



BAT32G135 User Manual

Ultra-low power 32-bit microcontrollers based on ARM® Cortex®-M0+

V1.0.5

Please note the following CMS IP policy

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

* The name and logo of Cmsemicon 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 authorized for use as critical components in lifesaving, life-sustaining devices or systems. The Company reserves the right to modify the products without prior notice. For the latest information, please visit the official website at www.mcu.com.cn.

Documentation Instructions

This manual is a [technical reference manual](#) for the BAT32G135 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:

BAT32G135xx: BAT32G135_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	17
1.1 Overview	17
1.2 Cortex-M0+ core features	17
1.3 Debugging features.....	18
1.4 SWD interface pins	19
1.5 ARM reference document.....	20
Chapter 2 Pin Functions	21
2.1 Port function	21
2.2 Port multiplexing function.....	21
2.3 Registers for controlling port functions	22
2.3.1 Port mode register (PMxx).....	24
2.3.2 Port register (Pxx).....	25
2.3.3 Port set control register (PSETxx).....	26
2.3.4 Port clear control register (PCLRxx).....	27
2.3.5 Pull-up resistor selection register (PUxx)	28
2.3.6 Pull-down resistor selection register (PDxx)	29
2.3.7 Port output mode register (POMxx).....	30
2.3.8 Port mode control register (PMCxx).....	31
2.3.9 Port multiplexing configuration register (PxxCFG).....	32
2.3.10 Port input multiplexing configuration registers (TI10PCFG, TI11PCFG, TI12PCFG, TI13PCFG, INT0PCFG, INT1PCFG, INT2PCFG, INT3PCFG, SDI0PCFG, SCLKI0PCFG, SS0PCFG, SDI20PCFG, SCLKI20PCFG, SDAA0PCFG, SCLA0PCFG, RXD1PCFG).....	37
2.3.11 SPI port multiplexing configuration register (SPIPCFG)	41
2.4 Handling of unused pins	42
2.5 Register settings when using multiplexing function	43
2.5.1 Basic idea when using multiplexing output function.....	43
2.5.2 Example of register setting for used port functions and multiplexing functions	44
2.5.3 EPWM port configuration methods.....	60
Chapter 3 System Structure	61
3.1 Overview	61
3.2 System address partitioning.....	62
Chapter 4 Clock Generation Circuit	64
4.1 Function of clock generation circuit	64
4.2 Structure of clock generation circuit.....	66
4.3 Registers for controlling clock generation circuit	69
4.3.1 Clock operation mode control register (CMC).....	69

4.3.2	System clock control register (CKC).....	71
4.3.3	Clock operation status control register (CSC).....	72
4.3.4	Oscillation stabilization time counter status register (OSTC).....	73
4.3.5	Oscillation stabilization time select register (OSTS).....	75
4.3.6	Peripheral enable registers 0, 1 (PER0, PER1).....	76
4.3.7	Subsystem clock supply mode control register (OSMC).....	80
4.3.8	High-speed on-chip oscillator frequency select register (HOCODIV).....	81
4.3.9	High-speed on-chip oscillator trimming register (HIOTRM).....	82
4.3.10	Subsystem clock selection register (SUBCKSEL).....	83
4.4	System clock oscillation circuit.....	84
4.4.1	X1 oscillation circuit.....	84
4.4.2	XT1 oscillation circuit.....	84
4.4.3	High-speed on-chip oscillator.....	88
4.4.4	Low-speed on-chip oscillator.....	88
4.5	Operation of clock generation circuit.....	89
4.6	Clock control.....	91
4.6.1	Example of setting up a high-speed on-chip oscillator.....	91
4.6.2	Example of setting X1 oscillation circuit.....	93
4.6.3	Example of controlling XT1 oscillation clock.....	94
4.6.4	CPU clock status transition diagram.....	95
4.6.5	Conditions before CPU clock transfer and post-transfer processing.....	100
4.6.6	Time required to switch CPU clock and main system clock.....	102
4.6.7	Conditions before clock oscillation is stopped.....	103
Chapter 5	General-Purpose Timer Unit Timer4.....	104
5.1	Function of general-purpose timer unit.....	105
5.1.1	Independent channel operation function.....	105
5.1.2	Multi-channel linkage operation functions.....	107
5.1.3	8-bit timer operation function (channels 1 and 3 of unit 0 only).....	108
5.1.4	LIN-bus supporting function (channel 3 of unit 0 only).....	108
5.2	Structure of general-purpose timer unit.....	109
5.2.1	Register list of general-purpose timer unit 0.....	112
5.2.2	Register list of general-purpose timer unit 1.....	113
5.2.3	Timer count register mn (TCRmn).....	114
5.2.4	Timer data register mn (TDRmn).....	115
5.3	Registers for controlling general-purpose timer unit.....	116
5.3.1	Peripheral enable register 0 (PER0).....	117
5.3.2	Timer clock select register m (TPSm).....	118
5.3.3	Timer mode register mn (TMRmn).....	121
5.3.4	Timer status register mn (TSRmn).....	126

5.3.5	Timer channel enable status register m (TEm).....	127
5.3.6	Timer channel start register m (TSm).....	128
5.3.7	Timer channel stop register m (TTm).....	129
5.3.8	Timer input output select register (TIOS0).....	130
5.3.9	Timer output enable register m (TOEm).....	131
5.3.10	Timer output register m (TOm).....	132
5.3.11	Timer output level register m (TOLm).....	133
5.3.12	Timer output mode register m (TOMm).....	134
5.3.13	Noise filter enable register 1 (NFEN1).....	135
5.3.14	Noise filter enable register 2 (NFEN2).....	136
5.3.15	Registers controlling port functions of timer input/output pins.....	137
5.4	Basic rules of general-purpose timer unit.....	138
5.4.1	Basic rules of multi-channel linkage operation function.....	138
5.4.2	Basic rules of 8-bit timer operation function (channels 1 and 3 of unit 0 only).....	140
5.5	Operation of counter.....	141
5.5.1	Count clock (F _{TCCLK}).....	141
5.5.2	Start timing of counter.....	143
5.5.3	Operation of counter.....	144
5.6	Channel output (TOm pin) control.....	149
5.6.1	TOm pin output circuit configuration.....	149
5.6.2	TOm pin output configuration.....	150
5.6.3	Cautions for channel output operation.....	151
5.6.4	One-time operation of TOm bit.....	155
5.6.5	Timer interrupt and TOm pin output when counting starts.....	156
5.7	Control of timer input (TImn).....	157
5.7.1	Structure of TImn pin input circuit.....	157
5.7.2	Noise filter.....	158
5.7.3	Cautions on channel input operation.....	159
5.8	Independent channel operation function of general-purpose timer unit.....	160
5.8.1	Operation as interval timer/square wave output.....	160
5.8.2	Operation as external event counter.....	164
5.8.3	Operation as frequency divider.....	167
5.8.4	Operation as input pulse interval measurement.....	171
5.8.5	Operation as input signal high-/low-level width measurement.....	174
5.8.6	Operation as delay counter.....	178
5.9	Multi-channel linkage operation function for general purpose timer unit.....	181
5.9.1	Operation as single trigger pulse output function.....	181
5.9.2	Operation as PWM function.....	188
5.9.3	Operation as multiple PWM output function.....	195

Chapter 6	EPWM Output Control Circuit.....	203
6.1	Structure of output control circuit.....	203
6.2	Registers for controlling EPWM output control circuit.....	204
6.2.1	Peripheral enable register 1 (PER1).....	205
6.2.2	EPWM input source select register (EPWMSRC).....	205
6.2.3	EPWM output control register (EPWMCTL).....	205
6.2.4	EPWM force truncated input select register (EPWMSTC).....	206
6.2.5	EPWM force truncated output select register (EPWMSTL).....	207
6.2.6	EPWM status register (EPWMSTR).....	207
6.2.7	Registers controlling port functions of EPWM output pins.....	208
6.3	Operation of EPWM output control circuit.....	209
6.3.1	Initial setup.....	209
6.3.2	Normal operation.....	210
6.3.3	Force truncation processing.....	210
6.4	Control example of brushless DC motor.....	212
6.4.1	Example of hardware connections.....	212
6.4.2	Control timing of three-phase brushless DC motors.....	213
6.4.3	Examples of register setting.....	214
6.5	Example of stepper motor control.....	215
6.5.1	Example of hardware connection.....	215
6.5.2	Control method.....	216
6.5.3	Example of register setting.....	217
Chapter 7	Real-Time Clock.....	218
7.1	Function of real-time clock.....	218
7.2	Structure of real-time clock.....	218
7.3	Registers for controlling real-time clock.....	220
7.3.1	Peripheral enable register 0 (PER0).....	221
7.3.2	Real-time clock selection register (RTCCCL).....	222
7.3.3	Real-time clock control register 0 (RTCC0).....	223
7.3.4	Real-time clock control register 1 (RTCC1).....	224
7.3.5	Clock error correction register (SUBCUD).....	226
7.3.6	Second count register (SEC).....	227
7.3.7	Minute count register (MIN).....	227
7.3.8	Hour count register (HOUR).....	228
7.3.9	Day count register (DAY).....	230
7.3.10	Week count register (WEEK).....	231
7.3.11	Month count register (MONTH).....	232
7.3.12	Year count register (YEAR).....	232
7.3.13	Alarm minute register (ALARMWM).....	233

7.3.14	Alarm hour register (ALARMWH)	233
7.3.15	Alarm week register (ALARMWW)	234
7.3.16	Port mode register and port register	234
7.4	Operation of real-time clock	235
7.4.1	Start of real-time clock operation	235
7.4.2	Shifting to sleep mode after starting operation	236
7.4.3	Reading/writing real-time clock	237
7.4.4	Setting alarm of real-time clock	239
7.4.5	1Hz output of real-time clock	240
7.4.6	Example of clock error correction of real-time clock	241
Chapter 8	15-Bit Interval Timer	243
8.1	Function of 15-bit interval timer	243
8.2	Structure of 15-bit interval timer	243
8.3	Registers for controlling 15-bit interval timer	244
8.3.1	Peripheral enable register 0 (PER0)	244
8.3.2	Real-time clock selection register (RTCCL)	245
8.3.3	15-bit interval timer control register (ITMC)	246
8.4	Operation of 15-bit interval timer	247
8.4.1	Operation timing of 15-bit interval timer	247
8.4.2	Start of count operation and re-enter to sleep mode after returned from sleep mode	248
Chapter 9	Clock Output/Buzzer Output Controller	249
9.1	Function of clock output/buzzer output controller	249
9.2	Structure of clock output/buzzer output controller	251
9.3	Registers for controlling clock output/buzzer output controller	251
9.3.1	Clock output select register n (CKSn)	251
9.3.2	Registers for controlling clock output/buzzer output port functions	253
9.4	Operation of clock output/buzzer output controller	254
9.4.1	Operation as output pin	254
9.5	Cautions for clock output/buzzer output controller	254
Chapter 10	Watch Dog Timer	255
10.1	Function of watchdog timer	255
10.2	Structure of watch dog timer	255
10.3	Registers for controlling watchdog timer	257
10.3.1	Watchdog timer enable register (WDTE)	257
10.3.2	LOCKUP control register (LOCKCTL)	258
10.4	Operation of watchdog timer	259
10.4.1	Operational control of watchdog timer	259
10.4.2	Setting overflow time of watchdog timer	260

10.4.3	Setting window open period of watchdog timer.....	261
10.4.4	Setting watchdog timer interval interrupt.....	262
10.4.5	Operation of watchdog timer during LOCKUP	262
Chapter 11 A/D Converter		263
11.1	Function of A/D converter	263
11.2	Registers for controlling A/D converter	265
11.2.1	Peripheral enable register 0 (PER0).....	266
11.2.2	A/D converter mode register 0 (ADM0)	267
11.2.3	A/D converter mode register 1 (ADM1)	272
11.2.4	A/D converter mode register 2 (ADM2)	273
11.2.5	A/D converter trigger mode register (ADTRG).....	274
11.2.6	Analog input channel specification register (ADS).....	275
11.2.7	12-bit A/D conversion result register (ADCR).....	277
11.2.8	8-bit A/D conversion result register (ADCRH).....	278
11.2.9	Conversion result comparison upper limit setting register (ADUL)	278
11.2.10	Conversion result comparison lower limit setting register (ADLL)	279
11.2.11	A/D converter sampling time control register (ADNSMP)	280
11.2.12	A/D sample time extension register (ADSMPWAIT)	282
11.2.13	A/D test register (ADTES).....	283
11.2.14	A/D status register (ADFLG).....	284
11.2.15	A/D charge/discharge control register (ADNDIS).....	285
11.2.16	Registers controlling port functions of analog input pins.....	285
11.3	Input voltage and conversion results	286
11.4	Operation mode of A/D converter	287
11.4.1	Software trigger mode (select mode, continuous conversion mode).....	287
11.4.2	Software trigger mode (select mode, single conversion mode).....	288
11.4.3	Software trigger mode (scan mode, sequential conversion mode).....	289
11.4.4	Software trigger mode (scan mode, single conversion mode).....	290
11.4.5	Hardware trigger no-wait mode (select mode, sequential conversion mode).....	291
11.4.6	Hardware trigger no-wait mode (select mode, single conversion mode).....	292
11.4.7	Hardware trigger no-wait mode (scan mode, sequential conversion mode).....	293
11.4.8	Hardware trigger no-wait mode (scan mode, single conversion mode).....	294
11.4.9	Hardware trigger wait mode (select mode, sequential conversion mode).....	295
11.4.10	Hardware trigger wait mode (select mode, single conversion mode).....	296
11.4.11	Hardware trigger wait mode (scan mode, sequential conversion mode).....	297
11.4.12	Hardware trigger wait mode (scan mode, single conversion mode).....	298
11.5	A/D converter setup flowchart.....	299
11.5.1	Software trigger mode settings.....	299
11.5.2	Hardware trigger no-wait mode settings.....	300

11.5.3	Hardware trigger wait mode settings	301
11.5.4	Settings when selecting the output voltage/internal reference voltage of the temperature sensor	302
11.5.5	Test mode settings	303
Chapter 12 Comparator		304
12.1	Function of comparator	304
12.2	Structure of comparator	305
12.3	Registers for controlling comparator	307
12.3.1	Peripheral enable register 1 (PER1).....	308
12.3.2	Comparator mode configuration register (COMPMDR)	309
12.3.3	Comparator filter control register (COMPFIR).....	310
12.3.4	Comparator output control register (COMPOCR)	312
12.3.5	Comparator built-in reference voltage control register (CVRCTL).....	314
12.3.6	Comparator built-in reference voltage selection register (CiRVM).....	315
12.3.7	Comparator 0 input signal selection control register (CMPSEL0).....	316
12.3.8	Comparator 1 input signal selection control register (CMPSEL1).....	317
12.3.9	Registers controlling port functions of analog input pins.....	318
12.4	Operation instructions	319
12.4.1	Digital filter of comparator i (i=0,1)	321
12.4.2	Comparator i interrupts (i=0,1).....	321
12.4.3	Event signals output to the linkage controller (EVENTC)	322
12.4.4	Output of comparator i (i=0,1).....	323
12.4.5	Stop and enable comparator clock supply	323
Chapter 13 Programmable Gain Amplifier (PGA).....		324
13.1	Function of programmable gain amplifier	324
13.2	Structure of programmable gain amplifier.....	325
13.3	Registers for programmable gain amplifier.....	326
13.3.1	Peripheral enable register 1 (PER1).....	326
13.3.2	Programmable gain amplifier control register (PGAnCTL).....	327
13.3.3	Registers controlling port functions of analog input pins.....	327
13.4	Operation of programmable gain amplifier	328
13.4.1	Starting operation steps of programmable gain amplifier	328
13.4.2	Stopping operation steps of programmable gain amplifier.....	329
Chapter 14 Universal Serial Communication Unit		330
14.1	Function of universal serial communication unit.....	331
14.1.1	3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)	331
14.1.2	UART (UART0~UART2)	332
14.1.3	Simplified I ² C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21).....	333

14.2 Structure of universal serial communication unit.....	334
14.2.1 Shift register.....	337
14.2.2 Low 8 bits or low 9 bits of serial data register mn (SDRmn).....	338
14.3 Registers for controlling universal serial communication unit.....	340
14.3.1 Peripheral enable register 0 (PER0).....	341
14.3.2 Serial clock select register m (SPSm).....	342
14.3.3 Serial mode register mn (SMRmn).....	343
14.3.4 Serial communication operation setting register mn (SCRmn).....	345
14.3.5 Serial data register mn (SDRmn).....	347
14.3.6 Serial flag clear trigger register mn (SIRmn).....	349
14.3.7 Serial status register mn (SSRmn).....	350
14.3.8 Serial channel start register m (SSm).....	352
14.3.9 Serial channel stop register m (STm).....	353
14.3.10 Serial channel enable status register m (SEm).....	354
14.3.11 Serial output enable register m(SOEm).....	355
14.3.12 Serial output register m (SOM).....	356
14.3.13 Serial output level register m (SOLm).....	357
14.3.14 Input switching control register (ISC).....	359
14.3.15 Noise filter enable register 0 (NFEN0).....	360
14.3.16 Registers controlling port functions of serial input/output pins.....	361
14.4 Run stop mode.....	362
14.4.1 Stopping the operation by units.....	362
14.4.2 Stopping the operation by channels.....	363
14.5 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21) communication operation.....	364
14.5.1 Master transmission.....	365
14.5.2 Master reception.....	374
14.5.3 Master transmission and reception.....	382
14.5.4 Slave transmission.....	390
14.5.5 Slave reception.....	398
14.5.6 Slave transmission and reception.....	404
14.5.7 Calculation of transfer clock frequency.....	413
14.5.8 Procedure for handling errors during 3-wire serial I/O communication (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21).....	415
14.6 Operation of clock-synchronous serial communication with slave selection input function.....	416
14.6.1 Slave transmission.....	419
14.6.2 Slave reception.....	429
14.6.3 Slave transmission and reception.....	436
14.6.4 Calculation of transfer clock frequency.....	446
14.6.5 Procedure for handling errors during clock-synchronous serial communication with the slave selection input function.....	447

14.7 Operation of UART (UART0~UART2) communication.....	448
14.7.1 UART transmission.....	449
14.7.2 UART reception.....	458
14.7.3 Calculation of baud rate.....	465
14.7.4 Handling steps when an error occurs during UART (UART0~UART 2) communication.....	469
14.8 Operation of LIN communication.....	470
14.8.1 LIN transmission.....	470
14.8.2 LIN reception.....	473
14.9 Simplified I ² C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication operation.....	478
14.9.1 Address field transmission.....	480
14.9.2 Data transmission.....	485
14.9.3 Data reception.....	488
14.9.4 Generation of stop condition.....	492
14.9.5 Calculation of transfer rate.....	493
14.9.6 Processing steps when an error occurs during simplified I ² C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication.....	495
Chapter 15 Serial Interface SPI.....	496
15.1 Serial interface SPI function.....	496
15.2 Structure of serial interface SPI.....	496
15.3 Registers for controlling serial interface SPI.....	497
15.3.1 Peripheral enable register 0 (PER0).....	498
15.3.2 SPI operation mode register (SPIM).....	499
15.3.3 SPI clock selection register (SPIC).....	500
15.3.4 Transmit buffer register (SDRO).....	501
15.3.5 Receive buffer register (SDRI).....	501
15.3.6 Registers controlling port functions of SPI pins.....	502
15.4 Operation of serial interface SPI.....	503
15.4.1 Master transmission and reception.....	504
15.4.2 Master reception.....	507
15.4.3 Slave transmission and reception.....	510
15.4.4 Slave reception.....	513
Chapter 16 Serial Interface IICA.....	516
16.1 Function of serial interface IICA.....	516
16.2 Structure of serial interface IICA.....	519
16.3 Registers for controlling serial interface IICA.....	522
16.3.1 Peripheral enable register 0 (PER0).....	523
16.3.2 IICA control register n0 (IICCTLn0).....	523
16.3.3 IICA status register n(IICSn).....	528
16.3.4 IICA flag register n(IICFn).....	530

16.3.5	IICA control register n1(IICCTLn1)	532
16.3.6	IICA low-level width setting register n(IICWLn)	534
16.3.7	IICA high-level width setting register n(IICWHn)	534
16.3.8	Registers controlling port functions of IICA pins.....	535
16.4	Function of I ² C-bus mode	536
16.4.1	Pin structure	536
16.4.2	Setting transfer clock via IICWLn and IICWHn registers	537
16.5	Definition and control method of I ² C-bus	539
16.5.1	Start condition	540
16.5.2	Address	541
16.5.3	Transfer direction specification	541
16.5.4	Acknowledge (ACK).....	542
16.5.5	Stop condition	543
16.5.6	Wait	544
16.5.7	Method of releasing wait state	546
16.5.8	Generation timing and waiting control of interrupt requests (INTIICAn)	547
16.5.9	Detection method for address matching.....	548
16.5.10	Error detection	548
16.5.11	Extension code	549
16.5.12	Arbitration.....	550
16.5.13	Wake-up function	552
16.5.14	Communication reservation.....	555
16.5.15	Cautions	559
16.5.16	Communication operation.....	560
16.5.17	Timing of I ² C interrupt request (INTIICAn) generation	568
16.6	Timing diagram.....	589
Chapter 17	IrDA	606
17.1	Function of IrDA	606
17.2	Registers for controlling IrDA	607
17.2.1	Peripheral enable register 0 (PER0).....	607
17.2.2	IrDA control register (IRCR).....	608
17.3	Operation of IrDA	609
17.3.1	Operating steps for IrDA communication.....	609
17.3.2	Transmission.....	610
17.3.3	Reception.....	610
17.3.4	High level pulse width selection.....	611
17.4	Cautions on using IrDA	611
Chapter 18	Enhanced DMA	612
18.1	Function of DMA	612

18.2 Structure of DMA.....	614
18.3 Registers for controlling DMA	615
18.3.1 DMA control data areas and DMA vector table areas allocation.....	616
18.3.2 Control data allocation	617
18.3.3 Vector table	619
18.3.4 Peripheral enable register 1 (PER1).....	621
18.3.5 DMA control register j(DMACRj) (j=0~23)	621
18.3.6 DMA block size register j (DMBLSj) (j=0~23).....	623
18.3.7 DMA transfer count register j(DMACTj) (j=0~23)	624
18.3.8 DMA transfer count reload register j(DMRLDj) (j=0~23).....	625
18.3.9 DMA source address register j(DMSARj) (j=0~23)	626
18.3.10 DMA destination address register j(DMDARj) (j=0~23).....	626
18.3.11 DMA boot enable register i (DMAENi) (i=0~2)	627
18.3.12 DMA base address register (DMABAR)	629
18.4 DMA operation	630
18.4.1 Boot source.....	630
18.4.2 Normal mode	631
18.4.3 Repeat mode	634
18.4.4 Chain transfer	637
18.5 Cautions on using DMA	639
18.5.1 DMA control data and vector table settings.....	639
18.5.2 DMA control data area and DMA vector table area allocation	639
18.5.3 Number of execution clocks for DMA	640
18.5.4 DMA response time	641
18.5.5 DMA boot source	641
18.5.6 Operation in standby mode	642
Chapter 19 Linkage Controller (EVENTC)	643
19.1 Function of EVENTC.....	643
19.2 Structure of EVENTC	643
19.3 Control registers.....	644
19.3.1 Output target selection register n (ELSELRn) (n=00~14).....	645
19.4 Operation of EVENTC.....	648
Chapter 20 Interrupt Function	649
20.1 Types of interrupt function.....	649
20.2 Interrupt source and structure.....	649
20.3 Registers controlling interrupt function	653
20.3.1 Interrupt request flag registers (IF00 to IF31).....	653
20.3.2 Interrupt mask flag register (MK00~MK31)	654
20.3.3 External interrupt rising edge enable register (EGP0), External interrupt falling edge enable	

register (EGN0)	657
20.4 Operation of interrupt handling	658
20.4.1 Reception of maskable interrupt requests	658
20.4.2 Reception of non-maskable interrupt requests	658
Chapter 21 Key Interrupt Function	659
21.1 Function of key interrupt	659
21.2 Structure of key interrupt.....	659
21.3 Registers for controlling key interrupt	661
21.3.1 Key return mode register (KRM).....	661
21.3.2 Port mode register (PMx).....	662
Chapter 22 Standby Function.....	663
22.1 Standby function	663
22.2 Sleep mode	664
22.2.1 Setting of sleep mode	664
22.2.2 Sleep mode release	667
22.3 Deep sleep mode	667
22.3.1 Setting of deep sleep mode	667
22.3.2 Deep sleep mode release.....	669
Chapter 23 Reset Function	671
23.1 Register for confirming reset sources	676
23.1.1 Reset control flag register (RESF).....	676
Chapter 24 Power-On Reset Circuit.....	679
24.1 Function of power-on reset circuit.....	679
24.2 Structure of power-on reset circuit.....	680
24.3 Operation of power-on reset circuit.....	680
Chapter 25 Voltage Detection Circuit	684
25.1 Function of voltage detection circuit	684
25.2 Structure of voltage detection circuit.....	685
25.3 Registers for controlling voltage detection circuit	686
25.3.1 Voltage detection register (LVIM).....	686
25.3.2 Voltage detection level register (LVIS)	687
25.4 Operation of voltage detection circuit	690
25.4.1 When used as reset mode.....	690
25.4.2 When used as interrupt mode	692
25.4.3 When used as interrupt & reset mode	694
25.5 Cautions for voltage detection circuit.....	700
Chapter 26 Safety Function	702

26.1 Function of safety function	702
26.2 Registers used by safety function	703
26.3 Operation of safety function	703
26.3.1 Flash CRC operation function (high-speed CRC)	703
26.3.2 CRC operation function (general CRC)	707
26.3.3 RAM parity error detection function	710
26.3.4 SFR protection function	712
26.3.5 Frequency detection function	713
26.3.6 A/D test function	714
26.3.7 Input/output pin digital output signal level detection function	716
26.3.8 Product unique identification register	717
Chapter 27 Temperature Sensor	718
27.1 Function of temperature sensor	718
27.2 Registers of temperature sensor	718
27.2.1 Temperature sensor calibration data register TSN25	718
27.2.2 Temperature sensor calibration data register TSN85	718
27.3 Temperature sensor usage instructions	719
27.3.1 Principle of temperature sensor	719
27.3.2 How to use temperature sensor	720
Chapter 28 Option Byte	721
28.1 Function of option byte	721
28.1.1 User option bytes (000C0H~000C2H/010C0H~010C2H)	721
28.1.2 Flash data protection option bytes (000C3H/010C3H, 500004H~500005H)	722
28.2 Format of user option byte	723
28.3 Format of flash data protection option bytes	729
Chapter 29 FLASH Control	730
29.1 Description of FLASH control function	730
29.2 Structure of FLASH memory	730
29.3 Registers for controlling FLASH	731
29.3.1 Flash write protection register (FLPROT)	731
29.3.2 FLASH operation control registers (FLOPMD1, FLOPMD2)	732
29.3.3 Flash erase control register (FLERMD)	732
29.3.4 Flash status register (FLSTS)	733
29.3.5 Flash chip erase time control register (FLCERCNT)	733
29.3.6 Flash sector erase time control register (FLSERCNT)	734
29.3.7 Flash write time control register (FLPROCNT)	735
29.4 How to operate FLASH	736
29.4.1 Sector erase	736

29.4.2	Chip erase.....	737
29.4.3	Word program	737
29.5	Flash read	738
29.6	Cautions for FLASH operation	738
Appendix Revision History		739

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.

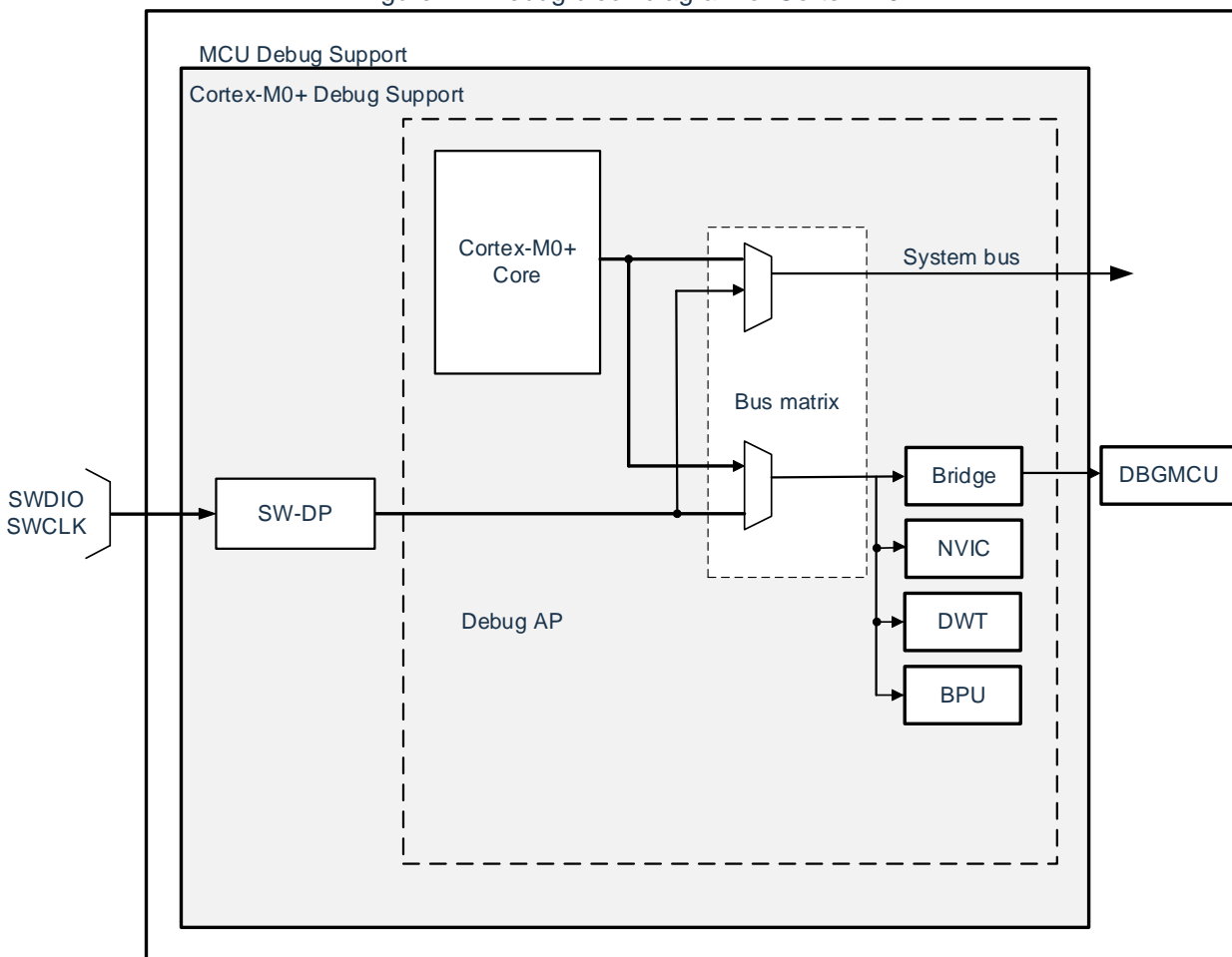
1.2 Cortex-M0+ core features

- ARM Cortex-M0+ processor is a 32-bit RISC core with a 2-stage pipeline that supports privileged mode only.
- 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 in execution

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	P137
SWDIO	Serial data input/output	P40

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

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. P40 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. P40 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 related 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 Pin Functions

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.

- Port mode register (PMxx)
- Port register (Pxx)
- Pull-up resistor selection register (PUxx)
- Pull-down resistor selection register (PDxx)
- Port output mode register (POMx)
- Port mode control register (PMCxx)
- Port set control register (PSETxx)
- Port clear control register (PCLRxx)
- Port output multiplexing configuration register (PxxCFG)
- Port input multiplexing configuration registers (TI10PCFG, TI11PCFG, TI12PCFG, TI13PCFG, INTPOPCFG, INTIP1PCFG, INTIP2PCFG, INTIP3PCFG, SDI00PCFG, SCLKI00PCFG, SS00PCFG, SDI20PCFG, SCLKI20PCFG, SDAA0PCFG, SCLA0PCFG, RXD1PCFG)
- SPI port multiplexing configuration register (SPIPCFG)

Notice The assigned registers and bits vary by product. For the registers and bits assigned to each product, please refer to Table 2-1. The initial value must be set for unassigned bits.

Table 2-1 PMxx, Pxx, PSETxx, PCLRxx, PUxx, PDxx, POMxx, PMCxx registers and their bits assigned by each product (1/2)

Ports		Bit name								48-pin (-A) Note1	48-pin	44-pin (-A) Note1	40-pin (-A) Note1	40-pin	32-pin (-A) Note1	32-pin
		PMxx register	Pxx register	PSETxx register	PCLRxx register	PUxx register	PDxx register	POMxx register	PMCxx register							
Port 0	0	PM00	P00	PSET00	PCLR00	PU00	PD00	POM00	PMC00	○	○	○	○	○	○	○
	1	PM01	P01	PSET01	PCLR01	PU01	PD01	POM01	PMC01	○	○	○	○	○	○	○
Port 1	0	PM10	P10	PSET10	PCLR10	PU10	PD10	POM10	PMC10	○	○	○	○	○	○	○
	1	PM11	P11	PSET11	PCLR11	PU11	PD11	POM11	PMC11	○	○	○	○	○	○	○
	2	PM12	P12	PSET12	PCLR12	PU12	PD12	POM12	PMC12	○	○	○	○	○	○	○
	3	PM13	P13	PSET13	PCLR13	PU13	PD13	POM13	PMC13	○	○	○	○	○	○	○
	4	PM14	P14	PSET14	PCLR14	PU14	PD14	POM14	PMC14	○	○	○	○	○	○	○
	5	PM15	P15	PSET15	PCLR15	PU15	PD15	POM15	PMC15	○	○	○	○	○	○	○
	6	PM16	P16	PSET16	PCLR16	PU16	PD16	POM16	PMC16	○	○	○	○	○	○	○
Port 2	0	PM20	P20	PSET20	PCLR20	PU20	PD20	POM20	PMC20	○	○	○	○	○	○	○
	1	PM21	P21	PSET21	PCLR21	PU21	PD21	POM21	PMC21	○	○	○	○	○	○	○
	2	PM22	P22	PSET22	PCLR22	PU22	PD22	POM22	PMC22	○	○	○	○	○	○	○
	3	PM23	P23	PSET23	PCLR23	PU23	PD23	POM23	PMC23	○	○	○	○	○	○	○
	4	PM24	P24	PSET24	PCLR24	PU24	PD24	POM24	PMC24	○	○	○	○	○	—	—
	5	PM25	P25	PSET25	PCLR25	PU25	PD25	POM25	PMC25	○	○	○	○	○	—	—
	6	PM26	P26	PSET26	PCLR26	PU26	PD26	POM26	PMC26	○	○	○	○	—	—	—
	7	PM27	P27	PSET27	PCLR27	PU27	PD27	POM27	PMC27	○	○	○	—	—	—	—

Note 1. (-A) indicates that it is limited to BAT32G135xx-A series products.

Table 2-1 PMxx, Pxx, PSETxx, PCLRxx, PUxx, PDxx, POMxx, PMCxx registers and their bits assigned by each product (2/2)

Ports		Bit name								48-pin (-A) Note1	48-pin	44-pin (-A) Note1	40-pin (-A) Note1	40-pin	32-pin (-A) Note1	32-pin
		PMxx register	Pxx register	PSETxx register	PCLRxx register	PUxx register	PDxx register	POMxx register	PMCxx register							
Port 3	0	PM30	P30	PSET30	PCLR30	PU30	PD30	POM30	PMC30	○	○	○	○	○	○	○
	1	PM31	P31	PSET31	PCLR31	PU31	PD30	POM31	PMC31	○	○	○	○	○	○	○
Port 4	0	PM40	P40	PSET40	PCLR40	PU40	—	POM40	—	○	○	○	○	○	○	○
	1	PM41	P41	PSET41	PCLR41	PU41	—	POM41	—	○	○	○	—	—	—	—
Port 5	0	PM50	P50	PSET50	PCLR50	PU50	PD50	POM50	PMC50	○	○	○	○	○	○	○
	1	PM51	P51	PSET51	PCLR51	PU51	PD51	POM51	PMC51	○	○	○	○	○	○	○
Port 6	0	PM60	P60	PSET60	PCLR60	PU60 Note2	PD60 Note2	POM60 Note2	PMC60 Note2	○	○	○	○	○	○	—
	1	PM61	P61	PSET61	PCLR61	PU61 Note2	PD61 Note2	POM61 Note2	PMC61 Note2	○	○	○	○	○	○	—
	2	PM62	P62	PSET62	PCLR62	PU62	PD62	POM62	PMC62	○	○	○	○	—	○	—
	3	PM63	P63	PSET63	PCLR63	PU63	PD63	POM63	PMC63	○	○	○	—	—	—	—
Port 7	0	PM70	P70	PSET70	PCLR70	PU70	PD70	POM70	PMC70	○	○	○	○	○	○	○
	1	PM71	P71	PSET71	PCLR71	PU71	PD71	POM71	PMC71	○	○	○	○	—	—	—
	2	PM72	P72	PSET72	PCLR72	PU72	PD72	POM72	PMC72	○	○	○	○	○	—	○
	3	PM73	P73	PSET73	PCLR73	PU73	PD73	POM73	PMC73	○	○	○	○	○	—	○
	4	PM74	P74	PSET74	PCLR74	PU74	PD74	POM74	PMC74	○	○	—	—	○	—	○
	5	PM75	P75	PSET75	PCLR75	PU75	PD75	POM75	PMC75	○	○	—	—	○	—	—
Port 12	0	PM120	P120	PSET120	PCLR120	PU120	PD120	POM120	PMC120	○	○	○	○	○	○	○
	1	PM121	P121	PSET121	PCLR121	—	—	—	—	○	○	○	○	○	○	○
	2	PM122	P122	PSET122	PCLR122	—	—	—	—	○	○	○	○	○	○	○
	3	PM123	P123	PSET123	PCLR123	—	—	—	—	○	○	○	○	○	—	—
	4	PM124	P124	PSET124	PCLR124	—	—	—	—	○	○	○	○	○	—	—
Port 13	0	PM130	P130	PSET130	PCLR130	PU130	PD130	POM130	PMC130	○	○	—	—	—	—	—
	6	PM136	P136	PSET136	PCLR136	PU136	PD136	POM136	PMC136	—	○	—	—	○	—	○
	7	PM137	P137	PSET137	PCLR137	PU137	—	POM137	—	○	○	○	○	○	○	○
Port 14	0	PM140	P140	PSET140	PCLR140	PU140	PD140	POM140	PMC140	○	○	—	—	○	—	—
	6	PM146	P146	PSET146	PCLR146	PU146	PD146	POM146	PMC146	○	○	○	—	—	—	—
	7	PM147	P147	PSET147	PCLR147	PU147	PD147	POM147	PMC147	○	○	○	○	○	○	○

Note 1. (-A) indicates that it is limited to BAT32G135xx-A series products.

2. It means that it is limited to BAT32G135xx-S series products, and for products other than BAT32G135xx-S series products, ports P60 and P61 are dedicated N-channel open drain output ports, which do not need to be configured with POM registers, and do not have their own pull-up and pull-down functions, and must be connected with external pull-up resistors when using them. They can only be used as digital pins.

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 “2.5 Register Settings for Multiplexing Function” 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 figure below.

Figure 2-1 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	1	PM01	PM00	0x020	FFH	R/W
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10	0x021	FFH	R/W
PM2	PM27	PM26	PM25	PM24	PM23	PM22	PM21	PM20	0x022	FFH	R/W
PM3	1	1	1	1	1	1	PM31	PM30	0x023	FFH	R/W
PM4	1	1	1	1	1	1	PM41	PM40	0x024	FFH	R/W
PM5	1	1	1	1	1	1	PM51	PM50	0x025	FFH	R/W
PM6	1	1	1	1	PM63	PM62	PM61	PM60	0x026	FFH	R/W
PM7	1	1	PM75	PM74	PM73	PM72	PM71	PM70	0x027	FFH	R/W
PM12	1	1	1	PM124	PM123	PM122	PM121	PM120	0x02C	FFH	R/W
PM13	PM137	PM136	1	1	1	1	1	PM130	0x02D	FEH	R/W
PM14	PM147	PM146	1	1	1	1	1	PM140	0x02E	FFH	R/W

PMmn	Selection of input/output mode for Pmn pin (m=0~7, 12~14, n=0~7)
0	Output mode (used as output port (output buffer ON))
1	Input mode (used as input port (output buffer OFF))

Notice The initial value must be set for unassigned bits.

2.3.2 Port register (Pxx)

This is a register that sets the value of the port's output latch in bits. Reading this register in input mode yields the pin level, while reading it in output mode yields the value of the port's output latch. After a reset signal is generated, the value of the register 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 figure below.

Figure 2-2 Format of port register

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
P0	0	0	0	0	0	0	P01	P00	0x000	00H (output latch)	R/W
P1	P17	P16	P15	P14	P13	P12	P11	P10	0x001	00H (output latch)	R/W
P2	P27	P26	P25	P24	P23	P22	P21	P20	0x002	00H (output latch)	R/W
P3	0	0	0	0	0	0	P31	P30	0x003	00H (output latch)	R/W
P4	0	0	0	0	0	0	P41	P40	0x004	00H (output latch)	R/W
P5	0	0	0	0	0	0	P51	P50	0x005	00H (output latch)	R/W
P6	0	0	0	0	P63	P62	P61 ^{Note2}	P60 ^{Note2}	0x006	00H (output latch)	R/W
P7	0	0	P75	P74	P73	P72	P71	P70	0x007	00H (output latch)	R/W
P12	0	0	0	P124	P123	P122	P121	P120	0x00C	00H (output latch)	R/W
P13	P137	P136	0	0	0	0	0	P130	0x00D	00H (output latch)	R/W
P14	P147	P146	0	0	0	0	0	P140	0x00E	00H (output latch)	R/W

Pmn	m=0~7, 12~14, n=0~7	
	Control of output data (output mode)	Reading of input data (input mode)
0	Outputs "0".	Inputs low level.
1	Outputs "1".	Inputs high level.

Notice 1. The initial value must be set for unassigned bits.

2. It indicates that it is limited to BAT32G135xx-S series products, and for products other than BAT32G135xx-S series products, ports P60 and P61 are dedicated N-channel open drain output ports, and only "0" and "Hiz" can be output.

2.3.3 Port set control register (PSETxx)

This is a register that resets the port output latch in bit units. The value of the register 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 figure below.

Figure 2-3 Format of port set control register

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PSET0	0	0	0	0	0	0	PSET01	PSET00	0x010	00H	W
PSET1	PSET17 PSET16 PSET15 PSET14 PSET13 PSET12 PSET11 PSET10								0x011	00H	W
PSET2	PSET27 PSET26 PSET25 PSET24 PSET23 PSET22 PSET21 PSET20								0x012	00H	W
PSET3	0	0	0	0	0	0	PSET31	PSET30	0x013	00H	W
PSET4	0	0	0	0	0	0	PSET41	PSET40	0x014	00H	W
PSET5	0	0	0	0	0	0	PSET51	PSET50	0x015	00H	W
PSET6	0	0	0	0	PSET63	PSET62	PSET61	PSET60	0x016	00H	W
PSET7	0	0	PSET75	PSET74	PSET73	PSET72	PSET71	PSET70	0x017	00H	W
PSET12	0	0	0	PSET124	PSET123	PSET122	PSET121	PSET120	0x01C	00H	W
PSET13	PSET137	PSET136	0	0	0	0	0	PSET130	0x01D	00H	W
PSET14	PSET147	PSET146	0	0	0	0	0	PSET140	0x01E	00H	W

PSETmn	Setting control of Pmn pin (m=0~7, 12~14, n=0~7)
0	No action
1	Set the corresponding Pmn to 1

Notice 1. The initial value must be set for unassigned bits.

2.3.4 Port clear control register (PCLRxx)

This is a 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 figure below.

Figure 2-4 Format of port clear control register

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PCLR0	0	0	0	0	0	0	PCLR01	PCLR00	0x070	00H	W
PCLR1	PCLR17	PCLR16	PCLR15	PCLR14	PCLR13	PCLR12	PCLR11	PCLR10	0x071	00H	W
PCLR2	PCLR27	PCLR26	PCLR25	PCLR24	PCLR23	PCLR22	PCLR21	PCLR20	0x072	00H	W
PCLR3	0	0	0	0	0	0	PCLR31	PCLR30	0x073	00H	W
PCLR4	0	0	0	0	0	0	PCLR41	PCLR40	0x074	00H	W
PCLR5	0	0	0	0	0	0	PCLR51	PCLR50	0x075	00H	W
PCLR6	0	0	0	0	PCLR63	PCLR62	PCLR61	PCLR60	0x076	00H	W
PCLR7	0	0	PCLR75	PCLR74	PCLR73	PCLR72	PCLR71	PCLR70	0x077	00H	W
PCLR12	0	0	0	PCLR124	PCLR123	PCLR122	PCLR121	PCLR120	0x07C	00H	W
PCLR13	PCLR137	PCLR136	0	0	0	0	0	PCLR130	0x07D	00H	W
PCLR14	PCLR147	PCLR146	0	0	0	0	0	PCLR140	0x07E	00H	W

PCLRmn	Clear control of Pmn pin (m=0~7, 12~14, n=0~7)
0	No action
1	Set the corresponding Pmn to 0

Notice: The initial value must be set for unassigned bits.

2.3.5 Pull-up resistor selection register (PUxx)

This is the internal pull-up resistor selection register. The internal pull-up resistor can only be used in bit units for the pins that are specified to use the internal pull-up resistor by the pull-up resistor selection register and the POM_{mn} bit is "0" and set to input mode (PM_{mn}=1). For the bit set to output mode, the internal pull-up resistor is not connected regardless of the setting of the pull-up resistor selection register. The same applies when the output pin is used as a multiplexed function or is set as an analog function.

After generating a reset signal, the pull-up function of the four ports P10, P26, P40, P137 will be turned on by default (the reset value of PU10, PU26, PU40, PU137 is "1"), and the pull-up function of the other ports will not be turned on by default.

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

Figure 2-5 Format of pull-up resistor selection register

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PU0	0	0	0	0	0	0	PU01	PU00	0x030	00H	R/W
PU1	PU17	PU16	PU15	PU14	PU13	PU12	PU11	PU10	0x031	01H	R/W
PU2	PU27	PU26	PU25	PU24	PU23	PU22	PU21	PU20	0x032	40H	R/W
PU3	0	0	0	0	0	0	PU31	PU30	0x033	00H	R/W
PU4	0	0	0	0	0	0	PU41	PU40	0x034	01H	R/W
PU5	0	0	0	0	0	0	PU51	PU50	0x035	00H	R/W
PU6	0	0	0	0	PU63	PU62	PU61 ^{Note2}	PU60 ^{Note2}	0x036	00H	R/W
PU7	0	0	PU75	PU74	PU73	PU72	PU71	PU70	0x037	00H	R/W
PU12	0	0	0	0	0	0	0	PU120	0x03C	00H	R/W
PU13	PU137	PU136	0	0	0	0	0	PU130	0x03D	80H	R/W
PU14	PU147	PU146	0	0	0	0	0	PU140	0x03E	00H	R/W

PU _{mn}	Selection of internal pull-up resistor for P _m n pin (m=0~7, 12~14, n=0~7)
0	No internal pull-up resistor is connected.
1	Connects to an internal pull-up resistor.

Notice The initial value must be set for unassigned bits.

Note2. It indicates that it is limited to BAT32G135xx-S series products, and for products other than BAT32G135xx-S series products, ports P60 and P61 do not have pull-up and pull-down functions, and external pull-up resistors must be connected for use.

2.3.6 Pull-down resistor selection register (PDxx)

This is the internal pull-down resistor selection register. The internal pull-down resistor can only be used in bit units for the pin that is specified to use the internal pull-down resistor by the pull-down resistor selection register, and for the bit that is set to input mode (PMmn=1) with the POMmn bit set to "0". For the bit set to output mode, the internal pull-down resistor is not connected regardless of the setting of the pull-down resistor selection register. The same applies when the bit is used as an output pin of the multiplexing function or when it is set to the analog function.

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 figure below.

Figure 2-6 Format of pull-down resistor selection register

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PD0	0	0	0	0	0	0	PD01	PD00	0x040	00H	R/W
PD1	PD17	PD16	PD15	PD14	PD13	PD12	PD11	PD10	0x041	00H	R/W
PD2	PD27	PD26	PD25	PD24	PD23	PD22	PD21	PD20	0x042	00H	R/W
PD3	0	0	0	0	0	0	PD31	PD30	0x043	00H	R/W
PD5	0	0	0	0	0	0	PD51	PD50	0x045	00H	R/W
PD6	0	0	0	0	PD63	PD62	PD61 ^{Note2}	PD60 ^{Note2}	0x046	00H	R/W
PD7	0	0	PD75	PD74	PD73	PD72	PD71	PD70	0x047	00H	R/W
PD12	0	0	0	0	0	0	0	PD120	0x04C	00H	R/W
PD13	0	PD136	0	0	0	0	0	PD130	0x04D	00H	R/W
PD14	PD147	PD146	0	0	0	0	0	PD140	0x04E	00H	R/W

PDmn	Selection of internal pull-down resistor for Pmn pin(m=0~3, 5~7, 12~14, n=0~7)
0	No internal pull-down resistor is connected.
1	Connects an internal pull-down resistor.

Notice The initial value must be set for unassigned bits.

Note2. It indicates that it is limited to the BAT32G135xx-S series products. For products other than the BAT32G135xx-S series, ports P60 and P61 do not have pull-up and pull-down functions and must be used with external pull-up resistors.

2.3.7 Port output mode register (POMxx)

This is a register that sets the output mode in bits. The N-channel open-drain output mode can be selected for the SD_{Axx} pin for serial communication with external devices at different potentials and simplified I²C communication with external devices at the same potential.

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 figure below.

Notice For the bit that sets the N-channel open-drain output mode (POM_{mn}=1), the internal pull-up resistor is unconnected.

Figure 2-7 Format of port output mode register

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
POM0	0	0	0	0	0	0	POM01	POM00	0x050	00H	R/W
POM1	POM17	POM16	POM15	POM14	POM13	POM12	POM11	POM10	0x051	00H	R/W
POM2	POM27	POM26	POM25	POM24	POM23	POM22	POM21	POM20	0x052	00H	R/W
POM3	0	0	0	0	0	0	POM31	POM30	0x053	00H	R/W
POM4	0	0	0	0	0	0	POM41	POM40	0x054	00H	R/W
POM5	0	0	0	0	0	0	POM51	POM50	0x055	00H	R/W
POM6	0	0	0	0	POM63	POM62	POM61 ^{Note2}	POM60 ^{Note2}	0x056	00H	R/W
POM7	0	0	POM75	POM74	POM73	POM72	POM71	POM70	0x057	00H	R/W
POM12	0	0	0	0	0	0	0	POM120	0x05C	00H	R/W
POM13	POM137	POM136	0	0	0	0	0	POM130	0x05D	00H	R/W
POM14	POM147	POM146	0	0	0	0	0	POM140	0x05E	00H	R/W

POM _{mn}	Selection of output mode for P _{mn} pin (m=0~3, 5~7, 12~14, n=0~7)
0	Typical output mode
1	N-channel open-drain output mode

Note 1. The initial value must be set for unassigned bits.

2. It indicates that it is limited to the BAT32G135xx-S series products, and for products other than the BAT32G135xx-S series products, ports P60 and P61 are dedicated N-channel open drain output ports, and there is no need to configure the POM register.

3. Ports P121~P124 do not have N-channel open drain output function.

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 generating a reset signal, P10, P26, P130 are used as digital channels by default (PMC10, PMC26, PMC130 reset value is "0"), and other ports are used as analog channels by default. P40, P41, P60, P61, P122~P124, P137 are digital only and cannot be used as analog channels.

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

Figure 2-8 Format of port mode control register

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PMC0	1	1	1	1	1	1	PMC01	PMC00	0x060	FFH	R/W
PMC1	PMC17	PMC16	PMC15	PMC14	PMC13	PMC12	PMC11	PMC10	0x061	FEH	R/W
PMC2	PMC27	PMC26	PMC25	PMC24	PMC23	PMC22	PMC21	PMC20	0x062	DFH	R/W
PMC3	1	1	1	1	1	1	PMC31	PMC30	0x063	FFH	R/W
PMC5	1	1	1	1	1	1	PMC51	PMC50	0x065	FFH	R/W
PMC6	1	1	1	1	PMC63	PMC62	PMC61 ^{Note2}	PMC60 ^{Note2}	0x066	FFH	R/W
PMC7	1	1	PMC75	PMC74	PMC73	PMC72	PMC71	PMC70	0x067	FFH	R/W
PMC12	1	1	1	1	1	1	1	PMC120	0x06C	FFH	R/W
PMC13	0	PMC136	1	1	1	1	1	PMC130	0x06D	7EH	R/W
PMC14	PMC147	PMC146	1	1	1	1	1	PMC140	0x06E	FFH	R/W

PMCmn	Selection of digital inputs/outputs or analog inputs for Pmn pins (m=0~3, 5~7, 12~14, n=0~7)
0	Digital inputs/outputs (multiplexing functions other than analog inputs)
1	Analog inputs

Note 1. The initial value must be set for unassigned bits.

2. It indicates that it is limited to the BAT32G135xx-S series, and for products other than the BAT32G135xx-S series, ports P60 and P61 can only be used as digital pins.
3. Ports P40, P41, P121~P124, P137 are not supported as analog channels.
4. P10, P26, P130 are used as digital channels by default after reset.
5. Except for the above ports, the other ports are used as analog channels by default after reset.

2.3.9 Port multiplexing 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.

Figure 2-9 List of port output multiplexing configuration registers

Register name	Offset address	R/W	Reset value
P00CFG	0x000	R/W	00H
P01CFG	0x001	R/W	00H
P10CFG	0x008	R/W	00H
P11CFG	0x009	R/W	00H
P12CFG	0x00a	R/W	00H
P13CFG	0x00b	R/W	00H
P14CFG	0x00c	R/W	00H
P15CFG	0x00d	R/W	00H
P16CFG	0x00e	R/W	00H
P17CFG	0x00f	R/W	00H
P20CFG	0x010	R/W	00H
P21CFG	0x011	R/W	00H
P22CFG	0x012	R/W	00H
P23CFG	0x013	R/W	00H
P24CFG	0x014	R/W	00H
P25CFG	0x015	R/W	00H
P26CFG	0x016	R/W	00H
P27CFG	0x017	R/W	00H
P30CFG	0x018	R/W	00H
P31CFG	0x019	R/W	00H
P40CFG	0x020	R/W	00H
P41CFG	0x021	R/W	00H
P50CFG	0x028	R/W	00H
P51CFG	0x029	R/W	00H
P60CFG	0x030	R/W	00H
P61CFG	0x031	R/W	00H
P62CFG	0x032	R/W	00H
P63CFG	0x033	R/W	00H
P70CFG	0x038	R/W	00H
P71CFG	0x039	R/W	00H
P72CFG	0x03a	R/W	00H
P73CFG	0x03b	R/W	00H
P74CFG	0x03c	R/W	00H
P75CFG	0x03d	R/W	00H
P120CFG	0x040	R/W	00H
P121CFG	0x041	R/W	00H
P122CFG	0x042	R/W	00H
P123CFG	0x043	R/W	00H
P124CFG	0x044	R/W	00H
P130CFG	0x048	R/W	00H
P136CFG	0x04e	R/W	00H
P137CFG	0x04f	R/W	00H
P140CFG	0x050	R/W	00H
P146CFG	0x056	R/W	00H
P147CFG	0x057	R/W	00H

Figure 2-10 Format of port multiplexing configuration register

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PxxCFG	0	0	0	0	pxxcfg[3:0]				See figure above	00H	R/W

By configuring the PxxCFG register, it is possible to map 15 concurrent output functions (TO10, TO11, TO12, TO13, SDO00, TxD0, SDO20, TxD2, IrTXD, CLKBUZ0, SCLKO00, SCL00, SCLKO20, SCL20, TxD1) to any port, other than these 15 concurrent outputs can only be mapped to fixed ports.

Register name	Register settings	Pxx port function
pxxcfg[3:0]	4'h00	Default concurrent feature /GPIO
	4'h01	TO10
	4'h02	TO11
	4'h03	TO12
	4'h04	TO13
	4'h05	SDO00/TxD0
	4'h06	SDO20/TxD2/IrTXD
	4'h07	CLKBUZ0
	4'h08	SCLKO00/SCL00
	4'h09	SCLKO20/SCL20
	4'h0a	TxD1
	others	Disable settings

Table 2-2 Configuration method for concurrent output function

Function name		Input/output	PxxCFP	PMCxx	PMxx	POMxx	Pxx	Remark
Analog channel		Input/output	4'h0	1	x	×	×	All analog functions are directed to fixed ports only and are not configurable, Refer to the data sheets for each product family
Digital GPIO		output	4'h0	0	0	0	0/1	
		N-channel open-drain output		0	0	1	0/1	
Concurrent output that can be mapped to any port	TO10	Output	4'h1	0	0	0	0	Can be mapped to any port
	TO11	Output	4'h2	0	0	0	0	Can be mapped to any port
	TO12	Output	4'h3	0	0	0	0	Can be mapped to any port
	TO13	Output	4'h4	0	0	0	0	Can be mapped to any port
	SDO00/TxD0	Output	4'h5	0	0	0	1	Can be mapped to any port
	SDO20/TxD2/IrTXD	Output	4'h6	0	0	0	1	Can be mapped to any port
	CLKBUZ0	Output	4'h7	0	0	0	0	Can be mapped to any port
	SCLKO00/SCL00	Output	4'h8	0	0	0	1	Can be mapped to any port
	SCLKO20/SCL20	Output	4'h9	0	0	0	1	Can be mapped to any port
	TxD1	Output	4'ha	0	0	0	1	Can be mapped to any port
Concurrent output mapped to fixed ports	TO00	Output	P01CFG=4'h0	0	0	0	0	P01 is used by default and cannot be mapped to other ports
	TO01	Output	P16CFG=4'h0	0	0	0	0	P16 is used by default and cannot be mapped to other ports
	TO02	Output	P17CFG=4'h0	0	0	0	0	P17 is used by default and cannot be mapped to other ports
	TO03	Output	P31CFG=4'h0	0	0	0	0	P31 is used by default and cannot be mapped to other ports
	SCLKO01/SCL01	Output	P75CFG=4'h0	0	0	0	1	P75 is used by default and cannot be mapped to other ports
	SDO01	Output	P73CFG=4'h0	0	0	0	1	P73 is used by default and cannot be mapped to other ports
	SDA01	Bi-directional	P74CFG=4'h0	0	0	1	1	P74 is used by default and cannot be mapped to other ports
	SCLKO11/SCL11	Output	P10CFG=4'h0	0	0	0	1	P10 is used by default and cannot be mapped to other ports
	SDA11	Bi-directional	P11CFG=4'h0	0	0	1	1	P11 is used by default and cannot be mapped to other ports
	SDO11	Output	P12CFG=4'h0	0	0	0	1	P12 is used by default and cannot be mapped to other ports
	SDA20	Bi-directional	P14CFG=4'h0	0	0	1	1	P14 is used by default and cannot be mapped to other ports
	SCLKO21/SCL21	Output	P70CFG=4'h0	0	0	0	1	P70 is used by default and cannot be mapped to other ports
	SDA21	Bi-directional	P71CFG=4'h0	0	0	1	1	P71 is used by default and cannot be mapped to other ports
	SDO21	Output	P72CFG=4'h0	0	0	0	1	P72 is used by default and cannot be mapped to other ports
	CLKBUZ1	Output	P15CFG=4'h0	0	0	0	0	P15 is used by default and cannot be mapped to other ports
	RTC1HZ	Output	P30CFG=4'h0	0	0	0	0	P30 is used by default and cannot be mapped to other ports
	VCOUT0	Output	P120CFG=4'h0	0	0	0	0	P120 is used by default and cannot be mapped to other ports
VCOUT1	Output	P50CFG=4'h0	0	0	0	0	P50 is used by default and cannot be mapped to other ports	

Note: When using the port's dual output function, you need to set the port output latch Pxx, the configuration method is detailed in the above table, for reasons please refer to 2.5.1 Basic idea when using multiplexing output function.

Configuration instructions:

- When using the port's concurrent output function, the port must be configured in digital mode (PMCxx=0).
- When using the port's concurrent output function, the port must be configured in output mode (push-pull or open-drain) (PMxx=0).
- When using the GPIO function or multiplexing function of the P121, P122 ports, verify that the X1 oscillation mode and external clock input mode are not turned on. Refer to “Section 4.3.1 of Chapter 4 Clock Generation Circuits”
- When using the GPIO function or multiplexing function of the P123, P124 port, verify that the XT1 oscillation mode and external clock input mode are not turned on. Refer to “Section 4.3.1 of Chapter 4 Clock Generation Circuits”
- Ports P60 and P61 are dedicated N-channel open-drain output ports and do not support push-pull concurrent outputs.
- When using the concurrent output function of the port, it is necessary to set the port output latch Pxx, and the configuration method is detailed in Table2-2 Configuration method for concurrent output function.
- The data port (SDAxx) of the Easy IIC, the clock port of the IICA (SCLA0) and the data port of the IICA (SDAA0) support bidirectional communication, and only the SDI00PCFG, SCLA0PCFG, SDAA0PCFG registers need to be configured when setting the mapped port, and there is no need to configure the PxxCFG register.

2.3.10 Port input multiplexing configuration registers (TI10PCFG, TI11PCFG, TI12PCFG, TI13PCFG, INTP0PCFG, INTP1PCFG, INTP2PCFG, INTP3PCFG, SDI00PCFG, SCLKI00PCFG, SS00PCFG, SDI20PCFG, SCLKI20PCFG, SDAA0PCFG, SCLA0PCFG, RXD1PCFG)

The port input multiplexing configuration registers enable the mapping of input functions from peripheral modules to each port. The reset value of the port input multiplexing function configuration register is “00H”. 20 concurrent input functions (TI10, TI11, TI12, TI13, INTP0, INTP1, INTP2, INTP3, SDI00, RXD0, SDA00, SCLKI00, SS00, SDI20, RXD2, IrRXD, SCLKI20, SDAA0, SCLA0, RXD1) can be mapped to any port. Other than these 20 types of concurrent inputs can only be input from a fixed port.

Register address = base address + offset address; the base address of the register is 0x40040800, and the offset address is shown in the following figure.

Figure 2-11 List of port input multiplexing configuration registers

Register name	Offset address	R/W	Reset value	Function
TI10PCFG	0x060	R/W	00H	Set the mapped port for TI10
TI11PCFG	0x061	R/W	00H	Set the mapped port for TI11
TI12PCFG	0x062	R/W	00H	Set the mapped port for TI12
TI13PCFG	0x063	R/W	00H	Set the mapped port for TI13
INTP0PCFG	0x064	R/W	00H	Set the mapped port for INTP0
INTP1PCFG	0x065	R/W	00H	Set the mapped port for INTP1
INTP2PCFG	0x066	R/W	00H	Set the mapped port for INTP2
INTP3PCFG	0x067	R/W	00H	Set the mapped port for INTP3
SDI00PCFG	0x068	R/W	00H	Set the mapped ports for SDI00/RXD0/SDA00
SCLKI00PCFG	0x069	R/W	00H	Set the mapped port for SCLKI00
SSI00PCFG	0x06a	R/W	00H	Set the mapped port for SS00
SDI20PCFG	0x06b	R/W	00H	Set the mapped ports for SDI20/RXD2/IrRXD
SCLKI20PCFG	0x06c	R/W	00H	Set the mapped port for SCLKI20
SDAA0PCFG	0x06d	R/W	00H	Set the mapped port for SDAA0
SCLA0PCFG	0x06e	R/W	00H	Set the mapped port for SCLA0
RXD1PCFG	0x06f	R/W	00H	Set the mapped port for RXD1

Figure 2-12 Format of port input multiplexing configuration register

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W	
xxPCFG	0	0	xxpcfg[5:0]							See figure above	00H	R/W

The xxPCFG register is used to map redirectable concurrent inputs to any port.

Register name	Register settings	Function
TI10PCFG/ TI11PCFG/ TI12PCFG/ TI13PCFG/ INTP0PCFG/ INTP1PCFG/ INTP2PCFG/ INTP3PCFG/ SDI00PCFG/ SCLKI00PCFG/ SS00PCFG/ SDI20PCFG/ SCLKI20PCFG/ SDAA0PCFG/ SCLA0PCFG/ RXD1PCFG	6'h00	Concurrent input does not map to any port
	6'h01	Maps to P00
	6'h02	Map to P01
	6'h03	Map to P10
	6'h04	Maps to P11
	6'h05	Map to P12
	6'h06	Map to P13
	6'h07	Maps to P14
	6'h08	Maps to P15
	6'h09	Maps to P16
	6'h0a	Maps to P17
	6'h0b	Maps to P20
	6'h0c	Map to P21
	6'h0d	Maps to P22
	6'h0e	Maps to P23
	6'h0f	Maps to P24
	6'h10	Maps to P25
	6'h11	Map to P26
	6'h12	Map to P27
	6'h13	Maps to P30
	6'h14	Maps to P31
	6'h15	Map to P40
	6'h16	Maps to P41
	6'h17	Maps to P50
	6'h18	Map to P51
	6'h19	Map to P60
	6'h1a	Map to P61
	6'h1b	Maps to P62
	6'h1c	Maps to P63
	6'h1d	Maps to P70
	6'h1e	Maps to P71
	6'h1f	Map to P72
6'h20	Maps to P73	
6'h21	Maps to P74	
6'h22	Maps to P75	
6'h23	Maps to P120	
6'h24	Maps to P121	
6'h25	Maps to P122	
6'h26	Maps to P123	
6'h27	Maps to P124	
6'h28	Maps to P130	
6'h29	Maps to P136	
6'h2a	Maps to P137	
6'h2b	Maps to P140	
6'h2c	Maps to P146	
6'h2d	Maps to P147	

Table 2-3 Configuration method of concurrent input function

Function name	Input/output	xxxPCFP[5:0]	PMCxx	PMx x	POMx x	Pxx	Remark	
Simulation function	Input/output	x	1	x	×	×	All analog functions are directed to fixed ports only and are not configurable, Refer to the data sheets for each product family	
GPIO	Input	x	0	1	×	×		
Concurrent input that can be mapped to any port	TI10	Input	Configure TI10PCFG	0	1	×	×	Can be mapped to any port
	TI11	Input	Configure TI11PCFG	0	1	×	×	Can be mapped to any port
	TI12	Input	Configure TI12PCFG	0	1	×	×	Can be mapped to any port
	TI13	Input	Configure TI13PCFG	0	1	×	×	Can be mapped to any port
	INTP0	Input	Configure INTP0PCFG	0	1	×	×	By default, P136 is used and can be mapped to any port
	INTP1	Input	Configure INTP1PCFG	0	1	×	×	P50 is used by default and can be mapped to any port
	INTP2	Input	Configure INTP2PCFG	0	1	×	×	P51 is used by default and can be mapped to any port
	INTP3	Input	Configure INTP3PCFG	0	1	×	×	P30 is used by default and can be mapped to any port
	SCLKI00	Input	Configure SCLKI00PCFG	0	1	×	×	Can be mapped to any port
	SDI00/RxD0	Input	Configure SDI00PCFG	0	1	×	×	Can be mapped to any port
	SDA00	Bi-directional		0	0	1	1	Can be mapped to any port, except P121 to P124
	SS00	Input	Configure SS00PCFG	0	1	×	×	Can be mapped to any port
	RxD1	Input	Configure RxD1PCFG	0	1	×	×	Can be mapped to any port
	SCLKI20	Input	Configure SCLKI20PCFG	0	1	×	×	Can be mapped to any port
	SDI20/RxD2/IrRXD	Input	Configure SDI20PCFG	0	1	×	×	Can be mapped to any port
	SCLA0	Bi-directional	Configure SCLA0PCFG	0	0	1*	0	It can be mapped to any port, except for P121~P124POMxx automatic setting 1, no software configuration is required
SDAA0	Bi-directional	Configure SDAA0PCFG	0	0	1*	0	It can be mapped to any port, except for P121~P124POMxx automatic setting 1, no software configuration is required	
Concurrent input mapped to fixed ports	TI00	Input	x	0	1	×	×	Fixed use P00
	TI01	Input	x	0	1	×	×	Fixed use P16
	TI02	Input	x	0	1	×	×	Fixed use P17
	TI03	Input	x	0	1	×	×	Fixed use P31
	SCLKI01	Input	x	0	1	×	×	Fixed use P75
	SDI01	Input	x	0	1	×	×	Fixed use P74
	SDA01	Bi-directional	x	0	0	1	1	Fixed use P74
	SCLKI11	Input	x	0	1	×	×	Fixed use P10
	SDI11	Input	x	0	1	×	×	Fixed use P11
	SDA11	Bi-directional	x	0	0	1	1	Fixed use P11
	SDA20	Bi-directional	x	0	0	1	1	Fixed use P14
	SCLKI21	Input	x	0	1	×	×	Fixed use P70
	SDI21	Input	x	0	1	×	×	Fixed use P71
	SDA21	Bi-directional	x	0	0	1	1	Fixed use P71
	KR0	Input	x	0	1	×	×	Fixed use P70
	KR1	Input	x	0	1	×	×	Fixed use P71
	KR2	Input	x	0	1	×	×	Fixed use P72
KR3	Input	x	0	1	×	×	Fixed use P73	
KR4	Input	x	0	1	×	×	Fixed use P74	
KR5	Input	x	0	1	×	×	Fixed use P75	

Note: The data port (SDAxx) of the simplified IIC, the clock port of the IICA (SCLA0) and the data port of the IICA (SDAA0) are bi-directional communication, and only SDI00PCFG, SCLA0PCFG, SDAA0PCFG, and PxxCFG need to be configured when used. And the port output latch Pxx needs to be set to the appropriate value, the configuration method is detailed in the above table, for reasons please refer to 2.5.1 Basic idea when using multiplexing output function.

Configuration Instructions:

- When using the port's concurrent input function, the port must be configured in digital mode (PMCxx=0).
- When using the port's concurrent input function, the port must be configured to input mode (PMxx=1).
- For bidirectional multiplexing, the port must be configured in output mode (push-pull or open-drain) (PMxx=0). At this point, the input driver is configured for floating input mode.
- When using the GPIO function or multiplexing function of the P121, P122 port, verify that the X1 oscillation mode and external clock input mode are not turned on. Refer to "Section 4.3.1 of Chapter 4 Clock Generation Circuits"
- When using the GPIO function or multiplexing function of the P123, P124 port, verify that the XT1 oscillation mode and external clock input mode are not turned on. Refer to "Section 4.3.1 of Chapter 4 Clock Generation Circuits"
- The data port (SDAxx) of the simplified IIC, the clock port of the IICA (SCLA0) and the data port of the IICA (SDAA0) support bidirectional communication, and only the SDI00PCFG, SCLA0PCFG, SDAA0PCFG registers need to be configured when setting the mapped port, and there is no need to configure the PxxCFG register.

2.3.11 SPI port multiplexing configuration register (SPIPCFG)

The SPI port multiplexing configuration register (SPIPCFG) enables the SPI communication function to be mapped to three different sets of port combinations. The reset value of the SPI port multiplexing function configuration register is “00H”, and the SPI communication function is not mapped to any port.

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

Figure 2-13 Format of port input multiplexing configuration register

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
SPIPCFG	0	0	0	0	0	0	spipcfcg[1:0]		0x07E	00H	R/W

Register name	Register settings	Mapping relationships			
		NSS	SCK	MISO	MOSI
SPIPCFG[1:0]	2'b00	Does not map to any port			
	2'b01	P50	P51	P17	P16
	2'b10	P63	P31	P75	P74
	1'b11	P25	P24	P23	P22

Table 2-4 SPI communication port configuration method

SPI port combination	Port name	Function name	Input/output	SPIPCFG	PxxCFP	xxPCFG	PMCxx	PMxx	POMxx	Pxx
spi_group1	P50	SPL_NSS	Input	2' b01	x	x	0	1	x	x
			Output		x	x	0	0	0	0
	P51	SPI_SCK	Input		x	x	0	1	x	x
			Output		x	x	0	0	0	0
	P16	SPI_MOSI	Input		x	x	0	1	x	x
			Output		x	x	0	0	0	0
P17	SPI_MISO	Input	x	x	0	1	x	x		
		Output	x	x	0	0	0	0		
spi_group2	P63	SPL_NSS	Input	2' b10	x	x	0	1	x	x
			Output		x	x	0	0	0	0
	P31	SPI_SCK	Input		x	x	0	1	x	x
			Output		x	x	0	0	0	0
	P74	SPI_MOSI	Input		x	x	0	1	x	x
			Output		x	x	0	1	x	x
P75	SPI_MISO	Input	x	x	0	1	x	x		
		Output	x	x	0	0	0	0		
spi_group3	P25	SPL_NSS	Input	2' b11	x	x	0	1	x	x
			Output		x	x	0	0	0	0
	P24	SPI_SCK	Input		x	x	0	1	x	x
			Output		x	x	0	0	0	0
	P22	SPI_MOSI	Input		x	x	0	1	x	x
			Output		x	x	0	1	x	x
P23	SPI_MISO	Input	x	x	0	1	x	x		
		Output	x	x	0	0	0	0		

2.4 Handling of unused pins

The handling of each unused pin is shown in Table 2-5.

Table 2-5 Handling of each unused pin

Pin name	Input/output	Recommended connection method when not in use
P00, P01	Input/output	Input: Individually connected via resistor EV_{DD} or EV_{SS} . Output: Set to open.
P10~P17		
P20~P27		
P30, P31		
P40		Input: Connect V_{DD} individually via resistor or set it open. Output: Set to open.
P41		Input: Individually connected via resistor EV_{DD} or EV_{SS} . Output: Set to open.
P50, P51		
P60, P61		Input: Individually connected via resistor EV_{DD} or EV_{SS} . Output: Set the port's output latch to "0" and set it open, or set the port's output latch to "1" and individually Connect EV_{DD} or EV_{SS} via resistors.
P62~P63		Input: Individually connected via resistor EV_{DD} or EV_{SS} . Output: Set to open.
P70~P75		
P120		
P121~P124		Separate resistors connect V_{DD} or V_{SS} .
P130, P136		Input: Individually connected via resistor EV_{DD} or EV_{SS} . Output: Set to open.
P137		Input: Connect V_{DD} individually via resistor or set it open. Output: Set to open.
P140, P146, P147		Input: Individually connected via resistor EV_{DD} or EV_{SS} . Output: Set to open.
RESETB		Input

Remark For products that do not have EV_{DD} , EV_{SS} pins, EV_{DD} must be replaced with V_{DD} and will EV_{SS} is replaced with V_{SS} .

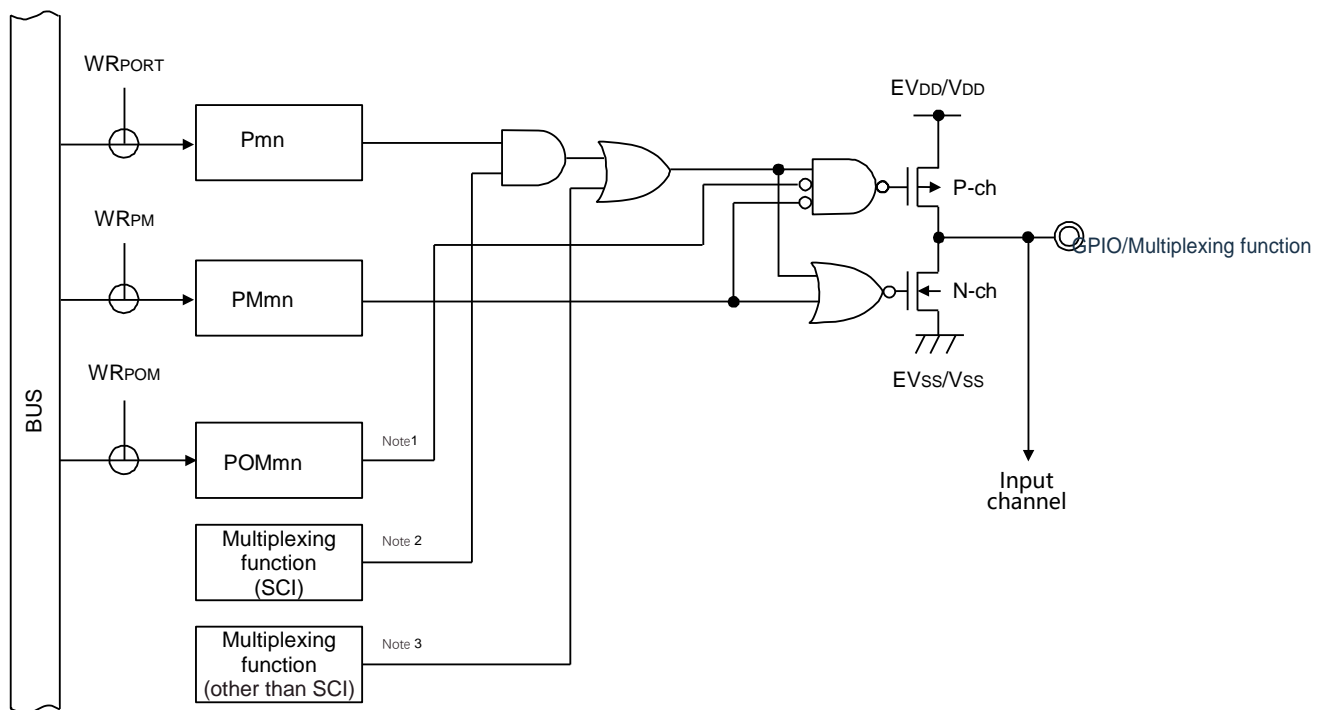
2.5 Register settings when using multiplexing function

2.5.1 Basic idea when using multiplexing output function

First, for ports with analog functions, the port mode control register (PMCxx) sets whether the port is used as an analog function or as a digital input/output.

The basic structure of the output circuit when used as a digital input/output is shown in Figure 2-14. The output of the SCI function multiplexed with the output latch of the port is input to the AND gate, the output of the AND gate is input to the OR gate, and the other inputs of the OR gate are connected to the multiplexed output of non-SCI functions (timer, RTC, clock/buzzer output, IICA, etc.). When such a port is used as a port function or a multiplexing feature, unused multiplexed features cannot affect the output of the functionality to be used. The basic idea of setting in this case is shown in Table 2-6.

Figure 2-14 Basic output structure of ports



- Note: 1. When there is no POM register, this signal is Low level (0).
 2. The signal is high level (1) when the multiplexing function is not available.
 3. The signal is low level (0) when the multiplexing function is not available.

Table 2-6 Basic principal for configuration

Output function of the pin used	Output settings for unused multiplexing		
	Port function	SCI output function	Output function other than SCI
Port output function	—	High output (1)	Low output (0)
SCI output function	High (1)	—	Low output (0)
Output function other than SCI	Low (0)	High output (1)	Low output (0) ^{Note}

Note: Since it is possible to multiplex more than one output function other than SCI on a single pin, it is necessary to set the output of the multiplexed function that is not used to a low level (0). For details, refer to Section 2.5.2 “Example of Register Setting for Used Port Functions and Multiplexing Functions”.

2.5.2 Example of register setting for used port functions and multiplexing functions

Examples of register settings (48-pin products) using the port function and multiplexing function are shown in Table 2-7~Table 2-17. In the table, “x” indicates that the register does not need to be configured, and “-” in the table indicates that there is no such register.

Table 2-7 Example of register setting when using the P00 to P01 pin functions

Pin name	Used function		PMCxx	PMxx	Pxx	POMxx	PxxCFG (Output multiplexing configuration register)	xxPCFG (Input multiplexing configuration register)	SPIPCFG	Remark
	Function name	Input/output								
P00	P00	Input	0	1	x	x	x	x	x	
		Output	0	0	0/1	0	P00CFG=4'h0	x	x	
		N-channel open drain output	0	0	0/1	1			x	
	ANI11/VCIN10	Analog channel	1	x	x	x	x	x	x	
	TI00	Input	0	1	x	x	x	x	x	
	Mappable multiplexing input	Input	0	1	x	x	x	Configure xxPCFG	x	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P00CFG	x	x	See 2.3.10
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P00CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	x	See 2.3.9	
P01	P01	Input	0	1	x	x	x	x	x	
		Output	0	0	0/1	0	P01CFG=4'h0	x	x	
		N-channel open drain output	0	0	0/1	1			x	
	ANI10/VCIN11	Analog channel	1	x	x	x	x	x	x	
	TO00	Output	0	0	0	0	P01CFG=4'h0	x	x	
	Mappable multiplexing input	Input	0	1	x	x	x	Configure xxPCFG	x	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P01CFG	x	x	See 2.3.10
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P01CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	x	See 2.3.9	

Table 2-8 Example of register setting when using P10 to P17 pin functions

Pin name	Used function		PMC xx	PMx x	Px x	POM xx	PxxCFG (Output multiplexing configuration register)	xxPCFG (Input multiplexing configuration register)	SPIPCFG	Remark
	Function name	Input/output								
P10	P10	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P10CFG=4' h0	×	×	
		N-channel open drain output	0	0	0/1	1			×	
	ANI9	Analog channel	1	×	×	×	×	×	×	
	SCLK11	Input	0	1	×	×	×	×	×	
		Output	0	0	1	0	P10CFG=4' h0	×	×	
	epwmo00	Output	0	0	0	0	P10CFG=4' h0	×	×	See 2.5.3
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
Mappable multiplexing output	Output	0	0	0/1	0/1	Configure P10CFG	×	×	See 2.3.10	
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P10CFG=4' h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9	
P11	P11	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P11CFG=4' h0	×	×	
		N-channel open drain output	0	0	0/1	1			×	
	ANI8	Analog channel	1	×	×	×	×	×	×	
	SDI11	Input	0	1	×	×	×	×	×	
	SDA11	Bi-directional	0	0	1	1	P11CFG=4' h0	×	×	
	epwmo01	Output	0	0	0	0	P11CFG=4' h0	×	×	See 2.5.3
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
Mappable multiplexing output	Output	0	0	0/1	0	Configure P11CFG	×	×	See 2.3.10	
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P11CFG=4' h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9	
P12	P12	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P12CFG=4' h0	×	×	
		N-channel open drain output	0	0	0/1	1			×	
	ANI13	Analog channel	1	×	×	×	×	×	×	
	SDO11	Output	0	0	1	0	P12CFG=4' h0	×	×	
	epwmo02	Output	0	0	0	0	P12CFG=4' h0	×	×	See 2.5.3
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P12CFG	×	×	See 2.3.10
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P12CFG=4' h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9	
P13	P13	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P13CFG=4' h0	×	×	
		N-channel open drain output	0	0	0/1	1			×	
	ANI16	Analog channel	1	×	×	×	×	×	×	
	epwmo03	Output	0	0	0	0	P13CFG=4' h0	×	×	See 2.5.3
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P13CFG	×	×	See 2.3.10
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P13CFG=4' h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9	

Pin name	Used function		PMCxx	PMxx	Pxx	POMxx	PxxCFG (Output multiplexing configuration register)	xxPCFG (Input multiplexing configuration register)	SPIPCFG	Remark	
	Function name	Input/output									
P14	P14	Input	0	1	×	×	×	×	×		
		Output	0	0	0/1	0	P14CFG=4'h0	×	×		
		N-channel open drain output	0	0	0/1	1					
	ANI17	Analog channel	1	×	×	×	×	×	×		
	SDA20	Bi-directional	0	0	1	1	P14CFG=4'h0	SDI20PCFG=6'h00	×		
	epwmo04	Output	0	0	0	0	P14CFG=4'h0	×	×	See 2.5.3	
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9	
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P14CFG	×	×	See 2.3.10	
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P14CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9		
P15	P15	Input	0	1	×	×	×	×	×		
		Output	0	0	0/1	0	P15CFG=4'h0	×	×		
		N-channel open drain output	0	0	0/1	1					
	ANI18	Analog channel	1	×	×	×	×	×	×		
	CLKBUZ1	Output	0	0	0	0	P15CFG=4'h0	×	×		
	epwmo05	Output	0	0	0	0	P15CFG=4'h0	×	×	See 2.5.3	
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9	
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P15CFG	×	×	See 2.3.10	
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P15CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9		
P16	P16	Input	0	1	×	×	×	×	×		
		Output	0	0	0/1	0	P16CFG=4'h0	×	×	≠2' b01	
		N-channel open drain output	0	0	0/1	1					
	ANI19	Analog channel	1	×	×	×	×	×	×		
	TI01	Input	0	1	×	×	×	×	×		
	TO01	Output	0	0	0	0	P16CFG=4'h0	×	×	≠2' b01	
	SPI_MOSI	Input	0	1	×	×	×	×	×	2' b01	See 2.3.11
		Output	0	0	0	0	×	×	×		
	epwmo06	Output	0	0	0	0	P16CFG=4'h0	×	×	≠2' b01	See 2.5.3
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9	
Mappable multiplexing output	Output	0	0	0/1	0	Configure P16CFG	×	×	≠2' b01	See 2.3.10	
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P16CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	≠2' b01	See 2.3.9	
P17	P17	Input	0	1	×	×	×	×	×		
		Output	0	0	0/1	0	P17CFG=4'h0	×	×	≠2' b01	
		N-channel open drain output	0	0	0/1	1					
	ANI20	Analog channel	1	×	×	×	×	×	×		
	TI02	Input	0	1	×	×	×	×	×		
	TO02	Output	0	0	0	0	P17CFG=4'h0	×	×	≠2' b01	
	SPI_MISO	Input	0	1	×	×	×	×	×	2' b01	See 2.3.11
		Output	0	0	0	0	×	×	×		
epwmo07	Output	0	0	0	0	P17CFG=4'h0	×	×	≠2' b01	See 2.5.3	

	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P17CFG	×	≠2' b01	See 2.3.10
	Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P17CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	≠2' b01	See 2.3.9

Table 2-9 Example of register setting when using P20 to P27 pin functions

Pin name	Used function		PMCxx	PMxx	Pxx	POMxx	PxxCFG (Output multiplexing configuration register)	xxPCFG (Input multiplexing configuration register)	SPIPCFG	Remark
	Function name	Input/output								
P20	P20	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P20CFG=4' h0	×	×	
		N-channel open drain output	0	0	0/1	1			×	
	ANI0/AVREFP/VCIN12/PGA0TO	Analog channel	1	×	×	×	×	×	×	
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P20CFG	×	×	See 2.3.10
	Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P20CFG=4' h0	Configure SDI00PCFG/ SDAA0PCFG/SCLA0PCFG	×	See 2.3.9
P21	P21	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P21CFG=4' h0	×	×	
		N-channel open drain output	0	0	0/1	1			×	
	ANI1/AVREFM/VCIN13	Analog channel	1	×	×	×	×	×	×	
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P21CFG	×	×	See 2.3.10
	Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P21CFG=4' h0	Configure SDI00PCFG/ SDAA0PCFG/SCLA0PCFG	×	See 2.3.9
P22	P22	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P22CFG=4' h0	×	≠2' b11	
		N-channel open drain output	0	0	0/1	1			≠2' b11	
	ANI2/AVREFM/VCIN0/PGA0IN	Analog channel	1	×	×	×	×	×	×	
	SPI_MOSI	Input	0	1	×	×	×	×	2' b11	See 2.3.11
		Output	0	0	0	0	×	×		
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P22CFG	×	≠2' b11	See 2.3.10
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P22CFG=4' h0	Configure SDI00PCFG/ SDAA0PCFG/SCLA0PCFG	≠2' b11	See 2.3.9	
P23	P23	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P23CFG=4' h0	×	≠2' b11	
		N-channel open drain output	0	0	0/1	1			≠2' b11	
	ANI3/PGA0GND	Analog channel	1	×	×	×	×	×	×	
	SPI_MISO	Input	0	1	×	×	×	×	2' b11	See 2.3.11
		Output	0	0	0	0	×	×		
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P23CFG	×	≠2' b11	See 2.3.10
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P23CFG=4' h0	Configure SDI00PCFG/ SDAA0PCFG/SCLA0PCFG	≠2' b11	See 2.3.9	

Pin name	Used function		PMC _x	PM _x	P _x	POM _x	PxxCFG (Output multiplexing configuration register)	xxPCFG (Input multiplexing configuration register)	SPIPCFG	Remark
	Function name	Input/output								
P24	P24	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P24CFG=4 'h0	×	≠2' b11	
		N-channel open drain output	0	0	0/1	1				
	ANI4/PGA1IN	Analog channel	1	×	×	×	×	×	×	
	SPL_SCK	Input	0	1	×	×	×	×	2' b11	See 2.3.11
		Output	0	0	0	0	×	×		
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P24CFG	×	≠2' b11	See 2.3.10
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P24CFG=4 'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	≠2' b11	See 2.3.9	
P25	P25	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P25CFG=4 'h0	×	×	
		N-channel open drain output	0	0	0/1	1				
	ANI5/PGA1GND	Analog channel	1	×	×	×	×	×	×	
	SPL_NSS	Input	0	1	×	×	×	×	2' b11	See 2.3.11
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P25CFG	×	×	See 2.3.10
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P25CFG=4 'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9	
P26	P26	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P26CFG=4 'h0	×	×	
		N-channel open drain output	0	0	0/1	1				
	ANI6	Analog channel	1	×	×	×	×	×	×	
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P26CFG	×	×	See 2.3.10
	Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P26CFG=4 'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9
P27	P27	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P27CFG=4 'h0	×	×	
		N-channel open drain output	0	0	0/1	1				
	ANI7	Analog channel	1	×	×	×	×	×	×	
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P27CFG	×	×	See 2.3.10
	Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P27CFG=4 'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9

Table 2-10 Example of register setting when using P30 to P31 pin functions

Pin name	Used function		PMCxx	PMxx	Pxx	POMxx	PxxCFG (Output multiplexing configuration register)	xxPCFG (Input multiplexing configuration register)	SPIPCFG	Remark
	Function name	Input/output								
P30	P30	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P30CFG=4'h0	×	×	
		N-channel open drain output	0	0	0/1	1				
	ANI21	Analog channel	1	×	×	×	×	×	×	
	INTP3	Input	0	1	×	×	×	INTP3PCFG=6'h00	×	INTP3 can also be mapped to other ports, see 2.3.9.
	RTC1HZ	Output	0	0	0	0	P30CFG=4'h0	×	×	
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P30CFG	×	×	See 2.3.10
	Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P30CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9
P31	P31	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P31CFG=4'h0	×	#2' b10	
		N-channel open drain output	0	0	0/1	1				
	ANI22	Analog channel	1	×	×	×	×	×	×	
	TI03	Input	0	1	×	×	×	×	×	
	TO03	Output	0	0	0	0	P31CFG=4'h0	×	#2' b10	
	SPI_SCK	Input	0	1	×	×	×	×	2' b10	See 2.3.11
		Output	0	0	0	0	×	×		
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
Mappable multiplexing output	Output	0	0	0/1	0	Configure P31CFG	×	#2' b10	See 2.3.10	
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P31CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	#2' b10	See 2.3.9	

Table 2-11 Example of register setting when using the P40 to P41 pin functions

Pin name	Used function		PMCx	PMx	Px	POMx	PxxCFG (Output multiplexing configuration register)	xxPCFG (Input multiplexing configuration register)	SPIPCFG	Remark
	Function name	Input/output								
P40	P40	Input	-	1	×	×	×	×	×	
		Output	-	0	0/1	0	P40CFG=4, h0	×	×	
		N-channel open drain output	-	0	0/1	1			×	
	Mappable multiplexing input	Input	-	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	-	0	0/1	0	Configure P40CFG	×	×	See 2.3.10
	Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	-	0	0/1	1	P40CFG=4, h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9
P41	P41	Input	-	1	×	×	×	×	×	
		Output	-	0	0/1	0	P41CFG=4, h0	×	×	
		N-channel open drain output	-	0	0/1	1			×	
	Mappable multiplexing input	Input	-	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	-	0	0/1	0	Configure P41CFG	×	×	See 2.3.10
	Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	-	0	0/1	1	P41CFG=4, h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9

Table 2-12 Example of register setting when using P50 to P51 pin functions

Pin name	Used function		PMC _x	PM _x	P _x	POM _x	P _{xx} CFG (Output multiplexing configuration register)	xxPCFG (Input multiplexing configuration register)	SPIPCFG	Remark
	Function name	Input/output								
P50	P50	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P50CFG=4'h0	×	×	
		N-channel open drain output	0	0	0/1	1			×	
	ANI23	Analog channel	1	×	×	×	×	×	×	
	INTP1	Input	0	1	×	×	×	×	×	INTP1 can also be mapped to other ports, see 2.3.9.
	VCOU1	Output	0	0	0	0	P50CFG=4'h0	×	×	
	SPI_NSS	Input	0	1	×	×	×	×	2' b01	See 2.3.11
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P50CFG	×	×	See 2.3.10
	Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P50CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9
P51	P51	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P51CFG=4'h0	×	≠2' b01	
		N-channel open drain output	0	0	0/1	1			≠2' b01	
	ANI24	Analog channel	1	×	×	×	×	×	×	
	INTP2	Input	0	1	×	×	×	×	×	INTP2 can also be mapped to other ports, see 2.3.9.
	SPI_SCK	Input	0	1	×	×	×	×	2' b01	See 2.3.11
		Output	0	0	0	0	×	×		
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P51CFG	×	≠2' b01	See 2.3.10
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P51CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	≠2' b01	See 2.3.9	

Table 2-13 Example of register setting when using the P60 to P63 pin functions

Pin name	Used function		PMCxx	PMxx	Pxx	POMxx	PxxCFG (Output multiplexing configuration register)	xxPCFG (Input multiplexing configuration register)	SPIPCFG	Remark
	Function name	Input/output								
P60	P60	Input	-	1	×	×	×	×	×	
		N-channel open drain output	-	0	0/1	-	P60CFG=4'h0	×	×	
	Mappable multiplexing input	Input	-	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	-	0	0/1	1	P60CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9
P61	P61	Input	-	1	×	×	×	×	×	
		N-channel open drain output	-	0	0/1	1	P61CFG=4'h0	×	×	
	Mappable multiplexing input	Input	-	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	-	0	0/1	1	P61CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9
P62	P62	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P62CFG=4'h0	×	×	
		N-channel open drain output	0	0	0/1	1			×	
	ANI27	Analog channel	1	×	×	×	×	×	×	
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P62CFG	×	×	See 2.3.10
	Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P62CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9
P63	P63	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P63CFG=4'h0	×	×	
		N-channel open drain output	0	0	0/1	1			×	
	ANI28	Analog channel	1	×	×	×	×	×	×	
	SPI_NSS	Input	0	1	×	×	×	×	2' b10	See 2.3.11
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P63CFG	×	×	See 2.3.10
	Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P63CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9

Table 2-14 Example of register setting when using the P70 to P75 pin functions

Pin name	Used function		PMCx	PMx	Px	POMx	PxxCFG (Output multiplexing configuration register)	xxPCFG (Input multiplexing configuration register)	SPIPCFG	Remark
	Function name	Input/output								
P70	P70	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0			×	
		N-channel open drain output	0	0	0/1	1	P70CFG=4'h0	×	×	
	ANI29	Analog channel	1	×	×	×	×	×	×	
	KR0	Input	0	1	×	×	×	×	×	
	SCLK21	Input	0	1	×	×	×	×	×	
		Output	0	0	1	0	P70CFG=4'h0	×	×	
	SCL21	Output	0	0	1	0	P70CFG=4'h0	×	×	
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
Mappable multiplexing output	Output	0	0	0/1	0	Configure P70CFG	×	×	See 2.3.10	
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P70CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9	
P71	P71	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0			×	
		N-channel open drain output	0	0	0/1	1	P71CFG=4'h0	×	×	
	ANI30	Analog channel	1	×	×	×	×	×	×	
	KR1	Input	0	1	×	×	×	×	×	
	SDI21	Input	0	1	×	×	×	×	×	
	SDA21	Bi-directional	0	0	1	1	P71CFG=4'h0	×	×	
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P71CFG	×	×	See 2.3.10
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P71CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9	
P72	P72	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0			×	
		N-channel open drain output	0	0	0/1	1	P72CFG=4'h0	×	×	
	ANI31	Analog channel	1	×	×	×	×	×	×	
	KR2	Input	0	1	×	×	×	×	×	
	SDO21	Output	0	1	×	×	×	×	×	
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P72CFG	×	×	See 2.3.10
	Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P72CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9

Pin name	Used function		PMCxx	PMxx	Pxx	POMxx	PxxCFG (Output multiplexing configuration register)	xxPCFG (Input multiplexing configuration register)	SPIPCFG	Remark
	Function name	Input/output								
P73	P73	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P73CFG=4'h0	×	×	
		N-channel open drain output	0	0	0/1	1				×
	ANI32	Analog channel	1	×	×	×	×	×	×	
	KR3	Input	0	1	×	×	×	×	×	
	SDO01	Output	0	1	×	×	×	×	×	
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P73CFG	×	×	See 2.3.10
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P73CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9	
P74	P74	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P74CFG=4'h0	×	≠2' b10	
		N-channel open drain output	0	0	0/1	1				×
	ANI33	Analog channel	1	×	×	×	×	×	×	
	KR4	Input	0	1	×	×	×	×	×	
	SDI01	Input	0	1	×	×	×	×	×	
	SDA01	Bi-directional	0	0	1	1	P74CFG=4'h0	×	≠2' b10	
	SPI_MOSI	Input	0	1	×	×	×	×	2' b10	See 2.3.11
		Output	0	0	0	0	×	×		
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
Mappable multiplexing output	Output	0	0	0/1	0	Configure P74CFG	×	≠2' b10	See 2.3.10	
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P74CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	≠2' b10	See 2.3.9	
P75	P75	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P75CFG=4'h0	×	≠2' b10	
		N-channel open drain output	0	0	0/1	1				×
	ANI34	Analog channel	1	×	×	×	×	×	×	
	KR5	Input	0	1	×	×	×	×	×	
	SCLK01	Input	0	1	×	×	P75CFG=4'h0	×	≠2' b10	
		Output	0	0	1	0				
	SCL01	Output	0	0	1	0	P75CFG=4'h0	×	≠2' b10	
	SPI_MISO	Input	0	1	×	×	×	×	2' b10	See 2.3.11
		Output	0	0	0	0	×	×		
Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9	
Mappable multiplexing output	Output	0	0	0/1	0	Configure P75CFG	×	≠2' b10	See 2.3.10	
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P75CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	≠2' b10	See 2.3.9	

Table 2-15 Example of register setting when using the P120 to P124 pin functions

Pin name	Used function		PMCxx	PMxx	Pxx	POMxx	PxxCFG (Output multiplexing configuration register)	xxPCFG (Input multiplexing configuration register)	SPIPCFG	Remark
	Function name	Input/output								
P120	P120	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P120CFG=4'h0	×	×	
		N-channel open drain output	0	0	0/1	1			×	
	ANI14	Analog channel	1	×	×	×	×	×	×	
	VCOU0	Output	0	0	0	0	P120CFG=4'h0	×	×	
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P120CFG	×	×	See 2.3.10
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P120CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	×	See 2.3.9
P121	P121	Input	-	1	×	-	×	×	×	
		Output	-	0	0/1	-	P121CFG=4'h0	×	×	
		N-channel open drain output	-	0	0/1	-			×	
	X1	-	-	×	×	-	×	×	×	EXCLK=0, OSCSEL=1
	Mappable multiplexing input	Input	-	1	×	-	×	Configure xxPCFG	×	EXCLK=0, OSCSEL=0 See 2.3.9
Mappable multiplexing output	Output	-	0	0/1	-	Configure P121CFG	×	×	EXCLK=0, OSCSEL=0 See 2.3.10	
P122	P122	Input	-	1	×	-	×	×	×	
		Output	-	0	0/1	-	P122CFG=4'h0	×	×	
		N-channel open drain output	-	0	0/1	-			×	
	X2	-	-	×	×	-	×	×	×	EXCLK=0, OSCSEL=1
	EXCLK	Input	-	×	×	-	×	×	×	EXCLK=1, OSCSEL=1
	Mappable multiplexing input	Input	-	1	×	-	×	Configure xxPCFG	×	EXCLK=0, OSCSEL=0 See 2.3.9
Mappable multiplexing output	Output	-	0	0/1	-	Configure P122CFG	×	×	EXCLK=0, OSCSEL=0 See 2.3.10	
P123	P123	Input	-	1	×	-	×	×	×	
		Output	-	0	0/1	-	P123CFG=4'h0	×	×	
		N-channel open drain output	-	0	0/1	-			×	
	XT1	-	-	×	×	-	×	×	×	EXCLKS=0, OSCSELS=1
	Mappable multiplexing input	Input	-	1	×	-	×	Configure xxPCFG	×	EXCLKS=0, OSCSELS=0 See 2.3.9
Mappable multiplexing output	Output	-	0	0/1	-	Configure P123CFG	×	×	EXCLKS=0, OSCSELS=0 See 2.3.10	
P124	P124	Input	-	1	×	-	×	×	×	
		Output	-	0	0/1	-	P124CFG=4'h0	×	×	
		N-channel open drain output	-	0	0/1	-			×	
	XT2	-	-	×	×	-	×	×	×	EXCLKS=0, OSCSELS=1
	EXCLKS	Input	-	×	×	-	×	×	×	EXCLKS=1, OSCSELS=1
Mappable multiplexing input	Input	-	1	×	-	×	Configure xxPCFG	×	EXCLKS=0, OSCSELS=0	

										See 2.3.9
	Mappable multiplexing output	Output	-	0	0/1	-	Configure P124CFG	×	×	EXCLKS=0, OSCSLS=0 See 2.3.10

Table 2-16 Example of register setting when using P130, P136, P137 pin functions

Pin name	Used function		PMCxx	PMxx	Pxx	POMxx	PxxCFG (Output multiplexing configuration register)	xxPCFG (Input multiplexing configuration register)	SPIPCFG	Remark
	Function name	Input/output								
P130	P130	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P130CFG=4' h0	×	×	
		N-channel open drain output	0	0	0/1	1			×	
	ANI35	Analog channel	1	×	×	×	×	×	×	
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P130CFG	×	×	See 2.3.10
	Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P130CFG=4' h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9
P136	P136	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P136CFG=4' h0	×	×	
		N-channel open drain output	0	0	0/1	1			×	
	ANI36	Analog channel	1	×	×	×	×	×	×	
	INTP0	Input	0	1	×	×	×	×	×	INTP0 can also be mapped to other ports, see 2.3.9.
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P136CFG	×	×	See 2.3.10
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P136CFG=4' h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9	
P137	P137	Input	-	1	×	×	×	×	×	
		Output	-	0	0/1	0	P137CFG=4' h0	×	×	
		N-channel open drain output	-	0	0/1	1			×	
	Mappable multiplexing input	Input	-	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	-	0	0/1	0	Configure P137CFG	×	×	See 2.3.10
	Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	-	0	0/1	1	P137CFG=4' h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9

Table 2-17 Example of register setting when using P140, P146, P147 pin functions

Pin name	Used function		PMCxx	PMxx	Pxx	POMxx	PxxCFG (Output multiplexing configuration register)	xxPCFG (Input multiplexing configuration register)	SPIPCFG	Remark
	Function name	Input/output								
P140	P140	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P140CFG=4' h0	×	×	
		N-channel open drain output	0	0	0/1	1			×	
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P140CFG	×	×	See 2.3.10
	Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P140CFG=4' h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9
P146	P146	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P146CFG=4' h0	×	×	
		N-channel open drain output	0	0	0/1	1			×	
	ANI15	Analog channel	1	×	×	×	×	×	×	
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P146CFG	×	×	See 2.3.10
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P146CFG=4' h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9	
P147	P147	Input	0	1	×	×	×	×	×	
		Output	0	0	0/1	0	P147CFG=4' h0	×	×	
		N-channel open drain output	0	0	0/1	1			×	
	ANI12/IVREF0	Analog channel	1	×	×	×	×	×	×	
	Mappable multiplexing input	Input	0	1	×	×	×	Configure xxPCFG	×	See 2.3.9
	Mappable multiplexing output	Output	0	0	0/1	0	Configure P147CFG	×	×	See 2.3.10
Mappable bi-directional communication (SDA00/SDAA0/SCLA0)	Bi-directional	0	0	0/1	1	P147CFG=4' h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	See 2.3.9	

2.5.3 EPWM port configuration methods

When using the EPWM output control circuit function, the EPWM output pins are fixedly mapped to P10~P17, and the configuration method is as follows.

Port name	Function	Input/output	PxxCFP	PMCxx	PMxx	POMxx	Pxx	Remark
P10	epwmo00	Output	P10CFG=4'h0	0	0	0	0	
P11	epwmo01	Output	P11CFG=4'h0	0	0	0	0	
P12	epwmo02	Output	P12CFG=4'h0	0	0	0	0	
P13	epwmo03	Output	P13CFG=4'h0	0	0	0	0	
P14	epwmo04	Output	P14CFG=4'h0	0	0	0	0	
P15	epwmo05	Output	P15CFG=4'h0	0	0	0	0	When epwmo05 is output, keep CLKBUZ1 output as "0".
P16	epwmo06	Output	P16CFG=4'h0	0	0	0	0	
P17	epwmo07	Output	P17CFG=4'h0	0	0	0	0	When epwmo07 is output, keep TO02 output as "0".

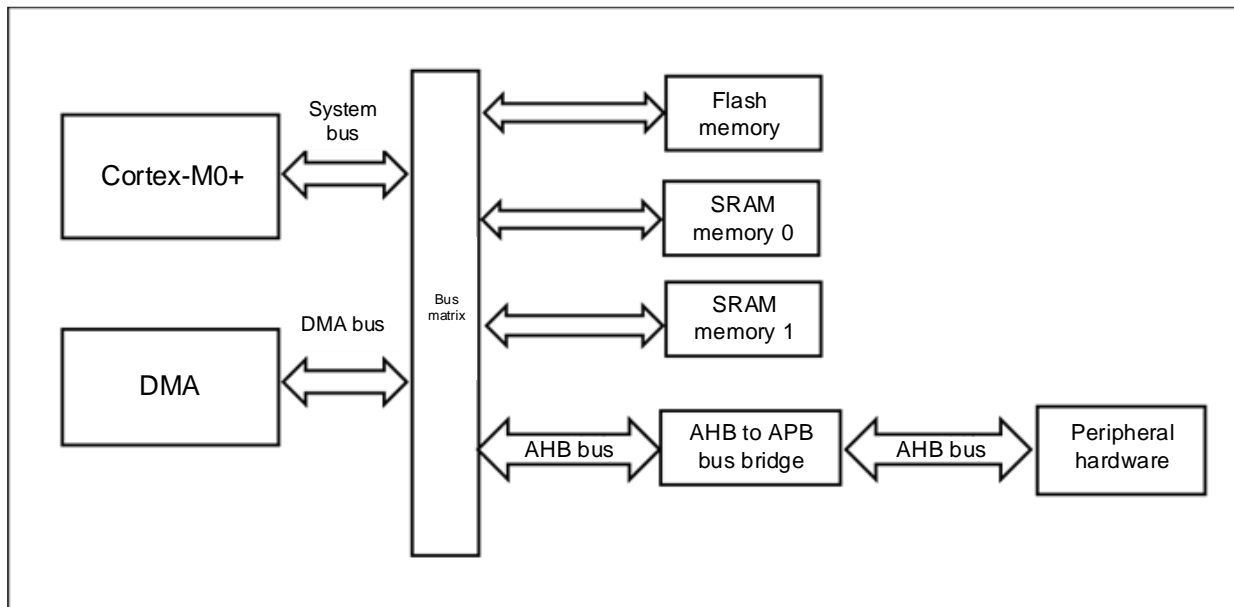
Chapter 3 System Structure

3.1 Overview

The product system consists of the following components.

- 2 AHB buses (Master):
 - Cortex-M0+
 - Enhanced DMA
- 4 AHB buses (Slave):
 - FLASH memory
 - SRAM memory 0
 - SRAM memory 1
 - AHB to APB Bridge, contains all APB interface peripherals

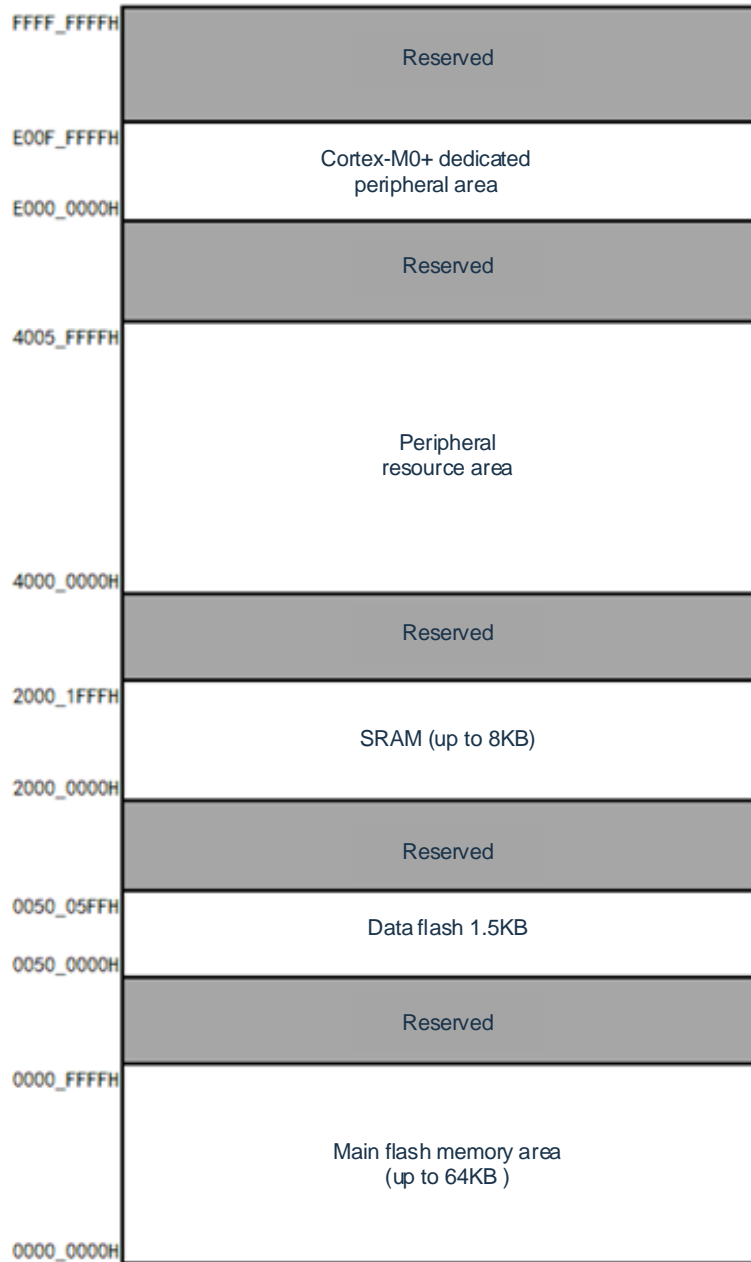
Figure 3-1 Diagram of system structure



- System bus: This bus connects the system bus (peripheral bus) of the Cortex-M0+ kernel to the bus matrix, which coordinates access between the kernel and DMA.
- DMA bus: The bus connects the AHB master interface of the DMA to a bus matrix that coordinates CPU and DMA access to SRAM, flash and peripherals.
- Bus matrix: The bus matrix coordinates the access arbitration between the kernel system bus and the DMA master bus, and the arbitration adopts a fixed priority, and the DMA priority is high.
- AHB to APB Bridge: AHB to APB Bridge provides a synchronous connection between the AHB and APB buses. Refer to for address mapping of the different peripherals connected to each bridge Table 3-1.

3.2 System address partitioning

Figure 3-2 Map of address partitioning



Peripheral address assignment

Table 3-1 Starting address of the peripheral's registers

Start address	Peripheral	Remark
0x4000_0000 - 0x4000_4FFF	Reserved	
0x4000_5000 - 0x4000_5FFF	DMA	
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	Clock control	
0x4002_1000 - 0x4002_1001	WDT	
0x4002_1002 - 0x4002_1800	Reserved	
0x4002_1800 - 0x4002_1BFF	High-speed CRC	See Chapter 26 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	See Chapter 26 Safety Function
0x4004_3300 - 0x4004_33FF	Reserved	
0x4004_3400 - 0x4004_37FF	Linkage controller	
0x4004_3800 - 0x4004_3BFF	CMP	
0x4004_3C00 - 0x4004_3FFF	Reserved	
0x4004_4000 - 0x4004_43FF	IrDA	
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 of resonator connection pin/external clock input pin for the main system clock and the resonator connection pin/external clock input pin for the subsystem clock are different among products.

4.1 Function of clock generation circuit

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

(1) Main system clock

① X1 oscillation circuit

The clock can be oscillated from $f_x=1$ to 20MHz by connecting resonators to pins X1 and X2, and the oscillation can be stopped by entering the 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 can be selected from $f_{HOCO}=64\text{MHz}$, 48MHz, 32MHz, 24MHz, 16MHz, 12MHz, 8MHz, 6MHz, 4MHz, 3MHz, 2MHz, and 1MHz (TYP.) with the option byte (000C2H). After the reset is released, the CPU must start operation with this high-speed on-chip oscillator clock. The 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 through the frequency select register (HOCODIV) of the high-speed on-chip oscillator. For the frequency setting, refer to “Figure 4-10 Format of high-speed on-chip oscillator frequency select register (HOCODIV)”.

In addition, an external master system clock ($f_{EX}=1\sim 20\text{MHz}$) can be provided by the EXCLK/X2/P122 pin, and the external master system clock input can be disabled by entering deep sleep mode or setting the MSTOP bit.

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

(2) Subsystem clock

- XT1 oscillation circuit

The XT1 pin and XT2 pin are connected to a 32.768kHz resonator to oscillate the clock with $f_{XT}=32.768\text{kHz}$ and to stop the oscillation by setting a XTSTOP bit (bit6 of the clock operation status control register (CSC)).

In addition, an external subsystem clock ($f_{EXS}=32.768\text{kHz}$) can be provided by the EXCLKS/XT2/P124 pin, and the external subsystem clock input can be disabled by setting the XTSTOP bit.

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

Can oscillate a clock with $f_{IL}=15\text{kHz}$.

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

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

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

Notice The low-speed internal oscillator clock (f_{IL}) can be selected as the count clock of the real-time clock only when a fixed period 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

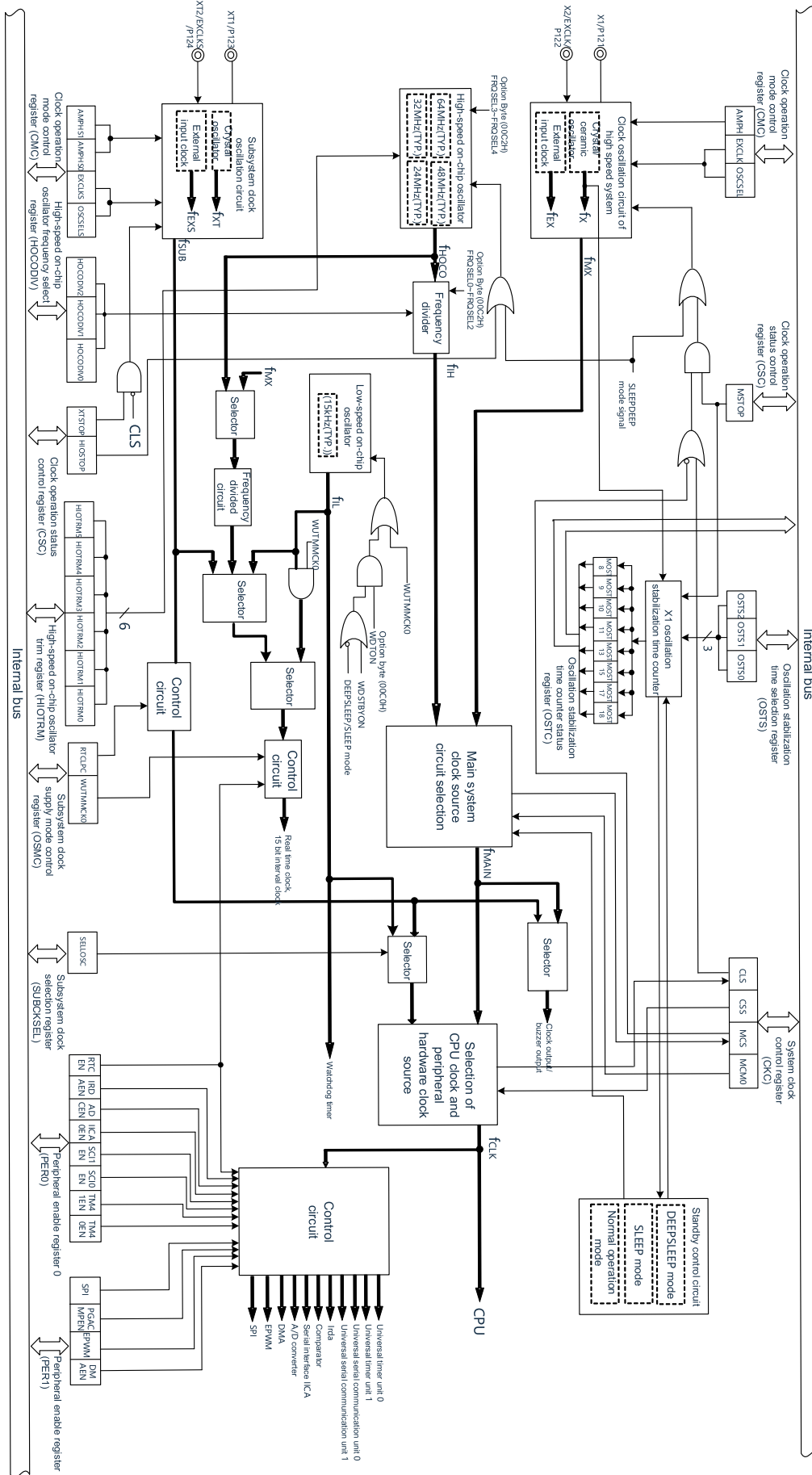
4.2 Structure of clock generation circuit

The clock generation circuit consists of the following hardware.

Table 4-1 Structure of clock generation circuit

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

Notice The assigned registers and bits differ depending on the product. The initial value must be set for unassigned bits.

4.3.1 Clock operation mode control register (CMC)

This is a register that sets the operation mode of the X1/P121, X2/EXCLK/P122, XT1/P123, XT2/EXCLKS/P124 pin and selects the gain of the oscillating circuit.

The CMC register can only be written 1 time by an 8-bit memory operation instruction after reset. The register can be read by an 8-bit memory operation instruction.

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

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

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

Symbol	7	6	5	4	3	2	1	0
CMC	EXCLK	OSCSEL	EXCLKS ^{Note}	OSCSELS ^{Note}	0	AMPHS1 ^{Note}	AMPHS0 ^{Note}	AMPH

EXCLK	OSCSEL	High-speed system clock pin operation mode	X1/P121 pin	X2/EXCLK/P122 pin
0	0	Port mode	Input/output port	
0	1	X1 oscillation mode	Connects 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	Subsystem clock pin operation mode	XT1/P123 pin	XT2/EXCLKS/P124 pin
0	0	Port mode	Input/output port	
0	1	XT1 oscillation mode	Connects 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	Ultra-low power oscillation
1	1	Disable setting.

AMPH	Control of X1 clock oscillation frequency
0	$1\text{MHz} \leq f_x \leq 10\text{MHz}$
1	$10\text{MHz} < f_x \leq 20\text{MHz}$

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

Notice1. After the reset is released, the CMC register can only be written 1 time through an 8-bit memory instruction.

When the CMC register is used at the initial value ("00H"), it is necessary to set the CMC register to "00H" after releasing the reset in order to prevent malfunction when the program is out of control (the value other than "00H" cannot be recovered if it is written by mistake).

2. After reset is removed, the CMC register must be set before starting X1 or XT1 oscillation by setting the Clock Operation Status Control Register (CSC).
3. When the X1 clock oscillation frequency exceeds 10MHz, the AMPH bit must be set to "1".
4. The AMPH bit, AMPHS1 bit, and AMPHS0 bit must be set after the reset is released and with f_{IH} selected as the state of f_{CLK} (the state before switching f_{CLK} to f_{MX} or f_{SUB}).
5. The oscillation stabilization time of the f_{XT} must be counted by software.
6. The upper frequency limit of the system clock is 64MHz, but the upper frequency limit of the X1 oscillator circuit is 20MHz.

Remark 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 operation instruction. After a reset signal is generated, the value of this register changes to “00H”.

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

Address: 40020404H After reset: 00H R/W ^{Note 1}

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

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

CSS ^{Note2}	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 ^{Note2}	Main system clock (f_{MAIN}) operation control
0	Select the high-speed on-chip oscillator clock (f_{IH}) as the mainsystem clock (f_{MAIN}).
1	Select the high-speed system clock (f_{MX}) as the main system clock (f_{MAIN}).

Note 1. Bit7 and bit5 are read-only bits.

2. It is prohibited to change the MCM0 bit value while the CSS bit is set to “1”.

Remark 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

Notice1. Bit0~3 must be set to 0.

- Supply CSS bit set 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.
- If the subsystem clock is used as the peripheral hardware clock, the A/D converter and IICA cannot be guaranteed. For operation characteristics of the peripheral hardware, refer to the electrical characteristics of the corresponding section and datasheet for each peripheral hardware.

4.3.3 Clock operation status control register (CSC)

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

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

Figure 4-4 Format of 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 oscillator circuit runs	EXCLK pin external clock is valid	Input/output port
1	X1 oscillator circuit stops	EXCLK pin external clock is invalid	

XTSTOP	Operation control of subsystem clock		
	XT1 oscillation mode	External clock input mode	Port mode
0	XTT1 oscillator circuit runs	EXCLKS pin external clock is valid	Input/output port
1	XT1 oscillator circuit stops	EXCLKS pin external clock 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		

Notice1. After the reset is released, the CSC register must be set after setting the Clock Operation Mode Control Register (CMC).

- After the reset is released and before the MSTOP bit set to "0", an oscillator stabilization time selection register (OSTS) must be set. However, when using the OSTS register at the initial value, you do not need to set the OSTS register.
- When X1 oscillation is started by setting the MSTOP bit, the oscillation stability time of the X1 clock must be confirmed by OSTC.
- When starting XT1 oscillation by setting the XSTOP bit, you must wait for the oscillation stabilization time required by the subsystem clock through software.
- The clock selected as the CPU/peripheral hardware clock (f_{CLK}) cannot be stopped through the CSC register.
- Refer to Table 4-2 for register flag settings for stopping clock oscillation (invalid external clock input) and conditions before stopping.

Table 4-2 Clock stopping method

Clock	Condition before clock stops (invalid external clock input)	Set CSC register flag
X1 clock	The CPU/peripheral hardware clock runs at 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	The CPU/peripheral hardware clock runs at a clock other than the subsystem clock. (CLS=0)	XTSTOP=1
External subsystem clock		
High-speed on-chip oscillator clock	The CPU/peripheral hardware clock operates at a clock other than the high-speed internal oscillator clock. (CLS=0 and MCS=1, or CLS=1)	HIOSTOP=1

4.3.4 Oscillation stabilization time counter status register (OSTC)

This is a register that represents the count state of the oscillating steady-time counter of the X1 clock. The oscillation stability time of the X1 clock can be confirmed under the following circumstances:

- When the CPU clock is a high-speed on-chip oscillator clock or a subsystem clock and the oscillation of the X1 clock is started.
- When the CPU clock is a high-speed on-chip oscillator clock and the sleep mode is released after transferring to deep sleep mode in the X1 clock oscillation state.

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

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

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

- When the X1 clock starts to oscillate (EXCLK, OSCSEL=0, 1→MSTOP=0)
- When deep sleep mode is released

Figure 4-5 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

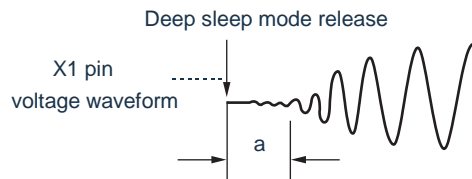
MOS T8	MOST 9	MOST 10	MOST 11	MOST 13	MOST 15	MOST 17	MOST 18	Oscillation stabilization time status		
								$f_X=10\text{MHz}$	$f_X=20\text{MHz}$	
0	0	0	0	0	0	0	0	Max. $2^8/f_X$	Max. 25.6 μs	Max. 12.8 μs
1	0	0	0	0	0	0	0	Min. $2^8/f_X$	Min. 25.6 μs	Min. 12.8 μs
1	1	0	0	0	0	0	0	Min. $2^9/f_X$	Min. 51.2 μs	Min. 25.6 μs
1	1	1	0	0	0	0	0	Min. $2^{10}/f_X$	Min. 102 μs	Min. 51.2 μs
1	1	1	1	0	0	0	0	Min. $2^{11}/f_X$	Min. 204 μs	Min. 102 μs
1	1	1	1	1	0	0	0	Min. $2^{13}/f_X$	Min. 819 μs	Min. 409 μs
1	1	1	1	1	1	0	0	Min. $2^{15}/f_X$	Min. 3.27ms	Min. 1.63ms
1	1	1	1	1	1	1	0	Min. $2^{17}/f_X$	Min. 13.1ms	Min. 6.55ms
1	1	1	1	1	1	1	1	Min. $2^{18}/f_X$	Min. 26.2ms	Min. 13.1ms

Notice1. After the above time, each bit is changed to "1" from the MOST8 bit and remains at "1".

2. The oscillation stabilization time counter counts only within the oscillation stable time set by the OSTs. In the following cases, the setting value of the oscillation stabilization time of the OSTs register must be greater than the count value confirmed by the OSTC register.

- When the CPU clock is a high-speed internal oscillator clock or a subsystem clock and the X1 clock is about to start oscillating.
- When the CPU clock is a high-speed internal oscillator clock and is released from deep sleep mode after shifting to deep sleep mode in the state of X1 clock oscillation (therefore, it must be noted that the OSTC register after release from deep sleep mode only sets the state within the oscillation stabilization time set by the OSTs register).

3. The oscillation stabilization time of the X1 clock does not include the time before the clock starts oscillating (Figure a below).



Remark f_X : X1 clock oscillation frequency

4.3.5 Oscillation stabilization time select register (OSTS)

This is a register that selects the oscillation stabilization time of the X1 clock.

If the X1 clock is oscillated, 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 internal on-chip 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 the deep sleep mode after being transferred to the deep sleep mode while the X1 clock is oscillating, it is necessary to confirm whether the oscillation stabilization time has elapsed by means of the Oscillation Stabilization Time Counter Status Register (OSTC).

The OSTC register can be used to confirm the time set by the OSTS register.

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

Figure 4-6 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	Selection of oscillation stabilization time		
				$f_X=10\text{MHz}$	$f_X=20\text{MHz}$
0	0	0	$2^8/f_X$	25.6 μs	12.8 μs
0	0	1	$2^9/f_X$	51.2 μs	25.6 μs
0	1	0	$2^{10}/f_X$	102 μs	51.2 μs
0	1	1	$2^{11}/f_X$	204 μs	102 μs
1	0	0	$2^{13}/f_X$	819 μs	409 μs
1	0	1	$2^{15}/f_X$	3.27ms	1.63ms
1	1	0	$2^{17}/f_X$	13.1ms	6.55ms
1	1	1	$2^{18}/f_X$	26.2ms	13.1ms

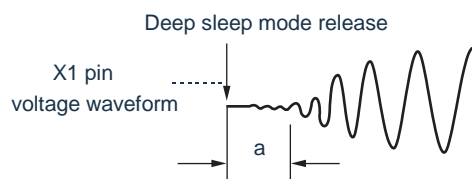
Notice1. To change the setting of the OSTS register, you must make the change before the MSTOP bit of the Clock Operation Status Control Register (CSC) set to 0.

2. The oscillation stabilization time counter is counted only in that oscillation stable time set in the OSTS register.

In the following cases, the setting value of the oscillation stabilization time of the OSTS register must be greater than the count value confirmed by the OSTC register after the start of the oscillation.

- When the CPU clock is a high speed on-chip oscillator clock or a subsystem clock and the X1 clock is to start oscillating.
- When the CPU clock is a high-speed on-chip oscillator clock and is released from deep sleep mode after shifting to deep sleep mode in the state of X1 clock oscillation (therefore, it must be noted that the OSTC register after release from deep sleep mode only sets the state within the oscillation stabilization time set by the OSTS register).

3. The oscillation stable time of the X1 clock does not include the time before the clock starts to oscillate (Figure a below).



Remark f_X : X1 clock oscillation frequency

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

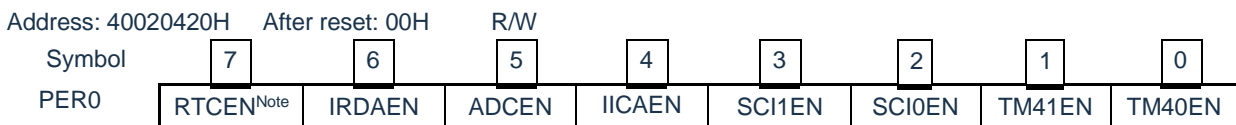
This is a register that sets a clock that is enabled or disabled for each peripheral hardware. Reduce power consumption and noise by stopping clock supply to unused hardware.

When the following peripheral functions controlled by these registers are used, the corresponding bit must be set to “1” before the initial setting of the peripheral functions.

- Real-time clock, 15-bit interval timer
- IrDA
- A/D converter
- Serial interface IICA0
- Universal serial communication unit 1
- Universal serial communication unit 0
- Universal timer unit 1
- Universal timer unit 0
- D/A converter
- Comparator
- Enhanced DMA
- EPWM
- SPI

The PER0 register and PER1 register are set by an 8-bit memory operation instruction. After the reset signal is generated, the values of these registers change to '00H'.

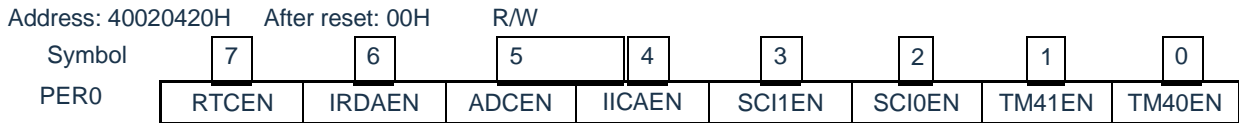
Figure 4-7 Format of peripheral enable register 0 (PER0) (1/3)



RTCEN	Input clock control of real-time clock (RTC) and 15-bit interval timer
0	Stops input clock supply. <ul style="list-style-type: none"> • SFR used by the real time clock (RTC) and 15-bit interval timer cannot be written. • The real-time clock (RTC) and 15-bit interval timer are in the reset status.
1	Enables input clock supply. <ul style="list-style-type: none"> • 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 when power-on reset, and remains unchanged during other reset conditions.

Figure 4-7 Format of peripheral enable register 0 (PER0) (2/3)



IRDAEN	Control of serial interface IRDA input clock supply
0	Stops input clock supply. • SFR used by the IRDA cannot be written. • The IRDA is in the reset status.
1	Enables input clock supply. • SFR used by the IRDA can be read and written.

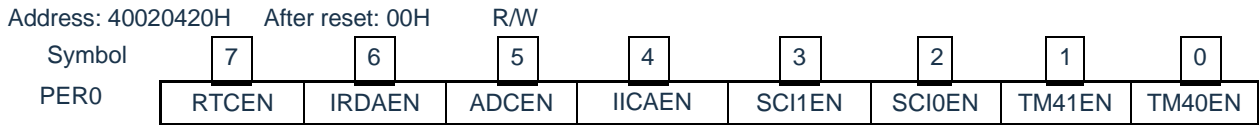
ADCEN	Control of A/D converter input clock supply
0	Stops input clock supply. • SFR used by the A/D converter cannot be written. • The A/D converter is in the reset status.
1	Enables input clock supply. • SFR used by the A/D converter can be read and written.

IICA0EN	Control of serial interface IICA0 input clock supply
0	Stops input clock supply. • SFR used by the serial interface IICA0 cannot be written. • The serial interface IICA0 is in the reset status.
1	Enables input clock supply. • SFR used by the serial interface IICA0 can be read and written.

SCI1EN	Control of universal communication unit 1 input clock supply
0	Stops input clock supply. • SFR used by the universal communication unit 1 cannot be written. • The universal communication unit 1 is in the reset status.
1	Enables input clock supply. • SFR used by the universal communication unit 1 can be read and written.

SCI0EN	Control of universal communication unit 0 input clock supply
0	Stops input clock supply. • SFR used by the universal communication unit 0 cannot be written. • The universal communication unit 0 is in the reset status.
1	Enables input clock supply. • SFR used by the universal communication unit 0 can be read and written.

Figure 4-7 Format of peripheral enable register 0 (PER0) (3/3)



TM41EN	Control of universal timer unit 1 input clock supply
0	Stops input clock supply. <ul style="list-style-type: none"> • SFR used by the universal timer unit 1 cannot be written. • The universal timer unit 1 is in the reset status.
1	Enables input clock supply. <ul style="list-style-type: none"> • SFR used by the universal timer unit 1 can be read and written.

TM40EN	Control of universal timer unit 0 input clock supply
0	Stops input clock supply. <ul style="list-style-type: none"> • SFR used by the universal timer unit 0 cannot be written. • The universal timer unit 0 is in the reset status.
1	Enables input clock supply. <ul style="list-style-type: none"> • SFR used by the universal timer unit 0 can be read and written.

Figure 4-8 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	PGACMPEN	0	DMAEN	EPWMEN	0	0

SPIEN	Control of SPI input clock supply
0	Stops input clock supply. • SPI cannot operate.
1	Enables input clock supply. • SPI can operate.

PGACMPEN	Control of amplifier and comparator input clock supply
0	Stops input clock supply. • SFR used by the amplifier and comparator cannot be written. • The amplifier and comparator are in the reset status.
1	Enables input clock supply. • SFR used by the amplifier and comparator can be read and written.

DMAEN	Control of DMA input clock supply
0	Stops input clock supply. • DMA can operate.
1	Enables input clock supply. • DMA can operate.

EPWMEN	Control of EPWM input clock supply
0	Stops input clock supply. • EPWM cannot operate.
1	Enables input clock supply. • EPWM can operate.

4.3.7 Subsystem clock supply mode control register (OSMC)

The OSMC register is a register that reduces power consumption by stopping unwanted clock functions.

If the RTCLPC bit is set to "1", it stops clocking peripheral functions other than the real-time clock and 15-bit interval timer in deep sleep mode or sleep mode where the CPU runs on the subsystem clock, thus reducing power consumption.

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

The OSMC register is set by an 8-bit memory operation instruction.

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

Figure 4-9 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	Settings in deep sleep mode and sleep mode while subsystem clock is selected as CPU clock
0	Enables supply of subsystem clock to peripheral functions (Refer to Table 19-1~Table 19-3 for enabled peripheral functions.
1	Stops supply of subsystem clock to peripheral functions other than real-time clock and 15-bit interval timer.

WUTMMCK0	Selection of operation clock for real-time clock and 15-bit interval timer
0	• Subsystem clock
1	• Low-speed on-chip oscillator clock

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

This is a register that changes the high-speed on-chip oscillator frequency 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 operation instruction.

After a reset signal is generated, the value becomes the value set by FRQSEL2~FRQSEL0 of the option byte (000C2H).

Figure 4-10 Format of high-speed on-chip oscillator frequency select register (HOCODIV)

Address: 40021C20H After reset: the value set by FRQSEL2 to FRQSEL0 of the option byte (000C2H)R/W

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

HOCODIV2	HOCODIV1	HOCODIV0	Selection of high-speed on-chip oscillator frequency			
			FRQSEL4=0		FRQSEL4=1	
			FRQSEL3=0	FRQSEL3=1	FRQSEL3=0	FRQSEL3=1
0	0	0	$f_{IH}=24\text{MHz}$ $f_{HOCO}=24\text{MHz}$	$f_{IH}=32\text{MHz}$ $f_{HOCO}=32\text{MHz}$	$f_{IH}=48\text{MHz}$ $f_{HOCO}=48\text{MHz}$	$f_{IH}=64\text{MHz}$ $f_{HOCO}=64\text{MHz}$
0	0	1	$f_{IH}=12\text{MHz}$ $f_{HOCO}=24\text{MHz}$	$f_{IH}=16\text{MHz}$ $f_{HOCO}=32\text{MHz}$	$f_{IH}=24\text{MHz}$ $f_{HOCO}=48\text{MHz}$	$f_{IH}=32\text{MHz}$ $f_{HOCO}=64\text{MHz}$
0	1	0	$f_{IH}=6\text{MHz}$ $f_{HOCO}=24\text{MHz}$	$f_{IH}=8\text{MHz}$ $f_{HOCO}=32\text{MHz}$	$f_{IH}=12\text{MHz}$ $f_{HOCO}=48\text{MHz}$	$f_{IH}=16\text{MHz}$ $f_{HOCO}=64\text{MHz}$
0	1	1	$f_{IH}=3\text{MHz}$ $f_{HOCO}=24\text{MHz}$	$f_{IH}=4\text{MHz}$ $f_{HOCO}=32\text{MHz}$	$f_{IH}=6\text{MHz}$ $f_{HOCO}=48\text{MHz}$	$f_{IH}=8\text{MHz}$ $f_{HOCO}=64\text{MHz}$
1	0	0	Setting prohibited	$f_{IH}=2\text{MHz}$ $f_{HOCO}=32\text{MHz}$	$f_{IH}=3\text{MHz}$ $f_{HOCO}=48\text{MHz}$	$f_{IH}=4\text{MHz}$ $f_{HOCO}=64\text{MHz}$
1	0	1	Setting prohibited	$f_{IH}=1\text{MHz}$ $f_{HOCO}=32\text{MHz}$	Setting prohibited	$f_{IH}=2\text{MHz}$ $f_{HOCO}=64\text{MHz}$
Other than above			Setting prohibited			

Notice1. The HOCODIV register must be set in a 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 through the HOCODIV register, the frequency switch is performed after the following transition time has elapsed:

- Operation for up to three clocks at the pre-change frequency
- CPU/peripheral hardware clock wait at the post-change frequency for up to three clocks

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. Self-measurement of the frequency of the high-speed internal oscillator and accuracy correction can be performed using a timer or the like with a high precision external clock input. The HIOTRM register is set by an 8-bit memory operation instruction.

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

Figure 4-11 Format of high-speed on-chip oscillator trimming 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	

Note The value after reset is the value adjusted at shipment.

Remark1. Every bit of the HIOTRM register can correct the clock accuracy of the high-speed internal oscillator by about 0.05%.

4.3.10 Subsystem clock selection register (SUBCKSEL)

The SUBCKSEL register is used to select 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 operation instruction.

After generating a reset signal, the value of this register becomes “00H”.

Figure 4-12 Format of subsystem clock selection 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	Selection of subsystem clock and low-speed on-chip oscillator clock
0	• Subsystem clock
1	• Low-speed on-chip oscillator clock

4.4 System clock oscillation circuit

4.4.1 X1 oscillation circuit

The X1 oscillation circuit oscillates by connecting a crystal resonator or a ceramic resonator (1 to 20MHz) of the X1 pin and X2 pins. An external clock can also be input, at which time a clock signal must be input to the EXCLK pin.

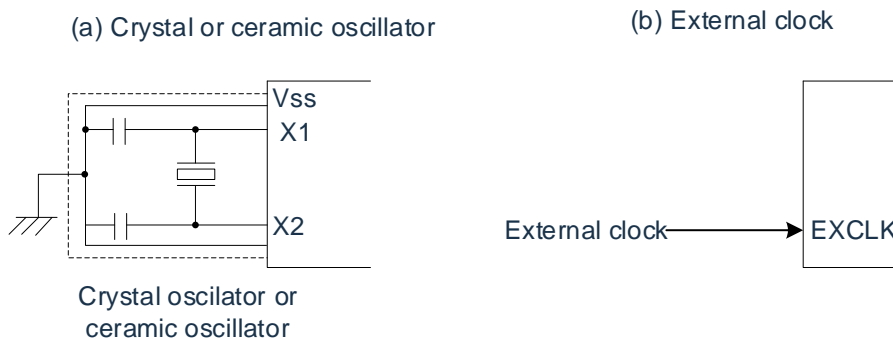
When using the X1 oscillator circuit, bit7 and bit6 (EXCLK, OSCSEL) of the clock mode control register must be set:

- Crystal or ceramic oscillation: EXCLK, OSCSEL=0, 1
- External clock input: EXCLK, OSCSEL=1, 1

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

Figure 4-13 shows an example of the external circuit of the X1 oscillator.

Figure 4-13 Example of external circuit of X1 oscillator



Notices are listed on the next page.

4.4.2 XT1 oscillation circuit

The XT1 oscillation circuit oscillates by connecting a crystal resonator (32.768kHz (TYP.)) of the XT1 pin and XT2 pin. When the XT1 oscillating circuit is used, the bit4 (OSCSELS) of the clock operation mode control register (CMC) must be set "1" to input the external clock, and the EXCLKS pin must be input.

When using the XT1 oscillator circuit, bit5 and bit4 (EXCLKS, OSCSELS) of the clock mode control (CMC) register must be set:

- Crystal oscillation : EXCLKS, OSCSELS=0, 1
- External clock input : EXCLKS, OSCSELS=1, 1

When the XT1 oscillator circuit is not used, it must be set to port mode (EXCLKS, OSCSELS=0, 0). Also, when not used as an input/output port, refer to “Table 2-3 Handling of unused pins”. Figure 4-14 shows an example of the external circuit of the XT1 oscillator.

Figure 4-14 Example of external circuit of XT1 oscillation circuit

(a) Crystal oscillation

(b) External clock

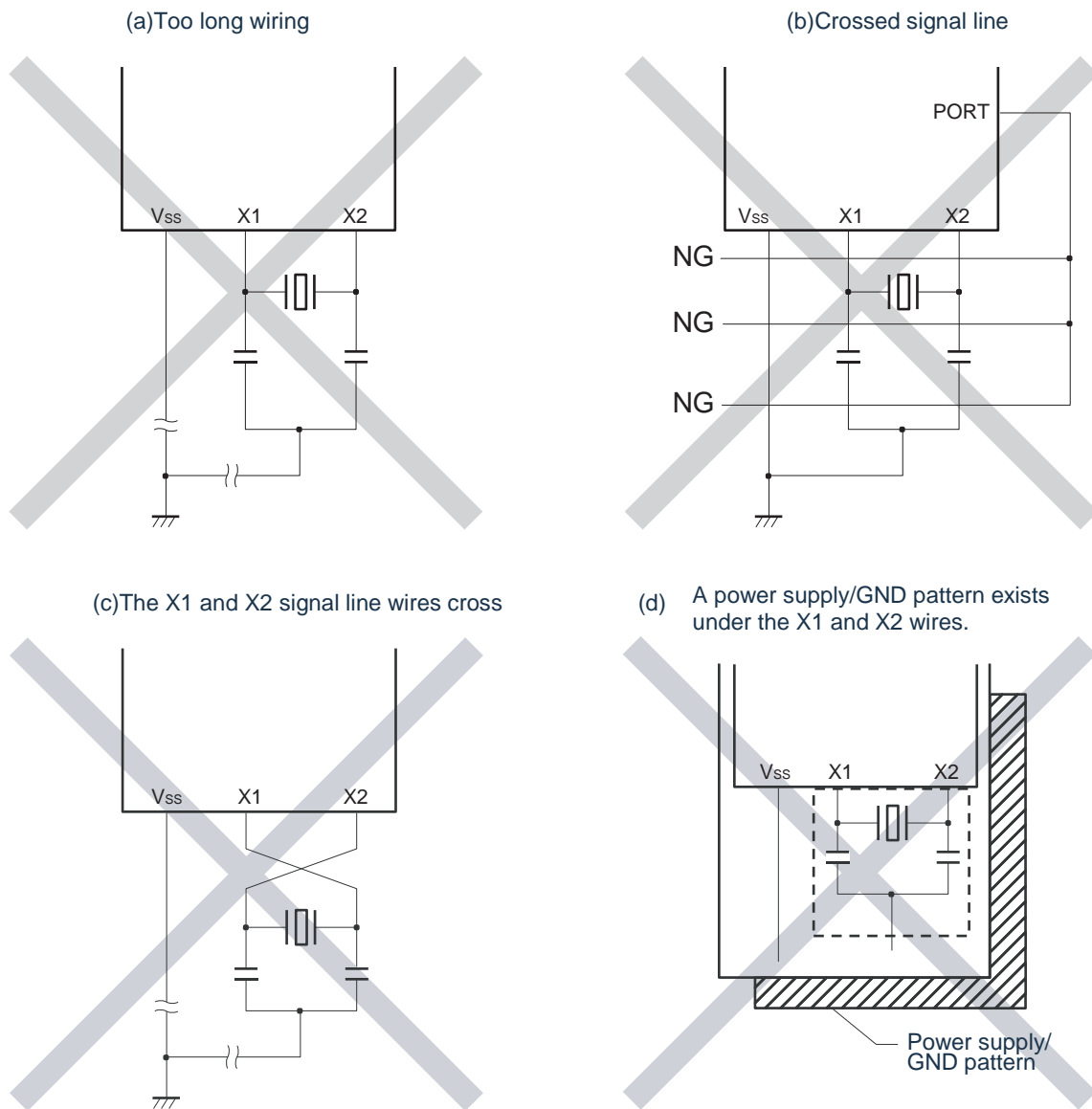


Notice When using the X1 oscillator and XT1 oscillator, wire as follows in the are enclosed by the broken lines in the Figure 4-13 and Figure 4-14 to avoid an adverse effect from wiring capacitance:

- Keep the wiring length as short as possible.
- Do not cross the wiring with the other signal lines. Do not route the wiring near a signal line through which a high fluctuating current flows.
- 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 flows.
- Do not fetch signals from the oscillator.

Figure 4-15 lists some incorrect resonator connection examples.

Figure 4-15 Incorrect resonator connection examples (1/2)

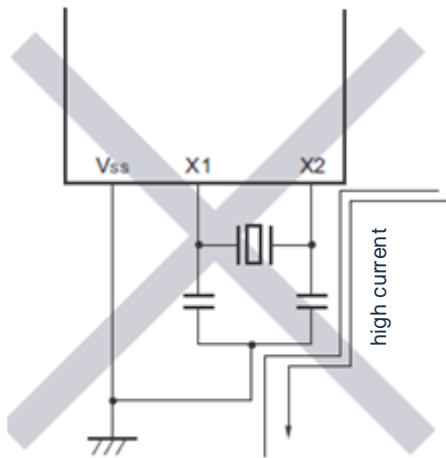


Note Do not place a power supply/GND pattern under the wiring section (section indicated by a broken line in the figure) of the X1 and X2 pins and the resonators in a multi-layer board or double-sided board. Do not configure a layout that will cause capacitance elements and affect the oscillation characteristics.

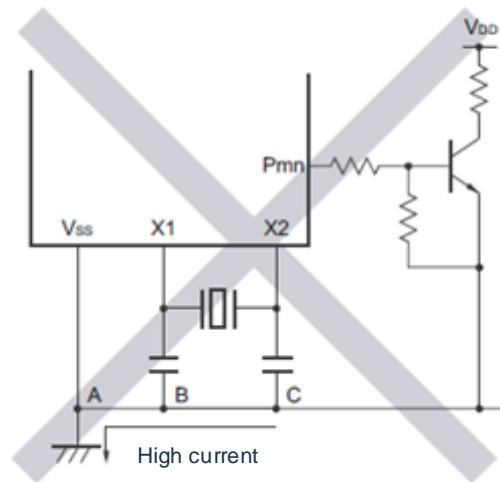
Remark When using the subsystem clock, replace X1 and X2 with XT1 and XT2, respectively, and insert a series resistor on the XT2 side when reading.

Figure 4-14 Incorrect resonator connection examples (2/2)

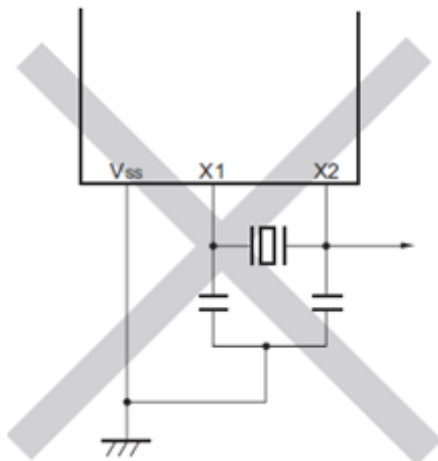
(e) Wiring near high alternating current



(f) Current flowing through ground line of oscillation circuit (Potential at points A,B and C fluctuates)



(g) Signals are fetched



Notice When X2 and XT1 pins are wired in parallel, the crosstalk noise of X2 pin may increase with XT1 pin, resulting in malfunctioning.

Remark When using the subsystem clock, replace X1 and X2 with XT1 and XT2, respectively, and insert a series resistor on the XT2 side when reading.

4.4.3 High-speed on-chip oscillator

The BAT32G135 has a built-in high-speed on-chip oscillator. Frequency can be selected from 64MHz, 48MHz, 32MHz, 24MHz, 16MHz, 12MHz, 8MHz, 6MHz, 4MHz, 3MHz, 2MHz, 1MHz and via option bytes (000C2H). The oscillation can be controlled by the bit0 (HIOSTOP) of the clock operation status control register (CSC).

After the reset is released, the high-speed on-chip oscillator automatically starts to oscillate.

4.4.4 Low-speed on-chip oscillator

The BAT32G135 has a built-in low-speed on-chip oscillator.

The low-speed on-chip oscillator clock is used as the watchdog timer, real-time clock, 15-bit interval timer clock, and an external reference clock of SysTick timer, it can also be used as CPU clock and peripheral module clock.

The low-speed on-chip oscillator oscillates when the bit4 (WDTON) of the option byte (000C0H) or bit4 (WUTMMCK0) of the sub-system clock providing mode control register (OSMC).

The low-speed internal oscillator continues to oscillate when the watchdog timer stops running and the WUTMMCK0 bit is not "0". However, if the watchdog timer runs and the WUTMMCK0 bit is 0, the low-speed on-chip oscillator stops oscillating when the WDSTBYON bit is 0. When the watchdog timer runs, the low-speed on-chip oscillator clock does not stop running even if the program is out of control.

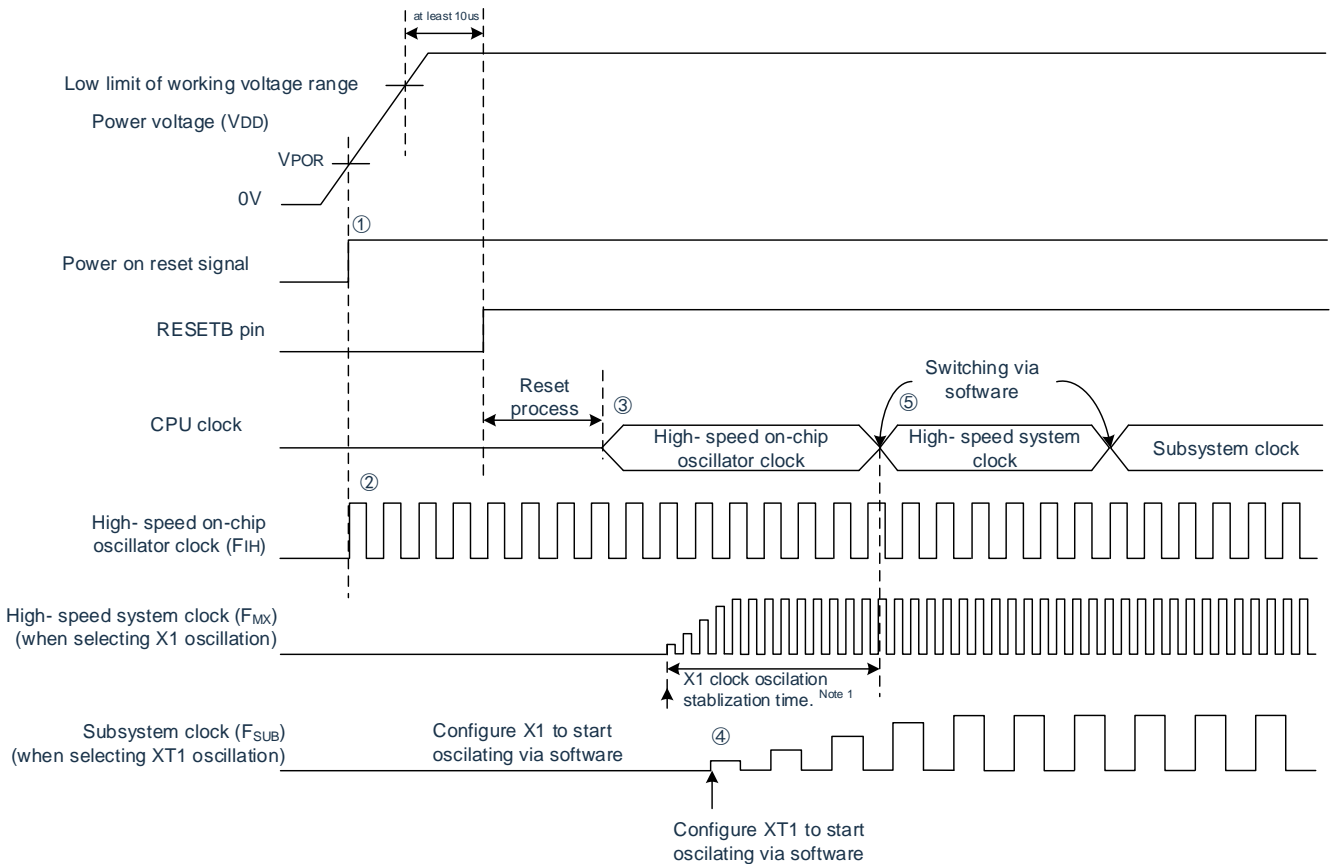
4.5 Operation of clock generation circuit

The clock generator generates the following clocks and controls the operation modes of the CPU, such as standby mode (refer to Figure 4-1).

- Main system clock f_{MAIN}
 - High speed system clock f_{MX}
 - X1 clock f_{X}
 - External main system clock f_{EX}
 - High speed internal oscillator clock f_{IH}
- Sub-system clock f_{SUB}
 - XT1 clock f_{XT}
 - External sub-system clock f_{EXS}
- Low-speed internal oscillator clock f_{IL}
- CPU/peripheral hardware clock f_{CLK}

After the BAT32G135 is released from reset, the CPU starts operation through the output of the high-speed on-chip oscillator. The operation of the clock generating circuit when the power is turned on is as shown in Figure 4-16.

Figure 4-16 Operation of clock generation circuit when 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).

Notice 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, 2MHz and 1MHz 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

Option byte (000C2H)	7	6	5	4	3	2	1	0
	1	1	1	FRQSEL4 0/1	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}
1	1	0	0	0	64MHz	64MHz
1	0	0	0	0	48MHz	48MHz
0	1	0	0	0	32MHz	32MHz
0	0	0	0	0	24MHz	24MHz
0	1	0	0	1	32MHz	16MHz
0	0	0	0	1	24MHz	12MHz
0	1	0	1	0	32MHz	8MHz
0	0	0	1	0	24MHz	6MHz
0	1	0	1	1	32MHz	4MHz
0	0	0	1	1	24MHz	3MHz
0	1	1	0	0	32MHz	2MHz
0	1	1	0	1	32MHz	1MHz
Other than 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=0		FRQSEL4=1	
			FRQSEL3=0	FRQSEL3=1	FRQSEL3=0	FRQSEL3=1
0	0	0	$f_{IH}=24\text{MHz}$ $f_{HOCO}=24\text{MHz}$	$f_{IH}=32\text{MHz}$ $f_{HOCO}=32\text{MHz}$	$f_{IH}=48\text{MHz}$ $f_{HOCO}=48\text{MHz}$	$f_{IH}=64\text{MHz}$ $f_{HOCO}=64\text{MHz}$
0	0	1	$f_{IH}=12\text{MHz}$ $f_{HOCO}=24\text{MHz}$	$f_{IH}=16\text{MHz}$ $f_{HOCO}=32\text{MHz}$	$f_{IH}=24\text{MHz}$ $f_{HOCO}=48\text{MHz}$	$f_{IH}=32\text{MHz}$ $f_{HOCO}=64\text{MHz}$
0	1	0	$f_{IH}=6\text{MHz}$ $f_{HOCO}=24\text{MHz}$	$f_{IH}=8\text{MHz}$ $f_{HOCO}=32\text{MHz}$	$f_{IH}=12\text{MHz}$ $f_{HOCO}=48\text{MHz}$	$f_{IH}=16\text{MHz}$ $f_{HOCO}=64\text{MHz}$
0	1	1	$f_{IH}=3\text{MHz}$ $f_{HOCO}=24\text{MHz}$	$f_{IH}=4\text{MHz}$ $f_{HOCO}=32\text{MHz}$	$f_{IH}=6\text{MHz}$ $f_{HOCO}=48\text{MHz}$	$f_{IH}=8\text{MHz}$ $f_{HOCO}=64\text{MHz}$
1	0	0	Prohibited settings.	$f_{IH}=2\text{MHz}$ $f_{HOCO}=32\text{MHz}$	$f_{IH}=3\text{MHz}$ $f_{HOCO}=48\text{MHz}$	$f_{IH}=4\text{MHz}$ $f_{HOCO}=64\text{MHz}$
1	0	1	Prohibited settings.	$f_{IH}=1\text{MHz}$ $f_{HOCO}=32\text{MHz}$	Prohibited settings.	$f_{IH}=2\text{MHz}$ $f_{HOCO}=64\text{MHz}$
Other than 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", and when f_x is greater than or equal to 10MHz, set the AMPH bit to "1" to operate the X1 oscillator circuit.

	7	6	5	4	3	2	1	0
CMC	EXCLK	OSCSEL	EXCLKS	OSCSELS		AMPHS1	AMPHS0	AMPH
	0	1	0	0	0	0	0	0/1

- ② 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						OSTS2	OSTS1	OSTS0
	0	0	0	0	0	0	1	0

- 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	MSTOP	XTSTOP						HIOSTOP
	0	1	0	0	0	0	0	0

- 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	MOST8	MOST9	MOST10	MOST11	MOST13	MOST15	MOST17	MOST18
	1	1	1	0	0	0	0	0

- 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	CLS	CSS	MCS	MCM0				
	0	0	0	1	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 (ultra-low consumption current).

	7	6	5	4	3	2	1	0
OSMC	RTCLPC 0/1	0	0	WUTMMCK0 0	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	EXCLK 0	OSCSEL 0	EXCLKS 0	OSCSELS 1	0	AMPHS1 0/1	AMPHS0 0/1	AMPH 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	MSTOP 1	XTSTOP 0	0	0	0	0	0	HIOSTOP 0

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

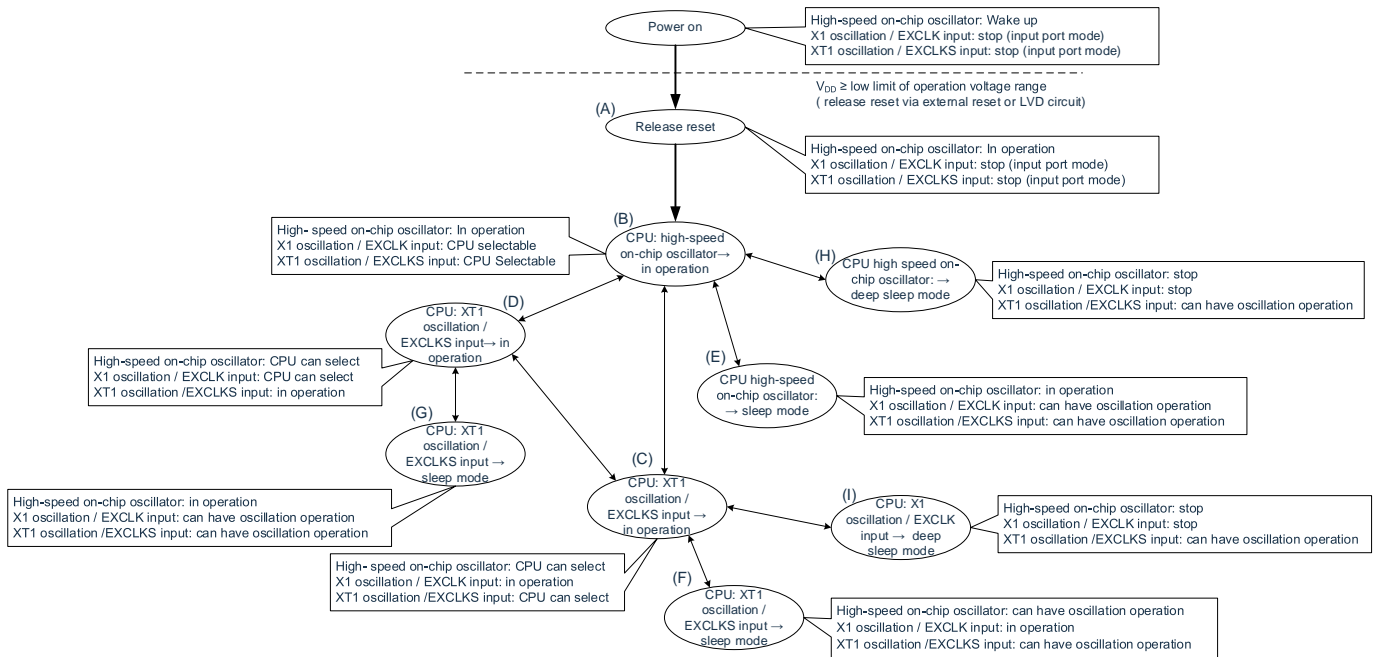
- ⑤ Set the XT1 oscillating clock to the CPU/peripheral hardware clock via the CSS bit of the CKC register.

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

4.6.4 CPU clock status transition diagram

Figure 4-17 shows the CPU clock status transition diagram of this product.

Figure 4-17 State transfer diagram for CPU clock



Examples of CPU clock transfer and SFR register setting are in Table 4-3.

Table 4-3 CPU clock transition and SFR register setting examples (1/5)

(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 ^{Note1}			OSTS register	CSC register	OSTC register	CKC register
	EXCLK	OSCSEL	AMPH		MSTOP		MCM0
(A)→(B)→(C) (X1 clock: $1\text{MHz} \leq f_x \leq 10\text{MHz}$)	0	1	0	Note2	0	Need to confirm	1
(A)→(B)→(C) (X1 clock: $10\text{MHz} < f_x \leq 20\text{MHz}$)	0	1	1	Note2	0	Need to confirm	1
(A)→(B)→(C) (External main system clock)	1	1	×	Note2	0	No need to confirm	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.

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) \leq 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).

(3) CPU operating with subsystem clock (D) after reset release (A)

(The CPU operates with the high-speed on-chip oscillator clock immediately after a reset is released (B))

(SFR register setting order) →

SFR register setting flag State transition	CMC register ^{Note}				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 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.

Remark1. ×: Ignore

2. (A) to (I) of Table 4-3 correspond to (A) to (I) of Figure 4-17.

Table 4-3 CPU clock transition and SFR register setting examples (2/5)

(4) CPU clock changing from high-speed on-chip oscillator clock (B) to high-speed system clock (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	AMPH		MSTOP		MCM0
(B)→(C) (X1 clock: $1\text{MHz} \leq f_x \leq 10\text{MHz}$)	0	1	0	Note2	0	Need to confirm	1
(B)→(C) (X1 clock: $10\text{MHz} < f_x \leq 20\text{MHz}$)	0	1	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.

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. Not required if already set.

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) \leq 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).

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

(SFR register setting order) →

SFR register setting flag State transition	CMC register ^{Note}			CSC register	Oscillation stabilization	CKC register
	EXCLKS	OSCSELS	AMPHS1, 0	XTSTOP	waiting	CSS
(B)→(D) (XT1 clock)	0	1	00: Low power oscillation 01: Normal oscillation 10: Ultra-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.

Note 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. Not required if already set.

Remark1. ×: Ignore

2. (A) to (I) of Table 4-3 correspond to (A) to (I) of Figure 4-17.

Table 4-3 CPU clock transition and SFR register setting examples (3/5)

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

(SFR register setting order) →

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

Not required for high-speed on-chip oscillator clock

Remark The oscillation accuracy of the high-speed on-chip oscillator clock stabilization wait time varies depending on temperature conditions and during deep sleep mode.

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

(SFR register setting order) →

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

Not required for subsystem clock operation.

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

(SFR register setting order) →

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

Not required for high-speed system clock operation.

Remark1. (A) to (I) of Table 4-3 correspond to (A) to (I) of Figure 4-17.

2. 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 4-3 CPU clock transition and SFR register setting examples (4/5)

(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: $1\text{MHz} \leq f_x \leq 10\text{MHz}$)	Note	0	Need to confirm	0
(D)→(C) (X1 clock: $10\text{MHz} < f_x \leq 20\text{MHz}$)	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) \leq 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 data sheet for electrical characteristics).

- (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.

Remark (A) to (I) of Table 4-3 correspond to (A) to (I) of Figure 4-17.

Table 4-3 CPU clock transition and SFR register setting examples (5/5)

- (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) →

State transition		Setting contents		
(B)→(H)		Stop	—	Bit2 of the SCR register (SLEEPDEEP) is set to 1 and the WFI instruction is executed.
(C)→(I)	X1 oscillation	Peripheral functions that cannot run in deep sleep mode.	Set OSTS register	
	External clock		—	

Remark (A) to (I) of Table 4-3 correspond to (A) to (I) of Figure 4-17.

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 4-4 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-chip 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 4-4 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 oscillator clock is used as the main system clock.	It can stop the XT1 oscillation (XTSTOP=1).
	Clock	<ul style="list-style-type: none"> • HIOSTOP=0, MCS=0 	
	X1 clock	X1 oscillation stabilization and select high-speed system clock as the main system clock.	
		Clock.	
		<ul style="list-style-type: none"> • 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. <ul style="list-style-type: none"> • OSCSEL=1, EXCLK=1, MSTOP=0 • MCS=1 		
External subsystem clock	Cannot transfer	-	
External subsystem clock	High-speed on-chip oscillator	High-speed on-chip oscillator is oscillating and selecting high-speed on-chip oscillator clock is used as the main system clock.	Able to set the input of the external subsystem clock invalid
	Clock	<ul style="list-style-type: none"> • HIOSTOP=0, MCS=0 	(XTSTOP=1).
	X1 clock	X1 oscillation stabilization and select high-speed system clock as the main system clock.	
		Clock.	
		<ul style="list-style-type: none"> • 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. <ul style="list-style-type: none"> • OSCSEL=1, EXCLK=1, MSTOP=0 • MCS=1 		
XT1 clock	Cannot 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 4-5~Table 4-7).

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-chip oscillator clock.

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

Table 4-5 Maximum time required to switch main system clock

Clock A	Switch direction	Clock B	Remark
f_{IH}	↔	f_{MX}	Refer to Table 4-6.
f_{MAIN}	↔	f_{SUB}	Refer to Table 4-7.

Table 4-6 Maximum number of clocks required for $F_{IH} \leftrightarrow F_{MX}$

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 4-7 Maximum number of clocks required for $F_{MAIN} \leftrightarrow F_{SUB}$

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	

Remark1. The number of clocks listed in Table 4-6 and Table 4-7 is the number of CPU clocks before switchover.

2. Calculate the number of clocks in Table 4-6 and Table 4-7 by removing 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 $F_{IH}=8\text{MHz}$, $F_{MX}=10\text{MHz}$)

$$2f_{MX}/f_{IH}=2(10/8)=2.5 \rightarrow 3 \text{ clocks}$$

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 4-8 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	(MCS=0 or CLS=1)	MSTOP=1
External main system clock	(CPU runs at a clock other than the high-speed system clock)	
XT1 clock	CLS=0 (CPU runs at a clock other than the subsystem clock)	XTSTOP=1

Chapter 5 General-Purpose Timer Unit Timer4

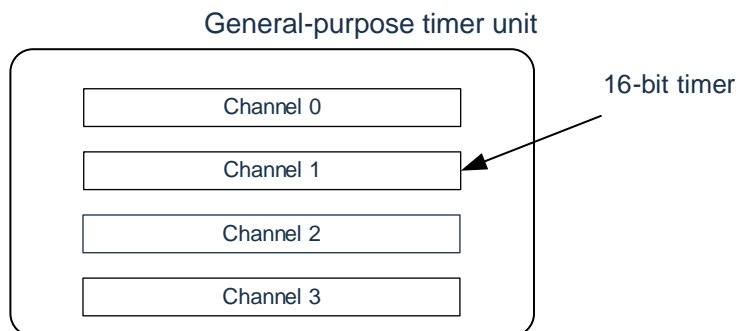
This product is equipped with two general-purpose timer units, each containing four channels.

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).
3. The following section of this chapter focuses on 48-pin products.

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
<ul style="list-style-type: none"> • 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) 	<ul style="list-style-type: none"> • 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

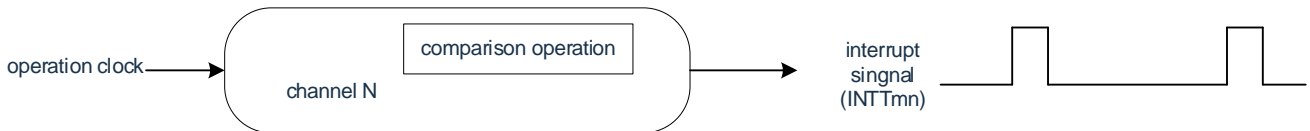
The general-purpose timer unit has the following functions.

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.

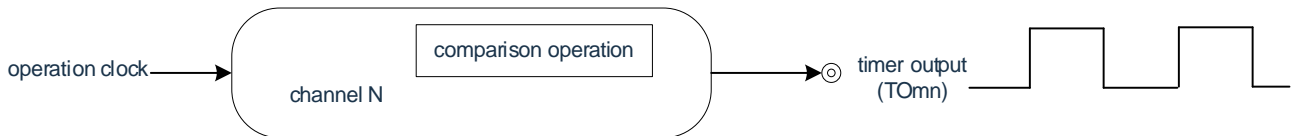
(1) Interval timer

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



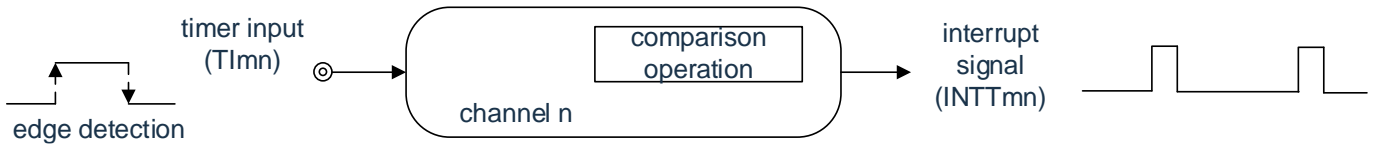
(2) Square wave output

A toggle operation is performed each time INTTmn 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.



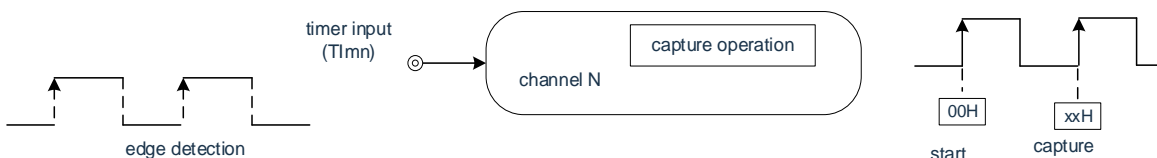
(4) Frequency divider function (channel 0 of unit 0 only)

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.



Remark1. m: unit number (m=0, 1) n: channel number (n=0~3)

2. Please refer to "Chapter 2 Pin Functions" for the configurable timer input/output pins of channel 0~3.

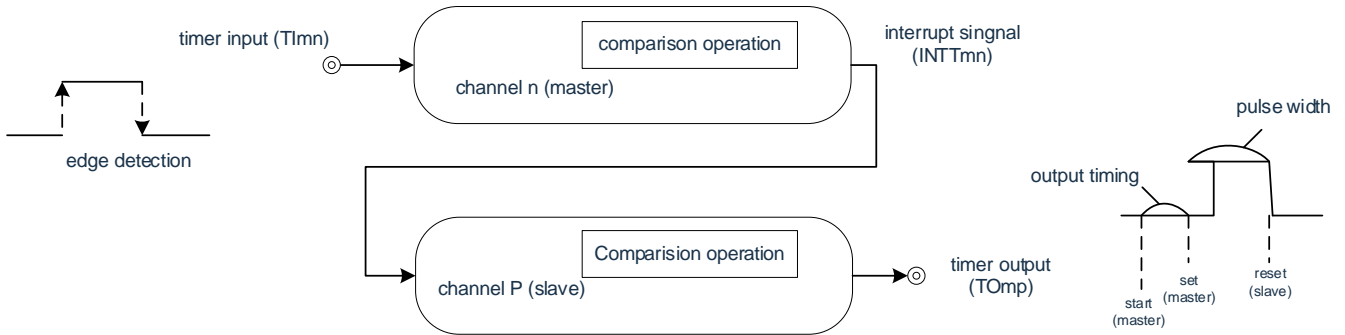
5.1.2 Multi-channel linkage operation functions

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

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

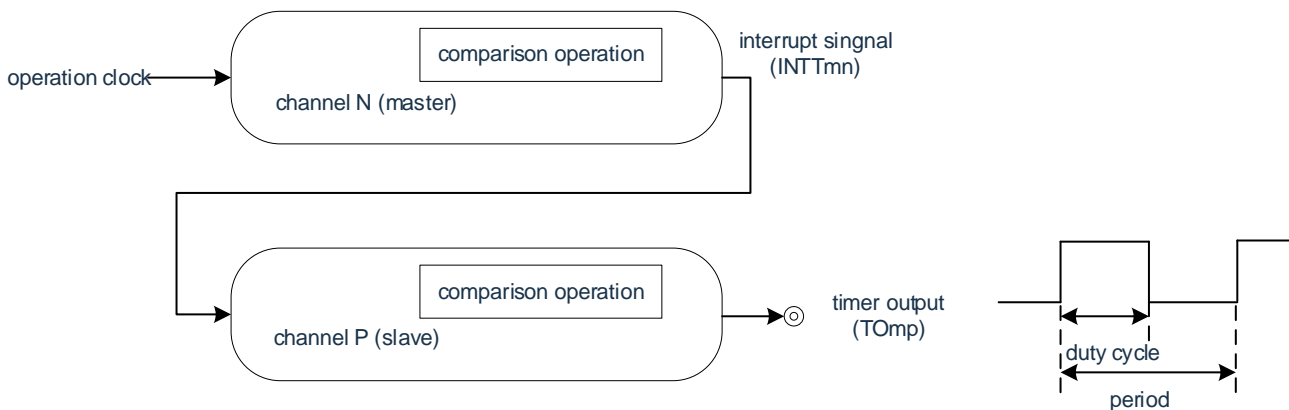
(1) Single trigger pulse output

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



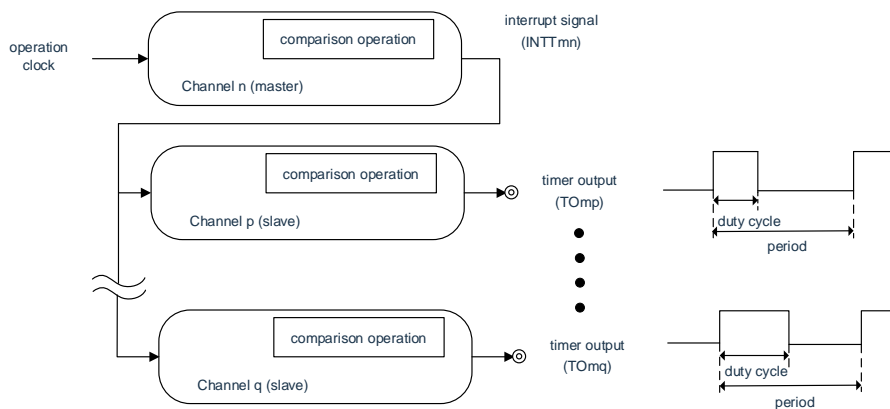
(2) PWM (Pulse Width Modulation) output

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+3 PWM signals of any duty cycle with a fixed period using one master channel and multiple slave channels.



Notice Please refer to “5.4.1 Basic rules of multi-channel linkage operation function” for the rule details of multi-channel linkage operation function.

Remark 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.

Notice There are several rules for using 8-bit timer operation function.

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> <ul style="list-style-type: none"> • 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> <ul style="list-style-type: none"> • 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 Pin Functions”.

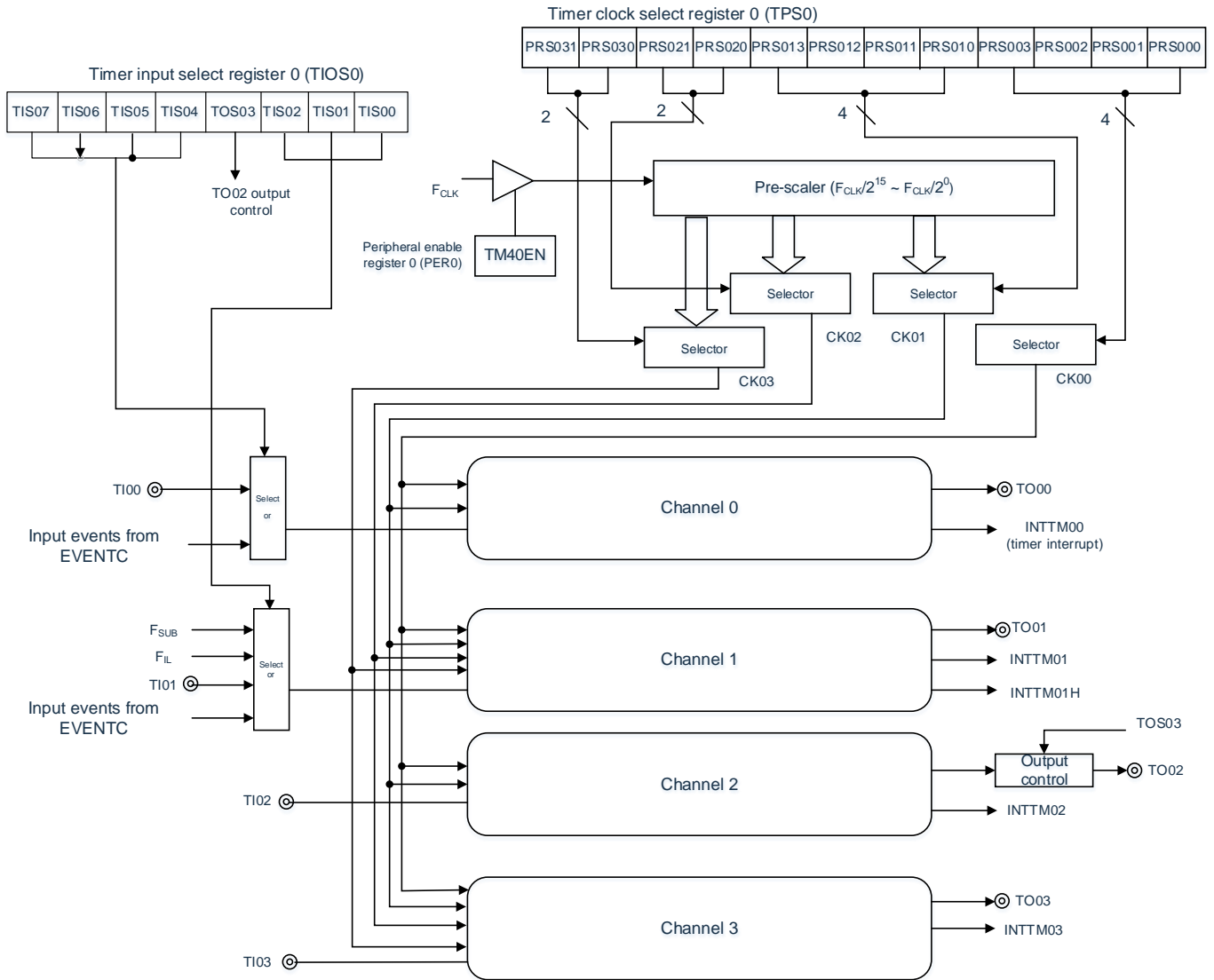
Note 2: Only for channel selection of unit 0.

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

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

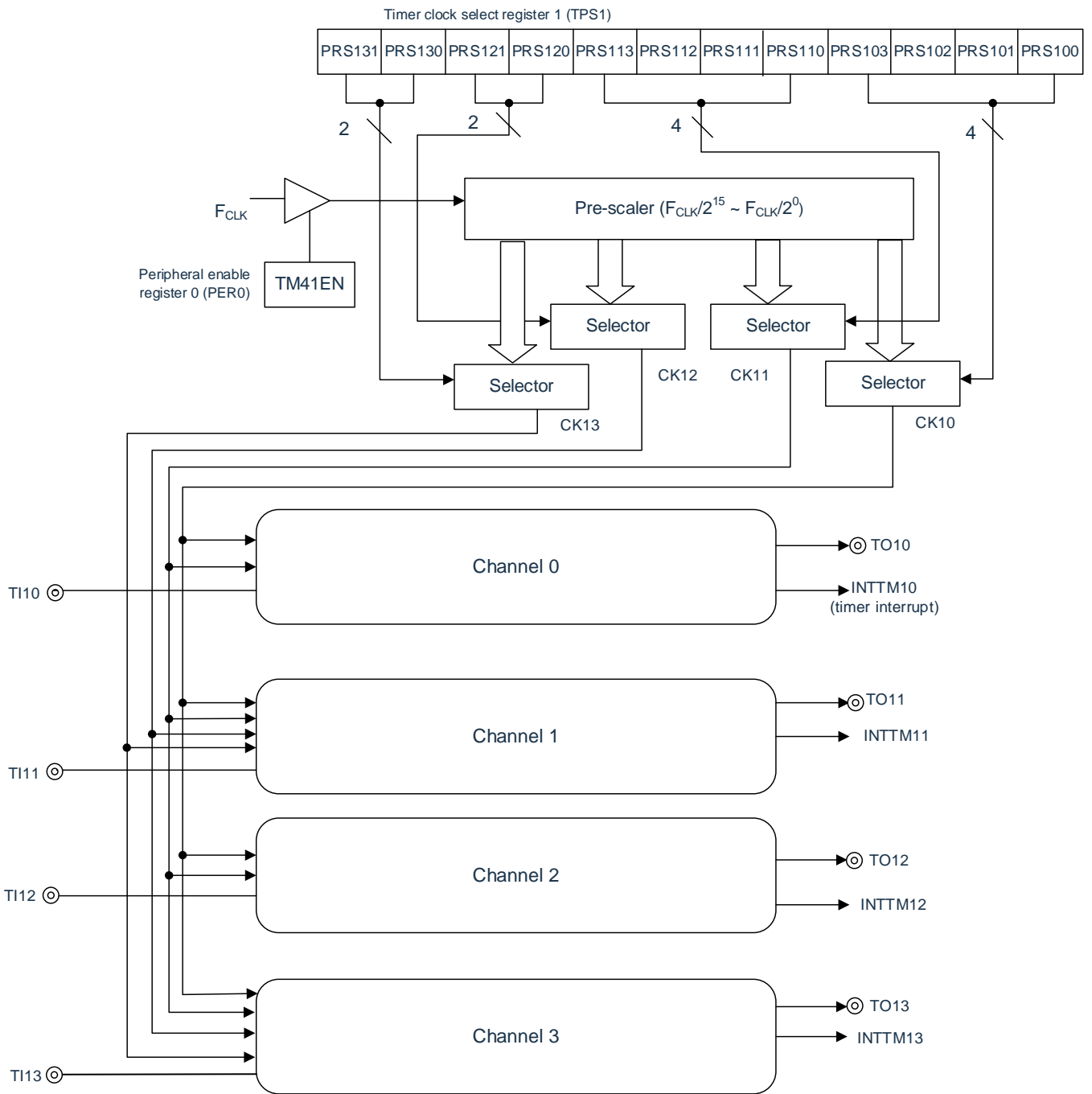
The block diagram of the general-purpose timer unit is shown in Figure 5-1.

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

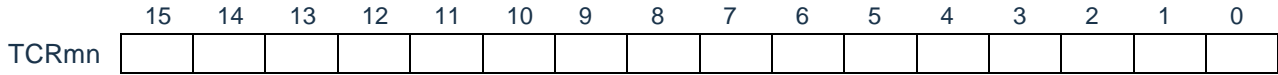
Offset address	Register name	R/W	Bit width	Reset value
0x180	TCR10	R	16	FFFFH
0x182	TCR11	R	16	FFFFH
0x184	TCR12	R	16	FFFFH
0x186	TCR13	R	16	FFFFH
0x190	TMR10	R/W	16	0000H
0x192	TMR11	R/W	16	0000H
0x194	TMR12	R/W	16	0000H
0x196	TMR13	R/W	16	0000H
0x1A0	TSR10	R	16	0000H
0x1A0	TSR10L	R	8	00H
0x1A2	TSR11	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)”).

Figure 5-3 Table of timer count register mn (TCRmn)



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

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

In the following cases, the count value becomes “FFFFH”.

- When a reset signal is generated
- When clearing the TM4mEN bit of the peripheral enable register 0 (PER0)
- At the end of the count of the slave channel in PWM output mode
- At the end of the count of the slave channel in delayed count mode
- At the end of counting of master/slave channels in single trigger pulse output mode
- 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”.

- When input starts triggering in capture mode
- At the end of the capture in capture mode

Notice Even if the TCRmn register is read, the count value is not captured to the timer data register mn (TDRmn).

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

Table 5-2 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 ^{Note}			
		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 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.

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

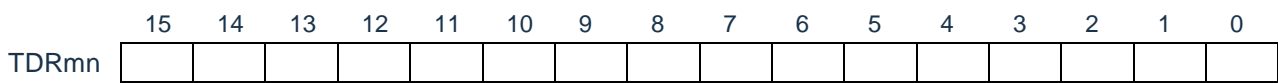
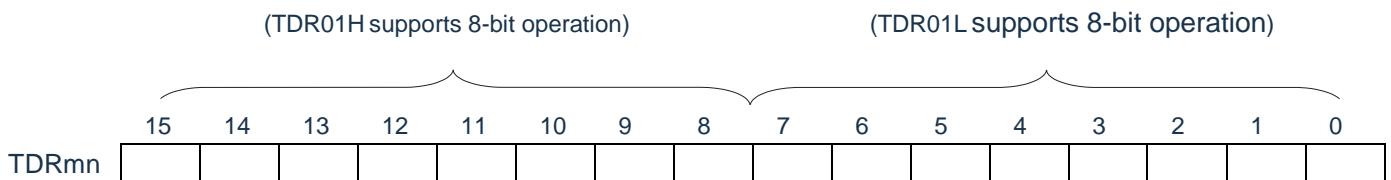


Figure 5-5 Table of timer data registers mn (TDRmn) (n=1, 3)



- (i) When timer data register mn (TDRmn) is used as compare register
Counting down is started from the value set to the TDRmn register. When the count value reaches 0000H, an interrupt signal (INTTMmn) is generated. The TDRmn register holds its value until it is rewritten.

Notice Even if a capture trigger signal is input, the TDRmn register set to the compare function does not perform capture operation.

- (ii) When timer data register mn (TDRmn) is used as capture register
The count value of timer count register mn (TCRmn) is captured to the TDRmn register when the capture trigger is input.
A valid edge of the TImn pin can be selected as the capture trigger. This selection is made by timer mode register mn (TMRmn).

Remark m: unit number (m=0, 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 configuration register (PxxCFG)

Notice The assigned registers and bits vary from product to product. The initial value 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".

Figure 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	IRDAEN	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. <ul style="list-style-type: none"> • 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. <ul style="list-style-type: none"> • 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. <ul style="list-style-type: none"> • 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. <ul style="list-style-type: none"> • The SFR used by general-purpose timer unit 1 can read and write.

Notice1. 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.

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).

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).

If the PRSm20 and PRSm21 bits can be rewritten (n = 1, 3):

All channels for which CKm1 is selected as the operation clock (CKSmn1, CKSmn0 = 1, 0) are stopped (TEmn = 0).

If the PRSm30 and PRSm31 bits can be rewritten (n = 1, 3):

All channels for which CKm3 is selected as the operation clock (CKSmn1, CKSmn0 = 1, 1) are stopped (TEmn = 0).

The TPSm register can be set by a 16-bit memory manipulation instruction. Reset signal generation clears this register to 0000H.

Figure 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	0	0	PRS m31	PRS m30	0	0	PRS m21	PRS m20	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00

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 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.

Notice1. Bit 15, 14, 11 and 10 must be set to "0".

- 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.

Remark1. f_{CLK} : CPU/peripheral hardware clock frequency

- 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})".

Figure 5-8 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	0	0	PRS m31	PRS m30	0	0	PRS m21	PRS m20	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00

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	$f_{CLK}/2^{14}$

Note 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.

Notice Bits 15, 14, 11, 10 must be set to "0".

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 5-3 Interval time that can be set by operation clocks CKSm2 and CKSm3

Clock		Interval time ^{Note} ($F_{CLK}=32\text{MHz}$)			
		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 ○ The margin is within 5%.

Remark 1. f_{CLK} : CPU/peripheral hardware clock frequency

2. 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 (FMCK), 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”.

Notice Bit11 of the TMRmn register varies from channel to channel.

TMRm2 : MASTERmn bit (n=2)
TMRm1, TMRm3: SPLITmn bit (n=1, 3)
TMRm0 : Fixed to “0”.

Figure 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	0	MD mn3	MD mn2	MD mn1	MD mn0

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

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

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

- 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}).

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

Figure 5-10 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	0	MD mn3	MD mn2	MD mn1	MD mn0

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

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

(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)

Figure 5-11 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	0	MD mn3	MD mn2	MD mn1	MD mn0

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

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

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)

Figure 5-12 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	0	MD mn3	MD mn2	MD mn1	MD mn0
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn (n=1,3)	CKS mn1	CKS mn0	0	CCS mn	SPLIT mn	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0	0	0	MD mn3	MD mn2	MD mn1	MD mn0
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn (n=0)	CKS mn1	CKS mn0	0	CCS mn	0 ^{Note 1}	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0	0	0	MD mn3	MD mn2	MD mn1	MD mn0

MD mn3	MD mn2	MD mn1	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
<ul style="list-style-type: none"> 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).
<ul style="list-style-type: none"> Event counter mode (0, 1, 1) 	0	No timer interrupt is generated when counting starts (the output of the timer does not change).
<ul style="list-style-type: none"> 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 valid ^{Note 3} . No interruption at this time.
<ul style="list-style-type: none"> 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.

2: In single count mode, the interrupt output (INTTMmn) and T0mn output at the start of counting are not controlled.

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 5-4 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".

Figure 5-13 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).	

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

Table 5-4 OVF bit change and set/clear conditions in each operation mode

Timer operation mode	OVF bit	Set/clear conditions
<ul style="list-style-type: none"> • Capture mode • Capture & single count mode 	Clear	No overflow occurred at the capture.
	Set	Overflow occurred at the capture.
<ul style="list-style-type: none"> • Interval timer mode • Event counter mode • Single count mode 	Clear	— (N/A)
	Set	

Remark 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 (TE_m)

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

Each of the TE_m 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 TE_m register is "1". If each bit of the TTM register is "1", the corresponding bit of the TE_m register is cleared to "0".

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

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

Figure 5-14 Table of timer channel enable status register m (TE_m)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TE _m	0	0	0	0	TEH _{m3}	0	TEH _{m1}	0	0	0	0	0	TE _{m3}	TE _{m2}	TE _{m1}	TE _{m0}

TEH _{m3}	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.

TEH _{m1}	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.

TE _m _n	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 TE _{m1} and TE _{m3} 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 (T_{Sm})

The T_{Sm} register is a trigger register that initializes the timer counter register m_n (TCR_{mn}) 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 (TE_m) is set to "1". Since the T_{Smn} bit, the TSH_{m1} bit and the TSH_{m3} bit are trigger bits, the T_{Smn} bit, the TSH_{m1} bit and the TSH_{m3} bit are cleared immediately if the operation enable state is changed (TE_{mn}, TEH_{m1}, TEH_{m3} = 1).

The T_{Sm} register is set by a 16-bit memory manipulation instruction.

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

Figure 5-15 Table of timer channel start register m (T_{Sm})

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T _{Sm}	0	0	0	0	TSH _{m3}	0	TSH _{m1}	0	0	0	0	0	T _{Sm} ₃	T _{Sm} ₂	T _{Sm} ₁	T _{Sm} ₀

TSH _{m3}	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 TEH _{m3} bit to "1" to enter the counting enable state. If the counting of the TCR _{m3} register is started in the count enable state, the interval timer mode is entered (refer to Table 5-5 of "5.5.2 Start Timing of Counter").

TSH _{m1}	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 TEH _{m1} bit to "1" to enter the counting enable state. If counting in the TCR _{m1} register is started in the count enable state, the interval timer mode is entered (refer to Table 5-5 of "5.5.2 Start Timing of Counter").

T _{Smn}	Operation enable (start) trigger of channel n
0	No trigger operation.
1	Set the TE _{mn} bit to "1" to enter the counting enable state. The start of counting in the TCR _{mn} register in the count enable state varies with each operation mode (refer to Table 5-5 of "5.5.2 Start Timing of Counter"). When channel 1 and channel 3 are in 8-bit timer mode, T _{Sm1} and T _{Sm3} are operation enable (start) triggers for the lower 8-bit timer.

Notice1. Bits 15~12, 10, 8~4 must be set to "0".

- When switching from a function that does not use T_{Imn} pin input to a function that uses T_{Imn} pin input, the following period of waiting is required from setting the timer mode register m_n (TMR_{mn}) until the T_{Smn} bit is set to "1":

When the T_{Imn} pin noise filter is valid (TNFEN_{mn}=1): 4 operating clocks (F_{MCK})

When the T_{Imn} pin noise filter is invalid (TNFEN_{mn}=0): 2 operating clocks (F_{MCK})

Remark 1. The T_{Sm} register always reads "0".

- 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".

Figure 5-16 Table of timer channel stop register m (TTm)

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".

Figure 5-17 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".

Figure 5-18 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	TOE m3	TOE m2	TOE m1	TOE m0

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".

Figure 5-19 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	TOm 3	TOm 2	TOm 1	TOm 0

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: Bits 15~4 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".

Figure 5-20 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	TOL m3	TOL m2	TOL m1	0

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

Notice Bits 15~4 and bit0 must be set to "0".

Remark 1. 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.

2. 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 resetting 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".

Figure 5-21 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	TOM m3	TOM m2	TOM m1	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".

Note For details, refer to "5.5.1(2) When valid edge of TImn pin input signal is selected (CCSmn=1)", "5.5.2 Start timing of counter", and "5.7 Control of timer input (TImn)".

Figure 5-22 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 Refer to "Chapter 2 Pin Functions" 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".

Note For details, refer to "5.5.1(2) When 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)".

Figure 5-23 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 Refer to "Chapter 2 Pin Functions" 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 can be arbitrarily configured to each port except RESETB. For details, refer to “Chapter 2 Pin Functions”.

When multiplexing the output pin of Timer 0 to a port, the corresponding bit of the Port Mode Control Register (PMCxx), the bit of the Port Mode Register (PMxx), and the bit of the Port Register (Pxx) must be set to “0”.

When using the multiplexed port of the Timer 0 input pin as a timer input, the port mode control register (PMCxx) must be set to “0” and the port mode register (PMxx) must be set to “1”.

When multiplexing the output pin of Timer1 to a port, you must set the bit of the Port Mode Control Register (PMCxx) corresponding to that port, and the bit of the Port Mode Register (PMxx) to “0”. The port multiplexing function configuration register (PxxCFG) is also set. In this case, 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 configuration register P21CFG to “0x01”.

When using the multiplexed ports of the Timer1 input pins as timer inputs, you must set the Port Mode Register (PMxx) corresponding to each port to “1” and the Port Mode Control Register (PMCxx) to “0”. Set the port multiplexing function configuration register (TI10PCFG). At this point, the port register (Pxx) bit 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 configuration register TI10PCFG to “0x0b”.

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.

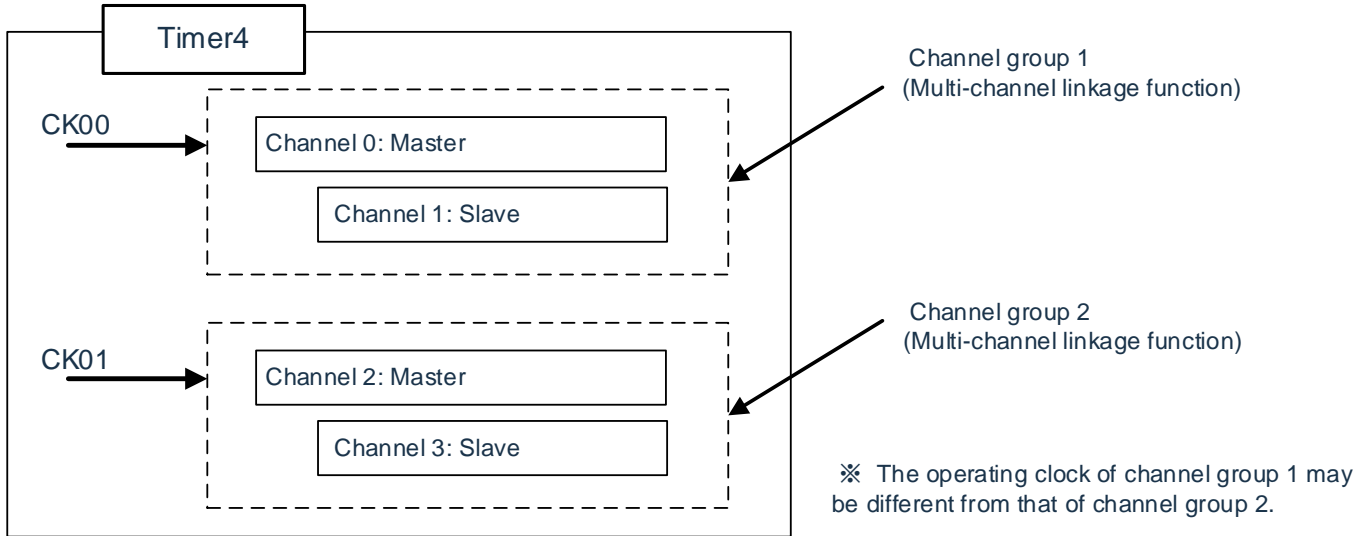
- 4) Multiple slave channels can be set for 1 master channel.
- 5) When multiple master channels are used, slave channels that span master channel cannot be set.
For example, when setting channel 0 and channel 2 as the master channel, channel 1 can be set as the slave channel of master channel 0, but 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).

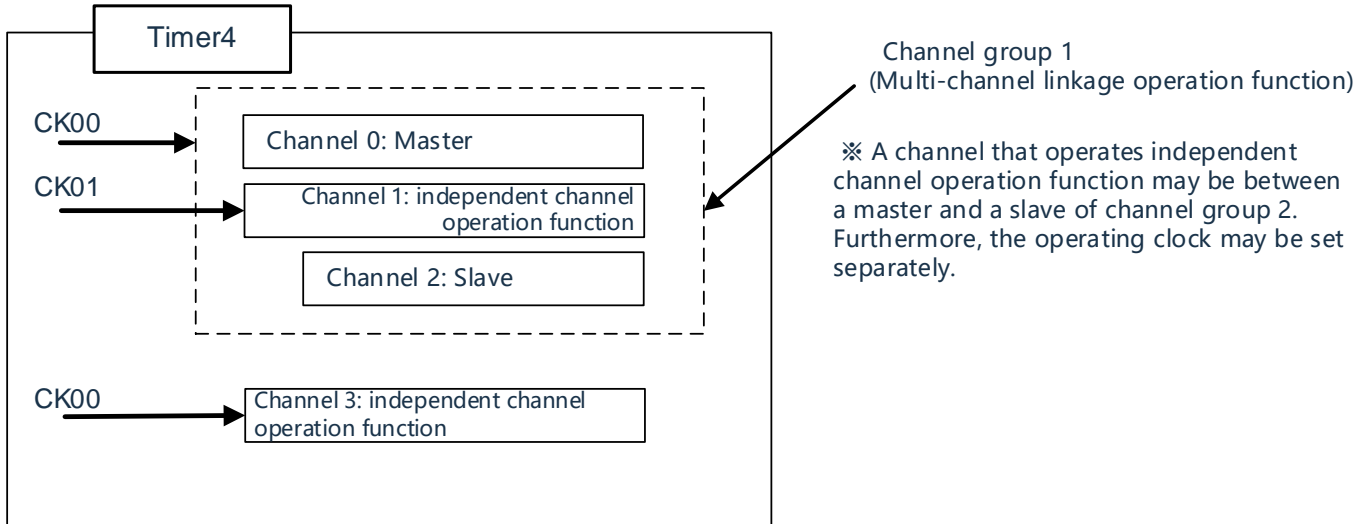
If you set 2 or more channel groups that are not linked to each other, the above basic rules do not apply to the channel groups.

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

Example 1:



Example 2:



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:
 - Interval timer function
 - External event counter function
 - Delay count function
- 8) For the lower 8 bits, the TSm1/TSm3 bit is manipulated to start channel operation and the TTm1/TTm3 bit is manipulated to stop channel operation. The channel status can be checked using the TEm1/TEm3 bit.
- 9) During 16-bit operation, manipulating the TSHm1, TSHm3, TTHm1, and TTHm3 bits is invalid. The TSm1, TSm3, TTm1, and TTm3 bits are manipulated to operate channels 1 and 3. The TEHm3 and TEHm1 bits are not changed.
- 10) For the 8-bit timer function, the 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:

- The CKSmn0 bit and CKSmn1 bit specified operation clock (F_{MCK})
- The active edge of the TImn pin input signal

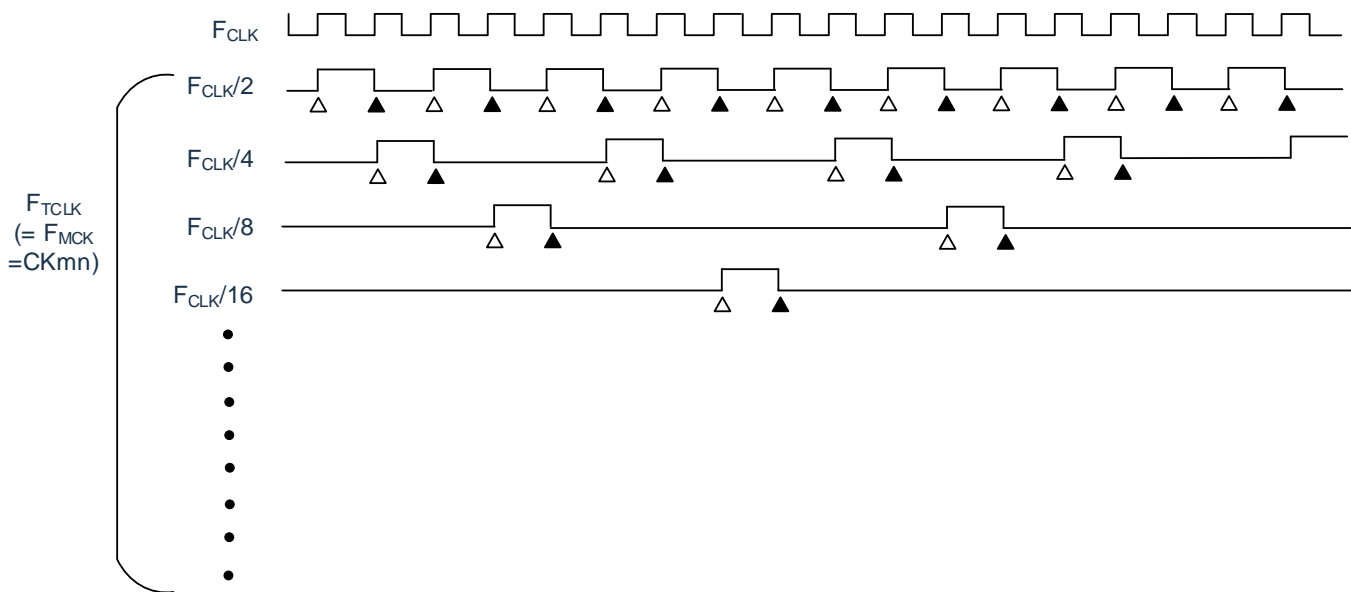
The general-purpose timer unit is designed to operate synchronously with F_{CLK} , so the timing of the count clock (F_{TCLK}) is as follows.

(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 $F_{CLK} \sim F_{CLK} / 2^{15}$. 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 5-24 Timing of F_{CLK} and count clock (F_{TCLK}) (When CCSmn = 0)



Remark: 1. Δ : Rising edge of the count clock

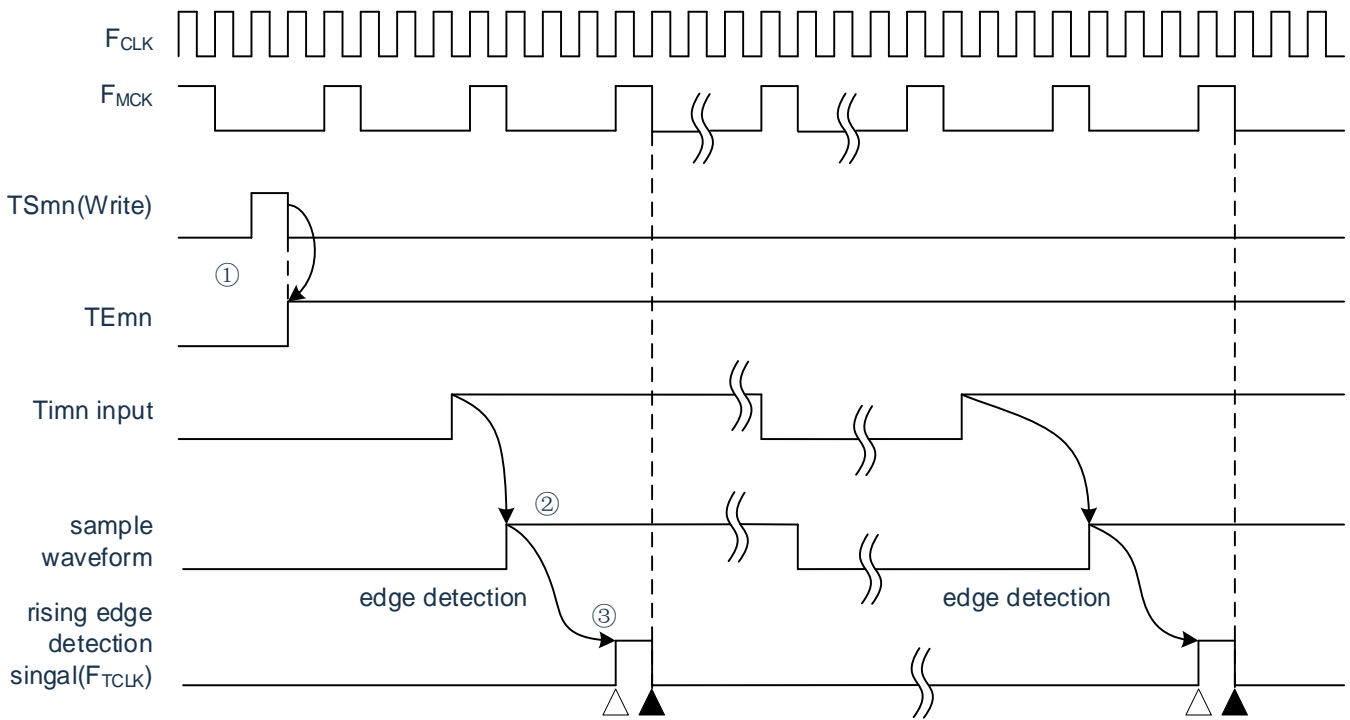
\blacktriangle : Synchronization, increment/decrement of counter

2. F_{CLK} : CPU/peripheral hardware clock

(2) When valid edge of input signal via the TImn pin is selected (CCSmn = 1)

The count clock (F_{TCLK}) 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 5-25 Timing of 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. Δ : Rising edge of the count clock
 \blacktriangle : Synchronization, increment/decrement of counter
- 2. F_{CLK} : CPU/peripheral hardware clock
 F_{MCK} : Operation clock of channel n
- 3. 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 5-5.

Table 5-5 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"
<ul style="list-style-type: none"> 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").
<ul style="list-style-type: none"> 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").
<ul style="list-style-type: none"> 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)").
<ul style="list-style-type: none"> 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").
<ul style="list-style-type: none"> 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)").

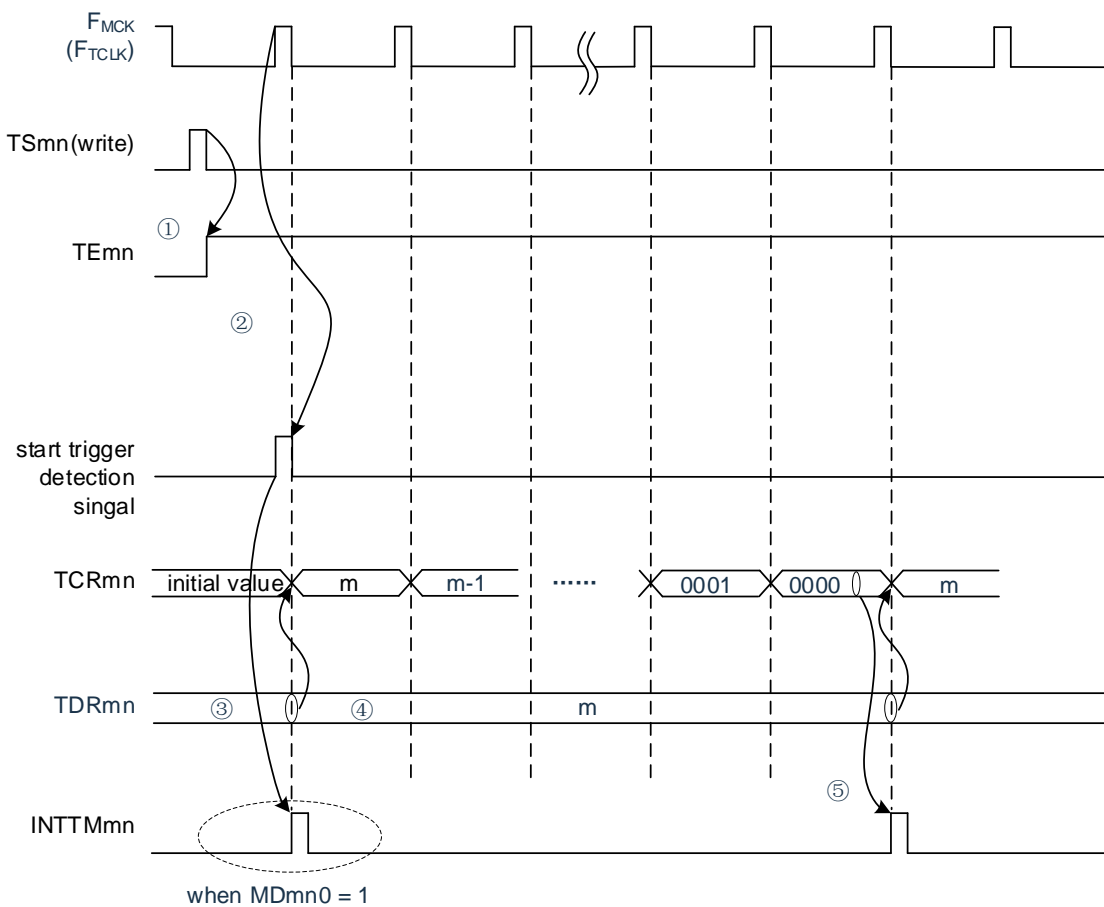
5.5.3 Operation of counter

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 5-26 Operation timing (interval timer mode)



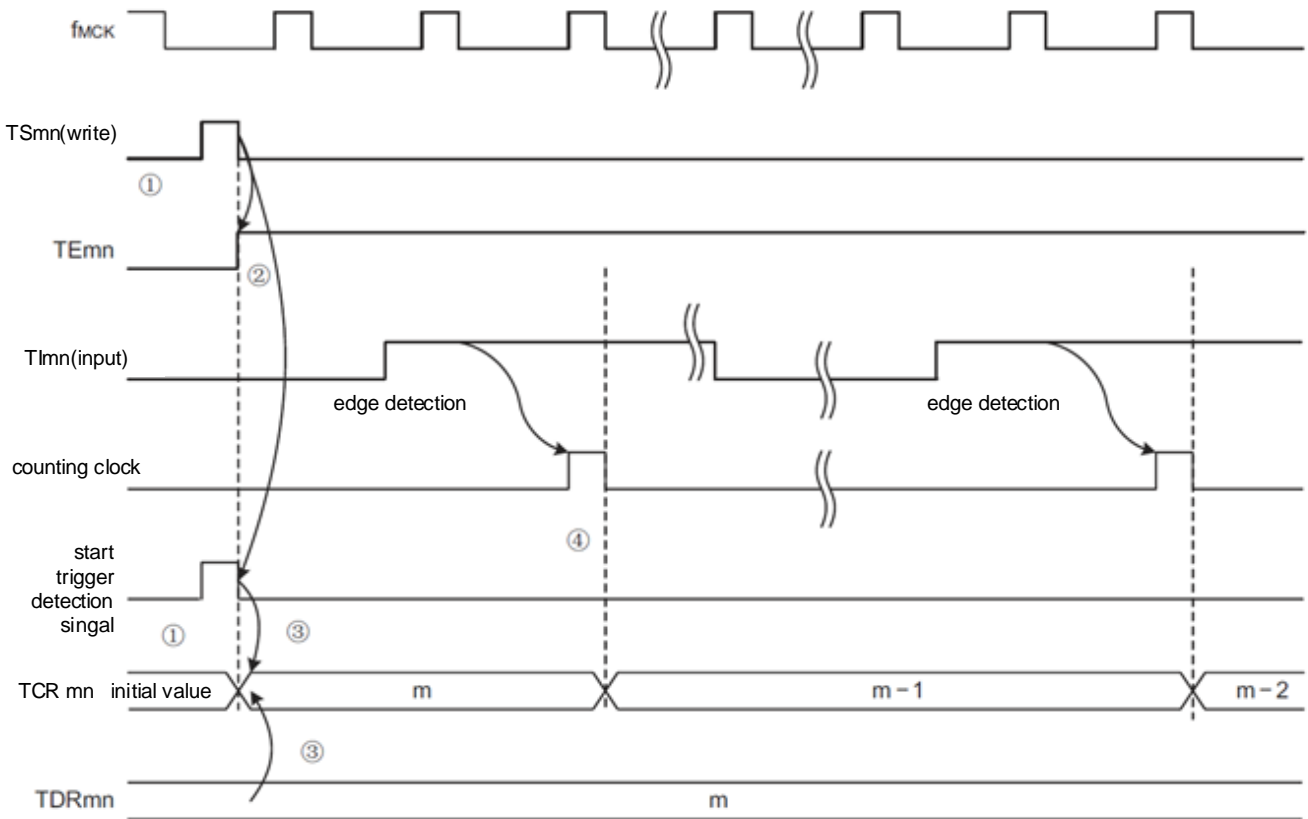
Notice In the first cycle operation of count clock after writing the TSmn bit, an error at a maximum of one clock is generated since count start delays until count clock has been generated. When the information on count start timing is necessary, an interrupt can be generated at count start by setting MDmn0 = 1.

Remark F_{MCK}, the start trigger detection signal, and INTTM_{mn} 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.
- ④ Thereafter, the value of the TCRmn register is counted decreasingly by the count clock at the active edge of the TImn input.

Figure 5-27 Operation timing (event counter mode)

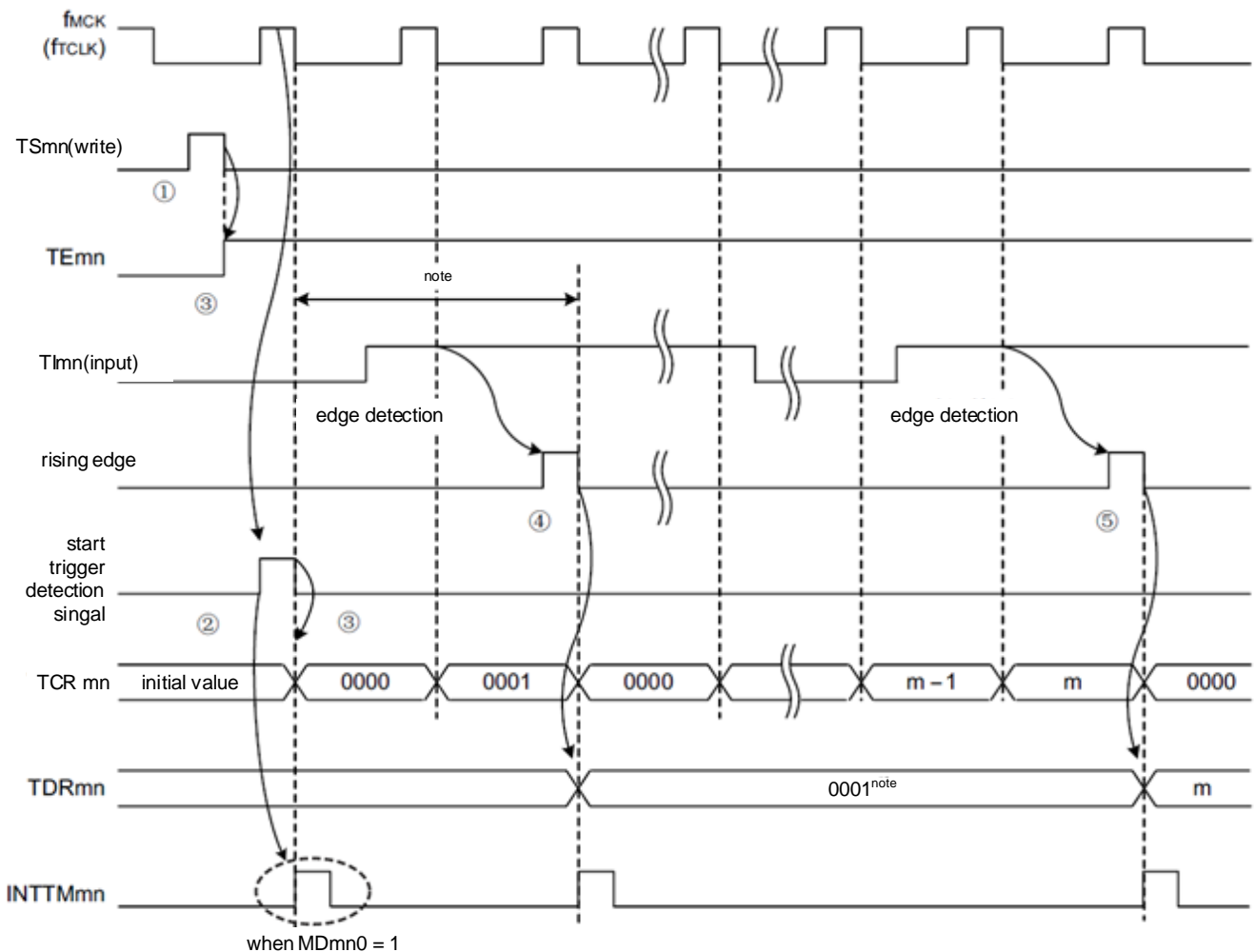


Remark This is a timing without the noise filter. If the noise filter is used, the edge detection is delayed by 2 more F_{MCK} cycles (3~4 cycles in total) from the $TImn$ input. The 1 cycle error is because the $TImn$ input is not synchronized with the count clock (F_{MCK}).

(1) 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 5-28 Operation timing (capture mode: interval measurement of input pulses)



Note 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.

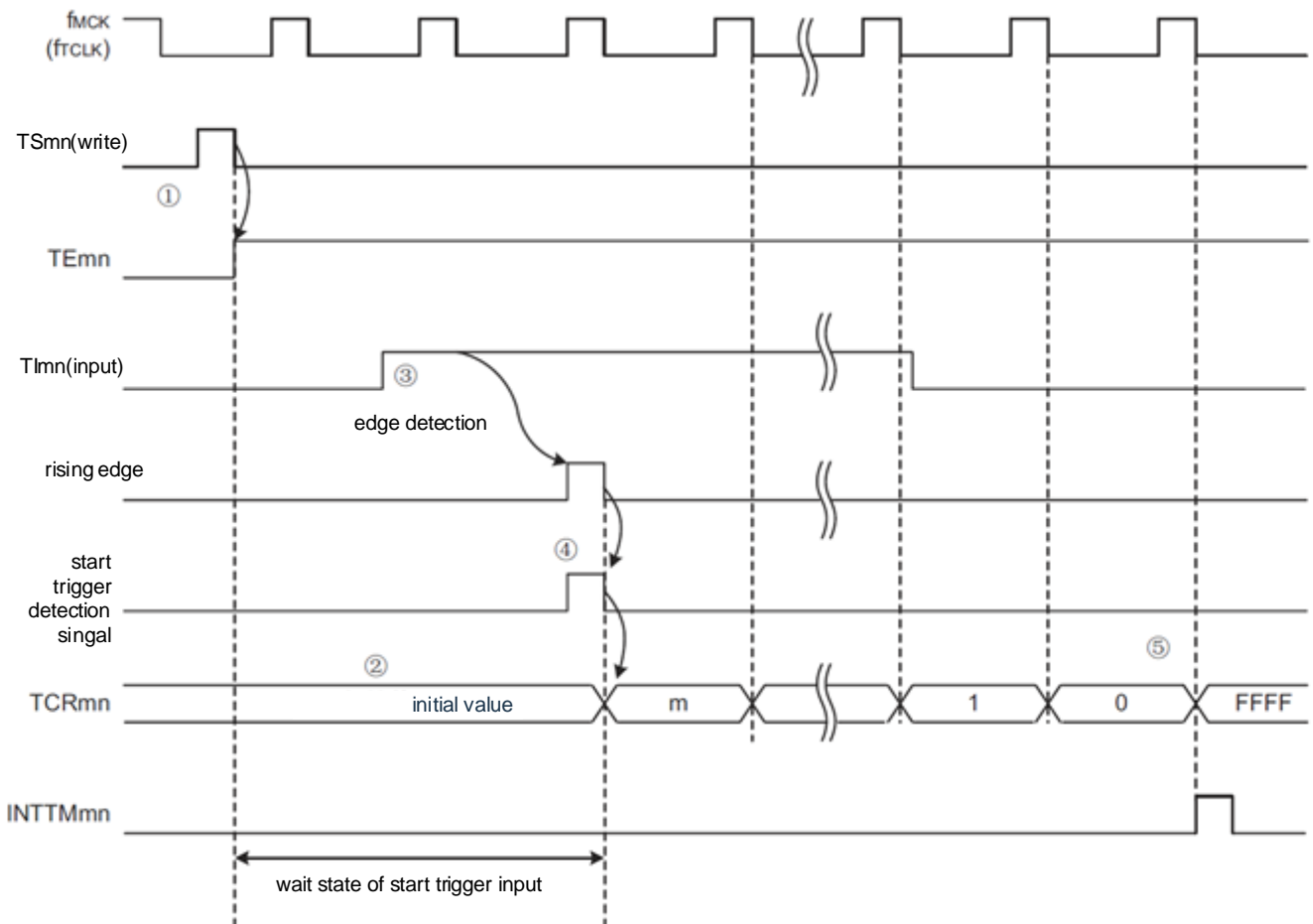
Notice Because the 1st count clock cycle runs after the TSmn bit is written and delays the start of counting before generating the count clock, an error of up to 1 clock cycle is generated. Also, if you need information about the start of the count timing, set MDmn0 to "1" so that an interrupt can be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

(2) 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 5-29 Operation timing (single count mode)

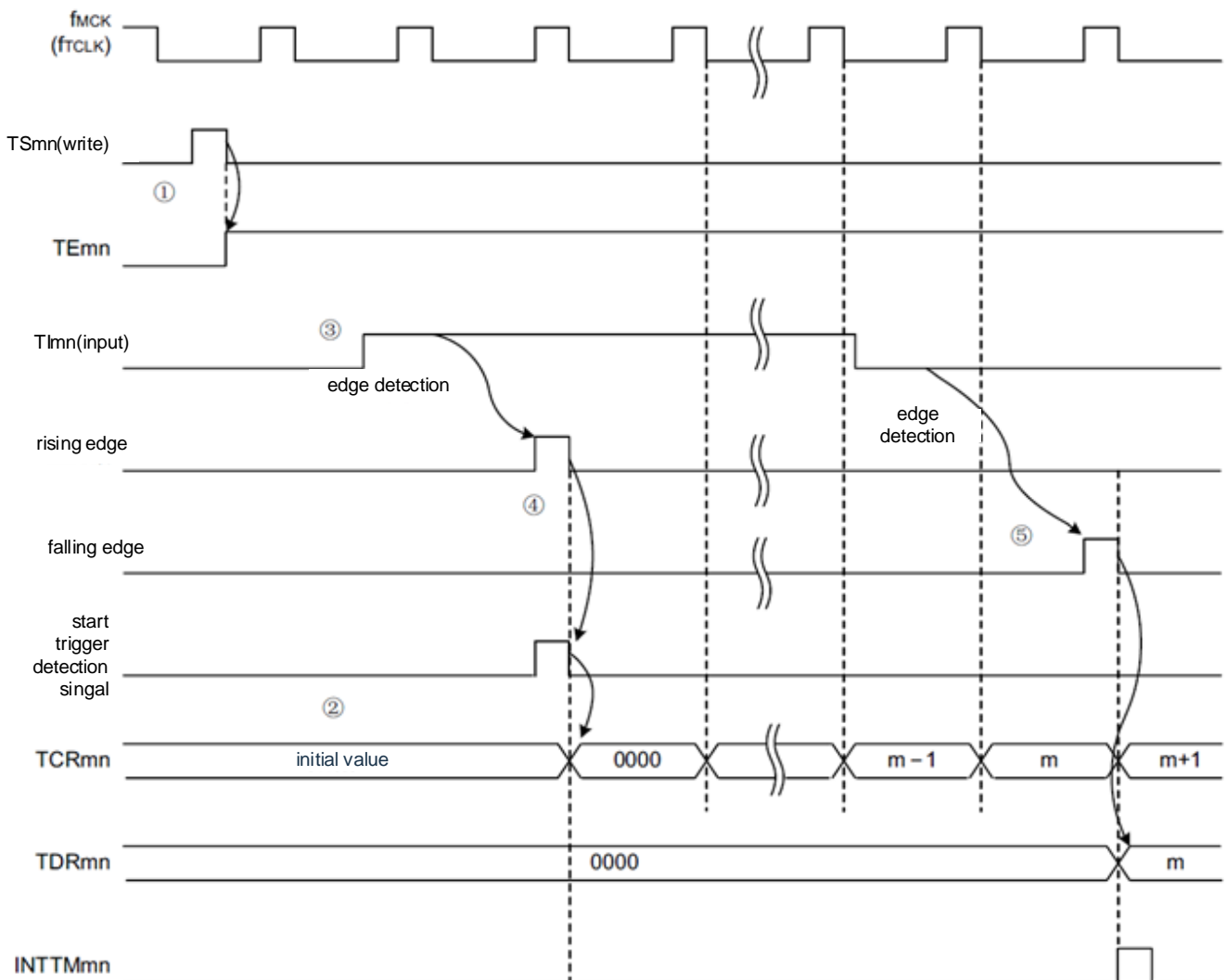


Remark This is a timing without the noise filter. If the noise filter is used, the edge detection is delayed by 2 more F_{MCK} cycles (3~4 cycles in total) from the TImn input. The 1 cycle error is because the TImn input is not synchronized with the count clock (F_{MCK}).

(3) 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 (TSMn)(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.

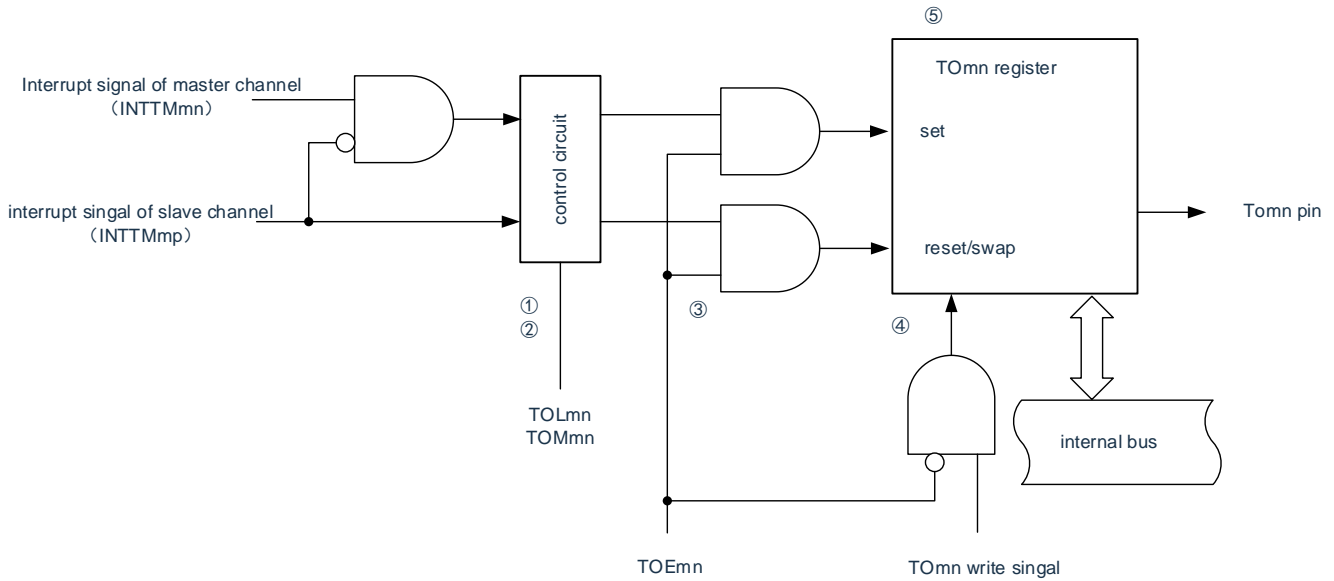
Figure 5-30 Operation timing (capture & single count mode: measurement of high-level width)



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 T_{Imn} input. The 1 cycle error is because the T_{Imn} 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 5-31 Output circuit configuration



The following explains the output circuit of the TOmn pin.

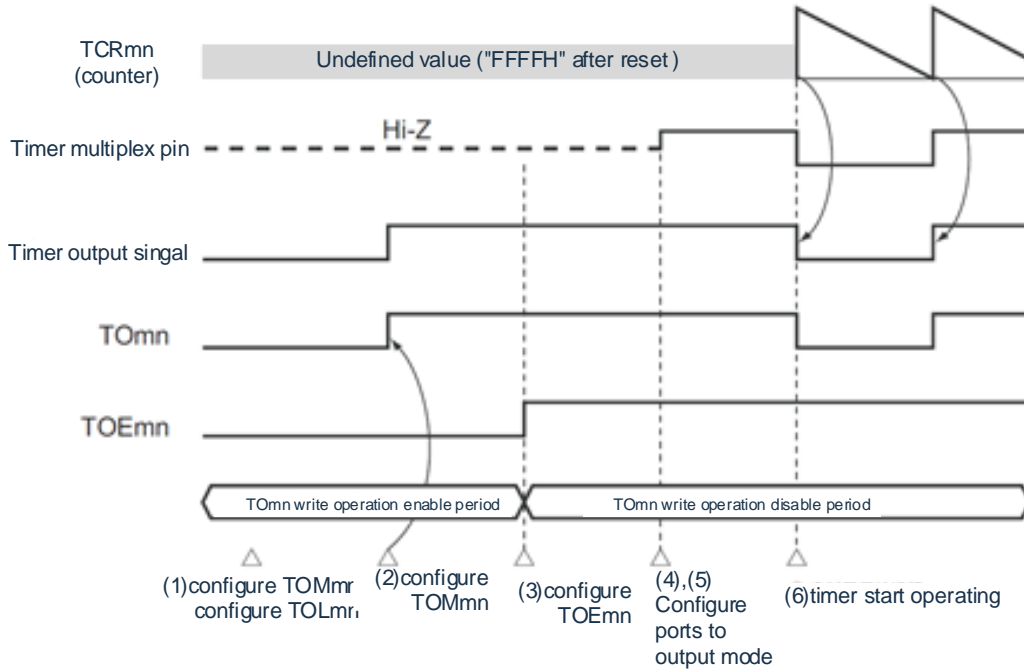
- ① When the TOMmn bit is "0" (master channel output mode), the setting value of timer output level register m (TOLm) is ignored and only INTTMmp (slave channel timer interrupt) is passed to timer output register m (TOm).
- ② When the TOMmn bit is "1" (slave channel output mode), INTTMmn (master channel 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 configuration

The following figure shows the procedure and status transition of the TOmn output pin from initial setting to timer operation start.

Figure 5-32 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.

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

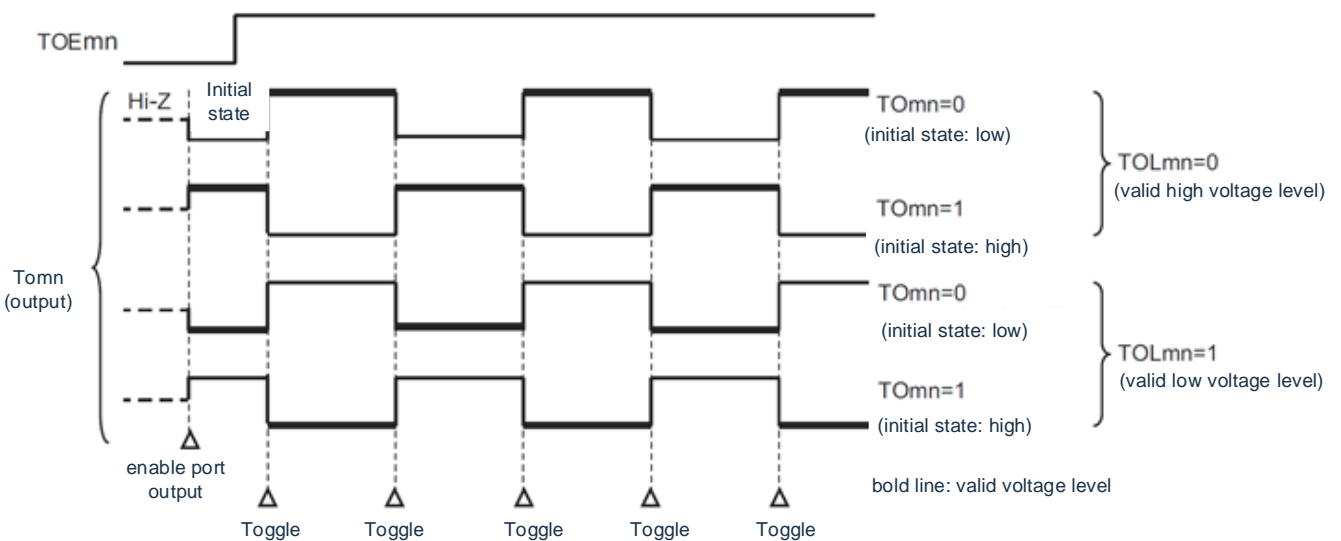
(2) Default level of TOMn pin and output level after timer operation start

The change in the output level of the TOMn pin when timer output register m (TOM) is written while timer output is disabled (TOEmn = 0), the initial level is changed, and then timer output is enabled (TOEmn = 1) before port output is enabled, is shown below.

(a) 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 a toggle signal.

Figure 5-33 Output state of TOMn pin at toggle output (TOMmn=0)

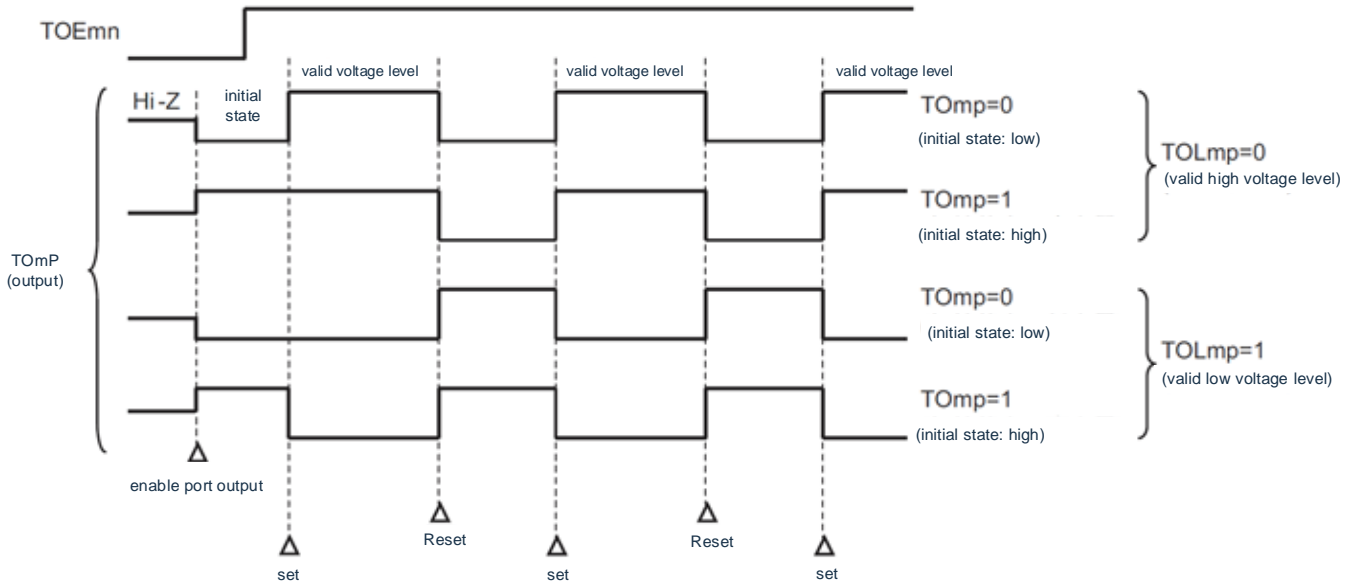


Remark:

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

- (b) 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 5-34 Output state of TOmn pin at PWM output (TOMmn=1)



Remark1. Set: The output signal from the TOmp pin changes from an invalid level to a valid level.
 Reset: The output signal from the TOmp pin changes from a valid level to an invalid level.
 2.m: unit number (m=0, 1) n: channel number (p=1~3)

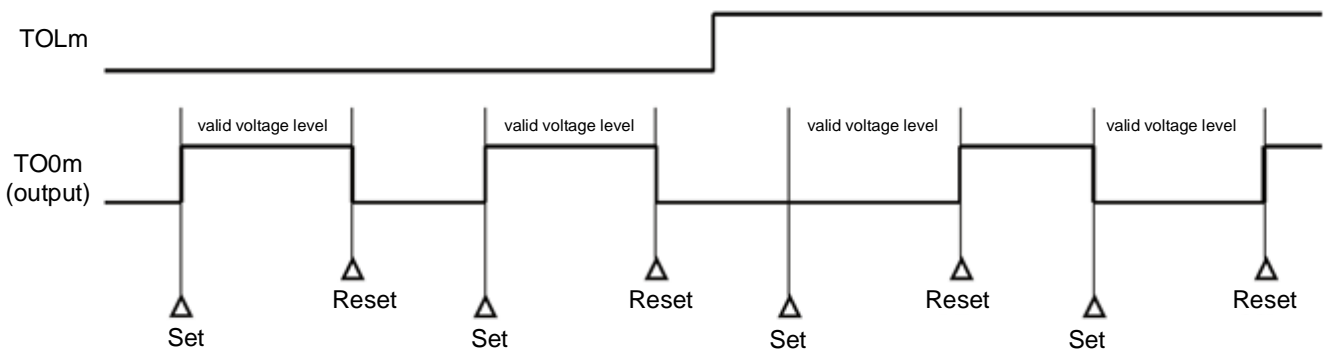
(3) Operation of TOMn pin in slave channel output mode (TOMmn = 1)

(a) When timer output level register m (TOLm) setting has been changed during timer operation

When the TOLm register setting has been changed during timer operation, the setting becomes valid at the generation timing of the TOMn pin change condition. Rewriting the TOLm register does not change the output level of the TOMn pin.

The operation when TOMmn is set to 1 and the value of the TOLm register is changed while the timer is operating (TEmn = 1) is shown below.

Figure 5-35 Operation when the contents of the TOLm register are changed during timer operation



Remark:

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

(b) 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.

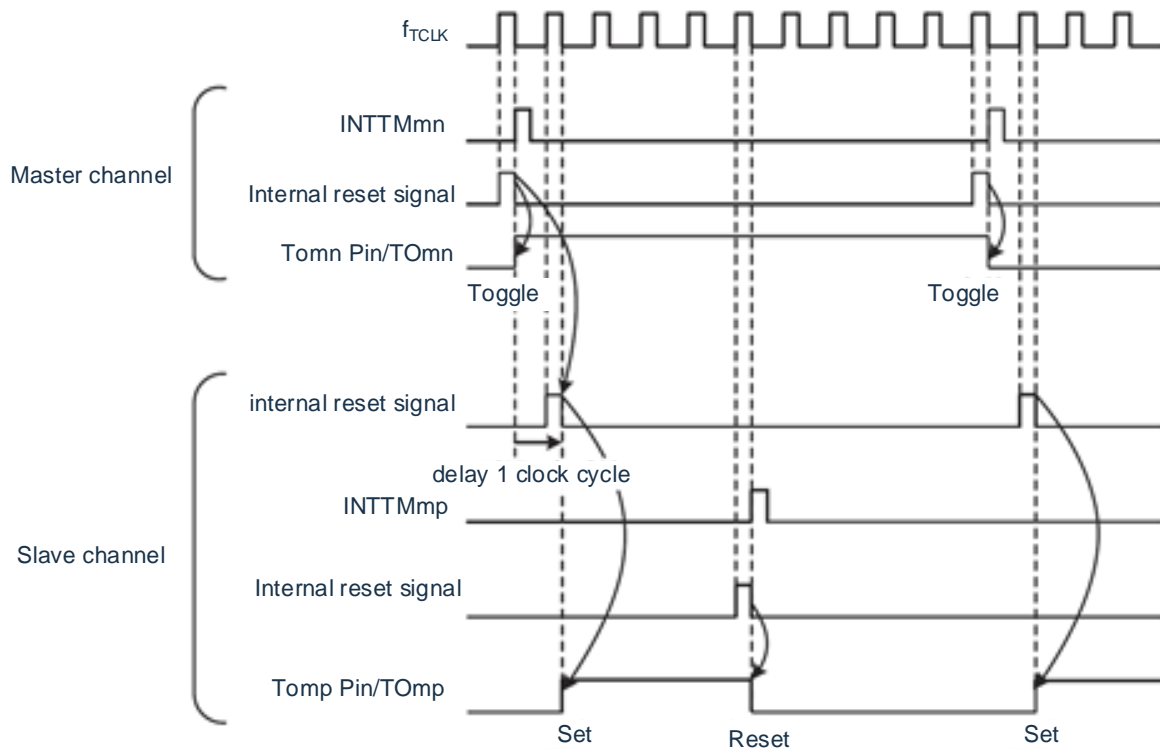
The set/reset operation status when setting the master/slave channel according to the following method is shown in Figure 5-35.

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

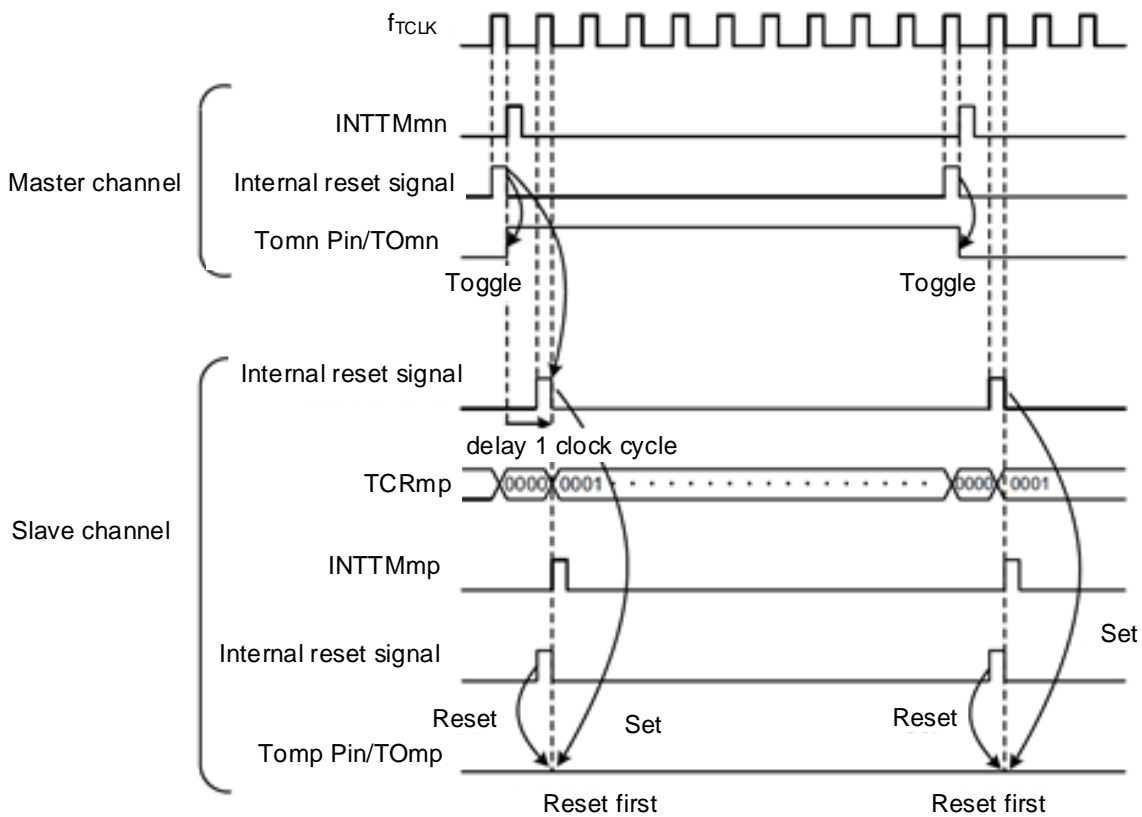
Slave channel: TOEmp=1, TOMmp=1, TOLmp=0

Figure 5-36 Set/reset timing operation status

(1) Basic operation timing



(2) Operation timing when duty cycle is 0%



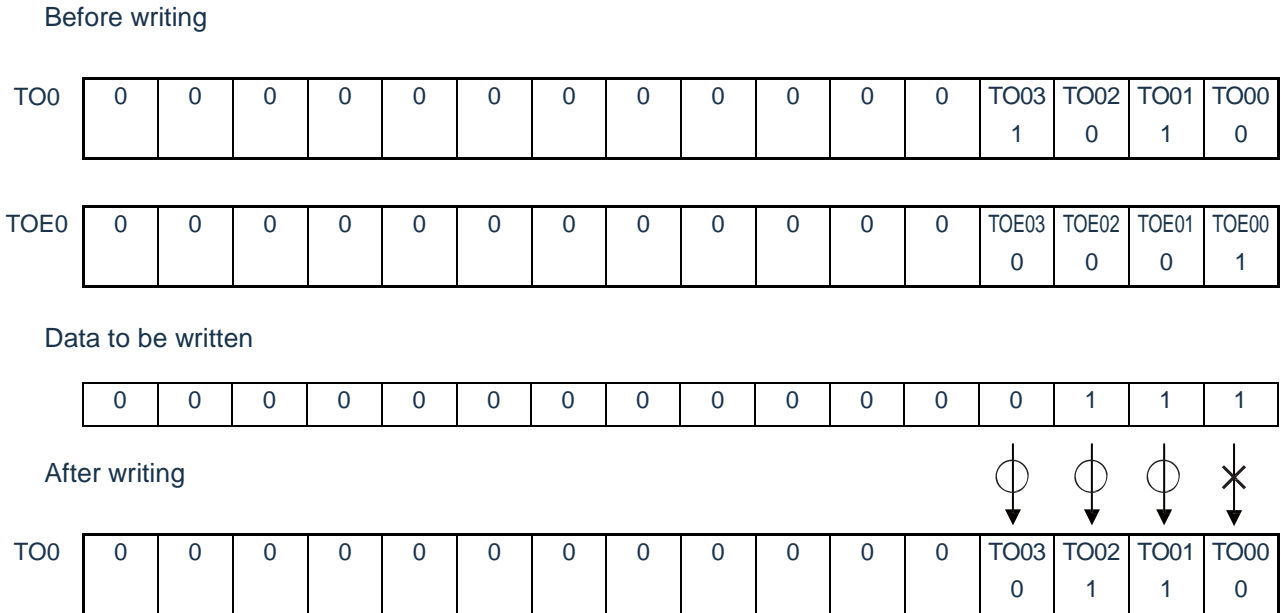
Remark:

1. Internal reset signal: TOMn pin reset/toggle signal
Internal set signal: TOMn pin set signal
2. 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 TO_mn bit

Like the timer channel start register *m* (TSM), the timer output register *m* (TOM) has the set bits (TO_mn) for all channels and can therefore operate the TO_mn bits for all channels at once.

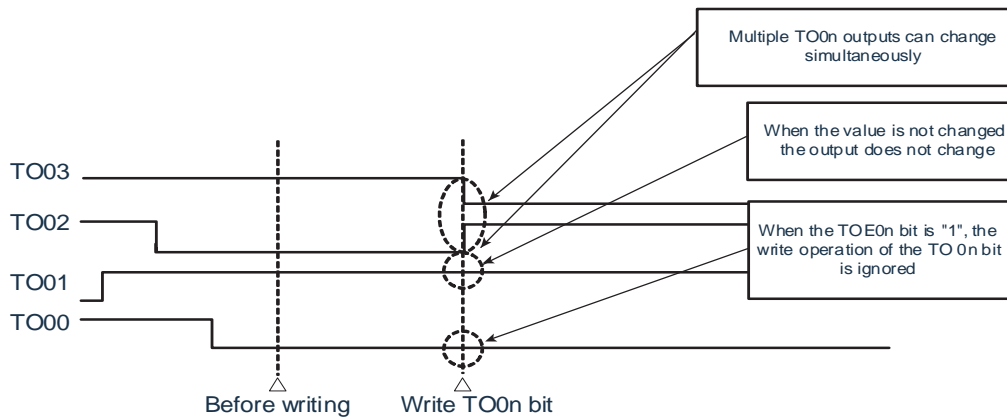
Figure 5-37 One-time operation example of TO₀n bit



Only TO_mn bits with TOE_mn bit "0" can be written, and write is ignored when the TO_mn bit is "1".

TO_mn (channel output) to which TOE_mn = 1 is set is not affected by the write operation. Even if the write operation is done to the TO_mn bit, it is ignored and the output change by timer operation is normally done.

Figure 5-38 TO₀n pin status when the TO₀n bit is operated at one time



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

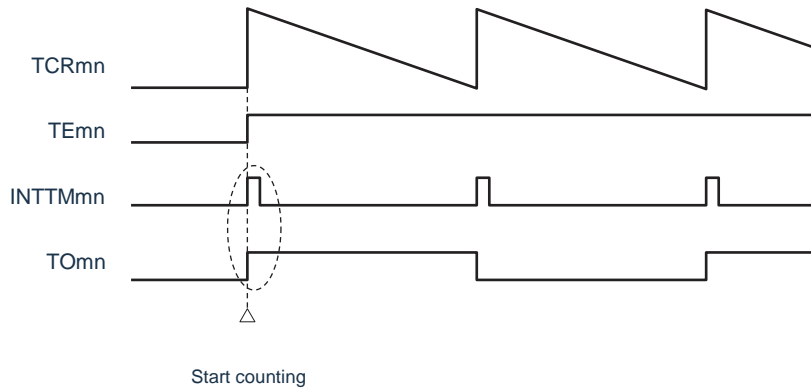
5.6.5 Timer interrupt and TOMn pin output when counting starts

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.

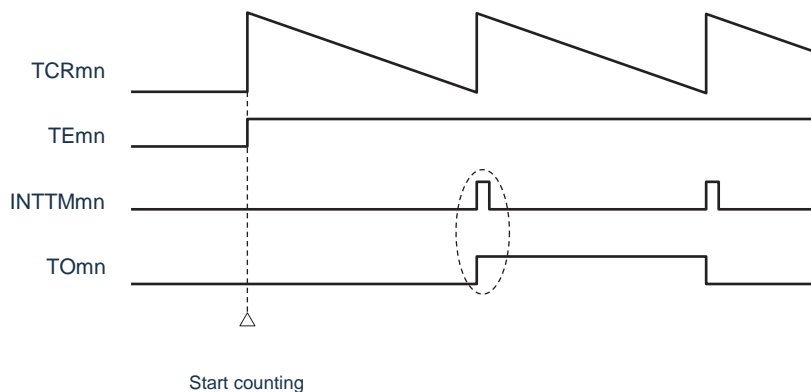
When the MDmn0 bit is "1", the start timing of the count can be known by generating a timer interrupt (INTTMmn). In other modes, the timer interrupt and TOMn output at the start of counting are not controlled. An example of operation when set to interval timer mode (TOEmn=1, TOMmn=0) is shown below.

Figure 5-39 An operation example of timer interrupt and TOMn output when counting starts

(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.

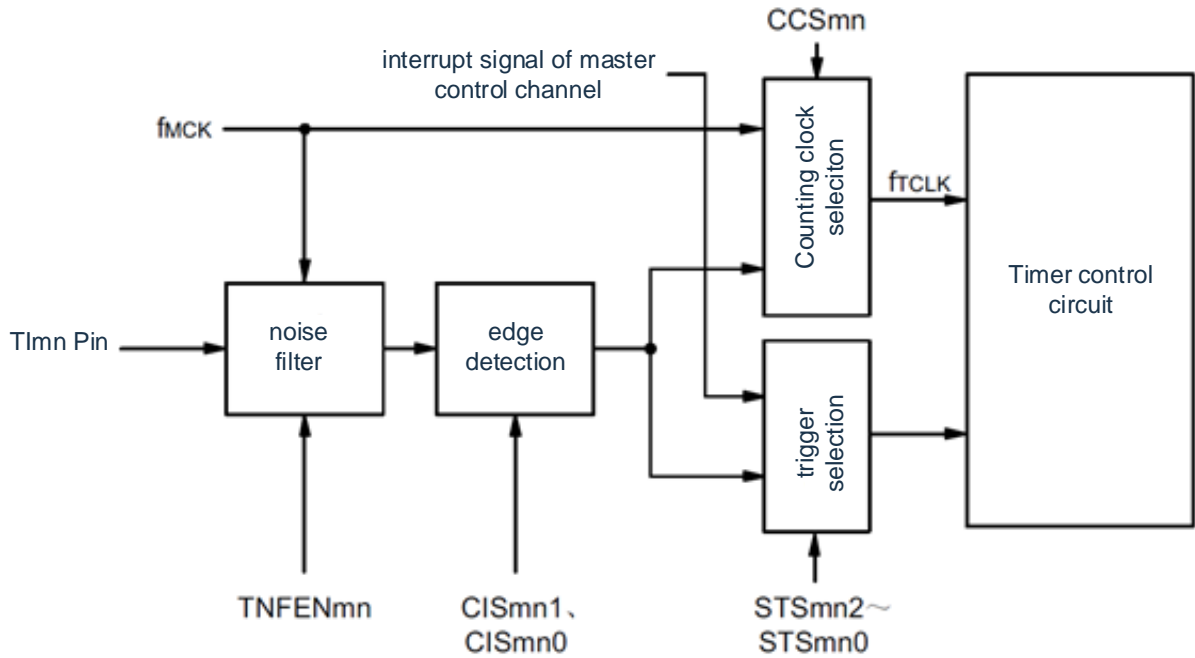
Remark: m: unit number (m=0, 1) n: channel number (n=0~3)

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.

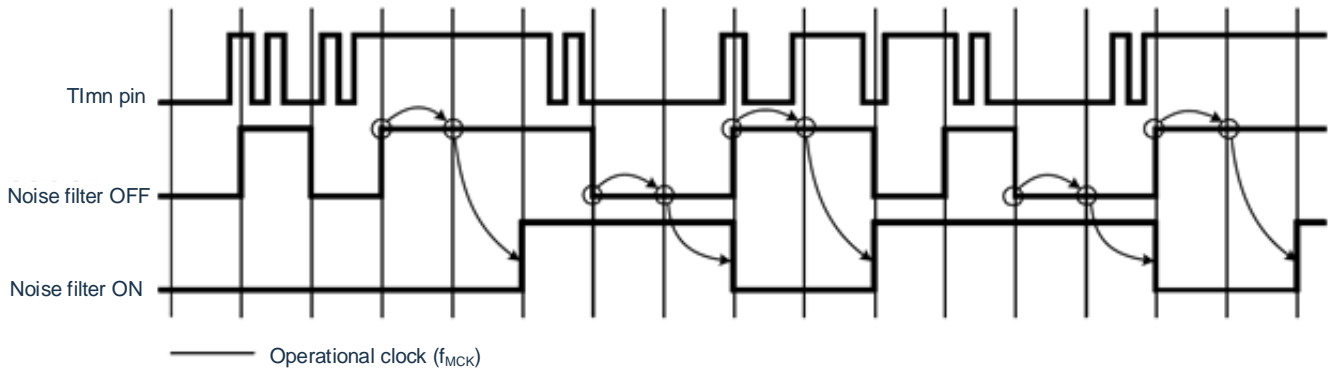
Figure 5-40 Structure of input circuit



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 TM4Inn input pin after the noise filter circuit with the noise filter ON or OFF is shown below.

Figure 5-41 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:

$$\text{INTTMmn (timer interrupt) generation period} = \text{count clock period} \times (\text{TDRmn set value} + 1)$$

(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:

$$\text{Period of square wave output from TOMn} = \text{Period of count clock} \times (\text{TDRmn set value} + 1) \times 2$$

$$\text{Frequency of square wave output from TOMn} = \text{Frequency of count clock} / \{(\text{TDRmn set value} + 1) \times 2\}$$

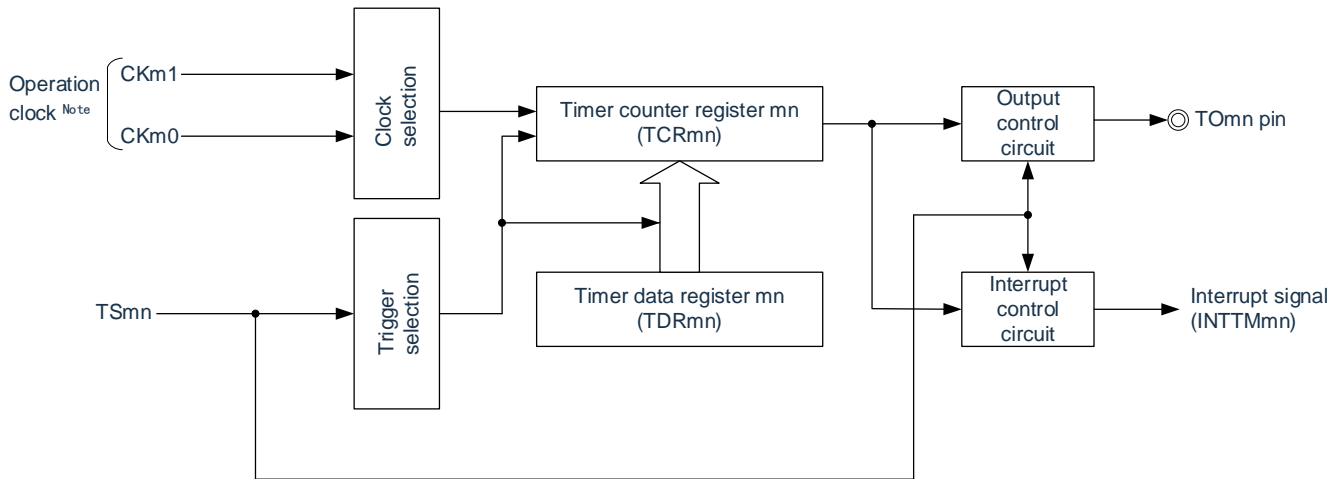
In the interval timer mode, the timer count register mn (TCRmn) is used as a decrement counter.

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.

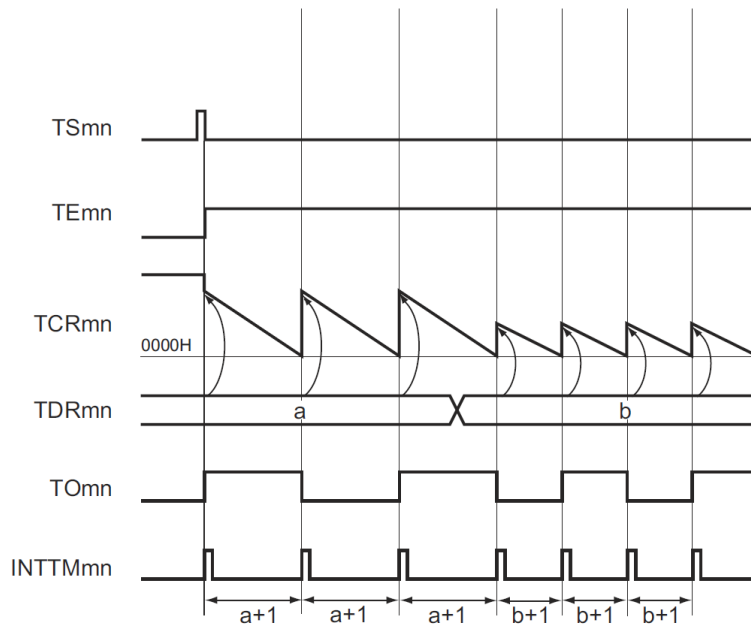
The TDRmn register can be rewritten at any time, and the rewritten TDRmn register value is valid from the next cycle.

Figure 5-42 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.

Figure 5-43 Example of basic timing operating as an interval timer/square wave output (MDmn0=1)

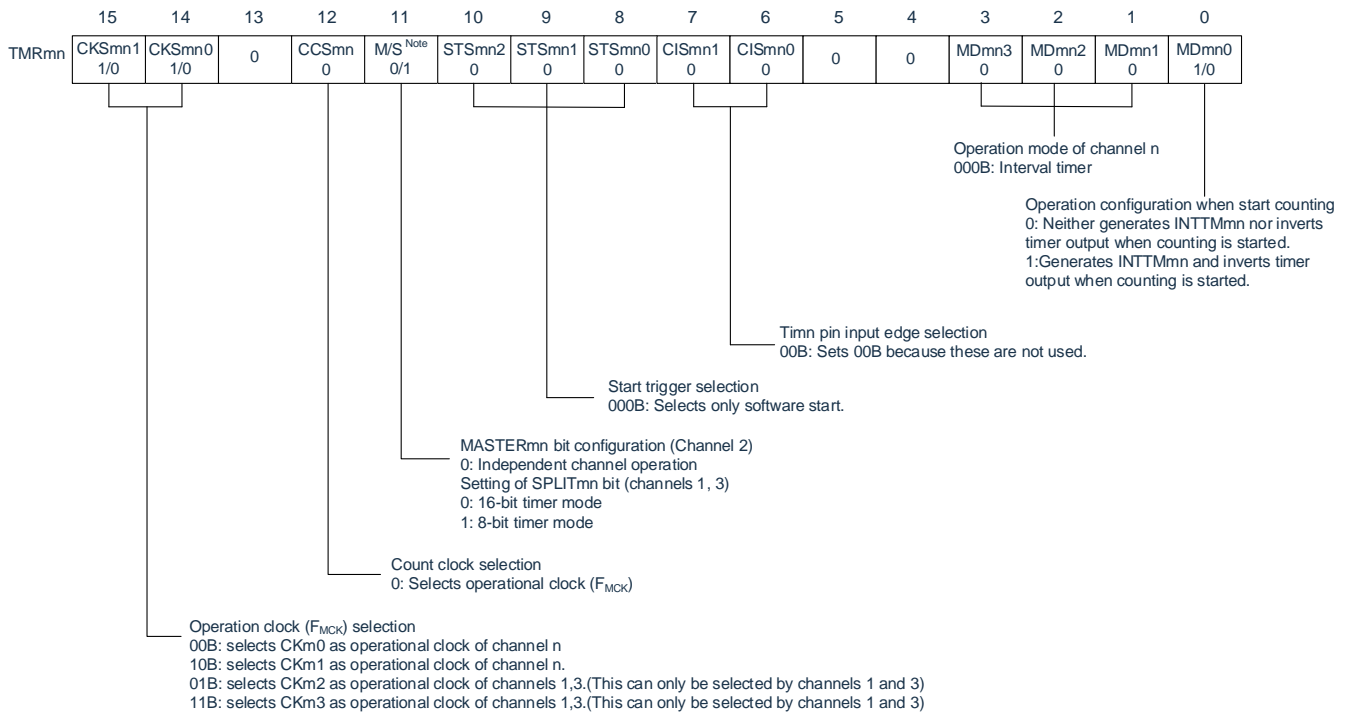


Remark:

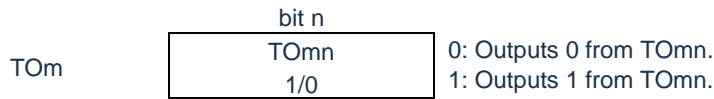
1. m: unit number (m=0, 1) n: channel number (n=0 ~ 3)
2. TSmn: Bit n of timer channel start register m (TSM)
- TEmn: Bit n of timer channel enable status register m (TEM)
- TCRmn: Timer count register mn (TCRmn)
- TDRmn: Timer data register mn (TDRmn)
- TOMn: TOMn pin output signal

Figure 5-44 Example of register setting contents for interval timer/square wave output

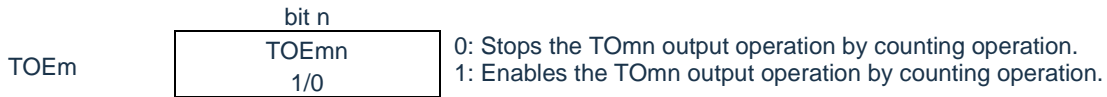
(a) Timer mode register mn (TMRmn)



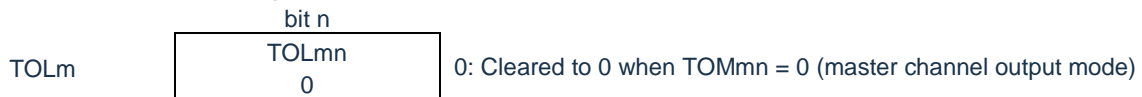
(b) Timer output register m (TOm)



(c) Timer output enable register m (TOEm)



(d) Timer output level register m (TOLm)



(e) Timer output mode register m (TOMm)



Note TMRm2: MASTERmn bit
TMRm1, TMRm3: SPLITmn bit
TMRm0: Fixed to "0".

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

Figure 5-45 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 supply 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 supply state. (Start clock supply, can write to each register)
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.
Restart operation	In operation	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	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.
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 TOMn pin outputs the level set by the TOMn bit. 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)

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:

$$\text{Specified count value} = \text{TDRmn set value} + 1$$

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".

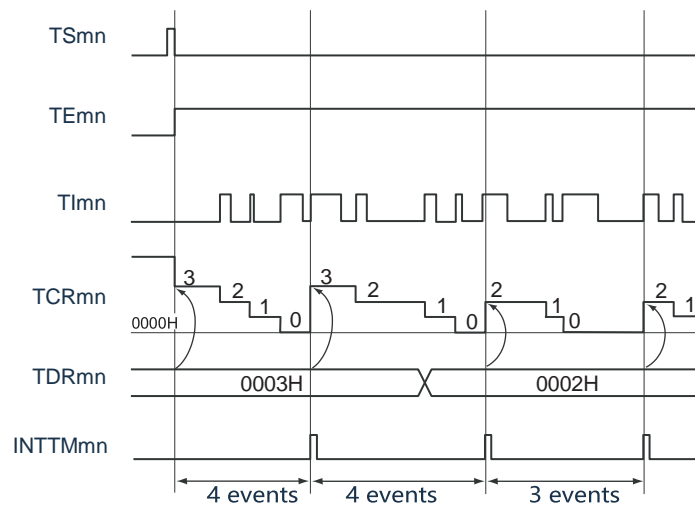
The TCRmn register decrements the count while detecting the active edge of the TImn pin input. If TCRmn becomes "0000H", the value of TDRmn register is loaded again and INTTMmn is output.

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.

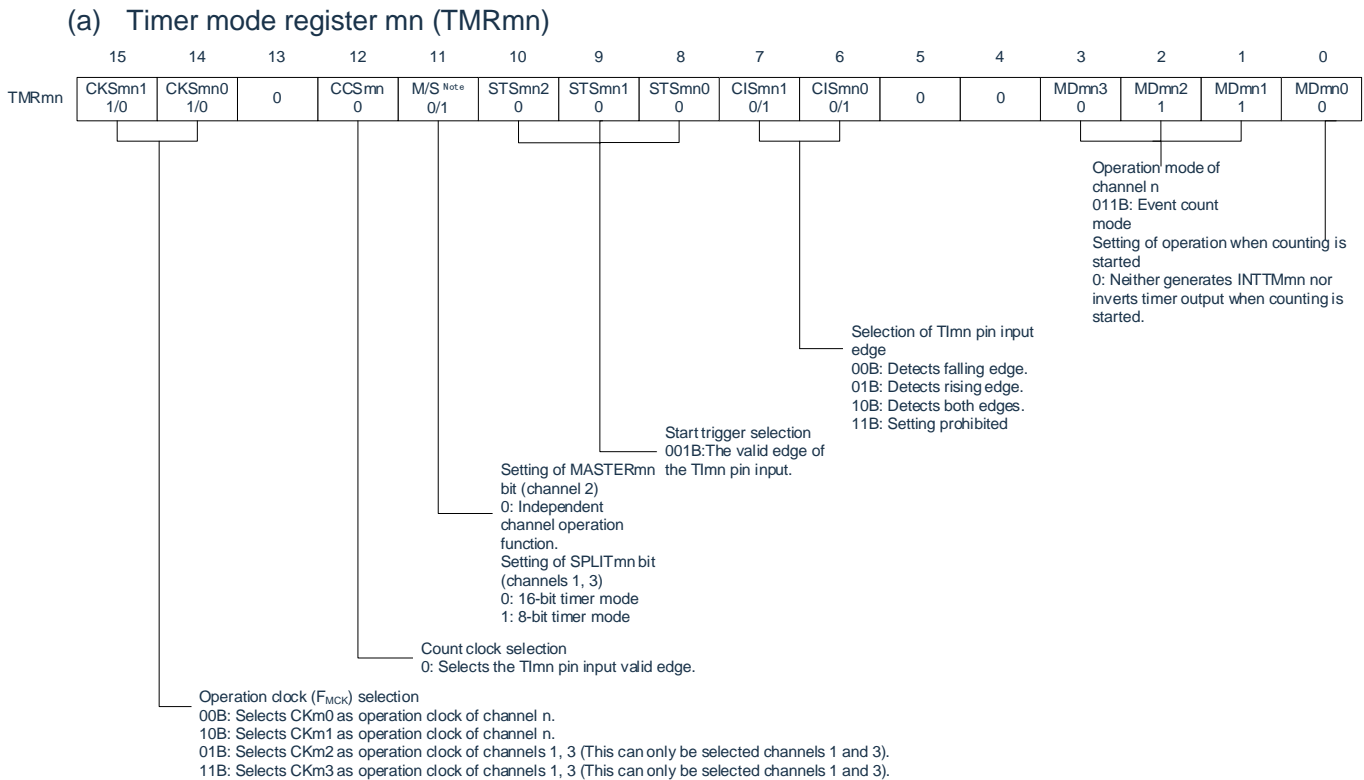
Figure 5-46 Example of basic timing operating as external event counter



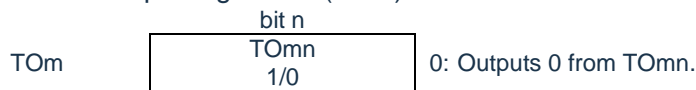
Remark:

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)

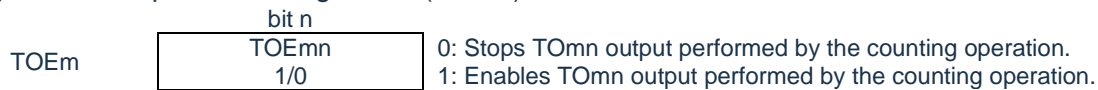
Figure 5-47 Example of register setting contents in external event counter mode



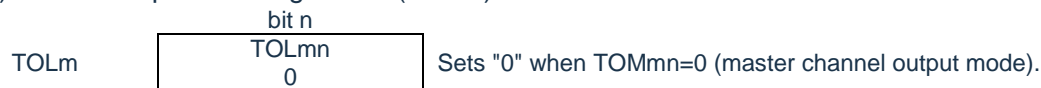
(b) Timer output register m (TOM)



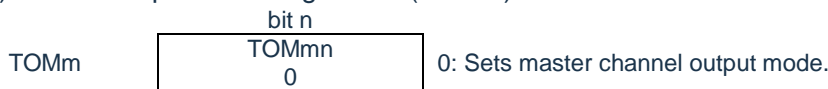
(c) Timer output enable register m (TOEm)



(d) Timer output level register m (TOLm)



(e) Timer output mode register m (TOMm)



Note TMRm2 : MASTERmn bit
 TMRm1, TMRm3: SPLITmn bit
 TMRm0 : Fixed to "0",

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

Figure 5-48 Procedure for external event counter function

	Software operation	Hardware status
Timer4 initial settings		The input clock of timer unit m is in the stop supply state. (Stop clock supply, 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 supply state and the channels are in the stop state. (Start providing clock, can write to each register)
Initial setting of channels	Set the timer clock selection register m (TPSm). Determine the clock frequency of CKm0 ~ CKm3.	
	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:

$$\text{Divider clock frequency} = \text{input clock frequency} / \{(\text{TDR00 set value} + 1) \times 2\}$$
- Select both edges:

$$\text{Divider clock frequency} \approx \text{input clock frequency} / (\text{TDR00 set value} + 1)$$

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.

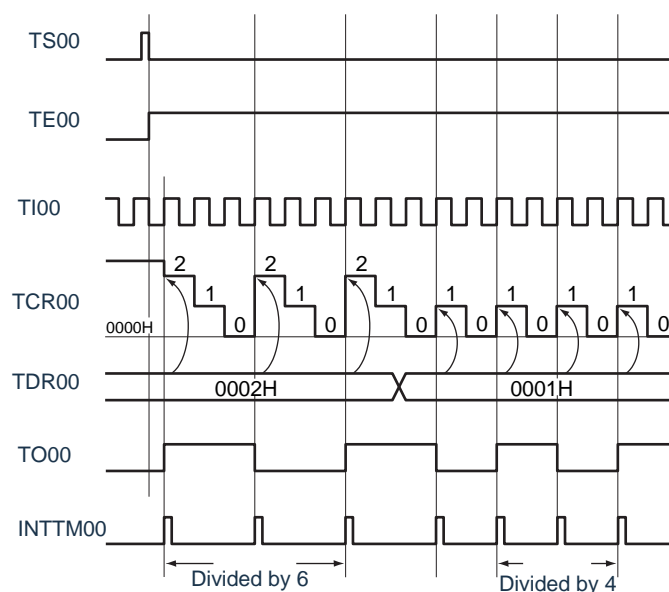
If double edge detection is selected for the TI00 pin input, the duty cycle error of the input clock affects the clock period of the TO00 output's division.

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

$$\text{clock period of the TOmn output} = \text{supposed TOmn output clock period} \pm \text{operating clock period (error)}$$

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

Figure 5-49 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: TI00pin input signal

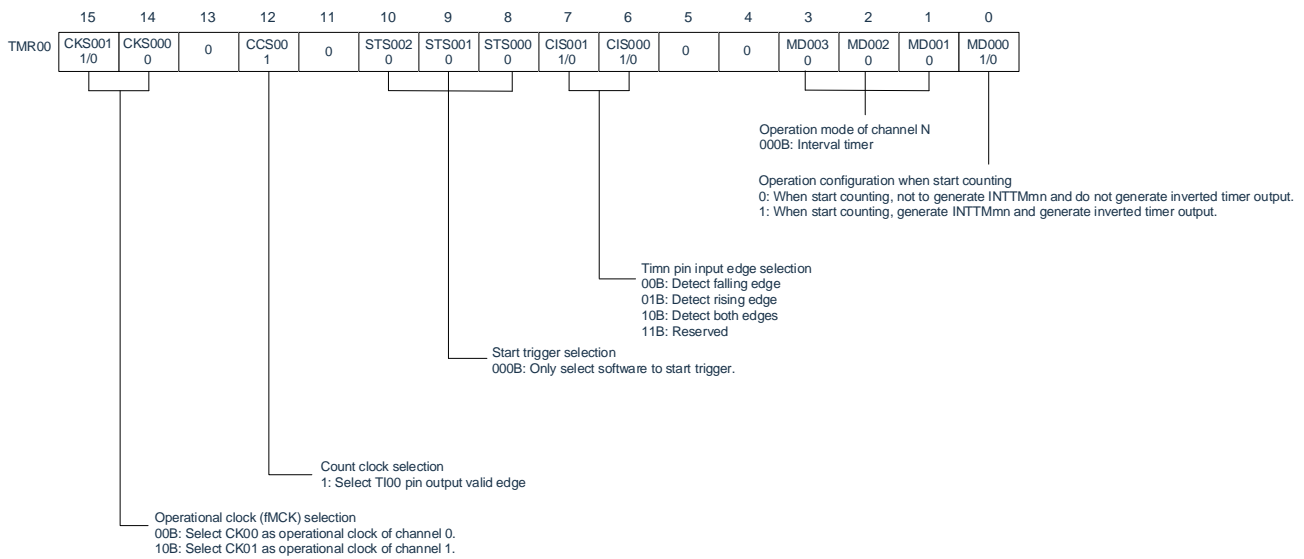
TCR00: Timer count register 00 (TCR00)

TDR00: Timer data register 00 (TDR00)

TO00: TO00pin output signal

Figure 5-50 Example of register contents setting when operating as frequency divider

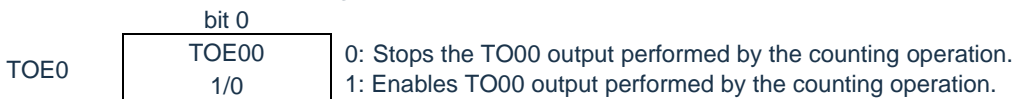
(a) Timer mode register 00 (TMR00)



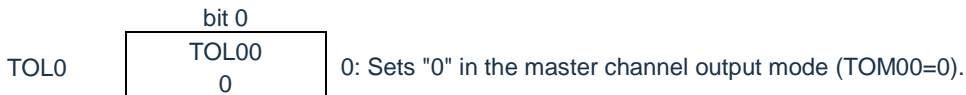
(b) Timer output register 0 (TO0)



(c) Timer output enable register 0 (TOE0)



(d) Timer output level register 0 (TOL0)



(e) Timer output mode register 0 (TOM0)

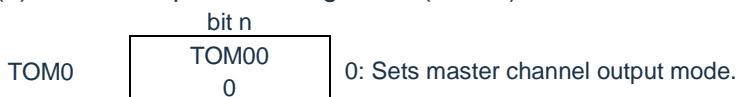


Figure 5-51 Procedure for frequency divider function

	Software operation	Hardware status
Timer4 initial settings		The input clock of timer unit 0 is in the stop supply 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 0 is in the supply 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. (Supply 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	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.
	Set TOE00 bit to "1" and enable TO00 output.	
	Set the Port Register and Port Mode Register to "0"	
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 state.
	Set the TOE00 bit to "0" and set the value for the TO00 bit.	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)



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:

$$TImn \text{ input pulse interval} = \text{period of counting clock} \times ((10000H \times TSRmn:OVF) + (TDRmn \text{ captured value} + 1))$$

Notice An 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).

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.

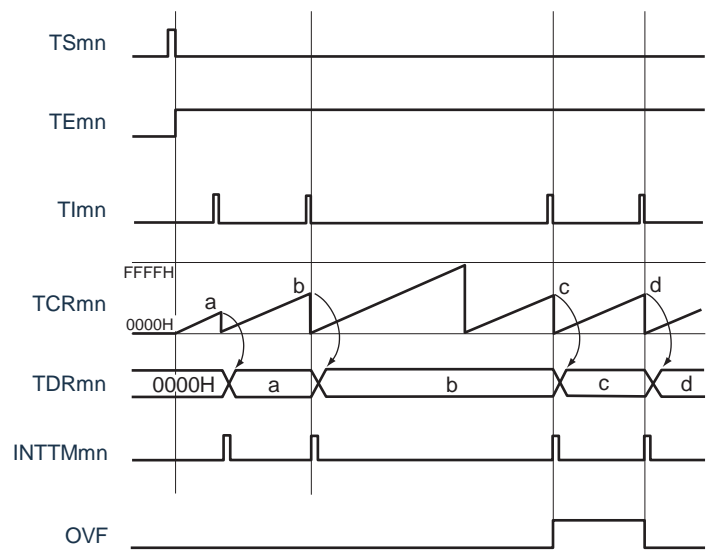
If the active edge of the TImn pin input is detected, the count value of TCRmn register is transferred (captured) to Timer Data Register mn (TDRmn), and the TCRmn register is cleared to "0000H", and then INTTMmn is output. If the counter overflows, the OVF bit of Timer Status Register mn (TSRmn) is set 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 5-52 Example of basic timing operating as an input pulse interval measurement (MDmn0=0)



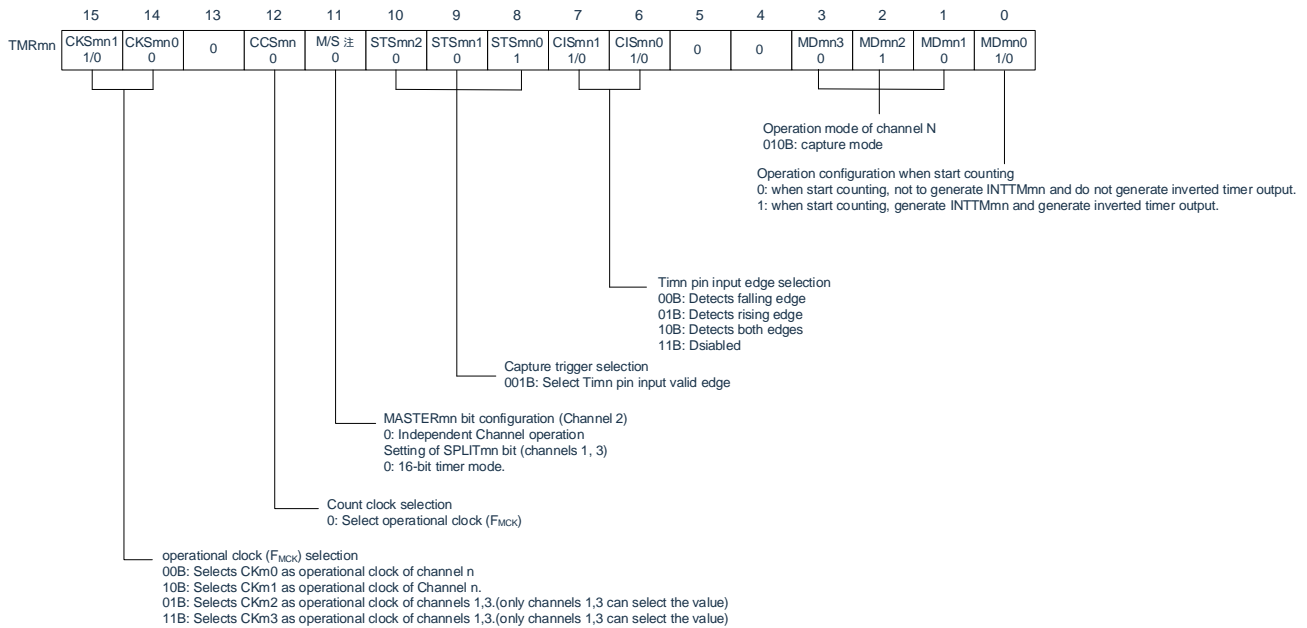
Remark:

1. m: unit number (m= 0) n: channel number (n=0 ~ 3)
2. TSmn: Bit n of timer channel start register m (TSM)

TE_mn: Bit n of timer channel enable status register m (TE_m)
 TI_mn: TI_mn pin input signal
 TCR_mn: Timer count register mn (TCR_mn)
 TDR_mn: Timer data register mn (TDR_mn)
 OV_F: Bit0 of timer status register mn (TSR_mn)

Figure 5-53 Example of register contents setting in measuring input pulse interval

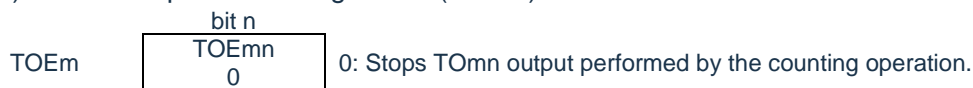
(a) Timer mode register mn (TMR_mn)



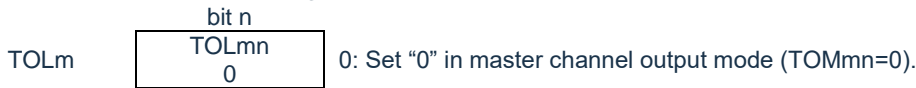
(b) Timer output register m (TO_m)



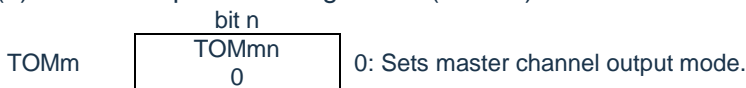
(c) Timer output enable register m (TOE_m)



(d) Timer output level register m (TOL_m)



(e) Timer output mode register m (TOM_m)



Note TMR_m2 : MASTER_mn bit
 TMR_m1, TMR_m3: SPLIT_mn bit
 TMR_m0 : Fixed to "0".

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

Figure 5-54 Procedure for entering the pulse interval measurement function

	Software operation	Hardware status
Timer4 initial settings		The input clock of timer unit m is in the stop supply state. (Stop to supply 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 supply state and the channels are in the stop state. (Start to supply 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.
	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.
Restart 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.
Stop operation	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.
Timer4 stop		

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

Notice 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.

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:

$$\text{Signal width of TImn input} = \text{period of count clock} \times ((10000\text{H} \times \text{TSRmn: OVF}) + (\text{TDRmn captured value} + 1))$$

Notice Because the TImn pin inputs are sampled by the operation clock selected by the CKSmn bit of the Timer Mode Registerm (TMRmn), an error of 1 operation clock is generated.

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.

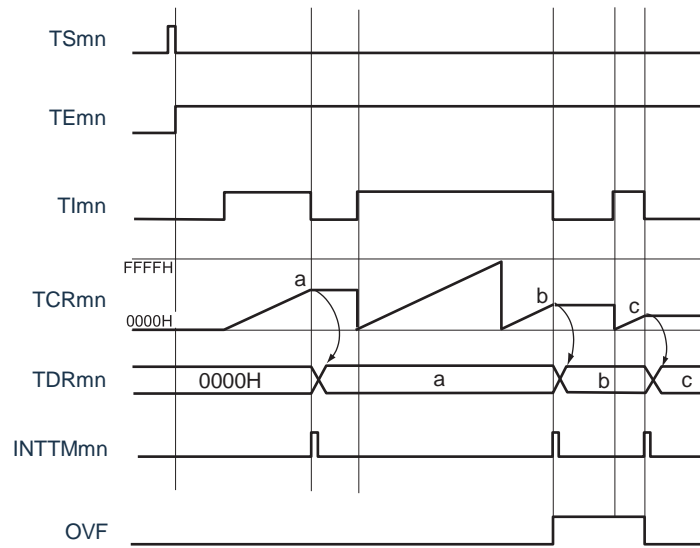
Even if the counter counts 2 or more complete cycles, the overflow is considered to have occurred and the OVF bit of the TSRmn register is set to "1". However, when two or more overflows occur, the interval value cannot be measured normally by the OVF bit.

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.

Figure 5-55 Example of basic timing operating as high-/low-level width measurement of input signal

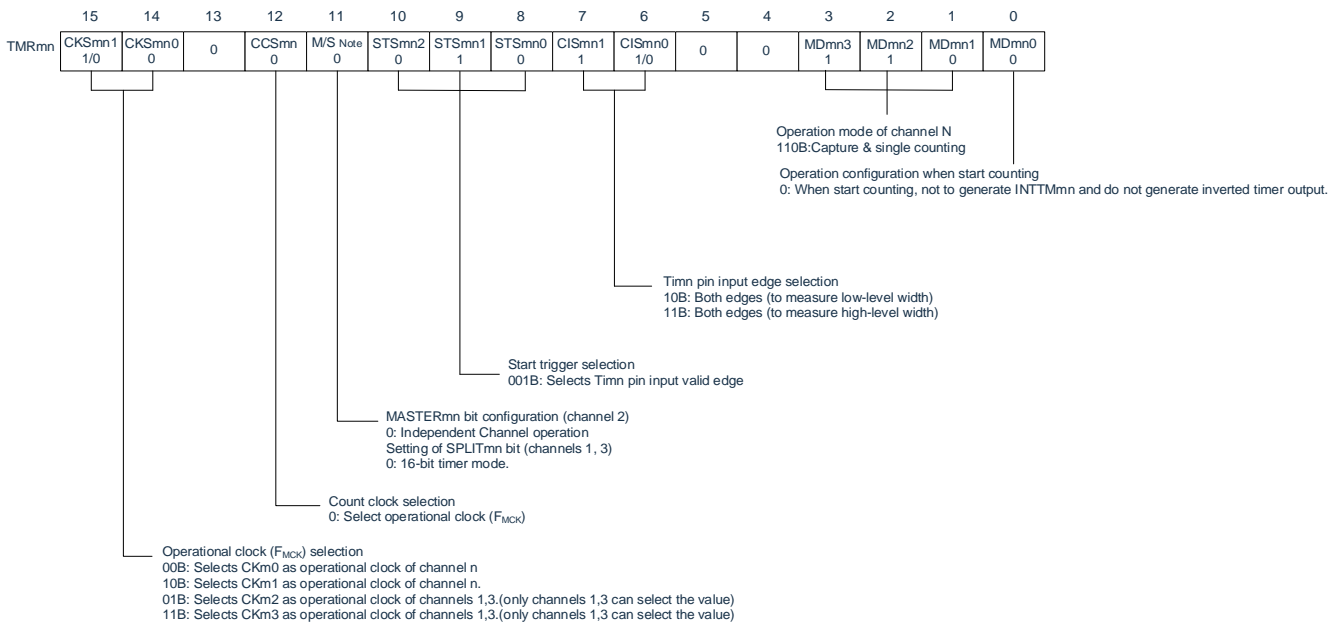


Remark:

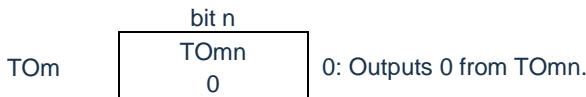
1. m: unit number (m= 0) n: channel number (n=0 ~ 3)
 2. TSmn: Bit n of timer channel start register m (TSm)
- TE mn: Bit n of timer channel enable status register (TEm)
- TImn: TImn pin input signal
- TCRmn: Timer count register mn (TCRmn)
- TDRmn: Timer data register mn (TDRmn)
- OVf: Bit 0 of timer status register mn (TSRmn)

Figure 5-56 Example of register contents setting in measuring high-/low-level width of input signal

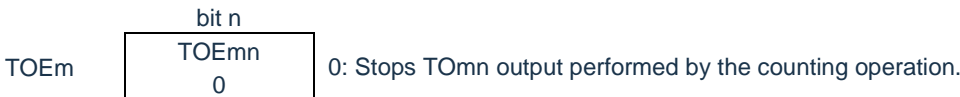
(a) Timer mode register mn (TMRmn)



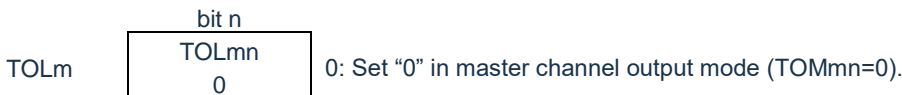
(b) Timer output register m (TOM)



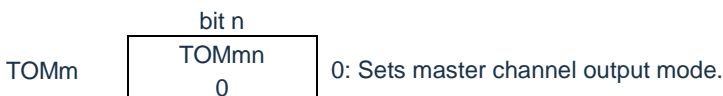
(c) Timer output enable register m (TOEm)



(d) Timer output level register m (TOLm)



(e) Timer output mode register m (TOMm)



Note TMRm2 : MASTERmn bit
 TMRm1, TMRm3 : SPLITmn bit
 TMRm0 : Fixed to "0".

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

Figure 5-57 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 supply state. (Stop to supply 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 to supply clock, can write to each register)
Initial setting of channels	Set the timer clock selection register m (TPSm). Determine the clock frequency of CKm0 ~ CKm3.	
	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 Registermn (TDRmn). Set the TOEmn bit to "0" and stop the operation of TOMn.	The channel is in the stop state. (Provides clock, and consumes some power)
Start Operation	Set the TSmn bit to "1".	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").
	Since the TSmn bit is a trigger bit, it automatically returns to "0". 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 INTTMmn (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:

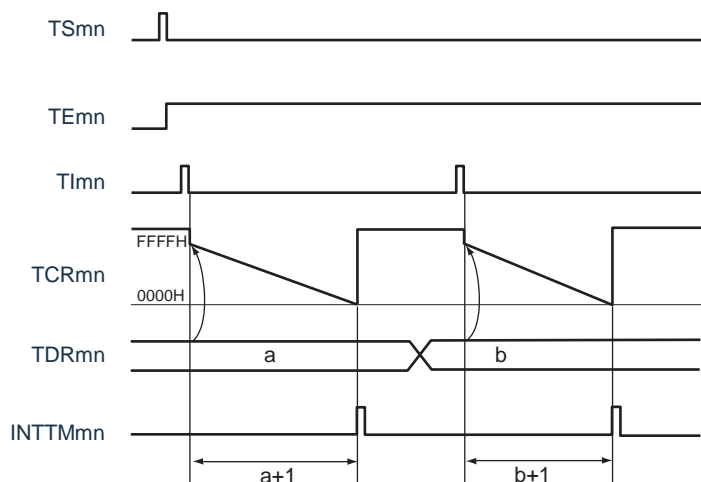
$$\text{INTTMmn (timer interrupt) generation period} = \text{counting clock period} \times (\text{TDRmn set value} + 1)$$

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 5-58 Example of basic timing operating as delay counter

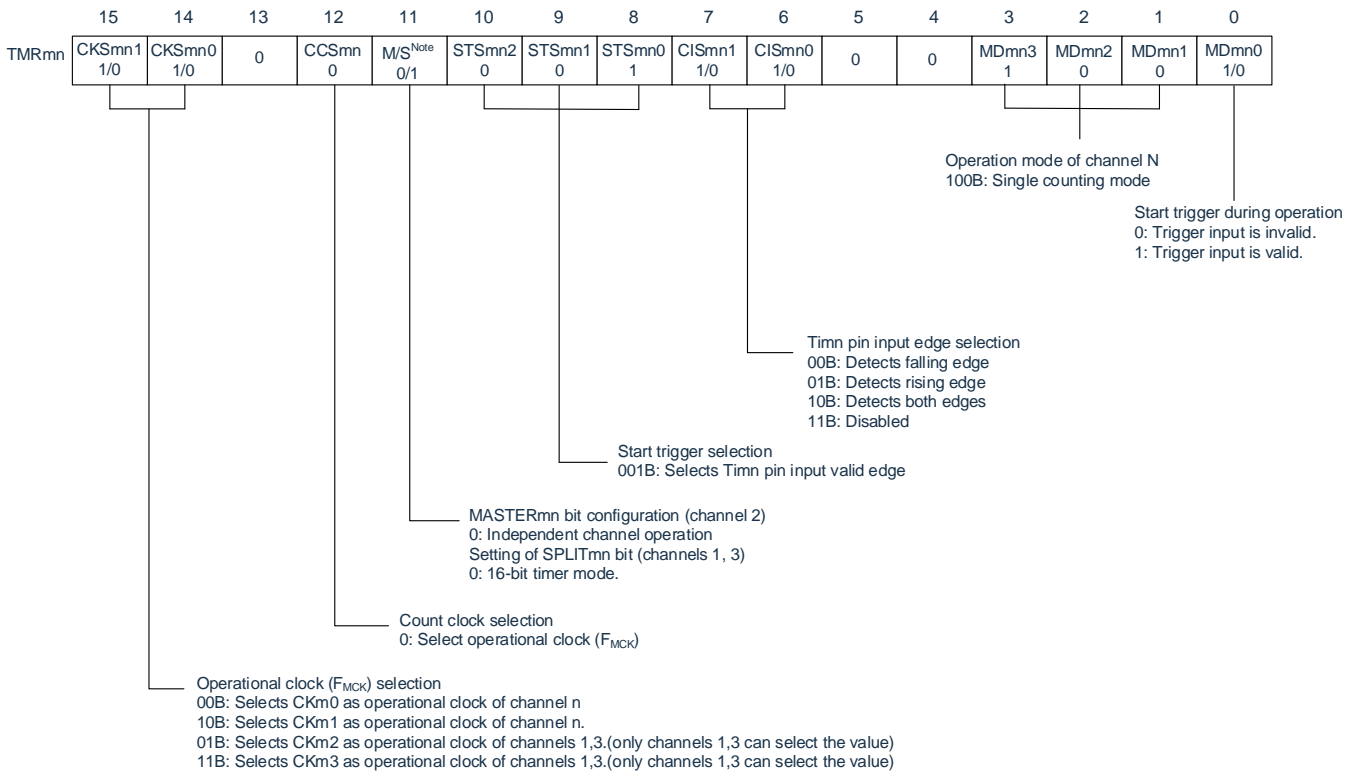


Remark:

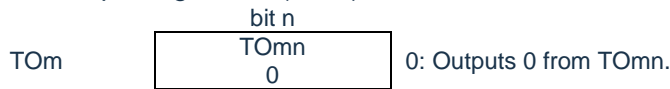
1. m: unit number (m=0, 1) n: channel number (n=0~3)
 2. TSmn: Bit n of timer channel start register m (TSm)
- TEmn: Bit n of timer channel enable status register m (TEm)
- TImn: TImn pin input signal
- TCRmn: Timer count register mn (TCRmn)
- TDRmn: Timer data register mn (TDRmn)

Figure 5-59 Example of register contents setting for delay counter function

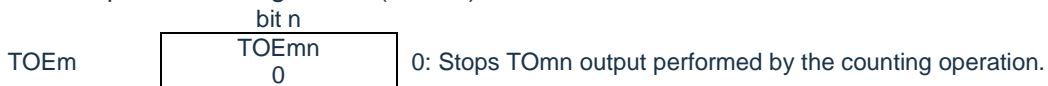
(a) Timer mode register mn (TMRmn)



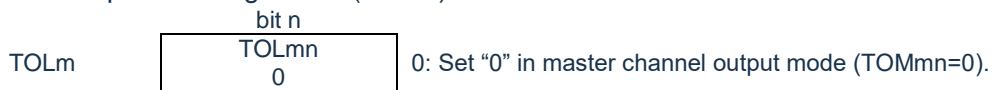
(b) Timer output register m (TOM)



(c) Timer output enable register m (TOEm)



(d) Timer output level register m (TOLm)



(e) Timer output mode register m (TOMm)



Note TMRm2 : MASTERmn bit
 TMRm1, TMRm3: SPLITmn bit
 TMRm0 : Fixed to "0".

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

Figure 5-60 Procedure for delay counter function

	Software operation	Hardware status
Timer4 initial settings		The input clock of timer unit m is in the stop supply 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 to supply 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.
	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:

$$\begin{aligned} \text{Delay} &= \{\text{TDRmn (master) set value} + 2\} \times \text{counting clock period} \\ \text{Pulse width} &= \{\text{TDRmp (slave) set value}\} \times \text{counting clock period} \end{aligned}$$

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.

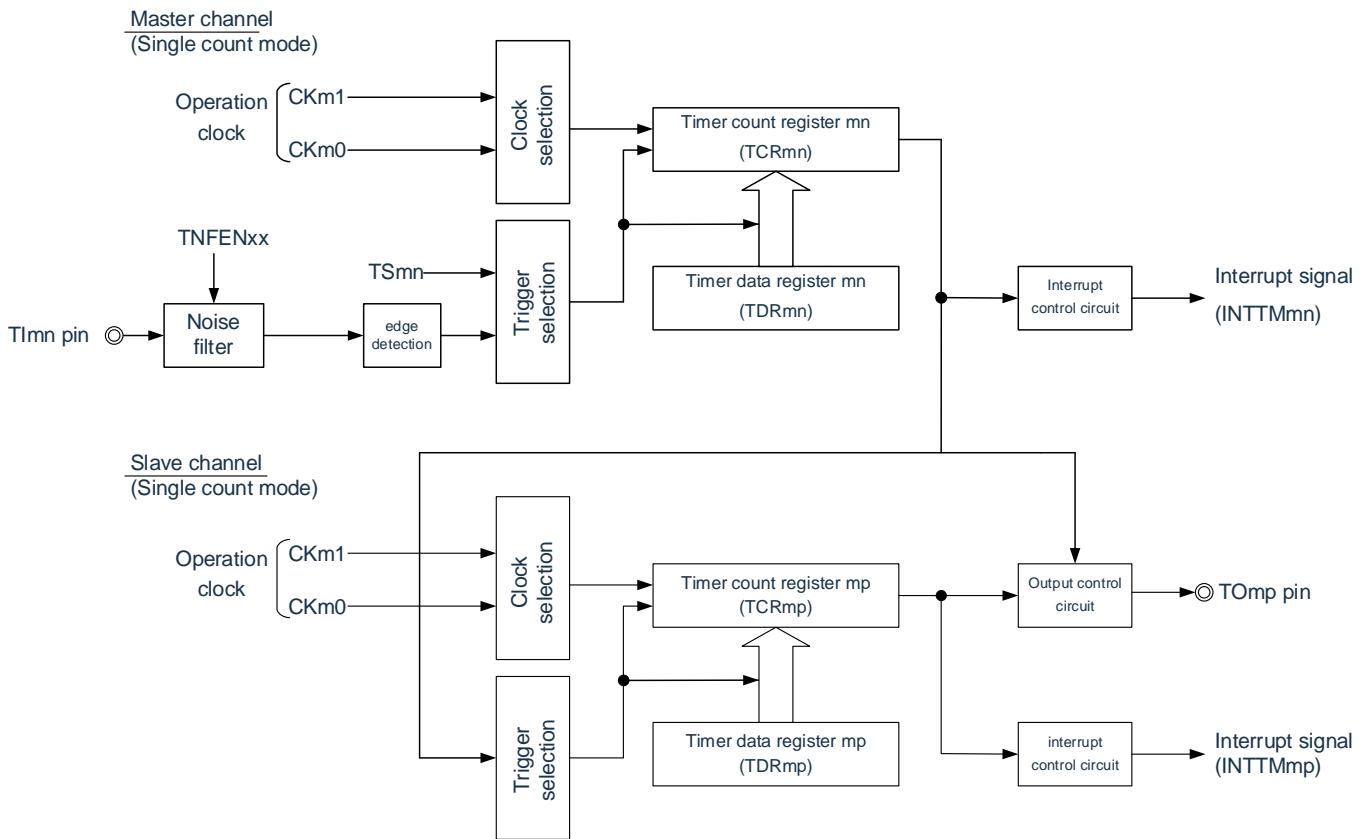
In single count mode, the slave channel operates and counts the pulse width. The INTTMmn of the master channel is used as the start trigger and the TCRmp register of the slave channel is started and loaded with the value of the TDRmp register. The TCRmp register counts decreasingly from the value of 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.

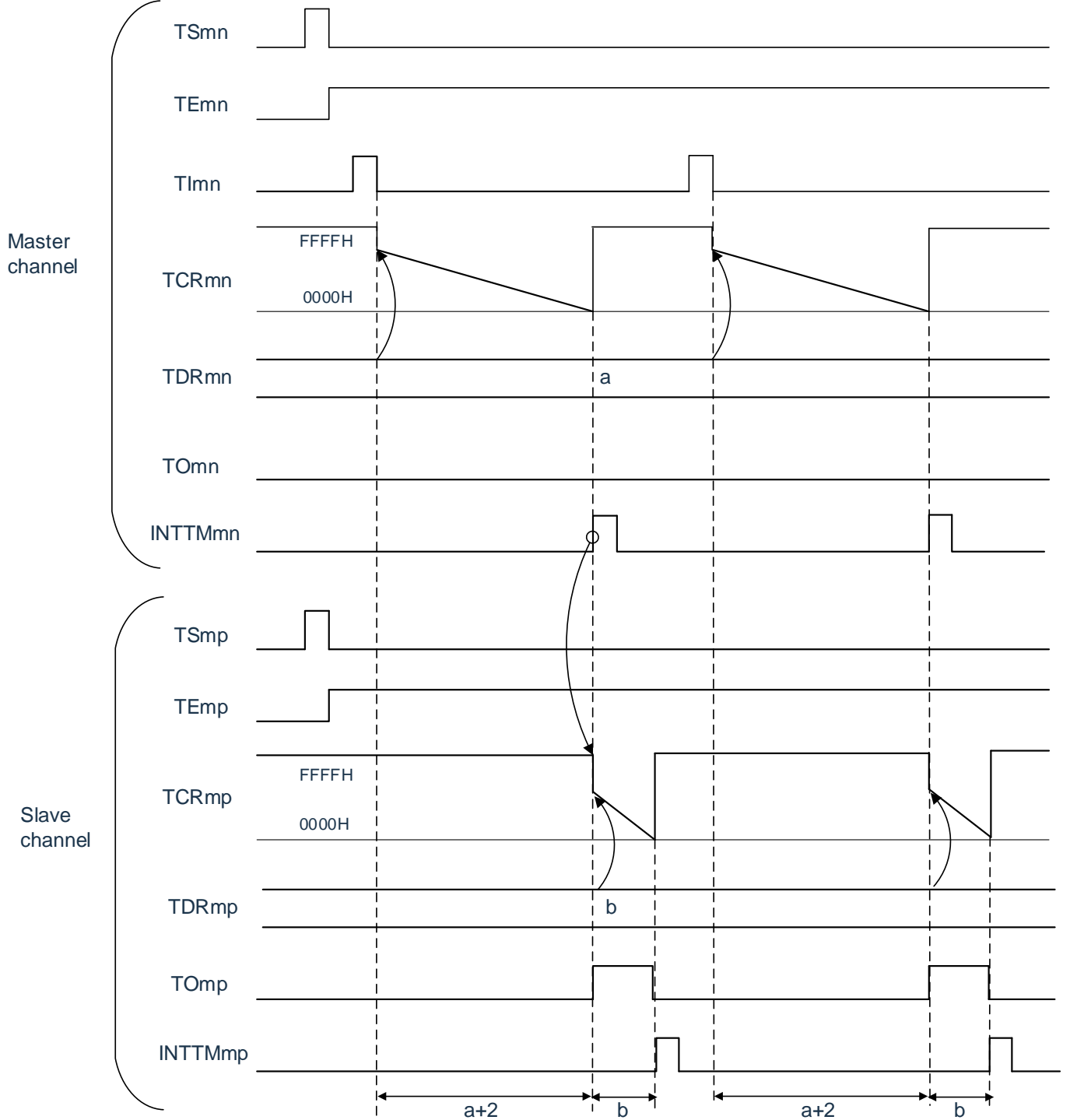
Remark: m: unit number (m=0, 1) n: master channel number (n=0, 2) p: slave channel number (n=0: p=1, 2, 3, n=2: p=3)

Figure 5-61 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 5-62 Example of basic timing operating as a single trigger pulse output function

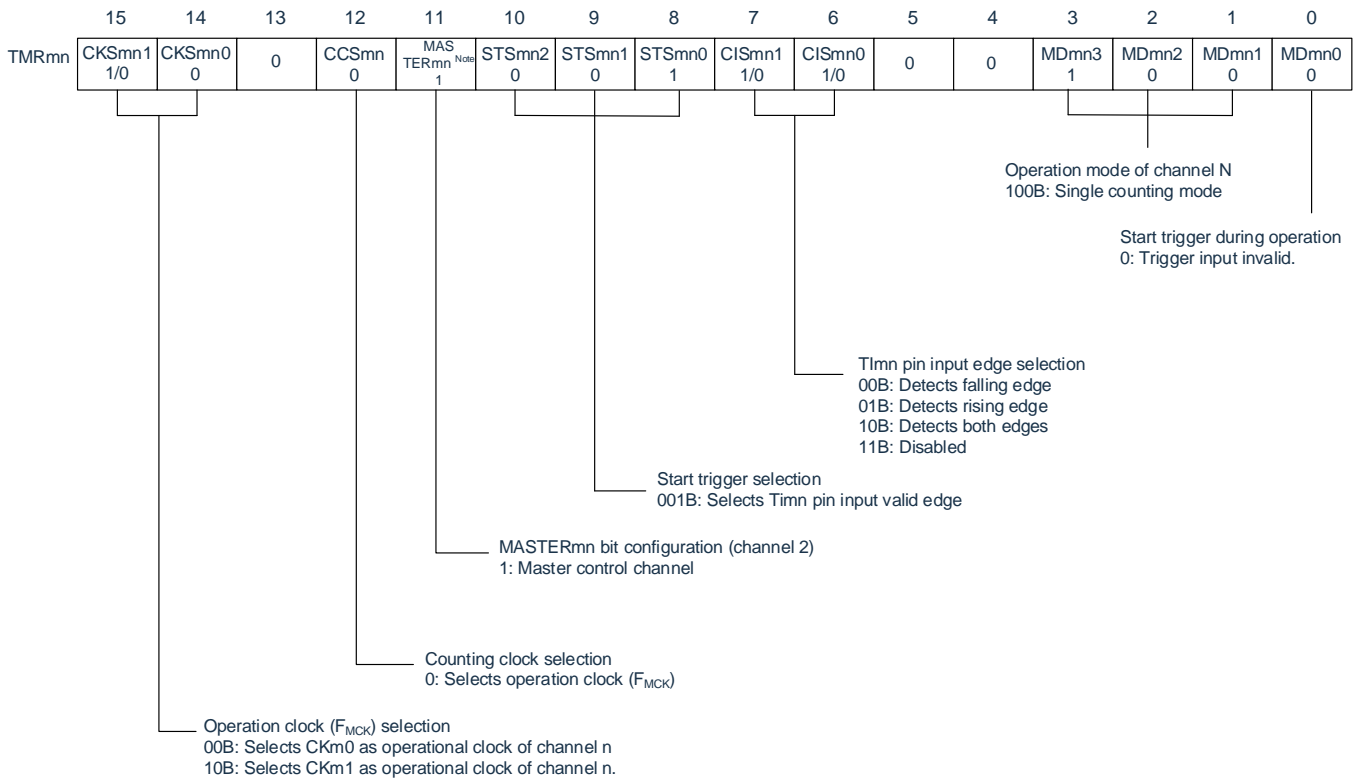


Remark:

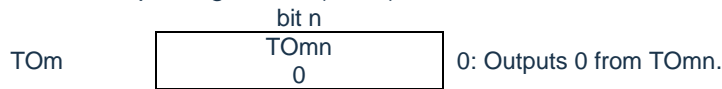
1. m: unit number (m= 0, 1) n: master channel number (n=0, 2) p: slave channel number (n=0: p=1, 2, 3, n=2: p=3)
2. TSmn, TSmp: Bit n, p of timer channel start register m (TSm)
 TEmn, TEmp: Bit n, p of timer channel enable status register m (TEm)
 TImn, TImp: Input signals of TImn pin and TImp pin
 TCRmn, 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 5-63 Example of register contents setting for single trigger pulse output function (master channel)

(a) Timer mode register mn (TMRmn)



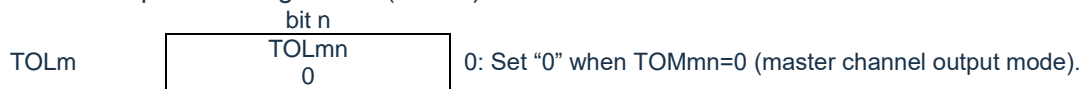
(b) Timer output register m (TOM)



(c) Timer output enable register m (TOEm)



(d) Timer output level register m (TOLm)



(e) Timer output mode register m (TOMm)



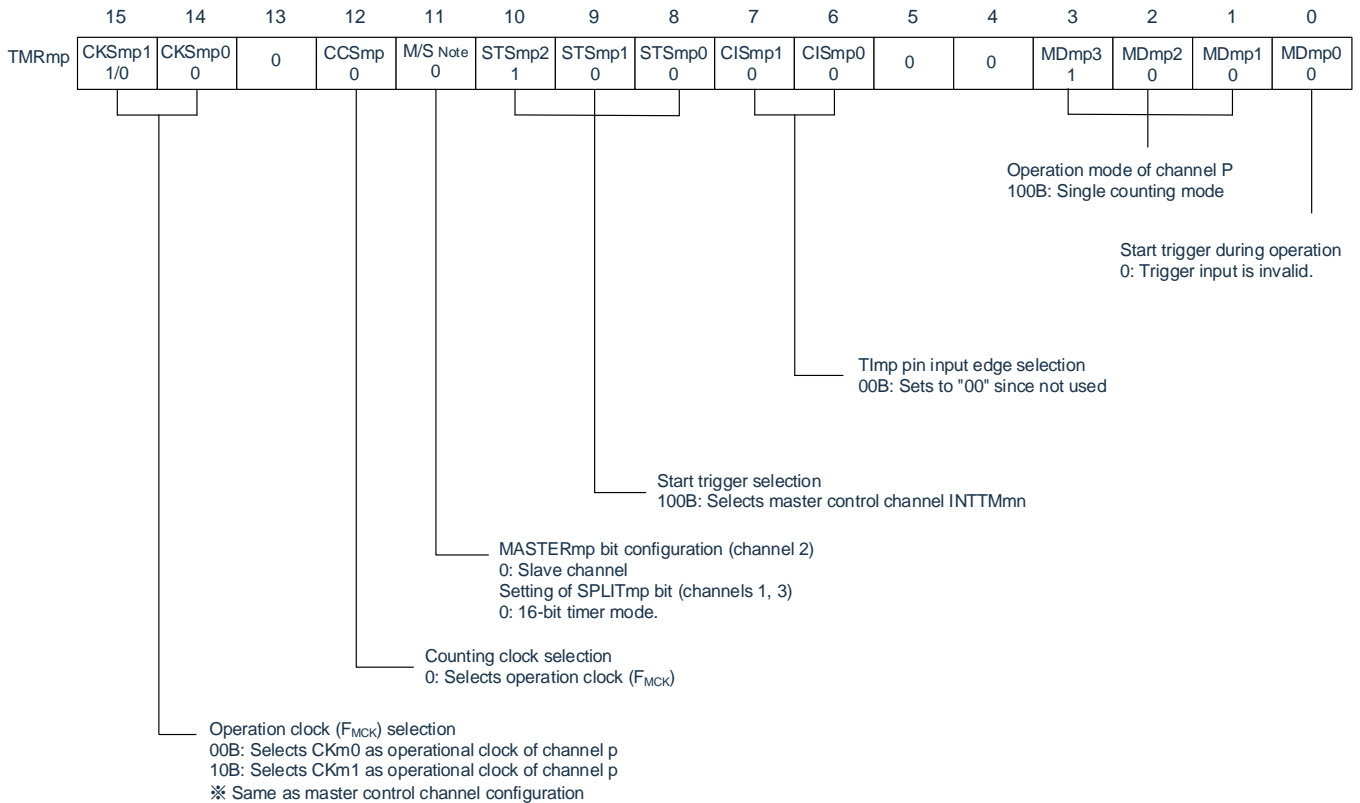
Note TMRm2: MASTERmn=1

TMRm0: Fixed to "0".

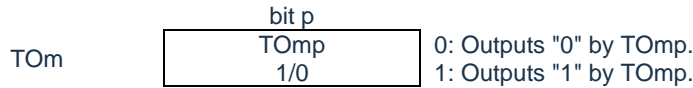
Remark m: unit number (m= 0, 1) n: master channel number (n=0, 2)

Figure 5-64 Example of register contents setting for single trigger pulse output function (slave channel)

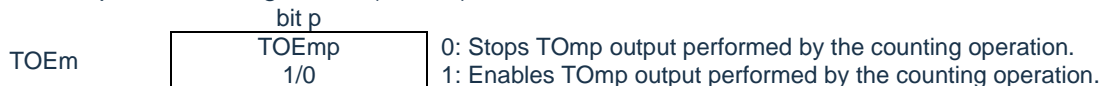
(a) Timer mode register mp (TMRmp)



(b) Timer output register m (TOM)



(c) Timer output enable register m (TOEm)



(d) Timer output level register m (TOLm)



(e) Timer output mode register m (TOMm)



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)

Figure 5-65 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 supply state. (Stop to supply 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 to supply 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.

Figure 5-66 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.

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

$n < p < q \leq 3$ (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:

$$\begin{aligned} \text{Pulse period} &= \{\text{TDRmn (master) set value} + 1\} \times \text{counting clock period} \\ \text{Duty cycle [\%]} &= \{\text{TDRmp (slave) set value}\} / \{\text{TDRmn (master) set value} + 1\} \times 100 \\ \text{0\% output: TDRmp (slave) set value} &= 0000\text{H} \\ \text{100\% output: TDRmp (slave) set value} &\geq \{\text{TDRmn (master) set value} + 1\} \end{aligned}$$

Remark: When the set value of TDRmp (slave) > {Set value of TDRmn (master) + 1}, the duty cycle exceeds 100% but is 100% output.

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.

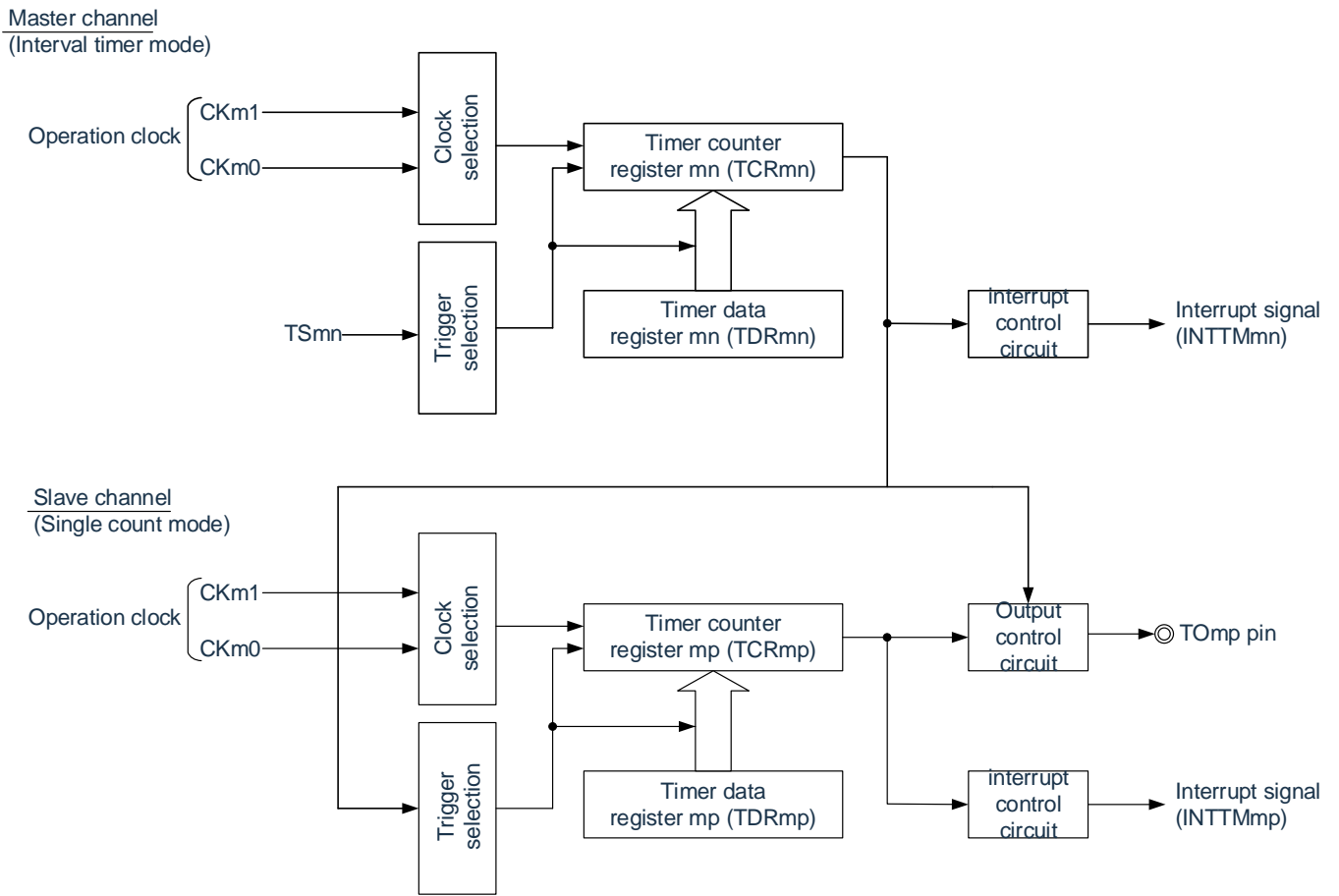
When used as PWM function, the slave channel decrements the count and the duty cycle of the PWM output (TOmp) for the period until "0000H" is counted.

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 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.

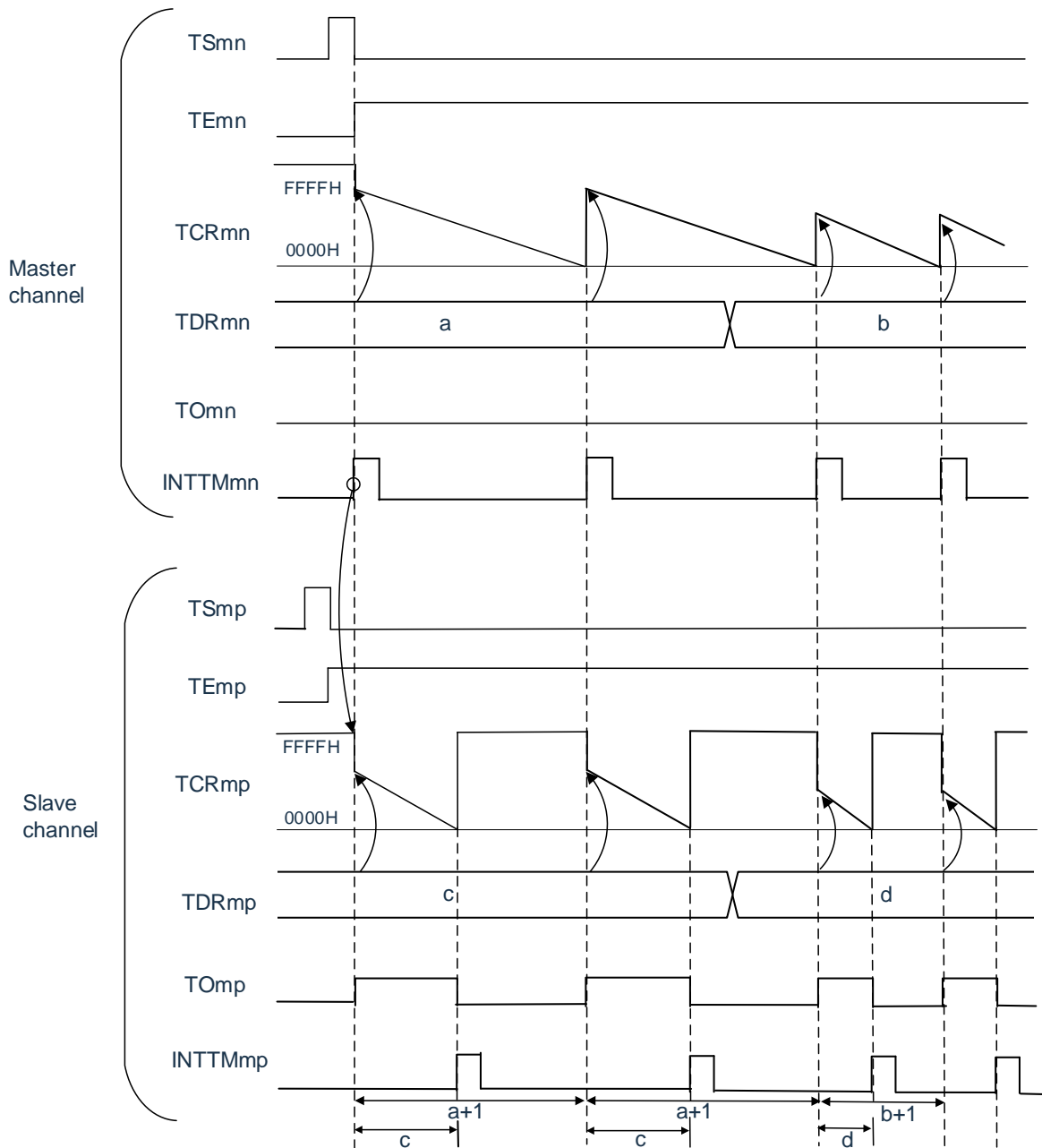
Remark: m: unit number (m= 0, 1) n: master channel number (n=0, 2) p: slave channel number (n=0: p=1, 2, 3, n=2: p=3)

Figure 5-67 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 5-68 Example of basic timing operating as PWM function

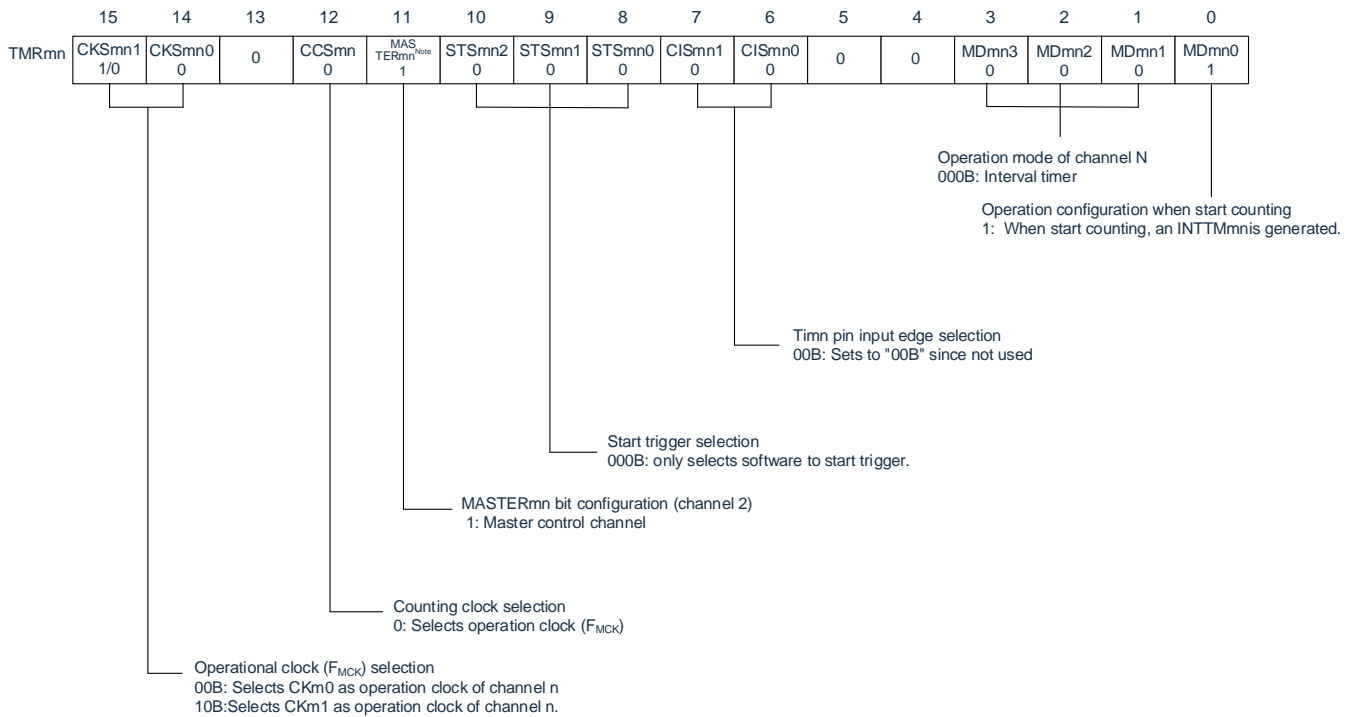


Remark:

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, TEmmp: 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 5-69 Example of register contents setting for PWM function (master channel)

(a) Timer mode register mn (TMRmn)



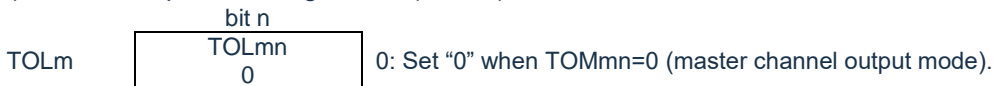
(b) Timer output register m (TOM)



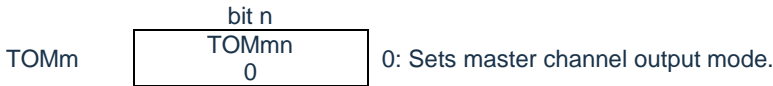
(c) Timer output enable register m (TOEm)



(d) Timer output level register m (TOLm)



(e) Timer output mode register m (TOMm)



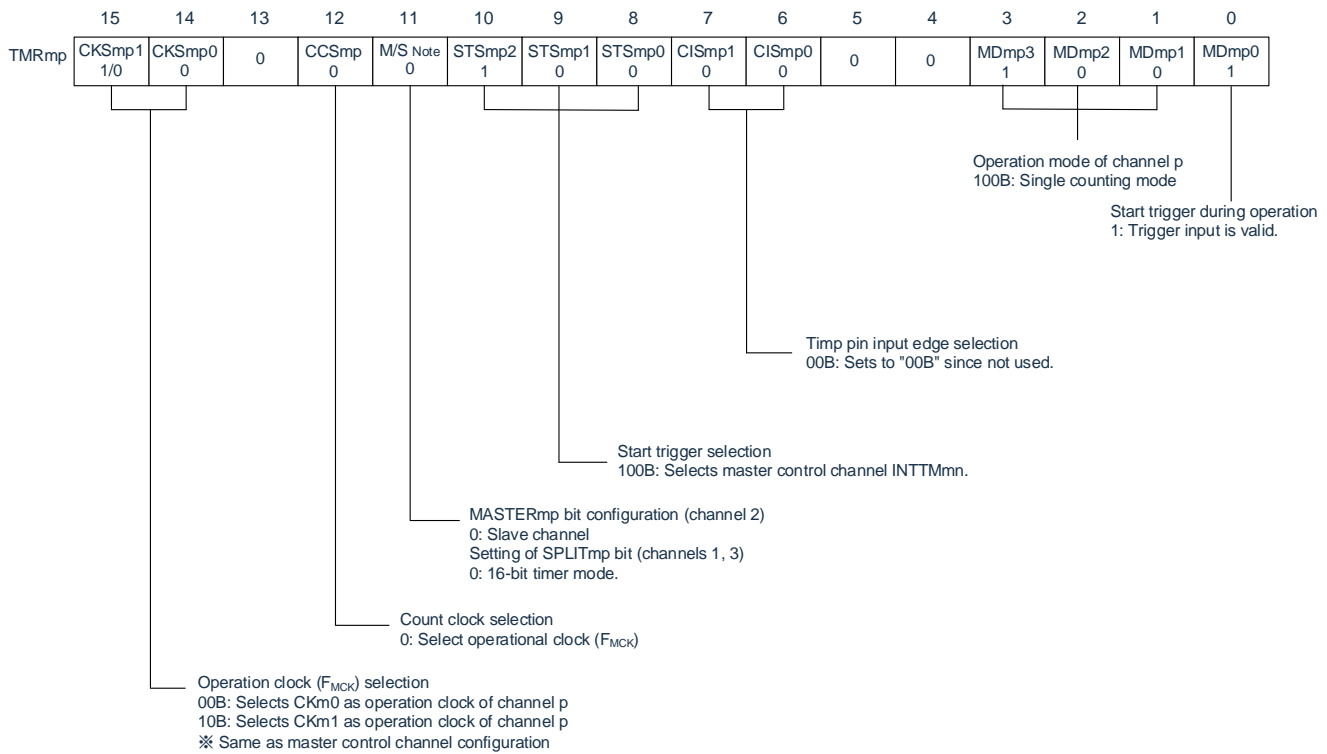
Note: TMRm2: MASTERmn=1

TMRm0: Fixed to "0".

Remark: m: unit number (m=0, 1) n: master channel number (n=0, 2)

Figure 5-70 Example of register contents setting for PWM function (slave channel)

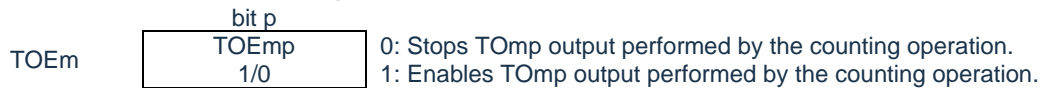
(a) Timer mode register mp (TMRmp)



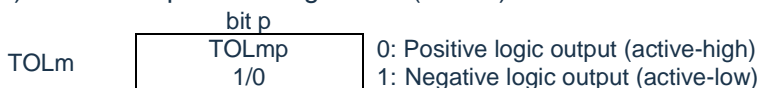
(b) Timer output register m (TOM)



(c) Timer output enable register m (TOEm)



(d) Timer output level register m (TOLm)



(e) Timer output mode register m (TOMm)



Note: TMRm2: 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)

Figure 5-71 Procedure for PWM function (1/2)

	Software operation	Hardware status
Timer4 initial settings		The input clock of timer unit m is in the stop supply state. (Stop to supply 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 to supply 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.

Figure 5-72 Procedure for PWM function (2/2)

	Software operation	Hardware status
Restart operation	<p>Start operation</p> <p>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.</p>	<p>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.</p>
	<p>In operation</p> <p>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.</p>	<p>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</p>
	<p>Stop operation</p> <p>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.</p>	<p>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</p>
	<p>Timer4 stop</p> <p>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.</p>	<p>The TOmp pin output level is held by port function.</p>
	<p>Set the TOmp bit of slave channel to "0" and set the value for the TOmp bit.</p>	<p>The TOmp pin outputs the TOmp set level.</p>
	<p>Set the TM4mEN bit of the PER0 register to "0".</p>	<p>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)</p>

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

p: slave channel number q: slave channel number $n < p < q \leq 3$ (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:

$$\begin{aligned} \text{Pulse period} &= \{\text{TDRmn}(\text{master}) \text{ set value} + 1\} \times \text{count clock period} \\ \text{Duty cycle 1[\%]} &= \{\text{TDRmp}(\text{slave 1}) \text{ set value}\} / \{\text{TDRmn}(\text{master}) \text{ set value} + 1\} \times 100 \\ \text{Duty cycle 2[\%]} &= \{\text{TDRmq}(\text{slave 2}) \text{ set value}\} / \{\text{TDRmn}(\text{master}) \text{ set value} + 1\} \times 100 \end{aligned}$$

Remark: 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.

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.

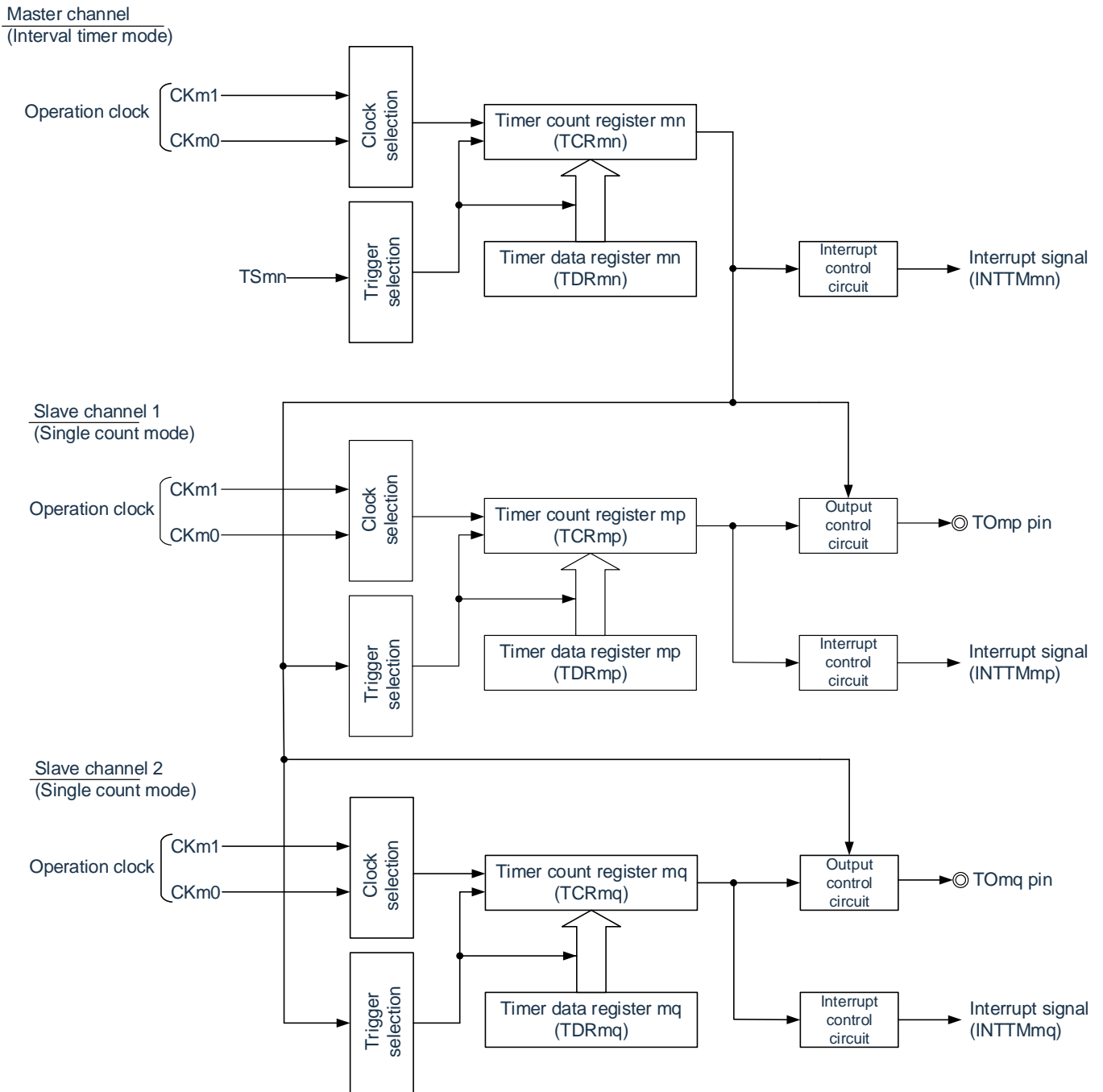
In the same way as the TCRmp register of the slave channel 1, the TCRmq register of the slave channel 2 operates in single count mode, counts the duty cycle, and outputs a PWM waveform from the TOmq pin. The TCRmq register loads the value of the TDRmq register, using INTTMmn of the master channel as 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 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
 $n < p < q \leq 3$ (p and q are integers greater than n)

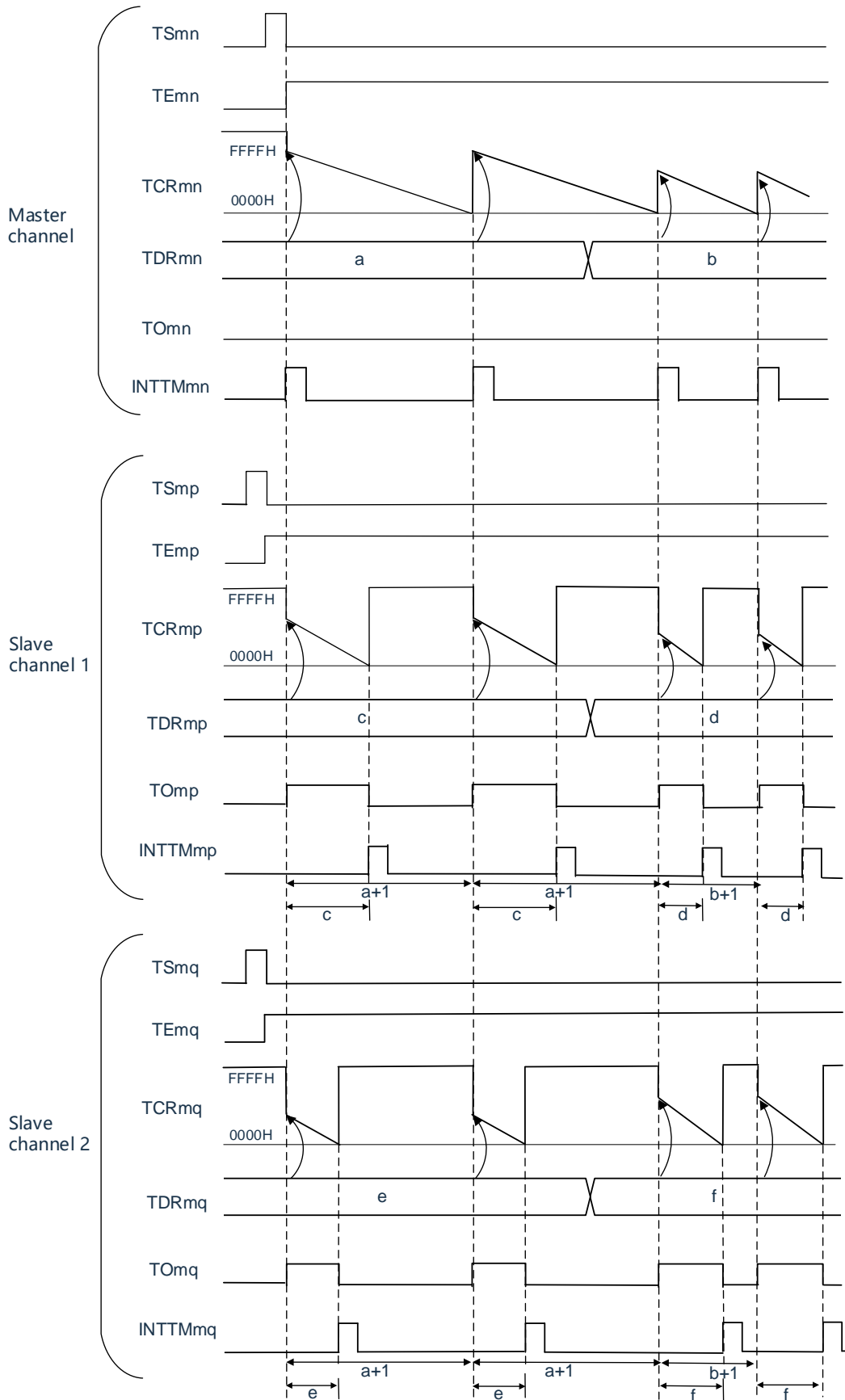
Figure 5-73 Block diagram of operation as multiple PWM output function (output two types of PWMs)



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

p: Slave channel number q: slave channel number $n < p < q \leq 3$ (p and q are integers greater than n)

Figure 5-74 Example of basic timing operating as multiple PWM output function (output two types of PWMs)

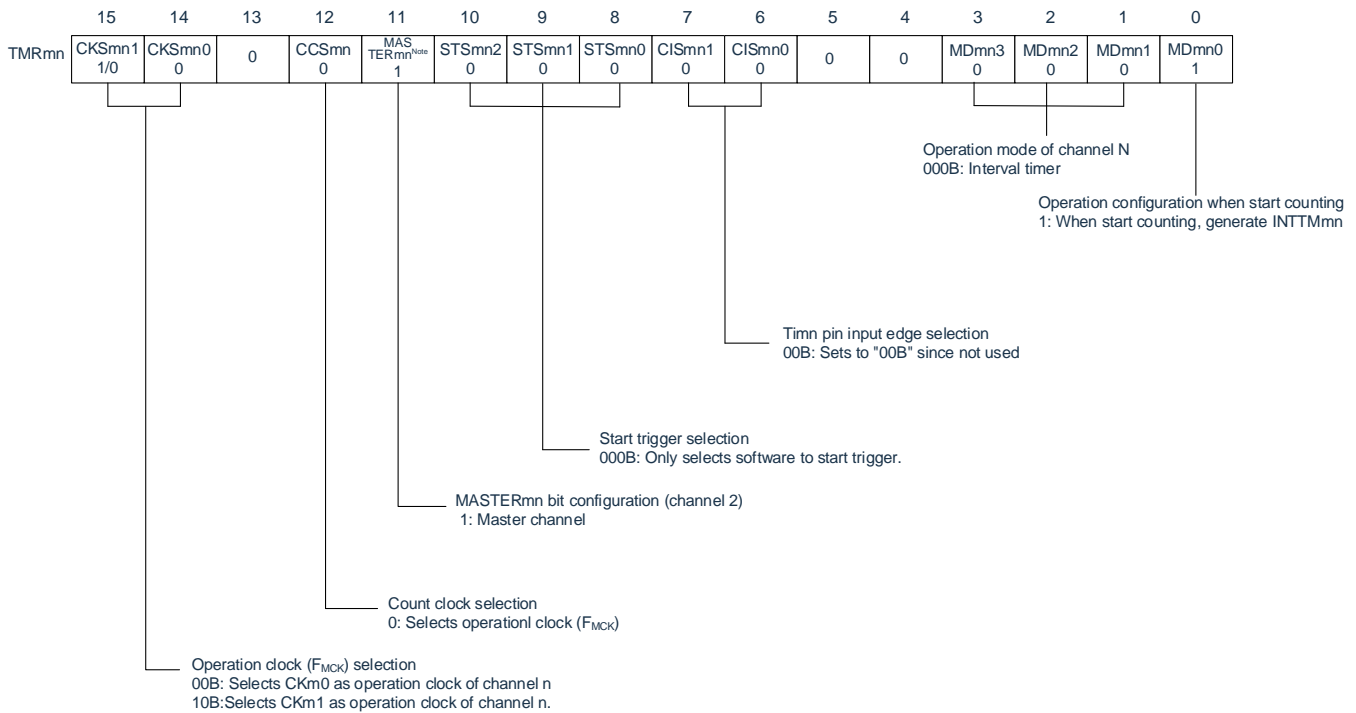


Remark:

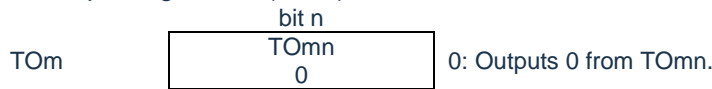
1. m: unit number (m= 0, 1) n: master channel number (n=0)
p: slave channel number q: slave channel number $n < p < q \leq 3$ (p and q are integers greater than n)
2. TSmn, TSmp, TSmq: Bit n, p of timer channel start register m (TSm), q
TEmn, TEmp, TEMq: Bit n, p of timer channel enable status register 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 5-75 Example of register contents setting for multiple PWM output function (master channel)

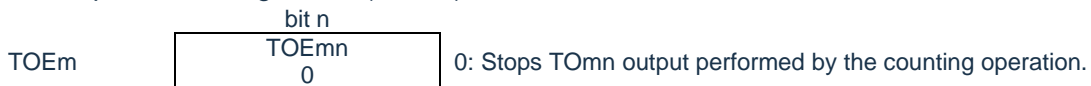
(a) Timer mode register mn (TMRmn)



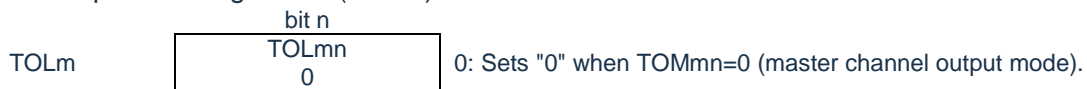
(b) Timer output register m (TOM)



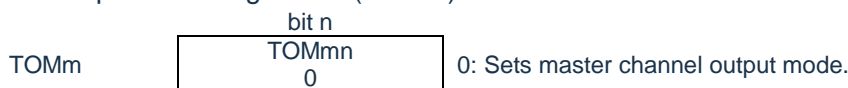
(c) Timer output enable register m (TOEm)



(d) Timer output level register m (TOLm)



(e) Timer output mode register m (TOMm)



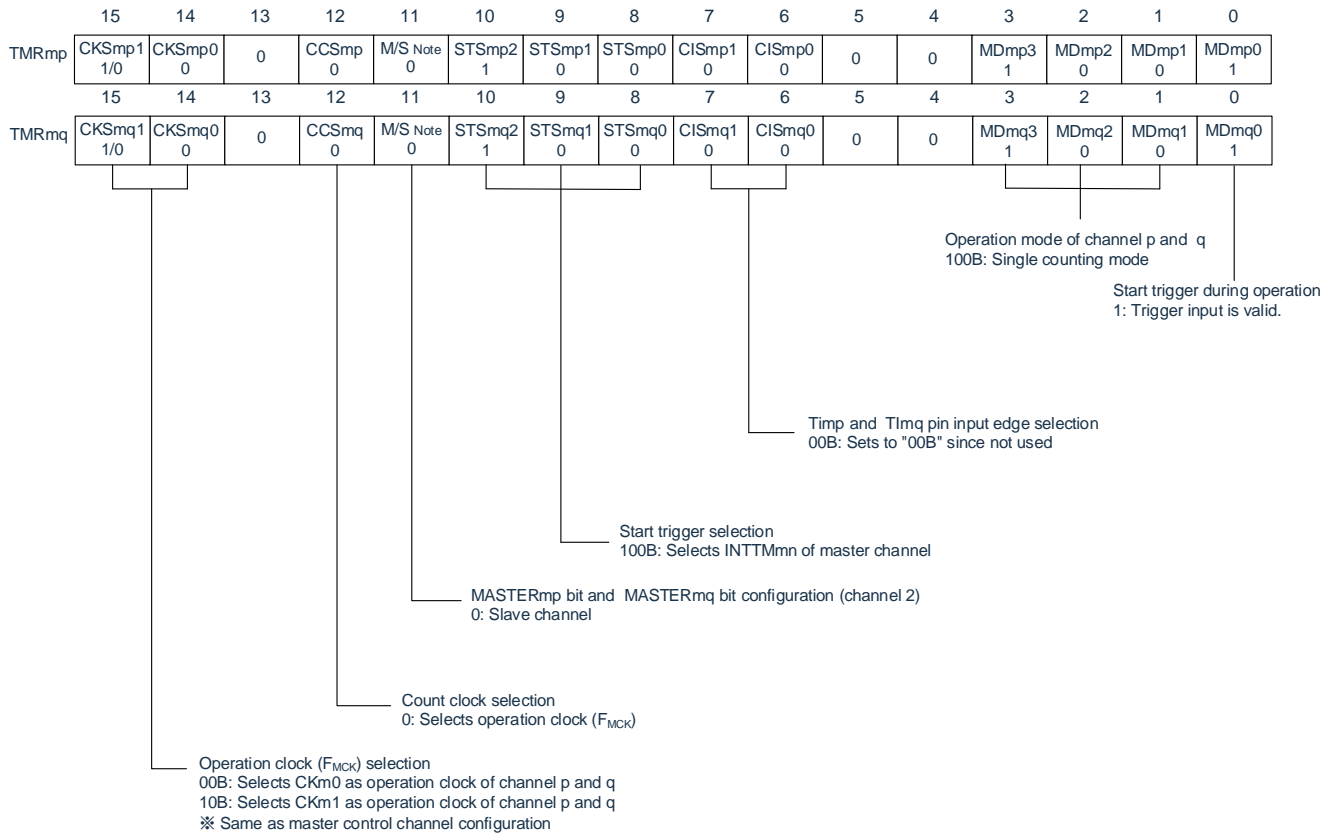
Note: TMRm2: MASTERmn=1

TMRm0: Fixed to "0".

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

Figure 5-76 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	TOELq 1/0	TOELp 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	TOMLq 1	TOMLp 1	1: Sets the slave channel output mode.

Note TMRm2 : MASTERmp bit, MASTERmq bit
TMRm1, TMRm3 : SPLITmp bit, SPLITmq bit

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 5-77: 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 supply state. (Stop to supply 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 to supply 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.	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.
	Set the TOEmp and TOEmq bits to "1" and enable TOmp and TOMq output.	The TOmp pin and TOMq pin output the levels set by the TOmp and TOMq.
	Set the Port Register and Port Mode Register to "0".	

Figure 5-78 Procedure for the multiple PWM output function (output two types of PWMs) (2/2)

	Software operation	Hardware status
Restart operation	<p>Start operation</p> <p>(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.</p>	<p>TEmn = 1, TEmq = 1</p> <p>When the master channel starts counting, INTTMmn is generated. Triggered by this interrupt, the slave channel also starts counting.</p>
	<p>In operation</p> <p>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.</p>	<p>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.</p>
	<p>Stop operation</p> <p>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.</p>	<p>TEmn, 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.</p>
	<p>The TOEmp and TOEmq bits of slave channels are cleared to 0 and value is set to the TOmp and TOmq bits</p>	<p>The TOmp and TOmq pins output the TOmp and TOmq set levels.</p>
	<p>Timer4 stop</p> <p>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 TM4mEN bit of the PER0 register is cleared to 0.</p>	<p>The TOmp and TOmq pin output levels are held by port function.</p> <p>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)</p>

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

p: slave channel number q: slave channel number $n < p < q \leq 3$ (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

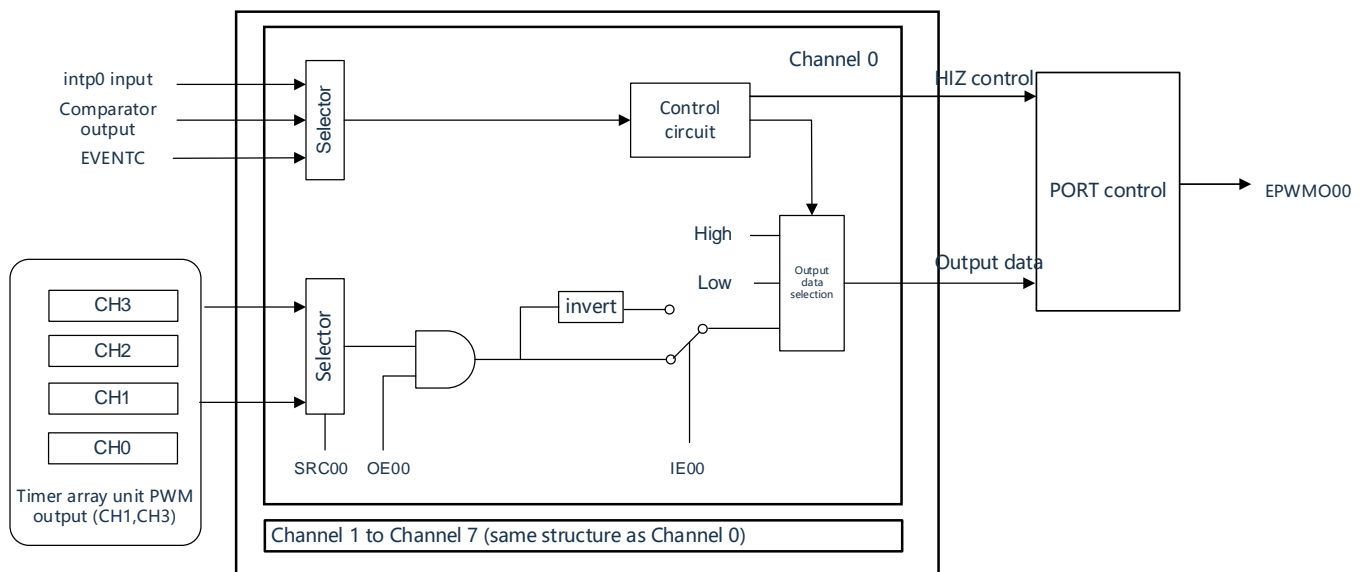
The EPWM output control circuit consists of the following hardware.

Table 6-1 Structure of EPWM output control circuit

Item	Structure
Control registers	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)
Output	EPWM output (EPWMO00~EPWMO07)

Block diagram of the EPWM output control circuit is shown in Figure 6-1.

Figure 6-1 Block diagram of 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)
- 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”.

Figure 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 output control register (EPWMCTL)

The EPWMCTL register enables control and reverse control of the waveform outputs of EPWMO00~EPWMO03.

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”.

Figure 6-3 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)

6.2.4 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”.

Figure 6-4 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	No selection
0	1	Comparator 0 output
1	0	INTP0 terminal input
1	1	Event input from EVENTC

IN_EG	Source of output forced truncation /Source of output forced truncation release edge selection ^{Note 1,2}
0	Rising edge: Output force truncation Falling edge: Output force truncation release
1	Rising edge: Output force truncation release 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.

Note 1: Set SC_SEL1 and SC_SEL0 at least three clocks apart after IN_EG is set.

Note 2: Valid only when INTP0 input is selected.

Note 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.

Note 4: The effective width of the select comparator 0 output and INTP0 input must be greater than one clock cycle.

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”.

Figure 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

IO _n 1	IO _n 0	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 truncancy 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”.

Figure 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

Notice: If truncation is prohibited by setting the forced truncated 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 Registers controlling 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 (EPWMO 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, please refer to “2.5 Register Settings for Multiplexing Function”.

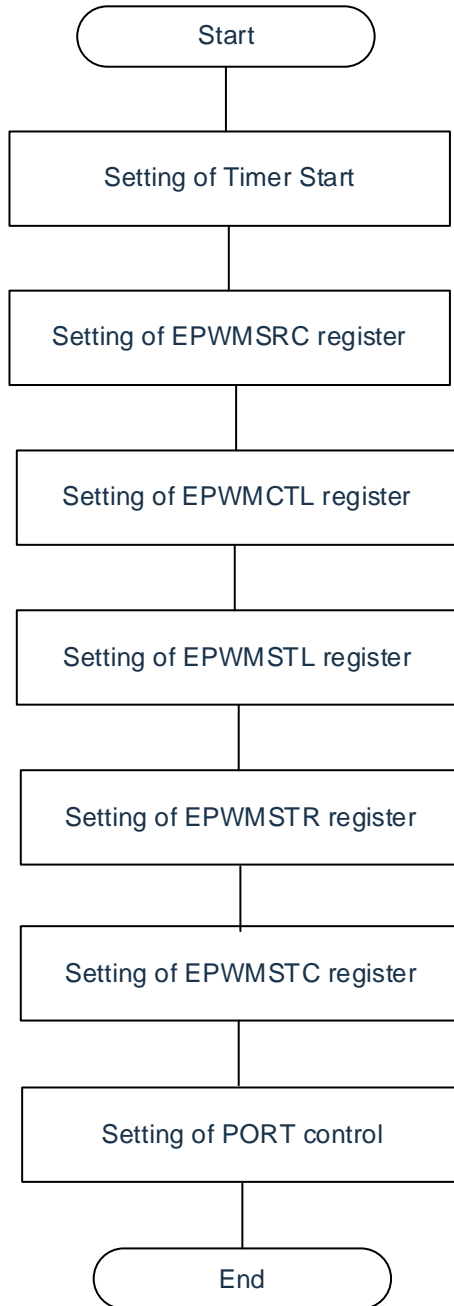
6.3 Operation of EPWM output control circuit

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 6-7 Initial configuration flow of registers

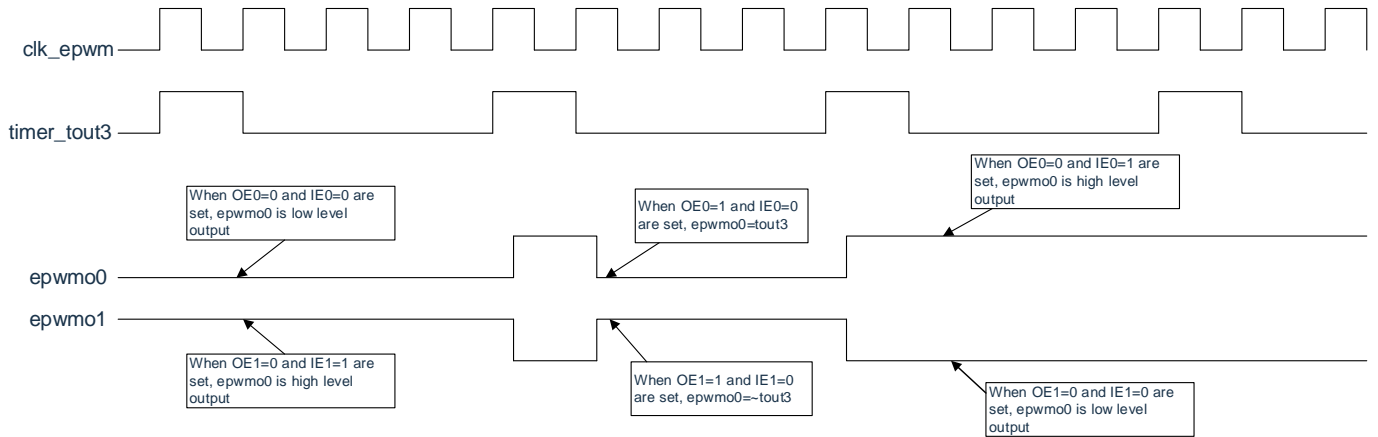


6.3.2 Normal operation

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 6-2 Operation Instructions for truncation signals”.

Figure 6-8 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 6-9.

(2) Release of forced truncation

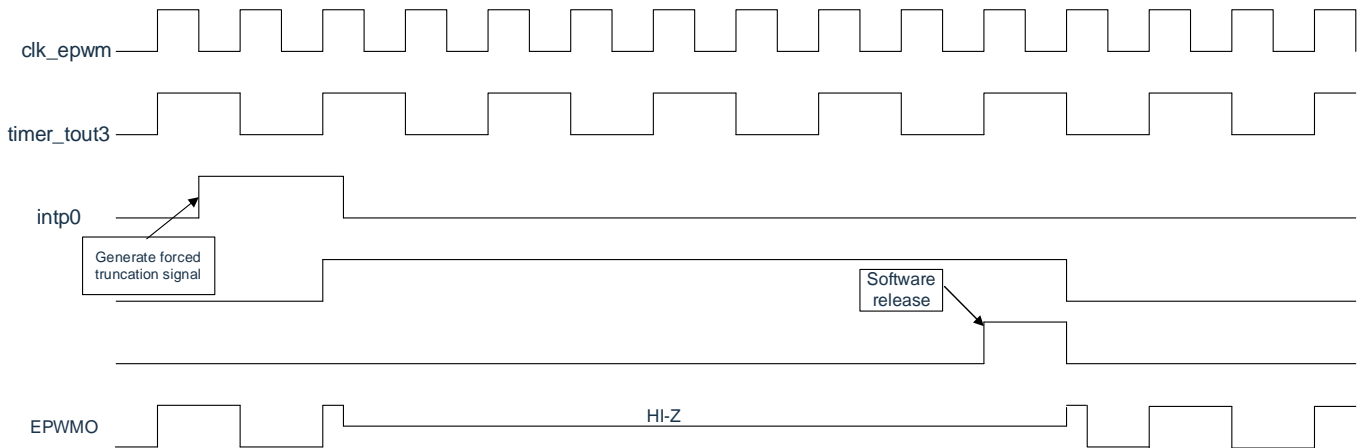
- a) Software release: When bit3 (HS_SEL) of EPWMSTC register is 0, the software release mode is used. Bit0 (HZCLR) of EPWMSTR register is the clear bit of truncated status. When the truncated status flag SHTFLG is high, if the HZCLR bit is set to “1”, the truncated status flag SHTFLG goes low and the 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 6-2 Table of operation Instructions for truncation signals

Bit	IO _n 1-0	OE ₀ n	IE ₀ n	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 6-9 Timing diagram for generation and release of INTP0 truncation (HS_SEL=0, REL_SEL=0)



Notice: 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.

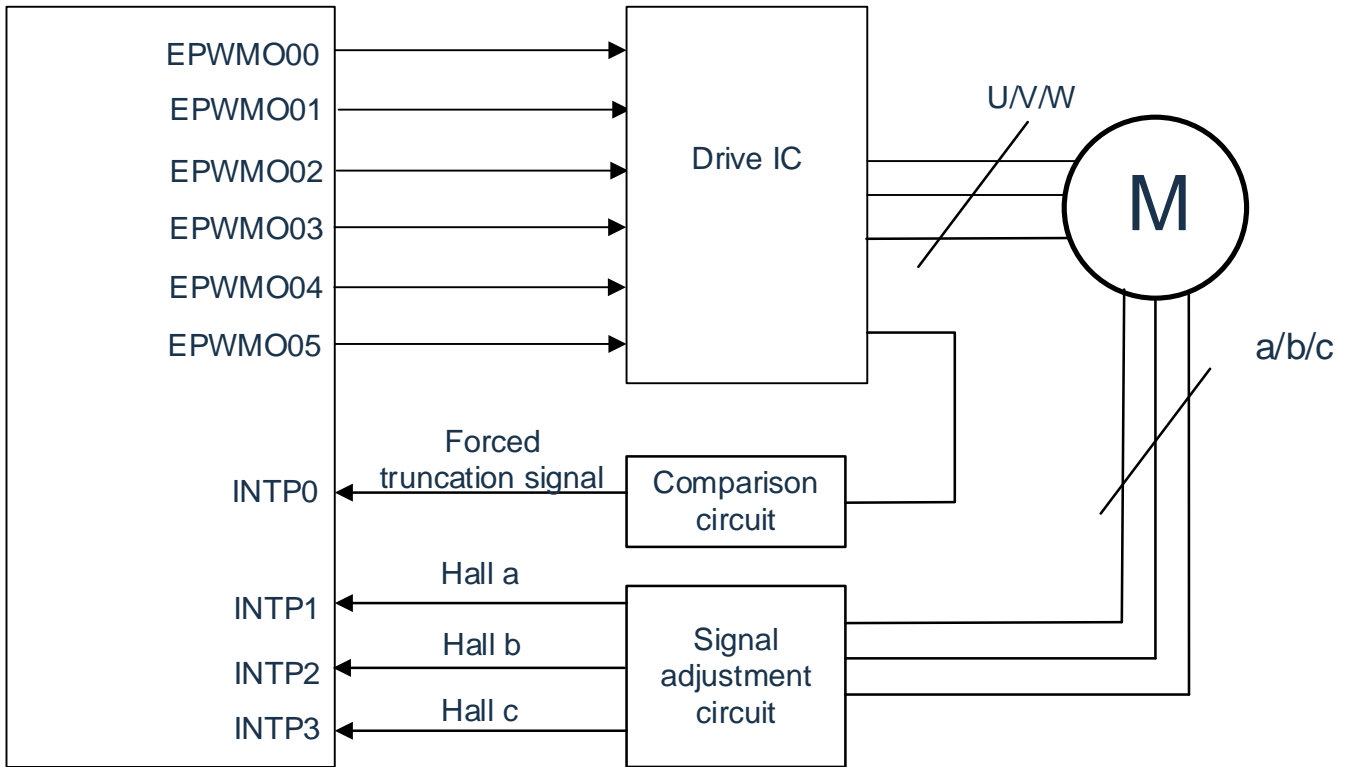
6.4 Control example of brushless DC motor

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

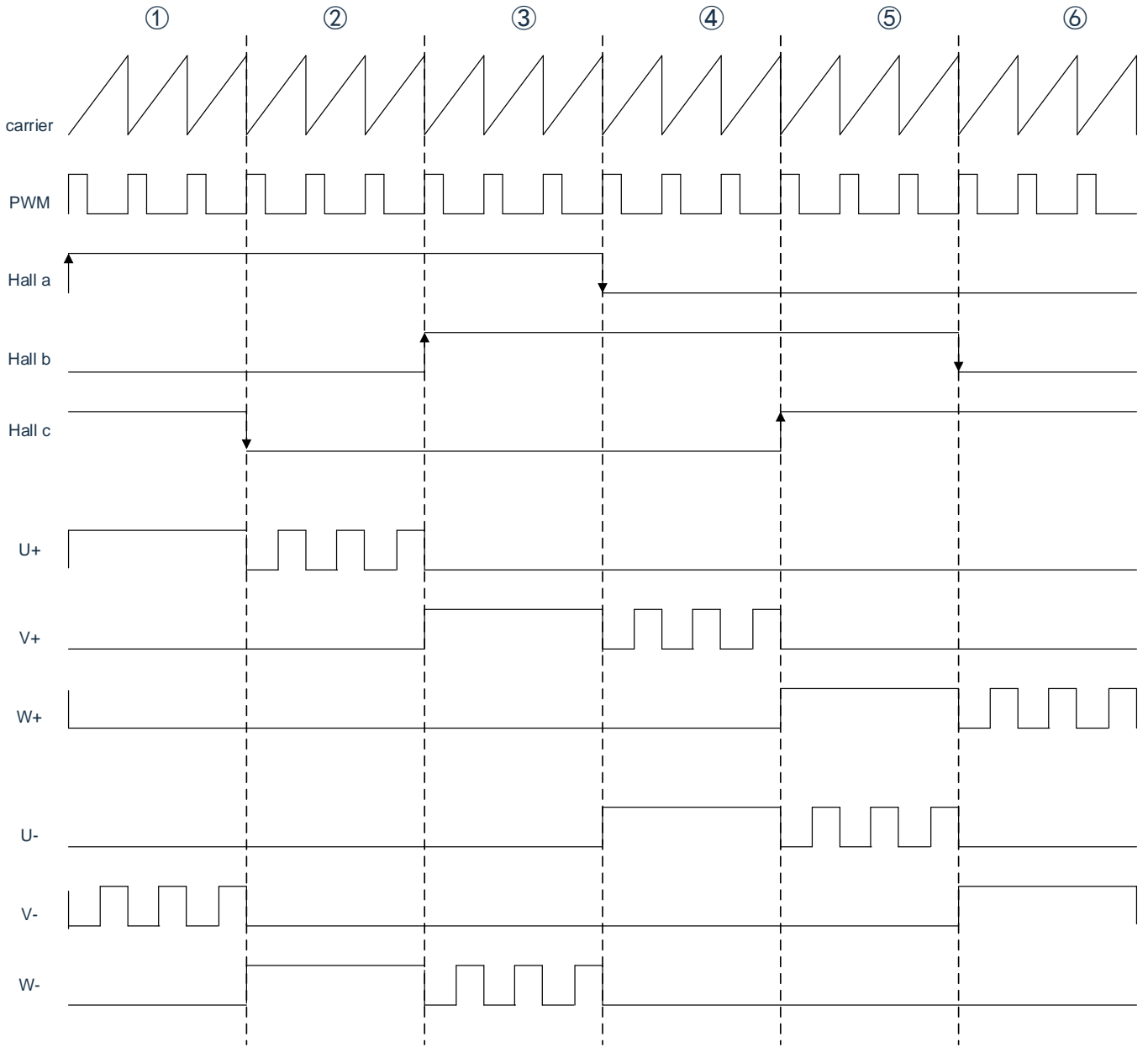
An example of a hardware connection for a brushless DC motor is shown in Figure 6-10. In this example, EPWMO00~EPWMO05 (output) is used for output control of BLDC motors, INTP1~INTP3 (input) are output signals for the Hall sensor, and INTP0 (input) is used to force a truncated signal.

Figure 6-10 Example of hardware connection



6.4.2 Control timing of three-phase brushless DC motors

Figure 6-11 Control timing of three-phase brushless DC motor



6.4.3 Examples 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 ~ EPWMO0 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 EPWMO4 to be output. Set EPWMIE5 ~ EPWMIE4 in the EPWMCTL register to “1” to reverse the output of EPWMO04~ EPWMO05.

Table 6-4 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

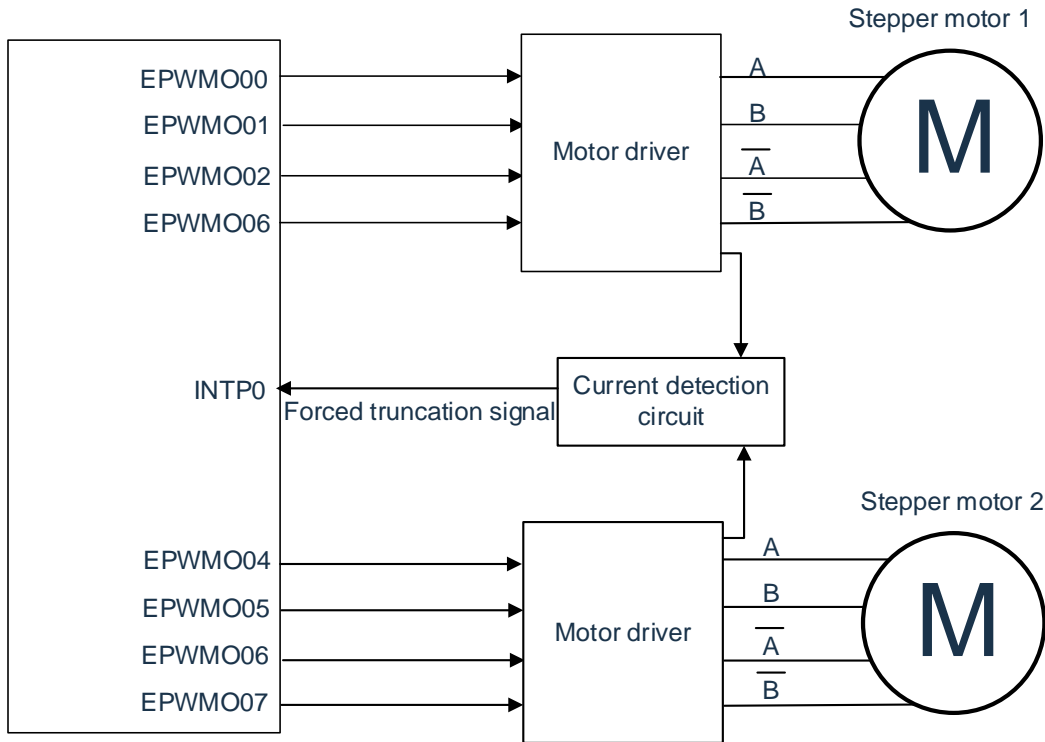
6.5 Example of stepper motor control

The following is an example of using eight real-time outputs to control two 2-phase stepper motors.

6.5.1 Example of hardware connection

Figure 6-12 shows hardware connection to control two stepper motors.

Figure 6-12 Example of hardware connection



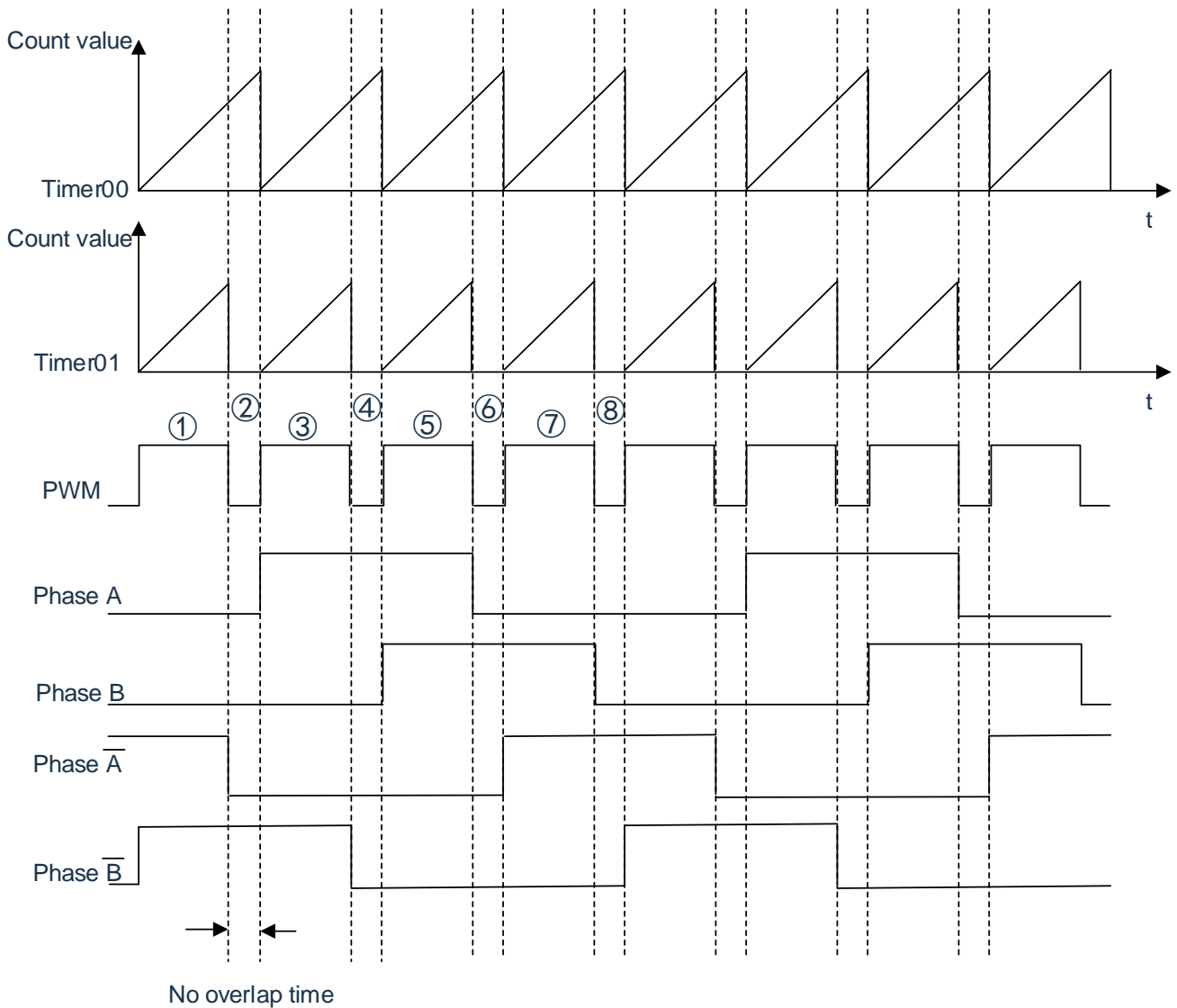
6.5.2 Control method

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 6-13.

Figure 6-13 Waveform example of stepper motor control



6.5.3 Example of register setting

Table 6-5 Example of setting registers 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

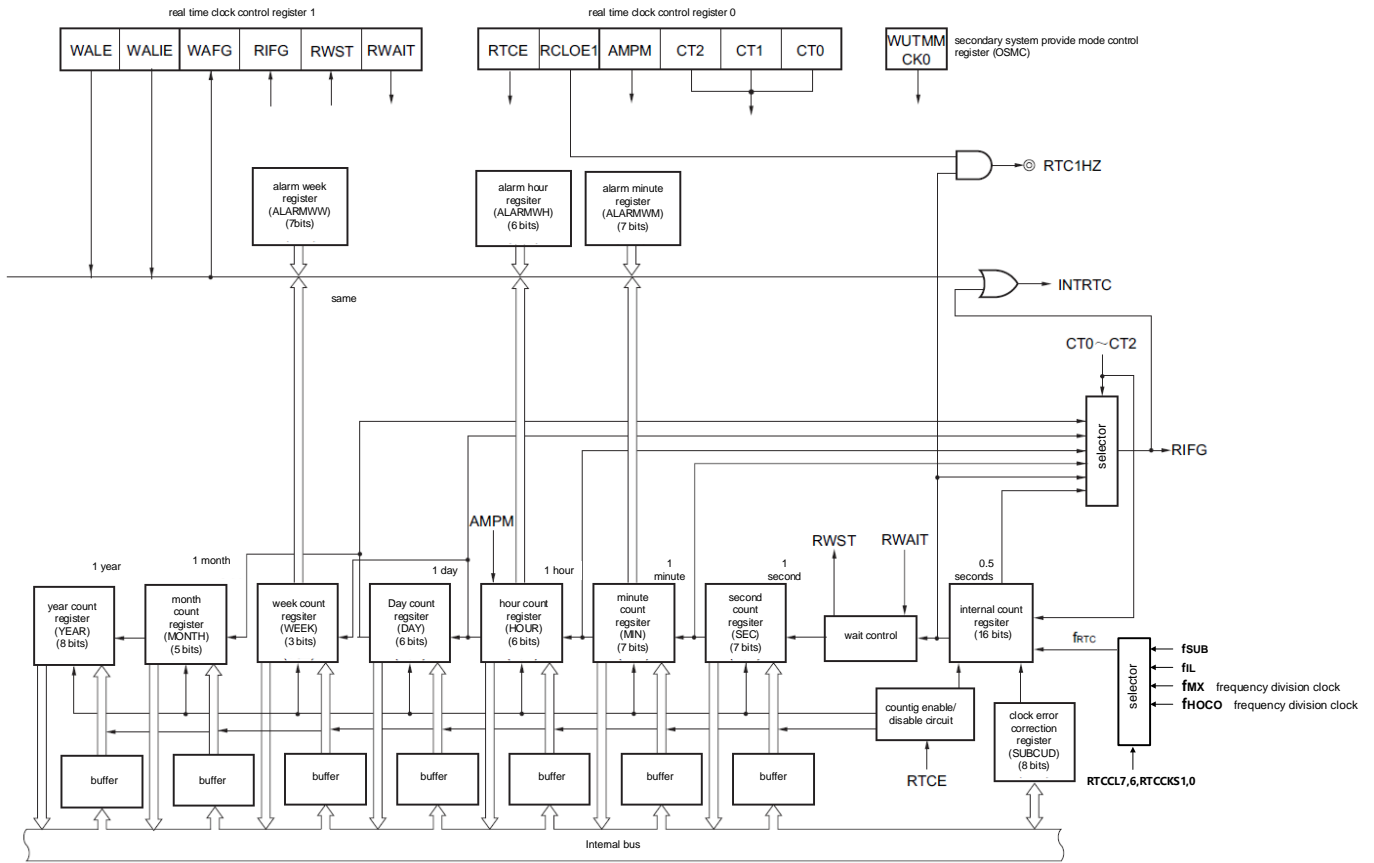
The real-time clock consists of the following hardware.

Table 7-1 Structure of real-time clock

Item	Structure
Counter	Internal counter (16-bit)
Control register	Peripheral enable register 0 (PER0.bit7)
	Real-time clock selection register (RTCCCL)
	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)	

Note: 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 ($\approx 32, 768\text{KHZ}$ after every week) or the secondary system clock ($F_{SUB}=32.768\text{kHz}$) as the running clock for the real-time clock. When a low-speed internal oscillator clock ($F_{IL}=15\text{kHz}$) 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:

$$\text{Fixed period (value selected by the RTCC0 register)} \times F_{SUB}/F_{IL}$$

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 (RTCCCL)
- 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 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.

Bit7 (RTCEN) must be set to "1" when using 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".

Figure 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	IRDAEN	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. <ul style="list-style-type: none"> • SFR used by the real-time clock (RTC) and the 15-bit interval timer cannot be written. • The real-time clock (RTC) and the 15-bit interval timer are in the reset status.
1	Supply the input clock. <ul style="list-style-type: none"> • SFR used by the real-time clock (RTC) and the 15-bit interval timer can be read and written.

Notice:

- 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)
- 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.

Figure 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	RTCKS1	RTCKS0

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})

RTCKS1	RTCKS0	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".

Figure 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 11-2.

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.

Notice1. When the RTCE bit is "1", the RCLOE1 bit cannot be changed.

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".

Figure 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 (ALARMVM), 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.	

Figure 7-5 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 at least 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.

Remark:

- 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).
- 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".

Figure 7-6 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.	
<ul style="list-style-type: none"> • 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 clock error correction value
0	Increases by $\{(F11, F10, F9, F8, F7, F6, F5, F4, F3, F2, F1, F0) - 1\} \times 2$
1	Decreases by $\{(\overline{F11}, \overline{F10}, \overline{F9}, \overline{F8}, \overline{F7}, \overline{F6}, \overline{F5}, \overline{F4}, \overline{F3}, \overline{F2}, \overline{F1}, \overline{F0}) + 1\} \times 2$
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	

Notice: "/" 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

Remark: 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”.

Figure 7-7 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 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”.

Remark 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”.

Figure 7-8 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”.

Figure 7-9 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 7-2 shows the relationship between the setting value of the AMPM bit, the hour count register (HOUR) value, and time.

Table 7-2 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”.

Figure 7-10 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".

Figure 7-11 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

Notice1. The value corresponding to the month count register (MONTH) or the day count register (DAY) is not stored in the week count register (WEEK) automatically. After reset release, set the week count register as follow:

Day	WEEK
Sunday	00H
Monday	01H
Tuesday	02H
Wednesday	03H
Thursday	04H
Friday	05H
Saturday	06H

2. When 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".

Figure 7-12 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".

Figure 7-13 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 (ALARMWWM)

This register is used to set minutes of alarm.

The ALARMWWM register can be set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "00H".

Notice 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.

Figure 7-14 Format of alarm minute register (ALARMWWM)

Address: 0x40044F5AH	After reset: 00H R/W							
Symbol	7	6	5	4	3	2	1	0
ALARMWWM	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.

Figure 7-15 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 operation instruction. After a reset signal is generated, the value of this register becomes “00H”.

Figure 7-16 Format of alarm week register (ALARMWW)

Address: 0x40044F5CH

After reset: 00H 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	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Hour 10	Hour 1	Minute 10	Minute 1	Hour 10	Hour 1	Minute 10	Minute 1
	W	W	W	W	W	W	W								
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

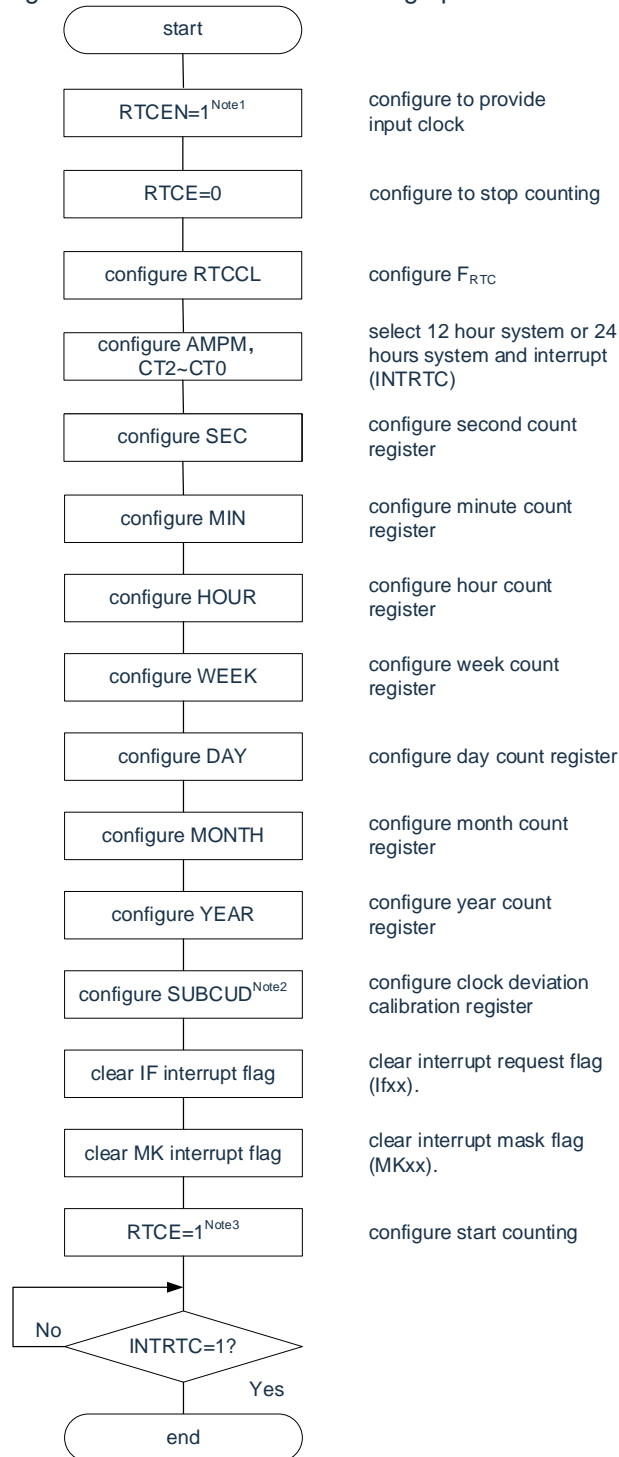
7.3.16 Port mode register and port register

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 “2.5 Register settings when using multiplexing function”.

7.4 Operation of real-time clock
7.4.1 Start of real-time clock operation

Figure 7-17 Procedure for starting operation of real-time clock



Note1. Set the RTCEN bit to 1, while oscillation of the count clock (F_{RTC}) is stable.

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”.

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 $RTCE = 1$.

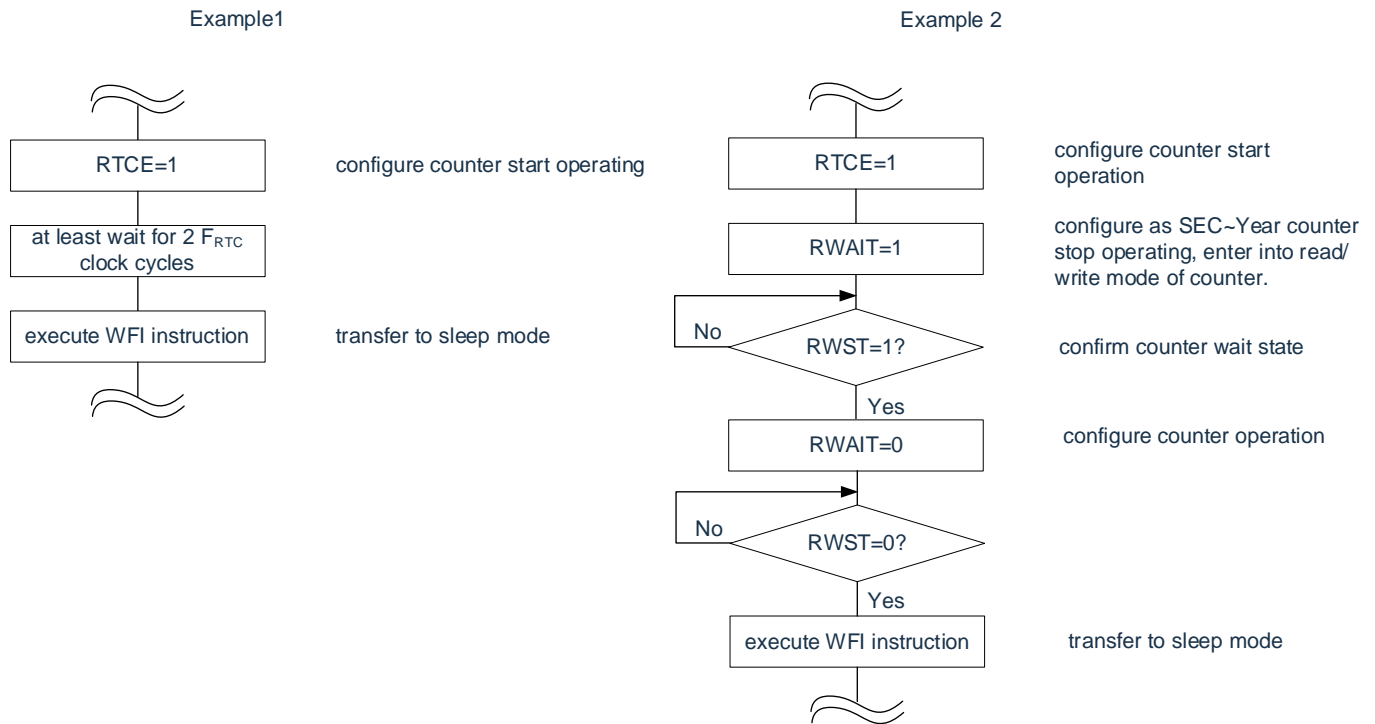
7.4.2 Shifting to sleep mode after starting operation

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 (FRTC) have elapsed after setting the RTCE bit to 1 (see Figure 7-18, 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 7-18, Example 2).

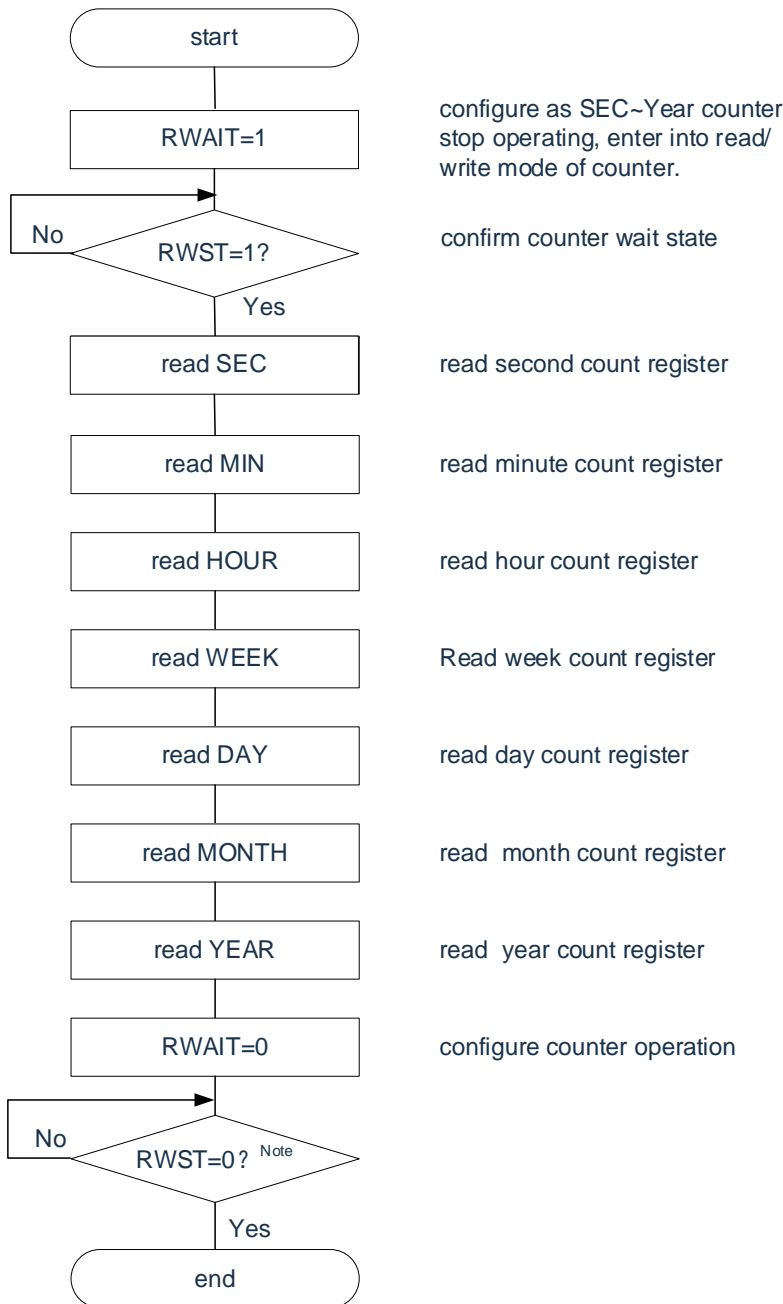
Figure 7-18 Procedure for shifting to sleep/deep sleep mode after setting RTCE bit to 1



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 7-19 Procedure for reading real-time clock

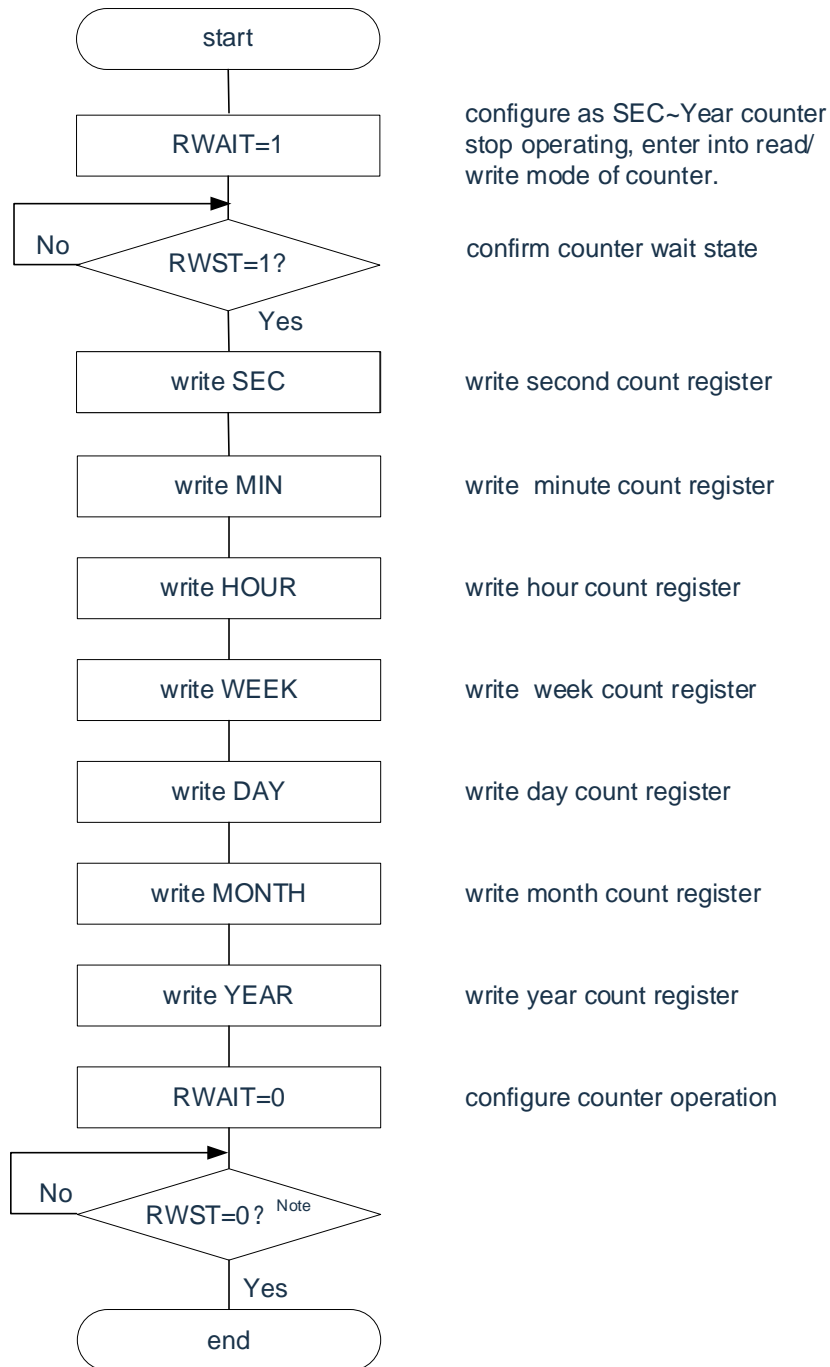


Note Be sure to confirm that RWST = 0 before setting sleep mode.

Notice Complete the series of process of setting the RWAIT bit to 1 to clearing the RWAIT bit to 0 within 1 second.

Remark: 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 7-20 Procedure for reading real-time clock



Note: Be sure to confirm that RWST = 0 before setting sleep mode.

Notice1. Complete the series of operations of setting the RWAIT bit to 1 to clearing the RWAIT bit to 0 within 1 second.

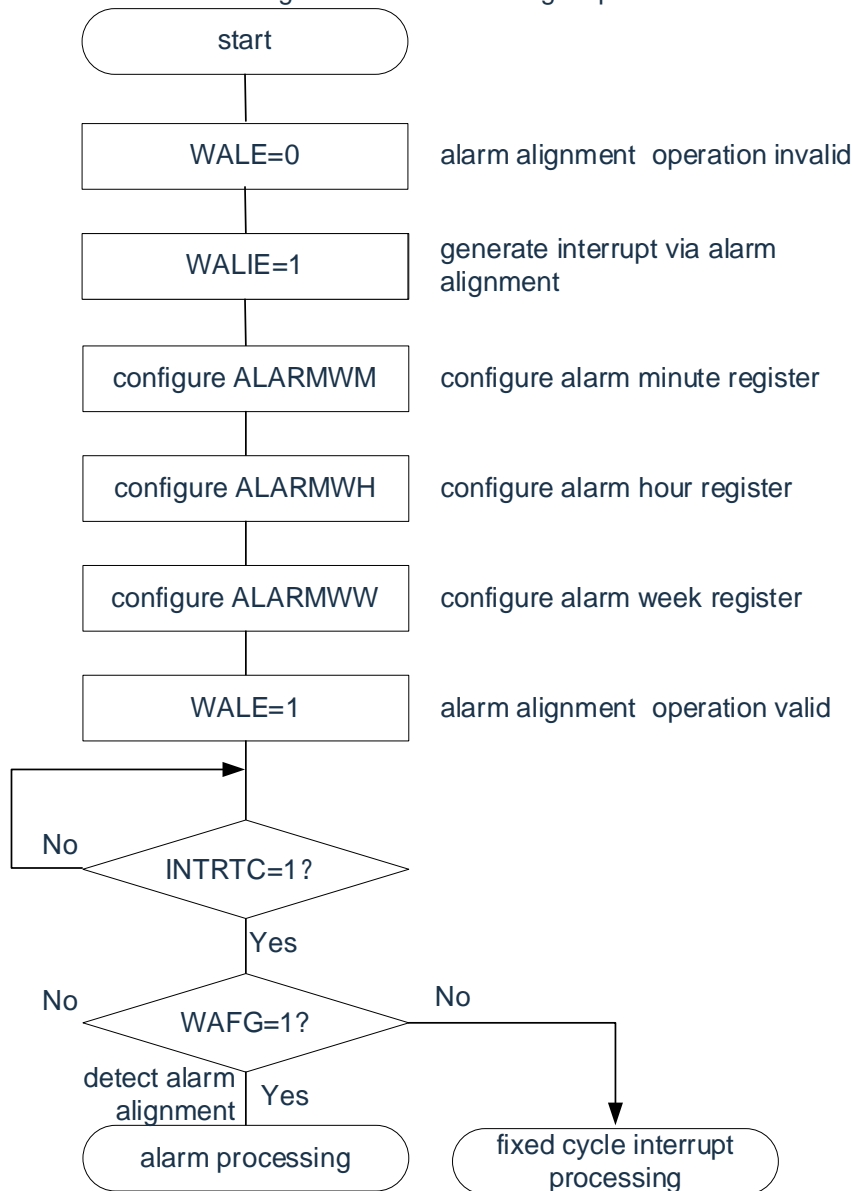
2. When changing the values of the SEC, MIN, HOUR, WEEK, DAY, MONTH, and YEAR register while the counter operates (RTCE = 1), rewrite the values after disabling interrupt servicing INTRTC by using the interrupt mask flag register, and the WAFG, RIFG, and RTCIF flags should be cleared after the rewrite.

Remark 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 7-21 Alarm setting steps

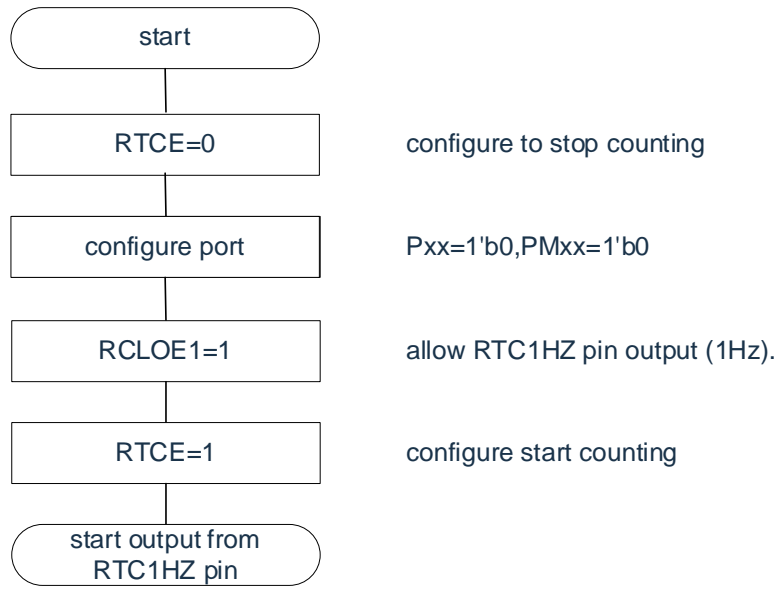


Remark1. There is no limit to the order of write operations for alarm minute registers (ALARMWM), alarm hour registers (ALARMWH), and alarm week registers (ALARMWW).

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 7-22 1Hz output setting procedure



Notice 1. 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)

$$\text{Correction value}^{\text{Note}} = \text{Number of correction counts in 1 minute} \div 3 = (\text{Oscillation frequency} \div \text{Target frequency} - 1) \times 32768 \times 60 \div 3$$

(When DEV=1)

$$\text{Correction value}^{\text{Note}} = \text{Number of correction counts in 1 minute} = (\text{Oscillation frequency} \div \text{Target frequency} - 1) \times 32768 \times 60$$

Note: The correction value is the clock error correction value calculated by using bits 12 to 0 of the clock error correction register (SUBCUD).

$$\text{(When F12=0) correction value} = \{(F11, F10, F9, F8, F7, F6, F5, F4, F3, F2, F1, F0) - 1\} \times 2$$

$$\text{(When F12=1) correction value} = -\{(/F11, /F10, /F9, /F8, /F7, /F6, /F5, /F4, /F3, /F2, /F1, /F0) + 1\} \times 2$$

When (F12~F0)=(*,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 = $32768 \times 0.9999817 \approx 32767.4\text{Hz}$

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).

$-\{(/F11~/F0)-1\} \times 2 = -36$

$(/F11~/F0) = 17$

$(/F11~/F0) = (0,0,0,0,0,0,0,1,0,0,0,1)$

$(F11~F0) = (1,1,1,1,1,1,1,0,1,1,1,0)$

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:1,1,1,1,1,1,1,0,1,1,1,0) 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.

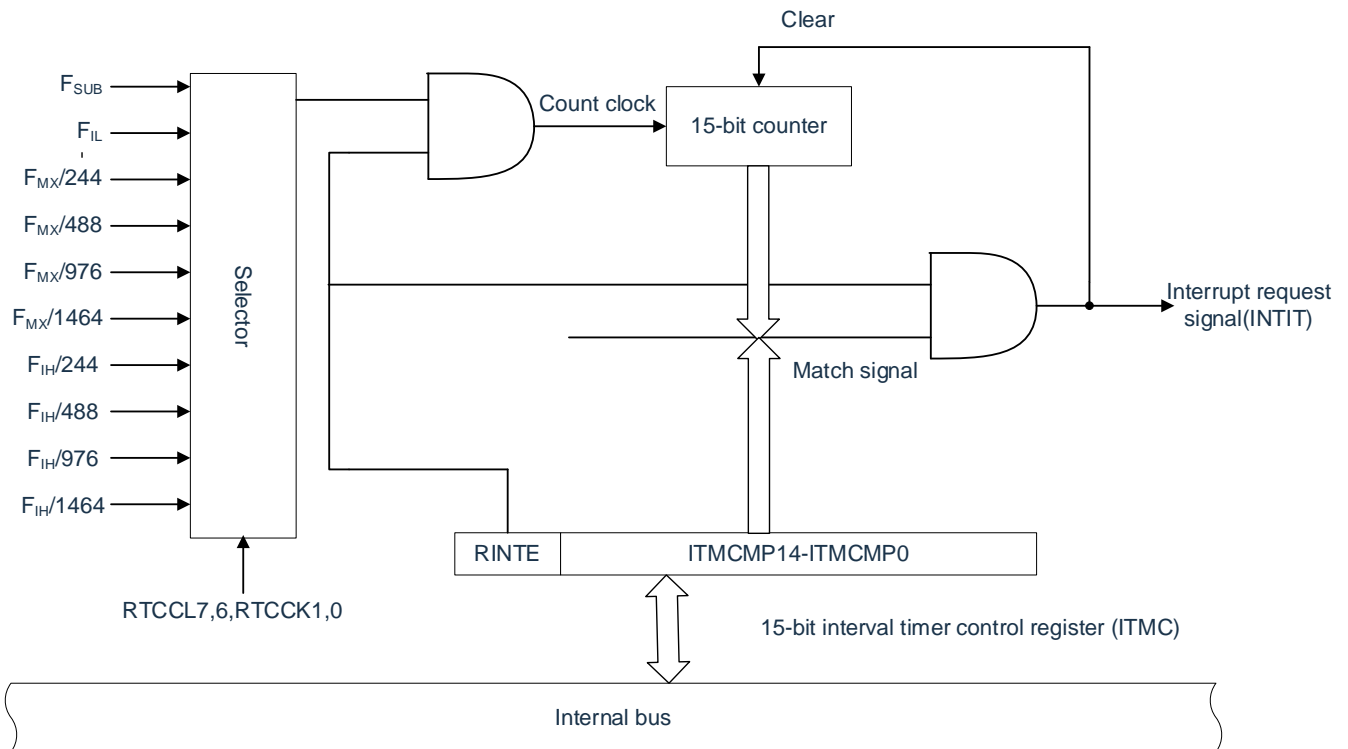
8.2 Structure of 15-bit interval timer

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 registers	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".

Figure 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	IRDAEN	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.

Figure 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	RTCKS1	RTCKS0

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})

RTCKS1	RTCKS0	RTCCL6	RTCCL5	Selection of operating clocks for real-time clocks, counting clocks for 15-bit interval timer
0	0	x	x	Subsystem 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”.

Figure 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	Settings are disabled.
0000H	
Example interrupt cycles when 001H or 7FFFH is specified for ITCMP14~ITCMP0	
<ul style="list-style-type: none"> • ITCMP14~ITCMP0=0001H, count clock: $F_{SUB}=32.768\text{KHz}$ $1/32.768[\text{KHz}] \times (1+1) = 0.06103515625[\text{ms}] \approx 61.03[\text{us}]$ • ITCMP14~ITCMP0=7FFFH, count clock: $F_{SUB}=32.768\text{KHz}$ $1/32.768[\text{KHz}] \times (32767+1) = 1000[\text{ms}]$ 	

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

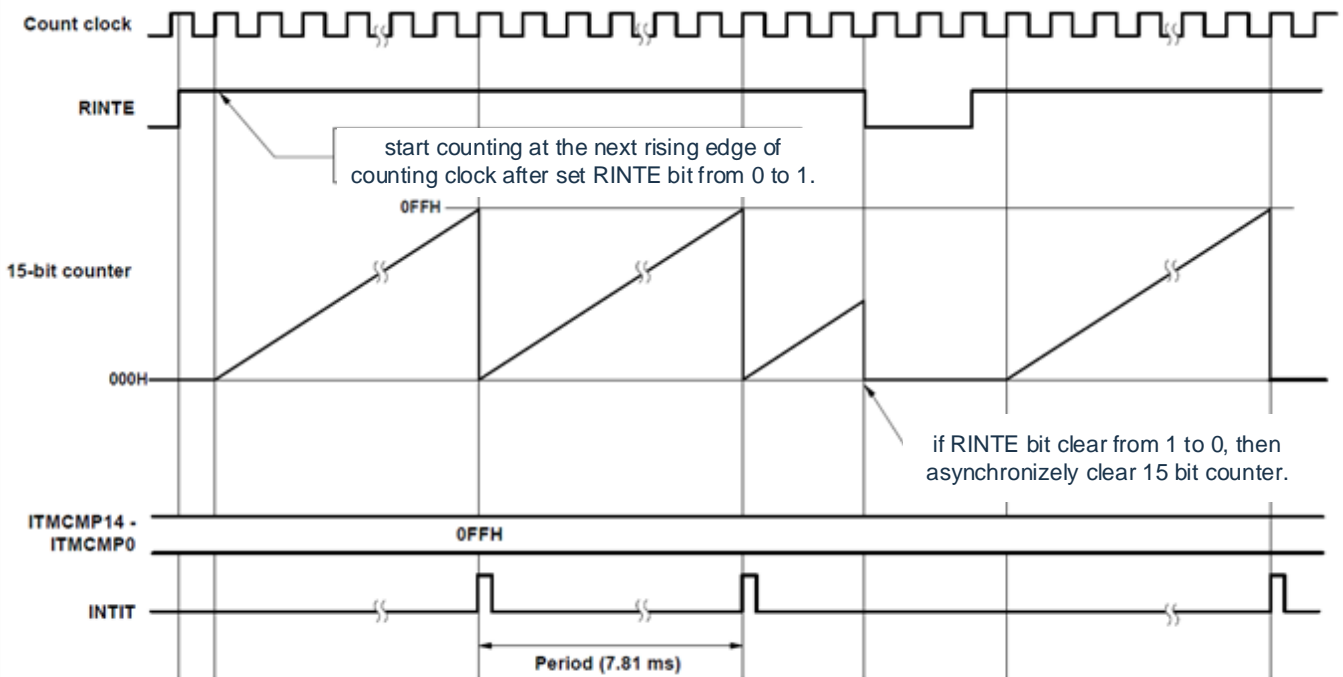
8.4.1 Operation timing of 15-bit interval timer

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 8-5.

Figure 8-5 Operation timing of 15-bit interval timer
(ITCMP14~ITCMP0=0FFH, count clock: $F_{SUB}=32.768\text{KHz}$)

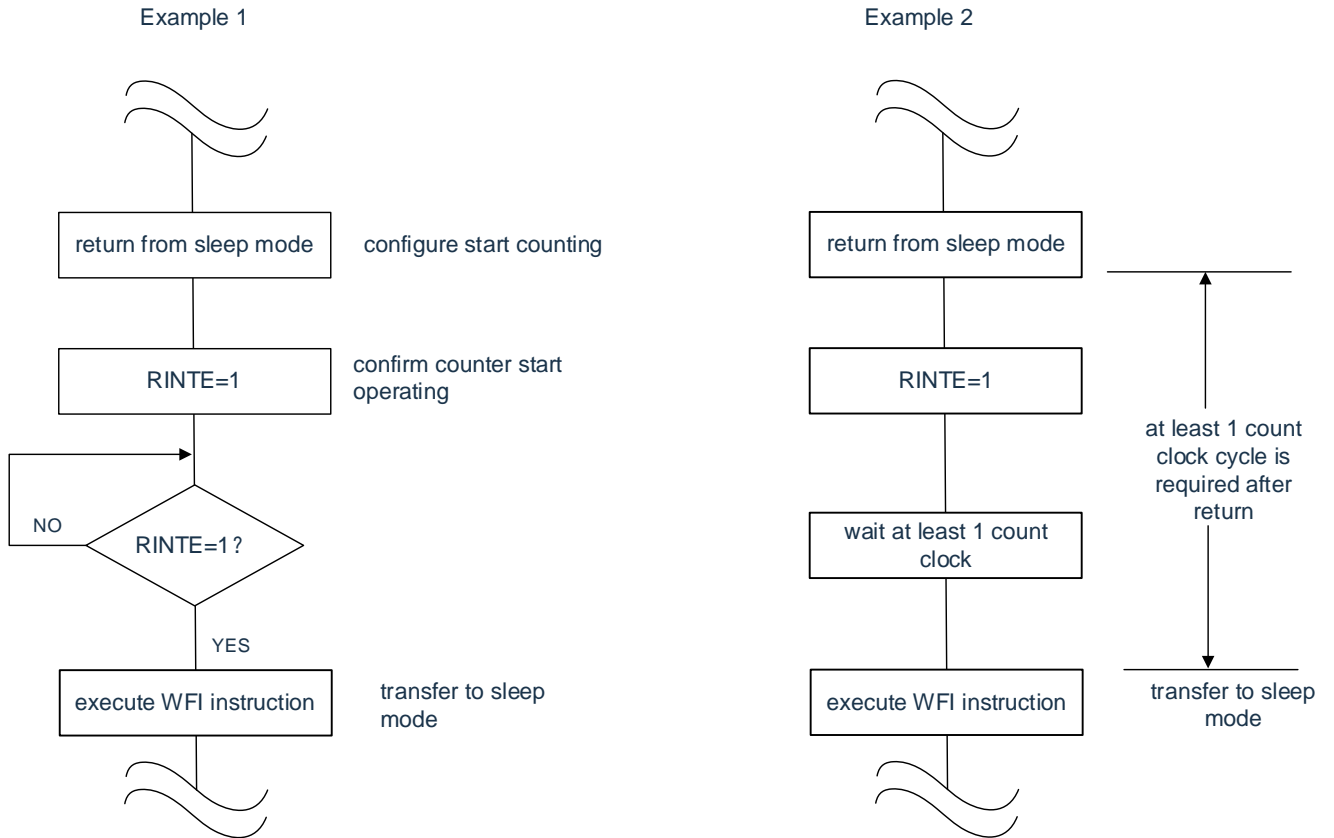


8.4.2 Start of count operation and re-enter to sleep mode after returned from sleep mode

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).



Chapter 9 Clock Output/Buzzer Output Controller

Notice: The following section of this chapter focuses on 48-pin products.

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.

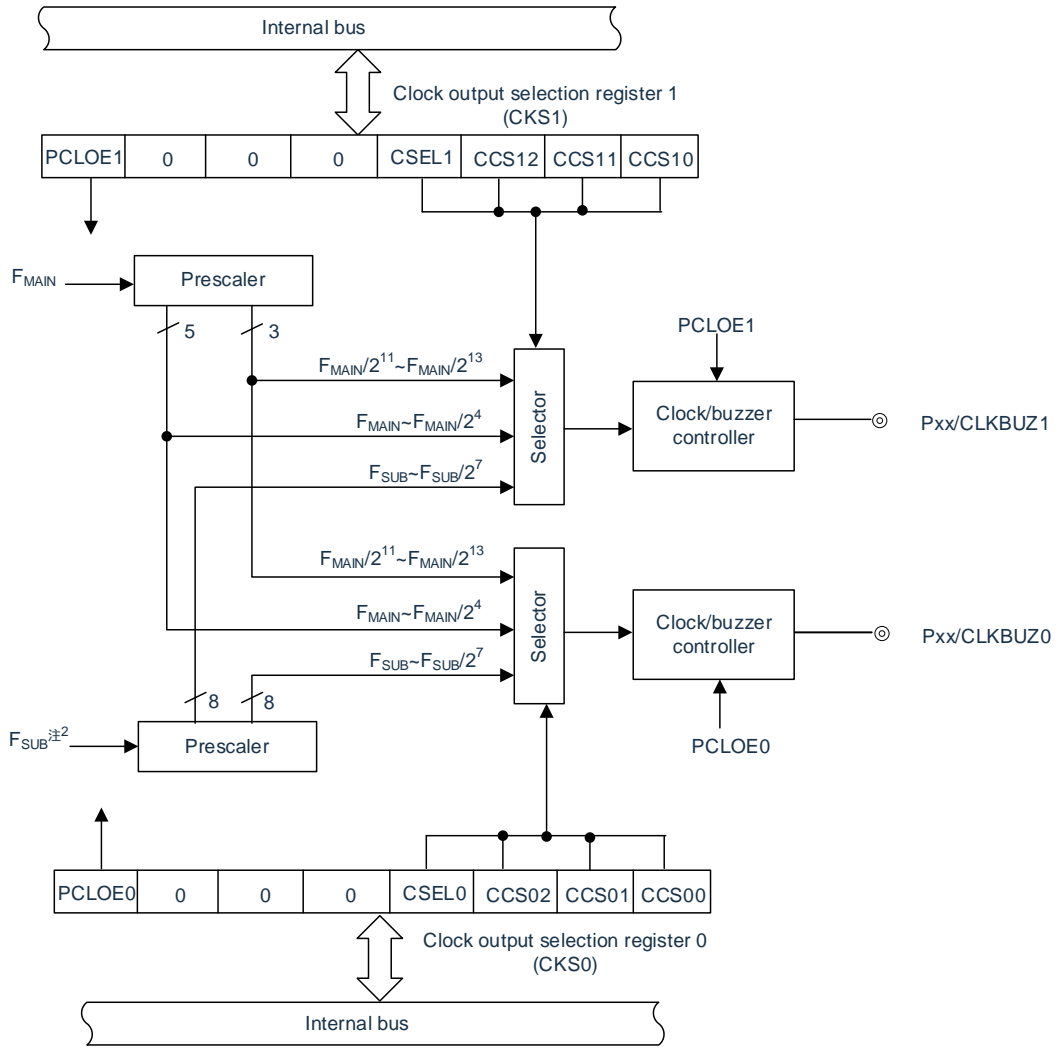
The CLKBUZn pin outputs the clock selected by the clock output selection register n (CKSn).

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



Note: 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".

Figure 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 Pin Functions”.

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).

Set the bit P20 of port register 2 to "0".

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

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

Set "0x18" to port multiplexing function configuration register P20CFG.

(Example) Using P15 as clock output/buzzer output (CLKBUZ1).

Set the bit P15 of the port register 1 to "0".

Set the PM15 bit of port mode register 1 to "0".

Set "0" to PMC15 bit of port mode control register 1.

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

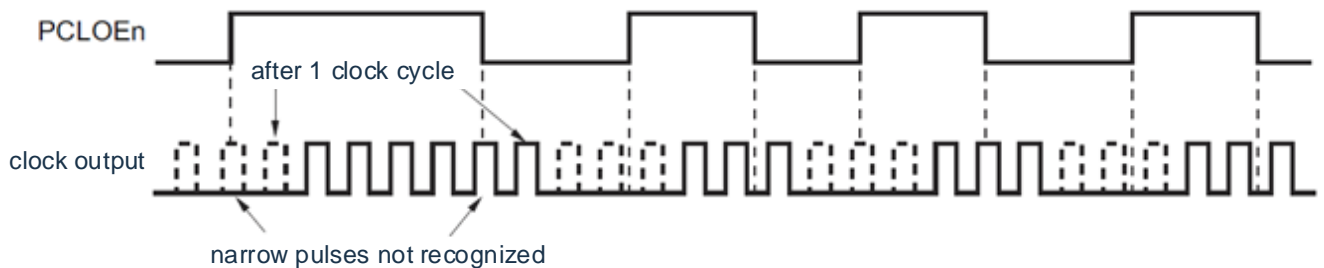
The CLKBUZn pin is output as the following procedure.

- 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. CLKBUZ1 is fixed-multiplexed to the P15 port, and there is no need to set the Port Multiplexing Function Configuration Register (PxxCFG) when using CLKBUZ1.
2. 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 9-3 shows enabling or stopping output using the PCLOEn bit and the timing of outputting the clock.
3. n=0, 1

Figure 9-3 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 a bit manipulation instruction is executed on the watchdog timer enable register (WDTE)
- (3) If data other than “ACH” is written to the WDTE register
- (4) 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 23 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)

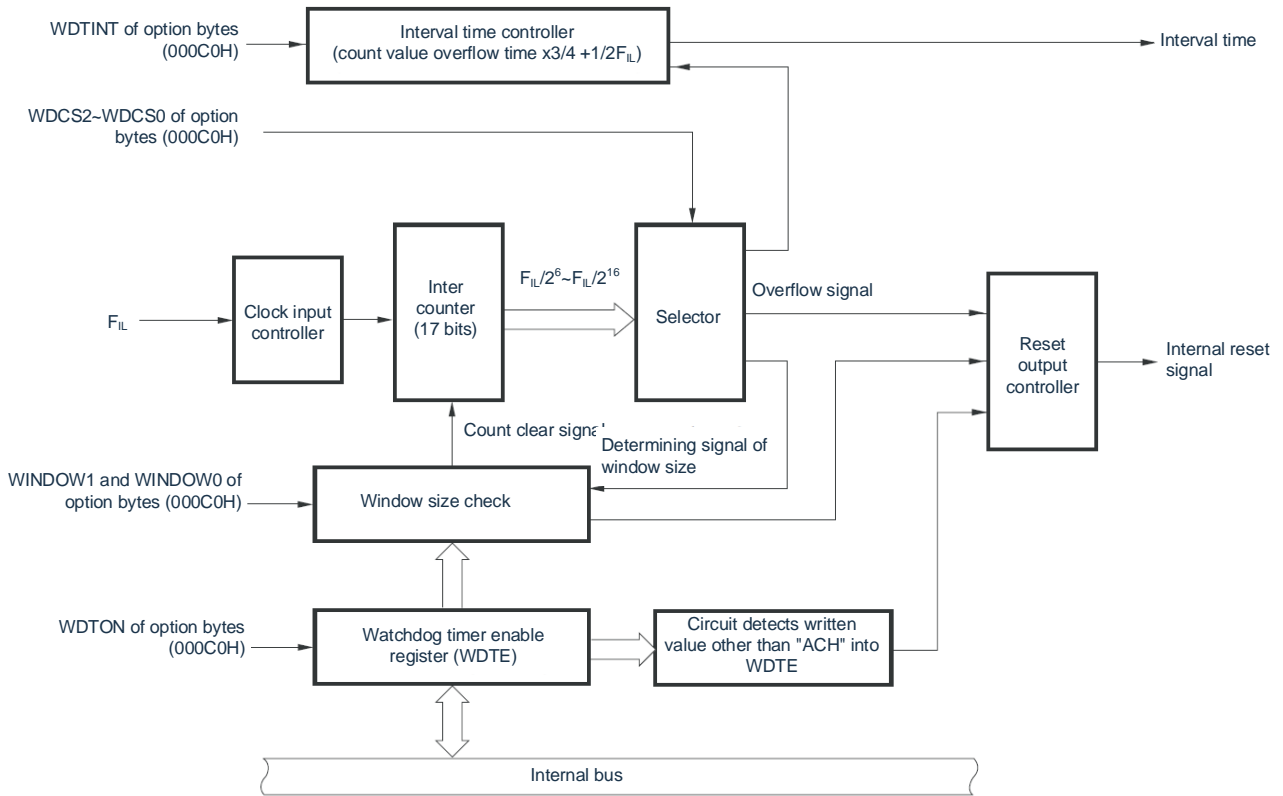
The operation of the counter is controlled by means of option bytes as well as setting the overflow time, the window opening period and the interval interruption.

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 28 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).

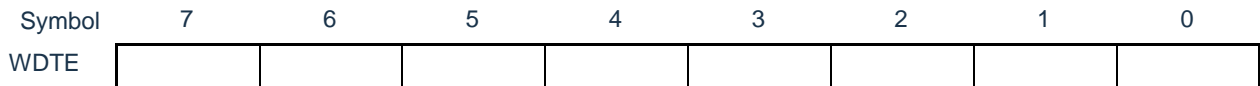
10.3.1 Watchdog timer enable register (WDTE)

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.

Figure 10-2 Format of watchdog timer enable register (WDTE)

Address: 0x40021001 After reset: 9AH/1AH^{Note} R/W



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

Notice

1. If a value other than “ACH” is written to the WDTE register, an internal reset signal is generated.
2. If a memory manipulation instruction is executed for the WDTE register, an internal reset signal is generated.
3. The value read from the WDTE register is 9AH/1AH (this differs from the written value (“ACH”)).

10.3.2 LOCKUP control register (LOCKCTL)

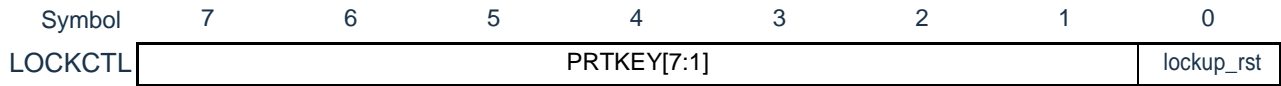
The LOCKCTL register is a configuration register for controlling the Cortex-M0+ LockUp function to operate the watchdog timer.

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

After generating a reset signal, the value of the LOCKCTL register changes to “00H”.

Figure 10-3 Format of LOCKUP control register (LOCKCTL)

Address: 40020406H After reset: 00H R/W



lockup_rst	Configuration of LOCKUP function
0	• LOCKUP does not cause a WDT reset
1	• LOCKUP causes the WDT to reset

PRTKEY[7:1]	Write protection of lockup_rst
78H	• lockup_rst is writable
Other	• lockup_rst is not writable

10.4 Operation of watchdog timer

10.4.1 Operational control of watchdog timer

- When using the watchdog timer, set the following items by option byte (000C0H):
 - 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 28 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)

- The overflow time must be set by bit3~1 (WDCS2~WDCS0) of the option byte (000C0H) (refer to 10.4.2 and Chapter 28 Option Byte for details).
 - 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 28 Option Byte for details).
- After the reset is released, the watchdog timer starts counting.
 - 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.
 - 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.
 - 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:
 - When a bit manipulation instruction is executed on a WDTE register
 - If data other than "ACH" is written to the WDTE register

Notice:

- 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.
- 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.
- The watchdog timer can be cleared immediately before the count value overflows.
- 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		

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.

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.

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 10-3 Setting of overflow time of watchdog timer

WDCS2	WDCS1	WDCS0	Overflow time of watchdog timer (When $F_{IL}=20\text{KHz (MAX.)}$)
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:

- If “ACH” is written to the watchdog timer enable register (WDTE) during the window open period, the watchdog timer is cleared and starts counting again.
- Even if “ACH” is written to the WDTE register during the window close period, an abnormality is detected and an internal reset signal is generated.

Notice 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 10-4 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%

Notice When bit 0 (WDSTBYON) of the option byte (000C0H) = 0, the window open period is 100% regardless of the values of the WINDOW1 and WINDOW0 bits.

Remark If the overflow time is set to $2^9/F_{IL}$, the window close time and open time are as follows.

	Setting of window open period	
	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%>

- Overflow time:
 $2^9/F_{IL} \text{ (Max.)} = 2^9/20\text{KHz (Max.)} = 25.6\text{ms}$
- Window close time:
 $0 \sim 2^9/F_{IL} \text{ (Min.)} \times (1-0.75) = 0 \sim 2^9/10\text{KHz} \times 0.25 = 0 \sim 12.8\text{ms}$
- Window open time:
 $2^9/F_{IL} \text{ (Min.)} \times (1-0.75) \sim 2^9/F_{IL} \text{ (Max.)} = 12.8 \sim 25.6\text{ms}$

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 10-5 Setting of watchdog timer interval interrupt

WDTINT	Use of watchdog timer interval interrupt
0	Interval interrupt is not used.
1	Interval interrupt is generated when $75\%+1/2F_{IL}$ of the overflow time is reached.

Notice: 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.

Remark: The watchdog timer continues counting even after INTWDTI is generated (until "ACH" is written to the watchdog timer enable register (WDTE)). If "ACH" is not written to the WDTE register before the overflow time, an internal reset signal is generated.

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.

Pins	32-pin	32-pin (-A) Note1	40-pin	40-pin (-A) Note1	44-pin (-A) Note1	48-pin	48-pin (-A) Note1	48-pin (-B) Note1
Analog input channels	25ch	22ch	28ch	28ch	31ch	35ch	35ch	37ch
	(ANI0~ANI3, ANI8~ANI14, ANI16~ANI24, ANI29, ANI31~ANI33, ANI36)	(ANI0~ANI3, ANI8~ANI14, ANI16~ANI24, ANI27, ANI29)	(ANI0~ANI5, ANI8~ANI14, ANI16~ANI24, ANI29, ANI31~ANI34, ANI36)	(ANI0~ANI6, ANI8~ANI14, ANI16~ANI24, ANI27, ANI29~ANI32)	(ANI0~AN24, ANI27~AN32)	(ANI0~AN24, ANI27~AN36)	(ANI0~AN24, ANI27~AN36)	(ANI0~AN36)

Note 1. (-A) indicates that it is limited to BAT32G135xx-A series products. (-B) indicates that it is limited to BAT32G135xx-B series products.

11.1 Function of A/D converter

The A/D converter is a converter that converts analog inputs to digital values, and can control the A/D conversion of up to 35 analog channels (15 pin input channels and 3 internal channels).

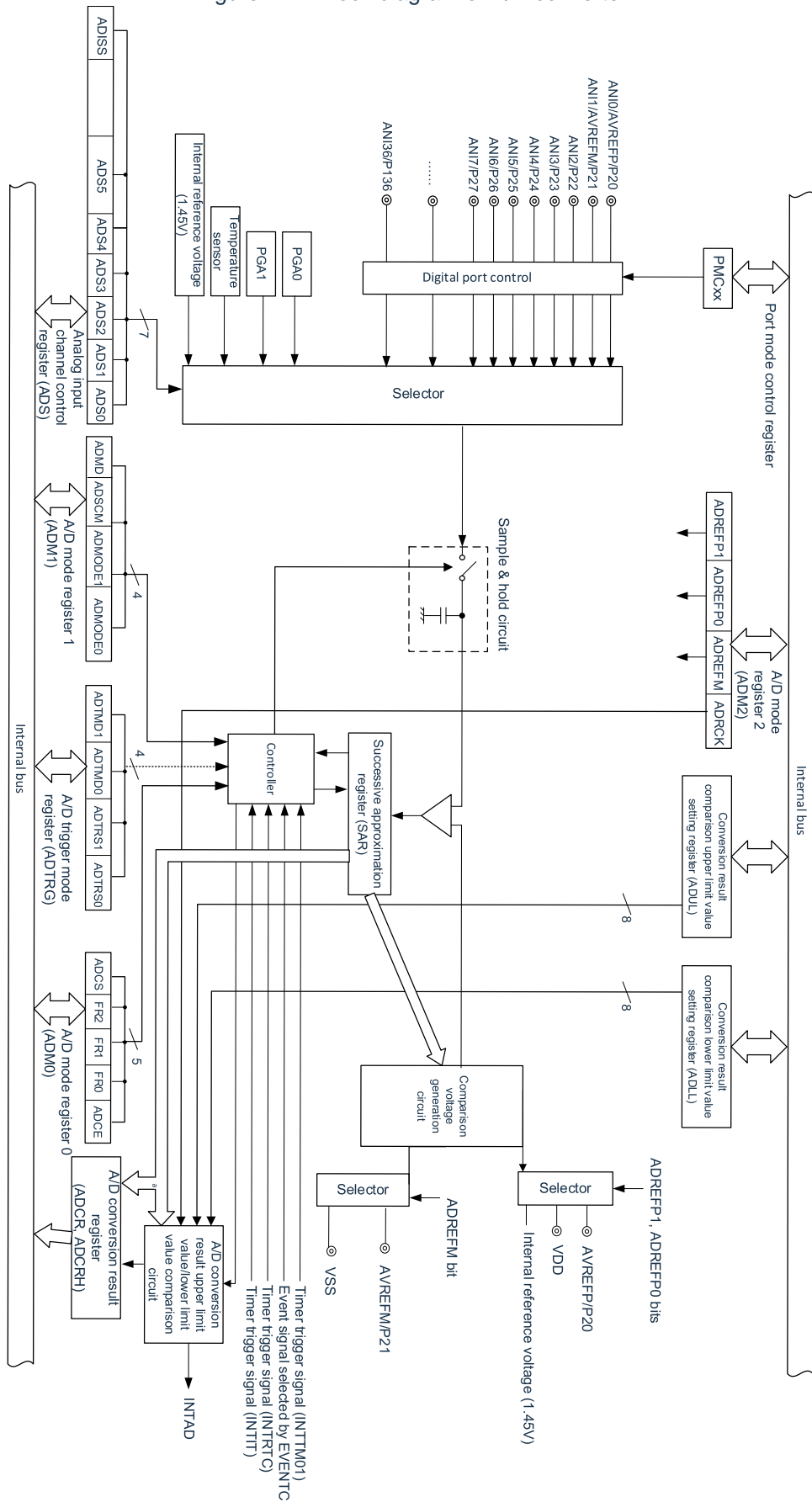
The A/D converter has the following functions.

- A/D conversion with 12-bit resolution
Repeat the 12-bit resolution A/D conversion by selecting the analog input of one channel from ANI0 to ANI24, ANI27 to ANI36, PGA0, PGA1 and the temperature sensor. An interrupt request (INTAD) is generated at the end of each A/D conversion (when the mode is selected).

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	Sampling clock 5.5~255 ADCLKs	The sampling time can be selected via the ADNSMP register, which uses 13.5 conversion clocks (f_{AD}) by default.

Figure 11-1 Block diagram of A/D converter



Note: For the selection of analog input channel ANIx, refer to 11. 2. 6 Analog 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
ADNSMP	A/D converter sampling time control register	R/W	0dH	ADC_BASE+0CH
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
ADTES	A/D test register	R/W	00H	ADC_BASE+10H
ADNDIS	A/D converter charge/discharge control register	R/W	00H	ADC_BASE+11H
ADSMPWAIT	A/D converter sampling time extension control register	R/W	00H	ADC_BASE+15H
ADFLG	A/D hardware module status register	R	00H	ADC_BASE+16H
PMCn	Port mode control register	R/W	Note1	PORT_BASE+Note1

R: read only, W: write only, R/W: both read and write

Note1: 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".

Figure 11-2 Format 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 input clock for A/D converter
0	Stops input clock supply. <ul style="list-style-type: none"> • SFR used by the A/D converter cannot be written. • The A/D converter is in the reset status.
1	Enables input clock supply. <ul style="list-style-type: none"> • SFR used by the A/D converter can be read and written.

Notice1. 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)).

- A/D converter mode register 0 (ADM0)
- A/D converter mode register 1 (ADM1)
- A/D converter mode register 2 (ADM2)
- A/D converter trigger mode register (ADTRG)
- Analog input channel specification register (ADS)
- Conversion result comparison lower limit setting register (ADLL)
- Conversion result comparison upper limit setting register (ADUL)
- A/D converter sampling time control register (ADNSMP)
- 12-bit A/D conversion result register (ADCR)
- 8-bit A/D conversion result register (ADCRH)
- A/D test register (ADTES)
- A/D converter charge/discharge control register (ADNDIS)
- A/D converter sampling time extension control register (ADSMPWAIT)
- A/D hardware module status register (ADFLG)

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".

Figure 11-3 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.

Note:

1. For details of the FR2~FR0 bits and A/D conversion, please refer to "Table 11-3 A/D Conversion Time Selection".
2. It takes 1 μ s 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 1 μ s 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 1us 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".

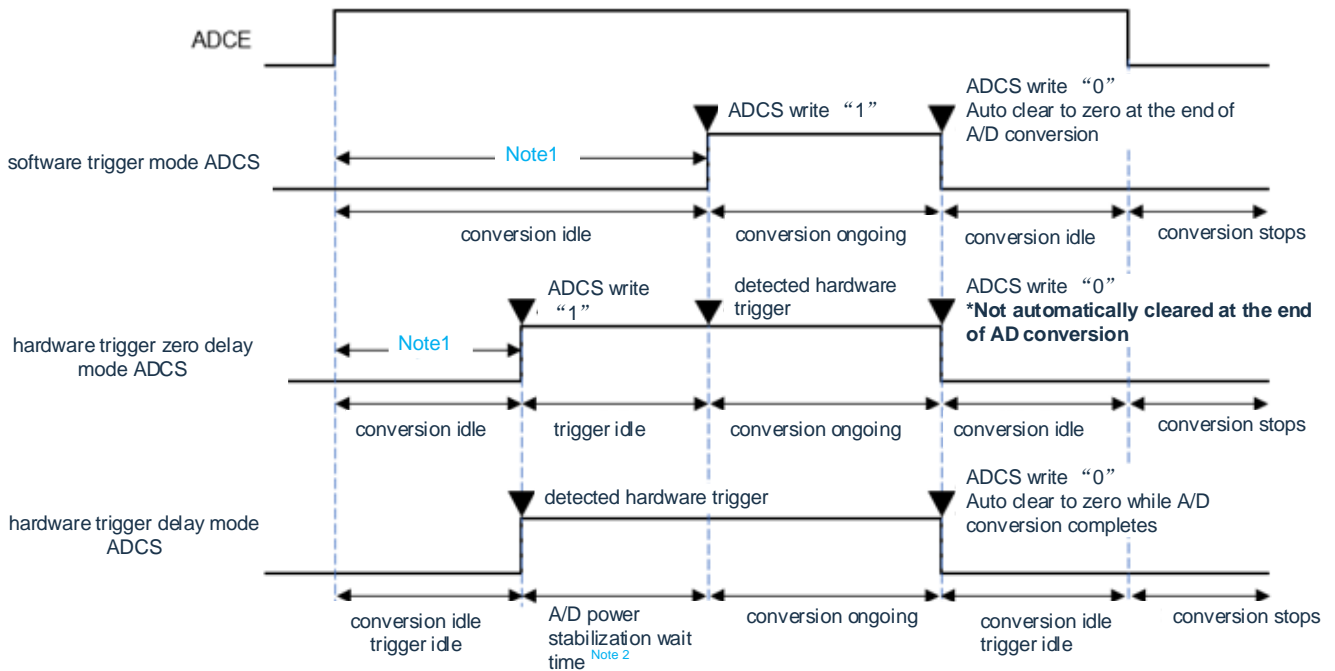
Table 11-1 ADCS bit and ADCE bit settings

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 11-2 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		<ul style="list-style-type: none"> 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		<ul style="list-style-type: none"> 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		<ul style="list-style-type: none"> When writing "0" to the ADCS bit
	Scan mode	Continuous conversion mode		When writing "0" to the ADCS bit
		Single conversion mode		<ul style="list-style-type: none"> 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		<ul style="list-style-type: none"> 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		<ul style="list-style-type: none"> When writing "0" to the ADCS bit Automatically clears to "0" when the set 4-channel conversion is completed.

Figure 11-4 Diagram of using various modes of A/D



Note 1: In software trigger mode or hardware trigger no-wait mode, it takes at least 1us (TBD) to rise from the ADCE bit to the ADCS bit to stabilize the internal circuitry.

2: In hardware trigger wait mode, the A/D power supply stabilization time of 1us is guaranteed by design.

Notice:

- 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.
- The ADCE bit must be overridden when the ADCS bit is "0" (Stop Transition/Transition Standby).
- 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: $2 F_{CLK} \text{ clocks} + \text{A/D conversion time}$

Hardware trigger wait mode: $2 F_{CLK} \text{ clocks} + \text{A/D power supply stable wait time} + \text{A/D conversion time}$

Remark f_{CLK} : CPU/peripheral hardware clock frequency

Table 11-3 Selection of 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 converter mode register 1 (ADM1) ^{Note 1}		Mode	Frequency of conversion clock ADCLK (F_{AD})	12-bit resolution conversion time ^{Note 2}	
FR2	FR1	FR0	ADM0[1]	ADM0[0]			ADC conversion time = (number of sample clocks + number of successive comparison clocks) / F_{AD}	ADC conversion clock number (13.5 sample clocks)
0	0	0	0	0	High-speed conversion mode	$f_{CLK}/32$	45 ADCLK (13.5 sample clocks +31.5 successive comparison clocks)	$45 / f_{AD}$
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	1	Low-current mode	$f_{CLK}/32$	54 ADCLK (13.5 sample clocks +40.5 successive comparison clocks)	$54 / f_{AD}$
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				$f_{CLK}/1$		

Note1. To override the FR2~FR0 bits and ADM0[1:0] bits into different data, it must be done in the conversion stop state (ADCS=0).

Note2. Time required to perform an ADC conversion = (number of sample clocks + number of successive comparison clocks) / F_{AD}

The number of sampling clocks can be adjusted through the ADNSMP register, the default is 13.5 ADCLK, the number of successive comparison clocks is determined by the conversion mode, the high-speed conversion mode is 31.5 ADCLK, the low current mode is 40.5 ADCLK, the fastest clock supported by the ADCLK is 64MHz in the high-speed conversion mode, the fastest clock supported by the ADCLK is 27MHz in the low current mode. For actual use, please configure the conversion mode and conversion clock frequency according to the "AC Characteristics" in the datasheet.

Remark f_{CLK} : CPU/peripheral hardware clock frequency

Table 11-4 Selection of A/D conversion time (2/2)

 (2) With A/D power stabilization wait time (hardware trigger wait mode ^{Note1})

A/D converter mode register 0 (ADM0)			A/D converter mode register 1 (ADM1)		Mode	Frequency of conversion clock ADCLK (f_{AD})	A/D power supply stabilization wait time	ADC conversion clock number	A/D power stabilization Waiting time +ADC conversion time ^{Note 2}
FR2	FR1	FR0	ADM0E[1]	ADM0D[0]					
0	0	0	0	0	High-speed conversion mode	$f_{CLK}/32$	1us	45 ADCLK (13.5 sample clocks +31.5 successive comparison clocks)	$1\mu s + 45/f_{AD}$
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	1	Low-current mode	$f_{CLK}/32$	1us	54 ADCLK (13.5 sample clocks +40.5 successive comparison clocks)	$1\mu s + 54/f_{AD}$
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				$f_{CLK}/1$			

Note 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.

Note 2. Time required for ADC conversion after hardware triggering = $1\mu s + (\text{number of sample clocks} + \text{number of successive comparison clocks})/f_{AD}$.

The number of sample clocks can be adjusted through the ADNSMP register, the default is 13.5 ADCLK, the number of successive comparison clocks is determined by the conversion mode, it is 31.5 ADCLK in high speed conversion mode, and 40.5 ADCLK in low current mode, the fastest clock supported by ADCLK is 64MHz in high speed conversion mode, and the fastest clock supported by ADCLK is 27MHz in low current mode.

Notice1. To override the FR2~FR0 bits and ADM0E[1:0] bits into different data, it must be done in the conversion stop state (ADCS=0).

2. The conversion time in the hardware trigger wait mode includes the A/D power stabilization wait time after the hardware trigger is detected.

Remark f_{CLK} : CPU/peripheral hardware clock frequency

11.2.3 A/D converter mode register 1 (ADM1)

This is a 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”.

Figure 11-5 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	ADMODE1	ADMODE0

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

ADMODE1	ADMODE0	A/D conversion mode
0	0	High-speed conversion mode (ADCLK fastest clock is 64MHz)
1	1	Low current mode (ADCLK fastest clock is 27 MHz)
Others		Setting disabled

Notice Bits 6 to 4, 2 must be set to “0”.

Notice1. To override the ADM1 register, it must be done in the conversion stop state (ADCS=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

3. For A/D conversion, the number of successive comparison clocks is determined by the conversion mode, which is 31.5 ADCLK for high speed conversion mode and 40.5 ADCLK for low current mode, and the fastest clock supported by ADCLK is 64 MHz for high speed conversion mode and 27 MHz for low current mode. For actual use, please configure the conversion mode and conversion clock frequency according to the “AC Characteristics” in the datasheet.

Remark1. 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 a reset signal is generated, the value of this register becomes “00H”.

Figure 11-6 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	ADREFM	0	ADRCK	0	CHRDE	0

ADREFP1	ADREFP0	Selection of positive (+) voltage references for A/D converters
0	0	Provided by V _{DD} .
0	1	Provided by the AV _{REFP} external terminal.
1	0	Provided by internal reference voltage (1.45V).
1	1	Setting disabled.

ADREFM	Selection of negative (-) voltage references for A/D converters
0	Provided by V _{SS} .
1	Provided by the AV _{REFM} external terminal.

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

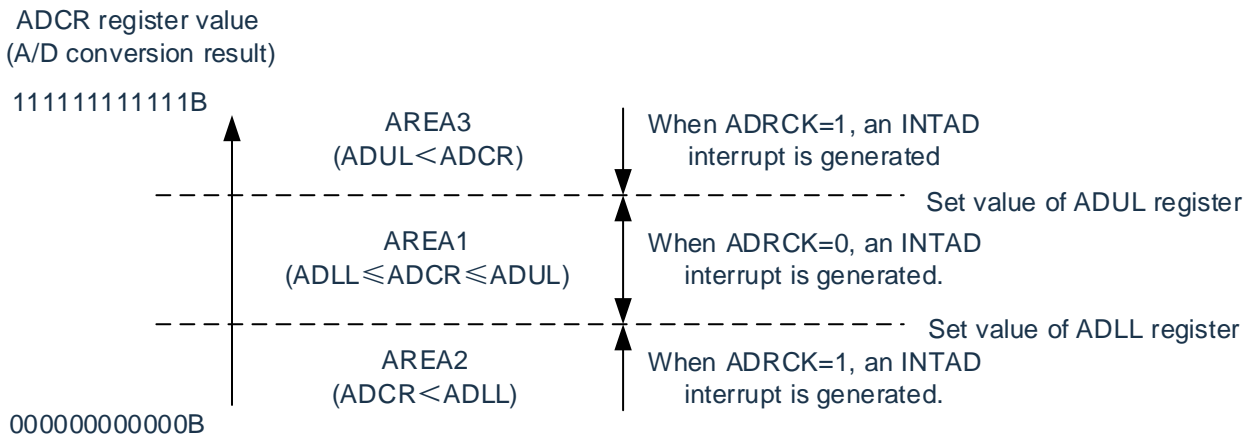


Figure 11-7 Range of interrupt signal generation for the ADRCK bit

Notice1. To override the ADM2 register, it must be done in the conversion stop state (ADCS=0).

2. When using AV_{REFP} and AV_{REFM}, the port must be set to analog port mode (PMCxx=1).

Remark 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 a 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 a reset signal is generated, the value of this register becomes “00H”.

Figure 11-8 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 (INTTMC1)
0	1	The event signal selected by ELC
1	0	Real-time clock interrupt signal (INTRTC).
1	1	Interval timer interrupt signal (INTIT).

Notice1. 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

Remark1. f_{CLK} : CPU/peripheral hardware clock frequency

11.2.6 Analog input channel specification register (ADS)

This is a 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".

Figure 11-9 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	ANI17(P27)
0	6'h08	ANI8(P11)
0	6'h09	ANI9(P10)
0	6'h0a	ANI10(P01)
0	6'h0b	ANI11(P00)
0	6'h0c	ANI12(P147)
0	6'h0d	ANI13(P12)
0	6'h0e	ANI14(P120)
0	6'h0f	ANI15(P146)
0	6'h10	ANI16(P13)
0	6'h11	ANI17(P14)
0	6'h12	ANI18(P15)
0	6'h13	ANI19(P16)
0	6'h14	ANI20(P17)
0	6'h15	ANI21(P30)
0	6'h16	ANI22(P31)
0	6'h17	ANI23(P50)
0	6'h18	ANI24(P51)
0	6'h19	ANI27(P60)
0	6'h1a	ANI28(P61)
0	6'h1b	ANI27(P62)
0	6'h1c	ANI28(P63)
0	6'h1d	ANI29(P70)
0	6'h1e	ANI30(P71)
0	6'h1f	ANI31(P72)
0	6'h20	ANI32(P73)
0	6'h21	ANI33(P74)
0	6'h22	ANI34(P75)

0	6'h23	ANI35(P130)
0	6'h24	ANI36(P136)
0	6'h25	PGA0
0	6'h26	PGA1
0	6'h3f	ANIx all OFF
1	6'h00	Temperature sensor
1	6'h01	Internal reference voltage (1.45V)
Settings are disabled.		

Note 1. If the internal reference voltage (1.45V) is selected as the reference voltage for comparator 0 or comparator 1, the temperature sensor output cannot be selected.

2. 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. When AV_{REFP} is used as the positive (+) reference voltage for the A/D converter, ANI0 cannot be selected as the A/D converter channel.
6. When AV_{REFM} is used as the negative (-) reference voltage for the A/D converter, ANI1 cannot be selected as the A/D converter channel.
7. After setting the ADISS bit to "1", the result of the first conversion cannot be used. For details of the setting procedure, refer to "11.5.4 Setting for Selecting the Temperature Sensor Output Voltage/Internal Reference Voltage".
8. 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.

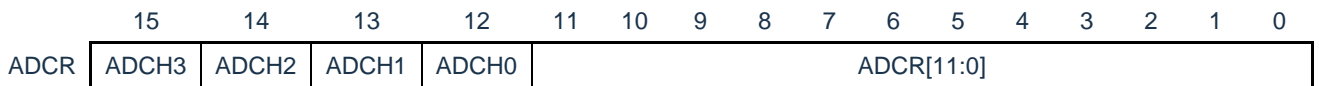
The ADCR register can be read by a 16-bit memory manipulation instruction.

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 (refer to Figure 11-7)), the A/D conversion results are not saved.

Figure 11-10 Format of 12-bit A/D conversion result register (ADCR)

Reset value: 0000H R



Notice

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)
ADCH0~3 read value 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.

Figure 11-11 Format of 8-bit A/D conversion result register (ADCRH)

Reset value: 00H R



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 11-7 Range of interrupt signal generation for the ADRCK bit). The ADUL register is set by an 8-bit memory manipulation instruction.

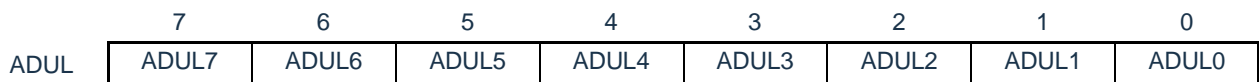
After a reset signal is generated, the value of this register becomes "FFH".

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.
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.

Figure 11-12 Format of conversion result comparison upper limit setting register (ADUL)

Reset value: FFH R/W



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 11-7 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".

Figure 11-13 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".

Figure 11-14 Format of A/D converter sampling time control register (ADNSMP)

Reset value: 0dH R/W



Sample clock number setting:

ADNSMP[7:0]	Sample time	Remark
8'h05	5.5 ADCLK	
8'h06	6.5 ADCLK	
8'h07	7.5 ADCLK	
8'h08	8.5 ADCLK	
8'h09	9.5 ADCLK	
8'h0a	10.5 ADCLK	
8'h0b	11.5 ADCLK	
8'h0c	12.5 ADCLK	
8'h0d	13.5 ADCLK	Default value
8'h0e	14.5 ADCLK	
8'h0f	15.5 ADCLK	
8'h10	16.5 ADCLK	
8'h11	17.5 ADCLK	
8'h12	18.5 ADCLK	
8'h13	19.5 ADCLK	
8'h14	20.5 ADCLK	
.....	
8'hff	255.5 ADCLK	

Notice: To rewrite the ADNSMP register, it must be in the conversion stop status (ADCS=0).

Time required to perform an ADC conversion:

High-speed conversion mode: ADC conversion time = (number of sampling clocks + number of successive

comparison clocks (31.5))/F_{AD}

Low-current conversion mode: ADC conversion time = (number of sampling clocks + number of successive

comparison clocks (40.5))/F_{AD}

The number of AD sampling clocks can be adjusted by the ADNSMP register, and the default value is 13.5 ADCLK. The number of successive comparison clocks are determined by the conversion mode, which is 31.5 ADCLK for high speed conversion mode and 40.5 ADCLK for low current mode.

Under different conditions, the sampling time of each channel should be guaranteed:

Sampling time calculation equations: Number of sampling clocks / $f_{AD} \geq$ recommended sampling time

A/D conversion mode	AVDD[V]	ANix[ns]	PGA/ temperature sensors/internal reference voltage [ns]
high speed transformation	4.5~5.5	211	633
	2.7~5.5	250	750
	2.4~5.5	422	1266
low current transformation	2.7~5.5	500	759
	2.4~5.5	844	1281
	1.8~5.5	1688	2563

Notice: For actual use, please configure the conversion mode, the number of sampling clocks and the conversion clock frequency according to the "AC Characteristics" requirement in the data sheet.

11.2.12 A/D sample time extension register (ADSMPWAIT)

This register is used to extend the A/D sampling time.

The ADSMPWAIT register is set by an 8-bit memory operation instruction.

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

Figure 11-15 Format of A/D sample time extension register (ADSMPWAIT)

Reset value: 00H

R/W

	7	6	5	4	3	2	1	0
ADSMPWAIT	0	0	0	0	0	0	0	ADSMPWAIT

ADSMTWIT	A/D conversion object
0	When "0", the A/D sampling time is set directly by the ADNSMP register
1	The sampling time of A/D is arbitrarily prolonged for "1", and is controlled continuously by ADNSMP after changing from "1" to "0"

Note: Set ADSMPWAIT=1 in the conversion stop state (ADCS=0), and rewrite ADSMPWAIT to "0" when (ADCS=1).

11.2.13 A/D test register (ADTES)

This register is used to set the test mode of the A/D converter.

The ADTES register is set by an 8-bit memory operation instruction.

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

Figure 11-16 Format of A/D test register (ADTES)

Reset value: 00H R/W

	7	6	5	4	3	2	1	0
ADTES	0	0	0	0	0	ADTES2	ADTES1	ADTES0

ADTES2	ADTES1	ADTES0	A/D operation mode
0	0	0	Normal conversion
0	0	1	Self-diagnostic testing for 0 code
0	1	1	Self-diagnostic testing of half code
1	0	1	Self-diagnostic testing of full codes
Other than above			Disable settings.

Notice: Bits 7 to 3 must be set to "0".

11.2.14 A/D status register (ADFLG)

This register represents the state of the A/D converter.

The ADFLG register is read by an 8-bit memory operation instruction.

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

Figure 11-17 Format of A/D status register (ADFLG)

Reset value: 00H R

	7	6	5	4	3	2	1	0
ADFLG	0	0	0	ADFLG4	ADFLG3	ADFLG2	ADFLG1	ADFLG0

ADFLG4	A/D transition state
0	A/D conversion is not complete when single conversion mode
1	End of conversion in single conversion mode (auto clear after 2 ADCLKs) ADFLG4 keeps 1'b0 during continuous conversion mode

ADFLG3	A/D transition state
0	1 ADCLK before non-A/D conversion ends
1	1 ADCLK before A/D conversion ends (auto clear after 1 ADCLK)

ADFLG2	A/D transition state
0	2 ADCLK before non-A/D conversion ends
1	2 ADCLK before A/D conversion ends (1 ADCLK auto-zeroing)

ADFLG1	A/D transition state
0	Non-successive compare period
1	Successive compare period

ADFLG0	A/D transition state
0	Non-A/D sampling period
1	A/D sampling period

11.2.15 A/D charge/discharge control register (ADNDIS)

The register is used to control the charging and discharging operation and time of the A/D converter.

The ADNDIS register is read and written by an 8-bit memory operation instruction.

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

Figure 11-18 Format of A/D charge/discharge control register (ADNDIS)

Reset value: 00HW

	7	6	5	4	3	2	1	0
ADNDIS	0	0	0	ADNDIS4	ADNDIS3	ADNDIS2	ADNDIS1	ADNDIS0

ADNDIS [4]	Charge and discharge control
1'b0	discharge
1'b1	charging

ADNDIS [3:0]	Charge and discharge time
4'b0000	No charging or discharging
4'b0010	2x ADCLK
4'b0011	3x ADCLK
4'b0100	4x ADCLK
4'b0101	5x ADCLK
4'b0110	6x ADCLK
.....
4'b1111	15x ADCLK

Note: Disable setting charge/discharge time for 1 ADCLK, i.e. ADNDIS[3:0]=4'b0001

11.2.16 Registers controlling port functions of analog input pins

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 Pin Functions".

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.

$$ADCR = INT \left(\frac{V_{AIN}}{AV_{REF}} \times 4096 + 0.5 \right)$$

or

$$(ADCR - 0.5) \times \frac{AV_{REF}}{4096} \leq V_{AIN} < (ADCR + 0.5) \times \frac{AV_{REF}}{4096}$$

INT(): A function that returns the integer portion of a numeric value in parentheses

V_{AIN}: Analog input voltage

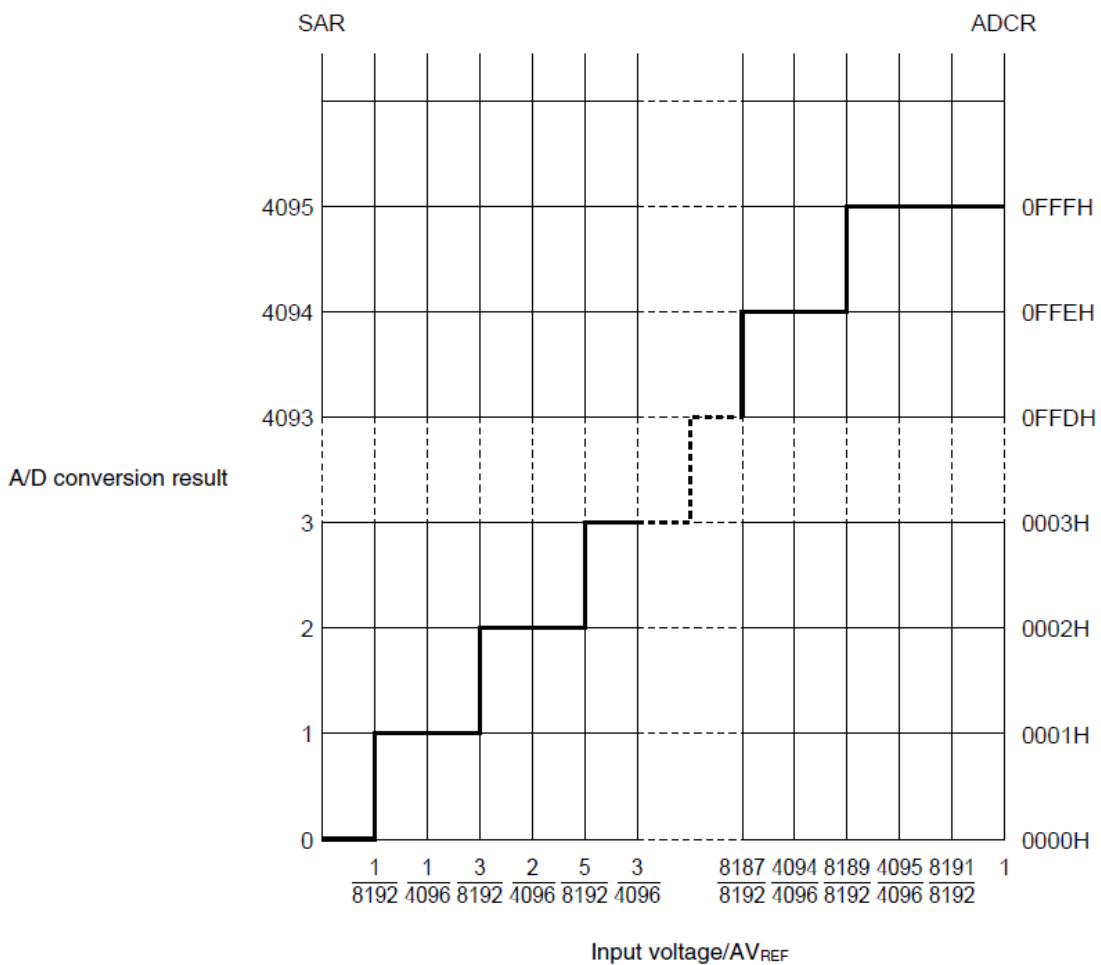
AV_{REF}: AV_{REF} pin voltage

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 11-19 Analog input voltage vs. A/D conversion results



Remark: AV_{REF} is the positive (+) reference voltage of the A/D converter.

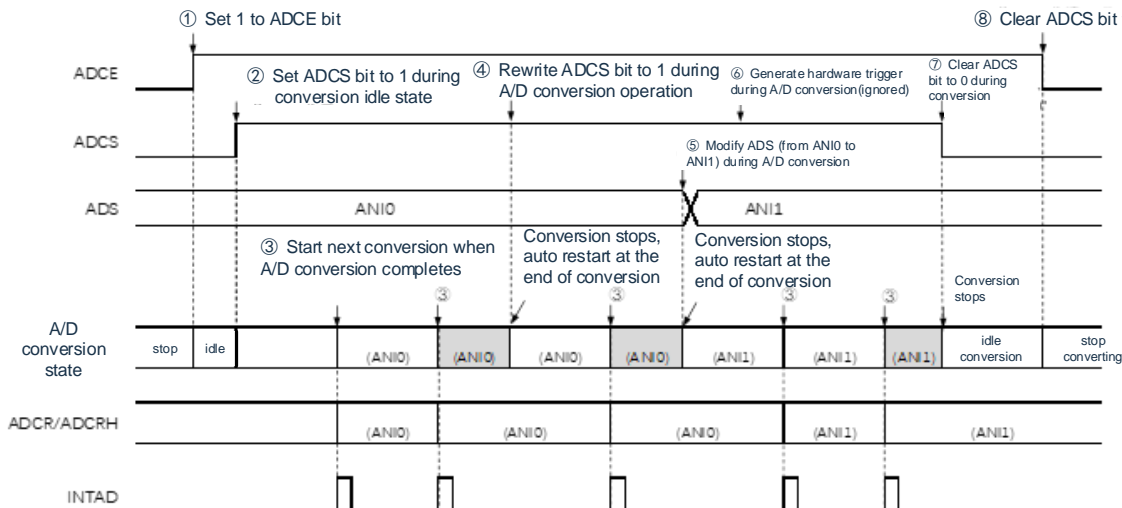
11.4 Operation mode of A/D converter

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, continuous 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 μ s), 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.

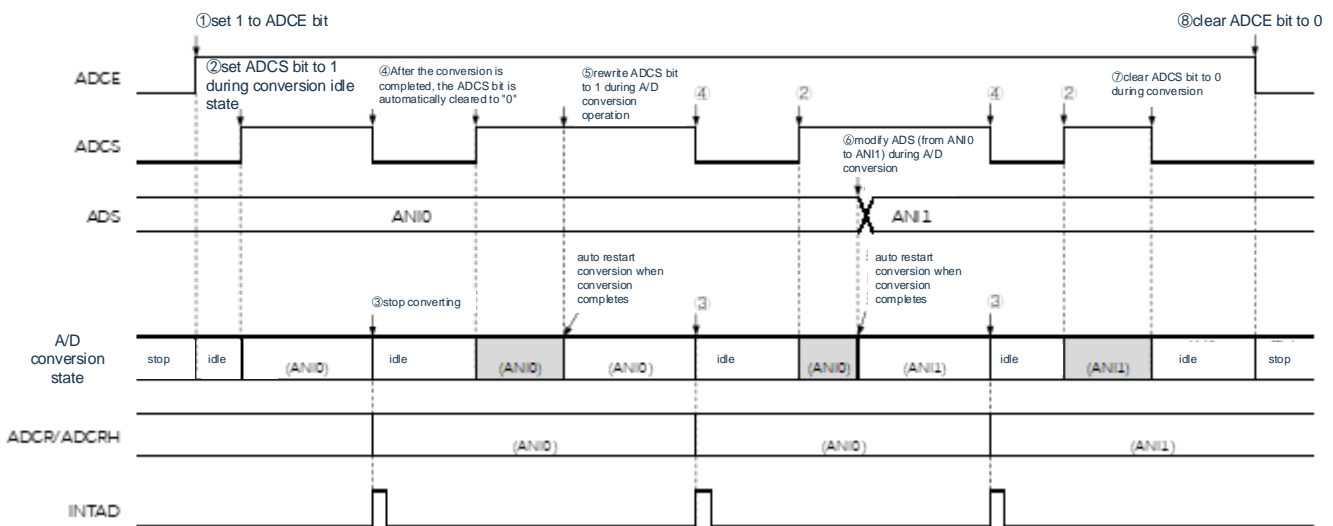
Figure 11-20 Example of software trigger mode (select mode, sequential conversion mode) operation timing



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.

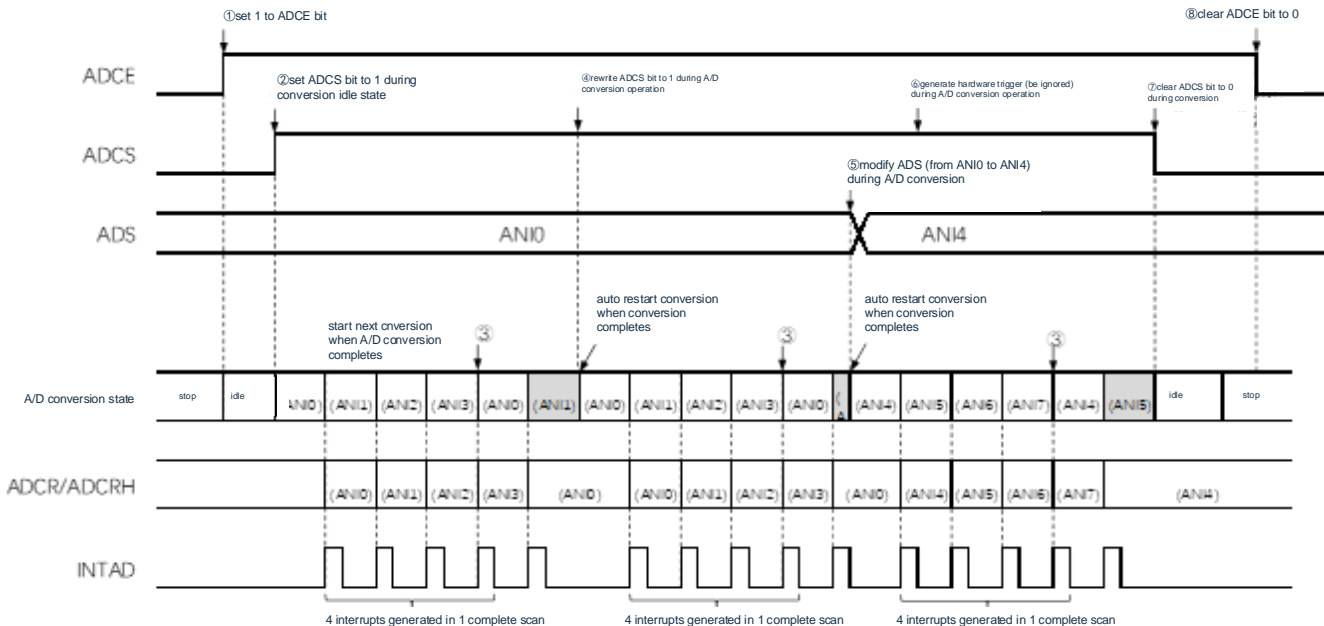
Figure 11-21 Example of software trigger mode (select mode, single conversion mode) operation timing



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.

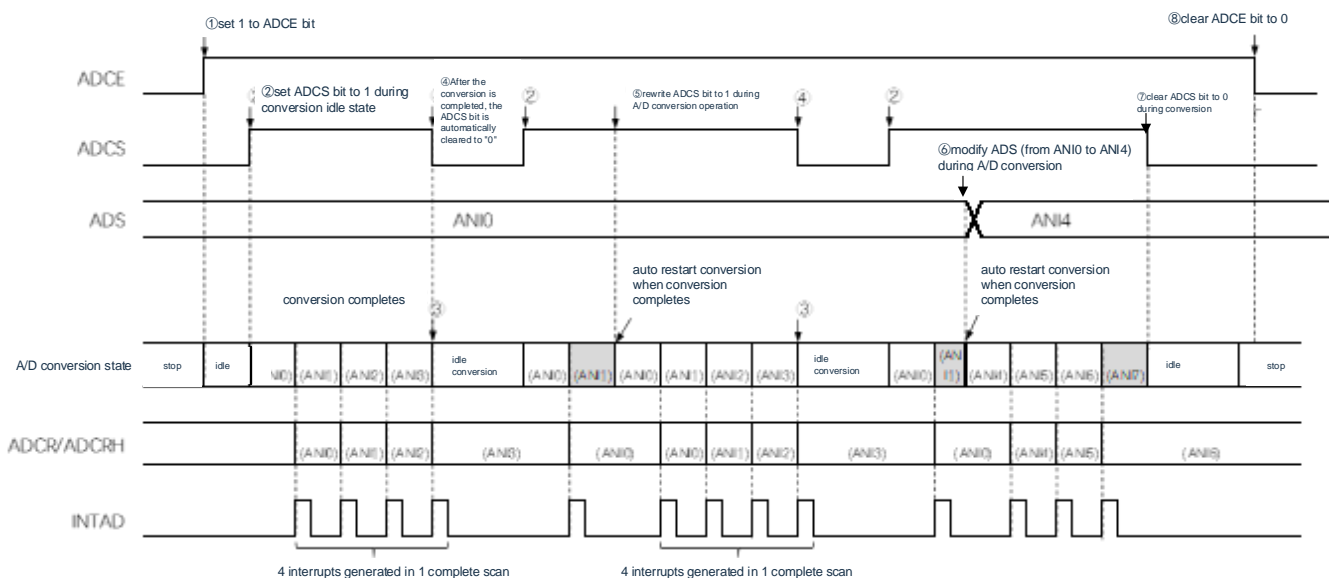
Figure 11-22 Example of software trigger mode (scan mode, sequential conversion mode) operation timing



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.

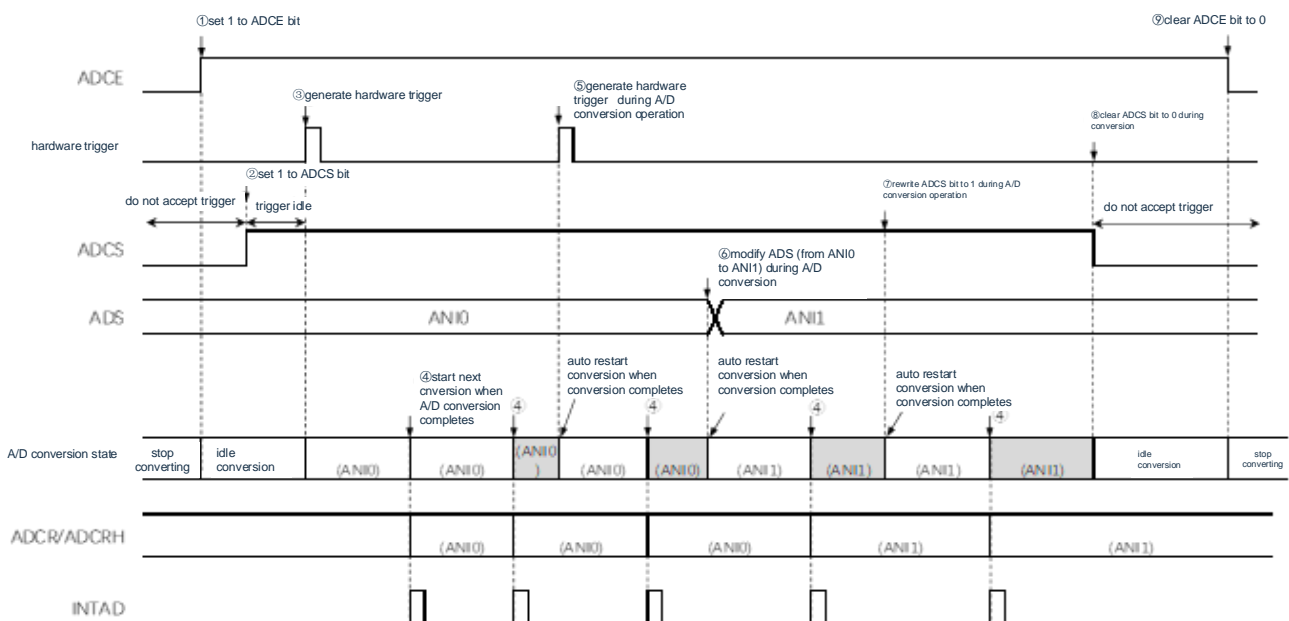
Figure 11-23 Example of software trigger mode (scan mode, single conversion mode) operation timing



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 μ s), 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.

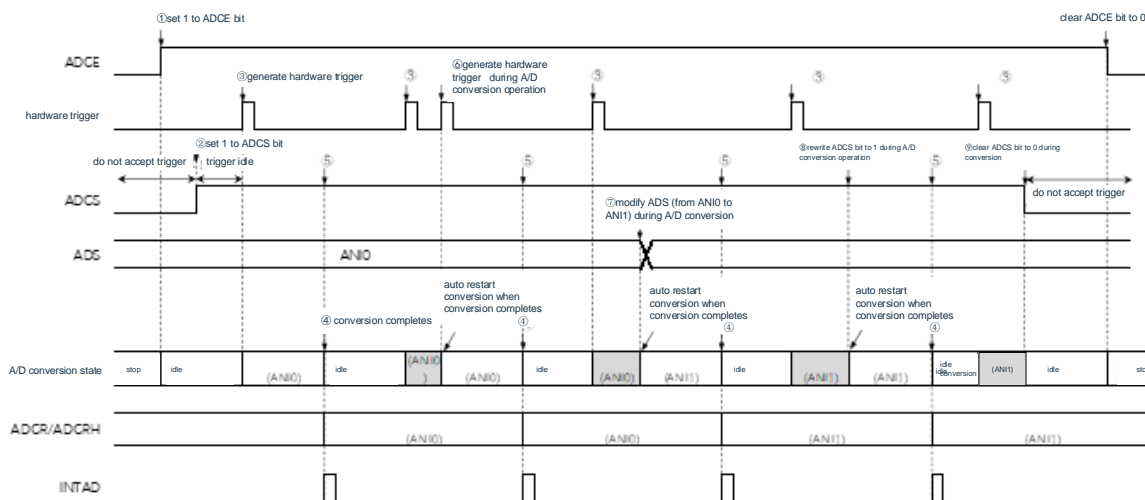
Figure 11-24 Example of hardware trigger no-wait mode (select mode, sequential conversion mode) operation timing



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 μ s), 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.

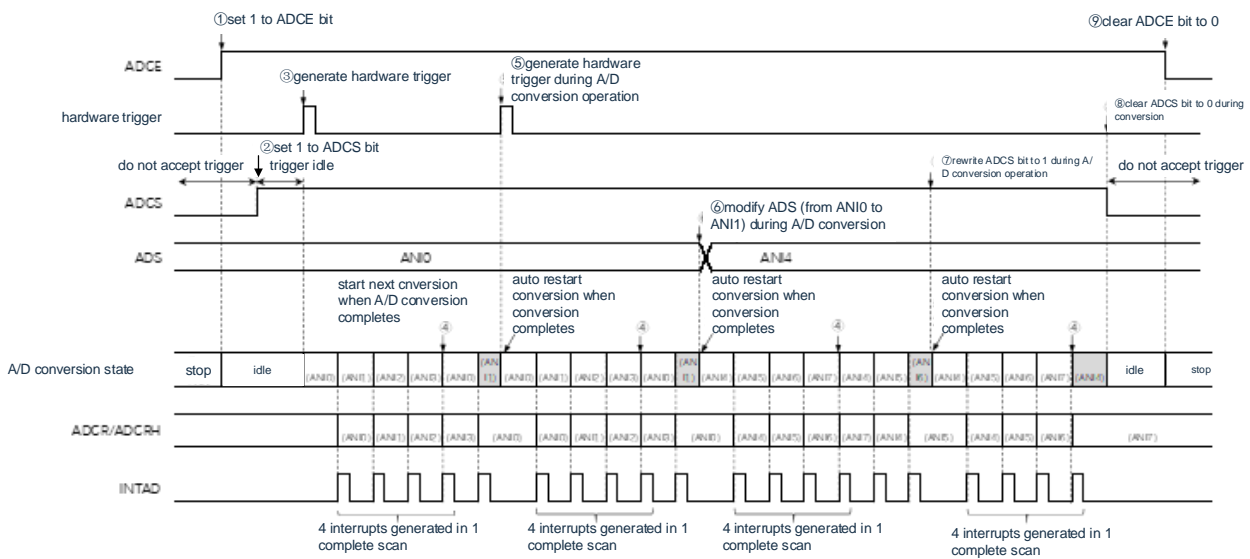
Figure 11-25 Example of hardware trigger no-wait mode (select mode, single conversion mode) operation timing



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 μ s), 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.

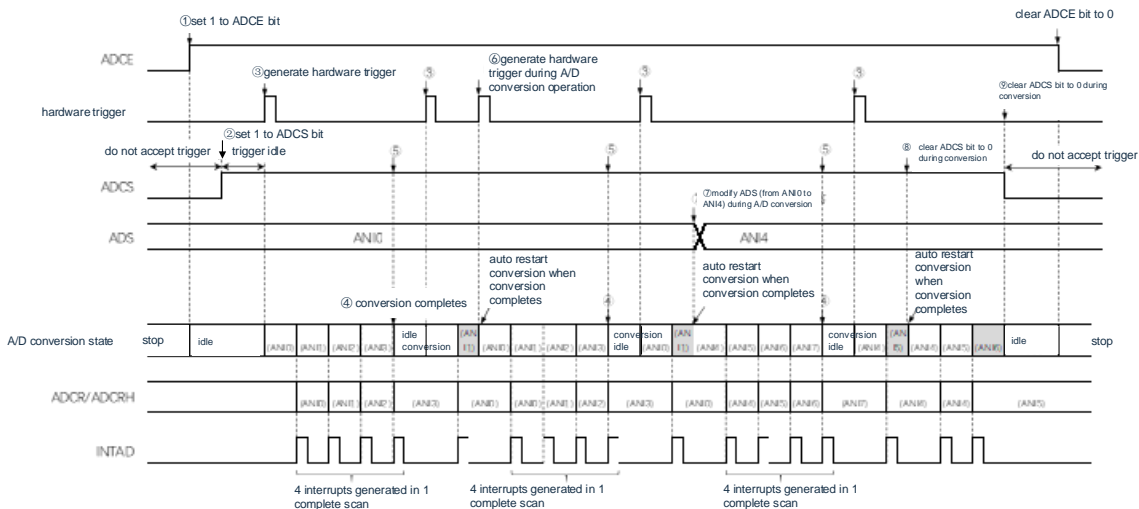
Figure 11-26 Example of hardware trigger no-wait mode (scan mode, sequential conversion mode) operation timing



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 μ s), 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.

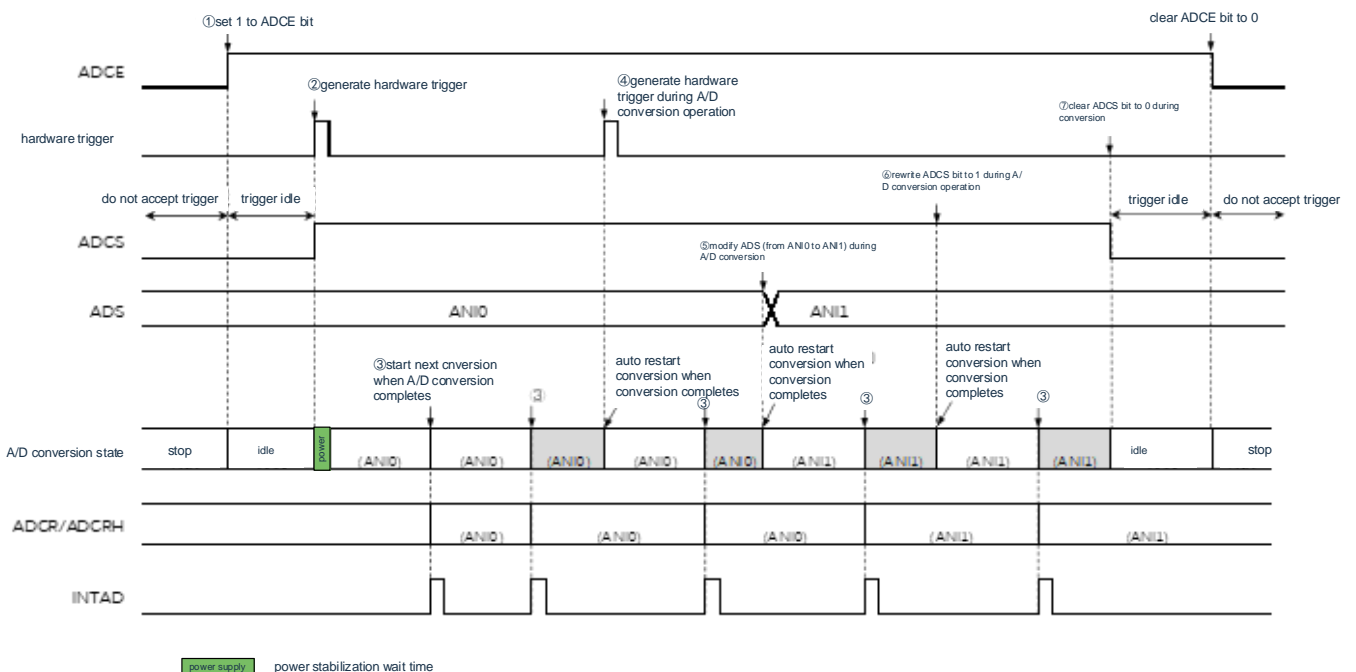
Figure 11-27 Hardware trigger no-wait mode (scan mode, single conversion mode)



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.

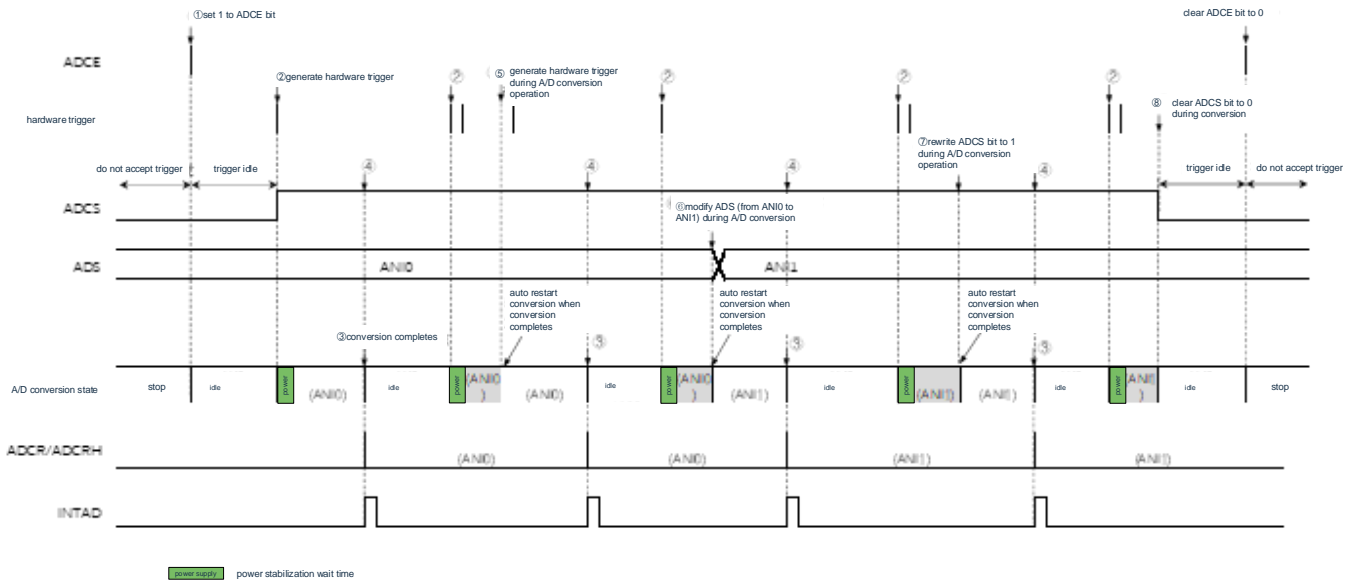
Figure 11-28 Example of hardware trigger wait mode (select mode, sequential conversion mode) operation timing



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.

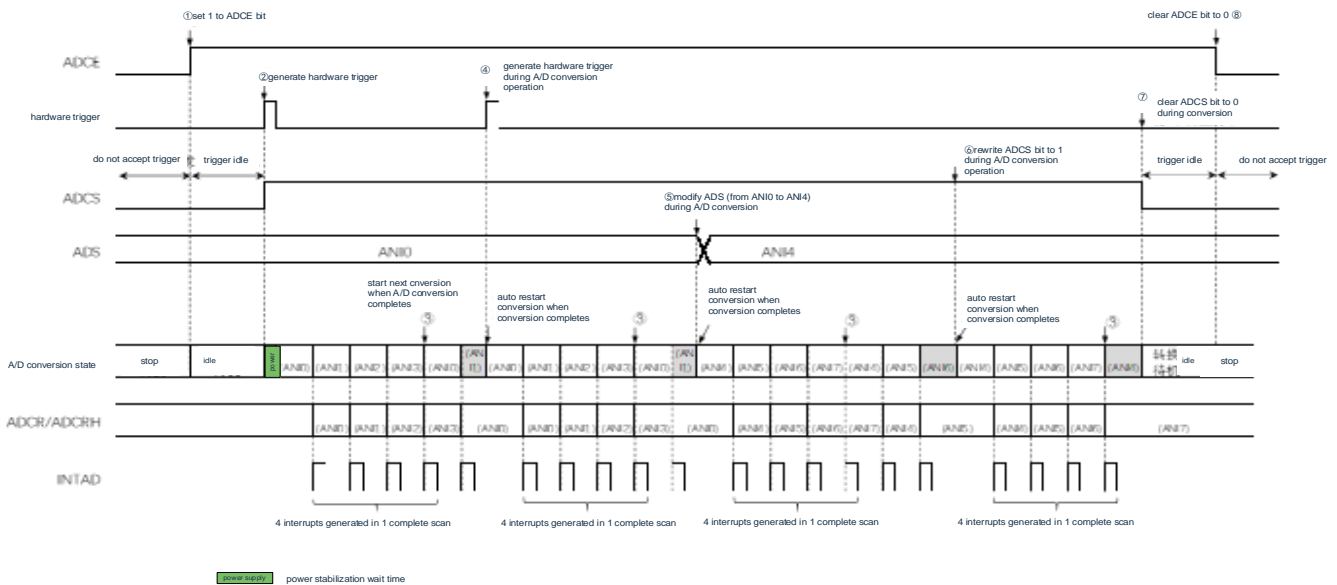
Figure 11-29 Example of hardware trigger wait mode (select mode, single conversion mode) operation timing



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.

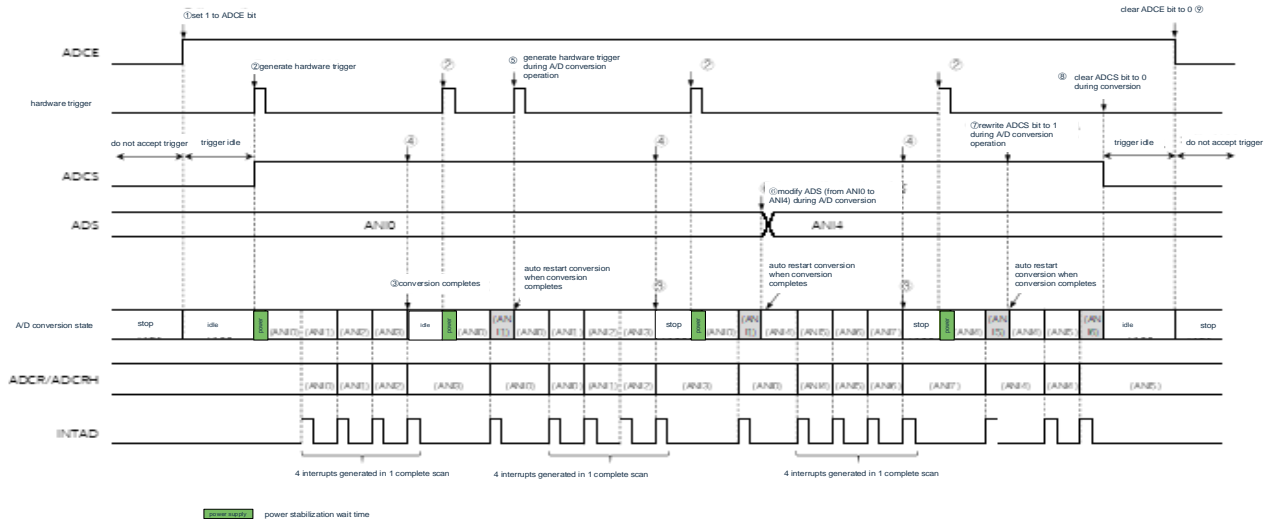
Figure 11-30 Example of hardware trigger wait mode (scan mode, sequential conversion mode) operation timing



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 11-31 Example of hardware trigger wait mode (scan mode, single conversion mode) operation timing

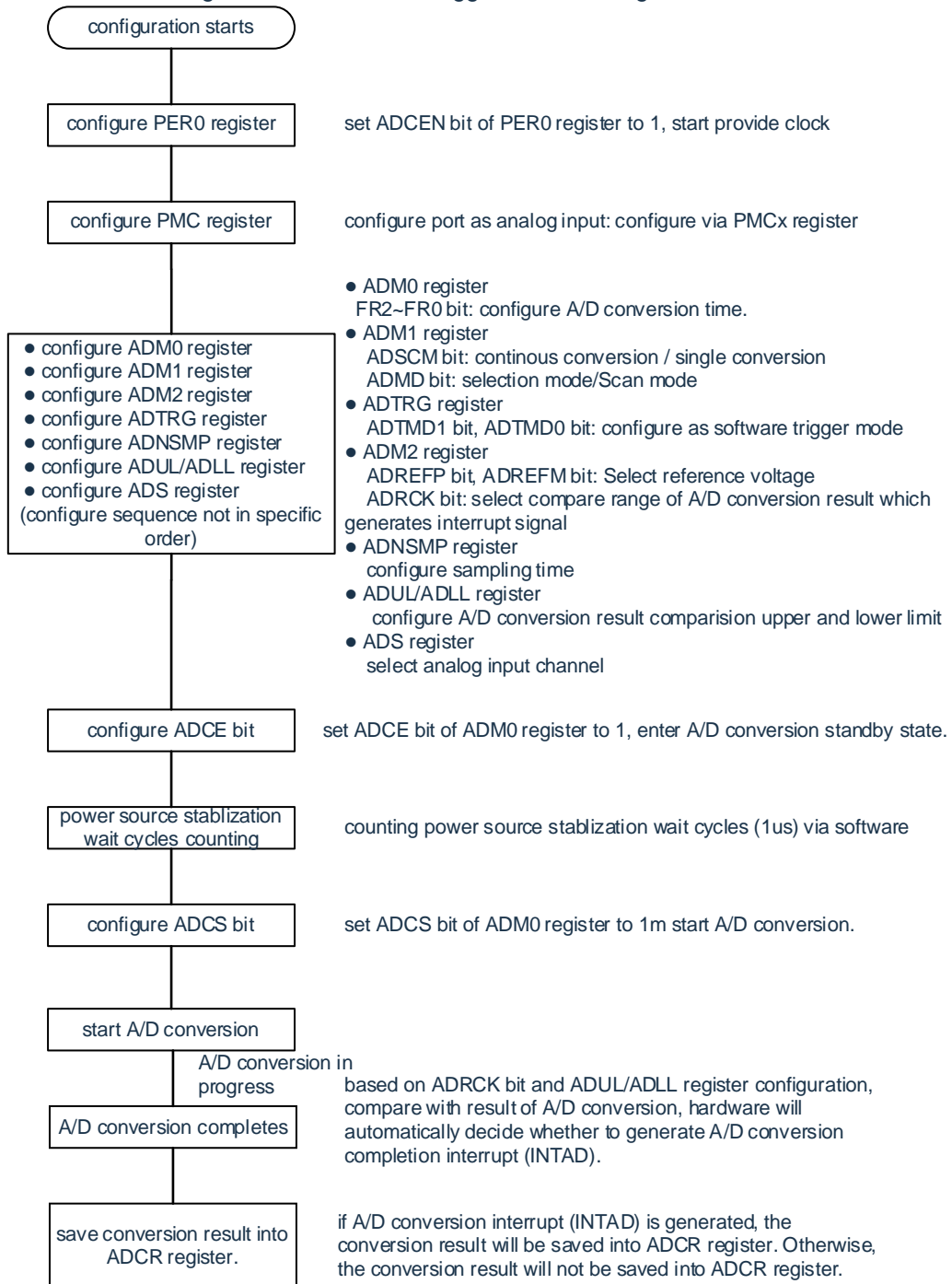


11.5 A/D converter setup flowchart

The A/D converter setup flowchart in each operation mode is described below.

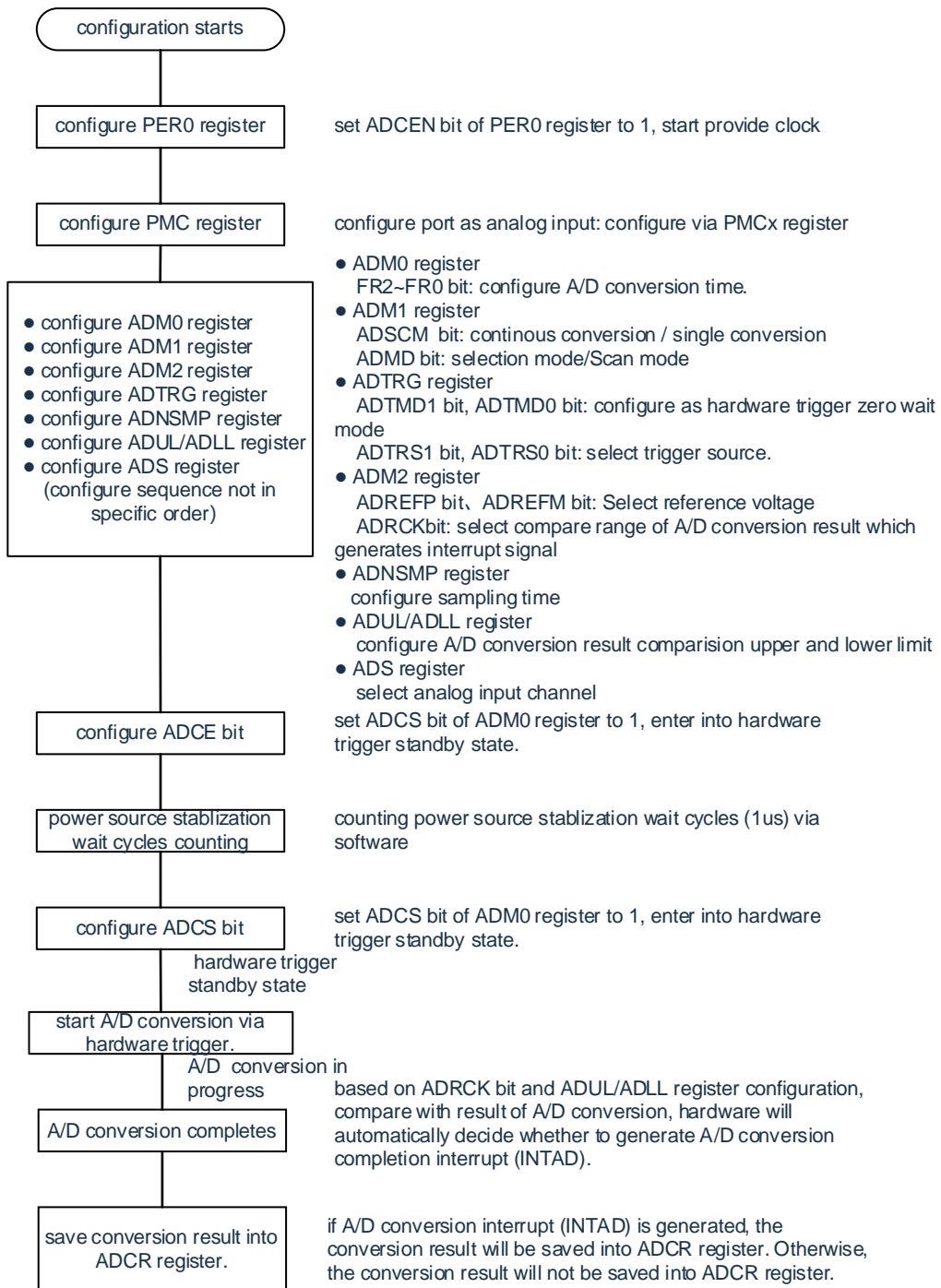
11.5.1 Software trigger mode settings

Figure 11-32 Software trigger mode settings



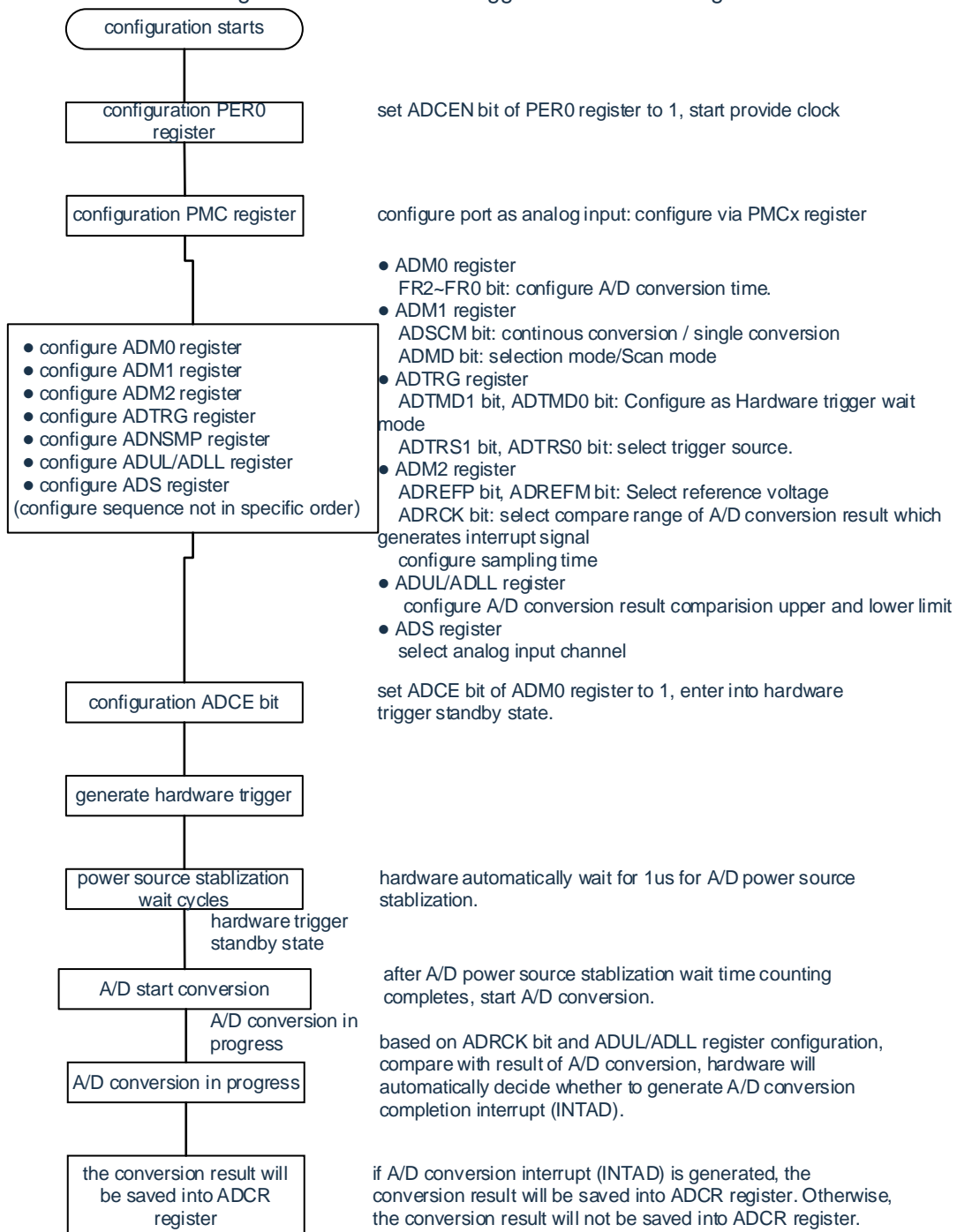
11.5.2 Hardware trigger no-wait mode settings

Figure 11-33 Hardware trigger no-wait mode settings



11.5.3 Hardware trigger wait mode settings

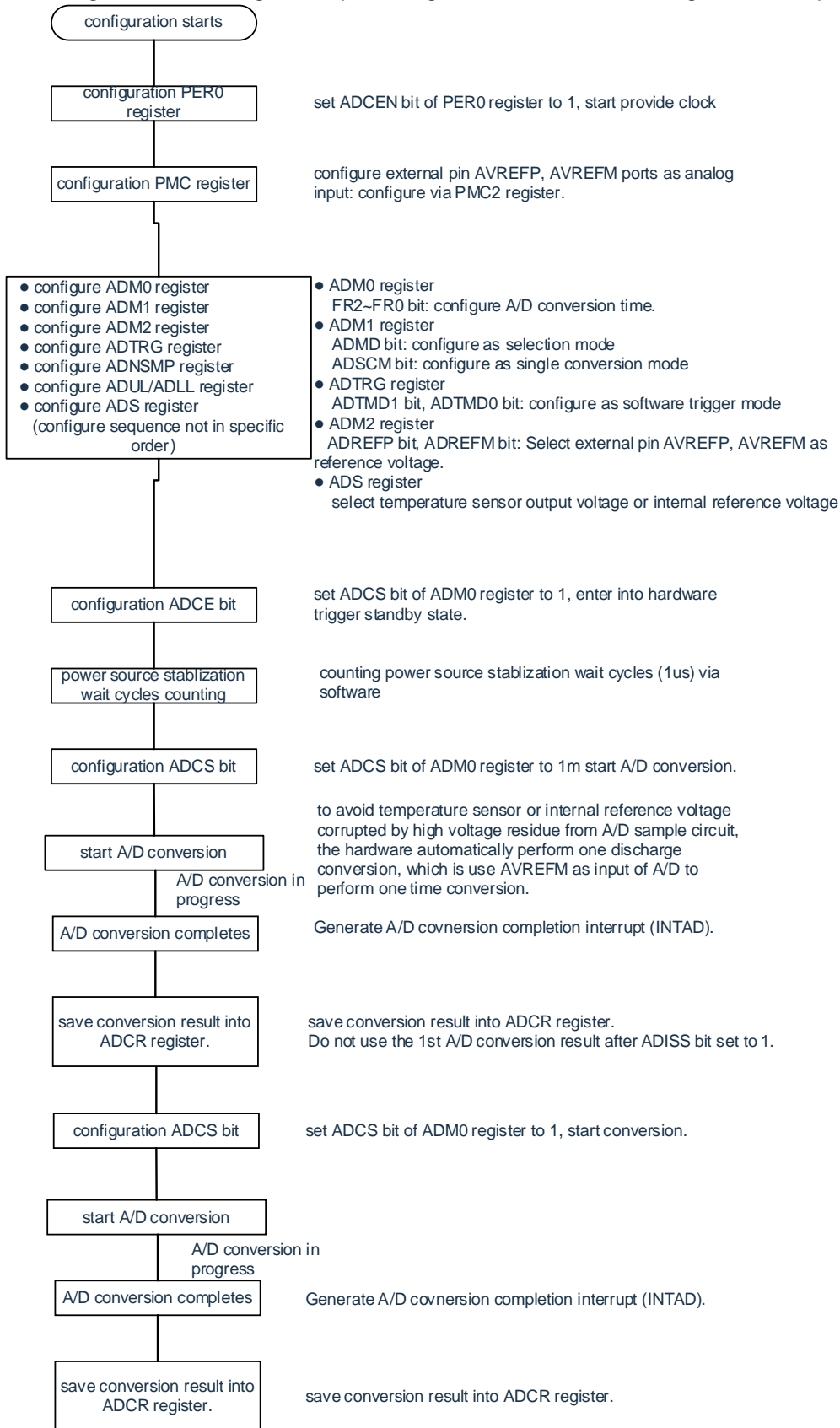
Figure 11-34 Hardware trigger wait mode settings



11.5.4 Settings when selecting the output voltage/internal reference voltage of the temperature sensor

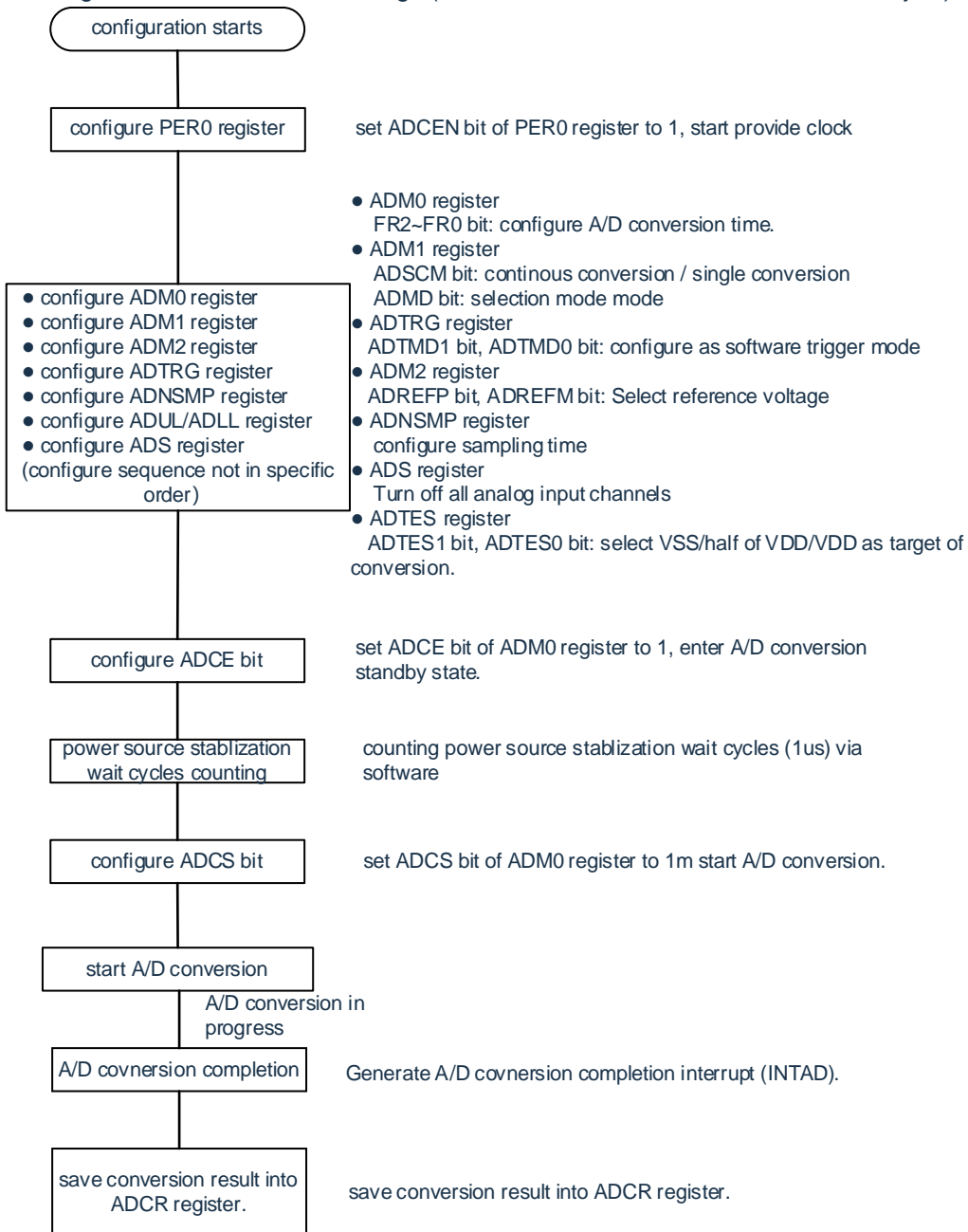
(Take software trigger mode, single conversion mode for example)

Figure 11-35 Settings when selecting the output voltage/internal reference voltage of the temperature sensor



11.5.5 Test mode settings

Figure 11-36 Test mode settings (VSS/half_VDD/VDD as the conversion object)



Chapter 12 Comparator

This product has a built-in 2-channel comparator.

12.1 Function of comparator

The comparator has the following functions:

- The input pin of the CMP1 can select an external port, internal reference voltage, and internal DAC reference voltage.
- The comparison results of comparator 0 and comparator 1 can be output by pins (VCOUT0, VCOUT1).

Table 12-1 Comparator feature summary

Item	Content
CMP	<ul style="list-style-type: none"> • 2-channel comparators (CMP0 and CMP1) • The negative end of the comparator can select a reference voltage: Optional external pin input on the negative side of the CMP0, built-in reference voltage of CMP0 and internal reference voltage (1.45V) Optional external pin input (4) for the negative end of the CMP1, built-in reference voltage of CMP1 and internal reference voltage (1.45V) • The internal reference voltage of the negative end may be set (256 steps) • The front end of the CMP0 selects the output of the PGA • The front end of the CMP1 can select external pin inputs (4) • When the input voltage of the positive end > the input voltage of the negative end, the output high level When the input voltage of the positive end is less than the input voltage of the negative end, the output level is low • Filter width of digital filter is optional • output inversion function • The comparison result can be output from the VCOUT0 (VCOUT1) • An effective edge of the comparator output can be detected and an interrupt signal generated • Combined with Timer4 to output TIMER WINDOW • Supports comparator positive hysteresis, negative hysteresis, and bilateral hysteresis with hysteresis voltages of 20mV, 40mV, and 60mV • Supports positive hysteresis, negative hysteresis, and bilateral hysteresis for comparators with selectable hysteresis voltages of 20mV, 40mV, and 60mV.

12.2 Structure of comparator

The registers that control the comparator are shown in Figure 12-1.

Figure 12-1 Block diagram of comparator 0

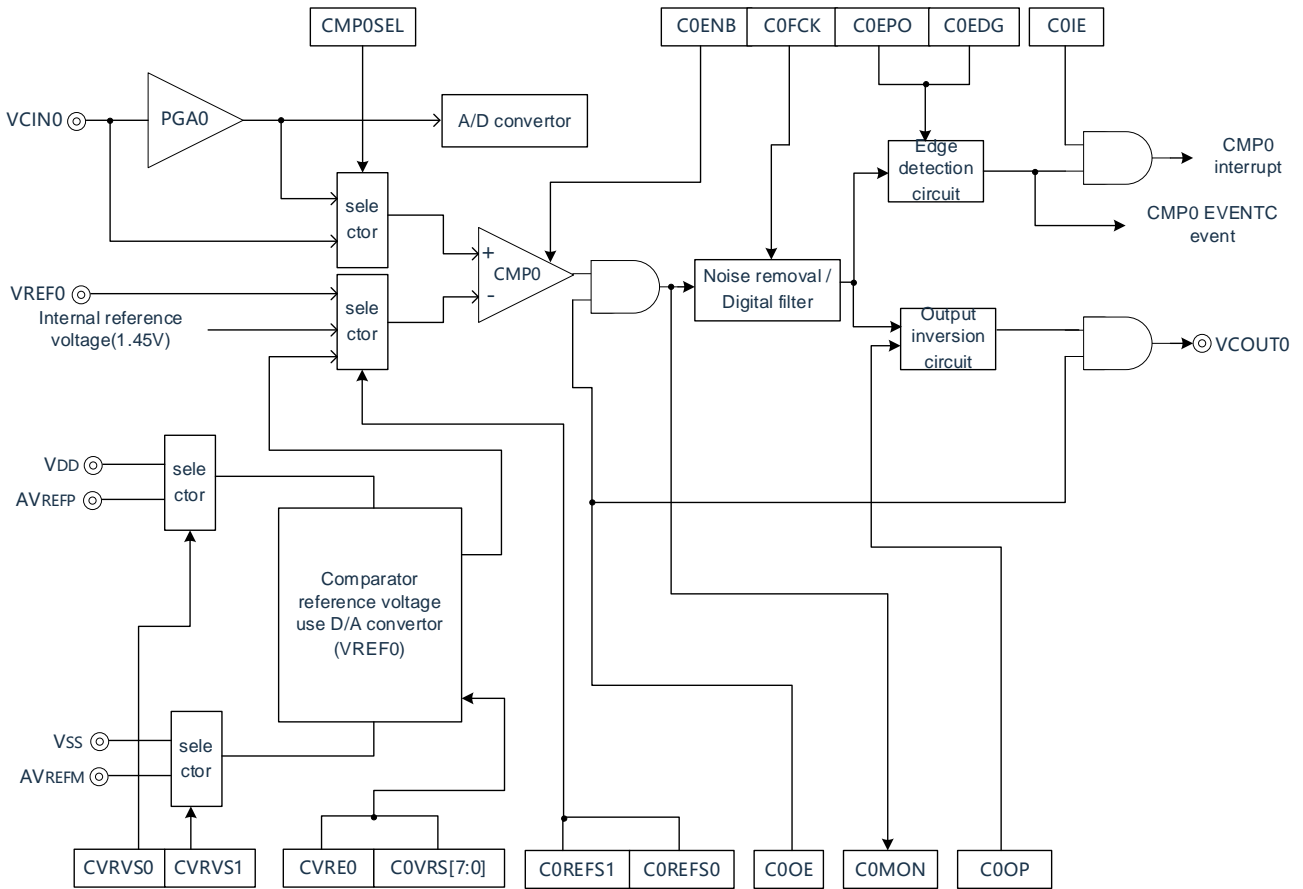
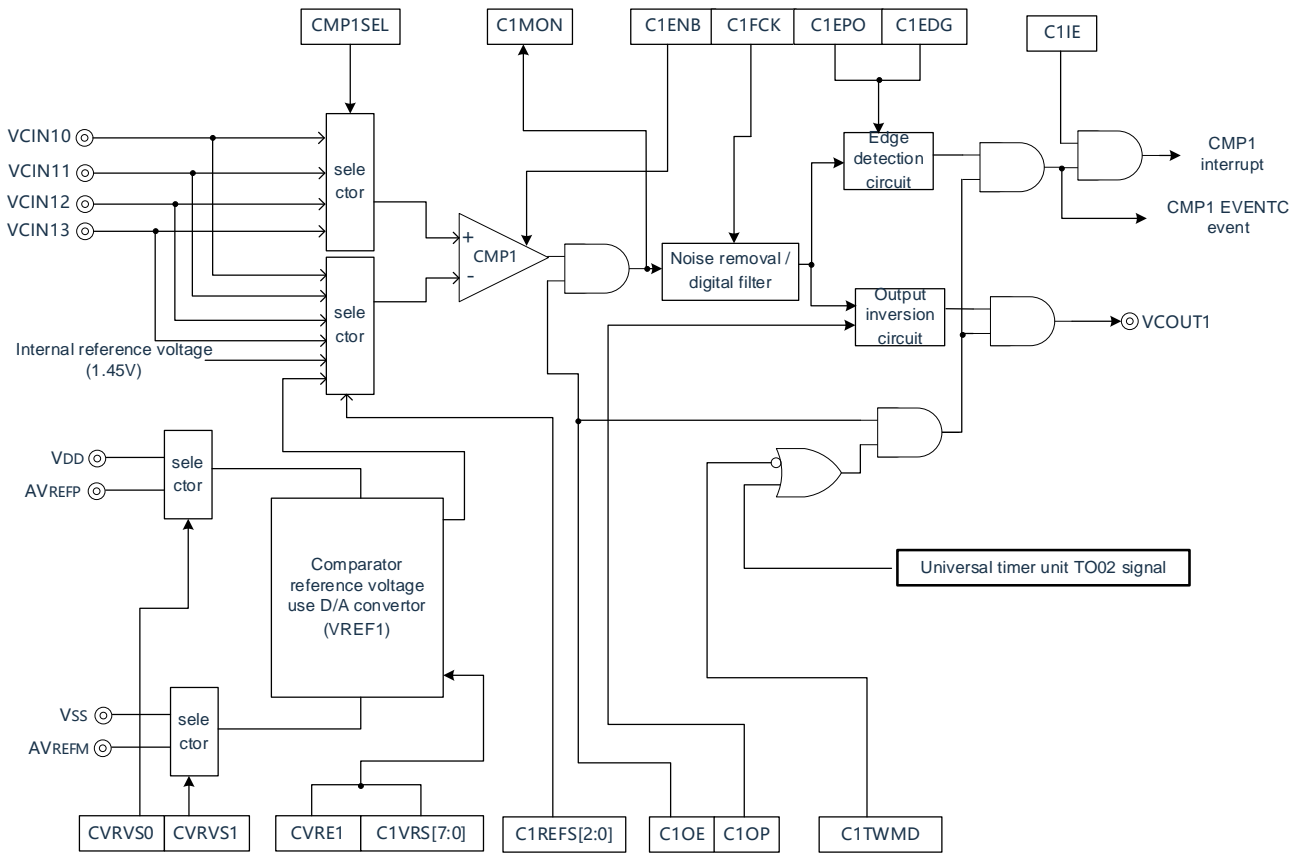


Figure 12-2 Block diagram of comparator 1



12.3 Registers for controlling comparator

Table 12-2 shows the registers controlling the comparator.

Table 12-2 Registers for controlling comparator

Register name	Symbol
Peripheral enable register 1	PER1
Comparator mode configuration register	COMPMDR
Comparator filter control register	COMPFIR
Comparator output control register	COMPOCR
Comparator built-in reference voltage control register	CVRCTL
Comparator built-in reference voltage selection register 0	C0RVM
Comparator built-in reference voltage selection register 1	C1RVM
Comparator 0 input selection control register	CMPSEL0
Comparator 1 input selection control register	CMPSEL1
Port mode control register	PMCxx
Port mode register	PMxx
Port multiplexing configuration register	PxxCFG

12.3.1 Peripheral enable register 1 (PER1)

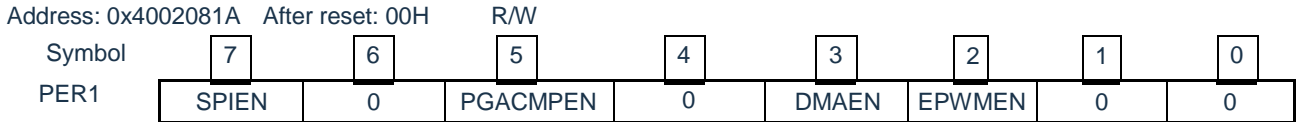
The PER1 register is a register that sets a clock that is allowed or prohibited to supply to each peripheral hardware. Reduce power consumption and noise by stopping clock supply to unused hardware.

You must set bit5 (PGACMPEN) to '1' when you want to use comparators.

The PER1 register is set by an 8-bit memory operation instruction.

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

Figure 12-3 Format of peripheral enable register 1(PER1)



PGACMPEN	Control of comparator input clock
0	Stop provide an input clock. <ul style="list-style-type: none"> • Cannot write the SFR used by the comparator. • The comparator is in a reset state.
1	Provides an input clock. <ul style="list-style-type: none"> • SFR that can read and write to the comparator.

Notice: To set the comparator, the PGACMPEN bit must first be set to "1".

When the PGACMPEN bit is "0", writes to the comparator's control register are ignored and the read values are initial (except Port Register (PMCxx, PMxx)).

12.3.2 Comparator mode configuration register (COMPMDR)

The COMPMDR register is a register that sets the comparator action enable/disable and detects the comparator output.

The CiENB bit is forbidden to be set to "0" when the comparator output is allowed (CiOE bit of COMPOCR register is set to "1").

Setting the CiENB bit to "1" (i=0,1) is prohibited in the following cases:

- The CMP negative input selects the built-in reference voltage while the built-in reference voltage action stops (the CVREi bit of the CVRCTL register is "0")

- The input of the CMP0 selects the output of the PGA, and when the PGA action stops (the CMPSEL0 bit of the CVRCTL register is '1' and the PGAEN bit of the PGAEN register is '0')

The COMPMDR register is set by an 8-bit memory operation instruction.

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

Figure 12-4 Format of comparator mode configuration register (COMPMDR)

Address: 40043840H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
COMPMDR	C1MON	0	0	C1ENB	C0MON	0	0	C0ENB

C1MON	Comparator 1 monitor flag ^{Notes 1, 2}
0	IVCMP1<Reference voltage of comparator 1, or comparator 1 is not running.
1	IVCMP1> Reference voltage for comparator 1

C1ENB	Enables comparator 1 operation
0	Disables comparator 1 operation.
1	Enables comparator 1 operation.

C0MON	Comparator 0 monitor flag ^{Note 1,2}
0	IVCMP0<Reference voltage of comparator 0, or comparator 0 is not running.
1	IVCMP0>Reference voltage of comparator 0

C0ENB	Enables comparator 0 operation
0	Disables comparator 0 operation.
1	Enables comparator 0 operation.

Note 1. It becomes "0" (initial value) immediately after the reset is released, or an undefined value if both the C0ENB bit and the C1ENB bit are set to "0" after the operation of the comparator is allowed.

2. Ignore the write value for this bit.

12.3.3 Comparator filter control register (COMPFIR)

The COMPFIR register is a control register for the digital filter. The COMPFIR register is set by an 8-bit memory operation instruction.

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

Figure 12-5 Format of comparator filter control register (COMPFIR)

Address: 40043841H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
COMPFIR	C1EDG	C1EPO	C1FCK1	C1FCK0	C0EDG	C0EPO	C0FCK1	C0FCK0

C1EDG	Comparator 1 edge detection selection ^{Note 1}
0	An interrupt request is generated by single edge detection of the comparator 1.
1	An interrupt request is generated by bilateral edge detection of the comparator 1.

C1EPO	Comparator 1 edge polarity switching ^{Note 1}
0	An interrupt request is generated by the rising edge of the comparator 1.
1	An interrupt request is generated by the descent edge of the comparator 1.

C1FCK1	C1FCK0	Comparator 1 filter selection ^{Note 1}
0	0	Comparator 1 has no filter.
0	1	Comparator 1 has a filter and samples it through the f_{CLK} .
1	0	Comparator 1 has a filter and samples it by $f_{CLK}/8$.
1	1	Comparator 1 has a filter and samples through $f_{CLK}/32$.

C0EDG	Comparator 0 edge detection selection ^{Note 2}
0	Interrupt requests are generated by single edge detection of comparator 0.
1	Interrupt requests are generated by bilateral edge detection of comparator 0.

C0EPO	Comparator 0 edge polarity switching ^{Note 2}
0	An interrupt request is generated by the rising edge of the comparator 0.
1	An interrupt request is generated by the descent edge of the comparator 0.

C0FCK1	C0FCK0	Comparator 0 filter selection ^{Note 2}
0	0	Comparator 0 has no filter.
0	1	Comparator 0 has a filter, through the f_{CLK} sampling.
1	0	Comparator 0 has a filter, and samples through $f_{CLK}/8$.
1	1	Comparator 0 has a filter, and samples through $f_{CLK}/32$.

Note:

- If C1FCK1~C1FCK0 bits, C1EPO bits and C1EDG bits are changed, interrupt request and event signals output to EVENTC may occur. These bits must be changed after setting the EVENTC ELSELR14 register (output of unlinked comparator 1) to "0". In addition, the IF of the interrupt request flag register must be cleared "0".

If C1FCK1~C1FCK0 bit is changed from '00B' (comparator1 has filter), the comparator 1 must use the interrupt request or the event signal output to EVENTC after 4 sampling before updating the output of the filter.
- If you change the C0FCK1~C0FCK0 bits, C0EPO bits, and C0EDG bits, you may generate a interrupt request for comparator 0 and event signals output to EVENTC. These bits must be changed after setting the EVENTC ELSELR13

register (output of unlinked comparator 0) to "0". In addition, the IF of the interrupt request flag register must be cleared "0".

If C0FCK1~C0FCK0 bit is changed from '00B' (comparator 0 without filter) to other values (comparator 0 with filter).

12.3.4 Comparator output control register (COMPOCR)

The COMPOCR register is a control register that sets the polarity of the comparator output, the permission/prohibition of the output, and the permission/prohibition of the interrupt output.

In the following cases, setting "1" to the CiOE bit of the COMPOCR register is prohibited (output enabled).
(i=0,1)

When the comparator action stops (the CiENB bit of the COMPMDR register is "0")

The CMP negative input selects the built-in reference voltage while the built-in reference voltage action stops (the CVREi bit of the CVRCTL register is "0")

The input of the CMP0 selects the output of the PGA, and the PGA action stops (the CMPSEL0 bit of the CVRCTL register is "1" and the PGAEN bit of the PGAEN register is "0")

The COMPOCR register is set by an 8-bit memory operation instruction.

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

Figure 12-6 Format of comparator output control register (COMPOCR)

Address: 40043842H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
COMPOCR	C1OTWMD	C1OP	C1OE	C1IE	0	C0OP	C0OE	C0IE

C1OTWMD	Comparator 1 TIMER WINDOW output mode control bit ^{Note 1}
0	Comparator 1 normal output mode (controlled by C1OE)
1	Comparator 1 TIMER WINDOW output mode (co-controlled by TO02 and C1OE)

C1OP	Selection of VCOU1 output polarity
0	Comparator 1 output from VCOU1.
1	Inverted output of comparator 1 from VCOU1.

C1OE	Enables VCOU1 pin output ^{Note 2,3}
0	Disables comparator1 VCOU1 pin output.
1	Enables comparator1 VCOU1 pin output.

C1IE	Enables Comparator 1 interrupt request ^{Note 4}
0	Disables the comparator 1 interrupt request.
1	Enables the comparator 1 interrupt request.

C0OP	Selection of VCOU0 output polarity
0	Comparator 0 output from VCOU0.
1	Inverted output of comparator 0 from VCOU0.

C0OE	Enables VCOU0 pin output ^{Note 5,6}
0	Disables comparator0 VCOU0 pin output.
1	Enables comparator0 VCOU0 pin output. ^{Note 4, 8}

C0IE	Enables Comparator 0 interrupt request ^{Note 7}
0	Disables the comparator 0 interrupt request.
1	Enables the comparator 0 interrupt request.

- Note 1. When comparator 1 uses the TIMER WINDOW mode, the bit7 (C1EDG) of register COMPFIR must be set to "1".
The C1OE and C1OTWMD bits cannot be set at the same time. Set the C1OTWMD bit first, and then set the C1OE bit to "1".
- Note 2. When the C1OE bit is rewritten, a comparator 1 interrupt request and an EVENTC event may be generated. Rewrite this bit after setting the ELSELR14 register of EVENTC to 0 (not linked to the output of comparator 1). Also, initialize the IF bit of the interrupt request flag register (no interrupt request) after rewriting the C1OE bit.
- Note 3. When the result of comparator 1 is output to the pin, Pxx, PMxx, PMCxx of this pin must be set to 0.
- Note 4. When the C0OE bit is rewritten, a comparator 0 interrupt request and an EVENTC event may be generated. Rewrite the C0OE bit after setting the ELSELR13 register of EVENTC to 0 (not linked to the output of comparator 0). Also, initialize the IF bit of the interrupt request flag register (no interrupt request) after rewriting the C0OE bit.
- Note 5. If C1IE is changed from 0 (disable interrupt request) to 1 (enable interrupt request), the IF of the interrupt request flag register may become 1 (with interrupt request), so interrupt must be used after clearing IF of the interrupt request flag register.
- Note 6. When the result of comparator 0 is output to the pin, Pxx, PMxx, PMCxx of this pin must be set to 0.
- Note 7. If C0IE is changed from 0(disable interrupt request) to 1 (enable interrupt request), the IF of the interrupt request flag register may become 1, so interrupt must be used after clearing IF of the interrupt request flag register.

12.3.5 Comparator built-in reference voltage control register (CVRCTL)

The CVRCTL register is a register that sets the built-in reference voltage permit/stop action of the comparator.

The CVRCTL register is set by an 8-bit memory operation instruction.

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

Notice The CVRVS_i bit of the CVRCTL register is overridden when the built-in reference voltage stop action (CVRE_i=0)

Figure 12-7 Format of comparator built-in reference voltage control register (CVRCTL)

Address: 40043843H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CVRCTL	0	0	CVRE1	CVRVS1	0	0	CVRE0	CVRVS0

CVRE1	Control bit with built-in reference voltage 1
0	Disables operation of built-in reference voltage 1
1	Enables operation of built-in reference voltage 1

CVRVS1	Ground side selection bit with built-in reference voltage
0	Internal reference voltage ground side select V _{SS}
1	Internal reference voltage ground side select AVREFM ^{Note 1}

CVRE0	Control bit with built-in reference voltage 0
0	Disables operation of built-in reference voltage 0
1	Enables operation of built-in reference voltage 0

CVRVS0	Power supply side selection bit with built-in reference voltage
0	Power supply side selection V _{DD} with internal reference voltage
1	Internal reference voltage power supply side selection AVREFP ^{Note 2}

Note 1. The P21 pin is used for both AVREFM and VCIN13, so it is prohibited to set the CVRVS1 bit to "1" when the P21 pin is used as the input signal for CMP1.

Note 2. The P20 pin is used for both AVREFP and VCIN12, so it is prohibited to set the CVRVS0 bit to "1" when P20 pin is used as the input signal for CMP1.

12.3.6 Comparator built-in reference voltage selection register (CiRVM)

The CiRVM register is a register that sets the built-in reference voltage of the comparator.

When the built-in reference voltage stops (CVREi=0), rewrite the CiRVM register

The CVRCTL register is set by an 8-bit memory operation instruction.

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

Figure 12-8 Format of comparator built-in reference voltage selection register i (CiRVM)

Address: 400438434H(C0RVM),400438435H(C1RVM), After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CiRVM	CiRVS7	CiRVS6	CiRVS5	CiRVS4	CiRVS3	CiRVS2	CiRVS1	CiRVS0

CiRV S7	CiRV S6	CiRV S5	CiRV S4	CiRV S3	CiRV S2	CiRV S1	CiRV S0	Setting of built-in reference voltage of comparator
0	0	0	0	0	0	0	0	{{(AVREFP or VDD)/256}x0
0	0	0	0	0	0	0	1	{{(AVREFP or VDD)/256}x1
0	0	0	0	0	0	1	0	{{(AVREFP or VDD)/256}x2
0	0	0	0	0	0	1	1	{{(AVREFP or VDD)/256}x3
...								...
1	1	1	1	1	1	0	0	{{(AVREFP or VDD)/256}x252
1	1	1	1	1	1	0	1	{{(AVREFP or VDD)/256}x253
1	1	1	1	1	1	1	0	{{(AVREFP or VDD)/256}x254
1	1	1	1	1	1	1	1	{{(AVREFP or VDD)/256}x255

12.3.7 Comparator 0 input signal selection control register (CMPSEL0)

The CMPSEL0 register is a selection register for the input signal of the positive end and the negative end of the comparator 0.

When comparator 0 stops (C0ENB=0), override the CMPSEL0 register.

The CMPSEL0 register is set by an 8-bit memory operation instruction.

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

Figure 12-9 Format of comparator 0 input signal selection control register (CMPSEL0)

Address: 4004384AH After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CMPSEL0	CMP0SEL	0	0	0	0	0	C0REFS1	C0REFS0

CMP0SEL	Comparator 0 positive input signal selection bit
0	Select External Pin (VCIN0 Pin)
1	Select PGA output signal

C0REFS1	C0REFS0	Comparator 0 negative input signal selection bit
0	0	Select the built-in reference voltage VREF0
0	1	Select Internal Reference Voltage (1.45V)
1	0	Select an external pin (IVREF0 pin)
1	1	Disable settings

12.3.8 Comparator 1 input signal selection control register (CMPSEL1)

The CMPSEL1 register is a selection register for the input signal of the positive end and the negative end of the comparator 1.

When comparator 1 stops (C1ENB=0), rewrite the CMPSEL1 register.

The CMPSEL1 register is set by an 8-bit memory operation instruction.

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

Figure 12-10 Format of comparator 1 input signal selection control register (CMPSEL1)

Address: 4004384BH After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CMPSEL1	CMP1SEL1	CMP1SELO	0	0	0	C0REFS1	C0REFS1	C0REFS0

CMP1SEL1	CMP1SELO	Comparator 1 positive input signal selection bit
0	0	Selects external pin (VCIN10 Pin)
0	1	Selects external pin (VCIN11 Pin)
1	0	Selects external pin (VCIN12 Pin)
1	1	Selects external pin (VCIN13 Pin)

C0REFS2	C0REFS1	C0REFS0	Negative input signal selection bit of comparator 1
0	0	0	Selects the built-in reference voltage VREF1
0	0	1	Selects internal reference voltage (1.45V)
0	1	0	Selects external pin (VCIN10 Pin)
0	1	1	Selects external pin (VCIN11 Pin)
1	0	0	Selects external pin (VCIN12 Pin)
1	0	1	Selects external pin (VCIN13 Pin)
1	1	0	Disable settings
1	1	1	Disable settings

Notice: When switching the analog input of the CMP1, the switching interval must be more than 3 us in order to prevent the through current before the two input signals.

12.3.9 Registers controlling port functions of analog input pins

When using the VCIN0 pin, VCIN10-VCIN1 3 pin and VREF0 pin as analog inputs to the comparator, the bits of port mode register (PMxx) and PMCxx.

When using the functions of VCOUT0 and VCOUT1, port register (Pxx), port mode register (PMxx) and port mode control register (PMCxx) must be set. Please refer to Chapter 2 Pin Functions for details.

12.4 Operation instructions

Comparator 0 and Comparator 1 can be run independently. The setting method is the same as operation. CMP0 and PGA can be combined and linked.

Table 12-3 shows the setting procedure for independent operation and linkage of the comparator.

Table 12-3 Set-up steps for comparator-related registers

Step	Register	Bit	Set value
1	PGACTL	PGAVG0/1/2	Select gain ^{Note 3}
2	PGACTL	PVRVS	0 (Vss pin selection) ^{Note 3}
3	PGACTL	PGAEN	1 (Enable operation) ^{Note 3}
4	Wait for PGA stability time (minimum 10μs)		
5	COMPSELi	CMP0SEL/CMP1SELi 1	comparator i positive input selection
6	COMPSELi	CiREFS	comparator i negative input selection
7	CiRVM	CiRVSn	Set the value of the built-in reference voltage
8	CVRCTL	CVRVSi	Select power and ground with built-in reference voltage
9	CVRCTL	CVREi	1 (built-in reference voltage i allowed to run)
10	Wait for stability time of reference voltage (minimum 20μs)		
11	Set VCIN0, VCIN1x, IVREF0 pin (input), PGAI ^{Note 3} to the analog input function. Function selection for PMCxx, VCIN0 pins, VCi and IVREFi pins. Set the PMCxx bit to "1" (analog input). Set the PMxx bit to "1" (input mode).		
12	COMPMDR.	CiENB	1 (Enable operation)
13	Wait for the stability time of the comparator (minimum 3μs)		
14	COMPFIR	CiFCK	Select a sampling clock using or without a digital filter.
		CiEOP, CiEDG	Select the edge detection criteria (rising, falling, or bilateral) for the interrupt request.
15	COMPOCR	CiOP, CiOE	Sets the output of the VCOUTi (select Polarity, set enable or disable output). Refer to "12.4.4 Output of comparator i (i=0,1)".
		CiIE	Sets the output that allows or disables the interrupt request. Refer to "12.4.4 Output of comparator i (i=0,1)".
		C1OTWMD	Set TIMER WINDOW output enable/disable for Comparator 1
16	MKxx ^{Note 1}	MKL	When using interrupts: select mask interrupt.
17	IFxx ^{Note 1}	IFL	When using interrupts: 0 (non-disruptive request: initialization) ^{Note 2}

Note 1 MKxx and IFxx are comparator interrupt control register, see "Chapter 20 Interrupt Function" for details.

Note 2 After the comparator is set, an undesired interrupt request may occur during stable operation and the interrupt request flag must be initialized.

Note 3 Comparator 0 must be set when interacting with PGA

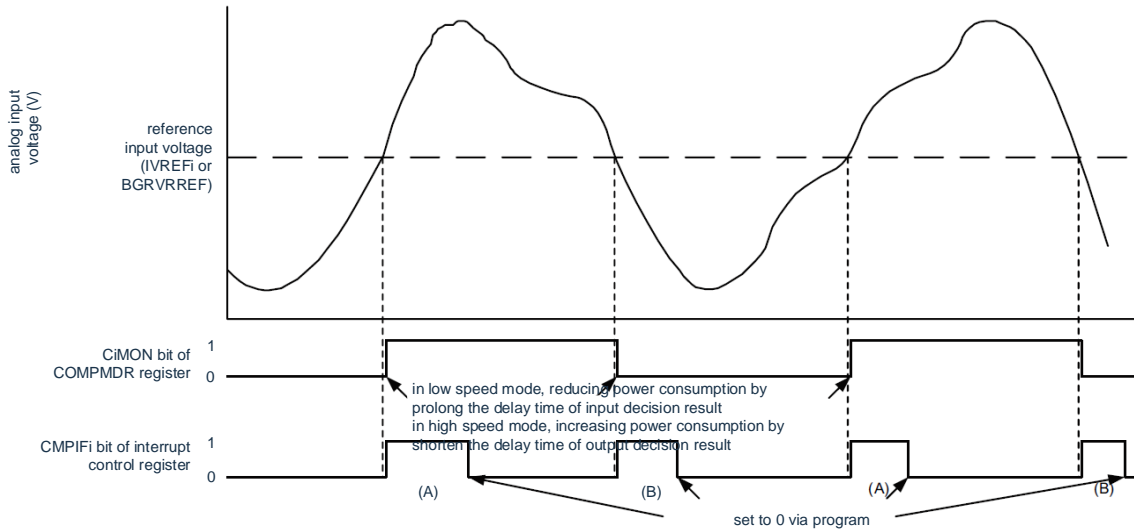
Remark i=0, 1, n=0-7, x=0-3

An example of operation of the comparator i (i=0,1) is as follows Figure 12-11. In the basic mode, when the analog input voltage is higher than the reference input voltage, the CiMON bit of the COMPMDR register is "1"; When the analog input voltage is lower than the reference input voltage, the CiMON bit is "0".

To use the comparator i interrupt, the CiIE bit of the COMPOCR register must be set to "1" (enable interrupt request). At this time, if the comparison result changes, an interrupt request of the comparator i is generated. For more information on interrupt requests, refer to "12.4.2 Comparator i interrupt (i=0,1)".

Figure 12-11 Operation example of comparator i (i=0,1) (basic mode)

- basic mode operation example



Notice The above figure shows the case when bits CiFCK1 to CiFCK0 of the COMPFIR register are "00B" (no filter) and the CiEDG bit is "1" (both edges) (CMPiFi is limited to (A) when the CiEDG bit is "0" and the CiEPO bit is "0" (rising edge), and CMPiFi is limited to (B) when the CiEDG bit is "0" and the CiEPO bit is "1" (falling edge)).

12.4.1 Digital filter of comparator i (i=0,1)

The comparator i has a built-in digital filter, which can select the sampling clock through the CiFCK1~CiFCK0 bit of the COMPFIR register. The output signal of comparator i is sampled according to each sampling clock, and the digital filter outputs the sampling value.

Figure 12-12 is a result of the digital filter of the comparator i, Figure 12-13 is an example of a digital filter and interrupt operation of a digital comparator i (i=0,1) of comparator i.

Figure 12-12 Comparator i digital filter (i=0,1) and edge detection structure

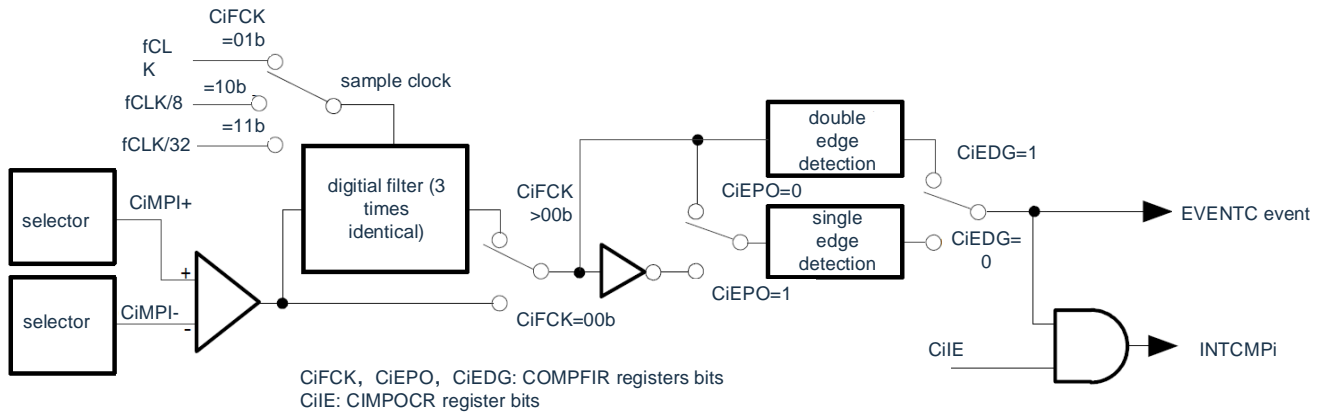
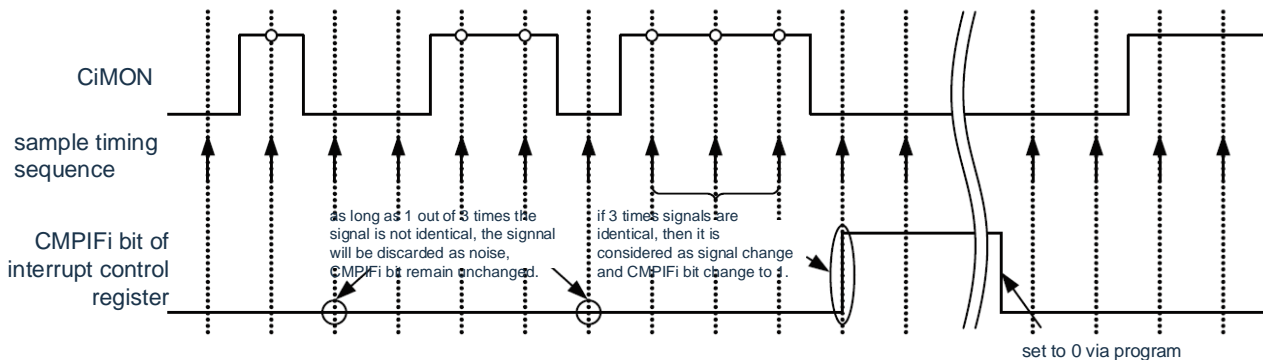


Figure 12-13 Example of digital filter and interrupt operation for comparator i (i=0, 1)



Notice The above figure is an example of the COMPFIR register running when the CiFCK1~CiFCK0 bit is "01B" or "11B" (with a digital filter).

12.4.2 Comparator i interrupts (i=0,1)

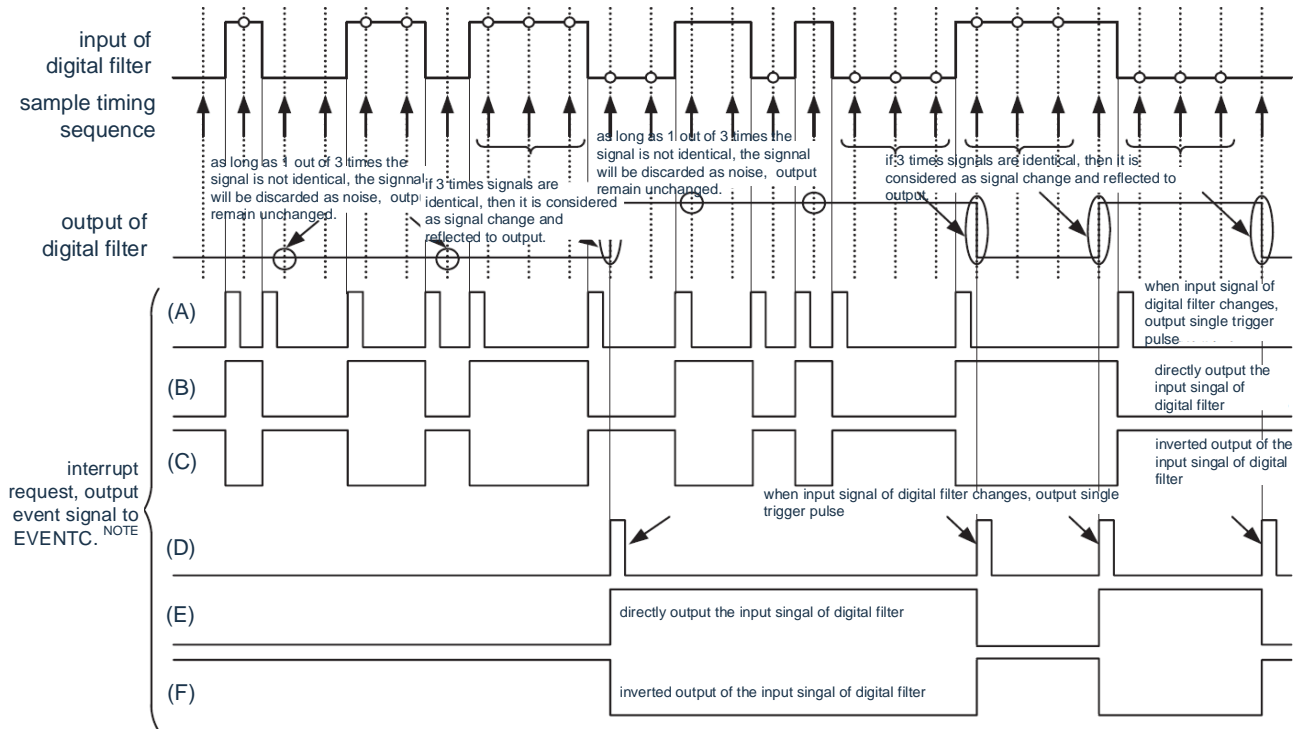
The comparator generates two interrupt requests for the comparator 0 and comparator 1. The comparator i interrupt has 1 priority specify flag, interrupt mask flag, interrupt request flag and interrupt vector respectively.

To use the comparator i interrupt, the CiIE bit of the COMPOCR register must be set to "1" (enable output of the interrupt request). The generation condition of the interrupt request is set through the COMPFIR register, and a digital filter can be added to the output of the comparator. The digital filter can select three kinds of sampling clock. Refer to "12.3.3 Comparator filter control register (COMPFIR)" and "12.3.4 Comparator output control register (COMPOCR)".

12.4.3 Event signals output to the linkage controller (EVENTC)

An event signal output to the COMPFIR is generated by detecting the output edge of the digital filter set by the EVENTC register. However, unlike an interrupt request, the event signal is always output to the EVENTC regardless of the CiIE bit of the COMPOCR register. You must set the selection of the event output target and the stop of the event link through the EVENTC's ELSELR13 register and ELSELR14 register.

Figure 12-14 Operation of digital filters, interrupt requests and output of event signals to EVENTC



Note When the CiIE bit (i=0, 1) is '1', the interrupt request and the event signal output to the EVENTC are identical waveforms.

(A), (B), (C) waveforms are the case that the CiFCK bit (i=0,1) of the COMPFIR register is '00B' (without a digital filter), and the waveforms of (D), (E), and (F) are the case that the CiFCK bit (i=0,1) of the COMPFIR register is '01B', '10B', or '11B' (with a digital filter). (A), (D) are cases where the CiEDG bit is 1 (bilateral edges), (B), (E) is the CiEDG bit '0' and the CiEPO bit '0' (rising edge), (C), (F) is cases where the CiEDG bit is '0' and the CiEPO bit is '1' (falling edge).

12.4.4 Output of comparator i (i=0,1)

The CiOE bit of the COMPOCR register can be used to set whether the comparison result of the comparator is output to an external pin, and the CiOP bit of the COMPOCR register can be used to set the output polarity (positive or negative output). Refer to “12. 3. 4 Comparator output control register (COMPOCR)” for the corresponding register settings and comparator outputs.

To output the comparator's comparison results to the VCOUTi pin, you must set the port as follows (after reset, the port defaults to the input state):

- ① Set the mode of the comparator (Steps 2-5 of "Table 12-3 Set-up steps for comparator-related registers").
- ② Sets the VCOUTi output of the comparator (sets the COMPOCR register, selects polarity, and allows output).
- ③ Set the port mode control register PMCxx corresponding to the output pin of VCOUTi to "0".
- ④ Set the bit of the port output latch register Pxx corresponding to the output pin of VCOUTi to "0".
- ⑤ Set the port direction register PMxx corresponding to the output pin of VCOUTi to output (output from the pin).

12.4.5 Stop and enable comparator clock supply

In the case of stopping the comparator clock by setting the peripheral admission register 1 (PER1), you must follow these steps:

- ① Set the CiENB bit of the COMPMDR register to "0" (to stop the operation of the comparator).
- ② Set the IF bit of the interrupt request flag register to "0" (clear the interrupts that are not needed before the comparator stops running).
- ③ Set the PGACMPEN bit of PER1 register to "0".

If you stop the clock by setting the PER1 register, all internal registers for the comparator are initialized, so you set the register according to the steps in Table 12-3.

Note:

1. If the comparator n reference voltage selection bit (CnVRF) of the comparator mode setting register (COMPMDR) is set to '1', the output of the temperature sensor cannot be A/D converted.
2. If DMA start is permitted in one of the following states, DMA transfer is started and an interrupt is generated after transfer. Therefore, the monitor flag (CnMON) of the comparator must be confirmed as necessary to allow the DMA to start.
 - An interrupt request (CnEDG=0) is generated by unilateral edge detection of the comparator and an interrupt request (CnEPO=0) and IVCMP>IVREF (or internal reference voltage 1.45V).
 - An interrupt request (CnEDG=0) is generated by unilateral edge detection of the comparator and an interrupt request (CnEPO=1) and IVCMP<IVREF (or internal reference voltage 1.45V).(n=0, 1)

Chapter 13 Programmable Gain Amplifier (PGA)

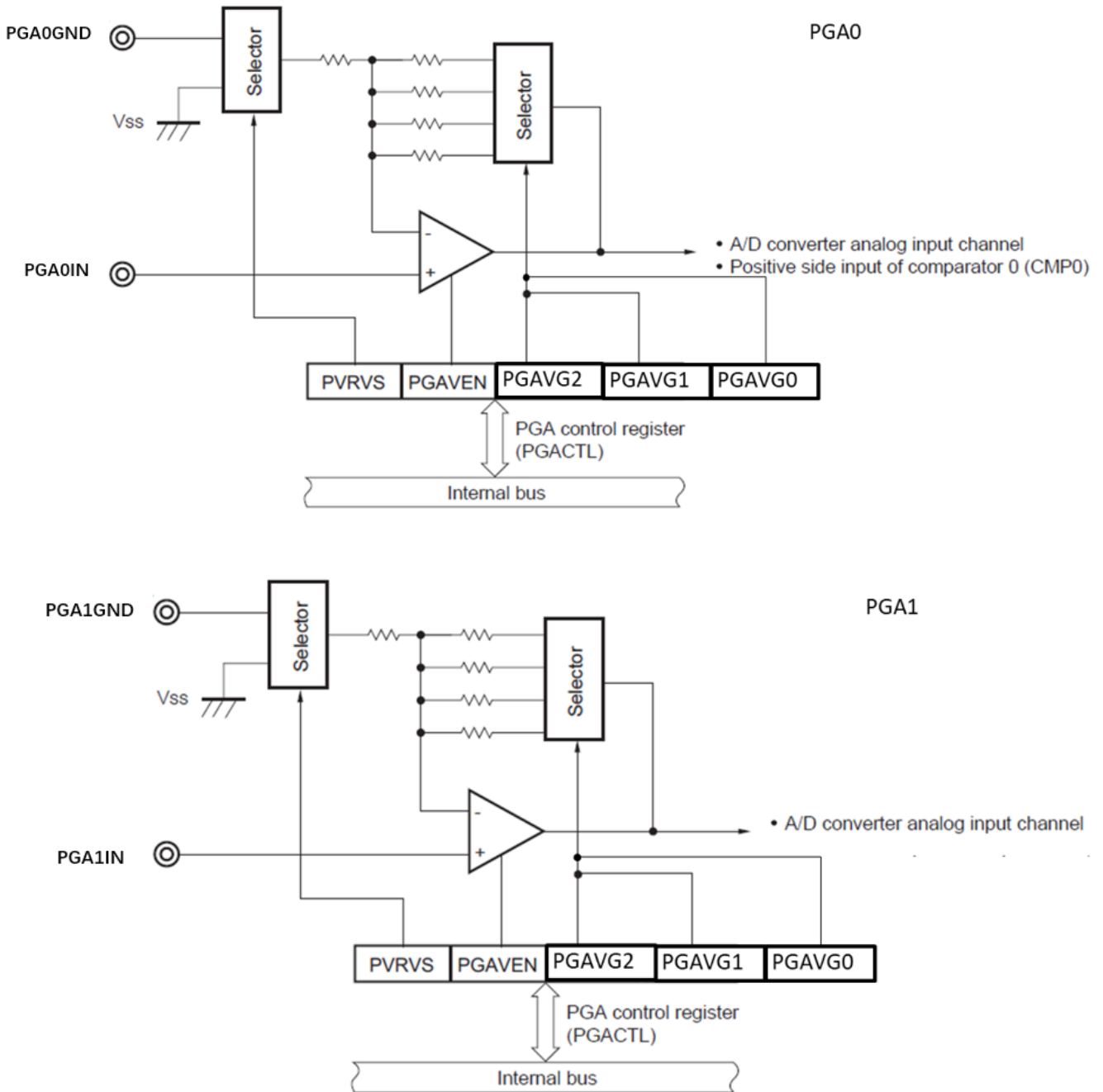
13.1 Function of programmable gain amplifier

This product has two built-in programmable gain amplifiers (PGA0 and PGA1) with the following functions.

- 7 choices of amplification gain per PGA: 4x, 8x, 10x, 12x, 14x, 16x, 32x
- External pin can be selected as the ground for the PGA negative feedback resistor
- The output of PGA0 can be selected as an analog input for the A/D converter or an analog input on the positive side of comparator 0 (CMP0)
- The output of GA1 can be selected as an analog input for the A/D converter

13.2 Structure of programmable gain amplifier

Figure 13-1: Block diagram of programmable gain amplifier



13.3 Registers for programmable gain amplifier

Table 13-1 Registers for controlling programmable gain amplifier

Register name	Symbol
Peripheral enable register 1	PER1
Programmable gain amplifier control register	PGA0CTL
Programmable gain amplifier control register	PGA1CTL
Port mode control register	PMCxx
Port mode register	PMxx

13.3.1 Peripheral enable register 1 (PER1)

The PER1 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.

The bit5 (PGACMPEN) of this PER1 register must be set to '1' when using a programmable gain amplifier.

The PER1 register is set by 1-bit or 8-bit memory operation instructions.

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

Figure 13-2 Format of peripheral enable register 1 (PER1)

Reset value: 00H

R/W

Symbol	7	6	5	4	3	2	2	1	0
PER1	SPIEN	0	PGACMPEN	0	DMAEN		EPWMEN	0	0

PGACMPEN	Control of input clock of comparator/programmable gain amplifier
0	Stops the input clock supply. Comparator or programmable gain amplifier registers are not writable Comparator or programmable gain amplifier is in the reset status
1	Enables the input clock supply. Comparator or programmable gain amplifier registers are readable and writable.

Notice

Before configuring the comparator or the register of the programmable gain amplifier, confirm that the bit bit of the PGACMPEN is set to 1.

If PGACMPEN=0, writing to the comparator or the programmable gain amplifier control register is invalid, and all read-out values are default values. (except for Port Mode Register (PMXX) and Port Register PXX).

13.3.2 Programmable gain amplifier control register (PGAnCTL)

The PGA0CTL and PGA1CTL registers are used to control the programmable gain amplifier start, stop and amplification.

The PGA0CTL and PGA1CTL registers can be set by a 1-bit or 8-bit memory operation instruction. This register is reset to 00H after a reset signal is generated.

Figure 13-3 Format of PGA control register (PGAnCTL)

Reset value: 00H R/W

	7	6	5	4	3	2	1	0
PGAnCTL	PGAnEN	-	-	-	PVRVS ^{Note}	PGAnVG2	PGAnVG1	PGAnVG0

n=0,1

PGAnEN	Programmable gain amplifier operation control
0	Stops the amplifier operation
1	Enables the amplifier operation

PVRVS	Selection of feedback resistor ground
0	Selects Vss
1	Selects PGAnGND pin

PGAnVG2	PGAnVG1	PGAnVG0	PGAn gains
0	0	0	4 x
0	0	1	8 x
0	1	0	10 x
0	1	1	12 x
1	0	0	14 x
1	0	1	16 x
1	1	0	32 x
Others			Disable settings

Notice: With PGAnEN set to 1, the programmable gain amplifier requires a 10us stabilization time to operate.

13.3.3 Registers controlling port functions of analog input pins

When using the PGA0IN pin, PGA1IN pin, PGA0GND pin, and PGA1GND pin as analog inputs to a programmable gain amplifier, you must set the bits in the Port Mode Register (PMxx) and the bits in the Port Digital/Analog Control Register (PMCxx) corresponding to each port to "1".

13.4 Operation of programmable gain amplifier

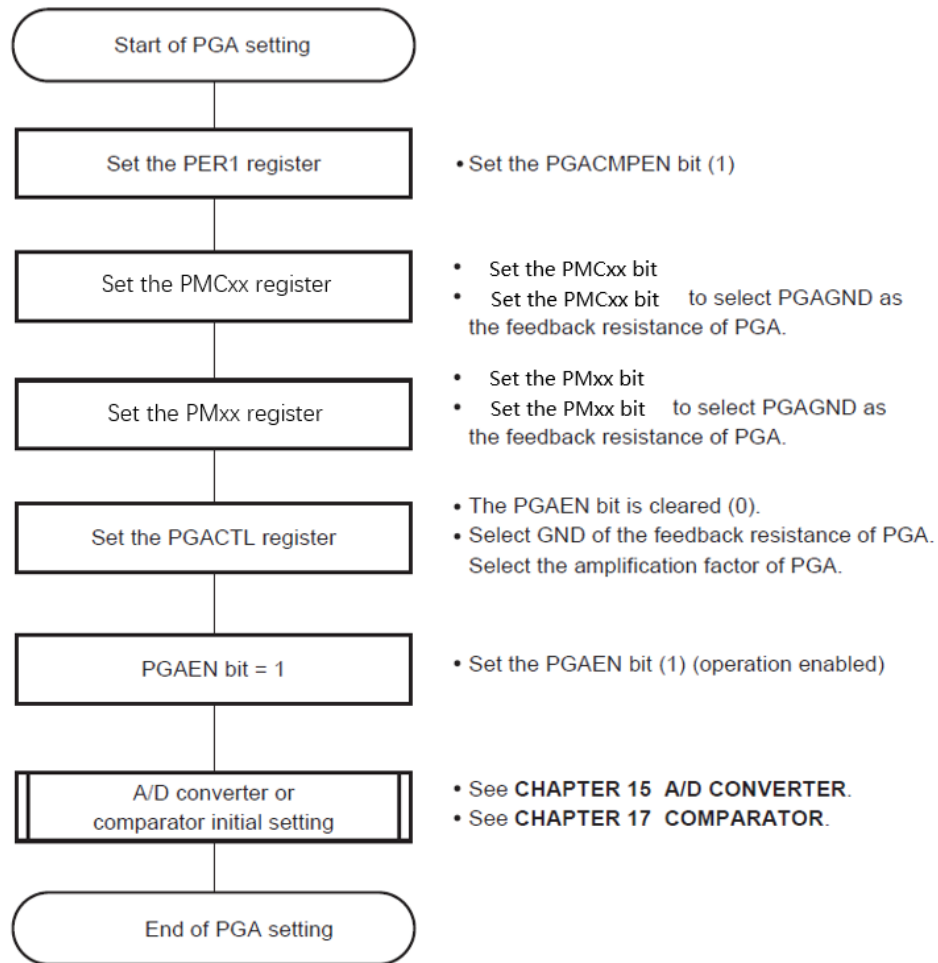
The analog voltage input from the PGAIN pin is amplified with seven choices of amplification gain: 4x, 8x, 10x, 12x, 14x, 16x, and 32x.

The amplified voltage may be used for analog input of the A/D converter and positive input signal of the comparator 0 (CMP 0).

The steps for starting and stopping the programmable gain amplifier are as follows.

13.4.1 Starting operation steps of programmable gain amplifier

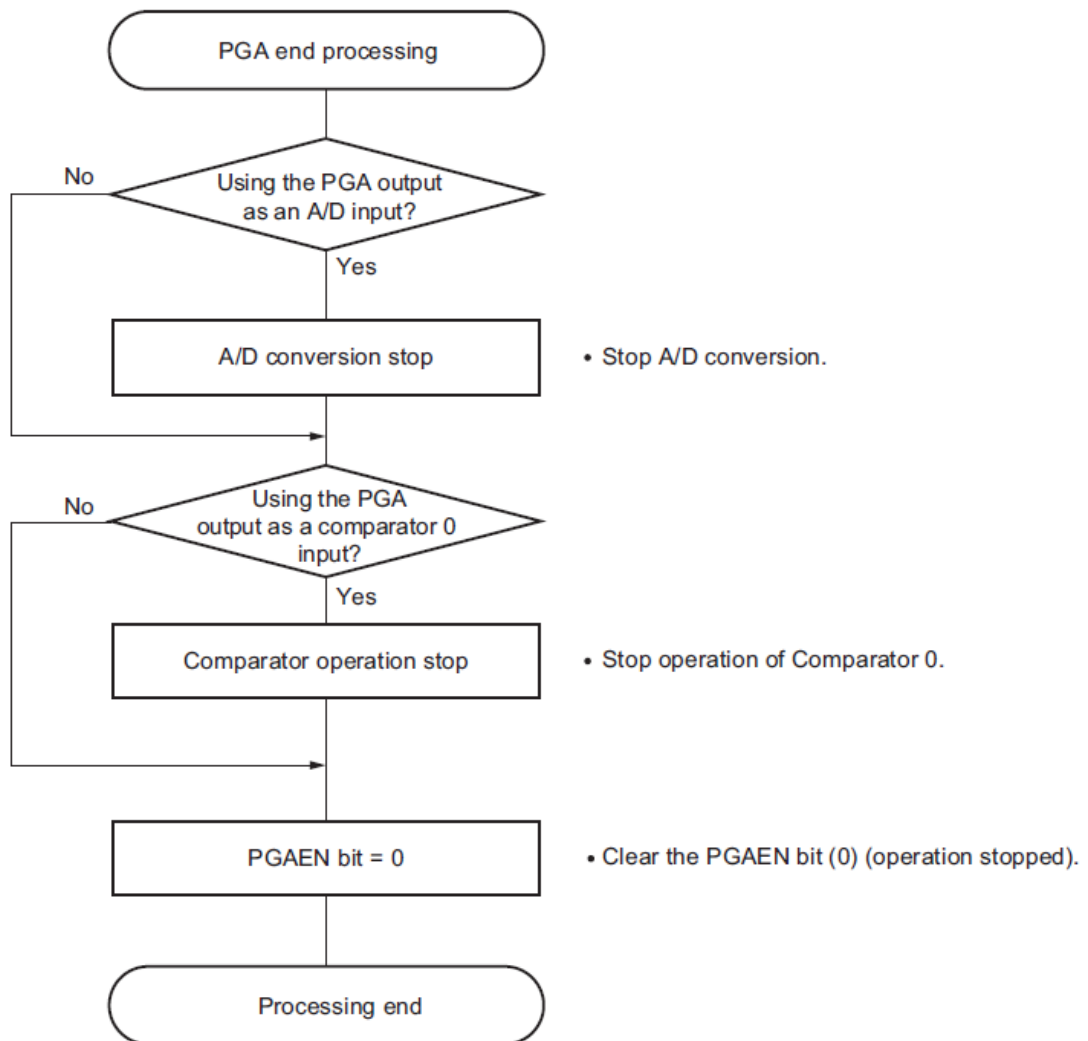
Take PGA0 as an example, set up as follows:



Notice 1. A PGA stabilization time of 10 μ s is required after setting the PGAEN bit to 1. The A/D conversion is then initiated.

13.4.2 Stopping operation steps of programmable gain amplifier

Take PGA0 as an example, set up as follows:



Notice 1. When restarting the PGA and A/D converter or amplifier, a PGA stabilization time of 10us is required after setting the PGAEN bit to 1.

2. Even if the PGA operation is stopped, A/D conversion and comparator operation can be performed using the pass-through pins.

Chapter 14 Universal Serial Communication Unit

The Universal Serial Communication Unit has four serial channels in Unit 0 and two serial channels in Unit 1, each of which can realize 3-wire serial (SSPI), UART, and simplified I²C communication functions.

The functions of each channel supported by this product are assigned as follows.

Unit	Channel	Used as SSPI	Used as UART	Used as simplified I ² C
0	0	SSPI00 (Supports slave selection input function)	UART0	IIC00
	1	SSPI01		IIC01
	2	SSPI10	UART1	IIC10
	3	SSPI11		IIC11
1	0	SSPI20	UART2	IIC20
	1	SSPI21		IIC21

When UART0 is used for channel 0 and channel 1 of Unit 0, SSPI00 and SSPI01 cannot be used, but SSPI10, UART1, and IIC10 for channel 2 and channel 3 can be used.

Notice The following section of this chapter describes the unit and channel structure of the 48-pin product.

14.1 Function of universal serial communication unit

The features of each serial interface supported by this product are shown below.

14.1.1 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)

Data is transmitted and received synchronously with a serial clock (SCLK) output from the master control device.

This is a clock synchronization function communicating using 1 SCLK, 1 SDO and 1 SDI.

For specific set-up examples, refer to “14.5 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)communication operation”.

[Transmit and receive data]

- 7-bit or 8-bit data length
- Phase control of transmitting and receiving data
- MSB/LSB preferred choice

[Clock control]

- Master or slave selection
- Phase control of input/output clocks
- Sets the transfer period generated by the prescaler and the in-channel counter.
- Maximum transfer rate ^{Note}

Master communication: $\text{Max.f CLK}/2$ (SSPI00 only) $\text{Max.f}_{\text{CLK}}/4$

Slave communication: $\text{Max.f}_{\text{MCK}}/6$

[Interrupt function]

- End of transfer interrupt, buffer null interrupt

[Error detection flag]

- Overflow error

Note It must be used within the range that satisfies the SCLK cycle time (t_{KCY}) characteristic. Please refer to the data sheet for details.

14.1.2 UART (UART0~UART2)

This is a function that communicates asynchronously through a total of two lines: serial data transmission (TxD) and serial data reception (RxD). Using these two communication lines, data is sent and received asynchronously (using the internal baud rate) with other communicating parties in a data frame (consisting of a start bit, data, parity bit, and stop bit). Full-duplex UART communication can be achieved by using two channels, send private (even channel) and receive private (odd channel).

For specific setting examples, see “14.7 Operation of UART (UART0~UART2) communication”.

[Transmit and receive data]

- 7-bit, 8-bit or 9-bit data length ^{Note}
- MSB/LSB preferred choice
- Level settings for sending and receiving data, selection of inversions
- Appending parity function for parity bits
- Appending stop bits

[Interrupt function].

- End of transfer interrupt, buffer empty interrupt
- Error interrupts caused by frame errors, parity errors, or overflow errors

[Error detection flag].

- Frame errors, parity errors, overflow errors

14.1.3 Simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21)

This is a function that synchronizes clock communication with multiple devices through a total of 2 lines of serial clock (SCL) and serial data (SDA). Because this simplified I²C is designed for single communication with EEPROM, flash memory, A/D converters, etc., it is only used as a master device.

For start and stop conditions, AC specifications must be adhered to and processed by software while operating the control registers. For specific setup examples, refer to “14.9 Simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21).

[Transmit and receive data]

- Master sending, master receiving (limited to single master control functions).
- ACK output function ^{Note}, ACK detection function
- 8 bits of data length (when sending the address, specify the address with a high 7 bits, and use the lowest bit for R/W control).

- Manual generation of start and stop conditions

[Interrupt function]

- End of transfer interruption

[Error detection flag]

- ACK error, overflow error.

※[Features not supported by Simplified I²C]

- Slave send, slave receive
- Quorum failure detection function
- Wait for detection function

Note : When receiving the last data, if you write “0” to the SOEmn bit (serial output enable register m (SOEm)) to stop the output of the serial communication data, the ACK is not output. For details, please refer to “14.9.3(2) Processing Flow”.

Note When using the full-featured I²C-bus, refer to Chapter 16, Serial Interface IICA.

14.2 Structure of universal serial communication unit

The universal serial communication unit consists of the following hardware.

Table 14-1 Structure of universal serial communication unit

Project	Structure
Shift register	8-bit or 9-bit ^{Note 1}
Buffer register	The serial data register mn (SDRmn) is 8 bits low or 9 bits ^{Note 1 and 2}
Serial clock input/output	SCLK00, SCLK01, SCLK10, SCLK11, SCLK20, SCLK21 pins (for 3-wire serial I/O). SCL00, SCL01, SCL10, SCL11, SCL20, SCL21 pins (for simplified I ² C)
Serial data input	SDI00, SDI01, SDI10, SDI11, SDI20, SDI21 pins (for 3-wire serial I/O), RxD0, RxD1, RxD2 pins (for UART).
Serial data output	SDO00, SDO01, SDO10, SDO11, SDO20, SDO21 pins (for 3-wire serial I/O), TxD0, TxD1, TxD2 pins (for UART).
Serial data input/output	SDA00, SDA01, SDA10, SDA11, SDA20, SDA21 pins (for Simplified I ² C)
Slave Select Input	SS00 pin (for slave select input function).
Control registers	< Register of Unit Setting Section > <ul style="list-style-type: none"> • Peripheral Enable register 0 (PER0). • Serial clock selection register m (SPSm). • Serial channel enable status register m (SEm). • Serial channel start register m (SSm). • Serial channel stop register m (STm). • Serial output allows register m (SOEm). • Serial output register m (SOM). • Serial output level register m (SOLm). • Input switch control register (ISC). • Noise filter enable register 0 (NFEN0). < register for each channel > <ul style="list-style-type: none"> • Serial data register mn (SDRmn). • Serial mode register mn (SMRmn). • Serial communication operation setting register mn (SCRmn). • Serial status register mn (SSRmn). • Serial flag clear trigger register mn (SIRmn). <ul style="list-style-type: none"> • Port multiplexing function configuration register (PxxCFG). • Port output mode register (POMxx). • Port mode register (PMxx). • Port register (Pxx).

Note 1 The number of bits used as shift registers and buffer registers varies by unit and channel.

- mn = 00, 01: low 9-bit
- Others: low 8-bit

2. Depending on the communication mode, the lower 8 bits of the serial data register mn (SDRmn) can be read and written with the following SFR name.

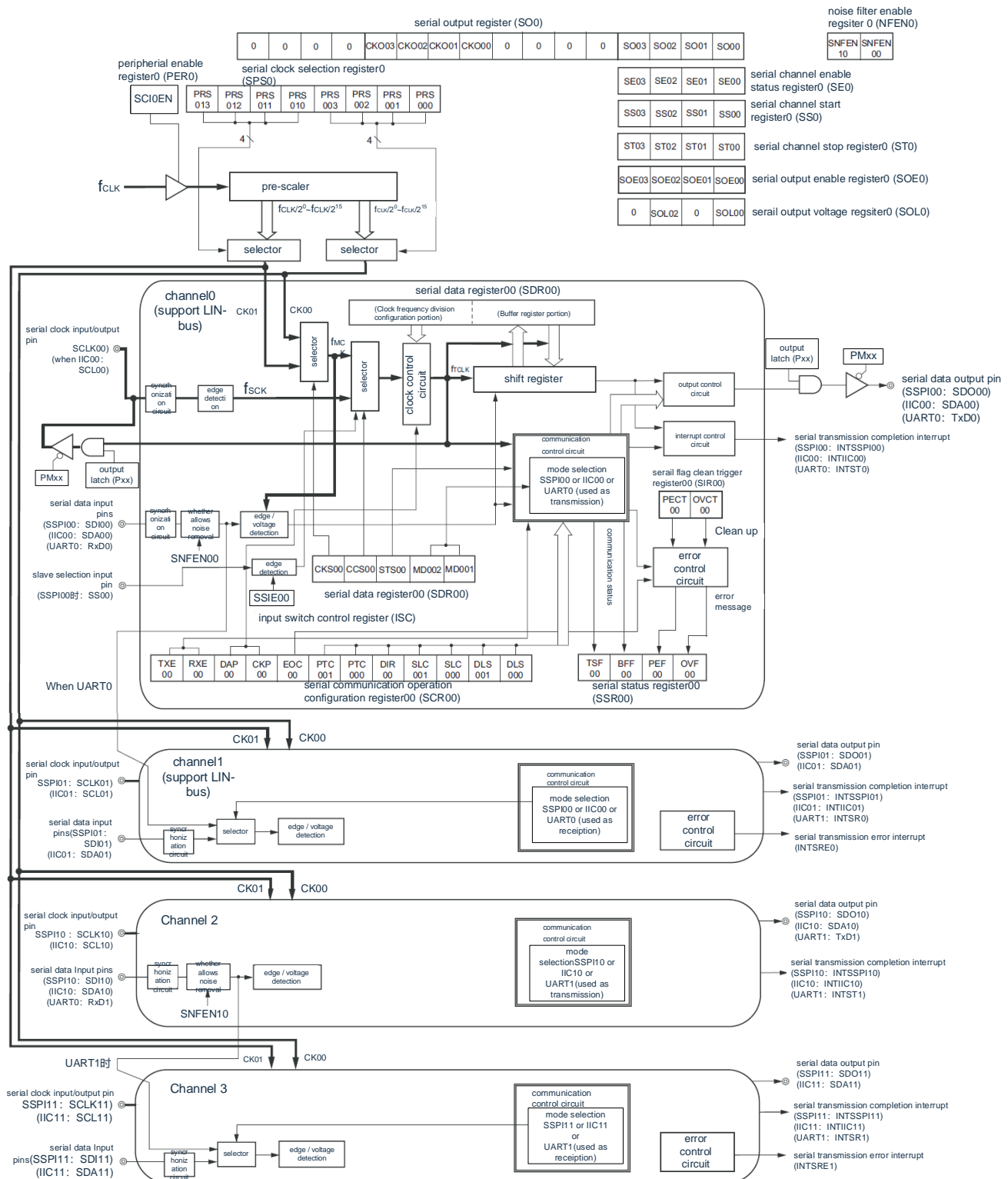
- SSPIp communication... SIOp (SSPIp Data Register).
- UARTq receives... RXDq (UARTq Receive Data Register).
- UARTq transmits... TXDq (UARTq Transmit Data Register).
- IICr Communications... SIOr (IICr Data Register).

Remark m: Unit number (m=0, 1) n: Channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)

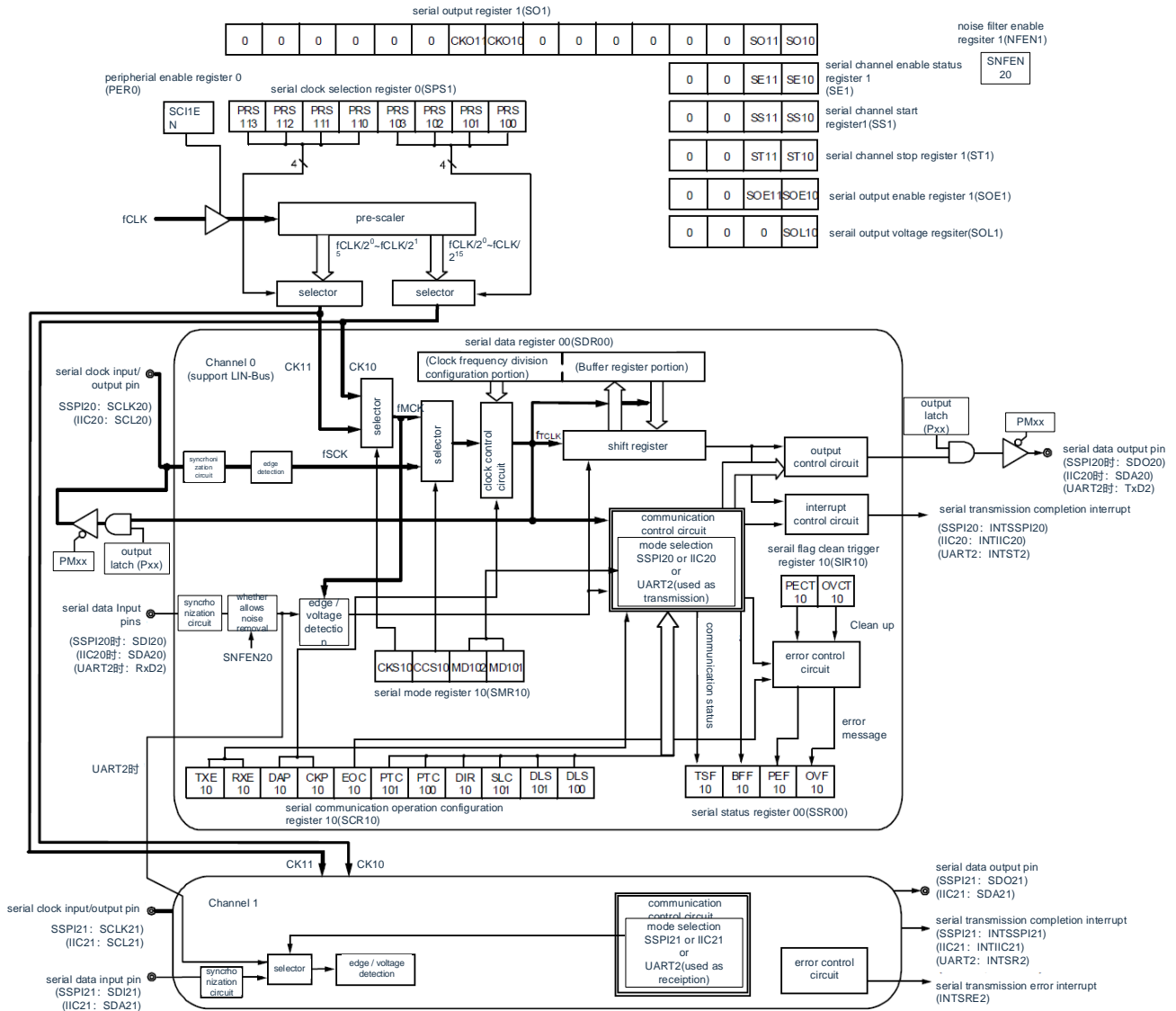
q: UART number (q=0~2) r: IIC number (r=00, 01, 10, 11, 20, 21)

A block diagram of universal serial communication unit 0 is shown in Figure 14-1.

Figure 14-1: Diagram of universal serial communication unit 0



A block diagram of a universal serial communication unit 1 is shown in Figure 14-2.
Figure 14-2: Diagram of universal serial communication unit 1



14.2.1 Shift register

This is a 9-bit register that converts parallel and serial to and from each other.

For UART communication at 9 bits of data length, use 9 bits (bit0 to 8) ^{Note 1}. Converts the input data of the serial input pin into parallel data when receiving data; When data is sent, the value to this register will be transferred as serial data from the serial output pin output ^{Note 1}. Shift registers cannot be manipulated directly through the program.

To read and write data from shift registers, use the low 8 or 9 bits of the serial data register mn (SDRmn).

	8	7	6	5	4	3	2	1	0
Shift register									

14.2.2 Low 8 bits or low 9 bits of serial data register mn (SDRmn)

The SDRmn register is the transmit and receive data registers (16-bit) of channel n.

Bit8~0 (low 9 bits)^{Note} or bit7~0 (low 8 bits) is used as the transmit and receive buffer registers bit15~9 is used as a crossover setting register for the running clock (f_{MCK}).

When receiving data, save the parallel data converted by the shift register to the lower 8 bits or the lower 9 bits; When transmitting data, the transmitted data to the shift register is set to a lower 8 bits or a low 9 bits.

Regardless of the output order of the data, set registers mn (SCRmn) bit0 and bit1 (DLSmn0, DLSmn1) are run according to serial communication) settings saved to the lowest 8 bits or lower 9 bits of data as follows:

- 7-bit data length (saved in bit0~6 of the SDRmn register).
- 8 bits of data length (saved in bit0~7 of the SDRmn register).
- 9 bits of data length (saved in bit0~8 of the SDRmn register)^{Note 1}

SDRmn registers can be read and written in 16-bit increments.

Depending on the communication mode, the low 8 bits of the SDRmn register or the low 9 bits of the SDRmn register can be read and written in 8-bit units with the following SFR name^{Note 2}.

- SSPIp communication... SDIOp (SSPIp Data Register).
- UARTq receives... RXDq (UARTq Receive Data Register).
- UARTq sends... TXDq (UARTq Transmit Data Register).
- IICr Communications... SDIO r (IICr Data Register).

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

Note 1 Only UART0 supports 9-bit data length.

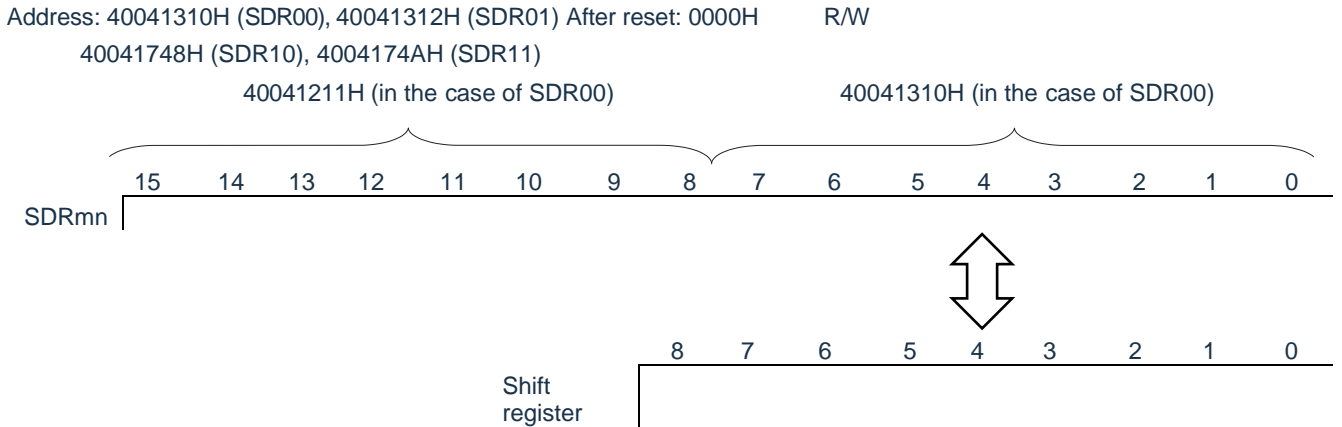
2. When the operation is stopped ($SEmn=0$), it is forbidden to override SDRmn [7:0] via 8-bit memory operation instructions (otherwise, SDRmn [15:9] is cleared).

Remark 1 After the reception ends, bits from bit0 to 8 that exceed the length of the data are "0".

2.m: Unit number (m=0, 1) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)

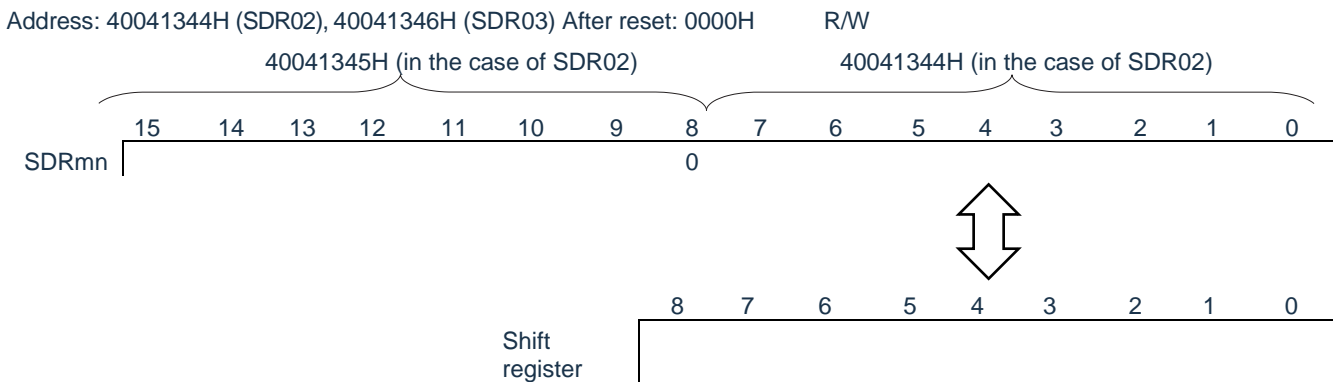
q: UART number (q=0~2) r: IIC number (r=00, 01, 10, 11, 20, 21)

Figure 14-3 Format of serial data register mn (SDRmn) (mn=00, 01, 10, 11)



Remark For the functions of the higher 7 bits of the SDRmn register, refer to "14.3 Registers for Controlling Universal Serial Communication Unit".

Figure 14-4 Format of serial data register mn (SDRmn) (mn=02, 03, 10, 11, 12, 13)



Notice: Bit8 must be set to "0".

For the functions of the higher 7 bits of the SDRmn register, refer to "14.3 Registers for Controlling Universal Serial Communication Unit".

14.3 Registers for controlling universal serial communication unit

The registers that control the universal serial communication unit are as follows:

- Peripheral enable register 0 (PER0).
- Serial clock selection register m (SPSm).
- Serial mode register mn (SMRmn).
- Serial communication operation setting register mn (SCRmn).
- Serial data register mn (SDRmn).
- Serial flag clears trigger register mn (SDIRmn).
- Serial status register mn (SSRmn).
- Serial channel start register m (SSm).
- Serial channel stop register m (STm).
- Serial channel allows status register m (SEm).
- Serial output allows register m (SOEm).
- Serial output level register m (SOLm).
- Serial output register m (SOM).
- Input Switch Control Register (ISC).
- Noise filter allows register 0 (NFEN0).
- Port multiplexing function configuration register (PxxCFG).
- Port output mode register (POMx).
- Port mode register (PMx).
- Port register (Px).

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

14.3.1 Peripheral enable register 0 (PER0)

The PER0 register is a register that sets the clock to be allowed or disallowed to be supplied to each peripheral hardware. Reduce power consumption and noise by stopping clocking unused hardware.

To use universal serial communication unit 0, bit2 (SCI0EN) must be set to “1”.

To use universal serial communication unit 1, bit3 (SCI1EN) must be set to “1”.

The PER0 register is set via an 8-bit memory operation command.

After the reset signal is generated, the value of the PER0 register changes to “00H”.

Figure 14-5 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	IRDAEN	ADCEN	IICA0EN	SCI1EN	SCI0EN	TM41EN	TM40EN

SCI _m EN	Provides control of the input clock of the universal serial communication unit m
0	Stops input clock supply. <ul style="list-style-type: none"> • SFR used by the universal serial communication unit m cannot be written. • Universal serial communication unit m is in the reset status.
1	Enables input clock supply. <ul style="list-style-type: none"> • SFR used by the universal serial communication unit m can be read and written.

Notice1. To set the universal serial communication unit m, the following registers must first be set when the SCI_mEN bit is “1”. When the SCI_mEN bit is “0”, the write operation of the control register of the universal serial communication unit m is ignored, and the read value is the initial value (except for input switching control register (ISC), noise filter enable register 0 (NFEN0), port multiplexing function configuration register (PxxCFG), and port output mode register (POMx), port mode registers (PMx), port mode control registers (PMCx), and port registers (Px).

- Serial clock selection register m (SPSm).
- Serial mode register mn (SMRmn).
- Serial communication operation setting register mn (SCRmn).
- Serial data register mn (SDRmn).
- Serial flag clear trigger register mn (SIRmn).
- Serial status register mn (SSRmn).
- Serial channel start register m (SSm).
- Serial channel stop register m (STm).
- Serial channel enable status register m (SEm).
- Serial output enable register m (SOEm).
- Serial output level register m (SOLm).
- Serial output register m (SOM).

14.3.2 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. CKm1 is selected by bit7~4 of the SPSm register, and by bit3~0 CKm0.

It is forbidden to overwrite the SPSm register during operation (SEmn=1).

The SPSm register is set via 16-bit memory operation instructions.

I can set the low 8 bits of the SPSm register with SPSmL and via 8-bit memory operation instructions.

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

Figure 14-6 Format of serial clock selection register m (SPSm)

Address: 40041126H (SPS0), 40041566H (SPS1)	After reset: 0000H																R/W
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
SPSm	0	0	0	0	0	0	0	0	PRSm13	PRSm12	PRSm11	PRSm10	PRSm03	PRSm02	PRSm01	PRSm00	

PRSmk3	PRSmk2	PRSmk1	PRSmk0	Selection of the operating clock (CKmk) ^{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 To change the clock selected as f_{CLK} (change the value of the system clock control register (CKC)) during the operation of the Universal Serial Communication Unit (SCI), the operation of the SCI must be stopped (serial channel stop register m (STm)=000FF) and then make the change.

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

14.3.3 Serial mode register mn (SMRmn)

The SMRmn register is a register that sets the channel n operating mode, selects the operating clock (f_{MCK}), specifies whether the serial clock (f_{SCLK}) input can be used, sets the start trigger, and operates the mode (SSPI, UART, Simplified I²C) settings and interrupt source selection. In addition, the inverting level of the received data is set only in UART mode.

Overwriting the SMRmn register during operation ($SEmn=1$) is prohibited, but the MDmn0 bit can be overridden during operation.

The SMRmn register is set via a 16-bit memory operation command.

After the reset signal is generated, the value of the SMRmn register changes to “0020H”.

Figure 14-7 Format of serial mode register mn (SMRmn) (1/2)

Address: 40041110H (SMR00)~40041116H (SMR03) After reset: 0020H R/W
 40041550H (SMR10)~40041552H (SMR11)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn	CCSmn	0	0	0	0	0	STSm _n ^{注1}	0	SISmn ₀ ^{注1}	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. The transmit clock (f_{TCLK}) is generated by setting the CCSmn bit and the SDRmn register high 7 bits.	

CCSmn	Selection of channel n transmit clock (f_{TCLK})
0	The CKSmn bit specifies the crossover clock of the running clock f_{MCK}
1	The 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 “0”, the runtime clock (f_{MCK}) is divided by the high 7 bits of the SDRmn register.	

STSmn ^{Note1}	Selection of start triggering sources
0	Only software triggers are valid (selected in SSPI, UART Send, Simplified I ² C).
1	The effective edge of the RxDq pin (selected when received by the UART).
Transmission begins when the above conditions are met after setting the SSm register to “1”.	

Note 1 SMR01, SMR03, SMR11 registers only.

Note You must set bit13~9, 7, 4, 3 (SMR00, SMR02, SMR10 registers are bit13~6, 4, 3) to “0”, and set bit5 to “1”.

Remark m: Unit number (m=0, 1) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)
 q: UART number (q=0~2) r: IIC number (r=00, 01, 10, 11, 20, 21)

Figure 14-7 Format of serial mode register mn (SMRmn) (2/2)

Address: 40041110H (SMR00)~40041116H (SMR03) After reset: 0020H R/W
 40041550H (SMR10)~40041552H (SMR11)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKS mn	CCS mn	0	0	0	0	0	STSm nNote 1	0	SISmn 0Note 1	1	0	0	MD mn2	MD mn1	MD mn0

SISmn0 ^{Note1}	Level inversion control of received data for channel n in UART mode
0	Detect the falling edge as the starting bit. The input communication data is not reversed.
1	Detect the rising edge as the starting point. The input communication data is reversed.

MDmn2	MDmn1	Setting of the channel n operating mode
0	0	SSPI mode
0	1	UART mode
1	0	Simplified I ² C mode
1	1	Disable settings.

MDmn0	Channel n interrupt source selection
0	The end of the transfer is interrupted
1	Buffer empty interrupt (Occurs when data is transferred from the SDRmn register to the shift register).
In continuous transmission, if the MDmn0 bit is "1" and the data for SDRmn is empty, the next sent data is made.	

Note 1 SMR01, SMR03, SMR11, registers only.

Notice You must set bit13 to 9, 7, 4, 3 (SMR00, SMR02, SMR10 registers are bit13~6, 4, 3) to "0", and set bit5 to 1".

Remark m: Unit number (m=0, 1) n: Channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)
 q: UART number (q=0~2) r: IIC number (r=00, 01, 10, 11, 20, 21)

14.3.4 Serial communication operation setting register mn (SCRmn)

The SCRmn register is the communication operation setting register for channel n, which sets the data transmit and receive modes, data and clock phases, whether to mask error signals, parity 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 via a 16-bit memory operation command.

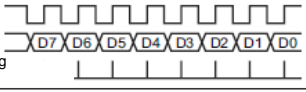
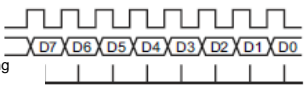
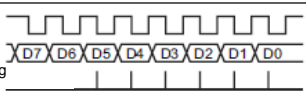
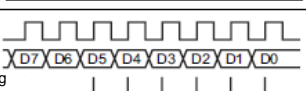
After the reset signal is generated, the value of the SCRmn register changes to "0087H".

Figure 14-8 Format of serial communication operation setting register mn (SCRmn) (1/2)

Address: 40041118H (SCR00)~4004111EH (SCR03) After reset: 0087H R/W
 40041558H (SCR10)~4004155AH (SCR13)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn mn	RXEmn mn	DAPmn	CKPmn	0	EOCmn	PTCmn1	PTCmn0	DIRmn	0	SLCmn1 ^{Note1}	SLCmn0	0	1	DLSmn1 ^{Note2}	DLSmn0

TXEmn	RXEmn	Setting of the channel n operating mode
0	0	Prohibited communication.
0	1	Receive only.
1	0	Transmit only.
1	1	Enables transmit and receive.

DAPmn	CKPmn	data and clock phase selection in SSPI mode	Type
0	0	SCLKp 	1
0	1	SCLKp 	2
1	0	SCLKp 	3
1	1	SCLKp 	4

in UART mode and simple I2C mode, must set DAPmn bit and CKPmn bit both to 0.

EOCmn	Mask control of error interrupt signal (INTSREx (x=0 to 3))
0	Disable the generation of error interrupts INTSREx (generate INTSRx).
1	Enable error interrupt INTSREx (no INTSRx is generated when an error occurs).

The EOCmn bit must be set to "0" in SSPI mode and Simplified I²C mode or when sending from UART ^{Note 3}.

Note 1 Limited to SCR00, SCR02, SCR10 registers only.

2. Limited to SCR00 register and SCR01 register, the others are fixed as "1".

3. When the EOCmn bit is "0" and SSPImn is not used, it is possible to generate an error interrupt INTSREn.

Notice Bit 3, 6, and 11 must be set to "0" (also bit 5 of SCR01, SCR03, and SCR11 registers must be set to "0"), and bit 2 must be set to "1".

Remark m: Unit number (m=0, 1) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)

Figure 14-8 Format of serial communication operation setting register mn (SCRmn) (2/2)

 Address: 40041118H (SCR00)~4004111EH (SCR03) After reset: 0087H RW
 40041558H (SCR10)~4004155AH (SCR13)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXE mn	RXE mn	DAP mn	CKP mn	0	EOC mn	PTC mn1	PTC mn0	DIR mn	0	SLCm n1 ^{Note1}	SLC mn0	0	1	DLSm n1 ^{Note2}	DLS mn0

PTCmn1	PTCmn0	Setting of parity bits in UART mode	
		Transmission	Reception
0	0	Parity bits are not output.	There is no parity at the time of reception.
0	1	Output parity ^{Note 3} .	Parity is not judged.
1	0	Output parity.	Judgment even.
1	1	Output odd check.	Judgment odd check.

In SSPI mode and Simplified I²C mode, both the PTCmn1 bit and the PTCmn0 bit must be set to "0".

DIRmn	Selection of data transfer order in SSPI and UART modes
0	Performs MSB-priority input/output.
1	Perform LSB-priority input/output.

In Simplified I²C mode, the DIRmn bit must be "0".

SLCmn1 ^{Note1}	SLCmn0	Setting of the stop bit in UART mode
0	0	No stop bits
0	1	Stop bit length = 1 bit
1	0	Stop bit length = 2 bits (limited to mn=00, 02, 10).
1	1	Disable settings.

If an end-of-transfer interrupt is selected, an interrupt occurs after all stop bits have been transferred. When UART is received or in Simplified I²C mode, it must be set to 1 stop bit (SLCmn1, SLCmn0=0, 1). In SSPI mode, it must be set to no stop (SLCmn1, SLCmn0=0, 0). When UART is sent, it must be set to 1 bit (SLCmn1, SLCmn0=0, 1) or 2 bits (SLCmn1, SLCmn0=1, 0).

DLSmn1 ^{Note2}	DLSmn0	Setting of data length in SSPI and UART modes
0	1	9 bits of data length (saved in bit0~8 of the SDRmn register) (selectable only in UART mode).
1	0	7 bits of data length (saved in bit0 to 6 of the SDRmn register).
1	1	8 bits of data length (bit0 to 7 in the SDRmn register).
other		Disable settings.

In Simplified I²C mode, both the DLSmn1 bit and the DLSmn0 bit must be set to "1".

Note 1 Limited to SCR00, SCR02, SCR10 registers only.

2. Limited to SCR00 register and SCR01 register, the others are fixed as "1".

3. Nothing to do with the content of the data, always appended "0".

Notice Bits 3, 6, and 11 must be set to "0" (also bit 5 of the SCR01, SCR03, and SCR11 registers must be set to "0"), and bit 2 must be set to "1".

Remark m: Unit number (m=0, 1) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)

14.3.5 Serial data register mn (SDRmn)

The SDRmn register is the data register (16 bits) that channel n transmits and receives.

The bits 8 to 0 (low 9 bits) of SDR00 and SDR01 or bits 7 to 0 (low 8 bits) of SDR02, SDR03, SDR10 and SDR11 are used as transmit and receive buffer registers, and bits 15 to 9 (high 7 bits) are used as operating clock (f_{MCK}) divider setting registers.

If the CCSmn bit of the serial mode register mn (SMRmn) is set to "0", the divider clock of the operation clock set by bits 15 to 9 (high 7 bits) of the SDRmn register is used as the transmit clock.

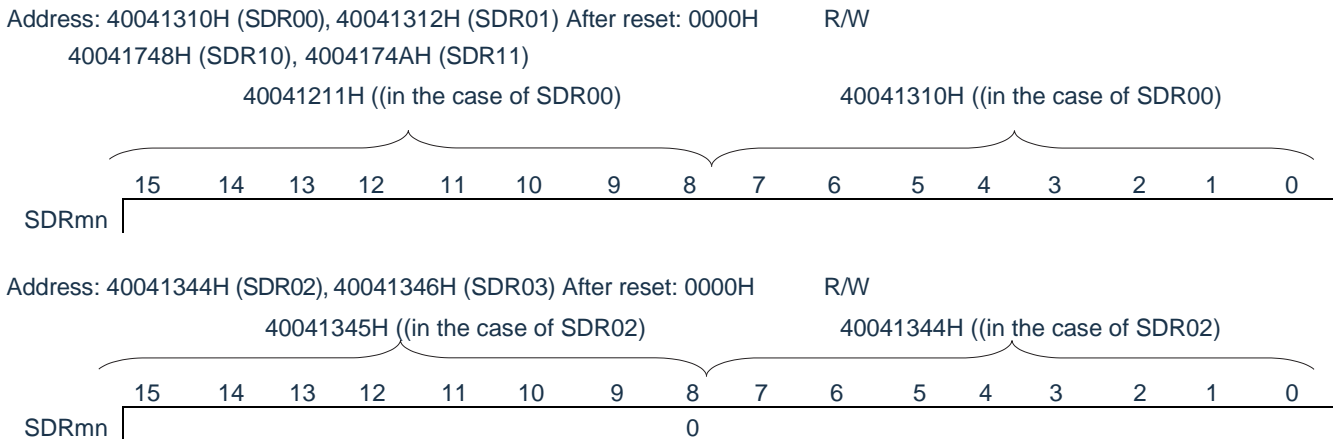
If the CCSmn bit is set to "1", bit15~9 (high 7 bits) of SDRmn must be set to "0000000B". The input clock f_{SCLK} (slave transfer in SSPI mode) of the SCLKp pin is the transmit clock.

The low 8 or 9 bits of the SDRmn register are used as transmit and receive buffer registers. When receiving data, shift registers are converted in parallel data is saved to the lowest 8 bits or the lower 9 bits; When transmitting data, the transmitted data to the shift register is set to a lower 8 bits or a low 9 bits.

SDRmn registers can be read and written in 16-bit increments. However, high 7 bits can only be read and written when the operation is stopped (SEmn=0). In operation (SEmn=1) only the low 8 bits or 9 bits of the SDRmn register can be written, and the high 7 bits of the SDRmn register are always read as "0".

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

Figure 14-9 Format of serial data register mn (SDRmn)



SDRmn[15:9]							Setting of transmission clock for operation clock division
0	0	0	0	0	0	0	$f_{MCK}/2$
0	0	0	0	0	0	1	$f_{MCK}/4$
0	0	0	0	0	1	0	$f_{MCK}/6$
0	0	0	0	0	1	1	$f_{MCK}/8$
.
.
.
1	1	1	1	1	1	0	$f_{MCK}/254$
1	1	1	1	1	1	1	$f_{MCK}/256$

- Note 1. Bit8 of the SDR02, SDR03, SDR10, and SDR11 registers must be set to "0".
- 2. When using UART, it is forbidden to set SDRmn [15:9] to "0000000B" and "0000001B".
- 3. When using Simplified I2C, it is forbidden to set SDRmn[15:9] to "0000000B", and the SDRmn [15:9] setting value must be greater than or equal to "0000001B".

4. When the operation is stopped ($SE_{mn}=0$), it is forbidden to override SDR_{mn} [7:0] via 8-bit memory operation instructions (otherwise, SDR_{mn} [15:9] is all cleared "0").

Note 1. For the function of the SDR_{mn} register with the low 8 or 9 bits, refer to "14.2 Structure of universal serial communication unit".

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

14.3.6 Serial flag clear trigger register mn (SIRmn)

This is the trigger register used to clear each error flag of channel n.

If you set the various (FECTmn, PECTmn, OVCTmn) to “1”, the corresponding bits (FEFmn, PEFmn, OVFmn) clear “0”. Because the SDIRmn register is a trigger register, if the corresponding bit of the SSRmn register is cleared, the SDIRmn register is also cleared immediately.

The SIRmn register is set via 16-bit memory operation instructions.

The low 8 bits of the SIRmn register can be set with SIRmnL and via 8-bit memory operation instructions.

After the reset signal is generated, the value of the SIRmn register changes to “0000H”.

Figure 14-10 Format of serial flag clear trigger register mn (SIRmn)

Address: 40041108H (SIR00)~4004110EH (SIR03) After reset: 0000H R/W
 40041548H (SIR10)~4004154AH (SIR11)

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 ^{Note1}	PEC Tmn	OVC Tmn

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 Restricted to SIR01, SIR03, SIR11 registers only.

Notice Bit 15 to 3 (bit 15 to 2 for SIR00, SIR02 and SIR10 registers) must be set to "0".

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

2. Read value of the SIRmn register is always “0000H”.

14.3.7 Serial status register mn (SSRmn)

The SSRmn register indicates the communication status of channel n and the occurrence of errors. The errors represented are frame errors, parity errors, and overflow errors. The SSRmn register is read via 16-bit memory operation instructions.

The lower 8 bits of the SSRmn register can be read with SSRmnL and via 8-bit memory operation instructions. After the reset signal is generated, the value of the SSRmn register changes to “0000H”.

Figure 14-11 Format of serial status register mn (SSRmn) (1/2)

Address: 40041100H (SSR00)~40041106H (SSR03)
 40041540H (SSR10)~40041542H (SSR11)

After reset: 0000H R

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSRmn	0	0	0	0	0	0	0	0	0	TSF mn	BFF mn	0	0	FEF mn ^{Note1}	PEF mn	OVF mn

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 SSm bit of SSm register is set to "1" (communication standby state)
- When 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 transmitted data from the SDRmn register to the shift register is transferred during the transmit process
- When the received data is finished reading from the SDRmn register during the receive process
- When the STmn bit of STm 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 data to the SDRmn register in the state where the TXEmn bit of the SCRmn register is "1" (transmit, transmit, and receive modes in each communication mode).
- When the received data is saved to the SDRmn register in the state where the RXEmn bit of the SCRmn register is "1" (receive mode, transmit and receive modes in each communication mode).
- When a receive error occurs

Note 1 For SSR01, SSR03, SSR11 registers only.

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

Figure 14-11 Format of serial status register mn (SSRmn) (2/2)

 Address: 40041100H (SSR00)~40041106H (SSR03)
 40041540H (SSR10)~40041542H (SSR11)

After reset: 0000H R

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSRmn	0	0	0	0	0	0	0	0	0	TSF mn	BFF mn	0	0	FEF mn ^{Note1}	PEF mn	OVF mn

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 [Accent 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) or the ACK was not detected (when the I2C was sent).
[Clear condition]. • When writing “1” to the PECTmn bit of the SIRmn register [Set condition]. • When the parity and parity bits of the data sent at the end of the UART reception are different (parity errors). • When it is sent by I2C and when the ACK receiving timing slave does not return an ACK signal (no ACK detected).	

OVFmn	Detection flag for channel n overflow error
0	No errors occurred.
1	An error has 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 modes in each communication mode), although the received data is saved in the SDRmn register, but when there is no read receive data and write send data or write down a received data • When data is not ready to be sent during slave send or slave send and receive in SSPI mode.	

Note 1 For SSR01, SSR03, SSR11 registers only.

Notice 1 If you write the SDRmn register when the BFFmn bit is “1”, the saved transmit or receive data is corrupted and an overflow error is detected (OVEmn=1).

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

14.3.8 Serial channel start register m (SSm)

The SSm register is a trigger register that sets the communication/start count enabled for each channel.

If you write "1" to you (SSmn), set the corresponding bit (SEmn) of the serial channel enable status register m (SEmn) to "1" (operation enable status). Because the SSmn bit is the trigger bit, the SSmn bit is cleared immediately if the SEmn bit is "1".

The SSm register is set via a 16-bit memory operation command.

I can set the lower 8 bits of the SSm register with SSmL and via 8-bit memory operation instructions.

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

Figure 14-12 Format of serial channel start register m (SSm)

Address: 40041122H (SS0)	After reset: 0000H															R/W	
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
SS0	0	0	0	0	0	0	0	0	0	0	0	0	SS03	SS02	SS01	SS00	

Address: 40041562H (SS1)	After reset: 0000H															R/W	
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
SS1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SS11	SS10	

SSmn	Trigger at the beginning of channel n operation
0	No triggering.
1	Sets the SEmn bit "1" and shift to communication standby state ^{Note} .

Note If the SSmn bit is set to "1" during communication, communication is stopped and enters standby. At this point, the values of the control register and shift register, the SCLKmn pin and the SDOmn pin, the FEFmn flag, the PEFmn flag, and the OVFmn flag remain in state.

Notice 1 Bit 15 to 4 of SS0 register and bit 15 to 2 of SS1 register must be set to "0".

2. For UART receive, at least 4 f_{MCK} clocks must be set to "1" after setting RXEmn in SCRmn register to "1", and then setting SSmn to "1".

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

2. The read value of the SSm register is always "0000H".

14.3.9 Serial channel stop register m (STm)

The STm register is a trigger register that sets the communication/stop count allowed 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 via a 16-bit memory operation command.

The low 8 bits of the STm register can be set with STmL and via 8-bit memory operation instructions.

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

Figure 14-13 Format of serial channel stop register m (STm)

Address: 40041124H (ST0)	After reset: 0000H								R/W							
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ST0	0	0	0	0	0	0	0	0	0	0	0	0	ST03	ST02	ST01	ST00

Address: 40041564H (ST1)	After reset: 0000H								R/W							
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ST1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ST11	ST10

STmn	Stop trigger for channel n operation
0	No triggering.
1	Clears the SEmn bit "0" to stop the communication from running.

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.

Note Bit15~4 of the ST0 register and bit15~2 of the ST1 register must be set to "0".

- Note 1. m: unit number (m=0, 1) n: channel number (n=0~3).
 2. The read value of the STm register is always "0000H".

14.3.10 Serial channel enable status register m (SEm)

The SEm register is used to confirm the allow or stop status of serial transmission and reception 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 that is allowed to run, the value of the CKOmn bit (the 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 operation is output from the serial clock pin.

For a stopped channel n, the value of the CKOmn bit of the SOM register can be set by software and output from the serial clock pin. Thus, an arbitrary waveform such as a start condition or a stop condition can be generated by software.

The SEm register is read via 16-bit memory operation instructions.

The lower 8 bits of the SEm register can be read with SEmL and via 8-bit memory operation instructions. After the reset signal is generated, the value of the SEm register changes to "0000H".

Figure 14-14 Format of serial channel enable status register m (SEm)

Address: 40041120H (SE0)	After reset: 0000H								R							
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SE0	0	0	0	0	0	0	0	0	0	0	0	0	SE03	SE02	SE01	SE00

Address: 40041560H (SE1)	After reset: 0000H								R							
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SE1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SE11	SE10

SEmn	Indication of the enable or stop state of channel n operation
0	Run stop status
1	Run enable status

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

14.3.11 Serial output enable register m(SOEm)

SOEm register settings allow or stop the output of serial communication for each channel.

For channel n that allows serial output, the value of the SOMn bit of the serial output register m (SOM) described later cannot be rewritten by software, and 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, an arbitrary waveform such as a start condition or a stop condition can be generated by software.

The SOEm register is set via a 16-bit memory operation command.

The lower 8 bits of the SOEm register can be set with SOEmL and via 8-bit memory operation instructions.

After the reset signal is generated, the value of the SOEm register changes to “0000H”.

Figure 14-15 Format of serial output enable register m (SOEm)

Address: 4004112AH	After reset: 0000H								R/W							
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOE0	0	0	0	0	0	0	0	0	0	0	0	0	SOE03	SOE02	SOE01	SOE00

Address: 4004156AH	After reset: 0000H								R/W							
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOE1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOE11	SOE10

SOE mn	Enable or stop of channel n serial output
0	Stops the output of serial communication.
1	Enables the output of serial communication.

Note Bits 15~4 of the SOE0 register and bits 15~2 of the SOE1 register must be set to “0” .

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

14.3.12 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 only be rewritten by software when serial output is disabled (SOEmn=0).

When serial output (SOEmn=1) is allowed, the value of the SOMn bit of this register can only be changed by serial communication by software overwriting.

The CKOMn bit of this register can only be rewritten by software only when the channel is stopped (SEmn=0).

When allowing the channel to run (SEmn=1), the value of the CKOMn bit of this register can only be changed by serial communication by overriding the software.

To use the serial interface pins 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 via a 16-bit memory operation command.

After the reset signal is generated, the value of the SOM register changes to "0F0FH".

Figure 14-16 Format of serial output register m (SOM)

Address: 40041128H	After reset: 0F0FH															R/W
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SO0	0	0	0	0	CKO	CKO	CKO	CKO	0	0	0	0	SO	SO	SO	SO
					03	02	01	00					03	02	01	00

Address: 40041568H	After reset: 0303H															R/W
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SO1	0	0	0	0	0	0	CKO	CKO	0	0	0	0	0	0	SO	SO
							11	10							11	10

CKO mn	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".

SO mn	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 15~12 and bits 7~4 of the SO0 register must be set to "0".
Bits 15~10 and bits 7~2 of the SO1 register must be set to "0".

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

14.3.13 Serial output level register m (SOLm)

The SOLm register is a register that sets the inverting of the data output level of each channel.

This register can be set only in UART mode. In SSPI mode and Simplified I²C mode, the corresponding bit must be set to "0". Only when serial output is allowed (SOEmn=1), the n-inverting setting of each channel of this register is reflected to the pin output. When serial output is disabled (SOEmn=0), the value of the SOmn bit is output directly. It is forbidden to override the SOLm register during operation (SEmn=1).

The SOLm register is set via a 16-bit memory operation command.

The low 8 bits of the SDOLm register can be set with SOLmL and via 8-bit memory operation instructions.

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

Figure 14-17 Format of serial output level register m (SOLm)

Address: 40041134H	After reset: 0000H														R/W	
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOL0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOL 02	0	SOL 00

Address: 40041574H	After reset: 0000H														R/W	
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOL1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOL 10

SOL mn	Selection of channel n transmit data level inversion in UART mode
0	The communication data is output directly.
1	The communication data is output inversely.

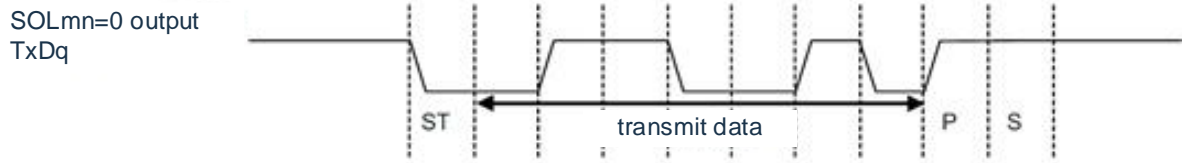
Notice Bits 15 to 3 and bit1 of the SOL0 register and the bits 15 to 1 of the SOL1 register must be set to "0".

Remark m: unit number (m=0, 1) n: channel number (n=0, 2).

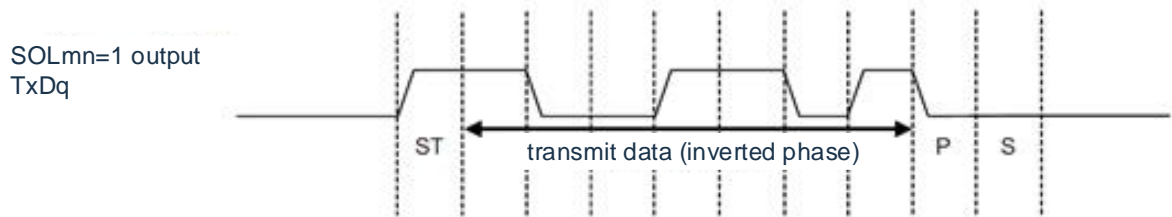
When UART is transmitted, an example of the level inversion of the transmitted data is shown in Figure 14-18.

Figure 14-18 Example of level inversion of transmitted data

(a) Non-inverting output (SOLmn=0)



(b) Inverting output (SOLmn=1)



Remark m: unit number (m=0, 1) n: channel number (n=0, 2).

14.3.14 Input switching control register (ISC)

When implementing LIN-bus communication via UART0, the ISC1 and ISC0 bits of the ISC register are used for the coordination of external interrupts and timer array units. If bit0 is set to “1”, the input signal from the serial data input (RxD0) pin is selected as the input for the external interrupt (INTP0) and therefore passes The INTP0 interrupt detects the wake-up signal.

If bit1 is set to “1”, the input signal from the serial data input (RxD0) pin is selected as the input to the timer, so that the wake-up signal can be detected by the timer and the low-level width of the interval segment and the pulse width of the synchronization segment can be measured.

The SS1E00 bit controls the SS00 pin input of channel 0 in slave mode of SSPI00 communication. During the period when the SS00 pin is input high, no transmission and reception occurs even if the serial clock is input; During the low input level to the SS00 pin, if a serial clock is entered, it is transmitted and received according to the settings of each mode.

The ISC register is set via an 8-bit memory operation command.

After the reset signal is generated, the value of the ISC register changes to “00H”.

Figure 14-19 Format 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 of channel 0 in slave mode for SSPI00 communication
0	The SS00 pin input is invalid.
1	The SS00 pin input is valid.

ISC1	Input switching of channel 3 of timer Timer4
0	Use the input signal from the TI03 pin as the input to the timer (usually running).
1	Use the input signal from the RxD0 pin as the input to the timer (detects the wake-up signal and measures the low-level width of the interval and the pulse width of the sync field).

ISC0	Input switching for external interrupt (INTP0)
0	Use the input signal from the INTP0 pin as the input for an external interrupt (usually in operation).
1	Use the input signal from the RxD0 pin as the input for the external interrupt (detect wake-up signal).

Notice Bits 6~0 must be set to “0”.

14.3.15 Noise filter enable register 0 (NFEN0)

The NFEN0 register sets whether the noise filter is used for the input signal of the serial data input pins of each channel.

For pins used for SSPI or simplified I²C communication, the corresponding bit must be "0" to invalidate the noise filter. For the pins used for UART communication, the corresponding bit must be set to "1" to make the noise filter active.

When the noise filter is active, detect whether the two clocks are consistent after synchronization through the running clock (f_{MCK}) of the object channel; When the noise filter is invalid, synchronization is performed only through the running clock (f_{MCK}) of the object channel.

The NFEN0 register is set via an 8-bit memory operation command.

After the reset signal is generated, the value of the NFEN0 register changes to "00H".

Figure 14-20 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	SNFEN20	0	SNFEN10	0	SNFEN00

SNFEN20	RxD2 pin noise filter is used or not
0	Noise filter OFF
1	Noise filter ON
When used as the RxD2 pin, SNFEN20 must be set to "1". When used as a function other than the RxD2 pin, the SNFEN20 must be set to "0".	

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".	

Note Bits 7~5, 3, and 1 must be set to "0".

14.3.16 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 must be set (PMCxx)).

For details, please refer to “Chapter 2 Pin Functions”.

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”.

14.4 Run stop mode

Each serial interface of the universal serial communication unit has a stop-and-run mode. Serial communication is not possible in run-stop mode, so power consumption can be reduced. In addition, pins for the serial interface can be used as port functions in run-stop mode.

14.4.1 Stopping the operation by units

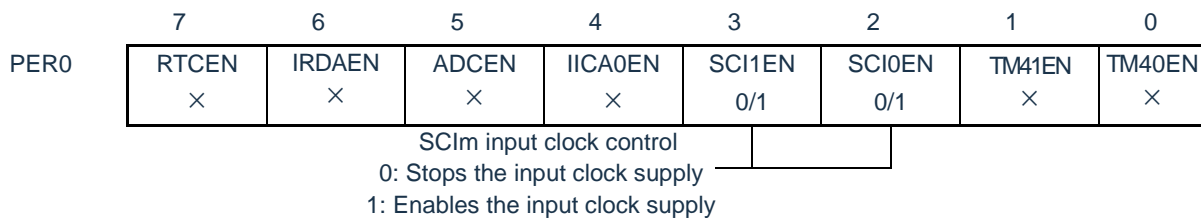
The peripheral enable register 0 (PER0) is to set stop operation in units.

The PER0 register is a register that sets the clock to be allowed or disallowed to be supplied to each peripheral hardware. Reduce power consumption and noise by providing a clock to hardware that is not in use.

To stop universal serial communication unit 0, bit2 (SCI0EN) must be set to “0”; To stop universal serial communication unit 1, bit3 (SCI1EN) must be set to “0”.

Figure 14-21 Configuration of peripheral enable register 0 (PER0) when stopping operation by unit

(a) Peripheral Enable Register 0 (PER0).... Only the corresponding bit of SCIm to be stopped is set to "0".



Note 1 When the SCImEN bit is “0”, the write operation of the control register of the universal serial communication unit m is ignored, and the read values are all initial. However, the following registers are excluded:

- Input switch control register (ISC).
- Noise filter enable register 0 (NFEN0).
- Port multiplexing function configuration register (PxxCFG).
- Port output mode register (POMx).
- Port mode register (PMx).
- Port register (Px).

Remark x: This is an unused bit in the universal serial communication unit (depends on the settings of other peripheral functions).

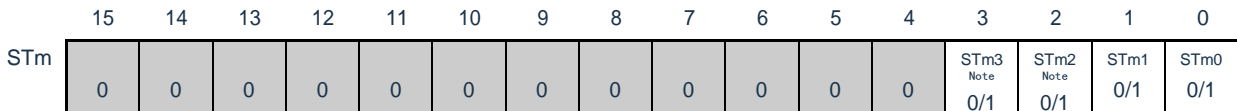
0/1: Set “0” or “1” according to the user's purpose.

14.4.2 Stopping the operation by channels

Stopping operation by channels is set by each of the following registers.

Figure 14-22 Setting of each register when stopping the operation by channels

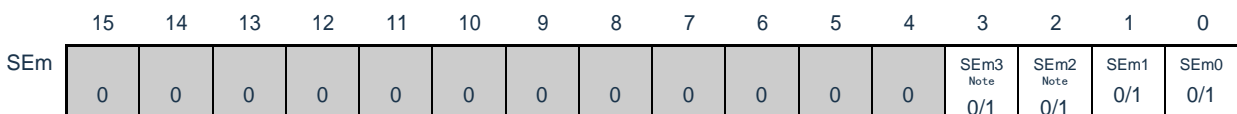
(a) Serial channel stop register m (STm)... This is a register that sets the communication/stop count allowed for each channel.



1: Clear the SE_mn bit "0" and stop the communication operation

※Because the ST_mn bit is the trigger bit, the ST_mn bit is immediately cleared when the SE_mn bit is "0".

(b) Serial channel enable status register m (SEm)... This register represents the running or stopped state of data transmission and reception for each channel.

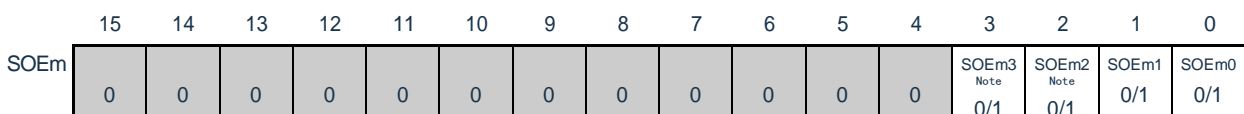


0: Run stop status

※The SEm register is a read-only status register that stops operation through the ST_m register.

For channels that have stopped running, the value of the CKO_mn bit of the SO_m register can be set by software.

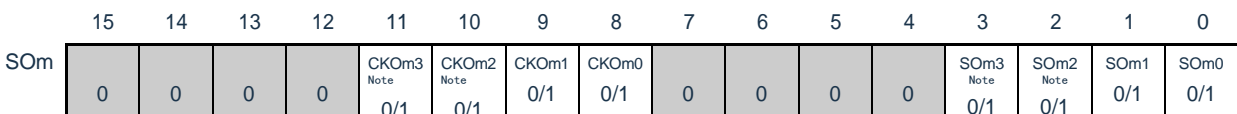
(c)Serial output enable register m (SOEm)... This is a register that sets the serial communication output that enables or stops each channel.



0: Stopping the output by serial communication operation

※ For channels that have stopped the serial output, the value of the SO_mn bit of the SO_m register can be set by software.

(d) Serial output register m (SOm)... This is the buffer register for the serial output of each channel.



1: The output value of the serial clock is "1".

1: The output value of serial data is "1".

※When using the corresponding pin for each channel as a port function, the corresponding CKO_mn bit and SO_mn bit must be set to "1".

Note Limited to universal serial communication unit 0 only.

Remark1. Unit number (m=0, 1) n: channel number (n=0~3).

2. : Cannot be set (set initial value). 0/1: Set "0" or "1" according to the user's purpose.

14.5 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21) communication operation

This is a clock synchronization communication function implemented by a total of 3 wires of serial clock (SCLK) and serial data (SDI and SDO).

[Transmit and receive data]

- 7-bit or 8-bit data length
- Phase control of sending and receiving data
- MSB/LSB preferred choice

[Clock control]

- Master or slave selection
- Phase control of input/output clocks
- Sets the transfer period generated by the prescaler and the in-channel counter.
- Maximum transfer rate ^{Note}

Master communication: $\text{Max.}f_{\text{CLK}}/2$ (SSPI00 only).

Master communication: $\text{Max.}f_{\text{CLK}}/4$

Slave communication: $\text{Max.}f_{\text{MCK}}/6$

[Interrupt function]

- End of transfer interrupt, buffer empty interrupt

[Error detection flag]

- Overflow error

Note It must be used within the scope of the SCLK Cycle Time (t_{KCY}) characteristics. Please refer to the data sheet for details.

Channels 0 to 3 of SCI0 and channels 0 to 1 of SCI1 support 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21) channels.

3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21) has the following 6 types of communication operation:

- Master transmission (see 14.5.1).
- Master reception (see 14.5.2).
- Master transmission and reception (refer to 14.5.3).
- Slave transmission (see 14.5.4).
- Slave reception (see 14.5.5).
- Slave transmission and reception (see 14.5.6).

14.5.1 Master transmission

Master transmission refers to the operation of this product output transmission clock and sending data to other devices.

3-wire serial I/O	SSPI00	SSPI01	SSPI10	SSPI11	SSPI20	SSPI21
Object channel	Channel 0 of SCI0	Channel 1 of SCI0	Channel 2 of SCI0	Channel 3 of SCI0	Channel 0 of SCI1	Channel 1 of SCI1
Used pin	SCLK00, SDO00	SCLK01, SDO01	SCLK10, SDO10	SCLK11, SDO11	SCLK20, SDO20	SCLK21, SDO21
interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11	INTSSPI20	INTSSPI21
Error detection flag	None					
Transmitted data length	7 or 8 bits					
Transfer rate Note	Max. $f_{CLK}/2$ [Hz] (SSPI00 only), $f_{CLK}/4$ [Hz] Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [Hz] f_{CLK} : System clock frequency					
Data phase	It can be selected via the DAPmn bit of the SCRmn register. <ul style="list-style-type: none"> DAPmn=0: The data output starts when the serial clock starts running. DAPmn=1: Starts data output half a clock before the serial clock starts running. 					
Clock phase	It can be selected via the CKPmn bit of the SCRmn register. <ul style="list-style-type: none"> CKPmn=0: Positive phase CKPmn=1: Negative phase 					
Data direction	MSB first or LSB first					

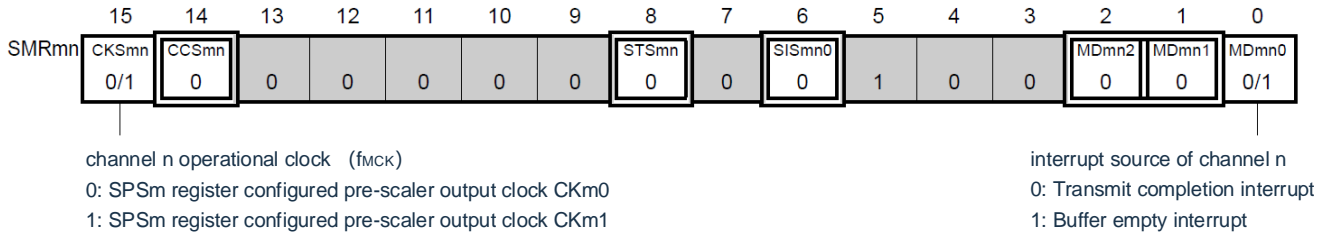
Note It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Remark m: Unit number (m=0, 1) n: channel number (n=0~3) mn=00~ 03, 10~11.

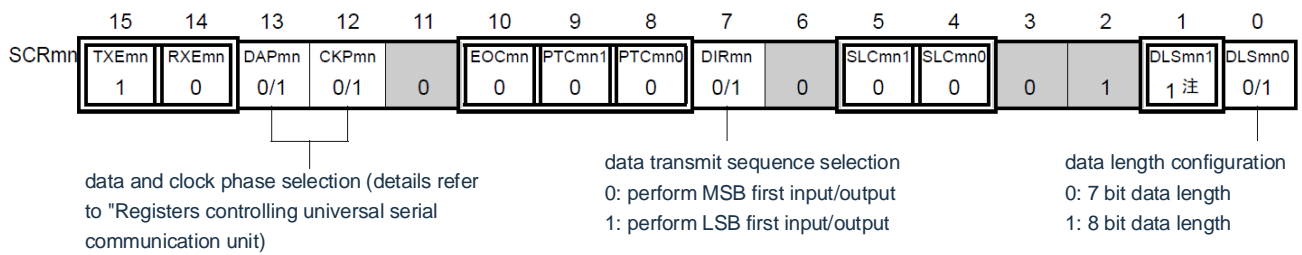
(1) Register setting

Figure 14-233 wire serial I/O(SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)
Example of register settings when the master is transmitted

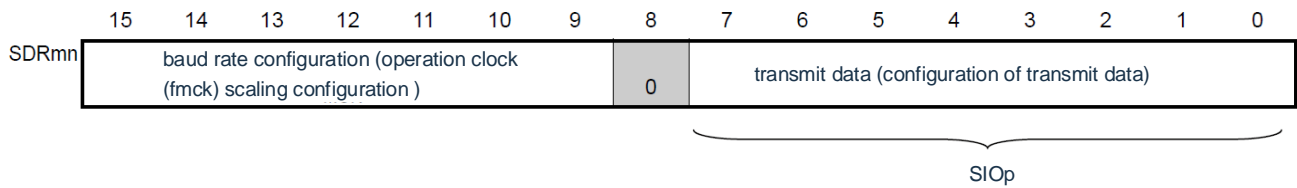
(a) serial mode register mn (SMRmn)



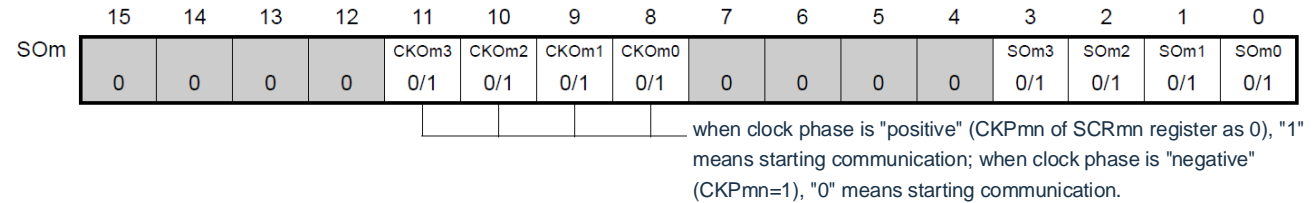
(b) serial communication operation configuration registermn mn(SCRmn)



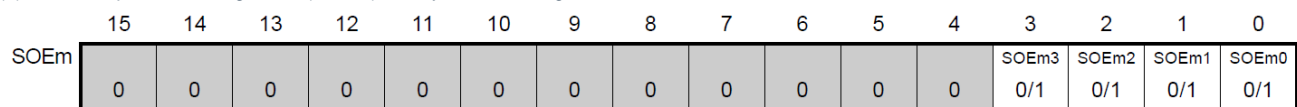
(c) serial data registermn mn(SDRmn)



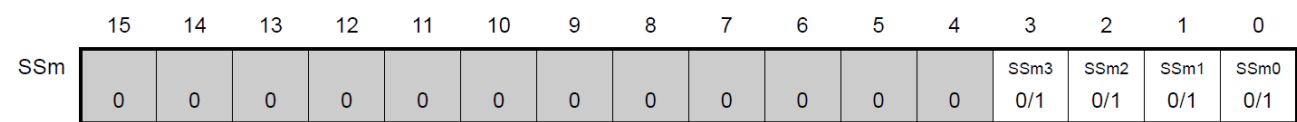
(d) serial output register m(SOm)Only configure bit of target channel



(e) serial output enable registerm (SOEm)....only set bit of target channel to 1.



(f) serial channel start registerm (SSm)....only set bit of target channel to 1.



Note: Limited to SCR00, SCR01 register, and others are fixed to "1".

Note 1. m: Unit number (m=0, 1) n: channel number (n=0~3) mn=00~ 03, 10~11.

2. : Cannot be set (set initial value). 0/1: Set "0" or "1" according to the user's purpose.

(2) Operation steps

Figure 14-24 Initial setup steps for master transmission

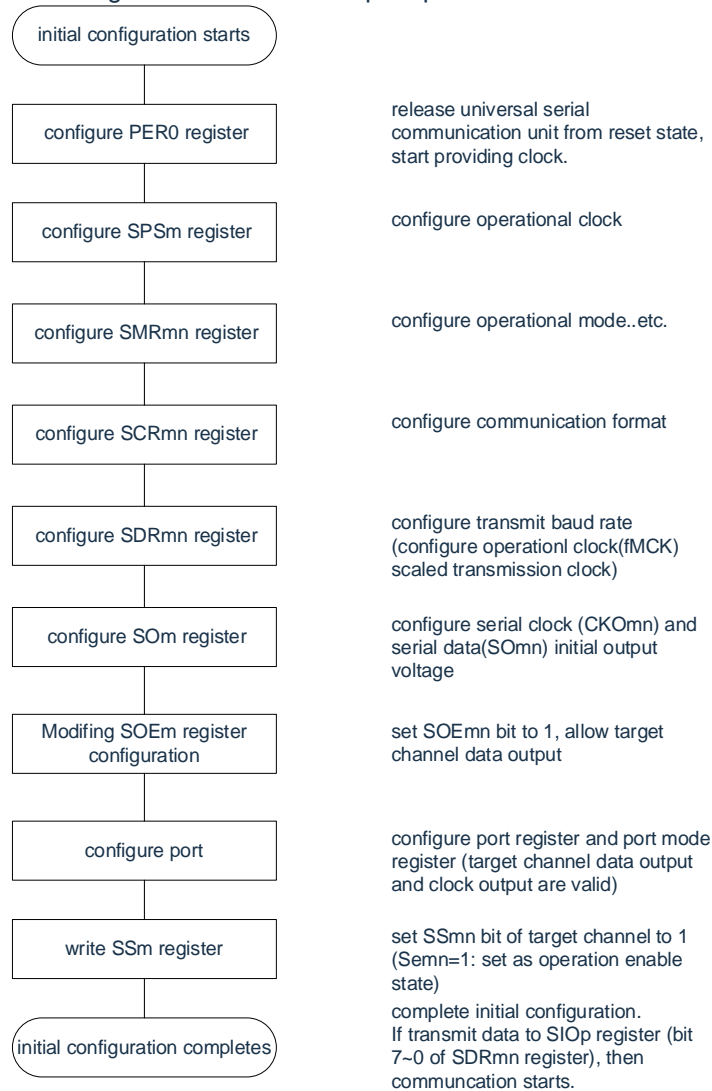


Figure 14-25 Stop steps for master transmission

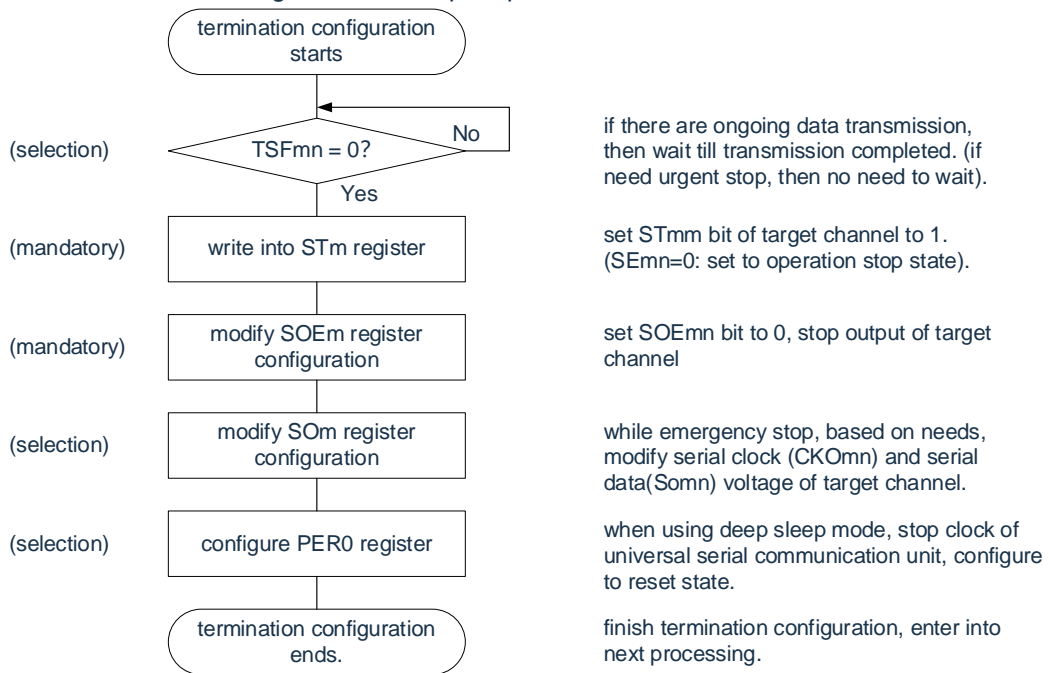
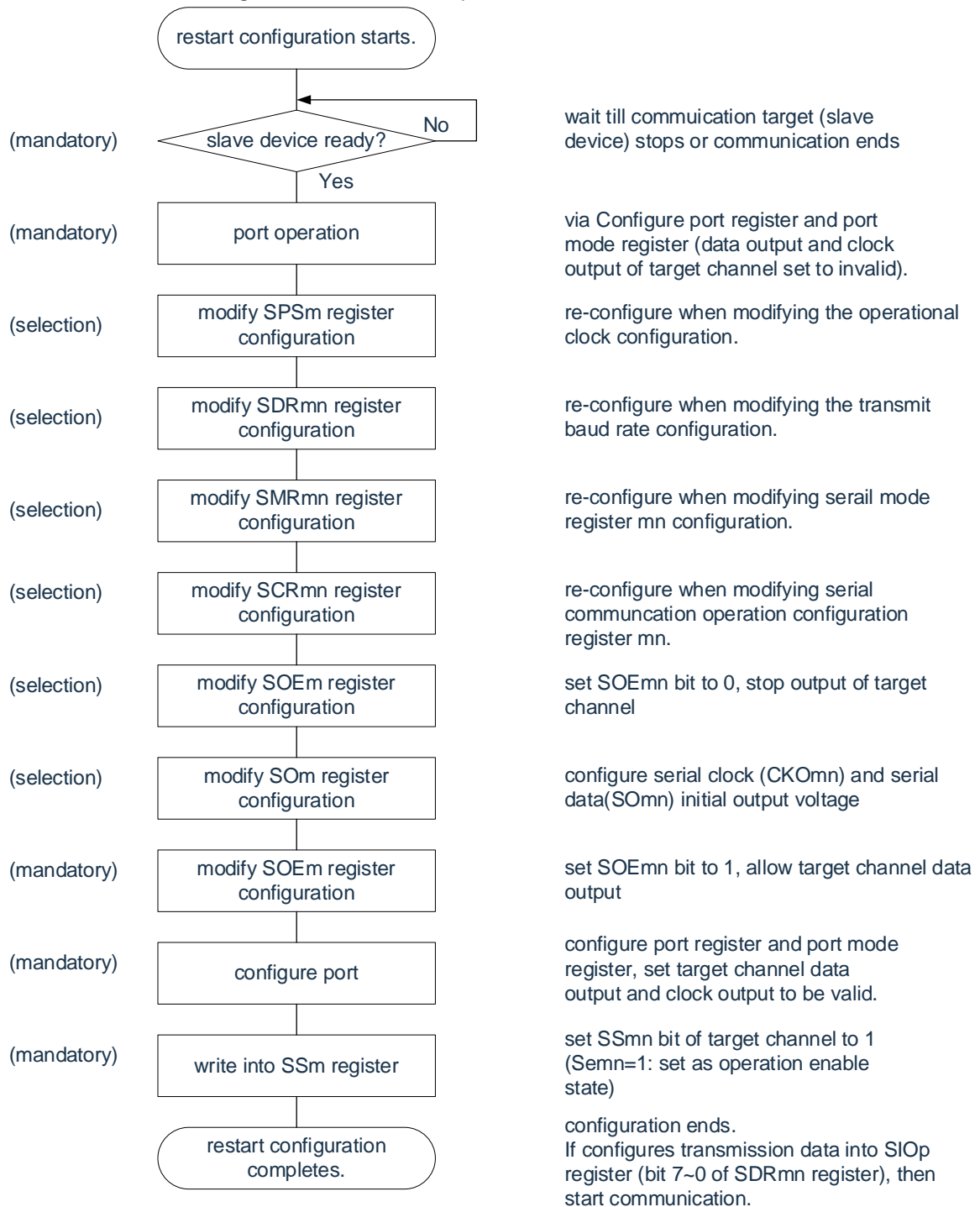


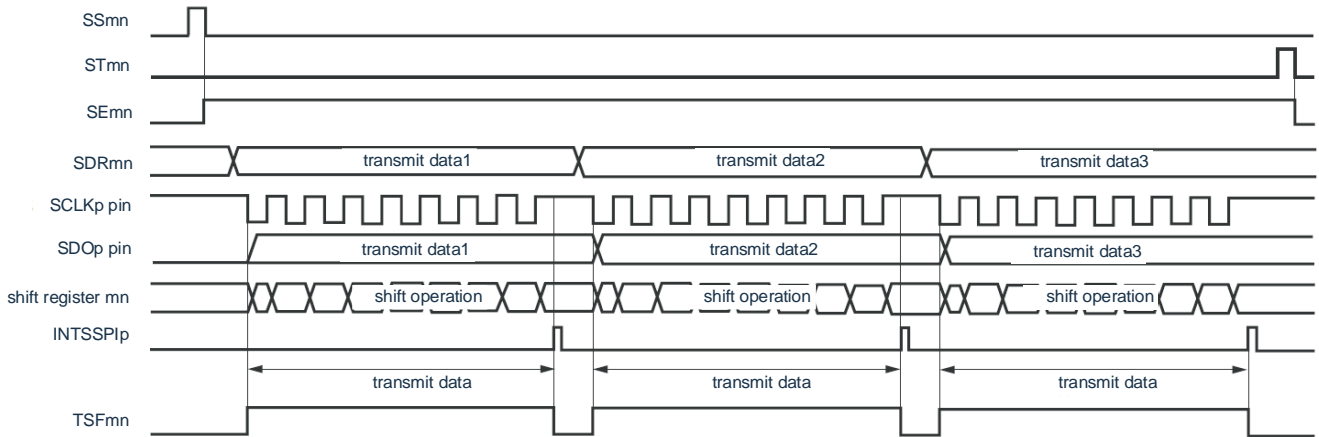
Figure 14-26 Restart steps for master transmission



Remark: If you override PER0 in the abort setting to stop the clock, you must wait until the communication object (slave device) stops or the communication is over to make the initial setting instead of starting the setting again.

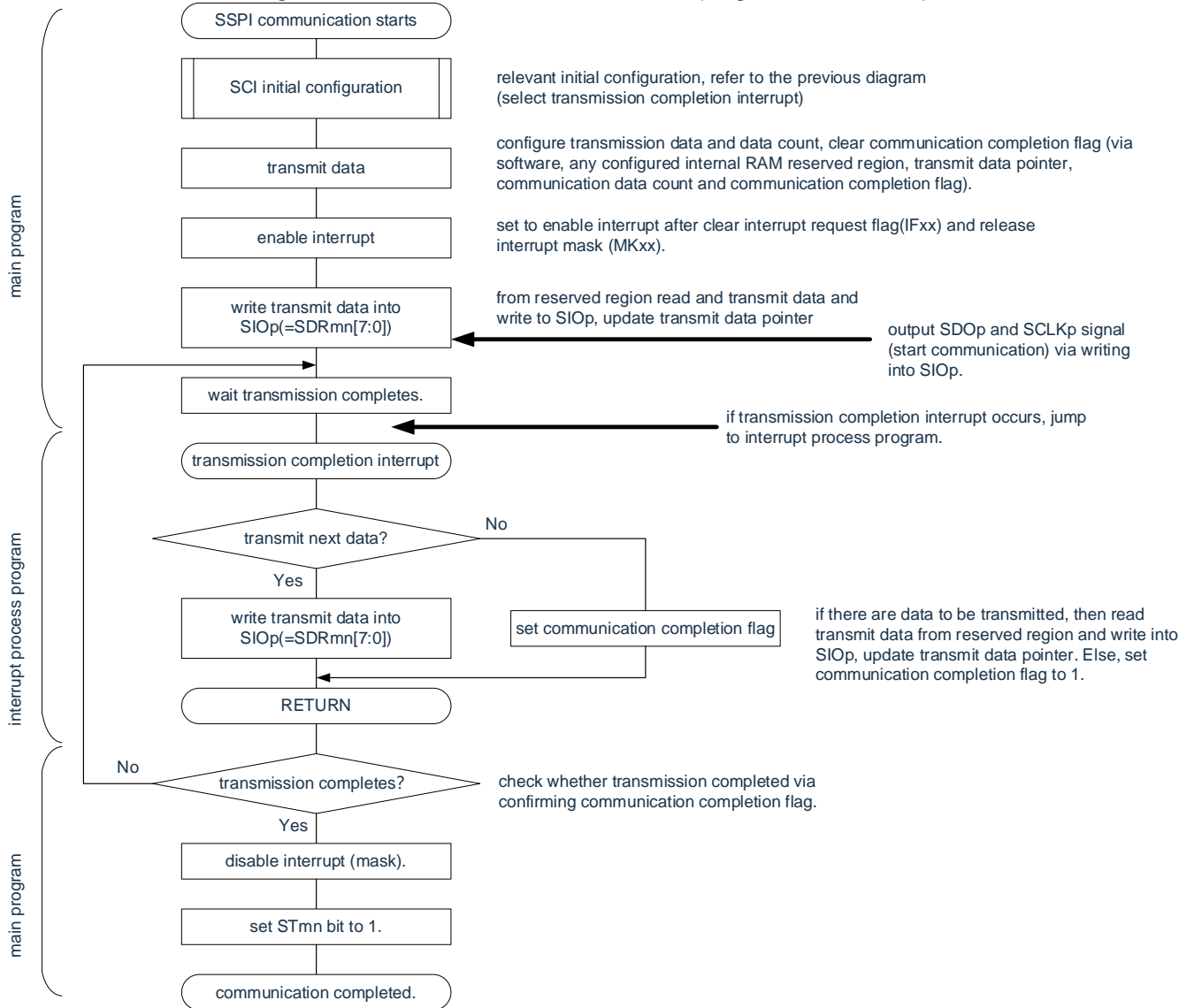
(3) Processing flow (single transmit mode)

Figure 14-27 Timing diagram of master transmission (single transmit mode) (type 1: DAPmn=0, CKPmn=0)



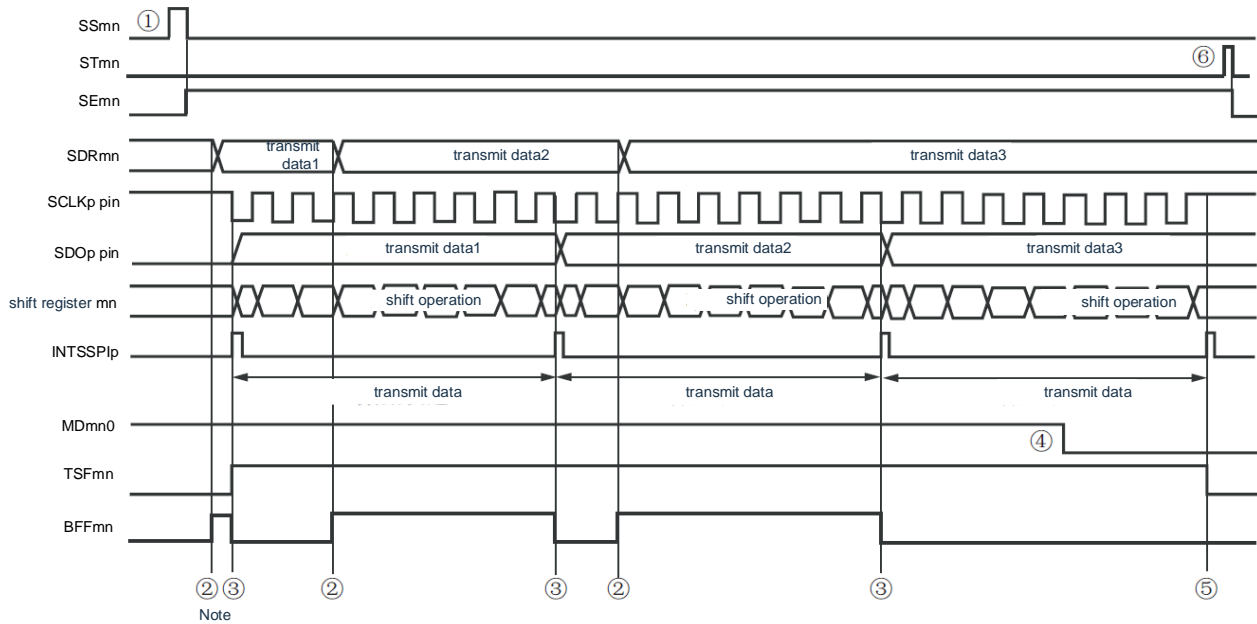
Remark m: Unit number (m=0, 1) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)
 mn=00~03, 10~11

Figure 14-28 Flow of master transmission (single transmit mode)



(4) Processing flow (continuous transmit mode)

Figure 14-29 Timing diagram of the master transmission (continuous transmit mode)
(type 1: DAPmn=0, CKPmn=0)

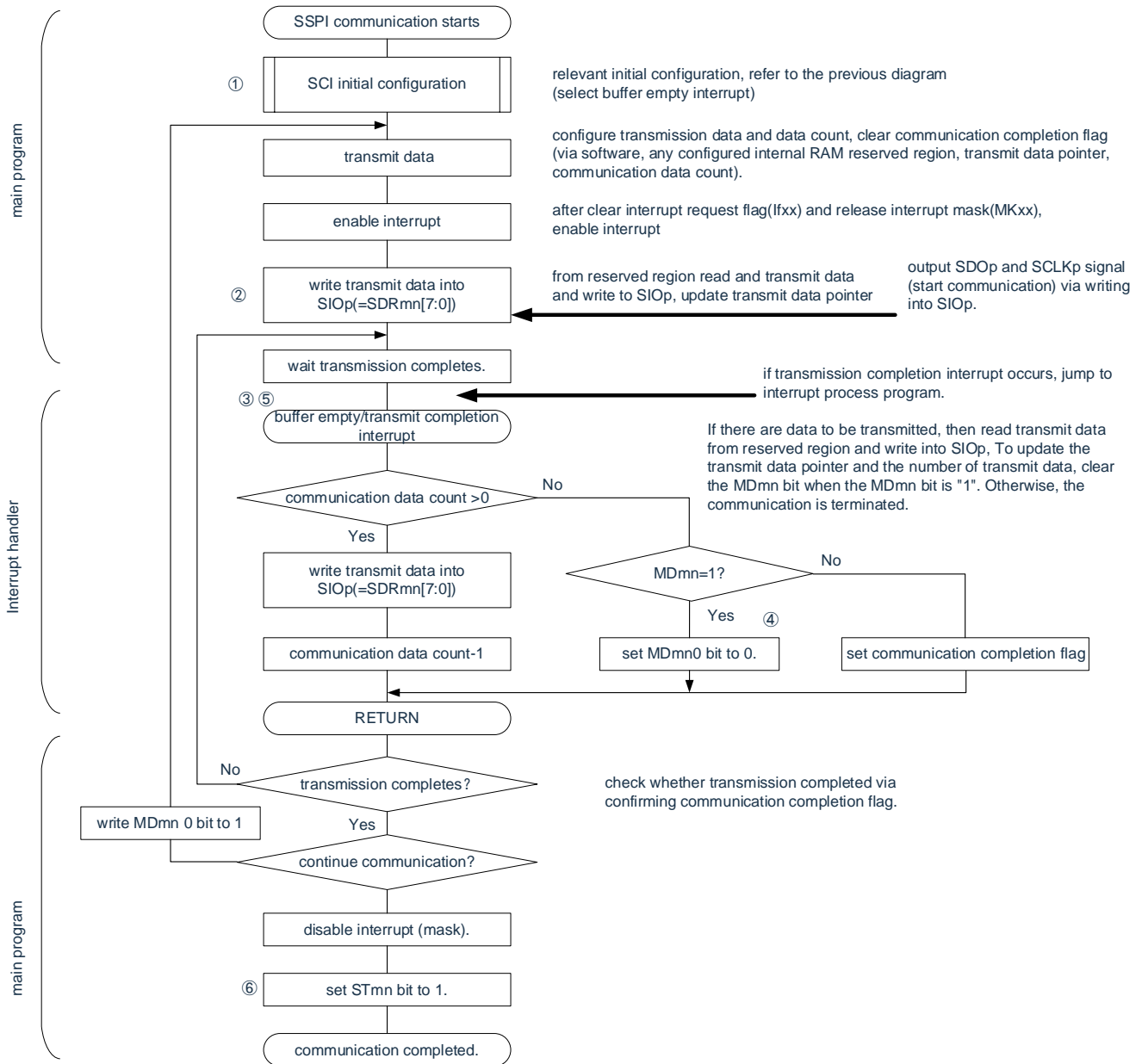


Note If the BFFmn bit of the serial status register mn (SSRmn) is “1” (when valid data is saved in the serial data register mn (SDRmn)) is given The SDRmn register writes the transmitted data and overrides the transmitted data.

Notice The MDmn0 bit of the serial mode register mn (SMRmn) can be overridden even during operation. However, in order to catch up with the end of the transmission interruption of the last transmitted data, it must be overwritten before the last bit of transmission begins.

Remark m: Unit number (m=0, 1) **n:** channel number (n=0~3) **p:** SSPI number (p=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11

Figure 14-30 Flowchart of master transmission (continuous transmit mode)



Remark ① to ⑥ correspond to ① to ⑥ in "Figure 14-29 Timing diagram of the master transmission (continuous transmit mode)".

14.5.2 Master reception

Master reception refers to the operation of this product output transmission clock and receiving data from other devices.

3-wire serial I/O	SSPI00	SSPI01	SSPI10	SSPI11	SSPI20	SSPI21
Object channel	Channel 0 of SCI0	Channel 1 of SCI0	Channel 2 of SCI0	Channel 3 of SCI0	Channel 0 of SCI1	Channel 1 of SCI1
Used pin	SCLK00, SDO00	SCLK01, SDO01	SCLK10, SDO10	SCLK11, SDO11	SCLK20, SDO20	SCLK21, SDO21
Interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11	INTSSPI20	INTSSPI21
Error detection flag	Can select transmit complete interrupt (single transmit mode) or buffer empty interrupt (continuous transmit mode).					
Transmitted data length	Only the overflow error detection flag (OVFmn)					
Transfer rate ^{Note}	7 or 8 bits					
Data phase	Max. $f_{CLK}/2$ [Hz](SSPI00 only), $f_{CLK}/4$ [Hz] Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [Hz] f_{CLK} : System clock frequency					
Clock phase	It can be selected via the DAPmn bit of the SCRmn register. • DAPmn=0: Data output starts when the serial clock starts running. • DAPmn=1: Starts data output half a clock before the serial clock starts running.					
Data direction	It can be selected by the CKPmn bit of the SCRmn register. • CKPmn=0: Positive phase • CKPmn=1: Negative phase					
	MSB first or LSB first					

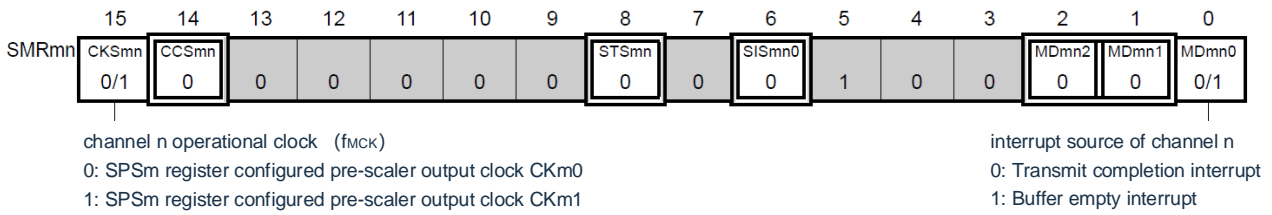
Note It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Remark m: Unit number (m=0, 1) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11.

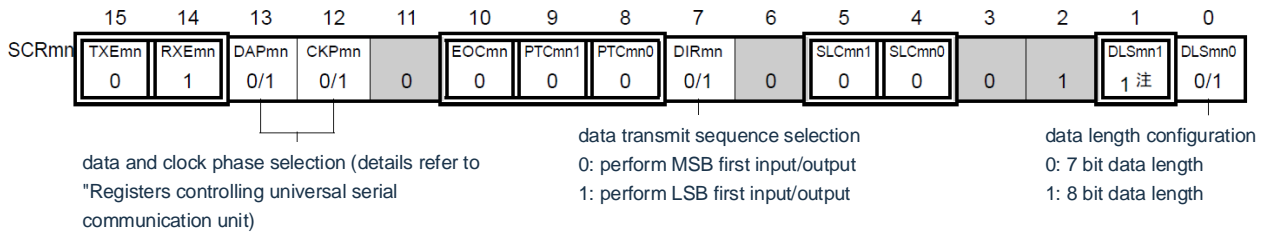
(1) Register setting

Figure 14-31 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)
Example of register setting content when the master receives

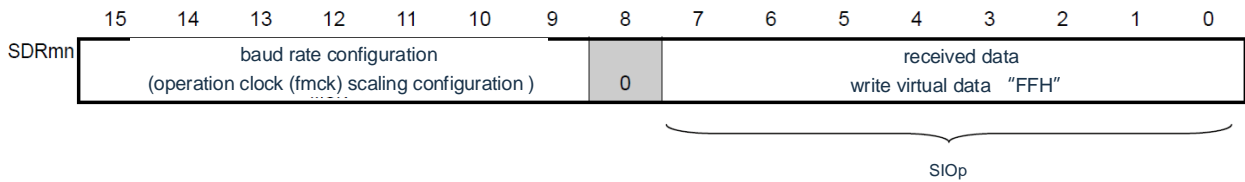
(a) serial mode register mn(SMRmn)



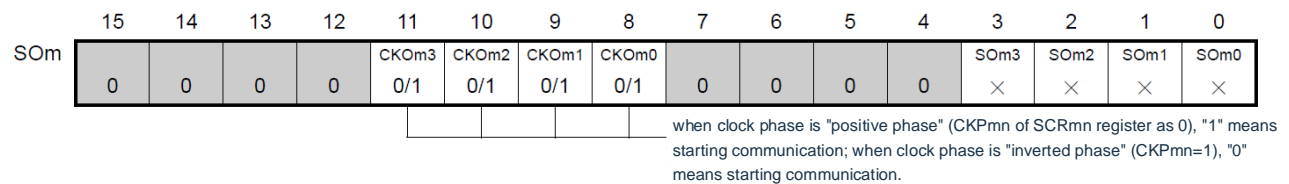
(b) serial communication operation configuration registermn mn(SCRmn)



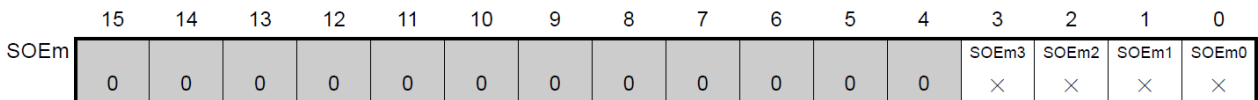
(c) serial data register mn(SDRmn) (low 8 bit: SIOp)



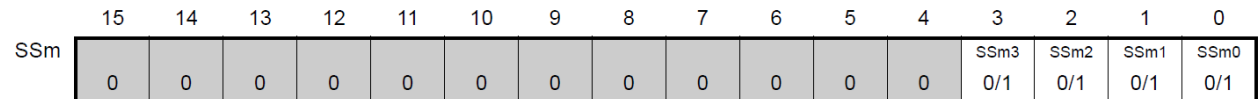
(d) serial output register (SOm)..... Only configure bit of target channel



(e) serial output register m(SOEm)Not used in this mode



(f) serial channel start register m (SSm) Only set bit of target channel to 1.



Note Limited to SCR00 register and SCR01 register, others are fixed as "1".

Remark1.m: Unit number (m=0, 1) n: Channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21.))
mn=00~03, 10~11

2. : Fixed in SSPI master receive mode : Cannot be set (initial value)
x: This is the bit that cannot be used in this mode (set 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) Operation steps

Figure 14-32 Initial setup steps for master reception

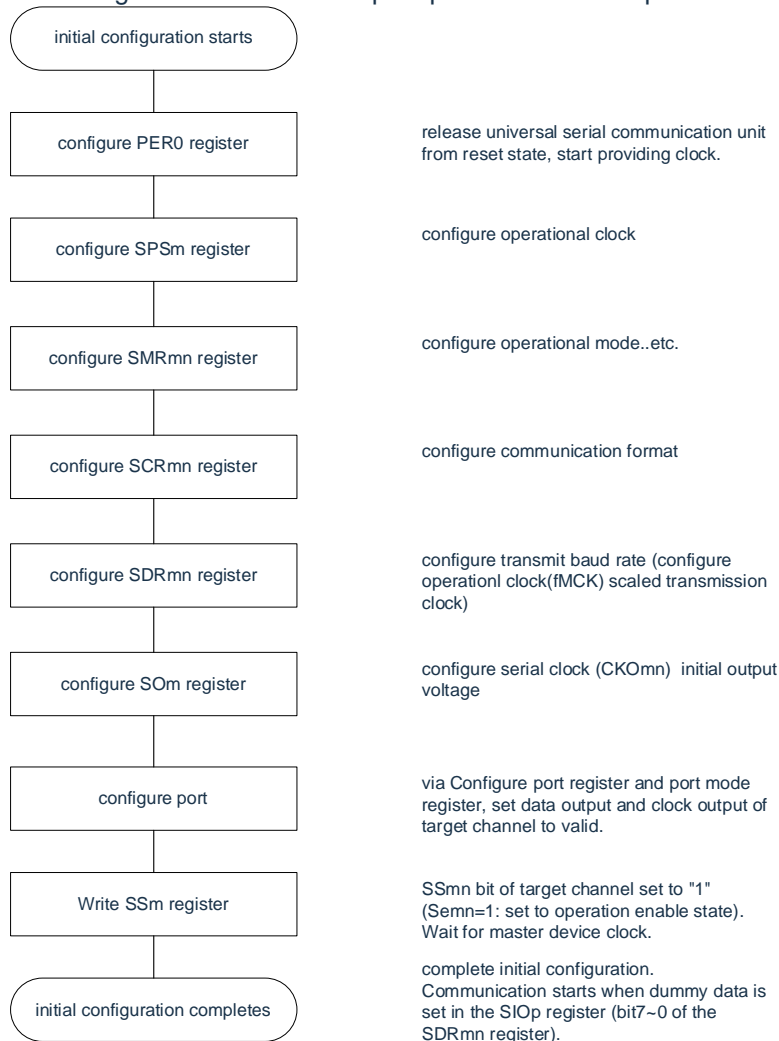


Figure 14-33 Stop steps for master reception

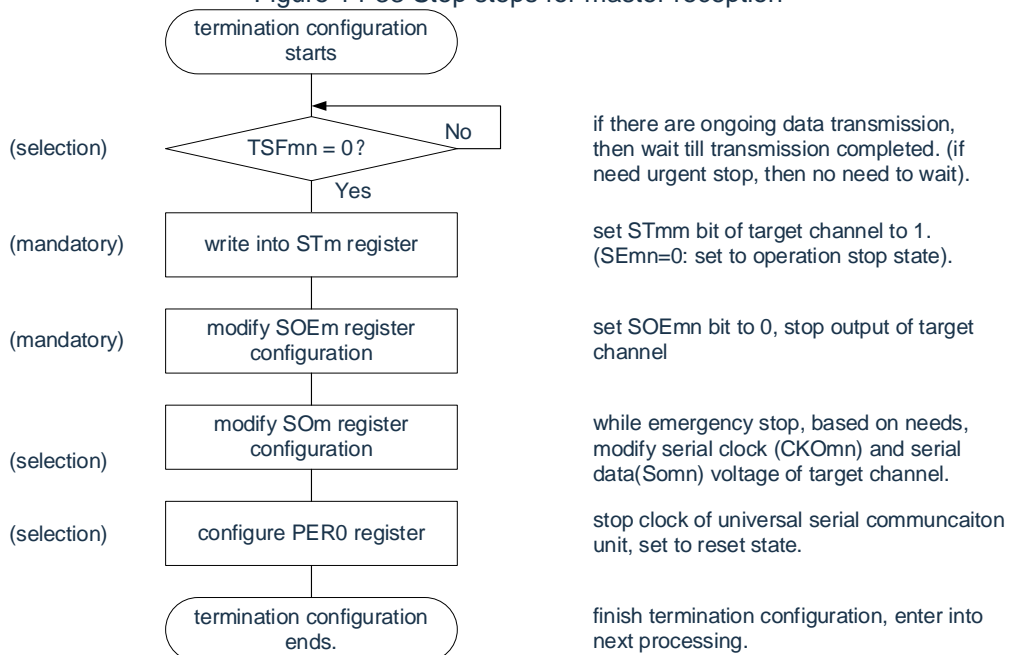
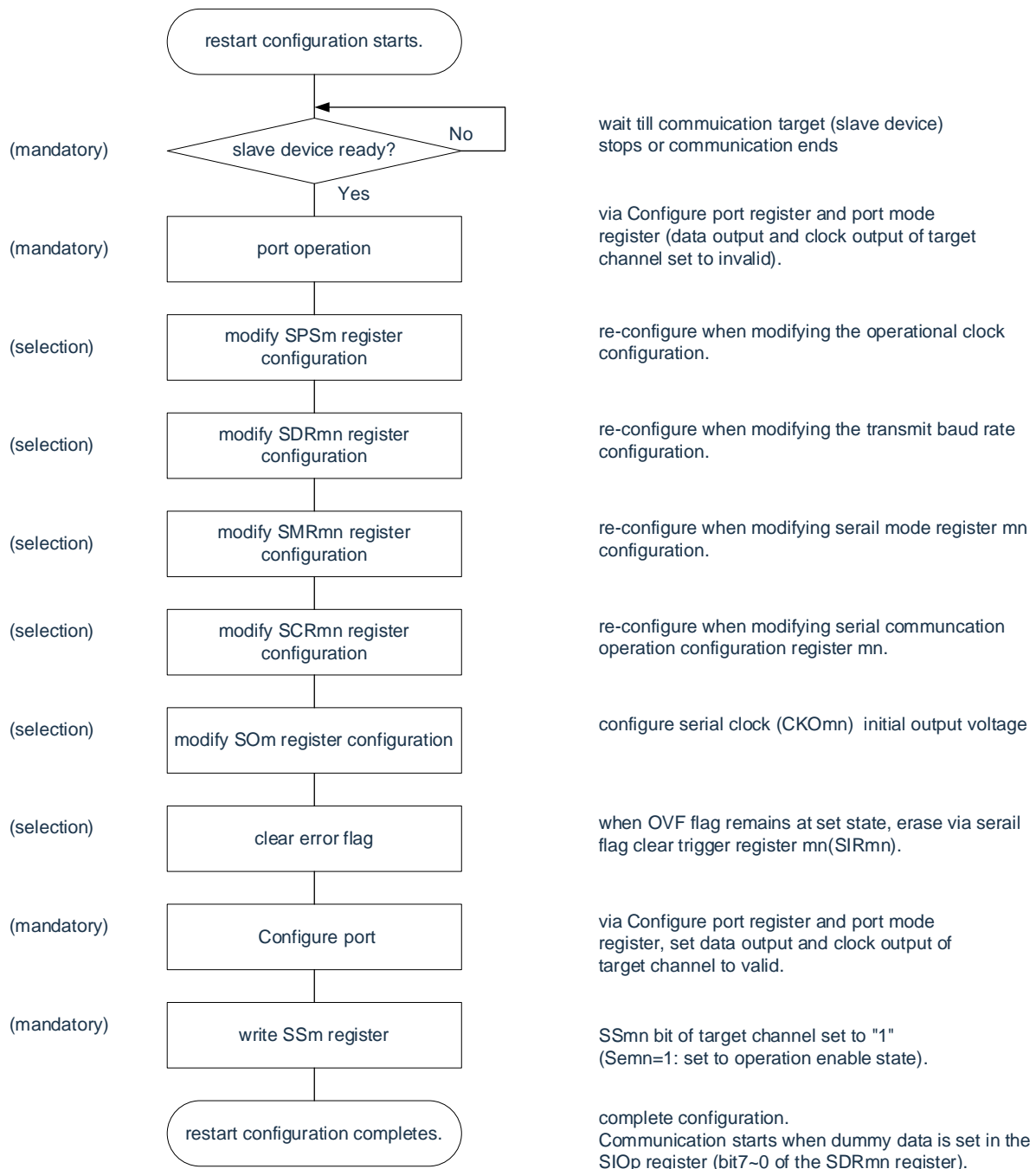


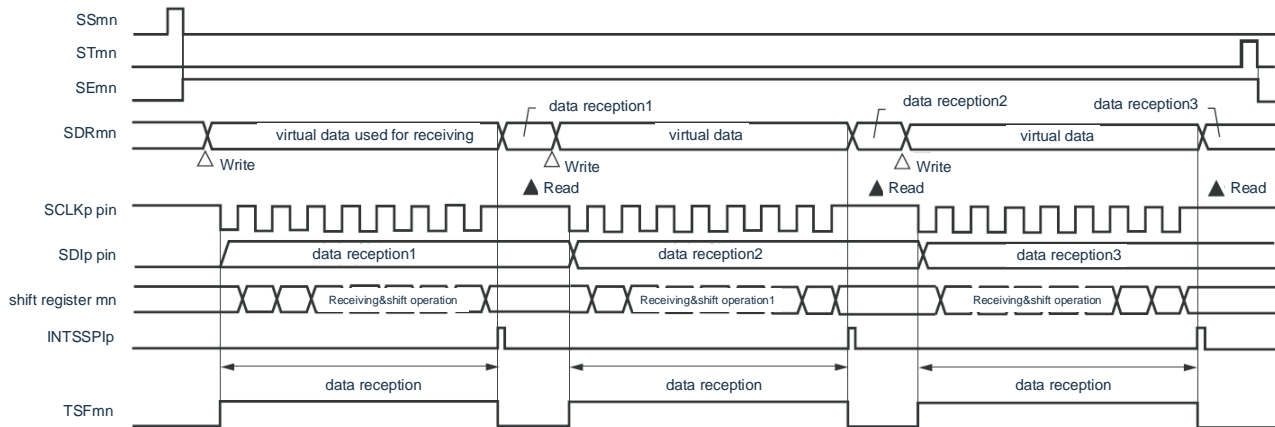
Figure 14-34 Restart steps for master reception



Remark If you override PER0 in the abort setting to stop the clock, you must wait until the communication object (slave device) stops or the communication is over to make the initial setting instead of starting the setting again.

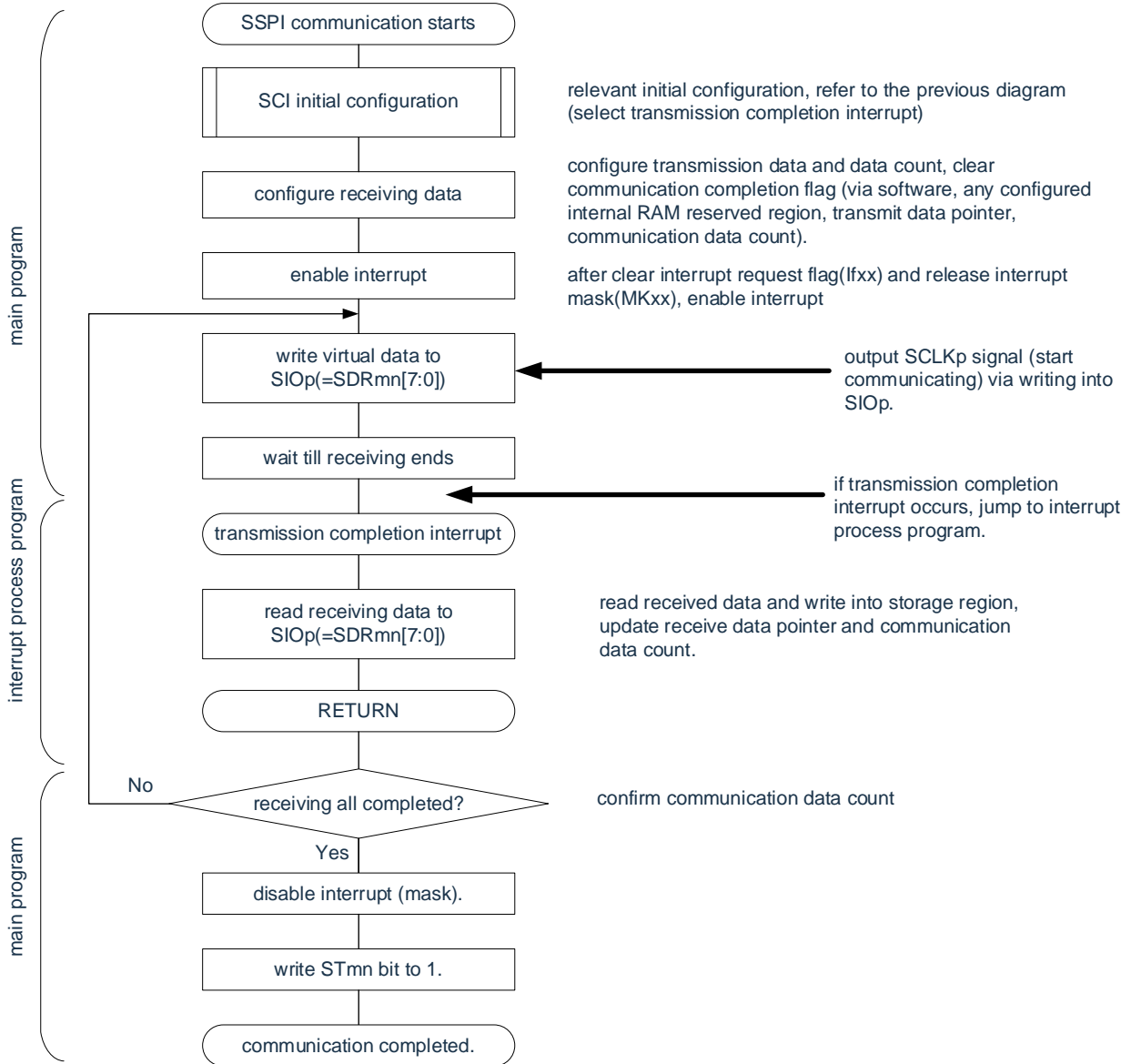
(3) Processing flow (single receive mode)

Figure 14-35 Timing diagram of master reception (single receive mode) (type 1: DAPmn=0, CKPmn=0)



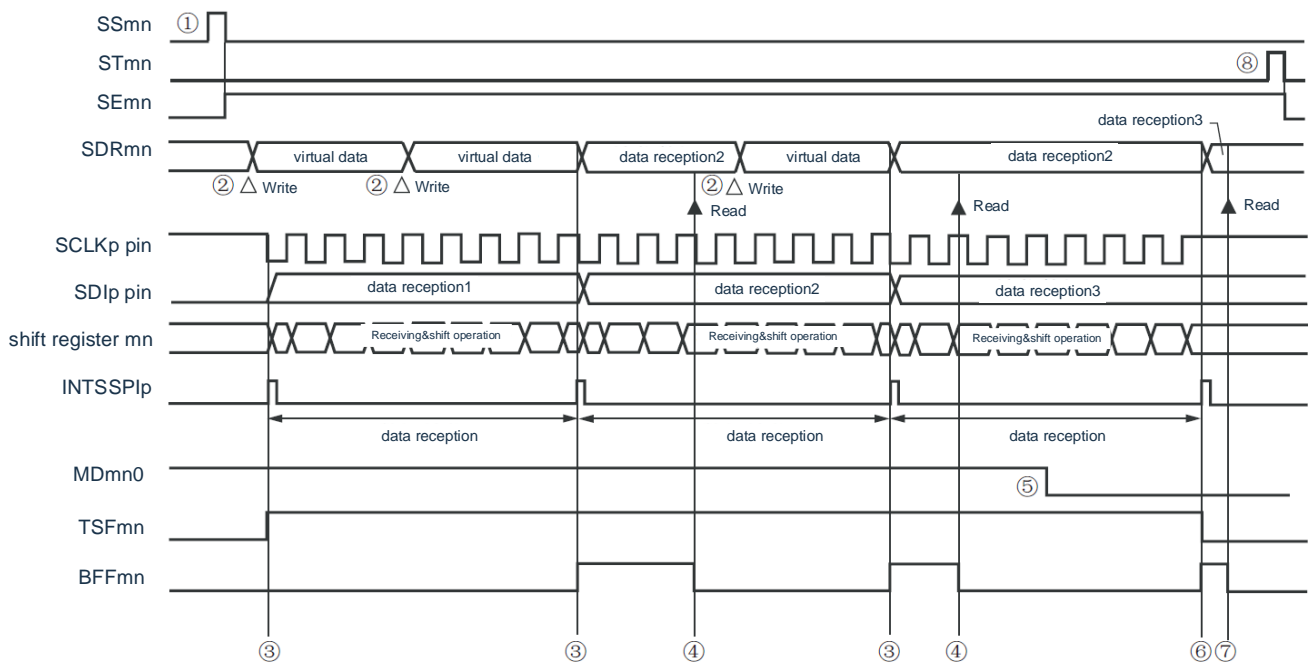
Remark m: Unit number (m=0, 1) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11

Figure 14-36 Flowchart of master reception (single receive mode)



(4) Processing flow (continuous receive mode)

Figure 14-37 Timing diagram of master reception (continuous receive mode)
(type 1: DAPmn=0, CKPmn=0).

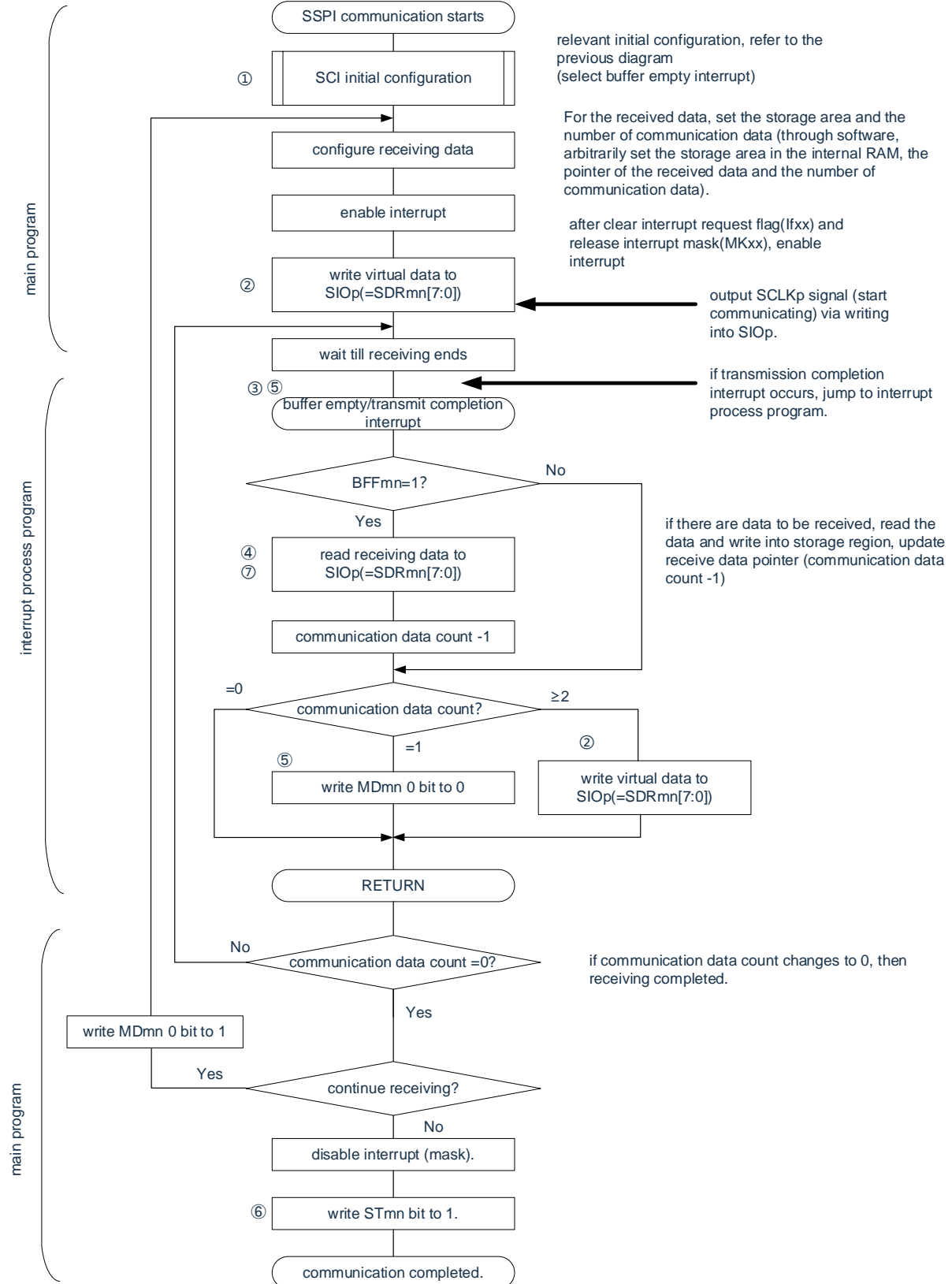


Notice The MDmn0 bit can be overridden even during operation. However, in order to catch up with the end of transmission interruption of the last received data, it must be overridden before the last bit of reception begins.

Remark 1. ① to ⑧ in the figure correspond to the ① to ⑧ in “Figure 14-38 Flowchart of master reception (continuous receive mode)”.

- 2. m: Unit number (m=0, 1) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11

Figure 14-38 Flowchart of master reception (continuous receive mode)



Remark ① to ⑧ in the diagram correspond to ① to ⑧ in "Figure 14-37 Timing diagram of master reception (continuous receive mode)".

14.5.3 Master transmission and reception

Master transmission and reception of the master refers to the operation of the output transmission clock of this product and the transmission and reception of data with other devices.

3-wire serial I/O	SSPI00	SSPI01	SSPI10	SSPI11	SSPI20	SSPI21
Object channel	SCI0 Channel 0	SCI0 Channel 1	SCI0 Channel 2	SCI0 Channel 3	SCI1 Channel 0	SCI1 Channel 1
Used pin	SCLK00, SDI00, SDO00	SCLK01, SDI01, SDO01	SCLK10, SDI10, SDO10	SCLK11, SDI11, SDO11	SCLK20, SDI20, SDO20	SCLK21, SDI21, SDO21
Interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11	INTSSPI20	INTSSPI21
Error detection flag	Can select transmit complete interrupt (single transmit mode) or buffer empty interrupt (continuous transmit mode).					
Transmitted data length	Only the overflow error detection flag (OVFmn)					
Transfer rate ^{Note}	7 or 8 bits					
Data phase	Max. $f_{CLK}/2$ [Hz](Only.SSPI00) $f_{CLK}/4$ [Hz] Min. $n.f_{CLK}/(2 \times 2^{15} \times 128)$ [Hz] f_{CLK} : System clock frequency					
Clock phase	It can be selected via the DAPmn bit of the SCRmn register. • DAPmn=0: Data output starts when the serial clock starts running. • DAPmn=1: Starts data output half a clock before the serial clock starts running.					
Data direction	It can be selected by the CKPmn bit of the SCRmn register. • CKPmn=0: Positive phase • CKPmn=1: Negative phase					
	MSB first or LSB first					

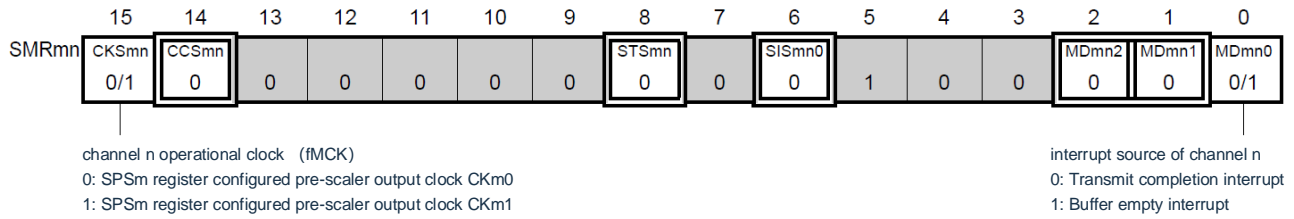
Note It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Remark m: Unit number (m=0, 1) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11

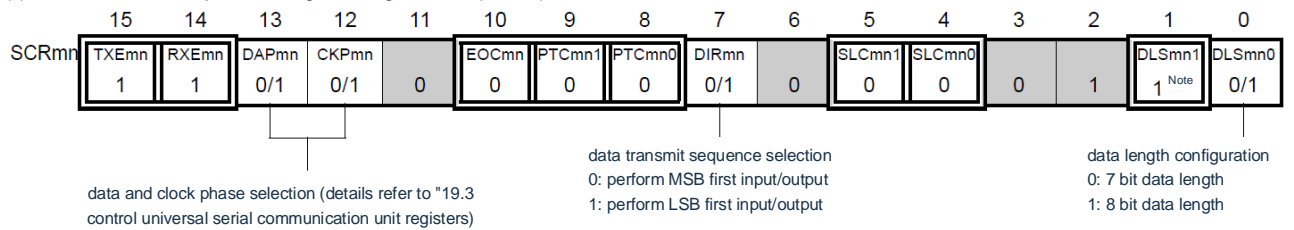
(1) Register setting

Figure 14-39 3-wire serial I/O(SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)
Example of register settings when the master transmits and receives

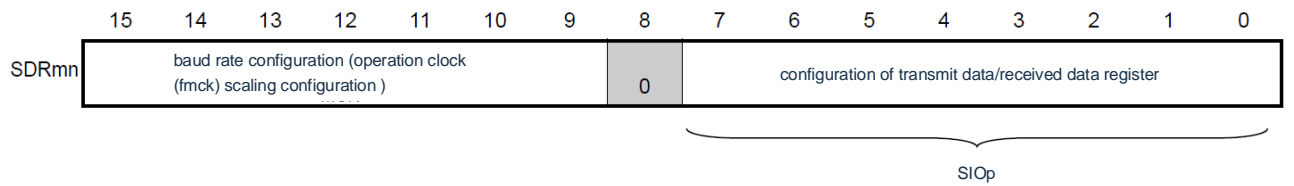
(a) serial mode register mn (SMRmn)



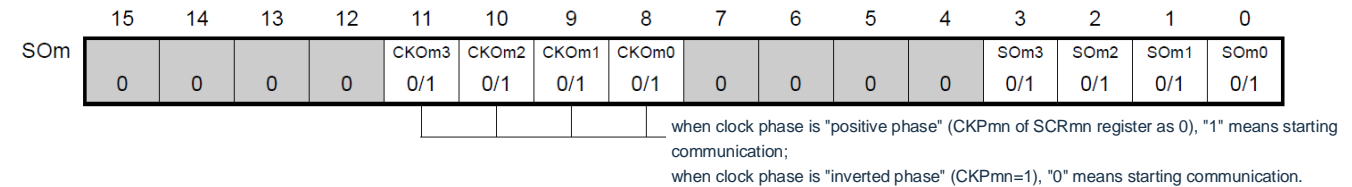
(b) serial communication operation configuration registermn mn(SCRmn)



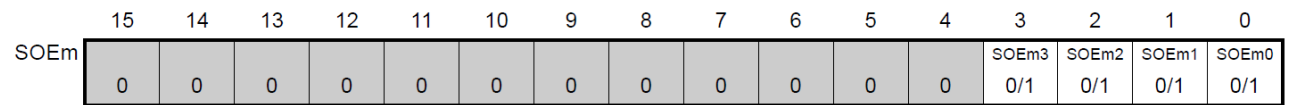
(c) serial data register mn (SDRmn) (low 8 bit: SIOp)



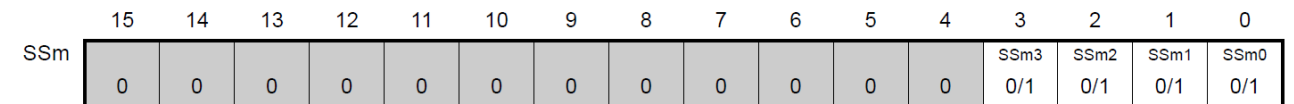
(d) serial output register m(SOm)Only configure bit of target channel



(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.



Note Limited to SCR00 register and SCR01 register, the others are fixed as "1".

Notice 1. m: Unit number (m=0, 1) n: Channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11

2. : Fixed in SSPI master T&R modes. : Cannot be set (initial value).

0/1: Set "0" or "1" according to the user's purpose.

(2) Operation steps

Figure 14-40 Initial setup steps for master transmission and reception

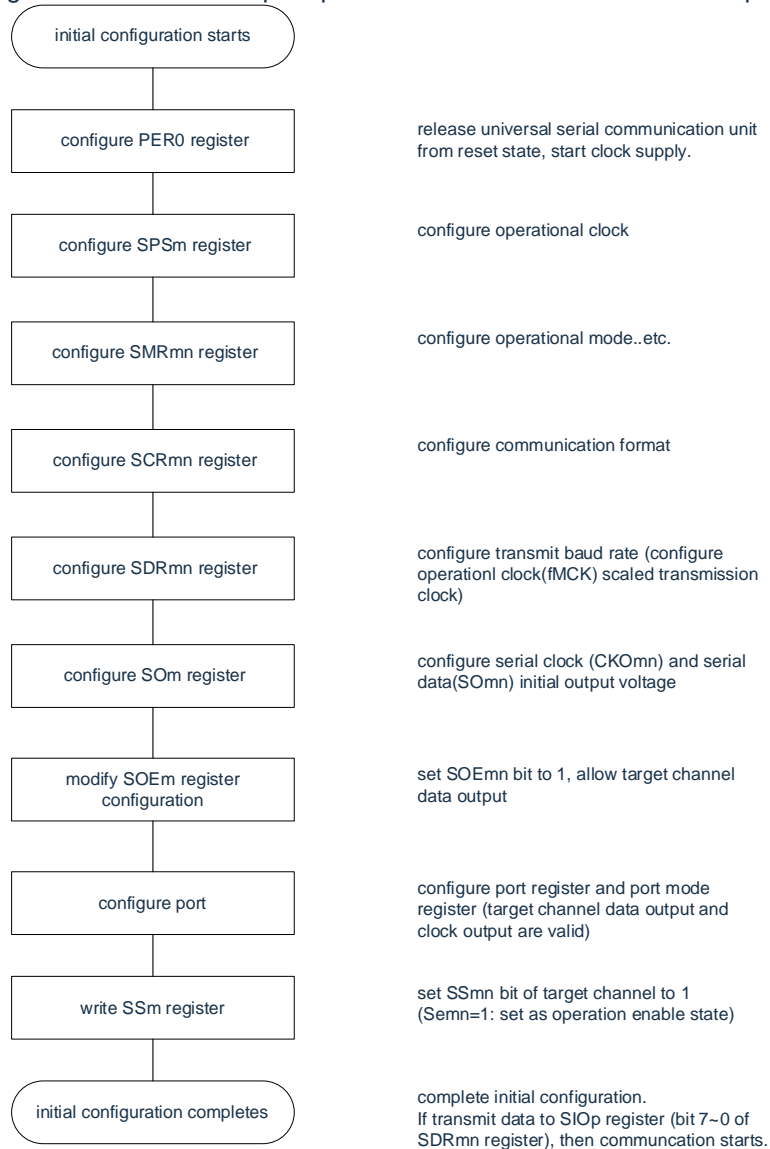


Figure 14-41 Stop steps for master transmission and reception

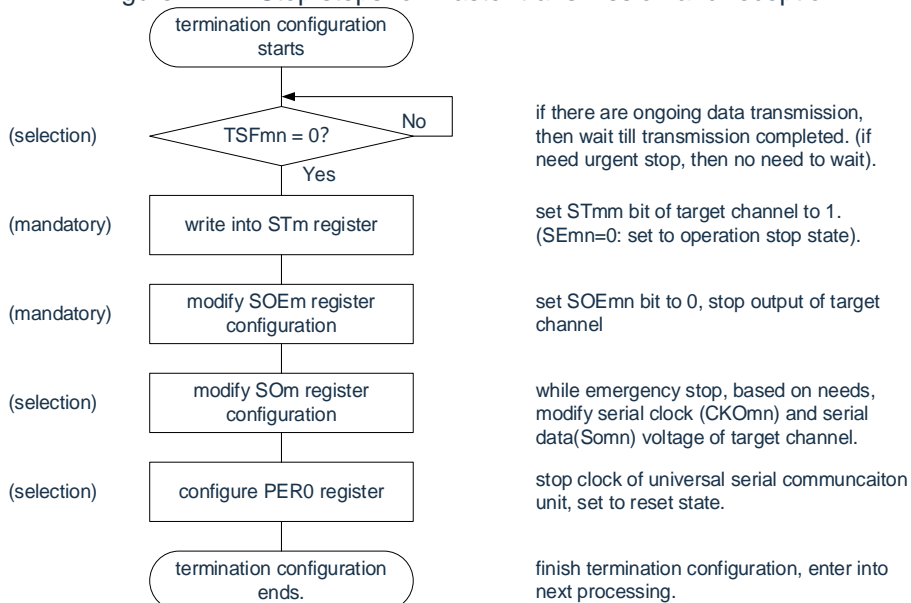
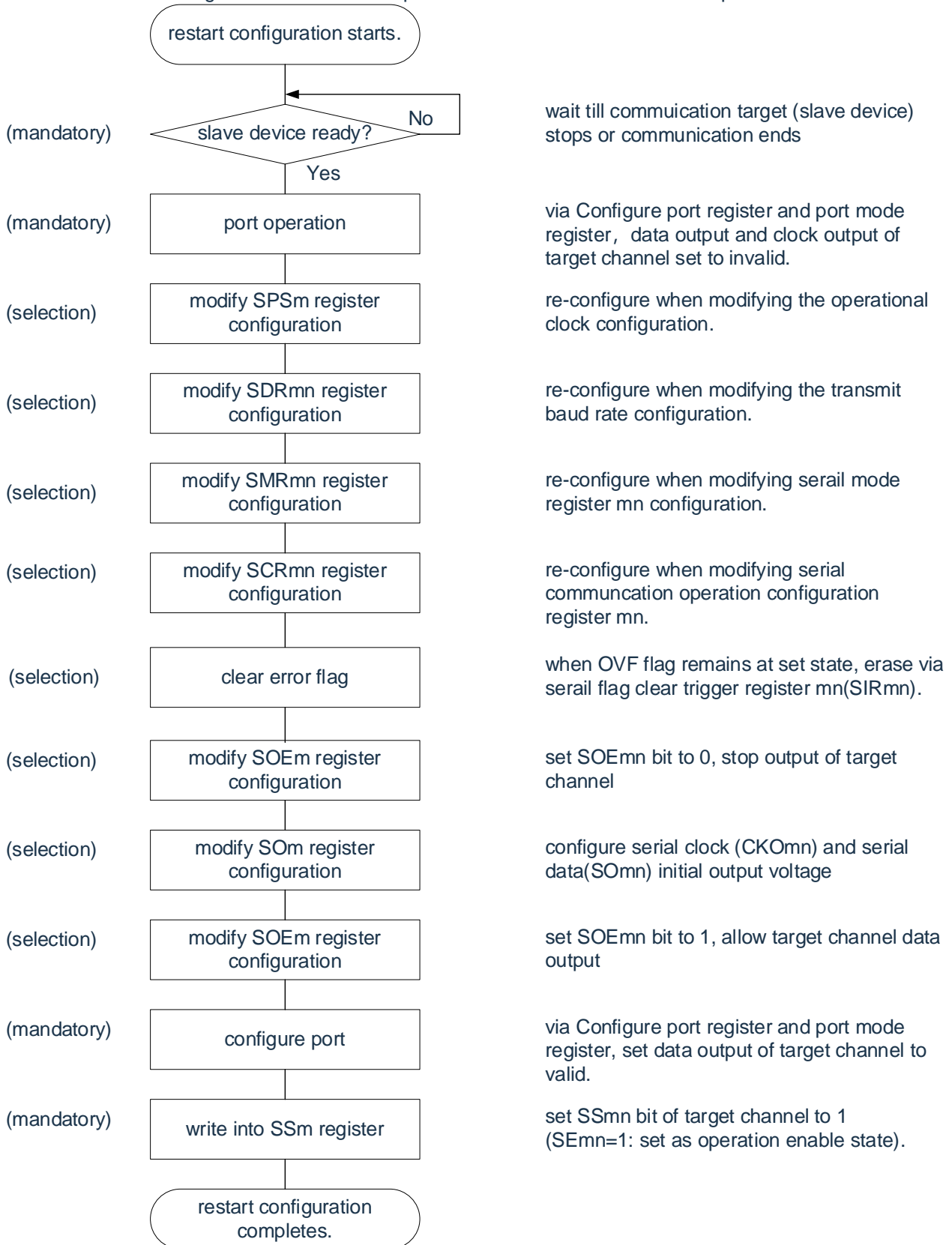
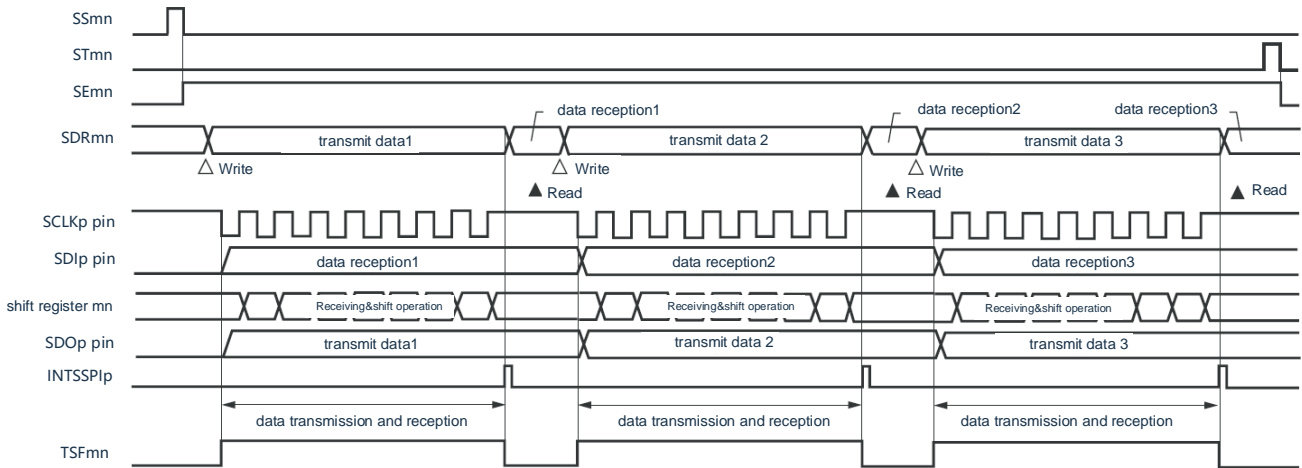


Figure 14-42 Restart steps for master transmission and reception



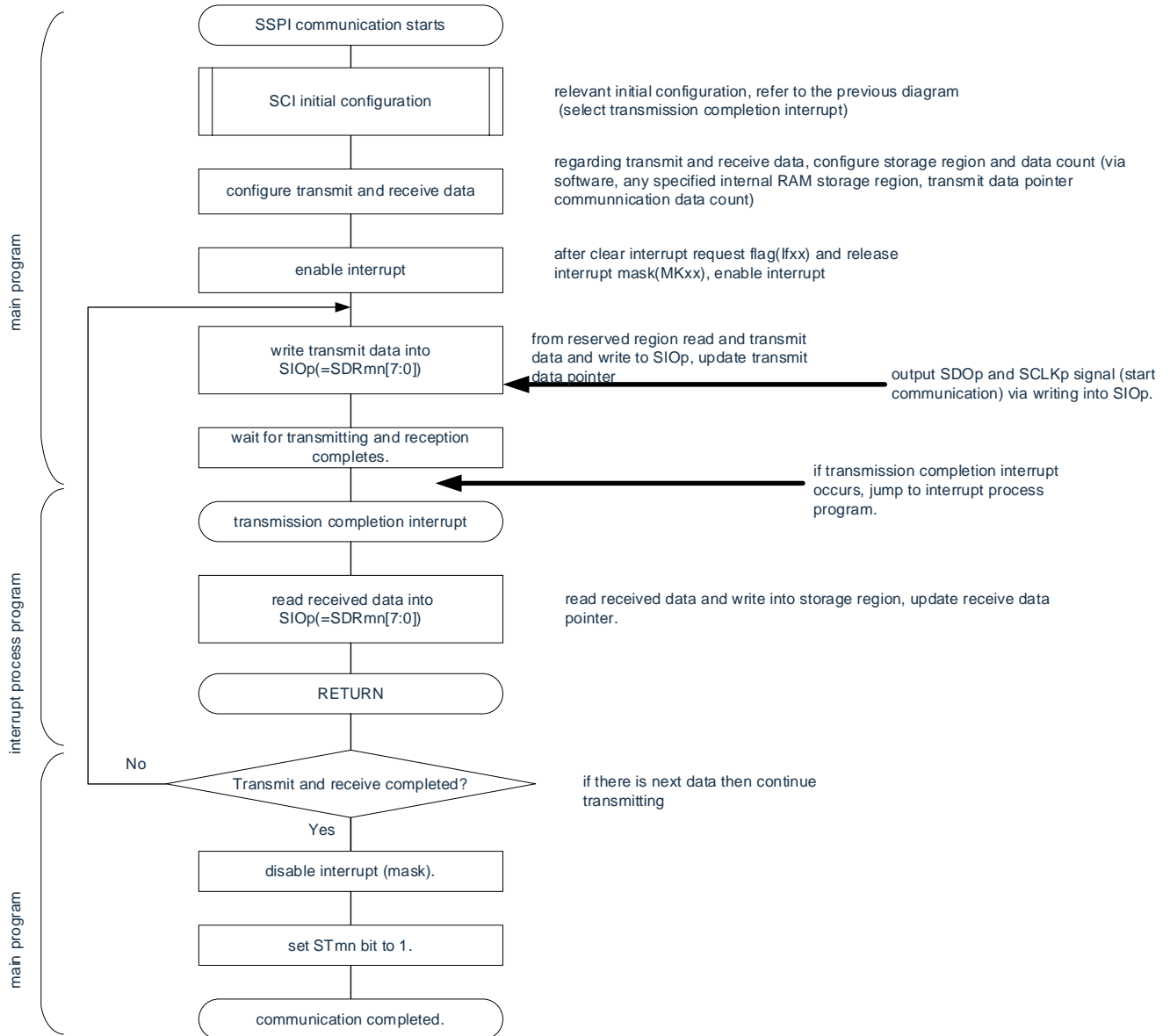
(3) Processing flow (single transmit and receive mode)

Figure14-43 Timing diagram of master transmission and reception (single transmit and receive mode)
(Type 1: DAPmn=0, CKPmn=0)



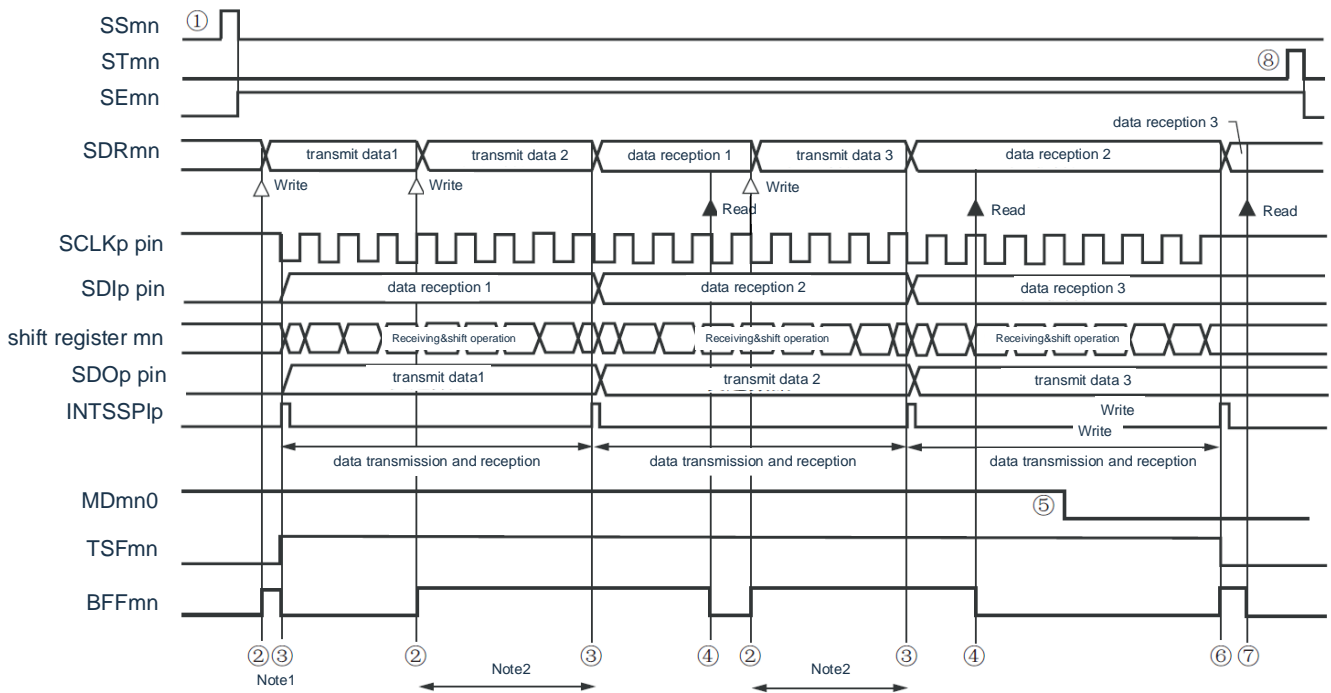
Remark m: Unit number (m=0, 1) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11.

Figure 14-44 Flowchart of master transmission and reception (single transmit and receive mode)



(4) Processing flow (continuous transmit and receive mode)

Figure 14-45 Timing diagram of master transmission and reception (continuous transmit and receive mode)
(type 1: DAPmn=0, CKPmn=0)



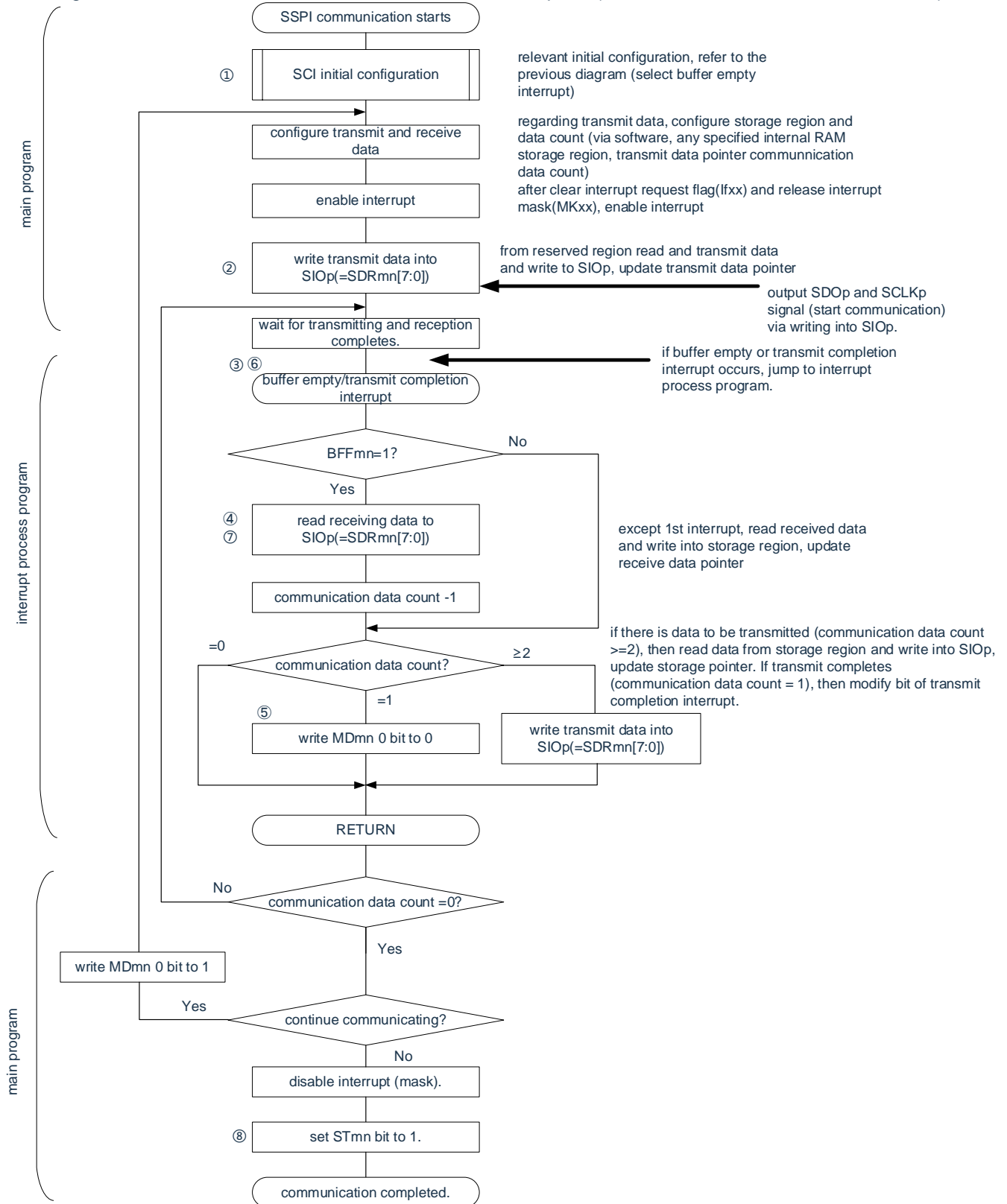
- Note 1 If the BFFmn bit of the serial status register mn (SSRmn) is “1” (valid data is saved in the serial data register mn (SDRmn) to write the send data to the SDRmn register, and rewrite the sent data.
2. If the SDRmn register is read during this period, the transmitted data can be read. At this point, the transfer run is not affected.

Notice The MDmn0 bit of the serial mode register mn (SMRmn) can be overridden even during operation. However, in order to catch up with the end of the transmission interruption of the last transmitted data, it must be overwritten before the last bit of transmission begins.

Remark 1. ① to ⑧ in the figure correspond to ① to ⑧ in the “Flowchart of Figure 14-46”.

2.m: Unit number (m=0, 1) n: Channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11

Figure 14-46 Flowchart of master transmission and reception (continuous transmit and receive mode)



Remark ① to ⑧ correspond to ① to ⑧ in “Figure 14-45 Timing diagram of master transmission and reception (continuous transmit and receive mode)”.

14.5.4 Slave transmission

Slave transmission is the operation of the BAT32G135 microcontroller to transmit data to other devices in the state of inputting the transfer clock from other devices.

3-wire serial I/O	SSPI00	SSPI01	SSPI10	SSPI11	SSPI20	SSPI21
Object channel	Channel 0 of SCI0	Channel 1 of SCI0	Channel 2 of SCI0	Channel 3 of SCI0	Channel 0 of SCI1	Channel 1 of SCI1
Used pin	SCLK00, SDO00	SCLK01, SDO01	SCLK10, SDO10	SCLK11, SDO11	SCLK20, SDO20	SCLK21, SDO21
Interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11	INTSSPI20	INTSSPI21
	Can select transmit complete interrupt (single transmit mode) or buffer empty interrupt (continuous transmit mode).					
Error detection flag	Only the overflow error detection flag (OVFmn)					
Transmitted data length	7 or 8 bits					
Transfer rate	Max. $f_{MCK}/6$ [Hz] ^{Note1, 2}					
Data phase	It can be selected via the DAPmn bit of the SCRmn register. • DAPmn=0: The data output starts when the serial clock starts running. • DAPmn=1: Starts data output half a clock before the serial clock starts running.					
Clock phase	It can be selected via the CKPmn bit of the SCRmn register. • CKPmn=0: Postive phase • CKPmn=1: Negative phase					
Data direction	MSB first or LSB first					

Note 1. The maximum transfer rate is $f_{MCK}/6$ [Hz] because the external serial clocks input from pins SCLK00, SCLK01, SCLK10, SCLK11, SCLK20, and SCLK21 are sampled internally and then used.

2. it must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Remark 1. f_{MCK} : Operation clock frequency of the object channel

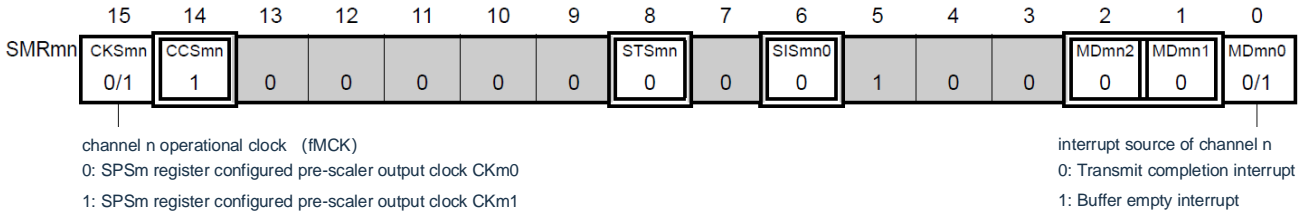
2. m: Unit number (m=0, 1) n: channel number (n=0~3) mn=00~03, 10~11.

(1) Register setting

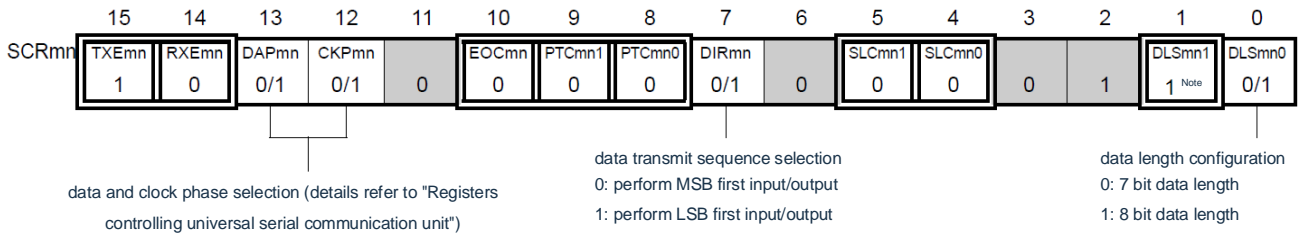
Figure 14-47 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)

Example of register settings for slave transmission

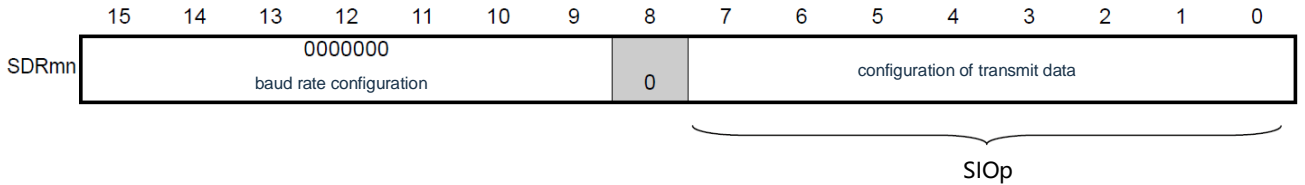
(a) serial mode register mn (SMRmn)



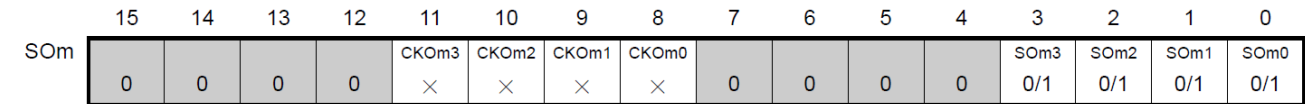
(b) serial communication operation configuration registermn mn(SCRmn)



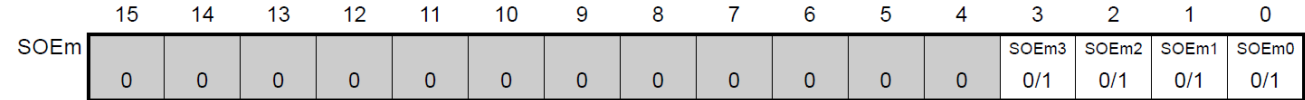
(c) serial data register mn (SDRmn) (low 8 bit: SIOp)



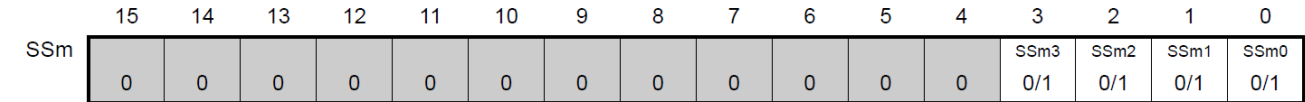
(d) serial output register m(SOm)Only configure bit of target channel



(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.



Note Limited to SCR00 register and SCR01 register, the others are fixed as "1".

Remark 1.m: Unit number (m=0, 1) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)
 mn=00~03, 10~11

- : Fixed in SSPI slave send mode. ■ : Cannot be set (initial value).
 x: This is the bit that cannot be used in this mode (set 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) Operation steps

Figure 14-48 Initial setup steps for slave transmission



Figure 14-49 Stop steps for slave transmission

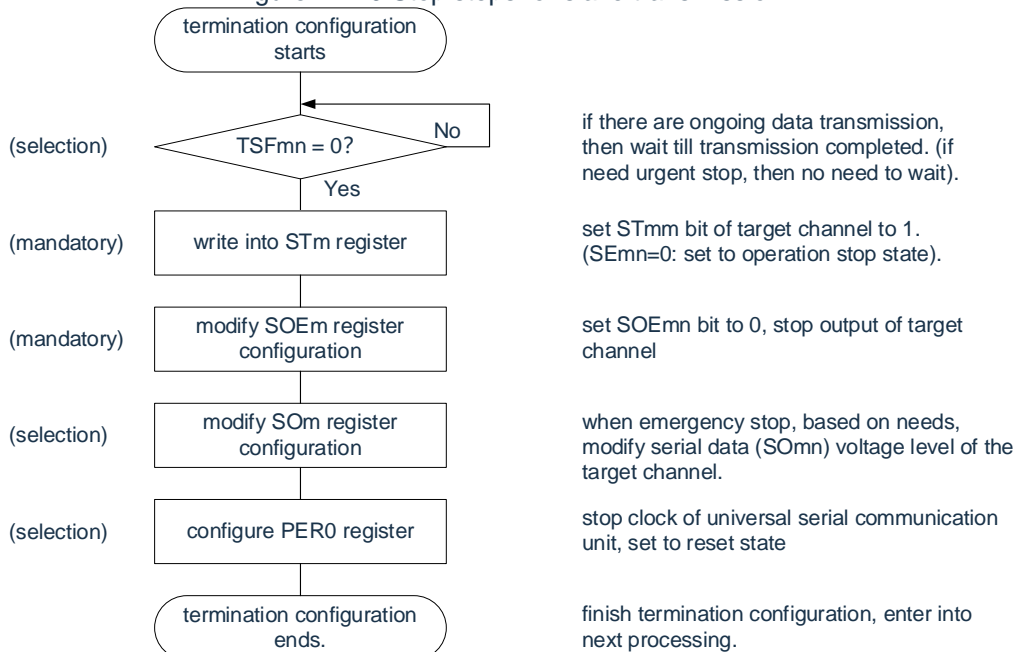
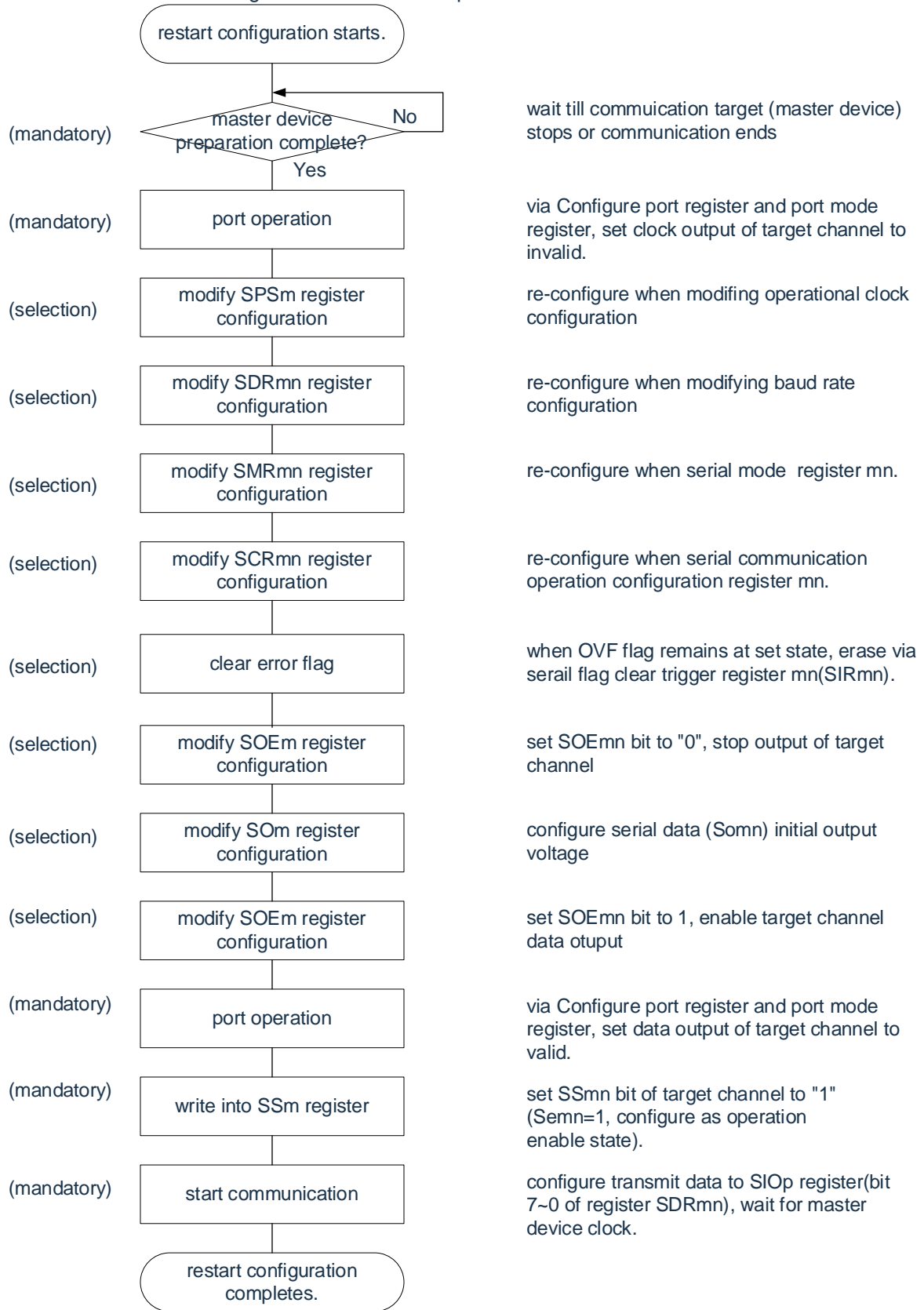


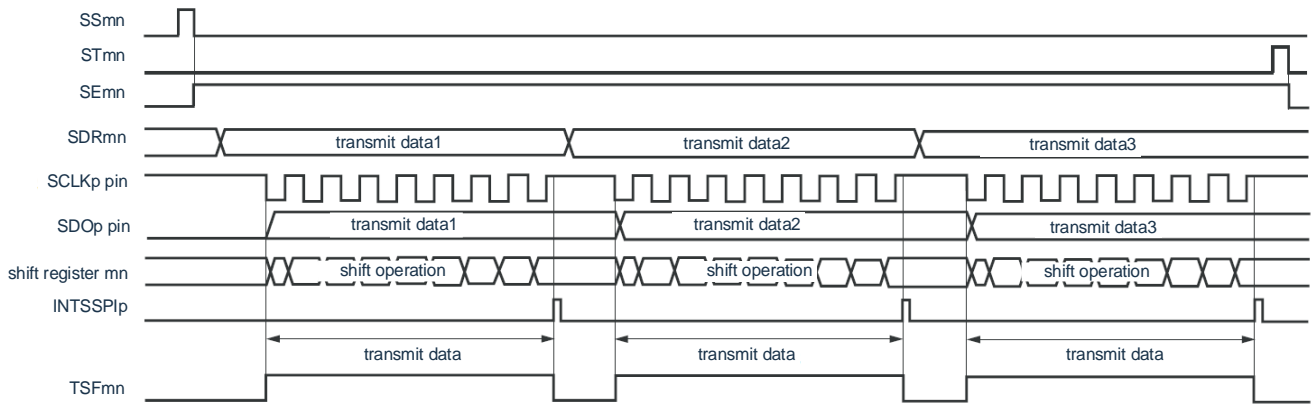
Figure 14-50 Restart steps for slave transmission



Remark If you override PER0 in the abort setting to stop the clock, you must wait until the communication object (the master device) stops or the communication is over to make the initial setting instead of the restart setting.

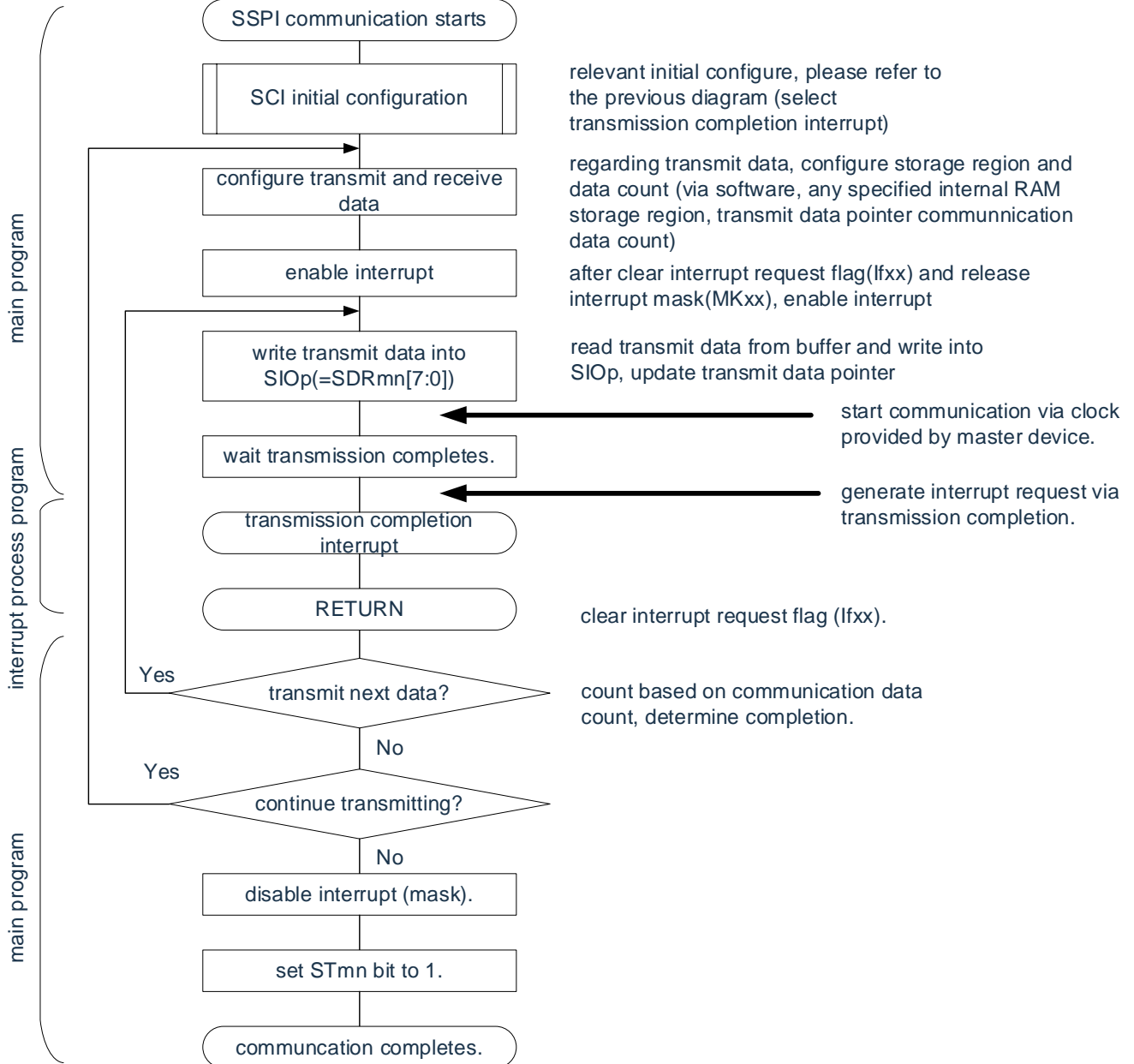
(3) Processing flow (single transmit mode)

Figure 14-51 Timing diagram of slave transmission (single transmit mode) (type 1: DAPmn=0, CKPmn=0)



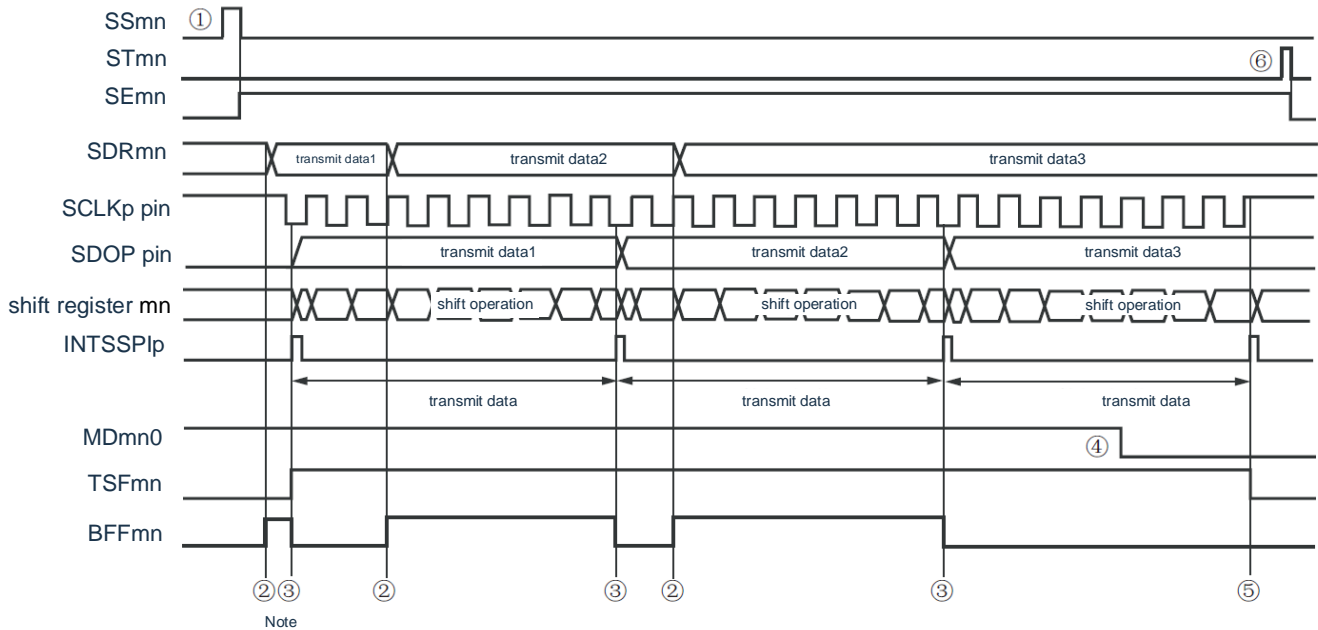
Remark m: Unit number (m=0, 1) n: Channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11

Figure 14-52 Flowchart of slave transmission (single transmit mode)



(4) Processing flow (continuous transmit mode)

Figure 14-53 Timing diagram of slave transmission (continuous transmit mode) (type 1: DAPmn=0, CKPmn=0)

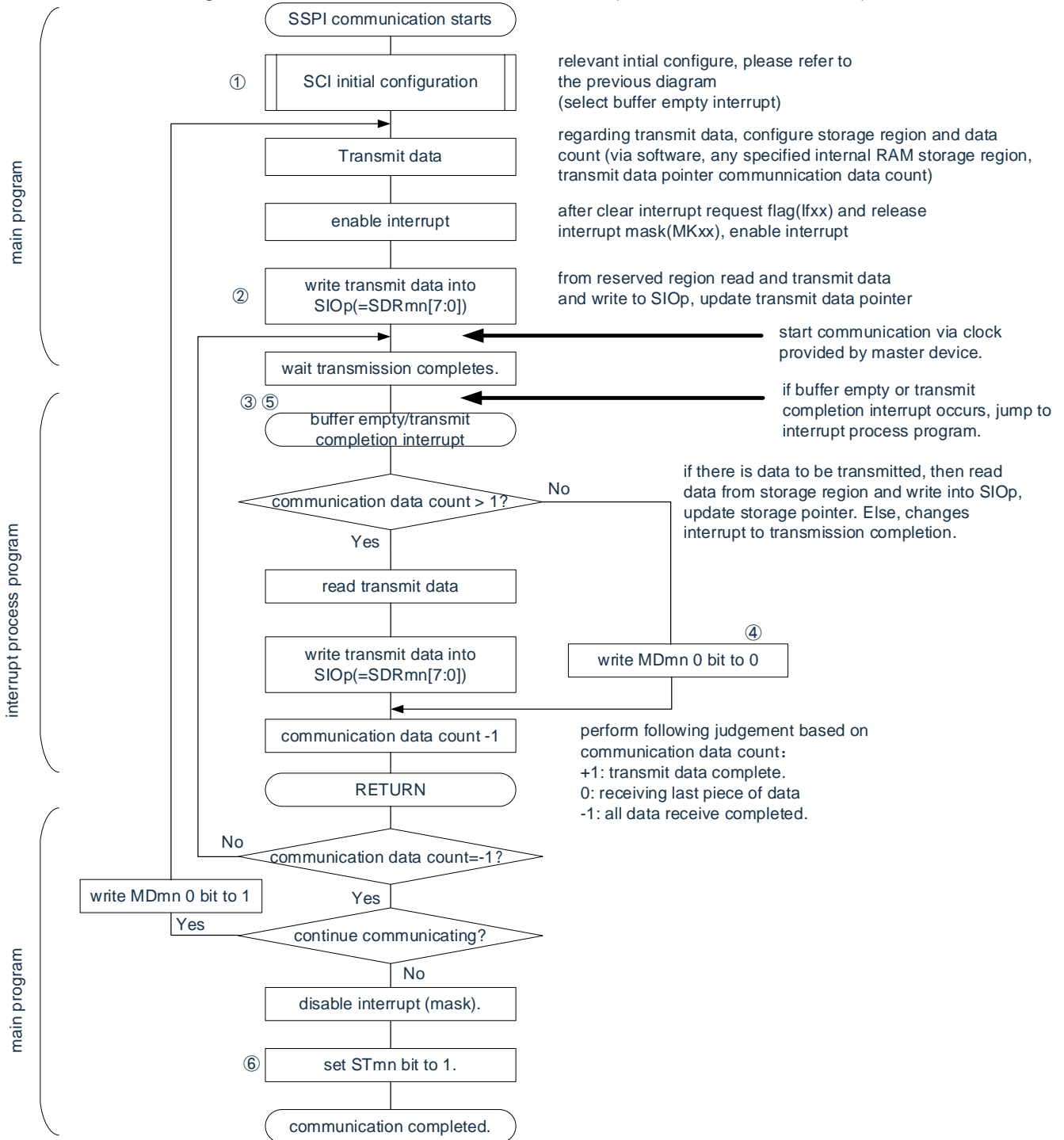


Note If the BFFmn bit of the serial status register mn (SSRmn) is “1” (when valid data is saved in the serial data register mn (SDRmn)) is given The SDRmn register writes the transmitted data and overrides the transmitted data.

Notice The MDmn0 bit of the serial mode register mn (SMRmn) can be overridden even during operation. However, it must be overridden before the last bit can be transferred.

Remark m: Unit number (m=0, 1) n: Channel number (n=0~3)p: SSPI number (p=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11.

Figure 14-54 Flowchart of slave transmission (continuous transmit mode)



Remark ① to ⑥ in the diagram correspond to ① to ⑥ in "Figure 14-53 Timing diagram of slave transmission (continuous transmit mode)".

14.5.5 Slave reception

Slave reception refers to the operation of this product to receive data from other devices in the state of transmitting clocks from other devices.

3-wire serial I/O	SSPI00	SSPI01	SSPI10	SSPI11	SSPI20	SSPI21
Object channel	SCI0 Channel 0	SCI0 Channel 1	SCI0 Channel 2	SCI0 Channel 3	SCI1 Channel 0	SCI1 Channel 1
Used pin	SCLK00, SDI00	SCLK01, SDI01	SCLK10, SDI10	SCLK11, SDI11	SCLK20, SDI20	SCLK21, SDI21
Interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11	INTSSPI20	INTSSPI21
	Limited to transmit complete interrupts (disable setting buffer empty interrupts).					
Error detection flag	Only the Overflow Error Detection Flag (OVFmn).					
Transmitted data length	7 or 8 bits					
Transfer rate	Max. $f_{MCK}/6$ [Hz] ^{Note 1, 2}					
Data phase	It can be selected via the DAPmn bit of the SCRmn register. <ul style="list-style-type: none"> • DAPmn=0: The data output starts when the serial clock starts running. • DAPmn=1: Starts data output half a clock before the serial clock starts running. 					
Clock phase	It can be selected via the CKPmn bit of the SCRmn register. <ul style="list-style-type: none"> • CKPmn=0: Positive phase • CKPmn=1: Negative phase 					
Data direction	MSB first or LSB first					

Note 1. The maximum transfer rate is $f_{MCK}/6$ [Hz] because the external serial clocks input from pins SCLK00, SCLK01, SCLK10, SCLK11, SCLK20, and SCLK21 are sampled internally and then used.

2. It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Remark 1. f_{MCK} : Operation clock frequency of the object channel

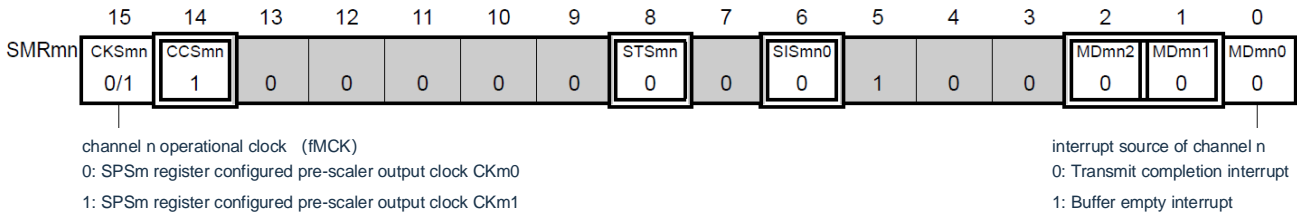
2. m: Unit number (m=0, 1) n: channel number (n=0~3) mn=00~03, 10~11.

(1) Register setting

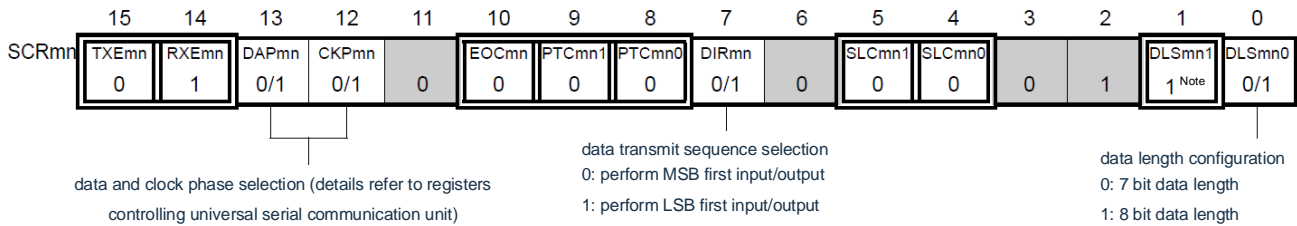
Figure 14-55 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)

Example of register settings for slave reception

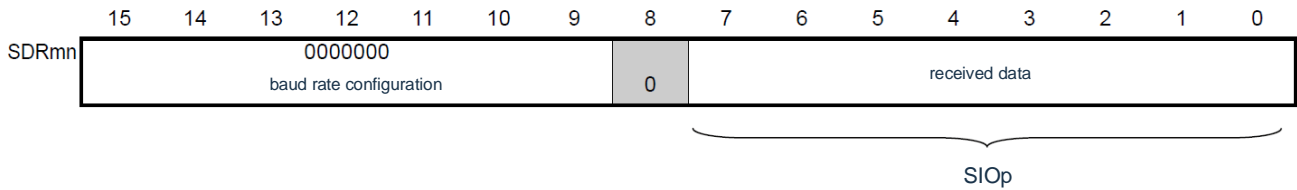
(a) serial mode register mn (SMRmn)



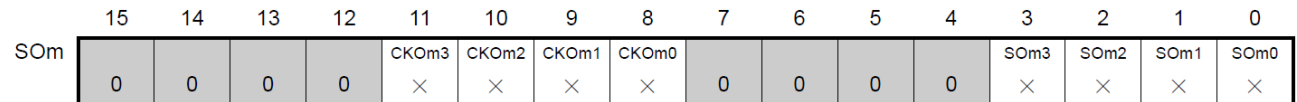
(b) serial communication operation configuration registermn mn(SCRmn)



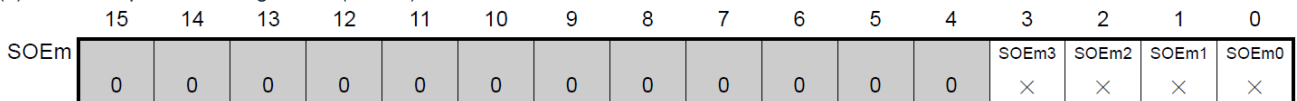
(c) serial data register mn (SDRmn) (low 8 bit: SIOp)



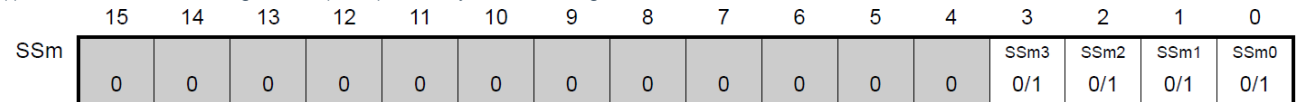
(d) serial output register m (Som) Not used in this mode.



(e) serial output enable register m (SOEm).... Not used in this mode.



(f) serial channel start register m (SSm) Only set bit of target channel to "1".



Note Limited to SCR00 register and SCR01 register, the others are fixed as "1".

Remark 1.m: Unit number (m=0, 1) n: Channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)
 mn=00~03, 10~11

2. : Fixed in Slave Receive mode. : Cannot be set (initial value).
 x: This is the bit that cannot be used in this mode (set 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) Operation steps

Figure 14-56 Initial setup steps for slave reception

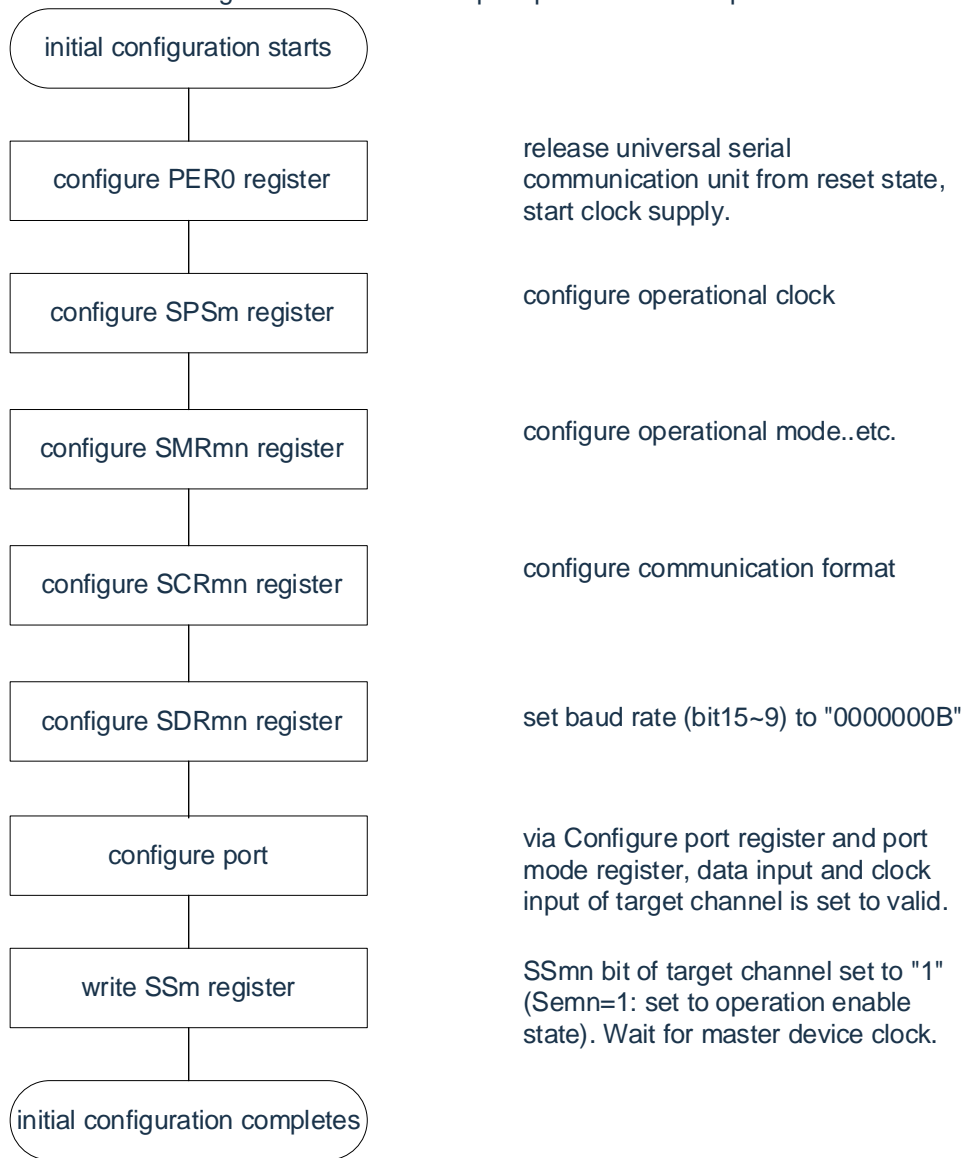


Figure 14-57 Stop steps for slave reception

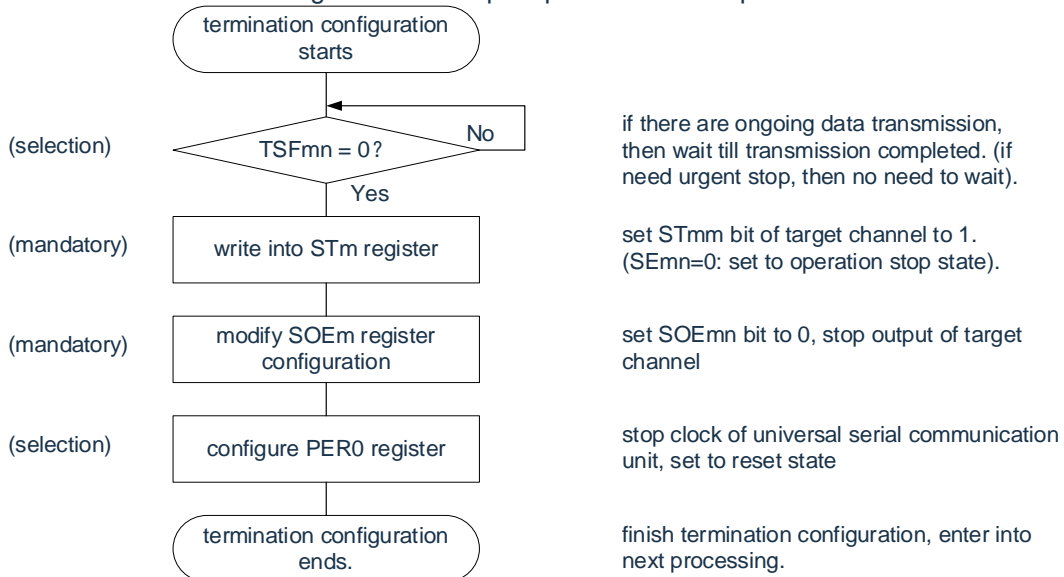
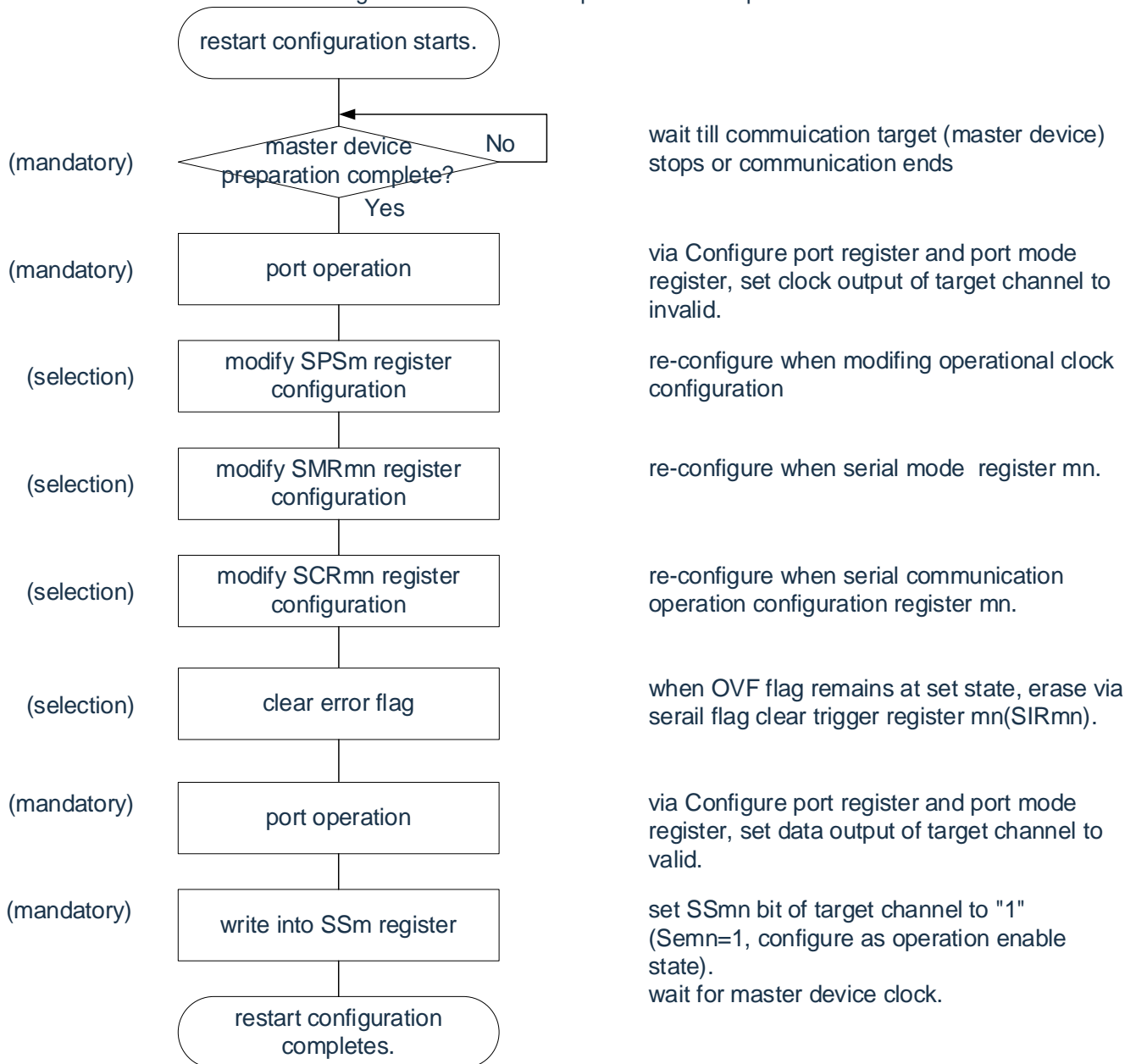


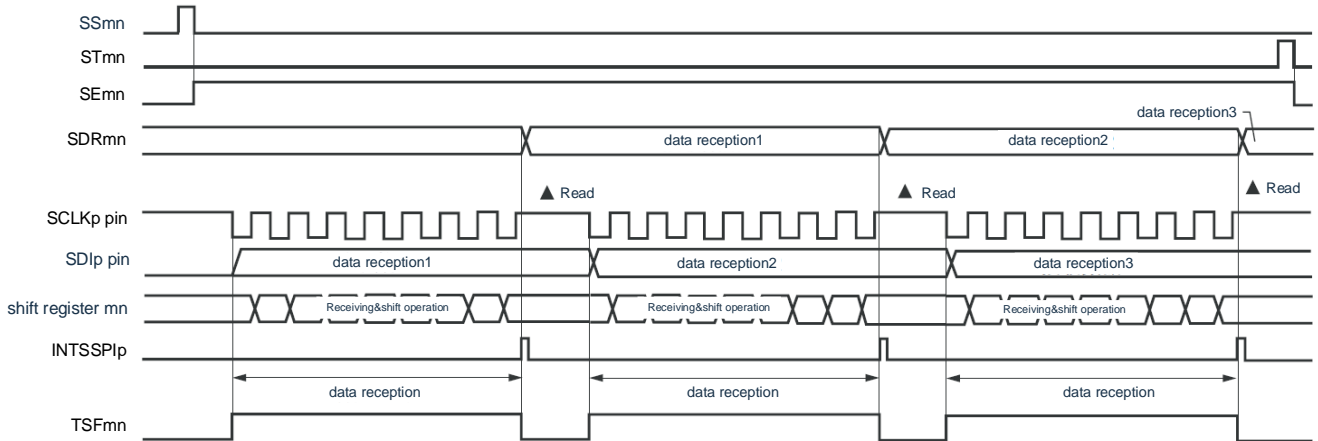
Figure 14-58 Restart steps for slave reception



Remark If you override PER0 in the abort setting to stop the clock, you must wait until the communication object (the master device) stops or the communication is over to make the initial setting instead of the restart setting.

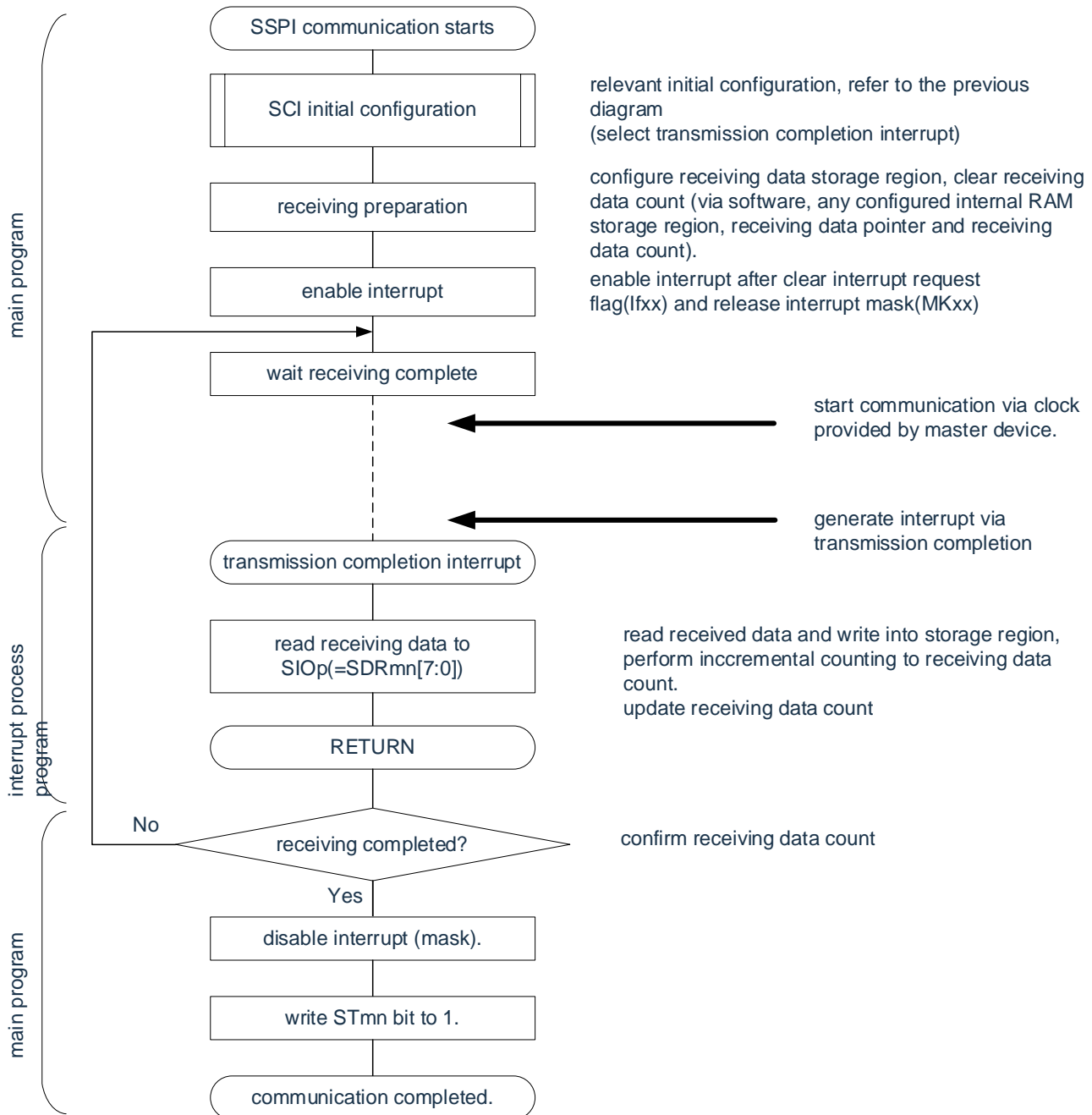
(3) Processing flow (single receive mode)

Figure 14-59 Timing diagram of slave reception (single receive mode) (type 1: DAPmn=0, CKPmn=0)



Remark m: Unit number (m=0, 1) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)
mn=00~03, 10~11.

Figure 14-60 Flowchart of slave reception (single receive mode)



14.5.6 Slave transmission and reception

Slave transmission and reception refers to the operation of the microcontroller and other devices of this product to transmit and receive data in the state of transmitting clocks from other devices.

3-wire serial I/O	SSPI00	SSPI01	SSPI10	SSPI11	SSPI20	SSPI21
Object channel	SCI0 Channel 0	SCI0 Channel 1	SCI0 Channel 2	SCI0 Channel 3	SCI1 Channel 0	SCI1 Channel 1
Used pin	SCLK00, SDI00, SDO00	SCLK01, SDI01, SDO01	SCLK10, SDI10, SDO10	SCLK11, SDI11, SDO11	SCLK20, SDI20, SDO20	SCLK21, SDI21, SDO21
Interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11	INTSSPI20	INTSSPI21
	Can select transmit complete interrupt (single transmit mode) or buffer empty interrupt (continuous transmit mode).					
Error detection flag	Only the Overflow Error Detection Flag (OVFmn).					
Transmitted data length	7 or 8 bits					
Transfer rate	Max. $f_{MCK}/6$ [Hz] Note ^{1, 2}					
Data phase	It can be selected via the DAPmn bit of the SCRmn register. • DAPmn=0: Data input/output starts 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 via the CKPmn bit of the SCRmn register. • CKPmn=0: Positive phase • CKPmn=1: Negative phase					
Data direction	MSB first or LSB first					

Note 1. The maximum transfer rate $f_{MCK}/6$ [Hz] is used because the external serial clock input from SCLK00, SCLK01, SCLK10, SCLK11, SCLK20, and SCLK21 pins is sampled internally and then used.

2. It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Remark 1. f_{MCK} : Operation clock frequency of the object channel

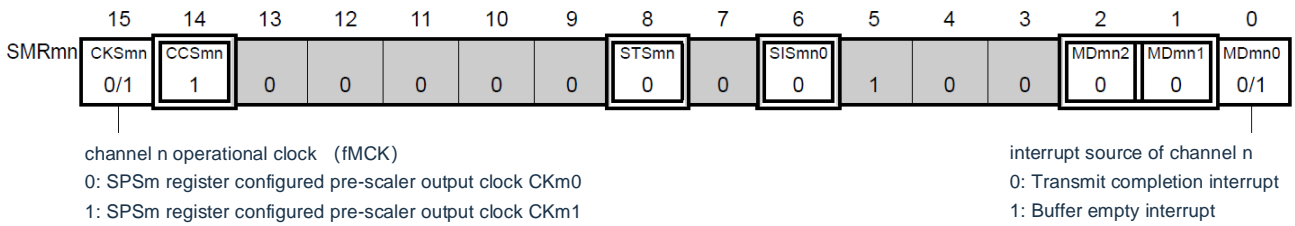
2.m: Unit number (m=0, 1) n: channel number (n=0~3) mn=00~03, 10~11

(1) Register setting

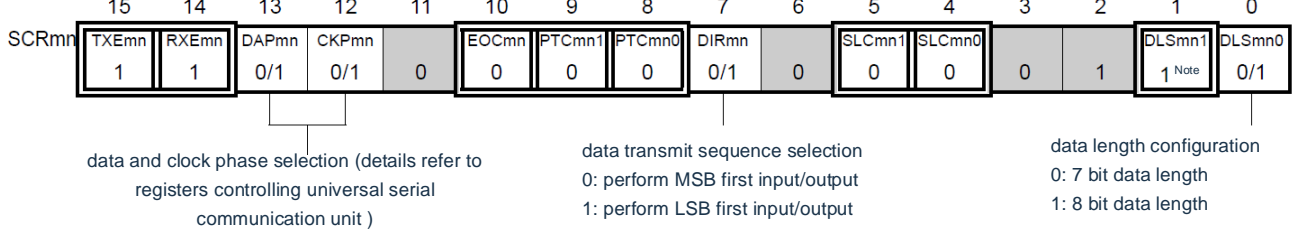
Figure 14-61 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)

Example of register settings for slave transmission and reception

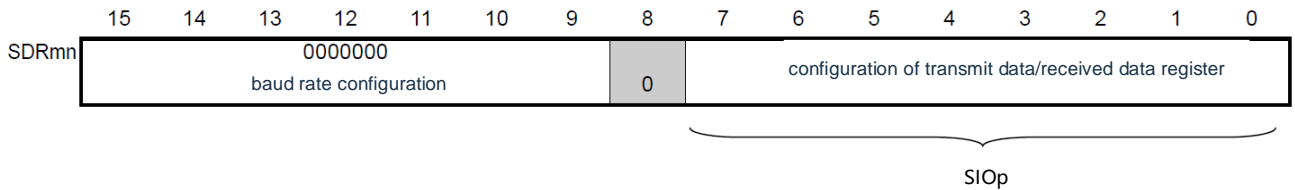
(a) serial mode register mn (SMRmn)



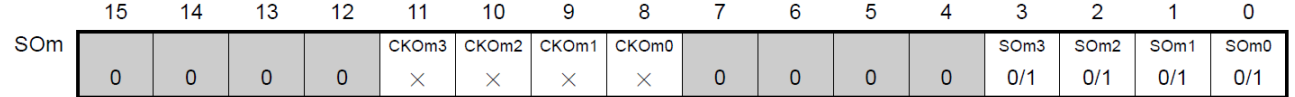
(b) serial communication operation configuration register mn (SCRmn)



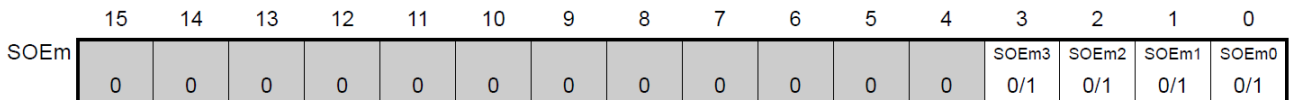
(c) serial data register mn (SDRmn) (low 8 bit: SIOp)



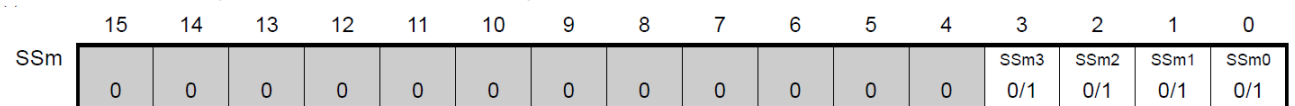
(d) serial output register m(SOm)Only configure bit of target channel



(e) serial output enable register m(SOEm)....only set bit of target channel to 1.



(f) serial channel start register m(SSm) Only set bit of target channel to 1.



Note Limited to SCR00 register and SCR01 register, the others are fixed as "1".

Notice Data must be sent to the SIOp register settings before the master device starts the output clock.

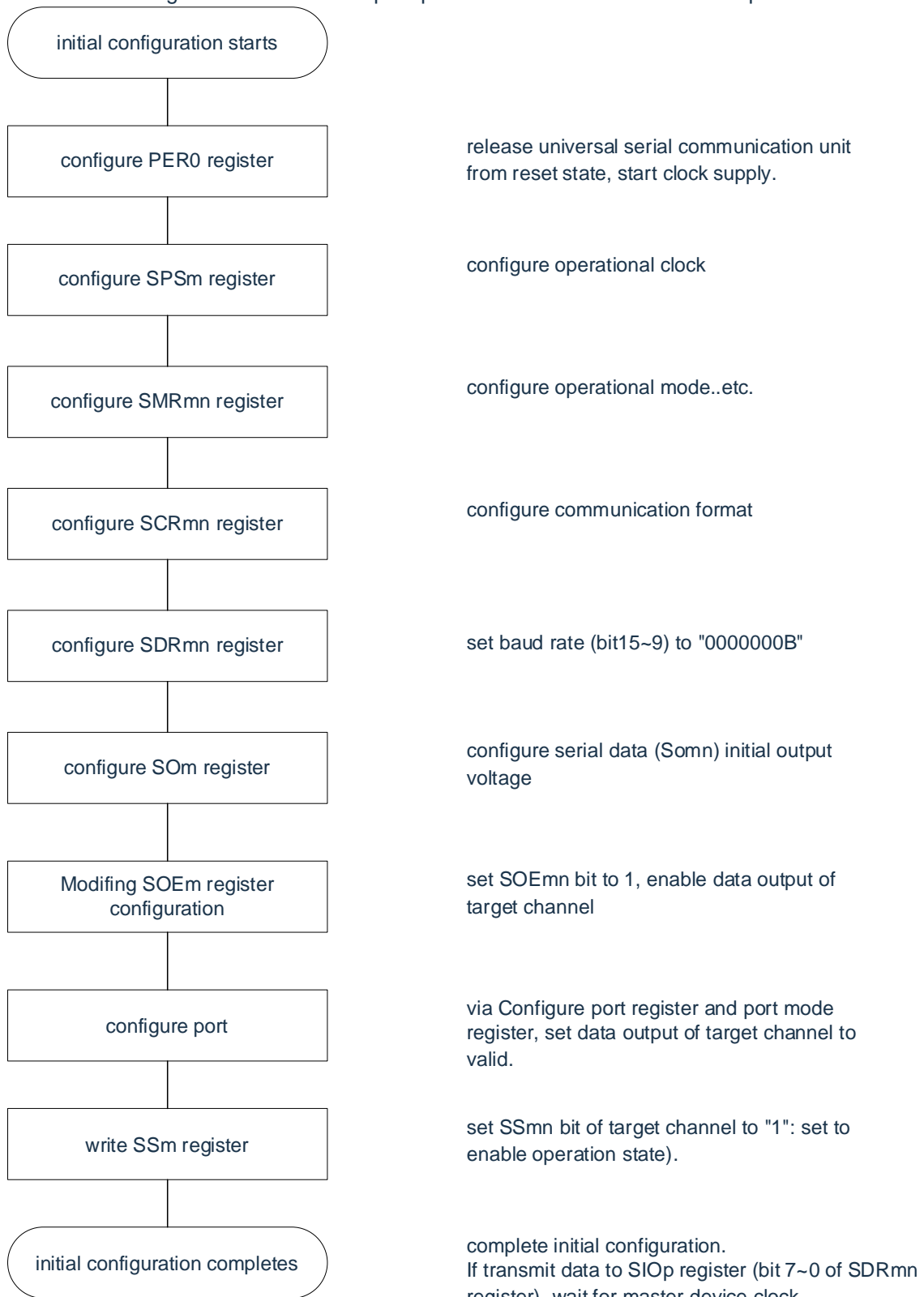
Remark 1. m: Unit number (m=0, 1) n: Channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)

mn=00~03, 10~11

2. : Fixed in SSPI slave T&R mode. Cannot be set (set initial value)
 x: This is the bit that cannot be used in this mode (set 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) Operation steps

Figure 14-62 Initial setup steps for slave transmission and reception



Notice Before the master device starts to output the clock, the SIOp register must be set to transmit data.

Figure 14-63 Stop steps for slave transmission and reception

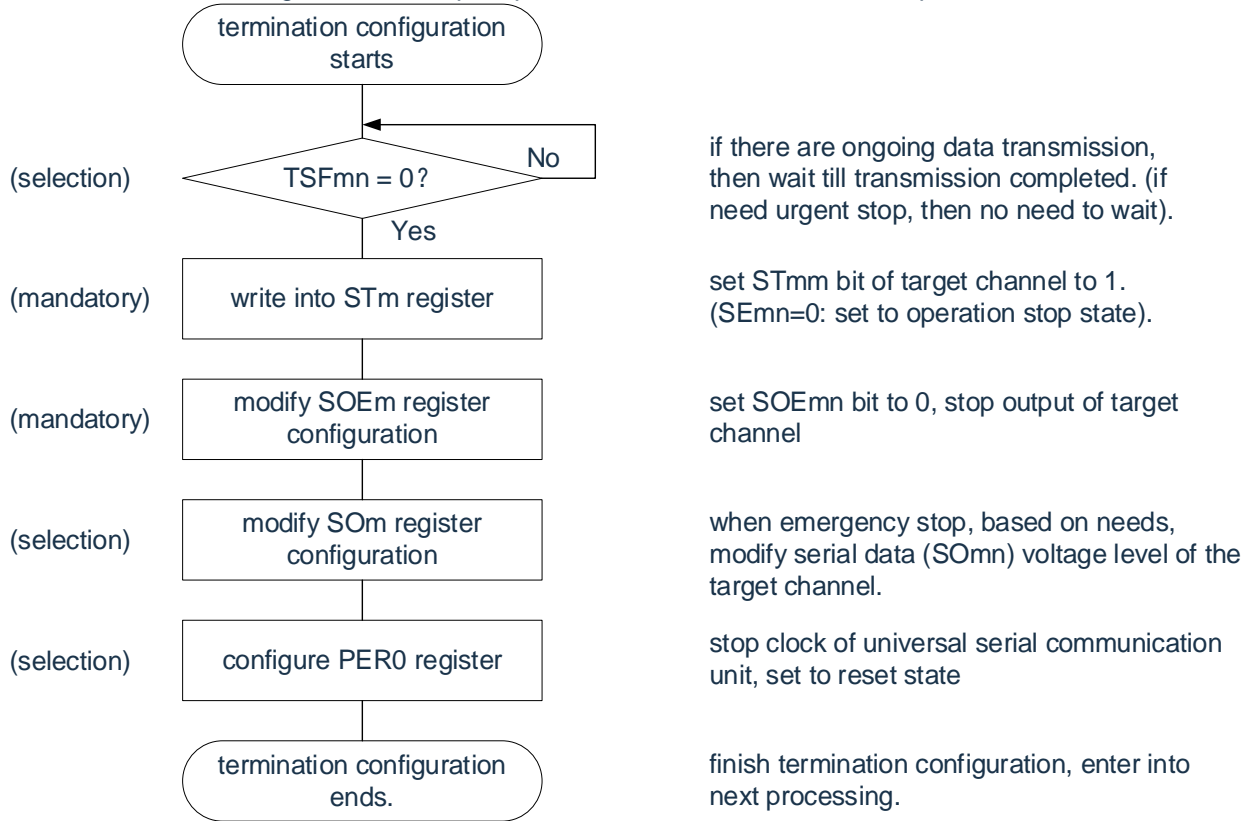
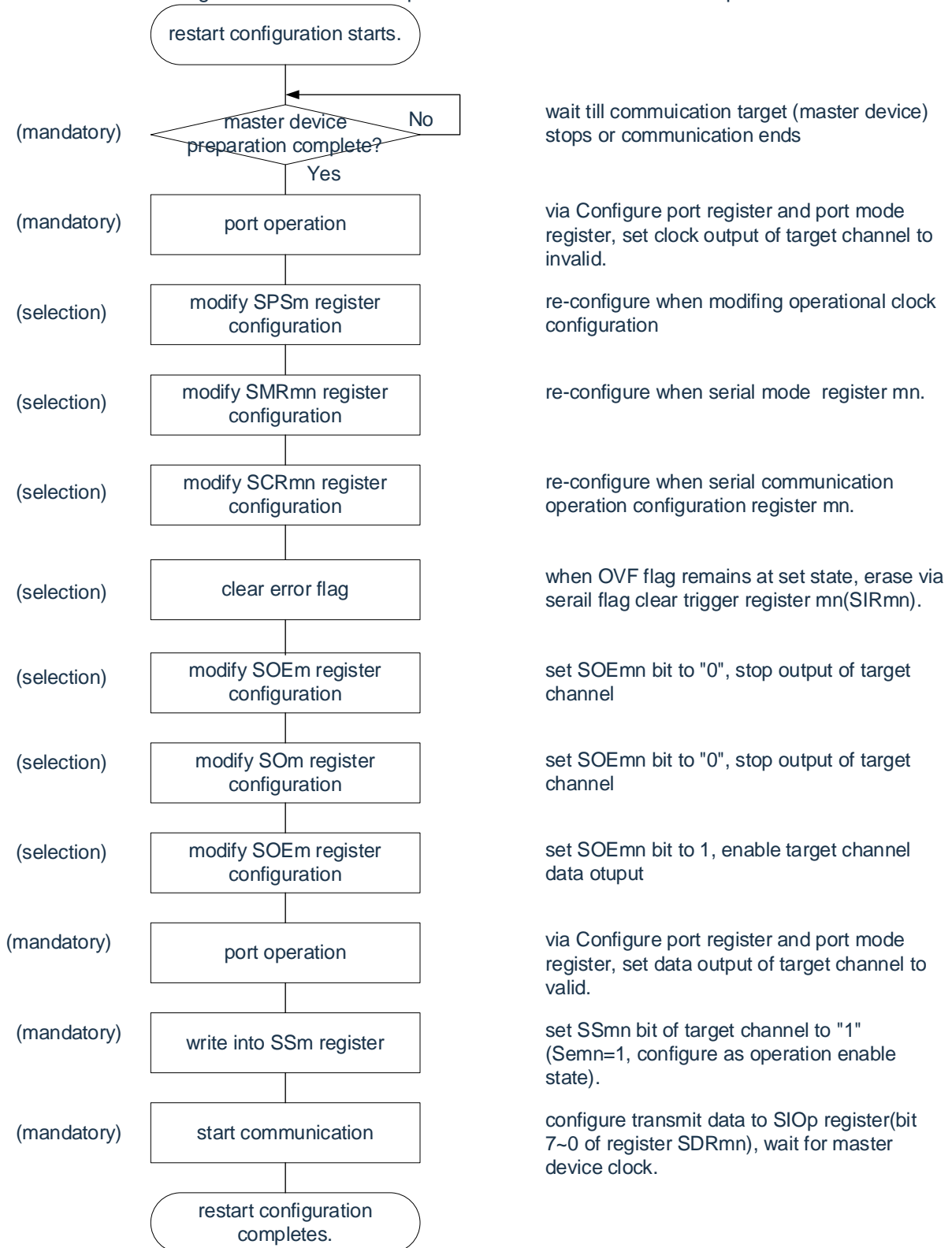


Figure 14-64 Restart steps for slave transmission and reception

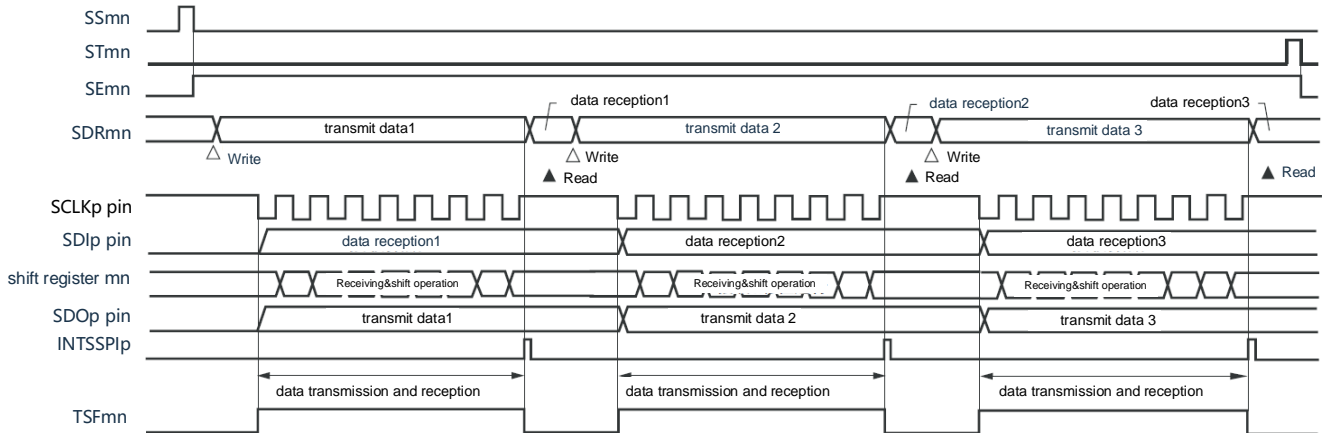


Notice 1. Before the master device starts to output the clock, data must be sent to the SIOp register settings.

2. If you override PER0 in the abort setting to stop the clock, you must wait until the communication object (the master device) stops or the communication is over to make the initial setting instead of the restart setting.

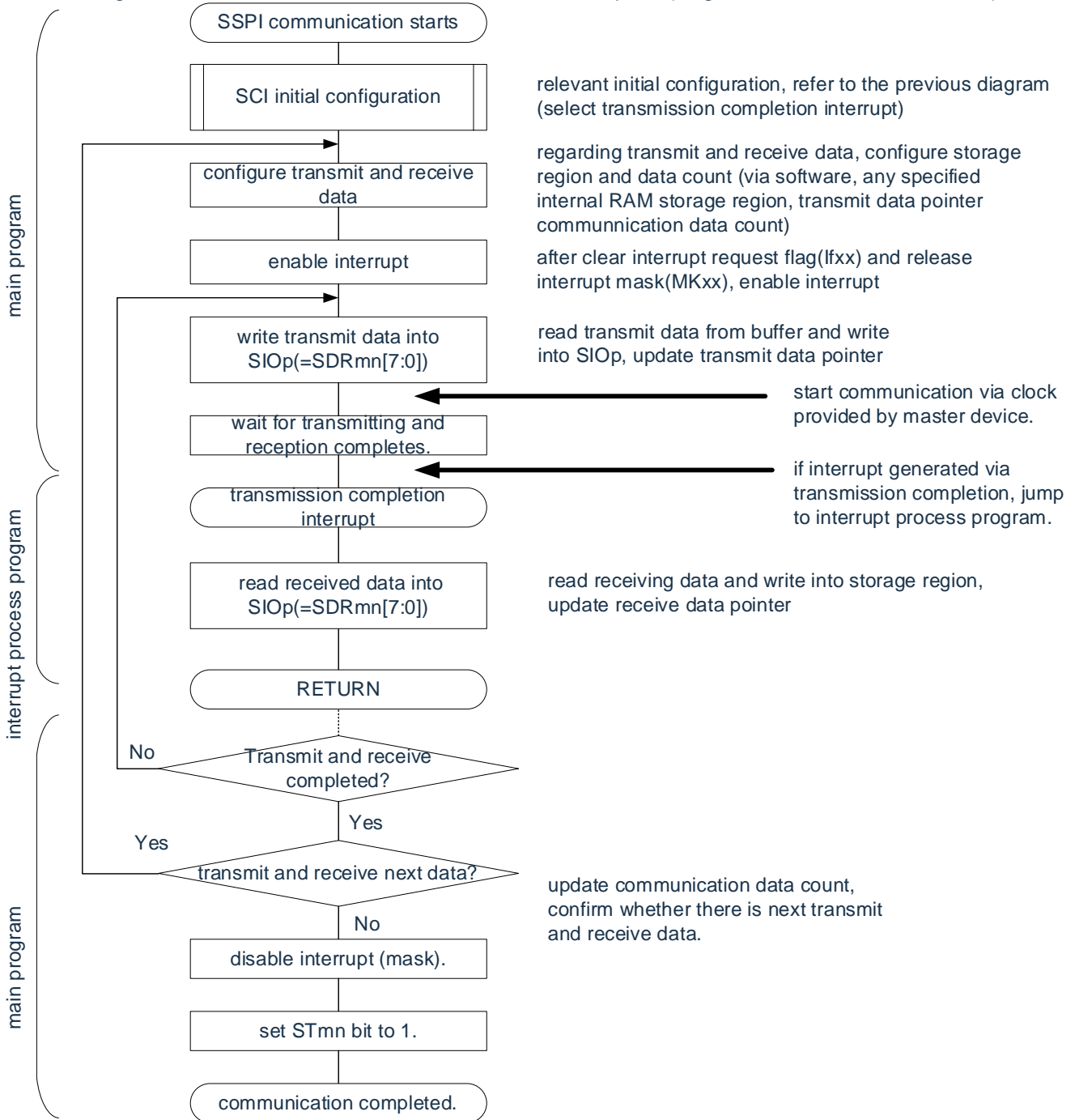
(3) Processing flow (single transmit and receive mode)

Figure 14-65 Timing diagram of slave transmission and reception (single transmit and receive mode)
(type 1: DAPmn=0, CKPmn=0)



Remark m: Unit number (m=0, 1) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11

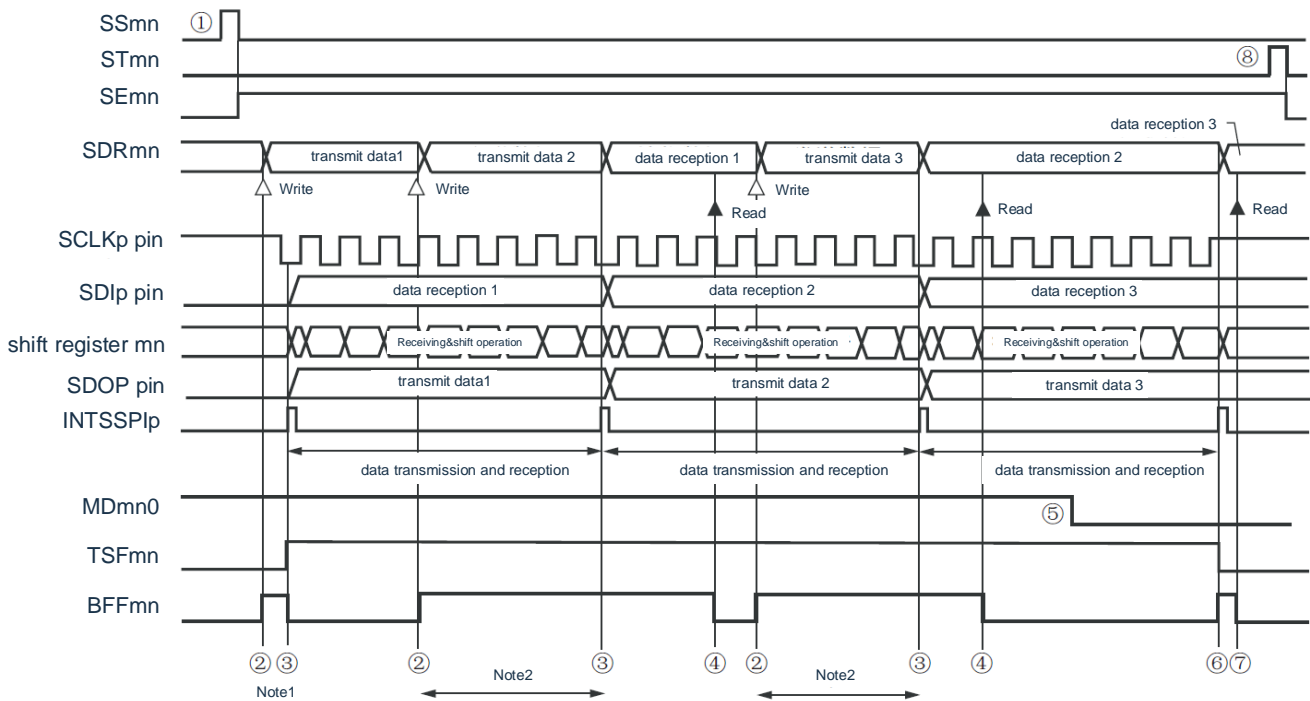
Figure 14-66 Flowchart of slave transmission and reception (single transmit and receive mode)



Notice Data must be sent to the SIOp register settings before the master device starts the output clock.

(4) Processing flow (continuous send and receive mode)

Figure 14-67 Timing diagram of slave transmission and reception (continuous transmit and receive mode)
(type 1: DAPmn=0, CKPmn=0)



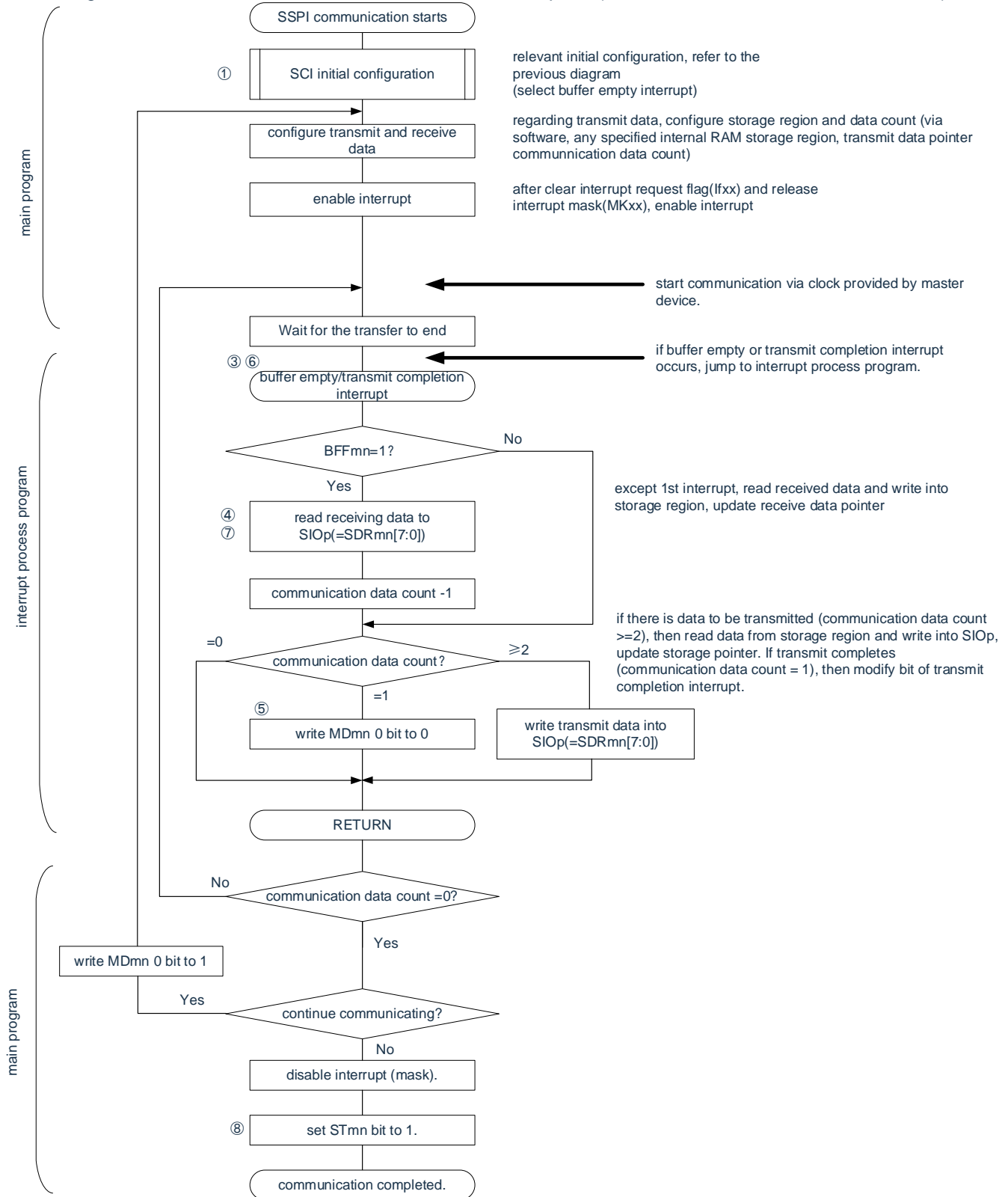
Note 1 If the BFFmn bit of the serial status register mn (SSRmn) is “1” (valid data is saved in the serial data register mn (SDRmn) to write the send data to the SDRmn memory, and override the sent data.
2. If the SDRmn register is read during this period, the transmitted data can be read. At this point, the transfer run is not affected.

Notice The MDmn0 bit of the serial mode register mn (SMRmn) can be overridden even during operation. However, in order to catch up with the end of the transmission interruption of the last transmitted data, it must be overwritten before the last bit of transmission begins.

Remark 1.① to ⑧ in the diagram correspond to ① to ⑧ in “Figure 14-68 Flowchart of slave transmission and reception (continuous transmit and receive mode)”.

2. m: Unit number (m=0, 1) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11

Figure 14-68 Flowchart of slave transmission and reception (continuous transmit and receive mode)



Notice Data must be sent to the SIOp register settings before the master device starts the output clock.

Remark ① to ⑧ correspond to ① to ⑧ in “Figure 14-67 Timing diagram of slave transmission and reception (continuous transmit and receive mode)”.

14.5.7 Calculation of transfer clock frequency

3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21) communication transmission clock frequency can be calculated using the following calculation equations.

(1) Master device

$$(\text{transfer clock frequency}) = \{ \text{Operation clock frequency of the object channel}(f_{MCK}) \} \div (\text{SDRmn}[15:9]+1) \div 2 [\text{Hz}]$$

(2) Slave device

$$(\text{transfer clock frequency}) = \{ \text{serial clock (SCLK) frequency provided by the master device} \} \text{ Note } [\text{Hz}].$$

Note The maximum enable transmit clock frequency is $f_{MCK}/6$.

Remark Since the value of SDRmn[15:9] is the value of bit15 to 9 (0000000B to 1111111B) of serial data register mn (SDRmn), it is 0 to 127.

The operation clock(F_{MCK}) depends on bit15 (CKSmn) of the serial clock selection register m (SPSm) and the serial mode register mn (SMRmn).

Table 14-2 Selection of 3-wire 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		$f_{CLK}=32\text{MHz}$ in operation
0	X	X	X	X	0	0	0	0	0	f_{CLK}	32MHz
	X	X	X	X	0	0	0	1	1	$f_{CLK}/2$	16MHz
	X	X	X	X	0	0	1	0	0	$f_{CLK}/2^2$	8MHz
	X	X	X	X	0	0	1	1	1	$f_{CLK}/2^3$	4MHz
	X	X	X	X	0	1	0	0	0	$f_{CLK}/2^4$	2MHz
	X	X	X	X	0	1	0	1	1	$f_{CLK}/2^5$	1MHz
	X	X	X	X	0	1	1	0	0	$f_{CLK}/2^6$	500kHz
	X	X	X	X	0	1	1	1	1	$f_{CLK}/2^7$	250kHz
	X	X	X	X	1	0	0	0	0	$f_{CLK}/2^8$	125kHz
	X	X	X	X	1	0	0	1	1	$f_{CLK}/2^9$	62.5kHz
	X	X	X	X	1	0	1	0	0	$f_{CLK}/2^{10}$	31.25kHz
	X	X	X	X	1	0	1	1	1	$f_{CLK}/2^{11}$	15.63kHz
	X	X	X	X	1	1	0	0	0	$f_{CLK}/2^{12}$	7.81kHz
	X	X	X	X	1	1	0	1	1	$f_{CLK}/2^{13}$	3.91kHz
	X	X	X	X	1	1	1	0	0	$f_{CLK}/2^{14}$	1.95kHz
X	X	X	X	1	1	1	1	1	$f_{CLK}/2^{15}$	977Hz	
1	0	0	0	0	X	X	X	X	X	f_{CLK}	32MHz
	0	0	0	1	X	X	X	X	X	$f_{CLK}/2$	16MHz
	0	0	1	0	X	X	X	X	X	$f_{CLK}/2^2$	8MHz
	0	0	1	1	X	X	X	X	X	$f_{CLK}/2^3$	4MHz
	0	1	0	0	X	X	X	X	X	$f_{CLK}/2^4$	2MHz
	0	1	0	1	X	X	X	X	X	$f_{CLK}/2^5$	1MHz
	0	1	1	0	X	X	X	X	X	$f_{CLK}/2^6$	500kHz
	0	1	1	1	X	X	X	X	X	$f_{CLK}/2^7$	250kHz
	1	0	0	0	X	X	X	X	X	$f_{CLK}/2^8$	125kHz
	1	0	0	1	X	X	X	X	X	$f_{CLK}/2^9$	62.5kHz
	1	0	1	0	X	X	X	X	X	$f_{CLK}/2^{10}$	31.25kHz
	1	0	1	1	X	X	X	X	X	$f_{CLK}/2^{11}$	15.63kHz
	1	1	0	0	X	X	X	X	X	$f_{CLK}/2^{12}$	7.81kHz
	1	1	0	1	X	X	X	X	X	$f_{CLK}/2^{13}$	3.91kHz
	1	1	1	0	X	X	X	X	X	$f_{CLK}/2^{14}$	1.95kHz
1	1	1	1	X	X	X	X	X	$f_{CLK}/2^{15}$	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 universal serial communication unit (SCI) (serial channel stop register m (STM) = 000FH).

Remark 1. X: Ignore

2. m: Unit number (m=0, 1) n: channel number (n=0~3) mn=00~03, 10~11.

14.5.8 Procedure for handling errors during 3-wire serial I/O communication (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)

In 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21), the processing steps when an error occurs during communication are shown in Figure 14-69.

Figure 14-69 Steps to handle when an overflow error occurs

Software operation	Hardware status	Remark
Read the serial data register →	The BFF _{mn} bit of the SSR _{mn} register is "0" and the channel n is in acceptable.	This is to prevent overflow errors from ending the next reception during mishandling.
Read the serial status register mn (SSR _{mn})		The type of error is judged, and the reading value is used to clear the error flag.
Clear trigger register mn to the serial flag →	Clear the error flag.	By writing the read value of the SSR _{mn} register directly to the SDIR _{mn} register, errors during read operations can only be cleared.

Remark m: Unit number (m=0, 1) n: channel number (n=0~3) mn=00~03, 10~11.

14.6 Operation of clock-synchronous serial communication with slave selection input function

Channel 0 of SCI0 is the channel that supports clock-synchronous serial communication with the slave select input function.

[Transmit and receive data]

- 7-bit or 8-bit data length
- Phase control of sending and receiving data
- MSB/LSB preferred choice
- Level settings for sending and receiving data

[Clock control].

- Phase control of input/output clocks
- Sets the transfer period generated by the prescaler and the in-channel counter.
- Maximum transfer rate ^{Note} for slave communication: $\text{Max.f}_{\text{MCK}}/6$

[Interrupt function].

- End of transfer interrupt, buffer empty interrupt

[Error detection flag].

- Overflow error

Note it must be used within the scope of the SCLK Cycle Time (t_{KCY}) characteristics. Please refer to the data sheet for details.

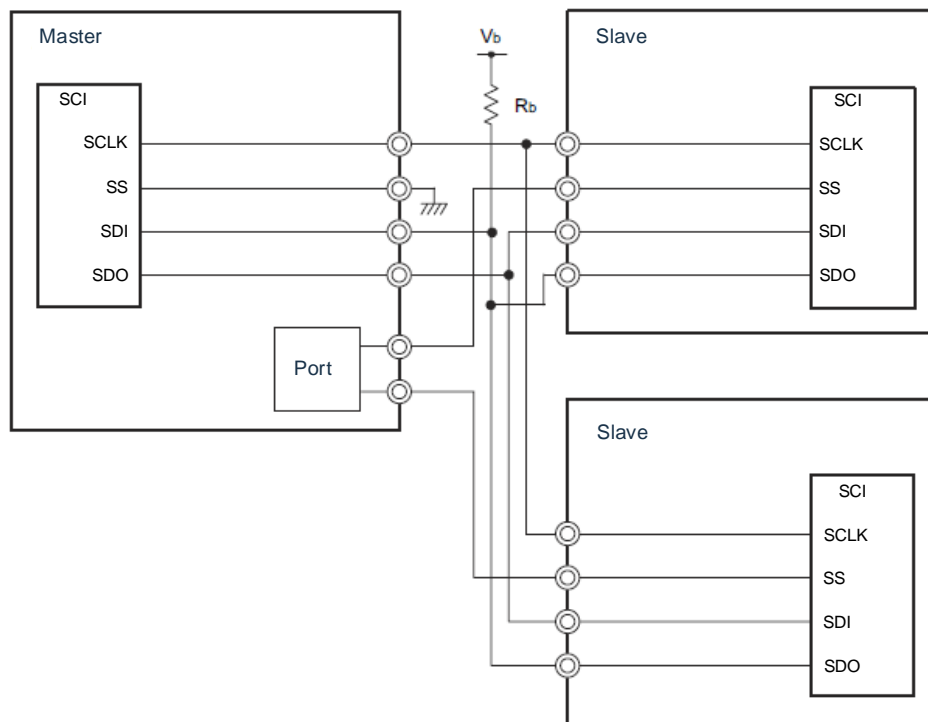
The slave select input function operates in three types of communication:

- Slave transmission (see 14.6.1).
- Slave reception (see 14.6.2).
- Slave transmission and reception (see 14.6.3).

By using the slave select input function, one master device can be connected to multiple slave devices for communication. The master device outputs a slave selection signal to the slave device (1) of the communication object, and each slave device determines whether it is selected as a communication object and controls the output of the SDO pin. When selected as a slave device for communication object, the SDO pin can communicate with the master device to send data; When a slave device is not selected as a communication object, the SDO pin becomes a high-level output, so in the environment where multiple slaves are connected, the SDO pin needs to be set to Nch-O.D and the node pulled up. In addition, even the serial clock of the input master device is not transmitted and received.

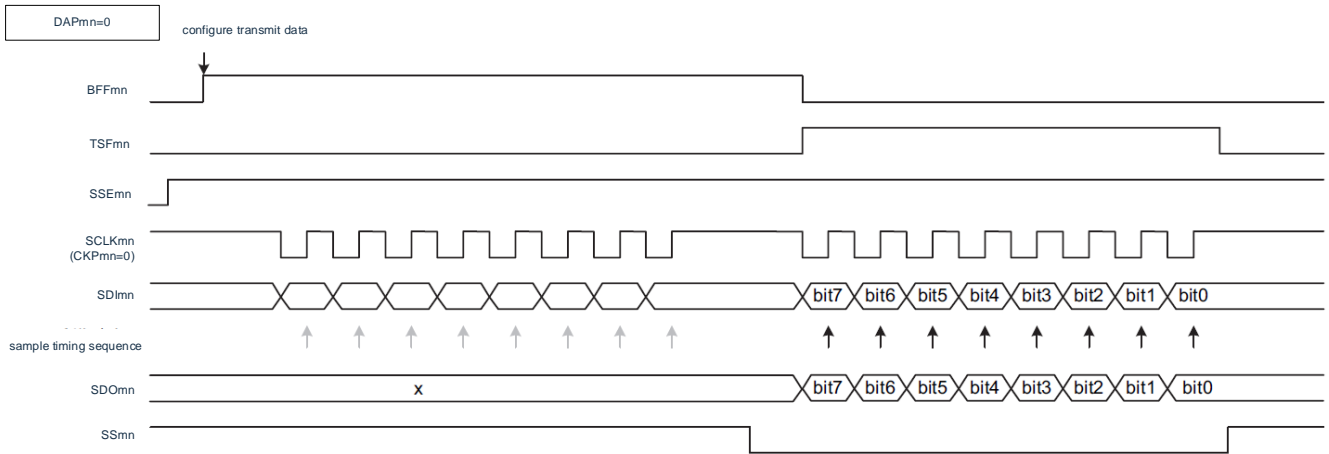
Notice The slave select signal must be output through the operation of the port.

Figure 14-70 Structural example of slave select input function



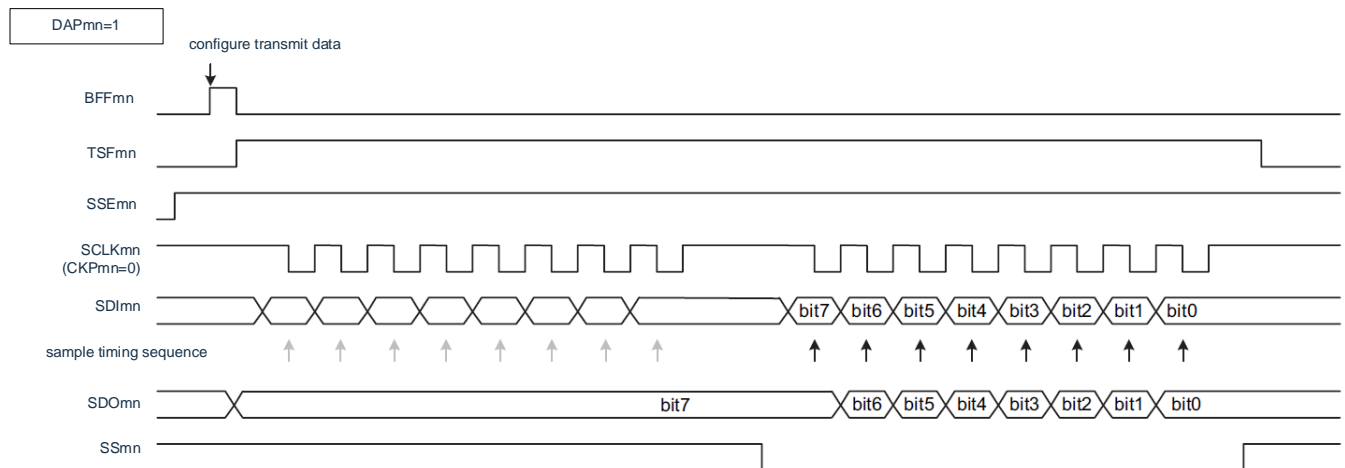
Notice The SDO00 pin is selected for N-channel open-drain output mode.

Figure 14-71 Slave timing diagram of the select input function



During SSmn is high, even on the falling edge of the SCKmn (serial clock), no transmission occurs, and no sampling of received data synchronized with the rising edge is taken.

During SSmn low, the output data is synchronized (shifted) with the falling edge of the serial clock and received synchronously with the rising edge.



When the DAPmn bit is “1”, the initial data (bit7) is supplied to the data output if the transmit data is set during when SSmn is high. However, even the rising edge of the SCLK mn (serial clock) is not shifted, and the accepted data synchronized with the falling edge is not sampled. If SSmn goes low, the output data is synchronized (shifted) with the next rising edge and received synchronously with the falling edge.

Remark m: unit number (m=0) n: channel number (n=0).

14.6.1 Slave transmission

Slave transmission refers to the operation of this product to send data to other devices in the state of input transmission clock from other devices.

Slave select input function	SSPI00
Object channel	Channel 0 for SCIO
Used pin	SCLK00, SDO00, SS00
Interrupt	INTSSPI00
	Can select transmit complete interrupt (single transmit mode) or buffer empty interrupt (continuous transmit mode).
Error detection flag	Only the Overflow Error Detection Flag (OVFmn).
Transmitted data length	7 or 8 bits
Transfer rate	Max. $f_{MCK}/6$ [Hz] ^{Note 1, 2}
Data phase	It can be selected via the DAPmn bit of the SCRmn register. <ul style="list-style-type: none"> • DAPmn=0: The data output starts when the serial clock starts running. • DAPmn=1: Starts data output half a clock before the serial clock starts running.
Clock phase	It can be selected via the CKPmn bit of the SCRmn register. <ul style="list-style-type: none"> • CKPmn=0: Positive phase • CKPmn=1: Negative phase
Data direction	MSB first or LSB first
Slave select input function	Operation of the slave selection function can be selected.

Note 1. Because the external serial clock of the SCLK00 pin input is used internally, the maximum transfer rate is $f_{MCK}/6$ [Hz].

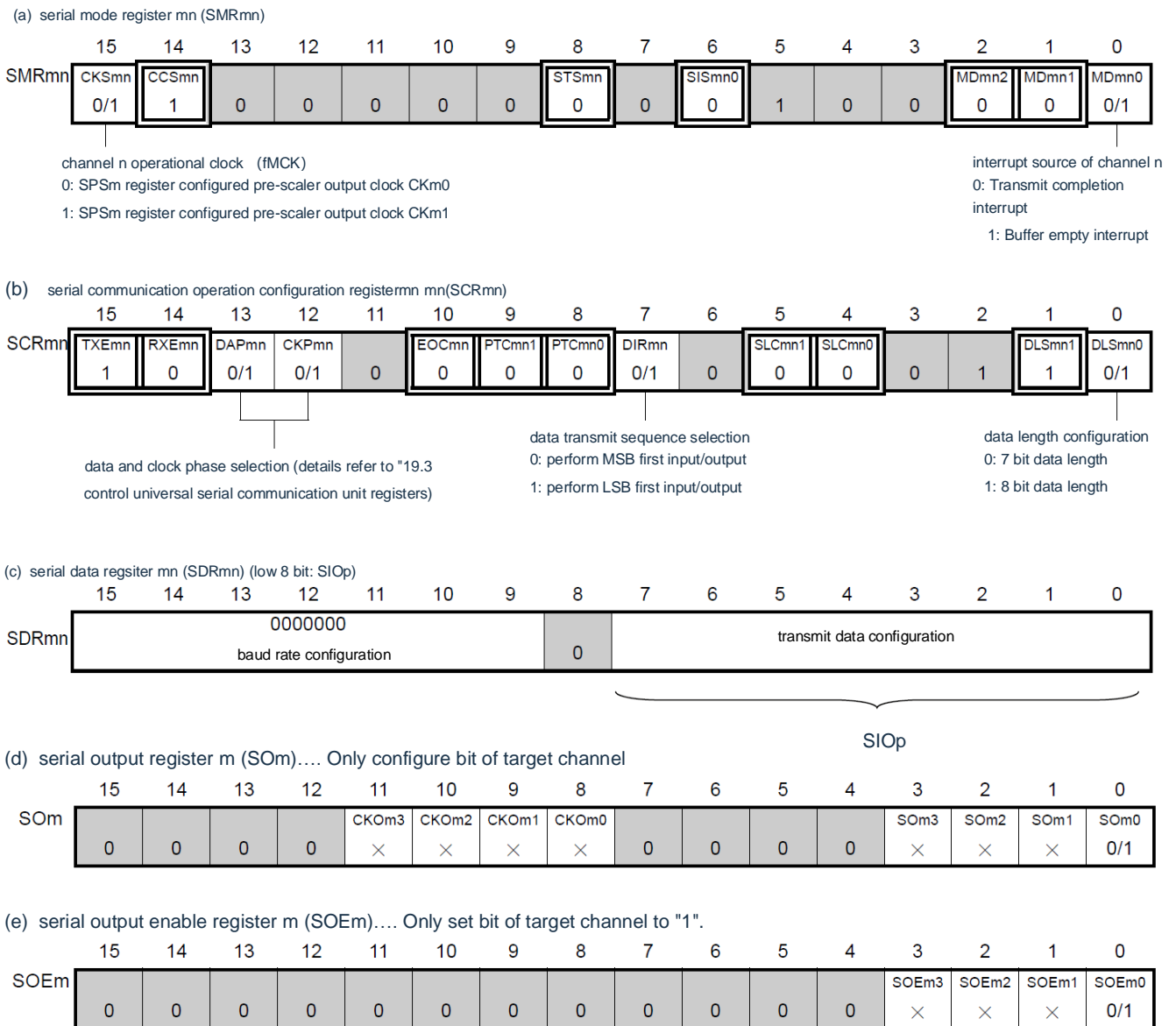
2. It must be used within the range of peripheral functional characteristics (refer to the datasheet) that fulfill this condition and satisfy the electrical characteristics.

Remark 1. f_{MCK} : Operation clock frequency of the object channel

2. m: unit number (m=0) n: channel number (n=0).

(1) Register setting

Figure 14-72 Example of register settings when slave select input function (SSPI00) slave transmits (1/2)



Remark 1. m: unit number (m=0) n: channel number (n=0) p: SSPI number (p=00)
 2. □ : Fixed in SSPI slave send mode. ■ : Cannot be set (initial value).
 x: This is the bit that cannot be used in this mode (set 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 14-72 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	SSm3	SSm2	SSm1	SSm0
													×	×	×	0/1

(g) input switch control register (ISC).... This is controlled by SS00 pin of SSPI00 slave channel (channel 0 of unit 0).

ISC	7	6	5	4	3	2	1	0
	SSIE00						ISC1	ISC0
	0/1	0	0	0	0	0	0/1	0/1

0: SS00 pin input invalid
 1: SS00 pin input valid

Remark1.m: Unit number (m=0) n: Channel number (n=0) p: SSPI number (p=00)

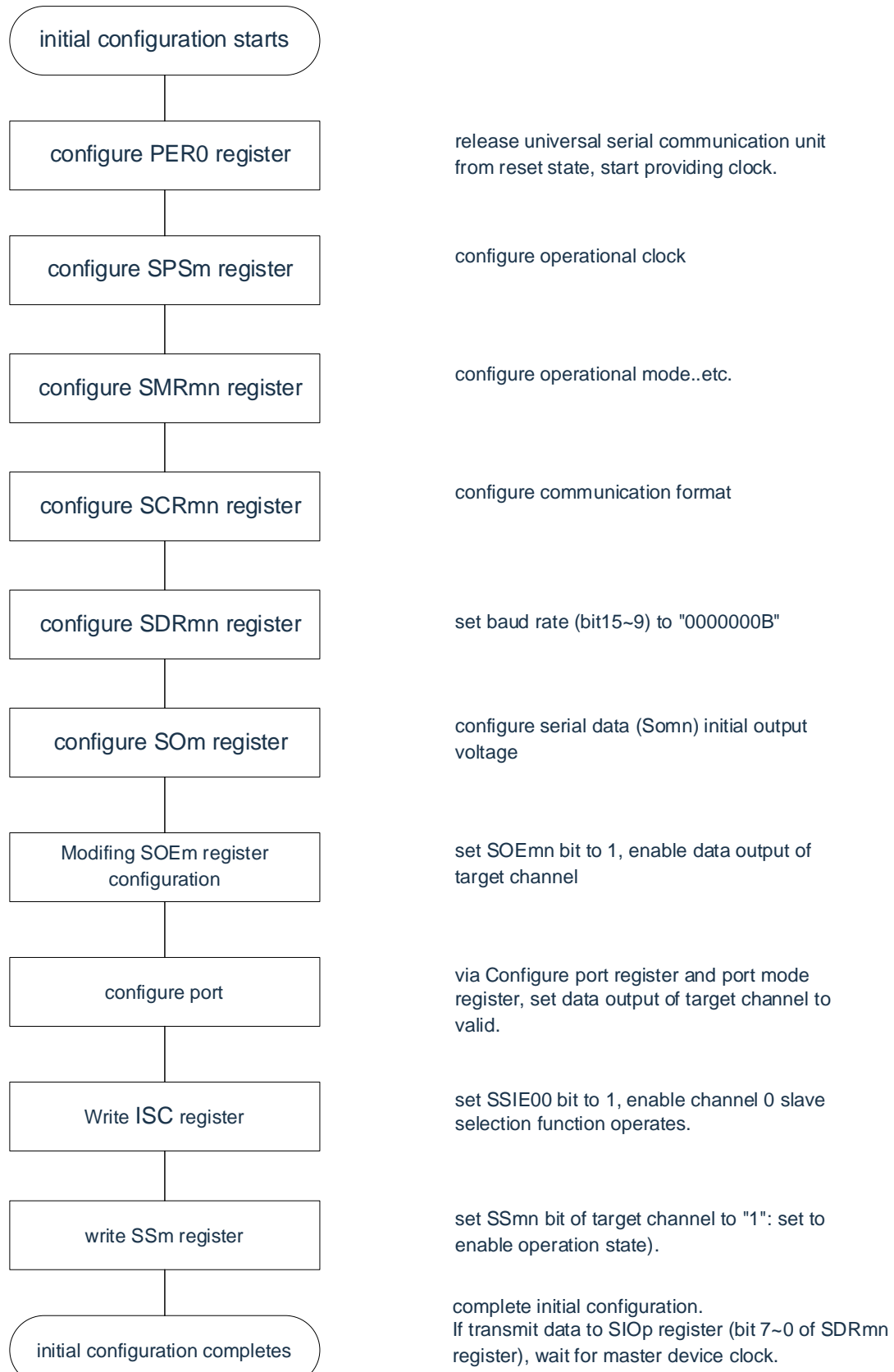
 2. : Fixed in SSPI slave transmit mode. : Cannot be set (initial value).

x: This is the bit that cannot be used in this mode (set the initial value if it is not used in other modes either).

0/1: Set "0" or "1" according to the user's purpose.

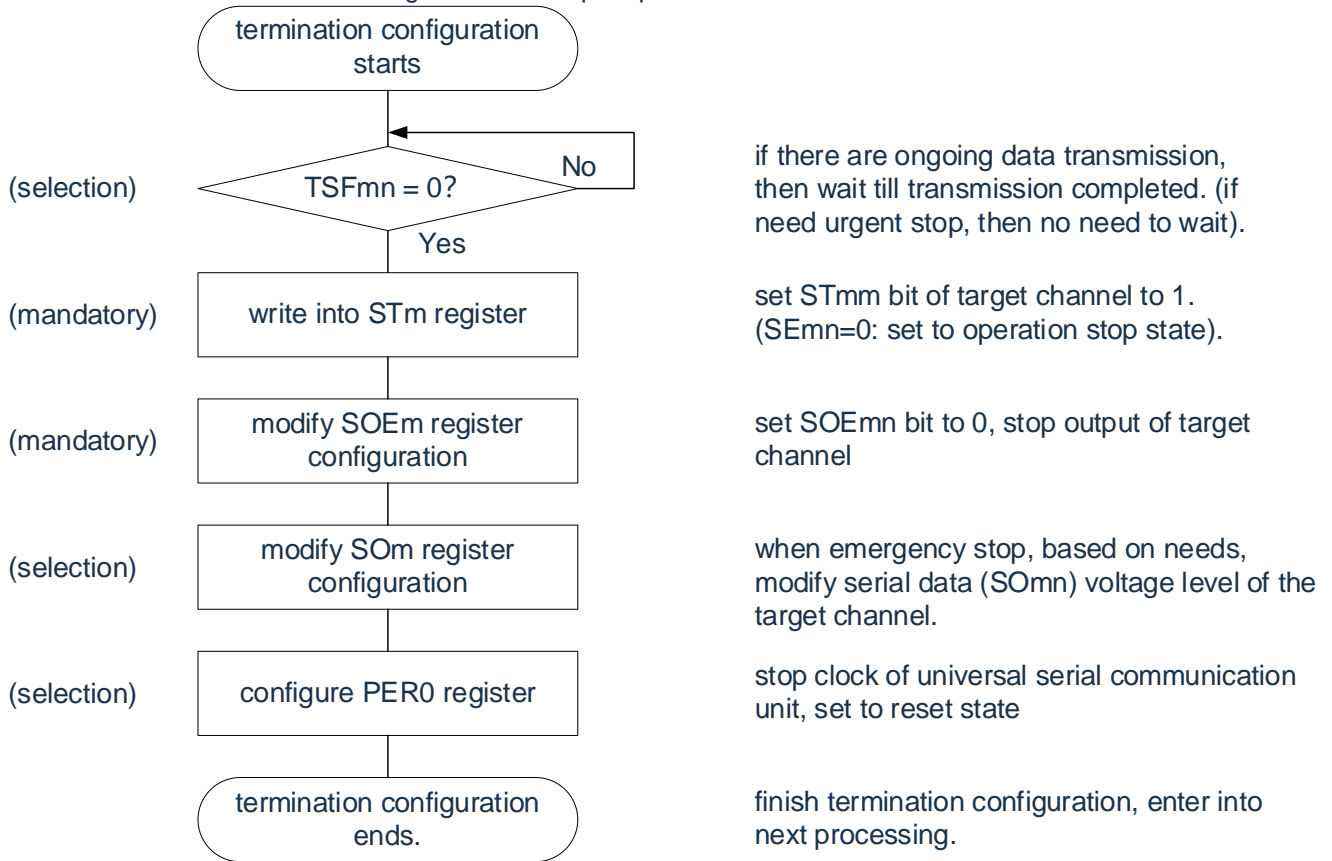
(2) Operation steps

Figure 14-73 Initial setup steps for slave transmission



Remark m: unit number (m=0) n: channel number (n=0) p: SSPI number (p=00)

Figure 14-74 Stop steps for slave transmission

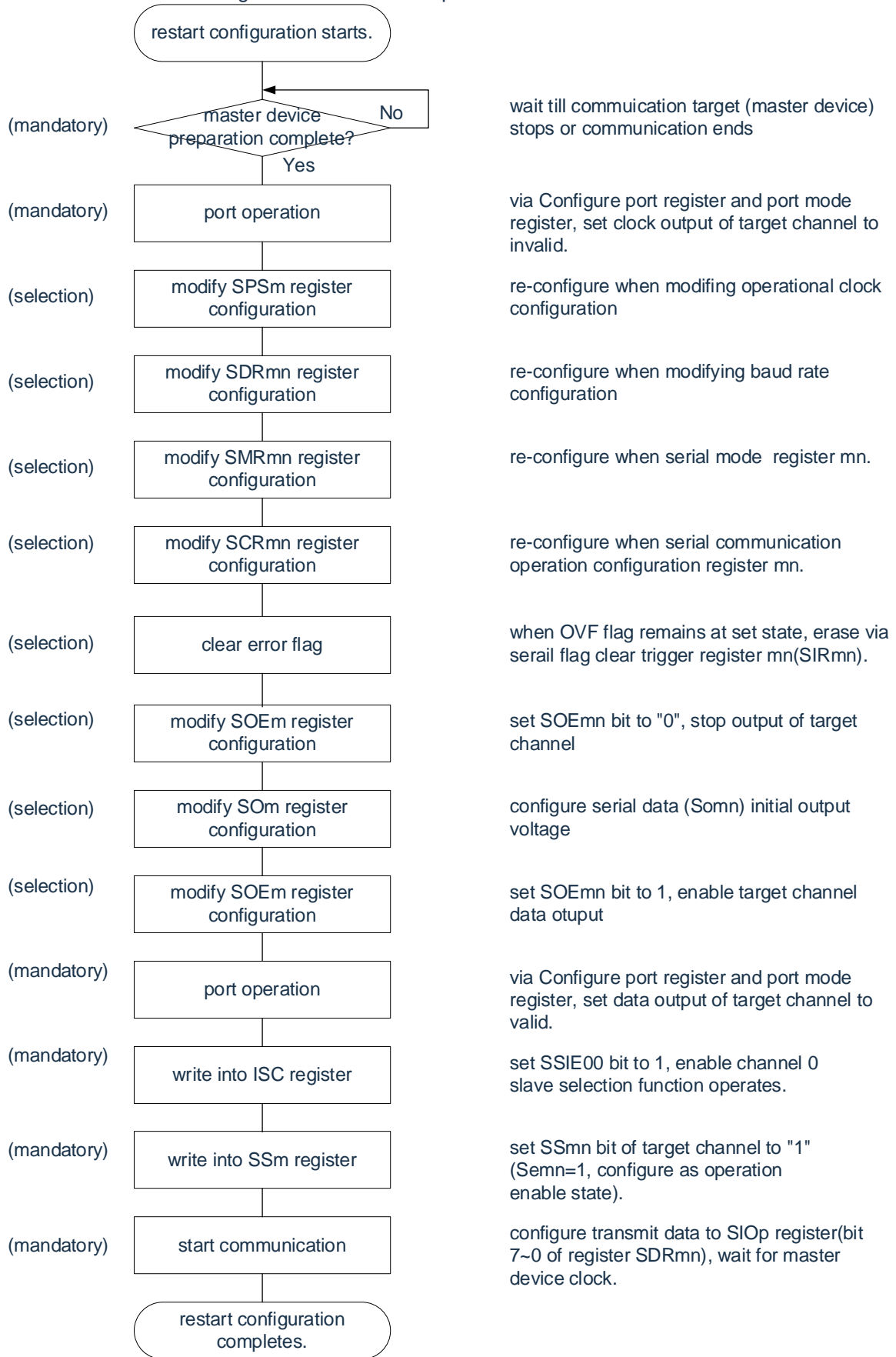


Remark m: unit number (m=0) n: channel number (n=0) p: SSPI number (p=00)

Note 1. If you override PER0 in the abort setting to stop the clock, you must wait until the communication object (master device) stops or the communication is over to make the initial setting instead of starting the setting again.

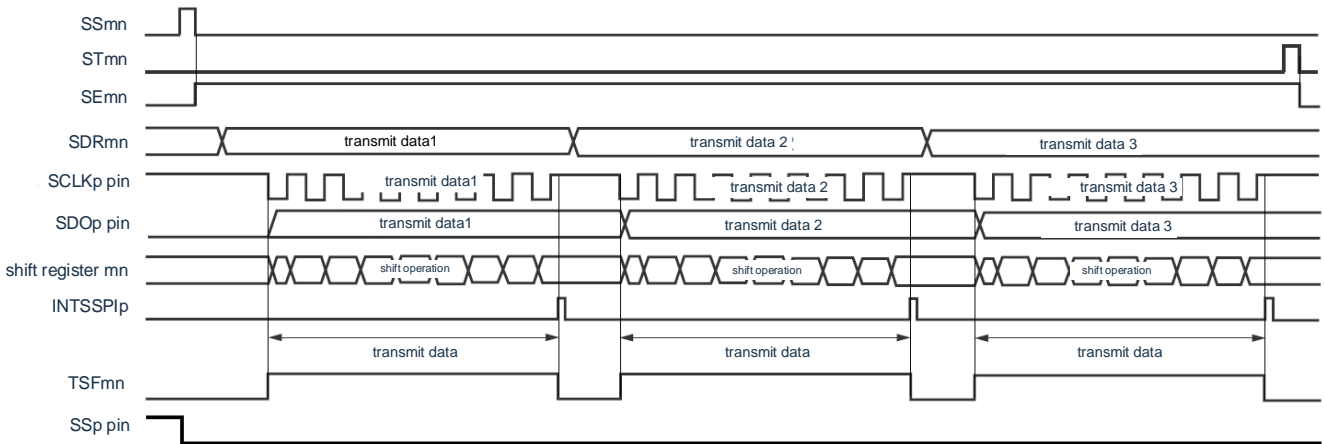
2. m: unit number (m=0) n: channel number (n=0) p: SSPI number (p=00).

Figure 14-75 Restart steps for slave transmission



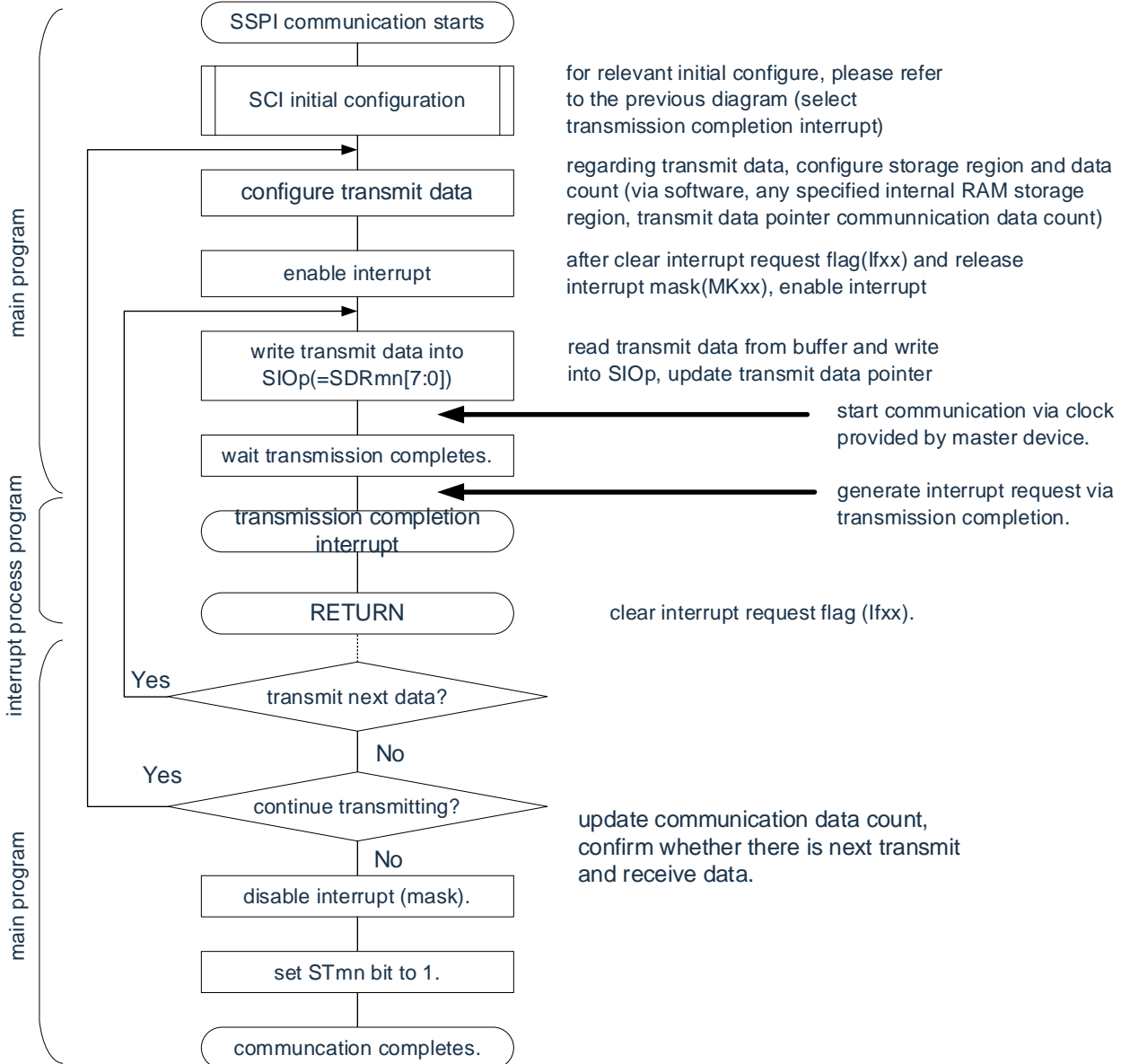
(3) Processing flow (single transmit mode)

Figure 14-76 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)

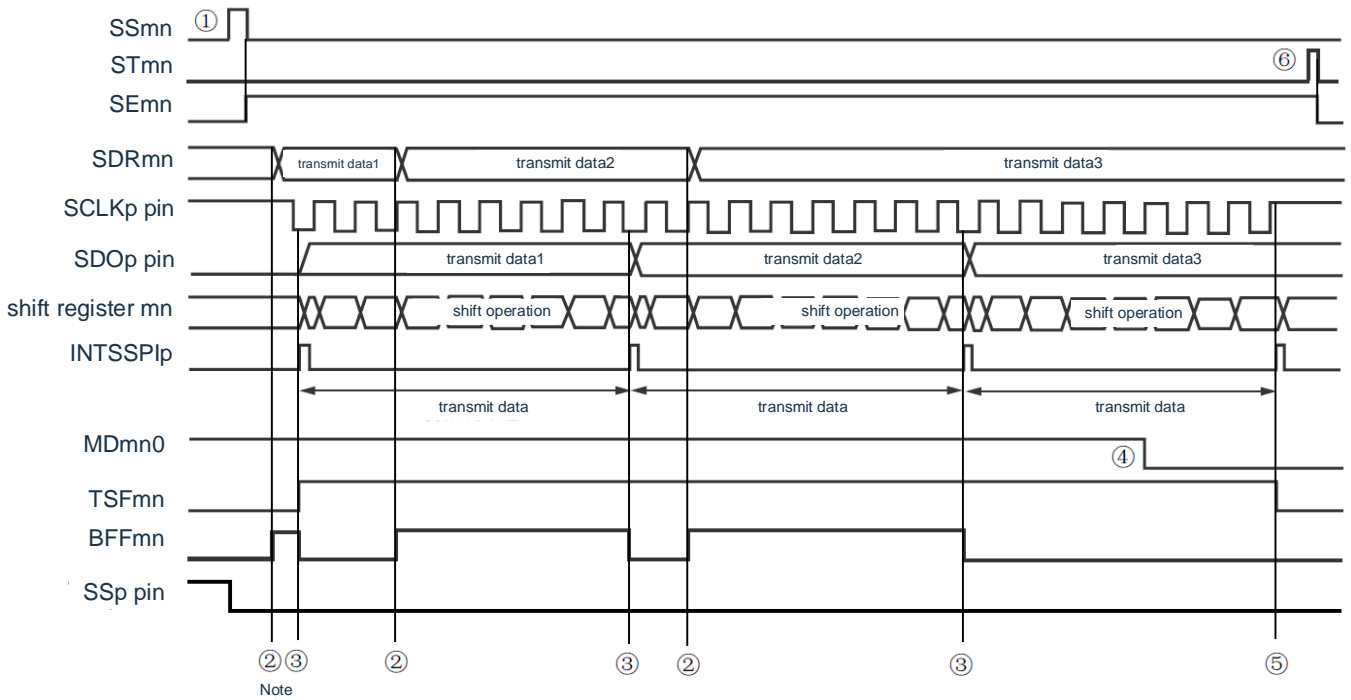
Figure 14-77 Flowchart of slave transmission (single transmit mode)



Remark m: unit number (m=0) n: channel number (n=0) p: SSPI number (p=00)

(4) Processing flow (continuous transmit mode)

Figure 14-78 Timing diagram of slave transmit (continuous transmit mode) (type 1: DAPmn=0, CKPmn=0)

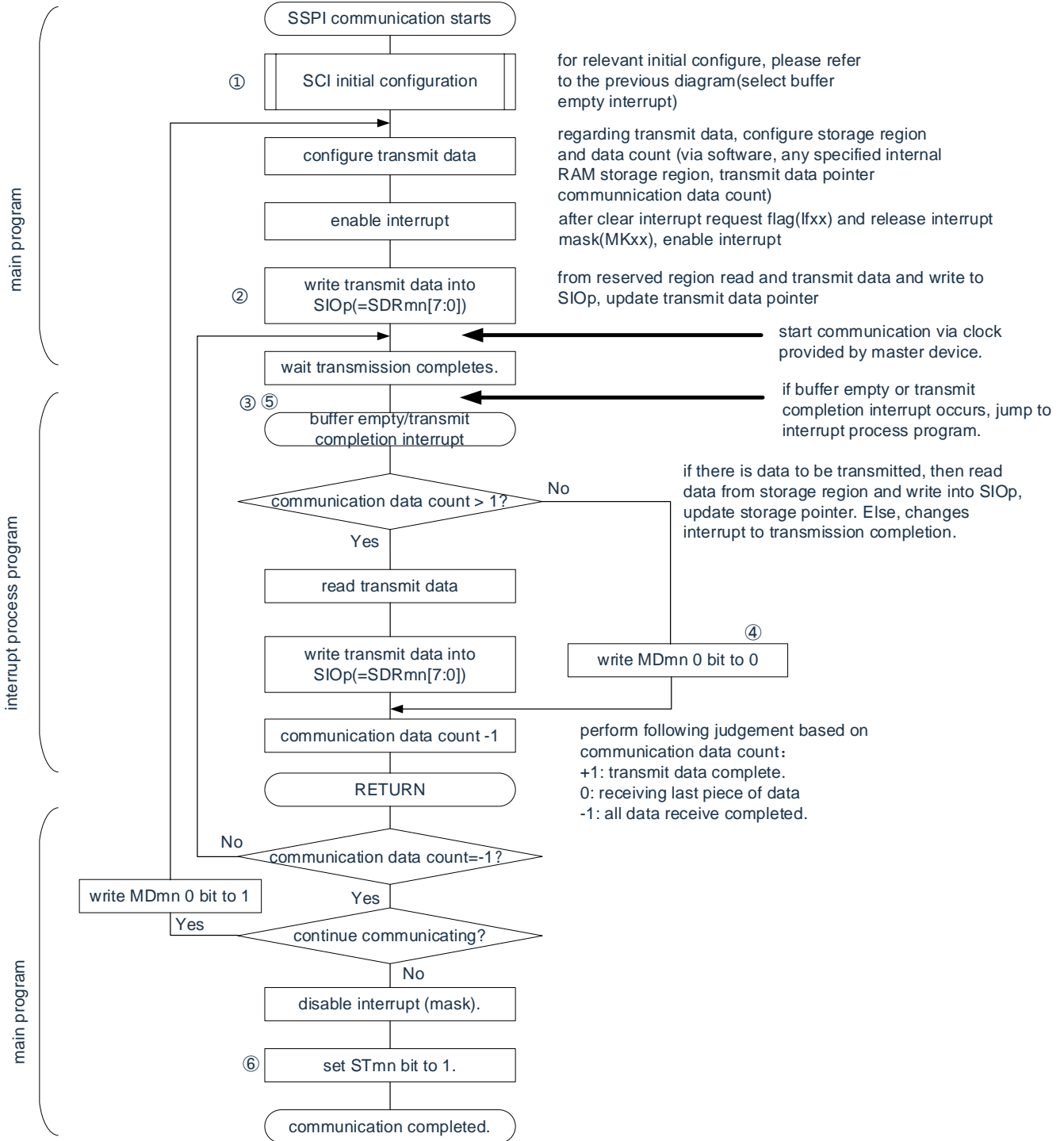


Note If the BFFmn bit of the serial status register mn (SSRmn) is “1” (when valid data is saved in the serial data register mn (SDRmn)) is given The SDRmn register writes the transmitted data and overrides the transmitted data.

Notice The MDmn0 bit of the serial mode register mn (SMRmn) can be overridden even during operation. However, it must be overridden before the last bit can be transferred.

Remark m: unit number (m=0) n: channel number (n=0) p: SSPI number (p=00)

Figure 14-79 Flowchart of slave transmission (continuous send mode)



Remark 1. ① to ⑥ correspond to ① to ⑥ in “Figure 14-78 Timing diagram of slave transmit (continuous transmit mode)”.

2. m: unit number (m=0) n: channel number (n=0) p: SSPI number (p=00)

14.6.2 Slave reception

Slave reception refers to the operation of this product to receive data from other devices in the state of transmitting clocks from other devices.

Slave select input function	SSPI00
Object channel	Channel 0 for SCIO
Used pin	SCLK00, SDI00, SS00
Interrupt	INTSSPI00 Limited to transmit complete interrupts (disable setting buffer empty interrupts).
Error detection flag	Only the Overflow Error Detection Flag (OVFmn).
Transmitted data length	7 or 8 bits
Transfer rate	$\text{Max}f_{\text{MCK}}/6[\text{Hz}]^{\text{Note1, 2}}$
Data phase	It can be selected via the DAPmn bit of the SCRmn register. <ul style="list-style-type: none"> • DAPmn=0: The data output starts when the serial clock starts running. • DAPmn=1: Starts data output half a clock before the serial clock starts running.
Clock phase	It can be selected via the CKPmn bit of the SCRmn register. <ul style="list-style-type: none"> • CKPmn=0: Positive phase • CKPmn=1: Negative phase
Data direction	MSB first or LSB first
Slave select input function	You can select the operation of the Slave select input function.

Note 1. The maximum transfer rate is $f_{\text{MCK}}/6$ [Hz] because the external serial clock input to the SCLK00 pin is sampled internally and then used.

2. It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

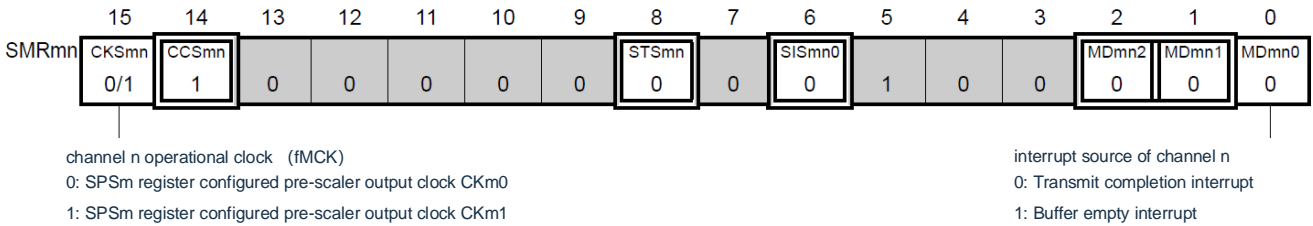
Remark 1. f_{MCK} : Operation clock frequency of the object channel

2. m: unit number (m=0) n: channel number (n=0).

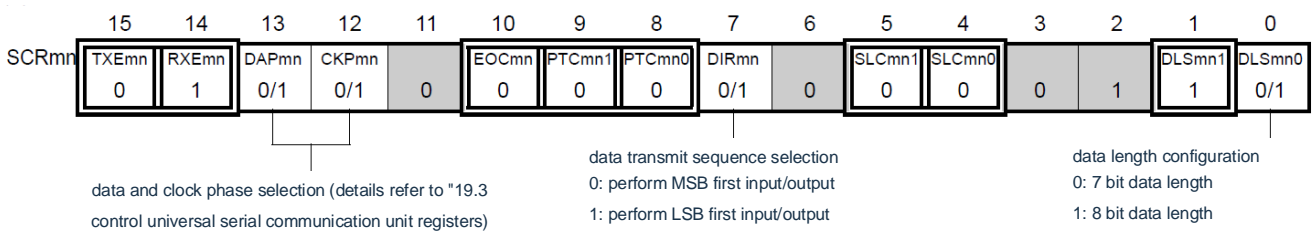
(1) Register setting

Figure 14-80 Example of register setting contents for slave selection input function (SSPI00) slave reception (1/2)

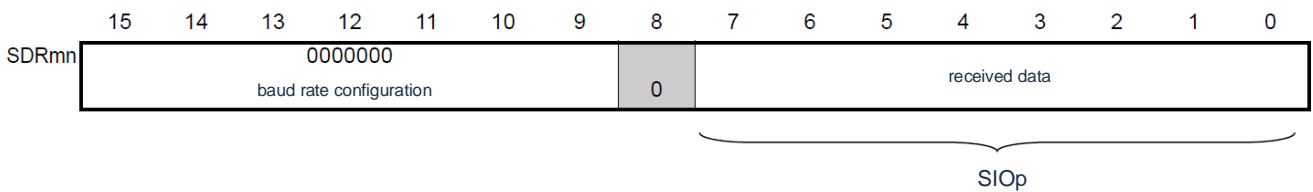
(a) serial mode register mn (SMRmn)



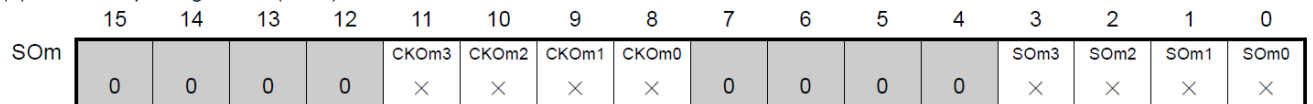
(b) serial communication operation configuration registermn mn(SCRmn)



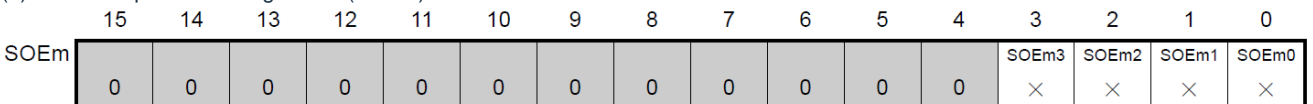
(c) serial data register mn (SDRmn) (low 8 bit: SIOp)



(d) serial output register m (SOM).... Not used in this mode.



(e) serial output enable register m (SOEm).... Not used in this mode.



Remark1.m: unit number (m=0) n: channel number (n=0) p: SSPI number (p=00)

2□: Fixed in slave receive mode. □: Cannot be set (initial value).

x: This is the bit that cannot be used in this mode (set 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 14-81 Example of register setting contents for slave selection input function (SSPI00) slave reception (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	SSm3	SSm2	SSm1	SSm0
													×	×	×	0/1

(g) input switch control register (ISC).... This is controlled by SS00 pin of SSPI00 slave channel (channel 0 of unit 0).

	7	6	5	4	3	2	1	0
ISC	SSIE00						ISC1	ISC0
	0/1	0	0	0	0	0	0/1	0/1

0: SS00 pin input invalid
1: SS00 pin input valid

Remark 1.m: unit number (m=0) n: channel number (n=0) p: SSPI number (p=00)

2. : Fixed in slave receive mode. : Cannot be set (initial value).

x: This is the bit that cannot be used in this mode (set 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) Operation steps

Figure 14-82 Initial setup steps for slave reception

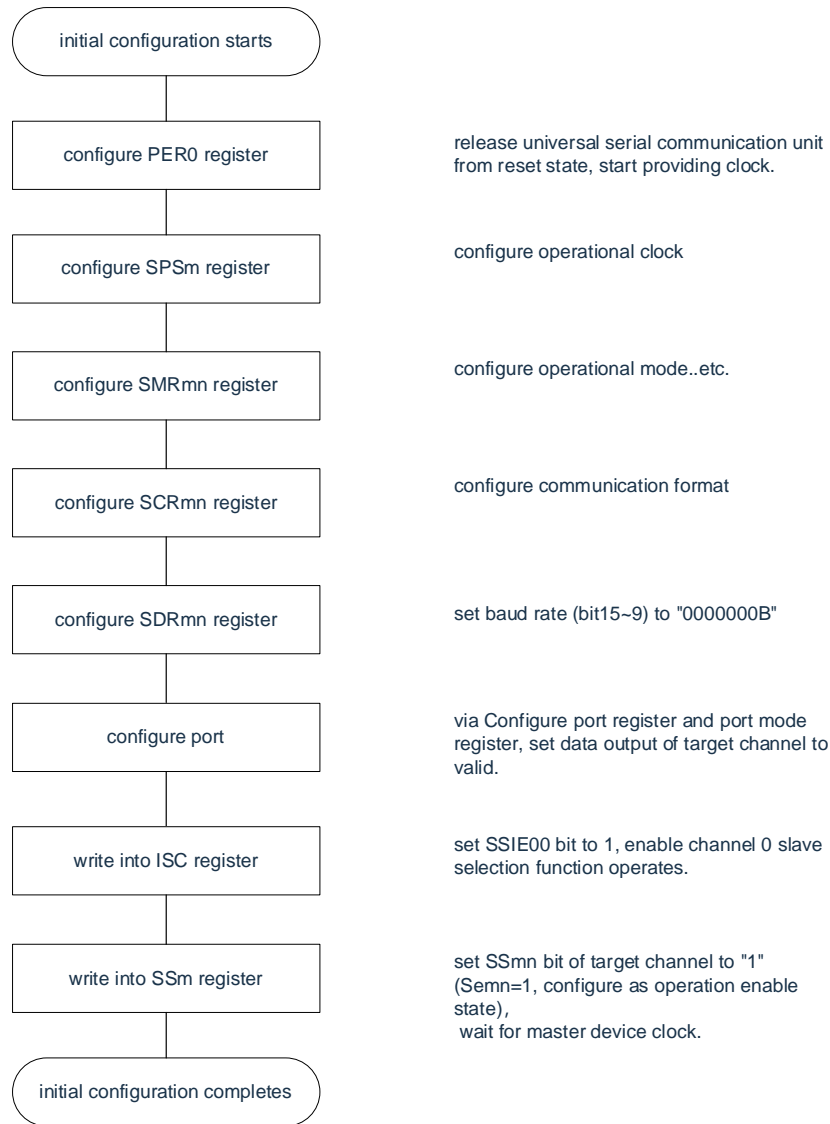
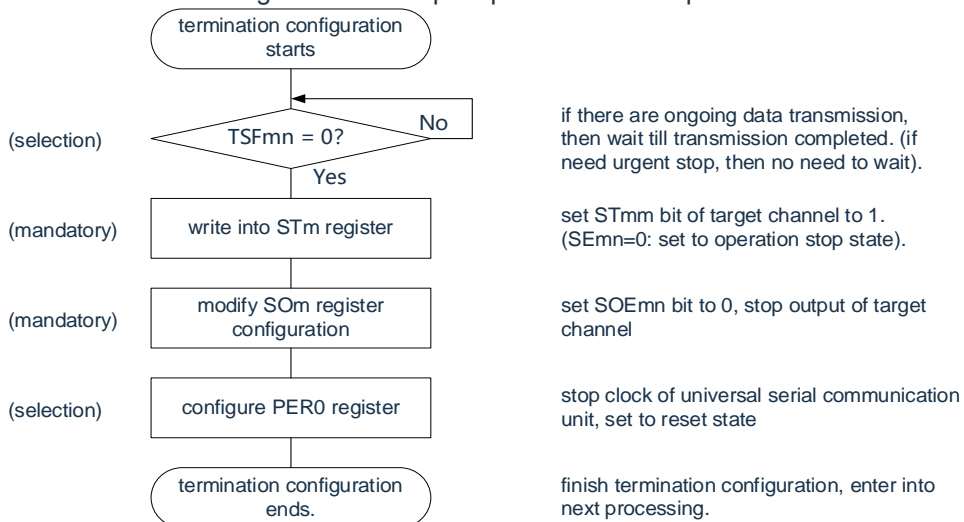
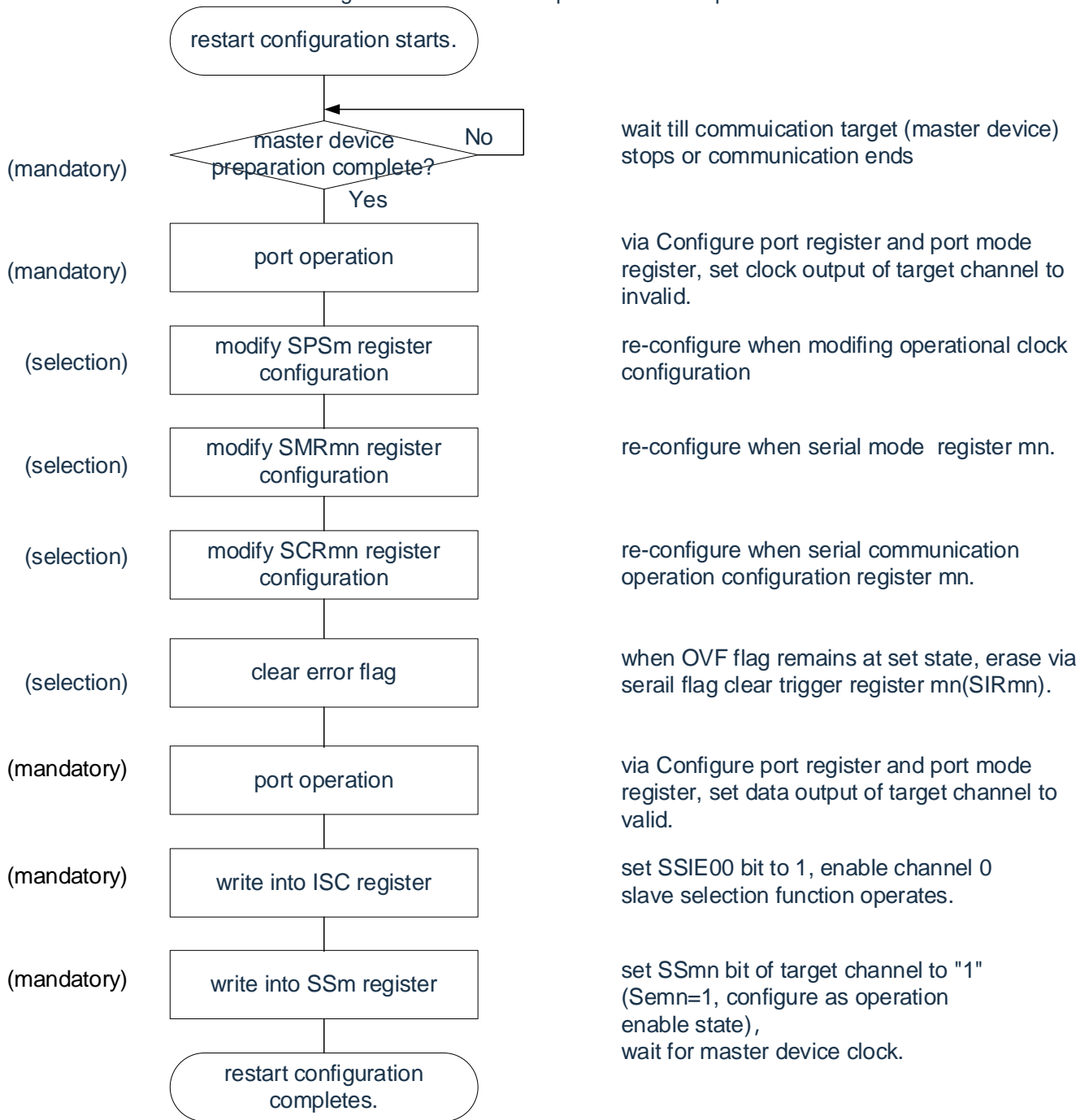


Figure 14-83 Stop steps for slave reception



Remark m: unit number (m=0) n: channel number (n=0) p: SSPI number (p=00)

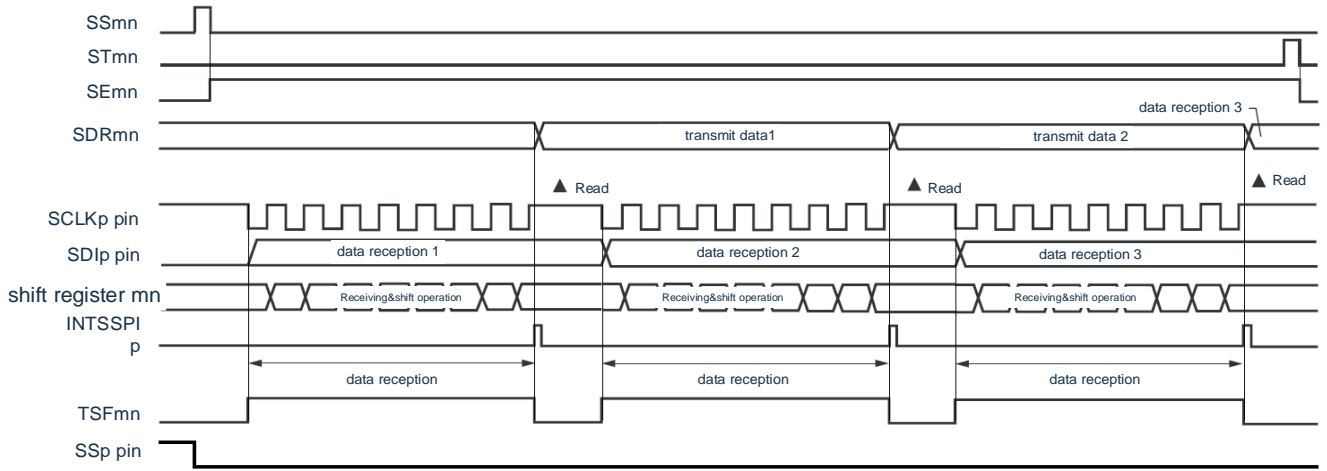
Figure 14-84 Restart steps for slave reception



Remark m: unit number (m=0) n: channel number (n=0) p: SSPI number (p=00)

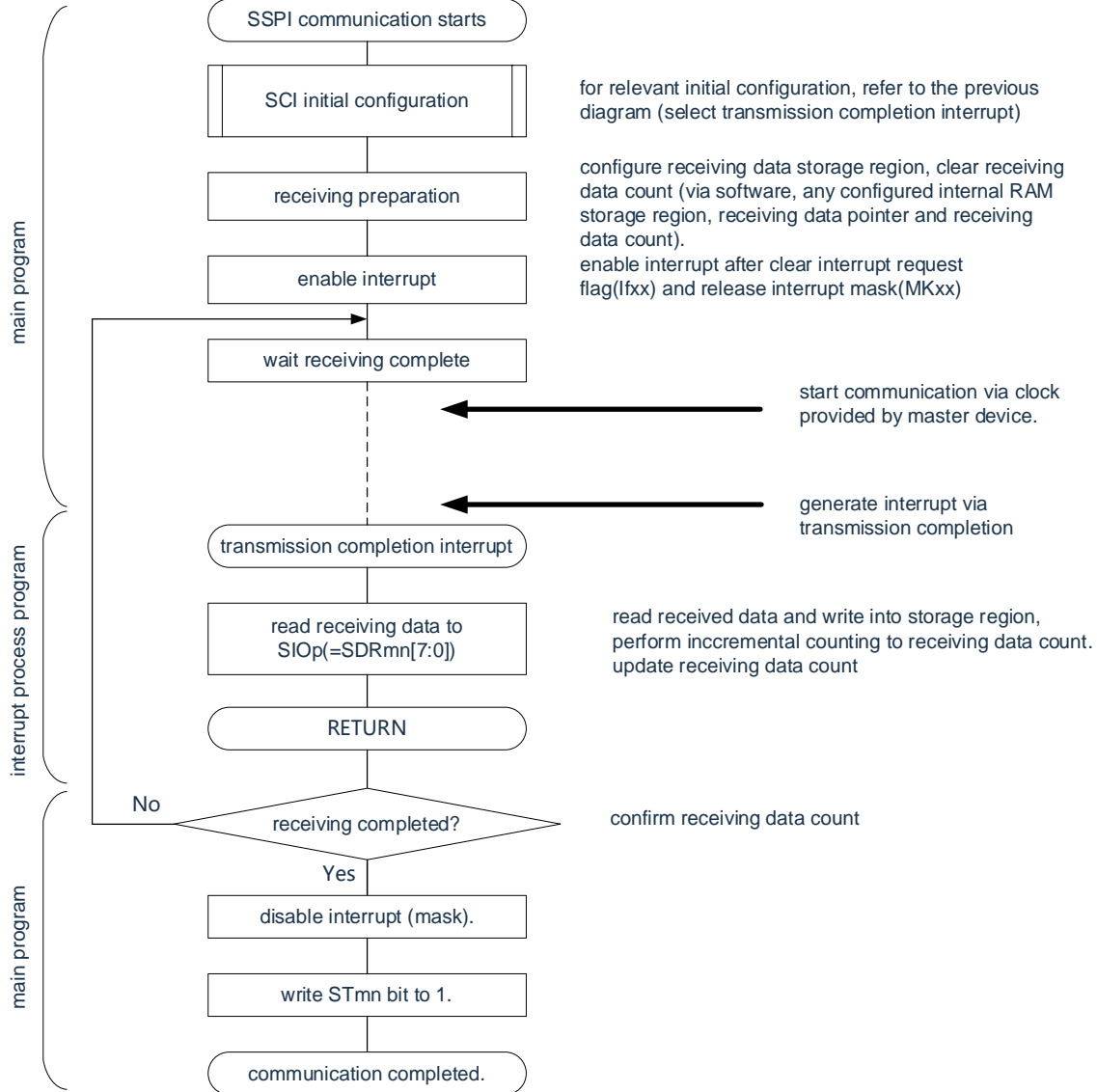
(3) Processing flow (single receive mode)

Figure 14-85 Timing diagram of slave reception (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 14-86 Flowchart of slave reception (single receive mode)



14.6.3 Slave transmission and reception

Slave transmission and reception refers to the operation of this product and other devices for data transmission and reception in the state of transmitting clocks from other devices.

Slave select input function	SSPI00
Object channel	Channel 0 for SCI0
Used pin	SCLK00, SDI00, SDO00, SS00
Interrupt	INTSSPI00 Can select transmit complete interrupt (single transmit mode) or buffer empty interrupt (continuous transmit mode).
Error detection flag	Only the Overflow Error Detection Flag (OVFmn).
Transmitted data length	7 or 8 bits
Transfer rate	Max. $f_{MCK}/6$ [Hz] ^{Note 1, 2}
Data phase	It can be selected via the DAPmn bit of the SCRmn register. <ul style="list-style-type: none"> • DAPmn=0: The data output starts when the serial clock starts running. • DAPmn=1: Starts data output half a clock before the serial clock starts running.
Clock phase	It can be selected via the CKPmn bit of the SCRmn register. <ul style="list-style-type: none"> • CKPmn=0: Positive phase • CKPmn=1: Negative phase
Data direction	MSB first or LSB first
Slave select input function	Operation of the slave select 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 sampled internally and then used.

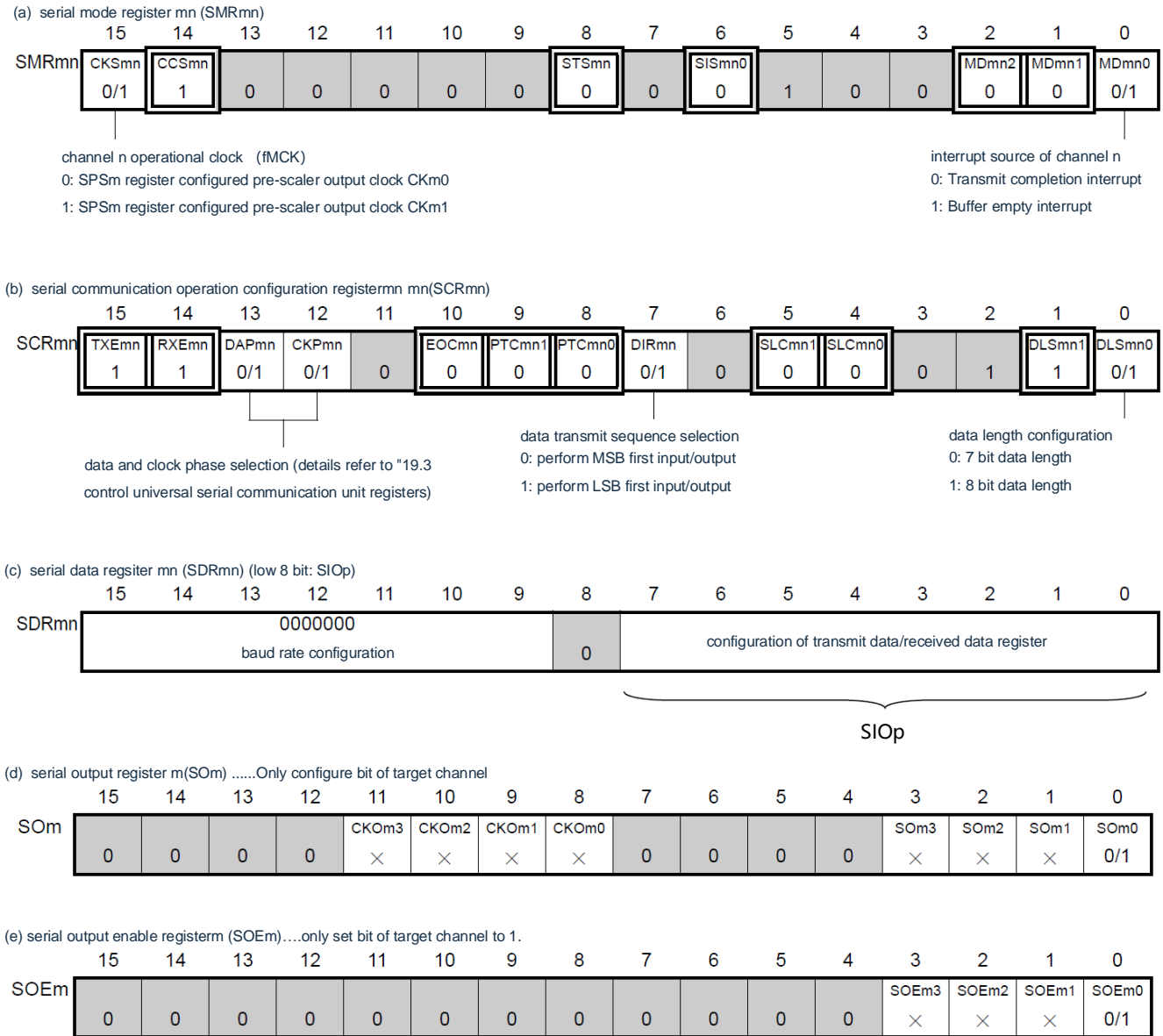
2. It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Remark 1. f_{MCK} : Operation clock frequency of the object channel

2. m: unit number (m=0) n: channel number (n=0).

(1) Register setting

Figure 14-87 Example of register setting contents for slave selection input function (SSPI00) slave transmission and reception (1/2)



Notice Data must be sent to the SIOp register settings before the master device starts the output clock.

Remark1.m: unit number (m=0) n: channel number (n=0) p: SSPI number (p=00)

2: Fixed in Slave Receive mode. : Cannot be set (initial value).

x: This is the bit that cannot be used in this mode (set 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 14-87 Example of register setting contents for slave selection input function (SSPI00) slave transmission and reception (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	SSm3 ×	SSm2 ×	SSm1 ×	SSm0 0/1

(g) input switch control register (ISC).... This is controlled by SS00 pin of 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 invalid
1: SS00 pin input valid

Notice Data must be sent to the SIOp register settings before the master device starts the output clock.

Remark 1. m: unit number (m=0) n: channel number (n=0) p: SSPI number (p=00)

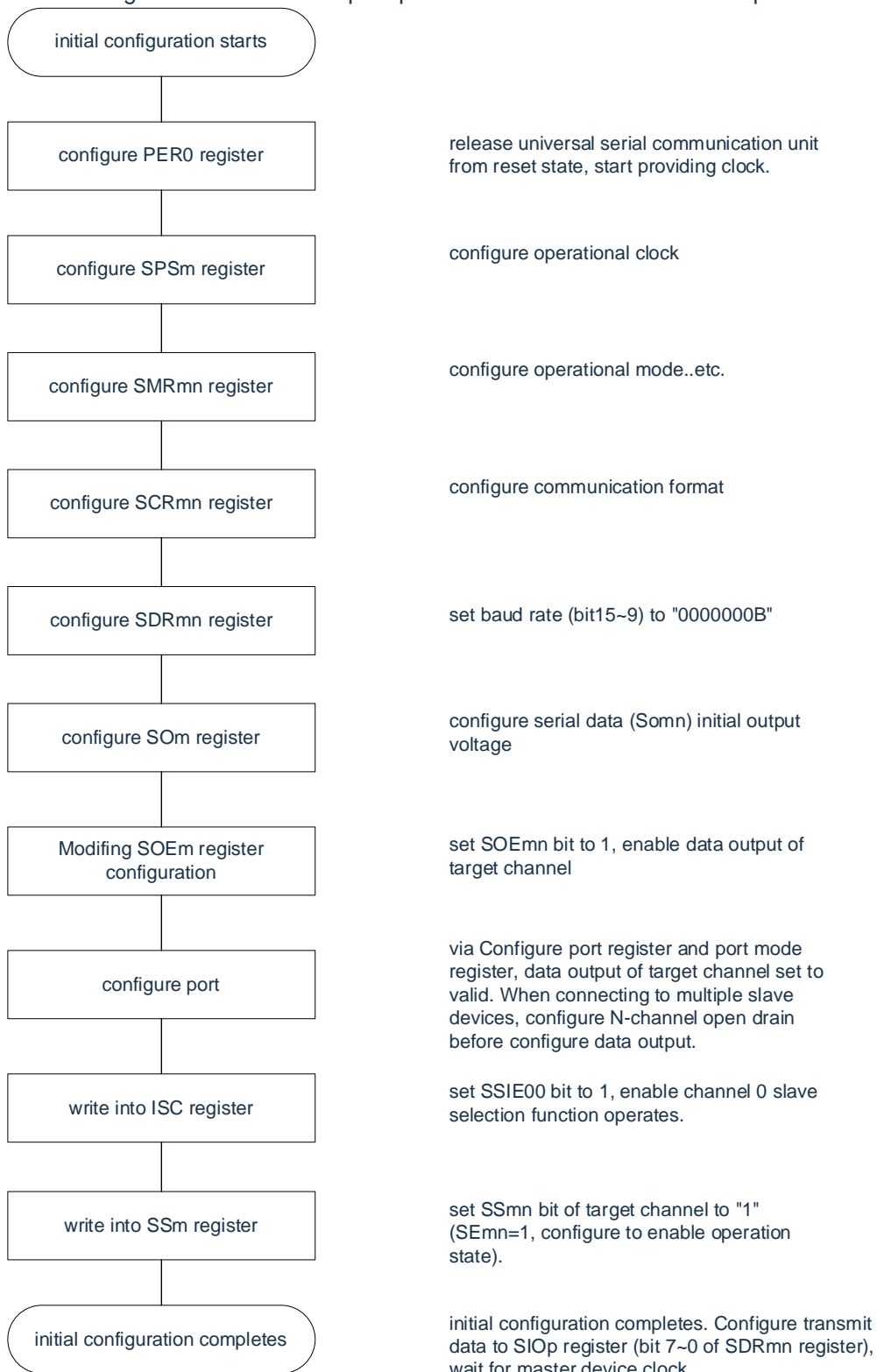
 2 : Fixed in slave receive mode. : Cannot be set (initial value).

x: This is the bit that cannot be used in this mode (set 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) Operation steps

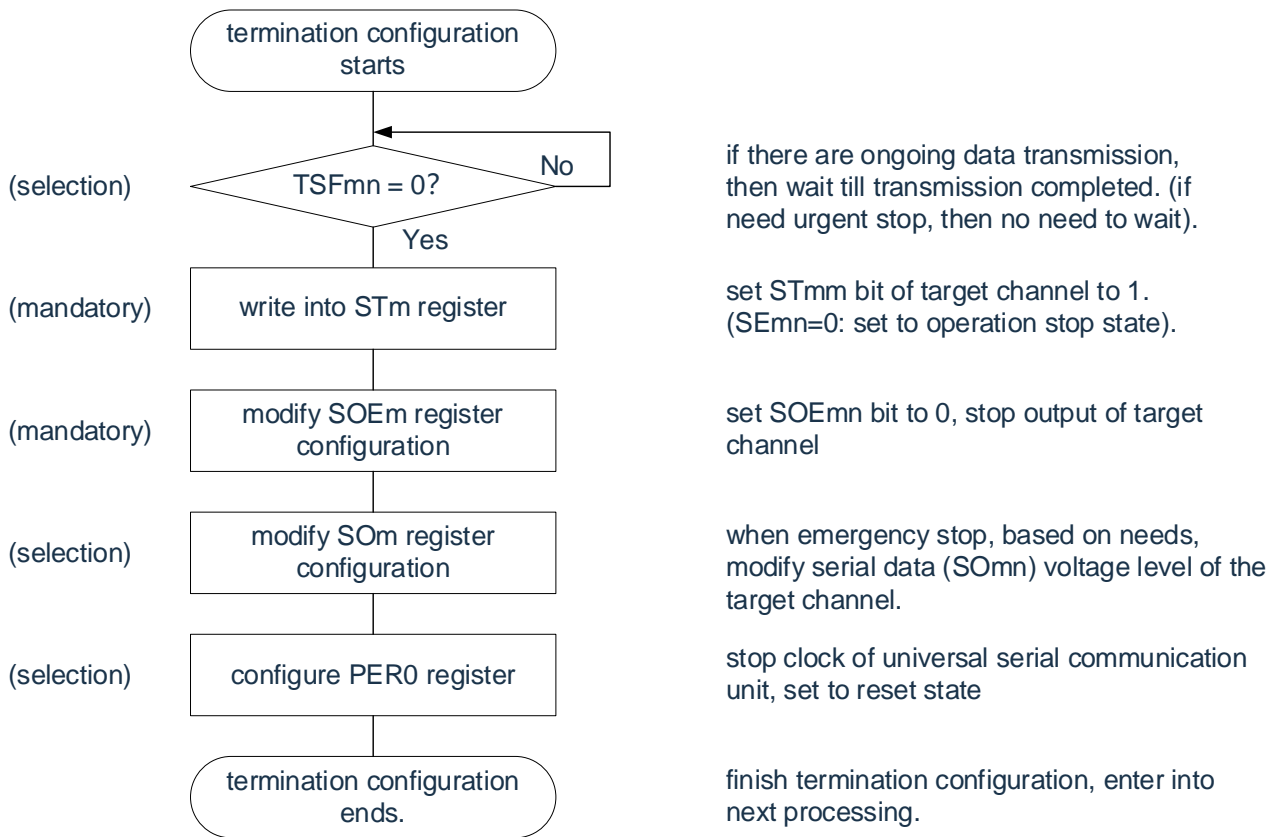
Figure 14-88 Initial setup steps for slave transmission and reception



Notice Data must be sent to the SIOp register settings before the master device starts the output clock.

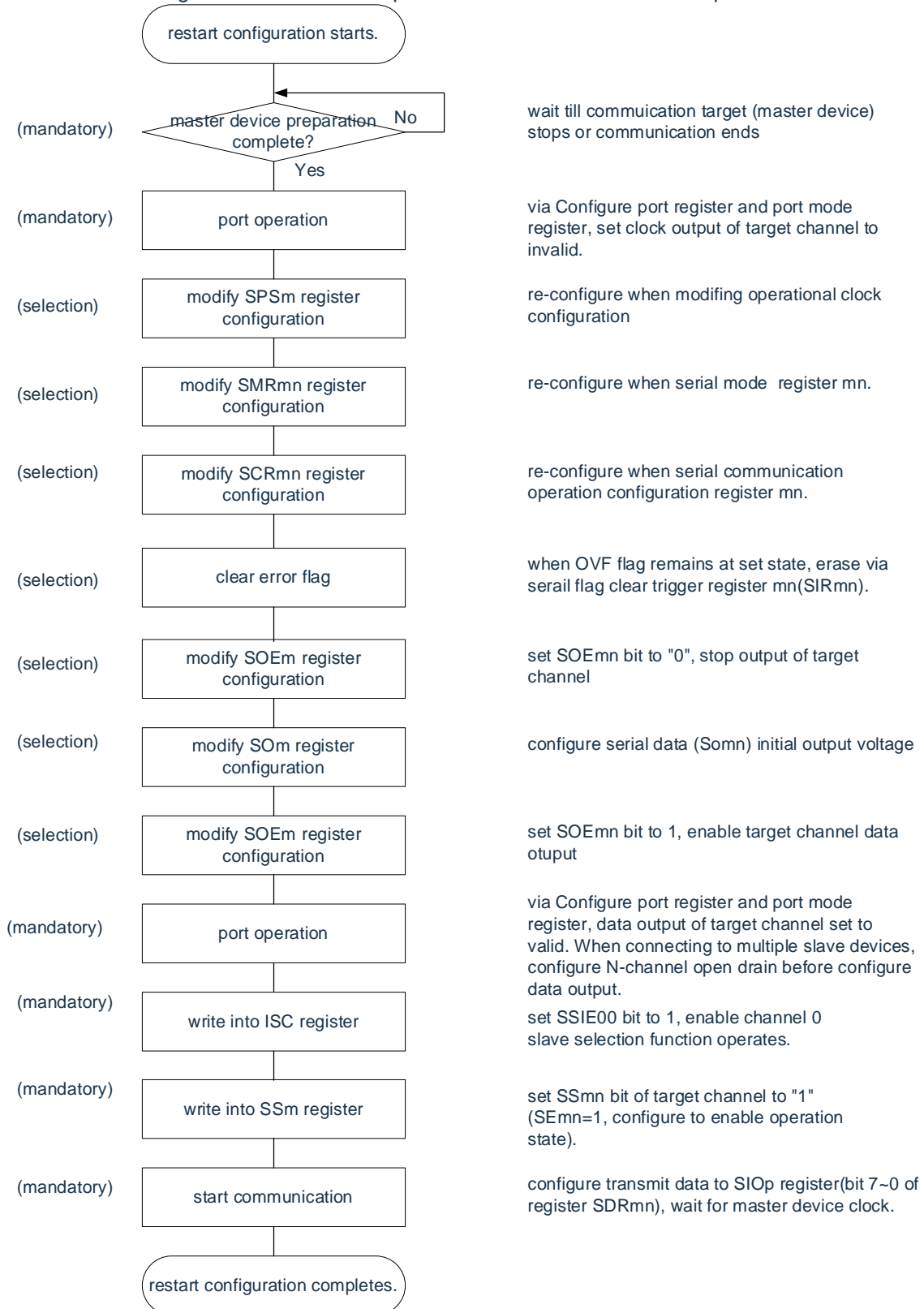
Remark m: unit number (m=0) n: channel number (n=0) p: SSPI number (p=00)

Figure 14-89 Stop steps of slave transmission and reception



Remark1.m: unit number (m=0) n: channel number (n=0) p: SSPI number (p=00)

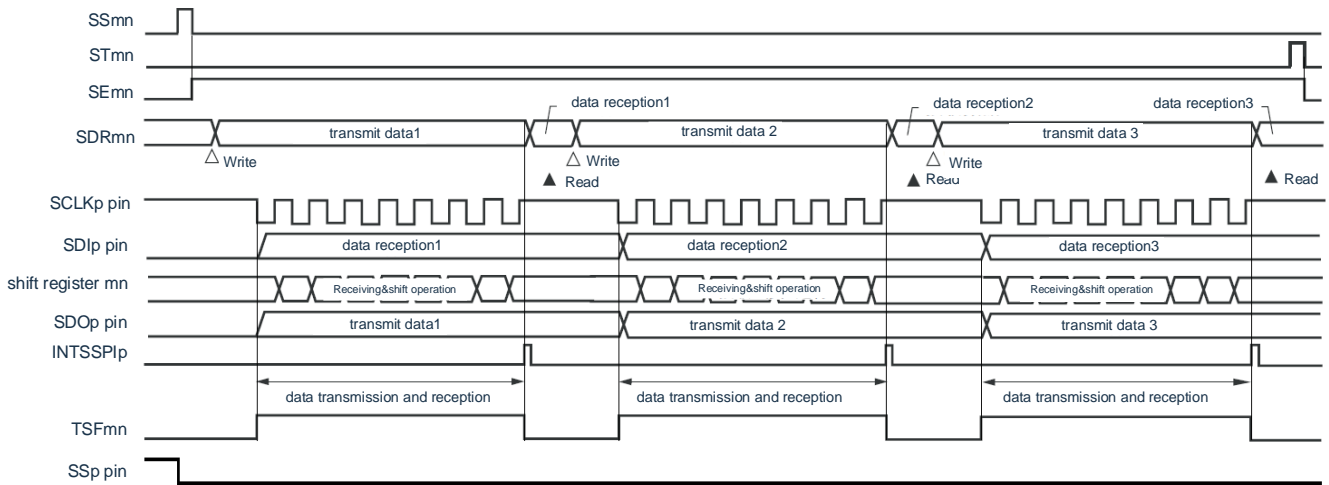
Figure 14-90 Restart steps of slave transmission and reception



- Notice 1. Before the master device starts to output the clock, data must be sent to the SIOp register settings.
2. If you override PER0 in the abort setting to stop the clock, you must wait until the communication object (master device) stops or the communication is over to make the initial setting instead of starting the setting again.

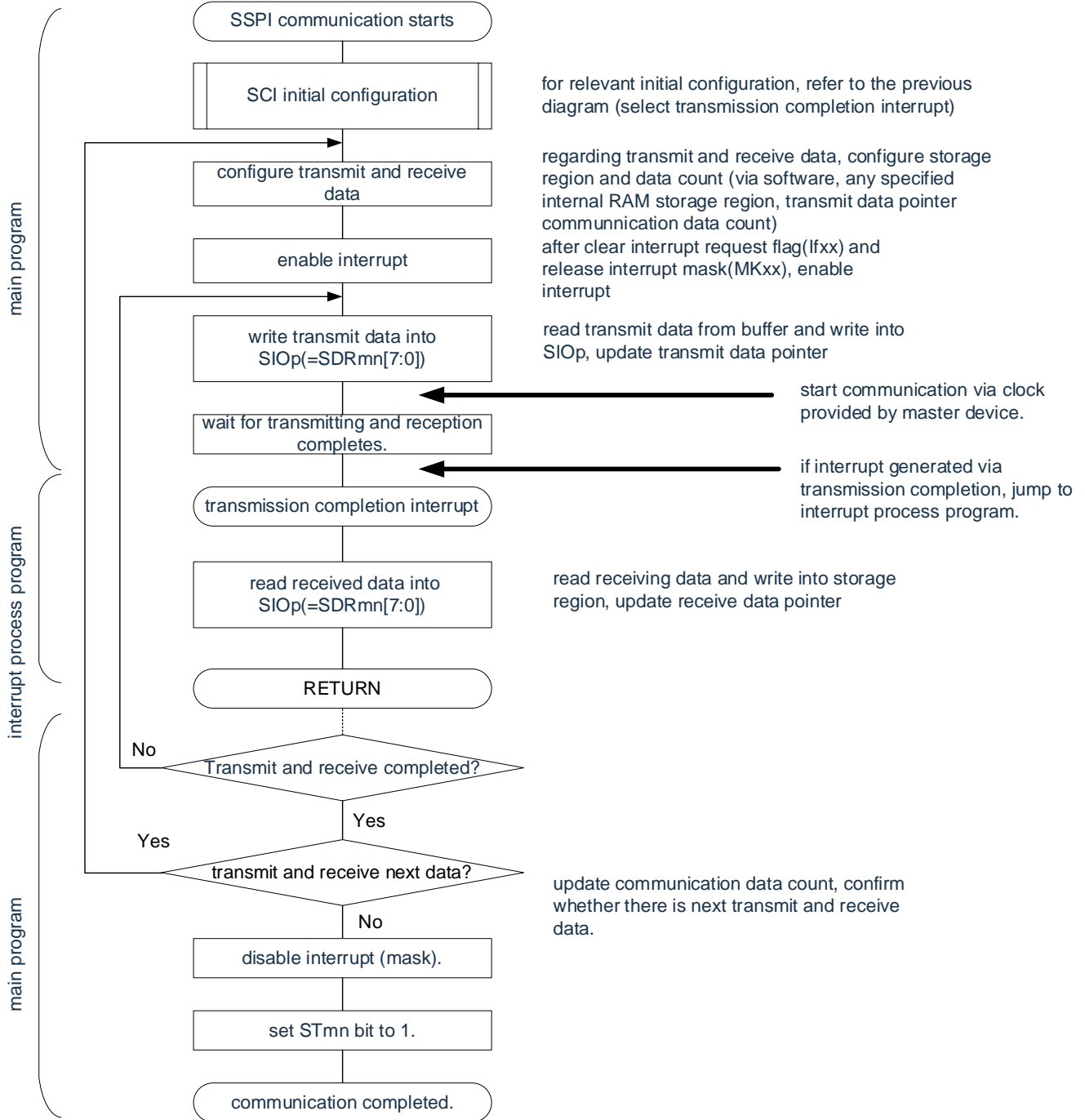
(3) Processing flow (single transmit and receive mode)

Figure 14-91 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 14-92 Flowchart of slave transmission and reception (single transmit and receive mode)

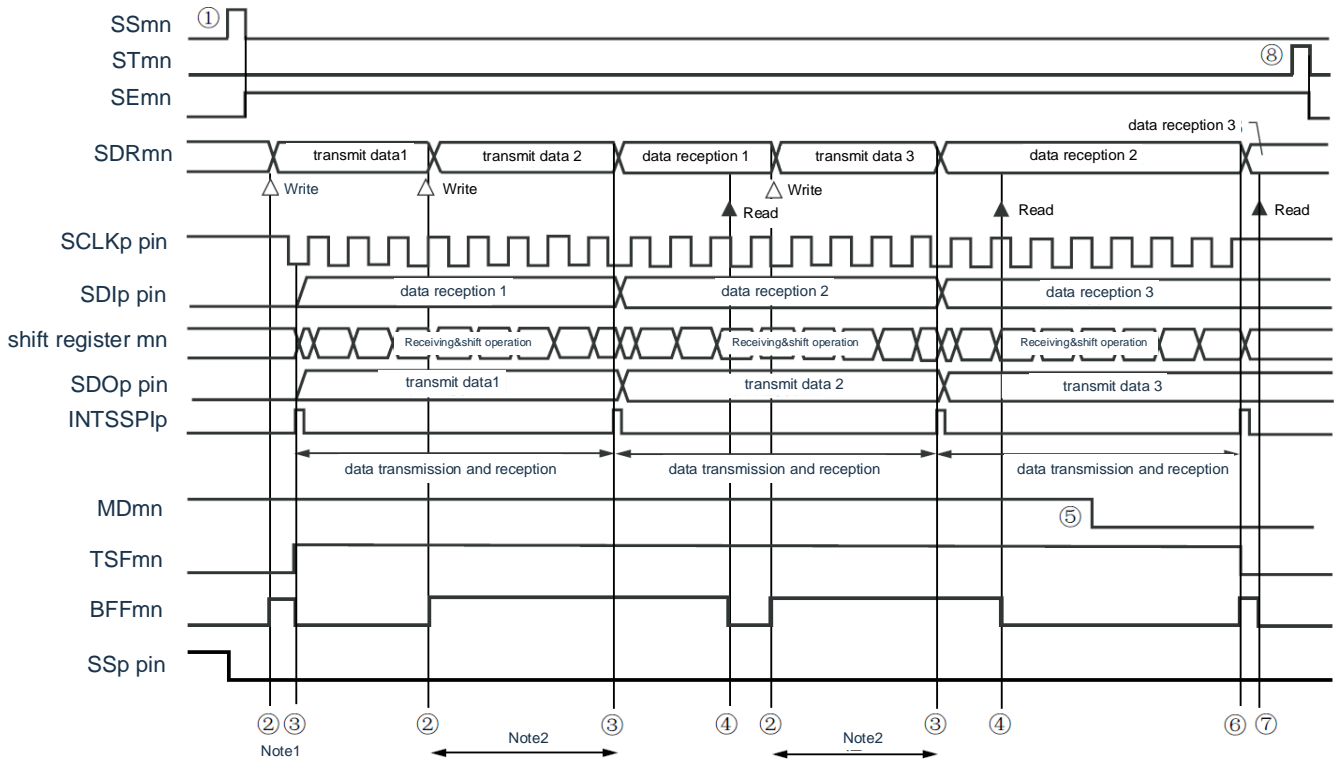


Notice Data must be sent to the SIOp register settings before the master device starts the output clock.

Remark m: unit number (m=0) n: channel number (n=0) p: SSPI number (p=00)

(4) Processing flow (continuous transmit and receive mode)

Figure 14-93 Timing diagram of slave transmission and reception (continuous transmit and receive mode)
(type 1: DAPmn=0, CKPmn=0)



Note 1. If the BFFmn bit of the serial status register mn (SSRmn) is “1” (valid data is saved in the serial data register mn (SDRmn) to write the send data to the SDRmn register, and rewrite the sent data.

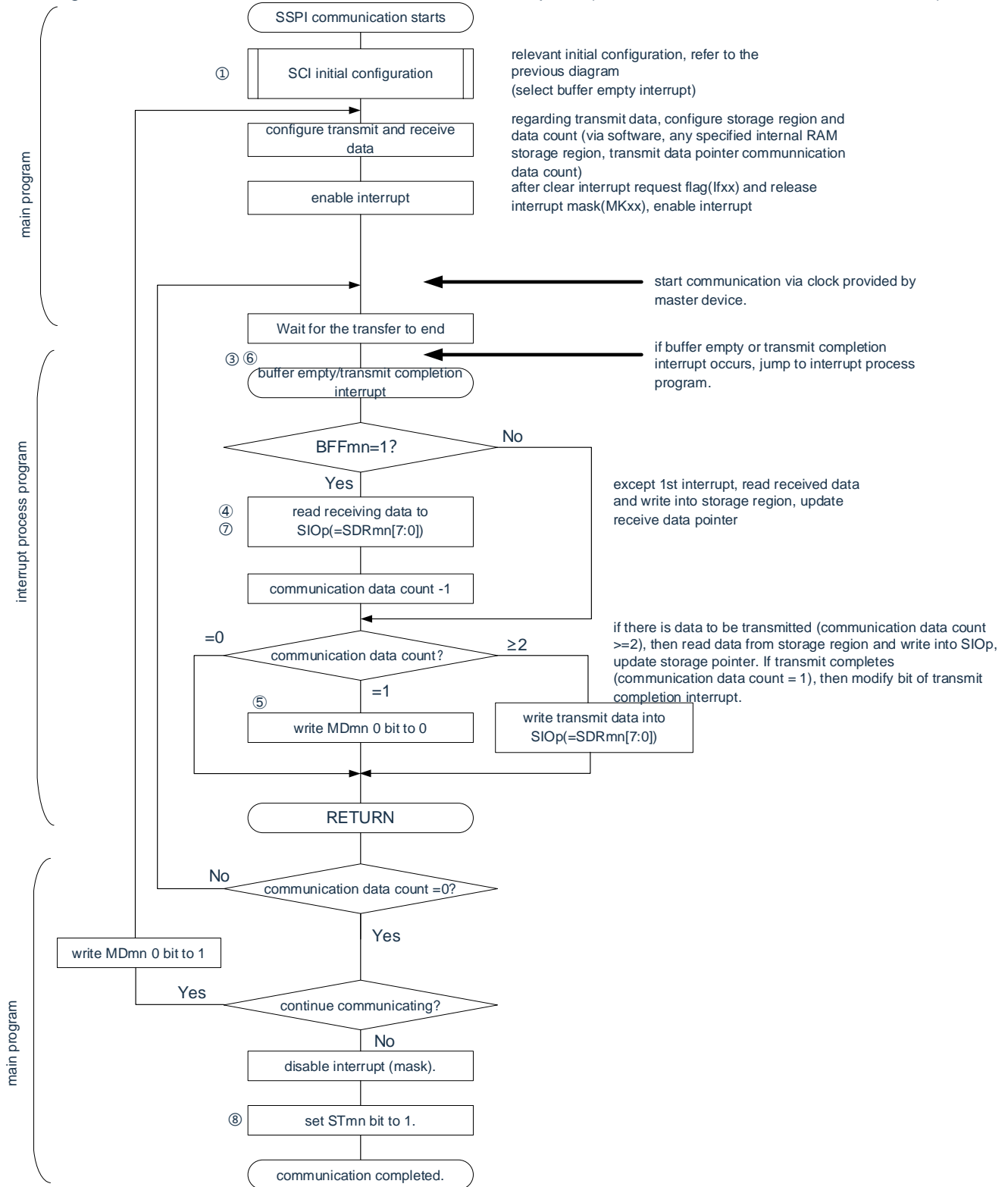
2. If the SDRmn register is read during this period, the transmitted data can be read. At this point, the transfer run is not affected.

Notice The MDmn0 bit of the serial mode register mn (SMRmn) can be overridden even during operation. However, in order to catch up with the end of the transmission interruption of the last transmitted data, it must be overwritten before the last bit of transmission begins.

Remark 1. ① to ⑧ in the diagram correspond to ① to ⑧ in “Figure 14-94 Flowchart of slave transmission and reception (continuous transmit and receive mode)”.

2. m: unit number (m=0) n: channel number (n=0) p: SSPI number (p=00)

Figure 14-94 Flowchart of slave transmission and reception (continuous transmit and receive mode)



Notice Data must be sent to the SIOp register settings before the master device starts the output clock.

Remark 1. ① to ⑧ correspond to ① to ⑧ in “Figure 14-93 Timing diagram of slave transmission and reception (continuous transmit and receive mode)”.

2. m: unit number (m=0) n: channel number (n=0) p: SSPI number (p=00)

14.6.4 Calculation of transfer clock frequency

The transmit clock frequency of the slave select input function (SSPI00) communication can be calculated using the following calculation formula.

(1) Slave device

$(\text{transmit clock frequency}) = \{ \text{Serial clock (SCLK) frequency provided by the master device} \}^{\text{Note}} [\text{Hz}]$

Note The maximum enable transmit clock frequency is $f_{\text{MCK}}/6$.

Remark m: unit number (m=0) n: channel number (n=0) p: SSPI number (p=00)

Table 14-3 Slave select input function operation clock selection

SMR _{mn} register	SPS _m register								Operation clock (f_{MCK}) ^{Note}		
	CKS _{mn}	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00	f_{CLK}	$f_{\text{CLK}}=32\text{MHz}$ in operation
0	X	X	X	X	0	0	0	0	0	f_{CLK}	32MHz
	X	X	X	X	0	0	0	1	1	$f_{\text{CLK}}/2$	16MHz
	X	X	X	X	0	0	1	0	0	$f_{\text{CLK}}/2^2$	8MHz
	X	X	X	X	0	0	1	1	1	$f_{\text{CLK}}/2^3$	4MHz
	X	X	X	X	0	1	0	0	0	$f_{\text{CLK}}/2^4$	2MHz
	X	X	X	X	0	1	0	1	1	$f_{\text{CLK}}/2^5$	1kHz
	X	X	X	X	0	1	1	0	0	$f_{\text{CLK}}/2^6$	500kHz
	X	X	X	X	0	1	1	1	1	$f_{\text{CLK}}/2^7$	250kHz
	X	X	X	X	1	0	0	0	0	$f_{\text{CLK}}/2^8$	125kHz
	X	X	X	X	1	0	0	1	1	$f_{\text{CLK}}/2^9$	62.5kHz
	X	X	X	X	1	0	1	0	0	$f_{\text{CLK}}/2^{10}$	31.25kHz
	X	X	X	X	1	0	1	1	1	$f_{\text{CLK}}/2^{11}$	15.63kHz
	X	X	X	X	1	1	0	0	0	$f_{\text{CLK}}/2^{12}$	7.81kHz
	X	X	X	X	1	1	0	1	1	$f_{\text{CLK}}/2^{13}$	3.91kHz
	X	X	X	X	1	1	1	0	0	$f_{\text{CLK}}/2^{14}$	1.95kHz
X	X	X	X	1	1	1	1	1	$f_{\text{CLK}}/2^{15}$	977Hz	

Note To change the clock selected as f_{CLK} (change the value of the system clock control register (CKC)), you must stop the operation of the universal serial communication unit (SCI) (serial channel stop register m(STm)=000FH) after making the change.

Remark 1. X: Ignore

2. m: unit number (m=0) n: channel number (n=0).

14.6.5 Procedure for handling errors during clock-synchronous serial communication with the slave selection input function

The processing steps when an error occurs during clock-synchronous serial communication that is subordinate to the select input function are shown in Figure 14-95.

Figure 14-95 Handling steps when an overflow error occurs

Software operation	Hardware status	Comments
Read the serial data register mn(SDRMN). →	The BFF m n bit of the SSRm n register is "0" and channel n is acceptable.	This is to prevent overflow errors from ending the next reception during mishandling.
Read the serial status register mn (SSRmn).		The type of error is judged, and the reading value is used to clear the error flag.
Clear the trigger register mn to the serial flag →	Clear the error flag.	By writing the read value of the SSRmn register directly to the SDIRmn register, errors during read operations can only be cleared.

Remark m: unit number (m=0) n: channel number (n=0).

14.7 Operation of UART (UART0~UART2) communication

This is a function that communicates asynchronously through a total of two lines: serial data transmission (TxD) and serial data reception (RxD). Using these two communication lines, data that are transmitted and received asynchronously (using internal baud rate) with other communicating parties in data frames (consisting of start bits, data, parity bits, and stop bits) are used to send and receive data. Full-duplex asynchronous UART communication can be achieved by using two channels, send private (even channels) and receive private (odd channels).

[Transmit and receive data]

- 7-bit, 8-bit or 9-bit data length^{Note}
- MSB/LSB preferred choice
- Level settings for sending and receiving data (choose whether the level is inverted).
- Parity bit appending and parity check functions
- Stop bit appending, and stop bit detection function

[Interrupt function]

- End of transfer interrupt, buffer empty interrupt
- Error interrupts caused by frame errors, parity errors, and overflow errors

[Error detection flag]

- Frame errors, parity errors, overflow errors

Note that only UART0 supports 9-bit data length.

UART0 uses channel 0 and channel 1 of SCI0.

UART1 uses channels 2 and 3 of SCI0.

UART2 uses Channel 0 and Channel 1 of SCI1.

Each channel can choose a function to use, except for the selected function, other functions can not operate.

For example, when UART0 is used for channel 0 and channel 1 of unit 0, SSPI00 and IIC01 cannot be used. However, while using UART0, channels 2 and 3 of different channels can use SSPI10, UART1, or SUDs IIC10.

Note When used as a UART, the sender (even channels) and receivers (odd channels) can only be used for the UART.

UART has the following types of communication operations:

- UART transmission (see 14.7.1).
- UART reception (see 14.7.2).

14.7.1 UART transmission

UART transmission is the operation of the microcontroller of this product to send data asynchronously to other devices.

The even numbered channel of the 2 channels used by the UART is used for UART transmit.

UART	UART0	UART1	UART2
Object channel	Channel 0 of SCI0	Channel 2 of SCI0	Channel 0 of SCI1
Used pin	TxD0	TxD1	TxD2
Interrupt	INTST0	INTST1	INTST2
	You can select either a transmit-end interrupt (single-pass mode) or a buffer null interrupt (continuous transfer mode).		
Error detection flag	not		
Length of transmitted data	7-bit, 8-bit or 9-digit ^{Note 1}		
Transfer rate	Max. $f_{MCK}/6$ [bps](SDRmn[15:9]≥2), Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [bps] ^{Note2}		
Data phase	Normal-phase output (default: high). Inverting output (default: low).		
Parity bits	You can choose from the following: <ul style="list-style-type: none"> • No parity bits. • Appending zero check. • Appending parity. • Appending odd parity. 		
Stop bit	You can choose from the following: <ul style="list-style-type: none"> • Appending 1 bit. • Appending 2 bits. 		
Data direction	MSB first or LSB first		

Note 1. Only UART0 supports 9-bit data length.

2. It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Remark 1. f_{MCK} : Operating clock frequency of the object channel

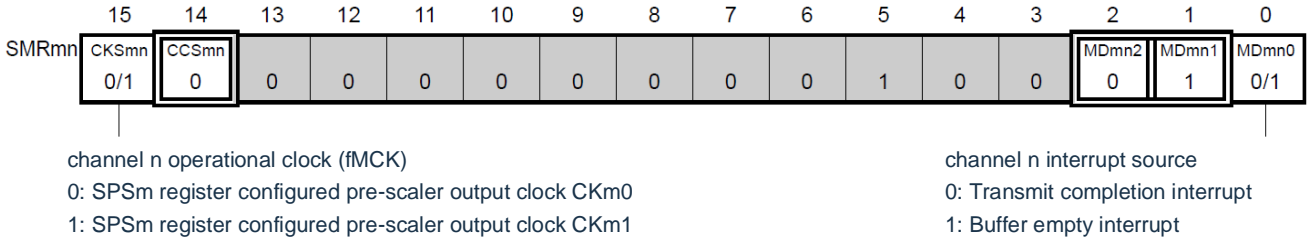
f_{CLK} : System clock frequency

2. m: unit number (m=0, 1) n: channel number (n=0, 2) mn=00, 02, 10.

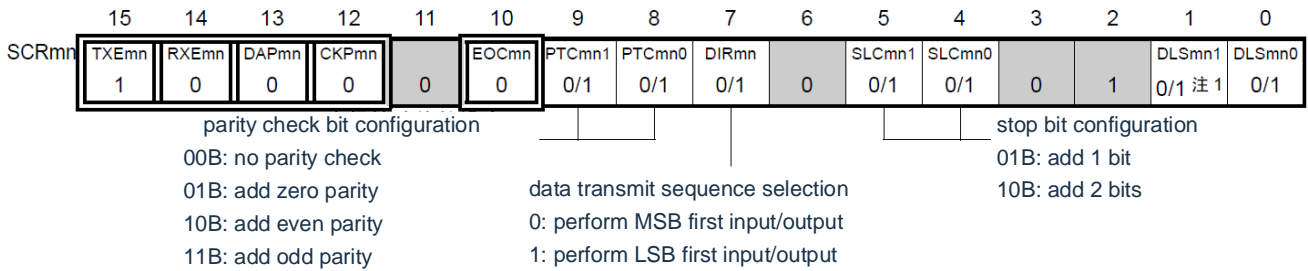
(1) Register setting

Figure 14-96 Example of register settings when UART is sent by UART (UART0~UART2) (1/2)

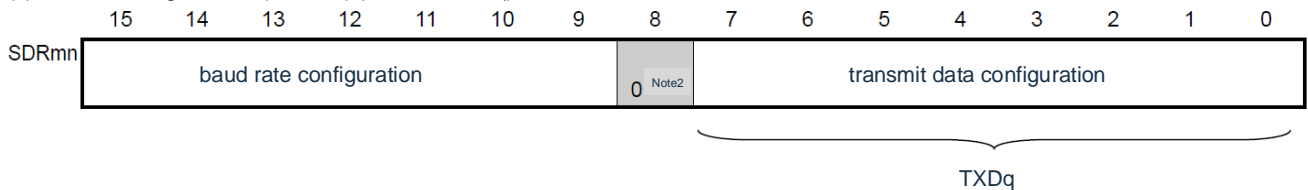
(a) serial mode register mn (SMRmn)



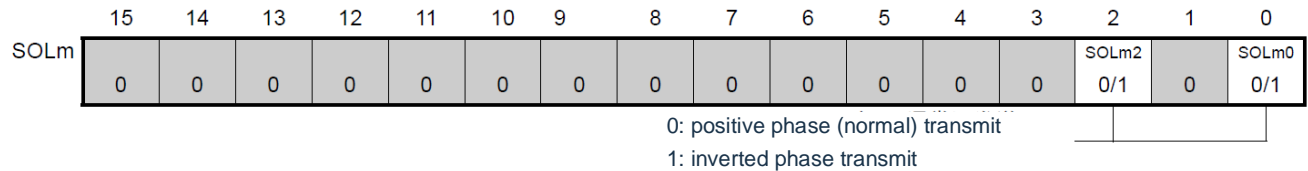
(b) serial communication operation configuration register mn (SCRmn)



(c) serial data register mn (SDRmn) (low 8 bit:TXDq)



(d) serial output voltage register m (SOLm) Only configure bit of target channel.



Note 1 Limited to SCR00 registers, other fixed as "1".

2. When communicating with a length of 9 bits of data, bit0 to 8 of the SDRm0 register is the setting area for sending data. Only UART0 can communicate with 9-bit data lengths.

Remark 1.m: unit number (m=0, 1) n: channel number (n=0, 2) q: UART numbers (q=0~2)mn=00, 02, 10

2. : Fixed in UART send mode. : Cannot be set (initial value).
 x: This is the bit that cannot be used in this mode (set 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 14-96 UART Example of register settings when UART is sent by UART (UART0~UART2) (2/2)

(e) 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	0	0	0	0	CKOm3	CKOm2	CKOm1	CKOm0	0	0	0	0	SOM3	SOM2	SOM1	SOM0
	0	0	0	0	×	×	×	×	0	0	0	0	×	0/1 ^{Note}	×	0/1 ^{Note}

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".

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm	0	0	0	0	0	0	0	0	0	0	0	0	SOEm3	SOEm2	SOEm1	SOEm0
	0	0	0	0	0	0	0	0	0	0	0	0	×	0/1	×	0/1

(g) 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	SSm3	SSm2	SSm1	SSm0
	0	0	0	0	0	0	0	0	0	0	0	0	×	0/1	×	0/1

Note When the SOLmn bit of the corresponding channel is "0", it must be set to "1", and when the SOLmn bit of the corresponding channel is "1", it must be set to "0" before starting transmission. During communication, the value changes depending on the communication data.

Remark1. m: unit number (m=0, 1) n: channel number (n=0, 2)q: UART numbers (q=0~2)mn=00, 02, 10

2. : Fixed in UART transmiy mode. : Cannot be set (initial value).

x: This is the bit that cannot be used in this mode (set 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) Operation steps

Figure 14-97 Initial setup steps for UART transmission

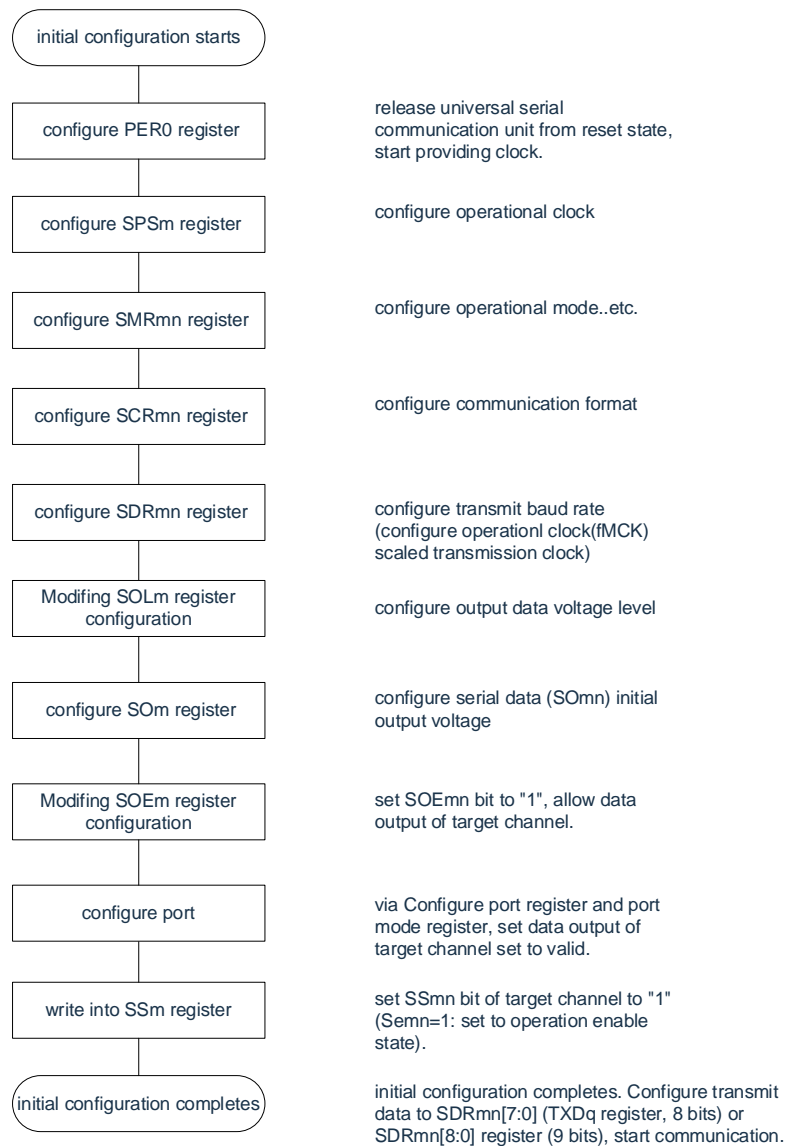


Figure 14-98 Stop steps for UART transmission

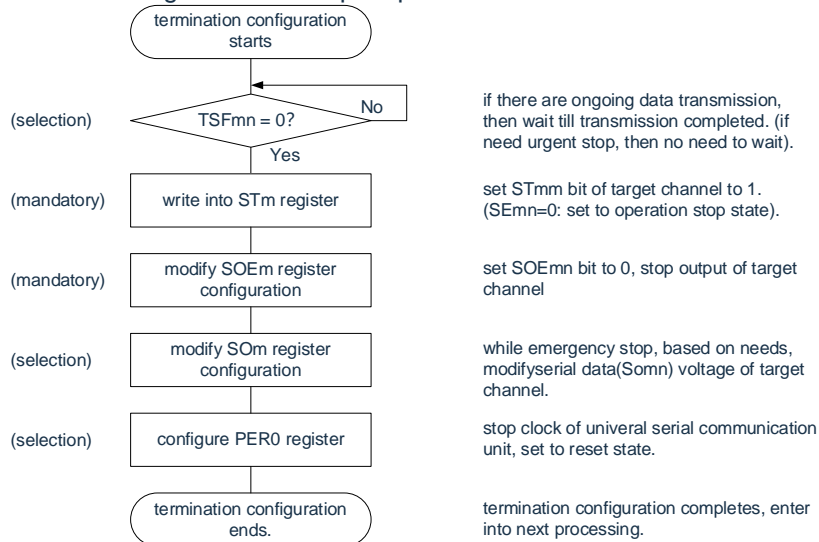
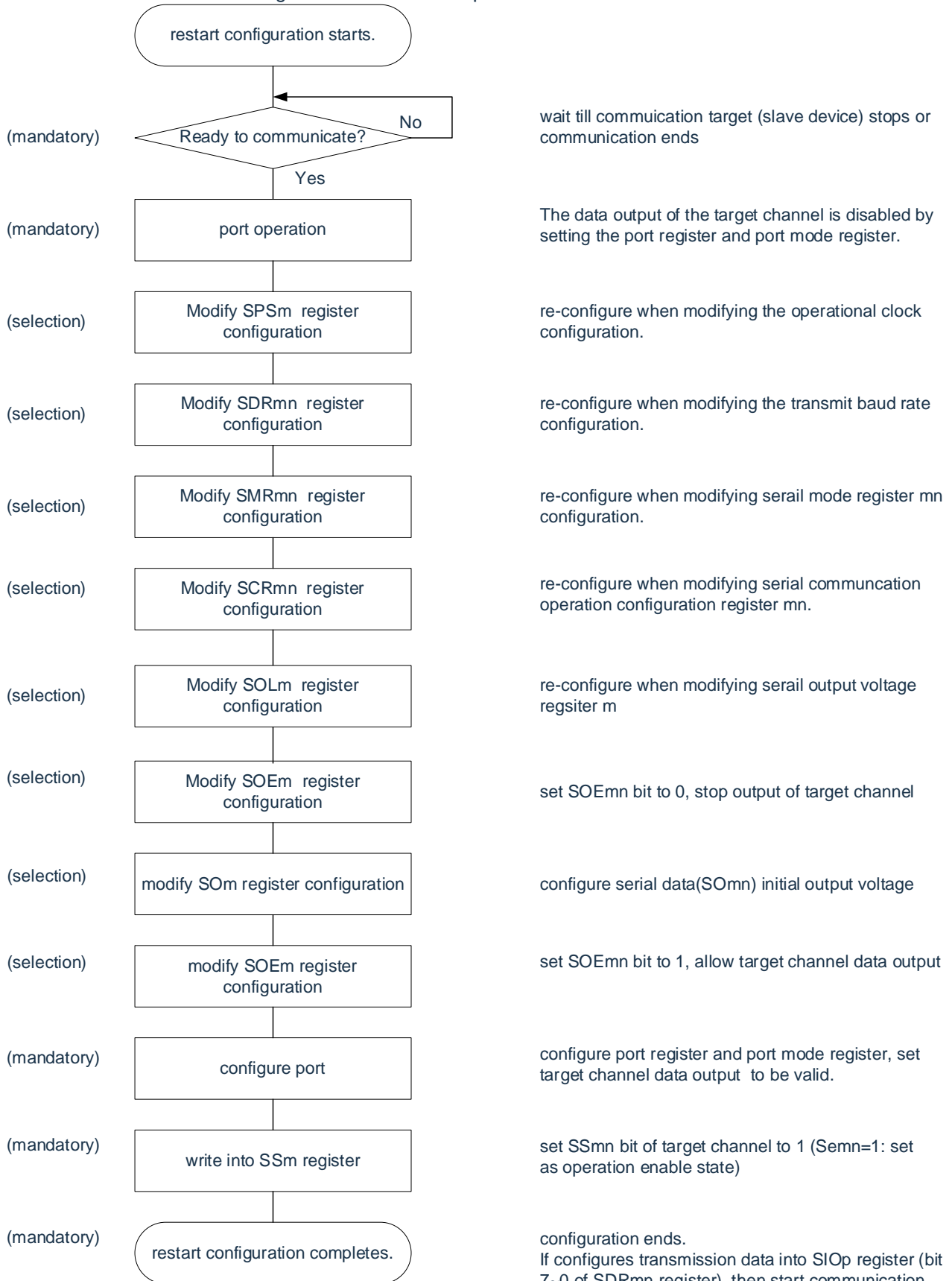


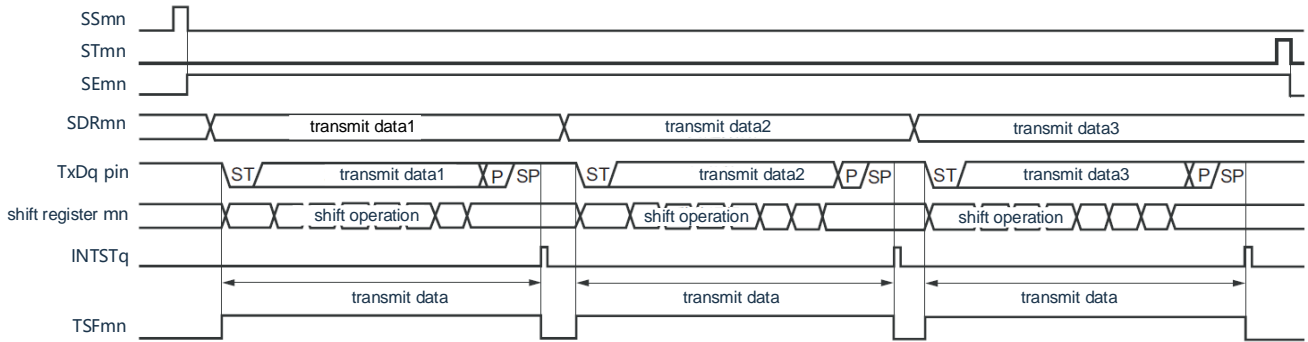
Figure 14-99 Restart steps for UART transmission



Remark If you override PER0 in the abort setting to stop the clock, you must wait until the communication object stops or the communication ends, instead of starting the setting again.

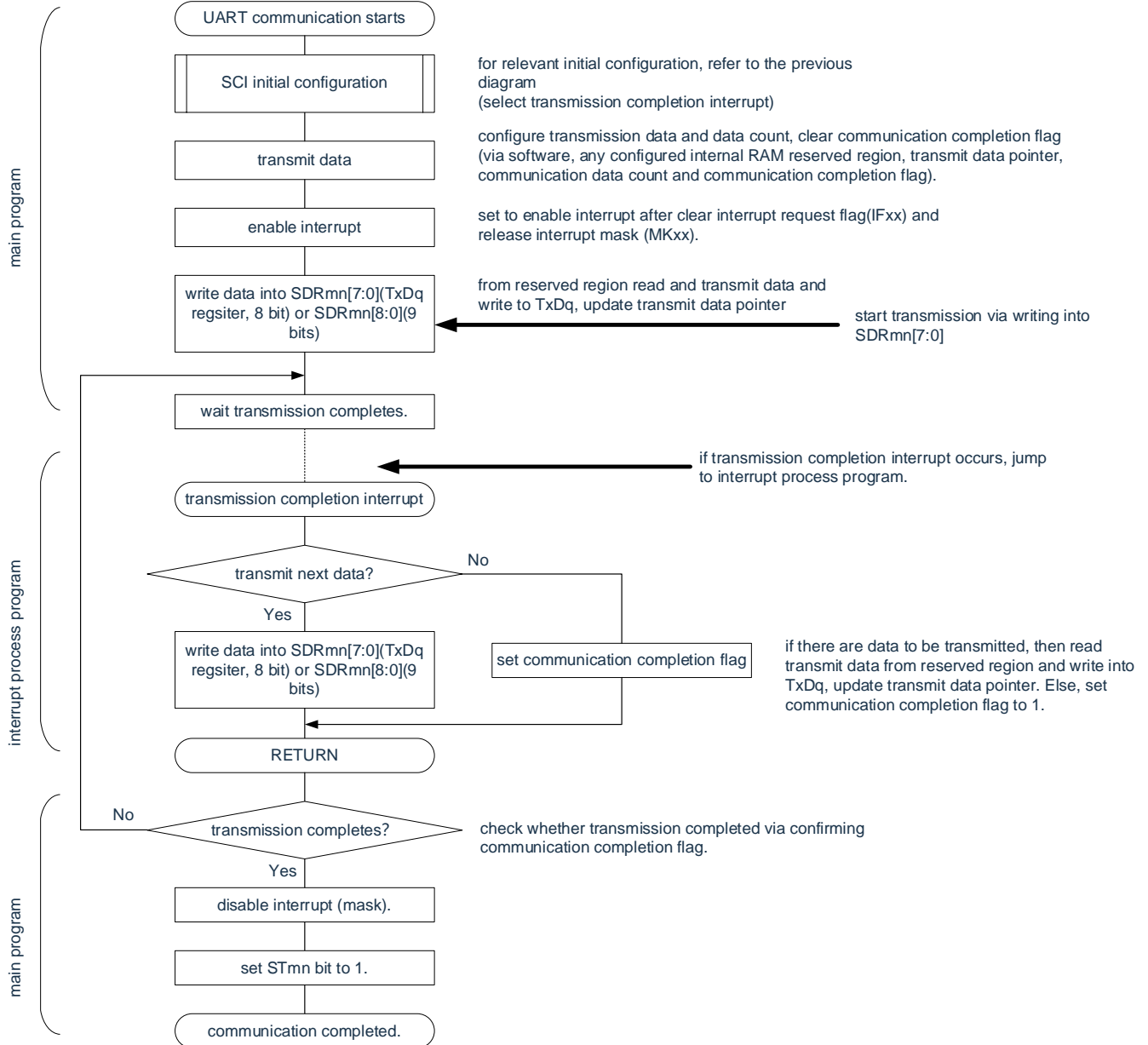
(3) Processing flow (single transmit mode)

Figure 14-100 Timing diagram of UART transmission (single transmit mode)



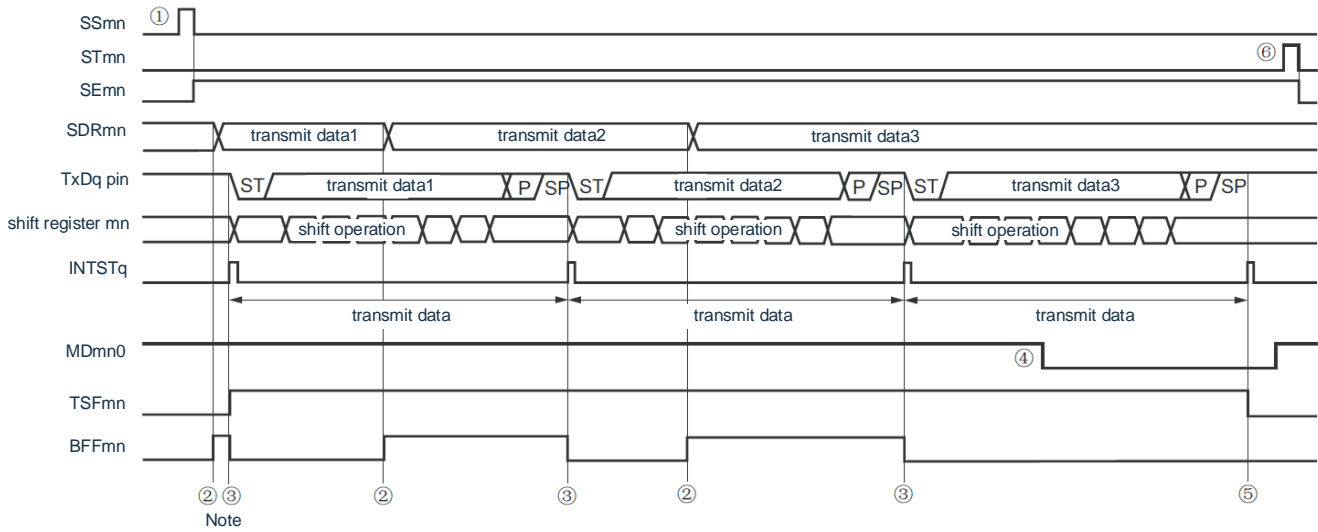
Remark m: unit number (m=0, 1) n: channel number (n=0, 2) q: UART numbers (q=0~2) mn=00, 02, 10

Figure 14-101 Flowchart of UART transmission (single transmit mode)



(4) Processing flow (continuous transmit mode)

Figure 14-102 Timing diagram of UART transmission (continuous transmit mode)

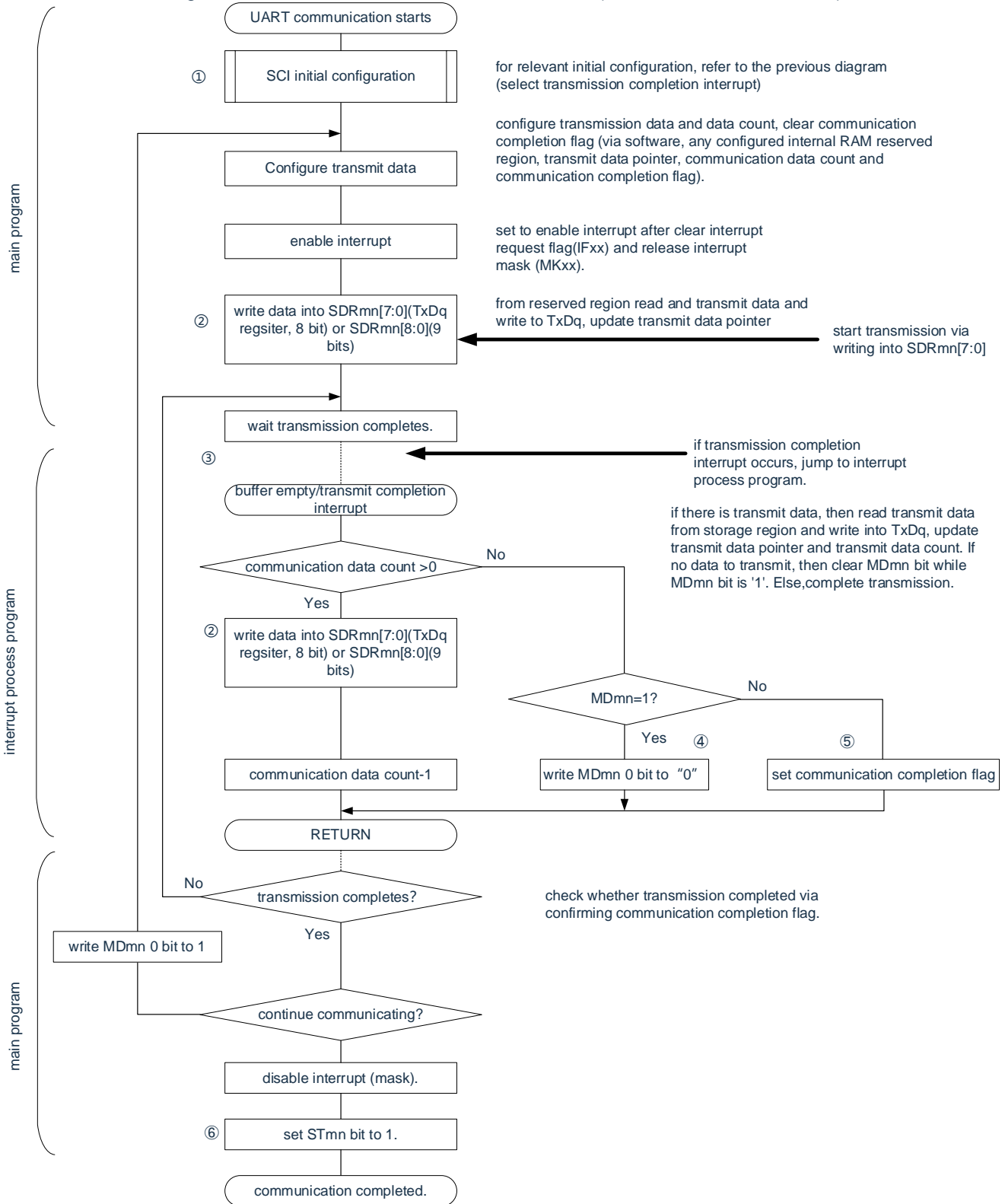


Note If the BFFmn bit of the serial status register mn (SSRmn) is “1” (when valid data is saved in the serial data register mn (SDRmn)) is given The SDRmn register writes the transmitted data and overrides the transmitted data.

Notice The MDmn0 bit of the serial mode register mn (SMRmn) can be overridden even during operation. However, in order to catch up with the end of the transmission interruption of the last transmitted data, it must be overwritten before the last bit of transmission begins.

Remark m: unit number (m=0, 1) n: channel number (n=0, 2) q: UART numbers (q=0~2) mn=00, 02, 10

Figure 14-103 Flowchart of UART transmission (continuous transmit mode)



Remark ① to ⑥ in the figure correspond to ① to ⑥ in "Figure 14-102 Timing diagram of UART transmission (continuous transmit mode)".

14.7.2 UART reception

UART reception is the operation of other devices of this product's microcontroller to receive data asynchronously.

The odd channel of the two channels used for UART is used for UART reception. However, it is necessary to set the SMR registers for the odd and even channels.

UART	UART0	UART1	UART2
Object channel	Channel 1 of SCI0	Channel 3 of SCI0	Channel 1 of SCI1
Used pin	RxD0	RxD1	RxD2
Interrupt	INTSR0	INTSR1	INTSR2
	Limited to transmit complete interrupts (disable setting buffer empty interrupts).		
Error interrupt	INTSRE0	INTSRE1	INTSRE2
Error detection flag	<ul style="list-style-type: none"> • Frame Error Detection Flag (FEFmn). • Parity Error Detection Flag (PEFmn). • Overflow Error Detection Flag (OVFmn). 		
Transmitted data length	7-bit, 8-bit or 9-digit ^{Note 1}		
Transfer rate	Max. $f_{MCK}/6$ [bps](SDRmn[15:9]≥2), Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [bps]		
Data phase	Normal-phase output (default: high). Inverting output (default: low).		
Parity bit	Select from the following: <ul style="list-style-type: none"> • No parity bits (no parity). • Appending zero check (no parity). • Even-check • Odd check 		
Stop bit	Appending 1 bit.		
Data direction	MSB first or LSB first		

Note 1. Only UART0 supports 9-bit data length.

2. Must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

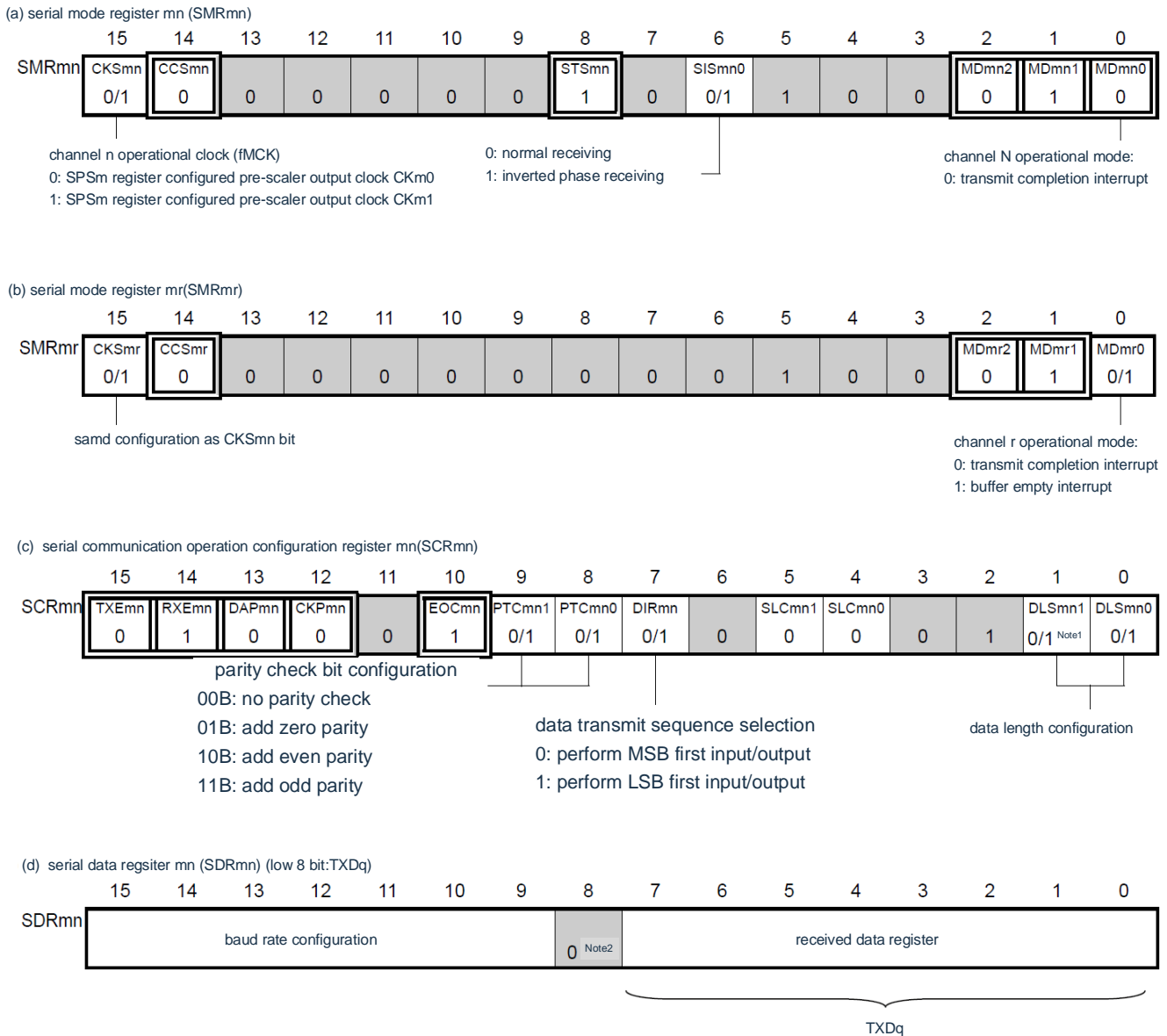
Remark 1. f_{MCK} : The operating clock frequency of the object channel

f_{CLK} : System clock frequency

2. m: Unit number (m=0, 1) n: channel number (n=1, 3) mn=01, 03, 11.

(1) Register setting

Figure 14-104 Example of register settings when UART is received by UART (UART0~UART2) (1/2)



Note 1 Limited to SCR01 registers, other fixed as "1".

2. When communicating with a length of 9 bits of data, bit0 to 8 of the SDRm1 register is the setting area for sending data. Only UART0 can communicate with 9-bit data lengths.

Notice When the UART is received, the SMRmr register of channel r paired with channel n must also be set.

Remark 1. m: unit number (m=0, 1) n: channel number (n=1, 3) mn=01, 03, 11.

r: Channel number (r=n-1) q: UART number (q=0~2)

2

--

 : Fixed in UART receive mode.

--

 : Cannot be set (initial value).

x: This is the bit that cannot be used in this mode (set 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 14-104 Example of register settings when UART is received by UART (UART0~UART2) (2/2)

(e) serial output register m (SOm)... Not used in this mode.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOm	0	0	0	0	CKOm3	CKOm2	CKOm1	CKOm0	0	0	0	0	SOm3	SOm2	SOm1	SOm0
	0	0	0	0	×	×	×	×	0	0	0	0	×	×	×	×

(f) 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	0	0	0	0	0	0	0	0	0	0	0	0	SOEm3	SOEm2	SOEm1	SOEm0
	0	0	0	0	0	0	0	0	0	0	0	0	×	×	×	×

(g) 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	SSm3	SSm2	SSm1	SSm0
	0	0	0	0	0	0	0	0	0	0	0	0	0/1	×	0/1	×

Remark 1. m: Unit number (m=0, 1).

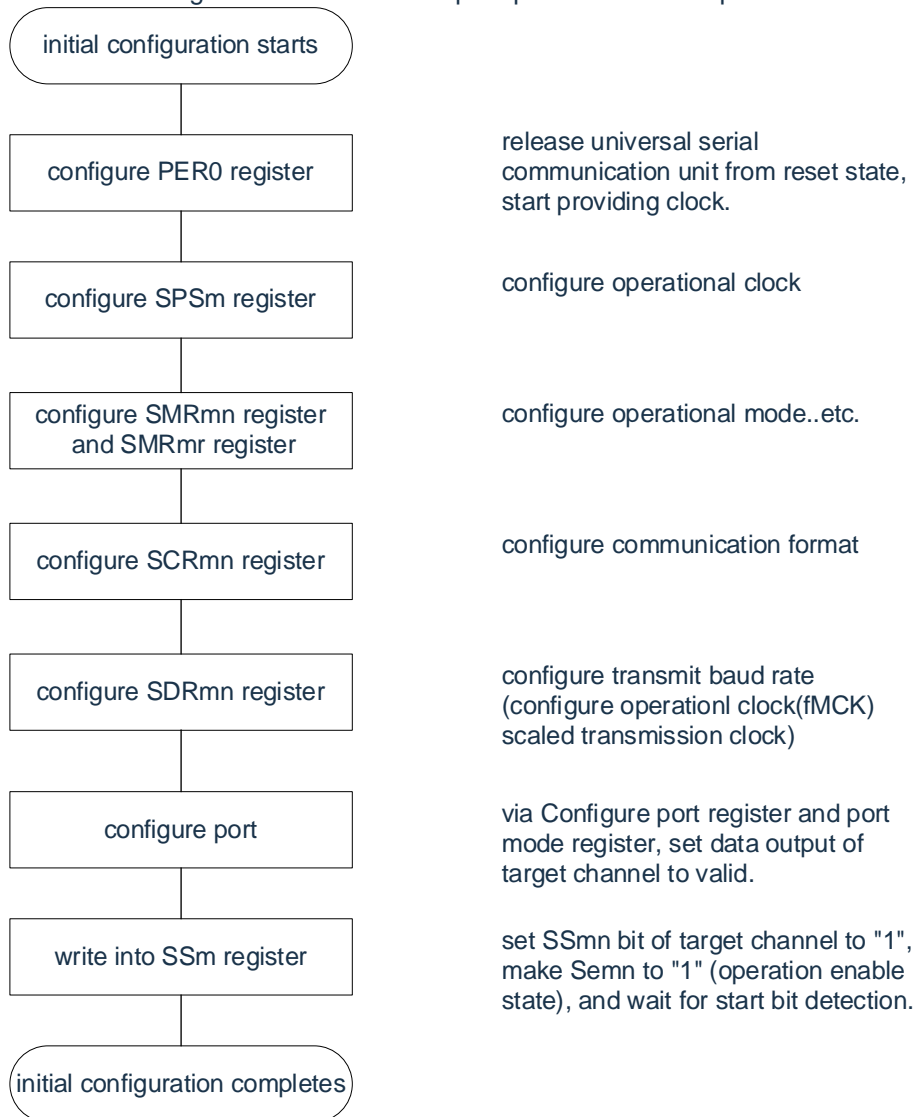
2□ : Fixed in UART receive mode. □ : Cannot be set (initial value).

x: This is the bit that cannot be used in this mode (set 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) Operation steps

Figure 14-105 Initial setup 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 14-106 Stop steps for UART reception

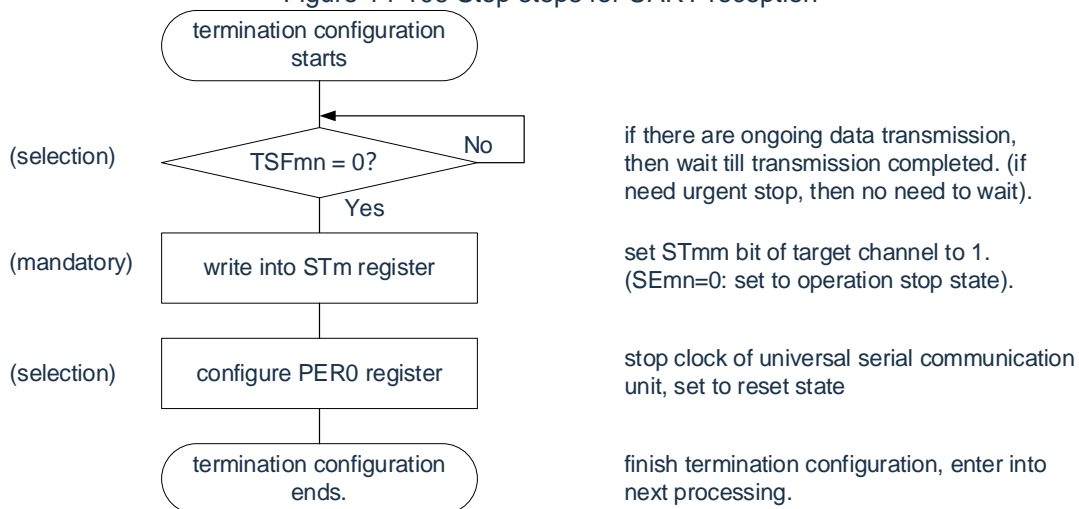
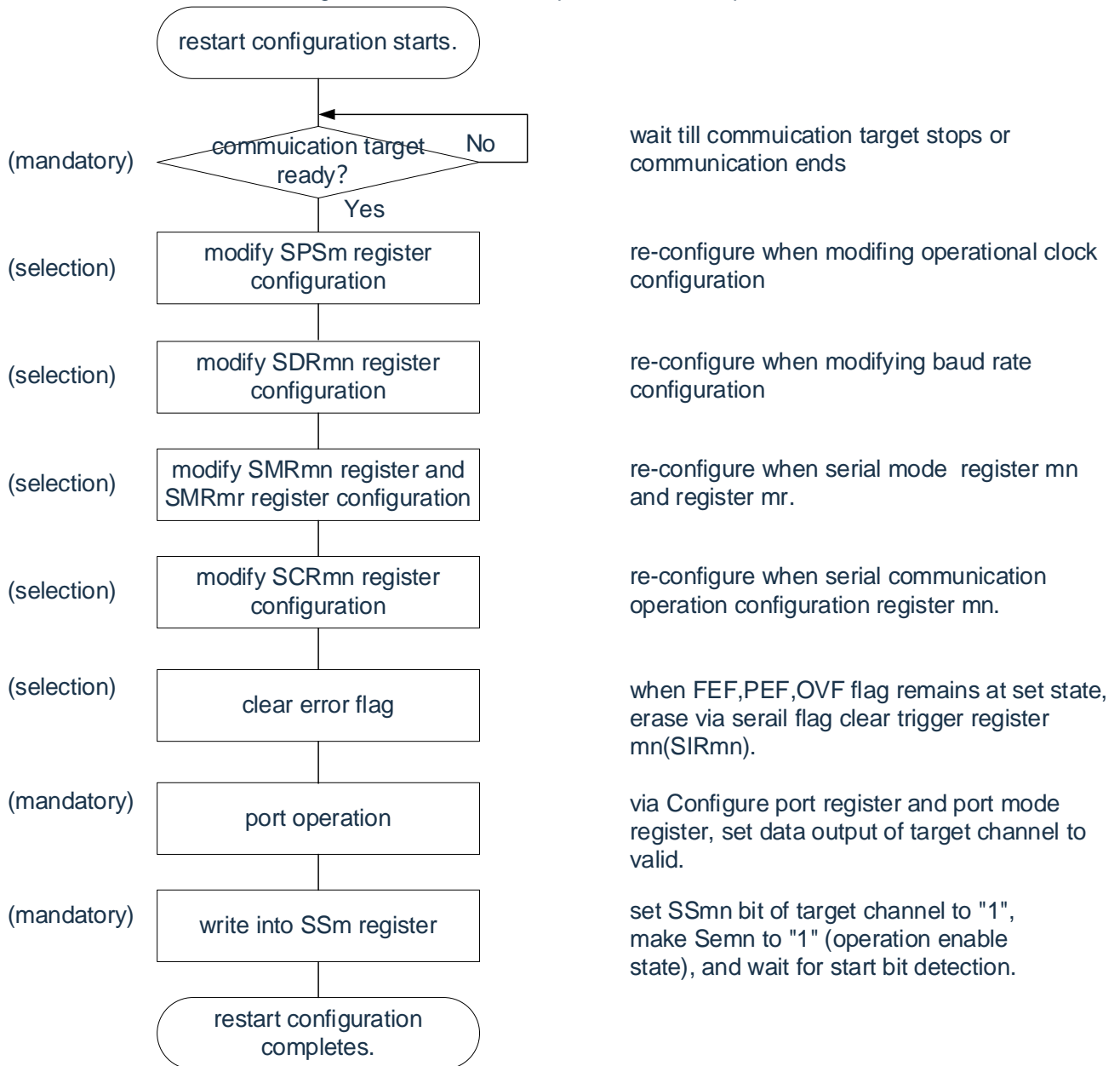


Figure 14-107 Restart steps for UART reception

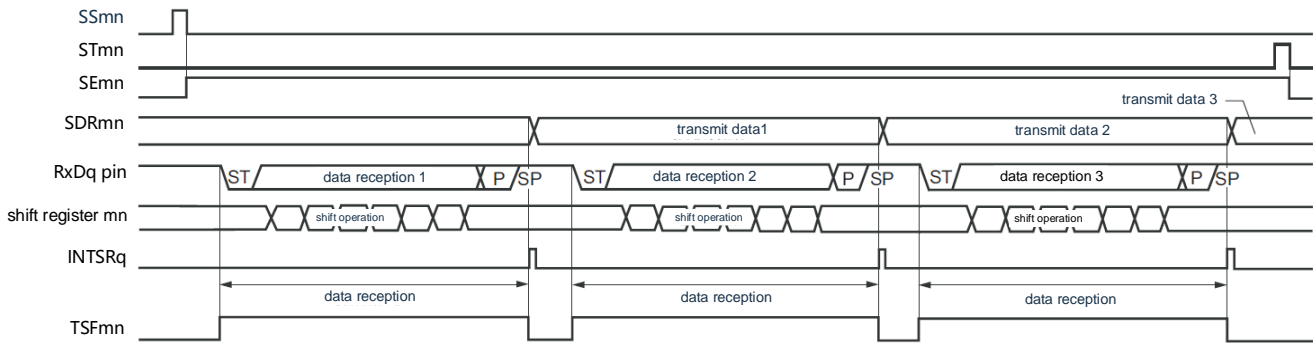


Notice At least 4 F_{MCK} clocks must be spaced after setting the RXEmn bit of the SCRmn register to "1" and then set the SSmn bit to "1".

Remark If you override PER0 in the abort setting to stop the clock, you must wait until the communication object stops or the communication ends, instead of starting the setting again.

(3) Processing flow

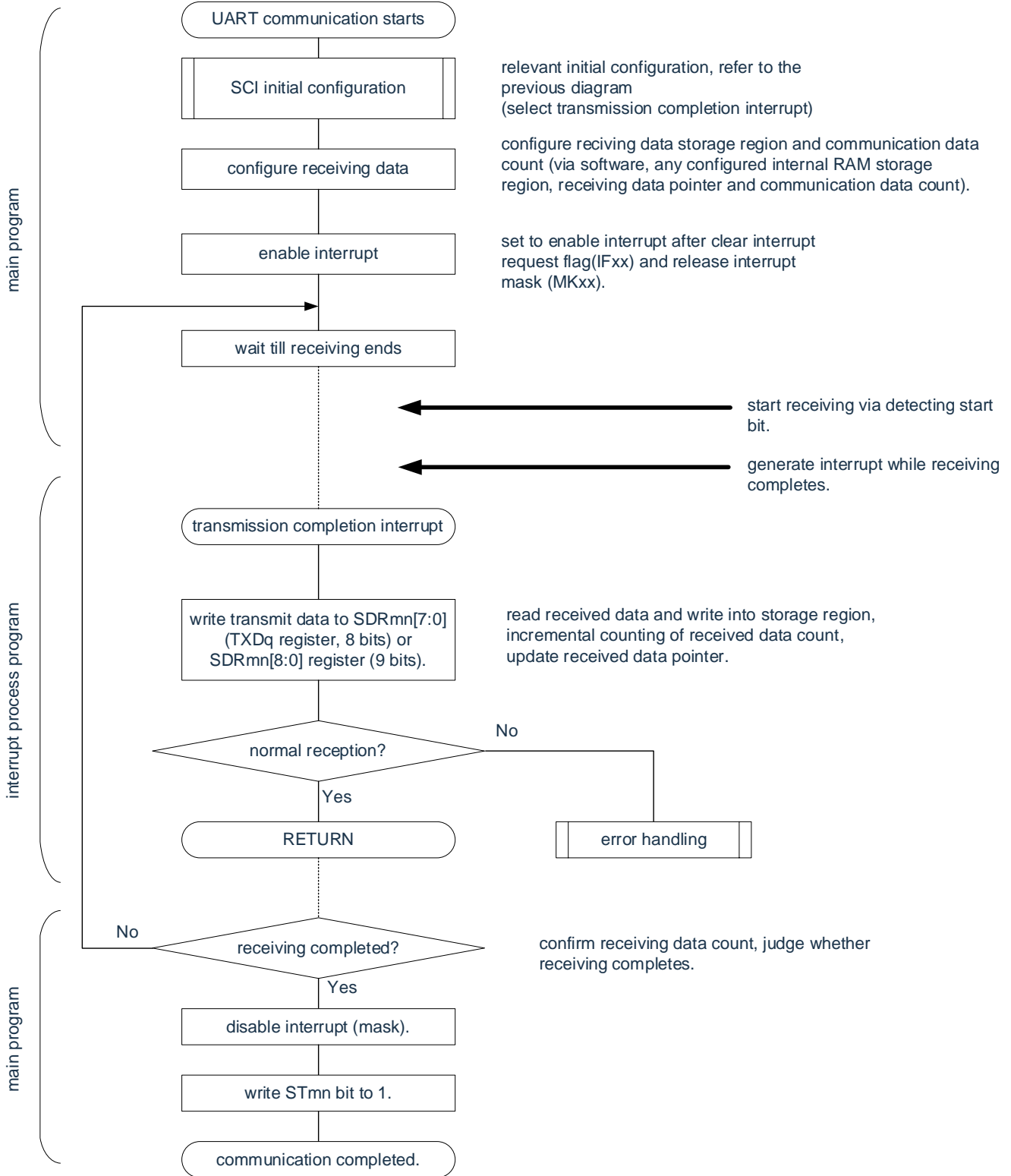
Figure 14-108 Timing diagram of UART reception



Remark m: unit number (m=0, 1) n: channel number (n=1, 3) mn=01, 03, 11.

r: Channel number (r=n-1) q: UART number (q=0~2)

Figure 14-109 Flowchart of UART reception



14.7.3 Calculation of baud rate

(1) Calculating the baud rate

The baud rate of UART (UART0~UART2) communication can be calculated using the following formula:

$$(\text{baud rate}) = \{\text{clock frequency of the object channel (f}_{\text{MCK}})\} \div (\text{SDRmn}[15:9] + 1) \div 2 [\text{bps}]$$

Notice The SDRmn [15:9] of the serial data register mn (SDRmn) is disabled from being set to “0000000B” and “0000001B”.

Remark 1. Because the value of SDRmn [15:9] when using UART is the value of bit15~9 of the SDRmn register (0000010B~1111111B), so 2~127.

2. m: unit number (m=0, 1) n: channel number (n=0~2). mn=00~03, 10~11.

The operating clock (f_{MCK}) depends on the serial clock select register m (SPSm) and bit 15 (CKSmn bit) of the serial mode register mn (SMRmn).

Table 14-4 Selection of UART operating 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}	$f_{CLK}=32\text{MHz}$ in operation
0	X	X	X	X	0	0	0	0	0	f_{CLK}	32MHz
	X	X	X	X	0	0	0	1	1	$f_{CLK}/2$	16MHz
	X	X	X	X	0	0	1	0	0	$f_{CLK}/2^2$	8MHz
	X	X	X	X	0	0	1	1	1	$f_{CLK}/2^3$	4MHz
	X	X	X	X	0	1	0	0	0	$f_{CLK}/2^4$	2MHz
	X	X	X	X	0	1	0	1	1	$f_{CLK}/2^5$	1MHz
	X	X	X	X	0	1	1	0	0	$f_{CLK}/2^6$	500kHz
	X	X	X	X	0	1	1	1	1	$f_{CLK}/2^7$	250kHz
	X	X	X	X	1	0	0	0	0	$f_{CLK}/2^8$	125kHz
	X	X	X	X	1	0	0	1	1	$f_{CLK}/2^9$	62.5kHz
	X	X	X	X	1	0	1	0	0	$f_{CLK}/2^{10}$	31.25kHz
	X	X	X	X	1	0	1	1	1	$f_{CLK}/2^{11}$	15.63kHz
	X	X	X	X	1	1	0	0	0	$f_{CLK}/2^{12}$	7.81kHz
	X	X	X	X	1	1	0	1	1	$f_{CLK}/2^{13}$	3.91kHz
	X	X	X	X	1	1	1	0	0	$f_{CLK}/2^{14}$	1.95kHz
X	X	X	X	1	1	1	1	1	$f_{CLK}/2^{15}$	977Hz	
1	0	0	0	0	X	X	X	X	X	f_{CLK}	32MHz
	0	0	0	1	X	X	X	X	X	$f_{CLK}/2$	16MHz
	0	0	1	0	X	X	X	X	X	$f_{CLK}/2^2$	8MHz
	0	0	1	1	X	X	X	X	X	$f_{CLK}/2^3$	4MHz
	0	1	0	0	X	X	X	X	X	$f_{CLK}/2^4$	2MHz
	0	1	0	1	X	X	X	X	X	$f_{CLK}/2^5$	1MHz
	0	1	1	0	X	X	X	X	X	$f_{CLK}/2^6$	500kHz
	0	1	1	1	X	X	X	X	X	$f_{CLK}/2^7$	250kHz
	1	0	0	0	X	X	X	X	X	$f_{CLK}/2^8$	125kHz
	1	0	0	1	X	X	X	X	X	$f_{CLK}/2^9$	62.5kHz
	1	0	1	0	X	X	X	X	X	$f_{CLK}/2^{10}$	31.25kHz
	1	0	1	1	X	X	X	X	X	$f_{CLK}/2^{11}$	15.63kHz
	1	1	0	0	X	X	X	X	X	$f_{CLK}/2^{12}$	7.81kHz
	1	1	0	1	X	X	X	X	X	$f_{CLK}/2^{13}$	3.91kHz
	1	1	1	0	X	X	X	X	X	$f_{CLK}/2^{14}$	1.95kHz
1	1	1	1	X	X	X	X	X	$f_{CLK}/2^{15}$	977Hz	

Note To change the clock selected as f_{CLK} (change the value of the system clock control register (CKC)), you must stop the operation of the universal serial communication unit (SCI) (serial channel stop register m(STm)=000FH) after making the change.

Remark 1. X: Ignore

2. m: unit number (m=0, 1) n: channel number (n=0~2) mn=00~03, 10~11.

(2) Baud rate error during transmission

The baud rate error when transmitting UART (UART0 to UART2) communication can be calculated using the following formula, and the baud rate of the transmitter must be set within the baud rate tolerance of the receiver.

$$(\text{Baud rate error}) = (\text{calculated value of baud rate}) \div (\text{value of target baud rate}) \times 100 - 100[\%]$$

An example of setting the UART baud rate at $f_{\text{CLK}}=32\text{MHz}$ is shown below.

UART baud rate (Target baud rate)	$f_{\text{CLK}}=32\text{MHz}$			
	Running clock (f_{MCK})	SDRmn[15:9]	Calculated value of the baud rate	Error with target baud rate
300bps	$f_{\text{CLK}}/2^9$	103	300.48bps	+0.16%
600bps	$f_{\text{CLK}}/2^8$	103	600.96bps	+0.16%
1200bps	$f_{\text{CLK}}/2^7$	103	1201.92bps	+0.16%
2400bps	$f_{\text{CLK}}/2^6$	103	2403.85bps	+0.16%
4800bps	$f_{\text{CLK}}/2^5$	103	4807.69bps	+0.16%
9600bps	$f_{\text{CLK}}/2^4$	103	9615.38bps	+0.16%
19200bps	$f_{\text{CLK}}/2^3$	103	19230.8bps	+0.16%
31250bps	$f_{\text{CLK}}/2^3$	63	31250.0bps	±0.0%
38400bps	$f_{\text{CLK}}/2^2$	103	38461.5bps	+0.16%
76800bps	$f_{\text{CLK}}/2$	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, 2) mn=00, 02, 10.

(3) Baud rate tolerance range for reception

The baud rate tolerance range for UART (UART0 to UART2) communication can be calculated using the following formula, and the baud rate of the transmitter must be set within the baud rate tolerance range of the receiver.

$$\text{(Max. baud rate that can be received)} = \frac{2 \times k \times \text{Nfr}}{2 \times k \times \text{Nfr} - k + 2} \times \text{Brate}$$

$$\text{(Min. baud rate that can be received)} = \frac{2 \times k \times (\text{Nfr} - 1)}{2 \times k \times \text{Nfr} - k - 2} \times \text{Brate}$$

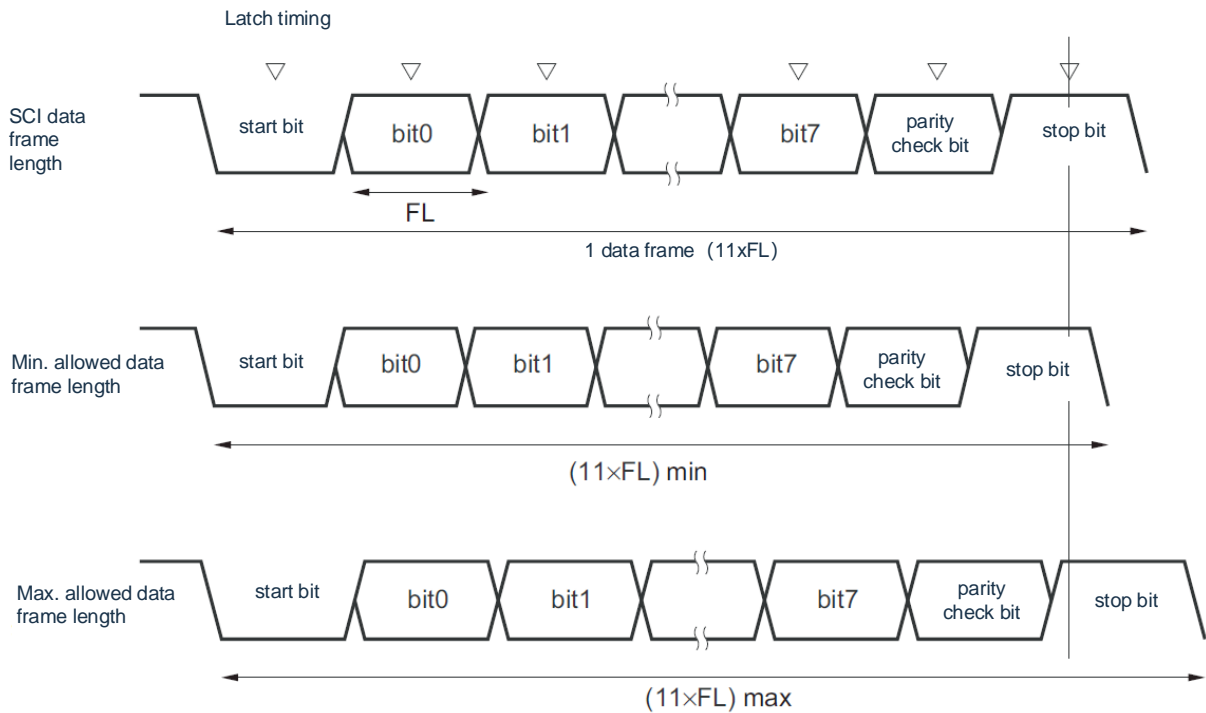
Brate: Calculated baud rate of the receiver (refer to “14.7.4 (1) Baud rate calculation formula”).

k : SDRmn[15:9]+1

Nfr : 1 frame length of data [bit]
 = (start bit) + (data length) + (parity bit) + (stop bit)

Remark m: unit number (m=0, 1) n: channel number (n=1, 3) mn=01, 03, 11

Figure 14-110 Baud rate tolerance range for reception (1 data frame length = 11 bits)



As shown in Figure 14-110 after the start bit is detected, the latch timing of the received data depends on the divider ratio set by bit15 to 9 of the serial data register mn (SDRmn). If the last data (stop bit) can catch up with this latch timing, it can be received normally.

14.7.4 Handling steps when an error occurs during UART (UART0~UART 2) communication

The handling steps when an error occurs during UART (UART0~UART 2) communication are shown in Figure 14-111 and Figure 14-112.

Figure 14-111 Processing steps when a parity error or overflow error occurs

Software operation	Hardware status	Remark
Read the serial data register mn (SDRmn).	→ The BFFmn bit of the SSRmn register is "0" and channel n is receiverable.	This is to prevent overflow errors from occurring when the next receive ends during error handling.
Read the serial status register mn (SSRmn).		Determine the type of error, and read the value to clear the error marker.
Clear the trigger register mn to the serial flag (SDIRmn) writes "1".	→ Clear the error flag.	By writing the read value of the SSRmn register directly to the SDIRmn register, errors during read operations can only be cleared.

Figure 14-112 Processing steps when a frame error occurs

Software operation	Hardware status	Remark
Read the serial data register mn(SDRMN).	→ The BFF m n bit of the SSRm n register is "0" and channel n is acceptable.	This is to prevent overflow errors from ending the next reception during mishandling.
Read the serial status register mn(SSRmn).		Determine the error category, and read the value to remove the error marker.
Write the serial flag to clear the trigger register mn (SIRmn).	→ Clear the error flag.	By writing the read value of the SSRmn register directly to the SDIRmn register, errors during read operations can only be cleared.
Set the STmn bit of the serial channel stop register m (STm) to "1".	→ The serial channel allows the Without n bit of status register m (Herself m) to be "0" and channel n is the running stop state.	
Synchronize processing with the communicating party.		Because the start bit is offset, a frame error can be considered to have occurred. Therefore, it is necessary to re-synchronize with the communicating party and restart the communication.
Set the SSmn bit of the serial channel start register m (SSm) to "1".	→ The serial channel allows the SE m n bit of status register m (Herself m) to be "1" and channel n to be operational.	

Remark m: unit number (m=0, 1) n: channel number (n=0~3) mn=00~ 03, 10~11.

14.8 Operation of LIN communication

14.8.1 LIN transmission

In UART sending, UART0 supports LIN communication.

The LIN transmission uses channel 0 of unit 0.

UART	UART0	UART1	UART2
LIN communication support	Yes	No	No
Object channel	Channel 0 of SCI0	—	—
Used pin	TxD0	—	—
Interrupt	INTST0	—	—
	Selectable transmission end interrupt (single transmission mode) or buffer air interrupt (continuous transmission mode).		
Error detection flag	Not		
Transmitted data length	8 bits		
Transfer rate ^{Note}	Max. $f_{MCK}/6$ [bps](SDR00[15:9] ≥ 2), Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [bps]		
Data phase	Positive phase output (default: high). Negative phase output (default: low).		
Parity bit	No parity bits.		
Stop bit	Appending 1 bit.		
Data direction	LSB first		

Note It must be used within the range of peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet), and 2.4/9.6/19.2kbps are often used in LIN communication.

Remark f_{MCK} : Operation clock frequency of the object channel
 f_{CLK} : System clock frequency

LIN is short for Local Interconnect Network and is a low-speed (1 to 20kbps) serial communication protocol to reduce the cost of automotive networks. LIN communication is a single master communication, a master device can connect up to 15 slave devices.

LIN slave devices are used for the control of switches, transmissions, sensors, etc., which are connected to the master control device via LIN.

The LIN master is generally connected to networks such as the Controller Area Network.

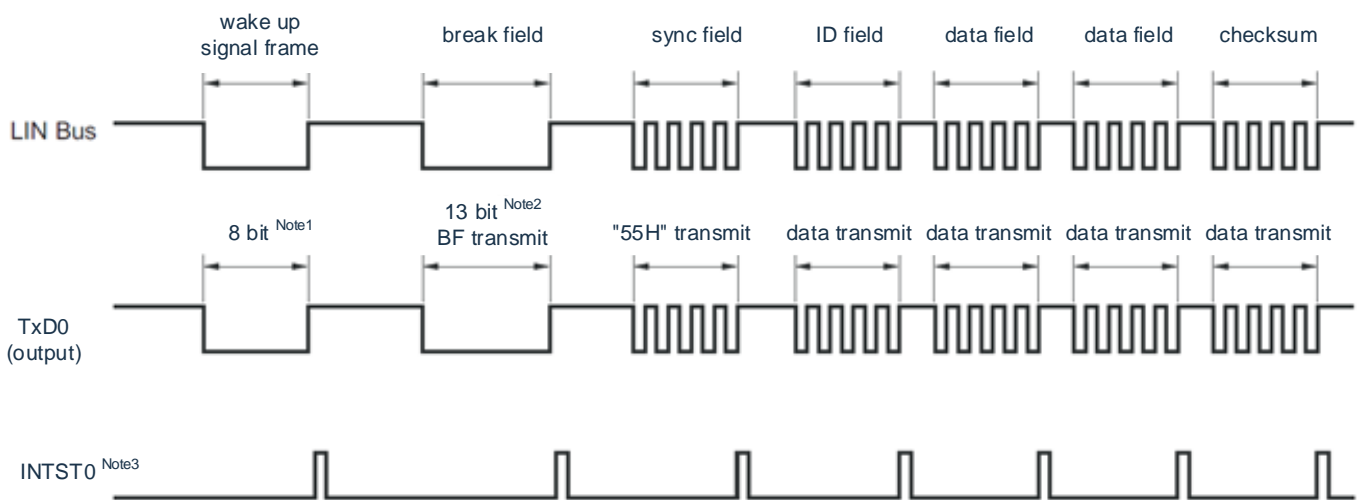
The LIN bus is a single-wire bus that connects nodes through an ISO9141-compliant transceiver.

According to the LIN protocol, the master device sends a frame with additional baud rate information, and the slave device receives this frame and corrects the baud rate error with the master control device.

Therefore, if the baud rate error of the slave device is not greater than ±15%, communication can be made.

A summary of the LIN's transmit operation is shown in Figure 14-113.

Figure 14-113 LIN transmission operation



Note 1. In order to meet the requirements of the wake-up signal, the baud rate is set and the corresponding data is transmitted by "80H".

- 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]. The baud rate used for the break field is as follows:

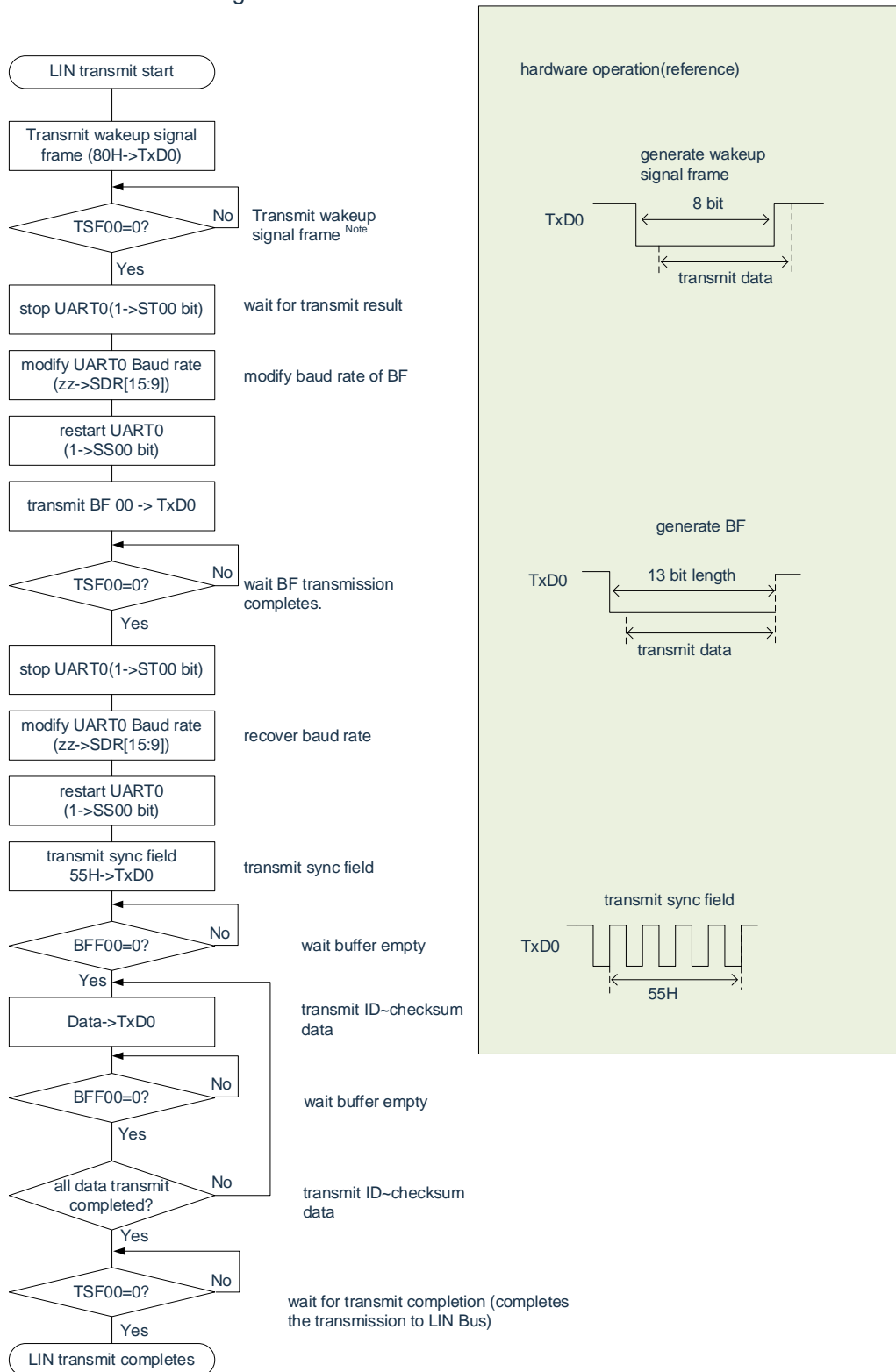
$$\text{(Baud rate for break field)} = 9/13 \times N$$

Transmit the data of "00H" through this baud rate to generate a break field.

- 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 14-114 Flowchart of LIN transmission



Note It is limit to situations starting from LIN-bus sleep.

Remark This is the process that starts by ending the initial set-up of the UART and allowing slave transmission.

14.8.2 LIN reception

In UART reception, UART0 supports LIN communication.

The LIN reception uses channel 1 of unit 0.

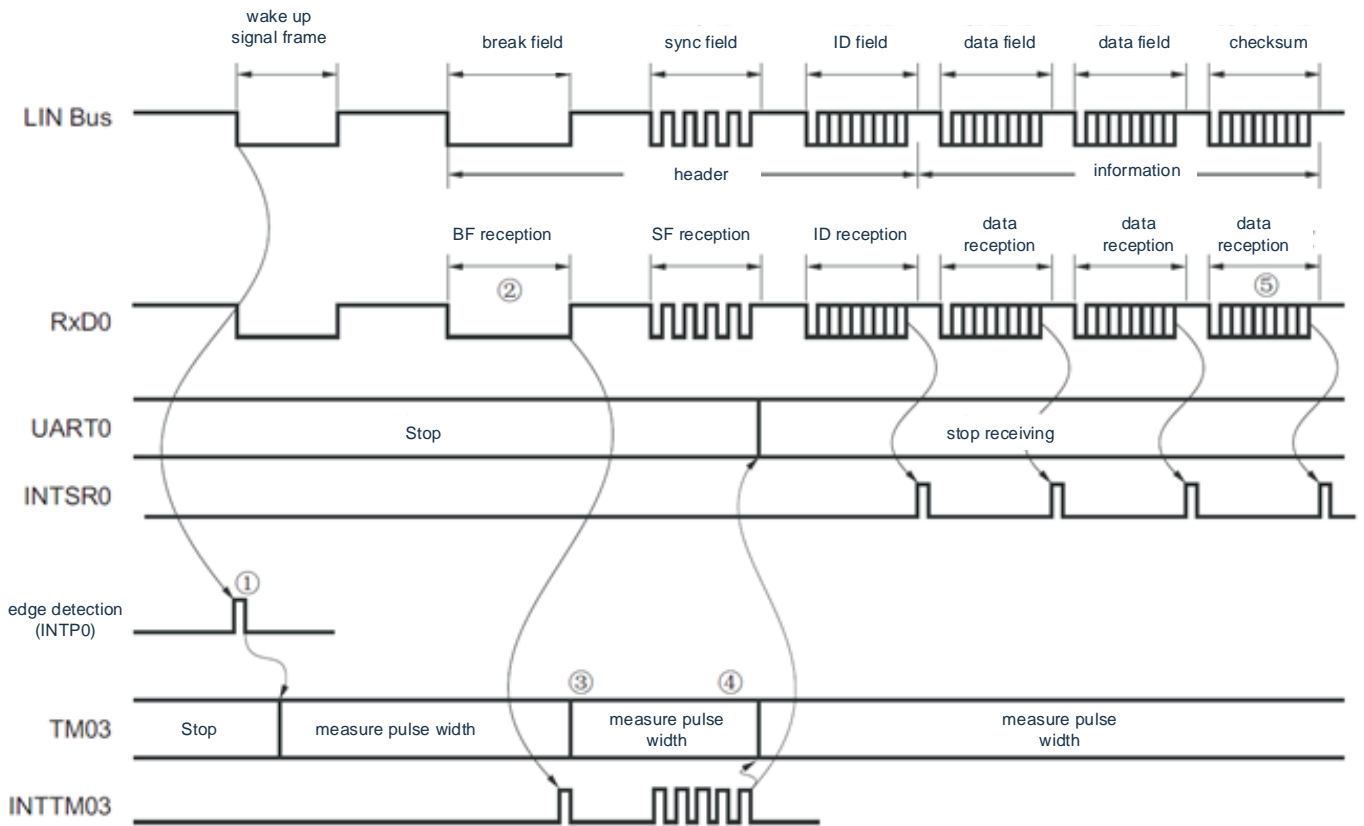
UART	UART0	UART1	UART2	UART3
LIN communication support	Yes	No	No	No
Object channel	Channel 1 of SCI0	—	—	—
Used pin	RxD0	—	—	—
Interrupt	INTSR0	—	—	—
	Limited to transmit complete interrupts (disable setting buffer empty interrupts).			
Error interrupt	INTSRE0	—	—	—
Error detection flag	<ul style="list-style-type: none"> • Frame error detection flag (FEF01). • Overflow error detection flag (OVF01). 			
Transmitted data length	8 bits			
Transfer rate ^{Note}	Max. $f_{MCK}/6$ [bps](SDR01[15:9]≥2), Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [bps]			
Data phase	Normal-phase output (default: high). Inverting output (default: low).			
Parity bit	No parity bits (no parity).			
Stop bit	Appending 1 bit.			
Data direction	LSB first			

Note It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Remark f_{MCK} : Operation clock frequency of the object channel
 f_{CLK} : System clock frequency

A summary of the LIN receive operation is shown in Figure 14-115:

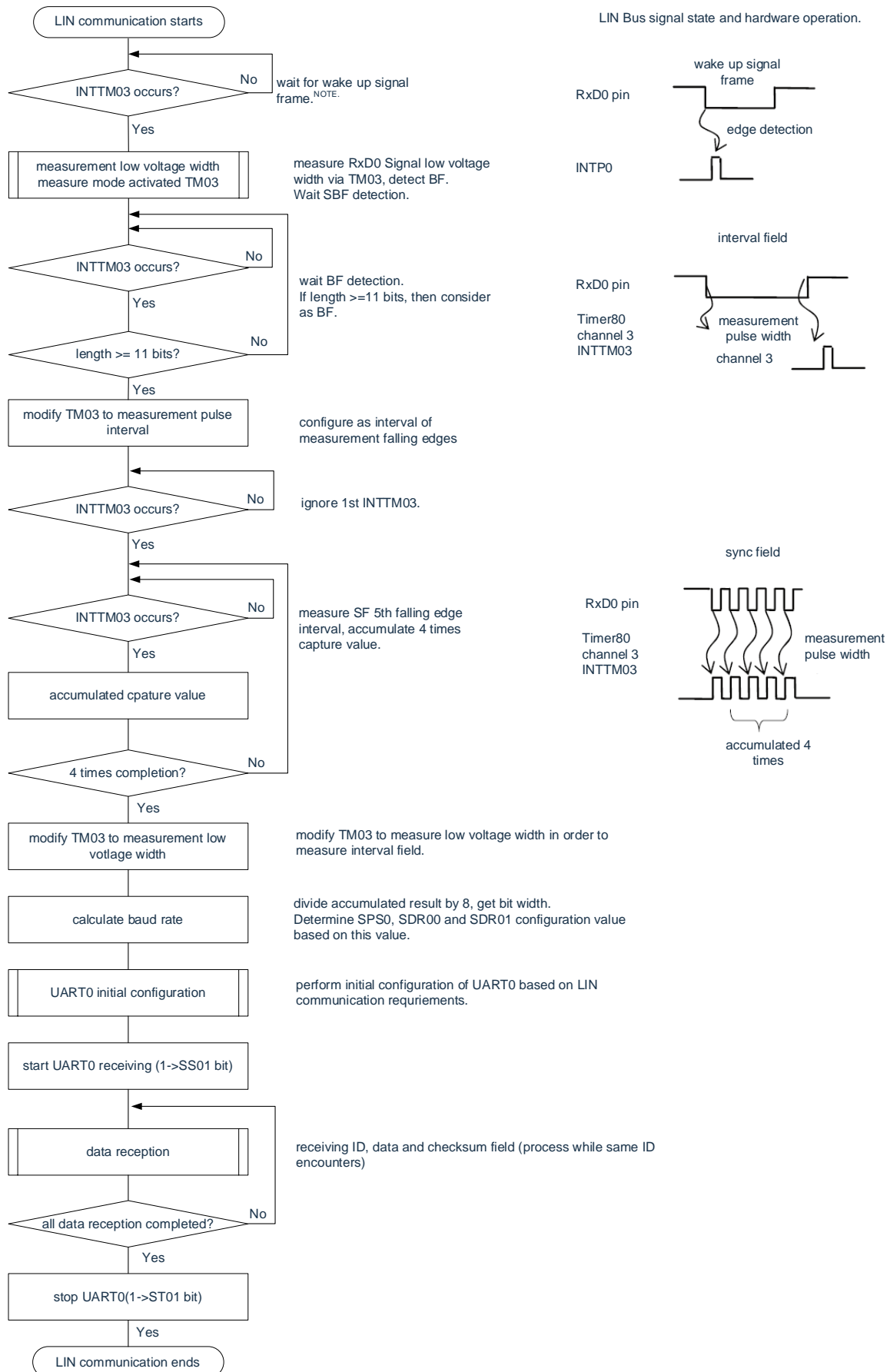
Figure 14-115 LIN reception operation



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 14-116 Flowchart of LIN reception



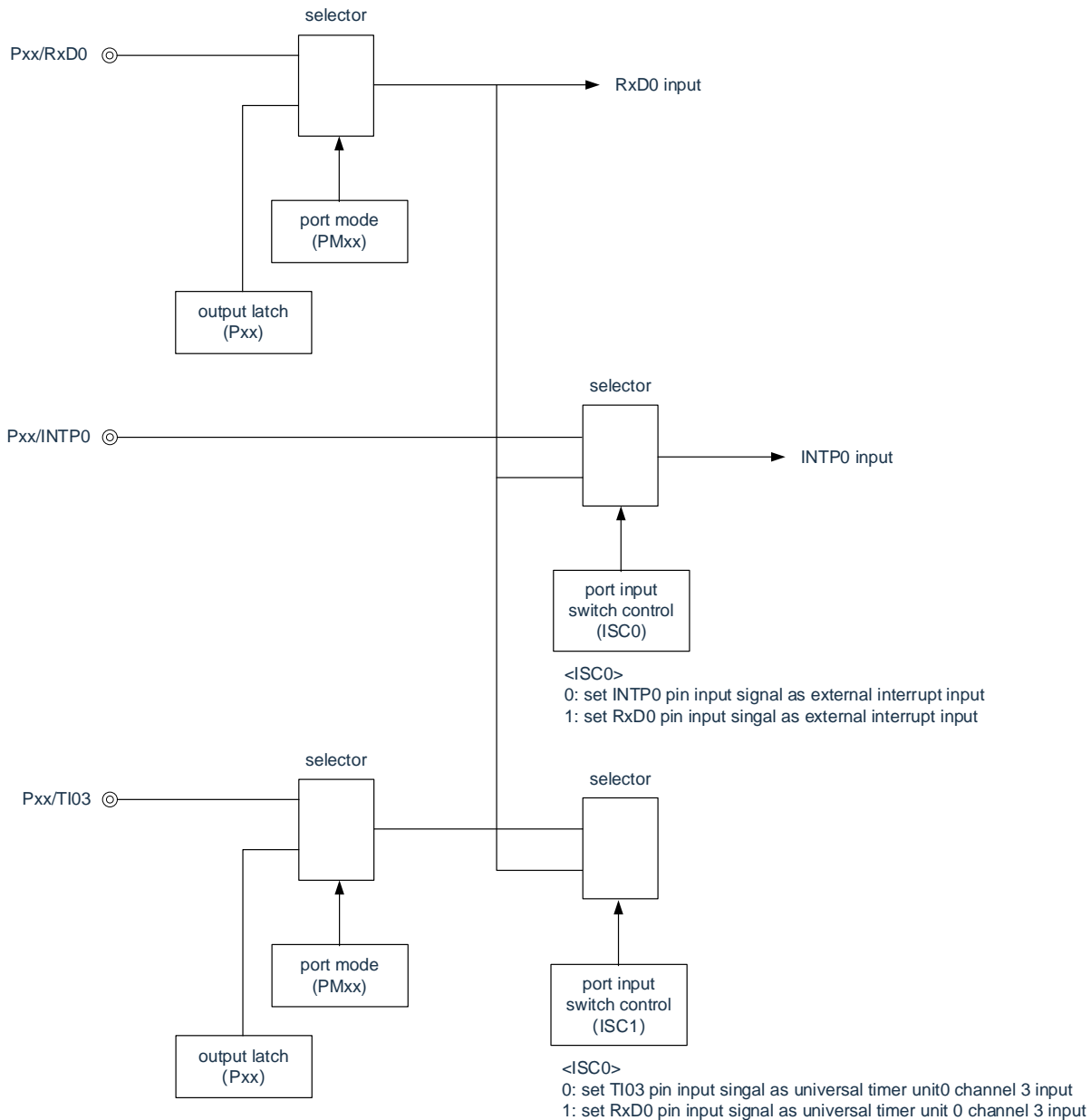
Note It is only needed in the sleeping status.

The port structure diagram for LIN receive operations is shown in Figure 14-117.

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 14-117 Port structure diagram for LIN receive operation



Remark ISC0, ISC1: bit0 and bit1 of the Input Switching Control Register (ISC) (See Figure 14-19)

Peripheral features for LIN communication operations are summarized as follows:

<Peripheral features used>

- External interrupt (INTP0): Detection of wake-up signal
Purpose: Detects edges of wake-up signals and the start of communication.
- Channel 3 of universal 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).
- Channel 0 and Channel 1 (UART0) of universal serial communication Unit 0 (SCI0)

14.9 Simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication operation

This is a function that synchronizes clock communication with multiple devices through a total of 2 lines of serial clock (SCL) and serial data (SDA). Because this simplified I²C is designed for single communication with EEPROM, flash memory, A/D converters, etc., it is only used as a master device.

For start and stop conditions, AC specifications must be adhered to and processed by software while operating the control registers.

[Transmit and receive data]

- Master sending, master receiving (limited to single master control functions).
- ACK output function ^{Note}, ACK detection function
- 8 bits of data length (when sending the address, specify the address with a high 7 bits, and use the lowest bit for R/W control).
- Generate start conditions and stop conditions through the software.

[Interrupt function]

- End of transfer interruption

[Error detection flag]

- ACK error

※[Features not supported by Simplified I²C]

- Slave transmission, slave reception
- Multi-master function (arbitration failure detection function).
- Wait for detection function

Note When receiving the last data, if you write "0" to the SDOEmn bit (SDOEm register) to stop the output of the serial communication data, the ACK is not output. For details, please refer to "14.9.3(2) Processing Flow".

Remark m: unit number (m=0, 1) n: channel number (n=0~3) mn=00~ 03, 10~11

Channels 0 to 3 of SCI0 and channels 0 to 1 of SCI1 support Simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) channels.

Simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21, IIC30, IIC31) have the following four types of communication operations:

- Address segment transmission (see 14.9.1).
- Data transmission (see 14.9.2).
- Data reception (see 14.9.3).
- Generation of stop conditions (see 14.9.4).

14.9.1 Address field transmission

Address field transmission is the first transmission operation to specifically specify the transmitting object (slave device) that is the first to occur during I2C communication. After generating the start condition, the address (7 bits) and the transmission direction (1 bit) are sent as 1 frame.

Simplified I ² C	IIC00	IIC01	IIC10	IIC11	IIC20	IIC21
Object channel	SCI0 Channel 0	SCI0 Channel 1	SCI0 Channel 2	SCI0 Channel 3	SCI1 Channel 0	SCI1 Channel 1
Used pin	SCL00, SDA00 ^{Note1}	SCL01, SDA01 ^{Note1}	SCL10, SDA10 ^{Note1}	SCL11, SDA11 ^{Note1}	SCL20, SDA20 ^{Note1}	SCL21, SDA21 ^{Note1}
Interrupt	INTIIC00	INTIIC01	INTIIC10	INTIIC11	INTIIC20	INTIIC21
Error detection flag	ACK error detection flag (PEFmn).					
Transmitted data length	8 bits (send the highest 7 bits as the address and the lower 1 bit as R/W control).					
Transfer rate ^{Note 2}	Max. $f_{MCK}/4$ [Hz] (SDRmn [15:9] ≥ 1) f_{MCK} : The operation clock frequency of the object channel must meet the following conditions in each mode of I ² C: <ul style="list-style-type: none"> • Max.1MHz (enhanced fast mode). • Max.400kHz (fast mode). • Max.100kHz (standard mode). 					
Data level	Positive phase output (default: high).					
Parity bits	No parity bits.					
Stop bit	Appending 1 bit (for ACK reception).					
Data direction	MSB first					

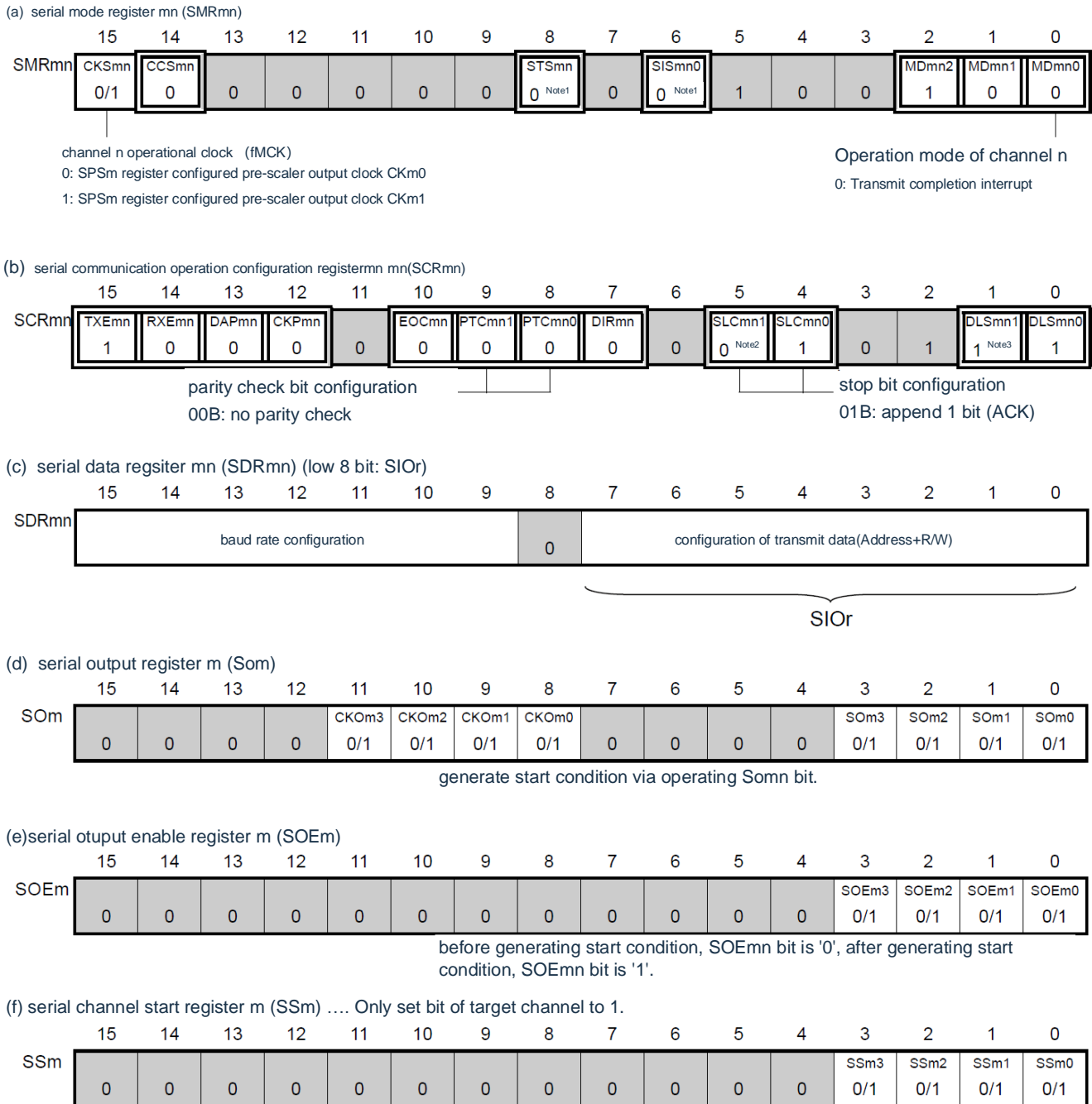
Note 1. To communicate via Simplified I²C, N-channel open-drain output mode (POMxx=1) must be set through the port output mode register (POMxx). For details, please refer to “Chapter 2 Pin Functions”

2. It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Remark m: unit number (m=0, 1) n: channel number (n=0~3) mn=00~ 03, 10~11.

(1) Register setting

Figure 14-118 Example of register setting contents when transmitting address field of simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21)



Note1. Only for SMR00, SMR03, SMR11.

2. Limited to SCR00, SCR02, SCR10 only.

3. Limited to SCR00 register and SCR01 register, other fixed as "1".

Remark1.m: unit number (m=0, 1) n: channel number (n=0~3) r: IIC number (r=00, 01, 10, 11, 20, 21)
 mn=00~03, 10~11

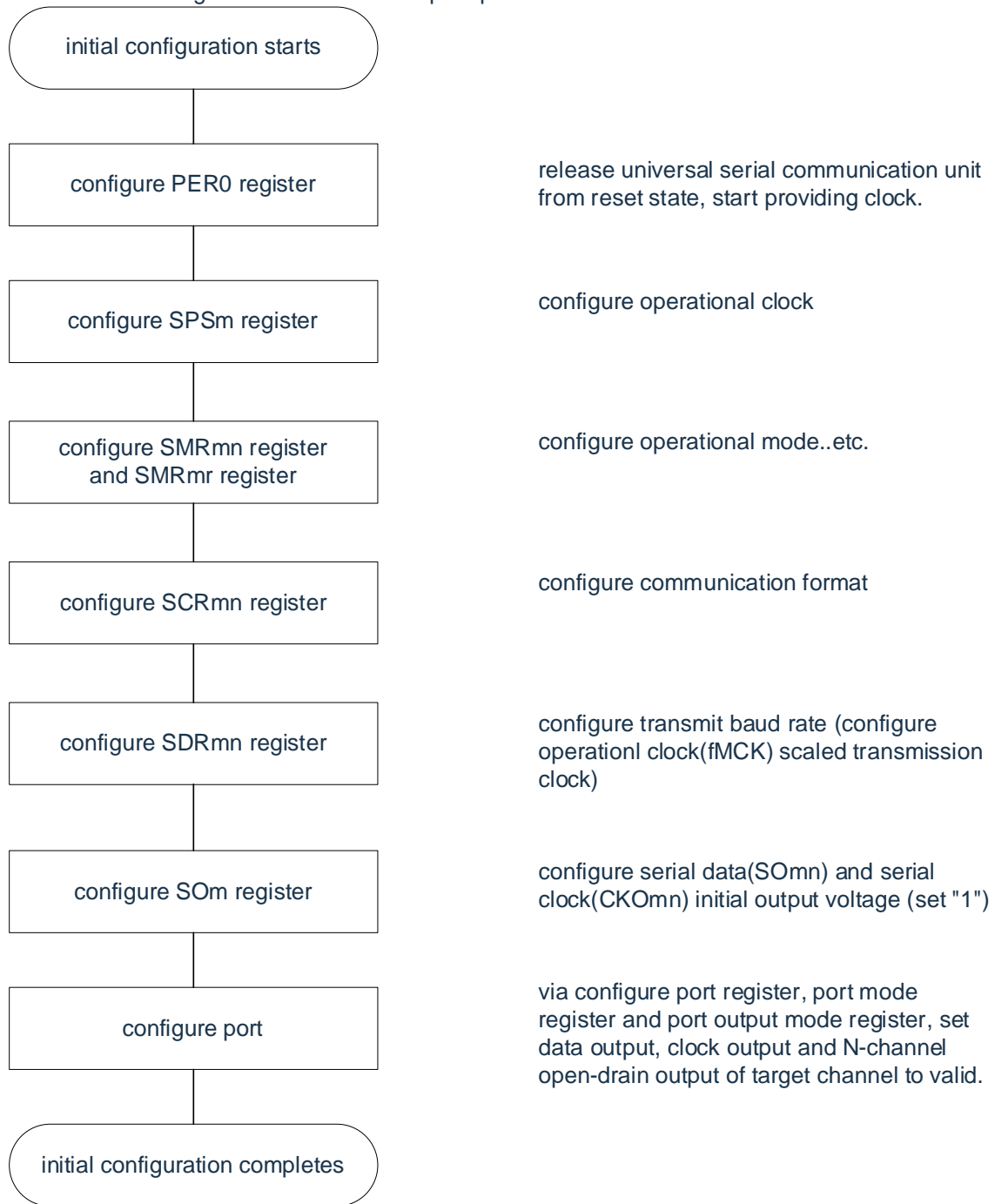
2. : Fixed in IIC mode. : Cannot be set (initial value).

×: This is the bit that cannot be used in this mode (set 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) Operation steps

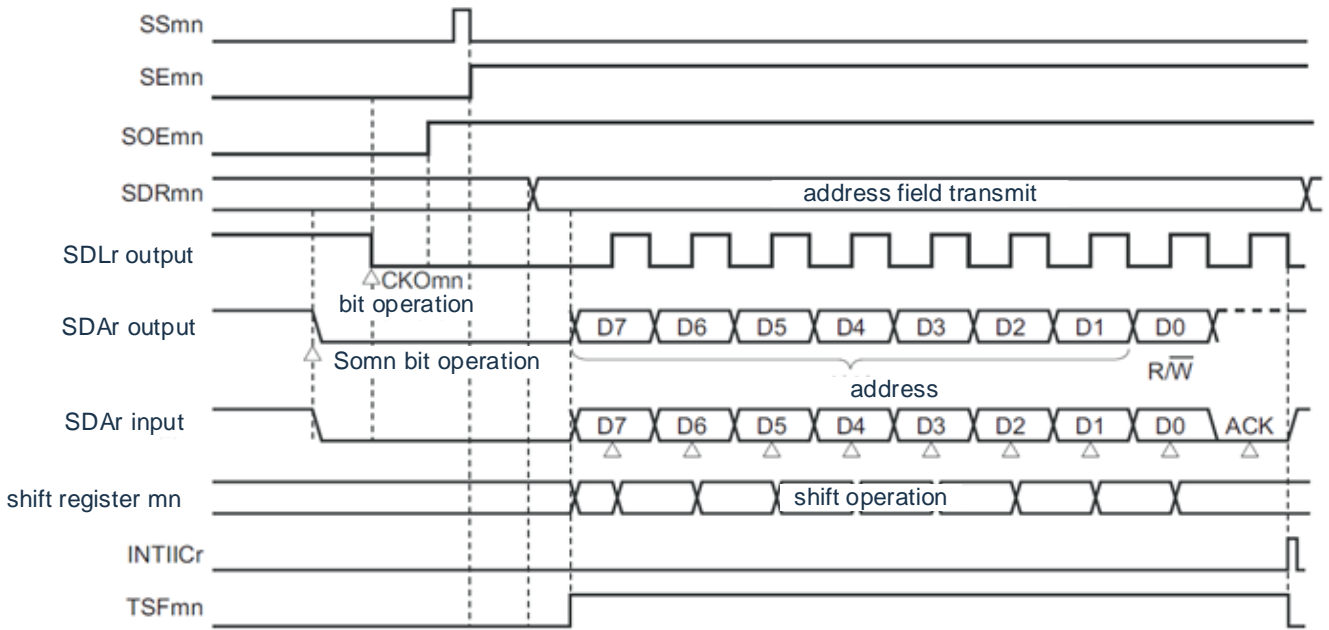
Figure 14-119 Initial setup step for the address field transmission



Remark At the end of the initial setup, Simplified I2C (IIC00, IIC01, IIC10, IIC11, IIC20 IIC21) is output disabled and is in the operation stop state.

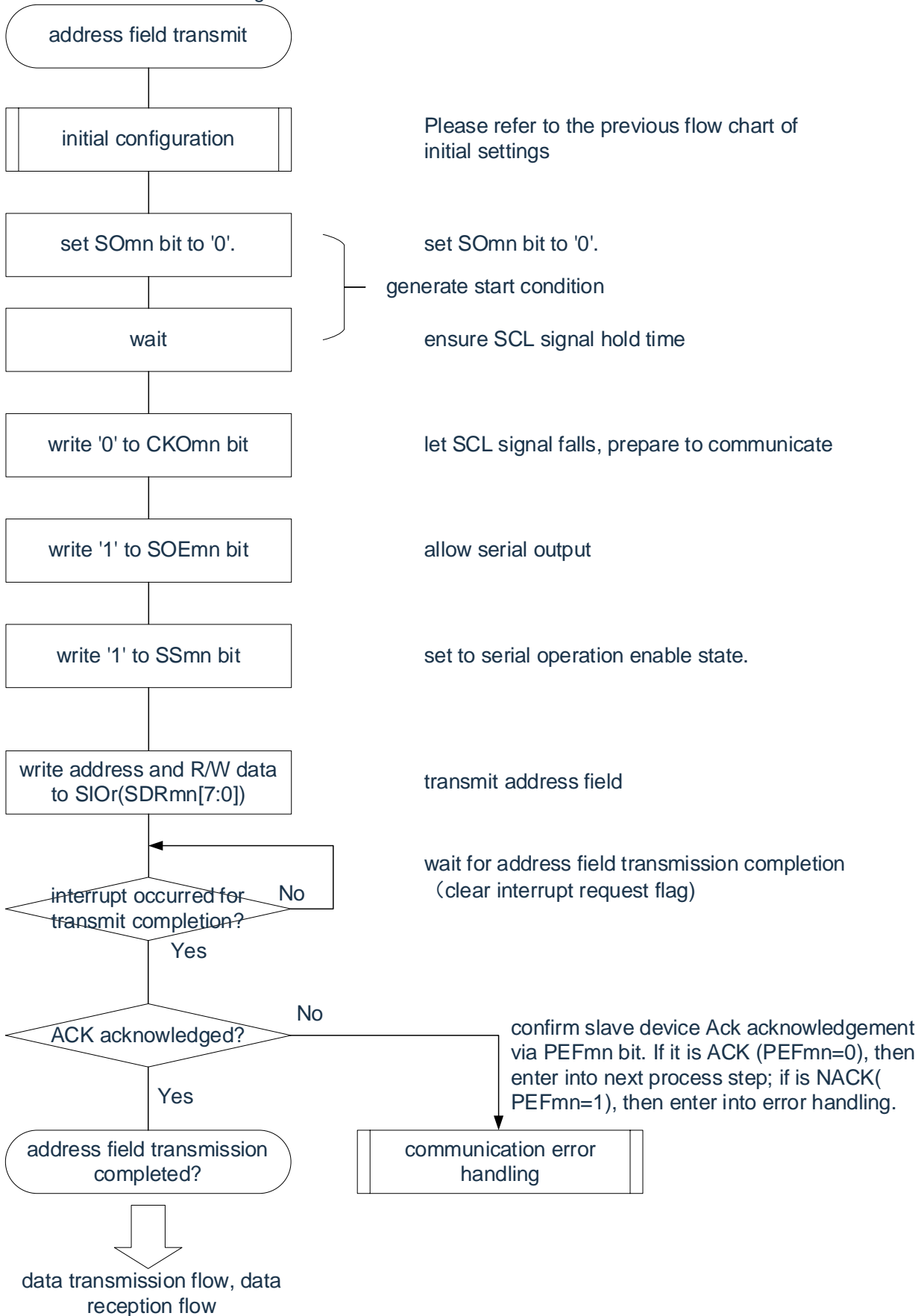
(3) Processing flow

Figure 14-120 Timing diagram of the address field transmission



Remark m: unit number (m=0, 1) n: channel number (n=0~3) r: IIC number (r=00, 01, 10, 11, 20, 21)
 mn=00~03, 10~11

Figure 14-121 Flowchart of address field transmission



14.9.2 Data transmission

Data transmission is the operation of transmitting data to the transmission object (slave device) after the address segment is transmitted. A stop condition is generated after all data is sent to the object slave and the bus is released.

Simplified I ² C	IIC00	IIC01	IIC10	IIC11	IIC20	IIC21
Object channel	SCI0 Channel 0	SCI0 Channel 1	SCI0 Channel 2	SCI0 Channel 3	SCI1 Channel 0	SCI1 Channel 1
The pin used	SCL00, SDA00 ^{Note1}	SCL01, SDA01 ^{Note1}	SCL10, SDA10 ^{Note1}	SCL11, SDA11 ^{Note1}	SCL20, SDA20 ^{Note1}	SCL21, SDA21 ^{Note1}
interrupt	INTIIC00	INTIIC01	INTIIC10	INTIIC11	INTIIC20	INTIIC21
	Limited to end-of-transmit interrupts (buffer null interrupts cannot be selected).					
Error detection flag	ACK error flag (PEFmn)					
Transmitted data length	8 bits					
Transfer rate ^{Note 2}	Max. $f_{MCK}/4$ [Hz] ($SDRmn[15:9] \geq 1$) f_{MCK} : operation clock frequency of the object channel. However, the following conditions must be met in each mode of I ² C. <ul style="list-style-type: none"> • Max.1MHz (enhanced fast mode). • Max.400kHz (fast mode). • Max.100kHz (standard mode). 					
Data level	Normal-phase output (default: high).					
Parity bit	No parity bits.					
Stop bit	Appending 1 bit (for ACK reception).					
Data direction	MSB first					

Note 1. To communicate via Simplified I²C, N-channel open-drain output mode (POMxx=1) must be set through the port output mode register (POMxx). For details, please refer to “2.3 Registers for Control Port Functions” and “2.5 Register Settings When Using the Multiplexing Function”.

2. It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Remark m: unit number (m=0, 1) n: channel number (n=0~3) mn=00~ 03, 10~11

(1) Register setting

Figure 14-122 Example of register setting contents for simplified I²C data transmission (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21)

(a) serial mode register mn (SMRmn).....do not operate this register while data is transmitting or receiving.

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	0	0	0	0	0	0	0 <small>Note1</small>	0	0 <small>Note1</small>	1	0	0	1	0	0

(b) serial communication operation configuration register mn (SCRmn).....do not operate bits other than TXEmn and RXEmn of this register while data is transmitting or receiving.

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		DLSmn1	DLSmn0
1	0	0	0	0	0	0	0	0	0	0	0 <small>Note2</small>	1	0	1	1 <small>Note3</small>

(c) serial data register mn (SDRmn) (low 8 bit: SIO_r)only lower 8 bits valid while data is transmitting or receiving.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
baud rate configuration <small>Note4</small>								0	configuration of transmit data						
 SIO _r															

(d) serial output register m (Som)do not operate this register while data is transmitting or receiving.

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	0/1 <small>Note5</small>	0/1 <small>Note5</small>	0/1 <small>Note5</small>	0/1 <small>Note5</small>	0	0	0	0	0/1 <small>Note5</small>	0/1 <small>Note5</small>	0/1 <small>Note5</small>	0/1 <small>Note5</small>

(e) serial output enable register m (SOEm)do not operate this register while data is transmitting or receiving.

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	1	1	1	1

(f) serial channel start register m (SSm)do not operate this register while data is transmitting or receiving.

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

Note 1. Limited to SMR01, SMR03, SMR11 registers.

2. Limited to SCR00, SCR02, SCR10 registers.

3. Limited to SCR00 register and SCR01 register, other fixed as "1".

4. Because it is already set when sending the address segment, it does not need to be set.

5. During the operation of communication, the value changes due to the communication data.

Remark 1.m: Unit number (m=0, 1) n: channel number (n=0~3)r: IIC number (r=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11

2. : Fixed in IIC mode. : Cannot be set (set initial value).

x: This is the bit that cannot be used in this mode (set 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) Processing flow

Figure 14-123 Timing diagram of data transmission

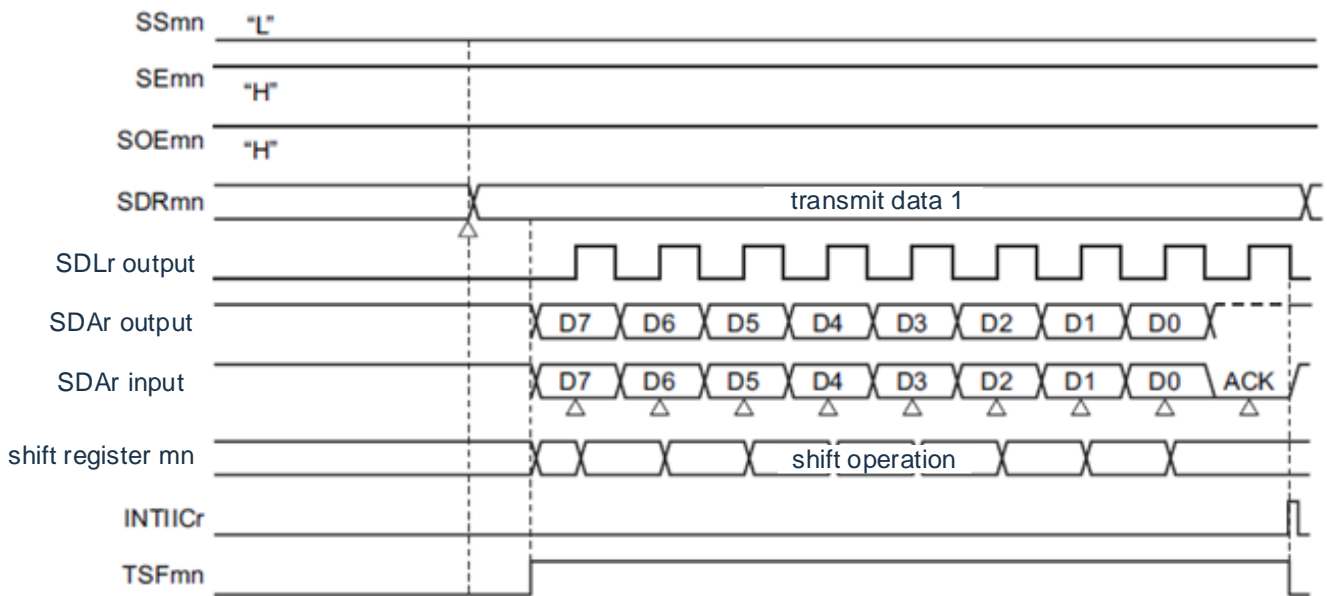
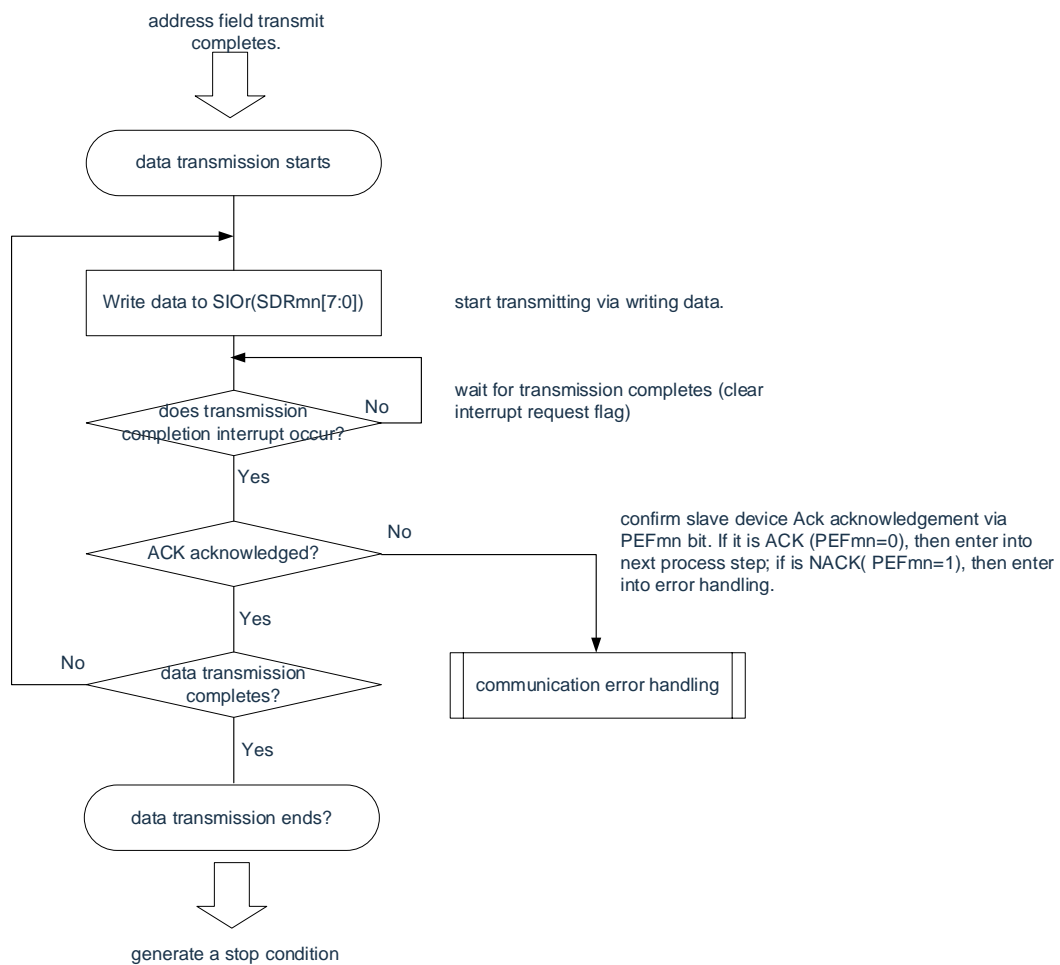


Figure 14-124 Flow chart of data transmission



14.9.3 Data reception

Data reception is an operation of receiving data from the transmission target (slave) after sending the address field. A stop condition is generated and the bus is released after all data is received from the target slave.

Simplified I ² C	IIC00	IIC01	IIC10	IIC11	IIC20	IIC21
Object channel	SCI0 Channel 0	SCI0 Channel 1	SCI0 Channel 2	SCI0 Channel 3	SCI1 Channel 0	SCI1 Channel 1
Used pin	SCL00, SDA00 ^{Note1}	SCL01, SDA01 ^{Note1}	SCL10, SDA10 ^{Note1}	SCL11, SDA11 ^{Note1}	SCL20, SDA20 ^{Note1}	SCL21, SDA21 ^{Note1}
Interrupt	INTIIC00	INTIIC01	INTIIC10	INTIIC11	INTIIC20	INTIIC21
Error detection flag	Limited to transmit complete interrupts (buffer empty interrupts cannot be selected).					
Transmitted data length	Only the Overflow Error Detection Flag (OVFmn).					
Transfer rate ^{Note2}	8 bits					
Data level	Max. $f_{MCK}/4$ [Hz] ($SDR_{mn}[15:9] \geq 1$) f_{MCK} : Operation clock frequency of the object channel. However, the following conditions must be met in each mode of I ² C.					
Parity bit	<ul style="list-style-type: none"> • Max.1MHz (enhanced fast mode). • Max.400kHz (fast mode). • Max.100kHz (standard mode). 					
Stop bit	Positive phase output (default: high).					
Data direction	No parity bits.					
	Append 1 bit (for ACK transmission).					
	MSB first					

Note 1 To communicate via Simplified I²C, N-channel open-drain output mode (POMxx=1) must be set through the port output mode register (POMxx). For details, please refer to “2.3 Registers for Control Port Functions” and “2.5 Register Settings When Using the Multiplexing Function”.

2. It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Remark m: unit number (m=0, 1) n: channel number (n=0~3) mn=00~ 03, 10~11.

(1) Register setting

Figure 14-125 Example of register setting contents for simplified I²C data reception (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21)

(a) serial mode register mn (SMRmn).....do not operate this register while data is transmitting or receiving.

	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 <small>Note1</small>	0	0 <small>Note1</small>	1	0	0	1	0	0

(b) serial communication operation configuration register mn (SCRmn).....do not operate bits other than TXEmn and RXEmn of this register while data is transmitting or receiving.

	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	
	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	1
														0 <small>Note2</small>	1	
															1 <small>Note3</small>	1

(c) serial data register mn (SDRmn) (low 8 bit: SIO_r)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	baud rate configuration <small>Note4</small>							0	virtual transmit data configuration (FFH)							
	SIO _r															

(d) serial output register m (Som)do not operate this register while data is transmitting or receiving.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOM					CKOm3	CKOm2	CKOm1	CKOm0					SOm3	SOm2	SOm1	SOm0
	0	0	0	0	0/1 <small>Note5</small>	0/1 <small>Note5</small>	0/1 <small>Note5</small>	0/1 <small>Note5</small>	0	0	0	0	0/1 <small>Note5</small>	0/1 <small>Note5</small>	0/1 <small>Note5</small>	0/1 <small>Note5</small>

(e) serial output enable register m (SOEm)do not operate this register while data is transmitting or receiving.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm													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)do not operate this register while data is transmitting or receiving.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm													SSm3	SSm2	SSm1	SSm0
	0	0	0	0	0	0	0	0	0	0	0	0	0/1	0/1	0/1	0/1

Note 1. SMR01, SMR03, SMR11 registers only.

2. Limited to SCR00, SCR02, SCR10 registers only.

3. Limited to SCR00 register and SCR01 register, other fixed as "1".

4. Because it is already set when sending the address segment, it does not need to be set.

5. During the operation of communication, the value changes due to the communication data.

Notice 1.m: unit number (m=0, 1) n: channel number (n=0~3) r: IIC number (r=00, 01, 10, 11, 20, 21) mn=00~03, 10~11.

2. : Fixed in IIC mode. : Cannot be set (set initial value).

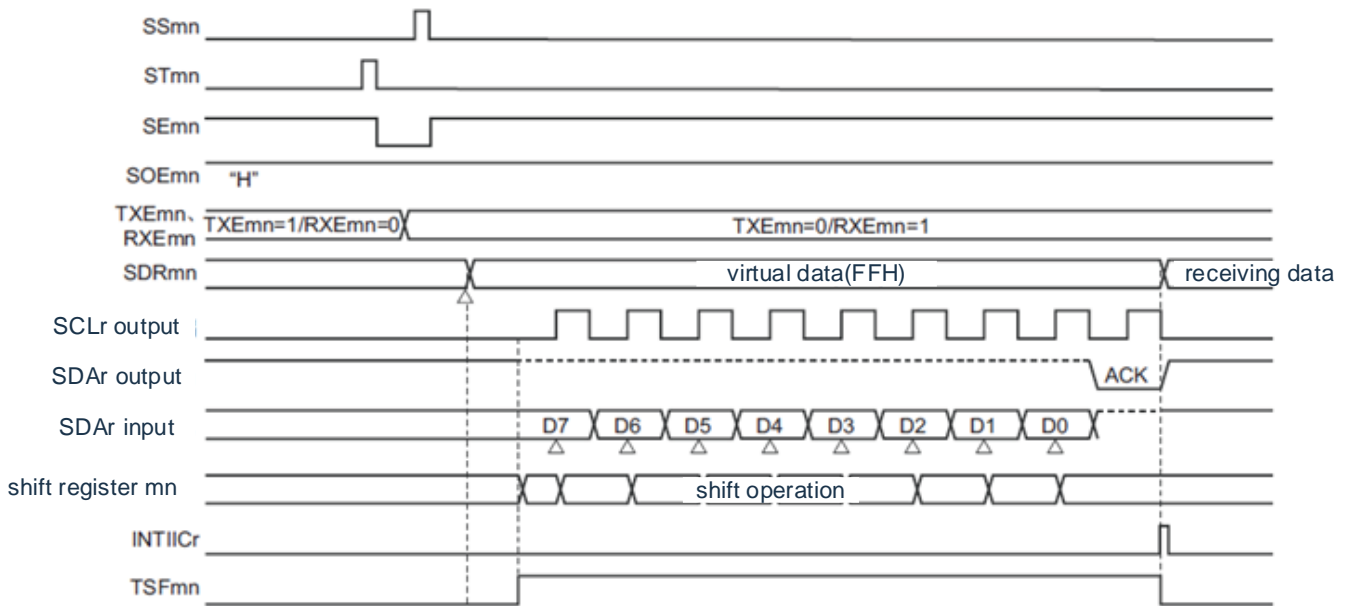
x: This is the bit that cannot be used in this mode (set the initial value if it is not used in other modes either).

0/1: Set "0" or "1" according to the user's purpose.

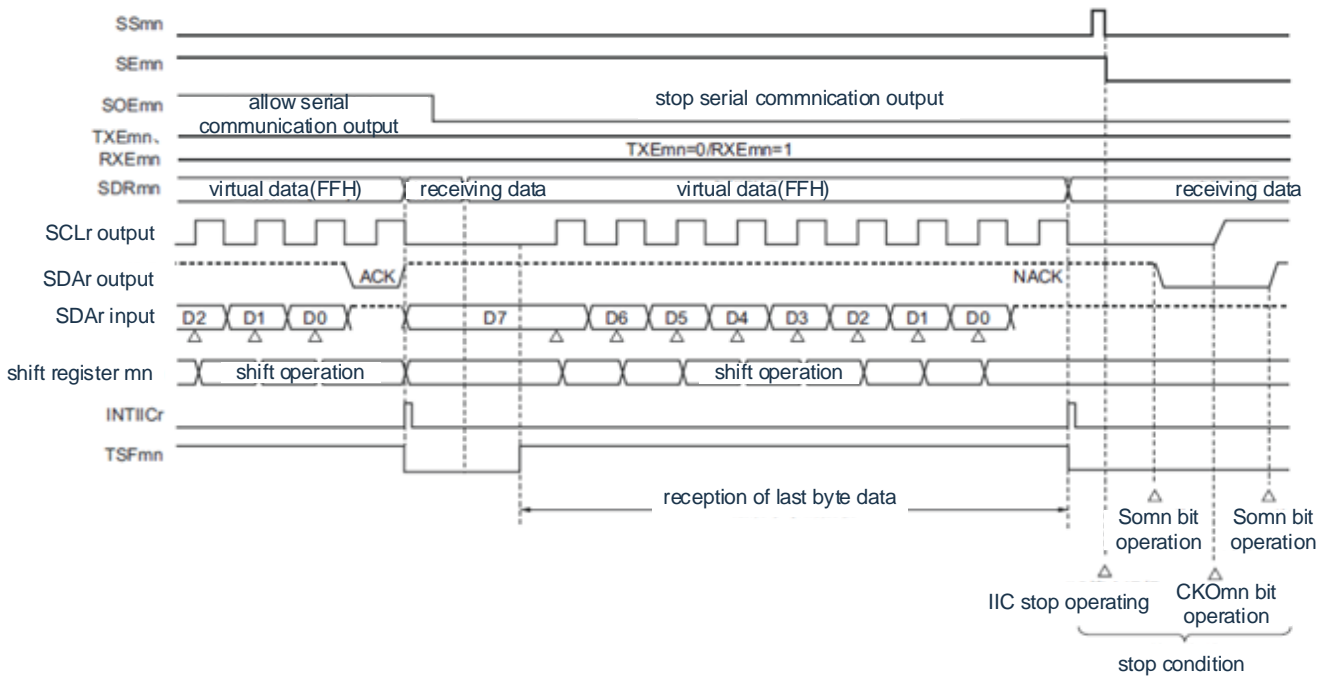
(2) Processing flow

Figure 14-126 Timing diagram of data reception

(a) Start of receiving data

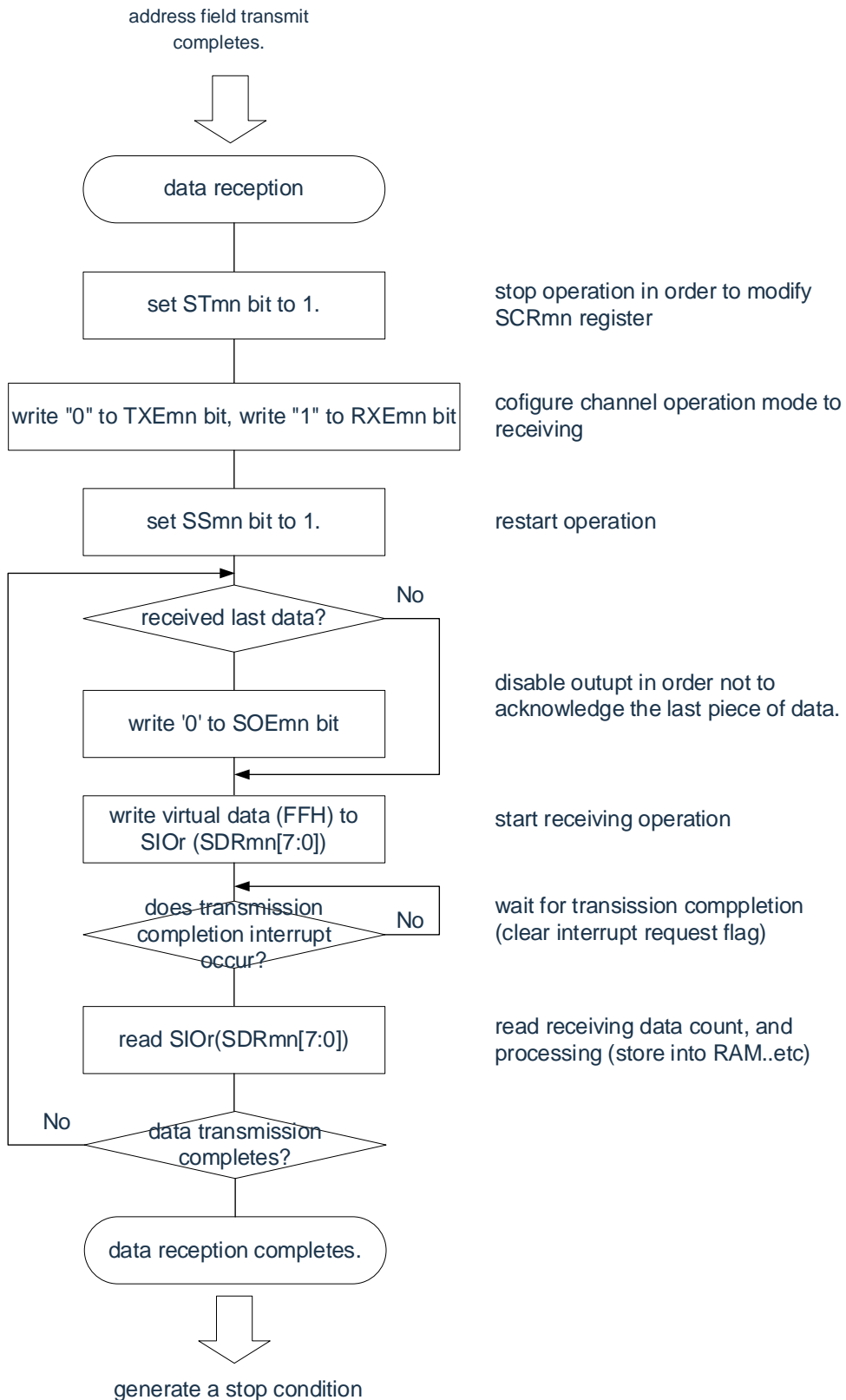


(b) Receive the last data



Remark m: unit number (m=0, 1) n: channel number (n=0~3) r: IIC number (r=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11

Figure 14-127 Flowchart of data reception



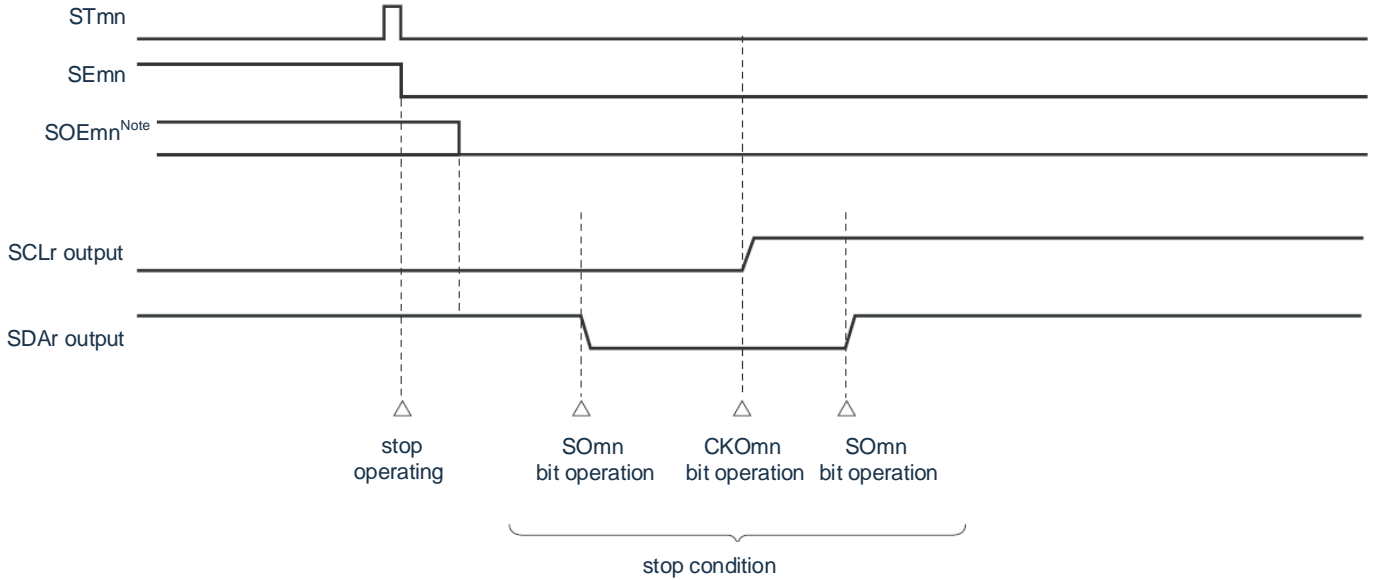
Notice No ACK(NACK) is output when the last data is received. Thereafter, the communication is terminated by stopping the STmn bit of the serial channel stop register m (STm) to '1'.

14.9.4 Generation of stop condition

After all data has been transmitted and received with the target slave, a stop condition is generated and the bus is released.

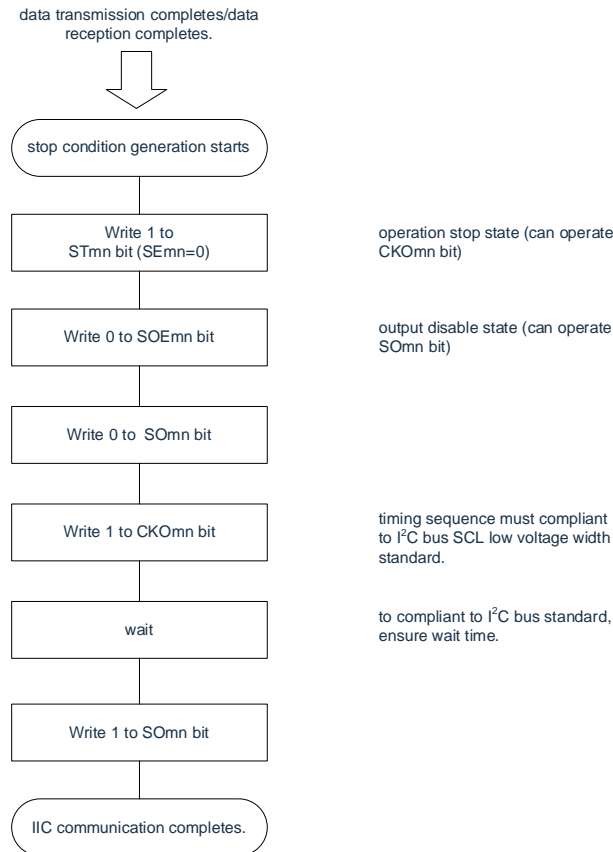
(1) Processing flow

Figure 14-128 Timing diagram for generating a stop condition



Note At the time of receiving, set the SOEmn bit of the serial output enable register m (SOEm) to "0".

Figure 14-129 Flow chart for generating a stop condition



14.9.5 Calculation of transfer rate

The transfer rate for simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication can be calculated using the following equation.

$$(\text{Transfer rate}) = \{ \text{Operation clock } (f_{\text{MCK}}) \text{ frequency of the object channel} \} \div (\text{SDRmn}[15:9] + 1) \div 2$$

Notice Setting SDRmn[15:9] to “0000000B” is prohibited, and the setting value for SDRmn[15:9] must be greater than or equal to “0000001B”. The duty ratio of the SCL signal output by the Simplified I²C is 50%. In I²C bus specification, the low-level width of the SCL signal is greater than the high-level width. Therefore, if 400kbps is set as a fast mode or 1Mbps is set as an enhanced fast mode, the low-level width of the SCL signal output is less than the specification value of the I²C bus. You must set a value for SDRmn[15:9] that meets the I²C bus specification.

Remark 1. It is 1~127 because the value of SDRmn[15:9] is the value of bit15~9 of serial data register (SDRmn) (0000001B~1111111B).

2. m: unit number (m=0, 1) n: channel number (n=0~3) mn=00~03, 10~11

The operating clock (f_{MCK}) depends on the serial clock select register m (SPSm) and bit 15 (CKSmn bit) of the serial mode register mn (SMRmn).

Table 14-5 Selection of Simplified I²C 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	$f_{CLK}=32\text{MHz}$ 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	
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	
Other than the above									Disable settings.	

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 universal serial communications init (SCI) (serial channel stop register m (STm) = 000FH).

Remark1. X: Ignore

2. m: unit number (m=0, 1) n: channel number (n=0~3) mn=00~03, 10~11

An example of setting the I²C transfer rate at $f_{MCK}=f_{CLK}=32\text{MHz}$ is shown below.

I ² C transfer mode (expected transfer rate)	$f_{CLK}=32\text{MHz}$			
	Operation clock (f_{MCK})	SDRmn[15:9]	Calculated transfer rate	Error with expected transfer rate
100kHz	$f_{CLK}/2$	79	100kHz	0.0%
400kHz	f_{CLK}	41	380kHz	5.0% ^{Note}
1MHz	f_{CLK}	18	0.84MHz	16.0% ^{Note}

Note The error cannot be set to '0%' because the SCL signal has a 50% duty cycle.

14.9.6 Processing steps when an error occurs during simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication

The processing steps when an error occurs during a simple I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication are shown in Figure 14-130 and

Figure 14-131.

Figure 14-130 Steps for handling when an overflow error occurs

Software operation	Hardware status	Remark
Read serial data register mn (SDRMN).	→ The BFF m n bit of the SSRm n register is "0" and channel n is receiveable.	This is to prevent overflow errors from ending the next reception during mishandling.
Read serial status register mn(SSRmn).		The type of error is judged, and the reading value is used to clear the error flag.
Write "1" to the serial flag clear trigger register mn (SDIRmn).	→ Clear the error flag.	By writing the read value of the SSRmn register directly to the SDIRmn register, errors during read operations can only be cleared.

Figure 14-131 Processing steps when an ACK error occurs during simplified I²C mode

Software operation	Hardware status	remark
Read the serial status register mn(SSRmn).		Determine the error category, and read the value to remove the error marker.
Write the serial flag to clear the trigger register mn (SDIRmn).	→ Clear the error flag.	By writing the read value of the SSRmn register directly to the SDIRmn register, errors during read operations can only be cleared.
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 running stop.	Because ACK is not returned, the slave device is not ready for receiving. Thus, a stop condition is generated and the bus is released, and communication is started again from the start bar, or a restart can also be generated and start again from the address to send.
Generate a stop condition.		
Generate a start condition		
Set the serial channel start register m (SSm) to SSmn bit 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=0~3) r: IIC number (r=00, 01, 10, 11, 20, 21)

mn=00~03, 10~1.

Chapter 15 Serial Interface SPI

15.1 Serial interface SPI function

The serial interface SPI has the following two modes.

(1) Operation Stop mode

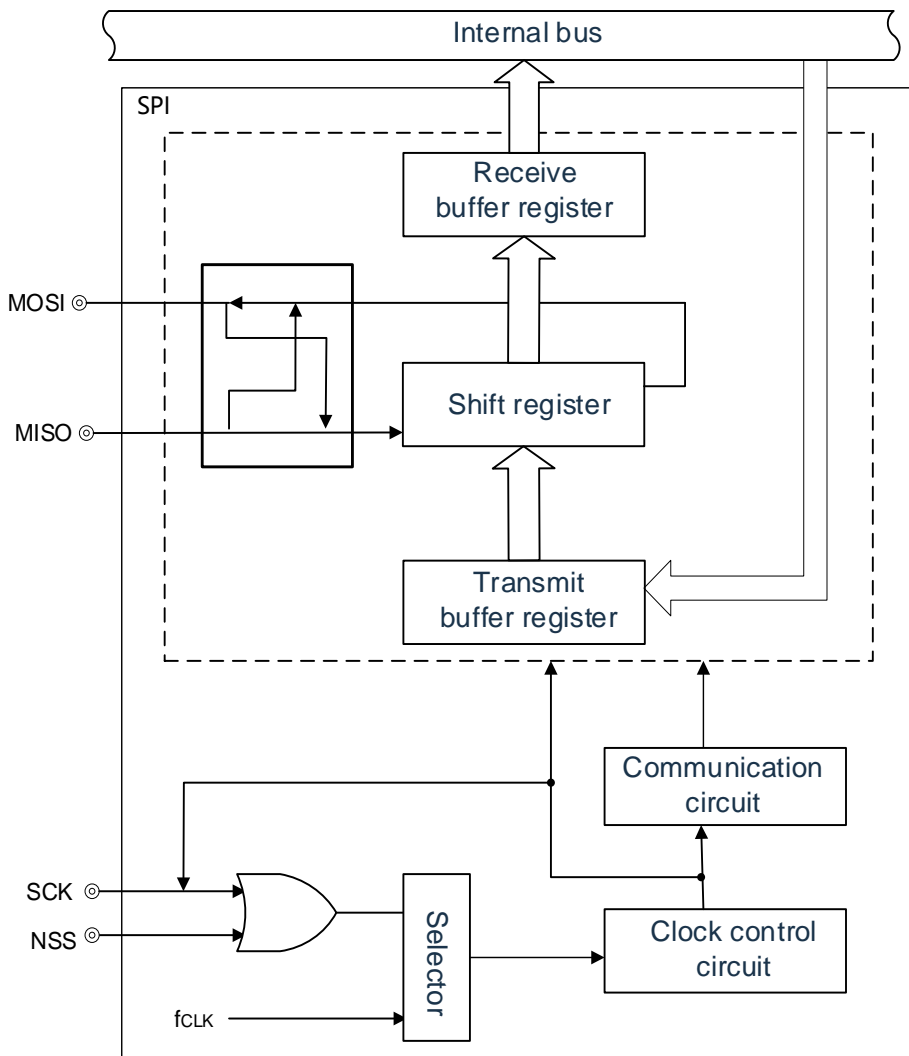
This is a mode used when no serial transfer is taking place, which reduces power consumption.

(2) 3-wire serial I/O mod

This mode transfers 8- or 16-bit data to multiple devices via 3 wires of the serial clock (SCK) and serial data bus (MISO and MOSI).

15.2 Structure of serial interface SPI

Figure 15-1 Block diagram of serial interface SPI



15.3 Registers for controlling serial interface SPI

The serial interface SPI is controlled through the following registers.

- Peripheral enable register 0 (PER0)
- Serial operation mode register (SPIM)
- Serial clock selection register (SPIC)
- Transmit buffer register (SDRO)
- Receive buffer register (SDRI)
- Port mode register (PMxx)
- Port mode control register (PMCxx)
- Port register (Pxx)

15.3.1 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.

15.3.2 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.

Figure 15-2 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 status
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	NSS pin is used

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 SDRIF 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.

2: When SPTF=1 (during serial communication), rewriting of TRMD, DIR, NSSE is prohibited.

3: When TRMD is 0, the MO or SO output is fixed low.

4: Fix the NSS pin input level to 0 or 1 before setting this bit to 1.

15.3.3 SPI clock selection register (SPIC)

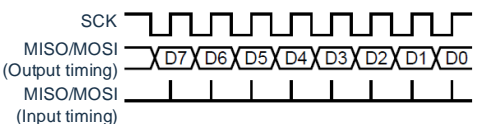
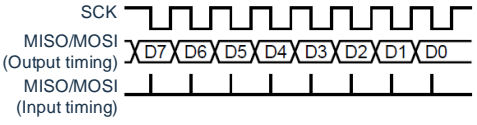
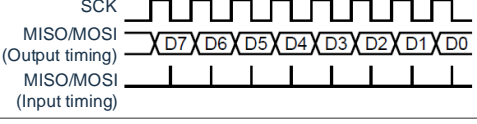
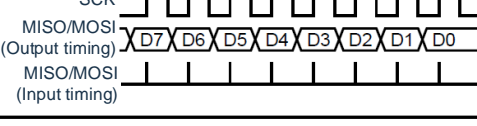
This register specifies the timing of data transmit/receive and sets the serial clock.

It can be set by an 8-bit memory manipulation instruction.

A reset signal is generated to clear this register to 01H.

Figure 15-3Format 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.

15.3.4 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.

Figure 15-4 Format of transmit buffer register (SDRO)



15.3.5 Receive buffer register (SDRI)

This register stores the received data.

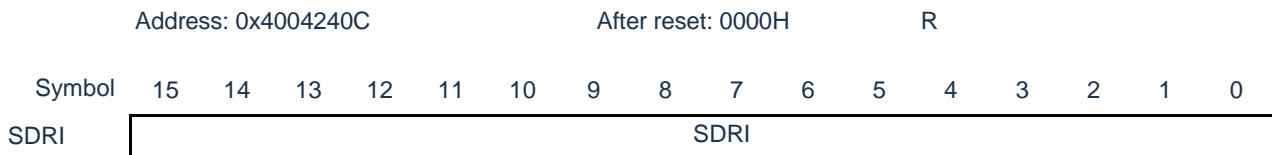
If bit 6 (TRMD) of the Serial Operation Mode Register (SPIM) is set to 0, reception is started by reading data from the SDRI.

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.

Figure 15-5 Format of receive buffer register (SDRI)



15.3.6 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 “2.5 Register Settings for Multiplexing Function”.

15.4 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.

15.4.1 Master transmission 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) Operation steps

Figure 15-6 Initial setup steps for master transmission/reception

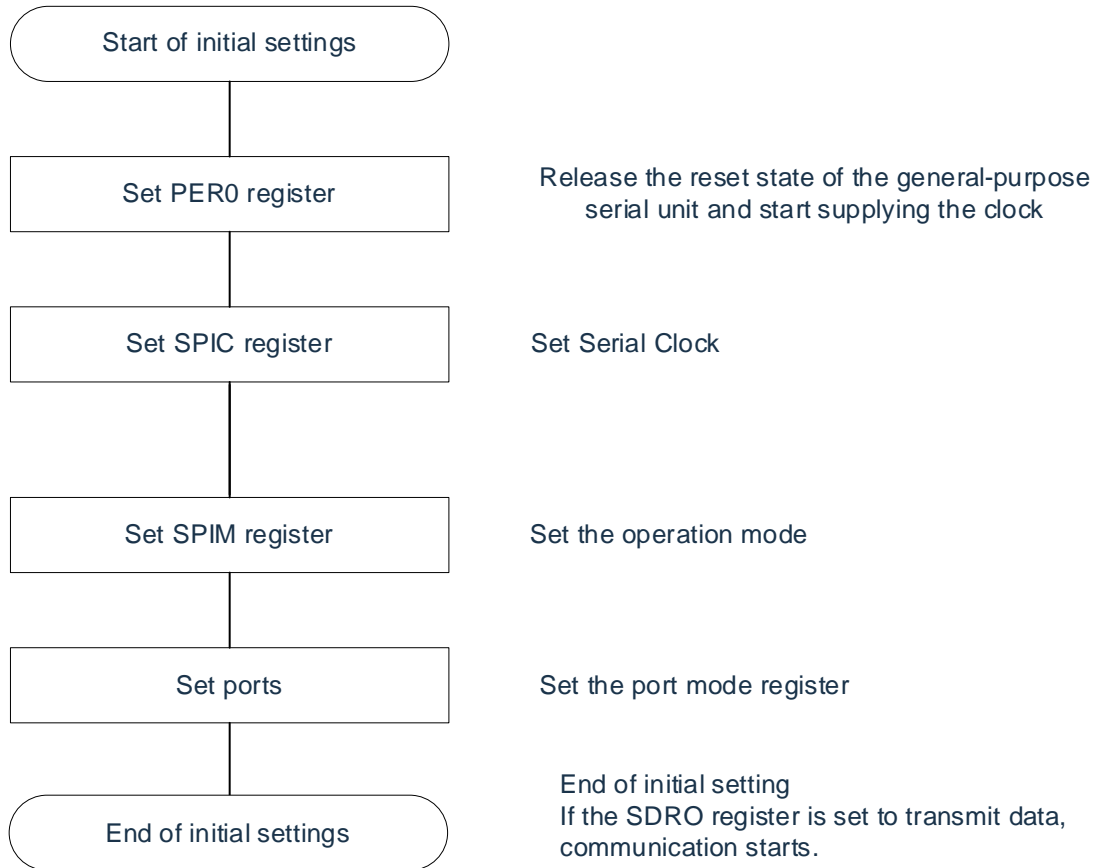
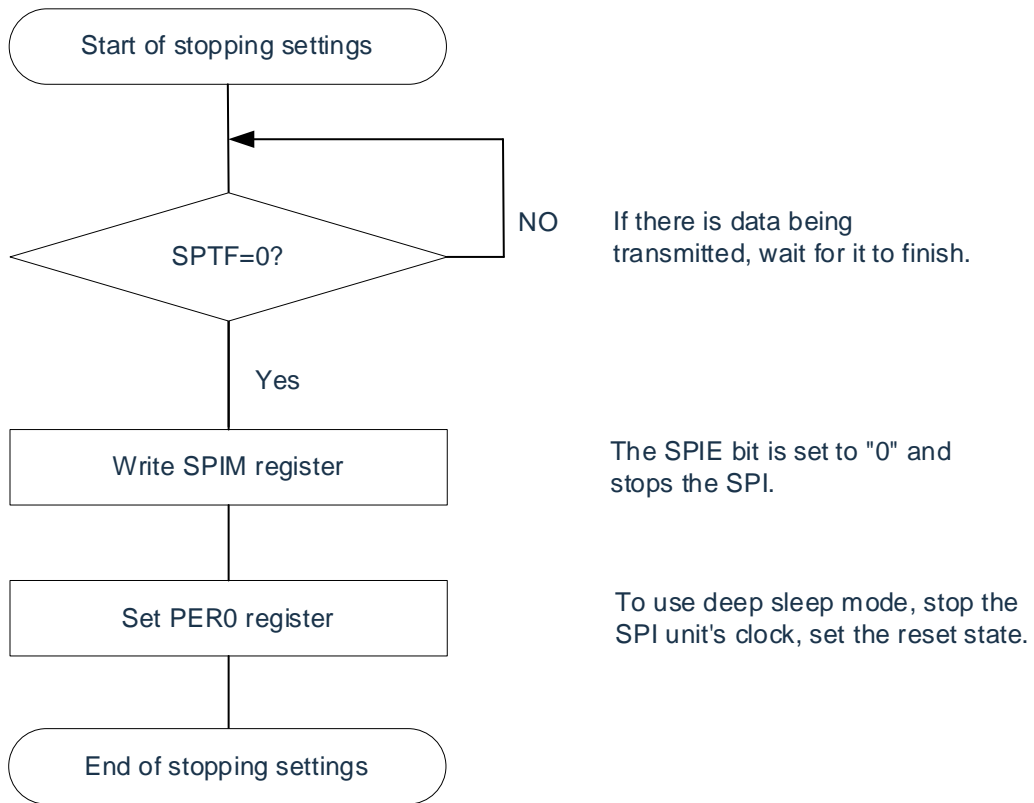


Figure 15-7 Stop steps of master transmission/reception



(2) Processing flow

Figure 15-8 Timing diagram for transmit/receive (single transmit mode) (INTMD=0, DAP=0, CKPmn=0)

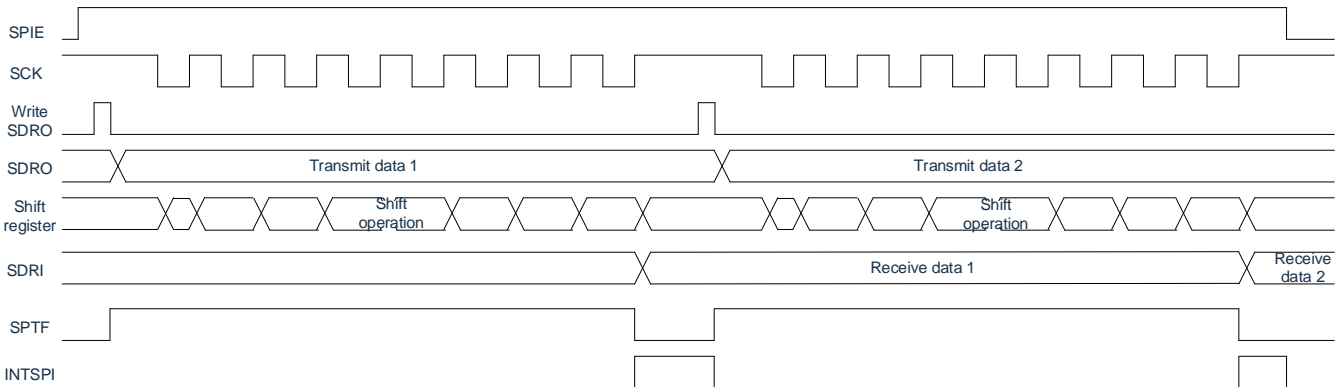
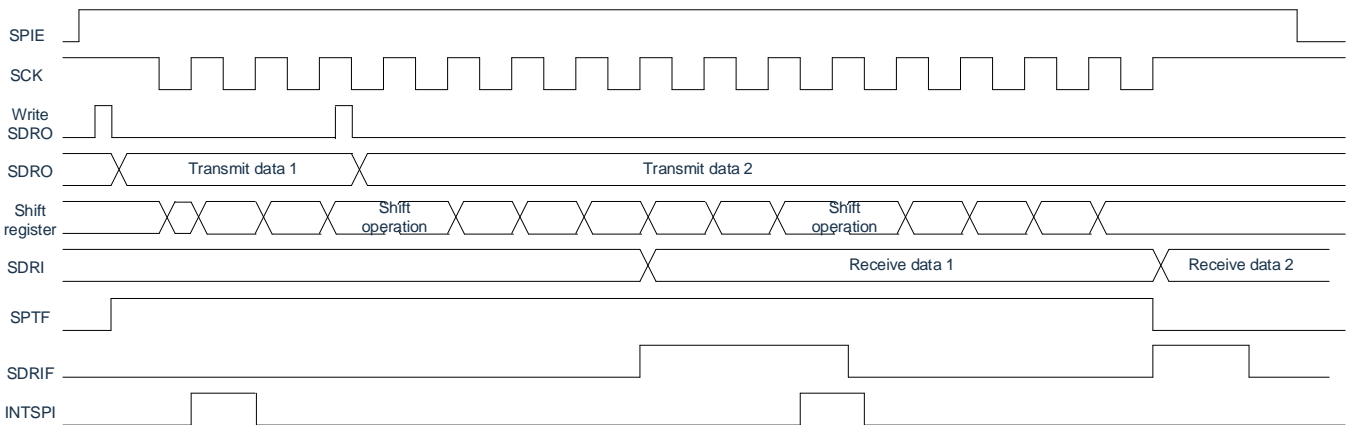


Figure 15-9 Timing diagram for transmit/receive (continuous transmit mode) (INTMD=1, DAP=0, CKPmn=0)



15.4.2 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) Operation steps

Figure 15-10 Initial setup steps for master reception

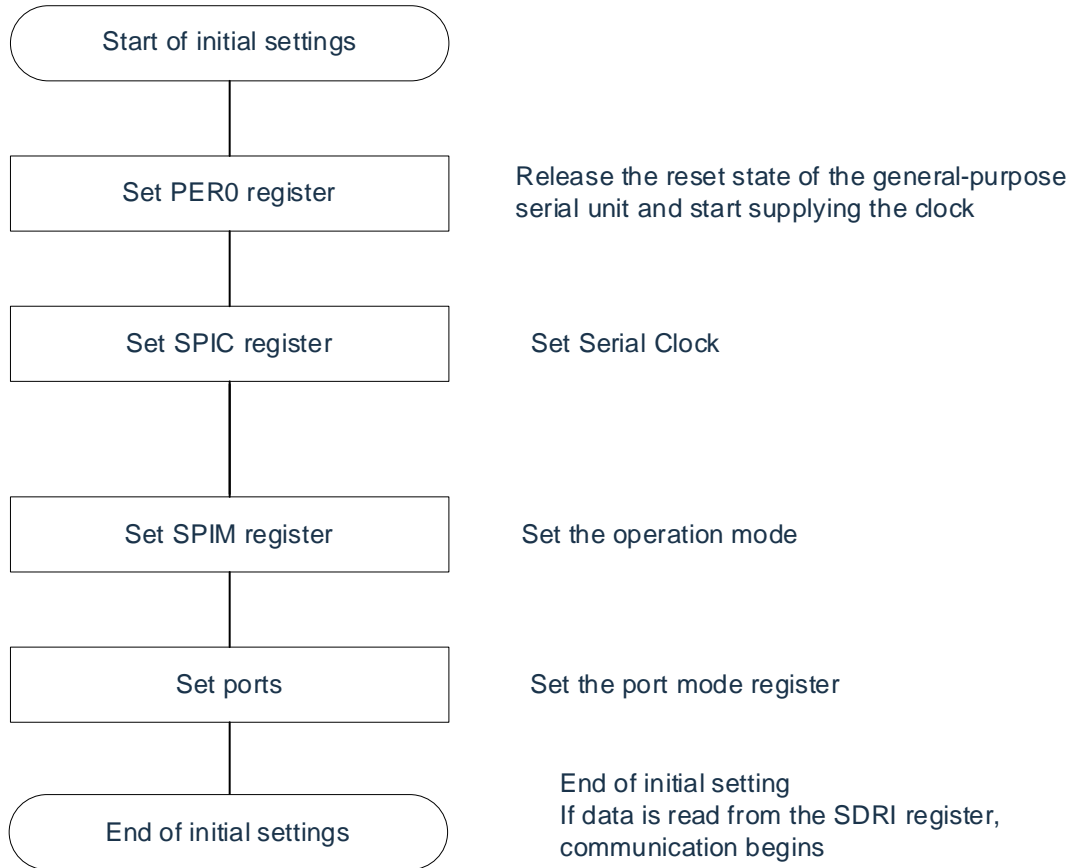
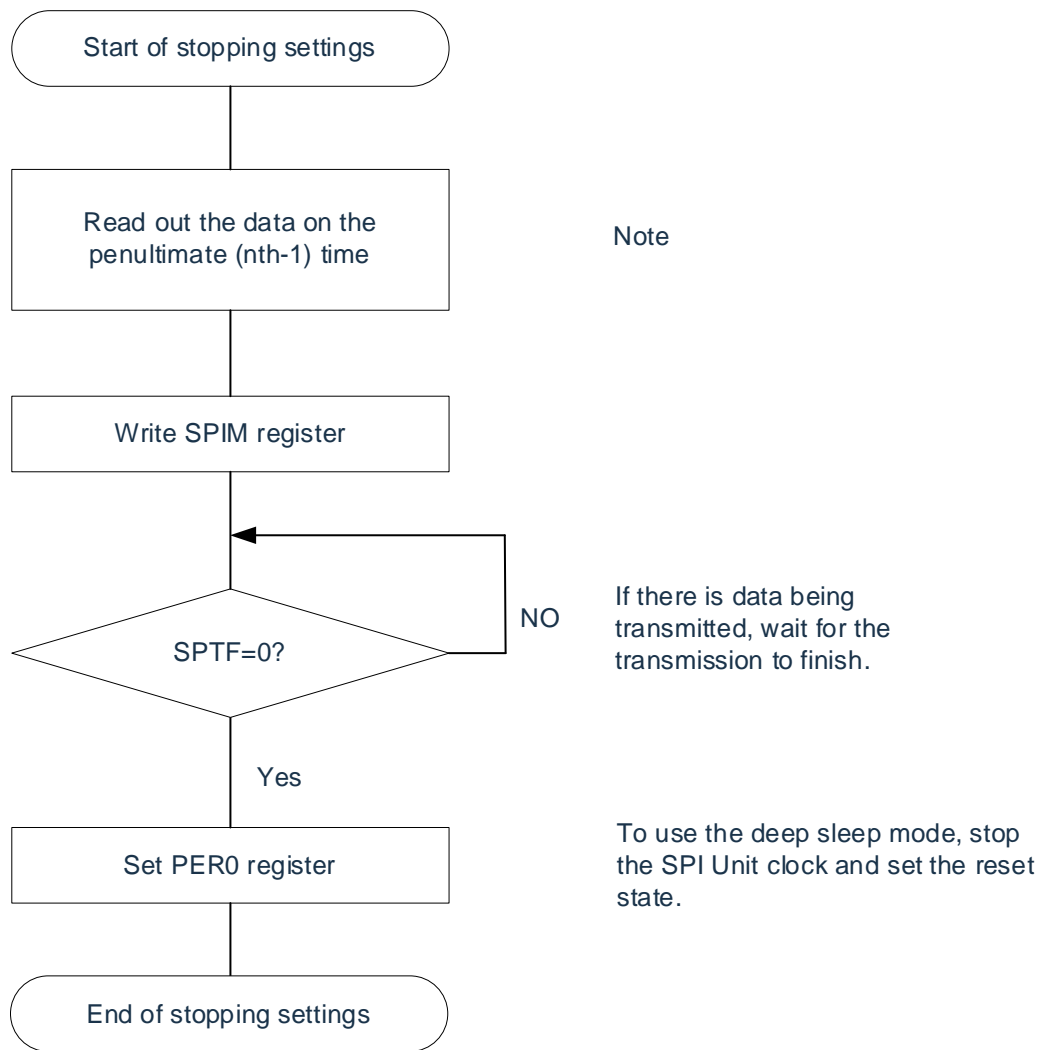


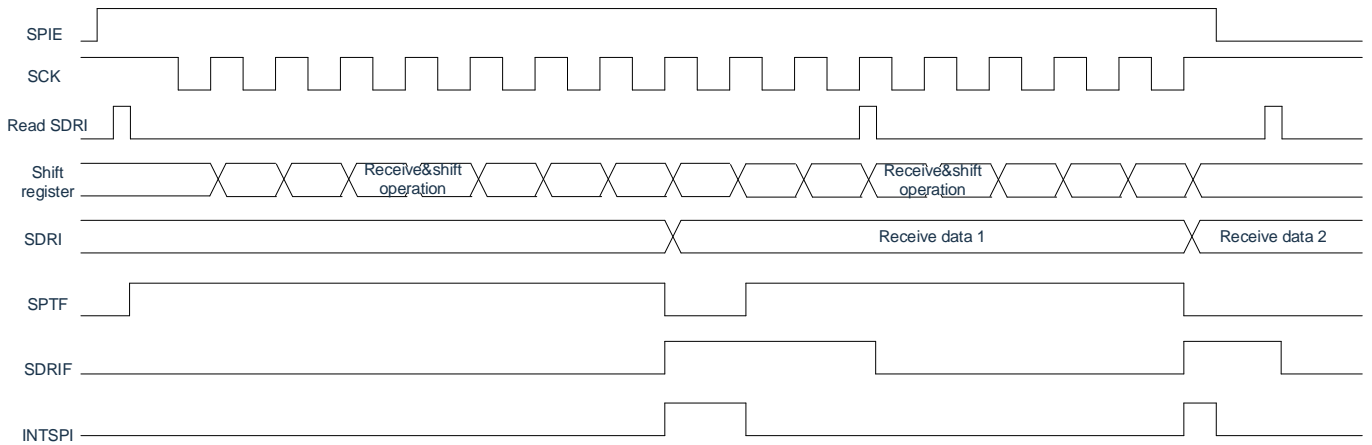
Figure 15-11 Stop steps for master reception



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) Processing flow

Figure 15-12 Timing diagram of reception (DAP=0, CKPmn=0)



15.4.3 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) Operation steps

Figure 15-13 Initial setup steps for slave transmission and reception

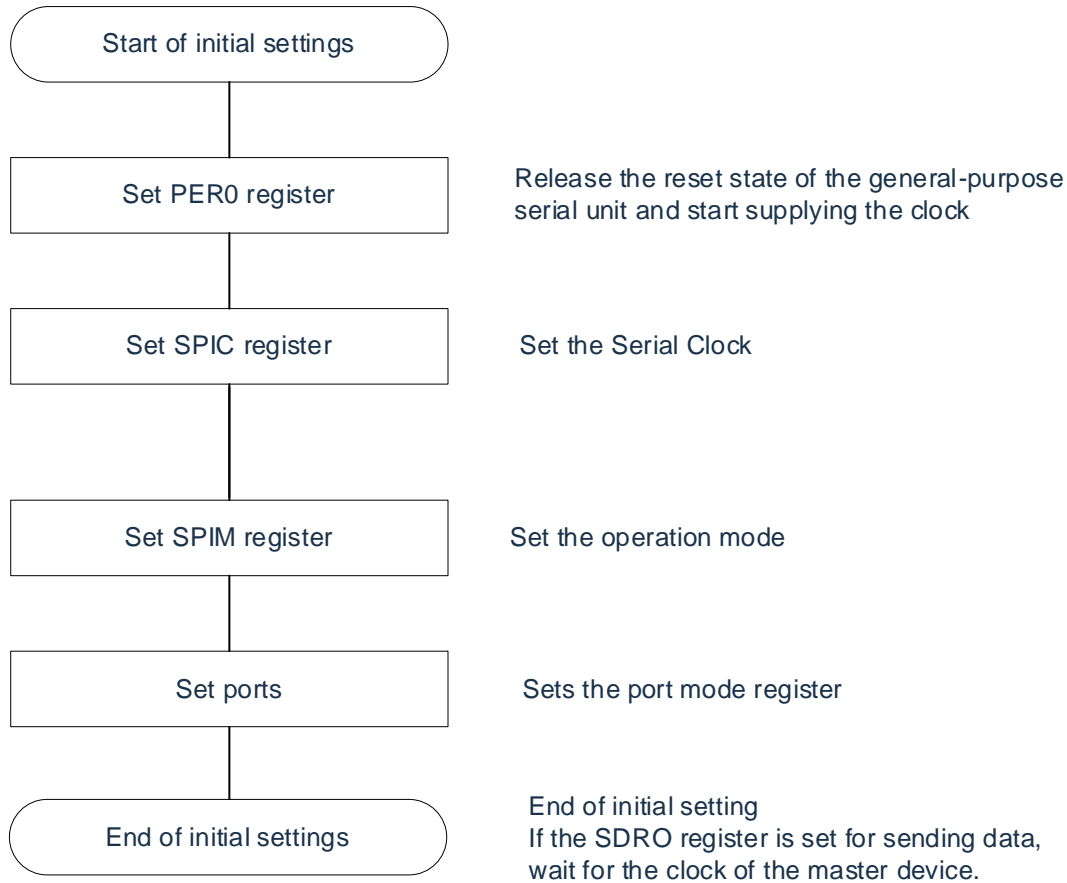
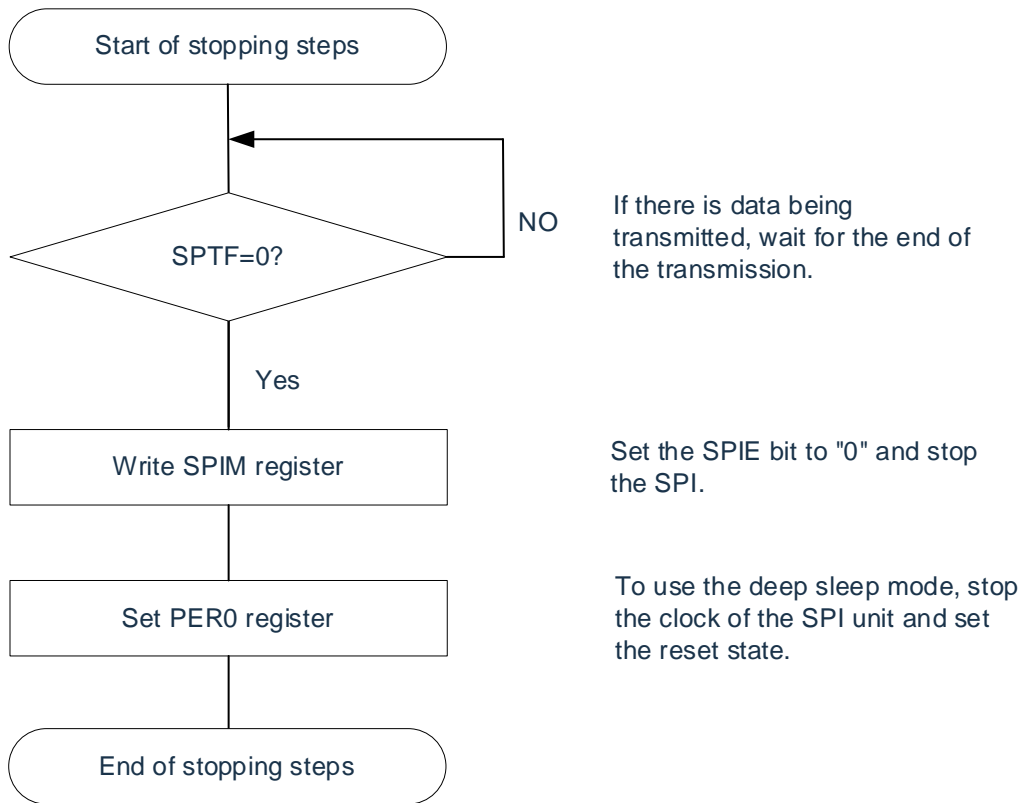


Figure 15-14 Stop steps for slave transmission and reception



(2) Processing flow

Figure 15-15 Timing diagram for transmit/receive (single transmission mode) (INTMD=0, DAP=0, CKPmn=0)

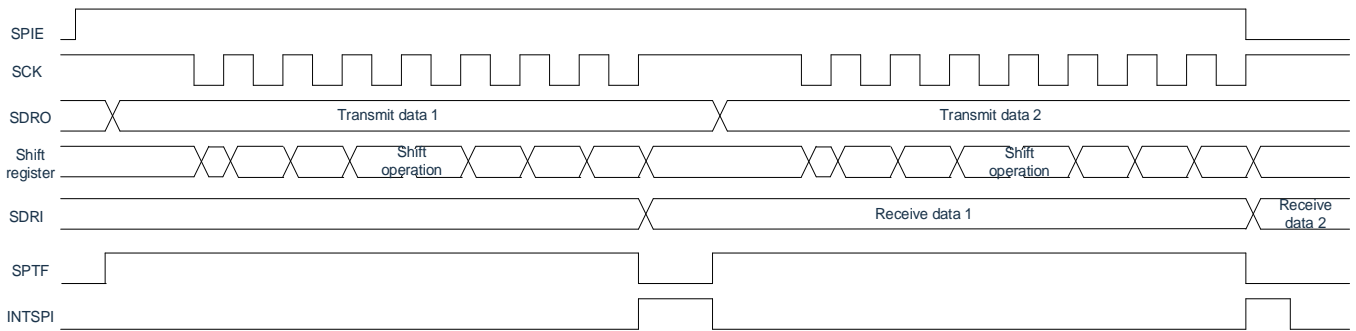
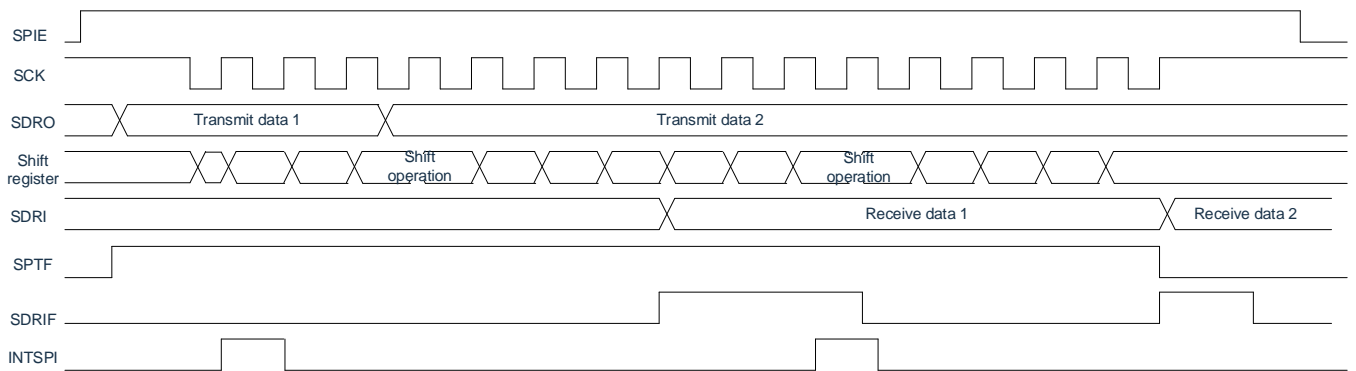


Figure 15-16 Timing diagram for transmit/receive (continuous transmit mode) (INTMD=1, DAP=0, CKPmn=0)



15.4.4 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) Operation steps

Figure 15-17 Initial setup steps for slave reception

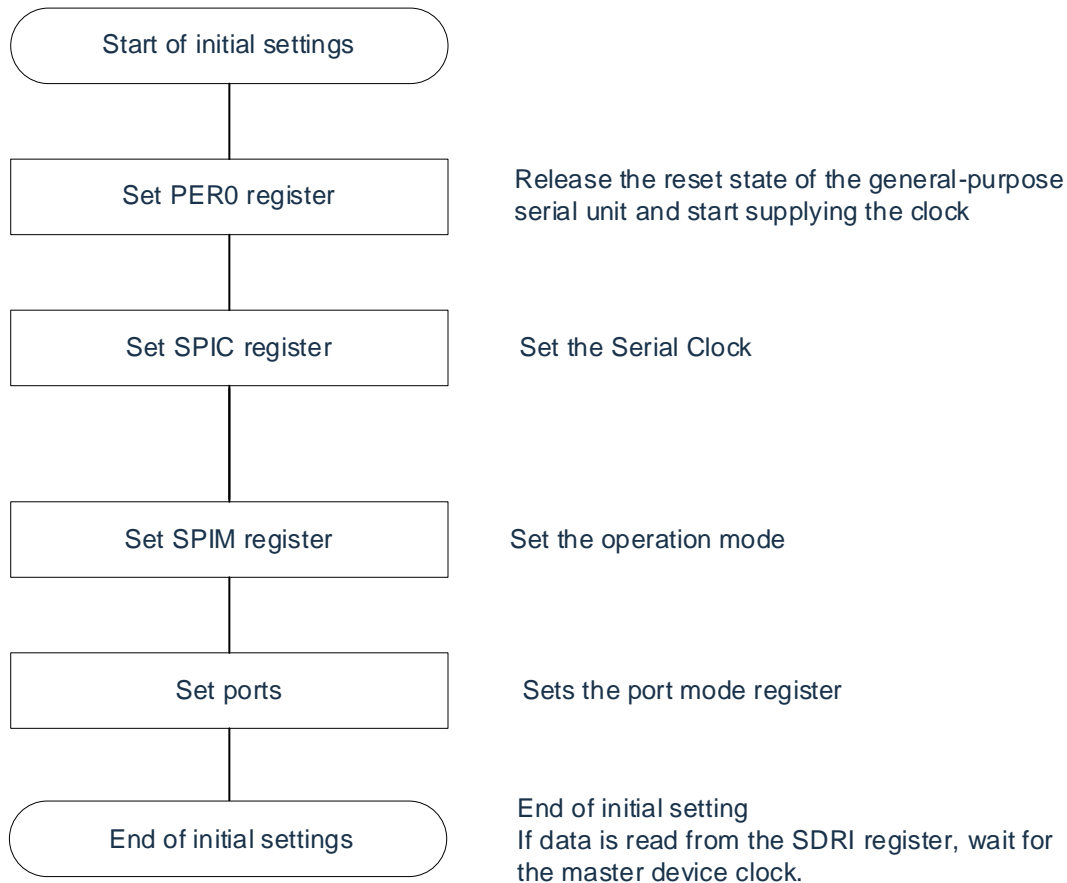
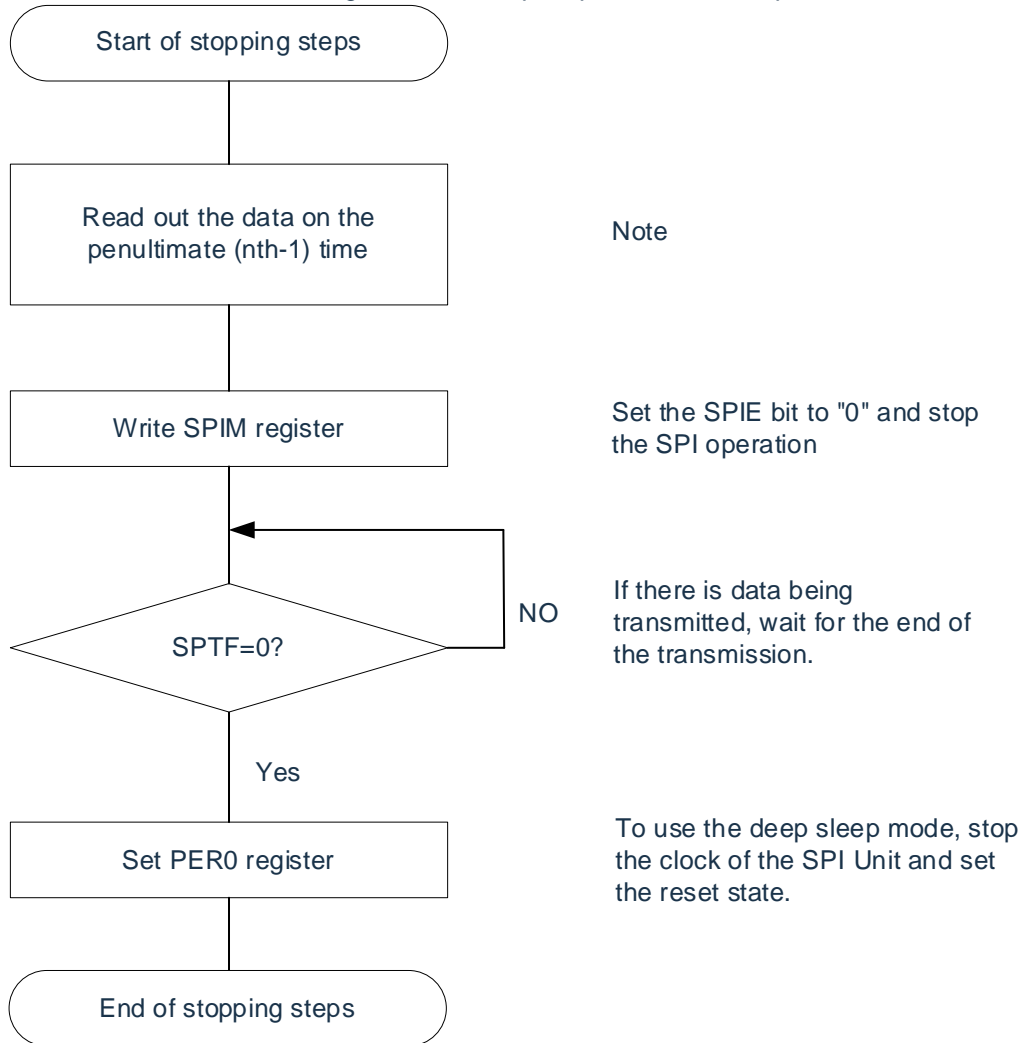


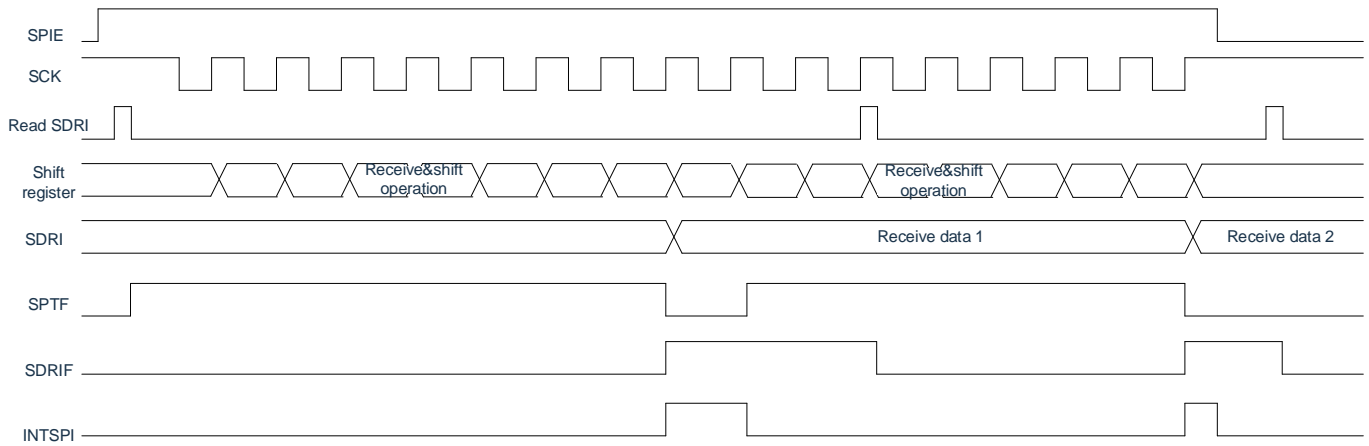
Figure 15-18 Stop steps for slave reception



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) Processing flow

Figure 15-19 Timing diagram for reception (DAP=0, CKPmn=0)



Chapter 16 Serial Interface IICA

16.1 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) I²C 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 I²C-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 I²C-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.

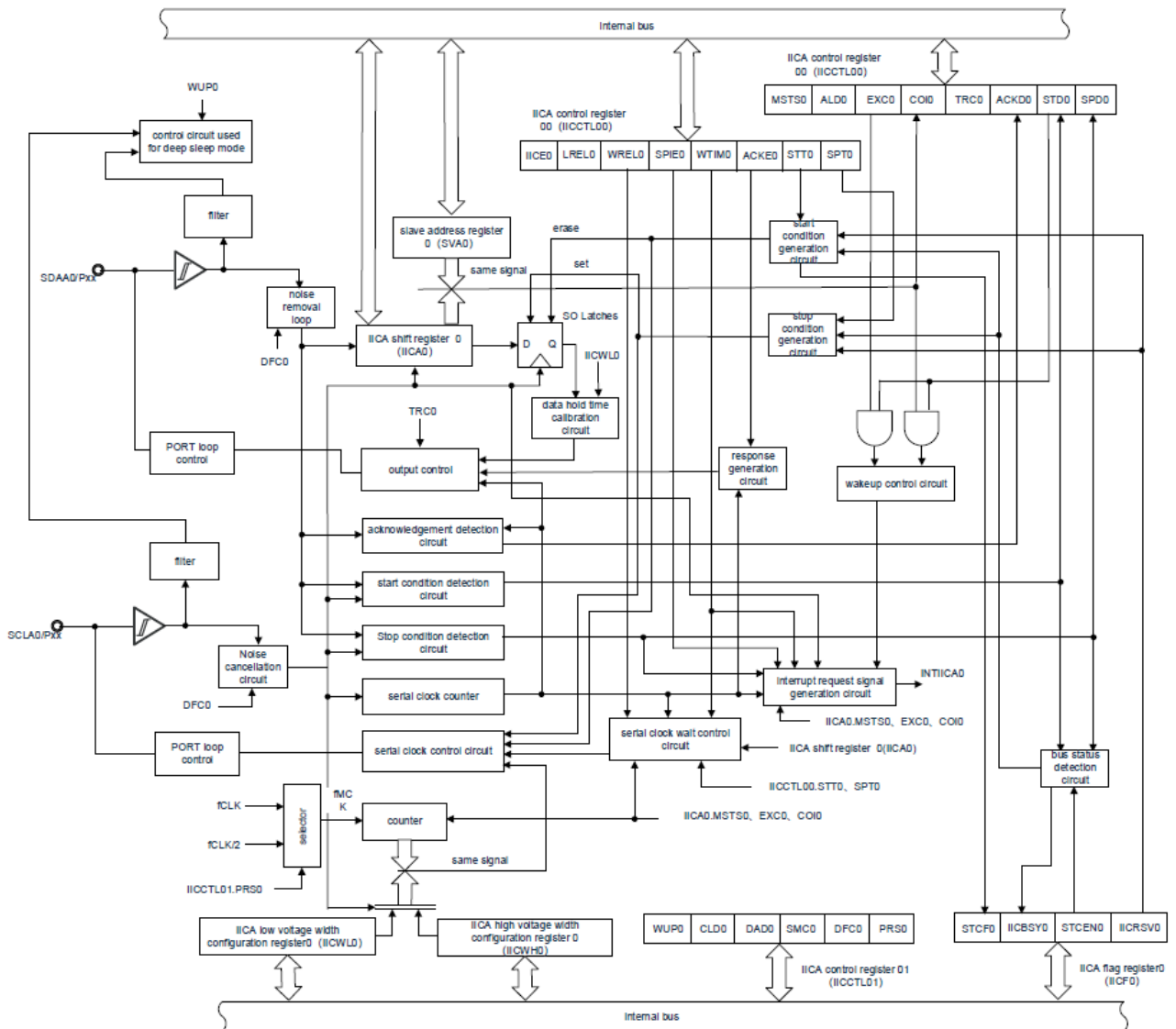
(3) Wake-up mode

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 16-1.

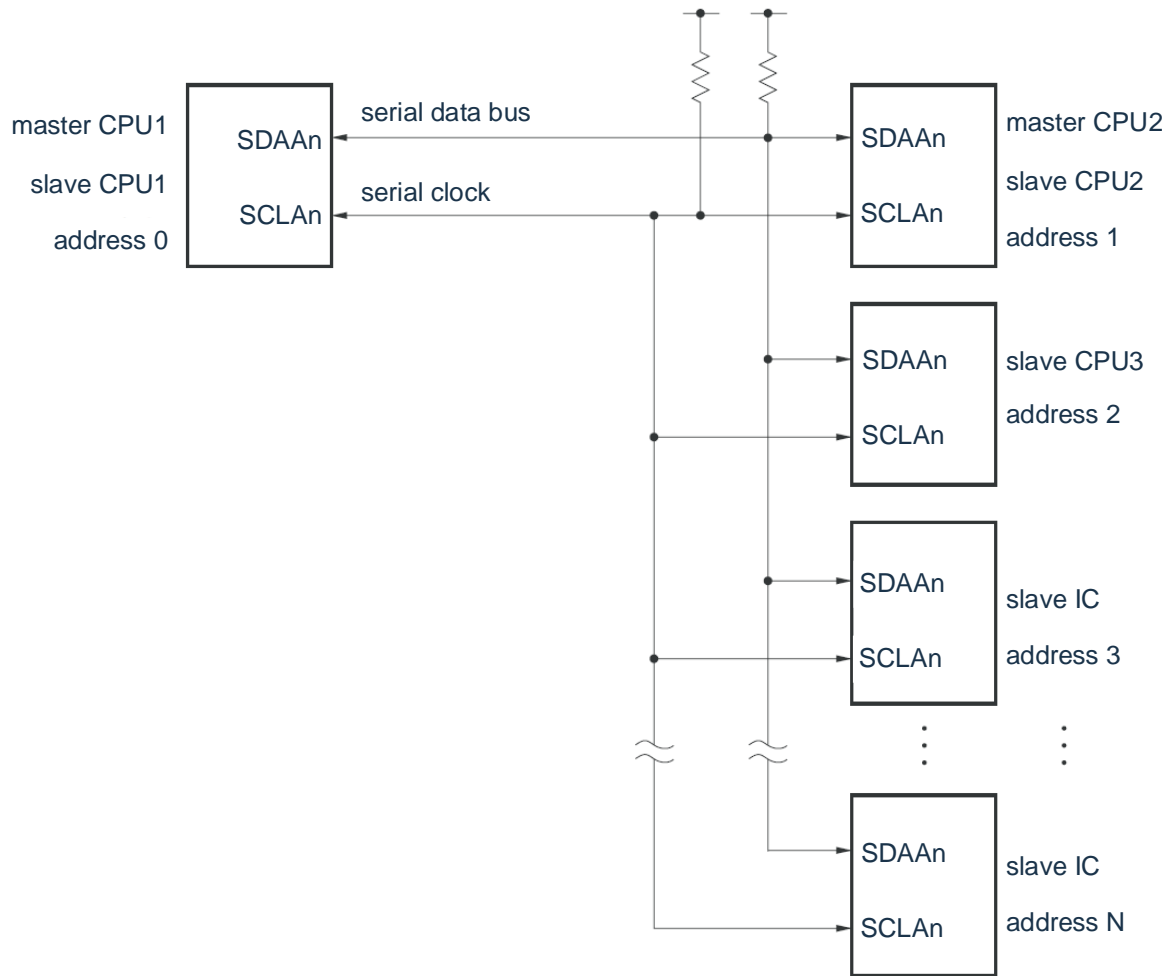
Remark: n=0

Figure 16-1 Block diagram of serial interface IICA



An example of the structure of a serial bus is shown in Figure 16-2.

Figure 16-2 Example of a serial bus structure for I2C bus



Remarkn=0

16.2 Structure of serial interface IICA

The serial interface IICA consists of the following hardware.

Table 16-1 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:

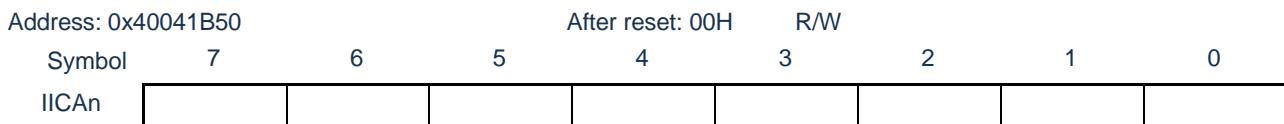
1. n=0
2. This product can multiplex the IICA input/output pin function to multiple ports. When a port is configured for multiplexing on the IICA pin, the N-channel open-drain output (V_{DD}/EV_{DD} withstand voltage) mode of the port is designed to automatically open, i.e. the POMxx register does not 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".

Figure 16-3 Format of IICA shift register n (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 a 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".

Figure 16-4 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 I²C 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)
SPTn bit: Bit0 of IICA control register n0 (IICCTLn0)
IICRSVn bit: Bit0 of IICA flag register n (IICFn)
IICBSYn bit: Bit6 of IICA flag register n (IICFn)
STCFn bit: Bit7 of IICA flag register n (IICFn)
STCENn bit: Bit1 of IICA flag register n (IICFn)
2. n=0

16.3 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)

Remark n=0

16.3.1 Peripheral enable register 0 (PER0)

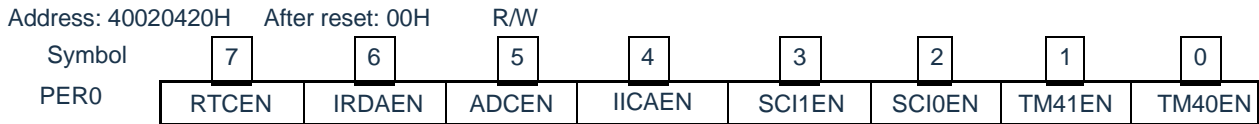
The PER0 register is the register that sets whether to enable or disable the supply of clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocks to hardware that is not in use.

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".

Figure 16-5 Format of peripheral enable register 0(PER0)



IICAnEN	Control of input clocks supplied to serial interface IICA
0	Stops input clock supply. • SFR used by the serial interface IICA cannot be written • The serial interface IICA is in the reset status.
1	Enables input clock supply. • SFR 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)).

- 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)

Remark n=0

16.3.2 IICA control register n0 (IICCTLn0)

This is a register that enables or stops I²C operation, sets the wait sequence, and sets other registers for I²C 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

Figure 16-6 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 ² C operation enable
0	Stops operation. Reset the IICA status register n (IICSn) ^{Note 1} and stop internal operation.
1	Enables 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)
<ul style="list-style-type: none"> • Cleared by instructions. • When resetting 	<ul style="list-style-type: none"> • Set by instructions

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.	
<ul style="list-style-type: none"> • 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)
<ul style="list-style-type: none"> • Automatically clears after execution. • When resetting 	<ul style="list-style-type: none"> • Set by instructions

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)
<ul style="list-style-type: none"> • Automatically cleared after execution. • When resetting 	<ul style="list-style-type: none"> • Set by instructions

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).

2: In the state where the IICEn bit is "0", the signal for this bit is invalid.

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 I²C 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 I²C to run (IICEn=1).

Remark: n=0

Figure 16-6 Format of IICA control register n0(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)
<ul style="list-style-type: none"> • Cleared by intructions. • When resetting 		<ul style="list-style-type: none"> • 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)
<ul style="list-style-type: none"> • Cleared by intructions. • When resetting 		<ul style="list-style-type: none"> • 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)
<ul style="list-style-type: none"> • Cleared by intructions. • When resetting 		<ul style="list-style-type: none"> • 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.

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

Figure 16-6 Format of IICA control register n0(IICCTLn0) (3/4)

STTn ^{Note1,2}	Triggering of the start condition				
0	No start conditions are generated.				
1	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: <ul style="list-style-type: none"> • When the communication reservation function is enabled (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. 				
Notices on setting timing: <ul style="list-style-type: none"> • 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. 					
<table border="1" style="width:100%; border-collapse: collapse;"> <thead> <tr> <th style="width:50%;">Clear condition (STTn=0)</th> <th style="width:50%;">Set condition (STTn=1)</th> </tr> </thead> <tbody> <tr> <td> <ul style="list-style-type: none"> • 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> <ul style="list-style-type: none"> • Set by intructions </td> </tr> </tbody> </table>		Clear condition (STTn=0)	Set condition (STTn=1)	<ul style="list-style-type: none"> • 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 	<ul style="list-style-type: none"> • Set by intructions
Clear condition (STTn=0)	Set condition (STTn=1)				
<ul style="list-style-type: none"> • 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 	<ul style="list-style-type: none"> • Set by intructions 				

Note 1: In the state where the IICEn bit is "0", the signal for this bit is invalid.

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)
STCFn: Bit7 of IICA flag register n (IICFn)
3. n=0

Figure 16-6 Format of IICA control register n0(IICCTLn0) (4/4)

SPTn ^{Note}	Trigger of stop condition	
0	No stop conditions are generated.	
1	Generates a stop condition (end of transfer as master).	
Notices on setting timing: <ul style="list-style-type: none"> • 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)
<ul style="list-style-type: none"> • 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 		<ul style="list-style-type: none"> • Set by instructions

Note: 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

16.3.3 IICA status register n(IICSn)

This is a register that indicates the I²C 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:

STTn: Bit1 of IICA control register n0 (IICCTLn0)

WUPn: Bit7 of IICA control register n1 (IICCTLn1)

Figure 16-7 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	MSTS _n	ALD _n	EXC _n	COL _n	TRC _n	ACKD _n	STD _n	SPD _n

MSTS _n	Acknowledgement flag of master control status	
0	Slave or communication idle status	
1	Master communication status	
Clear condition (MSTS _n =0)		Set condition (MSTS _n =1)
<ul style="list-style-type: none"> When a stop condition is detected When the ALD_n bit is "1" (arbitration fails). Cleared due to the LREL_n bit is "1" (Exit Communication). When the IICEn bit changes from "1" to "0" (stops running). When resetting 		<ul style="list-style-type: none"> When generating a start condition

ALD _n	Detection of arbitration failures	
0	It indicates that arbitration did not occur or was won.	
1	It indicates an arbitration failure. Clear the MSTS _n bit.	
Clear condition (ALD _n =0)		Set condition (ALD _n =1)
<ul style="list-style-type: none"> Automatically cleared after reading the IICSn register Note. When the IICEn bit changes from "1" to "0" (stops running). When resetting 		<ul style="list-style-type: none"> When arbitration fails

Note: This bit is cleared even if the bit memory manipulation instruction is executed on a bit other than the IICSn register.

Therefore, when using ALD_n bits, the data for the ALD_n bits must be read before reading other bits.

Remark:

- LREL_n: Bit6 of IICA control register n0 (IICCTLn0)
IICEn: Bit7 of IICA control register n0 (IICCTLn0)
- n=0

Figure 16-7 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)
<ul style="list-style-type: none"> 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 		<ul style="list-style-type: none"> 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)
<ul style="list-style-type: none"> 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 		<ul style="list-style-type: none"> 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> <ul style="list-style-type: none"> 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 operation). Cleared because the WRELn bit is "1" (released from wait) Note <ul style="list-style-type: none"> When the ALDn bit changes from "0" to "1" (arbitration fails) When resetting When not participating in communications (MSTSn, EXCn, COIn=0) <Master> <ul style="list-style-type: none"> When "1" is output in the LSB (transmission direction indication bit) of the 1st byte <Slave> <ul style="list-style-type: none"> When a start condition is detected When "0" is input to the LSB (transmission direction indication bit) of the 1st byte 		<Master> <ul style="list-style-type: none"> When generating a start condition When the LSB (transmission direction indicator bit) of byte 1 (address transmission) outputs "0" (master transmit). <Slave> <ul style="list-style-type: none"> When the LSB (transmission direction indication bit) in byte 1 (address transmission) of the master device inputs "1" (slave transmit)

Note: 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:

- LRELn: Bit6 of IICA control register n0 (IICCTLn0)
IICEn: Bit7 of IICA control register n0 (IICCTLn0)
- n=0

Figure 16-7 Format of IICA status register n (IICS_n) (3/3)

ACKD _n	Detection of acknowledge	
0	No Ack was detected.	
1	An Ack was detected.	
Clear condition (ACKD _n =0)		Set condition (ACKD _n =1)
<ul style="list-style-type: none"> • When resetting • When a stop condition is detected • When the next byte of the 1st clock rises • Cleared because the LREL_n bit is "1" (exits communication). • When the IICEn bit changes from "1" to "0" (stops operation). • When resetting 		<ul style="list-style-type: none"> • When the SDAAn line is set low on the 9th clock rising edge of the SCLAn line

STD _n	Detection of starting conditions	
0	No start condition detected.	
1	A start condition is detected, indicating that it is during address transfer.	
Clear condition (STD _n =0)		Set condition (STD _n =1)
<ul style="list-style-type: none"> • When a stop condition is detected • When the 1st clock rises after the next byte of the address is transmitted • Cleared because the LREL_n bit is "1" (exits communication). • When the IICEn bit changes from "1" to "0" (stops operation). • When resetting 		<ul style="list-style-type: none"> • When a start condition is detected

SPD _n	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 (SPD _n =0)		Set condition (SPD _n =1)
<ul style="list-style-type: none"> • After setting this bit, the address transmits the byte after the start condition is detected when the clock rises • When the WUP_n bit changes from "1" to "0" • When the IICEn bit changes from "1" to "0" (stops operation). • When resetting 		<ul style="list-style-type: none"> • When a stop condition is detected

Remark:

1. LREL_n: Bit6 of IICA control register n0 (IICCTL_n0)
IICEn: Bit7 of IICA control register n0 (IICCTL_n0)
2. n=0

16.3.4 IICA flag register n(IICF_n)

This is a register that sets the I²C operating mode and indicates the status of the I²C-bus.

The IICF_n register is set by an 8-bit memory manipulation instruction. However, only the STT_n clear flag (STCF_n) and the I²C-bus status flag (IICBSY_n) can be read.

The communication appointment function is enabled or disabled by the IICRSV_n bit setting, and the initial value of the IICBSY_n bit is set by the STCEN_n bit. Only bit7 (IICEn) = 0 can only write IICRSV_n bits and STCEN_n bits. After allowing operation, only the IICF_n registers can be read. After generating a reset signal, the value of this register changes to "00H".

Figure 16-8 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)	
<ul style="list-style-type: none"> • Cleared due to the STTn bit is "1" • When the ICEn bit is "0" (stop operation). • When resetting 	
Set condition (STCFn=1)	
<ul style="list-style-type: none"> • 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	I²C 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)	
<ul style="list-style-type: none"> • When a stop condition is detected • When the ICEn bit is "0" (stop operation) • When resetting 	
Set condition (IICBSYn=1)	
<ul style="list-style-type: none"> • When a start condition is detected • Sets the ICEn bit when the STCENn bit is "0" 	

STCENn	Initial start enable triggering
0	After enabling operation (ICEn=1), a start condition is allowed to be generated by detecting a stop condition.
1	After enabling operation (ICEn=1), the start condition is allowed to be generated without detecting a stop condition.
Clear condition (STCENn=0)	
<ul style="list-style-type: none"> • Cleared by intructions. • When a start condition is detected • When resetting 	
Set condition (STCENn=1)	
<ul style="list-style-type: none"> • Set by intructions 	

IICRSVn	Communication reservation function disable bit
0	Communication appointments are enabled.
1	Communication reservations are disabled.
Clear condition (IICRSVn=0)	
<ul style="list-style-type: none"> • Cleared by intructions. • When resetting 	
Set condition (IICRSVn=1)	
<ul style="list-style-type: none"> • 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 (ICEn=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 (ICIn=0).

Remark:

1. STTn: Bit1 of IICA control register n0 (IICCTLn0)
2. ICEn: Bit7 of IICA control register n0 (IICCTLn0)

16.3.5 IICA control register n1(IICCTLn1)

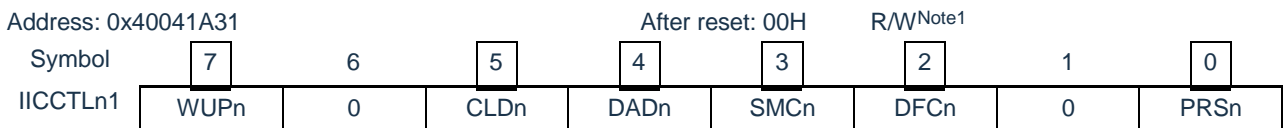
This is a register used to set the I²C operating mode and to detect the status of the SCLAn pin and SDAAn pin.

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 I²C 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".

Figure 16-9 Format of IICA control register n1 (IICCTLn1) (1/2)



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").

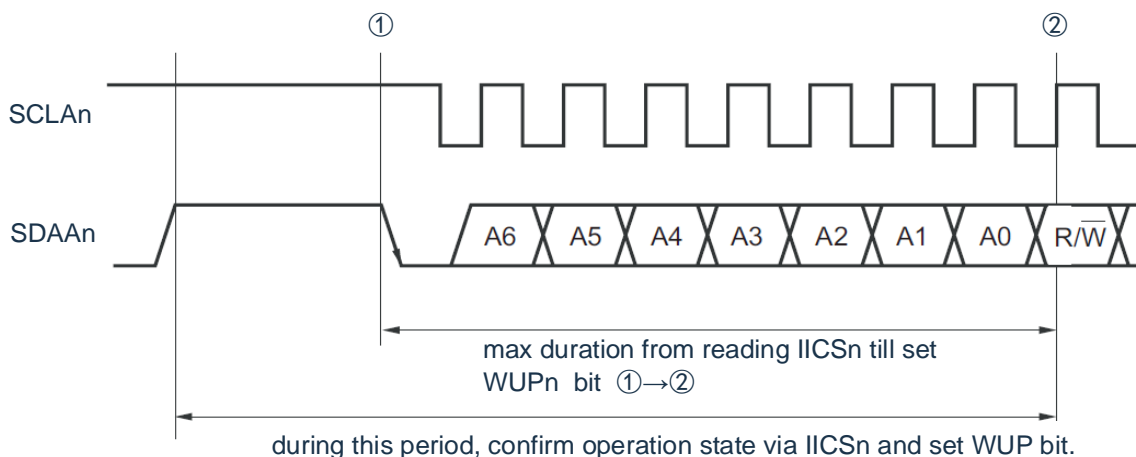
In the state of WUPn bit "1", the interrupt timing when the address is matched or the extension code is received is the same as the interrupt timing when WUPn bit is "0".

(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.

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

Figure 16-9 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)
<ul style="list-style-type: none"> • When the SCLAn pin is low • When the IICEn bit is "0" (stops operation) • When resetting 		<ul style="list-style-type: none"> • 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)
<ul style="list-style-type: none"> • When the SDAAn pin is low • When the IICEn bit is "0" (stops operation) • When resetting 		<ul style="list-style-type: none"> • 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} (1MHz ≤ F _{CLK} ≤ 20MHz).
1	Selects F _{CLK} /2 (20MHz < F _{CLK}).

Notice:

1. The maximum operating frequency of the IICA running clock (F_{MCK}) is 20MHz (Max.). Bit0 (PRSn) of the IICA control register n1 (IICCTLn1) must be set to "1" only if the F_{CLK} exceeds 20MHz.
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

16.3.6 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.

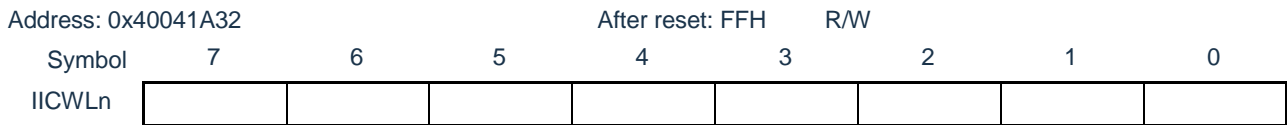
The IICWLn register is set by an 8-bit memory manipulation instruction.

The IICWLn register must be set when I²C 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 "16.4.2 Setting transfer clock via IICWLn and IICWHn registers".

The data retention time is 1/4 of the time set by IICWLn.

Figure 16-10 Format of IICA low-level width setting register n (IICWLn)

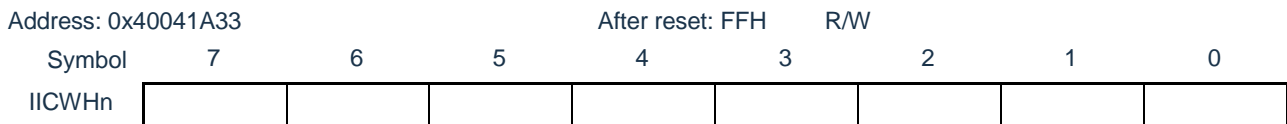


16.3.7 IICA high-level width setting register n(IICWHn)

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 I²C 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".

Figure 16-11 Format of high level width setting register n(IICWHn)



Remark 1. For the method of setting the clock transmitted by the main controller, please refer to 16.4.2(1); for how to set the slave IICWLn register and the IICWHn register, refer to 16.4.2(2).

2. n=0

16.3.8 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 Pin Functions".

16.4 Function of I²C-bus mode

16.4.1 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

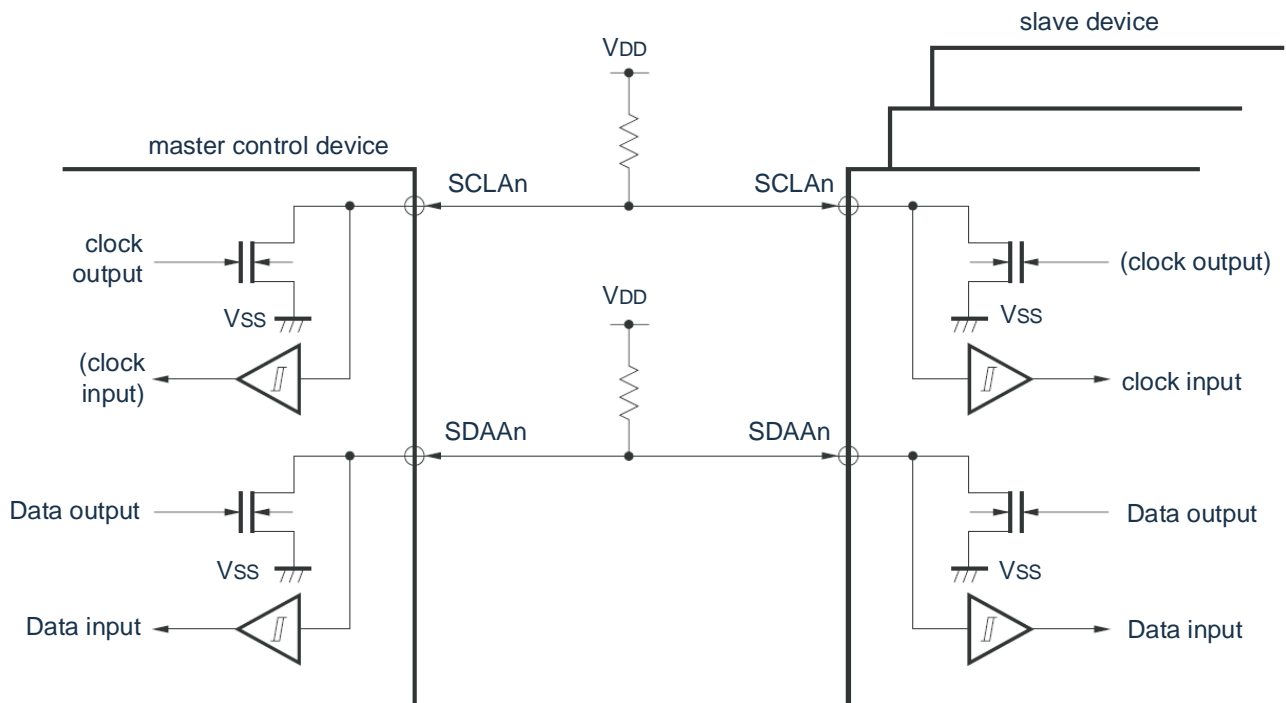
The outputs of both the master and slave devices are N-channel open-drain outputs, and the inputs are Schmidt inputs.

(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 16-12 Block diagram of pin structure



Remark n=0

16.4.2 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.)

- Fast mode

$$IICWLn = \frac{0.52}{\text{transfer clock}} \times F_{MCK}$$

$$IICWHn = \left(\frac{0.48}{\text{transfer clock}} - T_R - T_F \right) \times F_{MCK}$$

- Standard mode

$$IICWLn = \frac{0.47}{\text{transfer clock}} \times F_{MCK}$$

$$IICWHn = \left(\frac{0.53}{\text{transfer clock}} - T_R - T_F \right) \times F_{MCK}$$

- Enhanced fast mode

$$IICWLn = \frac{0.50}{\text{transfer clock}} \times F_{MCK}$$

$$IICWHn = \left(\frac{0.50}{\text{transfer clock}} - T_R - T_F \right) \times F_{MCK}$$

(2) Setting IICWLn and IICWHn registers on slave side

(The fractional parts of all setting values are rounded up.)

- Fast mode

$$IICWLn = 1.3\mu s \times F_{MCK}$$

$$IICWHn = (1.2\mu s - T_R - T_F) \times F_{MCK}$$

- Standard mode

$$IICWLn = 4.7\mu s \times F_{MCK}$$

$$IICWHn = (5.3\mu s - T_R - T_F) \times F_{MCK}$$

- Enhanced fast mode

$$IICWLn = 0.50\mu s \times F_{MCK}$$

$$IICWHn = (0.50\mu s - T_R - T_F) \times F_{MCK}$$

Notice:

1. The maximum operating frequency of the IICA operating clock (F_{MCK}) is 20MHz (Max.). Bit0 (PRSn) of the IICA control register n1 (IICCTLn1) must be set to "1" only when the F_{CLK} exceeds 20MHz.
2. Note the minimum F_{CLK} operation frequency when setting the transfer clock. The minimum F_{CLK} operation frequency for serial interface IICA is determined according to the mode.

Fast mode: $F_{CLK} = 3.5\text{MHz}(\text{Min.})$

Enhanced fast mode: $F_{CLK} = 10\text{MHz}(\text{Min.})$

Standard mode: $F_{CLK} = 1\text{MHz}(\text{Min.})$

Remark:

1. Calculate the rise time (T_R) and fall time (T_F) of the SDAAn and SCLAn signals separately, because they differ depending on the pull-up resistance and wire load.
2. IICWLn: IICA low-level width setting register n

IICWHn: IICA high-level width setting register n

T_F: SDAAn and SCLAn signal falling times

T_R: SDAAn and SCLAn signal rising times

F_{MCK}: IICA operation clock frequency

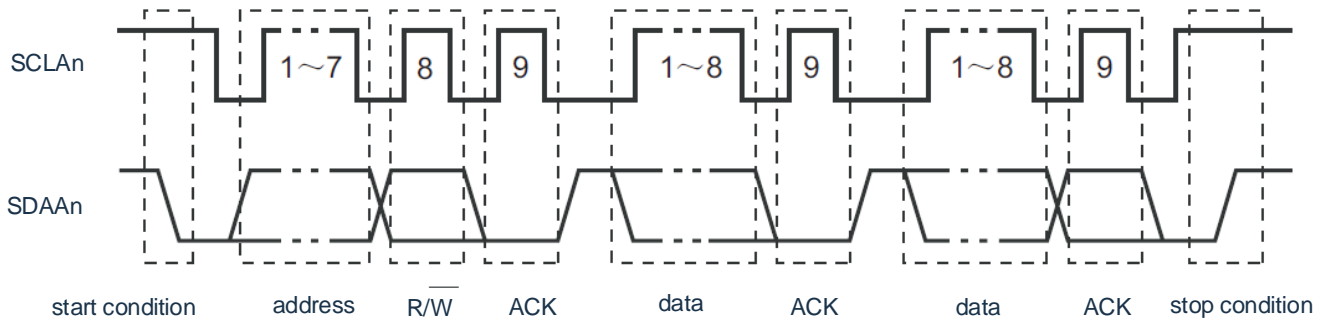
3. n=0

16.5 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 16-13 I²C bus serial data transfer timing



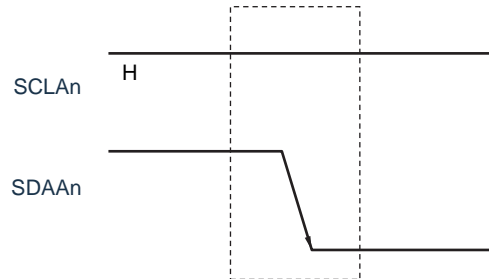
The master generates start conditions, slave addresses, and stop conditions.

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.

16.5.1 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 16-14 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

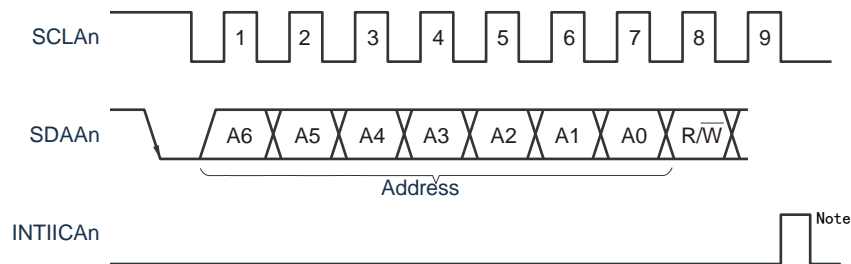
16.5.2 Address

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 16-15 Address



Note: If data other than the local station address or extension code is received while the slave is running, INTIICAn is not generated.

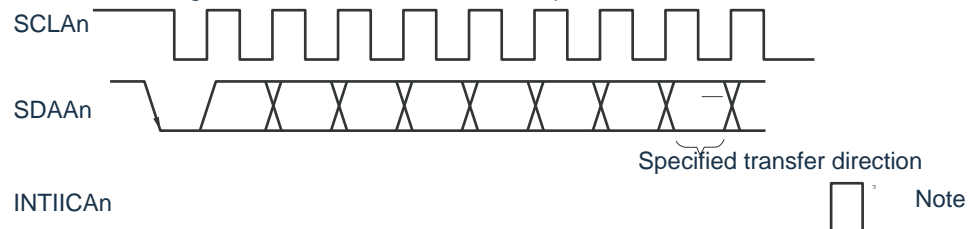
If the 8-bit data consisting of the slave address and the transfer direction described in “16.5.3 Transfer direction specification” is written to the IICA shift register n (IICAn), the address is output. The received address is written to the IICAn register. The slave address is assigned to the higher 7 bits of the IICAn register.

16.5.3 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 16-16 Transfer direction specification



Note INTIICAn is not issued if data other than a local address or extension code is received during slave device operation.

Remark n=0

16.5.4 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.

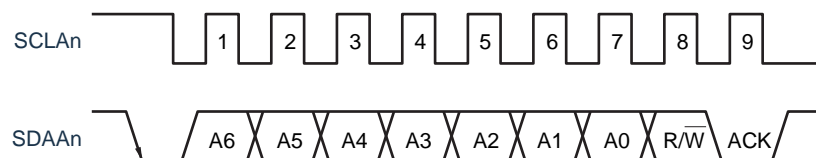
To generate ACK, the reception side makes the SDAAn line low at the ninth clock (indicating normal reception).

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 16-17 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.

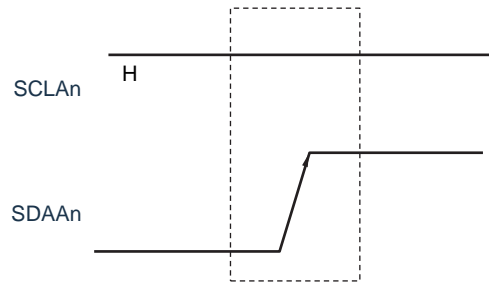
- When 8-clock clock stretch state is selected (bit 3 (WTIMn) of IICCTLn0 register = 0): By setting the ACKEn bit to 1 before releasing the clock stretch state, ACK is generated at the falling edge of the eighth clock of the SCLAn pin.
- When 9-clock clock stretch state is selected (bit 3 (WTIMn) of IICCTLn0 register = 1): ACK is generated by setting the ACKEn bit to 1 in advance.

Remark n=0

16.5.5 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 16-18 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

16.5.6 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 16-19 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)

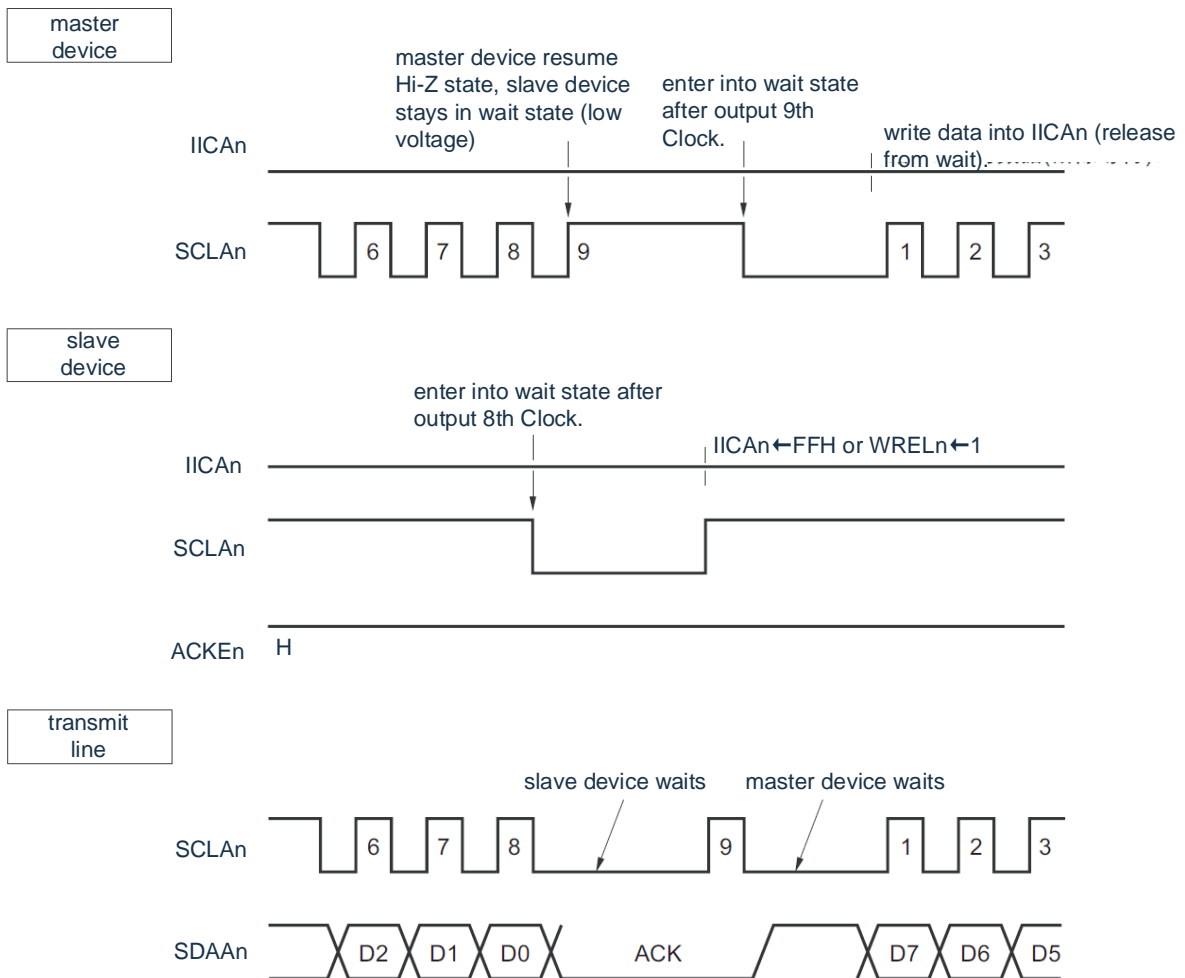
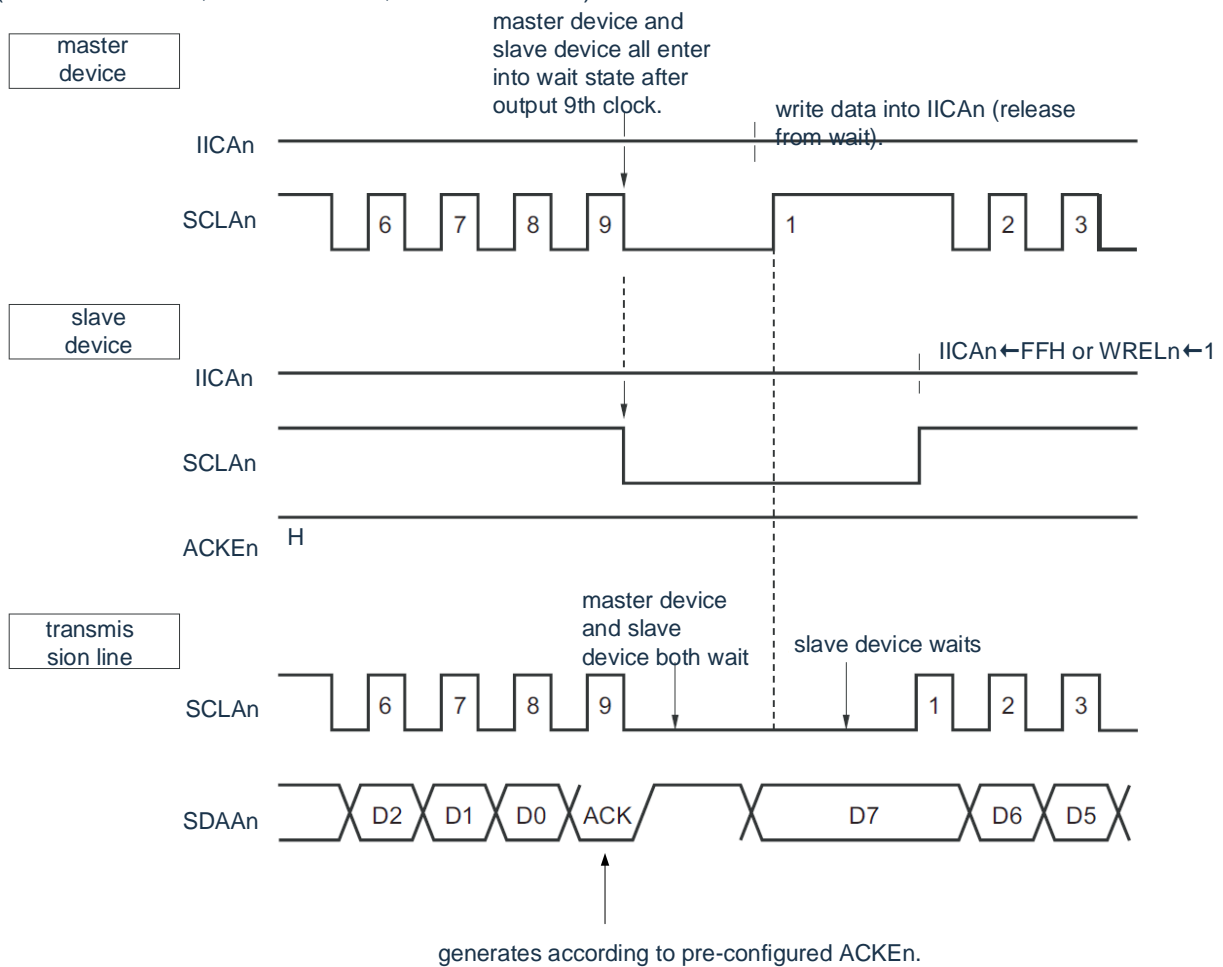


Figure 16-19 Wait (2/2)

(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

16.5.7 Method of releasing wait state

In general, I²C can release the wait with the following processing.

- Writes data to IICA shift register n (IICAn).
- Sets the bit5 (WRELn) of the IICA control register n0 (IICCTLn0) (wait released).
- Sets the bit1 (STTn) of the IICCTLn0 register (generates a start condition) ^{Note}.
- 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.

Notice If the wait release process is performed when the WUPn bit is "1", the wait will not be released.

Remark n=0

16.5.8 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 in Table 16-2 and wait control is performed.

Table 16-2 Generation timing and waiting control of INTIICAn

WTIMn	Slave operation			Master operation		
	Address	Data reception	Data transmission	Address	Data reception	Data transmission
0	g ^{Note1,2}	g ^{Note2}	g ^{Note2}	9	8	8
1	g ^{Note1,2}	g ^{Note2}	g ^{Note2}	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 INDICATIONn 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.

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

- 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.
- 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:

- Writes data to IICA shift register n (IICAn).
- Sets the bit5 (WRELn) of the IICA control register n0 (IICCTLn0) (wait released).
- Sets the bit1 (STTn) of the IICCTLn0 register (generates a start condition) ^{Note}.
- 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).

16.5.9 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.

16.5.10 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

16.5.11 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.

- High 4 bits data are the same: EXCn=1
- 7 bits of data are the same: COIn=1

Remark: EXCn: Bit5 of the IICA status register n

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 16-3 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 I²C bus specifications issued by NXP Semiconductors for details of extension codes other than those described above.
2. n=0

16.5.12 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: STDn: Bit1 of IICA status register n (IICSn)

STTn: Bit1 of IICA control register n0 (IICCTLn0)

Figure 16-20 Timing example of arbitration

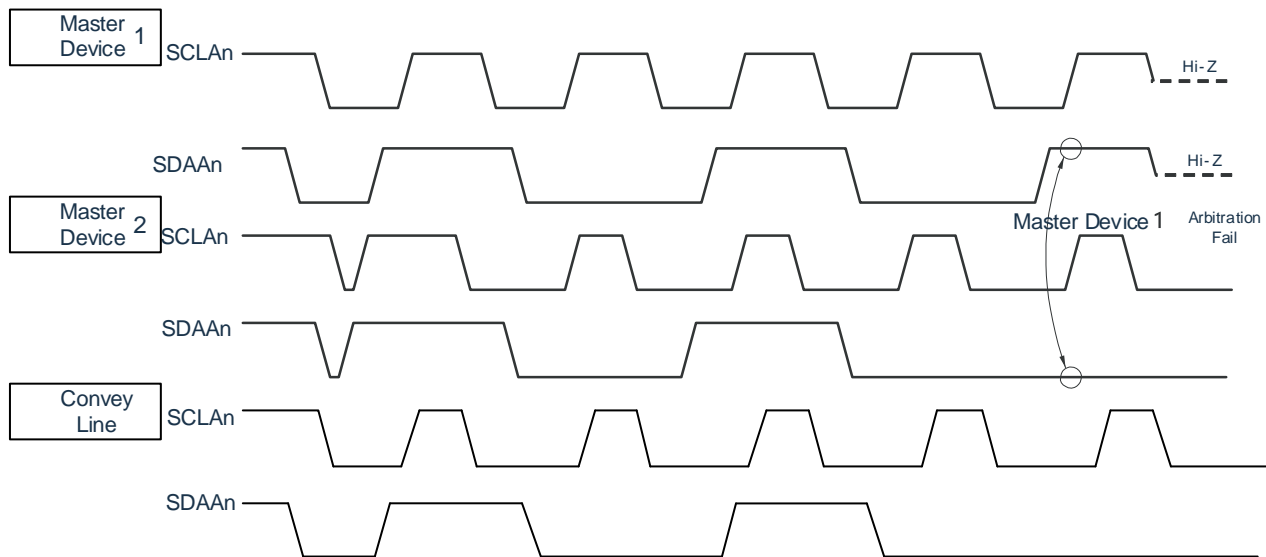


Table 16-4 Status at the time of arbitration and timing of generation of interrupt requests

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.	On the falling edge of the 8th or 9th clock after the byte is transmitted ^{Note1}

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.

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 (IICCTLn0)
2. n=0

16.5.13 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 16-21, and the flow when the WUPn bit is set to "0" by address matching is shown in Figure 16-22.

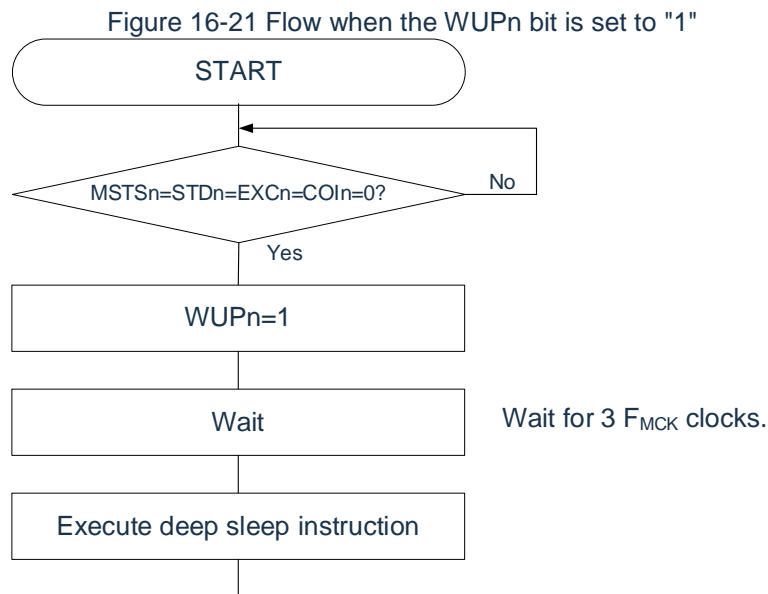
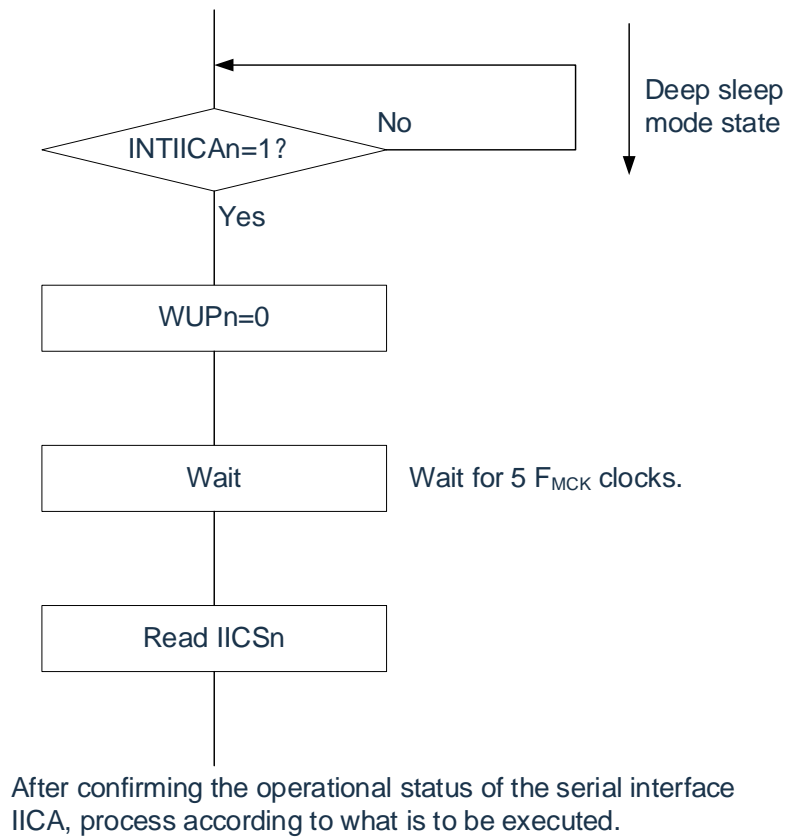


Figure 16-22 Flow when the WUPn bit is set to "0" by address matching (including receiving extension codes)

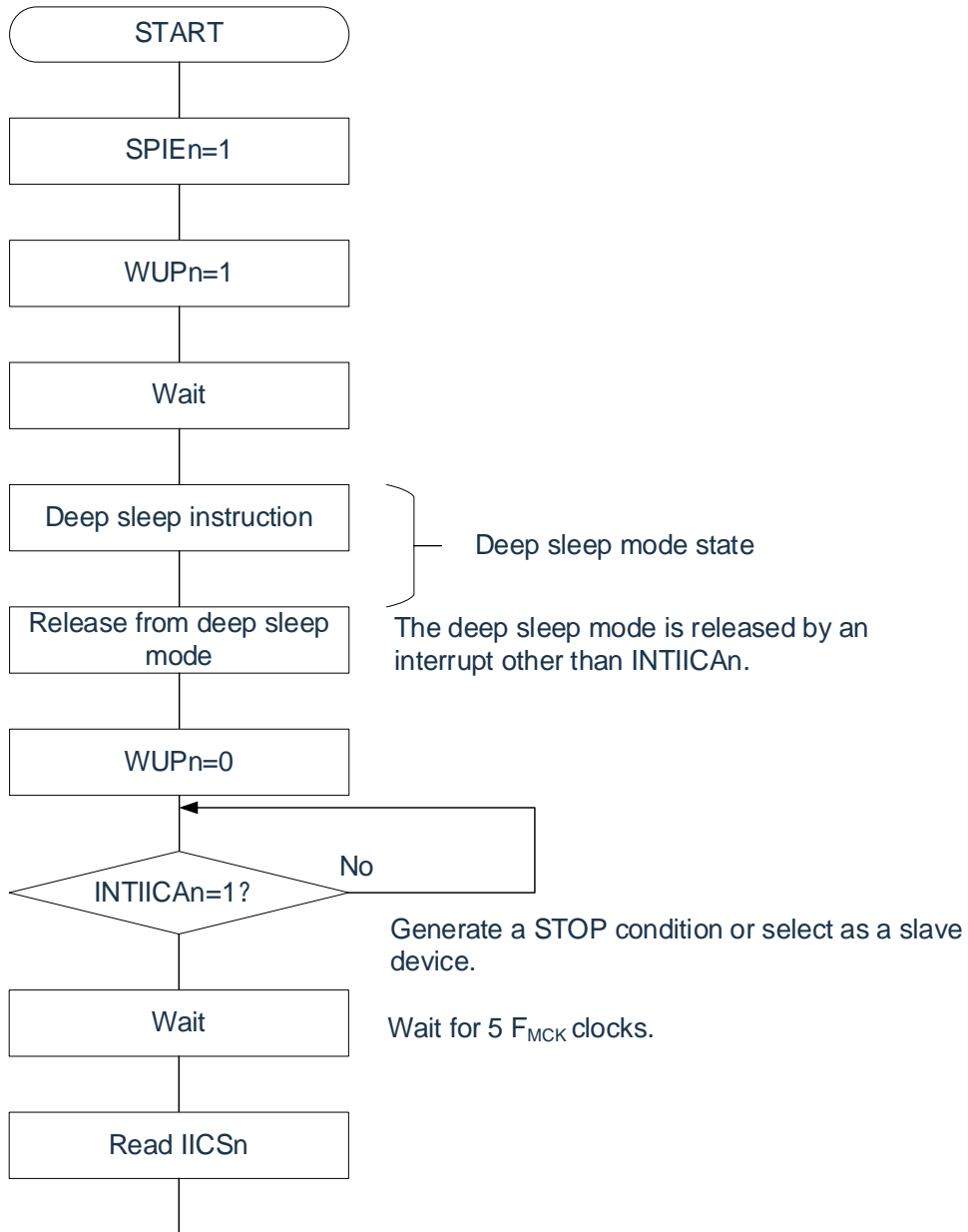


In addition to the interrupt request (INTIICAn) generated by the serial interface IICA, the deep sleep mode must be removed through the following procedure.

- Next IIC communication for the operation of the master device: the flow of Figure 16-23.
- Next IIC communication for slave device operation:
 Return via INTIICAn interrupt: same process as Figure 16-22.
 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 16-23 Operation as master device after being released from deep sleep mode by an interrupt other than INTIICAn



After confirming the operational status of the serial interface IICA, process according to what is to be executed.

Remark n=0

16.5.14 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.

- When the bus is released.....Generates a start condition
- When the bus is not released (standby state).....Communication reservation

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:

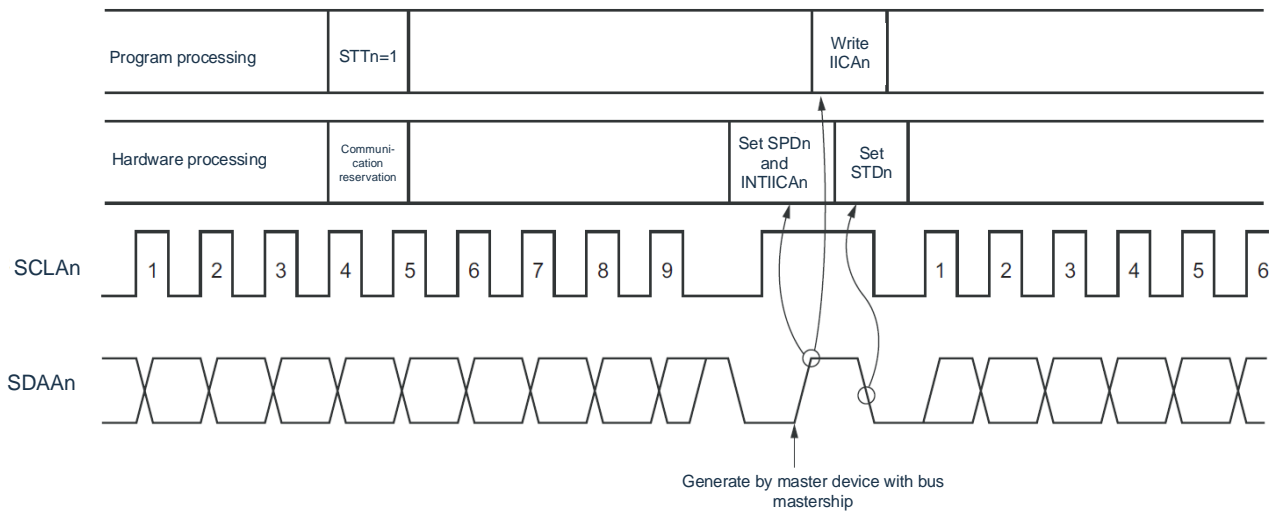
$$(IICWLn \text{ set value} + IICWHn \text{ set value} + 4) / F_{MCK} + T_F \times 2$$

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 16-24 Timing of communication reservation



Remark: IICAn: IICA shift register n

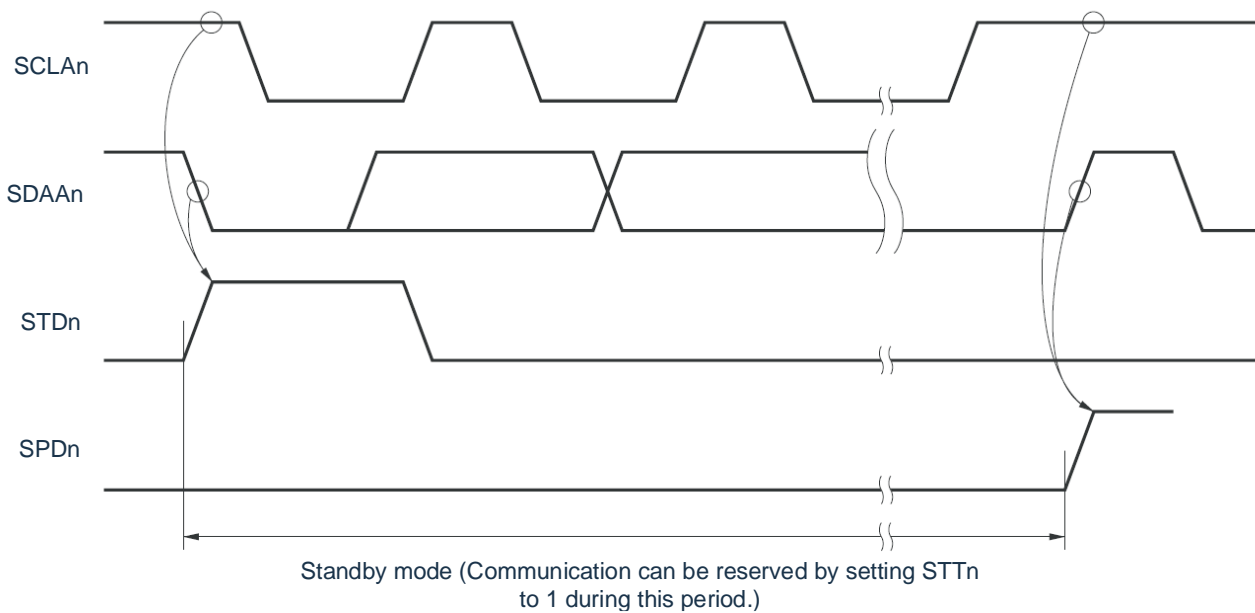
STTn: Bit1 of IICA control register n0 (IICCTLn0)

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 16-25. 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.

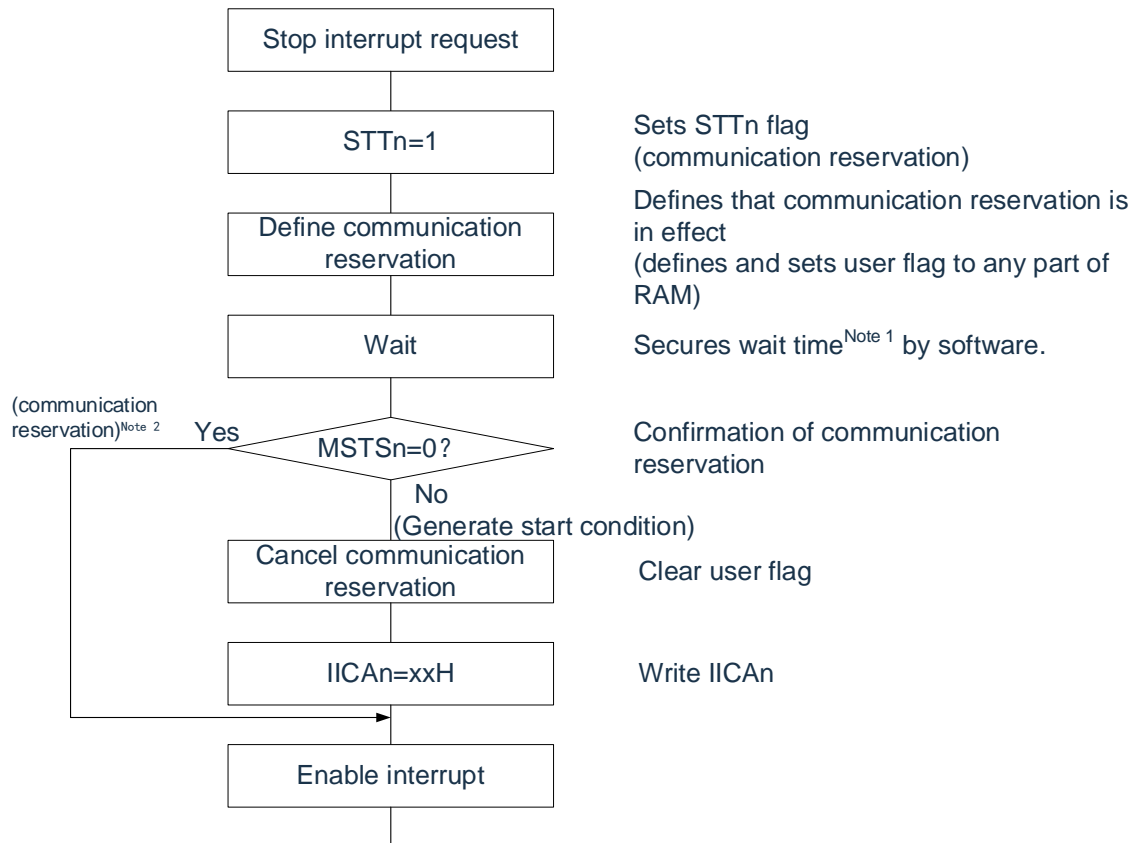
Figure 16-25 Timing for accepting communication reservations



Remark n=0

The steps for communication reservations are shown in Figure 16-26.

Figure 16-26 Steps for communication reservations



Note 1: The wait time is calculated as follows. $(IICWLn \text{ set value} + IICWHn \text{ set value} + 4) / F_{MCK} + T_F \times 2$

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
 TF: SDAAn and SCLAn signal falling times
 FMCK: IICA operation clock frequency
2. n=0

- (2) When communication reservation function is disabled (bit 0 (IICRSVn) of IICA flag register n (IICFn) = 1)
When bit 1 (STTn) of IICA control register n0 (IICCTLn0) is set to 1 when the bus is not used in a communication during bus communication, this request is rejected and a start condition is not generated. The following two statuses are included in the status where bus is not used.
- When arbitration results in neither master nor slave operation
 - When an extension code is received and slave operation is disabled (ACK is not returned and the bus was released by setting bit 6 (LRELn) of the IICCTLn0 register to 1 and saving communication)

To confirm whether the start condition was generated or request was rejected, check STCFn (bit 7 of the IICFn register). It takes up to 5 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

16.5.15 Cautions

(1) When $STCENn = 0$

Immediately after I²C 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 I²C 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 I²C communications are already in progress

If I²C 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 I²C communications. To avoid this, start I²C 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 I²C.
- 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

16.5.16 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 I²C 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 I²C 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 I²C bus slave is shown below.

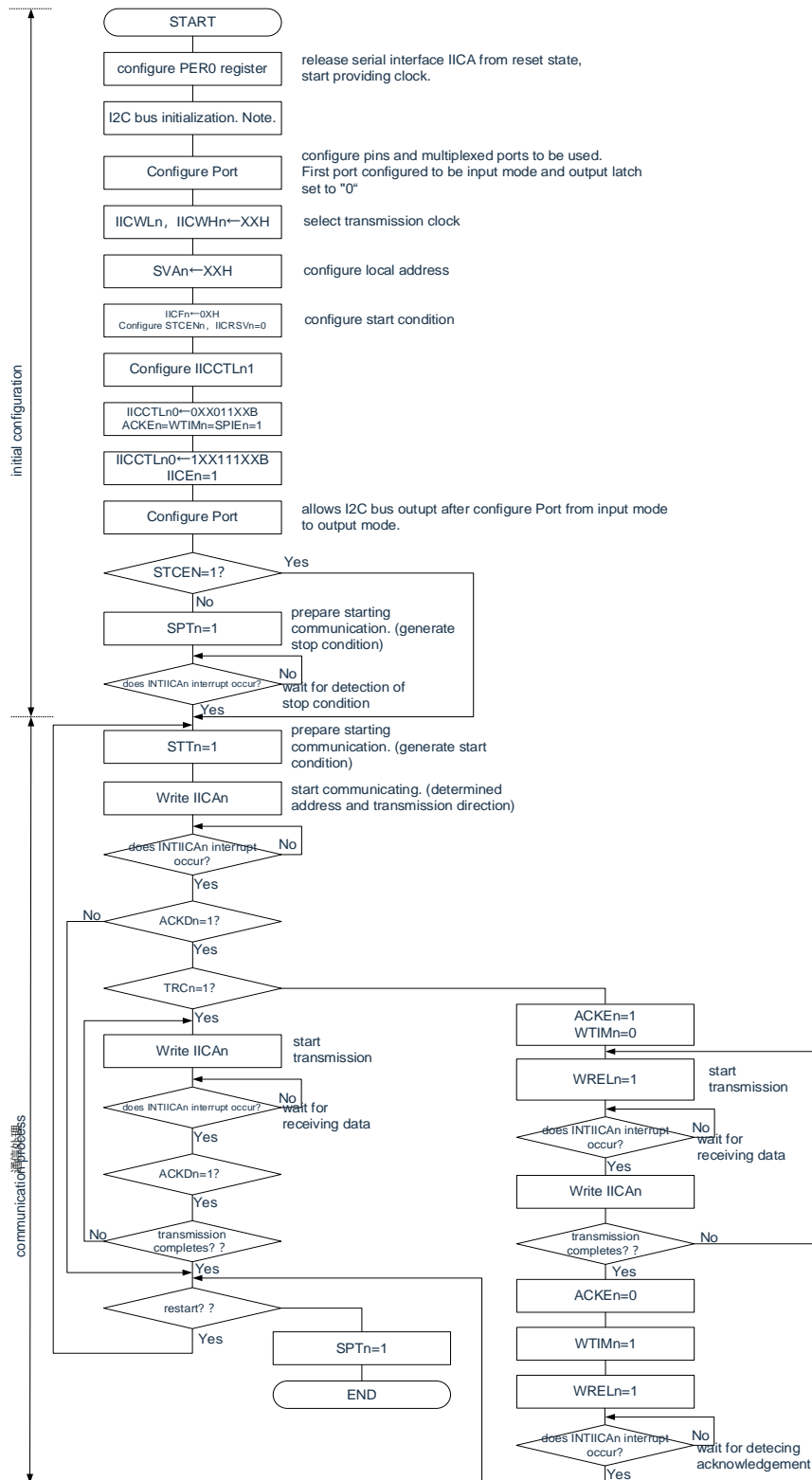
When used as the slave, operation is started by an interrupt. Execute the initial settings at startup, then wait for the INTIICAn interrupt occurrence (communication waiting). When an INTIICAn interrupt occurs, the communication status is judged and its result is passed as a flag over to the main processing.

By checking the flags, necessary communication processing is performed.

Remark n=0

(1) Master operation of single-master system

Figure 16-27 Master operation of single-master system



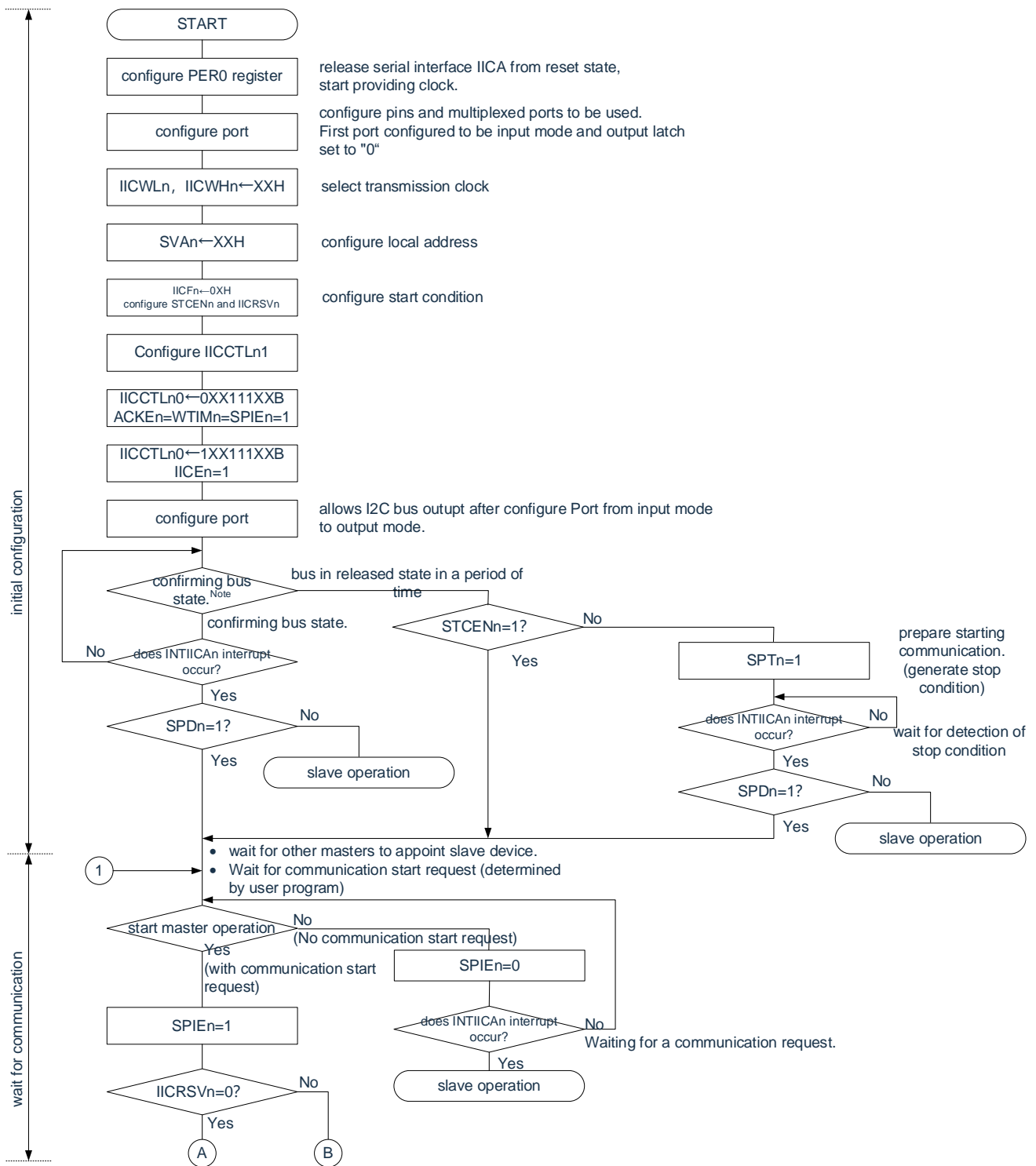
Note I²C-bus must be released (SCLAn pins and SDAAn pins are high) depending on the specifications of the product in communication. For example, if the EEPROM is in a state that outputs a low level to the SDAAn pin, the SCLAn pin must be set to the output port and a clock pulse must be output from the output port before the SDAAn pin is fixed high.

Remark 1. Conform to the specifications of the product that is communicating, with respect to the transmission and reception formats.

2. n=0

(2) Master operation in multi-master system

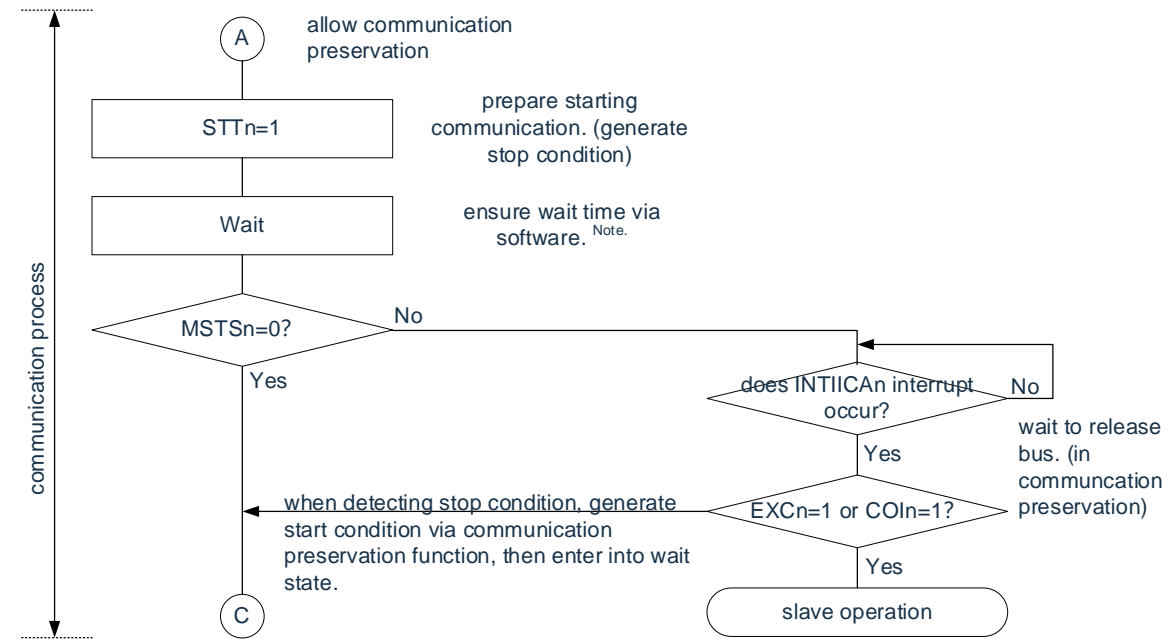
Figure 16-28 Master operation in multi-master system (1/3)



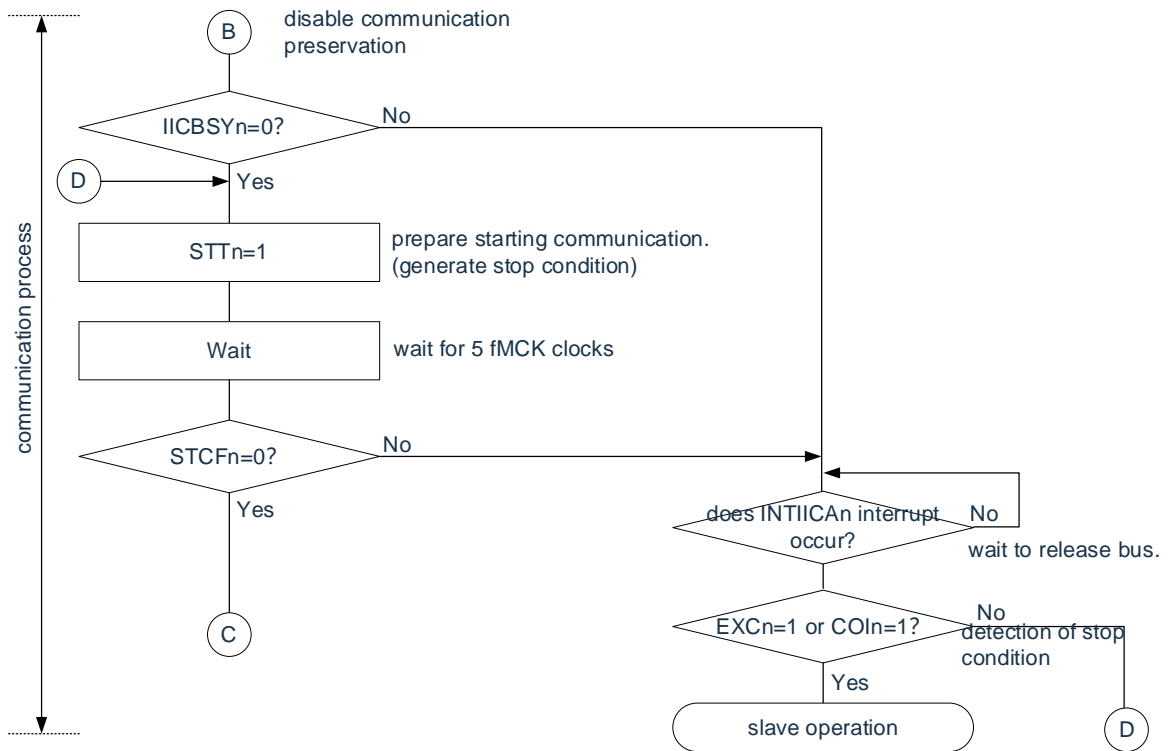
Note Confirm that the bus is released (CLDn bit = 1, DADn bit = 1) for a specific period (for example, for a period of one frame). If the SDAAn pin is constantly at low level, decide whether to release the I²C bus (SCLAn and SDAAn pins = high level) in conformance with the specifications of the product that is communicating.

Remark n=0

Figure 16-28 Master operation in multi-master system (2/3)



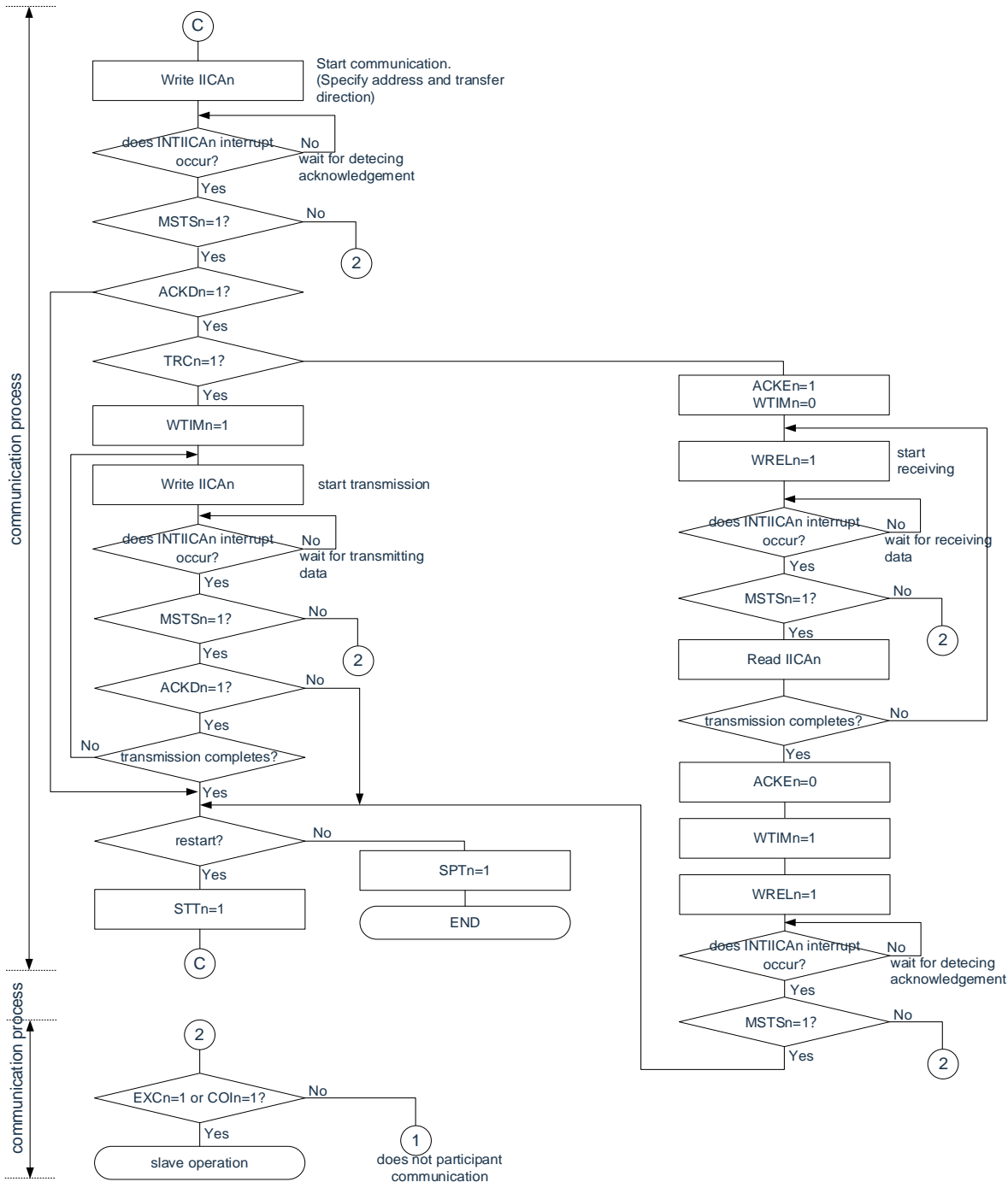
Note wait time as following:
 $(IICWLn \text{ configured value} + IICWHn \text{ configured value} + 4) / fMCK + tF \times 2$



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

Figure 16-28 Master operation in multi-master system (3/3)



Notice:

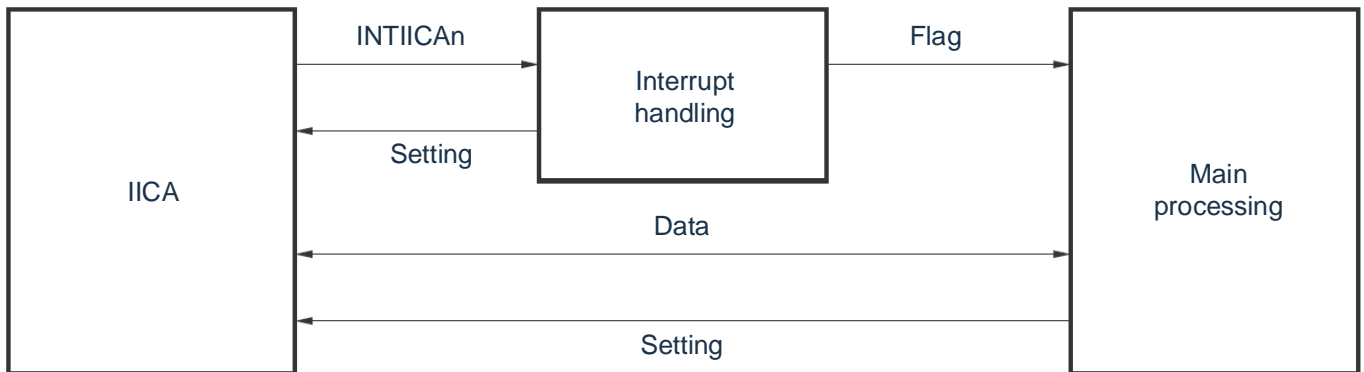
1. The format of transmission and reception must conform to the specifications of the product in communication.
2. When used as a master device in a multi-master system, the MSTSn bit must be read each time an INTIICAn interrupt occurs to acknowledge the arbitration result.
3. When used as a slave device 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.
4. n=0

(3) Slave operation

The processing procedure of the slave operation is as follows.

Basically, the slave operation is event-driven. Therefore, processing by the INTIICAn interrupt (processing that must substantially change the operation status such as detection of a stop condition during communication) is necessary.

In the following explanation, it is assumed that the extension code is not supported for data communication. It is also assumed that the INTIICAn interrupt servicing only performs status transition processing, and that actual data communication is performed by the main processing.



Therefore, 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.

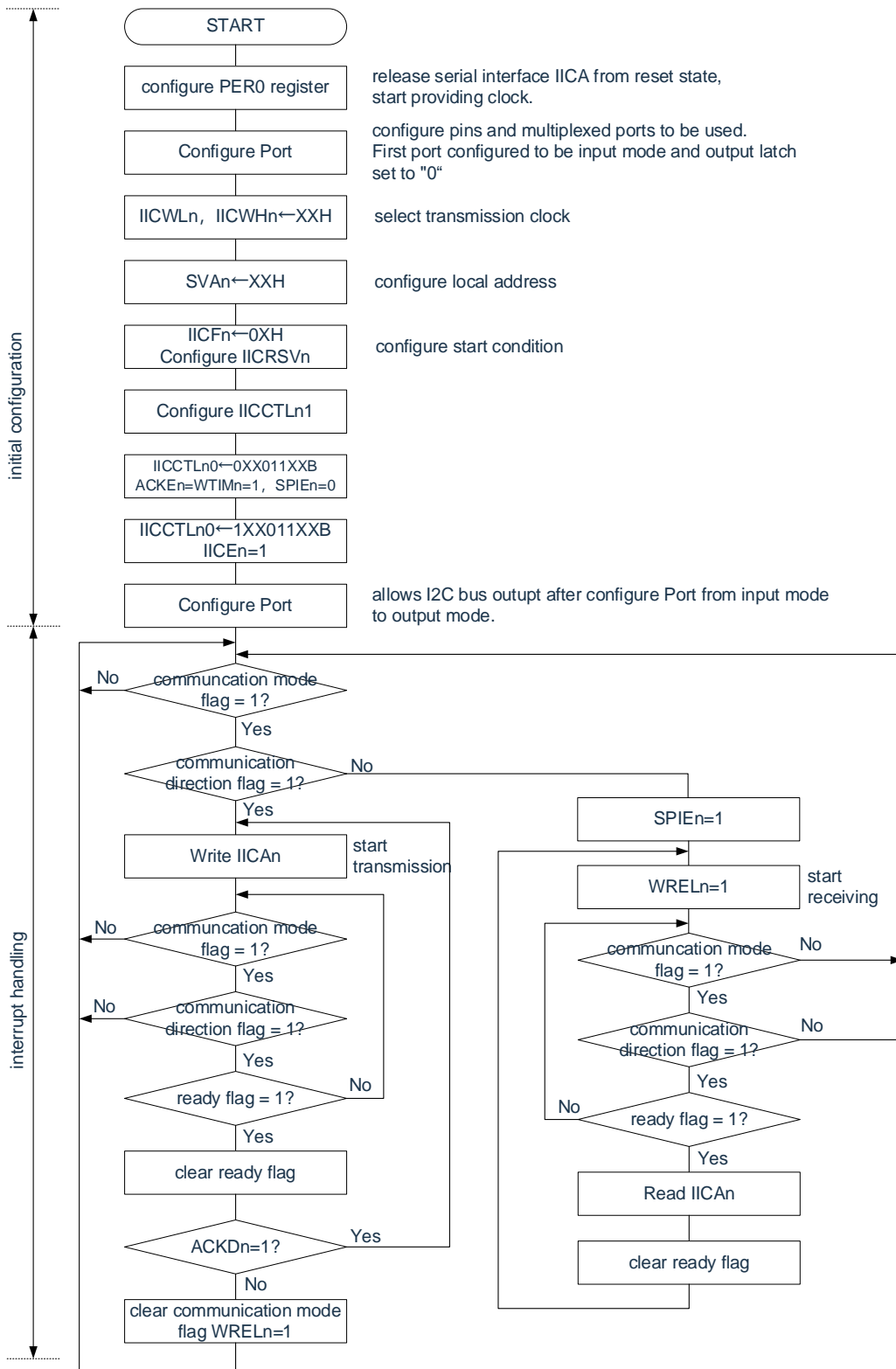
Remark n=0

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 16-29 Slave operation flowchart (1)



Remark 1. The format of transmission and reception must conform to the specifications of the product in communication.

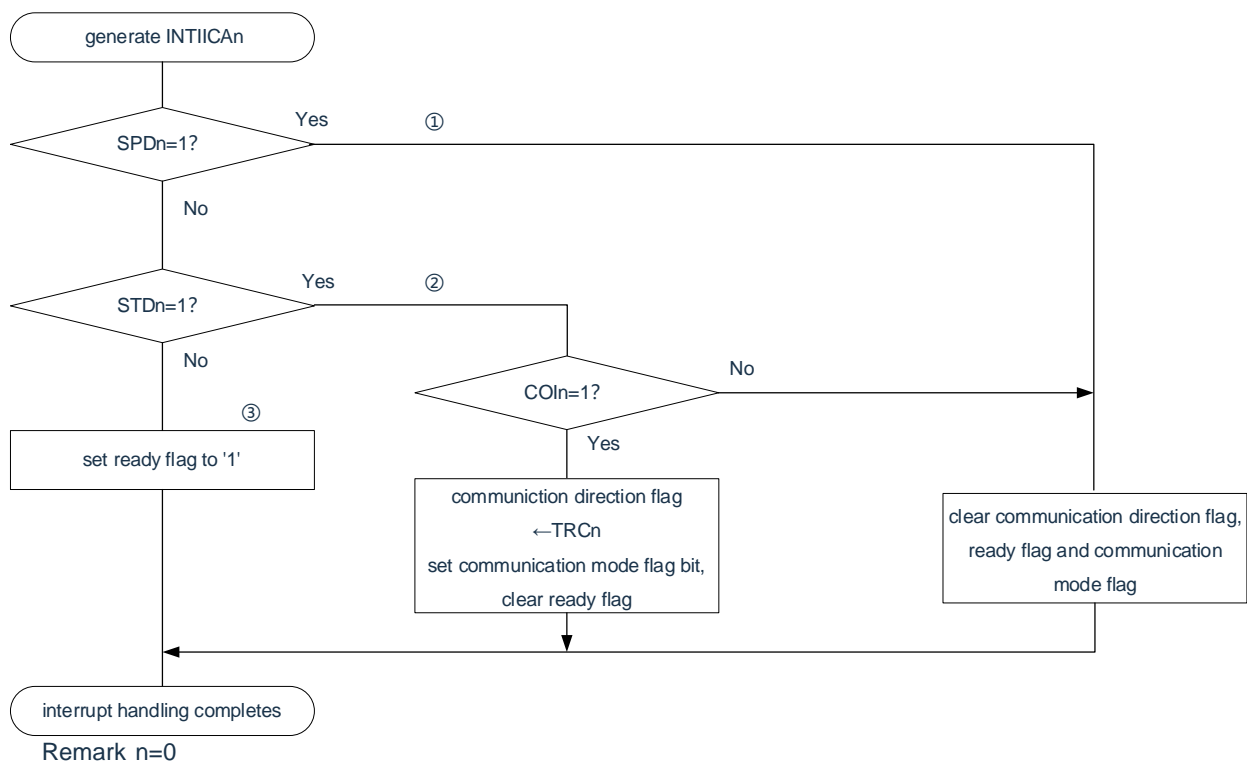
2. n=0

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 I²C bus remaining in the wait state.

Remark:① to ③ above correspond to ① to ③ in “Figure 16-30 Slave operation flowchart (2)”.

Figure 16-30 Slave operation flowchart (2)



16.5.17 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.

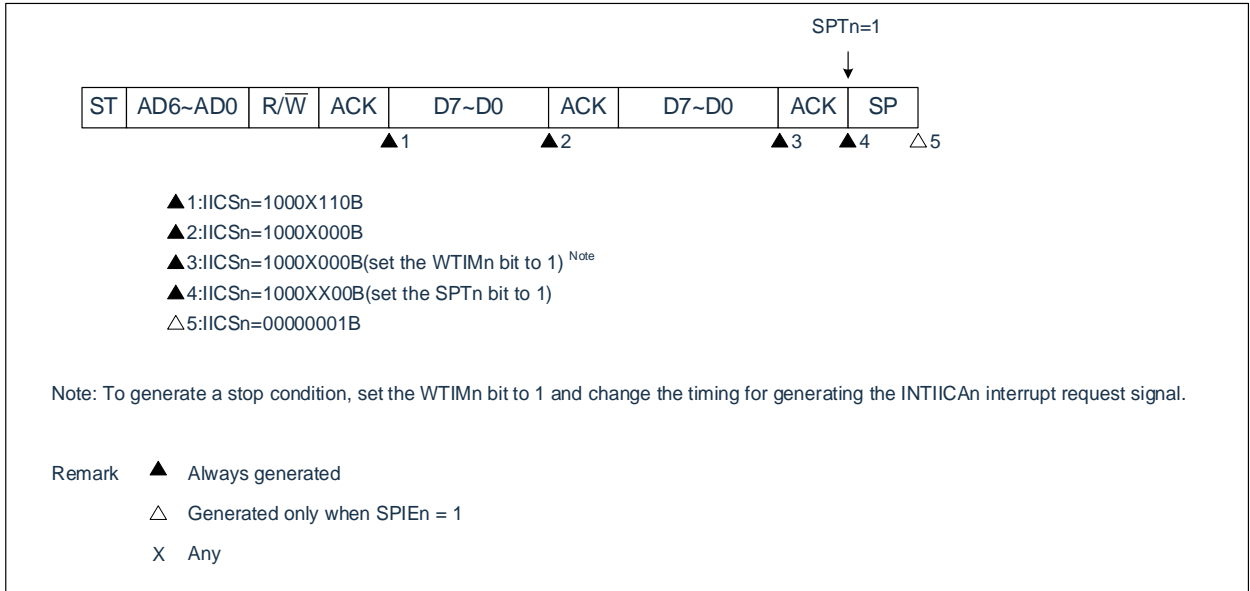
Remark:

1. ST: Start condition
AD6~AD0: Address
R/W: Transfer direction specification
ACK: Acknowledge
D7~D0: Data
SP: Stop condition
2. n=0

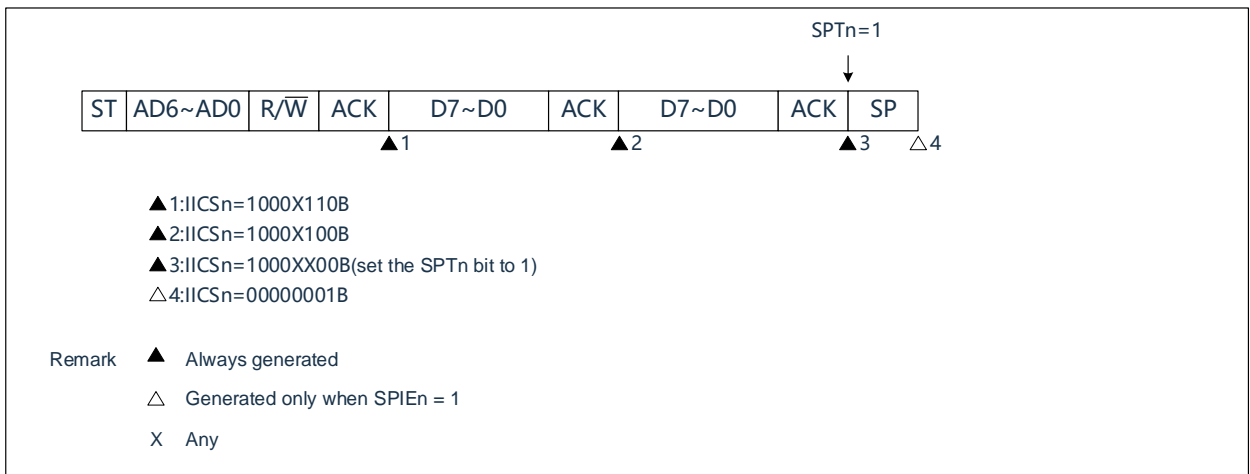
(1) Master operation

(a) Start~Address~Data~Data~Stop (transmit and receive)

(i) When WTIMn=0



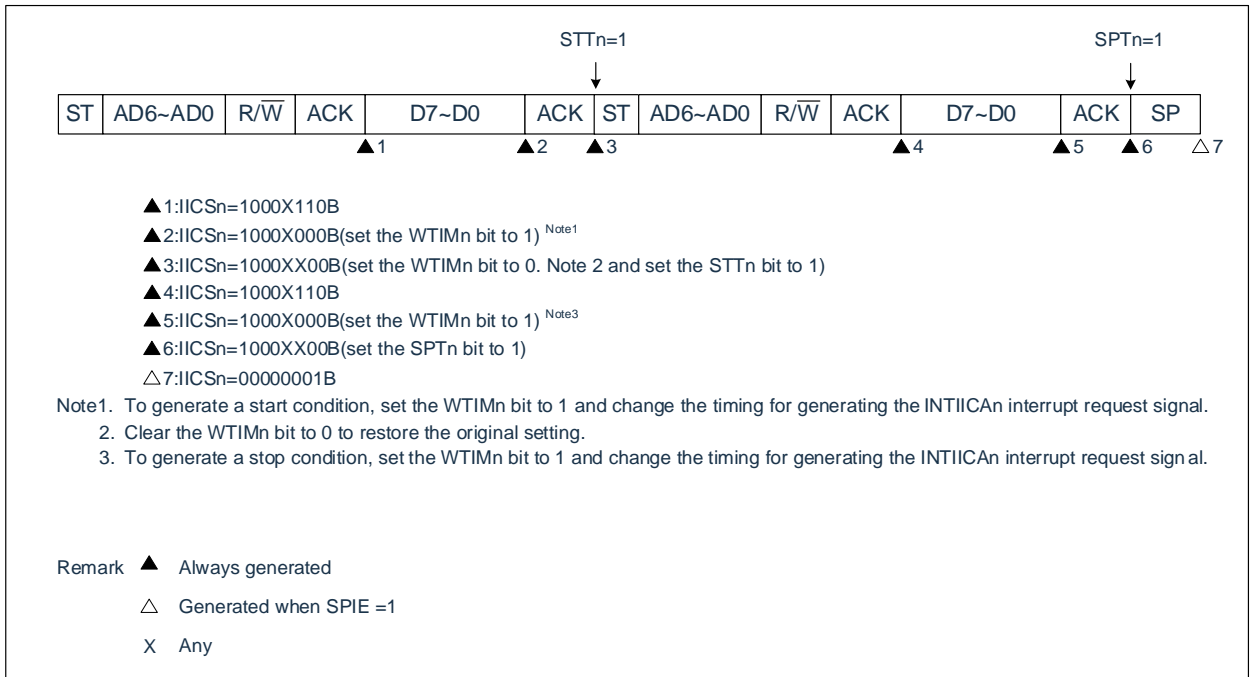
(ii) When WTIMn=1



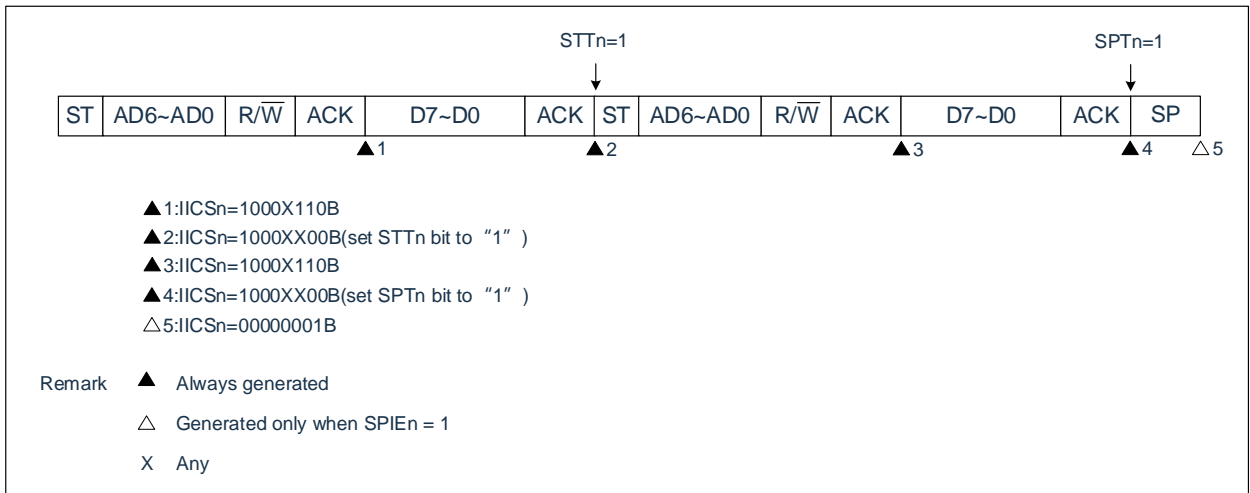
Remark n=0

(b) Start~Address~Data~Start~Address~Data~Stop (restart)

(i) When WTIMn=0



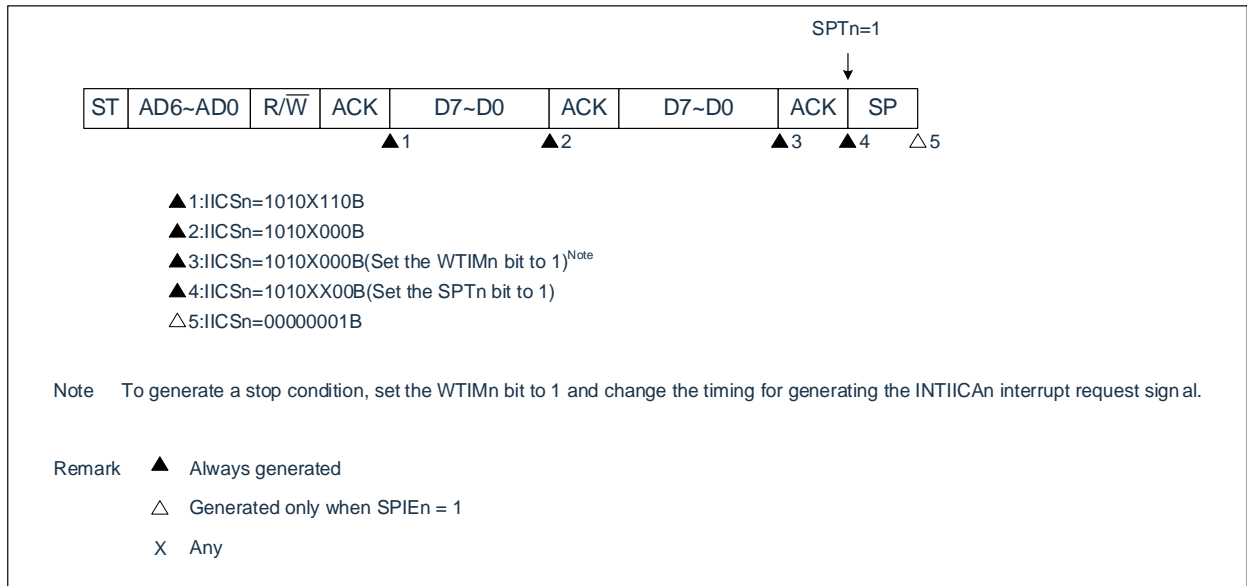
(ii) When WTIMn=1



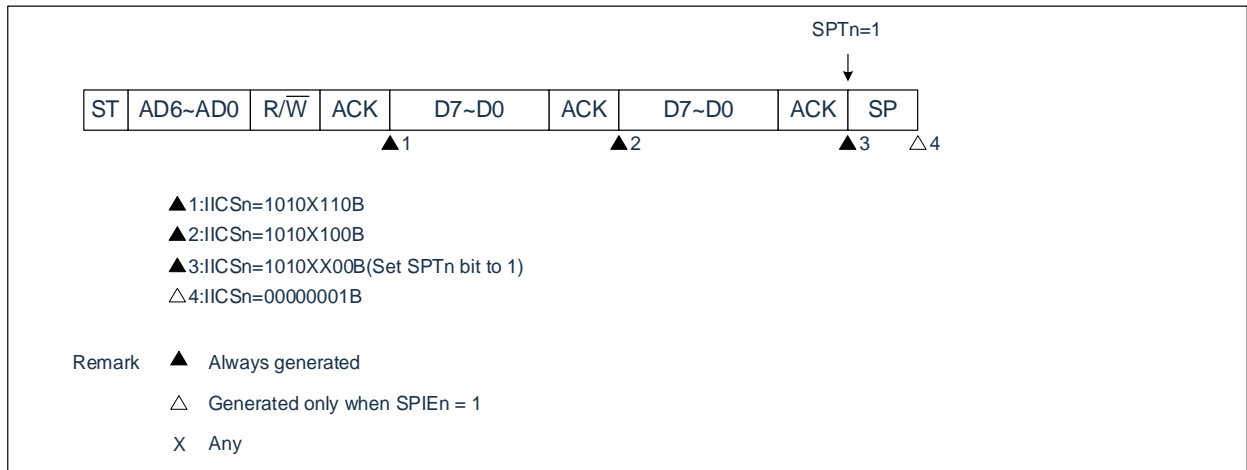
Remark n=0

(c) Start~Code~Data~Data~Stop (extension code transmission)

(i) When WTIMn=0



(ii) When WTIMn=1

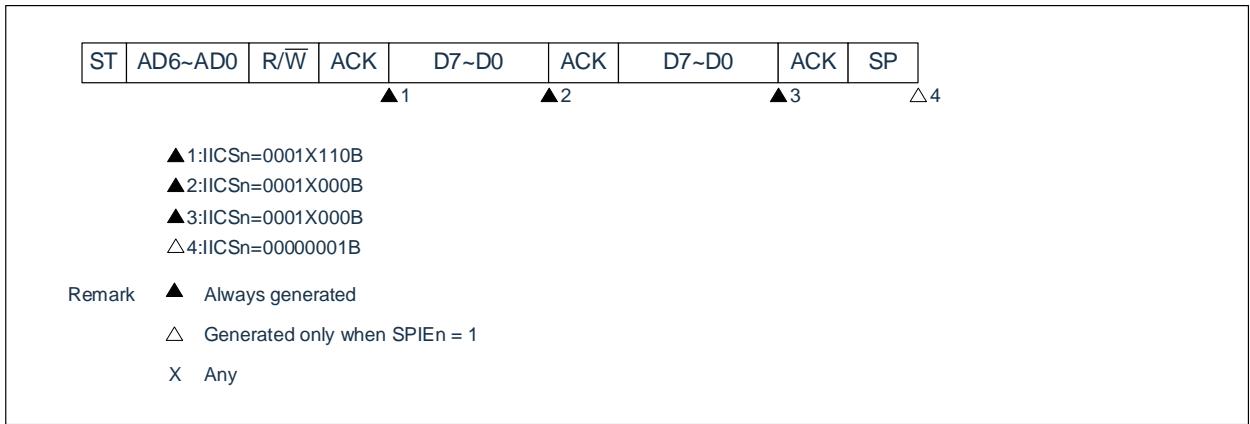


Remark n=0

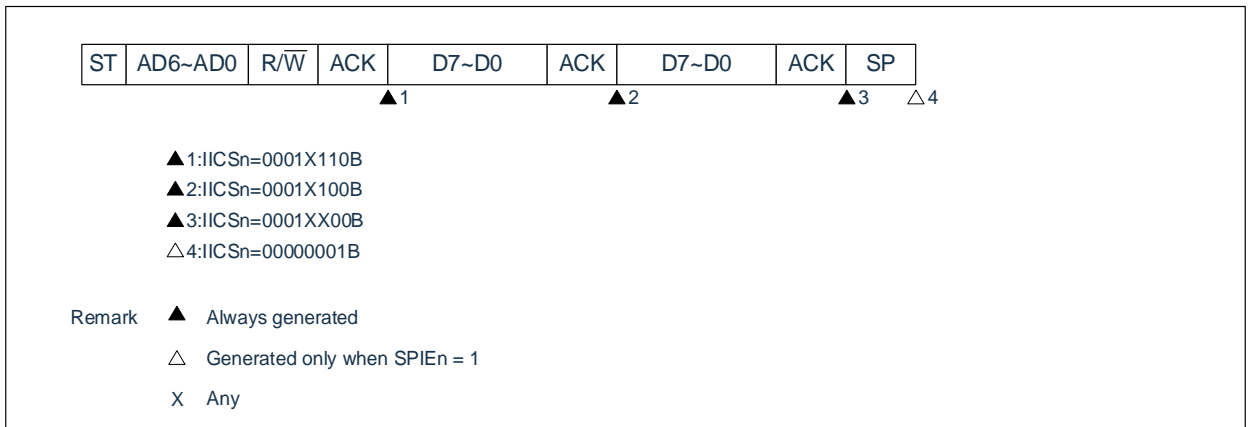
(2) Slave operation (when receiving a slave address)

(a) Start~Address~Data~Data~Stop

(i) When WTIMn=0



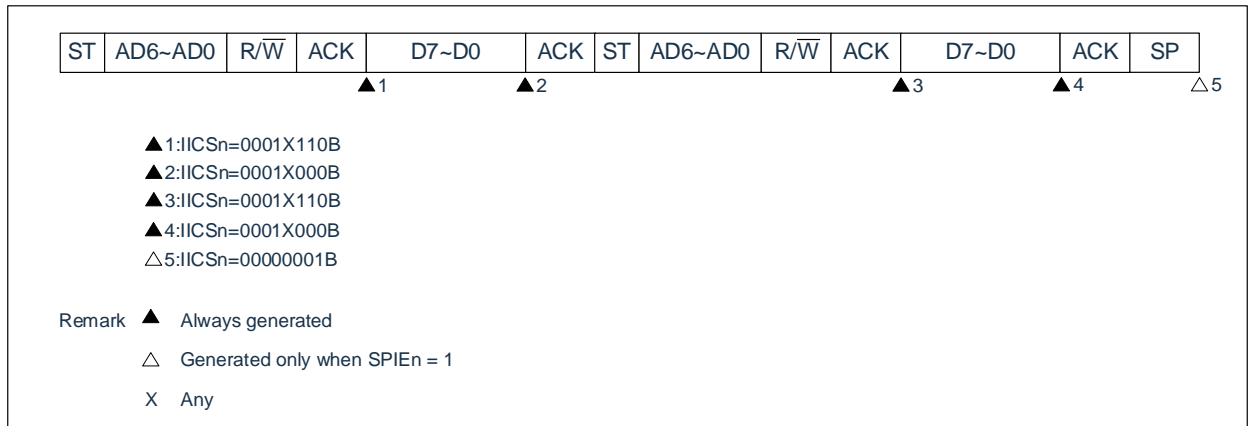
(ii) When WTIMn=1



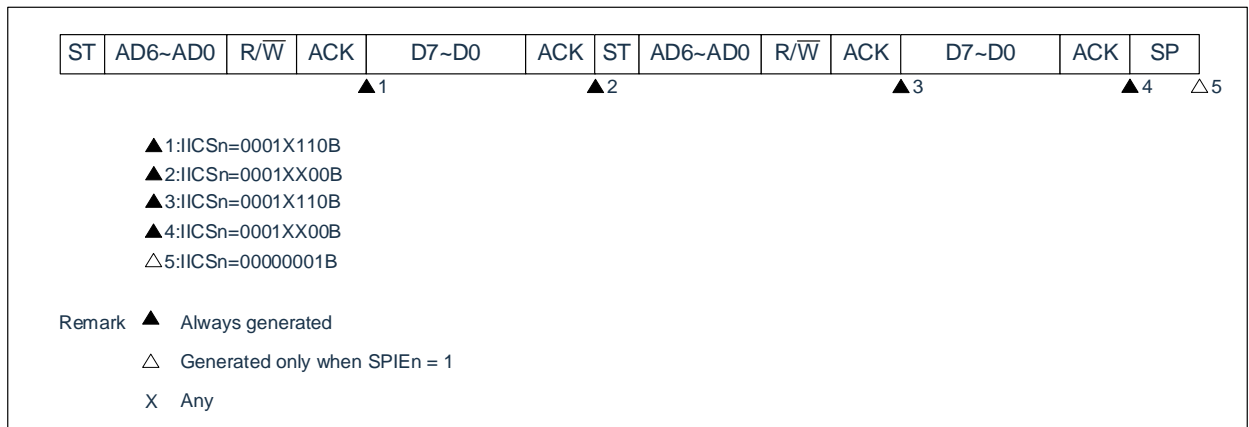
Remark n=0

(b) Start~Address~Data~Start~Address~Data~Stop

(i) When WTIMn=0 (after restart, matches with SVAn)



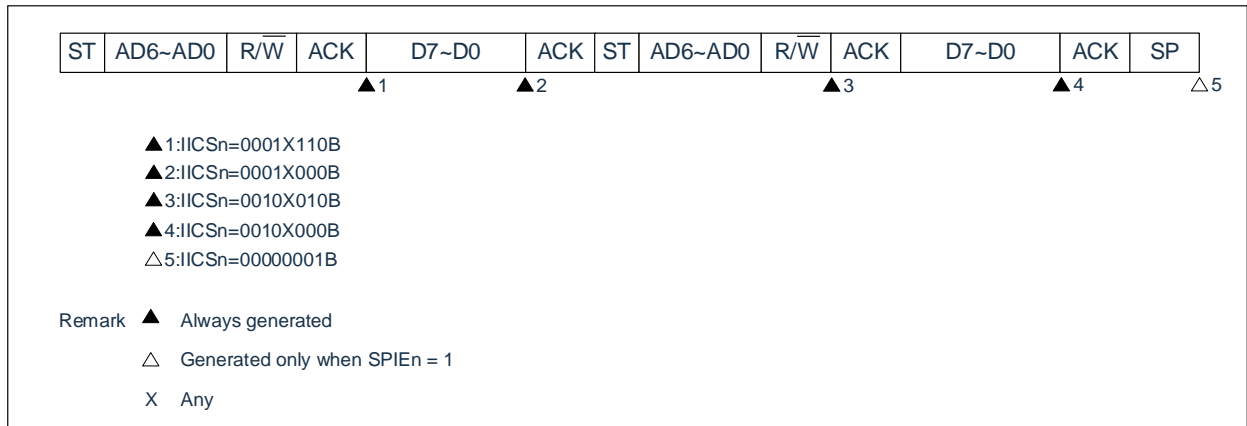
(ii) When WTIMn=1 (after restart, matches with SVAn)



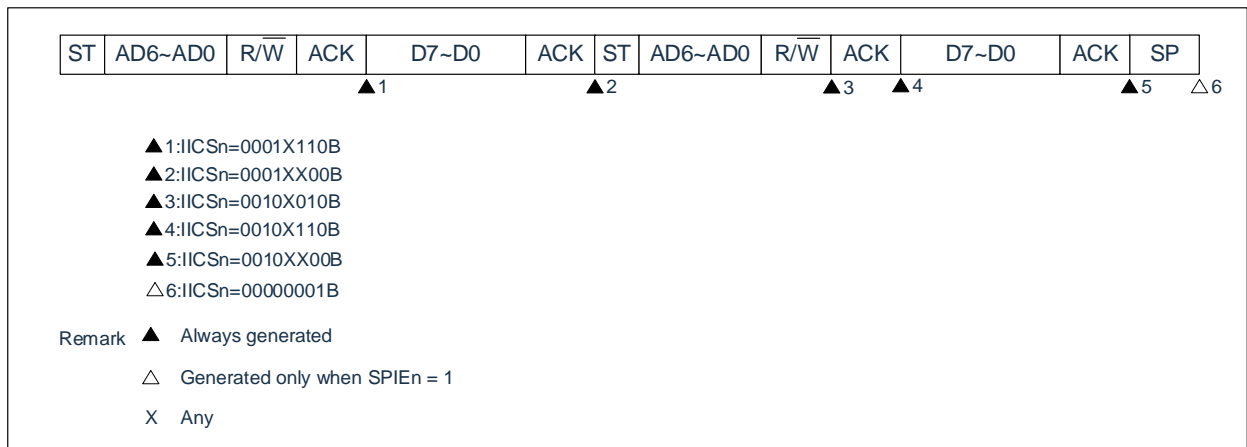
Remark n=0

(c) Start~Address~Data~Start~Code~Data~Stop

(i) When WTIMn=0 (after restart, does not match address (= extension code))



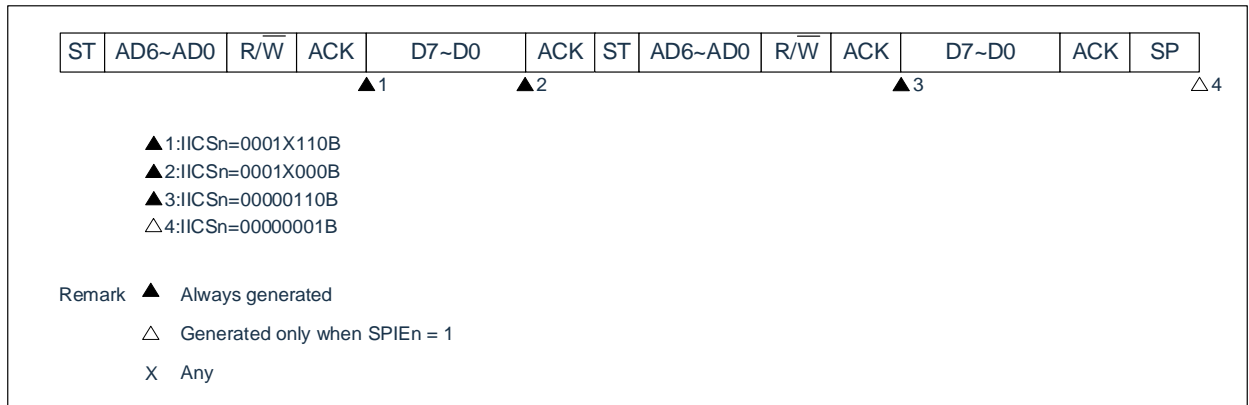
(ii) When WTIMn=1 (after restart, does not match address (= extension code))



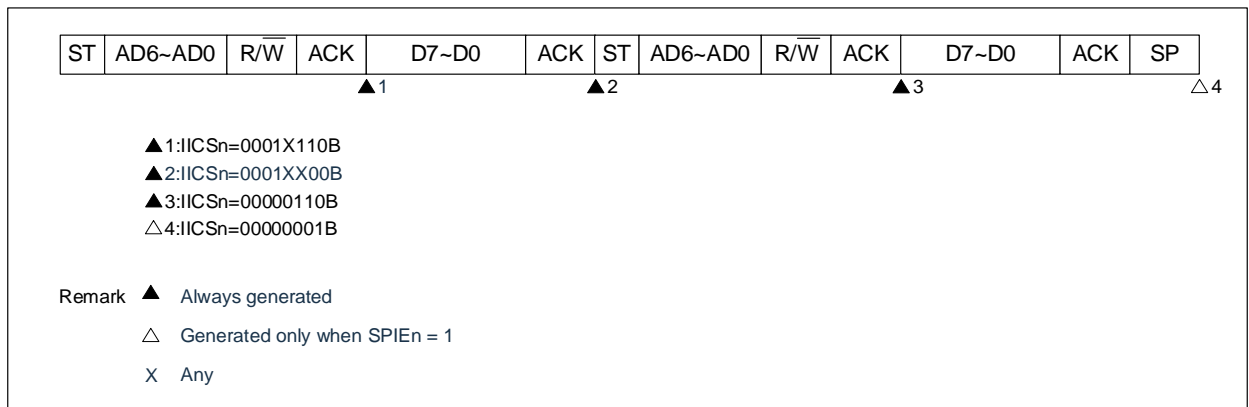
Remark n=0

(d) Start~Address~Data~Start~Address~Data~Stop

(i) When WTIMn=0 (after restart, does not match address (= not extension code))



(ii) When WTIMn=1 (after restart, does not match address (= not extension code))



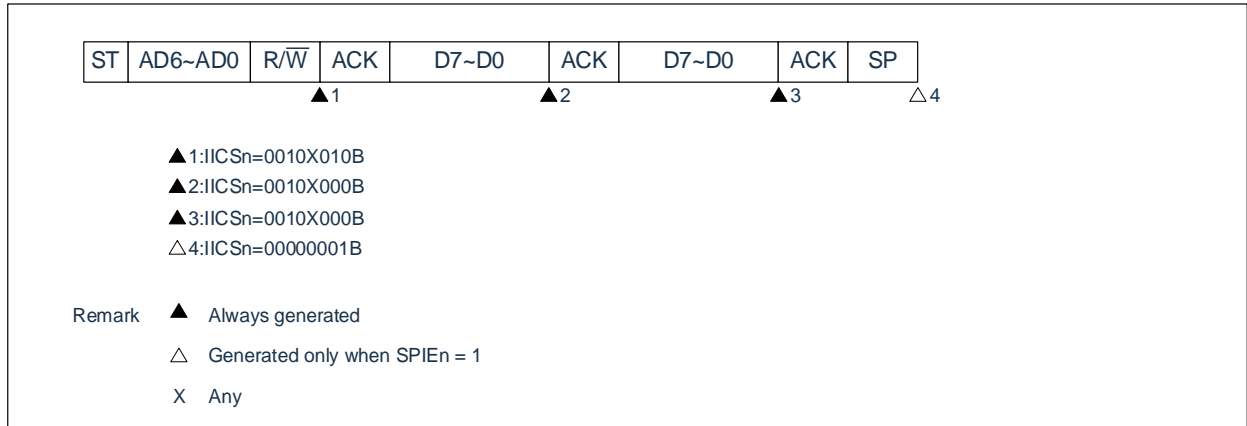
Remark n=0

(3) Slave operation (when receiving extension code)

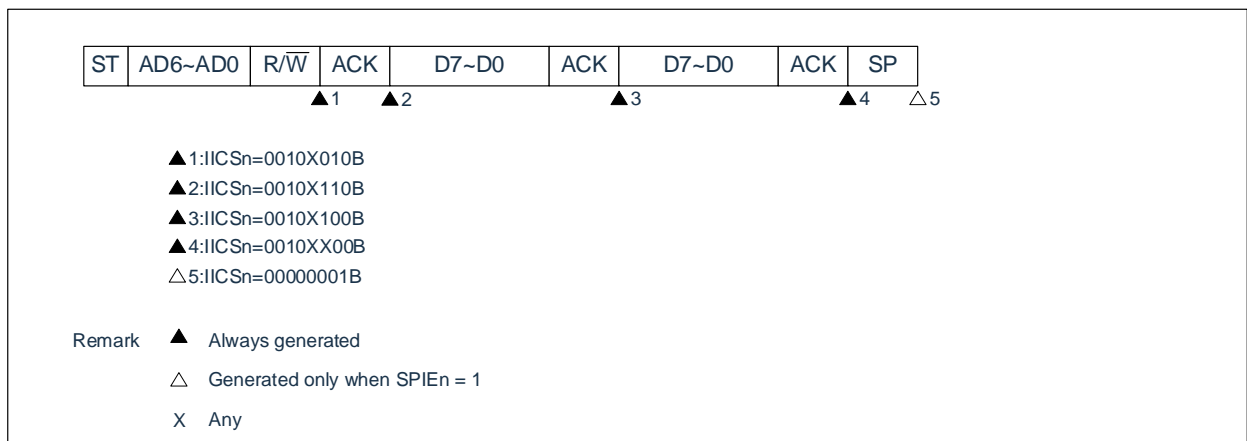
The device is always participating in communication when it receives an extension code.

(a) Start~Code~Data~Data~Stop

(i) When WTIMn=0



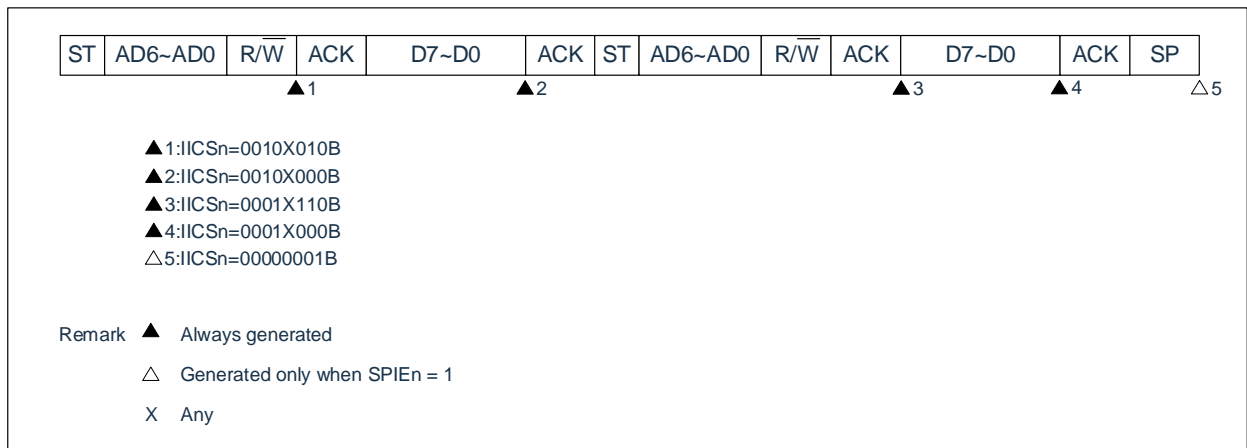
(ii) When WTIMn=1



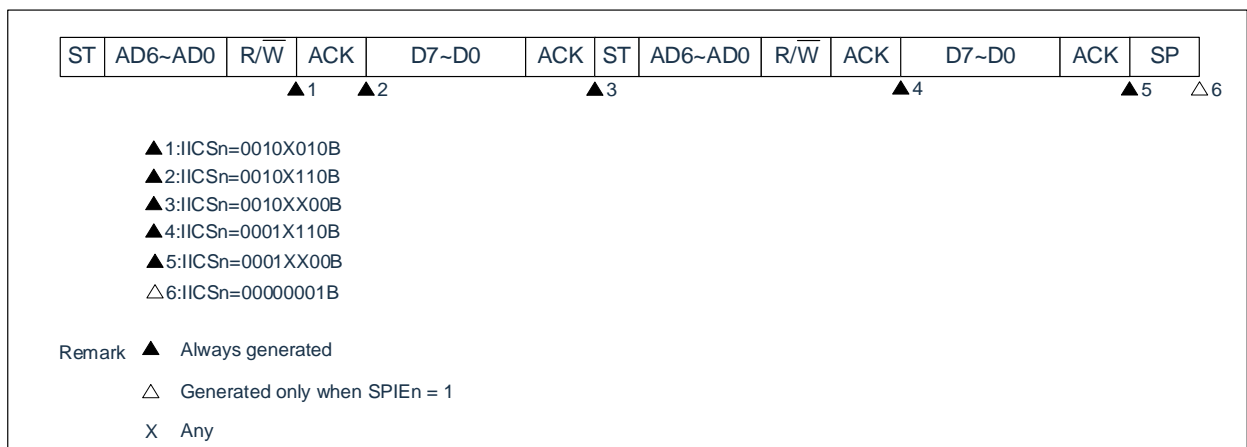
Remark n=0

(b) Start~Code~Data~Start~Address~Data~Stop

(i) When WTIMn=0 (after restart, matches SVAn)



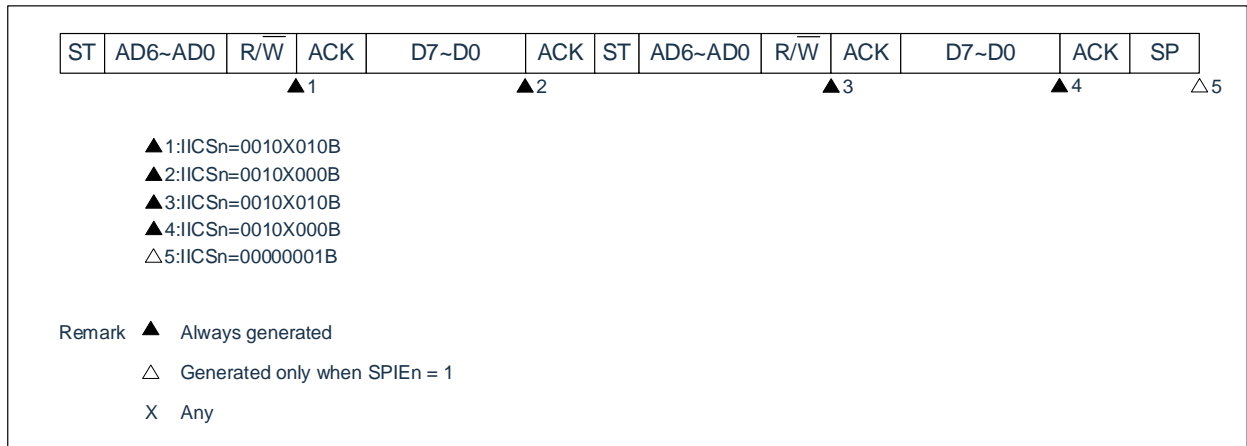
(ii) When WTIMn=1 (after restart, matches SVAn)



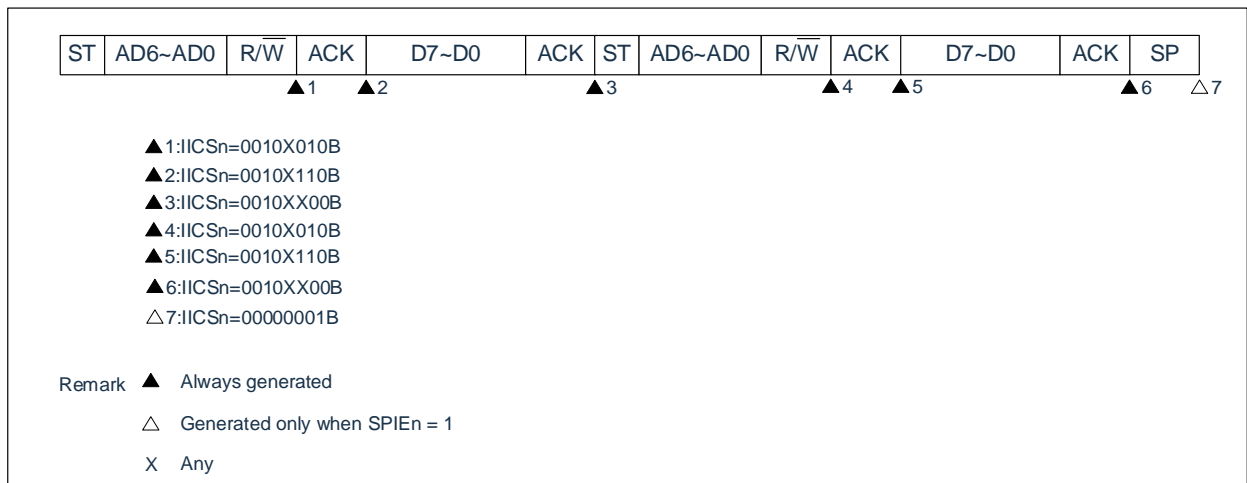
Remark n=0

(c) Start~Code~Data~Start~Code~Data~Stop

(i) When WTIMn=0 (after restart, extension code reception)



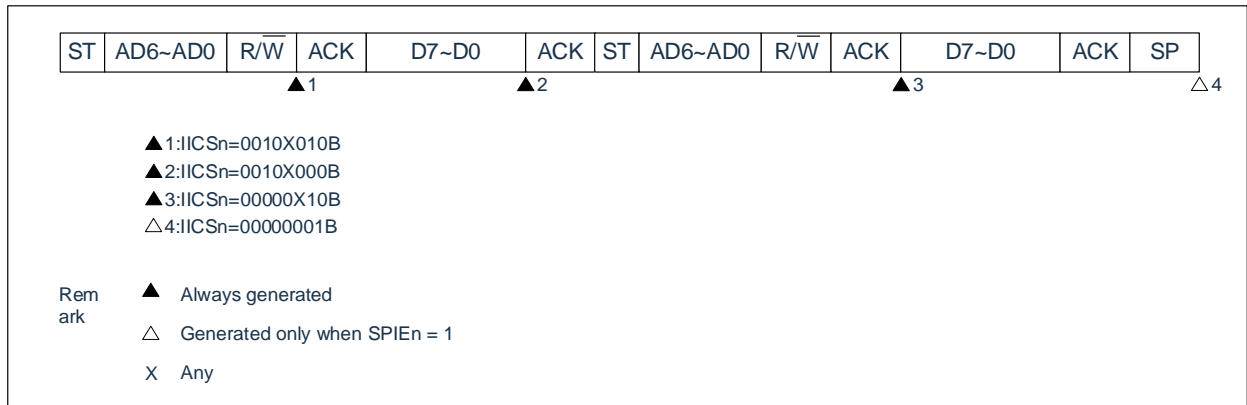
(ii) When WTIMn=1 (after restart, extension code reception)



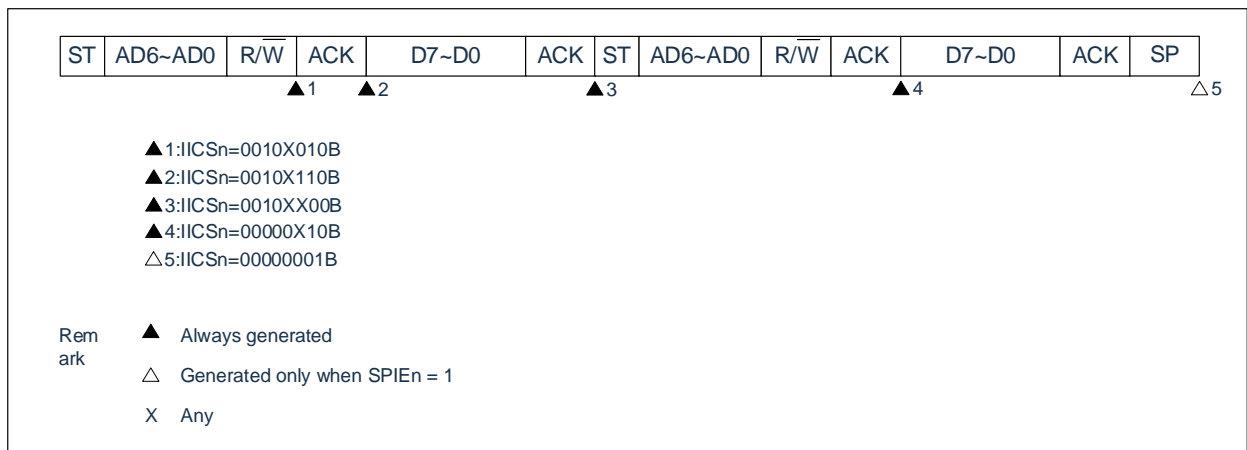
Remark n=0

(d) Start~Code~Data~Start~Address~Data~Stop

(i) When WTIMn=0 (after restart, does not match address (= not extension code))

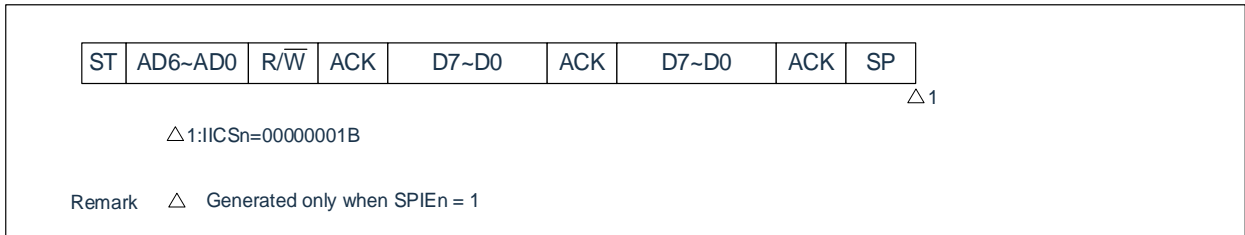


(ii) When WTIMn=1 (after restart, does not match address (= not extension code))



Remark n=0

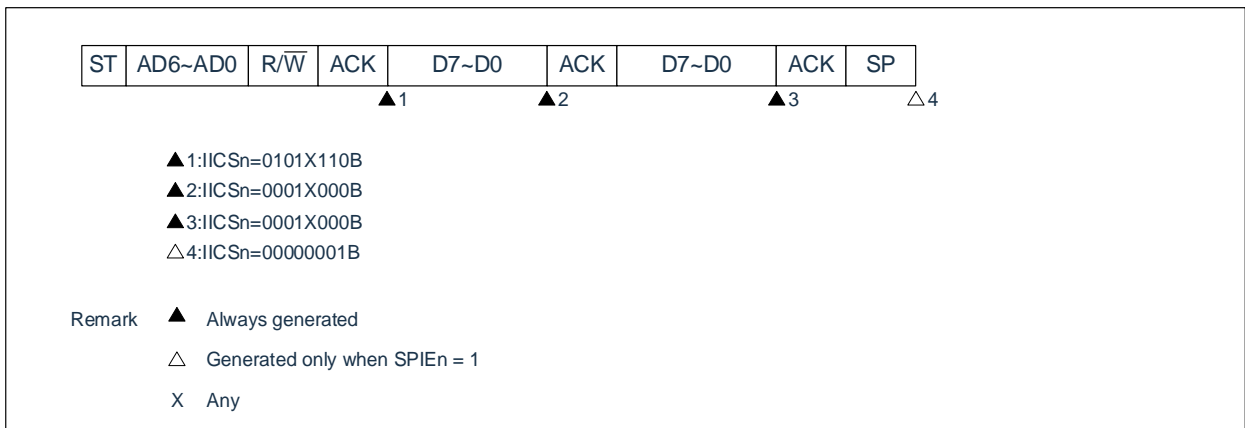
- (4) Operation without communication
 - (a) Start~Code~Data~Data~Stop



- (5) Arbitration loss operation (operation as slave after arbitration loss)

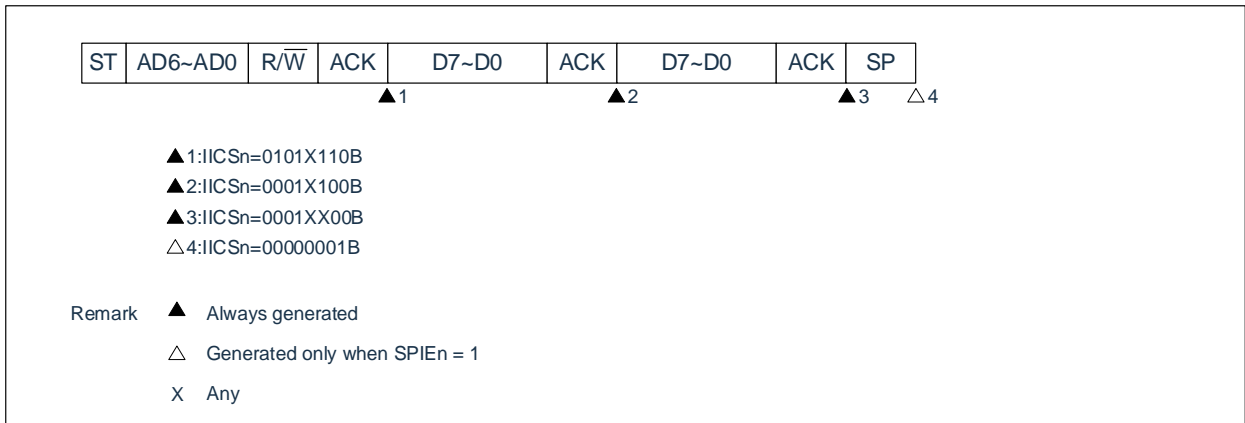
When the device is used as a master in a multi-master system, read the MSTSn bit each time interrupt request signal INTIICAn has occurred to check the arbitration result.

- (a) When arbitration loss occurs during transmission of slave address data
 - (i) When WTIMn=0



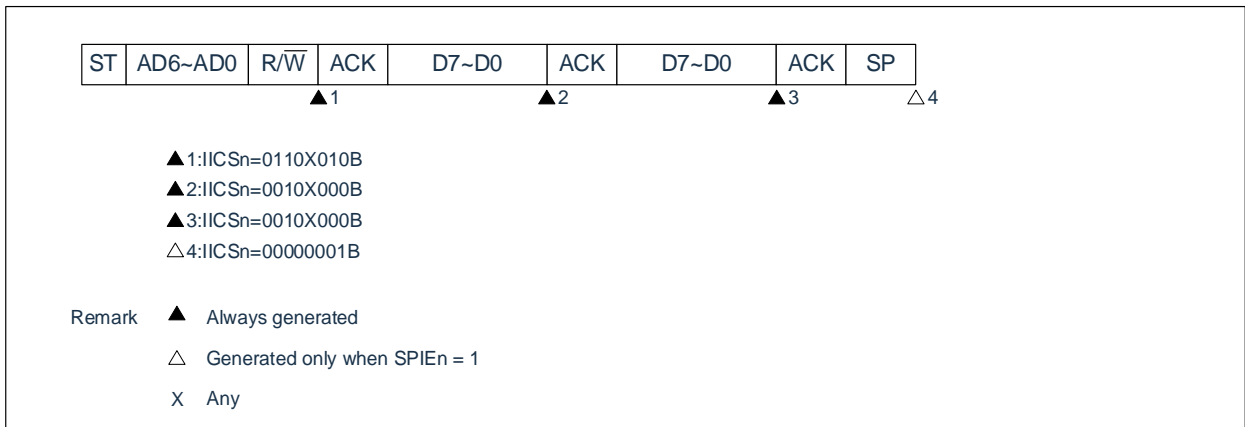
Remark n=0

(ii) When WTIMn=1



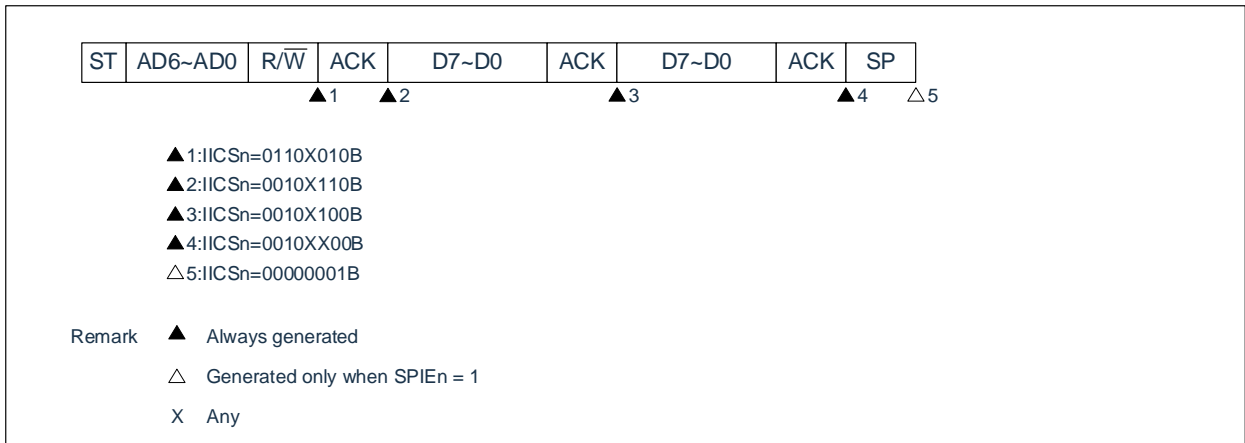
(b) When arbitration loss occurs during transmission of extension code

(i) When WTIMn=0



Remark n=0

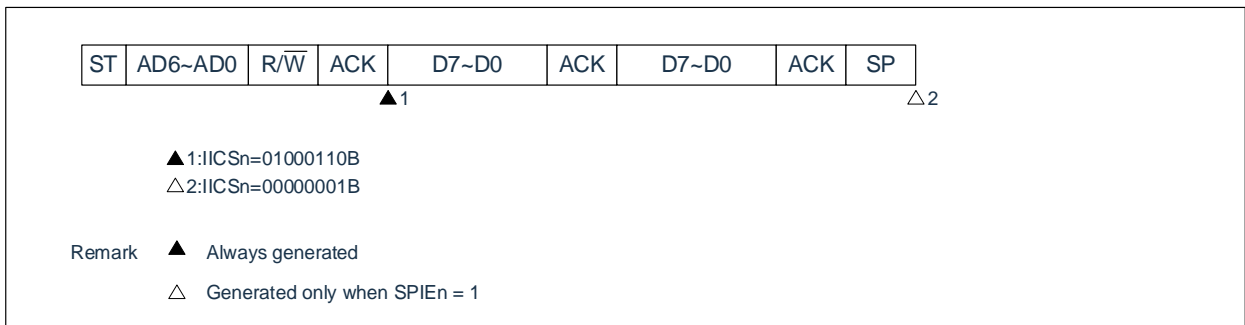
(ii) When WTIMn=1



(6) Operation when arbitration loss occurs (no communication after arbitration loss)

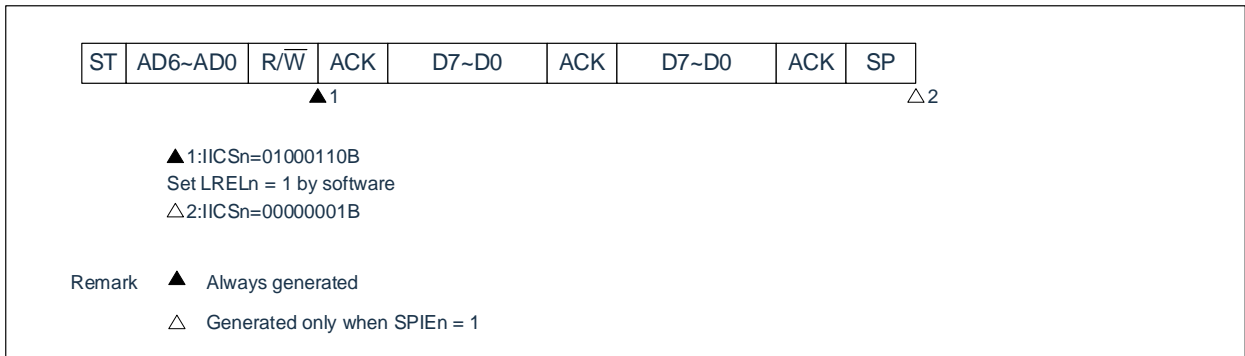
When the device is used as a master in a multi-master system, read the MSTSn bit each time interrupt request signal INTIICAn has occurred to check the arbitration result.

(a) When arbitration loss occurs during transmission of slave address data (when WTIMn = 1)



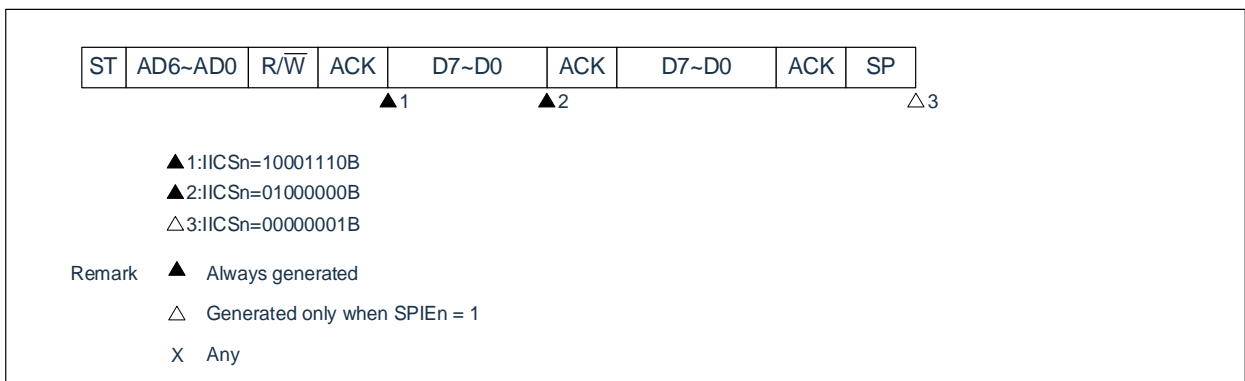
Remark n=0

(b) When arbitration loss occurs during transmission of extension code



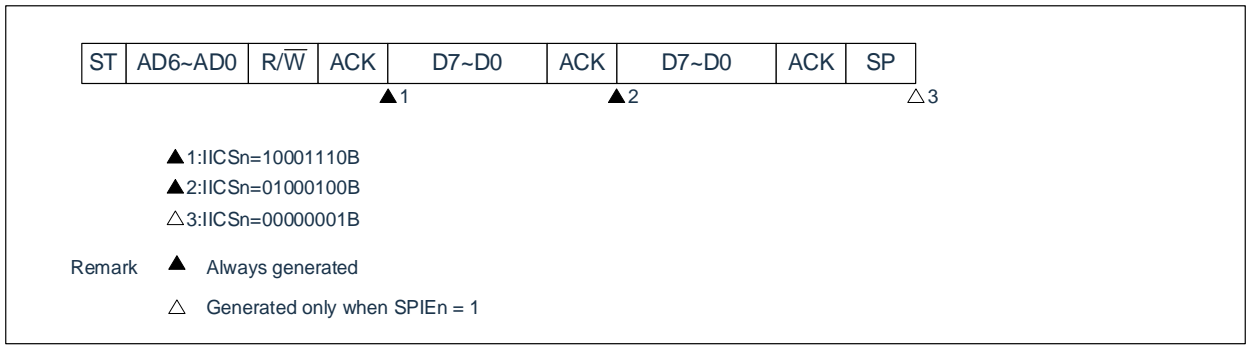
(c) When arbitration loss occurs during transmission of data

(i) When WTIMn=0



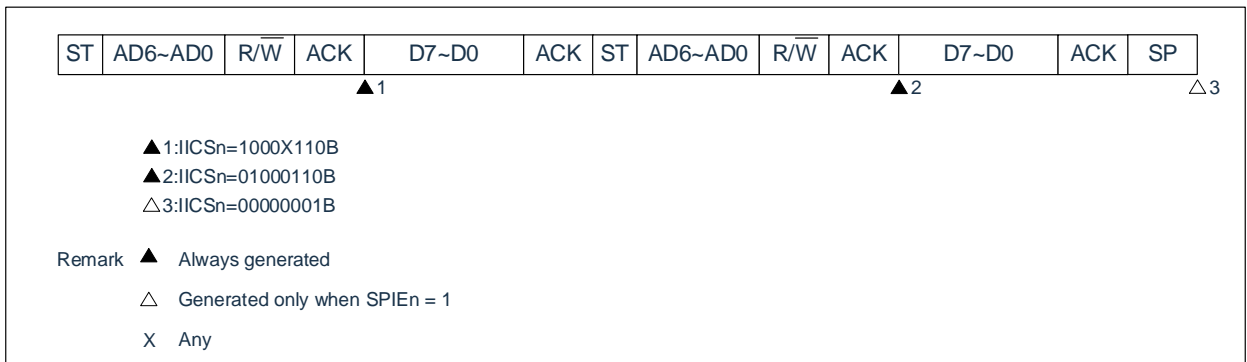
Remark n=0

(ii) When WTIMn=1



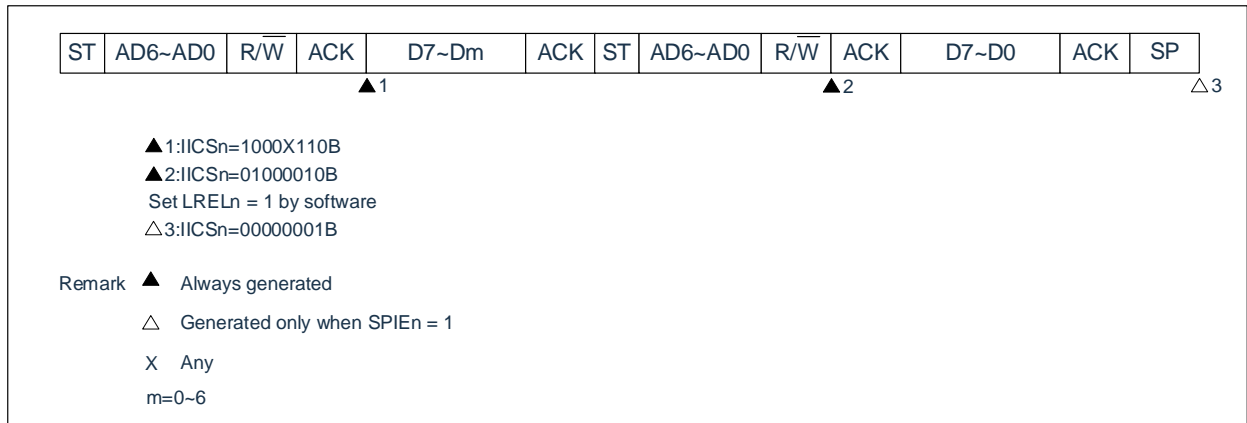
(d) When loss occurs due to restart condition during data transfer

(i) Not extension code (Example: unmatched with SVAn)

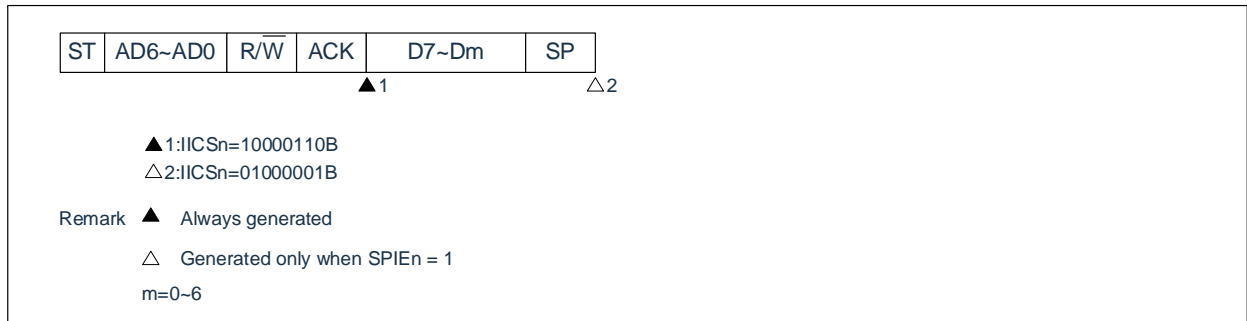


Remark n=0

(ii) Extension code



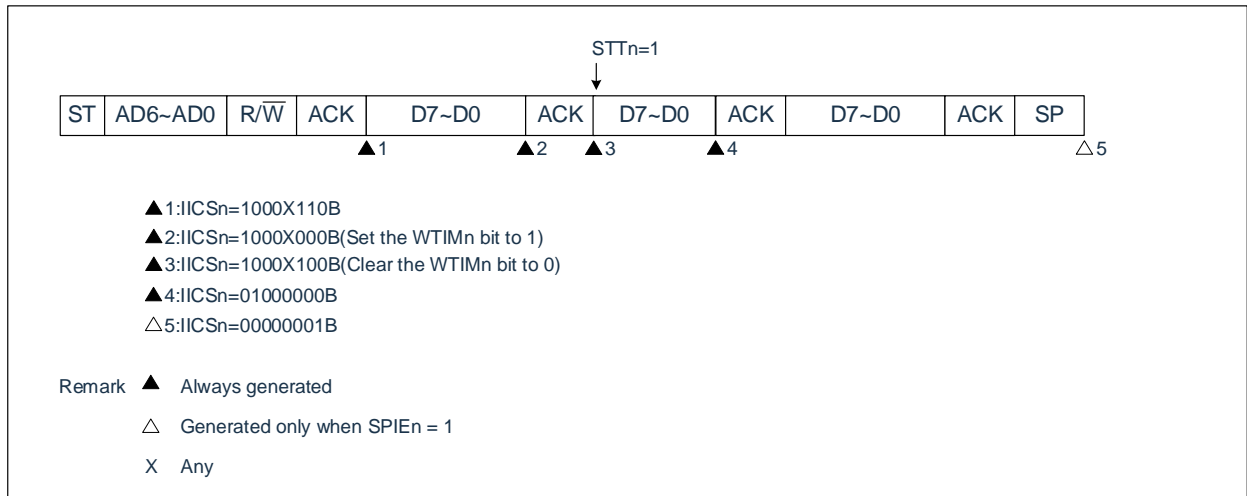
(e) When loss occurs due to stop condition during data transfer



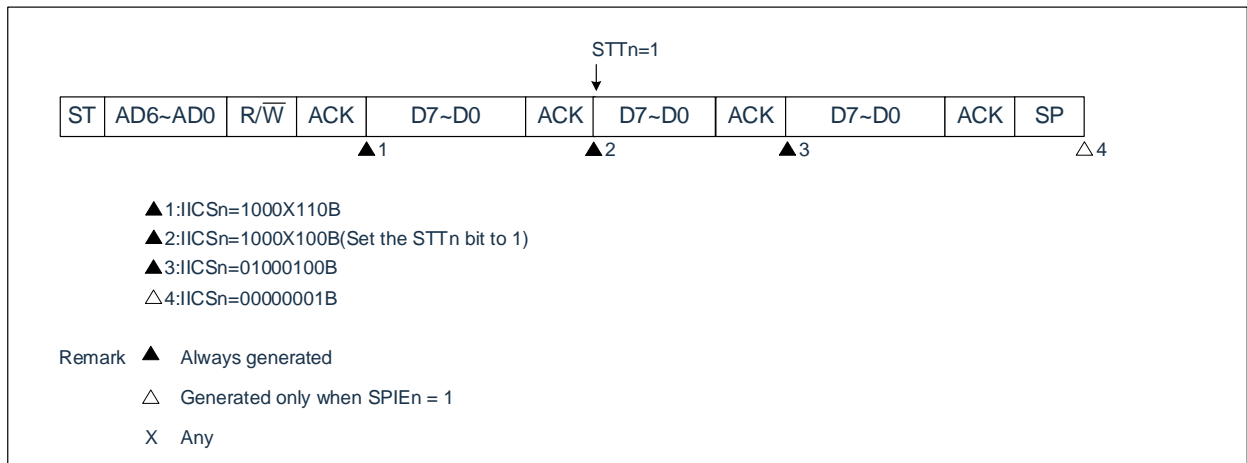
Remark n=0

(f) When arbitration loss occurs due to low-level data when attempting to generate a restart condition

(i) When WTIMn=0

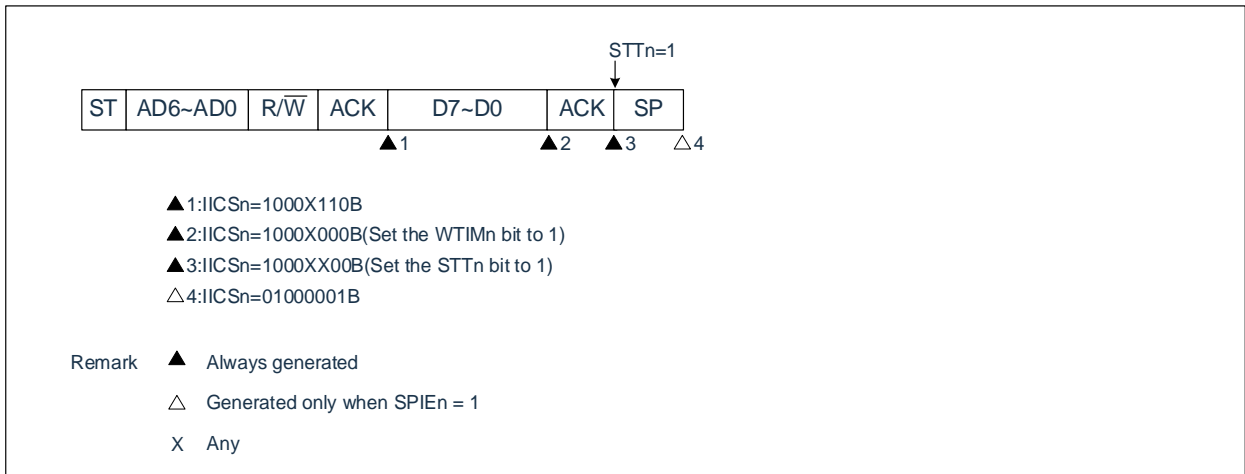


(ii) When WTIMn=1

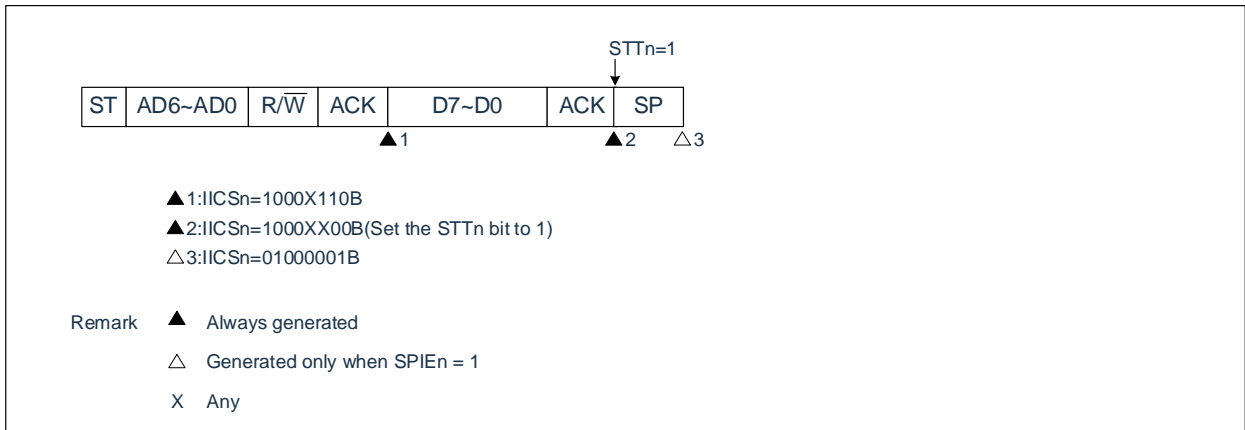


Remark n=0

- (g) When arbitration loss occurs due to a stop condition when attempting to generate a restart condition
 - (i) When WTIMn=0



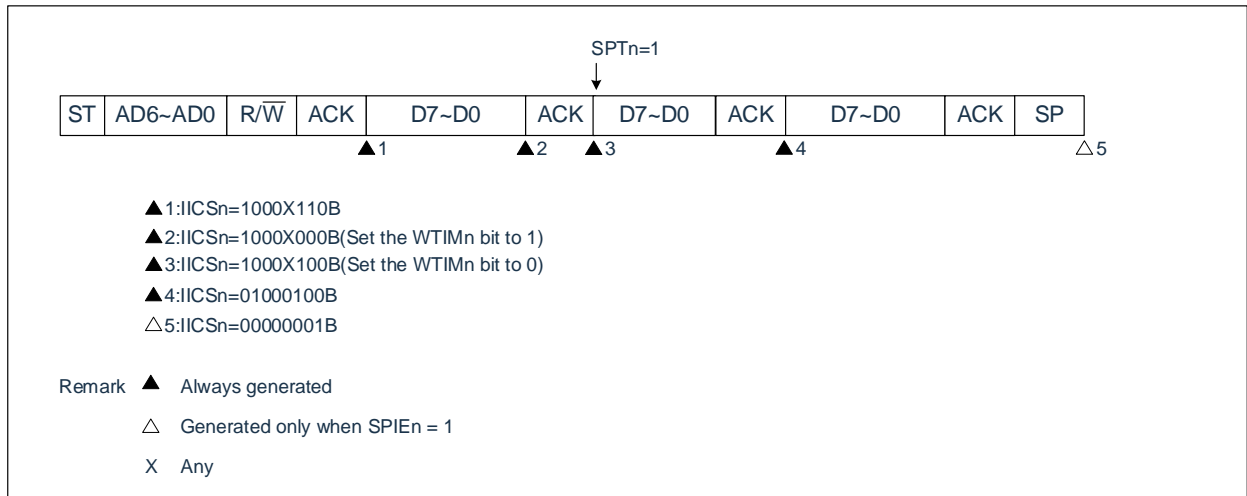
- (ii) When WTIMn=1



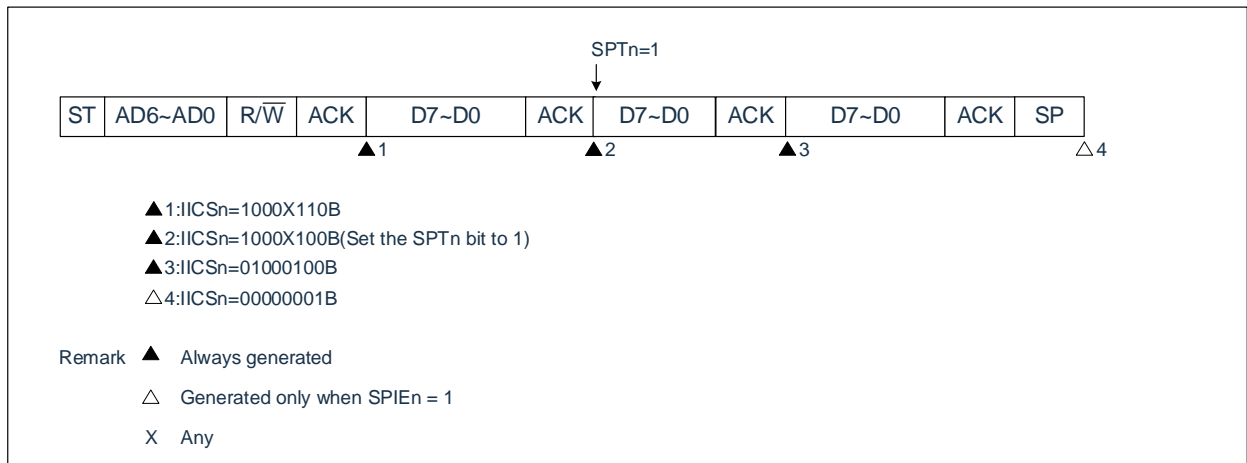
Remark n=0

(h) When arbitration loss occurs due to low-level data when attempting to generate a stop condition

(i) When WTIMn=0



(ii) When WTIMn=1



Remark n=0

16.6 Timing diagram

When using the I²C 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 16-31 and Figure 16-31.

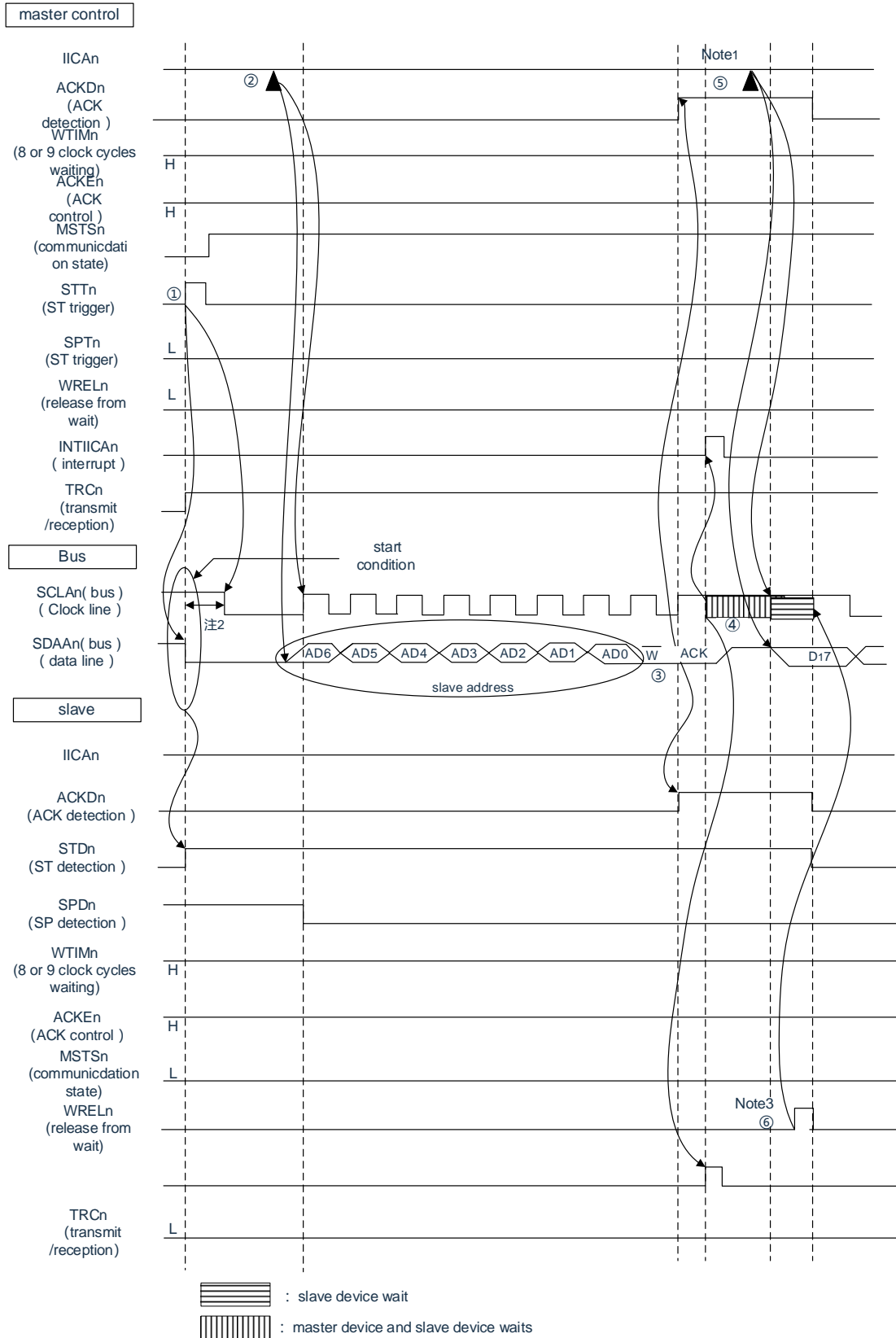
The IICA shift register n (IICAn)'s shift operation is synchronized with the falling edge of the serial clock (SCLAn). The transmit data is transferred to the SO latch and is output (MSB first) via the SDAAn pin.

Data input via the SDAAn pin is captured into IICAn at the rising edge of SCLAn.

Remark n=0

Figure 16-31 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.

- 2: Make sure that the time between the fall of the SDAAn pin signal and the fall of the SCLAn pin signal is at least 4.0 μs when specifying standard mode and at least 0.6 μs when specifying fast mode.
- 3: For releasing wait state during reception of a slave device, write “FFH” to IICAn or set the WRELn bit.

Figure 16-31 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 (MSTS_n=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.

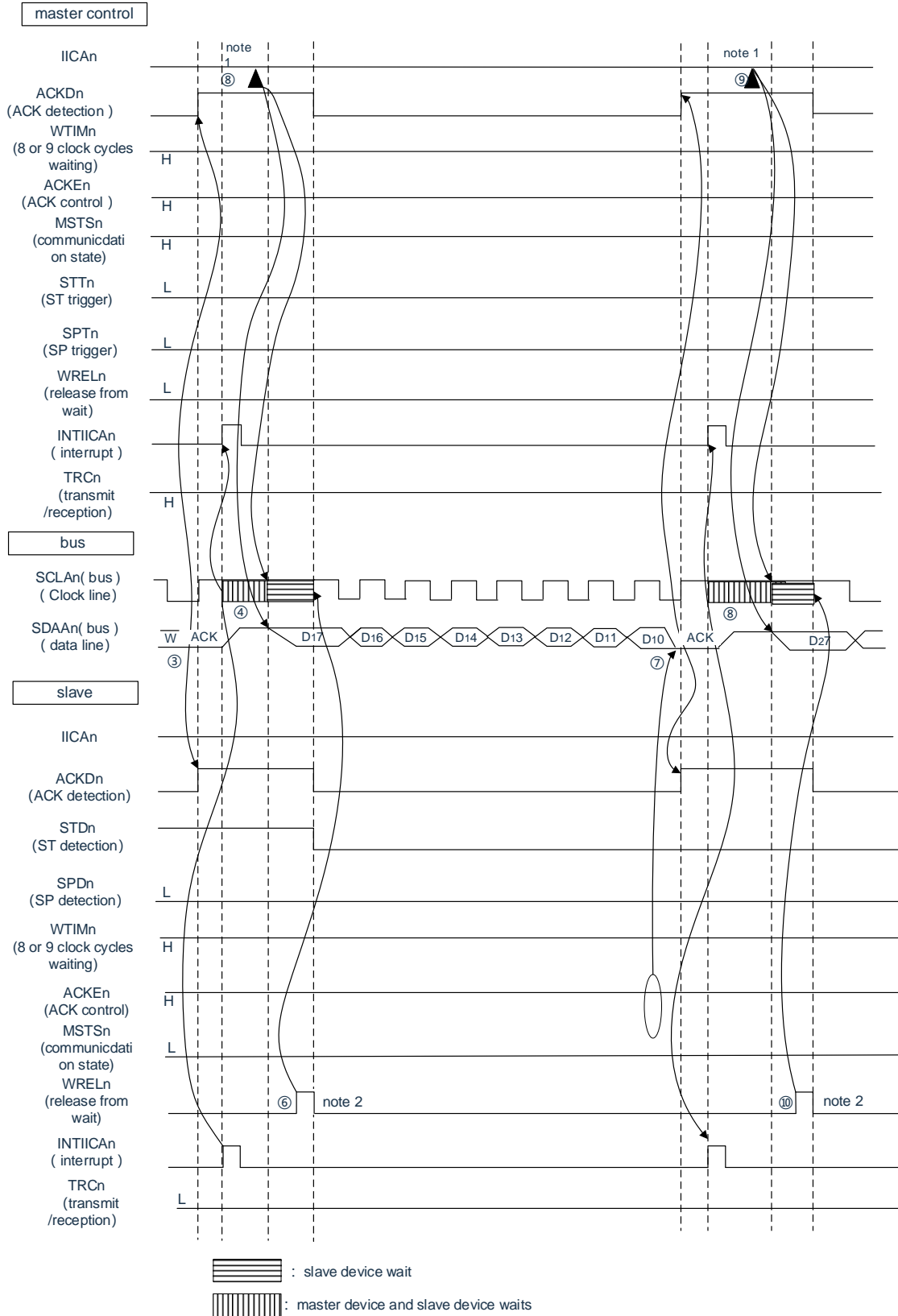
Note: If the sent address and the slave address are different, the slave does not return an ACK (NACK: SDAAn=1) to the master, and does not generate an INTIICAn interrupt (address matching interrupt) or enter a waiting state. However, the master generates ANTIICAn interrupts (address send end interrupts) for both ACK and NAK.

Remark:

1. ①~⑮ of Figure 16-31 shows a series of operational steps for data communication via the I2C bus.
 - “(1) Start Condition~Address~Data” of Figure 16-31 illustrates steps ①~⑥.
 - “(2) Address~Data~Data” of Figure 16-31 illustrates steps ③~⑩.
 - “(3) Data~Data~Stop Condition” of Figure 16-31 illustrates steps ⑦~⑮.
2. n=0

Figure 16-31 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.

2: For releasing wait state during reception of a slave device, write "FFH" to IICAn or set the WRELn bit.

Figure 16-31 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.

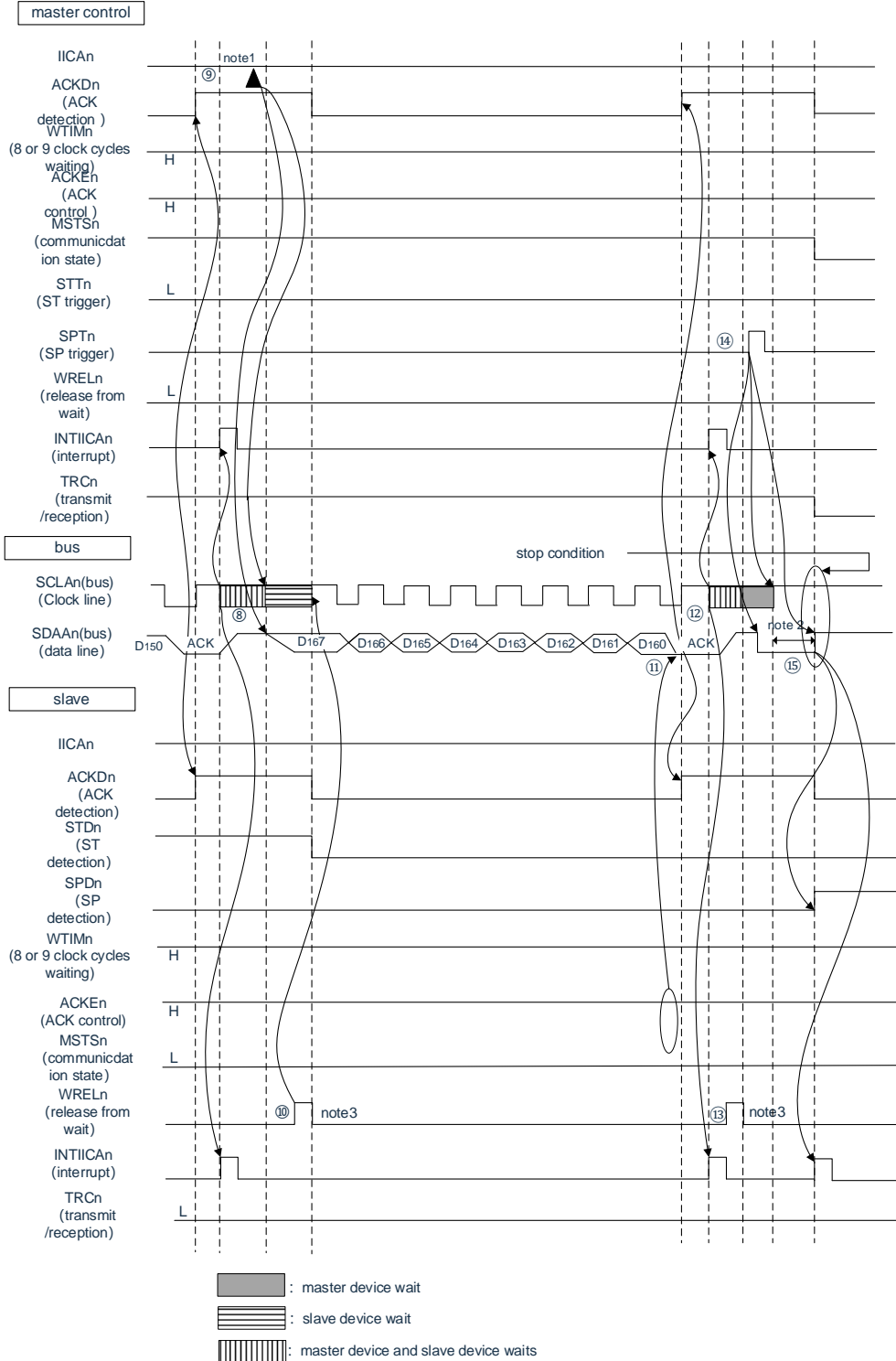
Note: If the sent address and the slave address are different, the slave does not return an ACK (NACK: SDAAn=1) to the master and does not generate an INTIICAn interrupt (address matching interrupt) or enter a waiting state. However, the master generates ANTIICAn interrupts (address send end interrupts) for both ACK and NAK.

Remark:

1. ①~⑮ of Figure 16-31 shows a series of operational steps for data communication via the I2C bus.
 - "(1) Start Condition~Address~Data" of Figure 16-31 illustrates steps ①~⑥.
 - "(2) Address~Data~Data" of Figure 16-31 illustrates steps ③~⑩.
 - "(3) Data~Data~Stop Condition" of Figure 16-31 illustrates steps ⑦~⑮.
2. n=0

Figure 16-31 Example of master to slave communication
(Master: selects 9 clocks to wait, slave: selects 9 clocks to wait) (3/4)

(1) 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.

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.

3: For releasing wait state during reception of a slave device, write "FFH" to IICAn or set the WRELn bit.

Figure 16-31 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).

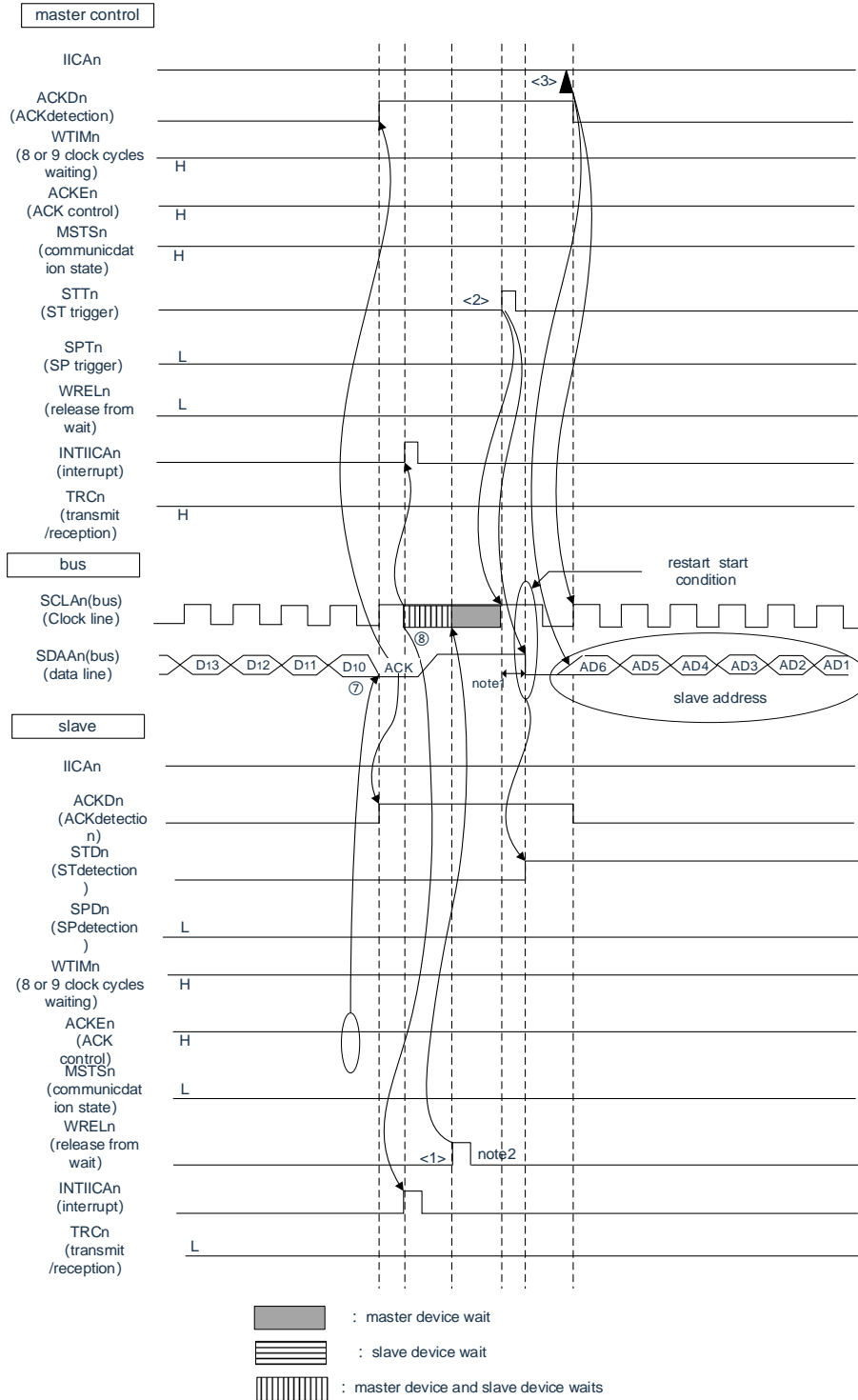
⑮ If a stop condition is generated, the slave detects the stop condition and generates an interrupt (INTIICAn: Stop condition interrupt).

Remark:

- ①~⑮ of Figure 16-31 shows a series of operational steps for data communication via the I²C bus.
 - “(1) Start Condition~Address~Data” of Figure 16-31 illustrates steps ①~⑥.
 - “(2) Address~Data~Data” of Figure 16-31 illustrates steps ③~⑩.
 - “(3) Data~Data~Stop Condition” of Figure 16-31 illustrates steps ⑦~⑮.
- n=0

Figure 16-31 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.

2: For releasing wait state during reception of a slave device, write "FFH" to IICAn or set the WRELn bit.

Figure 16-31 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).

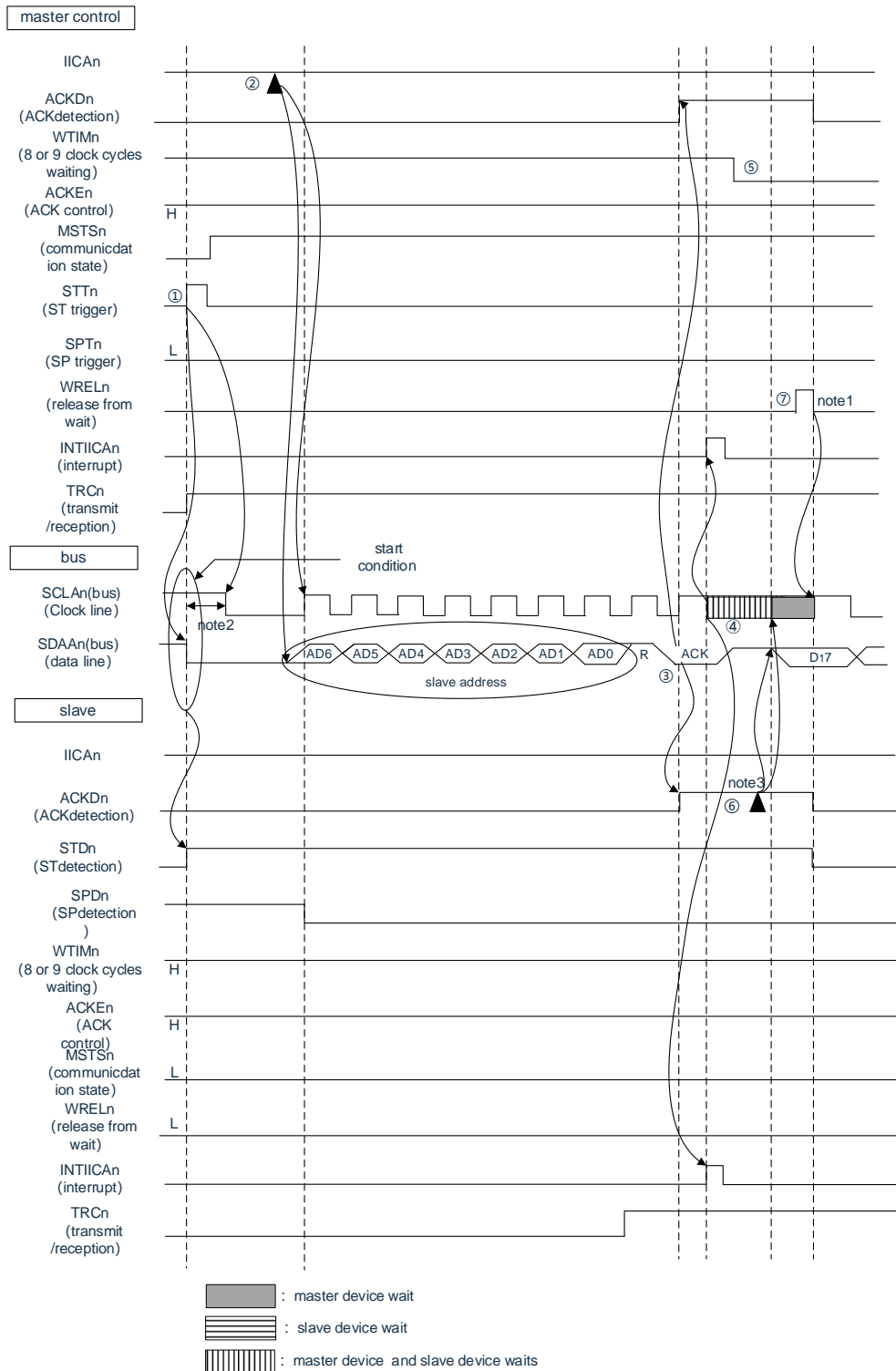
<2> The start condition trigger is set again by the master device (STTn = 1) and a start condition (i.e. SCLAn =1 changes SDAAn from 1 to 0) is generated once the bus clock line goes high (SCLAn = 1) and the bus data line goes low (SDAAn = 0) after the restart condition setup time has elapsed. When the 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 16-32 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.

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 μ s when specifying standard mode and at least 0.6 μ s when specifying fast mode.

3: To release the slave from waiting during transmission, you must write data to the IICn instead of setting the WRELn bit.

Figure 16-32 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 relieves the wait (WRELn=1) and begins data transfer from the slave.

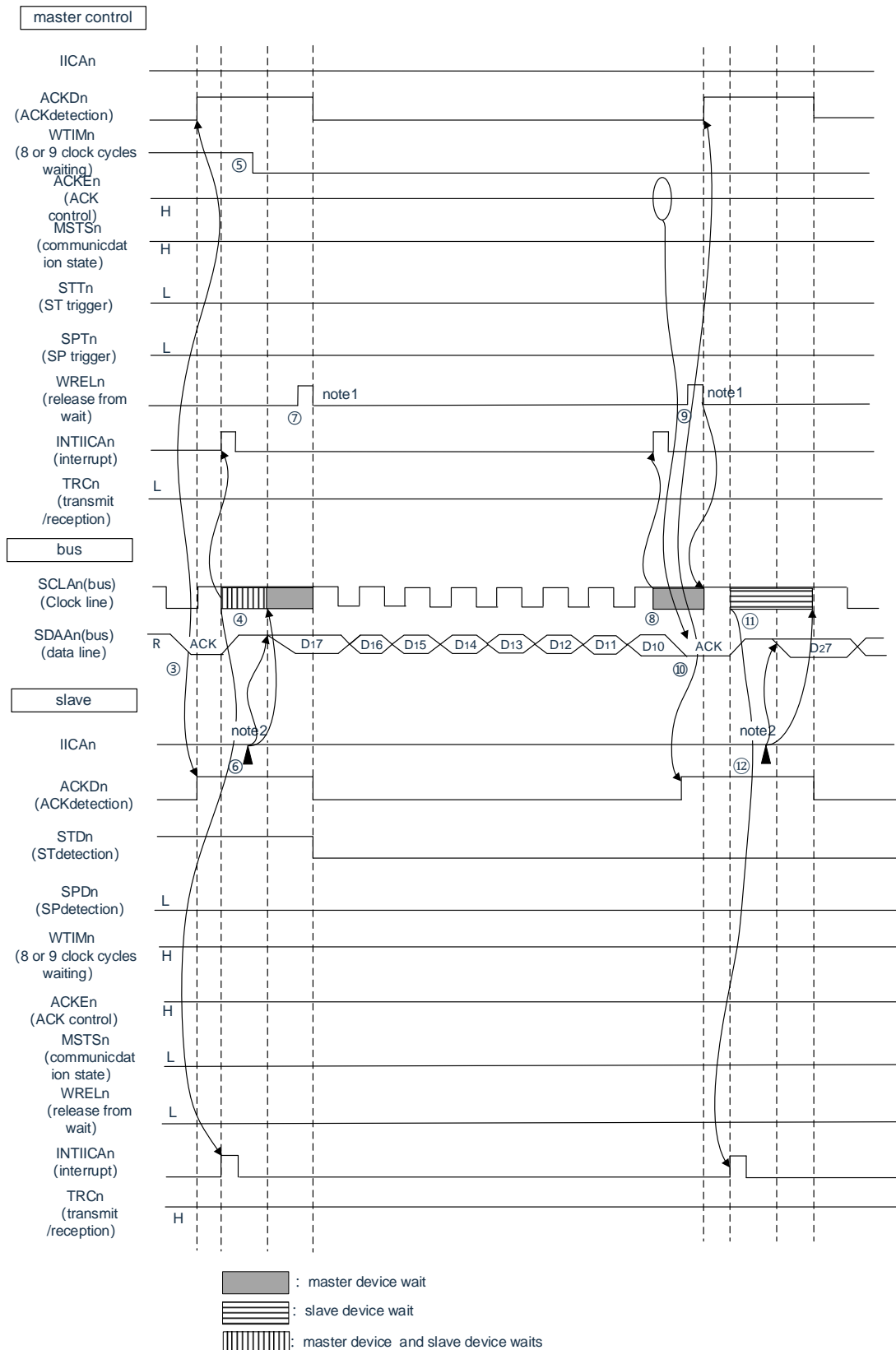
Note: If the sent address and the slave address are different, the slave does not return an ACK (NACK: SDAAn=1) to the master and does not generate an INTIICAn interrupt (address matching interrupt) or enter a waiting state. However, the master generates ANTIICAn interrupts (address send end interrupts) for both ACK and NAK.

Remark:

1. ①~⑱ of Figure 16-32 shows a series of operational steps for data communication via the I²C bus.
 - “(1) Start Condition ~ Address ~ Data” of Figure 16-32 illustrates steps ①~⑦.
 - “(2) Address~Data~Data” of Figure 16-32 illustrates steps ③~⑫.
 - “(3) Data~Data~Stop Condition” of Figure 16-32 illustrates steps ⑧~⑱.
2. n=0

Figure 16-32 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 16-32 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 relishes 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.

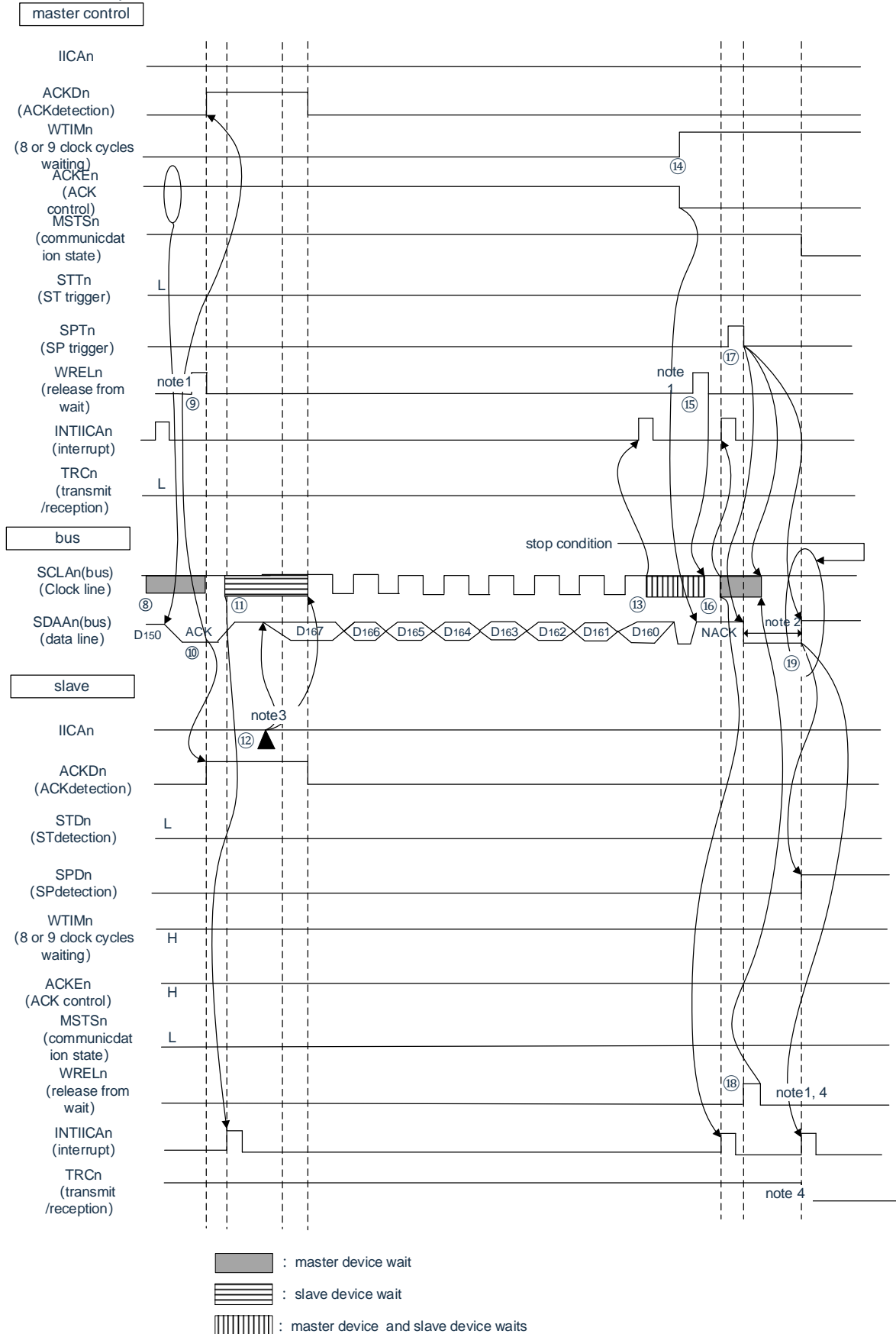
Note: If the sent address and the slave address are different, the slave does not return an ACK (NACK: SDAAn=1) to the master and does not generate an INTIICAn interrupt (address matching interrupt) or enter a waiting state. However, the master generates ANTIICAn interrupts (address send end interrupts) for both ACK and NAK.

Remark:

1. ①~⑱ of Figure16-32 shows a series of operational steps for data communication via the I²C bus.
 - “(1) Start Condition ~ Address ~ Data” of Figure16-32 illustrates steps ①~⑦.
 - “(2) Address~Data~Data” of Figure16-32 illustrates steps ③~⑫.
 - “(3) Data~Data~Stop Condition” of Figure16-32 illustrates steps ⑧~⑱.
2. n=0

Figure 16-32 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.

- 2: After the stop condition is issued, the time from the SCLAn pin signal to generate the stop condition is at least 4.0 μ s when set to standard mode and at least 0.6 μ s when set to fast mode.
- 3: To release the slave from waiting during transmission, you must write data to the IICn instead of setting the WRELn bit.
- 4: The TRCn bit is cleared if the wait is released by setting the WRELn bit during the slave transmission.

Figure 16-32 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 ($SDAAn=0$)
- ⑭ The master sets the NACK Acknowledge ($ACKEn=0$) and changes the wait sequence to the 9th clock ($WTIMn=1$). If the master relishes 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 17 IrDA

IrDA enables the transmission and reception of IrDA communication waveforms in accordance with the IrDA (InfraredDataAssociation) 1.0 protocol in cooperation with the Universal Serial Communication Unit (SCI).

17.1 Function of IrDA

If the IrDA function is set to active through the IRE bit of the IRCR register, SCI's TxD2 signal and RxD2 signal can encode or decode the waveform that conforms to the IrDA1.0 protocol (IrTxD/IrRxD pins), and then implement infrared transmission and reception that supports the IrDA1.0 protocol by connecting the transmitter or receiver that transmits/receives infrared rays.

In systems that support the IrDA1.0 protocol, after communication begins at a transfer rate of 9600bps, the transfer rate can be changed as needed. IrDA does not have a built-in function to automatically change the transfer rate, so the settings must be changed by software to change the transfer rate.

When selecting a high-speed internal oscillator ($f_{IH}=24, 12, 6, 3\text{MHz}$), the following baud rates can be set.

- 115.2kbps, 57.6kbps, 38.4kbps, 19.2kbps, 9600bps, 2400bps

A schematic block diagram of the collaboration between IrDA and SCI is shown in Figure 17-1.

Figure 17-1 Block diagram of the cooperation between IrDA and SCI

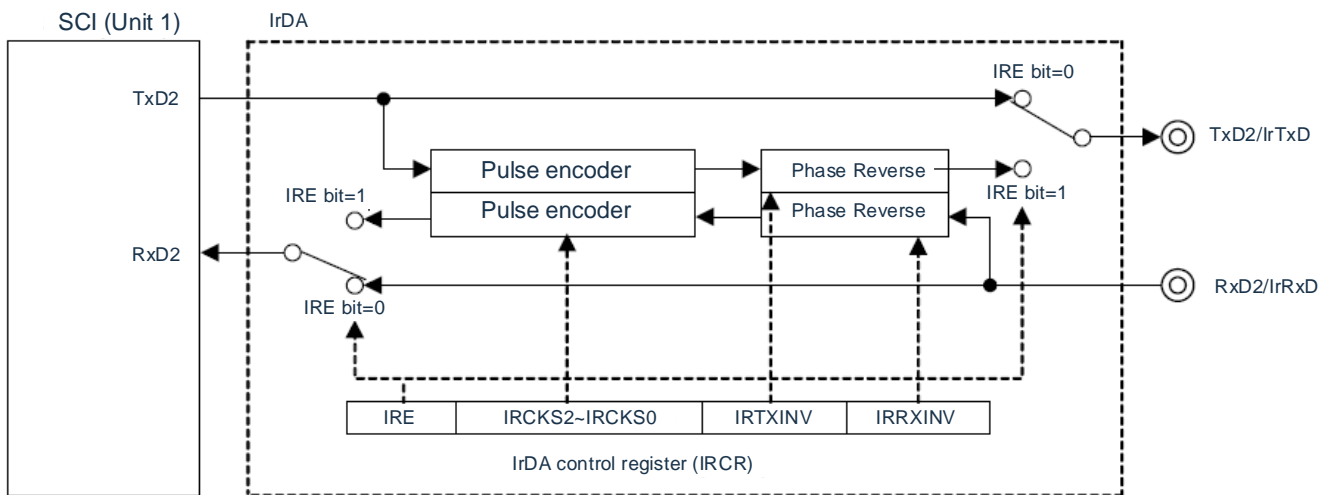


Table 17-1 Pin structure of the IrDA

Pin name	Input/output	Function
IrTxD	output	The output pin that sends the data
IrRxD	input	The input pin that receives the data

17.2 Registers for controlling IrDA

The IrDA function is controlled through the following registers.

- Peripheral enable register 0 (PER0)
- IrDA control register (IRCR)

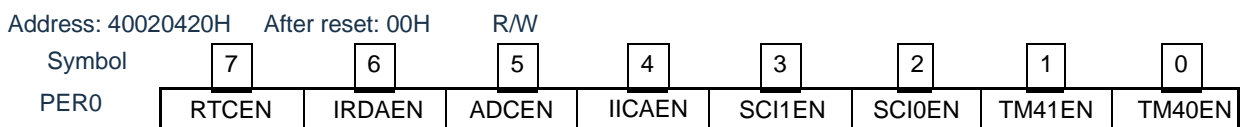
17.2.1 Peripheral enable register 0 (PER0)

The PER0 register is a register that sets the clock to be enable or disable to be supplied to each peripheral hardware. Reduce power consumption and noise by stopping clocking unused hardware.

To use IrDA, you must set bit6 (IRDAEN) to "1".

The PER0 register is set via an 8-bit memory operation instruction. After the reset signal is generated, the value of this register becomes "00H".

Figure 17-2 Format of peripheral enable register 0 (PER0)



IRDAEN	IrDA input clock control
0	Stops input clock supply <ul style="list-style-type: none"> • SFR used by the IrDA cannot be written. • IrDA is in the reset status.
1	Enables input clock supply <ul style="list-style-type: none"> • SFR used by the IrDA can be read and written.

Notice 1. When setting the IrDA, the IRDAEN bit must be set to "1" first. When the IRDAEN bit is "0", the write operation of the IrDA control register is ignored, and the read value is all initial.

17.2.2 IrDA control register (IRCR)

This is a register that controls the IrDA function. Selects for polarity switching of received and transmitted data, clock selection for IrDA, and switching of serial input/output pin functions (typically serial and IrDA functions). The IRCR register is set via an 8-bit memory operation command. After the reset signal is generated, the value of this register becomes “00H”.

Figure 17-3 Format of IrDA control register (IRCR)

Address: 40044000H	After reset: 00H R/W							
Symbol	7	6	5	4	3	2	1	0
IRCR	IRE	IRCKS2	IRCKS1	IRCKS0	IRTXINV	IRRXINV	0	0

IRE	IrDA enable
0	Serial input/output pins are used as the usual serial function
1	The serial input/output pins are used as IrDA functions

IRCKS2	IRCKS1	IRCKS0	Clock selection for IrDA
0	0	0	B3/16 (B = bit rate)
0	0	1	$f_{CLK}/2$
0	1	0	$f_{CLK}/4$
0	1	1	$f_{CLK}/8$
1	0	0	$f_{CLK}/16$
1	0	1	$f_{CLK}/32$
1	1	0	$f_{CLK}/64$
1	1	1	Disable settings

IRTXINV	Polarity switching of IrTxD data
0	IrTxD output of the transmitted data
1	Reverse the data sent for IrTxD output

IRRXINV	Polarity switching of IrRxD data
0	Use the input data from the IrRxD pin as the receive data
1	The data after inverting the input data of the IrRxD pin is used as the received data

Notice1. Bit1 and bit0 must be set to “0”.

2. IRCKS [2:0] bits, IRTXINV bits, and IRRXINV bits can be set only when the IRE bit is “0”.

17.3 Operation of IrDA

17.3.1 Operating steps for IrDA communication

(1) Initial setup process for IrDA communication

Follow the steps below to initialize IrDA.

1. Set the IRDAEN bit of the PER0 register to "1".
2. Set the IRCR register.
3. Set the relevant registers for SCI (refer to the setting steps of the UART mode).

(2) Stop process for IrDA communication

1. Set the IrTXD pin state after IrDA communication stops by setting the port register and the port mode register.

Remark When performing an IrDA reset via step 3, the IrTXD pin may change the output state by switching to the data output of the usual serial interface UART.

- Output low level from the IrTXD pin
Set the port register to "0". Immediately after this setting, the IrTXD pin is fixed low.
 - Output high level from the IrTXD pin
Set the port register to "1". With this setting, the IrTXD pin is fixed to high immediately after the IrDA reset in step 3.
 - Set the IrTXD pin to the Hi-Z state to set the port mode register to "1".
Immediately after this setting, the IrTXD pin changes to the Hi-Z state.
2. Set the STm0 and STm1 bits of the STm register (SCI's associated register) to "1" (to stop the operation of SCI's channel 0 and channel 1).
 3. Reset IrDA by setting IRDAEN of the PER0 register to "0".

You cannot set the STm0 and STm1 bits of the STm register to "1" or the IRE bit of the IrDA to "0" in cases other than the above steps.

(3) To send an IrDA frame error

When a frame error occurs during IrDA communication, the following settings must be made in order to set the state in which subsequent data can be received.

1. Set the STm1 bit of SCI's STm register to "1" (to stop the operation of SCI's channel 1).
2. Set the SSm1 bit of the SSm register of SCI to "1" (start the operation of channel 1 of SCI).

Remark m: unit number (m=0)

For frame error handling for SCI, refer to Chapter 14, Universal Serial Communication Unit.

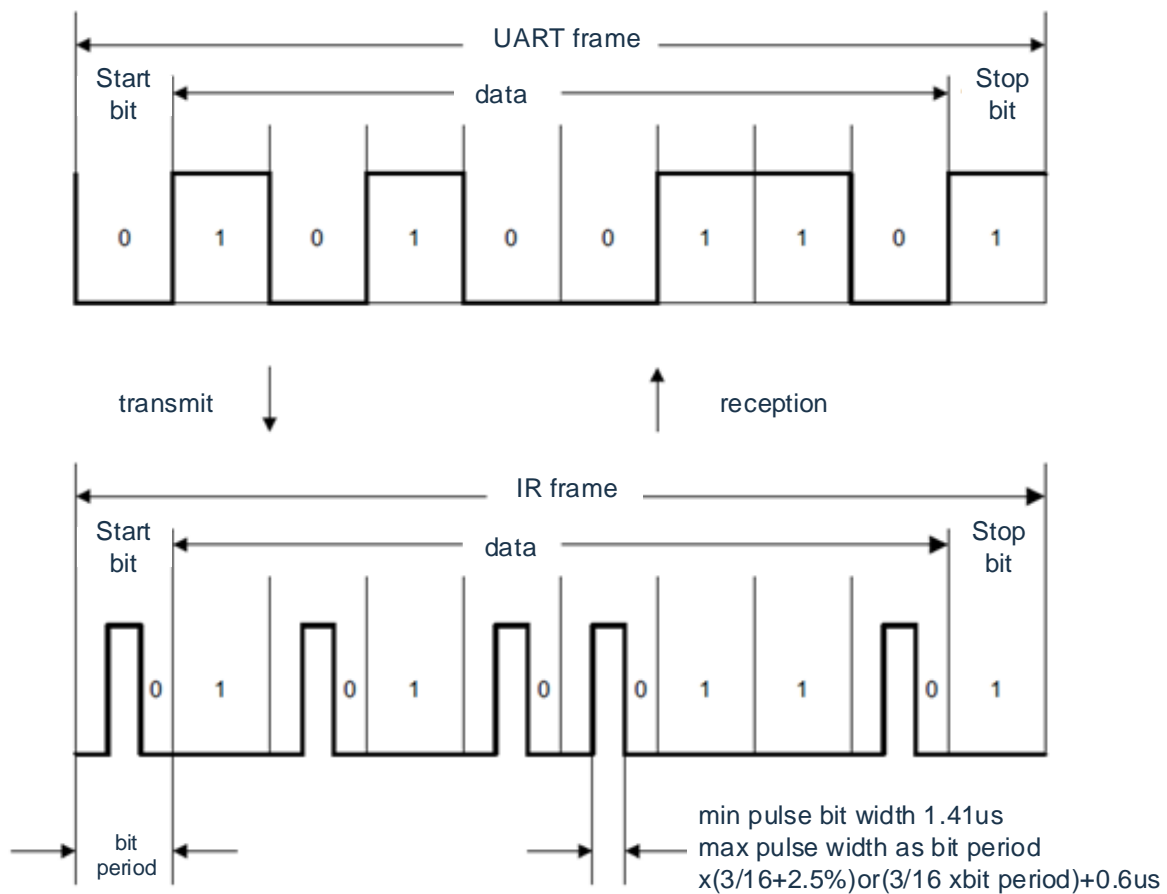
17.3.2 Transmission

At the time of transmission, the output signal (UART frame) from the SCI is converted to an IR frame via IrDA (see Figure 17-4).

At IRTXINV bit "0" and serial data is "0", the output bit period (1-bit width period) x 3/16 high level pulse (initial value). In addition, the high pulse width can be changed according to the setting value of IRCKS2 to IRCKS0 bits. As standard, a minimum pulse width of 1.41 μ s for high levels is specified for a maximum of (3/16+2.5%) x bit period, or (3/16 x bit period) of +0.6 μ s.

When the CPU or peripheral hardware clock (f_{CLK}) is 24MHz, the minimum high pulse width that can be set is 1.5 μ s (the condition that the high-level pulse width specified above is not less than 1.41 μ s is satisfied). In addition, when the serial data is "1", no pulse is output.

Figure 17-4 Transmit/receive operation diagram of IrDA



17.3.3 Reception

When received, the data of the IR frame is converted to a UART frame via IrDA and then entered into the SCI. When the IRRXINV bit is "0" and a high pulse is detected, low data is output. If there is no pulse within the 1-bit period, high level data is output. Care must be taken that pulses smaller than the minimum pulse width of 1.41 μ s cannot be recognized.

17.3.4 High level pulse width selection

If the pulse width at the time of transmission is less than the bit rate $\times 3/16$, the applicable IRCKS2 ~ IRCKS0 bit setting (minimum pulse width) and the high-level pulse width at the time of setting are shown in Table 17-2.

Table 17-2 Setting values of IRCKS2 ~ IRCKS0 bits

f _{CLK} [MHz]	Item	<Upper>Bit rate[kbps]					
		<Lower> Bit rate $\times 3/16$ [us]					
		2.4	9.6	19.2	38.4	57.6	115.2
		78.13	19.53	9.77	4.87	3.26	1.63
1	IRCKS2~IRCKS0	001	001	001	-Note1	-Note1	-Note1
	High level pulse width[μ s]	2.00	2.00	2.00	-Note1	-Note1	-Note1
2	IRCKS2~IRCKS0	010	010	010	010	010	-Note1
	High level pulse width[μ s]	2.00	2.00	2.00	2.00	2.00	-Note1
3	IRCKS2~IRCKS0	011	011	011	011	011	-Note1
	High level pulse width[μ s]	2.67	2.67	2.67	2.67	2.67	-Note1
4	IRCKS2~IRCKS0	011	011	011	011	011	000Note2
	High level pulse width[μ s]	2.00	2.00	2.00	2.00	2.00	1.50
6	IRCKS2~IRCKS0	100	100	100	100	100	000Note2
	High level pulse width[μ s]	2.67	2.67	2.67	2.67	2.67	1.50
8	IRCKS2~IRCKS0	100	100	100	100	100	000Note2
	High level pulse width[μ s]	2.00	2.00	2.00	2.00	2.00	1.50
12	IRCKS2~IRCKS0	101	101	101	101	101	000Note2
	High level pulse width[μ s]	2.67	2.67	2.67	2.67	2.67	1.50
16	IRCKS2~IRCKS0	101	101	101	101	101	000Note2
	High level pulse width[μ s]	2.00	2.00	2.00	2.00	2.00	1.50
24	IRCKS2~IRCKS0	110	110	110	110	110	000Note2
	High level pulse width[μ s]	2.67	2.67	2.67	2.67	2.67	1.50

Note 1. "-" indicates that the communication standard is not met.

2. The pulse width cannot be less than the bit rate $\times 3/16$.

17.4 Cautions on using IrDA

1. The operating clock of the IrDA can enable or disable by a peripheral enable register setting. The initial state is to disable the supply of clocks, so the registers cannot be accessed. Before the registers can be set, the peripheral allow registers must be set to allow the state of the IrDA operating clock to be provided.
2. In sleep mode, the IrDA function is continuously operated.
3. During IrDA communication, the initialization function of SCI (SS bit = 1) is prohibited.
4. The IRRXINV bit, IRTXINV bit, and IRCKS [2:0] bit of the IRCR register can be set only when the IRE bit is "0".

Chapter 18 Enhanced DMA

18.1 Function of DMA

DMA is a function that does not use a CPU and transfers data between memories. Initiate DMA for data transfer via peripheral function interrupts. When DMA and CPU access the same unit in FLASH, SRAM0, SRAM1, or peripheral modules at the same time, their bus usage rights are higher than those of the CPU. When DMA and CPU access different units in FLASH, SRAM0, SRAM1, or peripheral modules, respectively, the two do not interfere with each other and can be executed in parallel.

The specifications of DMA are shown in Table 18-1.

Table 18-1 DMA specification (1/2)

Item		Specification
Start the source		Up to 24 boot sources
Distributable control data		24 groups
The address space that can be transferred	Address space	Full address range space
	source	Full address range space is optional
	target	Full address range space is optional
The maximum number of transfers	Normal mode	65535 times
	Repeating pattern	65535 times
The maximum transfer block size	Normal mode (8-bit transfer).	65535 bytes
	Normal mode (16-bit transmission).	131070 bytes
	Normal mode (32-bit transfer).	262140 bytes
	Repeating pattern	65535 bytes
Transmission units		8-bit/16-bit/32-bit
Transfer mode	Normal mode	Ends after transferring the DMACTj register from "1" to "0".
	Repeating pattern	At the end of the transfer of the DMACTj register from "1" to "0", the address of the duplicate area is initialized before the DMRLDj is placed. The value of the register is reloaded into the DMACTj register and then transferred.
Address control	Normal mode	Fixed or incremental
	Repeating pattern	Fixed or incremented distinct addresses.
The priority of the startup source		Refer to Table 18-5 DMA startup source and vector address

Table18-1 DMA specification (2/2)

item		Specification
Interrupt the request	Normal mode	When transferring the DMACTj register from "1" to "0", an interrupt from the startup source is requested to the CPU and interrupt handling is performed.
	Repeating pattern	The RPTINT bit of the DMACRj register is "1" to allow interrupts to be generated) and the DMACTj register is transferred from "1" to "0" when the data transfer is made The CPU requests an interrupt from the start source and performs interrupt handling.
The transfer starts		If the DMAENi0~DMAENi7 bits of the DMAENi register is "1" (boot allowed), the transmission of data begins each time the DMA boot source occurs.
Delivery stopped	Normal mode	<ul style="list-style-type: none"> • Set DMAENi0~DMAENi7 bits to "0" (boot is prohibited). • When the DMACTj register changes from "1" to "0" at the end of the data transfer
	Repeating pattern	<ul style="list-style-type: none"> • Set DMAENi0~DMAENi7 bits to "0" (boot is prohibited). • When the RPTINT bit is "1" (allows interrupts to occur) and the DMACTj register changes from "1" to "0" at the end of the data transfer

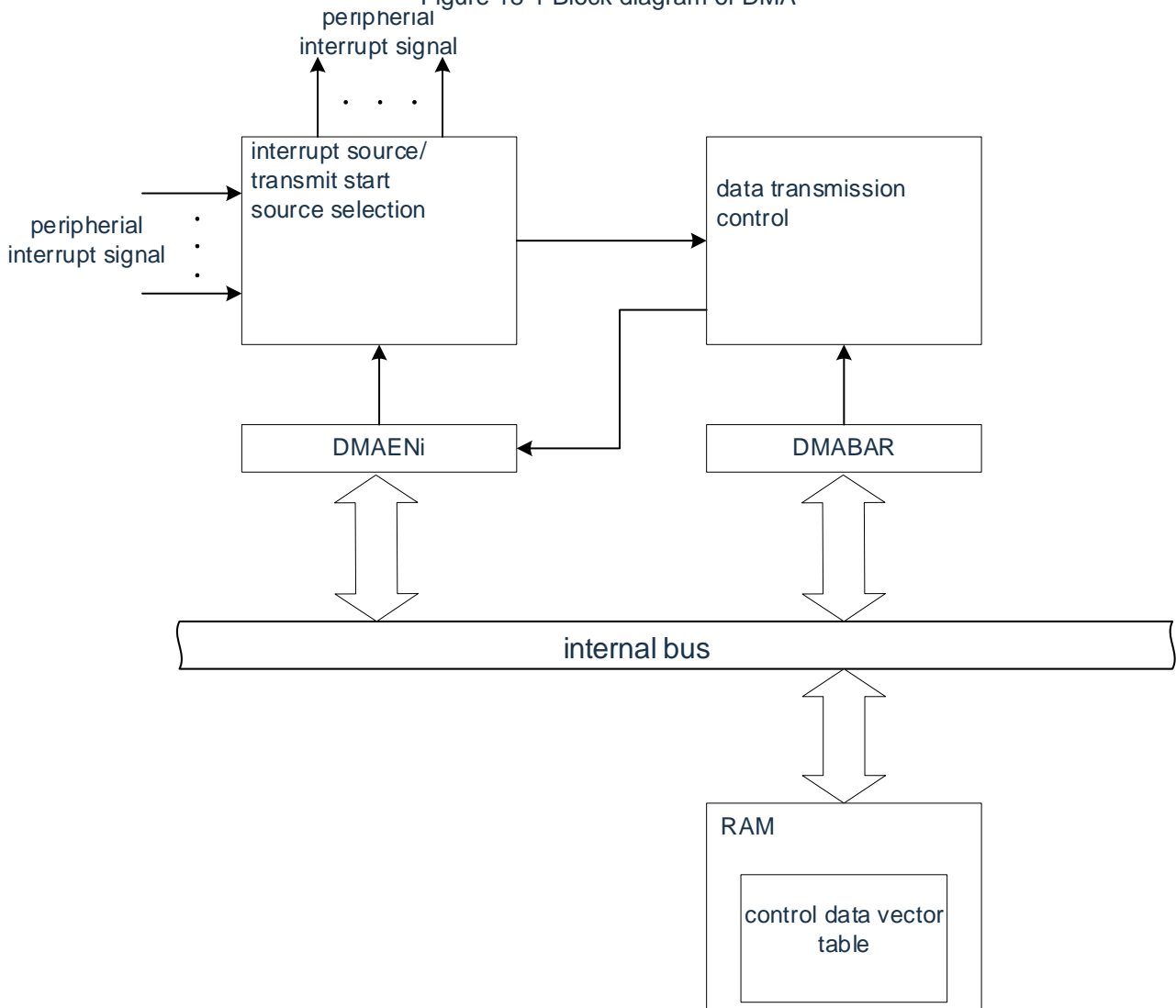
Note In deep sleep mode because the flash memory stops functioning and therefore cannot be used as a DMA transfer source.

Remark i=0~2, j=0~23

18.2 Structure of DMA

Figure 18-1 shows block diagram of the DMA.

Figure 18-1 Block diagram of DMA



18.3 Registers for controlling DMA

The registers that control the DMA are shown in Table 18-2.

Table 18-2 Registers for controlling DMA

Register name	Symbol
Peripheral enable register 1	PER1
DMA boot enable register 0	DMAEN0
DMA boot enable register 1	DMAEN1
DMA boot enable register 2	DMAEN2
DMA base address register	DMABAR

The control data of the DMA is shown in Table 18-3.

The DMA control data is distributed in the DMA control data area of the RAM. The DMA control data area and the 416-byte region containing the DMA vector table area (the starting address where the control data is saved) are set via the DMABAR register.

Table 18-3DMA control data

Register name	Symbol
DMA control register j	DMACRj
DMA block size register j	DMBLSj
DMA transfer count register j	DMACTj
DMA transfer number of times to reload register j	DMRLDj
DMA source address register j	DMSARj
DMA destination address register j	DMDARj

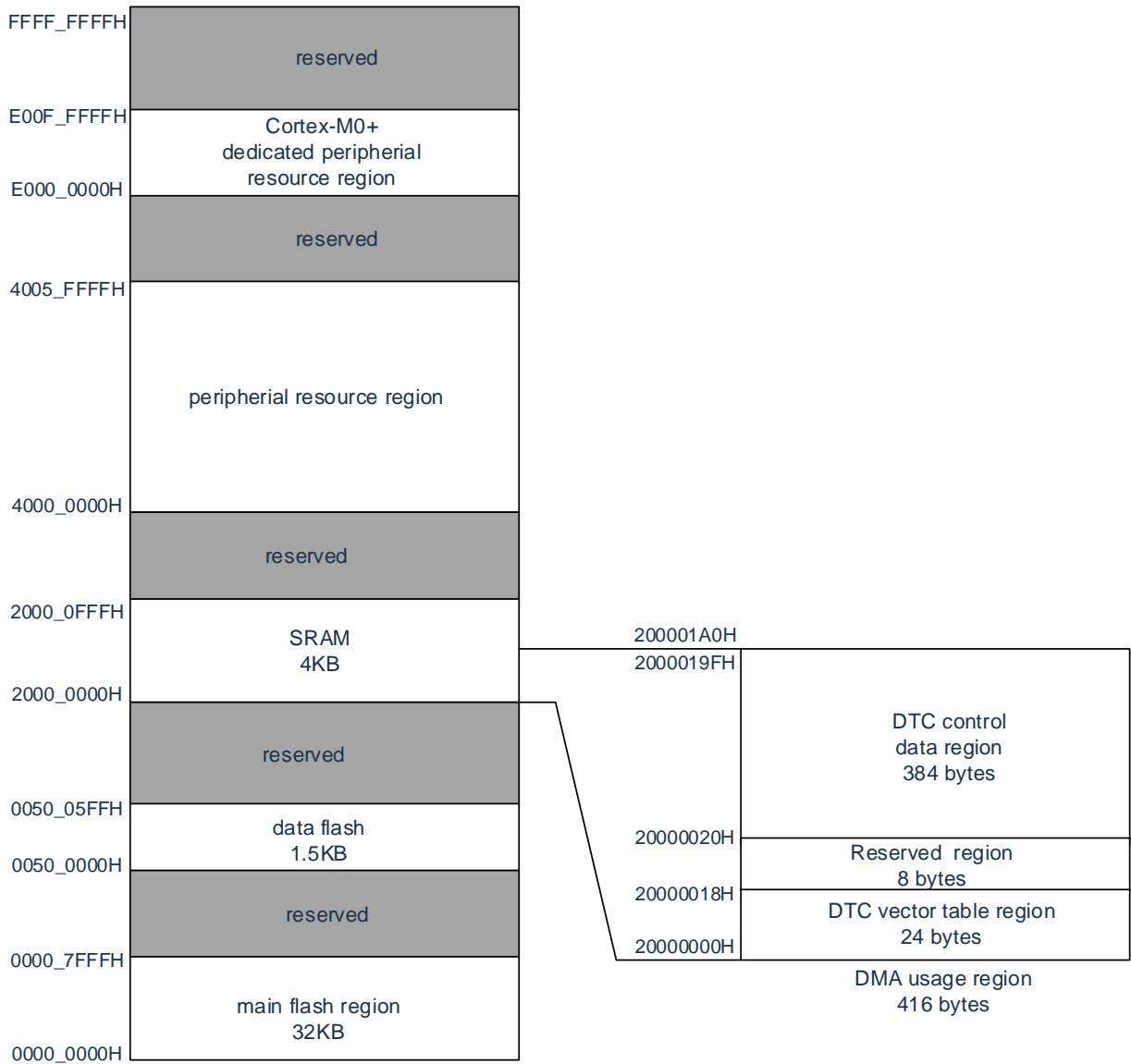
Remark j=0~23

18.3.1 DMA control data areas and DMA vector table areas allocation

The control data allocated to the DMA and the 416-byte region of the vector table are set to the RAM area via the DMABAR register.

An example of a memory image with a DMABAR register set to “20000000H” is shown in Figure 18-2. The 384 bytes of DMA control data area in the DMA unused space can be used as RAM.

Figure 18-2 Example of memory image when the DMABAR register is set to “20000000H”



18.3.2 Control data allocation

Starting from the start address, follow DMACR_j, DMBSL_j, DMACT_j, DMRLD_j, DMSAR_j, DMDAR_j (j=0~23) registers are assigned control data sequentially.

The start address is set by the DMABAR register, and the lower 10 bits are set separately by the vector table assigned by each startup source.

The distribution of control data is shown in Figure 18-3.

Notice1. The DMAEN_{i0}~DMAEN_{i7} bits must be “0” in the corresponding DMAEN_i (i=0~2). (Disable Startup) when changing DMACR_j, DMBSL_j, DMACT_j, DMRLD_j, DMSAR_j, DMSAR_j, Data for the DMDAR_j register.

2. DMACR_j, DMBSL_j, DMACT_j, DMRLD_j, DMSAR_j and DMA cannot be transmitted via DMA Access to DMDAR_j.

Figure 18-3 Control data allocation (DMABAR set to 2000000H)

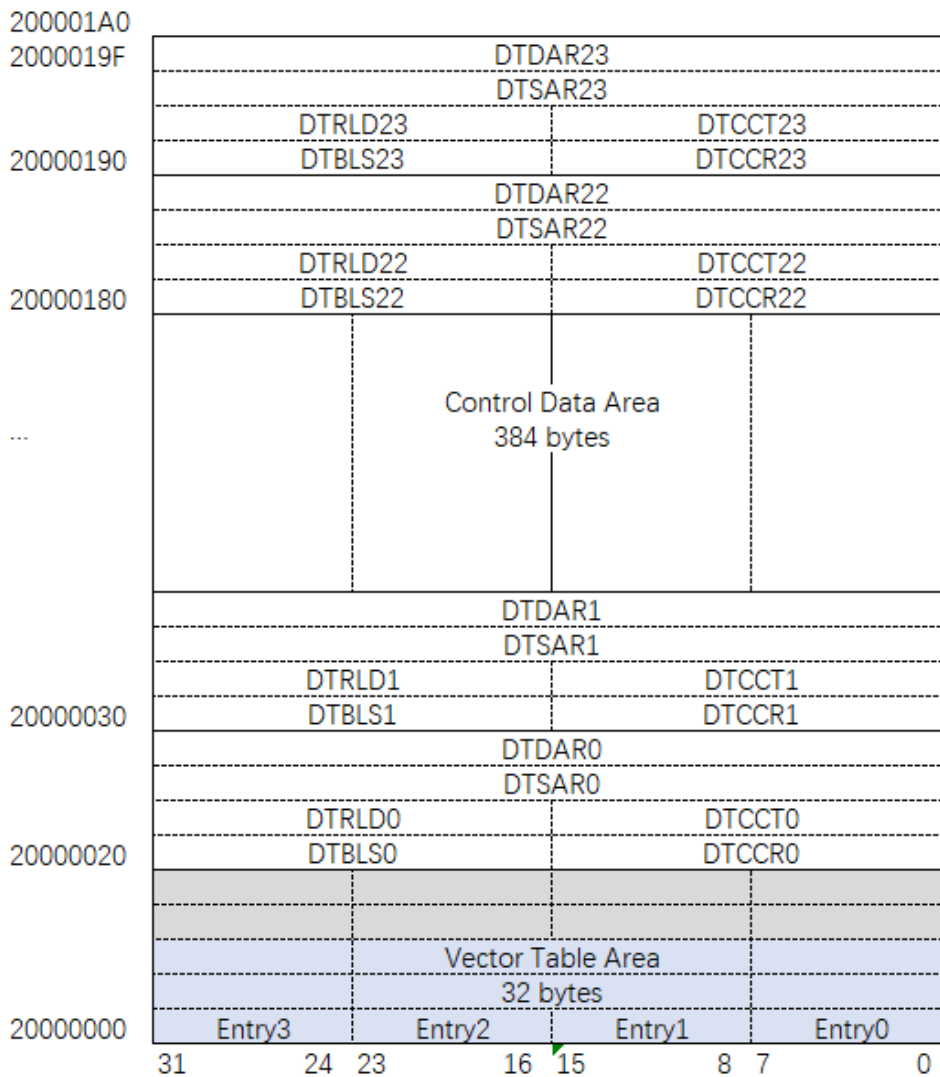


Table 18-4 Starting address of control data

j	Address	j	Address
11	baseaddr+D0H	23	baseaddr+190H
10	baseaddr+C0H	22	baseaddr+180H
9	baseaddr+B0H	21	baseaddr+170H
8	baseaddr+A0H	20	baseaddr+160H
7	baseaddr+90H	19	baseaddr+150H
6	baseaddr+80H	18	baseaddr+140H
5	baseaddr+70H	17	baseaddr+130H
4	baseaddr+60H	16	baseaddr+120H
3	baseaddr+50H	15	baseaddr+110H
2	baseaddr+40H	14	baseaddr+100H
1	baseaddr+30H	13	baseaddr+F0H
0	baseaddr+20H	12	baseaddr+E0H

Remark baseaddr: The Setting value of the DMABAR register

18.3.3 Vector table

Once the DMA is started, the control data is determined by reading the data from the vector table allocated by each startup source, and the control data assigned to the DMA control data area is read.

The DMA boot source and vector addresses are shown in Table 18-5. Each startup source vector table has 1 byte, holds the data from “00H” to “17H”, and selects 1 from 24 groups of control data Group data. The upper 22 bits of the vector address are set by the DMABAR register, and the lower 10 bits are assigned “00H” to “17H” for the corresponding startup source.

Notice The DMAENi0~DMAENi7 bits must be “0” in the corresponding DMAENi (i=0~2) registers (Disable Startup) when setting the starting address of the DMA control data area in the vector table.

Figure 18-4 Starting address and vector table of control data when DMABAR register is “2000000H” (example)

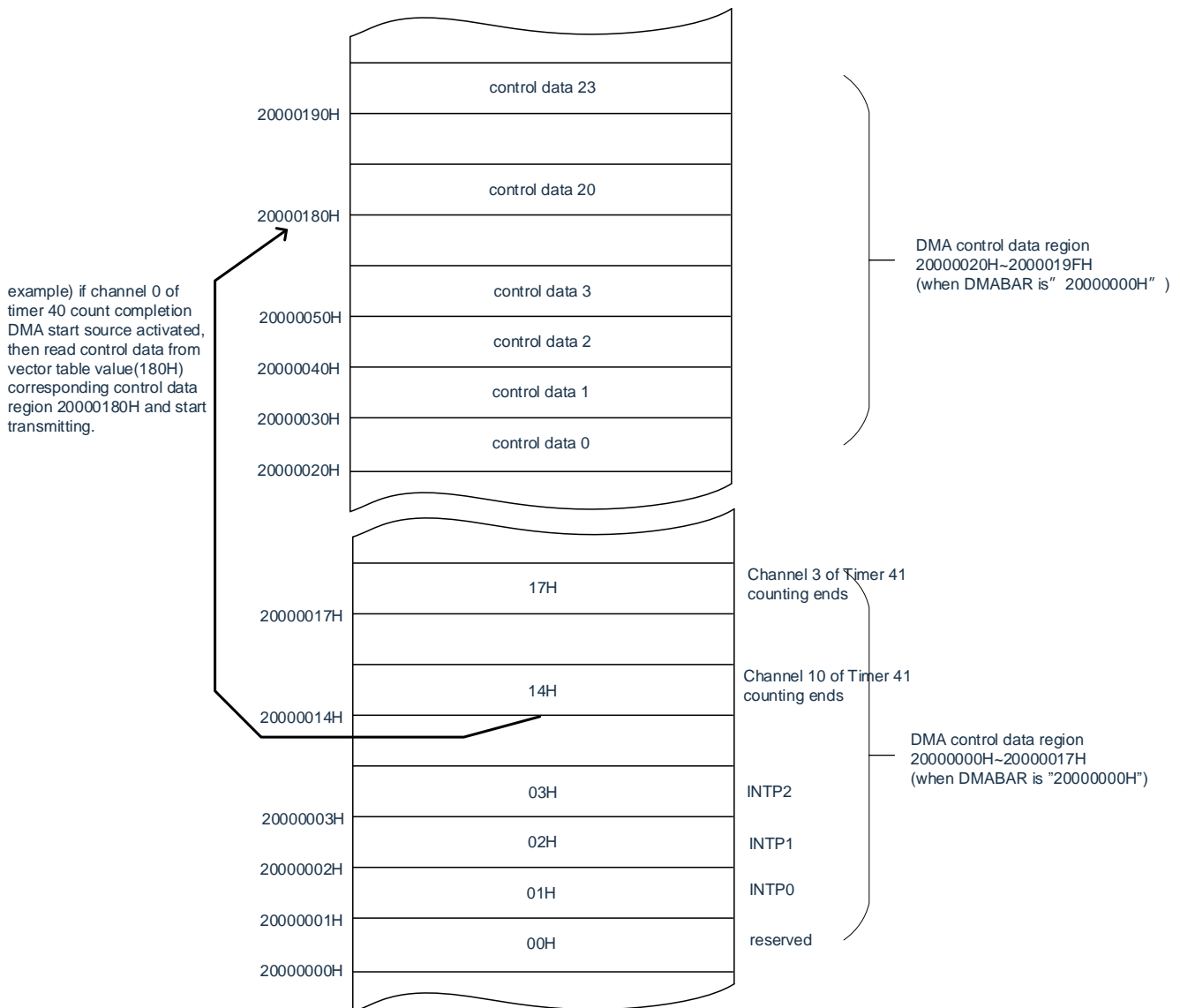




Table 18-5 DMA startup source and vector address

DMA start source (the source where the interrupt request occurred).	Source number	Address of vector	Priority
Flash read-write erase ends	0	The setting address of the DMABAR register is +00H	high   low
INTP0	1	The setting address of the DMABAR register is +01H	
INTP1	2	The setting address of the DMABAR register is +02H	
INTP2	3	The setting address of the DMABAR register is +03H	
INTP3	4	The setting address of the DMABAR register is +04H	
The A/D conversion ends	5	The setting address of the DMABAR register is +05H	
Comparator detect 0	6	The setting address of the DMABAR register is +06H	
Comparator detect 1	7	The setting address of the DMABAR register is +07H	
The end of transmission received by UART0 / the end of transmission of SSPI01 or the end of transmission of buffer NULL/IIC01	8	The setting address of the DMABAR register is +08H	
The end of the UART0 transmission / the end of the SSPI00 transmission or the end of the buffer NULL/IIC00 transmission	9	The setting address of the DMABAR register is +09H	
The end of transmission received by UART1 / the end of transmission of SSPI11 or the end of transmission of buffer NULL/IIC11	10	The setting address of the DMABAR register is +0AH	
End of transmission for UART1 transmission/end of transmission for SSPI10 or end of transmission for buffer NULL/IIC10/end of transmission for SPI	11	The setting address of the DMABAR register is +0BH	
The end of transmission received by UART2 / the end of transmission of SSPI21 or the end of transmission of buffer NULL/IIC21	12	The setting address of the DMABAR register is +0CH	
The end of the UART2 transmission / the end of the SSPI20 transmission or the end of the buffer null/IIC20 transmission	13	The DMABAR register is set to address +0DH	
IICA0 communication ends.	14	The setting address of the DMABAR register is +0EH	
A 15-bit interval timer generates a count interrupt	15	The setting address of the DMABAR register is +0FFH	
Timer40 ends with the count of channel 0 or capture	16	The setting address of the DMABAR register is +10H	
Timer40 for channel 1 counts or captures end	17	The setting address of the DMABAR register is +11H	
Timer40 for channel 2 counts or snaps ends	18	The setting address of the DMABAR register is +12H	
Timer40 ends with the count or snap of channel 3	19	The setting address of the DMABAR register is +13H	
Timer41 ends counting or snapping of channel 0	20	The setting address of the DMABAR register is +14H	
Timer41 ends with the counting or snapping of channel 1	21	The setting address of the DMABAR register is +15H	
Timer41 ends with the count or snap of channel 2	22	The setting address of the DMABAR register is +16H	
Timer41 ends with the counting or snapping of channel 3	23	The setting address of the DMABAR register is +17H	

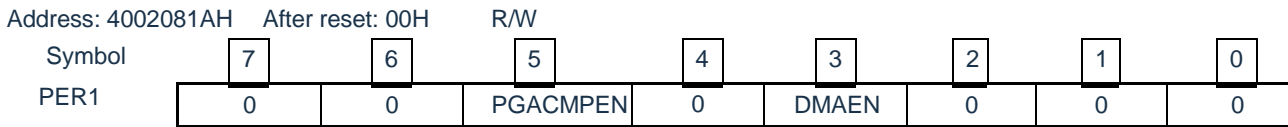
18.3.4 Peripheral enable register 1 (PER1)

The PER1 register is a register that sets the clock that enable or disables clocking each peripheral hardware. Reduce power consumption and noise by stopping clocking unused hardware.

To use DMA, bit3 (DMAEN) must be set to “1”.

The PER1 register is set via an 8-bit memory operation command. After the reset signal is generated, the value of this register becomes “00H”.

Figure 18-5 Format of peripheral enable register 1 (PER1)



DMAEN	Provides control of the input clock of the DMA
0	Stop supplying the input clock. • DMA cannot be run.
1	An input clock is provided. • DMA can run.

18.3.5 DMA control register j(DMACRj) (j=0~23)

The DMACRj register controls the operating mode of the DMA.

Figure 18-6 Format of DMA control register j (DMACRj)

Address: Refer to “18. 3. 2 Control data allocation”.

After reset: Undefined value R/W

Symbol:	15	14	13	12	11	10	9	8
DMACRj	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
	SZ		RPTINT	CHNE	DAMOD	SAMOD	RPTSEL	MODE

SZ	Selection of transmitted data length
00	8 bits
01	16 bits
10	32-bit
11	Disable the setting

RPTINT	Repeating pattern interrupts allow/disable
0	Interrupts are prohibited.
1	Interrupts are allowed.
When the MODE bit is “0” (normal mode), the RPTINT bit is not set.	

CHNE	Enable/disable chain transfers
0	Chain transfers are disabled.
1	Chain transfers are enabled,
The CHNE bit of the DMACR23 register must be “0” (chain transfer is prohibited).	

DAMOD	Control of the transfer destination address
0	Fixed
1	Control of the transmitting destination address
When the MODE bit is “1” (repeat pattern) and the RPTSEL bit is “0” (the transfer target is the repeat area), the DAMOD bit is not set.	

SAMOD	Control of the transmitting source address
0	Fixed
1	Increasing
When the MODE bit is “1” (repeat pattern) and the RPTSEL bit is “1” (the delivery source is the repeat region), the SAMOD bit is not set.	

RPTSEL	Selection of repeating areas
0	The delivery target is a repeating area.
1	The delivery source is a repeat.
When the MODE bit is “0” (normal mode), the setting of the RPTSEL bit is invalid.	

MODE	Selection of transfer mode
0	Normal mode
1	Repeat mode

Notice The DMACRj register cannot be accessed via DMA transfers.

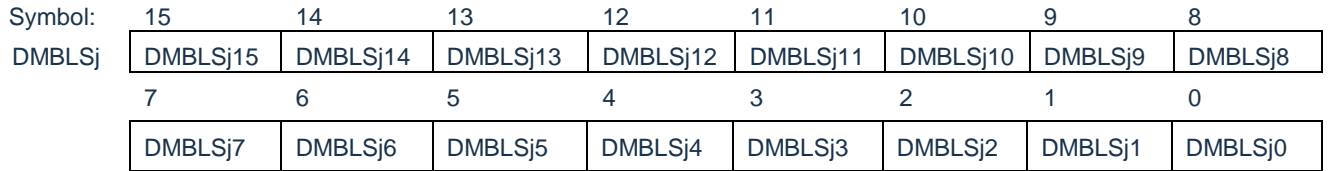
18.3.6 DMA block size register j (DMBLSj) (j=0~23)

This register sets the block size of the 1 initiation transfer of data.

Figure 18-7 Format of DMA block size register j (DMBLSj)

Address: Refer to “18.3.2 Control data allocation”.

After reset: Undefined value R/W



DMBLSj	Transfer block size		
	8-bit transfer	16-bit transfer	32-bit transfer
00H	Disable the setting	Disable the setting	Disable the setting
01H	1 byte	2 bytes	4 bytes
02H	2 bytes	4 bytes	8 bytes
03H	3 bytes	6 bytes	12 bytes
•	•	•	•
•	•	•	•
•	•	•	•
FDH	253 bytes	506 bytes	1012 bytes
FEH	254 bytes	508 bytes	1016 bytes
FFH	255 bytes	510 bytes	1020 bytes
•	•	•	•
•	•	•	•
•	•	•	•
FFFFH	65535 bytes	131070 bytes	262140 bytes

Notice 1. The DMBLSj register cannot be accessed via DMA transfers.

18.3.7 DMA transfer count register j(DMACTj) (j=0~23)

This register sets the number of data transfers to the DMA. Decrements 1 for every DMA transfer started.

Figure 18-8 Format of DMA transfer count register j (DMACTj)

Symbol:	15	14	13	12	11	10	9	8
DMACTj	DMACTj15	DMACTj14	DMACTj13	DMACTj12	DMACTj11	DMACTj10	DMACTj9	DMACTj8
	7	6	5	4	3	2	1	0
	DMACTj7	DMACTj6	DMACTj5	DMACTj4	DMACTj3	DMACTj2	DMACTj1	DMACTj0

Address: Refer to “18. 3. 2 Control data allocation”.

After reset: Undefined value R/W

DMACTj	Number of transfers
00H	Disable settings
01H	1 time
02H	2 times
03H	3 times
⋮	⋮
FDH	253 times
FEH	254 times
FFH	255 times
⋮	⋮
FFFFH	65535 times

Notice 1. The DMACTj register cannot be accessed via DMA transfers.

18.3.8 DMA transfer count reload register j(DMRLDj) (j=0~23)

This register sets the initial value of the number of transfers register in repeat mode. In repeat mode, because the value of this register is reloaded into the DMACT register, the set value must be the same as the initial value of the DMACT register.

Figure 18-9 Format of DMA transfer count reload register j (DMRLDj)

Address: Refer to “18. 3. 2 Control data allocation”.

After reset: Undefined value R/W

Symbol:	15	14	13	12	11	10	9	8
DMRLDj	DMRLDj15	DMRLDj14	DMRLDj13	DMRLDj12	DMRLDj11	DMRLDj10	DMRLDj9	DMRLDj8
	7	6	5	4	3	2	1	0
	DMRLDj7	DMRLDj6	DMRLDj5	DMRLDj4	DMRLDj3	DMRLDj2	DMRLDj1	DMRLDj0

Notice 1. The DMRLDj register cannot be accessed via DMA transfers.

18.3.9 DMA source address register j(DMSARj) (j=0~23)

This register specifies the source address at which data is transferred.

When the SZ bit of the DMACRj register is “01” (16 bits transferred), the lowest bit is ignored and processed as an even address.

When the SZ bit of the DMACRj register is “10” (32-bit transfer), the lower 2 bits are ignored and processed as a word address.

Figure 18-10 Format of DMA source address register j (DMSARj)

Address: Refer to “18. 3. 2 Control data allocation”.

After reset: Undefined value R/W

Symbol	31	30	29	28	27	26	25	24
DMSARj	DMSARj3 1	DMSARj3 0	DMSARj2 9	DMSARj2 8	DMSARj2 7	DMSARj2 6	DMSARj2 5	DMSARj2 4
	23	22	21	20	19	18	17	16
	DMSARj2 3	DMSARj2 2	DMSARj2 1	DMSARj2 0	DMSARj1 9	DMSARj1 8	DMSARj1 7	DMSARj1 6
	15	14	13	12	11	10	9	8
DMSARj1 5	DMSARj1 4	DMSARj1 3	DMSARj1 2	DMSARj1 1	DMSARj1 0	DMSARj9	DMSARj8	
7	6	5	4	3	2	1	0	
DMSARj7	DMSARj6	DMSARj5	DMSARj4	DMSARj3	DMSARj2	DMSARj1	DMSARj0	

Notice1. The DMSARj register cannot be accessed via DMA transfers.

18.3.10 DMA destination address register j(DMDARj) (j=0~23)

This register specifies the destination address at which data is transferred.

When the SZ bit of the DMACRj register is “01” (16 bits transferred), the lowest bit is ignored and processed as an even address.

When the SZ bit of the DMACRj register is “10” (32-bit transfer), the lower 2 bits are ignored and processed as a word address.

Figure 18-11 Format of DMA destination address register j (DMDARj)

Address: Refer to “18. 3. 2 Control data allocation”.

After reset: Undefined value R/W

Symbol	31	30	29	28	27	26	25	24
DMDARj	DMDARj3 1	DMDARj3 0	DMDARj2 9	DMDARj2 8	DMDARj2 7	DMDARj2 6	DMDARj2 5	DMDARj2 4
	23	22	21	20	19	18	17	16
	DMDARj2 3	DMDARj2 2	DMDARj2 1	DMDARj2 0	DMDARj1 9	DMDARj1 8	DMDARj1 7	DMDARj1 6
	15	14	13	12	11	10	9	8
DMDARj1 5	DMDARj1 4	DMDARj1 3	DMDARj1 2	DMDARj1 1	DMDARj1 0	DMDARj9	DMDARj8	
7	6	5	4	3	2	1	0	
DMDARj7	DMDARj6	DMDARj5	DMDARj4	DMDARj3	DMDARj2	DMDARj1	DMDARj0	

Notice: The DMDARj register cannot be accessed via DMA transfers.

18.3.11 DMA boot enable register i (DMAENi) (i=0~2)

This is the 8-bit register that controls the boot of the DMA through each interrupt source. The corresponding connection between the interrupt source and the DMAENi0~DMAENi7 bits is shown in Table 18-6.

The DMAENi register can be set via 8-bit memory operation instructions.

- Notice1. The DMAENi0~DMAENi7 bits must be changed at the boot source that does not produce the corresponding bits.
2. The DMAENi register cannot be accessed via DMA transfers.
 3. The assigned function varies from product to product, and the bits without the assigned function must be set to "0".

Figure 18-12 Format of boot enable register i (DMAENi) (i=0~2)

Address: 40005000H (DMAEN0), 40005001H (DMAEN1),
 40005002H (DMAEN2) After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
DMAENi	DMAENj7	DMAENj6	DMAENj5	DMAENj4	DMAENj3	DMAENj2	DMAENj1	DMAENj0

DMAENi7	DMA boot enable i7
0	Disables boot.
1	Enables boot.
Depending on the condition under which the end-of-transmission interrupt occurs, the DMAENi7 bit becomes "0" (disable boot).	

DMAENi6	DMA boot enable i6
0	Disables boot.
1	Enables boot.
Depending on the conditions under which the end-of-transmission interrupt occurs, the DMAENi6 bit becomes "0" (disable boot).	

DMAENi5	DMA boot enable i5
0	Disables boot.
1	Enables boot.
Depending on the condition under which the end-of-transmission interrupt occurs, the DMAENi5 bit becomes "0" (disable boot).	

DMAENi4	DMA boot enable i4
0	Disables boot.
1	Enables boot.
Depending on the conditions under which the end-of-transmission interrupt occurs, the DMAENi4 bit becomes "0" (disable boot).	

DMAENi3	DMA boot enable i3
0	Disables boot.
1	Enables boot.
Depending on the condition under which the end-of-transmission interrupt occurs, the DMAENi3 bit becomes "0" (disable boot).	

DMAENi2	DMA boot enable i2
0	Disables boot.
1	Enables boot.
Depending on the condition under which the end-of-transmission interrupt occurs, the DMAENi2 bit becomes "0" (disable boot).	

DMAENi1	DMA boot enable i1
0	Disables boot.
1	Enables boot.
Depending on the conditions under which the end-of-transmission interrupt occurs, the DMAENi1 bit becomes "0" (disable boot).	

DMAENi0	DMA boot enable i0
0	Disables boot.
1	Enables boot.
Depending on the condition under which the end-of-transmission interrupt occurs, the DMAENi0 bit becomes "0" (disable boot).	

Table 18-6 Interrupt sources correspond to DMAENi0~DMAENi7 bits

register	DMAENi 7 bits	DMAENi 6 bits	DMAENi 5 bits	DMAENi 4 bit	DMAENi 3 bits	DMAENi 2 bits	DMAENi 1 bit	DMAENi0 bit
DMAEN0	Comparator detect 1	Comparator detect 0	The A/D conversion ends	INTP3	INTP2	INTP1	INTP0	Flash erase/write ends
DMAEN1	15-bit interval timer interrupt	IICA0 communication ends	The end of the UART2 transmission / the end of the SSPI20 transmission or the end of the buffer null/IIC20 transmission	The end of transmission received by UART2 / the end of transmission of SSPI21 or the end of transmission of buffer NULL/IIC21	End of transmission for UART1 transmission/end of transmission for SSPI10 or end of transmission for buffer NULL/IIC10/end of transmission for SPI	The end of transmission received by UART1 / the end of transmission of SSPI11 or the end of transmission of buffer NULL/IIC11	The end of the UART0 transmission / the end of the SSPI00 transmission or the end of the buffer NULL/IIC00 transmission	The end of transmission received by UART0 / the end of transmission of SSPI01 or the end of transmission of buffer NULL/IIC01
DMAEN2	The counting end of channel 3 of the timer array unit 1 ends or the capture ends	The counting end of channel 2 of the timer array unit 1 ends or the capture ends	The counting end of channel 1 of timer array unit 1 or the end of the snap	The counting end of channel 0 of timer array unit 1 ends or the snap ends	The counting end of channel 3 of the timer array unit 0 or the end of the snap	The counting end of channel 2 of timer array unit 0 ends or the snap ends	The counting end of channel 1 of the timer array unit 0 ends or the capture ends	The counting end of channel 0 of the timer array unit 0 ends or the snap ends

Notice Bits with no assigned function must be set to "0".

Remark i=0~2

18.3.12 DMA base address register (DMABAR)

This is a 32-bit register that sets the vector address that holds the start address of the DMA control data area and the address of the DMA control data area.

- Notice 1. The DMABAR register must be changed with all DMA boot sources set to a state that disables startup.
2. DMABAR registers can only be rewritten once.
 3. DMABAR register access is not possible via DMA transfer.
 4. For the allocation of DMA control data area and DMA vector table area, please refer to the note “18.3.1 DMA control data areas and DMA vector table areas allocation”.
 5. Set the register to keep 512-byte aligned, i.e. the low 8 bits set to zero. DMA hardware ignores the low 8 bits.
 6. This register can only be accessed by WORD, ignored by BYTE and HALWORD access.

Figure 18-13 Format of DMA base address register (DMABAR)

Address: 40005008H After reset: 00000000H R/W

Symbol	31	30	29	28	27	26	25	24
DMABA Rj	DMABARj	DMABARj	DMABARj	DMABARj	DMABARj	DMABARj	DMABARj	DMABARj
	31	30	29	28	27	26	25	24
	23	22	21	20	19	18	17	16
	DMABARj	DMABARj	DMABARj	DMABARj	DMABARj	DMABARj	DMABARj	DMABARj
	23	22	21	20	19	18	17	16
	15	14	13	12	11	10	9	8
	DMABARj	DMABARj	DMABARj	DMABARj	DMABARj	DMABARj	DMABARj	DMABARj
	15	14	13	12	11	10	9	8
7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	

18.4 DMA operation

Once the DMA is started, the control data is read from the DMA control data area, the data is transmitted according to this control data, and the control data after the data transmission is written back to the DMA control data area. It can save 24 groups of control data to the DMA control data area and transfer 24 groups of data. There are normal and repeat modes in the transfer mode, and the transfer sizes are 8-bit transfer, 16-bit transfer, and 32-bit transfer. When the CHNE bit of the DMACRj (j=0 to 23) register is "1" (chain transmission is allowed), continuous data transmission (chain transmission) is performed by reading multiple control data through 1 boot source.

The transmit source address and the transmit destination address are specified through the 32-bit DMSARj register and the 32-bit DMDARj register, respectively. After data transfer, the values of the DMSARj register and the DMDARj register are incremented or fixed according to the control data.

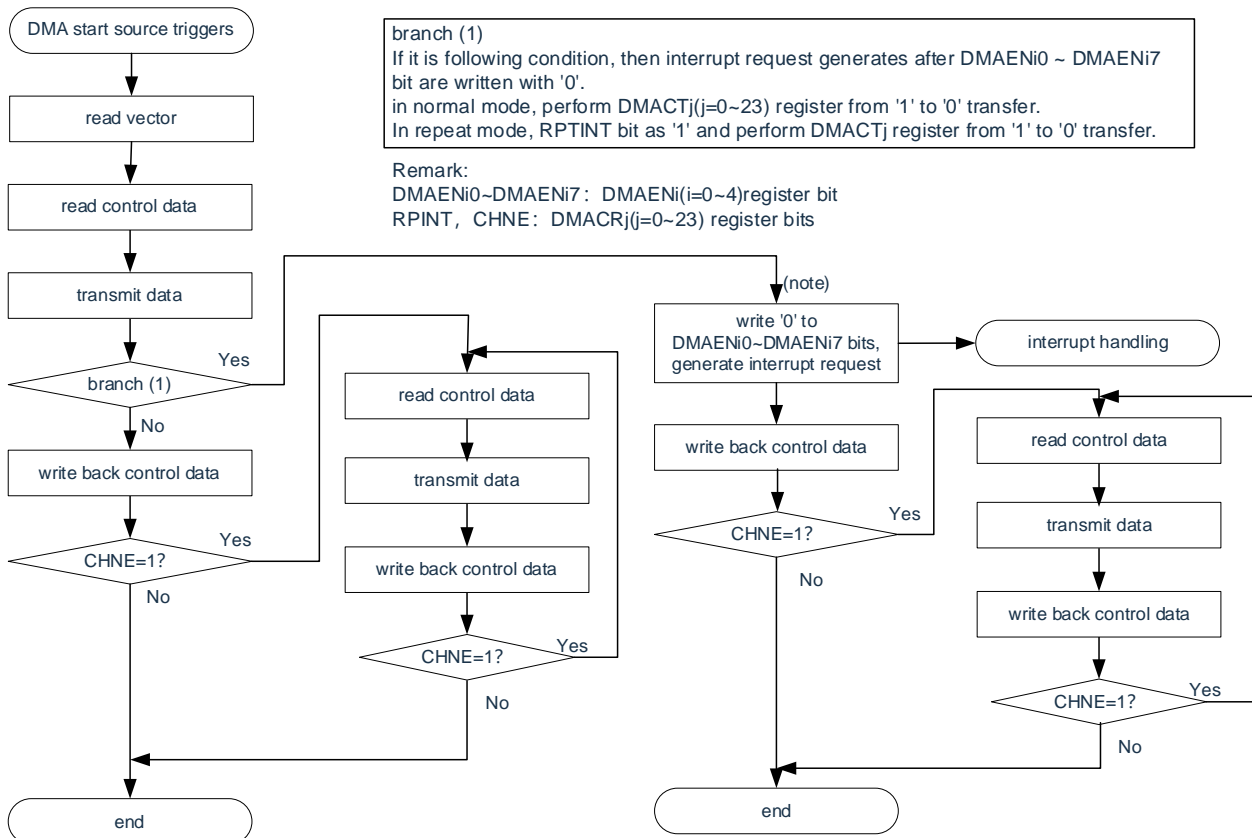
18.4.1 Boot source

The DMA is initiated by the interrupt signal of the peripheral function, and the interrupt signal to start the DMA is selected through the DMAENi (i=0~2) register. When the data transmission (in the case of chain transmission, continuous initial transmission) is set to the DMAENi0~DMAENi7 bits of the corresponding DMAENi register in the DMA operation "0" (disables startup).

- In normal mode, a DMACTj (j=0~23) register is transferred to "0".
- In repeat mode, the RPTINT bit of the DMACRj register is "1" (interrupts are enabled) and the DMACTj register is transferred to "0".

The internal operation flowchart of DMA is shown in Figure 18-14.

Figure 18-14 Flowchart of DMA internal operation



Note: In data transfers initiated through the setting of Enable Chain Transfer (CHNE=1), DMAENi0~DMAENi7 bits are not written "0" and no interrupt requests are generated.

18.4.2 Normal mode

In the case of 8-bit transmission, the transmission data for 1 start is 1 to 65535 bytes; in the case of 16-bit transmission, the transmission data for 1 start is 2 to 131070 bytes; in the case of 32-bit transmission, the transmission data for 1 start is 4 to 262140 bytes. The number of transmissions is 1 to 65535 times. If the DMACTj (j=0 to 23) register becomes "0", the interrupt request corresponding to the start-up source is generated to the interrupt controller during DMA operation, and the DMAENi0 to DMAENi7 bits of the corresponding DMAENi (i=0 to 2) register are set to "0" (disable start-up).

The register function and data transfer in normal mode are shown in Table 18-7 and Figure 18-15.

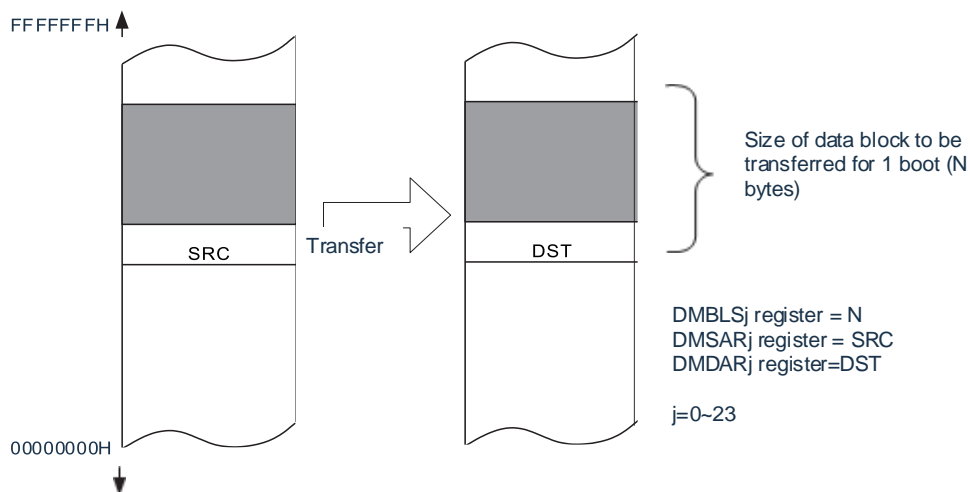
Table 18-7 Register function in normal mde

Register name	Symbol	Function
DMA block size register j	DMBLSj	The size of the data block to be transferred by 1 start
DMA transfer count register j	DMACTj	The number of times the data was transmitted
DMA transfer number of times to reload register j	DMRLDj	Not used ^{Note} .
DMA source address register j	DMSARj	The address of the source of the data
DMA destination address register j	DMDARj	The destination address of the data

Note When parity error reset (RPERDIS=0) is allowed by RAM parity error detection function, initialization (00H) must be performed.

Remark j=0~23

Figure 18-15 Data transfer in normal mode



Setting of the DMACR register				Control of the source address	Control of the destination address	The source address after transfer	Destination address after transfer
DAMOD	SAMOD	RPTSEL	MODE				
0	0	X	0	Fixed	Fixed	SRC	DST
0	1	X	0	Increasing	Fixed	SRC+N	DST
1	0	X	0	Fixed	Increasing	SRC	DST+N
1	1	X	0	Increasing	Increasing	SRC+N	DST+N

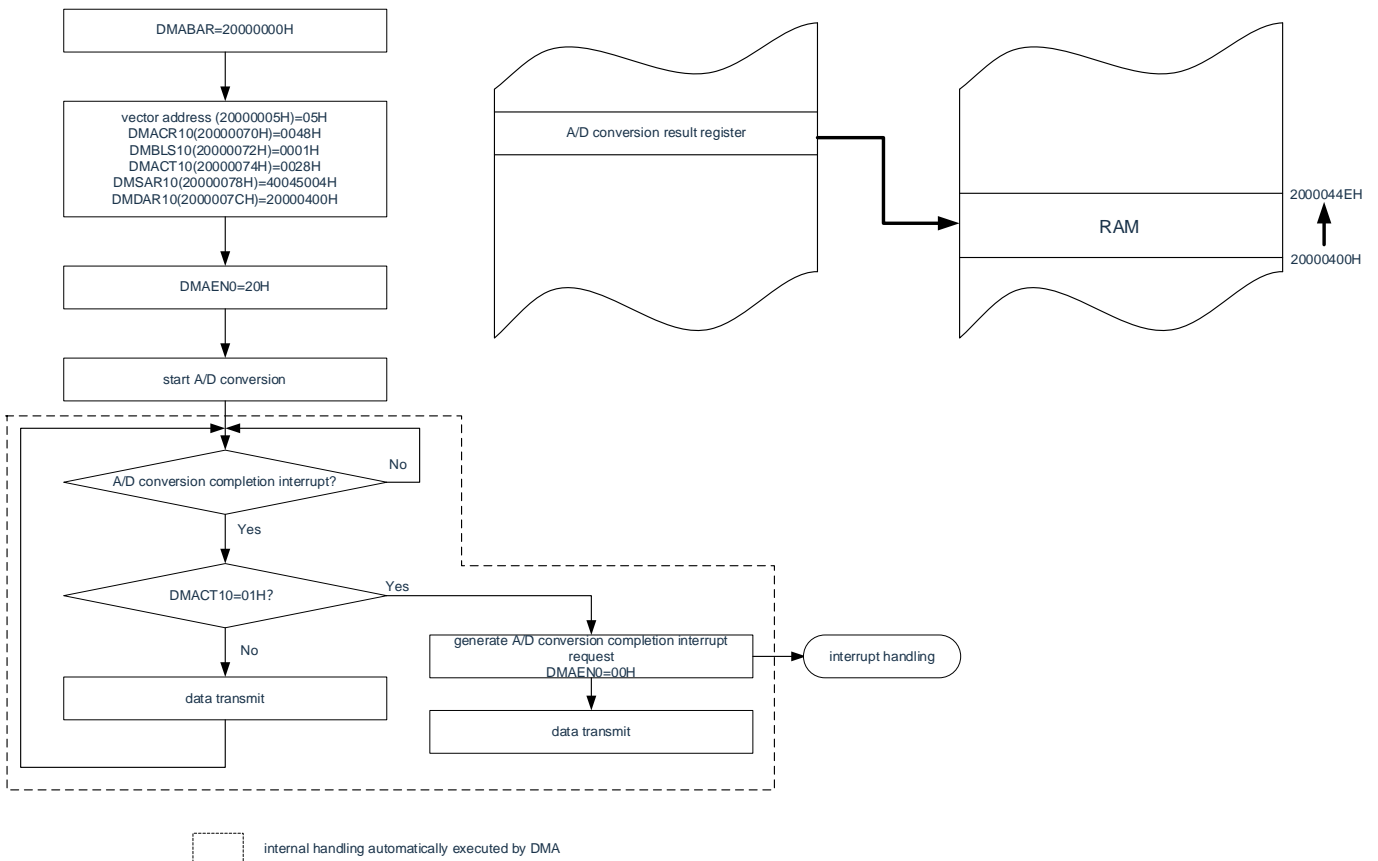
X: "0" or "1"

(1) Example 1 of normal mode usage: continuous A/D conversion results

DMA is started by an A/D conversion end interrupt, and the value of the A/D conversion result register is transferred to RAM.

- The vector address is allocated at 200,00005H, and the control data is distributed at 20000070 H~2000007FH.
- Transfer 2 bytes of data from the A/D conversion result registers (40045004H, 40045005H) 40 times to 20000400H~of RAM 2000044FH 80 bytes.

Figure 18-16 Normal mode usage example 1: Continuously take the A/D conversion result



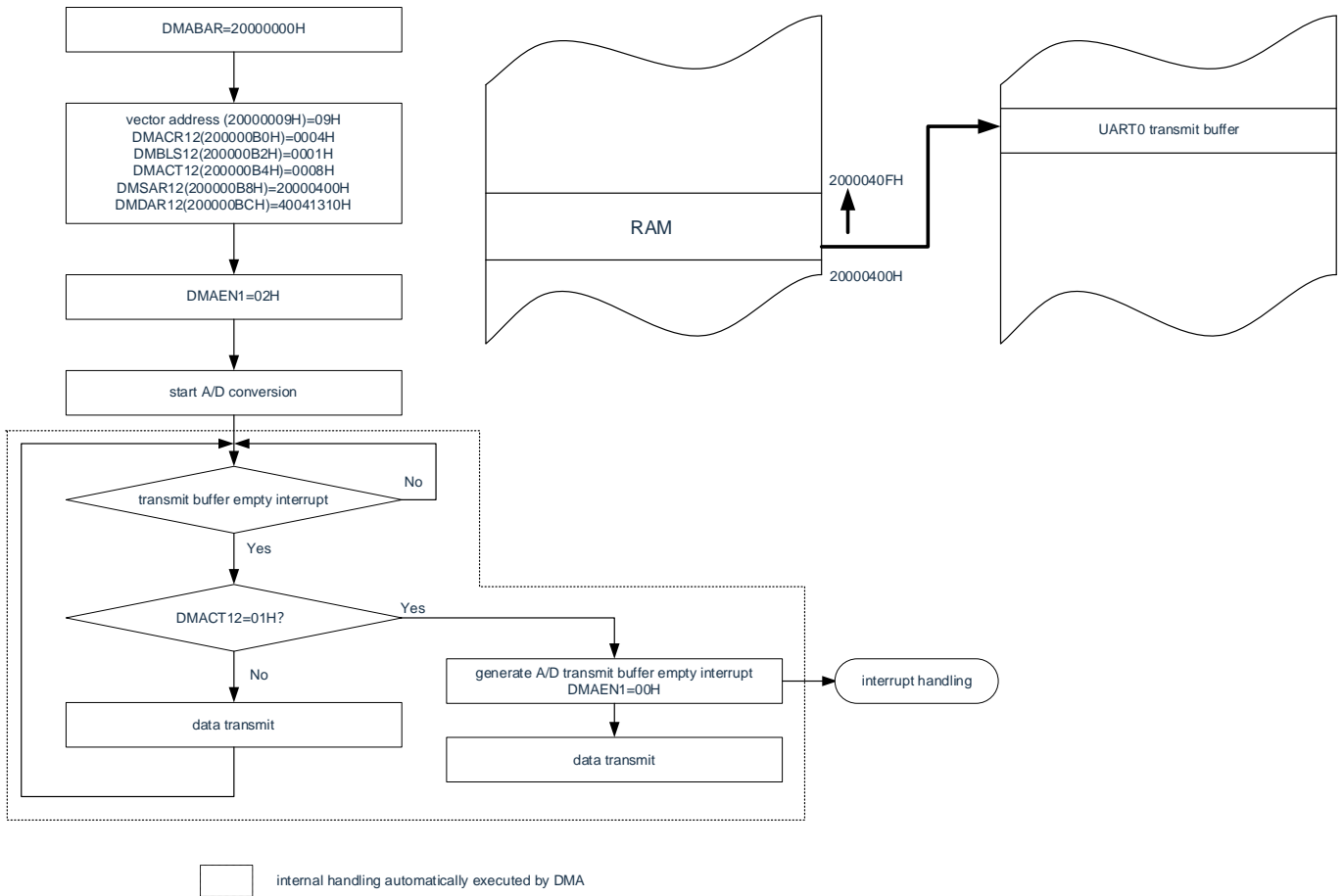
Because it is in normal mode, the value of the DMRLD10 register is not used. However, when parity error reset (RPERDIS=0) is allowed to occur via the RAM parity error detection function, the DMRLD10 register must be initialized (0000H).

(2) Example 2 of normal mode usage: UART0 transmits continuously

DMA is started through a blank interrupt from UART0's send buffer, and the value of RAM is transferred to UART0's send buffer.

- The vector address is allocated at 20000009H, and the control data is allocated at 200000B0H~200000BFH.
- Transfer 8 bytes of RAM 20000400H~20000407H to UART0's send buffer (40041310H).

Figure 18-17 Normal mode usage example 2: UART0 transmits continuously



Because it is in normal mode, the value of the DMRLD12 register is not used. However, when parity error reset (RPERDIS=0) is allowed through the RAM parity error detection function, the DMRLD12 register must be initialized (0000H).

The first UART0 send must be started through the software. Start DMA with an empty interrupt from the send buffer and then automatically send after the second time.

18.4.3 Repeat mode

The transfer data for one initiation is 1 to 65535 bytes. The source or destination is designated as a repeat area, and the number of transfers is 1 to 65535 times. Once the specified number of transfers is complete, initialize the DMACTj(j=0~23) register and the address specified as a repeat, and then repeat the transfer. This is when the RPTINT bit of the DMACRj register is “1” (interrupts are allowed) and a data transfer is made where the DMACTj register becomes “0” DMA generates an interrupt request for the corresponding start source to the interrupt controller during operation, and the DMAENi0~DMAENi7 of the corresponding DMAENi (i=0~2) registers Position “0” (disable startup). When the RPTINT bit of the DMACRj register is “0” (interrupt is prohibited), even if the DMACTj register becomes “0” data transfer, no interrupt requests are generated, and the DMAENi0~DMAENi7 bits are unchanged from “0”.

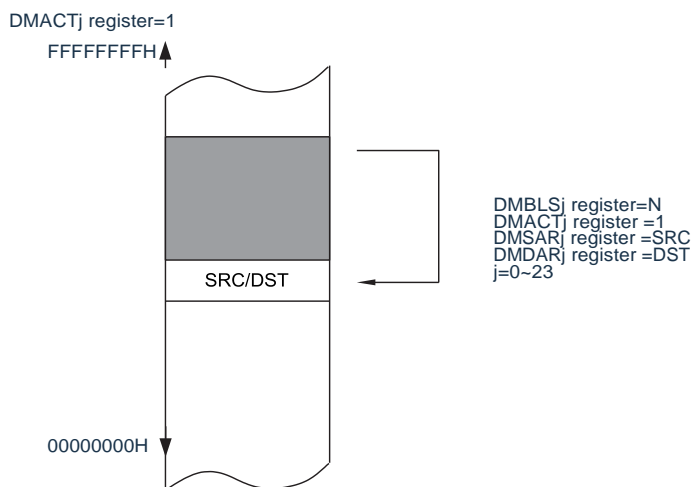
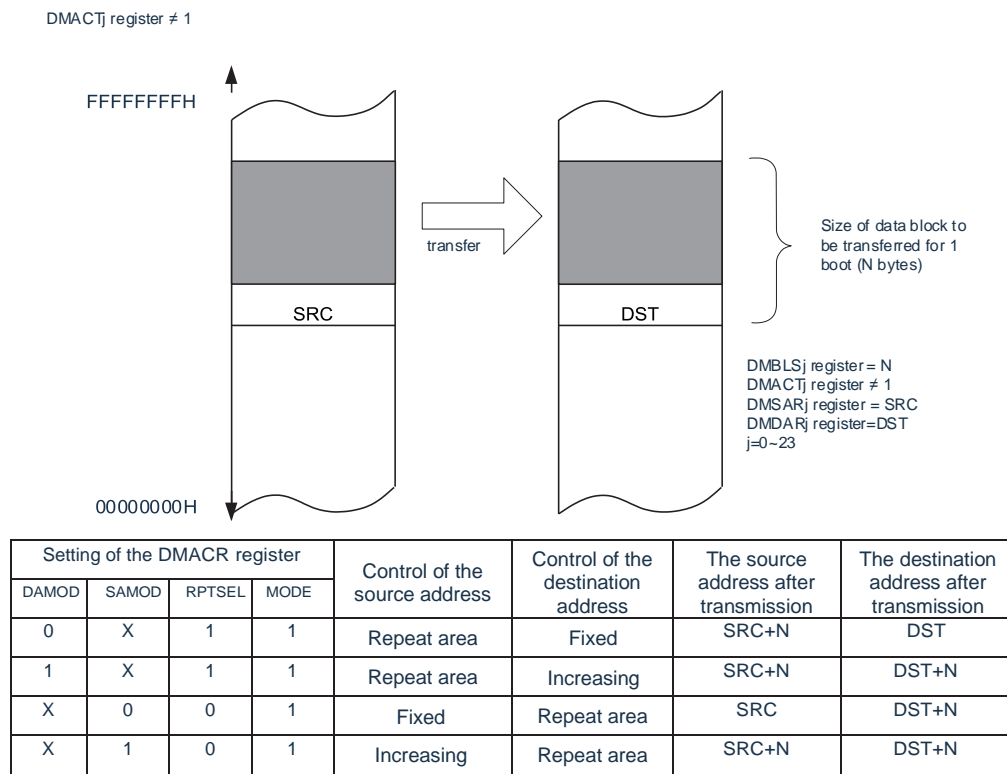
The register function and data transfer of the repeating pattern are shown in Table 18-8 Figure 18-18, respectively.

Table 18-8 Register functions in repeat mode

Register name	Symbol	Function
DMA block size register j	DMBLSj	The size of the data block to be transferred by 1 start
DMA transfer count register j	DMACTj	The number of times the data was transmitted
DMA transfer number of times to reload register j	DMRLDj	Reload the value of this register into the DMACT register. (Initialize the number of data transfers)
DMA source address register j	DMSARj	The address of the source of the data
DMA destination address register j	DMDARj	The destination address of the data

Remark j=0~23

Figure 18-18 Data transfer in repeat mode



Setting of the DMACR register				Control of the source address	Control of the destination address	The source address after transmission	The destination address after transmission
DAMOD	SAMOD	RPTSEL	MODE				
0	X	1	1	Repeat area	Fixed	SRC	DST
1	X	1	1	Repeat area	Increasing	SRC	DST+N
X	0	0	1	Fixed	Repeat area	SRC	DST
X	1	0	1	Increasing	Repeat area	SRC+N	DST

X: "0" or "1"

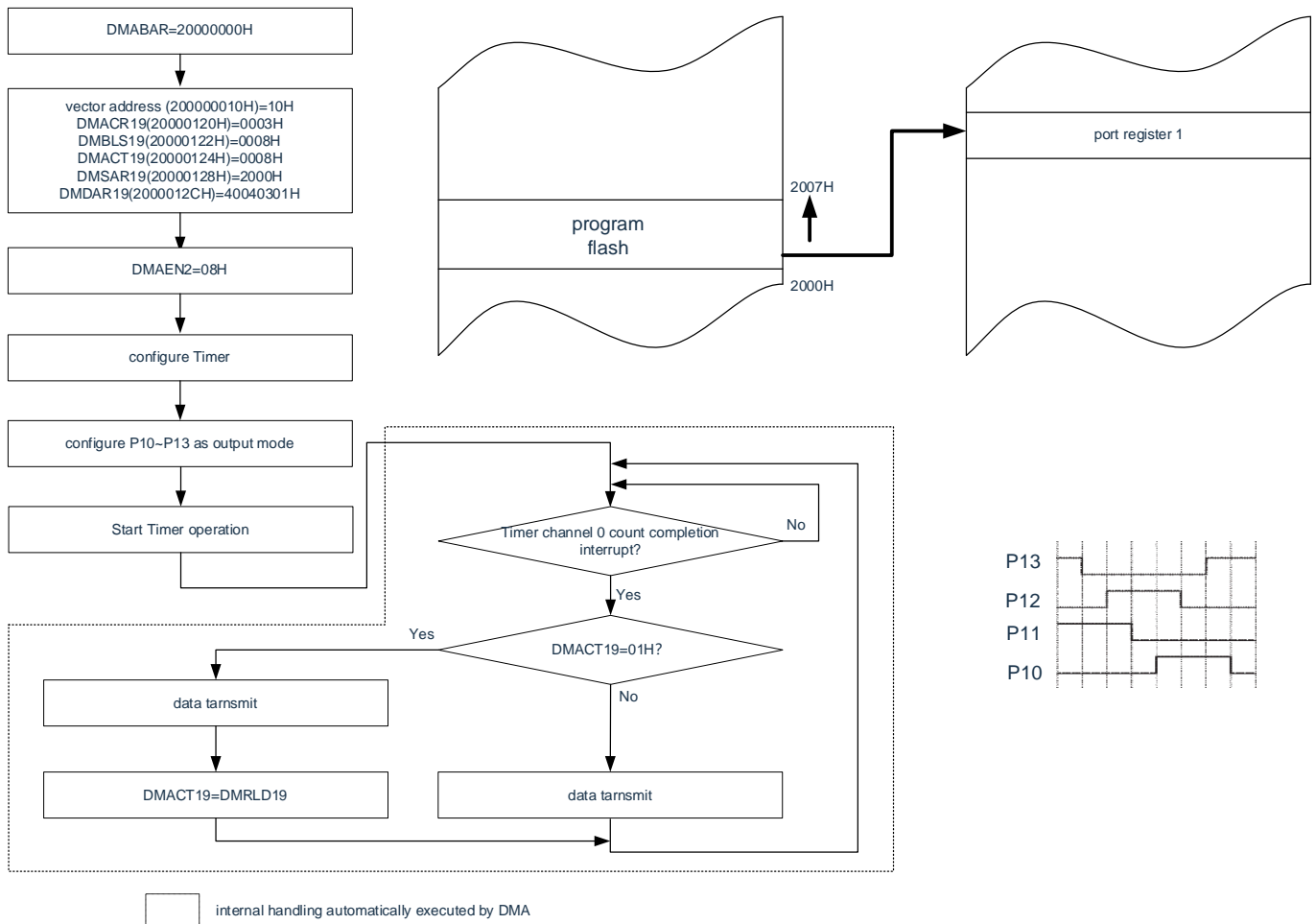
Notice1. When using repeat mode, the data length of the repeat must be set to less than 65535 bytes.

(1) Example of repeat mode usage: use the stepper motor of the port to control the pulse output

The DMA is started using the Channel 0 interval timer function of the Timer40, and the mode of the motor control pulse saved in the code flash memory is transferred to the universal port.

- The vector address is allocated at 20000010H, and the control data is allocated at 20000120H~2000012PH.
- Transfer 8 bytes of code flash 02000H~02007H to port register 1 (40040301H).
- Disable repeat mode interruption.

Figure 18-19 Example of repeat mode usage: A stepper motor using a port is used to control the pulse output



To stop the output, bit0 of DMAEN2 must be cleared after stopping the operation of the timer.

18.4.4 Chain transfer

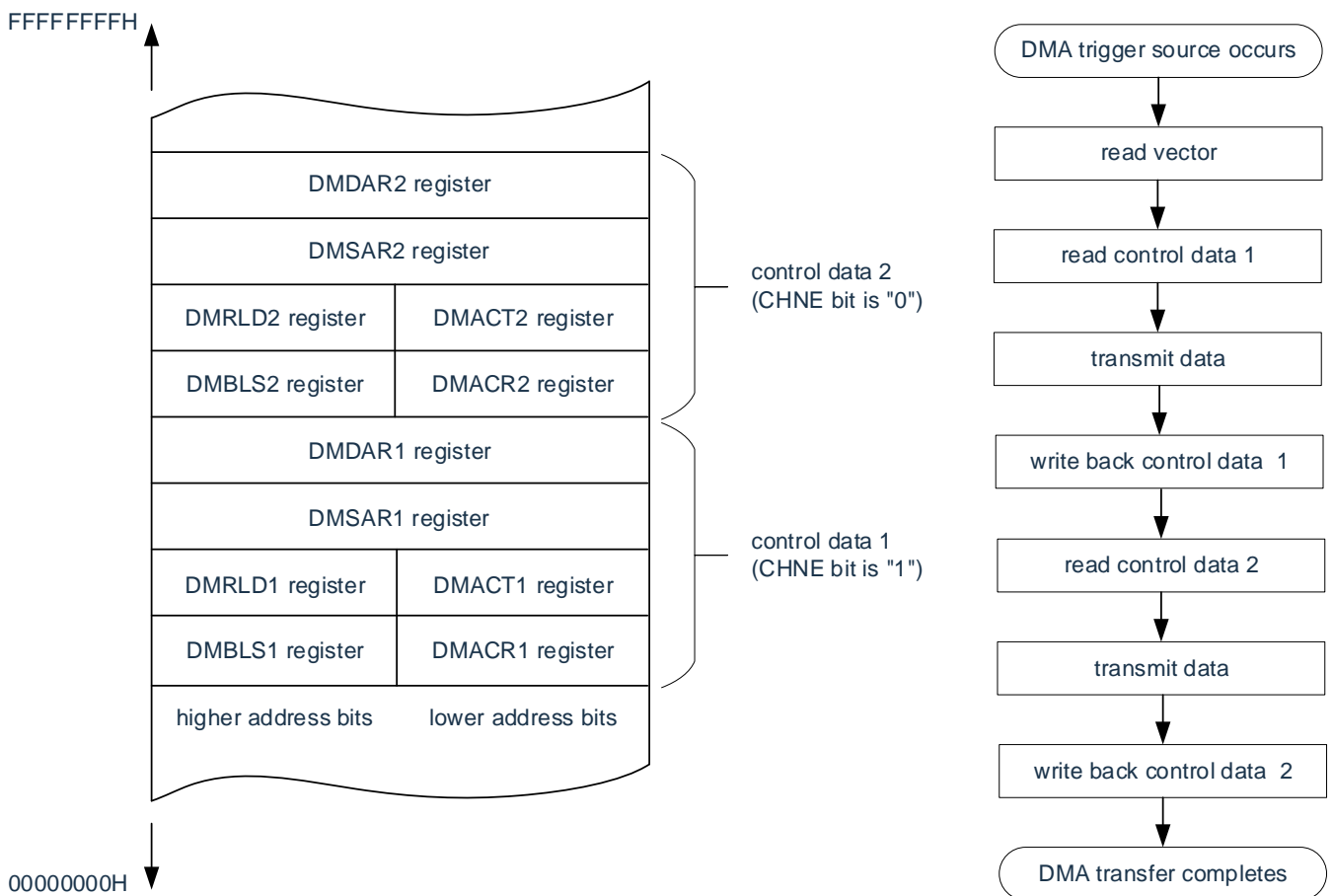
When the CHNE bit of the DMACRj(j=0~23) register is “1” (allow chain transfer), multiple data can be transferred continuously through one startup source.

Once the DMA is started, the control data is selected by reading the data from the corresponding vector address of the startup source, and the control data assigned to the DMA control data area is read. If the CHNE bit of the read control data is “1” (allowing chain transfer), the transfer continues after the transfer is completed by reading the next assigned control data. Repeat this operation until the control data transfer with the CHNE bit “0” (disable chain transmission) ends.

When multiple control data are used for chain transfer, the number of transmissions set by the first control data is valid, while the number of transmissions of the control data processed after the second is invalid.

The flowchart of chain transfer is shown in Figure 18-20.

Figure 18-20 Flow chart of chain transfers



Notice1. CHNE bit of the DMACR23 register must be “0” (chain transfer is prohibited).

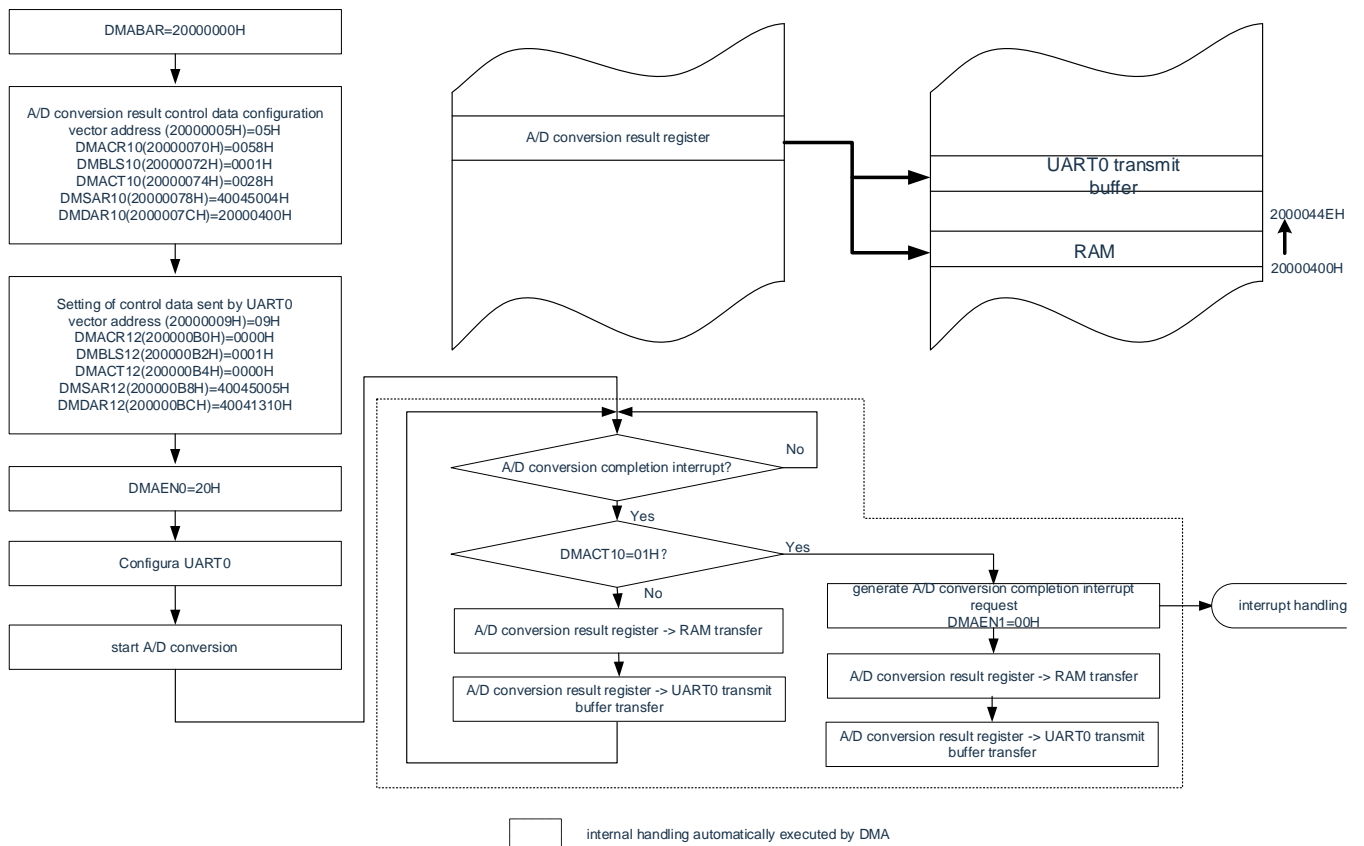
2. In the data transfer after the second time of the chain transfer, the bits DMAENi0~DMAENi7 of DMAENi (i=0~2) register does not change to “0” (DMA is prohibited from starting) and no interrupt requests are generated.

(1) Example of chain transfer usage: continuous A/D conversion result for UART0 transmission

DMA is started by interrupting the end of the A/D conversion, and the A/D conversion result is transferred to RAM for UART0 transmission.

- The vector addresses are 20000005 H and 20000009H, respectively.
- Control data distribution of A/D conversion results is 20000070H~2000007FH.
- The control data sent by UART0 is distributed between 200000B0 H and 200000BFH.
- Transfer 2 bytes of data from the A/D conversion result registers (40045004H, 40045005H) to 20000400H~2000044FH of RAM, and the high bit 1 byte (40045005H) of the A/D conversion result register is transferred to the send buffer (40041310 of UART 0 H).

Figure 18-21 Example of chain transfer usage: Continuous A/D conversion results are used for UART0 transmission



18.5 Cautions on using DMA

18.5.1 DMA control data and vector table settings

- The DMA Base Address Register (DMABAR) must be changed with all DMA boot sources set to a state that disables startup.
- DMA Base Address Register (DMABAR) can only be overridden once.
- The DMAENi0~DMAENi 7 bits must be “0” in the corresponding DMAENi (i=0~2) registers (DMA is prohibited Startup) when changing DMACRj, DMBSLsj, DMACTj, DMRLDj, DMSARj, Data for the DMDARj register.
- The DMAENi0~DMAENi 7 bits must be “0” in the corresponding DMAENi (i=0~2) registers (DMA is prohibited Start) when setting the starting address of the DMA control data area in the vector table.

18.5.2 DMA control data area and DMA vector table area allocation

The areas in which DMA control data and vector tables can be assigned vary depending on the product and usage conditions.

- The stack area, DMA control data area, and DMA vector table area cannot overlap.
- When parity error reset (RPERDIS=0) is allowed to occur via RAM parity error detection function, the DMRLD register must be initialized even when using normal mode (0 000H).

18.5.3 Number of execution clocks for DMA

The execution and number of clocks required at DMA boot are shown in Table 18-9.

Table 18-9 Execution and number of clocks required when DMA is started

Read vector	Control data		Read data	Write data
	Read	Write back		
1	4	Note 1	Note 2	Note 2

Note 1. For the number of clocks required to write back control data, refer to Table 18-10 Number of clocks required to write back control data

2. For the number of clocks required to read and write data, please refer to Table 18-11 Number of clocks required to read and write data

Table 18-10 Number of clocks required to write back control data

Setting of the DMACR register				Address settings		Control register write back				Clock number
DAMOD	SAMOD	RPTSEL	MODE	Source	Target	DMACTj register	DMRLDj register	DMSARj register	DMDARj register	
0	0	X	0	Fixed	Fixed	Write back	Write back	No write back	No write back	1
0	1	X	0	Increase	Fixed	Write back	Write back	Write back	No write back	2
1	0	X	0	Fixed	Increase	Write back	Write back	No write back	Write back	2
1	1	X	0	Increase	Increase	Write back	Write back	Write back	Write back	3
0	X	1	1	Repeat area	Fixed	Write back	Write back	Write back	No write back	2
1	X	1	1		Increase	Write back	Write back	Write back	Write back	3
X	0	0	1	Fixed	Repeat area	Write back	Write back	No write back	Write back	2
X	1	0	1	Increase		Write back	Write back	Write back	Write back	3

Remark j=0~23, X: "0" or "1"

Table 18-11 Number of clocks required to read and write data

Execution status	RAM	Code flash	Data flash	Special function registers (SFR)	Extended Special Function Register (2ndSFR)	
					No waiting	Wait
Read data	1	2	4	1	1	1+ wait number ^{Note}
Write data	1	—	—	1	1	1+ wait number ^{Note}

18.5.4 DMA response time

The DMA response time is shown in Table 18-12. DMA response time refers to the time from the time the DMA boot source is detected to the start of the DMA transfer, excluding the number of execution clocks for the DMA.

Table 18-12 Response time for DMA

	Minimum time	Maximum time
Response time	3 clocks	23 clocks

However, the response of the DMA may also be delayed in the following cases. The number of clocks delayed varies depending on the condition.

- When executing instructions from internal RAM
Maximum response time: 20 clocks

Remark 1 clock: $1/f_{CLK}$ (f_{CLK} : CPU/peripheral hardware clock)

18.5.5 DMA boot source

- It is not possible to input the same boot source during the period from the input of the DMA boot source to the end of the DMA transfer.
 - The DMA boot enable bit corresponding to the boot source cannot be operated at the location where the DMA boot source is generated.
 - If the DMA boot source sends contention, the CPU determines the boot source by judging the priority when it accepts the DMA transfer. Refer to “18. 3. 3 Vector table” for the priority of the boot source.
 - If DMA start is allowed in one of the following states, a DMA transfer is started and an interrupt is generated at the end of the transfer. Therefore, the monitor flag (CnMON) of the comparator must be set to allow DMA boot as necessary.
 - Set to generate an interrupt request via single edge detection of the comparator ^{Note} (CnEDG=0) and an interrupt request via the rising edge of the comparator (CnEPO=0) and $IVCMP > IVREF$ (or the internal reference voltage is 1.45V).
 - Set to generate an interrupt request via single edge detection of the comparator (CnEDG=0) and an interrupt request via the falling edge of the comparator (CnEPO=1) and $IVCMP < IVREF$ (or the internal reference voltage is 1.45V).
- (n=0, 1)

18.5.6 Operation in standby mode

Status	DMA operation
Sleep mode	Can be operated (disable operation in low-power RTC mode).
Deep sleep mode	Can accept the DMA start source and make DMA transfer <small>Note 1</small>

Note 1. In deep sleep mode, DMA transmission can be performed after the DMA startup source is detected, and the deep sleep mode can be returned after the transfer is completed. However, because the code flash and data flash stop running in deep sleep mode, you cannot set flash as transfer source.

Chapter 19 Linkage Controller (EVENTC)

19.1 Function of EVENTC

EVENTC links the events output by each peripheral function to each other between the peripheral functions. It can be operated directly through the event chain without going through the CPU, and can be operated directly between peripheral functions.

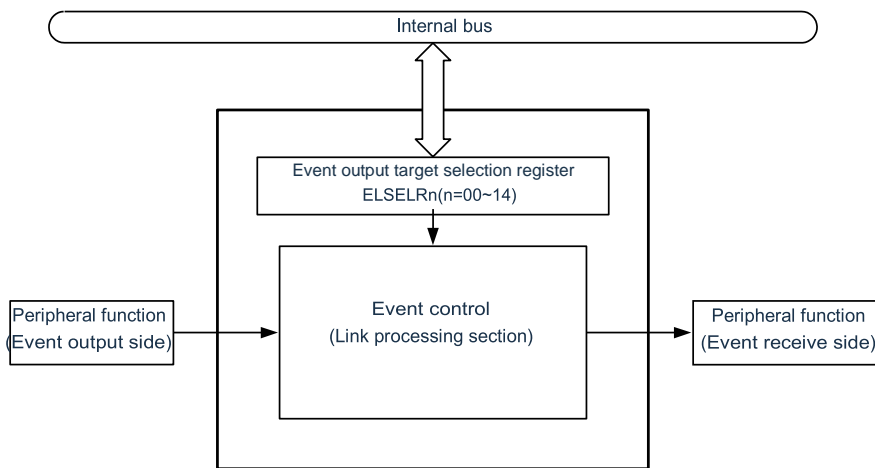
EVENTC has the following features:

- Depending on the product, the event signals of 15 peripheral functions can be directly linked to the specified peripheral functions.
- Depending on the product, the event signal can be used as the startup source for the operation of one of the four peripheral functions.

19.2 Structure of EVENTC

The block diagram of EVENTC is shown in Figure 19-1.

Figure 19-1 Block diagram of EVENTC



19.3 Control registers

The controller registers are shown in Table 19-1.

Table 19-1 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
Event output target selection register 13	ELSELR13
Event output target selection register 14	ELSELR14

19.3.1 Output target selection register n (ELSELRn) (n=00~14)

The ELSELRn register links each event signal to the event receiver peripheral function (link target peripheral function) to run when the event accepts the event. You cannot link multiple event inputs to the same event output destination (event acceptor). Otherwise, the event receiver's peripheral functionality may operate uncertainly and the event signal may not be accepted properly. Also, you cannot set the event link occurrence source and event output destination to the same function.

The ELSELRn register must be set during the period when the peripheral functions of all event outputs do not generate an event signal.

The correspondence between the ELSELRn register (n=00~14) and the peripheral functions is shown in Table 19-2. The corresponding operation between the config value of the ELSELRn register (n=00 ~14) and the link target peripheral function when receiving the event is shown Table 19-3.

Figure 19-2 Format of event output target selection register n (ELSELRn)

Address: 40043400H (ELSELR00)~4004340EH (ELSELR14) 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 linking.
0	0	1	Selection of the linked peripheral function 1 operation ^{Note 1.}
0	1	0	Selection of the linked peripheral function 2 operation ^{Note 1.}
0	1	1	Selection of the linked peripheral function 3 operation ^{Note 1.}
1	0	0	Selection of the linked peripheral function 4 operation ^{Note 1.}
Others			Disable settings

Note 1. Refer to Table 19-3 Correspondence between the setting value of ELSELRn register (n=00~14) and the operation when the link target peripheral function receives the event

Table 19-2 ELSELRn registers (n=00~14) and peripheral functions

Register	Event occurrence source (output source for event input n)	Content
ELSELR00	External interrupt edge detection 0	INTP0
ELSELR01	External interrupt edge detection1	INTP1
ELSELR02	External interrupt edge detection2	INTP2
ELSELR03	External interrupt edge detection3	INTP3
ELSELR04	RTC fixed period/alarm clock consistent 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 counts end/capture end	INTTM12
ELSELR12	Timer41 channel 03 count end/capture end	INTTM13
ELSELR13	Comparator detect 0	INTCMP0
ELSELR14	Comparator detect 1	INTCMP1

Table 19-3 Correspondence between the setting value of ELSELRn register (n=00~14) and the operation when the link target peripheral function receives the event

ELSELRn register ELSELn2~ELSELn0 bits	Link target No.	Link target peripheral function	Operation when the event is received
001B	1	A/D converter	Start the A/D conversion.
010B	2	Timer40 channel 0 Timer input ^{Note 1}	Delay counter, measurement of input pulse interval, external event counter
011B	3	Timer40 channel 1 Timer input ^{Note 2}	Delay counter, measurement of input pulse interval, external event counter
100B	4	The EPWM output controls the truncation source	Forced truncation of the pulse output

Note 1. To select the timer input of Timer40 channel 0 as the link target peripheral function, you must first set the operating clock of channel 0 to fCLK via Timer Clock Select Register 0 (TPS0), set the noise filter of TI00 pin to OFF via Noise Filter Enable Register 1 (NFEN1) (TNFEN00=0), and set the timer input used by channel 0 to the event input signal of the link controller via Timer Input Selection Register 0 (TIS0).

2. To select the timer input of Timer40 channel 1 as the link target peripheral function, you must first set the operating clock of channel 1 to fCLK via Timer Clock Selection Register 0 (TPS0), set the noise filter of 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).

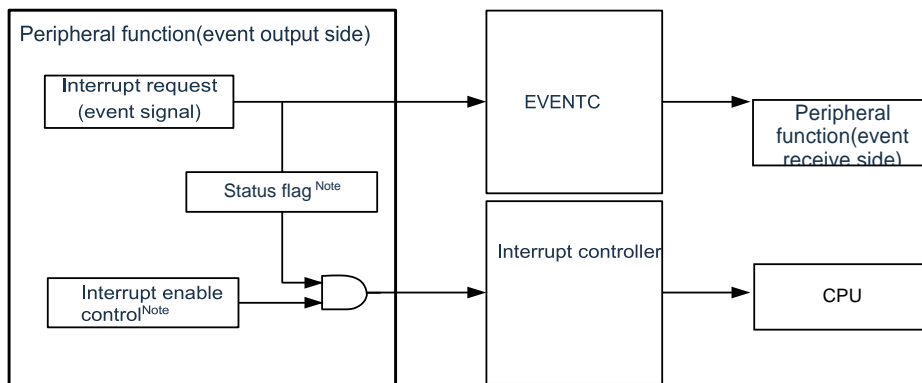
19.4 Operation of EVENTC

The path used by the event signal generated by each peripheral function as an interrupt request for the interrupt control circuit and the path used as an eventc event are independent of each other. Therefore, each event signal is independent of interrupt control and can be used as an event signal for the operation of peripheral functions of the event receiver.

The relationship between interrupt handling and EVENTC is shown in Figure 19-3. This figure takes the relationship between peripheral functions with interrupt request status flags and interrupt allow bits (which control whether to allow or disable) as an example.

The peripheral function that accepts an event through EVENTC operates according to the operation of the receiver peripheral function after receiving the event (refer to Table 19-3 Correspondence between the setting value of ELSELRn register (n=00~14) and the operation when the link target peripheral function receives the event

Figure 19-3 Relationship between interrupt processing and EVENTC



Note Some peripheral functions do not have this feature.

The response of the peripheral function that receives the event is shown in Table 19-4.

Table 19-4 Response of the peripheral function of the received event

Event receive target No.	Function of event link target	Operation after event acceptance	Response
1	A/D converter	A/D conversion	The EVENTC event becomes a hardware trigger for A/D conversion directly.
2	Timer input for Timer40 channel 0	The delay counter enters the measurement external event counter for pulse width	Edge detection is performed after 3 or 4 f_{CLK} cycles from the occurrence of the EVENTC event.
3	Timer4 Timer input for 0 channel 1	The delay counter enters the measurement external event counter for pulse width	Edge detection is performed after 3 or 4 f_{CLK} cycles from the occurrence of the EVENTC event.
4	The EPWM output controls the truncation source	Forced truncation of the pulse output	Becomes a forced truncation state after 2 or 3 EPWM operating clock cycles from the occurrence of an EVENTC event.

Chapter 20 Interrupt Function

The Cortex-M0+ processor has a built-in Nested Vector Interrupt Controller (NVIC) that supports up to 32 interrupt request (IRQ) inputs, as well as one non-maskable interrupt (NMI) input, and multiple internal exceptions.

The interrupt source for 32 interrupt request (IRQ) inputs and 1 non-maskable interrupt (NMI) input is processed in this system. This user manual only describes the handling in this system, Cortex-M0+ processors built-in NVIC functions, please refer to the Cortex-M0+ processor user manual.

20.1 Types of interrupt function

There are two types of interrupt functions.

- (1) Interrupts can be masked
This is a shielded controlled interrupt. If the interrupt mask flag register is not open, the interrupt request will not be responded to even if it is generated.
It can generate a standby release signal to cancel the deep sleep mode and sleep mode.
Masked interrupts are divided into external interrupt requests and internal interrupt requests.
- (2) Interrupts cannot be masked
This is an unmasked interrupt that the CPU must respond to once the interrupt request is made.

20.2 Interrupt source and structure

Refer to Table 20-1 for a list of interrupt sources.

Table 20-1 List of interrupt sources (1/3)

Interrupt handling	Interrupt source No.	Interrupt sources		Internal/External	Basic structure type ^{Note1}
		Name	Trigger		
Maskable	0	INTLVI	Voltage detection ^{Note 2}	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	Timer channel 01 count end or capture end (when the high 8-bit timer is operating).	Internal	(A)
	6	INTKR	Key interrupt		
	7	INTST2/ INTSSPI20/ INTIIC20	UART2 transmit end or buffer empty interrupt/SSPI20 transfer end or buffer empty interrupt/IIC20 transfer end		
	8	INTSR2/ INTSSPI21/ INTIIC21	UART2 receive end/SSPI21 transfer end or buffer empty interrupt/IIC21 transfer end		
	9	INTSRE2	Communication errors during UART2 reception		
	10	INTST0/ INTSSPI00/ INTIIC00	UART0 transmit end or buffer empty interrupt/SSPI00 transfer end or buffer empty interrupt/IIC00 transfer end		
	11	INTSR0/ INTSSPI01/ INTIIC01	UART0 receive/SSPI01 transfer end or buffer empty interrupt/IIC01 transfer end		
12	INTSRE0	Communication errors during UART0 reception			

Note: 1. The basic composition types (A) to (C) correspond to Figure 20-1 (A)~(C).

2. This is when bit7 (LVIMD) of the voltage sense level register (LVIS) is set to "0".

Table 20-1 List of interrupt sources (2/3)

Interrupt handling	Interrupt source No.	Interrupt sources		Internal/External	Basic structure type ^{Note1}
		Name	Trigger		
Maskable	13	INTST1/ INTSSPI10/ INTIIC10/ INTSPI	UART1 transmit end or buffer empty interrupt/SSPI10 transfer end or buffer empty interrupt/IIC10 transfer end	Internal	(A)
	14	INTSR1/ INTSSPI11/ INTIIC11	UART1 receive end or buffer empty interrupt/SSPI11 transfer end or buffer empty interrupt/IIC11 transfer end		
	15	INTSRE1	Communication errors during UART1 reception		
	16	INTIICA0	IICA0 communication end		
	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	Real-time clock fixed period/alarm clock consistency detection		
	23	INTIT	Detection of interval signals		
	24	INTCMP0	Comparator detect 0		
	25	INTCMP1	Comparator detect 1		
	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 programming is over			

Note: 1. The basic composition types (A) to (C) correspond to Figure 20-1 (A)~(C).

Table 20-1 List of interrupt sources (3/3)

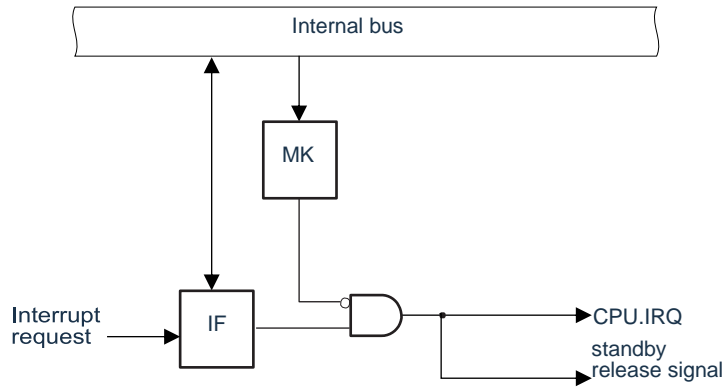
Interrupt handling	Interrupt source No.	Interrupt sources		Internal/External	Basic structure type ^{Note1}
		Name	Trigger		
Not maskable	—	INTWDT	Watchdog timer interval interrupt ^{Note 2}	Internal	(C)

Note: 1. The basic composition types (A) to (C) correspond to Figure 20-1 (A)~(C).

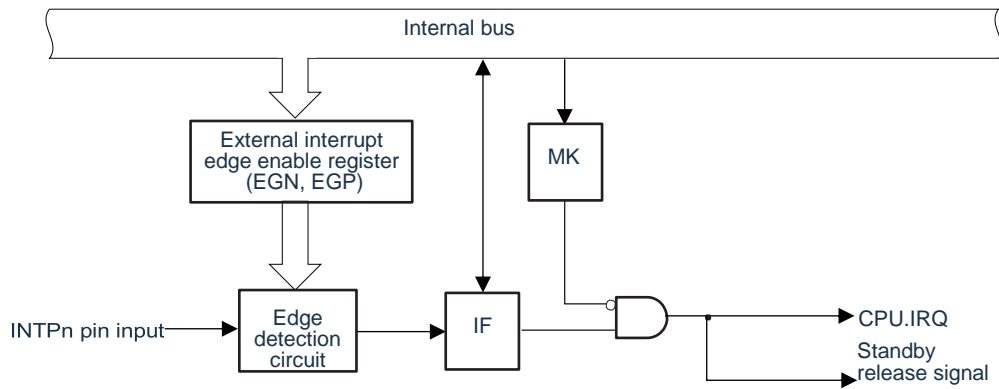
2. This is the case when bit7 (WDTINT) of the option byte (000C0H) is set to "1".

Figure 20-1 Basic structure of interrupt function

(A) Internally maskable interrupt

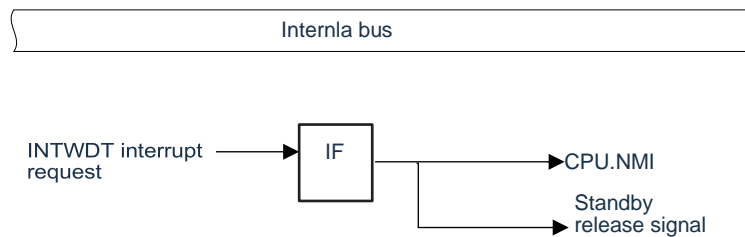


(B) Externally maskable interrupt (INTPn)



Note n=0~3

(C) Non-maskable interrupt



Note: The interrupt request flag for non-maskable interrupts has no physical registers and cannot generate interrupt requests through bus read and write registers.

20.3 Registers controlling interrupt function

The interrupt function is controlled by the following four registers.

- Interrupt request flag register (IF00~IF31)
- Interrupt mask flag register (MK00~MK31)
- External interrupt rising edge enable register (EGP0)
- External interrupt falling edge enable register (EGN0)

20.3.1 Interrupt request flag registers (IF00 to IF31)

The interrupt request flag is set to "1" by generating a corresponding interrupt request or executing an instruction.

The interrupt request flag is cleared to "0" by generating a reset signal or executing an instruction.

The IF00L to IF31L registers are set by 8-bit memory operation instructions.

Or set the IF00 to IF31 registers by 32-bit memory operation instructions.

After a reset signal is generated, the value of these registers changes to "0000_0000H".

Figure 20-2 Format of interrupt request flag register (IF_m) (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	The interrupt request flag for the interrupt source numbered 0 to 31
0	No interrupt request signal is generated.
1	An interrupt request is generated and is in the interrupt request state.

Note: 1. The correspondence between the interrupt source and the interrupt request flag register is shown in Table 20-2

2. The correspondence between the interrupt request flag register and CPU.IRQ is shown in Figure 20-4

20.3.2 Interrupt mask flag register (MK00~MK31)

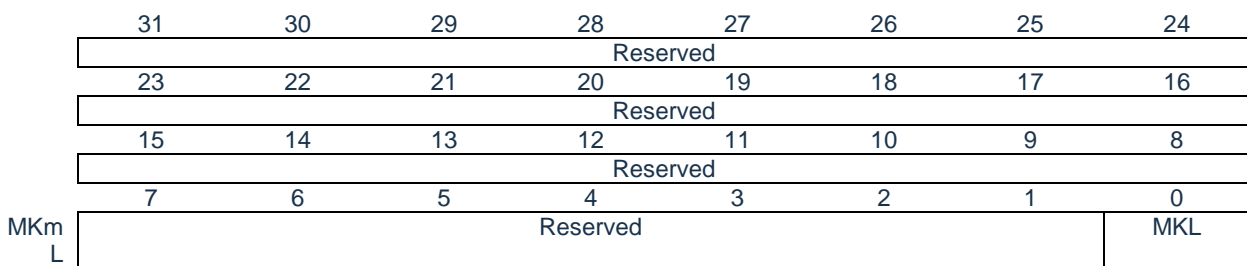
The interrupt masking flag setting allows or disables the corresponding maskable interrupt processing.

Set the MK00L~MK31L registers via 8-bit memory operation instructions or set MK00~MK31 registers via 32-bit memory operation instructions.

After the reset signal is generated, the values of these registers become “FFFF_FFFF”.

Figure 20-3 Format of interrupt request masking 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



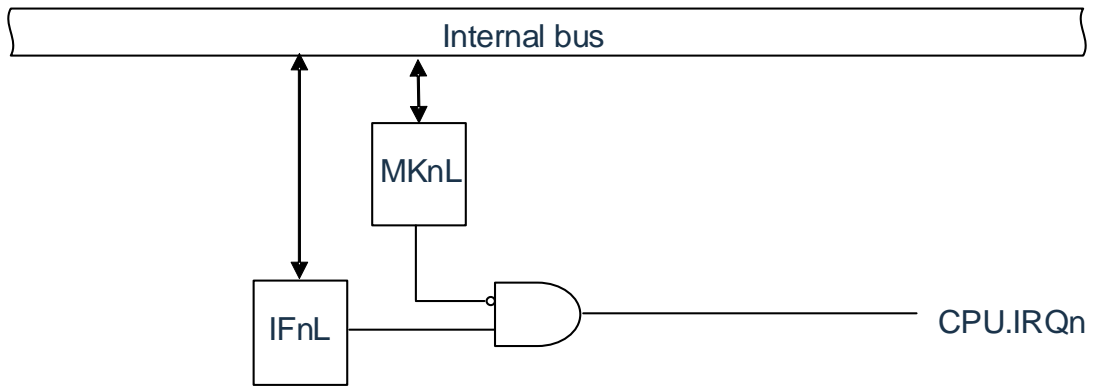
MKmL	Interrupt handling control for interrupt sources numbered 0 to 31 ^{Note 1}
0	Interrupt handling is allowed.
1	Interrupt processing is prohibited.

- Note: 1. The correspondence between the interrupt source and the interrupt request masking register is shown in Table 20-2
2. The correspondence between the interrupt request flag register and CPU.IRQ is shown in Figure 20-4

Table 20-2 Relationship between interrupt sources and flag registers

Number	Source of the interrupt	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	INTST2/INTSSPI20/INTIIC20	IF07.IFL	MK07.MKL
8	INTSR2/INTSSPI21/INTIIC21	IF08.IFL	MK08.MKL
9	INTSRE2	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	INTST1/INTSSPI10/INTIIC10	IF13.IFL	MK13.MKL
14	INTSR1/INTSSPI11/INTIIC11	IF14.IFL	MK14.MKL
15	INTSRE1	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	INTKR	IF23.IFL	MK23.MKL
24	INTCMP0	IF24.IFL	MK24.MKL
25	INTCMP1	IF25.IFL	MK25.MKL
26	INTRAMPRTERR	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 20-4 Relationship between each flag register and CPU.IRQ



20.3.3 External interrupt rising edge enable register (EGP0), External interrupt falling edge enable register (EGN0)

These registers set the effective edges of INTP0 to INTP3.

Set the EGP0 and EGN0 registers via 8-bit memory operation instructions.

After a reset signal is generated, the values of these registers become "00H".

Figure 20-5 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	Valid edge selection for the INTPn pin (n=0~11)
0	0	Disable detection of edges.
0	1	Falling edge
1	0	Rising edge
1	1	Rising and falling edges

The ports corresponding to the EGPn bit and the EGNn bit are shown in Table 20-3.

Table 20-3 Interrupt request signals corresponding to the EGPn and EGNn bits

Detect enable bits		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, a valid edge may be detected and an INTPn interrupt may be generated. When switching to output mode, the port mode register (PMxx) must be set to "0" after the disable detection edge (EGPn, EGNn=0, 0).

Remark 1. For ports detected by edge, refer to "2.1 Pin Functions".

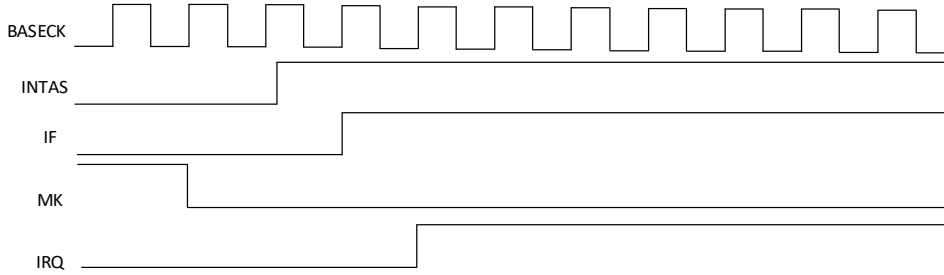
2. n=0~3

20.4 Operation of interrupt handling

20.4.1 Reception of maskable interrupt requests

If the interrupt request flag is set to “1” and the masked (MK) flag for the interrupt request is cleared “0”, it enters a state that accepts maskable interrupt requests and can pass the interrupt request to NVIC.

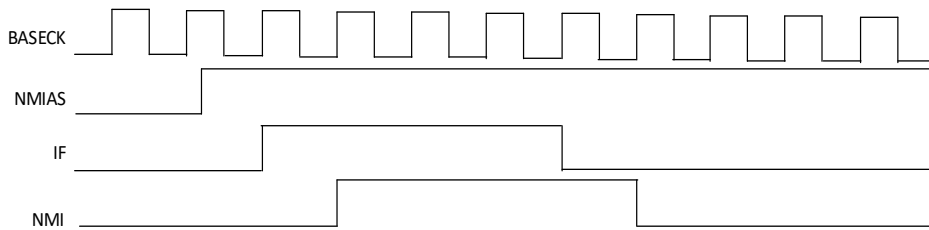
From setting the interrupt request flag to 1 to setting the IRQ of the CPU to 1, only 1 clock is required.



20.4.2 Reception of non-maskable interrupt requests

If a non-maskable interrupt request is generated, the interrupt request flag is set to “1” and passed directly to NVIC.

From the interrupt request flag being set to 1 to the CPU's NMI being set to 1, only 1 clock is required.



Chapter 21 Key Interrupt Function

The number of channels entered by key interrupt varies by product.

21.1 Function of key interrupt

A key interrupt (INTKR) can be generated by giving the key interrupt input pin (KR0 to KR5) on the falling edge of the input.

Table 21-1 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

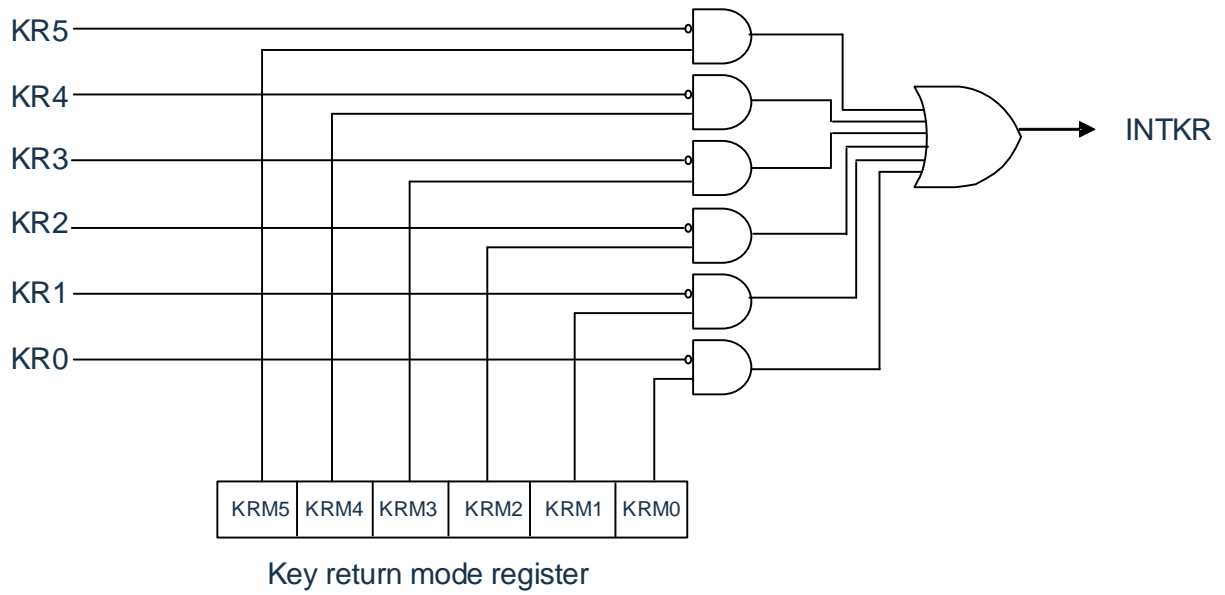
21.2 Structure of key interrupt

The key interrupt consists of the following hardware.

Table 21-2 Structure of key interrupts

Item	Control registers
Control registers	Key return mode register (KRM) Port mode register (PMx) Port mode control register (PMCx)

Figure 21-1 Block diagram of key interrupt



21.3 Registers for controlling key interrupt

The key interrupt function is controlled through the following registers.

- Key return mode register (KRM)
- Port mode register (PMx)

21.3.1 Key return mode register (KRM)

The KRM0 to KRM5 bits control the KR0 to KR5 signals.

The KRM register is set via an 8-bit memory operation instruction.

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

Figure 21-2 Format of key return mode register (KRM)

Address: 40044B37H	After reset: 00H	R/W						
Symbol	7	6	5	4	3	2	1	0
KRM	0	0	KRM5	KRM4	KRM3	KRM2	KRM1	KRM0

KRMn	Control of key interrupt mode
0	The key interrupt signal is not detected.
1	Detects key interrupt signals.

Note 1. The internal pull-up resistor can be used by setting the object bit of the pull-up resistor register (PUx) of the key interrupt input pin to "1".

2. An interrupt is generated if the object bit of the KRM register is set in a state that inputs a low level to the key interrupt input pin. To ignore this interrupt, the KRM register must be set after the interrupt handling is disabled by the interrupt masking flag. The interrupt request flag must then be cleared after waiting for the key interrupt input to be low level width (t_{KR}) (see data sheet) to allow interrupt handling.

3. Pins that are not used in key interrupt mode can be used as usual ports.

Remark1.n=0~5

21.3.2 Port mode register (PMx)

When used as a key interrupt input pin (KR0~KR5), the PMCxn bit must be set to “0” and the PMxn bit must be set to “1” respectively. In this case, the output latch of Pxn can be “0” or “1”.

The PM x register is set via an 8-bit memory operation command.

After the reset signal is generated, the value of this register changes to “FFH”.

An internal pull-up resistor can be used in bits using a pull-up resistor select register (PUx).

For the format of the port mode registers, refer to “2.3.1 Port Mode Registers (PMxx)”.

Chapter 22 Standby Function

22.1 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 a 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 28 Option Bytes".

22.2 Sleep mode

22.2.1 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 22-1.

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.

Notice 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 release sleep mode. Therefore, even if the WFI command is executed in this case, it does not shift to sleep mode.

Table 22-1 Operation status in sleep mode (1/2)

Sleep mode setting		Execution of WFI instructions while the CPU is running at the 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: 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					
Code flash memory							
RAM		Operation stopped (operable when DMA is executed).					
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							
Clock output/buzzer output		Operable					
A/D converter							
Comparator							
General-purpose serial communication unit (SCI)							
Serial interface (IICA)							
Data transfer controller (DMA)							
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				Operable when DMA is executed in the operation of RAM area.		
	General-purpose CRC						
RAM parity check function		Operable when DMA is executed.					
SFR protection function							

Remark Operation stopped: Operation is automatically stopped before switching to the sleep mode.

Operation disabled: Operation is stopped before switching to the sleep mode.

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

Table 22-1 Operation status in sleep mode (2/2)

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) <ul style="list-style-type: none"> • 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	
Code flash memory			
RAM		Operation stopped (operable when DMA is executed).	
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	
Comparator		Operable	
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 (IICA)		Operation disabled	
Data transfer controller (DMA)		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	Operable when DMA is executed in the operation of RAM area.	
RAM parity check function		Operable when DMA is executed.	
SFR protection function			

Remark Operation stopped: Operation is automatically stopped before switching to the sleep mode.

Operation disabled: Operation is stopped before switching to the sleep mode.

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

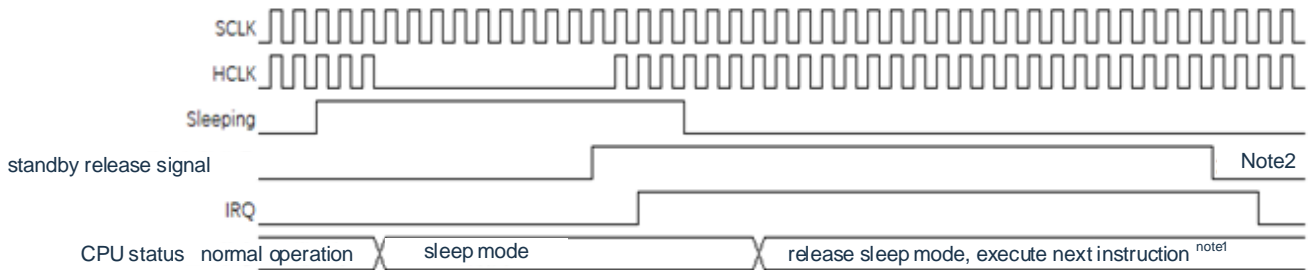
22.2.2 Sleep mode release

The sleep mode can be released by any interrupt or external reset, POR reset, low voltage detection reset, RAM parity check error 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 22-1 Release sleep mode by interrupt requests



Note1. 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.

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.

Notice: 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 22-2 Release sleep mode by resets



Note1: For reset processing, please refer to "Chapter 23 Reset Function". For reset processing of power-on reset (POR) and voltage detection (LVD), refer to "Chapter 24 Power-on Reset Circuit".

22.3 Deep sleep mode

22.3.1 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 22-2.

The deep sleep mode can only be set if the CPU clock before setting is as the main system clock.

Notice 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 release deep sleep mode. Therefore, if the WFI instruction is executed in this case, it is released as soon as it enters deep sleep mode. Returns to operation mode after executing the WFI instruction and after a deep sleep mode release time has elapsed.

Table 22-2 Operation status in deep sleep mode

Deep sleep mode setting		Execution of WFI instructions while the CPU is running at main system clock		
Item		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}			
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: 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		
Code flash memory		Operation stopped		
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		See "Chapter 10 Watchdog Timer"		
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		
Comparator		Operable (only if no digital filter is used).		
General-purpose serial communication unit (SCI)		Only SSPIp and UARTq can wake up. Except for SSPIp and UARTq, operation is disabled.		
Serial interface (SPI)		Operation disabled		
Serial interface (IICA)		Address matching for wakeup can be performed.		
Data transfer controller (DMA)		Can accept DMA boot sources.		
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			
RAM parity check function		Operation stopped		
SFR protection function				

Remark Operation stopped: Operation is automatically stopped before switching to the deep sleep mode.

Operation disabled: Operation is stopped before switching to the deep sleep mode.

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

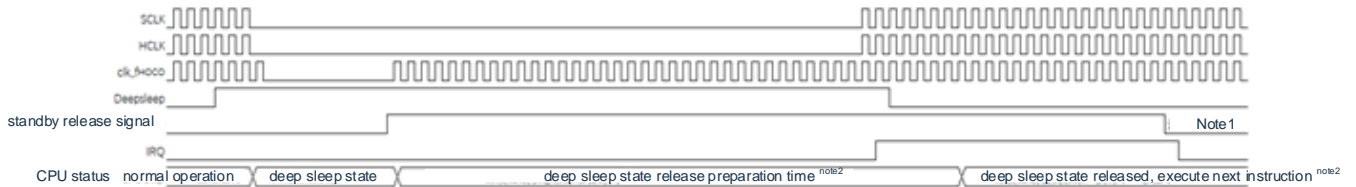
22.3.2 Deep sleep mode release

The deep sleep mode can be released by the following two sources.

(a) 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 instruction is executed.

Figure 22-3 Release deep sleep mode by interrupt requests



Note 1. Standby release signal: For details of the standby release signal, please refer to “Figure 20-1 Basic structure of interrupt function”.

2. Deep sleep release preparation time.

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.

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).

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.

Remark The oscillation accuracy of the high-speed on-chip oscillator clock varies steadily depending on temperature conditions and during deep sleep mode.

(b) 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 22-4 Release deep sleep mode by resetting



Note For reset processing, see “Chapter 23 Reset Function”. For reset processing of power-on reset (POR) and voltage detection (LVD), see “Chapter 24 Power-on Reset Circuit”.

Chapter 23 Reset Function

The following 7 methods generate reset signals.

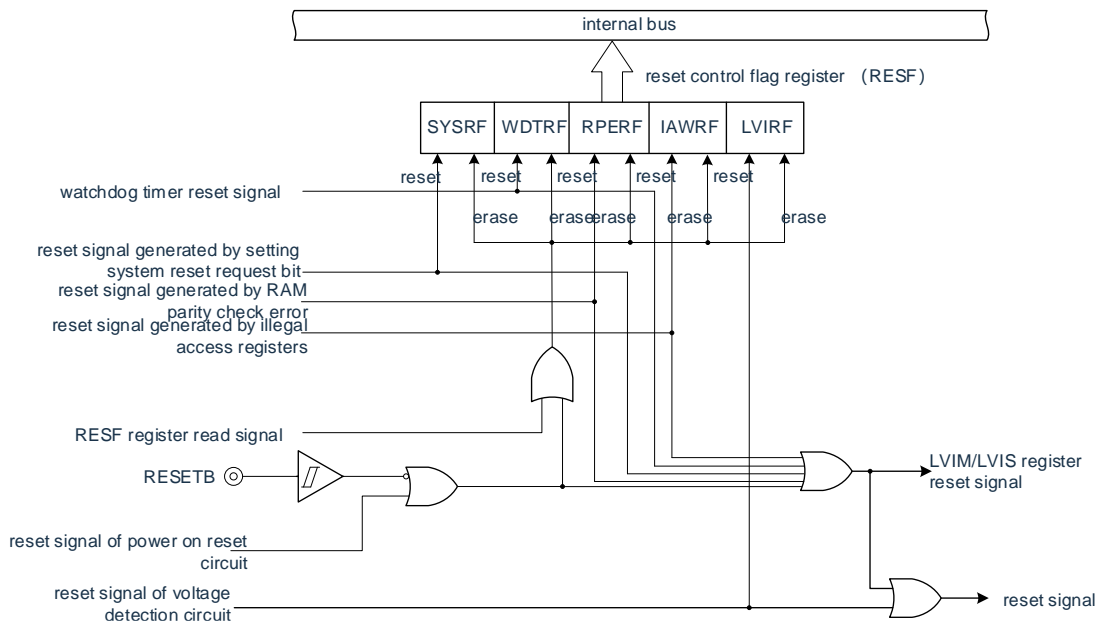
- (1) An external reset is entered via the RESETB pin.
- (2) An internal reset is generated by a program runaway detection of the watchdog timer.
- (3) An internal reset is generated by comparing the supply voltage to the sense voltage of the power-on reset (POR) circuit.
- (4) An internal reset is generated by comparing the supply voltage of the voltage detection circuit (LVD) with the sense voltage.
- (5) Request register bit due to system reset (AIRCR. SYSRESETREQ) is set to 1 to produce an internal reset.
- (6) Internal reset due to RAM parity error.
- (7) Internal reset due to access to illegal memory.

Internal reset is the same as external reset, and after a reset signal is generated, the program is executed from the user-defined program start address.

When a low level is entered into the RESET B pin, or the watchdog timer detects a program runaway, or detects the voltage of the POR circuit and the LVD circuit, or the system reset request bit is assessed, or a RAM parity error occurs, or the illegal memory is accessed, A reset is generated and each hardware becomes a state as shown in Table 23-1.

- Note 1. During an external reset, a low level of at least 10us must be entered into the RESETB pin. If an external reset is performed when the supply voltage rises, the supply must be turned on after the RESETB pin is low and maintained at least 10u over the operating voltage range shown in the AC characteristics of the user manual s low level, then enter high.
2. Stop oscillating the X1 clock, XT1 clock, high-speed internal oscillator clock, and low-speed internal oscillator clock during the reset signal. The inputs to the external master system clock and external subsystem clock are invalid.
 3. If a reset occurs, each SFR is initialized so that the port pins become the following state:
 - P10, P26, P40, P137: High impedance during external reset or POR reset. High during other resets and after receiving the reset (internal pull-up resistors are connected).
 - Ports other than P10, P26, P40, P137: High impedance during and after receiving a reset.

Figure 23-1 Block diagram of reset function



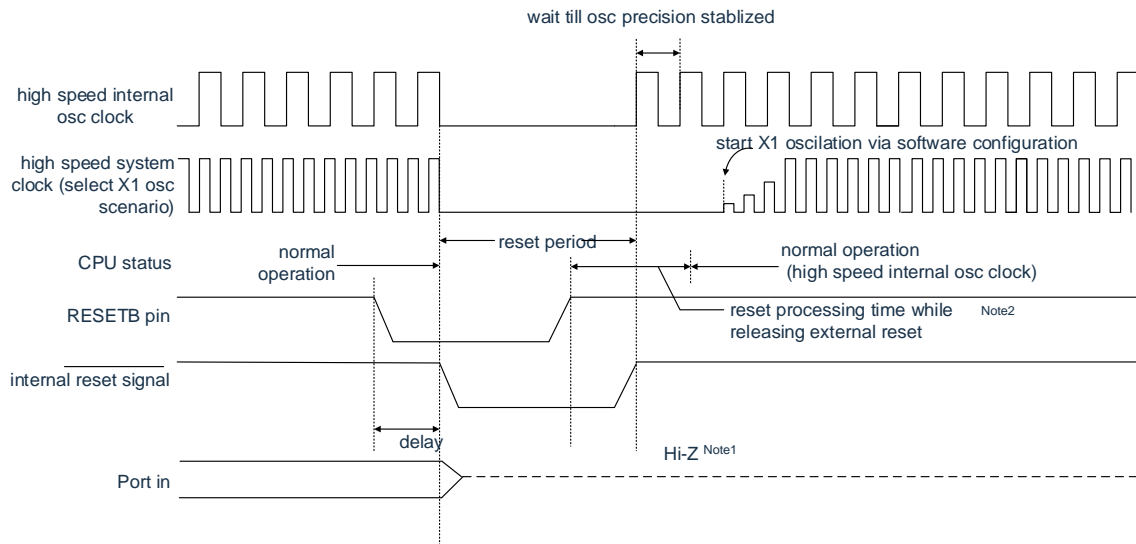
Notice An internal reset of the LVD circuit does not reset the LVD circuit.

- Remark 1. LVIM: Voltage detection register
 2. LVIS: Voltage detection level register

Reset timing

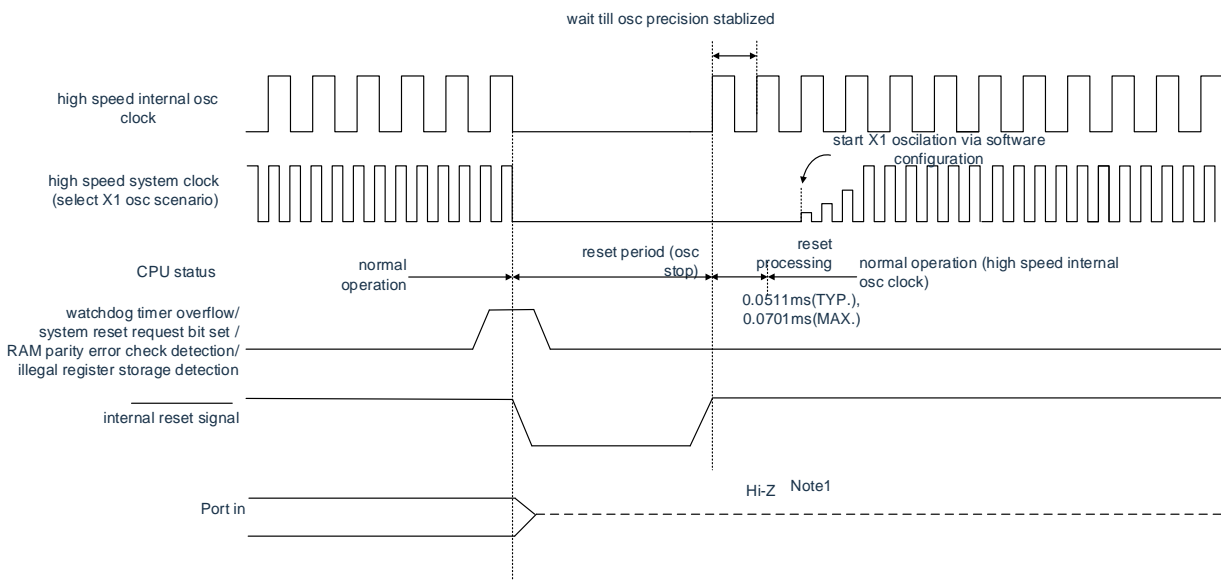
When low is input to the RESETB pin, a reset is generated. Then, if RESETB is quoted high, the reset state is released, and execution begins with a high-speed on-chip oscillator clock after the reset process is complete.

Figure 23-2 Timing of the RESETB input



For resets caused by watchdog timer overflow, system reset request bit assertion, RAM parity error detection, or detection of illegal memory access, the reset state is automatically released and execution begins with a high-speed internal oscillator clock after the reset process is completed.

Figure 23-3 Reset timing due to overflow of watchdog timer, set of system reset request bits, detection of RAM parity errors, or detection of illegal memory access



Note 1 Port pins P10, P26, P40, P137 become the following states:

- High impedance during external reset or POR reset.
- High during other resets and after receiving the reset (internal pull-up resistor is connected).

Notice The watchdog timer is no exception, and is reset when an internal reset occurs.

If $V_{DD} \geq V_{POR}$ or $V_{DD} \geq V_{LVD}$ is satisfied after the reset for the reset generated by the voltage detection of the POR circuit and the LVD circuit, the reset state is released and the program execution starts with the high-speed on-chip oscillator clock after the reset is processed. For details, refer to Chapter 24 Power-on Reset Circuit and Chapter 25 Voltage Detection Circuit.

Remark V_{POR} : POR supply voltage rising detection voltage
 V_{LVD} : LVD detection voltage

Table 23-1 Operation status during resetting

Item	During resetting			
System clock	Clock supply to the CPU is stopped			
Main system clock	f_{IH}	Operation stopped		
	f_X	Operation stopped (X1 pin and X2 pin are in input port mode).		
	f_{EX}	Clock input is invalid (pin is in input port mode).		
Subsystem clock	f_{XT}	Operable		
	f_{EXS}	Clock input is invalid (pin is in input port mode).		
f_{IL}	Operation stopped			
CPU	Operation stopped			
Code flash memory	Operation stopped			
RAM	Operation stopped			
Port (latch)	High impedance ^{Note1}			
General-purpose timer unit	Operation stopped			
Real-time clock (RTC)				
15-bit interval timer				
Watchdog timer				
Clock output/buzzer output				
A/D converter				
Comparator ^{Note1}				
General-purpose serial communication unit (SCI)				
Serial interface (IICA)				
Data transfer controller (DMA)				
Power-on-reset function			Detection operations can be performed.	
Voltage detection function	Operation is possible in the case of an LVD reset and stopped in the case of other types of reset.			
External interrupt	Operation stopped			
Key interrupt function				
CRC Calc. function			High-speed CRC	
			General-purpose CRC	
RAM parity check function				
SFR protection function				

Note 1. Pins P10,P26,P40,P137 change to the following states.

High impedance during an external reset or POR reset. High level during other reset periods (an internal pull-up resistor is connected).

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

23.1 Register for confirming reset sources

23.1.1 Reset control flag register (RESF)

The BAT32G135 microcontroller has multiple internal reset sources. The Reset Control Flag Register (RESF) holds the reset source where the reset request occurred. The RESF register can be read via an 8-bit memory operation instruction.

The SYSRF, WDTRF, RPERF, IAWRF, and LVIRF flags are cleared by the input of RESETB, the reset of the power-on reset (POR) circuit, and the reading of the RESF register. To determine the reset source, the value of the RESF register must be saved to any RAM, and then judged by its RAM value.

Figure 23-4 Format of reset control flag register (RESF)

Address: 40020440H After reset: Undefined value^{Note1} R

Symbol	7	6	5	4	3	2	1	0
RESF	SYSRF	0	0	WDTRF	0	RPERF	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.

RPERF	Internal reset request generated by RAM parity error
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 1. It varies depending on the reset source. Refer to Table 23-2.

Notice In the case of allowing RAM parity error reset (RPERDIS=0), the “RAM area” must be initialized when accessing data. When executing instructions from the RAM area, the area of “used RAM area + 10 bytes” must be initialized. By generating a reset, it enters a state that allows RAM parity error reset (RPERDIS=0). For more information, see “26.3.3 RAM Parity Error Detection Function”.

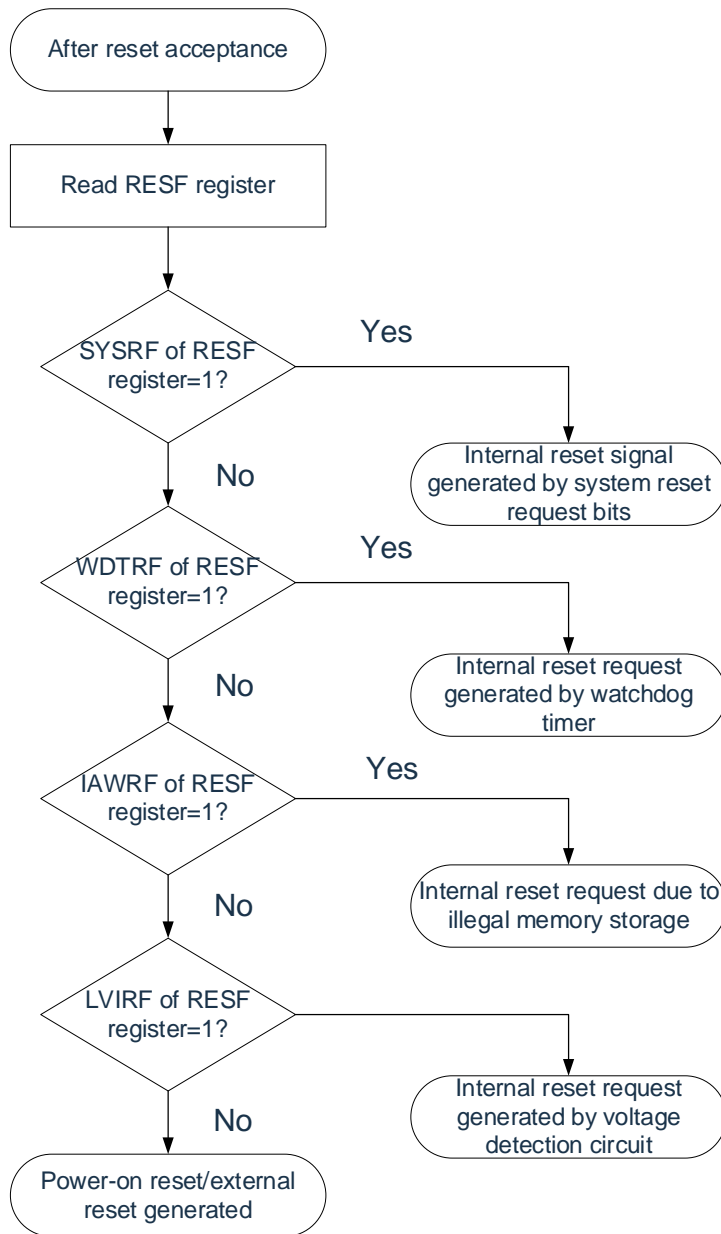
It varies depending on the reset source. Refer to Table 23-2.

Table 23-2 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 reset source confirmation procedure is shown in Figure 23-5.

Figure 23-5 Confirmation steps for reset source



Notice The above process is an example of confirmation steps.

Chapter 24 Power-On Reset Circuit

24.1 Function of power-on reset circuit

The power-on reset circuit (POR) has the following functions.

- Generates internal reset signal at power on.
The reset signal is released when the supply voltage (V_{DD}) exceeds the detection voltage (V_{POR}).
Note that the reset state must be retained until the operating voltage becomes in the range defined in AC Characteristics. This is done by utilizing the voltage detection circuit or controlling the externally input reset signal.
- Compares supply voltage (V_{DD}) and detection voltage (V_{PDR}). Generates internal reset signal when $V_{DD} < V_{PDR}$. Note that, after power 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.

Notice When the power-on reset circuit generates an internal reset signal, the reset control flag register (RESF) is cleared to "00H".

Remark 1. The BAT32G135 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 23 Reset Function".

2. V_{POR} : POR power supply rise detection voltage

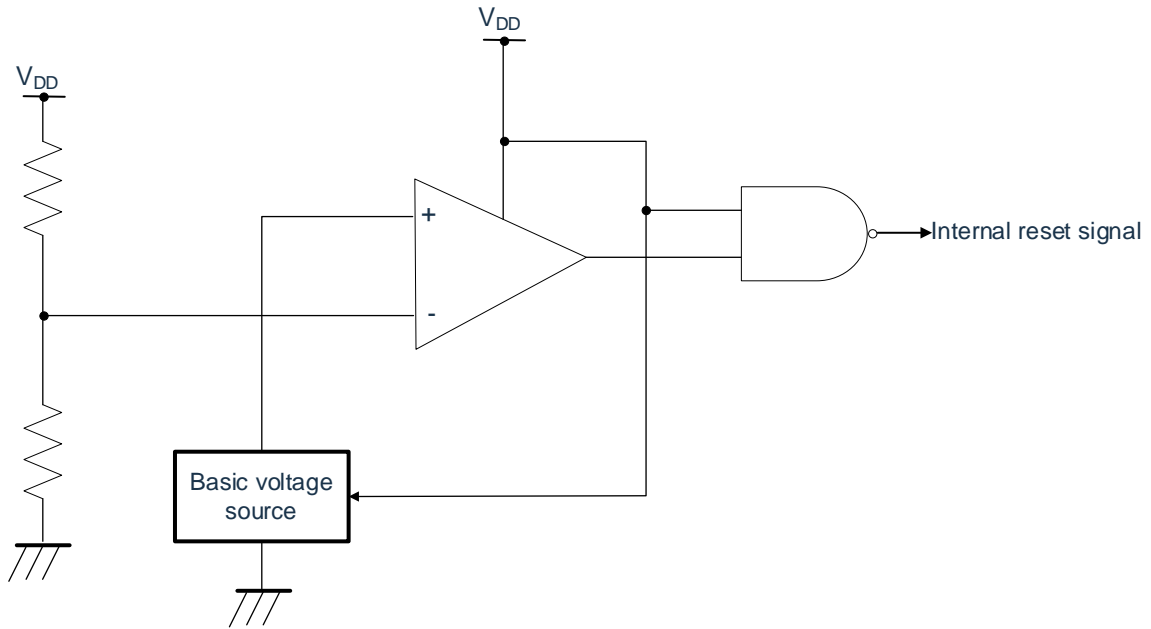
V_{PDR} : POR power supply fall detection voltage

For details, refer to the POR circuit characteristics in the data sheet.

24.2 Structure of power-on reset circuit

The block diagram of the power-on reset circuit is shown in Figure 24-1.

Figure 24-1 Block diagram of power-on reset circuit

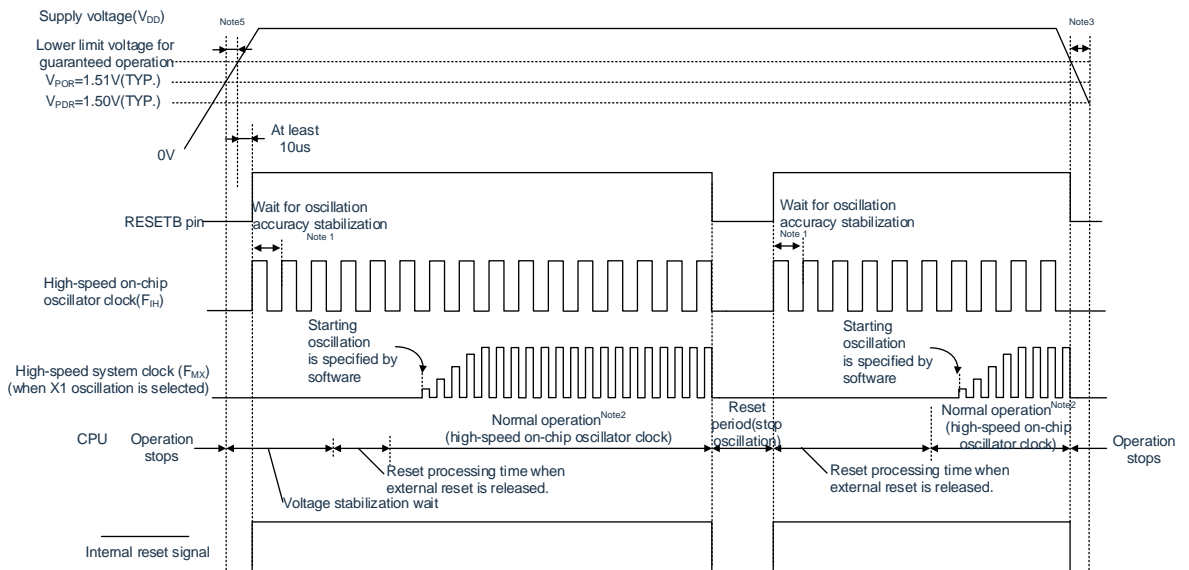


24.3 Operation of power-on reset circuit

The timing of the internal reset signal generation for the power-on reset circuit and the voltage detection circuit is shown below.

Figure 24-2 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.

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.
3. 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.

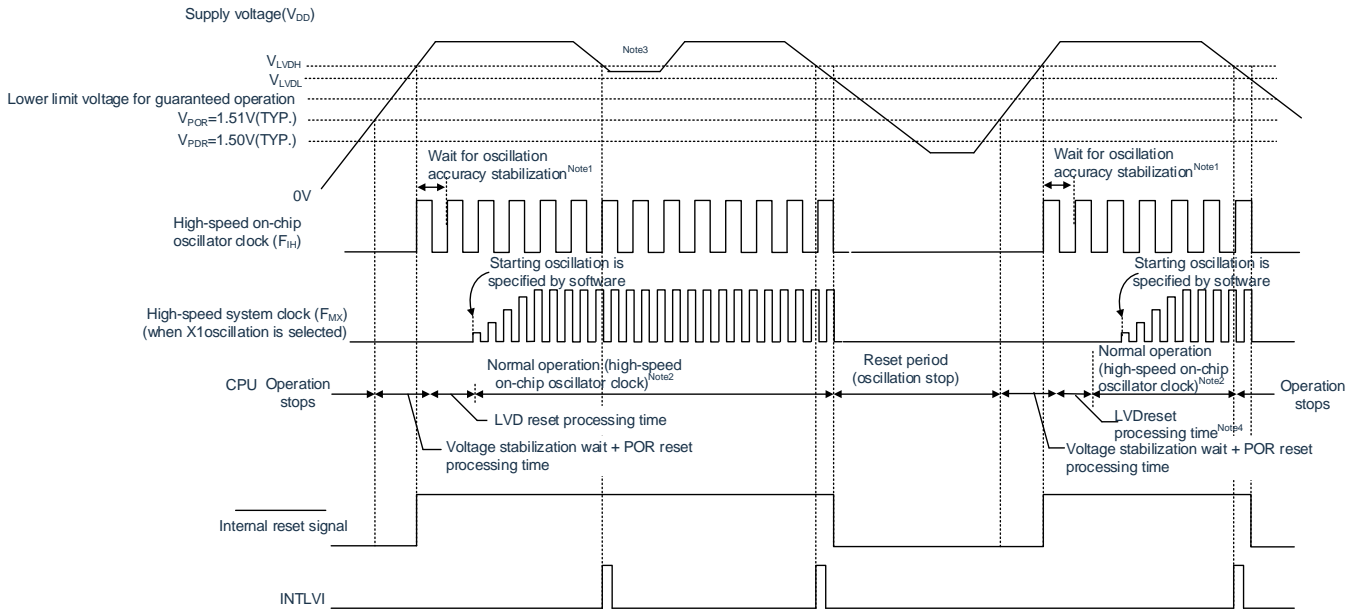
Remark V_{POR} : POR power supply rise detection voltage

V_{PDR} : POR power supply fall detection voltage

Notice When LVD is OFF, the external reset of RESETB pin must be used. For details, please refer to "Chapter 25 Voltage Detection Circuit".

Figure 24-2 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)

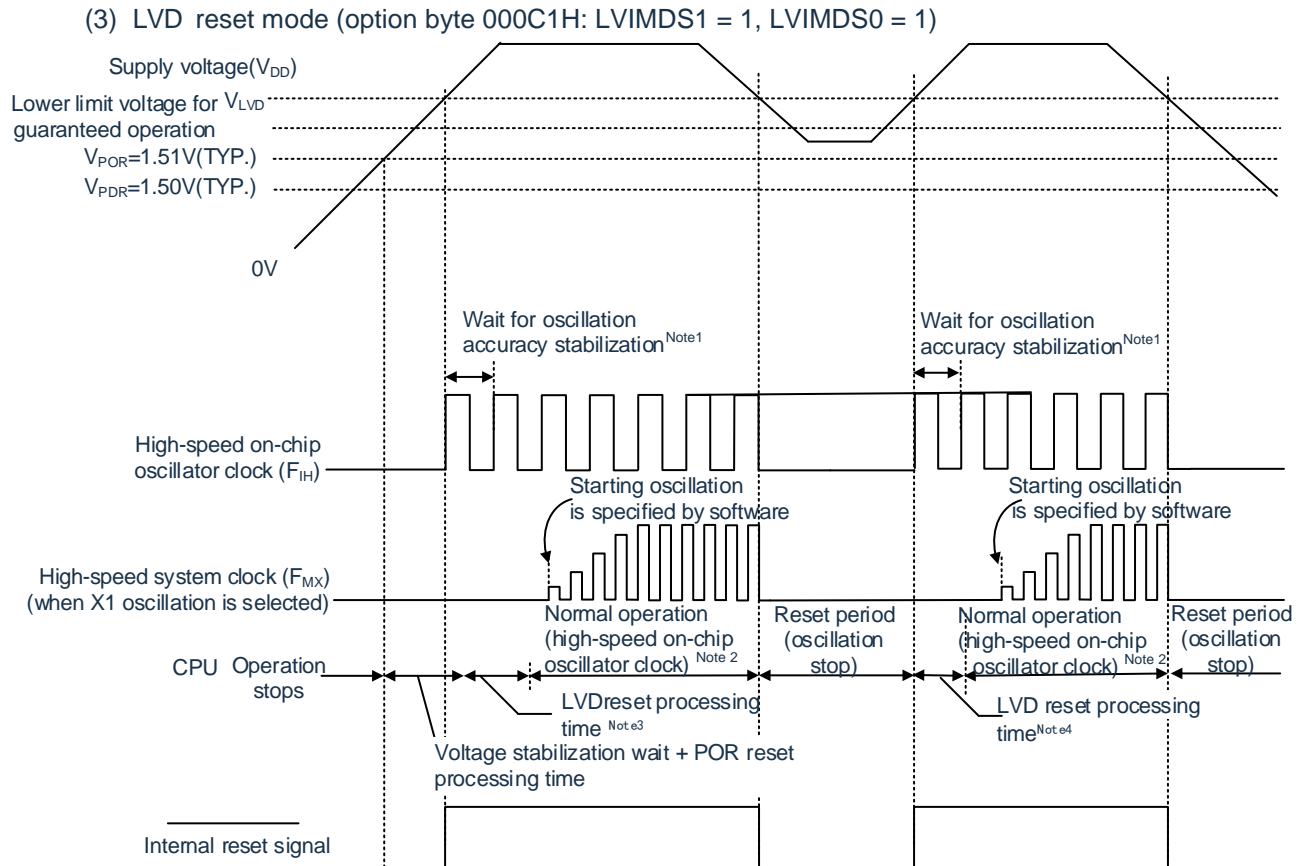


Note 1. The internal reset processing time includes the oscillation accuracy stabilization wait time of the high-speed on-chip oscillator clock.

- 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.
- 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 "Figure 23-8 Setting Procedure for Confirmation/Reset of Operating Voltage" and "Fig. 21-6 Setting Procedure for Interrupt and Reset" after generating INTLVI. "Figure 23-9 Initial Setting Procedure for Interrupt & Reset Mode" after generating INTLVI.
- 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 power supply rise detection voltage
 V_{PDR} : POR power supply fall detection voltage

Figure 24-2 Timing of internal reset signal generation for power-on reset circuit and voltage detection circuit (3/3)



Note 1. The internal reset processing time includes the oscillation accuracy stabilization wait time of the high-speed on-chip oscillator clock.

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.
3. The time until normal operation starts includes the following LVD reset processing time after the LVD detection level (V_{LVD}) is reached as well as the voltage stabilization wait + POR reset processing time after the $V_{POR}(1.51V(TYP.))$ is reached.
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 (V_{LVD}) is reached.

Remark1. V_{LVDH} , V_{LVDL} : LVD detection voltage

V_{POR} : POR supply voltage rise detection voltage

V_{PDR} : POR supply voltage fall detection voltage

2. When the LVD interrupt mode is selected (option byte 000C1H: LVIMD1 = 0, LVIMD0 = 1), the time until normal operation starts after power is turned on is the same as the time specified in Note 3 of Figure 24-2 (3/3).

Chapter 25 Voltage Detection Circuit

25.1 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.

- 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}).
- 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 28 Option Bytes”).
- It can also operate in deep sleep mode.
- 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).

(a) 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.

(b) Reset mode (LVIMDS1, LVIMDS0=1, 1 of option byte)

A detection voltage (V_{LVD}) selected by option byte 000C1H is used to generate or release the reset.

(c) Interrupt mode (LVIMDS1, LVIMDS0=0, 1 of option byte)

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.

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 $V_{DD} \geq V_{LVDH}$ is detected, an internal reset is released.	When $V_{DD} \geq V_{LVD}$ 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 $V_{DD} \geq V_{LVD}$. When $V_{DD} \geq V_{LVD}$ is detected, an internal reset of LVD is released. After the internal reset of LVD is released, if $V_{DD} < V_{LVD}$ or $V_{DD} \geq V_{LVD}$ is detected, then an interrupt request signal (INTLVI) is generated.

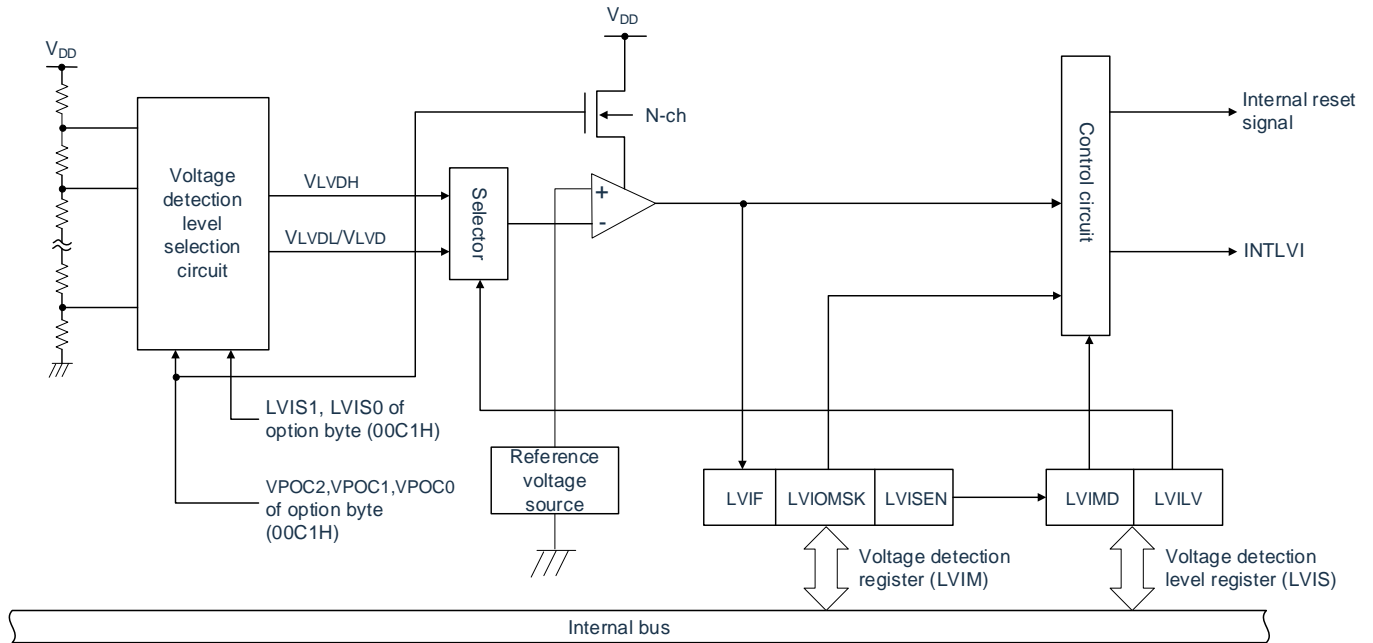
When the voltage detection circuit is in operation, it is possible to check whether the power supply voltage is greater than or less than the detection voltage by reading the voltage detection flag (LVIF: bit 0 of the voltage detection register (LVIM)).

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 23 Reset Function”.

25.2 Structure of voltage detection circuit

The block diagram of the voltage detection circuit is shown in Figure 25-1.

Figure 25-1 Block diagram of voltage detection circuit



25.3 Registers for controlling voltage detection circuit

The voltage detection circuit is controlled by the following registers.

- Voltage detection register (LVIM)
- Voltage detection level register (LVIS)

25.3.1 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".

Figure 25-2 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}	0	0	0	0	0	LVIOMSK	LVIF

LVISEN ^{Note3}	Enable/disable setting of voltage detection level register (LVIS)
0	Disables rewriting LVIS register (LVIOMSK=0 (LVD output mask is invalid)).
1	Enables 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}) \geq detection voltage (V_{LVD}) or LVD is OFF.
1	Supply voltage (V_{DD}) $<$ detection voltage (V_{LVD}).

Note1. 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".

2. Bit0 and bit1 of the LVIM register are read-only bits.

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.

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.

- When LVISEN=1.
- Waiting time from the occurrence of LVD interrupt to the stabilization of LVD detection voltage
- Waiting time from changing the value of the LVILV bit (bit0 of the LVIS register) until the LVD detection voltage stabilizes.

25.3.2 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}.

Figure 25-3 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	0	0	0	0	0	LVILV ^{注2}
LVIMD ^{Note2}	Operation mode of voltage detection							
0	Interrupt mode							
1	Reset mode							
LVILV ^{Note2}	LVD detection level							
0	High voltage detection level (VLVDH)							
1	Low voltage detection level (VLVDL or VLVD)							

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:

- LVIMDS1, LVIMDS0 of Option bytes =1, 0: 00H
- LVIMDS1, LVIMDS0 of Option bytes =1, 1: 81H
- LVIMDS1, LVIMDS0 of Option bytes =0, 1: 01H

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.

Notice1. To rewrite the LVIS registers, it must be done in accordance with the steps in Figure 25-7 and Figure 25-8.

2. Option byte 000C1H selects the mode of operation of the LVD and the detection voltage (VLVDH, VLVDL, VLVD) for each mode. The format of the user option byte (000C1H/010C1H) is shown in Table 25-1. For details of the option byte, refer to "Chapter 28 Option Bytes".

Table 25-1 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

- 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.						

- 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.								

Remark 1. The detection voltage is a TYP Value. For details, please refer to the LVD circuit characteristics in the data sheet.

Table 25-1 Format of user option bytes (000C1H) (2/2)

Address: 000C1H

	7	6	5	4	3	2	1	0
	VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

- 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.					

- LVD is OFF (external reset using the RESETB pin)

Detection voltage		Setting value of option byte						
V_{LVD}		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
rising	rising						LVIMDS1	LVIMDS0
—	—	1	×	×	×	×	×	1
—		Settings other than above are prohibited.						

Notice1. Bit4 must be set to "1".

- 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, it must be set to the reset state by the transfer of the deep 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).

Remark1. ×: Ignore

- The detection voltage is a TYP. value. For details, please refer to the LVD circuit characteristics in the data sheet.

25.4 Operation of voltage detection circuit

25.4.1 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.

- Set bit 7 (LVISEN) of the voltage detection register (LVIM) to "0" (disable rewriting the voltage detection level register (LVIS))
- 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}).

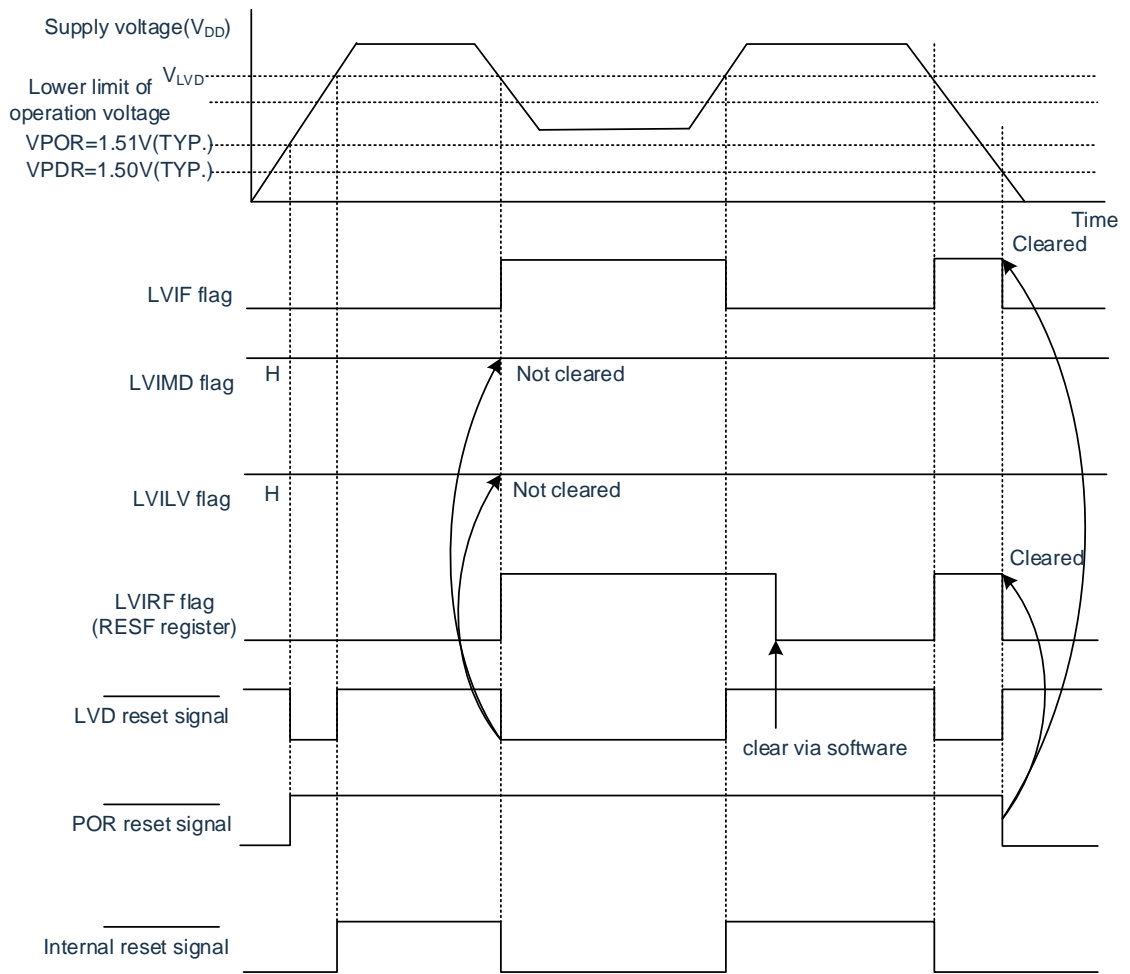
● Operation of LVD reset mode

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 25-4.

Figure 25-4 Timing of internal reset signal generation (LVIMDS1, LVIMDS0=1, 1 of option byte)



Remark V_{POR} : POR power supply rise detection voltage
 V_{PDR} : POR power supply fall detection voltage

25.4.2 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.

- Set bit 7 (LVISEN) of the voltage detection register (LVIM) to "0" (disables rewriting the voltage detection level register (LVIS)).
- 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}).

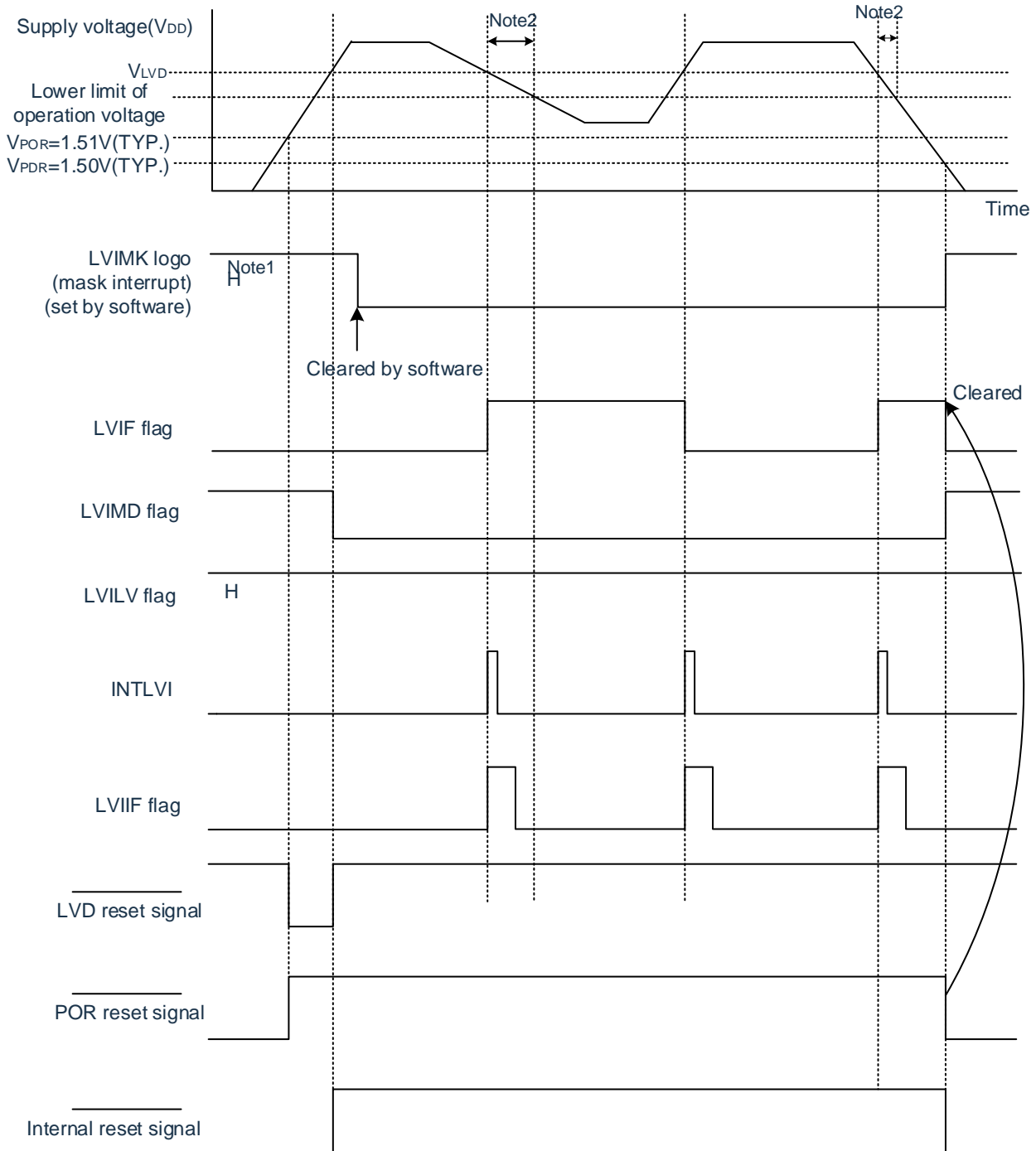
● Operation of LVD interrupt mode

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 25-5.

Figure 25-5 Timing of interrupt signal generation (LVIMDS1, LVIMDS0 of option byte =0, 1)



Note1. After generating a reset signal, the LVIMK flag changes to "1".

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

25.4.3 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.

- Set bit 7 (LVISEN) of the voltage detection register (LVIM) to "0" (disables rewriting the voltage detection level register (LVIS)).
- 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

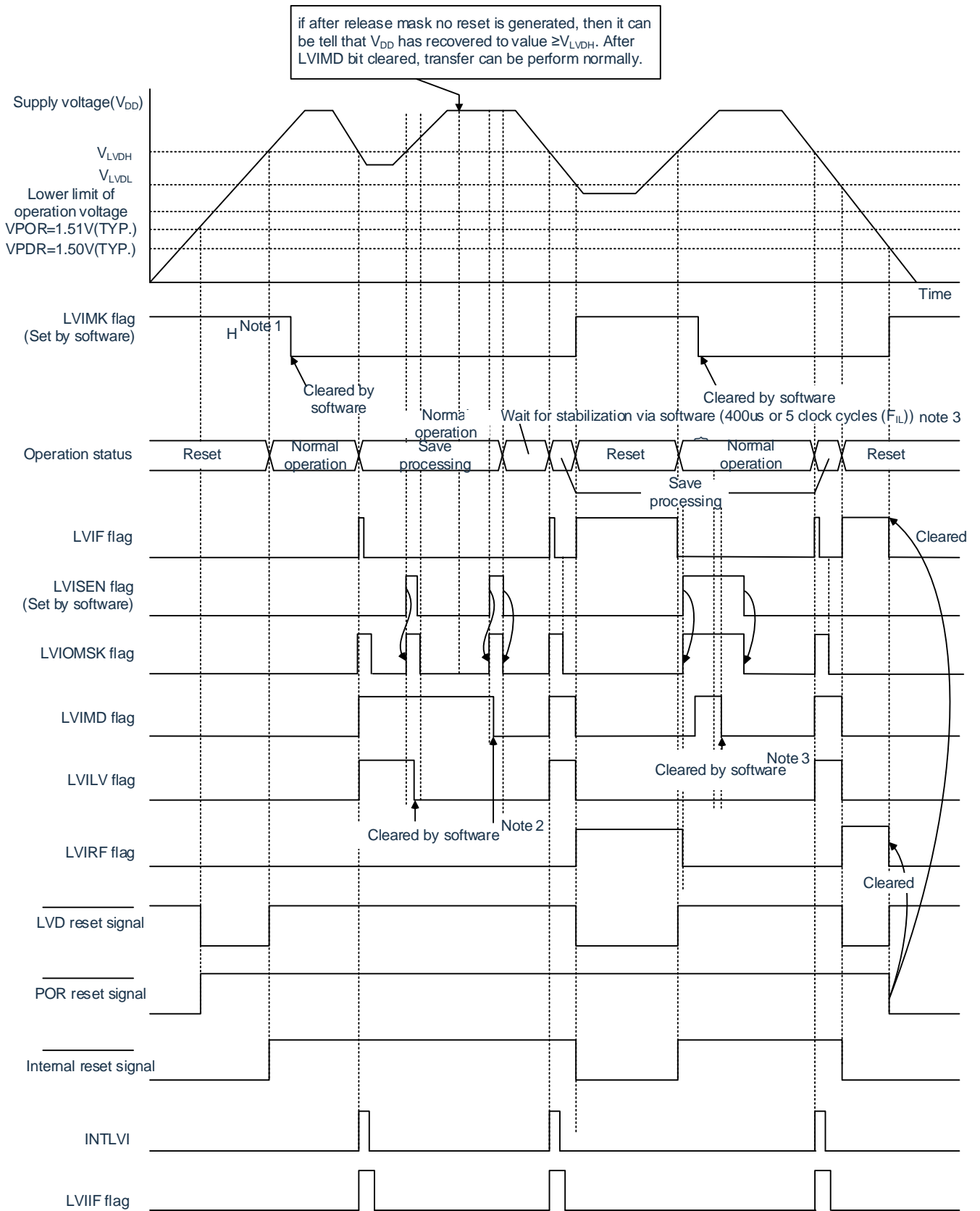
When power is turned on, the interrupt & reset mode (LVIMDS1, LVIMDS0=1, 0 of the option byte) maintains the internal reset state of the LVD until the power supply voltage (V_{DD}) exceeds the high voltage detection level (V_{LVDH}). If the supply voltage (V_{DD}) exceeds the high voltage detection level (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 25-7: Setting procedure for confirmation /reset of operating voltage" and "Figure 25-8: 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 25-6.

Figure 25-6 Reset & interrupt signal generation timing (LVIMDS1, LVIMDS0=1, 0) (1/2)



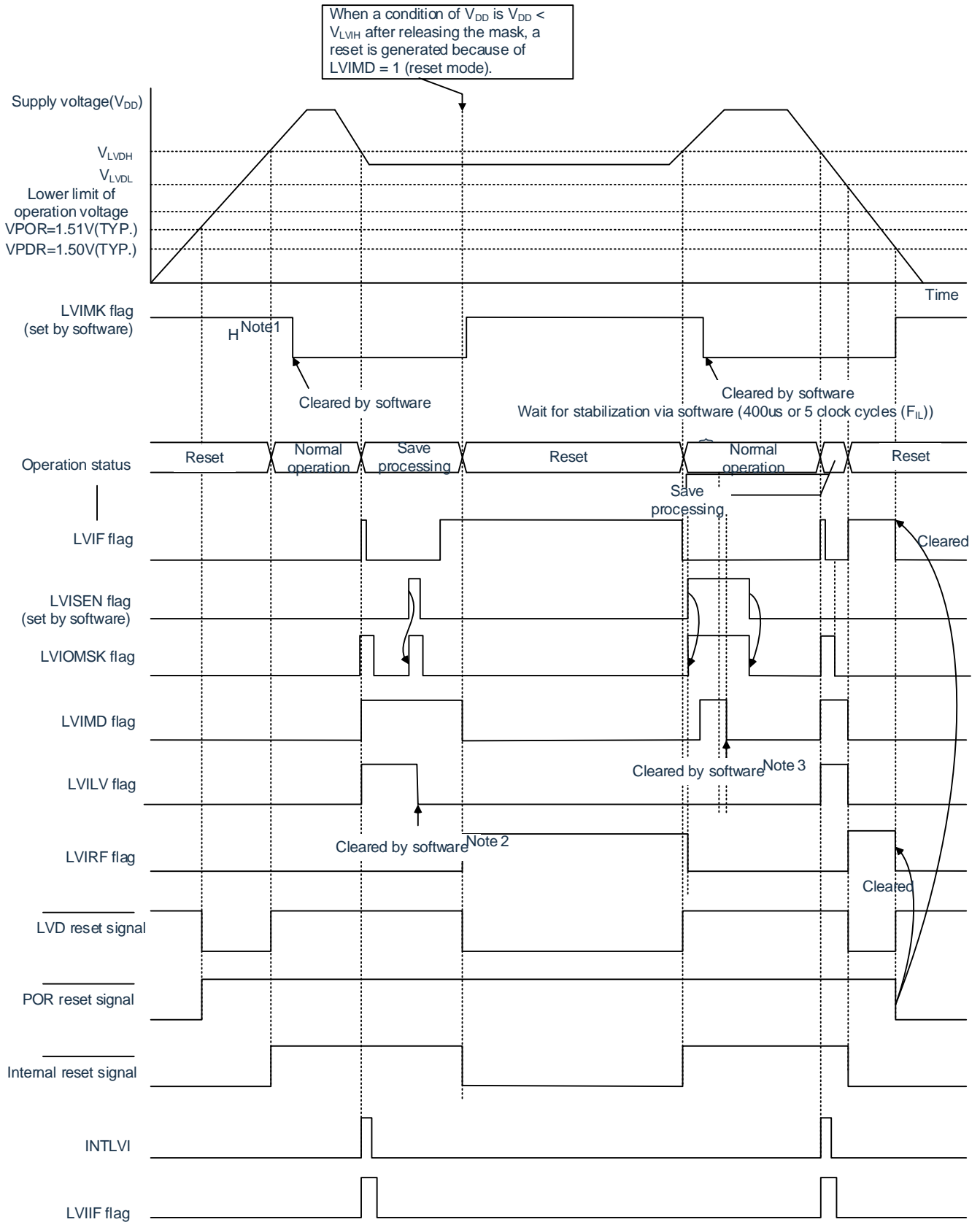
Note 1. After the reset signal is generated, the LVIMK flag becomes "1".

2. When using the interrupt & reset mode, you must follow "Figure 25-7: Setting procedure for confirmation /reset of operating voltage" after an interrupt occurs.
3. When using the interrupt&reset mode, you must follow the steps in "Figure 25-8: 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 25-6 Reset & interrupt signal generation timing (LVIMDS1, LVIMDS0=1, 0 of option byte) (2/2)



Note 1. After the reset signal is generated, the LVIMK flag becomes "1".

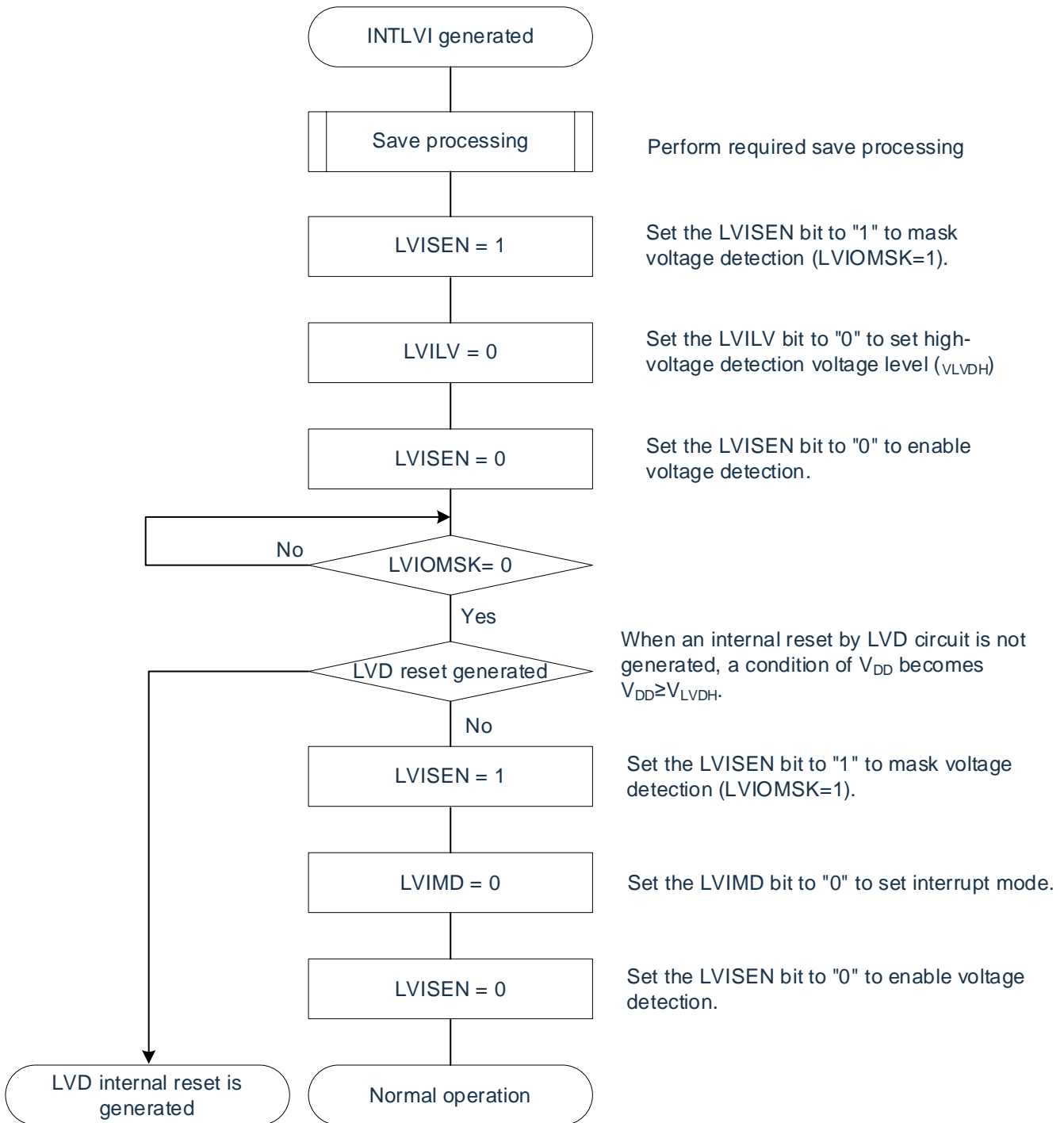
2. When using the interrupt & reset mode, you must follow "Figure 25-7: Setting procedure for confirmation /reset of operating voltage" after an interrupt occurs.

3. When using the interrupt&reset mode, you must follow the steps in "Figure 25-8: 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

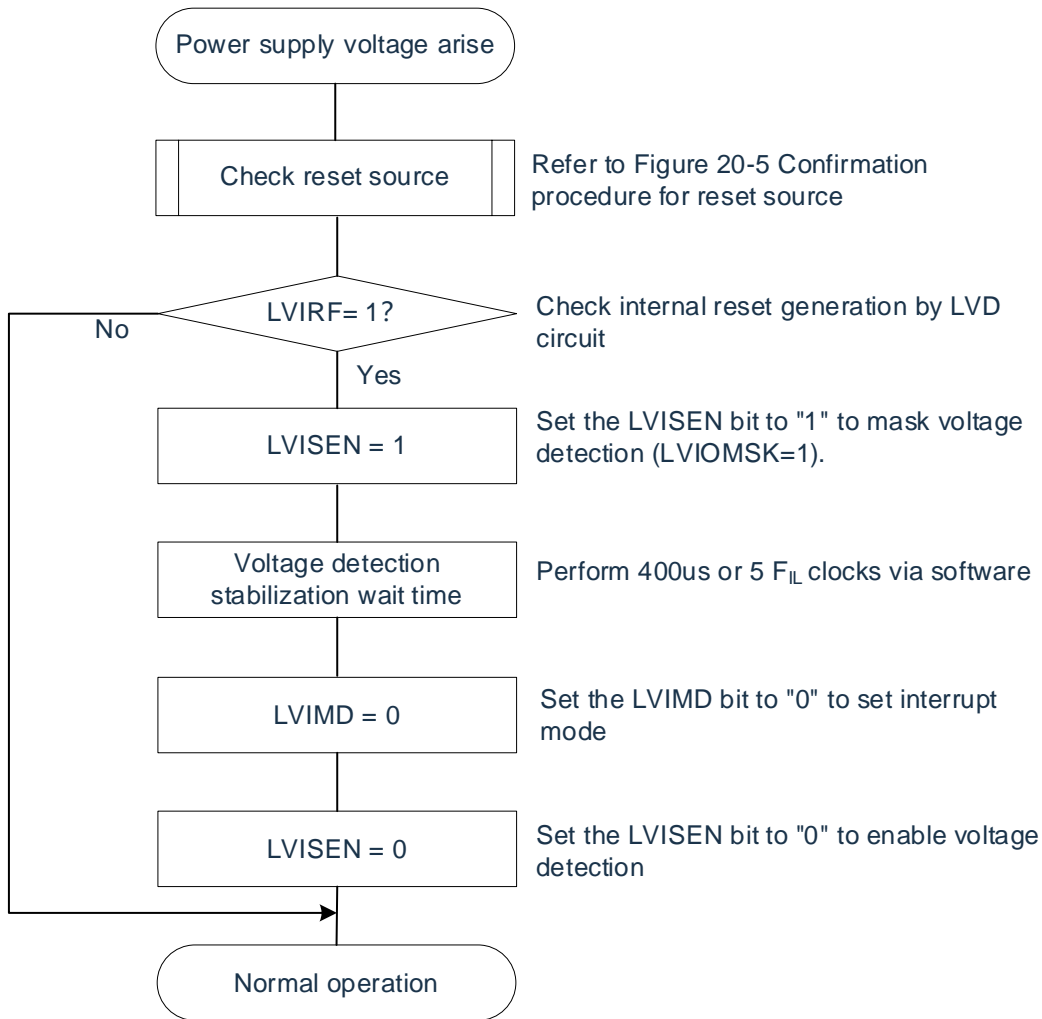
Figure 25-7 Setting procedure for confirmation /reset of operating voltage



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 25-8.

Figure 25-8 Initial setting procedure for interrupt & reset mode



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

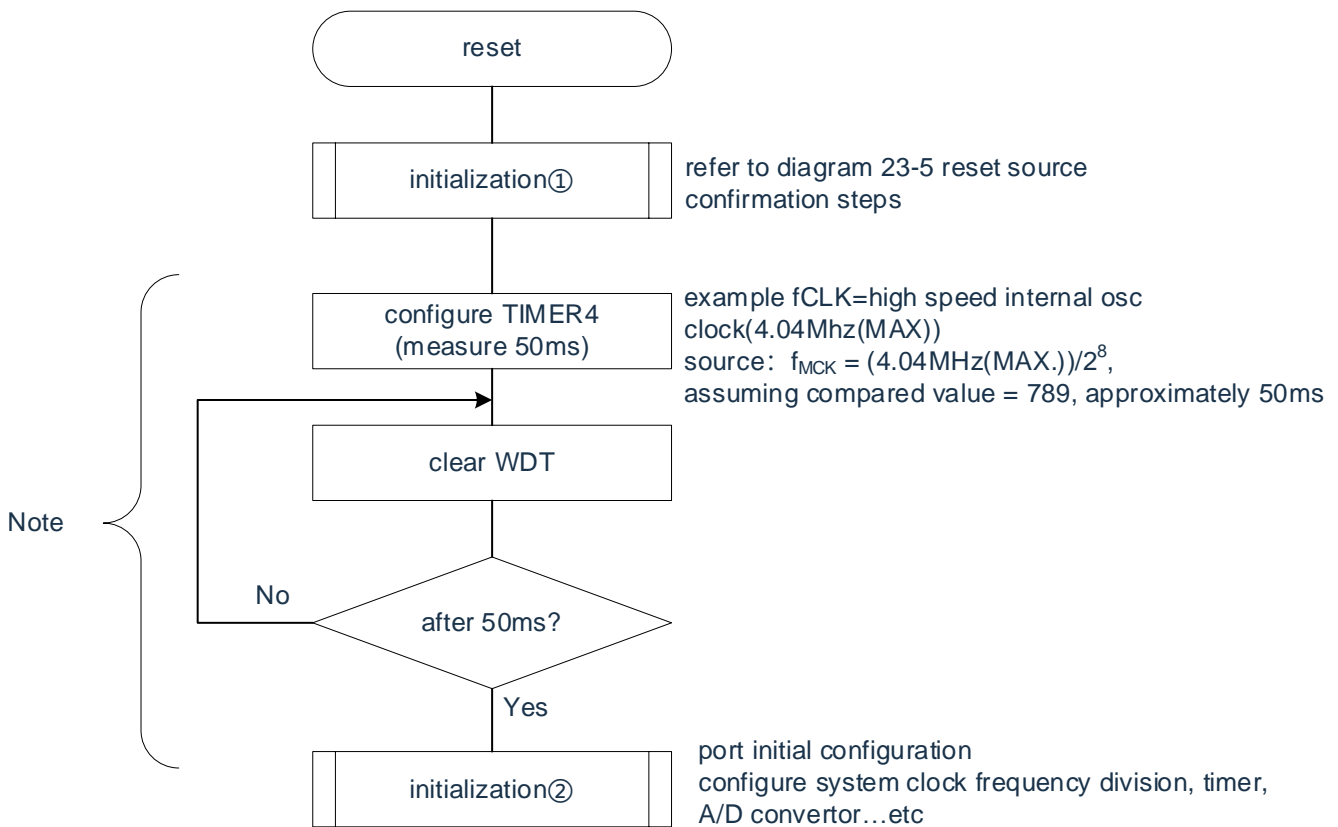
25.5 Cautions for voltage detection circuit

(1) Voltage fluctuation when power is supplied

For systems where the supply voltage (V_{DD}) fluctuates for a certain amount of time near the LVD sense voltage, it is possible to repeatedly enter the reset state and the reset release state. The following processing can be used to set the time of release reset to the start of the microcontroller operation arbitrarily.

< processing > after the reset is released, the initial setting of the port, etc. must be made by using the software counter of the timer and waiting for different supply voltage fluctuation times for each system.

Figure 25-9 Example of software processing when the supply voltage fluctuation near the LVD detection voltage does not exceed 50ms

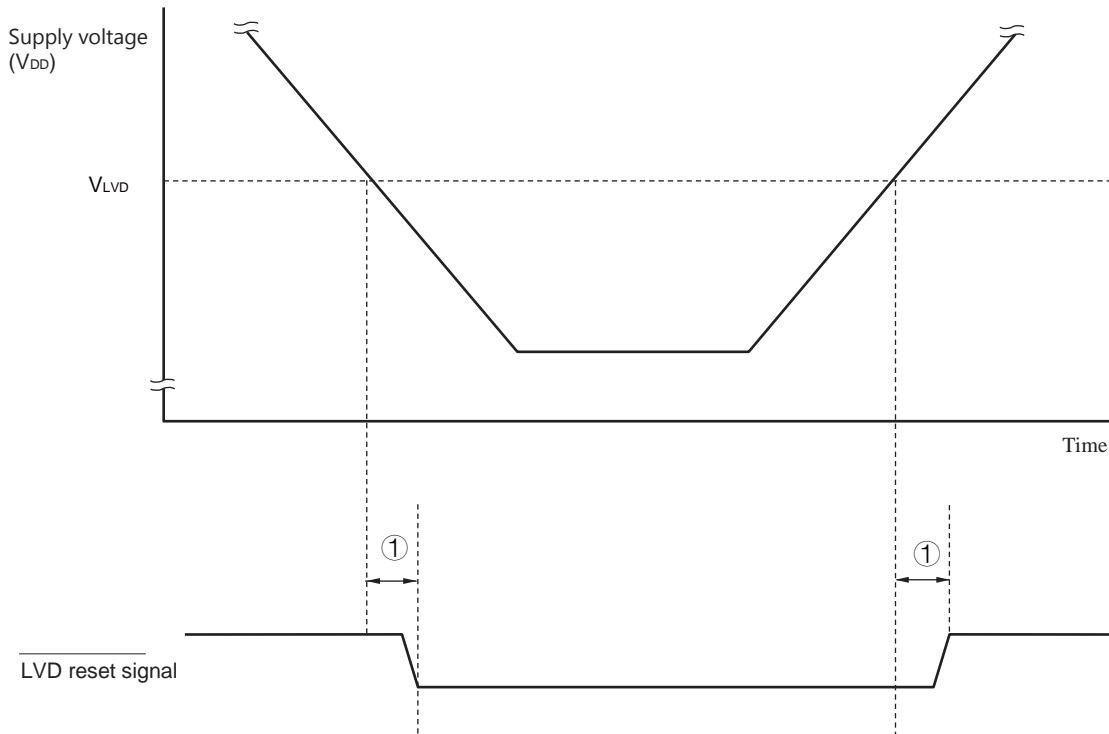


Note If a reset occurs again during this period, it is not transferred to initialization processing②.

(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 satisfied until the LVD reset is generated. Similarly, a delay occurs from LVD detection voltage (V_{LVD}) \leq supply voltage (V_{DD}) until LVD reset is released (refer to Figure 25-10).

Figure 25-10 Delay from generation of LVD reset source to generation or release of LVD reset



①: detection delay (300us(MAX.))

(3) When LVD is set to OFF, the power is turned on.

When setting the LVD to OFF, an external reset of the RESETB pin must be used.

During an external reset, a low level of at least 10us must be entered into the RESETB pin. If an external reset is performed when the supply voltage rises, the supply must be turned on after entering low on the RESETB pin and at least 10us low over the operating voltage range shown in the AC characteristics of the data sheet, and then input high.

(4) When LVD is turned OFF and, and the operating voltage drops with LVD interrupt mode is set.

With LVD set to OFF and set to LVD interrupt mode, if the operating voltage drops, it must be reset by transfer from deep sleep mode or external reset before the operating voltage drops below the operating voltage range shown in the AC characteristics of the data sheet. During restart operation, it must be confirmed that the supply voltage returns to the operating voltage range.

Chapter 26 Safety Function

26.1 Function of safety function

In response to IEC60730 and EC61508 safety standards, the BAT32G135 has the following built-in safety features.

The purpose of this safety function is to safely stop working when a fault is detected through self-diagnosis of the microcontroller.

(1) Flash CRC computing function (high-speed CRC, general-purpose CRC)

The CRC operation detects data errors in the flash memory. The following two CRCs can be used depending on the application and usage conditions.

- “High-speed CRC”... In the initialization program, the CPU can be stopped and the entire code flash area can be checked at high speed.
- “General CRC”... In CPU operation, it is not limited to the code flash memory area but can be used for multi-purpose inspection.

(2) RAM parity error detection function

When reading RAM data, parity errors are detected.

(3) SFR protection function

Prevents rewriting the SFR due to CPU runaway.

(4) Frequency detection function

Self-test CPU/peripheral hardware clock frequency can be performed using a general-purpose timer unit.

(5) A/D test function

It can perform 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 the A/D converter.

(6) 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.

26.2 Registers used by safety function

Each function of the safety function uses the following registers.

Register name	Safety function
<ul style="list-style-type: none"> • Flash CRC control register (CRC0CTL) • Flash CRC result register (PGCRCL) 	Flash CRC operation function (High-speed CRC)
<ul style="list-style-type: none"> • CRC input register (CRCIN) • CRC data register (CRCD) 	CRC operation function (General CRC).
<ul style="list-style-type: none"> • RAM parity error control register (RPECTL) 	RAM parity error detection function
<ul style="list-style-type: none"> • Special SFR protection control register (SFRGD) 	SFR protection function
<ul style="list-style-type: none"> • Timer input selection register 0 (TISO) 	Frequency detection function
<ul style="list-style-type: none"> • A/D test register (ADTES) 	A/D test function
<ul style="list-style-type: none"> • Port mode select register (PMS) 	Input/output pin digital output signal level detection function

The contents of each register are described in “26. 3 Operation of safety function”.

26.3 Operation of safety function

26.3.1 Flash CRC operation function (high-speed CRC)

The IEC60730 standard requires the confirmation of data in flash memory and recommends CRC as a means of confirmation. This high-speed CRC inspects the entire code flash memory area during the initialization (initialization) program.

The high-speed CRC stops the operation of the CPU and reads 32-bit data from the flash memory through 1 clock for operation. Therefore, it is characterized by a shorter time to complete the check (e.g., 64KB of flash: 512us@32MHz).

CRC generates a polynomial corresponding to CRC-16-CCITT's " $X^{16} + X^{12} + X^5 + 1$ ".

Operate on the MSB of bit31→bit0 first.

Remark The result is different because the general CRC is LSB-first.

Flash CRC control register (CRC0CTL)

This is a register that sets the operating control and operation range of a high-speed CRC operator. The CRC0CTL register is set via an 8-bit memory operation command. After a reset signal is generated, the value of this register becomes “00H”.

Figure 26-1 Format of flash 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	Operation control of high-speed CRC operators
0	Stop running.
1	Start the operation by executing the WFE instruction.

CRCCHK60	FEA2	FEA1	FEA0	Calculation range of high-speed CRC
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 ~ EFFFH(60K-4byte)

Note: Bit3~6 must be set to 0.

Remark The expected value of the CRC operation used for comparison must be stored in the last 4 bytes of flash memory in advance, so the operation range is subtracted from the range of 4 bytes.

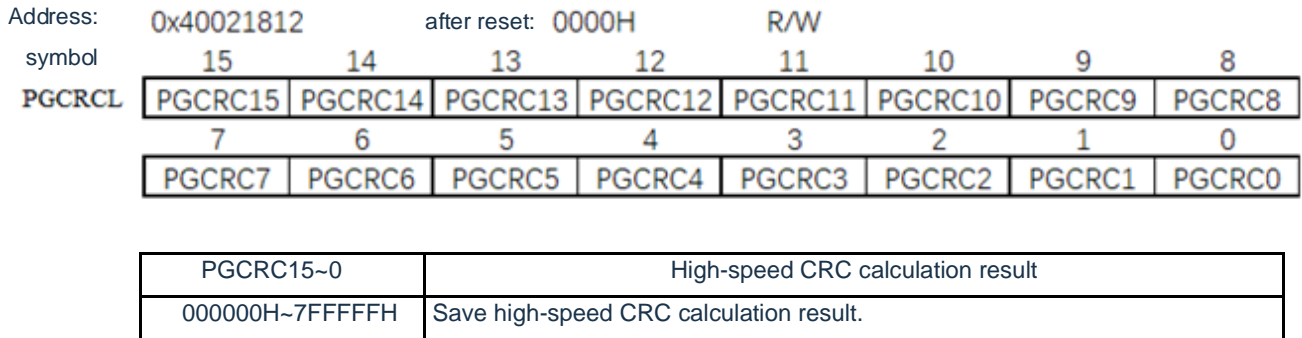
26.3.1.1 Flash CRC calculation result register (PGCRCL)

This is a register that holds the results of high-speed CRC operations.

The PGCRCL register is set via a 16-bit memory operation command.

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

Figure 26-2 Format of flash CRC calculation result register (PGCRCL)

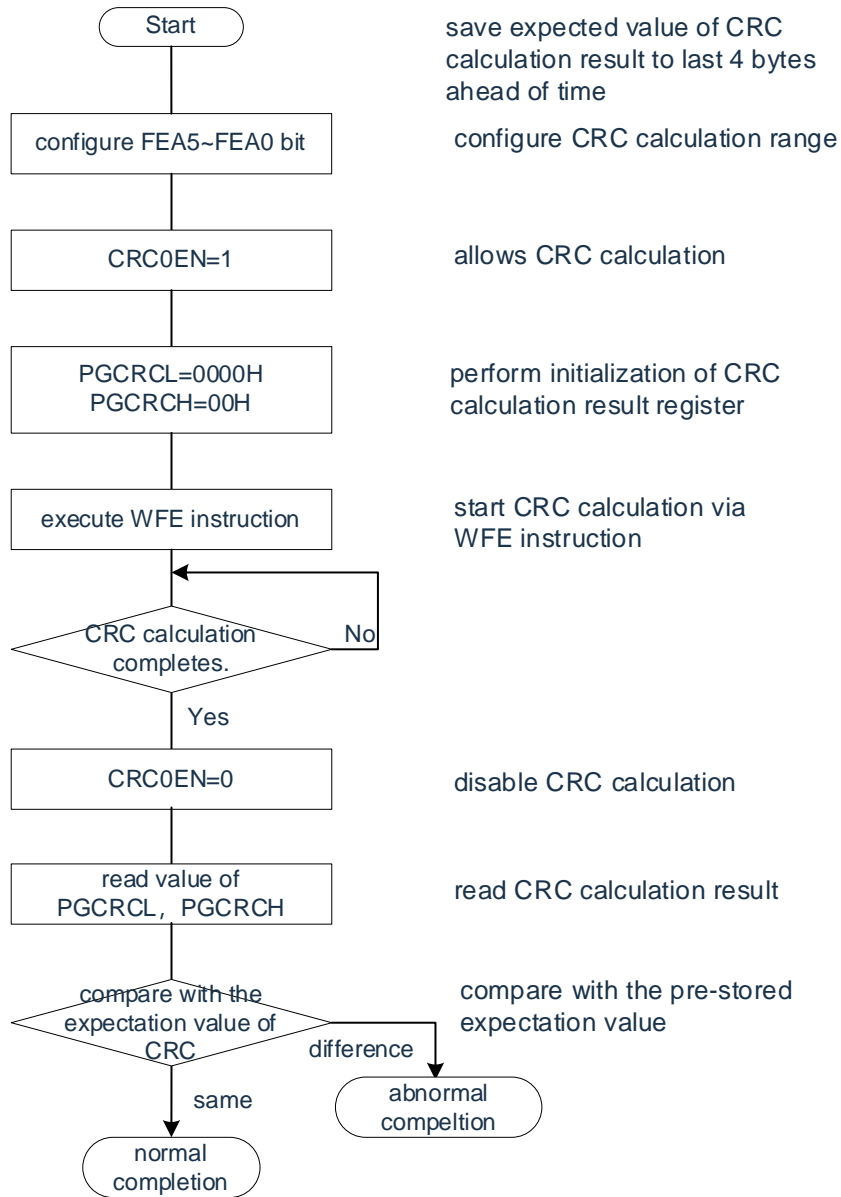


Notice The PGCRCL register can only be written if the CRC0EN (bit7 of the CRC0CTL register) bit is “1”.

A flowchart of the flash CRC operation function (high-speed CRC) is shown in Figure 26-3.

<Operation flow>

Figure 26-3 Flow chart of flash CRC operation function (high-speed CRC)



Notice 1. Only the code flash memory is an object for CRC operations.

2. The expected value of the CRC operation must be saved in the area behind the operation range in the code flash.

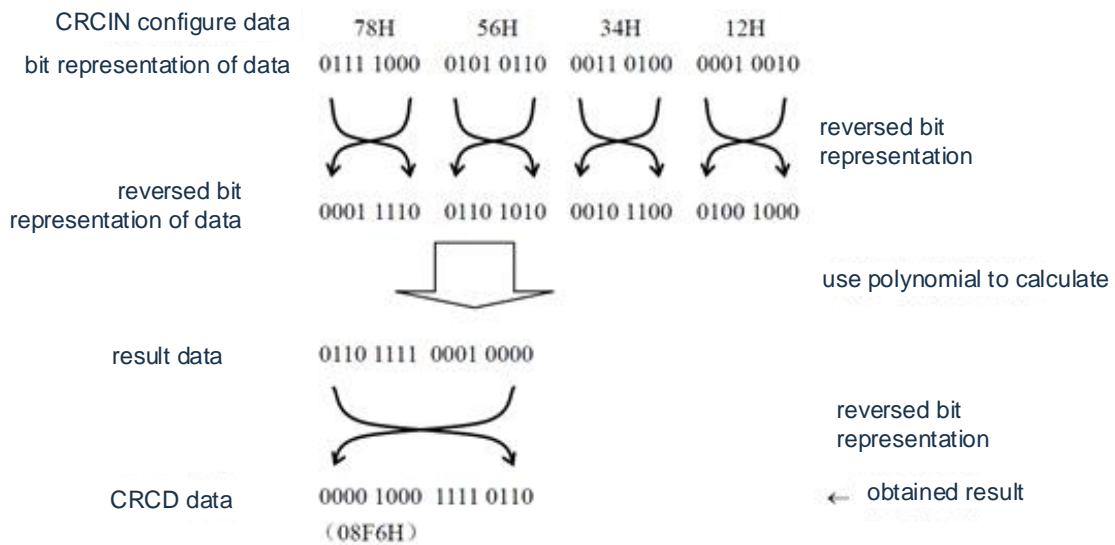
26.3.2 CRC operation function (general CRC)

In order to ensure safety during operation, the IEC61508 standard requires that the data need to be confirmed even in CPU operation.

This generic CRC can be used as a peripheral function for CRC operations in CPU operation. Generic CRC is not limited to code flash areas but can be used for multi-purpose inspections. The data to be confirmed is specified through the software (user program). The CRC operation function in sleep mode can only be used during DMA transmission.

CRC arithmetic functions can be used in either the main system clock operation mode or the secondary system clock operation mode.

CRC generates polynomials using CRC-16-CCITT's " $X^{16}+X^{12}+X^5+1$ ". Because the communication is considered to be carried out in LSB first, the calculation is performed after the bit order of the input data is reversed. For example, in the case of sending data "12345678H" from the LSB, follow the requirements of "78H", "56H", "34H", "34H" The "12H" sequence writes the value to the CRCIN register, and the value of "08F6H" is obtained from the CRCD register. This is the result of CRC operations for the following bit order after inverting the bit order of the data "12345678H".



Notice During the execution of the program, because the modulator rewrites the set line of the software breakpoint as a breakpoint instruction, if you set the software breakpoint in the object area of the CRC operation, the CRC operation result is different.

26.3.2.1 CRC input register (CRCIN)

This is the 8-bit register that sets the CRC calculation data for the general-purpose CRC. The range that can be set is “00H~FFH”.

The CRCIN registers are set via 8-bit memory operation instructions. After the reset signal is generated, the value of this register becomes “00H”.

Figure 26-4 Format of CRC input register (CRCIN)

Address: 400433ACH After reset: 00H R/W



bit7~0	Function
00H~FFH	Data input

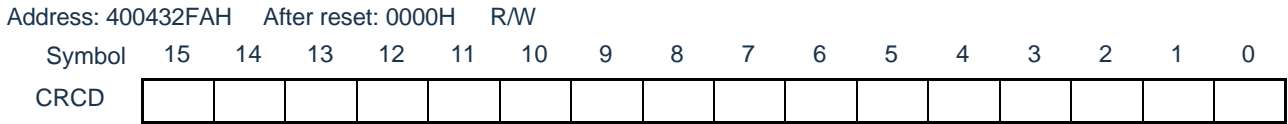
26.3.2.2 CRC data register (CRCD)

This is a register that holds the results of a general-purpose CRC operation. The range that can be set is "0000H~FFFFH".

After writing the CRCIN register, a CPU/peripheral hardware clock (f_{CLK}) is passed to save the CRC operation results to the CRCD Register. The CRCD registers are set via 16-bit memory operation instructions.

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

Figure 26-5 Format of CRC data register (CRCD)

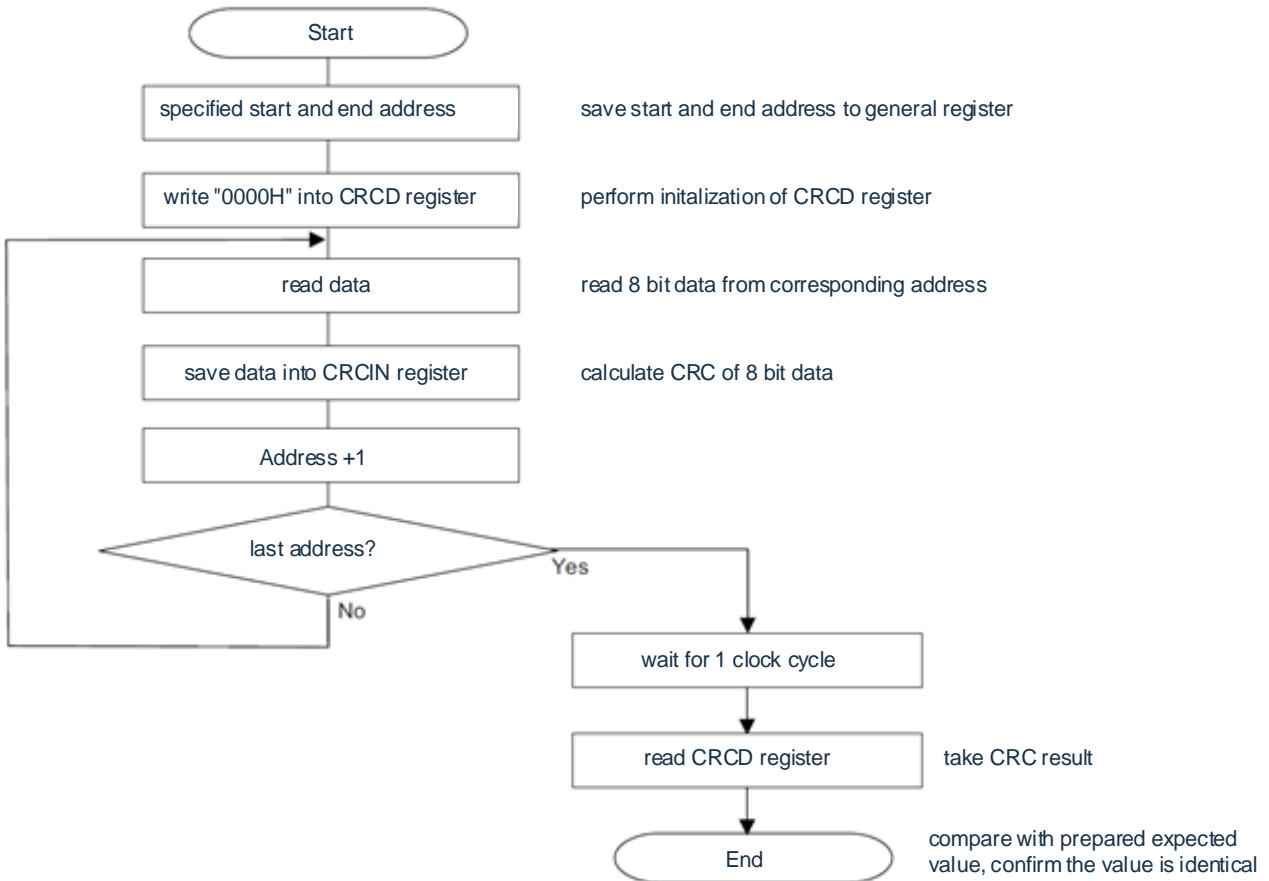


Notice1. To read the write value of the CRCD register, the CRCD register must be read before the CRCIN register is written.

2. If the write operation of the CRCD register competes with the saving of the operation result, the write operation is ignored.

<Operation process>

Figure 26-6 Flowchart of CRC operation function (general CRC)



26.3.3 RAM parity error detection function

The IEC60730 standard requires confirmation of RAM data. Therefore, the BAT32G135's RAM appends 1-bit parity bit every 8 bits. The RAM parity error detection function appends parity bits when writing data, checks parity bits when reading data, and can produce a reset when parity errors occur.

26.3.3.1 RAM parity check error control register (RPECTL)

This register controls the parity error acknowledge bit and the reset due to a parity error.

The RPECTL registers are set via 8-bit memory operation instructions.

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

Figure 26-7 Format of RAM parity error control register (RPECTL)

Address: 40020425H	After reset: 00H	R/W						
Symbol	7	6	5	4	3	2	1	0
RPECTL	RPERDIS	0	0	0	0	0	0	RPEF

RPERDIS	Parity check error reset mask flag
0	Enables generation of a parity error reset.
1	Disables generation of a parity error reset.

RPEF	Parity error status flag
0	No parity errors occurred.
1	A parity error occurred.

Notice Parity bits are appended when writing data and checked when reading data.

Therefore, to allow RAM parity error reset (RPERDIS=0), the "RAM area used" must be paired when accessing the data and before reading the data Initialize.

Because it is running on a pipeline, the CPU performs a pre-read, and a RAM parity error may occur due to the uninitialized RAM area before reading the RAM area used. Therefore, to allow the generation of RAM parity error reset (RPERDIS=0), the instructions must be executed from the RAM area to the "RAM area used." +10 bytes" area is initialized.

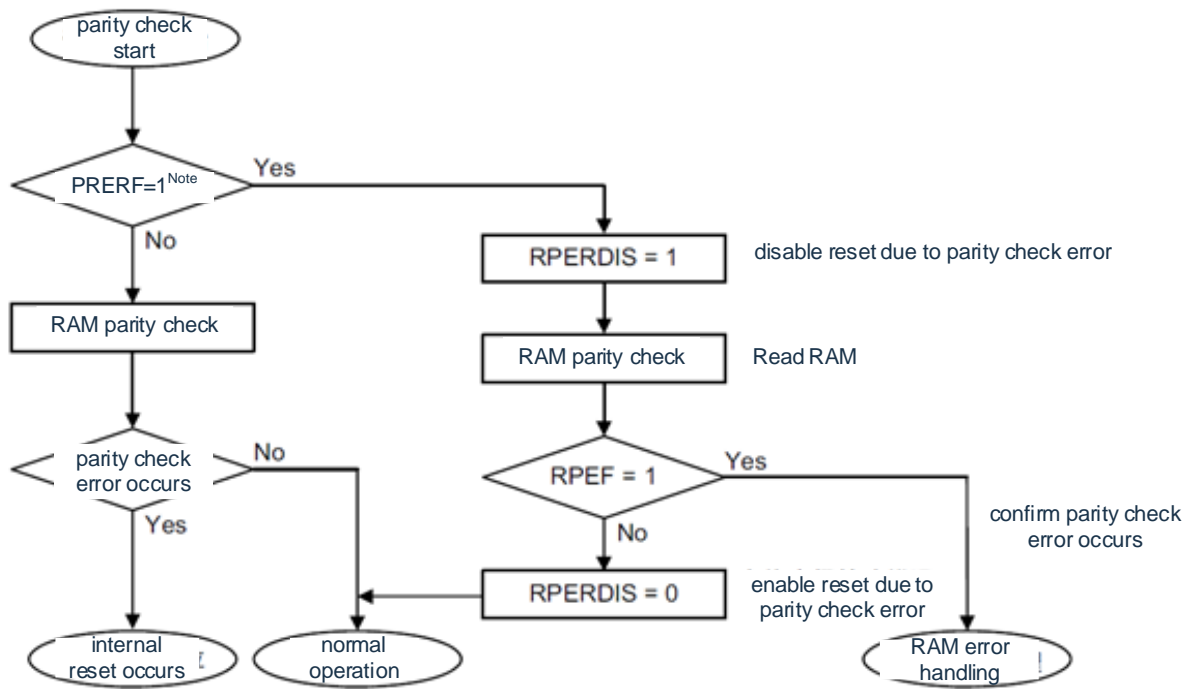
Remark 1. The initial state is to allow for parity error reset (RPERDIS=0).

2. Even if the parity error reset is disabled (RPERDIS=1), the RPEF flag is set to "1" in the event of a parity error. If the RPEF bit is set to allow parity error reset (RPERDIS=0) in the state of "1", the RPERDIS is cleared "0" A parity error is generated when the reset occurs.

3. Set the RPEF flag of the RPECTL register to "1" due to RAM parity error, and RPEF is reset by writing "0" or all the reset sources Flag clear "0". When the RPEF flag is "1", the RPEF flag remains in the state of "1" even if the RAM without parity errors is read.

4. The range of RAM parity detection does not include general-purpose registers.

Figure 26-8 Flow of RAM parity check



Note For confirmation of the internal reset of RAM parity errors, see Chapter 23 Reset Function.

26.3.4 SFR protection function

In order to ensure safety during operation, the IEC61508 standard requires that even if the CPU is out of control, important SFRs need to be protected from overriding the SFR protection function for the protection of port functions, interrupt functions, clock control functions, voltage detection circuitry and RAM. The parity error detection function controls the data of the register.

If the protection function is set to SFR, the write operation of the protected SFR is invalid, but it can be read normally.

26.3.4.1 SFR protection control register (SFRGD)

This register controls whether the SFR protection function is effective.

The SFR protection function uses the GCOMP bit, GPORT bit, GINT bit, and GCSC bit.

The SFRGD register is set via an 8-bit memory operation command.

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

Figure 26-9 Format of SFR protection control register (SFRGD)

Address: 40040478H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
SFRGD	0	0	0	0	GCOMP	GPORT	GINT	GCSC
GCOMP	Protection of control registers for comparator functions							
0	Invalid. Can read and write the control registers for the comparator function.							
1	Valid. Write operation of the control register of the comparator function is invalid, and it can be read. [Protected SFR] COMPPMDR, COMPFIR, COMPOCR, CVRCTL, CxRVM, PGAxCTL, PGASHMD, CMPSELx							
GPORT	Protection of control registers for port functions							
0	Invalid. Control registers that can read and write port functions.							
1	Valid. The port function of the control register is invalid and can be read. [Protected SFR] PMxx, PUxx, PDxx, POMxx, PMCxx, PxxCFG, PIORx ^{Note}							
GINT	Interrupt function register protection							
0	Invalid. Control registers that can read and write interrupts.							
1	Valid. The write operation of the control register of the interrupt function is invalid and can be read. [Protected SFR] IFxx, MKxx, PRxx, EGPx, EGNx							
GCSC	Protection of control registers for clock control functions, voltage detection circuits, and RAM parity error detection functions							
0	Invalid. Control registers that can read and write clock control functions, voltage detection circuitry, and RAM parity error detection functions.							
1	Valid. The write operation of the control register of the clock control function, voltage detection circuit, and RAM parity error detection function is invalid and can be read. [Protected SFR] CMC, CSC, OSTs, CKC, PERx, OSMC, LVIM, LVIS, RPECTL							

Note Pxx (port register) is not protected.

26.3.5 Frequency detection function

The IEC60730 standard requires confirmation of whether the oscillation frequency is normal.

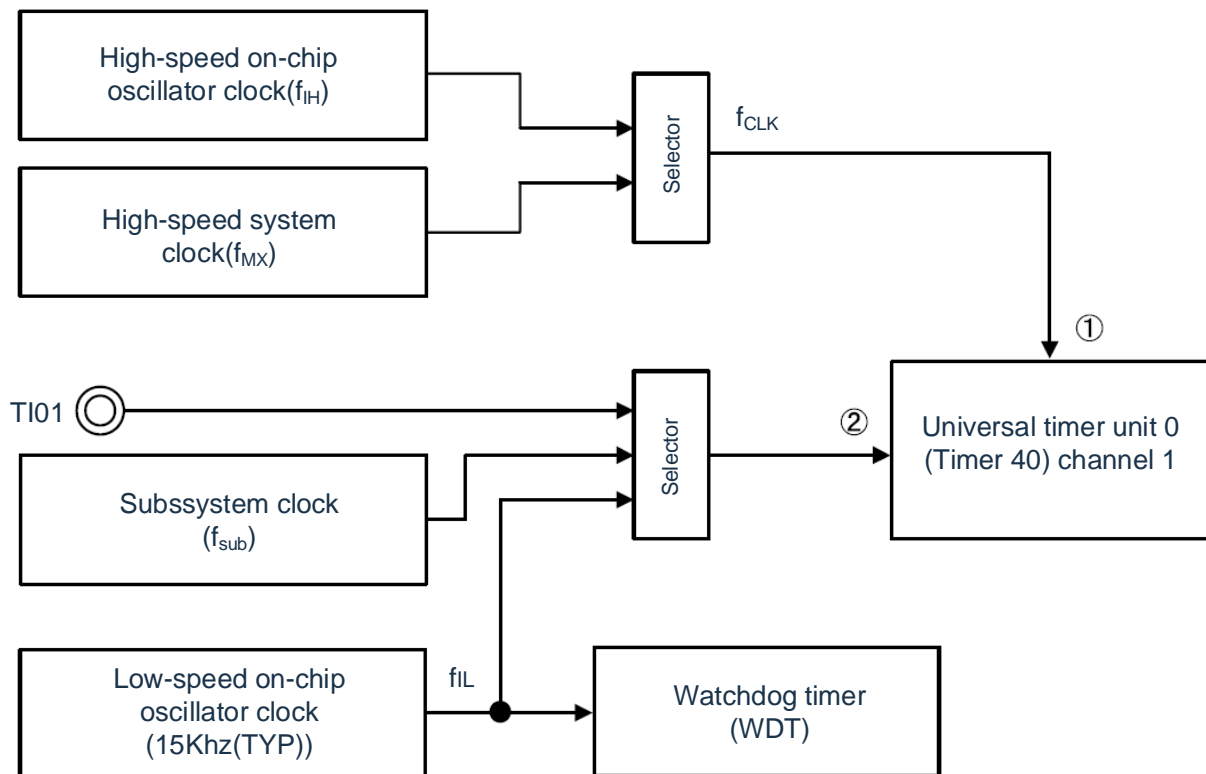
The frequency detection function uses the clock frequency (f_{CLK}) of the CPU/peripheral hardware and can determine whether the ratio relationship between the two clocks is correct by measuring the Timer40 channel 1 input pulse.

However, if 1 clock or 2 clocks stop oscillating, the ratio of 2 clocks cannot be determined.

<Clock to compare>

- ① Clock frequency of CPU/peripheral hardware (f_{CLK}):
 - High-speed on-chip oscillator clock (f_{IH})
 - High-speed system clock (f_{MX})
- ② Timer40 channel 1 input:
 - Timer input for channel 1 (TI01)
 - Low-speed on-chip oscillator clock (f_{IL} : 15kHz(TYP.))
 - Subsystem clock (f_{SUB})^{Note}

Figure 26-10 Structure of frequency detection function



When the measurement result of the input pulse interval is an outlier, it can be judged as “clock frequency anomaly”. For the measurement method of the input pulse interval, refer to “5.8.4”.

Note Only products with built-in sub-system clocks can be selected.

26.3.5.1 Timer input selection register 0 (TIS0)

Refer to Section 5.3.8 for register descriptions.

26.3.6 A/D test function

The IEC60730 standard requires A/D converter testing. This A/D test function verifies that the A/D converter is functioning properly by performing A/D conversion of the A/D converter's positive (+) reference voltage, negative (-) reference voltage, analog input channel (ANI), temperature sensor output voltage, and internal reference voltage.

The analog multiplexer can be verified by following these steps:

- ① The ANIx pin is selected as the A/D conversion object through the ADTES register (ADTES2, ADTES1, ADTES0=0, 0, 0).
- ② A/D conversion of the ANIx pin (conversion result 1-1).
- ③ The negative (-) reference voltage of the A/D converter is selected as the A/D conversion object through the ADTES register (ADTES2, ADTES1, ADTES0=0, 0, 1).
- ④ A/D conversion of the negative (-) reference voltage of the A/D converter (results 2-1).
- ⑤ The ANIx pin is selected as the A/D conversion object through the ADTES register (ADTES2, ADTES1, ADTES0=0, 0, 0).
- ⑥ A/D conversion of the ANIx pin (conversion results 1-2).
- ⑦ The positive (+) reference voltage of the A/D converter is selected from the ADTES register as the A/D conversion object (ADTES2, ADTES1, ADTES0=1, 0, 1).
- ⑧ A/D conversion of the positive (+) reference voltage of the A/D converter (conversion results 2-2).
- ⑨ The ANIx pin is selected as the A/D conversion object through the ADTES register (ADTES2, ADTES1, ADTES0=0, 0, 0).
- ⑩ A/D conversion of the ANIx pin (conversion result 1-3).
- ⑪ Verify that the Conversion Results 1-1, Conversion Results 1-2, and Conversion Results 1-3 are the same.
- ⑫ Confirm that the A/D conversion result of "Conversion Result 2-1" is all "0" and the A/D of "Conversion Result 2-2" The result of the conversion is all "1". With these steps, you can select an analog multiplexer and verify that the wiring is not broken.

Remark1. During the conversion of ①~⑩, if the analog input voltage is variable, other methods must be used to confirm the analog multiplexer.

2. The conversion result contains errors, so the error must be properly considered when comparing the conversion results.

26.3.6.1 A/D test registers (ADTES)

This register selects the A/D converter's positive (+) reference, negative (–) reference, analog input channel (ANLxx), temperature sensor output voltage, and internal reference voltage (1.45V). as an A/D conversion object.

When used as an A/D test function, the following settings are made:

- When measuring the zero scale, select the negative (–) reference voltage as the A/D conversion object.
- When measuring the full scale, select the positive (+) reference voltage as the A/D conversion object.

Refer to 11.2.10 for the A/D register description.

26.3.6.2 Analog input channel specification register (ADS)

This register specifies the input channel for the analog voltage of the A/D conversion.

To measure the ANLxx, temperature sensor output or internal reference voltage (1.45V) via the A/D test function, the A/D test register (ADTES) must be set to “00H”.

Refer to 11.2.7 for the register description.

26.3.7 Input/output pin digital output signal level detection function

The IEC60730 standard requires confirmation of proper I/O functionality.

The input/output pin digital output signal level detection function reads the digital output level of the pin when the pin is in output mode.

26.3.7.1 Port mode selection register (PMS)

This register selects whether to read the value of the port's output latch or the output level of the read pin when the pin is in output mode (the PM_mn bit of the port mode register (PM_m) is "0").

The PMS register is set via an 8-bit memory operation instruction.

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

Figure 26-11 Format of port mode selection 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 read data when the pin is in output mode
0	Read the value of the P _m n register.
1	Read the digital output level of the pin.

Notice1. For pins that use the pulse output forced truncation function of timer M to make the pin into a high impedance state, if the digital output level of the pin is read, the read value is "0".

Remark m=0~7, 12~14
n=0~7

26.3.8 Product unique identification register

The unique identification of the product is ideally suited for:

- Used as a serial number (e.g. USB character serial number or other terminal applications)
- 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 code security in flash memory.
- Used to activate the bootstrap process with a safety mechanism

The reference number provided by the 128-bit unique product identifier is unique to any BAT32 microcontroller and is unique in any case. Under no circumstances can the user modify this identity.

Base address: 0x0050_084C

Address offset: 0x00

Read-only, and its value is programmed at the factory.

U_ID[31:0]

Address offset: 0x04

Read-only, and its value is programmed at the factory.

U_ID[63:32]

Address offset: 0x08

Read-only, and its value is programmed at the factory.

U_ID[95:64]

Address offset: 0x0C

Read-only, and its value is programmed at the factory.

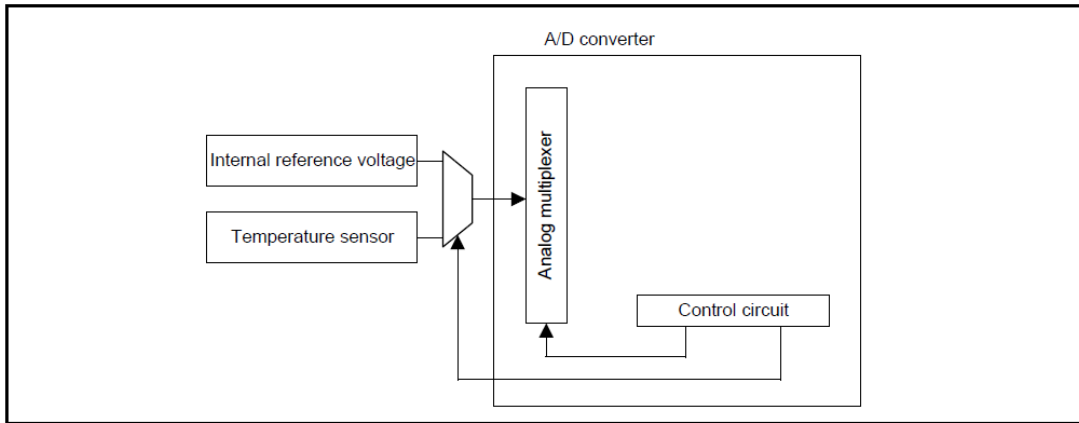
U_ID[127:96]

Chapter 27 Temperature Sensor

27.1 Function of temperature sensor

The on-chip temperature sensor measures and monitors the core temperature of the product, thus ensuring the 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 voltage and temperature. Its output voltage is supplied to the ADC for conversion. Figure 27-1 shows a block diagram of a temperature sensor.

Figure 27-1 Block diagram of temperature sensor



27.2 Registers of temperature sensor

27.2.1 Temperature sensor calibration data register TSN25

Address: 0x0050_066C

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	After reset	R/W
TSN25	-	-	-	-	TSN25[11:0]											-	R	

A read-only register that records calibration data for the temperature sensor1 is automatically loaded when power is turned on or reset starts, and each chip has its own calibration data.

27.2.2 Temperature sensor calibration data register TSN85

Address: 0x0850_0C68

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	After reset	R/W
TSN85	-	-	-	-	TSN85[11:0]											-	R	

A read-only register that records calibration data for temperature sensors2 is automatically loaded when power is turned on or reset starts, with each chip having its own calibration data.

27.3 Temperature sensor usage instructions

27.3.1 Principle of temperature sensor

The temperature (T) is proportional to the sensor voltage output (Vs), so the temperature is calculated as follows:

$$T = (V_s - V_1) / \text{slope} + T_1$$

T: Measured temperature (°C)

Vs: Output voltage of the temperature sensor during temperature measurement (V)

T1: Temperature at the first point for experimental measurements (°C)

V1: Voltage output when the temperature sensor measures T1 (V)

T2: Temperature at the second point for experimental measurements (°C)

V2: Voltage output when the temperature sensor measures T2 (V)

Slope: Temperature slope of the temperature sensor (V/°C), slope = $(V_2 - V_1) / (T_2 - T_1)$

Different sensors have different characteristics, so we recommend measuring the following two different sample temperatures:

- 1、 The A/D converter is used to measure the voltage V1 output by the temperature sensor at temperature T1.
- 2、 The A/D converter is used to measure the voltage V2 output by the temperature sensor at the second temperature T2.
- 3、 The temperature slope (slope = $(V_2 - V_1) / (T_2 - T_1)$) is calculated from the two results
- 4、 Subsequently, the temperature is obtained by substituting the slope into the formula for the temperature characteristics ($T = (V_s - V_1) / \text{slope} + T_1$).

27.3.2 How to use temperature sensor

Method 1: For this product, the TSN25 register stores the voltage conversion value (CAL25) of the temperature sensor measured under the conditions of $T_a=25^{\circ}\text{C}$ and $V_{CC}=3.0\text{V}$. The TSN85 register stores the voltage conversion values of the temperature sensor measured at $T_{aj}=85^{\circ}\text{C}$ and $V_{CC}=3.0\text{V}$ (CAL85). Using these two sets of values, the temperature slope can be calculated:

$$\text{slope} = (V_2 - V_1) / (85 - 25).$$

$$V_1 = 3.0 \times \text{CAL25} / 4096 \text{ [V]}$$

$$V_2 = 3.0 \times \text{CAL85} / 4096 \text{ [V]}$$

Using the above results, the temperature can be calculated according to the following formula:

$$T = (V_s - V_1) / \text{slope} + 25 \text{ [}^{\circ}\text{C]}$$

T: Measured temperature ($^{\circ}\text{C}$)

V_s : Output voltage of the temperature sensor obtained using an A/D converter at T temperature (V)

Method 2: If you use the temperature slope given in “Electrical Characteristics”, you can directly calculate the measured temperature using the following formula:

$$T = (V_s - V_1) / \text{slope} + 25 \text{ [}^{\circ}\text{C]}$$

Note: This method produces a temperature that is less accurate than Method 1.

Chapter 28 Option Byte

28.1 Function of option byte

BAT32G135 flash memory 000C0H to 000C3H, 500004H to 500005H are option byte area.

The option byte consists of the user option byte (000C0H to 000C2H) and the flash data protection option byte (000C3H, 500004H to 500005H). When the power is turned on or reset, the option byte is automatically used to set the specified function. The following functions must be set by the option byte when using this product. The initial value of the bit that has no configuration function cannot be changed. To use the boot swapping function in the self-programming process, the same value must be set for 010C0H to 010C3H as for 000C0H to 000C3H because 000C0H to 000C3H are replaced by 010C0H to 010C3H.

Notice Regardless of whether or not to use each feature, the option byte must be set.

28.1.1 User option bytes (000C0H~000C2H/010C0H~010C2H)

(1) 000C0H/010C0H

- Operation of watchdog timer
 - Enables or disables operation of the counter.
 - Enables or stops operation of the counter in sleep/deep sleep mode.
- Setting of overflow time of the watchdog timer
- Setting of the window opening period of the watchdog timer
- Setting the interval interrupt of the watchdog timer
 - Use or do not use interval interrupts.

Notice During boot swapping, 000C0H is replaced by 010C0H, so 010C0H must be set to the same value as 000C0H.

(2) 000C1H/010C1H

- Setting of LVD operation mode
 - Interrupt & reset mode
 - Reset mode
 - Interrupt mode
 - LVD is OFF (using an external reset input from the RESETB pin).
- Setting of LVD detection level (V_{LVDH} , V_{LVDL} , V_{LVD})

Notice1. When the supply voltage rises, the reset state must be maintained by voltage detection circuit or external reset 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 through deep sleep mode transfer, voltage detection circuitry, or external reset before the supply voltage drops below the operating voltage range.

The operating voltage range depends on the setting of the user option byte (000C2H/010C2H).

2. During the boot swapping, 000C1H is replaced by 010C1H, so 010C1H must be set to the same value as 000C1H.

(3) 000C2H/010C2H

- Setting of high-speed on-chip oscillator frequency
 - Select from 1MHz~32MHz, 48MHz, 64MHz.

Notice During the boot swapping, 000C2H is replaced by 010C2H, so 010C2H must be set to the same value as 000C2H.

28.1.2 Flash data protection option bytes (000C3H/010C3H, 500004H~500005H)

Control of flash data protection during on-chip debugging

Level0: Allows read/write/erase operations on flash data via debugger.

Level1: Allows chip full erase operations on flash data via debugger, does not allow read/write operations.

Level2: Manipulation of flash data via debugger is not allowed.

Control of the boot swapping function

- Disables or enables boot swapping function.

Notice During the boot swapping, 000C3H is replaced by 010C3H, so 010C3H must be set to the same value as 000C3H.

28.2 Format of user option byte

Figure 28-1 Format of user option byte (000C0H/010C0H)

Address: 000C0H/010C0H^{Note1}

Symbol	7	6	5	4	3	2	1	0
	WDTINT	WINDOW1	WINDOW0	WDTON	WDCS2	WDCS1	WDCS0	WDSTBYON

WDTINT	Interval interruption of watchdog timer use/non-use
0	Interval interrupts are not used.
1	When 75% of the overflow time is reached $+1/2f_{IL}$, an interval interrupt occurs.

WINDOW1	WINDOW0	Watchdog timer open period ^{Note2}
0	-	Disable settings
1	0	75%
1	1	100%

WDTON	Counter operation control of watchdog timer
0	Disables counter operation (stops counting after resetting is released).
1	Enables counter operation (starts counting after resetting is released).

WDCS2	WDCS1	WDCS0	Overflow time of the watchdog timer ($f_{IL}=20\text{kHz}(\text{MAX.})$)
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	Watchdog timer counter operation control (sleep mode)
0	Stops counter operation ^{Note 2} in sleep mode.
1	Enables counter operation in sleep mode.

Note 1. During boot swapping, 000C0H is replaced by 010C0H, so 010C0H must be set to the same value as 000C0H.

2. When the WDSTBYON bit is "0", regardless of the values of the WINDOW1 bit and WINDOW0 bit, the window is opened 100% of the time.

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

Figure 28-2 Format of user option bytes (000C1H/010C1H) (1/4)

 Address: 000C1H/010C1H^{Note}

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

• LVD setting (interrupt & reset mode)

Detection voltage			Setting value of the option byte						
VLVDH		VLVDL	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	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			
—			Values other than the above are prohibited.						

Note During the bootstrap swappinf, 000C1H is replaced by 010C1H, so 010C1H must be set to the same value as 000C1H.

Notice Bit4 must be set to “1”.

Remark 1. For more information about LVD circuits, refer to Chapter 25, Voltage Detection Circuit.

2. The detection voltage is TYP. value. For details, please refer to the LVD circuit characteristics in the data sheet.

Figure 28-2 Format of user option bytes (000C1H/010C1H) (2/4)

 Address: 000C1H/010C1H^{Note}

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

- LVD setting (reset mode)

Detection voltage		Setting value of option byte								
VLVD		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				
—	Values other than the above are prohibited.									

Note During the boot swapping, 000C1H is replaced by 010C1H, so 010C1H must be set to the same value as 000C1H.

Notice Bit4 must be set to “1”.

Remark 1. For more information about LVD circuits, refer to Chapter 25, Voltage Detection Circuit.

2. The detection voltage is TYP. value. For details, please refer to the LVD circuit characteristics in the data sheet.

Figure 28-2 Format of user option bytes (000C1H/010C1H) (3/4)

 Address: 000C1H/010C1H^{Note}

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

- LVD setting (interrupt mode)

Detection voltage		Setting value of option byte						
VLVD		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		
—	Values other than the above are prohibited.							

Note During the boot swapping, 000C1H is replaced by 010C1H, so 010C1H must be set to the same value as 000C1H.

Notice Bit4 must be set to “1”.

Remark 1. For more information about LVD circuits, refer to Chapter 25, Voltage Detection Circuit.

2. The detection voltage is TYP. value. For details, please refer to the LVD circuit characteristics in the data sheet.

Figure 28-2 Format of user option bytes (000C1H/010C1H) (4/4)

 Address: 000C1H/010C1H^{Note}

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

- LVD is OFF (using an external reset input from the RESETB pin)

Detection voltage		Setting value of option byte						
V_{LVDH}		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	模式设定	
rising	falling						LVIMDS1	LVIMDS0
—	—	1	×	×	×	×	×	1
—		Values other than the above are prohibited.						

Note During the boot swapping, 000C1H is replaced by 010C1H, so 010C1H must be set to the same value as 000C1H.

Notice1. Bit4 must be set to “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, it must be set to the reset state 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/010C2H).

Remark1. ×: Ignore

2. For more information about LVD circuits, refer to Chapter 25, Voltage Detection Circuit.
3. The detection voltage is TYP. value. For details, please refer to the LVD circuit characteristics in the data sheet.

Figure 28-3 Format of user option bytes (000C2H/010C2H)

 Address: 000C2H/010C2H^{Note}

	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}
1	1	0	0	0	64MHz	64MHz
1	0	0	0	0	48MHz	48MHz
0	1	0	0	0	32MHz	32MHz
0	0	0	0	0	24MHz	24MHz
0	1	0	0	1	32MHz	16MHz
0	0	0	0	1	24MHz	12MHz
0	1	0	1	0	32MHz	8MHz
0	0	0	1	0	24MHz	6MHz
0	1	0	1	1	32MHz	4MHz
0	0	0	1	1	24MHz	3MHz
0	1	1	0	0	32MHz	2MHz
0	1	1	0	1	32MHz	1MHz
Others					Disable settings	

Note During boot swapping, 000C2H is replaced by 010C2H, so 010C2H must be set to the same value as 000C2H.

Notice1. Bits 7 to 5 must be written as "1".

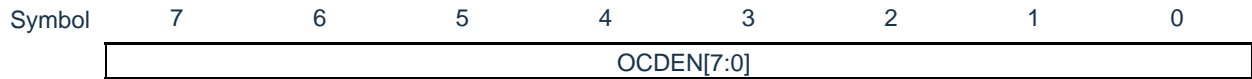
2. The operating frequency range and operating voltage range vary depending on each operating mode of the flash memory. For details, refer to AC Characteristics in the datasheet.

28.3 Format of flash data protection option bytes

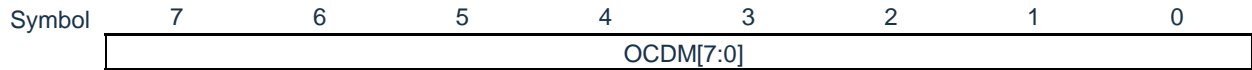
The format of the flash data protection option bytes is shown below.

Figure 28-4 Format of flash data protection option bytes (000C3H/010C3H)

Address: 000C3H/010C3H^{Note}



Address: 500004H



OCDM	OCDEN	Control of flash data protection
3C	C3	Manipulation of flash data via debugger is not allowed.
Other than 3C	C3	Allows chip full erase operations on flash data via debugger, does not allow read/write operations.
Others		Allows read/write/erase operations on flash data via debugger.

Note During the boot swapping, 000C3H is replaced by 010C3H, so 010C3H must be set to the same value as 000C3H.

Notice 50_0004H, 50_0005H address belongs to the data flash memory area, if you use this address for data storage, make sure that the value will not cause the protection option to be set incorrectly.

Address: 500005H



BTEN	Control of boot swapping function
0	Using the boot swap function, the contents of 0000H~0FFFH and 1000H~1FFFH are swapped.
1	Disables boot swap function

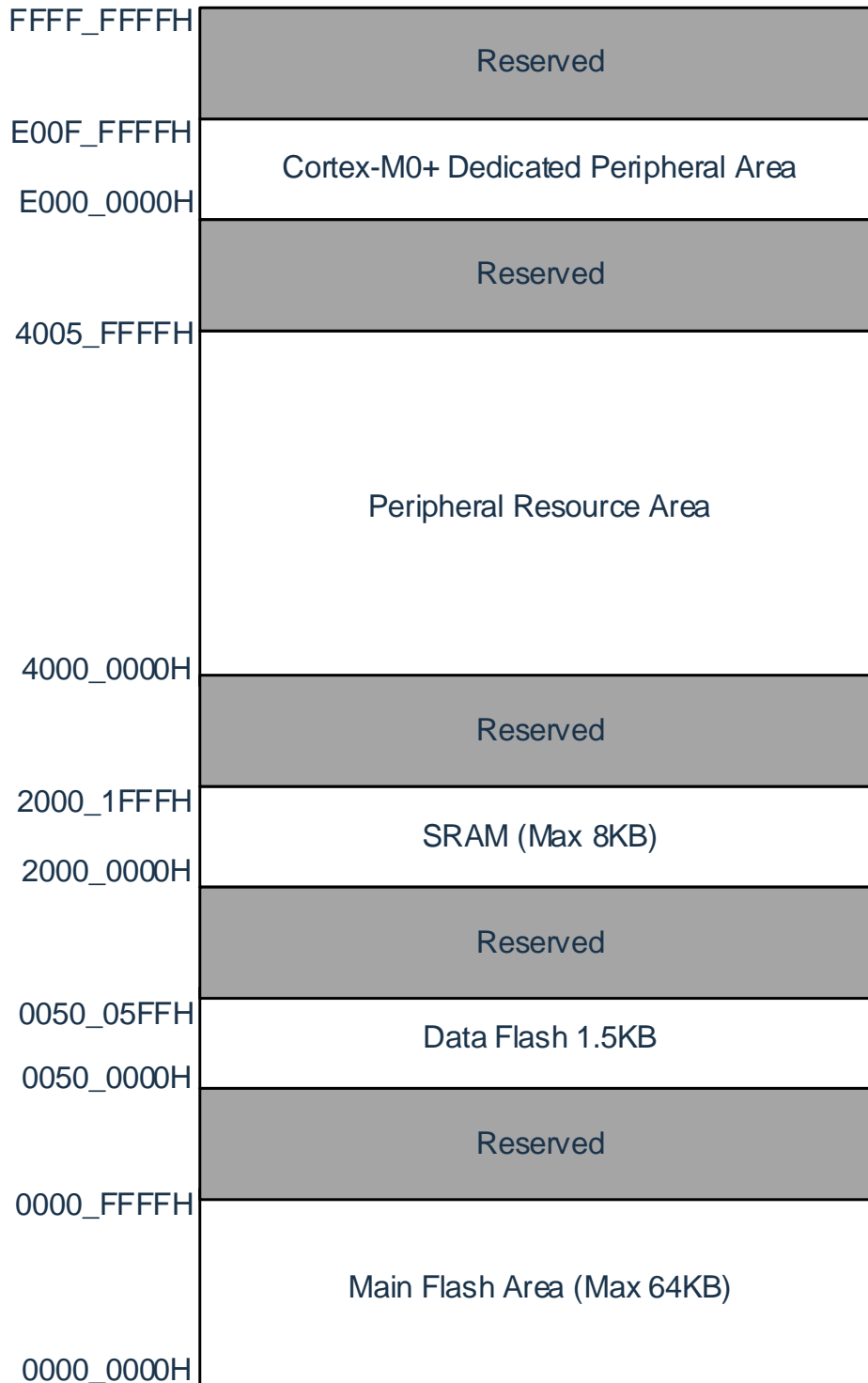
Notice 50_0004H, 50_0005H address belongs to the data flash memory area, if you use this address for data storage, make sure that the value will not cause the protection option to be set incorrectly.

Chapter 29 FLASH Control

29.1 Description of FLASH control function

This product contains a 64KByte capacity FLASH memory, divided into 128 sectors, each with a capacity of 512-byte. It can be used as program memory, data memory. This module supports erasing, programming, and reading of this memory.

29.2 Structure of FLASH memory



29.3 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 full-chip erase time control register (FLCERCNT).
- Flash sector erasure time control register (FLSERCNT).
- Flash write time control register (FLPROCNT).
- Flash mode time control register (FLNVSCNT/FLPRVCNT/FLLERVCNT).

29.3.1 Flash write protection register (FLPROT)

The Flash Protection Register is used to protect the FLASH operation control registers.

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 registers (FLOPMD1/FLOPMD2) are write-protected
0	Overwriting of FLOPMD1/FLOPMD2 is not allowed
1	Rewriting of FLOPMD1/FLOPMD2 is allowed

PRKEY[7:1]	WRP write protection
78h	Rewriting WRP is allowed
Others	Rewriting WRP is not allowed

29.3.2 FLASH operation control registers (FLOPMD1, FLOPMD2)

Flash operation control registers are 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	OPERATE
55	AA	Erase
AA	55	Write
00	00	Read
Other than the above		Set prohibited

29.3.3 Flash erase control register (FLERMD)

The 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	OPERATION
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: The chip only erases the code flash area, not the data flash area. And chip erase does not support hardware check.

29.3.4 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注	0	OVF注

OVF	FLASH erase operation complete flag
0	FLASH erase operation is not completed
1	FLASH erase operation is completed

Note: The OVF needs to be cleared by software by writing "1". 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.

29.3.5 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
load	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	FLCERCNT[9:0]									

FLCERCNT

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 $\leq 20\text{M}$, the hardware setting time can be used without setting FLCERCNT.

FLCERCNT[9:0]	Software erase time setting
Chip erase time = $(\text{CERCNT} * 2048 * T_{\text{CLK}})$, which meets the hardware requirement of $>20\text{ms}$	

29.3.6 Flash sector erase time control register (FLSERCNT)

FLSERCNT 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
load	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	FLSERCNT[9:0]									

FLSERCNT

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 $\leq 20\text{M}$, 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 >4ms	

29.3.7 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
	Load1	-	-	-	-	-	-					FLPGSCNT[8:0]				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLPROCNT	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[9:0])

Note: When the master clock is an on-chip high-speed OCO or external input clock $\leq 20M$, the time can be set in hardware without setting FLPROCNT.

FLPROCNT[8:0]	Software write time setting
Write time = $(PROCNT * 4 * T_{FCLK})$, which meets the hardware requirement of $> 24\mu s$	

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 (FLPGSCNT[8:0])

Note: When the master clock is an on-chip high-speed OCO or the external input clock is $\leq 20M$, you can set the time in hardware without setting FLPGSCNT.

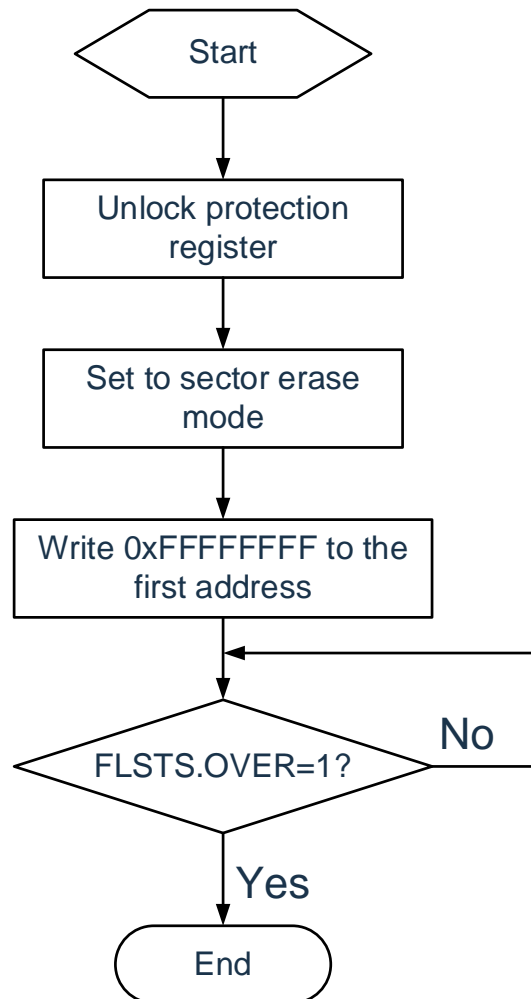
FLPGSCNT[8:0]	Write action setup time using the software setting
Write action setup time = $(PGSCNT * T_{FCLK})$, which meets the hardware requirement of $> 5\mu s$	

29.4 How to operate FLASH

29.4.1 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.



29.4.2 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.

29.4.3 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.

29.5 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.

29.6 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 = 20M, 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	Release date	Remark	Revised content
0.10	2019/9/20	-	Initial version
0.11	2020/1/16	-	Added "14.8 Operation of LIN communication".
1.00	2020/07/08	-	Corrected some mistakes
1.01	2020/11/20	-	Corrected some mistakes
1.02	2021/3/30	Section 2.3	Revised the description of P60 and P61 registers.
1.03	2022/06/2	Section 26.3 Section 10.3 Section 22.3	Deleted "Operates high-speed CRC function in RAM only". Modified the description of "LOCKUP control register". Modified the operating state of the standby function comparator.
V1.0.4	2023/12/30	Figure 25-1 Figure 25-6 Section 5.1.2 Section 7.3.4	Corrected some mistakes
V1.0.5	2024/9/5	Section 27.3.2	Modification of the formulae for calculating the voltage conversion values in section 27.3.2.
	2024/9/20	Cover page	Revised the cover page