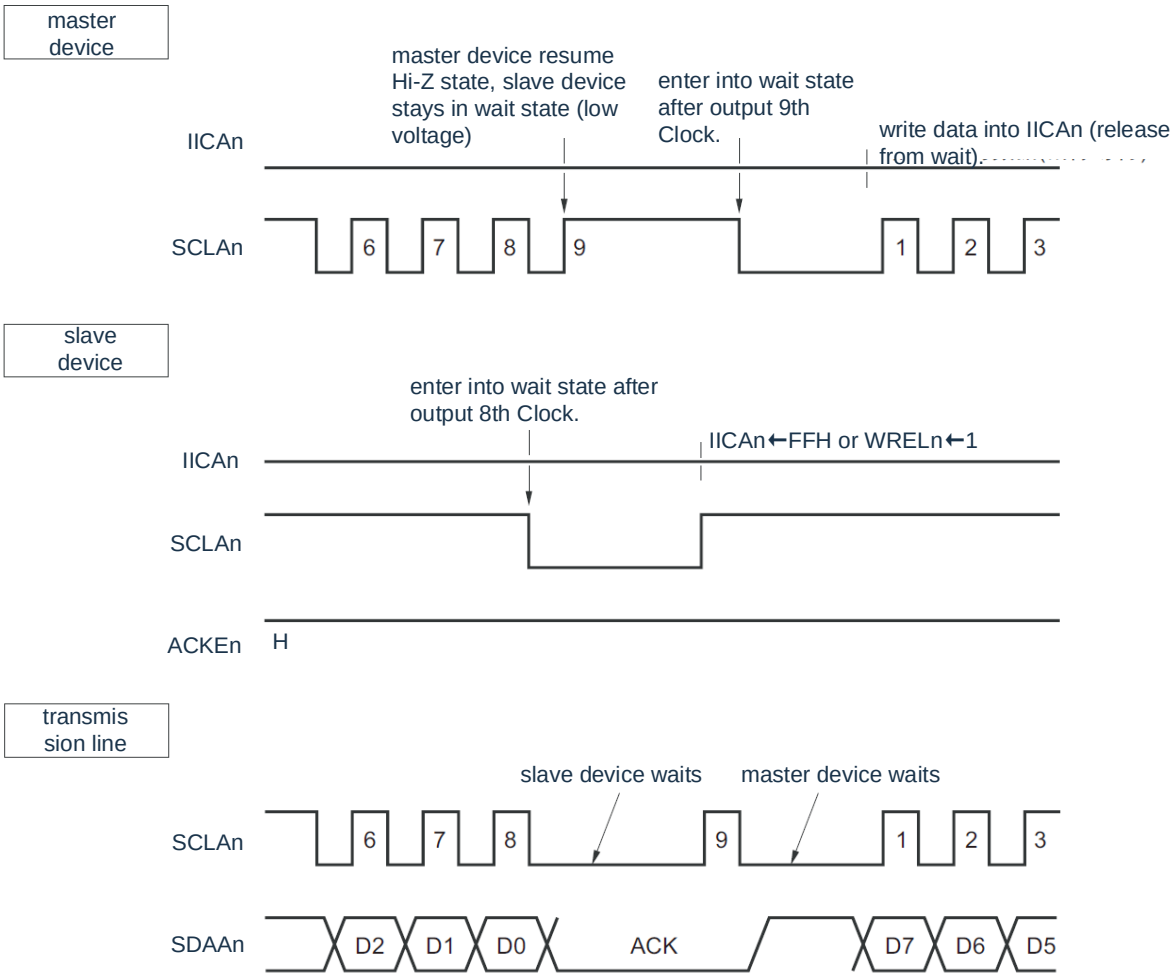
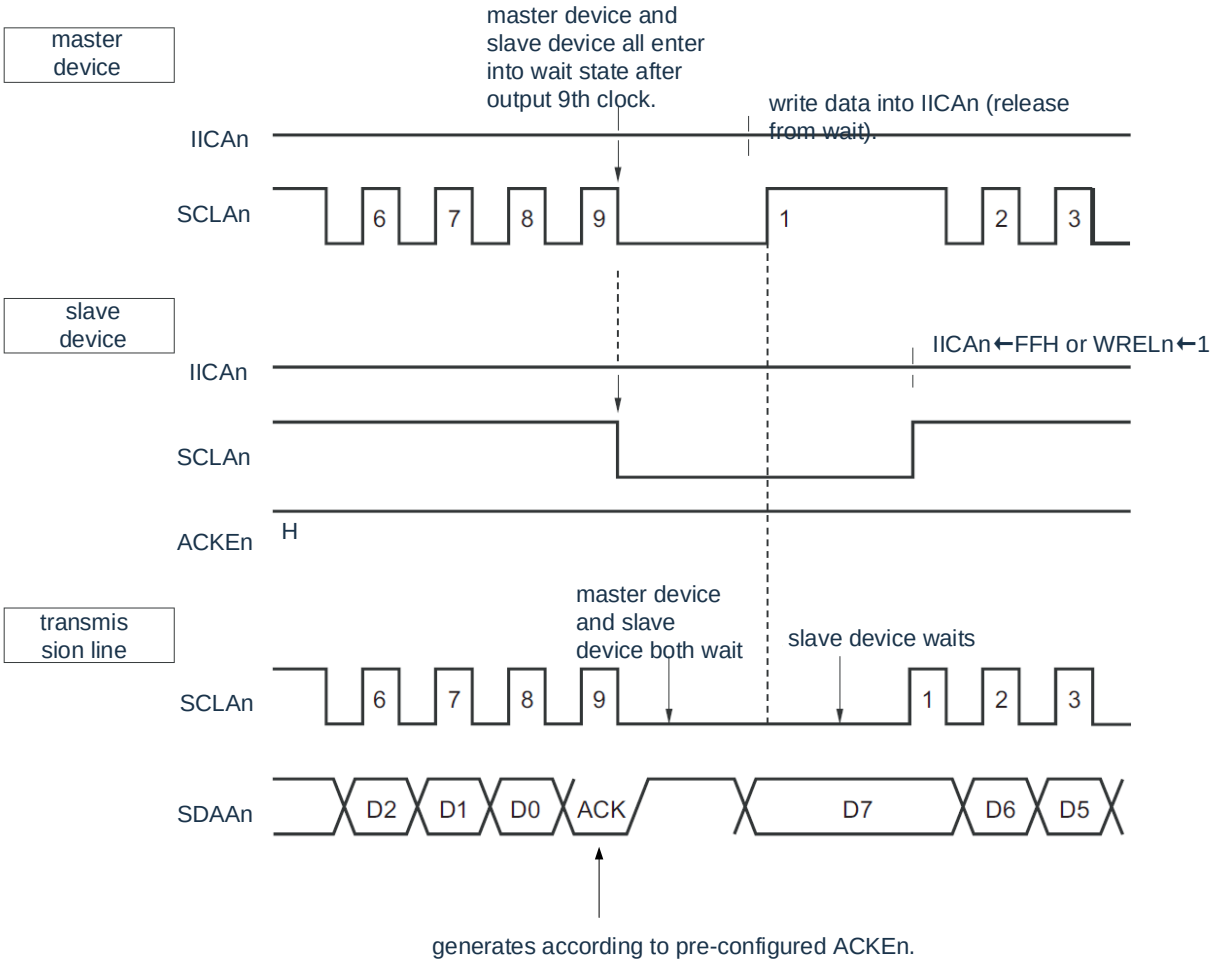


Figure 14-11: Wait (2/2)

(2) When both the master and slave devices are waiting for 9 clocks

(master transmits, slave receives, and ACKEn = 1)



Remark: ACKEn: Bit2 of IICA control register n0 (IICCTLn0)

WRELn: Bit5 of the IICA control register n0 (IICCTLn0)

The wait state is generated automatically by setting bit 3 (WTIMn) of the IICA control register n0 (IICCTLn0). Normally, on the receiving side, if bit5(WRELn) of IICCTLn0 register is "1" or if "FFH" is written to IICA shift register n(IICAn), the wait is released; on the transmitting side, if data is written to IICAn register, the wait is released. On the transmitting side, if data is written to the IICAn register, the wait is released. The master device can also release the wait by the following methods:

Set bit1 (STTn) of the IICCTLn0 register to "1"

Set bit0 (SPTn) of the IICCTLn0 register to "1"

Remark: n=0

14.5.7 Method of releasing wait state

In general, I²C can release the wait with the following processing.

(1) Writes data to IICA shift register n (IICAn).

(2) Sets the bit5 (WRELn) of the IICA control register n0 (IICCTLn0) (wait released).

(3) Sets the bit1 (STTn) of the IICCTLn0 register (generates a start condition) Note.

(4) Sets the bit0 (SPTn) of the IICCTLn0 register (generates a stop condition) Note.

Note: Limited to master devices.

If these wait release processes are performed, the I²C releases the wait and starts communication again. To send data (including the address) after the release wait, data must be written to the IICAn register.

To receive data after release from waiting or to end sending data, bit 5 (WRELn) of the IICCTLn0 register must be set to "1". To generate a restart condition after releasing the wait, bit 1 (STTn) of the IICCTLn0 register must be set to "1". To generate a stop condition after releasing a wait, bit 0 (SPTn) of the IICCTLn0 register must be set to "1". Only one release process can be performed for one wait.

For example, if data is written to the IICAn register after the wait is released by setting the WRELn bit to "1", the change timing of the SDAAn line may conflict with the write timing of the IICAn register, resulting in the wrong value being output to the SDAAn line. In addition to these processes, if the IICEn bit is cleared to "0" in the case of stopping communication in the middle of the communication, communication is stopped, so that waiting can be released. If the I2C bus state is deadlocked due to noise, if bit 6 (LRELn) of the IICCTLn0 register is set to "1", communication is exited, and thus waiting is released.

Remark:

1. If the wait release process is performed when the WUPn bit is "1", the wait will not be released.
2. n=0

14.5.8 Generation timing and waiting control of interrupt requests (INTIICAn)

By setting bit3 (WTIMn) of the IICA control register n0 (IICCTLn0), INTIICAn is generated at the timing shown inTable 14-11 and wait control is performed.

Table 14-11: Generation timing and waiting control of INTIICAn

WTIMn	Slave operation			Master operation		
	Address	Data reception	Data transmission	Address	Data reception	Data transmission
0	9Note1,2	8Note2	8Note2	9	8	8
1	9Note1,2	9Note2	9Note2	9	9	9

Note 1: Only when the received address and the set address of the slave address register n(SVAn) are the same, the slave generates an INDICATIONAn signal on the falling edge of the 9th clock and enters a waiting state. At this point, regardless of the bit2 (ACKEn) setting of the IICCTLn0 register, a reply is generated. The slave that receives the extension code generates INTIICAn on the descending edge of the 8th clock. If the addresses are different after restarting, INTIICAn is generated on the falling edge of the 9th clock, but does not enter the waiting state.

Note 2: If the contents of the received address and the slave address register n(SVAn) are different and the extension code is not received, THE INTIICAn is not generated and does not enter the waiting state.

Remark: The numbers in the table represent the number of clocks for a serial clock. Both interrupt request and wait control are synchronized with the falling edge of the serial clock.

(1) Address transmission and reception

① Slave operation: Regardless of the WTIMn bit, the timing of interruptions and waits is determined according to the conditions in Notes 1 and 2 above.

② Master operation: Regardless of the WTIMn bit, the timing of interrupts and waits is generated on the falling edge of the 9th clock.

(2) Data reception

Master/slave operation: Determines the timing of interrupts and waits via the WTIMn bit.

(3) Data transmission

Master/slave operation: Determines the timing of interrupts and waits via the WTIMn bit.

Remark: n=0

(4) Release method of waiting

There are 4 ways to release from waiting:

① Writes data to IICA shift register n (IICAn).

② Sets the bit5 (WRELn) of the IICA control register n0 (IICCTLn0) (wait released).

③ Sets the bit1 (STTn) of the IICCTLn0 register (generates a start condition) Note.

④ Sets the bit0 (SPTn) of the IICCTLn0 register (generates a stop condition) Note.

Note: Limited to master devices.

When you select a wait for 8 clocks (WTIMn=0), you need to decide whether to generate a reply before you release the wait.

(5) Detection of stop condition

If a stop condition is detected, INTIICAn is generated (limited to when SPIEn=1).

14.5.9 Detection method for address matching

In I²C-bus mode, the master device can select a specific slave by sending a slave address. Address matching can be automatically detected by hardware. When the slave address sent by the master device and the set address of the slave address register n(SVAn) are the same or only the extension code is received, an INTIICAn interrupt request is generated.

14.5.10 Error detection

In I²C-bus mode, because the status of the serial data bus (SDAAn) during the transmission process is taken to the IICA shift register n (IICAn) of the transmitting device, Therefore, it is possible to detect send errors by comparing the IICA data before and after the start of sending. At this point, if the two data are different, it is judged that a sending error has occurred.

Remark: n=0

14.5.11 Extension code

(1) When the high 4 bits of the receiving address are "0000" or "1111", as the received extension code, the extended code receive flag (EXCn) is set to "1", and in the 8th The falling edge of the clock generates an interrupt request (INTIICAn).
Does not affect local station addresses stored in slave address register n (SVAn).

(2) When the SVAn register is set to "11110xx0", if "11110xx0" is sent from the master device via a 10-bit address, the following assertion occurs. However, an interrupt request (INTIICAn) is generated on the falling edge of the 8th clock.
① High 4 bits data are the same: EXCn=1
② 7 bits of data are the same: COIn=1

Remark:
1. EXCn: Bit5 of the IICA status register n
2. COIn: Bit4 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 14-12: Bit definitions of major extension codes

Slave address	R/W bit	Definition
0000000	0	Full call address
11110xx	0	10-bit slave address specification (during address authentication)
11110xx	1	10-bit slave address specification (after address match, when read command is issued)

Remark:
1. See the I^2C bus specifications issued by NXP Semiconductors for details of extension codes other than those described above.
2. n=0

14.5.12 Arbitration

When multiple master devices generate start conditions at the same time (Set STTn bit to "1" before the STDn bit becomes "1"), the communication of the master device is carried out while adjusting the clock until the data is different. This run is called quorum.

When the arbitration fails, the master device that fails the arbitration places the arbitration failure flag (ALDn) of the IICA status register n (IICSn) to "1" and places the SCLAn Both the line and the SDAAn line are placed in a high impedance state, releasing the bus.

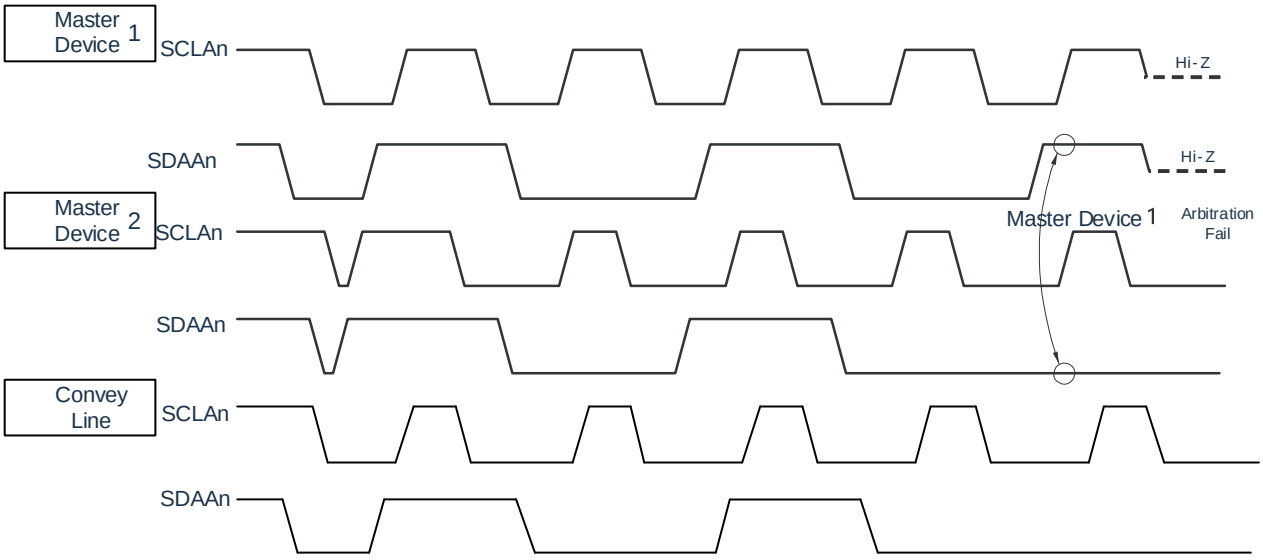
In the event of the next interrupt request (e.g., a stop condition is detected at the 8th or 9th clock), the ALDn bit is "1" via software to detect the failure of the quorum.

For the timing of interrupt requests, please refer to “16.5.8 Generation timing and waiting control of interrupt requests (INTIICAn)”.

Remark:

1. STDn: Bit1 of IICA status register n (IICSn)
2. STTn: Bit1 of IICA control register n0 (IICCTLn0)

Figure 14-12: Timing example of arbitration



Remark: n=0

Table 14-13: Status at the time of arbitration and timing of generation of interrupt requests

The state in which the arbitration occurred	Timing of the generation of interrupt requests
During address transmission	On the falling edge of the 8th or 9th clock after the byte is transmitted Note1
Reads and writes information after address transmission	
During address extension codes transmission	
Reads and writes information after extension codes transmission	
During data transmission	
During ACK transmission after transmitted data	
Restart condition detected during data transmission	
Stop condition detected during data transmission	When generating a stop condition (SPIEn=1) Note2
Trying to generate a restart condition, but the data is low.	On the falling edge of the 8th or 9th clock after the byte is transmitted Note1
Trying to generate a restart condition, but a stop condition was detected.	When generating a stop condition (SPIEn=1) Note2
Trying to generate a stop condition, but the data is low.	On the falling edge of the 8th or 9th clock after the byte is transmitted Note1
Trying to generate a restart condition, but SCLAn is low.	

Note 1: The interrupt request is generated on the falling edge of the 9th clock when the WTIMn bit (bit3 of the IICA control register n0 (IICCTLn0)) is "1", and on the falling edge of the 8th clock when the WTIMn bit is "0" and the slave address of the extended code is received.

Note 2: When there is a possibility of arbitration, the SPIEn bit must be "1" when the master is operating.

Remark:

1. SPIEn: Bit4 of IICA control register n0 (IICTLn0)
2. n=0

14.5.13 Wake-up function

This is a slave function of I2C, which is the function of generating an interrupt request signal (INTIICAn) when the local station address and extension code are received. The processing efficiency is improved by not generating unwanted INTIICAn signals under different addresses. If a start condition is detected, it enters wake-up standby. Because the master device (where a start condition has already been generated) may also become a slave due to an arbitration failure, it enters wake-up standby at the same time as the address is sent.

To use the wake function in deep sleep mode, you must place the WUPn at "1". The address can be received independent of the operating clock. Even in this case, an interrupt request signal (INTIICAn) is generated when the local station address and extension code are received. After this interrupt is generated, the WUPn bit is cleared to "0" by the instruction and returned to the normal operation.

The flow when the WUPn bit is set to "1" is shown in Figure 14-13, and the flow when the WUPn bit is set to "0" by address matching is shown in Figure 14-14.

Figure 14-13: Flow when the WUPn bit is set to "1"

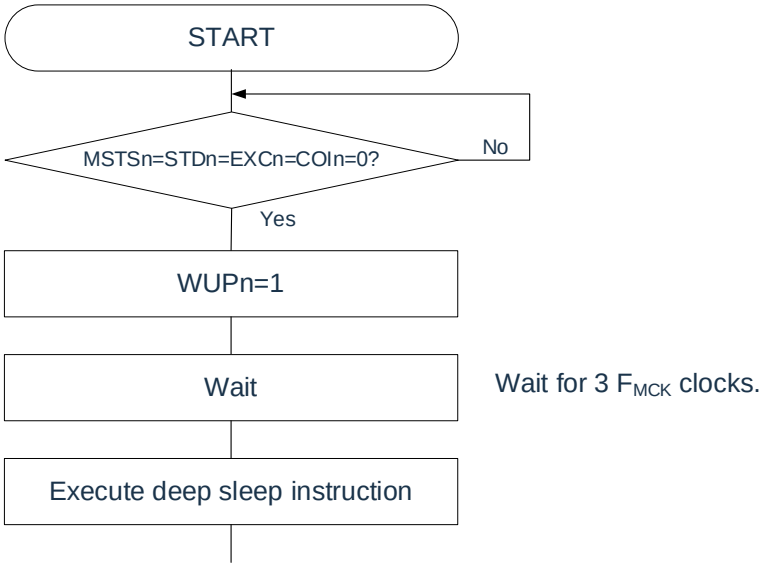
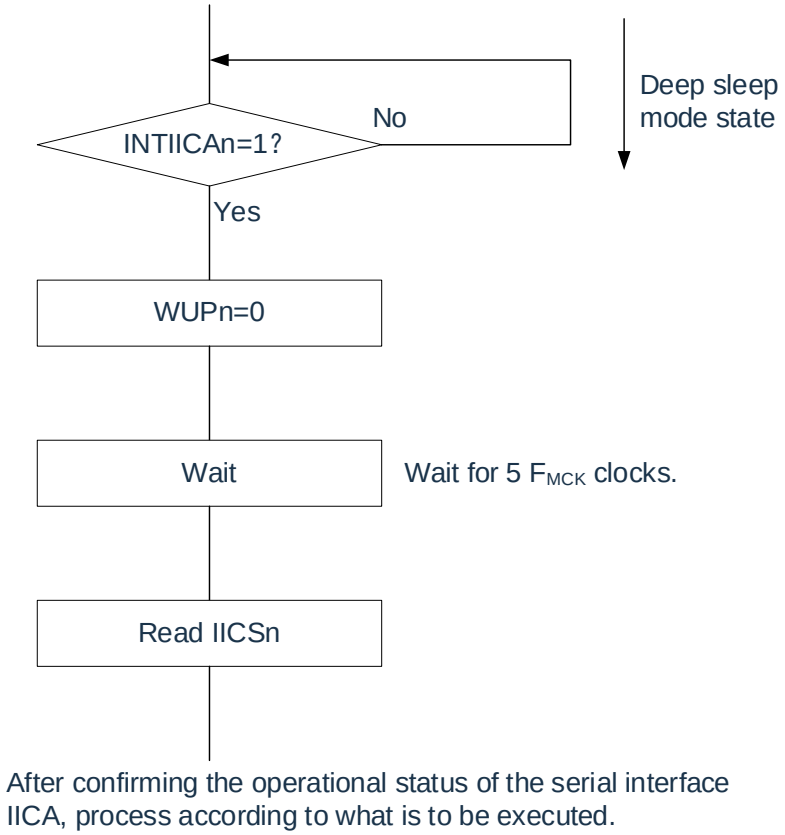


Figure 14-14: Flow when the WUPn bit is set to "0" by address matching (including receiving extension codes)



In addition to the interrupt request (INTIICAn) generated by the serial interface IICA, the deep sleep mode must be removed through the following procedure.

(1) Next IIC communication for the operation of the master device: the flow of Fig. 14-15

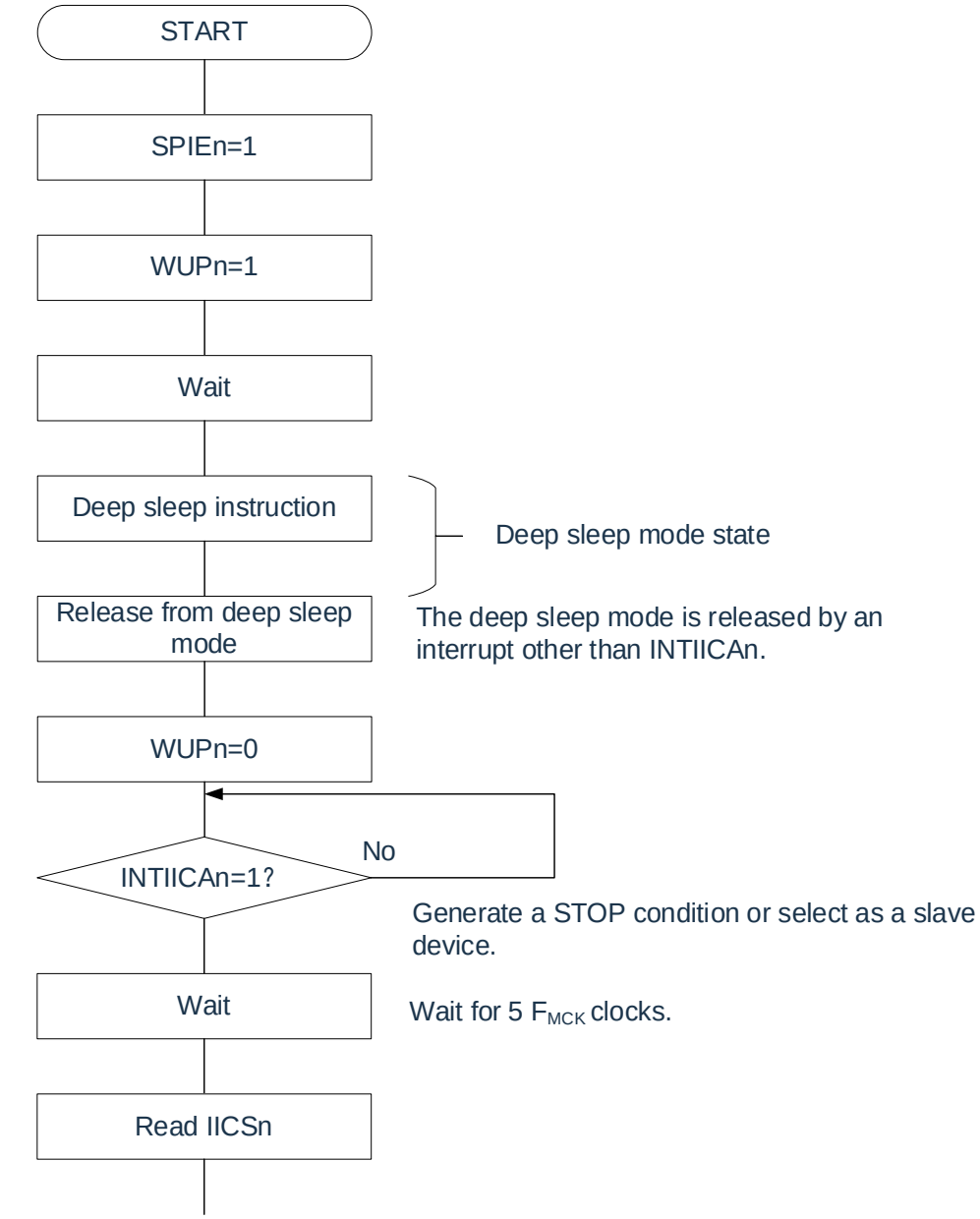
(2) Next IIC communication for slave device operation:

Return via INTIICAn interrupt: same process as Figure 14-14.

Return from an interrupt other than the INTIICAn interrupt: operation must be continued with the WUPn bit set to "1" before the INTIICAn interrupt is generated.

Remark: n=0

Figure 14-15: Operation as master device after being released from deep sleep mode by an interrupt other than INTIICAn



Remark: n=0

14.5.14 Communication reservation

(1) When communication reservation function is enabled (bit0 (IICRSVn)=0 of the IIC flag register n (IICFn))

To perform the next master communication without using the bus, you can send a start condition when the bus is released through a communication reservation. There are two modes under which the bus is not used.

① When arbitration results in neither master nor slave operation

② 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.

① Generate a start condition when the bus is released.

② Communication reservation when the bus is not released (standby state)

After the STTn bit is set to "1" and a wait time has elapsed, operation as a communication reservation is confirmed by the MSTSn bit (bit 7 of the IICA status register n (IICSn)).

Waiting times must be ensured by software for the following formula calculations.

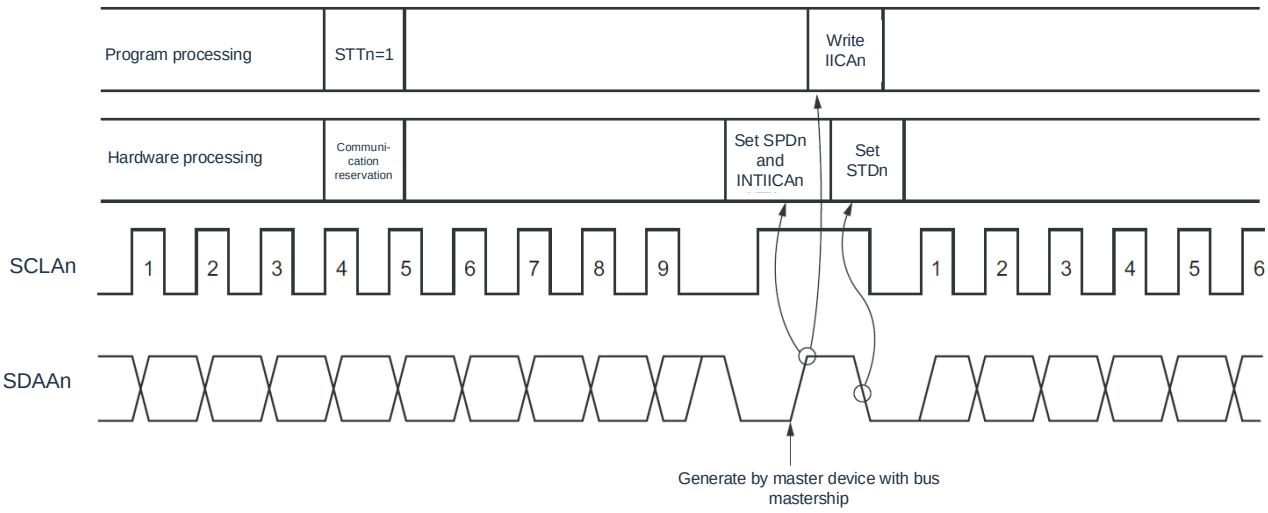
Wait time from setting STTn bit to "1" until the MSTSn flag is confirmed: (IICWLn set value+ IICWHn set value + 4) / F_{MCK} + $T_F \times 2$

Remark:

1. IICWLn: IICA low-level width setting register n
   IICWHn: IICA high-level width setting register n
   T_F: SDAAn and SCLAn signal falling times
   F_{MCK}: IICA operation clock frequency
2. n=0

The timing of the communication reservation is shown in the following diagram.

Figure 14-16: Timing of communication reservation



Remark: IICAn: IICA shift register n

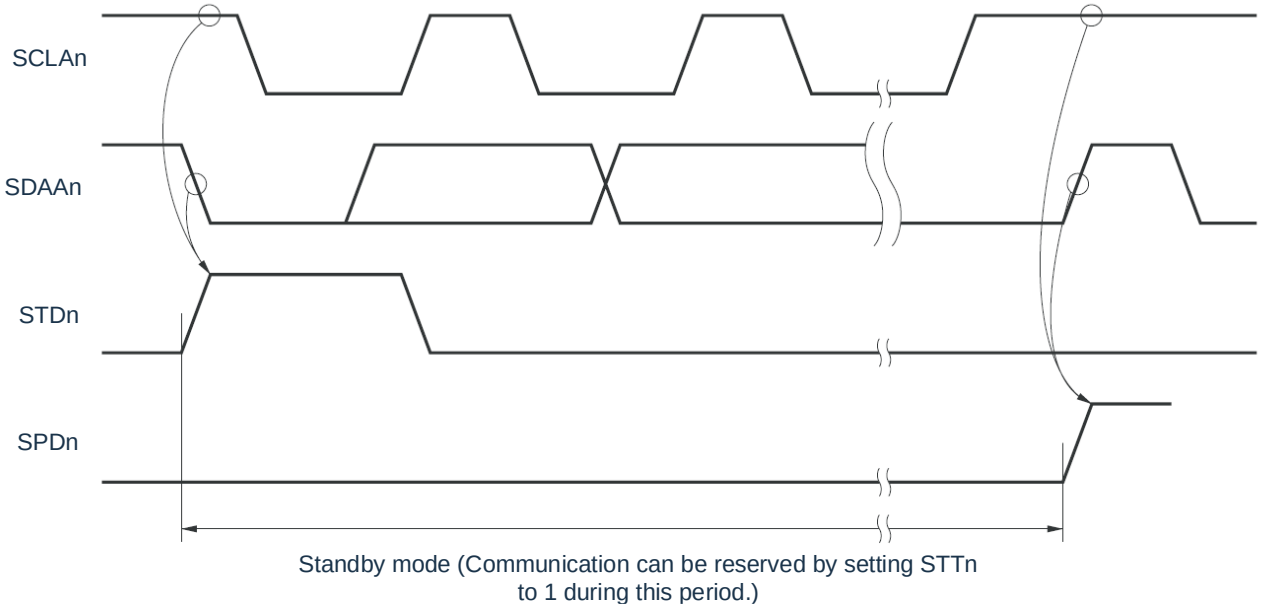
STTn: Bit1 of IICA control register n0 (IICCTLn0)

STDn: Bit1 of IICA status register n (IICSn)

SPDn: Bit0 of IICA status register n (IICSn)

The communication reservation is accepted by the timing sequence shown in Figure 14-17. After bit1 (STDn) of the IICA status register n (IICSn) becomes "1" and before a stop condition is detected, bit1 (STTn) of the IICA control register n0 (IICCTLn0) is set to "1 " to make a communication reservation.

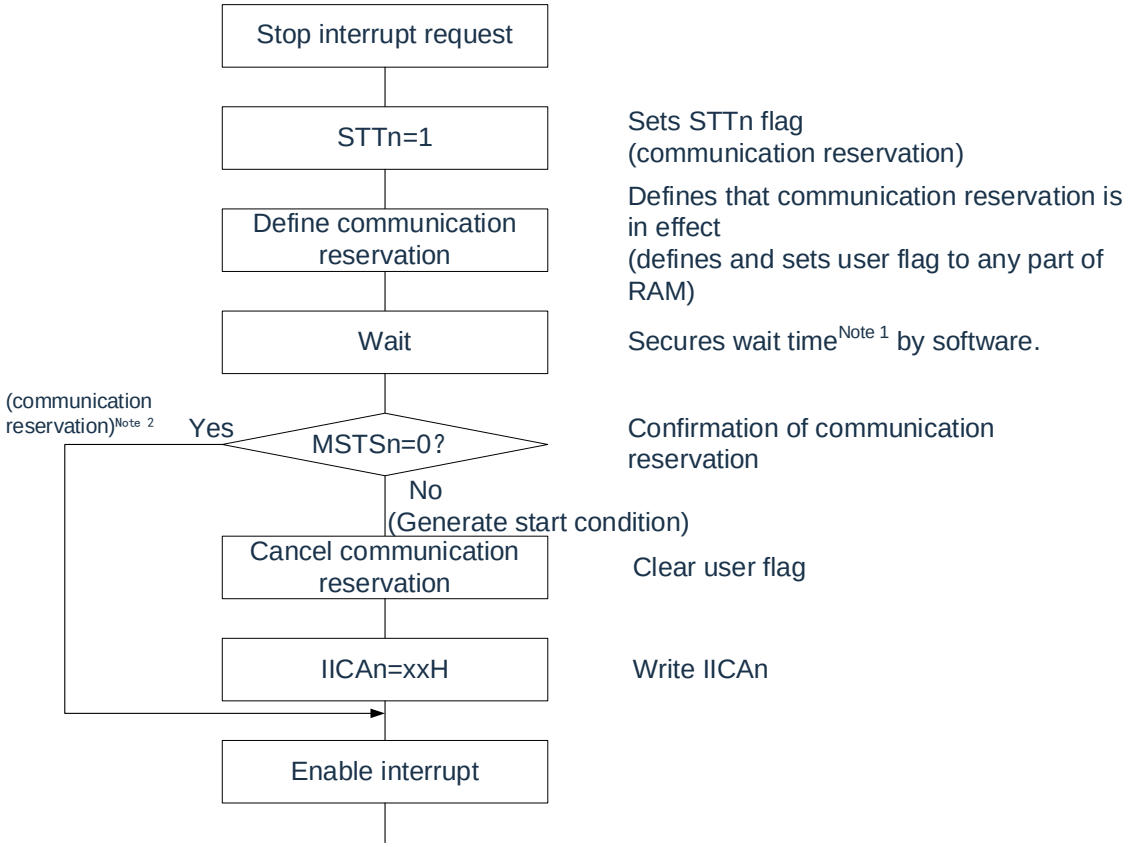
Figure 14-17: Timing for accepting communication reservations



Remark: n=0

The steps for communication reservations are shown in Figure 14-18.

Figure 14-18: Steps for communication reservations



Note 1: The wait time is calculated as follows. (IICWLn set value+IICWHn set value +4)/F_{MCK}+T_F×2

Note 2: The communication reservation operation executes a write to the IICA shift register n (IICAn) when a stop condition interrupt request occurs.

Remark:

1. STTn: Bit1 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.

① When arbitration results in neither master nor slave operation

② 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 cycles of F_{MCK} until the STCFn bit is set to 1 after setting STTn = 1. Therefore, secure the time by software.

Remark: n=0

14.5.15 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 I2C 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 cycles of F_{MCK} after setting the IICEn bit to 1), to forcibly disable detection.

(4) Setting the STTn and SPTn bits (bits 1 and 0 of the IICCTLn0 register) again after they are set and before they are cleared to 0 is prohibited.

(5) When transmission is reserved, set the SPIEn bit (bit 4 of the IICCTLn0 register) to 1 so that an interrupt request is generated when the stop condition is detected. Transfer is started when communication data is written to the IICA shift register n (IICAn) after the interrupt request is generated. Unless the interrupt is generated when the stop condition is detected, the device stops in the wait state because the interrupt request is not generated when communication is started. However, it is not necessary to set the SPIEn bit to 1 when the MSTSn bit (bit 7 of the IICA status register n (IICSn)) is detected by software.

Remark: n=0

14.5.16 Communication operation

The following shows three operation procedures with the flowchart.

(1) Master operation in single master system

The flowchart when using 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 a multi-master system of the I^2C bus, it is not possible to judge whether the bus is in the release state or in use during the stage of participating in the communication based on the specifications of the I^2C bus. Here, if the data and clock are high for a certain amount of time (1 frame), the bus participates in communication as a release state. This process is roughly divided into “initial settings”, "communication waiting" and "communication processing". The processing designated as a slave due to the failure of the arbitration is omitted here, and only the processing used as the master device is omitted. Join the bus after performing the Initial Setup section at startup, and then wait for a communication request from the master device or a designation of the slave device via Communication Wait. The actual communication is the "Communication Processing" section, which supports arbitration with other master devices in addition to data sending and receiving with the slave.

(3) Slave operation

An example of an I^2C bus slave is shown below.

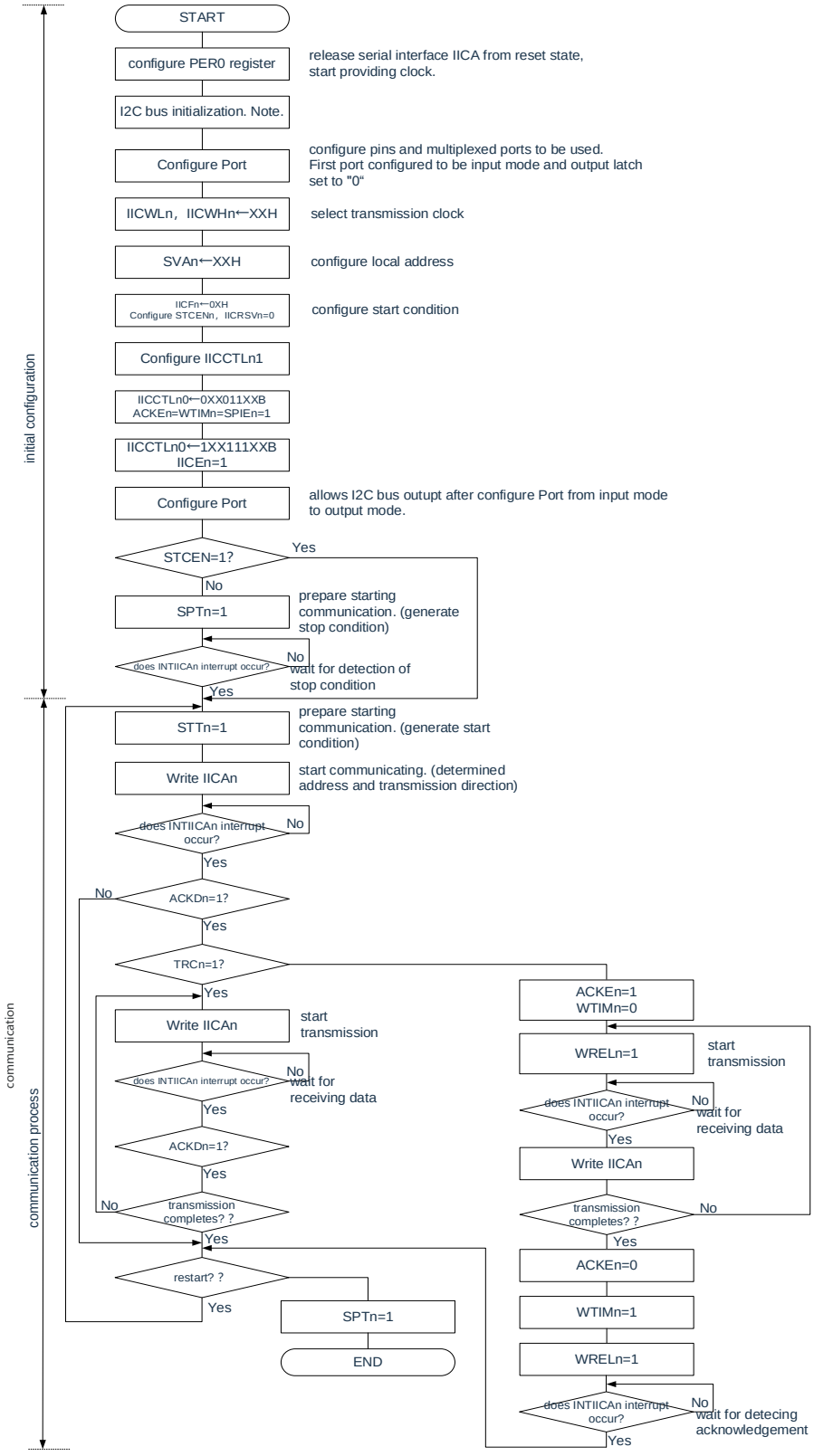
When used as the slave, operation is started by an interrupt. Execute the initial settings at startup, then wait for the INTIICAn interrupt occurrence (communication waiting). When an INTIICAn interrupt occurs, the communication status is judged and its result is passed as a flag over to the main processing.

By checking the flags, necessary communication processing is performed.

Remark: n=0

(1) Master operation of single-master system

Figure 14-19: Master operation of single-master system



Note: I²C-bus must be released (SCLAn pins and SDAAn pins are high) depending on the specifications of the product in communication. For example, if the EEPROM is in a state that outputs a low level to

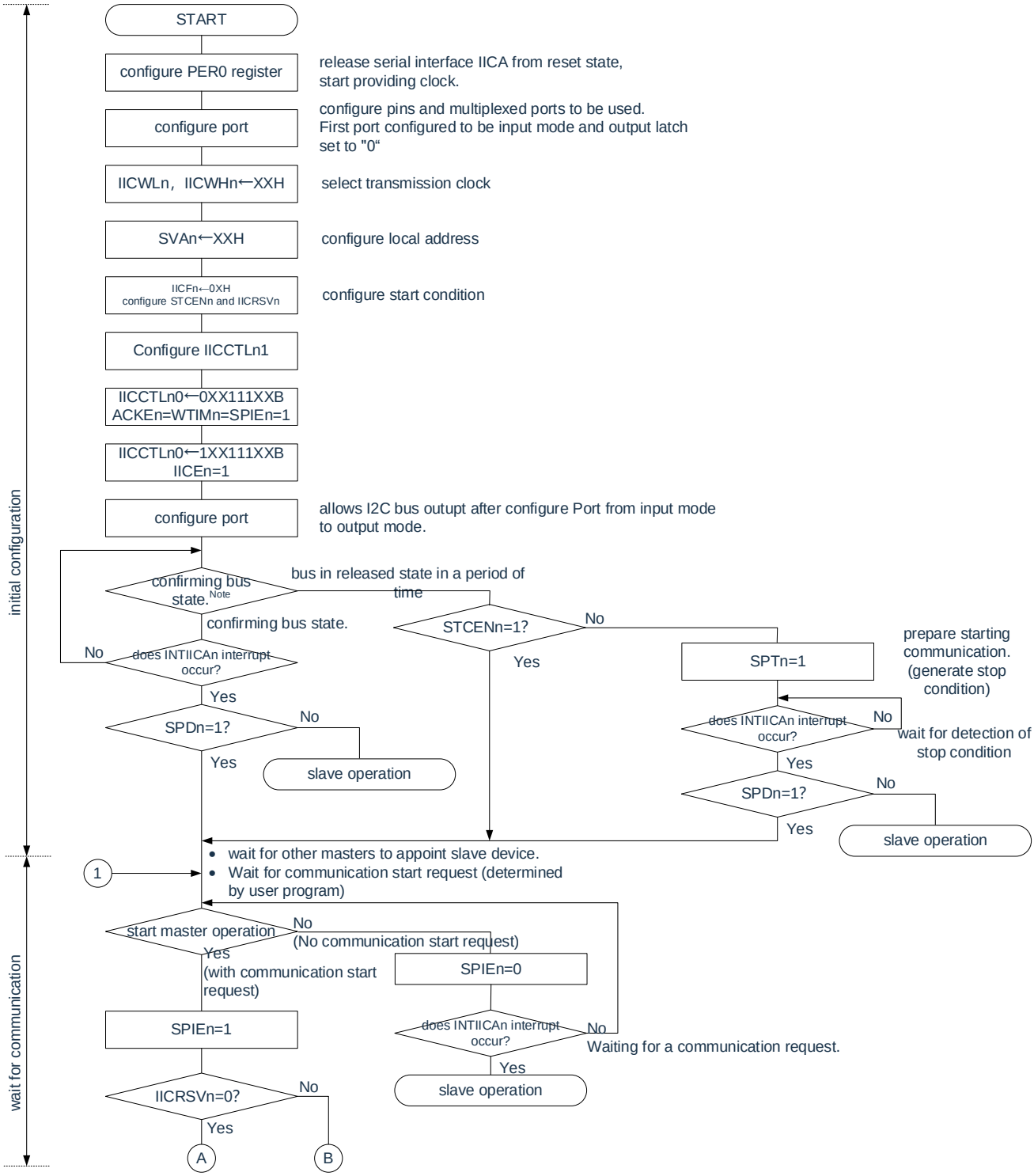
the SDAAn pin, the SCLAn pin must be set to the output port and a clock pulse must be output from the output port before the SDAAn pin is fixed high.

Notice: Conform to the specifications of the product that is communicating, with respect to the transmission and reception formats.

Remark: n=0

(2) Master operation in multi-master system

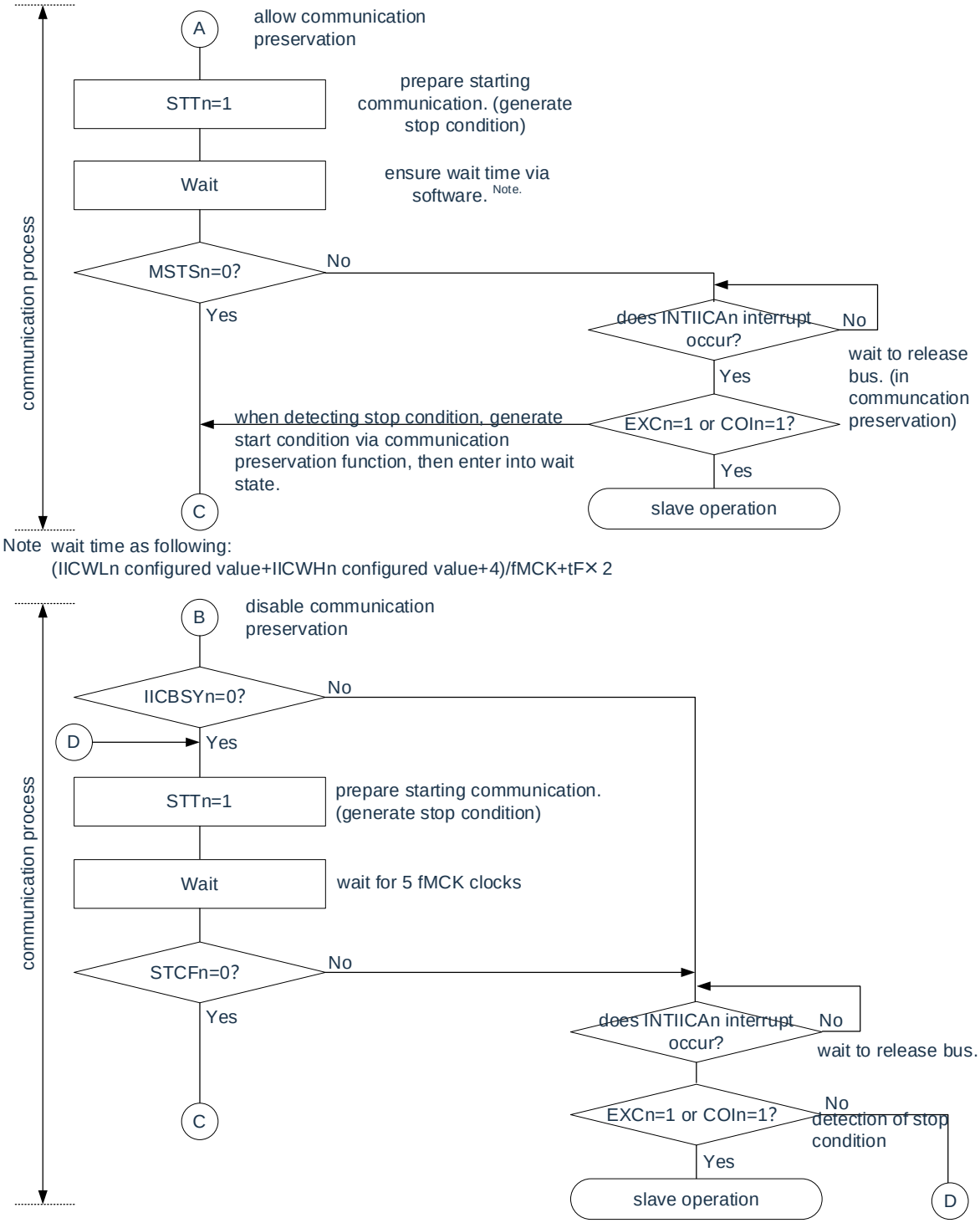
Figure14-20: Master operation in multi-master system (1/3)



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

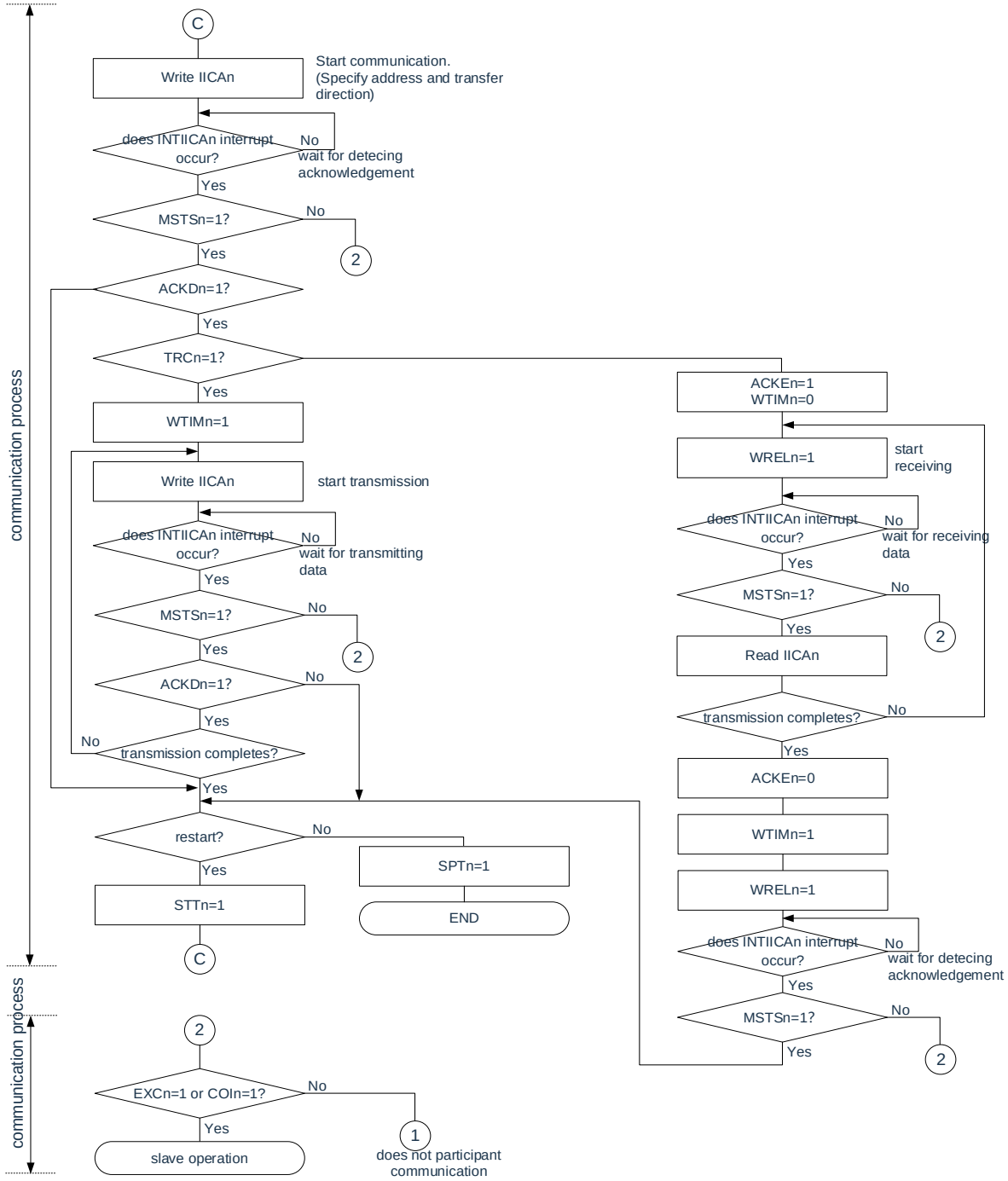
Figure 14-21: Master operation in multi-master system (2/3)



Remark:

1. IICWLn: IICA low-level width setting register n
   IICWHn: IICA high-level width setting register n
   T_F: SDAAn and SCLAn signal falling times
   F_{MCK}: IICA operation clock frequency
2. n=0

Figure 14-22: Master operation in multi-master system (3/3)



Notice:

1. The format of transmission and reception must conform to the specifications of the product in communication.
2. When used as a master device in a multi-master system, the MSTSn bit must be read each time an INTIICAn interrupt occurs to acknowledge the arbitration result.
3. When used as a slave device in a multi-master system, the status must be confirmed each time an INTIICAn interrupt occurs by means of the IICA status register n (IICSn) and the IICA flag register n (IICFn) to determine subsequent processing.

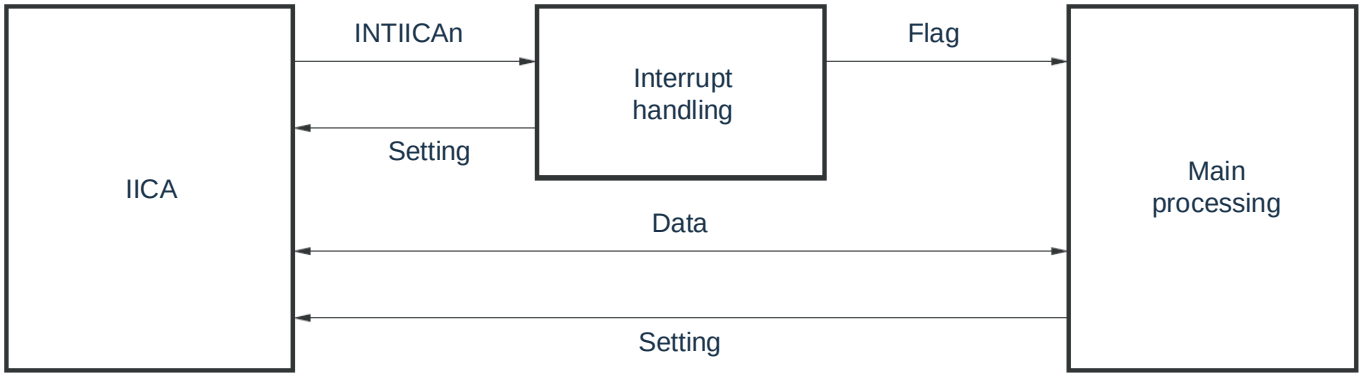
Remark: n=0

(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, the following three flags are prepared and pass the flags to the main processing department instead of INTRAICAn for data communication processing.

① Communication mode flag

This flag indicates the following 2 communication states:

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 handling 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.

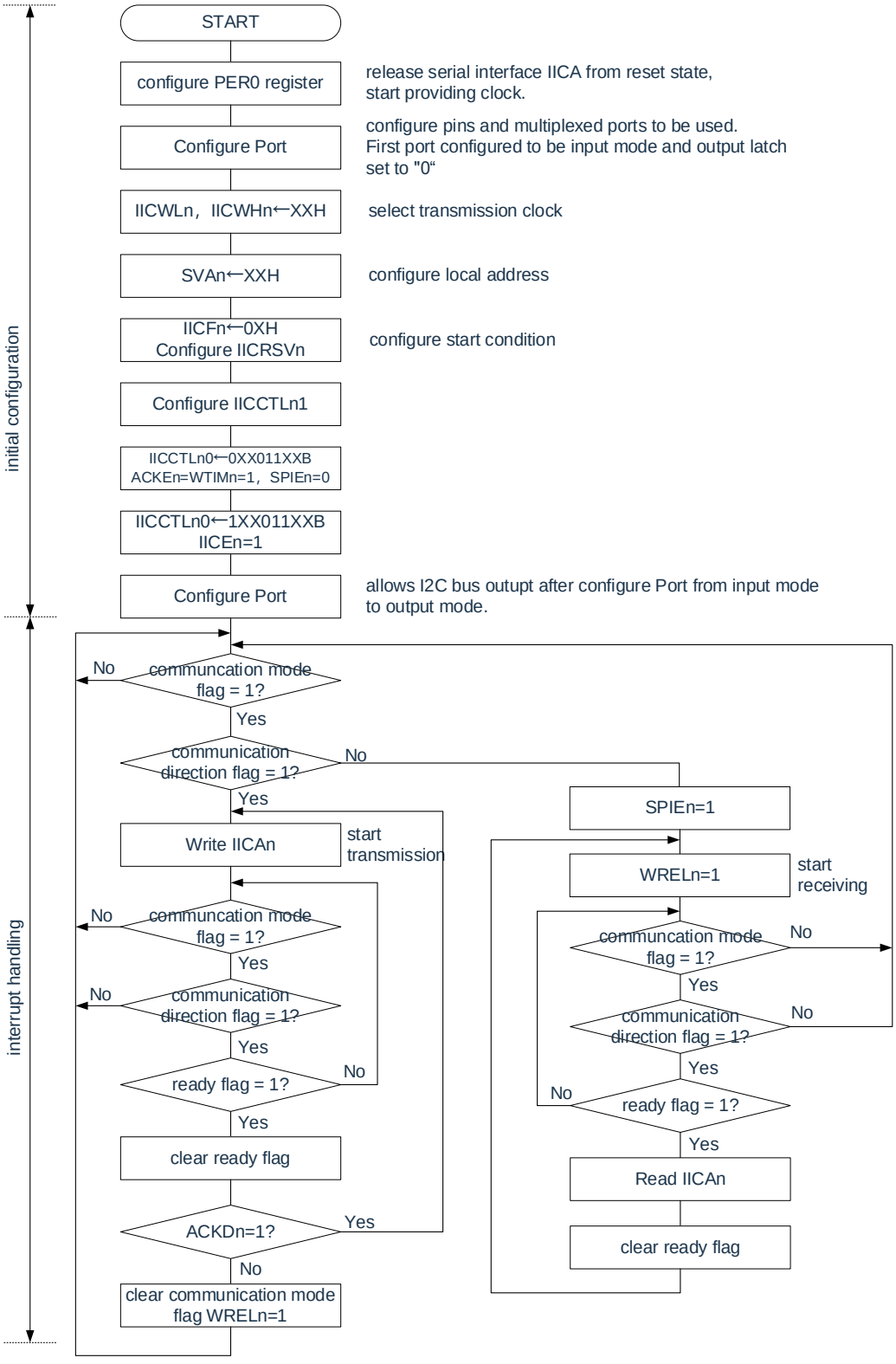
Remark: n=0

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, checks 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.

Figure 14-23: Slave operation flowchart (1)



Notice: The format of transmission and reception must conform to the specifications of the product in communication.

Remark: n=0

An example of the steps for a slave to process via an INTIICAn interrupt is shown below (assuming no extension code is used here). Confirm the status by interrupting THROUGH INTIICAn and perform the following processing.

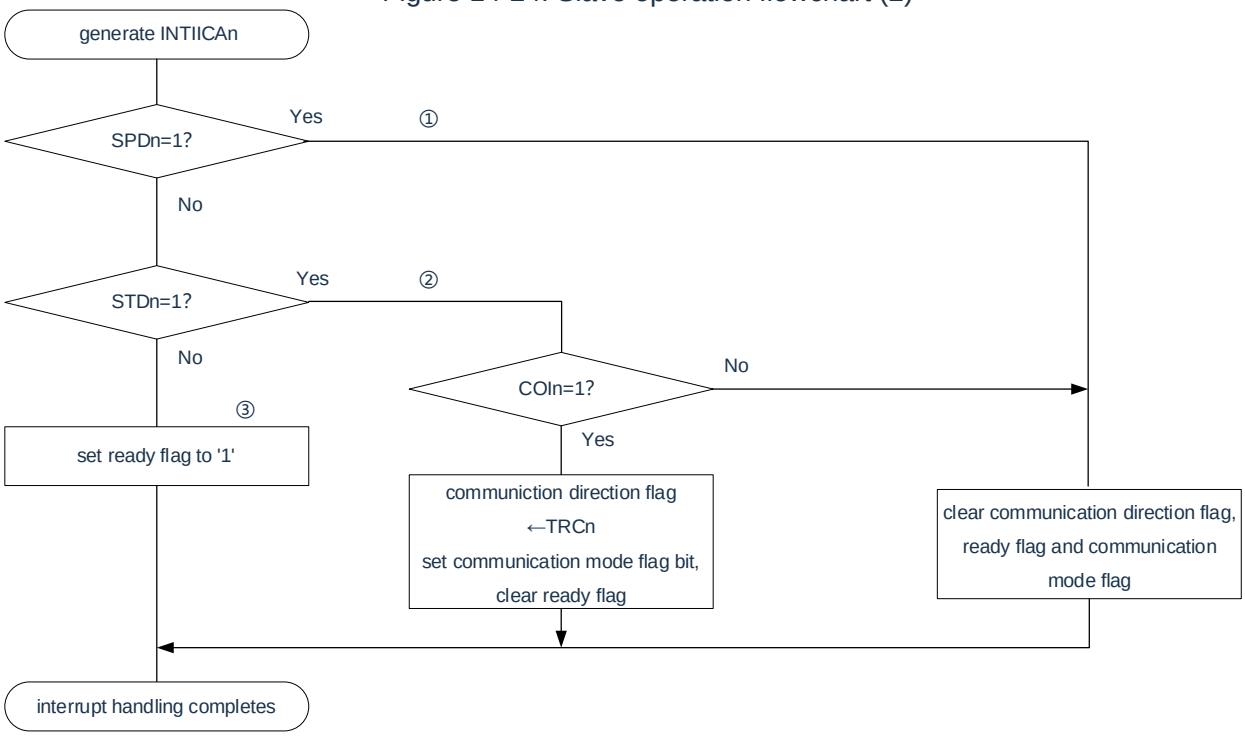
① Communication is stopped if the stop condition is issued.

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 14-24: Slave operation flowchart (2)”.

Figure 14-24: Slave operation flowchart (2)



Remark: n=0

14.5.17 Timing of I²C interrupt request (INTIICAn) generation

The values of the data send and receive timing, the timing of the generation of the INTIICAn interrupt request signal, and the IICA status register n (IICSn) when the INTIICAn signal is generated are shown below.

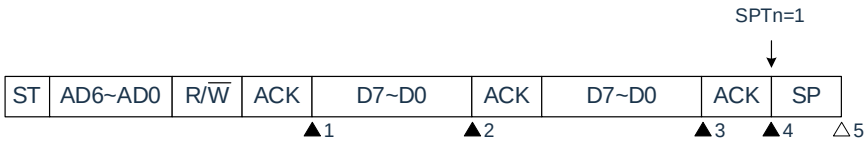
Remark:

1. ST: Start condition
   AD6~AD0: Address
   R/W: Transfer direction specification
   ACK: Acknowledge
   D7~D0: Data
   SP: Stop condition
2. n=0

(1) Master operation

① Start~Address~Data~Data~Stop (transmit and receive)

a) When WTIMn=0

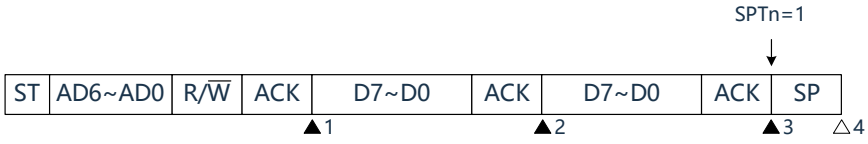


▲1:IICSn=1000X110B
▲2:IICSn=1000X000B
▲3:IICSn=1000X000B(set the WTIMn bit to 1) Note
▲4:IICSn=1000XX00B(set the SPTn bit to 1)
△5:IICSn=00000001B

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
X Any

b) When WTIMn=1



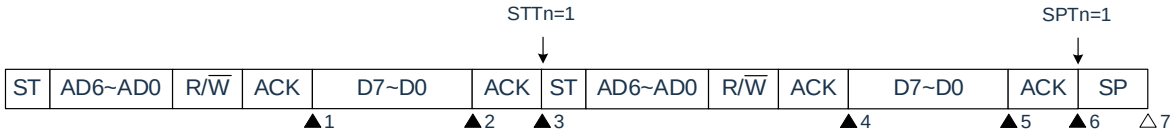
▲1:IICSn=1000X110B
▲2:IICSn=1000X100B
▲3:IICSn=1000XX00B(set the SPTn bit to 1)
△4:IICSn=00000001B

Remark ▲ Always generated
△ Generated only when SPIEn = 1
X Any

Remark: n=0

② Start~Address~Data~Start~Address~Data~Stop (restart)

a) When WTIMn=0

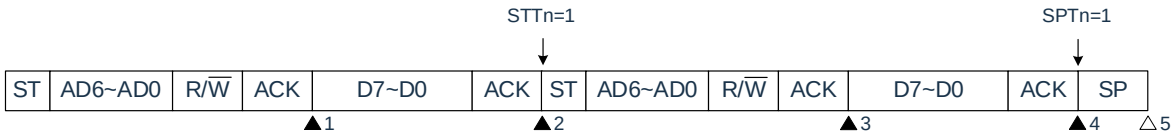


▲1:IICSn=1000X110B
▲2:IICSn=1000X000B(set the WTIMn bit to 1) Note1
▲3:IICSn=1000XX00B(set the WTIMn bit to 0. Note 2 and set the STTn bit to 1)
▲4:IICSn=1000X110B
▲5:IICSn=1000X000B(set the WTIMn bit to 1) Note3
▲6:IICSn=1000XX00B(set the SPTn bit to 1)
△7:IICSn=00000001B

Note1. 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 when SPIE =1
X Any

b) When WTIMn=1



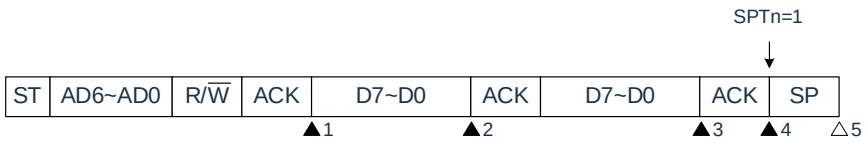
▲1:IICSn=1000X110B
▲2:IICSn=1000XX00B(set STTn bit to "1")
▲3:IICSn=1000X110B
▲4:IICSn=1000XX00B(set SPTn bit to "1")
△5:IICSn=00000001B

Remark ▲ Always generated
△ Generated only when SPIEn = 1
X Any

Remark: n=0

③ Start~Code~Data~Data~Stop (extension code transmission)

a) When WTIMn=0

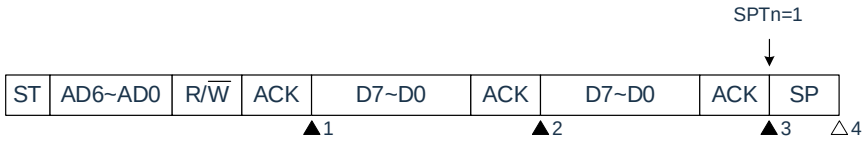


▲1:IICSn=1010X110B
▲2:IICSn=1010X000B
▲3:IICSn=1010X000B(Set the WTIMn bit to 1)Note
▲4:IICSn=1010XX00B(Set the SPTn bit to 1)
△5:IICSn=00000001B

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
X Any

b) When WTIMn=1



▲1:IICSn=1010X110B
▲2:IICSn=1010X100B
▲3:IICSn=1010XX00B(Set SPTn bit to 1)
△4:IICSn=00000001B

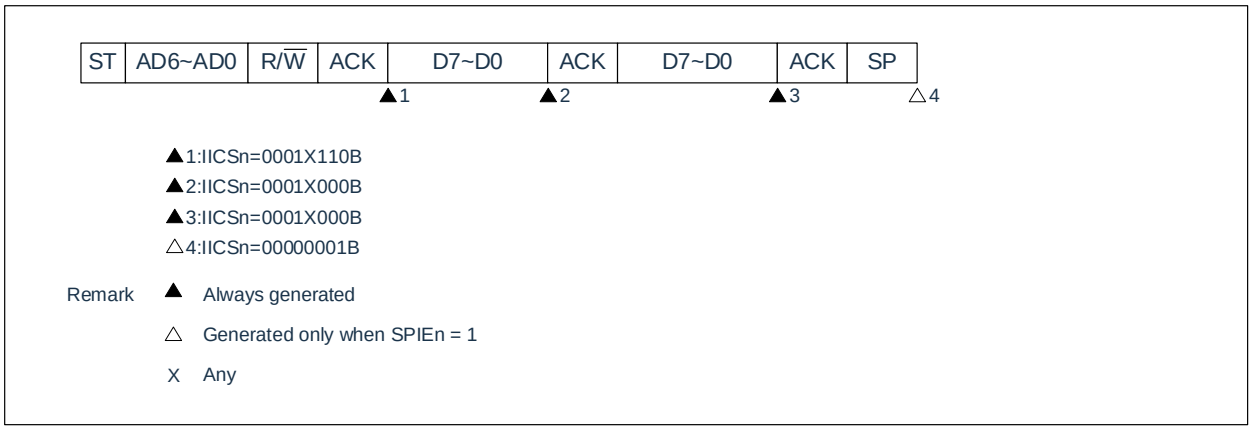
Remark ▲ Always generated
△ Generated only when SPIEn = 1
X Any

Remark: n=0

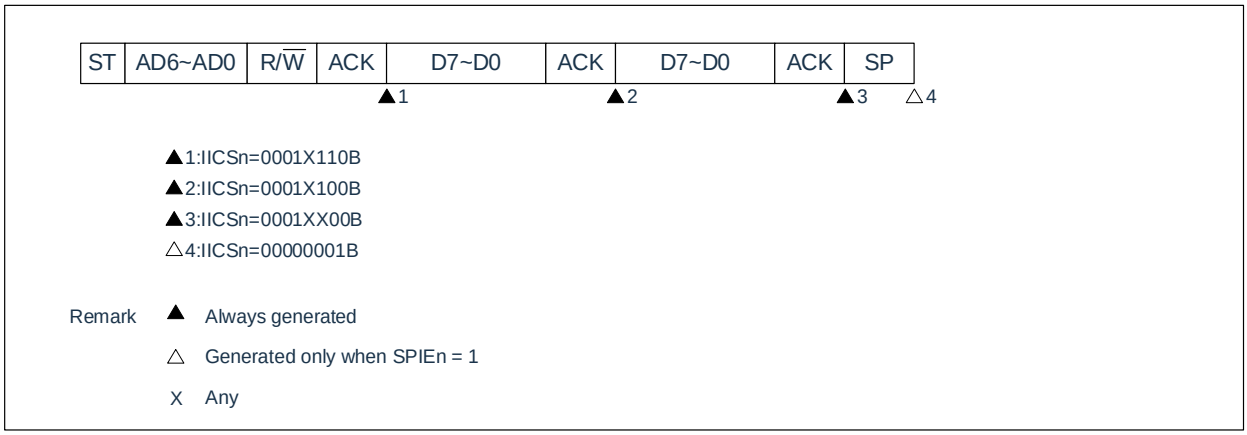
(2) Slave operation (when receiving a slave address)

① Start~Address~Data~Data~Stop

a) When WTIMn=0



b) When WTIMn=1



Remark: n=0

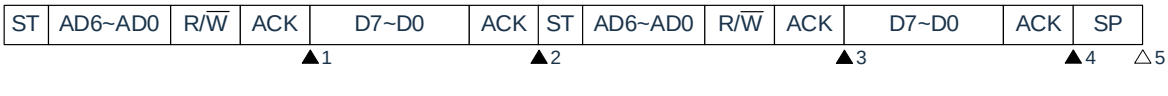
② Start~Address~Data~Start~Address~Data~Stop

a) When WTIMn=0 (after restart, matches with SVAn)

▲1:IICSn=0001X110B
▲2:IICSn=0001X000B
▲3:IICSn=0001X110B
▲4:IICSn=0001X000B
△5:IICSn=00000001B

Remark ▲ Always generated
△ Generated only when SPIEn = 1
X Any

b) When WTIMn=1 (after restart, matches with SVAn)



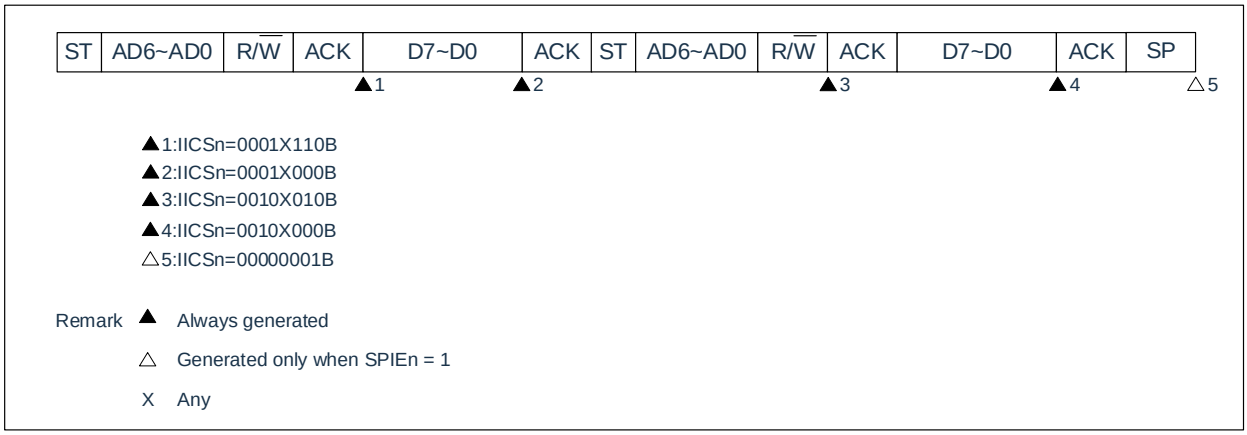
▲1:IICSn=0001X110B
▲2:IICSn=0001XX00B
▲3:IICSn=0001X110B
▲4:IICSn=0001XX00B
△5:IICSn=00000001B

Remark ▲ Always generated
△ Generated only when SPIEn = 1
X Any

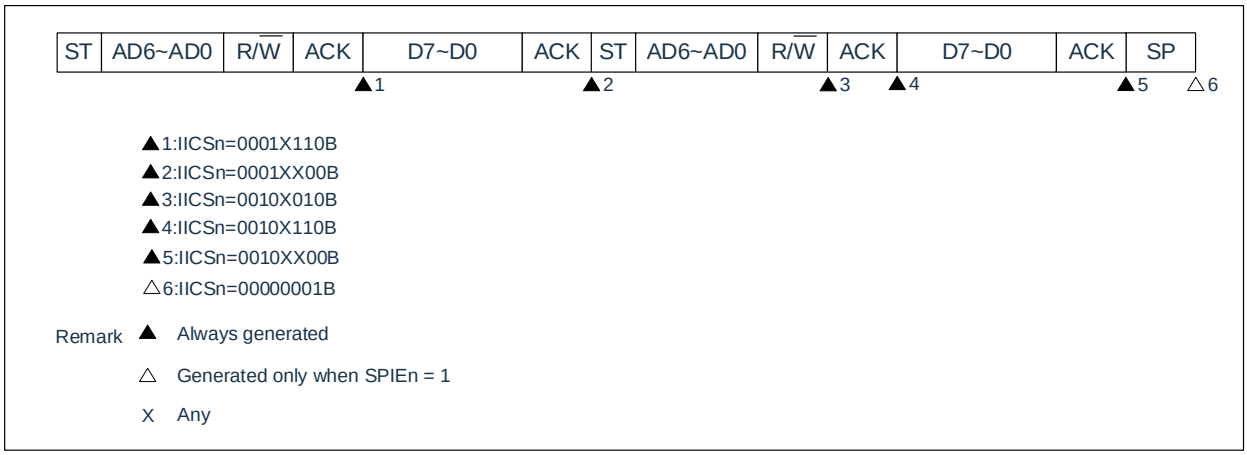
Remark: n=0

③ Start~Address~Data~Start~Code~Data~Stop

a) When WTIMn=0 (after restart, does not match address (= extension code))



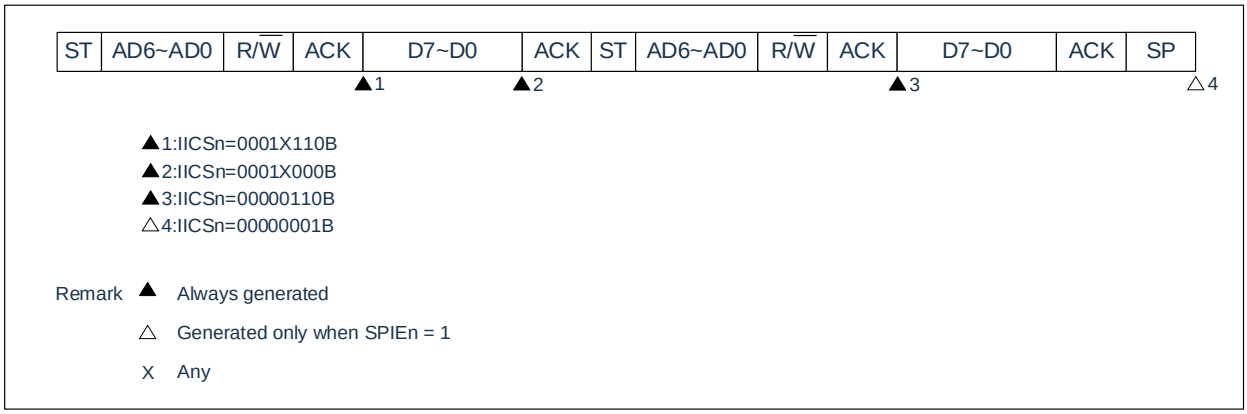
b) When WTIMn=1 (after restart, does not match address (= extension code))



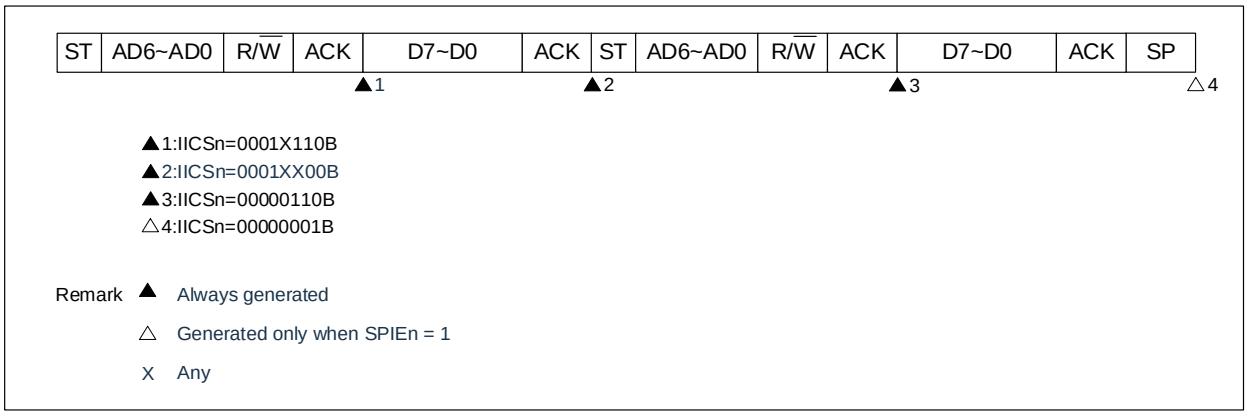
Remark: n=0

④ Start~Address~Data~Start~Address~Data~Stop

a) When WTIMn=0 (after restart, does not match address (= not extension code))



b) When WTIMn=1 (after restart, does not match address (= not extension code))



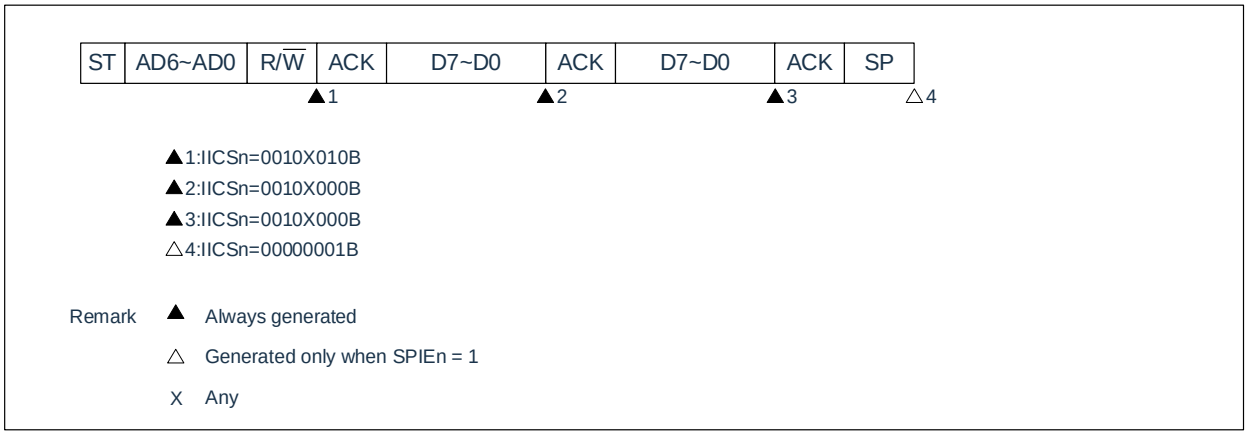
Remark: n=0

(3) Slave operation (when receiving extension code)

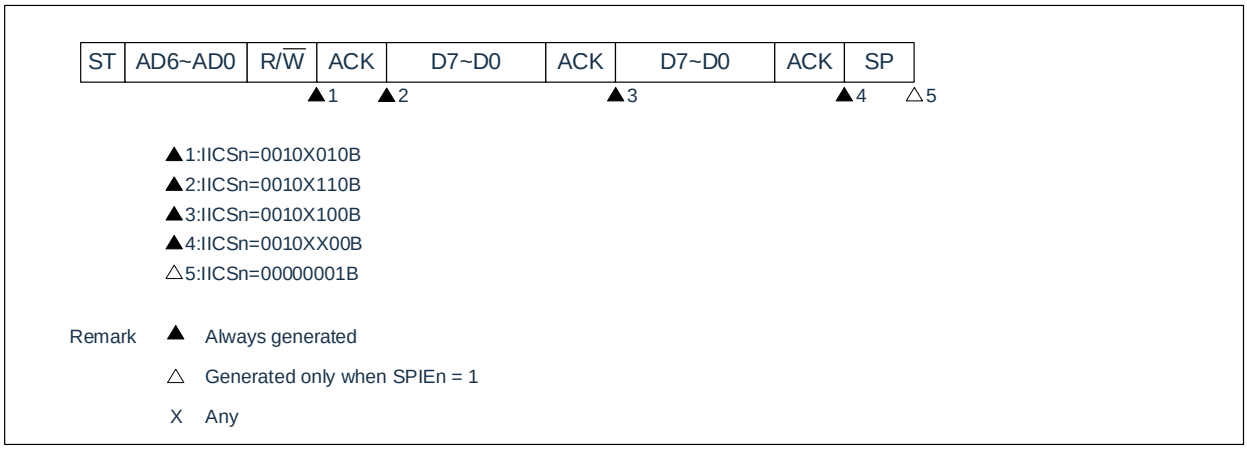
The device is always participating in communication when it receives an extension code.

① Start~Code~Data~Data~Stop

a) When WTIMn=250



b) When WTIMn=1



Remark: n=0

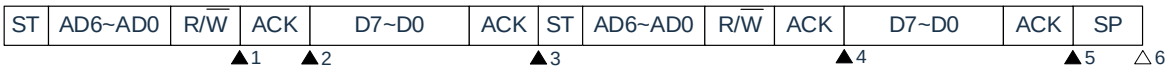
② Start~Code~Data~Start~Address~Data~Stop

a) When WTIMn=0 (after restart, matches SVAn)

▲1:IICSn=0010X010B
▲2:IICSn=0010X000B
▲3:IICSn=0001X110B
▲4:IICSn=0001X000B
△5:IICSn=00000001B

Remark ▲ Always generated
△ Generated only when SPIEn = 1
X Any

b) When WTIMn=1 (after restart, matches SVAn)



▲1:IICSn=0010X010B
▲2:IICSn=0010X110B
▲3:IICSn=0010XX00B
▲4:IICSn=0001X110B
▲5:IICSn=0001XX00B
△6:IICSn=00000001B

Remark ▲ Always generated
△ Generated only when SPIEn = 1
X Any

Remark: n=0

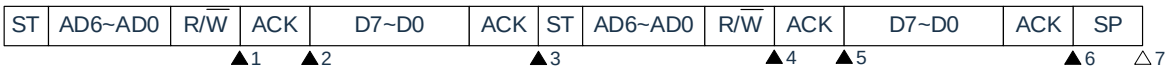
③ Start~Code~Data~Start~Code~Data~Stop

a) When WTIMn=0 (after restart, extension code reception)

▲1:IICSn=0010X010B
▲2:IICSn=0010X000B
▲3:IICSn=0010X010B
▲4:IICSn=0010X000B
△5:IICSn=00000001B

Remark ▲ Always generated
△ Generated only when SPIEn = 1
X Any

b) When WTIMn=1 (after restart, extension code reception)



▲1:IICSn=0010X010B
▲2:IICSn=0010X110B
▲3:IICSn=0010XX00B
▲4:IICSn=0010X010B
▲5:IICSn=0010X110B
▲6:IICSn=0010XX00B
△7:IICSn=00000001B

Remark ▲ Always generated
△ Generated only when SPIEn = 1
X Any

Remark: n=0

④ Start~Code~Data~Start~Address~Data~Stop

a) When WTIMn=0 (after restart, does not match address (= not extension code))

▲1:IICSn=0010X010B
▲2:IICSn=0010X000B
▲3:IICSn=00000X10B
△4:IICSn=00000001B

Remark
▲ Always generated
△ Generated only when SPIEn = 1
X Any

b) When WTIMn=1 (after restart, does not match address (= not extension code))

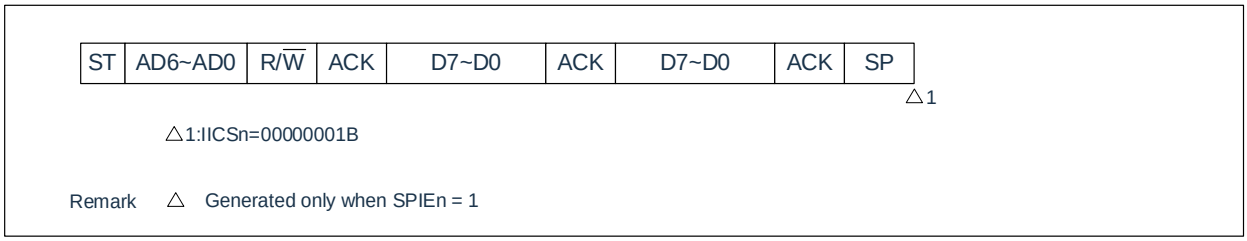
▲1:IICSn=0010X010B
▲2:IICSn=0010X110B
▲3:IICSn=0010XX00B
▲4:IICSn=00000X10B
△5:IICSn=00000001B

Remark
▲ Always generated
△ Generated only when SPIEn = 1
X Any

Remark: n=0

(4) Operation without communication

Start~Code~Data~Data~Stop

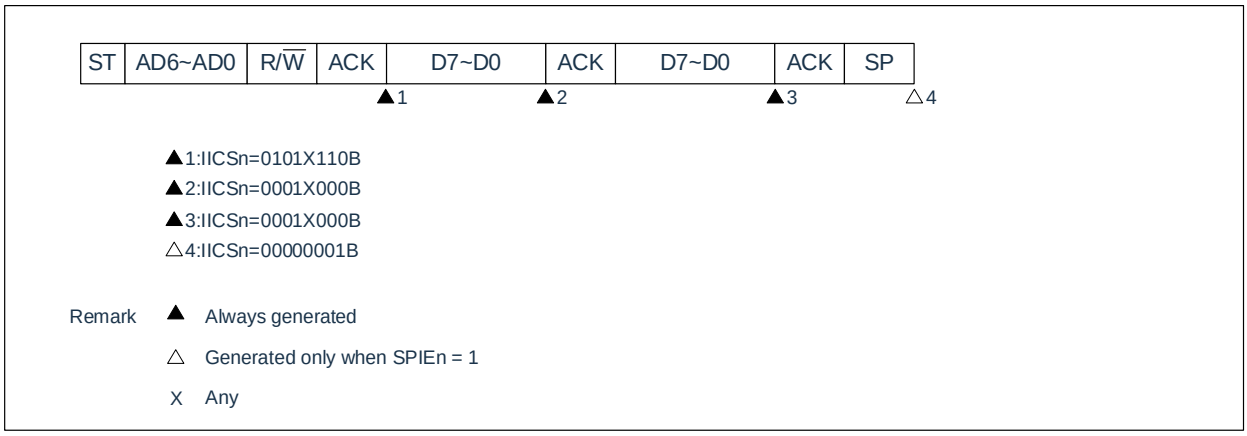


(5) Arbitration loss operation (operation as slave after arbitration loss)

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.

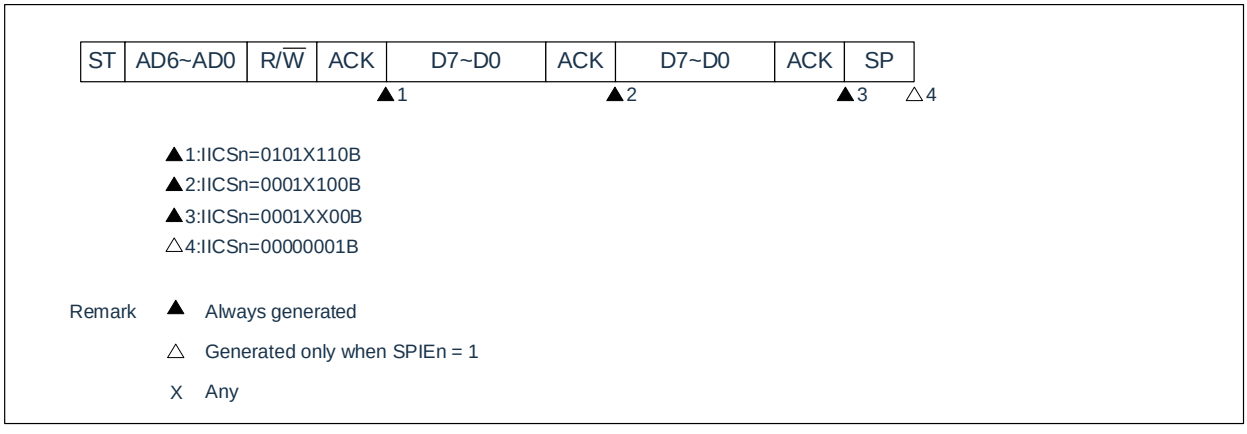
① When arbitration loss occurs during transmission of slave address data

a) When WTIMn=0



Remark: n=0

b) When WTIMn=1



② When arbitration loss occurs during transmission of extension code

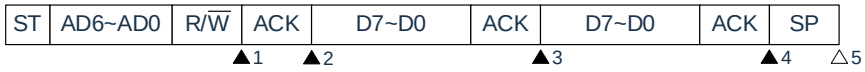
a) When WTIMn=0

▲1:IICSn=0110X010B
▲2:IICSn=0010X000B
▲3:IICSn=0010X000B
△4:IICSn=00000001B

Remark ▲ Always generated
△ Generated only when SPIEn = 1
X Any

Remark: n=0

b) When WTIMn=1



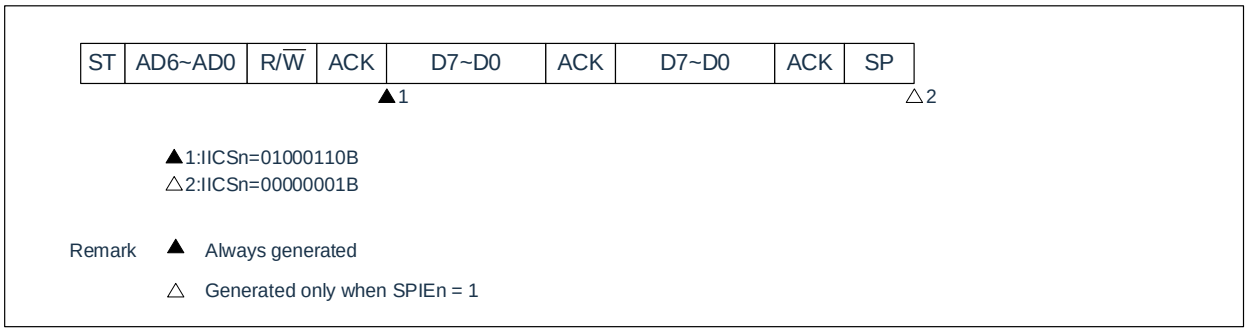
▲1:IICSn=0110X010B
▲2:IICSn=0010X110B
▲3:IICSn=0010X100B
▲4:IICSn=0010XX00B
△5:IICSn=00000001B

Remark ▲ Always generated
△ Generated only when SPIEn = 1
X Any

(6) Operation when arbitration loss occurs (no communication after arbitration loss)

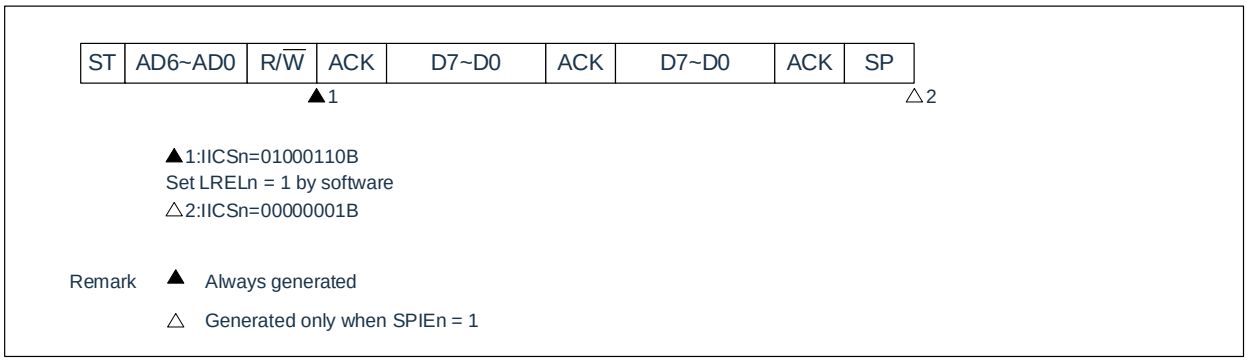
When the device is used as a master in a multi-master system, read the MSTSn bit each time interrupt request signal INTIICAn has occurred to check the arbitration result.

① When arbitration loss occurs during transmission of slave address data (when WTIMn = 1)



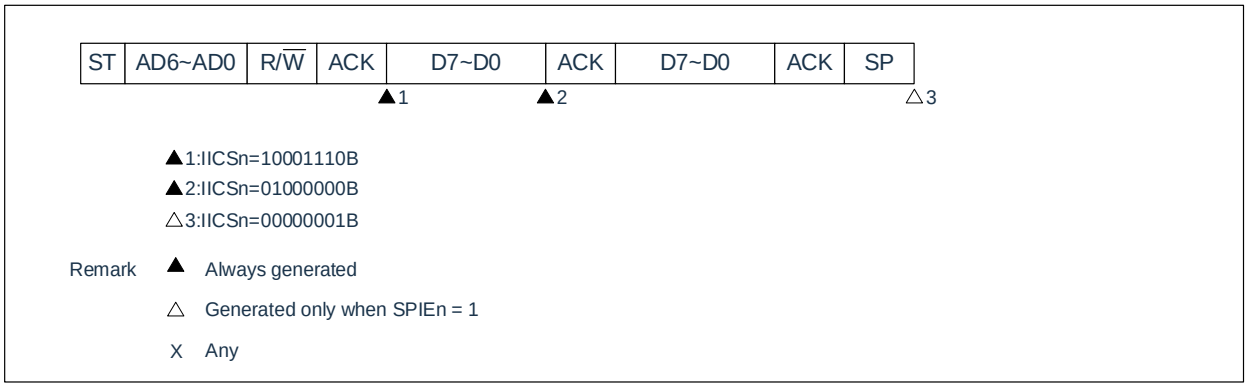
Remark: n=0

② When arbitration loss occurs during transmission of extension code



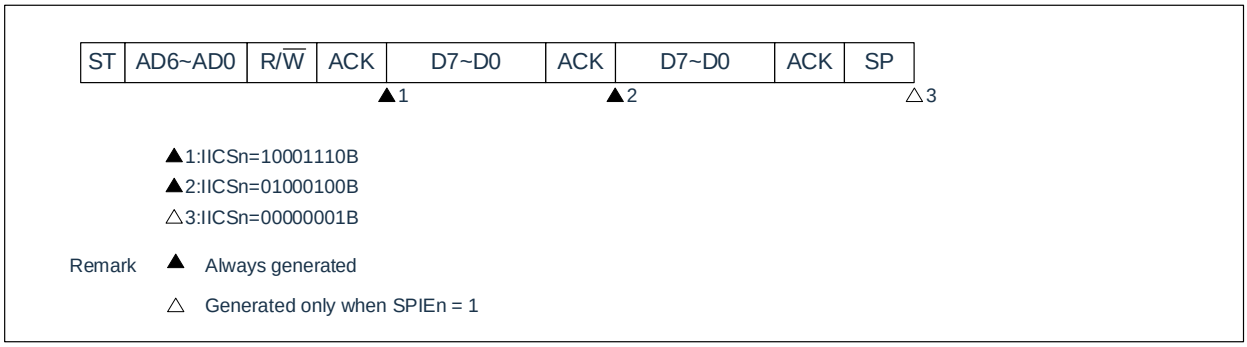
③ When arbitration loss occurs during transmission of data

a) When WTIMn=0



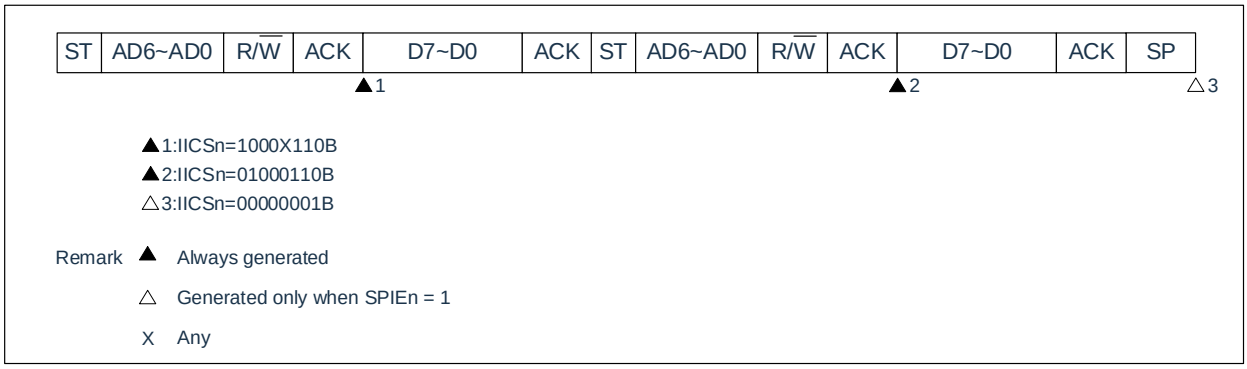
Remark: n=0

b) When WTIMn=1



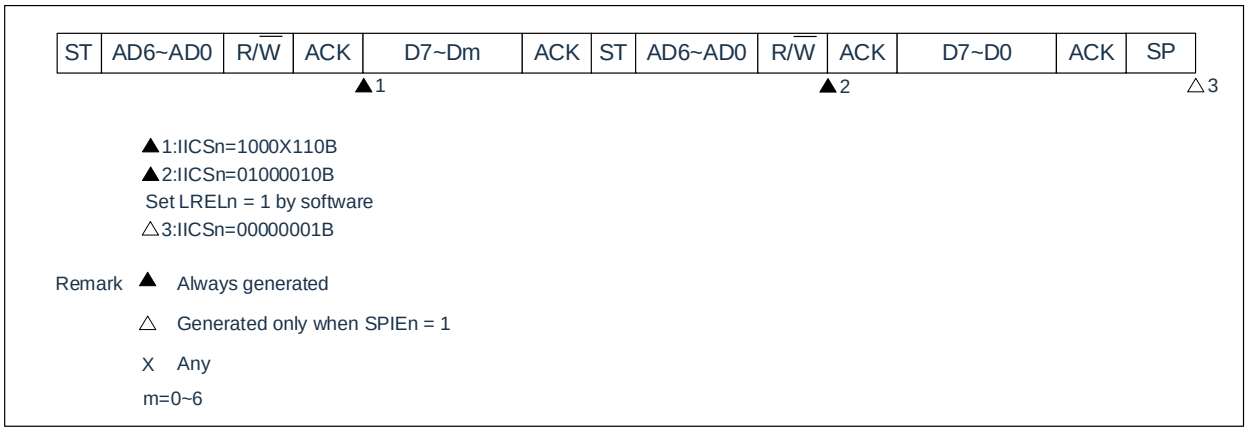
④ When loss occurs due to restart condition during data transfer

a) Not extension code (Example: unmatches with SVAn)

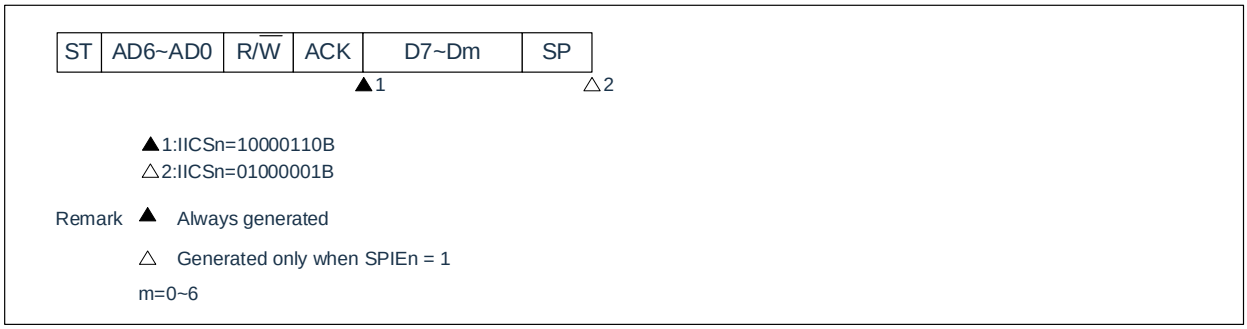


Remark: n=0

b) Extension code



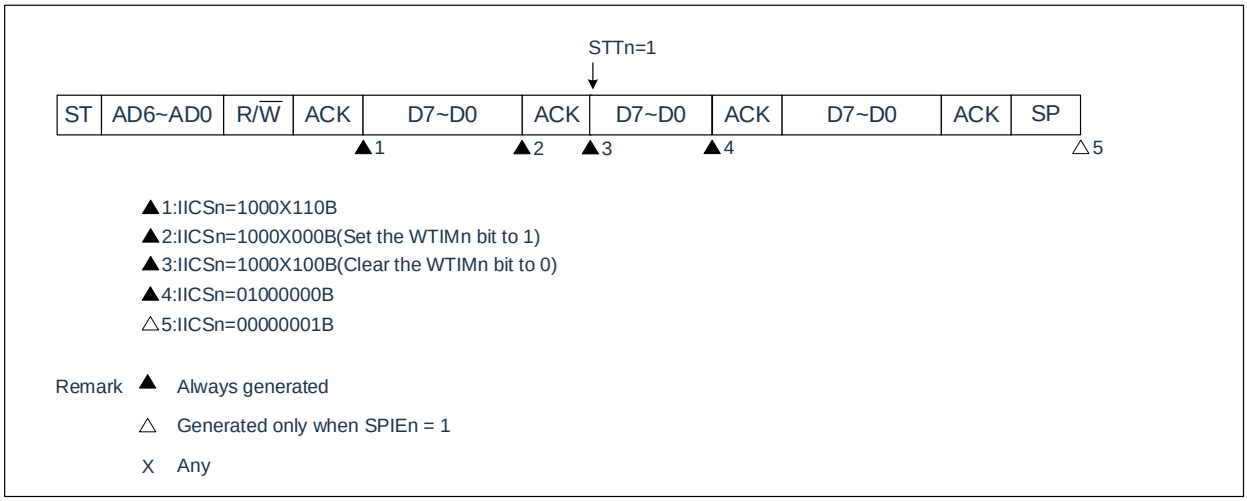
⑤ When loss occurs due to stop condition during data transfer



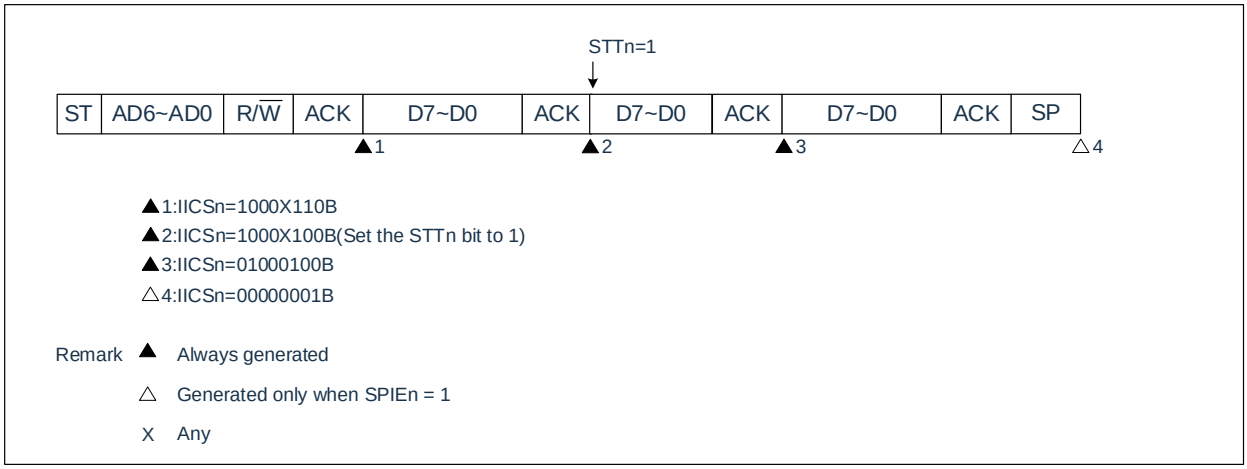
Remark: n=0

⑥ When arbitration loss occurs due to low-level data when attempting to generate a restart condition

a) When WTIMn=0



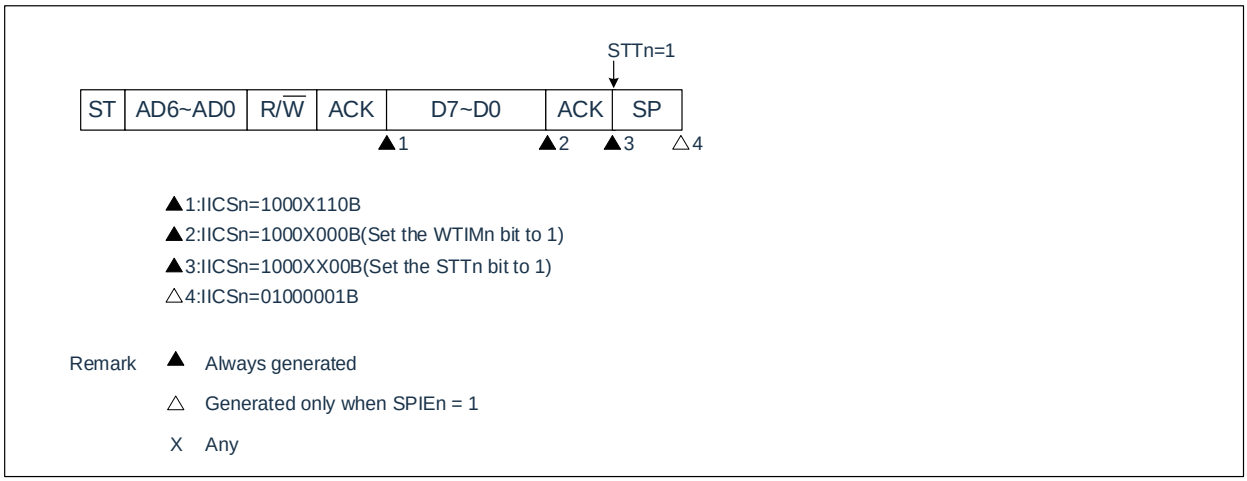
b) When WTIMn=1



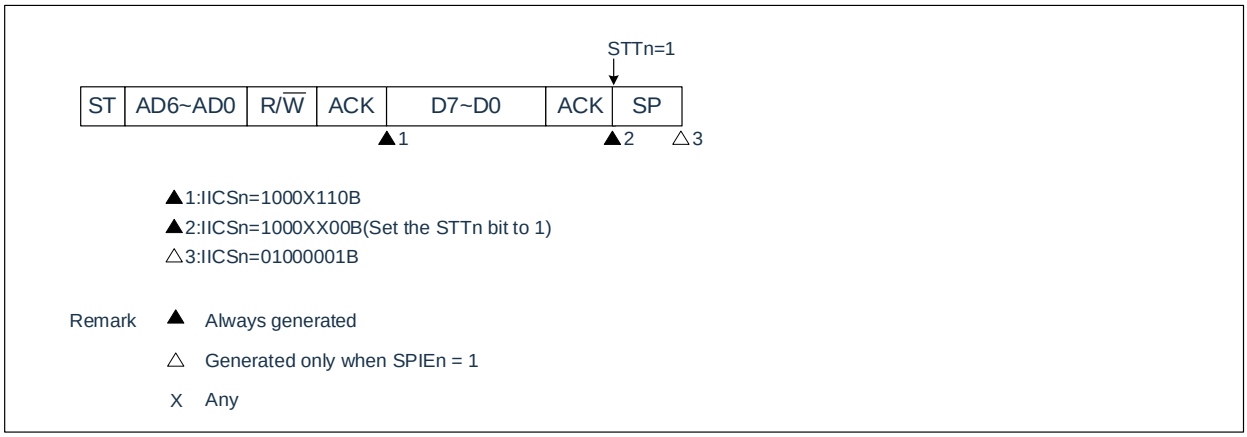
Remark: n=0

⑦ When arbitration loss occurs due to a stop condition when attempting to generate a restart condition

a) When WTIMn=0



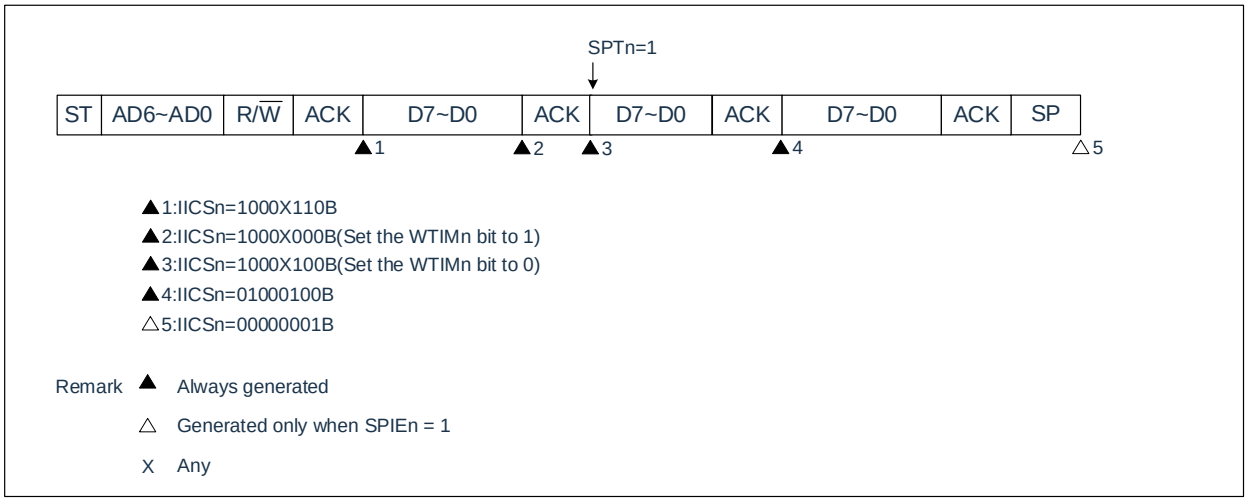
b) When WTIMn=1



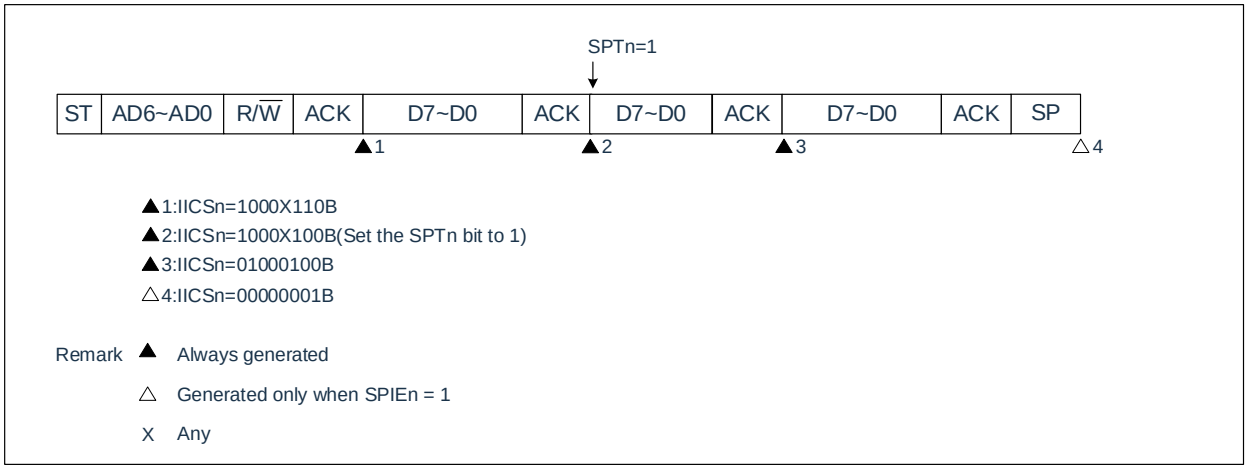
Remark: n=0

⑧ When arbitration loss occurs due to low-level data when attempting to generate a stop condition

a) When WTIMn=0



b) When WTIMn=1



Remark: n=0

14.6 Timing diagram

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. Timing diagrams of the data communication are shown in Figure 14-22 and Figure 14-23.

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.

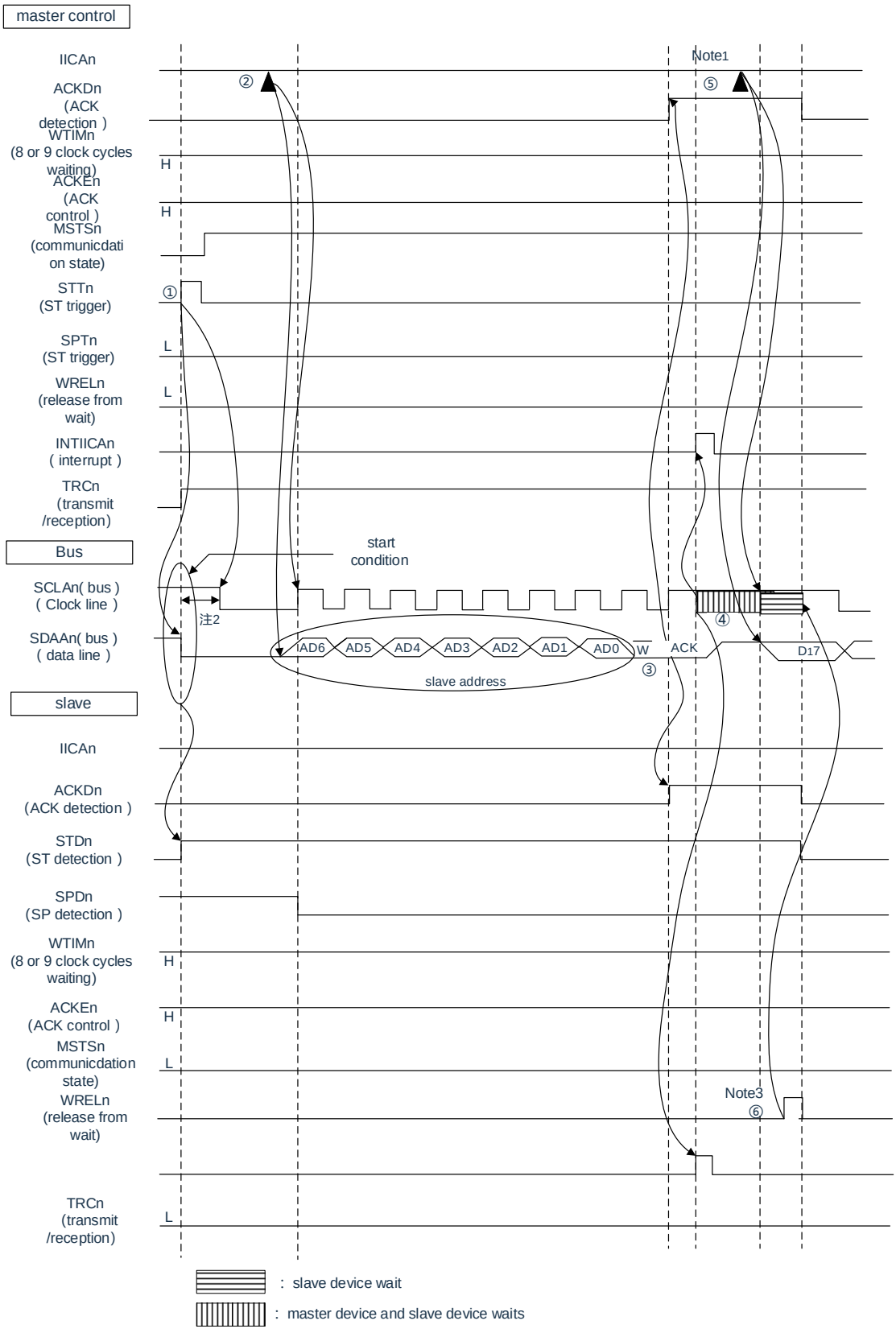
Data input via the SDAAn pin is captured into IICAn at the rising edge of SCLAn.

Remark: n=0

Figure 14-25: Example of master to slave communication

(Master: selects 9 clocks to wait, slave: selects 9 clocks to wait) (1/4)

(1) Start Condition ~ Address ~ Data



Note 1: Write data to IICAn, not setting the WRELn bit, in order to cancel a wait state during transmission by a master device.

Note 2: Make sure that the time between the fall of the SDAAn pin signal and the fall of the SCLAn pin signal is at least 4.0 μs when specifying standard mode and at least 0.6 μs when specifying fast mode.

Note 3: For releasing wait state during reception of a slave device, write “FFH” to IICAn or set the WRELn bit.

Figure 14-25 shows the descriptions of ① to ⑥ of “(1) Start condition ~ Address ~ Data”

① If the master sets the start condition trigger set (STTn=1), the bus data line (SDAAn) drops and the start condition is generated (SDAAn is changed from "1" to "0" by SCLAn=1). Thereafter, if a start condition is detected, the master enters the master communication state (MSTSn=1) and after the hold time elapses the bus clock line drops (SCLAn=0), ending the communication preparation.

② If the master writes address +W (transmit) to IICA shift register n (IICAn), the slave address is sent.

③ On the slave, if the receiving address and the local station address (the value of the SVAn) are the same [Note], an ACK is sent to the master through hardware. The master detects ACK on the rising edge of the 9th clock (ACKDn=1).

④ The master generates an interrupt on the falling edge of the 9th clock (INTIICAn: address send end interrupt). Slaves with the same address enter a waiting state (SCLAn=0) and an interrupt (INTIICAn: address matching interrupt) [Note].

⑤ The master writes and transmits data to the IICAn registers, relieving the master of waiting.

⑥ If the slave releases the wait (WRELn=1), the master begins to transmit data to the slave.

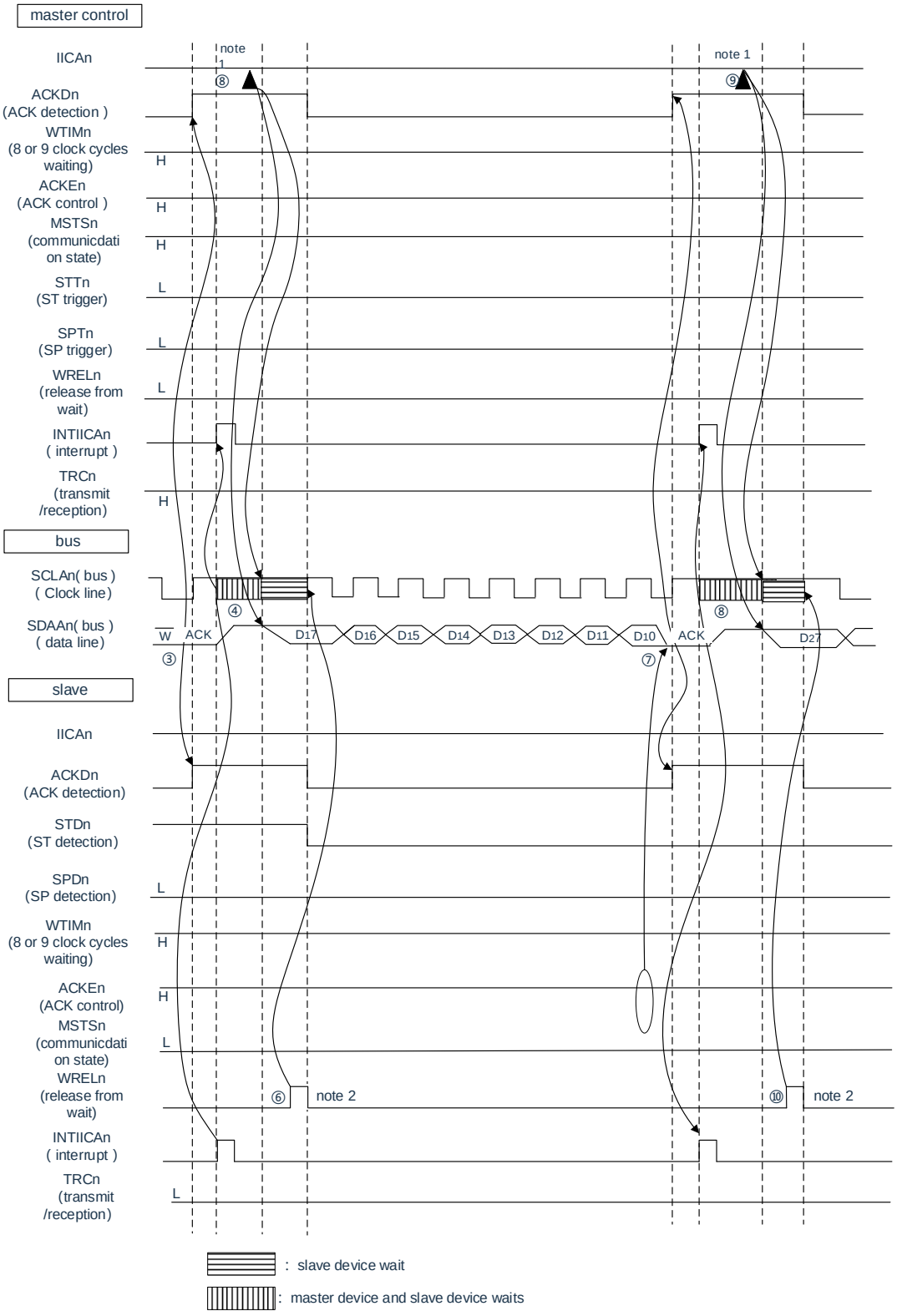
Note: If the sent address and the slave address are different, the slave does not return an ACK (NACK: SDAAn=1) to the master, and does not generate an INTIICAn interrupt (address matching interrupt) or enter a waiting state. However, the master generates ANTIICAn interrupts (address send end interrupts) for both ACK and NAK.

Remark:

1. ①~⑮ of Figure 14-25 shows a series of operational steps for data communication via the I2C bus.
   “(1) Start Condition~Address~Data” of Figure 14-25 illustrates steps ①~⑥.
   “(2) Address~Data~Data” of Figure 14-25 illustrates steps ③~⑩.
   “(3) Data~Data~Stop Condition” of Figure 14-25 illustrates steps ⑦~⑮.
2. n=0

Figure 14-25: Example of master to slave communication
(Master: selects 9 clocks to wait, slave: selects 9 clocks to wait) (2/4)

(2) Address ~ Data ~ Data



Note 1: Write data to IICAn, not setting the WRELn bit, in order to cancel a wait state during transmission by a master device.

Note 2: For releasing wait state during reception of a slave device, write “FFH” to IICAn or set the WRELn bit.

Remark: n=0

Figure 14-25 shows the descriptions of ③ to ⑩ of "(2) Address~Data~Data"

③ On the slave, if the receiving address and the local station address (the value of the SVAn) are the same [Note], the ACK is sent to the master through the hardware. The master detects ACK on the rising edge of the 9th clock (ACKDn=1).

④ The master generates an interrupt on the falling edge of the 9th clock (INTIICAn: address send end interrupt). Slaves with the same address enter a waiting state (SCLAn=0) and an interrupt (INTIICAn: address matching interrupt) [Note].

⑤ The master writes and sends data to the IICA shift register n (IICAn) to remove the wait of the main controller.

⑥ If the slave releases the wait (WRELn=1), the master party begins to transmit data to the slave.

⑦ After the data transfer is completed, because the ACKEn bit of the slave is "1", the ACK is sent to the master control through hardware. The master detects ACK on the rising edge of the 9th clock (ACKDn=1).

⑧ Both the master and the slave enter a waiting state (SCLAn= 0) on the falling edge of the 9th clock, and both produce interrupts (INTIICAn: Transmit End Interrupt).

⑨ The master controller writes and sends data to the IICAn register to remove the waiting of the main controller.

⑩ If the slave reads and receives the data and cancels the wait (WRELn=1), the master party begins to transmit data to the slave.

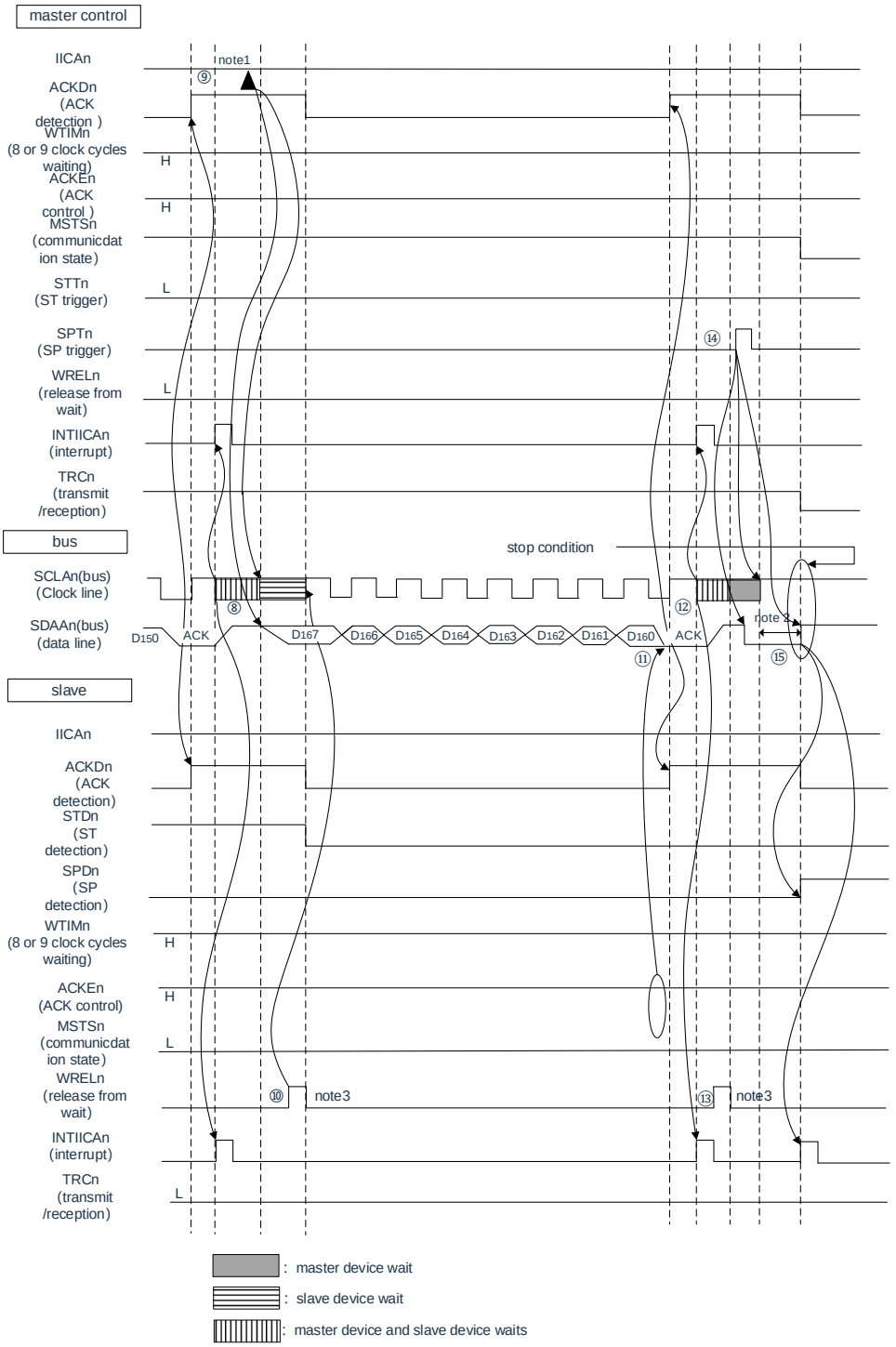
Note: If the sent address and the slave address are different, the slave does not return an ACK (NACK: SDAAn=1) to the master and does not generate an INTIICAn interrupt (address matching interrupt) or enter a waiting state. However, the master generates ANTIICAn interrupts (address send end interrupts) for both ACK and NAK.

Remark:

1. ①~⑮ of Figure 14-25 shows a series of operational steps for data communication via the I2C bus.
   “(1) Start Condition~Address~Data” of Figure 14-25 illustrates steps ①~⑥.
   “(2) Address~Data~Data” of Figure 14-25 illustrates steps③~⑩.
   “(3) Data~Data~Stop Condition” of Figure 14-25 illustrates steps ⑦~⑮.
2. n=0

Figure 14-25: Example of master to slave communication
(Master: selects 9 clocks to wait, slave: selects 9 clocks to wait) (3/4)

(3) Data ~ Data ~ Stop Condition



Note 1: Write data to IICAn, not setting the WRELn bit, in order to cancel a wait state during transmission by a master device.

Note 2: After the stop condition is issued, the time from the SCLAn pin signal to generate the stop condition is at least 4.0μs when set to standard mode and at least 0.6μs when set to fast mode.

Note 3: For releasing wait state during reception of a slave device, write “FFH” to IICAn or set the WRELn bit.

Remark: n=0

Figure 14-25 shows the descriptions of ⑦~⑮ of “(3) Data ~ Data ~ Stop”:

⑦ At the end of the data transfer, because the ACKEn bit of the slave is "1", the ACK is sent to the master through the hardware. The master detects ACK on the rising edge of the 9th clock (ACKDn=1)

⑧ Both the master and slave enter a waiting state (SCLAn=0) on the falling edge of the 9th clock, and both produce an interrupt (INTIICAn: end-of-transmit interrupt).

⑨ The master writes and transmits data to the IICA shift register n (IICAn), relieving the master of waiting.

⑩ If the slave reads the received data and dismisses the wait (WRELn=1), the master starts transmitting data to the slave

⑪ At the end of the data transfer, the slave (ACKEn=1) sends an ACK to the master through the hardware. The master detects ACK on the rising edge of the 9th clock (ACKDn=1).

⑫ Both the master and slave enter a waiting state (SCLAn=0) on the falling edge of the 9th clock, and both produce an interrupt (INTIICAn: end-of-transmit interrupt).

⑬ The slave reads the received data and dismisses the wait (WRELn=1).

⑭ If the master sets the stop condition trigger set (SPTn=1), the bus data line (SDAAn=0) is cleared and the bus clock line is set (SCLAn=1), and the bus data line is set after the preparation time for the stop condition is passed (SDAAn=1), Generate a stop condition (SDAAn from "0" to "1" by SCLAn=1).

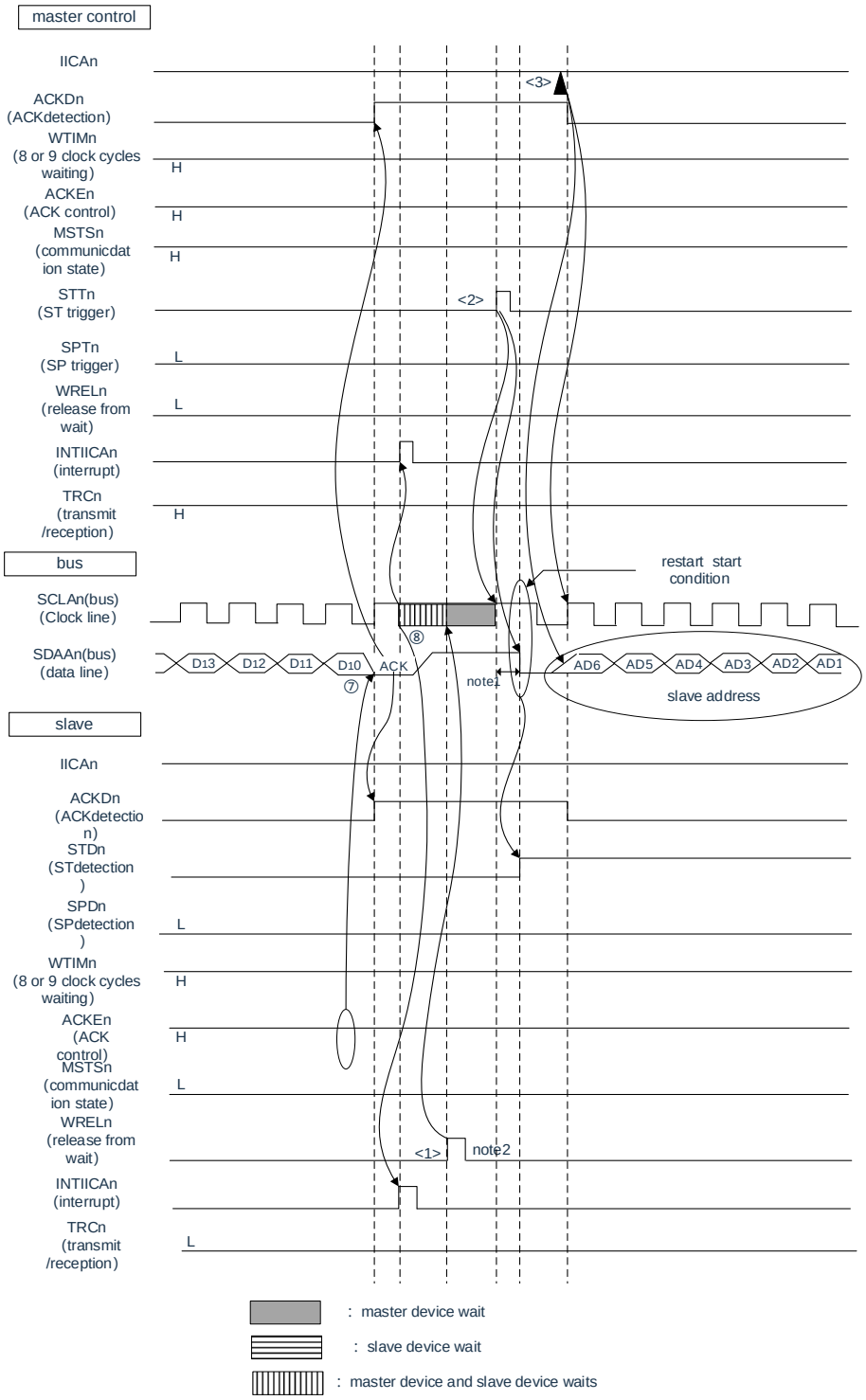
⑮ If a stop condition is generated, the slave detects the stop condition and generates an interrupt (INTIICAn: Stop condition interrupt).

Remark:

1. ①~⑮ of Figure 14-25 shows a series of operational steps for data communication via the I^2C bus.
   “(1) Start Condition~Address~Data” of Figure 14-25 illustrates steps ①~⑥.
   “(2) Address~Data~Data” of Figure 14-25 illustrates steps③~⑩.
   “(3) Data~Data~Stop Condition” of Figure 14-25 illustrates steps ⑦~⑮.
2. n=0

Figure 14-25: Example of master to slave communication
(Master: selects 9 clocks to wait, slave: selects 9 clocks to wait) (4/4)

(4) Data ~ Restart Condition ~ Address



Note 1: Make sure that the time between the rise of the SCLAn pin signal and the generation of the start condition after a restart condition has been issued is at least 4.7 μs when specifying standard mode and at least 0.6 μs when specifying fast mode.

Note 2: For releasing wait state during reception of a slave device, write “FFH” to IICAn or set the WRELn bit.

Remark: n=0

Figure 14-25 shows the operation of "(4) Data ~ Restart Condition ~ Address" as follows. After executing steps 7 and 8, execute <1> to <3> to return to the data sending step of step 3.

⑦ After the data transfer is completed, because the ACKEn bit of the slave is "1", the ACK is sent to the master control through hardware. The master detects ACK on the rising edge of the 9th clock (ACKDn=1).

⑧ Both the master and the slave enter a waiting state (SCLAn= 0) on the falling edge of the 9th clock, and both produce interrupts (INTIICAn: Transmit End Interrupt).

<1> The slave reads and receives the data, and the wait is released (WRELn=1).

<2> 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.

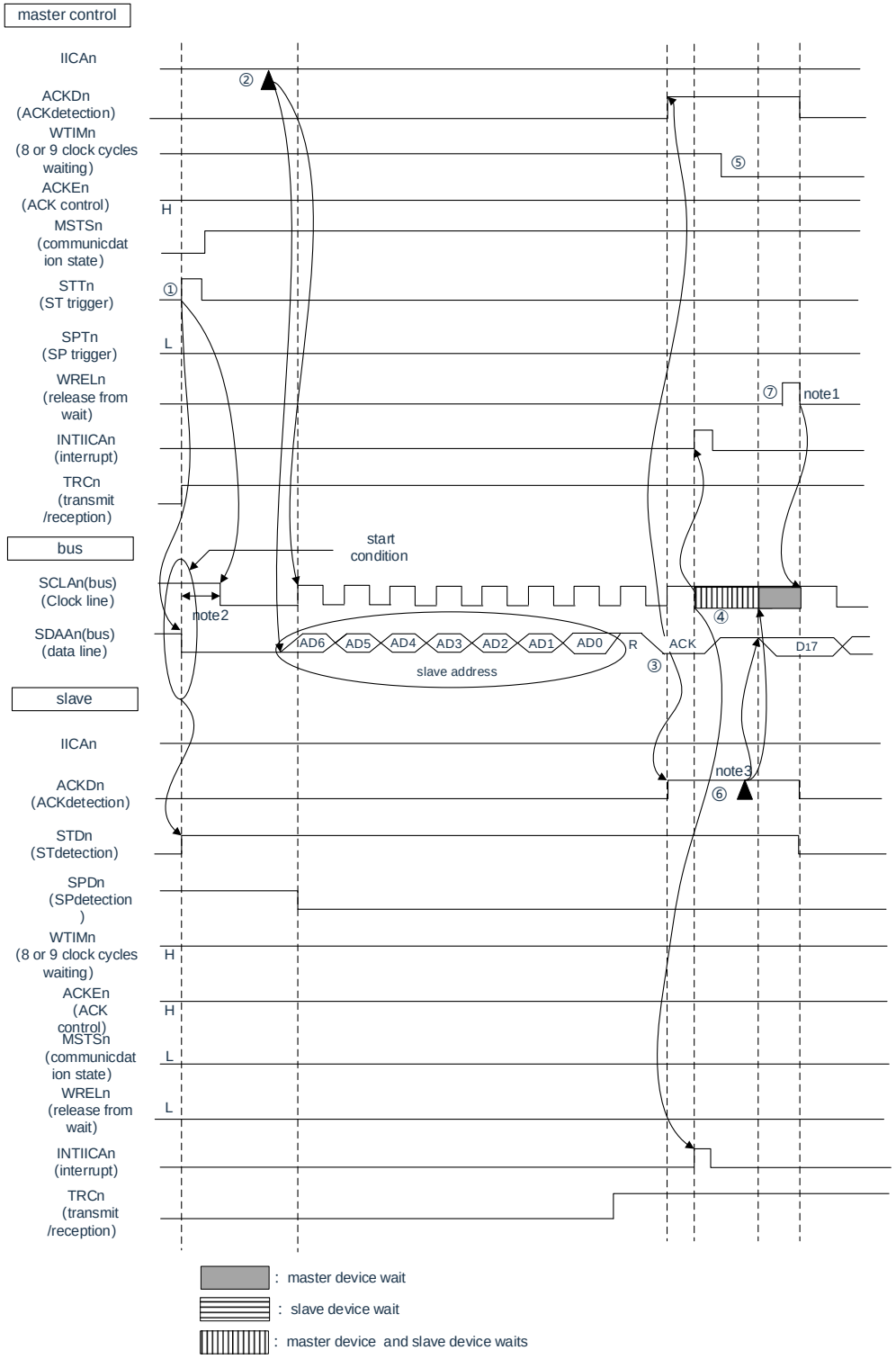
<3> The master device writing the address + R/W (transmission) to the IICA shift register (IICAn) enables the slave address to be transmitted.

Remark: n=0

Figure 14-26: Example of slave to master communication

(Master: selects 8 clocks to wait, slave: selects 9 clocks to wait) (1/3)

(1) Start Condition ~ Address ~ Data



Note 1: To release the master from waiting during transmit, you must write data to the IICAn instead of setting the WRELn bit.

Note 2: Make sure that the time between the fall of the SDAAn pin signal and the fall of the SCLAn pin signal is at least 4.0 μs when specifying standard mode and at least 0.6 μs when specifying fast mode.

Note 3: To release the slave from waiting during transmission, you must write data to the IICn instead of setting the WRELn bit.

Figure 14-26 shows ① to ⑦ of “(1) Start condition~Address~Data” as follows.

① If the master sets the start condition trigger set (STTn=1), the bus data line (SDAAn) drops and the start condition is generated (SDAAn is changed from "1" to "0" by SCLAn=1). Thereafter, if a start condition is detected, the master enters the master communication state (MSTSn=1) and after the hold time elapses the bus clock line drops (SCLAn=0), ending the communication preparation.

② If the master writes address +R (receive) to the IICA shift register n (IICAn), the slave address is sent.

③ On the slave, if the receiving address and the local station address (the value of the SVAn) are the same Note, an ACK is sent to the master through hardware. The master detects ACK on the rising edge of the 9th clock (ACKDn=1).

④ The master generates an interrupt on the falling edge of the 9th clock (INTIICAn: address send end interrupt). Slaves with the same address enter a waiting state (SCLAn=0) and an interrupt (INTIICAn: address matching interrupt) Note.

⑤ The master changes the wait sequence to the 8th clock (WTIMn=0).

⑥ The slave writes and sends data to the IICAn register, relieving the slave of the wait.

⑦ The master relieshes the wait (WRELn=1) and begins data transfer from the slave.

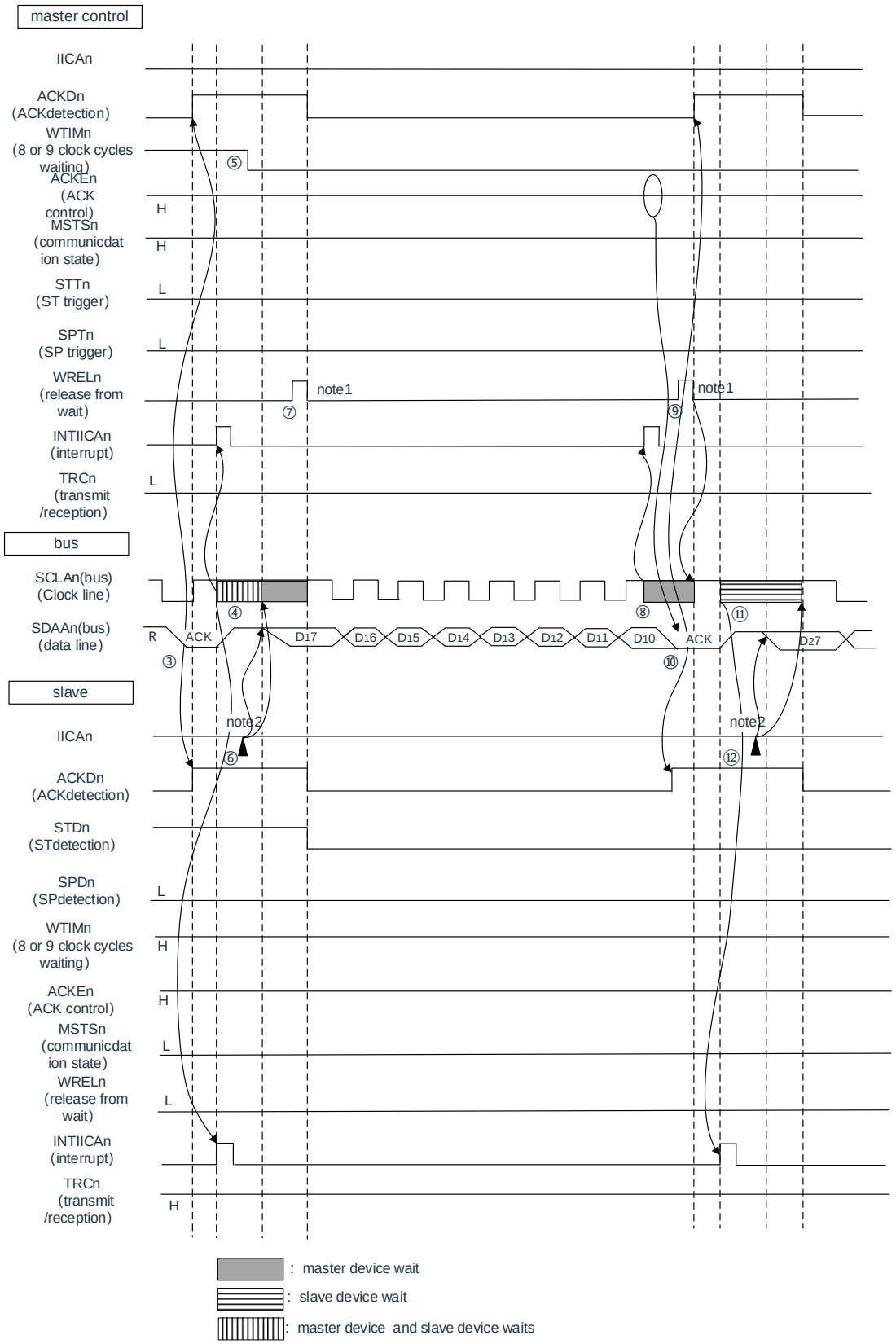
Note: If the sent address and the slave address are different, the slave does not return an ACK (NACK: SDAAn=1) to the master and does not generate an INTIICAn interrupt (address matching interrupt) or enter a waiting state. However, the master generates ANTIICAn interrupts (address send end interrupts) for both ACK and NAK.

Remark:

1. ①~⑲ of Figure 14-26 shows a series of operational steps for data communication via the I^2C bus.
   “(1) Start Condition ~ Address ~ Data” of Figure 14-26 illustrates steps ①~⑦.
   “(2) Address~Data~Data” of Figure 14-26 illustrates steps ③~⑫.
   “(3) Data~Data~Stop Condition” of Figure 14-26 illustrates steps ⑧~⑲.
2. n=0

Figure 14-26: Example of slave to master communication
(Master: selects 8 clocks to wait, slave: selects 9 clocks to wait) (2/3)

(2) Address ~ Data ~ Data



Note 1: To release the master from waiting during transmit, you must write data to the IICAn instead of setting the WRELn bit.

Note 2: To release the slave from waiting during transmission, you must write data to the IICn instead of setting the WRELn bit.

Figure 14-26 shows ③~⑫ of "(2) Address~Data~Data" as follows.

③ On the slave, if the receiving address and the local station address (the value of the SVAn) are the same [Note], the ACK is sent to the master through the hardware. The master detects ACK on the rising edge of the 9th clock (ACKDn=1).

④ The master generates an interrupt on the falling edge of the 9th clock (INTIICAn: address send end interrupt). Slaves with the same address enter a waiting state (SCLAn=0) and an interrupt (INTIICAn: address matching interrupt) [Note].

⑤ The master changes the wait sequence to the 8th clock (WTIMn= 0).

⑥ The slave writes and sends data to the IICA shift register n (IICAn) to release the slave's wait.

⑦ The master relieshes the wait (WRELn=1) and begins data transfer from the slave.

⑧ The master enters a waiting state (SCLAn= 0) on the falling edge of the 8th clock and produces an interrupt (INTIICAn: End of Transmission Interrupt). Because the ACKEn bit of the master is "1", the ACK is sent to the slave through the hardware.

⑨ The master controller reads the received data and cancels the wait (WRELn=1).

⑩ The slave detects ACK (ACKDn=1) on the rising edge of the 9th clock.

⑪ The slave enters a waiting state on the descending edge of the 9th clock (SCLAn = 0) and produces an interrupt (INTIICAn: end-of-transmit interrupt)

⑫ If the slave writes and transmits data to the IICAn register, the slave's wait is released and the data transfer from the slave to the master is started.

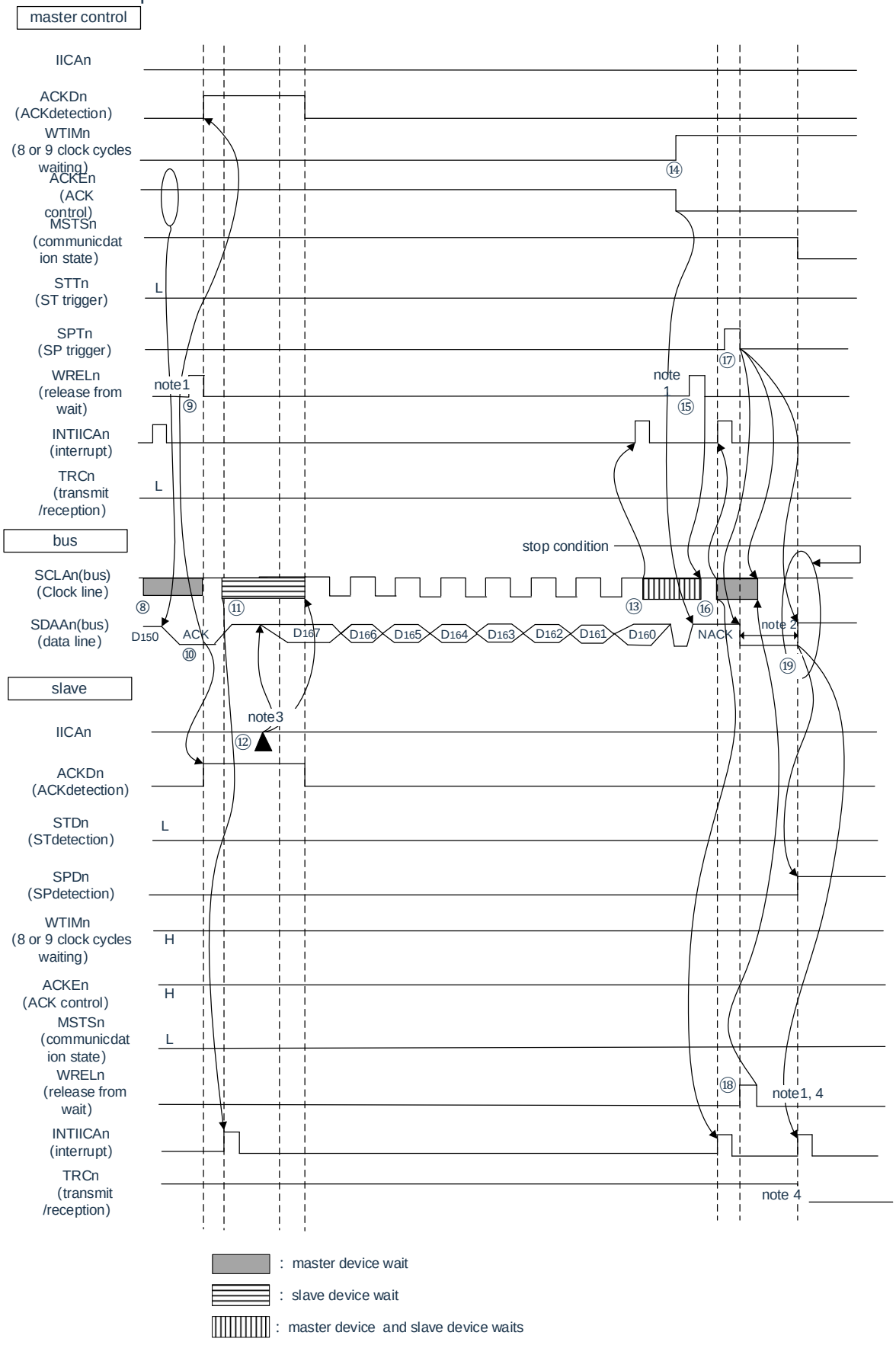
Note: If the sent address and the slave address are different, the slave does not return an ACK (NACK: SDAAn=1) to the master and does not generate an INTIICAn interrupt (address matching interrupt) or enter a waiting state. However, the master generates ANTIICAn interrupts (address send end interrupts) for both ACK and NAK.

Remark:

1. ①~⑲ of Figure 14-26 shows a series of operational steps for data communication via the I^2C bus.
   "(1) Start Condition ~ Address ~ Data" of Figure 14-26 illustrates steps ①~⑦.
   "(2) Address~Data~Data" of Figure 14-26 illustrates steps ③~⑫.
   "(3) Data~Data~Stop Condition" of Figure 14-26 illustrates steps ⑧~⑲.
2. n=0

Figure 14-26: Example of slave to master communication
(Master: selects 8 clocks to wait, slave: selects 9 clocks to wait) (3/3)

(3) Data ~ Data ~ Stop Condition



Note 1: To release the wait, the IICAn must be set to "FFH" or the WRELn bit must be set.

Note 2: After the stop condition is issued, the time from the SCLAn pin signal to generate the stop condition is at least 4.0μs when set to standard mode and at least 0.6μs when set to fast mode.

Note 3: To release the slave from waiting during transmission, you must write data to the IICn instead of setting the WRELn bit.

Note 4: The TRCn bit is cleared if the wait is released by setting the WRELn bit during the slave transmission.

Figure 14-26 shows ⑧~⑲ of “(3) Data~Data~Stop Condition” as follows.

⑧ The master enters a waiting state (SCLAn = 0) on the falling edge of the 8th clock and generates an interrupt (INTIICAn: Transmit End-of-Off). Because the ACKEn bit of the master is "0", the ACK is sent to the slave through the hardware

⑨ The master reads the received data and dismisses the wait (WRELn=1).

⑩ The slave detects ACK (ACKDn=1) on the rising edge of the 9th clock.

⑪ The slave enters a waiting state on the falling edge of the 9th clock (SCLAn= 0) and generates an interrupt (INTIICAn: transmit end interrupt).

⑫ If the slave writes and transmits data to the IICA shift register n (IICAn), the slave's wait is released and the transfer of data from the slave to the master begins.

⑬ The master generates an interrupt (INTIICAn: transmit end interrupt) on the falling edge of the 8th clock and enters a waiting state (SCLAn=0). Because ACK control (ACKEn=1) occurs, the bus data line at this stage becomes low (SDAAn=0)

⑭The master sets the NACK Acknolwdge (ACKEn=0) and changes the wait sequence to the 9th clock (WTIMn=1). If the master relieshes the wait (WRELn=1), the slave detects THEACK (ACKDn=0) on the rising edge of the 9th clock.

⑮ Both the master and slave enter a waiting state (SCLAn=0) on the falling edge of the 9th clock, and both produce an interrupt (INTIICAn: end-of-transmit interrupt).

⑯ If the master issues a stop condition (SPTn=1), the bus data cable (SDAAn=0) is cleared and the master's wait is released. After that, the master is on standby until the bus clock line is set in place (SCLAn=1).

⑰ The slave stops sending after confirming the NAK, in order to end the communication, the wait is released (WRELn=1). If the slave wait is released, the bus clock line is set in place (SCLAn=1).

⑱ If the master confirms that the bus clock line is set (SCLAn=1), the bus data line is set after the stop condition preparation time has elapsed.

⑲ (SDAAn=1), and then issue a stop condition (SDAAn is changed from "0" to "1" by SCLAn=1). If a stop condition is generated, the slave detects the stop condition and generates an interrupt (INTIICAn: Stop Condition Interrupt).

Chapter 15 Linkage Controller EVENTC

15.1 Function of EVENTC

EVENTC links the events output by each peripheral function to each other between the peripheral functions. It can be connected through event chaining without going through the CPU and directly perform collaborative operation between peripheral functions.

EVENTC has the following features:

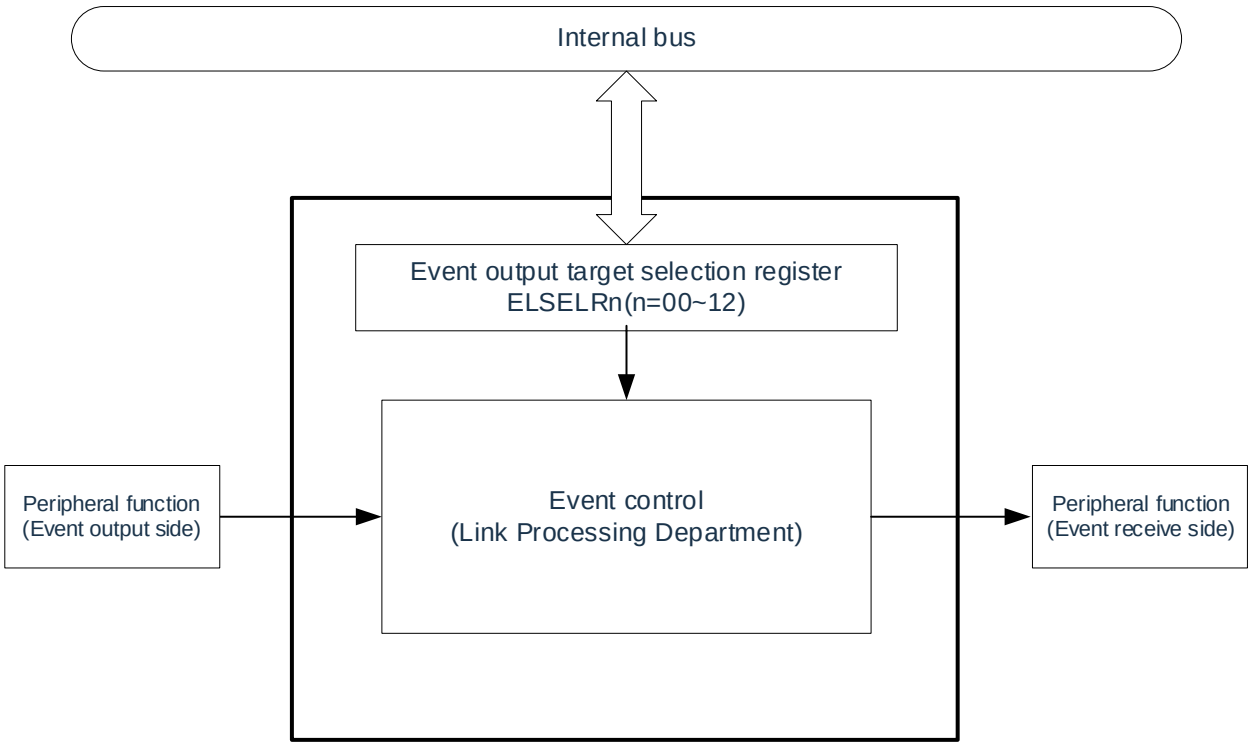
Depending on the product, event signals for 13 peripheral functions can be linked directly to specified peripheral functions.

Depending on the product, the event signal can be used as the starting source for the operation of 1 of the 4 peripheral functions.

15.2 Structure of EVENTC

Block diagram of EVENTC is shown in Figure 15-1.

Figure 15-1: Block diagram of EVENTC



15.3 Control registers

The controller registers are shown in Table 15-1.

Table 15-1: Control registers of EVENTC

Register Name	Symbol
Event output target selection register 00	ELSELR00
Event output target selection register 01	ELSELR01
Event output target selection register 02	ELSELR02
Event output target selection register 03	ELSELR03
Event output target selection register 04	ELSELR04
Event output target selection register 05	ELSELR05
Event output target selection register 06	ELSELR06
Event output target selection register 07	ELSELR07
Event output target selection register 08	ELSELR08
Event output target selection register 09	ELSELR09
Event output target selection register 10	ELSELR10
Event output target selection register 11	ELSELR11
Event output target selection register 12	ELSELR12

15.3.1 Output target selection register n(ELSELRn) (n=00~12)

The ELSELRn register links each event signal to the event receiver peripheral function (link target peripheral function) that runs when an event is accepted. You cannot link multiple event inputs to the same event output destination (event receiver). Otherwise, the event receiver peripheral function may operate indefinitely and may not accept the event signal properly. Also, you cannot set the event link occurrence source and event output destination to the same functionality.

The ELSELRn register must be set during periods when the peripheral functions of all event outputs do not generate an event signal.

Table 15-3 lists the correspondence between ELSELRn (n = 00 to 12) registers and peripheral functions, and Table 15-4 lists the correspondence between values set to ELSELRn (n = 00 to 12) registers and operation of link destination peripheral functions at reception.

Table 15-2: Format of event output target selection register n (ELSELRn)

Address: 40043400H (ELSELR00)~4004340CH (ELSELR12) After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
ELSELRn	0	0	0	0	0	ELSELn2	ELSELn1	ELSELn0

ELSELn2	ELSELn1	ELSELn0	Selection of event links
0	0	0	Disable event links.
0	0	1	Selection of the linked peripheral function 1 operation[Note].
0	1	0	Selection of the linked peripheral function 2 operation[Note].
0	1	1	Selection of the linked peripheral function 3 operation[Note].
1	0	0	Selection of the linked peripheral function 4 operation[Note].
Others:			Settings are disabled.

Note: Refer to “Correspondence between values set to ELSELRn (n = 00 to 12) registers and operation of link destination peripheral functions at reception”.

Table 15-3: Correspondence between ELSELRn (n = 00 to 12) registers and peripheral functions

Register name	Event occurrence source (output source for event input n).	Content
ELSELR00	External interrupt edge detection 0	INTP0
ELSELR01	External interrupt edge detection 1	INTP1
ELSELR02	External interrupt edge detection 2	INTP2
ELSELR03	External interrupt edge detection 3	INTP3
ELSELR04	RTC fixed cycle/alarm clock consistency detection	INTRTC
ELSELR05	Timer40 channel 00 count end/capture end	INTTM00
ELSELR06	Timer40 channel 01 count end/capture end	INTTM01
ELSELR07	Timer40 channel 02 count end/capture end	INTTM02
ELSELR08	Timer40 channel 03 count end/capture end	INTTM03
ELSELR09	Timer41 channel 00 count end/capture end	INTTM10
ELSELR10	Timer41 channel 01 count end/capture end	INTTM11
ELSELR11	Timer41 channel 02 count end/capture end	INTTM12
ELSELR12	Timer41 channel 03 count end/capture end	INTTM13

Table 15-4: Correspondence between values set to ELSELRn (n = 00 to 12) registers and operation of link destination peripheral functions at reception

Bits ELSELRn2 to ELSELRn0 in ELSELRn register	Link destination No.	Link destination peripheral function	Operation when receiving event
001B	1	A/D converter	Start A/D conversion.
010B	2	Timer input of timer40 channel 0 Note1	Delay counter, Measurement of input pulse interval, External event counter
011B	3	Timer input of timer40 channel 1 Note2	Delay counter, Measurement of input pulse interval, External event counter
100B	4	Truncated source for EPWM output control	Forced truncation of pulse output

Note 1: To select the Timer40 channel 0 timer input as the link target peripheral function, you must first set the operating clock of channel 0 to F_{CLK} via Timer Clock Select Register 0 (TPS0), set the noise filter on the TI00 pin to OFF (TNFEN00=0) via Noise Filter Enable Register 1 (NFEN1) and set the timer input used for channel 0 as the event input signal for the link controller via Timer Input Select Register 0 (TIS0).

Note 2: To select the Timer40 channel 1 timer input as the link target peripheral function, you must first set the operating clock of channel 1 to F_{CLK} via Timer Clock Select Register 0 (TPS0), set the noise filter of the TI01 pin to OFF via Noise Filter Enable Register 1 (NFEN1) (TNFEN01=0), and set the timer input used by channel 1 to the event input signal of EVENTC via Timer Input Select Register 0 (TIS0).

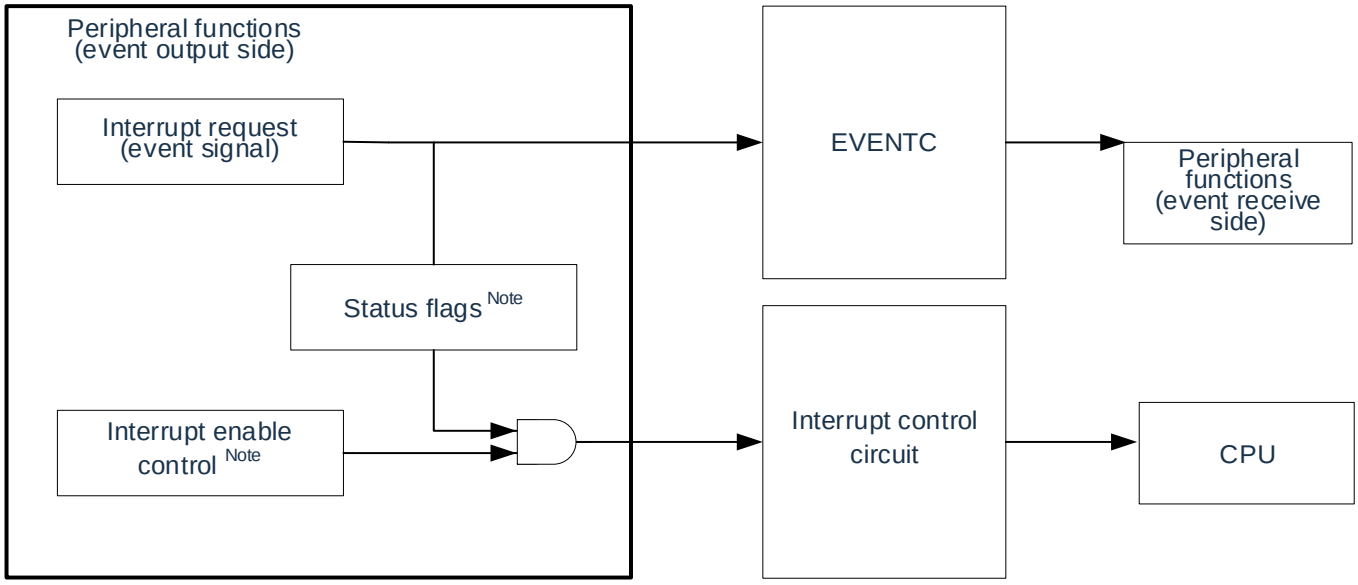
15.4 Operation of EVENTC

The paths used by the event signals generated by each peripheral function as interrupt requests for the interrupt control circuit and the paths used as EVENTC events are independent of each other. Therefore, each event signal is independent of the interrupt control and can be used as an event signal for the operation of the peripheral function at the event receive side.

The relationship between interrupt processing and EVENTC is shown in Figure 15-2. This diagram shows the relationship for a peripheral function with an interrupt request status flag and an interrupt enable bit (control enable or disable).

The operation of the peripheral function that receives an event via EVENTC is based on the operation of the recipient peripheral function after receiving the event, refer to Table 15-4: Correspondence between values set to ELSELRn (n = 00 to 12) registers and operation of link destination peripheral functions at reception

Figure 15-2: Relationship between interrupt handling and EVENTC



The responses of the peripheral functions that receive the events are shown in Table 15-5.

Table 15-5: Response of receiving event peripheral function

Event receive target No.	Function of event link target	Operation after event reception	Response
1	A/D converter	A/D conversion	EventC events become hardware triggers directly for the A/D conversions.
2	Timer input of channel 0 of Timer40	Delay counter input pulse width measure external event counter	Edge detection is performed after 3 or 4 F_{CLK} cycles from the occurrence of an EVENTC event.
3	Timer input of channel 1 of Timer40	Delay counter input pulse width measure external event counter	Edge detection is performed after 3 or 4 F_{CLK} cycles from the occurrence of an EVENTC event.
4	Truncated source for EPWM output control	Forced truncation of pulse output	Becomes a forced truncated state after 2 or 3 EPWM run clock cycles from the occurrence of an EVENTC event.

Note: Some peripheral functions do not have this feature.

Chapter 16 Interrupt Function

The Cortex-M0+ processor has a built-in nested vector interrupt controller (NVIC), which supports up to 32 interrupt request (IRQ) inputs and one non-maskable interrupt (NMI) input, in addition to multiple internal exceptions.

In this system, the interrupt sources for 32 interrupt request (IRQ) inputs and one non-maskable interrupt (NMI) input are handled. This user's manual only explains the processing for this system. Please refer to the user's manual of Cortex-M0+ processor for the functions of the built-in NVIC of Cortex-M0+ processor.

16.1 Types of interrupt functions

There are 2 types of interrupt functions as follows.

(1) Maskable interrupt

This is a mask-controlled interrupt. If the interrupt mask flag register is not opened, the interrupt request will not be responded even if it is generated.

It can generate standby release signals, release deep sleep mode and sleep mode.

Maskable interrupts are divided into external interrupt requests and internal interrupt requests.

(2) Non-maskable interrupt

This is an interrupt that is not controlled by masking. Once an interrupt request is generated, the CPU must respond to it.

16.2 Interrupt sources and structures

Refer to Table 16-1 for a list of interrupt sources.

Table 16-1: List of interrupt sources (1/3)

Interrupt handling	Interrupt source No.	Interrupt resources		Internal/external	Basic structure type Note1
		Name	Trigger		
Maskable	0	INTLVI	Voltage detection Note2	Internal	(A)
	1	INTP0	Detection of pin input edges	External	(B)
	2	INTP1	Detection of pin input edges		
	3	INTP2	Detection of pin input edges		
	4	INTP3	Detection of pin input edges		
	5	INTTM01H	End of count or end of capture for timer channel 01 (for high 8-bit timer operation)	Internal	(A)
	6	INTKR	Key interruption		
	7	INTST1/ INTSSPI10/ INTIIC10 INTIIC10	End of UART1 transmission or buffer empty interrupt/end of SSPI10 transmission or buffer empty interrupt/end of IIC10 transmission		
	8	INTSR1/ INTSSPI11/INTIIC11 INTIIC11	End of UART1 reception/end of SSPI11 transmission or buffer empty interrupt/end of IIC11 transmission		
	9	INTSRE1	A UART1 received communication error occurred		
	10	INTST0/ INTSSPI00/ INTIIC00 INTIIC00	End of UART0 transmission or buffer empty interrupt/end of SSPI00 transmission or buffer empty interrupt/end of transmission of IIC00		
	11	INTSR0/ INTSSPI01/INTIIC01	End of UART0 reception/end of SSPI01 transmission or buffer empty interrupt/end of IIC01 transmission		
	12	INTSRE0	A UART0 receive communication error occurred.		

Note 1: Basic types (A)~(C) correspond to (A)~(C) of Figure 16-1, respectively.

Note 2: This is the case when bit7 (LVIMD) of the voltage detection level register (LVIS) is set to “0”.

Table 16-1: List of interrupt sources (2/3)

Interrupt handling	Interrupt source No.	Interrupt resources		Internal/external	Basic structure type Note1
		Name	Trigger		
Maskable	13	INTSPI	Transmission end interrupt of serial interface SPI	Internal	(A)
	14	Reserved			
	15	Reserved			
	16	INTIICA0	End of IICA0 communication		
	17	INTTM00	Timer channel 00 count end or capture end		
	18	INTTM01	Timer channel 01 count end or capture end		
	19	INTTM02	Timer channel 02 count end or capture end		
	20	INTTM03	Timer channel 03 count end or capture end		
	21	INTAD	A/D conversion end		
	22	INTRTC	Fixed period of the real-time clock/ Alarm clock consistent detection		
	23	INTIT	Detection of interval signals		
	24	INTOCRV	High-speed on-chip oscillator calibration interrupt		
	25	Reserved			
	26	Reserved			
	27	INTTM10	Timer channel 10 count end or capture end		
	28	INTTM11	Timer channel 11 count end or capture end		
	29	INTTM12	Timer channel 12 count end or capture end		
	30	INTTM13	Timer channel 13 count end or capture end		
	31	INTFL	FLASH erase, programming operation end		

Note 1: Basic types (A)~(C) correspond to (A)~(C) of Figure 16-1, respectively.

Table 16-1: List of interrupt sources (3/3)

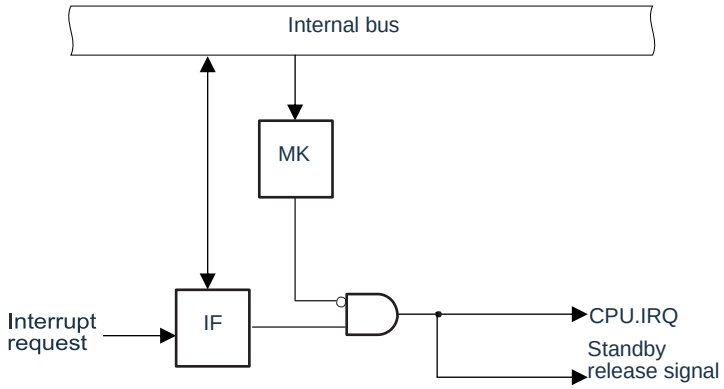
Interrupt handling	Interrupt source No.	Interrupt resources		Internal/external	Basic structure type Note1
		Name	Trigger		
Non-maskable	—	INTWDT	Watchdog timer interval interrupt Note2	Internal	(C)

Note 1: Basic types (A)~(C) correspond to (A)~(C) of Figure 16-1, respectively.

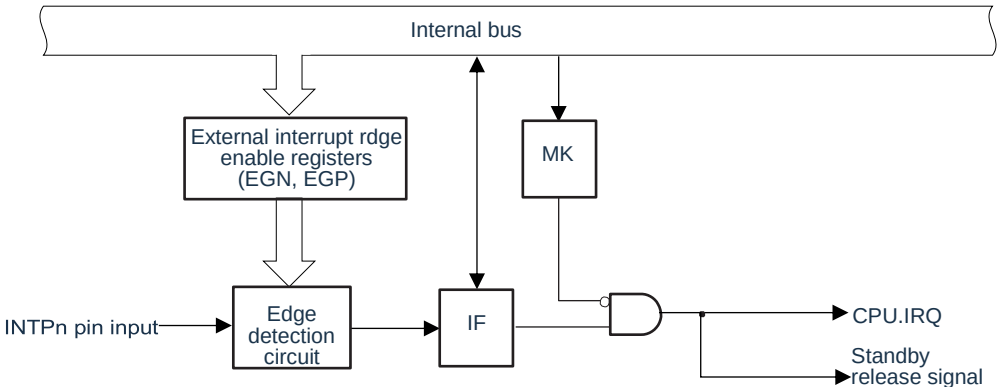
Note 2: This is the case when bit7 (WDTINT) of the option byte (000C0H) is set to "1".

Figure 16-1: Basic structure of interrupt function

(A) Internal maskable interrupt

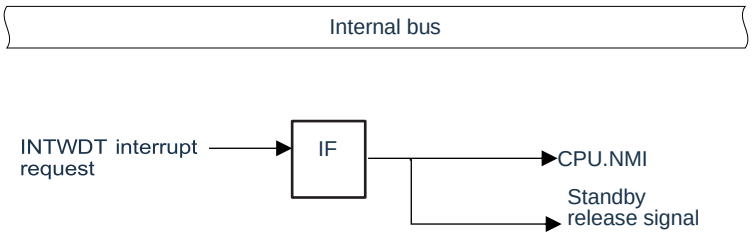


(B) External maskable interrupt (INTPn)



Remark: n=0~3.

(C) Non-maskable interrupt



Remark: The interrupt request flag IF of a non-maskable interrupt has no physical registers and cannot be used to generate an interrupt request by reading or writing registers on the bus.

16.3 Registers for controlling interrupt function

Interrupt function is controlled by the following four registers.

(1) Interrupt request flag register (IF00~IF31)

(2) Interrupt mask flag register (MK00~MK31)

(3) External interrupt rising edge enable register (EGP0)

(4) External interrupt falling edge enable register (EGN0)

16.3.1 Interrupt request flag register (IF00~IF31)

(1) The interrupt request flag is set to "1" by the occurrence of the corresponding interrupt request or execution instruction.

(2) The interrupt request flag is cleared to "0" by generating a reset signal or by executing an instruction.

(3) Set IF00L~IF31L registers by an 8-bit memory manipulation instruction

(4) Or set IF00~IF31 registers by a 32-bit memory manipulation instruction.

(5) After a reset signal is generated, the value of these registers becomes "0000_0000H".

Table 16-2: Format of interrupt request flag register (IFm) (m=0~31)

Address: IF00: 40006000H, IF01: 40006004H, IF02: 40006008H, IF03: 4000600CH
IF04: 40006010H, IF05: 40006014H, IF06: 40006018H, IF07: 4000601CH
IF08: 40006020H, IF09: 40006024H, IF10: 40006028H, IF11: 4000602CH
IF12: 40006030H, IF13: 40006034H, IF14: 40006038H, IF15: 4000603CH
IF16: 40006040H, IF17: 40006044H, IF18: 40006048H, IF19: 4000604CH
IF20: 40006050H, IF21: 40006054H, IF22: 40006058H, IF23: 4000605CH
IF24: 40006060H, IF25: 40006064H, IF26: 40006068H, IF27: 4000606CH
IF28: 40006070H, IF29: 40006074H, IF30: 40006078H, IF31: 4000607CH

Reset value: 0000_0000H R/W

	31	30	29	28	27	26	25	24
	Reserved							
	23	22	21	20	19	18	17	16
	Reserved							
	15	14	13	12	11	10	9	8
	Reserved							
	7	6	5	4	3	2	1	0
IFmL	Reserved							IF

IFmL	Interrupt request flags for interrupt sources numbered 0 to 31[Note1]
0	No interrupt request signal is generated.
1	Generates an interrupt request and is in the interrupt request state.

Note 1: The correspondence between the interrupt source and interrupt request mask register is shown in Table 16-4.

Remark: The correspondence between the interrupt request flag register and CPU.IRQ is shown in Figure 16-2.

16.3.2 Interrupt mask flag register (MK00~MK31)

(1) The interrupt mask flag is set to enable or disable the corresponding maskable interrupt processing.

(2) Set MK00L~MK31L registers by an 8-bit memory manipulation instruction

(3) Or set MK00~MK31 registers by a 32-bit memory manipulation instruction.

(4) After the reset signal is generated, the value of these registers becomes “FFFF_FFFF”.

Table 16-3: Format of interrupt request mask register (MKm) (m=0~31)

Address: MK00: 40006100H, MK01: 40006104H, MK02: 40006108H, MK03: 4000610CH
MK04: 40006110H, MK05: 40006114H, MK06: 40006118H, MK07: 4000611CH
MK08: 40006120H, MK09: 40006124H, MK10: 40006128H, MK11: 4000612CH
MK12: 40006130H, MK13: 40006134H, MK14: 40006138H, MK15: 4000613CH
MK16: 40006140H, MK17: 40006144H, MK18: 40006148H, MK19: 4000614CH
MK20: 40006150H, MK21: 40006154H, MK22: 40006158H, MK23: 4000615CH
MK24: 40006160H, MK25: 40006164H, MK26: 40006168H, MK27: 4000616CH
MK28: 40006170H, MK29: 40006174H, MK30: 40006178H, MK31: 4000617CH

Reset value: FFFF_FFFFH R/W

	31	30	29	28	27	26	25	24
	Reserved							
	23	22	21	20	19	18	17	16
	Reserved							
	15	14	13	12	11	10	9	8
	Reserved							
	7	6	5	4	3	2	1	0
MKmL	Reserved							MKL

MKmL	Interrupt processing control for interrupt sources numbered 0 to 31[Note1]
0	Enable interrupt processing.
1	Disable interrupt processing.

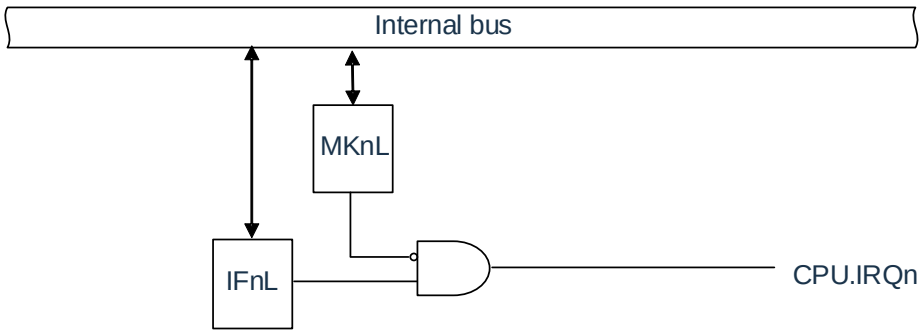
Note 1: The correspondence between the interrupt source and the interrupt request mask register is shown in Table 16-4.

Remark: The correspondence between the interrupt request mask register and CPU.IRQ is shown in Figure 16-2.

Table 16-4: Correspondence between interrupt sources and flag registers

Number	Interrupt source	Interrupt request flag register	Interrupt mask flag register
0	INTLVI	IF00.IFL	MK00.MKL
1	INTP0	IF01.IFL	MK01.MKL
2	INTP1	IF02.IFL	MK02.MKL
3	INTP2	IF03.IFL	MK03.MKL
4	INTP3	IF04.IFL	MK04.MKL
5	INTTM01H	IF05.IFL	MK05.MKL
6	INTKR	IF06.IFL	MK06.MKL
7	INTST1/INTSSPI10/INTIIC10	IF07.IFL	MK07.MKL
8	INTSR1/INTSSPI11/INTIIC11	IF08.IFL	MK08.MKL
9	INTSRE1	IF09.IFL	MK09.MKL
10	INTST0/INTSSPI00/INTIIC00	IF10.IFL	MK10.MKL
11	INTSR0/INTSSPI01/INTIIC01	IF11.IFL	MK11.MKL
12	INTSRE0	IF12.IFL	MK12.MKL
13	INTSPI	IF13.IFL	MK13.MKL
14		IF14.IFL	MK14.MKL
15		IF15.IFL	MK15.MKL
16	INTIICA0	IF16.IFL	MK16.MKL
17	INTTM00	IF17.IFL	MK17.MKL
18	INTTM01	IF18.IFL	MK18.MKL
19	INTTM02	IF19.IFL	MK19.MKL
20	INTTM03	IF20.IFL	MK20.MKL
21	INTAD	IF21.IFL	MK21.MKL
22	INTRTC	IF22.IFL	MK22.MKL
23	INTIT	IF23.IFL	MK23.MKL
24	INTOCRV	IF24.IFL	MK24.MKL
25		IF25.IFL	MK25.MKL
26		IF26.IFL	MK26.MKL
27	INTTM10	IF27.IFL	MK27.MKL
28	INTTM11	IF28.IFL	MK28.MKL
29	INTTM12	IF29.IFL	MK29.MKL
30	INTTM13	IF30.IFL	MK30.MKL
31	INTFL	IF31.IFL	MK31.MKL

Figure 16-2: Correspondence between each flag register and CPU.IRQ



16.3.3 External interrupt rising edge enable register (EGP0), External interrupt falling edge enable register (EGN0)

(1) These registers set the active edges of INTP0 to INTP3.

(2) The EGP0, EGN0 registers are set by 8-bit memory manipulation instructions.

(3) After a reset signal is generated, the value of these registers becomes “00H”.

Table 16-5: Format of external interrupt rising edge enable register (EGP0), external interrupt falling edge enable register (EGN0)

Address: 40045B38H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
EGP0	0	0	0	0	EGP3	EGP2	EGP1	EGP0

Address: 40045B39H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
EGN0	0	0	0	0	EGN3	EGN2	EGN1	EGN0

EGPn	EGNn	Active edge selection of INTPn pin (n=0~11)
0	0	Disable edge detection.
0	1	Falling edge
1	0	Rising edge
1	1	Rising and falling edges

The ports corresponding to the EGPn and EGNn bits are shown in Table 16-6.

Table 16-6: Interrupt request signals corresponding to the EGPn and EGNn bits

Detection enable bit		Interrupt request signal
EGP0	EGN0	INTP0
EGP1	EGN1	INTP1
EGP2	EGN2	INTP2
EGP3	EGN3	INTP3

Notice: If you switch the input port used by the external interrupt function to output mode, an INTPn interrupt may be detected and an INTPn interrupt may be detected. When switching to output mode, the port mode register (PMxx) must be set to "0" after disabling the detection edge (EGPn, EGNn=0, 0).

Remark:

1. For ports of edge detection, refer to “2.1 Port function”.
2. n=0~3.

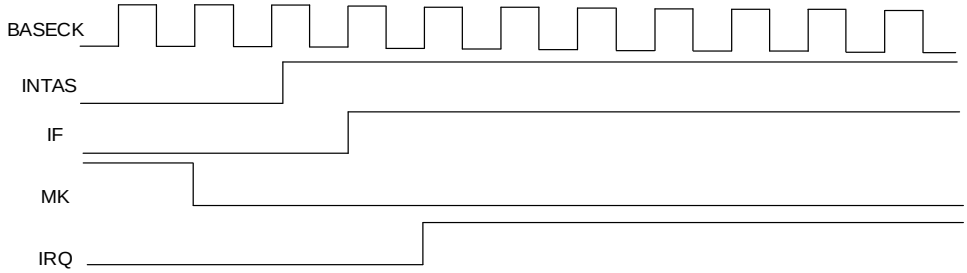
16.4 Operation of interrupt handling

16.4.1 Maskable interrupt request acknowledgment

If an interrupt request flag is set to "1" and the mask (MK) flag for the interrupt request is cleared to "0", the interrupt request is acknowledged and can be passed to the NVIC.

From the time the interrupt request flag is set to 1, to the time the CPU's IRQ is set to 1, it takes only 1 clock.

Figure 16-3: A maskable interrupt request flag transfers to CPU clock map

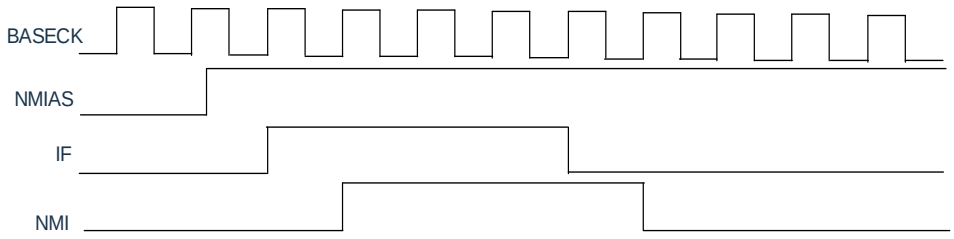


16.4.2 Non-maskable interrupt request acknowledgment

If a non-maskable interrupt request is generated, the interrupt request flag will be set to "1" and passed directly to the NVIC.

From the time the interrupt request flag is set to 1, to the time the CPU's NMI is set to 1, it takes only 1 clock.

Figure 16-4: A non-maskable interrupt request flag transfers to CPU clock map



Chapter 17 Key Interrupt Function

The number of channels for key interrupt input varies by product.

17.1 Function of key interrupt

A key interrupt (INTKR) can be generated by inputting a falling edge to the key interrupt input pins (KR0~KR7).

Table 17-1: Assignment of key interrupt detection pins

Key interrupt pin	Key return mode register (KRM)
KR0	KRM0
KR1	KRM1
KR2	KRM2
KR3	KRM3
KR4	KRM4
KR5	KRM5
KR6	KRM6
KR7	KRM7

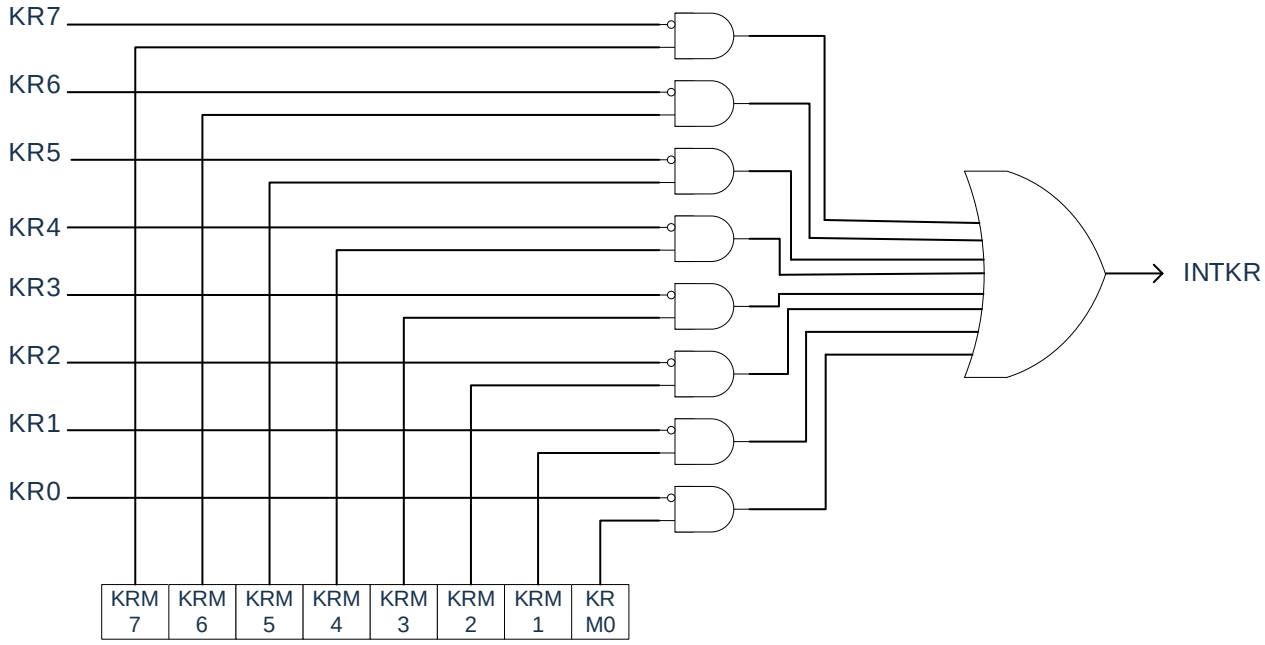
17.2 Structure of key interrupt

Key interrupts are made up of the following hardware.

Table 17-2: Structure of key interrupts

Item	Control register
Control register	Key return mode register (KRM) Port mode register (PMx). Port mode control register (PMCx).

Figure 17-1: Block diagram of key interrupt



Key return mode register

17.3 Registers for controlling key interrupts

The key interrupt function is controlled by the following registers.

(1) Key return mode register (KRM)

(2) Port mode register (PMx)

17.3.1 Key return mode register (KRM)

(1) The KRM0~KRM7 bits control the KR0~KR7 signals.

(2) The KRM register is set by an 8-bit memory manipulation instruction.

(3) After a reset signal is generated, the value of this register becomes “00H”.

Table 17-3: Format of key return mode register (KRM)

Address: 40044B37H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
KRM	KRM7	KRM6	KRM5	KRM4	KRM3	KRM2	KRM1	KRM0

KRMn	Control of key interrupt mode
0	No key interrupt signal is detected.
1	Detects key interrupt signals.

Notice:

1. The internal pull-up resistor can be used by setting the target bit of the pull-up resistor register (PUx) of the key interrupt input pin to “1”.
2. If the object bit of the KRM register is entered low on the input pin of the key interrupt, an interrupt is generated. To ignore this interrupt, the KRM register must be set after interrupt processing is disabled by the interrupt mask flag. The interrupt request flag must be cleared after waiting for the key interrupt input's low-level width (T_{KR}) (see data sheet) to allow interrupt processing.

Remark:

1. Unused pins in key interrupt mode can be used as normal ports.
2. n=0~7.

17.3.2 Port mode register (PMx)

(1) When it is used as the key interrupt input pins (KR0 to KR7), set the PMCxn bit to 0 and the PMxn to 1. The output latches of Pxn at this time may be 0 or 1.

(2) The PMx register can be set by an 8-bit memory manipulation instruction.

(3) Reset signal generation sets this register to FFH.

(4) Use of an on-chip pull-up resistor can be specified in 1-bit units by the pull-up resistor option register (PUx).

(5) Refer to “2.3.1 Port Mode Register (PMxx)” for the format of the port mode register.

Chapter 18 Standby Function

18.1 Standby function

The standby function reduces the operating current of the system, and the following two modes are available.

(1) Sleep mode

Sleep mode is the mode in which the CPU is stopped from running the clock. If the high-speed system clock oscillation circuit, high-speed on-chip oscillator, or subsystem clock oscillation circuit is oscillating before the sleep mode is set, the clocks continue to oscillate. Although this mode does not reduce the operating current to the level of deep sleep mode, it is an effective mode for wanting to restart processing immediately through interrupt requests or if you want to run frequently in intermittent operations.

(2) Deep sleep mode

Deep sleep mode is a mode that stops the oscillation of the high-speed system clock oscillation circuit and the high-speed on-chip oscillator and stops the entire system. The operating current of the CPU can be greatly reduced.

Because deep sleep mode can be dismissed by interrupt requests, intermittent operations can also be performed. However, in the case of the X1 clock, because the wait time to ensure oscillation stability is required when decommissioning the deep sleep mode, it is necessary to select the sleep mode if you need to start processing immediately through the interrupt request.

In either mode, registers, flags, and data memory are all left set to before standby mode, and the output latches and output buffers of the input/output ports are also maintained.

Notice:

1. Deep sleep mode is only available when the CPU is running on the main system clock. When the CPU is running on the subsystem clock, it cannot be set to deep sleep mode. Sleep mode can be used regardless of whether the CPU is running on the main system clock or the subsystem clock.
2. When moving to deep sleep mode, WFI instructions must be executed after stopping peripheral hardware running in the master system clock.
3. To reduce the operating current of the A/D converter, the bit7 (ADCS) of the A/D converter mode register 0 (ADM0) must be placed) and bit0 (ADCE) clear "0", and execute the WFI instruction after stopping the A/D conversion operation.
4. The option byte selects whether to continue or stop oscillation of the low-speed internal oscillator in sleep mode or deep sleep mode. For details, please refer to “Chapter 24 Option Bytes”.

18.2 Sleep mode

18.2.1 Setting of sleep mode

When the SLEEPDEEP bit of the SCR register is 0, execute the WFI instruction and enter sleep mode. In sleep mode, the CPU stops operating, but the values of the internal registers are still maintained, and the peripheral modules remain in the state they were in before they entered sleep mode. The status of peripheral modules, vibrators, etc. in sleep mode is shown in Table 18-1.

Sleep mode can be set regardless of whether the CPU clock before setup is a high-speed system clock, a high-speed on-chip oscillator clock, or a sub-system clock.

Remark: When the interrupt mask flag is “0” (enable interrupt processing) and the interrupt request flag is “1” (generating an interrupt request signal), the interrupt request signal is used to decommission sleep mode. Therefore, even if the WFI command is executed in this case, it does not shift to sleep mode.

Table 18-1: Operation status in sleep mode (1/2)

Item \ Sleep mode setting			Execution of WFI instructions while the CPU is running at main system clock		
			CPU runs on a high-speed on-chip oscillator clock (F_{IH})	CPU runs on a X1 clock (F_X)	CPU runs on an 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 sleep mode was set is retained		
		F_{EXS}			
	Low-speed on-chip oscillator clock	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) WUTMMCK0=1: Osicillates WUTMMCK0=0, and WDTON=0: Stops WUTMMCK0=0, WDTON=1, and WDSTBYON=1: Osicillates WUTMMCK0=0, WDTON=1, and WDSTBYON=0: Stops		
CPU			Operation stopped		
Code flash memory					
RAM			Operation stopped		
Port (latch)			Status before sleep mode was set is retained		
General-purpose timer unit			Operable		
Real-time clock (RTC)					
15-bit interval timer					
Watchdog timer			See “Chapter 10 Watchdog Timer”		
Clock output/buzzer output			Operable		
A/D converter					
General-purpose serial communication unit (SCI)					
Serial interface (SPI)					
Serial interface (IICA)					
Linkage controller			Able to link between operable function blocks.		
Power-on-reset function			Operable		
Voltage detection function					
External interrupt					
CRC Calc. function	High-speed CRC				
	General-purpose CRC		Operation stopped		
SFR guard function			Operation stopped		

Remark:

1. Operation stopped: Operation is automatically stopped before switching to the sleep mode.
2. Operation disabled: Operation is stopped before switching to the sleep mode.
3. 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 18-1: Operation status in sleep mode (2/2)

Item \ Sleep mode setting			Execution of WFI instructions while the CPU is running at subsystem clock	
			CPU runs on a XT1 clock (F_{XT})	CPU runs on an 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)
	Low-speed on-chip oscillation clock	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) WUTMMCK0=1: Osicillates WUTMMCK0=0, and WDTON=0: Stops WUTMMCK0=0, WDTON=1, and WDSTBYON=1: Osicillates WUTMMCK0=0, WDTON=1, and WDSTBYON=0: Stops	
CPU			Operation stopped	
Code flash memory				
RAM			Operation stopped	
Port (latch)			Status before sleep mode was set is retained	
General-purpose timer unit			Operates when the RTCLPC bit is 0 (operation is disabled when the RTCLPC bit is not 0).	
Real-time clock (RTC)			Operable	
15-bit interval timer				
Watchdog timer			See “Chapter 10 Watchdog Timer”	
Clock output/buzzer output			Operates when the RTCLPC bit is 0 (operation is disabled when the RTCLPC bit is not 0).	
A/D converter			Operation disabled	
General-purpose serial communication unit (SCI)			Operates when the RTCLPC bit is 0 (operation is disabled when the RTCLPC bit is not 0).	
Serial interface (SPI)			Operates when the RTCLPC bit is 0 (operation is disabled when the RTCLPC bit is not 0).	
Serial interface (IICA)			Operates when the RTCLPC bit is 0 (operation is disabled when the RTCLPC bit is not 0).	
Linkage controller			Able to link between operable function blocks.	
Power-on-reset function			Operable	
Voltage detection function				
External interrupt				
CRC Calc. function	High-speed CRC		Operation disabled	
	General-purpose CRC		Operation disabled	
SFR guard function			Operation disabled	

Note 1: Operation stopped: Operation is automatically stopped before switching to the sleep mode.

Note 2: Operation disabled: Operation is stopped before switching to the sleep mode.

Note 3: 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

18.2.2 Sleep mode release

The sleep mode can be released by any interrupt or external reset, POR reset, low voltage detection reset, WDT reset, software reset.

(1) Released by interrupts

When a non-maskable interrupt is generated and the interrupt is allowed to be accepted, sleep mode is released and the CPU begins processing interrupt services.

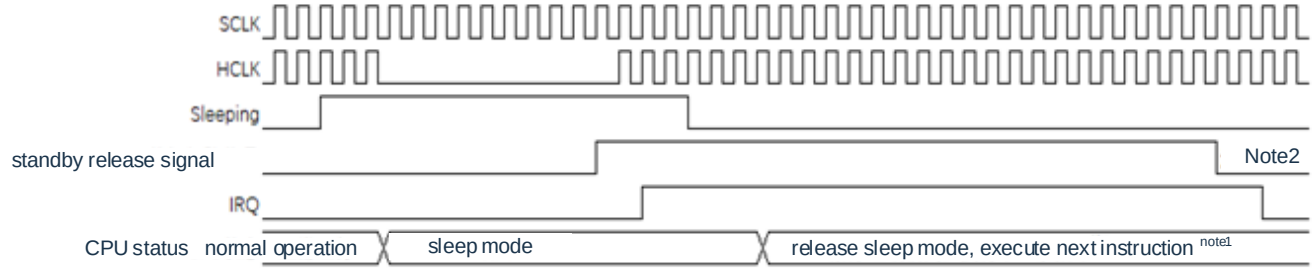


Figure 18-1: Release sleep mode by interrupt requests

Note 1: From the generation of the standby release signal to the release of sleep mode, it takes 16 clocks to start executing the interrupt service program.

Note 2: The standby release signal cannot be cleared by itself, and the register must be cleared. Write registers are typically cleared in interrupt service programs.

Remark: Before entering sleep mode, only the maskable bits corresponding to the interrupts expected to be used to release sleep mode should be cleared.

(2) Cleared by resets

When a reset signal is generated, the CPU is in reset state and the sleep mode is released. As with a normal reset, the program is executed after transferring to the reset vector address.

Figure 18-2: Release sleep mode by resets

Note 1: For reset processing, please refer to “Chapter 19 Reset Function”. For reset processing of power-on reset (POR) and voltage detection (LVD), refer to “Chapter 20 Power-on Reset Circuit”.

18.3 Deep sleep mode

18.3.1 Setting of deep sleep mode

When the SLEEPDEEP bit of the SCR register is 1, the WFI instruction is executed and deep sleep mode is entered. In this mode, the CPU, most of the peripheral modules, and the vibrator operation stops. However, the values of the CPU internal registers, the RAM data, the peripheral modules, the state of the I/O are maintained. The operating status of the peripheral module and the vibrator in deep sleep mode is shown in Table 18-2.

The deep sleep mode can only be set if the CPU clock before setting is as the main system clock.

Remark: When the interrupt mask flag is "0" (allows interrupt processing) and the interrupt request flag is "1" (generating an interrupt request signal), the interrupt request signal is used to dismiss deep sleep mode. Therefore, if the WFI instruction is executed in this case, it is dismissed as soon as it enters deep sleep mode. Returns to run mode after executing the WFI instruction and after a deep sleep mode release time has elapsed.

Table 18-2: Operation status in deep sleep mode

Item \ Deep sleep mode setting			Execution of WFI instructions while the CPU is running at main system clock		
			CPU runs on a high-speed on-chip oscillator clock (F_{IH})	CPU runs on a X1 clock (F_X)	CPU runs on an 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 deep sleep mode was set is retained		
		F_{EXS}			
	Low-speed on-chip oscillator clock	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) WUTMMCK0=1: Osicillates WUTMMCK0=0, and WDTON=0: Stops WUTMMCK0=0, WDTON=1, and WDSTBYON=1: Osicillates WUTMMCK0=0, WDTON=1, and WDSTBYON=0: Stops		
CPU			Operation stopped		
Code flash memory					
RAM					
Port (latch)			Status before deep sleep mode was set is retained		
General-purpose timer unit			Operation disabled		
Real-time clock (RTC)			Operable		
15-bit interval timer					
Watchdog timer			See “Chapter 10 Watchdog Timer”		
Clock output/buzzer output			Operates when the subsystem clock is selected as the clock source for counting and the RTCLPC bit is 0 (Otherwise, operation is disabled)		
A/D converter			Wake-up call can be performed		
General-purpose serial communication unit (SCI)			Operation disabled		
Serial interface (SPI)			Operation disabled		
Serial interface (IICA)			Address matching for wakeup can be performed.		
Linkage controller			Able to link between operable function blocks.		
Power-on-reset function			Operable		
Voltage detection function					
External interrupt					
CRC Calc. function	High-speed CRC		Operation stopped		
	General-purpose CRC				
SFR guard function					

Remark:

1. Operation stopped: Operation is automatically stopped before switching to the deep sleep mode.
2. Operation disabled: Operation is stopped before switching to the deep sleep mode.
3. 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

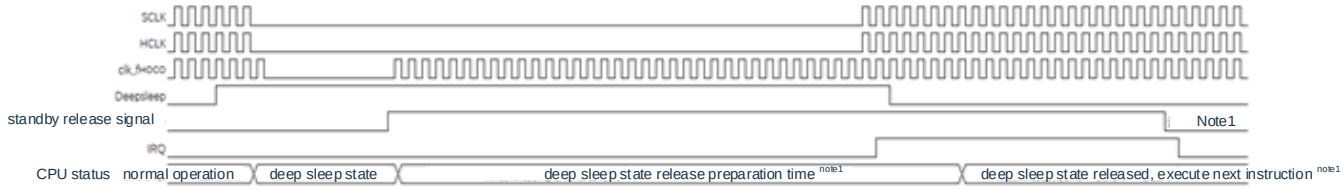
18.3.2 Deep sleep mode release

The deep sleep mode can be released by the following two sources.

(1) Released by non-maskable interrupt requests

If a non-maskable interrupt request occurs, deep sleep mode is released. After the oscillation stabilization time, if the interrupt is allowed to be accepted, the vector interrupt is processed. If the interrupt acceptance is disabled, the next address is executed.

Figure 18-3: Release deep sleep mode by interrupt requests



Note 1: Standby release signal: For details of the standby release signal, please refer to “Figure 16-1 Basic structure of interrupt function”.

Note 2: Deep sleep release preparation time.

① When the CPU clock is a high-speed on-chip oscillation clock or an external clock input before entering deep sleep mode: at least 20us.

② When entering deep sleep mode before the CPU clock is a high-speed system clock (X1 oscillation): at least 20us and a longer time in the oscillation settling time (set by OSTS).

Note 3: Wait: 14 clocks are required from when the time CPU.IRQ is valid to the interrupt service program starts.

Notice:

1. Before entering sleep mode, only the mask bits corresponding to the interrupts expected to be used to release sleep mode should be cleared to zero.
2. When the CPU is running at high speed system clock (X1 oscillation) and to shorten the oscillation stabilization time after the deep sleep mode is released, the CPU clock must be temporarily switched to the high-speed on-chip oscillator clock before the WFI instruction is executed.
3. The oscillation accuracy of the high-speed on-chip oscillator clock varies steadily depending on temperature conditions and during deep sleep mode.

(2) Released by generating reset signals

The deep sleep mode is released by generating a reset signal. Then, as with a normal reset, the program is executed after transferring to the reset vector address.

Figure 18-4: Release deep sleep mode by resetting

Note 1: For reset processing, see “Chapter 19 Reset Function”. For reset processing of power-on reset (POR) and voltage detection (LVD), see “Chapter 21 Power-on Reset Circuit”.

18.4 Deep sleep mode with partial power down

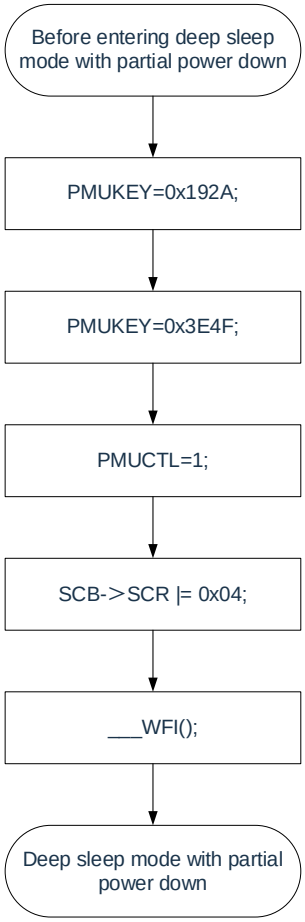
18.4.1 Setting of deep sleep mode with partial power down

The deep sleep mode with partial power loss is a deep sleep mode that further saves power consumption by turning off some peripheral power supplies on the basis of deep sleep mode. Enter the partial power-down deep sleep mode needs to configure the PWDNEEN bit of the PMUCTL register, the control bit is written to the power supply mode control protection register (PMUKEY) protection, when the deep sleep mode of partial power down requires reinitialization of the power-down periphery before it can re-operate normally, please refer to Table 18-3 The operation status in the deep sleep mode of the partial power-down is required for details.

When the SCR register has a SLEEPDEEP bit of 1 and the PMUCTL register has a PWDNEEN bit of 1, executing the WFI command can enter a partially powered-down deep sleep mode. In this mode, the CPU and the transmitter stop functioning, and most peripheral modules are powered off. However, the value of the CPU's internal registers, RAM data, the state of the I/O is maintained. The operating status of the peripheral module and the oscillator in the deep sleep mode of partial power failure is shown in Table 18-3.

A partially powered-down deep sleep mode can only be set if the CPU clock before setting is the main system clock. The PWDNEEN bit of the PMUCTL register is controlled with reference to the section 4.3.11 Power Supply Mode Control Protection Register (PMUKEY) and the 4.3.12 Power Supply Mode Control Register (PMUCTL).

Figure 18-5: Flowchart of entering deep sleep mode with partial power down



Remark: When the interrupt mask flag is "0" (interrupt processing is allowed) and the interrupt request flag is "1" (interrupt request signal is generated), the interrupt request signal is used to release the deep sleep mode. Therefore, if the WFI instruction is executed in this case, it is released as soon as the deep sleep mode is entered, and the partial power-down mode is not entered in this case. The WFI command is executed and returns to the operation mode after the deep sleep mode release time.

Table 18-3: Operation status in deep sleep mode with partial power down

Item \ Deep sleep mode with partial power down setting			Execution of WFI instructions while the CPU is running at main system clock		
			CPU runs on a high-speed on-chip oscillator clock (F_{IH})	CPU runs on a X1 clock (F_X)	CPU runs on an external main system clock (F_{EX})
System clock			Clock supply to the CPU is stopped, but power supply is maintained.		
	Main system clock	F_{IH}	Operation stopped		
		F_X			
		F_{EX}			
	Subsystem clock	F_{XT}	Maintains the state before it was set to partial power-down deep sleep mode.		
		F_{EXS}			
	Low-speed on-chip oscillator clock	F_{IL}	Set by bit0 (WDSTBYON) and bit4 (WDTON) of the option byte (000C0H) and the WUTMMCK0 bit of the Subsystem Clock Supply Mode Control Register (OSMC). WUTMMCK0=1: Oscillates WUTMMCK0=0, and WDTON=0: Stops WUTMMCK0=0, WDTON=1, and WDSTBYON=1: Oscillates WUTMMCK0=0, WDTON=1, and WDSTBYON=0: Stops		
CPU			Operation stopped, power supply maintained.		
Code flash memory					
RAM			RAM power supply maintained.		
Port (latch)			Maintains the state before it was set to partial power-down deep sleep mode.		
General-purpose timer unit			Operation disabled, power supply stopped.		
Real-time clock (RTC)			Operation enabled, power supply maintained.		
15-bit interval timer					
Watchdog timer			See “Chapter 10 Watchdog Timer”		
Clock output/buzzer output			Operation stopped, power supply stopped.		
A/D converter			Operation stopped, power supply stopped.		
General-purpose serial communication unit (SCI)			Operation disabled, power supply stopped.		
Serial interface (SPI)			Operation disabled, power supply stopped.		
Serial interface (IICA)			Operation disabled, power supply stopped.		
Linkage controller			Operation disabled, power supply stopped.		
Power-on-reset function			Operation enabled, power supply maintained.		
Voltage detection function					
External interrupt					
CRC Calc. function	High-speed CRC		Operation stopped, power supply stopped.		
	General-purpose CRC				
SFR guard function					

Remark:

1. Operation stopped: Operation is automatically stopped before switching to the deep sleep mode with partial power down.
2. Operation disabled: Operation is stopped before switching to the deep sleep mode with partial power down.
3. Power supply maintained: Maintains module power supply after switching to the deep sleep mode with partial power down.
4. Power supply stopped: After transferring to a deep sleep mode with partial power down, the power supply to the module is stopped, and the module needs to be re-initialized after the mode is released.
5. 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

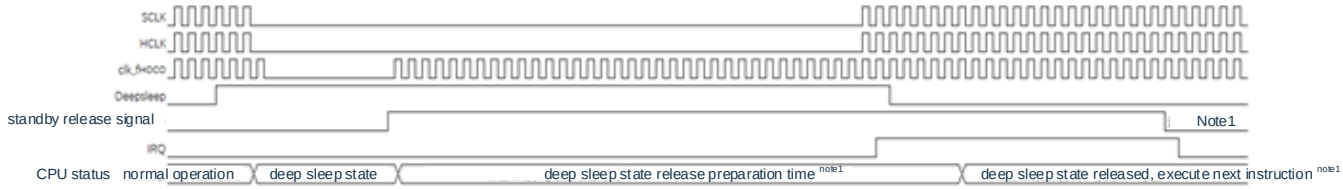
18.4.2 Release deep sleep mode with partial power down

Release the deep sleep mode with partial power down by the following 2 methods.

(1) Release deep sleep mode with partial power-down via interrupt requests

If INTP0-3, INTKR, INTRTC, INTIT, INTLVI and INTWDT interrupt requests are generated, the deep sleep mode with partial power down can be released. After the oscillation stabilization time, if it is in the state of enabling interrupt reception, it will process the vector interrupt. If the interrupt is disabled, the next address instruction is executed.

Figure 18-6: Release deep sleep mode by interrupting requests



Note 1: Standby release signals: Interrupt request signals for INTP0-3, INTKR, INTRTC, INTIT, INTLVI, and INTWDT.

Note 2: When the deep sleep state of partial power-down is ready to be released, it is necessary to re-initialize the peripheral functions in order to ensure that the program continues to run normally.

Note 3: Before entering the deep sleep mode with partial power-down, only the mask bit corresponding to the interrupt that is expected to be used to release the sleep mode should be cleared to zero.

(2) Release by generating a reset signal

The deep sleep mode with partial power-down is released by generating a reset signal. Then, as with a normal reset, the program is executed after switching to the reset vector address.

Figure 18-7: Release deep sleep mode with partial power-down by resetting

Note 1: For reset processing, refer to “Chapter 19 Reset Function”. Refer to “Chapter 20 Power-On Reset Circuit” and “Chapter 21 Voltage Detection Circuit” for reset processing of the power-on reset (POR) circuit and voltage detection (LVD) circuit.

Chapter 19 Reset Function

The following six operations are available to generate a reset signal.

(1) External reset input via RESETB pin.

(2) Internal reset is generated by programmed runaway detection of the watchdog timer.

(3) Internal reset is generated by comparison of the power supply voltage and the detection voltage of the power-on reset (POR) circuit.

(4) An internal reset is generated by comparing the supply voltage of the voltage detection circuit (LVD) with the detection voltage.

(5) An internal reset is generated by setting the system reset request register bit (AIRCR.SYSRESETREQ) to 1.

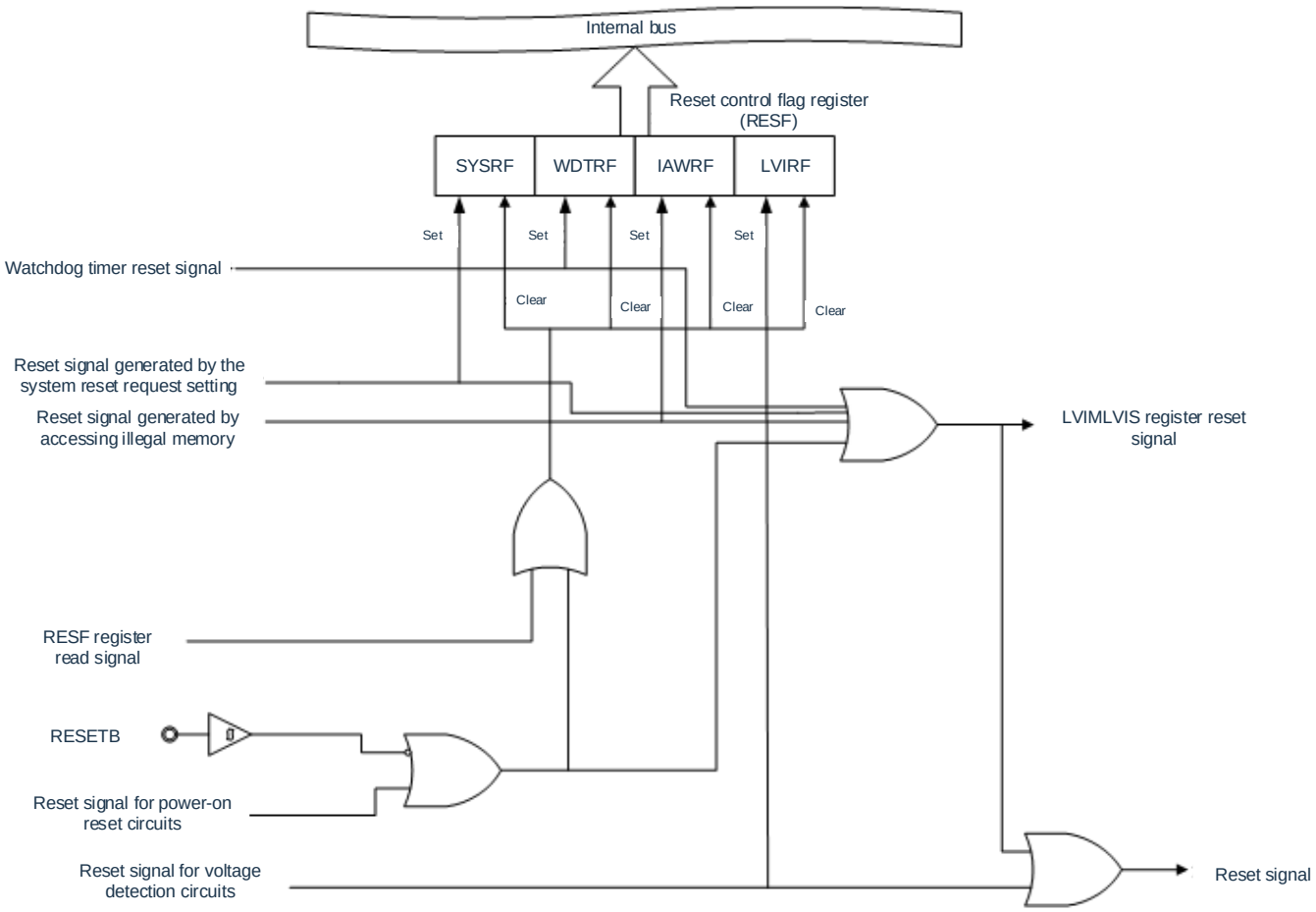
(6) Internal reset due to illegal memory access.

Internal reset is the same as external reset, and after generating a reset signal, the program is executed starting from the user-defined program start address. When a low level is supplied to the RESETB pin, or a program runaway is detected by the watchdog timer, or a voltage is detected in the POR and LVD circuits, or the system reset request bit is set, or an illegal memory access occurs, a reset is generated and the hardware changes to the state shown in Table 19-1.

Notice:

1. When performing an external reset, the RESETB pin must be held low for at least 10us. If an external reset is performed while the supply voltage is rising, the power must be turned on after supplying a low level to the RESETB pin, and must be held low for at least 10us over the operating voltage range shown in the AC Characteristics of the User's Manual, and then be supplied with a high level.
2. Stop oscillating of X1 clock, XT1 clock, high-speed on-chip oscillator clock, and low-speed internal oscillator clock during the reset signal. The inputs to the external master system clock and the external subsystem clock are invalid.
3. If a reset occurs, each SFR is initialized so that the pins change to the following states.
   ① P00: Low during external reset or POR reset, high during other reset periods and during normal operation.
   ② P20,P21,P36,P37: High during external reset or POR reset. High during other resets and after receiving a reset (connect internal pull-up resistor).
   ③ Other ports: High impedance during reset and after receiving reset.

Figure 19-1: Block diagram of reset function

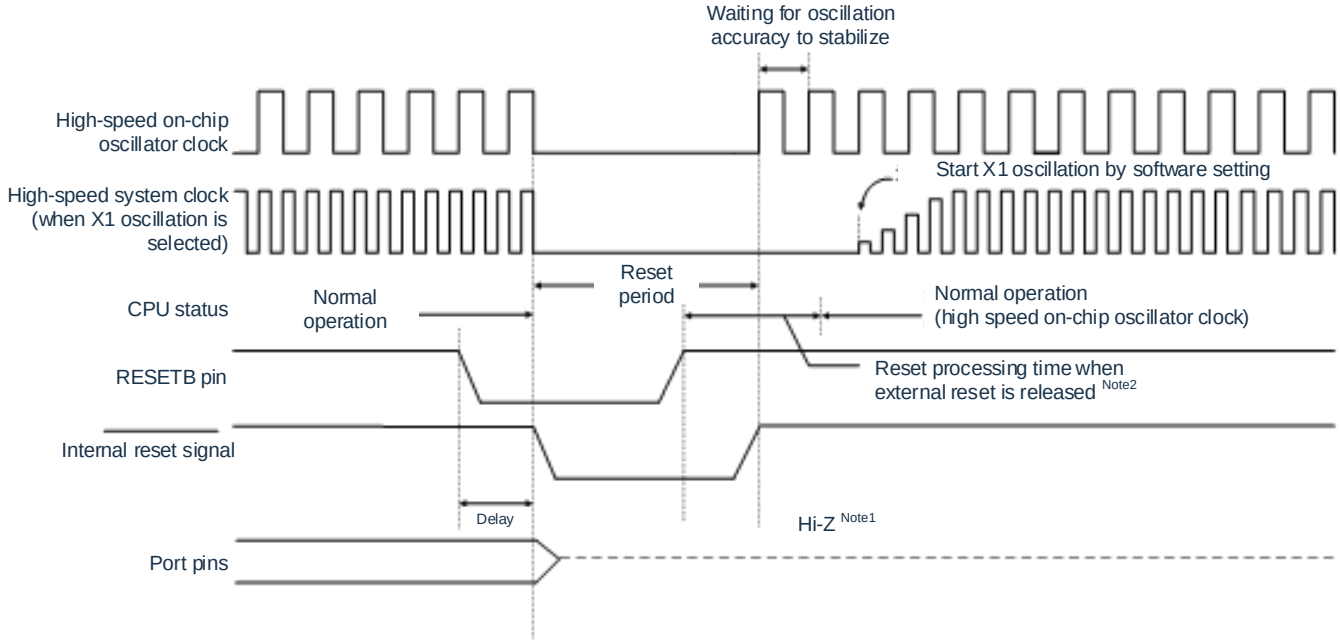


Remark:

1. An internal reset of the LVD circuit does not reset the LVD circuit.
2. LVIM: Voltage detection register
3. LVIS: Voltage detection level register

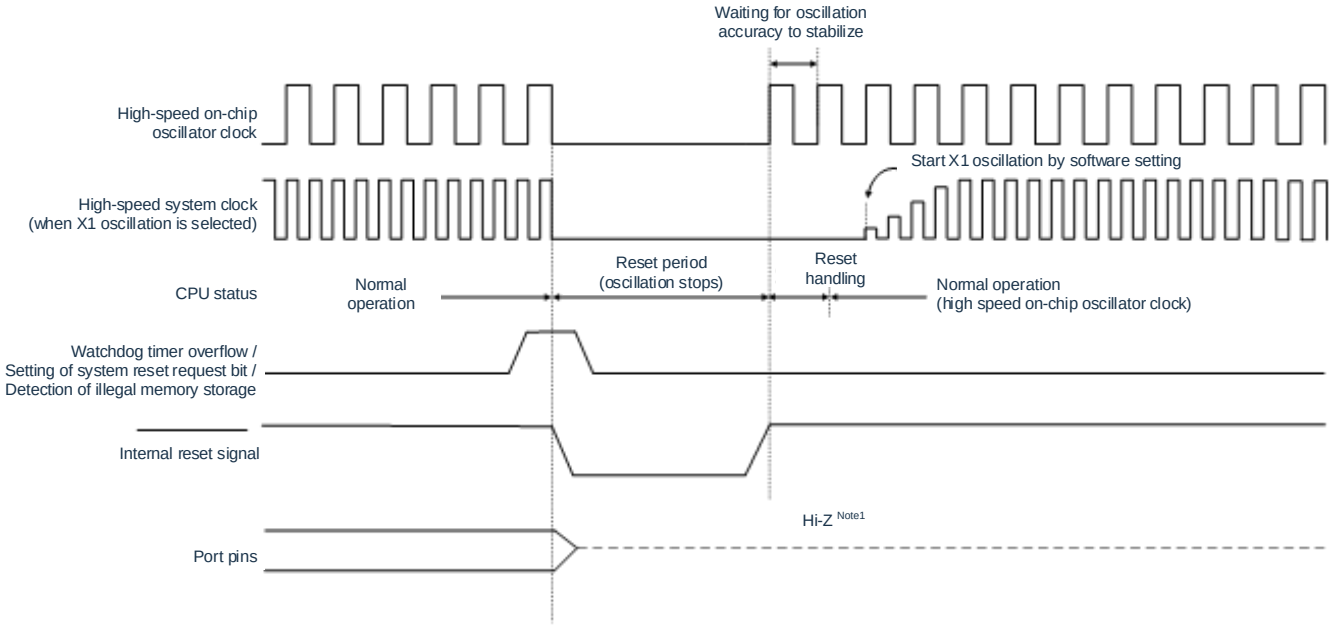
When the RESETB pin is input low, a reset is generated. The reset state is then released if the RESETB lead is entered high and the program begins with a high-speed on-chip oscillator clock after the reset process is complete.

Figure 19-2: Timing of RESETB input



For reset caused by overflow of the watchdog timer, setting of the system reset request bit, or detection of illegal memory access, the reset state is automatically released, and program execution starts with the high-speed on-chip oscillator clock after the reset processing is completed.

Figure 19-3: Reset timing due to watchdog timer overflow, system reset request bit setting, or illegal memory access detection



Note 1: P00,P20,P20,P36,P37 change to the following states.

① P00: Low during external reset or POR reset, high during other resets and during normal operation.

② P20,P21,P36,P37: High impedance during external reset or POR reset. High during other resets and after receiving a reset (internal pull-up resistor connected).

Remark: Reset is performed when an internal reset occurs, and the watchdog timer is no exception.

For the reset generated by the voltage detection of the POR circuit and the LVD circuit, if $V_{DD} \geqslant V_{POR}$ or $V_{DD} \geqslant V_{LVD}$ is satisfied after the reset, the reset state is released and the program is executed with a high-speed on-chip oscillator clock after the reset is processed. For details, please refer to “Chapter 20 Power-on Reset Circuit” and “Chapter 21 Voltage Detection Circuit”.

Remark:

1. V_{POR}: POR supply voltage rising detection voltage
2. V_{LVD}: LVD detection voltage

Table 19-1: Operation status during resetting

<table>
<tr><th colspan="3">Item</th><th>Reset period</th></tr>
<tr><td colspan="3">System clock</td><td>Clock supply to the CPU is stopped</td></tr>
<tr><td rowspan="5"></td><td rowspan="3">Main system clock</td><td>F_{IH}</td><td>Operation stopped</td></tr>
<tr><td>F_X</td><td>Operation stopped (X1 pin and X2 pin are in input port mode).</td></tr>
<tr><td>F_{EX}</td><td>Clock input is invalid (pin is in input port mode).</td></tr>
<tr><td rowspan="2">Subsystem clock</td><td>F_{XT}</td><td>Operable</td></tr>
<tr><td>F_{EXS}</td><td>Clock input is invalid (pin is in input port mode).</td></tr>
<tr><td>Low-speed on-chip oscillator clock</td><td>F_{IL}</td><td rowspan="2">Operation stopped</td></tr>
<tr><td colspan="3">CPU</td></tr>
<tr><td colspan="3">Code flash memory</td><td>Operation stopped</td></tr>
<tr><td colspan="3">RAM</td><td>Operation stopped</td></tr>
<tr><td colspan="3">Port (latch)</td><td>High impedance[Note1]</td></tr>
<tr><td colspan="3">General-purpose timer unit</td><td rowspan="8">Operation stopped</td></tr>
<tr><td colspan="3">Real-time clock (RTC)</td></tr>
<tr><td colspan="3">15-bit interval timer</td></tr>
<tr><td colspan="3">Watchdog timer</td></tr>
<tr><td colspan="3">Clock output/buzzer output</td></tr>
<tr><td colspan="3">A/D converter</td></tr>
<tr><td colspan="3">General-purpose serial communication unit (SCI)</td></tr>
<tr><td colspan="3">Serial interface (IICA)</td></tr>
<tr><td colspan="3">Power-on-reset function</td><td>Detection operations can be performed.</td></tr>
<tr><td colspan="3">Voltage detection function</td><td>Operation is possible in the case of an LVD reset and stopped in the case of other types of reset.</td></tr>
<tr><td colspan="3">External interrupt</td><td rowspan="5">Operation stopped</td></tr>
<tr><td colspan="3">Key interrupt function</td></tr>
<tr><td rowspan="2">CRC Calc. function</td><td colspan="2">High-speed CRC</td></tr>
<tr><td colspan="2">General-purpose CRC</td></tr>
<tr><td colspan="3">SFR guard function</td></tr>
</table>

Note 1: P00,P20,P20,P36,P37 change to the following states.

① P00: Low during external reset or POR reset, high during other reset periods and during normal operation.

② P20,P21,P36,P37: High during external reset or POR reset. High during other resets and after receiving a reset.

Remark: 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

19.1 Registers for confirming the reset source

19.1.1 Reset control flag register (RESF)

The CMS32L032 microcontroller has multiple internal reset generation sources. The Reset Control Flag register (RESF) holds the reset source where the reset request occurs. The RESF register can be read by an 8-bit memory manipulation instruction.

The SYSRF, WDTRF, IAWRF, LVIRF flags are cleared by inputting RESETB, resetting the power-on reset (POR) circuit, and reading the RESF register. To determine the reset source, the value of the RESF register must be saved to any RAM and then determined by its RAM value.

Table 19-2: Format of reset control flag register (RESF)

Address: 40020440H　After reset: undefined[Note]　R

Symbol	7	6	5	4	3	2	1	0
RESF	SYSRF	0	0	WDTRF	0	0	IAWRF	LVIRF

SYSRF	Internal reset request resulting from the system reset request bit being set
0	No internal reset request is generated or the RESF register is cleared.
1	Generates an internal reset request.

WDTRF	Internal reset request generated by the watchdog timer (WDT)
0	No internal reset request is generated or the RESF register is cleared.
1	Generates an internal reset request.

IAWRF	Access to internal reset requests generated by illegal memory
0	No internal reset request is generated or the RESF register is cleared.
1	Generates an internal reset request.

LVIRF	Internal reset request generated by the voltage detection circuit (LVD)
0	No internal reset request is generated or the RESF register is cleared.
1	Generates an internal reset request.

Note: It varies depending on the reset source, please refer to Table 19-3.

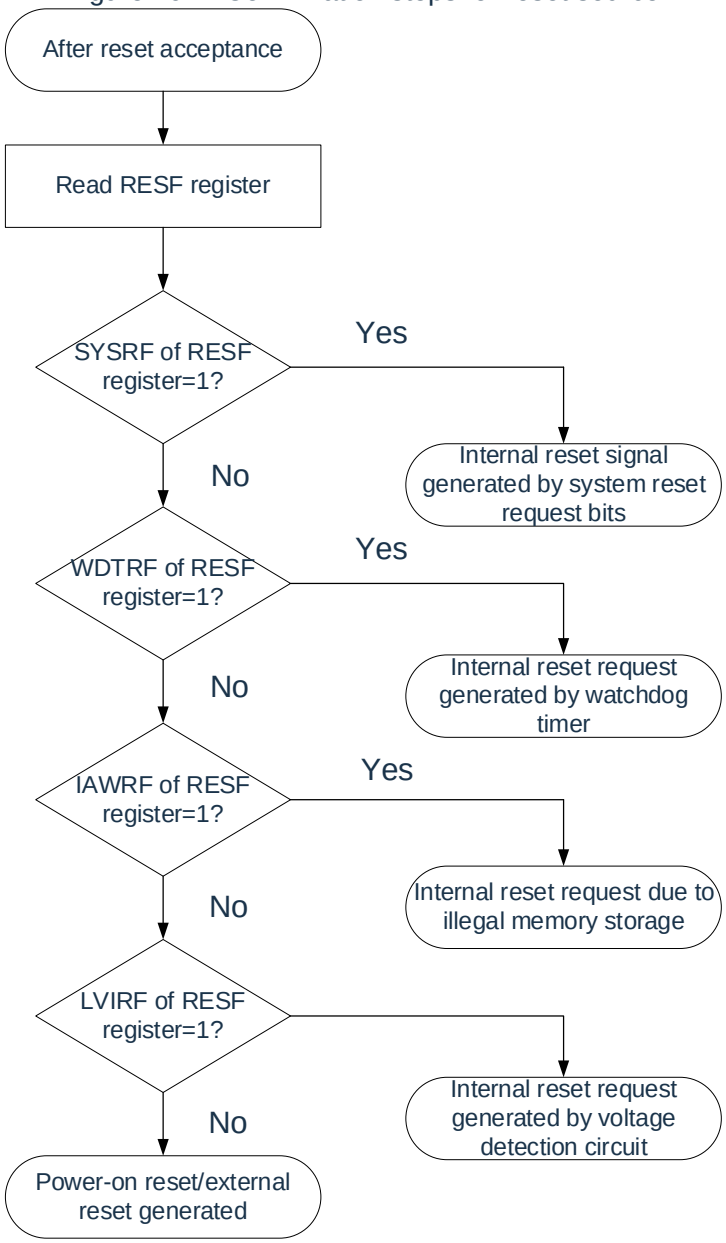
The status of the RESF register when a reset request occurs is shown in Table 19-3.

Table 19-3: RESF register status when a reset request occurs

Reset resource / Flag	RESETB input	Reset by POR	Reset generated by system reset request bit set	Reset generated by WDT	Reset generated by accessing illegal memory	Reset generated by LVD
SYSRF	Cleared to “0”	Cleared to “0”	Set to “1”	Held	Held	Held
WDTRF			Held	Set to “1”		
IAWRF					Set to “1”	
LVIRF					Held	Set to “1”

The confirmation steps for resetting the source are shown in Figure 19-4.

Figure 19-4: Confirmation steps for reset source



Remark: The flow described above is an example of the procedure for checking.

Chapter 20 Power-On Reset Circuit

20.1 Function of power-on reset circuit

The power-on reset circuit (POR) has the following functions.

(1) Generates internal reset signal at power on. The reset signal is released when the supply voltage (V_{DD}) exceeds the detection voltage (V_{POR}). Note that the reset state must be retained until the operating voltage becomes in the range defined in AC Characteristics. This is done by utilizing the voltage detection circuit or controlling the externally input reset signal.

(2) Compares supply voltage (V_{DD}) and detection voltage (V_{PDR}), generates internal reset signal when $V_{DD} < V_{PDR}$. Note that, after power is supplied, this should be placed in the deep sleep 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 AC Characteristics. When restarting the operation, make sure that the operation voltage has returned within the range of operation.

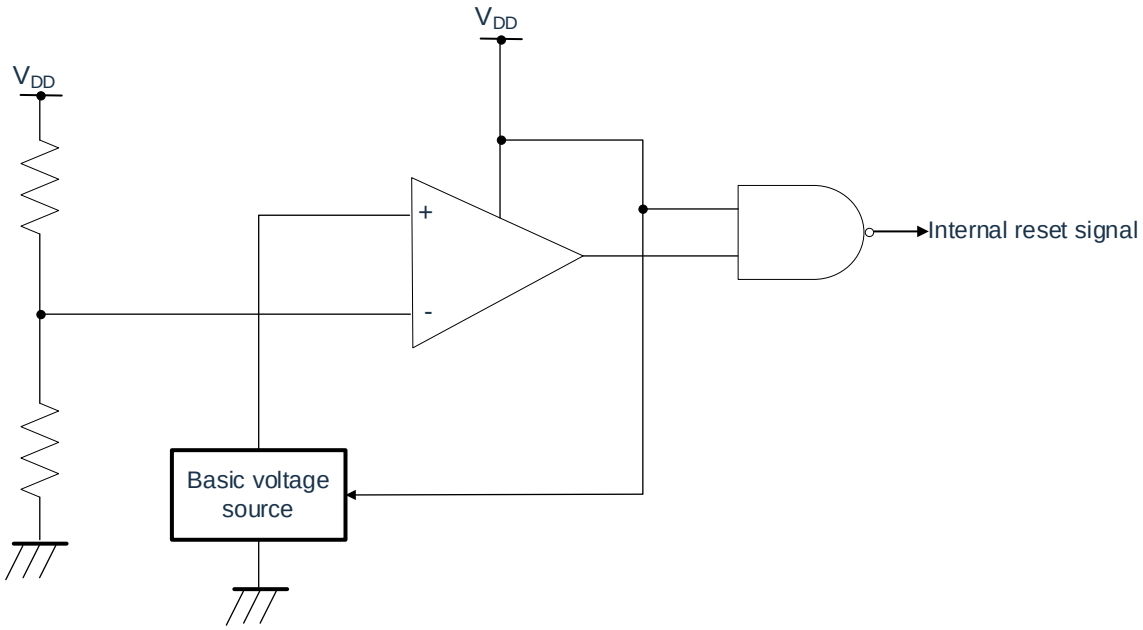
Remark:

1. When the power-on reset circuit generates an internal reset signal, the reset control flag register (RESF) is cleared to “00H”.
2. The CMS32L032 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), system reset request bit setting, 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), system reset request bit setting, or illegal-memory access. For details of RESF register, refer to “Chapter 19 Reset Function”.
3. V_{POR}: POR power supply rise detection voltage; V_{PDR}: POR power supply fall detection voltage
4. For details, refer to the POR circuit characteristics in the data sheet.

20.2 Structure of power-on reset circuit

The block diagram of the power-on reset circuit is shown in Figure 20-1.

Figure 20-1: Block diagram of power-on reset circuit

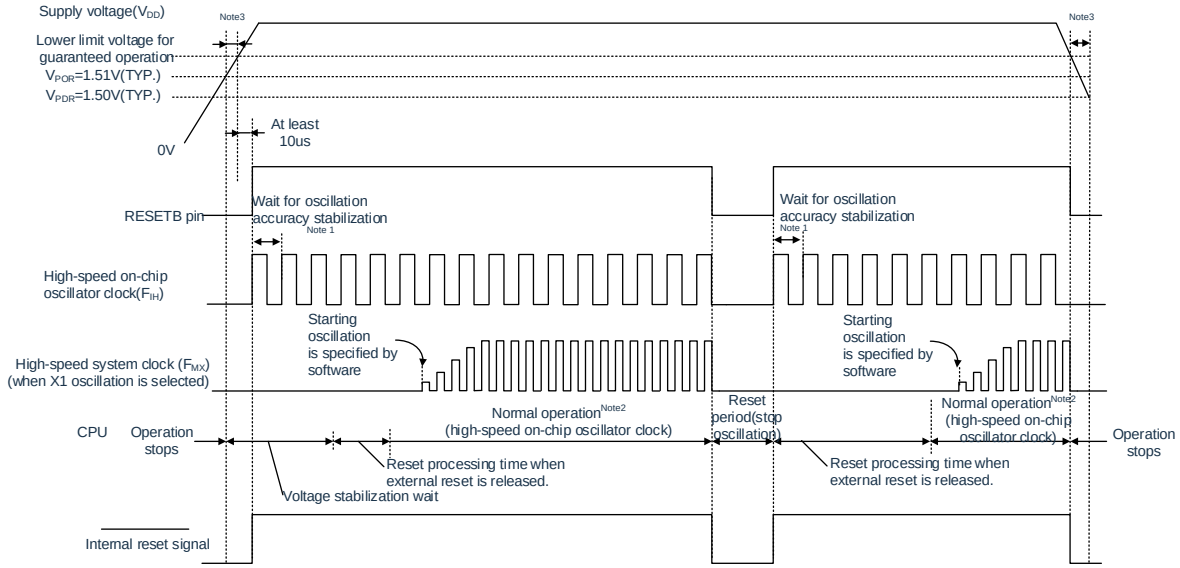


20.3 Operation of power-on reset circuit

The timing of the internal reset signal generation for the power-on reset circuit and the voltage detection circuit is shown below.

Figure 20-2: Timing of internal reset signal generation for power-on reset circuit and voltage detection circuit (1/3)

(1) When the externally input reset signal on the RESETB pin is used



Note 1: The internal reset processing time includes the oscillation accuracy stabilization wait time of the high-speed on-chip oscillator clock.

Note 2: The CPU clock can be switched from the high-speed on-chip oscillator clock to the high-speed system clock or the subsystem clock. In the case of X1 clock, the switch must be made after checking the oscillation stability time by the status register of the oscillation stability time counter (OSTC); in the case of XT1 clock, the switch must be made after checking the oscillation stability time by the timer function, etc.

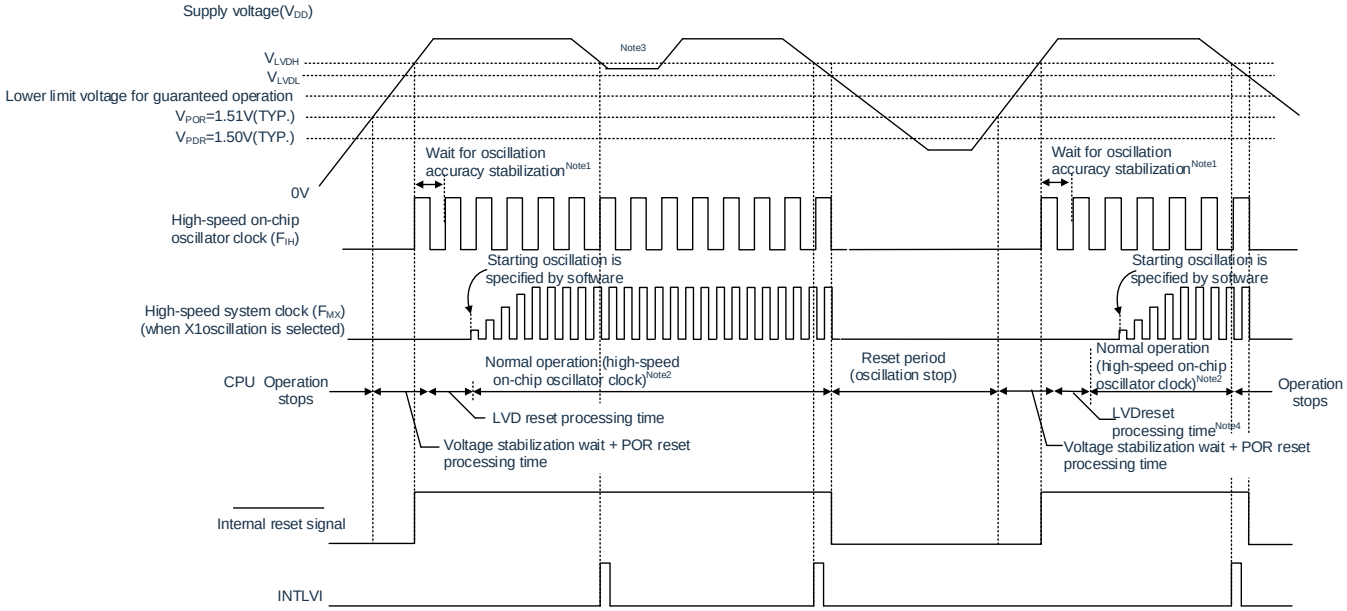
Note 3: When LVD is OFF, the external reset of RESETB pin must be used. For details, please refer to “Chapter 21 Voltage Detection Circuit”.

Notice: When the power supply voltage rises, the power supply voltage must be maintained by external reset before it reaches the working voltage range shown in the AC characteristics of the data sheet; When the supply voltage drops, it must be reset through deep sleep mode transfer, voltage detection circuitry, or external reset before the supply voltage falls below the operating voltage range. When restarting operation, you must confirm that the supply voltage has returned to the operating voltage range.

Remark: V_{POR}: POR power supply rise detection voltage; V_{PDR}: POR power supply fall detection voltage

Figure 20-2: Timing of internal reset signal generation for power-on reset circuit and voltage detection circuit (2/3)

(2) LVD interrupt & reset mode (option byte 000C1: LVIMDS1, LVIMDS0=1, 0)



Note 1: The internal reset processing time includes the oscillation accuracy stabilization wait time of the high-speed on-chip oscillator clock.

Note 2: The CPU clock can be switched from the high-speed on-chip oscillator clock to the high-speed system clock or the subsystem clock. In the case of X1 clock, the switch must be made after checking the oscillation stability time by the status register of the oscillation stability time counter (OSTC); in the case of XT1 clock, the switch must be made after checking the oscillation stability time by the timer function, etc.

Note 3: After generating the interrupt request signal (INTLVI), the LVIV bit and the LVIMD bit of the voltage detection level register (LVIS) are automatically set to "1". Therefore, considering the possibility that the power supply voltage may return to the high voltage detection voltage (VLVDH) or higher without falling below the low voltage detection voltage (VLVDL), follow the steps in "Fig. 21-5 Setting Procedure for Confirmation/Reset of Operating Voltage" and "Fig. 21-6 Setting Procedure for Interrupt and Reset" after generating INTLVI. "Figure 21-6 Initial Setting Procedure for Interrupt & Reset Mode" after generating INTLVI.

Note 4: The time until normal operation begins includes the “Voltage Stabilization Wait + POR Reset Processing Time” after VPOR (1.51V (typical)) is reached as well as the “LVD Reset Processing Time” after the LVD detection level (VLVD) is reached.

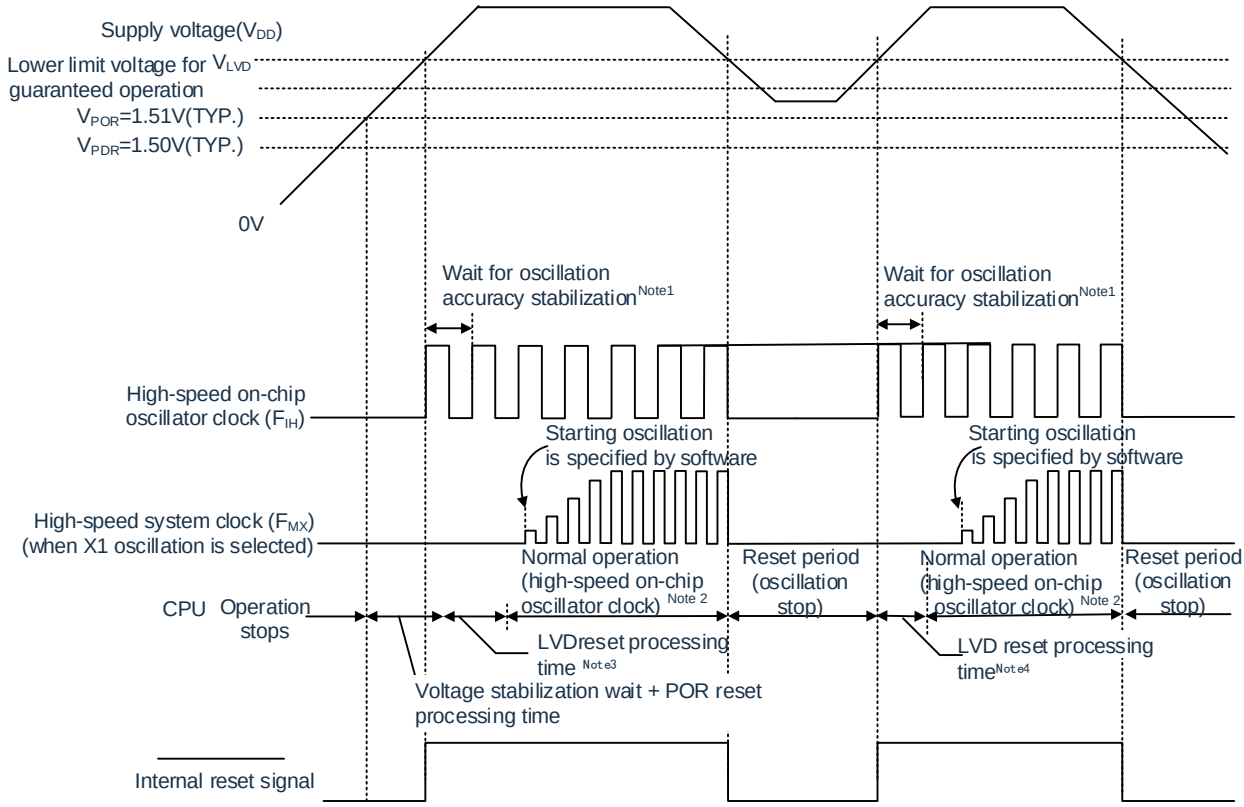
Remark: V_{LVDH}, V_{LVDL}: LVD detection voltage

V_{POR}: POR supply voltage rising detection voltage

V_{PDR}: POR power supply fall detection voltage

Figure 20-2: Timing of internal reset signal generation for power-on reset circuit and voltage detection circuit (3/3)

(3) LVD reset mode (option byte 000C1H: LVIMDS1 = 1, LVIMDS0 = 1)



Note 1: The internal reset processing time includes the oscillation accuracy stabilization wait time of the high-speed on-chip oscillator clock.

Note 2: The CPU clock can be switched from the high-speed on-chip oscillator clock to the high-speed system clock or the subsystem clock. In the case of X1 clock, the switch must be made after checking the oscillation stability time by the status register of the oscillation stability time counter (OSTC); in the case of XT1 clock, the switch must be made after checking the oscillation stability time by the timer function, etc.

Note 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 VPOR (1.51 V, typ.) is reached.

Note 4: When the power supply voltage is below the lower limit for operation and the power supply voltage is then restored after an internal reset is generated only by the voltage detection circuit (LVD), the following LVD reset processing time is required after the LVD detection level (VLVD) is reached.

Remark:

1. V_{LVDH}, V_{LVDL}: LVD detection voltage
   V_{POR}: POR supply voltage rise detection voltage
   V_{PDR}: POR supply voltage drop 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 20-2 (3/3).

Chapter 21 Voltage Detection Circuit

21.1 Function of voltage detection circuit

The voltage detection circuit sets the operating mode and detection voltage (V_{LVDH}, V_{LVDL}, V_{LVD}) by option byte (000C1H). The voltage detection circuit (LVD) has the following functions.

(1) The internal reset or internal interrupt signal is generated by comparing the supply voltage (V_{DD}) with the detection voltage (V_{LVDH}, V_{LVDL}, V_{LVD}).

(2) The detection voltage of the supply voltage (V_{LVDH}, V_{LVDL}) can be selected from 12 detection levels by means of option bytes (see “Chapter 24 Option Bytes”).

(3) It can also operate in deep sleep mode.

(4) When the supply voltage rises, the reset state must be maintained by the voltage detection circuit or external reset before the supply voltage reaches the operating voltage range shown in the AC characteristics of the datasheet; when the supply voltage falls, the reset state must be set by the deep sleep mode transfer, voltage detection circuit or external reset before the supply voltage falls below the operating voltage range. The operating voltage range depends on the setting of the user option byte (000C2H/010C2H).

The voltage detection circuit has the following 3 operation mode settings.

(1) Interrupt & reset mode (LVIMDS1, LVIMDS0=1, 0 of option byte)

Two detection voltages (V_{LVDH}, V_{LVDL}) are selected by the option byte 000C1H. The high voltage detection level (V_{LVDH}) is used to release the reset or generate an interrupt, and the low voltage detection level (V_{LVDL}) is used to generate a reset.

(2) Reset mode (LVIMDS1, LVIMDS0=1, 1 for option byte)

A detection voltage (V_{LVD}) selected by option byte 000C1H is used to generate or release the reset.

(3) Interrupt mode (option byte of LVIMDS1, LVIMDS0=0, 1)

A detection voltage (V_{LVD}) selected by option byte 000C1H is used to generate an interrupt or to release the reset. In each mode, the following interrupt signals and internal reset signals are generated.

Table 21-1: Modes of voltage detection circuits

Interrupt & reset mode (LVIMDS1, LVIMDS0=1, 0)	Reset mode (LVIMDS1, LVIMDS0=1, 1)	Interrupt mode (LVIMDS1, LVIMDS0=0, 1)
When the operating voltage drops, an interrupt request signal is generated when $V_{DD} < V_{LVDH}$ is detected; when $V_{DD} < V_{LVDL}$ is detected, an internal reset is generated. When $V_{DD} \geqslant V_{LVDH}$ is detected, an internal reset is released.	When $V_{DD} \geqslant V_{LVD}$ is detected, an internal reset is released; when $V_{DD} < V_{LVD}$ is detected, an internal reset is generated.	After a reset occurs, an internal reset state of LVD continues until $V_{DD} \geqslant V_{LVD}$. When $V_{DD} \geqslant V_{LVD}$ is detected, an internal reset of LVD is released. After the internal reset of LVD is released, if $V_{DD} < V_{LVD}$ or $V_{DD} \geqslant V_{LVD}$ is detected, then an interrupt request signal (INTLVI) is generated.

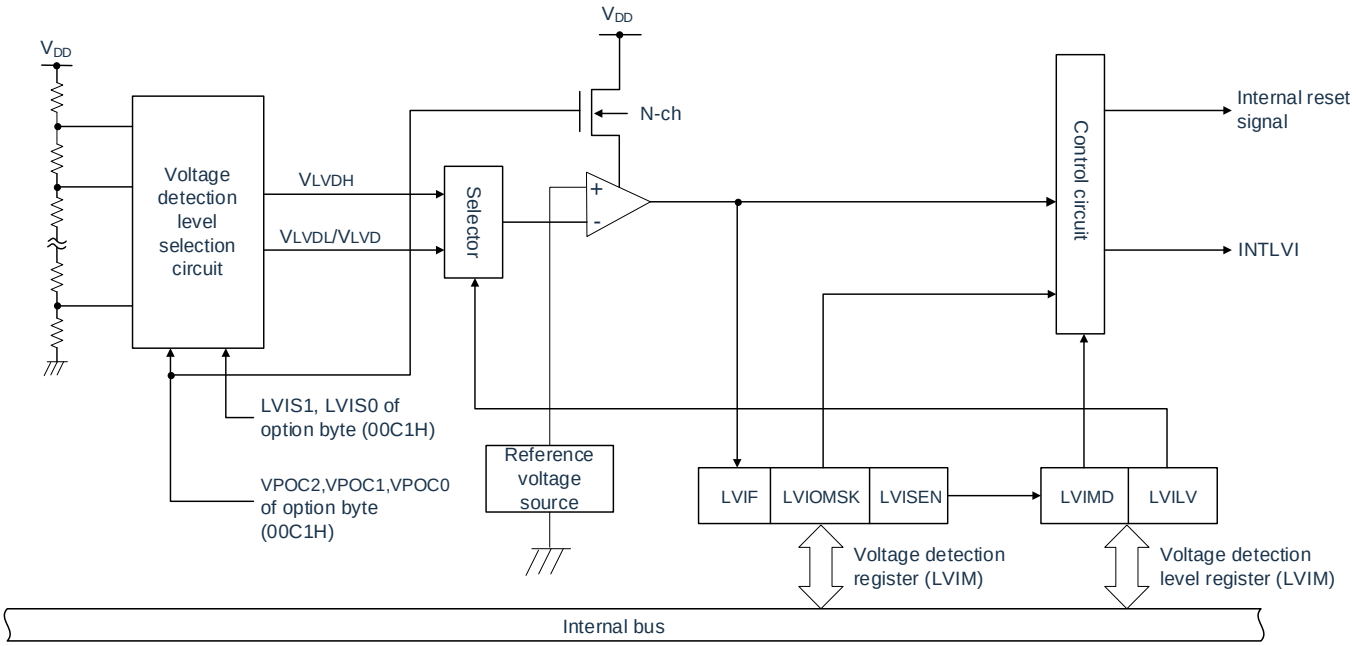
When the voltage detection circuit is in operation, it is possible to check whether the power supply voltage is greater than or less than the detection voltage by reading the voltage detection flag (LVIF: bit 0 of the voltage detection register (LVIM)).

If a reset occurs, bit 0 (LVIRF) of the reset control flag register (RESF) is set to "1". For details of the RESF register, please refer to “Chapter 19 Reset Function”.

21.2 Structure of voltage detection circuit

The block diagram of the voltage detection circuit is shown in Figure 21-2.

Figure 21-2: Block diagram of voltage detection circuit



21.3 Registers for controlling voltage detection circuit

The voltage detection circuit is controlled by the following registers.

(1) Voltage detection register (LVIM)

(2) Voltage detection level register (LVIS)

21.3.1 Voltage detection register (LVIM)

This register is set to enable or disable overwriting of the voltage detection level register (LVIS), and to confirm the masking status of the LVD output. The LVIM register is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes “00H”.

Table 21-3: Format of voltage detection register (LVIM)

Address: 40020441H After reset: 00H[Note1] R/W[Note2]

Symbol	7	6	5	4	3	2	1	0
LVIM	LVISEN[Note3]						LVIOMSK	LVIF

LVISEN[Note3]	Enable/disable setting of voltage detection level register (LVIS)
0	Disable rewriting LVIS register (LVIOMSK=0 (LVD output mask is invalid)).
1	Enable rewriting LVIS register (LVIOMSK=1 (LVD output mask valid)).

LVIOMSK	Mask status flag for LVD output
0	LVD output masking is invalid.
1	LVD output masking is valid [Note 4].

LVIF	Voltage detection flag
0	Supply voltage (V_{DD}) ≥ detection voltage (V_{LVD}) or LVD is OFF.
1	Supply voltage (V_{DD}) < detection voltage (V_{LVD})

Note 1: The reset value varies depending on the reset source. When the LVD is reset, the value of the LVIM register is not reset and the original value is maintained; During other resets, clear LVISEN to “0”.

Note 2: Bit0 and bit1 of the LVIM register are read-only bits.

Note 3: It can only be set when the interrupt & reset mode is selected (lvIMDS1 bits and LVIMDS0 bits of the option bytes are "1" and "0" respectively), the initial value cannot be changed in other modes.

Note 4: Only when the interrupt & reset mode is selected (the LVIMDS1 bit and LVIMDS0 bits of the option byte are "1" and "0" respectively). The LVIOMSK bit automatically changes to "1" during the following periods, masking the reset or interrupt generated by LVD.

① When LVISEN=1.

② Waiting time from the occurrence of LVD interrupt to the stabilization of LVD detection voltage

③ Waiting time from changing the value of the LVILV bit (bit0 of the LVIS register) until the LVD detection voltage stabilizes.

21.3.2 Voltage detection level register (LVIS)

This is a register that sets the voltage sense level. The LVIS register is set by an 8-bit memory manipulation instruction. After generating a reset signal, the value of this register changes to “00H/01H/81H” Note1.

Table 21-4: Format of voltage detection level register (LVIS)

Address: 40020442H After reset: 00H/01H/81H[Note1] R/W

Symbol	7	6	5	4	3	2	1	0
LVIS	LVIMD[Note2]						0	LVILV

LVIMD[Note2]	Operation mode of voltage detection
0	Interrupt mode
1	Reset mode

LVILV[Note2]	LVD detection level
0	High voltage detection level (V_{LVDH})
1	Low voltage detection level (V_{LVDL} or V_{LVD})

Note 1: The reset value varies depending on the setting of the reset source and option bytes. When an LVD reset occurs, this register is not cleared to "00H". When a reset other than LVD occurs, the values of this register are as follows:

① LVIMDS1, LVIMDS0 of Option bytes =1, 0: 00H

② LVIMDS1, LVIMDS0 of Option bytes =1, 1: 81H

③ LVIMDS1, LVIMDS0 of Option bytes =0, 1: 01H

Note 2: Write "0" only if interrupt & reset mode is selected (LVIMDS1 bit and LVIMDS0 bits for option bytes are "1" and "0" respectively). In other cases, it cannot be set. In interrupt & reset mode, value substitution is performed automatically by generating a reset or interrupt.

Notice: To rewrite the LVIS registers, it must be done in accordance with the steps in Figure 21-5 and Figure 21-6.

Remark: Option byte 000C1H selects the mode of operation of the LVD and the detection voltage (V_{LVDH}, V_{LVDL}, V_{LVD}) for each mode. The format of the user option byte (000C1H/010C1H) is shown in Table 21-4. For details of the option byte, refer to "Chapter 24 Option Bytes".

Table 21-4: Format of user option bytes (000C1H/010C1H) (1/2)

Address: 000C1H/010C1H[Note]

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

(1) LVD settings (interrupt & reset mode)

Detection voltage			Setting value of option byte						
V_{LVDH}		V_{LVDL}	VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
rising	falling	falling						LVIMDS1	LVIMDS0
1.77V	1.73V	1.63V	0	0	0	1	0	1	0
1.88V	1.84V					0	1		
2.92V	2.86V					0	0		
1.98V	1.94V	1.84V		0	1	1	0		
2.09V	2.04V					0	1		
3.13V	3.06V					0	0		
2.61V	2.55V	2.45V		1	0	1	0		
2.71V	2.65V					0	1		
3.75V	3.67V					0	0		
2.92V	2.86V	2.75V		1	1	1	0		
3.02V	2.96V					0	1		
4.06V	3.98V					0	0		
—			Settings other than above are prohibited.						

(2) LVD settings (reset mode)

Detection voltage		Setting value of option byte						
V_{LVD}		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
rising	falling						LVIMDS1	LVIMDS0
1.67V	1.63V	0	0	0	1	1	1	1
1.77V	1.73V		0	0	1	0		
1.88V	1.84V		0	1	1	1		
1.98V	1.94V		0	1	1	0		
2.09V	2.04V		0	1	0	1		
2.50V	2.45V		1	0	1	1		
2.61V	2.55V		1	0	1	0		
2.71V	2.65V		1	0	0	1		
2.81V	2.75V		1	1	1	1		
2.92V	2.86V		1	1	1	0		
3.02V	2.96V		1	1	0	1		
3.13V	3.06V		0	1	0	0		
3.75V	3.67V		1	0	0	0		
4.06V	3.98V		1	1	0	0		
—		Settings other than above are prohibited.						

Note: The detection voltage is a TYP Value. For details, please refer to the LVD circuit characteristics in the data sheet.

Table 21-4: Format of user option bytes (000C1H/010C1H) (2/2)

Address: 000C1H

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

(1) LVD settings (interrupt mode)

Detection voltage		Setting value of option byte						
V_{LVD}		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
rising	falling						LVIMDS1	LVIMDS0
1.67V	1.63V	0	0	0	1	1	0	1
1.77V	1.73V		0	0	1	0		
1.88V	1.84V		0	1	1	1		
1.98V	1.94V		0	1	1	0		
2.09V	2.04V		0	1	0	1		
2.50V	2.45V		1	0	1	1		
2.61V	2.55V		1	0	1	0		
2.71V	2.65V		1	0	0	1		
2.81V	2.75V		1	1	1	1		
2.92V	2.86V		1	1	1	0		
3.02V	2.96V		1	1	0	1		
3.13V	3.06V		0	1	0	0		
3.75V	3.67V		1	0	0	0		
4.06V	3.98V		1	1	0	0		
—		Settings other than above are prohibited.						

(2) LVD is OFF (external reset using the RESETB pin)

Detection voltage		Setting value of option byte						
V_{LVD}	V_{LVD}	VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
rising	rising						LVIMDS1	LVIMDS1
—	—	1	×	×	×	×	×	×
—		Settings other than above are prohibited.						

Notice:

1. Write "1" to bit4 in interrupt mode.
2. When the power supply voltage rises, the reset state must be maintained through the voltage detection circuit or external reset before the power supply voltage reaches the working voltage range shown in the AC characteristics of the data sheet; When the supply voltage drops, it must be reset by transferring in deep sleep mode, voltage detection circuitry, or external reset before the supply voltage falls below the operating voltage range.

Remark:

3. ×: Ignore.
4. The detection voltage is a TYP value. For details, please refer to the LVD circuit characteristics in the data sheet.

21.4 Operation of voltage detection circuit

21.4.1 When used as reset mode

The operation mode (reset mode (LVIMDS1, LVIMDS0=1, 1)) and the detection voltage (V_{LVD}) are set via the option byte 000C1H. If the reset mode is set, operation starts with the following initial settings.

(1) Set bit 7 (LVISEN) of the voltage detection register (LVIM) to "0" (disable rewriting the voltage detection level register (LVIS))

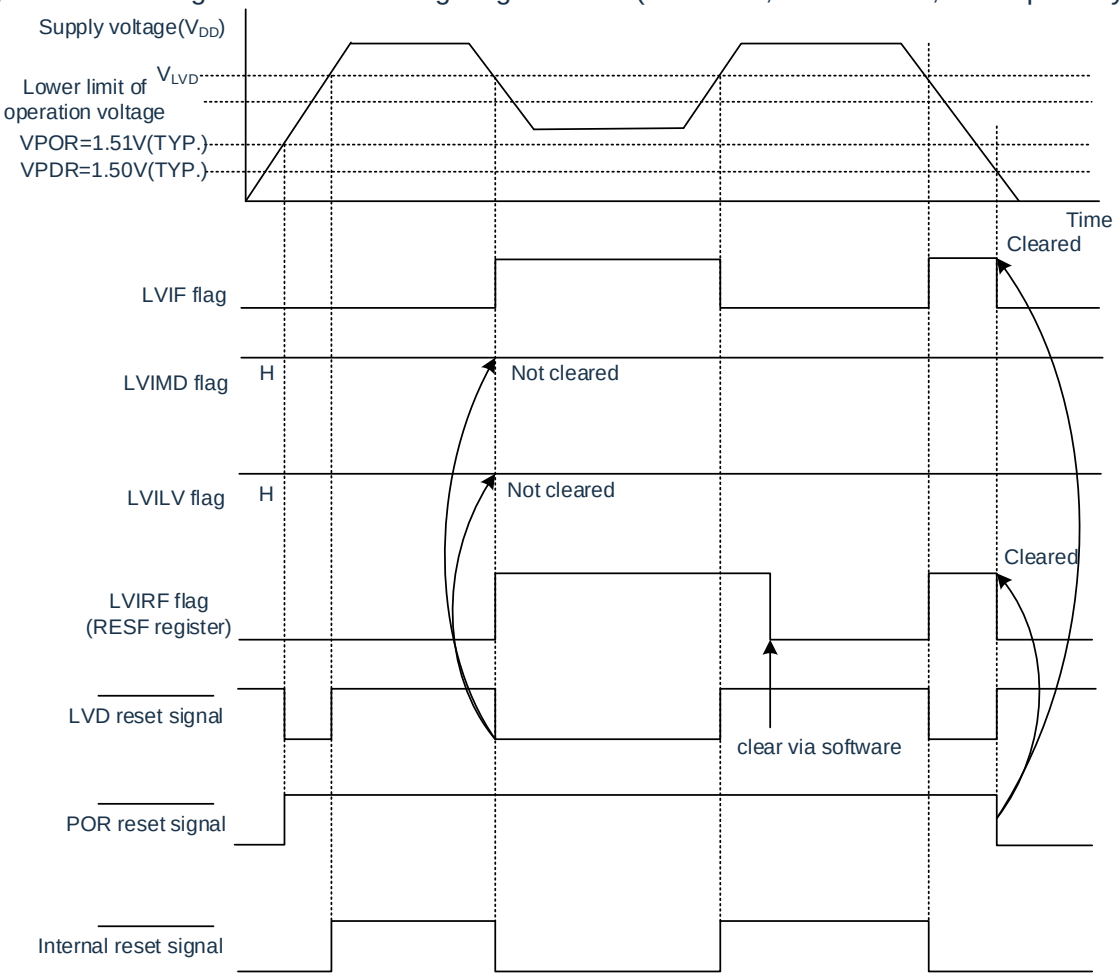
(2) Set the initial value of the voltage detection level register (LVIS) to "81H". Set bit7(LVIMD) to "1" (reset mode). Set bit0 (LVILV) to "1"(voltage detection level: V_{LVD}).

Operation of LVD reset mode:

When the power is turned on, the reset mode (LVIMDS1, LVIMDS0=1, 1 of the option byte) keeps the internal reset state of LVD until the supply voltage (V_{DD}) exceeds the voltage detection level (V_{LVD}). If the supply voltage (V_{DD}) exceeds the voltage detection level (V_{LVD}), the internal reset is released. When the operating voltage falls, an internal reset of LVD is generated if the supply voltage (V_{DD}) is below the voltage detection level (V_{LVD})

The timing of the internal reset signal generation for LVD reset mode is shown in Figure 21-2.

Figure 21-2: Timing of internal reset signal generation (LVIMDS1, LVIMDS0=1, 1 for option byte)



Remark: V_{POR}: POR power supply rise detection voltage; V_{PDR}: POR power supply fall detection voltage

21.4.2 When used as interrupt mode

The operation mode (interrupt mode (LVIMDS1, LVIMDS0=0, 1)) and the detection voltage (V_{LVD}) are set via the option byte 000C1H. If the interrupt mode is set, operation starts with the following initial settings.

(1) Set bit 7 (LVISEN) of the voltage detection register (LVIM) to "0" (disables rewriting the voltage detection level register (LVIS)).

(2) Set the initial value of the voltage detection level register (LVIS) to "01H". Set bit7 (LVIMD) to "0" (interrupt mode). Set bit0(LVILV) to "1" (voltage detection level: V_{LVD}).

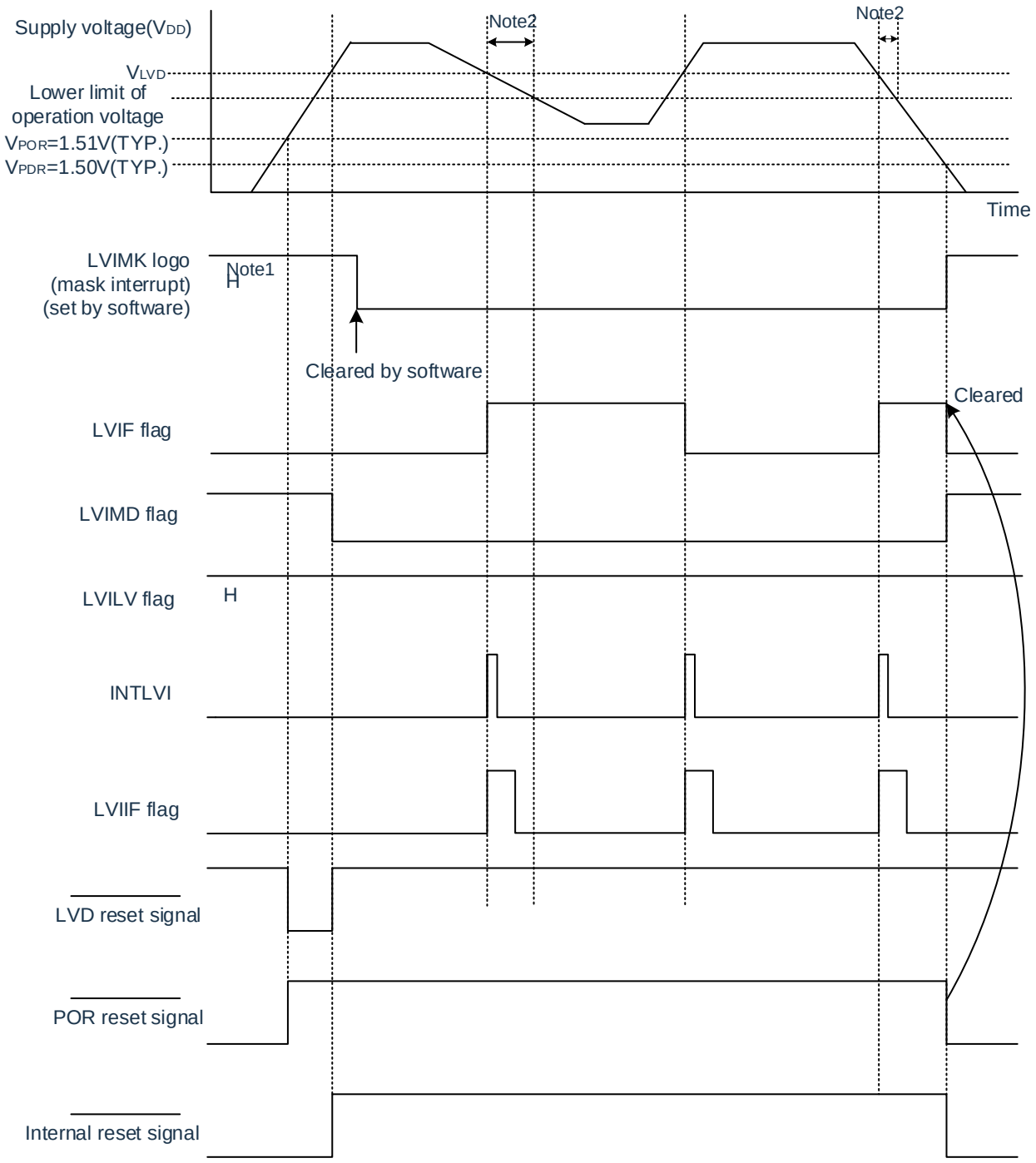
Operation of LVD interrupt mode:

After generating a reset, the interrupt mode (LVIMDS1, LVIMDS0 of the option byte =0, 1) maintains the internal reset state of the LVD until the supply voltage (V_{DD}) exceeds the voltage detection level (V_{LVD}). If the supply voltage (V_{DD}) exceeds the voltage detection level (V_{LVD}), the internal reset of the LVD is released.

If the supply voltage (V_{DD}) exceeds the voltage detection level (V_{LVD}) after the internal reset of the LVD is released, an interrupt request signal (INTLVI) of the LVD is generated. When the operating voltage drops, it must be set to the reset state by deep sleep mode transfer or external reset before the operating voltage falls below the operating voltage range shown in the AC characteristics of the datasheet. When restarting operation, it must be verified that the supply voltage has returned to the operating voltage range.

The timing of the interrupt request signal generation for LVD interrupt mode is shown in Figure 21-3.

Figure 21-3: Timing of interrupt signal generation (LVIMDS1, LVIMDS0 of option byte =0, 1)



Note 1: After generating a reset signal, the LVIMK flag changes to "1".

Note 2: When the operating voltage drops, it must be reset by deep sleep mode transfer or external reset before the operating voltage falls below the operating voltage range shown in the AC characteristics of the data sheet. When restarting operation, it must be verified that the supply voltage returns to the operating voltage range.

Remark: V_{POR}: POR power supply rise detection voltage; V_{PDR}: POR power supply fall detection voltage

21.4.3 When used as interrupt & reset mode

The operation mode (interrupt & reset mode (LVIMDS1, LVIMDS0=1, 0)) and the detection voltage (V_{LVDH}, V_{LVDL}) are set via the option byte 000C1H. If the interrupt & reset mode is set, the operation starts with the following initial settings.

(1) Set bit 7 (LVISEN) of the voltage detection register (LVIM) to "0" (disables rewriting the voltage detection level register (LVIS)).

(2) Set the initial value of the voltage detection level register (LVIS) to "00H". Set bit7 (LVIMD) to "0" (interrupt mode). Set bit0(LVILV) to "0" (high voltage detection level: V_{LVDH}).

Operation of LVD interrupt & reset mode:

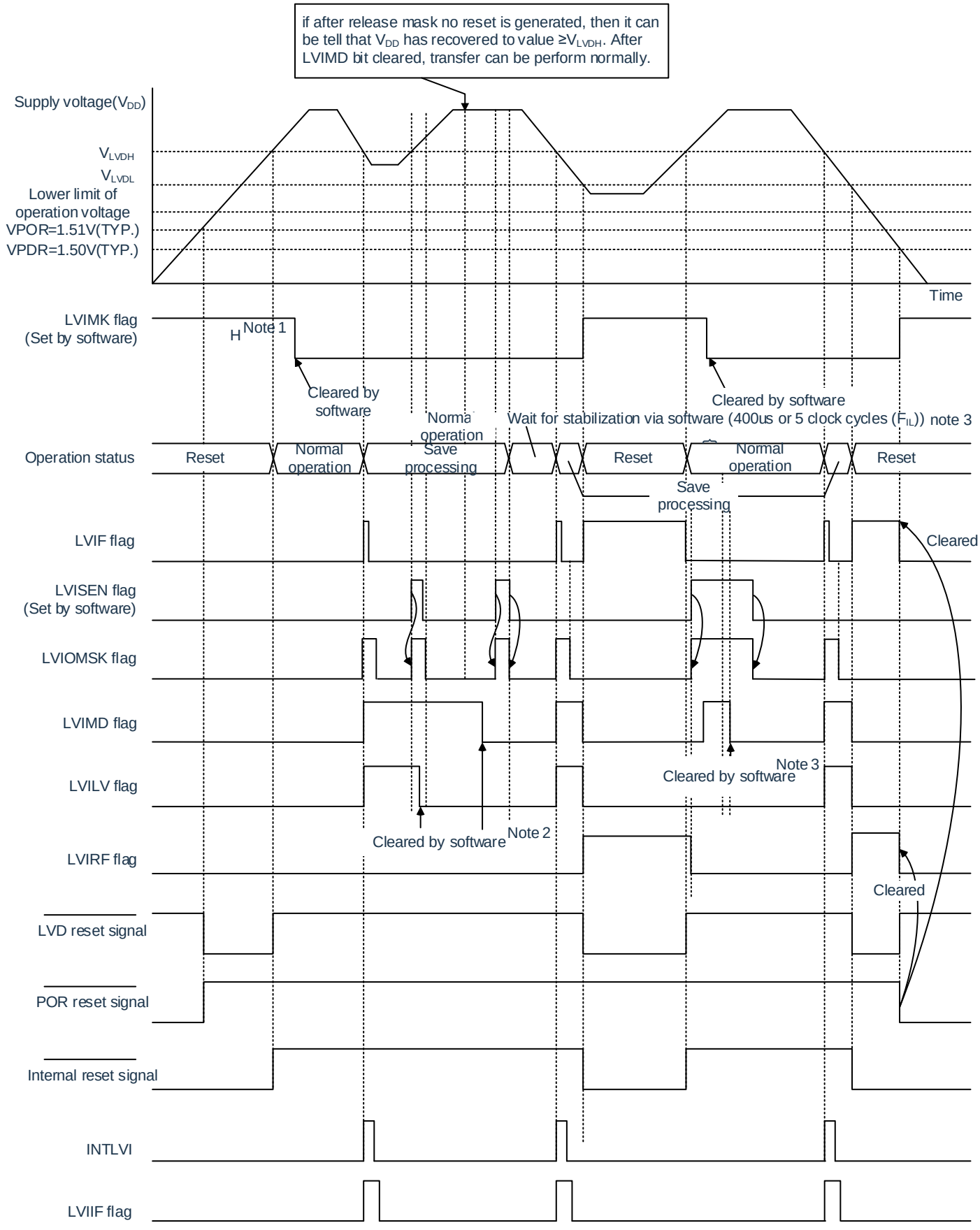
When power is turned on, the interrupt & reset mode (LVIMDS1, LVIMDS0=1, 0 of the option byte) maintains the internal reset state of the LVD until the power supply voltage (V_{DD}) exceeds the high voltage detection level (V_{LVDH}). If the supply voltage (V_{DD}) exceeds the high voltage detection level (V_{LVDH}), the internal reset is released.

When the operating voltage drops, if the supply voltage (V_{DD}) is below the high voltage detection level (V_{LVDH}), an interrupt request signal (INTLVI) is generated for the LVD and any stacking process can be performed. After that, if the supply voltage (V_{DD}) is below the low voltage detection level (V_{LVDL}), an internal reset of the LVD is generated. However, after INTLVI occurs, no interrupt request signal is generated even if the supply voltage (V_{DD}) returns to the high voltage detection voltage (V_{LVDH}) or higher without falling below the low voltage detection voltage (V_{LVDL}).

When using LVD interrupt & reset mode, you must follow “Figure 21-5: Setting procedure for confirmation /reset of operating voltage” and “Figure 21-6: Initial setting procedure for interrupt & reset mode”.

The timing of the internal reset signal and interrupt signal generation in LVD interrupt & reset mode is shown in Figure 21-4.

Figure 21-4: Reset & interrupt signal generation timing (LVIMDS1, LVIMDS0=1, 0) (1/2)



Note 1: After the reset signal is generated, the LVIMK flag becomes "1".

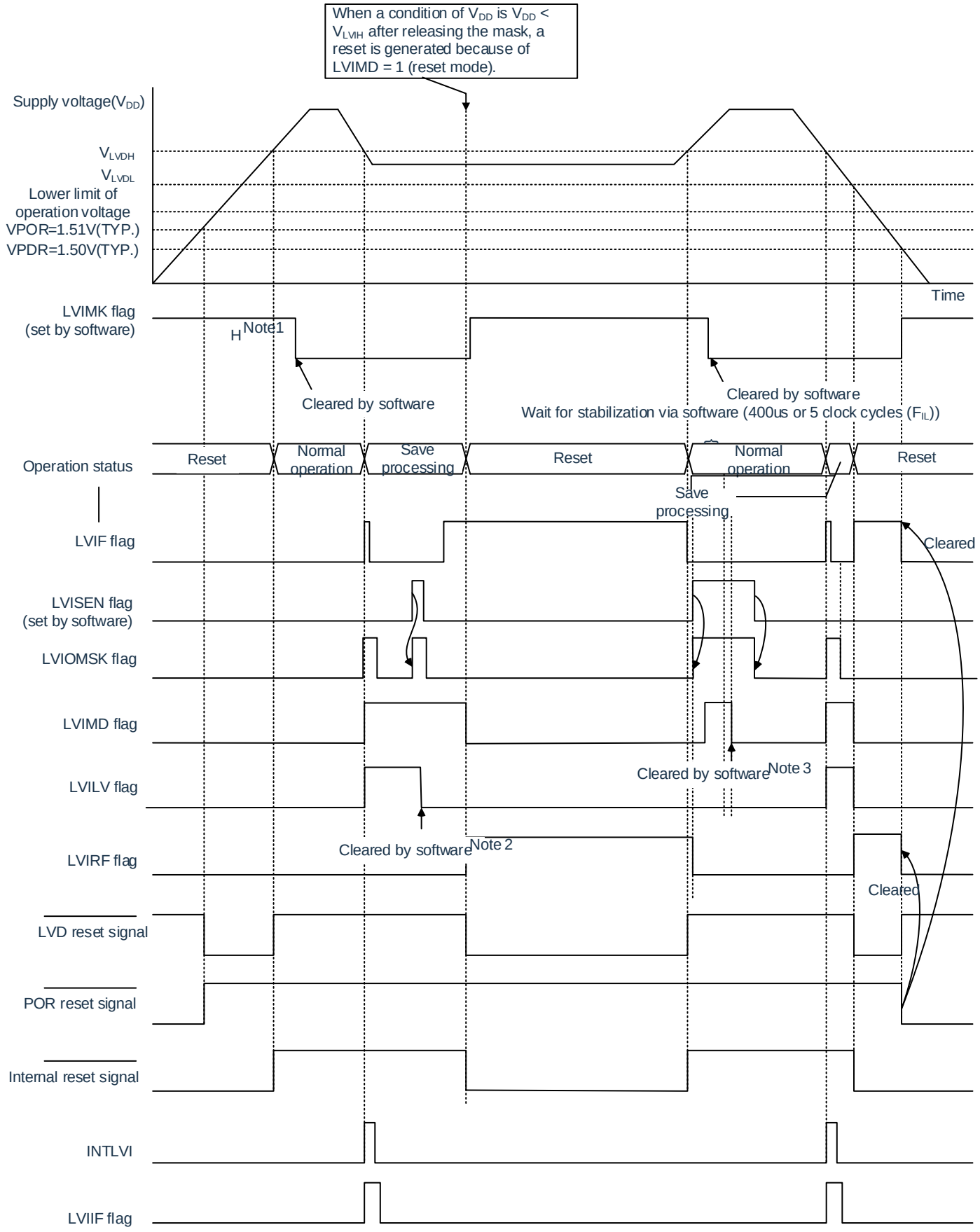
Note 2: When using the interrupt & reset mode, you must follow “Figure 21-5: Setting procedure for

confirmation /reset of operating voltage” after an interrupt occurs.

Note 3: When using the interrupt&reset mode, you must follow the steps in “Figure 21-6: Initial setting procedure for interrupt & reset mode” after the reset is released.

Remark: V_{POR}: POR power supply rise detection voltage; V_{PDR}: POR power supply fall detection voltage

Figure 21-4: Reset & interrupt signal generation timing (LVIMDS1, LVIMDS0=1, 0) (2/2)



Note 1: The LVIMK flag is set to “1” by reset signal generation.

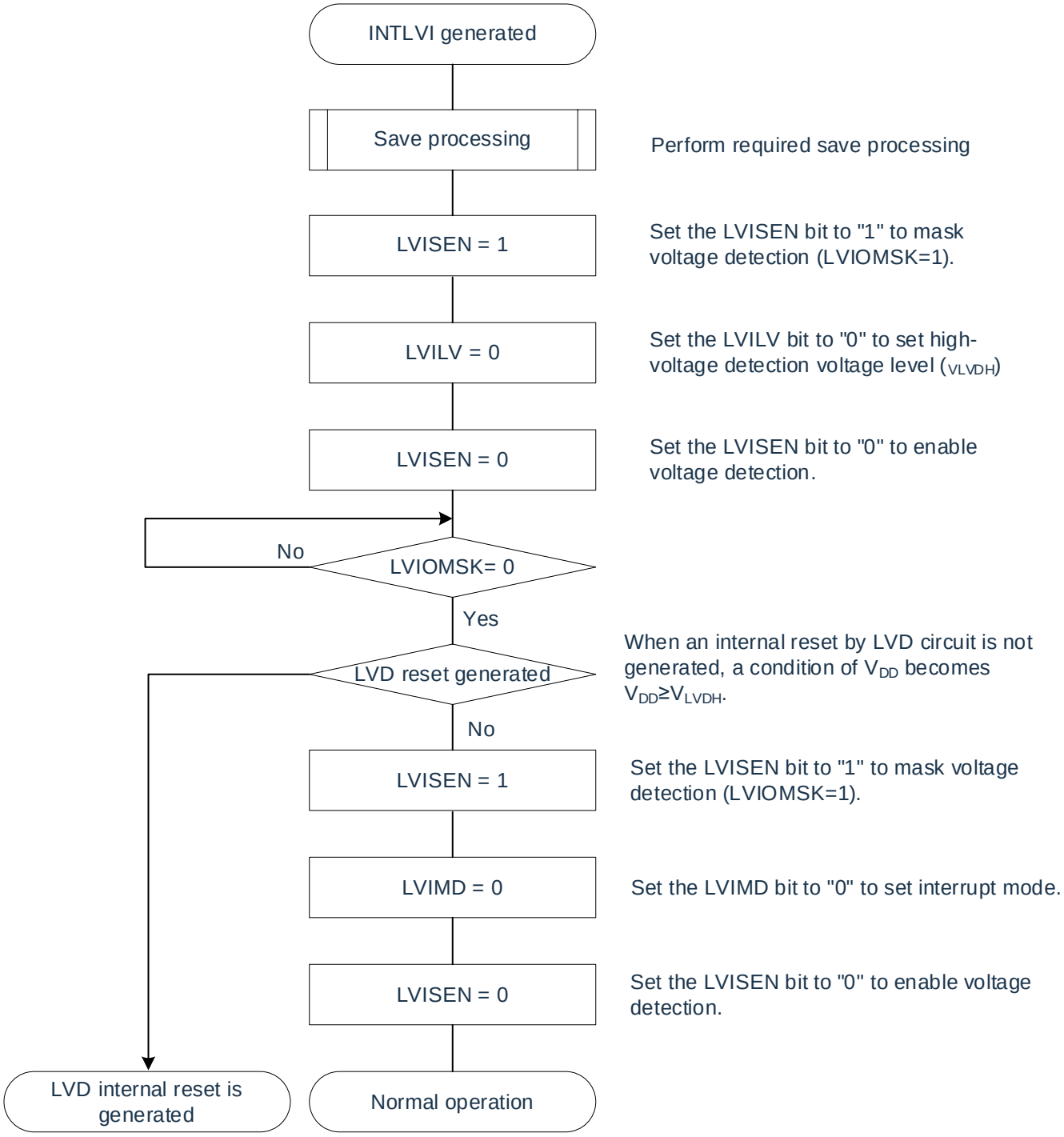
Note 2: When using the interrupt & reset mode, you must follow “Figure 21-5: Setting procedure for

confirmation /reset of operating voltage” after an interrupt occurs.

Note 3: When using the interrupt&reset mode, you must follow the steps in “Figure 21-6: Initial setting procedure for interrupt & reset mode” after the reset is released.

Remark: V_{POR}: POR power supply rise detection voltage; V_{PDR}: POR power supply fall detection voltage

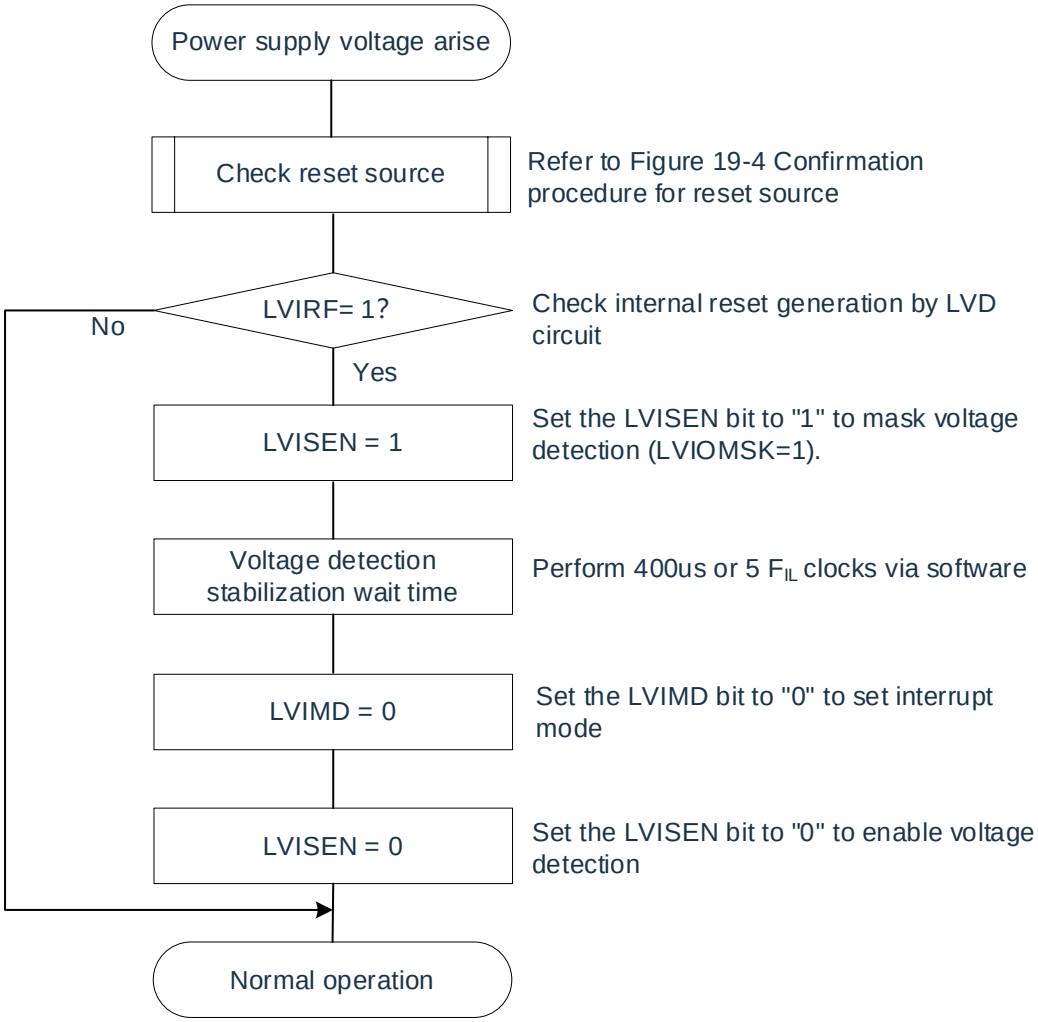
Figure 21-5: Setting procedure for confirmation/reset of operating voltage



If the interrupt & reset mode is set (LVIMDS1, LVIMDS0=1, 0), it will take 400us or 5 F_{IL} clocks for the voltage detection to stabilize after the LVD reset (LVIRF=1) is released. The LVIMD bit must be cleared to "0" for initialization after waiting for the voltage detection to stabilize. The LVISEN bit must be set to "1" during the count of the voltage detection stabilization time and when rewriting the LVIMD bit to block the generation of resets or interrupts generated by LVD.

The initial setting procedure for interrupt & reset mode is shown in Figure 21-6.

Figure 21-6: Initial setting procedure for interrupt & reset mode



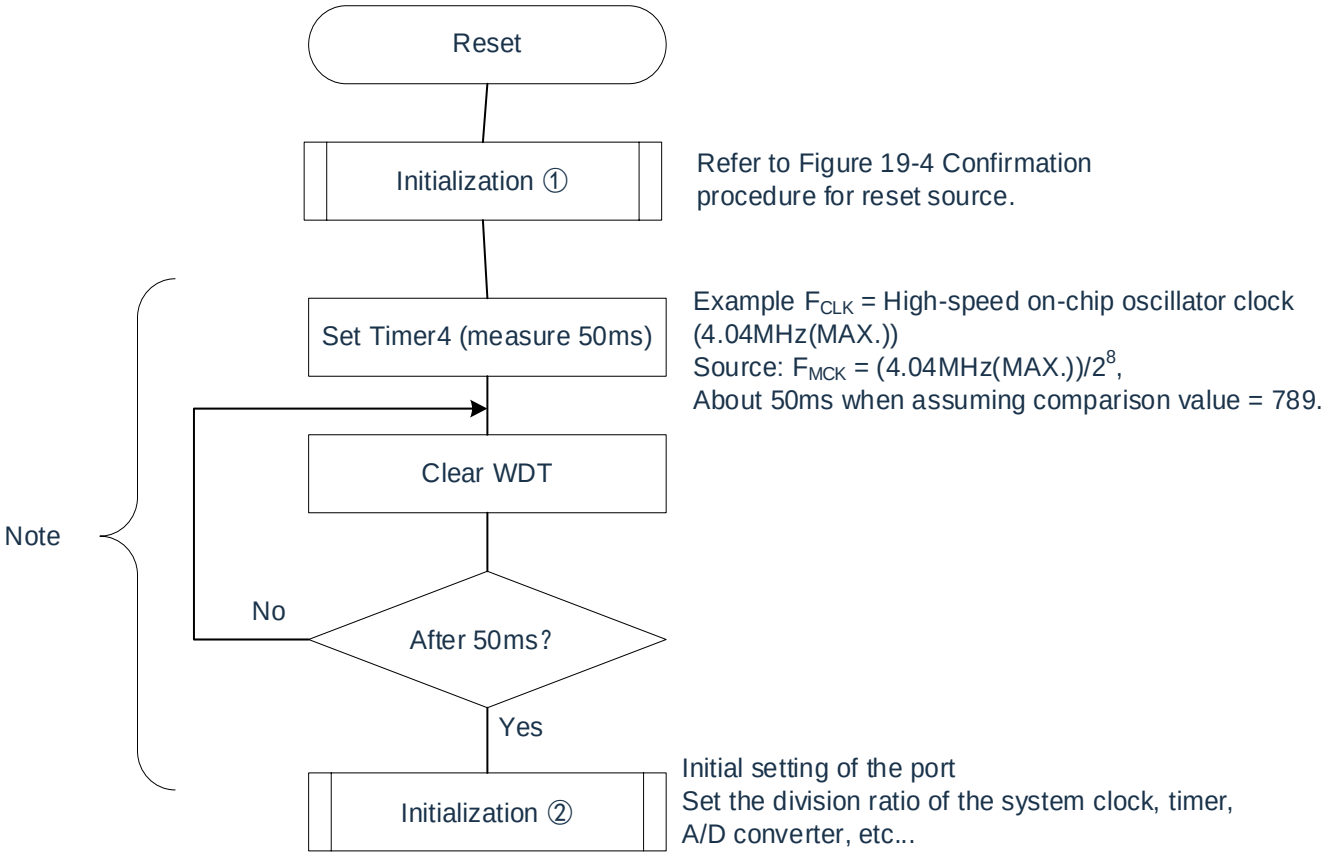
Remark: F_{IL}: Low-speed on-chip oscillator clock frequency

21.5 Cautions for voltage detection circuits

(1) Voltage fluctuation when power is supplied

In a system where the supply voltage (V_{DD}) fluctuates for a certain period in the vicinity of the 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. After releasing the reset signal, wait for the supply voltage fluctuation period of each system by means of a software counter that uses a timer, and then initialize the ports.

Figure 21-7: Example of software processing when the supply voltage fluctuation near the LVD detection voltage does not exceed 50ms

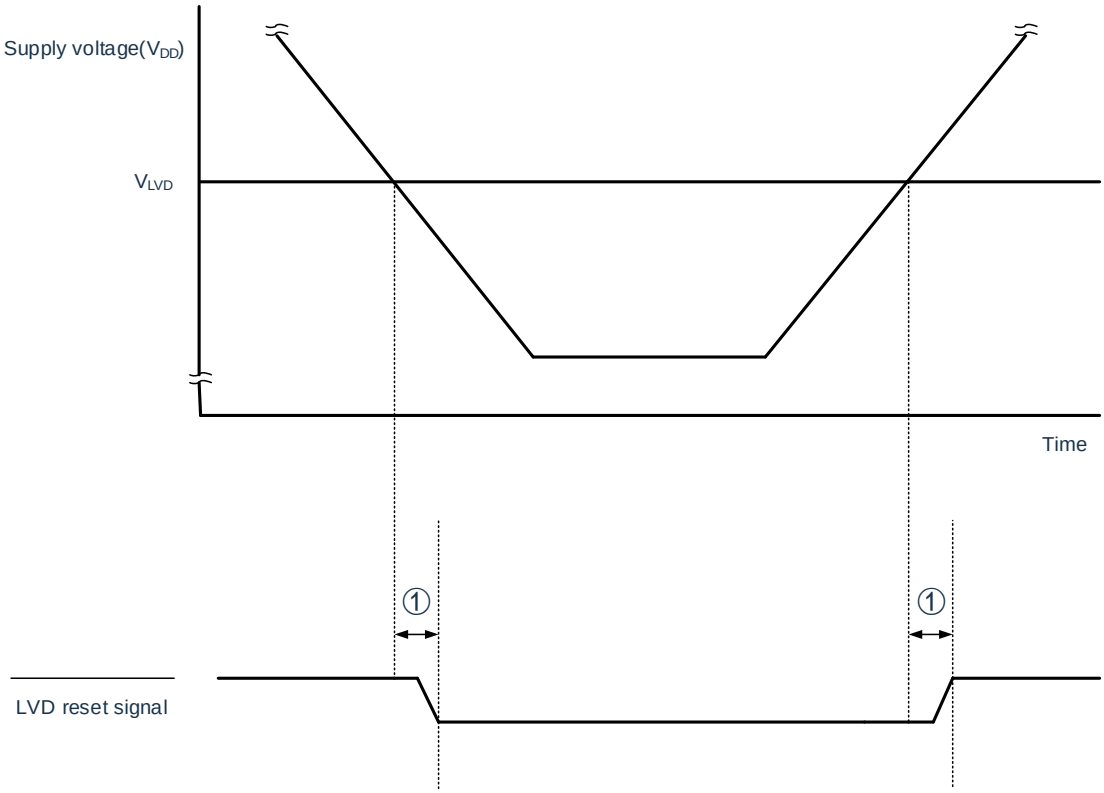


Note: If the reset occurs again during this period, it is not switched to initialization processing ②.

(2) Delay from generation of LVD reset source to generation or release of LVD reset

A delay occurs from the time the supply voltage (V_{DD}) < LVD detection voltage (V_{LVD}) is met to the time the LVD reset is generated. Similarly, a delay occurs from the time the LVD detection voltage (V_{LVD}) ≤ the supply voltage (V_{DD}) to the time the LVD reset is released (see Figure 21-8).

Figure 21-8: Delay from generation of LVD reset source to generation or release of LVD reset



Remark: ①: Detection delay (300us(MAX.))

(1) When the power is turned on with LVD set to OFF

When LVD is set to OFF, an external reset must be performed using the RESETB pin.

When performing an external reset, the RESETB pin must be input low for at least 10us. If an external reset is performed while the supply voltage is rising, the power must be turned on after a low level is input to the RESETB pin, and must be held low for at least 10us within the operating voltage range shown in the AC characteristics of the datasheet, followed by a high level.

(2) When LVD is set to OFF in LVD interrupt mode and the operating voltage drops

If the operating voltage drops when LVD is set to OFF and LVD interrupt mode is set, it must be reset by deep sleep mode transfer or external reset before the operating voltage falls below the operating voltage range shown in the AC characteristics of the data sheet. When restarting operation, it is necessary to verify that the supply voltage is restored in the operating voltage range.

Chapter 22 Safety Function

22.1 Summary of safety functions

In order to comply with IEC60730 and EC61508 safety standards, the CMS32L032 has the following built-in safety features.

The purpose of this function is to safely stop the operation when a fault is detected through the self-diagnosis of the microcontroller.

(1) Flash CRC function (high-speed CRC, general-purpose CRC)

The data error of flash memory is detected by CRC operation. The following two CRCs can be used depending on the application and usage conditions.

① "High-speed CRC"...During the initialization program, it is possible to stop the CPU and check the entire code flash area at high speed.

② "General-purpose CRC"...can be used for multi-purpose checks during CPU operation, not limited to the code flash area.

(2) SFR guard function

This prevents SFRs from being rewritten when the CPU freezes.

(3) Frequency detection function

Self-testing of CPU/peripheral hardware clock frequency using a general-purpose timer unit.

(4) A/D test function

A/D converter self-test by A/D conversion of positive (+) reference voltage, negative (-) reference voltage, analog input channel (ANI), temperature sensor output and internal reference voltage output of A/D converter.

(5) Digital output signal level detection function for input/output ports

When the input/output port is in output mode, the output level of the pin can be read.

22.2 Registers used for safety functions

The following registers are used for each function of the security function.

Register Name	Function
• Flash CRC control register (CRC0CTL) • Flash CRC operation result register (PGCRCL)	Flash CRC operation function (High-speed CRC)
• CRC input register (CRCIN) • CRC data register (CRCD)	CRC calculation function (General CRC)
• Special SFR protection control register (SFRGD)	SFR guard function
• Timer input select register 0 (TIS0)	Frequency detection function
• Analog input channel specification register (ADS)	A/D test function
• Port mode selection register (PMS)	Digital output signal level detection function for input/output pins

The contents of each register are described in "22.3 Operation of safety function".

22.3 Operation of safety functions

22.3.1 Flash CRC operation function (high-speed CRC)

The IEC60730 standard requires verification of the data in the flash memory and recommends CRC as a means of verification. This high-speed CRC can check the entire code flash area in the initial setup (initialization) procedure.

High-speed CRC stops the CPU and reads 32 bits of data from the flash memory with one clock for operation. Therefore, it is characterized by a short time to complete the verification (e.g., 64KB flash: 512us@32MHz).

The CRC generation polynomial used complies with "$X^{16}+X^{12}+X^5+1$" of CRC-16-CCITT.

MSB of bit31→bit0 is operated first.

Remark: The operation result is differentbecause the general CRC operates in LSB first order.

22.3.1.1 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 an 8-bit memory manipulation instruction. Reset signal generation clears this register to 00H.

Table 22-1: Format of flash memory CRC control register (CRC0CTL)

Address: 40021810H | After reset: 00H | R/W

Symbol	7	6	5	4	3	2	1	0
CRC0CTL	CRC0EN	CRCCHK60	0	0	0	FEA2	FEA1	FEA0

CRC0EN	Control of CRC ALU operation
0	Operation stopped
1	Operation is started by executing the WFE instruction.

CRCCHK60	FEA2	FEA1	FEA0	High-speed CRC operation range
0	0	0	0	00000H ~ 1FFBH(8K-4byte)
0	0	0	1	00000H ~ 3FFBH(16K-4byte)
0	0	1	0	00000H ~ 5FFBH(24K-4byte)
0	0	1	1	00000H ~ 7FFBH(32K-4byte)
0	1	0	0	00000H ~ 9FFBH(40K-4byte)
0	1	0	1	00000H ~ BFFBH(48K-4byte)
0	1	1	0	00000H ~ DFFBH(56K-4byte)
0	1	1	1	00000H ~ FFFBH(64K-4byte)
1	0	0	0	00000H ~ EFFFBH(60K-4byte)

Notice:

1. Bit3~5 must be set to 0.
2. 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.

22.3.1.2 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.

Table 22-2: Format of memory CRC operation result register (PGCRCL)

Address:	0x40021812			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

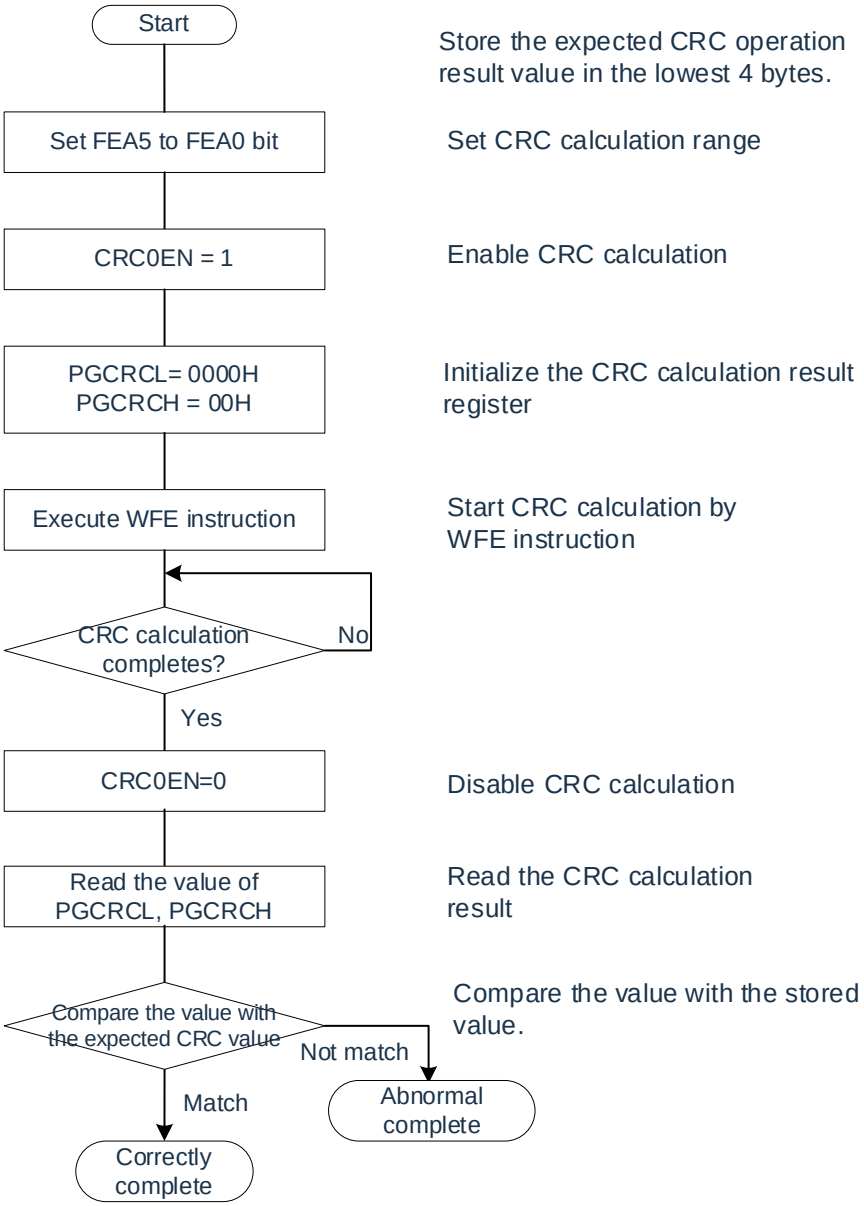
PGCRC15~0	High-speed CRC operation results
0000H~FFFFH	Store the high-speed CRC operation results.

Notice: The PGCRCL register can only be written if CRC0EN (bit 7 of the CRC0CTL register) = 1.

The flowchart of the flash memory CRC operation function (high-speed CRC) is shown in Figure 22-1.

＜Operation flow＞

Figure 22-1: Flow chart of flash CRC operation function (high-speed CRC)



Notice:

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.

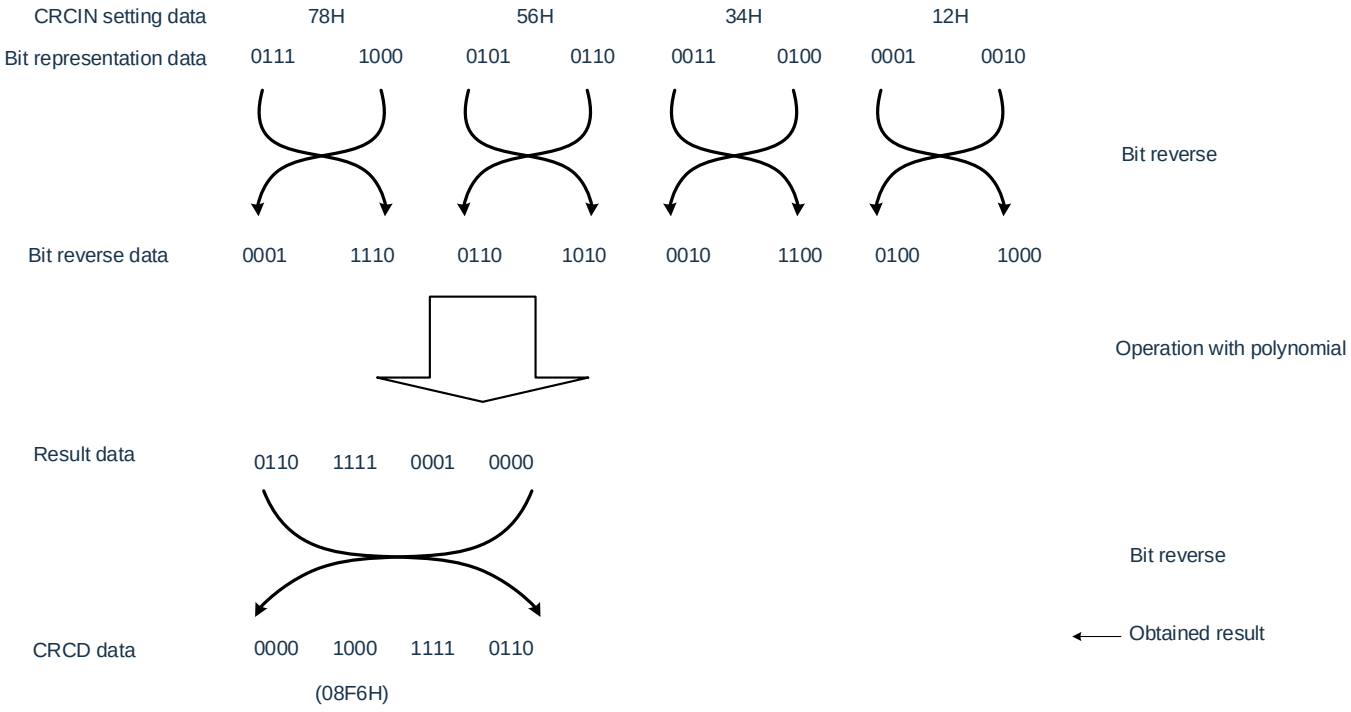
22.3.2 CRC operation function (general-purpose CRC)

In order to guarantee safety during operation, the IEC61508 standard mandates the checking of data even while the CPU is operating.

The 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).

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.



Remark: 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.

22.3.2.1 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.

Table 22-3: Format of CRC input register (CRCIN)

Address: 400433ACH After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CRCIN								

bit7~0	Function
00H~FFH	Data input

22.3.2.2 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.

Table 22-4: Format of CRC data register (CRCD)

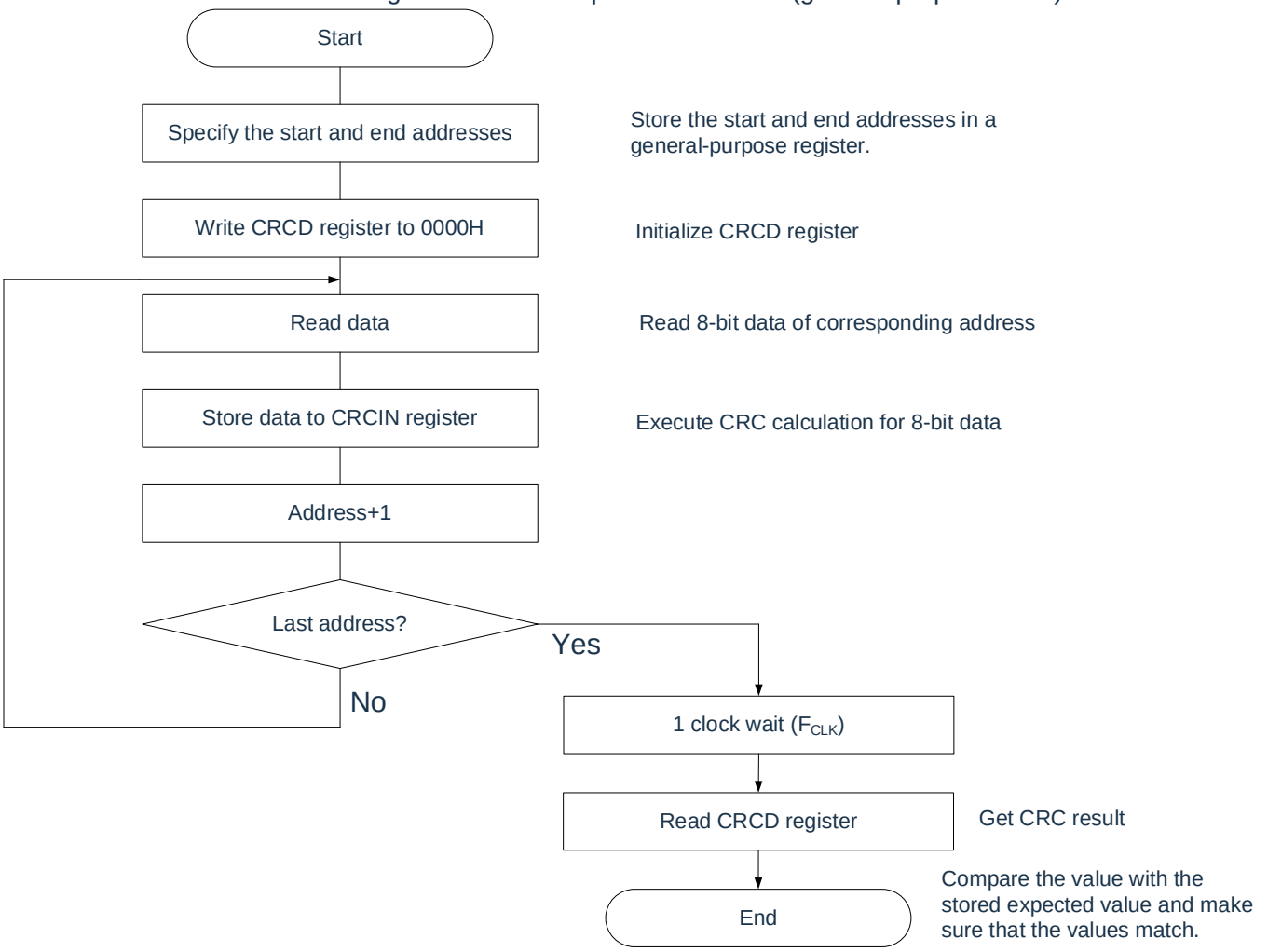
Address: 400432FAH After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRCD																

The flowchart of the CRC operation function is shown in Figure 22-2.

＜Operation flow＞

Figure 22-2: CRC operation function (general-purpose CRC)



Notice:

1. To read the write value of the CRCD register, the CRCD register must be read before the CRCIN register is written.
2. If a write operation to the CRCD register competes with the saving of an operation result, the write operation is ignored.

22.3.3 SFR guard function

In order to ensure safety during operation, the IEC61508 standard requires that even if the CPU is out of control, it is necessary to protect important SFR from being rewritten. The SFR protection function is used to protect data from the control registers of the comparator function, port function, interrupt function, clock control function, and voltage detection circuitry.

If the SFR protection function is set, the write operation of the protected SFR is invalid, but it can be read normally.

22.3.3.1 SFR guard control register (SFRGD)

This register controls whether the SFR guard function is valid.

The SFR guard function uses GCOMP bits, GPORT bits, GINT bits, and GCSC bits.

The SFRGD register is set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register becomes "00H".

Table 22-5: Format of SFR guard control register (SFRGD)

Address: 40040478H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
SFRGD	0	0	0	0	0	GPORT	GINT	GCSC

GPORT	Protection of control registers for port functions
0	Invalid. Can read and write control registers for port functions.
1	Valid. Write operation of the control register of the port function is invalid, and it can be read. [Protected SFR] PMxx, PUxx, PDxx, POMxx, PMCxx, PxxCFG[Note]

GINT	Protection of registers for interrupt functions
0	Invalid. Can read and write control registers for interrupt functions.
1	Valid. Write operation of the control register of the interrupt function is invalid, and it can be read. [Protected SFR] IFxx, MKxx, PRxx, EGPx, EGNx

GCSC	Clock control function, voltage detection circuit control register protection
0	Invalid. Can read/write clock control function, voltage detection circuit control register.
1	Valid. Clock control function, write operation of the control register of the voltage detection circuit is invalid and read operation is enabled. [Protected SFR] CMC, CSC, OSTS, CKC, PERx, OSMC, LVIM, LVIS

Remark: Pxx (port register) is not protected.

22.3.4 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 1 of the Timer40, whether the proportional relationship between the two clock frequencies is correct can be determined.

Note that, however, if one or both clock operations are completely stopped, the proportional relationship between the clocks cannot be determined.

＜Clocks to be compared＞

(1) CPU/peripheral hardware clock frequency (F_{CLK}):

① High-speed on-chip oscillator clock (F_{IH})

② High-speed system clock (F_{MX})

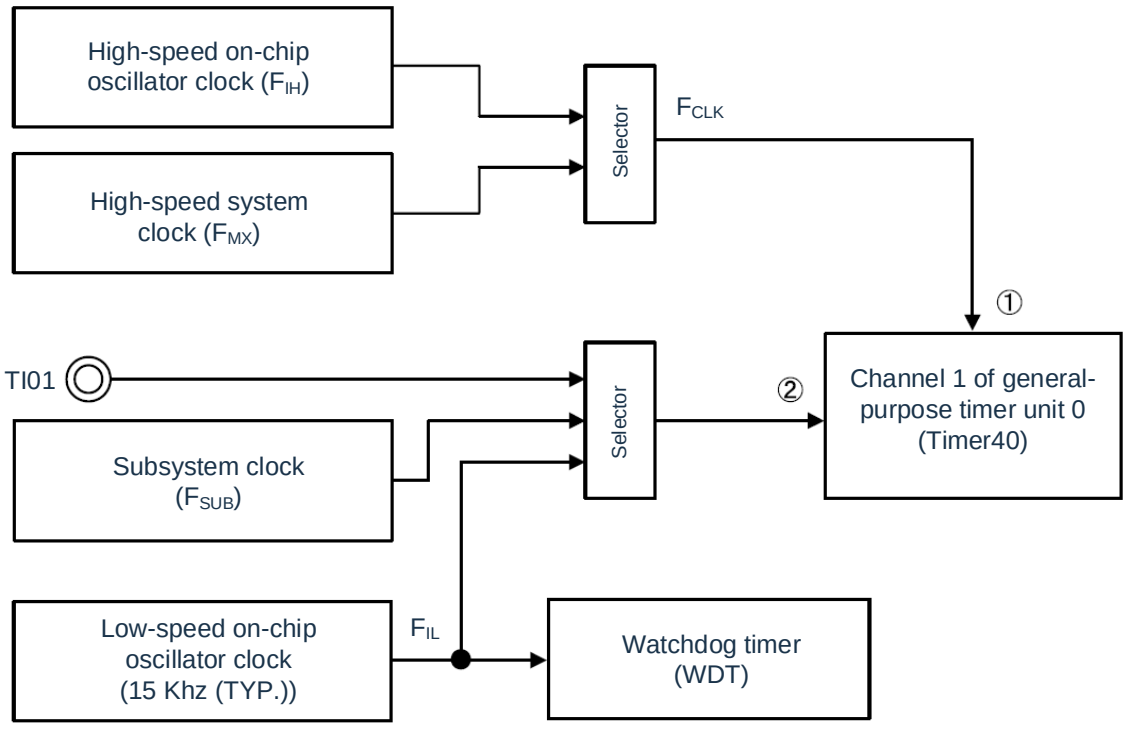
(2) Input to Chanel 1 of the Timer40:

① Timer input to channel 1 (TI01)

② Low-speed on-chip oscillator clock (F_{IL}: 15KHz(typ.))

③ Subsystem clock (F_{SUB})Note

Figure 22-3: Structure of frequency detection function



If the measurement result of the input pulse interval is an abnormal value, it can be judged as “clock frequency abnormality”. For the measurement method of the input pulse interval, refer to “5.8.4 Operation as input pulse interval measurement”.

Note: Only products with built-in subsystem clocks can be selected.

22.3.4.1 Timer input select register0(TIS0)

Refer to Section 5.3.8 for register descriptions.

22.3.5 A/D test function

The IEC60730 standard requires testing the A/D converter. This A/D test function is performed on the positive (+) reference voltage, negative (–) reference voltage, analog input channel (ANI), output voltage of the temperature sensor, and internal reference voltage for A/D conversion to confirm that the A/D converter is running normally.

The analog multiplexer can be confirmed by following these steps:

① Select the ANIx pin for A/D conversion using the ADS register.
② Perform A/D conversion for the ANIx pin (conversion result 1-1).
③ The positive (+) reference voltage of the A/D converter is selected for A/D conversion via the ADS register.
④ Perform A/D conversion of the positive (+) reference voltage of the A/D converter (conversion result 2-1).
⑤ Select the ANIx pin for A/D conversion using the ADS register.
⑥ Perform A/D conversion for the ANIx pin (conversion result 1-2).
⑦ Check that the conversion results 1-1 and 1-2 are equal.
⑧ Check that the A/D conversion result 2-1 is all zero
Using the procedure above can confirm that the analog multiplexer is selected and all wiring is connected.

Notice:

1. If the analog input voltage is variable during A/D conversion in steps ①-⑧above, 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.

22.3.5.1 Analog input channel specification register (ADS)

This register specifies the input channel of the analog voltage to be A/D converted.

ANIxx, temperature sensor output, internal reference voltage (1.45V) or positive (+) reference voltage can be measured via the A/D test function.

Refer to 11.2.7 for register descriptions.

22.3.6 Digital output signal level detection function for input/output pins

The IEC60730 standard requires confirmation that the I/O function is normal.

Input/output pin digital output signal level detection function reads the digital output level of a pin when the pin is in output mode.

22.3.6.1 Port mode select register (PMS)

This register selects whether to read the value of the port's output latch or the output level of the pin when the pin is in output mode (PMmn bit of the Port Mode Register (PMm) is "0").

The PMS register is set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of this register changes to "00H".

Table 22-6: Format of port mode select register (PMS)

Address: 4004087BH After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
PMS	0	0	0	0	0	0	0	PMS0

PMS0	Selection of reading data when the pin is in output mode
0	Reads the value of the Pmn register.
1	Read the digital output level of the pin.

Remark:

1. 1. If the digital output level of the pin is read, the read value is "0" for the pin that has been changed to a high impedance state by using the pulse output forced truncation function of Timer M.
2. m=0~3 n=0~7

22.3.7 Product unique identification register

The unique identification of the product is perfect for:

(1) Used as a serial number (e.g. USB character serial number or other terminal applications).

(2) Used as a password, this unique identifier is used in conjunction with a software encryption and decryption algorithm when writing flash memory to improve the security of the code in the flash memory.

(3) Used to activate a bootstrap process with a safety mechanism

The reference number provided by the 128-bit product unique identifier is unique to any BAT32 microcontroller in any case. Under any circumstances, the user cannot modify this identity.

Base address: 0x0050_0894

Offset address: 0x00

Read-only, and the value is written at the factory.

U_ID[31:0]

Offset address: 0x04

Read-only, and the value is written at the factory.

U_ID[63:32]

Offset address: 0x08

Read-only, and the value is written at the factory.

U_ID[95:64]

Offset address: 0x0C

Read-only, and the value is written at the factory.

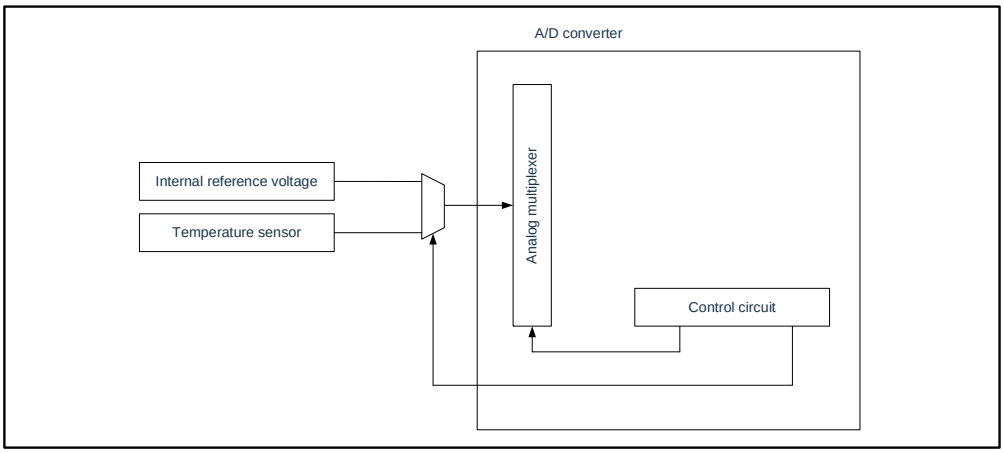
U_ID[127:96]

Chapter 23 Temperature sensor

23.1 Function of temperature sensor

The on-chip temperature sensor measures and monitors the core temperature of the product, thus ensuring reliable operation of the product. The voltage output by the temperature sensor is proportional to the core temperature, and there is a linear relationship between the voltage and temperature. Its output voltage is supplied to the ADC for conversion. Figure 23-1 shows a block diagram of a temperature sensor.

Figure 23-1: Temperature sensor block diagram



23.2 Temperature sensor register

23.2.1 Temperature sensor calibration data register TSN25

Address: 0x0050_066C

Symbol	15	0	After reset	R/W
TSN25	TSN25[11:0]		-	R

It is a read-only register for recording the calibration data1 of the temperature sensor, which is automatically loaded when power is turned on or reset is initiated, and each chip has its own calibration data.

23.3 Instructions for using temperature sensor

The temperature (T) is proportional to the sensor voltage output (Vs), so the temperature is calculated as follows:

T = (Vs - V1) / slope + 25℃

T: Measured temperature (℃)

Vs: Output voltage of the temperature sensor at temperature measurement (V)

V1: Voltage output at 25°C measured by temperature sensor (V)

Slope: Temperature slope of the temperature sensors(V/°C), slope = -3.5 mV/℃

Remark: Temperature sensors have low accuracy and are not recommended for use in applications where high accuracy is required.

Chapter 24 Option Bytes

24.1 Function of option bytes

Addresses 000C0H~000C3H, 500004H of the flash emory of the CMS32L032 form an option byte area.

Option bytes consist of user option byte (000C0H to 000C2H) and Flash data protection option byte (000C3H, 500004H). When powered on or reset is initiated, the specified function is set with reference to the option byte. When using this product, the following functions must be set by the option byte. For bits that do not have configuration capabilities, you cannot change the initial value.

Notice: Regardless of whether or not to use each function, you must set the option byte.

24.1.1 User option bytes (000C0H~000C2H)

(1) 000C0H

- Operation of watchdog timer
  - Enable or disable counter operation.
  - Enable or stop counter operation in sleep/deep sleep mode.
- Setting of watchdog timer overflow time
  - Setting of window open period of watchdog timer
- Setting of interval interrupt of watchdog timer
  - Whether or not to use the interval interrupt is selectable.

(2) 000C1H

- Setting of LVD operation mode
  - Interrupt & reset mode.
  - Reset mode.
  - Interrupt mode.
  - LVD off (by controlling the externally input reset signal on the RESETB pin)
- Setting of LVD detection level (V_{LVDH}, V_{LVDL}, V_{LVD})

Notice: When the supply voltage rises, the reset state must be maintained by voltage detection circuits or external resets before the supply voltage reaches the operating voltage range shown in the AC characteristics of the data sheet; When the supply voltage drops, it must be reset by transferring in deep sleep mode, voltage detection circuitry, or external reset before the supply voltage falls below the operating voltage range. The operating voltage range depends on the setting of the user option byte (000C2H).

(3) 000C2H

- Setting of the frequency of the high-speed on-chip oscillator
  - Select from 2MHz~32MHz, 48MHz, 64MHz.

24.1.2 Flash data protection option bytes (000C3H, 500004H)

- Control of flash data protection when debugging on-chip

  Level0: Read/write/erase operations on flash data are enabled via debugger.

  Level1: Chip erase operations on flash data via debugger are enabled, read/write operations are disabled.

  Level2: Operations on flash data via debugger are disabled.

24.2 Format of user option bytes

Table 24-1: Format of user option bytes (000C0H)

Address: 000C0H

Symbol	7	6	5	4	3	2	1	0
	WDTINT	WINDOW1	WINDOW0	WDTON	WDCS2	WDCS1	WDCS0	WDSTBYON

WDTINT	Interval interrupt of watchdog timer
0	Interval interrupt is not used.
1	When 75% of the overflow time + 1/2F_{IL} is reached, an interval interrupt is generated.

WINDOW1	WINDOW0	When watchdog timer window opens
0	-	Settings are disabled.
1	0	75%
1	1	100%

WDTON	Controlling counter operation of watchdog timer
0	Disable counter operation (stop counting after the reset is released).
1	Enable counter operation (start counting after the reset is released).

WDCS2	WDCS1	WDCS0	Overflow time of watchdog timer (F_{IL}=20KHz (MAX.))
0	0	0	$2^6/F_{IL}$ (3.2ms)
0	0	1	$2^7/F_{IL}$ (6.4ms)
0	1	0	$2^8/F_{IL}$ (12.8ms)
0	1	1	$2^9/F_{IL}$ (25.6ms)
1	0	0	$2^{11}/F_{IL}$ (102.4ms)
1	0	1	$2^{13}/F_{IL}$ (409.6ms)
1	1	0	$2^{14}/F_{IL}$ (819.2ms)
1	1	1	$2^{16}/F_{IL}$ (3276.8ms)

WDSTBYON	Counter operation control (sleep mode) of watchdog timer
0	In sleep mode, counter operations are stopped.
1	In sleep mode, counter operations are enabled.

Remark:

1. When the WDSTBYON bit is "0", regardless of the values of the WINDOW1 bit and the WINDOW0 bit, it is 100% during window opening.
2. F_{IL}: Low-speed on-chip oscillator clock frequency

Table 24-2: Format of user option bytes (000C1H) (1/4)

Address: 000C1H

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

- LVD settings (interrupt & reset mode)

Detect voltage			Setting value of option byte						
V_{LVDH}		V_{LVDL}	VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
Rising	Falling	Falling						LVIMDS1	LVIMDS0
1.98V	1.94V	1.84V	0	0	1	1	0	1	0
2.09V	2.04V					0	1		
3.13V	3.06V					0	0		
2.61V	2.55V	2.45V		1	0	1	0		
2.71V	2.65V					0	1		
3.75V	3.67V					0	0		
2.92V	2.86V	2.75V		1	1	1	0		
3.02V	2.96V					0	1		
4.06V	3.98V					0	0		
—			Settings other than above are prohibited.						

Notice: Bit4 must be written as "1".

Remark:

1. For details of LVD circuit, please refer to “Chapter 21 Voltage Detection Circuit”.
2. The detection voltage is a TYP value. For details, please refer to the LVD circuit characteristics in the data sheet.

Table 24-3: Format of user option bytes (000C1H) (2/4)

Address: 000C1H

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

- LVD setting (reset mode)

Detect voltage		Setting value of option byte						
V_{LVD}		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
Rising	Falling						LVIMDS1	LVIMDS0
1.88V	1.84V	0	0	1	1	1	1	1
1.98V	1.94V		0	1	1	0		
2.09V	2.04V		0	1	0	1		
2.50V	2.45V		1	0	1	1		
2.61V	2.55V		1	0	1	0		
2.71V	2.65V		1	0	0	1		
2.81V	2.75V		1	1	1	1		
2.92V	2.86V		1	1	1	0		
3.02V	2.96V		1	1	0	1		
3.13V	3.06V		0	1	0	0		
3.75V	3.67V		1	0	0	0		
4.06V	3.98V		1	1	0	0		
—		Settings other than above are prohibited.						

Notice: Bit4 must be written as "1".

Remark:

1. For details of LVD circuit, please refer to “Chapter 21 Voltage Detection Circuit”.
2. The detection voltage is a TYP value. For details, please refer to the LVD circuit characteristics in the data sheet.

Table 24-4: Format of user option bytes (000C1H) (3/4)

Address: 000C1H[Note]

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

- LVD setting (interrupt mode)

Detect voltage		Setting value of option byte						
V_{LVD}		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
Rising	Falling						LVIMDS1	LVIMDS0
1.88V	1.84V	0	0	1	1	1	0	1
1.98V	1.94V		0	1	1	0		
2.09V	2.04V		0	1	0	1		
2.50V	2.45V		1	0	1	1		
2.61V	2.55V		1	0	1	0		
2.71V	2.65V		1	0	0	1		
2.81V	2.75V		1	1	1	1		
2.92V	2.86V		1	1	1	0		
3.02V	2.96V		1	1	0	1		
3.13V	3.06V		0	1	0	0		
3.75V	3.67V		1	0	0	0		
4.06V	3.98V		1	1	0	0		
—		Settings other than above are prohibited.						

Note: Bit4 must be written as "1".

Remark:

1. For details of LVD circuit, please refer to “Chapter 21 Voltage Detection Circuit”.
2. The detection voltage is a TYP value. For details, please refer to the LVD circuit characteristics in the data sheet.

Table 24-5: Format of user option bytes (000C1H) (4/4)

Address: 000C1H

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

- Setting when LVD is OFF (external reset input using RESETB pin)

Detect voltage		Setting value of option byte						
V_{LVDH}		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
Rising	Falling						LVIMDS1	LVIMDS0
—	—	1	×	×	×	×	×	1
—		Settings other than above are prohibited.						

Notice:

1. Bit4 must be written as "1".
2. When the supply voltage rises, the reset state must be maintained by the voltage detection circuit or external reset before the supply voltage reaches the operating voltage range shown in the AC Characteristics of the datasheet; when the supply voltage falls, the reset state must be reset by the transfer of the sleep mode, the voltage detection circuit, or the external reset before the supply voltage falls below the operating voltage range. The operating voltage range depends on the setting of the user option byte (000C2H).

Remark:

1. ×: Ingore
2. For details of LVD circuit, please refer to “Chapter 21 Voltage Detection Circuit”.
3. The detection voltage is a TYP value. For details, please refer to the LVD circuit characteristics in the data sheet.

Table 24-6: Format of user option bytes (000C2H)

Address: 000C2H

7	6	5	4	3	2	1	0
1	1	1	FRQSEL4	FRQSEL3	FRQSEL2	FRQSEL1	FRQSEL0

FRQSEL4	FRQSEL3	FRQSEL2	FRQSEL1	FRQSEL0	High-speed on-chip oscillator clock frequency	
					F_{HOCO}	F_{IH}
0	1	0	0	0	64MHz	64MHz
0	0	0	0	0	48MHz	48MHz
0	1	0	0	1	64MHz	32MHz
0	0	0	0	1	48MHz	24MHz
0	1	0	1	0	64MHz	16MHz
0	0	0	1	0	48MHz	12MHz
0	1	0	1	1	64MHz	8MHz
0	0	0	1	1	48MHz	6MHz
0	1	1	0	0	64MHz	4MHz
0	0	1	0	0	48MHz	3MHz
0	1	1	0	1	64MHz	2MHz
Other than above					Disable settings.	

Notice: Bits 7 to 5 must be written as "1".

Remark: Operating frequency range and operating voltage range vary depending on each operating mode of the flash memory. For details, refer to AC Characteristics in the datasheet.

24.3 Format of flash data protection option bytes

The format of the flash data protection option bytes is as follows.

Table 24-7: Format of flash data protection option bytes (000C3H)

Address: 000C3H

Symbol	7	6	5	4	3	2	1	0
	OCDEN[7:0]							

Address: 500004H

Symbol	7	6	5	4	3	2	1	0
	OCDM[7:0]							

OCDM	OCDEN	Control of flash data protection
3C	C3	Manipulation of flash data via debugger is disabled.
Other than 3C	C3	Chip erase operation on flash data via debugger is enabled, read/write operation is disabled.
Other than above		Read/write/erase operations on flash data via debugger are enabled.

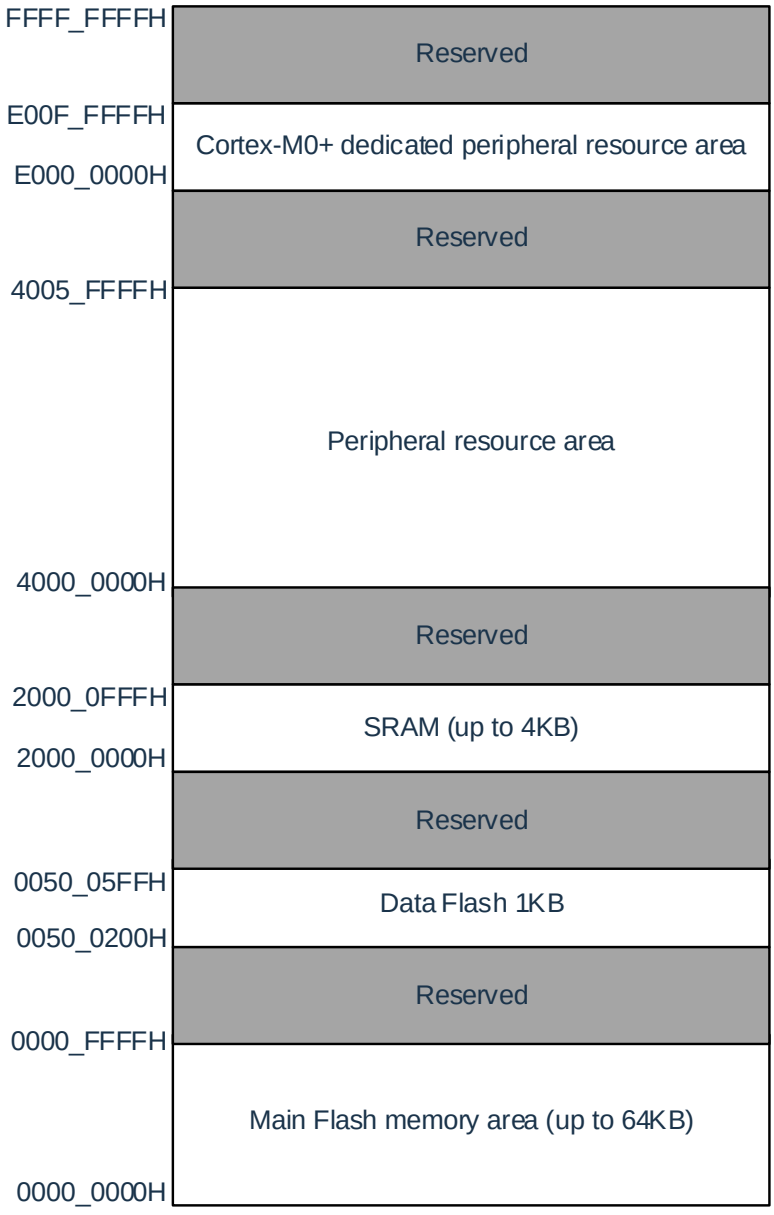
Notice: The 50_0004H address belongs to the data flash memory area. If you use this address for data storage, make sure that the value does not cause missetting of protection options.

Chapter 25 FLASH Control

25.1 FLASH control function description

This product contains a 64KByte FLASH memory, which is divided into 128 Sectors, each with a capacity of 512 Bytes. It can be used as program memory and data memory. This module supports erase, program and read operations for this memory.

25.2 Structure of FLASH memory



25.3 Registers for controlling FLASH

The registers that control FLASH are as follows:

- Flash write protection register (FLPROT)
- Flash operation control register (FLOPMD1, FLOPMD2)
- Flash erase mode control register (FLERMD)
- Flash status register (FLSTS)
- Flash chip erase time control register (FLCERCNT)
- Flash sector erase time control register (FLSERCNT)
- Flash write time control register (FLPROCNT)
- Flash mode time control register (FLNVSCNT/FLPRVCNT/FLERVCNT)

25.3.1 Flash write protection register (FLPROT)

Flash protection register is a register used to protect the FLASH operation control register.

Address: 0x40020020 After reset: 00000000H R/W

Symbol	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLPROT	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	PRKEY[7:1]							WRP

WRP	Operation register (FLOPMD1/FLOPMD2) write protection
0	Rewriting of FLOPMD1/ FLOPMD2 is not allowed.
1	Rewriting of FLOPMD1/ FLOPMD2 is allowed.

PRKEY[7:1]	WRP write protection
78h	Rewriting of WRP is allowed.
Other than above	Rewriting of WRP is not allowed.

25.3.2 FLASH operation control register (FLOPMD1, FLOPMD2)

Flash Operation Control Register is used to set the erase and write operations of FLASH.

Address: 0x40020004 After reset: 00000000H R/W

Symbol	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLOPMD1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	-	-	-	-	-	-	-	-	FLOPMD1[7:0]							

Address: 0x40020008 After reset: 00H R/W

Symbol	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLOPMD2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	-	-	-	-	-	-	-	-	FLOPMD2[7:0]							

FLOPMD1	FLOPMD2	MANIPULATION
55	AA	Erase
AA	55	Write
00	00	Read
Other than above		Disable settings

25.3.3 Flash erase control register (FLERMD)

Flash erase control register is used to set the type of FLASH erase operation.

Address: 0x4002000C After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
FLERMD	0	0	0	ERMD1	ERMD0	0	0	0

ERMD1	ERMD0	MANIPULATION
0	0	Sector erase, no hardware check after erase
1	0	Sector erase, hardware check after erase
0	1	Chip erase Note
1	1	Disable settings

Note: Chip erase only erases the code flash area, not the data flash area. And chip erase does not support hardware check.

25.3.4 Flash status register (FLSTS)

The status of the FLASH controller can be queried through the status register.

Address: 0x40020000 After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
FLSTS	0	0	0	0	0	EVF[Note]	0	OVF[Note]

OVF	FLASH erase/write operation complete flag
0	FLASH erase/write operation is not completed
1	FLASH erase/write operation is completed

Note: The OVF needs to be cleared by writing "1" in software. If it is not cleared, the next erase operation cannot be performed.

EVF	FLASH erase hardware check error flag
0	No hardware check error after FLASH erase
1	Hardware check error occurs after FLASH erase

Note: EVF needs to be cleared by writing "1" in the software.

25.3.5 Flash chip erase time control register (FLCERCNT)

FLCERCNT register enables to set the FLASH full chip erase time.

Address: 0x40020010 After reset: Undefined R/W

Symbol	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLCERCNT	load	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	-	-	-	-	-	-	FLCERCNT[9:0]									

LOAD	Select erase time[Note]
0	Erase time is set using the hardware
1	Erase time is set using the software (FLCERCNT[9:0])

Note: When the master clock is an on-chip high-speed OCO or the external input clock is ⩽16M, the hardware setting time can be used without setting FLCERCNT.

FLCERCNT[9:0]	Software erase time setting
Chip erase time = (CERCNT*2048*T_{FCLK}), which meets the hardware requirement of >30ms	

25.3.6 Flash sector erase time control register (FLSERCNT)

LSERCNT register enables to set the FLASH full erase time.

Address: 0x40020014 After reset: Undefined R/W

Symbol	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLSERCNT	load	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	-	-	-	-	-	-	FLSERCNT[9:0]									

Load	Select erase time[Note]
0	Erase time is set using the hardware
1	Erase time is set using the software (FLSERCNT[9:0])

Note: When the master clock is an on-chip high-speed OCO or the external input clock is ⩽16M, the time can be set in hardware without setting FLSERCNT.

FLSERCNT[9:0]	Software erase time setting
Sector erase time = (SERCNT*256*T_{FCLK}), which meets the hardware requirement of >2ms	

25.3.7 Flash write time control register (FLPROCNT)

FLPROCNT register enables to set the FLASH WORD write time.

Address: 0x4002001C After reset: Undefined R/W

Symbol	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLPROCNT	Load1	-	-	FLPGSCNT[12:0]												
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Load0	-	-	-	-	-	-	FLPROCNT[8:0]								

Load0	Write time setting (T_{PROG}) Note
0	Write time is set using the hardware
1	Write time is set using the software (FLPROCNT[8:0])

Note: When the master clock is an on-chip high-speed OCO or external input clock <=16M, the time can be set in hardware without setting FLPROCNT.

FLPROCNT[8:0]	Software write time setting
Write time = (PROCNT*T_{FCLK}), which meets the hardware requirement of >7us	

Load1	Write action setup time (T_{PGS}) setting Note
0	Write action setup time using the hardware
1	Write action setup time using the software (FLPGSCNT12:0])

Note: When the master clock is an on-chip high-speed OCO or the external input clock is ⩽16M, you can set the time in hardware without setting FLPGSCNT.

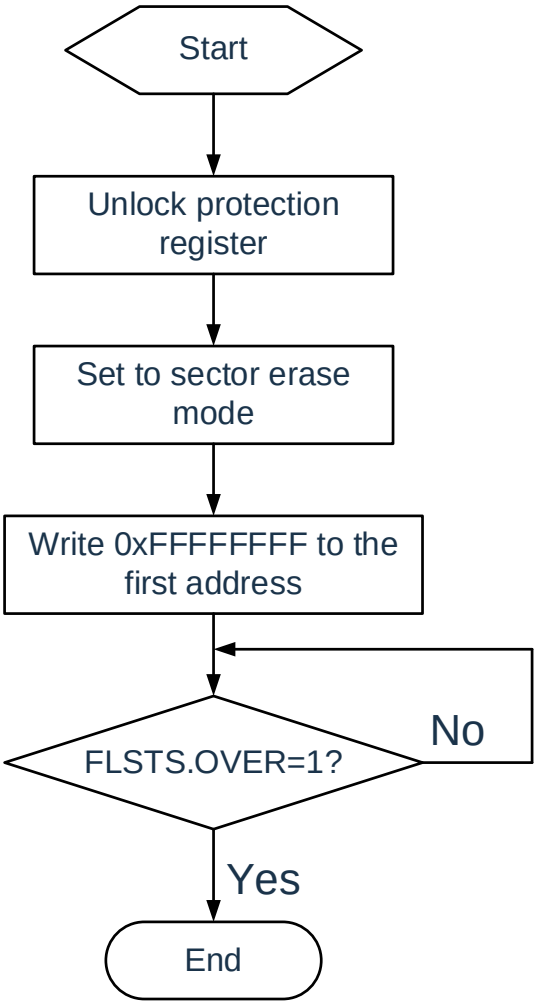
FLPGSCNT[12:0]	Write action setup time using the software setting
Write action setup time = (PGSCNT*T_{FCLK}) , Write bytes meet the hardware requirements of >70us, Write half a word meet the hardware requirements of >40us	

25.4 How to operate FLASH

25.4.1 Sector erase

The erase time is realized by the hardware or can be configured by FLSERCNT. The operation flow is as follows.

1) Set FLERMD.ERMD0 to 1'b0, select the sector erase mode, and set the value of ERMD1 according to whether or not hardware check is required.
2) Set FLPROT to 0xF1, unprotect FLOPMD. Then set FLOPMD1 to 0x55, FLOPMD2 to 0xAA, and
3) Write arbitrary data to the first address of the erase target sector. Example: * ((unsigned long *)0x00000200)=0xffffffff.
4) Query the status register FLSTS.OVF through software, when OVF=1, it means the erase operation is completed.
5) If hardware check after erase is set (ERMD1=1), you can determine whether the verification is correct by software for FLSTS.EVF.
6) Before the next operation, set the software to "1" to clear the FLSTS.



25.4.2 Chip erase

Chip erase, and the erase time are implemented by hardware and can also be configured via FLCERCNT. The operation process is as follows

1) Set FLERMD. ERMD0 to 1'b1, and select chip erase mode;
2) Set FLPROT to 0xF1 to unprotect FLOPMD. Then set FLOPMD1 to 0x55 and FLOPMD2 to 0xAA
3) Write arbitrary data to any address in the flash area of the code.
4) Query the status register FLSTS.OVF through software, when OVF=1, it means the erase operation is completed.
5) Before the next operation, set the software to "1" to clear the FLSTS.

25.4.3 Word program

Word programming and write time are implemented by hardware and can also be configured via PROCNT. The operation process is as follows:

1) Set FLPROT to 0xF1, unprotect FLOPMD. Then set FLOPMD1 to 0xAA, FLOPMD2 to 0x55, and
2) Write the corresponding data to the target address.
3) Query the status register FLSTS.OVF through software, when OVF=1, it means the write operation is completed.
4) Before the next operation, set the software to "1" to clear the FLSTS.

25.5 Flash read

The fastest fetch frequency supported by the built-in FLASH is 32 MHz. when the HCLK frequency exceeds 32 MHz, the hardware will insert 1 wait cycle when the CPU accesses the FLASH.

25.6 Cautions for FLASH operation

- Flash memory has strict time requirements for the control signal of erasing and programming operation, and the timing of the control signal is not qualified, which will cause the erase operation and programming operation to fail. The setting of the erase and write parameters can be implemented by hardware, or it can be modified by modifying the parameter registers; When using on-chip high-speed OCO, MAINOSC/ external input clock = 16M, it is recommended to use hardware-set erase and write parameters without setting parameter registers.
- If the erase/write operation is executed from FLASH, the CPU stops fetching and the hardware automatically waits for the completion of the operation to proceed to the next instruction. If the operation is executed from RAM, the CPU will not stop fetching and can continue the next instruction.
- If the CPU executes an instruction to enter deep sleep while the FLASH is in programming operation, the system waits for the programming action to end before entering deep sleep.

Appendix Revision History

Version	Date	Revised content
V1.0.0	June 2023	Initial version
V1.0.1	July 2023	Corrected text error in Section 21.5
V1.0.2	August 2023	1. Delete the section of Table 2-2: Digital mapping of pin functions for the simplified I^2C 2. Corrected the description of the bus in the 3.1 Overview section 3. Corrected Figure 5-44: Example of register contents setting for multiple PWM output function (slave channel) 4. Removed the description of the WDTE register execution of bit-manipulation instructions in 10.1, 10.3.1, 10.4.1 5. Deleted the section of Chapter 12 on General-Purpose Serial Communication Units on Simplified I^2C 6. Modify Figure 0-1: Block diagram of general-purpose serial communication unit 7. Corrected the software erase time of sections 25.3.5 and 25.3.6 8. Corrected the software write time setting and the software write action setup time setting in Section 25.3.7